

SpyGlass[®] Constraints Rules Reference Guide

Version N-2017.12-SP2, June 2018

SYNOPSYS[®]

Copyright Notice and Proprietary Information

©2018 Synopsys, Inc. All rights reserved. This Synopsys software and all associated documentation are proprietary to Synopsys, Inc. and may only be used pursuant to the terms and conditions of a written license agreement with Synopsys, Inc. All other use, reproduction, modification, or distribution of the Synopsys software or the associated documentation is strictly prohibited.

Destination Control Statement

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

Disclaimer

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

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at <http://www.synopsys.com/company/legal/trademarks-brands.html>.

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

Third-Party Links

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

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

Report an Error

The SpyGlass Technical Publications team welcomes your feedback and suggestions on this publication. Please provide specific feedback and, if possible, attach a snapshot. Send your feedback to spyglass_support@synopsys.com.

Contents

Preface	29
About This Book	29
Contents of This Book	30
Typographical Conventions	31
Overview	33
Prerequisites for Using the SpyGlass Constraints Solution	35
General Setup	36
Defining the Scope of Analysis	37
Exception Type Priority	37
Path Specification Priority	38
Examples of SDC Constraints Prioritization	38
Order of Reading SDC Data	40
Supported SDC Constraints	40
Supported Attributes List	74
Supported Non-SDC Constraints	75
Inferring Clock Domains When No Domain is Specified	76
Use Model for Inferring Domain	76
Specifying Domains	77
Defining Technology Library Information	84
Handling Technology Library Cell Pin Names	85
Handling Black Boxes	86
Using the SpyGlass Constraints Solution	87
Constraints Verification	88
Constraints Generation	92
Incrementally Generating SDC	92
Merging Multiple Modes for SDC Files	112
Merging SDC Files from Lower-level Blocks.....	114
Using SDC Autofix	115

Timing Exception Verification	119
Constraints Management	120
Constraints Management Rules	120
Understanding Equivalence of SDC Files	121
List of Supported SDC Constraints for Equivalence	123
Understanding the Equivalence Flow	123
Specifying the SDC File to Check SDC Equivalence	124
Specifying the Skeleton SDC File to Check SDC Equivalence	126
Checking Equivalence between the Same SDC Commands	132
 Goals	 134
 Rules in SpyGlass Constraints	 135
Block Rules	137
Block02 : Identifies combinational loops that cross block boundaries ...	138
Block05 : Reports significant combinational paths detected between blocks	140
Block06 : Reports nets connected to multiple outputs or reused internally	146
without dedicated buffering	146
Block10 : Reports combinational loops in the design	149
Block11 : Reports block outputs, inouts, or inputs that are not registered .	154
Block12 : Reports a violation if the combinational logic is lesser than the	160
specified value	160
Block13 : Reports ports driven by clocks through combinational	164
feedthrough paths	164
Clock Rules	167
Clock Consistency Rules	168
Clk_Consis05 : Identifies primary and generated clocks that are set on the	169
same port	169
Clock Generation Rules	171
Clk_Gen01 : Clock pin not driven by a clock constraint	173
Clk_Gen01a : Identifies sequential elements that are not constrained by a	174
clock	174
Clk_Gen01b : Clock driven by a constant value or hanging	183
Clk_Gen02 : Reports a constrained clock that is not used as a clock	191
Clk_Gen03 : Identifies a generated clock that is not in the fan-out of its	202
source clock	202

Clk_Gen05 : Reports synchronous clocks that have a different root clock..	208
Clk_Gen06 : Reports multiple paths that exist from the clock pin of a sequential cell to different clock sources	213
Clk_Gen07 : Identifies a clocktree net which is connected to a port without dedicated buffering	218
Clk_Gen08 : Reports clocks generated on an object that should not be a port	222
Clk_Gen09 : Reports a clock source pin that is in the fan-out of another clock, but is not generated by that clock	226
Clk_Gen10 : Identifies clocks that must be propagated in post-layout analysis.....	230
Clk_Gen13 : Identifies clock names that do not follow the naming convention.....	232
Clk_Gen14 : Reports virtual clock names that do not follow the naming convention.....	235
Clk_Gen15 : Identifies create_clock that do not have a waveform specified	239
Clk_Gen17 : Detects divided or multiplied clocks	241
Clk_Gen18 : Detects edge derived clock	243
Clk_Gen19 : Detects add or master_clock arguments in create_generated_clock.....	245
Clk_Gen20 : Reports clocks that are propagated in synthesis and/or pre-layout timing-analysis	247
Clk_Gen21 : Reports when set_propagated_clock is set on a virtual clock .	249
Clk_Gen22 : Reports set_input_delay/set_output_delay that have been specified on a clock port	252
Clk_Gen23 : Reports an incorrectly defined generated clock.....	255
Clk_Gen23a : Report for validation of create_generated_clock	269
Clk_Gen24 : Identifies clocks for which generated clock is not defined at a design object	277
Clk_Gen25 : Reports pins/ports that have multiple clocks defined.....	282
Clk_Gen26 : Identifies multiple clocks at the source pin of a generated clock that does not have a master_clock argument	285
Clk_Gen27 : Reports generated clocks that are set on non-sequential elements	288
Clk_Gen29 : Identifies instances when the same clock is converging through different combinational paths	290

Clk_Gen30	: Identifies generated clocks with an odd Divide_by factor .	294
Clk_Gen31	: Identifies the control pin of a mux in the clock path	296
Clk_Gen32	: Ensures a single path for the master clock to its generated clock	298
Clk_Gen33	: Reports unconstrained clock pins.....	301
Clk_Gen34	: Reports duplicate clock names	311
Clk_Gen35	: Reports clocks not defined on an input port or an output pin of a sequential cell.....	314
Clk_Gen36	: Reports missing generated clocks at the output of flip-flops .. 317	
Clock Latency Rules		321
Check_Timing04	: Reports clock source latency defined for clock and its source port/pin.....	322
Clk_Lat01	: Identifies clock_latency constraint that is set on an object which is not a clock	324
Clk_Lat02	: Source latency for generated clock less than or equal to source clock latency	328
Clk_Lat03	: Reports synchronous clocks that have different sum of network and source latency	331
Clk_Lat04	: Clock_latency constraint not defined or equal to zero	336
Clk_Lat04a	: Reports network latencies that have not been defined or are equal to zero for real clocks.....	337
Clk_Lat04b	: Reports source latencies that have not been defined or are equal to zero for a generated or real create clock	340
Clk_Lat05	: Identifies set_clock_latency in postlayout analysis	343
Clk_Lat06	: Reports missing set_clock_latency options.....	346
Clk_Lat07	: Identifies set_clock_latency defined with inconsistent option values	348
Clk_Lat08	: Identifies set_clock_latency constraints that have been set to a negative value.....	350
Clk_Lat09	: Identifies source or network latency that are not defined for a virtual clock	353
Clk_Lat10	: Identifies set_clock_latency specified with -early/-late arguments	355
Clk_Lat12	: Identifies clocks for which source latency is not defined....	357
Clock Transition Rules.....		359
Clk_Trans02	: Identifies set_clock_transition that have not been defined.. 360	
Clk_Trans02a	: Reports clock transition rate that have not been defined	

either using set_driving_cell or set_input_transition	362
Clk_Trans03 : Reports real clocks that have clock transition values outside the specified range	364
Clk_Trans04 : Reports clock pins that do not have a set_annotated_transition constraint defined	367
Clk_Trans05 : Reports set_annotated_transition value not within technology bounds.....	369
Clk_Trans06 : Reports instances of set_input_transition and set_driving_cell that have been used on clock ports in pre-layout	372
Clk_Trans07 : Reports clock ports that do not have set_input_transition ..	374
Clk_Trans08 : Reports set_driving_cell used on clock ports in post-layout stage	376
Clk_Trans09 : Identifies clocks that have set_clock_transition specified with incomplete options	378
Clk_Trans11 : Reports clocks that have set_clock_transition set.....	381
Clk_Trans12 : Identifies clock transition specified with inconsistent option values	383
Clk_Trans13 : Reports unusual set_input_transition option usage.....	386
Clk_Trans15 : Identifies clocks that have set_input_transition specified with incomplete arguments	388
Clk_Trans16 : Reports set_clock_transition on virtual clocks.....	391
Clk_Trans17 : Reports clock transition without proper constraint.....	394
Clock Uncertainty Rules	398
Clk_Uncert01 : Reports clock_uncertainty set on an object which is not a real or generated clock	399
Clk_Uncert02a : Reports clock uncertainty that has not been defined for a real clock.....	404
Clk_Uncert02b : Reports clock uncertainty that do have -setup or -hold...	406
Clk_Uncert02c : Reports clock uncertainty defined with zero value.....	409
Clk_Uncert03 : Reports inter-clock uncertainty that has not been defined between synchronous clocks	412
Clk_Uncert04 : Reports clock uncertainty values that are higher in the post-layout than in the pre-layout.....	417
Clk_Uncert05 : Reports set_clock_uncertainty that has been specified partially.....	420
Clk_Uncert06 : Reports set_clock_uncertainty that is set to a negative	

value.....	424
Clk_Uncert07 : Reports instances when no crossing exists between clocks for which inter-clock uncertainty is defined.....	427
Clk_Uncert08 : Reports inter-clock uncertainty that is not defined between a clock and its virtual clock.....	431
Clk_Uncert09 : Reports clock uncertainty when the hold value is greater than the setup value	434
Clk_Uncert10 : Reports inter-clock uncertainty values that are different for a pair of clocks	436
Clk_Uncert11 : Reports inter-clock uncertainty hold values that are greater than or equal to setup values	439
Dont Touch Rules.....	442
Dont_Touch02 : Reports clocks that are not defined as dont_touch_network	443
Dont_Touch03 : Reports non-clock net that is defined with dont_touch_network	445
Dont_Touch04 : Reports asynchronous reset/clear not marked with set_dont_touch or set_dont_touch_network	447
Dont_Touch05 : Reports dont_touch attribute set on a non netlist design. 450	
High Fan-out Rules.....	452
High_Fan01 : High fan-out net is not set as ideal or not set with annotated transition.....	453
High_Fan01a : Identifies high fan-out nets that are not set as ideal....	456
High_Fan02 : Reports ideal nets	459
High_Fan03 : High fan-out net identified	461
High_Fan03a : Reports identified high fan-out clock/reset/supply/scanenable nets.....	463
High_Fan03b : Reports high fan-out nets other than identified clock/reset/supply/scanenable nets.....	466
High_Fan04 : Reports set_ideal_transition that have incomplete limits defined.....	470
High_Fan05 : Reports set_ideal_latency that have incomplete limits defined.....	472
High_Fan06 : Reports when set_ideal_latency is defined for an invalid or non-ideal object.....	474
High_Fan07 : Reports when set_ideal_transition is set on an inappropriate object	477
High_Fan08 : Reports set_ideal_transition specified with inconsistent	

option values	480
High_Fan09 : Reports set_ideal_latency specified with inconsistent option values	482
High_Fan10 : Reports fan-out in the design when it is greater than the limit set by set_max_fanout	484
High_Fan11 : Reports set_max_fanout constraint greater than certain limit 486	
High_Fan12 : Reports set_max_fanout constraint that are not specified for input ports.....	489
High_Fan14 : Reports set_ideal_latency not set on an ideal object	492
High_Fan15 : set_ideal_transition not set on an ideal object.....	494
High_Fan16 : Reports start and end points of nets	496
Clock Group Rules	500
SCG01 : Reports generated relationships between clocks and set_clock_groups has been defined between them	501
SCG02 : Reports generated clocks that do not have set_clock_groups defined between them	506
SCG03 : Reports conflicting set_clock_uncertainty constraints for clocks defined as asynchronous/physically exclusive	511
SCG04 : Reports clocks, which have a set_clock_groups defined, share a harmonic relationship	516
SCG05 : Reports multiple clock definitions at the same node as being physically exclusive	519
Display Information Rules	521
Show_Case_Analysis :	
Highlights case-analysis settings	522
Show_Clock_Propagation :	
Shows clock propagation for the port/pin	526
Domain Rules	530
DomainAnalysis : Populates the domain information to be used by other constraint rules	531
DomainError : Reports clock domain errors extracted from SDC commands 533	
DomainError (contd.) : Messages and Suggested Fixes and Examples	538
Mnemonics in the Domains Spreadsheets	555
DomainInfo : Generates clock domain information from SDC commands .. 558	
Domain_SGDC_Consis : Reports inconsistency with domain extracted from SDC constraints	565

Down Streaming Rules	570
SDC_DnStrm01 : Identifies constraints requirements in mode of analysis .	
571	
SDC_DnStrm02 : Ensures set_clock_gating_style style matches with user	
style	574
SDC_DnStrm03 : Identifies physical complier constraints	577
SDC_DnStrm04 : Reports constraints that are not understood by	
implementation tools, such as Astro and Monterey	579
SDC_DnStrm04a : Reports all non-SDC and Tcl commands that are not	
understood by implementation tools	581
SDC_DnStrm05 : Reports set_max_dynamic_power that is missing or is	
set to zero	583
SDC_DnStrm06 : Reports set_max_leakage_power that is missing or is set	
to zero	586
SDC_DnStrm07 : Reports the use of set_operating_condition -	
analysis_type	589
SDC_DnStrm08 : Reports clock pins with abnormally large depth	591
I/O Rules	600
Combinational Paths Rules	601
Combo_Paths01 : Reports unconstrained combinational port-to-port paths	
602	
Combo_Paths02 : Reports inconsistent set_max_delay and set_min_delay	
commands	607
Combo_Paths03 : Reports intra-block combinational path that have a sum	
of output delay and input delay greater than clock period or sum	
of output delay and input delay greater than set_max_delay	612
Combo_Paths04 : Identifies set_max_delay or set_min_delay specified	
explicitly with only rise or fall options	620
Combo_Paths06 : Reports incomplete delay specifications for input and	
output ports	622
Input Delay Rules	627
Inp_Del01 : Input does not have an input constraint.....	629
Inp_Del01a : Reports unconstrained input/inout port by set_input_delay .	
630	
Inp_Del01b : Identifies input/inout port that is not constrained by	
set_input_delay	635
Inp_Del01c : Timing analysis not required for the input/inout ports	642
Inp_Del02 : Reports inconsistency in min/max delay value of input	
constraint	647

Inp_Del03	: Input constraint associated with wrong clock.....	652
Inp_Del03a	: Reports input constraint associated with incorrect or incomplete set of clocks.....	653
Inp_Del03b	: Reports input constraint associated with wrong or incomplete set of clocks.....	660
Inp_Del04	: Reports incompletely defined input delay constraint	666
Inp_Del05	: Reports Input constraint defined with clocks other than virtual clocks	670
Inp_Del07	: Reports correct Input constraints relative to clock period	673
Inp_Del07a	: Reports input constraints that are incorrect relative to a range of clock period.....	679
Inp_Del08	: Reports set_input_delay is set on the same input relative to multiple clocks, but -add_delay is missing.....	686
Inp_Del09	: Reports pins on a bus that do not have the same input delay	690
Inp_Del10	: Reports when level_sensitive is used in set_input_delay	695
Inp_Del11	: Reports usage of clock_fall in set_input_delay	698
Inp_Del12	: Reports use of unusual options like network_latency_included and source_latency_included in set_input_delay commands	701
Inp_Del13	: Reports if negative value is set on set_input_delay	704
Inp_Del14	: Reports set_input_delay specified with wrong/incomplete set of clocks	706
Inp_Del15	: Reports internal pins on which set_input_delay is specified ...	711
Input Transition Rules	714
Inp_Trans01	: Reports when input transition or drive or driving cell is not defined for input/inout	715
Inp_Trans01a	: Identifies unspecified input transition constraints for inputs/inouts.....	721
Inp_Trans02	: Reports input transition values that have a minimum value greater than the maximum value	725
Inp_Trans03	: Reports incomplete set_input_transition constraints	727
Inp_Trans03a	: Reports incomplete set_driving_cell constraints	730
Inp_Trans04	: Reports incorrectly specified input transition value	733
Inp_Trans05	: Reports when input transition rate is inconsistent with the associated clock transition rate	736
Inp_Trans06	: Reports when input transition rate is inconsistent with the associated clock transition rate	739
Inp_Trans07	: Reports set_load for input ports when using set_driving_cell	

	742
Inp_Trans08 : Reports any unusual usage of arguments in set_input_transition/set_driving_cell	745
Inp_Trans09 : Reports use of set_drive constraints when not allowed.	747
Input/Output Consistency Rules	749
IO_Consis01 : Reports inconsistent inter-block input/output delay constraints	750
IO_Consis02 : Reports when top-level set_max_delay or set_min_delay constraint is inconsistent with block-level constraint	754
IO_Consis04 : Reports the path delay when it exceeds the clock period ...	756
IO_Consis07 : Reports incompatible cells specified with set_driving_cell...	765
Output Delay Rules	770
Op_Del01 : Output does not have set_output_delay	772
Op_Del01a : Reports a violation when output does not have set_output_delay	773
Op_Del01b : Identifies output/inout port that is not constrained by set_output_delay	778
Op_Del01c : Reports a violation when timing analysis is not required by the output/inout port	785
Op_Del02 : Reports inconsistency in min/max delay of output constraint ..	789
Op_Del03 : Output constraint associated with wrong clock	794
Op_Del03a : Reports output constraint associated with incorrect or incomplete set of clocks	795
Op_Del03b : Identifies output constraints associated with wrong or incomplete set of clocks	802
Op_Del04 : Reports output delay constraints that are incompletely specified	808
Op_Del05 : Reports output delay constraints that are not defined relative to a virtual clock	810
Op_Del07 : Reports output constraints that are incorrect relative to the clock period	812
Op_Del07a : Reports output constraints that are incorrect relative to a range of clock period	817
Op_Del08 : Reports set_output_delay when it is set on the same output relative to multiple clocks, add_delay is missing	824
Op_Del09 : Reports pins on a bus that do not have the same output delay	

	827
Op_Del10 : Reports output delay constraints that have a level sensitive argument	833
Op_Del11 : Reports output delay constraints that have a clock_fall argument	835
Op_Del12 : Reports output delay constraints that use the network_latency_included or source_latency_included arguments	837
Op_Del13 : Reports output delay constraints that are set to a negative value	839
Op_Del14 : Reports a violation when set_output_delay is specified with wrong/incomplete set of clocks	841
Op_Del15 : Reports internal pins on which set_output_delay is specified ..	847
Load Rules.....	850
Load01 : Identifies Load constraints on output or inout ports that are not set or set to zero	851
Load02 : Load values are outside technology limits	854
Load02a : Reports load values that are outside technology limits	855
Load02b : Reports load values that are outside technology limits	861
Load03 : Reports the type of load specified to user-defined requirements..	865
Load04 : Reports the number of set_load commands when they are greater than the value specified	867
Methodology Rules	869
SDC_Methodology01 :	
Identifies design objects that may get optimized away	872
SDC_Methodology02 : Checks for user-defined forbidden constraints	879
SDC_Methodology03 : Checks for the presence of set_logic_dc, set_logic_one and set_logic_zero in the RTL phase	881
SDC_Methodology05a :	
Reports set_clock_gating_check when it is used with invalid -high or -low options	883
SDC_Methodology06 :	
Reports set_clock_gating_check that is not specified with all required options	885
SDC_Methodology07 :	
Reports set_max_time_borrow that is not within the limit based on the clock driving the latch	888

SDC_Methodology09	: Reports set_port_fanout_number with zero value. 892
SDC_Methodology10	: Reports presence of set_logic_dc, set_logic_one and set_logic_zero constraints in pre-layout and post-layout phases 894
SDC_Methodology11	: Reports output port driver pins that have a high fan-out load 896
SDC_Methodology12	: Reports presence of set_port_fanout_number in Post-layout phase 898
SDC_Methodology13	: Reports set_clock_gating_check in RTL, Pre-layout and Post-layout phases 900
SDC_Methodology16	: Reports missing options in set_data_check 902
SDC_Methodology18	: Reports incorrectly specified set_max_transition constraints..... 904
SDC_Methodology21	: Checks the usage of drive cells other than the user- specified drive cells 906
SDC_Methodology22	: Reports mismatch of units specified in the SDC and the library files 908
SDC_Methodology23	: Reports the use of the library argument in specific constraints..... 910
SDC_Methodology24	: Reports the use of set_load on nets..... 913
SDC_Methodology25	: Checks for set_max_capacitance..... 915
SDC_Methodology26	: Reports set_max_transition value is outside limits 919
SDC_Methodology27	: Reports the use of set_max_area 923
SDC_Methodology28	: Reports set_min_capacitance value when less than the limit 925
SDC_Methodology29	: Reports constraints set on flip-flop Q pin 928
SDC_Methodology30	: Reports missing set_case_analysis on a port .. 930
SDC_Methodology31	: Reports feedback loops comprising of clock paths. 933
SDC_Methodology32	: Reports missing combination options in set_load . 936
SDC_Methodology33	: Reports the missing combination options in set_drive 939
SDC_Methodology34	: Reports missing combination options in set_driving_cell 941
SDC_Methodology35	: Reports set_max_transition that is not completely specified. 943

SDC_Methodology36 : Lists the missing –rise or –fall options in the set_min_transition constraint	945
SDC_Methodology60 : Identifies set_operating_conditions that should not be redefined	947
SDC_Methodology62 : Determines missing set_max_capacitance on each port/design	949
SDC_Methodology63 : Determines the missing set_max_transition on each port/design	952
SDC_Methodology65 : Identifies multiple set_case_analysis getting propagated to different pins of the same instance.....	955
SDC_Methodology66 : Reports conflicting set_case_analysis on a design object	957
SDC_Methodology67 : Reports conflicting set_case_analysis applied on an output pin of a flip-flop or on a pin in its fan-out	961
SDC_Methodology68 : Reports all unconstrained non-clock input signals.	965
SDC_Methodology69 : Reports the use of get_nets	968
SDC_Methodology70 : Reports constraint(s) that do not have the comment option specified	970
SDC_Methodology71 : Reports a set_drive constraint that has been incorrectly specified	972
SDC_Methodology72 : Performs boundary checks for Design Rule Checks (DRC) constraints with user-specified limits	974
SDC_Methodology73 : Generates RTL-level SDC constraints from netlist-level SDC constraints	981
SDC_Methodology74 : Reports multiplexers through which the fastest clock is not selected	986
Miscellaneous Rules	989
SDC_Misc_Command01 : Reports similar constraints that have been applied to different objects	990
SDC_Misc_Power01 : Reports missing set_max_dynamic_power and set_max_leakage_power	992
SDC_Misc_Setup01 : Reports a violation when a clock is not specified for any of SDC files	995
SDC_Misc_WLM01 : Checks for designs with missing wire-load models....	

Other Rules	999
SDC_Case_Sanity01 : Reports multiple set_case_analysis set on the same path.....	1000
SDC_MergeBlocks : This rule has been deprecated.	1002
Constraints Generation	1003
SDC_GenerateIncr :	
Incrementally generate a template constraints file for a block ...	
1004	
Reporting Rules	1041
SDC_Report01 : Prints a table of generated clocks versus source clocks ...	
1042	
SDC_Report03 : Prints the list of dont touch module.....	1043
SDC_Report04 : Reports instances of the dont_use attribute set on a design object.....	1045
Cons_SDC_Report : Generates Consolidated reports for SDC constraints .	
1048	
Setup Rules	1053
Const_Sanity_Rule : Checks whether SDC or SGDC files are accessible or not	1054
DomainSanityCheck : Reads the clock_group information from SGDC files	
1057	
ParamSanityCheck01 :	
Performs checks on user-specified rule parameters.....	1061
ParamSanityCheck01a : Performs syntactic checks on user-specified rule parameters	1062
ParamSanityCheck01b : Performs a semantic check on user-specified rule parameters	1072
SDCPARSE : Reads the SDC and SGDC files for running rules in the SpyGlass Constraints solution	1076
SpyGlass Design Constraints File Structure Rules	1078
Const_Struct01 : Reports incomplete SGDC files.....	1079
Const_Struct02 : Reports incomplete blocks	1083
Const_Struct03 : Identifies set_case_analysis settings that are inconsistent with mode definitions	1085
Const_Struct04 : Detection of overwritten/duplicate constraints	1090
Const_Struct04a : Detects overwritten constraints.....	1091
Const_Struct04b : Detects duplicate constraints	1096

Const_Struct05 : Detects conflicting constraints	1099
Const_Struct07 : Reports constraints with objects directly specified	1103
Const_Struct08 : Constraints values for different corners are the same ...	
1105	
Const_Struct09 : Reports missing constraints file for best or worst corners	
1109	
Const_Struct10 : Checks for constraints applied on buffers/inverters	1112
Test Rules	1114
Test_Rules01 : Identifies a test_scan_enable port that is not defined	1115
Test_Rules02 : Identifies a test_scan_in port that is not defined	1118
Test_Rules03 : Identifies an undefined test_scan_out port	1124
Test_Rules04 : Identifies scan_configuration that is not defined for a	
constrained design unit	1131
Test_Rules05 : Identifies scan mode that is not enabled in the scan	
constraint file	1134
Test_Rules06 : Identifies scan mode that is not disabled in the functional	
mode constraint file	1138
Timing Checking Rules	1142
Check_Timing02 : Reports clock pairs that do not have expandable periods	
with respect to each other	1143
Check_Timing03 : Reports a cycle or a loop in the generated clock network	
1145	
Derate01 : Reports missing early/late arguments of the set_timing_derate	
constraint on a design object	1147
Timing Coverage Rules	1149
SDC_Coverage : Compute design coverage and report uncovered design	
objects and timing paths	1150
Timing Exception Rules	1155
xBuf	1156
XBuf01 : Detects nets for X-Buffer insertion	1157
False Path Rules	1159
False_Path01 : Identifies false path reference points that are not	
connected	1160
False_Path03 : Reports unnecessary use of -through in false path ...	1165
False_Path04 : Reports if the number of paths are more than the specified	
number	1168
False_Path04a : Identifies false path specifications that exclude a large	
number of paths	1171
False_Path04b : Identifies false path specifications that exceed a specific	

value.....	1174
False_Path07 : Reports when set_false_path is not specified between two connected asynchronous clock domains	1176
False_Path08 : Reports if false path is not specified between the two connected asynchronous clock domains	1181
False_Path09 : Reports false path that use the same clock in its -from and -to lists.....	1185
False_Path10 : Reports a violation for multiple -through options used.....	1188
False_Path11 : Reports when options -from or -to or both are not specified	1190
False_Path12 : Reports violation for the options missing in a false path ...	1192
False_Path13 : Reports a loop path at a generate clock that has false path defined.....	1194
Multi-Cycle Path Rules	1197
CheckMCP : Prints domain information between clocks used in multicycle path commands.....	1198
MCP01 : Identifies multi-cycle path reference points that are not connected	1201
MCP03 : Identifies redundant -through in the multi-cycle path	1205
MCP04 : Reports multi-cycle path that covers more than the specified number of paths	1208
MCP04a : Reports a multi-cycle path specification that covers a large number of paths	1211
MCP04b : Reports multi-cycle path specifications that exceed a specific value.....	1215
MCP05 : Reports set_multicycle_path setup, hold over or under defined....	1217
MCP08 : Reports absence of -from or -to or both options in set_multicycle_path	1222
MCP09 : Reports a violation when options are missing for a multi-cycle path	1224
MCP_B2BConsis01 : Reports multi-cycle paths spanning more than one block.....	1226
Other Timing Exception Rules	1230
Disable_Timing01 : Reports objects specified in set_disable_timing .	1231
Disable_Timing02 :	
Identifies clock paths that are blocked by set_disable_timing....	

	1233
TE_Conflict01 : Identifies overlaps between timing exceptions commands	1237
TE_Consis01 : Reports set_max_delay reference points that are not connected.....	1242
TE_Consis02 : Reports unconnected set_min_delay reference points	1245
TE_Methodology02 : Reports timing exceptions in which wildcard nodes refer to multiple reference points	1248
Abstract View Rules	1250
CONS_abstract01 : Generates an abstract view of a block for performing system level timing verification.....	1251
CONS_validate01 : Validates the timing constraints of top level block with the constraints of abstracted block	1254
Merge SDC Files	1263
SDC_ModeMerge : Merges multiple SDC files under different modes into a single SDC having equivalent timing analysis.....	1264
Constraints Management Rules	1277
ConstMgmtParamSanityCheck : Performs sanity checks on parameters .	1278
Equiv_SDC : Performs equivalence between two SDC files	1285
Equiv_SDC_Auto_Detect : Perform auto detection of equivalent ports and sequential elements between two designs	1348
Equiv_SDC_Block : Performs equivalence between the top-level and block-level design units.....	1358
Equiv_SDC_Dual_Design : Performs equivalence between two SDC files of two equivalent designs.....	1373
Equiv_SDC_Top : Performs equivalence between the top-level design unit	1387
SDC_multimode_equiv : Performs equivalence checks between a merge-mode SDC and individual modes SDC of the same design..	1392
SGDC Constraints	1399
SpyGlass Constraints Rule Parameters	1401
Overview	1401
allow_clock_on_output_port.....	1401
allow_drive.....	1402
allow_driving_cell	1402

allow_input_transition	1403
allowed_load_cells	1403
cg14_prefix	1404
cg14_suffix	1404
check_clock_group_violations	1405
chip	1406
clk_gen_module	1406
clk_gen01_generate_report	1407
clk_gen09_trav_over_sequential_arc	1407
default_max_capacitance	1408
default_max_transition	1408
default_min_capacitance	1409
default_min_transition	1410
del03_allow_setdelay	1411
drive_cells_list	1411
equiv_sdc_ambiguous_clock_file	1412
equiv_sdc_check_fanin	1416
equiv_sdc_check_fanin_fanout_for_clocks	1416
equiv_sdc_clk_prefix	1418
equiv_sdc_clk_suffix	1419
equiv_sdc_clock_precision	1420
equiv_sdc_constraint_file	1421
equiv_sdc_copy_sca_sdc	1432
equiv_sdc_design_equivalence_file	1433
equiv_sdc_design_equivalence_hier_file	1437
equiv_sdc_design_ignore_prefix	1439
equiv_sdc_enable_one_pass	1440
equiv_sdc_disambiguate_same_names_clock	1440
equiv_sdc_fast_exceptions_equivalence	1441
equiv_sdc_ignore_constant_pins_for_top_block_equivalence	1442
equiv_sdc_ignore_unequivalent_design_obj	1443
equiv_sdc_ignore_sca_for_top_block_equivalence	1443
equiv_sdc_ignore_value	1444
equiv_sdc_import_sca_from_top	1445
equiv_sdc_object_name_length	1445
equiv_sdc_report_blocked_clocks	1446
equiv_sdc_report_type	1447
equiv_sdc_script_file	1447
equiv_sdc_script_side_file	1448
equiv_sdc_show_violations	1449
equiv_sdc_set_tolerance	1449

equiv_sdc_tolerance	1450
gen_sdc_constraints_file	1451
gen_sdc_merge_seed	1451
gen_sdc_csv2sdc_tool.....	1452
gen_sdc_param_file.....	1452
gen_sdc_user_param_file	1457
gen_c2cVerify	1459
gen_sdc_phase	1459
ignore_incremental_equivalence	1460
ignore_io_if_fp.....	1461
ignore_sdc_constraints_file.....	1463
inp_delay_margin.....	1463
inp_percent	1464
inp_percent_max.....	1465
inp_percent_min	1465
library_clk_gen_naming	1466
io07_ports.....	1467
io07a_range	1467
logic_level_min	1468
logic_level_max	1469
multimode_sdc_ambiguous_clock_file	1469
num_falsepath_max	1474
num_mcpath_max.....	1474
op_delay_margin.....	1475
op_percent	1476
op_percent_max	1476
op_percent_min	1477
report_inactive_gen_clocks.....	1477
sdc_consist_ambiguous_clock_file	1478
sdc_consist_check_faninout_for_clocks	1482
sdc_consist_clock_precision	1484
sdc_consist_disambiguate_same_named_clocks	1485
sdc_consist_object_name_length.....	1486
sdc_consist_report_blocked_clocks	1487
sdc_consist_report_inactive_gen_clocks	1487
sdc_consist_report_type.....	1488
sdc_consist_show_violations	1489
sdc_consist_tolerance	1489
SDC_MergeBlocks_deglitch_cell.....	1490
SDC_MergeBlocks_deglitch_cell_clock	1491
SDC_DnStrm02_control_point.....	1491

SDC_DnStrm02_control_signal	1492
SDC_DnStrm02_pedge_logic	1493
SDC_DnStrm02_seq_cell	1493
SDC_Methodology01_allow_block_pins	1494
SDC_Methodology01_commands_list	1494
SDC_Methodology02_forbid_constraints	1495
SDC_DnStrm04_template	1496
SDC_DnStrm04a_template	1497
pt	1500
scan_insert	1503
scan_shift	1503
schematic_opt	1504
strict	1505
tc_allow_async_pins	1505
tc_at_speed_testing_mode	1506
tc_allow_design_obj	1507
tc_allow_mm_or_io	1507
tc_bus_merge	1508
tc_check_for_multiple_clock_fanin	1508
tc_checkmcp_report_sorted_clock	1509
tc_clk_compat	1510
tc_ignore_clk_outports	1510
tc_clk_register	1511
tc_clock_fanout_limit	1511
tc_combo_check	1512
tc_combopaths03_clk_multiplier	1513
tc_compare_cmd_file	1513
tc_comments_cmd_file	1514
tc_ct17_clock_trans	1515
tc_ct17_drive	1516
tc_ct17_driving_cell	1516
tc_ct17_input_trans	1517
tc_c2c_max_cycles	1518
tc_disable_caching	1518
tc_domain_mode	1519
tc_drc_spec_file	1521
tc_enable_preset_clear_arcs	1522
tc_high_fan_nets_file	1522
tc_honor_virtual_clk	1523
tc_express_check	1523
tc_enable_param_sdc_flow	1524

tc_ignore_async_ports	1525
tc_fanout_limit	1525
tc_ignore_block_path	1526
tc_ignore_if_clk_op_port	1527
tc_ignore_io_if_minmax_delay	1527
tc_ignore_iodelay_if_maxdelay	1528
tc_ignore_latch_enable	1529
tc_ignore_libcells	1529
tc_ignore_nets	1530
tc_ignore_min	1530
tc_ignore_missing_minmax_delay	1531
tc_ignored_commands	1531
tc_ignore_modes	1532
tc_ignore_scg	1532
tc_ignore_te	1533
tc_inter_block_delay	1534
tc_io_delay_bus_margin	1534
tc_io_reg	1535
tc_lvpc	1535
tc_modemerge_backref_info	1536
tc_num_load_max	1537
tc_num_fp_max	1537
tc_num_mcp_max	1538
tc_opt01_port_des	1538
tc_overconstrained_factor	1539
tc_regression_mode	1539
tc_report_backref_all	1540
tc_report_clks_start_end_points	1541
tc_report_csv_messages	1542
tc_report_unconnected_points	1542
tc_report_verbos	1543
tc_setup_hold	1544
tc_show_all_unconstrained_flops	1544
tc_show_sca_on_terminals	1545
tc_similar_clocks_tolerance_levels	1545
tc_solver_run_time	1546
tc_source_syn_clks	1547
tc_start_end_max_count	1549
tc_start_end_point_details	1550
tc_stop_parsing_ignored_commands	1550
tc_te_conflict_fast_run	1551

tc_tech.....	1552
tc_violate_ports.....	1552
tc_virtual_clock_prefixes	1553
tc_virtual_clock_suffixes	1554
tc_waive.....	1555
tc_waive_const_struct05	1555
tc_xbuf_ignore_clks	1556
tcdecompile	1557
verbose	1558
Other Rule Parameters	1559

SpyGlass Constraints Reports 1560

Timing Constraints Information Reports	1562
tc_report_disable_timing Report	1563
tc_unparsed_commands Report	1566
tc_block11_info Report	1568
Timing Constraints Domain Reports	1571
Domain Matrix Report.....	1572
Domain Cyclic Errors Report	1575
Missing Domains Report.....	1579
Timing Analysis Consolidated Report	1581
Timing Constraints Coverage Report	1585
Timing Exception Reports.....	1598
False_Path01 Report	1599
MCP01 Report	1602
TE_Consis01 Report	1606
TE_Consis02 Report	1610
Constraints Equivalence Reports	1613
Clock-Mapping Report.....	1614
EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT	1617
Equivalence Report	1620
Miscellaneous Reports.....	1627
tc_dont_touch_info Report	1630
tc_dont_use_info Report	1633
clk_domain_false_path Report.....	1636
sync_clock_uncertainty Report	1639
tc_clk_gen01a_info Report.....	1642
tc_clk_gen01b_info Report.....	1645
tc_clock_info Report.....	1648

tc_latency_info Report	1652
SDC_Methodology31 Report.....	1655
mcp_domain_<integer> Report.....	1657

Appendix: SDC Constraints.....	1661
create_clock.....	1663
create_generated_clock.....	1665
set.....	1668
set_annotated_transition.....	1669
set_case_analysis	1671
set_clock_gating_check.....	1672
set_clock_groups.....	1674
set_clock_latency	1676
set_clock_sense.....	1678
Stopping Clock Propagation.....	1680
set_sense.....	1682
set_clock_transition.....	1684
set_clock_uncertainty.....	1686
set_data_check.....	1688
set_disable_timing.....	1690
set_dont_touch.....	1691
set_drive.....	1692
set_driving_cell	1693
set_false_path	1696
set_fanout_load.....	1699
set_ideal_latency.....	1700
set_ideal_net	1702
set_ideal_network	1703
set_ideal_transition	1704
set_input_delay	1706
set_input_transition	1708
set_load.....	1710
set_logic_dc.....	1712
set_logic_one.....	1713
set_logic_zero.....	1714

set_max_area	1715
set_max_capacitance	1716
set_max_fanout	1718
set_max_delay/set_min_delay	1719
set_max_delay	1719
set_min_delay	1719
set_max_dynamic_power	1721
set_max_leakage_power	1722
set_max_time_borrow	1723
set_max_transition	1724
set_min_capacitance	1726
set_min_transition	1727
set_multicycle_path	1729
set_operating_conditions	1732
set_output_delay	1734
set_port_fanout_number	1736
set_propagated_clock	1737
set_resistance	1738
set_timing_derate	1739
set_wire_load_model	1741
set_wire_load_selection_group	1743

Preface

About This Book

The SpyGlass® Constraints Rules Reference Guide describes the SpyGlass rules that check Synopsys Design Constraints (SDC) and HDL designs for proper constraints specification.

Contents of This Book

The SpyGlass Constraints Rules Reference Guide has the following sections:

Section	Description
<i>Overview</i>	Provides an introduction to the SpyGlass Constraints solution
<i>Prerequisites for Using the SpyGlass Constraints Solution</i>	Provides licensing information
<i>General Setup</i>	Describes common concepts related to setup
<i>Using the SpyGlass Constraints Solution</i>	Describes Constraints Verification, Generation, and Timing Exception Verification.
<i>Goals</i>	List of GuideWare goals
<i>Rules in SpyGlass Constraints</i>	Describes SpyGlass Constraints rules
<i>SGDC Constraints</i>	Links of all SGDC constraints that are applicable to the SpyGlass Constraints solution.
<i>SpyGlass Constraints Rule Parameters</i>	Describes parameters used to customize rule behavior
<i>SpyGlass Constraints Reports</i>	Provides description of reports
<i>Appendix: SDC Constraints</i>	Provides description and syntax of SDC Constraints

Typographical Conventions

This document uses the following typographical conventions:

To indicate	Convention Used
Program code	OUT <= IN;
Object names	OUT
Variables representing objects names	<sig-name>
Message	Active low signal name '<sig-name>' must end with _X.
Message location	OUT <= IN;
Reworked example with message removed	OUT_X <= IN;
Important Information	NOTE: This rule...

The following table describes the syntax used in this document:

Syntax	Description
[] (Square brackets)	An optional entry
{ } (Curly braces)	An entry that can be specified once or multiple times
(Vertical bar)	A list of choices out of which you can choose one
. . . (Horizontal ellipsis)	Other options that you can specify

Overview

The purpose of the SpyGlass Constraints product is to help you validate the completeness and correctness of timing (SDC) constraints with respect to the corresponding RTL, and to help you build or augment those constraints if you do not yet have SDC or if the SDC is known to be incomplete.

SDC generation is particularly useful when you need to generate timing constraints for a design that you did not, perhaps, create and/or you may want to update existing SDC constraints to reflect changes in the RTL. You can learn more about this in the [Constraints Generation](#) section.

Just as linting RTL is a recommended best practice for finding and correcting problems prior to simulation or synthesis (you will likely find the problems in those steps, but probably after wasting significant time in tracking them down), similarly linting SDC constraints prior to synthesis or STA can help you quickly isolate and correct problem, avoiding unnecessary delays in timing closure constraints before they emerge in lengthy implementation analysis. This is covered in the [Constraints Verification](#) section.

Related to creating constraints is the need to merge constraints files from multiple modes. Eventually, you need to implement against a fixed set of constraints; while implementation tools support this capability, if you want fine-grained control over the merged constraints, you may find this capability useful. You can understand more under the [Merging Multiple Modes for SDC Files](#) and [Merging SDC Files from Lower-level Blocks](#) section.

SDC's typically evolve through the design flow. It is often valuable to be able to check exactly what has changed between two revisions of an SDC, just as it is valuable to be able to check differences between two revisions of an RTL using logic equivalence checking. You can learn more about this in the [Constraints Management](#) section.

Prerequisites for Using the SpyGlass Constraints Solution

You must be able to read the design successfully into SpyGlass. Refer to the *Setting up a Design* section of the *SpyGlass Explorer User Guide*.

You should ensure the analyzed design contains a minimum of unintended black boxes. You should also ensure that, if the design contains instantiated technology library cells, you perform the analysis linking to the corresponding technology library (.lib) files. See the [General Setup](#) section for details on linking to technology library files.

If you are verifying constraints, you should also have access to those SDC files.

Licensing Information

The SpyGlass Constraints solution requires standard SpyGlass to be installed and licensed. For running the rules of SpyGlass Constraints, the `constraints` SpyGlass Constraints license feature is also required.

The SpyGlass Constraints solution also offers some supplementary features with the following SpyGlass Constraints licenses:

- **For constraints management capabilities:** `constraints_mgmt`
- **For SDC merging:** `constraints_SDCmerge`
- **For SDC generation:** `constraints_SDCgen`
- **For SDC promotion:** `constraints_soc_integration`

General Setup

Setup for SpyGlass Constraints depends on whether you intend to generate, verify or manage constraints. This section describes only concepts that are common to all three of these objectives. Setup topics unique to each objective are covered under the appropriate sections of [Using the SpyGlass Constraints Solution](#).

This section contains the following topics:

- [Defining the Scope of Analysis](#)
- [Order of Reading SDC Data](#)
- [Inferring Clock Domains When No Domain is Specified](#)
- [Defining Technology Library Information](#)
- [Handling Technology Library Cell Pin Names](#)
- [Handling Black Boxes](#)

Defining the Scope of Analysis

The SpyGlass Constraints solution reads one or more SDC file specified in the SGDC file and verifies information on constraints against the actual HDL designs.

Unlike Synopsys tools that process information on constraints in a dynamic manner, the SpyGlass Constraints solution processes the SDC file information in a static manner. Therefore, SpyGlass works on the last instance of the constraint on an object even though the object is constrained by multiple instances of the same constraint.

The SDC constraints in the SDC files are processed based on:

- *Exception Type Priority*
- *Path Specification Priority*

Also, refer to *Examples of SDC Constraints Prioritization*.

Exception Type Priority

The following pairs of timing exception types are not considered to be in conflict. Therefore, both settings can be valid for a path:

- Two *set_false_path* settings
- *set_min_delay* and *set_max_delay* settings
- *set_multicycle_path* -setup and -hold settings

When a particular path has conflicting exceptions, the timing exception types have the following order of priority, from highest to lowest:

1. *set_false_path*
2. *set_max_delay* and *set_min_delay*
3. *set_multicycle_path*

The *set_min_delay*, *set_max_delay* and *set_multicycle_path* constraints are overwritten on the basis of their order in the SDC file, if the *set_false_path* constraint is not specified. If the *set_false_path* constraint is specified, it would completely overwrite all other exceptions.

For example, suppose a path has false path, minimum and maximum delay, and multicycle path specifications. Since the path has been declared

false by the `set_false_path` constraint, the minimum and maximum delay specifications are ignored because a false path specification has higher priority. Similarly, the `set_multicycle_path` is ignored.

Path Specification Priority

For path specification priorities, the SpyGlass Constraints solution gives higher priority to the more specific command.

Suppose, as shown in the following SDC snippet, you have a applied `set_min_delay` to all paths starting from CLK1 and you have also specified a minimum delay for all paths starting from CLK1 and ending at CLK3.

```
set_min_delay 10 -from [get_clocks CLK1]
set_min_delay 18 -from [get_clocks CLK1] -to [get_clocks
CLK3]
```

In this case, the remaining paths starting from CLK1 are still controlled by the first command.

The order of priority from highest to lowest for the -from/-to path specification is as follows:

1. -from pin
2. -to pin, -rise_to pin, -fall_to pin
3. -through, -rise_through, -fall_through
4. -from clock, -rise_from clock, -fall_from clock
5. -to clock, -rise_to clock, -fall_to clock

Examples of SDC Constraints Prioritization

The examples in this section illustrate the prioritization of SDC constraints by the SpyGlass Constraints solution.

Example 1

In the following SDC snippet, `set_false_path` would overwrite `set_multicycle_path` and `set_max_delay` because `set_false_path` holds a highest priority.

```
set_false_path -to out
set_multicycle_path 3 -to out
set_max_delay 3 -to out
```

Example 2

In the following SDC snippet, *set_max_delay* would overwrite *set_multicycle_path*.

```
set_multicycle_path 3 -to out
set_max_delay 3 -to out
```

Example 3

In the following SDC snippet, *set_max_delay* would not overwrite the *set_multicycle_path* exception, but both would be treated as valid exceptions because of the order in which they are specified in the SDC file.

```
set_max_delay 3 -to out
set_multicycle_path 3 -to out
```

Order of Reading SDC Data

This topic describes the following:

- [Supported SDC Constraints](#)
- [Supported Non-SDC Constraints](#)

Supported SDC Constraints

SDC is a format used to specify design intent, including the timing and the constraints for a design. SDC is based on the Tcl language and is used by Synopsys tools. The SpyGlass Constraints solution supports SDC 1.1, SDC 1.2, SDC 1.3, SDC 1.4, SDC 1.5, SDC 1.6, SDC 1.7, SDC 1.8, SDC 1.9, and SDC 2.0 version files.

The support should be seen in the context of the rules in the SpyGlass Constraints solution. If the rules require a few options only for a particular constraint, then only those options are checked and populated.

By default, the SpyGlass Constraints solution performs checks based on the SDC 2.0 version. You can also set for a different SDC version by specifying the [set](#) command as in the following example:

```
set sdc_version 1.3
```

NOTE: *The DC shell scripts and commands are not supported. Only the Tcl mode SDC files are supported.*

The following table summarizes the SDC support available in the SpyGlass Constraints solution.

TABLE 1 Supported SDC Commands

Command	Supported	Not supported	Only Parsed
all_clocks	•		
all_inputs	•		
[-level_sensitive]	•		
[-edge_triggered]	•		
[-clock clock_name]	•		
all_outputs	•		
[-level_sensitive]	•		
[-edge_triggered]	•		
[-clock clock_name]	•		
append_to_collection	•		
[var_name]	•		
[object_spec]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-unique]	•		
<p>Note: For the command, <append_to_collection var_name \$object_spec>, the collection referred by 'var_name' is appended by the contents of collection 'object_spec'.</p> <p>There are limitations for two cases when using these commands:</p> <ol style="list-style-type: none"> 1. var_name specified is not defined 2. var_name specified points to collection which is empty <p>In these cases, a temporary collection is created to act as the base_collection to be appended into. The actual collection 'var_name' is not changed, however in PT it does. Refer to the long help of message SDC_360 for more details.</p>			
compare_collections	•		
[collection1]	•		
[collection2]	•		
[-order_dependent]	•		
copy_collection	•		
[-collection1]	•		
create_clock	•		
-period <i>period_value</i>	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-name <i>clock_name</i>]	•		
[-waveform <i>edge_list</i>]	•		
[-add]	•		
[<i>port_pin_list</i>]	•		
[-comment]	•		
create_generated_clock	•		
-source <i>master_pin</i>	•		
[-name <i>clock_name</i>]	•		
[-edges <i>edge_list</i>]	•		
[-divide_by <i>factor</i>]	•		
[-multiply_by <i>factor</i>]	•		
[-combinational]	•		
[-duty_cycle <i>factor</i>]	•		
[-invert]	•		
[-edge_shift <i>shift_list</i>]	•		
[-add]	•		
[-master_clock <i>clock</i>]	•		
[<i>port_pin_list</i>]	•		
[-comment]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
create_lcd_operating_condition	•		
-library <i>library_name</i>	•		
-op_worst <i>worst_case_oprtng_cond_name</i>			•
-mult_worst <i>worst_case_multiplier</i>			•
-op_nominal <i>nominal_operating_cond_name</i>			•
-mult_nominal <i>nominal_multiplier</i>			•
-op_best <i>best_case_operating_cond_name</i>			•
-mult_best <i>best_case_multiplier</i>			•
<i>name</i>			•
create_voltage_area			•
-name <i>voltage_area_name</i>			•
cell_list <i>list1</i>			•
[-coordinate]			•
[-guard_band_x]			•
[-guard_band_y]			•
current_design	•		
[instance <i>string</i>]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
current_instance	•		
[<i>instance</i>]	•		
define_proc_attributes	•		
<i>proc_name</i>	•		
[-define_args <i>arg_defs</i>]	•		
[-info <i>info_text</i>]			•
[-command_group <i>group_name</i>]			•
[-hide_body]			•
[-hidden]			•
[-dont_abbrev]			•
[-permanent]			•
derive_clocks	•		
-period <i>value</i>	•		
[-waveform <i>list</i>]	•		
filter/filter_collection	•		
[-regexp]	•		
[-nocase]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-quiet]	•		
Note: The -quiet option is supported when the pt parameter is set to no.			
<i>expression object_list</i> See Supported Attributes List for the list of attributes.	•		
get_attribute	•		
<i>expression object_list</i> See Supported Attributes List for the list of attributes.	•		
[-class <i>class_name</i>]	•		
[<i>attribute_name</i>]	•		
[is_sequential]	•		
[-bus]			•
[-quiet]			•
Note: The -class argument of the get_attribute command supports only a limited number of attributes. See Supported Attributes List for the list of attributes.			
get_cells	•		
[-h -hierarchical]	•		
[-hsc <i>separator</i>]	•		
<i>patterns</i>	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-nocase]	•		
[-filter expression]	•		
Note: All attributes of the <i>filter/filter_collection</i> commands are supported.			
-of_objects objects	•		
[-exact]	•		
[-regexp]	•		
[-quiet]	•		
get_clocks	•		
<i>patterns</i>	•		
[-regexp]	•		
[-nocase]	•		
[-filter expression]	•		
Note: All attributes of the <i>filter/filter_collection</i> commands are supported.			
[-quiet]			•
get_lib_cells	•		
<i>patterns</i>	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-filter expression]	•		
Note: All attributes of the <i>filter/filter_collection</i> commands are supported.			
[-regexp]	•		
[-nocase]	•		
[-exact]		•	
[-quiet]		•	
[-hsc separator]		•	
-of_objects objects		•	
get_lib_pins	•		
<i>patterns</i>	•		
[-nocase]	•		
[-regexp]	•		
[-filter expression]	•		
[-quiet]			•
[-exact]			•
-of_objects objects			•
[-hsc separator]			•

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
get_libs	•		
<i>patterns</i>	•		
[-regexp]	•		
[-nocase]	•		
[-filter expression]	•		
Note: All attributes of the <i>filter/filter_collection</i> commands are supported.			
[-quiet]			•
[-exact]			•
-of_objects objects			•
get_nets	•		
<i>patterns</i>	•		
[-filter expression]	•		
Note: All attributes of the <i>filter/filter_collection</i> commands are supported.			
[-nocase]	•		
[-regexp]	•		
[-h -hierarchical]			•
[-hsc separator]			•

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-all]			•
[-quiet]	•		
[-exact]	•		
[-top_net_of_hierarchical_group]	•		
[-segments]	•		
[-boundary_type]	•		
-of_objects objects			•
get_pins	•		
patterns	•		
[-h -hierarchical]	•		
[-hsc separator]	•		
[-filter expression]	•		
Note: All attributes of the <i>filter/filter_collection</i> commands are supported.			
[-nocase]	•		
[-regexp]	•		
-of_objects objects	•		
[-leaf]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-quiet]	•		
[-exact]	•		
get_ports	•		
<i>patterns</i>	•		
[-filter <i>expression</i>]	•		
Note: All attributes of the <i>filter/filter_collection</i> commands are supported.			
[-nocase]	•		
[-regexp]	•		
-of_objects <i>objects</i>	•		
[-quiet]			•
[-exact]			•
index_collection	•		
[base_collection]	•		
[index Int [<i>\$par</i> >=0]]	•		
parse_proc_arguments	•		
-args <i>arg_list result_array</i>	•		
remove_clock_groups	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
-physically_exclusive	•		
-logically_exclusive	•		
-asynchronous	•		
[-name (<i>name_list</i>) -all]	•		
reset_path	•		
[-setup]	•		
[-hold]	•		
[-rise]	•		
[-fall]	•		
[-from <i>from_list</i> -rise_from <i>rise_from_list</i> -fall_from <i>fall_from_list</i>]	•		
[-through <i>through_list</i>]	•		
[-rise_through <i>rise_through_list</i>]	•		
[-fall_through <i>fall_through_list</i>]	•		
[-to <i>to_list</i> -rise_to <i>rise_to_list</i> - fall_to <i>fall_to_list</i>]	•		
set_active_clocks		•	
<i>active_clock_list</i>		•	
[all_clocks]		•	

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
set_annotated_delay			•
[-net -cell]			•
[-load_delay <i>load_value</i>]			•
[-rise]			•
[-fall]			•
[-max]			•
[-min]			•
[-worst]			•
-from <i>from_list</i>			•
-to <i>to_list</i>			•
<i>delay_value</i>			•
set_case_analysis	•		
<i>value_sca</i>	•		
<i>port_pin_list</i>	•		
set_clock_gating_check	•		
[-setup <i>setup_value</i>]	•		
[-hold <i>hold_value</i>]	•		
[-rise]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-fall]	•		
[-high]	•		
[-low]	•		
[<i>object_list</i>]	•		
set_clock_groups	•		
[-physically_exclusive -locally_exclusive -asynchronous]	•		
[-allow_paths]	•		
[-name name]	•		
[-group clock_list]	•		
[-comment]	•		
set_clock_latency	•		
<i>delay</i>	•		
<i>object_list</i>	•		
[-rise]	•		
[-fall]	•		
[-min]	•		
[-max]	•		
[-source]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-late]	•		
[-early]	•		
[-clock <i>clock</i>]			•
set_clock_sense	•		
[-stop_propagation -positive -negative -pulse <i>pulse_type</i>]	•		
[-clocks <i>clock_list</i>]	•		
<i>object_list</i>	•		
set_sense	•		
NOTE: Support for the <i>set_sense</i> constraint is provided for all Equivalence rules and most of the other constraints rules, except <i>SDC_GenerateIncr</i> rule.			
[-stop_propagation -positive -negative -pulse <i>pulse_type</i>]	•		
[-type clock data]			
[-clocks <i>clock_list</i>]	•		
<i>object_list</i>	•		
set_clock_transition	•		
[-rise]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-fall]	•		
[-min]	•		
[-max]	•		
<i>transition</i>	•		
<i>clock_list</i>	•		
set_clock_uncertainty	•		
[-from <i>from_clock</i>]	•		
[-to <i>to_clock</i>]	•		
[-rise]	•		
[-rise_from <i>from_clock</i>]	•		
[-rise_to <i>to_clock</i>]	•		
[-fall]	•		
[-fall_from <i>from_clock</i>]	•		
[-fall_to <i>to_clock</i>]	•		
[-setup]	•		
[-hold]	•		
[-from_edge <i>edge_list</i>]	•		
[-to_edge <i>edge_list</i>]	•		
<i>uncertainty</i>	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[<i>object_list</i>]	•		
set_data_check	•		
[-from <i>from_object</i>]	•		
[-to <i>to_object</i>]	•		
[-rise_from <i>from_object</i>]	•		
[-fall_from <i>from_object</i>]	•		
[-rise_to <i>to_object</i>]	•		
[-fall_to <i>to_object</i>]	•		
[-setup]	•		
[-hold]	•		
[-clock <i>clock_object</i>]	•		
<i>check_value</i>	•		
set_disable_timing	•		
[-from <i>from_pin_name</i>]	•		
[-to <i>to_pin_name</i>]	•		
[<i>object_list</i>]	•		
[-restore]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
set_drive	•		
[-rise -fall]	•		
[-min]	•		
[-max]	•		
<i>resistance</i>	•		
<i>port_list</i>	•		
set_driving_cell	•		
[-lib_cell <i>lib_cell_name</i>]	•		
[-rise]	•		
[-fall]	•		
[-max]	•		
[-min]	•		
[-library <i>lib_name</i>]	•		
[-pin <i>pin_name</i>]	•		
[-from_pin <i>from_pin_name</i>]	•		
[-multiply_by <i>multiply_factor</i>]	•		
[-input_transition_rise <i>rise_time</i>]	•		
[-input_transition_fall <i>fall_time</i>]	•		
[-clock <i>clock_name</i>]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-clock_fall -clock <i>clock_name</i>]	•		
<i>port_list</i>	•		
[-dont_scale]			•
[-no_design_rule]			•
setenv	•		
<i>variable_name new_value</i>	•		
set_false_path	•		
[-setup]	•		
[-hold]	•		
[-rise]	•		
[-fall]	•		
[-from <i>from_list</i>]	•		
[-to <i>to_list</i>]	•		
[-through <i>through_list</i>]	•		
[-rise_from <i>from_list</i>]	•		
[-fall_from <i>from_list</i>]	•		
[-rise_to <i>to_list</i>]	•		
[-fall_to <i>to_list</i>]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-rise_through <i>through_list</i>]	•		
[-fall_through <i>through_list</i>]	•		
[-reset_path]	•		
[-comment]	•		
set_fanout_load	•		
<i>value port_list</i>	•		
set_hierarchy_separator	•		
[<i>hchar separator</i>]	•		
set_input_delay	•		
[-clock <i>clock_name</i>]	•		
[-clock_fall]	•		
[-level_sensitive]	•		
[-rise]	•		
[-fall]	•		
[-max]	•		
[-min]	•		
[-add_delay]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-network_latency_included]	•		
[-source_latency_included]	•		
[-reference_pin <i>pin_name</i>]	•		
<i>delay_value</i>	•		
<i>port_pin_list</i>	•		
set_input_transition	•		
[-rise]	•		
[-fall]	•		
[-min]	•		
[-max]	•		
[-clock <i>clock_name</i>]	•		
[-clock_fall]	•		
<i>transition port_list</i>	•		
set_level_shifter_strategy			•
-rule			•
set_level_shifter_threshold			•
-voltage <i>volt</i>			•

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
<i>-percent diff</i>			•
set_load	•		
[<i>-min</i>]	•		
[<i>-max</i>]	•		
[<i>-fall</i>]	•		
[<i>-rise</i>]	•		
[<i>-subtract_pin_load</i>]	•		
[<i>-pin_load</i>]	•		
[<i>-wire_load</i>]	•		
<i>value</i>	•		
<i>objects</i>	•		
set_logic_dc	•		
<i>port_list</i>	•		
set_logic_one	•		
<i>port_list</i>	•		
set_logic_zero	•		
<i>port_list</i>	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
set_max_area	•		
<i>area_value</i>	•		
[<i>-ignore_tns</i>]	•		
set_max_capacitance	•		
[<i>capacitance_value</i>]	•		
[<i>object_list</i>]	•		
[<i>-clock_path</i>]	•		
[<i>-data_path</i>]	•		
[<i>-rise</i>]	•		
[<i>-fall</i>]	•		
set_max_delay	•		
[<i>-rise</i>]	•		
[<i>-fall</i>]	•		
[<i>-from from_list</i>]	•		
[<i>-to to_list</i>]	•		
[<i>-through through_list</i>]	•		
[<i>-rise_from from_list</i>]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-fall_from <i>from_list</i>]	•		
[-rise_to <i>to_list</i>]	•		
[-fall_to <i>to_list</i>]	•		
[-rise_through <i>through_list</i>]	•		
[-fall_through <i>through_list</i>]	•		
<i>delay_value</i>	•		
[-group_path <i>group_name</i>]	•		
[-reset_path]	•		
[-comment]	•		
set_max_dynamic_power	•		
<i>leakage_power</i>	•		
[unit GW MW KW W mW uW nW pW fW aW]	•		
[-effort low high]	•		
set_max_fanout	•		
<i>fanout_value</i>	•		
<i>object_list</i>	•		
set_max_leakage_power	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
<code>leakage_power</code>	•		
<code>[GW MW KW W mW uW nW pW fW aW]</code>	•		
<code>[-effort low high]</code>	•		
<code>[-use_sd_info]</code>	•		
<code>set_max_time_borrow</code>	•		
<code>delay_value</code>	•		
<code>object_list</code>	•		
<code>set_max_transition</code>	•		
<code>transition_value</code>	•		
<code>object_list</code>	•		
<code>[-clock_path]</code>	•		
<code>[-data_path]</code>	•		
<code>[-rise]</code>	•		
<code>[-fall]</code>	•		
<code>set_min_capacitance</code>	•		
<code>capacitance_value</code>	•		
<code>object_list</code>	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
set_min_delay	•		
[-rise]	•		
[-fall]	•		
[-from <i>from_list</i>]	•		
[-to <i>to_list</i>]	•		
[-through <i>through_list</i>]	•		
[-rise_from <i>from_list</i>]	•		
[-fall_from <i>from_list</i>]	•		
[-rise_to <i>to_list</i>]	•		
[-fall_to <i>to_list</i>]	•		
[-rise_through <i>through_list</i>]	•		
[-fall_through <i>through_list</i>]	•		
<i>delay_value</i>	•		
[-reset_path]	•		
[-group_path <i>group_name</i>]	•		
[-comment]	•		
set_min_porosity			•
<i>porosity_value</i>			•
[<i>object_list</i>]			•

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
set_min_transition	•		
<i>transition_value</i>	•		
<i>object_list</i>	•		
[-clock_path]	•		
[-data_path]	•		
[-rise]	•		
[-fall]	•		
set_multicycle_path	•		
[-setup]	•		
[-hold]	•		
[-rise]	•		
[-fall]	•		
[-start]	•		
[-end]	•		
[-from <i>from_list</i>]	•		
[-to <i>to_list</i>]	•		
[-through <i>through_list</i>]	•		
[-rise_from <i>from_list</i>]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-fall_from <i>from_list</i>]	•		
[-rise_to <i>to_list</i>]	•		
[-fall_to <i>to_list</i>]	•		
[-rise_through <i>through_list</i>]	•		
[-fall_through <i>through_list</i>]	•		
<i>path_multiplier</i>	•		
[-reset_path]	•		
[-comment]	•		
set_operating_conditions	•		
[-library <i>lib_name</i>]	•		
[-analysis_type <i>type</i>]	•		
[-max <i>max_condition</i>]	•		
[-min <i>min_condition</i>]	•		
[-max_library <i>max_lib</i>]	•		
[-min_library <i>min_lib</i>]	•		
[-object_list <i>object_list</i>]	•		
[<i>condition</i>]	•		
[-max_phy <i>max_condition</i>]			•
[-min_phy <i>min_condition</i>]			•

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
set_output_delay	•		
[-clock <i>clock_name</i>]	•		
[-clock_fall]	•		
[-level_sensitive]	•		
[-rise]	•		
[-fall]	•		
[-max]	•		
[-min]	•		
[-add_delay]	•		
[-network_latency_included]	•		
[-source_latency_included]	•		
[-reference_pin <i>pin_name</i>]	•		
<i>delay_value</i>	•		
<i>port_pin_list</i>	•		
[-group_path <i>group_name</i>]			•
set_port_fanout_number	•		
<i>fanout_number</i>	•		
<i>port_list</i>	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-min]	•		
[-max]	•		
set_propagated_clock	•		
<i>object_list</i>	•		
set_resistance	•		
[-min]	•		
[-max]	•		
<i>value</i>	•		
<i>net_list</i>	•		
set_timing_derate	•		
[-min]	•		
[-max]	•		
[-early]	•		
[-late]	•		
[-clock]	•		
[-data]	•		
[-net_delay]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[-cell_delay]	•		
[-cell_check]	•		
[-rise]			•
[-fall]			•
<i>derate_value</i>	•		
<i>object_list</i>	•		
set_units	•		
[-time]	•		
[-capacitance]	•		
[-resistance]	•		
[-voltage]	•		
[-current]	•		
[-power]	•		
<i>characterscale_value f p n u m k M</i>	•		
set_voltage			•
[-min min_case_value]			•
[-object_list list_of_supply_nets]			•
<i>max_case_voltage</i>			•

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
set_wire_load_min_block_size			•
<i>size</i>			•
set_wire_load_mode	•		
<i>mode_name</i>	•		
set_wire_load_model	•		
-name <i>model_name</i>	•		
[-library <i>lib_name</i>]	•		
[-min]	•		
[-max]	•		
[<i>object_list</i>]	•		
[-cluster <i>cluster_name</i>]	•		
set_wire_load_selection_group	•		
[-library <i>lib_name</i>]	•		
[-min]	•		
[-max]	•		
[-group_name]	•		

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported	Only Parsed
[<i>object_list</i>]	•		
[-cluster <i>cluster_name</i>]	•		

Supported Attributes List

The following attributes are supported:

expression object_list

port_direction
pin_direction
direction
is_hierarchical
full_name
name
ref_name
is_mux_select_pin
is_port
is_clock_gating_cell
clock_gating_integrated_cell
is_sequential
is_combinational
lib_pin_name
object_class
period
clock
is_clock_pin
is_data_pin
is_mux
is_clock_used_as_clock
base_name
rise_capacitance
fall_capacitance
capacitance
arrival
is_generated
generated_clocks

expression object_list

```
is_active
master_pin
master_clock
propagated_clock
clock_network_pins
waveform
ref_lib_pin_name
ref_lib_cell_name
is_memory_cell (for netlist designs only)
```

If `pt` is set to `no`, attributes are read in a case insensitive manner. However, when `pt` is set to `yes`, attributes are read in a case sensitive manner.

Supported Non-SDC Constraints

The list of all ignored constraints are written to a file named `tc_unparsed_command`.

The corresponding report file, `tc_unparsed_commands.rpt`, is created in the current working directory and can also be accessed from the *Report* Menu in Console GUI. Refer to the [tc_unparsed_commands Report](#) section for more details.

Inferring Clock Domains When No Domain is Specified

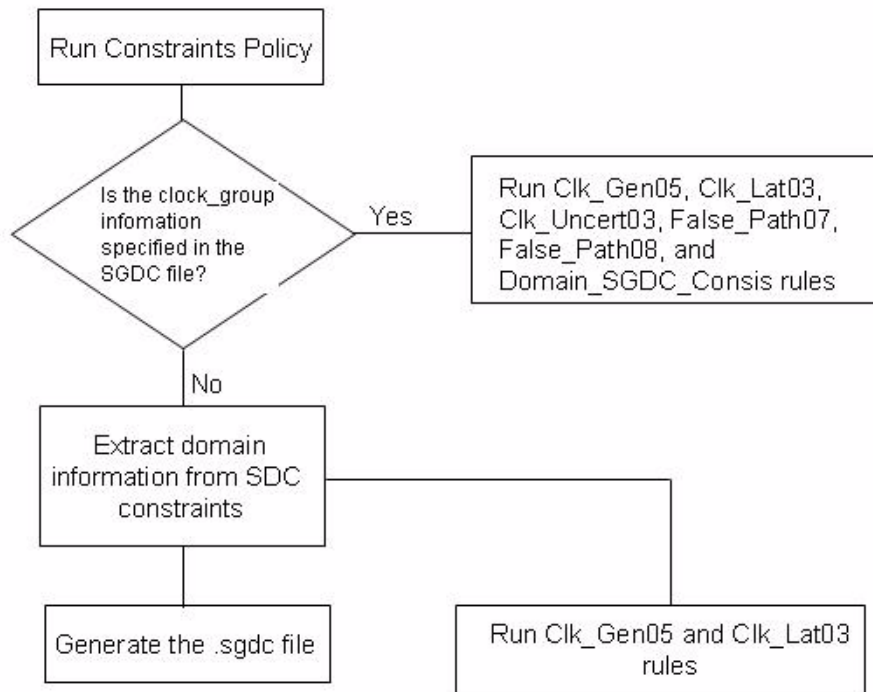
This topic describes the following:

- [Use Model for Inferring Domain](#)
- [Specifying Domains](#)

Use Model for Inferring Domain

For information on inferring clock domain information, refer to [Specifying Domains](#).

The following flowchart displays the use model for inferring domain:



If the *clock_group* constraint is not specified in the SGDC file, the *Clk_Uncert03*, *False_Path07*, *False_Path08*, and *Domain_SGDC_Consis* rules will

not report any violation. The [Clk_Gen05](#) and [Clk_Lat03](#) rules will use the domain inferred from the SDC constraints if the [clock_group](#) is not specified in SGDC file.

Specifying Domains

You can use domain inference feature to determine relationships between clocks. By using this feature, you can determine whether two clocks are synchronous or asynchronous to each other. However, the relationship inferred is dictated by the domain mode. There are four types of domain modes: STA (Default), TC_STA, STRICT, and SGDC. You can set the domain mode by using the [tc_domain_mode](#) parameter.

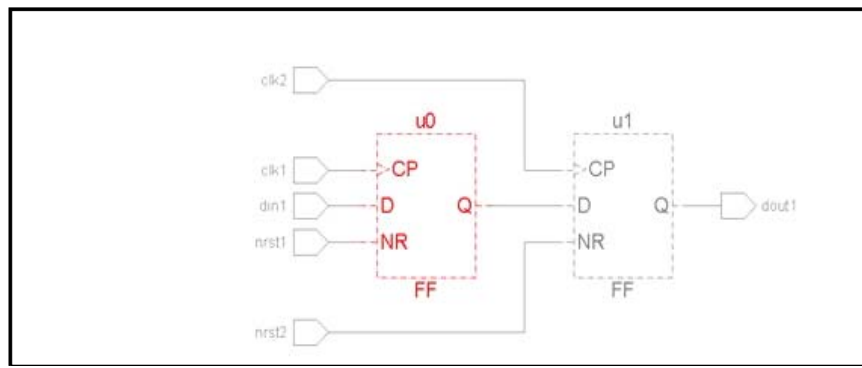
After you have specified the domain mode, the domain rules check the validity of the SDC constraints ([set_false_path/ set_clock_uncertainty](#)) and report any contradiction or conflicts between the design and constraints or between different types of SDC constraints.

For more information, refer to the following sections:

- [Inferring Clock Domain Information](#)
- [Comparing STA and TC_STA Domain Modes](#)
- [Setting the Domain Mode](#)
- [Identifying Missing Domains and Cyclic Dependencies](#)

Inferring Clock Domain Information

The [DomainInfo](#) and [DomainError](#) rules infer clock domain information. Clock crossings impact domain inference. For example, if there are two flip-flops, A and B, such that there exists a path from Q-pin of A to D-pin of B, as shown in the following schematic.



CLK1 is driving Flop A

CLK2 is driving Flop B

Therefore, there is a valid crossing from CLK1 to CLK2.

In the TC_STA mode, the relationship between CLK1 and CLK2 is inferred from SDC constraints only when there is a clock crossing from CLK1 to CLK2.

However, in all other modes, clock crossings are not considered while inferring domains.

NOTE: *Clock crossings are directional. In the above example, a clock crossing from CLK2 to CLK1 does not exist.*

Apart from clock crossings, the [DomainInfo](#) and [DomainError](#) rules infer clock domain information from the following:

- **Clock crossings:** If no relationship is specified between two clocks and there is a valid crossing from one clock to another, the clock relationship is considered synchronous in TC_STA mode.
- **source-generated relation:** A pair of clock is said to have a source-generated relation and are considered as synchronous, if one clock is generated from the other. For example, consider the following content from the *.sdc file:

```
create_clock -name CLK1 -period 10 {clk1}
create_generated_clock -name GCLK1 {ff1/Q} -source {clk1}
-divide_by 2
```

In the above example, the CLK1 and GCLK1 clocks are considered as

synchronous.

- **set_clock_groups:** There are two inferences for the `set_clock_groups` constraint: Asynchronous and Synchronous.

Intergroup (Asynchronous)

If two clocks are specified in different groups of the `set_clock_groups` constraint, these two clocks are considered as asynchronous.

For example, consider the following content from the *.sdc file:

```
create_clock -name C1 -period 10 clk1
create_clock -name C2 -period 13 clk2
set_clock_groups -asynchronous -group {C1} -group {C2}
```

In the above example, the C1 and C2 clocks are considered as asynchronous.

In STA mode, only `set_clock_groups` constraint with `-asynchronous` option are considered for domain inference. In TC_STA mode, the following `set_clock_groups` constraint options are considered for domain inference: `-asynchronous`, `-physically_exclusive`, and `-logically_exclusive`.

Intragroup (Synchronous)

If two clocks are specified in one group of the `set_clock_groups` constraint, these two clocks are considered as synchronous if there is no conflict. This inference is applicable only in STA mode.

For example, consider the following content from the *.sdc file:

```
create_clock -name C1 -period 10 clk1
create_clock -name C2 -period 13 clk2
create_clock -name C3 -period 10 clk3
set_clock_groups -asynchronous -group {C1 C2} -group {C3}
```

In the above example, the C1 and C2 clocks are considered as synchronous.

- **set_false_path:** If the `set_false_path` constraint is specified between two clocks, then these two clocks are considered as asynchronous.

NOTE: For more information on the `set_false_path` constraint, please refer to the

Appendix: SDC Constraints in the SpyGlass Constraints Rules Reference Guide.

For example, consider the following content from the *.sdc file:

```
create_clock -name CLK1 -period 10 {clk1}
create_clock -name CLK2 -period 11 clk2}
set_false_path -from {CLK1} -to {CLK2}
```

In the above example, the CLK1 and CLK2 clocks are considered as asynchronous.

- **set_clock_uncertainty:** If the `set_clock_uncertainty` constraint is specified between the two clocks, these two clocks are considered as synchronous.

NOTE: *For more information on the `set_clock_uncertainty` constraint, refer to the Appendix: SDC Constraints in the SpyGlass Constraints Rules Reference Guide.*

For example, consider the following content from the *.sdc file:

```
create_clock -name C1 -period 10 clk1
create_clock -name C2 -period 13 clk2
set_clock_uncertainty 2 -from C1 -to C2
```

In the above example, the C1 and C2 clocks are considered as synchronous.

- **Virtual-clock relation:** To synchronize the virtual clock with the corresponding real clock, set the value of `tc_virtual_clock_prefixes` and `tc_virtual_clock_suffixes` parameters.
- **Harmonic relationship:** A pair of clocks is said to have a harmonic relation if their clock periods can be synchronized in N cycles (where N is the value which is less than or equal to the value specified using the `tc_c2c_max_cycles` parameter) and their duty cycles are the same. Harmonic relationships are only considered in the TC_STA mode.
- `clock_group`: Considered only in SGDC mode.

Also, refer to [Use Model for Inferring Domain](#) in the Specifying the SGDC Files section.

Comparing STA and TC_STA Domain Modes

The following table shows the key differences in the analysis of domains

Inferring Clock Domains When No Domain is Specified

when the domain mode is set to STA and TC_STA.

TABLE 2 STA and TC_STA Domain Comparison

STA	TC_STA
Order of priority of constraints is: <ol style="list-style-type: none"> 1) set_clock_groups (Intergroup Async) 2) set_false_path 3) set_clock_uncertainty 4) create_generated_clock 5) Virtual clocks 6) set_clock_groups (Intragroup Sync) 	Order of priority of constraints is: <ol style="list-style-type: none"> 1) create_generated_clock 2) set_clock_groups (Intergroup Async) 3) set_false_path 4) set_clock_uncertainty 5) Harmonic Relationships 6) No-Constraint (having Clk Crossing - Sync) 7) Virtual clock relation (SYNC)
Clock crossings are not considered at all	Clock crossing is checked between every two clocks before applying any constraint to infer the relation. <p>Example 1</p> <p>If CLK1 and CLK2 have no clock crossing and the following is defined:</p> <pre>set_clock_groups -async -group CLK1 -group CLK2</pre> <p>CLK1 and CLK2 will not be assigned an asynchronous relation because no crossing exist between them. Therefore, it is as good as no set_clock_groups was defined between them.</p> <p>Example 2</p> <p>If CLK1 and CLK2 have clock crossing and set_clock_groups has been defined between them.</p> <p>CLK1 and CLK2 will be assigned two different domains.</p> <p>Example 3</p> <p>If CLK1 and CLK2 have clock crossing and no constraint was defined between them.</p> <p>These clocks will be considered as synchronous and will be assigned same domain.</p>

TABLE 2 STA and TC_STA Domain Comparison

STA	TC_STA
There is no handling for Harmonic relations.	Harmonic relationships are considered between clocks having clock crossing.
Conflicting clocks are not assigned domains and they are reported under missing domain.	Conflicting clocks are assigned same domains.
set_clock_groups is only considered with the asynchronous argument.	set_clock_groups is considered with logically_exclusive , physically_exclusive , and asynchronous.

Setting the Domain Mode

To set the domain, use the [tc_domain_mode](#) parameter. You can set the domain as STA, TC_STA, STRICT, or SGDC.

Identifying Missing Domains and Cyclic Dependencies

After you have set the domain mode, either run the:

- `sdc_audit` goal, or
- the [DomainError](#) and [DomainInfo](#) rules.

These rules generate the Domain Matrix, Cyclic Dependency, and Missing Domains reports, which displays domains assigned to clocks based on the SDC constraints specified. In addition, these reports provide the synchronous/asynchronous relationship with other clocks. By interpreting these reports, you can verify the results of other violation messages, which use the domain assignment shown in the reports.

For more information, refer to the following sections:

- [Timing Constraints Domain Reports](#) (Best viewed in Spreadsheet Viewer)
- [Mnemonics in the Domains Spreadsheets](#)
- [Example 3 - Domain Matrix](#)
- [Example 4 - Domain Matrix and Missing Domains](#)

Inferring Clock Domains When No Domain is Specified

Defining Technology Library Information

To use the target technology library file, you need to precompile it to the SpyGlass-format library using the SpyGlass Library Compiler feature and specify the compiled libraries by adding `read_file -type sglib <file>` in the project file.

When checking certain parameters, it is important to compare the corresponding `min` and `max` values in the target technology library file (Liberty™ format `.lib` file). In addition, the following rules require you to specify the SpyGlass-format precompiled target technology library file as they need the synthesizable description of all library cells:

- *Block05*
- *Clk_Gen01a*
- *Clk_Gen06*
- *Inp_DeI03b*
- *Test_Rules05*
- *Test_Rules06*

Case analysis support varies with the target technology library specified using `gateslib` and its precompiled version specified using `sglib` in the project file. This variance in support is because of the following issues:

- **Using the target technology library file:** When SpyGlass hits a library instantiated library gate while traversing the netlist, all of its terminals are considered to be blocked because the SpyGlass Logic Evaluator does not identify the functionality of the gate needed for simulation. Therefore, the fan-in or fan-out of that gate is masked, resulting in the corresponding rule behavior change when compared to analyzing a simple RTL gate.
- **Using the precompiled target technology library file:** SpyGlass converts cells, such as RAMs to black boxes. It may affect the path-traversal performed by the rules. Therefore, the number of rule messages may be different when compared to analyzing a simple RTL gate.

Handling Technology Library Cell Pin Names

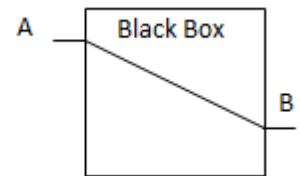
Some pre-synthesis SDC files refer to technology-specific pin names that will ultimately be present in the post-synthesis design. For example, a constraint may make reference to `foo_reg/CK`, or more commonly to `*/CK`. This means a reference to the CK pin on a register that will be inferred when the design is synthesized and mapped to the final technology. When SpyGlass analyzes the RTL description, there is no knowledge about post-synthesis pin names and the design is synthesized internally using SpyGlass's own synthesis library, which may use different pin names. Therefore, you need to define pin name mappings for the different types of pins for SpyGlass.

Use the [mapped_pin_map](#) constraint to declare these pin names to SpyGlass.

Handling Black Boxes

If your design contains black box models that are not otherwise qualified, e.g. by .lib models, you may need to further specify details for these models, e.g. by specifying combination paths through the black box using the SGDC assume-path constraint:

```
assume_path -name BB -input A -output B
```



Using the SpyGlass Constraints Solution

This section contains the following topics:

- *Constraints Verification*
- *Constraints Generation*
- *Timing Exception Verification*
- *Constraints Management*

Constraints Verification

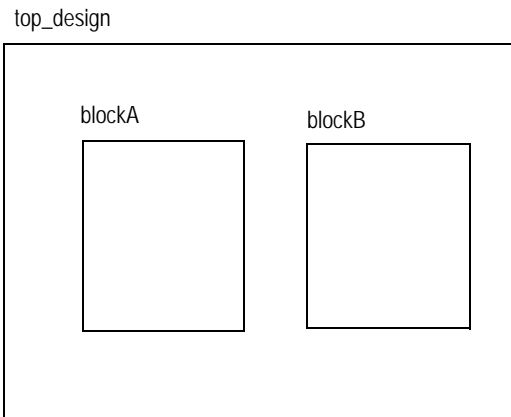
When you generate your constraints, it is wise to check they follow SDC best practices before embarking on lengthy STA runs. You should understand how to correctly setup for verification.

The SpyGlass Constraints solution works at two levels:

- **Block-level:** This is a design partition that would be synthesized. A block may consist of one or more Verilog modules or VHDL entity-architecture pairs.
- **Design:** This is a top-level design unit, which typically instantiates multiple blocks.

To specify SDC files at these two levels, the SpyGlass Constraints solution uses two main SGDC constraints, *sdc_data* and *block*.

Consider the following design that has a top-level design unit named `top_design` has two synthesis partitions or *blocks* named `blockA` and `blockB`:



Assuming that you have the SDC files for all units (`top_design`, `blockA`, and `blockB`) and want to perform rule-checking on these units, you need to create a SGDC file as follows:

```
current_design top_design
  sdc_data -file <top-design-SDC-file-list> ...
  block -name blockA blockB
```

```
current_design blockA
  sdc_data -file <blockA-SDC-file-list> ...
```

```
current_design blockB
  sdc_data -file <blockB-SDC-file-list> ...
```

The first *current_design* specification is for the top-level design unit `top_design` and specifies the corresponding SDC file(s) using the *sdc_data* constraint. It also specifies that there are two blocks — `blockA` and `blockB` under this top-level design unit using the *block* constraint.

The second *current_design* specification is for the block design unit `blockA` and specifies the corresponding SDC file(s) using the *sdc_data* constraint. Similarly, the third *current_design* specification is for the block design unit `blockB` and specifies the corresponding SDC file(s) using the *sdc_data* constraint.

The rules in the SpyGlass Constraints solution are then run on the following units:

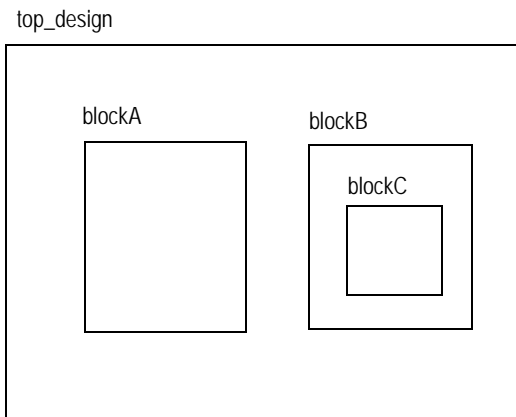
- The `top_design` design unit flattened down to the boundaries of blocks `blockA` and `blockB`.
- The block design unit `blockA` flattened down to leaf-level
- The block design unit `blockB` flattened down to leaf-level

By default, all design units specified with a *current_design* specification are considered blocks and Block-level rules are run on these design units. Hence, in the above example, `top_design` is a block along with `blockA` and `blockB`. To avoid running the Block-level rules on the top-level design, you need to set the *chip* parameter as follows for the above example:

```
set_parameter chip 'top_design'
```

Then, certain block-level rules are not run on the `top_design`. Refer to the *chip* parameter description for the list of these rules.

Consider another example where the design has a top-level design unit named `top_design` has two synthesis partitions or *blocks* named `blockA` and `blockB` and the block `blockB` in turn has another sub-block named `blockC`:



For the above example, you need to create a SGDC file as follows:

```
current_design top_design
  sdc_data -file <top-design-SDC-file-list> ...
  block -name blockA blockB

current_design blockA
  sdc_data -file <blockA-SDC-file-list> ...

current_design blockB
  sdc_data -file <blockB-SDC-file-list> ...
  block -name blockC

current_design blockC
  sdc_data -file <blockC-SDC-file-list> ...
```

The third *current_design* specification is for the block design unit `blockB` and specifies the corresponding SDC file(s) using the *sdc_data* constraint. It also specifies that there is one sub-block — `blockC` under this block using the *block* constraint.

The fourth *current_design* specification is for the block design unit `blockC` and specifies the corresponding SDC file(s) using the *sdc_data* constraint.

The rules in the SpyGlass Constraints solution are then run on the following units:

- The top_design design unit flattened down to the boundaries of blocks blockA and blockB
- The block design unit blockA flattened down to leaf-level
- The block design unit blockB flattened down to the boundary of the block blockC
- The block design unit blockC flattened down to leaf-level

NOTE: *SpyGlass Constraints does not read any SGDC constraint that has a corresponding SDC constraint.*

Constraints Generation

This topic describes the following:

- [Incrementally Generating SDC](#)
- [Merging Multiple Modes for SDC Files](#)
- [Merging SDC Files from Lower-level Blocks](#)
- [Using SDC Autofix](#)

Incrementally Generating SDC

The [SDC_GenerateIncr](#) rule generates the template specifications *incrementally* by using a seed file.

The following diagram illustrates the incremental flow of using this rule.

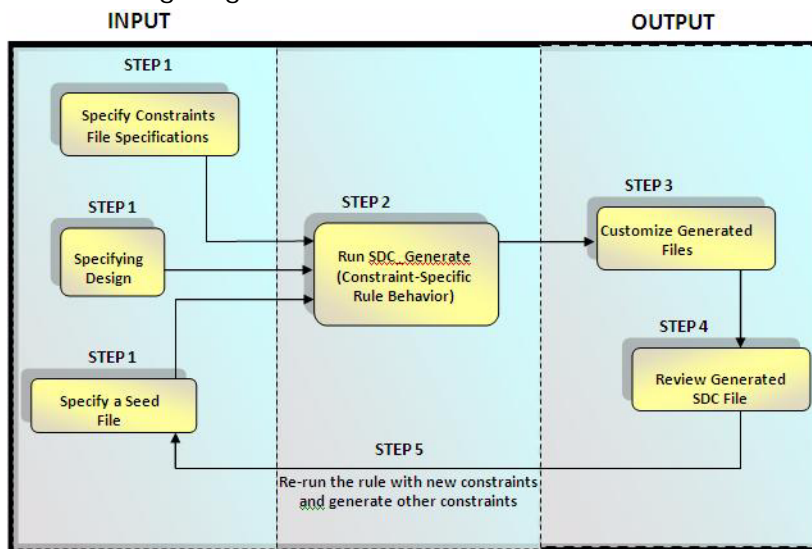


FIGURE 1. Using the `SDC_GenerateIncr` rule

NOTE: The template SDC file generated by the `SDC_GenerateIncr` rule is version 2.0. Therefore, you can now add comments in the template CSV file for the following constraints: `create_clock`, `create_generated_clock`, `set_false_path`, `set_multicycle_path`, `set_max_delay/set_min_delay`, and `set_clock_groups`.

Click the following links to explore each step in the process.

- [Specifying Design](#)
- [Specifying the Constraints Specification File](#)
- [Specifying a Seed File](#)
- [Constraint Specific Rule Behavior](#)
- [Customize Generated Files](#)

While generating the output SDC file, an auxiliary SGDC file is also generated which is derived from the SGDC file specified in the SpyGlass run. The SGDC file is created in the same directory where the SDC file is created. The name of the SGDC file is as follows:

```
<existing sgdc filename>_<output sdc file name>.sgdc
```

Use this SGDC file for subsequent SpyGlass runs, where the output SDC is considered as 'seed' constraints.

NOTE: To generate the SDC file for a block, make sure you specify `block` in the `current_design`.

Apart from generating the SDC constraints as illustrated in the previous diagram, you can generate Tcl parameters for the constraints in the SDC file. The only difference in the parameterized flow with the one illustrated in the previous diagram is that instead of containing the timing values, the SDC file contains references to the generated Tcl variable, which contains the timing value. For more details, refer to the [Parameterized SDC File](#) section.

Specifying Design

Before running the [SDC_GenerateIncr](#) rule, you need to read in the design in Console GUI. The design can be in Verilog or VHDL. While specifying the design, you also need to specify an SGDC file in Console GUI.

Though the SpyGlass Constraints solution also (including the [SDC_GenerateIncr](#) rule) works with the `-v/-y` models, it is recommended that for running the [SDC_GenerateIncr](#) rule, the `.lib` descriptions (in the form of `.sglib`) are provided for library cells instantiated in the design. This is because the SDC generated by the [SDC_GenerateIncr](#) rule is used by the synthesis/STA tools that use `.lib` (in the form of custom compiled format), rather than the `-v/-y` models. Therefore, in case there are discrepancies in the `-v/-y` models against the corresponding `.lib` models, the SDC

generated by the *SDC_GenerateIncr* rule might not work with the .lib models in the synthesis/STA tools. This discrepancy is not only limited to the difference in functionality, which is unlikely, but also about -v/-y being modeled differently. Therefore, causing certain objects, for example hierarchy and nets/registers, to be visible inside the -v/-y model, which are not visible in the .lib models.

Specifying the Constraints Specification File

SpyGlass provides a Constraints Specification file *gensdcConstraintsFile.txt*. This file contains a list SDC constraints that you can generate. The *set_case_analysis*, *create_clock*, and *create_generated_clock* constraints are generated by default. However, you can edit the file to generate more constraints.

It is recommended that you first generate the default constraints and then run the rule. In the second pass, edit the Constraints Specification file by deleting the comments from the constraints you want to generate and then run the rule again. Continue doing this until you have generated all the constraints for your design.

Specifying a Seed File

The seed file implies there exists constraints which are correct. The rule reads these constraints and generates additional constraints as desired by using the Constraints Specification file. The generated constraints are in the output template SDC file.

If you have generated an SDC file for a block, the generated constraints are propagated from top to block. The following constraints are propagated: *create_clock*, *create_generated_clock*, *set_output_delay*, and *set_input_delay*. Constraints are propagated only if they exist in the top.sdc file given as a seed. The *create_clock* and *create_generated_clock* constraints that are generated have the same characteristics as those in the top.sdc file.

The seed file can be an existing SDC or SGDC file. You can:

- *Specify the SGDC file as a Seed File*
- *Specify an SDC file as a seed file*

Specify the SGDC file as a Seed File

The SGDC file is considered as a seed file for a design unit if there are no *sdc_data* constraints with mode *seed* given for the design unit. The SGDC constraints are put into the output template SDC file.

The following seed SGDC constraints are read by the rule:

- clock
- set_case_analysis

All other constraints are ignored.

The correct identification of all clocks in the SGDC file prior to running the rule is required. The SGDC file should be as follows:

```
current_design <design_name>
clock -name "<design_name>.<clock1_name>" -domain domain1
clock -name "<design_name>.<clock2_name>" -domain domain2
```

The clock period is specified in the SGDC file and this is imported directly into the generated template. The domain specified in the SGDC file is used to generate [set_false_path/set_clock_uncertainty/set_clock_groups](#) between interacting clock pairs, as shown below:

```
current_design <design_name>
clock -name "<design_name>.<clk1_name>" -domain domain1 -
period 10
```

The [SDC_GenerateIncr](#) rule supports the `-domain` option of the `clock` constraint. If you specify the `-domain` option for at least one clock, the [set_false_path/set_clock_groups](#)/interclock uncertainty constraints are automatically marked for generation. This is triggered only when the SGDC file is used as seed.

For all interacting clock pairs, if you specify the `-domain` option for both the clocks, and the domain values are the same, the interclock uncertainty is generated. However, if the domain values are different, the [set_false_path/set_clock_groups](#) constraints are generated. All these pairs are dumped in the crossing file, which you can review and edit. For details on the crossing file, refer to the [Customize Generated Files](#) section.

If you define a clock using the naming convention `<top-name>.<port-name>` in the SGDC file, the [SDC_GenerateIncr](#) rule creates a [create_clock](#) constraint against the port `<port-name>` and names the clock as `<top-name>_<port-name>` in the SDC file.

The [SDC_GenerateIncr](#) rule assumes that the SGDC information required for the generation of constraints is present in the SGDC file. Whenever the rule generates a constraint that uses SGDC information, it extracts the information from the SGDC file. In addition, this rule assumes that the

SGDC data in the SGDC file remains the same. If the data changes, the generated constraint reflects the changed data. At present, only reset SGDC constraints are being used for the generation of SDC constraints. The SDC constraints using the reset information are [set_max_transition](#) and [set_ideal_net](#).

After the [SDC_GenerateIncr](#) rule uses the SGDC as a seed, all consequent runs of the rule use the generated SDC file as a seed to generate constraints as per the Constraints Specification file. The flow for this is described in the next section.

Specify an SDC file as a seed file

You specify the seed SDC file using the `-mode seed` argument of the [sdc_data](#) constraint, as shown in the following example:

```
sdc_data -file mySDC.sdc -mode seed
```

The following SDC commands are read by the [SDC_GenerateIncr](#) rule:

set_case_analysis	create_clock
create_generated_clock	set_input_delay
set_output_delay	set_clock_latency
set_clock_uncertainty	set_clock_transition
set_propagated_clock	set_input_transition
set_driving_cell	set_drive
set_load	set_max_delay/set_min_delay
virtual_clock	set_false_path
set_max_transition	set_clock_groups
set_max_fanout	set_max_capacitance
set_ideal_net	

All other commands in the seed SDC file are ignored.

The rule generates the template specifications for only those SDC commands that are specified with the [gen_sdc_constraints_file](#) rule parameter. This rule creates the template specifications for only those SDC commands for which the specifications are not already available in the seed file.

NOTE: You can specify a CSV file as an input by using the [gen_sdc_csv2sdc_tool](#) parameter. Then, the [SDC_GenerateIncr](#) rule automatically translates the CSV file

to SDC.

Constraint Specific Rule Behavior

The following table describes the behavior of the *SDC_GenerateIncr* rule when you generate specific constraints. Apart from this table, refer to the [Parameter\(s\)](#) section. To generate the constraint, specify the constraint in the Constraints Specification file.

Constraint To Be Generated	Description
set_case_analysis	This rule locates the select pins of all multiplexers by traversing backwards from all unconstrained clock pins. The rule adds the non-constant select pins of those multiplexers that have its data pins constrained by clocks. It generates <i>set_case_analysis</i> at all blackbox terminals, output pins of sequential cells, and ports reachable to these select pins.
create_clock	<p>Clock generation is a multi-step process, as follows:</p> <p>In the first step, the rule locates the ports reachable from the unconstrained clock pins of sequential cells through single-input, single-output (SISO) cells.</p> <p>In the second step, the ports reachable from the remaining unconstrained clock pins of sequential cells are located.</p> <p>If a clock is passing through the input pin of a mux and the fan-out of the mux is reaching a sequential cell clock pin, all input pins of the mux should be constrained through clock signals. The rule traverses backwards from these input pins and identifies the ports reachable. If the clock is passing through the select pin of a mux, a generated clock is put on the output port of the mux.</p>

Constraint To Be Generated	Description
<i>create_generated_clock</i>	<p>This rule locates the sequential output pins reachable from the unconstrained clock pins of the sequential cells. In addition, if a clock is passing through the input pins of a mux and the fan-out of the mux is reaching a sequential cell clock pin, all input pins of the mux should be constrained through clock signals. The rule traverses backwards from these input pins and identifies the sequential pins reachable.</p> <p>This rule generates generated clock on all these sequential output pins. Before generating the clock, see if the two or more output pins of flops can be merged through the fan-out into a single output pin. Then, instead of creating a clock per flip-flop, create one generated clock at the output pin with the source pin as the definition point of the master clock.</p> <p>For every master clock propagating to the existing generated clocks, a corresponding generated clock is created, if it does not exist. If the number of master clocks is more than one, create a generated clock for each master clock.</p>

Constraint To Be Generated	Description
<i>virtual_clock</i>	<p>This rule generates virtual clocks for all primary clocks not mapped to any existing virtual clock. The primary clocks are mapped to virtual clocks through the following methods:</p> <p>Naming Convention A virtual clock matches the real clock if their names satisfy the following structure: <virtual clock name> = <prefix>_<real clock name>_<suffix></p> <p>Clock Characteristics Method The virtual clock matches a real clock if both their period and waveform match.</p> <p>Each virtual clock is mapped to a single real clock. A real clock is first determined through the Naming Convention method and, if none is found, then through the Clock Characteristics method. If a real clock is identified, the virtual clock is mapped to it.</p>
<i>set_input_delay</i>	<p>If the port is constrained with a clock/case analysis constraints, the rule ignores the input port. The rule traverses the path from the port to the data pins of the flops. The input delay is generated on the port for each clock sampling the flops.</p> <p>You can generate I/O delays for virtual clocks. To do this, <i>virtual_clock</i> either needs to be generated or present in the seed SDC file.</p>
<i>set_output_delay</i>	<p>The rule traverses the path from the port to the output pins of the flip-flop. The output delay is generated on the port for each clock sampling the flops.</p> <p>You can generate I/O delays for virtual clocks. To do this, <i>virtual_clock</i> either needs to be generated or present in the seed SDC file.</p>
<i>set_input_delay</i> , <i>set_clock_uncertainty</i>	<p>The rule generates the clock latency and simple clock uncertainty constraints for all clocks, if they do not already exist.</p>

Constraint To Be Generated	Description
<i>set_input_transition</i>	If the ports are constrained by the case analysis constraint, the rule ignores these input and inout ports. For the remaining ports, the rule generates input transition for each port if the design phase is Post-layout. Otherwise, input transition is generated if the port is not constrained by a clock. Hanging ports are ignored.
<i>set_load</i>	If the ports are constrained by the case analysis or clock constraints, the rule ignores these output and inout ports. For the remaining ports, the rule generates load constraints, if they do not already exist. Hanging ports are ignored.
<i>set_max_delay/ set_min_delay</i>	The rule generates min delay/max delay for all port to port combinational paths. The rule ignores ports that are constrained by clocks.
<i>set_driving_cell</i>	The rule generates driving cell constraints for all input or inout ports, if they do not already exist. This rule ignores ports constrained by case analysis or clock constraints. Hanging ports are ignored.
<i>set_drive</i>	The rule generates drive constraints for all input and inout ports, if they do not already exist. This rule ignores ports constrained by case analysis or clock constraints. Hanging ports are ignored.
<i>set_max_transition</i>	The rule generates max transition constraints for the design if the <i>tc_opt01_port_des</i> parameter is set to any valid value other than 'port'. This rule generates max transition constraints for ports if the <i>tc_opt01_port_des</i> parameter is set to any valid value other than 'design'. If the <i>tc_opt01_port_des</i> parameter is set to 'design_or_port', the rule generates max transition constraints for both design and ports.
	The rule uses 'reset' SGDC constraints in the constraint generation. This rule ignores hanging ports and ports constrained by case analysis, clock, or 'reset'. For all other ports, max transition constraints are generated.

Constraint To Be Generated	Description
<i>set_max_capacitance</i>	<p>The rule generates max capacitance constraints for the design if the <i>tc_opt01_port_des</i> parameter is set to any valid value other than 'port'. This rule generates max capacitance constraints for ports if the <i>tc_opt01_port_des</i> parameter is set to any valid value other than 'design'. If the <i>tc_opt01_port_des</i> parameter is set to 'design_or_port', the rule generates max capacitance constraints for both design and ports.</p> <p>The rule ignores hanging ports and ports constrained by case analysis are ignored. For all other ports, max capacitance constraints are generated</p>
<i>set_max_fanout</i>	<p>The rule generates max fan-out constraints for the design if the <i>tc_opt01_port_des</i> parameter is set to any valid value other than 'port'. This rule generates max fan-out constraints for ports if the <i>tc_opt01_port_des</i> parameter is set to any valid value other than 'design'. If the <i>tc_opt01_port_des</i> parameter is set to 'design_or_port', the rule generates max fan-out constraints for both design and ports.</p> <p>This rule generates max fan-out constraints for all input ports.</p>

Constraint To Be Generated	Description
set_ideal_net	<p>The rule uses 'reset' SGDC constraints in the constraint generation. The constraint is generated for all design objects constrained by real clocks or 'reset' SGDC constraint. For all nets, if their fan-out count is greater than the fan-out limit, which is specified through the tc_fanout_limit parameter, ideal net constraint is generated on the net.</p>
set_false_path/inter clock uncertainty/ set_clock_groups	<p>The rule generates inter-clock uncertainty for all interacting clock pairs having the same root clock. Other interacting unconstrained clock pairs are dumped into crossings file. If clocks are read from the SGDC file and the 'domain' option is defined, the 'domain' option is used to decide the type of constraint for the clock crossing pairs. Inter-clock uncertainty is declared for clock pairs having the same 'domain' option value. Otherwise, false path or clock group is specified.</p> <p>If the 'domain' option is undefined for either of the clocks, '?' is specified in the crossings file. If both false path and clock group are specified in the Constraints Specification file, false path is specified. Refer to the Clock Crossings file section to generate appropriate constraints.</p>

Customize Generated Files

The `SDC_GenerateIncr` rule generates the following files:

■ Clock Crossings file

This file is named `<design-name>-crossings.csv`. It is in a comma-separated data file format. It contains name pairs of interacting but unconstrained clocks in the following format:

```
<clock1-name>, <clock2-name>, ?
```

The following message is appears after the crossing file is generated:

```
Crossings file containing unconstrained Clock
Pairs generated for design/block <name>
```

You need to first modify the clock crossings file so that the CSV file, <design-name>.csv, is updated. Unless you update the clock crossing file, <design-name>-crossings.csv, you cannot edit the <design-name>.csv file in Spreadsheet Viewer. However, you can edit the file after the crossing file has been processed.

To modify the clock crossing file, double-click the message for the generated constraints.

Next, replace the ? character for each pair as shown in the table below:

Replace ? Character with...	To Generate...	Relationship
F	set_false_path	Not applicable
U	set_clock_uncertainty	Not applicable
GA	set_clock_groups	asynchronous
GL	set_clock_groups	logically_exclusive
GP	set_clock_groups	physically_exclusive

After making changes for all clock pairs, save the changes. After you save the changes, Console GUI displays the status of the crossing pairs classification. After classification, the CSV file contains an updated list of [set_false_path](#), [set_clock_uncertainty](#), and/or [set_clock_groups](#) constraints as specified.

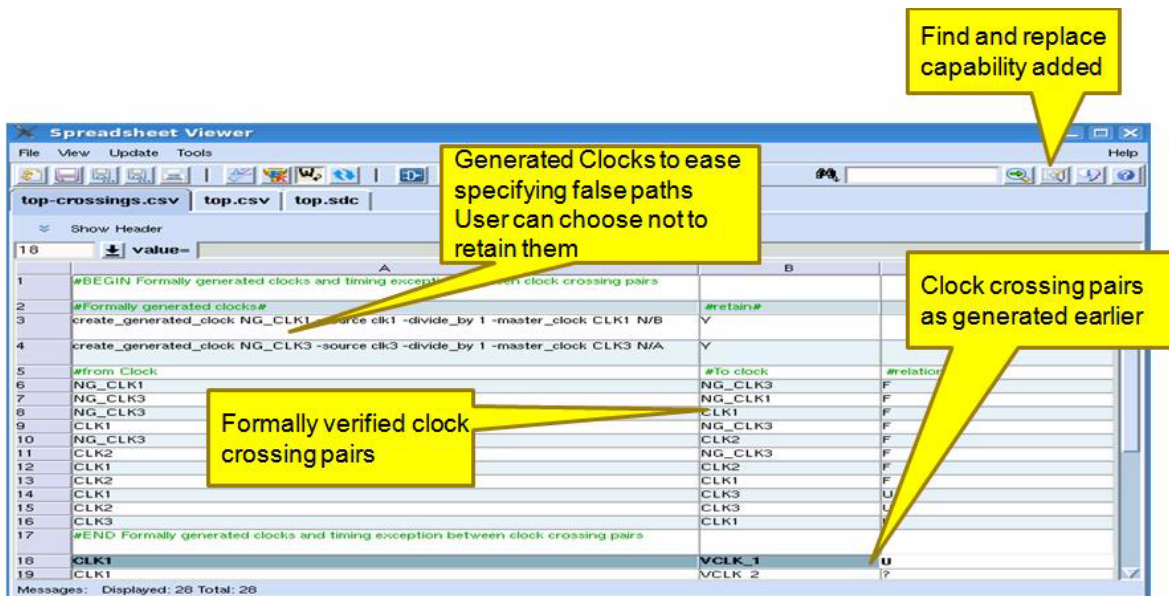
After you have modified the crossing file, click the **Update** button to process it.

NOTE: *You cannot edit the crossing file after you have clicked the Update button.*

Set the [gen_c2cVerify](#) policy parameter to formally verify the generated clock to clock paths. The verified pairs are set as F for false paths and U for uncertainty. The SpyGlass TXV product is invoked to verify the clock-to-clock paths.

Additional generated clocks may be generated during the verification process. You can choose to retain these clocks and the corresponding verified clock-to-clock pair classifications.

The general structure of the crossing file is shown in the following screenshot. The generated clock definitions are followed by formally verified clock-to-clock pair classifications. Following this is the non-verifiable clock-to-clock pairs.



■ CSV file

This file is created for each design unit. The CSV file contains generated SDC constraints for which you need to specify the timing information. For design objects for which the I/O delay is not generated, a CSV file, called unconstrained_<block_name>.csv, is generated. This file contains the list of such objects and the reason for not generating the I/O delay. The CSV file is displayed as a worksheet along with the CSV file containing generated constraints.

The following message is also generated:

CSV file "<file-path>" for importing timing numbers generated for design/block <name>

Click the message for each CSV file to open it the file in the Spreadsheet Viewer window of Console GUI, as shown in the following image. Enter the missing timing information.

Some formulae have been introduced in the spreadsheet that link the cells which require timing information. This makes editing of cells easier. For example, assume two input delay constraints are generated for two input ports (P1 and P2) corresponding to the same clock CLK1 and you want to specify the same value for both the constraints. To do this, you need to specify the value in the top-left cell only. All the corresponding

cells are automatically populated with the same value.

	A	B	C	D	E	F
1	#set	PName	PValue	PComment		
2	#source	FPath				
3	source	../tech_files/45nm.tcl				
4	set	period1	20			
5	set	a	10			
6	set	CCLK_1_p	40			
7	set	in_0_CCLK_1_mnr_it	3.2			
8	set	in_0_CCLK_1_mnf_it	3.2			
9	set	in_0_CCLK_1_mxr_it	2.3			
10	set	in_0_CCLK_1_mxf_it	2.3			
11						
12	#create_clock	name	objectlist	period		
13	create_clock	CCLK_1	{get_ports {clk}}	\$CCLK_1_p		
14						
15	#set_input_delay	objectlist	add_delay	clock	min rise value	min fall value
16	set_input_delay	{get_ports {in[0]}}	1	{get_clocks {CCLK_1}}	\$in_0_CCLK_1_mnr_it	\$in_0_CCLK_1_mnf_it
17						

To update the .CSV file, wherever applicable, you can enter a space-separated list of numbers (float/integer). However, do not enclose the list in curly braces {}.

After making the required changes, click **Generate SDC** from the **File** menu to generate the SDC file. This opens the **File** dialog box where you specify the name of the SDC file that needs to be generated.

Parameterized SDC File

Apart from generating the SDC constraints as explained in the previous sections, you can generate a parameterized SDC file. In this section, the following topics are covered:

- [Generating Parameterized Template SDC File](#)
- [Constraints Supported](#)
- [Parameterization Flow](#)
- [Parameter Generation for an SDC Constraint](#)
- [Naming Convention for Parameter Values](#)
- [Generating Comments for Parameters](#)

Generating Parameterized Template SDC File

You can generate the Tcl parameters in a template SDC file. By default, the *SDC_GenerateIncr* rule does not generate parameters for the timing values in the SDC template. To generate parameters, perform the following steps:

1. Set the value of the *tc_enable_param_sdc_flow* parameter to yes.
2. Specify the parameter description (in a specific format) for individual constraints in the SpyGlass Parameter Naming Convention (SPNC) file (see *Naming Convention for the File*) and pass the name of the file as the value to the *gen_sdc_param_file* parameter.

Constraints Supported

Currently, parameter support is provided for the following constraints:

<i>set_case_analysis</i>	<i>create_clock</i>
<i>create_generated_clock</i>	<i>set_input_delay</i>
<i>set_output_delay</i>	<i>set_clock_latency</i>
<i>set_clock_uncertainty</i>	<i>set_clock_transition</i>
<i>set_input_transition</i>	<i>set_driving_cell</i>
<i>set_drive</i>	<i>set_load</i>
virtual_clock	<i>set_max_transition</i>
<i>set_max_capacitance</i>	<i>set_max_fanout</i>
<i>set_max_delay/</i> <i>set_min_delay</i>	

The parameterization of the *set_load* constraint is implemented via two mnemonics, *pin_load* and *wire_load*. *pin_load* is for specifying the parameterization for pin loads and *wire_load* is for wire loads.

The *-name* fields of the clocks can also be parameterized. The name of the newly generated clocks is then changed according to the specified format. This change is reflected in all appropriate SDC constraints.

Parameterization Flow

The flow of the *SDC_GenerateIncr* rule is not affected with this change. The SDC file, instead of containing the timing values, directly contain a reference to the generated Tcl variable that contains the timing value. Following is the flow of the rule in this case:

1. The SPNC file is parsed. If there is an error in this file, the *SDCPARSE* rule reports a violation indicating the error and exits. When there are

multiple errors in the SPNC file, the [SDCPARSE](#) rule reports a violation for the first error it encounters. The file name and the line number are also displayed for each violation.

2. If the SPNC file is parsed correctly, the *SDC_GenerateIncr* rule runs as it does currently.
3. If the format for a constraint type is specified in the SPNC file, the constraints for that type are processed and the parameters are generated. In addition, the fields displayed in the CSV file depends on the SPNC format. Refer to the [Parameter Generation for an SDC Constraint](#) and [Naming Convention for Parameter Values](#) sections for details.
4. In the .CSV file, the constraints for which the parameters have been generated contain the reference to the parameter instead of the ?. Fill the timing values in the SET section created in CSV file for the generated parameters. The SET section is created for the generated parameters since the generated parameters are specified as set TCL commands in a new SDC file.

After filling the timing values, the template SDC is generated in a file whose name is given by you. All the generated parameters are written into a separate file whose name will be a derivative of the template SDC filename. For example, if the name of an SDC file is `template.sdc`, then `_param` is appended to the file `template.sdc` and the file name changes to `template_param.sdc`.

5. The Parameter SDC file is referred in the template SDC using the `source` command. The constraints for which the parameters have been generated refer to the parameters in their SDC translation.

Parameter Generation for an SDC Constraint

The template SDC contains the constraints of the seed SDC file and the new constraints. Currently, the phases for supported constraints are there in the [set_input_delay](#) and [set_output_delay](#) constraints only.

The SPNC file contains the following naming convention:

```
set_input_delay %objectlist.0.objname_clock.objname_mx -max
set_input_delay %objectlist.0.objname_clock.objname_mn -min
```

The above example means that the existing constraint should be partitioned into the `min` and `max` phase.

The following is the general method of generating parameters for fields of the supported constraint.

For each constraint in the template SDC:

- If the constraint is a new one:
 - a. Generate parameters for the fields for which SPNC is defined.
 - b. If SPNC is defined depending on the phases of the constraint, partition the new constraint into the phase combination specified in the SPNC. If phase dependence is not there, there will be a single partition containing all the phases.
 - c. Generate the parameter name from the SPNC template specified for each partition. The fields referred to in the SPNC template are then replaced with their actual values in the constraint to get the parameter name.
- If the constraint is an existing one, generate the parameters for the existing clocks only.

NOTE: For the clock pin of an inferred sequential cell, a parameter `cp_name` with the default value as `CP` is generated. The information about the parameter is present in the `<du-name>.sdc` file.

Naming Convention for Parameter Values

In addition to SPNC for parameter for all constraints, you can specify the SPNC for the parameter's values. For example, the [set_input_delay](#) constraints depend on the clock period. The same can be specified as follows:

```
set_input_delay %objectlist.0.objname_clock.objname_id
set %objectlist.0.objname_clock.objname_id %clock_period/2
```

If the above SPNC format is given for the [set_input_delay](#) constraints, the parameter is generated as explained above. In addition, SET constraints are generated where the generated parameter is set to the values specified in the template. For example, consider the following SDC commands:

```
create_clock -name Clk1 -period 10
set_input_delay ? -clock Clk1 in1
```

Following parameterization will be performed:

```
set $in1_Clk1_id 10/2

create_clock -name Clk1 -period 10
set_input_delay $in1_Clk1_id -clock Clk1 in1
```

NOTE: The generated parameter is set according to the SPNC defined for parameter's

value.

You can define multiple value expressions in a single statement by using ternary operator (?:) as follows:

```
set %objectlist.0.objname_clock.objname_id %clock_period < 5
? %clock_period/3 : %clock_period/2
```

If the clock period of the delay constraint is less than 5, the parameter value is `%clock_period/3`, otherwise it is `%clock_period/2`.

You can also use the design object's properties in the ternary operator expression. The following properties are defined, at present:

Object direction

- INPUT_OBJ: object direction is input
- OUTPUT_OBJ: object direction is output

Object type

- PORT_OBJ: Object is a port
- TERM_OBJ: object is a terminal
- INST_OBJ: object is a instance
- NET_OBJ: object is a net

You can use the object's properties as follows:

```
set %objectlist.0.objname_clock.objname_id
%objectlist.0.objname_objprop & $PORT_OBJ ? %clock_period/3
: %clock_period/2
```

The above statement sets the value to `%clock_period/3`, if the design object is a port. Otherwise, the value is set to `%clock_period/2`.

The fields to be shown in the `<design-name>.csv` file for a constraint depends upon the format specified in the SPNC file. In the following example, all the fields of the `create_clock` constraint are shown as per the SPNC format:

```
SPNC: create_clock -period %name_p -waveform %name_wv
Display: create_clock,name,objectlist,period,waveform
```

Similarly, if you do not specify a field in the SPNC format, it is not displayed. In the following example, the `waveform` field is not stated in the SPNC format and therefore the waveform is not displayed.

SPNC Purpose: waveform is not specified

SPNC: create_clock -period %name_p

Display: create_clock,period

Generating Comments for Parameters

Though the naming convention for parameters is intuitive, you can generate comments for each parameter by specifying a template containing placeholders for the constraints fields.

For example, in the following SPNC for a clock, a parameter for each clock period field is generated.

```
create_clock -period %name_p
```

For readability, use the `annotate` command to generate comments for a parameter. The syntax of the `annotate` command is:

```
annotate <parameter-format> <comment-template>
```

Where:

- `<parameter-format>`: To identify the parameter for which comment template is specified.
- `<comment-template>`: It is the string that defines the comment. `<comment-template>` starts with '#'. It can contain placeholders to refer to the parameter's associated constraints fields.

In this example, the `annotate` command would be:

```
annotate %name_p # period parameter for clock %name
```

In this example, the 'name' field of `create_clock` constraint is used. Similarly, for other constraints, default `annotate` commands have been added in the default SPNC file. You can adopt them and appropriately customize.

Specifying Clocks for the SDC_GenerateIncr Rule

The `SDC_GenerateIncr` rule requires you to specify the clocks in the design using the `clock` keyword in a SGDC file as follows:

```
current_design <du-name>
```

```
clock -name <clk-name>  
      -period <clk-period>  
      -edge <edge-list>
```

NOTE: The other arguments of the `clock` constraint including the `-testclock` argument (normally used by the SpyGlass DFT product) are not used in the analysis performed by the SpyGlass Constraints solution.

Where:

<du-name>

Specifies the module name (for Verilog designs) or the design unit name in `<entity-name>.<arch-name>` format (for VHDL designs).

<clk-name>

Specifies the clock port/pin name.

The pin can be a top-level port/pin as well as an internal pin.

You can specify a single port/pin name or a space-separated list of port/pin names.

For top-level port/pins, `<clk-name>` can be the port/pin's full hierarchical name or its simple name. For internal pins, `<clk-name>` must be the pin's full hierarchical name.

<clk-period>

Specifies the clock period as an integer value.

<edge-list>

(Optional) Specifies the clock edge value list.

By default, the clock edges are assumed to be 0, 50, 100, and so on.

The following command describes a 100MHz clock with posedge at 0 ns and negedge at 5 ns.

```
clock -name top.clk -period 10 -edge {0 5}
```

Merging Multiple Modes for SDC Files

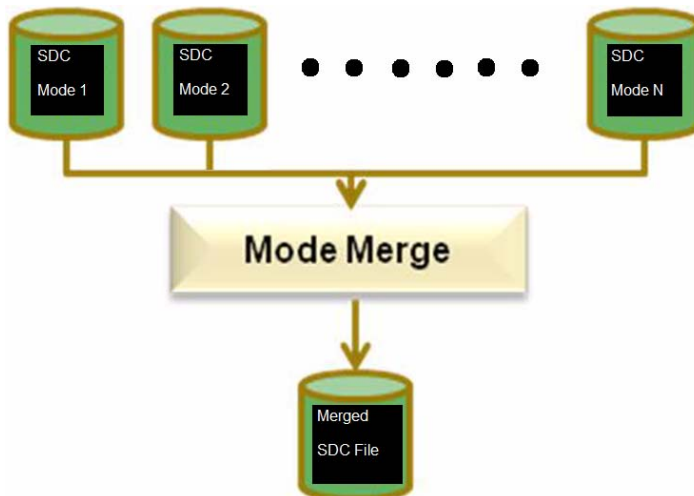
A design has several modes of operation. The timing constraint for each mode is captured in SDC files. For each mode, you optimize the design despite conflicting requirements.

By using the [SDC_ModeMerge](#) rule, you can create a merged SDC file that accounts for the various modes. As a result, you can use the merged SDC file in implementation tools (synthesis, STA, P&R) to save runtime.

The [SDC_ModeMerge](#) rule is applicable to the RTL, Pre-layout, and Post-layout phases.

The [SDC_ModeMerge](#) rule generates a merged mode SDC by merging the SDC constraints defined in multiple different modes. The merging criteria is to never under-constrain the design for the merged SDC file.

The merging of SDC constraints is performed in a specific order as per the type of constraint. In addition, the conflict resolution between the constraints is dependent on the constraint type. To resolve a conflict, a constraint is either retained, ignored or newly generated.



To understand this rule, refer to the following sections:

- [Prerequisites](#)
- [Merging SDC Files](#)

The merged constraints are accompanied by a detailed reporting of the constraints. Refer to the [Reports and Related Files](#) section for a description of each worksheet.

Prerequisites

Ensure the SGDC file is as follows:

```
current_design <design-name>
sdc_data -file <SDC-file-names> -mode <mode-name1>
sdc_data -file <SDC-file-names> -mode <mode-name2>
...
sdc_data -file <SDC-file-names> -mode <mode-nameN>
```

If you do not specify *<mode-name1>*, *<mode-name2>*, or *<mode-nameN>*, by default, the value of the mode argument is taken as *none*. Consequently, it is reflected in the merged SDC file.

Merging SDC Files

A mode is used to constrain an aspect of the design. An SDC might be defined to test only the scan logic of the design. While others might be defined to test another aspect of the design. Therefore, multiple modes might overlap. There could be conflicts in the overlapping constraints. For example, [set_case_analysis](#) could be different since a portion of the design might be disabled using [set_case_analysis](#), but the same portion could be tested under another mode.

To understand the scenarios for merging, you need to understand the effect of an SDC constraint.

An SDC constraint defines a design area to which it applies along with timing values.

If both SDC files do not overlap in terms of design area, it implies there is no conflict in merging.

If there is a design area constrained in both SDC files, conflict may arise as follows:

- Design area constrained by SDC constraint is not exactly same. It overlaps but is unequal.
 - clocks instantiated at different objects
- Number of SDC constraints types involved in each mode

- ❑ Three clocks versus two clocks
- SDC constraints constraining the area are different
 - ❑ *set_input_delay* versus timing exception
- Constraint value is different
 - ❑ *set_case_analysis* value is different

In the merged SDC file, conflict needs to be resolved when creating the merged SDC. These conflicts are resolved so that the effect of the merged SDC file is the same as the cumulative effect of the multiple SDC files.

Since each constraint is unique, conflict resolution is dependent on the constraint types involved.

Merging SDC Files from Lower-level Blocks

The *SDC_MergeBlocks* rule merges the SDC files of multiple blocks into a single SDC file for the combined block.

The *SDC_MergeBlocks* rule requires you to specify the following:

- Complete top-level design in RTL, gate-netlist, or mixed
- SGDC file(s) containing the block boundary definitions (using the *block* constraint)
- The SDC files for each of these blocks

Above is already present in Prerequisites so remove from here.

This rule reads the SDC files of multiple blocks and merges each of them with their corresponding SDC file for the combined block. The SDC constraints are combined for each block and propagated to the pins available on the parent logical block.

This rule creates the merged SDC file.

The merged SDC file is named as per the following conventions:

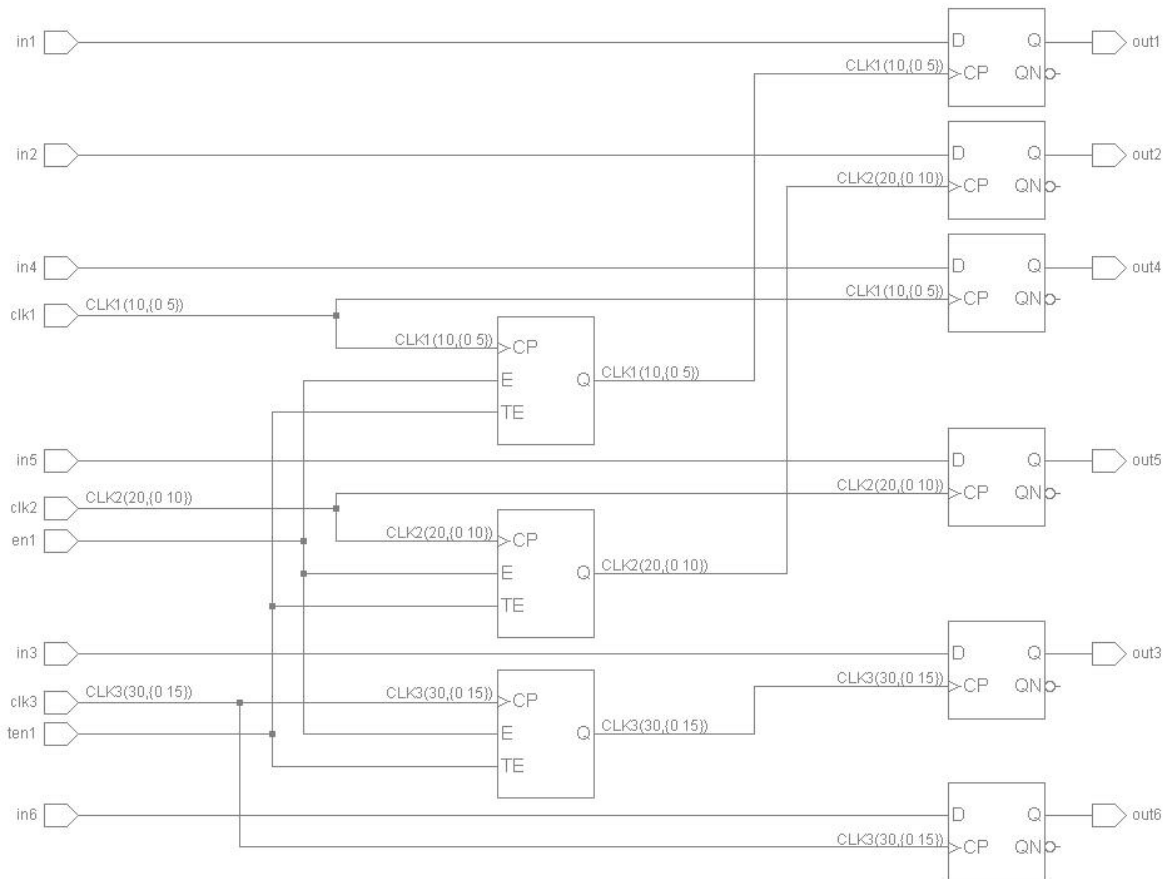
- If both corner and mode are specified:
`<top-level-block-name>_<mode>_<corner>.sdc`
- If only corner is specified:
`<top-level-block-name>_<corner>.sdc`

- If only mode is specified:
`<top-level-block-name>_<mode>.sdc`
- If neither corner nor mode is specified:
`<top-level-block-name>.sdc`

Using SDC Autofix

The SDC Autofix feature uses an `sg_shell` flow to generate an SDC file from violation messages reported by rules in the SpyGlass Constraints solution. Currently, the [Inp_De101a](#) rule is the only supported rule. The `autofix_sdc` Tcl command is used to generate the SDC file.

This section describes the steps for using the SDC Autofix feature. The example is based on the following design.



To generate the SDC file, perform the following steps:

1. Run the *Inp_DeI01a* rule and view the violation messages in Console GUI or in the *moresimple* report. The following screen grab shows the *moresimple* report. The report contains the violation messages.

Constraints Generation

```
##### Non-BuiltIn -> Goal-test_goal -> Constraints Mode-top@seed #####
+++++
ID      Rule      Alias      Severity  File      Line  Wt  Message
-----
[3]     Inp_Del01a    Warning   constraints.sdc  1      5   Input delay constraint not set on Port "in1" of design/block top and input d
elay needs to be specified with clock(s) CLK1 [Also File(Lines): top.v(3)]
[9]     Inp_Del01a    Warning   constraint3.sdc  1      5   Input delay constraint not set on Port "in4" of design/block top and input d
elay needs to be specified with clock(s) CLK1 [Also File(Lines): top.v(3)]
[F]     Inp_Del01a    Warning   constraints.sdc  1      5   Input delay constraint not set on Port "ten1" of design/block top and input
delay needs to be specified with clock(s) CLK1, CLK2, CLK3 with -add_delay [Also File(Lines): top.v(4), constraints.sdc(2,3)]
[11]    Inp_Del01a    Warning   constraints.sdc  1      5   Input delay constraint not set on Port "en1" of design/block top and input d
elay needs to be specified with clock(s) CLK1, CLK2, CLK3 with -add_delay [Also File(Lines): top.v(4), constraints.sdc(2,3)]
[5]     Inp_Del01a    Warning   constraints.sdc  2      5   Input delay constraint not set on Port "in2" of design/block top and input d
elay needs to be specified with clock(s) CLK2 [Also File(Lines): top.v(3)]
[B]     Inp_Del01a    Warning   constraints.sdc  2      5   Input delay constraint not set on Port "in5" of design/block top and input d
elay needs to be specified with clock(s) CLK2 [Also File(Lines): top.v(3)]
[7]     Inp_Del01a    Warning   constraints.sdc  3      5   Input delay constraint not set on Port "in3" of design/block top and input d
elay needs to be specified with clock(s) CLK3 [Also File(Lines): top.v(3)]
[D]     Inp_Del01a    Warning   constraints.sdc  3      5   Input delay constraint not set on Port "in6" of design/block top and input d
elay needs to be specified with clock(s) CLK3 [Also File(Lines): top.v(3)]
[4]     Inp_Del01b    Warning   constraints.sdc  1      5   Input delay constraint not set on Port "in1" of design/block top and input d
elay needs to be specified with clock(s) CLK1 [Also File(Lines): top.v(3)]
[A]     Inp_Del01b    Warning   constraints.sdc  1      5   Input delay constraint not set on Port "in4" of design/block top and input d
elay needs to be specified with clock(s) CLK1 [Also File(Lines): top.v(3)]
[10]    Inp_Del01b    Warning   constraints.sdc  1      5   Input delay constraint not set on Port "ten1" of design/block top and input
delay needs to be specified with clock(s) CLK1, CLK2, CLK3 with -add_delay [Also File(Lines): top.v(4), constraints.sdc(2,3)]
[12]    Inp_Del01b    Warning   constraints.sdc  1      5   Input delay constraint not set on Port "en1" of design/block top and input d
elay needs to be specified with clock(s) CLK1, CLK2, CLK3 with -add_delay [Also File(Lines): top.v(4), constraints.sdc(2,3)]
[6]     Inp_Del01b    Warning   constraints.sdc  2      5   Input delay constraint not set on Port "in2" of design/block top and input d
elay needs to be specified with clock(s) CLK2 [Also File(Lines): top.v(3)]
[C]     Inp_Del01b    Warning   constraints.sdc  2      5   Input delay constraint not set on Port "in5" of design/block top and input d
elay needs to be specified with clock(s) CLK2 [Also File(Lines): top.v(3)]
[8]     Inp_Del01b    Warning   constraints.sdc  3      5   Input delay constraint not set on Port "in3" of design/block top and input d
elay needs to be specified with clock(s) CLK3 [Also File(Lines): top.v(3)]
[E]     Inp_Del01b    Warning   constraints.sdc  3      5   Input delay constraint not set on Port "in6" of design/block top and input d
elay needs to be specified with clock(s) CLK3 [Also File(Lines): top.v(3)]
+++++
```

2. Launch `sg_shell`.
3. Run the `autofix_sdc` Tcl command. The following screen grab shows various inputs to the `autofix_sdc` Tcl command and the output message. After the command is successfully executed, the file name and path of the generated SDC file is displayed.

```
sg_shell> autofix_sdc Inp_Del01b
autofix_sdc: error: This rule is not supported. Please check User Guide.
sg_shell> autofix_sdc
autofix_sdc: error: Please specify some valid rule name.
sg_shell> autofix_sdc Inp_Del01a
autofix_sdc.sdc already exists. Use -f option for overwriting
sg_shell> autofix_sdc Inp_Del01a -f
out/test/test_goal/spyglass_reports/constraints/autofix_sdc.sdc is successfully generated.

sg_shell> █
```

Refer to the *SpyGlass Tcl Shell Interface User Guide* for more information on the `autofix_sdc` Tcl command.

4. Open the generated SDC file in a vi editor. The file name and location is stated in sg_shell. The naming convention and location is as follows:

File Name: autofix_sdc.sdc

File Location: spyglass_reports/constraints/

5. Review the generated SDC file for completeness.

```
#####  
# Created by SpyGlass AutoFix_SDC  
# SDC is generated for rule (Inp_Del01a)  
#####  
  
# Constaints generated corresponding to Inp_Del01a rule  
set_input_delay ? -clock [get_clocks { CLK1 }] [ get_ports { in1 } ] -add_delay  
set_input_delay ? -clock [get_clocks { CLK2 }] [ get_ports { in2 } ] -add_delay  
set_input_delay ? -clock [get_clocks { CLK3 }] [ get_ports { in3 } ] -add_delay  
set_input_delay ? -clock [get_clocks { CLK1 }] [ get_ports { in4 } ] -add_delay  
set_input_delay ? -clock [get_clocks { CLK2 }] [ get_ports { in5 } ] -add_delay  
set_input_delay ? -clock [get_clocks { CLK3 }] [ get_ports { in6 } ] -add_delay  
set_input_delay ? -clock [get_clocks { CLK1 }] [ get_ports { ten1 } ] -add_delay  
set_input_delay ? -clock [get_clocks { CLK2 }] [ get_ports { ten1 } ] -add_delay  
set_input_delay ? -clock [get_clocks { CLK3 }] [ get_ports { ten1 } ] -add_delay  
set_input_delay ? -clock [get_clocks { CLK1 }] [ get_ports { en1 } ] -add_delay  
set_input_delay ? -clock [get_clocks { CLK2 }] [ get_ports { en1 } ] -add_delay  
set_input_delay ? -clock [get_clocks { CLK3 }] [ get_ports { en1 } ] -add_delay
```

6. Update the generated SDC file by replacing the question marks (?) with values.

Timing Exception Verification

A critical part of verifying SDC correctness is verifying that timing exceptions (false path and multicycle path) are functionally valid. To do this effectively requires formal verification techniques and is covered by the SpyGlass TXV solution.

You should ask your SpyGlass Support team for more information.

Constraints Management

The Constraints Management rules help you manage constraints at different stages of the design. In the current version, the SpyGlass Constraints solution checks for equivalence between two SDC files specified for the same design.

To understand how the SpyGlass Constraints solution checks for equivalence, read the following sections:

- [Constraints Management Rules](#)
- [Understanding Equivalence of SDC Files](#)
- [List of Supported SDC Constraints for Equivalence](#)
- [Understanding the Equivalence Flow](#)
- [Specifying the SDC File to Check SDC Equivalence](#)
- [Specifying the Skeleton SDC File to Check SDC Equivalence](#)
- [Checking Equivalence between the Same SDC Commands](#)

Constraints Management Rules

The Constraints Management Rules that perform equivalence are listed below:

Rule	Description
ConstMgmtParamSanityCheck	Sets up syntactic checks on parameters
Equiv_SDC	Performs equivalence between two SDC files
Equiv_SDC_Block	Performs equivalence between the top-level and block-level design units
Equiv_SDC_Dual_Design	Performs equivalence between two SDC files of two equivalent designs
Equiv_SDC_Top	Performs equivalence between the top-level design unit

Apart from these rules, the [Equiv_SDC_Auto_Detect](#) rule is used to auto detect equivalent ports and/or sequential elements between two designs.

Understanding Equivalence of SDC Files

SDC files are used and modified at various stages of a design, such as RTL, pre-layout, and post-layout. When a design completes one stage and enters the second stage a new SDC file is created (either manually or automatically) to set the constraints appropriate for that stage. Therefore, it is important to ensure that the SDC files retain the design intent that has been verified in the previous stage.

In addition, the SDC files at the top and block levels are generated from one another. The block files are generated from the existing top files (top-down approach) or vice-versa (bottom-up approach). Since most of this generation work happens manually, it is error-prone and therefore there is a need to perform an equivalence check for these SDC files.

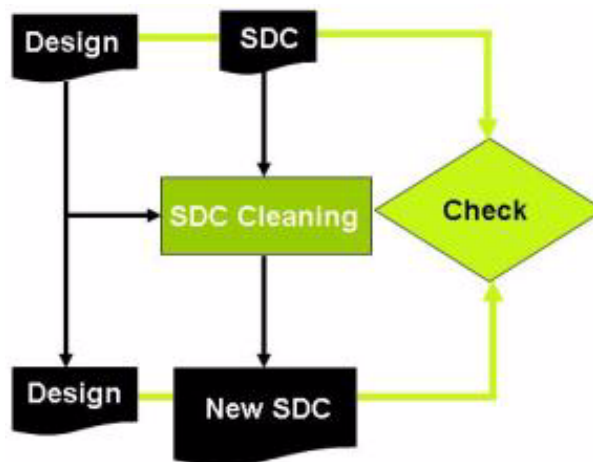
When the constraints of a block are propagated back to the tops' glue logic or top's input port, since the complete top glue constraints are given, they should match with the corresponding constraints specified at the top level. Therefore, equivalence check for missing or non-matching constraints across top/block hierarchy is required.

The *Equiv_SDC* rule checks whether the two SDC files of the same design are equivalent before and after the modification.

The *Equiv_SDC_Block* rule checks whether the two SDC files of block and top design unit are equivalent.

The *Equiv_SDC_Top* rule checks whether the two SDC files of the same design are equivalent for top and top glue constraints.

The *Equiv_SDC_Dual_Design* rule performs the equivalence check between two SDC files of two equivalent designs.



An SDC file may be changed for the following:

- Removing duplicate or redundant constraints
- Removing the `disable_timing` constraints and replacing them with the `false_path` constraints.
- Clustering similar constraints
- Expanding wildcards so that different constraints can be applied to each path.

The SpyGlass Constraints solution rules check that the SDC files before and after the transformation are equivalent for the following:

- The intent of the constraint is preserved between the newer and older version of the SDC
- There are additional constraints present in the newer version
- There are missing constraints in the newer version
- There is a difference between the newer and older version

The SpyGlass Constraints solution assumes that the SDC files being compared for equivalence are clean. This means that the SDC files do not contain any syntax errors and the overwritten/duplicate constraints have been removed.

List of Supported SDC Constraints for Equivalence

All *Constraints Management Rules* support the following SDC constraints for equivalence checking.

set_case_analysis	create_clock
create_generated_clock	set_clock_groups
set_clock_latency	set_clock_sense set_clock_sense is considered for clock propagation but is not represented in a separate spreadsheet.
set_clock_transition	set_clock_uncertainty
set_disable_timing	set_input_delay
set_output_delay	set_propagated_clock
set_false_path	set_max_delay
set_min_delay	set_multicycle_path
set_drive	set_driving_cell
set_input_transition	set_load
set_max_fanout	set_max_transition
set_dont_touch	set_max_capacitance
set_ideal_network	set_sense

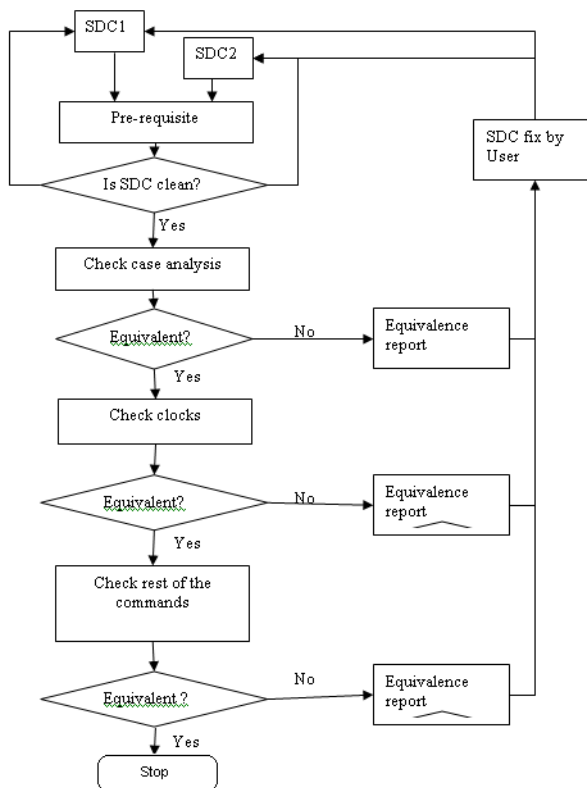
For constraint description and syntax of these constraints, see [Appendix: SDC Constraints](#).

Understanding the Equivalence Flow

Before performing the equivalence, first ensure that the SDC files are clean and do not contain duplicate or overwritten constraints.

You can specify the SDC commands in any order for which you want to perform equivalence checking. However, the equivalence check is first performed on commands that need to be equivalent before the commands specified by you.

The following flowchart displays the flow in which the equivalence check is performed:



Specifying the SDC File to Check SDC Equivalence

You use the `sdc_data` keyword of SGDC to specify the SDC files. The SGDC file created has the following structure for a design unit:

```

current_design <du-name>
  sdc_data
    -file <file-list>
    [ -level <level> ]
    [ -mode <mode> ]

block
  -name <name-list>
  
```


Where,

`<du-name>`

The top-level module name (for Verilog designs) or the top-level entity name (for VHDL designs) or a synthesis partition name specified using the `block` keyword.

`-file <file-list>`

Specifies the list of associated SDC files.

All following arguments are attached to these SDC files to qualify the scope of these SDC files.

`-level <level>`

(Optional) Specifies the scope of the SDC File type by design phase. Specify one of `rtl`, `prelayout`, `postlayout`, or `all`. If omitted, the current SDC file applies to all design phases.

`-mode <mode>`

Specifies the mode name to associate the constraints with the mode the design is in. The only purpose of the `-mode` argument is to differentiate one mode from the other.

For the `Equiv_SDC` rule, mode can either be `reference` (original SDC file) or `implement` (the SDC file after transformation).

For the `Equiv_SDC_Top` rule, mode can be `glueTop` or `flatTop`.

`glueTop` is a flattened view of the top-level design with only the interface-level instantiations of block-level modules. `flatTop` is a flattened view of the top-level design including all the blocks.

In the `flatTop` SDC file, you can refer to any point in the design, including points inside a block. However, in the `glueTop` SDC file, you can refer to only those points that lie outside the block and on the block interface.

For the `Equiv_SDC_Block` rule mode can be `block` for the block design unit, `flatTop` for the top-level design unit, and `skeleton` for the skeleton

SDC file. Refer to [Specifying the Skeleton SDC File to Check SDC Equivalence](#) for details.

NOTE: *You cannot modify the `-mode` argument for the `Equiv_SDC`, `Equiv_SDC_Top`, and `Equiv_SDC_Block` rules. If you specify the `-mode` argument with any other option, the SDC file will not be considered for the equivalence check.*

For the `Equiv_SDC_Dual_Design` rule, mode can either be `reference` (original design and the SDC file) or `implement` (transformed design and the SDC file).

-name *<name-list>*

(Optional) Specifies the sub-blocks that are synthesis partitions if the current design is actually the design top-level.

Each name in *<name-list>* should name a module (for Verilog) or entity (for VHDL) appearing in the design that should be treated as a top-level partition. More than one block constraints may appear, or all blocks may be specified in one constraint.

NOTE: *The design top is also considered to be a block and all block-level rules are run on it unless explicitly mentioned otherwise.*

Specifying the Skeleton SDC File to Check SDC Equivalence

The Skeleton SDC file is an optional file that is used to overwrite the constraints present at the block-level SDC file (specified using the mode `block`). The skeleton SDC file is specified using `skeleton` as the mode name.

NOTE: *The skeleton SDC file support is currently available for the `Equiv_SDC_Block` rule only.*

Example

Consider a design that has a top-level design unit `top` and a block-level design unit `block` is instantiated inside `top`. Let the SDC file for the top-level design unit be `top.sdc`, block-level design unit be `block.sdc`, and the skeleton SDC file be `skeleton.sdc`.

The SGDC file for the above is as follows:

```
current_design top
```

```

sdc_data -file top.sdc -mode flatTop
sdc_data -file skeleton.sdc -mode skeleton
block -name block

current_design block
sdc_data -file block.sdc -mode block

```

In the above example, the constraints in the file `skeleton.sdc` will overwrite the constraints in the file `block.sdc`.

The constraints present in the skeleton SDC file overwrite the constraints present in the block-level SDC based on the following:

- [Real Clock Matching](#)
- [Timing Exceptions Matching](#)
- [Matching of Other Commands](#)

Real Clock Matching

A real clock in the skeleton SDC overwrites a clock in the block SDC, if either of the following first two conditions and the third condition is satisfied for a clock port/pin:

1. Only one clock is set on the port/pin in both skeleton and block
2. If more than one clock is specified in a port/pin, the matching clock should be using either the [equiv_sdc_clk_suffix](#) parameter or the [equiv_sdc_ambiguous_clock_file](#) parameter
3. Clock characteristics in skeleton and block are different

Example 1

skeleton SDC

```
create_clock -name S_CLK -period 10 -waveform { 0 5}
[get_port clk]
```

block SDC

```
create_clock -name B_CLK -period 10 -waveform { 0 5}
[get_port bclk]
```

In this example, the clock `S_CLK` in skeleton SDC matches with the clock `B_CLK` in block SDC since there is only one clock specified at port `clk` in the skeleton SDC and port `bclk` in block SDC. However, the [create_clock](#)

constraint in skeleton SDC will not overwrite [create_clock](#) in block SDC since their characteristics are the same.

Example 2

skeleton SDC

```
create_clock -name S_CLK1 -period 11 -waveform { 0 5}
[get_port clk]
create_clock -name S_CLK2 -period 11 -waveform { 0 5}
[get_port clk] -add
```

block SDC

```
create_clock -name B_CLK -period 10 -waveform { 0 5}
[get_port bclk]
```

In this example, clock S_CLK1 or S_CLK2 in skeleton SDC does not match with the clock B_CLK in block SDC since there are multiple clocks specified at port `clk` in skeleton SDC. Since, neither the [equiv_sdc_clk_suffix](#) nor [equiv_sdc_ambiguous_clock_file](#) parameter is used for the clock specified in block SDC, the clock in skeleton SDC will not overwrite the clock in block SDC.

Timing Exceptions Matching

The timing exception commands are matched based on the following:

- If all timing exceptions points for a command in skeleton SDC are design objects, the command is added to the block timing exception only if they belong to the block, that is, the hierarchy also points to block.
- If any timing exception points for a command in the skeleton SDC are real clock, the timing exception command is added to the block SDC only if the clock has matching clock in the block SDC (refer to the [Real Clock Matching](#) section).

Matching Based on Design Objects

It is assumed that all `-from`, `-through`, and `-to` options in the skeleton SDC and block SDC are equivalent unless specified.

Example 1

skeleton SDC

```
set_false_path -from FD1/CP -to FD2/D
```

block SDC

```
//no set false path is specified
```

In this example, the `set_false_path` constraint from skeleton SDC will be added to block SDC, if the path in question is inside block.

Example 2

skeleton SDC

```
set_false_path -from b/FD1/CP -to b/FD2/D
```

block SDC

```
set_false_path -from FD1/CP -to FD2/D
```

In this example, since the paths are equivalent and the options are also same, the skeleton exception will overwrite the block exception.

Matching Based on Real Clocks

Example 1

skeleton SDC

```
create_clock -name CLK1 - period 10 [get_port clk]
create_clock -name CLK2 - period 10 [get_port clk2]
set_multicycle_path 2 -from CLK1 -to CLK2
```

block SDC

```
create_clock -name CLK1 - period 10 [get_port clk]
create_clock -name CLK2 - period 10 [get_port clk2]
set_multicycle_path 3 -from CLK1 -to CLK2
```

In this example the clock in the skeleton SDC matches the clock in block SDC, therefore, the multi cycle path from skeleton SDC will overwrite the multi-cycle path in block SDC.

Example 2

skeleton SDC

```
create_clock -name CLK1 - period 10 [get_port clk]
create_clock -name CLK2 - period 10 [get_port clk2]
set_multicycle_path 2 -from CLK1 -to CLK2 -through A/Z
```

block SDC

```
create_clock -name CLK1 - period 10 [get_port clk]
create_clock -name CLK2 - period 10 [get_port clk2]
set_multicycle_path 3 -from CLK1 -to CLK2
```

In this example, the clock in the skeleton SDC match the clock in block SDC. However, their paths do not match. Therefore, the multi cycle path from skeleton SDC will be added to block SDC.

Matching Based on Virtual Clocks

Example

skeleton SDC

```
create_clock -name CLK1 - period 10 [get_port clk]
create_clock -name CLK2_V - period 10
set_output_delay 3 -clock [get_clock CLK2_V] [get_port out1]
set_multicycle_path 2 -from CLK1 -to CLK2_V
```

block SDC

```
create_clock -name CLK1 - period 10 [get_port clk]
create_clock -name CLK2 - period 10 [get_port clk2]
set_output_delay 3 -clock [get_clock CLK2] [get_port out1]
```

In this example, the multi cycle path in skeleton SDC has a virtual clock CLK2_V that matches with the CLK2 in block SDC. Therefore, the multi cycle path in skeleton SDC will get added to block SDC.

Matching of Other Commands

For the [set_clock_latency](#), [set_clock_uncertainty](#), [set_clock_transition](#), [set_input_transition](#), and [set_load](#) commands, the value in the skeleton SDC will overwrite the value in block SDC based on matching clock and design object.

NOTE: *If a command is present in skeleton SDC but is absent in block SDC, it will be ignored.*

Example 1

skeleton SDC

```
create_clock -name CLK -period 10 -waveform { 0 5 } [get_port clk1]
```

```
set_clock_latency 3 [get_clock CLK]
set_clock_latency 4 -source [get_clock CLK]
```

block SDC

```
create_clock -name CLK -period 10 -waveform { 0 5} [get_port
clk1]
set_clock_latency 5 [get_clock CLK]
```

In this example, the network latency in skeleton SDC will overwrite the network latency in block SDC. The source latency in skeleton SDC will be ignored since there is no source latency in block SDC with the matching clock.

Example 2

skeleton SDC

```
create_clock -name CLK1 -period 10 -waveform { 0 5} [get_port
clk1]
create_clock -name CLK2 -period 10 [get_port clk2]
set_clock_uncertainty 3 [get_clock CLK1]
set_clock_uncertainty 4 -from [get_clock CLK1] -to [get_clock
CLK2]
```

block SDC

```
create_clock -name CLK1 -period 10 -waveform { 0 5} [get_port
clk1]
create_clock -name CLK2 -period 10 [get_port clk2]
set_clock_uncertainty 5 [get_clock CLK1]
set_clock_uncertainty 3 -from [get_clock CLK1 ] -to
[get_clock CLK2]
```

In this example, clock uncertainty in skeleton SDC will overwrite clock uncertainty in block SDC.

Example 3

skeleton SDC

```
create_clock -name CLK -period 10 -waveform { 0 5} [get_port
clk1]
set_clock_transition 3 [get_clock CLK] -rise
```

block SDC

```
create_clock -name CLK -period 10 -waveform { 0 5} [get_port
clk1]
set_clock_transition 5 [get_clock CLK]
```

In this example, only the `-rise` option of clock transition in skeleton SDC will overwrite the clock transition in block SDC.

Example 4

skeleton SDC

```
set_input_transition 4 in1
```

block SDC

```
set_input_transition 3 in1
```

In this example, all the options of input transition in skeleton SDC will overwrite the input transition in block SDC.

Example 5

skeleton SDC

```
set_load 3 [get_port out] -pin_load
```

block SDC

```
set_load 4 [get_port out] -wire_load
```

In this example, the load set on the port out in the skeleton SDC will be ignored since there is no pin load set in block SDC.

Checking Equivalence between the Same SDC Commands

The criteria for equivalence between two commands is as follows:

- Two constraints are equivalent if the application of a constraint in one SDC file results into the same effect as the application of the same or different constraint in the second SDC file on the same design.
- Two commands are equivalent when the options corresponding to these commands (such as `-rise/-fall`, `-max/-min`, `-setup/-hold`) also match, either explicitly or implicitly.

Example

Suppose the a.sdc file has the following code:

```
create_clock -name C1 -period 10
set_input_delay 3 -clock C1 -rise in1
set_input_delay 3 -clock C1 -fall in1
```

Suppose file b.sdc has the following code:

```
create_clock -name Clk -period 10
set_input_delay 3 -clock Clk in1
```

In the above example, the input delay specified in the file a.sdc is equivalent to the input delay specified in the file b.sdc, even though the input delay has been specified using two commands in the file a.sdc and one command in the file b.sdc.

- If the same constraint is set on an object (such as a pin) in one SDC file and set on an equivalent object (such as a net connected to the pin) in the second SDC file, then they are equivalent.

Goals

The following goals are available in the RTL submethodology:

- **SDC_QuickCheck:** Finds any syntax errors in the SDC
- **Gen_Constraints:** Creates SDC goals from RTL or Netlist
- **Consis_n_Clk_Def:** Checks for basic consistency and clean clock definition
- **Delay_Checks:** Checks input/output delay constraints
- **Combo_Path_Checks:** Checks combinational/feed-through path issues
- **Exceptions_Structural:** Checks timing exceptions structurally
- **Hierarchical_Checks:** Checks hierarchical constraints
- **Retargeting_Checks:** Checks for constraints that can help in better re-targeting
- **Uncommon_Timing:** Checks for certain uncommon timing requirements
- **Redundancy_Checks:** Checks for redundant or potentially redundant constraints

Rules in SpyGlass Constraints

The SpyGlass Constraints solution has the following types of rules:

- *Block Rules*
- *Clock Rules*
- *Clock Group Rules*
- *Display Information Rules*
- *Domain Rules*
- *Down Streaming Rules*
- *I/O Rules*
- *Methodology Rules*
- *Miscellaneous Rules*
- *Other Rules*
- *Constraints Generation*
- *Reporting Rules*
- *Setup Rules*
- *SpyGlass Design Constraints File Structure Rules*
- *Test Rules*

- *Timing Checking Rules*
- *Timing Coverage Rules*
- *Timing Exception Rules*
- *Abstract View Rules*
- *Merge SDC Files*
- *Constraints Management Rules*

Block Rules

The Block Rules Group `BlockRules` contains the following rules:

Rule	Description
<i>Block02</i>	Combinational loops that cross block boundaries
<i>Block05</i>	Combinational paths containing significant logic between two blocks
<i>Block06</i>	Nets that have multiple fanouts without dedicated buffering
<i>Block10</i>	Combinational loops that are not cut by a path with <i>set_disable_timing</i> constraint set
<i>Block11</i>	Block inputs and outputs that are not registered
<i>Block12</i>	Reports a violation if the combinational logic is lesser than the specified value
<i>Block13</i>	Reports ports driven by clocks through combinational feedthrough paths

Block02

Identifies combinational loops that cross block boundaries

When to Use

Use this rule in the RTL and Pre-layout phases of design.

Description

The *Block02* rule identifies combinational loops that cross block boundaries. The *Block02* rule reports a violation if a net, which is a part of the loop, crosses the block boundary even when the entire combinational loop lies inside a block.

Prerequisites

Ensure the SGDC file contains at least two blocks as shown in the following:

```
current_design top
sdc_data -file top.sdc
block -name block1 block2
```

Rule Exceptions

The *Block02* rule checks for partitions specified with the block constraints only; it does not check for top designs.

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

The following message appears for a combinational loop *<loop-list>* that crosses block boundaries:

```
[WARNING] Combinational loop crosses block boundaries (Path
'<loop-list>')
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

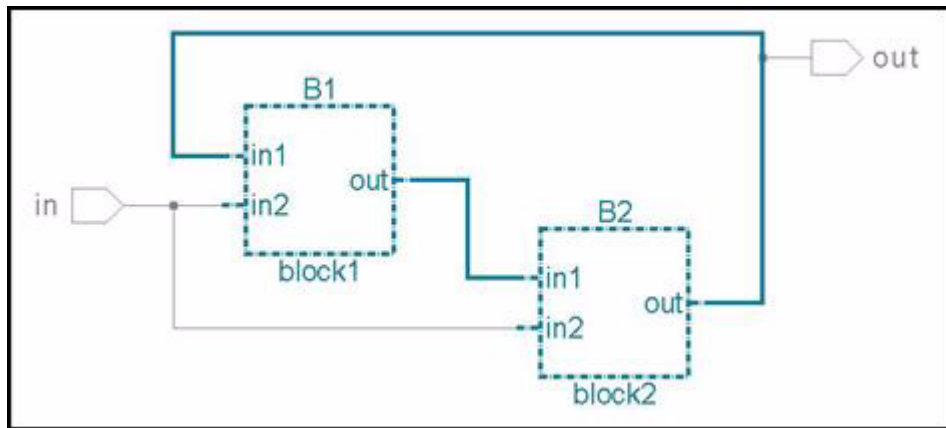
The presence of combinational loop across blocks results in unpredictable timing because inter-partition paths may need to be routed at the global level. In addition, synthesis tools may not optimize such paths.

How to Debug and Fix

To resolve this violation, shift the loop so that it is in a block instead of spanning over multiple blocks.

Example Code and/or Schematic

For the following design, the *Block02* rule reports a violation because the loop extends to multiple blocks.



Default Severity Label

Warning

Rule Group

BlockRules

Reports and Related Files

None

Block05

Reports significant combinational paths detected between blocks

When to Use

To check for significant combinational paths between blocks. This rule is applicable to the RTL phase.

Description

The *Block05* rule reports combinational paths containing significant logic between two blocks. The *Block05* rule counts the combinational logic between the register driving the sender block output port and the register driven by the receiver block's input port. By default, this rule reports combinational paths between two blocks that have more than 200 logic levels.

Refer to the [Parameter\(s\)](#) section to learn how to control the behavior of this rule.

This rule description contains the following sections:

- [Prerequisites](#)
- [Logic Gate Delay Depth Computation](#)
- [Correlation between Reported and Actual Logic Depth Values](#)
- [Rule Exceptions](#)

Prerequisites

The *Block05* rule requires the synthesizable description of all library cells. Therefore, use the SpyGlass Library Compiler feature before running this rule. Refer to the *SpyGlass Explorer User Guide* for details on the SpyGlass Library Compiler feature.

In addition, ensure the SGDC file contains:

```
current_design top
sdc_data -file top.sdc
block -name block1 block2
```

Logic Gate Delay Depth Computation

The delay value of all types of SpyGlass-compatible library cell instances (instances of cells from the .sglib files) is assumed to be 1. The *Block05* rule

calculates the delay depths of each type of logic gate as described in the following table:

Gate Type	Delay	Associated Rule Parameter
Buffer	1	delay_Buf
Tristate buffer	1	delay_Tribuf
Inverter	1	delay_Inv
N-input AND	$\log(\$size)/\log(2)$	delay_And
N-input NAND	$\log(\$size)/\log(2)$	delay_Nand
N-input OR	$\log(\$size)/\log(2)$	delay_Or
N-input NOR	$\log(\$size)/\log(2)$	delay_Nor
N-input XOR	$(\$size/4)+1$	delay_Xor
N-input XNOR	$(\$size/4)+1$	delay_Xnor
MUX with N select lines of which only one is active at a time and N inputs	2	delay_Mux
LogN MUX with N select lines and 2^N inputs	$(\$size/4)+1$	delay_LogN_mux
Decode operation, decoding from N bits to 2^N bits	$\log(\$size)/\log(2)+1$	delay_Decoder
Latch	2	delay_Latch
Unsigned full-adder with 2 N-bit inputs, plus carry-in and N+1-bit output plus carry-out	$\$size*2$	delay_Add_unsigned
Binary subtract (a-b) where a and b are both N-bit	$\$size*2$	delay_Subtract_binary
Unary subtract (-a) where a is N-bit	$\$size*2$	delay_Subtract_unary
Incrementor (a+1) where a is N-bit	$\$size*2$	delay_Increment

Gate Type	Delay	Associated Rule Parameter
Decrementor (a-1) where a is N-bit	$\$size*2$	delay_Decrement
Fixed-point multiplier (a*b) where a and b are both N-bit	$\$size*4$	delay_Multiply_unsigned
Fixed-point multiplier (a*b) where a and b are both N-bit	$\$size*4$	delay_Multiply_signed
Unsigned fixed-point greater-than comparison (a>b) where a and b are both N-bit	$\$size*2$	delay_GT_unsigned
Signed fixed-point greater-than comparison (a>b) where a and b are both N-bit	$\$size*2$	delay_GT_signed
Unsigned fixed-point greater-than/equals comparison (a>=b) where a and b are both N-bit	$\$size*2$	delay_GE_unsigned
Signed fixed-point greater-than/equals comparison (a>=b) where a and b are both N-bit	$\$size*2$	delay_GE_signed
Unsigned fixed-point less-than/equals comparison (a<=b) where a and b are both N-bit	$(\$size/4)*2$	delay_LEQ_unsigned
Signed fixed-point less-than/equals comparison (a<=b) where a and b are both N-bit	$(\$size/4)*2$	delay_LEQ_signed
Equality comparison (a==b) where a and b are both N-bit	$\$size*2$	delay_Equal
Non-equality comparison (a!=b) where a and b are both N-bit	$\$size*2$	delay_Not_equal

Gate Type	Delay	Associated Rule Parameter
Shift operation ($a > > b$ or $a < < b$) where a and b are both N-bit	$\$size * 2$	<code>delay_Shift</code>
Absolute value evaluation operation ($abs(a)$) where a is N-bit	$((\$size - 1) * 2) + 1$	<code>delay_Abs</code>

NOTE: In the above table, $\$size$ is the number of inputs to a gate.

Correlation between Reported and Actual Logic Depth Values

When you synthesize the source design using a synthesis or library, the generated netlist is different from the netlist generated by SpyGlass. Therefore, the logic depth values reported by SpyGlass could be different from those expected based on your synthesized netlist. For better correlation between the reported and actual logic depth values, use the `delay_*` rule parameters to tweak the SpyGlass delay calculations to follow your synthesis tool or library.

For example, SpyGlass defines the logic depth value of an N-input AND gate as $\log_2 N$ assuming that the N-input AND gate would be realized as a cascade of 2-input AND gates. Therefore, SpyGlass realizes an 8-input AND gate as three levels of a 2-input AND gate cascade. However, depending on your synthesis library, the same 8-input AND gate may be realized differently, even as a single 8-input AND gate. Therefore, the expected logic depth value is different. You can account for the difference in realization for AND gates using the `delay_And` parameter and specifying a value that follows your synthesis tool or library more closely.

Rule Exceptions

The *Block05* rule does not consider paths containing ports/pins that are constrained as clocks using [create_clock](#) or [create_generated_clock](#). In addition, it does not consider paths constrained as false paths using [set_false_path](#).

Parameter(s)

- [logic_level_max](#): Default is 200. This means the *Block05* rule flags combinational paths between two blocks that have more than 200 logic levels.

Refer to the [Logic Gate Delay Depth Computation](#) section for rule parameters that impact the calculation of delay depths of each type of logic gate.

Constraint(s)

None

Messages and Suggested Fix

The following message appears when a combinational path from the pin `<pin1-name>` of the block `<blk1-name>` to the pin `<pin2-name>` of another block `<blk2-name>` has `<num>` logic levels and `<num>` exceeds the maximum limit (`$logic_level_max`):

[WARNING] Path from `<blk1-name>` (pin `<pin1-name>`) to `<blk2-name>` (pin `<pin2-name>`) has `<num>` logic levels. This equals/exceeds the allowed maximum limit of `$logic_level_max`

Potential Issues

If there is a large combinational path between the blocks then synthesis tools cannot optimize the logic since it is outside the boundaries of blocks. Either the logic should be pulled inside the blocks or removed to get good results during synthesis.

Consequences of Not Fixing

Synthesis would be under optimized.

How to Debug and Fix

View the incremental schematic of the violation message.

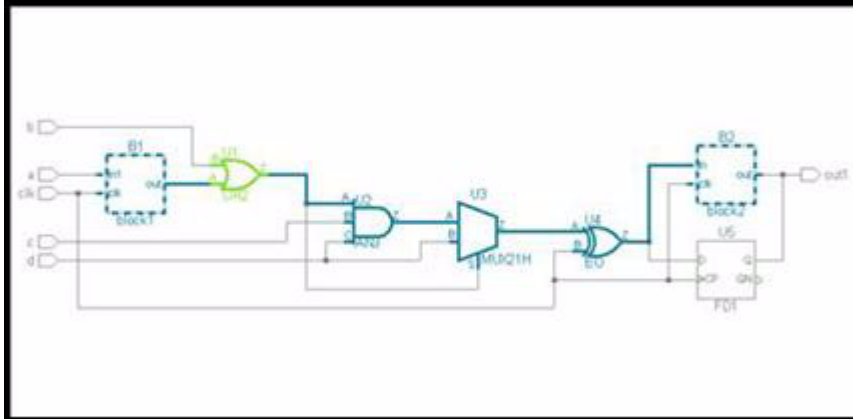
The schematic shows an inter-block combinational path between the sequential elements in the design. The paths that have a logic level greater than the maximum value specified by you or default (200) are shown. The logic levels between the sequential elements are stated in the violation message and you can correlate the logic levels in the schematic. Intra-block combinational paths between the sequential elements are not shown.

To resolve this violation, you can perform any of the following:

- Optimize the design to reduce the logical depth.
- Apply the [set_false_path](#) constraint between flip-flops.
- Increase the allowed logical depth.

Example Code and/or Schematic

In the following example, the *Block05* rule reports a violation because the permitted logical depth between blocks B1 and B2 is one. However, in the design, the depth is more than one.



Default Severity Label

Warning

Rule Group

BlockRules

Reports and Related Files

None

Block06

Reports nets connected to multiple outputs or reused internally without dedicated buffering

When to Use

This rule is applicable to all phases in the design.

Description

The *Block06* rule identifies output nets that have multiple fan-outs, which do not have dedicated buffering. The Block06 rule reports:

- nets that are directly connected to more than one output, or
- fan-outs connected to a single output, which is being used internally.

Following this guideline ensures that good timing modeling can be produced later in the flow.

Prerequisites

Ensure the SGDC file contains at least one block:

```
current_design top
sdc_data -file top.sdc
block -name block1 block2
current_design block1
sdc_data block1.sdc
current_design block2
sdc_data block2.sdc
```

Rule Exceptions

The *Block06* rule ignores the top-level design units specified with the *chip* parameter.

Parameter(s)

- *chip*: Default is unspecified. Set the value to the name of the design unit which corresponds to the chip-level.

Constraint(s)

None

Messages and Suggested Fix

The following message appears when the net *<net-name>* directly connected to output port and/or internal input in block *<blk-name>* has a fan-out of more than one:

[WARNING] Net (*<net-name>*) of block *<block-name>* is connected to more than one output and/or internal input without dedicated buffering

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

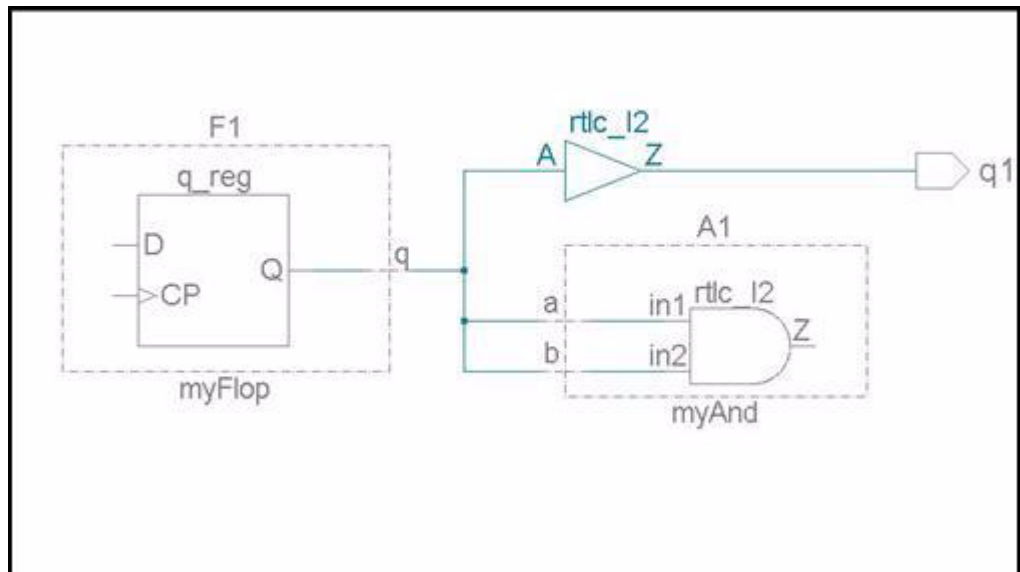
Buffering is required to limit the transition time to acceptable values.

How to Debug and Fix

Apply buffers on high fan-out nets.

Example Code and/or Schematic

For the following design, the *Block06* rule reports a violation because the net of one block is feeding to another block without a dedicated buffer.

**Default Severity Label**

Warning

Rule Group

BlockRules

Reports and Related Files

None

Block10

Reports combinational loops in the design

When to Use

To detect and remove combinational loops in your design. This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Block10* rule detects combinational loops in the design and breaks the loops so that all other rules use this information.

To break the loop, the rule identifies a library cell associated with the combinational loop. If a library cell is identified, the rule applies the [set_disable_timing](#) constraint for the arc of that library cell. If no library cell is identified, the rule breaks the loop, but does not generate the [set_disable_timing](#) constraint.

This rule generates the [set_disable_timing](#) constraint in a separate SDC file, which you can then use this file in the next run.

Prerequisites

Ensure the SGDC file contains:

```
current_design top
sdc_data -file top.sdc
block -name block1 block2
current_design block1
sdc_data block1.sdc
current_design block2
sdc_data block2.sdc
```

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message appears when a combinational loop is identified and [set_disable_timing](#) constraint is generated:

```
[INFO] The set_disable_timing constraints generated in file  
<file-name>
```

Potential Issues

Your design contains a combinational loop.

Consequences of Not Fixing

SpyGlass selects a node in the combinational loop to break it. Similarly, if you are using other design tools, they may have a different node selection to break the combinational loop. Therefore, the results from the design tools may not be consistent.

NOTE: *The results generated by the Block10 rule are used by other rules in the SpyGlass Constraints products in the same run. Review the generated file to understand the impact.*

How to Debug and Fix

The generated SDC file contains the [set_disable_timing](#) constraint applied by this rule. By reviewing the file, you can determine whether the combinational loops identified in the design are as per your expectations. If you need to make changes, identify a node in the combinational loop and apply the [set_disable_timing](#) constraint on that node. Update the original SDC file accordingly.

Message 2

The following message appears if a combinational loop is detected for the design/block *<name>*:

```
[Info] For design/block <name>, combinational loops are  
reported in file <report-file-name>
```

Potential Issues

Your design contains at least one combinational loop.

Consequences of Not Fixing

SpyGlass selects a node in the combinational loop to break it. Similarly, if you are using other design tools, they may have a different node selection to break the combinational loop. Therefore, the results from the design

tools may not be the same.

NOTE: *The results generated by the Block10 rule are used by other SpyGlass Constraints rules. Therefore, do not ignore this message. Rather, review the generated file to understand the impact.*

How to Debug and Fix

Double-click the violation message. A CSV appears. This file contains information about the nodes that are a part of the combinational loop. In the file, each row corresponds to a combinational loop. Select the row and view the incremental schematic associated with the combinational loop.

The CSV file also contains the node at which the *Block10* rule has automatically inserted a [set_disable_timing](#) to break the loop. By reviewing the file, you can determine whether the combinational loops identified in the design are as per your expectations. If you need to make changes, identify a node in the combinational loop and apply the [set_disable_timing](#) constraint on that node. Update the original SDC file accordingly.

Example Code and/or Schematic

In the following example, output pin of instance I1 is assigned back to input pin B of instance I1. This forms a combinational loop. Therefore, the *Block10* rule reports a violation.

```
//test.v
Module top(in, out);
Input in;
Output out;
AN2 I1(.A(in), .B(out), .Z(out));
Endmodule

Module AN2(A,B,Z);
Input A,B;
Output Z;

Assign Z = A & B;
```

endmodule

You can view the information about the nodes that are a part of the combinational loop by opening the CSV file generated, as follows.

Spreadsheet Viewer - top_1_Block10.csv (ReadOnly)

File View Tools Help

Show Header

value=

	B	C	D	E
	S.No.	found	Arc disabled	set_disable_timing constraint internally applied
1	1	I1/Z->	I1/B - I1/Z	set_disable_timing -from B -to Z [get_cells {I1}]

Messages: Displayed: 1 Total: 1

You can also view the `set_disable_timing` constraint that was inserted to break the loop, as follows:

```
top_1_sdt.sdc test.sgdc
1 # set_disable_timing constraints internally applied to break the combina
2 set_disable_timing -from B -to Z [get_cells {I1}]
```

Default Severity Label

INFO

Block Rules

Rule Group

BlockRules

Reports and Related Files

None

Block11

Reports block outputs, inouts, or inputs that are not registered

When to Use

To maximize timing slack available for inter-block routing by identifying block ports that should be registered. This rule is applicable during pre-synthesis development.

Description

The *Block11* rule reports block input, output, and inout ports that are not registered. The *Block11* rule requires that each block input port be directly driving a sequential element ignoring intervening buffers, inverters, and power gating cells. Similarly, each block output or inout port be driven by a sequential element ignoring intervening buffers, inverters, and power gating cells. Power gating cells are library cells with the `is_isolation_cell` attribute set.

Refer to the [Parameter\(s\)](#) section to learn how to control the behavior of this rule.

Prerequisites

Ensure the SGDC file contains:

```
current_design top
sdc_data -file top.sdc
block -name block1 block2
current_design block1
sdc_data block1.sdc
current_design block2
sdc_data block2.sdc
```

Rule Exceptions

The *Block11* rule does not report a violation if your design has:

- clock ports and ports that are in the transitive fan-out of a clock.
- the top-level design units specified with the [chip](#) parameter.
- the buffers and inverters while computing the logic level.

- that have a clock is applied on port.
- if path from/to port is disabled.
- if path from/to port is blocked or constant.

The *Block11* rule considers synchronous reset signals specified in the SGDC file. If a reset signal traverses to the sequential cell by skipping a 2-input combinational gate, no violation is reported for the reset signal. This rule ignores the ports connected to such a 2-input combinational gate.

Parameter(s)

- *tc_combo_check*: Default is no. Set the value to yes to report cases when a combinational path exists between a port and any other port.
- *tc_clk_register*: Default is no. Set the value to yes to specify whether clock ports are registered.
- *tc_io_reg*: Default is output. This indicates the *Block11* rule checks only the block output or inout ports. Set the value to input to check only the block input ports or set it to both to check both block input ports and block output ports.
- *verbose*: Default is no. This indicates the *Block11* rule reports the logic levels driven by or driving an unregistered port. Set the value to yes to identify the logic levels of an unregistered port.
- *tc_ignore_te* (Default value is yes) and *ignore_io_if_fp* (Default value is no). If the value of the *tc_ignore_te* parameter is set to no and the value of the *ignore_io_if_fp* parameter is set to yes, the *set_false_path* constraints are taken into consideration, if specified.

Constraints(s)

None

Messages and Suggested Fix

Message 1

The following message appears when any port <port-name> of the block <name> is unregistered with both *tc_combo_check* and *verbose* parameters set to no:

[WARNING] "<port-type>" port "<port-name>" of Block "<name>" is not registered

Where, <port-type> can be input, output, or inout (in fan-in or in fan-out), and <port-name-list> denotes the ports having combinational path from the unregistered port when the [tc_combo_check](#) parameter is set.

For debugging information, refer to [How to Debug and Fix](#).

Message 2

The following message appears when any port <port-name> of the block <name> is unregistered with the [tc_combo_check](#) parameter set to no and the [verbose](#) parameter set to yes:

[WARNING] "<port-type>" port "<port-name>" of Block "<name>" is not registered, maximum logic level is "<value>"

Where, <port-type> can be input, output, or inout (in fan-in or in fan-out), and <port-name-list> denotes the ports having combinational path from the unregistered port when the [tc_combo_check](#) parameter is set.

For debugging information, refer to [How to Debug and Fix](#).

Message 3

The following message appears when any port <port-name> of the block <name> is unregistered with the [tc_combo_check](#) parameter set to yes and the [verbose](#) parameter set to no:

[WARNING] "<port-type>" port "<port-name>" of Block "<name>" is not registered; has a combinational path to port <port-name-list>

Where, <port-type> can be input, output, or inout (in fan-in or in fan-out), and <port-name-list> denotes the ports having combinational path from the unregistered port when the [tc_combo_check](#) parameter is set.

For debugging information, refer to [How to Debug and Fix](#).

Message 4

The following message appears when any port <port-name> of the block

<name> is unregistered with both *tc_combo_check* and *verbose* parameters set to yes:

[WARNING] "<port-type>" port "<port-name>" of Block "<name>" is not registered, maximum logic level is "<value>"; has a combinational path to port <port-name-list>

Where, <port-type> can be input, output, or inout (in fan-in or in fan-out, and <port-name-list> denotes the ports having combinational path from the unregistered port when the *tc_combo_check* parameter is set.

Potential Issues

The *Block05* rule reports ports, governed by parameter *tc_io_reg*, which are not registered. This message is reported when the *verbose* and *tc_combo_check* parameters are set to no.

Consequence of Not Fixing

Inputs and outputs should be registered to maximize timing slack available for inter-block routing.

How to Debug and Fix

View the incremental schematic of the violation message.

If an input port is being reported as not registered, the schematic shows the combinational elements through which the input signal traverses before the signal enters a register.

If an output port is being reported as not registered, the schematic shows the combinational elements through which the output signal of a register traverses before reaching the output port.

If a combinational path is being reported, the schematic shows the traversed path from an input port to the output port. The traversed path is established without encountering any register.

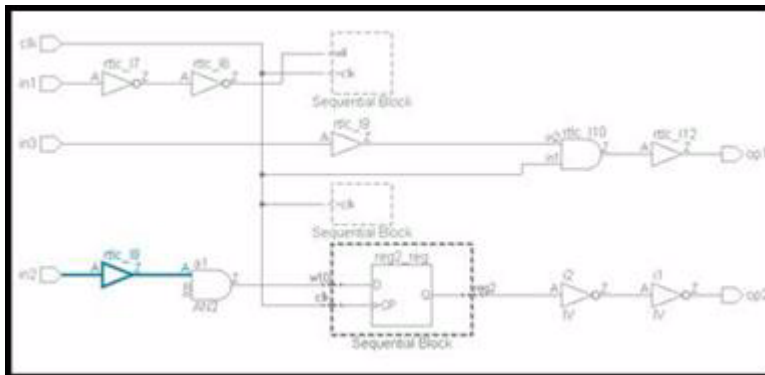
If you are expecting a specific port to be reported, but it is not reported, view *tc_block11_info Report*. This report contains reasons for not reporting specific ports.

To resolve this violation, register the unregistered ports.

Example Code and/or Schematic

Example 1

As shown in the following schematic, the *Block11* rule reports a violation for the unregistered port *in2*. This rule does not report a violation for the *in1* port because inverters/buffers are ignored. In addition, the *in1* port feeds into the *w4* block port directly, which means that the *in1* port is registered.



Example 2 - Understanding False Path Support

These examples illustrate how the *Block11* rule supports false path for combinational paths. To consider false paths, ensure the following parameters are set:

- *tc_ignore_te* and *ignore_io_if_fp*: If the value of the *tc_ignore_te* parameter is set to *no* and the value of the *ignore_io_if_fp* parameter is set to *yes*, the *set_false_path* constraints are taken into consideration, if specified.
- *tc_combo_check*: Set the value to *yes* to report cases when a combinational path exists between a port and any other port.

For the following snippet, the *Block11* rule does not report a violation because all combinational paths are false.

```
//test.sdc
create_clock -name c1 -period 20 clk
set_input_delay 5 -clock c1 in1
set_output_delay 10 -clock c1 out2
```

```
set_false_path -thr I1/Y
```

For the following snippet, the *Block11* rule reports a violation for the in2 - out2 path because the in1 - out2 path is a false path.

Similarly, for the following snippet, the *Block11* rule reports a violation for the in2 - out2 path because the in1-out2 path is a false path.

```
//test.sdc  
create_clock -name c1 -period 20 clk  
set_input_delay 5 -clock c1 in1  
set_output_delay 10 -clock c1 out2  
set_false_path -from in1 -thr I1/IN1 -thr I1/Y
```

Default Severity Label

Warning

Rule Group

BlockRules

Reports and Related Files

- [tc_block11_info Report](#)

Block12

Reports a violation if the combinational logic is lesser than the specified value

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Block12* rule reports a violation message if the level of combinational logic between the following is less than the value specified in the [logic_level_min](#) parameter:

- Between two sequential cells
- Between an input port and a sequential cell
- Between a sequential cell and an output port
- Between an input port and an output port

Refer to [Example 1 - Illustrates when Block12 reports a violation](#).

Prerequisites

Ensure the SGDC file contains:

```
current_design top
sdc_data -file top.sdc
block -name block1 block2
current_design block1
sdc_data block1.sdc
current_design block2
sdc_data block2.sdc
```

Rule Exceptions

The *Block12* rule does not report a violation if your design has:

- Paths blocked because of [set_case_analysis](#) or [set_disable_timing](#).
- Paths constrained with [set_false_path](#) and [set_clock_groups](#).

Refer to [Example 2 - Illustrates when Block12 does not report a violation](#).

In addition, clock paths are not reported.

Parameter(s)

- *logic_level_min*: Default is 0. Set the value to any positive integer that represents the minimum combinational logic level.

Constraints(s)

None

Messages and Suggested Fix

The following message appears when the level of combinational logic is determined to be less than the value of *logic_level_min*:

[WARNING] Path from "<design-object1>" to "<design-object2>" has "<number-of-levels>" logic levels. This is less than the specified value "<min-number-of-levels>"

Potential Issues

The violation message explicitly states the potential issue.

Consequence of Not Fixing

If path is less than the specified logic level, it might fail to meet the hold check for timing analysis.

How to Debug and Fix

View the incremental schematic of the violation message.

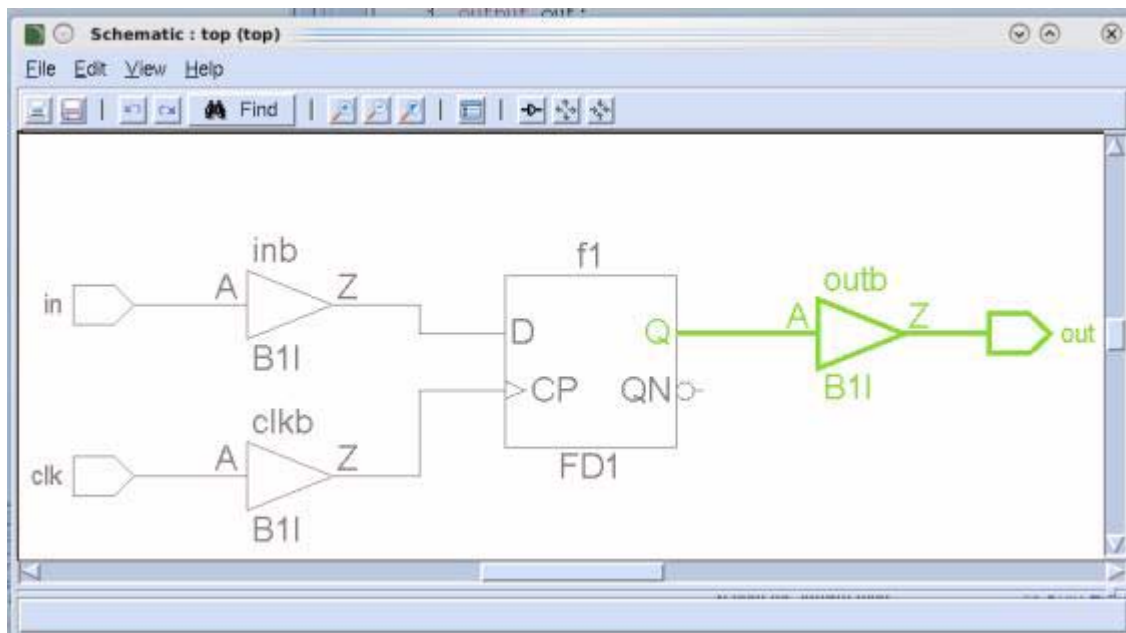
The schematic shows the paths that contain fewer levels of combinational logic than the value specified through the *logic_level_min* parameter.

To resolve this violation, either decrease the value of the *logic_level_min* parameter or update the design to increase the levels of combinational logic.

Example Code and/or Schematic

Example 1 - Illustrates when Block12 reports a violation

In this example, suppose *logic_level_min* is set to 5. The *Block12* rule reports a violation because the level of combinational logic between `top.f1.q` and `top.out` is 1, as shown in the following schematic.



To resolve this violation, you can reduce the value of the *logic_level_min* parameter or revise the design.

Example 2 - Illustrates when Block12 does not report a violation

In this example, suppose *logic_level_min* is set to 5. The *Block12* rule does not report a violation despite the levels of combinational logic determined is 0 because *set_false_path* is set from *inf/Q* to *out f/D*, as shown by the following SDC file snippet.

```
create_clock -period 10 clk
```

```
set_input_delay 1.0 -clock clk in[0]
```

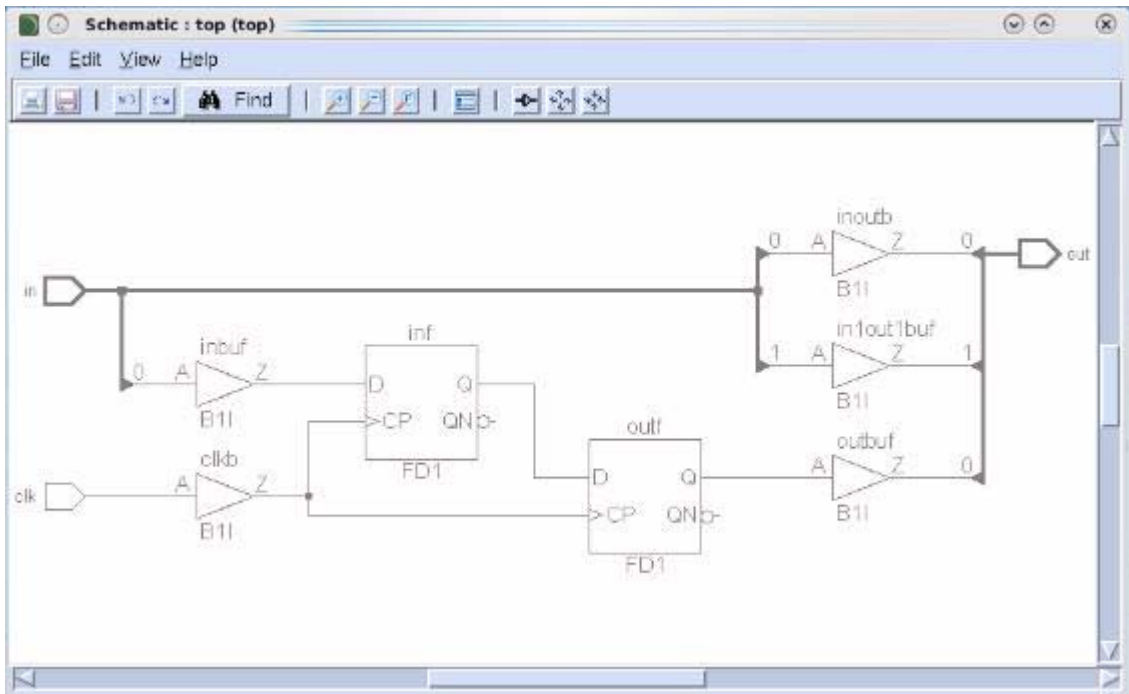
```
set_input_delay 1.0 -clock clk out[0]
```

```
set_false_path -from [get_clocks clk]
```

```
set_max_delay 2 -from in[1] -to out[1]
```

The following schematic shows the design structure.

Block Rules

**Default Severity Label**

Warning

Rule Group

BlockRules

Reports and Related Files

No reports or related files.

Block13

Reports ports driven by clocks through combinational feedthrough paths

When to Use

This rule is applicable during post-synthesis development.

Description

The *Block12* rule reports the combinational feedthrough paths from clock ports/pins to output/inout ports. The [set_case_analysis](#) and [set_disable_timing](#) settings are honored.

Prerequisites

Ensure the SGDC file contains:

```
current_design top
sdc_data -file top.sdc
block -name block1 block2
current_design block1
sdc_data block1.sdc
current_design block2
sdc_data block2.sdc
```

Parameter(s)

- None

Constraints(s)

- [set_case_analysis](#) (Optional): Use to specify the case analysis conditions.
- [set_disable_timing](#) (Optional): Use to disable the timing arc.

Messages and Suggested Fix

The following message appears when a combinational feedthrough path exists in a design or block <name>:

[WARNING] Combinational feedthrough path exists from

clocks <list-of-clocks> to port '<port-name>' for design/block <name>

Potential Issues

The violation message explicitly states the potential issue.

Consequence of Not Fixing

Since the path from the clock to the output port is combinational, the timing slack may not be enough for inter-block routing. Consequently, this may lead to a timing failure.

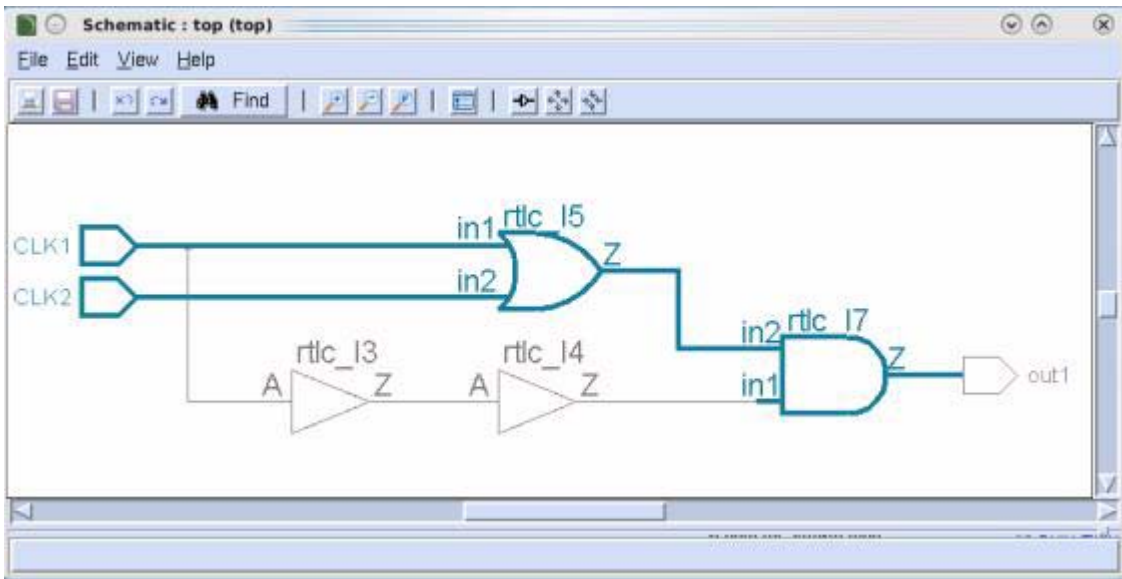
How to Debug and Fix

View the incremental schematic of the violation message. The schematic highlights the combinational feedthrough path.

To resolve this violation, place a register in an appropriate position in the highlighted path.

Example Code and/or Schematic

In this example, clocks are defined at CLK1 and CLK2 and there exists a combinational path between the clock ports to the output port, out1. Refer to the schematic.



To resolve this violation, place a register before out1.

Default Severity Label

Warning

Rule Group

BlockRules

Reports and Related Files

No reports or related files.

Clock Rules

The Clock Rules Group `ClockRules` has the following sub-groups:

- *Clock Consistency Rules*
- *Clock Generation Rules*
- *Clock Latency Rules*
- *Clock Transition Rules*
- *Clock Uncertainty Rules*
- *Dont Touch Rules*
- *High Fan-out Rules*

Clock Consistency Rules

The Clock Consistency Rules Sub-group Clk_Consis contains the following rules:

Rule	Description
<i>Clk_Consis05</i>	Ports where a source clock has been specified and a generated clock has been specified at a higher hierarchical level

Clk_Consis05

Identifies primary and generated clocks that are set on the same port

When to Use

This rule is applicable in all phases of the design.

Description

The *Clk_Consis05* rule reports ports where a source clock and a generated clock have been specified at a higher hierarchical level.

Parameter(s)

None

Constraint(s)

SDC

- *create_clock* (Mandatory): Use to create a clock.
- *create_generated_clock* (Mandatory): Use to create a clock.

Messages and Suggested Fix

The following message appears when a generated clock has been specified on port `<port-name>` when a source clock `<clk-name>` has also been specified on the same port at a lower hierarchical level:

```
[WARNING] create_generated_clock is set on the same port (<port-name>) as create_clock '<clk-name>' but at a higher hierarchical level
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Applying two clocks with different annotations (*create_clock* vs. *create_generated_clock*) on the same port is incorrect.

How to Debug and Fix

Remove one of the clocks.

Example Code and/or Schematic

For the following snippet, the *Clk_Consis05* rule reports a violation because two clocks are applied on logically same object.

```
//test.sdc  
create_clock -period 1 [get_ports clk]  
create_generated_clock -name clock_top -edges {1 3 5} -source  
clk [get_pins B1/clk]
```

Default Severity Label

Warning

Rule Group

Clk_Consis

Reports and Related Files

None

Clock Generation Rules

The Clock Generation Rules Sub-group Clk_Gen contains the following rules:

Rule	Description
<i>Clk_Gen01</i>	Runs the <i>Clk_Gen01a</i> and <i>Clk_Gen01b</i> rules
<i>Clk_Gen01a</i>	Ports and pins driving clock pins of sequential cells that are not specified as clocks
<i>Clk_Gen01b</i>	Ports, pins, and nets driving clock pins of sequential cells that are not specified as clocks
<i>Clk_Gen02</i>	Ports and flip-flop output pins that are not driving flip-flop clock pins or latch enable pins but are specified as clocks
<i>Clk_Gen03</i>	Generated clocks where one or more source pins are not in the fan-out of the specified source clock pin
<i>Clk_Gen05</i>	Clocks that are inferred/specified to be in the same clock domain but have different roots
<i>Clk_Gen06</i>	Flip-flop clocks or latch enables that have multiple source paths and no <i>set_case_analysis</i> constraint directive exists that ensures only one path is active at one time
<i>Clk_Gen07</i>	Nets in clocktrees of clocks that are connected to output ports without dedicated buffering
<i>Clk_Gen08</i>	Clocks generated clocks on an input/output/inout port of a block
<i>Clk_Gen09</i>	Clocks whose source pin falls in the fan-out of source pin of another clock but not generated by the later clock
<i>Clk_Gen10</i>	Clocks that are not propagated in the post-layout phase
<i>Clk_Gen13</i>	Clock names that do not follow the naming convention
<i>Clk_Gen14</i>	Virtual clocks that do not follow the recommended naming convention
<i>Clk_Gen15</i>	<i>create_clock</i> constraints without the waveform specifications
<i>Clk_Gen17</i>	<i>create_generated_clock</i> constraints with non-preferred clock generation method options (-divide_by and -multiply_by)

Rule	Description
<i>Clk_Gen18</i>	<i>create_generated_clock</i> constraints with non-preferred clock generation method option (-edges/ -edge_shift)
<i>Clk_Gen19</i>	Unsupported (for Synthesis) options used in <i>create_generated_clock</i> constraints
<i>Clk_Gen20</i>	<i>set_propagated_clock</i> constraints set for clocks in the synthesis and prelayout constraint scripts
<i>Clk_Gen21</i>	<i>set_propagated_clock</i> constraints set on virtual clocks
<i>Clk_Gen22</i>	Clock ports that have a <i>set_input_delay</i> or <i>set_output_delay</i> constraint specified
<i>Clk_Gen23</i>	Incorrectly defined generated clocks
<i>Clk_Gen23a</i>	Report for validation of <i>create_generated_clock</i>
<i>Clk_Gen24</i>	Clocks reaching sequential cells that do not have a corresponding <i>create_generated_clock</i> command at the sequential cell's output pin
<i>Clk_Gen25</i>	Ports/pins with more than one clock defined using the -add option
<i>Clk_Gen26</i>	<i>create_generated_clock</i> specifications without the -master_clock option when the source pin of the generated clock has either multiple clocks specified or multiple clocks are reaching the source pin
<i>Clk_Gen27</i>	Generated clocks created on pins of non-sequential elements
<i>Clk_Gen29</i>	Clocks that converge through different combinational paths
<i>Clk_Gen30</i>	Generated clock with an odd <i>divide_by</i> factor
<i>Clk_Gen31</i>	Mux's control pin in clock path
<i>Clk_Gen32</i>	Master clock reaching to its generated clocks design object via multiple paths
<i>Clk_Gen33</i>	Reports for unconstrained clock pins
<i>Clk_Gen34</i>	Reports duplicate clock names
<i>Clk_Gen35</i>	Reports clocks not defined on an input port or an output pin of a sequential cell
<i>Clk_Gen36</i>	Reports missing generated clocks at the output of flip-flops

Clk_Gen01

Clock pin not driven by a clock constraint

The *Clk_Gen01* rule runs the *Clk_Gen01a* and *Clk_Gen01b* rules.

Clk_Gen01a

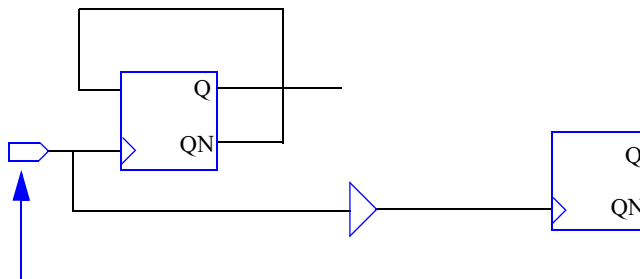
Identifies sequential elements that are not constrained by a clock

When to Use

Use this rule to detect unconstrained sequential elements and when timing coverage is less than 100% for clocks. This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen01a* rule identifies ports/pins driving the clock pins of sequential elements that do not have a clock, [create_clock](#) or [create_generated_clock](#). These sequential elements are unconstrained because no clock reaches the flip-flop clock/control/enable pin of the sequential cell, as shown below.



The clock has to be constrained

This rule also reports ports/pins if a valid path exists from a clock/control/enable pin of a constrained sequential cell to any of the violating terminal points.

For example, if a clock/control/enable pin of a sequential cell is being reached through a MUX in a path, all paths approaching to that sequential cell through the MUX must be constrained by clocks. Suppose, P1 and P2 are connected to mux data inputs, M1/A and M1/B, and P3 is connected to the Select pin of M1. M1/Z is connected to F1/CP, where F1 is flip-flop. P1 has clock c1.

In this scenario, flip-flop F1 is constrained by clock c1 through path p1 -> M1/A -> M1/Z -> M1/CP. F1 is not being constrained through path

p2 ->M1/B ->M1/Z ->M1/CP.

Therefore, the rule reports a violation.

Refer to the [Parameter\(s\)](#) section to learn how to control the behavior of this rule.

Prerequisites

Before running this rule on your design, make sure you:

- Use the SpyGlass Library Compiler feature because the *Clk_Gen01a* rule requires the synthesizable description of all library cells. See the *SpyGlass Explorer User Guide* for details of the SpyGlass Library Compiler feature.
- Check the SDCPARSE results for syntax issues related to the defined clock.

Rule Exceptions

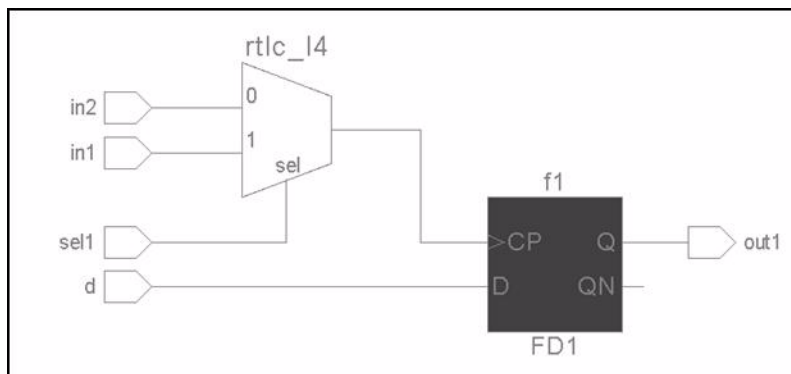
The *Clk_Gen01a* rule does not report a violation if your design has:

- an intended clock that is not driven, such as a hanging clock pin
- a constant value propagated to the CP pin of a sequential clock
- a clock created on the Q pin of a sequential cell and the sequential cell will not toggle
- a sequential cell that will not toggle while traversing back from the CP pin of the sequential cell

Parameter(s)

- *clk_gen01_generate_report*: Default is `no`. Set the value to `yes` to enable report generation. The *tc_clk_gen01a_info Report* is generated.
- *tc_check_for_multiple_clock_fanin*: Default is `yes`. Set the value to `no` to not report violations for clock inputs that are not constrained, if there is one input that is constrained.

For example, in the following schematic, suppose a *create_clock* is defined on the `in2` port and the `in1` port is not constrained. The *Clk_Gen01a* does not report a violation for the `in1` port, if this parameter is set to `no`.



- *tc_ignore_latch_enable*: Default is no. Set the value to yes to ignore checking on control/enable pins of latches.

ignore_io_if_fp (Default is no) and *tc_ignore_te* (Default is yes): By default, the *Clk_Gen01a* rule does not consider the effect of the *set_false_path* constraint.

Set *tc_ignore_te* rule to no and *ignore_io_if_fp* to yes, to consider the *set_false_path* constraints. To understand how the *Clk_Gen01a* rule considers false paths, refer to [Example 5](#).

The following list shows that false paths that are supported:

```
set_false_path -from [get_ports input_port] -to [get_pins
clk_pin]
```

```
set_false_path -from [get_ports input_port]
```

```
set_false_path -through [get_pins
intermediate_clk_path_pin]
```

```
set_false_path -from [get_pins BBox_pin]
```

```
set_false_path -from [get_pins clk_pin]
```

```
set_false_path -to [get_pins clk_pin]
```

```
set_false_path -to [get_inst inst-name]
```

The following list illustrates the type of false paths that are not supported:

```
set_false_path -from [get_clocks c1] -to [get_clocks c2]
```

```
set_false_path -from [get_clocks c1]
```

```

set_false_path -to [get_clocks c2]
set_false_path -from [get_pins FF1/CP] -to [get_pins FF2/D]
set_false_path -from [get_ports input_port] -to [get_pins D_pin]
set_false_path -from [get_inst FF1] -to [get_inst FF2]
set_false_path -from [get_inst inst-name]

```

Constraint(s)

SDC

- `create_clock` (Mandatory): Use to create a clock
- `create_generated_clock` (Mandatory): Use to create a clock
- `set_case_analysis` (Optional): Use to specify the case analysis conditions
- `set_clock_sense/set_sense` (Optional): Use to specify the clock propagation conditions

SGDC

- `current_design` (Mandatory): Use to specify the directive scope

Messages and Suggested Fix

The following message appears for the object `<obj-name>` of type `<obj-type>` (of design/block `<name>`) driving the clock pin `<pin-name>` of the sequential cell instance `<cell-inst-name>` when the object does not have the `create_clock` or `create_generated_clock` constraint set:

```

[WARNING] <constr> constraint not set on <obj-type>
'<obj-name>' driving clock pin <pin-name> of sequential cell
'<cell-inst-name>' of design/block "<name>"

```

Where `<constr>` can be `create_clock` (when `<obj-type>` is port) or `create_generated_clock` (when `<obj-type>` is port, pin, memory pin, or a blackbox pin).

Potential Issues

Violations can arise from situations, such as when your design has:

- A black box in the clock path between the point where the clock constraint is applied and the clock pin of the sequential element. In this case, the black box will block the clock path.
- A clock path that is blocked because you have applied the [set_case_analysis](#) or [set_clock_sense/set_sense](#) with the `-stop_propagation` option in the clock path.
- No clock defined at certain points where they are required.
- The pins of a mux in the clock path are not constrained by using the clock constraint

Consequences of Not Fixing

In your design, sequential elements that are not being driven by a clock leads to a potential design flaw because the timing analysis will not involve the sequential elements. In turn, this may lead to delays in timing closure, inaccuracies in clock tree synthesis, and issues in P&R. If the potential design flaw is not rectified, silicon failure can also be expected.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic highlights the clock pin of a sequential element along with a path to a port or a pin. This indicates that the clock pin of the sequential element is not constrained because neither [create_clock](#) nor [create_generated_clock](#) is applied to the port/pin. Even though the fan-out path of the port/pin reaches the clock pins of other sequential elements, only the sequential element causing the violation is highlighted.

If the [clk_gen01_generate_report](#) parameter is set to `yes`, the *Clk_Gen01a* rule generates the [tc_clk_gen01b_info Report](#). View this report to identify ports or pins to constrain the sequential elements driven by the port or pin.

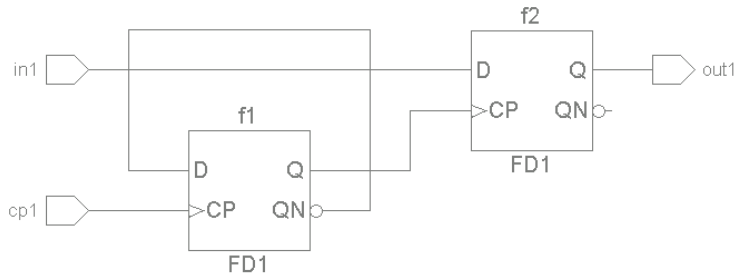
Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows the use of the *Clk_Gen01a* rule to identify ports/pins driving the clock pins of sequential elements that do not have a clock, [create_clock](#) or [create_generated_clock](#). The schematic of the top module is as follows:

Clock Rules



The top .sdc is as follows:

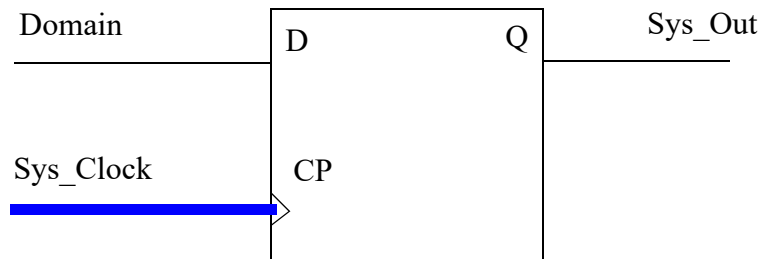
```
top.sdc
1 create_clock -name clk1 -period 10 [get_ports cp1]
2
```

This rule reports a violation message because the clock pin (CP) of the f2 flip-flop is not constrained by [create_generated_clock/create_clock](#).

Example 2

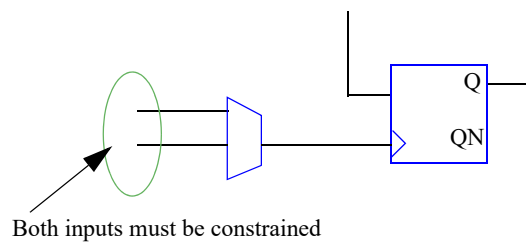
Test Case Files Not Available

In the following schematic, the `sys_clock` port must be constrained by a clock. Otherwise, this rule reports a violation.



Example 3Test Case Files Not Available

In the following figure, if the *set_case_analysis* constraint is not specified on the select pin of the mux, both inputs must have clocks. Otherwise, the input of the mux specified with the *set_case_analysis* constraint must be constrained.



Example 4Test Case Files Not Available

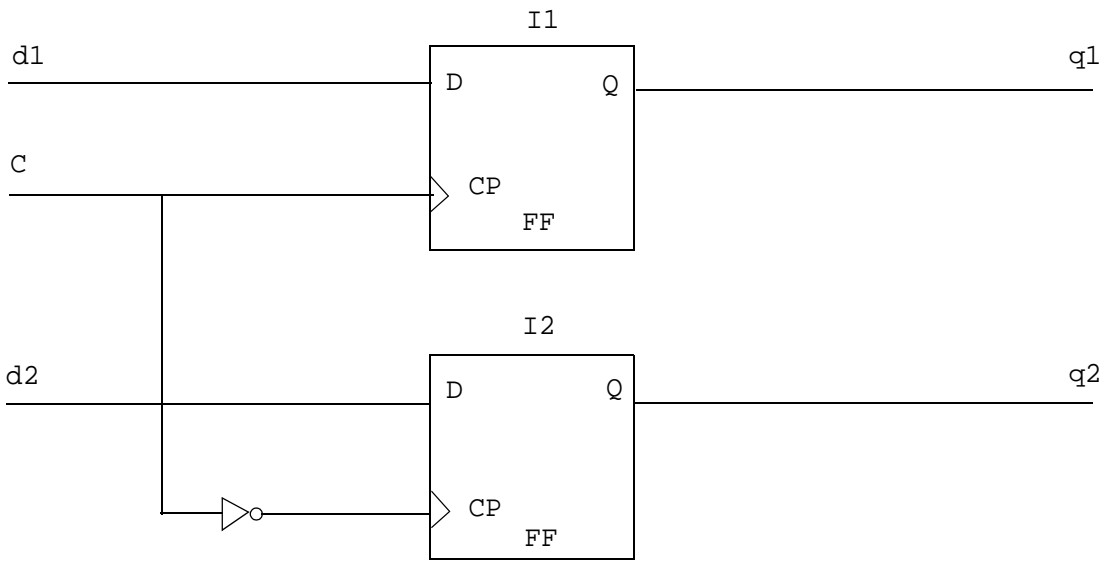
In the following example, since the block will be inferred as a sequential element during synthesis, the clock `sys_clock` must be constrained.

```
reg sys_out;
always (@posedge sys_clock)
begin
    ....
    sys_out <= datain;
    ....
end
```

Example 5Test Case Files Not Available

In the following example, the *Clk_Gen01a* rule does not report a violation because the false path command is specified in the clock path.

```
//test.sdc
set_false_path -from [get_ports c]
```



Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

- [tc_clk_gen01a_info Report](#)

Clk_Gen01b

Clock driven by a constant value or hanging

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen01b* rule reports nets/pins driving clock pins of sequential cells but do not have the *create_clock* or *create_generated_clock* constraint. A sequential cell is said to be unconstrained if no clock reaches the flip-flop clock/control/enable pin of the sequential cell.

This rule flags nets/pins if a valid path exists from a clock/control/enable pin of an unconstrained sequential cell to any of the following valid violating terminal points:

- constant nets (both supply nets and nets due to *set_case_analysis* settings)
- hanging nets
- output pin or a non-toggling sequential cell (cell whose either data or clock pin is constant)
- If there is generated clock applied on the output pin of a non-toggling sequential cell and this generated clock is reaching a clock pin of another sequential cell.

Prerequisites

Use the SpyGlass Library Compiler feature because the Clk_Gen01b rule requires the synthesizable description of all library cells. Refer to the *SpyGlass Explorer User Guide* for details of the SpyGlass Library Compiler feature.

Parameter(s)

- *clk_gen01_generate_report*: Default is no. Set the value to yes to enable report generation. The *tc_clk_gen01b_info Report* is generated. It prints the list of all sequential cells whose clock pins are driven by the reported clock pin/net.
- *tc_ignore_latch_enable*: Default is no. Set the value to yes to ignore checking on control or enable pins of latches.

Constraint(s)

SDC

- *create_clock* (Mandatory): Use to create a clock
- *create_generated_clock* (Mandatory): Use to create a clock
- *set_case_analysis*(Optional): Use to specify the case analysis conditions
- *set_clock_sense/set_sense* (Optional): Use to specify the clock propagation conditions

SGDC

- *current_design* (Mandatory): Use to specify the directive scope

Messages and Suggested Fix

Message 1

The following message appears when a hanging object *<obj-name>* of type *<obj-type>* (of design/block *<name>*) is driving the clock pin *<pin-name>* of a sequential cell instance *<cell-inst-name>*:

```
[WARNING] Hanging '<obj-type>' '<obj-name>' of design/block "<name>" is driving clock pin <pin-name> of sequential cell '<cell-inst-name>'
```

Where, *<obj-type>* is a net or pin and *<obj-name>* is the name of the net or pin.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Clocks being driven by a constant value is a serious error in the design because the timing analysis results will be invalid without proper clock constraints. The timing results may give a false sense of security and you may get silicon timing failure. This can cause sections of the chip to be non-functional.

How to Debug and Fix

View the incremental schematic of the violation message.

The unconstrained clock pin connected a net/pin is hanging. The schematic shows the path from the clock pin to the net/pin. This is incorrect because

each unconstrained clock pin should be constrained with a clock.

Message 2

The following message appears when net `<net-name>` is directly driving the clock pin `<pin-name>` of a sequential cell instance `<cell-inst-name>` in design/block `<name>` but is being set to a constant value `<value>` due to case analysis settings:

[WARNING] Net '`<net-name>`' driving clock pin `<pin-name>` of sequential cell '`<cell-name>`' of design/block "`<name>`" is getting set to constant value '`<value>`' due to `set_case_analysis` settings

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

All messages of this rule have the same consequence of not fixing. Refer to [Consequences of Not Fixing](#).

How to Debug and Fix

View the incremental schematic of the violation message.

The net to which the unconstrained clock pin is connected is set to a constant value using the [set_case_analysis](#). The schematic shows the path from the clock pin to the net.

Message 3

The following message appears when a pin `<pin-name>` of a sequential cell instance `<cell-inst-name>` in design/block `<name>` but is being set to a constant value `<value>`:

[WARNING] Pin '`<pin-name>`' of sequential cell '`<cell-name>`' of design/block "`<name>`" is getting set to constant value '`<value>`' due to `set_case_analysis` settings

Where, `<value>` can be 0 or 1

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

All messages of this rule have the same consequence of not fixing. Refer to

Consequences of Not Fixing.

How to Debug and Fix

The unconstrained clock pin is set to a constant value using [set_case_analysis](#). The schematic highlights the path from the clock pin of the sequential cell to the port/pin where `set_case_analysis` is set.

Message 4

The following message appears if the [create_generated_clock](#) constraint is set/not set `<value>` on a pin `<pin-name>` which is driving the clock pin `<clk-pin-name>` of a sequential cell instance `<cell-inst-name>` of the design/block `<name>` and the pin `<pin-name>` does not toggle as the clock/data pin of the sequential device `<device-name>` is set to a constant value:

```
[WARNING] create_generated_clock constraint <value> on pin
<pin-name> driving clock pin <clk-pin-name> of sequential cell
<cell-name> of design/block <name> and pin <pin-name> does not
toggle because clock/data pin of sequential device of
device_name is getting set to constant value
```

Where, `<value>` can be an empty string if the [create_generated_clock](#) constraint is set and not if the `create_generated_clock` constraint is not set.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

All messages of this rule have the same consequence of not fixing. Refer to [Consequences of Not Fixing](#).

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic highlights the path from the clock pin to the pin driving that sequential cell. The pin may or may not have the [create_generated_clock](#) constraint set. By running [set_case_analysis](#) with the `Clk_Gen01b` rule, the schematic generated by [set_case_analysis](#) shows the constant value being propagated to the input pin of the sequential cell.

Update the SDC file to ensure the value propagated to the input pin of the sequential cell is not constant.

Message 5

The following message appears when the timing path to the unconstrained clock pin `<pin3-name>` is disabled due to the [set_disable_timing](#) constraint specified between pins, `<pin1-name>` and `<pin2-name>`:

[WARNING] The timing arc between `<pin1-name>` and `<pin2-name>` is disabled due to `set_disable_timing` constraint, which is driving the clock pin `<pin3-name>` of sequential cell `<cell-name>` of design/block `<name>`

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

All messages of this rule have the same consequence of not fixing. Refer to [Consequences of Not Fixing](#).

How to Debug and Fix

View the incremental schematic of the violation message.

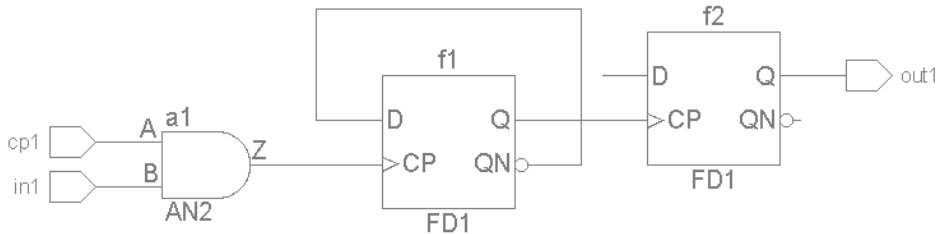
The schematic highlights the timing path from the clock pin to the pin where the [set_disable_timing](#) is applied. Update the SDC file by removing the [set_disable_timing](#) constraint.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when [Message 2](#) is reported. The schematic of the top module is as follows:



The top.sdc is as follows:

```
top.sdc
1 create_clock -name clk1 -period 10 [get_ports cp1]
2 set_case_analysis 0 [get_ports in1]
```

This rule reports a violation message because the clock pin CP of the f1 flip-flop is driving by a constant value. In the top.sdc file, the in1 port is set to 0.

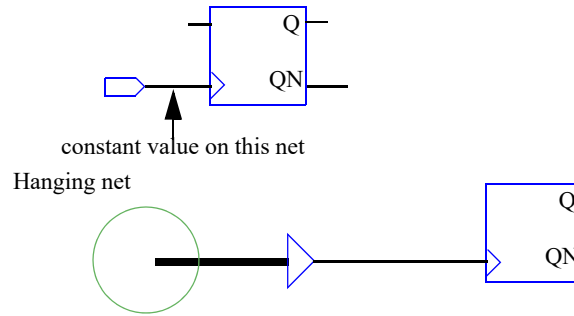
Example 2

Test Case Files Not Available

In this example, a constant value is reaching to clock pin of the flip-flop. In addition, a hanging net is driving the clock pin of the flip-flop. Therefore, the *Clk_Gen01b* rule reports a violation for both cases.

Clock Rules

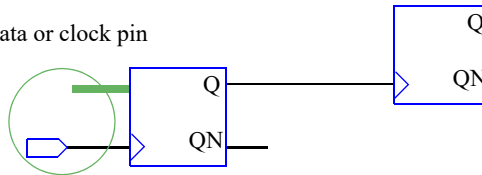
set_case_analysis on data or clock pin

**Example 3**

Test Case Files Not Available

The first flip-flop is F1 and second is F2 in the diagram given below. Let clock gc1 is applied on F1/Q and the *set_case_analysis* constraint is applied on the data/clock pin of F1. As F1/CP or F1/D is constant, F1 will not toggle. The *Clk_Gen01b* rule reports a violation because gc1 is applied on F1/Q and feeding to clock pin of flip-flop F2/CP.

set_case_analysis on data or clock pin

**Default Severity Label**

Warning

Rule Group

Clk_Gen

Reports and Related Files

- [tc_clk_gen01b_info Report](#)

Clk_Gen02

Reports a constrained clock that is not used as a clock

When to Use

To identify redundant clocks or clocks that are not defined at the place where they should have been. This rule is applicable to all design phases.

Description

The *Clk_Gen02* rule reports ports and pins that are not driving flip-flop clock pins or latch enable pins but have the [create_clock](#) constraint or the [create_generated_clock](#) constraint set. In addition, this rule reports:

- virtual clocks that are not used in any other constraint, such as in [set_input_delay](#) and [set_output_delay](#).
- ports/pins that have the [create_clock](#) or the [create_generated_clock](#) constraint set and the net connected to the port/pin is forced to a fixed value due to [set_case_analysis](#) or [set_disable_timing](#) specifications.

The *Clk_Gen02* rule reports only one message even when the clock reaches more than one non-clock pin.

This rule supports the [Source Synchronous Interfaces](#) feature.

The *Clk_Gen02* rule reports clocks, which do not reach any sequential cells. When the [tc_ignore_clk_outports](#) parameter is set to yes, this rule does not report violations for clocks that reach output ports. For more information on the impact of the [tc_ignore_clk_outports](#) parameter, see [Example 3](#).

Parameter(s)

- *strict*: Default is no. This indicates that the *Clk_Gen02* rule reports port/pins if no fan-out of a clock design object reaches the clock pin of the flip-flop. When the *strict* parameter is set to yes or *Clk_Gen02*, the *Clk_Gen02* rule flags clock ports/pins if any fan-out of the clock design object reaches any of the following:
 - blackbox instance
 - sequential cell data pin (any pin except the clock pin)
 - output/inout port
 - enable pin of a tristate

- ❑ latch
- ❑ sequential element that has a data pin is driven by a constant
- ❑ any other clock constrained with the *create_clock* or *create_generated_clock* constraint
- ❑ if a pin is constrained using the *set_clock_sense* command with the *-stop_propagation* option. Refer to the *Stopping Clock Propagation* section for more details.
- *tc_source_syn_clks*: Default is yes and all dependent rules will the *Source Synchronous Interfaces* feature. This impacts generated clocks (divide_by 1 only) applied at inout/output ports.
- *tc_ignore_clk_outports*: Default is no and the *Clk_Gen02* rule reports clocks which terminate on output ports. Set the value to yes to ignore clocks that terminate on output ports. For more information on the impact of the *tc_ignore_clk_outports* parameter, see *Example 3*.

Constraint(s)

SDC

- *create_clock* (Mandatory): Use to create a clock
- *create_generated_clock* (Mandatory): Use to create a clock
- *set_case_analysis* (Optional): Use to specify the case analysis conditions
- *set_input_delay* (Optional): Use to specify the delay value of the data path.
- *set_output_delay* (Optional): Use to specify the delay value of the data path.
- *set_disable_timing* (Optional): Use to disable the timing arc.
- *set_clock_sense/set_sense* (Optional): Use to specify the clock propagation conditions

SGDC

- *current_design* (Mandatory): Use to specify the directive scope

Messages and Suggested Fix

Message 1

The following message appears when a clock *<clk-name>* of type *<clk-type>* is set on an object *<obj-name>* that is not used as a clock and the *strict* parameter is set to *yes*:

[WARNING] *<clk-type>* *<clk-name>* set on an object *<obj-name>* which is not used as a clock

Where, *<clk_type>* can be *create_clock* or *create_generated_clock* and *<obj-name>* is the name of the design object.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

A constrained clock not being used as a clock is an error in the design because the timing analysis results will be invalid without proper clock constraints. The timing results may give a false sense of security and you may get silicon timing failure. If the constraint is redundant and should be removed since unnecessary constraints increase the run time of the implementation tools.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows one element of the following list where the clock is reaching:

- Black box
- Tristate enable pin
- Latch
- Mux select pin
- Sequential cell data pin
- Inout port
- Output port

To fix this violation, remove the *create_clock/create_generated_clock* constraint.

Message 2

The following message appears when the virtual clock `<vclk-name>` has been defined but not used in any other constraint and the *strict* parameter is set to *yes*:

[WARNING] Virtual clock "`<vclk-name>`" not used

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

All messages of this rule have the same consequence of not fixing. Refer to [Consequences of Not Fixing](#).

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic highlights the fan-out path from the clock definition point till the violation condition. This means that the virtual clock is declared but not used in any of the other constraint. Therefore, this declaration is considered redundant.

To fix this violation, remove the [create_clock/create_generated_clock](#) constraint.

Message 3

The following message appears when a clock `<clk-name>` of type `<clk-type>` is set on an object `<obj-name>` reaches a blackbox instance, a sequential cell data pin, an output/inout port, or the enable pin of a tristate and the *strict* parameter is set to *yes*:

[WARNING] `<clk-type>` `<clk-name>` set on an object `<obj-name>`, reaches on `<cell-type>`

Where, `<clk-type>` can be [create_clock](#), [create_generated_clock](#), `<obj-name>` is the name of the design object, and `<cell-type>` can be a blackbox, latch, enable pin of a tristate, output/inout port, or a sequential cell's data pin.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

All messages of this rule have the same consequence of not fixing. Refer to [Consequences of Not Fixing](#).

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the fan-out of the clock is blocked due to constant or the path itself is disabled.

To fix this violation, remove the [create_clock/create_generated_clock](#) constraint.

Message 4

The following message appears when a clock `<clk-name>` of type `<clk-type>` is set on an object `<obj-name>` reaches a black box instance, a sequential cell data pin, an output/inout port, or the enable pin of a tristate and the `strict` parameter is set to `no`:

[WARNING] `<clk-type>` `<clk-name>` set on an object `<obj-name>`, is not used as a clock and terminates on `<cell-type>`

Where, `<clk_type>` can be [create_clock](#), [create_generated_clock](#), `<obj-name>` is the name of the design object, and `<cell-type>` can be a black box, latch, enable pin of a tristate, output/inout port, or a sequential cell's data pin.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

All messages of this rule have the same consequence of not fixing. Refer to [Consequences of Not Fixing](#).

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the fan-out of the clock is blocked due to constant or the path itself is disabled.

To fix this violation, remove the [create_clock/create_generated_clock](#) constraint.

For more information on the impact of the [tc_ignore_clk_outports](#) parameter, see [Example 3](#).

Message 5

The following message appears for a port or pin *<name>* that is not driving a flip-flop clock pin or a latch enable pin but has a clock *<clk-name>* of type *<clk-type>* set when the net connected to the port/pin is forced to a fixed value due to [set_case_analysis](#) or [set_disable_timing](#) specifications and the value of the [strict](#) parameter is set to yes:

[WARNING] *<clk-type>* *<clk-name>* set on an object "*<name>*" which is not used as a clock since the net is blocked by either [set_case_analysis](#) or [set_disable_timing](#) or the sequential element is driven by a constant

Where, *<clk_type>* can be [create_clock](#), [create_generated_clock](#), *<obj-name>* is the name of the design object and *<cell-type>* can be a blackbox, latch, enable pin of a tristate, output/inout port, or a sequential cell's data pin.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

All messages of this rule have the same consequence of not fixing. Refer to [Consequences of Not Fixing](#).

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the fan-out of the clock is blocked due to constant or the path itself is disabled. If clock is not reaching any clock pin or if the [strict](#) parameter is set to yes, the violation message has some extra phrases pointing to the conditions which might have blocked the clock not reaching to clock pin.

To fix this violation, either:

- Review the [set_case_analysis](#) and [set_disable_timing](#),
- Remove the [create_clock/create_generated_clock](#) constraint, or
- Ensure that the data pin of the sequential element is not driven by a constant

Message 6

The following message appears when a clock *<clk-name>* of type *<clk-*

type> is set on an object <*obj-name*> that is not used as a clock and the *strict* parameter is set to no:

[WARNING] <clk-type> <clk-name> on object <obj-name> is not used as a clock because clock's propagation is stopped due to set_clock_sense in its fan-out at <variable name>

Where, <clk_type> can be [create_clock](#), [create_generated_clock](#), and <obj-name> is the name of the design object.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

All messages of this rule have the same consequence of not fixing. Refer to [Consequences of Not Fixing](#).

How to Debug and Fix

The clock propagation is blocked due to the -stop_propagation option in set_clock_sense. The [set_clock_sense/set_sense](#) constraint is highlighted.

To fix this violation, either:

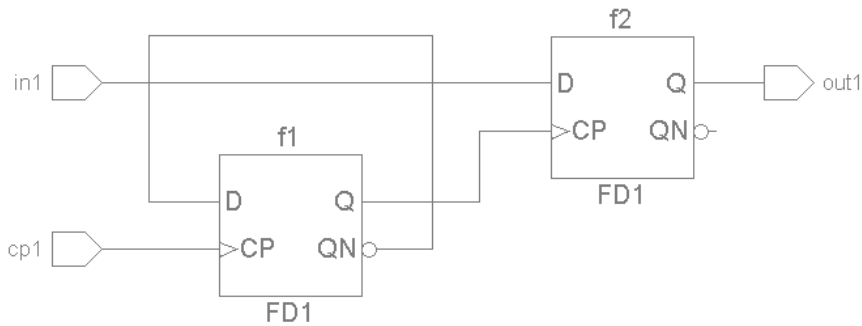
- review the [set_case_analysis](#) and [set_disable_timing](#), or
- remove the [create_clock/create_generated_clock](#) constraint.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when [Message 4](#) is reported. The schematic of the top module is as follows:



The top.sdc is as follows:

```
top.sdc | top.sdc |
create_clock -name clk1 -period 10 [get_ports cp1]
create_clock -name clk2 -period 10 [get_ports in1]
```

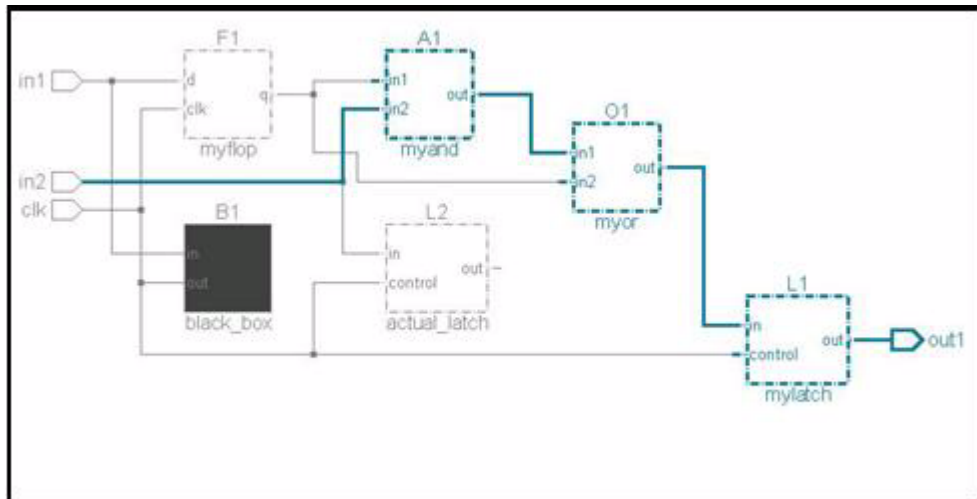
This rule reports a violation message because the clock `clk2` is not used as a clock. In the `top.sdc` file, the clock `clk2` defined at the port `in1`, which is connected to the data pin of the flip-flop `f2`.

Example 2

Test Case Files Not Available

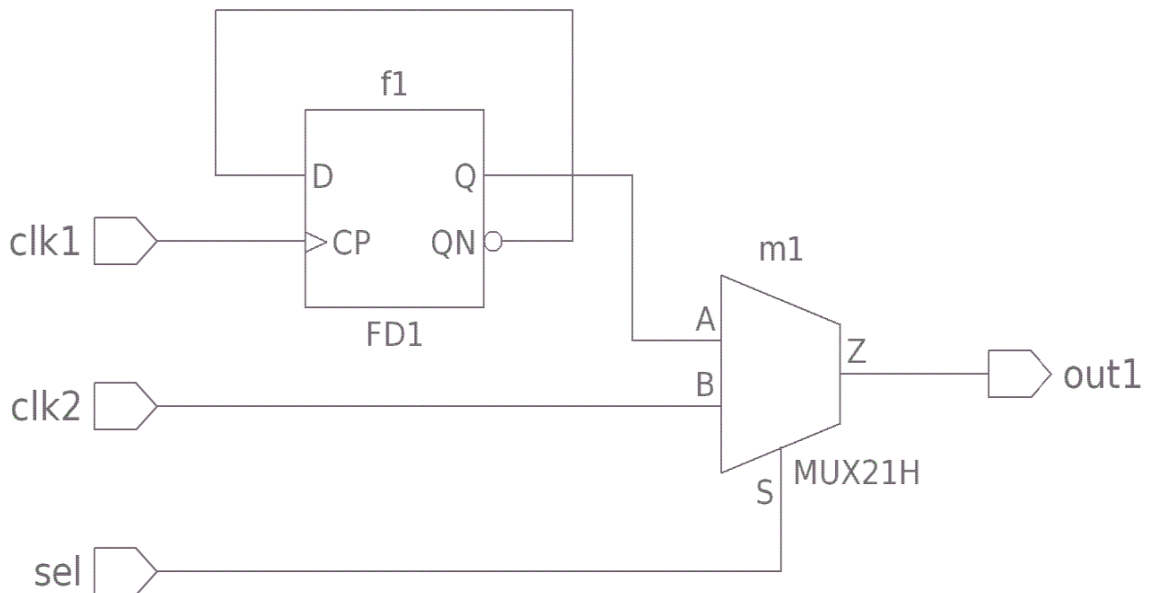
In this example, the `Clk_Gen02` rule reports a violation because the clock applied on `in2` is not being used.

Clock Rules

**Example 3**

Test Case Files Not Available

This example shows when [Message 4](#) is reported by the *Clk_Gen02* rule.



Clocks, C1 and C2, are applied on the clk1 and clk2 ports. The gC1 generated clock is applied on the f1/Q pin. The C2 and gC1 clocks are propagating till the out1 output port. The SDC commands are as follows:

```

1 create_clock -name C1 -period 10 [get_ports clk1]
2 create_clock -name C2 -period 15 [get_ports clk2]
3 create_generated_clock -name gC1 -divide_by 2 -source [get_ports clk1] [get_pins f1/Q]
4

```

If the `tc_ignore_clk_outports` parameter is set to no, the `Clk_Gen02` rule reports the following violations because the C2 and gC1 clocks are not used as clocks, even though they are propagating to the out1 output port.

Clock Rules

jest_goal (2)			
Clk_Gen02 (2) : Constrained clock not used as a clock			
⚠	↕	create_clock "C2" set on an object "clk2", is not used as a clock and terminates on output port	top.sdc 2
⚠	↕	create_generated_clock "gC1" set on an object "I/Q", is not used as a clock and terminates on output port	top.sdc 3

If the `tc_ignore_clk_outports` parameter is set to `yes`, the `Clk_Gen02` rule does not report these violations because the C2 and gC1 clocks are propagating to the out1 output port.

Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen03

Identifies a generated clock that is not in the fan-out of its source clock

When to Use

To detect whether the generated clock is defined correctly. This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen03* rule reports generated clocks where no source object is found. Generated clocks are specified using the [create_generated_clock](#) constraint. In addition, this rule flags generated clocks that have a blackbox in their fan-in path.

The *Clk_Gen03* rule does a backward traversal to search the source of a generated clock, *gc2*. If another generated clock, *gc1*, is found in its path during the search, then the search will continue beyond the generated clock, *gc1*, until the source is found. Alternatively, if a create clock, *c1*, is encountered then the search stops there, assuming there are no alternate paths in the fan-in to the source. When no source is found, this rule reports a violation.

Rule Exception

The *Block10* rule breaks a combinational loop by applying [set_disable_timing](#) constraints. Therefore, the *Clk_Gen03* traversal is stopped where the [set_disable_timing](#) constraint is applied. Hence, may lead unexpected results. [Message 4](#) is reported.

Parameter(s)

None

Constraint(s)

SDC

- [create_generated_clock](#) (Mandatory): Use to create a clock

Messages and Suggested Fix

Message 1

The following message appears for the generated clock `<clk-name>` with defined source port/pin `<pin-name>` where one or more source ports/pins are not in the fan-out of the specified source clock port/pin:

[WARNING] Generated clock "`<clk-name>`" is not in the transitive fan-out of its defined source port/pin "`<pin-name>`"

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Constraining and generating with a wrong clock could potentially allow a greater slack than available. Even if the timing analysis is run successfully, the chip might eventually have a timing failure.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the transitive fan-in from the generated clock to its source/black box. The declared source point is also shown and does not appear in the fan-in of the generated clock.

To fix this violation, modify the `-source` argument of [create_generated_clock](#) constraint to provide the correct clock source.

Message 2

The following message appears for the generated clock `<clk-name>` is in the fan-out of a black box `<bb-name>`:

[WARNING] Generated clock `<clk-name>` is in the transitive fan-out of black-box `<bb-name>`

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

All messages of this rule have the same consequence of not fixing. Refer to [Consequences of Not Fixing](#).

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the transitive fan-in from the generated clock to its source/black box. The declared source point is also shown and does not appear in the fan-in of the generated clock.

To fix this violation, modify the `-source` argument of [create_generated_clock](#) constraint to provide the correct clock source.

Message 3

The following message appears for the generated clock `<gclk-name>` when its source clock port/pin `<pp1-name>` is the same as the generated clock port/pin `<pp2-name>`:

```
[WARNING] Source port/pin '<pp1-name>' of generated clock '<gclk-name>' is same at flat level to its port/pin '<pp2-name>'
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

All messages of this rule have the same consequence of not fixing. Refer to [Consequences of Not Fixing](#).

How to Debug and Fix

After clicking the violation message, the SDC file appears. Review the generated clock definition in the SDC file. The source point and point at which the clock is declared is same.

To fix this violation, modify the `-source` argument of [create_generated_clock](#) constraint to provide the correct clock source.

Message 4

If the [Block10](#) rule has applied [set_disable_timing](#) to a loop, the following message appears:

```
[INFO] The set_disable_timing constraints generated in file <file-name>
```

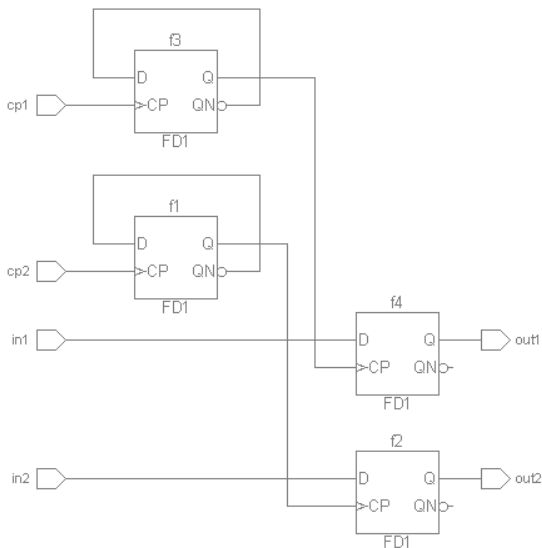
Refer to the [Block10](#) rule documentation for more information.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when *Message 1* is reported. The schematic of the top module is as follows:



The top.sdc is as follows:

```

top.sdc top.sdc
create_clock -name clk1 -period 10 [get_ports cp1]
create_clock -name clk2 -period 10 [get_ports cp2]
create_generated_clock -name gclk1 -divide_by 2 -source [get_ports cp1] [get_pins f2/CP]

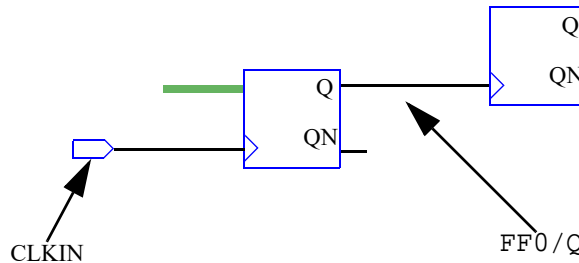
```

This rule reports a violation message because the generated clock gclk1 is not a fan-out of the port cp1. Instead, gclk1 is a fan-out of cp2.

Example 2

Test Case Files Not Available

Consider the following example:



```
create_clock -name CLK0 -period 5.0 \
  -waveform {0.0 2.5} [get_ports CLKIN]
create_generated_clock -name CLK \
  -source [get_ports CLKIN] \
  -divide_by 2 [get_pins FF0/Q]
create_generated_clock -name CLK \
  -source [get_ports CLKIN] \
  -divide_by 2 [get_pins FF1/Q]
```

Here, the source clock CLK0 is associated with port CLKIN and its generated clock CLK is triggering flip-flops FF0 and FF1. Then, the *Clk_Gen03* rule expects that the pins FF0/Q and FF1/Q are in the fan-out from port CLKIN.

Example 2

Test Case Files Not Available

The *Clk_Gen03* rule reports cases where the source port/pin of a generated clock is the same as or is directly connected to the generated clock pin. Consider the following example:

```
create_clock -name clk1 -period 10 [get_ports cp]
create_generated_clock -name gclk1 -divide_by 2
  -source cp [get_ports cp]
```

```
create_generated_clock -name gclk2 -divide_by 2
    -source cp [get_pins b1/cp]
```

The *Clk_Gen03* rule reports a violation if the source port/pin (*cp*) and clock port/pin (*b1/cp*) of generated clock *gclk1* are directly connected in the design.

Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen05

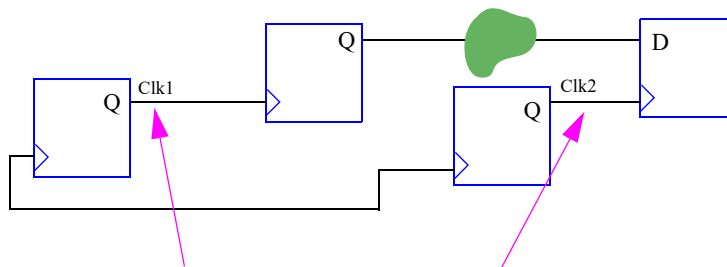
Reports synchronous clocks that have a different root clock

When to Use

This rule is applicable to all design phases.

Description

The *Clk_Gen05* rule reports clocks that are inferred/specified to be in the same clock domain but have different roots. For clocks of the same clock domain, this rule checks whether a combinational path exists between two flip-flops driven by clocks of the same clock domain. If such a path is found in the design, this rule checks whether the two clocks have the same root. *Clk_Gen05* reports a violation message when the clocks have different roots.



Clocks Clk1 and Clk2 should have the same root

The *Clk_Gen05* rule processes the [clock_group](#) constraint, if specified.

Prerequisites

Run the *Clk_Gen05* rule only after specifying the [clock_group](#) information in the SGDC file. However, if you do not specify the [clock_group](#) information in the SGDC file, the *Clk_Gen05* rule accesses the clock domain information from the [DomainAnalysis](#) rule.

Parameter(s)

- *tc_clk_compat*: Default is no. Set the value to yes to report a violation if one clock `clk1` is applied directly or through a combinational logic to

the D pin of flip-flop F1 and another clock `clk2` is applied at the CP pin of the same flip-flop F1. Here, `clk1` and `clk2` are synchronous clocks.

Constraint(s)

SGDC

- *clock_group* (Optional): Use to specify the relationship between clocks, such as synchronous or asynchronous.

If you do not specify any *clock_group* constraint for any clocks, a pair of clocks specified in a *set_false_path* or *set_clock_groups* constraint is assumed to be asynchronous and a pair of clocks specified in a *set_clock_uncertainty* constraint is assumed to be synchronous. This rule does not consider *set_false_path* constraints that have the `-through` list specified.

Messages and Suggested Fix

The following message is generated for two clocks `<clk1-name>` (root clock `<root-clk1>`) and `<clk2-name>` (root clock `<root-clk2>`) that are in the same *clock_group*, have different roots, and have a combinational path between flip-flops driven by these clocks:

[WARNING] Clocks (`<clk1-name>`, `<clk2-name>`) have distinct roots (`<root-clk1>`, `<root-clk2>`) but are in the same *clock_group* and there is at least one communicating path between these clocks

Where, root clock names are the names of the root clock port/pin.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

The *Clk_Gen05* rule ensures that the clock tree synthesis can align together all the registers of a clock domain. If there are more than one root clock for a given clock domain, the alignment has to be done during the clocktree synthesis at the upper level of implementation hierarchy. Then, the timing analysis results within the block can be trusted only under the assumption that the upper level alignment is correct. This situation must be avoided as much as possible.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the path to the clock pins where clocks are declared. These clock pins are declared on different points. The schematic also shows the data path starting from the output of one sequential element to another sequential element data pin.

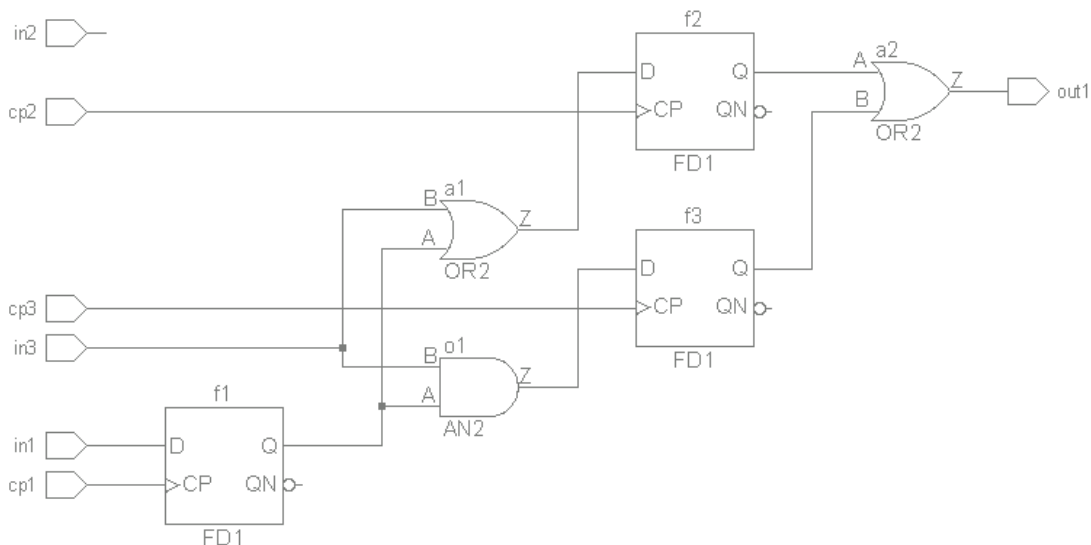
In addition, check if the *clock_group* constraint specified in the SGDC file are correct. Either fix the *clock_group* definitions or clock declaration to resolve this violation.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The SGDC file is as follows:

Clock Rules

```

test.sgdc x
1 current_design top
2 domain -name d1 -clock { clk1 clk2 clk3 }

```

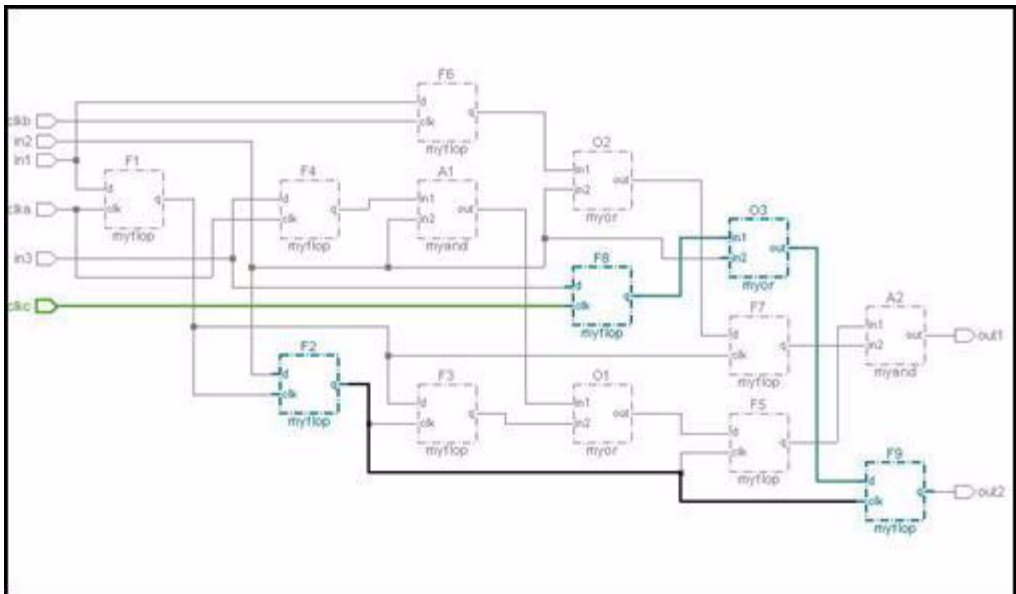
This rule reports a violation message indicates that clocks `clk1` and `clk2` belong to the same domain, though they have distinct roots. View the SGDC file.

In the SDC file, the domains related to the clocks are defined.

Example 2

Test Case Files Not Available

Suppose clock `c1` is applied on the `clk_c` object and `gc1` is applied on `F2/q`. Clocks `c1` and `gc1` are assigned to the same `clock_group` in the SGDC file. The `Clk_Gen05` rule reports a violation because clocks are crossing on the `F9` flip-flop, but the clocks are applied on a different object `clk_c` & `F2/q`.



Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen06

Reports multiple paths that exist from the clock pin of a sequential cell to different clock sources

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen06* rule reports clock pins of sequential cells that have multiple source paths and no *set_case_analysis* constraint directive is specified in the SDC files. The *Clk_Gen06* rule requires that only a single path must exist from the clock pin of a sequential element to its source.

In case of multiple clocks defined on the same object using the `-add` option of the *create_clock/create_generated_clock* constraint, this rule retains only the last clock defined without the `-add` option and any clocks defined with the `-add` option after this (last) clock.

The *Clk_Gen06* rule reports only one message for all nets connected to the converging port/pin listing name of the first sequential cell instance found and the total number of sequential cell instances in the complete fan-out of the converging port/pin.

Multiple paths from source to point-of-use are normally found when multiplexers are employed to select between test clocks and source clocks. For such cases, a *set_case_analysis* constraint should be specified in the SDC file to correctly select only one of the clocks to be active.

If multiple clocks reach the clock pin of a flip-flop and only one clock is active and rest of the clocks are stopped by using the *set_clock_sense/set_sense* command with the `-stop_propagation` option, the *Clk_Gen06* rule does not flag a violation message Refer to the *Stopping Clock Propagation* section for details.

Prerequisites

Use the SpyGlass Library Compiler feature because the *Clk_Gen06* rule requires the synthesizable description of all library cells.

Rule Exception

If *set_false_path* or *set_clock_groups* is specified between two clocks, C1 and C2, and these clocks are merging at the clock pin of a flip-flop or in the

fan-in of the clock pin of a flip-flop, the *Clk_Gen06* rule does not report a violation.

Parameter(s)

- *tc_ignore_scg*: Default is no. Set this parameter to yes to ignore *set_clock_groups* and to report the corresponding violation message.

Constraint(s)

- *set_clock_sense/set_sense* (Optional): Use to specify the clock propagation conditions

Messages and Suggested Fix

The following message appears when multiple source paths from clocks *<clk-name-list>* exist for net *<net1-name>* connected to the clock pin of sequential element *<seq-elem-inst-name>* (and the clock pins of *<num>* other sequential elements) in design/block *<name>* and no *set_case_analysis* constraint directive exists in the SDC file to ensure that only one path is active at one time:

[INFO] In *<type>* "*<name>*", net "*<net1-name>*" connected to clock pin of sequential cell "*<seq-elem-inst-name>*" (and *<num>* other cells) is driven by multiple clocks (*<clk-name-list>*). However, activation of only one path is not being ensured (e.g. by *set_case_analysis*)

Where, *<type>* can be design or block.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Paths from the clock pin of a sequential cell to multiple clock sources can cause inconsistency between pre and post layout timing. This can also result in an unbalanced clock trees after clock tree synthesis.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows all clock pins connected to the net and the fan-in

path till the gate at which the clock sources are converging. For each net connected to the clock pins receiving multiple clocks, a violation message is reported. The schematic also highlights if the converging clock source is a multiplexer. In this case, the select pin of the multiplexer is traversed to find probable end points at which case analysis should have been applied.

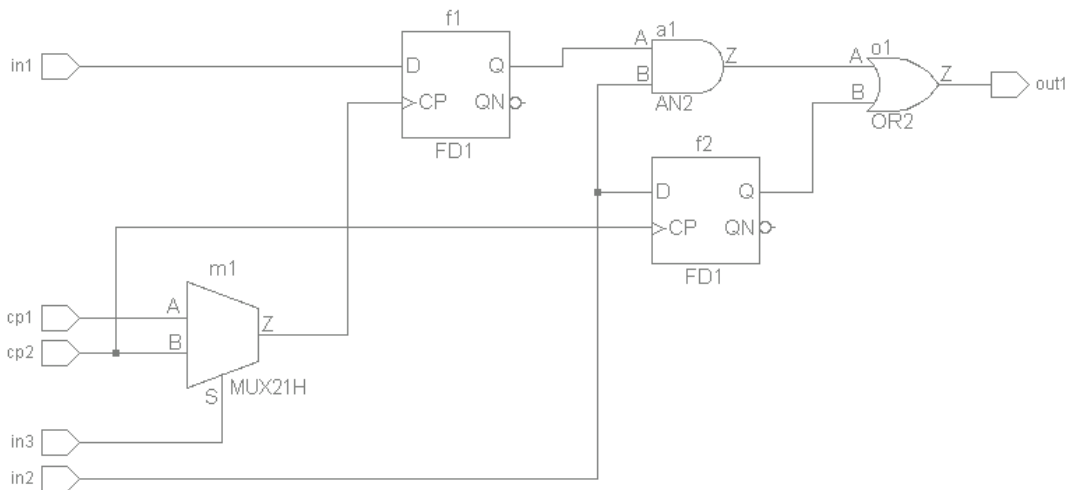
By applying appropriate case analysis constraints should resolve this situation.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a message. The schematic of the top module is as follows:



The top.sdc is as follows:

```

top.sdc x
1 create_clock -name clk1 -period 10.000000 {cp1}
2 create_clock -name clk2 -period 10.000000 {cp2}

```

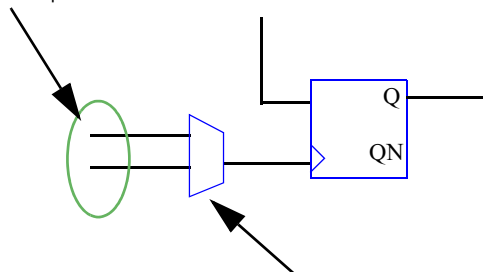
This rule reports a violation message because because the clock pin CP of the f1 flip-flop is driven by two clocks clk1 and clk2. In top.sdc, there is no logic value is defined on the select pin of multiplexer m1.

Example 2

 Test Case Files Not Available

Suppose two clocks are applied on the inputs of a mux, as shown in the following image. The *Clk_Gen06* rule reports a violation because clocks are mixing at the output pin of the mux and reaching to the clock pin of the flip-flop.

create_clock set on both inputs of mux



If the set_case_analysis constraint is not specified on the select pin of the mux then the rule will flag

Default Severity Label

Info

Rule Group

Clk_Gen

Clock Rules

Reports and Related Files

None

Clk_Gen07

Identifies a clocktree net which is connected to a port without dedicated buffering

When to Use

To detect whether clock nets are connected to inout or output ports through a dedicated buffer. This rule is in the post-synthesis phase.

Description

The *Clk_Gen07* rule reports nets in clock trees of clocks that are connected to output ports without dedicated buffering. These clocks are specified using the *create_clock* and *create_generated_clock* constraint. Sequentially generated clock must be also buffered.

The *Clk_Gen07* rule allows buffer cells from the associated .lib file only. Inverter cells are not accepted for buffering. Acceptable buffers are any instantiated combinational gates, such as buf, inverter, gating, and mux.

Rule Exceptions

The *Clk_Gen07* rule does not perform any checks in the design hierarchy of top-level design units specified using the *chip* rule parameter.

Parameter(s)

None

Constraint(s)

SDC

- *create_clock* (Mandatory): Use to create a clock.
- *create_generated_clock* (Mandatory): Use to create a clock.

Messages and Suggested Fix

Message 1

The following message appears when an output port *<port-name>* of design/block *<name>* is driven by a clock tree net of clock *<clk-name>* without dedicated buffering:

```
[Clk_Gen07_01] [INFO] In design/block "<name>", port "<port-
```

name>" is driven by a clocktree net (of clock <clk-name>) without dedicated buffering

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Clocktree net connected to a port without dedicated buffering could result in delays in the clock tree network. You have a more deterministic approach when it comes to delays through a clock tree network. That is why it is recommended that the output of a clock tree be connected to a port through a suitable buffer.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the direct connection from the source clock to the port. To resolve this violation, insert the buffer between the clock port and clock net.

Message 2

The following message appears when an output port <port-name> of design/block <name> is driven by a clock tree net of clock <clk-name> through a non-library cell instance <inst-name>:

```
[Clk_Gen07_02][INFO] In design/block "<name>", port "<port-name>" is driven by a clocktree net (of clock <clk-name>) through a non-library buffer cell instance "<inst-name>"
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

All messages of this rule have the same consequence of not fixing. Refer to [Consequences of Not Fixing](#).

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows a non-library gate instance driving the port. To resolve this violation, insert the buffer between the instance and port.

Message 3

The following message appears when an output port *<port-name>* of design/block *<name>* is driven by a clocktree net of clock *<clk-name>* through a library cell instance *<inst-name>* that has at least one more fan-out:

```
[Clk_Gen07_03][INFO] In design/block "<name>", port "<port-name>" is driven by a clocktree net (of clock <clk-name>) with a combinatorial gate instance <inst-name> with at least one more fan-out
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

All messages of this rule have the same consequence of not fixing. Refer to [Consequences of Not Fixing](#).

How to Debug and Fix

View the incremental schematic of the violation message.

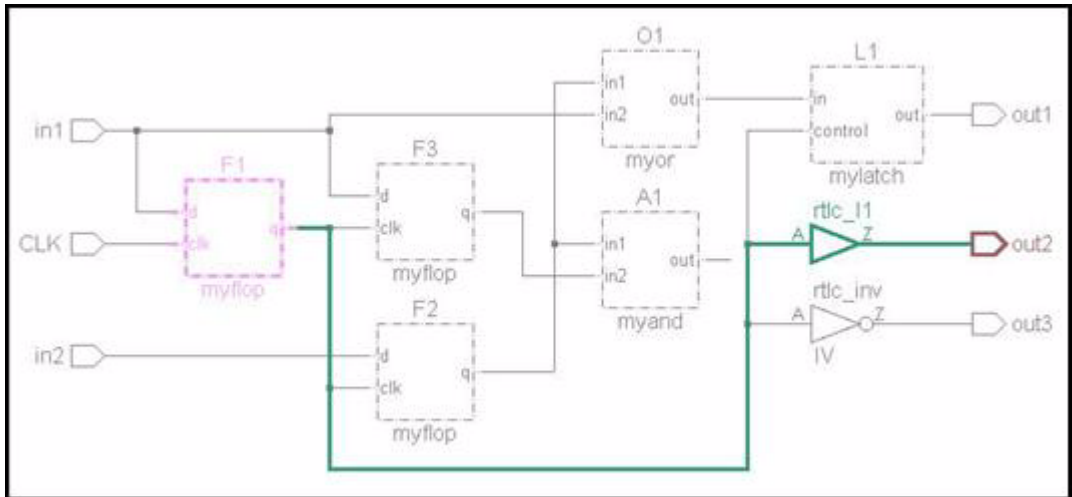
The schematic shows a library gate instance that is driving at least one more gate or port. To resolve this violation, insert a buffer between the instance and port so that there is determinism. Alternatively, rework the logic to ensure that the gate drives only this port.

Example Code and/or Schematic

Clock nets should connect to an output port through a dedicated buffer, that is, this buffer should not be driving any nets. If an output port is connected directly to a clock tree net, then all the slopes and delays after this net in the clocktree depend strongly on the load capacitance of the output port.

In the following schematic, the *Clk_Gen07* rule reports a violation because the generated clock applied on F1/q is connected to the output port directly. The clock net connected to the output port should be connected through some dedicated buffer. *Rtlc_I1* is a buffer generated by SpyGlass.

Clock Rules

**Default Severity Label**

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen08

Reports clocks generated on an object that should not be a port

When to Use

To detect whether a generated clock is given on a port. This rule is applicable to all design phases.

Description

The *Clk_Gen08* rule reports generated clocks created on an input/output/inout port of a block. These clocks are specified using the [create_generated_clock](#) constraint.

This rule supports the [Source Synchronous Interfaces](#) feature.

Parameter(s)

- *tc_source_syn_clks*: Default is yes and all dependent rules will the [Source Synchronous Interfaces](#) feature. This impacts generated clocks (divide_by 1 only) applied at inout/output ports.

Constraint(s)

SDC

- [create_generated_clock](#) (Mandatory): Use to create a clock

Messages and Suggested Fix

The following message appears for a generated clock `<clk-name>` (specified in a [create_generated_clock](#) constraint) that has been created on port `<port-name>` of design/block `<name>`:

```
[Clk_Gen08_01][ERROR] Generated clock "<clk-name>" has been
created on <port-type> port "<port-name>" for design/block
<name>
```

Or

```
[Clk_Gen08_02][WARNING] Generated clock "<clk-name>" has been
created on <port-type> port "<port-name>" for design/block
<name>
```

Potential Issues

The violation message explicitly states the potential issues.

The severity violation message varies with the type of port. For an input port, the severity is ERROR. For an inout or output port, the severity is WARNING.

Consequences of Not Fixing

The consequence depends on the type of port being reported in the violation message.

1. Having a generated clock on an input port is not correct because, a clock has to be generated inside the block. For generating the clock, you require some other signal. This signal can be available on an input port, and then, generation needs may take place beyond the input port.
2. Having a generated clock on an output port might not serve much purpose. Since the clock is generated at the output port, so, it cannot be used anywhere inside the block.
3. If a generated clock is set on a port, it is possible that inside the block there is clock tree net that is directly connected to this port. This could result in delays in the clock tree network. You may want more deterministic approach when it comes to delays through a clock tree network. It is recommend that the output of a clock tree be connected to a port through a suitable buffer.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the generated clock is defined on an input/output/inout port of a block.

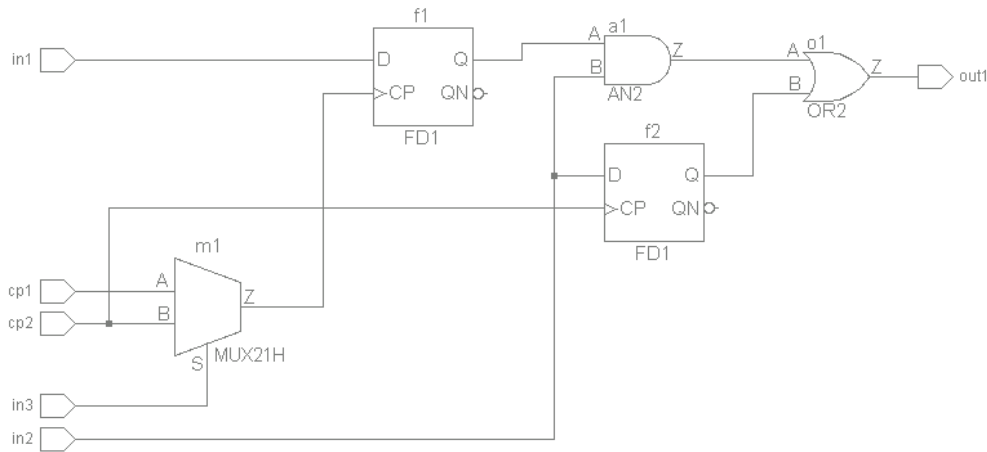
To resolve this violation, review the clock applied on the port and determine the reason for applying it. Most probably it should be removed and redefined.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a message. The schematic of the top module is as follows:



The top .sdc is as follows:

```
top.sdc | top.sdc (1) |
create_clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } { cp1 }
create_generated_clock -name gclk1 -source cp1 -multiply_by 2 { in1[1] in1[3] }
```

This rule reports a violation message because the generated clock is defined on the input port.

Example 2

Test Case Files Not Available

The *Clk_Gen08* rule reports a violation because the generated clock gcl is applied on the port.

```
create_generated_clock -name gcl -source p1 -divide_by 1 p1
```

Default Severity Label

Error/Warning

Clock Rules

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen09

Reports a clock source pin that is in the fan-out of another clock, but is not generated by that clock

When to Use

To check whether *create_clock* is defined in the fan-in of another *create_clock*. This rule is applicable to all design phases.

Description

The *Clk_Gen09* rule reports clocks, specified using the *create_clock* constraint, which have a source pin in the fan-out of the source pin of another clock, specified using the *create_clock* or *create_generated_clock* constraint, but is not generated by the latter clock, using the *create_generated_clock* constraint.

Rule Exceptions

The *Clk_Gen09* rule does not check for clocks (specified using the *create_clock* constraint) set on input and inout ports.

Parameter(s)

- *clk_gen09_trav_over_sequential_arc*: Default is yes. Set this parameter to no to stop the traversal at a sequential arc while checking whether the source clock is in the fan-in of the generated clock.

Constraint(s)

SDC

- *create_clock* (Mandatory): Use to create a clock
- *create_generated_clock* (Mandatory): Use to create a clock

Messages and Suggested Fix

The following message appears for a clock *<clk1-name>* whose source pin *<pin1-name>* is in the fan-out of the source pin *<pin2-name>* of another clock *<clk2-name>* but the first clock *<clk1-name>* is not generated from the second clock *<clk2-name>* using the *create_generated_clock* constraint:

[WARNING] Source port/pin "<pin1-name>" of clock "<clk1-name>" which falls in fan-out of created clock "<clk2-name>" with source port/pin "<pin2-name>" is expected to be generated from latter clock for design/block "<name>"

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Constraining and generating with a wrong clock can allow a greater slack than available. Even though the timing analysis is successful, but chip might eventually fail timing.

How to Debug and Fix

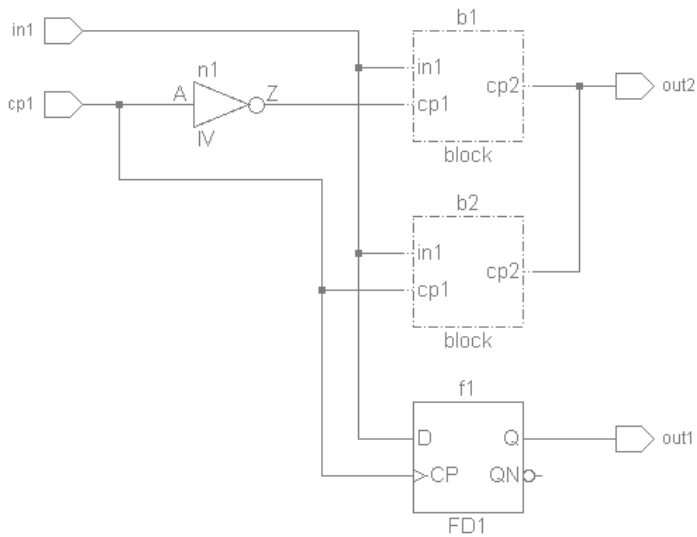
View the incremental schematic of the violation message.

The schematic shows the path from the source port/pin of the create clock to the source port/pin of fan-in of the created clock. To resolve this violation, remove or redefine the clock.

Example Code and/or Schematic

Example 1[View Test Case Files](#)

This example shows when this rule reports a message. The schematic of the top module is as follows:



The top.sdc is as follows:

```
top.sdc | top.sdc (2) |
create_clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } {cp1}
create_clock -name gclk1 -period 20.000000 -waveform { 0.000000 10.000000 } b1/cp1
create_clock -name gclk2 -period 20.000000 -waveform { 0.000000 10.000000 } b2/cp1
```

This rule reports a violation message because the clock gclk1 is a fan-out of the clock clk1, but gclk1 is not generated using clk1.

Example 2

Test Case Files Not Available

The *Clk_Gen09* rule reports a violation because the clock C1 is in the fan-in

Clock Rules

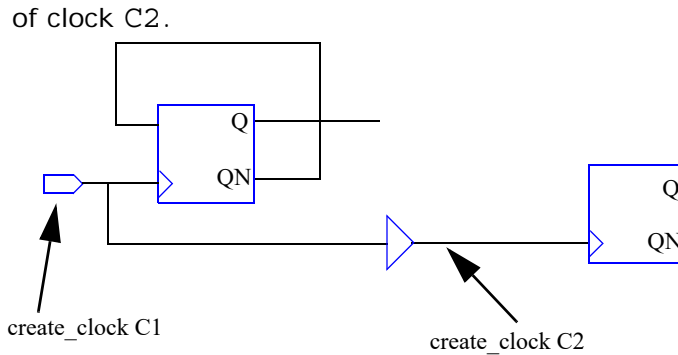


FIGURE 1. Schematic showing when Clk_Gen09 reports a violation message

To resolve the violation, remove clock C2. If you remove clock c1, the [Clk_Gen01a](#) rule reports a violation because the first flip-flop is unconstrained.

Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen10

Identifies clocks that must be propagated in post-layout analysis

When to Use

This rule is applicable to the Post-layout phase.

Description

The *Clk_Gen10* rule flags clocks in a post-layout SDC file, which do not have a corresponding *set_propagated_clock* constraint. You specify these clocks using the *create_clock* and *create_generated_clock* constraint.

Parameter(s)

None

Constraint(s)

SDC

- *create_clock* (Mandatory): Use to create a clock
- *create_generated_clock* (Mandatory): Use to create a clock

Messages and Suggested Fix

The following message appears when a clock *<clk-name>* (specified using the *create_clock* and *create_generated_clock* constraint) does not have the corresponding *set_propagated_clock* constraint:

```
[INFO] set_propagated_clock constraint not set for clock  
"<clk-name>" in post-layout analysis
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

In the post-layout phase, the netlist is fully defined and the actual timing for the clock paths are known. Not propagating clocks at this stage results in incorrect timing.

How to Debug and Fix

The *set_propagated_clock* constraint is not provided for each clock in the

design. A violation message is reported for clocks that do not have the [set_propagated_clock](#) constraint set.

To fix this violation, add the [set_propagated_clock](#) constraint.

Example Code and/or Schematic

Suppose you have specified only one constraint in the SDC file:

```
create_clock -name c1 -period 20 p1
```

The *Clk_Gen10* rule reports a violation because the [set_propagated_clock](#) constraint is not specified in the SDC file. Therefore, the correct method to specify is:

```
create_clock -name c1 -period 20 p1
set_propagated_clock c1
```

Default Severity Label

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen13

Identifies clock names that do not follow the naming convention

When to Use

Some methodologies use a default naming convention. However, if you are using a tool that does not have a default naming convention, use this rule to enforce a clock naming convention.

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen13* rule reports clocks that either do not have a name explicitly specified or have the same explicit name as any port/pin.

This rule checks names in a case-sensitive manner. Therefore, CLK and Clk are considered different names.

Parameter(s)

None

Constraint(s)

SDC

- *create_clock* (Optional): Use to create a clock.
- *create_generated_clock* (Optional): Use to create a clock.

Messages and Suggested Fix

Message 1

The following message appears for a *create_clock* or a *create_generated_clock* specification that does not have the `-name` argument:

[INFO] Name of the clock is not set explicitly

Message 2

The following message appears for a clock `<clk-name>` (specified using the *create_clock* and *create_generated_clock* constraint) that has the same explicit name (specified using `-name` argument) as its port/pin:

[INFO] Explicit name of the clock "`<clk-name>`" is same as its

port/pin name

Message 3

The following message appears for a clock `<clk-name>` (specified using the `create_clock` and `create_generated_clock` constraint) that has the same explicit name (specified using `-name` argument) as any port/pin:

```
[INFO] Explicit clock name "<clk-name>" matches with port name of block "<blk-name>"
```

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

The benefits of using a naming convention are:

- Improves readability
- Reduces errors to interpretation
- Improves the understanding
- Makes it efficient for the people to find issues, if any

How to Debug and Fix

Double-click the violation message to view the definitions of the clock and port.

Example Code and/or Schematic

Example 1

Provide an example of a correctly defined clock name.

```
create_clock -name clk -period 10.000000 -waveform { 0.000000
5.000000 } clk
```

In this example, the name of the clock is the same as port name.

Example 2

In this example, the name of the clock is not set.

```
create_clock -period 5.000000 [find port CLK]
```

Default Severity Label

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen14

Reports virtual clock names that do not follow the naming convention

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen14* rule flags virtual clocks that do not follow the recommended naming convention.

Rule Exceptions

The *Clk_Gen14* rule runs only if you specify either the *cg14_suffix* or *cg14_prefix* parameters with a valid string, or you specify both parameters. Otherwise, this rule reports a violation.

Parameter(s)

- *cg14_prefix*: Default is unspecified. Set the value to the permitted virtual clock signal name prefix.
- *cg14_suffix*: Default is unspecified. Set the value to the permitted virtual clock signal name suffix.

Constraint(s)

SDC

- *create_clock* (Optional): Use to create a clock.
- *set_input_delay* (Optional): Use to specify the delay value of the data path.
- *set_output_delay* (Optional): Use to specify the delay value of the data path.

Messages and Suggested Fix

Message 1

The following message appears if neither the value of the *cg14_suffix* nor *cg14_prefix* parameters is not specified.

[INFO] Please specify a value for the parameter `cg14_suffix` and/or `cg14_prefix`

Message 2

The following message appears for the virtual clock `<clkv-name>` in the block `<name1>` that does not follow the recommended naming convention:

[INFO] Virtual clock "`<clkv-name>`" in block "`<name1>`" (corresponding real clock "`<clkr-name>`") should be named `<clkv-name>_<value-of-cg14_suffix>`

Where, `<clkr-name>` is the name of the corresponding master (real) clock.

Message 3

The following message appears for a virtual clock `<clkv-name>` in block `<name1>` that does not follow the recommended naming convention:

[INFO] Virtual clock "`<clkv-name>`" in block "`<name1>`" (corresponding real clock "`<clkr-name>`") should be named `<value-of-cg14_prefix>_<clkr-name>`

Where, `<clkr-name>` is the name of the corresponding master (real) clock.

Message 4

The following message appears for a virtual clock `<clkv-name>` in block `<name1>` that does not follow the recommended naming convention:

[INFO] Virtual clock "`<clkv-name>`" in block "`<name1>`" (corresponding real clock "`<clkr-name>`") should be named `<value-of-cg14_prefix>_<clkr-name>_<value-of-cg14_suffix>`

Where, `<clkr-name>` is the name of the corresponding master (real) clock.

Potential Issues

Not applicable.

Consequences of Not Fixing

Not applicable.

How to Debug and Fix

To identify whether a virtual clock refers to a real clock, traverse forward/backward from the input/output port on which *set_input_delay*/*set_output_delay* constraint is specified with respect to the virtual clock being considered. If while traversing, the data/output pin of a sequential cell is reached, then compare the timing characteristics of this virtual clock with the real clock which is sampling the sequential cell. If the timing characteristics of the virtual clock matches with the real clock, the check for the naming convention as follows:

`<virtual-clk-name> = <prefix>_<real-clock-name>_<suffix>`

If no real clock is found, the Clk_Gen14 rule does not report a violation.

Example Code and/or Schematic

Example 1 - How to Create a Virtual Clock

You can create virtual clocks by using the *create_clock* constraint with the name argument instead of referencing a real port or pin. The syntax to name the virtual clock is:

`<virtual-clk-name> = <prefix>_<real-clock-name>_<suffix>`

Where,

<code><virtual-clk-name></code>	Name of the virtual clock
<code><prefix></code>	Recommended prefix specified using the <i>cg14_prefix</i> parameter
<code><real-clock-name></code>	Name of its master (real) clock
<code><suffix></code>	Recommended suffix specified using the <i>cg14_suffix</i> parameter

If the *cg14_suffix* and/or *cg14_prefix* parameters are not specified, the *Clk_Gen14* rule reports a violation that states you should specify a parameter.

Suppose, the constraints are specified as follows:

```
create_clock -name Clk1 -period 10 {CLK1}
create_clock -name vClk1 -period 10
```

```
set_input_delay 2.3 -clock vClk1 {in1}
```

The *cg14_prefix* parameter is set to v and the in1 port is feeding into the D pin of the flip-flop. The *Clk_Gen14* rule reports a violation that states the clock name v_Clk1 should be replaced with vClk1.

Default Severity Label

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen15

Identifies `create_clock` that do not have a waveform specified

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen15* rule reports `create_clock` constraints without the waveform specifications.

Parameter(s)

None

Constraint(s)

SDC

- `create_clock` (Mandatory): Use to create a clock.

Messages and Suggested Fix

The following message appears for a `create_clock` constraint for the clock `<clk-name>` without the waveform specifications:

```
[INFO] Option "waveform" not defined in command "create_clock"
for clock "<clk-name>"
```

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

This check is a methodology choice since most tools will default a 50% duty-cycle waveform in the absence of a specification.

How to Debug and Fix

To resolve this violation, open the SDC file and specify the waveform.

Example Code and/or Schematic

Example 1

The following snippet shows a correctly defined *create_clock* that has a waveform specification.

```
create_clock -name c1 -period 10 -waveform {0 5} clk
```

Example 2

The following snippet shows a *create_clock* constraint that does not have a waveform specification.

```
create_clock -name c1 -period 10 clk
```

Default Severity Label

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen17

Detects divided or multiplied clocks

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The Clk_Gen17 rule reports [create_generated_clock](#) constraints that have `divide_by` or `multiply_by` clock generation methods specified.

Parameter(s)

None

Constraint(s)

SDC

- [create_generated_clock](#) (Mandatory): Use to create a clock

Messages and Suggested Fix

Message 1

The following message appears for a [create_generated_clock](#) constraint for clock `<clk-name>` with the `-divide_by` option:

```
[INFO] Divided clock <clk-name> detected
```

Message 2

The following message appears for a [create_generated_clock](#) constraint for clock `<clk-name>` with the `-multiply_by` option:

```
[INFO] Multiplied clock <clk-name> detected
```

Potential Issues

Not applicable

Consequences of Not Fixing

Not applicable

How to Debug and Fix

Refer to the Example Code and/or Schematic section for examples of `divide_by` or `multiply_by` specifications.

Example Code and/or Schematic

Example 1: `divide_by`

This example shows a divided clock CLK1:

```
create_generated_clock -name CLK1 \  
  -source [get_ports CLKIN] \  
  -divide_by 2 [get_pins FF0/Q]
```

Since `divide_by` is used, SpyGlass generates the following message:

[INFO] Divided clock CLK1 detected

Example 2: `multiply_by`

This example shows a multiplied clock CLK2:

```
create_generated_clock -name CLK2 \  
  -source [get_ports CLKIN] \  
  -multiply_by 2 [get_pins FF0/Q]
```

Since `multiply_by` is used, SpyGlass generates the following message:

[INFO] Multiplied clock CLK2 detected

Default Severity Label

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen18

Detects edge derived clock

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen18* rule identifies *create_generated_clock* constraints that have edges or edge_shift clock generation methods specified.

Parameter(s)

None

Constraint(s)

SDC

- *create_generated_clock* (Mandatory): Use to create a clock.

Messages and Suggested Fix

The following message appears for a *create_generated_clock* constraint for clock *<clk-name>* with the *-edges/-edge_shift* options:

[INFO] Edge derived clock <clk-name> detected

Potential Issues

Not applicable

Consequences of Not Fixing

Not applicable

How to Debug and Fix

Not applicable

Example Code and/or Schematic

This example has an edge derived clock CLK1 specified. Therefore, SpyGlass reports a violation message.

```
create_generated_clock -name CLK1 \
```

```
-source [get_ports CLKIN] \  
-edges {1 3 5} -edge_shift {0 1 0} [get_pins FF0/Q]
```

Default Severity Label

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen19

Detects `add` or `master_clock` arguments in `create_generated_clock`

When to Use

This rule is applicable to the RTL phase.

Description

The *Clk_Gen19* rule reports *create_generated_clock* constraints with unusual arguments, such as `add` or `master_clock`.

Parameter(s)

None

Constraint(s)

SDC

- *create_generated_clock* (Mandatory): Use to create a clock

Messages and Suggested Fix

The following message appears for a *create_generated_clock* constraint for clock `<clk-name>` with the `add` and/or `master_clock` argument:

Unusual option(s) "<option-list>" not supported by certain synthesis tools used in command "create_generated_clock"

Where, `<option-list>` can be a combination of `add` and `master_clock`.

Potential Issues

Not applicable

Consequences of Not Fixing

Some *create_generated_clock* arguments are not supported by certain synthesis tools. Therefore, do not use these options.

How to Debug and Fix

Remove `add` and `master_clock` arguments.

Example Code and/or Schematic

For the following snippet, the Clk_Gen19 rule reports a violation.

```
create_clock -name c1 -period 20 in
create_generated_clock -name gc1 -divide_by 2 -master_clock
c1 -add out
```

To fix the violation message, remove the `add` and `master_clock` arguments.

Default Severity Label

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen20

Reports clocks that are propagated in synthesis and/or pre-layout timing-analysis

When to Use

This rule is applicable to the RTL and Pre-layout phases.

Description

The *Clk_Gen20* rule reports the [set_propagated_clock](#) constraints specified for clocks, using the [create_clock](#) and [create_generated_clock](#) constraint, in the synthesis and pre-layout constraint scripts.

The *Clk_Gen20* rule checks real clocks and generated clocks only.

Rule Exceptions

The *Clk_Gen20* rule does not check for virtual clocks.

Parameter(s)

None

Constraint(s)

SDC

- [set_propagated_clock](#) (Mandatory): Use to specify delays propagated through the clock network to determine latency at register clock pins.
- [create_clock](#) (Mandatory): Use to create a clock
- [create_generated_clock](#) (Mandatory): Use to create a clock

Messages and Suggested Fix

The following message appears when a clock `<clk-name>` (specified using the [create_clock](#) and [create_generated_clock](#) constraint) in the synthesis and prelayout constraint scripts also has a [set_propagated_clock](#) constraint:

```
[INFO] set_propagated_clock should not be set for clock
"<clk-name>" in synthesis/prelayout script
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

This is not meaningful since clock trees are not fully defined and we do not know the actual timing for the clock paths.

How to Debug and Fix

A violation message is reported for clocks that have the [set_propagated_clock](#) constraint set. Do not specify the [set_propagated_clock](#) constraint for any clock in the design.

To resolve this violation, remove the [set_propagated_clock](#) constraint from the SDC file.

Example Code and/or Schematic

The *Clk_Gen20* rule reports a violation because the [set_propagated_clock](#) constraint is specified.

```
create_clock -name c1 -period 20 p
set_propagated_clock c1
```

Remove the [set_propagated_clock](#) constraint to resolve the violation.

Default Severity Label

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen21

Reports when `set_propagated_clock` is set on a virtual clock

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen21* rule reports `set_propagated_clock` constraints set on virtual clocks.

Parameter(s)

None

Constraint(s)

SDC

- `set_propagated_clock` (Mandatory): Use to specify delays propagated through the clock network to determine latency at register clock pins.
- `create_clock` (Mandatory): Use to create a clock. Virtual clocks are created using the `create_clock` constraint with the `-name` argument instead of referencing a real port/pin.

Messages and Suggested Fix

The following message appears for a virtual clock `<clkv-name>` that has `set_propagated_clock` constraint set:

```
[WARNING] set_propagated_clock has been set on virtual clock
"<clkv-name>"
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Setting the `set_propagated_clock` constraint on a virtual clock is incorrect because there is no design object on which virtual clocks are applied. Therefore, there is nothing to propagate for virtual clocks.

How to Debug and Fix

View the incremental schematic of the violation message.

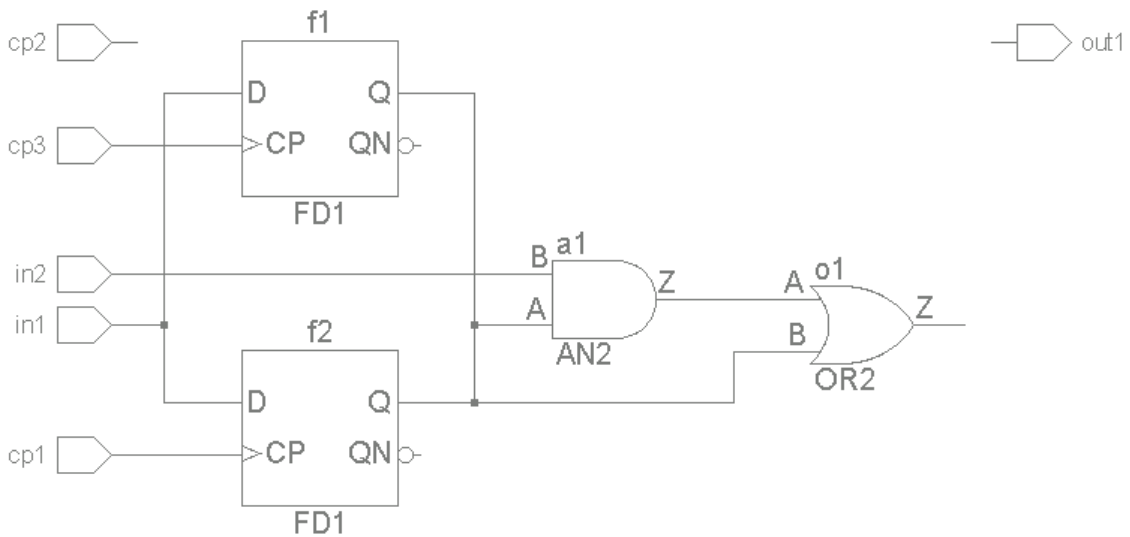
The schematic shows the *set_propagated_clock* constraint set on virtual clocks in the SDC file. To resolve this violation, remove the highlighted constraint from the SDC file.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a message. The schematic of the top module is as follows:



The top .sdc is as follows:

Clock Rules

```

top.sdc | top.sdc (2)
create_clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } { cp1}
create_clock -name vclk1 -period 10.000000 -waveform { 0.000000 5.000000 }
create_clock -name vclk2 -period 10.000000 -waveform { 0.000000 5.000000 }
set_clock_transition 10 {vclk1}
set_propagated_clock {vclk1}
set_propagated_clock {vclk2}

```

This rule reports a violation message because the `set_propagated_clock` constraint is defined on a virtual clock `vclk1`.

Example 2

Test Case Files Not Available

The `Clk_Gen21` rule reports a violation because the `set_propagated_clock` is applied on virtual clock `c1`.

```

create_clock -name c1 -period 30
set_propagated_clock c1

```

Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen22

Reports `set_input_delay`/`set_output_delay` that have been specified on a clock port

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen22* rule reports clock ports that have a `set_input_delay` or `set_output_delay` constraint specified. A port is a clock port if the `create_clock` constraint has been set on an input/inout port or a `create_generated_clock` is set on an output/inout port.

If both `set_input_delay` and `set_output_delay` constraints are set on an inout (clock) port, then the *Clk_Gen22* rule reports two violations, one for each delay specification.

Parameter(s)

None

Constraint(s)

SDC

- `set_input_delay` (Mandatory): Use to set input delay on pins or output ports.
- `set_output_delay` (Mandatory): Use to set output delay on pins or output ports.
- `create_clock` (Optional): Use to create a clock.
- `create_generated_clock` (Optional): Use to create a clock.

Messages and Suggested Fix

The following message appears for a clock port `<port-name>` that has an unsupported constraint `<constr>` set:

[WARNING] `<constr>` has been set on a clock port "`<port-name>`"

Where, `<constr>` can be `set_input_delay` or `set_output_delay`.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Specifying input/output delay on a clock port is incorrect. The [set_clock_latency](#) constraint should be used for this purpose. If an output delay is specified on a clock port, then implementation tools can unnecessarily insert buffers to meet this.

Clock tree synthesis may fail or result in an unbalanced clock tree.

How to Debug and Fix

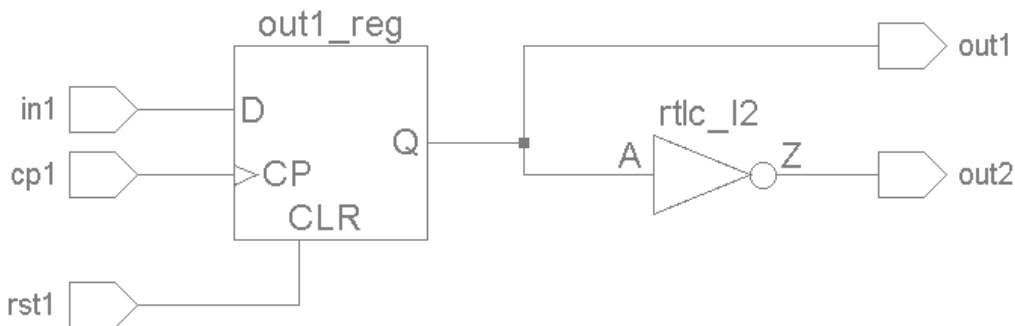
View the incremental schematic of the violation message.

Inspect the SDC file for ports/pins that have the [set_input_delay](#) or [set_output_delay](#) constraints specified and have clocks ([create_clock](#) and [create_generated_clock](#)) declared on them.

To resolve this violation, remove the constraints that are not required.

Example Code and/or Schematic**Example 1**
[View Test Case Files](#)

This example shows when this rule reports a message. The schematic of the top module is as follows:



The top.sdc is as follows:

```
top.sdc | top.sdc (2) | test.sgcd (1) |
create_clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } {cp1}
set_input_delay 0.000000 -clock clk1 [all_inputs]
```

This rule reports a violation message because a [set_input_delay](#) constraint is defined on the clock port cp1 of the design.

Example 2

Test Case Files Not Available

Suppose, in1 is an input port and the constraints applied are:

```
create_clock -name c1 -period 20 in1
create_clock -name c2 -period 20 in2
set_input_delay 1 -clock c1 in1
```

The *Clk_Gen22* rule reports a violation for the input port in1 because the clock and input delay are applied to it. To resolve the violation, you can remove the [create_clock](#) constraint applied on in1 or the [set_input_delay](#) constraint applied on in1.

Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen23

Reports an incorrectly defined generated clock

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen23* rule performs checks to verify if a generated clock is defined correctly with respect to its source object as far as their waveforms are concerned. The *Clk_Gen23* rule requires that the waveform of the generated clock is derived correctly from the waveform of the source clock.

From a source object, a generated clock can be derived using either `-divide_by`, `-multiply_by`, or `-edges` options. The rule checks generated clocks derived using the `-divide_by 1` (or equivalently `-combinational`) option, the `-divide_by 2` option, or the `-edges` option.

This section describes the following:

- [Handling of the `-divide by 1` or `-combinational` option](#)
- [Handling of the `divide-by-2` option](#)
- [Handling of the `-edges` option](#)
- [Rule Exceptions](#)

Handling of the `-divide by 1` or `-combinational` option

For a generated clock derived using `-divide_by 1` (or equivalently `-combinational`) option, all of the following statements hold true:

- There must exist at least one combinational path from the source object to the generated clock.
- If there is no net inversion of polarity along the paths from the source object to the generated clock, the `-invert` option should not be used along with the `-divide_by 1` or `-combinational` option.
- If there is a net inversion of polarity along the paths from the source object to the generated clock, the `-invert` option must be used along with the `-divide_by 1` or `-combinational` option.

Handling of the divide-by-2 option

After you use the `-divide_by_2` option to define a generated clock, there should be a divide-by-2 flip-flop lying on the path from the source object to the generated clock object. The *Clk_Gen23* rule reports a violation when no divide-by-2 flip-flop is found in the path.

If the *strict* parameter is set to *yes*, the *Clk_Gen23* rule assumes the first edge is the rising edge of the waveform. If the *strict* parameter is set to *no*, the first edge is either a rising or falling edge of the waveform.

The following table describes the scenarios in which the *Clk_Gen23* rule reports a violation. The columns of this table are as follows:

- **Strict Parameter:** Refers to whether the *strict* parameter has been set
- **Clock Path:** Refers to the path from the source design object of the generated clock to the clock pin of the divide-by-2 flip-flop.
- **Q Path:** Refers to the path from the Q pin of the divide-by-2 flip-flop to the point where the generated clock is applied.
- **Invert Option:** Refers to whether the `-invert` option has been applied on the generated clocks defined with the `-divide_by_2` option.

Strict Parameter	Clock Path	Q Path	Invert Option	Violation
Yes	+ve	+ve	Yes	Yes
Yes	+ve	-ve	No	Yes
No	+ve	+ve	Yes or No	No
No	+ve	-ve	Yes or No	No
Yes	+ve	+ve & -ve	Yes or No	Yes
No	+ve	+ve & -ve	Yes or No	Yes
Yes	+ve	+ve	Yes & No	Yes
No	+ve	+ve	Yes & No	Yes
Yes	+ve	-ve	Yes & No	Yes
No	+ve	-ve	Yes & No	Yes

As you can see in the table above, there is no polarity inversion in the Clock Path column. When there is polarity inversion, do not use the -

`divide_by` option to generate clocks. Rather, use the `-edges` option.

Handling of the `-edges` option

You can also specify a generated clock by using the `-edges` option. When you use this option, you specify the edges of the master clock that are used to form the edges of the generated clock. The edges of the master clock are interpreted as alternating rising and falling edges of the generated clock.

For example, `-edges 1 3 5` indicates:

- edge 1 of the master clock is the first rising edge of the generated clock,
- edge 3 of the master clock is the first falling edge of the generated clock, and
- edge 5 is the next rising edge of the generated clock.

Depending upon whether there is a path inversion on the clock-path or the Q-path, the following scenarios exist:

- **There is no path inversion on clock-path:** Here the rising and the falling edges of the generated clock are aligned to the rising edge of the master clock. If you use the `-edges` option, all edges of the generated clock should be an odd-numbered edge of the master clock.
- **There is path inversion on clock-path:** Here the rising and the falling edges of the generated clock are aligned to the falling edge of the master clock. If you use the `-edges` option, all edges of the generated clock should be an even-numbered edge of the master clock.

Rule Exceptions

The `Clk_Gen23` rule handles only `divide_by 1` and `divide_by 2` clocks. In addition, this rule does not report violations for `divide_by 2` clocks if any sequential elements is present in the fan-ins of a mux and the source object of the generated clock is present in the fan-in of the control pin.

Parameter(s)

- *strict*: Default is no. Refer to the [Handling of the divide-by-2 option](#) section for information on how this parameter impacts the `Clk_Gen23` rule.

Constraint(s)

SDC

- *create_clock* (Optional): Use to create a clock.
- *create_generated_clock* (Optional): Use to create a clock.

Messages and Suggested Fix

Message 1

The following message appears for a generated clock *<gclk-name>* specification for the design/block *<name>* when the path from the source object of the generated clock to the port/pin of the generated clock:

- lacks divide-by-2 flip-flop and the specified *<divide_by_factor>* is 2, or
- contains a sequential cell and the specified *<divide_by_factor>* is 1

[WARNING] Generated clock "*<gclk-name>*" should not be specified with a divide_by factor of *<divide_by_factor>* for design/block "*<name>*"

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Deriving a waveform incorrectly from a source clock can result in tools to assume a greater slack than available. Even though the timing analysis may run successfully, the chip might eventually fail timing.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the number of sequential elements between the specified source and the point of *create_generated_clock*. The factor specified is incorrect.

To resolve this violation, correct the *divide_by* factor based on the sequential elements in the path.

Message 2

The following message appears for a generated clock *<gclk-name>*

specification for the design/block *<name>* when the path from the source object of the generated clock to the port/pin of the generated clock is:

- Combinational, with *<divide_by_factor>* specified as 1, and there is net inversion of polarity but no *-invert* option is used, or
- Sequential with no clock inversion, *<divide_by_factor>* specified as 2, and there is Q-path inversion but no *-invert* option is used.

[WARNING] Generated clock "*<gclk-name>*" should be defined with option *-invert*, along with *-divide_by* for design/block "*<name>*" as the generated clock object is being inverted

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Deriving a waveform incorrectly from a source clock can result in tools to assume a greater slack than available. Even though the timing analysis may run successfully, the chip might eventually fail timing.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the path from the specified source to the inverting point of the generated clock. The specified source could be different from the point of master clock declaration.

To resolve this violation, apply the *invert* option with the generated clock.

Message 3

The following message appears for a generated clock *<gclk-name>* specification for the design/block *<name>* when the path from the source object of the generated clock to the port/pin of the generated clock is:

- Combinational, with *<divide_by_factor>* specified as 1, and there is no net inversion of polarity but the *-invert* option is used, or
- Sequential with no clock inversion, *<divide_by_factor>* specified as 2, and there is no Q-path inversion but the *-invert* option is used.

[WARNING] Generated clock "*<gclk-name>*" should not be defined with option *-invert*, along with *-divide_by* for design/block "*<name>*" as the generated clock object is not being inverted

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Deriving a waveform incorrectly from a source clock can result in tools to assume a greater slack than available. Even though the timing analysis may run successfully, the chip might eventually fail timing.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the path from the specified source to the non-inverting point of the generated clock. The specified source could be different from the point of master clock declaration.

To resolve this violation, remove the `invert` option with the generated clock.

Message 4

The following message appears for a generated clock `<gclk-name>` specification for the design/block `<name>` when:

- the path from the source object of the generated clock to the port/pin of the generated clock is sequential,
- `<divide_by_factor>` is greater than one, and
- clock-path has polarity inversion.

In this case, clock-path refers to the path from the source object of the generated clock to the clock-pin of the sequential cell.

[WARNING] Generated clock `<gclk-name>` should be defined either with different source or with option `-edges` with an `edgelist` whose edges should be falling edges of master clock, instead of `-divide_by` option for design/block `<name>`

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Deriving a waveform incorrectly from a source clock can result in tools to assume a greater slack than available. Even though the timing analysis may run successfully, the chip might eventually fail timing.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows a falling edge at the specified source that can trigger the clock-generating element. Therefore, move the source either after or before the inverting element. Alternatively, align the edges specified with the falling edge of the source.

To resolve this violation, define the `edgelist` option for the generated clock.

Message 5

The following message appears for a generated clock `<gclk-name>` specification for the design/block `<name>` when

- the path from the source object of the generated clock to the port/pin of the generated clock is sequential,
- `clock-path` has no polarity inversion, and
- the `-edges` option is used with an even-numbered edge of the master clock.

In this case, `clock-path` refers to the path from the source object of the generated clock to the clock-pin of the sequential cell.

[WARNING] Generated clock "`<gclk-name>`" should be defined with option `-edges`, with an `edgelist` whose edges should be rising edges of master clock (like `<edge-list>`) for design/block "`<name>`"

Where, `<edge-list>` is the suggested edge list specification.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Deriving a waveform incorrectly from a source clock can result in tools to assume a greater slack than available. Even though the timing analysis may run successfully, the chip might eventually fail timing.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows a rising edge at the specified source that can trigger the clock generating element. Align the edges specified with the rising edge

of the source.

To resolve this violation, define the `edge` option for the generated clock.

Message 6

The following message appears for a generated clock `<gclk-name>` specification for the design/block `<name>` when:

- the path from the source object of the generated clock to the port/pin of the generated clock is sequential,
- the clock-path has polarity inversion, and
- the `-edges` option is used with an odd-numbered edge of the master clock.

[WARNING] Generated clock `<gclk-name>` should be defined with option `-edges`, with an `edgelist` whose edges should be a falling edges of master clock (like `<edge-list>`) for design/block `<name>`

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Deriving a waveform incorrectly from a source clock can result in tools to assume a greater slack than available. Even though the timing analysis may run successfully, the chip might eventually fail timing.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows a falling edge at the specified source that can trigger the clock generating element. Align the edges specified with the falling edge of the source.

To resolve this violation, define the `odd` option for the generated clock.

Message 7

The following message appears only for a generated clock `<gclk-name>` defined by the `-divide_by_2` option for the design/block `<name>` when:

- the path from the source object of the generated clock to the port/pin of the generated clock is sequential

- the clock path is non inverted
- the generated clock is defined with the `-invert` option. A non inverted source clock is also reaching its generated clock

[WARNING] Generated clock `<gclk-name>` is defined with option `-invert` for design/block `<name>`, another generated clock should be defined without option `-invert` as inverted source clock is also reaching to generated clock via one of the paths

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Deriving a waveform incorrectly from a source clock can result in tools to assume a greater slack than available. Even though the timing analysis may run successfully, the chip might eventually fail timing.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows a falling edge at the specified source that can trigger the clock generating element. Align the edges specified with the falling edge of the source.

To resolve this violation, define another generated clock without the `invert` option.

Message 8

The following message appears only for a generated clock `<gclk-name>` defined by the `-divide_by_2` option for the design/block `<name>` when:

- the path from the source object of the generated clock to the port/pin of the generated clock is sequential
- the clock path is non inverted
- the generated clock is defined without the `-invert` option. An inverted source clock is also reaching its generated clock

[WARNING] Generated clock `<gclk-name>` is defined without option `-invert` for design/block `<name>`, another generated clock should be defined with option `-invert` as non inverted source clock is also reaching to generated clock via one of the paths

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Deriving a waveform incorrectly from a source clock can result in tools to assume a greater slack than available. Even though the timing analysis may run successfully, the chip might eventually fail timing.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows a falling edge at the specified source that can trigger the clock generating element. Align the edges specified with the falling edge of the source.

To resolve this violation, define another generated clock with the `invert` option.

Message 9

The following message appears only for a generated clock `<gclk-name>` defined by the `-divide_by_2` option for the design/block `<name>` when:

- the path from the source object of the generated clock to the port/pin of the generated clock is sequential,
- clock-path either has inverted or non-inverted polarity, and
- the generated clocks are defined with and without the `-invert` option.

[WARNING] Either of inverted or non inverted source clock can reach to generated clock's object. Generated clocks `<gclk-name>` have been defined with and without `-invert` option for design/block `<name>`. Only one of them should have been defined

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Deriving a waveform incorrectly from a source clock can result in tools to assume a greater slack than available. Even though the timing analysis may run successfully, the chip might eventually fail timing.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows a falling edge at the specified source that can trigger the clock generating element.

To fix this violation, remove one of the generated clocks from the SDC file.

Message 10

The following message appears when generated clock is defined for one path, but a generated clock is not defined for the path stated in the message:

[WARNING] For <combination/sequential> clock path -from <clk-source-obj> -to <clk-obj>, there is no generated clock defined for design/block <name>

Potential Issues

This message appears because there are multiple paths existing in the clock path as stated in the message. For one of these paths, you have defined the generated clock. However, for the other paths, the rule is expecting another generated clock.

Consequences of Not Fixing

Deriving a waveform incorrectly from a source clock can result in tools to assume a greater slack than available. Even though the timing analysis may run successfully, the chip might eventually fail timing.

How to Debug and Fix

View the incremental schematic of the violation message.

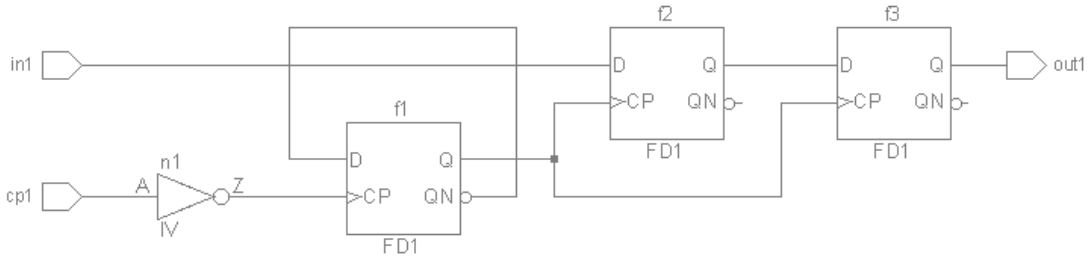
The schematic shows the path for which the generated clock is not defined. To fix this violation, define another generated clock for the path specified in the message.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports [Message 4](#). The schematic of the top module is as follows:



The top.sdc is as follows:

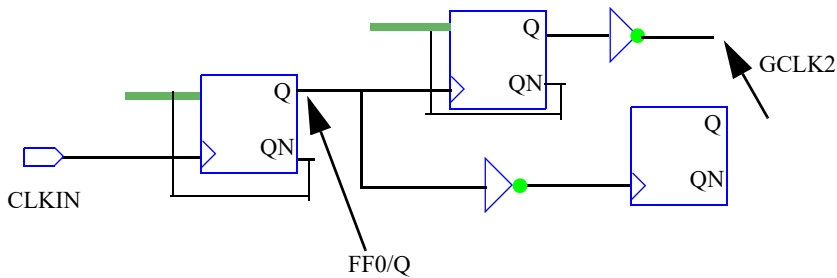
```

top.sdc | top.sdc (1) | test.sgdc (1) |
create_clock -name clk1 -period 10 [get_ports {cp1}]
create_generated_clock -name gclk1 -divide_by 2 -source [get_ports cp1] [get_pins {f1/Q}]
    
```

This rule reports a violation message because the clock **gclk1** is incorrectly defined with reference to the design.

Example 2

Test Case Files Not Available



Assume that master clock CCLK corresponding to port CLKIN is

generating clock GCLK1 from pin FF0/Q. If clock CCLK gets inverted before reaching the flip-flop FF0, then according to DC/PT semantics, the correct definition of GCLK1 should use the `-edges { }` option, with first edge in the option being a falling edge of the master clock, rather than specifying with the `-divide_by` option. Therefore, the following definitions of GCLK1 are flagged as highlighted by the rule:

```
create_clock -name CCLK -period 4.0 CLKIN
create_generated_clock -name GCLK1
    -source [get_ports CLKIN] -edges {2 4 6} FF0/Q
create_generated_clock -name GCLK1
    -source [get_ports CLKIN] -edges {1 3 5} FF0/Q
create_generated_clock -name GCLK1
    -source [get_ports CLKIN] -divide_by 2 -invert FF0/Q
create_generated_clock -name GCLK1
    -source [get_ports CLKIN] -divide_by 2 FF0/Q
create_generated_clock -name GCLK2
    -source [get_pins FF0/Q] -divide_by 2 FF1/Q
```

Similarly, if the GCLK1 waveform inversion occurs after the `divide_by/` `multiply_by` logic from sequential element FF1 but before it reaches the generated object GCLK2, then the `-invert` option should be used with the `-divide_by` option in the GCLK2 definition. This clock can also be defined with correct `-edges { }` option. Here, the *Clk_Gen23* rule flags only if the `-divide_by` option is used without the `-invert` option.

Example 3

 Test Case Files Not Available

Suppose there exists more than one path with different polarities between a generated clock and its source. If a generated clock is not created for one of the polarities, then the generated clocks should be defined corresponding to that polarity. Otherwise, the user may get a violation of this rule even by correcting the definition of the generated clock.

For example:

```
create_generated_clock -name GC -divide_by 1 -source
CP Q
```

If there exists paths with positive and negative polarity between Q and CP, then a violation will be reported for the negative polarity. If you attempt to correct the definition of the generated clock by adding the `-invert` option, then a violation will be reported for the positive polarity. The right way is to define another generated clock on Q with the `-invert` option.

Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen23a

Report for validation of create_generated_clock

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The Clk_Gen23a rule checks the following for all of the generated clocks:

- Whether the generated clock defined in the SDC file is correct with reference to its master clock.
- the missing paths for which there is no generated clock.

Difference between Clk_Gen23 and Clk_Gen23a

The Clk_Gen23a verifies the generated clocks using a formal engine. It provides enhanced debugging capabilities by leveraging the Waveform Viewer. Unlike [Clk_Gen23](#), the Clk_Gen23a rule is not limited to only `divide_by 2` and `divide_by 1` clocks.

Rule Exceptions

The *Clk_Gen23a* rule has the following limitations:

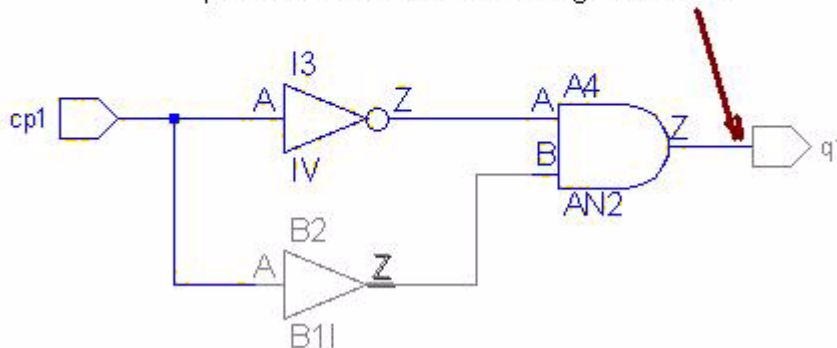
- **The waveform of missing generated clock for comb-path is not reported**

Suppose, the design contains a path from `cp1` to `q1` with both positive and negative polarity, and you have specified a generated clock for positive polarity only. This rule reports the missing generated clock for negative polarity, but it does not report the waveform of the missing

generated clock.

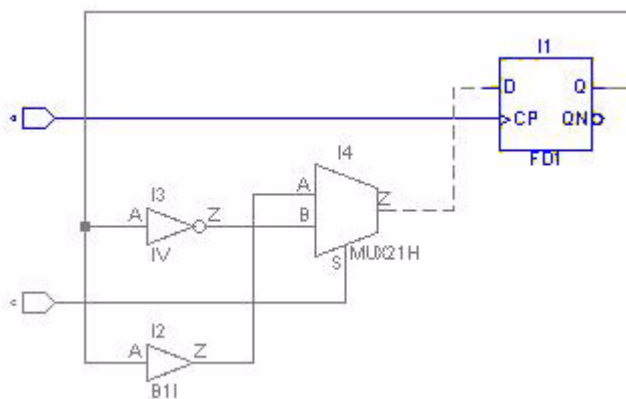
```
create_generated_clock -name G1 -divide_by 1 -source cp1 q1
```

Generated clock not defined on the inverted path - the waveform is not generated.



■ Unable to check multiple generated clock for mux structure in feedback loop

For a mux structure in a feedback loop (Q-D), the Clk_Gen23a rule does not check if generated clocks are specified for all the paths. In the following design, the output of MUX I4 is feed into D-pin of flop and inputs to MUX are inverted and non-inverted signal of Q-pin of the flop. Therefore, there should be two generated clocks defined at output-pin of the flop.

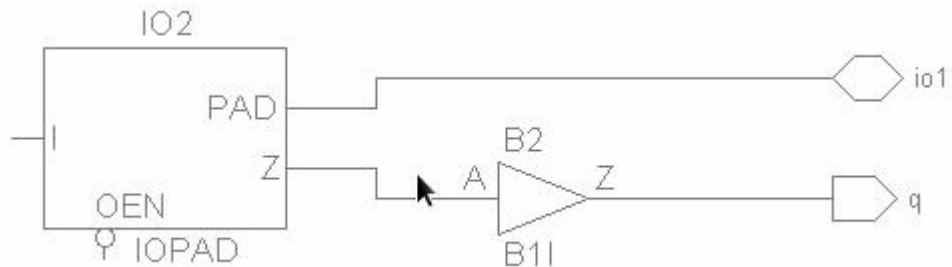


However, if the value at the Select pin ensures that only one value can be propagated at the output of the MUX, the `Clk_Gen23a` rule reports correctly.

■ IO pad is not handled correctly

If the path from the definition point of the generated clock and source contains an IO pad, the polarity computation for the IO pad is not correct due to which the verification status of that generated clock might be incorrect.

For example, in the following schematic, suppose the generated clock is defined on the port `q` and the source is `io1`. The `Clk_Gen23a` rule assumes an inversion between PAD and Z pins of IO2. Therefore, the rule expects an inverted generated clock at port `q`.



■ Multiple generated clocks on a port/pin with the same name, but different period

If there exist multiple generated clocks on a port/pin with the same name, but with a different period, the `Clk_Gen23a` rule selects any of the generated clocks.

Suppose, the generated clocks G1 with `divide_by 1` and `divide_by 2` are defined on a port `q`, this rule can select either one for validation because the rule is not able to differentiate the clocks based on period.

Parameter(s)

- `tc_solver_run_time`: Default is 2h. This signifies the default runtime is 2 hours. Set the value to any positive integer with s for second, h for hours, and m for minutes.

Constraint(s)

SDC

- [create_generated_clock](#) (Mandatory): Use to create a clock.

Messages and Suggested Fix

The following message appears when the CSV report is generated for the verification of generated clocks:

[Cl k_Gen23a_01][INFO] Generated Clock Waveform verification w.r.t. master clock is reported for block <block-name>

Or

[Cl k_Gen23a_02][WARNING] Generated Clock Waveform verification w.r.t. master clock is reported for block <block-name>

Or

[Cl k_Gen23a_03][ERROR] Generated Clock Waveform verification w.r.t. master clock is reported for block <block-name>

Potential Issues

The severity of the violation message is dependent on the definition of the generated clock:

- **INFO**: This severity appears if all the generated clocks are correct.
- **WARNING**: This severity appears if all the generated clocks are correct and some generated clocks are not defined.
- **ERROR**: This severity appears if any of the generated clock is incorrect.

Refer to the [How to Debug and Fix](#) section for details about the potential issues.

Consequences of Not Fixing

For an Error or Warning message, the consequences of not fixing are:

- **Error**: Tools assume a greater slack than available when the waveform is derived incorrectly from a source clock. Even though the timing analysis may run successfully, the chip might eventually fail timing.
- **Warning**: Incompletely specified generated clocks lead to incomplete timing analysis. For the given constraints, the timing analysis would be complete, but there would be paths which are not timed for the missing generated clocks. Therefore, the chip might eventually fail timing.

How to Debug and Fix

To open the CSV report, double-click the violation message. The CSV report provides the generated clock, the corresponding Master clock, the source and destination points as per the design. The status of the validation of each generated clock is specified in the Status column.

The status of the CSV report can be:

- **FAILED:** This status appears when the generated clock specified is incorrectly defined.
- **MISSING:** This status appears because there are multiple paths existing in the clock path as stated in the message. For one of these paths, you have defined the generated clock. However, for the other paths, the rule is expecting another generated clock. Schematic shows the path for which generated clock is not defined.
- **PASSED:** This status appears when all of the generated clocks are correct. View the incremental schematic to validate the results. Clk_Gen23a reports PASSED even if there is a shift between the specified generated clock waveform and the actual generated clock waveform. In such cases, the actual waveform is recommended to the user and the actual waveform is reported in the 'Reason' field.
- **NO_PATH:** This status appears when no path exists between the source point and the definition point of the generated clock.
- **PARTIAL:** This status appears when the verification of the generated clock could not be completed because of the time limit.
- **TIME_OUT:** This status appears when the verification of the generated clock could not be started because of the time limit.
- **SKIPPED:** This status appears when the generated clock definition has error(s) or the clock period is zero.

To view the incremental schematic, select the row in the CSV report, and then click the **Incremental Schematic** icon.

In addition, click the **Waveform Viewer** icon to view the waveform of the master and generated clock. Through the Waveform Viewer, you can validate the waveform of the clock at the point where the generated clock is defined. The edges of the actual and the user generated clocks must be aligned. In the following waveform viewer image, the actual and user generated clocks are not aligned and is reported as FAILED.

To resolve the FAILED or MISSING generated clocks, update the SDC or

Design file.

Example Code and/or Schematic

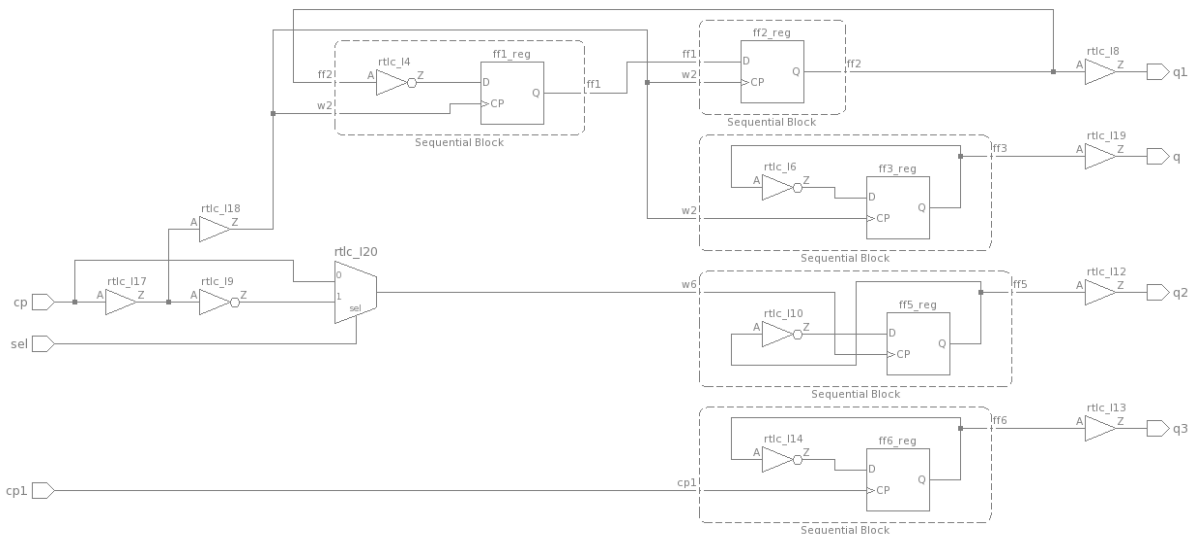
This example illustrates the use of the Clk_Gen23a rule. Suppose, you have the following design and SDC file.

```

1 create_clock -name CLK -period 10 cp
2 create_clock -name CLK1 -period 20 cp1
3
4 create_generated_clock -name GCLK1 -source cp -divide_by 2 q
5 create_generated_clock -name GCLK2 -add -source cp -master CLK -divide_by 2 q1
6 create_generated_clock -name GCLK3 -add -source cp -master CLK -divide_by 2 q2
7 create_generated_clock -name GCLK4 -add -source cp -master CLK1 -divide_by 4 q3

```

The schematic of this design is as follows:

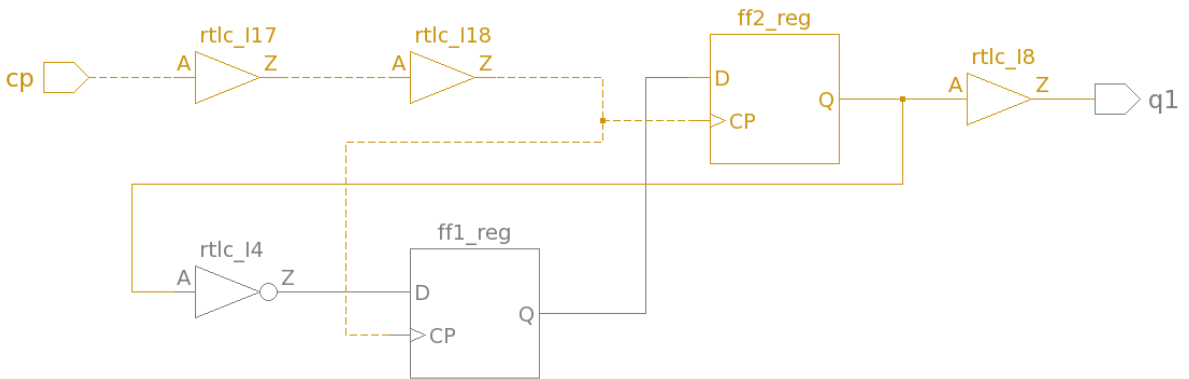


After running the Clk_Gen23a rule, the following CSV report is generated:

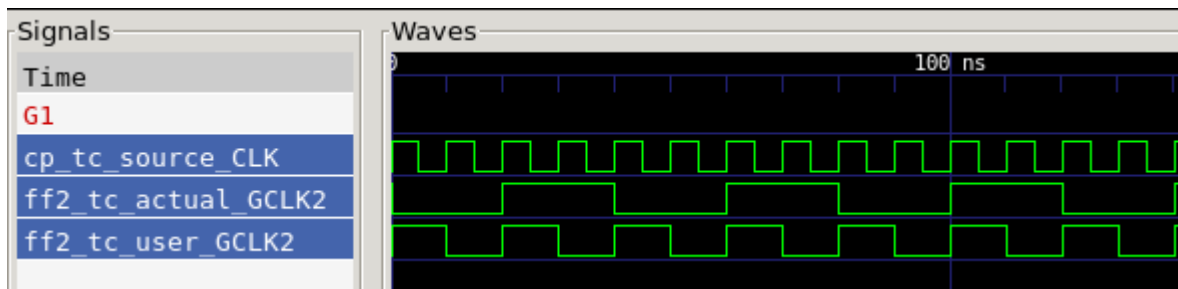
Clock Rules

	B	C	D	E	F	G	H	I
	File Name	Line No	Generated Clock	Master Clock	Destination Point	Source Point	Status	Reason
1	top1.sdc	7	GCLK4				SKIPPED	Inactive create_generated_clock master clock is not present in it's fan-in
2	top1.sdc	4	GCLK1	CLK	q	cp	↕ PASSED	The recommended waveform for this generated clock is 3-5-7
3	top1.sdc	5	GCLK2	CLK	q1	cp	↕ FAILED	waveform at definition point is not matching with the specified clock
4	top1.sdc	6	GCLK3	CLK	q2	cp	↕ PASSED	The recommended waveform for this generated clock is 3-5-7
5	top1.sdc	6	GCLK3	CLK	q2	cp	↕ MISSING	no create_generated_clock is specified for the reported waveform. Please correlate the generated waveform with the design before defining the generated clock. User clock GCLK3 is used for comparison purpose in the waveform viewer

The incremental schematic for the GCLK2 is as shown:



The waveform is as follows:



Default Severity Label

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen24

Identifies clocks for which generated clock is not defined at a design object

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen24* rule reports violations for clocks applied at a design object when the clocks do not have a corresponding create generated clock. Clocks with the `add` argument are supported. This rule considers case analysis settings and therefore case analysis is simulated in this check.

create_generated_clock, with multiple clocks at source, specified without the `master_clock` argument will associate it with the first clock present at source, as defined in the SDC file.

Generated clock should be defined with respect to each clock reaching the generated clock's object.

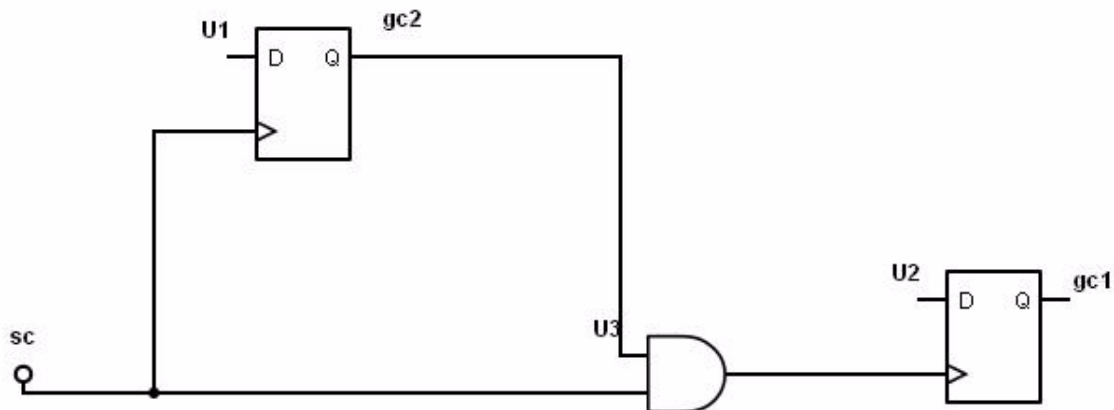
Prerequisites

It is recommended to run the *Clk_Gen24* rule after running the *Clk_Gen01*, *Clk_Gen02*, and *Clk_Gen26* rules because the violations of these rules must be fixed before resolving the violations of this rule.

Rule Exceptions

The *Clk_Gen24* rule ignores cases where the generated clock and the corresponding source are applied at the same design object.

In addition, the *Clk_Gen24* rule ignores generated clocks that have the same source clock as of the generated clock defined on the reported design object. For example, in the following schematic, for `gc1` no violation message is reported because `gc2` has the same source clock `sc` as `gc1`.



Parameter(s)

None

Constraint(s)

SDC

- [create_generated_clock](#) (Mandatory): Use to create a clock.

Messages and Suggested Fix

The following message appears when a generated clock at design object *<obj-name>*, corresponding to the source clock *<clk-name>* is missing in design/block *<block-name>*:

[INFO] Missing create_generated_clock(s) at design object *<obj-name>*, corresponding to source clock(s) *<clk-name>*, in design/block *<block-name>*.

Where, *<obj-name>* is the name of the design object on which [create_generated_clock](#) is missing.

Potential Issues

This violation message appears when:

- A design object, which has a generated clock, has more than one path reaching this design object and all are driven by the same clock, then only one violation will be issued for this clock.
- The generated clock is defined incorrectly, the source clock of the generated clock could not reach the design object where this generated clock is applied. In this case, the Clk_Gen24 rule violates for all such clocks that reach this design object and do not have corresponding generated clock.

Consequences of Not Fixing

Constraining and generating with a wrong clock could potentially allow a greater slack than available. Timing analysis may run successfully, but the chip might eventually fail timing.

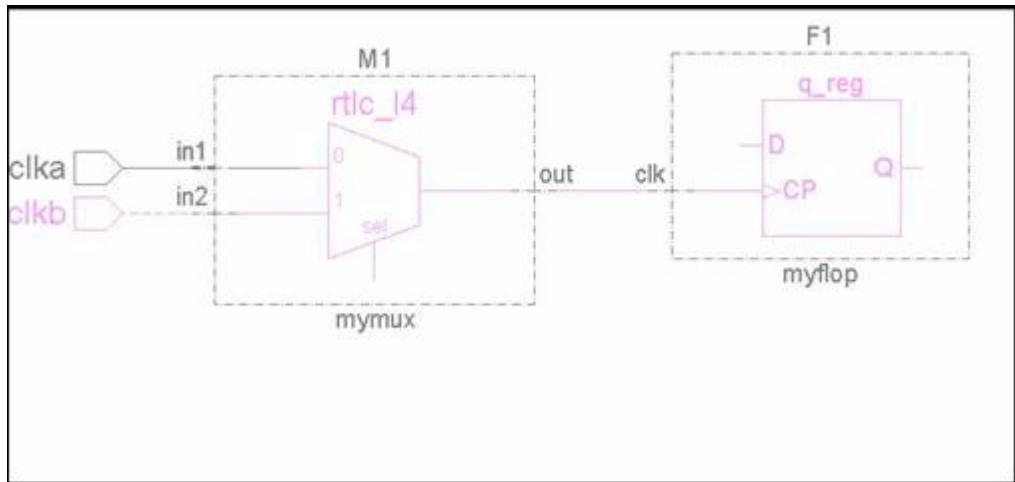
How to Debug and Fix

Check the clocks driving the source point of the generated clock and if generated clocks are defined with respect to all the source clocks.

Example Code and/or Schematic

If a design object, which has a generated clock, has more than one path reaching this design object and all are driven by the same clock, only one violation is reported for this clock. The schematic shows any one of these paths.

Consider the following design.



The constraints are defined as follows:

```
//test.sdc
create_clock -name Clk1 -period 10.000000 -waveform {
0.000000 5.000000 } {clka}
create_clock -name Clk2 -period 10.000000 -waveform {
0.000000 5.000000 } {clkb}
create_generated_clock -name gClk1 -source clka -divide_by 2
F1/q
```

In this example, the generated clock gClk1 is defined with respect to the source clock Clk1 only. The violation is reported because the generated clock is not also defined with respect to Clk2.

Default Severity Label

Info

Rule Group

Clk_Gen

Clock Rules

Reports and Related Files

None

Clk_Gen25

Reports pins/ports that have multiple clocks defined

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen25* rule reports ports/pins that have multiple clocks defined with more than one clock defined using the `add` argument.

Parameter(s)

None

Constraint(s)

SDC

- [create_clock](#) (Optional): Use to create a clock.
- [create_generated_clock](#) (Mandatory): Use to create a clock.

Messages and Suggested Fix

The following message appears:

```
[INFO] Multiple clocks(<clk-name-list>) defined at <type>  
<pp-name> for design <name> with respect to schema specified in  
sgdc (file: <file-name> and line: <num>)
```

Where, *<type>* can be port or pin.

Potential Issues

The violation message appears because:

- Multiple clocks are defined on the same port/pin using the [create_clock](#) constraint and the `add` argument. Refer to **Example 1**.
- Multiple generated clocks are defined on the same port/pin and same source pin using the [create_generated_clock](#) constraint and the `add` argument.

- Multiple generated clocks are defined on the same port/pin using the [create_generated_clock](#) constraint set on different source pins. Refer to **Example 2**.

Consequences of Not Fixing

Typically, multiple clocks on a port exist when the same SDC is meant for multiple modes. This can be an issue with some tools and could lead to longer times or inferences based on single mode only.

How to Debug and Fix

Not applicable

Example Code and/or Schematic

Example 1

In this example, multiple clocks are defined on the same port/pin by using the [create_clock](#) constraint and the `add` argument. Two clocks, Clk2 and Clk3, are defined for the FF0/Q pin. As a result, the *Clk_Gen25* rule reports a violation.

```
create_clock -name Clk1 -period 10
  -waveform {0.0 5.0} [get_ports P1]
create_clock -name Clk2 -period 10
  -waveform {0.0 5.0} [get_pins FF0/Q]
create_clock -name Clk3 -period 5
  -waveform {0.0 2.5} [get_pins FF0/Q] -add
```

Example 2

In this example, multiple generated clock are defined on the same port/pin by specifying the [create_generated_clock](#) constraint on different source pins. Two generated clocks, gClk1 and gClk2, are defined for the FF0/Q pin with different source pins. Therefore, the FF0/Q pin is reported by the *Clk_Gen25* rule.

```
create_generate_clock -name gClk1
  -source [get_pins SP1]
  -divide_by 2 [get_pins FF0/Q]
```

```
create_generate_clock -name gClk2  
-source [get_pins SP2]  
-divide_by 2 [get_pins FF0/Q]
```

Default Severity Label

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen26

Identifies multiple clocks at the source pin of a generated clock that does not have a master_clock argument

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen26* rule reports *create_generated_clock* specifications without the *master_clock* argument when the source pin of the generated clock has either multiple clocks specified or multiple clocks are reaching the source pin.

Rule Exceptions

The *Clk_Gen26* rule does not check whether generated clocks are specified for all master clocks specified on the source pin. It checks that any generated clock specified with respect to the source pin should be with the *master_clock* argument when the source pin of the generated clock has either multiple clocks specified or multiple clocks are reaching the source pin.

Parameter(s)

- *pt*: Default is yes. This indicates that the rule checks if the source port/pin of a generated clock is in the transitive fan-out of a *create_clock* or *create_generated_clock* constraint.

Constraint(s)

SDC

- *create_clock* (Optional): Use to create a clock.
- *create_generated_clock* (Mandatory): Use to create a clock.

Messages and Suggested Fix

The following message appears for a *create_generated_clock* specification for the clock *<clk-name>*:

[INFO] Generated clock "<clk-name>" should be defined with

option -master_clock

Potential Issues

Not applicable

Consequences of Not Fixing

If *create_generated_clock* is specified using every multiple clocks coming into its fan-in, the last definition will overwrite all *create_generated_clock* specifications. It leads to incomplete timing because some clocks will not be taken into account while timing the design.

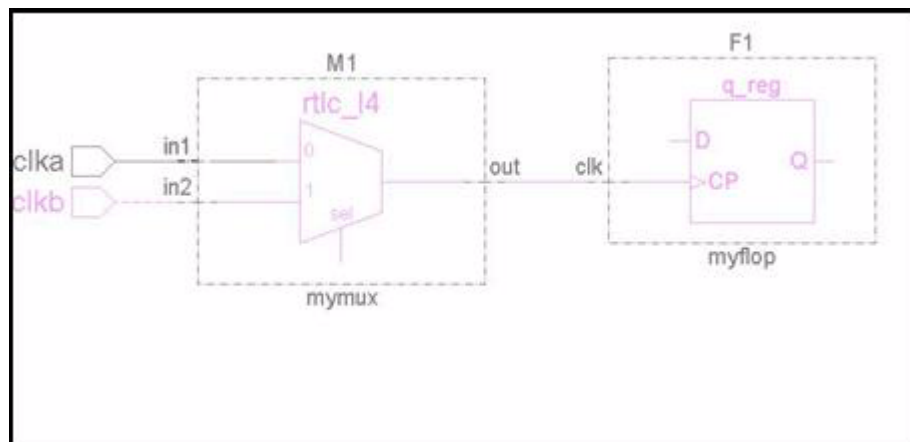
How to Debug and Fix

Not applicable

Example Code and/or Schematic

Example 1

Consider the following design.



In this example, Clk1 and Clk2 are reaching the generated clock gClk1, however the *master_clock* argument is not specified in the definition of clock gClk1. Therefore, the *Clk_Gen26* rule reports a violation.

```
//test.sdc
```

```
create_clock -name Clk1 -period 10.000000 -waveform { 0.000000
5.000000 } {clka}
```

```
create_clock -name Clk2 -period 10.000000 -waveform { 0.000000  
5.000000 } {clka} -add
```

```
create_generated_clock -name gClk1 -source clka -divide_by 2 F1/q
```

Example 2

In this example, source object M1/out is in fan-out of clocks, Clk1 and Clk2. However, the generated clock gClk1 is not defined with the `master_clock` argument. Therefore, the `Clk_Gen26` rule reports a violation.

```
//test.sdc
```

```
create_clock -name Clk1 -period 10.000000 -waveform {  
0.000000 5.000000 } {clka}
```

```
create_clock -name Clk2 -period 10.000000 -waveform {  
0.000000 5.000000 } {clkb}
```

```
create_generated_clock -name gClk1 -source M1/out -divide_by  
2 F1/q
```

Default Severity Label

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen27

Reports generated clocks that are set on non-sequential elements

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen27* rule reports generated clocks created on pins of non-sequential elements.

Rule Exceptions

The *Clk_Gen27* rule ignores generated clocks that are:

- defined on pins of a sequential cell (flip-flop or latch)
- defined using `divide by 1`
- defined on an Integrated Clock Gating (ICG) cell
- defined as `combinational`

Parameter(s)

None

Constraint(s)

SDC

- [create_clock](#) (Optional): Use to create a clock.
- [create_generated_clock](#) (Mandatory): Use to create a clock.

Messages and Suggested Fix

The following message appears for a rule-violating [create_generated_clock](#) specification for clock `<gclk-name>` created on combinational instance pin `<pin-name>`:

```
[INFO] Generated clock "<gclk-name>" is created on object  
"<pin-name>" which is not a sequential element
```

Potential Issues

Not applicable

Consequences of Not Fixing

It is expected that the generated clock be defined at the output of the sequential cell.

How to Debug and Fix

To view the generated clocks created on pins of non-sequential elements in the Schematic Window, double-click the violation message. The schematic highlights the object at which the generated clock is applied.

Example Code and/or Schematic

For the following snippet, A1 is not a sequential element. Therefore, the Clk_Gen27 rule reports a message for the generated clock GCLK.

```
create_clock -name CLK -period 10 [get_ports clk]
create_generated_clock -name GCLK -divide_by 2
    -source clk [get_pins A1/in1]
```

Default Severity Label

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen29

Identifies instances when the same clock is converging through different combinational paths

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen29* rule identifies clocks that converge through different combinational paths and reach the clock pin of the sequential cell. In addition, you can set the *tc_violate_ports* parameter to *yes* to enable this rule to report converging paths that reach an output port.

You can break the convergence of the path by applying *set_clock_sense/set_sense* at any path that is converging. This rule reports a violation message when *set_clock_sense/set_sense* is applied at the output pin of a XOR/XNOR. Refer to the *Example Code and/or Schematic* for details. In addition, this rule honors *set_case_analysis* applied on an input pin of a XOR/XNOR.

Rule Exceptions

The Clk_Gen29 rule does not report a violation message violate if any clock type, *create_clock* or *create_generated_clock*, is applied at a converging point.

Parameter(s)

- *tc_violate_ports*: Default is *no*. Set this parameter to *yes* to report converging paths that are reaching an output port.

Constraint(s)

SDC

- *create_clock* (Optional): Use to create a clock.
- *create_generated_clock* (Optional): Use to create a clock.
- *set_clock_sense/set_sense* (Optional): Use to specify the clock propagation conditions.
- *set_case_analysis* (Optional): Use this constraint to specify the list of ports or pins on which the specified constant logic value or transition is assigned.

Messages and Suggested Fix

Message 1

The following message appears for clock `<clk-name>` that converges through different combinational paths:

```
[Clk_Gen29_01][WARNING] Same clock <clk-name> defined on port/
pin "<pp-name>", converges through different combinational
paths
```

Message 2

The following message appears for clock `<clk-name>` that converges through different combinational paths and `set_clock_sense` is applied at converging points:

```
[Clk_Gen29_02][WARNING] Same clock <clk-name> defined on port/
pin "<pp-name>", converges through different combinational
paths (applied set_clock_sense at converging point)
```

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

Convergence in the clock path is not expected. This could lead to different characteristics at the converging point. The clock itself would not be accurately modeled leading to incorrect timing. In most such cases, all but one converging path should have been controlled so that only one active path is reaching to the converging point.

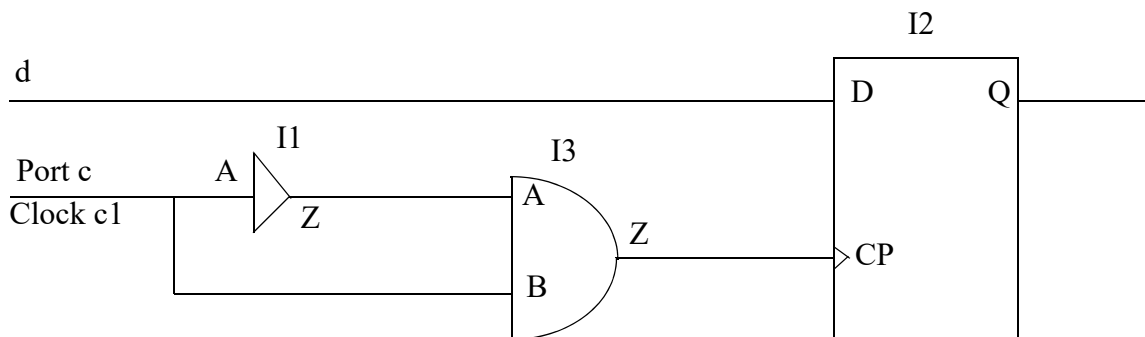
How to Debug and Fix

To fix these violation, break the convergence in the clock path by either applying the `set_clock_sense` constraint or remove the convergence.

Example Code and/or Schematic

Example 1: set_clock_sense not applied on a converging point

In the following figure, assume `create_clock c1` is applied at Port c.



Clock c1 is converging at Pin I3/Z and `set_clock_sense` is not applied at the converging point (I3/Z). This rule reports the following violation:

[WARNING] Same clock c1 defined on port/pin "c", converges through different combinational paths

Example 2: `set_clock_sense` applied on a converging point

In the previous figure, if `set_clock_sense` is applied at the converging point (I3/Z), as shown below:

```
set_clock_sense -pulse rise_triggered_high_pulse [get_pins
I3/Z]
```

The `Clk_Gen29` rule reports the following violation:

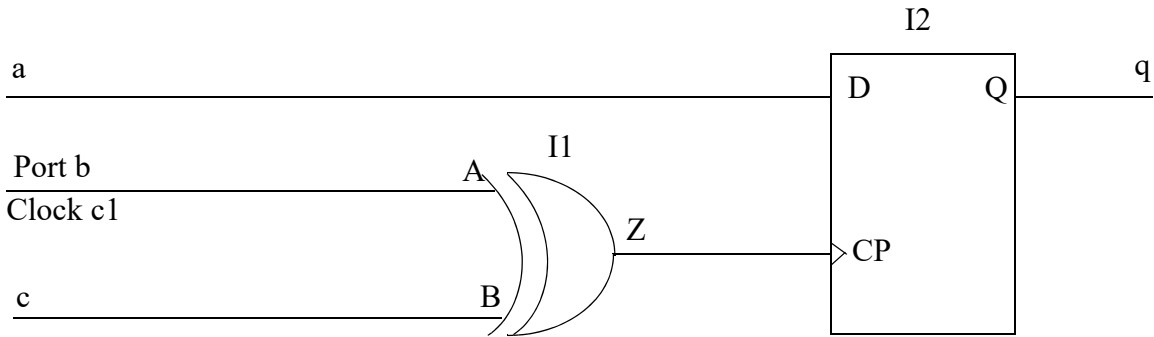
[WARNING] Same clock c1 defined on Port/Pin c, converges through different combinational paths (applied `set_clock_sense` at converging point)

NOTE: NOTE: The `Clk_Gen29` rule checks `set_clock_sense` only if you have specified `set_clock_sense` with the `-pulse rise_triggered_high_pulse` argument.

Example 3: `set_clock_sense` applied at the XOR/XNOR's output pin

In the following figure, assume `create_clock` c1 is applied at Port b and `set_clock_sense` is applied at pin I1/Z.

Clock Rules



Since an XOR/XNOR gate has a converging behavior, the *Clk_Gen29* rule reports the following violation:

[WARNING] Same clock c1 defined on port/pin b, converges through different combinational paths (applied set_clock_sense at converging point)

If you do not apply [set_clock_sense](#) at pin I1/Z, this rule reports the following violation:

[WARNING] Same clock c1 defined on port/pin "b", converges through different combinational paths

Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen30

Identifies generated clocks with an odd Divide_by factor

When to Use

This rule can be run in RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen30* rule reports a violation if the `divide_by` factor of the generated clock is an odd number greater than 1.

Parameter(s)

None

Constraint(s)

SDC

- [create_generated_clock](#) (Mandatory): Use to create a clock.

Messages and Suggested Fix

The following message appears for clock `<clk-name>` with an odd `divide_by` factor `<factor>` greater than 1:

[WARNING] Generated clock "`<clk-name>`" defined with an odd divide by factor (`<factor>`), for design/block "`<des>`"

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Not applicable

How to Debug and Fix

The rule highlights the master clock and generated clock. The violation message also states the odd factor.

To fix this violation message, change the odd value to an even value.

Example Code and/or Schematic

For the following snippet, the *Clk_Gen30* rule reports a violation because the `divide_by` value is 3 in the definition of the generated clock `gc1`.

```
//test.sdc
```

```
create_clock -name c1 -period 20 in
```

```
create_generated_clock -name gc1 -source in -divide_by 3 out
```

To fix this violation, change the value to an even number, such as 2 or 4.

Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen31

Identifies the control pin of a mux in the clock path

When to Use

This is a methodology rule and can be run in all phases of design.

Description

The *Clk_Gen31* rule reports a violation if a clock is detected at the control pin of a multiplexer.

Parameter(s)

None

Constraint(s)

SDC

- *create_clock* (Mandatory): Use to create a clock.
- *create_generated_clock* (Mandatory): Use to create a clock.

Messages and Suggested Fix

The following message appears for clock *<clk-name>* which is detected at the select pin of a mux located in design/block *<des>*:

[INFO] Clock *<clk-name>* is either created or reaching the select pin of the mux for design/block *<des>*

Potential Issues

Resolve this violation message if you are following a design methodology that restricts clocks reaching the select pin of a multiplexer.

Consequences of Not Fixing

Some design methodologies have restrictions on clocks reaching the select pin of a multiplexer. Typically, select pins of multiplexers use control signals to switch between modes.

How to Debug and Fix

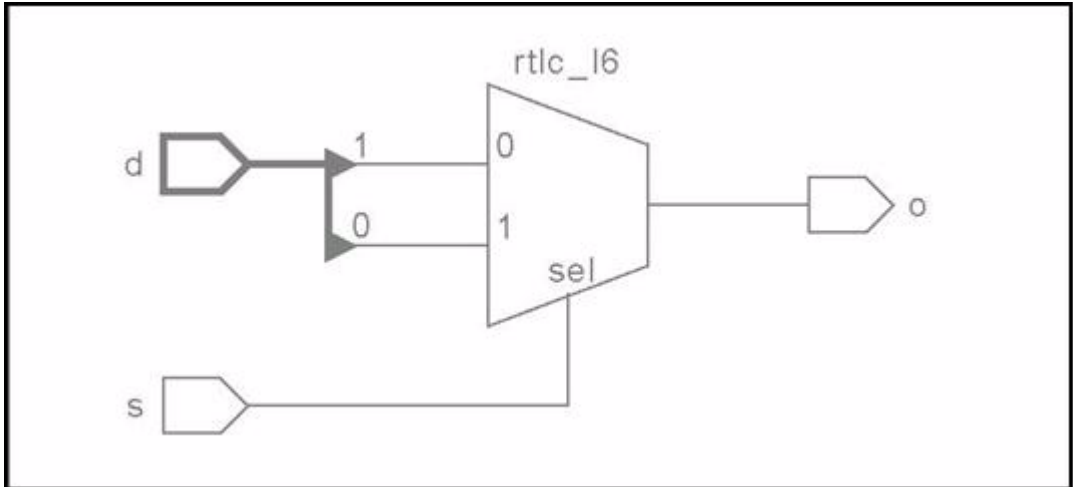
Double-click the violation message to view the clock in the SDC file.

To fix this violation message, either remove the clock or modify the design

by removing the mux.

Example Code and/or Schematic

Consider the following design.



For the given design, the clock applied on the s port is:

```
create_clock -name c1 -period 20 s
```

The *Clk_Gen31* rule reports a violation because the clock c1 is reaching the Select pin of the mux.

Default Severity Label

Info

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen32

Ensures a single path for the master clock to its generated clock

When to Use

This rule is applicable to all phases of design.

Description

The *Clk_Gen32* rule reports a violation when the master clock of a generated clock is detected at the generated clock's design object through multiple paths.

The *Clk_Gen32* rule checks if the *create_generated_clock* satisfies all of the following conditions:

- There is convergence in the clock path.
- The *create_generated_clock* constraint is defined after reconvergence.
- Master of the *create_generated_clock* is defined before the branch of the clock path.

Parameter(s)

None

Constraint(s)

SDC

- *create_generated_clock* (Mandatory): Use to create a clock.

Messages and Suggested Fix

The following message appears for clock *<clk-name>*, which is being reached by the master clock *<master-clk-name>* for design/block *<des>*:

[WARNING] Generated clock *<clk-name>* is being reached by master clock *<master-clk-name>* through multiple paths for design/block *<des>*

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

This rule detects and avoids *create_generated_clock*, which can have multiple clock paths, so that the Static Timing Analysis (STA) tool does not pick up the wrong clock path.

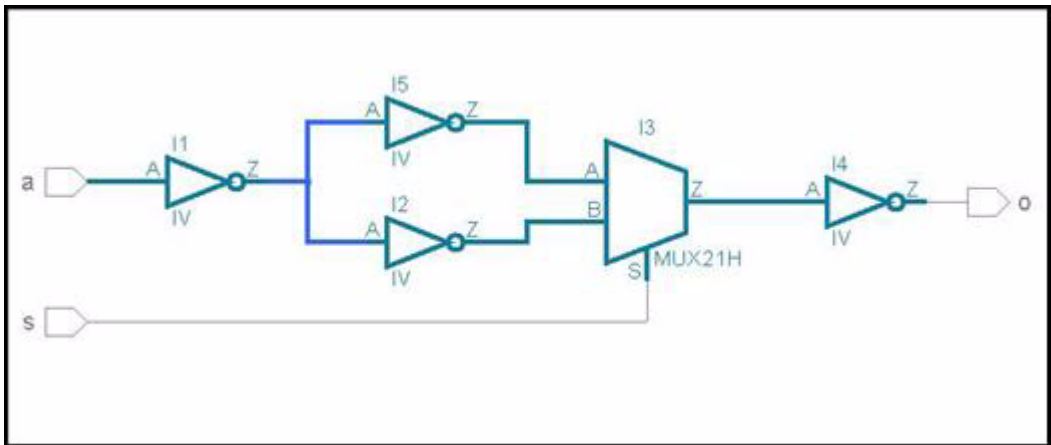
How to Debug and Fix

Double-click the violation message to view the schematic. The schematic highlights the multiple paths and the related clock.

To resolve this violation, leave only one path as active by applying the *set_clock_sense* constraint in other paths.

Example Code and/or Schematic

Consider the following design.



For this design, the generated clock is defined in the SDC file as:

```
create_clock -name c1 -period 20 a
```

```
create_generated_clock -name gc1 -source a -divide_by 1 I4/Z
```

The *Clk_Gen32* rule identifies two paths for source clock c1 to generated clock gc1. This rule does not report a violation if *set_clock_sense* is applied at either I2/Z or I5/Z with respect to clock c1.

Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen33

Reports unconstrained clock pins

When to Use

Use this rule to detect unconstrained sequential elements and when timing coverage is less than 100% for clocks. This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen33* rule finds and reports situations where the clock pin of a sequential element is not constrained. It reports clock pins that are driven by mux having unconstrained data pins. This rule lists all the unconstrained pins of flip-flop in a CSV format report named <top_module>_2_Gen33_uncons.csv.

Parameter(s)

The following table mentions how specific rule parameter settings change the *Clk_Gen33* rule behavior.

Parameter	Parameter Value	Rule Behavior
<i>tc_show_all_unconstrained_flops</i>	No (Default)	Reports only one unconstrained clock pin affected due to the same reason. NOTE: If the clock pin is being driven by a constant value, <i>tc_show_all_unconstrained_flops</i> is ignored. Therefore, all clock pins are reported, regardless of the rule parameter value.
	Yes	Reports all the unconstrained clock pins affected due to the same reason.
<i>tc_ignore_latch_enable</i>	No (Default)	Considers all control/enable pins of the latches that are unconstrained.
	Yes	Ignores all control/enable pins of the latches that are unconstrained.

Parameter	Parameter Value	Rule Behavior
<i>strict</i>	No (Default)	The <i>Clk_Gen33</i> rule traverses through Mux select pins to find the probable clock port.
	Yes	The <i>Clk_Gen33</i> rule does not traverse through Mux select pins to find the probable clock port.

Constraint(s)

SDC

- *create_generated_clock* (Mandatory): Use to create a clock.
- *set_clock_sense/set_sense* (Optional): Use to specify the clock propagation conditions.

Messages and Suggested Fix

Message 1

The following message appears when CSV report *<report-name>* is generated for clock pins of sequential elements that are driven by hanging net/terminal in design *<des>*:

[WARNING] Clock pins of sequential elements are driven by hanging net/terminal in design *<des>*. CSV report *<report-name>* generated

Potential Issues

The situations when the clock pin might not receive a clock are:

- Clock pin is hanging.
- Clock pin is driven by a constant value applied on the clock pin itself or a constant value is being propagated to the clock pin from fan-in (due to *set_case_analysis* or VSS/VDD)
- *set_disable_timing* or *set_clock_sense/set_sense* is set, or a constant value at other inputs of the sequential cell
- Clock pin is unconstrained because no clock is defined in the fan-in (port, black box, output pin of sequential cell) of the clock pin in the sequential element

Consequences of Not Fixing

The output of flip-flop may be producing garbage for the design in the fan-out. This will cause the design to malfunction.

How to Debug and Fix

To fix this violation, either remove the flip-flop from the design or net in the fan-in or connect the clock pin to the desired pin.

Message 2

The following message appears when CSV report *<report-name>* is generated for clock pins of sequential elements that are driven by a constant value in design *<des>*:

[WARNING] Clock pins of sequential elements are driven by constant value in design *<des>*. CSV report *<report-name>* generated

Potential Issues

The clock pin is driven by a constant value applied on the clock pin or a constant value is being propagated to the clock pin from a fan-in, due to [set_case_analysis](#) or VSS/VDD.

Consequences of Not Fixing

There is no toggling at flip-flop clock pin, which makes the flip-flop redundant.

How to Debug and Fix

Remove the constraint that is causing the clock pin to reach a constant value.

Message 3

The following message appears when CSV report *<report-name>* is generated for clock pins of sequential elements that are getting blocked in design *<des>*:

[WARNING] Clock pins of sequential elements are getting blocked in design *<des>*. CSV report *<report-name>* generated

Potential Issues

[set_disable_timing](#) or [set_clock_sense/set_sense](#) is set, or a constant value at other inputs of the sequential cell.

Consequences of Not Fixing

No clock is reaching the clock pin of the flip-flop. Therefore, either the flip-flop is redundant or the clocks are not defined appropriately. This leads to incomplete timing analysis.

How to Debug and Fix

Double-click the violation to view the related path from the blocked pin to the clock pin are highlighted.

To fix this violation, remove the constraint so that the pin is not blocked.

Message 4

The following message appears when CSV report *<report-name>* is generated for clock pins of sequential elements that are driven by the output of a sequential element in design *<des>*:

[WARNING] Clock pins of sequential elements are driven by the output of a sequential element (whose output will not toggle due to constant value at its data or clock pin) in design *<des>*. CSV report *<report-name>* generated

Potential Issues

This violation message appears when the clock pin of a flip-flop is driven by a non-toggling flip-flop (D or CP pin is constant).

Consequences of Not Fixing

No clock is reaching the clock pin of the flip-flop. Therefore, either the flip-flop is redundant or the clocks are not defined appropriately. This leads to incomplete timing analysis.

How to Debug and Fix

To fix this violation, ensure the output pin of the driving flip-flop is not generating a permanent constant output value.

Message 5

The following message appears when CSV report *<report-name>* is generated for clock pins of sequential elements that are not driven by a clock in design *<des>*:

[WARNING] Clock pins of sequential elements are not driven by a clock in design *<des>*. CSV report *<report-name>* generated

Potential Issues

This violation message appears because the clock is missing in the fan-in of

the clock pin.

Consequences of Not Fixing

Sequential elements will be redundant because no toggling occurs.

How to Debug and Fix

To fix this violation, apply the clock in the fan-in of the clock pin.

Message 6

The following message appears when CSV report `<report-name>` is generated for all the unconstrained clock pins in design `<des>`:

```
[WARNING] CSV report <report-name> generated for all the  
unconstrained clock pins in the design <des>
```

Potential Issues

The violation messages explicitly states the potential issues.

Consequences of Not Fixing

All clocks must be defined such that each and every sequential element in the design gets a clock. Otherwise, the timing will be incomplete.

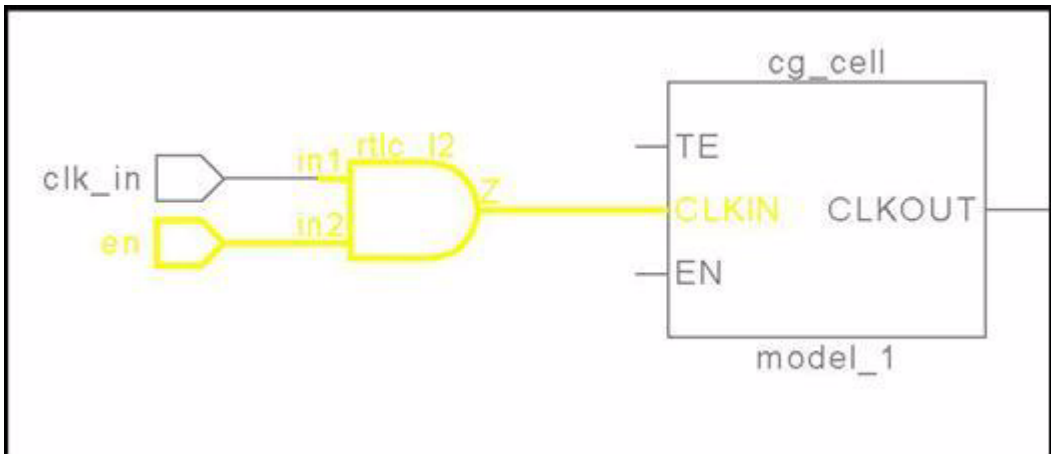
How to Debug and Fix

If the clock pin of a sequential element is being driven by the output of a Mux and all the data pins of the Mux are not clocked, the report generated includes all such clock pins. However, the report for all the unconstrained clock pins does not include these clock pins.

To fix this violation, ensure all clocks are defined such that each and every sequential element in the design gets a clock.

Example Code and/or Schematic

Consider the following design.



The following constraints are defined.

```
//test.sdc
set_case_analysis 0 en
```

The *Clk_Gen33* rule reports a violation because a constant is reaching the clock pin of the flip-flop.

Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

The reports generated by the *Clk_Gen33* rule are:

- [<top_module>_2_Gen33_constant.csv](#)
- [<top_module>_2_Gen33_blocked.csv](#)
- [<top_module>_2_Gen33_hanging.csv](#)
- [<top_module>_2_Gen33_noclock.csv](#)
- [<top_module>_2_Gen33_nontoggle.csv](#)

■ *<top_module>_2_Gen33_uncons.csv*

Examples of each report are as follows.

<top_module>_2_Gen33_constant.csv

This file contains all the unconstrained clock-pins of the sequential cells

which are getting constant value due to :

- 0/1 applied at pin itself.

- set_case_analysis applied on it or in the fanin of this pin.

- connected to VSS/VDD in the fanin.

#CrossProbeTag:GEN33_TAG

##Sheet_Prop:hideCols:1

##Sheet_Prop:cross-probe-data-col:1

,S.No.,Unconstrained Pins,Constant value on pin,Driven by,Constant value applied

12,1,cg_cell/CLKIN,0,en,0

13,2,f1/CP,0,en,0

14,3,f2/CP,0,en,0

15,4,cg_cell2/CLKIN,0,cg_cell2/CLKIN,0

16,5,f3/CP,0,cg_cell2/CLKIN,0

17,6,f4/CP,0,cg_cell2/CLKIN,0

18,7,f5/CP,0,f5/CP,0

19,8,f6/CP,0,f6/CP,0

20,9,f7/CP,1,f7/CP,1

21,10,f8/CP,1,f8/CP,1

<top_module>_2_Gen33_blocked.csv

```
# This file contains all the unconstrained clock-pin which
are blocked either due to
# set_disable_timing constraint or due to set_clock_sense
constraint
```

```
#CrossProbeTag:GEN33_TAG
##Sheet_Prop:hideCols:1
##Sheet_Prop:cross-probe-data-col:1
```

```
,S.No.,Unconstrained Pins,Path is blocked due to
```

<top_module>_2_Gen33_hanging.csv

```
# This file contains all the unconstrained clock-pin which
are driven by a hanging
# net of terminal
```

```
#CrossProbeTag:GEN33_TAG
##Sheet_Prop:hideCols:1
##Sheet_Prop:cross-probe-data-col:1
```

```
,S.No.,Unconstrained Pins,Driven by hanging net/terminal
```

<top_module>_2_Gen33_noclock.csv

```
# This file contains all the probable points where clock can
be defined to constrain
# the reported unconstrained clock pins.
```



```
#CrossProbeTag:GEN33_TAG
##Sheet_Prop:hideCols:1
##Sheet_Prop:cross-probe-data-col:1
```

,S.No.,Unconstrained Pins,Clock may be applied on,Terminal where generated clock may be applied the master clock should be

<top_module>_2_Gen33_nontoggle.csv

```
# This file contains all the unconstrained clock-pin which
are driven by a sequential
# cell whose output will not toggle because the data-pin or
clock-pin of this cell is
# getting constant value
```

```
#CrossProbeTag:GEN33_TAG
##Sheet_Prop:hideCols:1
##Sheet_Prop:cross-probe-data-col:1
```

,S.No.,Unconstrained Pins,Driven by,Constant value on data/clock pin,Clock applied on driven by terminal

<top_module>_2_Gen33_uncons.csv

```
# This file contains all the unconstrained clock-pins of the
sequential cells.
# This report is only for enlisting all the unconstrained
pins, the reason
# for being unconstrained is dumped in separate csv files
generated by this rule.
```

```
#CrossProbeTag:GEN33_TAG
```

```
##Sheet_Prop:hideCols:1  
##Sheet_Prop:cross-probe-data-col:1  
  
,S.No.,Unconstrained Pins
```

Clk_Gen34

Reports duplicate clock names

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen34* rule reports a violation if duplicate clocks exist in the SDC file. The check is based on the clock names and not the clock characteristics.

Rule Exceptions

This rule does not report a violation if the characteristics of clocks are same and applied on the same port, but names are different.

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

The following message appears for design objects that have duplicate clock names:

[WARNING] Clocks have duplicate name <clk-name> for design <design-name>

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Since the clock names are the same, constraints that refer to this clock name might refer to the wrong clock. This can lead to inaccurate timing results.

How to Debug and Fix

Double-click the violation message. The SDC file appears. The duplicate

clock names are highlighted. The incremental schematic highlights the duplicate clocks.

To resolve the violation message, update the clock names to remove the duplicate clock names.

Example Code and/or Schematic

For the following snippet, the *Clk_Gen34* rule reports a violation message because two clocks have the same name.

```
create_clock -name c1 -period 20 -waveform { 0 10} CLK
create_clock -name c1 -period 20 -waveform { 0 10} CLK -add
```

To resolve the violation, change the name of one of the clocks, as shown in the following snippet:

```
create_clock -name c1 -period 20 -waveform { 0 10} CLK
create_clock -name c2 -period 20 -waveform { 0 10} CLK -add
```

Similarly, the *Clk_Gen34* rule reports a violation for the following snippets:

```
//test1.sdc - The rule does not report a violation.
create_clock -name c1 -period 20 -waveform { 0 10} CLK
create_clock -name c2 -period 20 -waveform { 0 10} CLK
```

```
//test2.sdc - The rule reports a violation.
create_clock -name c1 -period 20 -waveform { 0 10} CLK
create_clock -name c1 -period 20 -waveform { 0 10} CLK
```

```
//test.sdc - The rule reports a violation.
create_generated_clock -name gc1 -source clk -divide_by 2 F1/
Q
create_generated_clock -name gc1 -source clk -divide_by 2 F2/
Q
//test.sdc - The rule reports a violation.
```

Clock Rules

```
create_clock -period 20 -waveform { 0 10} CLK  
create_clock -period 20 -waveform { 0 10} CLK
```

Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen35

Reports clocks not defined on an input port or an output pin of a sequential cell

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

The *Clk_Gen35* rule reports a violation if a *create_clock* or *create_generated_clock* constraint is not defined on an input port or an output pin of a sequential cell.

Rule Exceptions

This rule does not check:

- If *set_case_analysis* is set on the same port/pin where clock is applied.
- For conditions, such as *set_false_path* is defined from the port/pin where the clock is defined.

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

The following message appears when clocks that are not defined on an input port or an output pin of a sequential cell are identified:

[WARNING] Clock object "<object-name>" for clock(s) {<clock-name-list>} for design <design-name> is neither an input port nor an output pin of a sequential cell.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

As a general practice, clocks should be defined on the output of ports or

flip-flop. If not, there might be some clock pins that may not be reachable.

How to Debug and Fix

Double-click the violation message. The SDC file appears. The clock definitions are highlighted.

To resolve the violation message, update clock definitions in the SDC file.

Example Code and/or Schematic

In this example, FF1 is a flip-flop. GCLK1 and GCLK2 are defined on FF3/D, the *Clk_Gen35* rule reports a violation for FF3/D because it is neither an input/inout port nor a sequential cell output pin.

```
create_clock -name CLK1 -period 10 -waveform {0 5} [get_ports
{clk2}]
```

```
create_clock -name CLK2 -period 10 -waveform {0 5} [get_pins
{FF1/CP}]
```

```
create_clock -name CLK3 -period 10 -waveform {0 5} [get_pins
{FF1/CP}] -add
```

```
create_generated_clock -name GCLK1 -source clk2 -divide_by 2
[get_pins {FF3/D}]
```

```
create_generated_clock -name GCLK2 -source clk2 -divide_by 3
[get_pins {FF3/D}] -add -master_clock CLK1
```

The following violation messages are reported.

```
[WARNING] Clock object "FF3/D" for clock(s) { GCLK1, GCLK2 }
for design top is neither an input port nor an output pin of
sequential cell
```

To resolve the violation, redefine the violating clocks on FF3/Q to correct the violating scenario.

Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

None

Clk_Gen36

Reports missing generated clocks at the output of flip-flops

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

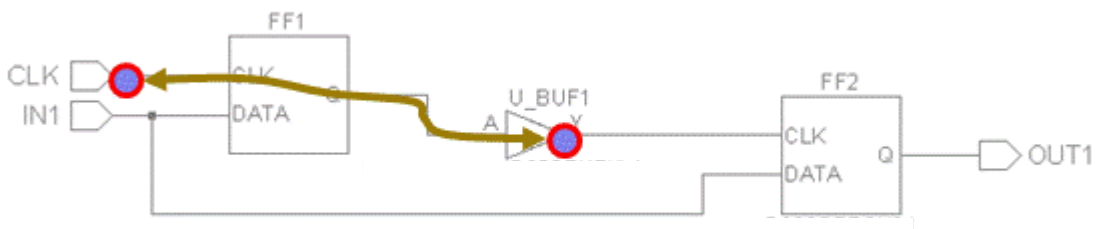
The *Clk_Gen36* rule reports generated clocks that have not been defined at the output of flip-flops between a source clock and the corresponding generated clock.

Rule Exceptions

This rule does not report a violation if there exists a unique path from the source clock to the generated clock.

This rule does not report a violation for flip-flops that do not have a generated clock defined at the output pin if there exists a buffer in the fan-out where a generated clock has been specified.

For example, in the following design, U_BUF1/Y has a generated clock specified with CLK as source. In this case, no generated clock is reported on FF1/Q.



The SDC snippet is as shown:

```
create_clock -period 10 -waveform {0 5} -name CLK [get_ports
CLK]
```

```
create_generated_clock -name GCLK -divide_by 1 [get_pins
U_BUF/Y] -source [get_ports CLK]
```

Parameter(s)

None

Constraint(s)

- `create_clock` (Optional): Use to create a clock.

Messages and Suggested Fix

The following message appears when generated clocks are missing at the output of flip-flops:

[WARNING] For generated clock <gclk-name>, generated clocks are missing at flops <flop-pin-list>

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

If there are unconstrained flip-flop outputs between the generated clock and the corresponding source clock, the source clock waveform propagating to the unconstrained flip-flop outputs might be unknown or incorrect. If the intended clock waveform is specified, the reported generated clock waveform validation shall be correct.

How to Debug and Fix

The generated clock design object, source clock design object and unconstrained flops are referenced in the message.

Double-click the violation message. The SDC file appears. The clock definitions are highlighted.

The incremental schematic shows the generated clock path to the corresponding source clock.

To resolve the violation message, update clock definitions in the SDC file.

Example Code and/or Schematic

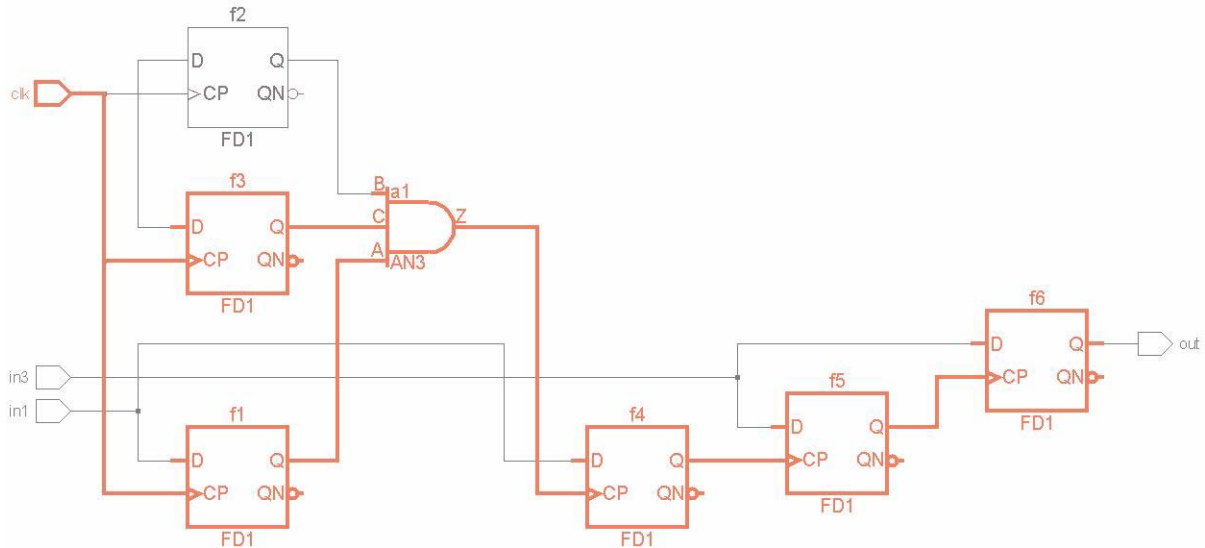
In this example, there are no generated clocks defined for the reported flops. The SDC snippet is shown as follows:

```
create_clock -name Clk1 -period 10.000000 -waveform {  
0.000000 5.000000 } {clk}
```

Clock Rules

```
create_generated_clock -name g6 -source clk -divide_by 2 f6/Q
set_disable_timing a1/B
```

The schematic is as follows:



The *set_disable_timing* constraint is defined at a1/B, therefore a missing generated clock violation message is not reported at f2/Q. However, the following violation is reported for generated clock g6:

[WARNING] For generated clock 'g6', generated clocks are missing at f1 ops 'f1/Q, f3/Q, f4/Q, f5/Q'

To resolve the violation, define generated clocks with an appropriate source at each of the reported flip-flops.

Default Severity Label

Warning

Rule Group

Clk_Gen

Reports and Related Files

None

Clock Latency Rules

The Clock Latency Rules Sub-group `Clk_Lat` contains the following rules:

Rule	Flags
Check_Timing04	Clocks and their sources when clock source latency has been defined for both the clock and its source design object
Clk_Lat01	Incorrect Clock Latency constraints
Clk_Lat02	Generated clocks with incorrect latency values
Clk_Lat03	Clocks in the same domain that have different sums of the network latency and source latency
Clk_Lat04	Runs Clk_Lat04a and Clk_Lat04b rules
Clk_Lat04a	Create clocks for which the set_clock_latency constraint has not been specified or has been specified with latency value 0
Clk_Lat04b	Reports source latencies that have not been defined or are equal to zero for a generated or real create clock
Clk_Lat05	Clocks for which the set_clock_latency constraint is present in the post-layout analysis phase
Clk_Lat06	Clocks with incompletely defined clock latency options
Clk_Lat07	set_clock_latency constraints with inconsistent option values
Clk_Lat08	set_clock_latency constraints with negative values set
Clk_Lat09	Virtual clocks with for which the set_clock_latency constraint has not been specified or has been specified with latency value 0
Clk_Lat10	set_clock_latency constraints specified with the <code>-early/-late</code> options
Clk_Lat12	clocks (specified using the <code>create_clock</code> constraint) for which source latency is not specified in the prelayout or postlayout analysis phases

Check_Timing04

Reports clock source latency defined for clock and its source port/
pin

When to Use

Use this rule in RTL and Pre-layout phases.

Description

The *Check_Timing04* rule reports clocks and their sources when clock source latency has been defined for both the clock and its source design object.

Parameter(s)

None

Constraint(s)

SDC

- *set_clock_latency* (Mandatory): Use to specify the latency of the clock network.
- *create_clock* (Mandatory): Use to create a clock.

Messages and Suggested Fix

The following message appears for clock *<clk-name>* and its source port/pin *<port-pin-name>* when clock source latency has been defined for both the clock and its source design object:

[WARNING] Clock source latency defined for both clock '*<clk-name>*' and its source '*<port-pin-name>*'

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Some tools infer any of the latencies. This may result to wrong analysis of the design.

How to Debug and Fix

Specify either of the latencies.

Example Code and/or Schematic

For the following snippet, the *Check_Timing04* rule reports a violation because the source latency is defined for the object and for clock.

```
//test.sdc
create_clock -name c1 -period 20 in
set_clock_latency 2.0 -source [get_ports in]
set_clock_latency 1.0 -source [get_clocks c1]
```

Default Severity Label

Warning

Rule Group

Clk_Lat

Reports and Related Files

None

Clk_Lat01

Identifies `clock_latency` constraint that is set on an object which is not a clock

When to Use

This rule is applicable to RTL and Pre-layout phases. At Post- layout the delay numbers can be computed based on the clock-network, therefore the rule is not applicable for Post-layout analysis.

Description

The *Clk_Lat01* rule reports [set_clock_latency](#) constraints that have been incorrectly set. The rule behavior is dictated by the settings of the *pt* and *tc_allow_design_obj* parameters.

Parameter(s)

<i>pt</i>	<i>tc_allow_design_obj</i>	Clk_Lat01 Behavior
yes	yes	Reports the set_clock_latency constraints set on design objects whose transitive fan-out does not lead to the clock pin of sequential elements
no	yes	Reports set_clock_latency constraints set on design objects for which no real clock is specified. You can specify real clocks by using the create_clock or create_generated_clock constraints.
yes	no	Reports set_clock_latency constraints for all design objects

Constraint(s)

SDC

- [set_clock_latency](#) (Mandatory): Use to specify the latency of the clock network.

Messages and Suggested Fix

Message 1

The following message appears for an object *<name>* used as a clock in a *set_clock_latency* constraint but has no corresponding *create_clock* or *create_generated_clock* constraint:

[WARNING] Clock latency set on object *<name>* that has no real clock applied on it

Potential Issues

The clock latency should be specified on a clock.

Consequences of Not Fixing

There is some latency in the clock network. At RTL and Pre-layout phases, this is assumed to the desired value. If it is not specified, there is a wider difference in timing analysis at RTL and pre-layout stage as opposed to the Post-layout phase, where actual latency is calculated based on the clock network. To avoid timing closure later, this constraint should be applied at RTL and Pre-layout phases.

How to Debug and Fix

The SDC file highlights the object name.

There might be a real clock in the fan-in of the object. Set the *pt* parameter to *yes* to resolve this violation. If the *pt* parameter is set to *no*, the real clock should be declared on the object name. If the *pt* parameter is set to *yes*, the real clock should exist in the fan-in of the object.

Refer to the *Parameter(s)* section for more details on controlling the behavior of the *Clk_Lat01* rule.

Message 2

The following message appears when the *set_clock_latency* constraint is set on a port/pin *<name>* that does not drive the clock pin of a sequential element, and the values of the *tc_allow_design_obj* and *pt* rule parameters is set to *yes*:

[WARNING] Clock latency is set on port/pin *<name>* which does not lead to a clock pin of sequential element

Potential Issues

The *set_clock_latency* constraint is used to specify the latency in the clock

network. If the port or pin is not reaching the clock-pin of the sequential element, the desired effect cannot be achieved.

Consequences of Not Fixing

The latency specified will not have any effect in timing analysis.

How to Debug and Fix

The SDC file highlights the port or pin.

Set the *pt* parameter to *yes* to resolve this violation. If the *pt* parameter is set to *no*, this violation message does not appear. If the *pt* parameter is set to *yes*, the fan-out of the object should lead to a clock pin of the sequential element.

Refer to the *Parameter(s)* section for more details on controlling the behavior of the *Clk_Lat01* rule.

Validate that the fan-out of the node on which *set_clock_latency* is applied is not leading to the clock pin of a sequential element. Use the **Show fan-out cone** feature of Console GUI.

If the node is not leading to the clock pin of a sequential element, remove the *set_clock_latency* on this object, as it is not of any use.

Message 3

The following message appears if the *set_clock_latency* constraint is set on an invalid design object *<name>*:

[WARNING] Clock latency set on invalid design object <name>
(Design object is invalid because some back end tools do not support clock latency on port/pin)

Potential Issues

The latency can be specified for the clock network only. If the node is neither clock nor leading to the clock pin of a sequential element, the latency specified is not of any use.

Consequences of Not Fixing

The latency specified will not have any effect in timing analysis.

How to Debug and Fix

The SDC file highlights the invalid design object.

Design objects are only valid objects in the *set_clock_latency* constraint, if the *tc_allow_design_obj* parameter is set to *yes*. This violation appears if

the parameter is set to no and the [set_clock_latency](#) constraints contain design objects in the object list.

Refer to the [Parameter\(s\)](#) section for more details on controlling the behavior of the *Clk_Lat01* rule.

To resolve this violation, remove the [set_clock_latency](#) constraint reported in the violation message.

Example Code and/or Schematic

In this example, the *Clk_Lat01* rule reports a violation because the [set_clock_latency](#) constraint is set on an object that has not been declared as a clock or a generated clock.

```
set_clock_latency -rise {B1}
```

To fix the violation, define a [create_clock](#) or [create_generated_clock](#) constraint on B1.

Default Severity Label

Warning

Rule Group

Clk_Lat

Reports and Related Files

None

Clk_Lat02

Source latency for generated clock less than or equal to source clock latency

When to Use

This rule is applicable to RTL and Pre-layout phases. At Post- layout the delay numbers can be computed based on the clock-network, therefore the rule is not applicable for Post-layout analysis.

Description

The *Clk_Lat02* rule flags generated clocks with source latency less than or equal to the source latency of their source clocks. You specify generated clocks by using the *create_generated_clock* constraint.

Parameter(s)

None

Constraint(s)

SDC

- *create_generated_clock* (Mandatory): Use to create a clock

Messages and Suggested Fix

The following message appears for a generated clock *<clk-name>* specified in a *create_generated_clock* constraint that has a source latency less than or equal to source latency of the corresponding source clock *<clks-name>*:

Source latency *<numg>* for generated clock "*<clk-name>*" is less than or equal to the source latency *<num>* of its source clock *<clks-name>*

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

When the source latency for generated clocks is less than or equal to the source clock latency, the clock reaches the destination before the source.

This causes pre-layout timing failure.

How to Debug

The latency constraints are highlighted in the SDC file for the generated clock and its source clocks. You can compare the latency values.

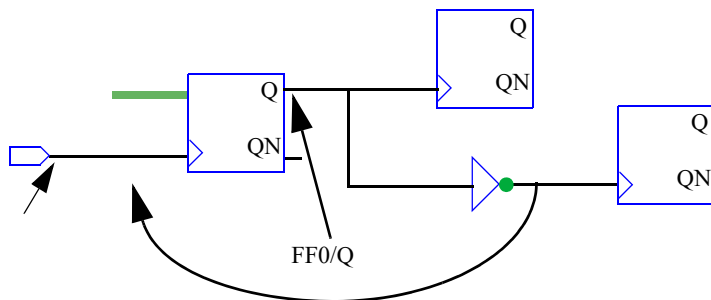
To resolve this violation, modify the latency value of generated clock or the source clock.

Example Code and/or Schematic

Due to the logic between the source of the generated clock and generated clock definition point, the latency value of the generated clock should be greater than the latency value of the source clock.

For example, in the following SDC snippet, the latency value of the source clock is greater than that of the generated clock. Therefore, the *Clk_Lat02* rule reports a violation.

```
set_clock_latency 0.8 [get_clocks CLK]
set_clock_latency 0.7 [get_clocks genclk]
```



Source latency for generated clock < source latency for source clock

Default Severity Label

Warning

Rule Group

Clk_Lat

Reports and Related Files

None

Clk_Lat03

Reports synchronous clocks that have different sum of network and source latency

When to Use

When clocks belong to the same domain, they must have the same sum of network and source latency. Having the same sum of source and network latency within *clock_group* denotes a balanced tree structure.

Use this rule to ensure a balanced tree structure. This rule is applicable to the RTL and Pre-layout phases.

Description

The *Clk_Lat03* rule reports synchronous clocks, specified using the *create_clock* or *create_generated_clock* constraint, that have different sums of the network and source latency.

The *Clk_Lat03* rule reports a violation message for a given clock pair that satisfy all of the following conditions:

- The two clocks are synchronous
- A clock crossing exists between the two clocks
- The sum of the network latency and source latency for the two clocks is different

Prerequisites

The *Clk_Lat03* rule should be run only after specifying the *clock_group* information in the SGDC file. However, if the *clock_group* information is not provided in the SGDC file, this rule obtains the clock group information from the *DomainAnalysis* rule.

Rule Exceptions

This rule does not consider *set_false_path* constraints having -through list.

Parameter(s)

None

Constraint(s)

SDC

- *create_clock* (Mandatory): Use to create a clock.
- *create_generated_clock* (Mandatory): Use to create a clock.
- *set_clock_latency* (Mandatory): Use to specify the expected post-layout delay value. This defines the time taken by the clock from the root to the leaf of the tree.
- *set_clock_groups* (Optional): Use to specify a list of clocks that is used to establish exclusive or asynchronous relationship with clocks of another group.
- *set_false_path* (Optional): Use to specify timing paths that are not considered in timing-analysis. These timing paths are marked as false.
- *set_input_delay* (Optional): Use to define the arrival time relative to a clock.
- *set_output_delay* (Optional): Use to set the output path delay values for the current design.
- *set_clock_uncertainty* (Optional): Use to specify the uncertainty (skew) of clock networks.

Messages and Suggested Fix

The following message appears for clocks *<clk1name>* and *<clk2-name>* of the same clock_group that have different sums (*<sum1>* and *<sum2>*) of network latency (*<n11>* and *<n12>*) and source latency (*<s11>* and *<s12>*):

[WARNING] Sum of network + source latency differ for same clock_group clocks "*<clk1-name>*" (*<s11>* + *<n11>*, sum = *<sum1>*) and clock "*<clk2-name>*" (*<s12>* + *<n12>*, sum = *<sum2>*)

Potential Issues

The *Clk_Lat03* rule reports a violation message for a given clock pair under the following conditions:

- The two clocks are synchronous

Two clocks are considered synchronous, if the clocks are assigned to the same group using the *clock_group* constraint and clocks having different *clock_group* are assumed as asynchronous.

NOTE: If you do not specify any *clock_group* constraint for a clock, a pair of clocks specified in a *set_false_path* or *set_clock_groups* constraint are assumed to be asynchronous and a pair of clocks specified in a *set_clock_uncertainty* constraint are assumed to be synchronous by the *Clk_Lat03* rule.

- A clock crossing exists between the two clocks

Clock crossing exists between two clocks, Clk1 and Clk2, if they belong to any one of the following:

- Clk1 is driving flip-flop F1 and clk2 is driving flip-flop F2. In addition, the output pin of flip-flop F1 is connected to the D pin of flip-flop F2.
- Clk1 is driving a sequential cell F and the *set_input_delay* constraint is set on the D pin of the sequential cell with respect to clk2 (real or virtual)
- Clk1 is driving a sequential cell F and the *set_output_delay* constraint is set on the output pin of the sequential cell with respect to the clock clk2 (real or virtual)

- The sum of the network latency and source latency for the two clocks is different

The *Clk_Lat03* rule checks the sum of network and clock latencies for any combination of (min-max, rise-fall, early-late) options. If latency is not set for any combination, that value is assumed to be zero.

Consequences of Not Fixing

The *Clk_Lat03* rule ensures that all the register clock pins in a domain see the clock signal arriving simultaneously, and therefore minimize skew and achieve the balance clock tree.

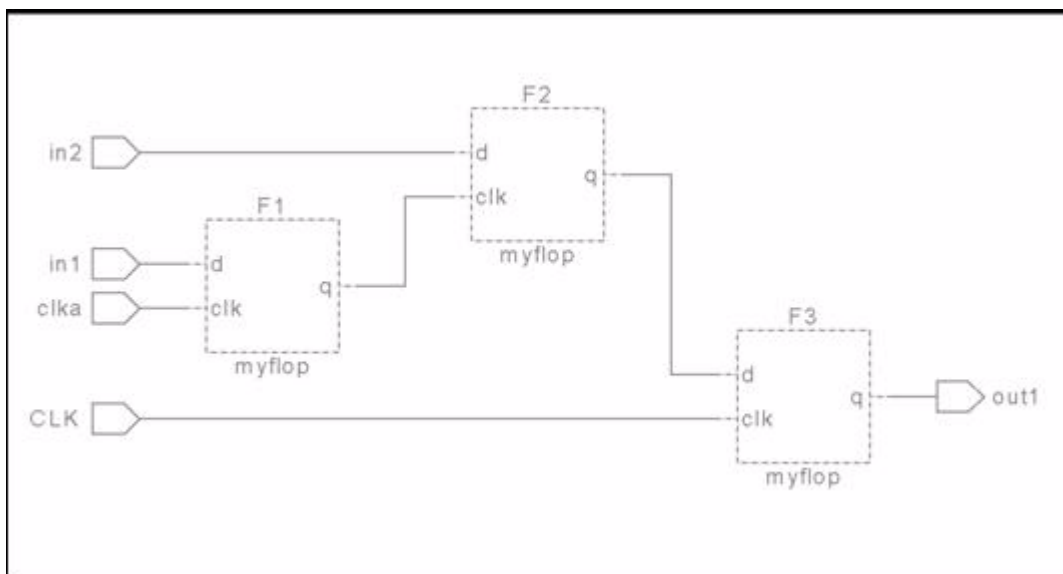
How to Debug and Fix

Review the *set_clock_latency* constraint in the SDC file and ensure that the sum of source latency and network latency match for synchronous clocks that are crossing.

Example Code and/or Schematic

The *Clk_Lat03* rule requires that the sum of the network latency and source clocks latency (if any) for synchronous must be equal.

Consider the following design:



In this example, the *Clk_Lat03* rule reports a violation because the sum of the source latency and network latency of the crossing clocks, Clk2 and GClk1, are not equal.

The clocks and latencies are defined in the SDC file as follows:

```
create_clock -name Clk1 -period 10.000000 -waveform {
0.000000 5.000000 } {CLK}

create_clock -name Clk2 -period 10.000000 -waveform {
0.000000 5.000000 } clka

create_generated_clock -name GClk1 -source clka {F1/q} -
divide_by 2

set_clock_latency 2 {Clk1}
set_clock_latency 9 {GClk1}
set_clock_latency -source 2 {GClk1}
```

The domains in SGDC are defined as:

```
current_design top

sdc_data -type latency.sdc -level rtl -mode func -corner
```

Clock Rules

```
Worst
block -name top
blocksize -min 0 -max 100

domain -name d1 -clock {Clk1 GClk1 Clk2}
```

Default Severity Label

Warning

Rule Group

Clk_Lat

Reports and Related Files

[tc_latency_info Report](#): Contains all clocks, arranged according to clock domain, along with the sum of source latency and network latency for various options (min-max, rise-fall, early-late).

Clk_Lat04

Clock_latency constraint not defined or equal to zero

The Clk_Lat04 rule runs the [Clk_Lat04a](#) and [Clk_Lat04b](#) rules.

Clk_Lat04a

Reports network latencies that have not been defined or are equal to zero for real clocks

When to Use

This rule is applicable to RTL and Pre-layout phases. At Post- layout the delay numbers can be computed based on the clock-network, therefore the rule is not applicable for Post-layout analysis.

Description

The *Clk_Lat04a* rule flags real clocks for which the network latency using the *set_clock_latency* constraint has not been specified or has been specified with latency value 0. You specify real clocks by using the *create_clock* or *create_generated_clock* constraint.

You can specify the network latency using the following constraint:
set_clock_latency <value>

For source latency, specify `-source` with this constraint.

Parameter(s)

None

Constraint(s)

SDC

- *create_clock* (Mandatory): Use to create a clock
- *create_generated_clock* (Mandatory): Use to create a clock
- *set_clock_latency* (Mandatory): Use to specify the expected post-layout delay value. This defines the time taken by the clock from the root to the leaf of the tree.

Messages and Suggested Fix

Message 1

The following message appears for a create clock <*clk-name*> (specified using the *create_clock* constraint) without the corresponding *set_clock_latency* constraint:

[WARNING] Clock network latency for create clock "<clk-name>" has not been set

For information on debugging, click [How to Debug and Fix](#).

Message 2

The following message appears for a create clock <clk-name> (specified using the [create_clock](#) constraint) having a corresponding [set_clock_latency](#) constraint with latency value of 0:

[WARNING] Clock network latency value for create clock "<clk-name>" is zero

For information on debugging, click [How to Debug and Fix](#).

Message 3

The following message appears for a generated clock <clk-name> (specified using the [create_generated_clock](#) constraint) without the corresponding [set_clock_latency](#) constraint:

[WARNING] Clock network latency for generated clock "<clk-name>" has not been set

For information on debugging, click [How to Debug and Fix](#).

Message 4

The following message appears for a generated clock <clk-name> (specified using the [create_generated_clock](#) constraint) having a corresponding [set_clock_latency](#) constraint with latency value of 0:

[WARNING] Clock network latency value for generated clock "<clk-name>" is zero

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

There is some latency in the clock network. At RTL and Pre-layout phases, this is assumed to be the desired value. If it is not specified, there is a wider difference in timing analysis at RTL and pre-layout stage as opposed to the Post-layout phase, where actual latency is calculated based on the clock network. To avoid timing closure later, this constraint should be applied at RTL and Pre-layout phases.

How to Debug and Fix

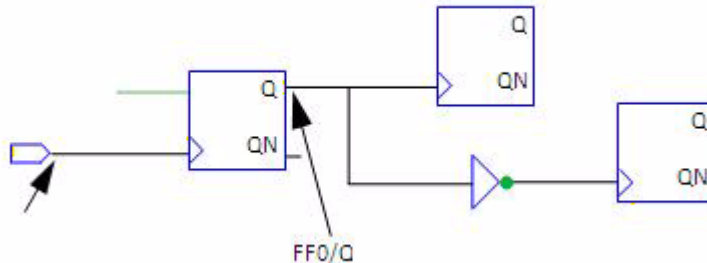
To resolve the violation messages, update the SDC file by:

- **Message 1:** Set the network clock latency for the specified clock as stated in the violation message.
- **Message 2:** Set the network clock latency value for the specified clock to more than zero.
- **Message 3:** Set the network clock latency for the specified clock as stated in the violation message.
- **Message 4:** Set the network clock latency value for the specified clock to more than zero.

Example Code and/or Schematic

In this example, the *Clk_Lat04a* rule reports a violation because no *set_clock_latency* is set for the generated clock. To resolve the violation, insert a *set_clock_latency* constraint on the generated clock.

```
set_clock_latency 0.9 [get_clocks genclk]
```



Default Severity Label

Warning

Rule Group

Clk_Lat

Reports and Related Files

None

Clk_Lat04b

Reports source latencies that have not been defined or are equal to zero for a generated or real create clock

When to Use

This rule is applicable to RTL and Pre-layout phases. At post-layout the delay numbers can be computed based on the clock-network, therefore the rule is not applicable for Post-layout analysis.

Description

The *Clk_Lat04a* rule reports generated or real create clocks for which the source latency using the *set_clock_latency* constraint has not been specified or has been specified with latency value 0. You specify real clocks by using the *create_generated_clock* or *create_clock* constraint.

To specify the source latency, use the `-source` option with the *set_clock_latency* constraint.

Parameter(s)

None

Constraint(s)

SDC

- *create_generated_clock* (Mandatory): Use to create a clock.
- *create_clock* (Mandatory): Use to create a clock.
- *set_clock_latency* (Mandatory): Use to specify the expected post-layout delay value. This defines the time taken by the clock from the root to the leaf of the tree.

Messages and Suggested Fix

Message 1

The following message appears for a generated or real *create_clock* clock `<clk-name>` without the corresponding *set_clock_latency* constraint specifying source latency:

[WARNING] Clock source latency for <clock-type> clock "<clk-

name>" has not been set

Where, <clock-type> is either *generated* or *clock*.

For information on debugging, click [How to Debug and Fix](#).

Message 2

The following message appears for a generated clock <clk-name> (specified using the [create_generated_clock](#) constraint) having a corresponding [set_clock_latency](#) constraint with source latency value of 0:

[WARNING] Clock source latency value for generated clock "<clk-name>" is zero

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

There is some latency in the clock network. At RTL and Pre-layout phases, this is assumed to be the desired value. If it is not specified, there is a wider difference in timing analysis at RTL and pre-layout stage as opposed to the Post-layout phase, where actual latency is calculated based on the clock network. To avoid timing closure later, this constraint should be applied at RTL and Pre-layout phases.

How to Debug and Fix

To resolve the violation messages, update the SDC file by:

- **Message 1:** Set the source clock latency for the specified clock as stated in the violation message.
- **Message 2:** Set the source clock latency value for the specified clock to more than zero.

Example Code and/or Schematic

In this example, the *Clk_Lat04b* rule reports a violation because [set_clock_latency](#) is defined on the generated clock is set to 0.

```
set_clock_latency 0 -rise [get_clocks genclk]
```

To resolve the violation, change the value of the [set_clock_latency](#) constraint to a non-zero positive value.

Default Severity Label

Warning

Rule Group

Clk_Lat

Reports and Related Files

None

Clk_Lat05

Identifies `set_clock_latency` in postlayout analysis

When to Use

The `set_clock_latency` constraints should not appear in Post-layout constraints because the actual path/delay from the clock source pin to flip-flop is known in that phase. Therefore, the delay can be computed from the actual design.

Description

The `Clk_Lat05` rule reports real clocks or clock port/pin for which the `set_clock_latency` constraint is present in the Post-layout analysis phase. You specify clocks by using the `create_clock` or `create_generated_clock` constraint.

Parameter(s)

- `strict`: Default is `no`. This means the `Clk_Lat05` rule does not report primary clocks for which the source latency is specified in the post-layout analysis phase. Set the value to `yes` to report all clocks for which the `set_clock_latency` constraint is specified in the post-layout analysis phase.

Constraint(s)

SDC

- `create_clock` (Mandatory): Use to create a clock
- `create_generated_clock` (Mandatory): Use to create a clock
- `set_clock_latency` (Mandatory): Use to specify the expected post-layout delay value. This defines the time taken by the clock from the root to the leaf of the tree.

Messages and Suggested Fix

Message 1

The following message appears for a clock `<clk-name>` (specified using the `create_clock` or `create_generated_clock` constraint) for which a `set_clock_latency` constraint is present in the postlayout analysis phase:

[WARNING] `set_clock_latency` should not be defined for clock "`<clk-name>`" in postlayout

For information on debugging, click [How to Debug and Fix](#).

Message 2

The following message appears for a clock port/pin `<name>` for which a [set_clock_latency](#) constraint is present in the postlayout analysis phase:

[WARNING] `set_clock_latency` should not be defined for clock port/pin "`<name>`" in postlayout

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

At Post-layout, the clock tree is defined. Therefore, the actual delay in clock network is known. By setting the clock latency at postlayout, the timing engine computes incorrect delay values. As a result, timing may pass, but the chip could fail.

How to Debug and Fix

If the [strict](#) parameter is set to `yes`, then both source/network latencies are reported. Otherwise, only the source clock latency constraints is reported.

To resolve this violation, remove the [set_clock_latency](#) constraint for the clock or the clock port/pin.

Example Code and/or Schematic

In this example, the `Clk_Lat05` rule reports a violation because [set_clock_latency](#) is defined on the clocks during the Post-analysis phase.

```
set_clock_latency 1.2 -rise [get_clocks clk1]
```

To resolve the violation, make sure [set_clock_latency](#) is not defined on the clocks.

Default Severity Label

Warning

Clock Rules

Rule Group

Clk_Lat

Reports and Related Files

None

Clk_Lat06

Reports missing set_clock_latency options

When to Use

Use this rule to check the completeness of the [set_clock_latency](#) constraint. This rule is applicable to the RTL, Pre-layout, Post-layout phases.

Description

The Clk_Lat06 rule reports clocks with incompletely defined clock latency options. This rule checks whether the [set_clock_latency](#) constraints have been explicitly specified with all possible combination of options, such as `-max/-min`, `-early/-late`, and `-rise/-fall`, for each clock.

Parameter(s)

- [tc_ignore_min](#): Default is no. Set the value to yes to ignore the missing `-min` option for worst-case SDC.

Constraint(s)

SDC

- [set_clock_latency](#) (Mandatory): Use to specify the expected post-layout delay value. This defines the time taken by the clock from the root to the leaf of the tree.

Messages and Suggested Fix

The following message appears for a clock `<clk-name>` of design/block `<name>` for which the complete combination set of [set_clock_latency](#) constraint options has not been specified:

```
[WARNING] Clock <type> latency is not specified explicitly with
[<option-list1>] option(s) for clock "<clk-name>" of design/
block "<name>"; option(s) [<option-list2>] have been specified
explicitly
```

Where, `<type>` can be source or network and `<optionlist1>` and `<option-list2>` is any combination of `-max/-min`, `-early/-late`, and `-rise/-fall` options.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

If only one option is specified, the other one remains unspecified, that is zero latency. Therefore, clock delays are not taken into account while performing synthesis/STA.

How to Debug and Fix

The latency constraints are highlighted in the SDC file for the generated clock and its source clocks. You can compare the latency values and insert the missing [set_clock_latency](#) options.

Example Code and/or Schematic

In this example, the source latency is specified with all options. Therefore, *Clk_Lat06* rule does not report a violation.

```
set_clock_latency 1.0 -source -clock c1
```

```
set_clock_latency -rise 1.0 -clock c1
```

However, since the Network latency is specified with only `-rise -max -min` options, this rule reports a violation to provide the missing options `-fall -max, -fall -min`.

Default Severity Label

Warning

Rule Group

Clk_Lat

Reports and Related Files

None

Clk_Lat07

Identifies `set_clock_latency` defined with inconsistent option values

When to Use

This rule is applicable to RTL and Pre-layout phases. At Post-layout the delay numbers can be computed based on the clock-network, therefore the rule is not applicable for Post-layout analysis.

Description

The *Clk_Lat07* rule reports `set_clock_latency` constraints with inconsistent option values, such as:

- A maximum value is less than a minimum value.
- An early value is less than a late value.

Parameter(s)

None

Constraint(s)

SDC

- `set_clock_latency` (Mandatory): Use to specify the expected post-layout delay value. This defines the time taken by the clock from the root to the leaf of the tree.

Messages and Suggested Fix

The following message appears for a clock `<clk-name>` of design/block `<name>` for which the value `<value1>` specified with option `<option1>` is less than value `<value2>` specified with option `<option2>`:

[WARNING] `<option1>` `<type>` latency value `<value1>` is less than `<option2>` latency value `<value2>` for `<clk-type>` clock "`<clk-name>`" of design/block "`<name>`"

Where, `<type>` can be source or network, `<option1>` and `<option2>` are a concatenation of applicable argument names as in following examples:

<Option1>	<option2>
Max	Min
Max_Fall	Min_Fall
Max_Early	Min_Early
Min_Late_Fall	Min_Early_Fall

Potential Issues

The maximum delay value should always be greater than or equal to the minimum value.

Consequences of Not Fixing

The minimum values being more than the maximum values causes the tool to make assumptions. The tool behaves differently from what was specified. The assumptions could be different. from your expectations.

How to Debug and Fix

The SDC file highlights both constraints indicated in the violation message. To resolve this violation, review the latency values and modify them appropriately.

Example Code and/or Schematic

In this example, the *Clk_Lat07* rule reports a violation because the [set_clock_latency](#) values are inconsistent. The *early* latency is more than the *late* latency.

```
set_clock_latency -early 0.8 [get_clocks clk1]
set_clock_latency -late 0.2 [get_clocks clk1]
```

Default Severity Label

Warning

Rule Group

Clk_Lat

Reports and Related Files

None

Clk_Lat08

Identifies `set_clock_latency` constraints that have been set to a negative value

When to Use

This rule is applicable to RTL and Pre-layout phases. At Post-layout the delay numbers can be computed based on the clock-network, therefore the rule is not applicable for Post-layout analysis.

Description

The *Clk_Lat08* rule reports `set_clock_latency` constraints that have negative values set.

Parameter(s)

None

Constraint(s)

SDC

- `set_clock_latency` (Mandatory): Use to specify the expected post-layout delay value. This defines the time taken by the clock from the root to the leaf of the tree.

Messages and Suggested Fix

The following message appears when the `set_clock_latency` constraint has been specified with negative value for clocks `<clk-name-list>`:

```
[INFO] set_clock_latency is set to negative value for clocks  
<clk-name-list> [Also File(Lines): <file-name>(<num>) <file-  
name>(<num>)]
```

Where, `<file-name>(<num>)` is the file name and line number at which location some other limit value is being set to a negative number for the same object.

NOTE: *In case, more than one limit value (min/max/rise/fall etc.) is set to a negative value (in separate constraints), only a single message is issued for that object.*

Potential Issues

The latency should be either zero or some positive value.

Consequences of Not Fixing

Latency values being negative do not have any physical significance.

How to Debug and Fix

The SDC file highlights the clock latency constraints that have a negative value.

To resolve this violation, review the value of the constraints and set them to an appropriate positive value.

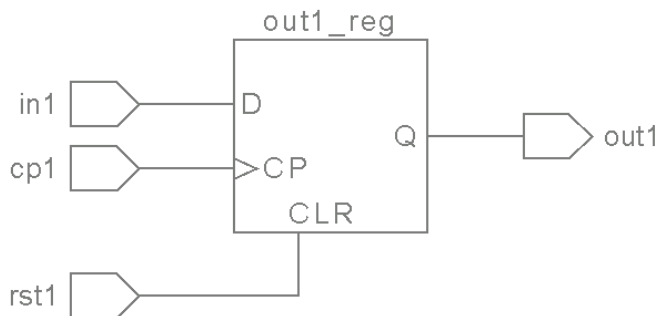
In case, more than one limit value (min/max/rise/fall etc.) is set to a negative value (in separate constraints), a single message is issued for that object as in the following example:

```
set_clock_latency is set to negative value for clocks
"B_clk2_w" [Also File(Lines): case1.sdc(19)]
```

The file names and lines numbers of other related specifications are also listed.

Example Code and/or Schematic**Example 1**[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top.sdc is as follows:

```
top.sdc top.sdc (2)
create clock cp1 -name clk1 -period 10 -waveform {0 5}
set_clock_latency -2.2 [get_clocks clk1]
set_clock_latency -0.2 [get_clocks clk1]
```

This rule reports a violation message because a negative *set_clock_latency* value is defined.

Example 2Test Case Files Not Available

In this example, the *Clk_Lat08* rule reports a violation because *set_clock_latency* is set to a negative value.

```
set_clock_latency -0.2 -rise -early [get_clocks clk1]
```

To resolve this violation, set the clock latency to a positive value.

Default Severity Label

Info

Rule Group

Clk_Lat

Reports and Related Files

None

Clk_Lat09

Identifies source or network latency that are not defined for a virtual clock

When to Use

This rule is applicable to the RTL, Pre-layout, Post-layout phases.

Description

The *Clk_Lat09* rule reports three types of violations:

- If neither source of network latency is specified for given virtual clock.
- If specified network latency is zero.
- If specified source latency is zero.

Latency with zero value is equivalent to unspecified.

Parameter(s)

None

Constraint(s)

SDC

- [*set_clock_latency*](#) (Optional): Use to specify the expected post-layout delay value. This defines the time taken by the clock from the root to the leaf of the tree.
- [*create_clock*](#) (Mandatory): Use to create a clock.

Messages and Suggested Fix

Message 1

The following message appears for a virtual clock *<clkv-name>* without the corresponding [*set_clock_latency*](#) constraint:

[WARNING] Either source or network latency has not been defined for virtual clock "*<clkv-name>*"

Message 2

The following message appears for a virtual clock *<clkv-name>* having a corresponding [*set_clock_latency*](#) constraint with source latency value of 0:

[WARNING] Zero source latency has been specified for virtual clock "<clkv-name>"

Message 3

The following message appears for a virtual clock <clkv-name> having a corresponding *set_clock_latency* constraint with network latency value of 0:

[WARNING] Zero network latency has been specified for virtual clock "<clkv-name>"

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

Virtual clocks usually correspond to real clocks. Therefore, they should also have latencies. Otherwise, timing inferred for Input/Outputs, specified with respect to virtual clocks, would be incorrect.

How to Debug and Fix

The latency constraints are highlighted in the SDC file for the virtual clocks. To resolve this issue, define the source or network latency and ensure the latency values are not zero.

Example Code and/or Schematic

In the following snippet, neither the network nor the source latency is specified for virtual clock `vc`. Therefore, the *Clk_Lat09* rule reports a violation.

```
//test.sdc
create_clock -name vc -period 20
```

Default Severity Label

Warning

Rule Group

Clk_Lat

Reports and Related Files

None

Clk_Lat10

Identifies `set_clock_latency` specified with `-early`/`-late` arguments

When to Use

This rule is applicable to the RTL, Pre-layout, Post-layout phases.

Description

The *Clk_Lat10* rule reports `set_clock_latency` constraints specified with the `-early` or `-late` arguments.

Parameter(s)

None

Constraint(s)

SDC

- `set_clock_latency` (Optional): Use to specify the expected post-layout delay value. This defines the time taken by the clock from the root to the leaf of the tree.

Messages and Suggested Fix

The following message appears a `set_clock_latency` constraint that has the `-early` or `-late` arguments specified:

```
[WARNING] Clock latency specified using '<option-list>'
option(s)
```

Where, `<option-list>` can be `-early`, `-late`, or `early late`.

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

Using the `-early` or `-late` arguments of the `set_clock_latency` constraint is a methodology choice.

How to Debug and Fix

In the SDC file, review the requirement of the `set_clock_latency` constraint.

Example Code and/or Schematic

For the following snippet, the *Clk_Lat10* rule reports a violation message because the [set_clock_latency](#) constraint is specified with the `-late` argument.

```
set_clock_latency 1.0 -source -late clk1
```

Default Severity Label

Warning

Rule Group

Clk_Lat

Reports and Related Files

None

Clk_Lat12

Identifies clocks for which source latency is not defined

When to Use

This rule is applicable to the Pre-layout and Post-layout phases.

Description

The Clk_Lat12 rule reports clocks for which source latency is not specified in the pre-layout or post-layout analysis phases. These clocks are specified using the *create_clock* constraint.

Parameter(s)

None

Constraint(s)

SDC

- *set_clock_latency* (Optional): Use to specify the expected post-layout delay value. This defines the time taken by the clock from the root to the leaf of the tree.
- *create_clock* (Mandatory): Use to define a clock

Message and Suggested Fix

The following message appears if source latency is not specified for a clock that is specified using the *create_clock* constraint:

[WARNING] Source latency is not specified for clock <clk-name>

Potential Issues

Source latency models the off-chip delay between the actual clock generation point to the point of the design where create clock is defined. Therefore, irrespective of phase, this delay should be modeled. If source latency is not defined, this violation message is reported.

Consequences of Not Fixing

If off-chip delay is not modeled for a clock, the timing analysis will not be accurate since it is missing this delay, which actually will be present in the real design.

How to Debug and Fix

In the SDC file, specify the `set_clock_latency` constraint for each `create_clock` constraint.

Example Code and/or Schematic

In the following snippet, source latency is not defined for `clk1`. Therefore, the `Clk_Lat12` rule reports a violation message.

```
//test.sdc
create_clock -name clk1 -period 20 [get_ports {in}]
```

Default Severity Label

Warning

Rule Group

Clk_Lat

Reports and Related Files

None

Clock Transition Rules

The Clock Transition Rules Sub-group Clk_Trans contains the following rules:

Rule	Flags
<i>Clk_Trans02</i>	Clocks for which the <i>set_clock_transition</i> constraint has not been specified
<i>Clk_Trans02a</i>	Clock ports for which the <i>set_input_transition</i> or the <i>set_driving_cell</i> constraint have not been specified
<i>Clk_Trans03</i>	Clocks with improper clock transition values
<i>Clk_Trans04</i>	Flip-flop clock pins or latch enable pins without <i>set_annotated_transition</i> value set
<i>Clk_Trans05</i>	Improper annotated transition values
<i>Clk_Trans06</i>	Clock ports with <i>set_input_transition</i> and <i>set_driving_cell</i> constraints set in the pre-layout phase
<i>Clk_Trans07</i>	Clock ports without <i>set_input_transition</i> constraint set in the post-layout phase
<i>Clk_Trans08</i>	Clock ports with <i>set_driving_cell</i> constraint set in the post-layout phase
<i>Clk_Trans09</i>	<i>set_clock_transition</i> constraints that are not explicitly set with all required options in the RTL or prelayout phases
<i>Clk_Trans11</i>	<i>set_clock_transition</i> constraints set on clocks in the postlayout phase
<i>Clk_Trans12</i>	<i>set_clock_transition</i> constraints (in RTL and prelayout level) and <i>set_input_transition</i> constraints (in postlayout level) with inconsistent maximum and minimum values
<i>Clk_Trans13</i>	<i>set_input_transition</i> constraints with unsupported (for synthesis) options (-clock or -clock_fall) in the postlayout phase
<i>Clk_Trans15</i>	<i>set_input_transition</i> constraints that are not explicitly set with all required options in the postlayout phase
<i>Clk_Trans16</i>	<i>set_clock_transition</i> constraints set on virtual clocks
<i>Clk_Trans17</i>	Improperly specified clock transition constraints

Clk_Trans02

Identifies `set_clock_transition` that have not been defined

When to Use

To ensure clock transition characterization for clocks in the RTL and Pre-layout phases

Description

The *Clk_Trans02* rule reports clocks for which the [set_clock_transition](#) constraint has not been specified. You specify these clocks using the [create_clock](#) or [create_generated_clock](#) constraint. Transition in this context is the time taken by the clock to switch from 0 to 1 or vice versa.

Rule Exceptions

The *Clk_Trans02* rule does not report a violation for virtual clocks.

Parameter(s)

None

Constraint(s)

SDC

- [create_clock](#) (Mandatory): Use to create a clock
- [create_generated_clock](#) (Mandatory): Use to create a clock
- [set_clock_transition](#) (Mandatory): Use to characterize clock transition times, which is the time taken by clock to change from 0 to 1 and vice-versa.

Messages and Suggested Fix

Message 1

The following message appears for the real clock `<clk-name>`:

```
[WARNING] set_clock_transition not applied for clock "<clk-name>" declared using create_clock
```

For debugging information, refer to [How to Debug and Fix](#).

Message 2

The following message appears for the generated clock `<clk-name>`:

```
[WARNING] set_clock_transition not applied for clock "<clk-name>" declared using create_generated_clock
```

Potential Issues

Implementation tools might assume undesired clock transition for the clock. This affects the static timing analysis for the design.

Consequences of Not Fixing

Synthesis may use over optimistic timing values in delay calculations, while actual delays in post-layout may cause timing failures.

How to Debug and Fix

In the SDC file, specify the [set_clock_transition](#) constraint for the clocks stated in the violation message.

Example Code and/or Schematic

For the following SDC file snippet, the `Clk_Trans02` rule reports a violation because only [create_clock](#) is specified and [set_clock_transition](#) is not specified.

```
create_clock -name clk -period 10 [get_ports {clk}]
```

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans02a

Reports clock transition rate that have not been defined either using `set_driving_cell` or `set_input_transition`

When to Use

To ensure clock transition characterization through the `set_driving_cell` or `set_input_transition` constraints in the Post-layout phase

Description

The `Clk_Trans02a` rule reports clock ports defined in the `create_clock` constraint for which either the `set_input_transition` or the `set_driving_cell` constraint have not been specified. Transition in this context is the time taken by clock to switch from 0 to 1 or vice versa.

Rule Exceptions

The `Clk_Trans02a` rule does not report a violation for virtual and generated clocks.

Parameter(s)

None

Constraint(s)

SDC

- `create_clock` (Mandatory): Use to create a clock.
- `set_input_transition` (Optional): Use to characterize the transition time taken to change the input signal from 0 to 1 and vice-versa.
- `set_driving_cell` (Optional): Use to set the driving cell for a port.

Messages and Suggested Fix

The following message appears at the location of a `create_clock` constraint on the port `<port-name>` for the clock `<clk-name>` when a `set_input_transition` or a `set_driving_cell` constraint is not specified for the port:

```
[WARNING] Clock transition rate not specified for port
"<port-name>" associated with create_clock "<clk-name>"
```

Potential Issues

The port is not declared through the [set_driving_cell](#) or [set_input_transition](#) constraints. Implementation tools might assume undesired clock transition for the clock. This affects the static timing analysis for the design.

Consequences of Not Fixing

In post-layout, the clock transition and delay values are propagated through the network. If clock transition is not specified, over-optimistic values are propagated through the network by the tool leading to timing violations in the chip. Therefore, define the transition values at the input port so that the values are propagated across the clock network.

How to Debug and Fix

Specify the [set_clock_transition](#) constraint for the port stated in the violation message.

Example Code and/or Schematic

For the following SDC file snippet, the *Clk_Trans02a* rule reports a violation because only [create_clock](#) is specified and neither the [set_clock_transition](#) nor [set_driving_cell](#) is specified.

```
create_clock -name clk -period 10 [get_ports {clk}]
```

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans03

Reports real clocks that have clock transition values outside the specified range

When to Use

This rule is applicable to the RTL and Pre-layout phases.

Description

The *Clk_Trans03* rule identifies real clocks with improper clock transition values specified with the *set_clock_transition* constraints. This rule requires the input transition value should be within the maximum and minimum allowed transition values.

The *Clk_Trans03* rule checks the clock transition values specified with the *set_clock_transition* constraints as follows:

1. If no library file is supplied, clock transition value limits are set by the *default_max_transition* and *default_min_transition* parameters.
2. If a library file is supplied and the clock object is a port or block pin, clock transition values are compared with the default maximum transition.
3. If a library file is supplied and the clock object is a pin on a cell instance, clock transition values are compared against the minimum and maximum transition times for that pin.

NOTE: *If the transition value is 0 and the value of the *default_max_transition* parameter is also 0, the Clk_Trans03 rule reports a violation message.*

Rule Exceptions

The *Clk_Trans03* rule ignores virtual clocks.

Parameter(s)

- *default_max_transition*: Default is 5.0. Set the value to a positive float value to specify the maximum transition value.
- *default_min_transition*: Default is 0. Set the value to a positive float value to specify the minimum transition value.

Constraint(s)

SDC

- [create_clock](#) (Mandatory): Use to create a clock
- [create_generated_clock](#) (Mandatory): Use to create a clock
- [set_clock_transition](#) (Mandatory): Use to characterize clock transition times, which is the time taken by the clock to change from 0 to 1 and vice versa.

Messages and Suggested Fix

Message 1

The following message appears for a clock `<clk-name>` where its clock transition value `<value>` is more than the maximum clock transition value `<max>` specified in the associated technology library (`.lib` file) or specified through the [default_max_transition](#) parameter (if the value is not set in the technology library) for the corresponding port/pin `<portpin-name>`:

[WARNING] Clock transition value `<value>` set on `<clk-type>` clock "`<clk-name>`" is greater than the maximum transition value `<max>` (specified through technology library or rule parameter) of corresponding port/pin "`<portpin-name>`"

Where, `<clk-type>` can be [create_clock](#) or [create_generated_clock](#).

Message 2

The following message appears for a clock `<clk-name>` where its clock transition value `<value>` is less than or equal to the minimum clock transition value `<min>` specified in the associated technology library (`.lib` file) or specified through the [default_min_transition](#) parameter (if the value is not set in the technology library) for the corresponding port/pin `<portpin-name>`:

[WARNING] Clock transition value `<value>` set on `<clk-type>` clock "`<clk-name>`" is not greater than the minimum transition value `<min>` (specified through technology library or rule parameter) of corresponding port/pin "`<portpin-name>`"

Where, `<clk-type>` can be [create_clock](#) or [create_generated_clock](#).

Potential Issues

The violation messages appear because the clock transition value set is not within the clock transition range.

Consequences of Not Fixing

Every technology imposes certain restrictions on the design functioning. If constraints are chosen beyond these restrictions, the design may not operate as desired. Since the value of clock transition as constrained is greater than the maximum or lesser than minimum, the design may not operate as desired in the current technology.

How to Debug and Fix

Specify value in the range reported in the violation message.

Example Code and/or Schematic

For the following snippet, the *Clk_Trans03* rule reports a violation message because the transition value specified in the SDC file, 11.0, is greater than specified in library, 10.5.

```
//test.sdc
create_clock -name clk1 -period 10.000000 -waveform {
0.000000 5.000000 } {clk_a}
set_clock_transition 11.0 clk1

//test.lib
default_max_transition : 10.5;
```

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans04

Reports clock pins that do not have a `set_annotated_transition` constraint defined

When to Use

Use this rule in the RTL and Pre-layout phases.

Description

The *Clk_Trans04* rule reports sequential element clock pins that do not have the *set_annotated_transition* constraint set. Every sequential element clock pin should have a *set_annotated_transition* value in the delay calculation mode. Transition in this context is time taken by the clock pin to switch from 0 to 1 or vice-versa.

set_annotated_transition is typically applied in delay calculation mode and while using it *set_clock_transition* should not be used.

Parameter(s)

None

Constraint(s)

SDC

- *create_clock* (Mandatory): Use to create a clock
- *create_generated_clock* (Mandatory): Use to create a clock
- *set_annotated_transition* (Optional): Use set the transition time to be annotated on specified pins in the current design.
- *set_clock_transition* (Mandatory): Use to characterize clock transition times, which is the time taken by clock to change from 0 to 1 and vice-versa.

Messages and Suggested Fix

The following message appears for a sequential element clock pin *<pin-name>* of instance *<inst-name>* that is driven by net *<net-name>* in design unit *<name>* when there is no *set_annotated_transition* value set on the sequential element clock pin:

[WARNING] set_annotated_transition constraint not defined for a <inst-name> sequential cell clock pin <pin-name> driven by net '<net-name>' of design "<name>"

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

This constraint is typically used in delay calculation mode and hence if it is not specified, delays will not get modeled accurately.

How to Debug and Fix

Specify transition delay values.

Example Code and/or Schematic

If you have not specified [set_annotated_transition](#) for a sequential instance, the *Clk_Trans04* rule reports a violation message. However, if the constraint is specified for that instance, this rule does not report a violation:

```
set_annotated_transition 1.0 top/subblock/cell/CP
```

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans05

Reports set_annotated_transition value not within technology bounds

When to Use

This rule is applicable to the RTL and Pre-layout phases.

Description

The *Clk_Trans05* rule identifies improper annotated transition values specified using the *set_annotated_transition* constraints. This rule requires the input transition value should be within the maximum and minimum allowed transition values.

The *Clk_Trans05* rule checks the annotated transition values specified with the *set_annotated_transition* constraints as follows:

1. If no library file is supplied, annotated transition value limits are set by the *default_max_transition* and *default_min_transition* parameters.
2. If a library file is supplied and the clock object is a port or block pin, annotated transition values are compared with the default maximum transition.
3. If a library file is supplied and the clock object is a pin on a cell instance, annotated transition values are compared against the minimum and maximum transition times for that pin.

NOTE: *If the transition value is 0 and the value of the *default_max_transition* parameter is also 0, the Clk_Trans05 rule reports a violation message.*

Parameter(s)

- *default_max_transition*: Default is 5.0. Set the value to a positive float value to specify the maximum transition value.
- *default_min_transition*: Default is 0. Set the value to a positive float value to specify the minimum transition value.

Constraint(s)

SDC

- *create_clock* (Mandatory): Use to create a clock.
- *create_generated_clock* (Mandatory): Use to create a clock.

- *set_annotated_transition* (Mandatory): Use to set the transition time to be annotated on the specified pins in the current design.

Messages and Suggested Fix

Message 1

The following message appears when an annotated transition value *<value>* is more than the maximum annotated transition value *<max>* specified in the associated technology library (.lib file) or specified through *default_max_transition* rule parameter (if the value is not set in the technology library):

```
set_annotated_transition_value <value> is greater than the
maximum transition value <max> (specified through technology
library or rule parameter)
```

Message 2

The following message appears when an annotated transition value *<value>* is less than or equal to the minimum annotated transition value *<min>* specified in the associated technology library (.lib file) or specified through *default_min_transition* rule parameter (if the value is not set in the technology library):

```
set_annotated_transition_value <value> is not greater than the
minimum transition value <min> (specified through technology
library or rule parameter)
```

Potential Issues

The violation messages appear because the annotated transition value set is not within the range.

Consequences of Not Fixing

Every technology imposes certain restrictions on the design functioning. If constraints are chosen beyond these restrictions, the design may not operate as desired.

How to Debug and Fix

Specify the value in the range reported in the violation message.

Example Code and/or Schematic

For the following snippet, the *Clk_Trans03* rule reports a violation message

because the transition value specified in the SDC file, 11.0, is greater than specified in library, 10.5.

```
//test.sdc
create_clock -name clk1 -period 10.000000 -waveform {
0.000000 5.000000 } {clka}
set_clock_transition 11.0 clk1
```

```
//test.lib
default_max_transition : 10.5;
```

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans06

Reports instances of `set_input_transition` and `set_driving_cell` that have been used on clock ports in pre-layout

When to Use

RTL phase, Pre-layout phase

Description

The `Clk_Trans06` rule reports clock ports with `set_input_transition` and `set_driving_cell` constraints set in the pre-layout phase. These constraints on clock ports can result in DRC (Design Rule Check) violations due to transition propagation on clock nets that have a high fan-out.

You specify clock ports using the `create_clock` constraint.

Parameter(s)

None

Constraint(s)

SDC

- `create_clock` (Mandatory): Use to create a clock
- `set_input_transition` (Mandatory): Use to characterize the transition time taken to change the input signal from 0 to 1 and vice-versa.

Messages and Suggested Fix

The following message appears when a clock port `<clk-name>` has a `set_input_transition` constraint or `set_driving_cell` constraint set for the pre-layout phase:

```
[WARNING] <constr> on clock port "<clk-name>" is not recommended
```

Where, `<constr>` can be `set_input_transition` or `set_driving_cell`.

Potential Issues

In the pre-layout phase, specifying `set_input_transition` or `set_driving_cell` on clock ports can lead to Design Rule Check violations for the clock port.

Consequences of Not Fixing

Setting a drive on a clock port has problems. With an "ideal" clock you get 0 ns propagation delay, therefore the drive strength is ignored. However, when a drive is set on a clock you get a transition delay. This transition delay can be significant because the load on the clock is large. The libraries are typically not characterized to handle large transition delays. As a result, you may have false propagation delays, which can be negative sometimes.

How to Debug and Fix

The [set_input_transition](#) or the [set_driving_cell](#) constraint is highlighted in the SDC file. These constraints are not recommended in the clock ports during the pre-layout design stage.

To resolve this violation, remove the constraint from the SDC file.

Example Code and/or Schematic

In the following SDC file snippet, the *Clk_Trans06* rule reports a violation because [set_driving_cell](#) is set on a clock port.

```
set_driving_cell -lib_cell AND [get_ports CLK]
```

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans07

Reports clock ports that do not have `set_input_transition`

When to Use

Use this rule in the Post-layout phase.

Description

The *Clk_Trans07* rule reports primary clock ports that do not have the `set_input_transition` constraint set in the Post-layout phase. You can specify primary clock ports using the `create_clock` constraint.

Parameter(s)

None

Constraint(s)

SDC

- `create_clock` (Mandatory): Use to create a clock
- `set_input_transition` (Optional): Use to characterize the transition time taken to change the input signal from 0 to 1 and vice-versa.

Messages and Suggested Fix

The following message appears when a primary port `<port-name>` related to clock `<clk-name>` has no `set_input_transition` constraint set for the postlayout phase:

```
set_input_transition is not defined on port "<port-name>"  
(relating to clock "<clk-name>")
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

In post-layout, the clock transition and delay values are propagated through the network. Therefore, you should define the transition values at the input port to propagate the values across the clock network.

In addition, `set_clock_transition` should not be used in post-layout for clock

nets because this command overrides the calculated value on the clock pins of the registers and associated nets.

How to Debug and Fix

Specify [set_input_transition](#) for clock ports.

Example Code and/or Schematic

For the following snippet, the Clk_Trans07 rule reports a violation message because [set_input_transition](#) is not specified for port in.

```
create_clock -name c1 -period 20 in
```

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans08

Reports `set_driving_cell` used on clock ports in post-layout stage

When to Use

Use this rule in the Post-layout phase.

Description

The *Clk_Trans08* rule reports primary clock ports that have the `set_driving_cell` constraint set in the post-layout phase. You can specify primary clock ports using the `create_clock` constraint.

Parameter(s)

None

Constraint(s)

SDC

- `create_clock` (Mandatory): Use to create a clock.
- `create_generated_clock` (Mandatory): Use to create a clock.
- `set_driving_cell` (Optional): Use to set the driving cell for a port.

Messages and Suggested Fix

The following message appears when a primary clock port `<clk-name>` has `set_driving_cell` constraint set for the post-layout phase:

```
[WARNING] set_driving_cell on clock port "<clk-name>" not recommended in postlayout
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Typically, `set_driving_cell` is not used for clock ports rather `set_input_transition` is used since clocks are modeled quite accurately so it is not left to the cell driving the clock port to decide the transition time for the clock.

How to Debug and Fix

Remove the *set_driving_cell* constraint.

Example Code and/or Schematic

For the following snippet, the *Clk_Trans08* rule reports a violation because *set_driving_cell* is applied on the CLK port.

```
create_clock -name Clk1 -period 10.000000 -waveform {  
0.000000 5.000000 } {CLK}  
  
set_driving_cell -lib_cell RTL_NOT -library rtlc.prim.slf  
[all_inputs]
```

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans09

Identifies clocks that have `set_clock_transition` specified with incomplete options

When to Use

This rule is applicable to the RTL and Pre-layout phases.

Description

The *Clk_Trans09* rule reports *set_clock_transition* constraints that are not explicitly set with all required options. This rule checks that a *set_clock_transition* constraint is not explicitly specified with only one of the `rise` and `fall` options or one of the `max` and `min` options.

Parameter(s)

- *tc_ignore_min*: Default is no. Set the value to yes to ignore the missing -min option for worst-case SDC.

Constraint(s)

SDC

- *create_clock* (Optional): Use to create a clock.
- *create_generated_clock* (Optional): Use to create a clock.
- *set_clock_transition* (Mandatory): Use to characterize clock transition times, which is the time taken by clock to change from 0 to 1 and vice-versa.

Messages and Suggested Fix

The following message appears for a *set_clock_transition* constraint on a clock `<clk-name>` (specified using the *create_clock* or *create_generated_clock* constraint in file `<file-name>`, line `<num>`) in block `<blk-name>` that has not been specified with all required options in the RTL or pre-layout phases:

Not all `set_clock_transition` options specified explicitly for clock "`<clk-name>`" (file "`<file-name>`", line `<num>`) of block "`<blk-name>`" (Specified: [`<option-list1>`], Not specified: [`<option-list2>`])

Where, *<option-list1>* is the list of options that have specified for the clock in the following format:

```
[-option1](<file1>:<line1>,<line2> <file2>:<line1,line2>
... )
[-option2](<file1>:<line1>,<line2> <file2>:<line1,line2>
... )
```

<option-list2> is the list of options for which [set_clock_transition](#) constraint(s) are not found in the following format:

```
[-option3] [-option4]...
```

Potential Issues

The [set_clock_transition](#) constraints are not explicitly defined with all required combinations of max/min and rise/fall options. For example, if [set_clock_transition](#) constraints have been specified for a clock with the max and rise options, it is expected that there must exist other [set_clock_transition](#) constraints for the same clock with max/fall, min/rise, and min/fall options.

Consequences of Not Fixing

If only one option is present, the value for the other option is assumed to be zero, which is highly optimistic. This could falsely impact the delay through the clock tree.

How to Debug and Fix

Specify the missing options

Example Code and/or Schematic

For the following snippet, the *Clk_Trans09* rule report violation for options `[[-min -rise] [-min -fall] [-max -fall]]`.

```
create_clock -name clka -period 10.000000 -waveform {
0.000000 5.000000 } {clk1}
set_clock_transition 3.1 -max -rise [find clock clka]
```

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans11

Reports clocks that have `set_clock_transition` set

When to Use

This rule is applicable to the Post-layout phase.

Description

The *Clk_Trans11* rule reports `set_clock_transition` constraints set on clocks in the Post-layout phase.

Parameter(s)

None

Constraint(s)

SDC

- `create_clock` (Optional): Use to create a clock.
- `create_generated_clock` (Optional): Use to create a clock.
- `set_clock_transition` (Mandatory): Use to characterize clock transition times, which is the time taken by clock to change from 0 to 1 and vice-versa.

Messages and Suggested Fix

The following message appears for a `set_clock_transition` constraint on a clock `<clk-name>` for design unit `<du-name>` in the postlayout phase:
 set_clock_transition detected into constraints for module "<du-name>" (for <clk-type> clock "<clk-name>") (if your context is postlayout, this command must be removed)

Where, `<clk-type>` can be `create_clock` or `create_generated_clock`.

Potential Issues

In post-layout, transition values are supposed to be computed through the clock network.

Consequences of Not Fixing

If you specify `set_clock_transition`, it directly annotates the transition value

on the clock terminal of the register, which could be different from what will actually propagate.

How to Debug and Fix

Remove the `set_clock_transition` constraint from the SDC file.

Example Code and/or Schematic

Assume that the design is in the Post-layout phase. For the following snippet, the `Clk_Trans11` rule reports a violation because `set_clock_transition` is in the SDC file.

```
create_clock -name clka -period 10.000000 -waveform {  
0.000000 5.000000 } {clk1}  
  
set_clock_transition 3.1 -max -rise [find clock clka]
```

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans12

Identifies clock transition specified with inconsistent option values

When to Use

To ensure consistency in clock transition values for options in the RTL, Pre-layout, and Post-layout phases

Description

The *Clk_Trans12* rule identifies *set_clock_transition* constraints specified in the RTL and pre-layout phases, and *set_input_transition* constraints, specified in the post-layout phase, with inconsistent maximum and minimum values.

Parameter(s)

None

Constraint(s)

SDC

- *set_input_transition* (Mandatory): Use to characterize the transition time taken to change the input signal from 0 to 1 and vice-versa.
- *set_clock_transition* (Mandatory): Use to characterize clock transition times, which is the time taken by clock to change from 0 to 1 and vice-versa.

Messages and Suggested Fix

Message 1

The following message appears for a *set_clock_transition* constraint on the clock *<clk-name>* in the design/block *<name>* where the maximum transition value *<value1>* is less than the minimum transition value *<value2>* in the RTL or pre-layout phases:

```
[WARNING] <type1> clock transition value <value1> is less than
<type2> transition value <value2> for <clk-type> clock
"<clk-name>" of design/block "<name>"
```

Where,

- `<type1>` can be `Max_Fall` or `Max_Rise`
- `<type2>` can be `Min_Fall` or `Min_Rise`
- `<clk-type>` can be `create` or `generated`

For debugging information, refer to [How to Debug and Fix](#).

Message 2

The following message appears for a [set_input_transition](#) constraint on port `<port-name>` for design/block `<name>` in the postlayout phase where the maximum transition value `<value1>` is less than the minimum transition value `<value2>` in the post-layout phase:

[WARNING] `<type1>` input transition value `<value1>` is less than `<type2>` transition value `<value2>` for clock input "`<port-name>`" of design/block "`<name>`"

Where,

- `<type1>` can be `Max_Fall` or `Max_Rise`
- `<type2>` can be `Min_Fall` or `Min_Rise`

Potential Issues

The minimum value is greater than the maximum value.

Consequences of Not Fixing

If the minimum value is greater than the maximum value, the implementation tools internally modify some of the values. The modification might not be required. It could potentially lead to incorrect timing analysis.

How to Debug and Fix

Both input transition constraints stated in the violation message are highlighted in the SDC file. Review the transition values and modify them appropriately.

Example Code and/or Schematic

For the following SDC file snippet, the `Clk_Trans12` rule reports a violation because the max value is less than min value.

```
set_clock_transition 0.35 -max [get_clocks CLK1]
set_clock_transition 0.55 -min [get_clocks CLK2]
```

Clock Rules

```
set_input_transition 0.33 -max [get_ports DATA*]  
set_input_transition 0.55 -min [get_ports DATA*]
```

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans13

Reports unusual `set_input_transition` option usage

When to Use

This rule is applicable to the Post-layout phase.

Description

The *Clk_Trans13* rule reports *set_input_transition* constraints with unusual options, such as `clock` or `clock_fall`, in the Post-layout phase.

Prerequisites

Set the *pt* parameter to `yes` to use this rule.

Parameter(s)

- *pt*: Default is `yes`. This indicates the *Clk_Trans13* rule shall be used.

Constraint(s)

SDC

- *create_clock* (Optional): Use to create a clock.
- *create_generated_clock* (Optional): Use to create a clock.
- *set_input_transition* (Mandatory): Use to characterize clock transition times, which is the time taken by clock to change from 0 to 1 and vice-versa.

Messages and Suggested Fix

The following message appears for a *set_input_transition* constraint for clock `<clk-name>` with the `-clock` and/or `-clock_fall` option:

```
Unusual option(s) "<option-list>", used in command
"set_input_transition" are not supported by all implementation
tools
```

Where, `<option-list>` is `clock` if only the `-clock` option is specified, `clock_fall` if only the `-clock_fall` option is used, and `clock clock_fall` if both the `-clock` and `-clock_fall` options are specified.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Some [set_input_transition](#) options are not supported by certain implementation tools and should not be used.

How to Debug and Fix

Remove the unusual options from the constraint or remove the constraint from the SDC file.

Example Code and/or Schematic

For the following snippet, the *Clk_Trans13* rule reports a violation because the `clock_fall` argument is set.

```
//test.sdc  
set_input_transition -clock_fall 2 {inport}
```

Default Severity Label

Info

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans15

Identifies clocks that have `set_input_transition` specified with incomplete arguments

When to Use

This rule is applicable to the Post-layout phase.

Description

The *Clk_Trans15* rule reports *set_input_transition* constraints that are not explicitly set with all required arguments in the Post-layout phase. This rule checks that *set_input_transition* is not explicitly specified with only one of the `rise` and `fall` arguments or one of the `max` and `min` arguments.

Parameter(s)

- *tc_ignore_min*: Default is no. Set the value to yes to ignore the missing -min option for worst-case SDC.

Constraint(s)

SDC

- *create_clock* (Optional): Use to create a clock.
- *create_generated_clock* (Optional): Use to create a clock.
- *set_input_transition* (Mandatory): Use to characterize clock transition times, which is the time taken by clock to change from 0 to 1 and vice-versa.

Messages and Suggested Fix

The following message appears for a *set_input_transition* constraint on a clock port `<port-name>` in design unit or block `<name>` that has not been specified with all required options in the Post-layout phase:

Not all set_input_transition options specified explicitly for port "<port-name>" of design "<name>" (Specified: <option-list1>, Not specified: <option-list2>)

Where, `<option-list1>` is the list of options that have specified for the clock in the following format:


```
[-option1](<file1>:<line1>,<line2> <file2>:<line1,line2>
... )
[-option2](<file1>:<line1>,<line2> <file2>:<line1,line2>
... )
```

<option-list2> is the list of options for which *set_input_transition* constraint(s) are not found in the following format:

```
[-option3] [-option4]...
```

Potential Issues

The *set_input_transition* constraints are not explicitly defined with all required combinations of max/min and rise/fall arguments. For example, if *set_input_transition* constraints have been specified for a clock port with max and rise arguments, it is expected that there must exist other *set_input_transition* constraints for the same clock port with max/fall, min/rise, and min/fall arguments.

Consequences of Not Fixing

If only one option is present, the value for the other option is assumed to be 0 which is highly optimistic. This could falsely impact the delay through the clock tree.

How to Debug and Fix

Specify the missing options.

Example Code and/or Schematic

In the following snippet, the *set_input_transition* constraint is specified for the min rise and min fall arguments. The *Clk_Trans15* rule reports a violation because the max rise and max fall arguments have not been specified.

```
//test.sdc
set_input_transition 5 -min {data[0] data[1] data[2]}
```

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans16

Reports `set_clock_transition` on virtual clocks

When to Use

To report clock transition applied on virtual clocks in the RTL and Pre-layout phases

Description

The *Clk_Trans16* rule reports [set_clock_transition](#) constraints set on virtual clocks. You can create virtual clocks by using the [create_clock](#) constraint with the `-name` argument instead of referencing a real port/pin.

Parameter(s)

None

Constraint(s)

SDC

- [set_clock_transition](#) (Mandatory): Use to characterize clock transition times, which is the time taken by clock to change from 0 to 1 and vice-versa.
- [create_clock](#) (Optional): Use to create a clock.

Messages and Suggested Fix

The following message appears for a virtual clock `<clkv-name>` with a [set_clock_transition](#) constraint set:

```
[WARNING] set_clock_transition has been set on virtual clock
"<clkv-name>"
```

Potential Issues

Clock transition on a virtual clock is unnecessary because a virtual clock is not applied on a design object. This can lead to confusion for the person reading the SDC file.

Consequences of Not Fixing

Since virtual clocks cannot be propagated, there is no need for specifying transition constraints on them. If a virtual clock has a transition value, it is

ignored in the timing analysis by the tool.

How to Debug and Fix

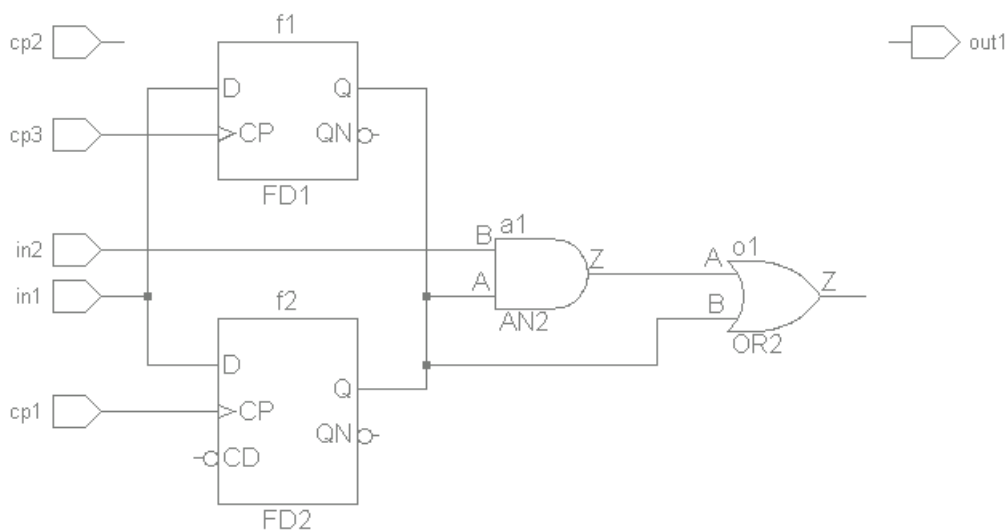
The SDC file highlights the clock transition constraints. The constraint should be removed since it is not meaningful to specify *set_clock_transition* on virtual clocks.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top .sdc is as follows:

Clock Rules

```
top.sdc | top.sdc (2) |
create_clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } { cp1 }
create_clock -name vclk1 -period 10.000000 -waveform { 0.000000 5.000000 }
create_clock -name vclk2 -period 10.000000 -waveform { 0.000000 5.000000 }
set_clock_transition 10 {vclk2}
```

This rule reports a violation message because a *set_clock_transition* is defined on the virtual clock.

Example 2

Test Case Files Not Available

For the following SDC file snippet, the *Clk_Trans16* rule reports a violation because VLCK1/VCLK2 are virtual clocks and *set_clock_transition* is set on them.

```
set_clock_transition 0.44 -rise [get_clocks VCLK1]
set_clock_transition 0.45 -fall [get_clocks VCLK1]
```

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clk_Trans17

Reports clock transition without proper constraint

When to Use

To ensure clock transition is defined through the appropriate constraint in the RTL, Pre-layout, and Post-layout phases

Description

The *Clk_Trans17* rule reports clock transition constraints that have not been specified correctly.

By default, all controlling rule parameters are set to ignore and the *Clk_Trans17* rule does not perform any checking. The *Clk_Trans17* rule checks for the following constraints specified for clocks ports based on the setting of the corresponding rule parameter:

- *set_clock_transition* constraints is controlled by the *tc_ct17_clock_trans* parameter.
- *set_drive* and *set_load* constraints is controlled by the *tc_ct17_drive* parameter.
- *set_driving_cell* constraints is controlled by the *tc_ct17_driving_cell* parameter.
- *set_input_transition* constraints is controlled by the *tc_ct17_input_trans* parameter.

Refer to the *Parameter(s)* section to learn how to use these parameters to control the behavior of the *Clk_Trans17* rule.

Parameter(s)

- *tc_ct17_input_trans*: Default is ignore. Set the value to disallow to make the *Clk_Trans17* rule report *set_input_transition* constraints that are present. Set the value to mandatory to report *set_input_transition* constraints that are missing.
- *tc_ct17_drive*: Default is ignore. Set the value to disallow to make the *Clk_Trans17* rule report *set_drive* constraints that are present. Set the value to mandatory to report *set_drive* constraints that are missing.

- *tc_ct17_driving_cell*: Default is ignore. Set the value to disallow to make the *Clk_Trans17* rule report *set_driving_cell* constraints that are present. Set the value to mandatory to report *set_driving_cell* constraints that are missing.
- *tc_ct17_clock_trans*: Default is ignore. Set the value to disallow to make the *Clk_Trans17* rule report *set_input_transition* constraints that are present. Set the value to mandatory to report *set_input_transition* constraints that are missing.

Constraint(s)

SDC

- *set_clock_transition* (Optional): Use to characterize clock transition times, which is the time taken by clock to change from 0 to 1 and vice-versa.
- *set_drive* (Optional): Use to set the resistance on specified input or inout ports in the current design.
- *set_load* (Optional): Use to set the capacitance on specified ports and nets in the current design.
- *set_driving_cell* (Optional): Use to set the driving cell for a port.
- *set_input_transition* (Optional): Use to characterize the transition time taken to change the input signal from 0 to 1 and vice-versa.

Messages and Suggested Fix

Message 1

The following message appears when the constraint *<constr>* is specified on the clock port *<port-name>* related to the clock *<clk-name>*, but should not be specified based on the rule parameter controlling that constraint:

[WARNING] *<constr>* on clock port "*<port-name>*" (relating to clock "*<clk-name>*") is not recommended

For debugging information, refer to [How to Debug and Fix](#).

Message 2

The following message appears when the constraint *<constr>* has not been specified on the clock port *<port-name>* related to the clock

`<clk-name>`, but should have been specified based on the value of the rule parameter controlling that constraint:

[WARNING] `<constr>` on clock port "`<port-name>`" (relating to clock "`<clk-name>`") is not specified

Potential Issues

This is a Methodology rule. As per the methodology requirements, the constraint should not be specified.

Consequences of Not Fixing

This is a Methodology rule. As per the methodology requirements, the constraint should not be specified.

How to Debug and Fix

The constraint stated in the message is not specified on the clock port. The parameter that corresponds to this constraint has been set to mandatory. Refer to the [Parameter\(s\)](#) section to learn how to use these parameters to control the behavior of the `Clk_Trans17` rule.

Message 3

The following message appears when none of the controlling rule parameters have been specified or have been set to ignore:

[WARNING] All parameters `tc_ct17_input_trans`, `tc_ct17_drive`, `tc_ct17_clock_trans` and `tc_ct17_driving_cell` are set to "ignore" and therefore rule will not be checked

Potential Issues

All rule controlling parameters are set to ignore. Therefore, there is nothing to examine.

Consequences of Not Fixing

All rule controlling parameters are set to ignore. Therefore, there is nothing to examine.

How to Debug and Fix

By default, the `Clk_Trans17` rule does not check for the [set_clock_transition](#), [set_drive](#) and [set_load](#), [set_driving_cell](#), and [set_input_transition](#) constraints specified for clocks ports.

To check for these constraints, set the corresponding parameter value to allow. Refer to the [Parameter\(s\)](#) section to control the behavior of the

Clk_Trans17 rule.

Example Code and/or Schematic

Suppose the *tc_ct17_clock_trans* is set to Mandatory. The SDC file contains the following:

```
create_clock -name CLK1 -period 10 [get_ports CLK1]
create_clock -name CLK2 -period 20 [get_ports CLK2]
set_clock_transition 0.5 [get_clocks CLK1]
```

In this example, the *Clk_Trans17* rule reports a violation because *set_clock_transition* is not set on the clock port CLK2.

Similarly, all missing constraints are reported when the appropriate parameter is set to mandatory. Refer to the *Parameter(s)* section to learn how to control the behavior of the *Clk_Trans17* rule.

Default Severity Label

Warning

Rule Group

Clk_Trans

Reports and Related Files

None

Clock Uncertainty Rules

The Clock Uncertainty Rules Sub-group Clk_Uncert contains the following rules:

Rule	Flags
Clk_Uncert01	Incorrect Clock Uncertainty constraints
Clk_Uncert02a	Missing set_clock_uncertainty for real clocks
Clk_Uncert02b	Incompletely specified set_clock_uncertainty constraints
Clk_Uncert02c	Zero-value set_clock_uncertainty constraints
Clk_Uncert03	Synchronous clocks that do not have inter-clock uncertainty specified
Clk_Uncert04	set_clock_uncertainty constraints in postlayout phase that have the same or higher uncertainty values than the corresponding constraints in the prelayout phase
Clk_Uncert05	Incompletely specified set_clock_uncertainty constraints
Clk_Uncert06	set_clock_uncertainty constraints with negative values set
Clk_Uncert07	set_clock_uncertainty constraints specifying inter-clock uncertainty when no crossing exists between the two clocks
Clk_Uncert08	Clocks and their virtual clocks without inter-clock uncertainty defined between them when the virtual clock is used in a set_input_delay / set_output_delay constraint
Clk_Uncert09	set_clock_uncertainty constraints where the setup value is less than or equal to the hold value
Clk_Uncert10	set_clock_uncertainty constraints where the inter-clock uncertainty values between a pair of clocks are different
Clk_Uncert11	set_clock_uncertainty constraints specifying inter-clock uncertainty where the setup value is not same as the hold value

Clk_Uncert01

Reports clock_uncertainty set on an object which is not a real or generated clock

When to Use

To identify clock uncertainty set on design objects in RTL, Pre-layout, Post-layout phases

Description

The *Clk_Uncert01* rule reports *set_clock_uncertainty* constraints set on objects that are not real or generated clocks. In addition, the *Clk_Uncert01* rule reports *set_clock_uncertainty* constraints set on design objects where the specified clock has no corresponding *create_clock* or *create_generated_clock* constraints.

The rule behavior is controlled by the settings of the *pt* and *tc_allow_design_obj* parameters. Refer to the *Parameter(s)* section for more details.

Parameter(s)

<i>pt</i>	<i>tc_allow_design_obj</i>	Clk_Uncert01 Behavior
yes	yes	Reports <i>set_clock_uncertainty</i> constraints set on all design objects whose transitive fan-out does not lead to the clock pin of sequential elements.
no	yes	Reports <i>set_clock_uncertainty</i> constraints set on design objects for which no real clock is specified. You can specify real clocks by using the <i>create_clock</i> or <i>create_generated_clock</i> constraints.
yes	no	Reports <i>set_clock_uncertainty</i> constraints for all design objects.

Constraint(s)

- *set_clock_uncertainty* (Mandatory): Use to specify the uncertainty, or skew, of clock networks.

- *create_clock* (Optional): Use to create a clock
- *create_generated_clock* (Optional): Use to create a clock

Messages and Suggested Fix

Message 1

The following message appears for the object *<name>* used as a clock in a *set_clock_uncertainty* constraint, but has not been defined as a real clock using the *create_clock* constraint or as a generated clock using the *create_generated_clock* constraint:

[WARNING] Clock uncertainty set on object "<name>" that has not been specified as real or generated clock

Potential Issues

The violation message appears because the *pt* parameter is set to no. In addition, since there is not any clock declared on the object, the clock uncertainty value is unnecessary and could create confusion.

Consequences of Not Fixing

Clock uncertainty has to be set on clocks. Setting clock uncertainty on objects other than clocks indicates mismatch between the intent and actual specification.

How to Debug and Fix

Declare the uncertainty constraint on an object that has a real or generated clock.

Message 2

The following message appears for a design object *<name>* used as a clock in a *set_clock_uncertainty* constraint, but the transitive fan-out of the object does not lead to the clock pin of a sequential element and the *pt* parameter is set to yes:

[WARNING] Clock uncertainty is set on port/pin "<name>" which does not lead to a clock pin of sequential element

Potential Issues

The violation message appears because the *pt* parameter is set to yes. In addition, since there is not any clock pin in the transitive fan-out of the object, the clock uncertainty value is unnecessary and could create

confusion.

Consequences of Not Fixing

Clock uncertainty has to be set on clocks. Setting clock uncertainty on objects other than clocks indicates mismatch between the intent and actual specification.

How to Debug and Fix

Declare the uncertainty constraint on an object having transitive fan-out to the clock pin of a sequential cell.

Message 3

The following message appears if the [set_clock_uncertainty](#) constraint is set on an invalid design object `<name>`:

[WARNING] Clock uncertainty set on invalid design object `<name>`
(Design object is invalid because some back end tools do not support clock latency on port/pin)

Potential Issues

The violation message is displayed because the [tc_allow_design_obj](#) parameter is set to `no`. Implementation tools do not support clock uncertainty on design objects. Therefore, these constraints could be ignored leading to incorrect timing analysis.

Consequences of Not Fixing

Clock uncertainty has to be set on clocks. Setting clock uncertainty on objects other than clocks indicates mismatch between the intent and actual specification.

How to Debug and Fix

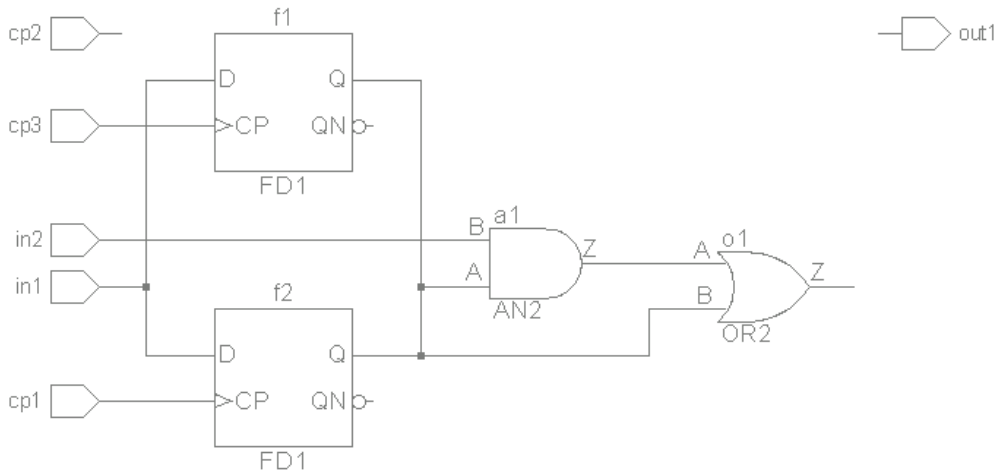
Declare the uncertainty constraint on a valid design object.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports [Message 3](#). The schematic of the top module is as follows:



The top.sdc is as follows:

```

top.sdc | top.sdc |
create_clock -period 10.000000 -waveform { 0.000000 5.000000 } [get_ports c*]
set_clock_latency 10 [all_inputs]
set_clock_transition 10 [all_inputs]
set_clock_uncertainty 10 [all_inputs]

```

This rule reports a violation message because a [set_clock_uncertainty](#) constraint is defined on the invalid port in1.

Example 2

Test Case Files Not Available

In the following SDC file snippet, [set_clock_uncertainty](#) is set on an object where the transitive fan-out does not lead to the clock pin of a sequential cell. Therefore the *Clk_Uncert01* reports a violation.

```
set_clock_uncertainty 2 -from [get_pins TOP/MODULE/OBJECT/
```

```
OUT1] -to [get_clocks CLK1]
```

Example 3

 Test Case Files Not Available

In the following SDC file snippet, *set_clock_uncertainty* on an object is not specified as a clock using *create_clock* or *create_generated_clock*, but is used as a clock inside the design. Therefore, the *Clk_Uncert01* reports a violation.

```
set_clock_uncertainty 2 -from [get_ports CKJN1]
```

Example 4

 Test Case Files Not Available

In the following SDC file snippet, *set_clock_uncertainty* is defined on a non-clock object. Therefore, the *Clk_Uncert01* reports a violation.

```
set_clock_uncertainty 2 -from [get_ports IN1]
```

Default Severity Label

Warning

Rule Group

Clk_Uncert

Reports and Related Files

None

Clk_Uncert02a

Reports clock uncertainty that has not been defined for a real clock

When to Use

To ensure clock uncertainty declaration for clocks in the RTL, Pre-layout, Post-layout phases

Description

The *Clk_Uncert02a* rule reports missing simple [set_clock_uncertainty](#) constraints for real clocks. It reports clocks that do not have the [set_clock_uncertainty](#) constraint specified. The clocks are specified using the [create_clock](#) or [create_generated_clock](#) constraint,

The *Clk_Uncert02a* rule checks for multiple clocks on the same design object. In case of multiple clocks on the same design object, the *Clk_Uncert02a* rule does not check for all overridden clocks.

Rule Exceptions

The *Clk_Uncert02a* rule does not check for virtual clocks or clocks specified with the `-to` option of the [set_false_path](#) command.

This rule does not report inter-clock uncertainty.

Parameter(s)

None

Constraint(s)

SDC

- [set_clock_uncertainty](#) (Mandatory): Use to specify the uncertainty (skew) of clock networks.
- [create_clock](#) (Optional): Used to create a clock
- [create_generated_clock](#) (Optional): Used to create a clock

Messages and Suggested Fix

The following message appears for a clock `<clk-name>` without the corresponding [set_clock_uncertainty](#) constraint:

[WARNING] Clock Uncertainty for clock "<clk-name>" has not been set

Potential Issues

Tools assume optimistic skew timing for the clock, due to which timing analysis gives no violation in pre-layout. However, it will lead to timing failure of the path in post-layout.

Consequences of Not Fixing

When clock uncertainty is not defined for a real clock, it can result in a lack of margin to allow for clock skew between the different path lengths. Even if pre-layout succeeds, post-layout timing may fail.

How to Debug and Fix

The *Clk_Uncert02a* rule reports real and generated clocks that are not constrained through the [set_clock_uncertainty](#) constraint. Overwritten clocks and the clocks specified in the to list of a [set_false_path](#) command having no from and through list are ignored.

Constrain all real clocks using the [set_clock_uncertainty](#) constraints.

Example Code and/or Schematic

In the following SDC file snippet, there is [create_clock](#) is defined, but [set_clock_uncertainty](#) is not defined. Therefore, the *Clk_Uncert02a* rule reports a violation.

```
create_clock -name CLK -period 10 [get_ports {CLK}]
```

Default Severity Label

Warning

Rule Group

Clk_Uncert

Reports and Related Files

None

Clk_Uncert02b

Reports clock uncertainty that do have -setup or -hold

When to Use

To ensure clock uncertainty declaration for clocks in the RTL, Pre-layout, Post-layout phases.

Description

The *Clk_Uncert02b* rule reports real clocks for which the [set_clock_uncertainty](#) constraint has been specified with the `setup` argument only or the `hold` argument only.

The *Clk_Uncert02b* rule checks [set_clock_uncertainty](#) constraints that are specifying individual clock uncertainty values and those specifying the inter-clock uncertainty values.

Rule Exceptions

The *Clk_Uncert02b* rule ignores the following:

- Clocks that are specified with the `to` argument of a [set_false_path](#) specification that do not have the `from/through` arguments specified as in the following example:

```
set_false_path -to [get_clock CLK1]
```

However, the *Clk_Uncert02b* rule reports a violation when the above type of [set_false_path](#) or [set_clock_groups](#) specification is specified with the `setup` argument only or the `hold` argument only as in the following example:

```
set_false_path -to [get_clock CLK1] -setup
```

- Inter-clock uncertainty values for clocks that are specified together in a [set_false_path](#) or [set_clock_groups](#) specification

```
set_clock_uncertainty -setup 10 -from C1 -to C2
```

```
set_false_path -from C1 -to C2
```

```
set_clock_uncertainty -setup 10 -from C1 -to C2
```

```
set_clock_groups -async -group C1 -group c2
```

Here, the [set_clock_uncertainty](#) constraint with the `hold` argument is not

available. The *Clk_Uncert02b* rule ignores such cases.

- Overridden clocks

Parameter(s)

- *tc_setup_hold*: Default is both. Set the value to *setup* or *hold* to restrict the checking for missing *setup* or *hold* arguments.

Constraint(s)

SDC

- *set_clock_uncertainty* (Mandatory): Use to specify the uncertainty of clock networks. This constraint defines the uncertainty in the clock skew value when it reaches to the leaf (input of flip-flop) of tree. In PrimeTime, the *set_clock_uncertainty* constraint could be specified for *setup* and *hold* separately.
- *create_clock* (Optional): Used to create a clock.
- *create_generated_clock* (Optional): Used to create a clock.
- *set_clock_groups* (Optional): Use to establish exclusive or asynchronous relationship with clocks of another group.

Messages and Suggested Fix

Message 1

The following message appears for a clock *<clk-name>* for which the *set_clock_uncertainty* constraint has been explicitly specified only with the *<option>* argument:

[WARNING] Clock uncertainty has not been specified with "*<option>*" option for clock "*<clk-name>*"

Where, *<option>* can be *hold* or *setup*.

Message 2

The following message appears for a clock *<clk1-name>* when the inter-clock uncertainty value with the clock *<clk2-name>* has not been specified with the *<option>* argument:

[WARNING] Inter-clock uncertainty from clock "*<clk1-name>*" to clock "*<clk2-name>*" has not been specified with "*<option>*"

option

Where, *<option>* can be hold or setup.

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

When clock uncertainty is not defined properly for a real clock, it can result in a lack of margin to allow for clock skew between the different path lengths. Even if pre-layout succeeds, post-layout timing may fail.

How to Debug and Fix

Specify both the hold and setup argument for the [set_clock_uncertainty](#) constraint highlighted in the SDC file.

Example Code and/or Schematic

In the following snippet, the [set_clock_uncertainty](#) constraint is applied with the setup argument only. Therefore, the *Clk_Uncert02b* rule reports a violation message.

```
set_clock_uncertainty 1.0 -setup clk1
```

Default Severity Label

Warning

Rule Group

Clk_Uncert

Reports and Related Files

None

Clk_Uncert02c

Reports clock uncertainty defined with zero value

When to Use

To ensure clock uncertainty declaration for clocks in the RTL, Pre-layout, Post-layout phases

Description

The *Clk_Uncert02c* rule reports real clocks for which the [set_clock_uncertainty](#) constraint has been specified with uncertainty value 0 for either setup or hold.

The *Clk_Uncert02c* rule checks [set_clock_uncertainty](#) constraints that are specifying individual clock uncertainty values and those specifying the inter-clock uncertainty values.

Rule Exceptions

The Clk_Uncert02c rule ignores the following:

- Clocks that are specified with the `to` option of a [set_false_path](#) specification that does not have the `from/through` arguments specified as in the following example:


```
set_false_path -to [get_clock CLK1]
```
- Inter-clock uncertainty values for clocks that are specified together in a [set_false_path](#) or [set_clock_groups](#) specification


```
set_clock_uncertainty -setup 0 -from C1 -to C2
set_false_path -from C1 -to C2
set_clock_uncertainty -setup10 -from C1 -to C2
set_clock_groups -async -group C1 -group c2
```

 Here, the [set_clock_uncertainty](#) constraint has been specified with zero-value setup argument.
- Overridden clocks: Only for [set_clock_uncertainty](#) constraints that are specifying individual clock uncertainty values.

Parameter(s)

- *tc_setup_hold*: Default is both. Set the value to setup or hold to restrict the checking for missing setup or hold arguments.
- *tc_waive*: Default is none. Set the value to zero_hold to ignore *set_clock_uncertainty* constraints with the hold argument specified with zero value.

Constraint(s)

SDC

- *set_clock_uncertainty* (Mandatory): Use to specify the uncertainty (skew) of clock networks. This constraint defines the uncertainty in the clock latency value when it reaches to the leaf (input of flip-flop) of tree. In PrimeTime, the *set_clock_uncertainty* constraint could be specified for setup and hold separately.
- *create_clock* (Optional): Used to create a clock.
- *create_generated_clock* (Optional): Used to create a clock.

Messages and Suggested Fix

Message 1

The following message appears for a clock *<clk-name>* (specified using the *create_clock* or *create_generated_clock* constraint) having a corresponding *set_clock_uncertainty* constraint (in SDC file *<file-name>* at line *<num>*) with uncertainty value of 0 for either setup or hold:

[WARNING] Clock Uncertainty value for clock "<clk-name>" is zero

Message 2

The following message appears for a source clock *<clks-name>* (specified in SDC file *<files-name>* at line *<nums>*) when the inter-clock uncertainty value with the destination clock *<clkd-name>* (specified in SDC file *<filed-name>* at line *<numd>*) is zero:

[WARNING] Inter clock uncertainty value between source clock "<clks-name>" and destination clock "<clkd-name>" is zero

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

When clock uncertainty is defined with zero value for real clock, it can result in a lack of margin to allow for clock skew between different path lengths. The post-layout timing may fail even if pre-layout was successful.

How to Debug and Fix

In the SDC file, specify the clock uncertainty with the real value.

Example Code and/or Schematic

For the following snippet, the *Clk_Uncert02c* rule reports a violation because inter-clock uncertainty is set to zero.

```
//test.sdc
set_clock_uncertainty 0 -from clk1 -to clk2
```

Default Severity Label

Warning

Rule Group

Clk_Uncert

Reports and Related Files

None

Clk_Uncert03

Reports inter-clock uncertainty that has not been defined between synchronous clocks

When to Use

RTL phase, Pre-layout phase, Post-layout phase

Description

The *Clk_Uncert03* rule flags synchronous clocks that do not have interclock uncertainty specified. Clocks are synchronous if they have been defined in the same *clock_group*. The *Clk_Uncert03* rule processes the *clock_group* constraint, if specified.

The rule behavior is controlled by the *tc_clk_compat* parameter. Refer to the *Parameter(s)* section for more details.

Prerequisites

Run the *Clk_Uncert03* rule only after specifying the *clock_group* information in the SGDC file. If the *clock_group* information is not specified in the SGDC file, this rule does not report a violation. Refer to *Use Model for Inferring Domain* for details.

Rule Exceptions

Clock propagation is stopped if the *set_clock_sense/set_sense* constraint with the *-stop_propagation* option is applied on the path of a clock. For details, refer to the *Stopping Clock Propagation* section. If a clock is not propagated, it affects the clock crossings. For all clocks from the same domain and interacting through crossings, specify inter-clock uncertainty.

Parameter(s)

- *tc_clk_compat*: Default is no. This means that the *Clk_Uncert03* rule does not consider it to be a clock domain interaction when a clock reaches the data pin of a flip-flop triggered by another clock. Set the value to *yes* to consider such cases to be of clock domain interaction.

Constraint(s)

SDC

- *set_clock_sense/set_sense* (Optional): Use to specify the clock propagation conditions.

SGDC

- *clock_group*: Use to define domain relation between clocks.

Messages and Suggested Fix

Message 1

The following message appears for a clock *<clk1-name>* where the clock uncertainty is not specified for its synchronous clock *<clk2-name>*:

```
[Clk_Uncert03_01][WARNING] Clock uncertainty not defined
between synchronous clocks "<clk1-name>" and "<clk2-name>"
```

Potential Issues

If there are two clocks, the uncertainty between them could be different from the uncertainty on either of them. The uncertainty is not specified.

Consequences of Not Fixing

The uncertainty value used by the tool is the uncertainty defined for the target clock. The uncertainty of the target clock could be different from the relative uncertainty between the two clocks. This could lead to incorrect timing analysis.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic highlights crossing path between the two clocks. The *Clk_Uncert03* rule also highlights the clock constraints and inter-clock uncertainty constraint in the SDC file.

To resolve this violation, specify clock uncertainty between the two clocks.

Message 2

The following message appears if the *clock_group* information is not specified in the SGDC file for design *<name>*:

```
[Clk_Uncert03_02][WARNING] Rule will not violate for design
<name> since clock_group information is not specified in SGDC
file
```

Potential Issues

Two clocks from the same domain are considered synchronous. Therefore, the rule needs the *clock_group* SGDC constraints.

Consequences of Not Fixing

Since the *clock_group* information is not available, the rule does not report potential interacting synchronous clock pairs. This could potentially lead to timing failure of the design.

How to Debug and Fix

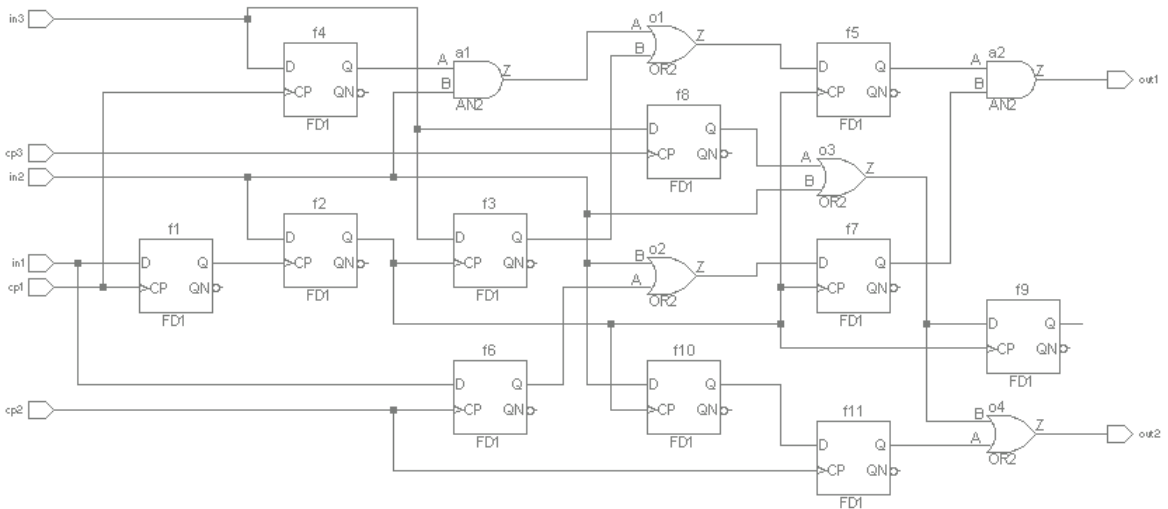
Without the *clock_group* information, it is not possible to determine if a clock pair is synchronous or asynchronous. Specify the *clock_group* information in the SGDC file because this information is used to classify the clocks into domains.

Example Code and/or Schematic

Example 1[View Test Case Files](#)

This example shows when this rule reports *Message 1*. The schematic of the top module is as follows:

Clock Rules



The test .sgdc is as follows:

```

test.sgdc x
1  current_design top
2  sdcschema -type top.sdc -level rtl -mode func -corner Worst
3  block -name top
4  blocksize -min 0 -max 100
5
6  domain -name d1 -clock {clk1 gclk2}
7

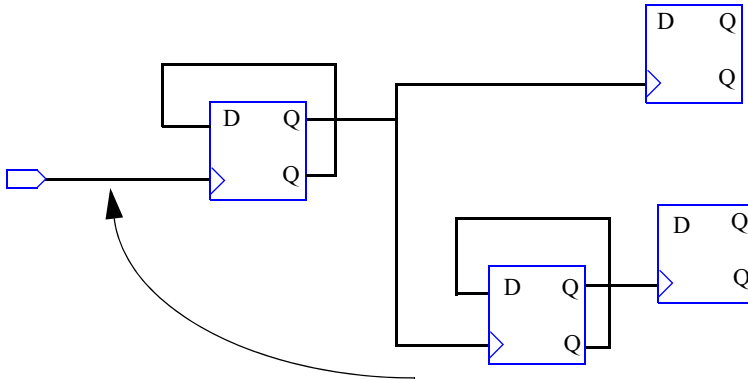
```

This rule reports a violation message because a [set_clock_uncertainty](#) constraint is not defined between the synchronous clocks `clk1` and `gclk2`. In the `test.sgdc` file, the clocks `clk1` and `gclk2` are defined to be in the same domain.

Example 2

Test Case Files Not Available

In the following example, *set_clock_uncertainty* has not been specified on synchronous clocks. Therefore, the *Clk_Uncert03* rule reports a violation.



Inter clock uncertainty should be defined between synchronous clocks

Default Severity Label

Warning

Rule Group

Clk_Uncert

Reports and Related Files

- [sync_clock_uncertainty Report](#)

Clk_Uncert04

Reports clock uncertainty values that are higher in the post-layout than in the pre-layout

When to Use

This rule ensures the consistency of uncertainty value defined in Pre-layout and Post-layout phase.

Description

The *Clk_Uncert04* rule reports *set_clock_uncertainty* constraints in the post-layout phase that have the same or higher uncertainty values than the corresponding constraints in the pre-layout phase. You can specify the design phases, such as pre- and post-layout, by using the `level` argument of the *sdc_data* constraint.

Rule Exceptions

The *Clk_Uncert04* rule checks for real and generated clocks only and does not check for virtual clocks.

Parameter(s)

None

Constraint(s)

SDC

- *set_clock_uncertainty* (Mandatory): Use to specify the uncertainty (skew) of clock networks. This constraint defines the uncertainty in the clock latency value when it reaches to the leaf (input of flip-flop) of tree. In PrimeTime, the *set_clock_uncertainty* constraint could be specified for `setup` and `hold` separately.
- *create_clock* (Optional): Used to create a clock
- *create_generated_clock* (Optional): Used to create a clock

SGDC

- *sdc_data*: Use to specify the SDC file.

Messages and Suggested Fix

Message 1

The following message appears when the postlayout level inter-clock or clock uncertainty value `<value1>` is greater than the prelayout level inter-clock or clock uncertainty value `<value2>` for the design unit `<du-name>`:

```
Postlayout <type> uncertainty value (<value1>) is greater than
the prelayout <type> uncertainty value (<value2>) for design
unit '<du-name>'
```

Where, `<type>` is inter-clock or clock.

Message 2

The following message appears when the postlayout level inter-clock or clock uncertainty value `<value1>` is same as the prelayout level inter-clock or clock uncertainty value `<value2>` for the design unit `<du-name>`:

```
Postlayout <type> uncertainty value (<value1>) is same as the
prelayout <type> uncertainty value (<value2>) for design unit
'<du-name>'
```

Where, `<type>` is inter-clock or clock.

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

Pre-layout timing may not correlate well with post-layout timing.

How to Debug and Fix

Review the SDC file. To resolve the violation messages, specify a pre-layout uncertainty value that is greater than the post-layout value.

Example Code and/or Schematic

In the following snippet, inter-clock uncertainty, from clock `clk1` to `clk3`, in pre-layout is less than in post-layout. Therefore, the `Clk_Uncert04` rule reports a violation message.

```
//prelayout.sdc
```

Clock Rules

```
set_clock_uncertainty 13 -rise -hold -from {clk1 clk2 clk4}
-to {clk3 clk4}
//postlayout.sdc
set_clock_uncertainty 20 -rise -hold -from {clk1 clk3} -to
clk3
```

Default Severity Label

Warning

Rule Group

Clk_Uncert

Reports and Related Files

None

Clk_Uncert05

Reports `set_clock_uncertainty` that has been specified partially

When to Use

To ensure clock uncertainty declaration for clocks in the RTL, Pre-layout, Post-layout phases.

Description

The *Clk_Uncert05* rule checks whether the `set_clock_uncertainty` constraint with only one of the `rise` and `fall` arguments for any real clock. When the `set_clock_uncertainty` constraint is specified with the `setup` or `hold` arguments, the *Clk_Uncert05* rule checks whether matching combinations of the `rise` and `fall` arguments and the `setup` or `hold` arguments have been specified.

PrimeTime Specific Behavior

In PrimeTime, the `from_edge` or `to_edge` options are used in place of the `rise` or `fall` for specifying clock uncertainties with respect to different edges of source and destination clocks. The *Clk_Uncert05* rule checks for each source-destination clock pair assuming that if any one of `from_edge` or `to_edge` options is used, both edges of each clock are specified.

The *Clk_Uncert05* rule also supports the new `fall_from/rise_from` and `fall_to/rise_to` PrimeTime options.

Parameter(s)

- `tc_setup_hold`: Default is both. Set the value to `setup` or `hold` to restrict the checking for missing `setup` or `hold` arguments.

Constraint(s)

SDC

- `set_clock_uncertainty` (Mandatory): Use to specify the uncertainty (skew) of clock networks. This constraint defines the uncertainty in the clock latency value when it reaches to the leaf (input of flip-flop) of tree. In

PrimeTime, the *set_clock_uncertainty* constraint could be specified for setup and hold separately.

- *create_clock* (Optional): Used to create a clock
- *create_generated_clock* (Optional): Used to create a clock

Messages and Suggested Fix

Message 1

The following message appears for clocks *<clk1-name>* and *<clk2-name>* in design/block *<name>* for which the inter-clock *set_clock_uncertainty* constraint has been specified incompletely:

[WARNING] Inter clock uncertainty from clock "*<clk1-name>*" to clock "*<clk2-name>*" is specified incompletely for design/block "*<name>*" (Specified: [*<option-list1>*] Not specified: [*<option-list2>*])

Where *<option-list1>* is the option list of specified options and *<option-list2>* is the option list of missing options.

Message 2

The following message appears for clocks *<clk1-name>* and *<clk2-name>* in design/block *<name>* for which the inter-clock *set_clock_uncertainty* constraint has not been explicitly specified for options *<option-list>*:

[WARNING] Inter clock uncertainty from clock "*<clk1-name>*" to clock "*<clk2-name>*" has not been set for "[*<option-list>*]" in design/block "*<name>*"

Where, *<option-list>* can be a combination of the following options:

rising edge-> falling edge	rising edge-> rising edge
falling edge-> falling edge	falling edge-> rising edge

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

Some EDA tools take default value for undefined value and hence timing

analysis may not match with actual.

How to Debug and Fix

In the SDC file, specify the missing options as stated in the violation messages.

Example Code and/or Schematic

Example 1

If `-setup` and `-rise` options have been specified for a clock, the matching combination of `-setup` and `-fall` options must also be specified for the clock. This example illustrates complete and incomplete clock uncertainty specifications.

The following snippet shows a complete clock uncertainty specification:

```
set_clock_uncertainty 1.0 [get_clocks clk1]
```

Similarly, the following snippet also shows a complete clock uncertainty specification:

```
set_clock_uncertainty 1.0 -setup [get_clocks clk1]
```

```
set_clock_uncertainty 1.0 -hold [get_clocks clk1]
```

In the first line, clock uncertainty is defined for `setup`, `rise`, and `fall` arguments. In the second line, clock uncertainty is defined for `hold`, `rise`, and `fall` arguments.

In the below example, clock uncertainty is said incomplete because only `setup`, `rise`, and `fall` arguments have been defined.

```
set_clock_uncertainty 1.0 -setup [get_clocks clk1]
```

Example 2

For the following snippet, clock uncertainty is defined for `setup`, `rise`, and `fall` arguments. The clock uncertainty defined is incomplete. Therefore, the `Clk_Uncert05` rule reports a violation for the arguments that have not been specified: `hold`, `rise`, and `fall`.

```
set_clock_uncertainty 1.0 -setup [get_clocks clk1]
```

Default Severity Label

Warning

Clock Rules

Rule Group

Clk_Uncert

Reports and Related Files

None

Clk_Uncert06

Reports `set_clock_uncertainty` that is set to a negative value

When to Use

RTL phase, Pre-layout phase, Post-layout phase

Description

The *Clk_Uncert06* rule reports `set_clock_uncertainty` constraints that have negative values set.

Rule Exceptions

The *Clk_Uncert02a* rule does not check for virtual clocks or clocks specified with the `-to` option of the `set_false_path` constraint.

Parameter(s)

None

Constraint(s)

SDC

- `set_clock_uncertainty` (Mandatory): Use to specify the uncertainty (skew) of clock networks.

Messages and Suggested Fix

The following message appears for a `set_clock_uncertainty` constraint that is specified with a negative value:

```
[INFO] set_clock_uncertainty is set to negative value
```

Potential Issues

Negative skew value indicates optimism in the margin available for the clock paths.

Consequences of Not Fixing

Designs cleaned through various Static Timing Analysis type analysis might not actually work.

How to Debug and Fix

Check the value of the `set_clock_uncertainty` constraint, which is highlighted

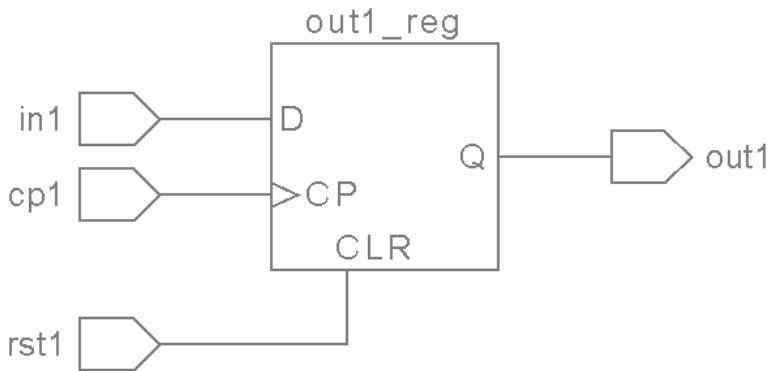
in the SDC file. To fix this violation, specify a positive value.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top.sdc is as follows:

```
top.sdc top.sdc
create_clock cp1 -name clk1 -period 10 -waveform {0 5}
set_clock_uncertainty 0 [get_clocks clk1]
set_clock_uncertainty -2 [get_clocks clk1]
set_clock_uncertainty 5 [get_clocks clk1]
```

This rule reports a violation message because the `set_clock_uncertainty` is set to a negative value.

Example 2

Test Case Files Not Available

In the following example, `set_clock_uncertainty` is set to a negative value. Therefore, the `Clk_Uncert06` rule reports a violation.

```
set_clock_uncertainty -5.23 -from [get_clocks CLK1] -to  
[get_clocks CLK2]
```

Default Severity Label

Info

Rule Group

Clk_Uncert

Reports and Related Files

None

Clk_Uncert07

Reports instances when no crossing exists between clocks for which inter-clock uncertainty is defined

When to Use

RTL phase, Pre-layout phase, Post-layout phase

Description

The *Clk_Uncert07* rule highlights [set_clock_uncertainty](#) constraints specified between two clocks. The clocks are not interacting between each other, therefore no crossing exists between them.

Rule Exceptions

Clock propagation is stopped if the [set_clock_sense/set_sense](#) constraint with the `-stop_propagation` option is applied on the path of a clock. For details, refer to the [Stopping Clock Propagation](#) section. If clock is not propagated, it affects the clock crossings.

Parameter(s)

None

Constraint(s)

SDC

- [set_clock_uncertainty](#) (Mandatory): Use to specify the uncertainty (skew) of clock networks
- [create_clock](#) (Optional): Use to create a clock
- [create_generated_clock](#) (Optional): Use to create a clock
- [set_clock_sense/set_sense](#) (Optional): Use to specify the clock propagation conditions.

Messages and Suggested Fix

The following message appears for a [set_clock_uncertainty](#) constraint that specifies inter-clock uncertainty between clocks `<clk1-name>` and `<clk2-name>` of the design/block `<name>` when no crossing exists between the two clocks:

[INFO] Inter-clock uncertainty specified when no crossing exists from clock "<clock1-name>" to clock "<clock2-name>" in design/block "<name>"

Potential Issues

Since there is no path between the clocks, the inter-clock uncertainty constraint does not apply to the design. It is unnecessary and might create confusion.

Consequences of Not Fixing

The uncertainty specification is not required. It is ignored since there is no design scenario where the crossing exists between the specified clocks.

How to Debug and Fix

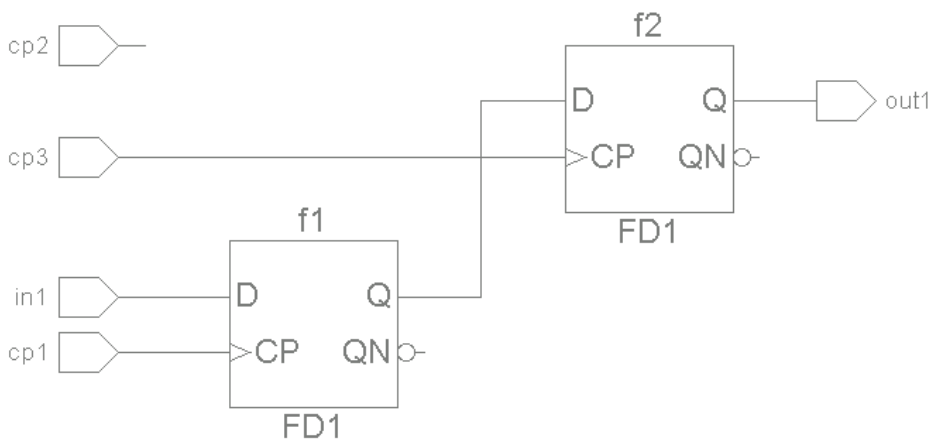
The clock constraint and inter-clock constraints are highlighted in the SDC file. Review the uncertainty specification since there is no design scenario to which it is applicable.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



Clock Rules

The top.sdc is as follows:

```
top.sdc | top.sdc (3)
create_clock -name clk1 -period 10.000000 -waveform {0.000000 5.000000} {cp1}
create_clock -name clk2 -period 10.000000 -waveform {0.000000 5.000000} {cp2}
create_clock -name clk3 -period 10.000000 -waveform {0.000000 5.000000} {cp3}
create_generated_clock -name gclk1 -source cp1 [get_pins f1/Q] -divide_by 2

set_clock_uncertainty 6.95 -from clk1 -to clk2
set_clock_uncertainty 6.95 -from [get_clocks clk* ] -to [get_clocks clk*]
set_clock_uncertainty 6.95 -from clk1 -to clk3
```

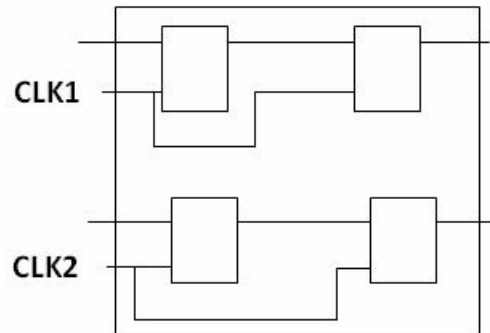
This rule reports a violation message because the [set_clock_uncertainty](#) constraint is specified when there is no crossing between clocks clk1 and clk3.

Example 2

 Test Case Files Not Available

In the following example, [set_clock_uncertainty](#) is specified between clocks that do not have any crossings. Therefore, the *Clk_Uncert07* rule reports a violation.

```
create_clock -name CLK1 -period 10 [get_ports {CLK1}]
create_clock -name CLK2 -period 20 [get_ports {CLK2}]
set_clock_uncertainty 2 -from [get_clocks CLK1] -to
[get_clocks CLK2]
```

**Default Severity Label**

Info

Rule Group

Clk_Uncert

Reports and Related Files

None

Clk_Uncert08

Reports inter-clock uncertainty that is not defined between a clock and its virtual clock

When to Use

To ensure clock uncertainty declaration for clocks in the RTL, Pre-layout, Post-layout phases.

Description

The *Clk_Uncert08* rule reports clocks and their virtual clocks, which is used in a *set_input_delay/set_output_delay* constraint, when the inter-clock uncertainty is not defined between them.

The *Clk_Uncert08* rule requires the following:

- *set_input_delay* specification on an input port, specify the inter-clock uncertainty between the virtual clock and the clock sampling the sequential element driven by the input port.
- *set_output_delay* specification for an output port, specify the inter-clock uncertainty between the virtual clock and the clock sampling a sequential element whose fan-out contains the output port.

Parameter(s)

None

Constraint(s)

SDC

- *set_clock_uncertainty* (Mandatory): Use to specify the uncertainty (skew) of clock networks. This constraint defines the uncertainty in the clock latency value when it reaches to the leaf (input of flip-flop) of tree. In PrimeTime, the *set_clock_uncertainty* constraint could be specified for setup and hold separately.
- *create_clock* (Optional): Used to create a clock
- *create_generated_clock* (Optional): Used to create a clock
- *set_input_delay* (Optional): Use to set input delay on pins or output ports.

- *set_output_delay* (Optional): Use to set output delay on pins or output ports.

Messages and Suggested Fix

Message 1

The following message appears for the clock *<clkname>* when its virtual clock *<vclk-name>* is used in a *set_output_delay* specification but inter-clock uncertainty has not been defined between the clock and its virtual clock:

[WARNING] Inter-clock uncertainty not defined between clock "*<clkname>*" and it's virtual clock "*<vclk-name>*" (file: *<filename>*, line: *<num>*)

Message 2

The following message appears for virtual clock *<vclk-name>* that is used in a *set_input_delay* specification but inter-clock uncertainty has not been defined with its clock *<clk-name>*:

[WARNING] Inter-clock uncertainty not defined between virtual clock "*<vclk-name>*" and it's clock "*<clk-name>*" (file: *<filename>*, line: *<num>*)

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

Inter-clock uncertainty comes into play when there is a crossing between the two clocks. If this is not specified, skew assumptions between the clocks could be incorrect. This leads to incorrect timing.

How to Debug and Fix

In the SDC file, specify the missing arguments as stated in the violation messages.

Example Code and/or Schematic

For the following snippet, the *Clk_Uncert08* rule reports a violation because the uncertainty is not defined between clocks *c1* and *vc1*.

```
//test.sdc
```

Clock Rules

```
create_clock -name c1 -period 20 CLK
```

```
create_clock -name vc1 -period 20
```

However, if you specify the uncertainty as shown in the following snippet, this rule does not report a violation.

```
set_clock_uncertainty 1.0 -from c1 -to vc1
```

Default Severity Label

Warning

Rule Group

Clk_Uncert

Reports and Related Files

None

Clk_Uncert09

Reports clock uncertainty when the hold value is greater than the setup value

When to Use

To ensure clock uncertainty declaration for clocks in the RTL, Pre-layout, Post-layout phases.

Description

The *Clk_Uncert09* rule reports *set_clock_uncertainty* constraints where the setup value is less than or equal to the hold value.

Parameter(s)

None

Constraint(s)

SDC

- *set_clock_uncertainty* (Mandatory): Use to specify the uncertainty (skew) of clock networks. This constraint defines the uncertainty in the clock latency value when it reaches to the leaf (input of flip-flop) of tree. In PrimeTime, the *set_clock_uncertainty* constraint could be specified for setup and hold separately.
- *create_clock* (Optional): Used to create a clock
- *create_generated_clock* (Optional): Used to create a clock

Messages and Suggested Fix

The following message appears for a *set_clock_uncertainty* constraint specifying clock uncertainty for the clock *<clkname>* when the setup value *<svalue>* is less than or equal to the hold value *<hvalue>*:

[WARNING] Clock uncertainty hold value *<hvalue>* for clock "*<clkname>*" should be less than the setup value *<svalue>* (file: *<file-name>* line: *<num>*)

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

The setup value should be greater than the hold value as clock jitter is also taken into account during setup analysis.

How to Debug and Fix

In the SDC file, specify a hold value that is less than the setup value.

Example Code and/or Schematic

For the following snippet, the *Clk_Uncert09* rule reports a violation because the hold value is greater than setup value.

```
//test.sdc  
set_clock_uncertainty 1.2 -hold clk1  
set_clock_uncertainty 0.9 -setup clk1
```

However, if you specify a hold value that is less than the setup value as shown in the following snippet, this rule does not report a violation.

```
set_clock_uncertainty .9 -hold clk1  
set_clock_uncertainty 1.2 -setup clk1
```

Default Severity Label

Warning

Rule Group

Clk_Uncert

Reports and Related Files

None

Clk_Uncert10

Reports inter-clock uncertainty values that are different for a pair of clocks

When to Use

To ensure clock uncertainty declaration for clocks in the RTL, Pre-layout, Post-layout phases.

Description

The *Clk_Uncert10* rule reports *set_clock_uncertainty* constraints where the inter-clock uncertainty values between a pair of clocks are different.

Rule Exceptions

The Clk_Uncert10 rule ignores clocks specified with the *from* and *to* arguments of the *set_false_path* constraint.

Parameter(s)

None

Constraint(s)

SDC

- *set_clock_uncertainty* (Mandatory): Use to specify the uncertainty (skew) of clock networks. This constraint defines the uncertainty in the clock latency value when it reaches to the leaf (input of flip-flop) of tree. In PrimeTime, the *set_clock_uncertainty* constraint could be specified for setup and hold separately.
- *create_clock* (Optional): Used to create a clock
- *create_generated_clock* (Optional): Used to create a clock
- *set_false_path* (Optional): Use to identify paths as false, therefore are ignored during timing analysis.

Messages and Suggested Fix

The following message appears for a *set_clock_uncertainty* constraint specifying inter-clock uncertainty from clock *<clk2-name>* to clock *<clk1-name>* when the value *<value2>* is not same as the inter-clock

uncertainty value `<value1>` from clock `<clk1-name>` to clock `<clk2-name>` specified earlier:

[WARNING] Inter-clock uncertainty values are not same (`<value2>` from `<clk2-name>` to `<clk1-name>`; `<value1>` from `<clk1-name>` to `<clk2-name>`)

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Tools can pick any of the values. This leads to inaccurate timing analysis.

How to Debug and Fix

In the SDC file, specify the same value for the clock pair.

Example Code and/or Schematic

The specified inter-clock uncertainty value from one clock, `clk1`, to another clock, `clk2`, should be same as the specified inter-clock uncertainty value from the second clock, `clk2`, to the first clock, `clk1`.

For the following snippet, the `Clk_Uncert10` rule reports a violation because `c1` to `c2` is different when compared with `c2` to `c1`.

```
//test.sdc
set_clock_uncertainty 1.0 -from c1 -to c2
set_clock_uncertainty .9 -from c2 -to c1
```

However, if you specify the uncertainty as shown in the following snippet, this rule does not report a violation because the values are the same.

```
set_clock_uncertainty 1.0 -from c1 -to c2
set_clock_uncertainty 1.0 -from c2 -to c1
```

Default Severity Label

Warning

Rule Group

Clk_Uncert

Reports and Related Files

None

Clk_Uncert11

Reports inter-clock uncertainty hold values that are greater than or equal to setup values

When to Use

To ensure clock uncertainty declaration for clocks in the RTL, Pre-layout, Post-layout phases.

Description

The *Clk_Uncert11* rule reports inter-clock uncertainty for clocks that have a hold value greater than or equal to the setup value.

Rule Exceptions

The *Clk_Uncert11* rule ignores *set_false_path* constraint specified on these clocks.

Parameter(s)

None

Constraint(s)

SDC

- *set_clock_uncertainty* (Mandatory): Use to specify the uncertainty (skew) of clock networks. This constraint defines the uncertainty in the clock latency value when it reaches to the leaf (input of flip-flop) of tree. In PrimeTime, the *set_clock_uncertainty* constraint could be specified for setup and hold separately.
- *create_clock* (Optional): Used to create a clock
- *create_generated_clock* (Optional): Used to create a clock
- *set_false_path* (Optional): Use to identify paths as false, therefore are ignored during timing analysis.

Messages and Suggested Fix

The following message appears for a *set_clock_uncertainty* constraint specifying inter-clock uncertainty from clock *<clk1-name>* to clock *<clk2-name>* when the setup value *<svalue>* is not greater than the

hold value *<hvalue>*:

[WARNING] Inter-clock uncertainty setup value *<svalue>* for clocks '*<clk1-name>*, *<clk2-name>*' is not greater than the hold value *<hvalue>*

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

When the setup value is lesser than the hold value, it leads to clock jitter.

How to Debug and Fix

Double-click the violation message. The SDC file appears. In this file, specify a setup value that is greater than the hold value for the clocks mentioned in the violation message.

Example Code and/or Schematic

For the following snippet, the *Clk_Uncert11* rule reports a violation because the setup value is not greater than the hold value.

```
//test.sdc
set_clock_uncertainty 0.8 -setup clk1
set_clock_uncertainty 0.9 -hold clk1
```

However, if the values are defined as shown in the following snippet, this rule does not report a violation message because the setup value is greater than the hold value.

```
//test.sdc
set_clock_uncertainty 1.0 -setup clk1
set_clock_uncertainty 0.9 -hold clk1
```

Default Severity Label

Warning

Rule Group

Clk_Uncert

Clock Rules

Reports and Related Files

None

Dont Touch Rules

The Dont Touch Rules Sub-group `Dont_Touch` contains the following rules:

Rule	Description
<i>Dont_Touch02</i>	Clocks without <code>set_dont_touch_network</code> constraint
<i>Dont_Touch03</i>	Non-clock objects or on-existent clocks with <code>set_dont_touch_network</code> constraint
<i>Dont_Touch04</i>	Asynchronous set or reset/clear signals without a <code>set_dont_touch_network</code> constraint or a <code>set_dont_touch</code> constraint
<i>Dont_Touch05</i>	<code>set_dont_touch</code> commands used in non-netlist-level designs

Dont_Touch02

Reports clocks that are not defined as `dont_touch_network`

When to Use

To ensure that clock network is not tampered with while synthesizing the RTL. Therefore, the design intentions are not lost.

Description

The *Dont_Touch02* rule reports clocks that do not have a corresponding `set_dont_touch_network` constraint. Clocks are specified using the *create_clock* or *create_generated_clock* constraint.

Parameter(s)

None

Constraint(s)

SDC

- *create_clock*: Use to create a clock.
- *create_generated_clock*: Use to create a clock.
- `set_dont_touch_network`: Use to set `dont_touch` attribute on the clock networks for synthesis.

Messages and Suggested Fix

The following message appears for the clock `<clk-name>` for the design/block `<name>` that has no corresponding `set_dont_touch_network` constraint:

Clock signal "`<clk-name>`" for design/block "`<name>`" has not been constrained using `set_dont_touch_network`

Potential Issues

The clock network is left open to the synthesis tool for undesirable optimizations.

Consequences of Not Fixing

In the absence of `dont_touch` attribute on clock networks, the synthesis

tool may perform optimizations in the clock network that could result in an unbalanced clock network and other issues related to skew and latency. This impacts timing closure at the back-end. In addition, the assumptions made at the front-end and back-end will not match.

How to Debug

All real clocks, primary or generated, should be constrained with the `set_dont_touch_network` constraint. The SDC file highlights the clock and the violation message states the clock name that is not constrained by the `set_dont_touch_network` constraint.

Example Code and/or Schematic

In the following code snippet, the `clk` clock is buffered through four inverters and `bclk` is clocking a flip-flop. The buffer/inverter chains are usually introduced into the clock path for clock balancing and clock signal boosting. If `set_dont_touch_network` is not set on `clk`, the synthesis tool may optimize the buffer chain.

```
input clk;
wire w1 = !clk;
wire w2 = !w1;
wire w3 = !w2;
wire bclk = !w3;
always@(posedge bclk)
    out <= in;
```

Default Severity Label

Warning

Rule Group

Dont_Touch

Reports and Related Files

None

Dont_Touch03

Reports non-clock net that is defined with `dont_touch_network`

When to Use

Use this rule in the RTL phase.

Description

The *Dont_Touch03* rule reports non-clock objects specified with the `set_dont_touch_network` or `set_dont_touch` constraint. This rule assumes that the objects specified with `create_clock` or `create_generated_clock` constraints are clocks and all other objects are non-clock objects.

Rule Exceptions

The *Dont_Touch03* rule ignores asynchronous set or reset nets.

Parameter(s)

None

Constraint(s)

SDC

- `set_dont_touch` (Mandatory): Use to specify a list of cells, nets, references, designs, and library cells on which the `dont_touch` attribute is to be set.

Messages and Suggested Fix

The following message appears for a non-clock object `<obj-name>` (not specified using `create_generated_clock` or `create_generated_clock` constraint) in design/block `<name>` that has a corresponding `set_dont_touch_network` constraint or `set_dont_touch` constraint:

```
[WARNING] <constr> has been set for a design object '<obj-name>' of design/block "<name>" which is used neither as a clock nor a reset
```

Where, `<constr>` can be `set_dont_touch_network` or `set_dont_touch`.

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

Specifying non clock nets with the `set_dont_touch_network` constraint can affect the results of re-optimization of a netlist.

How to Debug and Fix

Remove the `set_dont_touch_network` constraint from non-clock nets/ports/pins.

Example Code and/or Schematic

Assume, `in1` is feeding into the D pin of a flip-flop. For the following snippet, the `Dont_Touch03` rule reports a violation because `in1` is neither being used as a clock nor as a reset.

```
//test.sdc
set_dont_touch_network in1
```

Default Severity Label

Warning

Rule Group

Dont_Touch

Reports and Related Files

None

Dont_Touch04

Reports asynchronous reset/clear not marked with `set_dont_touch` or `set_dont_touch_network`

When to Use

RTL phase

Description

The *Dont_Touch04* rule reports signals that are used as asynchronous set/reset (clear) but have not been constrained using the `set_dont_touch_network` or the *set_dont_touch* constraints.

The *Dont_Touch04* rule considers signals that are going into the asynchronous terminal of a flip-flop as asynchronous set/reset (clear) signals.

Parameter(s)

None

Constraint(s)

SDC

- *set_dont_touch* (Mandatory): Use to specify a list of cells, nets, references, designs, and library cells on which the `dont_touch` attribute is to be set.

Messages and Suggested Fix

The following message appears for an asynchronous set/reset (clear) signal `<name>` that has no corresponding `set_dont_touch_network` or *set_dont_touch* constraint:

[WARNING] `set_dont_touch_network` (or `set_dont_touch`) has not been set for a design object '`<name>`' which is used either as an asynchronous set or clear

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

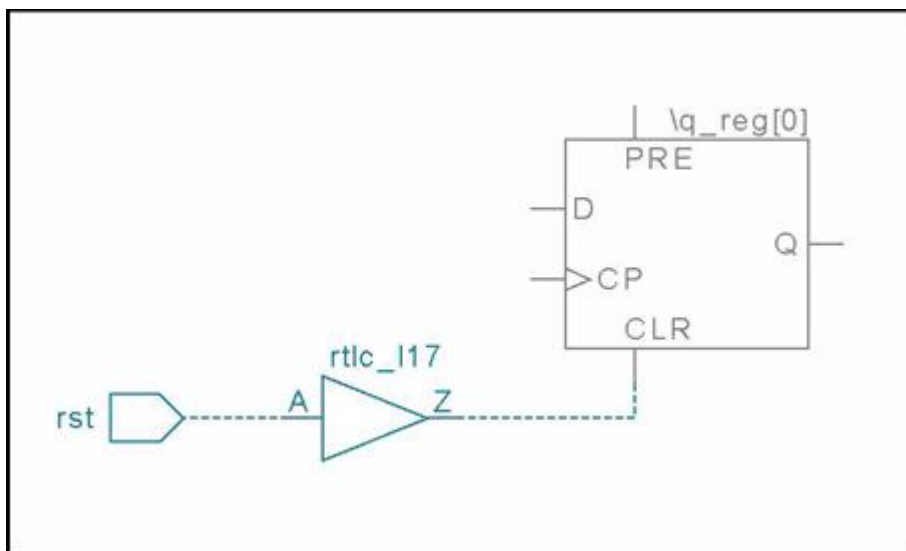
Signals used as asynchronous set/reset and not marked with the `set_dont_touch` or `set_dont_touch_network` constraints can affect the results of re-optimization of a netlist.

How to Debug and Fix

Apply `set_dont_touch`/`set_dont_touch_network` to port/pin reaching to clock pin of flip-flop.

Example Code and/or Schematic

For the following design, the `Dont_Touch04` rule reports a violation because the `set_dont_touch`/`set_dont_touch_network` is not applied on the net `rst`.



Default Severity Label

Warning

Rule Group

Dont_Touch

Clock Rules

Reports and Related Files

None

Dont_Touch05

Reports `dont_touch` attribute set on a non netlist design

When to Use

Use this rule in the RTL phase.

Description

The *Dont_Touch05* rule reports *set_dont_touch* constraints used in non-netlist-level (use RTL) designs.

The *set_dont_touch* constraint sets the `dont_touch` attribute on cells, nets, references, and designs in the current design and on library cells, to prevent these objects from being modified or replaced during optimization. However, the `dont_touch` attribute is valid on netlist-level designs only.

Parameter(s)

None

Constraint(s)

SDC

- *set_dont_touch* (Mandatory): Use to specify a list of cells, nets, references, designs, and library cells on which the `dont_touch` attribute is to be set.

Messages and Suggested Fix

The following message appears for a *set_dont_touch* constraint found in a non-netlist-level design unit `<du-name>` in design/block `<name>`:

```
[WARNING] set_dont_touch has been set to true (or default) for  
rtl design "<du-name>" of design/block "<name>"
```

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

Non netlist design that has the `dont_touch` attribute set can affect the results of optimization of a netlist.

How to Debug and Fix

Apply constraint for which object this rule is flagging violation

Example Code and/or Schematic

For the following snippet, the *Dont_Touch05* rule reports a violation because [set_dont_touch](#) is applied on and RTL instance.

```
//test.v
Module top(d,c,q);
Input d,c;
Output q;
MyFF I1(d,c,q);
Endmodule

Module MyFF(input d,c, output q);
Reg q;
Always @(posedge c)
Q <= d;
Endmodule

//test.sdc
set_dont_touch I1
```

Default Severity Label

Warning

Rule Group

Dont_Touch

Reports and Related Files

None

High Fan-out Rules

The High Fan-out Rules Sub-group `High_Fan` contains the following rules:

Rule	Description
<i>High_Fan01</i>	Nets whose fan-out exceeds a user-specified limit and none of the constraints related to high fan-out description are set on them
<i>High_Fan01a</i>	Non-clock nets whose fan-out exceeds a user-specified limit and none of the constraints related to high fan-out description are set on them
<i>High_Fan02</i>	Ideal nets
<i>High_Fan03</i>	Runs the <i>High_Fan03a</i> and <i>High_Fan03b</i> rules
<i>High_Fan03a</i>	Large fan-out special nets
<i>High_Fan03b</i>	Large fan-out nets other than special nets
<i>High_Fan04</i>	<i>set_ideal_transition</i> constraints that are not explicitly set with all required options
<i>High_Fan05</i>	<i>set_ideal_latency</i> constraints that are not explicitly set with all required options
<i>High_Fan06</i>	Intermediate nets with the <i>set_ideal_latency</i> constraint set
<i>High_Fan07</i>	Intermediate nets with the <i>set_ideal_transition</i> constraint set
<i>High_Fan08</i>	<i>set_ideal_transition</i> constraints with inconsistent maximum and minimum values
<i>High_Fan09</i>	<i>set_ideal_transition</i> constraints with inconsistent maximum and minimum values
<i>High_Fan10</i>	Objects with fan-out more than the specified limit
<i>High_Fan11</i>	<i>set_max_fanout</i> constraints that set a value more than a user-specified limit
<i>High_Fan12</i>	SDC files with a <i>set_max_fanout</i> constraint
<i>High_Fan14</i>	Ideal objects without a <i>set_ideal_latency</i> constraint
<i>High_Fan15</i>	Ideal objects without a <i>set_ideal_transition</i> constraint
<i>High_Fan16</i>	Start and end points of nets

High_Fan01

High fan-out net is not set as ideal or not set with annotated transition

When to Use

Use this rule in the RTL phase to ensure that overloading is not caused by any net.

Description

The *High_Fan01* rule reports nets that have a fan-out exceeding the limit and none of the constraints related to the high fan-out description are set on them.

The *High_Fan01* rule calculates the fan-out limit in the following order:

1. The value of the `max_fanout` attribute set for the pin driving the net if the pin is on an instantiated library cell, or
2. The value of the `default_max_fanout` attribute for the library. If multiple libraries are specified, the largest of all `default_max_fanout` attributes is used.
3. Otherwise, the value of the *tc_fanout_limit* parameter

Rule Exceptions

The High_Fan01 rule ignores following types of nets:

- Nets with the following constraints set:
 - `set_resistance 0` and `set_load 0` together
 - `set_ideal_network`
 - `set_ideal_net`
- Nets connected to leaf cell pins or top-level ports with *set_annotated_transition* constraint set
- Nets that are supply nets.

Parameter(s)

- *tc_fanout_limit*: Default is 200. The *High_Fan01* rule reports all nets that have a fan-out count higher than 200. As per your design needs, set the value to report nets that have a higher or lower fan-out count.

Constraint(s)

- *set_annotated_transition* (Optional): Use to set the transition time at a specified pin.
- *set_ideal_net* (Optional): Use to set a net as ideal.
- *set_ideal_network* (Optional): Use to set a net in fan-out as ideal.

Messages and Suggested Fix

The following message appears for a net `<net-name>` in design/block `<name>` that has a fan-out of `<num>` exceeding the maximum fan-out limit `<limit>` and has no high fan-out description-related constraints set:

```
[WARNING] No ideal constraint is set on high fan-out net
"<net-name>" (fan-out count <num>, limit <limit> (taken from
<limit-source>)) of design/block "<name>"
```

Where, `<limit-source>` can be `max_fanout` pin attribute, `default_max_fanout` library attribute, or *tc_fanout_limit* parameter specification.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

A degraded (overloaded) net leads to greater difficulty in meeting timing requirements because the load increases proportionally and the pin driving net would not transit to the desired state or would reach the desired state slowly.

How to Debug and Fix

The fan-out of the net is greater than the limit you have specified. The violation message states the location of the limit and the limit value. The design file highlights the definition of the net.

To resolve this violation, replicate the logic above the high fan-out net or increase the fan-out limit.

Example Code and/or Schematic

Suppose, you have a net with a fan-out count of 3. If the upper limit set by the *tc_fanout_limit* parameter is 2, the *High_Fan01* rule reports a violation.

Clock Rules

Default Severity Label

Warning

Rule Group

High_Fans

Reports and Related Files

None

High_Fan01a

Identifies high fan-out nets that are not set as ideal

When to Use

To determine nets that have a fan-out count, which exceeds a library related or user-specified limit. This rule is applicable to the RTL and Pre-layout phases.

Description

The *High_Fan01a* rule reports non-clock nets that have a fan-out, which exceeds a user-specified limit, and none of the constraints related to the high fan-out description are set on them.

The *High_Fan01a* rule calculates the fan-out limit in the following order:

1. The value of the `max_fanout` attribute set for the pin driving the net if the pin is on an instantiated library cell.
2. Otherwise, the `default_max_fanout` attribute value for the library. If multiple libraries are specified, the largest of all the `default_max_fanout` attributes is used.
3. Otherwise, the value of the `tc_fanout_limit` parameter.

Rule Exceptions

The *High_Fan01a* rule ignores following types of nets:

- Nets with the following constraints set:
 - `set_resistance 0` and `set_load 0` together
 - `set_ideal_network`
 - `set_ideal_net`
- Nets connected to leaf cell pins or top-level ports with `set_annotated_transition` constraint set
- Nets where the `create_clock` or `create_generated_clock` constraint is specified on the port/pin driving the net
- Nets that are supply nets or are tied high or low
- Nets that are specified using the `tc_ignore_nets` parameter

Parameter(s)

- *tc_fanout_limit*: Default is 200. The *High_Fan01a* rule reports all nets that have a fan-out count higher than 200. As per your design needs, set the value to report nets that have a higher or lower fan-out count.
- *tc_ignore_nets*: Default is unspecified. This indicates the *High_Fan01a* rule checks all nets. Set the value to the nets that you want the rule to ignore.

Constraint(s)

None

Messages and Suggested Fix

The following message appears for the non-clock net *<net-name>* in the design/block *<name>* that has the fan-out of *<num>* exceeding the maximum fan-out limit and has not got a high fan-out description related constraints set:

```
[WARNING] No ideal constraint is set on high fan-out net
"<net-name>" (fan-out count <num>, limit <limit> (taken from
<limit-source>)) of design/block "<name>"
```

Where, *<limit-source>* can be *max_fanout* pin attribute, *default_max_fanout* library attribute, or *tc_fanout_limit* parameter specification.

Potential Issues

If a net has a high fan-out then computing the transition time on this net could cause a degraded net.

Consequences of Not Fixing

A degraded net leads to greater difficulty in meeting timing requirements because the load increases proportionally and the pin driving net would not transit to the desired state or would reach the desired state slowly.

How to Debug and Fix

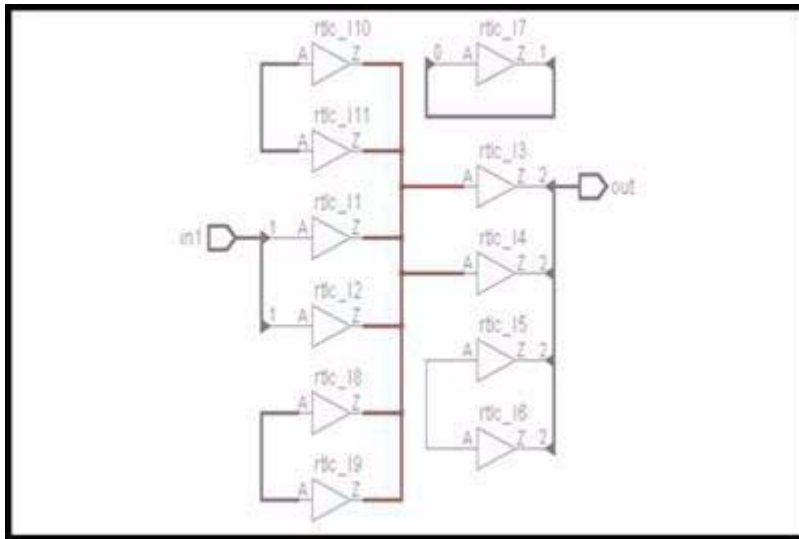
The fan-out of the net is greater than the limit you have specified. The violation message states the location of the limit and the limit value. The design file highlights the definition of the net.

To resolve this violation, replicate the logic above the high fan-out net or

increase the fan-out limit.

Example Code and/or Schematic

The *High_Fan01a* rule reports a violation for the net shown in the following schematic because the *tc_fanout_limit* parameter is specified as 1, but the fan-out count is more than one.



Default Severity Label

Warning

Rule Group

High_Fan

Reports and Related Files

None

High_Fan02

Reports ideal nets

When to Use

Use this rule in the RTL phase

Description

The *High_Fan02* rule reports ideal clock or non-clock nets. The *High_Fan02* rule considers nets connected to the clock pins of sequential elements as clock nets. All other nets are assumed to be non-clock nets.

This rule considers the following types of nets as ideal nets:

- Nets where one or more of the following constraints are set:
 - set_resistance 0 and set_load 0 together
 - set_ideal_network
 - set_ideal_net
- Nets connected to leaf cell pins or top-level ports with [set_annotated_transition](#) constraint set

Parameter(s)

None

Constraint(s)

- [set_annotated_transition](#) (Optional): Use to set the transition time at a specified pin.

Messages and Suggested Fix

Message 1

The following message appears for a non-clock net `<net-name>` in design/block `<name>` that has at least one of the ideal constraints set:

[WARNING] Ideal non-clock net "<net-name>" detected in design/block "<name>"

Message 2

The following message appears for a clock net `<net-name>` in design/block `<name>` that has at least one of the ideal constraints set:

```
[WARNING] Ideal clock net '<net-name>' detected in design/block "<name>"
```

Potential Issues

Clock nets are declared ideal since it is understood that they will drive many of sequential elements. Therefore, during CTS special care is taken for these kind of nets. Non-clock nets are not supposed to drive many items, except resets/supply nets.

Consequences of Not Fixing

Ideal non clock net may cause to chip failure due to huge transition time

How to Debug and Fix

To resolve this violation, you can either apply a clock to the net or remove the constraint applied on the net.

Example Code and/or Schematic

Suppose, you have specified the following and net `n1` has no clock:

```
//test.sdc
```

```
set_ideal_net n1
```

The *High_Fan02* rule will report a violation.

Default Severity Label

Warning

Rule Group

High_Fans

Reports and Related Files

None

High_Fan03

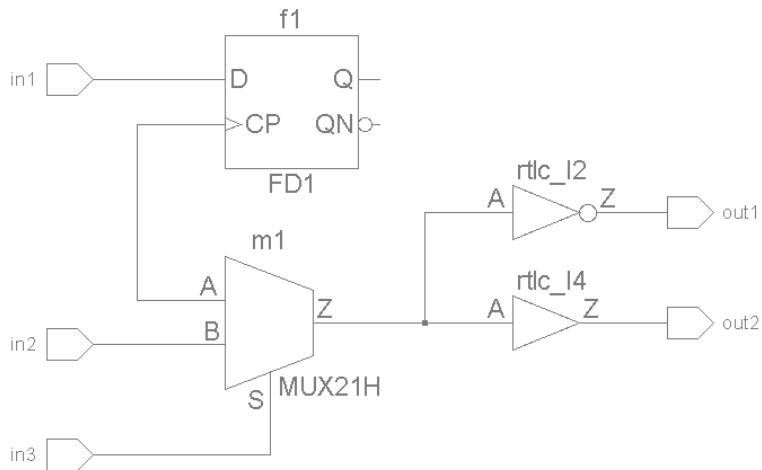
High fan-out net identified

The *High_Fan03* rule runs the *High_Fan03a* and *High_Fan03b* rules.

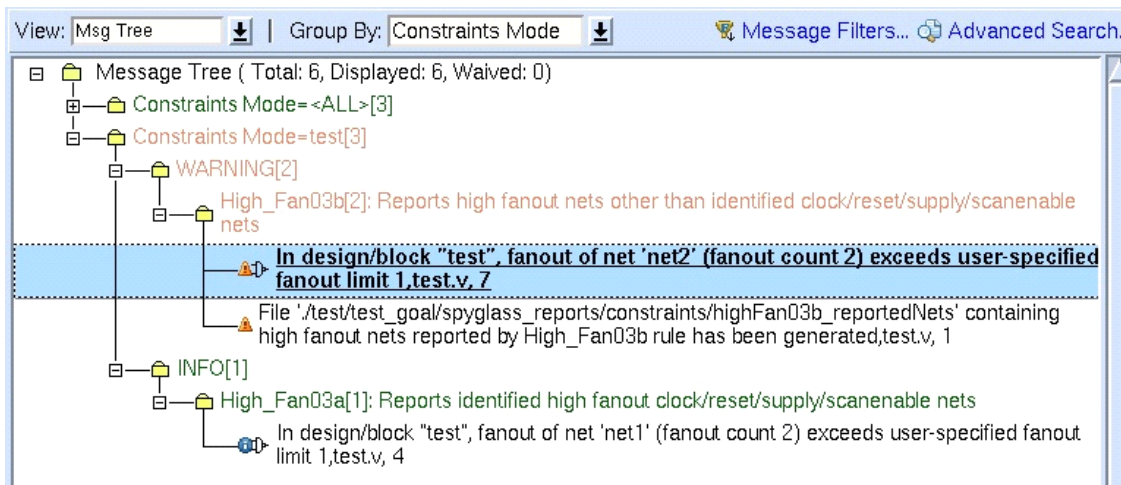
Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The following messages are generated:



The *High_Fan03a* message indicates that net1 in the design has exceeded the user specified fan-out limit for clock. View the project file for the specified limit ([View Test Case Files](#)).

The *High_Fan03b* message indicates that net2 in the design has exceeds the user specified fan-out limit for net2. View the project file for the specified limit ([View Test Case Files](#)).

High_Fan03a

Reports identified high fan-out clock/reset/supply/scanenable nets

When to Use

To identify control nets that have a fan-out count greater than the specified limit. This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *High_Fan03a* rule reports nets that have a large fan-out. This rule checks the following types of nets:

- Clock nets
- Generated clock nets
- Tied_off nets
- Enable (Scan Enable) nets
- Generated enable (Scan Enable) nets
- Reset (CLRZ/PREZ) nets
- Generated reset (CLRZ/PREZ) nets

A generated net is reported only if a non-generated net is identified in its fan-in. During the fan-in traversal, only single input gates, such as buffers and inverters are traversed, multiple input gates are not.

All other types of nets are checked by the [High_Fan03b](#) rule.

Parameter(s)

- *tc_clock_fanout_limit*: Default is 200. The *High_Fan03a* rule reports nets that have a fan-out count of more than 200. Set the value as per your design needs to report nets that have a higher or lower fan-out count.

Constraint(s)

None

Messages and Suggested Fix

The following message appears for the non-clock net *<net-name>* of the

design/block <name> with the fan-out count <num> that exceeds the limit specified by the *tc_clock_fanout_limit* parameter:

```
[INFO] In design/block "<name>", fan-out of net '<net-name>'
(fan-out count <num>) exceeds user-specified fan-out limit
$tc_clock_fanout_limit
```

Potential Issues

Not applicable

Consequences of Not Fixing

Not applicable

How to Debug and Fix

The net is highlighted in the design file. The net is driven by the clock/reset/scan-enable pin or is a supply net. The fan-out of the net is greater than the limit you have specified through the *tc_clock_fanout_limit* parameter. The fan-out count of the net and the limit value is stated in the violation message.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

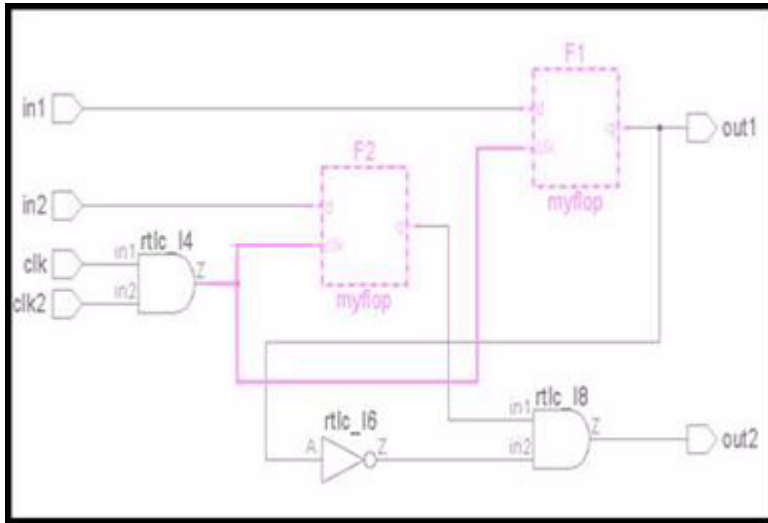
See [High_Fan03](#).

Example 2

Test Case Files Not Available

In the following schematic, the *High_Fan03a* rule reports a violation for the highlighted net because the *tc_clock_fanout_limit* is set to 1, but the fan-out is more than one.

Clock Rules

**Default Severity Label**

Info

Rule Group

High_Fan

Reports and Related Files

None

High_Fan03b

Reports high fan-out nets other than identified clock/reset/supply/scanenable nets

When to Use

To identify nets, other than control nets, that have a fan-out count greater than the specified limit. This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *High_Fan03b* rule reports nets that have a large fan-out. The *High_Fan03b* rule checks nets, including generated nets, which are not checked by the *High_Fan03a* rule. The *High_Fan03b* rule generates a file that contains a list of all high fan-out nets.

A generated net is reported only if a non-generated net is identified in its fan-in. During the fan-in traversal, only single input gates, such as buffers and inverters are traversed, multiple input gates are not.

If the *High_Fan16* rule is enabled and the *tc_high_fan_nets_file* parameter is not specified, the *High_Fan16* rule reports nets, which are contained in this file, that have more than the specified number of start-points or end-points.

Parameter(s)

- *tc_fanout_limit*: Default is 200. The *High_Fan03b* rule reports all nets that have a fan-out count higher than 200. Set the value as per your design needs to report nets that have a higher or lower fan-out count.

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message appears for the non-clock net *<net-name>* of the design/block *<name>* with the fan-out count *<num>* that exceeds the limit specified by the *tc_fanout_limit* parameter:

[WARNING] In design/block "<name>", fan-out of net '<net-name>' (fan-out count <num>) exceeds user-specified fan-out limit \$tc_fanout_limit

Potential Issues

If a net has a high fan-out then computing the transition time on this net could cause a degraded net.

Consequences of Not Fixing

A degraded net leads to greater difficulty in meeting timing requirements because the load increases proportionally and the pin driving net would not transit to the desired state or would reach the desired state slowly.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows a net. The net is not a clock, reset, scan-enable, or a supply net. The fan-out of the net is greater than the limit you have specified through the [tc_fanout_limit](#) parameter. The violation message states the fan-out of the net and the limit.

To resolve this violation, replicate the logic above the high fan-out net or increase the fan-out limit.

Message 2

The following message appears when the file <file-name> containing the list of all high fan-out nets is generated:

[INFO] File <file-name> containing high fan-out nets reported by High_Fan03b rule has been generated

Potential Issues

Not applicable

Consequences of Not Fixing

Not applicable

How to Debug and Fix

All the high fan-out nets reported by the *High_Fan03b* rule are listed in the file mentioned in the violation message. Review the file.

Example Code and/or Schematic

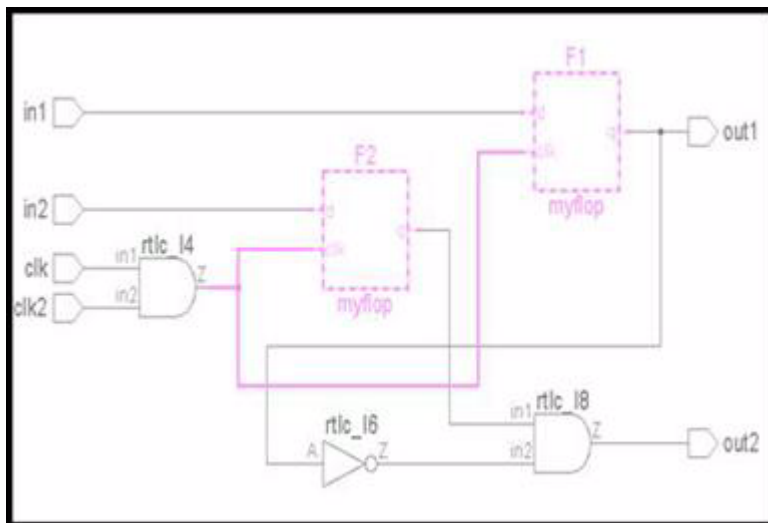
Example 1
[View Test Case Files](#)

See [High_Fan03](#).

Example 2

 Test Case Files Not Available

In the following schematic, the *High_Fan03b* rule reports a violation for the highlighted net because the *tc_fanout_limit* is set to 1, but the fan-out is more than one.



Default Severity Label

Warning

Rule Group

High_Fan

Clock Rules

Reports and Related Files

None

High_Fan04

Reports `set_ideal_transition` that have incomplete limits defined

When to Use

Use this rule in the RTL phase.

Description

The *High_Fan04* rule reports `set_ideal_transition` constraints that do not have all the required options set explicitly. This rule checks that a `set_ideal_transition` constraint is explicitly specified with both the `rise` and `fall` options or one of the `max` and `min` limits.

Rule Exceptions

The *High_Fan04* rule ignores the missing `min` option for worst-case SDC when the `tc_ignore_min` rule parameter is set.

Parameter(s)

- `tc_ignore_min`: Default is no. Set the value to yes to ignore the missing `-min` option in SDC command (in worst-case SDC file) whose completeness is being checked.

Constraint(s)

- `set_ideal_transition` (Mandatory): Use to specify ideal transition for an ideal network and ideal nets.

Messages and Suggested Fix

The following message appears for a `set_ideal_transition` constraint on an object `<obj-name>` of design/block `<name>` that has not been specified with all options explicitly:

```
[WARNING] Not all set_ideal_transition options specified explicitly for <obj-type> "<obj-name>" of design/block "<name>" (Specified: <option-list1>, Not specified: <option-list2>)
```

Where, `<option-list1>` is the list of options specified explicitly and `<option-list2>` is the list of options not specified explicitly.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

You need to explicitly define the values for the unspecified options to ensure the tool infers the desired values.

How to Debug and Fix

Specify the missing options for the [set_ideal_transition](#) constraint.

Example Code and/or Schematic

For the following snippet, the *High_Fan04* rule reports a violation because the [set_ideal_transition](#) constraint is not defined for the `fall` option.

```
//test.sdc  
set_ideal_transition 1.0 -rise in1
```

Default Severity Label

Warning

Rule Group

High_Fans

Reports and Related Files

None

High_Fan05

Reports `set_ideal_latency` that have incomplete limits defined

When to Use

Use this rule in the RTL phase.

Description

The *High_Fan05* rule reports `set_ideal_latency` constraints that are not explicitly set with all required arguments/limits. This rule checks that `set_ideal_latency` constraints for an object have been explicitly specified for both the `rise` and `fall` options or the `max` and `min` limits.

Rule Exceptions

The *High_Fan05* rule ignores the missing `-min` option for worst-case SDC when the `tc_ignore_min` parameter is set.

Parameter(s)

- `tc_ignore_min`: Default is no. Set the value to yes to ignore the missing `min` option in SDC command (in worst-case SDC file) whose completeness is being checked.

Constraint(s)

- `set_ideal_latency` (Mandatory): Use to specify ideal latency values for pins in an ideal network.

Messages and Suggested Fix

The following message appears for a `set_ideal_latency` constraint on an object `<obj-name>` of design/block `<name>` that has not been specified with all options explicitly:

[WARNING] Not all `set_ideal_latency` options specified explicitly `<obj-type> "<obj-name>"` of design/block "`<name>`" (Specified: `<option-list1>`, Not specified: `<option-list2>`)

Where, `<option-list1>` is the list of options specified explicitly and `<option-list2>` is the list of options not specified explicitly.

Potential Issues

The violation message appears because all *set_ideal_latency* options have not been defined explicitly.

Consequences of Not Fixing

You need to explicitly define the values for the unspecified options to ensure the tool infers the desired values.

How to Debug and Fix

To resolve the violation, specify the missing options for the *set_ideal_latency* constraint.

Example Code and/or Schematic

For the following snippet, the High_Fan05 rule reports a violation because the *set_ideal_latency* constraint is not defined for `fall` option.

```
//test.sdc
set_ideal_latency 1.0 -rise in1
```

Default Severity Label

Warning

Rule Group

High_Fans

Reports and Related Files

None

High_Fan06

Reports when `set_ideal_latency` is defined for an invalid or non-ideal object

When to Use

To identify whether the `set_ideal_latency` constraint is applied on valid objects. This rule is applicable to the RTL and Pre-layout phases.

Description

The `High_Fan06` rule reports invalid or non-ideal objects that have the `set_ideal_latency` constraint set. This rule reports the following types of objects:

- Objects that are neither top-level ports nor leaf-level pins
- The `High_Fan07` rule considers a port/pin to be ideal if it is constrained using the `set_ideal_network` or `set_annotated_transition` commands, or if the port/pin drives an ideal net. An ideal net is constrained using the `set_ideal_net` command.

Rule Exceptions

This rule does not consider nets that are constrained by `set_resistance 0` and/or `set_load 0` commands as ideal.

Parameter(s)

None

Constraint(s)

SDC

- `set_ideal_latency` (Mandatory): Use to set the `set_ideal_latency` constraint on either leaf pins or top level ports.
- `set_annotated_transition` (Optional): Use to set the transition time at a specified pin.
- `set_ideal_net` (Optional): Use to set a net as ideal.
- `set_ideal_network` (Optional): Use to set a net in fan-out as ideal.

Messages and Suggested Fix

Message 1

The following message appears for a *set_ideal_latency* constraint set on the invalid object *<obj-name>* of the design/block *<name>* that is not a top-level port or a leaf-level pin:

```
[WARNING] set_ideal_latency specified for an invalid object
"<obj-name>" of design/block "<name>" which is not a top level
port or a leaf-cell pin
```

Potential Issues

The *set_ideal_latency* constraint is applied only on top-level ports or leaf level pins. In this case, it is not.

Consequences of Not Fixing

If the *set_ideal_latency* constraint is applied on invalid objects, it is not considered. In addition, even if this constraint is applied on ports or leaf level pins, without marking them as ideal through the specified constraints, it is not considered while performing timing analysis.

How to Debug and Fix

The violation message indicates the invalid objects other than the top-level port or leaf-level pin. Review the invalid objects as highlighted in the SDC file.

Message 2

The following message appears for a *set_ideal_latency* constraint set on the non-ideal object *<obj-name>* of the design/block *<name>*:

```
[WARNING] set_ideal_latency specified for a non-ideal object
"<obj-name>" of design/block "<name>"
```

Potential Issues

The *set_ideal_latency* constraint is applied only on top-level ports or leaf level pins. In this case, it is not.

Consequences of Not Fixing

If the *set_ideal_latency* constraint is applied on invalid objects, it is not considered. In addition, even if this constraint is applied on ports or leaf level pins, without marking them as ideal through the specified constraints, it is not considered while performing timing analysis.

How to Debug and Fix

This violation message is reported when the [set_ideal_latency](#) constraint is specified on an object that is either a top-level port or a leaf-level pin but is not ideal. Ideal objects are those on which either the [set_ideal_network](#), [set_ideal_net](#) or [set_annotated_transition](#) constraints are set.

Review the constraints as highlighted in the SDC file.

Example Code and/or Schematic

Example 1

In this example, the *High_Fan06* rule reports a violation because the [set_ideal_latency](#) constraint is set on an object that is neither a leaf pin nor a top-level port.

```
set_ideal_latency 1 B1
```

Example 2

In this example, the *High_Fan06* rule reports a violation because the object on which [set_ideal_latency](#) is set is not an ideal object.

```
set_ideal_latency 1 B1/in1
```

To resolve this violation, specify the [set_ideal_net](#) constraint on the pin, and set the [set_ideal_latency](#) constraint.

Default Severity Label

Warning

Rule Group

High_Fan

Reports and Related Files

None

High_Fan07

Reports when `set_ideal_transition` is set on an inappropriate object

When to Use

To check whether `set_ideal_transition` is applied on valid objects. This rule is applicable to the RTL and Pre-layout phases.

Description

The *High_Fan07* rule reports invalid objects that have the `set_ideal_transition` constraint set. This rule reports the following types of objects:

- Objects that are neither top-level ports nor leaf-level pins
- Top-level ports or leaf-level pins that have the `set_ideal_transition` constraint set, but are not declared as ideal objects. The *High_Fan07* rule considers a port/pin to be ideal only if it is constrained using the `set_ideal_network` or `set_annotated_transition` commands, or if the port/pin drives an ideal net. An ideal net is constrained using the `set_ideal_net` command.

Rule Exceptions

This rule does not consider nets that are constrained by `set_resistance 0` and/or `set_load 0` commands as ideal.

Parameter(s)

None

Constraint(s)

SDC

- `set_ideal_transition` (Mandatory): Used to specify ideal transition for an ideal network and ideal nets.
- `set_annotated_transition` (Optional): Use to set the transition time at a specified pin.
- `set_ideal_net` (Optional): Use to set a net as ideal.
- `set_ideal_network` (Optional): Use to set a net in fan-out as ideal.

Messages and Suggested Fix

Message 1

The following message appears for the [set_ideal_transition](#) constraint set on an invalid object `<obj-name>` of design/block `<name>` that is not a top-level port or leaf-level pin:

```
[WARNING] set_ideal_transition specified for an invalid object
"<obj-name>" of design/block "<name>" which is not a top level
port or a leaf-cell pin
```

Potential Issues

The [set_ideal_transition](#) constraint is applied only on top-level ports or leaf level pins. In this case, it is not.

Consequences of Not Fixing

If the [set_ideal_transition](#) constraint is applied on invalid objects, it is not considered. In addition, even if this constraint is applied on ports or leaf level pins, without marking them ideal through the specified constraints, it is not considered while doing timing analysis.

How to Debug and Fix

The violation message indicates the invalid objects other than top-level port or leaf-level pin.

Review the invalid objects as highlighted in the SDC file.

Message 2

The following message appears for the [set_ideal_transition](#) constraint set on a non-ideal object `<obj-name>` of design/block `<name>`:

```
[WARNING] set_ideal_transition specified for a non-ideal object
"<obj-name>" of design/block "<name>"
```

Potential Issues

The [set_ideal_transition](#) constraint is applied only on top-level ports or leaf level pins. In this case, it is not.

Consequences of Not Fixing

If the [set_ideal_transition](#) constraint is applied on invalid objects, it is not considered. In addition, even if this constraint is applied on ports or leaf level pins, without marking them ideal through the specified constraints, it is not considered while doing timing analysis.

How to Debug and Fix

This violation message is reported when the [set_ideal_transition](#) constraint is specified on an object that is either a top-level port or a leaf-level pin, but the object is not ideal. Ideal objects are those on which either the [set_ideal_network](#), [set_ideal_net](#) or [set_annotated_transition](#) constraints are set.

Review the constraints as highlighted in the SDC file.

Example Code and/or Schematic

Example 1

In this example, the *High_Fan07* rule reports a violation because the [set_ideal_transition](#) constraint is set on an object that is neither a leaf pin nor a top-level port.

```
set_ideal_transition 1 B1
```

Example 2

In this example, the *High_Fan07* rule reports a violation because the object on which [set_ideal_transition](#) is set is not an ideal object.

```
set_ideal_transition 1 B1/in1
```

Default Severity Label

Warning

Rule Group

High_Fan

Reports and Related Files

None

High_Fan08

Reports `set_ideal_transition` specified with inconsistent option values

When to Use

Use this rule in the RTL phase.

Description

The High_Fan08 rule reports `set_ideal_transition` constraints with inconsistent maximum and minimum values.

Parameter(s)

None

Constraint(s)

- `set_ideal_transition` (Mandatory): Use to specify ideal transition for an ideal network and ideal nets.

Messages and Suggested Fix

The following message appears for a `set_ideal_transition` constraint on an object `<obj-name>` in design/block `<name>` where the maximum transition value `<value1>` is less than the minimum transition value `<value2>` (set in file `<file-name>`, line `<num>`):

```
[WARNING] <type1> set_ideal_transition value <value1> is less than <type2> ideal transition value <value2> (set at file: <file-name> line: <num>) for <obj-type> "<obj-name>" of design/block "<name>"
```

Where `<type1>` can be `Max_Fall` or `Max_Rise` and `<type2>` can be `Min_Fall` or `Min_Rise`.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

When you specify minimum values greater than maximum values, the tool makes assumptions and can change one of the two specifications. This

leads to undesired behavior.

How to Debug and Fix

Ensure that the value of the `min` argument is lower than that of the `max` value.

Example Code and/or Schematic

For the following snippet, the *High_Fan08* rule reports a violation because the value of the `min` argument is greater than the value of the `max` argument.

```
//test.sdc
set_ideal_transition 1.2 -min -rise {rst}
set_ideal_transition 1.1 -max -rise {rst}
```

Default Severity Label

Warning

Rule Group

High_Fans

Reports and Related Files

None

High_Fan09

Reports `set_ideal_latency` specified with inconsistent option values

When to Use

Use this rule in the RTL phase.

Description

The *High_Fan09* rule reports `set_ideal_latency` constraints with inconsistent maximum and minimum values.

Parameter(s)

None

Constraint(s)

- `set_ideal_latency` (Mandatory): Use to specify ideal latency values for pins in an ideal network.

Messages and Suggested Fix

The following message appears for a `set_ideal_latency` constraint on an object `<obj-name>` in design/block `<name>` where the maximum value `<value1>` is less than the minimum value `<value2>` (set in file `<file-name>`, line `<num>`):

```
<type1> set_ideal_latency value <value1> is less than <type2>
ideal latency value <value2> (set at file: <file-name> line:
<num>) for <obj-type> "<obj-name>" of design/block "<name>"
```

Where `<type1>` can be `Max_Fall` or `Max_Rise` and `<type2>` can be `Min_Fall` or `Min_Rise`.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

When you specify minimum values greater than maximum values, the tool makes assumptions and can change one of the two specifications. This leads to undesired behavior.

How to Debug and Fix

Ensure that the value of the `max` argument is greater than that of the `min` value.

Example Code and/or Schematic

For the following snippet, the *High_Fan09* rule reports a violation because the value of the `max` argument is less than that of the `min` argument.

```
//test.sdc
set_ideal_latency 1.2 -min -rise {rst}
set_ideal_latency 1.1 -max -rise {rst}
```

Default Severity Label

Warning

Rule Group

High_Fans

Reports and Related Files

None

High_Fan10

Reports fan-out in the design when it is greater than the limit set by `set_max_fanout`

When to Use

Use this rule in the post-layout phase.

Description

The *High_Fan10* rule reports objects with a fan-out that is more than the specified limit. The maximum allowed fan-out for an object is specified using the *set_max_fanout* constraint.

Parameter(s)

None

Constraint(s)

- *set_max_fanout*: Use to specify maximum fan-out load in library fan-out units.

Messages and Suggested Fix

The following message appears for an object `<name>` with a fan-out count `<num>` that exceeds the limit `<max>` specified by the corresponding *set_max_fanout* constraint:

[WARNING] Fan-out (`<num>`) in the design/block "`<name>`" for design object "`<obj-name>`" is more than the `set_max_fanout` (`<max>`)

Potential Issues

There are nets in the design that have a high fan-out.

Consequences of Not Fixing

The high fan-out could potentially make it difficult to meet timing.

How to Debug and Fix

Redo the logic to reduce the fan-out count.

Example Code and/or Schematic

For the following snippet, the High_Fan10 rule will report a violation if the fan-out count of `in` is greater than 1.

```
//test.sdc  
set_max_fanout 1 [get_ports in]
```

Default Severity Label

Warning

Rule Group

High_Fans

Reports and Related Files

None

High_Fan11

Reports `set_max_fanout` constraint greater than certain limit

When to Use

Use this rule in the RTL phase.

Description

The *High_Fan11* rule reports `set_max_fanout` constraints that set a value more than the specified limit. This rule reports input/inout ports as follows:

- Ports that have both the `set_max_fanout` and the `set_driving_cell` constraint set when the value set with the `set_max_fanout` constraint is greater than the value of the `max_fanout` attribute defined in the library for the pin associated with the `set_driving_cell` constraint.
- Ports that have only the `set_max_fanout` constraint set when the value set with the `set_max_fanout` constraint is greater the value of the `tc_fanout_limit` parameter.

Parameter(s)

- `tc_fanout_limit`: Default is 200. The *High_Fan11* rule reports all nets that have a fan-out count higher than 200. As per your design needs, set the value to report nets that have a higher or lower fan-out count.

Constraint(s)

- `set_max_fanout` (Mandatory): Use to define input ports or designs where the maximum fan-out needs to be set.
- `set_driving_cell` (Optional): Use to define input or inout port names in the current design on which the driving cell information is set.

Messages and Suggested Fix

Message 1

The following message appears for port `<port-name>` with both the `set_max_fanout` and the `set_driving_cell` constraints set and the value `<value>` set with `set_max_fanout` is greater than the value of the `max_fanout` attribute defined in the library for the pin (of cell `<cell-`

name>) associated with [set_driving_cell](#):

[WARNING] `max_fanout(<value>)` specified for port "`<port-name>`" is greater than the maximum allowed fan-out (`<limit>`) (inferred from `max_fanout` attribute in library for cell `<cell-name>`)

Message 2

The following message appears for port `<port-name>` with the [set_max_fanout](#) constraint set and its value `<value>` exceeds the maximum limit `<limit>` specified by the [tc_fanout_limit](#) parameter:

[WARNING] `max_fanout(<value>)` specified for port "`<port-name>`" is greater than the maximum allowed fan-out (`<limit>`) (inferred from rule parameter `-tc_fanout_limit`)

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Since the constraint specified using the [set_max_fanout](#) constraint is conflicting with the upper limit imposed in the library, the timing analysis may not be accurate.

How to Debug and Fix

To resolve this violation, either remove the constraint or make it equal to the value in the library.

Example Code and/or Schematic

For the following snippet, the `High_Fan11` rule reports a violation because `default_max_fanout` is not equal to the [set_max_fanout](#) value.

```
//test.sdc
set_max_fanout 1 [get_ports in]
```

The value defined in lib is:

```
library(abc){
.
.
}
```

```
default_max_fanout : 10;  
  
}
```

Default Severity Label

Warning

Rule Group

High_Fans

Reports and Related Files

None

High_Fan12

Reports `set_max_fanout` constraint that are not specified for input ports

When to Use

Use this rule in the RTL phase.

Description

The *High_Fan12* rule reports input ports or a design without a `set_max_fanout` constraint. The *High_Fan12* rule reports input ports or design without the `set_max_fanout` constraint depending on the value of the `tc_opt01_port_des` parameter as shown below:

Value	Rule flags...
design	The design for which the <code>set_max_fanout</code> constraint is not specified
port	The input ports for which the <code>set_max_fanout</code> constraint is not specified
both	The design and input ports for which the <code>set_max_fanout</code> constraint is not specified
design_or_port	The input ports if the <code>set_max_fanout</code> constraint is not specified for a design, where design is the circuit that needs to be analyzed.

Parameter(s)

- `tc_opt01_port_des`: Default value is both. To generate the `set_max_transition`, `set_max_capacitance`, and `set_max_fanout` constraints, set the value of this parameter to either `design`, `port`, `both`, or `design_or_port` (either port or design).

Constraint(s)

- `set_max_fanout` (Mandatory): Use to define input ports or designs where the maximum fan-out needs to be set.

Messages and Suggested Fix

Message 1

The following message appears for an input port `<port-name>` without a `set_max_fanout` constraint set:

[WARNING] set_max_fanout is not specified for input port <port-name> for design/block design_or_block_name

Message 2

The following message appears for a design `<name>` without a `set_max_fanout` constraint set:

[WARNING] set_max_fanout is not specified for design <name>

Potential Issues

The violation messages explicitly specify the potential issues.

Consequences of Not Fixing

You are totally dependent on the .lib for identifying the `set_max_fanout` constraint. If the .lib does not have this limit (or, if this limit is high), the tool can insert a very high fan-out value. As a result, it is difficult to meet timing.

How to Debug and Fix

Depending on the value of the `tc_opt01_port_des` parameter, you should specify the value for each `current_design` or `port`.

Example Code and/or Schematic

For the following snippet, the `High_Fan12` rule reports a violation because the `set_max_fanout` constraint is not defined for the input port `in1` and the value of the `tc_opt01_port_des` parameter is set to `port`.

```
//test.sdc
create_clock -name CLK -period 20
set_input_delay in1 1.0 -clock CLK
```

Default Severity Label

Warning

Clock Rules

Rule Group

High_Fans

Reports and Related Files

None

High_Fan14

Reports `set_ideal_latency` not set on an ideal object

When to Use

Use this rule in the RTL phase.

Description

The *High_Fan14* rule reports ideal objects that do not have a `set_ideal_latency` constraint specified. The *High_Fan14* rule considers objects where one or more of the following constraints are set as ideal objects:

- `set_ideal_latency`
- `set_ideal_net`

Parameter(s)

Not applicable

Constraint(s)

- `set_ideal_latency` (Mandatory): Use to specify ideal latency values for pins in an ideal network.

Messages and Suggested Fix

The following message appears for ideal object `<obj-name>` of design/block `<name>` without a `set_ideal_latency` constraint set:

```
[WARNING] set_ideal_latency not specified for an ideal object
"<obj-name>" of design/block "<name>"
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

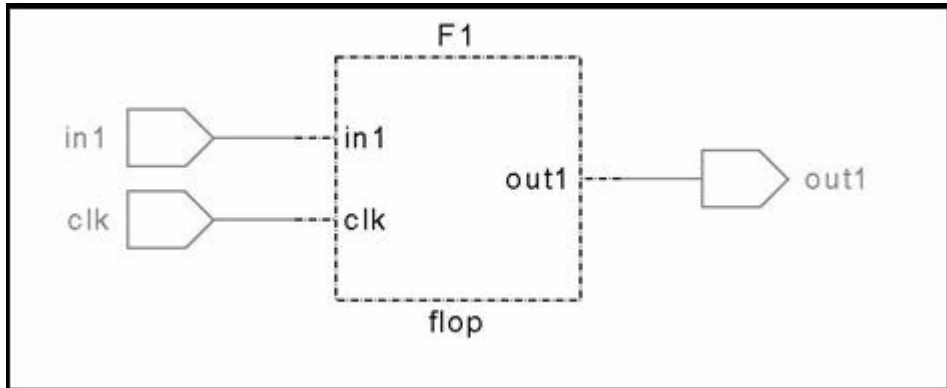
The tool might assume its own default or internally calculate the value that can be different from the intended value.

How to Debug and Fix

Apply the `set_ideal_latency` constraint on the reported ideal object.

Example Code and/or Schematic

Consider the following design.



The constraint applied is as follows:

```
set_ideal_network in1
```

The *High_Fan14* rule reports a violation because the [set_ideal_latency](#) constraint is not applied on object `in1`.

Default Severity Label

Warning

Rule Group

High_Fans

Reports and Related Files

None

High_Fan15

set_ideal_transition not set on an ideal object

Reports set_ideal_transition that is not set on an ideal object

When to Use

Use this rule in the RTL phase.

Description

The *High_Fan15* rule reports ideal objects that do not have a *set_ideal_transition* constraint specified. The *High_Fan15* rule considers objects where one or more of the following constraints are set as ideal objects:

- *set_ideal_transition*
- *set_ideal_net*

Parameter(s)

None

Constraint(s)

- *set_ideal_transition* (Mandatory): Use to specify ideal transition for an ideal network and ideal nets.

Messages and Suggested Fix

The following message appears for ideal object *<obj-name>* of design/block *<name>* without a *set_ideal_transition* constraint set:

```
[WARNING] set_ideal_transition not specified for an ideal object "<obj-name>" of design/block "<name>"
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

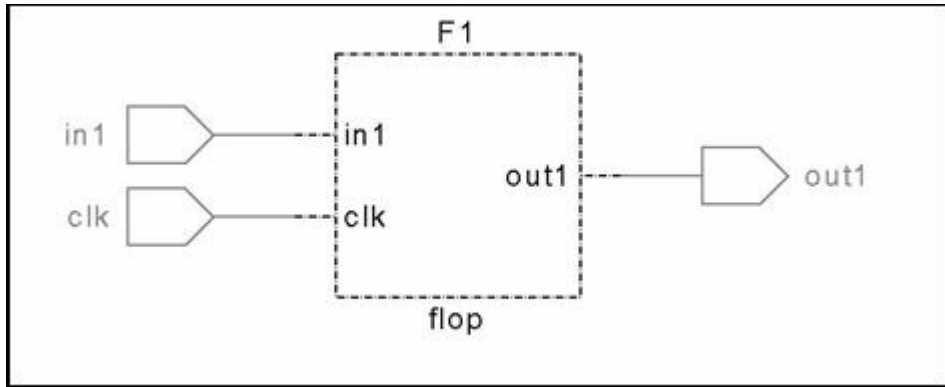
The tool might assume its own default or internally calculate the value that can be different from the intended value.

How to Debug and Fix

Apply the *set_ideal_transition* constraint on the reported ideal object.

Example Code and/or Schematic

Consider the following design.



The constraint applied is as follows:

```
set_ideal_network in1
```

The *High_Fan15* rule reports a violation because the *set_ideal_transition* constraint is not applied on object `in1`.

Default Severity Label

Warning

Rule Group

High_Fans

Reports and Related Files

None

High_Fan16

Reports start and end points of nets

When to Use

Use this rule in the RTL phase.

Description

The *High_Fan16* rule reports nets that have more than the specified number of start- or end-points. The *High_Fan16* rule reports nets where the total number of start-points (end-points) in the fan-in (fan-out) of the net is more than the limit specified by the *tc_start_end_max_count* parameter.

The *High_Fan16* rule considers input ports, inout ports, and outputs of sequential cells as start-points and considers output ports, inout ports, and inputs of sequential cells as end-points.

Rule Exceptions

If the *tc_high_fan_nets_file* parameter is not specified and the *High_Fan03b* rule is not enabled, the *High_Fan16* rule does not check for any nets.

Parameter(s)

- *tc_high_fan_nets_file*: Default is none. Set the value to a file name that contains a list of RTL nets that need to be checked.
- *tc_start_end_max_count*: Default is 100. Set the value to a positive integer to define the maximum limit for the High-Fan16 rule.

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message appears when the CSV file named `<du-name>-StartPoints.csv` is generated for design unit `<du-name>`:

```
[INFO] Startpoint details for design "<du-name>" have been generated in file <file-name>
```

Where, *<file-name>* is the complete path of the generated *<du-name>-StartPoints.csv* file.

Message 2

The following message appears when the CSV file named *<du-name>-EndPoints.csv* is generated for design unit *<du-name>*:

[INFO] Endpoint details for design "*<du-name>*" have been generated in file *<file-name>*

Where, *<file-name>* is the complete path of the generated *<du-name>-EndPoints.csv* file.

Message 3

The following message appears when nets *<net-name-list>* specified in the file *<file-name>* using the *tc_high_fan_nets_file* parameter are not found in the design unit *<du-name>*:

[INFO] Nets (*<net-name-list>*) mentioned in "-tc_high_fan_nets_file *<file-name>*" not found in design "*<du-name>*"

Message 4

The following message is generated when neither the *tc_high_fan_nets_file* parameter is specified nor the High_Fan03b rule is enabled:

[INFO] Neither tc_high_fan_nets_file rule parameter is specified, nor 'High_Fan03b' rule is enabled, so no checking is done

Potential Issues

Not applicable.

Consequences of Not Fixing

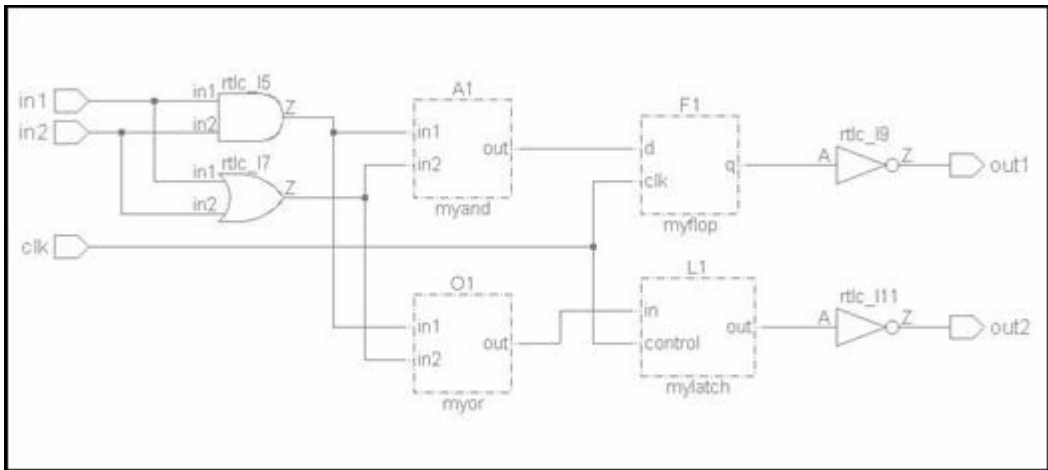
Not applicable.

How to Debug and Fix

Double-click the violation messages to view the generated files.

Example Code and/or Schematic

Consider the following design.



The constraints defined are as follows:

```
//test.sdc
set_ideal_network w1 #w1 is a net connected to port in1
set_ideal_network w2 #w2 is a net connected to port in2
```

The reports generated are:

```
// top-StartPoints.csv
Start Nets,Start Points count,Start Points
top/w1,2,in1
,,in2 #where in1, in2 are start points for top/w1
top/w2,2,in1
,,in2 #where in1, in2 are start points for top/w2

// top-EndPoints.csv
Start Nets,End Points count,End Points
top/w1,2,F1/q_reg/D
```

```

,,L1/out_reg/D # F1/q_reg/D & L1/out_reg/D are end points of
top/w1
top/w2,2,F1/q_reg/D
,,L1/out_reg/D
top/c1k,2,F1/q_reg/CP
,,L1/out_reg/E
top/in1,2,F1/q_reg/D
,,L1/out_reg/D
top/in2,2,F1/q_reg/D
,,L1/out_reg/D

```

Default Severity Label

Info

Rule Group

High_Fans

Reports and Related Files

The *High_Fan16* rule generates the two CSV files named <duname>-StartPoints.csv and <duname>-EndPoints.csv that contain the details of nets where the start-points (end-points) exceed the specified limit.

The CSV reports have the following fields:

- Name of the net
- Actual count of start-points (or end-points)
- Name list of start-points (or end-points) only when the [tc_start_end_point_details](#) parameter is set to yes

Refer to the Example Code and/or Schematic section for samples of these files and how to interpret them.

Clock Group Rules

The following rules belong to the SCGRules group.

Rule	Description
<i>SCG01</i>	Reports generated relationships between clocks and <i>set_clock_groups</i> constraint has been defined between them
<i>SCG02</i>	Reports generated clocks that do not have <i>set_clock_groups</i> defined between them
<i>SCG03</i>	Reports conflicting <i>set_clock_uncertainty</i> constraint for clocks defined as asynchronous/physically exclusive
<i>SCG04</i>	Reports clocks, which have a <i>set_clock_groups</i> defined, share a harmonic relationship
<i>SCG05</i>	Reports multiple clock definitions at the same node as being physically exclusive

SCG01

Reports generated relationships between clocks and `set_clock_groups` has been defined between them

When to Use

Use this rule for RTL and netlists.

Description

The *SCG01* rule checks for generated relationship between clocks of the asynchronous or `physically_exclusive` clock groups. For example, relationships between generated clocks from the same master or between generated clock and master clock.

Refer to the [Example Code and/or Schematic](#) section to view SDC commands that generate violations.

Parameter(s)

None

Constraint(s)

- [set_clock_groups](#) (Mandatory): Use to specify the clock groups that are mutually exclusive or asynchronous so that the paths between these clocks are not considered during timing analysis.

Messages and Suggested Fix

Message 1

The following message appears when a generated relationship exists between two clocks:

```
[SCG01_01][ERROR] Parent generated relationship exists between  
Clocks <cl k1> and <cl k2> and set_clock_groups is defined  
between them
```

For debugging information, refer to [How to Debug and Fix](#).

Message 2

The following message appears when two clocks are from the same master:

[SCG01_02][ERROR] Generated Clocks <clk1> and <clk2> are from same master and set_clock_groups is defined between them

Potential Issues

These messages appear because either generated relationship exists between two clocks that have the [set_clock_groups](#) constraint applied.

Consequences of Not Fixing

If you do not fix this violation, other tools can assume the clock relationship differently.

How to Debug and Fix

To resolve these violations, remove the [set_clock_groups](#) constraint.

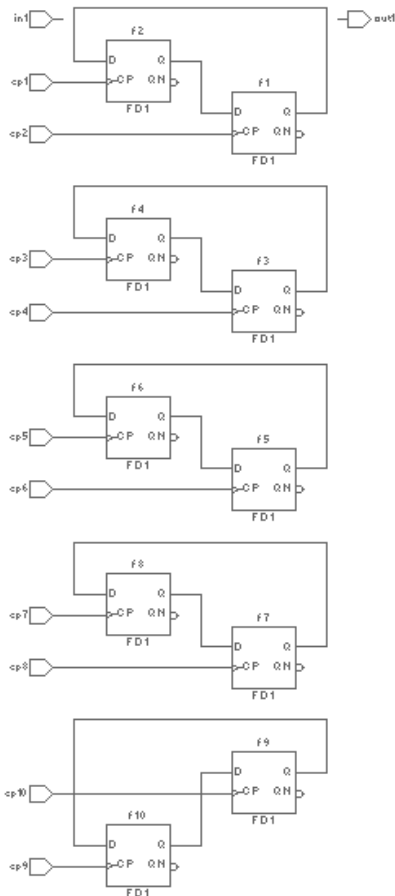
Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:

Clock Group Rules



The top.sdc file snippet is as follows:

```

11 set_clock_groups -async -group {clk1} -group {clk2}
12 set_clock_groups -async -group {clk1} -group {gclk2}
13 set_clock_groups -async -group {clk3} -group {clk4}

```

The violation message appears because the generated clock `gclk1` is generated from the create clock `clk1`, but in the `top.sdc` file there is a `set_clock_groups` constraint defined with `-async` argument.

Example 2Test Case Files Not Available

The following examples describe some scenarios where the *SCG01* rule reports a violation. All examples in this section pertain to the following SDC snippet.

```
create_clock -name C1 -period 10 clk1
create_clock -name C2 -period 13 clk2
create_clock -name C3 -period 14 clk3
create_clock -name C4 -period 17 clk4
create_generated_clock -name gC1 -source clk1 -divide_by 1
clk5
create_generated_clock -name gC12 -source clk1 -divide_by 1
clk8
create_generated_clock -name gC2 -source clk2 -divide_by 1
clk6
create_generated_clock -name gC11 -source clk5 -divide_by 1
clk7
```

Generated Relationship Between Clocks

This example shows SDC commands that cause *Message 1* to be reported because a generated relationship exists between two clocks:

```
set_clock_groups -async -group {C1} -group {gC1}
set_clock_groups -async -group {C1} -group {gC11}
```

To resolve these violations, remove the *set_clock_groups* constraint.

Two Clocks From the Same Master

This example shows an SDC command that causes *Message 2* to be reported because the generated both clocks are from the same master clk1:

```
set_clock_groups -async -group {gC1} -group {gC12}
```

To resolve these violations, remove the *set_clock_groups* constraint.

No Violation Reported

This example illustrates SDC commands that will not result in any violation. This is because the *set_clock_groups* is applied on clocks that are not related.

```
set_clock_groups -async -group {C1} -group {C2}
set_clock_groups -async -group {C1} -group {gC2}
set_clock_groups -async -group {C3} -group {C4}
```

Default Severity Label

Error

Rule Group

SCGRules

Reports and Related Files

None

SCG02

Reports generated clocks that do not have `set_clock_groups` defined between them

When to Use

Use this rule for RTL and netlists.

Description

The *SCG02* rule reports a violation when two clocks are defined as asynchronous/physically_exclusive and their generated clocks are not defined as asynchronous/physically_exclusive.

Refer to the [Example Code and/or Schematic](#) section to view SDC commands that generate violations.

Parameter(s)

None

Constraint(s)

- *set_clock_groups* (Optional): Use to specify the clock groups that are mutually exclusive or asynchronous so that the paths between these clocks are not considered during timing analysis.

Messages and Suggested Fix

The following message appears when a generated clocks do not have *set_clock_groups* defined:

[ERROR] `set_clock_groups` are defined between clocks `<clk1>` and `<clk2>`, so their generated clocks `<genclk1>` and `<genclk2>` should also have `set_clock_groups` defined between them

Potential Issues

The violation message explicitly specifies the potential issue.

Consequences of Not Fixing

If you do not fix this violation, other tools can assume the clock relationship differently.

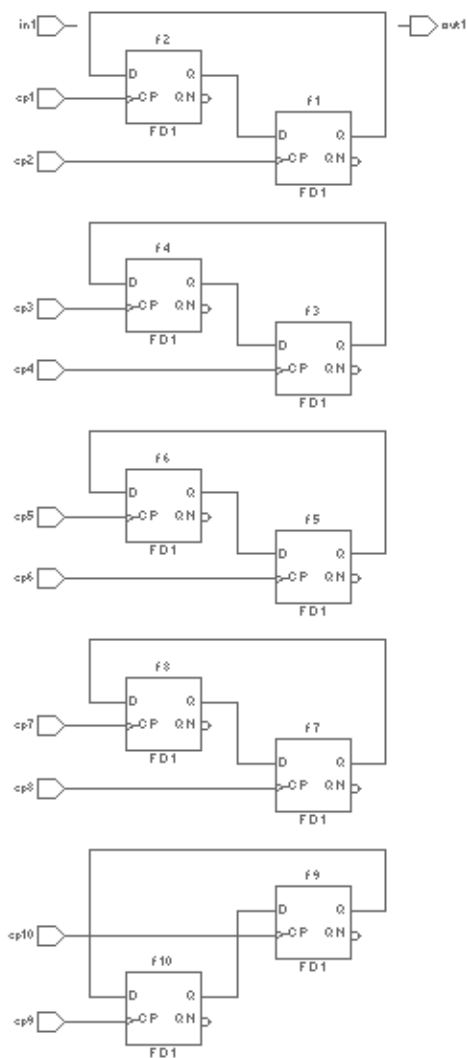
How to Debug and Fix

To resolve this violation, apply the [set_clock_groups](#) constraint on the generated clocks.

Example Code and/or Schematic

Example 1[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top .sdc file snippet is as follows:

Clock Group Rules

```

top.sdc x
1 create_clock -name clk1 -period 10 cp1
2 create_clock -name clk2 -period 13 cp2
3 create_clock -name clk3 -period 14 cp3
4 create_clock -name clk4 -period 17 cp4
5 create_generated_clock -name gclk1 -source cp1 -divide_by 1 cp5
6 create_generated_clock -name gclk12 -source cp1 -divide_by 1 cp8
7 create_generated_clock -name gclk2 -source cp2 -divide_by 1 cp6
8 create_generated_clock -name gclk11 -source cp5 -divide_by 1 cp7
9
10 #scg on unrelated clocks, error for missing scg between gclk1-gclk2, gclk12-gclk2
11 set_clock_groups -async -group {clk1} -group {clk2}
12 #still flag for -async
13 set_clock_groups -physically_exclusive -group {gclk1} -group {gclk2}
14 #do not flag for gcl2-gc2
15 set_clock_groups -async -group {gclk12} -group {gclk2}

```

The violation message appears because the `set_clock_groups` constraint has been defined on `clk1` and `clk2` clocks, but not for their generated clocks `gclk1` and `gclk2`.

Example 2

Test Case Files Not Available

The examples in this section provide some example of SDC commands that are reported by the `SCG02` rule. The examples pertain to the following SDC snippet:

```

create_clock -name C1 -period 10 clk1
create_clock -name C2 -period 13 clk2
create_clock -name C3 -period 14 clk3
create_clock -name C4 -period 17 clk4
create_generated_clock -name gC1 -source clk1 -divide_by 1
clk5
create_generated_clock -name gC12 -source clk1 -divide_by 1
clk8
create_generated_clock -name gC2 -source clk2 -divide_by 1
clk6
create_generated_clock -name gC11 -source clk5 -divide_by 1

```

```
clk7
```

The SCG02 Rule Reports a Violation Message

In this example, the `set_clock_groups` is not specified between gC1-gC2 and gC12-gC2. Therefore, an error message is reported.

```
set_clock_groups -async -group {C1} -group {C2}
```

To resolve this violation, apply the `set_clock_groups` constraint on the generated clocks.

The SCG02 Rule Does Not Report a Violation Message

In this example, the `SCG02` rule does not report a violation message. This is because `set_clock_groups` is applied on generated clocks.

```
set_clock_groups -async -group {gC12} -group {gC2}
```

Default Severity Label

Error

Rule Group

SCGRules

Reports and Related Files

None

SCG03

Reports conflicting `set_clock_uncertainty` constraints for clocks defined as asynchronous/physically exclusive

When to Use

Use this rule for RTL and netlists.

Description

The *SCG03* rule reports a violation when `set_clock_uncertainty` constraint is defined for clocks that have the `set_clock_groups` constraint defined between them.

Refer to the *Example Code and/or Schematic* section to view SDC commands that generate violations.

Parameter(s)

None

Constraint(s)

- `set_clock_groups` (Mandatory): Use to specify the clock groups that are mutually exclusive or asynchronous so that the paths between these clocks are not considered during timing analysis.
- `set_clock_uncertainty` (Mandatory): Use to specify uncertainty of specified clock networks.

Messages and Suggested Fix

The following message appears when there is a conflicting `set_clock_uncertainty` constraint defined:

```
[ERROR] set_clock_uncertainty defined between clocks <cl k1>
and <cl k2> having set_clock_groups constraint defined between
them
```

Potential Issues

The violation message explicitly specifies the potential issue.

Consequences of Not Fixing

If you do not fix this violation, other tools can assume the clock

relationship differently.

How to Debug and Fix

To resolve this violation, remove the conflicting [set_clock_uncertainty](#) constraint.

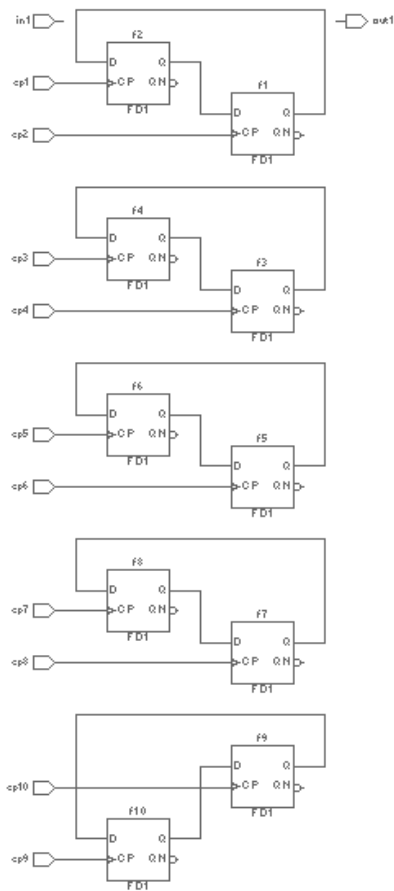
Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:

Clock Group Rules



The top .sdc file snippet is as follows:

```
top.sdc x
1 create_clock -name clk1 -period 10 cp1
2 create_clock -name clk2 -period 13 cp2
3 create_clock -name clk3 -period 14 cp3
4 create_clock -name clk4 -period 17 cp4
5 create_generated_clock -name gclk1 -source cp1 -divide_by 1 cp5
6 create_generated_clock -name gclk12 -source cp1 -divide_by 1 cp8
7
8
9 set_clock_groups -async -group {clk1} -group {clk2}
10
11
12 set_clock_uncertainty 1 -from clk2 -to clk1
```

The violation message appears because the [set_clock_uncertainty](#) constraint has been defined between the two asynchronous clocks.

Example 2Test Case Files Not Available

This example shows an example that is reported by the *SCG03* rule. In the following SDC snippet, the SCG03 rule reports violation because [set_clock_uncertainty](#) is specified between clocks that have [set_clock_groups](#) defined.

```
create_clock -name C1 -period 10 clk1
create_clock -name C2 -period 13 clk2
set_clock_groups -async -group {C1} -group {C2}
set_clock_uncertainty 1 -from C1 -to C2
```

Default Severity Label

Error

Rule Group

SCGRules

Reports and Related Files

None

Clock Group Rules

SCG04

Reports clocks, which have a `set_clock_groups` defined, share a harmonic relationship

When to Use

Use this rule for RTL and netlists.

Description

The *SCG04* rule reports a violation if clocks share a harmonic relationship and a *set_clock_groups* constraint has been defined between the clocks.

Refer to *Example 1 - Harmonic Relationship of Clocks* for more information on harmonic relationships.

Parameter(s)

- *tc_c2c_max_cycles*: Default value is 50. Set the value to any positive integer to set the maximum value of the number of cycles in which two clocks can be synchronized.

Constraint(s)

- *set_clock_groups* (Mandatory): Use to specify the clock groups that are mutually exclusive or asynchronous so that the paths between these clocks are not considered during timing analysis.

Messages and Suggested Fix

The following message appears when there is a harmonic relationship between clocks:

```
[ERROR] set_clock_groups is specified between clocks <clk1> and <clk2> sharing a harmonic relationship
```

Potential Issues

The *set_clock_groups* constraint is specified between clocks that share a harmonic relationship.

Consequences of Not Fixing

You can specify any clocks even if they are synchronous to be in different groups. If clocks have harmonic relationships, they are more likely to be

synchronous than asynchronous. Therefore, you should resolve the violation.

How to Debug and Fix

To resolve this violation, either:

- Remove the [set_clock_groups](#) constraint, or
- Reduce the value of the [tc_c2c_max_cycles](#) parameter so that the clocks do not share a harmonic relationship. Refer to [Example 1 - Harmonic Relationship of Clocks](#) for more information on harmonic relationships.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. In this example, the violation message appears because the [set_clock_groups](#) constraint has been defined between the clocks `clk1` and `clk2`, which have a harmonic relationship.

To view the SDC and design files, [View Test Case Files](#).

Example 2

Test Case Files Not Available

The example in this section provide SDC commands that are reported by the *SCG04* rule. The examples pertain to the following SDC snippet:

```
create_clock -name C1 -period 14 clk1
create_clock -name C2 -period 15 clk2
set_clock_groups -async -group {C1} -group {C2}
```

The *SCG04* rule reports a violation because C1 and C2 share a harmonic relationship.

Example 1 - Harmonic Relationship of Clocks

This example describes a harmonic relationship between clocks.

Two clocks exhibit a harmonic relationship if they have the same duty cycles and their clock periods can be synchronized in N cycles, where N

should not be greater than value of the *tc_c2c_max_cycles* parameter.

In this example, the value of the *tc_c2c_max_cycles* parameter is set to 50. The clocks have the following periods:

Clock	Period	Cycles Required for Clock to be Synchronized
C1	14	15
C2	15	14

Since the cycles required for the clocks to be synchronized is less than 50, clocks C1 and C2 share a harmonic relationship.

Default Severity Label

Error

Rule Group

SCGRules

Reports and Related Files

None

SCG05

Reports multiple clock definitions at the same node as being physically exclusive

When to Use

Use this rule for RTL and netlists.

Description

The *SCG05* rule reports a violation for clocks, which do not have [set_clock_groups](#) defined, but are defined the same port/pin.

Parameter(s)

None

Constraint(s)

- [set_clock_groups](#) (Mandatory): Use to specify the clock groups that are mutually exclusive or asynchronous so that the paths between these clocks are not considered during timing analysis.

Messages and Suggested Fix

The following message appears when you need define [set_clock_groups](#) between clocks:

```
[ERROR] Clocks <clk1> and <clk2> defined on same port/pin  
<name> should have set_clock_groups (-<option>) constraint  
defined between them
```

How to Debug and Fix

To resolve this violation, either:

- Change the definition of the clocks in the SDC file, or
- Define the [set_clock_groups](#) constraint between the clocks.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. In this example, the violation message appears because `clk1` and `clk11` clocks are defined on the same port, but there is no [set_clock_groups](#) defined between these two clocks with the `-physically_exclusive` argument.

To view the SDC and design files, [View Test Case Files](#).

Example 2Test Case Files Not Available

This example describes when the `SCG05` rule reports a violation. In the following SDC snippet, the clocks `rclk` and `tclk` do not have a [set_clock_groups](#) constraint defined between them, but they are defined on the same pin `clkgen/p11/clk1x`. Therefore, this rule reports a violation.

```
create_clock -name "rclk" -period $CK1_PERIOD_1 [get_pins  
clkgen/p11/clk1x]
```

```
create_clock -name "tclk" -period $CK1_PERIOD_2 [get_pins  
clkgen/p11/clk1x]
```

To resolve the violation, define the [set_clock_groups](#) constraint between `rclk` and `tclk`.

Default Severity Label

Error

Rule Group

SCGRules

Reports and Related Files

None

Display Information Rules

The following display information rules are available:

Rule	Purpose
<i>Show_Case_Analysis</i>	Highlights case-analysis settings
<i>Show_Clock_Propagation</i>	Shows clock propagation for the port/pin

Show_Case_Analysis

Highlights case-analysis settings

When to Use

Use the *Show_Case_Analysis* rule to debug those rules where the behavior is sensitive to case-analysis settings. Use this rule for debugging the violations reported by other rules that are related to constant logic value.

This rule is applicable to the RTL, Pre-layout, Post-layout phases.

Description

The *Show_Case_Analysis* rule highlights the case analysis values and power/ground information on design objects. You can view this information in the Schematic Viewer of Console GUI.

The rule uses the [set_case_analysis](#) SDC constraints and the power-ground information from the design to annotate the simulation value, such as 0 or 1, for each accessible net by simulating the design.

Parameter(s)

- [tc_show_sca_on_terminals](#): Default is no and the schematic displays the [set_case_analysis](#) attributes only on nets. Set this parameter to yes to display the [set_case_analysis](#) attributes on nets and terminals.

Constraint(s)

SDC

- [set_case_analysis](#): Use specify that a port or pin is at a constant logic value 1 or 0.

Messages and Suggested Fix

The *Show_Case_Analysis* rule generates the following message:

```
[INFO] Case-analysis information has been propagated for  
module <module-name> (mode: <mode-name>)
```

NOTE: *The mode name is shown in the message only if you have specified the mode in the SGDC file.*

Potential Issues

Not applicable

Consequences of Not Fixing

Not applicable

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic highlights the constant value propagation that is due to VSS/VDD or the [set_case_analysis](#) constraint. The schematic also displays the text attribute 0(F) or 1(F) for the nodes where a constant value is applied by the [set_case_analysis](#) constraint. For all other cases, either 0 or 1 is displayed as text attributes.

In the Incremental Schematic window, while debugging a rule, you can view the case analysis settings along with the violation of other rules.

For this, you can do any of the following:

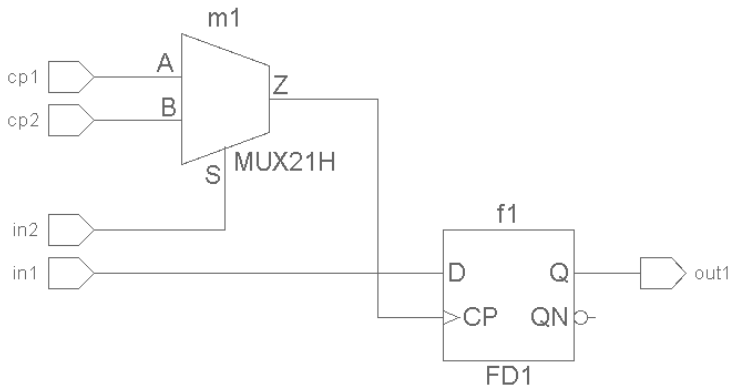
- Select the rule to be debugged and then double-click the violation message of the *Show_Case_Analysis* rule while holding down the <Ctrl> key.
- Select the rule violation and open the Incremental Schematic window.
- Click the **Show Case Analysis** option from the **Edit** menu.

For details on working with multiple messages, refer to *SpyGlass Explorer User Guide*.

Example Code and/or Schematic

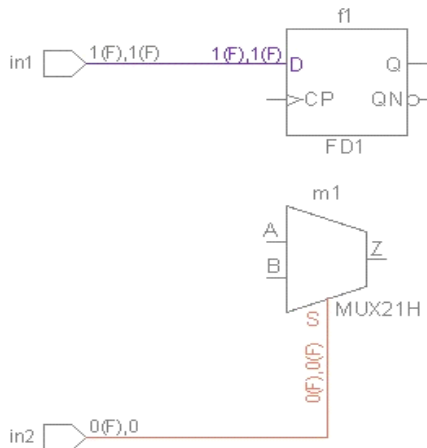
Example 1[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The violation message generated by this rule indicates that the [set_case_analysis](#) information has been propagated through the design.

To view the propagated values, click the **Incremental Schematic** icon. The incremental schematic appears with the [set_case_analysis](#) values.



Example 2Test Case Files Not Available

Refer to the SpyGlass Constraints Methodology Guide.

Default Severity Label

Info

Rule Group

Constraints

Reports and Related Files

None

Show_Clock_Propagation

Shows clock propagation for the port/pin

When to Use

To debug violation messages generated from other rules that are related to propagation of clocks. This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Show_Clock_Propagation* rule provides information about the clock propagation for all ports/pins in the clock paths. The rule annotates the clock information on all the nodes of the path, from the start point, where the clock is defined, to its end points, such as flip-flop clock pin, blackbox instance pin, tristate enable pin, blocked terminal, or output port.

In addition, the pins that are blocked by constant values and pins that are disabled by the [set_disable_timing](#) constraints are shown in the schematic.

This rule also considers the effect of the [set_clock_sense/set_sense](#) constraint.

Parameter(s)

None

Constraint(s)

- [set_clock_sense/set_sense](#) (Optional): Use to specify the clock propagation conditions.

Messages and Suggested Fix

The *Show_Clock_Propagation* rule generates the following message:

[INFO] Clock information marked on all ports/pins in clock paths for module <module-name> (mode: <mode-name>)

NOTE: *The mode name is shown in the message only if you have specified the mode in the SGDC file.*

Potential Issues

Display Information Rules

Not applicable

Consequences of Not Fixing

Not applicable

How to Debug and Fix

View the incremental schematic of the violation message.

The names, period and/or waveform of clocks, which are reaching the design object are visible in the modular and incremental schematic.

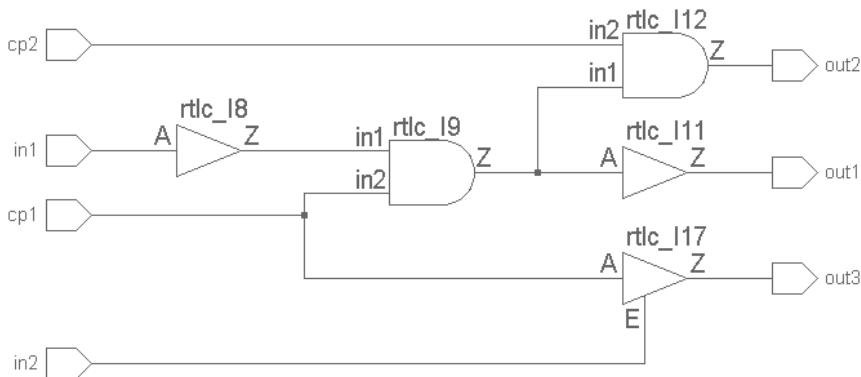
When no clock is found on any object in the clock path, the path might be blocked or disabled. Alternatively, the clock might be stopped by the [set_clock_sense/set_sense](#) constraint, a black box, or the enable pin of a tristate logic.

Example Code and/or Schematic

Example 1

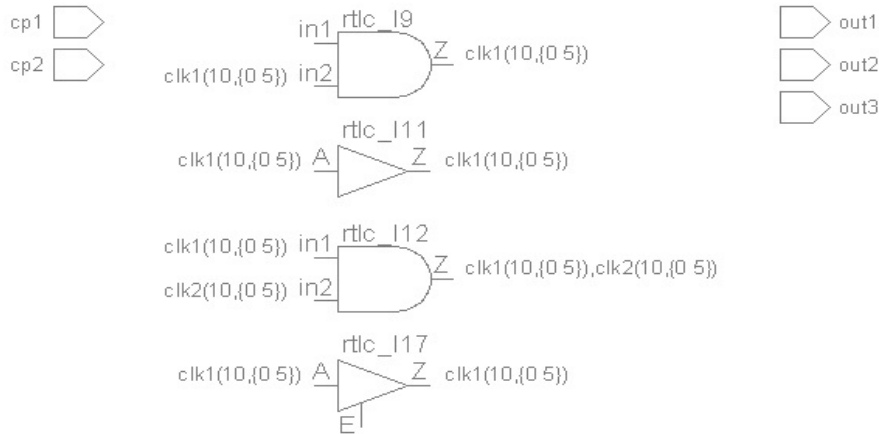
[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The violation message generated by this rule indicates that the propagated clocks are marked in all ports/pins.

To view the information, click the **Incremental Schematic** icon. The incremental schematic appears.



Example 2

Test Case Files Not Available

In this example, the *Show_Clock_Propagation* rule is being used to determine the clock reaching the out_reg/CP pin.

Consider the following SDC file snippet:

```
create_clock -name C1 -period 30 -waveform { 0.0 5.0 }
[get_ports clk1] -comment "clock is C1"

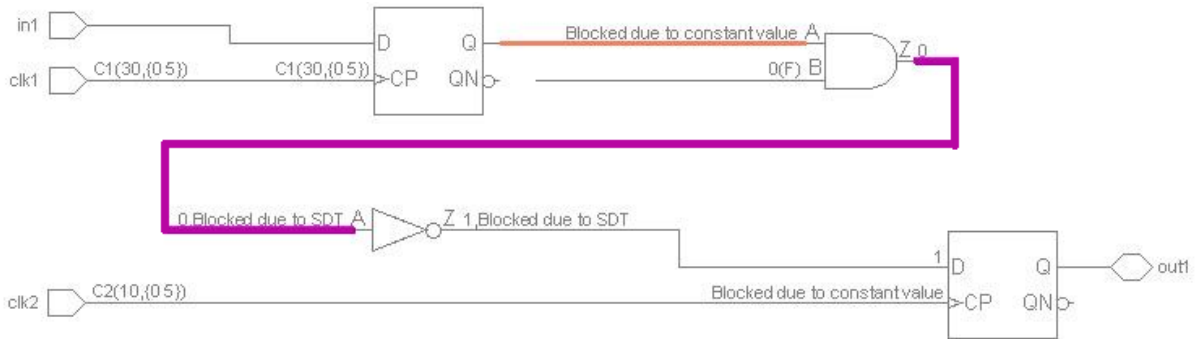
create_clock -name C2 -period 10 -waveform { 0.0 5.0 }
[get_ports clk2] -comment "clock is C2"

set_disable_timing -from A -to Z [get_lib_cells {lsi_10k/IV}
]

set_disable_timing iv1/Z
set_disable_timing iv2/Z
```

Display Information Rules

The names, period and/or waveform of clocks, which are reaching the design object are visible in the modular and incremental schematic, as shown in the following schematic.



Default Severity Label

Info

Rule Group

Constraints

Reports and Related Files

None

Domain Rules

The Domain Rules Group `DomainRules` contains the following rules:

Rule	Purpose
<i>DomainAnalysis</i>	Extracts domain information from SDC constraints
<i>DomainError</i>	Infers pair-wise clock domain information from SDC files and flags errors in clock domain information
<i>DomainInfo</i>	Infers pair-wise clock domain information from SDC files
<i>Domain_SGDC_Consis</i>	Reports inconsistency between the domain specified in the SGDC file and the domain extracted from the SDC constraints

DomainAnalysis

Populates the domain information to be used by other constraint rules

When to Use

The *DomainAnalysis* rule is a pre-requisite setup rule that is always run when one or more SpyGlass Constraints policy rules are selected to run. It identifies the rule(s) that require clock domain information and then checks whether the *clock_group* information for that rule(s) is specified in the SGDC file for the current design. If the *clock_group* information is not specified in the SGDC file, the *DomainAnalysis* rule extracts the domain information from the *set_clock_groups*, *set_false_path*, and *set_clock_uncertainty* constraints specified in the SDC file.

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *DomainAnalysis* rule populates the domain information to be used by other SpyGlass Constraints rules. The rules that require clock domain information are *Clk_Gen05*, *Clk_Lat03*, *Clk_Uncert03*, *False_Path07*, *False_Path08*, and *Domain_SGDC_Consis*.

The *DomainAnalysis* rule considers the source-generated and harmonic relationship between two clocks while extracting domain information and uses the *clock_group* constraint, if specified.

Rule Exceptions

This rule does not consider *set_false_path* constraints having through list.

Parameter(s)

- *tc_domain_mode*: Default is STA. Set the value to specify the mode for inferring domains. Other possible values are: TC_STA, STRICT, and SGDC.
- *tc_virtual_clock_prefixes*: Default is unspecified. Set the value of this parameter to the clock name prefix for virtual clock names for the domain flow.

- *tc_virtual_clock_suffixes*: Default is unspecified. Set the value of this parameter to the clock name suffix for virtual clock names for the domain flow.

Constraint(s)

- *clock_group* (Optional): Use this constraint to specify the domain relationship between the clocks.
- *set_clock_groups* (Optional): Use this constraint to specify the clock groups that are mutually exclusive or asynchronous so that the paths between these clocks are not considered during timing analysis.
- *set_false_path* (Optional): Use this constraint to specify the paths in a design that are to be marked as false, so that they are not considered during timing analysis.
- *set_clock_uncertainty* (Optional): Use this constraint to specify uncertainty of specified clock networks.

Messages and Suggested Fix

There are no messages.

Example Code and/or Schematic

None

Default Severity Label

Info

Rule Group

DomainRules

Reports and Related Files

None

DomainError

Reports clock domain errors extracted from SDC commands

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *DomainError* rule reports an error depending on the mode. The following table displays the checks performed when this rule is run in the STA (Default), TC_STA, STRICT, and SGDC modes. You can set the mode through the *tc_domain_mode* parameter. Refer to the following sections for more information on the checks performed in each mode.

- [Check Performed for STA \(Default\), STRICT, and SGDC](#)
- [Checks Performed for TC_STA](#)

In addition, refer to the [Specifying Domains](#) and [Rule Exceptions](#) sections.

Check Performed for STA (Default), STRICT, and SGDC

The following table describes the checks performed in the STA, Strict, and SGDC modes. For information on debugging the message generated, refer to the More Information column.

Check Performed	Mode	More Information
Either a set_clock_groups or a set_false_path constraint overrides set_clock_uncertainty despite synchronous phase relationship between master-generated clocks.	STA	Refer to Message 17 .
No domain assigned to a clock	STA	Refer to Message 16 (view the spreadsheet generated).

Check Performed	Mode	More Information
There exists a source-generated relationship/ <i>set_clock_uncertainty</i> between C1, C2 and C1, C3 and either a <i>set_clock_groups</i> or a <i>set_false_path</i> constraint is specified between C2 and C3. This leads to a conflict, which is reported, and no domain is assigned.	STA	Refer to Message 15 (view the spreadsheet generated).
Both <i>set_false_path</i> and <i>set_clock_uncertainty</i> constraints are specified between interacting clock pairs	STA	Refer to Message 21 .
<i>set_clock_groups</i> not specified for clocks.	Strict	Refer to Message 19 .
<i>set_clock_groups</i> overrides synchronous phase relationship between master-generated clocks	Strict	Refer to Message 18 .
Missing <i>clock_group</i> SGDC constraint for clocks	SGDC	Refer to Message 20 .

Checks Performed for TC_STA

The following table describes the checks performed in the TC_STA mode. For information on debugging the message generated, refer to the More Information column.

Check Performed	More Information
No constraint is specified between the two interacting clocks.	Refer to Message 1 .
A <i>set_false_path</i> constraint is specified from the first clock to the second clock, but not from the second clock to the first clock and a crossing exists from the second clock to the first clock.	Refer to Message 2 .

Domain Rules

Check Performed	More Information
A <i>set_clock_uncertainty</i> constraint is specified from the first clock to the second clock, but not from the second clock to the first clock and a crossing exists from the second clock to the first clock.	Refer to Message 3 .
Either a <i>set_clock_groups</i> or a <i>set_false_path</i> constraint is specified between the source and generated clock pair having a clock crossing between them.	Refer to Message 4 and Message 5 .
Either a <i>set_clock_groups</i> or a <i>set_false_path</i> constraint is specified between the two interacting clocks and these two clocks exhibit a harmonic relation.	Refer to Message 12 and Message 13 .
A <i>set_clock_groups/set_false_path</i> , and <i>set_clock_uncertainty</i> constraints are specified between the two interacting clocks.	Refer to Message 6 and Message 7 .
A <i>set_clock_groups/set_false_path</i> constraint is specified for first clock (C1) and second clock (C2) and a <i>set_clock_uncertainty</i> constraint is specified from C2 to C1 (if a clock crossing exists from C1 to C2 and from C2 to C1).	Refer to Message 8 and Message 9 . Example 5 - Cyclic Dependency and Overridden set_false_path
There exists a source-generated relationship/ <i>set_clock_uncertainty</i> between C1 and C2 and C1 and C3 and either a <i>set_clock_groups</i> or a <i>set_false_path</i> constraint is specified between C2 and C3. Then, from the source-generated relationship/ <i>set_clock_uncertainty</i> , clocks C1, C2 and C3 are resolved as synchronous so <i>set_clock_groups</i> or <i>set_false_path</i> constraint is not considered and clocks C1, C2 and C3 are assumed to be synchronous in TC_STA and STA.	Refer to Example 3 - Domain Matrix and Message 10/Message 11/Message 15 (view the spreadsheet generated).

Check Performed	More Information
There exists a source-generated relationship/ set_clock_uncertainty between C1 and C2 and either a set_clock_groups or a set_false_path constraint is specified between C1 and C2. Then, from the source-generated relationship overrides set_clock_groups or set_false_path , clocks C1, C2 are resolved as synchronous.	Refer to Message 10/Message 11 .
No domain assigned to a clock	Refer to Message 16 (view the spreadsheet generated).

Rule Exceptions

This rule does not consider [set_false_path](#) constraints having -through list.

Parameter(s)

- [tc_c2c_max_cycles](#): Default value is 50. Set the value to any positive integer to set the maximum value of the number of cycles in which two clocks can be synchronized.
- [tc_domain_mode](#): Default is STA. Set the value to specify the mode for inferring domains. Other possible values are: TC_STA, STRICT, and SGDC.
- [tc_virtual_clock_prefixes](#): Default is unspecified. Set the value of this parameter to the clock name prefix for virtual clock names for the domain flow.
- [tc_virtual_clock_suffixes](#): Default is unspecified. Set the value of this parameter to the clock name suffix for virtual clock names for the domain flow.

Constraint(s)

SGDC

- [clock_group](#) (Optional): Use this constraint to specify the domain relationship between clocks.

SDC

- *set_clock_groups* (Optional): Use to specify the clock groups that are mutually exclusive or asynchronous so that the paths between these clocks are not considered during timing analysis.

While inferring domain, *logically_exclusive*, *physically_exclusive*, and *asynchronous* argument are considered only in the TC_STA mode. However, in all other modes, only the *asynchronous* argument is considered. You can specify the mode by setting the *tc_domain_mode* parameter.

- *set_false_path* (Optional): Use to specify the paths in a design that are to be marked as false, so that they are not considered during timing analysis.
- *set_clock_uncertainty* (Optional): Use to specify uncertainty of specified clock networks.

More Information DomainError

For information on the DomainError rule, refer to the following sections:

- *Messages and Suggested Fix*
- *Example Code and/or Schematic*
- *Default Severity Label*
- *Rule Group*
- *Reports and Related Files*

DomainError (contd.)

Messages and Suggested Fixes and Examples

This section contains the following information pertaining to the *DomainError* rule:

- [Messages and Suggested Fix](#)
- [Example Code and/or Schematic](#)
- [Default Severity Label](#)
- [Rule Group](#)
- [Reports and Related Files](#)

Messages and Suggested Fix

Message 1

The following message appears when none of the [set_false_path](#), [set_clock_groups](#), or [set_clock_uncertainty](#) constraints have been specified from clock `<clk1-name>` to clock `<clk2-name>`:

```
[DomainError_01][ERROR] Missing constraint between interacting clock pairs "<clk1-name>" and "<clk2-name>", assumed as synchronous
```

Potential Issues

This violation arises if no constraint is specified between interacting clocks.

Consequences of Not Fixing

If you do not fix this violation, other tools can assume the clock relationship differently as for the SpyGlass Constraints solution these clocks are assumed as synchronous.

How to Debug and Fix

Ensure that you have specified the valid constraint between the interacting clock pairs.

Message 2

The following message appears when only a [set_false_path](#) constraint has been specified from source clock `<clk1-name>` to its generated clock `<clk2-name>`:

[DomainError_02] [ERROR] Missing constraint from "<clk1-name>" to "<clk2-name>", but `set_false_path` specified from "<clk2-name>" to "<clk1-name>", assumed as asynchronous

Potential Issues

This violation arises if no constraint is specified between the interacting clock pair.

Consequences of Not Fixing

If you do not fix this violation, other tools can assume the clock relationship differently as for the SpyGlass Constraints solution these clocks are assumed as asynchronous.

How to Debug and Fix

Ensure that you have specified a valid constraint for the interacting clock pair.

Message 3

The following message appears when only a `set_clock_uncertainty` constraint has been specified from source clock <clk1-name> to its generated clock <clk2-name>:

[DomainError_03] [ERROR] Missing constraint from "<clk1-name>" to "<clk2-name>", but `set_clock_uncertainty` specified from "<clk2-name>" to "<clk1-name>", assumed as synchronous

Potential Issues

This violation message arises if no constraint is specified between the interacting clock pair.

Consequences of Not Fixing

If you do not fix this violation, other tools can assume the clock relationship differently as for the SpyGlass Constraints solution these clocks are assumed as synchronous.

How to Debug and Fix

Ensure that a valid constraint is specified for the interacting clock pair.

Message 4

The following message appears when one clock <clk1-name> is generated from another clock <clk2-name> and the `set_clock_groups` constraint is specified between them:

[DomainError_04] [ERROR] Clocks "<clk1-name>" and "<clk2-name>" are considered synchronous because "<clk1-name>" is generated from "<clk2-name>" even though `set_clock_groups` is specified between them

Potential Issues

This violation arises if the `set_clock_groups` constraint is specified between the interacting clock pair which has source-generated relationship.

Consequences of Not Fixing

If you do not fix this violation and the interacting clocks have source-generated relation, these clocks are assumed as synchronous. The `set_clock_groups` constraint specified is not considered for domain extraction.

How to Debug and Fix

Ensure that you have removed the `set_clock_groups` constraint between the reported clocks.

Message 5

The following message appears when one clock <clk1-name> is generated from another clock <clk2-name> and the `set_false_path` constraint is specified between them:

[DomainError_05] [ERROR] Clocks "<clk1-name>" and <clk2-name> are considered synchronous because "<clk1-name>" is generated from "<clk2-name>" even though `set_false_path` is specified between them

Potential Issues

This violation arises if `set_false_path` constraint is specified between the interacting clock pair which has source-generated relationship.

Consequences of Not Fixing

If you do not fix this violation and the interacting clocks have source-generated relation, these clocks are assumed as synchronous. The `set_false_path` constraint specified is not considered for domain extraction.

How to Debug and Fix

Ensure that you have removed the `set_false_path` constraint between the reported clocks.

Message 6

The following message appears when two clocks `<clk1-name>` and `<clk2-name>` have both `set_clock_groups` and `set_clock_uncertainty` constraints and the `set_clock_groups` constraint overrides the `set_clock_uncertainty` constraint:

```
[DomainError_08][INFO] Cl ocks "<clk1-name>" and "<clk2-name>"  
are considered asynchronous because set_clock_groups  
overrides set_clock_uncertainty specified between them
```

Potential Issues

This violation arises if both `set_clock_groups` and `set_clock_uncertainty` constraints are specified between the interacting clock pairs. Only one constraint should be specified as the `set_clock_groups` constraint is used to specify the asynchronous relationship while the `set_clock_uncertainty` constraint is used to specify the synchronous relationship.

Consequences of Not Fixing

If you do not fix this violation, the SpyGlass Constraints solution is considering these clocks as asynchronous, other tools can assume it differently.

How to Debug and Fix

Ensure that you remove either `set_clock_groups` or `set_clock_uncertainty` command.

Message 7

The following message appears when two clocks `<clk1-name>` and `<clk2-name>` have both `set_false_path` and `set_clock_uncertainty` constraints and the `set_false_path` constraint override the `set_clock_uncertainty` constraint:

```
[DomainError_09][ERROR] Cl ocks "<clk1-name>" and "<clk2-name>"  
are considered asynchronous because set_false_path overri des  
set_clock_uncertainty speci fied between them
```

Potential Issues

This violation arises if both `set_false_path` and `set_clock_uncertainty` constraints are specified between the interacting clock pairs. Only one constraint should be specified as the `set_false_path` constraint, it is used to specify the asynchronous relationship while the `set_clock_uncertainty` is used

to specify the synchronous relationship.

Consequences of Not Fixing

If you do not fix this violation, the SpyGlass Constraints solution is considering these clocks as asynchronous, other tools can assume it differently.

How to Debug and Fix

Ensure that you remove either [set_false_path](#) or [set_clock_uncertainty](#) command.

Message 8

The following message appears when two clocks `<clk1-name>` and `<clk2-name>` have a [set_clock_groups](#) constraint specified between them and a [set_clock_uncertainty](#) constraint is specified from `<clk2-name>` to `<clk1-name>`:

```
[DomainError_10][INFO] Cl ocks "<cl k1-name>" and "<cl k2-name>"  
are considered asynchronous because set_clock_groups is  
speci fied between the two clocks even though  
set_clock_uncertainty is speci fied from clock "<cl k2-name>" to  
"<cl k1-name>"
```

Potential Issues

This violation arises if both [set_clock_groups](#) and [set_clock_uncertainty](#) constraints are specified between the interacting clock pairs. Only one constraint should be specified as the [set_clock_groups](#) constraint is used to specify the asynchronous relationship while the [set_clock_uncertainty](#) constraint is used to specify the synchronous relationship.

Consequences of Not Fixing

If you do not fix this violation, the SpyGlass Constraints solution is considering these clocks as asynchronous, other tools can assume it differently.

How to Debug and Fix

Ensure that you remove either [set_clock_groups](#) or [set_clock_uncertainty](#) command.

Message 9

The following message appears when two clocks `<clk1-name>` and `<clk2-name>` have a [set_false_path](#) constraint specified between them

and a *set_clock_uncertainty* constraint is specified from `<clk2-name>` to `<clk1-name>`

[DomainError_11][ERROR] Clocks "`<clk1-name>`" and "`<clk2-name>`" are considered asynchronous because *set_false_path* is specified between the two clocks even though *set_clock_uncertainty* is specified from clock "`<clk2-name>`" to "`<clk1-name>`"

Potential Issues

This violation arises if the clocks are interacting in both ways, for one direction *set_false_path* is specified and for another *set_clock_uncertainty* is specified. Only one constraint should be specified as the *set_false_path* constraint is used to specify the asynchronous relationship while the *set_clock_uncertainty* is used to specify the synchronous relationship.

Consequences of Not Fixing

If you do not fix this violation, the SpyGlass Constraints solution is considering these clocks as asynchronous, other tools can assume it differently.

How to Debug and Fix

Ensure that you remove either *set_false_path* or *set_clock_uncertainty* command.

Message 10

The following message appears when two clocks `<clk1-name>` and `<clk2-name>` have been declared as synchronous based on the priority given the constraints even though a *set_clock_groups* constraint is specified between them:

[DomainError_12][ERROR] Based on priority of constraints, clocks `<clk1-name>` and `<clk2-name>` have already been resolved as synchronous even though *set_clock_groups* is specified between them

Potential Issues

This violation arises if the *set_clock_groups* constraint is specified between the interacting clock pairs, but these clocks have circular dependency due to which it should not be treated as asynchronous.

Consequences of Not Fixing

If you do not fix this violation, the SpyGlass Constraints solution is

considering these clocks as synchronous, other tools can assume it differently.

How to Debug and Fix

Ensure that you remove the [set_clock_groups](#) constraint specified between the interacting clocks reported.

Message 11

The following message appears when two clocks `<clk1-name>` and `<clk2-name>` have been declared as synchronous based on the priority given the constraints even though a [set_false_path](#) constraint is specified between them:

```
[DomainError_13][ERROR] Based on priority of constraints, clocks "<clk1-name>" and "<clk2-name>" have already been resolved as synchronous even though set_false_path is specified between them
```

Potential Issues

This violation arises if the [set_false_path](#) constraint is specified between the interacting clock pairs, but these clocks have circular dependency due to which it should not be treated as asynchronous.

Consequences of Not Fixing

If you do not fix this violation, the SpyGlass Constraints solution is considering these clocks as synchronous, other tools can assume it differently.

How to Debug and Fix

Ensure that you remove the [set_false_path](#) constraint specified between the interacting clocks reported.

Refer to [Example 5 - Cyclic Dependency and Overridden set_false_path](#) for more information.

Message 12

The following message appears when two clocks `<clk1-name>` with period `<period-1>` and `<clk2-name>` with `<period-2>` share a harmonic relationship and have a [set_clock_groups](#) constraint specified between them:

```
[DomainError_06][ERROR] set_clock_groups is specified between clocks <clk1-name> (period : <period-1>) and <clk2-name>
```

(period : <period-2> sharing a harmonic relationship, for every N cycles of <clk1-name>/<clk2-name> (parameter tc_c2c_max_cycles : <value>), assumed as asynchronous

Potential Issues

This violation arises if the [set_clock_groups](#) constraint is specified between the interacting clock pairs but these clocks exhibit a harmonic relation also.

Consequences of Not Fixing

If you do not fix this violation, the SpyGlass Constraints solution is considering these clocks as asynchronous, other tools can assume it differently.

How to Debug and Fix

Ensure that you either remove the [set_clock_groups](#) constraint or modify the value of the [tc_c2c_max_cycles](#) rule parameter.

Message 13

The following message appears when two clocks <clk1-name> with period <period-1> and <clk2-name> with period <period-2> share a harmonic relationship and have a [set_false_path](#) constraint specified between them:

```
[DomainError_07][ERROR] set_false_path is specified between
clocks <clk1-name> (period : <period-1>) and <clk2-name>
(period : <period-2>) sharing a harmonic relationship, for
every N cycles of <clk1-name>/<clk2-name> (parameter
tc_c2c_max_cycles : <value 1>), assumed as asynchronous
```

Potential Issues

This violation arises if the [set_false_path](#) constraint is specified between the interacting clock pairs but these clocks exhibit a harmonic relation also.

Consequences of Not Fixing

If you do not fix this violation, the SpyGlass Constraints solution is considering these clocks as asynchronous, other tools can assume it differently.

How to Debug and Fix

Ensure that you either remove the [set_false_path](#) constraint or modify the value of the [tc_c2c_max_cycles](#) rule parameter.

Message 14

The following message appears after clocks domain computation in TC_STA mode:

```
[INFO] Domain Matrix for design "<design-name>", based on the individual constraints defined in the SDC as inferred in the TC_STA mode, <file-name>, <line no. >
```

For debugging information, refer to [How to Debug and Fix](#).

Message 15

The following message appears when some clocks have cyclic dependency and the same domain is assigned in TC_STA mode while no domain is assigned in STA mode:

```
[ERROR] For design "<design-name>", <no. of dependencies> cyclic dependency exists between the clocks, same domain is assumed. Report is generated, <file-name>, <line no. >
```

For debugging information, refer to [How to Debug and Fix](#).

Message 16

The following message appears, in the STA and TC_STA modes, when the domain could not be assigned to certain clocks:

```
[ERROR] Missing domain for <no. of clocks> clock(s), <file name>, <line no. >
```

Potential Issues

This violation messages explicitly states the potential issues.

Consequences of Not Fixing

If you do not fix this violation, the SpyGlass Constraints solution is considering these clocks as synchronous, other tools can assume it differently.

How to Debug and Fix

Message 14: Double-click the violation messages to view the spreadsheet that shows the domains and the relation between the clocks. Inside the spreadsheet, click a hyperlink to view the clock relation in the SDC file.

Message 15: Double-click the violation messages to view the spreadsheet that shows the clocks having cyclic dependency. Refer to [Example 3 - Domain](#)

[Matrix](#) for further details on how to interpret the information in the generated spreadsheets.

You can also view the spreadsheet by accessing the `tc_top_cyclic_errors_<number>.csv` file located in the `spyglass_reports/constraints` directory.

Message 16: Double-click the violation messages to view the spreadsheet that shows the clocks not being assigned to a domain. Ensure the clock relation is specified in the SDC file. Refer to [Example 4 - Domain Matrix and Missing Domains](#) for further details on how to interpret the information in the generated spreadsheets.

Message 17

The following message appears in STA mode when two clocks are assumed asynchronous:

[INFO/ERROR] Cl ocks <Cl k1-name> and <Cl k2-name> are assumed asynchronous because `set_clock_groups/set_false_path` overrides `set_clock_uncertainty`; Cl ocks <Cl k1-name and <Cl k2-name> are assumed asynchronous because `set_clock_groups/set_false_path` overrides synchronous phase relationship between master-generated cl ocks

NOTE: *The severity of this message is INFO if `set_clock_groups` is the overriding constraint. It is ERROR if `set_false_path` is the overriding constraint.*

Potential Issues

This violation arises if both [set_false_path/set_clock_groups](#) and [set_clock_uncertainty](#) constraints are specified between the interacting clock pairs. Only one constraint should be specified as the [set_false_path/set_clock_groups](#) constraints are used to specify the asynchronous relationship while the [set_clock_uncertainty](#) is used to specify the synchronous relationship.

In addition, master-generated clock relationship should not exist when the [set_false_path/set_clock_groups](#) constraints are specified.

Consequences of Not Fixing

If you do not fix this violation, the SpyGlass Constraints solution is considering these clocks as asynchronous, other tools can assume it differently.

How to Debug and Fix

Ensure that you remove [set_false_path/set_clock_groups](#), or remove the [set_clock_uncertainty](#) constraint.

Message 18

The following message appears in STRICT mode when two clocks are assumed asynchronous:

[ERROR] Clocks <Clk1-name> and <Clk2-name> are assumed asynchronous because [set_clock_groups](#) overrides synchronous phase relationship between master-generated clocks

Potential Issues

This violation arises if both [set_clock_groups](#) and master-generated clock relationship is specified between the interacting clock pairs. The [set_clock_groups](#) constraint is used to specify the asynchronous relationship while the master-generated clock relationship indicates the synchronous relationship. The master-generated clock relationship should not exist when the [set_clock_groups](#) constraints are specified.

Consequences of Not Fixing

If you do not fix this violation, the SpyGlass Constraints solution is considering these clocks as asynchronous, other tools can assume it differently.

How to Debug and Fix

Ensure that you remove the [set_clock_groups](#) constraint.

Message 19

The following message appears in STRICT mode when the [set_clock_group](#) is not specified for a specific number of clocks:

[FATAL] No [set_clock_group](#) constraints specified for <no. of clocks> clock(s)

Potential Issues

This violation messages explicitly states the potential issues.

Consequences of Not Fixing

If you do not fix these violations, SpyGlass will not run further.

How to Debug and Fix

Double-click the message to view the clocks that do not have the [set_clock_groups](#) constraint. Update the SDC file for these clocks.

Message 20

The following message appears in SGDC mode when the [clock_group](#) constraint is not specified for a specific number of clocks:

```
[ERROR] Missing clock_group constraint for <no. of clocks>  
clock(s)
```

Potential Issues

This violation messages explicitly states the potential issues.

Consequences of Not Fixing

If you do not fix this violation, the SpyGlass Constraints solution will not consider the clocks in any domain.

How to Debug and Fix

Double-click the violation message to view the list of clocks that do not have the [clock_group](#) constraint. Update the SGDC file.

Message 21

The following message appears in STA mode when two clocks are assumed synchronous:

```
[ERROR] Clocks <Clk1-name> and <Clk2-name> are assumed  
synchronous because set_clock_uncertainty from <Clk1-name> to  
<Clk2-name> overrides set_false_path from <Clk1-name> to <Clk2-  
name>
```

Potential Issues

This violation message appears if both [set_false_path](#) and [set_clock_uncertainty](#) constraints are specified between interacting clock pairs.

Only one constraint should be specified because the [set_clock_uncertainty](#) constraints are used to specify a synchronous relationship, while the [set_false_path](#) constraints are used to specify an asynchronous relationship.

Consequences of Not Fixing

If you do not fix this violation, the SpyGlass Constraints solution considers these clocks as synchronous; Other tools can assume it differently.

How to Debug and Fix

Either remove the [set_false_path](#) constraint or the [set_clock_uncertainty](#) constraint.

Example Code and/or Schematic

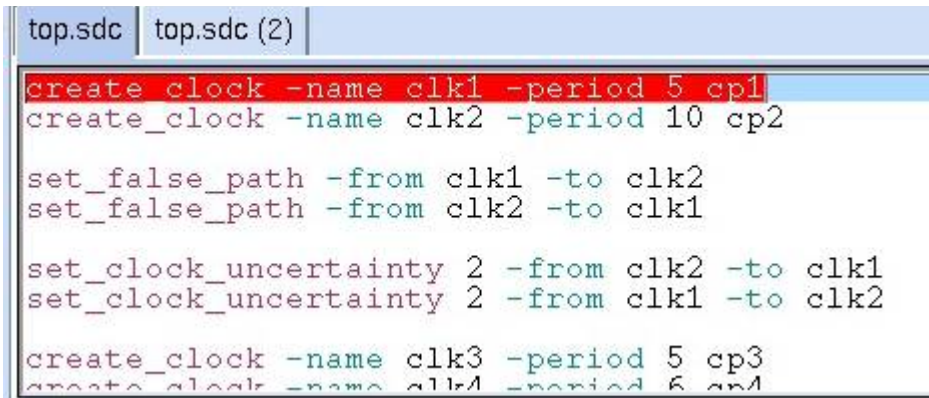
This section contains the following examples:

- [Example 1 - Clock Domain Error](#)
- [Example 2 - Invalid Specification of set_false_path](#)
- [Example 3 - Domain Matrix](#)
- [Example 4 - Domain Matrix and Missing Domains](#)
- [Example 5 - Cyclic Dependency and Overridden set_false_path](#)

Example 1 - Clock Domain Error

[View Test Case Files](#)

This example shows a clock domain error caused by [set_false_path](#) and the [set_clock_uncertainty](#) constraints in top.sdc. The top.sdc file is shown in the following screen shot:



```
top.sdc | top.sdc (2) |
create_clock -name clk1 -period 5 cp1
create_clock -name clk2 -period 10 cp2

set_false_path -from clk1 -to clk2
set_false_path -from clk2 -to clk1

set_clock_uncertainty 2 -from clk2 -to clk1
set_clock_uncertainty 2 -from clk1 -to clk2

create_clock -name clk3 -period 5 cp3
create_clock -name clk4 -period 6 cp4
```

Example 2 - Invalid Specification of `set_false_path`

Test Case Files Not Available

In this example, the mode is TC_STA.

Consider two clocks; clock C1 with a period of 10 and clock C2 with a period of 25 that have a `set_false_path` constraint specified between them. Let the value of the `tc_c2c_max_cycles` rule parameter be set to 50.

In this example, the clock C1 will be harmonic after 5 cycles and the clock C2 will be harmonic after 2 cycles. Therefore, both the clocks exhibit a harmonic relation as value of the `tc_c2c_max_cycles` parameter is greater than 5 (the maximum cycles required for the two clocks to be harmonic).

The following message will be generated for the above example

```
set_false_path is specified between clocks C1 (period : 10) and
C2 (period : 25) sharing a harmonic relationship, for every 5
cycles of C1 (parameter tc_c2c_max_cycles : 50), assumed as
asynchronous
```

Example 3 - Domain Matrix

Test Case Files Not Available

In this example, the mode is TC_STA.

For the following snippet, the `DomainError` rule reports violation messages.

```
create_clock -name C1 -period 10 clk1
create_generated_clock -name C2 -divide_by 1 -source clk1 i1/
Z
create_generated_clock -name C3 -divide_by 1 -source clk1 i2/
Z

set_clock_groups -asynchronous -group {C2} -group {C3}
```

After you run this rule, the following spreadsheets are also generated.

Domain Matrix Spreadsheet

The Domain Matrix spreadsheet displays the relation between the clocks and the domain assigned to each clock. When there is no domain assigned to a clock, the corresponding domain cell is empty. For example, C2 is generated from C1 and C2 has a harmonic relationship with C1. You can determine this by referring to cell E1. Click the hyperlinks to see the reference in the SDC file.

	A	B	C	D	E	F
	Clock Name	Domain	Filename:Line	C1	C2	C3
1	C1	d1	top.sdc:1	NA	\$(GEN,HMR)	\$(GEN,HMR)
2	C2	d1	top.sdc:2	\$(GEN)	NA	S(SCG,HMR)
3	C3	d1	top.sdc:3	\$(GEN)	\$(SCG)	NA

In addition, the relation between C2 and C3 is defined to be SYNC because the source-generated has a higher priority than [set_clock_groups](#).

Example 4 - Domain Matrix and Missing Domains

Test Case Files Not Available

In this example, the mode is TC_STA.

This example illustrates the Domain Matrix spreadsheet for a design that has clocks, which have not been assigned to a domain.

	A	B	C	D	E	F	G	H
	Clock Name	Domain	Filename:Line	C2	C3	C4	C5	C6
1	C2	d1	./top.sdc:2	NA	A(SCG)	NC	NC	NC
2	C3	d2	./top.sdc:3	A(SCG)	NA	NC	NC	NC
3	C4		./top.sdc:4	NC	NC	NA	NC	NC
4	C5		./top.sdc:5	NC	NC	NC	NA	NC
5	C6		./top.sdc:6	NC	NC	NC	NC	NA

The following spreadsheet is generated for clocks that have not been assigned any domain.

Domain Rules

	A	B	C
	List of clocks with no domains	Filename:Line	Reason for NoDomain
1	C4	../top.sdc:4	No Constraint specified
2	C5	../top.sdc:5	No Constraint specified
3	C6	../top.sdc:6	No Constraint specified

Example 5 - Cyclic Dependency and Overridden `set_false_path`

Test Case Files Not Available

In this example, the mode is TC_STA. This example describes when [Message 11](#) is reported.

For example, consider a case where a crossing exists between C1 and C2, C2 and C3, and C3 and C1. In addition, the [set_clock_uncertainty](#) constraint is specified between C1 and C2 and C2 and C3, and the [set_false_path](#) constraint is specified between C3 and C1.

From the above constraints it seems that C1 and C3 are in different domains if the [set_false_path](#) is considered. However, if the [set_clock_uncertainty](#) constraint is considered then C1, C2, and C3 are in the same domain.

For the above scenario the domain relationship cannot be resolved irrespective of whether C1 is synchronous with C2, or C3 cannot be identified. Therefore, all the three clocks are considered as synchronous.

For this scenario, the following Domain Cyclic Error spreadsheet is generated:

	A	B	C
	Conflicting Clock Pair	Related Clocks	Type
1	C2 C3	C1 C2 C3	Cyclic error

Default Severity Label

Error

Rule Group

DomainRules

Reports and Related Files

- [Domain Matrix Report](#)
- [Domain Cyclic Errors Report](#)
- [Missing Domains Report](#)

Mnemonics in the Domains Spreadsheets

The mode set in the `tc_domain_mode` parameter determines the Mnemonics you see in the domain spreadsheets, such as the *Domain Matrix Report*, generated by the DomainError rule. The following table provides information on the Mnemonics used.

Mode	Mnemonic	Constraint	Relationship Inferred
Strict	SCGA	set_clock_group	Asynchronous
	SCGS	set_clock_group	Synchronous
	GEN	parent generated clock	Synchronous

Each cell contains information in the format

1. S (SCGA | SCGS | GEN)
2. A (SCGA | SCGS | GEN)
3. <blank> - no relationship could be inferred
4. NA - not applicable - for same clocks

S signifies a synchronous relation between the clocks.

A signifies an asynchronous relation between the clocks.

Cells highlighted in red represent the clock pairs for which conflict was detected because *set_clock_groups* and *create_generated_clock* were defined for a clock pair. In this case, clock pairs are considered asynchronous.

Mode	Mnemonic	Constraint	Relationship Inferred
STA	SCG	set_clock_group (Intergroup)	Asynchronous
	SFP	set_false_path	Asynchronous
	SCU	set_clock_uncertainty	Synchronous
	GEN	parent generated clock	Synchronous
	GRP	set_clock_group (Intragroup)	Synchronous
	V2R	virtual parent real clock	Synchronous
<p>Each cell contains information in the format</p> <ol style="list-style-type: none"> 1. S (SCG SFP SCU GEN GRP V2R) 2. A (SCG SFP SCU GEN) 3. NC - No domain assigned because of cyclic error. One or both Clock(s) is/part of cyclic conflict. 4. CE - No domain assigned because no constraint is specified. 5. NA - not applicable - for same clocks <p>S signifies a synchronous relation between the clocks. A signifies an asynchronous relation between the clocks.</p> <p>Cells highlighted in red represent the clock pairs for which conflict was detected because of either of the reasons:</p> <ol style="list-style-type: none"> 1. set_clock_groups and set_clock_uncertainty were defined for a clock pair. In this case, clock pairs are considered asynchronous. 2. set_clock_groups and create_generated_clock were defined for a clock pair. In this case, clock pairs are considered asynchronous. 3. set_false_path and set_clock_uncertainty were defined for a clock pair. In this case, clock pairs are considered asynchronous. 4. set_false_path and create_generated_clock were defined for a clock pair. In this case, clock pairs are considered asynchronous. 			
SGDC	SGDC	Not applicable	Relationship inferred from domain defined in SGDC
	GEN	Not applicable	Relationship inferred from parent generated clock constraint and SGDC
<p>Each cell contains information in the format</p> <ol style="list-style-type: none"> 1. S (SGDC GEN) 2. A (SGDC GEN) 3. <blank> - no relationship could be inferred 4. NA - not applicable - for same clocks. For example, the relationship of clock "C1" with clock "C1". 			

Domain Rules

Mode	Mnemonic	Constraint	Relationship Inferred
TC_STA	SCG	set_clock_group (Intergroup)	Asynchronous
	SFP	set_false_path	Asynchronous
	SCU	set_clock_uncertainty	Synchronous
	GEN	parent generated clock	Synchronous
	HMR	harmonic relation	Synchronous
	NCT	no constraint clock	Synchronous

Each cell contains information in the format

1. S (SCG | SFP | SCU | GEN | HMR | NCT)
2. A (SCG | SFP | SCU | GEN | HMR | NCT)
3. NC - No domain assigned because of cyclic error. One or both Clock(s) is/part of cyclic conflict.
4. CE - No domain assigned because no constraint is specified.
5. NA - not applicable - for same clocks

S signifies a synchronous relation between the clocks.

A signifies an asynchronous relation between the clocks.

Cells highlighted in red represent the clock pairs for which conflict was detected.

For more information on the **NC** and **CE** mnemonics, refer to the documentation of *Txv_DomainConflict* and *Txv_DomainMissing* rules in the *SpyGlass TXV Rules Reference Guide*.

DomainInfo

Generates clock domain information from SDC commands

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *DomainInfo* rule reports an error depending on the mode. The following table displays the checks performed when this rule is run in the STA (Default), TC_STA, STRICT, and SGDC modes. You can set the mode through the *tc_domain_mode* parameter.

For more information on domain inference, refer to the [Specifying Domains](#) section.

The *DomainInfo* reports messages, depending on the mode,

- If the *set_clock_groups* constraint is specified and there exists a crossing between two clocks
- If only *set_false_path* constraint is specified and there exists a crossing between the clocks
- If only *set_clock_uncertainty* constraint is specified and there exists a crossing between the clocks
- If the two clocks exhibit a harmonic relation and if neither a *set_false_path* nor a *set_clock_groups* constraints is specified and there exists a crossing between the two clocks
- If one clock is generated from another and there exists a crossing between the two clocks
- Harmonic relation - A pair of clocks is said to have a harmonic relation if their clock periods can be synchronized in N cycles (where N is the value which is less than or equal to the value specified using the *tc_c2c_max_cycles* rule parameter) and their duty cycles are same.
- If the *clock_group* constraint is not specified in the SGDC file and also none of the *set_false_path*/*set_clock_groups*/*set_clock_uncertainty*/source-generated relation/harmonic relation exists, but a clock crossing exists between two clocks, the clocks are assumed to be synchronous.

Rule Exception

This rule does not consider *set_false_path* constraints having -through list.

Parameter(s)

- *tc_c2c_max_cycles*: Default value is 50. Set the value to any positive integer to set the maximum value of the number of cycles in which two clocks can be synchronized.
- *tc_domain_mode*: Default is STA. Set the value to specify the mode for inferring domains. Other possible values are: TC_STA, STRICT, and SGDC.
- *tc_virtual_clock_prefixes*: Default is unspecified. Set the value of this parameter to the clock name prefix for virtual clock names for the domain flow.
- *tc_virtual_clock_suffixes*: Default is unspecified. Set the value of this parameter to the clock name suffix for virtual clock names for the domain flow.

Constraint(s)

SGDC

- *clock_group*: This is an SGDC constraint, used to specify the domain relationship between the clocks.

SDC

- *set_clock_groups*: Use to specify the clock groups that are mutually exclusive or asynchronous so that the paths between these clocks are not considered during timing analysis.
- *set_false_path*: Use to specify the paths in a design that are to be marked as false, so that they are not considered during timing analysis.
- *set_clock_uncertainty*: Use to specify uncertainty of specified clock networks.

Messages and Suggested Fix

For the message and suggested fix for clocks domain computation, refer to

Message 14.

Message 1

The following message appears in the TC_STA mode when inter-clock uncertainty has been specified from clock `<clk1-name>` to clock `<clk2-name>`:

[DomainInfo_01][INFO] Clocks "`<clk1-name>`" and "`<clk2-name>`" are synchronous as `set_clock_uncertainty` is set

For debugging information, refer to [How to Debug and Fix](#).

Message 2

The following message appears in the TC_STA mode when a [set_false_path](#) constraint has been specified from clock `<clk1-name>` to clock `<clk2name>`:

[DomainInfo_03][INFO] Clocks "`<clk1-name>`" and "`<clk2-name>`" are asynchronous as `set_false_path` is set

For debugging information, refer to [How to Debug and Fix](#).

Message 3

The following message appears in the TC_STA mode when a [set_clock_groups](#) constraint has been specified with clock `<clk1-name>` and clock `<clk2-name>` that belong to different groups:

[DomainInfo_02][INFO] Clocks "`<clk1-name>`" and "`<clk2-name>`" are asynchronous as `set_clock_groups` is set

For debugging information, refer to [How to Debug and Fix](#).

Message 4

The following message appears in the TC_STA mode when two clocks `<clk1-name>` and `<clk2-name>` show a source-generated relationship:

[DomainInfo_04][INFO] Clocks "`<clk1-name>`" and "`<clk2-name>`" are synchronous as "`<clk1-name>`" is generated from "`<clk2-name>`"

For debugging information, refer to [How to Debug and Fix](#).

Message 5

The following message appears in all modes when the [clock_group](#)

information is generated in a file *<file-name>* for design/block *<name>*:

[DomainInfo_06][INFO] For design/block "*<name>*", clock_group information are reported in file "*<file-name>*"

For debugging information, refer to [How to Debug and Fix](#).

Message 6

The following message appears when the Domain Matrix is generated in STA, TC_STA, and STRICT modes:

[INFO] Domain Matrix for design *<design-name>*, based on the individual constraints defined in the SDC as inferred in the *<mode-name>* mode

For debugging information, refer to [How to Debug and Fix](#).

Message 7

The following message appears when the Domain Matrix is generated in SGDC mode:

[INFO] Domain Matrix for design *<design-name>*, based on the individual constraints defined in the SGDC as inferred in the SGDC mode

For debugging information, refer to [How to Debug and Fix](#).

Message 8

The following message appears in the TC_STA mode when two clocks *<clk1-name>* with period *<period-1>* and *<clk2-name>* with period *<period-2>* exhibit a harmonic relationship and neither a [set_false_path](#) nor a [set_clock_groups](#) constraints is specified:

[DomainInfo_05][INFO] set_false_path or set_clock_groups not specified between clocks "*<clk1-name>*" (period : *<period1>*) and "*<clk2-name>*" (period : *<period2>*) sharing a harmonic relationship, for every N cycles of *<clk1-name>/<clk2-name>* (parameter tc_c2c_max_cycles : *<value>*), assumed as synchronous

Potential Issues

Messages 1 to 7: There are no potential issues. The messages are informational.

Message 8: This violation arises if the clocks are sharing a harmonic

relationship, assumed as synchronous.

Consequences of Not Fixing

Messages 1 to 7: There are no consequences of not fixing.

Message 8: If you do not fix this violation, the SpyGlass Constraints solution assumes these clocks as synchronous, other tools can assume it differently.

How to Debug and Fix

Message 1: Check whether the [set_clock_uncertainty](#) constraint reported is specified between the clocks reported in the violation message.

Message 2: Check whether the [set_false_path](#) constraint reported is specified between the clocks reported in the violation message.

Message 3: Check whether the [set_clock_groups](#) constraint reported is specified between the clocks reported in the violation message.

Message 4: No fix required. This is an informational message.

Message 5: The [clock_group](#) relationship is reported in the SGDC file that can be used in further analysis.

Message 6 and Message 7: Double-click the message to view the matrix. The following figure shows a Domain Matrix, which shows the clocks for all instances in the design.

	A	B	C	D	E	F	G	H	I
	Clock Name	Domain	Filename:Line	c1	c2	c3	c4	c5	c6
1	c1	d1	top.sdc:1	NA	A (SFP)	A	A	A	A
2	c2	d2	top.sdc:2	A (SFP)	NA	A	A	A	A
3	c3	d3	top.sdc:10	A	A	NA	A (SFP)	A	A
4	c4	d4	top.sdc:11	A	A	A (SFP)	NA	A	A
5	c5	d5	top.sdc:15	A	A	A	A	NA	A (SFP)
6	c6	d6	top.sdc:16	A	A	A	A	A (SFP)	NA

To view clocks specific to an instance, enter the full instance hierarchical name and click **Go**. The following figure shows the clocks crossing the block interface for `top/m1/b1`.

Domain Rules

	A	B	C	D	E	F
	Clock Name	Domain	Filename:Line	c1	c2	c3
1	c1	d1	top.sdc:1	NA	A (SFP)	A
2	c2	d2	top.sdc:2	A (SFP)	NA	A
3	c3	d3	top.sdc:10	A	A	NA

To revert back to seeing all the clocks of all instances in the design, clear the **Instance Filter** and click **Go**.

See [Example 1](#) for a sample Domain Matrix and the corresponding test case files.

Message 8: Ensure that you specify a valid constraint between the interacting clock pair.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows the Domain Matrix that is generated for a top design.

	A	B	C	D	E	F	G
	Clock Name	Domain	Filename:Line	clk1	clk2	clk3	clk4
1	clk1	d1	top.sdc:1	NA	A (SFP)	A	A
2	clk2	d2	top.sdc:2	A (SFP)	NA	A	A
3	clk3	d3	top.sdc:10	A	A	NA	A (SFP)
4	clk4	d4	top.sdc:11	A	A	A (SFP)	NA
5	clk5	d5	top.sdc:15	A	A	A	A
6	clk6	d6	top.sdc:16	A	A	A	A

Example 2

Test Case Files Not Available

Consider two clocks; clock C1 with a period of 10 and clock C2 with a period of 25. Let the value of the [tc_c2c_max_cycles](#) parameter be set to 50

In this example, the clock C1 will be harmonic after 5 cycles and the clock C2 will be harmonic after 2 cycles. Therefore, both the clocks exhibit a harmonic relation as value of [tc_c2c_max_cycles](#) rule parameter is greater than 5 (the maximum cycles required for the two clocks to be harmonic).

The following message is generated:

```
set_false_path or set_clock_groups not specified between clocks
"C1" (period : 5>) and "C2" (period : 25) sharing a harmonic
relationship, for every 5 cycles of "C1" (parameter
tc_c2c_max_cycles : 50), assumed as synchronous
```

Default Severity Label

Info

Rule Group

DomainRules

Reports and Related Files

The *DomainInfo* rule generates an SGDC file that contains all the domain information inferred from the SDC constraints that can be used for further analysis. In addition, refer to the [Domain Matrix Report](#).

Domain_SGDC_Consis

Reports inconsistency with domain extracted from SDC constraints

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *Domain_SGDC_Consis* rule reports inconsistency between the *clock_group* specified in the SGDC file and the domain extracted from the SDC constraints.

The *Domain_SGDC_Consis* rule reports an inconsistency in the following cases:

- If two clocks are specified as synchronous in the SGDC file but they are inferred as asynchronous from the SDC constraints
- If two clocks are specified as asynchronous in the SGDC file but they are inferred as synchronous from the SDC constraints
- If two clocks are inferred as synchronous from the SDC constraints but the *clock_group* relation between these clocks is not specified in the SGDC file
- If two clocks are inferred as asynchronous from the SDC constraints but the *clock_group* relation between these clocks is not specified in the SGDC file

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message appears when two clocks *<clk1-name>* and *<clk2-name>* are specified as synchronous in the SGDC file but they are inferred as asynchronous from the SDC constraints:

[WARNING] Clock pair "<clk1-name>" and "<clk2-name>" are specified as synchronous in SGDC, but inferred as asynchronous from the SDC

Message 2

The following message appears when two clocks <clk1-name> and <clk2-name> are specified as asynchronous in the SGDC file but they are inferred as synchronous from the SDC constraints:

[WARNING] Clock pair "<clk1-name>" and "<clk2-name>" are specified as asynchronous in SGDC, but inferred as synchronous from the SDC

Message 3

The following message appears when two clocks <clk1-name> and <clk2-name> are specified as synchronous from the SDC constraints but they are not specified in the SGDC file:

[WARNING] Clock pair "<clk1-name>" and "<clk2-name>" are inferred as synchronous from the SDC, but not specified in SGDC

Message 4

The following message appears when two clocks <clk1-name> and <clk2-name> are inferred as asynchronous from the SDC constraints but they are not specified in the SGDC file:

[WARNING] Clock pair "<clk1-name>" and "<clk2-name>" are inferred as asynchronous from the SDC, but not specified in SGDC

Message 5

The following message appears if the clock_group information is not specified in the SGDC file for design <name>:

[WARNING] Rule will not violate for design <name> since clock_group information is not specified in SGDC file

Potential Issues

The violation message appears because:

Messages 1 to 4: The clock relationship specified in the .sgdc and extracted from SDC are different.

Message 5: The `clock_group` constraint is not specified in the SGDC file.

Consequences of Not Fixing

The following are the consequences of not fixing:

Messages 1 to 4: The relationship specified in the SGDC file takes precedence.

Message 5: The rule checks the inconsistency between the clock relationship specified in SGDC and extracted from SDC files. So, no inconsistency will be reported.

How to Debug and Fix

Messages 1 to 4: The clock relationship inferred from SDC can be debugged using the violations reported by [DomainInfo](#) and [DomainError](#) rules. Modify the SDC or SGDC constraints.

Message 5: Specify the clock relationship in the SGDC file.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows a violation message that appears because the clocks `clk1` and `clk2` are inferred as synchronous in `top.sdc` because the [set_clock_uncertainty](#) constraint has been defined. However, this information is missing in the `test.sgdc` file.

The `top.sdc` file is shown as follows:

```

top.sdc | top.sdc (2) |
create_clock -name clk1 -period 1.912 cp1
create_clock -name clk2 -period 13 cp2

set_clock_uncertainty 1 -from clk1 -to clk2

create_clock -name clk3 -period 11.23 cp3
create_clock -name clk4 -period 14 cp4

set_false_path -from clk3 -to clk4

```

The test.sgdc file is shown as follows:

```

test.sgdc x
1 current_design top
2 sdcschema -file top.sdc
3
4 domain -name d1 -clock {clk1 clk4}
5

```

Example 2

Test Case Files Not Available

The following example illustrates inconsistency between the clock relationship of C1 and C2 in SGDC and SDC.

```

top.sgdc:
current_design top
sdc_data -file top.sdc
clock_group -name G1 -clock "C1"
clock_group -name G1 -clock "C2"

```

From the `clock_group` constraint, it is clear that C1 and C2 are synchronous.

```
top.sdc:
```

```
create_clock -name C1 -period 10 [get_ports clk1]
create_clock -name C2 -period 12 [get_ports clk2]
set_false_path -from [get_clocks C1] -to [get_clocks C2]
```

From the constraints specified above the clocks C1 and C2 are asynchronous.

There is an inconsistency between the clock relationship of C1 and C2 in SGDC and SDC, the rule reports a violation.

Default Severity Label

Warning

Rule Group

DomainRules

Reports and Related Files

None

Down Streaming Rules

The Down Streaming Rules Group `SDC_DnStrm` contains the following rules:

Rule	Description
<i>SDC_DnStrm01</i>	Constraints that appear in the analysis phases where they are not allowed or do not appear in analysis phases where they must appear
<i>SDC_DnStrm02</i>	Missing or incorrect <i>set_clock_gating_style</i> constraints
<i>SDC_DnStrm03</i>	Missing Physical Compiler related constraints
<i>SDC_DnStrm04</i>	Constraints specified in the SDC files and are forbidden by the specified tool
<i>SDC_DnStrm04a</i>	Commands specified in the SDC files that may be problematic for downstream tools
<i>SDC_DnStrm05</i>	Blocks without corresponding <i>set_max_dynamic_power</i> constraints or blocks with corresponding <i>set_max_dynamic_power</i> constraint set to zero
<i>SDC_DnStrm06</i>	Blocks without corresponding <i>set_max_leakage_power</i> constraints or blocks with corresponding <i>set_max_leakage_power</i> constraint set to zero
<i>SDC_DnStrm07</i>	Use of the <code>-analysis_type</code> option of the <code>set_operating_condition</code> command with a value other than <code>on_chip_variation</code>
<i>SDC_DnStrm08</i>	Clock pins with an abnormally large depth

SDC_DnStrm01

Identifies constraints requirements in mode of analysis

When to Use

Identify inappropriate constraints in the Post-layout phase.

Description

The *SDC_DnStrm01* rule identifies constraints that are not allowed or do not appear in the analysis phases. The *SDC_DnStrm01* rule requires that the following constraints appear in *sdc_data* for the RTL or Pre-layout analysis phases and not in the *sdc_data* for the Post-layout analysis phase:

- *set_dont_touch*
- `set_dont_touch_network`
- *set_ideal_net*
- *set_ideal_network*
- *set_ideal_latency*
- *set_ideal_transition*
- *set_resistance*

Parameter(s)

None

Constraint(s)

SDC

- *set_dont_touch* (Optional): Use to set the `dont_touch` attribute on cells, nets, designs, and library cells to prevent synthesis from replacing or modifying them during optimization.
- `set_dont_touch_network` (Optional): Use to set the `dont_touch` attribute on clock networks for synthesis.
- *set_ideal_net* (Optional): Use to define the net as ideal.

- *set_ideal_network* (Optional): Use to mark a set of ports or pins in the design as sources of an ideal network. This disables timing update of cells and nets in the transitive fan-out of the specified objects.
- *set_ideal_latency* (Optional): Use to specify ideal latency values for pins in an ideal network.
- *set_ideal_transition* (Optional): Use to specify ideal transition values for pins in an ideal network.
- *set_resistance* (Optional): Use to set the `ba_net_resistance` attribute with a resistance value on specified nets.

Messages and Suggested Fix

The following message appears when the constraint `<constr>` is set in an SDC file for the Post-layout analysis phase:

```
[WARNING] <constr> constraint should not be set in postlayout analysis
```

Where, `<constr>` can be *set_dont_touch*, *set_dont_touch_network*, *set_ideal_net*, *set_ideal_network*, *set_ideal_latency*, *set_ideal_transition*, or *set_resistance*.

Potential Issues

A violation is reported when these constraints are present in the post-layout SDC file.

Consequences of Not Fixing

Specific constructs are not expected in some analysis phases. This means that constraints are not appropriate for the stage of the design and can cause incorrect timing-analysis.

How to Debug and Fix

The SDC file highlights the constraint stated in the violation message. To resolve this violation, update the SDC file.

Example Code and/or Schematic

Suppose the design phase is Post-layout analysis and you have set the *set_ideal_network* constraint, as shown below:

```
set_ideal_network {IN1 IN2}
```

The `SDC_DnStrm01` rule reports a violation because the *set_ideal_network*

Down Streaming Rules

constraint should not be set during the Post-layout analysis phase.

Default Severity Label

Warning

Rule Group

SDC_DnStrm

Reports and Related Files

None

SDC_DnStrm02

Ensures `set_clock_gating_style` style matches with user style

When to Use

This rule is applicable to all phases of the design.

Description

The `SDC_DnStrm02` rule reports `set_clock_gating_style` constraints for the following cases:

- `set_clock_gating_style` constraint has not been set.
- `set_clock_gating_style` constraint has been set with arguments different from the user-specified styles. Refer to the Parameters section for details.

Parameter(s)

- `SDC_DnStrm02_seq_cell`: Default is unspecified. Set the value to none or latch to specify the sequential cell parameter type for `set_clock_gating_style` constraint.
- `SDC_DnStrm02_pedge_logic`: Default is unspecified. Set the value to a list of string values to specify the values of the `positive_edge_logic` option for the `set_clock_gating_style` constraint.
- `SDC_DnStrm02_control_point`: Default is unspecified. Set the value to before, after, or none to specify the `control_point` option for the `set_clock_gating_style` constraint.
- `SDC_DnStrm02_control_signal`: Default is unspecified. Set the value to scan_enable, test_mode, and unspecified to specify the `control_signal` option for the `set_clock_gating_style` constraint.

Constraint(s)

SGDC

- `set_clock_gating_style` (Mandatory): Use to set the clock gating style to be used for clock gating.

Messages and Suggested Fix

Message 1

The following message appears when the [set_clock_gating_style](#) constraint has not been set:

```
[WARNING] set_clock_gating_style missing from SDC constraints file
```

Message 2

The following message appears when the [set_clock_gating_style](#) constraint has been set with arguments different from the user-specified style `<style>`:

```
[WARNING] set_clock_gating_style not consistent with user-defined style <style>
```

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

Power Compiler has not been tuned with clock-gating options. Power Compiler will decide its own default values to perform the gating. These values will be different from what might be desired.

How to Debug and Fix

Ensure the format of [set_clock_gating_style](#) matches the user style as specified through parameters.

Example Code and/or Schematic

For the following snippet, the `SDC_DnStrm02` rule reports a violation because the `SDC_DnStrm02_seq_cell` parameter is set to "latch". This value is not matching with the command value `sequential_cell`, which is set to "none".

```
//test.sdc
set_clock_gating_style -sequential_cell "none" -
control_signal "test_mode" -control_point "before"
```

The parameter values are:

```
set_parameter SDC_DnStrm02_seq_cell "latch"  
set_parameter SDC_DnStrm02_control_point="none"  
set_parameter SDC_DnStrm02_control_signal="scan_enable"
```

This rule reports a violation if any of the values do not match. In this example, none of the parameter value matches with the command option.

Default Severity Label

Warning

Rule Group

SDC_DnStrm

Reports and Related Files

None

SDC_DnStrm03

Identifies physical compiler constraints

When to Use

Use this rule in the RTL and Pre-layout phases.

Description

The *SDC_DnStrm03* identifies missing Physical Compiler-related constraints. This rule reports a violation if any of the following constraints are not present:

- `set_floorplan_options` constraint (with at least utilization option)
- `set_floorplan_port_options`
- `set_floorplan_macro_options`
- `set_keepout_margin`, `create_bounds`
- `set_port_location`
- `set_cell_location`
- `set_dont_touch_placement`

Parameter(s)

None

Constraint(s)

SDC

- `set_floorplan_options` (with at least utilization option)
- `set_floorplan_port_options`
- `set_floorplan_macro_options`
- `set_keepout_margin`, `create_bounds`
- `set_port_location`
- `set_cell_location`
- `set_dont_touch_placement`

Messages and Suggested Fix

The following message appears when the constraint `<constr>` has not been specified for design unit `<du-name>`:

[WARNING] No `<constr>` specified for design unit `<du-name>`

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

These commands impose certain restrictions on placement for ports/cells. If you do not specify these constraints, the Physical Compiler may not be able to place modules on chip efficiently.

How to Debug and Fix

Specify the constraints reported in the violation message.

Example Code and/or Schematic

Not applicable

Default Severity Label

Warning

Rule Group

SDC_DnStrm

Reports and Related Files

None

SDC_DnStrm04

Reports constraints that are not understood by implementation tools, such as Astro and Monterey

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_DnStrm04* reports constraints specified in SDC files that are not permitted by tools other than Synopsys Design Compiler or PrimeTime.

Tool-specific files Astro and Monterey are installed with SpyGlass.

Parameter(s)

- *SDC_DnStrm04_template*: Default is Astro. Use this parameter to specify an ASCII file containing the list of forbidden constraints. Then, all uses of these constraints in the specified SDC files are flagged.

Constraint(s)

None

Messages and Suggested Fix

The following message appears when a forbidden constraint *<constr>* listed in the file specified with the *SDC_DnStrm04_template* rule parameter is encountered in the SDC file:

```
[WARNING] Command "<constr>" is not valid in flow
$SDC_DnStrm04_template
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Reporting constraints that are not understood by implementation tools, such as Synopsys, improves interoperability of SDC across tools, such as Synopsys.

How to Debug and Fix

Remove the constraint reported in the violation message from the SDC file.

Example Code and/or Schematic

For the following snippet, the *SDC_DnStrm04* reports a violation because `current_design` specified in the SDC file is not permitted by tools.

```
//test.sdc  
current_design flop
```

Default Severity Label

Warning

Rule Group

SDC_DnStrm

Reports and Related Files

None

SDC_DnStrm04a

Reports all non-SDC and Tcl commands that are not understood by implementation tools

When to Use

Rule ensures the correctness of Tcl commands

Description

The *SDC_DnStrm04a* reports constraints specified in SDC files that may be problematic for downstream tools. By default, the *SDC_DnStrm04a* rule reports the use of constraints other than those listed in Table: [Default List of Allowed Commands](#).

Parameter(s)

SDC_DnStrm04a_template: Default is `SdcCommands`. Use this parameter to specify your own list of allowed constraints. You can also specify SDC commands and Tcl commands.

Constraint(s)

None

Messages and Suggested Fix

The following message appears at the location where a forbidden constraint `<constr>` is encountered in the SDC file:

```
Command "<constr>" is not valid as not specified in  
'$SDC_DnStrm04a_template' file
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Reports non-SDC and Tcl commands that are not understood by implementation tools, such as Synopsys, improves interoperability of SDC across these tools.

How to Debug and Fix

Review the reported constraints in the SDC file. Use the correct syntax of

the command.

Example Code and/or Schematic

For the following snippet, the *SDC_DnStrm04a* rule reports a violation for the 2nd SDC line.

```
//test.sdc  
global var1  
expr 2 + 1
```

Default Severity Label

Warning

Rule Group

SDC_DnStrm

Reports and Related Files

None

SDC_DnStrm05

Reports `set_max_dynamic_power` that is missing or is set to zero

When to Use

To ensure design is constrained with `set_max_dynamic_power` constraint in the RTL phase.

Description

The `SDC_DnStrm05` rule checks whether the `set_max_dynamic_power` constraint has been specified with a non-zero value for each block. A block is specified using the `block` constraint in a SpyGlass Design Constraints file. This rule reports blocks without the corresponding `set_max_dynamic_power` constraint or with the corresponding `set_max_dynamic_power` constraint set to zero.

This rule separately identifies the top-level designs. These designs are actual top-level design units that are also specified using the `block` constraint in a SpyGlass Design Constraints file.

Parameter(s)

None

Constraint(s)

SDC

- `set_max_dynamic_power` (Mandatory): Use to set maximum dynamic power that is used by design.

Messages and Suggested Fix

Message 1

The following message appears for the partition `<partition-name>` where the `set_max_dynamic_power` constraint is set to a zero:

```
[WARNING] set_max_dynamic_power value is zero for <partition-type> <partition-name>
```

Where, `<partition-type>` can be block or top-level design.

Potential Issues

A violation can arise when either this rule is not applicable for the current run. Alternatively, you need to change the maximum power number in the constraint.

Consequences of Not Fixing

The rule is important if you are setting and analyzing the power related constraints. To do this, a specific power target is required. If the target is not specified or is set to zero, the tool does not optimize the design for power.

How to Debug and Fix

The `set_max_dynamic_power` constraint is set to a zero power value. The value should not be zero. The SDC file highlights the constraint.

To resolve this violation, update the SDC file.

Message 2

The following message appears for the partition `<partition-name>` that does not have a corresponding `set_max_dynamic_power` constraint:

```
[WARNING] set_max_dynamic_power is not defined for <partition-type> <partition-name>
```

Where, `<partition-type>` can be block or top-level design.

Potential Issues

A violation can arise when either this rule is not applicable for the current run or the max power number in the constraint is not valid.

Consequences of Not Fixing

The rule is important if you are setting and analyzing the power related constraints. To do this, a specific power target is required. If the target is not specified or is set to zero, the tool does not optimize the design for power.

How to Debug and Fix

To resolve this violation, update the SDC file with the `set_max_dynamic_power` constraint for the partition name.

Example Code and/or Schematic

In the following SDC file snippet, the max dynamic power is set to zero. Therefore, the `SDC_DnStrm05` rule reports a violation.

Down Streaming Rules

```
set_max_dynamic_power 0 uW
```

Default Severity Label

Warning

Rule Group

SDC_DnStrm

Reports and Related Files

None

SDC_DnStrm06

Reports `set_max_leakage_power` that is missing or is set to zero

When to Use

RTL phase

Description

The `SDC_DnStrm06` rule checks whether a `set_max_leakage_power` constraint has been specified with a non-zero value for each block. A block is specified using the `block` constraint in a SpyGlass Design Constraints file. The rule reports blocks that do not have the corresponding `set_max_leakage_power` constraints or have the corresponding `set_max_leakage_power` constraint set to zero.

These designs are actual top-level design units that are also specified using the `block` constraint in a SpyGlass Design Constraints file.

Parameter(s)

None

Constraint(s)

SDC

- `set_max_leakage_power` (Mandatory): Use to set maximum leakage power permitted for the design.

Messages and Suggested Fix

Message 1

The following message appears for a partition where the `set_max_leakage_power` constraint is set to zero:

```
[WARNING] set_max_leakage_power value is zero for <partition-type> <partition-name>
```

Where, `<partition-type>` can be `block` or `top-level design`.

Potential Issues

A violation can arise when either this rule is not applicable for the current run. Alternatively, you need to change the maximum power number in the

constraint.

Consequences of Not Fixing

The rule is important if you are setting and analyzing the power related constraints. To do this, a specific power target is required. If the target is not specified or is set to zero, the tool does not optimize the design for power.

How to Debug and Fix

The `set_max_leakage_power` constraint has a zero power value. The value should not be zero. The SDC file highlights the constraint.

To resolve this violation, update the SDC file.

Message 2

The following message appears for a partition `<partition-name>` that does not have a corresponding `set_max_leakage_power` constraint:

```
[WARNING] set_max_leakage_power is not defined for <partition-type> <partition-name>
```

Where, `<partition-type>` can be block or top-level design.

Potential Issues

A violation can arise when either this rule is not applicable for the current run. Alternatively, you need to change the maximum power number in the constraint.

Consequences of Not Fixing

The rule is important if you are setting and analyzing the power related constraints. To do this, a specific power target is required. If the target is not specified or is set to zero, the tool does not optimize the design for power.

How to Debug and Fix

To resolve this violation, update the SDC file with the `set_max_leakage_power` constraint for the partition name.

Example Code and/or Schematic

In the following SDC file snippet, the max leakage power is set to zero. Therefore, the `SDC_DnStrm06` rule reports a violation.

```
set_max_dynamic_power 0 uW
```

Default Severity Label

Warning

Rule Group

SDC_DnStrm

Reports and Related Files

None

SDC_DnStrm07

Reports the use of `set_operating_condition -analysis_type`

When to Use

This rule is applicable to all phases of design.

Description

The `SDC_DnStrm07` rule reports the use of the `analysis_type` argument of the `set_operating_conditions` constraint with a value other than `on_chip_variation`.

The `on_chip_variation` value specifies that the minimum and maximum operating conditions, which represent the lower and upper bounds of the maximum variation of operating conditions on the chip.

Parameter(s)

None

Constraint(s)

SDC

- `set_operating_conditions` (Mandatory): Defines the cells or ports on which to set operating conditions. If you do not use this option, operating conditions are set on the design.

Messages and Suggested Fix

The following message appears at the location of a `set_operating_conditions` constraint where the `-analysis_type` option is specified with value `<value>` other than `on_chip_variation`:

[WARNING] set_operating_condition -analysis_type <value> is not recommended as only on_chip_variation is the recommended analysis_type

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

To check the chip under extreme conditions, the `analysis_type` argument should be set to `on_variation_type`. Otherwise, the chip may fail on extreme conditions.

How to Debug and Fix

The violation message highlights the constraint specified in the SDC file. Consider using the `on_chip_variation` argument with the [set_operating_conditions](#) constraint.

Example Code and/or Schematic

For the following snippet, the `SDC_DnStrm07` rule reports a violation because `on_chip_variation` is not set for the [set_operating_conditions](#) constraint in the SDC file.

```
//test.sdc  
set_operating_conditions -library lsi_10k WCMIL
```

Default Severity Label

Warning

Rule Group

SDC_DnStrm

Reports and Related Files

None

SDC_DnStrm08

Reports clock pins with abnormally large depth

When to use

- Use this rule to detect abnormally deep clock pins.
- This rule is applicable in the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_DnStrm08* rule reports clock pins with an abnormally large depth. The threshold depth is specified using the *logic_level_max* rule parameter. The *SDC_DnStrm08* rule reports a violation for any depth exceeding the value specified with the *logic_level_max* rule parameter.

The *SDC_DnStrm08* rule reports a violation for:

- Each clock pin with a depth greater than the value specified with the *logic_level_max* rule parameter.
- Each clock design object, if the clock design object has at least one clock pin in its fan-out with a logic depth that exceeds the threshold depth and the value of the *verbose* rule parameter is set to no.
- The clock pins in the fan-out of the clock design object with a logic depth that exceeds the threshold depth if the value of the *verbose* rule parameter is set to yes.

The *SDC_DnStrm08* rule ignores the buffers and inverters while computing the logic level.

NOTE: *The depth of the clock pin is the sum of the instance_delay of the combinational logic from the clock source.*

The *SDC_DnStrm08* rule calculates the delay depths of each type of logic gate as described in the following table:

Gate Type	Delay	Associated Rule Parameter
Buffer	1	delay_Buf
Tristate buffer	1	delay_Tribuf
Inverter	1	delay_Inv
N-input AND	$\log(\$size)/\log(2)$	delay_And

Gate Type	Delay	Associated Rule Parameter
N-input NAND	$\log(\$size)/\log(2)$	delay_Nand
N-input OR	$\log(\$size)/\log(2)$	delay_Or
N-input NOR	$\log(\$size)/\log(2)$	delay_Nor
N-input XOR	$(\$size/4)+1$	delay_Xor
N-input XNOR	$(\$size/4)+1$	delay_Xnor
MUX with N select lines of which only one is active at a time and N inputs	2	delay_Mux
LogN MUX with N select lines and 2^N inputs	$(\$size/4)+1$	delay_LogN_mux
Decode operation, decoding from N bits to 2^N bits	$\log(\$size)/\log(2)+1$	delay_Decoder
Latch	2	delay_Latch
Unsigned full-adder with 2 N-bit inputs, plus carry-in and N+1-bit output plus carry-out	$\$size*2$	delay_Add_unsigned
Binary subtract (a-b) where a and b are both N-bit	$\$size*2$	delay_Subtract_binary
Unary subtract (-a) where a is N-bit	$\$size*2$	delay_Subtract_unary
Incrementor (a+1) where a is N-bit	$\$size*2$	delay_Increment
Decrementor (a-1) where a is N-bit	$\$size*2$	delay_Decrement
Fixed-point multiplier (a*b) where a and b are both N-bit	$\$size*4$	delay_Multiply_unsigned
Fixed-point multiplier (a*b) where a and b are both N-bit	$\$size*4$	delay_Multiply_signed

Down Streaming Rules

Gate Type	Delay	Associated Rule Parameter
Unsigned fixed-point greater-than comparison ($a > b$) where a and b are both N-bit	$\$size * 2$	<code>delay_GT_unsigned</code>
Signed fixed-point greater-than comparison ($a > b$) where a and b are both N-bit	$\$size * 2$	<code>delay_GT_signed</code>
Unsigned fixed-point greater-than/equals comparison ($a \geq b$) where a and b are both N-bit	$\$size * 2$	<code>delay_GE_unsigned</code>
Signed fixed-point greater-than/equals comparison ($a \geq b$) where a and b are both N-bit	$\$size * 2$	<code>delay_GE_signed</code>
Unsigned fixed-point less-than/equals comparison ($a \leq b$) where a and b are both N-bit	$(\$size/4) * 2$	<code>delay_LEQ_unsigned</code>
Signed fixed-point less-than/equals comparison ($a \leq b$) where a and b are both N-bit	$(\$size/4) * 2$	<code>delay_LEQ_signed</code>
Equality comparison ($a = b$) where a and b are both N-bit	$\$size * 2$	<code>delay_Equal</code>
Non-equality comparison ($a \neq b$) where a and b are both N-bit	$\$size * 2$	<code>delay_Not_equal</code>
Shift operation ($a >> b$ or $a << b$) where a and b are both N-bit	$\$size * 2$	<code>delay_Shift</code>
Absolute value evaluation operation (<code>abs(a)</code>) where a is N-bit	$((\$size - 1) * 2) + 1$	<code>delay_Abs</code>

NOTE: *In the above table, \$size is the number of inputs to a gate.*

The *SDC_DnStrm08* rule performs the traversal for each clock where the terminating points are:

- blocked pins
- black boxes
- sequential elements
- tri-state enable pins
- another clock, which can be *create_clock* and *create_generated_clock*
- pins stopped using the *set_clock_sense* command with the *stop_propagation* option. Refer to the *Stopping Clock Propagation* section for details.

Rule Exceptions

The *SDC_DnStrm08* rule ignores all hanging or unconnected nets.

Parameter(s)

- *logic_level_max*: Default value is 200. Sets the maximum number of logic levels.
- *verbose*: Default value is no. The *SDC_DnStrm08* rule reports a single violation from each clock source to all clock pins if the *verbose* parameter is set on default. If it is set to yes, then the rule reports violation for each single violating path from clock to pin.

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message appears if the value of the *verbose* parameter is set to no:

[WARNING] Fan-out of clock design object '*<design-object-name>*' has *<num1>* clock pin(s) having logic depth(s) greater than allowed maximum depth *<num2>* - maximum logic depth found is *<num3>*

Where,

- *<num1>* is the number of clock pins with a depth exceeding the mean depth as specified by the *logic_level_max* rule parameter.
- *<num2>* is the value of *logic_level_max* rule parameter.
- *<num3>* is the maximum value of logic levels of the clock pins in the design.

Potential Issues

The violation message appears if large number of combinatorial logic is present in between the clock-pin and the source of the clock.

Consequences of Not Fixing

If you do not fix this violation, large depth may result in spurious clocking or clock clipping.

How to Debug and Fix

- Check the logic present between the source of the clock and either one of the clock-pin reported.
- Either modify the value of rule parameter *logic_level_max* or modify the design.

Message 2

The following message appears if the value of the *verbose* parameter is set to *yes*:

[WARNING] Clock pin '*<pin-name>*' in fan-out of clock design object '*<design-obj-name>*' has depth *<num1>* greater than allowed maximum depth *<num2>*

Where,

- *<num1>* is the depth of the clock pin.
- *<num2>* is the mean depth specified by the *logic_level_max* rule parameter.

Potential Issues

The violation message appears if large number of combinatorial logic is present in between the clock-pin and source of the clock.

Consequences of Not Fixing

If you do not fix this violation, large depth may result in spurious clocking or clock clipping.

How to Debug and Fix

- Check the logic present between the source of the clock and either one of the clock-pin reported.
- Either modify the value of rule parameter *logic_level_max* or modify the design.

Message 3

The following message appears when a report *<report-name>* is generated if the depth of a clock pin exceeds the mean depth specified by the *logic_level_max* rule parameter for the design *<name>*:

[WARNING] For design *<name>*, clock pins have depth greater than mean *<should be replaced by: allowed maximum>* depth have been *<have been>* *<remove>* reported in file *<report-name>*

Where,

- *<name>* is the name of the top module.
- *<report-name>* is the name of the report.

Potential Issues

The violation message appears if large number of combinatorial logic is present in between the clock-pin and source of the clock.

Consequences of Not Fixing

If you do not fix this violation, large depth may result in spurious clocking or clock clipping.

How to Debug and Fix

- Check the logic present between the source of the clock and either one of the clock-pin reported.
- Either modify the value of rule parameter *logic_level_max* or modify the design.

Example Code and/or Schematic

Example 1

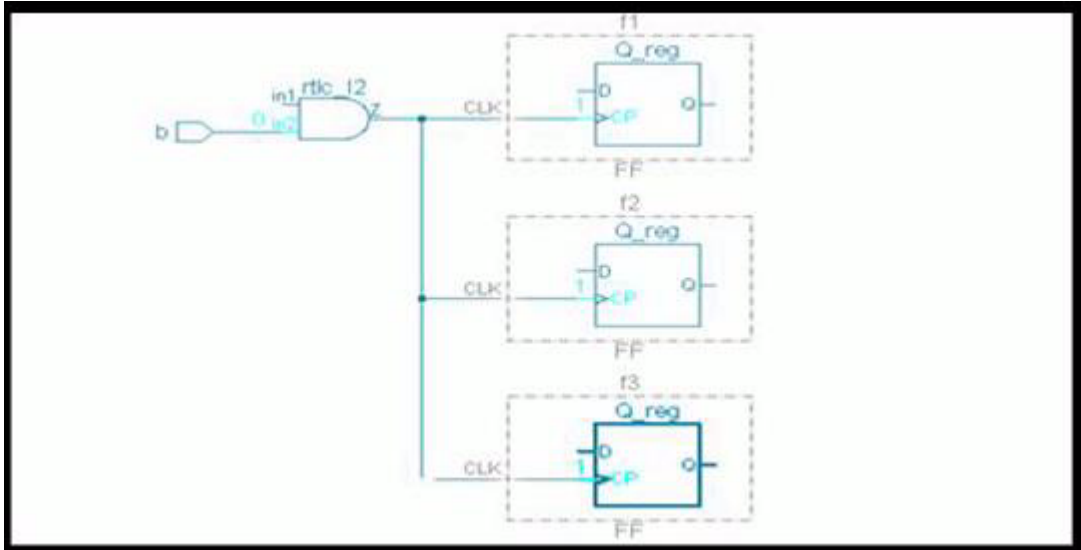
For the following snippet, the *SDC_DnStrm08* rule reports Message 1.

Down Streaming Rules

The value of the `logic_level_max` parameter is 0.

```
create_clock -name clk1 -period 10 b
```

The following schematic is generated:



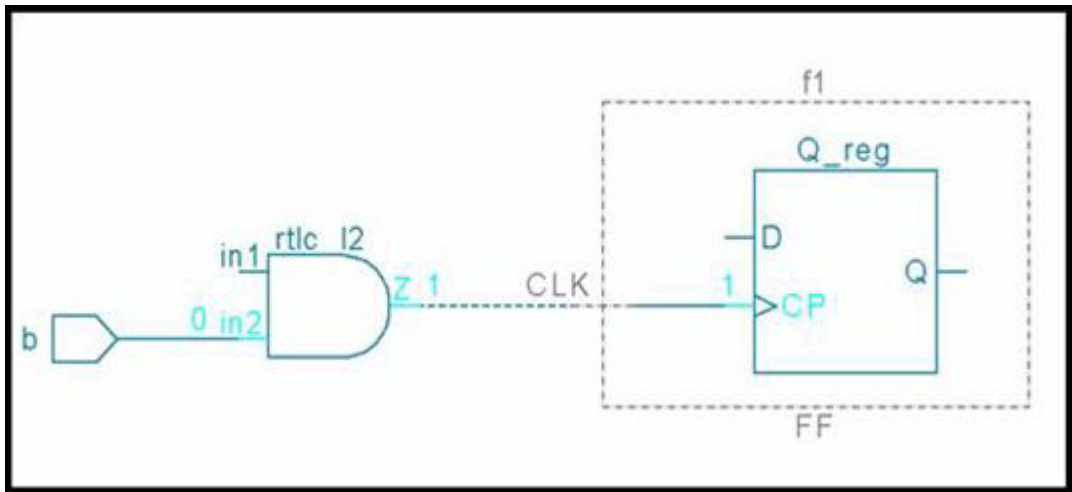
Example 2

For the following snippet, the `SDC_DnStrm08` rule reports Message 2.

The value of the `logic_level_max` parameter is 0.

```
create_clock -name clk1 -period 10 b
```

The following schematic is generated:



Example 3

The rule reports all the clock-pins with a logic level greater than the maximum allowed depth. Click the violation message to open the report. The CSV file will look like:

From Design Object	To Clock Pin	Clock Depth	File Name	Line No
B	f1/Q_reg/CP	1	test.sdc	1
C	F2/CP	2	test.sdc	2

Default Severity Label

Warning

Rule Group

SDC_DnStrm

Reports and Related Files

<module-name>-clk_depth_info<x>: The *SDC_DnStrm08* rule generates this report in a spreadsheet (.csv) format for each violation message.

The report is named as *<module-name>-clk_depth_info<x>* where *<module-name>* is the name of the design and *<x>* is a unique unsigned integer. The report is located at *<current-working-directory>/spyglass_reports/constraints* directory and displays clock pins whose depth is greater than the mean depth.

The report contains the following information:

- Clock design object
- Clock pin
- Depth of the clock pin from the clock source
- The name of the SDC file
- The line number of the SDC file

I/O Rules

The I/O Rules Group `IORules` contains the following sub-groups:

- *Combinational Paths Rules*
- *Input Delay Rules*
- *Input Transition Rules*
- *Input/Output Consistency Rules*
- *Output Delay Rules*
- *Load Rules*

Combinational Paths Rules

The Combination Paths Rules Sub-group `Combo_Paths` contains the following rules:

Rule	Description
<i>Combo_Paths01</i>	Port-to-port combinational paths without delay constraints set
<i>Combo_Paths02</i>	Paths with <code>set_max_delay</code> and <code>set_min_delay</code> constraints set but the maximum delay value is less than the minimum delay value
<i>Combo_Paths03</i>	Port-to-port combinational paths with incorrect input and output delay constraints set
<i>Combo_Paths04</i>	Incompletely specified <code>set_max_delay/set_min_delay</code> constraints
<i>Combo_Paths06</i>	Incomplete delay specifications for input ports and output ports connected by a combinational path

Combo_Paths01

Reports unconstrained combinational port-to-port paths

When to Use

RTL phase, Pre-layout phase, Post-layout phase

Description

The *Combo_Paths01* rule reports port-to-port combinational paths that do not have delay constraints set. By default, this rule checks for both [set_input_delay](#)/[set_output_delay](#) constraints and the [set_max_delay](#)/[set_min_delay](#) constraints. Therefore, this rule requires one of the following sets of constraints be specified for ports that have a combinational path between them:

- [set_max_delay](#)/[set_min_delay](#) constraints
- [set_input_delay](#) and [set_output_delay](#) constraints

Refer to the [Parameter\(s\)](#) section for more details on how to control the rule behavior.

Rule Exceptions

The *Combo_Paths01* rule ignores:

- input ports with the [set_case_analysis](#) command set.
- clock ports. These are ports used in a [create_clock](#) or [create_generated_clock](#) specification.

Parameter(s)

- [ignore_io_if_fp](#) (Default is no) and [tc_ignore_te](#) (Default is yes): By default, the *Combo_Paths01* rule does not consider the effect of the [set_false_path](#) constraint. Set [tc_ignore_te](#) rule to no and [ignore_io_if_fp](#) to yes, to consider the [set_false_path](#) constraints.

In GuideWare2.0, the default value of [tc_ignore_te](#) is no and the default value of [ignore_io_if_fp](#) is yes.

- [tc_allow_mm_or_io](#): Default is both. Set the value to io to check for [set_input_delay](#) and [set_output_delay](#) constraints only. Set the value to mm to check for the [set_max_delay](#)/[set_min_delay](#) constraints only.

Constraint(s)

SDC

- [set_max_delay/set_min_delay](#) (Mandatory): Use to define maximum and minimum delay for timing paths.
- [set_input_delay](#) (Mandatory): Use to define the arrival time relative to a clock.
- [set_output_delay](#) (Mandatory): Use to set output path delay values for the *current_design*.
- [set_false_path](#) (Optional): Use to identify paths as false, therefore are ignored during timing analysis.

Messages and Suggested Fix

Message 1

The following message appears for the design/block *<name>* that has a combinational path from the port *<port1-name>* to the port *<port2-name>* and a required constraint *<constr>* has not been specified:

[WARNING] Design/Block "*<name>*" contains a combinational path from port "*<port1-name>*" to port "*<port2-name>*" which has a missing "*<constr>*".

Where, *<constr>* can be [set_max_delay/set_min_delay](#), [set_input_delay](#), or [set_output_delay](#).

Potential Issues

The missing constraint is required for correct static timing analysis for the combinational path. Therefore, if the constraint is not specified, incorrect timing analysis is performed for the combinational path.

Consequences of Not Fixing

Implementation tools perform incorrect timing analysis for the combinational path. Therefore, the operation of the device might fail to meet design objectives.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the unconstrained combinational path starting from

an input port to an output port. The path is unconstrained when one of the pair of constraints, [set_input_delay/set_output_delay](#) combination or [set_max_delay/set_min_delay](#) combination, is missing.

Message 2

The following message appears for the design/block `<name>` that has a combinational path from the port `<port1-name>` to the port `<port2-name>` without any corresponding delay constraints set:

[WARNING] Design/Block "`<name>`" contains a combinational path from port "`<port1-name>`" to port "`<port2-name>`" which has neither `set_min/max_delay` nor `set_input/output_delay`

Potential Issues

The delay constraints are not set. Therefore, the combinational path is unconstrained.

Consequences of Not Fixing

If a combinational path is unconstrained, no timing check is performed. Therefore, the design could potentially fail to work in practice.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the unconstrained combinational path starting from an input port to an output port. The path is unconstrained when none of the constraints in both pairs is provided for the path.

To resolve this violation, constrain the combinational path with [set_max_delay/set_min_delay](#) or [set_input_delay/set_output_delay](#).

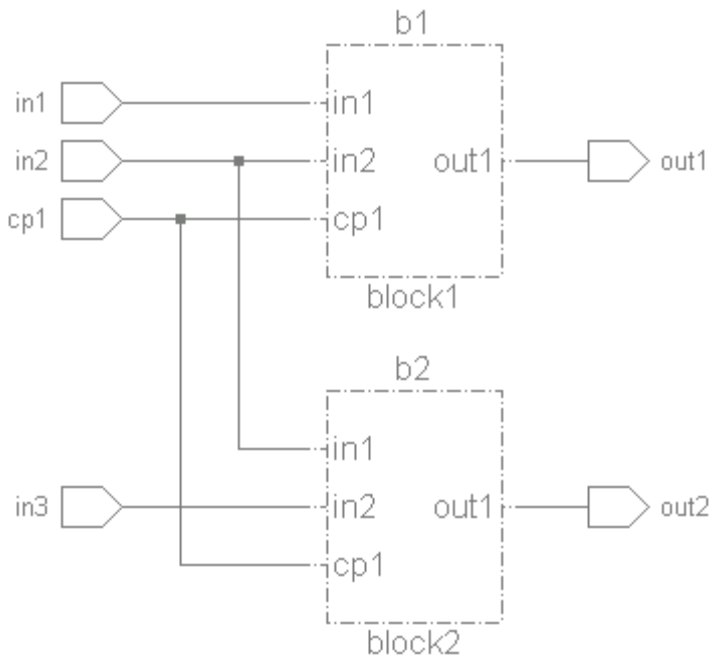
Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports [Message 1](#). The schematic of the top module is as follows:

I/O Rules



The block3 .sdc is as follows:

```

top.sdc
1 create_clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } {cp1}
2 set_input_delay 2.3 -clock clk1 {in1}
3 set_input_transition 1.2 cp1
4 set_input_transition 1.1 in1
5

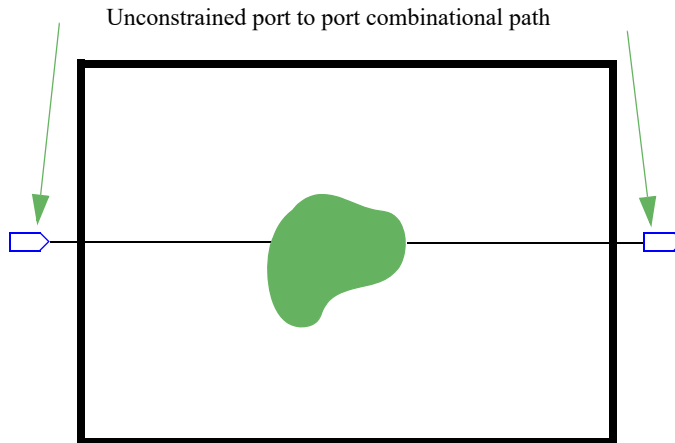
```

This violation appears the path from input port in1 to output port out1 is not constrained by [set_output_delay](#).

Example 2

Test Case Files Not Available

In the following example, constraints are not defined on the ports for which a combinational path exists between the input and output ports. Therefore, the *Combo_Paths01* rule reports a violation.



Default Severity Label

Warning

Rule Group

Combo_Paths

Reports and Related Files

None

Combo_Paths02

Reports inconsistent `set_max_delay` and `set_min_delay` commands

When to Use

To ensure consistency in `set_max_delay/set_min_delay` constraints in the RTL, Pre-layout, and Post-layout phases

Description

The *Combo_Paths02* rule reports paths with `set_max_delay/set_min_delay` constraints set, but the maximum delay value is less than the minimum delay value. This rule does not check for existence of the path; it checks whether both maximum and minimum delay values are specified for the path and the maximum delay value is not less than the minimum delay value.

The *Combo_Paths02* rule also checks for missing `set_min_delay` command for the corresponding `set_max_delay` command, and vice-versa.

Parameter(s)

- `tc_ignore_missing_minmax_delay`: Default is no and the *Combo_Paths02* rule reports a violation when `set_max_delay` is specified without a corresponding `set_min_delay`, and vice-versa. Set this parameter to yes to not perform this check.

Constraint(s)

SDC

- `set_max_delay/set_min_delay` (Mandatory): Use to define maximum and minimum delay for timing paths.

Messages and Suggested Fix

Message 1

The following message appears at the location of a `set_max_delay` constraint (at line `<num1>` in file `<file1-name>`) when the maximum

delay value *<value1>* specified for a path is less than the minimum delay value *<value2>* specified using the `set_min_delay` constraint (at line *<num2>* in file *<file2-name>*) for the same path in design/block *<name>*:

[WARNING] Design/Block "*<name>*" has max delay (*<value1>* set in file *<file1-name>*, line *<num1>*) less than min delay (*<value2>* set in file *<file2-name>*, line *<num2>*)

Potential Issues

The minimum delay values are greater than the maximum delay values. In addition, the combinational path is not constrained.

Consequences of Not Fixing

1. If the minimum delay values are greater than the maximum delay values, the tool can make its own assumptions regarding delay values for the timing path. The assumptions of the tool may be different from what was desired.
2. If a combinational path is unconstrained, no timing check is performed. Therefore, the operation of the device at any specified speed may vary.

How to Debug and Fix

The value of the `set_max_delay` constraint is less than the value of the `set_min_delay` constraint. This comparison occurs only if the paths are the same for both constraints. Review the constraints in the SDC as highlighted.

Message 2

The following message appears at the location of a `set_max_delay` constraint (at line *<num>* in file *<file-name>*) when only the maximum (or minimum) delay value *<value>* is specified for a path without the corresponding minimum (or maximum) delay in design/block *<name>*:

[WARNING] Design/Block "*<name>*" has *<delay-type>* delay (*<value>* set in file *<file-name>*, line *<num>*) specified with no corresponding *<delay-type>* delay

Where, *<delay-type>* is maximum or minimum.

Potential Issues

The maximum delay or minimum delay constraints are not specified. In

addition, the combinational path is not constrained.

Consequences of Not Fixing

1. If the minimum delay values are greater than the maximum delay values, the tool can make its own assumptions regarding delay values for the timing path. The assumptions of the tool may be different from what was desired.
2. If a combinational path is unconstrained, no timing check is performed. Therefore, it might lead to a timing failure in the design.

How to Debug and Fix

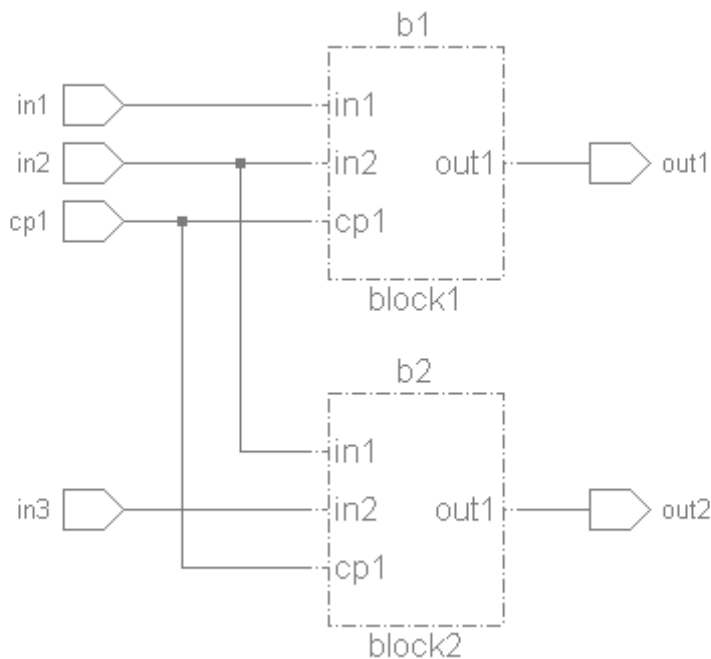
This violation message states the missing constraint, which is either [set_max_delay/set_min_delay](#), for the combinational path. Therefore, no comparison can take place as only one constraint is provided. Update the SDC file with the missing constraint.

Alternatively, if you do not want this check performed, set the [tc_ignore_missing_minmax_delay](#) parameter to yes.

Example Code and/or Schematic

Example 1[View Test Case Files](#)

This example shows when this rule reports [Message 2](#). The schematic of the top module is as follows:



The block3 .sdc is as follows:

```

block3.sdc | block3.sdc |
create clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } {cp1}
set_max_delay 2.0 -from in2 -to out1
set_max_delay 2.0 -from in2
set_max_delay 2.0 -to out1

```

This violation appears because the no min delay constraint is defined in the block3.sdc file, but *set_max_delay* has been defined. In the block3.sdc file, the corresponding *set_max_delay* constraint is highlighted.

Example 2

Test Case Files Not Available

In the following SDC file snippet, inconsistent *set_max_delay/set_min_delay* constraint values are provided. The *set_min_delay* value is more than the *set_max_delay* value. Therefore, the *Combo_Paths02* rule reports a violation.

```
set_max_delay 15.1 -from {ff1a} -to {ff1b}
set_min_delay 17.2 -from {ff1a} -to {ff1b}
```

Example 3Test Case Files Not Available

In the following SDC file snippet, *set_min_delay* is not set. Therefore, the *Combo_Paths02* rule reports a violation.

```
set_max_delay 3 -from {ff23a}
```

However, if the *tc_ignore_missing_minmax_delay* parameter is set to yes, the *Combo_Paths02* rule does not report a violation.

Default Severity Label

Warning

Rule Group

Combo_Paths

Reports and Related Files

None

Combo_Paths03

Reports intra-block combinational path that have a sum of output delay and input delay greater than clock period or sum of output delay and input delay greater than set_max_delay

When to Use

RTL phase, Pre-layout phase, Post-layout phase

Description

The *Combo_Paths03* rule reports port-to-port combinational paths with incorrect input and output delay constraints set. This rule works with the [set_input_delay/set_output_delay](#) constraint values and not with the [set_max_delay/set_min_delay](#) constraint values.

The *Combo_Paths03* rule reports port-to-port combinational paths in the following cases:

- The sum of input and output delays is greater than the maximum delay specified between the two ports where the value of the `set_max_delay` constraint is less than the clock period.
- The sum of input and output delays is greater than the clock period where the value of the `set_max_delay` constraint is greater than the clock period.
- The input and output delay constraints have been set with `-min` and `-max` arguments and the sum of `-min` argument values is more than the sum of `-max` argument values.
- The [tc_combopaths03_clk_multiplier](#) parameter is set and sum of input delay and output delay values is greater than or equal to:
 $\text{Clock Period} * \text{tc_combopaths03_clk_multiplier}$
- A [set_multicycle_path](#) constraint is set across the ports and the sum of the input delay and output delay values is greater than or equal to:
 $((\text{Multiplier of multi-cycle Path}) * \text{Clock Period})$
- A [set_multicycle_path](#) constraint is set across the ports, the [tc_combopaths03_clk_multiplier](#) is set and the sum of input delay and output delay values is greater than or equal to:
 $((\text{Multiplier of multi-cycle Path}) * \text{Clock Period} * \text{tc_combopaths03_clk_multiplier})$

Refer to the [Parameter\(s\)](#) section for more details on how to control the rule behavior.

Parameter(s)

- *tc_combopaths03_clk_multiplier*: Default is 1. Set the value to any positive float value.
- *ignore_io_if_fp* (Default is no) and *tc_ignore_te* (Default is yes): By default, the *Combo_Paths03* rule does not flag port-to-port paths that are specified as false paths. Set *tc_ignore_te* to no and *ignore_io_if_fp* to yes, to consider the *set_false_path* constraints.
In GuideWare2.0, the default value of *tc_ignore_te* is no and the default value of *ignore_io_if_fp* is yes.
- *tc_ignore_iodelay_if_maxdelay*: Default is no and the *Combo_Paths03* rule reports a violation when the sum of input and output delays is greater than the clock period where the value of the *set_max_delay* constraint is greater than the clock period. Set this parameter to yes to not report such violations.

Constraint(s)

SDC

- *set_input_delay* (Mandatory): Use to define the arrival time relative to a clock.
- *set_output_delay* (Mandatory): Use to set the output path delay values for the *current_design*.
- *set_multicycle_path* (Mandatory): Use to define the multi-cycle path.

Messages and Suggested Fix

Message 1

The following message appears for the design/block *<name>* that has a combinational path from the port *<port1-name>* to the port *<port2-name>* and sum *<value1>* of values set in the *-max* arguments of *set_input_delay* and *set_output_delay* is less than the sum *<value2>* of values set in the corresponding *-min* arguments:

[WARNING] Design/Block "*<name>*" contains a combinational path from port "*<port1-name>*" to port "*<port2-name>*" which has sum (*<value1>*) of max delay (input delay and output delay) less than sum (*<value2>*) of min delay (input delay and output delay)

Potential Issues

The sum of the max delay is less than the sum of the min delay. Therefore, the delay value is inconsistent.

Consequences of Not Fixing

During timing analysis, implementation tools modify the values internally for consistency. The criteria for modification may be different from your expectation.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the port-to-port combinational path constrained with [set_input_delay](#) and [set_output_delay](#) constraints. The sum of the maximum delay (input delay and output delay) is less than the sum of the minimum delay (input delay and output delay). Impact of the [set_false_path](#) constraint is considered for these paths through the [tc_ignore_te](#) parameter. Update the SDC file as highlighted.

Message 2

The following message appears for the design/block `<name>` that has a combinational path from the port `<port1-name>` to the port `<port2-name>` and sum `<value1>` of input delay value and output delay value is more than the clock period `<value2>` of the corresponding virtual clock `<clk-name>`:

[WARNING] Design/Block "`<name>`" contains a combinational path from port "`<port1-name>`" to port "`<port2-name>`" which has sum (`<value1>`) of input delay and output delay more than the clock period (`<value2>`) * (`<value3>`)

Where `<value3>` can be:

- Value of the path multiplier
- Value of [tc_combopaths03_clk_multiplier](#)
- Value of the path multiplier * [tc_combopaths03_clk_multiplier](#)

Potential Issues

The sum of the input and output delay for the combinational path should be within the corresponding clock period. In this case it is not. Therefore, there is a timing violation for the path.

Consequences of Not Fixing

Timing violation leads to data corruption issues in the design.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the port to port combinational path constrained with [set_input_delay](#) and [set_output_delay](#) constraints. The sum of the input and output delay is more than the associated clock's period. Impact of the [set_false_path](#) constraint is considered for these paths through the [tc_ignore_te](#) parameter. Update the SDC file as highlighted.

Message 3

The following message appears for the design/block *<name>* that has a combinational path from the port *<port1-name>* to the port *<port2-name>* and sum *<value1>* of input delay value and output delay value is equal to the clock period *<value2>* of the corresponding virtual clock *<clk-name>*:

[WARNING] Design/Block "*<name>*" contains a combinational path from port "*<port1-name>*" to port "*<port2-name>*" which has sum (*<value1>*) of input delay and output delay equal to the clock period (*<value2>*) * (*<value3>*)

Where *<value3>* can be either of the following:

- Value of the path multiplier
- Value of [tc_combopaths03_clk_multiplier](#)
- Value of the path multiplier * [tc_combopaths03_clk_multiplier](#)

Potential Issues

The sum of the input and output delay for the combinational path should be within the corresponding clock period. In this case it is not. Therefore, there is a timing violation for the path.

Consequences of Not Fixing

Timing violation leads to data corruption issues in the design.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the port to port combinational path constrained with

set_input_delay and *set_output_delay* constraints. The sum of the input delay and output delay is equal to the clock period. In case *set_multicycle_path* constraint is defined for same path, sum is compared to the product of the multiplier value and clock period rather than just the clock period. Impact of the *set_false_path* constraint is considered for these paths through the *tc_ignore_te* parameter. Update the SDC file as highlighted.

Message 4

The following message appears for the design/block *<name>* that has a combinational path from port *<port1-name>* to port *<port2-name>* and sum *<value1>* of input delay value and output delay value is greater than the specified maximum delay *<value2>*:

[WARNING] Design/Block "*<name>*" contains a combinational path from port "*<port1-name>*" to port "*<port2-name>*" which has sum (*<value1>*) of input delay and output delay more than the maximum delay (*<value2>*)

Potential Issues

The sum of the input and output delay for the combinational path should be within the corresponding clock period. In this case it is not. Therefore, there is a timing violation for the path.

Consequences of Not Fixing

Timing violation leads to data corruption issues in the design.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the port to port combinational path constrained with *set_input_delay* and *set_output_delay* constraints. The sum of the input and output delay is greater than the maximum delay value. Impact of the *set_false_path* constraint is considered for these paths through the *tc_ignore_te* parameter. Update the SDC file as highlighted.

Message 5

The following message appears for the design/block *<name>* that has a combinational path from port *<port1-name>* to port *<port2-name>* and the maximum delay value *<value1>* is greater than the clock period *<value2>*:

[WARNING] Design/Block *<name>* contains a combinational path

from port <port1-name> to port <port2-name> which has max delay <value1> greater than clock period <value2>

Potential Issues

The maximum delay for the combinational path should be within the corresponding clock period. In this case it is not. Therefore, there is a timing violation for the path.

Consequences of Not Fixing

Timing violation leads to data corruption issues in the design.

How to Debug and Fix

View the incremental schematic of the violation message.

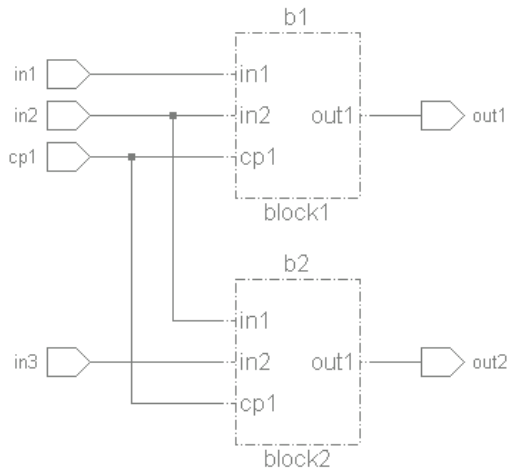
The schematic shows the port to port combinational path constrained with [set_input_delay](#) and [set_output_delay](#) constraints. The max delay value is greater than the clock period. Impact of the [set_false_path](#) constraint is considered for these paths through the [tc_ignore_te](#) parameter. Update the SDC file as highlighted.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

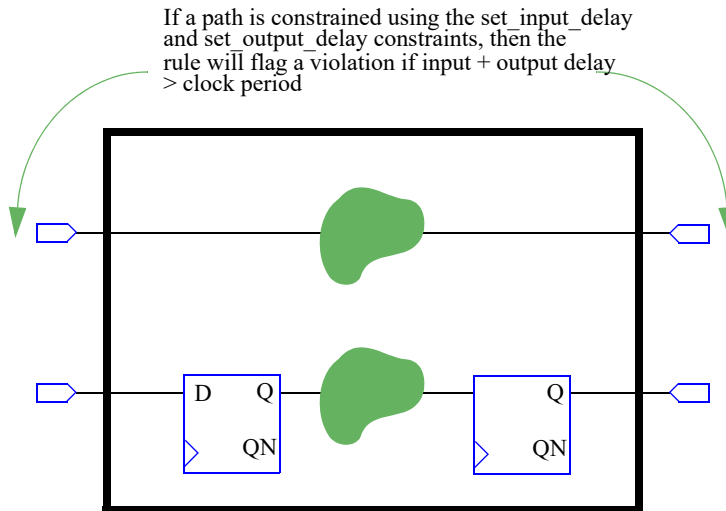
This example shows when this rule reports [Message 3](#). The schematic of the top module is as follows:



This violation appears because the sum of the input and output delay is more than the corresponding clock period for block3. The related calculation is shown in the message.

Example 2Test Case Files Not Available

In the following example, the sum of [set_input_delay](#) and [set_output_delay](#) is more than the clock period of the virtual clock. Therefore, the *Combo_Paths03* rule reports a violation.



Default Severity Label

Warning

Rule Group

Combo_Paths

Reports and Related Files

None

Combo_Paths04

Identifies `set_max_delay` or `set_min_delay` specified explicitly with only rise or fall options

When to Use

This rule is applicable to all phases in the design.

Description

The *Combo_Paths04* rule reports incompletely specified [set_max_delay/set_min_delay](#) constraints. This rule requires that for every `set_max_delay` constraint (or `set_min_delay`) specified with the `rise` argument for a set of delay paths, there must exist a `set_max_delay` constraint (or `set_min_delay`) specified with the `fall` argument for exactly the same set of delay paths. All other cases are reported.

Parameter(s)

None

Constraint(s)

SDC

- [set_max_delay/set_min_delay](#) (Mandatory): Use to specify the maximum or minimum delay.

Messages and Suggested Fix

The following message appears for a [set_max_delay/set_min_delay](#) constraint specified with the `-rise` argument (the `-fall` argument) for a set of delay paths in design/block `<name>` if there is no corresponding `set_max_delay` constraint (or `set_min_delay` constraint) specified with the `-fall` argument (the `-rise` argument) for exactly the same set of delay paths in the block:

```
No <constr> constraint defined (<path>) with <option> option in design/block "<name>"
```

Where,

- `<constr>` can be `set_max_delay` or `set_min_delay`,
`<option>` can be `-rise` or `-fall`
- `<path>` is the combinational path in the format `from <name>`,
`from <name> to <name>`, or
`from <name> through <name> to <name>`.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Values for the unspecified options get inferred by the tool and the inferred value can be different from what was desired.

How to Debug and Fix

Specify all the options.

Example Code and/or Schematic

For the following snippet, the `rise` argument is specified only. Therefore, the `Combo_Paths04` rule reports a violation.

```
//test.sdc
set_max_delay 1.0 -from {in1 in2 in3} -through {F1/q F2/q} -
to {out1} -rise
```

Default Severity Label

Warning

Rule Group

Combo_Paths

Reports and Related Files

None

Combo_Paths06

Reports incomplete delay specifications for input and output ports

When to Use

This rule is applicable to the RTL, pre-layout, and post-layout phases.

Description

The *Combo_Paths06* rule reports incomplete delay specifications for input and output ports which are connected by a combinational path. If there is a combinational path between an input and output port and input delay is specified for the input port with a clock, the output delay should be specified with the same or similar clock. Similarly, if output delay is applied on the output, input delay should be specified on the output side. A similar clock is a clock that has a matching period and waveform.

Parameter(s)

- *ignore_io_if_fp* (Default is no) and *tc_ignore_te* (Default is yes): By default, the *Combo_Paths06* rule does not flag port-to-port paths that are specified as false paths. Set *tc_ignore_te* to no and *ignore_io_if_fp* to yes to consider the *set_false_path* constraints.

In GuideWare2.0, the default value of *tc_ignore_te* is no and the default value of *ignore_io_if_fp* is yes.

Constraint(s)

- *set_input_delay* (Optional): Use to specify input delay on input or inout ports.
- *set_output_delay* (Optional): Use to specify output delay on inout or output ports.
- *set_false_path* (Optional): Use to identify paths in a design that are to be marked as false.

Messages and Suggested Fix

Message 1

The following message appears:

[INFO] set_input_delay for port (<port-name>) should be specify with clock '<clk-name>'

Potential Issues

This violation message appears when all of the following are true:

- the input port <port-name> when it does not have a *set_input_delay* specification with respect to clock <clk-name>,
- the input port is connected to an output port by a combinational path, and
- the output port has a *set_output_delay* specification with respect to the clock and the input port does not have any other *set_input_delay* specification.

Consequences of Not Fixing

It can be difficult meeting the timing requirements. Even if the timing requirements are met, due to false path/multi-cycle path specification, the actual design might not be able to meet the timing requirements.

How to Debug and Fix

The SDC file/line of the related *set_output_delay* constraint is highlighted in the Console GUI. To fix the issue, specify *set_input_delay* on the reported port with the clock reported or with a clock having similar characteristics.

Message 2

The following message appears:

[INFO] set_output_delay for port (<port-name>) should be specify with clock '<clk-name>'

Potential Issues

This violation message appears when all of the following are true:

- the output port <port-name> when it does not have a *set_output_delay* specification with respect to clock <clk-name>,
- the output port is connected to an input port by a combinational path, and
- the input port has a *set_input_delay* specification with respect to the clock <clk-name> and the output port does not have any other *set_output_delay* specification.

Consequences of Not Fixing

It can be difficult meeting the timing requirements. Even if the timing requirements are met, due to false path/multi-cycle path specification), the actual design might not be able to meet the timing requirements.

How to Debug and Fix

The SDC file/line of the related [set_input_delay](#) constraint is highlighted in the Console GUI. To fix the issue, specify [set_output_delay](#) on the reported port with the clock reported or with a clock having similar characteristics.

Message 3

The following message appears:

```
[INFO] set_input_delay for port (<port-name>) should be specify with clock '<clk-name>' with -add_delay
```

Potential Issues

This violation message appears when all of the following are true:

- the input port *<port-name>* when it does not have a [set_input_delay](#) specification with respect to clock *<clk-name>*,
- the input port is connected to an output port by a combinational path, and
- the output port has a [set_output_delay](#) specification with respect to the clock *<clk-name>* and the input port has one or more [set_input_delay](#) specifications with respect to other clocks.

Consequences of Not Fixing

It can be difficult meeting the timing requirements. Even if the timing requirements are met, due to false path/multi-cycle path specification), the actual design might not be able to meet the timing requirements.

How to Debug and Fix

The SDC file/line of the related [set_output_delay](#) constraint is highlighted in the Console GUI. To fix the issue, specify [set_input_delay](#) with the `-add_delay` option on the reported port with the clock reported or with a clock having similar characteristics. The `-add_delay` option should be specified, otherwise the [set_input_delay](#) constraint overwrites the specified input delay on this port.

Message 4

The following message appears:

[INFO] `set_output_delay` for port (<port-name>) should be specified with clock '<clk-name>' with `-add_delay`

Potential Issues

This violation message appears when all of the following are true:

- the output port <port-name> when it does not have a `set_output_delay` specification with respect to clock <clk-name>
- the output port is connected to an input port by a combinational path and the input port has a `set_input_delay` specification with respect to the clock <clk-name>, and
- the output port has one or more `set_output_delay` specifications with respect to other clocks

Consequences of Not Fixing

Meeting timing can be difficult and even if the timing is met (due to false path/multi-cycle path specifications), the actual design might not be able to meet the timing.

How to Debug and Fix

The SDC file/line of related `set_input_delay` constraint is highlighted in the GUI. To fix the issue, specify `set_output_delay` with `-add_delay` option on the reported port with the clock reported or with a clock having similar characteristics. Here, `-add_delay` option should be specified, otherwise `set_output_delay` constraint will overwrite already specified output delay on this port.

Example Code and/or Schematic

In the following example, the input/output delays for the `in1` input port and the `out1` output port are set:

```
set_input_delay 2 -clock [get_clock CLK1]
  [get_port in1]
set_output_delay 3 -clock [get_clock CLK2]
  [get_port out1]
```

Assume that the clock characteristics of the clocks, `CLK1` and `CLK2`, do not match. In addition, `in1` and `out1` are connected by a combinational path. In this case, the `Combo_Paths06` rule expects an input delay specification for `in1` with respect to `CLK2` and an output delay

specification for out1 with respect to clock CLK1.

Default Severity Label

Info

Rule Group

Combo_Paths

Reports and Related Files

None

Input Delay Rules

The Input Delay Rules sub-group `Inp_Del` contains the following rules:

Rule	Description
<i>Inp_Del01</i>	Runs <i>Inp_Del01a</i> , <i>Inp_Del01b</i> , and <i>Inp_Del01c</i> rules
<i>Inp_Del01a</i>	Input ports without input delay constraints
<i>Inp_Del01b</i>	Input ports without a delay constraint set and without any of the specified constraints set
<i>Inp_Del01c</i>	Input ports without a <i>set_input_delay</i> constraint set and with any of the <i>set_case_analysis</i> / <i>set_disable_timing</i> / <i>set_false_path</i> constraints set
<i>Inp_Del02</i>	Input ports having Input Delay constraints with incorrect min and max values
<i>Inp_Del03</i>	Runs <i>Inp_Del03a</i> and <i>Inp_Del03b</i> rules
<i>Inp_Del03a</i>	Input ports having Input Delay constraints using incorrect clocks
<i>Inp_Del03b</i>	Input ports having Input Delay constraints using incorrect clocks
<i>Inp_Del04</i>	<i>set_input_delay</i> constraints without either <code>min</code> or <code>max</code> argument
<i>Inp_Del05</i>	Input constraints specified with clock other than virtual clocks
<i>Inp_Del07</i>	Input ports where the input delay value is more than the user-specified percentage of the corresponding clock period
<i>Inp_Del07a</i>	Input ports where the input delay value is within/outside (selectable) of the specified percentage range of the corresponding clock period
<i>Inp_Del08</i>	Input ports with <i>set_input_delay</i> constraints set for two different clocks without the <code>add_delay</code> argument
<i>Inp_Del09</i>	Bus input ports where not all bits have the same <i>set_input_delay</i> constraint delay values
<i>Inp_Del10</i>	<i>set_input_delay</i> constraints specified with the <code>level_sensitive</code> argument
<i>Inp_Del11</i>	<i>set_input_delay</i> constraints specified with the <code>clock_fall</code> argument

Rule	Description
<i>Inp_Del12</i>	<i>set_input_delay</i> constraints specified with the <i>network_latency_included</i> or the <i>source_latency_included</i> arguments that are unsupported (for synthesis) arguments
<i>Inp_Del13</i>	<i>set_input_delay</i> constraints with negative values set
<i>Inp_Del14</i>	<i>set_input_delay</i> constraints specified with wrong/incomplete set of clocks
<i>Inp_Del15</i>	Reports internal pins on which <i>set_input_delay</i> is specified

Inp_Del01

Input does not have an input constraint

The Inp_Del01 rule runs the [Inp_Del01a](#), [Inp_Del01b](#), and [Inp_Del01c](#) rules.

Inp_Del01a

Reports unconstrained input/inout port by `set_input_delay`

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Inp_Del01a* rule identifies input/inout ports that do not have a [set_input_delay](#) constraint set and satisfy any of the following conditions:

- The input/inout port reaches the data pin of a sequential cell if the [tc_allow_async_pins](#) parameter is set, the *Inp_Del01a* rule also considers the asynchronous pins of a sequential cell.
- All pins except clock pins are considered to be asynchronous pins.
- The input/inout port does not reach the data pin of a sequential cell but a combinational path exists from this port to an output/inout port, and input/inout port is not constrained by the [set_max_delay/set_min_delay](#) constraints.

Rule Exceptions

The *Inp_Del01a* rule ignores the following:

- Hanging path from an input/inout port
- Path from an input/inout port with a [set_case_analysis](#) constraint
- Path from an input/inout port with a [create_clock](#) constraint
- Path from an input/inout port with a [create_generated_clock](#) constraint
- Path from an input/inout port to a blackbox
- Input/inout port with a [set_disable_timing](#) constraint
- The clock path that starts from the clock pin of a flip-flop (D pin of the flip-flop is reached from an input/inout port), and has a pin constrained with the [set_clock_sense/set_sense](#) constraint with the `-stop_propagation` option. Refer to the [Stopping Clock Propagation](#) section for details.
- Transitive fan-out ends on the scan enable pin. Instead, the [SDC_Methodology30](#) rule reports a violation for these ports if you have not specified the [set_case_analysis](#) constraint.

- Ports connected to the scan enable pins of sequential elements.

In case of multiple clocks defined on the same object using the `-add` option of the `create_clock/create_generated_clock` constraint, the `Inp_DeI01a` rule retains only the last clock defined without the `-add` option and any clocks defined with the `-add` option after this last clock.

Parameter(s)

- `tc_allow_async_pins`: Default value is `no`. Set the value to `yes` so that the tool considers asynchronous pins of sequential elements for computing sampling clocks information of input delays.
- `tc_at_speed_testing_mode`: Default value is `no`. Set the value to `yes` to make the `Inp_DeI01a` rule treat the scan enable pins as the data pins of sequential elements.
- `tc_ignore_te` (Default value is `yes`) and `ignore_io_if_fp` (Default value is `no`). If the value of the `tc_ignore_te` parameter is set to `no` and the value of the `ignore_io_if_fp` parameter is set to `yes`, the `set_false_path` constraints are taken into consideration, if specified.

In GuideWare2.0, the default value of `tc_ignore_te` is `no` and the default value of `ignore_io_if_fp` is `yes`.

- `tc_ignore_async_ports`: Default is `unspecified` and this rule does not ignore asynchronous ports. Specify the file name that contains the list of asynchronous ports, which this rule should ignore.

Constraint(s)

- `set_input_delay`: Use this constraint to define the arrival time relative to a clock.
- `set_max_delay/set_min_delay`: Use this constraint to specify a maximum/minimum delay for timing paths.
- `set_case_analysis`: Use this constraint to specify that a port or pin is at a constant logic value 1 or 0, or is considered with a rising or falling transition.
- `create_clock/create_generated_clock`: Use this constraint to create clock at design object.
- `set_disable_timing`: Use this constraint to disable timing arcs of instances.

- *set_clock_sense/set_sense*: Use to specify the clock propagation conditions.
- *set_false_path*: Use this constraint to falsified paths from timing analysis.

Messages and Suggested Fix

If the name of the port, *<port-name>*, is listed as an asynchronous port through the *tc_ignore_async_ports* parameter, the severity of each message listed in this section is INFO.

Message 1

The following message appears when an input/inout port *<port-name>* of design/block *<name>* does not have a corresponding *set_input_delay* constraint and the input/inout port:

- Reaches the data pin of a sequential cell which is not driven by any clock
- Does not reach any data pin of a sequential cell but reaches an output/inout port

[I np_Del 01a_01] [WARNING] Input delay constraint not set on Port "*<port-name>*" of design/block *<name>*

Or

[I np_Del 01a_02] [INFO] Input delay constraint not set on Port "*<port-name>*" of design/block *<name>*

Message 2

The following message appears when an input/inout port *<port-name>* of design/block *<name>* does not have any *set_input_delay* constraint set and the input/inout port reaches the data pin of a sequential cell (triggered by clocks *<clk-name-list>*) in its fan-out:

[I np_Del 01a_03] [WARNING/INFO] Input delay constraint not set on Port "*<port-name>*" of design/block *<name>* and input delay needs to be specified with clock(s) *<clkname-list>*

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

The timing analysis results will be invalid without proper [set_input_delay](#) constraints and this results in a serious error in the design. The timing results may give a false sense of security and you may get silicon timing failure. Tools may assume optimistic delay to the ports.

How to Debug and Fix

View the incremental schematic of the violation message. The schematic displays the clock and data paths that lead to the port that does not have [set_input_delay](#) set.

To resolve this violation, specify the [set_input_delay](#) constraint.

This rule supports the SDC Autofix feature. For more information, see [Using SDC Autofix](#).

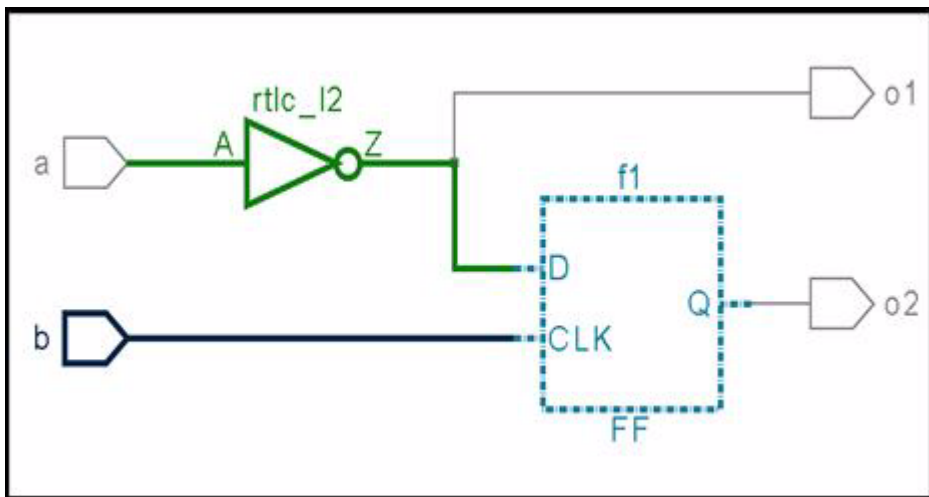
Example Code and/or Schematic

Suppose, a clock is applied at Port b as follows:

```
create_clock -name c1 -period 20 b
```

No [set_input_delay](#) constraint is applied on Port a.

This rule reports a violation and highlights the data and clock paths to help you specify [set_input_delay](#) on Port a. The following schematic is generated:



Default Severity Label

Warning

Rule Group

Inp_De1

Reports and Related Files

None

Inp_Del01b

Identifies input/inout port that is not constrained by `set_input_delay`

When to Use

To identify ports that are not constrained with the `set_input_delay` constraint. This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The `Inp_Del01b` rule identifies input/inout ports that do not have a `set_input_delay` constraint set and satisfy any of the following conditions:

- The input/inout port reaches the data pin of a sequential cell
If the `tc_allow_async_pins` parameter is set, this rule also considers the asynchronous pins of a sequential cell.
- The input/inout port does not reach the data pin of a sequential cell. In addition, a combinational path exists from the input/inout port to an output/inout port and the input/inout port is not constrained by the `set_max_delay/set_min_delay` constraints.
- If a pin is constrained with the `set_clock_sense/set_sense` command by using the `-stop_propagation` option. Refer to the *Stopping Clock Propagation* section for details.

In case of multiple clocks defined on the same object using the `-add` option of the `create_clock` or `create_generated_clock` constraint, the `Inp_Del01b` rule retains only the last clock defined without the `-add` option and any clocks defined with the `-add` option after this (last) clock.

Rule Exceptions

The `Inp_Del01b` rule ignores the following:

- Hanging path from an input/inout port
- Path from an input/inout port with a `set_case_analysis` constraint
- Path from an input/inout port with a `create_clock` constraint
- Path from an input/inout port with a `create_generated_clock` constraint
- Path from an input/inout port to a blackbox

- Input/inout port with a `set_dont_touch_network` constraint
- Input/inout port with a `set_ideal_network` constraint
- The clock path that starts from the clock pin of a flip-flop (D pin of the flip-flop is reached from an input/inout port), and has a pin constrained with the `set_clock_sense/set_sense` constraint with the `-stop_propagation` option. Refer to the *Stopping Clock Propagation* section for details.
- Transitive fan-out ends on the scan enable pin. Instead, the *SDC_Methodology30* rule reports a violation for these ports if you have not specified the `set_case_analysis` constraint.
- Ports connected to the scan enable pins of sequential elements. Refer to the *Parameter(s)* section for more details.

Parameter(s)

- If the value of the `tc_ignore_te` rule parameter is set to `no` and the value of the `ignore_io_if_fp` rule parameter is set to `yes`, then the `set_false_path` constraints are taken into consideration, if specified.
In GuideWare2.0, the default value of `tc_ignore_te` is `no` and the default value of `ignore_io_if_fp` is `yes`.
- `tc_at_speed_testing_mode`: Default is `no`. Set the value to `yes` to make the *Inp_Del01b* rule treat the scan enable pins as the data pins of sequential elements.
- `tc_ignore_async_ports`: Default is `unspecified` and this rule does not ignore asynchronous ports. Specify the file name that contains the list of asynchronous ports, which this rule should ignore.
- `tc_ignore_io_if_minmax_delay`: Default is `no` and this rule reports missing IO delays. Set this parameter to `yes` to ignore missing IO delays, when `set_max_delay` or `set_min_delay` is specified, even if the port is connected to a register.

Constraint(s)

SDC

- `set_clock_sense/set_sense` (Optional): Use to specify the clock propagation conditions.

- *set_false_path* (Optional): Use to identify paths in a design that are to be marked as false, so that they are not considered during timing analysis. Refer to the *Parameter(s)* section for details on how to consider *set_false_path* constraints.
- *set_input_delay* (Optional): Use to define the arrival time relative to a clock.
- *create_clock* (Optional): Used to create a clock.
- *create_generated_clock* (Optional): Used to create a clock.

Messages and Suggested Fix

If the name of the port, *<port-name>*, is listed as an asynchronous port through the *tc_ignore_async_ports* parameter, the severity of each message listed in this section is INFO.

Message 1

The following message appears when the input/inout port *<port-name>* of the design/block *<name>* does not have a corresponding *set_input_delay* constraint and the input/inout port:

- Reaches the data pin of a sequential cell that is not driven by any clock
- Does not reach any data pin of a sequential cell but reaches an output/inout port

[I np_Del 01b_01] [WARNI NG] I nput delay constraint not set on Port "*<port-name>*" of desi gn/bl ock *<name>*

Or,

[I np_Del 01b_02] [I NFO] I nput delay constraint not set on Port "*<port-name>*" of desi gn/bl ock *<name>*

Potential Issues

Implementation tools might assume zero delay value for the port.

Consequences of Not Fixing

This is a design error because the timing analysis results are not valid without proper *set_input_delay* constraints. The timing results may give a false sense of security and you may get silicon timing failure. In addition, implementation tools may assume optimistic delay to the ports.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the path from the port to the data pins of the sequential cells, inout, or output port. This violation message arises because there is a valid timing path starting from the port, but an input delay constraint is not applied on the port.

To resolve this violation, ensure:

- The `clock`, `set_dont_touch_network`, `set_ideal_network`, `set_case_analysis` constraints are not applied on this port.
- The timing path from the port is not negated by the `set_false_path` constraint (only if the `tc_ignore_te` parameter is set to `no` and the `ignore_io_if_fp` parameter is set to `yes`)
- The timing path to another port does not have the `set_max_delay/`
`set_min_delay` constraint set

Message 2

The following message appears when the input/inout port `<port-name>` of the design/block `<name>` does not have any `set_input_delay` constraint set and the input/inout port reaches the data pin of a sequential cell, triggered by clocks `<clk-name-list>`, in its fan-out:

```
[Inp_Del 01b_03] [WARNING/INFO] Input delay constraint not set on
Port "<port-name>" of design/block <name> and input delay needs
to be specified with clock(s) <clk-name-list>
```

Potential Issues

Implementation tools might assume an undesired delay value for the path from the port to the flip-flop sampled by the clock. It could lead to incorrect timing analysis for the port.

Consequences of Not Fixing

This is a design error because the timing analysis results are not valid without proper `set_input_delay` constraints. The timing results may give a false sense of security and you may get silicon timing failure. In addition, implementation tools may assume optimistic delay to the ports.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the path from the port to the data pins of the sequential cells and the clock-paths to the clock pins of the sequential cells.

This violation arises because there is a valid timing path starting from the port, but an input delay constraint is not applied on the port.

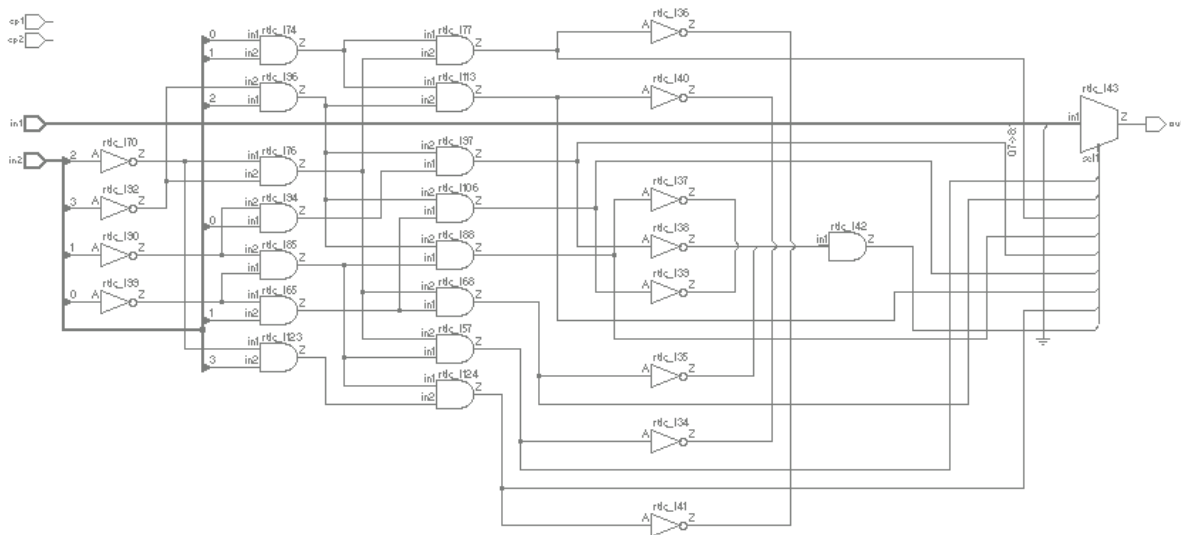
To resolve this violation, ensure:

- The `clock`, `set_dont_touch_network`, [set_ideal_network](#), [set_case_analysis](#) constraints are not applied on this port.
- The timing path from the port is not negated by the [set_false_path](#) constraint (only if the [tc_ignore_te](#) parameter is set to `no` and the [ignore_io_if_fp](#) parameter is set to `yes`)
- The timing path to another port does not have the [set_max_delay](#)/[set_min_delay](#) constraint set

Example Code and/or Schematic

Example 1[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top .sdc is as follows:

```

top.sdc
1 create_clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } [get_ports cp1]
2 create_clock -name clk2 -period 10.000000 -waveform { 0.000000 5.000000 } [get_ports cp2]
3 set_input_delay 3.000000 -clock {clk1} -add_delay {in1[1] in1[2] in1[3]}
4 set_input_delay 3.000000 -clock clk1 [get_ports in2*]
5 set_input_delay 3.000000 -clock clk2 [get_ports in2*]
6

```

The violation message appears because the input delay constraint has not been defined on the in1 [7 : 4] port.

Example 2

Test Case Files Not Available

In the following example, the *Inp_Del01b* rule reports a violation for the in1 and in2 ports because both ports are not constrained with

set_input_delay and there exists a path to the sequential data pin from *in1* and a combinational path to *out* from *in2*.

```
--Design begin-
Input in1, in2;
Input clk;
Output out;
Flop f1( in1, clk, w1 );
Assign out = in2;
--Design end-

--SDC begin-
Create_clock -period 10 clk
--SDC end-
```

Default Severity Label

Warning

Rule Group

Inp_Del

Reports and Related Files

None

Inp_Del01c

Timing analysis not required for the input/inout ports

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Inp_Del01c* rule reports input/inout ports with a *set_input_delay* constraint set and every fan-out of the input/inout ports satisfies the following conditions:

- Path from an input/inout port with a *create_clock* constraint.
- Path from an input/inout port with a *create_generated_clock* constraint.
- Path from an input/inout port with *set_case_analysis* or *set_disable_timing* constraints.
- Path from an input/inout port is hanging or reaching a black box.
- Path from an input/inout port reaches a sequential cell but does not hit its data pin.

If the *tc_allow_async_pins* rule parameter is set to *yes*, then the *Inp_Del01c* rule also considers the asynchronous pins of a sequential cell.

- The input/inout port with a *set_max_delay/set_min_delay* constraint reaches an output/inout port.

Rule run with parameters:

By default, the *Inp_Del01c* rule reports a violation for the ports connected to the scan enable pins of sequential elements. However, when you set the value of the *tc_at_speed_testing_mode* rule parameter to *yes*, then the *Inp_Del01c* rule treats the scan enable pins as the data pins of sequential elements.

If the value of the *tc_ignore_te* rule parameter is set to *no* and the value of the *ignore_io_if_fp* rule parameter is set to *yes*, then the *set_false_path* constraints are taken into consideration, if specified.

Parameter(s)

- *tc_at_speed_testing_mode*: Default value is no. Set the value to yes to include the asynchronous pins.
- *tc_ignore_te*: Default value is yes. Set the value to no to consider the *set_multicycle_path* constraints but the *set_false_path* constraints are considered only if the value of the *ignore_io_if_fp* parameter is set to yes.
ignore_io_if_fp: Default value is no. Set the value to yes to consider the *set_false_path* constraints else they are ignored.
In GuideWare2.0, the default value of *tc_ignore_te* is no and the default value of *ignore_io_if_fp* is yes.
- *tc_ignore_async_ports*: Default is unspecified and this rule does not ignore asynchronous ports. Specify the file name that contains the list of asynchronous ports, which this rule should ignore.

Constraint(s)

- *set_input_delay* (Optional): Use this constraint to define the arrival time relative to a clock.
- *create_clock/create_generated_clock* (Optional): Use this constraint to create clock at design object.
- *set_case_analysis* (Optional): Use this constraint to specify that a port or pin is at a constant logic value 1 or 0, or is considered with a rising or falling transition.
- *set_disable_timing* (Optional): Use this constraint to disable timing arcs of instances.
- *set_max_delay* (Optional): Use this constraint to specify a maximum delay for timing paths.
- *set_min_delay* (Optional): Use this constraint to specify a minimum delay for timing paths.
- *set_false_path* (Optional): Use this constraint to falsified paths from timing analysis.

Messages and Suggested Fix

Message 1

The following message appears when an input/inout port `<port-name>` has a corresponding `set_input_delay` constraint and one of the specified constraints `<constr>` is set on the input/inout port or on every fan-out path of the input/inout port:

[INFO] "`<constr>`" constraint has been set for the port "`<port-name>`" hence `set_input_delay` need not be specified

Where, `<constr>` can be `set_disable_timing`, `set_false_path`, `create_clock`, `set_max_delay/set_min_delay`.

Potential Issues

The message appears if your design has an input/inout port with a corresponding `set_input_delay` constraint and one of the specified constraints is set on the input/inout port or on every fan-out path of the input/inout port.

Consequences of Not Fixing

Timing analysis will be invalid without the proper `set_input_delay` constraint.

How to Debug and Fix

Along with the `constraint_type`, message tells the port for which rule is violating, hence either of the constraints can be eliminated to prevent violation message.

Message 2

The following message appears when an input/inout port `<port-name>` has a corresponding `set_input_delay` constraint (with no other specified constraint `<constr>`) and the input/inout port never reaches the data pin of a sequential cell:

[INFO] No data pin of any sequential cell is being reached from the port "`<port-name>`" hence `set_input_delay` need not be specified

Where, `<constr>` can be `set_case_analysis`, `set_disable_timing`, `set_false_path`, `create_clock`, `create_generated_clock`, `set_max_delay/set_min_delay`.

Potential Issues

The message appears if your design has an input/inout port which has a corresponding [set_input_delay](#) constraint (with no other specified constraint) and the input/inout port never reaches the data pin of a sequential cell.

Consequences of Not Fixing

Timing analysis will be invalid without the proper [set_input_delay](#) constraint.

How to Debug and Fix

This port never reaches to any D pin of the flip-flop, as one of the paths from D pin is shown in path highlight, so you can remove the redundant constraint [set_input_delay](#) from this port.

Message 3

The following message appears when a [set_input_delay](#) constraint is applied on an input/inout port `<port-name>` which is either constrained with the [set_case_analysis](#) constraint or connected to the supply (1 or 0):

```
[INFO] Constant value has been set for the port <port-name>
hence set_input_delay need not be specified
```

Potential Issues

The message appears if your design has a [set_input_delay](#) constraint, which is applied on an input/inout port which is either constrained with the [set_case_analysis](#) constraint or connected to the supply (1 or 0).

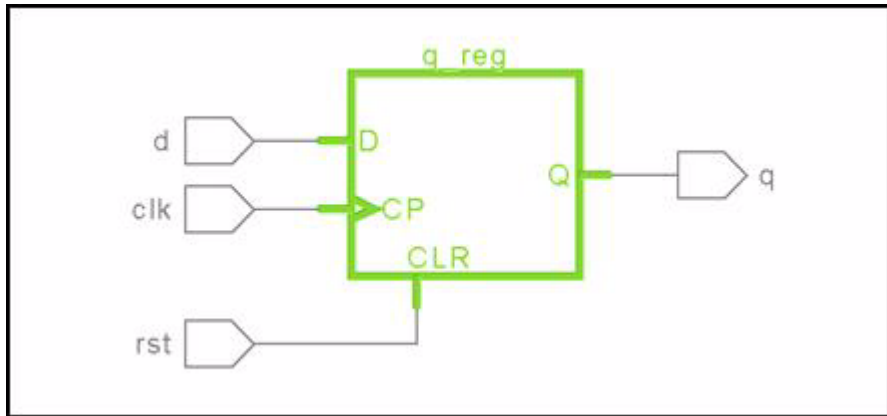
Consequences of Not Fixing

Timing analysis will be invalid without the proper [set_input_delay](#) constraint.

How to Debug and Fix

User can fix the violation by removing [set_input_delay](#) if port is supply port or removing either of constraints ([set_input_delay](#) or [set_case_analysis](#)).

Example Code and/or Schematic



In the above example, *set_input_delay* is applied at port d and simultaneously *set_case_analysis* is applied.

This rule violates and if you remove either *set_case_analysis* or *set_input_delay* then violation will not be reported.

Default Severity Label

Info

Rule Group

Inp_De1

Reports and Related Files

None

Inp_Del02

Reports inconsistency in min/max delay value of input constraint

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *Inp_Del02* rule reports inconsistency between the delay value and the clock period for a given *set_input_delay* constraint. The *Inp_Del02* rule checks if the delay value is less than the clock period.

However, if the *set_max_delay* constraint is applied on a port, the value of this constraint takes precedence over the values specified by the *set_input_delay* (SID) constraint.

The rule reports a violation if:

- Min (SID -min) is not specified and SID -max > period
- Max (SID -max) is not specified and SID -min > period
- SID -max > period > SID -min
- SID -max > SID -min > period
- SID -min > period > SID -max
- SID -min > SID -max
- SID -min > SID -max > period

Conditions for the above-mentioned Constraints:

- SID -max is value specified with option -max for a given *set_input_delay*.
- SID -min is value specified with option -min for a given *set_input_delay*.
- The period is equal to the clock period of clock for which *set_input_delay* is specified. The period = 0.5 * clock_period if clock_fall is specified with *set_input_delay* constraint. This is only valid when there is no clock inversion. However, if clock inversion is taking place and you have not specified clock_fall, the period = 0.5 * clock_period

***Inp_Del02* and *set_multicycle_path* constraint**

- The *Inp_Del02* rule considers *set_multicycle_path* constraint if multi-cycle path is applied between the port and the clock with parameter *tc_ignore_te* set to no.
- The *Inp_Del02* rule takes the period = MCP_multiplier * clock_period if *set_multicycle_path* is applied on the given port and clock specified in constraint *set_input_delay*.

Rule Exception(s)

- The *Inp_Del02* rule ignores the *set_input_delay* constraint specified without the clocks.
- It ignores the port specified with *set_input_delay* constraint if all the paths from port to clocks are falsified with parameter *tc_ignore_te* and *ignore_io_if_fp* values set as yes and no, respectively.

Parameter(s)

None

Constraint(s)

- *set_input_delay* (Optional): Use this constraint to define the arrival time relative to a clock.
- *create_clock/create_generated_clock* (Optional): Use this constraint to create clock at design object.

Messages and Suggested Fix

Message 1

The following message appears when an input port *<port-name>* has -min argument value equal to or greater than the clock period of the corresponding clock *<clk-name>* set using the -clock argument in the *set_input_delay* constraint:

[WARNING] Input -min delay value set on port '*<port-name>*' is greater than clock period of clock *<clk-name>*

Potential Issues

The violation message appears if your design has an input port with -min

argument value that is equal to or greater than the clock period of the corresponding clock.

Consequences of Not Fixing

If you do not fix this violation, meeting timing is very difficult if the delay values seem higher than the associated clock period.

How to Debug and Fix

Along with this violation message, rule highlights the corresponding constraints in the SDC file. Look at the line number and change the value so that period is more than `SID -min`.

Message 2

The following message appears when an input port `<port-name>` has `-max` argument value equal to or greater than the clock period of the corresponding clock `<clk-name>` set using the `-clock` argument in the [set_input_delay](#) constraint:

[WARNING] Input `-max` delay value set on port '`<port-name>`' is greater than clock period of clock `<clk-name>`

Potential Issues

The violation message appears if your design has an input port with `-max` argument value equal to or greater than the clock period of the corresponding clock.

Consequences of Not Fixing

If you do not fix this violation, meeting timing is very difficult with higher delay values than the associated clock period.

How to Debug and Fix

Along with this violation message, rule highlights the corresponding constraints in the SDC file. Look at the line number and change the value so that period is more than `SID -max`.

Message 3

The following message appears when an input port `<port-name>` has `-min` argument value greater than the corresponding `-max` argument value in [set_input_delay](#) constraints:

[WARNING] Input `-min` delay value is greater than `-max` delay on port '`<port-name>`'

Potential Issues

The violation message appears if your design has an input port with `-min` argument value greater than the corresponding `-max` argument value in [set_input_delay](#) constraints.

Consequences of Not Fixing

If you do not fix this violation, meeting timing is very difficult with higher delay values than the associated clock period.

How to Debug and Fix

Make sure to change the values so that SID `-min` is less than the SID `-max`.

Message 4

The following message appears when an input port `<port-name>` has a typical delay value equal to or greater than the clock period of the corresponding clock `<clk-name>` set using the `-clock` argument in the [set_input_delay](#) constraint:

[WARNING] Input delay value set on port '`<port-name>`' is greater than clock period of clock `<clk-name>`

Potential Issues

The violation message appears if your design has an input port with a typical delay value equal to or greater than the clock period of the corresponding clock.

Consequences of Not Fixing

Meeting timing is difficult if the delay value is higher than the associated clock period.

How to Debug and Fix

Make sure to change the values so that `delay_value` is less than the associated clock period.

Example Code and/or Schematic

For example, constraints are applied on data port and clock port as given below:

```
Create_clock -name c1 -period 20 c
```

```
set_input_delay 5 -clock c1 d
```

In the above constraints, rule checks the inconsistency between delay value i.e. 5 and the clock period i.e. 10 of clock c1.

Consider the following case, where the delay value is greater than the clock period but is still valid:

```
create_clock -name clk1 -period 10 -waveform {0 5}
```

```
[get_ports clock]
```

```
set_input_delay 11 -clock clk1 [get_ports data_in]
```

This rule reports a violation because the delay_value(11) is greater than the clock_period.

Default Severity Label

Warning

Rule Group

Inp_De1

Reports and Related Files

None

Inp_Del03

Input constraint associated with wrong clock

The *Inp_Del03* rule runs the *Inp_Del03a* and *Inp_Del03b* rules.

Inp_Del03a

Reports input constraint associated with incorrect or incomplete set of clocks

When to Use

To validate input delays for associated clocks. This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Inp_Del03a* rule reports incorrect or missing *set_input_delay* constraints. This rule reports the following cases:

- *set_input_delay* constraints that are set on an input port with reference to a clock that is not a sampling clock for the input port.
- Missing *set_input_delay* constraints for a sampling clock of an input port. The *Inp_Del01* rule reports input ports where the *set_input_delay* constraint has not been set.
- A combination of the above two conditions.

When *set_input_delay* constraint has not been set on either the create clock or generated clock, this rule reports the create clock or generated clock that is a sampling clock of the input port.

For cascading generated clocks, this rule reports the create clock or generated clock when the clocks in the cascade do not have a *set_input_delay* constraint set.

Prerequisites

The *Inp_Del03a* rule requires the synthesizable description of all library cells. Hence, use the SpyGlass Library Compiler feature before running this rule. Refer to the *SpyGlass Explorer User Guide* for details on the SpyGlass Library Compiler feature.

Rule Exceptions

The *Inp_Del03a* rule does not report a wrong clock violation if that clock is strongly related to the specified clock for which the *set_input_delay* constraint is defined. A clock pair is considered to be strongly related if it satisfies any of the following conditions:

- One clock is generated from the other clock

- One clock of the pair is related to the third clock, and the third clock is related to the other clock of the pair. For example, if the clock `clk1` of the pair is related to a third clock `clk3` and `clk3` is related to other clock `clk2` of the pair, `clk1` and `clk2` are assumed to be strongly related.

This rule does not consider [set_input_delay](#) constraints that are normally overwritten by the Synthesis Timing Analysis (STA) tools.

In case of multiple clocks defined on the same object using the `-add` option of the [create_clock](#) or [create_generated_clock](#) constraint, the [Inp_Del03a](#) rule retains only the last clock defined without the `-add` option and any clocks defined with the `-add` option after the last clock.

If the input port has more than one sampling clock, use the `-add_delay` option in the [set_input_delay](#) constraints for the second and subsequent clocks.

This rule reports the same or fewer rule violations than the [Inp_Del03b](#) rule because this rule does not consider overwritten [set_input_delay](#) constraints.

Parameter(s)

- [tc_allow_async_pins](#): Default is no. This indicates that the [Inp_Del03a](#) rule considers only those pins of a sequential element that are synchronous with respect to the clocks of the sequential element. Set the value to yes to also consider the asynchronous pins of sequential cells.
- [del03_allow_setdelay](#): Default is no. Set the value to yes to ignore those port-clock pairs for which a `set_max_delay` constraint is defined.
- [chip](#): Default is unspecified. This indicates that the [Inp_Del03a](#) rule considers matching timing characteristics between real and virtual clocks. Therefore, the [Inp_Del03a](#) rule does not report a missing input constraint violation for a real clock when an input constraint has been set for a matching virtual clock.
- [tc_ignore_te](#) and [ignore_io_if_fp](#) rule parameters is set to no, or the value of the [tc_ignore_te](#) rule parameter is set to yes, the [Inp_Del03a](#) rule ignores those port-clock pairs for which the [set_false_path](#) or [set_clock_groups](#) constraint is defined.

In GuideWare2.0, the default value of [tc_ignore_te](#) is no and the default

value of *ignore_io_if_fp* is yes.

- *tc_ignore_async_ports*: Default is unspecified and this rule does not ignore asynchronous ports. Specify the file name that contains the list of asynchronous ports, which this rule should ignore.
- *tc_honor_virtual_clk*: Default is no. Set this parameter to yes to honor the *set_input_delay*/*set_output_delay* defined with virtual clocks that do not have any matching real clock in the SDC file.

Constraint(s)

SDC

- *create_clock* (Optional): Use to create a clock.
- *create_generated_clock* (Optional): Use to create a clock.
- *set_input_delay* (Optional): Use to define the arrival time relative to a clock.
- *set_max_delay* (Optional): Use define the maximum delay for timing paths.

Messages and Suggested Fix

If the name of the port, *<port-name>*, is listed as an asynchronous port through the *tc_ignore_async_ports* parameter, the severity of each message listed in this section is INFO.

Message 1

The following message appears when *set_input_delay* constraints are set on the input port *<port-name>* using clocks *<clk-name-list>* when these clocks are not the sampling clocks for the input port:

```
[I np_Del 03a_01][WARNING] Incorrect input constraint for port
'<port-name>' - incorrect for clock(s) <clk-name-list>
```

Or,

```
[I np_Del 03a_02][INFO] Incorrect input constraint for port
'<port-name>' - incorrect for clock(s) <clk-name-list>
```

Potential Issues

A timing path does not exist from the input port to any flip-flop sampled

with the clocks. It implies that input delay need to be updated with the correct set of clocks.

Consequences of Not Fixing

While the input is being sampled with respect to one clock, the delay has been specified with respect to another clock. This can result in difficulty in meeting timing. Even if timing is met, due to false path or multi-cycle path specifications, the actual design might not be able to meet the timing.

How to Debug and Fix

A clock is classified as *incorrect* when the clock:

- specified is a virtual clock and the timing-characteristics of any of the sampling clocks are not matching with the virtual clock.
- is a real clock but it is not sampling any sequential element that has a data pin connected to the reported port through combinational logic.

For incorrect clock, no schematic is generated because the path does not exist. Therefore, to debug this violation, you need to manually validate the timing-path from port to clock.

Message 2

The following message appears when no *set_input_delay* constraint has been set on the input port *<port-name>* referencing its sampling clocks *<clk-name-list>*:

```
[I np_DeI 03a_03] [WARNI NG/I NFO] Mi ssi ng i nput constraint for port '<port-name>' - mi ssi ng for cl ock(s) <cl k-name-l i st>
```

Potential Issues

A path exists from the port to a flip-flop sampled with clocks, but an input delay constraint is not provided. Implementation tools might assume an undesired delay value during timing analysis of the path.

Consequences of Not Fixing

While the input is being sampled with respect to one clock, the delay has been specified with respect to another clock. This can result in difficulty in meeting timing. Even if timing is met, due to false path or multi-cycle path specifications, the actual design might not be able to meet the timing.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic highlights the data path joining the port to the data pin of the sequential element. The schematic also shows the clock path of the sampling sequential element.

Message 3

The following message appears when *set_input_delay* constraints on the input port *<port-name>* referencing its sampling clocks *<clk-name-list2>* are missing while *set_input_delay* constraints are set on the input port using clocks *<clk-name-list1>* that are not its sampling clocks:

```
[WARNING/INFO] Incorrect/Missing input constraint for port
'<port-name>' - incorrect for clock(s) <clk-name-list1> and
missing for clock(s) <clk-name-list2>
```

Potential Issues

Some paths do not have an associated input delay constraints. It might lead to undesired delay assumptions by implementation tools during the timing analysis of paths. In addition, there are some input delay constraints that do not apply to any paths from the port. Therefore, the constraints need to be updated appropriately.

Consequences of Not Fixing

While the input is being sampled with respect to one clock, the delay has been specified with respect to another clock. This can result in difficulty in meeting timing. Even if timing is met, due to false path or multi-cycle path specifications, the actual design might not be able to meet the timing.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic highlights the data path joining the port to the data pin of the sequential element. The schematic also shows the clock-paths of the sampling sequential element. Review the reported constraints in the SDC file for missing or incorrect clocks.

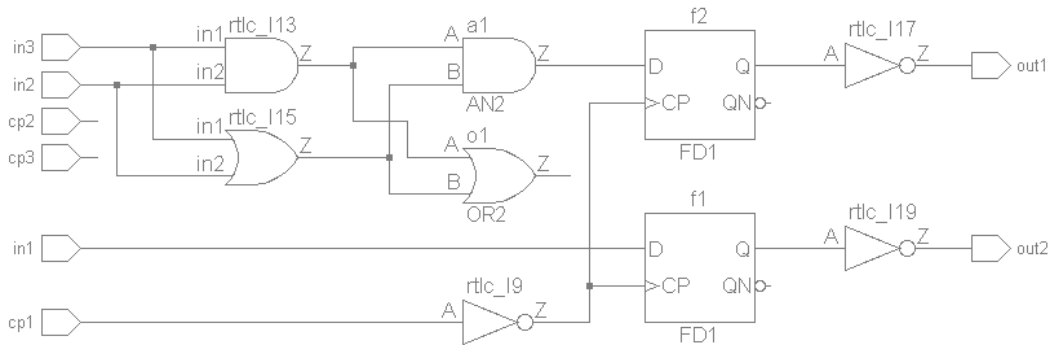
Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The

schematic of the top module is as follows:



The top.sdc is as follows:

```
top.sdc top.sdc (3)
set_input_delay -clock clk2 35.000000 in2
set_input_delay -clock clk3 35.000000 in3
```

The violation message appears because `set_input_delay` is defined with reference to the wrong clock `clk3`. It should be `clk1`. In the `top.sdc` file, the corresponding constraint is highlighted.

Example 2

Test Case Files Not Available

In the following example, the `Inp_Del03a` rule reports a violation for `in` because:

1. `in` should be constrained with `set_input_delay` for `clk1`.
2. `in` should not be constrained with `set_input_delay` for `clk2`.

--Design begin--

```

Input in, clk1, clk2;
Output out;
Flop f1( in, clk1, out );
--Design end-

--SDC begin-
Create_clock -period 10 clk1
Create_clock -period 12 clk2
Set_input_delay 1.0 -clock clk2 in
--SDC end-

```

Example 2

Test Case Files Not Available

The *Inp_Del03a* rule does not consider [set_input_delay](#) constraints that are normally overwritten by the Synthesis Timing Analysis (STA) tools. In the following snippet, this rule considers only the second constraint specification because STA tools overwrite the first constraint specification with the second constraint specification:

```

set_input_delay <value1> I1 -clock C1
set_input_delay <value2> I1 -clock C2

```

Default Severity Label

Warning

Rule Group

Inp_Del

Reports and Related Files

None

Inp_Del03b

Reports input constraint associated with wrong or incomplete set of clocks

When to Use

Use this rule in RTL, Pre-layout and Post-layout phases.

Description

The *Inp_Del03b* rule reports incorrect *set_input_delay* constraints.

The *Inp_Del03b* rule reports the following cases:

- *set_input_delay* constraints that are set on an input port with reference to a clock that is not a sampling clock for the input port.
- Missing *set_input_delay* constraints for an input port.

The *Inp_Del03b* rule considers all *set_input_delay* constraints including those normally overwritten by the STA tools.

Refer to Example Code and/or Schematic section for further details.

Difference between Inp_Del03a and Inp_Del03b

The *Inp_Del03a* rule reports same or fewer rule-violations than the *Inp_Del03b* rule as the *Inp_Del03a* rule does not consider overwritten *set_input_delay* constraints whereas the *Inp_Del03b* rule considers *set_input_delay* overwritten constraints.

When *set_input_delay* constraint has not been set on either the create clock or generated clock, the *Inp_Del03b* rule reports the create clock or generated clock that is a sampling clock of the input port.

For cascading generated clocks, the *Inp_Del03b* rule reports the create clock or generated clock when the clocks in the cascade do not have a *set_input_delay* constraint set.

Rule Exception(s)

The *Inp_Del03b* rule does not report a wrong clock violation if that clock is strongly related to the specified clock (clock pair) for which the *set_input_delay* constraint has been defined. A clock pair is considered to be strongly related if it satisfies any of the following conditions:

- One clock is generated from the other clock.

- One clock of the pair is related to the third clock, and the third clock is related to the other clock of the pair. For example, if the clock `clk1` of the pair is related to a third clock `clk3` and `clk3` is related to other clock `clk2` of the pair then, `clk1` and `clk2` are assumed to be strongly related.

If there no *set_input_delay* constraints defined, this rule does not report violation for missing *set_input_delay* constraint. This check is performed by *Inp_Del01*.

Parameter(s)

- *tc_allow_async_pins*: Default value is `no`. Set the value to `yes` to consider the asynchronous pins of sequential cells.
- *tc_ignore_te*: Default value is `yes`. For *ignore_io_if_fp* the default value is `no`. When the value of both the *tc_ignore_te* and *ignore_io_if_fp* rule parameters is `unset`, or the value of the *tc_ignore_te* rule parameter is `unset`, then the *Inp_Del03b* rule ignores those port-clock pairs for which the *set_false_path* or *set_clock_groups* constraint is defined.

In GuideWare2.0, the default value of *tc_ignore_te* is `no` and the default value of *ignore_io_if_fp* is `yes`.

- *chip*: Default value is `unspecified`. Set the value to a `string` value to specify the name of the design unit which corresponds to the chip level. When the *chip* parameter is not set, the *Inp_Del03b* rule considers matching timing characteristics between real and virtual clocks. Thus the *Inp_Del03b* rule does not report a missing input constraint violation for real clock when an input constraint has been set for a matching virtual clock.
- *tc_ignore_async_ports*: Default is `unspecified` and this rule does not ignore asynchronous ports. Specify the file name that contains the list of asynchronous ports, which this rule should ignore.
- *tc_honor_virtual_clk*: Default is `no`. Set this parameter to `yes` to honor the *set_input_delay*/*set_output_delay* defined with virtual clocks that do not have any matching real clock in the SDC file.

Constraint(s)

- *set_input_delay* (Mandatory): Use this constraint to specify the input delay of the port/pin with respect to a clock.
- *set_false_path* (Optional): Use this constraint to remove timing constraints for the specified path(s).
- *set_clock_groups* (Optional): Use this constraint to specify relationship between clocks in different groups, relationship could be one of asynchronous, *Physically_exclusive* or *logical_exclusive*.
- *create_clock* (Mandatory): Use this constraint to specify a clock on a port/pin.
- *create_generated_clock* (Mandatory): Use this constraint to specify a generated clock on a port/pin.
- *set_clock_sense/set_sense* (Optional): Use to specify the clock propagation conditions.

Messages and Suggested Fix

If the name of the port, *<port-name>*, is listed as an asynchronous port through the *tc_ignore_async_ports* parameter, the severity of each message listed in this section is INFO.

Message 1

The following message appears when *set_input_delay* constraints are set on input port *<port-name>* using clocks *<clk-name-list>*, when these clocks are not the sampling clocks for the input port:

```
[I np_Del 03b_01] [WARNING] Incorrect input constraint for port
'<port-name>' -incorrect for clock(s) <clk-name-list>
```

Or,

```
[I np_Del 03b_02] [INFO] Incorrect input constraint for port
'<port-name>' -incorrect for clock(s) <clk-name-list>
```

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

Either the sampling clock is same as the generating clock, which is feeding data to the input port, or they are different. If they are different then it is

expected that some exception like *set_false_path* or *set_clock_groups* constraints should cover this crossing. In that case, rule does not violate if particular parameters are ON.

Alternatively, if the clocks are same then clocks specified in the *set_input_delay* constraint should match sampling clock. So, either *set_input_delay* constraint needs a fix or you need to put *set_false_path* or *set_clock_groups* constraints between the clocks.

If you do not fix this violation, then applied timing constraint (in this case *set_input_delay* constraint) is invalidated and it is as good as timing not done, which brings in the risk of “timing not met” for the design, leading to chip failure.

How to Debug and Fix

set_input_delay constraints are highlighted in the violation message and in the SDC file.

One can look at the clocks in the *set_input_delay* constraints and the clock sampling the port in question.

Schematics highlight the path to the sequential element. Identify the correct clock and modify the *set_input_delay* constraint or check if there is a need to specify *set_false_path* constraint or *set_clock_groups* constraint between the clocks referred in *set_input_delay* constraint and the sampling clock.

Message 2

The following message appears when no *set_input_delay* constraint has been set on the input port *<port-name>* referencing its sampling clocks *<clk-name-list>*:

```
[I np_De1 03b_03] [WARNING/INFO] Missing input constraint for port
'<port-name>' - missing for clock(s) <clk-name-list>
```

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

While the input is being sampled with respect to one clock, the delay has been specified with respect to another clock. This is as good as timing not done, which brings in the risk of “timing not met”. Even if the timing is met (due to false path/multi-cycle path specifications), the risk of “timing not met” might occur in the actual design, leading to chip failure.

How to Debug and Fix

Identify the correct clock and modify the [set_input_delay](#) constraint.

Message 3

The following message appears when [set_input_delay](#) constraints on the input port `<port-name>` referencing its sampling clocks `<clkname-list2>` are missing while [set_input_delay](#) constraints are set on the input port using clocks `<clk-name-list1>` that are not its sampling clocks:

```
[WARNING/INFO] Incorrect/Missing input constraint for port
'<port-name>' -incorrect for clock(s) <clk-name-list1> and
missing for clock(s) <clk-name-list2>
```

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

If you do not fix this violation, while the input is being sampled with respect to one clock, the delay has been specified with respect to another clock. This is as good as timing not done, which brings in the risk of “timing not met”. Even if the timing is met (due to false path/multi-cycle path specifications), the risk of “timing not met” might occur in the actual design, leading to chip failure.

How to Debug and Fix

[set_input_delay](#) constraints are highlighted in the message and in SDC file. One can look at the clocks in the [set_input_delay](#) constraints and the clock sampling the port in question. Schematic highlights the path to the sequential element. Identify the correct clock and modify the [set_input_delay](#) constraint.

Example Code and/or Schematic

For the following snippet, the `Inp_Del03b` rule reports a violation.

```
module top (in1, d, CP, out);
    input in1, d, CP;
    output out;

    reg q;
```

```
always @(posedge CP)
    q<=d;

    assign out= q;
endmodule
```

```
create_clock -name clk -period 11 CP
create_clock -name vclk -period 10
set_input_delay 10 -clock vclk d -add
```

The *Inp_Del03b* rule considers both of the following constraints even though the Synthesis/Timing Analysis tools will overwrite the first constraint specification with the second constraint specification:

```
set_input_delay <value1> I1 -clock C1
set_input_delay <value2> I1 -clock C2
```

Default Severity Label

Warning

Rule Group

Inp_De1

Reports and Related Files

None

Inp_Del04

Reports incompletely defined input delay constraint

When to Use

Use this rule in RTL phase, pre-layout phase, and post-layout phase for timing analysis.

Description

The *Inp_Del04* rule reports *set_input_delay* constraints set on input or inout ports but defined incompletely.

As required by this rule, a properly defined *set_input_delay* constraint must have:

- Both or none of `-min` and `-max` values specified.
- Both or none of `-rise` and `-fall` values specified.

Rule Exception(s)

The *Inp_Del04* rule ignores missing `-min` option for worst-case SDC when the *tc_ignore_min* rule parameter is set.

Parameter(s)

- *tc_ignore_min*: Default value is no. Set the value to yes to ignore missing `-min` option in SDC command.

Constraint(s)

- *set_input_delay* (Mandatory): Use this constraint to specify delay value on a port with respect to a clock.

Messages and Suggested Fix

Message 1

The following message appears when *set_input_delay* constraint is not defined with all options `<option-list>` on a port `<port-name>`:

[WARNING] Set_input_delay option(s) `<option-list>` not specified for port `<port-name>`

In the above message, `<option-list>` can be `(-max, -rise)`, `(-max,`

-fall), (-min, -rise), and (-min, -fall).

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

If you do not fix this violation, the STA tool may arbitrarily assign an undesired value to the unspecified option in the [set_input_delay](#) constraint.

Values for the unspecified options are inferred by the STA tools and the inferred value can be different from what was desired.

How to Debug and Fix

Provide all the missing options and values.

Message 2

The following message appears for a [set_input_delay](#) constraint that is not specified with all options `<option-list>` related to the clock `<clk-name>` on a port:

```
[WARNING] Set_input_delay option(s) <option-list> not specified
for port <port-name>, relative to clock <clk-name>
```

In the above messages, `<option-list>` can be (-max, -rise), (-max, -fall), (-min, -rise), and (-min, -fall)

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

The STA tool may arbitrarily assign an undesired value to the unspecified option in the [set_input_delay](#) constraint.

Values for the unspecified options are inferred by the STA tools and the inferred value can be different from what was desired.

How to Debug and Fix

Provide missing options and values.

Example Code and/or Schematic

Consider the following snippet.

```
create_clock -name clk -period 10.000000 -waveform { 0.000000
```

```
5.000000 } {CLK}
set_input_delay 5.000000 -min -clock clk [get_ports din*]
set_input_delay 8.000000 -max -clock clk [get_ports sel*]
```

The snippet for the design is as follows:

```
module mux (din, sel, out, CLK);
    input [7:0] din;
    input [3:0] sel;
    input CLK;
    output out;
    reg out, qout;
    always @(din or sel)
        case(sel)
            4'b0000 : qout = din[0];
            4'b0010 : qout = din[1];
            4'b0011 : qout = din[2];
            4'b0100 : qout = din[3];
            4'b0101 : qout = din[4];
            4'b0110 : qout = din[5];
            4'b0111 : qout = din[6];
            4'b1000 : qout = din[7];
            default : qout = 1'b0;
        endcase

    always @(CLK or qout)
        if (CLK)
            out = qout;
```



```
endmodule
```

Hence, the *Inp_De104* rule reports violation messages.

Default Severity Label

Warning

Rule Group

Inp_De1

Reports and Related Files

None

Inp_Del05

Reports Input constraint defined with clocks other than virtual clocks

When to Use

Use this rule in RTL phase, Pre-layout phase, Post-layout phase.

Description

The *Inp_Del05* rule reports input constraints specified with clock other than virtual clocks.

Parameter(s)

- *chip*: Default value is unspecified. Set the value to a string value to specify the name of the design unit which corresponds to the chip-level.

Constraint(s)

- *set_input_delay* (Mandatory): Use this constraint to specify delay value on a port with respect to a clock.

Messages and Suggested Fix

The following message appears when an input constraint is specified for input port *<port-name>* with a real clock:

[WARNING] Input constraint for port '*<port-name>*' is defined with real clock '*<clock-name>*' as reference

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

This is a methodology choice.

How to Debug and Fix

The *set_input_delay* constraint is highlighted. Check the clock referred in the *set_input_delay* and modify either the constraint or the clock.

Example Code and/or Schematic

For the following design, the *Inp_De/05* rule reports a violation because

```
module mux(din,sel,qout, CLK1, CLK2);
```

```
  input [7:0]din;
```

```
  input [3:0]sel;
```

```
  input CLK1,CLK2;
```

```
  output qout;
```

```
  reg qout;
```

```
  always @(din or sel)
```

```
  case(sel)
```

```
    4'b0000 : qout = din[0];
```

```
    4'b0010 : qout = din[1];
```

```
    4'b0011 : qout = din[2];
```

```
    4'b0100 : qout = din[3];
```

```
    4'b0101 : qout = din[4];
```

```
    4'b0110 : qout = din[5];
```

```
    4'b0111 : qout = din[6];
```

```
    4'b1000 : qout = din[7];
```

```
    default : qout = 1'b0;
```

```
  endcase
```

```
endmodule
```

The constraints defined in the SDC file are as follows:

```
create_clock -name Clk3 -period 10.000000 -waveform {  
0.000000 5.000000 } {CLK1}
```

```
set_input_delay 9.000000 -clock Clk3 [get_ports din*]
```

Default Severity Label

Warning

Rule Group

Inp_De1

Reports and Related Files

No report and related file

Inp_Del07

Reports correct Input constraints relative to clock period

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *Inp_Del07* rule reports such ports where the input delay value is more than the user-specified percentage of the corresponding clock period.

Prominent Features:

- The *Inp_Del07* rule specifies the names of the input ports that are checked using the *io07_ports* parameter.
- By default, the *Inp_Del07* rule reports those ports where the input delay value is more than 80% of the corresponding clock period. Use the *inp_percent* rule parameter to set a different limit (as the specified percentage of the clock period) or use the *inp_delay_margin* rule parameter to set a different limit (as clock period value minus the specified value).
- If a *set_input_delay* command has the *-clock_fall* option specified, the delay is compared with the part of the clock waveform starting at the falling edge. Thus, the delay is compared with half the clock period for default waveforms.
- The *Inp_Del07* rule considers multi-cycle paths set between the input port and the clock. For example, if a *set_multicycle_path* command is set with a multiplier of 2, then the *Inp_Del07* rule uses 2x of the associated clock period for comparison. The *Inp_Del07* rule considers all paths from the input port to the clock.
- If more than one *set_multicycle_path* command is specified with the *-setup* option, then the *Inp_Del07* rule considers the *set_multicycle_path* command that contains the least value of *path_multiplier*.
- If the *set_multicycle_path* command is specified with the *-hold* option then it is ignored.
- If the value of the *tc_ignore_te* parameter is set to *no* and the value of the *ignore_io_if_fp* parameter is set to *yes*, then the *Inp_Del07* rule considers the *set_false_path* constraints, if specified.

Parameter(s)

- *io07_ports*: Default value is *, it signifies all the ports are considered. Set the value to a list of string values to specify the port names directly.
- *inp_percent*: Default value is 80. Set this parameter to a positive float value between 0 and 100 to specify the limit of clock period.
- *inp_delay_margin*: Default value is 0. Set this parameter to a positive float value to specify the maximum delay margin between the clock period and input delay.
- *tc_ignore_te*: Default value is yes. Set the value to yes, no, 1, 0 to control the behavior of rules regarding checks related to *set_false_path* and *set_multicycle_path* settings.
ignore_io_if_fp: Default value is no. Set the value to yes, no, 1, 0 to control the behavior of some rules regarding checks related to the *set_false_path* constraints.
In GuideWare2.0, the default value of *tc_ignore_te* is no and the default value of *ignore_io_if_fp* is yes.

Constraint(s)

- *set_false_path* (Optional): Use this constraint to remove timing constraints for the specified path(s).
- *set_input_delay* (Mandatory): Use this constraint to specify the input delay of the port/pin with respect to a clock.
- *set_output_delay* (Optional): Use this constraint to set the output path delay value.
- *set_multicycle_path* (Optional): Use this constraint to define the multi-cycle paths.

Messages and Suggested Fix

Message 1

The following message appears when a *set_input_delay* value *<value1>* specified for input port *<port-name>* is more than \$inp_percent

percent of the corresponding clock period `<value2>`:

[WARNING] Input delay `<value1>` for port "`<port-name>`" is more than the specified limit, `$inp_percent%` of clock period `<value2>`

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

If you do not fix this violation, the specified delay value seems too low/high. This could result in difficulty in meeting timing for either this block/next block or the top level.

How to Debug and Fix

Specify the input delay value on the specified limit. You can make a change to `set_input_delay` values if needed.

Message 2

The following message appears when a `set_input_delay` value `<value1>` specified for input port `<port-name>` is more than the difference `<value2>` between the clock period and `$inp_delay_margin` value:

[WARNING] Input delay `<value1>` for port "`<port-name>`" is more than the specified limit, clock period minus `inp_delay_margin` `<value2>`

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

If you do not fix this violation, the specified delay value seems too low/high. This could result in difficulty in meeting timing either for this block/next block or the top level.

How to Debug and Fix

Specify the input delay value on the specified limit. User can make a change to `set_input_delay` values if needed.

Message 3

The following message appears when a `set_input_delay` value `<value1>` specified with options `<option-list>` for input port `<port-name>` is

more than `$inp_percent` percent of the corresponding clock period `<value2>`:

[WARNING] Input delay `<value1>` (specified with `<option-list>` options) for port "`<port-name>`" is more than the specified limit, `$inp_percent%` of clock period `<value2>`

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

If you do not fix this violation, the specified delay value seems too low/high. This could result in difficulty in meeting timing either for this block/next block or the top-level.

How to Debug and Fix

Specify the input delay value on the specified limit. You can make a change to `set_input_delay` values, if needed.

Message 4

The following message appears when a `set_input_delay` value `<value1>` specified with options `<option-list>` for input port `<port-name>` is more than the difference `<value2>` between the clock period and `$inp_delay_margin` value:

[WARNING] Input delay `<value1>` (specified with `<option-list>` options) for port "`<port-name>`" is more than the specified limit, clock period minus `inp_delay_margin` `<value2>`

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

The specified delay value seems too low/high. This could result in difficulty in meeting timing either for this block/next block or the top level.

How to Debug and Fix

To fix the violation specify the input delay value on the specified limit. User can make a change to `set_input_delay` values if needed.

Example Code and/or Schematic

The `Inp_Del07` rule reports a violation message for the following the

snippet.

```
module mux(din,sel,qout, CLK1, CLK2);
    input [7:0]din;
    input [3:0]sel;
    input CLK1,CLK2;
    output qout;
    reg qout;

    always @(din or sel)
        case(sel)
            4'b0000 : qout = din[0];
            4'b0010 : qout = din[1];
            4'b0011 : qout = din[2];
            4'b0100 : qout = din[3];
            4'b0101 : qout = din[4];
            4'b0110 : qout = din[5];
            4'b0111 : qout = din[6];
            4'b1000 : qout = din[7];
            default : qout = 1'b0;
        endcase

endmodule
```

The snippet for the SDC file is as follows:

```
create_clock -name Clk1 -period 10.000000 -waveform {
0.000000 5.000000 } \
{CLK1}
set_input_delay 9.000000 -clock Clk1 [get_ports din*]
```

The *io07_ports* parameter is set to *din*.

Default Severity Label

Warning

Rule Group

Inp_De1

Reports and Related Files

None

Inp_Del07a

Reports input constraints that are incorrect relative to a range of clock period

When to Use

To validate the input delay values for associated clock periods. This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Inp_Del07a* rule reports incorrect *set_input_delay* constraints. This rule reports the following *set_input_delay* constraints on ports specified using the *io07_ports* parameter:

- The input delay value is not within the range specified by the *inp_percent_min* and *inp_percent_max* parameters when the *io07a_range* parameter is not set or is set to *within*.
- The input delay value is not outside the range specified by the *inp_percent_min* and *inp_percent_max* parameters when the *io07a_range* parameter is set to *outside*.

If a *set_input_delay* command has the *-clock_fall* option specified, the delay is compared with the part of the clock waveform starting at the falling edge. Therefore, the delay is compared with half the clock period for default waveforms.

Multi-cycle Considerations

The *Inp_Del07a* rule considers multi-cycle paths set between the input port and clock. For example, if a *set_multicycle_path* command is set with a multiplier of 2, then the *Inp_Del07a* rule uses 2x of the associated clock period for comparison. This rule considers all paths from the input port to the clock.

If more than one *set_multicycle_path* command is specified with the *-setup* option, this rule considers the *set_multicycle_path* command that contains the least value of *path_multiplier*.

If the *set_multicycle_path* command is specified with the *-hold* option, it will be ignored.

If the *set_multicycle_path* command is specified and the *-clock_fall* option is also used in the *set_input_delay/set_output_delay*

constraint, the value of the clock period is calculated as follows:
(`path_multiplier` x `clk_period`) - (0.5 x `clk_period`)

If a multi-cycle path constraint is not provided for the path, the `path_multiplier` value is set to 1.

Parameter(s)

- `io07_ports`: Default is *. This indicates all ports are checked. Set this value port names to check specific ports.
- `io07a_range`: Default is `within`. This indicates the `Inp_Del07a` rule reports input or output delay values that are not within the range specified by the `inp_percent_min` and `inp_percent_max` parameters. Other possible value is `outside`.
- `tc_ignore_te` set to `no` and the `ignore_io_if_fp` parameter is set to `yes`, the `set_false_path` constraints are taken into consideration, if specified.
In GuideWare2.0, the default value of `tc_ignore_te` is `no` and the default value of `ignore_io_if_fp` is `yes`.
- `inp_percent_min`: Default is 20. This indicates that the `Inp_Del07a` rule checks input delay relative to 20% of the clock period value as the minimum limit. Set the value to a positive float value between 0 and 100 to change this percentage.
- `inp_percent_max`: Default is 80. This indicates that the `Inp_Del07a` rule checks input delay relative to 80% of the clock period value as the maximum limit. Set the value to a positive float value between 0 and 100 to change this percentage.

Constraint(s)

SDC

- `set_input_delay` (Mandatory): Use to define the arrival time relative to a clock.
- `set_output_delay` (Optional): Use to define the output path delay value.
- `set_multicycle_path` (Optional): Use to define a multi-cycle path.
- `set_false_path` (Optional): Use to identify paths in a design that are to be marked as false, so that they are not considered during timing analysis.
Refer to the [Parameter\(s\)](#) section for details on how to consider

set_false_path constraints.

Messages and Suggested Fix

Message 1

The following message appears when the *set_input_delay* value `<value1>` specified for the input port `<port-name>` is outside the range `<inp_percent_min-value>` and `<inp_percent_max-value>` of the corresponding clock period `<value2>` with the *io07a_range* parameter not set or set to within:

```
[WARNING] Input delay <value1> for port "<port-name>" is not
within a band of <inp_percent_min-value> to
<inp_percent_max-value> of clock period <value2>
```

Potential Issues

The delay value should be reset to be within the budgeted limits. Otherwise, the rest of the timing path from the port to the other timing end points has less time.

Consequences of Not Fixing

The specified delay value seems too low or too high. This could result in difficulty in meeting timing either for this block, the next block, or the top level.

How to Debug and Fix

The constraint value is not within the band of range specified in the violation message. The SDC file highlights the constraint. Update the value to within the range band.

Message 2

The following message appears when a *set_input_delay* value `<value1>` specified for input port `<port-name>` is within the range `<inp_percent_min-value>` and `<inp_percent_max-value>` of the corresponding clock period `<value2>` with the *io7a_range* parameter set to outside:

```
[WARNING] Input delay <value1> for port "<port-name>" is not
outside a band of <inp_percent_min-value> to
<inp_percent_max-value> of clock period <value2>
```

Potential Issues

The constraint value is inside the range, but it is outside the range.

Consequences of Not Fixing

The specified delay value seems too low or too high. This could result in difficulty in meeting timing either for this block, the next block, or the top level.

How to Debug and Fix

Constraint is specified with all options and the value is not outside the band of range specified in the violation message. The SDC file highlights the constraint that should be set outside the range.

Message 3

The following message appears when a *set_input_delay* value `<value1>` specified with options `<option-list>` for input port `<port-name>` is outside the range `<inp_percent_min-value>` and `<inp_percent_max-value>` of the corresponding clock period `<value2>` with the *io07a_range* parameter not set or set to within:

```
[WARNING] Input delay <value1> specified with <option-list>
options) for port "<port-name>" is not within a band of
<inp_percent_min-value> to <inp_percent_max-value> of
clock period <value2>
```

Potential Issues

The delay value should be reset to within the budgeted limits. Otherwise, the rest of the timing path from the port to other timing end points has less time

Consequences of Not Fixing

The specified delay value seems too low or too high. This could result in difficulty in meeting timing either for this block, the next block, or the top level.

How to Debug and Fix

Constraint is specified with min, max, rise, and fall options and the value is not within the band of range specified in the violation message. The SDC file highlights the constraint that should be set to outside the range.

Message 4

The following message appears when a *set_input_delay* value `<value1>` specified with options `<option-list>` for input port `<port-name>` is within the range `<inp_percent_min-value>` and `<inp_percent_max-value>` of the corresponding clock period `<value2>` with the *io07a_range* parameter set to outside:

[WARNING] Input delay `<value1>` specified with `<option-list>` options) for port "`<port-name>`" is not outside a band of `<inp_percent_min-value>` to `<inp_percent_max-value>` of clock period `<value2>`

Potential Issues

The constraint value is inside the range, but should be outside the range.

Consequences of Not Fixing

The specified delay value seems too low or too high. This could result in difficulty in meeting timing either for this block, the next block, or the top level.

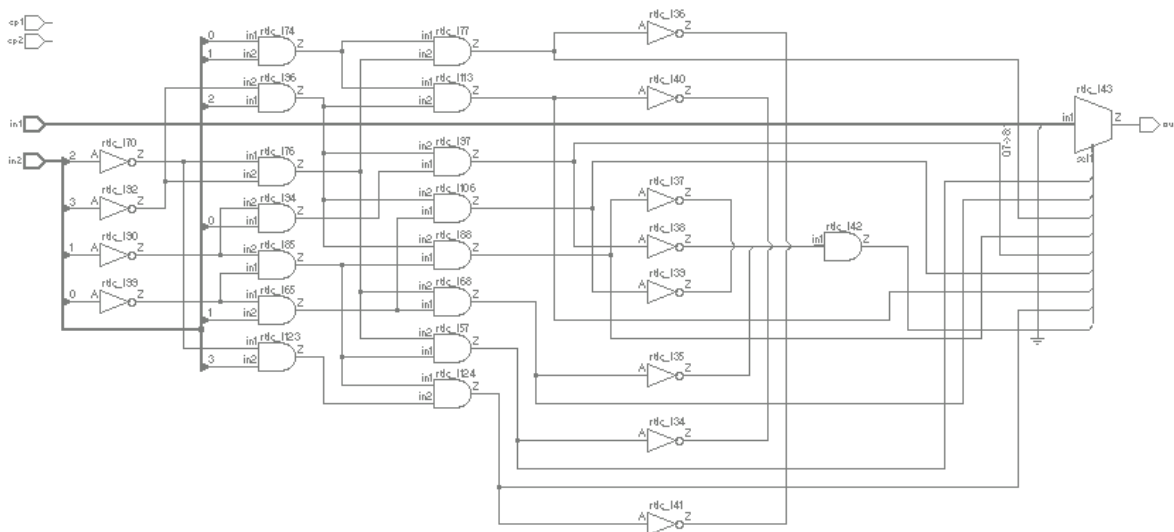
How to Debug and Fix

Constraint is specified with min, max, rise, and fall options and the value is not outside the band of range specified in the violation message. The SDC file highlights the constraint that should be set outside the range.

Example Code and/or Schematic

Example 1[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top.sdc is as follows:

```

top.sdc | top.sdc (2)
create_clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } {cp1}
set_input_delay 9.000000 -clock clk1 [get_ports in1*]

```

The violation message appears because `set_input_delay` specified is not within the specified band. In the top.sdc file, the corresponding constraint is highlighted.

Example 2

Test Case Files Not Available

In this example, suppose the `inp_percent_min` and `inp_percent_max` parameters are set to 20 and 80, respectively. Since there exists a multi-

cycle path for the path from the port `in` to `f1/D`, the clock period is computed as:

```
multiplier value * original clock period = 20
```

Therefore, the valid range is $(20\% * 20)$ to $(80\% * 20) = 4$ to 16 .

```
--Design begin-
```

```
Input in, clk;
```

```
Output out;
```

```
Flop f1( in, clk, out );
```

```
--Design end-
```

```
--SDC begin-
```

```
create_clock -period 10 clk
```

```
set_input_delay $in1_id -clock clk in
```

```
set_multicycle_path 2 -from in -to [get_clocks clk]
```

```
--SDC end-
```

Suppose the *io07a_range* parameter is set to *outside*, the *Inp_Del07a* rule reports a violation when `$in1_id` is not outside the range.

Otherwise, this rule reports a violation when `$in1_id` is not inside the range.

Default Severity Label

Warning

Rule Group

Inp_Del

Reports and Related Files

None

Inp_Del08

Reports `set_input_delay` is set on the same input relative to multiple clocks, but `-add_delay` is missing

When to Use

To validate that no `set_input_delay` constraints is overwritten with another. This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The `Inp_Del08` rule reports input ports with `set_input_delay` constraints set for two different clocks without the `-add_delay` argument. This rule marks the first `set_input_delay` specification and checks whether the second and subsequent specifications have the `-add_delay` argument specified. If the first specification itself contains two clocks, specify the `-add_delay` argument.

Rule Exceptions

The `Inp_Del08` rule ignores the input ports on which a `create_clock` or a `create_generated_clock` constraint is applied.

Parameter(s)

None

Constraint(s)

SDC

- `set_input_delay` (Mandatory): Use to define the arrival time relative to a clock.

Messages and Suggested Fix

The following message appears when the input port `<port-name>` has `set_input_delay` constraints set with different clocks and without the `-add_delay` argument in the second and subsequent specifications:

[WARNING] Second and subsequent cases of `input_delay` on same port '`<port-name>`' should use '`-add_delay`'

Potential Issues

An input delay constraint without the `-add_delay` option is provided. It overwrites the previous delay constraints on the port.

Consequences of Not Fixing

If the `-add_delay` argument is missing, the constraint will get overwritten and can result in incorrect timing.

How to Debug and Fix

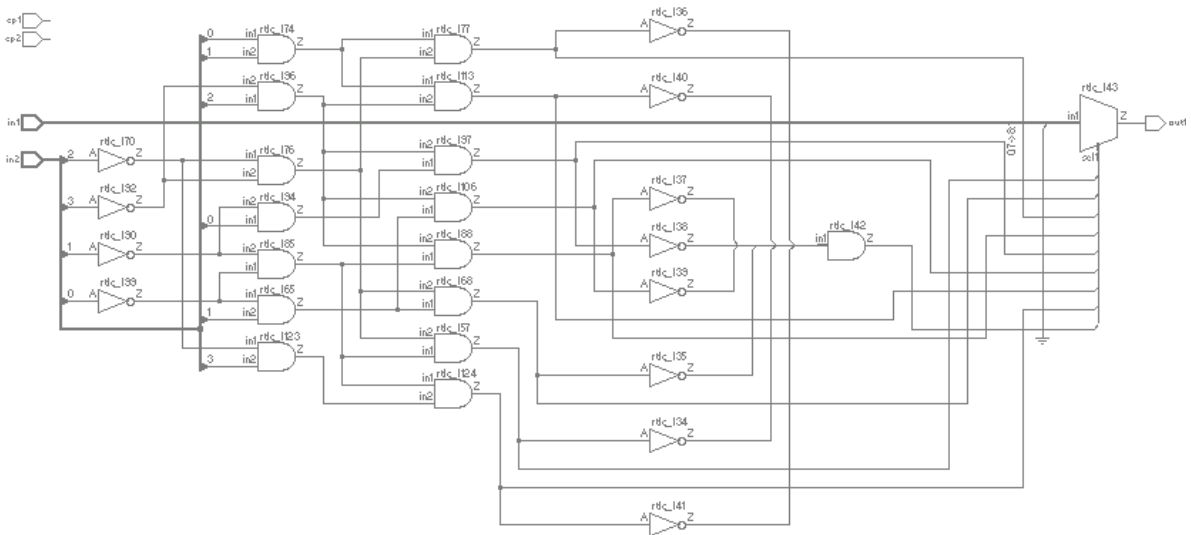
Apply the `add_delay` option to the remaining [set_input_delay](#) constraints as highlighted in the SDC file.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top .sdc is as follows:

```

top.sdc | top.sdc (2)
create_clock -name clk1 -period 10.000000 -waveform {0.000000 5.000000} [get_ports cp1]
create_clock -name clk2 -period 10.000000 -waveform {0.000000 5.000000} [get_ports cp2]
set_input_delay 3.000000 -clock {clk1} [get_ports in1*]
set_input_delay 3.000000 -clock {clk2} [get_ports in1*]
set_input_delay 3.000000 -clock clk1 [get_ports in2*]

```

The violation message appears because the `-add_delay` argument is missing. In the `top.sdc` file, the corresponding constraint is highlighted.

Example 2

Test Case Files Not Available

The `-add_delay` argument allows you to capture information about multiple paths leading to an input port that are relative to different clocks or clock edges. In the following example, the `Inp_Del08` rule reports a violation for the `set_input_delay` constraint provided for `in` relative to `clk2` because `-add_delay` is missing and there exists an earlier `set_input_delay` constraint specified for `in`.

```

--Design begin-
Input in, clk1, clk2;
Output out;
Flop f1( in, clk1, out );
Flop f2( in, clk2, out );
--Design end-

--SDC begin-
create_clock -period 10 clk1
create_clock -period 12 clk2
set_input_delay 1.0 -clock clk1 in
set_input_delay 1.0 -clock clk2 in

```

I/O Rules

--SDC end--

Default Severity Label

Warning

Rule Group

Inp_Del

Reports and Related Files

None

Inp_Del09

Reports pins on a bus that do not have the same input delay

When to Use

To verify same delay values on all pins of a bus. This rule is applicable to the RTL, Pre-layout, Post-layout phases.

Description

The *Inp_Del09* rule reports bus input or inout ports where not all bits of a bus have the same *set_input_delay* constraint delay values.

If the bits of a bus have input delays specified with respect to a clock, this rule checks whether all input delays are within the acceptable margin, as specified by the *tc_inter_block_delay* parameter, of the mean bus delay and reports a violation if the delays fall outside it. This means the bus delay is defined as the sum of the delays defined on each pin of the bus divided by the number of the pins.

If the *tc_inter_block_delay* parameter is not set, the *Inp_Del09* rule checks all the bits of a bus that have the same input delay.

This rule checks the bits of a bus separately for each of the maximum rise, maximum fall, minimum rise, and minimum fall input delays.

Prerequisites

It is recommended that the *Inp_Del09* rule be run after the *Inp_Del02*, *Inp_Del03*, *Inp_Del04*, *Inp_Del07*, and *Inp_Del08* rules.

Parameter(s)

- *tc_inter_block_delay*: Default is 0. Set the value to a float less than 1 to specify the interconnect delay as a factor of the clock period.

Constraint(s)

SDC

- *set_input_delay* (Mandatory): Use to define the arrival time relative to a clock.

Messages and Suggested Fix

Message 1

The following message appears when the input delay for the delay-type *<delay-type-name>*, of the input/inout port bit *<port-name>*, is not in the margin range *<margin-value>* percent of the mean bus delay *<delay-value>*.

[WARNING] Input delay constraint(s) *<delay-type-name>*, for port *<port-name>*, is not in the *<margin-value>*% range of mean bus delay *<delay-value>*

Where:

- *<delay-type-name>* can be (-max, -rise), (-max, -fall), (-min, -rise), or (-min, -fall)
- *<margin-value>* is the value of *tc_io_delay_bus_margin* rule parameter
- *<port-name>* is the name of the violating bit of the input/inout bus
- *<delay-value>* is the value of the mean bus delay

For debugging information, click [How to Debug and Fix](#).

Message 2

The following message appears when the input delay for the delay-type *<delay-type-name>*, of the input/inout port bit *<port-name>*, relative to a clock *<clk-name>*, is not in the margin range *<margin-value>* percent of the mean bus delay *<delay-value>*:

[WARNING] Input delay constraint(s) *<delay-type-name>*, for port *<port-name>*, is not in the *<margin-value>*% range of mean bus delay *<delay-value>*, relative to clock *<clk-name>*

Potential Issues

Indicative of possible typos. However, some variation is possible on account of different routing and/or if different bits are driven by different registers. For example, they may have been clocked differently.

Consequences of Not Fixing

The delay values should be near the mean delay value. Any diverging value for a bus bit might negatively impact the timing analysis for the

corresponding path.

How to Debug and Fix

Mean bus delay is the average of the delay values on all the bits. The delay value for each bit should be less than the margin value, which is computed with the following equation:

$$(\text{mean bus delay} * tc_io_delay_bus_margin) / 100$$

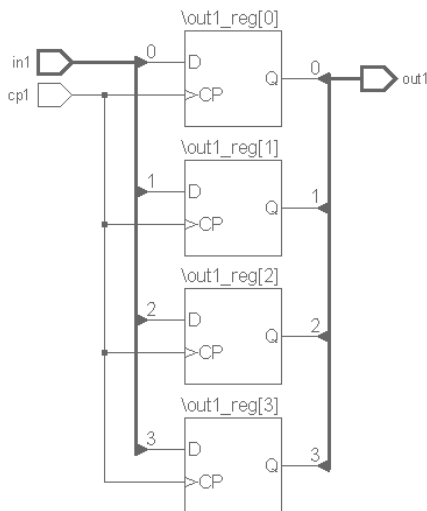
The constraint that has a delay value greater than the margin value is highlighted in the SDC file. Update it to within the margin.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top .sdc is as follows:


```
top.sdc | top.sdc (1) |
set_input_delay 5 -max -rise in1[0]
set_input_delay 4 -in1[1]
set_input_delay 2 -min -rise in1[2]
set_input_delay 3 -min -rise in1[3]
```

The violation message appears because pins on a bus do not have the same input delay. In the top.sdc file, the corresponding constraint is highlighted.

Example 2

 Test Case Files Not Available

The mean bus delay is the average delay for the bus. In this example, the `in` bus consists of two bits and the mean bus delay computes to 7.5. Suppose the `tc_io_delay_bus_margin` parameter is set to 10. The maximum permissible deviation of each bit's delay value from the mean bus delay value is:

$$(\text{mean bus delay} * \text{tc_io_delay_bus_margin}) / 100$$

Therefore, the maximum permissible deviation is 0.75. Since the delay values for both bits deviate more than 0.75, the `Inp_De109` rule reports a violation for both of the bits.

```
--Design begin-
Input [0:1] in;
Input clk;
Output out;
Flop f1( in[0], clk, out );
Flop f2( in[1], clk, out );
--Design end-
--SDC begin-
create_clock -period 10 clk
```

```
set_input_delay 10 -clock clk in[0]
set_input_delay 5 -clock clk in[1]
--SDC end-
```

Default Severity Label

Warning

Rule Group

Inp_Del

Reports and Related Files

None

Inp_Del10

Reports when `level_sensitive` is used in `set_input_delay`

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Inp_Del10* rule reports `set_input_delay` constraints specified with the `level_sensitive` argument.

Level-sensitive input port delays are not supported by some implementation tools.

Parameter(s)

None

Constraint(s)

- `set_input_delay` (Mandatory): Use this constraint to specify the input delay of the port/pin with respect to a clock.

Messages and Suggested Fix

Message

The following message appears when a `set_input_delay` constraint is specified with the `-level_sensitive` argument in the SDC file:

```
[INFO] Option "level_sensitive" used in command  
"set_input_delay" is not supported by all implementation tools
```

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

If you do not fix this violation, not all tools support all the options. As a result, same constraints files are interpreted differently by various tools in the flow.

How to Debug and Fix

The SDC line impacted by this violation message is highlighted in the

Console GUI.

Update the SDC file by removing the `-level_sensitive` argument from the `set_input_delay` constraint.

Example Code and/or Schematic

The `Inp_Del10` rule reports violation messages for the following snippet.

```
module flop (inport, clk, rst, output);
```

```
    input inport;
    input clk;
    input rst;
    output output;
    reg output;
```

```
    always @(posedge clk or posedge rst )
```

```
    begin : FD
```

```
        if (rst == 1'b1)
```

```
            begin
```

```
                output <= 1'b0 ;
```

```
            end
```

```
        else
```

```
            begin
```

```
                output <= inport ;
```

```
            end
```

```
        end
```

```
    endmodule
```

The code snippet for the SDC file is as follows:

```
set_input_delay 2.0 -clock cclk {inport} -level_sensitive
set_input_delay 2.0 -clock cclk {inport}
set_input_delay -rise 2.0 -clock cclk {inport} -
level_sensitive
set_input_delay -2.0 -clock cclk {inport} -level_sensitive
```

Default Severity Label

Info

Rule Group

Inp_Del

Reports and Related Files

None

Inp_Del11

Reports usage of `clock_fall` in `set_input_delay`

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Inp_Del11* rule reports `set_input_delay` commands specified with `-clock_fall` option. Such edge-sensitive input port delays are not supported by all implementation tools.

Parameter(s)

None

Constraint(s)

- `set_input_delay` (Mandatory): Use this constraint to specify the input delay of the port/pin with respect to a clock.

Messages and Suggested Fix

Message

The following message appears when a `set_input_delay` command is specified with the `-clock_fall` option in the SDC file:

```
[INFO] Option "clock_fall" used in command "set_input_delay" is not supported by all implementation tools
```

Potential Issues

The violation message appears when `-clock_fall` option is used in `set_input_delay` command.

Consequences of Not Fixing

If you do not fix, same constraints files are interpreted differently by various tools in the flow because not all tools support some of these options.

How to Debug and Fix

The SDC line impacted by this violation message is highlighted in the

Console GUI.

Update the SDC file by removing the `-clock_fall` option from the `set_input_delay` command.

Example Code and/or Schematic

The `Inp_De/11` rule reports violation messages for the following snippet.

```
module flop (inport, clk, rst, output);
```

```
    input inport;
    input clk;
    input rst;
    output output;
    reg output;

    always @(posedge clk or posedge rst )
    begin : FD
        if (rst == 1'b1)
            begin
                output <= 1'b0 ;
            end
        else
            begin
                output <= inport ;
            end
        end
    end

endmodule
```

The code snippet for the SDC file is as follows:

```
set_input_delay 2.0 -clock cclk {inport} -level_sensitive -
clock_fall
```

```
set_input_delay 2.0 -clock cclk {inport} -clock_fall
set_input_delay -rise 2.0 -clock cclk {inport} -
level_sensitive
set_input_delay 0 -clock cclk {inport} -level_sensitive -
clock_fall
set_input_delay -2.0 -clock cclk {inport} -level_sensitive -
clock_fall
```

Default Severity Label

Info

Rule Group

Inp_Del

Reports and Related Files

None

Inp_Del12

Reports use of unusual options like `network_latency_included` and `source_latency_included` in `set_input_delay` commands

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The `Inp_Del12` rule reports `set_input_delay` commands specified with `network_latency_included` and/or the `source_latency_included` options.

Parameter(s)

None

Constraint(s)

- `set_input_delay` (Mandatory): Use this constraint to specify the input delay of the port/pin with respect to a clock.

Messages and Suggested Fix

The following message appears when `set_input_delay` command is specified with an unsupported option, where the `<option-list>` has one or both of `network_latency_included` and `source_latency_included` in the SDC file:

```
[INFO] Un-Supported (by Implementation) option(s)
"<optionlist>" used in command "set_input_delay" are not
supported by all implementation tools
```

Potential Issues

The violation message appears when the `set_input_delay` command is specified with unsupported options.

Consequences of Not Fixing

If you do not fix, same constraints files are interpreted differently by various tools in the flow because not all tools support some of these options.

How to Debug and Fix

The SDC line impacted by this violation message is highlighted in the Console GUI.

Update the SDC file by removing the `-network_latency_included` and `-source_latency_included` options from the [set_input_delay](#) command.

Example Code and/or Schematic

Consider the following snippet for the design.

```
module flop (in, clkt, out);
    input in, clk;
    output out;
    reg out;
    always @(posedge clk)
out <= in;
endmodule
```

The snippet for the SDC file is as follows:

```
create_clock clk -name cclk -period 10 -waveform {0 5}
set_input_delay 2.0 -clock cclk {inport} -clock_fall -
network_latency_included
set_input_delay 2.0 -clock cclk {inport} -clock_fall -
source_latency_included
```

You will get the violation message for the two [set_input_delay](#) constraints.

Default Severity Label

Info

Rule Group

Inp_Del

I/O Rules

Reports and Related Files

None

Inp_Del13

Reports if negative value is set on `set_input_delay`

When to Use

This rule is applicable to the RTL, pre-layout, and post-layout phases.

Description

The *Inp_Del13* rule reports `set_input_delay` constraints with negative values set.

Parameter(s)

None

Constraint(s)

- `set_input_delay` (Mandatory): Use this constraint to specify the input delay of the port/pin with respect to a clock.

Messages and Suggested Fix

The following message appears for a `set_input_delay` constraint specified with a negative value:

```
[INFO] set_input_delay is set to negative value
```

Potential Issues

The violation message appears when `set_input_delay` constraint is specified with a negative value.

Consequences of Not Fixing

If you do not fix, most timing methodologies do not handle negative delays. Ensure to use correct value.

How to Debug and Fix

The SDC line impacted by this violation message is highlighted in the Console GUI.

Update the SDC file by setting the negative value to `set_input_delay` constraint.

Example Code and/or Schematic

The *Inp_De13* rule reports a violation for the following snippet.

```
set_input_delay -2.0 -clock cclk {inport}
```

Default Severity Label

Info

Rule Group

Inp_De1

Reports and Related Files

None

Inp_Del14

Reports `set_input_delay` specified with wrong/incomplete set of clocks

When to Use

This rule is applicable to the RTL, Pre-layout, and Postlayout phases.

Description

The *Inp_Del14* rule reports [set_input_delay](#) constraints that have been specified with wrong or incomplete set of clocks.

The *Inp_Del14* rule requires that the [set_input_delay](#) constraint be applied on a port with respect to the clocks generating the data. The generating clock is identified by propagating the top-level clocks through the blocks. If the timing-characteristics of the clock specified in the [set_input_delay](#) constraint matches the timing characteristics of the generating clock, this rule assumes that the [set_input_delay](#) constraint specified is correct. The delay value specified is not considered by the *Inp_Del14* rule.

Rule Exceptions

The *Inp_Del14* rule does not run for top-level.

While propagating the clocks defined at the top level, the *Inp_Del14* rule does not consider the [set_false_path](#) constraints. The support for these constraints will be added in a later release.

Parameter(s)

None

Constraint(s)

- [set_input_delay](#) (Mandatory): Use this constraint to specify the input delay of the port/pin with respect to a clock.
- [set_false_path](#) (Optional): Use to identify paths in a design that are to be marked as false, so that they are not considered during timing analysis.

Messages and Suggested Fix

Message 1

The following message appears when a [set_input_delay](#) constraint for a port

`<port-name>` has been specified with clock `<clk-name>`, period `<value 1>`, and waveform `<value 2>` but no clock is generating the port:

[WARNING] `set_input_delay` for port `<port-name>` is specified with clock { `<clk-name>` [period: `<value 1>`, waveform { `<value 2>` }] `<-clock_fall>` } in block {`<block-name>`}, but no clock is generating this port

For debugging information, click [How to Debug and Fix](#).

Message 2

The following message appears when a `set_input_delay` constraint for a port `<port-name>` has been specified with clock `<clk-name>`, period `<value 1>`, and waveform `<value 2>` but a `set_input_delay` constraint has already been specified with all the generating clocks:

[WARNING] `set_input_delay` for port `<port-name>` is specified with clock { `<clk-name>` [period: `<value 1>`, waveform { `<value 2>` }] `<-clock_fall>` } in block {`<block-name>`}, but `set_input_delay` is already specified with all the generating clocks

For debugging information, click [How to Debug and Fix](#).

Message 3

The following message appears when a `set_input_delay` constraint for a port `<port-name>` has not been specified with clock `<clk-name>`, period `<value 1>`, and waveform `<value 2>` for block `<block-name>`:

[WARNING] `set_input_delay` for port `<port-name>` should be specified with clock { `<clk-name>` [period: `<value 1>`, waveform { `<value 2>` }] `<-clock_fall>` }, but no `set_input_delay` is specified in block { `<block-name>` }

For debugging information, click [How to Debug and Fix](#).

Message 4

The following message appears when a `set_input_delay` constraint for a port `<port-name>` has been specified with clock `<clk-name>` for block `<block-name>` but the information is not complete:

[WARNING] `set_input_delay` for port `<port-name>` should be specified with clock { `<clk-name>` [period: `<value 1>`, waveform

```
{ <value 2> }] <-clock_fall> }, in block { <block-name> }
```

For debugging information, click [How to Debug and Fix](#).

Message 5

The following message appears when the timing-characteristics between the clocks specified in the [set_input_delay](#) constraint and the clocks generating the data do not match:

```
[WARNING] set_input_delay for port <port-name> is specified with clock { <clk-name> [period: <value 1>, waveform { <value 2> }] <-clock_fall> }, in block { <block-name> } but it should be specified with clock { <clk2-name> [period: <value 3>, waveform { <value 4> }] <-clock_fall> }
```

Potential Issues

The *Inp_Del14* rule reports a violation under the following conditions:

- No [set_input_delay](#) is specified for a port, but some clocks are generating the data.
- [set_input_delay](#) is specified for a port, but no clock is generating the data.
- There is a mismatch in timing-characteristics between the clocks specified in [set_input_delay](#) and the clocks generating the data.

Consequences of Not Fixing

Not fixing these violations can result in difficulty in meeting timing. Even if timing is met, due to false path or multi-cycle path specifications, the actual design might not be able to meet the timing.

How to Debug and Fix

The *Inp_Del14* rule highlights the following:

- The path from the port to the flip-flop generating the port
- The path from the flip-flop to the port/pin of the clock

To fix these violation, perform the following:

- **Message 1 and 2:** Remove the [set_input_delay](#) constraint.
- **Messages 3 and 4:** The port is being generated by the clock for which the [set_input_delay](#) constraint is not specified, which can be verified using fan-in traversal (in the Console GUI) from the input port. To fix, specify the [set_input_delay](#) constraint with respect to all the reported clocks.

- **Message 5:** Check the timing characteristics of both the clocks and update either one.

Example Code and/or Schematic

For the following design and SDC files, the *Inp_De/14* rule reports violation messages.

```
module top(in, clk1, clk2, out);
input in, clk1, clk2;
output out;

block1 B1(in, clk1, w);
block2 B2(w, clk2, out);

endmodule

module block1(in, clk, out);
input in, clk;
output out;

FD1 F1(.D(in), .CP(clk), .Q(out));

endmodule

module block2(in, clk, out);
input in, clk;
output out;

FD1 F2(.D(in), .CP(clk), .Q(out));
```

```
endmodule
```

The constraints are defined in the following SDC files:

B1.sdc

```
create_clock -name BC1 -period 10 clk
```

```
set_input_delay 1 -clock BC1 in  
set_output_delay 1 -clock BC1 out
```

B2.sdc

```
create_clock -name BC2 -period 15 clk
```

```
set_input_delay 1 -clock BC2 in  
set_output_delay 1 -clock BC2 out
```

Default Severity Label

Warning

Rule Group

Inp_De1

Reports and Related Files

No report and related file

Inp_Del15

Reports internal pins on which `set_input_delay` is specified

When to Use

This rule is applicable to the RTL, Pre-layout, and Postlayout phases.

Description

The *Inp_Del15* rule reports a violation when `set_input_delay` constraints are specified on internal pins. The `set_input_delay` constraint should be specified on a port.

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

The following message appears:

[WARNING] Input delay specified on internal pin(s) <pin-name>

Potential Issues

Not applicable

Consequences of Not Fixing

Not applicable

How to Debug and Fix

The *Inp_Del15* rule highlights the `set_input_delay` constraints that are specified on the internal pin (terminal).

To fix these violation, remove the `set_input_delay` constraint.

Example Code and/or Schematic

For the following design and SDC files, the *Inp_Del14* rule reports violation messages.

The design file is as follows:

```
module top(input in1,in2,clk1,output out1,out2);
AN2 a1(.A(in1),.B(in2),.Z(n1));
FD1 f1(.D(n1),.CP(clk1),.Q(out1));
block1 b1(.in1(n1),.clk1(clk1),.out1(out2));
endmodule
```

```
module block1(input in1,clk1,output out1);
AN2 a1(.A(in1),.B(),.Z(n1));
FD1 f1(.D(n1),.CP(clk1),.Q(out1));
endmodule
```

The constraints are defined in the following SDC files:

top.sdc

```
create_clock -name C1 -period 10 [get_ports clk1]

set_input_delay 1 -clock C1 [get_pins f1/D]
set_input_delay 1 [get_pins a1/A]
set_input_delay 1 [get_pins b1/a1/A]
set_input_delay 1 -clock C1 [get_pins b1/f1/D]
set_input_delay 3 [get_ports in1]
```

block1.sdc

```
create_clock -name C1 -period 10 [get_ports clk1]
set_input_delay 1 -clock C1 [get_pins f1/D]
```

```
set_input_delay 1 [get_pins a1/A]
```

After you run this rule, the following violations are reported:

test_goal (6)		
Inp_Del15 (6) : set_input_delay specified on internal pins		
⚠	Input delay specified on internal pin(s) "f1/D"	block1.sdc
⚠	Input delay specified on internal pin(s) "a1/A"	block1.sdc
⚠	Input delay specified on internal pin(s) "f1/D"	top.sdc
⚠	Input delay specified on internal pin(s) "a1/A"	top.sdc
⚠	Input delay specified on internal pin(s) "b1/a1/A"	top.sdc
⚠	Input delay specified on internal pin(s) "b1/f1/D"	top.sdc

Default Severity Label

Warning

Rule Group

Inp_De1

Reports and Related Files

No report and related file

Input Transition Rules

The Input Transition Rules sub-group `Inp_Trans` contains the following rules:

Rule	Description
<i>Inp_Trans01</i>	Input ports without transition delay constraints
<i>Inp_Trans01a</i>	Input ports without transition delay constraints
<i>Inp_Trans02</i>	Input ports with incorrect Transition Delay constraints
<i>Inp_Trans03</i>	Incomplete <i>set_input_transition</i> constraints
<i>Inp_Trans03a</i>	Incomplete <i>set_driving_cell</i> constraints
<i>Inp_Trans04</i>	Input ports with transition value limits outside the technology limits
<i>Inp_Trans05</i>	Inconsistent input transition values between input ports and associated clocks in the RTL and pre-layout analysis phases
<i>Inp_Trans06</i>	Inconsistent input transition values between input ports and associated clocks in the port-layout analysis phase
<i>Inp_Trans07</i>	Input ports where a <i>set_driving_cell</i> constraint is set but no <i>set_load</i> constraint
<i>Inp_Trans08</i>	<i>set_input_transition</i> constraints and <i>set_driving_cell</i> constraints specified with the arguments that are unsupported (for synthesis) arguments
<i>Inp_Trans09</i>	<i>set_drive</i> constraints used

Inp_Trans01

Reports when input transition or drive or driving cell is not defined for input/inout

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *Inp_Trans01* rule reports *input/inout* ports without recommended transition constraints set.

By default, the *Inp_Trans01* rule reports the *input/inout* ports where:

- None of the input transition constraints - *set_input_transition*, *set_drive* and *set_load*, or *set_driving_cell* constraint are specified.
- Only *set_drive* and *set_load* constraints are defined.
- Only *set_input_transition* constraint is defined.

For the above-mentioned last two cases, it is recommended to use the *set_driving_cell* constraint instead, as these constraints associate a library pin with an *input/inout* port so that the drive capability is accurately modeled using the library data.

If the *chip* parameter is set, use *set_input_transition* instead of *set_driving_cell*.

Rule Exception(s)

The *Inp_Trans01* rule ignores the following:

- All hanging, power or ground *input/inout* ports.
- *Input/inout* ports with *set_case_analysis/set_disable_timing* constraints.
- *Input/inout* ports driven by a sequential cell if a constant value is propagated to all clock-pins of that sequential cell.
- Clock ports.

Parameter(s)

- *tc_ignore_te*: Default value is *no*. If *tc_ignore_te* is set as *no* and *ignore_io_if_fp* parameter is set to *yes*, the *set_false_path* constraints are taken into consideration, if specified.

In GuideWare2.0, the default value of *tc_ignore_te* is no and the default value of *ignore_io_if_fp* is yes.

- *allow_input_transition*: Default value is yes. Set the value to no to ensure that this parameter reports use of *set_driving_cell* constraints.
- *allow_drive*: Default value is no. When on default, reports use of *set_driving_cell* constraints along with *set_load* constraint.
- *allow_driving_cell*: Default value is yes. When on default, reports use of *set_driving_cell* constraints.

Constraint(s)

- *set_driving_cell* (Optional): Use this constraint to set the port driving cell.
- *set_load* (Optional): Use this constraint to set the capacitance on ports and nets in the current design.

Messages and Suggested Fix

Message 1

The following message appears when an input/inout port *<port-name>* does not have all input transition constraints specified:

```
[WARNING] <constr> is not defined for Input/inout port '<port-name>'
```

Where *<constr>* can be *set_input_transition*, *set_drive*, *set_driving_cell*, or a combination of both.

Potential Issues

The violation message appears when an input/inout port *<port-name>* does not have all input transition constraints specified.

Consequences of Not Fixing

If you do not fix this violation, it affects results of delay calculations.

How to Debug and Fix

Specify *set_input_transition*, *set_drive* or *set_driving_cell* for the input/inout port.

Message 2

The following message appears when an input/inout port `<port-name>` has `set_drive` and `set_load` constraints defined with `allow_drive` parameter not set:

[WARNING] Using `set_drive` and `set_load` for input/inout port "`<port-name>`" is not the recommended way of specifying transition. Use `<constrlist>` instead

Potential Issues

The violation message appears when `allow_drive` parameter is not set on an input/inout port that is specified with `set_drive` and `set_load` constraints.

Consequences of Not Fixing

If you do not fix this violation, not defining input transition for input ports affects results of delay calculations.

How to Debug and Fix

Specify either of `set_input_transition`, `set_drive` or `set_driving_cell` for the input/inout port.

Message 3

The following message appears when an input/inout port `<port-name>` has `set_input_transition` constraint defined with `allow_input_transition` rule parameter not set:

[WARNING] Using `set_input_transition` for input/inout port "`<port-name>`" is not the recommended way of specifying transition. Use `<constr-list>` instead

Where, `<constr-list>` is the list of recommended constraints based on the combination of `allow_drive` and `allow_driving_cell` parameters specified.

Potential Issues

The violation message appears when `allow_input_transition` rule parameter is not set on an input/inout port that is specified with `set_input_transition` constraint.

Consequences of Not Fixing

If you do not fix this violation, not defining input transition for input ports affects results of delay calculations.

How to Debug and Fix

Specify *<constr-list>* which is a list of recommended constraints based on the combination of *allow_drive* and *allow_driving_cell* parameters.

Message 4

The following message appears when an input/inout port *<port-name>* has *set_driving_cell* constraint defined with *allow_driving_cell* rule parameter not set:

[WARNING] Using *set_driving_cell* for input/inout port "*<port-name>*" is not the recommended way of specifying transition. Use *<constr-list>* instead

Where, *<constr-list>* is the list of recommended constraints based on the combination of *allow_drive* and *allow_input_transition* parameters specified.

Potential Issues

The violation message appears when the *allow_driving_cell* parameter is not set on an input/inout port that is specified with the *set_driving_cell* constraint.

Consequences of Not Fixing

If you do not fix this violation, not defining input transition for input ports affects results of delay calculations.

How to Debug and Fix

Specify a *<constr-list>* which is the list of recommended constraints based on the combination of the *allow_drive* and *allow_input_transition* parameters.

Message 5

The following message appears when none of the *allow_input_transition*, *allow_driving_cell*, and *allow_drive* parameters are set:

[WARNING] All parameters *allow_input_transition*, *allow_drive* and *allow_driving_cell* are disabled and therefore rule will not be checked

Potential Issues

The violation message appears when none of the *allow_input_transition*, *allow_driving_cell*, and *allow_drive* parameters are set.

Consequences of Not Fixing

If you do not fix this violation, not defining input transition for input ports affects results of delay calculations.

How to Debug and Fix

Ensure that all parameters [allow_input_transition](#), [allow_driving_cell](#), and [allow_drive](#) are not disabled.

Example Code and/or Schematic

The `Inp_Trans01` rule reports a violation for the following snippet.

```
//test.v
module top (CLK, data, rst, out, outbar);

input CLK, rst;
input data;
output out;

reg q;

always @(posedge CLK or posedge rst)
begin
    if (rst)
        q <= 1'b0;
    else if (CLK)
        q <= data;
end

assign out = q;
endmodule
```

In the above design file, suppose no [set_driving_cell](#) is defined for input port

"rst". This rule will then report a violation.

Default Severity Label

Warning

Rule Group

Input Transition Rules

Reports and Related Files

None

Inp_Trans01a

Identifies unspecified input transition constraints for inputs/inouts

When to Use

RTL phase, Pre-layout phase, Post-layout phase

Description

The *Inp_Trans01a* rule reports input or inout ports that have neither *set_input_transition*, *set_drive*, nor *set_driving_cell* constraint specified.

Rule Exceptions

The *Inp_Trans01a* rule ignores the following:

1. Hanging, power, or ground input/inout ports
2. Input/inout ports with *set_case_analysis/set_disable_timing* constraints
3. Input/inout ports driven by a sequential cell if a constant value is propagated to all clock pins of that sequential cell
4. Clock ports

The *Inp_Trans01a* rule does not report a violation if the *chip* rule parameter is set.

Parameter(s)

- *chip*: Default is unspecified. Set the value to the name of the design unit which corresponds to the chip level.
- *tc_ignore_te* and *ignore_io_if_fp*: If the *tc_ignore_te* parameter is set to no and the *ignore_io_if_fp* parameter is set to yes, the *Inp_Trans01* rule considers *set_false_path* constraints, if specified.

In GuideWare2.0, the default value of *tc_ignore_te* is no and the default value of *ignore_io_if_fp* is yes.

Constraint(s)

SDC

- *set_drive* (Optional): Use to set the resistance on input or inout ports in the current design.
- *set_driving_cell* (Optional): Use to set the port driving cell.

- *set_input_transition* (Optional): Use to set a fixed transition time on input or inout ports.
- *set_false_path* (Optional): Use to identify paths in a design that are to be marked as false, so that they are not considered during timing analysis. For more information, refer to the *Parameter(s)* section.

Messages and Suggested Fix

The following message appears when the input/inout port *<port-name>* of the design/block *<name>* does not have any input transition constraints specified:

[WARNING] No transition constraint (set_input_transition or set_drive or set_driving_cell) has been specified for input/inout port "<port-name>" of design/block "<name>"

Potential Issues

Not defining the *set_input_transition*, *set_drive*, or *set_driving_cell* constraint can affect results of delay calculation.

Consequences of Not Fixing

Implementation tools could assume ideal transition. This leads to optimistic timing analysis and potential timing failure in post-layout.

How to Debug and Fix

The violation message indicates the input or inout port for which no transition constraint is specified in the SDC file. The port is highlighted in the design file.

To resolve this violation, constrain the port with a suitable transition constraint.

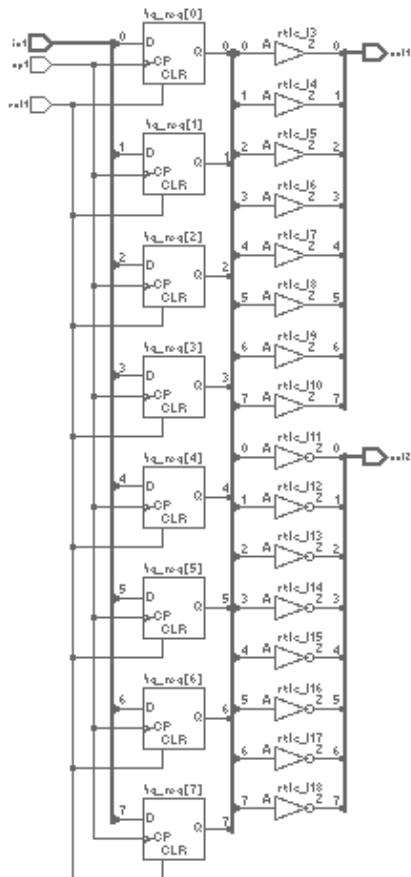
Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:

I/O Rules



The top.sdc is as follows:

```

top.sdc x
1 create_clock -name clk1 -period 10 [get_ports cp1]
2

```

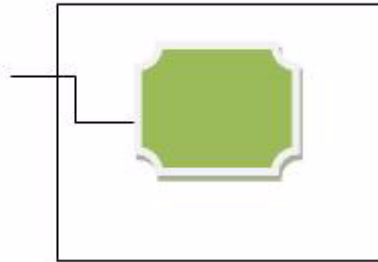
The violation message appears because no transition constraint has been defined for the port `rst1`. In the `top.sdc` file, the corresponding constraint

is shown.

Example 2Test Case Files Not Available

In the following example, the *Inp_Trans01a* rule reports a violation because no transition constraint is set on the port. To resolve the violation, set transition constraints at the input port. The transition constraints can be *set_input_transition*, *set_drive*, nor *set_driving_cell*.

**No transition
constraint
set at this input
port**

**Default Severity Label**

Warning

Rule Group

Inp_Trans

Reports and Related Files

None

Inp_Trans02

Reports input transition values that have a minimum value greater than the maximum value

When to Use

RTL phase, Pre-layout phase, Post-layout phase

Description

The Inp_Trans02 rule reports *set_input_transition* constraints that have a greater `-min` argument value than the corresponding `-max` argument value.

Parameter(s)

None

Constraint(s)

SDC

- *set_input_transition* (Mandatory): Use to set a fixed transition time on input or inout ports.

Messages and Suggested Fix

The following message appears when the minimum transition delay value `<value1>` for the input port `<port-name>` is greater than the corresponding maximum transition delay value `<value2>`:

```
[WARNING] Input transition minimum value <value1> is greater
than input transition maximum value <value2> for Input port
"<port-name>"
```

Potential Issues

The maximum or minimum transition value could be modified internally by implementation tools as per their internal criteria.

Consequences of Not Fixing

Not fixing this violation could lead to mismatch between the specification intention and actual timing calculation.

How to Debug and Fix

The violation message states the port for which minimum value of the input transition constraint is greater than the maximum value. Update the input transition constraints for minimum and maximum value as highlighted in the SDC file.

Example Code and/or Schematic

In the following SDC file snippet, the max transition value is less than the min transition value. Therefore, the *Inp_Trans02* rule reports a violation.

```
set_input_transition 0.75 -max [get_ports {IN2*}]  
set_input_transition 1.4 -min [get_ports {IN2*}]
```

Default Severity Label

Warning

Rule Group

Inp_Trans

Reports and Related Files

None

Inp_Trans03

Reports incomplete `set_input_transition` constraints

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *Inp_Trans03* rule reports incomplete [set_input_transition](#) constraints.

The [set_input_transition](#) constraint is specified with four transition values and these are as follows:

- -min
- -max
- -rise
- -fall

The constraint specifications should be balanced so that no value is inferred if a related value is specified. For example, if one constraint specification has -min -rise values specified, then three more values -min -fall, -max -rise, and -max -fall are expected to be specified on the same object. This is a methodology requirement.

Prerequisite(s)

The *Inp_Trans03* rule requires the following:

- The [set_input_transition](#) constraint should be specified with either none or both -min and -max options.
- The [set_input_transition](#) constraint should be specified with either none or both -rise and -fall options.

Rule Exception(s)

The *Inp_Trans03* rule ignores missing -min option for worst-case SDC when the [tc_ignore_min](#) parameter is set.

Parameter(s)

None

Constraint(s)

- *set_input_transition* (Mandatory): Use this constraint to set a fixed transition time on input or inout ports.

Messages and Suggested Fix

Message

The following message appears for a *set_input_transition* constraint on a port *<port-name>* of design *<name>* that is not specified with all options explicitly:

[WARNING] Not all *set_input_transition* options specified explicitly for port "*<port-name>*" of design "*<name>*" (Specified: *<option-list1>*, Not specified: *<option-list2>*)

Where *<option-list1>* is the list of options specified explicitly and *<option-list2>* is the list of options not specified explicitly.

Potential Issues

The violation message appears when *set_input_transition* constraint is not specified with all the options.

Consequences of Not Fixing

If you do not fix this violation, most tools infer missing values but it is a safer practice to specify each value.

How to Debug and Fix

Ensure to specify each value correctly.

Example Code and/or Schematic

Consider the following snippet.

```
...
set_input_transition 2.0 -min {IN2}
set_input_transition 2.1 -max {IN2}
...
```

The above specifications are not reported as the *set_input_transition* constraints for the same input port IN2 have been specified for both *-min* and *-max* arguments.

However, the following example is reported as all combinations of `-max/-min` and `-rise/-fall` arguments are not specified for input port `IN2`:

```
...  
set_input_transition 2.0 -min -rise {IN2}  
set_input_transition 2.1 -max -fall {IN2}  
...
```

Default Severity Label

Warning

Rule Group

Inp_Trans

Reports and Related Files

None

Inp_Trans03a

Reports incomplete `set_driving_cell` constraints

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *Inp_Trans03a* rule reports incomplete `set_driving_cell` constraints.

The `set_driving_cell` constraint is specified with the following four transition values:

- `-min`
- `-max`
- `-rise`
- `-fall`

The constraint specifications should be balanced so that no value is inferred if a related value is specified. For example, if one constraint specification has the `-min -rise` values specified, then three more values are expected to be specified on the same object with `-min -fall`, `-max -rise`, and `-max -fall` arguments. This is a methodology requirement.

Prerequisite(s)

The *Inp_Trans03a* rule requires the following:

- The `set_driving_cell` constraint should be specified with either none or both `-min` and `-max` options.
- The `set_driving_cell` constraint should be specified with either none or both `-rise` and `-fall` options.

Rule Exception(s)

The *Inp_Trans03a* rule ignores missing `-min` option for worst-case SDC when the `tc_ignore_min` rule parameter is set.

Parameter(s)

None

Constraint(s)

- *set_driving_cell* (Optional): Use this constraint to set the port driving cell.

Messages and Suggested Fix

The following message appears for a *set_driving_cell* constraint on a port `<port-name>` of design `<name>` that is not specified with all options explicitly:

[WARNING] Not all set_driving_cell options specified explicitly for port "<port-name>" of design "<name>" (Specified: <option-list1>, Not specified: <option-list2>)

Where `<option-list1>` is the list of options specified explicitly and `<option-list2>` is the list of options not specified explicitly.

Potential Issues

The violation message appears when all the options of *set_driving_cell* constraint are not specified explicitly.

Consequences of Not Fixing

If you do not fix this violation, most tools infer missing values but it is a safer practice to specify each value.

How to Debug and Fix

Ensure to specify each value correctly.

Example Code and/or Schematic

Consider the following snippet.

```
//test.v
module top (CLK, data, out, outbar);

input CLK;
input [1:0] data;
output [1:0] out, outbar;
reg [1:0] out, outbar;
```

```
always @(posedge CLK)
begin
    out = data;
    outbar = ~data;
end
```

```
endmodule
```

Following is the code snippet in the SDC file:

```
//test.sdc
set_driving_cell -lib_cell FD1 -min {data[0]}
set_driving_cell -lib_cell FD1 -max {data[0]}

set_driving_cell -lib_cell FD1 -min {data[1]}
```

Suppose, the [set_driving_cell](#) constraint is not defined in test.sdc, then this rule reports a violation.

Default Severity Label

Warning

Rule Group

Inp_Trans

Reports and Related Files

None

Inp_Trans04

Reports incorrectly specified input transition value

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *Inp_Trans04* rule reports input ports with transition values outside the technology limits.

The *Inp_Trans04* rule requires that the input transition value should not be greater than the maximum allowed transition value or less than or equal to the minimum allowed transition value specified. In case, the input transition value is specified using the *set_driving_cell* constraint, only presence of the cell in the library is checked.

The *Inp_Trans04* rule does not interpolate/extrapolate the transition value based on the drive strength of the cell versus the load value.

If the *set_driving_cell* constraint is not used, the comparison is made as follows:

- If no library file is supplied, the limits are set by the *default_max_transition* parameter and *default_min_transition* parameter with default value 5.0 and default value 0, respectively.
- If a library file is supplied and the clock object is a port or a block pin, compare with the default max transition in the library.
- If a library file is supplied and the clock object is a pin on a cell instance, compare with the min and max transition times for that pin.

Rule Exception(s)

If the transition value is 0 and the value of the *default_min_transition* parameter is also 0 then the *Inp_Trans04* rule reports a violation.

Parameter(s)

- *default_max_transition*: Default value is 5.0. Set the value to any positive float value to define the maximum transition value.
- *default_min_transition*: Default value is 0. Set the value to any positive float value to define the minimum transition value.

Constraint(s)

- *set_driving_cell*: (Optional): Use this constraint to set the port driving cell.

Messages and Suggested Fix

Message 1

The following message appears for an input port *<portname>* with transition value, *<num>*, that is more than the maximum value *<max>* specified in the corresponding `.lib` technology library or the *default_max_transition* parameter:

[WARNING] Input transition value *<num>* set on port "*<portname>*" is greater than maximum transition value *<max>* (specified through technology library or rule parameter)

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

If you do not fix this violation, the chip may not work in the current technology.

How to Debug and Fix

Set the value below the max value specified in the library for the driven pin.

Message 2

The following message appears for an input port *<portname>* with a transition value, *<num>*, less than or equal to the minimum value *<min>* specified in the corresponding `.lib` technology library or the *default_min_transition* parameter:

[WARNING] Input transition value *<num>* set on port "*<portname>*" is not greater than the minimum transition value *<min>* (specified through technology library or rule parameter)

Potential Issues

The violation message appears when input transition value on port is less than or equal to the minimum transition value.

Consequences of Not Fixing

If you do not fix this violation, the chip may not work in the current technology.

How to Debug and Fix

Set the value below the min value specified in library for the driven pin.

Example Code and/or Schematic

The *default_max_transition* parameter is set to 10. For the following snippet, the *Inp_Trans04* rule reports a violation message.

```
//test.sdc
set_input_transition 14 -max [get_ports data]
```

Default Severity Label

Warning

Rule Group

Inp_Trans

Reports and Related Files

None

Inp_Trans05

Reports when input transition rate is inconsistent with the associated clock transition rate

When to Use

Use this rule in the RTL phase and Pre-layout phase.

Description

The *Inp_Trans05* rule reports inconsistent input transition values between input ports and associated clocks in the RTL and Pre-layout analysis phases.

The *Inp_Trans05* rule compares the input transition value against the *set_clock_transition* value for the associated clock object or *set_annotated_transition* value for nets connected to clock pins of registers.

The *Inp_Trans05* rule reports a message in the following cases:

- **For worst-case analysis:** if the input transition value is less than the associated clock transition.
- **For best-case analysis:** if the input transition value is more than the associated clock transition.

Parameter(s)

None

Constraint(s)

- *set_clock_transition* (Optional): Use this constraint to specify transition time of register clock pins.
- *set_annotated_transition* (Optional): Use this constraint to set the transition time to be annotated on specified pins in the current design.

Messages and Suggested Fix

Message 1

The following message appears for an input port *<portname>* where the transition rate *<value1>* is inconsistent with the corresponding clock transition rate *<value2>*:

[WARNING] Transition rate `<value1>` specified for input "`<portname>`" is inconsistent with the transition rate (`<value2>`) specified for the associated clock. It should not be `<type>` than the clock transition rate for `<case-type>`-case analysis

Where, `<type>` and `<case-type>` can be more and worst or less and best.

Potential Issues

The violation message appears when transition rate of an input port is inconsistent with the transition rate of the associated clock.

Consequences of Not Fixing

If you do not fix this violation, mismatch between driver and driven capacity may lead to chip failure.

How to Debug and Fix

Specify the value between the max to min value.

Message 2

The following message appears for an input port `<portname>` where the transition rate `<value1>` is inconsistent with the associated annotated transition rate `<value2>`:

[WARNING] Transition rate `<value1>` specified for input "`<portname>`" is inconsistent with the associated annotated transition rate (`<value2>`). It should not be `<type>` than the annotated transition rate for `<case-type>`-case analysis

Where, `<type>` and `<case-type>` can be more and worst or less and best.

Potential Issues

The violation message appears when the transition rate for an input port is inconsistent with the associated annotated transition rate.

Consequences of Not Fixing

If you do not fix this violation, mismatch between driver and driven capacity may lead to chip failure.

How to Debug and Fix

Increase the rate of `set_annotated` value.

Message 3

The following message appears when neither best corner nor the worst corner cases are specified in the SDC files for design/block *<name>*:

```
[WARNING] Neither best nor worst corner cases specified in
schema for design/block "<name>"
```

Potential Issues

The violation message appears when neither best corner nor the worst corner case is specified in the SDC files for design/block.

Consequences of Not Fixing

If you do not fix this violation, mismatch between driver and driven capacity may lead to chip failure.

How to Debug and Fix

Specify the `-corner` option in the SGDC file.

Example Code and/or Schematic

If `current_design` in the SGDC file is specified as:

```
current_design top
sdc_data -type test.sdc
```

This rule reports a violation message.

Default Severity Label

Warning

Rule Group

Inp_Trans

Reports and Related Files

None

Inp_Trans06

Reports when input transition rate is inconsistent with the associated clock transition rate

When to Use

Use this rule in the Post-layout phase.

Description

The *Inp_Trans06* rule reports inconsistent input transition values between input ports and associated clocks in the post-layout analysis phase.

The *Inp_Trans06* rule compares the input transition value against the *set_clock_transition* value for the associated clock object.

The *Inp_Trans06* rule reports a message in the following cases:

- **For worst-case analysis:** if the input transition value is less than the associated clock transition.
- **For best-case analysis:** if the input transition value is more than the associated clock transition.

Parameter(s)

None

Constraint(s)

- *set_clock_transition* (Optional): Use this constraint to specify transition time of register clock pins.
- *set_annotated_transition* (Optional): Use this constraint to set the transition time to be annotated on specified pins in the current design.

Messages and Suggested Fix

Message 1

The following message appears for an input port *<portname>* where the transition rate *<value1>* is inconsistent with the corresponding clock transition rate *<value2>* (set at file *<filename>*, line *<num>*):

[WARNING] Transition rate *<value1>* specified for input "*<portname>*" is inconsistent with the input transition rate

(<value2>) specified for the associated clock (file <file-name>, line <num>). It should not be <type> than the clock transition rate for <case-type>-case analysis

Potential Issues

The violation message appears when the transition rate of an input port is inconsistent with the corresponding clock transition rate.

Consequences of Not Fixing

If you do not fix this violation, the STA on the basis of these numbers can be unreliable because the violation message indicates a typo in the transition rate of input signals and/or clocks.

How to Debug and Fix

Rule's violation highlights the constraints on data port and corresponding clock port.

You can fix the violation by making the transition rate consistent applied on the data port and the clock port.

Message 2

The following message appears when neither the best corner nor the worst corner cases are specified in the SDC files for design/block <name>:

[WARNING] Neither best nor worst corner cases specified in schema for design/block "<name>"

Potential Issues

The violation message appears when neither the best corner nor the worst corner cases are specified in the SDC files for design/block.

Consequences of Not Fixing

If you do not fix this violation, timing analyzer may presume any corner option and may result to wrong analysis.

How to Debug and Fix

- Click on the violation message.
- Rule highlights the [sdc_data](#) definition in the sgd file.
- Accordingly, specify the correct -corner option.

Example Code and/or Schematic

The test.sdc and test.sgcd files are specified as shown:

```
//test.sgcd
current_design top
sdc_data -type test.sdc -level postlayout -mode func -corner
Worst
//test.sdc
create_clock -name Clk1 -period 10.000000 -waveform {
0.000000 5.000000 } {CLK}
set_input_delay 2.3 -clock Clk1 {in1}
set_input_transition 1.2 CLK
set_input_transition 1.1 in1
```

This rule reports a violation message. To fix this violation, increase the value on the in1 port to value 1.2 or decrease the value on port CLK to 1.1.

Default Severity Label

Warning

Rule Group

Inp_Trans

Reports and Related Files

None

Inp_Trans07

Reports `set_load` for input ports when using `set_driving_cell`

When to Use

RTL phase, Pre-layout phase, Post-layout phase

Description

The *Inp_Trans07* rule reports input ports where a `set_driving_cell` constraint is set and either the `set_load` constraint is not set or is set to zero.

Parameter(s)

None

Constraint(s)

SDC

- `set_driving_cell` (Mandatory): Use to set the port driving cell.
- `set_load` (Optional): Use to set the capacitance on ports and nets in the current design.

Messages and Suggested Fix

Message 1

The following message appears for the input port `<port-name>` of the design/block `<name>` in which a `set_driving_cell` constraint is set but the `set_load` constraint is not set:

[WARNING] set_load is also required to specify transition with set_driving_cell for input port "<port-name>" of design/block "<name>"

Potential Issues

The cell specified in `set_driving_cell` might be driving pins in the block other than the specified port. In this case, `set_load` is required to enable implementation tools to accurately model the transition curves on the port.

Consequences of Not Fixing

If the load constraint is not specified, the tool could assume that the cell is

driving the specified port only and model an optimistic transition curve. It could lead to timing analysis failure.

How to Debug and Fix

The violation message indicates the port on which the [set_driving_cell](#) constraint is specified, but the [set_load](#) constraint is not specified. The [set_driving_cell](#) constraint is highlighted in the SDC file.

Update the SDC file with the missing [set_load](#) constraint.

Message 2

The following message appears for the input port `<port-name>` of the design/block `<name>` in which a [set_driving_cell](#) constraint is set and a [set_load](#) constraint is set with value 0:

[WARNING] set_load with non zero positive value is required to specify transition with set_driving_cell for input port "<port-name>" of design/block "<name>"

Potential Issues

Zero capacitance specified through [set_load](#) implies zero signal strength for the specified port. Positive signal strength is needed at the port. Therefore, the specified [set_load](#) constraint could be invalid.

Consequences of Not Fixing

Not fixing this violation could lead to timing analysis violation for the paths starting from the port.

How to Debug and Fix

The violation message indicates the port on which both the [set_driving_cell](#) and [set_load](#) constraints are specified, but the [set_load](#) constraint is set to zero. Both the [set_driving_cell](#) and [set_load](#) constraints are highlighted in the SDC file.

To resolve this violation, update the SDC file with a positive load value.

Example Code and/or Schematic

In the following SDC file snippet, the message appears because `set_load 0` is provided for the port for which [set_driving_cell](#) is specified.

```
set_driving_cell -lib_cell AN2 [get_ports A]
set_load 0 [get_ports A]
```

Default Severity Label

Warning

Rule Group

Inp_Trans

Reports and Related Files

None

Inp_Trans08

Reports any unusual usage of arguments in `set_input_transition/`
`set_driving_cell`

When to Use

Use this rule in the RTL, pre-layout, and post-layout phases.

Description

The `Inp_Trans08` rule reports `set_input_transition` constraints and `set_driving_cell` constraints specified with unusual arguments.

The `Inp_Trans08` rule reports `set_input_transition` constraints specified with `-clock` and `-clock_fall` arguments and the `set_driving_cell` constraints specified with `-clock`, `clock_fall`, `-max`, and `-min` arguments.

Parameter(s)

None

Constraint(s)

- `set_input_transition`(Mandatory): Use this constraint to specify the floating point number in library time units representing the transition.
- `set_driving_cell` (Optional): Use this constraint to set the port driving cell.

Messages and Suggested Fix

Message

The following message appears if `set_input_transition/set_driving_cell` constraint is specified with an unsupported argument in the SDC file:

```
[INFO] "Uncommon option(s) "<option_list>" used in
<command_type> command - not supported by all synthesis tools";
```

Potential Issues

The violation message appears when `set_input_transition/set_driving_cell` constraint is specified with an unsupported argument in the SDC file.

Consequences of Not Fixing

If you do not fix this violation, not all tools support some of these options.

This may cause the same constraints files to be interpreted differently by various tools in the flow.

How to Debug and Fix

Remove these uncommon options which are not supported by all synthesis tools.

Example Code and/or Schematic

In the following SDC file snippet, the message appears because `set_load 0` is provided for the port for which `set_driving_cell` is specified.

```
set_driving_cell -lib_cell AN2 [get_ports A]
set_load 0 [get_ports A]
```

Default Severity Label

Info

Rule Group

Inp_Trans

Reports and Related Files

None

Inp_Trans09

Reports use of `set_drive` constraints when not allowed

When to Use

Use this rule in the RTL phase, Pre-layout phase and Post-layout phase.

Description

The *Inp_Trans09* rule reports the use of `set_drive` constraints, when not allowed.

- The *Inp_Trans09* rule is primarily an informational rule.
- It helps report cases where the constraints are specified in terms of non-portable technology-specific or drive-strength numbers. Some design approaches disallow this methodology and prefer use of `set_driving_cell` constraint. Use the *Inp_Trans01a* rule to perform such checks.

Rule Exception(s)

If the *chip* rule parameter is set, the *Inp_Trans09* rule does not report such `set_drive` constraints.

Parameter(s)

None

Constraint(s)

- `set_driving_cell` (Optional): Use this constraint to set the port driving cell.
- `set_drive` (Optional): Use this constraint to set the resistance on specified input or inout ports in the current design.

Messages and Suggested Fix

Message

The following message appears for a `set_drive` constraint encountered in the SDC file:

```
[INFO] set_drive command used
```

Example Code and/or Schematic

The *Inp_Trans09* rule reports a message that states the *set_drive* constraint is being used in the SDC file.

```
//test.sdc  
set_drive 3 in
```

Default Severity Label

Info

Rule Group

Inp_Trans

Reports and Related Files

None

Input/Output Consistency Rules

The Input/Output Consistency Rules Sub-group `IO_Consis` contains the following rules:

Rule	Description
<i>IO_Consis01</i>	Reports inconsistent inter-block input/output delay constraints
<i>IO_Consis02</i>	<i>set_max_delay/set_min_delay</i> constraints that are not consistent between top level and block level
<i>IO_Consis04</i>	The path delay is inconsistent with the clock period
<i>IO_Consis07</i>	Incompatible cells specified with the <i>set_driving_cell</i> command

IO_Consis01

Reports inconsistent inter-block input/output delay constraints

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

The *IO_Consis01* rule checks consistency of constraints applied on output port of a block with the consistency of constraints applied on input port of another block

The *IO_Consis01* rule verifies that when an output port of block A is clocked by `clkA` and that output port drives possibly through a combinational logic. Similarly, an input port on block B is clocked by `clkB`, then clocks `clkA` and `clkB` must match in period and waveform or the period of clock `clkA` must be greater than the period of clock `clkB`.

If both period and waveform of the two clocks are different, only the period is reported.

Parameter(s)

None

Constraint(s)

- *set_input_delay* (Mandatory): Use this constraint to specify the arrival time of data relative to a clock.
- *set_output_delay* (Mandatory): Use this constraint to set output path delay value relative to a clock.

Messages and Suggested Fix

The following message appears when a *set_input_delay* constraint is specified for port `<port1-name>` clocked by `<clk1name>` and a *set_output_delay* constraint is specified (set at file `<filename>`, line `<num>`) for the corresponding port `<port2-name>` clocked by `<clk2-name>` in the preceding block. Also, the two clocks differ in period and/or waveform or the clock period of `<clk2-name>` is less than the clock

period of *<clk1-name>*:

[WARNING] *<property>* of clock (*<clk1-name>*) in *set_input_delay* at port '*<port1-name>*' is inconsistent with *<property>* of clock (*<clk2-name>*) in *set_output_delay* at port '*<port2name>*' (file *<file-name>*, line *<num>*)

Where, *<property>* can be the clock period in the format

Period: *<value>* or the waveform in the format.

waveform: {*<value>* *<value>* *<value>*...}.

Potential Issues

The violation message appears when property of a clock specified with *set_output_delay* constraint at port is inconsistent with property of another clock specified with *set_output_delay* constraint.

Consequences of Not Fixing

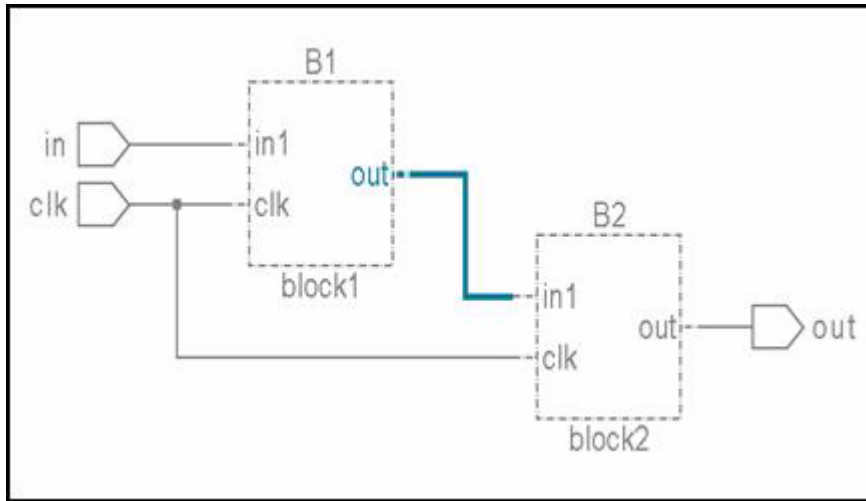
If you do not fix this violation, some of the tools may infer delays according to their assumption which may be contradicting with your intention and hence may cause chip failure.

How to Debug and Fix

Ensure to make the constraints consistent between the blocks.

Example Code and/or Schematic

Consider the following design.



The constraints are defined for block block1 as:

```
create_clock -period 5 [get_ports clk] -waveform {1 2 3 4}
set_output_delay 2 -clock clk [get_ports out]
```

The constraints are defined for block block2 as:

```
create_clock -period 5 -name clk_block2 [get_ports clk] -
waveform {1 3}
set_input_delay 1 -clock [get_clocks clk_block2] [get_ports
in1]
```

Where, *set_output_delay* defined on output port "out" (i.e. value 2) is not consistent with *set_input_delay's* value (i.e. value 1).

The *IO_Consis01* rule highlights the path from output port of a block to input port of another block.

Default Severity Label

Warning

Rule Group

IO_Consis

I/O Rules

Reports and Related Files

None

IO_Consis02

Reports when top-level `set_max_delay` or `set_min_delay` constraint is inconsistent with block-level constraint

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The `IO_Consis02` rule reports `set_max_delay/set_min_delay` constraints when the constraint value specified at the top-level is not greater than the sum of all constraint values specified for all the blocks under the top.

Parameter(s)

None

Constraint(s)

- `set_max_delay/set_min_delay` (Mandatory): Use this constraint to define maximum and minimum delay for timing paths.

Messages and Suggested Fix

The following message appears when the value `<value1>` of constraint `set_max_delay/set_min_delay <constr>` specified at the top-level is not greater than the sum `<value2>` of values of constraint `<constr>` specified for all blocks:

[WARNING] `<constr>` at top (`<value1>`) is not greater than the sum of `<constr>` of blocks(s) (`<value2>`)

Potential Issues

The violation message appears when the value of `set_max_delay/set_min_delay` constraint at top-level is not greater than the sum of values of `set_max_delay/set_min_delay` constraint at the block-level.

Consequences of Not Fixing

If you do not fix this violation, same signal is assigned to two different characteristics, one at the block-level and the other at the top-level. Clearly, one is incorrect. The one with the incorrect characteristics causes faulty timing analysis.

How to Debug and Fix

Increase the value at the top-level or decrease the value for blocks.

Example Code and/or Schematic

For the following snippet the *IO_Consis02* rule reports a violation that indicates the block-level sum is 2.1 which is greater than top value 2.0.:

```
//block1.sdc
set_max_delay 1 -to [get_ports out]

//block2.sdc
set_max_delay 1.1 -to out

//top.sdc
set_max_delay 2 -to out
```

Default Severity Label

Warning

Rule Group

IO_Consis

Reports and Related Files

None

IO_Consis04

Reports the path delay when it exceeds the clock period

When to Use

To determine whether the path delay is consistent with the clock period when there is a feed-through path through blocks. This rule is applicable to all design phases.

Description

The *IO_Consis04* rule checks the total path maximum delay is less than the available time for a timing path spanning across multiple blocks. It computes the delay of the segment within a block based on the *set_input_delay* and *set_output_delay* constraints specified on the ports of the blocks.

If the timing path starts/ends at top-level port, the *set_input_delay* (at start) and *set_output_delay* (at end) is also considered in the computation of delay available for the timing path.

If the *set_multicycle_path* constraint is specified for the timing path, the multiplier value is considered in the computation of the delay available for the timing path. The *set_multicycle_path* is considered with the *setup* option only. This rule requires the *-mode* argument defined for the *sdc_data* constraint to be same at top and block levels. However, if the top level mode is *flatTop* and block level mode is *block*, this rule performs the check.

For delays in a path specified with respect to clocks with different characteristics, this rule uses Shortened Clock Period (SCP) in the delay computation. For more details on SCP Calculation, refer to *Example 4*.

Refer to the *Parameter(s)* section to control the behavior of this rule for inter-block interconnect delays and tightly constrained timing paths.

NOTE: *For the computation of path-segment delay, all max-rise are considered together, and all max-fall are considered together.*

Rule Exceptions

The *IO_Consis04* rule does not check for the following conditions:

- If *set_input_delay/set_output_delay* is specified with *-min* option
- If any port on the timing path does not have an I/O delay specified

- Situations where the path is between top-level and block-level and there is no other intermediate block. The *Equiv_SDC_Block* rule covers these situations. Refer to the constraints management documentation for more details. The *IO_Consis04* rule requires timing paths to be spanning more than one block.

Parameter(s)

- *tc_inter_block_delay*: Default is 0. This signifies that there is no inter-block interconnect delay. Set the value between 0 and 1 to specify the net inter-block delay as a factor of the clock period.
- *tc_overconstrained_factor*: Default is 0.8. This value specifies the over-constrained delay as a factor of available time. Set the value between 0 and 1 to check that the timing path is not constrained too tightly. A total path delay being less than $0.8 * \text{available time}$ is reported as over-constrained.

Constraint(s)

SDC

- *set_input_delay* (Mandatory): Use to specify the arrival time of data relative to a clock.
- *set_output_delay* (Mandatory): Use to set output path delay value relative to a clock.
- *set_multicycle_path* (Optional): Use to define multi-cycle path.

Messages and Suggested Fix

Message 1

The following message appears when the path delay *<delay-val1>* for option *<delay-option>* exceeds the delay available *<delay-val4>* for path *<timing-path>*:

[WARNING] The path (*<timing-path>*) delay *<delay-val1>* for option *<delay-option>* (intra-blocks : *<delay-val2>*, interconnect : *<delay-val3>*), exceeds the delay available *<delay-val4>*

Where,

- intra-blocks is the sum of delay available inside the blocks

- interconnect is the delay between blocks

Potential Issues

Either the delay value specified for ports is incorrect, or the period of clock is incorrect.

Consequences of Not Fixing

The data is not be captured in expected time. Therefore, the timing analysis would fail, though there might not be a problem at the individual block level.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the timing path between the block boundaries. The total path delay exceeds the delay available. In addition, if [set_multicycle_path](#) constraint is taken into account for the delay computation, the constraint is highlighted in the SDC file.

To resolve this violation, update either the delay values specified on the ports or the period of the clock.

Refer to the [Parameter\(s\)](#), [Example 2](#), and [Example 5](#) sections for more information.

Message 2

The following message appears when the path delay <delay-val1> for option <delay-option> is over-constrained (limit : <delay-val4>) for path <timing-path>:

[WARNING] The path (<timing-path>) delay <delay-val 1> for option <delay-option> (intrablocks : <delay-val 2>, interconnect : <delay-val 3>), is overconstrained (limit : <delay-val 4>)

Where,

- intrablocks is the sum of delay available inside the blocks
- interconnect is the delay between blocks

Potential Issues

Either the delay value specified for ports is incorrect, or the period of clock is incorrect.

Consequences of Not Fixing

Paths constrained too tightly waste area, power, and run-time of the implementation tools.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the timing path between the block boundaries. The total path delay exceeds the delay available. In addition, if [set_multicycle_path](#) constraint is taken into account for the delay computation, the constraint is highlighted in the SDC file.

To resolve this violation, update either the delay values specified on the ports or the period of the clock.

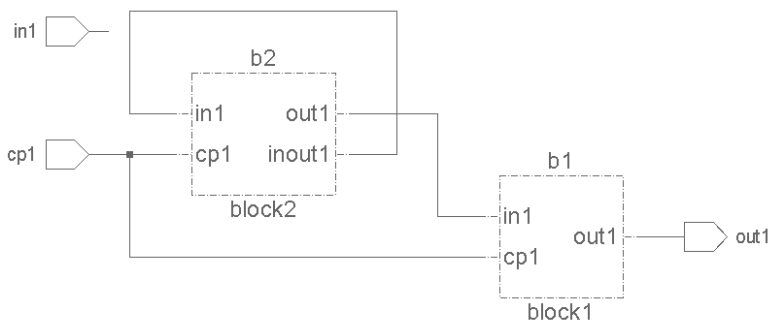
Refer to the [Parameter\(s\)](#) and [Example 3](#) sections to control the behavior of this rule for inter-block interconnect delays.

Example Code and/or Schematic

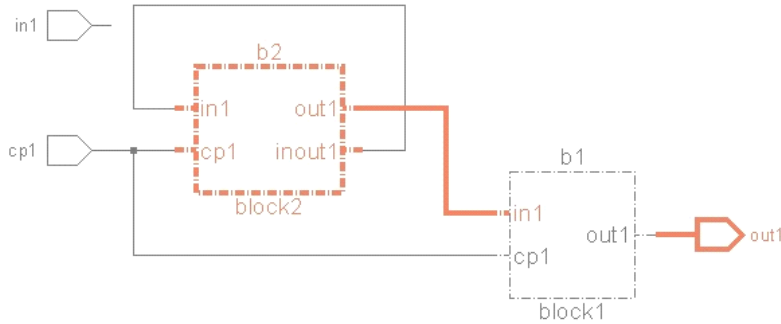
Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The violation message appears because the path delay exceeds the clock period. The calculations are shown in the message. The module schematic is as follows.



Example 2

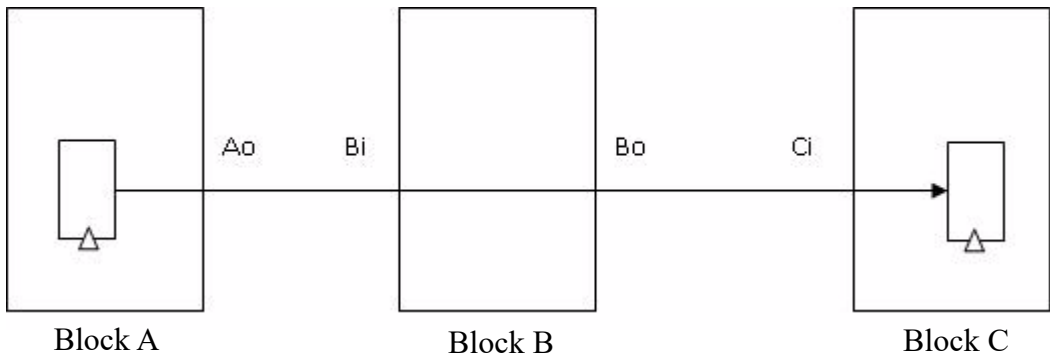
Test Case Files Not Available

This example illustrates single-cycle path with the same clock.

There is a path associated with a clock period of 10 and the *tc_inter_block_delay* parameter is set to 0.2. This means that the "2" is the delay through the interconnect. Therefore, the delay within all of the blocks should subtotal to less than 8. The timing path is as follows:

BA/F1/Q -> BA/OUT -> B/IN -> B/OUT ->

C/IN -> C/F2/D



If the delays specified at BA/OUT, B/IN, B/OUT, C/IN are 2,3,2,3 respectively, then the intrablock delay is 20 and the interconnect is 2. Therefore, the total path delay is 22 and the delay available is 10. Hence, a violation message is reported.

Example 3Test Case Files Not Available

This example illustrates single-cycle path with the same clock.

There is a path associated with a clock period of 10 and the [tc_inter_block_delay](#) parameter is set to 0.2. This means that the "2" is the delay through the interconnect. Therefore, the delay within all of the blocks should subtotal to less than 8. The timing path is as follows:

B1/F1/Q -> B1/OUT -> B2/IN -> B2/OUT ->

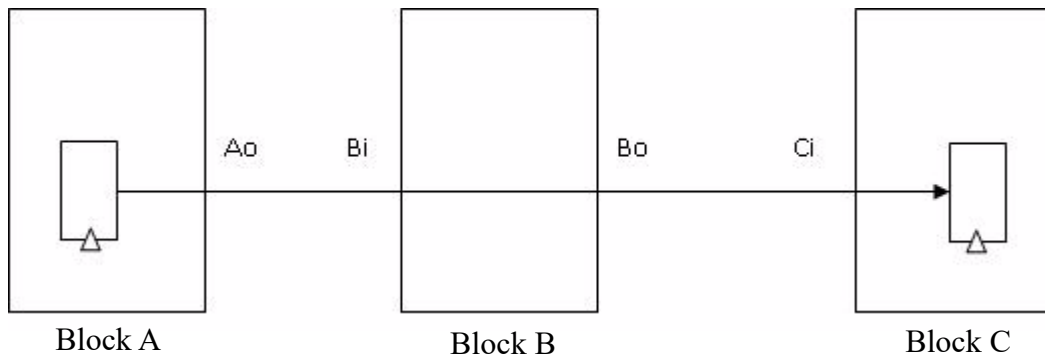
B3/IN -> B3/F2/D

If the [tc_overconstrained_factor](#) parameter is 0.8, then the total delay should exceed 8. Otherwise, it is over-constrained.

If the delays specified at B1/OUT, B2/IN, B2/OUT, B3/IN are 9, 3, 4, 9 respectively, the intrablock delay is 5 and the interconnect is 2. Therefore, the total path delay is 7, which is less than the limit 8. Hence, a violation message is reported.

Example 4Test Case Files Not Available

This example illustrates how the *IO_Consis04* rule calculates SCP.



In this example, the path is launched from a flip-flop in Block A. It then passes through Block B and is captured in Block C.

In the SDC file:

- For Block A, there is a *set_output_delay* constraint set on the *Ao* port.
- For Block B, there is a *set_input_delay* constraint set on the *Bi* port and a *set_output_delay* set on the *Bo* port.
- For Block C, there is a *set_input_delay* constraint set on the *Ci* port.

In addition, CLK1 has a period of 4 ns and it drives Block A. CLK2 has a period of 10ns and it drives Block C. The clocks are constrained in the following manner:

```
create_clock -name CLK1 -period 4 -waveform { 1 4 }
```

```
create_clock -name CLK2 -period 10 -waveform { 4 8 }
```

Since the edges are not aligned, the *IO_Consis04* rule calculates the SCP by taking the minimum difference between the active edges of CLK1 and CLK2. In this case:

SCP = 1ns

If active edges of both clocks are aligned, SCP is computed as:

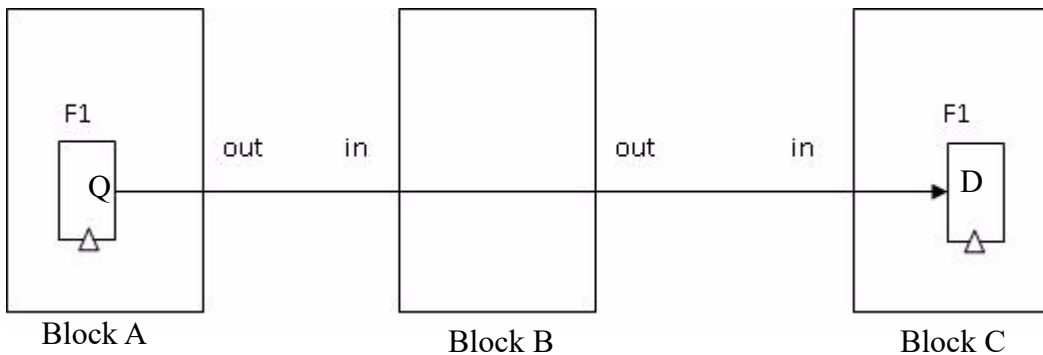
- When the capture clock period is greater than the launch clock period, SCP is the difference between the capture clock period and the launch clock period.

- When the capture clock period is less than the launch clock period, SCP is the launch clock period.

Example 5

Test Case Files Not Available

This example illustrates multi-cycle path with different synchronous clocks.



The timing path is as follows:

A/F1/Q----->A/out----->B/in----->B/out----->C/in----->C/F1/D

CLK1 has a period of 4 ns and it drives Block A. CLK2 has a period of 10ns and it drives Block C. The clocks are constrained in the following manner:

```
create_clock -name CLK1 -period 4 -waveform { 1 4}
```

```
create_clock -name CLK2 -period 10 -waveform {4 8}
```

Therefore,

SCP = 1ns

A multi-cycle path constraint is given as follows:

```
set_multicycle_path 2 -start -setup -from A/F1/Q -to C/F1/D
```

The reference clock period (RCP) with `-start` in the [set_multicycle_path](#) constraint is 4 ns. The total time available for setup is 5 ns.

Constraint given for Block A

```
set_output_delay -clock CLK1 1 [get_ports out]
```

Constraint given for Block B

```
set_input_delay -clock CLK1 2 [get_ports in]
```

```
set_output_delay -clock CLK2 3 [get_ports out]
```

```
set_multicycle_path 2 -start -hold -from in -to out
```

Constraint given for Block C

```
set_output_delay -clock CLK2 1 [get_ports out]
```

Assume the default values of the [tc_inter_block_delay](#) (Default = 0) and [tc_overconstrained_factor](#) (Default = 0.8) parameters are set.

The total path delay is 8 ns. Since the total path delay (8) is greater than the delay available (5), the *IO_Consis04* rule reports a violation.

Default Severity Label

Warning

Rule Group

IO_Consis

Reports and Related Files

None

IO_Consis07

Reports incompatible cells specified with `set_driving_cell`

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *IO_Consis07* rule reports incompatible cells specified with the [set_driving_cell](#) constraint.

The driving cell specified for an input/inout port should be compatible with the cell-pin to which the port is connected. That is, the voltage swing of the driving cell should be same as that of the cell-pin. The voltage swing information is modeled as the `voltage_map` attribute or the `power_supply` construct in the gates library (.lib files).

The *IO_Consis07* rule reports a violation if:

- Neither the `voltage_map` attribute, nor the `power_supply` construct is defined in the library.
- The `voltage_map` value or the `power_rail` value does not match (or does not exist) for the cell connected to the port.

The `voltage_map` or `power_rail` values for a driving cell are inferred as follows:

- Value of the `related_power_pin` attribute on the pin specified with the `-pin` option.

If the `-pin` option is not specified, then value is inferred from the `related_power_pin` attribute set on any output pin of the driving cell.

- Value of the `output_signal_level` attribute on the pin specified with the `-pin` option.

If the `-pin` option is not specified, the value is inferred from the value of the `output_signal_level` attribute set on any output pin of the driving cell.

- The value of the `default_power_rail` attribute of the driving cell's technology library.

The `voltage_map` or `power_rail` values for the cell connected to the port are inferred as follows:

- Value of the `related_power_pin` attribute on the pin reached through the port.
- Value of the `input_signal_level` attribute on the pin reached through the port.
- Value of the `default_power_rail` attribute of the parent cell's technology library.

If the `related_power_pin` attribute is specified on the pin, then the `IO_Consis07` rule obtains the value corresponding to this pin from the `voltage_map` attribute defined in the `.lib` file. If the `input_signal_level` attribute or the `output_signal_level` attribute or the `default_power_rail` attribute is specified on the pin, the `IO_Consis07` rule obtains the value corresponding to this pin from the `power_rail` attribute defined in the `.lib` file.

Parameter(s)

None

Constraint(s)

- [`set_driving_cell`](#) (Mandatory): Use this constraint to set attributes on input or inout ports, specifying that a library cell or pin drives the ports.

Messages and Suggested Fix

Message 1

The following message appears when the voltage value `<volt1-value>` of driving cell `<cell1-name>` specified for port `<port-name>` is not same as the voltage value `<volt2-value>` of cell `<cell2-name>` driven by the port:

```
[WARNING] Cell '<cell-name>' (<arg1>) specified with
set_driving_cell is not compatible with cell '<cell-name>'
(<arg2>) connected through port '<port-name>'
```

Where, `<arg1>` can be any of the following:

- pin '<pin-name>', voltage_map '<related-pin-name>(<voltage_value>)'
- pin '<pin-name>', power_rail '<related-pin-name>(<voltage_value>)'
- power_rail '<related-pin-name>(<voltage_value>)'

<arg2> can be any of the following:

- pin '<pin-name>', voltage_map 'not found'
- pin '<pin-name>', power_rail 'not found'
- pin '<pin-name>', voltage_map '<related-pin-name>(<voltage_value>)'
- pin '<pin-name>', power_rail '<related-pin-name>(<voltage_value>)'

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

Not Applicable.

How to Debug and Fix

To fix the violation user should specify correct [set_driving_cell](#) constraint with correct cell pin.

Message 2

The following message appears when the required attributes are not present in the gates library <lib-name>:

[INFO] Check could not be made, due to lack of required attributes in the library '<lib-name>'

Potential Issues

Not Applicable.

Consequences of Not Fixing

Not Applicable.

How to Debug and Fix

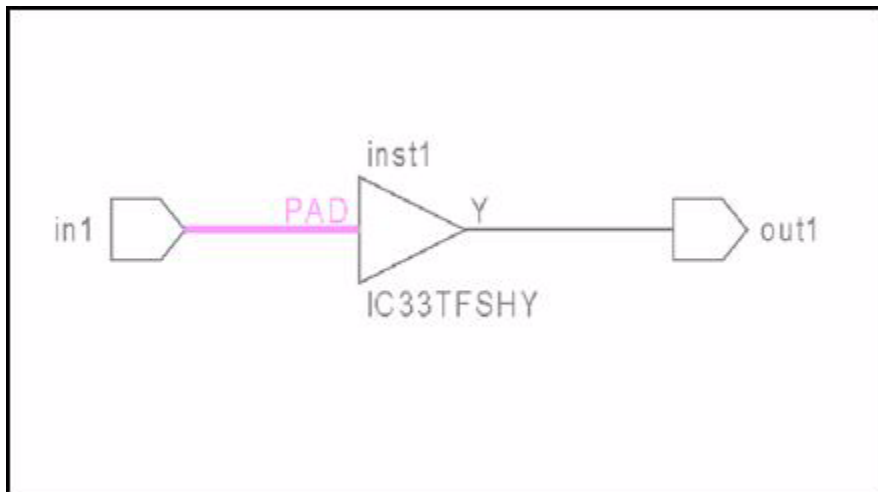
Not Applicable.

Example Code and/or Schematic

For the following snippet, the *IO_Consis07* rule reports a violation

```
message.  
module top(in1, out1) ;  
    input in1;  
    output out1;  
    IC33TFSHY inst1(.Y(out1), .PAD(in1));  
Endmodule
```

```
// test.sdc  
set_driving_cell -lib_cell PAD1 -pin Y -library  
power_rail.db:pr_sample {in1}
```



Default Severity Label

Warning

Rule Group

IO_Consis

I/O Rules

Reports and Related Files

None

Output Delay Rules

The Output Delay Rules Sub-group `Op_Del` contains the following rules:

Rule	Description
<i>Op_Del01</i>	Runs the <i>Op_Del01a</i> , <i>Op_Del01b</i> , and <i>Op_Del01c</i> rules
<i>Op_Del01a</i>	Output ports without output delay constraints
<i>Op_Del01b</i>	Output ports without output delay constraints
<i>Op_Del01c</i>	Output ports without a <i>set_output_delay</i> constraint set and with any of the <i>set_case_analysis/set_disable_timing/set_false_path</i> constraints set
<i>Op_Del02</i>	Output ports having Output Delay constraints with incorrect min and max values
<i>Op_Del03</i>	Runs the <i>Op_Del03a</i> and <i>Op_Del04</i> rules
<i>Op_Del03a</i>	Output ports having Output Delay constraints using incorrect clocks
<i>Op_Del03b</i>	Output ports having Output Delay constraints using incorrect clocks
<i>Op_Del04</i>	<i>set_output_delay</i> constraints without either <code>min</code> or <code>max</code> argument
<i>Op_Del05</i>	Output constraints specified with clock other than virtual clocks
<i>Op_Del07</i>	Output ports where the output delay value is more than the user-specified percentage of the corresponding clock period
<i>Op_Del07a</i>	Output ports where the output delay value is within/outside (selectable) of the specified percentage range of the corresponding clock period
<i>Op_Del08</i>	Output ports with <i>set_output_delay</i> constraints set for two different clocks without the <code>add_delay</code> argument
<i>Op_Del09</i>	Bus output ports where not all bits have the same <i>set_output_delay</i> constraint delay values
<i>Op_Del10</i>	<i>set_output_delay</i> constraints specified with the <code>level_sensitive</code> argument
<i>Op_Del11</i>	<i>set_output_delay</i> constraints specified with the <code>clock_fall</code> argument

Rule	Description
<i>Op_Del12</i>	<i>set_output_delay</i> constraints specified with the <i>network_latency_included</i> or the <i>source_latency_included</i> arguments that are unsupported (for synthesis) arguments
<i>Op_Del13</i>	<i>set_output_delay</i> constraints set with negative value
<i>Op_Del14</i>	<i>set_output_delay</i> constraints specified with wrong/incorrect set of clocks
<i>Op_Del15</i>	Reports internal pins on which <i>set_output_delay</i> is specified

Op_Del01

Output does not have set_output_delay

The *Op_Del01* rule runs the *Op_Del01a*, *Op_Del01b*, and *Op_Del01c* rules.

Op_Del01a

Reports a violation when output does not have `set_output_delay`

When to Use

Use this rule for the RTL, pre-layout, and post-layout phases.

Description

The *Op_Del01a* rule reports output/inout ports without a `set_output_delay` constraint set.

The *Op_Del01a* rule reports a violation for every output/inout port without a `set_output_delay` constraint and satisfying any of the following conditions:

- The output/inout port is driven by the output pin of a sequential cell
- The output/inout port is not driven by the output pin of a sequential cell but a combinational path exists from this output/inout port to a input/inout port and this output/inout port is not constrained by the `set_max_delay/set_min_delay` constraints.

In case of multiple clocks defined on the same object using the `-add` option of the `create_clock/create_generated_clock` constraint, the *Op_Del01a* rule retains only the last clock defined without the `-add` option and any clocks defined with the `-add` option after this last clock.

If `tc_ignore_te` is set to `no` and `ignore_io_if_fp` is set to `yes` then the rule will consider the effect of `set_false_path` constraint also.

Prerequisite

None

Rule Exceptions

The *Op_Del01a* rule ignores the following:

- Clock is reaching to the port.
- No valid timing path exists to the port. The reason could be: path is hanging, constant value is reaching, path is disabled by `set_disable_timing` constraint or there exists only blackbox in the fan-in of the port.

Parameter(s)

- *tc_ignore_te* (Optional): Default value is *yes*. Set the value to *no* to consider *set_multicycle_path* constraints.

ignore_io_if_fp (Optional): Default value is *no*. Set the value to *yes* to consider *set_false_path* constraints otherwise they are ignored.

In GuideWare2.0, the default value of *tc_ignore_te* is *no* and the default value of *ignore_io_if_fp* is *yes*.

- *tc_ignore_async_ports*: Default is *unspecified* and this rule does not ignore asynchronous ports. Specify the file name that contains the list of asynchronous ports, which this rule should ignore.
- *tc_ignore_if_clk_op_port*: Default is *no* and unconstrained output ports, which have a clock propagating to them, are reported. Set this parameter to *yes* to not report such output ports.

Constraint(s)

- *set_output_delay* (Optional): Use to specify the output path delay value for the port.
- *set_max_delay/set_min_delay* (Optional): Use to specify the maximum and minimum delay for the timing path.
- *create_clock* (Optional): Use to create the clock object.
- *create_generated_clock* (Optional): Use to create the generated clock object.

Messages and Suggested Fix

If the name of the port, *<port-name>*, is listed as an asynchronous port through the *tc_ignore_async_ports* parameter, the severity of each message listed in this section is INFO.

Message 1

The following message appears when an output/inout port

<port-name> of design/block *<name>* does not have a corresponding *set_output_delay* constraint and the output/inout port:

- Is being reached by any output pin of a sequential cell, which is not driven by any clock

- Is not being reached by any output pin of a sequential cell but is driven by an input/inout port through a combinational path

[Op_Del 01a_01] [WARNING] Output delay constraint not set on Port "<port-name>" of design/block <name>

Or,

[Op_Del 01a_02] [INFO] Output delay constraint not set on Port "<port-name>" of design/block <name>

Potential Issues

The violation message appears, if the [set_output_delay](#) constraint is not specified for the port, but there exists a valid timing path to this port.

Consequences of Not Fixing

Output delays are used to model external delays leaving the output ports of the constraint module. Timing analysis will be invalid without the proper [set_output_delay](#) constraint. The timing result may give a false sense of security and you may get silicon timing failure. Tools may assume optimistic delay from the ports.

How to Debug and Fix

To fix the violation, review the schematic highlighted in the Console GUI and specify the [set_output_delay](#) constraint in the SDC file.

Message 2

The following message appears when an output/inout port <port-name> of design/block <name> does not have any [set_output_delay](#) constraint set and the input/inout port reaches the data pin of a sequential cell (triggered by clocks <clk-name-list>) in its fan-out:

[Op_Del 01a_03] [WARNING/INFO] Output delay constraint not set on Port "<port-name>" of design/block <name> and output delay needs to be specified with clock(s) <clk-name-list>

Potential Issues

The violation message appears if the [set_output_delay](#) constraint is not specified for the port, but there exists a valid timing path to this port.

Consequences of Not Fixing

Output delays are used to model external delays leaving the output/input ports of the constraint module. Timing analysis will be invalid without proper [set_output_delay](#) constraint. The timing result may give a false sense

of security and you may get silicon timing failure. Tools may assume optimistic delay from the ports.

How to Debug and Fix

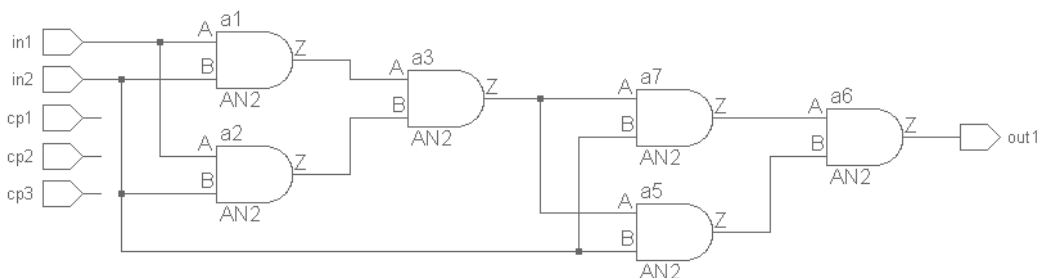
To fix the violations, review the schematic highlighted in the Console GUI and, in the SDC file, specify the `set_output_delay` constraint with respect to the reported clock.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top.sdc is as follows:

```
top.sdc top.sdc
set_multicycle_path 2 -from {in1} -through {a3/Z} -to {out1}
```

The violation message indicates that the `-through` argument with the `set_multicycle_path` constraint is not required. In the top.sdc file, the corresponding constraint is shown.

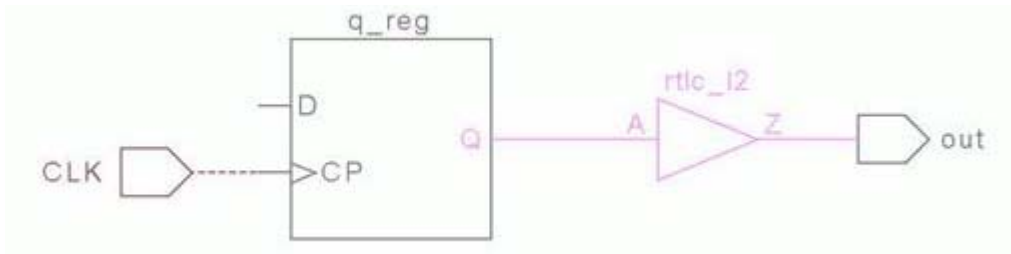
Example 2

Test Case Files Not Available

Consider the example:

```
create_clock -name C1 -period 10 [get_ports CLK]
```

Here, no output delay is specified for port out. The rule will report violation for missing *set_output_delay* constraint with respect to clock C1. The following schematic is generated.

**Default Severity Label**

Warning

Rule Group

Op_Del

Reports and Related Files

None

Op_Del01b

Identifies output/inout port that is not constrained by `set_output_delay`

When to Use

To identify ports that are not constrained with the `set_output_delay` constraint. This rule is applicable to the RTL, pre-layout, and post-layout phases.

Description

The `Op_Del01b` rule identifies output/inout ports that do not have a `set_output_delay` constraint set and satisfy any of the following conditions:

- The output/inout port is driven by the output pin of a sequential cell
- The output/inout port is not driven by the output pin of a sequential cell. In addition, a combinational path exists from this output/inout port to an input/inout port and this output/inout port is not constrained by the `set_max_delay/set_min_delay` constraints.
- If a pin is constrained with the `set_clock_sense/set_sense` command using the `-stop_propagation` option. Refer to the *Stopping Clock Propagation* section for details.

All outputs excluding clocks must be constrained. Output delays are used to model external delays leaving the output/inout ports of the constrained module. These delays are defined relative to a real or virtual clock and are specified to the left of the active clock edge.

In case of multiple clocks defined on the same object using the `-add` option of the `create_clock/create_generated_clock` constraint, the `Op_Del01b` rule retains only the last clock defined without the `-add` option and any clocks defined with the `-add` option after this clock.

Rule Exceptions

The `Op_Del01b` rule ignores the following:

- Path to an output/inout port that is hanging
- Path to an output/inout port with a `set_case_analysis` constraint
- Path to an output/inout port with a `create_clock` constraint
- Path to an output/inout port with a `create_generated_clock` constraint

- Path to an output/inout port from a blackbox
- Output/inout port with a `set_dont_touch_network` constraint
- Output/inout port with a `set_ideal_network` constraint
- Path from an input/inout port having a pin constrained with `set_clock_sense/set_sense` command using the `-stop_propagation` option. Refer to the *Stopping Clock Propagation* section for details.

Parameter(s)

- If the value of the `tc_ignore_te` parameter is set to `no` and the value of the `ignore_io_if_fp` parameter is set to `yes`, the `set_false_path` constraints are taken into consideration, if specified.

In GuideWare2.0, the default value of `tc_ignore_te` is `no` and the default value of `ignore_io_if_fp` is `yes`.

- `tc_ignore_async_ports`: Default is `unspecified` and this rule does not ignore asynchronous ports. Specify the file name that contains the list of asynchronous ports, which this rule should ignore.
- `tc_ignore_io_if_minmax_delay`: Default is `no` and this rule reports missing IO delays. Set this parameter to `yes` to ignore missing IO delays, when `set_max_delay` or `set_min_delay` is specified, even if the port is connected to a register.
- `tc_ignore_if_clk_op_port`: Default is `no` and unconstrained output ports, which have a clock propagating to them, are reported. Set this parameter to `yes` to not report such output ports.

Constraint(s)

SDC

- `set_clock_sense/set_sense` (Optional): Use to specify the clock propagation conditions.
- `set_false_path` (Optional): Use to identify paths in a design that are to be marked as false, so that they are not considered during timing analysis. Refer to the *Parameter(s)* section for details on how to consider `set_false_path` constraints.
- `set_output_delay` (Optional): Use to set the output path delay value.

- `create_clock` (Optional): Used to create a clock.
- `create_generated_clock` (Optional): Used to create a clock.

Messages and Suggested Fix

Message 1

The following message appears when an output/inout port `<port-name>` of design/block `<name>` does not have a corresponding `set_output_delay` constraint and the output/inout port:

- Is being reached by any output pin of a sequential cell, which is not driven by any clock
- Is not being reached by any output pin of a sequential cell but is driven by an input/inout port through a combinational path

[Op_Del 01b_01] [WARNING] Output delay constraint not set on Port "`<port-name>`" of design/block `<name>`

Or,

[Op_Del 01b_02] [INFO] Output delay constraint not set on Port "`<port-name>`" of design/block `<name>`

Potential Issues

Implementation tools might assume zero delay value for the port.

Consequences of Not Fixing

This is a design error because the timing analysis results are not valid without proper `set_output_delay` constraints. The timing results may give a false sense of security and you may get silicon timing failure. In addition, implementation tools may assume optimistic delay to the ports.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the path from the output or inout port to the output pins of the sequential cells, inout, or input port. This violation message appears because there is a valid timing path ending at the port, but there is no output delay constraint applied on the port.

Check the following:

- The `clock`, `set_dont_touch_network`, `set_ideal_network`, `set_case_analysis` constraints are not applied on this port.

- The timing path to the port is not negated by the [set_false_path](#) constraint (only if the [tc_ignore_te](#) parameter is set to no and the [ignore_io_if_fp](#) parameter is set to yes).
- The timing path to another port does not have the [set_max_delay/](#)
[set_min_delay](#) constraint set.

Message 2

The following message appears when an output/inout port `<port-name>` of design/block `<name>` does not have any [set_output_delay](#) constraint set and the output/inout port reaches the output pin of a sequential cell (triggered by clocks `<clk-name-list>`) in its fan-in:

```
[Op_De] 01b_03] [WARNING/INFO] Output delay constraint not set on
Port "<port-name>" of design/block <name> and output delay
needs to be specified with clock(s) <clk-name-list>
```

Potential Issues

Implementation tools might assume an undesired delay value for the path from the port to the flip-flop sampled by the clock. It could lead to incorrect timing analysis for the port.

Consequences of Not Fixing

This is a design error because the timing analysis results are not valid without proper [set_output_delay](#) constraints. The timing results may give a false sense of security and you may get silicon timing failure. In addition, implementation tools may assume optimistic delay to the ports.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the path from the port to the output pins of the sequential cells and the clock-paths to the clock pins of the sequential cells. This violation message appears because there is a valid timing path ending at the port, but there is no output delay constraint applied on the port.

Check the following:

- The clock, [set_dont_touch_network](#), [set_ideal_network](#), [set_case_analysis](#) constraints are not applied on this port.
- The timing path to the port is not negated by the [set_false_path](#) constraint (only if the [tc_ignore_te](#) parameter is set to no and the [ignore_io_if_fp](#) parameter is set to yes).

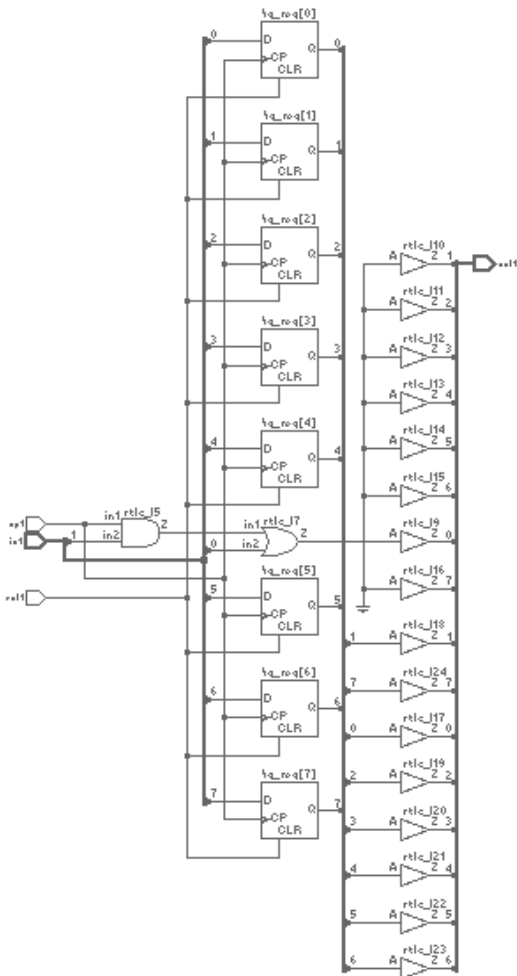
- The timing path to another port does not have the [set_max_delay/](#)
[set_min_delay](#) constraint set.

Example Code and/or Schematic

Example 1[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:

I/O Rules



The top.sdc is as follows:

```

top.sdc x
1 set_output_delay 3.000000 { out1[0] out1[1] out1[2] out1[3] out1[4] }
2

```

The violation message appears because the [set_output_delay](#) constraint is not defined on the port out1 [7:5]. In the top.sdc file, the corresponding constraint is shown.

Example 2Test Case Files Not Available

In the following example, the *Op_Del01b* rule reports a violation for the out1 and out2 ports because both ports are not constrained with [set_output_delay](#) and there exists a path to the sequential output pin from out1 and a combinational path to out2 from in.

```
--Design begin-
Input in;
Input clk;
Output out1, out2;
Flop f1( in, clk, out1 );
Assign out2 = in;
--Design end-

--SDC begin-
create_clock -period 10 clk
--SDC end-
```

Default Severity Label

Warning

Rule Group

Op_Del

Reports and Related Files

None

Op_Del01c

Reports a violation when timing analysis is not required by the output/inout port

When to Use

Use rule for the RTL, Pre-layout and Post-layout phases.

Description

The *Op_Del01c* rule reports output/inout ports with a *set_output_delay* constraint set and satisfy the following conditions:

- Path to an output/inout port with *create_clock*, *create_generated_clock*, *set_case_analysis*, or *set_disable_timing* constraints.
- Path to an output/inout port that is hanging or is from a black box.
- Path to an output/inout port is not being reached from the output pin of a sequential cell
- The output/inout port with a *set_max_delay/set_min_delay* constraint, which has a combinational path to an input/inout port

Parameter(s)

- *ignore_io_if_fp* (Default is no) and *tc_ignore_te* (Default is yes): By default, the *Op_Del01a* rule does not consider the effect of the *set_false_path* constraint. Set *tc_ignore_te* rule to no and *ignore_io_if_fp* to yes to consider the *set_false_path* constraints.

In GuideWare2.0, the default value of *tc_ignore_te* is no and the default value of *ignore_io_if_fp* is yes.

- *tc_ignore_async_ports*: Default is unspecified and this rule does not ignore asynchronous ports. Specify the file name that contains the list of asynchronous ports, which this rule should ignore.
- *tc_ignore_if_clk_op_port*: Default is no and unconstrained output ports, which have a clock propagating to them, are reported. Set this parameter to yes to not report such output ports.

Constraint(s)

- *set_output_delay* (Mandatory): Use to specify the output path delay value for a port.
- *set_max_delay/set_min_delay* (Optional): Use to specify the maximum and minimum delay for the timing path.
- *create_clock* (Optional): Use to create a clock object.
- *create_generated_clock* (Optional): Use to create a generated clock object.
- *set_case_analysis* (Optional): Use to specify that a port or pin is at a constant logic value.
- *set_disable_timing* (Optional): Use to disable a timing arc.

Messages and Suggested Fix

Message 1

The following message appears when an output/inout port *<port-name>* has a corresponding *set_output_delay* constraint and one of the specified constraint *<constr>* has been set on the output/inout port or on every fan-out path of the output/inout port:

```
[INFO] "<constr>" constraint has been set for the port "<port-name>" hence set_output_delay need not be specified
```

Where, *<constr>* can be *set_disable_timing*, *set_false_path*, *create_clock*, *set_max_delay/set_min_delay*

Potential Issues

This violation message appears if the *set_output_delay* constraint is not required because there is no valid timing path for this port.

Consequences of Not Fixing

Output delays are used to model external delays leaving the output/inout ports of the constraint module. Timing analysis will be invalid without proper *set_output_delay* constraint. The timing result may give a false sense of security and you may get silicon timing failure. Tools may assume an optimistic delay from the ports.

How to Debug and Fix

The SDC line is highlighted in the Console GUI.

To resolve this violation message, remove the *set_output_delay* constraint. Sometime, you can resolve this violation by removing the commands reported in the violation message.

Message 2

The following message appears when an output/inout port *<port-name>* is not driven by a sequential cell:

[INFO] No sequential cell is driving the port *<port-name>* hence *set_output_delay* need not be specified

Potential Issues

The violation message appears if the *set_output_delay* constraint is not required as there is no valid timing path for this port.

Consequences of Not Fixing

Output delays are used to model external delays leaving the output/input ports of the constraint module. Timing analysis will be invalid without proper *set_output_delay* constraint. The timing result may give a false sense of security and you may get silicon timing failure. Tools may assume optimistic delay from the ports.

How to Debug and Fix

Use the Show Fan-in/Fan-out Code feature of the Console GUI to check whether any valid timing path exists to this port.

The SDC line is highlighted in the Console GUI. To resolve this violation message, remove the *set_output_delay* constraint.

Message 3

The following message appears when a *set_output_delay* constraint is applied on an output/inout port *<port-name>* that is either constrained with the *set_case_analysis* constraint or connected to the supply (1 or 0):

[INFO] Constant value reaches to port *<port-name>* hence *set_output_delay* need not be specified

Potential Issues

The violation message appears explicitly states the potential issue.

Consequences of Not Fixing

Output delays are used to model external delays leaving the output/input ports of the constraint module. Timing analysis will be invalid without

proper [set_output_delay](#) constraint. The timing result may give a false sense of security and you may get silicon timing failure. Tools may assume optimistic delay from the ports.

How to Debug and Fix

Use the *Show_Case_Analysis* rule to validate whether a constant value is reaching the port.

The SDC line of both constraints are highlighted in the Console GUI. To resolve this violation, remove the [set_output_delay](#) constraint.

Example Code and/or Schematic

In this example, a timing path to the port out is not valid because of the [set_false_path](#) constraint. Therefore, the *Op_De101c* rule reports a violation.

```
create_clock -name C1 -period 10 [get_ports clk]
set_output_delay 1 -clock C1 [get_ports out]
set_false_path -to [get_ports out]
```

Default Severity Label

Info

Rule Group

Op_De1

Reports and Related Files

None

Op_Del02

Reports inconsistency in min/max delay of output constraint

When to Use

Use this rule for the RTL, pre-layout, and post-layout phases.

Description

The *Op_Del02* rule reports inconsistency between the delay value and the clock period for a given *set_output_delay* constraint. The *Op_Del02* rule checks if the delay value is less than the clock period.

However, if the *set_max_delay* constraint is applied on a port, the value of this constraint takes precedence over the values specified by the *set_output_delay* (SOD) constraint.

The rule reports a violation if:

- Min (SOD -min) is not specified and SOD -max > period
- Max (SOD -max) is not specified and SOD -min > period
- SOD -max > period > SOD -min
- SOD -max > SOD -min > period
- SOD -min > period > SOD -max
- SOD -min > SOD -max
- SOD -min > SOD -max > period

Conditions for the above-mentioned Constraints:

- SOD -max is value specified with option -max for a given *set_output_delay*.
- SOD -min is value specified with option -min for a given *set_output_delay*.
- The period is equal to the clock period of clock for which *set_output_delay* is specified. The period = 0.5 * clock_period if clock_fall is specified with *set_output_delay* constraint. This is only valid when there is no clock inversion. However, if clock inversion is taking place and you have not specified clock_fall, the period = 0.5 * clock_period

***Op_Del02* and *set_multicycle_path* constraint**

- The *Op_Del02* rule considers *set_multicycle_path* constraint if multi-cycle path is applied between the port and the clock with parameter *tc_ignore_te* set to no.
- The *Op_Del02* rule takes the period = MCP_multiplier * clock_period if *set_multicycle_path* is applied on the given port and clock specified in constraint *set_output_delay*.

Rule Exception(s)

- The *Op_Del02* rule ignores the *set_output_delay* constraint specified without the clocks.
- It ignores the port specified with *set_output_delay* constraint if all the paths from port to clocks are falsified with parameter *tc_ignore_te* and *ignore_io_if_fp* values set as yes and no, respectively.

Parameter(s)

- *tc_ignore_te* (Default is yes): Set to no to consider the effect of *set_multicycle_path* constraints. When multiple *set_multicycle_path* constraints are specified, the *Op_Del02* rule will consider the constraint having the minimum multiplier value. The rule ignores the *set_multicycle_path* constraint with the -hold option.
- *ignore_io_if_fp* (Default is no) and *tc_ignore_te* (Default is yes): By default, the *Op_Del01a* rule does not consider the effect of the *set_false_path* constraint. Set *tc_ignore_te* rule to no and *ignore_io_if_fp* to yes to consider the *set_false_path* constraints.

In GuideWare2.0, the default value of *tc_ignore_te* is no and the default value of *ignore_io_if_fp* is yes.

Constraint(s)

- *set_output_delay* (Mandatory): Use to specify the output path delay value for the port. If the *set_output_delay* constraint has the -clock_fall option specified, the delay is compared with the part of the clock waveform starting at the falling edge. Therefore, the delay is compared with half the clock period for default waveforms.
- *set_multicycle_path* (Optional): Use to modify the single-cycle timing relationship of a constrained path.

Messages and Suggested Fix

Message 1

The following message appears when an output port `<port-name>` has a `set_output_delay` constraint set with a `-min` argument value `<value1>` that is equal to or greater than the clock period `<value2>` of the corresponding clock `<clk-name>` (set with the `-clock` argument):

```
[WARNING] Output -min delay value (<value1>) set on port
'<port-name>' is greater than clock period <value2> of clock
<clk-name>
```

Potential Issues

The violation message appears if the `-min` delay value of the `set_output_delay` constraint is greater than the clock period.

Consequences of Not Fixing

Meeting timing can be difficult if the delay values are higher than the associated clock period.

How to Debug and Fix

The SDC line of `set_output_delay` and `clock` constraints are highlighted in the Console GUI.

To resolve this violation message, update the delay value or clock period.

Message 2

The following message appears when the output port `<port-name>` has a `set_output_delay` constraint set with a `-max` argument value `<value>` that is, equal to or greater than the clock period `<value2>` of the corresponding clock `<clk-name>` set with the `-clock` argument:

```
[WARNING] Output -max delay value (<value1>) set on port
'<port-name>' is greater than clock period <value2> of clock
<clk-name>
```

Potential Issues

The violation message appears if the `-max` delay value of the `set_output_delay` constraint is greater than the clock period.

Consequences of Not Fixing

Meeting timing can be difficult if the delay values are higher than the

associated clock period.

How to Debug and Fix

The SDC line of the *set_output_delay* and *clock* constraints are highlighted in the Console GUI.

To resolve this violation message, update the delay value or clock period.

Message 3

The following message appears when the output port *<port-name>* has a *set_output_delay* constraint set with a *-min* argument value *<value1>* that is greater than the corresponding *-max* argument value *<value2>*:

[WARNING] Output -min delay value (*<value1>*) is greater than -max delay (*<value2>*) on port '*<port-name>*'

Potential Issues

The violation message appears if the *-min* delay is greater than the *-max* delay.

Consequences of Not Fixing

Meeting timing can be difficult if the delay values are higher than the associated clock period.

How to Debug and Fix

The SDC line of the *set_output_delay* constraint is highlighted in the Console GUI.

To resolve this violation message, update the delay value corresponding to the *-min* and *-max* options.

Message 4

The following message appears when an output port *<port-name>* has a *set_output_delay* constraint set with a typical delay value *<value1>* that is equal to or greater than the clock period *<value2>* of the corresponding clock *<clk-name>* (set with the *-clock* argument):

[WARNING] Output delay value "*<value1>*" set on port '*<port-name>*' is greater than clock period "*<value2>*" of clock '*<clk-name>*'

Potential Issues

The violation message appears if the delay value for both *-max* and *-min*

options are greater than the clock period.

Consequences of Not Fixing

Meeting timing can be difficult if the delay values are higher than the associated clock period.

How to Debug and Fix

The SDC line of the [set_output_delay](#) constraint is highlighted in the Console GUI.

To resolve this violation message, update the delay value corresponding to the `-min` and `-max` options or update the clock period.

Example Code and/or Schematic

In this example, the delay value is greater than the clock period but is still valid:

```
create_clock -name "clk1" -period 10 [get_ports clock]
create_clock -name "clk2_io" -period 10 -waveform {0 5}
set_output_delay 11 -clock clk2_io [get_ports data_out]
```

where, `data_out` is captured by a flip-flop that is clocked by the clock `clk1`.

Since the delay value is greater than the clock period, the *Op_De102* rule reports a violation message.

Default Severity Label

Warning

Rule Group

Op_De1

Reports and Related Files

None

Op_Del03

Output constraint associated with wrong clock

The *Op_Del03* rule runs the *Op_Del03a* and *Op_Del03b* rules.

Op_Del03a

Reports output constraint associated with incorrect or incomplete set of clocks

When to Use

To validate output delays for associated clocks. This rule is applicable to the RTL, pre-layout, and post-layout phases.

Description

The *Op_Del03a* rule reports incorrect or missing [set_output_delay](#) constraints. This rule reports the following cases:

- [set_output_delay](#) constraints that are set on an output port with reference to a clock that is not a clock for the output port
- Missing [set_output_delay](#) constraints for a clock of an output port. The *Op_Del01* rule reports output ports where the [set_output_delay](#) constraint has not been set.
- A combination of the above two conditions

When [set_output_delay](#) constraint has not been set on either the create clock or generated clock, this rule reports the create clock or generated clock that is a sampling clock of the input port.

For cascading generated clocks, this rule reports the create clock or generated clock when the clocks in the cascade do not have a [set_output_delay](#) constraint set.

The *Op_Del03a* rule also reports candidate clocks for output ports.

Prerequisites

The *Op_Del03a* rule requires the synthesizable description of all library cells. Hence, use the SpyGlass Library Compiler feature before running this rule. Refer to the *SpyGlass Explorer User Guide* for details on the SpyGlass Library Compiler feature.

Rule Exceptions

The *Op_Del03a* rule does not report a wrong clock violation if that clock is strongly related to the specified clock for which the [set_output_delay](#) constraint is defined. A clock pair is considered to be strongly related if it satisfies any of the following conditions:

- One clock is generated from the other clock

- One clock of the pair is related to the third clock, and the third clock is related to the other clock of the pair. For example, if the clock `clk1` of the pair is related to a third clock `clk3` and `clk3` is related to other clock `clk2` of the pair, `clk1` and `clk2` are assumed to be strongly related.

In case of multiple clocks defined on the same object using the `-add` option of the [create_clock/create_generated_clock](#) constraint, the `Op_Del03a` rule retains only the last clock defined without the `-add` option and any clocks defined with the `-add` option after this (last) clock.

If the input port has more than one sampling clock, use the `-add_delay` option in the [set_output_delay](#) constraints for the second and subsequent clocks.

The `Op_Del03a` rule reports the same or fewer rule-violations than the `Op_Del04` rule because this rule does not consider overwritten [set_output_delay](#) constraints.

Parameter(s)

- [del03_allow_setdelay](#): Default is `no`. Set the value to `yes` to ignore those port-clock pairs for which a `set_max_delay` constraint is defined.
- [chip](#): Default is unspecified. This indicates that the `Op_Del03a` rule considers matching timing characteristics between real and virtual clocks. Therefore, the `Op_Del03a` rule does not report a missing input constraint violation for a real clock when an output constraint has been set for a matching virtual clock.
- [tc_ignore_te](#) and [ignore_io_if_fp](#) rule parameters is set to `no`, or the value of the [tc_ignore_te](#) rule parameter is set to `yes`, the `Op_Del03a` rule ignores those port-clock pairs for which the [set_false_path](#) or [set_clock_groups](#) constraint is defined.

In GuideWare2.0, the default value of [tc_ignore_te](#) is `no` and the default value of [ignore_io_if_fp](#) is `yes`.

- [tc_ignore_async_ports](#): Default is unspecified and this rule does not ignore asynchronous ports. Specify the file name that contains the list of asynchronous ports, which this rule should ignore.

- *tc_honor_virtual_clk*: Default is no. Set this parameter to yes to honor the *set_input_delay*/*set_output_delay* defined with virtual clocks that do not have any matching real clock in the SDC file.

Constraint(s)

SDC

- *create_clock* (Optional): Use to create a clock.
- *create_generated_clock* (Optional): Use to create a clock.
- *set_output_delay* (Optional): Use to define the output path delay value.
- *set_max_delay* (Optional): Use define the maximum delay for timing paths.

Messages and Suggested Fix

If the name of the port, *<port-name>*, is listed as an asynchronous port through the *tc_ignore_async_ports* parameter, the severity of each message listed in this section is INFO.

Message 1

The following message appears when *set_output_delay* constraints are set on output port *<port-name>* using clocks *<clk-name-list>* when these clocks are not the clocks for the output port:

[Op_Del 03a_01] [WARNING/INFO] Incorrect output constraint for port '*<port-name>*' - incorrect for clock(s) *<clk-name-list>*

Or,

[Op_Del 03a_02] [WARNING/INFO] Incorrect output constraint for port '*<port-name>*' - incorrect for clock(s) *<clk-name-list>*

Potential Issues

A timing path does not exist from the output port to any flip-flop sampled with the clocks. It implies that the output delay needs to be updated with the correct set of clocks.

Consequences of Not Fixing

While the output is for one clock, the delay has been specified with respect to another clock. This can result in difficulty in meeting timing. Even if timing is met, due to false path or multi-cycle path specifications, the

actual design might not be able to meet the timing.

How to Debug and Fix

A clock is classified as *incorrect* when the clock:

- specified is a virtual clock and the timing-characteristics of any of the clock are not matching with the virtual clock.
- is a real clock but it is not any sequential element that has an output pin connected to the reported port through combinational logic.

For incorrect clock no schematic is dumped because the path does not exist. Therefore, to debug this violation, you need to manually validate the timing-path from port to clock.

Message 2

The following message appears when no *set_output_delay* constraint has been set on the output port `<port-name>` referencing its clocks `<clk-name-list>`:

```
[Op_Del 03a_03] [WARNING/INFO] Missing output constraint for port '<port-name>' - missing for clock(s) <clk-name-list>
```

Potential Issues

A path exists from the port to a flip-flop sampled with clocks, but an output delay constraint is not provided. Implementation tools might assume an undesired delay value during timing analysis of the path.

Consequences of Not Fixing

While the output is generated with respect to one clock, the delay has been specified with respect to another clock. This can result in difficulty in meeting timing. Even if timing is met, due to false path or multi-cycle path specifications, the actual design might not be able to meet the timing.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic highlights the data path joining the port to the output pin of the sequential element. The schematic also shows the clock path of the sampling sequential element. Review the reported constraints in the SDC file for missing clocks.

Message 3

The following message appears when *set_output_delay* constraints on the

output port `<port-name>` referencing its generating clocks `<clk-name-list2>` are missing while `set_output_delay` constraints are set on the output port using clocks `<clk-name-list1>` that are not its generating clocks:

[WARNING] Incorrect/Missing output constraint for port '`<port-name>`' - incorrect for clock(s) `<clk-name-list1>` and missing for clock(s) `<clk-name-list2>`

Potential Issues

Some paths do not have an associated output delay constraints. It might lead to undesired delay assumptions by implementation tools during the timing analysis of paths. In addition, there are some output delay constraints that do not apply to any paths from the port. Therefore, the constraints need to be updated appropriately.

Consequences of Not Fixing

While the output is being sampled for one clock, the delay has been specified with respect to another clock. This can result in difficulty in meeting timing. Even if timing is met, due to false path or multi-cycle path specifications, the actual design might not be able to meet the timing.

How to Debug and Fix

View the incremental schematic of the violation message.

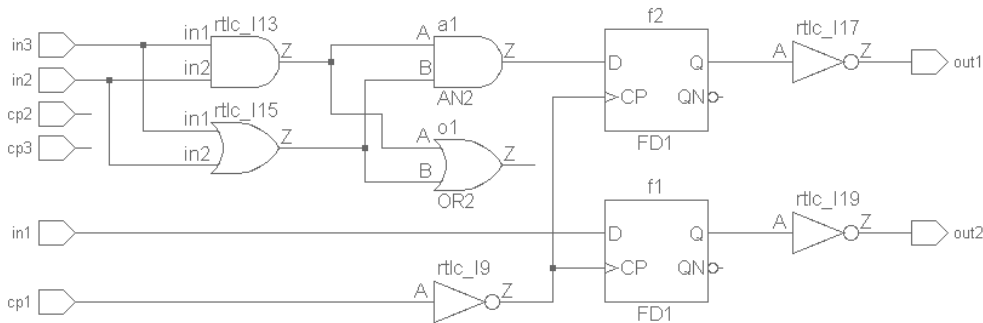
The schematic highlights the data path joining the port to the output pin of the sequential element. The schematic also shows the clock-paths of the sampling sequential element. Review the reported constraints in the SDC file for missing or incorrect clocks.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top.sdc is as follows:

```
top.sdc | top.sdc (2) |
create_clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } cp1
create_clock -name clk2 -period 20.000000 -waveform { 0.000000 10.000000 } cp2
set_output_delay -clock clk1 15.000000 out1
set_output_delay -clock clk2 25.000000 -add_delay out1
```

The violation message appears because the `set_output_delay` constraint defined on the port out1 is associated with the wrong clock clk2. In the top.sdc file, the corresponding constraint is shown.

Example 2

Test Case Files Not Available

In the following example, the `Op_Del03a` rule reports a violation for out because:

1. out should be constrained with `set_output_delay` for clk1.
2. out should not be constrained with `set_output_delay` for clk2.

--Design begin--

Input in, clk1, clk2;

Output out;

```
Flop f1( in, clk1, out );
--Design end-

--SDC begin-
create_clock -period 10 clk1
create_clock -period 12 clk2
set_output_delay 1.0 -clock clk2 out
--SDC end-
```

Example 3Test Case Files Not Available

The *Op_Del03a* rule does not consider *set_output_delay* constraints that are normally overwritten by the Synthesis Timing Analysis (STA) tools. In the following snippet, this rule considers only the second constraint specification because STA tools overwrite the first constraint specification with the second constraint specification:

```
set_output_delay <value1> 01 -clock C1
set_output_delay <value2> 01 -clock C2
```

Default Severity Label

Warning

Rule Group

Op_Del

Reports and Related Files

None

Op_Del03b

Identifies output constraints associated with wrong or incomplete set of clocks

When to Use

Use this rule for the RTL, Pre-layout and Post-layout phases.

Description

The *Op_Del03b* rule reports a violation message when:

- *set_output_delay* constraints are set on an output port with reference to a clock that is not a generating clock for the output port.
- *set_output_delay* constraints are not set for a generating clock of an output port.
- a combination of the above two conditions exist.

In addition, this rule reports candidate clocks for output ports.

In case of multiple clocks defined on the same object using the `-add` option of the *create_clock/create_generated_clock* constraint, the *Op_Del03b* rule retains only the last clock defined without the `-add` option and any clock defined with the `-add` option after this (last) clock.

When *set_output_delay* constraint has not been set on either the create clock or generated clock, this rule reports the create clock or generated clock that is a sampling clock of the input port.

For cascading generated clocks, this rule reports the create clock or generated clock when the clocks in the cascade do not have a *set_output_delay* constraint set.

The *Op_Del03b* rule will not report a wrong clock violation if that clock is strongly related to the specified clock for which the *set_output_delay* constraint is defined.

Rule Exceptions

The rule considers all *set_output_delay* constraints including those normally overwritten by the Synthesis/Timing Analysis tools.

If there no *set_output_delay* constraints defined, this rule does not report violation for missing *set_output_delay* constraint. This check is performed by *Op_Del01*.

Parameter(s)

- *tc_ignore_te* (Default is `yes`) and *ignore_io_if_fp* (Default is `no`): When the value of parameter *tc_ignore_te* is set to `no` and *ignore_io_if_fp* to `yes`, the rule will consider the effect of *set_false_path* or *set_clock_groups* constraint.

In GuideWare2.0, the default value of *tc_ignore_te* is `no` and the default value of *ignore_io_if_fp* is `yes`.

- *chip*: By default, it is `unspecified`. Set the value to a string value to specify the name of the design unit which corresponds to the chip level. When the *chip* parameter is not set, the *Op_De103b* rule considers matching timing characteristics between real and virtual clocks. Therefore, this rule does not report a missing output constraint violation for real clock when an output constraint has been set for a matching virtual clock.
- *tc_ignore_async_ports*: Default is `unspecified` and this rule does not ignore asynchronous ports. Specify the file name that contains the list of asynchronous ports, which this rule should ignore.
- *tc_honor_virtual_clk*: Default is `no`. Set this parameter to `yes` to honor the `set_input_delay/set_output_delay` defined with virtual clocks that do not have any matching real clock in the SDC file.

Constraint(s)

- *set_output_delay* (Mandatory): Use this constraint to specify the arguments for output/inout ports.
- *create_clock* (Optional): Use this constraint to specify multiple clocks using the `-add` argument.
- *create_generated_clock* (Optional): Use this constraint to specify multiple clocks using the `-add` argument.

Messages and Suggested Fix

If the name of the port, `<port-name>`, is listed as an asynchronous port through the *tc_ignore_async_ports* parameter, the severity of each message listed in this section is INFO.

Message 1

The following message appears when *set_output_delay* constraints are set on output port *<port-name>* using clocks *<clk-name-list>* when these clocks are not the generating clocks for the output port:

```
[Op_Del 03b_01] [WARNING] Incorrect output constraint for port  
'<port-name>' - incorrect for clock(s) <clk-name-list>
```

Or,

```
[Op_Del 03b_02] [INFO] Incorrect output constraint for port  
'<port-name>' - incorrect for clock(s) <clk-name-list>
```

Potential Issues

The violation message appears when a timing path does not exist from the output port to any flip-flop sampled with the clocks. It implies that the output delay needs to be updated with the correct set of clocks.

Consequences of Not Fixing

While the output is generated with respect to one clock, the delay has been specified with respect to another clock. This can result in difficulty in meeting timing. Even if timing is met, due to false path or multi-cycle path specifications, the actual design might not be able to meet the timing.

How to Debug and Fix

A clock is classified as *incorrect* when the clock:

- Specified is a virtual clock and the timing-characteristics of any of the clock are not matching with the virtual clock.
- Is a real clock but there is no sequential element that has an output pin connected to the reported port through combinational logic.

For an incorrect clock, no schematic is generated because the path does not exist. Therefore, to resolve this violation message, you need to manually validate the timing path from the port to the clock. If the timing path from clock to port does not exist, remove the incorrect *set_output_delay* constraint reported.

Message 2

The following message appears when *set_output_delay* constraint has not been set on the output port *<port-name>* referencing its generating clocks *<clk-name-list>*:

[Op_Del 03b_03] [WARNING/INFO] Missing output constraint for port '<port-name>' -missing for clock(s) <clk-name-list>

Potential Issues

The violation message appears when a path exists from the port to a flip-flop sampled with clocks, but an output delay constraint is not provided.

Consequences of Not Fixing

The output is generated with respect to one clock but the delay has not been specified with respect to that clock. This can result in difficulty in meeting timing. Even if timing is met, due to false path or multi-cycle path specifications, the actual design might not be able to meet the timing. In addition, implementation tools might assume an undesired delay value during the timing analysis of the path.

How to Debug and Fix

View the incremental schematic of the violation message. The schematic highlights the data path joining the port to the output pin of the sequential element. The schematic also shows the clock path of the sampling sequential element.

To resolve the violation message, specify the [set_output_delay](#) constraints on the reported port with respect to clocks which are reported as missing.

Message 3

The following message appears when [set_output_delay](#) constraints on the output port <portname> referencing its generating clocks <clk-name-list2> are missing while [set_output_delay](#) constraints are set on the output port using clocks <clk-namelist1> that are not its generating clocks:

[WARNING/INFO] Incorrect/Missing output constraint for port '<portname>' -incorrect for clock(s) <clk-name-list1> and missing for clock(s) <clk-name-list2>

Potential Issues

The violation message appears when some paths do not have an associated output delay constraints. In addition, there are some output delay constraints that do not apply to any paths from the port. Therefore, the constraints need to be updated appropriately.

Consequences of Not Fixing

While the output is being sampled for one clock, the delay has been specified with respect to another clock. This can result in difficulty in meeting timing. Even if timing is met, due to false path or multi-cycle path specifications, the actual design might not be able to meet the timing. In addition, it might lead to undesired delay assumptions by implementation tools during the timing analysis of paths.

How to Debug and Fix

View the incremental schematic of the violation message. The schematic highlights the data path joining the port to the output pin of the sequential element. The schematic also shows the clock-paths of the sampling sequential element.

To resolve the violation message, specify the [set_output_delay](#) constraints on the reported port with respect to clocks which are reported as missing. In addition, remove the [set_output_delay](#) constraints which are incorrect.

Example Code and/or Schematic

Example 1

In the following example, the [Op_Del03a](#) rule reports a violation for out because:

- out should be constrained with [set_output_delay](#) for clk1 .
- out should not be constrained with [set_output_delay](#) for clk2 .

```
--Design begin-
Input in, clk1, clk2;
Output out;
Flop f1( in, clk1, out );
--Design end
--SDC begincreate_
clock -period 10 clk1
create_clock -period 12 clk2
set_output_delay 1.0 -clock clk2 out
--SDC end-
```

Example 2

The *Op_Del03b* rule considers both of the following constraints even though Synthesis Timing Analysis tools will overwrite the first constraint specification with the second constraint specification:

```
set_output_delay <value1> 01 -clock C1  
set_output_delay <value2> 01 -clock C2
```

Default Severity Label

Warning

Rule Group

Op_De1

Reports and Related Files

None

Op_Del04

Reports output delay constraints that are incompletely specified

When to Use

Use this rule for the RTL, pre-layout, and post-layout phases to validate the completeness of the [set_output_delay](#) constraint.

Description

The *Op_Del04* rule reports incomplete [set_output_delay](#) constraints set on the output/inout ports. An incomplete [set_output_delay](#) constraint has at least one of the following options not defined on the same design object:

- max -rise
- max -fall
- min -rise
- min -fall

Rule Exceptions

The *Op_Del04* rule ignores the `-min` option for the worst-case SDC when the [tc_ignore_min](#) parameter is set.

Parameter(s)

- [tc_ignore_min](#): Default value is no. Set the value to yes to ignore missing `-min` option.

Constraint(s)

- [set_output_delay](#) (Mandatory): Use this constraint to specify output path delay value for the port.

Messages and Suggested Fix

The following message appears for a [set_output_delay](#) constraint on a port `<port-name>` that has not been defined with the options `<option-list>`:

[WARNING] Set_output_delay option(s) `<option-list>` not specified for port `<port-name>`

Potential Issues

This violation message appears if the design has a [set_output_delay](#) constraint on a port that has not been defined with all options.

Consequences of Not Fixing

Values for the unspecified options will get inferred by the tool and the inferred value can be different from what was desired.

How to Debug and Fix

The SDC file of constraint is highlighted in the Console GUI. To fix it, specify the missing options stated in the violation message.

Example Code and/or Schematic

Consider the following SDC snippet:

```
set_output_delay 1 -max -rise -clock C1 out
set_output_delay 1 -min -rise -clock C1 out -add_delay
```

The options (-max, -fall) and (-min, -fall) have not been specified. The rule reports a violation for the missing options.

Default Severity Label

Warning

Rule Group

Op_Del

Reports and Related Files

None

Op_Del05

Reports output delay constraints that are not defined relative to a virtual clock

When to Use

Use this rule in the pre-layout and post-layout phases.

Description

The *Op_Del05* rule reports output delay constraints specified with clocks that are not virtual clocks.

Parameter(s)

None

Constraint(s)

- *set_output_delay* (Mandatory): Use this constraint to specify the output path delay value for the port.

Messages and Suggested Fix

The following message appears when an output constraint is specified for an output port *<port-name>* with a real clock:

[WARNING] Output constraint for port '*<port-name>*' is defined with real clock '*<clock-name>*' as reference

Potential Issues

The violation message appears if the design has a *set_output_delay* constraint, which is specified for an output port with a real clock.

Consequences of Not Fixing

This is a methodology rule which can be useful in post-layout when adjusting insertion delays.

How to Debug and Fix

The SDC file is highlighted in the Console GUI. To fix it, modify the real clock specified in the reported constraint with a virtual clock.

Example Code and/or Schematic

In the following SDC snippet, the *Op_De/05* rule reports a violation because the output delay is specified with a real clock:

```
create_clock -name C1 -period 10 [get_ports clk]
create_clock -name V1 -period 10
set_output_delay 1 -clock C1 [get_ports out]
```

To resolve the violation message, specify the delay with a virtual clock V1.

Default Severity Label

Warning

Rule Group

Op_De1

Reports and Related Files

None

Op_Del07

Reports output constraints that are incorrect relative to the clock period

When to Use

Use this rule for the RTL, Pre-layout and Post-layout phases.

Description

The *Op_Del07* rule reports output ports where the output delay value is more than the user-specified percentage of the corresponding clock period. Use the *op_percent* and *op_delay_margin* parameters to set the limits.

If a *set_output_delay* constraint has the *-clock_fall* option specified, the delay is compared with the part of the clock waveform starting at the falling edge. Therefore, the delay is compared with half the clock period for default waveforms.

Effect of the Multi-cycle Constraint

The *Op_Del07* rule considers multi-cycle paths set between the output port and the clock. For example, if a *set_multicycle_path* constraint is set with a multiplier of 2, this rule uses 2x of the associated clock period for comparison.

If multiple *set_multicycle_path* constraints are specified with the *-setup* option, this rule considers the *set_multicycle_path* constraint that contains the least value of *path_multiplier*.

If the *set_multicycle_path* constraint is specified with the *-hold* option, it is ignored.

Parameter(s)

- *op_percent*: Default value is 20. Set the value to a positive float value between 0 and 100 to specify the limit of clock period.
- *op_delay_margin*: Default value is 0. Set the value to a positive float value to specify the maximum delay margin between the clock period and output delay.
- *ignore_io_if_fp* (Default is no) and *tc_ignore_te* (Default is yes): By default, the *Combo_Paths01* rule does not consider the effect of the

`set_false_path` constraint. Set `tc_ignore_te` rule to no and `ignore_io_if_fp` to yes to consider the `set_false_path` constraints.

In GuideWare2.0, the default value of `tc_ignore_te` is no and the default value of `ignore_io_if_fp` is yes.

Constraint(s)

- `set_output_delay` (Mandatory): Use this constraint to specify the arguments for output/inout ports.
- `set_false_path` (Optional): Use this constraint to specify the paths in a design which are to be marked as false, so that they are not considered during timing analysis.
- `set_multicycle_path` (Optional): Use this constraint to define the multi-cycle paths.

Messages and Suggested Fix

Message 1

The following message appears when a `set_output_delay` value `<value1>` specified for output port `<port-name>` is more than the `$op_percent` percent of the corresponding clock period `<value2>`:

[WARNING] Output delay `<value1>` for port "`<port-name>`" is more than the specified limit, `<op_percent>` of clock period `<value2>`

Potential Issues

The violation message appears when the specified `set_output_delay` constraint is not within the range.

Consequences of Not Fixing

There is difficulty in meeting timing either for this block/next block or for the top level.

How to Debug and Fix

The SDC line is highlighted in the Console GUI. Compare the delay value with the limit specified in the violation message.

To resolve this violation message, update either the delay value or the value of the `op_percent` parameter or the period of the clock.

Message 2

The following message appears when a *set_output_delay* value *<value1>* specified for output port *<port-name>* is more than the difference *<value2>* between the clock period and *\$op_delay_margin* value:

[WARNING] Output delay *<value1>* for port "*<port-name>*" is more than the specified limit, clock period minus *op_delay_margin* *<value2>*

Potential Issues

The violation message appears if the specified *set_output_delay* constraint is not within the range.

Consequences of Not Fixing

There is difficulty in meeting timing either for this block/next block or for the top level.

How to Debug and Fix

The SDC line is highlighted in the Console GUI. Compare the delay value with the limit specified in the violation message.

To resolve this violation message, update either of the delay value or the value of the *op_delay_margin* parameter or the period of the clock.

Message 3

The following message appears when a *set_output_delay* value *<value1>* specified with options *<option-list>* for output port *<port-name>* is more than *\$op_percent* percent of the corresponding clock period *<value2>*:

[WARNING] Output delay *<value1>* (specified with *<option-list>* options) for port "*<port-name>*" is more than the specified limit, *<op_percent>* of clock period *<value2>*

Potential Issues

The violation message appears if the specified *set_output_delay* constraint is not within the range.

Consequences of Not Fixing

This can result in difficulty in meeting timing either for this block/next block or for the top level.

How to Debug and Fix

The SDC line is highlighted in the Console GUI. Compare the delay value with the limit specified in the violation message.

To resolve this violation message, update either of the delay value or the value of the *op_percent* parameter or the period of the clock.

Message 4

The following message appears when a *set_output_delay* value *<value1>* specified with options *<option-list>* for output port *<port-name>* is more than the difference *<value2>* between the clock period and \$op_delay_margin value:

[WARNING] Output delay *<value1>* (specified with *<option-list>* options) for port "*<port-name>*" is more than the specified limit, clock period minus op_delay_margin *<value2>*

Potential Issues

The violation message appears if the specified *set_output_delay* constraint is not within the range.

Consequences of Not Fixing

This can result in difficulty in meeting timing either for this block/next block or for the top level.

How to Debug and Fix

The SDC line is highlighted in GUI. Compare the delay value with the limit specified in the violation message.

To resolve this violation message, update either of the delay value or the value of the *op_delay_margin* parameter or the period of the clock.

Example Code and/or Schematic

In this example, suppose the value of the *op_percent* parameter is 2. The Op_Del07 rule reports a violation message because the limit is 2, but the delay value is 4, which is greater than the limit.

```
create_clock -name C1 -period 10 [get_ports clk]
set_output_delay 4 -clock C1 [get_ports out]
```

Default Severity Label

Warning

Rule Group

Op_De1

Reports and Related Files

None

Op_Del07a

Reports output constraints that are incorrect relative to a range of clock period

When to Use

To validate the output delay values for associated clock periods. This rule is applicable to the RTL, pre-layout, and post-layout phases.

Description

The *Op_Del07a* rule reports *set_output_delay* constraints with values within/outside the specified percentage range of the corresponding clock period. This rule reports the following *set_output_delay* constraints on ports specified using the *io07_ports* rule parameter:

- The output delay value is not within the range specified by the *op_percent_min* and *op_percent_max* parameters when the *io07a_range* parameter is not set or is set to *within*.
- The output delay value is not outside the range specified by the *op_percent_min* and *op_percent_max* parameters when the *io07a_range* parameter is set to *outside*.

If a *set_output_delay* command has the *-clock_fall* option specified, the delay is compared with the part of the clock waveform starting at the falling edge. Therefore, the delay is compared with half the clock period for default waveforms.

Multi-cycle Considerations

The *Op_Del07a* rule considers multi-cycle paths set between the output port and clock. For example, if a *set_multicycle_path* command is set with a multiplier of 2, this rule uses 2x of the associated clock period for comparison. This rule considers all paths from the clock to the output port ignoring the *-through* specification, if any.

If more than one *set_multicycle_path* command is specified with the *-setup* option, this rule considers the *set_multicycle_path* command that contains the least value of *path_multiplier*.

If the *set_multicycle_path* command is specified with the *-hold* option, it will be ignored.

If the *set_multicycle_path* command is specified and the *-clock_fall* option is also used in the *set_input_delay*/*set_output_delay* constraint, then the value of the clock period will be calculated as follows:

$$(\text{path_multiplier} \times \text{clk_period}) - (0.5 \times \text{clk_period})$$

Parameter(s)

- *io07_ports*: Default is *. This indicates all ports are checked. Set this value port names to check specific ports.
- *io07a_range*: Default is within. This indicates the *Op_Del07a* rule reports input or output delay values that are not within the range specified by the *op_percent_min* and *op_percent_max* parameters. Other possible value is outside.
- *tc_ignore_te* set to no and the *ignore_io_if_fp* parameter is set to yes, the *set_false_path* constraints are taken into consideration, if specified.
In GuideWare2.0, the default value of *tc_ignore_te* is no and the default value of *ignore_io_if_fp* is yes.
- *op_percent_min*: Default is 20. This indicates that the *Op_Del07a* rule checks output delay relative to 20% of the clock period value as the minimum limit. Set the value to a positive float value between 0 and 100 to change this percentage.
- *op_percent_max*: Default is 80. This indicates that the *Op_Del07a* rule checks output delay relative to 80% of the clock period value as the maximum limit. Set the value to a positive float value between 0 and 100 to change this percentage.

Constraint(s)

SDC

- *set_input_delay* (Mandatory): Use to define the arrival time relative to a clock.
- *set_output_delay* (Optional): Use to define the output path delay value.
- *set_multicycle_path* (Optional): Use to define a multi-cycle path.
- *set_false_path* (Optional): Use to identify paths in a design that are to be marked as false, so that they are not considered during timing analysis.
Refer to the *Parameter(s)* section for details on how to consider

set_false_path constraints.

Messages and Suggested Fix

Message 1

The following message appears when a *set_output_delay* value *<value1>* specified for output port *<port-name>* is outside the range \$op_percent_min% and \$op_percent_max% of the corresponding clock period *<value2>* with the *io07a_range* parameter not set or set to within:

[WARNING] Output delay *<value1>* for port "*<port-name>*" is not within a band of \$op_percent_min% to \$op_percent_max% of clock period *<value2>*

Potential Issues

The delay value should be reset to be within the budgeted limits. Otherwise, the rest of the timing path from the port to the other timing end points has less time.

Consequences of Not Fixing

The specified delay value seems too low or too high. This could result in difficulty in meeting timing either for this block, the next block, or the top level.

How to Debug and Fix

The constraint is specified with all options and the value is not within band of range. The SDC file highlights the constraint.

Message 2

The following message appears when a *set_output_delay* value *<value1>* specified for output port *<port-name>* is within the range \$op_percent_min% and \$op_percent_max% of the corresponding clock period *<value2>* with the *io07a_range* parameter set to outside:

[WARNING] Output delay *<value1>* for port "*<port-name>*" is not outside a band of \$op_percent_min% to \$op_percent_max% of clock period *<value2>*

Potential Issues

The constraint value is inside the range, but it is outside the range.

Consequences of Not Fixing

The specified delay value seems too low or too high. This could result in difficulty in meeting timing either for this block, the next block, or the top level.

How to Debug and Fix

Constraints is specified with all options and the value is not outside band of range. The SDC file highlights the constraint.

Message 3

The following message appears when a *set_output_delay* value `<value1>` specified with options `<option-list>` for output port `<port-name>` is outside the range `$op_percent_min%` and `$op_percent_max%` of the corresponding clock period `<value2>` with the *io07a_range* parameter not set or set to within:

```
[WARNING] Output delay <value1> specified with <option-list>
options) for port "<port-name>" is not within a band of
$op_percent_min% to $op_percent_max% of clock period <value2>
```

Potential Issues

The delay value should be reset to within the budgeted limits. Otherwise, the rest of the timing path from the port to other timing end points has less time

Consequences of Not Fixing

The specified delay value seems too low or too high. This could result in difficulty in meeting timing either for this block, the next block, or the top level.

How to Debug and Fix

The constraint is specified with the min, max, rise, and fall options and the value is not within band of range. The SDC file highlights the constraint.

Message 4

The following message appears when a *set_output_delay* value `<value1>` specified with options `<option-list>` for output port `<port-name>` is within the range `$op_percent_min%` and `$op_percent_max%` of the corresponding clock period `<value2>` with the *io07a_range* parameter set to outside:

[WARNING] Output delay <value1> specified with <option-list> options) for port "<port-name>" is not outside a band of \$op_percent_min% to \$op_percent_max% of clock period <value2>

Potential Issues

The constraint value is inside the range, but should be outside the range.

Consequences of Not Fixing

The specified delay value seems too low or too high. This could result in difficulty in meeting timing either for this block, the next block, or the top level.

How to Debug and Fix

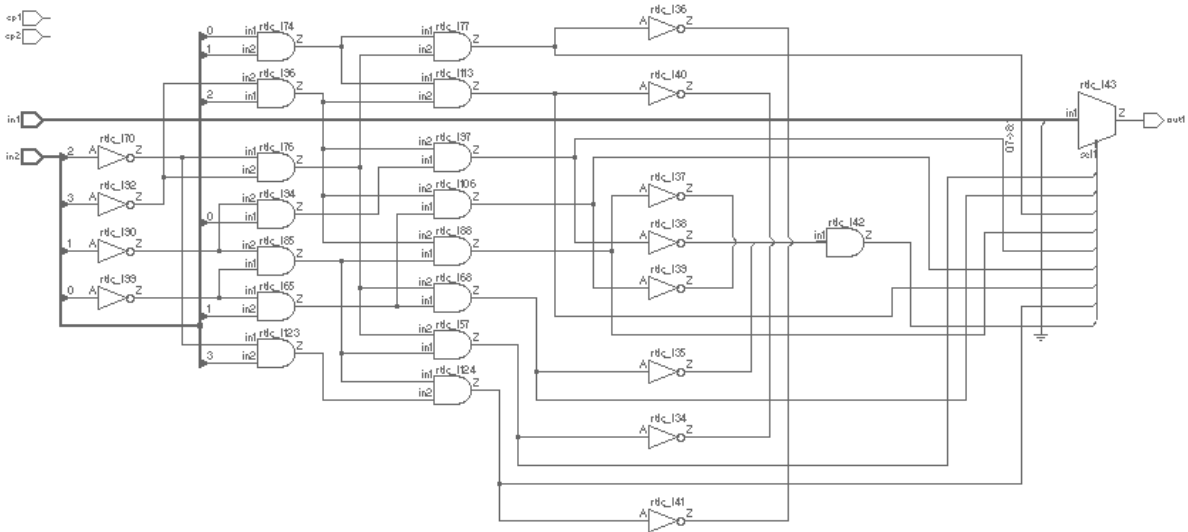
The constraint is specified with the min, max, rise, and fall options and value is not outside band of range.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top.sdc is as follows:

```
top.sdc top.sdc (2)
create_clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } {cp1}
set_output_delay 9.000000 -clock clk1 [get_ports out1*]
```

The violation message appears because the *set_output_delay* defined on the port out1 is not within the limits. In the top.sdc file, the corresponding constraint is shown.

Example 2

Test Case Files Not Available

In this example, suppose the *op_percent_min* and *op_percent_max* parameters are set to 20 and 80, respectively. Since there exists a multi-cycle path for the path from the port out to f1/CP, the clock period is

computed as:

```
multiplier value * original clock period = 20
```

Therefore, the valid range is (20% * 20) to (80% * 20) = 4 to 16.

```
--Design begin-
```

```
Input in, clk;
```

```
Output out;
```

```
Flop f1( in, clk, out );
```

```
--Design end-
```

```
--SDC begin-
```

```
clock -period 10 clk
```

```
set_output_delay $out_id -clock clk out
```

```
set_multicycle_path 2 -from [get_clocks clk] -to out
```

```
--SDC end-
```

Suppose the *io07a_range* parameter is set to *outside*, the *Op_Del07a* rule reports a violation when \$out_id is not outside the range. Otherwise, this rule reports a violation when \$out_id is not inside the range.

Default Severity Label

Warning

Rule Group

Op_Del

Reports and Related Files

None

Op_Del08

Reports `set_output_delay` when it is set on the same output relative to multiple clocks, `add_delay` is missing

When to Use

To validate that no `set_output_delay` constraints is overwritten with another. This rule is applicable to the RTL, pre-layout, and post-layout phases.

Description

The `Op_Del08` rule reports output ports with `set_output_delay` constraints set for two different clocks without the `-add_delay` argument. This rule marks the first `set_output_delay` specification and checks whether the second and subsequent specifications have the `-add_delay` argument specified. If the first specification itself contains two clocks, this rule requires that the `-add_delay` argument be specified.

Rule Exceptions

The `Op_Del08` rule ignores the input ports on which a `create_clock` or a `create_generated_clock` constraint is applied

Parameter(s)

None

Constraint(s)

SDC

- `set_output_delay` (Mandatory): Use to define the output path delay value.

Messages and Suggested Fix

The following message appears when an output port `<port-name>` has `set_output_delay` constraints set with different clocks and without the `-add_delay` argument:

```
[WARNING] Output delay constraint not set with '-add_delay' option. Paths from registers clocked by different clocks converge at port '<port-name>'
```

Potential Issues

An output delay constraint without the `-add_delay` option is provided. It overwrites the previous delay constraints on the port.

Consequences of Not Fixing

If the `-add_delay` argument is missing, the constraint will get overwritten and can result in incorrect timing.

How to Debug and Fix

Apply the `add_delay` option to the remaining [set_output_delay](#) constraints as highlighted in the SDC file.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The `top.sdc` is as follows:

```
top.sdc | top.sdc (1)
create_clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } {cp1}
create_clock -name clk2 -period 15.000000 -waveform { 0.000000 5.000000 } {cp2}
set_output_delay 3.000000 -clock {clk1} {out1[0] out1[1] out1[2] out1[3] out1[4] out1[5] out1[6] out1[7] }
set_output_delay 3.000000 -clock {clk2} {out1[0] out1[1] out1[2] out1[3] out1[4] out1[5] out1[6] out1[7] }
set_output_delay 3.000000 -clock {clk1} -add_delay [get_ports out2*]
set_output_delay 3.000000 -clock {clk2} -add_delay [get_ports out2*]
```

The violation message appears because `-add_delay` argument is not specified. In the `top.sdc` file, the corresponding constraint is shown.

Example 2

Test Case Files Not Available

The `-add_delay` argument allows you to capture information about multiple paths leading to an output port that are relative to different clocks or clock edges. In the following example, the `Op_DeI08` rule reports a violation for the [set_output_delay](#) constraint provided for `in` relative to `clk2` because `-add_delay` is missing and there exists an earlier [set_output_delay](#) constraint specified for `out`.

```
--Design begin-
Input in, clk1, clk2;
Output out;
Flop f1( in, clk1, out );
Flop f2( in, clk2, out );
--Design end-

--SDC begin-
create_clock -period 10 clk1
create_clock -period 12 clk2
set_output_delay 1.0 -clock clk1 out
set_output_delay 1.0 -clock clk2 out
--SDC end-
```

Default Severity Label

Warning

Rule Group

Op_Del

Reports and Related Files

None

Op_Del09

Reports pins on a bus that do not have the same output delay

When to Use

To verify same delay values on all pins of a bus. This rule is applicable to the RTL, Pre-layout, Post-layout phases.

Description

The *Op_Del09* rule reports bus output or inout ports where not all bits of a bus have the same *set_output_delay* constraint delay values.

If the bits of a bus have output delays specified with respect to a clock, this rule checks whether all output delays are within the acceptable margin, as specified by the *tc_inter_block_delay* parameter, of the mean bus delay and reports a violation if the delays fall outside the acceptable margin. This means the bus delay is defined as the sum of the delays defined on each pin of the bus divided by the number of the pins.

If the *tc_inter_block_delay* parameter is not set then, this rule checks that all the bits of a bus have the same output delay.

The *Op_Del09* rule checks the bits of a bus separately for each of the maximum rise, maximum fall, minimum rise, and minimum fall output delays.

This rule supports the *Source Synchronous Interfaces* feature.

Prerequisites

It is recommended that the *Op_Del09* rule be run after the *Op_Del03*, *Op_Del03*, *Op_Del07a*, *Op_Del07a*, and *Op_Del08* rules.

Parameter(s)

- *tc_inter_block_delay*: Default is 0. Set the value to a float less than 1 to specify the interconnect delay as a factor of the clock period.
- *tc_source_syn_clks*: Default is yes and all dependent rules will the *Source Synchronous Interfaces* feature. This impacts generated clocks (divide_by 1 only) applied at inout/output ports.

Constraint(s)

SDC

- [set_output_delay](#) (Mandatory): Use to define the output path delay value.

Messages and Suggested Fix

Message 1

The following message appears when the output delay for the delay-type `<delay-type-name>`, of an output/inout port bit `<port-name>`, is not in the margin range `<margin-value>` percent of the mean bus delay `<delay-value>`.

[WARNING] output delay constraint(s) `<delay-type-name>`, for port `<port-name>`, is not in the `<margin-value>%` range of mean bus delay `<delay-value>`

Where:

- `<delay-type-name>` can be `(-max, -rise)`, `(-max, -fall)`, `(-min, -rise)`, or `(-min, -fall)`
- `<margin-value>` is the value of the [tc_io_delay_bus_margin](#) parameter
- `<port-name>` is the name of the violating bit of the output/inout bus
- `<delay-value>` is the value of the mean bus delay

For debugging information, click [How to Debug and Fix](#).

Message 2

The following message appears when the output delay for the delay-type `<delay-type-name>`, of an output/inout port bit `<port-name>`, relative to clock `<clk-name>`, is not in the margin range `<margin-value>` percent of the mean bus delay `<delay-value>`.

[WARNING] output delay constraint(s) `<delay-type-name>`, for port `<port-name>`, is not in the `<margin-value>%` range of mean bus delay `<delay-value>`, relative to clock `<clk-name>`

Potential Issues

Indicative of possible typos. However, some variation is possible on account of different routing and/or if different bits are driven by different registers. For example, they may have been clocked differently.

Consequences of Not Fixing

The delay values should be near the mean delay value. Any diverging value for a bus bit might negatively impact the timing analysis for the corresponding path.

How to Debug and Fix

Mean bus delay is the average of the delay values on all the bits. The delay value for each bit should be less than the margin value, which is computed as follows:

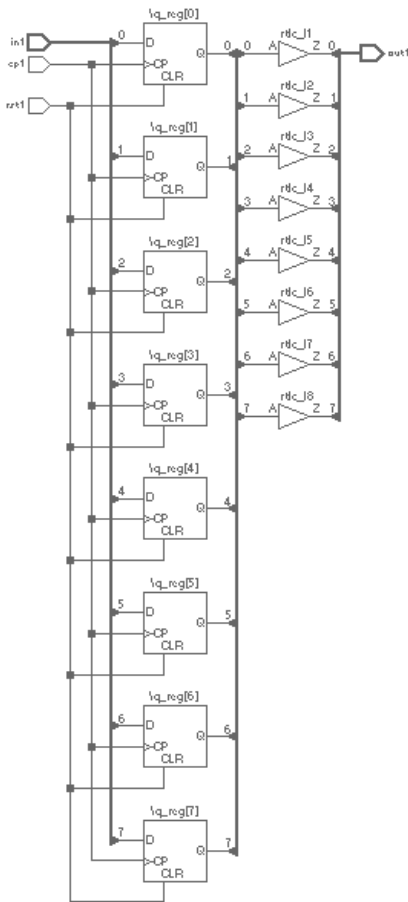
$$(\text{mean bus delay} * \text{tc_io_delay_bus_margin}) / 100$$

The constraint that has a delay value greater than the margin value is highlighted in the SDC file.

Example Code and/or Schematic

Example 1[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top.sdc is as follows:

```
top.sdc top.sdc (1)
create clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } {cp1}
set output_delay 3.000000 -clock clk1 { out1[0] out1[1] out1[2] out1[3] }
set_output_delay 0.000000 -clock clk1 { out1[4] out1[5] out1[6] out1[7] }
```

The violation message appears because the bus **out1** does not have the

same output delay. In the top.sdc file, the corresponding constraint is shown.

Example 2

 Test Case Files Not Available

The mean bus delay is the average delay for the bus. In this example, the `in` bus consists of two bits and the mean bus delay computes to 7.5. Suppose the `tc_io_delay_bus_margin` parameter is set to 10. The maximum permissible deviation of each bit's delay value from the mean bus delay value is:

$$(\text{mean bus delay} * \text{tc_io_delay_bus_margin}) / 100$$

Therefore, the maximum permissible deviation is 0.75. Since the delay values for both bits deviate more than 0.75, the `Op_De109` rule reports a violation for both of the bits.

```
--Design begin-
Input in;
Input clk;
Output [0:1] out;
Flop f1( in, clk, out[0] );
Flop f2( in, clk, out[1] );
--Design end-
--SDC begin-
clock -period 10 clk
set_output_delay 10 -clock clk out[0]
set_output_delay 5 -clock clk out[1]
--SDC end-
```

Default Severity Label

Warning

Rule Group

Op_Del

Reports and Related Files

None

Op_Del10

Reports output delay constraints that have a level sensitive argument

When to Use

Use this rule for the RTL, pre-layout, and post-layout phases.

Description

The *Op_Del10* rule reports *set_output_delay* constraints specified with the *-level_sensitive* argument.

Parameter(s)

None

Constraint(s)

- *set_output_delay* (Optional): Use this constraint to specify the output path delay value for a port.

Messages and Suggested Fix

The following message appears:

```
[INFO] Option "level_sensitive " used in command  
"set_output_delay" is not supported by all implementation tools
```

Potential Issues

This violation message appears if the design has a *set_output_delay* constraint specified with the *level_sensitive* argument in the SDC file.

Consequences of Not Fixing

The same constraints files can be interpreted differently by various tools in the flow.

How to Debug and Fix

The SDC file/line is highlighted in the GUI. Consider removing the *-level_sensitive* argument.

Example Code and/or Schematic

In the following SDC snippet, the Op_Del10 rule reports a violation because the output delay is specified with the `-level_sensitive` argument:

```
set_output_delay 1 -clock C1 -level_sensitive out
```

Default Severity Label

Info

Rule Group

Op_Del

Reports and Related Files

None

Op_Del11

Reports output delay constraints that have a `clock_fall` argument

When to Use

Use this rule for the RTL, pre-layout, and post-layout phases.

Description

The *Op_Del11* rule reports *set_output_delay* constraints specified with the `clock_fall` argument.

Parameter(s)

None

Constraint(s)

- *set_output_delay* (Optional): Use this constraint to specify the output path delay value for a port.

Messages and Suggested Fix

The following message appears:

```
[INFO] Option "clock_fall " used in command "set_output_delay" is not supported by all implementation tools
```

Potential Issues

This violation message appears when a *set_output_delay* constraint is specified with the `clock_fall` argument in the SDC file.

Consequences of Not Fixing

The same constraints files can be interpreted differently by various tools in the flow.

How to Debug and Fix

The SDC file/line is highlighted in the Console GUI. Consider removing the `clock_fall` argument.

Example Code and/or Schematic

In the following SDC snippet, the *Op_Del11* rule reports a violation

because the output delay constraint is specified with the `clock_fall` argument.

```
set_output_delay 1 -clock C1 -clock_fall out
```

Default Severity Label

Info

Rule Group

Op_Del

Reports and Related Files

None

Op_Del12

Reports output delay constraints that use the `network_latency_included` or `source_latency_included` arguments

When to Use

Use this rule for the RTL, pre-layout, and post-layout phases.

Description

The *Op_Del12* rule reports `set_output_delay` constraints specified with the `network_latency_included` and/or the `source_latency_included` argument.

Parameter(s)

None

Constraint(s)

- `set_output_delay` (Mandatory): Use this constraint to set output path delay value for a port.

Messages and Suggested Fix

The following message appears when a `set_output_delay` constraint is specified with an unsupported argument in the SDC file:

```
[INFO] Uncommon option(s) "<option-list>" used in command "set_output_delay" is not supported by all implementation tools
```

Where, the `<option-list>` is one or both of `network_latency_included` and `source_latency_included`.

Potential Issues

The violation message appears explicitly states the potential issue.

Consequences of Not Fixing

The same constraints files can be interpreted differently by various tools in the flow.

How to Debug and Fix

The SDC file/line is highlighted in the Console GUI. Consider removing the

reported option from the constraint.

Example Code and/or Schematic

In the following SDC snippet, the Op_Del12 rule reports a violation message because the output delay constraint is specified with the `source_latency_included` argument.

```
set_output_delay 1 -clock C1 out -source_latency_included
```

Default Severity Label

Info

Rule Group

Op_Del

Reports and Related Files

None

Op_Del13

Reports output delay constraints that are set to a negative value

When to Use

Use this rule for the RTL, pre-layout, and post-layout phases.

Description

The *Op_Del13* rule reports *set_output_delay* constraints set with a negative value.

Parameter(s)

None

Constraint(s)

- *set_output_delay* (Mandatory): Use this constraint to set output path delay value for a port.

Messages and Suggested Fix

The following message appears for a *set_output_delay* constraint that is set with negative value *<value>*:

```
[INFO] set_output_delay is set to negative value (<value>)
```

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

Most timing methodologies do not handle negative delays.

How to Debug and Fix

The SDC file/line is highlighted in the Console GUI. To resolve this violation message, specify a positive delay value.

Example Code and/or Schematic

In the following SDC snippet, the *Op_Del13* rule reports a violation message because the output delay has the negative value *-1*.

```
set_output_delay -1 -clock C1 out
```

Default Severity Label

Info

Rule Group

Op_De1

Reports and Related Files

None

Op_Del14

Reports a violation when `set_output_delay` is specified with wrong/incomplete set of clocks

When to Use

Use this rule to check completeness of `set_output_delay` constraint in a block. This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The `Op_Del14` rule reports `set_output_delay` constraints when:

- No `set_output_delay` constraint is specified for a port, but some clocks are sampling it.
- The `set_output_delay` constraint is specified for a port, but no clock is sampling the data.
- There is a mismatch in timing-characteristics between the clocks specified in the `set_output_delay` constraint and the clocks sampling the data.

Rule Exceptions

- The `Op_Del14` rule does not:
 - run for top-level.
 - consider `set_false_path` constraints while propagating the clocks defined at the top level. The support for these constraints will be added in a later release.

Parameter(s)

None

Constraint(s)

- `set_output_delay` (Optional): Use this constraint to specify the related clock of the port or pin on which the output delay is to be applied.

Messages and Suggested Fix

Message 1

The following message appears when a `set_output_delay` constraint for a

port *<port-name>* is specified with clock *<clk-name>*, period *<value 1>*, and waveform *<value 2>* but no clock is sampling the port:

[WARNING] set_output_delay for port *<port-name>* is specified with clock { *<clk-name>* [period: *<value 1>*, waveform { *<value 2>* }] -clock_fall } in block {*<block-name>*}, but no clock is sampling this port

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

The delay specified is redundant and therefore should be removed. This could lead to invalid timing analysis results or it may take more time to validate the redundant constraint.

How to Debug and Fix

Check whether the path from this port is reaching an end point. If it is reaching an end point, check whether the end point is constrained by any clock. If it is constrained, the violation is false and you should report it to SpyGlass Support. If it is not constrained, you need to consider why this port is required at all.

To fix this violation message, remove the [set_output_delay](#) constraint specified for this port, if required.

Message 2

The following message appears when a [set_output_delay](#) constraint for a port *<port-name>* has been specified with clock *<clk-name>*, period *<value 1>*, and waveform *<value 2>* which is not sampling this port but a [set_output_delay](#) constraint has already been specified with all the generating clocks:

[WARNING] set_output_delay for port *<port-name>* is specified with clock { *<clk-name>* [period: *<value 1>*, waveform { *<value 2>* }] -clock_fall } in block {*<block-name>*}, but set_output_delay is already specified with all the sampling clocks

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

The delay specified is redundant and therefore should be removed. This could lead to invalid timing analysis results or it may take more time to validate the redundant constraint.

How to Debug and Fix

Check whether the path from this port is reaching an end point. If it is reaching an end point, check whether the end point is constrained by any clock. If it is constrained, the violation is false and you should report it to SpyGlass Support. If it is not constrained, you need to consider why this port is required at all.

To fix this violation message, remove the [set_output_delay](#) constraint specified for this port, if required.

Message 3

The following message appears, when a [set_output_delay](#) constraint for a port `<port-name>` has not been specified with clock `<clk-name>`, period `<value 1>`, and waveform `<value 2>` for block `<block-name>`:

```
[WARNING] set_output_delay for port <port-name> should be
specified with clock { <clk-name> [period: <value 1>, waveform
{ <value 2> }] <-clock_fall> }, but no set_output_delay is
specified in block { <block-name> }
```

Potential Issues

The violation message appears when a [set_output_delay](#) constraint for a port has not been specified for a port which is being sampled by some of the clocks.

Consequences of Not Fixing

The timing path ending to this port will not be validated.

How to Debug and Fix

View the incremental schematic in the Console GUI. The schematic shows the clock sampling the port.

To resolve this violation message, specify the [set_output_delay](#) constraints for the reported clocks.

Message 4

The following message appears when a [set_output_delay](#) constraint for a

port *<port-name>* has been specified with clock *<clk-name>* for block *<block-name>* but the information is not complete:

```
[WARNING] set_output_delay for port <port-name> should be
specified with clock { <clk-name> [period: <value 1>, waveform
{ <value 2> }] <-clock_fall> }, in block { <block-name> }
```

Potential Issues

The violation message appears when the *set_output_delay* constraint is not specified with respect to all the sampling clocks.

Consequences of Not Fixing

The timing path ending to this port with respect to the reported clock will not be validated.

How to Debug and Fix

View the incremental schematic in the Console GUI. The schematic shows the clock sampling the port.

To resolve this violation message, specify the *set_output_delay* constraints for the reported clocks.

Message 5

The following message appears when a *set_output_delay* constraint for a port *<port-name>* has been specified with an incorrect clock *<clk-name>*, instead of clock *<clk2-name>*, in block *<block-name>*:

```
[WARNING] set_output_delay for port <port-name> is specified
with clock { <clk-name> [period: <value 1>, waveform { <value
2> }] <-clock_fall> }, in block { <block-name> } but it should
be specified with clock { <clk2-name> [period: <value 3>,
waveform { <value 4> }] <-clock_fall> }
```

Potential Issues

The violation message appears when the timing-characteristics between the clocks specified in the *set_output_delay* constraint and the clocks sampling the data do not match.

Consequences of Not Fixing

The timing analysis performed for this port will be done with respect to wrong clock. Therefore, the analysis will not be valid.

How to Debug and Fix

Check the timing characteristics of the reported clock. View the incremental schematic in the Console GUI.

To resolve this violation message, update the `set_output_delay` constraint reported and specify it with respect to the reported clock.

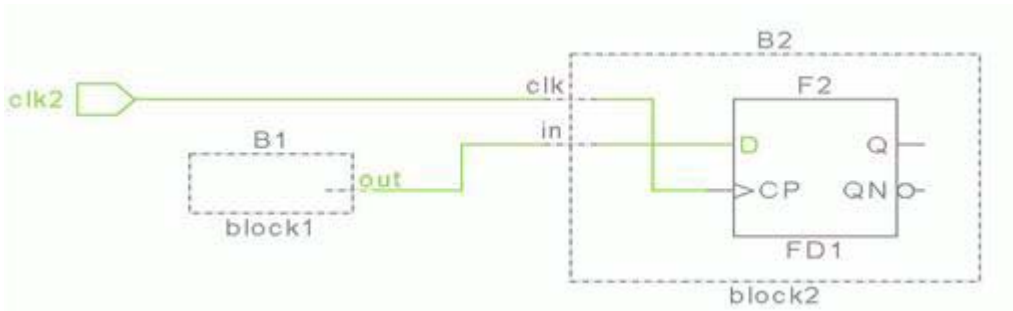
Example Code and/or Schematic

In the following SDC snippet, the period of B1 is not matching with C1. Therefore, the `Op_Del14` rule reports a violation message.

```
top.sdc:
create_clock -name C1-period 15 clk2
block.sdc:
create_clock -name B1 -period 10
set_output_delay 1 -clock B1 out
```

The following schematic is generated and it highlights the following:

- The path from the port to the flip-flop sampling the port.
- The path from the flip-flop to the port/pin of the clock.



Default Severity Label

Warning

Rule Group

Op_De1

Reports and Related Files

None

Op_Del15

Reports internal pins on which `set_output_delay` is specified

When to Use

This rule is applicable to the RTL, Pre-layout, and Postlayout phases.

Description

The *Op_Del15* rule reports a violation when `set_output_delay` constraints are specified on internal pins. The `set_output_delay` constraint should be specified on a port.

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

The following message appears:

[WARNING] Output delay specified on internal pin(s) <pin-name>

Potential Issues

Not applicable

Consequences of Not Fixing

Not applicable

How to Debug and Fix

The *Op_Del15* rule highlights the `set_output_delay` constraints that are specified on the internal pin (terminal).

To fix these violation, remove the `set_output_delay` constraint.

Example Code and/or Schematic

For the following design and SDC files, the *Op_De/14* rule reports violation messages.

The design file is as follows:

```
module top(input in1,in2,clk1,output out1,out2);
AN2 a1(.A(in1),.B(in2),.Z(n1));
FD1 f1(.D(n1),.CP(clk1),.Q(out1));
block1 b1(.in1(n1),.clk1(clk1),.out1(out2));
endmodule
```

```
module block1(input in1,clk1,output out1);
AN2 a1(.A(in1),.B(),.Z(n1));
FD1 f1(.D(n1),.CP(clk1),.Q(out1));
endmodule
```

The constraints are defined in the following SDC files:

top.sdc

```
create_clock -name C1 -period 10 [get_ports clk1]

set_output_delay 1 -clock C1 [get_pins b1/f1/Q]
set_output_delay 2 [get_ports out1]
```

block1.sdc

```
create_clock -name C1 -period 10 [get_ports clk1]
set_output_delay 1 -clock C1 [get_pins f1/Q]
set_output_delay 2 -clock C1 [get_ports out1]
```

After you run this rule, the following violations are reported:

Op_Del15 (2) : set_output_delay specified on internal pins		
⌵	⚠ Output delay specified on internal pin(s) "f1/Q"	block1.sdc
└	⚠ Output delay specified on internal pin(s) "b1/f1/Q"	top.sdc

Default Severity Label

Warning

Rule Group

Op_Del

Reports and Related Files

No report and related file

Load Rules

The Load Rules sub-group Load contains the following rules:

Rule	Description
Load01	Output or inout ports with missing or incorrect Load constraints
Load02	Runs the Load02a and Load02b rules
Load02a	Output ports/inout ports/nets that have loads outside the specified technology limits (ignoring SDC settings)
Load02b	Output ports/inout ports/nets that have loads outside the specified technology limits (considering SDC settings)
Load03	Cells other than the user-specified cells specified in the <code>load_of</code> commands
Load04	Reports the number of set_load commands when they are greater than the value specified

Load01

Identifies Load constraints on output or inout ports that are not set or set to zero

When to Use

This rule is applicable to all design phases.

Description

The *Load01* rule reports missing or incorrect load constraints for output or inout ports. The *Load01* rule reports violation messages for the following cases:

- An output or inout port without a corresponding *set_load* constraint set
- An output or inout port with the *set_load* constraint value set to zero

Parameter(s)

None

Constraint(s)

SDC

- *set_load* (Mandatory): Use to set the capacitance value on a specified object.

Messages and Suggested Fix

Message 1

The following message appears when a *set_load* constraint is not set on output or inout port *<port-name>*:

[WARNING] Load not set on Output/Inout Port '*<port-name>*'

Potential Issues

set_load is not set.

Consequences of Not Fixing

Every output port is driving certain logic. In an electrical sense, this is load. If the load value is not considered while doing timing analysis, the timing results will be different to the actual realization of the circuit. As a result,

certain paths may not meet timing.

How to Debug and Fix

The violation message states the port on which the `set_load` constraint is missing. The port is highlighted in the design.

Update the SDC file for the missing `set_load` constraint.

Message 2

The following message appears when a `set_load` constraint is set on output or inout port `<port-name>` with zero value:

[WARNING] Load set to '0' on Output/Inout Port '`<port-name>`'

Potential Issues

`set_load` is set to a zero value.

Consequences of Not Fixing

Every output port is driving certain logic. In an electrical sense, this is load. If the load value is not considered while doing timing analysis, the timing results will be different to the actual realization of the circuit. As a result, certain paths may not meet timing.

How to Debug and Fix

The violation message indicates the port on which the value of the `set_load` constraint is set to zero. The `set_load` constraint is highlighted in the SDC file.

Update the SDC file with a positive `set_load` constraint value.

Example Code and/or Schematic

In this example, the module has an output port `out`. The `Load01` rule reports a violation because the `set_load` constraint is set to 0.

```
set_load 0 [get_ports out]
```

To resolve the violation, set the `set_load` constraint to a non-zero positive value.

Default Severity Label

Warning

I/O Rules

Rule Group

Load

Reports and Related Files

None

Load02

Load values are outside technology limits

The Load02 rule runs the [Load02a](#) and [Load02b](#) rules.

Design Impact

Not defining [set_load](#) constraint can affect the results of delay calculation.

Load02a

Reports load values that are outside technology limits

When to Use

This rule is applicable to all design phases.

Description

The *Load02a* rule reports output ports, inout ports, and hierarchical nets that have load values outside the technology limit. Load values are specified through the *set_load* command.

If the port/net is driven by a netlist, the *Load02a* rule sets the maximum/minimum limit in the following order of precedence:

1. The maximum/minimum capacitance attribute set on the driver; in case of multiple drivers, maximum value of all and minimum value of all the capacitance value identified in the library
2. The default maximum capacitance attribute defined in the `.lib` technology library.
3. The values of the *default_max_capacitance* and *default_min_capacitance* parameters. If the port/net is driven by an RTL, the *Load02a* rule sets the maximum/minimum limit as per these parameters.

NOTE: The *default_min_capacitance* parameter is not supported in the `.lib` technology library.

Rule Exceptions

The *Load02a* rule does not consider SDC settings (*set_max_capacitance*/*set_min_capacitance* command); the *Load02b* rule considers these settings.

Parameter(s)

- *default_max_capacitance*: Default is 2.0. This value enables the *Load02* rule to report design objects that have a load of more than 2.0. Set this parameter to the appropriate positive float value to flag objects in your design.
- *default_min_capacitance*: Default is 0.0. The minimum load value should be 0. The value of the parameter can be customized based on your requirement.

Constraint(s)

SDC

- *set_load* (Mandatory): Use to set capacitance value on the specified object.

Messages and Suggested Fix

Message 1

The following message appears for an output port, an inout port, or a hierarchical net *<port-or-net-hier-name>* of design/block *<name>* that has maximum load value (*<num>*) and is more than the maximum value *<max>*:

[WARNING] Load value *<num>* set on *<type>* "*<port-or-net-hier-name>*" of design/block *<name>* is greater than maximum capacitance value *<max>* (taken from *<reference>*)

Where, *<type>* can be port or net and *<reference>* can be as follows:

<i><reference></i>	When the limit is picked from...
default_max_capacitance attribute of the library <i><lib-name></i>	the <i>default_max_capacitance</i> attribute of the library
rule parameter <i>default_max_capacitance</i>	the <i>default_max_capacitance</i> rule parameter
max_capacitance attribute on pin <i><pin-name></i> , (cell <i><cell-name></i>) of library <i><lib-name></i>	the <i>max_capacitance</i> attribute defined on the output pin of the cell

Potential Issues

Either the load value specified is not correct or the limit needs to be changed.

Consequences of Not Fixing

Every technology has certain support parameters, such as transistor size, PVT parameters, and capacitance limit. Similarly, each technology has a

load limit that a cell can drive. If it exceeds this limit, the chip will fail.

How to Debug and Fix

The violation message states the port or net on which the *set_load* constraint is specified. The value of the *set_load* constraint is greater than the maximum capacitance value as defined in the library or through the *default_max_capacitance* parameter. The *set_load* constraint is highlighted in the SDC file.

To resolve this violation, update either the value specified in the *set_load* constraint or the limit value as reported in the message.

Message 2

The following message appears for an output port, an inout port, or a hierarchical net *<port-or-net-hier-name>* of design/block *<name>* that has minimum load value (*<num>*) that is less than the minimum value *<min>*:

[WARNING] Load value *<num>* set on *<type>* "*<port-or-net-hier-name>*" of design/block *<name>* is less than or equal to the minimum capacitance value *<min>* (taken from *<reference>*)

Where *<type>* can be *port* or *net* and *<reference>* can be as follows:

<i><reference></i>	When the limit is picked from...
rule parameter <i>default_min_capacitance</i>	the <i>default_min_capacitance</i> rule parameter
min_capacitance attribute on pin <i><pin-name></i> , (cell <i><cell-name></i>) of library <i><lib-name></i>	the min_capacitance attribute defined on the output pin of the cell

Potential Issues

Either the value specified in *set_load* is less or the limit is too large.

Consequences of Not Fixing

Every technology has certain support parameters, such as transistor size, PVT parameters, and capacitance limit. Similarly, each technology has a load limit that a cell can drive. If it exceeds this limit, the chip will fail.

How to Debug and Fix

The violation message states the port or net on which the *set_load* constraint is specified. The value of the *set_load* constraint is less than the minimum capacitance value as defined in the library or through the *default_min_capacitance* parameter. The *set_load* constraint is highlighted in the SDC file.

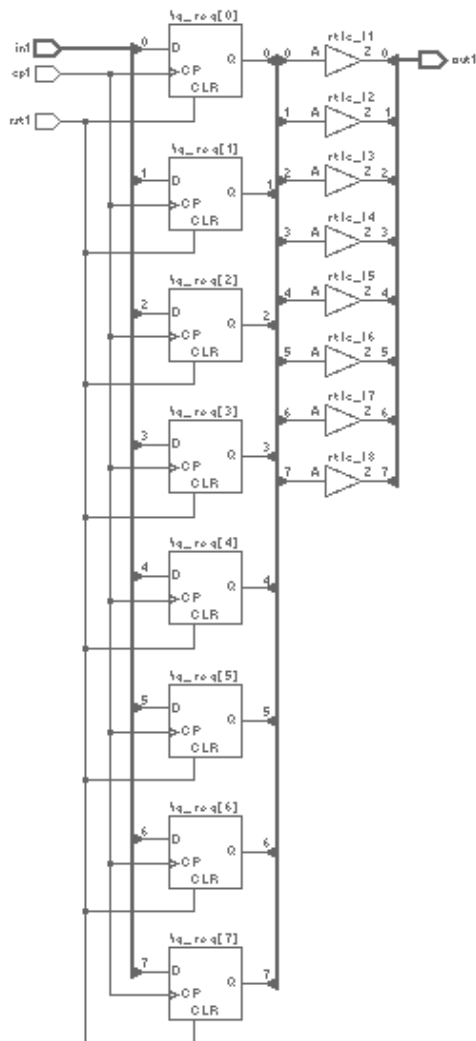
To resolve this violation, update either the value specified in the *set_load* constraint or the limit value as reported in the message.

Example Code and/or Schematic

Example 1[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:

I/O Rules



The top .sdc is as follows:

```
top.sdc | top.sdc |
set_load 3.000000 {out1[0] out1[1] out1[2] out1[3]}
set_load 0.000000 {out1[4] out1[5] out1[6] out1[7]}
```

The violation message appears because the load value is greater than maximum capacitance value defined in the project file. In the top.sdc file, the corresponding constraint is shown.

Example 2Test Case Files Not Available

Suppose the output port, *out*, of the design is being driven by F1/Q and the *default_max_capacitance* parameter defined in the library is set to 2. The *set_load* value on this parameter is 3, which is greater than the limit. In this case, the *Load02a* rule reports a violation.

Default Severity Label

Warning

Rule Group

Load

Reports and Related Files

None

Load02b

Reports load values that are outside technology limits

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *Load02b* rule reports such output ports, inout ports and hierarchical nets that have load values outside the specified technology limits. If the port/net is inferred by a netlist i.e. the port/net is connected to the output pin of a cell, then the *Load02b* rule sets the maximum/minimum limit in the following manner, whichever is defined first:

- Maximum/minimum capacitance attribute set on the driver (in case of multiple driver maximum value of all and minimum value of all).
- Default maximum capacitance attribute defined in the .lib technology library.
- Value of *set_max_capacitance*/*set_min_capacitance* set in the SDC file on the object on which *set_load* constraint is set.
- Values of the maximum and minimum capacitances set on the design.
- Values of the *default_max_capacitance* and *default_min_capacitance* parameters.

If the port/net is inferred by RTL, the *Load02b* rule sets the maximum/minimum limit as specified in 3, 4 and 5 above, whichever is defined first.

Parameter(s)

- *default_max_capacitance*: Default value is 2.0. This value enables the rule to report design objects with a load of more than 2.0. Set this parameter to the appropriate positive float value to report objects in your design.
- *default_min_capacitance*: Default value is 0.0. The minimum load value should be 0. The value of the parameter can be customized based on your requirement.

Constraint(s)

- *set_load* (Mandatory): Use this constraint to set the capacitance value on the specified object.

Messages and Suggested Fix

Message 1

The following message appears for an output port, an inout port or a *<name>* with maximum load value (*<num>*) that is more than the maximum value *<max>* specified in the corresponding `.lib` technology library, or using the *set_max_capacitance* constraint or the *default_max_capacitance* parameter:

[WARNING] Load value *<num>* set on *<type>* "*<port-or-net-hier-name>*" of design/block *<name>* is greater than maximum capacitance value *<max>* (taken from *<reference>*)

Where, *<type>* can be port or net and *<reference>* can be as follows:

<i><Reference></i>	When the reference is picked from...
<i>default_max_capacitance</i> attribute of the library <i><lib-name></i>	the <i>default_max_capacitance</i> attribute of the library.
rule parameter <i>default_max_capacitance</i>	the <i>default_max_capacitance</i> rule parameter.
<i>max_capacitance</i> attribute on pin <i><pinname></i> , (cell <i><cell-name></i>) of library <i><lib-name></i>	the <i>max_capacitance</i> attribute defined on the output pin of the cell.
command <i>set_max_capacitance</i>	the <i>set_max_capacitance</i> value specified in the SDC file.

Potential Issues

The violation message explicitly states that the load value set on port or net is more than the maximum allowed capacitance value.

Consequences of Not Fixing

Chip may not work in the current technology.

How to Debug and Fix

User should specify the value lower to the maximum specified limit.

Message 2

The following message appears for an output port, an inout port, or a net `<port-or-net-hier-name>` of design/block `<name>` that has minimum load value (`<num>`) that is less than the minimum value `<min>` specified in the corresponding `.lib` technology library, or using the `set_min_capacitance` constraint or the `default_min_capacitance` parameter:

[WARNING] Load value `<num>` set on `<type>` "`<port-or-net-hier-name>`" of design/block `<name>` is less than or equal to the minimum capacitance value `<min>` (taken from `<reference>`)

Where, `<type>` is `port` or `net` and `<reference>` is as follows:

<Reference>	When the reference is picked from...
rule parameter <code>default_min_capacitance</code>	the <code>default_min_capacitance</code> rule parameter.
<code>min_capacitance</code> attribute on pin <code><pinname></code> , (cell <code><cell-name></code>) of library <code><libname></code>	the <code>min_capacitance</code> attribute defined on the output pin of the cell.
command <code>set_min_capacitance</code>	the <code>set_min_capacitance</code> value specified in the <code>.sdc</code> file.

Potential Issues

The violation message explicitly states that the load value set on port or net is less than or equal to the minimum capacitance value.

Consequences of Not Fixing

Chip may not work in the current technology.

How to Debug and Fix

User should specify the value higher than the minimum specified limit.

Example Code and/or Schematic

Following is the code snippet for the rule *Load02b*:

```
//test.sdc  
set_load 10000 [get_ports tout]
```

Here, the rule reports a violation, because the load value set on port 'tout' of the design block is greater than the maximum capacitance value 2, taken from the [default_max_capacitance](#) rule parameter.

Default Severity Label

Warning

Rule Group

Load Rules

Reports and Related Files

None

Load03

Reports the type of load specified to user-defined requirements

When to Use

Use this rule to ensure that the rule is picking the right load value.

Description

The *Load03* rule reports cells other than the user-specified cells specified in the `load_of` constraints.

The *Load03* rule requires you to specify the list of allowed load cells using the [allowed_load_cells](#) parameter. Then, the *Load03* rule reports those `load_of` commands where a cell other than the allowed cells is used.

Parameter(s)

- [allowed_load_cells](#): Default value is `unspecified`. Use this parameter with the comma-separated name list of load cells for the [load_of](#) command.

Constraint(s)

- [set_load](#) (Mandatory): Use this constraint to set the capacitance value on a specified object.

Messages and Suggested Fix

The following message appears for `load_of` command where you specify load cells `<cell-name-list>` other than the allowed cells:

```
[WARNING] Load cells '<cell-list>' specified using command
load_of are not one of those specified using the parameter
allowed_load_cells
```

Potential Issues

The violation message explicitly states that the load cells specified using the `load_of` command are not the ones that are specified using the parameter [allowed_load_cells](#).

Consequences of Not Fixing

This is more of a checklist and methodology rule.

How to Debug and Fix

Specify the libcell name for the allowed cells with the [allowed_load_cells](#) parameter.

Example Code and/or Schematic

This example illustrates when the *Load03* rule reports a violation message. Suppose you have set the [allowed_load_cells](#) parameter to FD1PA, FD3SPA. The library contains the following cells: FD1A FD1SA FD1SPA FD2A FD1PA FD3SPA.

The *Load03* rule reports a violation because the load cells FD1A, FD1SA, FD1SPA, and FD2A are not specified using [allowed_load_cells](#) parameter.

Default Severity Label

Warning

Rule Group

Load Rules

Reports and Related Files

None

Load04

Reports the number of `set_load` commands when they are greater than the value specified

When to Use

Use this rule to ensure that the rule is picking the right load value.

Description

The *Load04* rule reports a violation if the number of objects on which load is applied through the `set_load` SDC constraint exceeds the value set by the `tc_num_load_max` parameter.

Parameter(s)

- `tc_num_load_max`: Default value is 0. Set the value to any positive integer that indicates the maximum number of `set_load` constraints, which can be provided for design objects.

Constraint(s)

- `set_load` (Mandatory): Use this constraint to set the capacitance value on a specified object.

Messages and Suggested Fix

The following message appears when the number of `set_load` constraints exceeds the value set by `tc_num_load_max`:

```
[WARNING] Number of set_load constraints present (<actual -  
value>) for design/block <name> is greater than the specified  
value <value1>
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

This is a methodology rule.

How to Debug and Fix

The instances of the `set_load` constraints are highlighted in the SDC file.

To resolve this violation, either increase the value of the `tc_num_load_max`

parameter to greater than the actual value reported in the violation message, or reduce the number of *set_load* constraints in the SDC file.

Example Code and/or Schematic

Example 1 - Illustrates when the Load04 rule reports a violation

Suppose *tc_num_load_max* is set to two. For the following snippet, the *Load04* rule reports a violation because the number of *set_load* constraints present is three.

```
...
set_load 2.5 [get_ports in1]
set_load 2.8 [get_ports in2]
set_load 3.0 [get_ports in3]
...
```

Example 2 - Illustrates when the Load04 rule does not report a violation

Suppose *tc_num_load_max* is set to one. For the following snippet, the *Load04* rule does not report a violation because the number of *set_load* constraints present is one.

```
...
set_load 2.5 -min [get_ports in1]
set_load 3.5 -max [get_ports in1]
...
```

Default Severity Label

Warning

Rule Group

Load Rules

Reports and Related Files

None

Methodology Rules

The Methodology Rule group `SDC_Methodology` contains the following rules:

Rule	Description
<i>SDC_Methodology01</i>	Constraints specified with logical pins
<i>SDC_Methodology02</i>	User-specified forbidden constraints
<i>SDC_Methodology03</i>	<i>set_logic_dc</i> , <i>set_logic_one</i> , and <i>set_logic_zero</i> constraints present in the RTL phase
<i>SDC_Methodology05a</i>	<i>set_clock_gating_check</i> constraints set on non-cell or non-pin objects with <code>-high</code> option and/or <code>-low</code> option
<i>SDC_Methodology06</i>	<i>set_clock_gating_check</i> constraints that are not explicitly set with all required options
<i>SDC_Methodology07</i>	<i>set_max_time_borrow</i> constraints where the specified delay value is less than the clock period of the corresponding latches
<i>SDC_Methodology09</i>	<i>set_port_fanout_number</i> constraints set with zero value
<i>SDC_Methodology10</i>	<i>set_logic_dc</i> , <i>set_logic_one</i> , and <i>set_logic_zero</i> constraints are present in the prelayout and postlayout phases
<i>SDC_Methodology11</i>	Output port driver pins where the maximum fan-out load exceeds the total of fan-out loads of the output port and all other driven pins
<i>SDC_Methodology12</i>	<i>set_port_fanout_number</i> constraints present in the postlayout level
<i>SDC_Methodology13</i>	<i>set_clock_gating_check</i> constraints
<i>SDC_Methodology16</i>	Incompletely specified <i>set_data_check</i> constraints
<i>SDC_Methodology18</i>	Incorrectly specified <i>set_max_transition</i> constraints
<i>SDC_Methodology21</i>	Use of drive cells other than the user-specified drive cells

Rule	Description
<i>SDC_Methodology22</i>	Mismatch in units specified in the SDC files and the library files
<i>SDC_Methodology23</i>	-library option used with specific commands
<i>SDC_Methodology24</i>	Internal nets specified in <i>set_load</i> commands
<i>SDC_Methodology25</i>	<i>set_max_capacitance</i> commands that have a value greater than the <i>max_capacitance</i> value defined in the corresponding library
<i>SDC_Methodology26</i>	<i>set_max_capacitance</i> commands that have a value greater than the <i>max_transition</i> value defined in the corresponding library
<i>SDC_Methodology27</i>	Use of <i>set_max_area</i> command
<i>SDC_Methodology28</i>	<i>set_min_capacitance</i> values that are less than a specified limit
<i>SDC_Methodology29</i>	Constraints set on the Q pin of a flip-flop
<i>SDC_Methodology30</i>	Enable or mode setting pins where a case value is not propagated
<i>SDC_Methodology31</i>	Feedback loops comprising of clock paths
<i>SDC_Methodology32</i>	Missing combination options that are not specified for the given <i>set_load</i> constraint
<i>SDC_Methodology33</i>	Missing combination options that are not specified for the given <i>set_drive</i> constraint
<i>SDC_Methodology34</i>	Missing combination options that are not specified for the given <i>set_driving_cell</i> constraint
<i>SDC_Methodology35</i>	Missing combination options that are not specified for the given <i>set_max_transition</i> constraint
<i>SDC_Methodology36</i>	Missing combination options that are not specified for the given <i>set_min_transition</i> constraint
<i>SDC_Methodology60</i>	Designs for which the <i>set_operating_conditions</i> constraint has been redefined

Methodology Rules

Rule	Description
<i>SDC_Methodology62</i>	Input/output/inout ports or a design/block without a <i>set_max_capacitance</i> constraint
<i>SDC_Methodology63</i>	Input/output/inout ports or a design/block without a <i>set_max_transition</i> constraint
<i>SDC_Methodology65</i>	Instances where multiple <i>set_case_analysis</i> settings are being propagated to different pins of the instance
<i>SDC_Methodology66</i>	Reports conflicting <i>set_case_analysis</i> on a design object
<i>SDC_Methodology67</i>	<i>set_case_analysis</i> commands applied on a flip-flop output pin or a port/pin in its combinational fan-out that conflict with propagated case analysis setting for the flip-flop data pin or a port/pin in its combinational fan-in
<i>SDC_Methodology68</i>	Unconstrained input/inout ports
<i>SDC_Methodology69</i>	Use of the <code>get_nets</code> command
<i>SDC_Methodology70</i>	Reports constraint(s) that do not have the comment option specified

SDC_Methodology01

Identifies design objects that may get optimized away

When to Use

This rule is applicable to all design phases. However, it is used more in the RTL and Pre-layout phases.

Description

The *SDC_Methodology01* rule reports constraints specified on objects that may be optimized out of the design. This rule reports the following:

- Constraints specified with design objects that are hierarchical pins, nets, or pins of combinational cells. Only port or leaf-level pins should be used because other objects might be optimized away by synthesis tools. The use of design objects that are hierarchical pins, nets, or pins of combinational cells in constraints helps in reusability of SDC. Since pin names can change between stages, such as RTL to layout, referring to points which remain invariant in a design flow improves re-usability.
- Use of the *set_dont_touch* constraint.

To disable reporting of constraints specified on hierarchical pins of library cells, set the *tc_ignore_libcells* parameter to *yes*. For more information, refer to [Example 2: Disabling Reporting of Constraints Defined on Hierarchical Pins of Library Cells](#).

To view the messages generated by this rule in CSV format, set the *tc_report_csv_messages* parameter to *SDC_Methodology01*. You can then view these messages by using Spreadsheet Viewer. Refer to [Example 2: Disabling Reporting of Constraints Defined on Hierarchical Pins of Library Cells](#).

Rule Exceptions

The *SDC_Methodology01* rule ignores flip-flop pins and blackbox pins as they are typically unaffected by synthesis. In addition, this rule does not report a violation if a constraint is applied on a pad cell/pin.

Parameter(s)

- *SDC_Methodology01_commands_list*: Default is unspecified. This indicates the *SDC_Methodology01* rule checks all supported commands.

Using this parameter, you can customize the check for specific constraints.

- *SDC_Methodology01_allow_block_pins*: Default is `no`. Set the value to `yes` to not report block-level logical pin names used in commands.
- *clk_gen_module*: Default is `none`. This indicates the *SDC_Methodology01* rule checks all design units. Set the value to the names of special design units, which should not be checked.
- *tc_ignore_libcells*: Default is `no`. This indicates that the *SDC_Methodology01* rule reports constraints specified on hierarchical pins of library cells. Set the value to `yes` to not report constraints specified on hierarchical pins of library cells.
- *tc_report_csv_messages*: Default is `unspecified`. Set this parameter to *SDC_Methodology01* to report the messages generated in CSV format. You can then view these messages by using Spreadsheet Viewer.

Constraint(s)

SDC

- *set_dont_touch* (Optional): Use to set the attribute that informs the synthesis tool to not update or optimize away that object.

Messages and Suggested Fix

Message 1

The following message appears at the location of a constraint that uses the logical pin name `<pin-name>`:

[WARNING] Use of object `<pin-name>` is not recommended. The object might get optimized away in downstream tools

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

The *SDC_Methodology01* rule ensures that constraints are not tied to any object that may be optimized away. The constraints file would need modification for later stages and lead to inconsistency between stages. There is greater chances or error creeping in.

How to Debug and Fix

The object reported is either a hierarchical pin or a combinational cell pin. This object may not exist in the downstream stages of the design.

Do not use:

- a hierarchical pin in a constraint as the design may be flattened and consequently the hierarchy is eliminated.
- a gate cell pin as the gate cell may be optimized away or re-sized.
- a net as it can also be optimized away. For example, some synthesis tools may take a gate-level netlist and upmap the logic for critical path re-synthesis. If the net is in a re-mapped area, then it is lost. Similarly, a net can also be buffered in which case it is split into two nets.

Review the SDC file as highlighted in the Console GUI. Apply the constraint on the node that should not be optimized away. If removed, that constraint will not have any effect.

Message 2

The following message appears at the location of a [set_dont_touch](#) constraint set on object `<obj-name>`:

```
[INFO] Dont_touch attribute set on object <obj-name> is not recommended. The object might get optimized away in downstream tools
```

Potential Issues

Not applicable.

Consequences of Not Fixing

Not applicable

How to Debug and Fix

The object reported is either a hierarchical pin or a combinational cell pin. This object may not exist in the downstream stages of the design. Review the SDC file as highlighted in the Console GUI.

Message 3

The following message appears when the [tc_report_csv_messages](#) parameter is set to `SDC_Methodology01`:

```
[INFO] Constraints defined on hierarchical pins of the design <design-name>. Details are reported in <file-name> file.
```

Potential Issues

Not applicable.

Consequences of Not Fixing

Not applicable

How to Debug and Fix

To view the CSV file in Spreadsheet Viewer, double-click the message.

Example Code and/or Schematic**Example 1**

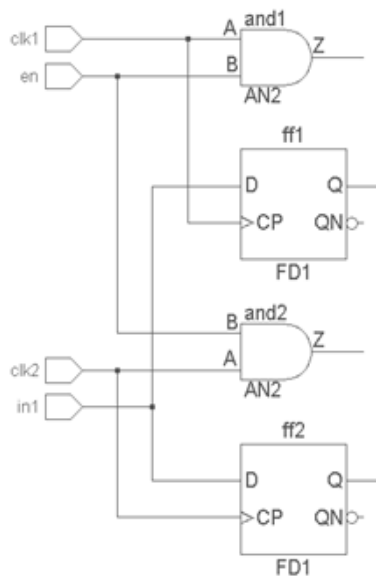
In this example, the `set_false_path` constraint has been set through an internal port.

```
set_false_path -through block1/block2/port1
```

Downstream tools may flatten the design. Therefore, the hierarchical pin will no longer exist in the design.

Example 2: Disabling Reporting of Constraints Defined on Hierarchical Pins of Library Cells

This example describes how you can disable reporting of constraints defined on hierarchical pins of library cells. It also shows you how to enable reporting of messages in CSV format. In this example, the following design is used.



The following SDC file is defined for this design.

```
create_clock [get_ports CLK1] -name ck1 -period 10.0 -
waveform {0.0 5.0}

create_generated_clock [get_pins block/clk1] -name gck1 -
source [get_ports CLK1] -divide_by 1

create_generated_clock [get_pins block/and1/Z] -name gckland
-source [get_ports CLK1] -divide_by 1

create_clock [get_pins block/clk2] -name ck2 -period 12.0 -
waveform {0.0 6.0}

create_generated_clock [get_pins block/and2/Z] -name gck2and
-source [get_pins block/clk2] -divide_by 1

set_input_delay 2.3 [get_pins block/in1]

set_false_path -from [get_pins block/and1/A]

set_max_delay 3.0 -from [get_pins block/and2/A]
```

By default, the *SDC_Methodology01* rule reports constraints defined on

Methodology Rules

hierarchical pins. Therefore, for this constraint specification the *SDC_Methodology01* rule generates the following messages:

[WARNING] Use of object block/and2/A is not recommended. The object might get optimized away in downstream tools

[WARNING] Use of object block/clk1 is not recommended. The object might get optimized away in downstream tools

[WARNING] Use of object block/and1/Z is not recommended. The object might get optimized away in downstream tools

[WARNING] Use of object block/clk2 is not recommended. The object might get optimized away in downstream tools

[WARNING] Use of object block/and2/Z is not recommended. The object might get optimized away in downstream tools

[WARNING] Use of object block/clk2 is not recommended. The object might get optimized away in downstream tools

[WARNING] Use of object block/in1 is not recommended. The object might get optimized away in downstream tools

[WARNING] Use of object block/and1/A is not recommended. The object might get optimized away in downstream tools

For ease of reading the messages, you can view them in Spreadsheet Viewer by setting the *tc_report_csv_messages* parameter to *SDC_Methodology01*.

The following image shows the results in Spreadsheet Viewer:

	B	C	D	E	F
	Constraint Type	Constraint Name	Object Name	File Name	Line Number
1	create_generated_clock	gck1	block/clk1	test.sdc	2
2	create_generated_clock	gck1and	block/and1/Z	test.sdc	3
3	create_clock	ck2	block/clk2	test.sdc	5
4	create_generated_clock	gck2and	block/and2/Z	test.sdc	6
5	create_generated_clock	gck2and	block/clk2	test.sdc	6
6	set_input_delay		block/in1	test.sdc	7
7	set_false_path		block/and1/A	test.sdc	8
8	set_max_delay		block/and2/A	test.sdc	9

To disable reporting of constraints defined on hierarchical pins of library

cells, set the [tc_ignore_libcells](#) parameter to yes. This ensures that constraints defined on and1 and and2 are not reported. The following image shows the results in Spreadsheet Viewer:

	B	C	D	E	F
	Constraint Type	Constraint Name	Object Name	File Name	Line Number
1	create_generated_clock	gck1	block/clk1	test.sdc	2
2	create_clock	ck2	block/clk2	test.sdc	5
3	create_generated_clock	gck2and	block/clk2	test.sdc	6
4	set_input_delay		block/in1	test.sdc	7

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology02

Checks for user-defined forbidden constraints

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_Methodology02* rule checks for user-defined forbidden constraints. These constraints are specified in an ASCII file. For example, to indicate *set_max_time_borrow* as a forbidden constraint write *set_max_time_borrow* in the ASCII file. Then, all uses of these constraints in the specified SDC files are checked.

Set the value of the *SDC_Methodology02_forbid_constraints* parameter to the name of the ASCII file.

Parameter(s)

- *SDC_Methodology02_forbid_constraints*: Default value is no. Set the parameter to specify an ASCII file that contains the list of forbidden constraints with its options.

Constraint(s)

None

Messages and Suggested Fix

The following message appears when a forbidden constraint *<constr>* listed in the file specified with the *SDC_Methodology02_forbid_constraints* parameter is encountered in the SDC file:

[WARNING] Forbidden constraint *<constr>* used

Potential Issues

Forbidden constraints listed in the file specified with the *SDC_Methodology02_forbid_constraints* parameter is encountered.

Consequences of Not Fixing

Forbidden constraints will not be checked. Checking of forbidden constraints improves the interoperability of SDC across tools, such as Synopsys.

How to Debug and Fix

Since you have defined this constraint as a forbidden constraint, review its use in the SDC file. If you did not intend to define this constraint as a forbidden constraint, update the ASCII file accordingly.

Example Code and/or Schematic

Refer to [SDC_Methodology02_forbid_constraints](#) for an example of an ASCII file.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology03

Checks for the presence of `set_logic_dc`, `set_logic_one` and `set_logic_zero` in the RTL phase

When to Use

Use this rule in the RTL phase.

Description

The *SDC_Methodology03* rule checks for the presence of `set_logic_dc`, `set_logic_one`, and `set_logic_zero` constraints in the RTL phase. Since these commands affect the design during compile, they must be used very carefully.

Parameter(s)

None

Constraint(s)

- `set_logic_dc`: Use to specify input port names in the current design that are driven by logic dont care.
- `set_logic_one`: Use to specify input port names in the current design that are to be driven by logic 1.
- `set_logic_zero`: Use to specify input port names in the current design that are to be driven by logic 0.

Messages and Suggested Fix

The following message appears when a forbidden constraint `<constr>` is used in the RTL phase:

```
[WARNING] <constr> constraint detected - this constraint should be used with care
```

Where, `<constr>` can be `set_logic_dc`, `set_logic_one`, and `set_logic_zero`

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

If these commands are present in the post-layout phase, they are passed

on to the backend tools for re-optimization. DC/PT can then remove logic since these commands are interpreted during compile.

Since these commands affect the design during compile, they must be used carefully.

How to Debug and Fix

The SDC file/line is highlighted in the Console GUI. Remove the constraint that is not required on the specified object.

Example Code and/or Schematic

For the following command, the *SDC_Methodology03* rule reports a violation because the XYZ input port is driven by dont care (*set_logic_dc*).

```
set_logic_dc XYZ
```

Similarly, when a port is driven by *set_logic_one* and *set_logic_zero*, this rule reports a violation.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology05a

Reports `set_clock_gating_check` when it is used with invalid `-high` or `-low` options

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_Methodology05a* rule reports `set_clock_gating_check` constraints that are set on non-cell or non-pin objects with the `-high` and/or `-low` options.

Parameter(s)

None

Constraint(s)

SDC

- `set_clock_gating_check` (Mandatory): Used to specify the value of the setup and hold time for clock gating check.

Messages and Suggested Fix

The following message appears when the `set_clock_gating_check` constraint is set with an invalid option on object `<obj-name>` in the design/block `<name>` that is not a cell or a pin:

```
[WARNING] set_clock_gating_check is specified with option
"<option>" on <obj-type> "<obj-name>" of design/block "<name>"
```

Where, `<option>` can be `-high` or `-low`.

Potential Issues

The `-high` or `-low` option cannot be specified with port and clocks.

Consequences of Not Fixing

The use of invalid options may cause the power compiler to ignore these options or the command.

How to Debug and Fix

Power compilers can ignore the high or low options of the [set_clock_gating_check](#) constraints. Review the SDC file as highlighted in the Console GUI and remove the `-high` or `-low` option.

Example Code and/or Schematic

In this example, the [set_clock_gating_check](#) constraint is set on an input port with the `-high` option.

```
set_clock_gating_check -high -setup 0.2 [get_ports in1]
```

This is not correct because the [set_clock_gating_check](#) constraint should not be specified with the `-high` or `-low` option when the object list contains clocks or ports.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology06

Reports `set_clock_gating_check` that is not specified with all required options

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_Methodology06* rule checks whether matching combinations of the `-rise` and `-fall` options and the `-setup` or `-hold` options have been specified for the *set_clock_gating_check* constraint. For example, if `-setup` and `-rise` options have been specified, the matching combination of `-setup` and `-fall` options must also be specified.

Parameter(s)

- *tc_setup_hold*: Default is both. This indicates the *SDC_Methodology06* rule checks for both `-setup` and `-hold` options. Set the value to `-setup` or `-hold` to restrict checking as mentioned in the following table.

tc_setup_hold Value	Rule Behavior
Both	Rule will check all the <code>-setup -rise</code> , <code>-setup -fall</code> , <code>-hold -rise</code> , <code>-hold -fall</code> options
Setup	Rule will check <code>-setup -rise</code> , <code>-setup -fall</code> options
Hold	Rule will check <code>-hold -rise</code> , <code>-hold -fall</code> options

Constraint(s)

SDC

- *set_clock_gating_check* (Mandatory): Use to specify the value of the setup and hold time for clock gating check.

Messages and Suggested Fix

The following message appears for a [set_clock_gating_check](#) constraint on the object `<obj-name>` of the design/block `<name>` that is not specified with all required options:

```
[WARNING] set_clock_gating_check is explicitly specified with only "<option>" for object "<obj-name>" of design/block "<name>"
```

Where, `<option>` can be `-fall`, `-rise`, `-setup`, or `-hold`.

Potential Issues

The value for all the options is not specified.

Consequences of Not Fixing

If only one of the options is specified, for the other complementary option, the Static Timing Analysis (STA) tool might assume a certain value, probably 0. This could be different from what is desired.

How to Debug and Fix

The [set_clock_gating_check](#) constraint with incomplete options is reported in the message.

Review the SDC file as highlighted in the Console GUI. Mention all options, such as `rise`, `fall`, `setup`, and `hold`, for the [set_clock_gating_check](#) constraint in the SDC file.

Example Code and/or Schematic

In this example, assume that the `tc_setup_hold` parameter is set to both and the following [set_clock_gating_check](#) constraint is specified:

```
set_clock_gating_check -setup 0.2 [get_clocks CLK]
```

The `SDC_Methodology06` rule reports a violation because the rule expects the following options be specified: `-setup -rise`, `-setup -fall`, `-hold -rise`, `-hold -fall`.

Default Severity Label

Warning

Methodology Rules

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology07

Reports `set_max_time_borrow` that is not within the limit based on the clock driving the latch

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_Methodology07* rule reports `set_max_time_borrow` constraints where the specified value is not within the limit based on the clock driving the corresponding latch. This rule ensures that the maximum time that can be borrowed cannot exceed the time for which the latch is transparent.

The limit is the time for which the latch is transparent and is equal to:

$(\text{latch closing edge time}) - (\text{latch opening edge time})$

This rule assumes negative polarity for gates like NAND, NOR, INVERTER and XOR.

The rule considers only overridden clocks.

Rule Exceptions

The SDC_Methodology07 rule:

- Does not report a violation if the `set_max_time_borrow` constraint is set on a latch and no clock is being propagated to the latch due to `set_case_analysis` settings. Such cases are reported by the *Clk_Gen01* rule.
- Handles simple waveforms and may give incorrect results for complex waveforms.

Parameter(s)

None

Constraint(s)

SDC

- `set_max_time_borrow` (Mandatory): Used to limit the time borrowing for latches.

Messages and Suggested Fix

The following message appears when the `set_max_time_borrow` constraint is set with a value `<value>` that is greater than the inferred limits for the corresponding clocks:

```
[WARNING] set_max_time_borrow value (<value>) is greater than
the limit based on clocks <clk-name-value-list>
```

Where, `<clk-name-value-list>` is the comma-separated pair list of clock names and the corresponding inferred limits.

Potential Issues

The value is greater than the limit. It should be less than the limit.

Consequences of Not Fixing

The amount of time that you can borrow is not more than the transparent pulse of the latch's gate, which in turn is less than the clock period. Therefore, a borrow time being greater than the clock period indicates a specification that is technically incorrect.

How to Debug and Fix

The limit specified in the violation message is the period for which the latch is transparent. This period is computed by taking difference between the latch closing edge and the latch opening edge.

If there is no inversion in the path between the latch and clock, the limit is the difference between the falling edge and the rising edge. If there is path inversion, the limit is the clock period minus falling edge plus rising edge.

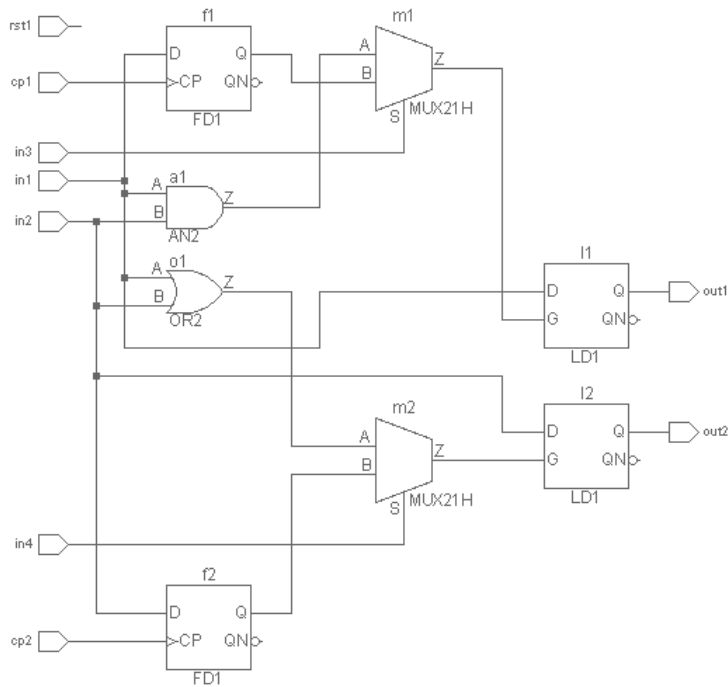
Update the value of the `set_max_borrow_time` constraint in the SDC file as highlighted in the Console GUI.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top.sdc file snippet is as follows:

```

top.sdc | top.sdc (3) | test.sgcd (1)
create_clock f1/Q -name clk1 -period 10 -waveform {0 5}
create_clock f2/Q -name clk2 -period 12 -waveform {0 5}
create_generated_clock -name gclk1 -source f1/Q -divide_by 1 f1/Q
set_max_time_borrow 11 [get_clocks clk*]

```

The violation message appears because the `set_max_time_borrow` value is greater than the limit. In the `top.sdc` file, the corresponding constraint is highlighted.

Example 2

Test Case Files Not Available

The following example illustrates how the limit is computed.

Suppose there is a clock C1 with period 10, waveform $\{0\ 4\}$.

If there is no inversion between the clock and the latch, the maximum time that can be borrowed is 4. This is the time for which the clock-edge is active.

If there is inversion between the clock and the latch, the maximum time that can be borrowed is:

$(\text{period} - \text{falling_edge of waveform}) = 6.$

This is the time for which the clock-edge is active.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology09

Reports `set_port_fanout_number` with zero value

When to Use

Use this rule in the RTL and pre-layout phases.

Description

The *SDC_Methodology09* rule checks for `set_port_fanout_number` constraints with zero value.

The `set_port_fanout_number` constraint sets the fan-out attribute on specified ports in the current design. Therefore, setting the number of external fan-out points that each port drives. You can use this information to calculate an external wire load value for the corresponding port by adding the external wire load value to the wire load of the net connected to the port.

Parameter(s)

None

Constraint(s)

- `set_port_fanout_number`: Use to specify the number of external fan-out point.

Messages and Suggested Fix

The following message appears when the `set_port_fanout_number` constraint is set with zero value:

```
[WARNING] set_port_fanout_number should not be zero
```

Potential Issues

If the `set_port_fanout_number` constraint is set to zero and the driver of the port has no other internal fan-out, then the net driving the port is treated as unconnected.

Consequences of Not Fixing

A `set_port_fanout_number` of zero is an under-constrained design because it is unlikely that there is an output port that is not supposed to

drive a net/pin.

How to Debug and Fix

Review the SDC line highlighted in the Console GUI. Validate whether the port fan-out is zero. If it is not zero, change the number as per your requirement.

Example Code and/or Schematic

For the following command, the *SDC_Methodology09* rule reports a violation because the `set_port_fanout_number` constraint is set to zero.

```
set_port_fanout_number 0 [get_ports OUT11*]
```

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology10

Reports presence of `set_logic_dc`, `set_logic_one` and `set_logic_zero` constraints in pre-layout and post-layout phases

When to Use

Use this rule in the Pre-layout and Post-layout phases.

Description

The *SDC_Methodology10* rule reports the presence of `set_logic_dc`, `set_logic_one`, and `set_logic_zero` constraints in the pre-layout and post-layout phases.

Parameter(s)

None

Constraint(s)

- `set_logic_dc`: Use to specify input port names in the current design that are driven by logic dont care.
- `set_logic_one`: Use to specify input port names in the current design that are to be driven by logic 1.
- `set_logic_zero`: Use to specify input port names in the current design that are to be driven by logic 0.

Messages and Suggested Fix

The following message appears when a forbidden constraint `<constr>` is in prelayout and postlayout phases:

[ERROR] `<constr>` constraint detected - this constraint should be used with care

Where, `<constr>` can be `set_logic_dc`, `set_logic_one` and `set_logic_zero`.

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

If these commands are present in the prelayout and postlayout phases, they are passed on to the backend tools for re-optimization. DC/PT can then remove logic since these commands are interpreted during compile.

Since these commands affect the design during compile, they must be used carefully.

How to Debug and Fix

Check the SDC file/line highlighted in the Console GUI. Validate whether the constraint should be applied on the constrained object. If not, remove the constraint.

Example Code and/or Schematic

Refer to the example shown in the *SDC_Methodology03* rule.

Default Severity Label

Error

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology11

Reports output port driver pins that have a high fan-out load

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_Methodology11* rule checks for output port driver pins where the maximum fan-out load exceeds the total fan-out loads of the output port and all other driven pins.

Parameter(s)

None

Constraint(s)

- *set_fanout_load*: Use to specify the load value in units consistent with the technology library.
- *set_max_fanout*: Use to specify maximum fan-out load in library fan-out units.

Messages and Suggested Fix

The following message appears when the pin *<pinname>* of the master design unit *<du-name>* drives an output port and the maximum fan-out load of the pin *<max>* is less than the total fan-out load *<total>* of the output port and all other pins/ports driven by the pin:

```
[WARNING] total fan-out load '<total>' for pin '<pinname>'(<du-name>)' exceeds max_fanout limit '<max>'
```

Potential Issues

The violation message appears in the Design Rule Check (DRC) when the fan-out load of this pin exceeds the limit defined by the *set_max_fanout* constraint.

Consequences of Not Fixing

The fan-out load of the master design unit pin should be less than the *max_fanout* of the driver in DRC. If the fan-out load is higher than the

`max_fanout` of the driver, it results in a DRC violation.

How to Debug and Fix

Review the SDC file/line highlighted in the Console GUI. Either increase the limit or make some changes in the netlist to decrease the fan-out count of the reported pin.

Example Code and/or Schematic

For the following snippet, the *SDC_Methodology11* rule reports a violation because the total fan-out load is set to 2.5, but the maximum fan-out load is set 1.5.

```
set_fanout_load 2.5 [get_port OUT1]
set_max_fanout 1.5 [ge_ports OUT1]
```

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology12

Reports presence of `set_port_fanout_number` in Post-layout phase

When to Use

Use this rule in the Post-layout phase.

Description

The *SDC_Methodology12* rule checks for the presence of the *set_port_fanout_number* constraint in the Post-layout phase.

Parameter(s)

None

Constraint(s)

- *set_port_fanout_number*: Use to specify the number of external fan-out points.

Messages and Suggested Fix

The following message appears when a `set_port_fanout_number` constraint is stated in the Post-layout phase in the SDC file:

[WARNING] `set_port_fanout_number` is present in postlayout

Potential Issues

The violation message appears when `set_port_fanout_number` constraint is present in the post-layout phase. The `set_port_fanout_number` constraint is used to estimate wire-loads that are external. Since it is the post-layout phase, there is no need for using estimates.

Consequences of Not Fixing

By retaining the `set_port_fanout_number` constraint, the load specified overrides the actual load values. This leads to incorrect analysis. Instead of estimates, use the actual load values.

How to Debug and Fix

Review the SDC line highlighted in the Console GUI. Remove the

constraint.

Example Code and/or Schematic

Suppose, you specify the `set_port_fanout_number` constraint during the Post-layout phase, as shown in the following snippet, the *SDC_Methodology12* rule reports a violation.

```
set_port_fanout_number 20 [get_ports OUT11*]
```

To fix the reported violation, remove the `set_port_fanout_number` constraint from the SDC file.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology13

Reports `set_clock_gating_check` in RTL, Pre-layout and Post-layout phases

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_Methodology13* rule reports the use of `set_clock_gating_check` constraint in the RTL, Pre-layout and Post-layout phases.

Parameter(s)

None

Constraint(s)

- `set_clock_gating_check`: Use to specify the value of setup and hold time for clock gating signals.

Messages and Suggested Fix

The following message appears when a `set_clock_gating_check` constraint is stated in the SDC file:

```
[WARNING] set_clock_gating_check is used
```

Potential Issues

The violation message appears when `set_clock_gating_check` constraint is used in the SDC file. The checks performed by `set_clock_gating_check` act as a constraint for the compile command.

Consequences of Not Fixing

If you do not fix this violation, certain logic transformations are disabled.

How to Debug and Fix

Review the SDC file/line highlighted in the Console GUI. Remove the `set_clock_gating_check` constraint, if not required.

Example Code and/or Schematic

For the following snippet, the *SDC_Methodology13* rule reports a violation because the `set_clock_gating_check` constraint has been used.

...

```
set_clock_gating_check -setup 0.4 -hold 0.6 [get_clocks CK11]
```

...

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology16

Reports missing options in `set_data_check`

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_Methodology16* rule reports incompletely specified [set_data_check](#) constraints.

The *SDC_Methodology16* rule requires that for a set of data paths every [set_data_check](#) constraint specified with the `setup` argument for, there must exist a corresponding [set_data_check](#) constraint specified with the `hold` argument for the exact set of data paths.

Parameter(s)

- *tc_setup_hold*: Default value is `both`. Set the value of the parameter to `setup` to check for the setup option only, `hold` to check for the hold option only, `both` to check for both setup and hold options.

Constraint(s)

- *set_data_check*: Use to specify that the data check value is for setup or hold analysis only. If neither setup nor hold is specified, both setup and hold is applied to the data check value.

Messages and Suggested Fix

The following message appears for a `set_data_check` constraint specified for a set of data paths, if there is no corresponding `set_data_check` constraint specified with the appropriate option for the exact set of data paths:

```
[WARNING] Missing option '<option>' for set_data_check constraint for this same path
```

Where, *<option>* can be `-setup` or `-hold`.

Potential Issues

This violation message appears because not all of the options for the

set_data_check constraint have been specified.

Consequences of Not Fixing

If you do not specify corresponding set_data_check constraint with appropriate option for exact set of data paths, default values are used, which might not match with the expected value.

How to Debug and Fix

Review the SDC file/line highlighted in the Console GUI.

Specify the missing option.

Example Code and/or Schematic

Consider the following example, where both the specifications must exist:

```
set_data_check -from D2 -to D1 -setup 3.5
```

```
set_data_check -from D2 -to D1 -hold 4.1
```

If you do not set the *tc_setup_hold* parameter, the *SDC_Methodology16* rule works as described in the above example. Once you set up the *tc_setup_hold* parameter to the different given options, this parameter regulates *SDC_Methodology16* rule as mentioned in the table below:

Parameter	Set to	<i>SDC_Methodology16</i> reports
<i>tc_setup_hold</i>	setup	Only when -hold option is specified without a matching setup option.
<i>tc_setup_hold</i>	hold	Only when the -setup option is specified without a matching hold option.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology18

Reports incorrectly specified `set_max_transition` constraints

When to Use

Use this rule in the RTL and Pre-layout phases.

Description

The *SDC_Methodology18* rule checks for incorrectly specified `set_max_transition` constraints. It reports `set_max_transition` constraints specified with maximum transition value greater than the allowed transition value specified through the `default_max_transition` parameter.

Parameter(s)

- `default_max_transition`: Default value is 5.0. Set the value to a positive float number that signifies the maximum transition value.

Constraint(s)

- `set_max_transition`: Use to specify the transition limit, which is the maximum transition time in library time units.

Messages and Suggested Fix

The following message appears for a `set_max_transition` constraint specified with maximum transition value `<value>` which is more than the maximum clock transition value `<max>` specified through `default_max_transition` parameter:

[WARNING] `set_max_transition` value (`<value>`) is greater than the maximum allowed transition value (`<max>`) (specified through rule parameter)

Potential Issues

The violation message appears when the transition value is more than the specified limit.

Consequences of Not Fixing

This is a Methodology related rule.

How to Debug and Fix

Review the SDC line highlighted in the Console GUI. Modify the constraint or the limit.

Example Code and/or Schematic

Consider the following example. The `default_max_transition` parameter is set to 4. The *SDC_Methodology 18* rule reports a violation because the max transition value is set to 6.

```
set_max_transition 6 [get_ports in]
```

The `set_max_transition` value should not be greater than the value of the `default_max_transition` parameter.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology21

Checks the usage of drive cells other than the user-specified drive cells

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_Methodology21* rule reports *set_driving_cell* and *get_lib_cells* constraints that have a drive cell other than the user-specified drive cells, when you specify the names of allowed drive cells using the *drive_cells_list* parameter.

Prerequisite(s)

Specify a valid list of allowed drive cells using the *drive_cells_list* parameter.

Parameter(s)

- *drive_cells_list*: Default value is unspecified. Set the value to specify the comma-separated name list of drive cells.

Constraint(s)

- *set_driving_cell*: Use to set attributes on input or inout ports to associate an external driving cell with the ports.
- *get_lib_cells*: Use to collect library cells.

Messages and Suggested Fix

The following message appears for *set_driving_cell* or *get_lib_cells* constraints with drive cells *<cell-list>* not specified in the *drive_cells_list* parameter:

```
[WARNING] Drive cells '<cell-list>' specified using command  
<constr> are not one of those specified using the parameter  
drive_cells_list
```

Potential Issues

The violation message appears when the reported object is not specified in the *drive_cells_list*.

Consequences of Not Fixing

This is a methodology related rule.

How to Debug and Fix

Either specify the reported object in the `drive_cells_list` parameter or remove the `set_driving_cell` constraint for the reported object.

Example Code and/or Schematic

Suppose you have set the `drive_cells_list` parameter as follows:

```
set_parameter drive_cells_list 'AND22'
```

In the SDC file, you have defined the `set_driving_cell` constraint as follows:

```
set_driving_cell -lib_cell AND23 {IN11}
```

The *SDC_Methodology21* rule reports a violation because the library cell AND23 has not been set in the `drive_cells_list` parameter.

If the constraint was defined as follows:

```
set_driving_cell -lib_cell AND22 {IN11}
```

In this case, the *SDC_Methodology21* rule would not report a violation.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology22

Reports mismatch of units specified in the SDC and the library files

When to Use

Use this rule in the Pre-layout and Post-layout phases.

Description

The *SDC_Methodology22* rule reports the mismatch in units specified in the SDC files and the library files. The rule checks for the units specified as comments in the SDC files and the corresponding values of the following library attributes:

- capacitive_load_unit
- current_unit
- resistance_unit
- time_unit
- voltage_unit

The *SDC_Methodology22* rule also checks the unit comments that are generated by the `write_sdc` command.

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

The following message appears when units for attribute *<attribute>* in SDC unit *<sdc-unit>* do not match with the corresponding unit *<lib-unit>* of library *<lib-name>*:

[WARNING] Units for *<attribute>* in SDC (*<sdc-unit>*) and library "*<lib-name>*" (*<lib-unit>*) do not match

Potential Issues

The violation message appears because the unit of SDC and library is not

the same.

Consequences of Not Fixing

If the number of units specified in the library files and SDC files do not match, the numbers interpreted by the tool would be different from the numbers that you have specified.

How to Debug and Fix

To cover up the mismatch, either update the units specified in the library or in the SDC file. The units of library file are updated manually. You cannot update the units through the Console GUI because it uses the encrypted version of the library files.

Example Code and/or Schematic

Consider the following example. The *SDC_Methodology22* rule reports a violation for the mismatch in value of *current_unit*.

```
Top.sdc:
```

```
#  
# Units  
# capacitive_load_unit      : 0.0001 pF  
# current_unit              : 1e-05 A
```

In this example, the unit of current is 1e-05A in the SDC file. Now, if the unit of current in the library is 1uA, *SDC_Methodology22* rule reports a violation for the mismatch in the *current_unit*.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology23

Reports the use of the `library` argument in specific constraints

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The `SDC_Methodology23` rule reports the use of the `library` argument in the following constraints:

- `set_lcd_pulse_width_multipliers`
- `create_lcd_operating_condition`
- `create_operating_conditions`
- `read_db`
- `report_lib`
- `set_driving_cell`
- `set_operating_conditions`
- `set_qtm_technology`
- `set_wire_load_model`
- `set_wire_load_selection_group`

If you specify the `library` argument in these constraints, the constraints would work particularly in one corner and might not work when you use the SDC for another corner. Though, it is not incorrect to specify the `library` option, avoid using this option in the listed constraints.

Parameter(s)

None

Constraint(s)

- `set_driving_cell`: Use to specify attributes on the specified input or inout ports to associate an external driving cell with the ports.
- `set_operating_conditions`: Use to specify operating conditions for the current design.

- *set_wire_load_model*: Use to specify wire load models.
- *set_wire_load_selection_group*: Use to specify the wire load selection group.
- *set_lcd_pulse_width_multipliers*: Use to specify LCD multipliers specific to pulse width check.
- *create_lcd_operating_condition*: Use to create LCD operating condition by using different operating conditions in the library.
- *create_operating_conditions*: Use to specify a new set of operating conditions in the library.
- *read_db*: Use to read in one or more design or library files.
- *report_lib*: Use to specify library information.
- *set_qtm_technology*: Use to specify the QTM technology library.

Messages and Suggested Fix

The following message appears when the `library` argument is used with any of the commands checked:

[WARNING] "-library" option used

Potential Issues

The violation message appears because you have used the `library` argument with any one of the following constraints:

- *set_lcd_pulse_width_multipliers*
- *create_lcd_operating_condition*
- *create_operating_conditions*
- *read_db*
- *report_lib*
- *set_driving_cell*
- *set_operating_conditions*
- *set_qtm_technology*
- *set_wire_load_model*
- *set_wire_load_selection_group*

Consequences of Not Fixing

Libraries are usually specific to each corner which is a combination of P, V, T. If the `library` argument is specified, the same library and numbers might get retrieved for all the corners. This leads to accurate analysis for only one corner and inaccurate analysis for the rest of the corners.

How to Debug and Fix

Review the impacted SDC line highlighted in the Console GUI. Remove the `library` argument, where not required.

Example Code and/or Schematic

For the following snippet, the *SDC_Methodology23* rule reports a violation because the `library` argument is specified in the first *set_wire_load_model* constraint.

```
current_design TOP
set_wire_load_model -name LION -library LIB1
current_instance u11
set_wire_load_model -name TIGER
current_instance u22
set_wire_load_model -name LEOPARD
```

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology24

Reports the use of `set_load` on nets

When to Use

Use this rule in the Post-layout phase.

Description

The *SDC_Methodology24* rule reports the use of `set_load` constraint on nets.

Parameter(s)

None

Constraint(s)

- `set_load`: Use this to set the capacitance on the specified port or net.

Messages and Suggested Fix

The following message appears for a `set_load` constraint used on the net `<net-name>`:

[WARNING] `set_load` specified for net `<net-name>`

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

In the Post-layout phase, the actual load is computed. The load specified using the `set_load` constraint on the nets overrides the actual load due on the net. This causes incorrect analysis.

How to Debug and Fix

Remove the constraint highlight in the Console GUI.

Example Code and/or Schematic

During the Post-layout phase, the following command is reported as a violation because it can cause the load value specified using the `set_load` constraint on the nets overrides the actual load due on the net.

```
set_load 4 XYZ
```

Therefore, in this case, if the actual load is 3, the analysis will use 4 as stated in the [set_load](#) constraint.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology25

Checks for set_max_capacitance

When to Use

Use this rule in the RTL, pre-layout, and post-layout phases.

Description

The *SDC_Methodology25* rule checks for *set_max_capacitance* constraints with greater value than the *max_capacitance* value defined in the corresponding library.

The following table describes how the setting of the *set_max_capacitance* constraints changes the behavior of the rule:

<i>set_max_capacitance</i> when set on	Result
inputs or inouts	Value is compared with the minimum of the <i>default_max_capacitance</i> values for all specified libraries.
output port	Value is compared with the minimum of the <i>default_max_capacitance</i> values for all specified libraries and the <i>max_capacitance</i> value of the pin driving the port.
design	Value is compared with the minimum of <i>default_max_capacitance</i> values of all specified libraries.

Parameter(s)

None

Constraint(s)

- *set_max_capacitance*: Use to specify maximum capacitance for ports, clocks or designs.

Messages and Suggested Fix

Message 1

The following message appears for a [set_max_capacitance](#) constraint set on the output port `<portname>` when the value `<value>` is greater than the maximum allowed capacitance value `<limit>`, *which is* inferred from the `max_capacitance` attribute set on driving pin `<pin-name>` of the cell `<cell-name>`:

```
[WARNING] max_capacitance (<value>) specified for port
"<portname>" is greater than the maximum allowed capacitance
(<limit>) (inferred from max_capacitance attribute of pin <pin-
name> of cell <cell-name>)
```

Potential Issues

The violation message appears when the maximum capacitance value is greater than the limit.

Consequences of Not Fixing

The chip may not work as expected with the current set of libraries because the capacitance value is not matching.

How to Debug and Fix

Review the SDC line highlighted in the Console GUI. Either update the value of the [set_max_capacitance](#) constraint or change the limit value specified by the `max_capacitance` attribute in the library.

Message 2

The following message appears for a [set_max_capacitance](#) constraint set on input/inout/output port `<portname>` when the value `<value>` is greater than the maximum allowed capacitance value `<limit>`, inferred from the [default_max_capacitance](#) attribute values of the specified libraries:

```
[WARNING] max_capacitance (<value>) specified for port
"<portname>" is greater than the maximum allowed capacitance
(<limit>) (inferred from default_max_capacitance attribute of
library)
```

Potential Issues

The violation message appears when the maximum capacitance value is greater than the limit.

Consequences of Not Fixing

The chip may not work as expected with the current set of libraries because the capacitance value is not matching.

How to Debug and Fix

Review the SDC line highlighted in the Console GUI. Either update the value of the *set_max_capacitance* constraint or change the limit value specified by the *default_max_capacitance* attribute in the library.

Message 3

The following message appears for a *set_max_capacitance* constraint set on the design *<name>* when the value *<value>* is greater than the maximum allowed capacitance value *<limit>*, inferred from the *default_max_capacitance* attribute values of the specified libraries:

```
[WARNING] max_capacitance (<value>) specified for design
"<name>" is greater than the maximum allowed capacitance
(<limit>) (inferred from default_max_capacitance attribute of
library)
```

Potential Issues

The violation message appears when the maximum capacitance value is greater than the limit.

Consequences of Not Fixing

The chip may not work as expected with the current set of libraries because the capacitance value is not matching.

How to Debug and Fix

Review the SDC line highlighted in the Console GUI. Either update the value of the *set_max_capacitance* constraint or change the limit value specified by the *default_max_capacitance* attribute in the library.

Example Code and/or Schematic

For the following snippet, the *SDC_Methodology25* rule reports a violation. Assume the *max_capacitance* attribute for F1/Q is set to 4.0.

```
//test.v
module top(in, clk, out);
    input in, clk;
```

```
output out ;

FD1 F1(.D(in), .CP(clk), .Q(out));

Endmodule

//test.sdc
set_max_capacitance 54 [get_ports {out}]
```

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology26

Reports `set_max_transition` value is outside limits

When to Use

Use this rule in the RTL, pre-layout, and post-layout phases.

Description

The *SDC_Methodology26* rule reports when the *set_max_transition* constraint has greater value than the `max_transition` value defined in the corresponding library.

The *SDC_Methodology26* rule compares the values in the following conditions:

- If the *set_max_transition* constraint is set to an input port, inout port, or design, the value is compared with the minimum of the *default_max_transition* attribute value in all libraries.
- If the *set_max_transition* constraint is set on an output port, the value is compared with minimum of the *default_max_transition* attribute value in all libraries and the `max_transition` attribute value of the pin driving the port.

Parameter(s)

- *default_max_transition*: The default value is 5.0. Use this parameter to specify the default maximum transition value. This parameter is specified in the library to decide transition values on ports/pins driving lib cell pins.

Constraint(s)

- *set_max_transition*: Use this to set a maximum transition limit on clock groups, pins, ports or designs.

Messages and Suggested Fix

Message 1

The following message appears for a *set_max_transition* constraint set on output port `<portname>` when its value `<value>` is greater than the maximum allowed transition value `<limit>`. *This value* is inferred

from the `max_transition` attribute, set on the driving pin `<pin-name>` of the cell `<cell-name>`:

[WARNING] `max_transition (<value>)` specified for port "`<portname>`" is greater than the maximum allowed transition (`<limit>`) (inferred from `max_transition` attribute of pin `<pin-name>` of cell `<cell-name>`)

Potential Issues

The violation message appears if `set_max_transition` value of the port is greater than the minimum value of `max_transition` of all libraries.

Consequences of Not Fixing

Each cell in the library is characterized according to some load and transitions and they have limits set for driving the load. If this limit is violated, the inferred timing may be inaccurate and can lead to chip failure in the current technology.

How to Debug and Fix

Review the SDC line highlighted in the Console GUI. Either update the value of the `set_max_transition` constraint or change the limit value specified by the `max_transition` attribute in the library.

Message 2

The following message appears for a `set_max_transition` constraint set on input/inout/output port `<portname>` when its value `<value>` is greater than the maximum allowed transition value `<limit>` which is inferred from the `default_max_transition` attribute values of the specified libraries:

[WARNING] `max_transition (<value>)` specified for port "`<portname>`" is greater than the maximum allowed transition (`<limit>`) (inferred from `default_max_transition` attribute of library)

Potential Issues

The violation message appears if the `set_max_transition` value on port is greater than minimum value of `default_max_transition` of all libraries.

Consequences of Not Fixing

Each cell in the library is characterized according to some load and transitions and they have limits set for driving the load. If this limit is violated, the inferred timing may be inaccurate and can lead to chip failure

in the current technology.

How to Debug and Fix

Review the SDC line highlighted in the Console GUI. Compare the max transition value specified for the given cell and the pin name in the library. Accordingly, decrease the limit to the value specified in the library by the [default_max_transition](#) parameter.

Message 3

The following message appears for a [set_max_transition](#) constraint set on design `<name>` when its value `<value>` is greater than the maximum allowed transition value `<limit>` that is inferred from the [default_max_transition](#) attribute values of the specified libraries:

```
[WARNING] max_transition (<value>) specified for design
"<name>" is greater than the maximum allowed transition
(<limit>) (inferred from default_max_transition attribute of
library)
```

Potential Issues

The violation message appears if the max transition value specified for the design is greater than the value specified by the library parameter.

Consequences of Not Fixing

The chip may not work properly in the current technology.

How to Debug and Fix

Review the SDC line highlighted in the Console GUI. Decrease the max transition value accordingly.

Example Code and/or Schematic

In the following example, the value of [set_max_transition](#) is defined in the SDC file as 5.0 for input port "in", which is greater than [default_max_transition](#) value 0.5 defined in the library file. Since the value defined in the SDC file and the library files do not match, the *SDC_Methodolgy26* rule reports a violation.

```
//test.sdc
```

```
Set_max_transition 5.0 [get_ports in]
```

```
//mylib.lib
Library (mylib) {
  Default_max_transition : 0.5;

  Cell () {
    . .
  }
}
```

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology27

Reports the use of `set_max_area`

When to Use

Use this rule in the RTL phase.

Description

The *SDC_methodology27* rule reports when you specify the target area of the current design using the *set_max_area* constraint in the SDC file.

Parameter(s)

None

Constraint(s)

- *set_max_area* (Mandatory): Use this to specify the `max_area` attribute value. This value must be greater than or equal to zero and the units used must be same as in the technology library during optimization.

Messages and Suggested Fix

The following message appears for a *set_max_area* constraint encountered in the SDC file:

```
[INFO] set_max_area constraint has been set
```

Potential Issues

The message appears to inform the use of *set_max_area* constraint in the SDC file. This is an informational rule.

Consequences of Not Fixing

This is an informational rule that indicates the use of `max_area` constraint in the SDC file. There are no such consequences that might affect the design.

How to Debug and Fix

Not applicable

Example Code and/or Schematic

For the following snippet, the *SDC_Methodology27* rule reports an informational message which states that the *set_max_area* constraint has been set.

```
//test.sdc  
Set_max_area 250.0
```

Default Severity Label

Info

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology28

Reports `set_min_capacitance` value when less than the limit

When to Use

Use this rule in the RTL phase.

Description

The *SDC_Methodology28* rule checks if the value set with the `set_min_capacitance` in the SDC file is less than a specified limit.

For an input/inout port, the value of the `min_capacitance` is defined in the library for the pin associated with `set_driving_cell`. This value is the minimum possible value of the total capacitance for the input/inout port. If you set a lower value using `set_min_capacitance`, this rule reports a violation.

Parameter(s)

None

Constraint(s)

- `set_min_capacitance` (Mandatory): Use this constraint to specify the minimum total capacitance on input or inout ports or designs.
- `set_driving_cell` (Mandatory): Use this constraint to specify the library cell to drive the port(s).

Messages and Suggested Fix

The following message appears for the input/inout port `<portname>` when the value `<value1>` specified with `set_min_capacitance` is less than the value `<value2>` of the `min_capacitance` attribute defined in the library `<lib-name>` for the driving pin of the cell `<cell-name>` associated with `set_driving_cell`:

```
[WARNING] min_capacitance (<value1>) specified for port
"<portname>" is less than the minimum capacitance (<value2>)
inferred from min_capacitance attribute in library <lib-name>
for driving cell <cell-name>
```

Potential Issues

The violation message appears when the `min_capacitance` value specified for port using `set_min_capacitance` is less than the minimum capacitance value defined in the library for driving cell.

Consequences of Not Fixing

The technology mode might not support the value because the specified value is lower than that supported by the methodology.

How to Debug and Fix

Specify a value greater than or equal to that stated by the minimum capacitance value defined for the driving cell referred from `set_driving_cell` in the library.

Example Code and/or Schematic

In the following example, the drive capability of the input port A is dictated by the library cell AND. Suppose, the `min_capacitance` value defined in the library is 2.0. In the SDC file, the `set_min_capacitance` value is 1.0, as shown in the following snippet.

```
//test.sdc
set_driving_cell -lib_cell AND [get_ports A]
set_min_capacitance 1.0 A

//mylib.lib
Library (mylib) {
...
Cell (Mycell) {
..
Min_capacitance : 2.0;
};
};
```

Since the minimum value in the library file is more than the value specified with `set_min_capacitance`, the `SDC_Methodology28` rule reports a violation.

Methodology Rules

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology29

Reports constraints set on flip-flop Q pin

When to Use

Use this rule in the RTL phase.

Description

The *SDC_Methodology29* rule generates the *tc_QPinConstraints* report that lists all constraints set on the Q pin of all flip-flops in the design. However, this rule reports the leaf-level pins only.

For each flip-flop Q pin, the report stacks the following information in a tabular format:

- The pin name
- The source HDL file name and line number for the pin
- The SDC file name and line number for each constraint set on the pin

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

The following message displays the generation of the report *<file-name>* for constraints set on flip-flop Q pin:

```
[INFO] Report ' <file-name>' generated for constraints set on  
flip-flop Q pin
```

Potential Issues

The message appears to display the file name that is generated for constraints set on flip-flop Q pin.

Consequences of Not Fixing

This is an informational rule that indicates the generation of a report which contains the list of all constraints set of Q pin of the flip-flops.

How to Debug and Fix

Double-click the message to view the report in the Console GUI.

Example Code and/or Schematic

In this example, FF/Q is a leaf cell pin. The *SDC_Methodology29* rule reports a message stating that the tc_QPinConstraints report is generated.

```
//test.sdc  
create_generated_clock -name gcl -source -divide_by 2 FF/Q
```

Default Severity Label

Info

Rule Group

SDC_Methodology

Reports and Related Files

- *tc_QPinConstraints*: It contains the list of all constraints set on the Q pin of all flip-flops in the design.

SDC_Methodology30

Reports missing `set_case_analysis` on a port

When to Use

Use this rule in the RTL, pre-layout, and post-layout phases.

Description

The *SDC_Methodology30* rule checks for the ports on which the [set_case_analysis](#) is missing.

When a mux lies on a clock path and drives the inputs, the constant value that reaches the select pin determines the selection of the clock. The *SDC_Methodology30* reports a violation when no constant value reaches the select pin so as to select the clock.

Many cells have `test_scan_enable` pins that are used for scan-shift. The constant value that reaches this pin determines the use of scan-in data. The *SDC_Methodology30* rule reports a violation when no constant value reaches the `test_scan_enable` pin.

The *SDC_Methodology30* rule ignores the scan enable pins when the [tc_at_speed_testing_mode](#) parameter is set to `yes`. This rule also checks the ports in the fan-in of the mux select-pin or `test_scan_enable` pin on which no constant value is applied.

Rule Exception(s)

The *SDC_Methodology30* rule does not report a violation if during traversal, all the fan-ins of clock pin do not terminate at stop points and are assumed to be dangling. The stop points for the traversal are ports, black boxes, other sequential cells, tristate enable pins, and blocked pins.

Parameter(s)

- [tc_at_speed_testing_mode](#): Default value is `no`. It specifies that the design runs in the functional mode. Set the value to `yes` to specify that the design runs in the at-speed test mode.

Constraint(s)

- *set_case_analysis* (Mandatory): Use this constraint to specify the list of ports or pins on which the specified constant logic value or transition is assigned.

Messages and Suggested Fix

The following message appears if a case analysis value is missing on the port `<port-name>` of the design/block `<name>`:

[INFO] For design/block `<name>`, `set_case_analysis` is probably missing on `<port-name>`

Potential Issues

The message appears if a case analysis value is missing on the port of a design/block.

Consequences of Not Fixing

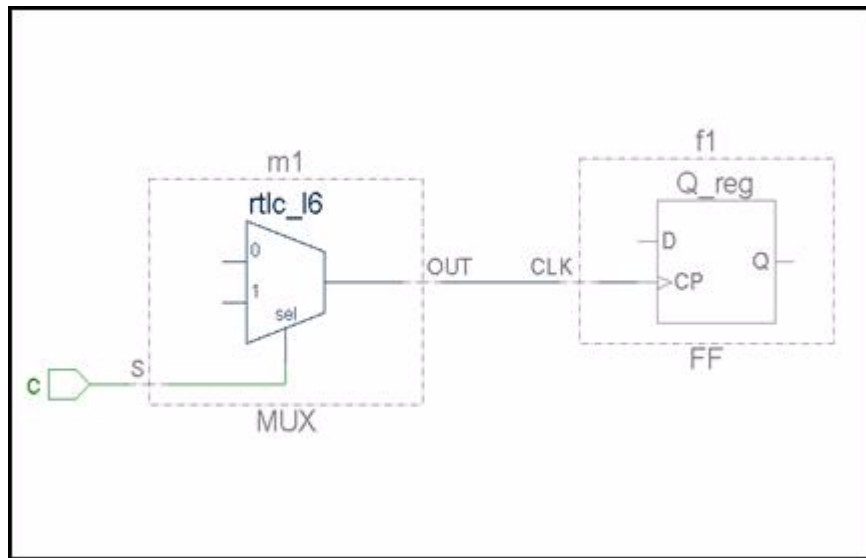
This is an informational rule that reports ports on which a *set_case_analysis* is missing. There are no such consequences that might affect the design.

How to Debug and Fix

View the schematic generated by the rule. The schematic shows the path from the port to point where *set_case_analysis* should be effective. Apply the *set_case_analysis* on the port, if required.

Example Code and/or Schematic

In following example, the *set_case_analysis* constraint is missing at port c. Therefore, the *SDC_Methodology30* rule reports a violation on that port.



Default Severity Label

Info

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology31

Reports feedback loops comprising of clock paths

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_Methodology31* rule reports feedback loops comprising of clock paths. A feedback loop comprising of clock paths exists in a design, if the output of a sequential element drives the clock pin of the same sequential element.

The path from the output pin to the clock pin comprises:

- Combinational logic cells.
- Timing path from the clock pin to the output pin of a sequential cell.

To identify the existence of a feedback loop, the *SDC_Methodology31* rule performs a fan-in traversal from the design object on which a clock is defined to the point beyond which the traversal does not proceed, such as:

- Beyond a pin being blocked due to *set_case_analysis* settings or other constant values.
- From an output pin to an input pin of an instance if no timing arc exists between the two pins.
- Beyond the enable pin of a tristate.

Parameter(s)

None

Constraint(s)

- *set_case_analysis* (Optional): Use this constraint to specify the list of ports or pins on which the specified constant value or transition is assigned.

Messages and Suggested Fix

The following message appears if a feedback loop comprising of clocks *<clock-names>* is detected for the design/block *<name>*, where

`<clock-names>` is a string of clocks applied on the design objects lying on the feedback path:

[WARNING] Feedback loop exists comprising of clocks `<clock-names>` for design/block `<name>`

Potential Issues

The violation message appears if a feedback loop comprising of clocks is detected for the design.

Consequences of Not Fixing

If you do not fix this violation, the path from the output of the last sequential element to input of the clock pin of the first sequential element sensitizes. As a result, the output of all the sequential elements on the feedback loop does not change as long as that path remains sensitized and the circuit is in a dead state.

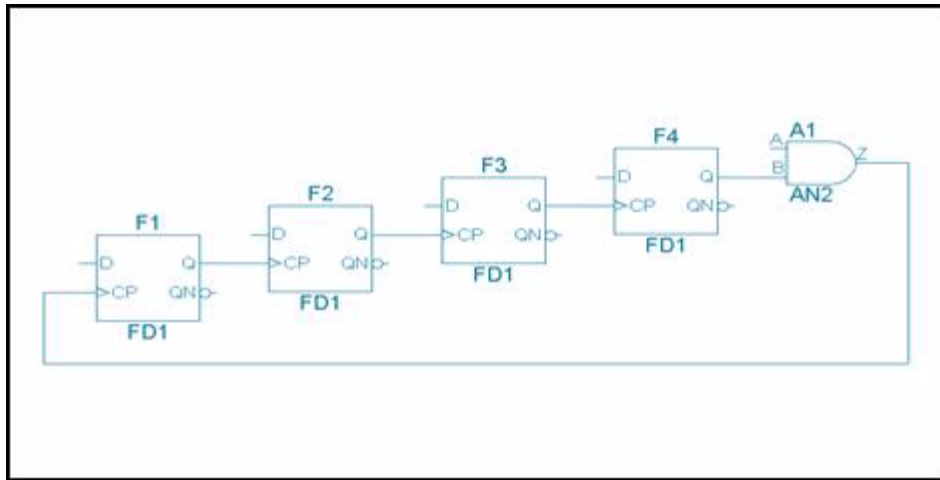
How to Debug and Fix

To debug the violation, break the loop and define the clocks again. In addition, view the [SDC_Methodology31 Report](#) for details of the clock paths comprising the feedback path.

Example Code and/or Schematic

Consider the following diagram as an example where the `SDC_Methodology31` rule reports a violation because a feedback loop FD1 exists in the clock path.

When the clock is applied on pin F1/Q, the clock is comes back to clock-pin of F1 through F2, F3, F4, and A1. Therefore, forming a feedback loop FD1.



Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

- [SDC_Methodology31 Report](#): It contains details of the clock paths comprising the feedback path.

SDC_Methodology32

Reports missing combination options in `set_load`

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_Methodology32* rule checks and reports the missing combination options that are not specified for the *set_load* constraint. The various combinations of options checked by this rule are:

- `-min -rise`
- `-max -rise`
- `-min -fall`
- `-max -fall`

Rule Exceptions

Only `min` and `max` options are checked for nets.

Parameter(s)

None

Constraint(s)

- *set_load* (Mandatory): Use this constraint to specify the minimum or maximum capacitance.

Messages and Suggested Fix

The following message appears if a combination of options `<options>` is not specified in the *set_load* constraint for the `<design-object-specification>`:

[WARNING] Option(s) `<options>` not specified for `set_load`: `<design-object-specification>`

Where,

- `<options>` are comma-separated string of missing options.

- *<design-object-specification>* represents the combination of type of the design objects such as net, port etc and the hierarchical name of the design object.

For example, if the design object type is "pin" and the design object is hier_name, the design object specification is "pin_hier_name".

Potential Issues

The violation message appears if all options are not specified in the [set_load](#) constraint for a design.

Consequences of Not Fixing

If you do not fix this violation, the timing analysis tool may assume default value on the options that are not specified. As a result, timing could go wrong leading to delay in timing closure or even wrong analysis.

How to Debug and Fix

To debug this violation, specify all the missing options as stated in the violation message.

Example Code and/or Schematic

Example

Consider the following example. The value is specified for the combinations -max -rise and -max -fall. However, the value is not specified for the combinations -min -rise and -min -fall. This means that the [set_load](#) constraint is incompletely specified for the port p.

```
set_load value -max [get_ports p]
```

Therefore, the *SDC_Methodology32* rule reports a single violation for the combinations -min -rise and -min -fall.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology33

Reports the missing combination options in `set_drive`

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_Methodology33* rule checks and reports the missing combination options that are not specified for the given *set_drive* constraint. The various combinations of options checked by this rule are:

- `-min -rise`
- `-max -rise`
- `-min -fall`
- `-max -fall`

Parameter(s)

None

Constraint(s)

- *set_drive* (Mandatory): Use this to set the resistance value on the specified input or inout port.

Messages and Suggested Fix

The following message appears if a combination of options `<options>` is not specified in the *set_drive* constraint for the `<design-object-specification>`:

[WARNING] Option(s) `<options>` not specified for `set_drive: <design-object-specification>`

Where,

- `<options>` is a comma-separated string of missing options.
- `<design-object-specification>` represents the combination of type of design object, such as pin, net, port etc., and the hierarchical name of the design object.

Potential Issues

The violation message appears if all the options are not specified in the `set_drive` constraint for a design.

Consequences of Not Fixing

If you do not fix this violation, the timing analysis tool may assume default value on the options that are not specified. As a result, timing could go wrong leading to delay in timing closure or even wrong analysis.

How to Debug and Fix

To debug this violation, specify all the missing options as stated in the violation message.

Example Code and/or Schematic

In this example, the `set_drive` value is applied on port p as:

```
set_drive resistance -max [get_ports p]
```

Resistance is specified for the combinations `-max -rise` and `-max -fall`. However, the resistance is not specified for the combinations `-min -rise` and `-min -fall`. This means that the `set_drive` constraint is incompletely specified for port p. Therefore, the `SDC_Methodology33` rule reports a single violation for the combinations `-min -rise` and `-min -fall`.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology34

Reports missing combination options in `set_driving_cell`

When to Use

Use this rule in the RTL, pre-layout, and post-layout phases.

Description

The *SDC_Methodology34* rule lists the missing combination options that are not specified for the given *set_driving_cell* constraint. The various combinations of options checked by this rule are:

- `-min -rise`
- `-max -rise`
- `-min -fall`
- `-max -fall`

Parameter(s)

None

Constraint(s)

- *set_driving_cell* (Mandatory): Use this constraint to specify the library cell name used to drive the port.

Messages and Suggested Fix

The following message appears if a combination of options *<options>* is not specified in the *set_driving_cell* constraint for the *<design-object-specification>*:

[WARNING] Option(s) *<options>* not specified for `set_driving_cell`: *<design-object-specification>*

Where,

- *<options>* is a comma-separated string of missing options.
- *<design-object-specification>* represents the combination of type of design object such as net, port, pins and the hierarchical name of the design object.

Potential Issues

The violation message appears if a combination of options is not specified in the [set_driving_cell](#) constraint for the design specification object.

Consequences of Not Fixing

If you do not fix this violation, the timing analysis tool may assume default value on the options that are not specified. As a result, timing could go wrong leading to delay in timing closure or even wrong analysis.

How to Debug and Fix

To debug this violation, specify all the missing options as stated in the violation message.

Example Code and/or Schematic

In the following example, cell_name drives port p, the corresponding [set_driving_cell](#) constraint is specified as:

```
set_driving_cell -lib_cell cell_name -library lib_name -max  
[get_ports p]
```

The [set_driving_cell](#) constraint is specified for the combinations -max -rise and -max -fall. However, the constraint is not specified for the combinations -min -rise and -min -fall. This implies that the [set_driving_cell](#) constraint is incompletely specified for port p. Therefore, the *SDC_Methodology34* rule reports a single violation for the combinations -min -rise and -min -fall.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology35

Reports `set_max_transition` that is not completely specified

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_Methodology35* rule reports `rise` or `fall` arguments that are not specified for the given *set_max_transition* constraint.

Parameter(s)

None

Constraint(s)

- *set_max_transition*: Use to specify the transition limit, which is the maximum transition time in library time units.

Messages and Suggested Fix

The following message appears if a combination of options `<options>` is not specified in the *set_max_transition* constraint for the `<design-object-specification>`:

[WARNING] Option(s) `<options>` not specified for `set_max_transition: <design-object-specification>`

Here,

`<options>` is a comma-separated string of missing options.

`<design-object-specification>` is the type of the design object, such as `port`, `pin`, or `net`. `<design-object-specification>` is the name of the design object type appended to the design object. For example, if the design object type is `pin` and the design object is `hier_name`, the design object specification is `pin_hier_name`.

Potential Issues

The violation message appears if a combination of options is not specified

in the `set_max_transition` constraint for the design object type.

Consequences of Not Fixing

If you do not fix this violation, timing analysis tool may pick up default value which may cause chip failure.

How to Debug and Fix

Ensure that all the missing options are provided.

Example Code and/or Schematic

Consider the following example:

```
set_max_transition value -rise [get_ports p]
```

In the above example, the value has been specified for the `rise` argument. However, the value is not specified for the `fall` argument. This means that the `set_max_transition` constraint is incompletely specified for the port `p`. Therefore, the `SDC_Methodology35` rule reports a single violation for the `fall` argument.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology36

Lists the missing `-rise` or `-fall` options in the `set_min_transition` constraint

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_Methodology36* rule lists the missing `-rise` or `-fall` options that are not specified for the given *set_min_transition* constraint.

Parameter(s)

None

Constraint(s)

- *set_min_transition* (Mandatory): Use this constraint to specify the rising or falling transition for the pins selected.

Messages and Suggested Fix

The following message appears when a combination of options `<options>` is not specified in the *set_min_transition* constraint for the `<design-object-specification>`:

[WARNING] Option(s) `<options>` not specified for `set_min_transition: <design-object-specification>`

Where,

- `<options>` is a comma-separated string of missing options.
- `<design-object-specification>` is the type of the design object, such as port, pin, net and so on.
- `<design-object-specification>` is the name of the design object type appended to the design object. For example, if the design object type is `pin` and the design object is `hier_name` then the design object specification is `pin_hier_name`.

Potential Issues

The violation message appears if a combination of options is not specified in the [set_min_transition](#) constraint for the design object type.

Consequences of Not Fixing

If you do not fix this violation, timing analysis tool may pick up default value which may cause chip failure.

How to Debug and Fix

Ensure that all the missing options are provided.

Example Code and/or Schematic

Consider the following code snippet in *SDC_Methodology36* rule:

```
set_min_transition value -rise [get_ports p]
```

In the above example, the value has been specified for the option `-rise`. However, the value is not specified for the `-fall` option. This means that the [set_min_transition](#) constraint is incompletely specified for the port `p`. Therefore, the *SDC_Methodology36* rule will report a single violation for the option `-fall`.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology60

Identifies `set_operating_conditions` that should not be redefined

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_Methodology60* rule reports designs for which the `set_operating_conditions` constraint has been redefined. If more than one `set_operating_conditions` constraint is specified, the *SDC_Methodology60* rule reports for each redefinition of the `set_operating_conditions` constraint specified in the SDC file.

Parameter(s)

None

Constraint(s)

- `set_operating_conditions` (Mandatory): Use this constraint to define the cells or ports on which to set operating conditions.

Messages and Suggested Fix

The following message appears for the design/block `<name>` that has a redefined `set_operating_conditions` constraint set:

```
[WARNING] set_operating_conditions is redefined for design/  
block <name>
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Multiple operating conditions can lead to chip failure.

How to Debug and Fix

Ensure that the `set_operating_conditions` constraint is defined once in the SDC file.

Example Code and/or Schematic

Consider the following *set_operating_conditions* constraints:

```
set_operating_conditions -analysis_type on_chip_variation -  
library mn7slp_s1_mn7sstd3wellmvt012v_C1V12S4_MAX [get_cells  
FD1]
```

```
set_operating_conditions -analysis_type on_chip_variation -  
library mn7slp_s1_mn7sstd3wellmvt012v_C1V12S4_MAX [get_cells  
FD2]
```

The *SDC_Methodology60* rule generates a violation message for the second constraint in the same SDC file even if a different object list is specified in both the constraints.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology62

Determines missing `set_max_capacitance` on each port/design

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_Methodology62* rule determines missing `set_max_capacitance` on each port/design. It determines the input/output/inout ports or a design/block depending on the value of the `tc_opt01_port_des` parameter as shown below:

Value	Rule reports
design	The design/block for which the <code>set_max_capacitance</code> constraint is not specified.
port	The input/output/inout ports for which the <code>set_max_capacitance</code> constraint is not specified.
both	The design and input/output/inout ports for which the <code>set_max_capacitance</code> constraint is not specified.
design_or_port	The input/output/inout ports if the <code>set_max_capacitance</code> constraint is not specified for a design.

Rule Exceptions

The *SDC_Methodology62* rule ignores the input/output/inout ports in following conditions:

- The `set_case_analysis` or `set_disable_timing` constraint is applied on a port.
- A port is hanging, power, or ground.
- The `set_false_path` constraint is applied on a port (the `set_false_path` constraint is considered only if the `tc_ignore_te` rule parameter is unset and the `ignore_io_if_fp` rule parameter is set).
- The fan-out/fan-in of a port reaches the D/Q pin of a flip-flop whose clock pins are constant.

However, it does not list input/output/inout ports or a design/block if the `chip` rule parameter is set for a top-level block.

Parameter(s)

- *tc_opt01_port_des*: Default value is both. Set the value to `port`, `design`, `both`, `design_or_port` to report ports or design.
- *chip*: Default value is unspecified. Set the value to a string value to specify the name of the design unit which corresponds to the chip level.
- *tc_ignore_te*: Default value is yes. Set the value to no to consider the *set_multicycle_path* constraints.
- *ignore_io_if_fp*: Default value is no. Set the value to yes to consider *set_false_path* constraint.

Constraint(s)

- *set_max_capacitance* (Optional): Use this constraint to specify the rising or falling capacitance for all selected pins. This constraint sets maximum capacitance for pins, ports, clocks or designs.
- *set_false_path* (Optional): Use this constraint to identify the paths in a design that are to be marked as false such that they are not considered during timing analysis.
- *set_disable_timing* (Optional): Use this constraint to disable the timing arcs in a circuit.

Messages and Suggested Fix

Message 1

The following message appears for an input/output/inout port *<port-name>* without the *set_max_capacitance* constraint specified:

```
[WARNING] set_max_capacitance is not specified for <port-type>
<port-name> for design/block design_or_block_name
```

Potential Issues

This violation message appears if the *set_max_capacitance* constraint is not specified for an input/output/inout port.

Consequences of Not Fixing

If you do not fix this violation, timing analysis tool may pick up default value that may cause chip failure.

How to Debug and Fix

Ensure to specify capacitance value.

Message 2

The following message appears for a design *<name>* without the [set_max_capacitance](#) constraint set:

[WARNING] set_max_capacitance is not specified for design *<name>*

Potential Issues

This violation message appears if the [set_max_capacitance](#) constraint is not specified for the design.

Consequences of Not Fixing

If you do not fix this violation, timing analysis tool may pickup default value which may cause chip failure.

How to Debug and Fix

Ensure to select the parameter value and then provide the value for design or ports depending on the methodology required.

Example Code and/or Schematic

Assume the `tc_opt01_des_port` parameter is set to port and no value is specified for the [set_max_capacitance](#) constraint for the port that is defined. Therefore, this rule reports a violation.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology63

Determines the missing `set_max_transition` on each port/design

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_Methodology63* rule determines missing `set_max_transition` constraint on each port/design. It determines the input/output/inout ports or a design/block depending on the value of the `tc_opt01_port_des` parameter as shown below:

Value	Rule Reports
design	The design/block for which the <code>set_max_transition</code> constraint is not specified
port	The input/output/inout ports for which the <code>set_max_transition</code> constraint is not specified
both	The design and input/output/inout ports for which the <code>set_max_transition</code> constraint is not specified
design_or_port	The input/output/inout ports if the <code>set_max_transition</code> constraint is not specified for a design

Rule Exceptions

The *SDC_Methodology63* rule ignores input/output/inout ports if:

- `set_case_analysis` or `set_disable_timing` constraint is applied on a port that is hanging, power, or ground.
- `set_false_path` constraint is applied on a port. The `set_false_path` constraint is considered only if the `tc_ignore_te` rule parameter is unset and the `ignore_io_if_fp` rule parameter is set.
- The fan-out/fan-in of a port reaches the D/Q pin of a flip-flop whose clock pins are constant.

However, the *SDC_Methodology63* rule does not list input/output/inout ports or a design/block if the `chip` rule parameter is set for a top-level block.

Parameter(s)

- *tc_opt01_port_des*: Default value is both. Set the value to `port`, `design`, `both`, `design_or_port` to report ports or design.
- *chip*: Default value is unspecified. Set the value to a string value to specify the name of the design unit which corresponds to the chip level.
- *tc_ignore_te*: Default value is yes. Set the value to no to consider the `set_multicycle_path` constraints.
- *ignore_io_if_fp*: Default value is no. Set the value to yes to consider `set_false_path` constraint.

Constraint(s)

- *set_max_transition* (Optional): Use to specify the rising or falling transition for all selected pins.
- *set_false_path* (Optional): Use this constraint to identify the paths in a design that are to be marked as false, so that they are not considered during timing analysis.
- *set_case_analysis* (Optional): Use to specify the list of ports or pins on which the specified constant value or transition is to be applied.
- *set_disable_timing* (Optional): Use this constraint to disable the timing arcs in a circuit.

Messages and Suggested Fix

Message 1

The following message appears for an input/output/inout port `<portname>` without the *set_max_transition* constraint set:

```
[WARNING] set_max_transition is not specified for <port-type>
<portname> for design/block design_or_block_name
```

Potential Issues

This violation message appears when a port is not specified with the *set_max_transition* constraint.

Consequences of Not Fixing

If you do not fix this violation, timing analysis tool may pick up default

value that can cause chip failure.

How to Debug and Fix

Ensure to specify capacitance value.

Message 2

The following message appears for a design `<name>` without a `set_max_transition` constraint set:

```
[WARNING] set_max_transition is not specified for design <name>
```

Potential Issues

This violation message appears when a design is not specified with `set_max_transition` constraint.

Consequences of Not Fixing

If you do not fix this violation, timing analysis tool may pick up default value which can cause chip failure.

How to Debug and Fix

Ensure to specify capacitance value.

Example Code and/or Schematic

For `tc_opt01_des_port = port`, no `set_max_transition` value for port is defined. Therefore, the rule reports a violation in this case.

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology65

Identifies multiple `set_case_analysis` getting propagated to different pins of the same instance

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_Methodology65* rule reports instances where multiple `set_case_analysis` settings are being propagated to different pins of the instance. In such a case, one of the `set_case_analysis` constraints may become redundant for further propagation as the other constraints may dominate the instance.

Parameter(s)

None

Constraint(s)

SDC

- `set_case_analysis` (Mandatory): Use to specify that a port or pin is at a constant logic value 1 or 0.

Messages and Suggested Fix

The following message appears for instance `<inst-name>` when multiple `set_case_analysis` settings are being propagated to different pins of the instance:

[WARNING] Multiple constraints values are reaching different pins of '`<inst-name>`'

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

This can cause a portion of a design to fail functionally.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the design object that has conflicting case analysis values. Use the information reported by the [Show_Case_Analysis](#) rule to debug the reported violation message.

Update the value of [set_case_analysis](#) to resolve the conflict.

Example Code and/or Schematic

For the following snippet, the *SDC_Methodology65* rule reports a violation message because there is a [set_case_analysis](#) conflict in value propagated from the input port A to the value applied on I1/Z.

```
//test.v
module top (A,B,C);
input A, B;
output C;
AN2 I1(.A(A),.B(B),.Z(C));
endmodule;

//test1.sdc
set_case_analysis 1 A
set_case_analysis 0 I1/Z
```

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology66

Reports conflicting `set_case_analysis` on a design object

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_Methodology66* rule reports `set_case_analysis` constraints that conflict with the propagated case analysis settings for the same design object. In addition, this rule identifies `set_case_analysis` constraints that have a conflicting value with the value of the supply port.

Rule Exceptions

This rule does not check the conflict beyond the sequential element.

Parameter(s)

None

Constraint(s)

SDC

- `set_case_analysis` (Mandatory): Use to specify that a port or pin is at a constant logic value 1 or 0.

Messages and Suggested Fix

The following message appears for port or pin `<ppin-name>` when the value `<value1>` specified with a `set_case_analysis` constraint is not the same as the propagated case analysis value `<value2>`:

```
[WARNING] set_case_analysis (value: <value1>) at "<ppin-name>"
conflicts with propagated value (value: <value2>)"
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

This can cause a portion of a design to fail functionally.

This is pointing to inconsistency of specification of the [set_case_analysis](#) command. If inconsistency is allowed to persist, the timing results will be different and you may perform timing analysis with the wrong set of [set_case_analysis](#) commands.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the design object that has conflicting case analysis values. Use the information dumped by the [Show_Case_Analysis](#) rule to debug the reported violation message.

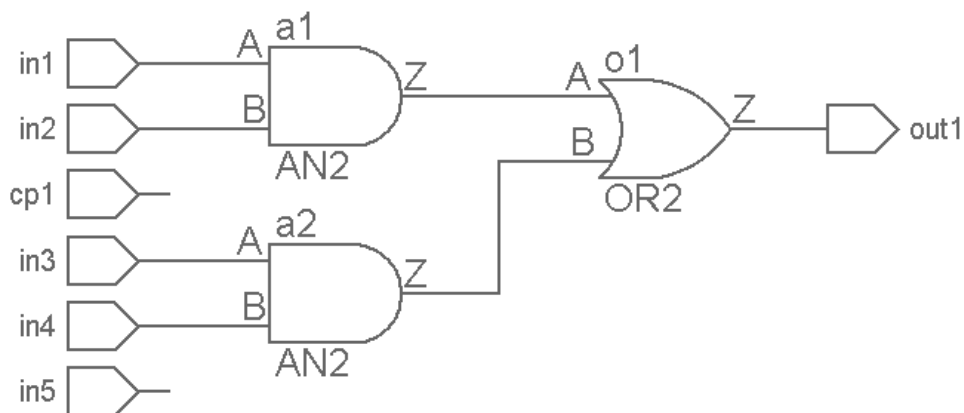
Update the value of [set_case_analysis](#) to resolve the conflict.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top .sdc file snippet is as follows:

Methodology Rules

```

top.sdc | top.sdc (2) |
set_case_analysis 1 in1
set_case_analysis 0 in2
set_case_analysis 1 a1/Z
set_case_analysis 0 in3
set_case_analysis 1 a2/Z

```

The violation message appears because there is a conflict between the propagated and the `set_case_analysis` constraint values. In the top.sdc file, the corresponding constraint is highlighted.

Example 2

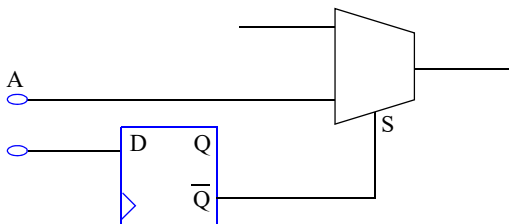
Test Case Files Not Available

In this example, the value that is being propagated to the S pin of the mux is 0. However, the value 1 is forced at this pin. Therefore, 1 is used. This would be incorrect if your intention was to perform the analysis with value 0.

```

set_case_analysis 0 [get_pins a_reg/Q]
set_case_analysis 1 [get_pins select/S]

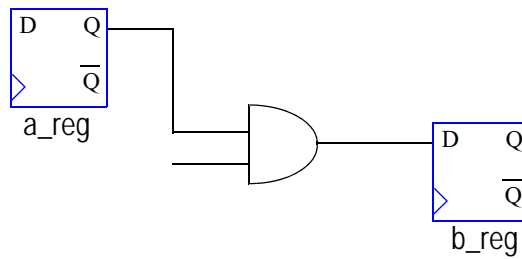
```

**Example 3**

Test Case Files Not Available

In this example, the value that is being propagated to b_reg/D is 0, but the value 1 is forced at this pin. Therefore, 1 is used. This would be incorrect if your intention was to perform the analysis with value 0.

```
set_case_analysis 0 [get_pins a_req/Q]  
set_case_analysis 1 [get_pins b_reg/D]
```



Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology67

Reports conflicting `set_case_analysis` applied on an output pin of a flip-flop or on a pin in its fan-out

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_Methodology67* rule reports `set_case_analysis` constraints applied on an output pin, or in its fan-out, for a flip-flop, if the value propagated to or applied to the input of the flip-flop is inconsistent. This rule does one extra cycle of simulation for sequential cells having constant value propagated at their input pins. After simulating one extra cycle, the constant value is propagated to some flops. Those which are in the fan-out of a flip-flop that has a constant output after the extra cycle simulation.

Rule Exceptions

The *SDC_Methodology67* rule reports a violation only when a flip-flop is encountered on traversing back from the node where there is a conflict, while the *SDC_Methodology66* rule stops at a sequential cell while doing the traversal.

Parameter(s)

None

Constraint(s)

SDC

- `set_case_analysis` (Mandatory): Use to specify that a port or pin is at a constant logic value 1 or 0.

Messages and Suggested Fix

The following message appears for flip-flop output pin `<flop-name>` when the value `<value1>` specified with a `set_case_analysis` constraint on the flip-flop output pin or a port/pin in its combinational fan-out is not the same as propagated case analysis value `<value2>` for the flip-flop data pin or a port/pin in its combinational fan-in:

[WARNING] `set_case_analysis (value: <value1>) at "<pin-name>" conflicts with propagated value (value: <value2>)`

Potential Issues

The constant value applied and value being propagated should be the same.

Consequences of Not Fixing

This can cause a portion of a design to fail functionally. This can cause a portion of a design to fail functionally.

How to Debug and Fix

View the incremental schematic of the violation message.

The schematic shows the design object that has conflicting case analysis values. Use the information dumped by the [Show_Case_Analysis](#) rule to debug the reported violation message.

The rule reports a violation only when a flip-flop is encountered while traversing back from the object where there is a conflict, and the [SDC_Methodology66](#) rule reports violation for all the other cases.

To resolve this violation, update the constant value applied on the output pin of the flip-flop or the value applied in the fan-in of that flip-flop.

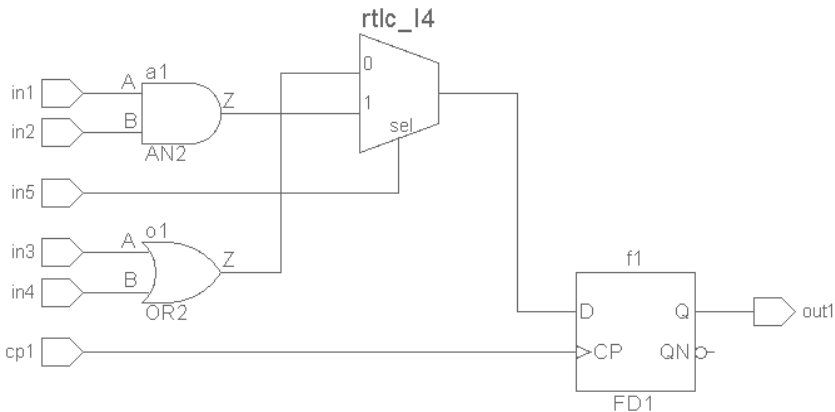
Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:

Methodology Rules



The `top.sdc` file snippet is as follows:

```

top.sdc top.sdc (3)
set_case_analysis 1 in1
set_case_analysis 0 in2
set_case_analysis 0 in3
set_case_analysis 1 in4

set_case_analysis 1 in5
set_case_analysis 1 out1

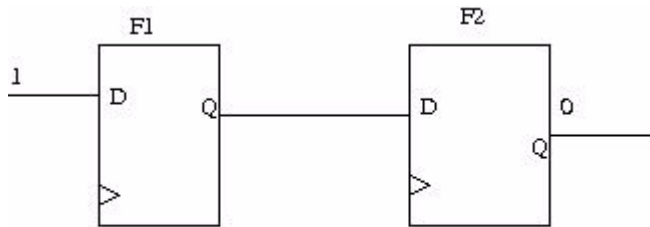
```

The violation message appears because there is a conflict between the fan-out of the flip-flop `f1` and the specified `set_case_analysis` value. In the `top.sdc` file, the corresponding constraint is highlighted.

Example 2

Test Case Files Not Available

Suppose, a constant value 1/0 is being propagated to the data-pin of a flip-flop `F1`, `F1/Q` is connected to data-pin of another flip-flop `F2`, and a constant value 0/1 is applied at `F2/Q`. In this case, there is a conflict at flip-flop `F2`.



Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology68

Reports all unconstrained non-clock input signals

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_Methodology68* rule reports all unconstrained input and output ports. The ports used in the *set_false_path* constraint are assumed to be unconstrained.

Rule Exception(s)

The *SDC_Methodology68* rule reports violation if the *set_false_path* constraint is given with a port. Rule does not consider *set_false_path* constraints approaching input/inout port.

Parameter(s)

None

Constraint(s)

- *set_false_path* (Mandatory): Use this constraint to identify the paths in a design that are to be marked as false, so that they are not considered during timing analysis.

Messages and Suggested Fix

Message 1

The following message appears for an unconstrained input port *<port-name>*:

```
[INFO] Input port "<port-name>" is unconstrained
```

Potential Issues

The message appears when an input port is unconstrained.

Consequences of Not Fixing

If you do not fix this, STA tool makes assumptions about the input and output delays. Hence, even if timing is met, chip may not meet timing in

real world.

How to Debug and Fix

Depending on the design requirements, you may need to constraint the reported ports. If this is the case, update the SDC file accordingly.

Message 2

The following message appears for an unconstrained inout port *<port-name>*:

```
[INFO] Inout port "<port-name>" is unconstrained
```

Potential Issues

The message appears when an inout port is unconstrained.

Consequences of Not Fixing

If you do not fix this, STA tool makes assumptions about the input and output delays. Hence, even if timing is met, chip may not meet timing in real world.

How to Debug and Fix

Depending on the design requirements, you may need to constrain the reported ports. If this is the case, update the SDC file accordingly.

Example Code and/or Schematic

For the following snippet, the *SDC_Methodology68* rule reports a violation.

```
//test.sdc
set_false_path -from [get_ports {a}]
set_load 2.0 a
```

In the above example, input port is to be assumed unconstrained as [*set_false_path*](#) is applied.

Default Severity Label

Info

Rule Group

SDC_Methodology

Methodology Rules

Reports and Related Files

None

SDC_Methodology69

Reports the use of `get_nets`

When to Use

Use this rule to ensure that the commands specified are not referring to nets directly.

Description

The *SDC_Methodology69* rule reports the use of the `get_nets` command.

Parameter(s)

None

Constraint(s)

- *get_nets* (Mandatory): Use this constraint to create a collection of nets from the netlist.

Messages and Suggested Fix

Message

The following message appears for a *get_nets* command:

[ERROR] The `get_nets` command should not be used

Potential Issues

The violation message appears when `get_nets` command is used on the nets directly.

Consequences of Not Fixing

This is a methodology choice.

How to Debug and Fix

Move the command to either fan-in or to fan-out pin.

Example Code and/or Schematic

None

Methodology Rules

Default Severity Label

Error

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology70

Reports constraint(s) that do not have the comment option specified

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_Methodology70* rule reports violations when any of the following constraints do not have the `comment` option specified:

- *create_clock*
- *create_generated_clock*
- `group_path`
- *set_clock_groups*
- *set_false_path*
- *set_max_delay/set_min_delay*
- *set_multicycle_path*

Prerequisites

For this rule to run, specify the list of constraints using the *tc_comments_cmd_file* parameter. If the file specified through this parameter is empty or contains unsupported constraints, this rule does not run.

Parameter(s)

tc_comments_cmd_file: Default is unspecified. Set this parameter to the file that contains the list of constraints that the *SDC_Methodology70* rule should check for the `comment` option.

Constraint(s)

None

Messages and Suggested Fix

The following message appears for constraints that do not have the `comment` option specified:

[WARNING] Comment(s) not provided for constraint(s) using `-comment` option

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

There are no consequences of not fixing this violation message.

How to Debug and Fix

Double-click the violation message. The SDC file appears. In this file, specify the `comment` option for the constraints that do not have the `comment` option. The list of constraints that are checked by the *SDC_Methodology70* rule are located in the file fed through the [tc_comments_cmd_file](#) parameter.

Example Code and/or Schematic

For the following snippet, the *SDC_Methodology70* rule reports a violation because constraint does not have the `comment` option specified.

```
create_clock -period 11 [get_ports CLK]
```

To resolve the violation, insert the `comment` option, as shown in the following snippet:

```
create_clock -period 11 [get_ports CLK] -comment "Modified  
the period from 5 to 11"
```

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology71

Reports a `set_drive` constraint that has been incorrectly specified

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_Methodology71* rule reports ports in a block in which the `-min` value of the `set_drive` constraint is greater than the `-max` value.

Parameter(s)

None

Constraint(s)

- `set_drive` (Mandatory): Use this constraint to set the resistance on specified input or inout ports in the current design.

Messages and Suggested Fix

The following message appears when the `set_drive` constraint is incorrectly specified for ports in a block:

```
[WARNING] set_drive min value is greater than max value for  
<ports-list> for block <block-name>
```

Potential Issues

The violation message explicitly states the affected ports when the number of ports is less than 11. However, when the number ports is more than 10, the violation message appends “...” after the tenth port stated in the message.

Consequences of Not Fixing

As the `-min` value cannot be greater than the `-max` value, the STA tool can make assumptions on either of the values and produce erroneous results.

How to Debug and Fix

Double-click the violation message. The SDC file appears. Every port that has an incorrect definition is highlighted.

To resolve the violation message, ensure the `-min` value of the `set_drive` constraint is lesser than the `-max` value for the ports stated in the violation message.

Example Code and/or Schematic

For the following snippet, the `SDC_Methodology71` rule reports a violation because the `-min` value of the `set_drive` constraint is lesser than the `-max` value.

```
set_drive -min -rise 6.5 in
set_drive -max -fall 5 in
```

To resolve the violation, decrease the `-min` value so that it is less than the `-max` value or increase the `-max` value, as shown in the following snippet:

```
set_drive -min -rise 6.5 in
set_drive -max -fall 9 in
```

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology72

Performs boundary checks for Design Rule Checks (DRC) constraints with user-specified limits

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_Methodology72* rule reports violations for the following SDC constraints, if the constraint value is greater than the corresponding boundary limit:

- set_load
- set_input_transition
- set_max_capacitance
- set_max_transition
- set_clock_transition
- set_clock_uncertainty
- set_clock_latency
- set_driving_cell

See [DRC Constraints - Sources of Limit](#) for more information.

Prerequisites

To use this rule, specify a DRC specification file. This file defines the boundary limits (See [Example Code and/or Schematic](#)). The *tc_drc_spec_file* and *tc_tech* parameters are used to provide the file path and file name.

If you do not specify the DRC specification file, this rule reports a violation.

DRC Constraints - Sources of Limit

The limits may also be present in other places such as library level/cell pin level attributes. The most conservative values are selected as the limit. The following table lists the sources of the selected limit.

Methodology Rules

Constraint	Object Type	Sources of Limit
<i>set_load</i>	ports, nets	<p>Input and Output Ports:</p> <ul style="list-style-type: none"> ■ default_max_capacitance in all libraries ■ set_max_capacitance SDC constraint ■ input_load/output_load parameter <p>Nets:</p> <ul style="list-style-type: none"> ■ max_capacitance of driving pins ■ default_max_capacitance in all libraries
<i>set_max_capacitance</i>	ports, design	<p>Input Port:</p> <ul style="list-style-type: none"> ■ default_max_capacitance in all libraries ■ sig_max_cap parameter <p>Output Port:</p> <ul style="list-style-type: none"> ■ default_max_capacitance in all libraries ■ max_capacitance of driving pins ■ sig_max_cap parameter <p>Design:</p> <ul style="list-style-type: none"> ■ default_max_capacitance in all libraries
	clocks	<ul style="list-style-type: none"> ■ default_max_capacitance in all libraries ■ clk_max_cap parameter

Constraint	Object Type	Sources of Limit
<i>set_max_transition</i>	ports, design	<p>Input Port:</p> <ul style="list-style-type: none"> ■ default_max_transition in all libraries ■ sig_max_tran/rst_max_tran parameter <p>Output Port:</p> <ul style="list-style-type: none"> ■ default_max_transition in all libraries ■ max_transition of driving pins ■ sig_max_tran/rst_max_tran parameter <p>Design:</p> <ul style="list-style-type: none"> ■ default_max_transition in all libraries
	clocks	<ul style="list-style-type: none"> ■ default_max_capacitance in all libraries ■ clk_max_cap parameter
<i>set_clock_transition</i>	clocks	<ul style="list-style-type: none"> ■ max_transition of driving pins ■ default_max_transition in all libraries ■ clk_max_tran parameter
<i>set_input_transition</i>	ports (input/inout)	<ul style="list-style-type: none"> ■ max_transition of driving pins ■ default_max_transition in all libraries ■ clk_max_tran parameter
intra_clock_uncertainty	clocks, ports, pins	<ul style="list-style-type: none"> ■ clk_uncert_setup_intra parameter for checking setup value ■ clk_uncert_hold_intra parameter for checking hold value

Constraint	Object Type	Sources of Limit
inter_clock_uncertainty	clocks	<ul style="list-style-type: none"> ■ clk_uncert_setup_inter parameter for checking setup value ■ clk_uncert_hold_inter parameter for checking hold value
set_clock_latency	clock, ports, pins	<ul style="list-style-type: none"> ■ clk_max_latency parameter
set_driving_cell	ports	<ul style="list-style-type: none"> ■ max_driving_cell_tran parameter for checking input_transition_rise and input_transition_fall value ■ driving_cell parameter for checking lib_cell value

Parameter(s)

- [tc_drc_spec_file](#) and [tc_tech](#): Default is none. Use these parameters to specify a template file that contains limiting values for DRC constraints.

Constraint(s)

- None

Messages and Suggested Fix

Message 1

The following message appears when the DRC specification file does not exist at path `<path-name>`:

```
[WARNING] DRC spec file does not exist at following path
<path-name>
```

Message 2

The following message appears when the constraint `<constraint-name>` has a constraint field (constraint-field), which has a value `<constraint-value>` greater than the limit `<limit-value>` as selected from either the variables `<source-attribute>` of the DRC specification file

<limit-source> or library <limit-source> level attributes <source-attribute>:

[WARNING] <constraint-name> constraint's <constraint-field> (constraint-value) is greater than the limit (<limit-value> taken from <limit-source> <source-attribute>)

Message 3

The following message appears when the constraint <constraint-name> has a constraint field (constraint-field), which has a value <constraint-value> that is not the same as the specified limit value <limit-value> as selected from the variables <source-attribute> of the DRC specification file <limit-source>:

[WARNING] <constraint-name> constraint's <constraint-field> (constraint-value) is not the same as the variable value (<limit-value> taken from <limit-source> <source-attribute>)

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

The consequences of not fixing the messages are:

Message 1: The user-specified limits cannot be applied.

Message 2: The timing can break or the results may be incorrect.

Message 3: The timing can break or the results may be incorrect.

How to Debug and Fix

Double-click the violation message. The SDC file appears.

To resolve the violation message,

Message 1: Check the value of the *tc_drc_spec_file* and *tc_tech* parameters.

Message 2: Review the DRC file or update the SDC file.

Message 3: Review the DRC file or update the SDC file.

Example Code and/or Schematic

The following is a sample DRC Specification file containing complete list of supported variables with values.

Methodology Rules

```
# Load on input port
set input_load [expr 2.0/$period]
# Load on output port
set output_load 9.0

# Max capacitance of clock
set clk_max_cap 4.0
# Max capacitance of signals
set sig_max_cap 4.0

# Maximum transition for clocks
set clk_max_tran 4.0
# Maximum transition on reset
set rst_max_tran 4.0
# Maximum transition on signals
set sig_max_tran 4.0

# Intra clock uncertainty for worst case
set clk_uncert_setup_intra 7.0
# Intra clock uncertainty for best case
set clk_uncert_hold_intra 6.0

# Inter clock uncertainty for worst case
set clk_uncert_setup_inter 5.0
# Inter clock uncertainty for best case
set clk_uncert_hold_inter 4.0
```

```
# Maximum clock latency
set clk_max_latency 3.0
# Input driving cell
set driving_cell FD1
# Maximum transition at the input of driving cell
set max_driving_cell_tran 1.0
```

Default Severity Label

Warning

Rule Group

SDC_Methodology

Reports and Related Files

None

SDC_Methodology73

Generates RTL-level SDC constraints from netlist-level SDC constraints

When to Use

Use this rule when you have an RTL design, but SDC constraints on netlist.

Description

The *SDC_Methodology73* rule generates RTL-level SDC constraints from netlist-level SDC constraints.

For *create_clock*, *create_generated_clock*, and *set_case_analysis*, the new constraints are generated on the module boundary ports. The constraints defined on combinational cell pins are processed and the constraints are migrated to the corresponding parent module boundary skipping buffers and invertors.

For the following SDC constraints, migration is not performed. In addition, the constraints are generated with the same definition points as specified in the netlist-level SDC file:

- *set_clock_uncertainty*
- *set_input_delay*
- *set_output_delay*
- *set_false_path*
- *set_multicycle_path*
- *set_clock_groups*
- *set_clock_sense*
- *set_disable_timing*

The constraints are generated in a file called *new_clock_definitions_0*, which is located in *spyglass_reports/constraints* area.

Rule Exceptions

The *SDC_Methodology73* exceptions are as follows:

1. Module boundary ports are found only for combinational instance (not black box, grey box) pins on which constraints are specified.

2. Module boundary ports are found by skipping only buffers and inverters. If other types of instances are encountered, boundary ports are not identified.
3. If there is a change in polarity from the specified point to the module boundary boundary, migration is not performed.

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

The following message appears when the file that contains the SDC clock constraints have been migrated:

```
[INFO] Migrated SDC clock constraints are generated in file  
<file-name>
```

Potential Issues

Not applicable

Consequences of Not Fixing

Not applicable

How to Debug and Fix

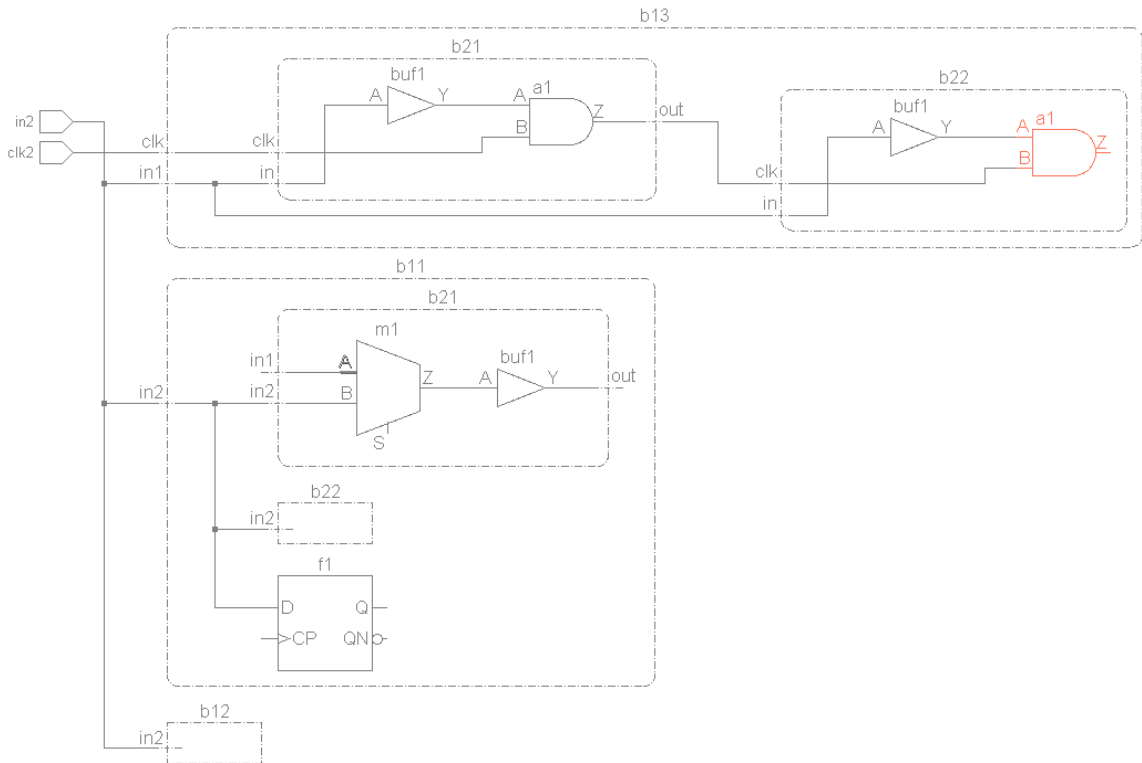
Double-click the violation message to view the generated SDC file.

Example Code and/or Schematic

This example shows the file that is generated after migrating the netlist SDC constraints to RTL level constraints.

Consider the following schematic:

Methodology Rules



For this design, the following top-level netlist SDC constraints file is specified:

Original Sdc file: top1.sdc

```
create_clock -name C1 -period 10 -waveform { 0.0 5.0 }
[get_ports in2]
```

```
create_clock -name C2 -period 20 -waveform { 0.0 10.0 }
[get_pins b13/b21/a1/A]
```

```
create_generated_clock -name GC1 -source [get_pins b11/b21/
m1/A] { b11/b21/m1/Z b11/b21/out} -edges {1 3 5}
```

```
create_generated_clock -name GC2 -master C1 -add -source
[get_pins b11/b21/m1/B] [get_ports out1] -divide_by 2
```

After running this rule, the following file, called new_clock_definitions_0, is

generated. This file contains SDC clock constraints that can be used at the RTL level.

```
Generated file: new_clock_definitions_0
```

```
#in2 -----> Not Migrated
```

```
#FileName: top1.sdc Line Number: 1
```

```
create_clock -name C1 -period 10 -waveform {0 5 } { in2 }
```

```
#b13/b21/a1/A -----> b13/b21/in
```

```
#FileName: top1.sdc Line Number: 2
```

```
create_clock -name C2 -period 20 -waveform {0 10 } { b13/b21/  
in }
```

```
#b11/b21/m1/A -----> b11/b21/in1
```

```
#b11/b21/m1/Z -----> b11/b21/out
```

```
#b11/b21/out -----> Not Migrated
```

```
#FileName: top1.sdc Line Number: 3
```

```
create_generated_clock -name GC1 -edges { 1 3 5 } -source  
b11/b21/in1 { b11/b21/out b11/b21/out }
```

```
#b11/b21/m1/B -----> b11/b21/in2
```

```
#out1 -----> Not Migrated
```

```
#FileName: top1.sdc Line Number: 4
```

```
create_generated_clock -name GC2 -divide_by 2 -add -source  
b11/b21/in2 -master_clock C1 { out1 }
```

Default Severity Label

Info

Rule Group

SDC_Methodology

Methodology Rules

Reports and Related Files

None

SDC_Methodology74

Reports multiplexers through which the fastest clock is not selected

When to Use

Use this rule when you have an RTL design, but SDC constraints on netlist.

Description

The *SDC_Methodology74* rule checks all multiplexers in a design and reports the multiplexers through which the fastest clock is not selected. A CSV file is generated, which contains more information on the multiplexer.

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

The following message appears when the spreadsheet is generated:

```
[INFO] CSV file <file-name> generated to report MUXes through  
which fastest clock is not selected for design/block <design-  
name/block-name>
```

Potential Issues

Not applicable

Consequences of Not Fixing

Not applicable

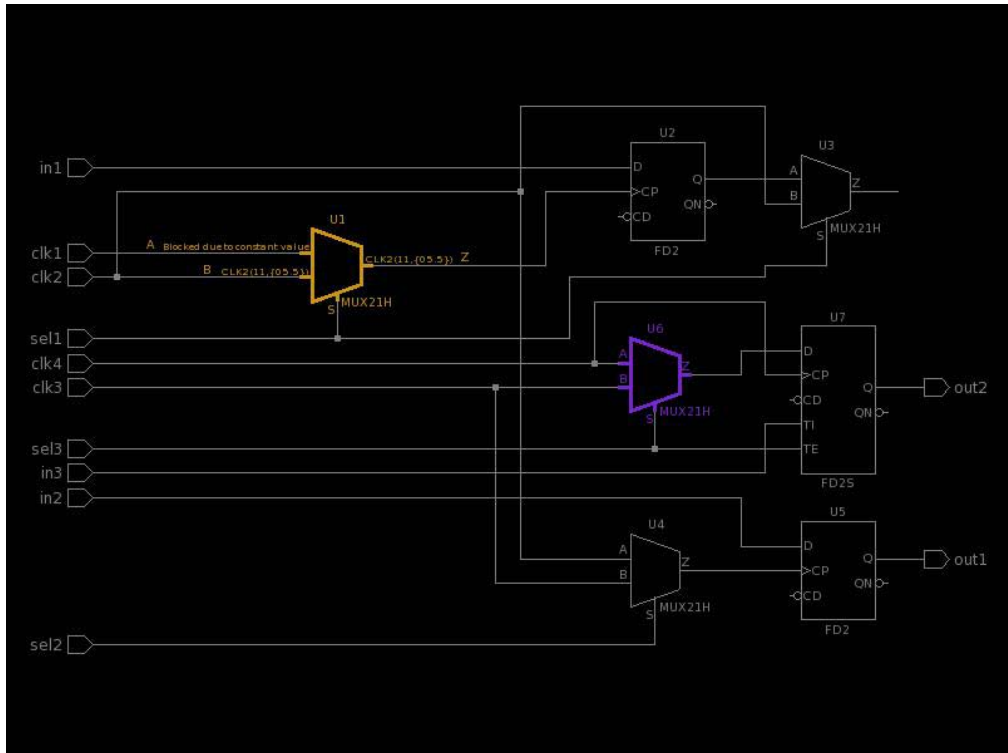
How to Debug and Fix

Double-click the violation message to view the generated CSV file in Spreadsheet Viewer.

Methodology Rules

	B	C	D	E
	Mux	Fastest Clock not selected	All input clocks	All selected clocks
1	top.mux1	C2:10.1	C2:10.1 C1:10.114	C1:10.114

The schematic highlights the multiplexer for which the fastest clock is not selected. The multiplexer is highlighted in yellow.



Example Code and/or Schematic

See [How to Debug and Fix](#).

Default Severity Label

Info

Rule Group

SDC_Methodology

Reports and Related Files

None

Miscellaneous Rules

The Miscellaneous Rules Group `SDC_Misc` contains the following rules:

Rule	Description
<i>SDC_Misc_Command01</i>	Commands specified on both design and port(s) when the command should be specified on only one (either design or port(s))
<i>SDC_Misc_Power01</i>	Missing <i>set_max_dynamic_power</i> or <i>set_max_leakage_power</i> commands
<i>SDC_Misc_Setup01</i>	<i>sdc_data</i> without any clocks specified
<i>SDC_Misc_WLM01</i>	Design units for which no wire load model has been specified in a particular analysis phase

SDC_Misc_Command01

Reports similar constraints that have been applied to different objects

When to Use

Use this rule for the RTL, Pre-layout and Post-layout phases.

Description

The *SDC_Misc_Command01* rule reports constraints that are specified on both design and port(s), whereas the constraints should be specified on either design or port(s). This rule checks for the *set_max_capacitance*, *set_max_transition*, and *set_max_fanout* constraints.

Rule Exception

This rule does not check the *set_min_capacitance* constraint because it can only be set on ports.

Parameter(s)

None

Constraint(s)

- *set_max_capacitance* (Optional): Use this constraint to specify pins, ports, clocks, or designs that need to have the maximum capacitance set.
- *set_max_transition* (Optional): Use this constraint to specify pins, ports, clocks, or designs where the maximum transition is set.
- *set_max_fanout* (Optional): Use this constraint to specify input ports or designs where the maximum fan-out needs to be set.

Messages and Suggested Fix

The following message appears when *<constr>* command is set on both the design and the port(s):

```
[INFO] <const > set for both design and ports
```

Where, *<constr>* can be *set_max_capacitance*, *set_max_transition*, or *set_max_fanout*.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

This is a Methodology related rule. Some tools may not behave as per your expectations.

How to Debug and Fix

Review the SDC file. The reported constraint is highlighted in the Console GUI.

To resolve this violation message, define the reported constraints for either the design or all ports.

Example Code and/or Schematic

In the following SDC snippet, the *SDC_Misc_Command01* rule reports a violation because *set_max_transition* is applied on both the design and port.

```
set_max_transition 1.0 in1 // where in1 is port
set_max_transition 2.0 [current_design]
```

Default Severity Label

Info

Rule Group

SDC_MISC

Reports and Related Files

None

SDC_Misc_Power01

Reports missing `set_max_dynamic_power` and `set_max_leakage_power`

When to Use

This rule is applicable to the RTL phase.

Description

The *SDC_Misc_Power01* rule reports missing `set_max_dynamic_power` or `set_max_leakage_power` constraints. For *SDC_Misc_Power01* rule, both `set_max_dynamic_power` and `set_max_leakage_power` constraints are required to be specified for a design or a block.

When both `set_max_dynamic_power` and `set_max_leakage_power` constraints are correctly specified, the *SDC_Misc_Power01* reports the power breakup, which includes the values of Max Dynamic Power, Max Leakage Power, and Total Power.

Parameter(s)

None

Constraint(s)

- `set_max_dynamic_power` (Optional): Use this constraint to specify the dynamic power for the current design.
- `set_max_leakage_power` (Optional): Use this constraint to specify the target leakage power for the current design.

Messages and Suggested Fix

The following message appears when both `set_max_dynamic_power` and `set_max_leakage_power` constraints are specified for design/block `<name>`:

```
[INFO] Max dynamic power (<value> <unit>), max leakage power (<value> <unit>) and total power (<value> <unit>) set for design/block "<name>"
```

Potential Issues

This violation message appears when both `set_max_dynamic_power` and `set_max_leakage_power` constraints have been set for current design.

Consequences of Not Fixing

A user running Power Compiler may forget to specify the leakage-power-target. Therefore, the tool might give very little importance to the leakage power target since there is none specified. As a result, the design will not be optimized for leakage power.

How to Debug and Fix

Not applicable

Message 2

The following message appears when both [set_max_dynamic_power](#) and [set_max_leakage_power](#) constraints are not found for design/block `<name>`:

```
[INFO] Neither set_max_dynamic_power nor set_max_leakage_power
is set for design/block "<name>"
```

Potential Issues

The message appears when both [set_max_dynamic_power](#) and [set_max_leakage_power](#) constraints are missing from the current design.

Consequences of Not Fixing

A user running Power Compiler may forget to specify the leakage-power-target. Therefore, the tool might give very little importance to the leakage power target since there is none specified. As a result, the design will not be optimized for leakage power.

How to Debug and Fix

Specify the missing dynamic and leakage constraint.

Message 3

The following message appears when only one of [set_max_dynamic_power](#) and [set_max_leakage_power](#) constraints has been found for design/block `<name>`:

```
[INFO] <constr> (<value> <unit>) has been set for design
"<name>", but no value is specified for <constr>
```

Where, `<constr>` can be [set_max_dynamic_power](#) or [set_max_leakage_power](#).

Potential Issues

The message appears when only one of [set_max_dynamic_power](#) or [set_max_leakage_power](#) constraints has been set for current design.

Consequences of Not Fixing

If no units are specified along with the constraint then units as specified in library would be taken. This may or may not be intended.

How to Debug and Fix

If you have intentionally left the units so that the units are the same as specified in the library, you do not need to take any action. Otherwise, specify appropriate units.

Example Code and/or Schematic

For the following snippet, the *SDC_Misc_Power01* rule reports a violation because both *set_max_dynamic_power* and *set_max_leakage_power* constraints have been set for current design.

```
set_max_dynamic_power 5 W  
set_max_leakage_power 5 KW
```

Default Severity Label

Info

Rule Group

SDC_MISC

Reports and Related Files

None

SDC_Misc_Setup01

Reports a violation when a clock is not specified for any of SDC files

When to Use

This rule is applicable for the RTL, pre-layout, and post-layout phases.

Description

The *SDC_Misc_Setup01* rule reports *sdc_data* which do not have any clocks specified.

Parameter(s)

None

Constraint(s)

- *create_clock* (Optional): Use this constraint to create clock on a design object.
- *create_generated_clock* (Optional): Use this constraint to create clock on a design object.

Messages and Suggested Fix

The following message appears:

[WARNING] No clock specified in any of the SDC files

Potential Issues

The violation message appears because the *sdc_data* constraint is specified in a SpyGlass Design Constraints (SGDC) file and the SDC file does not have any clocks specified.

Consequences of Not Fixing

When the clock does not define the register to register paths that are totally unconstrained, there is no performance target for the design.

How to Debug and Fix

Review the SGDC file. The *sdc_data* constraint is highlighted in the SGDC file.

To resolve the violation message, define the clocks in the SDC file specified with the `sdc_data` constraint in the SGDC file.

Example Code and/or Schematic

In the following SDC snippet, the `SDC_Misc_Setup01` rule reports a violation message because no clock (`create_clock/create_generated_clock`) is defined in test.sdc.

```
//test.sgdc  
current_design top  
sdc_data -type test.sdc
```

```
//test.sdc  
set_case_analysis 1 in
```

Default Severity Label

Warning

Rule Group

SDC_MISC

Reports and Related Files

None

SDC_Misc_WLM01

Checks for designs with missing wire-load models

When to Use

This rule is applicable to the RTL and pre-layout phases.

Description

The *SDC_Misc_WLM01* rule checks for design units for which no wire-load model has been specified.

Parameter(s)

None

Constraint(s)

- *set_wire_load_model* (Optional): Use this constraint to set wire load model on designs, ports, or hierarchical cells.

Messages and Suggested Fix

The following message appears when the design unit *<du-name>* has no wire-load model specified for the *<phase-name>* phase:

[WARNING] Design unit "*<du-name>*" has no wire load model defined

Potential Issues

The violation message appears if there is no wire-load model specified for a design.

Consequences of Not Fixing

If a wire-load model is missing, the block might not be able to get a good estimate of the interconnect delay, causing the delay computations to be overly optimistic.

How to Debug and Fix

This rule highlights the first command of the SDC file where you can view whether the *set_wire_load_model* constraint is defined in the SDC file. To fix this violation, define the *set_wire_load_model* constraint.

Example Code and/or Schematic

For the following snippet, the *SDC_Misc_WLM01* rule reports a violation because the *set_wire_load_model* constraint is given in the SDC file.

```
//test.sdc  
  
create_clock -name Clk1 -period 10.000000 -waveform {  
0.000000 5.000000 }  
  
set_clock_latency 10 {clka}  
  
#set_wire_load_model -name "10x10"
```

Default Severity Label

Warning

Rule Group

SDC_MISC

Reports and Related Files

None

Other Rules

The Other Rules Group `Others` contains the following rules:

Rule	Description
<i>SDC_Case_Sanity01</i>	Multiple <i>set_case_analysis</i> constraints set on the same path
<i>SDC_MergeBlocks</i>	Merges the SDC files for blocks under a top design

SDC_Case_Sanity01

Reports multiple `set_case_analysis` set on the same path

When to Use

This rule is applicable to the RTL, Pre-layout, Post-layout phases.

Description

The *SDC_Case_Sanity01* rule reports multiple `set_case_analysis` constraints set on the same path.

The *SDC_Case_Sanity01* rule traverses the fan-out of the nodes specified in the `set_case_analysis` specification and reports any other `set_case_analysis` specification found in the path.

Parameter(s)

None

Constraint(s)

- `set_case_analysis` (Mandatory): Use to specify that a port or pin is at a constant logic value 1 or 0 or is considered with a rising or falling transition.

Messages and Suggested Fix

Message

The following message appears for a `set_case_analysis` constraint (in file `<sdc-file2-name>` at line `<num2>`) when a node specified with this command is encountered while traversing a node specified with another `set_case_analysis` constraint (in file `<sdc-file1-name>` at line `<num1>`):

```
[INFO] set_case_analysis command at (file:<sdc-file2-name>
line:<num2>) is in the path of set_case_analysis command at
(file:<sdc-file1-name> line:<num1>)
```

Potential Issues

The violation message appears when multiple `set_case_analysis` constraints are specified for the same path.

Consequences of Not Fixing

Other Rules

If you do not fix this violation, multiple *set_case_analysis* constraints set on the same path can cause part of the design to fail functionally.

How to Debug and Fix

The SDC file/line of *set_case_analysis* constraints are highlighted in the GUI. To resolve this violation message, perform the following:

- If the values of these constraints are same, remove either one.
- If the values are conflicting, one of the value takes precedence. Validate it from the design perspective and update the value of *set_case_analysis* constraint, if required.

Example Code and/or Schematic

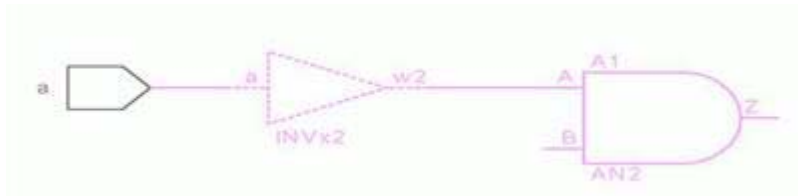
Consider the following snippet:

```
top.sdc:
```

```
set_case_analysis 1 a
```

```
set_case_analysis 1 A1/Z
```

The following schematic is generated.

**Default Severity Label**

Info

Rule Group

Others

Reports and Related Files

None

SDC_MergeBlocks

This rule has been deprecated.

Constraints Generation

For constraints generation, the SpyGlass Constraints solution provides the following rule:

Rule	Description
<i>SDC_GenerateIncr</i>	Generates a template SDC file incrementally for a design

SDC_GenerateIncr

Incrementally generate a template constraints file for a block

When to Use

Use this rule to generate SDC constraints in a systematic and incremental manner for your RTL design.

Description

The *SDC_GenerateIncr* rule automatically generates constraints incrementally into a template SDC file for your design. The Console GUI provides a wizard to guide you through the process of generating constraints.

Prerequisites

Before running this rule on your design, make sure you check the design for unintentional black boxes.

The *SDC_GenerateIncr* rule uses the SpyGlass TXV product for formal verification of clock-to-clock paths. Therefore, if you want to perform formal verification of clock-to-clock paths, you need the SpyGlass TXV license.

How to Use SDC_GenerateIncr

The *SDC_GenerateIncr* rule generates the template specifications *incrementally* by using a seed file.

The following diagram illustrates the incremental flow of using this rule.

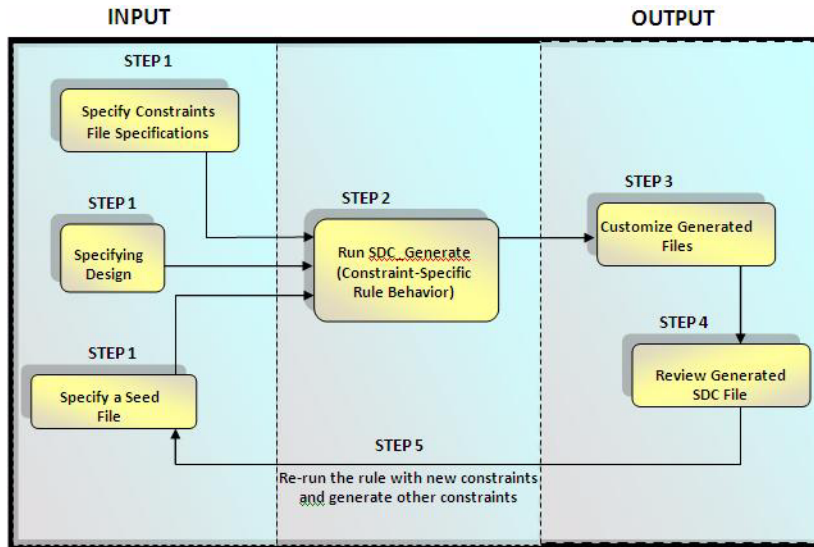


FIGURE 2. Using the SDC_GenerateIncr rule

NOTE: The template SDC file generated by the SDC_GenerateIncr rule is version 2.0. Therefore, you can now add comments in the template CSV file for the following constraints: [create_clock](#), [create_generated_clock](#), [set_false_path](#), [set_multicycle_path](#), [set_max_delay/set_min_delay](#), and [set_clock_groups](#).

Click the following links to explore each step in the process.

- [Specifying Design](#)
- [Specifying the Constraints Specification File](#)
- [Specifying a Seed File](#)
- [Constraint Specific Rule Behavior](#)
- [Customize Generated Files](#)

While generating the output SDC file, an auxiliary SGDC file is also generated which is derived from the SGDC file specified in the SpyGlass run. The SGDC file is created in the same directory where the SDC file is created. The name of the SGDC file is as follows:

```
<existing sgdc filename>_<output sdc file name>.sgdc
```

Use this SGDC file for subsequent SpyGlass runs, where the output SDC is considered as 'seed' constraints.

NOTE: *To generate the SDC file for a block, make sure you specify `block` in the `current_design`.*

Apart from generating the SDC constraints as illustrated in the previous diagram, you can generate Tcl parameters for the constraints in the SDC file. The only difference in the parameterized flow with the one illustrated in the previous diagram is that instead of containing the timing values, the SDC file contains references to the generated Tcl variable, which contains the timing value. For more details, refer to the [Parameterized SDC File](#) section.

Specifying Design

Before running the `SDC_GenerateIncr` rule, you need to read in the design in the Console GUI. The design can be in Verilog or VHDL. While specifying the design, you also need to specify an SGDC file in the Console GUI.

Though the SpyGlass Constraints solution also (including the `SDC_GenerateIncr` rule) works with the `-v/-y` models, it is recommended that for running the `SDC_GenerateIncr` rule, the `.lib` descriptions (in the form of `.sglib`) are provided for library cells instantiated in the design. This is because the SDC generated by the `SDC_GenerateIncr` rule is used by the synthesis/STA tools that use `.lib` (in the form of custom compiled format), rather than the `-v/-y` models. Therefore, in case there are discrepancies in the `-v/-y` models against the corresponding `.lib` models, the SDC generated by the `SDC_GenerateIncr` rule might not work with the `.lib` models in the synthesis/STA tools. This discrepancy is not only limited to the difference in functionality, which is unlikely, but also about `-v/-y` being modeled differently. Therefore, causing certain objects, for example hierarchy and nets/registers, to be visible inside the `-v/-y` model, which are not visible in the `.lib` models.

Specifying the Constraints Specification File

SpyGlass provides a Constraints Specification file `gensdcConstraintsFile.txt`. This file contains a list SDC constraints that you can generate. The [set_case_analysis](#), [create_clock](#), and [create_generated_clock](#) constraints are generated by default. However, you can edit the file to generate more constraints.

It is recommended that you first generate the default constraints and then run the rule. In the second pass, edit the Constraints Specification file by deleting the comments from the constraints you want to generate and then run the rule again. Continue doing this until you have generated all the constraints for your design.

Specifying a Seed File

The seed file implies there exists constraints which are correct. The rule reads these constraints and generates additional constraints as desired by using the Constraints Specification file. The generated constraints are in the output template SDC file.

If you have generated an SDC file for a block, the generated constraints are propagated from top to block. The following constraints are propagated: *create_clock*, *create_generated_clock*, *set_output_delay*, and *set_input_delay*. Constraints are propagated only if they exist in the top.sdc file given as a seed. The *create_clock* and *create_generated_clock* constraints that are generated have the same characteristics as those in the top.sdc file.

The seed file can be an existing SDC or SGDC file. You can:

- *Specify the SGDC file as a Seed File*
- *Specify an SDC file as a seed file*

Specify the SGDC file as a Seed File

The SGDC file is considered as a seed file for a design unit if there are no *sdc_data* constraints with mode *seed* given for the design unit. The SGDC constraints are put into the output template SDC file.

The following seed SGDC constraints are read by the rule:

- *clock*
- *set_case_analysis*

All other constraints are ignored.

The correct identification of all clocks in the SGDC file prior to running the rule is required. The SGDC file should be as follows:

```
current_design <design_name>
clock -name "<design_name>.<clock1_name>" -domain domain1
clock -name "<design_name>.<clock2_name>" -domain domain2
```

The clock period is specified in the SGDC file and this is imported directly into the generated template. The domain specified in the SGDC file is used

to generate [set_false_path](#)/[set_clock_uncertainty](#)/[set_clock_groups](#) between interacting clock pairs, as shown below:

```
current_design <design_name>
clock -name "<design_name>.<clk1_name>" -domain domain1 -
period 10
```

The *SDC_GenerateIncr* rule supports the `-domain` option of the `clock` constraint. If you specify the `-domain` option for at least one clock, the [set_false_path](#)/[set_clock_groups](#)/interclock uncertainty constraints are automatically marked for generation. This is triggered only when the SGDC file is used as seed.

For all interacting clock pairs, if you specify the `-domain` option for both the clocks, and the domain values are the same, the interclock uncertainty is generated. However, if the domain values are different, the [set_false_path](#)/[set_clock_groups](#) constraints are generated. All these pairs are dumped in the crossing file, which you can review and edit. For details on the crossing file, refer to the [Customize Generated Files](#) section.

If you define a clock using the naming convention `<top-name>.<port-name>` in the SGDC file, the *SDC_GenerateIncr* rule creates a [create_clock](#) constraint against the port `<port-name>` and names the clock as `<top-name>_<port-name>` in the SDC file.

The *SDC_GenerateIncr* rule assumes that the SGDC information required for the generation of constraints is present in the SGDC file. Whenever the rule generates a constraint that uses SGDC information, it extracts the information from the SGDC file. In addition, this rule assumes that the SGDC data in the SGDC file remains the same. If the data changes, the generated constraint reflects the changed data. At present, only reset SGDC constraints are being used for the generation of SDC constraints. The SDC constraints using the reset information are [set_max_transition](#) and [set_ideal_net](#).

After the *SDC_GenerateIncr* rule uses the SGDC as a seed, all consequent runs of the rule use the generated SDC file as a seed to generate constraints as per the Constraints Specification file. The flow for this is described in the next section.

Specify an SDC file as a seed file

You specify the seed SDC file using the `-mode seed` argument of the [sdc_data](#) constraint, as shown in the following example:

```
sdc_data -file mySDC.sdc -mode seed
```

The following SDC commands are read by the *SDC_GenerateIncr* rule:

set_case_analysis	create_clock
create_generated_clock	set_input_delay
set_output_delay	set_clock_latency
set_clock_uncertainty	set_clock_transition
set_propagated_clock	set_input_transition
set_driving_cell	set_drive
set_load	set_max_delay/set_min_delay
virtual_clock	set_false_path
set_max_transition	set_clock_groups
set_max_fanout	set_max_capacitance
set_ideal_net	

All other commands in the seed SDC file are ignored.

The rule generates the template specifications for only those SDC commands that are specified with the [gen_sdc_constraints_file](#) rule parameter. This rule creates the template specifications for only those SDC commands for which the specifications are not already available in the seed file.

NOTE: You can specify a CSV file as an input by using the [gen_sdc_csv2sdc_tool](#) parameter. Then, the *SDC_GenerateIncr* rule automatically translates the CSV file to SDC.

Constraint Specific Rule Behavior

The following table describes the behavior of the *SDC_GenerateIncr* rule when you generate specific constraints. Apart from this table, refer to the [Parameter\(s\)](#) section. To generate the constraint, specify the constraint in the Constraints Specification file.

Constraint To Be Generated	Description
<i>set_case_analysis</i>	<p>This rule locates the select pins of all multiplexers by traversing backwards from all unconstrained clock pins. The rule adds the non-constant select pins of those multiplexers that have its data pins constrained by clocks. It generates <i>set_case_analysis</i> at all blackbox terminals, output pins of sequential cells, and ports reachable to these select pins.</p>
<i>create_clock</i>	<p>Clock generation is a multi-step process, as follows:</p> <p>In the first step, the rule locates the ports reachable from the unconstrained clock pins of sequential cells through single-input, single-output (SISO) cells.</p> <p>In the second step, the ports reachable from the remaining unconstrained clock pins of sequential cells are located.</p> <p>If a clock is passing through the input pin of a mux and the fan-out of the mux is reaching a sequential cell clock pin, all input pins of the mux should be constrained through clock signals. The rule traverses backwards from these input pins and identifies the ports reachable. If the clock is passing through the select pin of a mux, a generated clock is put on the output port of the mux.</p>

Constraint To Be Generated	Description
<i>create_generated_clock</i>	<p>This rule locates the sequential output pins reachable from the unconstrained clock pins of the sequential cells. In addition, if a clock is passing through the input pins of a mux and the fan-out of the mux is reaching a sequential cell clock pin, all input pins of the mux should be constrained through clock signals. The rule traverses backwards from these input pins and identifies the sequential pins reachable.</p> <p>This rule generates generated clock on all these sequential output pins. Before generating the clock, see if the two or more output pins of flops can be merged through the fan-out into a single output pin. Then, instead of creating a clock per flip-flop, create one generated clock at the output pin with the source pin as the definition point of the master clock.</p> <p>For every master clock propagating to the existing generated clocks, a corresponding generated clock is created, if it does not exist. If the number of master clocks is more than one, create a generated clock for each master clock.</p>

Constraint To Be Generated	Description
<i>virtual_clock</i>	<p>This rule generates virtual clocks for all primary clocks not mapped to any existing virtual clock. The primary clocks are mapped to virtual clocks through the following methods:</p> <p>Naming Convention A virtual clock matches the real clock if their names satisfy the following structure: <virtual clock name> = <prefix>_<real clock name>_<suffix></p> <p>Clock Characteristics Method The virtual clock matches a real clock if both their period and waveform match.</p> <p>Each virtual clock is mapped to a single real clock. A real clock is first determined through the Naming Convention method and, if none is found, then through the Clock Characteristics method. If a real clock is identified, the virtual clock is mapped to it.</p>
<i>set_input_delay</i>	<p>If the port is constrained with a clock/case analysis constraints, the rule ignores the input port. The rule traverses the path from the port to the data pins of the flops. The input delay is generated on the port for each clock sampling the flops.</p> <p>You can generate I/O delays for virtual clocks. To do this, <i>virtual_clock</i> either needs to be generated or present in the seed SDC file.</p>
<i>set_output_delay</i>	<p>The rule traverses the path from the port to the output pins of the flip-flop. The output delay is generated on the port for each clock sampling the flops.</p> <p>You can generate I/O delays for virtual clocks. To do this, <i>virtual_clock</i> either needs to be generated or present in the seed SDC file.</p>
<i>set_input_delay</i> , <i>set_clock_uncertainty</i>	<p>The rule generates the clock latency and simple clock uncertainty constraints for all clocks, if they do not already exist.</p>

Constraint To Be Generated	Description
<i>set_input_transition</i>	If the ports are constrained by the case analysis constraint, the rule ignores these input and inout ports. For the remaining ports, the rule generates input transition for each port if the design phase is Post-layout. Otherwise, input transition is generated if the port is not constrained by a clock. Hanging ports are ignored.
<i>set_load</i>	If the ports are constrained by the case analysis or clock constraints, the rule ignores these output and inout ports. For the remaining ports, the rule generates load constraints, if they do not already exist. Hanging ports are ignored.
<i>set_max_delay/ set_min_delay</i>	The rule generates min delay/max delay for all port to port combinational paths. The rule ignores ports that are constrained by clocks.
<i>set_driving_cell</i>	The rule generates driving cell constraints for all input or inout ports, if they do not already exist. This rule ignores ports constrained by case analysis or clock constraints. Hanging ports are ignored.
<i>set_drive</i>	The rule generates drive constraints for all input and inout ports, if they do not already exist. This rule ignores ports constrained by case analysis or clock constraints. Hanging ports are ignored.
<i>set_max_transition</i>	<p>The rule generates max transition constraints for the design if the <i>tc_opt01_port_des</i> parameter is set to any valid value other than 'port'. This rule generates max transition constraints for ports if the <i>tc_opt01_port_des</i> parameter is set to any valid value other than 'design'. If the <i>tc_opt01_port_des</i> parameter is set to 'design_or_port', the rule generates max transition constraints for both design and ports.</p> <p>The rule uses 'reset' SGDC constraints in the constraint generation. This rule ignores hanging ports and ports constrained by case analysis, clock, or 'reset'. For all other ports, max transition constraints are generated.</p>

Constraint To Be Generated	Description
<i>set_max_capacitance</i>	<p>The rule generates max capacitance constraints for the design if the <i>tc_opt01_port_des</i> parameter is set to any valid value other than 'port'. This rule generates max capacitance constraints for ports if the <i>tc_opt01_port_des</i> parameter is set to any valid value other than 'design'. If the <i>tc_opt01_port_des</i> parameter is set to 'design_or_port', the rule generates max capacitance constraints for both design and ports.</p> <p>The rule ignores hanging ports and ports constrained by case analysis are ignored. For all other ports, max capacitance constraints are generated</p>
<i>set_max_fanout</i>	<p>The rule generates max fan-out constraints for the design if the <i>tc_opt01_port_des</i> parameter is set to any valid value other than 'port'. This rule generates max fan-out constraints for ports if the <i>tc_opt01_port_des</i> parameter is set to any valid value other than 'design'. If the <i>tc_opt01_port_des</i> parameter is set to 'design_or_port', the rule generates max fan-out constraints for both design and ports.</p> <p>This rule generates max fan-out constraints for all input ports.</p>

Constraint To Be Generated	Description
set_ideal_net	The rule uses 'reset' SGDC constraints in the constraint generation. The constraint is generated for all design objects constrained by real clocks or 'reset' SGDC constraint. For all nets, if their fan-out count is greater than the fan-out limit, which is specified through the tc_fanout_limit parameter, ideal net constraint is generated on the net.
set_false_path/inter clock uncertainty/ set_clock_groups	The rule generates inter-clock uncertainty for all interacting clock pairs having the same root clock. Other interacting unconstrained clock pairs are dumped into crossings file. If clocks are read from the SGDC file and the 'domain' option is defined, the 'domain' option is used to decide the type of constraint for the clock crossing pairs. Inter-clock uncertainty is declared for clock pairs having the same 'domain' option value. Otherwise, false path or clock group is specified. If the 'domain' option is undefined for either of the clocks, '?' is specified in the crossings file. If both false path and clock group are specified in the Constraints Specification file, false path is specified. Refer to the Clock Crossings file section to generate appropriate constraints.

Customize Generated Files

The *SDC_GenerateIncr* rule generates the following files:

■ Clock Crossings file

This file is named `<design-name>-crossings.csv`. It is in a comma-separated data file format. It contains name pairs of interacting but unconstrained clocks in the following format:

```
<clock1-name>, <clock2-name>, ?
```

The following message is appears after the crossing file is generated:

```
Crossings file containing unconstrained Clock
Pairs generated for design/block <name>
```

You need to first modify the clock crossings file so that the CSV file, <design-name>.csv, is updated. Unless you update the clock crossing file, <design-name>-crossings.csv, you cannot edit the <design-name>.csv file in Spreadsheet Viewer. However, you can edit the file after the crossing file has been processed.

Updating the Crossing File Using the Console GUI

To modify the clock crossing file, double-click the message for the generated constraints.

Next, replace the ? character for each pair as shown in the table below:

Replace ? Character with...	To Generate...	Relationship
F	set_false_path	Not applicable
U	set_clock_uncertainty	Not applicable
GA	set_clock_groups	asynchronous
GL	set_clock_groups	logically_exclusive
GP	set_clock_groups	physically_exclusive

NOTE: *If SpyGlass Constraints detects a physically_exclusive relation between a clock-pair crossing, the crossing file is automatically updated with GP.*

After making changes for all clock pairs, save the changes. After you save the changes, Console GUI displays the status of the crossing pairs classification. After classification, the CSV file contains an updated list of [set_false_path](#), [set_clock_uncertainty](#), and/or [set_clock_groups](#) constraints as specified.

After you have modified the crossing file, click the **Update** button to process it.

NOTE: *You cannot edit the crossing file after you have clicked the Update button.*

Set the [gen_c2cVerify](#) parameter to formally verify the generated clock to clock paths. The verified pairs are set as F for false paths and U for uncertainty. The SpyGlass TXV product is invoked to verify the clock-to-clock paths.

Additional generated clocks may be generated during the verification process. You can choose to retain these clocks and the corresponding verified clock-to-clock pair classifications.

The general structure of the crossing file is shown in the following screenshot. The generated clock definitions are followed by formally

Constraints Generation

verified clock-to-clock pair classifications. Following this is the non-verifiable clock-to-clock pairs.

The screenshot shows a spreadsheet with the following content:

	A	B	C
1	#BEGIN Formally generated clocks and timing exception between clock crossing pairs		
2	#Formally generated clocks#		
3	create_generated_clock NG_CLK1 -source clk1 -divide_by 1 -master_clock CLK1 N/B		Y
4	create_generated_clock NG_CLK3 -source clk3 -divide_by 1 -master_clock CLK3 N/A		Y
5	#from Clock		
6	NG_CLK1	NG_CLK3	F
7	NG_CLK3	NG_CLK1	F
8	CLK1	CLK1	F
9	NG_CLK3	NG_CLK3	F
10	CLK2	CLK2	F
11	CLK1	CLK2	F
12	CLK2	CLK1	F
13	CLK1	CLK3	U
14	CLK2	CLK3	U
15	CLK2	CLK3	U
16	CLK3	CLK1	U
17	#END Formally generated clocks and timing exception between clock crossing pairs		
18	CLK1	VCLK_1	U
19	CLK1	VCLK_2	?

Updating the Crossing File in Batch Mode

To update the crossing file in Batch Mode, perform the following steps:

- Open the crossing file using an editor or a spreadsheet viewer.
- Edit the crossing file.
- Use the `update_crossing_file sg_shell Tcl` command to translate the crossing file to CSV format.

Refer to the Tcl Shell Interface User Guide for more information on using the `update_crossing_file` Tcl command.

To convert the CSV file to SDC, see [Updating the CSV File in Batch Mode](#).

NOTE: You can translate the crossing file only once.

For more information, refer to the [Example Code and/or Schematic](#) section.

■ CSV File

This file is created for each design unit. The CSV file contains generated SDC constraints for which you need to specify the timing information. For design objects for which the I/O delay is not generated, a CSV file, called `unconstrained_<block_name>.csv`, is generated. This files

contains the list of such objects and the reason for not generating the I/O delay. The CSV file is displayed as a worksheet with the CSV file containing generated constraints.

Updating the CSV File Using Console GUI

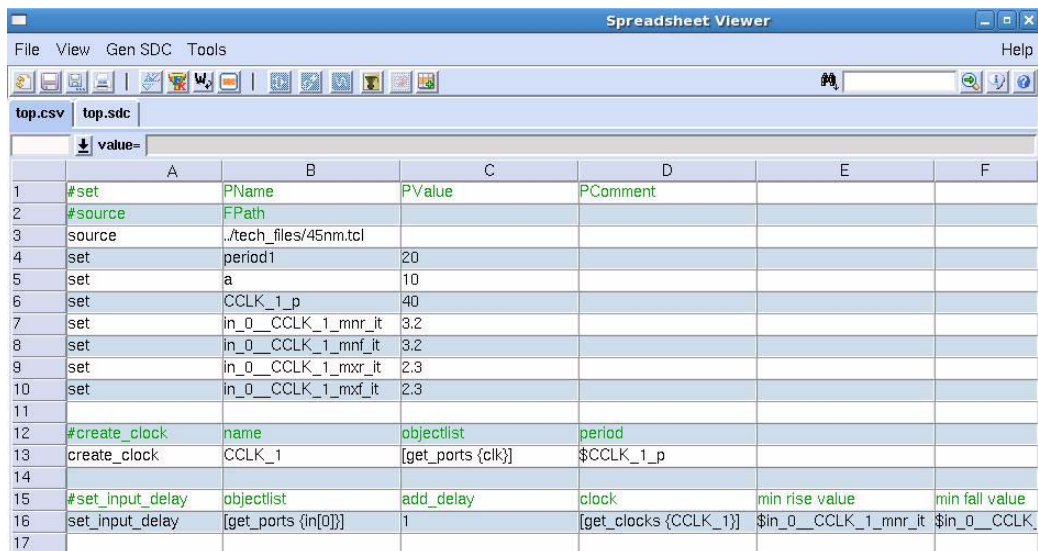
The following message is also generated:

CSV file "<file-path>" generated for importing timing numbers for design/block <name>.

Update CSV and export it to generate SDC.

Click the message for each CSV file to open it the file in the Spreadsheet Viewer window of Console GUI, as shown in the following image. Enter the missing timing information.

Some formulae have been introduced in the spreadsheet that link the cells which require timing information. This makes editing of cells easier. For example, assume two input delay constraints are generated for two input ports (P1 and P2) corresponding to the same clock CLK1 and you want to specify the same value for both the constraints. To do this, you need to specify the value in the top-left cell only. All the corresponding cells are automatically populated with the same value.



	A	B	C	D	E	F
1	#set	PName	PValue	PComment		
2	#source	FPath				
3	source	../tech_files/45nm.tcl				
4	set	period1	20			
5	set	a	10			
6	set	CCLK_1_p	40			
7	set	in_0_CCLK_1_mnr_it	3.2			
8	set	in_0_CCLK_1_mnf_it	3.2			
9	set	in_0_CCLK_1_mxr_it	2.3			
10	set	in_0_CCLK_1_mxf_it	2.3			
11						
12	#create_clock	name	objectlist	period		
13	create_clock	CCLK_1	{get_ports {clk}}	#{CCLK_1_p}		
14						
15	#set_input_delay	objectlist	add_delay	clock	min rise value	min fall value
16	set_input_delay	{get_ports {in[0]}}	1	{get_clocks {CCLK_1}}	#{in_0_CCLK_1_mnr_it}	#{in_0_CCLK_1_mnf_it}
17						

To update the CSV file, wherever applicable, you can enter a space-separated list of numbers (float/integer). However, do not enclose the

list in curly braces { }.

After making the required changes, click **Export SDC** from the **File** menu to generate the SDC file. This opens the **File** dialog box where you specify the name of the SDC file that needs to be generated.

Updating the CSV File in Batch Mode

To update the CSV file in Batch Mode, perform the following steps:

- a. Open the CSV file using an editor or a spreadsheet viewer.
- b. Edit the CSV file.
- c. (Optional) By default, the SDC file is generated in the following directory:

```
spyglass_reports/constraints/exportSDCgen
```

To specify a different directory, make sure you have created the directory in which you want the SDC files to be generated. In addition, ensure you have write permissions to the directory.

- d. Use the `export_sdc sg_shell Tcl` command to generate the SDC files.

Refer to the Tcl Shell Interface User Guide for more information on using the `export_sdc` Tcl command.

For more information, refer to the [Example Code and/or Schematic](#) section.

Parameterized SDC File

Apart from generating the SDC constraints as explained in the previous sections, you can generate a parameterized SDC file. In this section, the following topics are covered:

- [Generating Parameterized Template SDC File](#)
- [Constraints Supported](#)
- [Parameterization Flow](#)
- [Parameter Generation for an SDC Constraint](#)
- [Naming Convention for Parameter Values](#)
- [Generating Comments for Parameters](#)

Generating Parameterized Template SDC File

You can generate the Tcl parameters in a template SDC file. By default, the `SDC_GenerateIncr` rule does not generate parameters for the timing values in the SDC template. To generate parameters, perform the following steps:

1. Set the value of the *tc_enable_param_sdc_flow* parameter to yes.
2. Specify the parameter description (in a specific format) for individual constraints in the SpyGlass Parameter Naming Convention (SPNC) file (see *Naming Convention for the File*) and pass the name of the file as the value to the *gen_sdc_param_file* parameter.

Constraints Supported

Currently, parameter support is provided for the following constraints:

<i>set_case_analysis</i>	<i>create_clock</i>
<i>create_generated_clock</i>	<i>set_input_delay</i>
<i>set_output_delay</i>	<i>set_clock_latency</i>
<i>set_clock_uncertainty</i>	<i>set_clock_transition</i>
<i>set_input_transition</i>	<i>set_driving_cell</i>
<i>set_drive</i>	<i>set_load</i>
<i>virtual_clock</i>	<i>set_max_transition</i>
<i>set_max_capacitance</i>	<i>set_max_fanout</i>
<i>set_max_delay/ set_min_delay</i>	

The parameterization of the *set_load* constraint is implemented via two mnemonics, *pin_load* and *wire_load*. *pin_load* is for specifying the parameterization for pin loads and *wire_load* is for wire loads.

The *-name* fields of the clocks can also be parameterized. The name of the newly generated clocks is then changed according to the specified format. This change is reflected in all appropriate SDC constraints.

Parameterization Flow

The flow of the *SDC_GenerateIncr* rule is not affected with this change. The SDC file, instead of containing the timing values, directly contain a reference to the generated Tcl variable that contains the timing value. Following is the flow of the rule in this case:

1. The SPNC file is parsed. If there is an error in this file, the *SDCPARSE* rule reports a violation indicating the error and exits. When there are multiple errors in the SPNC file, the *SDCPARSE* rule reports a violation for the first error it encounters. The file name and the line number are also displayed for each violation.

2. If the SPNC file is parsed correctly, the *SDC_GenerateIncr* rule runs as it does currently.
3. If the format for a constraint type is specified in the SPNC file, the constraints for that type are processed and the parameters are generated. In addition, the fields displayed in the CSV file depends on the SPNC format. Refer to the [Parameter Generation for an SDC Constraint](#) and [Naming Convention for Parameter Values](#) sections for details.
4. In the .CSV file, the constraints for which the parameters have been generated contain the reference to the parameter instead of the ?. Fill the timing values in the SET section created in CSV file for the generated parameters. The SET section is created for the generated parameters since the generated parameters are specified as set TCL commands in a new SDC file.

After filling the timing values, the template SDC is generated in a file whose name is given by you. All the generated parameters are written into a separate file whose name will be a derivative of the template SDC filename. For example, if the name of an SDC file is `template.sdc`, then `_param` is appended to the file `template.sdc` and the file name changes to `template_param.sdc`.

5. The Parameter SDC file is referred in the template SDC using the `source` command. The constraints for which the parameters have been generated refer to the parameters in their SDC translation.

Parameter Generation for an SDC Constraint

The template SDC contains the constraints of the seed SDC file and the new constraints. Currently, the phases for supported constraints are there in the [set_input_delay](#) and [set_output_delay](#) constraints only.

The SPNC file contains the following naming convention:

```
set_input_delay %objectlist.0.objname_clock.objname_mx -max
set_input_delay %objectlist.0.objname_clock.objname_mn -min
```

The above example means that the existing constraint should be partitioned into the `min` and `max` phase.

The following is the general method of generating parameters for fields of the supported constraint.

For each constraint in the template SDC:

- If the constraint is a new one:

- a. Generate parameters for the fields for which SPNC is defined.
 - b. If SPNC is defined depending on the phases of the constraint, partition the new constraint into the phase combination specified in the SPNC. If phase dependence is not there, there will be a single partition containing all the phases.
 - c. Generate the parameter name from the SPNC template specified for each partition. The fields referred to in the SPNC template are then replaced with their actual values in the constraint to get the parameter name.
- If the constraint is an existing one, generate the parameters for the existing clocks only.

NOTE: For the clock pin of an inferred sequential cell, a parameter `cp_name` with the default value as `CP` is generated. The information about the parameter is present in the `<du-name>.sdc` file.

Naming Convention for Parameter Values

In addition to SPNC for parameter for all constraints, you can specify the SPNC for the parameter's values. For example, the [set_input_delay](#) constraints depend on the clock period. The same can be specified as follows:

```
set_input_delay %objectlist.0.objname_clock.objname_id
set %objectlist.0.objname_clock.objname_id %clock_period/2
```

If the above SPNC format is given for the [set_input_delay](#) constraints, the parameter is generated as explained above. In addition, SET constraints are generated where the generated parameter is set to the values specified in the template. For example, consider the following SDC commands:

```
create_clock -name Clk1 -period 10
set_input_delay ? -clock Clk1 in1
```

Following parameterization will be performed:

```
set $in1_Clk1_id 10/2
create_clock -name Clk1 -period 10
set_input_delay $in1_Clk1_id -clock Clk1 in1
```

NOTE: The generated parameter is set according to the SPNC defined for parameter's value.

You can define multiple value expressions in a single statement by using ternary operator (`?:`) as follows:

```
set %objectlist.0.objname_clock.objname_id %clock_period < 5
? %clock_period/3 : %clock_period/2
```

If the clock period of the delay constraint is less than 5, the parameter value is `%clock_period/3`, otherwise it is `%clock_period/2`.

You can also use the design object's properties in the ternary operator expression. The following properties are defined, at present:

Object direction

- INPUT_OBJ: object direction is input
- OUTPUT_OBJ: object direction is output

Object type

- PORT_OBJ: Object is a port
- TERM_OBJ: object is a terminal
- INST_OBJ: object is a instance
- NET_OBJ: object is a net

You can use the object's properties as follows:

```
set %objectlist.0.objname_clock.objname_id
%objectlist.0.objname_objprop & $PORT_OBJ ? %clock_period/3
: %clock_period/2
```

The above statement sets the value to `%clock_period/3`, if the design object is a port. Otherwise, the value is set to `%clock_period/2`.

The fields to be shown in the `<design-name>.csv` file for a constraint depends upon the format specified in the SPNC file. In the following example, all the fields of the `create_clock` constraint are shown as per the SPNC format:

```
SPNC: create_clock -period %name_p -waveform %name_wv
Display: create_clock,name,objectlist,period,waveform
```

Similarly, if you do not specify a field in the SPNC format, it is not displayed. In the following example, the waveform field is not stated in the SPNC format and therefore the waveform is not displayed.

```
SPNC Purpose: waveform is not specified
SPNC: create_clock -period %name_p
```

```
Display: create_clock,period
```

Generating Comments for Parameters

Though the naming convention for parameters is intuitive, you can generate comments for each parameter by specifying a template containing placeholders for the constraints fields.

For example, in the following SPNC for a clock, a parameter for each clock period field is generated.

```
create_clock -period %name_p
```

For readability, use the `annotate` command to generate comments for a parameter. The syntax of the `annotate` command is:

```
annotate <parameter-format> <comment-template>
```

Where:

- `<parameter-format>`: To identify the parameter for which comment template is specified.
- `<comment-template>`: It is the string that defines the comment. `<comment-template>` starts with '#'. It can contain placeholders to refer to the parameter's associated constraints fields.

In this example, the `annotate` command would be:

```
annotate %name_p # period parameter for clock %name
```

In this example, the 'name' field of `create_clock` constraint is used. Similarly, for other constraints, default `annotate` commands have been added in the default SPNC file. You can adopt them and appropriately customize.

Parameter(s)

- `gen_sdc_csv2sdc_tool`: Default value is none. Specify a CSV file as an input to automatically translate the CSV file to an SDC file.
- `gen_sdc_constraints_file`: Default value is `gensdcConstraintsFile.txt`. The `SDC_GenerateIncr` rule generates the template specifications for only those SDC commands that are specified with this parameter. This rule creates the template specifications for only those SDC commands for which the specifications are not already available in the seed file.

- *ignore_sdc_constraints_file*: Default value is no. Set this parameter to yes to ignore all SDC commands mentioned in the seed SDC file except the *set_case_analysis*, *create_clock*, and *create_generated_clock* commands.
- *tc_bus_merge*: Default value is yes. Set the value to no to prevent the constraints from getting merged on vector ports in the template SDC file.
- *tc_opt01_port_des*: Default value is both. To generate the *set_max_transition*, *set_max_capacitance*, and *set_max_fanout* constraints, set the value of the *tc_opt01_port_des* parameter to either design, port, both, or design_or_port (either port or design).
- *tc_fanout_limit*: Default value is 200. When generating *set_ideal_network* constraints, the *SDC_GenerateIncr* rule considers those high fan-out nets whose fan-out count is greater than the count specified using the *tc_fanout_limit* parameter.
- *tc_allow_async_pins*: Default value is no. When generating the *set_input_delay* constraints, Set the value to yes to make the *SDC_GenerateIncr* rule consider only those pins of a sequential element that are asynchronous with respect to the clocks of the sequential element.
- *tc_enable_param_sdc_flow*: Default is no. Set the value to yes to generate Tcl parameters for the constraints in the SDC template based on the format specified by the *gen_sdc_param_file* parameter. For details, refer to the *Parameterized SDC File* section.
- *gen_sdc_param_file*: Default is gensdcParamSdc.txt. Through this parameter, you specify the file that contains the parameter description. Refer to the *Parameterized SDC File* section for more information.
- *gen_sdc_merge_seed*: Default is no. Set the parameter to yes to generate a merged SDC file that contains seed and generated constraints.
- *gen_c2cVerify*: Default is no. Set this parameter to yes to invoke the SpyGlass TXV product flow for formal verification of clock-to-clock paths.

Constraint(s)

SDC

For constraint specific rule behavior, refer to the [Constraint Specific Rule Behavior](#) section.

SGDC

- [current_design](#) (Mandatory): Use to define SGDC constraints for the design.

Messages and Suggested Fixes

Message 1

When the parameter generation is enabled, following message appears to indicate the file name specified with the [gen_sdc_param_file](#) parameter:

[INFO] Value for parameter gen_sdc_param_file is <filename>

Potential Issues

This violation message is for your information only (Info). There are no potential issues.

Consequences of Not Fixing

As this is an Info message, there are no consequence of not fixing this violation message.

How to Debug and Fix

When you click the message, the file specified by the [gen_sdc_param_file](#) parameter is highlighted. You do not need to take any action.

Message 2

The following message appears to indicate the file name specified with the [gen_sdc_constraints_file](#) parameter:

[INFO] Value for parameter gen_sdc_constraints_file is <file-name>

Potential Issues

This violation message is for your information only (Info). There are no potential issues.

Consequences of Not Fixing

As this is an Info message, there are no consequence of not fixing this

violation message.

How to Debug and Fix

When you click the message, the file specified by the [gen_sdc_constraints_file](#) parameter is highlighted. You do not need to take any action.

Message 3

The following message appears for command `<constr>` in the file specified using the [gen_sdc_constraints_file](#) parameter when the command is not supported by the `SDC_GenerateIncr` rule:

```
[WARNING] Invalid/unsupported command: <constr>
```

Potential Issues

The constraint name stated in the violation message is not supported or is an invalid constraint.

Consequences of Not Fixing

The `SDC_GenerateIncr` rule cannot create the constraints.

How to Debug and Fix

The file specified by the [gen_sdc_constraints_file](#) parameter is highlighted showing the constraint name. Refer to the [Constraints Supported](#) for a list of supported constraints in the parameterized flow. Alternatively, check for typo errors.

Message 4

The following message appears when the `SDC_GenerateIncr` rule is reading a seed SDC file for design/block `<name>`:

```
[INFO] Reading initial constraints from seed sdc for design/  
block <name>
```

Potential Issues

This violation message is for your information only (Info). There are no potential issues.

Consequences of Not Fixing

As this is an Info message, there are no consequence of not fixing this violation message.

How to Debug and Fix

This message informs you of the source of the seed constraints. In this case, it is the SDC file. You do not need to take any action.

Message 5

The following message appears when the seed SDC file for design/block <name> does not contain any *create_clock/create_generated_clock* commands:

[INFO] seed sdc for design/block <name> does not contain any clocks

Potential Issues

There are no clocks defined in the seed SDC file. The *SDC_GenerateIncr* rule first checks for *set_case_analysis*, *create_clock*, and *create_generated_clock* constraints.

Consequences of Not Fixing

You cannot generate SDC constraints for your design until the *SDC_GenerateIncr* rule correctly identifies all clocks in your design.

How to Debug and Fix

This message informs you that there are no clocks specified in the seed SDC constraints. Define clocks in the seed SDC file by specifying the *create_clock* and *create_generated_clock* constraints.

Message 6

The following message appears when the *SDC_GenerateIncr* rule is reading a seed SGDC file for design/block <name>:

[INFO] Reading initial constraints from sgdc file for design/block <name>

Potential Issues

This violation message is for your information only (Info). There are no potential issues.

Consequences of Not Fixing

As this is an Info message, there are no consequence of not fixing this violation message.

How to Debug and Fix

This message informs you of the source of the seed constraints. In this case, it is the SGDC file. You do not need to take any action.

Message 7

The following message appears when the seed SGDC file for design/block <name> does not contain any clock constraints:

```
[INFO] sgdc for design/block <name> does not contain any clocks
```

Potential Issues

There are no clocks defined in the seed SGDC file. The *SDC_GenerateIncr* rule first checks for [set_case_analysis](#), [create_clock](#), and [create_generated_clock](#) constraints

Consequences of Not Fixing

You cannot generate SDC constraints for your design until the *SDC_GenerateIncr* rule correctly identifies all clocks in your design.

How to Debug and Fix

This message informs you that there are no clocks specified in the seed SGDC constraints. You can either define clocks by specifying the [clock](#) constraint in the seed SGDC file. Alternatively, you can define the clocks in the SDC file by using the [create_clock](#) and [create_generated_clock](#) constraints.

For details, refer to the [Specifying a Seed File](#) section.

Message 8

The following message appears if the naming convention file contains the format of an unknown SDC constraint <constr-name>:

```
[WARNING] Unknown constraints specified in spnc: <constr-name>
```

Potential Issues

The constraint specified in the SpyGlass Parameter Naming Convention (SPNC) file is not supported or is invalid.

Consequences of Not Fixing

You cannot generate constraints

How to Debug and Fix

The parameter conventions are specified in the SPNC file for an unsupported or invalid constraint. Review the SPNC file as highlighted in the Console GUI.

For details on the parameterized flow for generating SDC constraints, refer to the [Parameterized SDC File](#) section.

Message 9

The following message appears when invalid characters are used while specifying the parameters in the SPNC file *<file-name>*:

[WARNING] Invalid characters specified in the Parameter name format: *<parameter-name>*

Potential Issues

The SPNC file contains invalid characters.

Consequences of Not Fixing

The SPNC file cannot be read and there the Tcl parameters for the constraints in the SDC file cannot be generated.

How to Debug and Fix

The parameter name format contains invalid characters. Review the SPNC file as highlighted in the Console GUI. The valid characters that can be used while specifying the parameter format are: A-Z, a-z, 0-9, and %.

For details on the parameterized flow for generating SDC constraints, refer to the [Parameterized SDC File](#) section.

For details on the SpyGlass Parameter Naming Convention (SPNC) file, refer to [Naming Convention for the File](#).

Message 10

The following message appears when an unknown field identifier *<field-identifier-name>* is used for a constraint in the SPNC file:

[WARNING] Unknown Field Identifier: *<field-identifier-name>*

Potential Issues

The SPNC file contains an unknown field identifier.

Consequences of Not Fixing

The SPNC file cannot be read and there the Tcl parameters for the constraints in the SDC file cannot be generated.

How to Debug and Fix

The SPNC specification for a constraint contains an unsupported field identifier. Review the SPNC file as highlighted in the Console GUI.

For details on the SpyGlass Parameter Naming Convention (SPNC) file, refer to [Naming Convention for the File](#).

For details on the parameterized flow for generating SDC constraints, refer to the [Parameterized SDC File](#) section.

Message 11

The following message appears when a `field_identifier` is specified in an incorrect location in the SPNC file:

```
[WARNING] Source Field Identifier mentioned in SPNC:  
<field_identifier>
```

Potential Issues

The *source* field identifier is specified at a place other than the required place in the SPNC file for an SDC constraint.

Consequences of Not Fixing

The SPNC file cannot be read and there the Tcl parameters for the constraints in the SDC file cannot be generated.

How to Debug and Fix

Review the SPNC file as highlighted in the Console GUI for source field identifiers. Source field identifiers should not be mentioned in the SPNC specification for a constraint.

For details on the SpyGlass Parameter Naming Convention (SPNC) file, refer to [Naming Convention for the File](#).

For details on the parameterized flow for generating SDC constraints, refer to the [Parameterized SDC File](#) section.

Message 12

The following message appears when a non-source field is mentioned in the parameter naming format in the SPNC file `<file-name>`:

```
[WARNING] Non source field referenced: <file-name>
```

Potential Issues

A non-source field is mentioned in the parameter naming format in the SPNC file.

Consequences of Not Fixing

The SPNC file cannot be read and there the Tcl parameters for the constraints in the SDC file cannot be generated.

How to Debug and Fix

Review the SPNC file as highlighted in the Console GUI for a parameter name specification that refers to a non-source field of the corresponding constraint.

For details on the SpyGlass Parameter Naming Convention (SPNC) file, refer to [Naming Convention for the File](#).

For details on the parameterized flow for generating SDC constraints, refer to the [Parameterized SDC File](#) section.

Message 13

The following message appears if, for the constraint `<constr-name>`, all the phases are not specified in the SPNC file:

```
[WARNING] Not all phases for <constr-name> have been defined in SPNC
```

Potential Issues

In the SPNC file, you have not specified all phases for the constraint mentioned in the violation message.

Consequences of Not Fixing

The SPNC file cannot be read and therefore the Tcl parameters for the constraints in the SDC file cannot be generated.

How to Debug and Fix

All phases, such as `min`, `max`, `rise`, and `fall`, for a constraint should be defined in the SPNC. The SPNC file is highlighted in the Console GUI. Edit the file so that all phases are defined for the constraint mentioned in the violation message.

For details on the SpyGlass Parameter Naming Convention (SPNC) file, refer to [Naming Convention for the File](#).

For details on the parameterized flow for generating SDC constraints, refer to the [Parameterized SDC File](#) section.

Message 14

The following message appears if the naming convention for a phase of a constraint is specified more than once:

```
[WARNING] Phase fields mentioned twice in the same spnc:  
<field_identifier>
```

Potential Issues

The phase field stated in the violation message is mentioned twice in the same SPNC specification for a constraint.

Consequences of Not Fixing

The SPNC file cannot be read and there the Tcl parameters for the constraints in the SDC file cannot be generated.

How to Debug and Fix

Edit the SPNC file as highlighted in the Console GUI so that the phase field is mentioned once.

For details on the SpyGlass Parameter Naming Convention (SPNC) file, refer to [Naming Convention for the File](#).

For details on the parameterized flow for generating SDC constraints, refer to the [Parameterized SDC File](#) section.

Message 15

The following message appears if no crossing information is specified:

```
[INFO] No unconstrained clock pairs found for design/block top
```

Potential Issues

This violation message is for your information only (Info). There are no potential issues.

Consequences of Not Fixing

As this is an Info message, there are no consequence of not fixing this violation message.

How to Debug and Fix

You do not need to take any action.

Message 16

The following message appears when reading the seed constraints from the provided SGDC file, the period and waveform information is not specified:

```
[WARNING] virtual clock is enabled for generation, but period information for SGDC clocks is missing
```

Potential Issues

The message is given when the virtual clock is enabled for generation. The seed constraints are read from the SGDC file. No period information for the

clocks mentioned in the SGDC file is provided. Period information is required for the generation of virtual clocks.

Consequences of Not Fixing

A single virtual clock is generated while you may want to generate a virtual clock for each real clock.

How to Debug and Fix

Enter the period waveform information for each real clock.

Message 17

The following message appears when you specify a CSV file as an input via the *gen_sdc_csv2sdc_tool* parameter:

```
[INFO] SDC file translated from the CSV file given via  
parameter gen_sdc_csv2sdc_tool
```

Potential Issues

This violation message is for your information only (Info). There are no potential issues.

Consequences of Not Fixing

As this is an Info message, there are no consequence of not fixing this violation message.

How to Debug and Fix

This message informs you when the translation of the CSV constraints into the SDC format is successful. The SDC file that contains the translated constraints is highlighted in the Console GUI.

Message 18

The following message appears when you specify an SDC command that is not supported by the *SDC_GenerateIncr* rule:

```
[WARNING] Unsupported command <command-name> specified in  
constraint file <file-name>
```

Potential Issues

The constraint name stated in the violation message is not a supported constraint.

Consequences of Not Fixing

The *SDC_GenerateIncr* rule cannot create the constraints.

How to Debug and Fix

Refer to the [Specify an SDC file as a seed file](#) section for a list of supported constraints. Edit the constraints file to ensure all constraints mentioned are supported.

Message 19

If you specify a command that is neither supported by the *SDC_GenerateIncr* rule nor understood by SpyGlass, the following message appears:

```
[WARNING] Invalid command <command-name> specified in
constraint file <file-name>
```

Potential Issues

The constraint name stated in the violation message is not a valid constraint.

Consequences of Not Fixing

The *SDC_GenerateIncr* rule cannot create the constraints.

How to Debug and Fix

Edit the constraints file to ensure all constraints mentioned are supported.

For a list of supported constraints, refer to the [Specify an SDC file as a seed file](#) section.

Example Code and/or Schematic**Example 1**[View Test Case Files](#)

This example shows how to generate an SDC file using this rule.

To view the example, see [View Test Case Files](#).

Example 2

Test Case Files Not Available

Generating Constraints Using the sg_shell Batch Flow

This example shows how to use the `update_crossing_file` and `export_sdc` Tcl commands to generate SDC constraints. Invoke the

sg_shell and enter the following commands.

```
sg_shell> open_project test.prj
current_methodology: info: methodology is now `~/delsoft/spyint/integration/5.4.0-Beta-C8/RELEASE/SpyGlass-5.4.0/SPYGLASS_HOME/GuideWare2.0/block/rtl_handoff'
current_goal: info: loading goal `test_goal' (in progress)
Reading sgdc file "constraints.sgdc" ...
current_goal: info: restoring synthesized design view ...
current_goal: warning: design database not present for
currently selected goal, design query would not work
(please perform `run_goal' to create the design database)
current_goal: info: finished loading goal `test_goal' (ok)
sg_shell> run_goal test_goal
```

After the rule is run, the crossing and CSV files are generated. First, update the crossing file, if generated, by using a VI editor. The following is a sample crossing file.

```
1 [ ]CLK_1,CCLK_2,?
```

To define the relationship between the following clock crossing pair, update the crossing file by either replacing the ? mark with:

- **F** for defining false path
- **U** for defining clock uncertainty
- **GA, GP, GL** for set_clock_groups

The following shows the updated crossing file for defining false path between the clock crossing pair.

```
. [ ]CLK_1,CCLK_2,F
```

Save the file and execute the `update_crossing_file` Tcl command to translate this crossing file into the CSV file.


```
sg_shell> update_crossing_file
```

```
0 pair(s) as classified in crossing file ./test/test_goal/
spyglass_reports/constraints/Block1-crossings.csv have been
updated successfully.
```

```
1 pair(s) as classified in crossing file ./test/test_goal/
spyglass_reports/constraints/top-crossings.csv have been
updated successfully.
```

The CSV file is updated with the crossing pair.

Open the CSV file in a vi editor and replace the ? mark with the relevant values.

The following CSV file is a sample CSV file.

```
#create_clock#,name,add,objectlist,period,waveform,comment
create_clock,CCLK_2,,[get_ports {clk2}],?,?,

#simple_clock_uncertainty#,objectlist,value
simple_clock_uncertainty,[get_clocks {CCLK_1}],?
simple_clock_uncertainty,[get_clocks {CCLK_2}],=C5

#set_max_transition#,objectlist,value
set_max_transition,[get_ports {in1}],?
set_max_transition,[get_ports {in3}],=C9
set_max_transition,[get_ports {out1}],=C10
set_max_transition,[get_ports {out2}],=C11
set_max_transition,[get_designs {top}],=C12

#clock_max_transition#,objectlist,value
clock_max_transition,[get_clocks {CCLK_1}],?
clock_max_transition,[get_clocks {CCLK_2}],=C16

#set_max_capacitance#,objectlist,value
set_max_capacitance,[get_ports {in1}],?
set_max_capacitance,[get_ports {in3}],=C20
set_max_capacitance,[get_ports {out1}],=C21
set_max_capacitance,[get_ports {out2}],=C22
set_max_capacitance,[get_designs {top}],=C23

#clock_max_capacitance#,objectlist,value
clock_max_capacitance,[get_clocks {CCLK_1}],?
clock_max_capacitance,[get_clocks {CCLK_2}],=C27
```

The updated CSV file, which has the crossing pair (See highlighted section), is shown in the following screen grab.

```
#create_clock#,name,add,objectlist,period,waveform,comment
create_clock,CCLK_2,,[get_ports {clk2}],10,0 5,

#simple_clock_uncertainty#,objectlist,value
simple_clock_uncertainty,[get_clocks {CCLK_1}],1
simple_clock_uncertainty,[get_clocks {CCLK_2}],2

#set_max_transition#,objectlist,value
set_max_transition,[get_ports {in1}],3
set_max_transition,[get_ports {in3}],4
set_max_transition,[get_ports {out1}],5
set_max_transition,[get_ports {out2}],6
set_max_transition,[get_designs {top}],7

#clock_max_transition#,objectlist,value
clock_max_transition,[get_clocks {CCLK_1}],8
clock_max_transition,[get_clocks {CCLK_2}],8

#set_max_capacitance#,objectlist,value
set_max_capacitance,[get_ports {in1}],9
set_max_capacitance,[get_ports {in3}],9
set_max_capacitance,[get_ports {out1}],9
set_max_capacitance,[get_ports {out2}],9
set_max_capacitance,[get_designs {top}],9

#clock_max_capacitance#,objectlist,value
clock_max_capacitance,[get_clocks {CCLK_1}],1
clock_max_capacitance,[get_clocks {CCLK_2}],1

#set_false_path#,setup rise constr,setup fall constr,hold rise constr,hold fall constr,from,to,through_list,comment
set_false_path,1,1,1,1,[get_clocks {CCLK_1}],,[get_clocks {CCLK_2}],,,
```

To generate the SDC file from the updated CSV file, use the `export_sdc` Tcl command.

```
sg_shell> export_sdc
```

```
Block1.csv is successfully translated to ./test/test_goal/
spyglass_reports/constraints/exportSDCgen/Block1.sdc
```

```
top.csv is successfully translated to ./test/test_goal/
spyglass_reports/constraints/exportSDCgen/top.sdc
```

The generated SDC file is shown in the following screen grab.

Constraints Generation

```
#####
# Created by SpyGlass SDC GenerateIncr on Thu Nov 27 13:52:55 2014
#####

# BEGIN PRIMARY CLOCK
create_clock -name CCLK_2 -period 10 -waveform { 0 5} [get_ports {clk2}]
# END PRIMARY CLOCK

# BEGIN SIMPLE CLOCK UNCERTAINTY
set_clock_uncertainty 1 [get_clocks {CCLK_1}]
set_clock_uncertainty 2 [get_clocks {CCLK_2}]
# END SIMPLE CLOCK UNCERTAINTY

# BEGIN MAX TRANSITION
set_max_transition 3 [get_ports {in1}]
set_max_transition 4 [get_ports {in3}]
set_max_transition 5 [get_ports {out1}]
set_max_transition 6 [get_ports {out2}]
set_max_transition 7 [get_designs {top}]
# END MAX TRANSITION

# BEGIN CLOCK MAX TRANSITION
set_max_transition 8 [get_clocks {CCLK_1}]
set_max_transition 8 [get_clocks {CCLK_2}]
# END CLOCK MAX TRANSITION

# BEGIN MAX CAPACITANCE
set_max_capacitance 9 [get_ports {in1}]
set_max_capacitance 9 [get_ports {in3}]
set_max_capacitance 9 [get_ports {out1}]
set_max_capacitance 9 [get_ports {out2}]
set_max_capacitance 9 [get_designs {top}]
# END MAX CAPACITANCE

# BEGIN CLOCK MAX CAPACITANCE
set_max_capacitance 1 [get_clocks {CCLK_1}]
set_max_capacitance 1 [get_clocks {CCLK_2}]
# END CLOCK MAX CAPACITANCE

# BEGIN FALSE PATH
set_false_path -from [get_clocks {CCLK_1}] -to [get_clocks {CCLK_2}]
# END FALSE PATH
```

Default Severity Label

Info

Rule Group

Others

Report and Related Files

- SPNC: [Naming Convention for the File](#)

Reporting Rules

The Reporting Rules Group `SDC_Report` contains the following rules:

Rule	Description
<i>SDC_Report01</i>	Prints a table of source clocks and their generated clocks corresponding to primary ports
<i>SDC_Report03</i>	Reports design units with the <i>set_dont_touch</i> command set
<i>SDC_Report04</i>	Reports library objects with the <i>set_dont_use</i> command set
<i>Cons_SDC_Report</i>	Generates Consolidated reports for SDC constraints

SDC_Report01

Prints a table of generated clocks versus source clocks

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_Report01* rule prints the information about clocks (*create_clock*, *create_generated_clock*, and virtual clock), primary ports (input/inout/output) and their relationship in the *tc_clock_info Report*.

If multiple clocks are defined on the same object using the `add` argument of the *create_clock/create_generated_clock* constraint, the *SDC_Report01* rule retains only the last clock defined without the `add` argument and any clocks defined with the `add` argument after this (last) clock.

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

None

Default Severity Label

Warning

Rule Group

SDC_Report

Reports and Related Files

- [tc_clock_info Report](#)

SDC_Report03

Prints the list of dont touch module

When to Use

Use to identify the modules that were not taken through optimizations. This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_Report03* rule reports design units with the [set_dont_touch](#) command set.

Parameter(s)

None

Constraint(s)

SDC

- [set_dont_touch](#) (Mandatory): Use to specify a list of cells, nets, references, designs, and library cells on which the `dont_touch` attribute is to be set.

Messages and Suggested Fix

None

Example Code and/or Schematic

Refer to [tc_dont_touch_info Report](#) for a sample report.

Default Severity Label

Warning

Rule Group

SDC_Report

Reports and Related Files

[tc_dont_touch_info Report](#): Contains the table of design units with the

set_dont_touch command set along with the SDC file name and line number.

SDC_Report04

Reports instances of the `dont_use` attribute set on a design object

When to Use

Use to identify the modules that were not taken through optimizations. This rule is applicable to the RTL phase.

Description

The *SDC_Report04* rule reports those library cells and modules where either the `dont_use` library attribute or a *set_dont_use* constraint is set.

Parameter(s)

None

Constraint(s)

SGDC

- *set_dont_use* (Mandatory)

Messages and Suggested Fix

Message 1

The following message appears where a *set_dont_use* command is specified on cell `<cell-name>` of library `<lib-name>` that is instantiated in design/block `<name>`:

[WARNING] Design/block `<name>` has `set_dont_use` specify for library cell `<cell-name>` (library: `<lib-name>`)

Message 2

The following message appears where cell `<cell-name>` of library `<lib-name>` with the `dont_use` library attribute set, is instantiated as `<inst-name>` in design/block `<name>`:

[WARNING] Design/block `<name>` has `dont_use` attribute set for instance `<inst-name>` through technology library `<lib-name>` (cell: `<cell-name>`)

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

Certain cells or modules are not recommended to use for a particular design. Those are characterized by the `dont_use` attributes or constraints. If they are still used in the design, you should review the reasons for recommending them as `dont_use`.

How to Debug and Fix

Review the reasons for specifying the `dont_use` library attribute or a [set_dont_use](#) constraint and update as required.

Example Code and/or Schematic

For the following snippet, the *SDC_Report04* rule reports a violation because the instance of AN2 is being used in design since AN2 has the `dont_use` attribute in `test.lib`.

```
//test.lib
cell(AN2) {
    area : 2;
    dont_use : true;
.
.
}

//test.v
Module top(input a,b, output c);
AN2 I1(.A(a), .B(b), .Z(c));
Endmodule
```

Default Severity Label

Warning

Rule Group

SDC_Report

Reports and Related Files

- *tc_dont_use_info Report*: Lists those library cells and modules where either the dont_use library attribute is set or a set_dont_use command is set.

Cons_SDC_Report

Generates Consolidated reports for SDC constraints

When to Use

Use to generate a consolidated report for timing analysis. This rule is applicable all design phases.

Description

The *Cons_SDC_Report* rule generates a consolidated report, which provides detailed timing analysis information, for each SDC constraint specification of the following type:

- set_case_analysis
- Clocks
- I/O Delays
- Timing Exceptions

Prerequisites

The following table lists the SpyGlass Constraints rules that are run to generate the Consolidated Report:

Constraint Type	Rules Run
set_case_analysis	NA
Clock	<i>Clk_Gen02, Clk_Gen03, Clk_Gen05, Clk_Gen22, Clk_Gen23, Clk_Gen24, Clk_Gen34, Const_Struct04a, Clk_Lat02, Clk_Lat03, Clk_Lat06, Clk_Uncert02b, DomainError</i>
Unconstrained Design	<i>Clk_Gen33</i>
set_input_delay	<i>Const_Struct04a, Inp_Del02, Inp_Del03, Inp_Del09</i>
set_output_delay	<i>Const_Struct04a, Op_Del02, Op_Del03, Op_Del09</i>
Timing Exceptions	<i>False_Path01, False_Path03, False_Path12, MCP01, MCP03, MCP05, MCP09, TE_Conflict01, Combo_Paths04, TE_Consis01, TE_Consis02</i>

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

The following message appears when the report is generated for a block <block-name>:

[INFO] Consolidated report generated for block <block-name>, schema line num : <line-num>

Potential Issues

Refer to the generated report to perform detailed timing analysis.

Consequences of Not Fixing

Not applicable

How to Debug and Fix

Double-click the violation message to view the consolidated report in the Spreadsheet Viewer. The consolidated report contains a separate tab for each constraint. In addition, it contains a summary report that displays the issues detected for each type of constraint.

You can begin to debug, by viewing the summary report. When a constraint has an issue, it appears in a red font, and back references to the more information that can help you debug and resolve the issue.

For more details on the generated report, refer to [Timing Analysis Consolidated Report](#).

Example Code and/or Schematic

By default, the Summary Report is displayed in the Schematic Viewer window. The **Header** section displays the rules that were run to generate the report. In this example, you can see that each of the Constraint type has an issue. For example, the **Clock** constraint has **16** issues, while the input delay constraint has **135** issues.

The screenshot shows the 'Spreadsheet Viewer' application window. The title bar reads 'Spreadsheet Viewer'. The menu bar includes 'File', 'View', 'Tools', and 'Help'. Below the menu bar is a toolbar with various icons. The main area displays a list of spreadsheets: 'consSummarytest2.csv', 'caseAnalysistest.csv', 'clockIssuetest2.csv', and 'genClocktest2'. The 'consSummarytest2.csv' spreadsheet is selected. Below the spreadsheet list, there is a 'Show Header' checkbox which is checked. A search bar labeled 'value=' is present. The main data table has three columns: A (Constraint Type), B (Total Constraints), and C (Issues Count). The table contains six rows of data. At the bottom of the window, a status bar shows 'Messages: Displayed: 6 Total: 6'.

	A	B	C
	Constraint Type	Total Constraints	Issues Count
1	set_case_analysis	13	N.A.
2	Clock	16	16
3	Unconstrained Design	-	2
4	set_input_delay	262	135
5	set_output_delay	230	114
6	Timing Exceptions	356	207

If you want to further analyze any of the constraint, click the appropriate number in the **Issues Count** column. The related worksheet appears. In this example, to explore the issues with the **Clock** constraint, click **16**. Spreadsheet Viewer displays the details of the issues detected, as shown.

Reporting Rules

Spreadsheet Viewer

File View Tools Help

Spreadsheets (9): consSummarytest2.csv

consSummarytest2.csv caseAnalysistest.csv clockIssuetest2.csv genClocktest2.csv clocktest2.csv UnConsClocksPintu

▼ Show Header

-1 value=

	A	B	C	D	E	F	G
	Clock Name	Clock Type	Polarity	Mode	Domain Name(Conflicting	Other clocks with same n	Issues
1	clk0	Primary	positive	A	A1()	0	1
2	clk1	Primary	positive	A	A1()	0	1
3	clk2	Primary	positive	A	A1()	0	1
4	clk3	Primary	positive	A	A1()	0	1
5	clk4	Primary	positive	A	A1()	0	1
6	clk5	Primary	positive	A	A1()	0	1
7	clk6	Primary	positive	A	A1()	0	1
8	clk7	Primary	positive	A	A1()	0	1
9	genclk0_reg/Q	Generated	positive	A	A1()	0	1
10	genclk1_reg/Q	Generated	positive	A	A1()	0	1
11	genclk2_reg/Q	Generated	positive	A	A1()	0	1
12	genclk3_reg/Q	Generated	positive	A	A1()	0	1
13	genclk4_reg/Q	Generated	positive	A	A1()	0	1
14	genclk5_reg/Q	Generated	positive	A	A1()	0	1
15	genclk6_reg/Q	Generated	positive	A	A1()	0	1
16	genclk7_reg/Q	Generated	positive	A	A1()	0	1

Messages: Displayed: 16 Total: 16

Similarly, you can analyze and resolve issues for each constraint type.

Default Severity Label

Info

Rule Group

SDC_Report

Reports and Related Files

Timing Analysis Consolidated Report

Setup Rules

The Setup Rules Group `Prereqs` contains the following rules:

Rule	Description
<i>Const_Sanity_Rule</i>	Checks for correct environment settings
<i>DomainSanityCheck</i>	Incorrect <i>clock_group</i> constraint specifications
<i>ParamSanityCheck01</i>	Runs <i>ParamSanityCheck01a</i> and <i>ParamSanityCheck01b</i> rules
<i>ParamSanityCheck01a</i>	Performs syntactic checks on user-specified rule parameters
<i>ParamSanityCheck01b</i>	Performs post-synthesis syntactic checks on user-specified rule parameters
<i>SDCPARSE</i>	Reads the SDC and SGDC files

Const_Sanity_Rule

Checks whether SDC or SGDC files are accessible or not

When to Use

This rule is applicable to the RTL, pre-layout, and post-layout phases.

Description

The *Const_Sanity_Rule* rule checks whether:

- The SpyGlass Design Constraints (SGDC) file has been specified.
- The SDC file is located at the expected location as specified in the SGDC file.

SpyGlass exits when any of the above conditions are not met.

Parameter(s)

None

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message appears:

[FATAL] No sgdc file specified

Potential Issues

The violation message appears if the SpyGlass Design Constraints (SGDC) file has not been specified.

Consequences of Not Fixing

SpyGlass rule checking is performed only when the SGDC file is specified. If the issue is not fixed, SpyGlass does not check any rules of the SpyGlass Constraints solution.

How to Debug and Fix

You can specify the SGDC file by entering the following in the project file:

```
read_file -type sgdc <file-name>
```

In addition, ensure the file is accessible.

Message 2

The following message appears:

```
[FATAL] <sdc-file-name>: either doesn't exist or couldn't be
open for reading
```

Potential Issues

The violation message appears if the SDC file is not found at the expected location or could not be opened.

Consequences of Not Fixing

The rules that require the SDC file do not perform any checking.

How to Debug and Fix

Check whether SDC file exists on the specified location or not. If not, modify the SGDC file to reflect the correct path. Alternatively, check whether the file is accessible. If not, ensure there is read-permission specified to that file.

Example Code and/or Schematic

Example 1

Consider the case when `top.sgdc` does not exist in the current working directory (CWD), but it is specified in the following:

```
read_file -type sgdc top.sgdc
```

This rule reports a FATAL violation message for the `top.sgdc` file because it is not present in the CWD.

Example 2

Consider the case when `top.sdc` does not exist in the CWD, but it is specified in the `top.sgdc` file.

```
top.sgdc:
```

```
current_design top
```

```
sdc_data -file top.sdc -mode functional
```

Since `top.sdc` does not exist in the CWD, the *Const_Sanity_Rule* rule

reports a FATAL violation message.

Default Severity Label

Fatal

Rule Group

Prereqs

Reports and Related Files

None

DomainSanityCheck

Reads the `clock_group` information from SGDC files

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *DomainSanityCheck* rule reads the `clock_group` constraint information from SGDC files and performs the following sanity checks:

- All the clocks specified with the `clock` argument exist in the SDC file
- Some clock is defined on the pins specified with the `clock_pin` argument
- Same clock appears in more than one `clock_group` constraint
- The `clock_group` constraint is not specified in the allowed format
- If one clock is generated from another and there exists a crossing between the two clocks.

Prerequisite(s)

The `clock_group` constraint should be specified in the SGDC file.

Parameter(s)

None

Constraint(s)

- `clock_group` (Mandatory): Use this constraint to specify the clock relationship (domain, synchronous/asynchronous, exclusive) in the SGDC file.

Messages and Suggested Fix

Message 1

The following message appears when a clock `<clk-name>` specified with the `-clock` argument does not exist in the SDC file:

```
[FATAL] Clock "<clk-name>" specified with -clock option does not exist
```

Message 2

The following message appears when no clock is defined on a pin `<pin-name>` that has been specified with the `clock_pin` argument:

```
[FATAL] No clock defined on pin "<pin-name>", specified with clock_pin option
```

Message 3

The following message appears when the object `<obj-name>` specified with the `clock_group` constraint is not specified as a clock in the SDC file:

```
[FATAL] Clock "<obj-name>" does not exist
```

Message 4

The following message appears when a clock `<clk-name>` is specified in two `clock_group` constraints `<clock_group1-name>` and `<clock_group2-name>`:

```
[FATAL] Clock "<clk-name>" specified in clock_group "<clock_group1-name>" and "<clock_group2-name>"
```

Message 5

The following message appears when a `clock_group` constraint `<clock_group-name>` is not specified in the prescribed format:

```
[FATAL] clock_group <clock_group-name> not specified in the allowed format
```

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

If you do not fix these violations, SpyGlass will not run any other rules.

How to Debug and Fix

Message 1: Ensure that you have specified a valid clock.

Message 2: Ensure that you have specified a valid clock pin.

Message 3: Ensure that you have specified a valid clock.

Message 4: Ensure that a clock is specified in only one group, remove this clock from either one.

Message 5: Ensure that the `clock_group` constraint is specified in correct

format.

Example Code and/or Schematic

Example 1

Suppose there are three clocks, C1, C2, and C3, in the design. The *clock_group* constraints are specified as follows in the SGDC file:

```
clock_group -name G1 -clock {C1 C2}
clock_group -name G2 -clock {C1 C3}
```

Here, the clocks C1 is specified in both the groups, hence the rule reports a FATAL violation for this clock.

Example 2

The *clock_group* constraint is specified for a clock which does not exist.

```
clock_group -name G1 -clock {C4 C5}
```

Here, clocks C4 and C5 does not exist, hence the rule reports a FATAL violation for these clocks.

```
clock_group -name G2 -clock_pin {ck}
```

The port/pin ck does not exist in the design, hence the rule reports a FATAL violation for this constraint.

```
clock_group -name G3 -clock_pin {ck1}
```

The port/pin ck1 exists in the design, but no clock is defined on this pin, hence the rule reports a FATAL violation for this constraint.

Example 3

For the following snippet, the *DomainSanityCheck* rule reports a FATAL violation because the `-clk` argument is not valid for the *clock_group* constraint.

```
clock_group -name G1 -clk {C1}
```

Default Severity Label

Fatal

Rule Group

Prereqs

Reports and Related Files

None

ParamSanityCheck01

Performs checks on user-specified rule parameters

The *ParamSanityCheck01* rule runs the *ParamSanityCheck01a* and *ParamSanityCheck01b* rules.

ParamSanityCheck01a

Performs syntactic checks on user-specified rule parameters

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases and to perform sanity checks on rule parameters.

Description

The *ParamSanityCheck01a* rule performs parsing check in the SNPC file.

Parameter(s)

The *ParamSanityCheck01a* rule checks the following rule parameters:

Rule Parameter	Check whether the specified value is...
<i>default_max_capacitance</i>	A positive floating-point number
<i>default_min_capacitance</i>	A positive floating-point number that is less than the value of the <i>default_max_capacitance</i> rule parameter
<i>default_max_transition</i>	A positive floating-point number
<i>SDC_MergeBlocks_deglitch_cell</i>	A valid library cell name
<i>SDC_MergeBlocks_deglitch_cell_clock</i>	A valid pin of the library cell specified using the <i>SDC_MergeBlocks_deglitch_cell</i> rule parameter
<i>scan_insert</i>	"yes" or "no"
<i>strict</i>	"yes", "no", "1", or "0"
gen_sdc_phase	One of "RTL", "PRELAYOUT", or "POSTLAYOUT" (case-sensitive)
<i>SDC_DnStrm02_seq_cell</i>	One of "latch", "none", or "unspecified"
<i>SDC_DnStrm02_control_point</i>	One of "before", "none", "after", or "unspecified"
<i>SDC_DnStrm02_control_signal</i>	One of "scan_enable", "test_mode", or "unspecified"
<i>io07a_range</i>	Either "outside" or "within"
<i>SDC_DnStrm04_template</i>	One of "Astro" and "Monterey"

Setup Rules

Rule Parameter	Check whether the specified value is...
<i>op_percent</i> , <i>op_percent_min</i> , <i>op_percent_max</i> , <i>op_percent_min</i> , <i>op_percent_max</i> , <i>inp_percent</i>	A positive floating-point number less than 100
<i>inp_delay_margin</i> , <i>op_delay_margin</i>	A positive floating-point number
<i>inp_percent</i> and <i>inp_delay_margin</i> , <i>op_percent</i> and <i>op_delay_margin</i>	<i>inp_percent</i> and <i>inp_delay_margin</i> rule parameters cannot be specified together. <i>op_percent</i> and <i>op_delay_margin</i> rule parameters cannot be specified together.
<i>num_falsepath_max</i> , <i>num_mcpath_max</i> , <i>logic_level_max</i> , <i>timing_path_max</i>	A positive integer number
<i>tc_comments_cmd_file</i>	<i>set_max_delay</i> and <i>set_min_delay</i> should not be specified together
<i>gen_sdc_constraints_file</i>	file specified using the parameter exists
<i>gen_sdc_param_file</i>	file specified using the parameter exists

Constraint(s)

None

Messages and Suggested Fix**Message 1**

The following message appears when an incorrect value is passed to the `<rule-parameter>` parameter:

[FATAL] Wrong value passed to parameter "<rule-parameter>"

Potential Issues

Incorrect value is specified to the rule parameter.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Ensure that the correct value is passed to the rule parameter.

Message 2

The following message appears when a non-integer value is passed to the integer-type rule parameter *<rule-parameter>*:

[FATAL] Value passed to parameter "*<rule-parameter>*" not of integer type

Potential Issues

Incorrect value is specified to the rule parameter.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Ensure that the correct value is passed to the rule parameter.

Message 3

The following message appears when the non-positive integer value passed to the rule parameter *<rule-parameter>* that accepts only positive integer values:

[FATAL] Value passed to parameter "*<rule-parameter>*" should be a positive integer

Potential Issues

Incorrect value is specified to the rule parameter.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Ensure that the correct value is passed to the rule parameter.

Message 4

The following message appears when a non-floating-point value is passed to the rule parameter *<rule-parameter>* that accepts only floating-point values:

[FATAL] Value passed to parameter "*<rule-parameter>*" not of Float

Potential Issues

Incorrect value is specified to the rule parameter.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Ensure that the correct value is passed to the rule parameter.

Message 5

The following message appears when a non-positive floating-point value is passed to the rule parameter *<rule-parameter>* that accepts only positive floating-point values:

[FATAL] Value passed to parameter "*<rule-parameter>*" should be a positive float

Potential Issues

Incorrect value is specified to the rule parameter.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Ensure that the correct value is passed to the rule parameter.

Message 6

The following message appears when a value beyond the expected range (0 to 100.00) is passed to the rule parameter *<rule-parameter>*:

[FATAL] Value passed to parameter "*<rule-parameter>*" should be between 0 and 100.00

Potential Issues

Incorrect value is specified to the rule parameter.

Consequences of Not Fixing

SpyGlass exits without any rule checking.

How to Debug and Fix

Ensure that the correct value is passed to the rule parameter.

Message 7

The following message appears the value passed to the rule parameter

<rule-parameter> is greater than the corresponding value specified in the associated library:

[FATAL] Value passed to parameter "*<rule-parameter>*" is more than the value given in library

Potential Issues

Incorrect value is specified to the rule parameter.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Ensure that the correct value is passed to the rule parameter.

Message 8

The following message appears when the file name specified with the rule parameter *<rule-parameter>* is not found:

[FATAL] File specified through parameter "*<rule-parameter>*" not Found

Potential Issues

Incorrect value is specified to the rule parameter.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Ensure the file specified in the rule parameter is present.

Message 9

The following message appears when the file name specified with the rule parameter *<rule-parameter>* could not be opened for reading:

[FATAL] File specified through parameter "*<rule-parameter>*" cannot be opened for reading

Potential Issues

The file is either not present in the specified location or read permission is not set for that file.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Ensure the file is present in the specified location and read permission is also set for that file.

Message 10

The following message appears when no value is specified for the rule parameter `<rule-parameter>` that requires a value:

[FATAL] No value passed to parameter "<rule-parameter>"

Potential Issues

No value is specified to the rule parameter.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Ensure that the correct value is passed to the rule parameter.

Message 11

The following message appears when value 0 is specified as the second value in the name-value pair of a design unit for the `blackbox` rule parameter:

[FATAL] Wrong value passed to parameter "blackbox"

Potential Issues

Incorrect value is specified to the rule parameter.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Ensure that the correct value is passed to the rule parameter.

Message 12

The following message appears when a non-positive integer value is specified as the second value in the name-value pair of design unit `<du-name>` for the `blackbox` rule parameter:

[FATAL] Value passed to parameter "blackbox" for design unit '`<du-name>`' not of integer type

Potential Issues

Incorrect value is specified to the rule parameter.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Ensure that positive integer value is passed as second value in the name value pair.

Message 13

The following message appears when no value is specified as the second value in the name-value pair of design unit `<du-name>` for the blackbox parameter:

```
[FATAL] No value passed to parameter "blackbox" for design unit '<du-name>'
```

Potential Issues

Incorrect value is specified to the rule parameter.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Ensure that the correct value is passed to the rule parameter.

Message 14

The following message appears when the value of the `default_min_capacitance` is greater than the value of the `default_max_capacitance` rule parameter:

```
[FATAL] Value of parameter default_min_capacitance is greater than the value of parameter default_max_capacitance
```

Potential Issues

The value of `default_min_capacitance` is greater than the value of `default_max_capacitance`.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

The value of `default_min_capacitance` should be less than the value of `default_max_capacitance`.

Message 15

The following message appears when the value of the `default_min_transition` is greater than the value of the `default_max_transition` rule parameter:

[FATAL] Value of parameter `default_min_transition` is greater than the value of parameter `default_max_transition`

Potential Issues

The value of `default_min_transition` is greater than the value of `default_max_transition`.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

The value of `default_min_transition` should be less than the value of `default_max_transition`.

Message 16

The following message appears when both `inp_percent` and `inp_delay_margin` parameters are specified:

[FATAL] Either specify "inp_percent" parameter or "inp_delay_margin" parameter

Potential Issues

Both `inp_percent` and `inp_delay_margin` is specified in the same run.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Specify either one `inp_percent` or `inp_delay_margin`.

Message 17

The following message appears when both `op_percent` and `op_delay_margin` parameters are specified:

[FATAL] Either specify "op_percent" parameter or

"op_delay_margin" parameter

Potential Issues

Both `op_percent` and `op_delay_margin` is specified in the same run. **Consequences of Not Fixing**

SpyGlass exits without any rule-checking

How to Debug and Fix

Specify either one `op_percent` or `op_delay_margin`.

Example Code and/or Schematic

Specify the rule parameter in a file say 'param.f' and include it through `-f param.f` as command-line arguments:

The contents of param.f are:

```
-scan_insert='xyz'  
-tc_fanout_limit='10.5'  
-logic_level_max='-3'  
-inp_percent_min='x'  
-inp_percent_max='-4'  
-op_percent_min='500'  
-tc_high_fan_nets_file='a.x' //a.x doesn't exist  
-c_compare_cmd_file='1.txt' //no read permission for '1.txt'  
-tc_ignore_min  
-blackbox='myb1=0,myb2=10,myb3=xyz'  
-default_min_capacitance=5  
-default_max_capacitance=3  
-default_min_transition=4  
-default_max_transition=1  
-pt=yes  
-inp_percent=60  
-inp_delay_margin=60
```

Setup Rules

```
-op_percent=30
```

```
-op_delay_margin=30
```

The FATAL violation message will be reported for all the above rule parameters.

Default Severity Label

Fatal

Rule Group

Prereqs

Reports and Related Files

None

ParamSanityCheck01b

Performs a semantic check on user-specified rule parameters

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases and to perform sanity checks on rule parameters after synthesis.

Description

The *ParamSanityCheck01b* rule performs post-synthesis syntactic checks on user-specified rule parameters.

The *ParamSanityCheck01b* rule checks the following rule parameters:

Rule Parameter	Check whether the specified value is
<i>chip</i>	A list of top-level design units (as detected by the SpyGlass DetectTopDesignUnits built-in rule)
<i>blackbox</i>	The first value in each name-value pair is an existing design unit.
<i>io07_ports</i>	Ports that are part of design units defined as <i>current_design</i> in SGDC file (Only one message is generated for all rule-violating ports.)

In case, any of the conditions checked by the *ParamSanityCheck01b* rule are not met, SpyGlass exits without further rule-checking.

Parameter(s)

- *chip*: Default value is unspecified. Set the parameter to a string value to specify the name of the design unit which corresponds to the chip level.
- *blackbox*:
- *io07_ports*: Default value is * and all ports are checked. Set the parameter to a list of string values to list the specified ports.

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message appears when no definition of design units `<du-name-list>` specified with the `blackbox` parameter is found in the design:

[FATAL] Design units '`<du-name-list>`' passed to parameter "blackbox" not found in the design

Potential Issues

No definition of design unit specified with the rule parameter is found in the design.

Consequences of Not Fixing

SpyGlass exits without any rule checking.

How to Debug and Fix

Ensure the presence of correct definition of design unit specified with the rule parameter.

Message 2

The following message appears when no ports names `<name-list>` as specified with the `io07_ports` parameter are found in the design unit specified as `current_design`:

[FATAL] Ports '`<name-list>`' passed to parameter "io07_ports" not found in the design specified in SGDC file(s) under `current_design`

Potential Issues

Reported port names specified with the rule parameter is not found in the design.

Consequences of Not Fixing

SpyGlass exits without any rule checking.

How to Debug and Fix

Specify only those ports which are present in the design.

Example Code and/or Schematic

Message 3

The following message appears when the design units `<du-name-list>` specified with the `chip` parameter are not top level design units in the design:

[FATAL] Wrong value (`<du-name-list>`) passed to parameter "chip"

Potential Issues

Incorrect design unit is specified with the rule parameter.

Consequences of Not Fixing

SpyGlass exits without any rule-checking.

How to Debug and Fix

Ensure the presence of correct design unit specified with the rule parameter.

Example Code and/or Schematic

Example 1

Suppose, there is no instance of `myb1` in the Verilog file and you have specified the following:

```
set_parameter blackbox 'myb1:1'
```

This rule will report Message 1.

Example 2

Suppose, there `in100` is not present in the top-level and you have specified the following:

```
set_parameter io07ports 'in1,in100'
```

This rule will report Message 2.

Example 3

Suppose, the `mytest` module is not the top-level design unit and you have specified the following:

```
set_parameter chip 'mytest'
```

This rule will report Message 3.

Setup Rules

Default Severity Label

Fatal

Rule Group

Prereqs

Reports and Related Files

None

SDCPARSE

Reads the SDC and SGDC files for running rules in the SpyGlass Constraints solution

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *SDCPARSE* rule reads the Synopsys Design Constraints (SDC) and SpyGlass Design Constraints files (SGDC) for running the rules in the SpyGlass Constraints solution.

The *SDCPARSE* rule uses an internal SDC parsing engine that reports messages for rules named SDC_*. See *SpyGlass Built-In Rules Reference Guide* for details of these rules.

Parameter(s)

- *tc_ignored_commands*: Default is unspecified. Set the value to the file name that contains the list of SDC commands, which the SDCPARSE rule should ignore while parsing the SDC files.
- *tc_stop_parsing_ignored_commands*: Default is no. Set this parameter to yes to stop parsing any nested SDC commands, which are specified inside SDC commands passed through the *tc_ignored_commands* parameter.
- *tc_disable_caching*: Default is no. Use this parameter to disable caching of auxiliary commands.
- *sdMsg2Fatal*: Default is no. Set the parameter to yes to change the severity of all built-in SDC_* rules to FATAL.
- *pt*: Default is yes. Refer to the *pt* description for more details.
- *library_clk_gen_naming*: Default is yes and you can generate the clock names in accordance with other industry tools, i.e. the hierarchical name in the instance name or pin name format. Set this parameter to no to generate clock names as per the Liberty Reference Manual.

Constraint(s)

None

Setup Rules

Messages and Suggested Fix

No impact

Example Code and/or Schematic

Not applicable

Default Severity Label

ERROR

Rule Group

Prereqs

Reports and Related Files

None

SpyGlass Design Constraints File Structure Rules

The SpyGlass Design Constraints (SGDC) File Rules Group `ConstraintFileStructure` contains the following rules:

Rule	Description
<i>Const_Struct01</i>	Incomplete definitions in the SGDC files
<i>Const_Struct02</i>	Missing or incomplete <i>current_design</i> specification for a block
<i>Const_Struct03</i>	Improper <i>set_case_analysis</i> settings
<i>Const_Struct04</i>	Runs the <i>Const_Struct04a</i> and <i>Const_Struct04b</i> rules
<i>Const_Struct04a</i>	Overwritten constraints
<i>Const_Struct04b</i>	Duplicated constraints
<i>Const_Struct05</i>	Conflicting constraints
<i>Const_Struct07</i>	Constraints where objects have been directly specified (that is, without the corresponding <code>get_*</code> argument)
<i>Const_Struct08</i>	Same constraint values specified for different corners
<i>Const_Struct09</i>	Missing <i>sdc_data</i> constraints for best or worst corners
<i>Const_Struct10</i>	Constraints applied on buffers/inverters

Const_Struct01

Reports incomplete SGDC files

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Const_Struct01* rule checks the SpyGlass Design Constraints (SGDC) files for structural completeness. It checks for the existence of:

- **Two modes:** At least two modes should be specified. Modes are specified using the `-mode` argument of the *sdc_data* constraint. Since, it is not possible to infer the mode type by the mode name, the *Const_Struct01* rule just checks that at least two modes are specified.
- ***sdc_data* constraints for each block:** For each mode specified at the design-level, *sdc_data* constraints must also be specified for each block.

Parameter(s)

None

Constraint(s)

SGDC

sdc_data (Mandatory): Use to specify the SDC file for a design module.

Messages and Suggested Fix

Message 1

The following message appears when *sdc_data* directives for at least two different modes have not been specified for the design/block *<name>*:

[INFO] Both functional and scan shift modes not specified in schema for design/block "<name>"

Potential Issues

The violation message appears if functional and scan shift modes are not specified for current design.

Consequences of Not Fixing

The validation for the missing modes will not be done.

How to Debug and Fix

Check the SGDC file for the design/block reported. Specify *sdc_data* for both functional and scan shift mode separately.

Refer to Example 1.

Message 2

The following message appears when *sdc_data* directives for at least two different modes have been specified for the design/block *<name>*, but all required *sdc_data* directives (for different corners) have not been specified:

```
[INFO] A schema has not been defined for each meaningful variant (functional/scan shift mode) for design/block "<name>"
```

Potential Issues

The violation message appears if all required *sdc_data* constraints have not been specified.

Consequences of Not Fixing

The validation for the reported block for the missing mode will not be performed.

How to Debug and Fix

Specify *sdc_data* for both functional and scan shift mode for the reported block separately.

Refer to Example 2.

Message 3

The following message appears when a *sdc_data* directive has been specified for design/block *<name>* under mode *<mode-name>*, but not for the corresponding top-level design/block *<top-name>*:

```
[INFO] Mode "<mode-name>" specified for design/block "<name>" is not defined for top level design/block "<top-name>"
```

Potential Issues

The violation message appears if a *sdc_data* directive has been specified for design but not for the corresponding top-level design.

Consequences of Not Fixing

The timing analysis of top and block will be done for two different modes.

Therefore, the result might be incorrect.

How to Debug and Fix

Check the SGDC file for the reported mode name with respect to the top-level design. Specify *sdc_data* for top-level corresponding to the missing mode-name.

Refer to Example 3.

Message 4

The following message appears when an *sdc_data* directive has not been specified for the design/block *<name>*:

```
[INFO] Mode not specified for design/block "<name>"
```

Potential Issues

The violation message appears if an *sdc_data* directive has not been specified for the design.

Consequences of Not Fixing

The reported block will not be checked.

How to Debug and Fix

Specify *sdc_data* with both functional and scan shift mode separately.

Refer to Example 4.

Example Code and/or Schematic

Example 1

In this example, the *Const_Struct01* rule reports a violation because both the functional and scan shift mode are not specified for the top.

```
constraints.sgdc:
```

```
current_design top
```

```
sdc_data -type ConstStruct011.sdc -mode func -corner Worst
```

```
sdc_data -type ConstStruct012.sdc -mode func -corner Worst
```

Example 2

In this example, the *Const_Struct01* rule reports a violation message for top because both modes are not specified.

```
constraints.sgdc:
```

```
current_design top
sdc_data -type ConstStruct011.sdc -mode func -corner Worst
```

Example 3

In this example, the *Const_Struct01* rule reports a violation message because mode scan is specified for block 'myflop', but it not specified for top.

constraints.sgcd:

```
current_design top
sdc_data -type ConstStruct011.sdc -mode func -corner Worst
sdc_data -type ConstStruct012.sdc -mode func -corner Worst
block -name myflop myand top
blocksize -min 0 -max 100
current_design myflop
sdc_data -type ConstStruct011.sdc -mode func -corner Worst
sdc_data -type ConstStruct012.sdc -mode scan -corner Best
```

Example 4

In this example, the *Const_Struct01* rule reports a violation message because no mode is specified.

constraints.sgcd:

```
current_design top
sdc_data -type ConstStruct012.sdc -corner Worst
```

Default Severity Label

Info

Rule Group

ConstraintFileStructure

Reports and Related Files

None

Const_Struct02

Reports incomplete blocks

When to Use

This rule is applicable to the RTL, pre-layout, and post-layout phases.

Description

The *Const_Struct02* rule reports blocks with missing or incomplete *current_design* specifications. This rule requires that each block specified using the *block* constraint has a corresponding *current_design* specification with at least one *sdc_data* directive.

Parameter(s)

None

Constraint(s)

SGDC

- *sdc_data* (Mandatory): Use to specify the SDC file for a design module.
- *block* (Optional): Use to specify the blocks.
- *current_design* (Optional): Use to specify the directive scope.

Messages and Suggested Fix

The following message appears:

```
[INFO] At least one expected SDC constraints file is missing
for design/block "<name>"
```

Potential Issues

The violation message appears when the design/block *<name>* has neither:

- the corresponding *current_design* specification nor
- a *current_design* specification, which does not have any *sdc_data* directive

The design/block *<name>* is specified using the *block* constraint.

Consequences of Not Fixing

For a specific block, if there are no violation messages reported, there is a possibility that no files were specified. As a result, the rules were also not checked.

How to Debug and Fix

Review the SGDC file for the reported block and specify the missing [sdc_data](#) constraint corresponding to that block.

Example Code and/or Schematic

In this example, there is no [sdc_data](#) constraint for 'blockA'. Therefore, the *Const_Struct02* rule reports a violation corresponding to the missing [sdc_data](#) constraint.

constraints.sgdc:

```
current_design top
sdc_data -type top.sdc -level rtl -mode func -corner Worst
block -name blockA
```

Default Severity Label

Info

Rule Group

ConstraintFileStructure

Reports and Related Files

None

Const_Struct03

Identifies `set_case_analysis` settings that are inconsistent with mode definitions

When to Use

Use to ensure that the design has a unique setup for each operational mode. This rule is applicable to all design cycle stages after you have multiple modes in the SDC file.

Description

The *Const_Struct03* rule reports improper *set_case_analysis* settings. The *Const_Struct03* rule checks on the basis of the *set_case_analysis* settings only and is not based on any electrical checks.

The *Const_Struct03* rule reports a violation in the following cases:

- None of the *set_case_analysis* settings is different in different modes at the same level
- The *set_case_analysis* settings are not the same for different corners under the same mode.
- The *set_case_analysis* constraint is not specified for any mode.
- The *set_case_analysis* constraint is present in one *sdc_data* and not in the other *sdc_data*, and all other *set_case_analysis* constraints are the same in both *sdc_data*. This is when the two SDC *sdc_data* belong to the same mode.

Prerequisites

This rule requires the *set_case_analysis* constraint in the SDC file.

Rule Exceptions

The *Const_Struct03* rule does not:

- Compare *sdc_data* mode pair of reference and implement.
- Report a violation when only a reference/implement *sdc_data* mode is defined and no case analysis is given.

There is no interpretation of the mode value. Therefore, different values of the `-mode` argument are assumed to be in different modes. This rule compares the reference/implement mode with any other mode.

Parameter(s)

None

Constraint(s)

SDC

- [set_case_analysis](#) (Mandatory): Use to define case settings.

Messages and Suggested Fix

Message 1

The following message appears when the [set_case_analysis](#) settings for analysis mode `<mode1-name>` in files `<file-name-list1>` are the same as those for analysis mode `<mode2-name>` in files `<file-name-list2>` when the two sets of files are for different levels:

```
[WARNING] set_case_analysis settings for mode <mode1-name>
(file(s): <file-name-list1>) and <mode2-name> (file(s): <file-
name-list2>) are identical but expected to be different
```

Potential Issues

Case analysis settings are not unique for a mode. Design operates in different modes using different case analysis.

Consequences of Not Fixing

If case analysis is the same across modes, timing for one of the modes (partially or fully) might be missed. Therefore, the timing would not be complete. This leads to chip failure when the design is operated in a particular mode.

How to Debug and Fix

The violation message indicates that though the modes are different, the [set_case_analysis](#) settings have the same value. The violation message mentions the [sdc_data](#) for the mode settings.

Update the inconsistent case analysis values as highlighted in the SDC file.

Message 2

The following message appears when the [set_case_analysis](#) value `<value1>` for pin/port `<name>` under analysis mode `<mode-name>` is different from the [set_case_analysis](#) value `<value2>` for the same mode

for another level in the SDC file *<file2-name>* at line *<num2>*:

[WARNING] Case analysis value *<value1>* for pin/port *<name>* (mode *<mode-name>*, file *<file1-name>*, line *<num1>*) does not match case analysis value *<value2>* for pin/port *<name>* (mode *<mode-name>*, file *<file2-name>*, line *<num2>*)

Potential Issues

Design operates in different modes using different case analysis.

Consequences of Not Fixing

If case analysis is the same across modes, timing for one of the modes (partially or fully) might be missed. Therefore, the timing would not be complete. This leads to chip failure when the design is operated in a particular mode.

How to Debug and Fix

The violation message indicates that though the modes are the same but the case analysis is different. The violation message mentions the *sdc_data* constraints for the mode settings. The violation message refers to the pin at which values differ.

Update the inconsistent case analysis values as highlighted in the SDC file.

Message 3

The following message appears when *set_case_analysis* settings are not in files *<file-name-list>* specified in the *sdc_data* constraints:

[WARNING] No *set_case_analysis* specified for file(s) (*<file-name-list>*)

Potential Issues

Different case analysis values forces design to run in different modes. This message is reported when the design is running in no mode.

Consequences of Not Fixing

Typically, *set_case_analysis* is expected in the SDC file. One exception is when the SDC file is mode merged. Therefore, check if the SDC file should be without *set_case_analysis*.

How to Debug and Fix

The violation message indicates the absence of *set_case_analysis* constraints for a mode. The violation message also mentions the SDC files for that

mode.

If your design requires *set_case_analysis* constraints, update the SDC file.

Message 4

The following message appears when *set_case_analysis* defined for <mode-name> at <file-name>, <line-num> for a pin/port <name> is present in one *sdc_data*, but not in the other *sdc_data* (<file-name-list>) and all other *set_case_analysis* constraints are the same in both *sdc_data*:

[WARNING] *set_case_analysis* setting for pin/port <name> (<mode-name>, <file-name>, <line-num>) does not exist in file(s): <file-name-list> (<mode-name>)

Potential Issues

Different case analysis values may dissimilar timing analysis for the two modes. This message is reported when the design is running in the same mode.

Consequences of Not Fixing

Timing analysis may not be the same for the two SDC *sdc_data* even though they belong to the same mode.

How to Debug and Fix

The inconsistent case analysis is highlighted in the Console GUI. To resolve this violation message, update the SDC file by either adding a similar case analysis constraints to the second *sdc_data* or removing the case analysis from the first *sdc_data*.

Example Code and/or Schematic

Consider the following example:

```
#Constraints.sgdc
current_design top
  sdc_data -file top_function_pre.sdc \
    -level prelayout -mode function
  sdc_data -file top_function_post.sdc \
    -level postlayout -mode function
  sdc_data -file top_scan_all.sdc \
    -level all -mode scan

#top_function_pre.sdc
```

SpyGlass Design Constraints File Structure Rules

```

set_case_analysis 1 [get_ports "MODE1"]
set_case_analysis 1 [get_ports "MODE2"]

#top_function_post.sdc
set_case_analysis 1 [get_ports "MODE1"]
set_case_analysis 0 [get_ports "MODE2"]

#top_scan_all.sdc
set_case_analysis 1 [get_ports "MODE1"]
set_case_analysis 1 [get_ports "MODE2"]

```

Since, the *set_case_analysis* settings for the function mode, which is specified in `top_function_pre.sdc`, and scan mode, which is specified in `top_scan_all.sdc`, are identical, SpyGlass generates the following message:

```
[WARNING] set_case_analysis settings for file(s)
top_function_pre.sdc in analysis mode function are identical to
the settings for file(s) top_scan_all.sdc in analysis mode scan
```

In addition, the *set_case_analysis* settings for the same function mode in different phases, which is `top_function_pre.sdc` file for the Pre-layout phase and `top_function_post.sdc` file for the Post-layout phase, are not the same. Therefore, SpyGlass generates the following message:

```
set_case_analysis for analysis mode function is not same as in
file top_function_pre.sdc, line 2
```

Default Severity Label

Warning

Rule Group

ConstraintFileStructure

Reports and Related Files

None

Const_Struct04

Detection of overwritten/duplicate constraints

The Const_Struct04 runs the [Const_Struct04a](#) and [Const_Struct05](#) rules.

Const_Struct04a

Detects overwritten constraints

When to Use

To identify redundant constraints which could have been overwritten intentionally or unintentionally. This rule is applicable to all design stages.

Description

The *Const_Struct04a* rule reports overwritten constraints. A constraint is said to be overwritten when the constraint with a different value has been specified for the same design object at two or more places in same or different SDC files.

The *Const_Struct04a* rule currently checks the following constraints for overwritten status:

<i>create_clock</i>	<i>create_generated_clock</i>	<i>set_annotated_transition</i>
<i>set_case_analysis</i>	<i>set_clock_latency</i>	<i>set_clock_transition</i>
<i>set_clock_uncertainty</i>	<i>set_drive</i>	<i>set_driving_cell</i>
<i>set_fanout_load</i>	<i>set_ideal_latency</i>	<i>set_input_delay</i>
<i>set_input_transition</i>	<i>set_load</i>	<i>set_max_capacitance</i>
<i>set_max_delay/ set_min_delay^a</i>	<i>set_max_fanout</i>	<i>set_max_transition</i>
<i>set_min_capacitance</i>	<i>set_output_delay</i>	<i>set_multicycle_path^b</i>
<i>set_operating_conditions</i>	<i>set_scan_signal</i>	<i>set_port_fanout_number</i>
<i>set_resistance</i>	<i>set_signal_type</i>	

a. Only if the *strict* parameter is set.

b. Only if the *strict* parameter is set.

Rule Exceptions

The *Const_Struct04a* rule does not check:

- Overwritten constraints are found to have the same file names and line numbers.

- Overwrites of the *set_false_path* constraints because they can only be duplicated and not be overwritten with a different value. While it is possible to change the scope of the *set_false_path* constraints in PT through a combination of options and the *timing_rise_fall_clock_compatibility* variable, this also changes the scope of paths being considered as false rather than actually modifying their values.
- Inter-clock *set_clock_uncertainty*.
- For overwrites of the *set_ideal_net* and *set_dont_touch_network* commands as these commands may only be duplicated and not overwritten.

Parameter(s)

- *strict*: Default is no. Set the value to yes to check for overwritten status of the *set_max_delay/set_min_delay* and *set_multicycle_path* constraints.

Constraint(s)

Refer to the *Description* for a list of constraints that the *Const_Struct04a* rule checks.

Messages and Suggested Fix

The following message appears when the constraint *<constr>* is overwritten for object *<obj-name>* at different locations: SDC file *<file1-name>*, line *<num1>* and SDC file *<file2-name>*, line *<num2>*:

```
[INFO] <constr> is overwritten for design object <obj-name> in  
(File <file1-name> Line <num1>), (File <file2-name> Line  
<num2>)
```

Potential Issues

This message is reported when the same constraint is overwritten in the same or different files.

Consequences of Not Fixing

Since you may be loading several SDC files at the same time, sometimes constraints overwrite themselves. In such situations, the tools keep the last constraint command and ignore all earlier constraints. Such situations

do not prevent the tools from working but are not clean or optimal. This can lead to unexpected behavior and lengthy debug times.

How to Debug and Fix

An SDC constraint is overwritten by the other if certain options are changed between the two constraints and one constraint is occurring later than the other in the SDC file. The violation message indicates the overwritten constraint.

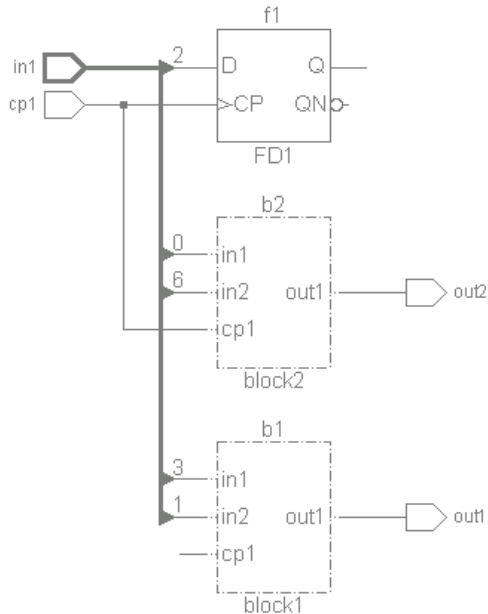
The overwritten constraint is not checked for [set_false_path](#) because this constraint can only be duplicated and not overwritten. The [set_multicycle_path](#) constraint are checked only when the [strict](#) parameter is set to `yes`.

Update the SDC file as highlighted.

Example Code and/or Schematic

Example 1[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The block1.sdc is as follows:

```

block1.sdc  block1.sdc (2)
create_clock -name vclk1 -period 10.000000 -waveform { 0.000000 5.000000 } cp1
set input_delay 2.3 -clock vclk1 {in2}
set case_analysis 0 {in1}
set_case_analysis 1 {in1}

```

The violation message appears because there are two `set_case_analysis` constraints with different values defined on the same port `in1`. In the `block1.sdc` file, the corresponding constraint is highlighted.

Example 2

Test Case Files Not Available

The `set_input_delay` constraint is considered overwritten in the following example:

```
...
set_input_delay 2 -clock clk IN1
...
set_input_delay 5 -clock clk IN1
...
```

Example 2

In the following example, the `Const_Struct04a` rule reports a violation for such `set_false_path` constraints.

```
Set_false_path -from [get_ports {i1}]
Set_false_path -from [get_ports {i1}]
```

However, even if a single path exists from a input port `i1` for a output port `o1`, this rule does not report a violation for the constraints given below:

```
Set_false_path -from [get_ports {i1}]
Set_false_path -from [get_ports {i1}] -end [get_ports {o1}]
```

Default Severity Label

Info

Rule Group

ConstraintFileStructure

Reports and Related Files

None

Const_Struct04b

Detects duplicate constraints

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *Const_Struct04b* rule detects duplicate constraints. A duplicate constraint is when a constraint with the same value has been specified more than once in the same design object or different SDC files.

Rule Exceptions

Since it is possible that an SDC file gets sourced multiple times, the *Const_Struct04b* rule does not report situations where the duplicate constraints are found to have the same file names and line numbers.

Parameter(s)

- *strict*: Default value is no. Set the value to yes to check for overwritten status of the *set_max_delay/set_min_delay* and *set_multicycle_path* constraints.

Constraint(s)

The *Const_Struct04b* rule currently checks the following constraints for duplicate status:

<i>create_clock</i>	<i>create_generated_clock</i>	<i>set_annotated_transition</i>
<i>set_case_analysis</i>	<i>set_clock_latency</i>	<i>set_clock_transition</i>
<i>set_clock_uncertainty</i>	<i>set_dont_touch_network</i>	<i>set_drive</i>
<i>set_driving_cell</i>	<i>set_fanout_load</i>	<i>set_false_path</i> ^a
<i>set_ideal_latency</i>	<i>set_ideal_net</i>	<i>set_input_delay</i>
<i>set_input_transition</i>	<i>set_load</i>	<i>set_max_capacitance</i>
<i>set_max_delay/ set_min_delay</i> ^b	<i>set_max_fanout</i>	<i>set_max_transition</i>

<i>set_min_capacitance</i>	<i>set_output_delay</i>	<i>set_multicycle_path^c</i>
<i>set_operating_conditions</i>	set_scan_signal	<i>set_port_fanout_number</i>
<i>set_resistance</i>	set_signal_type	

- Only if the *strict* rule parameter is set.
- Only if the *strict* rule parameter is set.
- Only if the *strict* rule parameter is set.

Messages and Suggested Fix

The following message appears:

[INFO] <constr> is duplicated for design object <obj-name> in (File <file1-name> Line <num1>), (File <file2-name> Line <num2>)

Potential Issues

This message is reported when a constraint <constr> is duplicated for object <obj-name> at two locations — SDC file <file1-name>, line <num1> and SDC file <file2-name>, line <num2>

The constraint can be duplicated across the same or different files.

Consequences of Not Fixing

Since several SDC files are being loaded at the same time, it is possible that the constraints get duplicated. In such a situation, the tool keeps the last constraint command and ignores all earlier constraints. Though such situations do not prevent the tools from working, they are not clean or optimal. This can lead to lengthy debug times.

How to Debug and Fix

The SDC file/line of duplicate constraint is highlighted in the Console GUI. Remove that constraint.

Example Code and/or Schematic

In the following SDC snippet, the *set_input_delay* constraint is considered to be duplicate.

```
...
set_input_delay 2 -clock clk IN1
```

```
...  
set_input_delay 2 -clock clk IN1  
...
```

Therefore, the *Const_Struct04b* rule reports a violation message.

Default Severity Label

Info

Rule Group

ConstraintFileStructure

Reports and Related Files

None

Const_Struct05

Detects conflicting constraints

When to Use

To identify conflicts in specified constraints. This rule is applicable to all design stages.

Description

The *Const_Struct05* rule reports conflicting constraints between the following pairs of commands:

- *create_clock* and *create_generated_clock*
- *set_driving_cell* and *set_input_transition*

Parameter(s)

- *tc_waive_const_struct05*: Default is none. This indicates that the Const_Struct05 rule checks each pair you want to waive. For example, For example:

```
-tc_waive_const_struct05="set_driving_cell: create_clock"
```

Constraint(s)

SDC

- *create_clock*: Use to create a clock.
- *create_generated_clock*: Use to create a clock.
- *set_driving_cell*: Use to set attributes on input or inout ports, specifying that a library cell or pin drives the ports.
- *set_input_transition*: Use to set transition values on input or inout ports.

Messages and Suggested Fix

The following message appears when conflicting constraints *<constr1>* (at SDC file *<file1-name>*, line *<num1>*) and *<constr2>* (at SDC file *<file2-name>*, line *<num2>*) are applied on the object *<obj-name>*:

```
[INFO] Conflicting constraints are set for design object
<obj-name> on File <file1-name> Line <num1> (<constr1>), and
```

File <file2-name> Line <num2> (<constr2>)

Potential Issues

This message is reported when two conflicting constraints are applied on the same object.

Consequences of Not Fixing

One of the constraint will be ignored. Therefore, the results could be intentional or unintentional.

How to Debug

The violation message indicates the types of constraints. The SDC file highlights the conflicting constraints. Open the SDC file and check for the following conflicted constraints on the same object:

- [create_clock](#) and [create_generated_clock](#)
- [set_driving_cell](#) and [set_input_transition](#)

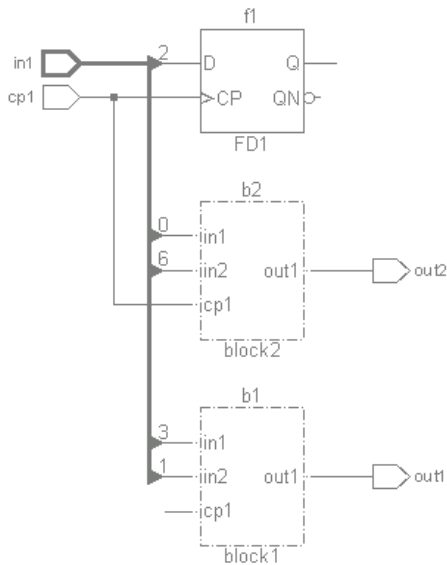
Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:

SpyGlass Design Constraints File Structure Rules



The `top.sdc` is as follows:

```

top.sdc | top.sdc (2)
create_clock -name clk1 -period 14.000000 -waveform { 0.000000 5.000000 } { in1[4] }
create_clock -name clk2 -period 12.000000 -waveform { 0.000000 5.000000 } { cp1 }
create_generated_clock -name gclk1 -source in1[4] -divide_by 2 { cp1 }
set_input_delay 2.3 -clock clk1 { in1[1] }
set_input_delay 1.5 -clock gclk1 { in1[1] } -add_delay

```

The violation message appears because conflicting constraints are defined on the same ports `cp1`. In the `top.sdc` file, the corresponding constraint is highlighted.

Example 2

Test Case Files Not Available

In the following example, the same clock name, `clk1`, and object name, `U1/Q`, are specified for both `create_clock` and `create_generated_clock`.

Therefore, the two constraints are conflicting constraints.

```
...
create_clock -name clk1 -period 10 \
  -waveform {0 5} [get_pins U1/Q]
...
create_generated_clock -name clk1 -divide_by 2 \
  -source [get_ports CLOCK1] [get_pins U1/Q]
...
```

Example 3Test Case Files Not Available

In the following example, both *set_driving_cell* and *set_input_transition* constraints are specified for the IN1 object. Since the *set_driving_cell* constraint has precedence over the *set_input_transition* constraint, the tools ignore the latter constraint.

```
...
set_driving_cell -library CORELIB -lib_cell BFX8 \
  -pin Z [get_ports IN1]
...
set_input_transition 0.8[get_ports IN1]
...
```

Default Severity Label

Info

Rule Group

ConstraintFileStructure

Reports and Related Files

None

Const_Struct07

Reports constraints with objects directly specified

When to Use

This rule is applicable to the RTL, pre-layout and post-layout phases.

Description

The *Const_Struct07* rule reports constraints where objects have been directly specified without the corresponding `get_*` argument. However, if there is no ambiguity while referring to an object or the object type is obvious in the syntax, then the *Const_Struct07* rule does not report a violation even if the object is referenced directly.

Parameter(s)

None

Constraint(s)

All the supported `get_*` constraints.

Messages and Suggested Fix

The following message appears:

```
[INFO] Object "<obj -name>" referenced directly. Use commands like get_ports, get_clocks, etc to reference objects indirectly
```

Potential Issues

This message is reported for a constraint where the object name `<obj-name>` has been specified without a corresponding `get_*` argument.

Consequences of Not Fixing

While accessing the objects, avoid any ambiguity. This will improve runtime of tools. If the names are directly specified, tools such as DC search the object in a list.

How to Debug and Fix

Specify whether the object is port/pin/cell by using `get_ports`, `get_pins`, and `get_cells`.

Example Code and/or Schematic

Example 1

In this example, the object `clk` does not require any referencing because the `-clock` option clearly indicates that the object referenced is a clock.

```
set_input_delay 2 -clock clk
```

Therefore, the *Const_Struct07* rule does not report a violation message.

Example 2

In this example, the *Const_Struct07* rule reports a violation message for object `clk1` since the object type is not obvious in the syntax:

```
create_clock -name clka -period 10 -waveform {0 5} {clk1}
```

The recommended way is as follows:

```
create_clock -name clka -period 10 -waveform {0 5} [get_ports  
clk1]
```

Default Severity Label

Info

Rule Group

ConstraintFileStructure

Reports and Related Files

None

Const_Struct08

Constraints values for different corners are the same

When to Use

To conduct analysis for different corners. This rule is applicable to all design stages.

Description

The *Const_Struct08* rule reports violations when the same constraint values are specified for different corners. This rule considers two constraints specifications to be the same if all arguments and values are identical. This rule reports the following values if they are the same for different corners (best and worst):

- Transition values of inputs and clocks
- Clock uncertainty values
- Clock latency values

Rule Exceptions

The *Const_Struct08* rule does not check the *set_driving_cell* because the timing values for the same cell in best corner library and worst corner library are expected to be different.

Parameter(s)

None

Constraint(s)

SDC

- *set_input_transition* (Mandatory): Use to set transition values on input or inout ports.

Messages and Suggested Fix

Message 1

The following message appears when the values specified for different corners in clock constraints of type *<constr-type>* are the same for the clock *<clk-name>* (set on port/pin *<pp-name>*) of the design unit *<du-*

name>:

[INFO] Clock <constr-type>(<option-list>) is same for both best corner and worst corner for clock ' <clk-name>' (port/pin <pp-name>) of design unit ' <du-name>'

Where, <constr-type> can be latency, transition, or uncertainty. The <option-list> has the list of options for which the values have been specified.

Potential Issues

Usually values for different corners are not expected to be the same.

Consequences of Not Fixing

The same values indicate that something is wrong in the file specification and could lead to incorrect analysis and long time for timing closure.

How to Debug and Fix

The SDC file highlights the clock constraints that have the same value.

Update the SDC file.

Message 2

The following message appears when the drive values specified for different corners are same for the port <port-name> of the design unit <du-name>:

[INFO] Drive (<option-list>) is same for both best corner and worst corner for port ' <port-name>' of design unit ' <du-name>'

Where <option-list> has the list of options for which the values have been specified.

Potential Issues

Usually values for different corners are not expected to be the same.

Consequences of Not Fixing

The same values indicate that something is wrong in the file specification and could lead to incorrect analysis and long time for timing closure.

How to Debug and Fix

The SDC file highlights the *set_drive* constraints that have the same value.

Update the SDC file.

Message 3

The following message appears when the input transition values specified for different corners are same for the port `<port-name>` of the design unit `<du-name>`:

```
[INFO] Input transition(<option-list>) is same for both best corner and worst corner for port '<port-name>' of design unit '<du-name>'
```

Where `<option-list>` has the list of options for which the values have been specified.

Potential Issues

The message is reported when the same `set_input_transition` value is applied under different corner values.

Consequences of Not Fixing

The same values indicate that something is wrong in the file specification and could lead to incorrect analysis and long time for timing closure.

How to Debug and Fix

The SDC file highlights the `set_input_transition` constraints that have the same value.

Update the SDC file.

Message 4

The following message appears when the inter-clock uncertainty values specified for different corners are same for the design unit `<du-name>`:

```
[INFO] Inter clock uncertainty is same for both best corner and worst corner of design unit '<du-name>'
```

Potential Issues

The message is reported when the same inter-clock uncertainty is applied under different corner options.

Consequences of Not Fixing

The same values indicate that something is wrong in the file specification and could lead to incorrect analysis and long time for timing closure.

How to Debug and Fix

The SDC file highlights the inter-clock uncertainty constraints that have the

same value.

Update the SDC file.

Example Code and/or Schematic

In the following example, the same value is assigned to the input port `in1` under different corner options:

```
//.sgdc
Current_design top
sdc_data -type test1.sdc -corner best
sdc_data -type test2.sdc -corner worst

//test1.sdc
set_input_transition 1 [get_ports {in1}]

//test2.sdc
set_input_transition 1 [get_ports {in1}]
```

Default Severity Label

Info

Rule Group

ConstraintFileStructure

Reports and Related Files

None

Const_Struct09

Reports missing constraints file for best or worst corners

When to Use

This rule is applicable to the RTL, pre-layout, and post-layout phases.

Description

The *Const_Struct09* rule reports a violation message when *sdc_data* directives are not specified for worst and best case in each mode of the design unit and block.

Parameter(s)

None

Constraint(s)

SGDC

- *sdc_data* (Mandatory): Use to specify the SDC file for a design module.

Messages and Suggested Fix

Message 1

The following message appears when an *sdc_data* directive for worst corner, best corner, or both corners is missing for design/block *<name>* in mode *<mode>*:

```
[INFO] <type> corner case must be specified in schema for design/block "<name>" under mode "<mode>"
```

Where *<type>* can be Best, Worst, or Best and worst.

Potential Issues

The reported corner is not specified for the block.

Consequences of Not Fixing

You may assume that the constraints for all corners have been checked. However, if the constraints are missing for a specific corner, you should know that the constraints for that specific corner have not been checked.

How to Debug and Fix

Check the SGDC file for the reported block. Specify the *sdc_data* constraint for the missing mode of the reported block

Message 2

The following message appears when only one *sdc_data* directive for one corner and no mode specification is found for design/block *<name>*:

```
[INFO] <type> corner case must be specified in schema for design/block "<name>" under unspecified mode
```

Where *<type>* can be Best, Worst, or Best and worst.

Potential Issues

The reported corner is not specified for the block.

Consequences of Not Fixing

You may assume that the constraints for all corners have been checked. However, if the constraints are missing for a specific corner, you should know that the constraints for that specific corner have not been checked.

How to Debug and Fix

Check the SGDC file for the reported block. Specify the *sdc_data* constraint for the missing mode of the reported block.

Example Code and/or Schematic

In this example, the required *sdc_data* directives have been specified for the top module and the blockB block. However, an *sdc_data* directive for the blockA block is not specified for the worst corner case in the fn1 functional mode. Therefore, the *Const_Struct09* rule reports a violation message.

```
current_design top
  sdc_data -file top_fn1_1_w.sdc top_fn1_2_w.sdc -mode fn1 -
corner Worst
  sdc_data -file top_fn1_1_b.sdc top_fn1_2_b.sdc -mode fn1 -
corner Best
  sdc_data -file top_test1_1_w.sdc top_test1_2_w.sdc -mode
test1 -corner Worst
  sdc_data -file top_test1_1_b.sdc top_test1_2_b.sdc -mode
test1 -corner Best
```

```
block -name blockA blockB
```

```
current_design blockA
```

```
  sdc_data -file blockA_fn1_b.sdc -mode fn1 -corner Best  
  sdc_data -file blockA_test1_w.sdc -mode test1 -corner Worst  
  sdc_data -file blockA_test1_b.sdc -mode test1 -corner Best
```

```
current_design blockB
```

```
  sdc_data -file blockB_fn1_w.sdc -mode fn1 -corner Worst  
  sdc_data -file blockB_fn1_b.sdc -mode fn1 -corner Best  
  sdc_data -file blockB_test1_w.sdc -mode fn1 -corner Worst  
  sdc_data -file blockB_test1_b.sdc -mode fn1 -corner Best
```

Default Severity Label

Info

Rule Group

ConstraintFileStructure

Reports and Related Files

None

Const_Struct10

Checks for constraints applied on buffers/inverters

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *Const_Struct10* rule checks for the presence of constraints applied on buffers/inverters or the pins of these cells. Inverters or buffers are the instances of the .lib cells.

Parameter(s)

None

Constraint(s)

The *Const_Struct10* rule supports the following constraints:

<i>create_clock</i>	<i>create_generated_clock</i>	<i>set_annotated_transition</i>
<i>set_case_analyses</i>	<i>set_clock_uncertainty</i>	<i>set_false_path</i>
<i>set_ideal_latency</i>	<i>set_input_delay</i>	<i>set_max_delay/ set_min_delay</i>
<i>set_signal_type</i>	<i>set_multicycle_path</i>	<i>set_operating_conditions</i>
<i>set_output_delay</i>	<i>set_clock_latency</i>	<i>set_disable_timing</i>
<i>set_ideal_transition</i>		

Messages and Suggested Fix

The following message appears:

[WARNING] Constraint <constr> is applied on buffer/inverter(s) <bname> for design/block <name>

Where, <bname> can be a comma-separated names of buffers/inverters.

Potential Issues

The violation message appears when a constraint `<constr>` is applied on buffer/inverter `<bname>` in the design/block `<name>`.

Consequences of Not Fixing

You should avoid setting constraints on these cells or on their pins because the cells may get optimized by the synthesis or logic optimization tools.

How to Debug and Fix

Check the SDC file/line highlighted in the Console GUI. Either remove the constraint or modify it for any other cell/pin.

Example Code and/or Schematic

In this example, assume B1 is a buffer. The `Const_Struct10` rule reports a violation message because the constraint is applied on B1/A.

top.sdc:

```
create_clock -name C1 -period 10 B1/A
```

Default Severity Label

Warning

Rule Group

ConstraintFileStructure

Reports and Related Files

None

Test Rules

The Test Rules Group TestRules contains the following rules:

Rule	Description
<i>Test_Rules01</i>	Missing <code>test_scan_enable</code> port directives
<i>Test_Rules02</i>	Missing <code>test_scan_in</code> port directives
<i>Test_Rules03</i>	Missing <code>test_scan_out</code> port directives
<i>Test_Rules04</i>	Missing <code>set_scan_configuration</code> directives for design units or blocks specified with <i>current_design</i> specification
<i>Test_Rules05</i>	Cell instance pins with <code>test_scan_enable</code> attribute where no <i>set_case_analysis</i> value propagates under the scan mode during the prelayout and postlayout analysis stages
<i>Test_Rules06</i>	Cell instance pins with <code>test_scan_enable</code> attribute where no <i>set_case_analysis</i> value propagates under the functional mode during the postlayout analysis stage

Test_Rules01

Identifies a `test_scan_enable` port that is not defined

When to Use

Verify `test_scan_enable` port is declared in Scan phase

Description

The *Test_Rules01* rule reports missing `test_scan_enable` port directives.

The *Test_Rules01* rule requires the following:

- If test has already been inserted, the `test_scan_enable` port should be defined through a `set_signal_type` constraint.
- If test has not been inserted, then the `test_scan_enable` port should be defined through a `set_scan_signal` constraint.

By default, the *Test_Rules01* rule assumes that test has not been inserted. Set the *scan_insert* rule parameter to indicate that test has been inserted.

Parameter(s)

- *scan_insert*: Default is no. This indicates that test is not inserted. Set the parameter to yes to indicate that test is inserted.

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message appears when there is no `set_signal_type` constraint for a `test_scan_enable` port in the design/block `<name>` and the *scan_insert* parameter is set:

[WARNING] `test_scan_enable` port not defined through `set_signal_type` when test has already been inserted (user-specified) for design/block "`<name>`"

Potential Issues

The `test_scan_enable` port is not defined through a `set_signal_type` constraint.

Consequences of Not Fixing

A missing `test_scan_enable` port can affect testability of the design.

How to Debug and Fix

This violation appears because the value of the `scan_insert` parameter is set to `yes` and `set_signal_type` is not defined for the `test_scan_enable` port.

Update the SDC file as highlighted in the Console GUI.

Message 2

The following message appears when there is no `set_scan_signal` constraint for a `test_scan_enable` port in the design/block `<name>` and the `scan_insert` parameter is not set:

```
[WARNING] scan_insert setting declares test has not been inserted, yet test_scan_enable port is not defined for design/block "<name>"
```

Potential Issues

The `test_scan_enable` port is not defined through a `set_signal_type` constraint.

Consequences of Not Fixing

A missing `test_scan_enable` port can affect testability of the design.

How to Debug and Fix

This violation appears because the value of the `scan_insert` parameter is set to `no` and `set_scan_signal` is not defined for the `test_scan_enable` port.

Update the SDC file as highlighted in Console GUI.

Example Code and/or Schematic

In the following example, suppose the `scan_insert` parameter is set to `no`. This indicates that the test has not been inserted. Therefore, the `Test_Rules01` rule reports a violation for SDC2 because


```
test_scan_enable should be declared through set_scan_signal.  
--Design begin-  
Input in;  
...  
--Design end-  
--SDC1 begin-  
set_scan_signal "test_scan_enable" -port in  
--SDC1 end-  
--SDC2 begin-  
set_signal_type "test_scan_enable" -port in  
--SDC2 end--
```

Now, suppose the *scan_insert* parameter is set to *yes*. This indicates that the test has been inserted. In this case, the *Test_Rules01* rule reports a violation for SDC1 because *test_scan_enable* should be declared through *set_signal_type*.

Default Severity Label

Warning

Rule Group

Test_Rules

Reports and Related Files

None

Test_Rules02

Identifies a test_scan_in port that is not defined

When to Use

To verify that the test_scan_in port is identified in Scan phase

Description

The *Test_Rules02* rule reports missing test_scan_in port directives.

The *Test_Rules02* rule verifies the following:

- If test has already been inserted, then the test_scan_in port should be defined through a set_signal_type constraint.
- If test has already been inserted, then the test_scan_in port has been defined through a set_signal_type constraint, the port should be driving at least one test_scan_in pin of a sequential cell.
- If test has not been inserted, then the test_scan_in port should be defined through a set_scan_signal constraint.
- A test_scan_in pin of a sequential cell should be driven by a test_scan_in port or a test_scan_out pin of a sequential cell.

Parameter(s)

- *scan_insert*: Default is no. This indicates that test is not inserted. Set the parameter to yes to indicate that test is inserted.

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message appears when design/block <name> does not have any test_scan_in port defined using the set_scan_signal constraint and the *scan_insert* parameter is not set:

[WARNING] scan_insert setting declares test has not been

inserted, yet test_scan_in port is not for design/block "`<name>`"

Potential Issues

The test_scan_in port is not defined through the set_scan_signal constraint.

Consequences of Not Fixing

A missing test_scan_in port can affect testability of the design.

How to Debug and Fix

View the incremental schematic of the violation message.

The *Test_Rules02* rule checks whether the test_scan_in port is defined properly. The path from port to sequential cell is highlighted and the missing information is reported in the violation message.

Message 2

The following message appears when design/block `<name>` does not have any test_scan_in port defined using the set_signal_type constraint and the *scan_insert* parameter is set:

```
[WARNING] test_scan_in port not defined through set_signal_type
when test has already been inserted (user-specified) for
design/block "<name>"
```

Potential Issues

The test_scan_in port is not defined through the set_scan_signal constraint.

Consequences of Not Fixing

A missing test_scan_in port can affect testability of the design.

How to Debug and Fix

View the incremental schematic of the violation message.

The *Test_Rules02* rule checks whether the test_scan_in port is defined properly. The path from port to sequential cell is highlighted and the missing information is reported in the violation message.

Message 3

The following message appears when there is no set_signal_type

constraint for a test_scan_in port <port-name> that is driving a test_scan_in pin <pin-name> of sequential cell <cell-name> in design/block <name> and the *scan_insert* parameter is set:

[WARNING] test_scan_in port not defined through set_signal_type for port "<port-name>" which is driving pin "<pin-name>" of sequential cell "<cell-name>", in design/block "<name>"

Potential Issues

A port is driving the test_scan_in pin of a sequential cell but is not defined as a test_scan_in port through the set_signal_type constraint.

Consequences of Not Fixing

A missing test_scan_in port can affect testability of the design.

How to Debug and Fix

View the incremental schematic of the violation message.

The *Test_Rules02* rule checks whether the test_scan_in port is defined properly. The path from port to sequential cell is highlighted and the missing information is reported in the violation message.

Message 4

The following message appears when the test_scan_in pin <pin-name> of sequential cell <cell-name> in design/block <name> is not driven by any test_scan_in port or any test_scan_out pin of a sequential cell:

[WARNING] Pin "<pin-name>" of sequential cell "<cell-name>" is not driven by any sequential cell (having test_scan_out pin) or port, in block "<name>"

Potential Issues

A test_scan_in pin of a sequential cell is not driven by any test_scan_in port or test_scan_out pin of a sequential cell.

Consequences of Not Fixing

A missing test_scan_in port can affect testability of the design.

How to Debug and Fix

View the incremental schematic of the violation message.

The *Test_Rules02* rule checks whether the `test_scan_in` port is defined properly. The path from port to sequential cell is highlighted and the missing information is reported in the violation message.

Message 5

The following message appears when pin `<pin-name>` of sequential cell `<cell-name>` in block `<name>` is being driven by a `test_scan_in` port `<port-name>` but there is no `test_scan_in` attribute set on the pin and the *scan_insert* parameter is set:

```
[WARNING] test_scan_in attribute not set on pin "<pin-name>" of sequential cell "<cell-name>" driven by port "<port-name>" which is defined through set_signal_type, in design/block "<name>"
```

Potential Issues

The `set_signal_type` is set to on. It could be on the wrong port.

Consequences of Not Fixing

A missing `test_scan_in` port can affect testability of the design.

How to Debug and Fix

The *Test_Rules02* rule checks whether the `test_scan_in` port is defined properly. The missing information is reported in the violation message.

Message 6

The following message appears when a `test_scan_in` port `<port-name>` with the `set_signal_type` constraint is not driving any sequential cell having a `test_scan_in` pin in block `<name>` and the *scan_insert* parameter is set:

```
[WARNING] Port "<port-name>" which is defined through set_signal_type is not reaching to any sequential cell (having test_scan_in pin), in block "<name>"
```

Potential Issues

The `set_signal_type` is set to on. It could be on the wrong port.

Consequences of Not Fixing

A missing `test_scan_in` port can affect testability of the design.

How to Debug and Fix

The *Test_Rules02* rule checks whether the `test_scan_in` port is defined properly. The missing information is reported in the violation message.

Example Code and/or Schematic

In the following example, suppose the *scan_insert* parameter is set to no. The *Test_Rules02* rule reports a violation for SDC1 because the `test_scan_in` port should be declared through the `set_scan_signal` constraint.

```
--Design begin-
Input in1, in2, in3;
Flop f1( .test_scan_in( in1 ), ... );
Flop f2( .data( in1 ), ... );
Flop f3( .test_scan_in( in2 ), ... );
...
--Design end-
--SDC1 begin-
--SDC1 end-
--SDC2 begin-
set_signal_type "test_scan_in" -port {in1 in3}
--SDC2 end--
```

Now, suppose the *scan_insert* parameter is set to yes, this rule checks the following scenarios:

1. The `test_scan_in` port should be declared through the `set_signal_type` constraint. For SDC2, this rule does not report a violation for this scenario.
2. The `test_scan_in` port should drive at least one `test_scan_in` pin of the sequential cell. For SDC2, this rule reports a violation for `in3` and not for `in1`.

Test Rules

3. The `test_scan_in` port should not drive any non-`test_scan_in` pin of a sequential cell. For SDC2, this rule reports a violation for the `f2/data` pin.
4. Each `test_scan_in` pin of the sequential cell should be driven by either a `test_scan_in` port or a `test_scan_out` pin of the sequential cell. For SDC2, this rule reports a violation for the `f3/test_scan_in` pin.

Default Severity Label

Warning

Rule Group

Test_Rules

Reports and Related Files

None

Test_Rules03

Identifies an undefined `test_scan_out` port

When to Use

To verify `test_scan_out` ports are identified in the Scan phase.

Description

The *Test_Rules03* rule reports missing `test_scan_out` port directives.

The *Test_Rules03* rule verifies the following:

- If test has already been inserted, then the `test_scan_out` port should be defined through a `set_signal_type` constraint.
- If test has already been inserted, then the `test_scan_out` port has been defined through a `set_signal_type` constraint, the port should be driven by at least one `test_scan_out` pin of a sequential cell.
- If test has not been inserted, then the `test_scan_out` port should be defined through a `set_scan_signal` constraint.
- A `test_scan_out` pin of a sequential cell should be driving a `test_scan_out` port or a `test_scan_in` pin of a sequential cell.

Parameter(s)

- *scan_insert*: Default is no. This indicates that test is not inserted. Set the parameter to yes to indicate that test is inserted.

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message appears when design/block *<name>* does not have any `test_scan_out` port defined using the `set_scan_signal` constraint and the *scan_insert* parameter is not set:

[WARNING] scan_insert setting declares test has not been

inserted, yet `test_scan_out` port is not defined through `set_scan_signal` for design/block "<name>"

Potential Issues

The `test_scan_out` port is not defined through `set_scan_signal` constraint.

Consequences of Not Fixing

A missing `test_scan_out` port can affect testability of the design.

How to Debug and Fix

The *Test_Rules03* rule checks whether the `test_scan_out` port is defined properly. The missing information is reported in the violation message.

Update the SDC file as highlighted in Console GUI.

Message 2

The following message appears when design/block <name> does not have any `test_scan_in` port defined using the `set_signal_type` constraint and the *scan_insert* parameter is set:

```
[WARNING] test_scan_out port not defined through
set_signal_type when test has already been inserted (user-
speci fi ed) for desi gn/bl ock "<name>"
```

Potential Issues

The `test_scan_out` port is not defined through `set_scan_signal` constraint.

Consequences of Not Fixing

A missing `test_scan_out` port can affect testability of the design.

How to Debug and Fix

The *Test_Rules03* rule checks whether the `test_scan_out` port is defined properly. The missing information is reported in the violation message.

Update the SDC file as highlighted in Console GUI.

Message 3

The following message appears when there is no `set_signal_type`

constraint for a test_scan_out port *<port-name>* that is being driven by a test_scan_out pin *<pin-name>* of sequential cell *<cell-name>* in design/block *<name>* and the *scan_insert* parameter is set:

[WARNING] test_scan_out port not defined through set_signal_type for port "<port-name>" which is driven by pin "<pin-name>" of sequential cell "<cell-name>", in design/block "<name>"

Potential Issues

The test_scan_out port is not defined through set_scan_signal constraint.

Consequences of Not Fixing

A missing test_scan_out port can affect testability of the design.

How to Debug and Fix

View the incremental schematic of the violation message.

The *Test_Rules03* rule checks whether the test_scan_out port is defined properly. The path from port to sequential cell is highlighted and the missing information is reported in the violation message.

Update the SDC file as highlighted in Console GUI.

Message 4

The following message appears when the test_scan_out pin *<pin-name>* of sequential cell *<cell-name>* in design/block *<name>* is not driving any test_scan_out port or any test_scan_in pin of a sequential cell:

[WARNING] Pin "<pin-name>" of sequential cell "<cell-name>" is not driving any sequential cell (having test_scan_in pin) or Port, in design/block "<name>"

Potential Issues

The test_scan_out pin is not driving any other scan pin or port. The pin's value is of no use in testability.

Consequences of Not Fixing

A missing test_scan_out port can affect testability of the design.

How to Debug and Fix

View the incremental schematic of the violation message.

The *Test_Rules03* rule checks whether the `test_scan_out` port is defined properly. The path from port to sequential cell is highlighted and the missing information is reported in the violation message.

Update the SDC file as highlighted in Console GUI.

Message 5

The following message appears when pin `<pin-name>` of sequential cell `<cell-name>` in block `<name>` is driving a `test_scan_out` port `<port-name>` but there is no `test_scan_out` attribute set on the pin and the `scan_insert` parameter is set:

```
[WARNING] test_scan_out attribute not set on pin "<pin-name>"
of sequential cell "<cell-name>" driving port "<port-name>"
which is defined through set_signal_type, in design/block
"<name>"
```

Potential Issues

The sequential cell's pin is driving a `test_scan_in` port. the testability for the port might be affected.

Consequences of Not Fixing

A missing `test_scan_out` port can affect testability of the design.

How to Debug and Fix

View the incremental schematic of the violation message.

The *Test_Rules03* rule checks whether the `test_scan_out` port is defined properly. The path from port to sequential cell is highlighted and the missing information is reported in the violation message.

Update the SDC file as highlighted in the Console GUI.

Message 6

The following message appears when a `test_scan_out` port `<port-name>` with the `set_signal_type` constraint is not being driven by any sequential cell having a `test_scan_out` pin in block `<name>` and the `scan_insert` parameter is set:

```
[WARNING] Port "<port-name>" which is defined through
```

set_signal_type is not driven by any sequential cell (having test_scan_out pin), in design/block "<name>"

Potential Issues

The test_scan_out port might be invalid since there is no test_scan_out pin of any sequential cell driving it.

Consequences of Not Fixing

A missing test_scan_out port can affect testability of the design.

How to Debug and Fix

View the incremental schematic of the violation message.

The *Test_Rules03* rule checks whether the test_scan_out port is defined properly. The path from port to sequential cell is highlighted and the missing information is reported in the violation message.

Update the SDC file as highlighted in the Console GUI.

Message 7

The following message appears when the test_scan_out pin <pin-name> of sequential cell <cell-name> in design/block <name> is driving the pin <pin1-name> of sequential cell <cell2-name>:

[WARNING] Pin "<pin-name>" of sequential cell "<cell-name>" is driving "<pin1-name>" of sequential cell "<cell2-name>" which does not have test_scan_in attribute set on it, in design/block "<name>"

Potential Issues

A test_scan_out pin should be connected to only the test_scan_in pin of another sequential cell. Connection to any other pin of a sequential cell could affect path testability.

Consequences of Not Fixing

A missing test_scan_out port can affect testability of the design.

How to Debug and Fix

View the incremental schematic of the violation message.

The *Test_Rules03* rule checks whether the test_scan_out port is defined properly. The path from port to sequential cell is highlighted and the missing information is reported in the violation message.

Update the SDC file as highlighted in the Console GUI.

Example Code and/or Schematic

In the following example, suppose the *scan_insert* parameter is set to no. The *Test_Rules03* rule reports a violation for SDC1 because the `test_scan_out` port should be declared through the `set_scan_signal` constraint.

```
--Design begin-
Output out1, out2, out3;
Flop f1( .test_scan_out( out1 ), ... );
Flop f2( .out( out1 ), ... );
Flop f3( .test_scan_out( out2 ), ... );
...
--Design end-
--SDC1 begin-
--SDC1 end-
--SDC2 begin-
set_signal_type "test_scan_out" -port {out1 out3}
--SDC2 end--
```

Now, suppose the *scan_insert* parameter is set to yes, this rule checks the following scenarios:

1. The `test_scan_out` port should be declared through the `set_signal_type` constraint. For SDC2, this rule does not report a violation for this scenario.
2. The `test_scan_out` port should drive at least one `test_scan_out` pin of the sequential cell. For SDC2, this rule reports a violation for `out3` and not for `out1`.
3. The `test_scan_out` port should not drive any non-`test_scan_out` pin of a sequential cell. For SDC2, this rule reports a violation for the `f2/out` pin.
4. Each `test_scan_out` pin of the sequential cell should drive either a `test_scan_in` port or a `test_scan_in` pin of the sequential cell.

For SDC2, this rule reports a violation for the f3/test_scan_out pin.

Default Severity Label

Warning

Rule Group

Test_Rules

Reports and Related Files

None

Test_Rules04

Identifies `scan_configuration` that is not defined for a constrained design unit

When to Use

To verify the `scan_configuration` declaration for each design unit in the Scan phase

Description

The *Test_Rules04* rule reports missing `set_scan_configuration` constraints for design units or blocks specified with the *current_design* SGDC constraint. The *Test_Rules04* rule requires that a `set_scan_configuration` constraint be present for each design unit or block specified with a *current_design* specification.

Parameter(s)

None

Constraint(s)

SGDC

- `current_design` (Mandatory): Use to define the constraints associated with the design.

SDC

- `set_scan_configuration` (Optional): Use to set scan configuration for a design or block.

Messages and Suggested Fix

The following message appears when there is no `set_scan_configuration` constraint for a design unit or block `<name>` specified with the *current_design* specification:

[WARNING] `set_scan_configuration` not defined for the design `<name>`

Potential Issues

The `set_scan_configuration` constraint is missing for the design.

Consequences of Not Fixing

A missing `set_scan_configuration` constraint can affect testability of the design.

How to Debug and Fix

To define the `set_scan_configuration` constraint, update the SDC file with the following command:

```
set_scan_configuration <design-name>
```

Example Code and/or Schematic

If the `set_scan_configuration` constraint is not specified in the SDC file, the `Test_Rules04` rule reports a violation. In the following example, this rule reports a violation for `top.sdc` and `blk.sdc`.

```
--SGDC begin-  
current_design top_du  
sdc_data -file top.sdc  
  
current_design blk  
sdc_data -file blk.sdc  
--SGDC end-  
  
--top.sdc begin-  
--top.sdc end--  
  
--blk.sdc begin--  
--blk.sdc end--
```

Default Severity Label

Warning

Test Rules

Rule Group

Test_Rules

Reports and Related Files

None

Test_Rules05

Identifies scan mode that is not enabled in the scan constraint file

When to Use

To verify all `test_scan_enable` pins are active high in the Scan phase.

Description

The *Test_Rules05* rule reports cell instance pins with the `test_scan_enable` attribute set, where value 1 does not propagate under the specified mode during Pre-layout and Post-layout Analysis stages.

After scan insertion in scan mode, the *Test_Rules05* rule checks for an active `test_scan_enable` pin in cells. A `test_scan_enable` pin is identified by pin-type in the associated `.lib` description of a cell (which should be named `test_scan_enable`).

The *Test_Rules05* rule checks the pins specified in the associated library files (`.lib` files) that have the `test_scan_enable` attribute set. When all case analysis values are propagated under the specified mode, if any such pins do not have a value 1, the *Test_Rules05* rule flags a message.

Prerequisites

- Use the SpyGlass Library Compiler feature before running the *Test_Rules05* rule because it requires the synthesizable description of all library cells. See the *SpyGlass Explorer User Guide* for details of the SpyGlass Library Compiler feature.
- The *Test_Rules05* rule runs only if the *scan_shift* rule parameter is set to a valid mode name. The mode name is specified with the `-mode` argument of the *sdc_data* constraint.

Parameter(s)

- *scan_insert*: Default is no. This indicates that test is not inserted. Set the parameter to yes to indicate that test is inserted.
- *scan_shift*: Default is unspecified. The *Test_Rules05* rule runs only if this parameter is set to any valid mode name.

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message appears for a cell instance pin `<pin-name>` with `test_scan_enable` attribute set where value 1 does not propagate under the scan mode during Pre-layout and Post-layout Analysis stages:

```
[Test_Rules05_01][WARNING] simulation of constraints (<const>)
does not propagate value "1" to leaf_cell pin <pin-name> (with
test_scan_enabled attribute) of instance <inst-name> of design/
block <name>
```

Where, `<constr>` is the user-specified case analysis constraint(s) and can be `set_case_analysis`, `set_case_analysis/set_signal_type`, or `set_signal_type`. When neither constraint is specified, `<constr>` is not displayed.

Potential Issues

Each `test_scan_enable` pin should be active high.

Consequences of Not Fixing

If the scan mode is not enabled in the scan constraint file, it can affect testability of the design.

How to Debug and Fix

View the incremental schematic of the violation message. It highlights the pin for which the value 1 is not propagated through simulation of the `set_signal_type` and/or `set_case_analysis` constraints. You need to validate the propagated value manually.

Message 2

The following messages appears when the `Test_Rules05` rule is not run as no `-mode` argument is specified in any `sdc_data` constraint:

```
[Test_Rules05_02][WARNING] No scan mode is specified for the
sdsc schema to activate Test_Rule05
```

Potential Issues

The mode argument is missing for the *sdc_data* SGDC constraint. The *Test_Rule05* ignores such an *sdc_data* constraint.

Consequences of Not Fixing

If the scan mode is not enabled in the scan constraint file, it can affect testability of the design.

How to Debug and Fix

In the SGDC file, mention the mode arguments for the *sdc_data* constraints. This is necessary to run the *Test_Rules05* rule.

Message 3

The following messages appears when the *Test_Rules05* rule is not run because the *scan_shift* parameter is not set or is set to a value other than a valid mode name (as specified with the *-mode* argument of the *sdc_data* constraint):

```
[Test_Rules05_03][WARNING] No scan mode is enabled by using scan_shift parameter to activate Test_Rule05 for any sdc schema
```

Potential Issues

The *Test_Rule05* ignores such *sdc_data* constraints.

Consequences of Not Fixing

If the scan mode is not enabled in the scan constraint file, it can affect testability of the design.

How to Debug and Fix

Set the value of the *scan_shift* parameter with a valid modes list. The *Test_Rules05* rule processes *sdc_data* constraints that have a mode value present in the valid mode list.

Example Code and/or Schematic

In the following example, the *Test_Rules05* reports a violation for SDC1 because no mode argument is defined for SDC1. For SDC2, this rule reports a violation because a non-one value is being propagated to the *f1/test_scan_enable* pin.

```
--Design begin--  
Module top( in, ... );  
Input in;
```

Test Rules

```
Flop fl( .test_scan_enable( in ), ...);  
...  
endmodule  
--Design end-  
--SDC1 begin-  
--SDC1 end-  
--SDC2 begin-  
Set_case_analysis 0 in  
--SDC2 end-  
--SGDC begin-  
current_design top  
sdc_data -file SDC1  
sdc_data -file SDC2 -mode scan_top  
--SGDC end--
```

Default Severity Label

Warning

Rule Group

Test_Rules

Reports and Related Files

None

Test_Rules06

Identifies scan mode that is not disabled in the functional mode constraint file

When to Use

To verify that scan pins are disabled in functional modes in the Scan phase.

Description

The *Test_Rules06* rule flags cell instance pins with `test_scan_enable` attribute set, where value 0 does not propagate under the specified mode during prelayout and postlayout analysis stages.

The *Test_Rules06* rule checks that, after scan insertion in functional mode, the `test_scan_enable` pin is active for any cells which have such a pin. A `test_scan_enable` pin is identified by pin-type in the associated .lib description of a cell (which should be named `test_scan_enable`).

The *Test_Rules06* rule checks the pins specified in the associated library files (.lib files) that have the `test_scan_enable` attribute set. When all case analysis values are propagated under the specified mode, if any such pins do not have a value 0, the *Test_Rules06* rule flags a message.

Prerequisites

- Use the SpyGlass Library Compiler feature before running the *Test_Rules05* rule because it requires the synthesizable description of all library cells. See the *SpyGlass Explorer User Guide* for details of the SpyGlass Library Compiler feature.
- The *Test_Rules06* rule runs only if the `scan_shift` parameter is set to a valid mode name. The mode name is specified with the `-mode` argument of the `sdc_data` constraint.

Parameter(s)

- `scan_insert`: Default is no. This indicates that test is not inserted. Set the parameter to yes to indicate that test is inserted.
- `scan_shift`: Default is unspecified. The *Test_Rules05* rule runs only if this parameter is set to any valid mode name.

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message appears for a cell instance pin `<pin-name>` with `test_scan_enable` attribute set where value 0 does not propagate under the scan mode during Pre-layout and Post-layout Analysis stages:

```
[Test_Rules06_01][WARNING] simulation of constraints (<const>) does not propagate value "0" to leaf_cell pin <pin-name> (with test_scan_enabled attribute) of instance <inst-name> of design/block <name>
```

Where, `<constr>` is the user-specified case analysis constraint(s) and can be `set_case_analysis`, `set_case_analysis/set_signal_type`, or `set_signal_type`. When neither constraint is specified, `<constr>` is not displayed.

Potential Issues

An incorrect `set_case_analysis/set_signal_type` value could propagate invalid value to the scan pin.

Consequences of Not Fixing

If the scan mode is not disabled in the functional mode constraint file, it can affect testability of the design.

How to Debug and Fix

View the incremental schematic of the violation message. It highlights the pin for which a 0 value is not propagated through simulation of the `set_signal_type` and/or `set_case_analysis` constraints. You need to validate the propagated value manually.

Message 2

The following messages appears when the `Test_Rules06` rule is not run as no `-mode` argument is specified in any `sdc_data` constraint:

```
[Test_Rules06_02][WARNING] No scan mode is specified for the sdc schema to activate Test_Rule06
```

Potential Issues

The mode argument is missing for the *sdc_data* SGDC constraint. *Test_Rule06* ignores such an *sdc_data* constraint.

Consequences of Not Fixing

If the scan mode is not disabled in the functional mode constraint file, it can affect testability of the design.

How to Debug and Fix

In the SGDC file, mention the mode arguments for the *sdc_data* constraints. This is necessary to run the *Test_Rules06* rule.

Example Code and/or Schematic

In the following example, the *Test_Rules06* rule reports a violation for SDC1 because no mode argument is defined for SDC1. For SDC2, this rule reports a violation because a non-zero value is propagated to the f1/test_scan_enable pin.

```
--Design begin-
Module top( in, ... );
Input in;
Flop f1( .test_scan_enable( in ), ...);
...
endmodule
--Design end-
--SDC1 begin-
--SDC1 end-
--SDC2 begin-
set_case_analysis 1 in
--SDC2 end-
--SGDC begin-
current_design top
sdc_data -file SDC1
sdc_data -file SDC2 -mode scan_top
```

Test Rules

--SGDC end--

Default Severity Label

Warning

Rule Group

Test_Rules

Reports and Related Files

None

Timing Checking Rules

The following timing checking rules are available:

Rule	Description
<i>Check_Timing02</i>	Pairs of interacting clocks when their periods are not expandable with respect to each other
<i>Check_Timing03</i>	Generated-clock networks with cycles/loops
<i>Derate01</i>	Reports missing early/late arguments of the <i>set_timing_derate</i> constraint on a design object

Check_Timing02

Reports clock pairs that do not have expandable periods with respect to each other

When to Use

Use this rule in the RTL, pre-layout, and post-layout phases.

Description

The *Check_Timing02* rule reports pairs of interacting clocks when their periods are not expandable with respect to each other.

The *Check_Timing02* rule computes the LCM of clocks and reports if either LCM of clocks is greater than 1000 times of the smaller clock period or greater than 101 times of longer clock period.

Rule Exception(s)

The *Check_Timing02* rule checks only interacting clock domains, that is, when there is at least one path from one clock domain to the other.

Parameter(s)

None

Constraint(s)

- *set_clock_sense* (Optional): Use this constraint to stop clock propagation with the option `-stop_propagation`.

Messages and Suggested Fix

Message

The following message appears for clocks, *<clk1-name>* and *<clk2-name>* when their clock periods *<period1>* and *<period2>*, respectively are not expandable with respect to each other:

[WARNING] Cl ocks "*<clk1-name>* (peri od *<period1>*)" and "*<clk2-name>* (peri od *<period2>*)" are unexpandabl e

Potential Issues

The violation message appears when interacting clock periods do not have expandable clock periods with respect to each other.

Consequences of Not Fixing

If asynchronous clocks with unexpandable periods are interacting, then timing would fail in most cases.

How to Debug and Fix

Check if periods of clocks are defined correctly since they are not expandable with respect to each other.

Example Code and/or Schematic

Suppose the clocks in the following snippet are interacting or crossing, this rule reports a violation because the LCM odd periods $> 101 * 11.33$, where LCM of numbers are 11536.91.

```
//test.sdc  
create_clock -name CLK1 -period 11.33 [get_ports {clk1}]  
create_clock -name CLK2 -period 10.27 [get_ports {clk2}]
```

Default Severity Label

Warning

Rule Group

Check_Timing

Reports and Related Files

None

Check_Timing03

Reports a cycle or a loop in the generated clock network

When to Use

Use this rule in the RTL, pre-layout, and post-layout phases.

Description

The *Check_Timing03* rule reports generated-clock network with cycles/loops.

The *Check_Timing03* rule checks whether there are any cycles in the generated-clock network.

Parameter(s)

None

Constraint(s)

- [create_clock/create_generated_clock](#) (Mandatory): Use this constraint to create a clock.

Messages and Suggested Fix

The following message appears for clocks *<clk-name-list>* when a loop exists in the generated-clock network:

[WARNING] Clocks '*<clk-name-list>*' form a loop in the generated-clock network

Potential Issues

The violation message appears when clocks form loop in the generated-clock network.

Consequences of Not Fixing

If you do not fix this violation, one clock is feeding the other without a master clock source because of the loop whereas the master clock must be driven by a clock source. Such clock network is not feasible physically.

How to Debug and Fix

Ensure to break the loop between the clocks mentioned in the violation message.

Example Code and/or Schematic

Following is the code snippet for *Check_Timing03* rule:

```
create_clock -name C1 -period 10.000000 -waveform { 0.000000
5.000000 } {clk1 clk2}

create_generated_clock -name G1 -multiply_by 2 -source clk1
{clk3 clk4}

create_generated_clock -name G2 -multiply_by 2 -source clk3
{clk5}

create_generated_clock -name G3 -multiply_by 2 -source clk4
{clk2}

create_generated_clock -name G4 -multiply_by 2 -source clk2
{clk1 clk3}
```

In the above example, G1—G3—G4 are forming loop hence the rule reports a violation.

Default Severity Label

Warning

Rule Group

Check_Timing

Reports and Related Files

None

Derate01

Reports missing early/late arguments of the `set_timing_derate` constraint on a design object

When to Use

Use this rule in the RTL, pre-layout, and post-layout phases.

Description

The *Derate01* rule reports `set_timing_derate` constraints that do not have either the `early` or `late` arguments specified. The design objects affected are also reported.

Parameter(s)

None

Constraint(s)

- `set_timing_derate` (Mandatory): Use this constraint to specify the delay derating factors for the current design or a specified list of instances.

Messages and Suggested Fix

The following message appears when a `set_timing_derate` constraint does not have both `early` and `late` arguments specified:

```
[WARNING] set_timing_derate '<early | late>' missing for
'<design | cell | library-cell | net>' '<object-name>', objects
affected : <list-of-objects-affected>
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

When the `early` or `late` argument is not defined, it leads to assumptions in STA tools that might be unintended. For correct timing analysis, both options should be defined.

How to Debug and Fix

Double-click the violation message. The SDC file appears and the `set_timing_derate` command that has the missing `early` and `late` arguments

is highlighted.

To resolve this violation, ensure the [set_timing_derate](#) constraint specification in the SDC file has both the early and late argument.

Example Code and/or Schematic

For the following SDC snippet, the *Derate01* rule reports a violation because the late argument is missing.

```
create_clock -period 10 -waveform {0 5} -name CLK1 [get_ports  
CLK]
```

```
set_timing_derate 0.2 -cell_delay -early
```

To resolve the violation, specify the late argument, as shown.

```
create_clock -period 10 -waveform {0 5} -name CLK1 [get_ports  
CLK]
```

```
set_timing_derate 0.2 -cell_delay -early
```

```
set_timing_derate 1.2 -cell_delay -late
```

Default Severity Label

Warning

Rule Group

None

Reports and Related Files

None

Timing Coverage Rules

The Timing Coverage Rules Group `TimingCoverage` contains the following rule:

Rule	Description
<i>SDC_Coverage</i>	Generates a report that displays uncovered design objects.

SDC_Coverage

Compute design coverage and report uncovered design objects and timing paths

When to Use

To determine whether an SDC file is covering the whole design. This rule is applicable to all stages in the design.

Description

The *SDC_Coverage* rule provides a quantified measure of the timing-readiness of a design and reports uncovered design objects and timing paths. This rule applies a set of SDC constraints specified using the *sdc_data* SGDC constraint and reports the coverage of the design in terms of uncovered design objects, such as ports and registers, and timing paths.

The criteria for coverage is:

- Input/Output/Inout port is covered if all the paths leading to other design objects are covered.
- Register is covered if all paths leading to its input clock pins are covered.
- Timing paths are reported as uncovered if it is falsified in all the modes. Applicable only in multimode.

A design is classified as timing-ready if it meets the timing directives that are specified using the SDC constraints. A design object is covered if all paths between the design object are constrained or considered irrelevant through the *set_false_path* or *set_multicycle_path* constraint.

Prerequisites

The following table lists the SpyGlass Constraints rules that are run to generate the Coverage Report:

Constraint Type	Rules Run
set_case_analysis	NA
Clock	<i>Clk_Gen01</i>
set_input_delay	<i>Inp_Del01</i>
set_output_delay	<i>Op_Del01</i>

Coverage Paths

The *SDC_Coverage* rule considers the following paths:

- Port to port
 - Ports constrained with the *set_input_delay/set_output_delay* or *set_max_delay/set_min_delay* constraints.
- Port to register
 - Port to non-clock pins of a register. The port is constrained with the *set_input_delay* constraint.
 - Port to clock pin of a register. The port is constrained with clock.
- Register to register
 - The Q pin of a register reaches the CP pin of the ending register. The clock pin of the start point register should be constrained with a clock
- Register to port
 - Port constrained with the *set_output_delay* constraint.

Single-Mode and Multimode Analysis

The *SDC_Coverage* rule considers the *-mode* argument of the *sdc_data* constraint and generates a summarized and detailed *Timing Constraints Coverage Report* for each *sdc_data* constraint. This mode of analysis is called single-mode analysis. In single-mode analysis, the *SDC_Coverage* rule processes the *set_case_analysis* constraints. This rule also considers the constant or blocked pins while computing the coverage of design objects. In addition, this rule does not report black boxes, hanging terminals, or net terminals for unconstrained ports.

A multimode analysis is also performed in which the *SDC_Coverage* rule considers all *sdc_data* constraints for the design. The uncovered design objects in all *sdc_data* constraints are reported. In multimode analysis, the *SDC_Coverage* rule ignores the *set_case_analysis* constraints. However, the rule reports traversals reaching black boxes and/or hanging nets and terminals.

NOTE: *If multiple *sdc_data* constraints that have the same mode value are specified, the *SDC_Coverage* rule only process the first of these *sdc_data* constraints and ignores the others.*

Parameter(s)

- *tc_regression_mode*: Default is 0 and the regression mode is disabled. This indicates that the timing exceptions spreadsheet contains all SDC and schematic back references. When you enable regression mode by setting this parameter to 1, SDC and schematic back references are not created in the timing exceptions spreadsheet.
- *tc_report_backref_all*: Default is 0 and only the first 10,000 rows have SDC and schematic back references. This saves runtime. Set this parameter to 1 to report all SDC and schematic back references in the timing exceptions spreadsheet.

Constraint(s)

SDC

- *set_case_analysis* (Optional): Use to specify whether a port or a pin is at the constant logic value of 1 or 0, or is considered with a rising or falling transition.
- *set_false_path* (Optional): Use to identify paths in a design to be marked as false and therefore not considered during the timing analysis.
- *set_input_delay* (Optional): Use to define the arrival time relative to a clock.
- *set_output_delay* (Optional): Use to set the output path delay values for the current design.
- *set_multicycle_path* (Optional): Used to define the multi-cycle path.

SGDC

- *sdc_data* (Mandatory): Use to specify the SDC file for a design module.

Messages and Suggested Fix

The following message appears when a *Timing Constraints Coverage Report* for the design/block <name> is generated:

[INFO] Timing coverage report for design/block <name> prepared

Potential Issues

The report points to the areas not covered. These could be uncovered due to missing constraints.

Consequences of Not Fixing

If the potential design flaw is not fixed, it will lead to incomplete timing analysis of the design. It might also lead to the silicon failure.

How to Debug and Fix

To view the report, double-click the message. The [Timing Constraints Coverage Report](#) single-mode or multi-mode, appears as a separate window called Spreadsheet Viewer.

If the *SDC_Coverage* rule is for a single-mode, one report is generated. However, if this rule is run for multiple modes, a merged report of all the modes is generated. For more information, refer to [Single-Mode and Multimode Analysis](#).

The naming convention of the report is as follows:

- For single mode analysis: `<design-name>_<mode-value>.rpt`
- For multimode analysis: `<design-name>_mergedmode.rpt`
- When `-mode` is not specified: `<design-name>.rpt`

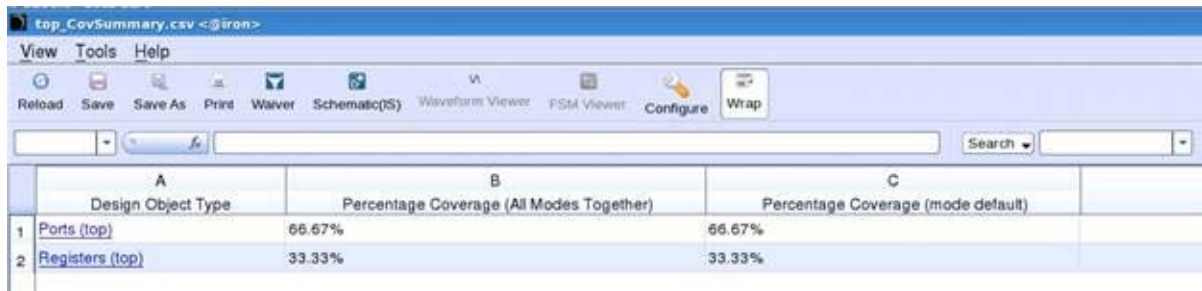
In addition, refer to the [Type of Information in the Timing Constraints Coverage Report](#) section for information on the worksheets that are part of the Coverage Report.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows the Timing Constraints Coverage report that is generated by the *SDC_Coverage* rule. A sample report is as shown in the following screen grab:



The screenshot shows a window titled 'top_CovSummary.csv <@Iron>' with a menu bar (View, Tools, Help) and a toolbar (Reload, Save, Save As, Print, Waiver, Schematic(S), Waveform Viewer, PSM Viewer, Configure, Wrap). Below the toolbar is a search bar. The main content is a table with the following data:

	A	B	C
	Design Object Type	Percentage Coverage (All Modes Together)	Percentage Coverage (mode default)
1	Ports (top)	66.67%	66.67%
2	Registers (top)	33.33%	33.33%

Default Severity Label

Info

Rule Group

TimingCoverage

Reports and Related Files

- [Timing Constraints Coverage Report](#)

Timing Exception Rules

The Timing Exception Rule Group `TimingException` contains the following sub-groups:

- *xBuf*
- *False Path Rules*
- *Multi-Cycle Path Rules*
- *Other Timing Exception Rules*

xBuf

The buffer rule group XBuf contains the following rule:

Rule	Description
<i>XBuf01</i>	Detects RTL nets where X-buffers can be inserted for each <i>set_multicycle_path</i> constraint

XBuf01

Detects nets for X-Buffer insertion

When to Use

This rule is applicable to the RTL phase.

Description

The *XBuf01* rule detects RTL nets where X-buffers can be inserted for each *set_multicycle_path* constraint.

Prerequisite(s)

The *XBuf01* rule requires both the SpyGlass Constraints license and the SpyGlass TXV license to run successfully.

Parameter(s)

- *tc_xbuf_ignore_clks*: Default value is `yes`. By default, the *XBuf01* rule ignores the *set_multicycle_path* constraints that have clocks specified in the `-from/-to` lists. To have the *XBuf01* rule consider these commands, set the value to `no`.

Constraint(s)

- *set_multicycle_path* (Mandatory): Use this constraint to set the number of cycles required to reach data for the given path.

Messages and Suggested Fix

The following message appears when the X-buffer information for design `<name>` is generated in the file `<file-name>`:

```
[INFO] X-Buffer info for design <name> is generated in file  
<file-name>
```

Where, `<file-name>` is the CSV file that contains the X-buffer information.

Potential Issues

The message appears when the X-buffer information for design is generated in the file.

Consequences of Not Fixing

Not applicable.

How to Debug and Fix

View the generated report for information on the buffers required for these multi-cycle paths.

Example Code and/or Schematic

For the following snippet, the *XBuf01* rule generates a report for this multi-cycle path.

```
//test.v
module top(in1, in2, clk1, out1);
    input in1, in2, clk1;
    output out1;

    assign out1 = in1 & in2;
endmodule

//test.sdc
set_multicycle_path 2 -from {in1, in2} -to out1 -hold -fall
```

The rule will report for this multi-cycle path in the report file.

Default Severity Label

Info

Rule Group

XBuf

Reports and Related Files

- *xbuf_info01.csv*: Contains details of the X-buffers for each [sdc_data](#) command.

False Path Rules

The False Path Rules sub-group `False_Path` contains the following rules:

Rule	Description
<i>False_Path01</i>	Unconnected False Path reference points
<i>False_Path03</i>	Redundant <code>-through</code> argument in <i>set_false_path</i> constraint
<i>False_Path04</i>	<i>set_false_path</i> constraints that exclude more than specified number of paths
<i>False_Path04a</i>	<i>set_false_path</i> constraints that exclude more than specified number of paths
<i>False_Path04b</i>	Identifies false path specifications that exceed a specific value
<i>False_Path07</i>	Asynchronous clock domains that have connecting paths but no <i>set_false_path</i> constraint set
<i>False_Path08</i>	Asynchronous clock domains that have connecting paths and a <i>set_multicycle_path</i> constraint set but no <i>set_false_path</i> constraint set
<i>False_Path09</i>	<i>set_false_path</i> commands that use the same clock in their <code>-from</code> and <code>-to</code> lists
<i>False_Path10</i>	<i>set_false_path/set_multicycle_path</i> commands where more than two <code>-through</code> options have been used
<i>False_Path11</i>	<i>set_false_path</i> commands where <code>-from</code> , <code>-to</code> or both options are not specified
<i>False_Path12</i>	<i>set_false_path</i> commands where any of the <code>-setup -rise</code> , <code>-setup -fall</code> , <code>-hold -rise</code> , and <code>-hold -fall</code> options are missing

False_Path01

Identifies false path reference points that are not connected

When to Use

To check the correctness of false paths specified in the SDC files. This rule is applicable to all phases of the design cycle.

Description

The *False_Path01* rule identifies unconnected false path reference points by checking for existence of paths specified using the `-from`, `-to`, or `-through` arguments of the *set_false_path* constraint in the SDC files. The unconnected points are listed in the *False_Path01 Report*.

By default, timing exceptions are not propagated through clear and preset arcs of sequential cells. However, if you set the *tc_enable_preset_clear_arcs* parameter to `yes`, clear and preset pins are not valid end points and timing exceptions are propagated through clear and preset arcs of sequential cells. This support is added only for net list designs with library cells that have "preset-to-q" and "clear-to-q" arcs. It will not work in RTL designs.

Parameter(s)

- *tc_enable_preset_clear_arcs*: Default is `no`. Set this parameter to `yes` to enable preset and clear arcs of sequential cells.
- *strict*: Default is `no`. This indicates that this rule reports unconnected points in a specified exception. When this parameter is set to `yes` or `False_Path01`, this rule reports a violation only if all the specified points are unconnected. If at least one point is connected, this rule does not report any violation.
- *tc_report_unconnected_points*: Default is 5. This indicates that a maximum of 5 unconnected points are reported for exceptions specified in the SDC file. You can set this parameter to an integer between 0 and 1000 to increase or decrease this limit. For example, if you set this parameter to 20, up to 20 unconnected points are reported in the generated CSV file.

Constraint(s)

SDC

- `set_false_path` (Mandatory): Use to denote a timing path as false.

Messages and Suggested Fix

The following message appears for a `set_false_path` constraint after the CSV is generated:

[WARNING] CSV file <file-name> generated to report unconnected points for design/block <name> for `set_false_path`

Potential Issues

This violation is reported because either the `set_false_path` is blocked or disabled. Alternatively, there might not be any connectivity between the objects being referenced.

Consequences of Not Fixing

The `set_false_path` constraints are ignored by the Static Timing Analysis (STA) tool. This results in longer runtime for implementation tools.

How to Debug and Fix

Double-click the violation message to open the CSV file in Spreadsheet Viewer. This file contains the list of unconnected reference points. A sample CSV file is as follows:

	B	C	D	E
	All points unconnected	Type	Unconnected points	File Name:Line No.
☐ 1	Yes		Points that are not connected with any specified points in <code>set_false_path</code>	top.sdc:3
▪ 2		-from	f1/CP	
▪ 3		-through	a1/A	
▪ 4		-to	out2	

You can also view the incremental schematic. The schematic highlights all objects stated in the violation message. The objects referenced in the

`set_false_path` do not form a valid timing path, including all the `from/through/to` list objects.

Verify the violation using the **Show input cone** or **Show output cone** feature of Schematic Viewer. Check whether the objects specified in the `from/through/to` list exist in the fan-in/fan-out cone.

In addition, run the `Show_Case_Analysis` rule to check whether the timing path is being blocked.

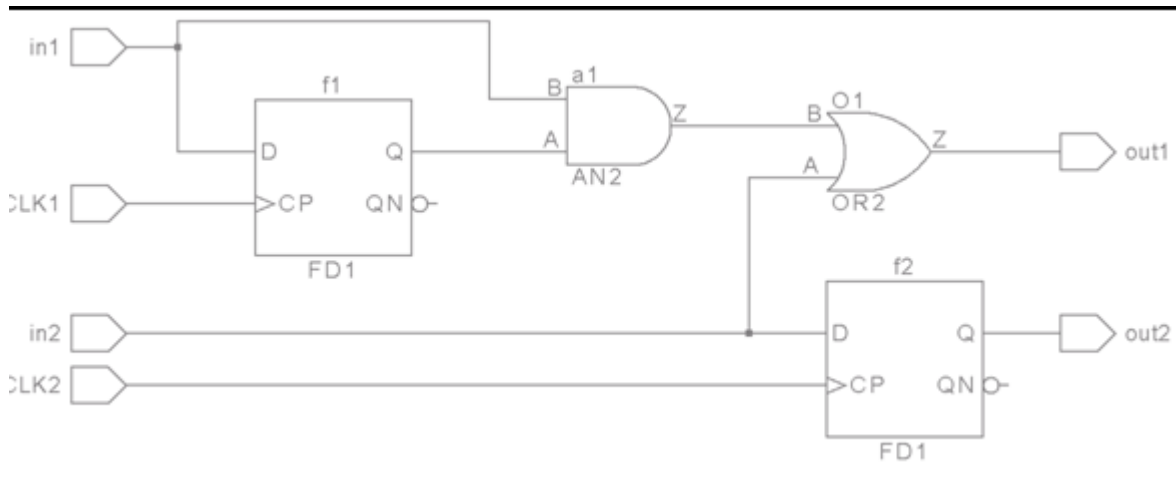
To resolve this violation, update the SDC file to create a valid false path.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows the use of the `False_Path01` rule to identify invalid timing paths in a false path specification.



The following SDC specification is provided:

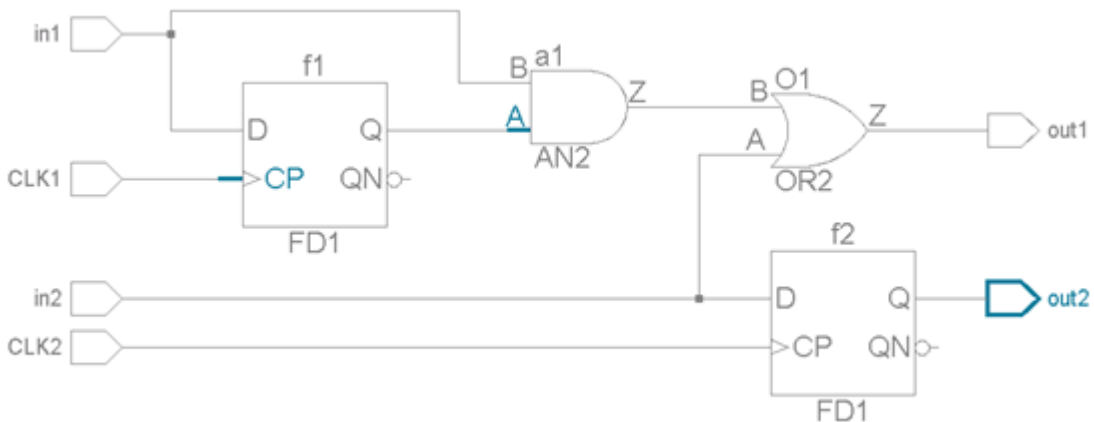
```
set_false_path -from {f1/CP} -through {a1/A} -to {out2}
```

The `False_Path01` rule reports a violation because the path in the

Timing Exception Rules

set_false_path specification does not exist. The path from f1/CP to out2 through a1/A is not valid because from the a1/Z pin, the path goes to out1 through the O1/B pin.

The schematic generated highlights all objects that form the invalid timing path.

**Example 2**

Test Case Files Not Available

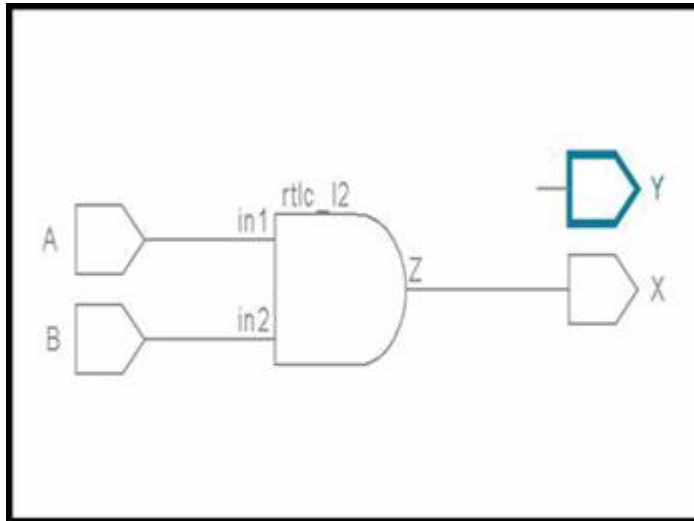
Consider the following specification:

```
set_false_path -from {A B} -to {X Y}
```

In this example, the *False_Path01* rule expects the following paths to exist:

1. Path starting from A and ending at either X or Y
2. Path starting from B and ending at either X or Y
3. Path starting from either A or B and ending at X
4. Path starting from either A or B and ending at Y

In this example, the *False_Path01* rule reports a violation for object Y because only paths A->X and B->X exist, as shown by the following image.



If the example was modified to have only paths A->X and A->Y, the *False_Path01* rule would report a violation for B. Similarly, if only paths A->X and B->Y existed, this rule would not report a violation as all required paths exist.

Default Severity Label

Warning

Rule Group

False_Paths

Reports and Related Files

[False_Path01 Report](#)

False_Path03

Reports unnecessary use of `-through` in false path

When to Use

To eliminate unnecessary use of the `through` option to improve run time of synthesis and static timing analysis. This rule is applicable to all phases of the design cycle.

Description

The *False_Path03* rule reports redundant `-through` argument specification in the *set_false_path* constraints of the SDC files. To reduce redundancy through specifications, avoid specifying the `-through` specification for all possible paths between the points given by the `-from` and `-to` arguments.

Parameter(s)

None

Constraint(s)

SDC

- *set_false_path* (Mandatory): Use to denote a timing path as false.

Messages and Suggested Fix

The following message appears when the `-through` argument specified for objects `<obj-list>` in a *set_false_path* constraint is unnecessary:

[WARNING] Unnecessary use of `-through` { `<obj-list>` }

Potential Issues

There is redundant use of the `-through` argument for some objects.

Consequences of Not Fixing

Unnecessary `-through` points increases the runtime of tools and causes synthesis tools to treat the `through` points as `dont_touch`. Therefore, these points are not optimized during synthesis.

How to Debug and Fix

Design object specified in the `through` option of the `set_false_path` constraint is not required because there is only one path from an object in the previous `from/through` option to the next `through/to` option. The violation message states the object present in the `-through` list which is not required.

Update the SDC file as highlighted.

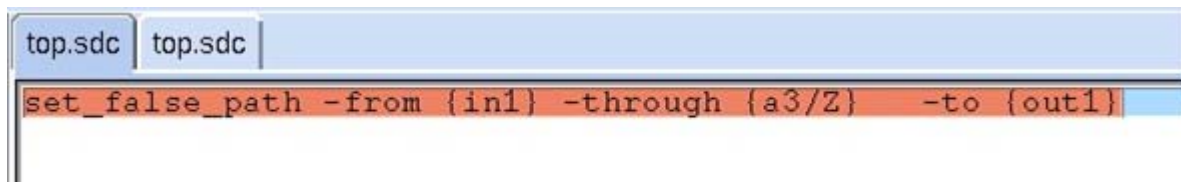
Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows the use of the `False_Path03` rule to identify redundant `-through` argument specifications in the `set_false_path` constraints of the SDC files.

In SDC file, the constraint with the redundant `-through` argument is highlighted.



The screenshot shows a code editor with two tabs labeled 'top.sdc'. The main window displays a single line of SDC code: `set_false_path -from {in1} -through {a3/Z} -to {out1}`. The entire line is highlighted in red, indicating a violation or error.

Example 2

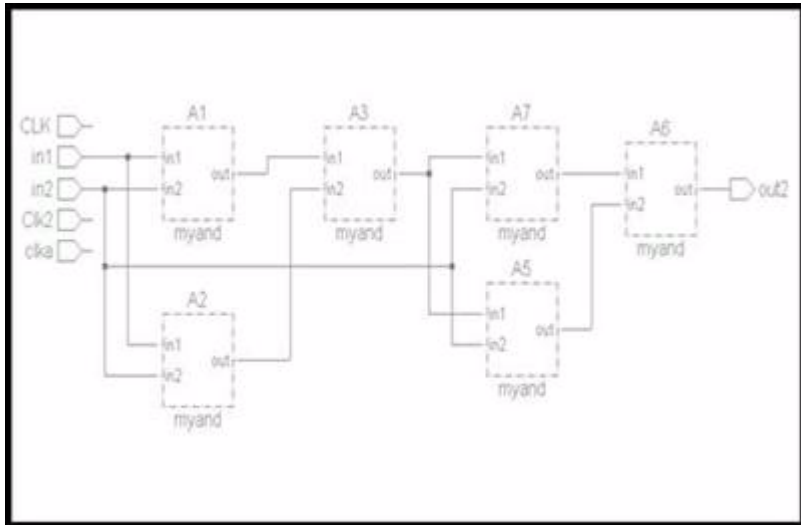
Test Case Files Not Available

In the following SDC file snippet, the `set_false_path` constraint is specified as follows:

```
set_false_path -through {in1} -through {A3/out} -through out2
```

For all paths from `in1` to `out1`, the `A3/out` point is common. Therefore, the `A3/out` point is redundant and the `False_Path03` rule reports a violation message.

Timing Exception Rules

**Default Severity Label**

Warning

Rule Group

False_Paths

Reports and Related Files

None

False_Path04

Reports if the number of paths are more than the specified number

When to Use

Use this rule to reduce timing analysis.

Description

The *False_Path04* rule reports *set_false_path* constraints that cover more paths than the specified number of paths. The *False_Path04* checks all *set_false_path* constraints except those set between clocks objects (real, virtual, or generated) in order to avoid heavy over-reporting.

Rule Exception(s)

The *False_Path04* rule does not report the total number of paths if the value of the total number of paths is greater than 1.5 times the value specified using the *num_falsepath_max* parameter.

Parameter(s)

- *num_falsepath_max*: The default value is 20. Use any positive integer to set a different number of paths to be covered.

Constraint(s)

- *set_false_path* (Mandatory): Use this constraint to specify the paths between clocks.

Messages and Suggested Fix

Message

The following message appears showing the report generated for each SDC file, where at least one path in the *set_false_path/set_multicycle_path* constraint covers paths greater than the parameter limit *<falsepath_limit>*:

```
[WARNING] All false paths excluding paths greater than
parameter limit <falsepath_limit> for design <des-name> given
in file <file-name>
```

Potential Issues

This violation message appears if your design has at least one path which covers paths greater than the parameter list.

Consequences of Not Fixing

If you do not fix this violation, the tool may take huge time because the false path may be covering more than the intentional paths.

How to Debug and Fix

Ensure that you comment on the constraint or specify each intended [set_false_path](#) constraint in the SDC file.

Example Code and/or Schematic

Consider the following snippet.

```
set_false_path -from {5 elements} -to {4 elements}
```

The *False_Path04a* rule would simply perform the calculation $5*4$ and return 20 paths whereas in the *False_Path04* rule, it traverse to find the actual number of timing paths that match the given command.

The *False_Path04* rule also generates a report in a spread sheet (.csv) format with the violation message. The report is named as:
<modulename>-fp_cover_path_info<x>.csv

where, <module-name> is the name of the design and <x> is a unique unsigned integer. The report displays related multi-cycle paths where covering path is less than the specified parameter limit <falsepath_limit>.

To view the report you click the violation message.

SDC File Name	SDC File Line No	False Path	Paths Covered
test.sdc	3	from {a} -to {c}	
test1.sdc	5	from {a1} -to {c1}	14

The report contains the following information:

- SDC File Name

- SDC File Line No
- False Path
- Paths Covered: If the reported false path is less than 1.5 times of the specified *<falsepath_limit>* parameter, Paths Covered column shows the actual paths. Otherwise, this column is left blank.

Default Severity Label

Warning

Rule Group

False_Path

Reports and Related Files

None

False_Path04a

Identifies false path specifications that exclude a large number of paths

When to Use

To detect paths that should not have been excluded by the `set_false_path` constraint. This rule is applicable to the RTL phase.

Description

The *False_Path04a* rule reports `set_false_path` constraints that cover more paths than the number of paths specified in the `num_falsepath_max` parameter. Refer to the *Parameter(s)* section to control the behavior of this rule.

Refer to the *Example Code and/or Schematic* section to understand the difference between the *False_Path04* and the *False_Path04a* rules.

Parameter(s)

- `num_falsepath_max`: Default is 20. The *False_Path04a* rule counts the number of paths being covered as the number of `from/to` endpoint pairs implied by the specification. When only `-through` points are specified, this rule counts the implied `from/to` end-points. While counting, a `-from` clock (or a `-to` clock) is considered as only one point. After all paths have been counted, the *False_Path04a* rule compares this total count against the value of the `num_falsepath_max` parameter and reports a message if the total count is more than the value set.

Constraint(s)

SDC

- `set_false_path` (Mandatory): Use to denote a timing path as false.

Messages and Suggested Fix

The following message appears for a `set_false_path` constraint that excludes `<num>` paths where `<num>` is more than the `$num_falsepath_max` value:

[WARNING] False path excludes <num> paths which is above the parameter limit of \$num_falsepath_max

Potential Issues

The violation message indicates the number of paths, which are greater than the limit you have specified in the [num_falsepath_max](#) parameter, for the [set_false_path](#) constraint.

Consequences of Not Fixing

Potentially, a large number of timing paths may be excluded.

How to Debug and Fix

Review the SDC file as highlighted and provide the intended objects in the false path. In addition, avoid using a wildcard to reduce the number of timing paths. Lastly, you can increase the limit specified in the [num_falsepath_max](#) parameter.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows the violation reported by the *False_Path04a* rule. In this example, the [num_falsepath_max](#) parameter is set to 0, as shown.

```
30      ##### Parameters #####
31      set_parameter num_falsepath_max 0
```

Example 2

Test Case Files Not Available

In this example the difference between the [False_Path04](#) and the *False_Path04a* rules is highlighted.

The [False_Path04](#) and the *False_Path04a* rules both report a large number of paths, but the way they count the paths is different. Hence, you should run only one of the rules.

The [False_Path04](#) rule counts the actual number of timing paths identified through traversal whereas the *False_Path04a* rule uses heuristics to

Timing Exception Rules

perform the count of the timing paths. Therefore, the *False_Path04a* rule saves time by reducing the number of traversals.

In the following example, the *False_Path04a* rule would simply perform the calculation $5*4$ and return 20 paths whereas in the *False_Path04* rule, traversal is done to find the actual number of timing paths that match the given command.

```
set_false_path -from {5 elements} -to {4 elements}
```

Default Severity Label

Warning

Rule Group

False_Paths

Reports and Related Files

None

False_Path04b

Identifies false path specifications that exceed a specific value

When to Use

This is more of a caution. The [set_false_path](#) constraint should not exceed a user-specified limit. This rule is applicable to the RTL phase.

Description

The *False_Path04b* rule reports a violation if the number of [set_false_path](#) commands specified in the SDC file exceeds the value set by the [tc_num_fp_max](#) parameter.

To understand the difference between the *False_Path04* and the *False_Path04a* rules, refer to the [Example Code and/or Schematic](#) section.

Parameter(s)

- [tc_num_fp_max](#): Default is 0. Set the value to the maximum number of [set_false_path](#) commands that can be specified in the SDC file

Constraint(s)

- [set_false_path](#) (Mandatory): Use to denote a timing path as false.

Messages and Suggested Fix

The following message appears when the number of [set_false_path](#) commands in the SDC file exceeds the value set by [tc_num_fp_max](#):

[WARNING] Number of set_false_path commands <value1> given for design/block <name> is greater than the specified limit of <value2>

Potential Issues

The violation message explicitly states the potential issue.

Consequences of Not Fixing

The runtime of timing analysis tools increases significantly as the number of [set_false_path](#) constraints increase. This is a methodology rule.

How to Debug and Fix

All the [set_false_path](#) commands are referenced in the violation message.

Review the SDC file, as highlighted in the Console GUI, and ensure no redundant `set_false_path` commands are specified.

In addition, you can increase the limit specified in the `tc_num_fp_max` parameter.

Example Code and/or Schematic

Suppose, the `tc_num_fp_max` parameter is set to two. For the following snippet, the `False_Path04b` rule reports a violation the number of `set_false_path` commands is three.

```
...  
set_false_path -from in1 -to out1  
set_false_path -from in2 -to out2  
set_false_path -from in3 -to out3  
...
```

Default Severity Label

Warning

Rule Group

False_Paths

Reports and Related Files

None

False_Path07

Reports when set_false_path is not specified between two connected asynchronous clock domains

When to Use

Use this rule in all the phases of designing.

Description

The *False_Path07* rule reports asynchronous clock domains with connecting paths but *set_false_path*, *set_clock_groups*, or *set_multicycle_path* constraints not set.

The *False_Path07* rule processes the *clock_group* constraint, if specified.

Prerequisite(s)

Specify the *clock_group* information in the SGDC file. If you do not specify the *clock_group* information in the SGDC file, the *False_Path07* rule does not report a violation.

Rule Exception(s)

The *False_Path07* rule does not consider the *set_false_path* constraints having -through list.

Parameter(s)

- *tc_clk_compat*: Default value is no. Set this parameter to Yes to consider the path from the port to D pin of flip-flop.
- *tc_setup_hold*: The default value is both. Set the value to -setup or to -hold to restrict the checking of the *set_false_path* constraints.

Constraint(s)

- *set_false_path* (Mandatory): Use this constraint to specify the paths between the clocks.
- *set_clock_groups* (Optional): Use this constraint to specify the *clock_group* information.
- *set_multicycle_path* (Optional): Use this constraint to specify the *clock_group* information.

Messages and Suggested Fix

Message 1

The following message appears when a combinational path exists between the clock domain `<clk1-name>` to its asynchronous clock `<clk2-name>` and a `set_false_path` or `set_clock_groups` constraint is not set:

```
[False_Path07_01][WARNING] Expected but did not find
set_false_path or set_clock_groups from clock <clk1-name> to
its asynchronous clock <clk2-name>
```

Potential Issues

This violation message appears if your design has a combinational path between the clock domains to its asynchronous clock but `set_false_path` or `set_clock_groups` constraint is not set.

Consequences of Not Fixing

If you do not fix this violation, the toll consumes more analysis timing for the crossing asynchronous clocks, though the tool should not perform analysis for synchronous clocks.

How to Debug and Fix

You can see the constraints in the SDC and the schematic of clock crossings. Ensure that you either specify both clocks in the same domain or apply the `set_false_path` constraint between the clocks.

Message 2

The following message appears when a combinational path exists between the clock domain `<clk1-name>` to its asynchronous clock `<clk2-name>` and a `set_false_path` constraint is set without the `<option>` option:

```
[False_Path07_02][WARNING] Missing <option> option for
set_false_path from <clk1-name> to its asynchronous clock
<clk2-name>
```

Where, `<option>` can be `-setup` or `-hold`.

Potential Issues

This violation message appears if your design has a combinational path between the clock domain to its asynchronous clock and `set_false_path` is set without option.

Consequences of Not Fixing

If you do not fix this violation, the toll consumes more analysis timing for the crossing asynchronous clocks, though the tool should not perform analysis for synchronous clocks.

How to Debug and Fix

You can see the constraints in the SDC and the schematic of clock crossings. Ensure that you either specify both clocks in the same domain or apply the [set_false_path](#) constraint between the clocks.

Message 3

The following message appears if the design `<name>` does not have [clock_group](#) information specified:

```
[False_Path07_03][WARNING] Rule will not violate for design  
<name> since clock_group information is not specified in SGDC  
file
```

Potential Issues

This violation message appears if your design has not specified the [clock_group](#) constraint information in the SGDC file.

Consequences of Not Fixing

If you do not fix this violation, the tool consumes more analysis timing.

How to Debug and Fix

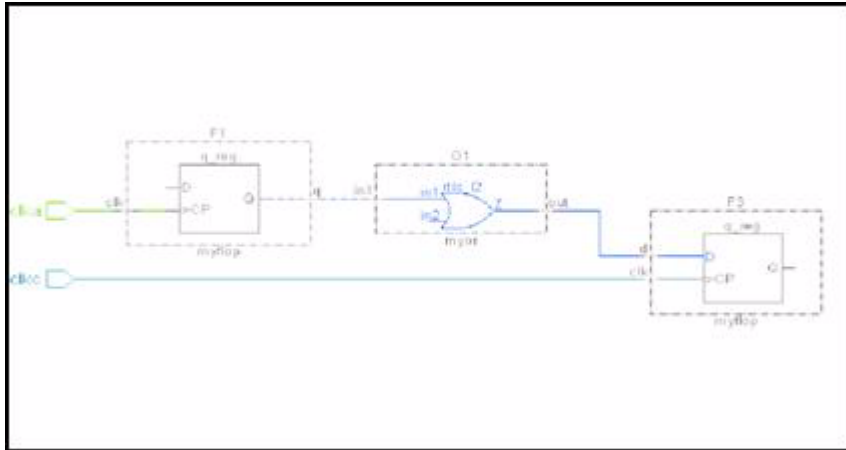
On clicking this violation, user can see the SGDC file having no domain information specified.

If you do not specify domains of clocks, the tool will infer domains of clocks by its own.

Example Code and/or Schematic

Consider the following design.

Timing Exception Rules



In this design, clocks applied on the ports c1 (on clka) and c2 (on clk) are asynchronous as per domain constraint specified in the SGDC file. If you do not specify the *set_false_path* constraint and the clocks are crossing, the rule reports a violation.

Suppose, you change the domain as given below, this rule will not report a violation because c1 & c2 are synchronous.

```
//test.sgdc
domain -name d1 -clock {c1 c2}
```

Default Severity Label

Warning

Rule Group

False_Path

Reports and Related Files

- *clk_domain_false_path Report*: The *False_Path07* rule generates the following details in this report:
 - List of clock names by clock domain.

- ❑ Existence of a path in both directions between each pair of clocks of different domains.
- ❑ Whether the *set_false_path*, *set_clock_groups*, *set_multicycle_path* constraints are set for each pair of the clocks of different domains in both directions.
- ❑ Whether the *set_false_path* constraint is specified with only one of the `-setup` and `-hold` options.

False_Path08

Reports if false path is not specified between the two connected asynchronous clock domains

When to Use

Use this rule in the RTL, pre-layout, and post-layout phases.

Description

The *False_Path08* rule reports *set_multicycle_path* constraint specified between two asynchronous clock domains that have connecting paths and do not have *set_false_path* or *set_clock_groups* constraint set.

Prerequisite(s)

Specify the *clock_group* information in the SGDC file to run this rule.

Rule Exception(s)

The *False_Path08* rule does not consider *set_false_path* or *set_multicycle_path* constraints having `-through` option.

Parameter(s)

- *tc_clk_compat*: The default value is `no`. Set the value to `yes` for the rule to consider the path from the port to the data pin of a flip-flop as a case of clock domain interaction.

Constraint(s)

- *set_false_path* (Mandatory): Use this constraint to specify the paths between the clocks.
- *set_clock_groups* (Optional): Use this constraint to specify the *clock_group* information.
- *set_multicycle_path* (Optional): Use this constraint to specify the *clock_group* information.

Messages and Suggested Fix

Message 1

The following message appears when a combinational path exists between

the clock domain `<clk1-name>` to its asynchronous clock `<clk2-name>` and a `set_multicycle_path` constraint is specified between two asynchronous clock domains but no `set_false_path` or `set_clock_groups` constraint set:

[False_Path08_01][INFO] No `set_false_path` or `set_clock_groups` for combinational path from clock `<clk1-name>` to its asynchronous clock `<clk2-name>`, but a `set_multicycle_path` exist

Potential Issues

The message explicitly states the potential issues.

Consequences of Not Fixing

If you do not fix the violation, timing analysis does analysis for futile paths and consumes more time.

How to Debug and Fix

You can see the `set_multicycle_path` constraint and clock crossing path. Either remove the `set_multicycle_path` constraint or specify the clocks as synchronous.

Message 2

The following message appears if the `clock_group` information is not specified in the SGDC file for a design `<name>`:

[False_Path08_02][INFO] Rule will not violate for design `<name>` since `clock_group` information is not specified in SGDC file

Potential Issues

The message explicitly states the potential issues.

Consequences of Not Fixing

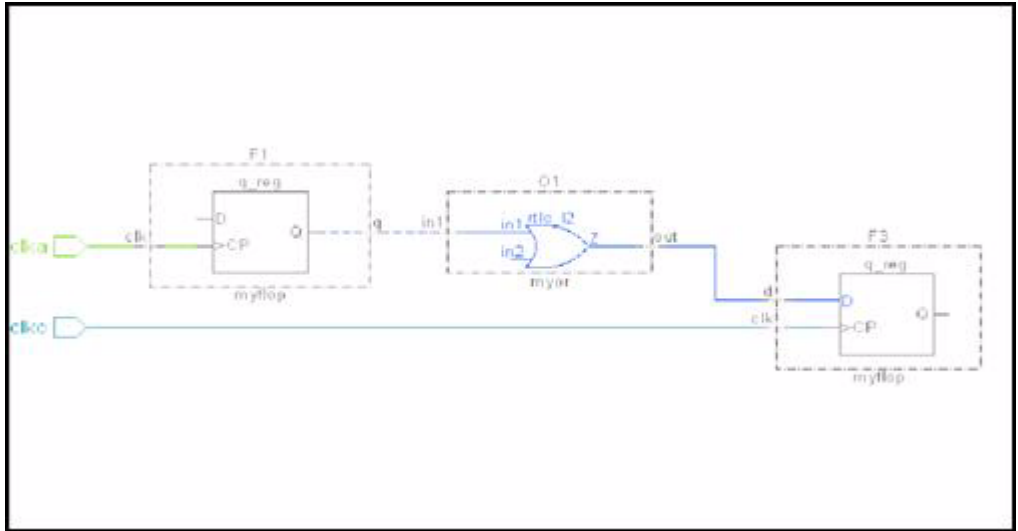
If you do not fix the violation, the tool performs analysis on all paths between clocks and consumes more time.

How to Debug and Fix

- Click on the violation message.
- You can see the SGDC file having no domain information specified.
- Ensure to provide the domain information.

Example Code and/or Schematic

Consider the following schematic.



In the above schematic, clocks applied on ports c1(on clk_a) and c2(on clk_c) are asynchronous as per domain constraint specified in the sgdc file. These clocks are crossing and the *set_multicycle_path* constraint is specified in the SDC file. Hence, this rule reports a violation.

Suppose, you change the domain as given below:

```
//test.sgdc
Domain -name d1 -clock {c1 c2}
```

In this case, this rule does not violate because c1 & c2 are synchronous.

Default Severity Label

Info

Rule Group

False_Path

Reports and Related Files

None

False_Path09

Reports false path that use the same clock in its -from and -to lists

When to Use

This rule is applicable to all phases in the design cycle.

Description

The *False_Path09* rule reports [set_false_path](#) commands that use the same clock in their -from and -to lists.

Parameter(s)

None

Constraint(s)

SDC

- [set_false_path](#) (Mandatory): Use to denote a timing path as false.

Messages and Suggested Fix

The following message appears at the location of a [set_false_path](#) constraint that uses the same clock(s) `<clk-name-list>` in both the -from and -to lists:

[ERROR] Same clock(s) `<clk-name-list>` is specified in both -from and -to list of false path command

Potential Issues

If a false path is specified between the same clocks, all the paths starting from that clock and ending on that clock are falsified. There is no reason why you should do that.

Consequences of Not Fixing

Potentially, you are not performing timing for paths between the same clocks.

How to Debug and Fix

The SDC file highlights the false path constraint.

Check the [set_false_path](#) constraint for a clock that is mentioned in both the

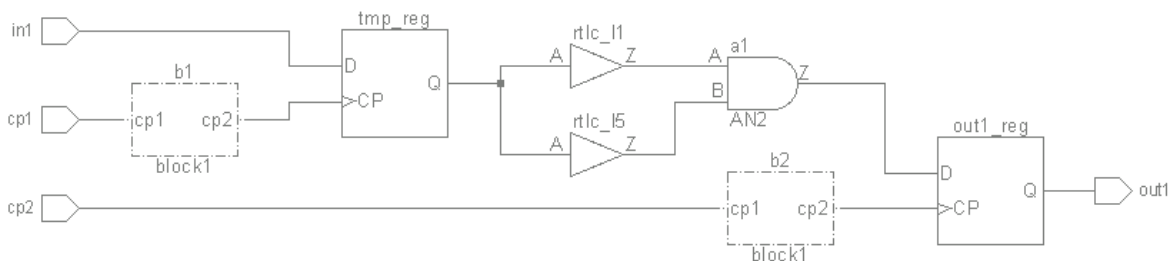
from and to options.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top .sdc is as follows:

```
top.sdc top.sdc
create_clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000} {cp1}
create_clock -name clk2 -period 10.000000 -waveform { 0.000000 5.000000} {cp2}
set_false_path -from {clk1} -to {clk1} -through a1/Z
set_false_path -from {clk2} -to {clk2} -through a1/A
```

The violation message appears because same clock has been specified for the -from and -to arguments.

Example 2

Test Case Files Not Available

The *False_Path09* rule reports a violation for the following *set_false_path* commands as highlighted:

```
set_false_path -from clkA -to clkB
```

Timing Exception Rules

```
set_false_path -from {clkA} -to {clkA} -through A1/out  
set_false_path -from {clkA clkB} -to {clkB clkA}
```

Default Severity Label**Error****Rule Group**

False_Paths

Reports and Related Files

None

False_Path10

Reports a violation for multiple -through options used

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *False_Path10* rule reports when more than two -through options are used in the *set_false_path/set_multicycle_path* commands.

Parameter(s)

None

Constraint(s)

- *set_false_path* (Mandatory): Use this constraint to specify the paths between the clocks.
- *set_multicycle_path* (Optional): Use this constraint to specify the clock_group information.

Messages and Suggested Fix

Message

The following message appears if a *set_false_path* constraint uses *<num>* -through options and *<num>* is more than two:

[INFO] Multiple(*<num>*) -through options are specified for set_false_path command

Potential Issues

This violation message appears if your design has multiple -through options specified for the *set_false_path* constraint.

Consequences of Not Fixing

If you do not fix the violation, the runtime of implementation tools increases.

How to Debug and Fix

Ensure to specify one `-through` option for the `set_false_path` constraint.

Example Code and/or Schematic

Consider the following snippet.

```
set_false_path -from f1/CP -through A/Z -through B/Z -through  
C/Z
```

In this case, the rule reports a violation because the number of `through` arguments mentioned are 3, which exceed the specified limit by 2.

Default Severity Label

Info

Rule Group

False_Path

Reports and Related Files

None

False_Path11

Reports when options `-from` or `-to` or both are not specified

When to Use

Use this rule in all the phases of designing.

Description

The *False_Path11* rule reports when `-from`, `-to` or `-both` options are not specified in the *set_false_path* constraints.

False paths should ideally be specified using both `-from` and `-to` options. This reduces the possibility of specifying a single cycle path as false path.

Single cycle path specified as false path leads to sub-optimization because the STA tool gets relaxed for such paths.

Parameter(s)

None

Constraint(s)

- *set_false_path* (Mandatory): Use this constraint to specify the paths between the clocks.

Messages and Suggested Fix

Message

The following message appears if `-from` or `-to` options are not specified in the *set_false_path* constraint:

```
[INFO] Option(s) <options-list> not specified for  
set_false_path command
```

Where, *<options-list>* is that name of the option, which is not specified.

Potential Issues

The violation message appears if `-from` or `-to` options are not specified in the *set_false_path* constraint.

Consequences of Not Fixing

If you do not fix this violation, the timing tool consumes more time in analysis as the path can have multiple `-from` and `-to` points.

How to Debug and Fix

Ensure that you specify at least one of the following options: `-from`, `-to`, or `-both`.

Example Code and/or Schematic

Consider the following snippet.

```
set_false_path -through A/Z
```

For this snippet, many paths may exist that are passing through the point. Hence, you should specify either `-from/ -to` or both `-from` and `-to` points.

Default Severity Label

Info

Rule Group

False_Path

Reports and Related Files

None

False_Path12

Reports violation for the options missing in a false path

When to Use

Use this rule to check the completeness of false paths.

Description

The *False_Path12* rule reports when *set_false_path* constraint is not specified with any one of the following options:

- `-setup -rise`
- `-setup -fall`
- `-hold -rise`
- `-hold -fall`

If the *set_false_path* constraint is set with `-to`, `-from` and `-through` options, all paths with each combination of `-to`, `-from` and `-through` options are considered separately.

Parameter(s)

None

Constraint(s)

- *set_false_path* (Mandatory): Use this constraint to set the objects in `-to`, `-from`, and `-through` options.

Messages and Suggested Fix

Message

The following message appears if the options *<options-list>* are not specified in the *set_false_path* constraint:

```
[WARNING] Option(s) <options-list> not specified for
set_false_path: <path-specification>
```

Where, *<options-list>* is a comma-separated list of the missing options and *<path-specification>* is the list of false paths.

Potential Issues

The violation message appears options are not specified in the [set_false_path](#) constraint.

Consequences of Not Fixing

If you do not fix the violation, timing analyzer does the analysis for the path falsified by the other options.

How to Debug and Fix

Ensure to specify all the options for the falsified path.

Example Code and/or Schematic

Consider the following snippet.

```
set_false_path -setup -from in1
```

In this case, the rule reports a violation for the `-hold -rise` and `-hold -fall` arguments because they are not specified in the constraint.

Default Severity Label

Warning

Rule Group

False_Path

Reports and Related Files

None

False_Path13

Reports a loop path at a generate clock that has false path defined

When to Use

Use this rule to automatically check loop paths for timing violations.

Description

The *False_Path13* rule detects and reports generated clocks, which have false path defined, and have a loop path. Since the loop path is falsified, the path does not need to meet timing requirements.

Parameter(s)

None

Constraint(s)

- *set_false_path* (Optional): Use this constraint to set the objects in `-to`, `-from`, and `-through` options.

Messages and Suggested Fix

Message

The following message appears if there is a loop path at a generate clock, which has a false path defined:

[WARNING] Loop path at generated clock <gen-clock-name> on instance <instance-name> exists, which have false path defined.

Potential Issues

The violation message appears because loop paths, which are falsified, on generated clocks. These paths can be avoided for checking timing violations.

Consequences of Not Fixing

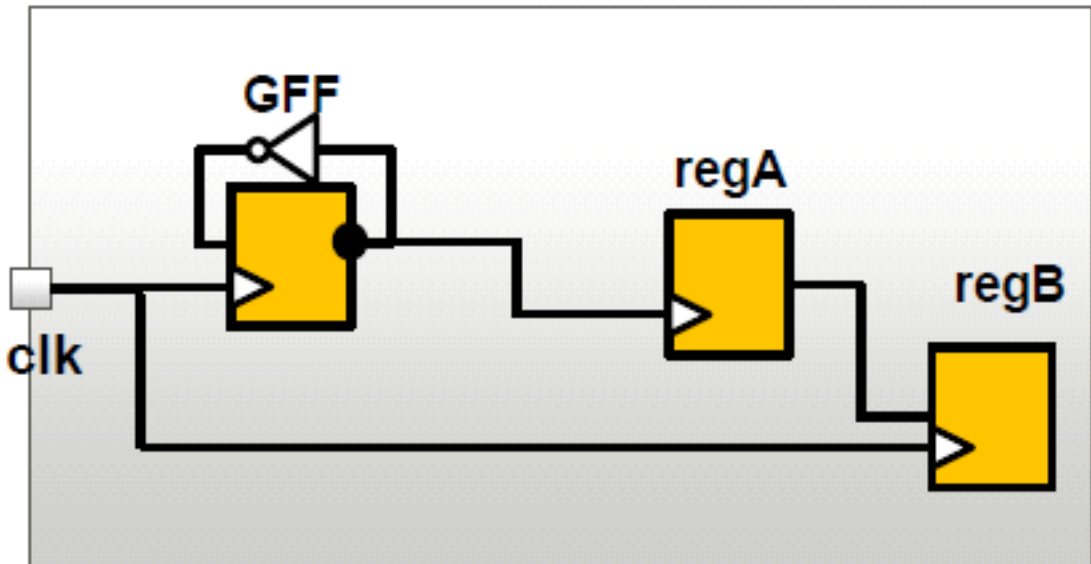
This increases redundancy because additional timing checks are performed.

How to Debug and Fix

View the loop path in schematic viewer. In addition, review the SDC file.

Example Code and/or Schematic

The following schematic illustrates a loop back from GFF/Q to GFF/D.



This rule reports the loop back if the generated clock is falsified, as shown in the following SDC snippet:

```
create_clock -name CLK [get_ports clk]
create_generated_clock -name GCLK [get_pins GFF/Q]
set_false_path -from GCLK -to CLK
```

Default Severity Label

Warning

Rule Group

False_Path

Reports and Related Files

None

Multi-Cycle Path Rules

The Multi-Cycle Path Rules Sub-group MCP contains the following rules:

Rule	Description
<i>CheckMCP</i>	Domain relationship information between clocks used in multi-cycle path commands
<i>MCP01</i>	Unconnected Multi-Cycle Path reference points
<i>MCP03</i>	Redundant -through argument in <i>set_multicycle_path</i> constraint
<i>MCP04</i>	<i>set_multicycle_path</i> constraints that exclude more than specified number of paths
<i>MCP04a</i>	<i>set_multicycle_path</i> constraints that exclude more than specified number of paths
<i>MCP04b</i>	Reports multi-cycle path specifications that exceed a specific value
<i>MCP05</i>	Incorrect specification of setup and hold of Multi-Cycle paths
<i>MCP08</i>	<i>set_multicycle_path</i> command where from, to or both arguments are not specified
<i>MCP09</i>	<i>set_multicycle_path</i> command where any of the required options are not specified for a multi-cycle path
<i>MCP_B2BCons01</i>	Reports multi-cycle paths spanning more than one block

CheckMCP

Prints domain information between clocks used in multicycle path commands

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *CheckMCP* rule prints the domain relationship information between clocks used in multi-cycle path commands in a report *mcp_domain_<integer> Report*. Here, *<integer>* is a number which is unique for each *sdc_data*. The report is located in the *spyglass_reports/constraints/* directory.

If the value of the *tc_checkmcp_report_sorted_clock* rule parameter is set to *yes*, then the *mcp_domain_<integer>.rpt* report displays the instance names sorted with respect to clocks.

The *CheckMCP* rule infers the domain information from the SGDC file specified using the *clock_group* constraint. However, if the *clock_group* information is not specified in the SGDC file, the *CheckMCP* rule infers the domain information from the SDC file.

Prerequisite(s)

The *CheckMCP* rule uses the SpyGlass TXV license.

Parameter(s)

- *tc_checkmcp_report_sorted_clock*: Default value is *no*. Set the value to *yes*, and the *mcp_domain_<integer>.rpt* report displays the instance names sorted with respect to clocks.
- *tc_ignore_block_path*: Default is *yes*. Set the value to *no* so that the *set_case_analysis*, input delays, and/or output delays are also considered.

Constraint(s)

- *clock_group* (Optional): Use this constraint to specify the clock relationship (domain, synchronous/asynchronous, exclusive).

Messages and Suggested Fix

Message 1

The following message appears for the *CheckMCP* rule:

```
[INFO] set_multicycle_path with -from domain (<clk1-list>) and
-to domain (<clk2-list>)
```

Message 2

The following message appears for each design, SDC file, and provides the report file containing the domain information for the multicycle path commands:

```
[INFO] Clock Domain Information of set_multicycle_path commands
for design <design-name>, schema Line : <line-num> is generated
in <report-location>
```

Potential Issues

Not applicable

Consequences of Not Fixing

Not applicable

How to Debug and Fix

For Message 2. Double-click the violation message to view the report.

Example Code and/or Schematic

Consider a case where an SGDC file `test.sgdc` is given as follows:

```
current_design
  sdc_data -file test.sdc
```

The SDC file `test.sdc` contains the following:

```
create_clock -name CLK1 -period 10 [get_ports clk1]
create_clock -name CLK2 -period 12 [get_ports clk2]
set_multicycle_path 3 -from {a_reg/CP} -to {b_reg/D}
```

Assume that register `a_reg` is driven by `clk1` and `clk2` whereas register `b_reg` is driven by `clk1`.

In such scenario, the *CheckMCP* rule prints the domain information between clocks `clk1` and `clk2` in the report `mcp_domain_<integer> Report`.

```
//test.sdc
create_clock -name clk1 -period 1000 -waveform { 0 500 }
[get_pins clock_gen/p11_lib_inst/ousbdummyclk]

create_clock -name scan_clk -period 10000 -waveform { 0 5000
} [get_ports scan_clk_in]

set_multicycle_path -setup 2 -from [get_pins I1/CK] -to
[get_pins I2/D]
```

Suppose no domain information is given in the sgdc file. Therefore, the rule picks up information from *DomainAnalysis* rule and reports an info message.

Default Severity Label

Info

Rule Group

MCP

Reports and Related Files

- [mcp_domain_<integer> Report](#): Generated by the *CheckMCP* rule. It displays the domain relationships between clocks used the [set_multicycle_path](#) constraint.

MCP01

Identifies multi-cycle path reference points that are not connected

When to Use

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *MCP01* rule reports unconnected multi-cycle paths. This rule checks for the existence of paths specified using the `-from`, `-to`, or `-through` arguments of *set_multicycle_path* constraint in the SDC files.

Parameter(s)

- *strict*: Default is `no`. This indicates that this rule reports unconnected points in a specified exception. When this parameter is set to `yes` or `MCP01`, this rule reports a violation only if all the specified points are unconnected. If at least one point is connected, this rule does not report any violation.
- *tc_report_unconnected_points*: Default is 5. This indicates that a maximum of 5 unconnected points are reported for exceptions specified in the SDC file. You can set this parameter to an integer between 0 and 1000 to increase or decrease this limit. For example, if you set this parameter to 20, up to 20 unconnected points are reported in the generated CSV file.

Constraint(s)

- *set_multicycle_path* (Mandatory): Use to modify the single-cycle timing relationship of a constrained path.

Messages and Suggested Fix

The following message appears for a *set_multicycle_path* constraint after the CSV is generated:

```
[WARNING] CSV file <file-name> generated to report unconnected points for design/block <name> for set_multicycle_path
```

Potential Issues

The objects referenced in the *set_multicycle_path* constraint do not form a

valid timing path (including all the from/through/to list objects). This violation may be reported because either the path is blocked or there is no connectivity between the objects being referenced.

Consequences of Not Fixing

The [set_multicycle_path](#) constraint would be ignored by the Static Timing Analysis (STA) tool. This results in longer runtime for implementation tools.

How to Debug and Fix

Double-click the violation message to open the CSV file in Spreadsheet Viewer. This file contains the list of unconnected reference points. A sample CSV file is as follows.

	B	C	D	E
	All points unconnected	Type	Unconnected points	File Name:Line No.
☰ 1	Yes		Points that are not connected with any specified points in set_multicycle_path	top.sdc:5
• 2		-from	clk1	
• 3		-to	f1/CP	

You can also view the incremental schematic of the violation message. The schematic shows the objects as stated in the violation message.

Verify the violation using the **Show fan-in/Show fan-out cone** feature of the schematic viewer. Then, check whether the objects specified in the from/through/to list exist in the fan-in/fan-out cone or not.

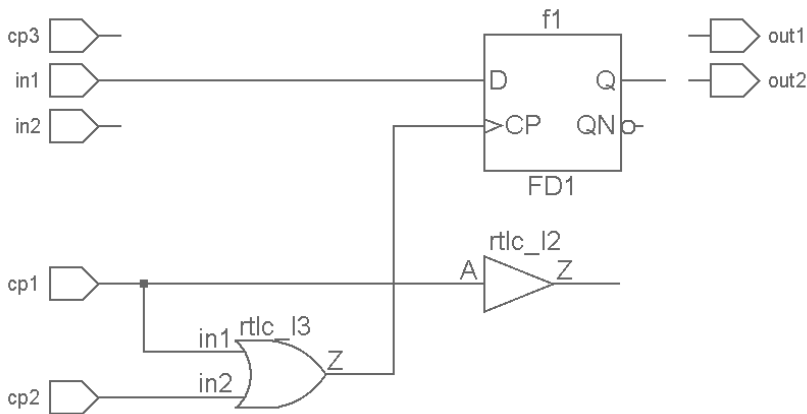
Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:

Timing Exception Rules



The top.sdc is as follows:

```
top.sdc | top.sdc
create_clock -name clk1 -period 10.000000 -waveform { 0.000000 5.000000 } {cp1}
create_clock -name clk2 -period 10.000000 -waveform { 0.000000 5.000000 } {cp2}
create_clock -name clk3 -period 10.000000 -waveform { 0.000000 5.000000 } {cp3}
set_output_delay 0.0 -clock clk3 {f1/CP}
set_multicycle_path -from {clk1} -to {f1/CP} 3
```

The violation message appears because no path exists from clock clk1 to pin f1/CP. In the top.sdc file, the corresponding constraint is shown.

Example 2

Test Case Files Not Available

Consider the following specification:

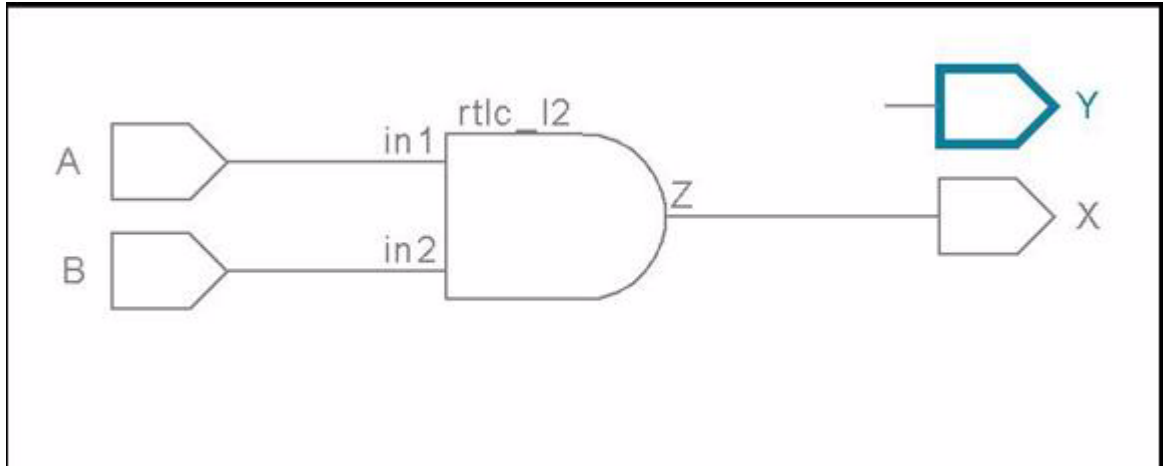
```
set_multicycle_path -from {A B} -to {X Y}
```

The *MCP01* rule expects the following paths to exist:

1. Path starting from A and ending at either X or Y
2. Path starting from B and ending at either X or Y
3. Path starting from either A or B and ending at X

4. Path starting from either A or B and ending at Y

If only paths A->X and A->Y exist, then the MCP01 rule flags for B. Similarly, if only paths A->X and B->X exist, then the MCP01 rule flags for Y. If only paths A->X and B->Y exist, then the MCP01 rule does not flag as all required paths exist.



Similarly, the *MCP01* rule checks for paths when the `-through` argument is also specified.

Default Severity Label

Warning

Rule Group

MCP

Reports and Related Files

- [MCP01 Report](#)

MCP03

Identifies redundant -through in the multi-cycle path

When to Use

To eliminate unnecessary use of the `through` option to improve run time of synthesis and static timing analysis. This rule is applicable to all design stages.

Description

The MCP03 rule reports redundant `-through` argument specification in the [set_multicycle_path](#) constraints. If you have specified the `-through` specification for all possible paths between points identified by `-from` and `-to` arguments of a [set_multicycle_path](#) constraint, all of the `-through` specifications are redundant.

Parameter(s)

None

Constraint(s)

- [set_multicycle_path](#) (Mandatory): Use to modify the single-cycle timing relationship of a constrained path.

Messages and Suggested Fix

The following message appears when the `-through` argument specified for objects `<obj-list>` in a [set_multicycle_path](#) constraint is redundant:

```
[WARNING] Unnecessary use of -through specification { <obj-list> }
```

Potential Issues

The violation messages explicitly states the potential issues.

Consequences of Not Fixing

Having redundant `-through` points increases the runtime of tools and also causes synthesis tools to treat them as `dont_touch`. Therefore, use only those points that are important.

How to Debug and Fix

The design object specified in the `-through` option of the `set_multicycle_path` constraint is redundant because there is only one path from an object in the previous `from/through` option to the next `through/to` option. Validate the timing path between the previous `from/through` option to the next `through/to` option and ensure only one path exists.

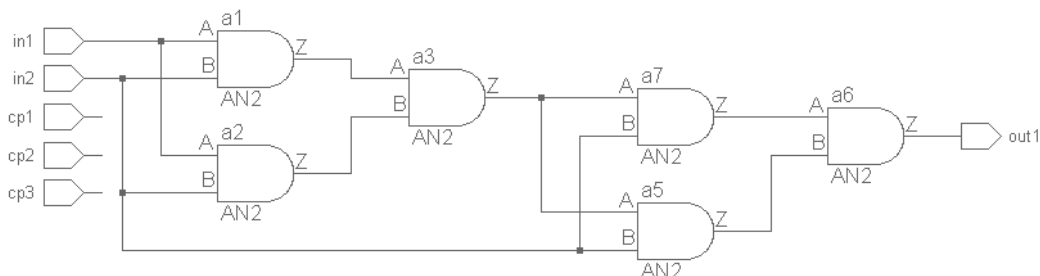
Update the `through` specification for the `set_multicycle_path` constraint in the SDC file as highlighted.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top.sdc is as follows:

```
top.sdc top.sdc
set_multicycle_path 2 -from {in1} -through {a3/Z} -to {out1}
```

The violation message indicates that the `-through` argument with the `set_multicycle_path` constraint is not required. In the top.sdc file, the

corresponding constraint is shown.

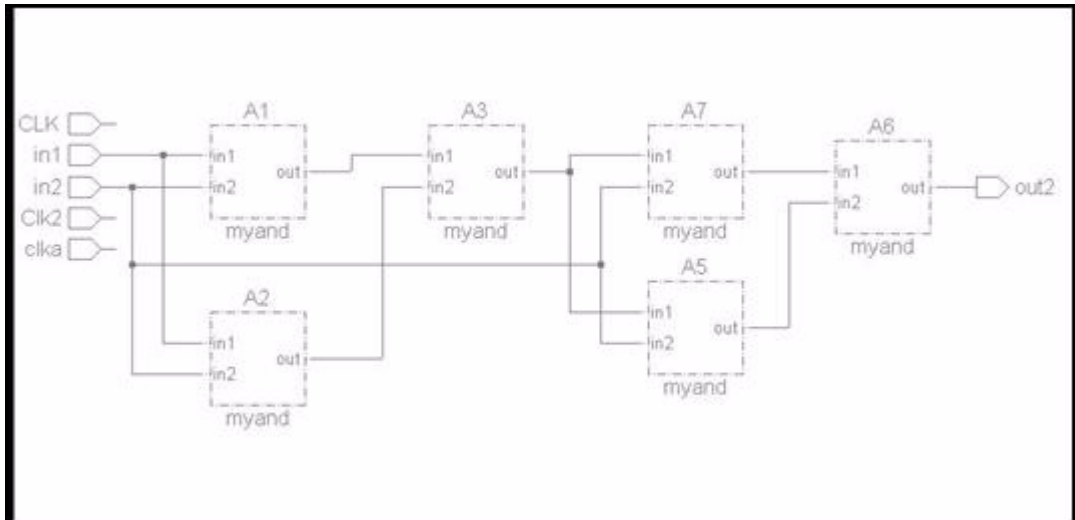
Example 2

 Test Case Files Not Available

The `set_multicycle_path` constraint is specified in the SDC file as follows:

```
set_multicycle_path 3 -through {in1} -through {A3/out} -
through out2
```

A3/out is common in all paths for all paths from in1 to out1. Therefore, A3/out is redundant and the *MCPO3* rule reports a violation.


Default Severity Label

Warning

Rule Group

MCP

Reports and Related Files

None

MCP04

Reports multi-cycle path that covers more than the specified number of paths

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *MCP04* rule reports *set_multicycle_path* constraints that cover more paths than specified number of paths.

By default, the *MCP04* rule reports *set_multicycle_path* constraints that cover more than 20 paths. Use the *num_mcp_path_max* parameter to set a different number.

Rule Exceptions

The *MCP04* rule does not report the total number of paths if the value of the total number of paths is greater than 1.5 times the value specified using the *num_mcp_path_max* rule parameter.

Parameter(s)

- *num_mcp_path_max*: Default value is 20. Set the parameter to any positive integer value to increase or decrease the number of paths specified.

Constraint(s)

- *set_multicycle_path* (Mandatory): Use to modify the single-cycle timing relationship of a constrained path.

Messages and Suggested Fix

Message

The following message appears showing the report generated for each SDC file, where at least one path in *set_false_path/set_multicycle_path* constraint covers paths greater than parameter limit *<mcp_path_limit>*:

[WARNING] All multi-cycle paths excluding paths greater than parameter limit *<mcp_path_limit>* for design *<des-name>* given in file *<file-name>*

Potential Issues

The violation message appears, if your design has constraints that cover more paths than the specified number of paths.

Consequences of Not Fixing

If you do not fix this violation, tool may take huge time because multicycle path would be covering more than the intentional paths.

How to Debug and Fix

- Click on the violation message.
- Corresponding constraint line in the SDC file is highlighted.
- Either comment the constraint or specify each intended `set_multicycle_path` constraint in the SDC file.

Example Code and/or Schematic

Consider the following example:

```
set_multicycle_path -from {5 elements} -to {4 elements}
```

The *MCPO4a* rule would perform the calculation $5*4$ and return 20 paths whereas in the *MCPO4* rule, traversal is done to find the actual number of timing paths that match the given command.

Default Severity Label

Warning

Rule Group

MCP

Reports and Related Files

The *MCPO4* rule also generates a report in a spread sheet (.csv) format with the violation message.

The report is named as

```
<modulename>- mcp_cover_path_info<x>.csv
```

where, <module-name> is the name of the design and <x> is a unique unsigned integer.

The report displays related multi-cycle paths where covering path is less

than the specified parameter limit `<mcpath_limit>`. To view the report, click the violation message.

SDC File Name	SDC File Line No	Multi-cycle Path	Paths Covered
test.sdc	3	from {a} -to {c}	
test1.sdc	5	from {a1} -to {c1}	14

The report contains the following information:

- SDC File Name
- SDC File Line No
- Multi-cycle Path
- Paths Covered: If the reported multi-cycle path is less than 1.5 times of the specified `<mcpath_limit>` parameter, Paths Covered column shows the actual paths. Otherwise, this column is left blank.

MCP04a

Reports a multi-cycle path specification that covers a large number of paths

When to Use

This is more of a caution. The `set_multicycle_path` constraint should not exclude large number of paths as potentially it could have excluded something that is not desired. This rule is applicable to the RTL phase.

Description

The `MCP04a` rule reports `set_multicycle_path` constraints that cover more paths than the specified number of paths.

The MCP4a rule counts the number of paths being covered as the number of `from/to` end point pairs implied by the specification. When you specify `-through` points only, this rule counts the implied `from/to` end points. While counting a `-from` clock or a `-to` clock is considered as one point. After all paths have been counted, this rule compares the total count against the `num_mcp_path_max` parameter.

NOTE: *The `MCP04a` and the `MCP04a` rules both report large number of paths but the way they count the paths is different. Hence, run only one of the rules. The `MCP04` rule counts the actual number of timing paths identified through traversal. The `MCP04a` rule, to save on a lot of traversals, uses some heuristics to perform the count of the timing paths.*

Parameter(s)

- `num_mcp_path_max`: Default is 20. Set the value to the maximum number of paths that can be specified in the `set_multicycle_path` command.

Constraint(s)

- `set_multicycle_path` (Mandatory): Use to modify the single-cycle timing relationship of a constrained path.

Messages and Suggested Fix

The following message appears for a `set_multicycle_path` constraint that excludes `<num>` paths where `<num>` is more than the `<num_mcp_path_max-value>`:

[WARNING] Multicycle path excludes <num> paths which is above the parameter limit of <num_mcpath_max-value>

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

This check is to ensure that you do not use a wildcard irresponsibly as it potentially can exclude a large selection of paths and some of them may be true.

How to Debug and Fix

Validate the number of paths manually using the **Show fan-in/Show fan-out** feature of Schematic Viewer. Since the MCP04a rule does not consider all of the flip-flops sampled by the clock as start/end point of the timing path, the clock specified in the [set_multicycle_path](#) constraint is counted as only one start-point/end-point. The SDC file highlights the constraint.

To resolve this violation, specify the intended objects in the multi-cycle path.

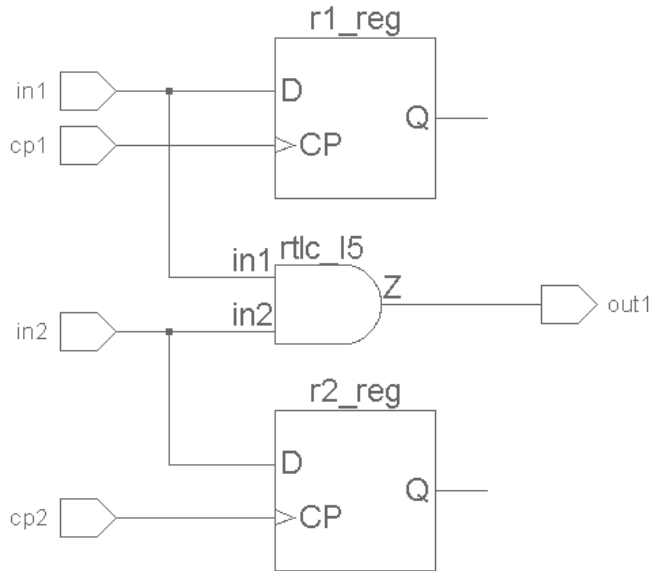
Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:

Timing Exception Rules



The violation message is reported because the number of multicycle paths exceeds the limit specified in the project file. The project file snippet is shown.

```

29 ##### Parameters #####
30 set_parameter num_mcpath_max 0

```

Example 2

Test Case Files Not Available

Consider the following example:

```
set_multicycle_path -from {5 elements} -to {4 elements}
```

The *MCPO4a* rule performs the calculation $5*4$ and returns 20 paths. In the *MCPO4a* rule the traversal is done to find the actual number of timing paths that match the given command. If the *num_mcp_path_max* parameter is set to 10, the *MCPO4a* rule reports a violation because the *num_mcp_path_max* is less than 20.

Default Severity Label

Warning

Rule Group

MCP

Reports and Related Files

None

MCP04b

Reports multi-cycle path specifications that exceed a specific value

When to Use

This is more of a caution. The `set_multicycle_path` constraint should not exceed a user-specified limit. This rule is applicable to the RTL phase.

Description

The *MCP04b* reports a violation if the number of `set_multicycle_path` commands specified in the SDC file exceeds the value set by the `tc_num_mcp_max` parameter.

Parameter(s)

- `tc_num_mcp_max`: Default is 0. Set the value to the maximum number of `set_multicycle_path` commands that can be specified in the SDC file.

Constraint(s)

- `set_multicycle_path` (Mandatory): Use to modify the single-cycle timing relationship of a constrained path.

Messages and Suggested Fix

The following message appears when the number of `set_multicycle_path` commands in the SDC file exceeds the value set by `tc_num_mcp_max`:

```
[WARNING] Number of set_multicycle_path commands <value1> given for design/block <name> is greater than the specified limit of <value2>
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

The runtime of timing analysis tools increases significantly as the number of `set_multicycle_path` constraints increase. This is a methodology rule.

How to Debug and Fix

All the `set_multicycle_path` commands are referenced in the violation message.

Review the SDC file, as highlighted in the Console GUI, and ensure no redundant `set_multicycle_path` commands are specified.

To resolve this violation, update the SDC file or set the `tc_num_mcp_max` parameter to another value.

Example Code and/or Schematic

Suppose, the `tc_num_mcp_max` parameter is set to two. For the following snippet, the MCP04b rule reports a violation the number of `set_multicycle_path` commands is three.

```
...
set_multicycle_path -from in1 -to out1
set_multicycle _path -from in2 -to out2
set_multicycle _path -from in3 -to out3
...
```

Default Severity Label

Warning

Rule Group

MCP

Reports and Related Files

None

MCP05

Reports `set_multicycle_path` setup, hold over or under defined

When to Use

To check setup versus hold relationship for the `set_multicycle_path` constraint. This rule is applicable to all design phases.

Description

The *MCP05* rule reports a violation in the any of the following cases:

1. The `set_multicycle_path` constraint does not occur in pair, once with the `setup` argument and once with the `hold` argument. Use the `tc_setup_hold` parameter to check for missing `setup` argument only or missing `hold` argument only.
2. The value of the `setup` argument of a `set_multicycle_path` constraint is at least one greater than the `hold` argument of the corresponding `set_multicycle_path` constraint. This case is checked when the `strict` parameter is set to `no`.
3. The value of the `setup` argument of a `set_multicycle_path` constraint is exactly one greater than the `hold` argument of the corresponding `set_multicycle_path` constraint. This case checked when the `strict` parameter is set to `yes` or `MCP05`.

In case, neither the `setup` argument nor the `hold` argument has been specified in a `set_multicycle_path` specification, the specified value is assumed to the `setup` value and the `hold` value is assumed to be zero.

Parameter(s)

- `tc_setup_hold`: Default is `Both`. This indicates that the rule checks for both `setup` and `hold` options. Other possible values are `setup` and `hold`.
- `strict`: Default is `no`. Set the value to `yes`, `1`, or a comma separated list of rule names to perform checking in strict mode.

Constraint(s)

SDC

- `set_multicycle_path` (Mandatory): Use to modify the single-cycle timing relationship of a constrained path.

Messages and Suggested Fix

Message 1

The following message appears when the `set_multicycle_path` constraint does not occur in pair, once with the `-setup` argument and once with the `-hold` argument in the specified SDC file(s):

```
[WARNING] Only "<option>" option specified for  
set_multicycle_path
```

Where, `<option>` is the specified option: `-setup` or `-hold`.

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

The Static Timing Analysis tool calculates the hold timing on the incorrect clock edge. This could lead to hold violations in silicon.

How to Debug and Fix

You have not specified either the setup or hold option for the `set_multicycle_path` constraint. Update the SDC file as highlighted in the Console GUI.

Message 2

The following message appears when the value `<value1>` of `-setup` argument is not at least one more than the value `<value2>` of `-hold` argument in the specified SDC file(s):

```
[WARNING] Multicycle path -setup value should be at least one  
greater than -hold value (setup : <value1>, hold : <value2>)
```

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

The Static Timing Analysis tool calculates the hold timing on the incorrect clock edge. This could lead to hold violations in silicon.

How to Debug and Fix

Both [set_multicycle_path](#) constraints are highlighted in the SDC file. The setup value of the [set_multicycle_path](#) constraint should be at least one greater than the hold value. Update the SDC file.

Message 3

The following message appears when the value `<value1>` of `-setup` argument is not one more than the value `<value2>` of `-hold` argument in the specified SDC file(s) and the value of the [strict](#) parameter is set to `yes`:

[WARNING] Multicycle path -setup value should be one greater than -hold value (setup : `<value1>`, hold : `<value2>`)

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

The Static Timing Analysis tool calculates the hold timing on the incorrect clock edge. This could lead to hold violations in silicon.

How to Debug and Fix

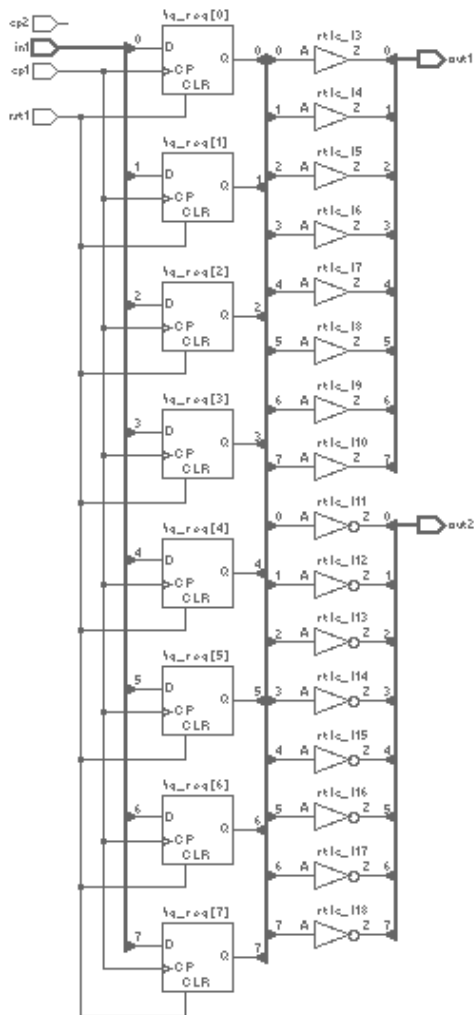
Both [set_multicycle_path](#) constraints are highlighted in the SDC file. As the [strict](#) parameter is set to `yes`, the setup value of the [set_multicycle_path](#) constraint should be exactly one greater than the hold value. Update the SDC file.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The violation message is reported because there is an under defined multicyle path constraint. Double-click the violation to see the details about the violation as shown.

Timing Exception Rules

	B Reason	C Timing-path	D Options	E File name:Line No.
1	under defined	clk1 => clk1 => out1[0]	Setup : 2 ;Hold : 3	top.sdc:2,top.sdc:3

Example 2

Test Case Files Not Available

In this example, the *MCPO5* rule reports a violation because the hold and setup value are equal. The hold value should be greater than the setup value by at least one.

```
set_multicycle_path 2 -setup -hold -from CLK
set_multicycle_path 2 -hold -setup -from CLK
```

Default Severity Label

Warning

Rule Group

MCP

Reports and Related Files

None

MCP08

Reports absence of `-from` or `-to` or both options in `set_multicycle_path`

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

The *MCP08* rule reports when `-from`, `-to` or both the arguments are not specified in `set_multicycle_path` command. Multi-cycle paths should ideally be specified using both the `-from` and `-to` arguments. This reduces the possibility of specifying a single cycle path as multi-cycle path. Single cycle path specified as multi-cycle path leads to sub-optimization since the STA tool gets relaxed for such paths.

Parameter(s)

None

Constraint(s)

- `set_multicycle_path` (Mandatory): Use this constraint to modify the single-cycle timing relationship of a constrained path.

Messages and Suggested Fix

Message

The following message appears if the `-from` or `-to` arguments are not specified in the `set_multicycle_path` command:

```
[INFO] Option(s) <options-list> not specified for  
set_multicycle_path command
```

Potential Issues

The violation message appears because arguments are not specified for `set_multicycle_path` command.

Consequences of Not Fixing

If you do not specify the multi-cycle path completely then the chances of including non multi-cycle paths increases. Hence, there can be timing

issues.

How to Debug and Fix

Specify the options “from” and “to” for the given multi-cycle path command in the SDC file.

This fix is just a suggestion to put `-from` and `-to` arguments and does not impose any restriction.

Example Code and/or Schematic

For the following snippet, the *MCPO8* rule reports a violation because the `-to` argument has not been specified.

```
//test.sdc
set_multicycle_path 1 -from {out1} -through {CLK}
```

Default Severity Label

Info

Rule Group

Multi-Cycle Path Rules

Reports and Related Files

None

MCP09

Reports a violation when options are missing for a multi-cycle path

When to Use

Use this rule in the RTL, Pre-layout and Post-layout phases.

Description

The *MCP09* rule reports when any of the following options are not specified for a multi-cycle path in the *set_multicycle_path* command:

- -setup -rise
- -setup -fall
- -hold -rise
- -hold -fall

Parameter(s)

None

Constraint(s)

- *set_multicycle_path* (Optional): Use this constraint to modify the single-cycle timing relationship of a constrained path.

Messages and Suggested Fix

Message

The following message appears when any of the options *<option-list>* are not specified in the *set_multicycle_path* constraint:

[WARNING] Option(s) *<options-list>* not specified for *set_multicycle_path* *<path-specification>*

Where *<options-list>* is a comma-separated string of missing options and *<path-specification>* is the path list that is a multi-cycle path.

Potential Issues

The violation message appears if *set_multicycle_path* constraint is not specified with arguments.

Consequences of Not Fixing

If you do not fix this violation, some tools infer values which may result in wrong timing analysis.

How to Debug and Fix

Specify all the arguments for [set_multicycle_path](#) constraint.

Example Code and/or Schematic

For the following snippet, the *MCP09* rule reports a violation because not all arguments have been specified.

```
set_multicycle_path 2 -setup -from {A B} -to {C}
set_multicycle_path 1 -hold -from {A} -to {C}
```

Default Severity Label

Warning

Rule Group

MCP

Reports and Related Files

None

MCP_B2BConsis01

Reports multi-cycle paths spanning more than one block

When to Use

This rule is applicable to all the design phases.

Description

The *MCP_B2BConsis01* rule reports a multi-cycle path that spans into more than one block under the immediate hierarchy. This rule checks irrespective of options specified for the multi-cycle command.

Rule Exceptions

The MCP_B2BConsis01 rule:

- Does not check if the multi-cycle path is non-topological.
- Ignores all the other timing exception commands if given for the same path.
- Ignores the path extending from top to another block for top SDC
- Ignores the path extending to the same block
- Ignores the path from top - block - top

Parameter(s)

- None

Constraint(s)

SDC

- *set_multicycle_path* (Mandatory): Use to modify the single-cycle timing relationship of a constrained path.

Messages and Suggested Fix

The following message appears when a multi-cycle path spans across multiple blocks, *<block1>* and *<block2>*, for the current_design *<design-name>* for which the multi-cycle path is specified:

[WARNING] Multicycle path is extending from block *<Block1>* to block *<Block2>* for design *<design-name>*

Potential Issues

The violation message appears because there is a multi-cycle path that spans over multiple blocks. The violation message appears for the following paths:

- Path from block1 - block2
- Path from block1- block2-top
- Path from top-block1-block2-top
- Path extending from top-block1-block2

In addition, this rule ignores path options, such as `setup`, `hold`, and `rise_from`, defined in the multi-cycle path. Therefore, the rule reports this violation message.

Consequences of Not Fixing

Multi-cycle paths extended to more than one block could cause timing closure problems because

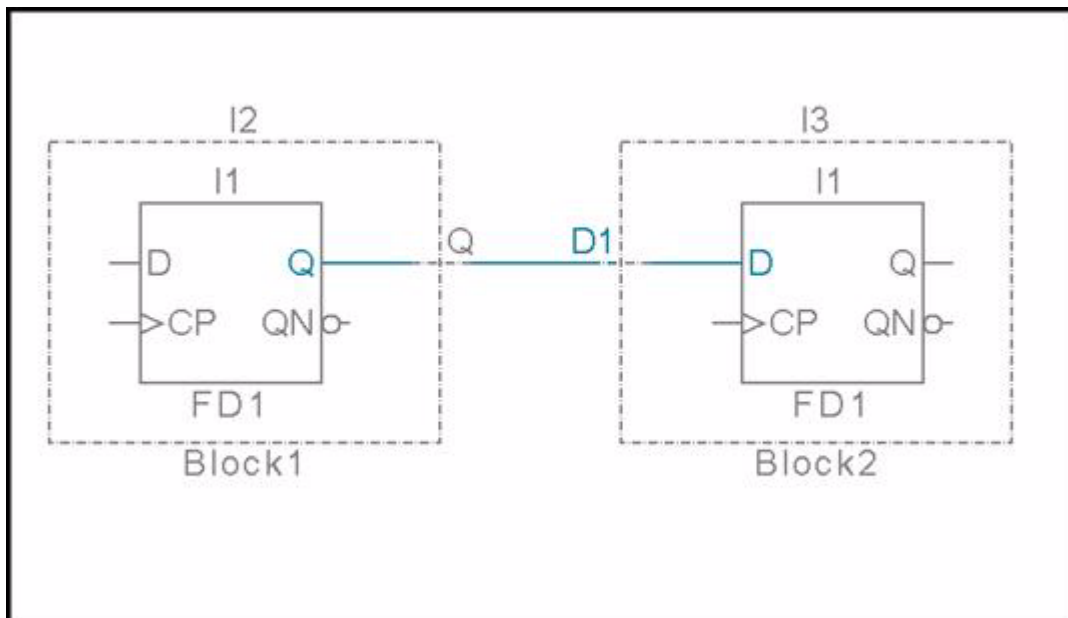
- The path tends to be a very long path.
- The path is related to multiple block designers, which may lead to error.
- The path is related to timing budgeting, which may lead to an error.

Manually identifying and analyzing these kind of paths is a daunting task.

How to Debug and Fix

The `MCP_B2BConsis01` rule highlights the SDC constraint in file and a violating path between blocks. Click the **Incremental Schematic** button to view the schematic. As shown in the image below, this rule highlights the path as follows:

1. Path from start terminal to block's output port.
2. Path from block's output port to input port of another block.
3. Input and output pins of feed through block. Path continues Step 2 & Step 3 for feed through blocks.
4. Path from input port to the path terminating terminal.



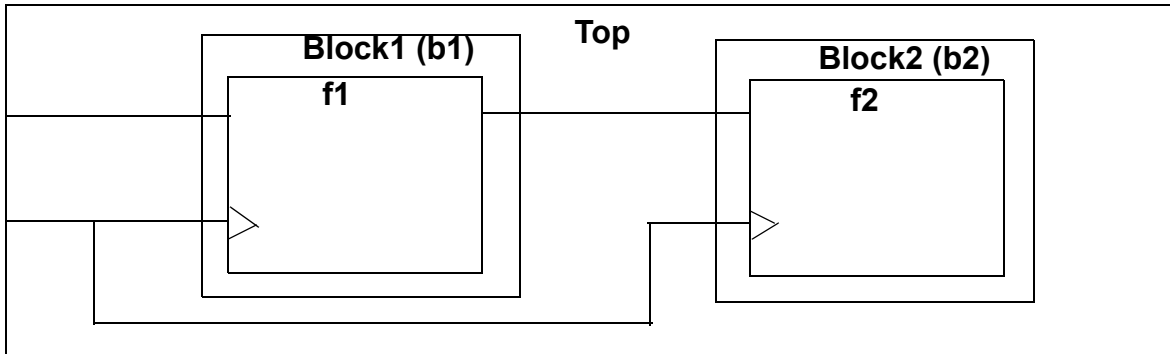
To resolve this violation, you can either re-design all such paths so that they are in a single block or make conditions that the paths are no longer an exception.

Example Code and/or Schematic

Example 1

In this example, the *MCP_B2BConsis01* rule reports a violation because for top, the multi-cycle path is extending from Block1 to Block2. The `set_multicycle_path` for top is specified as:

```
set_multicycle_path 3 -from B1/f1/CP -to B2/f2/D
```

This rule ignores all unconnected multi-cycle paths and paths that are extending from top to block and vice-versa.

Example 2

Suppose for top, the `set_multicycle_path` constraint is specified as:

```
set_multicycle_path 3 -from B1/f1/CP -to f2/D
```

In this example, the `MCP_B2BConsis01` rule does not report a violation because for top, the multi-cycle path is extending from Block1 to top.

Default Severity Label

Warning

Rule Group

MCP

Reports and Related Files

None

Other Timing Exception Rules

The Other Timing Exception Rules Sub-group `TE_Others` contains the following rules:

Rule	Description
<i>Disable_Timing01</i>	Generates schematic view of paths disabled using the <i>set_disable_timing</i> constraints
<i>Disable_Timing02</i>	Generates schematic view of clock paths disabled using the <i>set_disable_timing</i> constraints
<i>TE_Conflict01</i>	Overlapping timing exception commands
<i>TE_Consis01</i>	Unconnected <code>set_max_delay</code> reference points
<i>TE_Consis02</i>	Unconnected <code>set_min_delay</code> reference points
<i>TE_Methodology02</i>	Multiple use of * wildcard character in <i>set_false_path</i> and <i>set_multicycle_path</i> commands

Disable_Timing01

Reports objects specified in `set_disable_timing`

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *Disable_Timing01* rule reports objects specified in the [set_disable_timing](#) constraint and generates the corresponding schematic view.

Parameter(s)

None

Constraint(s)

- [set_disable_timing](#) (Mandatory): Use this constraint to specify the list of arcs to be disabled.

Messages and Suggested Fix

For each [set_disable_timing](#) constraint, the rule generates the following message:

[WARNING] All timing arcs leading to/from the specified objects are disabled

Potential Issues

The violation message appears when the rule highlights the schematic, to validate the timing path which gets disabled corresponding to a constraint. This is a methodology rule.

Consequences of Not Fixing

The arc disabled by the [set_disable_timing](#) constraint can disable a timing path which has to be analyzed, so the impact of [set_disable_timing](#) constraint should be considered while specifying it.

How to Debug and Fix

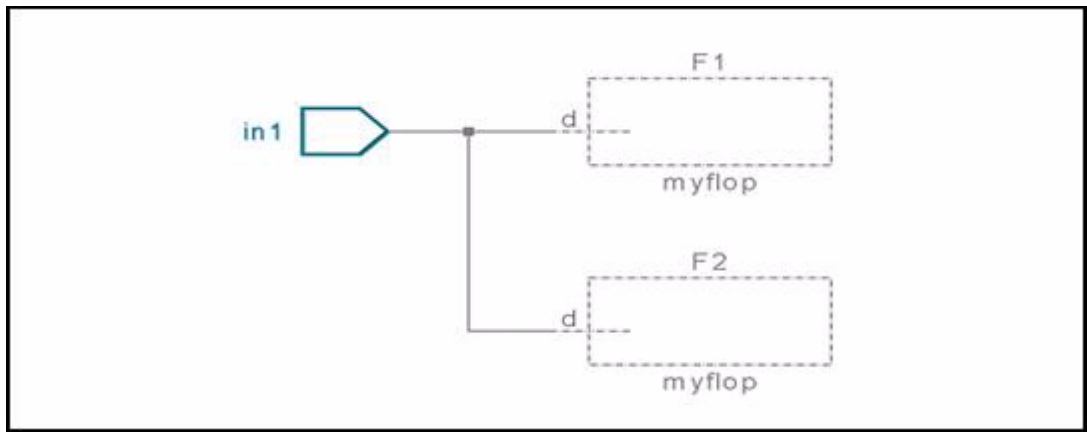
The objects specified in the [set_disable_timing](#) constraint are highlighted in the GUI. If you do not want to disable the timing path that is getting

disabled by the corresponding [set_disable_timing](#) constraint, that constraint should be removed from the SDC file.

Example Code and/or Schematic

In the following figure, F1 and F2 are reported because the [set_disable_timing](#) constraint is specified in the following code:

```
set_disable_timing in1
```



Default Severity Label

Warning

Rule Group

Disable_Timing

Reports and Related Files

- [tc_report_disable_timing Report](#): Lists all objects specified in the [set_disable_timing](#) constraint.

Disable_Timing02

Identifies clock paths that are blocked by `set_disable_timing`

When to Use

To ensure unbridled clock network and full clock coverage. It is applicable in all stages of the design cycle, including RTL, Pre-layout and Post-layout phases.

Description

The *Disable_Timing02* rule checks the clock paths starting from clock sources, which are specified using the *create_clock/create_generated_clock* commands, and reports the paths that are disabled using the *set_disable_timing* constraint.

Parameter(s)

None

Constraint(s)

SDC

- *create_clock*: Use to create a clock
- *create_generated_clock*: Use to create a clock
- *set_clock_sense*: Use to specify pins/timing-arcs to stop the clock/data propagation for specified clocks.
- *set_disable_timing*: Use to disable timing-arcs in the design.

Messages and Suggested Fix

The following message appears when a clock path for clock `<clk-name>` is disabled:

[WARNING] Clock path "<clk-name>" is disabled by `set_disable_timing`

Potential Issues

Disabling a timing-arc in the clock path may leave some flip-flops unconstrained, which could result in a faulty Static Timing Analysis (STA),

and may also reduce timing-coverage.

Consequences of Not Fixing

Blocked clock paths would be excluded from STA analysis leading to potential timing failure in silicon.

How to Debug and Fix

View the incremental schematic of the violation message.

The violation message reports clock paths blocked by the [set_disable_timing](#) constraints. The schematic highlights the clock path till the point of blockage. The SDC file highlights the clock and the disable timing constraint that is blocking the path.

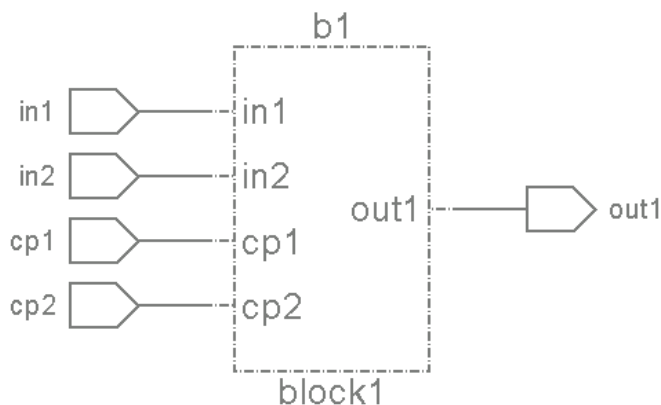
To fix the issue, ascertain the rationale of having a disable-timing command in a clock path. Accordingly, either remove it, or use case-analysis commands if the intention is to have mode-based clocks.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



Timing Exception Rules

The top.sdc is as follows:

```
top.sdc | top.sdc (2) |
create_clock -name clk1 -period 10 -waveform { 0 5} cp1
create_clock -name clk2 -period 10 -waveform { 0 5} cp2
set_disable_timing -from CP -to Q b1/f2
```

The violation message appears because the clock path `clk1` is disabled by the `set_disable_timing` constraint. In the `top.sdc` file, the corresponding constraint is highlighted.

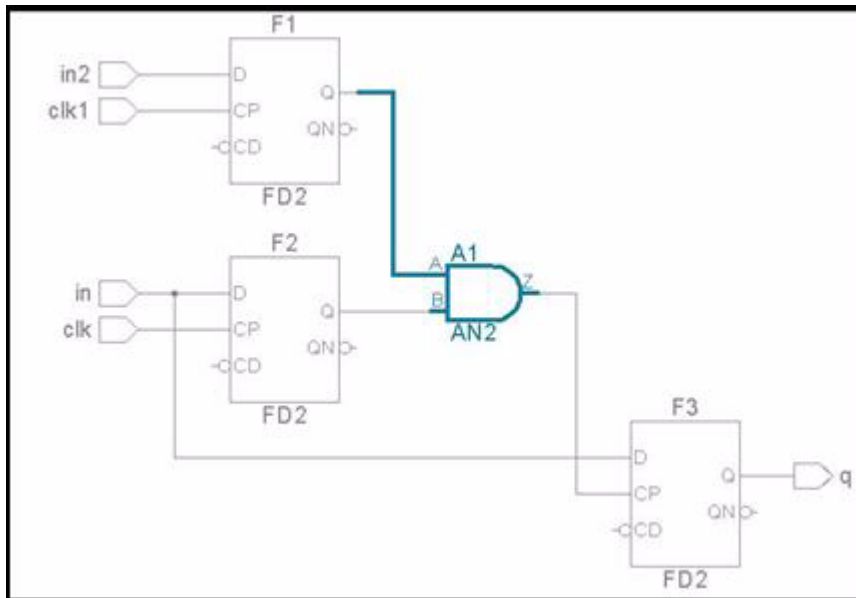
Example 2

Test Case Files Not Available

In the following commands, the `Disable_Timing02` rule reports a violation because a disable-timing at `A1/Z`, `F3` is left unconstrained. This causes the timing-path from `F3` to `Q` be left out of the timing-coverage.

```
create_clock -name CLK -period 10 -waveform { 0 5} clk
create_clock -name CLK1 -period 10 -waveform { 0 5} clk1
create_generated_clock -name gclk1 -source clk1 -divide_by 1
F1/Q
create_generated_clock -name gclk2 -source clk1 -divide_by 1
F2/Q
set_disable_timing A1/Z
```

The schematic illustration is as follows.



Default Severity Label

Warning

Rule Group

TE_Others

Reports and Related Files

None

TE_Conflict01

Identifies overlaps between timing exceptions commands

When to Use

To decrease redundancy and unintentional overlapping in the specifications of the timing exceptions specifications in the SDC file. It is applicable in all stages of the design cycle, including RTL, Pre-layout and Post-layout phases.

Description

The *TE_Conflict01* rule compares and reports the following overlapping timing exception commands:

- *set_clock_groups* with *set_clock_groups*
- *set_false_path* with *set_false_path*
- *set_multicycle_path* with *set_multicycle_path*
- *set_max_delay* with *set_max_delay*
- *set_min_delay* with *set_min_delay*

Refer to the [Parameter\(s\)](#) section to control the behavior of this rule.

Rule Exceptions

The *TE_Conflict01* rule does not flag overwritten timing exception commands. Such commands are reported by the [Const_Struct04a](#) and [Const_Struct05](#) rules.

Parameter(s)

- *schematic_opt*: Default is `yes`. Set the value to `no` to highlight the overlapping paths in Schematic Viewer.
- *tc_compare_cmd_file*: Default is `none` and the *TE_Conflict01* rule checks for the overlapping timing exception commands as specified in the rule description. You can specify a file that contains the timing exception command name pairs (one pair in each line) that restricts the *TE_Conflict01* rule to check the specified pairs.
- *tc_te_conflict_fast_run*: Default is `no` and the *TE_Conflict01* rule generates a schematic. Set this parameter to `yes` to not generate a schematic. This results

in a faster rule runtime. If the *schematic_opt* parameter is set to no, this parameter does not have any effect.

Constraint(s)

SDC

- *set_clock_groups*: Use to specify a list of clocks that is used to establish exclusive or asynchronous relationship with clocks of another group.
- *set_false_path*: Use to specify timing paths that are not considered in timing-analysis. These timing paths are marked as false.
- *set_max_delay/set_min_delay*: Use to specify maximum and minimum delay for timing paths.
- *set_multicycle_path*: Use to identify multi-cycle timing paths.

Messages and Suggested Fix

The following message appears for the timing exception commands *<command-name1>* and *<command-name2>* in the SDC file *<file1-name>* at the line *<num1>* when it overlaps with another timing exception constraint in the SDC file *<file2-name>* at the line *<num2>*:

```
[INFO] Overlap found between <command-name1> at file <file1-name>, line:<num1> and <command-name2> at file <file2-name>, line <num2>
```

Where, *<command-name1>* and *<command-name2>* can be any of the following: *set_clock_groups*, *set_false_path*, *set_multicycle_path*, or *set_max_delay/set_min_delay*.

Potential Issues

The violation is reported because an overlap:

1. Introduces redundancy to the SDC file.
2. May hide the original intentions of the designer. For example, if a timing path is being covered unintentionally by a false-path command and correctly by a multi-cycle path command. Then, since the false path is of higher precedence, the timing path would be excluded from the Static Timing Analysis (STA), instead of being considered as a multi-cycle path.

Consequences of Not Fixing

The consequences of not fixing this violation are:

1. Overlapping timing exceptions can result in longer runtime of implementation tools.
2. If designer's intentions get hidden by overlapping of timing-paths between two different commands, STA of some paths could be incorrect or excluded from timing-coverage. This results in chip failure.

How to Debug

If the *schematic_opt* parameter is set to `yes`, no timing path is highlighted. Determine the overlapping timing path by using **Show input cone** and **Show output cone** feature of the Schematic Viewer on the design objects specified in the timing exceptions constraints.

In some cases a quick reference to the overlapping exceptions can give a clear indication where the overlapping is occurring. In case *schematic_opt* is set to `no`, examine the schematic to identify the overlapping path.

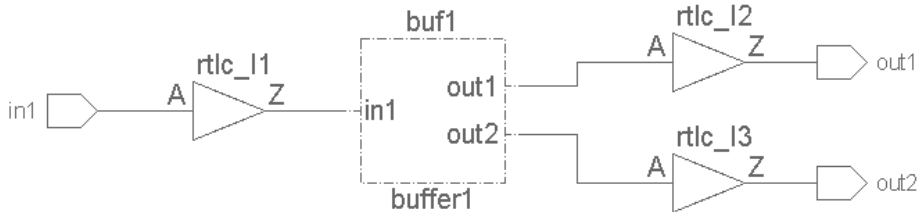
To fix the issue, exclude the overlapping path from either of the exception where it is unintentional. Since it is highly time and data intensive to show/report all overlapping timing-paths between two overlapping exception commands, a single path is reported. Refer to the overlapping exceptions in the SDC file to determine the overlapping pattern between these commands.

Example Code and/or Schematic

Example 1

[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



The top.sdc file snippet is as follows:

```
top.sdc | top.sdc (2)
set_max_delay -from [get_ports in1] -through [get_pins buf1/in1] 20
set_max_delay -from [get_ports in1] -to [get_ports out1] 10

set_min_delay -from [get_ports in1] -through [get_pins buf1/in1] 20
set_min_delay -from [get_ports in1] -to [get_ports out1] 10
```

The violation message appears because there is an overlap between the [set_max_delay](#) constraints defined in the top.sdc file.

Example 2

Test Case Files Not Available

In the following example, the *TE_Conflict01* rule reports a violation because [set_false_path](#) specifies all paths starting from in1 and overlaps the [set_max_delay](#) setting from in1 to in2:

```
set_max_delay 2 -from in1 -to in2
set_false_path -from in1
```

Example 3

Test Case Files Not Available

Timing Exception Rules

In the following example, if you intend to mark the timing-path from `in1` to `in2` as a multi-cycle path, exclude that path from the following false path exception. Otherwise, the timing path from `in1` to `in2` is not considered for Static Timing Analysis and can cause timing-issues.

```
set_multicycle_path 2 -from in1
set_false_path -to in2
```

The rule reports a violation because the timing path `in1` to `in2` is covered by both exceptions.

Example 4

Test Case Files Not Available

In the following example, if you intend to mark the timing-path from `in1` to `in2` as a multi-cycle path, exclude that path from the following [set_clock_groups](#) exception. Otherwise, the timing path from `in1` to `in2` is not considered for Static Timing Analysis and can cause timing-issues

```
set_multicycle_path 2 -from in1 -to in2
set_input_delay 1 -clock Clk1 in1
set_output_delay 1.5 -clock Clk2 in2
set_clock_groups -asynchronous -group Clk1 -group Clk2
```

The rule reports a violation because the timing path `in1` to `in2` is covered by both exceptions, [set_multicycle_path](#) and [set_clock_groups](#).

Default Severity Label

Info

Rule Group

TE_Others

Reports and Related Files

None

TE_Consis01

Reports `set_max_delay` reference points that are not connected

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The *TE_Consis01* rule checks the connectivity of paths specified by the `set_max_delay` constraint. This rule checks the partial and full connectivity of specified path. This rule reports the unconnected nodes specified in path.

A `set_max_delay` constraint is said to be partially unconnected if at least one path from a node specified in the `-from` list is connected to a node specified in the `-to` list.

A constraint is said to be fully unconnected if no path exists for nodes specified in the `-from` list to nodes specified in the `-to` list.

Parameter(s)

- *strict*: Default is `no`. This indicates that this rule reports unconnected points in a specified exception. When this parameter is set to `yes` or `TE_CONSIS01`, this rule reports a violation only if all the specified points are unconnected. If at least one point is connected, this rule does not report any violation.
- *tc_report_unconnected_points*: Default is 5. This indicates that a maximum of 5 unconnected points are reported for exceptions specified in the SDC file. You can set this parameter to an integer between 0 and 1000 to increase or decrease this limit. For example, if you set this parameter to 20, up to 20 unconnected points are reported in the generated CSV file.

Constraint(s)

- `set_max_delay` (Mandatory): Use this constraint to specify maximum delay for timing paths.

Messages and Suggested Fix

The following message appears for a [set_max_delay](#) constraint after the CSV is generated:

[WARNING] CSV file <file-name> generated to report unconnected points for design/block <name> for set_max_delay

Potential Issues

The violation message appears when no path has been specified for the [set_max_delay](#) constraint.

Consequences of Not Fixing

Unconnected [set_max_delay](#) reference points result in longer runtime of timing analysis tools.

How to Debug and Fix

Double-click the violation message to open the CSV file in Spreadsheet Viewer. This file contains the list of unconnected reference points. A sample CSV file is as follows.

	B	C	D	E
	All points unconnected	Type	Unconnected points	File Name:Line No.
1	Yes		Points that are not connected with any specified points in set_max_delay	blockA-rtl.or1.sdc:24
• 2		-from	Ain1	
• 3		-to	Aout	

You can also view the incremental schematic. The rule highlights the nodes in the schematic. These nodes are not forming any path. Check whether this timing exception should be kept.

Example Code and/or Schematic

Consider the following snippet:

```
set_max_delay -from {A B} -to {X Y}
```

Here, the *TE_Consis01* rule expects the following paths to exist:

- Path starting from A and ending at either X or Y
- Path starting from B and ending at either X or Y
- Path starting from either A or B and ending at X
- Path starting from either A or B and ending at Y

If only paths A->X and A->Y exist, then the *TE_Consis01* rule reports for B. Similarly, if only paths A->X and B->X exist, then the *TE_Consis01* rule reports for Y. If only paths A->X and B->Y exist, then the *TE_Consis01* rule does not report as all required paths exist.

Similarly, the *TE_Consis01* rule check for paths when the `-through` argument is also specified.

Default Severity Label

Warning

Rule Group

TE_Others

Reports and Related Files

- *TE_Consis01 Report*: Lists the unconnected points specified in `-from/-through/-to` arguments of *set_max_delay* constraints.

TE_Consis02

Reports unconnected `set_min_delay` reference points

When to Use

This rule is applicable to the RTL, Pre-layout and Post-layout phases.

Description

The `TE_Consis02` rule checks the connectivity of paths specified by the `set_min_delay` constraint. This rule checks the partial and full connectivity of specified path. This rule reports the unconnected nodes specified in the path.

A `set_min_delay` constraint is partially unconnected if at least one path from a node specified in the `-from` list is connected to node specified in the `-to` list.

A constraint is fully unconnected if no path exists for nodes specified in the `-from` list to nodes specified in the `-to` list.

Parameter(s)

- `strict`: Default is `no`. This indicates that this rule reports unconnected points in a specified exception. When this parameter is set to `yes` or `TE_CONSIS02`, this rule reports a violation only if all the specified points are unconnected. If at least one point is connected, this rule does not report any violation.
- `tc_report_unconnected_points`: Default is 5. This indicates that a maximum of 5 unconnected points are reported for exceptions specified in the SDC file. You can set this parameter to an integer between 0 and 1000 to increase or decrease this limit. For example, if you set this parameter to 20, up to 20 unconnected points are reported in the generated CSV file.

Constraint(s)

- `set_min_delay` (Mandatory): Use this constraint to specify a minimum delay for timing paths.

Messages and Suggested Fix

The following message appears for a `set_min_delay` constraint after the CSV is generated:

[WARNING] CSV file <file-name> generated to report unconnected points for design/block <name> for `set_min_delay`

Potential Issues

The violation message appears when no path is connected for specified `set_min_delay` constraint.

Consequences of Not Fixing

Unconnected `set_min_delay` reference points result in longer runtime of timing analysis tools.

How to Debug and Fix

Double-click the violation message to open the CSV file in Spreadsheet Viewer. This file contains the list of unconnected reference points. A sample CSV file is as follows.

	B	C	D	E
	All points unconnected	Type	Unconnected points	File Name:Line No.
1	Yes		Points that are not connected with any specified points in <code>set_min_delay</code>	blockA-rtl.or1.sdc:23
2		-from	Ain1	
3		-to	Aout	

The rule highlights the nodes in schematic. The nodes are not forming any path. Check such timing exceptions should be kept.

Example Code and/or Schematic

Consider the following snippet:

```
set_min_delay -from {A B} -to {X Y}
```

Here, the `TE_Consis02` rule expects the following paths to exist:

- Path starting from A and ending at either X or Y
- Path starting from B and ending at either X or Y
- Path starting from either A or B and ending at X

- Path starting from either A or B and ending at Y

If only paths A->X and A->Y exist, then the *TE_Consis02* rule reports for B. Similarly, if only paths A->X and B->X exist, then the *TE_Consis02* rule reports for Y. If only paths A->X and B->Y exist, then the *TE_Consis02* rule does not report as all required paths exist.

Similarly, the *TE_Consis02* rule check for paths when the `-through` argument is also specified.

Default Severity Label

Warning

Rule Group

TE_Others

Reports and Related Files

- [TE_Consis02 Report](#): Lists the unconnected points specified in `-from/-through/-to` fields of `set_min_delay` constraints.

TE_Methodology02

Reports timing exceptions in which wildcard nodes refer to multiple reference points

When to Use

Use this rule at the RTL phase of the design.

Description

The *TE_Methodology02* rule checks the *set_false_path* and *set_multicycle_path* constraints that have more than one instance of the wildcard character ***, while specifying timing exceptions.

Rule Exceptions

This rule does not check the *set_false_path* and *set_multicycle_path* constraints if one or more built-in *SDC_** rules have already been reported.

Parameter(s)

None

Constraint(s)

- *set_false_path* (Optional): Use this constraint to identify the paths that you do not want to include in timing analysis of the design.
- *set_multicycle_path* (Optional): Use this constraint to define a multi-cycle path.

Messages and Suggested Fix

The following message appears for the constraint *<constr>* that has multiple wildcard characters (***):

[INFO] In command: *<constr>* wild character *** is used more than once to specify timing exception

Where,

<constr> is either the *set_false_path* or the *set_multicycle_path* constraint.

Potential Issues

This is an informational message that appears for a constraint that has

multiple wildcard characters.

Consequences of Not Fixing

High usage of the wildcard character * may require you to specify more than the intended paths.

How to Debug and Fix

Check whether it is intentional that the wildcard object is referring to multiple instances. If it is unintentional, specify the hierarchical name. Otherwise, you can waive by using the SpyGlass utility.

Example Code and/or Schematic

Consider the following example in which the *TE_Methodology02* rule reports the use of the wildcard character *:

```
set_multicycle_path -from abc/efg[*] -to abc/adc[*]/adf
```

This rule does not report cases in which the wildcard character * is used to specify the signal name itself, as shown in the following examples:

```
set_multicycle_path -from ab*/efg*/xyz
```

```
set_multicycle_path -from abc/efg* -to abc/adc*/adf
```

Default Severity Label

Info

Rule Group

TE_Others

Reports and Related Files

None

Abstract View Rules

For information on using the SoC abstraction flow, refer to the *SpyGlass SoC Methodology Guide*. The SpyGlass Constraints solution provides the following rules:

Rule	Description
<i>CONS_abstract01</i>	Generates an abstract view of a block for doing system level Timing verification
<i>CONS_validate01</i>	Validates the timing constraints of top-level block with the constraints of abstracted block

CONS_abstract01

Generates an abstract view of a block for performing system level timing verification

When to Use

Use this rule for RTL and netlists.

The abstracted view can be used in verifying the timing at system level. For information on using the SoC abstraction flow, refer to the *SpyGlass SoC Methodology Guide*.

Description

The *CONS_abstract01* rule generates an abstract view of block(s). The abstraction of the block is performed in correlation with the SDC constraints specified through *sdc_data* constraints. All such *sdc_data* constraints are processed and corresponding *abstract_port* constraints generated by the rule. The *mode* field of the *abstract_port* constraint specifies the corresponding *sdc_data* constraint. These constraints are output into sgdc file. The sgdc file path is highlighted in the message given by the rule.

Prerequisites

The design and SDC constraints should be checked through the SpyGlass Constraints platform for validation.

Parameter(s)

None

Constraint(s)

- *abstract_port* (optional): Used to abstract the block characteristics.

Messages and Suggested Fix

The following message appears after an abstract port is generated for a design/block <name>:

[INFO] Abstract Port has been generated for design/block <name>

Potential Issues

This is a methodology rule. The limitations of adopting this hierarchical methodology are captured in the *SpyGlass SoC Methodology Guide*.

Consequences of Not Fixing

Not applicable

How to Debug and Fix

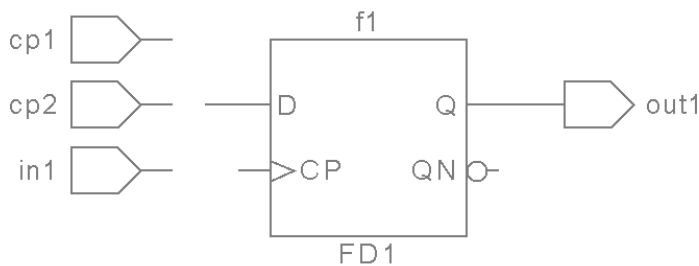
For information on using the SoC abstraction flow, refer to the *SpyGlass SoC Methodology Guide*.

Example Code and/or Schematic

Example 1

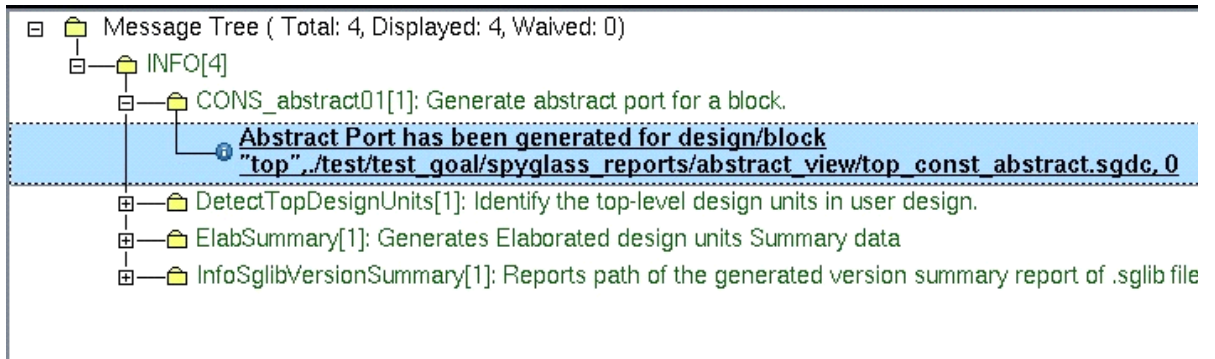
[View Test Case Files](#)

This example shows when this rule reports a violation message. The schematic of the top module is as follows:



This violation appears to indicate the abstract view for the design has been generated. You can open the `top_const_abstract.sgdc` file by double-clicking the message.

Abstract View Rules

**Default Severity Label**

Info

Rule Group

None

Reports and Related Files

The SGDC file is provided for each block. The file is highlighted in the message.

CONS_validate01

Validates the timing constraints of top level block with the constraints of abstracted block

When to Use

Use this rule for RTL and netlists.

For information on using the SoC abstraction flow, refer to the *SpyGlass SoC Methodology Guide*.

Description

The *CONS_validate01* rule validates the timing constraints of top-level block with the constraints of the abstracted block. It performs the validation of top versus block for the following constraints

1. *create_clock* on block boundary
2. *set_case_analysis* on block boundary
3. *set_input_delay*/*set_output_delay*

Rule Exceptions

This rule does not report a violation message for a block that is not being abstracted.

Parameter(s)

None

Constraint(s)

- *abstract_port* (optional): Used to abstract the block characteristics.

Messages and Suggested Fix

Message 1

The following message appears when the *set_case_analysis* differs:

[ERROR] set_case_analysis are not equivalent as their effect differ at object '<port-name>'. Value defined in block <block-name>: <block-value>, Value from top <top-name>: <top-value>

For debugging information, refer to *How to Debug and Fix*.

Message 2

The following message appears when the clocks are applied only at top or block levels:

[WARNING] Clocks are applied on port <port-name> only at <top/block> <name>

For debugging information, refer to [How to Debug and Fix](#).

Message 3

The following message appears when [set_input_delay/set_output_delay](#) is defined at only the top or block level:

[WARNING] <delay-type> is only defined at <top/block> <name>

For debugging information, refer to [How to Debug and Fix](#).

Message 4

The following message appears when [set_input_delay/set_output_delay](#) is defined with different options for the top and block level:

[WARNING] <delay-type> defined with different option at block <b-name> and top <top-name>

For debugging information, refer to [How to Debug and Fix](#).

Message 5

The following message appears when [set_input_delay/set_output_delay](#) is defined with different values for the top and block level:

[WARNING] <delay-type> delay value is different at block <b-name> and top <top-name>

For debugging information, refer to [How to Debug and Fix](#).

Message 6

The following message appears when there is a clock mismatch at the block and top levels:

[WARNING] No clock at Block <b-name> corresponding to the clock <clk-name> (period: <value1>, waveform: <value2> at SoC <top-name>

For debugging information, refer to [How to Debug and Fix](#).

Message 7

The following message appears when no SDC file is specified for the

abstracted block. This message also appears when the top has an abstracted model, but no SDC constraint file is specified for the top.

[WARNING] No sdc constraint file is specified for <top/block> <name>

For debugging information, refer to [How to Debug and Fix](#).

Message 8

The following message appears when the clock at the SoC is slower than the clock at the block level.

[INFO] Clock "<top-clock-name>(period: <value1>/waveform: <value2>)" at SoC <top-name> is slower than Clock "<block-clock-name>(period: <value1>/waveform: <value2>)" at Block <b-name>

For debugging information, refer to [How to Debug and Fix](#).

Message 9

The following message appears when the clock at the SoC is faster than the clock at the block level.

[WARNING] Clock "<top-clock-name>(period: <value1>/waveform: <value2>)" at SoC <top-name> is faster than Clock "<block-clock-name>(period: <value1>/waveform: <value2>)" at Block <b-name>

For debugging information, refer to [How to Debug and Fix](#).

Message 10

The following message appears when there is a port (s) in the block for which no abstract port is specified in the abstracted block's SGDC file.

[WARNING] No abstract block port for <port-name>

Potential Issues

Messages 1 to 9: It indicates a mismatch in the timing characteristics of top and block level. This can potentially cause serious timing issues for the SoC.

Message 10: The violation message explicitly states the potential issues.

Consequences of Not Fixing

Messages 1 to 9: There exists serious timing issues in the design due to which timing closure of the design shall not be met.

Message 10: There is an inconsistency in the design or constraints specifications.

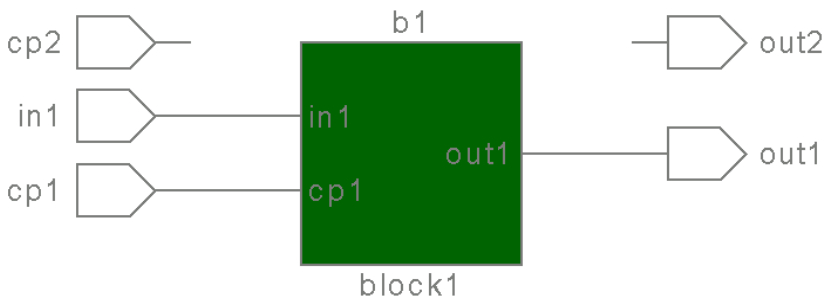
How to Debug and Fix

Messages 1 to 9: The violating constraints are highlighted in the HDL window. Update the constraints to eliminate the timing issues.

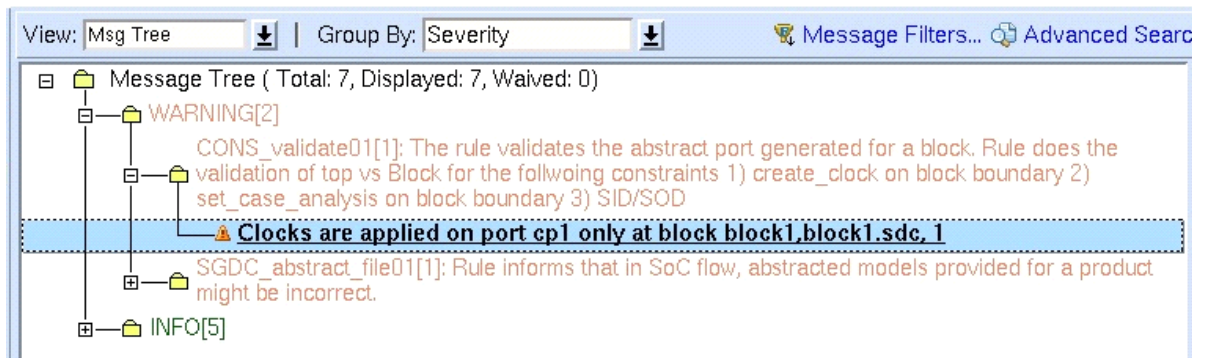
Message 10: Run the SpyGlass with correct constraints specifications at block level before generating the abstracted model.

Example Code and/or Schematic**Example 1**[View Test Case Files](#)

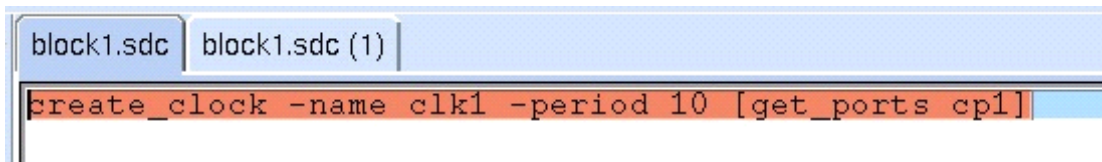
This example shows when this rule reports [Message 2](#). The schematic of the top module is as follows:



This violation appears because there is a violation related to timing constraint of the top-level block with the constraint of the abstracted block.



In the `block1.sdc` file, the corresponding constraint is highlighted.



Example 2

Test Case Files Not Available

This example illustrates the condition under which [Message 1](#) is reported. Consider the following design, SGDC, and SDC files.

```
--design file begin--
module top( in, clk, out);
input in;
input clk;
output out;
block b(.in(in), .clk(clk), .out(out));
endmodule
```

```

module block( in, clk, out);
input in, clk;
output out;
AN2 a1(.A(in), .B(clk), .Z(out));
Endmodule
--design file begin--

--SGDC file begin--
current_design top
sdc_data -file top.sdc -mode reference
sgdc -import block block_abstract.sgdc
block -name block

current_design block
sdc_data -file block.sdc -mode reference
--SGDC file end-

--top.sdc begin-
set_case_analysis 0 in
--top.sdc end-

--block.sdc begin--
--block.sdc end--

```

Since there shall be inconsistency in case analysis value at the in port of block 'block'. The rule reports the following violation:

[ERROR] set_case_analysis are not equivalent as their effect differ at object 'in'. Value defined in block: X, Value from

Top: 0

Example 3Test Case Files Not Available

This example illustrates the condition under which [Message 6](#), [Message 8](#), and [Message 9](#), are reported. Consider the following top and block SDCs.

```
--top.sdc begin-
create_clock -name clk1 -period 5 [get_ports clk1]
create_clock -name clk2 -period 10 -waveform {0 2} [get_ports
clk1] -add
create_clock -name clk3 -period 20 [get_ports clk1] -add
create_clock -name clk4 -period 25 [get_ports clk1] -add
create_clock -name clk5 -period 25 [get_ports clk1] -add
create_clock -name clk6 -period 30 [get_ports clk1] -add
create_clock -name clk7 -period 40 [get_ports clk1] -add
create_clock -name clk8 -period 50 [get_ports clk1] -add
create_clock -name clk9 -period 50 -waveform {0 8} [get_ports
clk1] -add
--top.sdc end-

--block.sdc begin-
create_clock -name bclk1 -period 10 [get_ports clk]
create_clock -name bclk2 -period 10 -waveform {0 5}
[get_ports clk] -add
create_clock -name bclk3 -period 20 [get_ports clk] -add
create_clock -name bclk4 -period 30 [get_ports clk] -add
create_clock -name bclk5 -period 40 [get_ports clk] -add
create_clock -name bclk6 -period 50 [get_ports clk] -add
create_clock -name bclk7 -period 60 [get_ports clk] -add
```



```
--block.sdc end-
```

In this example, the following clocks have the same clock characteristics. Since the clock characteristics are the same, these clocks are validated and therefore not reported.

```
bclk3(period: 20) ==> clk3(period: 20)
```

```
bclk4(period: 30) ==> clk6(period: 30)
```

```
bclk5(period: 40) ==> clk7(period: 40)
```

```
bclk6(period: 50) ==> clk8(period: 50)
```

The following are the clock pairs where period is the same, but the waveform is different. In addition, Clk9 and Bclk7 have different periods.

```
bclk1(waveform: 0 5) ==> clk2(waveform: 0 2)
```

```
bclk7(period: 60) ==> clk9(period: 50)
```

The top clock clk2 is faster than block clock bclk1. Therefore, the following message is reported.

```
[WARNING] Clock "clk2( waveform: 0 2)" at SoC "top" is faster than Clock "bclk1( waveform: 0 5)" at Block "block1"
```

Similarly, the following message would be reported.

```
[WARNING] Clock "clk9(period: 50)" at SoC "top" is faster than Clock "bclk7(peri od: 60)" at Block "block1"
```

Similarly, the following clock pairs have a slower SoC clock than the block clock:

```
bclk2(period: 10) ==> clk4(period: 25)
```

Therefore, the following message is reported.

```
[INFO] Clock "clk4(period: 25)" at SoC "top" is slower than Clock "bclk2(peri od: 10)" at Block "block1"
```

Lastly, the following are the SoC clocks that have no corresponding block clocks:

```
clk1(period: 5, waveform: 0 2.5)
```

```
clk5(period: 25, waveform: 0 12.5)
```

Therefore, the following messages are reported.

[WARNING] No clock at Block "block1" corresponding to the clock "clk1(period: 5, waveform: 0 2.5)" at SoC "top"

[WARNING] No clock at Block "block1" corresponding to the clock "clk5(period: 25, waveform: 0 12.5)" at SoC "top"

Default Severity Label

Error

Rule Group

CONS_Validate

Reports and Related Files

No related reports or files

Merge SDC Files

For merging SDC files, the SpyGlass Constraints solution provides the following rule:

Rule	Description
<i>SDC_ModeMerge</i>	Merges multiple SDC files under different modes into a single SDC having equivalent timing analysis

SDC_ModeMerge

Merges multiple SDC files under different modes into a single SDC having equivalent timing analysis

When to Use

A design has several modes of operation. The timing constraint for each mode is captured in SDC files. For each mode, you optimize the design despite conflicting requirements.

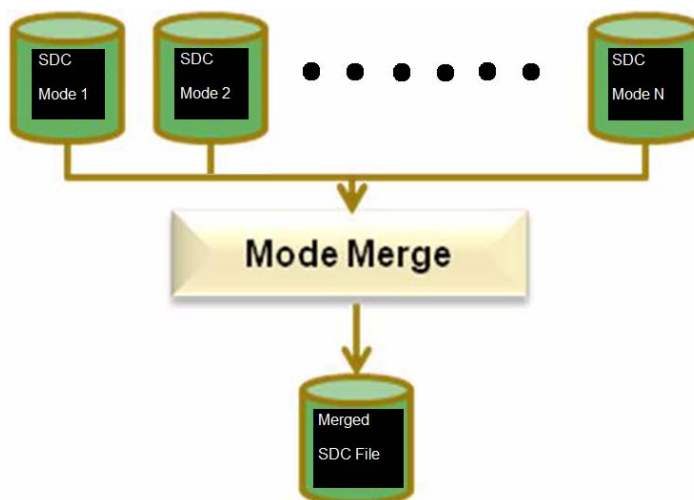
By using this rule, you can create a merged SDC file that accounts for the various modes. As a result, you can use the merged SDC file in implementation tools (synthesis, STA, P&R) to save runtime.

This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *SDC_ModeMerge* rule generate a merged mode SDC by merging the SDC constraints defined in multiple different modes. The merging criteria is to never under-constrain the design for the merged SDC file.

The merging of SDC constraints is performed in a specific order as per the type of constraint. In addition, the conflict resolution between the constraints is dependent on the constraint type. To resolve a conflict, a constraint is either retained, ignored or newly generated.



To understand this rule, refer to the following sections:

- [Prerequisites](#)
- [Merging SDC Files](#)
- [Merging Constraints Considerations](#)
- [Rule Exceptions](#)

The merged constraints are accompanied by a detailed reporting of the constraints. Refer to the [Reports and Related Files](#) section for a description of each worksheet.

Prerequisites

Ensure the SGDC file is as follows:

```
current_design <design-name>
sdc_data -file <SDC-file-names> -mode <mode-name1>
sdc_data -file <SDC-file-names> -mode <mode-name2>
...
sdc_data -file <SDC-file-names> -mode <mode-nameN>
```

If you do not specify *<mode-name1>*, *<mode-name2>*, or *<mode-nameN>*, by default, the value of the mode argument is taken as *none*. Consequently, it is reflected in the merged SDC file.

Merging SDC Files

A mode is used to constrain an aspect of the design. An SDC might be defined to test only the scan logic of the design. While others might be defined to test another aspect of the design. Therefore, multiple modes might overlap. There could be conflicts in the overlapping constraints. For example, [set_case_analysis](#) could be different since a portion of the design might be disabled using [set_case_analysis](#), but the same portion could be tested under another mode.

To understand the scenarios for merging, you need to understand the effect of an SDC constraint.

An SDC constraint defines a design area to which it applies along with timing values.

If both SDC files do not overlap in terms of design area, it implies there is no conflict in merging.

If there is a design area constrained in both SDC files, conflict may arise as follows:

- Design area constrained by SDC constraint is not exactly same. It overlaps but is unequal.
 - clocks instantiated at different objects
- Number of SDC constraints types involved in each mode
 - Three clocks versus two clocks
- SDC constraints constraining the area are different
 - *set_input_delay* versus timing exception
- Constraint value is different
 - *set_case_analysis* value is different

In the merged SDC file, conflict needs to be resolved when creating the merged SDC. These conflicts are resolved so that the effect of the merged SDC file is the same as the cumulative effect of the multiple SDC files.

Since each constraint is unique, conflict resolution is dependent on the constraint types involved.

Merging Constraints Considerations

See [List of Supported SDC Constraints for Equivalence](#) for the supported SDC constraints.

Key considerations while merging constraints are described as follows:

- [Merging of set_case_analysis](#)
- [Merging of set_disable_timing](#)
- [Merging of set_clock_sense](#)
- [Merging of Clocks](#)
- [Merging I/O Delays](#)
- [Merging Timing Exceptions](#)

Merging of set_case_analysis

In the merged mode, the net state satisfies the following table:

	Mode 1	Mode 2	Merged Mode
Same value	value	value	Any mode
Conflict value	value	different value	No mode
Super set case	value	no value	Mode 2

Merging of set_disable_timing

Disable timing constraints are created in the merged mode when they are present in all modes.

Merging of set_clock_sense

Clock sense merging is performed while merging clocks. Read the [Merging of Clocks](#) section for details.

Merging of Clocks

Clocks are generated in merged mode, such that at each flip-flop in the design, same/similar waveforms are propagated cumulatively in individual modes.

Refer to the [Example Code and/or Schematic](#) section for the key considerations while merging clocks.

Merging I/O Delays

For each object constrained by an I/O delay in any mode, I/O delays are created in the merged mode.

The I/O delay list on the object for each mode is determined. Each I/O delay is merged into the merged mode by identifying the mapping clock of the merged mode.

Merging Timing Exceptions

Timing exception constraints specify timing conditions for a path. A constraint could specify a collection of paths using its path specifiers `from`, `through`, and `to`. The `through` option could be used multiple times to exactly specify the path while `from`, `to` are optional. Clocks could also be specified in `from`, `to` options. If a clock is specified, the constraint applies to all the flops that have clock pins, which are reachable by the clock.

Timing exception constraints could define a variety of timing conditions for a path, as follows:

- **Falsification:** Specified through [set_false_path](#) and [set_clock_groups](#)
- **Multicycle nature:** Specified through [set_multicycle_path](#)
- **Maximum and Minimum delay:** Specified through [set_max_delay/](#)
[set_min_delay](#)

A timing exception constraint is generated in merged mode if the path is constrained in both the modes by any of the exception constraints.

A key objective of merging exceptions is to minimize the constraints so that P&R tools can provide rapid timing closure.

A path in the design can be constrained by exception constraints in both the modes. The following scenarios exist while generating an exception constraint in the merged mode:

- Exception constraint types are different
- Exception constraint types are same
 - values are different
 - applies to different timing conditions such as setup/hold etc.

For the first scenario, the following timing exception precedence list is used to determine constraint type for merged-mode:

`set_max_delay`

`set_min_delay`

Rule Exceptions

This rule supports the merging of up to 63 modes.

Parameter(s)

- [tc_modemerge_backref_info](#): Default is yes. Set this parameter to no to not provide the debugging information in the merged SDC file.
- [tc_similar_clocks_tolerance_levels](#): Default is 10. Set the value of this parameter to the tolerance percentage that is used to compare clock waveforms when identifying similar clocks.
- [tc_ignore_modes](#): Default is none. Set the value of this parameter to the mode names that should be ignored while merging modes.
- [tc_regression_mode](#): Default is 0 and the regression mode is disabled. This indicates that the timing exceptions spreadsheet contains all SDC and schematic back references. When you enable regression mode by

setting this parameter to 1, SDC and schematic back references are not created in the timing exceptions spreadsheet.

- *tc_report_backref_all*: Default is 0 and only the first 10,000 rows have SDC and schematic back references. This saves runtime. Set this parameter to 1 to report all SDC and schematic back references in the timing exceptions spreadsheet.
- *tc_report_verbose*: Default is 1 and verbose reporting is enabled. Set this parameter to 0 to disable verbose reporting. The spreadsheet generated by the [SDC_ModeMerge](#) rule does not show ignored timing exceptions.

Constraint(s)

None

Messages and Suggested Fix

Message

The following message appears when multiple SDC files have been merged successfully:

```
[SDC_ModeMerge_02][INFO] Merged SDC is generated for design/
block "<name>"
```

Potential Issues

Not applicable

Consequences of Not Fixing

Not applicable

How to Debug and Fix

Double-click the violation message to view the merged SDC file. In addition, the Spreadsheet Viewer opens and displays the justification for the content of the merged SDC file. Information for each constraint type is displayed in a separate worksheet within Spreadsheet Viewer.

Refer to the [Reports and Related Files](#) section for a description of each worksheet.

Message 2

The following message is generated for SDC commands that are not supported:

[WARNING] Ignoring following unsupported SDC commands: <Unsupported SDC commands separated by commas>

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

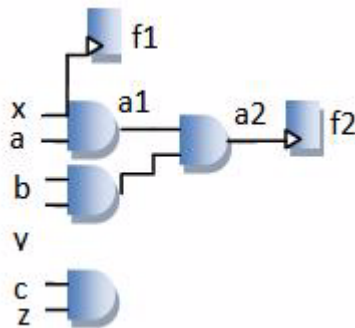
These SDC constraints will be ignored.

How to Debug and Fix

See [List of Supported SDC Constraints for Equivalence](#) for the supported SDC constraints.

Example Code and/or Schematic

This example illustrates the key considerations while merging clocks. In this example, mode1 clocks are propagated in the design with mode1 blocking constraints. Since clock at the x port for mode2 is being blocked, it is not retained in the merged mode. In addition, at the y port clocks have distinct waveforms. Therefore, both are retained.



Model

```
set_case_analysis 1 a
set_case_analysis 1 b
create_clock -period 15 x
```

```
create_clock -period 20 y
create_clock -period 25 z
```

Mode2

```
set_case_analysis 0 a
set_case_analysis 1 b
create_clock -period 10 x
create_clock -period 10 y
create_clock -period 25 z
```

Merged Mode

```
// conflicting blocking constraint removed from a
set_case_analysis 1 b
create_clock -period 15 x1 x
create_clock -period 10 x2 x -add
create_clock -period 10 -name y1 y
create_clock -period 20 -name y2 y -add
create_clock -period 25 z
set_clock_sense -stop_propagation -clock x1 a1/a
```

The same/similar clocks is defined as a clock on the same/similar object having the same/similar clock characteristics. Two clock objects are similar if they reach the same set of sequential clock pins.

Similar clock characteristics are defined as period and waveform edges being within a certain percentage. At the moment, 10% variation is tolerated.

Due to different blocking settings in the merged mode, a clock propagates to more flops in the merged mode than in the individual mode.

Clock at the x port in mode1 reaches more flops in the merged mode since a clock is propagating to more flops in the merged mode than in individual mode, it results in incorrect timing analysis. For equivalent timing analysis,

these clock paths are disabled in the merged mode. Since a physical path might be constrained with multiple clocks belonging to distinct modes, path disabling in the merged mode only applies to appropriate clocks.

[set_clock_sense](#) constraints are generated with `stop_propagation` to block these paths. In case, where path disabling needs to be applied to a subset of the clocks constraining the path, the `clock` option is used to specify such clocks.

In this example, since clock at port `x` in `mode1` is retained, and blocking constraint [set_case_analysis](#) at `a` is removed in the merged mode, a [set_clock_sense](#) constraint is created to block the clock propagation through `a1/a`. Since multiple clocks (clock at `x` in `mode2`) are passing through `a1/a`, blocking clock is specified through the `clock` option.

The paths to be blocked can be determined by comparing the blocking settings of each mode. Wherever the blocking setting of an object is different between the individual and the merged mode, it is noted.

In this example, `a1/a` is one such object. There could be scenarios where there exists no clock pin from a conflicting blocking object. If this is the case, no [set_clock_sense](#) is generated. In this example, if `a1/a` does not traverse to any clock pin, no [set_clock_sense](#) is generated.

After creating the clocks, clock crossing is determined. For interacting clocks belonging to different modes, a [set_clock_groups](#) constraint is created between the clock pair.

Default Severity Label

Info

Rule Group

ModeMerge

Reports and Related Files

The *SDC_ModeMerge* rule generates the merged constraints in SDC format for a design. The merged constraints are accompanied by a detailed reporting of the constraints. For each constraints category, a worksheet provides details about each constraint. This section provides a description of each worksheet.

Merge SDC Files

- [Worksheet for Merging Case Analysis Constraints](#)
- [Worksheet for Merging Clocks Constraints](#)
- [Worksheet for Merging IO Delay Constraints](#)
- [Worksheet for Merging Timing Exceptions Constraints](#)

Worksheet for Merging Case Analysis Constraints

The entries in the spreadsheet are grouped into three categories.

- Values dropped
- Values retained
- New value generated

The columns show merged value, result, reason along with individual mode values along with file and line number for each mode constraint being merged.

	C	D	E	F	G	H	I
	Design object	'merged' value	Result	Reason	'mode1' value (file:line)	'mode2' value (file:line)	'mode3' value (file:line)
1	Values dropped						
5	Values retained						
6	in[3]	1	value retained from 'mode1' 'mode2' 'mode3'	Logical equivalence is found at this object where logical value (1) is same in all the modes	1 (m1.sdc:5)	1 (m2.sdc:5)	1 (m3.sdc:5)
7	New value generated						

Worksheet for Merging Clocks Constraints

In this worksheet, the clocks retained for each design object are shown. All the dropped clocks are grouped together as **Dropped Clocks** group.

The columns for each mode clock shows clocks with period, and file and line numbers for each reference.

Spreadsheets (5): set_case_analysis_top.csv

set_case_analysis_top.csv | clock_top.csv | input_delay_top.csv | output_delay_top.csv | timing_exception_top.csv

Show Header

	C	D	E	F	G	H
	Design object	'merged' clock{period} /	Reason	'mode1' clocks{period} (file:line(s))	'mode2' clocks{period} (file:line(s))	'mode3' clocks{period} (file:line(s))
13	gc/CP	clk040{20}	Only one clock waveform arrives at clock pin gc/CP and hence retained in merged mode with clock name clk040	clk040{20} (m1.sdc: 23)		
15	fd/CP	clk01{12}	Only one clock waveform arrives at clock pin fd/CP and hence retained in merged mode with clock name clk01	clk01{12} (m1.sdc: 15)		
16	Dropped clocks					
17			The clock waveform is getting blocked at 'ab/Z' (clock sense with stop_propagation). Hence dropped in merged mode.	clk050{23} (m1.sdc: 38)		

Worksheet for Merging IO Delay Constraints

This worksheet shows the individual mode and merged mode delay values with reference to the merged mode clocks for each port. Entries are grouped together for each port. The file and line number for each individual mode are shown.

Merge SDC Files

Design object	'merged' delay(clock{period})	'mode1' delay (clock{period}) (file:line(s))	'mode2' delay (clock{period}) (file:line(s))	'mode3' delay (clock{period}) (file:line(s))
in[4]	5 (vclk1{13})	-	5 ("vclk1{13}") (m2.sdc:18)	-
in[4]	min_rise '4' max_rise '5' min_fall '4' max_fall '5' (vclk0{10})	4 ("vclk0{10}") (m1.sdc:33)	-	5 ("vclk1{10}") (m3.sdc:17)
in[4]	min_rise '0.5' max_rise '4' min_fall '0.5' max_fall '4' (clk120{18})	2 ("clk020{18}") (m1.sdc:29)	min_rise '1' max_rise '3' min_fall '1' max_fall '3' ("clk120{18}") (m2.sdc:13 14)	min_rise '0.5' max_rise '4' min_fall '0.5' max_fall '4' ("clk120{18}") (m3.sdc:13 14)

Worksheet for Merging Timing Exceptions Constraints

The entries are grouped together in four groups:

- Exactly same exception constraints defined in all the modes
- Same type exception constraints (with different values) defined in all the modes
- Different type exception constraints on same path
- Ignored exception constraints

The individual mode constraints, and the file and line number are shown.

Spreadsheets (5): set_case_analysis_top.csv

set_case_analysis_top.csv | clock_top.csv | input_delay_top.csv | output_delay_top.csv | **timing_exception_top.csv**

Show Header

	C	D	E	F	G
	'Merged' Constraint	Justification	'mode1' Constraint (file:line(s))	'mode2' Constraint (file:line(s))	'mode3' Constraint (file:line(s))
1	Exactly same exception constraints defined in all the modes.				
2	set_multicycle_path -to pf/D	Exactly same exception constraints defined in all the modes.	set_multicycle_path -to pf/D (m1.sdc:54)	set_multicycle_path -to pf/D (m2.sdc:22)	set_multicycle_path -to pf/D (m3.sdc:22)
3	set_false_path -through fc8/CP	Exactly same exception constraints defined in all the modes.	set_false_path -through fc8/CP (m1.sdc:55)	set_false_path -through fc8/CP (m2.sdc:23)	set_false_path -through fc8/CP (m3.sdc:23)
4	Same type exception constraints (with different values) defined in all the modes.				
7	Different type exception constraints on same path.				
10	Ignored exception constraints.				

Constraints Management Rules

The Constraints Management rules help you manage constraints at different stages of the design. In the current version, the SpyGlass Constraints solution checks for equivalence between two SDC files specified for the same design.

To understand how the SpyGlass Constraints solution checks for equivalence, read the following section:

- [Understanding Equivalence of SDC Files](#)
- [Understanding the Equivalence Flow](#)
- [Specifying the SDC File to Check SDC Equivalence](#)
- [Specifying the Skeleton SDC File to Check SDC Equivalence](#)
- [Checking Equivalence between the Same SDC Commands](#)

The Constraints Management Rules that perform equivalence are listed below:

Rule	Description
ConstMgmtParamSanityCheck	Sets up syntactic checks on parameters
Equiv_SDC	Performs equivalence between two SDC files
Equiv_SDC_Block	Performs equivalence between the top-level and block-level design units
Equiv_SDC_Dual_Design	Performs equivalence between two SDC files of two equivalent designs
Equiv_SDC_Top	Performs equivalence between the top-level design unit
SDC_multimode_equiv	Performs equivalence checks between a merge-mode SDC and individual modes SDC of the same design

Apart from these rules, the [Equiv_SDC_Auto_Detect](#) rule is used to auto detect equivalent ports and/or sequential elements between two designs.

ConstMgmtParamSanityCheck

Performs sanity checks on parameters

When to Use

Use this rule at the RTL, pre-layout, and post-layout stages of the design.

Description

The *ConstMgmtParamSanityCheck* rule is a setup rule that checks whether the specified parameters contain valid values. In addition, if you have specified the parameters through a file, this rule checks whether the file is accessible and can be read.

Parameter(s)

- *equiv_sdc_design_equivalence_file*: Default value is none. Set this parameter to the name of a file that contains the mapping of reference and implement design objects.
- *equiv_sdc_design_ignore_prefix*: Default value is none. Set this parameter to a `string` value to ignore the prefix of the design hierarchy in the equivalent design file.
- *equiv_sdc_ignore_unequivalent_design_obj*: Default value is `yes`. This indicates that the *Equiv_SDC_Dual_Design* rule ignores the design object that does not have an equivalent design object in the equivalence report. Set this parameter to `no` if you want the *Equiv_SDC_Dual_Design* rule not to ignore such design objects.
- *equiv_sdc_constraint_file*: Default value is none. This indicates that equivalence is checked for all SDC commands. Set this parameter to the name of a file containing the SDC commands that need to be checked for equivalence
- *equiv_sdc_tolerance*: Default value is 0. Set this parameter to a tolerance value between 0 and 100 to check for equivalency between two commands.
- *equiv_sdc_ambiguous_clock_file*: Default value is none. Set this parameter to the name of a `file` containing ambiguous clocks.

- *equiv_sdc_clk_suffix*: Default value is none. Set this parameter to a string value specifying the suffix for a virtual clock.
- *equiv_sdc_ignore_value*: Default value is no. Set this parameter to yes to ignore the value between two commands while performing an equivalency check.
- *equiv_sdc_ignore_constant_pins_for_top_block_equivalence*: Default value is no. Set this parameter to yes to ignore the constant power pins, VSS and VDD, lying outside the top block.
- *ignore_incremental_equivalence*: Default value is no. Set this parameter to yes to check the equivalence of other commands even if the equivalence of commands with high precedence is not established.
- *tc_lvpc*: Default value is 20. Set this parameter to a positive integer value to specify the limit for reporting the number of violations per command. Set this parameter to -1 to report all the violations for a constraint.
- *equiv_sdc_enable_one_pass*: Default is no. Set this parameter to yes enable the one-pass flow for dual design equivalence.
- *equiv_sdc_script_file*: Default is none. Set this parameter to the file path of the mapping script file for the one-pass flow.
- *equiv_sdc_script_side_file*: Default is none. Set this parameter to the file path of the side file used in the one-pass flow.

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message appears if the parameter *<param-name>* contains a file name that does not exist:

[FATAL] File specified through parameter *<param-name>* not found

Potential Issues

The violation message appears when a parameter contains a non-existent file name.

Consequences of Not Fixing

If a parameter contains a non-existent file name, SpyGlass does not run.

How to Debug and Fix

Check the file name specified in the parameter.

To fix this problem, ensure that the file, which is specified in the parameter, exists. Specify the correct file name.

Message 2

The following message appears if a file specified in the parameter `<param-name>` fails to open:

[FATAL] File specified through parameter `<param-name>` can not be opened for reading

Potential Issues

The violation message appears when a specified file does not open.

Consequences of Not Fixing

If a parameter contains the name of a file that cannot be opened, SpyGlass does not run.

How to Debug and Fix

Try to open the file or check the attributes of this file for read permissions.

To fix this problem, provide the read permissions to the reported file.

Message 3

The following message appears if a different value is specified in the parameter `<param-name>` that accepts a float value:

[FATAL] Value passed through parameter `<param-name>` should be a float value between 0 to 10.0

Potential Issues

The violation message appears when the value specified in the parameter is not in the range of 0 to 10.

Consequences of Not Fixing

If a parameter contains a float value outside the range of 0 to 10, SpyGlass does not run.

How to Debug and Fix

Check the value specified in the parameter.

To fix this problem, specify the value of the parameter in the range of 0 to 10.

Message 4

The following message appears when no value is supplied to a parameter that accepts a value as an input:

[FATAL] No value passed to parameter <param-name>

Potential Issues

The violation message appears when no value is supplied to a parameter that accepts a value as an input.

Consequences of Not Fixing

If no value is supplied to a parameter that accepts a value as an input, SpyGlass does not run.

How to Debug and Fix

Check whether a value is specified in the parameter.

To fix this problem, specify a valid input.

Message 5

The following message appears when the wrong option <option-name> is supplied to the parameter <param-name>:

[FATAL] Wrong option <option-name> passed to parameter <param name>, valid options are <option-list>

Where,

<option-name> is the wrong option supplied to the parameter.

<option-list> is the correct option or a list of options that can be specified in the parameter.

Potential Issues

The violation message appears when the option specified in the parameter is not valid.

Consequences of Not Fixing

If an invalid option is specified in the parameter, SpyGlass does not run.

How to Debug and Fix

Check whether a valid option is specified in the parameter.

To fix this problem, specify a valid option or a list of options in the parameter, as suggested in the violation message.

Message 6

The following message appears when more than one option is supplied to a parameter that accepts only one option:

[FATAL] Only one option can be passed to parameter <param-name>

Potential Issues

The violation message appears when multiple options are specified in the parameter that only accepts a single option.

Consequences of Not Fixing

If multiple options are specified in the parameter that only accepts a single option, SpyGlass does not run.

How to Debug and Fix

Check the options specified in the parameter.

To fix this problem, specify a valid input to the parameter.

Message 7

The following message appears if you do not specify the [equiv_sdc_design_equivalence_file](#) parameter in the [Equiv_SDC_Dual_Design](#) rule:

[FATAL] "equiv_sdc_design_equivalence_file" parameter is mandatory for rules (except Equiv_SDC_Auto_Detect) to run in dual design mode

Potential Issues

The violation message appears when you do not specify the [equiv_sdc_design_equivalence_file](#) parameter in the [Equiv_SDC_Dual_Design](#) rule.

Consequences of Not Fixing

If you do not specify the [equiv_sdc_design_equivalence_file](#) parameter in the [Equiv_SDC_Dual_Design](#) rule, SpyGlass does not run.

How to Debug and Fix

Check whether the [equiv_sdc_design_equivalence_file](#) parameter is set in the

Equiv_SDC_Dual_Design rule.

To fix this problem, specify the mapping file corresponding to the dual design mode in the parameter reported in the violation message.

Message 8

The following message appears if you specify an inappropriate value in the parameter `<param-name>` that accepts an integer value:

[FATAL] Value passed to parameter `<param-name>` not of integer type

Potential Issues

The violation message appears when the value specified is not an integer type.

Consequences of Not Fixing

If you specify an inappropriate value in the parameter that accepts an integer value, SpyGlass does not run.

How to Debug and Fix

Check the value specified in the parameter.

To fix this problem, specify the integer value to the parameter.

Example Code and/or Schematic

Suppose you have set the following parameters:

```
set_parameter equiv_sdc_ambiguous_file "abc"  
set_parameter equiv_sdc_constraint_file ""  
set_parameter equiv_sdc_tolerance 11  
set_parameter equiv_sdc_ignore_value="yesno" "  
set_parameter tc_lvpc="xyz"
```

Since each parameter is set to an invalid value, the *ConstMgmtParamSanityCheck* rule reports violation messages for each parameter.

Default Severity Label

Fatal

Rule Group

`const_mgmt`

Reports and Related Files

None

Equiv_SDC

Performs equivalence between two SDC files

When to Use

Use this rule to check whether two SDC files are equivalent. This rule is applicable to the RTL, Pre-layout, and Post-layout phases.

Description

The *Equiv_SDC* rule performs the equivalence check between two SDC files of the same top-level or block-level design.

To understand equivalence checking, read the following sections:

- [Basic Concepts for Performing Equivalence Checks](#)
- [Performing Equivalence between set_case_analysis Constraints](#)
- [Performing Equivalence between Clock Constraints](#)
- [Performing Equivalence between Input/Output Delays](#)
- [Performing Equivalence between Input Transition, Clock Transition, Load, Clock Latency, and Clock Uncertainty](#)
- [Performing Equivalence between Timing Exceptions](#)
- [Performing Equivalence between set_disable_timing Constraints](#)
- [Performing Equivalence between set_drive Constraints](#)
- [Performing Equivalence between set_max_fanout Constraints](#)
- [Performing Equivalence between set_max_capacitance Constraints](#)
- [Performing Equivalence between set_max_transition Constraints](#)
- [Performing Equivalence between set_driving_cell Constraints](#)
- [Performing Equivalence between set_dont_touch Constraints](#)
- [Performing Equivalence between set_propagated_clock Constraints](#)
- [Performing Equivalence between set_ideal_network Constraints](#)

Basic Concepts for Performing Equivalence Checks

The two SDC files are specified using reference and implement as the mode names as shown below.

```
current_design top
```

```
sdc_data -file reference.sdc -mode reference
sdc_data -file implement.sdc -mode implement
```

Here, the name of the top design unit is top. The two SDC files for the design top are reference.sdc and implement.sdc. In addition, reference and implement are the modes that differentiate the two SDC files.

The rule performs an equivalence check incrementally between two SDC files in the following order:

1. set_case_analysis
2. create_clock/create_generated_clock/
set_input_delay/set_output_delay

Since an incremental equivalence check is performed, when one check fails, the rule does not perform the equivalence check for the others. For example, if the [set_case_analysis](#) constraints in the two SDC files are not equivalent, then the *Equiv_SDC* rule does not perform the equivalence check further until you fix the problem.

After the incremental equivalence check is performed in the order mentioned above, the following constraints are checked:

```
set_clock_transition/set_input_transition/  
set_clock_uncertainty/set_clock_latency/set_load/  
set_false_path/set_multicycle_path/set_min_delay/  
set_max_delay/set_drive/set_max_fanout/  
set_max_capacitance/set_max_transition/  
set_driving_cell/set_dont_touch/  
set_propagated_clock/set_ideal_network
```

Refer to the [Parameter\(s\)](#) section to see how to perform a non-incremental equivalence check.

There are some commands for which the equivalence check is always performed before the commands specified by you. For details on those commands, refer to the [Understanding the Equivalence Flow](#) section.

Performing Equivalence between set_case_analysis Constraints

The *Equiv_SDC* rule first checks that the [set_case_analysis](#) constraints in the two SDC files are equivalent. This rule considers all the end points, such as all inputs of sequential elements, all output ports, and all input pins of a black box and grey box, in the fan-out of the [set_case_analysis](#) constraints

and reports commands that cause the difference for the specified point. In addition, it also reports the commands that are equivalent. However, the [set_case_analysis](#) constraints, which are blocked, are not reported.

The *Equiv_SDC* rule traverses back from the end points while obtaining the values for the [set_case_analysis](#) constraints and considers only those points where the values differ.

To further understand how this equivalence check is performed, read the [Example Code and/or Schematic](#) and [Messages and Suggested Fix](#) sections.

Performing Equivalence between Clock Constraints

The *Equiv_SDC* rule performs clock equivalence after the [set_case_analysis](#) constraints are equivalent.

The *Equiv_SDC* rule considers all the end points on which the clock ([create_clock](#) and [create_generated_clock](#)) constraints have been applied and checks that the clocks have similar characteristics, polarity, and drive the same set of flops with the same waveform pattern. The effect of the [set_clock_sense](#) constraints is incorporated while determining clock waveforms reaching a flip-flop. Each waveform of the clock reaching a flip-flop is checked for equivalence. In addition, the *Equiv_SDC* rule flags clocks that are found in one SDC file but not in the second.

For two clocks to be equivalent, there should be a fan-in and fan-out relationship between them. The end points considered are:

- All clock pins of flip-flops
- All enable points of latches
- All clock pins of grey boxes
- All the pins of black boxes where the clock is reaching

All other type of clocks do not impact equivalence, therefore are not reported in the [Equivalence Report](#). However, if you want all the clocks to be reported in the Equivalence Report, set the [equiv_sdc_report_blocked_clocks](#) parameter to yes.

The *Equiv_SDC* rule considers two sets of clocks, one in each mode, that has the same overall effect on the design as equivalent. It supports one-to-many and many-to-many clock mapping for clock equivalence. Refer to the [Example 1: One-to-Many Clock Mapping](#) and [Example 2: Many-to-Many Clock Mapping](#) sections for more information.

If the equivalence check fails for either clock or input/output delays, the

Equiv_SDC rule does not perform the equivalence check further, until you have fixed the problem.

To further understand how this equivalence check is performed, read the [Example Code and/or Schematic](#) and [Messages and Suggested Fix](#) sections.

Performing Equivalence between Input/Output Delays

After performing the check for the [create_clock](#) and [create_generated_clock](#) constraints, the *Equiv_SDC* rule performs equivalence checking for the [set_input_delay](#) and [set_output_delay](#) constraints. However, the equivalence check for input/output delays is not dependent on the result of clock equivalence.

A set of input/output delay constraints is equivalent only if their values are the same, they are specified with respect to the matching clock characteristics, and are applied on the same port or pin.

To further understand how this equivalence check is performed, read the [Example Code and/or Schematic](#) and [Messages and Suggested Fix](#) sections.

Performing Equivalence between Input Transition, Clock Transition, Load, Clock Latency, and Clock Uncertainty

The *Equiv_SDC* rule performs equivalence checking for the each of the following commands:

- [set_input_transition](#)
- [set_clock_transition](#)
- [set_load](#)
- [set_clock_latency](#)
- [set_clock_uncertainty](#)

If [set_input_transition](#) is specified with the `-clock` option, the input delay should be applied on that port for the clock. Otherwise, the [set_input_transition](#) constraint is ignored.

In addition, the *Equiv_SDC* rule perform pre- and post-layout checks for the [set_input_transition](#) and [set_clock_transition](#) commands. For details, refer to the [Pre and Post Layout Checks](#) section.

Here, the *Equiv_SDC* rule performs a point-to-point, clock-to-clock, and value-to-value matching and flags a message if the constraints are not equivalent.

NOTE: *Real clocks and virtual clocks that do not affect the design (real clock set on hanging clock port or on blocked path and virtual clocks for which no input/output delay is given) are ignored for such comparison.*

To further understand how this equivalence is performed, read the [Example Code and/or Schematic](#) and [Messages and Suggested Fix](#) sections.

Performing Equivalence between Timing Exceptions

The *Equiv_SDC* rule considers all timing paths that are constrained by the following timing exceptions:

- [set_clock_groups](#)
- [set_false_path](#)
- [set_max_delay/set_min_delay](#)
- [set_multicycle_path](#)

The *Equiv_SDC* rule compares the exception type and other parameters such as the clock domain, *-setup*, *-hold*, *-rise*, *-fall*, and *delay* values and reports a violation in case of a difference.

For supporting equivalence, the [set_clock_groups](#) constraint is treated as equivalent to the [set_false_path](#) constraint.

NOTE: *The *Equiv_SDC* rule considers only those timing exceptions that are reduced to timing paths.*

The *Equiv_SDC* rule ignores invalid commands while performing equivalence checking. For example, if a command `set_false_path -through AND1/Z` is present in one SDC file but not in the other, and if no timing path passes through AND1/Z, the *Equiv_SDC* rule does not report a violation.

The *Equiv_SDC* rule performs equivalence between timing exceptions specified in an expanded form.

To further understand how this equivalence check is performed, read the [Example Code and/or Schematic](#) and [Messages and Suggested Fix](#) sections.

Performing Equivalence between set_disable_timing Constraints

The *Equiv_SDC* rule considers all timing paths that are disabled by [set_disable_timing](#) in any of the two modes. The disabled timing path of one mode, due to `set_disable_timing`, would be compared with disabled timing-paths of the other mode. Also, the falsified timing paths due to

[set_false_path](#) or [set_clock_groups](#) would be compared with the disabled timing paths using the [set_disable_timing](#) command.

The equivalence checks only for the timing paths and not for the clock paths where the clock(s) are blocked using [set_disable_timing](#).

To further understand how this equivalence check is performed, read the [Example Code and/or Schematic](#) section.

Performing Equivalence between set_drive Constraints

Two [set_drive](#) constraints are considered equivalent only if:

- They are applied on the same/equivalent set of input/inout ports
- The value and the optional options are the same for the same port

If none of the `-rise`, `-fall`, `-min`, and `-max` options are specified, the *Equiv_SDC* rule assumes that all the four options are provided. In addition, if in one SDC file, there are two separate constraints for the `-rise` and `-fall` options and in other SDC file, the constraint is specified with none of the two options (hence applies to both `-rise` and `-fall` options) and if the values are same, they are considered equivalent.

You can ignore the comparison of value by setting the value of the [equiv_sdc_ignore_value](#) parameter to `yes`, or you can specify a tolerance value using the [equiv_sdc_tolerance](#) parameter.

To further understand how this equivalence check is performed, read the [Example Code and/or Schematic](#) and [Messages and Suggested Fix](#) sections.

Performing Equivalence between set_max_fanout Constraints

The syntax of the [set_max_fanout](#) constraint is as follows:

```
set_max_fanout <fanout_value> <object_list>
```

Two [set_max_fanout](#) constraints are equivalent only if:

- They are applied on the same/equivalent set of input/inout ports or designs. If the constraint is applied on the current design, then it is considered as applied on individual ports and all the nets. If the constraint is specified for both the current design and the port, then the most constrained value (the minimum of the two) is considered
- Their value is the same for the object list

- While calculating the fan-out value, the most constrained value from the port value and design value is considered. In addition, if the port value is missing but the same current design value is specified, then also the constraints are considered to be equivalent

You can ignore the comparison of value by setting the value of the [equiv_sdc_ignore_value](#) parameter to `yes`, or you can specify a tolerance value using the [equiv_sdc_tolerance](#) parameter.

To further understand how this equivalence check is performed, read the [Example Code and/or Schematic](#) and [Messages and Suggested Fix](#) sections.

Performing Equivalence between `set_max_capacitance` Constraints

The syntax of the [set_max_capacitance](#) constraint is as follows:

```
set_max_capacitance capacitance_value [-clock_path]
[-data_path] [-rise][-fall] object_list
```

The object list can have ports, designs or clocks. The options are valid only with the clock object. Therefore, for the ports and the designs equivalence is performed similar to the [set_max_fanout](#) constraint.

To further understand how this equivalence check is performed, read the [Example Code and/or Schematic](#) and [Messages and Suggested Fix](#) sections.

Performing Equivalence between `set_max_transition` Constraints

The syntax of the [set_max_transition](#) constraint is as follows:

```
set_max_transition transition_value [-clock_path]
[-data_path] [-rise][-fall] object_list
```

The object list can have ports, designs, or clocks. The options are valid only with the clock object. Therefore, for the ports and the designs, the equivalence is performed similar to the [set_max_fanout](#) constraint.

Two [set_max_transition](#) constraints with the clocks are said to be equivalent only if:

- They are applied on equivalent set of clocks
- The value and the options are the same for each pair of equivalent clocks

You can ignore the comparison of value by setting the value of the [equiv_sdc_ignore_value](#) parameter to `yes`, or you can specify a tolerance value using the [equiv_sdc_tolerance](#) parameter.

To further understand how this equivalence check is performed, read the [Example Code and/or Schematic](#) and [Messages and Suggested Fix](#) sections.

Performing Equivalence between `set_driving_cell` Constraints

Equivalence between `set_driving_cell` constraints is performed only after equivalence between clocks and input/output delays has been established.

The syntax of the `set_driving_cell` constraint is as follows:

```
set_driving_cell [-lib_cell lib_cell_name] [-rise] [-fall]
[-min] [-max] [-library lib_name] [-pin pin_name] [-from_pin
from_pin_name] [-multiply_by factor] [-input_transition_rise
rtrans] [-input_transition_fall ftrans] [-clock clock_name]
[-clock_fall] port_list
```

Two `set_driving_cell` constraints are considered equivalent only if:

- They are applied on the same/equivalent set of ports.
- Each `set_driving_cell` command on a port should have the same clock set with the clock edge value and also the other options, such as library name and cell name. The multiply factor and the transition values, if applied, should also be the same. To compare library and library cells, only string comparison is done.

If none of the `-rise`, `-fall`, `-min`, `-max` options are provided, they are assumed as set.

If `set_driving_cell` is specified with the `-clock` option, the input delay should be applied on that port for the clock. Otherwise, the `set_driving_cell` constraint is ignored.

You can ignore the comparison of value by setting the value of the `equiv_sdc_ignore_value` parameter to `yes`, or you can specify a tolerance value using the `equiv_sdc_tolerance` parameter.

To further understand how this equivalence check is performed, read the [Example Code and/or Schematic](#) and [Messages and Suggested Fix](#) sections.

Performing Equivalence between `set_dont_touch` Constraints

Equivalence between the `set_dont_touch` constraints is performed only after equivalence between clocks and input/output delays has been established.

The syntax of the `set_dont_touch` constraint is as follows:

```
set_dont_touch object_list [value]
```


Two [set_dont_touch](#) constraints are considered equivalent only if:

- They are applied on the same/equivalent object list
- Each object contains the same value

Performing Equivalence between [set_propagated_clock](#) Constraints

Equivalence between the [set_propagated_clock](#) constraints is performed only after equivalence between clocks and input/output delays has been established.

The syntax of the [set_propagated_clock](#) constraint is as follows:

```
set_propagated_clock object_list
```

Two [set_propagated_clock](#) constraints are considered equivalent only if they are applied on the same object list.

To further understand how this equivalence check is performed, read the [Example Code and/or Schematic](#) and [Messages and Suggested Fix](#) sections.

Performing Equivalence between [set_ideal_network](#) Constraints

Equivalence between the [set_ideal_network](#) constraints is performed only after equivalence between clocks and input/output delays has been established.

The syntax of the [set_ideal_network](#) constraint is as follows:

```
set_ideal_network [-no_propagate] object_list
```

Two [set_ideal_network](#) constraints are considered equivalent only if:

- They are applied on the same object list.
- Each object contains the same option

To further understand how this equivalence check is performed, read the [Example Code and/or Schematic](#) and [Messages and Suggested Fix](#) sections.

Parameter(s)

- [equiv_sdc_constraint_file](#): Default is none, therefore equivalence is checked for all commands. To check for specific commands, state them in a file and set the value of this parameter to that file.
- [ignore_incremental_equivalence](#): Default is no. Set the value to yes to check the equivalence of other commands even when the equivalence of commands with high precedence cannot be established.

- *equiv_sdc_ignore_value*: Default is no. Set the value to yes to ignore the value between two commands while performing an equivalence check.
- *equiv_sdc_tolerance*: Default is 0. Set the value to any value between 0 and 100 to specify the degree of tolerance when checking for equivalence between commands.
- *tc_lvpc*: Default is 20. Set the value to any positive integer to limit the reporting of violations that have a similar reason. To report all violations, set the value to -1.
- *equiv_sdc_show_violations*: Default is no. Set the value to yes to make the messages visible in the message tree.
- *equiv_sdc_ambiguous_clock_file*: Default value is none. Set this parameter to the name of a file containing ambiguous clocks.
- *equiv_sdc_ignore_sca_for_top_block_equivalence*: Default is no. Set this parameter to yes to not display all messages of type "specified only in top" for the top-block equivalence check in the *Equivalence Report*.
- *equiv_sdc_report_blocked_clocks*: Default is no. Set this parameter to yes if you want all the clocks, such as blocked clocks, to be reported in the *Equivalence Report*.
- *equiv_sdc_clock_precision*: Default is 2 and the clock period comparison is till the second decimal place. You can set this parameter to any value between 1 and 4 to signify the decimal place for clock period comparison.
- *tc_regression_mode*: Default is 0 and the regression mode is disabled. This indicates that the timing exceptions spreadsheet contains all SDC and schematic back references. When you enable regression mode by setting this parameter to 1, SDC and schematic back references are not created in the timing exceptions spreadsheet.
- *tc_report_backref_all*: Default is 0 and only the first 10,000 rows have SDC and schematic back references. This saves runtime. Set this parameter to 1 to report all SDC and schematic back references in the timing exceptions spreadsheet.
- *tc_report_verbose*: Default is 1 and verbose reporting is enabled. Set this parameter to 0 to disable verbose reporting. When verbose reporting is disabled, equivalent timing exceptions are not reported.

- [*equiv_sdc_check_fanin_fanout_for_clocks*](#): Default is `no`. This indicates that the fan-in, fan-out relationship is not checked during clock equivalence. Set this parameter to `yes` to consider fan-in, fan-out relationships during clock equivalence. See [*Example: Impact on Clock Equivalence*](#).
- [*equiv_sdc_fast_exceptions_equivalence*](#): Default is `no`. Set this parameter to `yes` to enable the new equivalence check algorithm. This algorithm provides runtime improvement for timing exception equivalence.
- [*equiv_sdc_report_type*](#): Default is `both`. This means that both equivalent and inequivalent constraints are reported. Set this parameter to `equiv_only` to report only equivalent constraints and `inequiv_only` to report inequivalent constraints only.
- [*equiv_sdc_disambiguate_same_names_clock*](#): Default is `no`. When you set this parameter to `yes`, you need not specify the ambiguous clocks which have the same name, through the [*equiv_sdc_ambiguous_clock_file*](#) parameter.

Constraint(s)

SDC

- [*create_clock*](#) (Mandatory): Use to create a clock
- [*create_generated_clock*](#) (Mandatory): Use to create a clock
- [*set_case_analysis*](#) (Optional): Use to specify the case analysis conditions
- [*set_clock_sense*](#) (Optional): Use to specify the clock propagation conditions
- [*set_sense*](#) (Optional): Use to specify the clock propagation conditions

SGDC

- [*current_design*](#) (Mandatory): Use to specify the directive scope

Messages and Suggested Fix

If you do not specify the mode names (`implement/reference`) for either of the two SDC files, the *Equiv_SDC* rule generates the following violation messages:

```
[FATAL] No schema with -mode "implement" is provided for design "top"
```

[FATAL] No schema with -mode "reference" is provided for design "top"

[FATAL] No schema with -mode "reference" and -mode "implement" is provided for design "top"

Report Generation Messages

The following message appears for block <block-name>:

[WARNING] Equivalence report is prepared for block "<block-name>"

or

[INFO] Equivalence report is prepared for block "top" (No In-equivalence reported)

Potential Issues

Not applicable

Consequences of Not Fixing

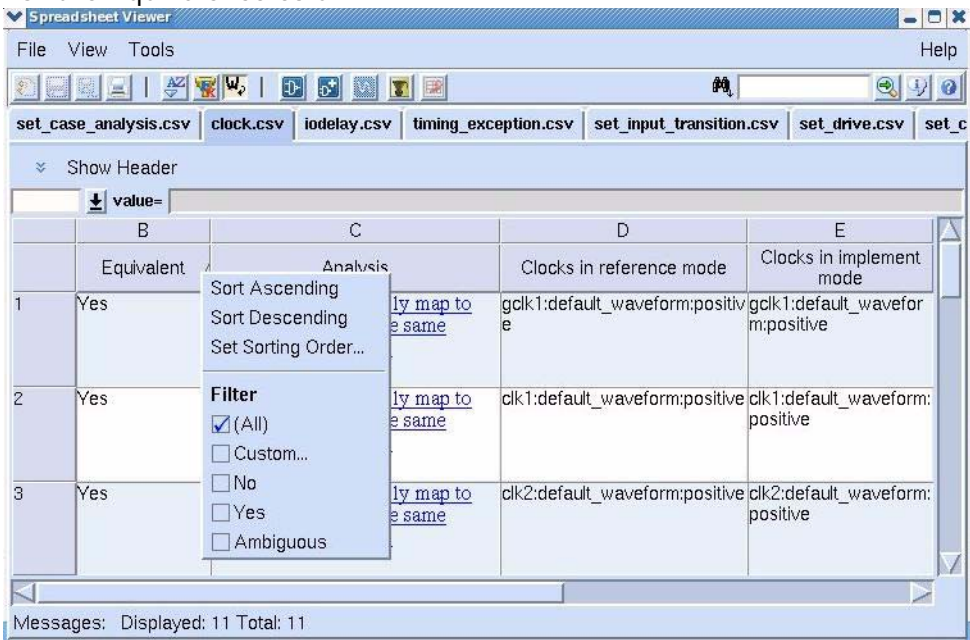
The Equivalence report contains equivalence information on SDC constraints.

How to Debug and Fix

Double-click the message to view the [Equivalence Report](#). The equivalence mapping information is displayed in a separate tab for each constraint. The first column of each tab displays the status of equivalence. Browse each tab to identify constraints that are matching, not matching, or have an ambiguous equivalence status.

To sort the columns, right-click the column header and select the appropriate filter. The following image shows the filtering options available

for the Equivalence column.



To view the schematic associated with a row, click the corresponding Analysis cell and then click the **Modular Schematic** or **Incremental Schematic** icon.

To view information on the generated report, click the **Show Header** link.

Unsupported SDC Constraints Message

The following message appears when unsupported SDC commands are detected.

[WARNING] Ignoring following unsupported SDC commands:
<Unsupported SDC commands separated by commas>

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

Equivalence checking will not be done for these commands.

How to Debug and Fix

See [List of Supported SDC Constraints for Equivalence](#) for the supported SDC

constraints.

Constraint-Specific Messages

Use the information in this section to debug violation messages pertaining to the following constraints. These messages appear in the Message Tree only if you have set the *equiv_sdc_show_violations* to *yes*.

- *set_case_analysis Constraints Messages*
- *Clock Constraints Messages*
- *Input/Output Delay Constraints Messages*
- *Input Transition, Clock Transition, Load, Clock Latency, and Clock Uncertainty Constraints*
- *Timing Exception Constraints*
- *set_drive Constraints*
- *set_max_fanout Constraints*
- *set_max_capacitance Constraints*
- *set_max_transition Constraint*
- *set_driving_cell Constraints*
- *set_dont_touch Constraints*
- *set_propagated_clock Constraints*
- *set_ideal_network Constraints*

set_case_analysis Constraints Messages

The *Equiv_SDC* rule generates the following message when the *set_case_analysis* constraints are not equivalent:

[WARNING] *set_case_analysis* are not equivalent as their effect differ at point

Potential Issues

The potential issues are as follows:

- the *set_case_analysis* is not applied in the fan-in of the reported object,
- the constant value applied does not reach the reported object, or

- the values applied in both modes are different.

Consequences of Not Fixing

The consequences of not fixing are

- If [set_case_analysis](#) is not equivalent, the result of equivalence of other exceptions might be incorrect.
- Two SDC files considered for equivalence do not belong to same mode. Hence should have not been considered for equivalence.

How to Debug and Fix

The schematic shows the path that has differing case analysis values. The SDC file highlights the lines causing the difference. The different value is being propagated on the reported object either in both modes or value X is being propagated in both modes. However, a different constant value is applied in the fan-in of the reported object. The constant value due to VSS/VDD is also considered.

The [Equivalence Report](#) generated for case analysis is grouped under the following headings:

- specified only in mode1
- specified only in mode2
- specified in both

For the [Equiv_SDC_Block](#) rule, mode1 would be flatTop and mode2 would be block. To not display the **specified only in flatTop** group, set the [equiv_sdc_ignore_sca_for_top_block_equivalence](#) parameter to yes.

Clock Constraints Messages

To understand the violation messages pertaining to clock constraints, you first need to understand clock waveforms and polarity.

A clock can have multiple waveforms, as follows:

1. default_waveform
2. rise_triggered_high_pulse
3. rise_triggered_low_pulse
4. fall_triggered_high_pulse
5. fall_triggered_low_pulse

default_waveform is the clock's waveform as stated in the clock

definition in the SDC file. Other waveforms are stated through the pulse option of `set_clock_sense`.

A clock waveform can be inverted through inverters in the clock path. This is referred to as the polarity of the clock waveform. Polarity can be either positive or negative.

The violation messages mention the clock waveform and polarity in addition to the clock name, as follows:

```
<clock-name>:<waveform>:<polarity>
```

Therefore, if a clock equivalence violation message appears for a clock, the waveform and polarity is mentioned to correctly identify the waveform.

Message 1

The following message appears when a flip-flop is driven by one clock's waveform `<clk-waveform>` in one mode (reference/implement) and by multiple clocks in the second mode (implement/reference):

- Clock equivalence based on clock propagation to flops

[WARNING] Flops driven by clock `<clk-waveform>` in `<mode1-name>` are driven by multiple clocks `<clk-waveform-list>` in `<mode2-name>`

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

Only one clock waveform is sampling the flip-flop in `<mode1-name>` but multiple clocks waveforms are sampling that flip-flop in `<mode2-name>`. The schematic highlights the path to the clock-port in both modes. The SDC file highlights the lines causing the difference.

Message 2

The following message is generated when a flip-flop is driven by a clock's waveform `<clk-waveform>` in one mode `<mode1-name>` but not in the second mode `<mode2-name>`:

- Clock equivalence based on clock propagation to flops

[WARNING] Flops driven by clock `<clk-waveform>` in `<mode1-name>` are not driven by any clock in `<mode2-name>`

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The flip-flop is being sampled by a clock waveform in `<mode1-name>` only. It may be possible that either the clock is not created in `<mode2-name>` or the clock is getting blocked before reaching the clock pin of the flip-flop. The schematic highlights the path to the clock-port in both modes. The SDC file highlights the lines causing the difference.

Message 3

The following message appears when a flip-flop is driven by multiple clocks waveforms in both reference and implement.

- Clock equivalence based on clock propagation to flops

[WARNING] Flops driven by multiple clocks `<clk-waveforms>` in `<mode1-name>` are driven by multiple clocks `<clk-waveforms>` in `<mode2-name>`

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

In both modes, multiple clocks waveforms are sampling the flip-flop. In this case, the mapping of clocks waveforms must be specified in the ambiguous clock-mapping file. Otherwise, the rule will not do equivalence of other constraints. The schematic highlights the path to the clock-port in both modes. The SDC file highlights the lines causing the difference.

Message 4

The following message appears when a flip-flop is driven by clock waveform in both modes but the clock characteristic is different:

- Clock equivalence based on clock propagation to flops

[WARNING] Flops driven by clock `<clk-waveform1>` in `<mode1-name>` is driven by clock `<clk2-waveform>` in `<mode2-name>`

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The timing characteristics of clocks waveforms reported are not matching. Therefore, update the timing characteristics of the clock. The schematic highlights the path to the clock-port in both modes. The SDC file highlights the lines causing the difference.

Message 5

The following message appears when two flip-flops are driven by one clock

waveform *<clk1-waveform>* in mode *<mode1-name>* and only one flip-flop is driven by the clock *<clk2-waveform>* in the second mode *<mode2-name>*:

- Clock equivalence based on clock propagation to flops

[WARNING] Flops driven by clocks *<clk-waveform>* in *<mode1-name>* are not driven by clocks *<clk2-waveform>* in *<mode2-name>*

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

In one mode, more than one flip-flop is being driven by the *clk-waveform*. While, in another mode, all those flops are not driven by *clk2-waveform*. It may be due to the difference in the count of the port-pin list of the clocks waveform in both modes. The schematic highlights the path to the clock-port in both modes. The SDC file highlights the lines causing the difference.

Message 6

The following message appears when a flip-flop is driven by the same clock waveform in both modes, but the polarity of both of the clocks is different:

- Clock equivalence based on clock propagation to flops

[WARNING] Flops driven by clock *<clk-waveform1>* in *<mode1-name>* is driven by clock *<clk2-waveform>* in *<mode2-name>*; clock polarity at flop is different in both modes

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The schematic highlights the path to the clock-port in both modes. The SDC file highlights the lines causing the difference. Update the clock polarity of either clock.

Potential Issues

The violation message explicitly states potential issues.

Consequences of Not Fixing

If the clocks characteristics sampling the same flip-flop is not same, you are performing timing analysis for the same design using different assumptions. Since only one of them could be valid, there is a chance that analysis is not appropriate and your chip may fail though your timing

numbers will be met.

If the number of clocks sampling the same flip-flop is different, you are performing one set/mode of analysis in one case, while doing multi-mode analysis in the other case. From an equivalence perspective, these would not be equivalent. Therefore, modes that are left out in one case should be provided so that meaningful equivalence check can be done.

Input/Output Delay Constraints Messages

Message 1

The following message appears when an input/output delay `<delay-type>` is set on port `<port-name>` for only mode `<mode-name>`:

```
[WARNING] <delay-type> for port <port-name> with clocks <clocks-name> set only in <mode-name>
```

Where, delay type can be `set_input_delay` or `set_output_delay`.

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The delay (input/output) is set only in one mode. The delay has to be set for the other mode also. It may be possible that the delay is specified in both modes, but the timing characteristics of the clock are not matching. The SDC file highlights the lines causing the difference.

Message 2

The following message appears when an input/output delay `<delay-type>` is set on port `<port-name>` for only mode `<mode-name>`

```
[WARNING] <delay-type> for port <port-name> with clocks <clk1-name-list> set in reference and with clocks <clk2-name-list> set in implement
```

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The delay is set for the port in both modes, but either the delay value is not equal or the timing-characteristics of the clocks with respect to which the delay is specified are not equal. The SDC file highlights the lines causing the difference.

Potential Issues

The violation message explicitly states potential issues.

Consequences of Not Fixing

If delay is not specified for one mode, the timing analysis results will not match between the two modes being compared. Either there could be more re-work later or analysis would be incomplete leading to timing closure issues later.

If delay is specified but with different clocks for two modes, the timing paths being considered for two modes are different and hence timing analysis is not equivalent. At least one of the SDC files is incorrect. If you go ahead with wrong SDC file, timing closure would occur with the wrong SDC file. This could lead to failure of the chip.

Input Transition, Clock Transition, Load, Clock Latency, and Clock Uncertainty Constraints

Message 1

The following message appears when the input transition for a port `<port-name>` is not equivalent:

```
[WARNING] set_input_transition is not equivalent (Reference = <value>, Implement = <value>) for port <port-name>
```

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The value of the `max`, `rise`, `min`, and `fall` options specified in the reference and implement modes are not the same. If any of the options are not matching, all options are reported. Ignore the values which are matching and update the non-matching option in the SDC file.

Message 2

The following message appears when the input transition for a port `<port-name>` is not equivalent:

```
[WARNING] set_input_transition is not equivalent (Reference = <value>, Implement = <value>) for port <port-name> and clock <clk-name> with clock_fall
```

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The value of the `max`, `rise`, `min`, and `fall` options specified in the reference and implement modes are not the same or the

-clock_fall option is specified in either or both modes. If any of the options are not matching, all options are reported. Ignore the values which are matching and update the non-matching option in the SDC file.

Message 3

The following message appears when the input transition for a port `<port-name>` is set only in one mode `<mode-name>`:

[WARNING] set_input_transition is given for port `<port-name>` only in `<mode-name>` (`<value>`)

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The [set_input_transition](#) constraint is specified in one mode only. You need to specify it for the other mode. Either remove the [set_input_transition](#) constraint from the SDC file of the mode reported in the violation message, or specify the [set_input_transition](#) constraint in the SDC file of the other mode.

Message 4

The following message appears when the input transition for a port `<port-name>` is set only in one mode `<mode-name>`:

[WARNING] set_input_transition is given for port `<port-name>` and clock `<clock-name>` only in `<mode-name>` (`<value>`)

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The [set_input_transition](#) constraint is specified in one mode only. You need to specify it for the other mode. If the equivalent clock is reported then the [set_input_transition](#) constraint is not specified for the port, equivalent-clock pair.

Message 5

The following message appears when the input transition for a port `<port-name>` is set only in one mode `<mode-name>`:

[WARNING] set_input_transition is given for port `<port-name>` and clock `<clock-name>` with clock_fall only in `<mode-name>` (`<value>`)

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The [set_input_transition](#) constraint is specified in one mode only. You need to specify it for the other mode. Either remove the [set_input_transition](#) constraint from the SDC file of the mode reported in the violation message, or specify the [set_input_transition](#) constraint in the SDC file of the other mode.

Message 6

The following message appears when the input transition for a port `<port-name>` and clock `<clk-name>` is set only in one mode `<mode-name>`:

```
[WARNING] set_input_transition is given for port <port-name>
and clock <clk-name> only in <mode1-name> (<value>); equivalent
clock in <mode2-name> is <clk1-name>
```

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The [set_input_transition](#) constraint is specified in one mode only. You need to specify it for the other mode. The [set_input_transition](#) constraint is not specified for the port, equivalent-clock pair.

Message 7

The following message appears when the input transition for a port `<port-name>` and clock `<clk-name>` is set only in one mode `<mode-name>`:

```
[WARNING] set_input_transition is given for port <port-name>
and clock <clk-name> with clock_fall only in <mode-name>
(<value>); equivalent clock in <mode1-name> is <clk1-name>
```

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The [set_input_transition](#) constraint is specified in one mode only. You need to specify it for the other mode. Either remove the [set_input_transition](#) constraint from the SDC file of the mode reported in the violation message, or specify the [set_input_transition](#) constraint with the `-clock_fall` option in the SDC file of the other mode.

Message 8

The following message appears when the input transition for a port `<port-name>` and clock `<clk-name>` is set in both modes but the values are different:

[WARNING] `set_input_transition` is not equivalent (Reference = `<value>`, Implement = `<value>`) for port `<port-name>` and clock `<clk-name>`

Potential Issues

Violation Message 1 to Message 8 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 1 to Message 8.

If input transition between the two SDC files being compared are not equivalent for a particular port, it would mean that Static Timing Analysis tool would infer different delay numbers for the ramp time, leading to varied results. This leads to incorrect analysis for one of them and multiple iterations for timing closure.

How to Debug and Fix

The value of the `max`, `rise`, `min`, and `fall` options specified in reference and implement modes are not the same. If any of the options are not matching, all options are reported. Ignore the values that are matching and update the non-matching option in the SDC file.

Message 9

The following message appears when `set_clock_latency` set for a port or clock in both modes is not equivalent:

[WARNING] `set_clock_latency` (`<source or network>`) is not equivalent (Reference = `<value1>`, Implement = `<value2>`) for `<port-clk-name>`

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The value of all the options (`early`, `late`, `max`, `min`, `rise`, `fall`) are not the same in both modes. If any of the options are not matching then all the options are being reported. Ignore the values that are matching and update the non-matching options in the SDC file.

Message 10

The following message appears when *set_clock_latency* is set for a port/clock *<port-clk-name>* in only mode *<mode-name>*:

[WARNING] set_clock_latency (<source or network>) is given for <port-clock-name> only in <mode1-name> (value); equivalent clock in <mode2-name> is <port-clock-name1>

Refer to *Potential Issues* and *Consequences of Not Fixing*.

How to Debug and Fix

The *set_clock_latency* constraint source or network is not specified for both modes. In the SDC files, either specify the *set_clock_latency* constraint for the equivalent clock of Mode 2 or remove the *set_clock_latency* constraint from Mode 1.

Message 11

The following message appears when *set_clock_latency* is set for a port/clock *<port-clk-name>* in only mode *<mode-name>*:

[WARNING] set_clock_latency (<source or network>) is given for <port-clock-name> only in <mode1-name> (value)

Potential Issues

Violation Message 9 to Message 11 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 9 to Message 11.

If latency between the two SDC files being compared are not equivalent for a particular port, it would mean that the Static Timing Analysis tool would infer different delay number for the clock network, leading to varied results. This leads to incorrect analysis for one of them and multiple iterations for timing closure.

How to Debug and Fix

The *set_clock_latency* constraint source or network is not specified for both modes. For equivalence to be performed, ensure the *set_clock_latency* constraint is specified in the SDC files for both modes.

Message 12

The following message appears when *set_clock_transition* set for a clock *<clk-name>* is not equivalent in both modes:

[WARNING] `set_clock_transition` is not equivalent (Reference = <value1>, Implement = <value2>) for clock <clk-name>

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The value of all the options (`max`, `min`, `rise`, `fall`) are not the same in both modes. If any of the options are not matching then all the options are being reported. Ignore the values that are matching and update the non-matching option in the SDC file.

Message 13

The following message appears when `set_clock_transition` is set for a clock <clk-name> in only one mode <mode-name>:

[WARNING] `set_clock_transition` is given for clock <clk-name> only in <mode1-name>; equivalent clock in <mode2-name> is <clk-name1>

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The `set_clock_transition` constraint is not specified for both modes. In addition, the `set_clock_latency` constraint is not specified for the equivalent clock in Mode 2.

Message 14

The following message appears when `set_clock_transition` is set for a clock <clk-name> in only one mode <mode-name>:

[WARNING] `set_clock_transition` is given for clock <clk-name> only in <mode1-name>

Potential Issues

Violation Message 12 to Message 14 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 12 to Message 14.

If clock transition between the two SDC files being compared are not equivalent for a particular clock, it would mean that the Static Timing Analysis tool would infer different delay number for the clock ramp time, leading to varied results. This leads to incorrect analysis for one of them and multiple iterations for timing closure.

How to Debug and Fix

The [set_clock_transition](#) constraint is not specified for both modes. For equivalence to be performed, ensure the [set_clock_transition](#) constraint is specified in the SDC files for both modes.

Message 15

The following message appears when [set_clock_uncertainty](#) set from clock <clk1-name> to clock <clk2-name> is not equivalent in both the modes:

[WARNING] set_clock_uncertainty is not equivalent (Reference = <value1>, Implement = <value2>) for clock <clk1-name> to clock <clk2-name>

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The value of all the options (setup/hold, rise_from/fall_from, rise_to/fall_to) are not the same in both modes. If any of the options are not matching, all options are reported. Ignore the values that are matching and update the non-matching option. If the equivalent from and to clock is reported then the [set_clock_uncertainty](#) is not specified from *clk1-name* to *clk2-name*.

Message 16

The following message appears when [set_clock_uncertainty](#) set from clock <clk1-name> to clock <clk2-name> in only one mode <mode-name>:

[WARNING] set_clock_uncertainty is given from clock <clk1-name> to clock <clk2-name> only in <mode-name> (value); equivalent from clock is <clk11-name> and to clock is <clk21-name>

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The value of all the options (setup/hold, rise_from/fall_from, rise_to/fall_to) are not the same in both modes. If any of the options are not matching, all options are reported. Ignore the values that are matching and update the non-matching option. In addition, the [set_clock_uncertainty](#) constraint is not specified from *clk1-name* to *clk2-*

name.

Message 17

The following message appears when [set_clock_uncertainty](#) set for a clock *<clk-name>* is not equivalent in both the modes:

[WARNING] set_clock_uncertainty is not equivalent (Reference = *<value1>*, Implement = *<value2>*) for clock *<clk-name>*

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The value of all the options (setup/hold, rise_from/fall_from, rise_to/fall_to) are not the same in both modes. If any of the options are not matching, all options are reported. Ignore the values that are matching and update the non-matching option.

Message 18

The following message appears when [set_clock_uncertainty](#) set for port *<port-name>* is not equivalent in both the modes:

[WARNING] set_clock_uncertainty is not equivalent (Reference = *<value1>*, Implement = *<value2>*) for port *<port-name>*

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The value of all the options (setup/hold, rise_from/fall_from, rise_to/fall_to) are not the same in both modes. If any of the options are not matching, all options are reported. Ignore the values that are matching and update the non-matching option.

Message 19

The following message appears when [set_clock_uncertainty](#) is set for clock *<clk-name>* in only mode *<mode-name>*:

[WARNING] set_clock_uncertainty is given for clock *<clk-name>* only in *<mode1-name>* (*value*); equivalent clock in *<mode2-name>* is *<clk-name1>*

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The *set_clock_uncertainty* constraint is specified only in one of the modes. In addition, the *set_clock_uncertainty* constraint is not specified for the clock mentioned in the violation message.

Message 20

The following message appears when *set_clock_uncertainty* is set for port *<port-name>* in only mode *<mode-name>*:

[WARNING] set_clock_uncertainty is given for port *<port-name>* only in *<mode1-name>* (value)

Potential Issues

Violation Message 15 to Message 20 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 15 to Message 20.

If clock uncertainty between the two SDC files being compared are not equivalent for a particular clock, it would mean that the Static Timing Analysis tool would infer different delay number for the clock network, leading to varied results. This leads to incorrect analysis for one of them and multiple iterations for timing closure.

How to Debug and Fix

The *set_clock_uncertainty* constraint is specified only in one of the modes. For equivalence to be performed, ensure the *set_clock_uncertainty* constraint is specified in the SDC files for both modes.

Message 21

The following message appears when *set_load* set on a port *<port-name>* is not equivalent in both the modes:

[WARNING] set_load is not equivalent (Reference = *<value1>*, Implement = *<value2>*) for port *<port-name>*

Refer to *Potential Issues* and *Consequences of Not Fixing*.

How to Debug and Fix

The value of all the options (*max*, *min*, *rise*, *fall*) are not same in both modes. If any of the options are not matching then all the options are being reported. Ignore the values that are matching and update/modify non-matching option.

Message 22

The following message appears when *set_load* is set for a port *<port-name>* in only mode *<mode-name>*:

[WARNING] *set_load* is given for port *<port-name>* only in *<mode-name>* (value)

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The *set_load* is not specified for both modes.

Message 23

The following message appears when *set_load* set on a net *<net-name>* is not equivalent in both the modes:

[WARNING] *set_load* is not equivalent (Reference = *<value1>*, Implement = *<value2>*) for net *<net-name>*

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The value of all the options (max, min, rise, fall) are not same in both modes. If any of the options are not matching then all the options are being reported. Ignore the values that are matching and modify the non-matching option.

Message 24

The following message appears when *set_load* is set for a net *<net-name>* in only mode *<mode-name>*:

[WARNING] *set_load* is given for net *<net-name>* only in *<mode-name>* (value)

Potential Issues

Violation Message 21 to Message 24 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 21 to Message 24.

If load between the two SDC files being compared is not equivalent for a specific port/net, the Static Timing Analysis tool would infer different delay number leading to varied results. This leads to incorrect analysis for one of

them and multiple iterations for timing closure.

How to Debug and Fix

The [set_load](#) is not specified for both modes.

Pre and Post Layout Checks

The *Equiv_SDC* rule also performs post layout and pre layout checks for [set_input_transition](#) and [set_clock_transition](#) constraints only. However, post layout and pre layout checks are done only if the `-level` argument is set to post-layout and pre-layout for each mode.

The *Equiv_SDC* rule flags the following messages if [set_input_transition](#)/[set_clock_transition](#) in the first mode is not applied at pre-layout/post-layout level, but [set_clock_transition](#)/[set_input_transition](#) is applied at post-layout/pre-layout level in the second mode:

Messages

[WARNING] set_input_transition not applied for clock port <port1-name> in < mode1-name > at prelayout but set_clock_transition is applied for clock port <port2-name> in <mode2-name > at postlayout

[WARNING] set_clock_transition not applied for clock <clk-name> in <mode1-name> at postlayout but set_input_transition is applied for clock port <port-name> in <mode2-name> at prelayout

How to Debug and Fix

Set the [set_input_transition](#) constraint for the clock or the clock-port as specified in the violation message. Updated the SDC file.

Timing Exception Constraints

Message 1

The following message appears when [set_multicycle_path](#) with different multiplier values in both the modes is given for a path segment with different options for both the modes:

[WARNING] set_multicycle_path with different multiplier (Reference = <multiplier1>, Implement <multiplier2>) is given for path segment <segment-name> with options (Reference = <option1>, Implement = <option2>)

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The multiplier value or the options are not the same in both modes.

The timing path constrained by the specified timing exception command is highlighted. In addition, the lines causing the difference are highlighted in the SDC file.

Message 2

The following message appears when [set_max_delay/set_min_delay](#) with different delay values in both the modes is given for a path segment with different options for both the modes:

[WARNING] <delay-type> with different delay (Reference = <delay1>, Implement <delay2>) is given for path segment <segment-name> with options (Reference = <option1>, Implement = <option2>)

Where, <delay-type> can be either be [set_min_delay](#) or [set_max_delay](#).

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The delay value or the options are not the same in both modes.

The timing path constrained by the specified timing exception command is highlighted. In addition, the lines causing the difference are highlighted in the SDC file.

Message 3

The following message appears when different timing exceptions are given for the path segment <segment-name> in both the modes:

[WARNING] Different timing exceptions (Reference = <exception1> with options <option1>; Implement <exception2> with options <option2>) is given for path segment <segment-name>

NOTE: *Since there are no options, such as -setup and -hold, in [set_clock_groups](#), the options do not appear. Hence, the exception description will not contain the 'with options' phrase.*

NOTE: *Irrespective of the properties, such as mutually exclusive or asynchronous, SpyGlass considers the falsified timing-paths by [set_clock_groups](#) between the clock groups*

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

For a timing path different timing exception is specified.

The timing path constrained by the specified timing exception command is highlighted. In addition, the lines causing the difference are highlighted in the SDC file.

Message 4

The following message appears when an exception *<exception>* is given for a path segment *<segment-name>* with options in only one mode *<mode-name>*:

[WARNING] *<exception>* is given for path segment *<segment-name>* with options *<options1>* only in *<mode-name>*

Where, the exception can be [set_clock_groups](#), [set_false_path](#), [set_multicycle_path](#), [set_max_delay/set_min_delay](#).

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The timing exception is not specified in both modes. The timing path constrained by the specified timing exception command is highlighted. In addition, the lines causing the difference are highlighted in the SDC file.

Message 5

The following message appears when an exception *<exception>* is given for a path segment *<segment-name>* with options in both modes but in *<mode1-name>* the path segment is not fully covered by that constraint (*<exception>*):

[WARNING] *<exception>* is given for path segment *<segment-name>* with options *<options1>* only in *<mode-name>* but in *<mode1-name>* at least one path is not constrained

Where, the exception can be [set_clock_groups](#), [set_false_path](#), [set_multicycle_path](#), [set_max_delay/set_min_delay](#).

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

The timing exception is not specified in both modes or all the path is not constrained in another mode. The timing path constrained by the specified timing exception command is highlighted. In addition, the lines causing the difference are highlighted in the SDC file.

Message 6

The following message appears when *set_false_path* with different options is given for a path segment *<segment-name>*:

[WARNING] *set_false_path* with different options (Reference = *<option1>*; Implement = *<option2>*) is given for path segment *<segment-name>*

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

In Mode 2, the *set_clock_groups* constraint is specified. However, in Mode 1, the *set_false_path* constraint is not specified with all the options.

The timing path constrained by the specified timing exception command is highlighted. In addition, the lines causing the difference are highlighted in the SDC file.

Message 7

The following message appears when a timing exception is provided for the path, which is not specified with clocks, with the options stated in the violation message:

[WARNING] Options *rise_from/fall_from/rise_to/fall_to/rise_through/fall_through* in timing exception's concerning data paths will be interpreted as *from/to/through* respectively. e.g. *rise_from* as *from*

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

Double-click the violation message. View the SDC file. All constraints that are causing the violation are highlighted.

Update the options stated in path which is not specified with clocks. Use the:

- *from* option instead of *rise_from/fall_from*
- *to* option instead of the *rise_to/fall_to* option
- *through* option instead of the *rise_through/fall_through*

Refer to the examples section of [Timing Exception Constraints](#) to understand the constraints specifications that can cause this violation.

Message 8

The following message appears when [set_false_path](#) with incomplete options in <mode-name1> with respect to [set_clock_groups](#) in <mode-name2> is given for a path segment <segment-name>:

[WARNING] set_false_path with incomplete options (<mode-name1> = <options>) w. r. t. set_clock_groups(<mode-name2>) is given for path segment <segment-name>

Potential Issues

Violation Message 2 to Message 8 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 2 to Message 8.

If timing exceptions are not equivalent, this would mean that there is a path in one mode that is being timed while in other mode there is an exception on it. Timing analysis would be different and if this is not corrected, it can lead to chip failure.

How to Debug and Fix

In Mode 2, the [set_clock_groups](#) constraint is specified. However, in Mode 1, the [set_false_path](#) constraint is not specified with all the options.

The timing path constrained by the specified timing exception command is highlighted. In addition, the lines causing the difference are highlighted in the SDC file.

set_drive Constraints

The following messages appears when [set_drive](#) is given with different options for the same port <port-name>.

Message 1

[WARNING] set_drive is not equivalent (reference = "-rise -fall -max -min: <value>", implement = "-rise -fall -max -min: <value>") for port <port-name>

For information on debugging, click [How to Debug and Fix](#).

Message 2

[WARNING] set_drive is not equivalent (reference = "-rise -fall -min: <value>", implement = "-rise -fall -max: <value>") for port <port-name>

For information on debugging, click [How to Debug and Fix](#).

Message 3

[WARNING] set_drive is not equivalent (reference = "-rise -fall -max: <value>", implement = " -rise -fall -min: <value>") for port <port-name>

For information on debugging, click [How to Debug and Fix](#).

Message 4

[WARNING] set_drive is not equivalent (reference = "-fall -min: <value>, -rise -min: <value>", implement = " -rise -max: <value>, -fall -max: <value>") for port <port-name>

For information on debugging, click [How to Debug and Fix](#).

Message 5

[WARNING] set_drive is not equivalent (reference = "-fall -max: <value>, -rise -max: <value>", implement = " -fall -min: <value>, -rise -min: <value>") for port <port-name>

For information on debugging, click [How to Debug and Fix](#).

Message 6

[WARNING] set_drive is not equivalent (reference = "-rise -fall -max: <value>", implement = " -rise -max: <value>, -fall -max: <value>") for port <port-name>

For information on debugging, click [How to Debug and Fix](#).

Message 7

[WARNING] set_drive is not equivalent (reference = "-rise -fall -min: <value>", implement = " -fall -min: <value>, -rise -min: <value>") for port <port-name>

For information on debugging, click [How to Debug and Fix](#).

How to Debug and Fix

The value of all the options (max, min, rise, fall) are not the same in both modes. If any of the options are not matching then all the options are being reported. Ignore the values that are matching and update the non-matching option.

Message 8

[WARNING] `set_drive` is given for port `<port-name>` only in `<mode-name>` (value)

Potential Issues

Violation Message 1 to Message 8 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 1 to Message 8.

The values of resistance set in both the modes are not same. Therefore, the wire delay of the port will not be same in both the modes.

How to Debug and Fix

For equivalence to be performed, ensure the `set_drive` constraint is specified in the SDC files for both modes.

set_max_fanout Constraints

The following messages appear when `set_max_fanout` is given for a port `<port-name>` or design `<name>` on only one of the mode `<mode-name>`.

Message 1

[WARNING] `set_max_fanout` is given for port `<port-name>` only in `<mode-name>`

For information on debugging, click [How to Debug](#).

Message 2

[WARNING] `set_max_fanout` is given for design `<name>` only in `<mode-name>`

For information on debugging, click [How to Debug](#).

How to Debug and Fix

For equivalence to be performed, ensure the `set_drive` constraint is specified in the SDC files for both modes.

Message 3

[WARNING] `set_max_fanout` is not equivalent (reference = `<value>`, implement = `<value>`) for port `<port-name>`

For information on debugging, click [How to Debug](#).

Message 4

[WARNING] `set_max_fanout` is not equivalent (reference = <value>, implement = <value>) for design <name>

Potential Issues

Violation Message 1 to Message 4 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 1 to Message 4.

If `set_max_fanout` value in two SDC files being compared is not same for a port, the port can potentially see different loads in its fan-out. This impacts the ramp time. Therefore, timing analysis would differ and multiple iterations would be required in timing closure. Alternatively, if it is incorrect, timing may not be met that can lead to chip failure.

How to Debug

The value specified in both modes are not equivalent. For equivalence, update the SDC files of the modes.

set_max_capacitance Constraints

The following messages appear when `set_max_capacitance` is given for a port <port-name> or design <name> on only one of the mode <mode-name>.

Message 1

[WARNING] `set_max_capacitance` is given for port <port-name> only in <mode-name>

For information on debugging, click [How to Debug and Fix](#).

Message 2

[WARNING] `set_max_capacitance` is given for design <name> only in <mode-name>

For information on debugging, click [How to Debug and Fix](#).

How to Debug and Fix

For equivalence to be performed, ensure the `set_max_capacitance` constraint is specified in the SDC files for both modes

Message 3

[WARNING] `set_max_capacitance` is not equivalent (reference =

<value>, implement = <value>) for port <port-name>

For information on debugging, click [How to Debug and Fix](#).

Message 4

[WARNING] set_max_capacitance is not equivalent (reference = <value>, implement = <value>) for design <name>

For information on debugging, click [How to Debug and Fix](#).

Potential Issues

Violation Message 1 to Message 4 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 1 to Message 4.

If [set_min_capacitance](#) value in two SDC files being compared is not same for a port, the port can potentially see different loads in its fan-out. This impacts the ramp time. Therefore, timing analysis would differ and multiple iterations would be required in timing closure. Alternatively, if it is incorrect, timing may not be met that can lead to chip failure.

How to Debug and Fix

The value specified in both modes are not equivalent. For equivalence, update the SDC files of the modes.

set_max_transition Constraint

The following messages appear when [set_max_transition](#) is given for a port <port-name> or design <name> with different options.

Message 1

[WARNING] set_max_transition is not equivalent (reference = <option1>, implement = <option2>) for clock <clk-name>

For information on debugging, click [How to Debug and Fix](#).

Message 2

[WARNING] set_max_transition is not equivalent (reference = <value>, implement = <value>) for design <name>

For information on debugging, click [How to Debug and Fix](#).

Message 3

[WARNING] set_max_transition is not equivalent (reference = <value>, implement = <value>) for port <port-name>

For information on debugging, click [How to Debug and Fix](#).

How to Debug and Fix

The values/options specified for the `set_max_transition` constraint in both SDC files are not the same. Update the SDC files.

Message 4

[WARNING] `set_max_transition` is given for clock <clk-name> only in <mode-name>(<option>)

Potential Issues

Violation Message 1 to Message 4 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 1 to Message 4.

If the `set_min_transition` value in two SDC files being compared is not same for a port/clock, the port/clock transition times could be different. Therefore, timing analysis would vary and multiple iterations would be required in timing closure. Alternatively, if it is incorrect, timing may not be met that can lead to chip failure.

How to Debug and Fix

The value specified in both modes are not equivalent. For equivalence, update the SDC files of the modes.

set_driving_cell Constraints

The following messages appear when `set_driving_cell` is given for a port <port-name> with different options.

Message 1

[WARNING] `set_driving_cell` is not equivalent for port <port-name> and clock (reference = <clk-name>, implement = <clk-name>)

For information on debugging, click [How to Debug and Fix](#).

Message 2

[WARNING] `set_driving_cell` is not equivalent for port <port-name> and clock (reference = <clk-name>, implement = <clk-name>) with `clock_fall`

For information on debugging, click [How to Debug and Fix](#).

Message 3

[WARNING] `set_driving_cell` is given for port `<port-name>` only in `<mode-name>`

For information on debugging, click [How to Debug and Fix](#).

Message 4

[WARNING] `set_driving_cell` is given for port `<port-name>` and clock `<clk-name>` only in `<mode1-name>`; equivalent clock in `<mode2-name>` is `<clk1-name>`

For information on debugging, click [How to Debug and Fix](#).

Message 5

[WARNING] `set_driving_cell` is given for port `<port-name>` and clock `<clk-name>` with `clock_fall` only in `<mode1-name>`; equivalent clock in `<mode2-name>` is `<clk1-name>`

For information on debugging, click [How to Debug and Fix](#).

Potential Issues

Violation Message 1 to Message 5 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 1 to Message 5.

If the `set_driving_cell` value in two SDC files being compared is not same for a port, the port is constrained to be driven by different cells in the two SDC files. This impacts the ramp time at the port. Therefore, timing analysis would differ and multiple iterations would be required in timing closure. Alternatively, if it is incorrect, timing may not be met that can lead to chip failure.

How to Debug and Fix

The options specified in both modes are not matching. The SDC file highlights both modes.

set_dont_touch Constraints

The following messages appear when `set_dont_touch` is given for a port `<port-name>` or design `<name>` on only one of the mode `<mode-name>`.

Message 1

[WARNING] set_dont_touch is given for design <name> only in implement

For information on debugging, click [How to Debug and Fix](#).

Message 2

[WARNING] set_dont_touch is given for design <name> only in reference

For information on debugging, click [How to Debug and Fix](#).

Message 3

[WARNING] set_dont_touch is given for cell <cell-name> only in implement

For information on debugging, click [How to Debug and Fix](#).

Message 4

[WARNING] set_dont_touch is given for cell <cell-name> only in reference

For information on debugging, click [How to Debug and Fix](#).

Message 5

[WARNING] set_dont_touch is given for net <net-name> only in reference

For information on debugging, click [How to Debug and Fix](#).

Message 6

[WARNING] set_dont_touch is given for net <net-name> only in implement

For information on debugging, click [How to Debug and Fix](#).

Message 7

[WARNING] set_dont_touch is given for library cell <lib-cell-name> only in reference

For information on debugging, click [How to Debug and Fix](#).

Message 8

[WARNING] set_dont_touch is given for library cell <lib-cell-name> only in implement

For information on debugging, click [How to Debug and Fix](#).

Message 9

[WARNING] `set_dont_touch` is not equivalent (reference = <value>, implement = <value>) for net <net-name>

For information on debugging, click [How to Debug and Fix](#).

Message 10

[WARNING] `set_dont_touch` is not equivalent (reference = <value>, implement = "<value>") for cell <cell-name>

For information on debugging, click [How to Debug and Fix](#).

Message 11

[WARNING] `set_dont_touch` is not equivalent (reference = <value>, implement = <value>) for design <name>

For information on debugging, click [How to Debug and Fix](#).

Message 12

[WARNING] `set_dont_touch` is not equivalent (reference = <value>, implement = <value>) for library cell <lib-cell-name>

For information on debugging, click [How to Debug and Fix](#).

Potential Issues

Violation Message 1 to Message 12 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 1 to Message 12.

If the [set_dont_touch](#) constraint is not set in any of the modes, the node might be optimized away by synthesis tools. Therefore, the timing analysis results will not be same.

How to Debug and Fix

The [set_dont_touch](#) constraint is not specified in both modes. Two [set_dont_touch](#) constraints are considered equivalent only if:

- They are applied on the same/equivalent object list
- Each object contains the same value

set_propagated_clock Constraints

The following messages appear when *set_propagated_clock* is given for a clock *<clk-name>* or port *<port-name>* on only one of the mode *<mode-name>*.

Message 1

[WARNING] set_propagated_clock is given for clock *<clk-name>* only in *<mode-name>*

For information on debugging, click [How to Debug and Fix](#).

Message 2

[WARNING] set_propagated_clock is given for port *<port-name>* only in *<mode-name>*

For information on debugging, click [How to Debug and Fix](#).

Message 3

[WARNING] set_propagated_clock is given for pin *<pin-name>* only in *<mode-name>*

For information on debugging, click [How to Debug and Fix](#).

Potential Issues

Violation Message 1 to Message 3 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 1 to Message 3.

If one mode has a propagated clock and the other mode does not, the timing analysis in one mode occurs with actual delays of the circuit while other mode uses estimated values. Therefore, results would vary and multiple iterations would be required in timing closure. Alternatively, if it is incorrect, timing may not be met that can lead to chip failure.

How to Debug and Fix

The *set_propagated_clock* constraint is not specified in both modes. Two *set_propagated_clock* constraints are considered equivalent only if they are applied on the same object list. Update the SDC files for both modes.

set_ideal_network Constraints

The following messages appear when *set_ideal_network* is given for a port *<port-name>* or terminal *<term-name>* on only one of the mode

<mode-name>.

Message 1

[WARNING] `set_ideal_network` is given for port <port-name> only in implement

For information on debugging, click [How to Debug and Fix](#).

Message 2

[WARNING] `set_ideal_network` is given for port <port-name> only in reference

For information on debugging, click [How to Debug and Fix](#).

Message 3

[WARNING] `set_ideal_network` is given for terminal <term-name> only in implement

For information on debugging, click [How to Debug and Fix](#).

Message 4

[WARNING] `set_ideal_network` is given for terminal <term-name> only in reference

For information on debugging, click [How to Debug and Fix](#).

Potential Issues

Violation Message 1 to Message 4 explicitly state potential issues.

Consequences of Not Fixing

These consequences are for Message 1 to Message 4.

If the [set_ideal_network](#) is set in any of the modes, the timing update of the cells and nets in the fan-out of that object gets disabled. Therefore, the timing analysis results will vary.

How to Debug and Fix

The [set_ideal_network](#) is not specified in both modes. Two [set_ideal_network](#) constraints are considered equivalent only if:

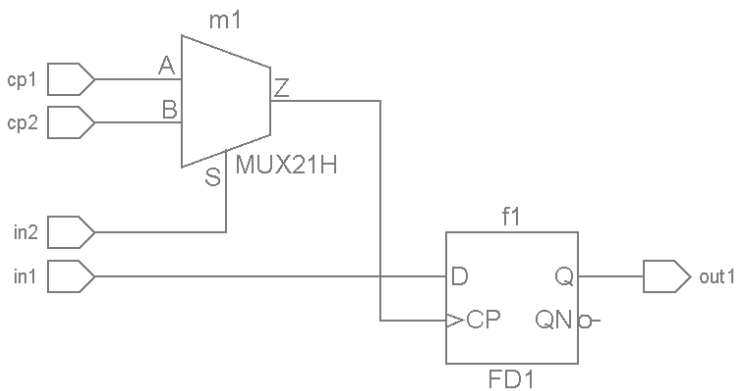
- They are applied on the same object list
- Each object contains the same option

Example Code and/or Schematic

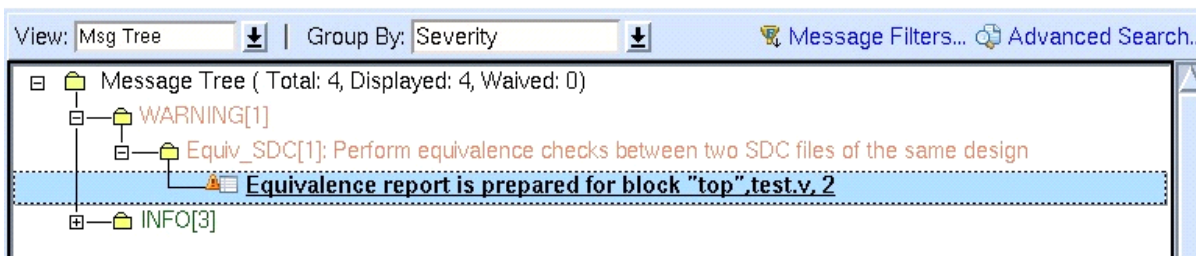
Example 1

[View Test Case Files](#)

This example shows how to open the Equivalence report. The schematic of the top module is as follows:



Double-click the violation message to view the Equivalence Report.



Other Examples

Test Case Files Not Available

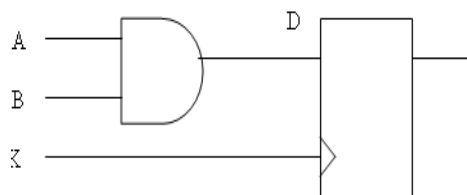
Use this section to understand equivalence checking between:

- [set_case_analysis Constraints Examples](#)
- [Clock Constraints Examples](#)
- [Input/Output Delay Constraints Examples](#)
- [Input Transition, Clock Transition, Load, Clock Latency, and Clock Uncertainty Constraints](#)
- [Timing Exception Constraints](#)
- [set_disable_timing equivalence](#)

set_case_analysis Constraints Examples

The equivalence analysis is reported in the [Equivalence Report](#).

Example 1



Let the constraints for two files, File1 and File2 be defined as follows:

File1:

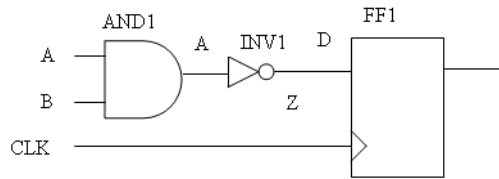
```
set_case_analysis 0 [get_ports {A}]
```

File2:

```
set_case_analysis 0 [get_ports {B}]
```

The function at the pin D reduces to 0 for both the constraints. Therefore, the constraints are equivalent.

Example 2



Let the constraints for two files, File1 and File2 be defined as follows:

File1:

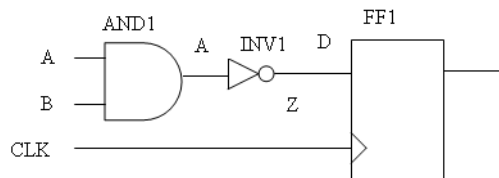
```
set_case_analysis 0 [get_ports {A}]
```

File2:

```
set_case_analysis 1 [get_pins {INV1/Z}]
```

In this example, the function at pin D of the flip-flop FF1 reduces to 1 for both the constraints. Therefore, they are equivalent.

Example 3



Let the constraints for two files, File1 and File2 be defined as follows:

File1:

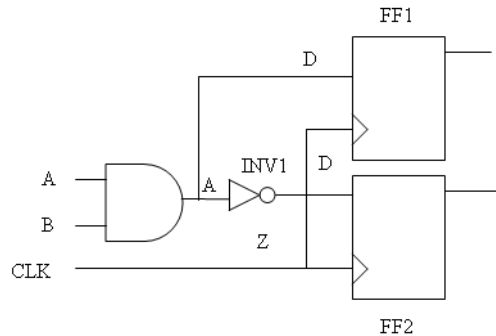
```
set_case_analysis 0 [get_ports {A}]
```

File2:

```
set_case_analysis 0 [get_pins {INV1/Z}]
```

In this example, the function at pin D of the flip-flop FF1 reduces 1 for the constraint in File1, but reduces to 0 for File2. Therefore, they are not equivalent.

Example 4



Let the constraints for two files, File1 and File2 be defined as follows:

File1:

```
set_case_analysis 0 [get_ports {A}]
```

File2:

```
set_case_analysis 1 [get_pins {INV1/Z}]
```

In this example, the function at pin D of instance FF2 reduces 1 for both the constraints. Therefore, the constraints are equivalent with respect to the flip-flop FF2.

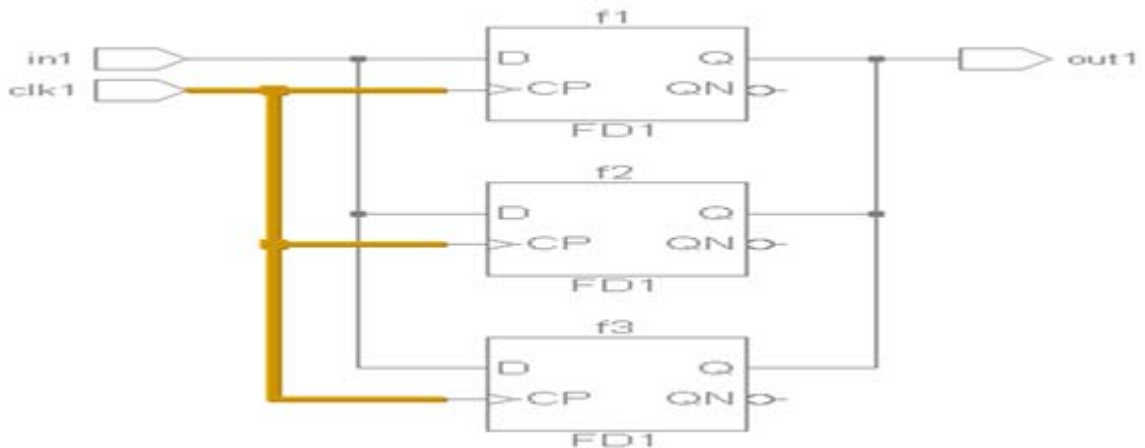
However, the function at pin D of the instance FF1 is 0 for first case while it is $D = A.B$ for the second case. Therefore, the constraints are not equivalent with respect to the flip-flop FF1. In this case, the constraint for File1 is missing in File2.

Clock Constraints Examples

The equivalence analysis is reported in the [Equivalence Report](#).

Example 1: One-to-Many Clock Mapping

This example illustrates one-to-many clock mapping. Consider the following design.



The SDC file for reference and implement modes are:

Reference.sdc

```
create_clock -name r1 -period 10 [get_ports {clk1}]
```

Implement.sdc

```
create_clock -name i1 -period 10 [get_pins {f1/CP}]
```

```
create_clock -name i2 -period 10 [get_pins {f2/CP}]
```

```
create_clock -name i3 -period 10 [get_pins {f3/CP}]
```

As per the SDC specification, clock r1, in reference mode, is constraining three clock pins, which are constrained by three clocks i1, i2, i3 collectively in implement mode. Therefore, r1 is equivalent to i1, i2, and i3 all taken together. This one-to-many clock relationship is depicted in the following Equivalence Report.

The screenshot shows a 'Spreadsheet Viewer' window displaying a table with the following structure:

	B	C	D	E	F	G
	Equivalent	Analysis	Clocks in reference mode	Clocks in implement mode	reference Design Object	implement Design Object
1	Yes	Clocks are Equivalent	r1:default_waveform:positive	i1:default_waveform:positive i2:default_waveform:positive i3:default_waveform:positive	f1/CP,i2/CP,i3/CP	i1/CP,i2/CP,i3/CP
2	Yes	implement clock is equivalent to reference clock but differ in name and the object on which it is defined.	r1:default_waveform:positive	i1:default_waveform:positive	f1/CP	i1/CP
3	Yes	implement clock is equivalent to reference clock but differ in name and the object on which it is defined.	r1:default_waveform:positive	i2:default_waveform:positive	f2/CP	i2/CP
4	Yes	implement clock is equivalent to reference clock but differ in name and the object on which it is defined.	r1:default_waveform:positive	i3:default_waveform:positive	f3/CP	i3/CP

Callouts in the image provide the following information:

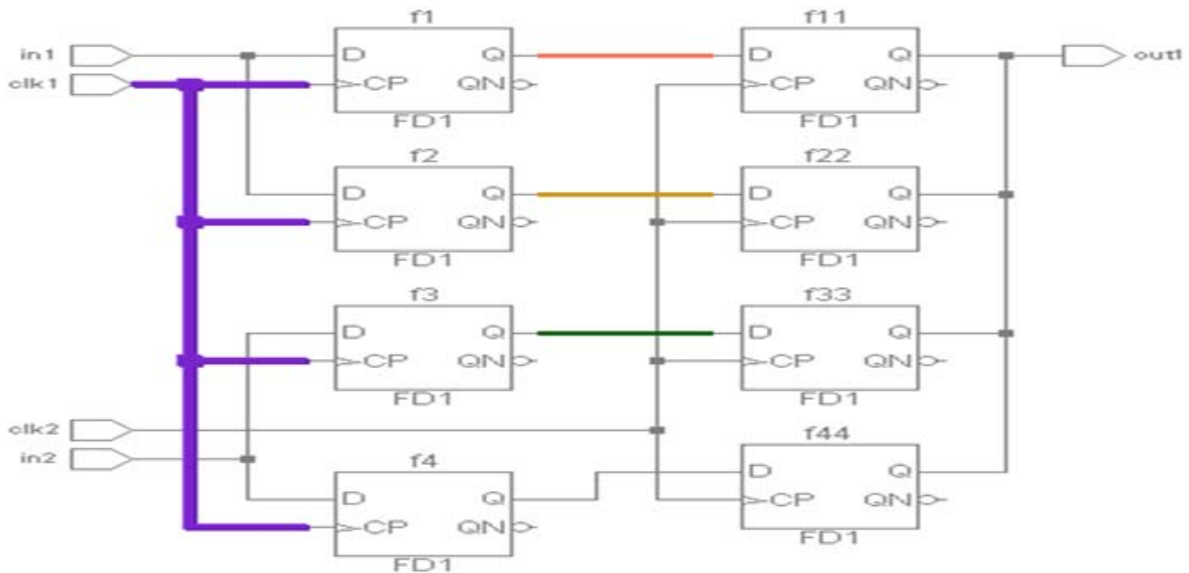
- "Click to collapse." (pointing to the collapse icon in the top-left corner)
- "Shows one to many relationship." (pointing to the first row of data)
- "Each row displays the clock Equivalence for each pair." (pointing to the 'Equivalent' column)
- "See the individual relationship." (pointing to the 'Analysis' column)
- "Shows the sample flops for each of the clocks" (pointing to the 'reference Design Object' and 'implement Design Object' columns)

Messages: Grouped: 1 Displayed: 4 Total: 4

Example 2: Many-to-Many Clock Mapping

This example illustrates many-to-many clock mapping. Consider the following design.

Constraints Management Rules



The SDC file for reference and implement modes are:

Reference.sdc

```
create_clock -name rstart1 -period 10 [get_pins {f1/CP f2/CP}]
create_clock -name rstart2 -period 10 [get_pins {f3/CP f4/CP}]
create_clock -name rend1 -period 10 [get_ports {clk2}]
set_false_path -from rstart1 -to rend1
```

Implement.sdc

```
create_clock -name istart1 -period 10 [get_pins {f1/CP}]
create_clock -name istart2 -period 10 [get_pins {f2/CP f3/CP}]
create_clock -name istart3 -period 10 [get_pins {f4/CP}]
create_clock -name iend1 -period 10 [get_pins {f11/CP f22/CP}]
create_clock -name iend2 -period 10 [get_pins {f33/CP f44/CP}]
```

```
set_false_path -from {istart1 istart2} -to {iend1 iend2}
```

This many-to-many clock relationship is depicted in the following Equivalence Report.

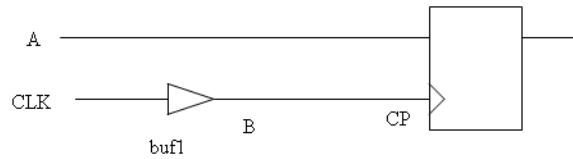
	B	C	D	E	F	G
	Equivalent	Analysis	Clocks in reference mode	Clocks in Implement mode	reference Design Object	implement Design Object
B1	Yes	Clocks are Equivalent	rstart1:default_waveform:positive rstart2:default_waveform:positive	istart1:default_waveform:positive istart2:default_waveform:positive istart3:default_waveform:positive	f1/CP, f2/CP, f3/CP, f4/CP	f1/CP, f2/CP, f3/CP, f4/CP
2	Yes	implement clock is equivalent to reference clock but differ in name and the object on which it is defined.	rstart1:default_waveform:positive	istart1:default_waveform:positive	f1/CP	f1/CP
3	Yes	implement clock is equivalent to reference clock but differ in name and the object on which it is defined.	rstart2:default_waveform:positive	istart3:default_waveform:positive	f4/CP	f4/CP
4	Yes	Equivalent Clocks.	rstart1:default_waveform:positive	istart2:default_waveform:positive	f2/CP	f2/CP
5	Yes	Equivalent Clocks.	rstart2:default_waveform:positive	istart2:default_waveform:positive	f3/CP	f3/CP
B6	Yes	Clocks are Equivalent	rend1:default_waveform:positive	lend1:default_waveform:positive lend2:default_waveform:positive	f11/CP, f33/CP	f11/CP, f33/CP

Example 3

In this example, the *create_clock* constraints are equivalent because all the end points in the design are driven by clocks with the same characteristics (period=10).

The design is as follows:

Constraints Management Rules



Let the constraints for two files, File1 and File2 be defined as follows:

File1:

```
create_clock -name CLK_A1 -period 10 [get_ports {CLK}]
```

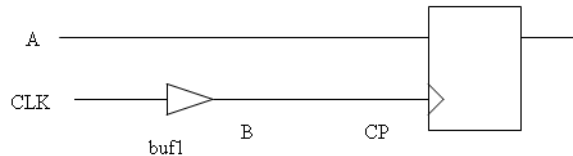
File2:

```
create_clock -name CLK_A2 -period 10 [get_pins {buf1/B}]
```

Example 4

In this example, the *create_clock* constraints are not equivalent because all the end points in the design are driven by clocks with different characteristics (periods 5 and 10).

The design is as follows:



Let the constraints for two files, File1 and File2 be defined as follows:

File1:

```
create_clock -name CLK_A1 -period 10 [get_ports {CLK}]
```

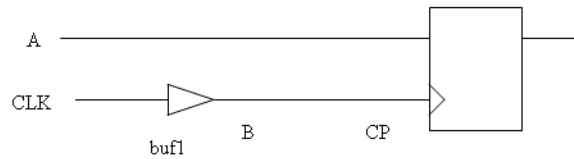
File2:

```
create_clock -name CLK_A2 -period 5 [get_ports {CLK}]
```

Example 5

In this example, the *create_clock* constraints are equivalent because the end points of the design are driven with the same characteristics (period=10).

The design is as follows:



Let the constraints for two files, File1 and File2 be defined as follows:

File1:

```
create_clock -name CLK_A1 -period 10 [get_ports {CLK}]
```

File2:

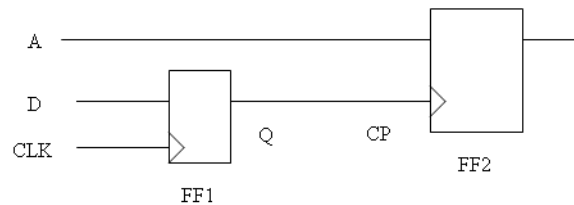
```
create_clock -name CLK_A2 -period 5 [get_ports {CLK}]
```

```
create_generated_clock -name CLK_B2 -divide_by 2 -source
[get_ports {CLK}] [get_pins {buf1/B}]
```

Example 6

In this example, the two clock sets are equivalent as the end points in the design (FF1 and FF2) are driven by clocks with the same characteristics (period of 10 and 20).

The design is as follows:



Let the constraints for two files, File1 and File2 be defined as follows:

File1:

```
create_clock -name CLK_A1 -period 10 [get_ports {CLK}]
```

```
create_clock -name CLK_B1 -period 20 [get_pins {FF1/Q}]
```

File2:

```
create_clock -name CLK_A2 -period 10 [get_ports {CLK}]
```

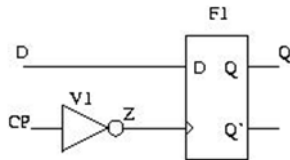
```
create_generated_clock -name CLK_B2 -divide_by 2 -source
```

```
[get_ports {CLK}] [get_pins {FF1/Q}]
```

Example 7

In this example, the *create_clock* constraints are equivalent because the polarity of the clocks at the end point is the same.

The design is as follows:



Let the constraints for two files, File1 and File2 be defined as follows:

File1:

```
create_clock -name CLK1 -period 10 -waveform {0 5} CP
```

File2:

```
create_clock -name CLK2 -period 10 -waveform {5 10} V1/Z
```

Example 8

Let the constraints for two files, File1 and File2 be defined as follows:

File1:

```
create_clock -name ACLK1 -period 10 -waveform {0 5} CP
```

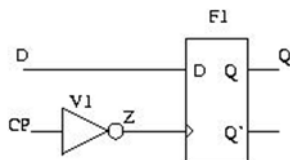
```
create_clock -name ACLK2 -period 10 -waveform {0 5} CP -add
```

File2:

```
create_clock -name BCLK1 -period 10 -waveform {0 5} CP
```

```
create_clock -name BCLK2 -period 10 -waveform {0 5} CP -add
```

The design is as follows:



In this example, all clocks have the same characteristics. However, there is ambiguity as to which clocks are equivalent. To resolve this ambiguity, specify the value of the *equiv_sdc_ambiguous_clock_file* parameter as *amb_clock_report* in the command line.

Here, *amb_clock_report* is a file with the following entry:

```
ACLK1 BCLK1
```

This will resolve the ambiguity and ACLK1 will be equivalent to BCLK1 and ACLK2 will be equivalent to BCLK2.

Example 9

Let the constraints for two files, File1 and File2 be defined as follows:

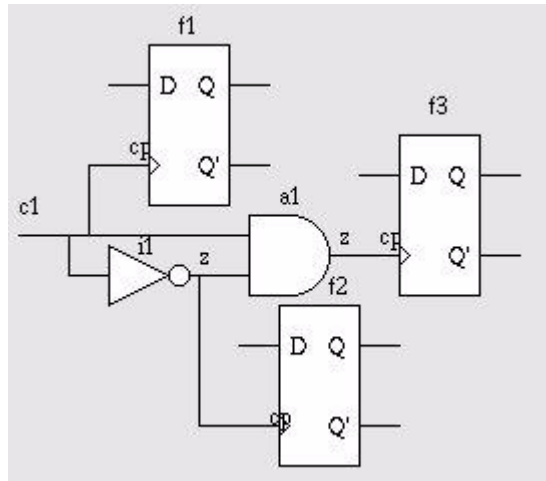
File1:

```
create_clock -name clk -period 10 -waveform {0 5} c1
set_clock_sense -negative f1/cp
set_clock_sense -positive i1/z
set_clock_sense -pulse rise_triggered_high_pulse a1/z
```

File2:

```
create_clock -name clk -period 10 -waveform {0 5} c1
create_generated_clock -name gclk -source c1 -master_clock
clk -edges {1 1 3} a1/z
```

The design is as follows:



The [set_clock_sense](#) constraints at f1/cp and i1/z are invalid because a negative waveform is not propagating to f1/cp and a positive waveform is not propagating to i1/z. Therefore, these constraints are ignored and a positive waveform propagates to f1/cp and negative propagates to f2/cp.

The [set_clock_sense](#) constraint at a1/z contains the pulse option. It is beyond the scope of the rule to validate 'pulse' option's validity. Therefore, it is assumed correct and the [rise_triggered_high_pulse](#) waveform of clk propagates to f3/cp.

The two SDC files are equivalent since equivalent clock waveforms reach at the clock pins of each flip-flop. At f1/cp, positive waveforms of clock clk match. At f2/cp, negative waveforms of clock clk match. At f3/cp, the [rise_triggered_high_pulse](#) waveform of clk matches with gclk default waveform.

Input/Output Delay Constraints Examples

The equivalence analysis is reported in the [Equivalence Report](#).

Example 1

Let the constraints for two files, File1 and File2 be defined as follows:

File1:

```
create_clock -name CLK -period 10 -waveform {0 5}
set_input_delay 2 -clock CLK in1
```

File2:

```
create_clock -name CLK -period 10 -waveform {0 5}  
set_input_delay 3 -clock CLK in1
```

In this example, the *set_input_delay* constraints are not equivalent because the delay values are different.

Example 2

Let the constraints for two files, File1 and File2 be defined as follows:

File1:

```
create_clock -name VCLK1 -period 10 -waveform {0 5}  
create_clock -name VCLK2 -period 10 -waveform {0 5}  
set_input_delay 2 -clock VCLK1 in1  
set_input_delay 2 -clock VCLK2 in1 -add_delay
```

File2:

```
create_clock -name VCLK -period 10 -waveform {0 5}  
set_input_delay 2 -clock VCLK in1
```

In this example, the input delays set on the port *in1* are not equivalent because the virtual clock *VCLK* cannot be equivalent to more than one clock.

Example 3

Let the constraints for two files, File1 and File2 be defined as follows:

File1:

```
create_clock -name CLK1 -period 10 -waveform {0 5} clk1  
create_clock -name VCLK -period 10 -waveform {0 5}  
set_input_delay 2 -clock VCLK in1  
set_input_delay 2 -clock CLK1 in1 -add_delay
```

File2:

```
create_clock -name CLK -period 10 -waveform {0 5} clk1  
set_input_delay 2 -clock CLK in1
```

In this example, the input delays set on the port *in1* are not equivalent because there is no equivalent input delay specification for *VCLK* in *in2*.

Input Transition, Clock Transition, Load, Clock Latency, and Clock

Uncertainty Constraints

The equivalence analysis is reported in the [Equivalence Report](#).

Let the constraints for two files, File1 and File2 be defined as follows:

File1:

```
create_clock -name CLK1 -period 10 -waveform {0 5} clk1
create_clock -name CLK2 -period 10 -waveform {0 5} clk2
set_load 2 in1
set_clock_transition 3 CLK1
set_input_transition 2 clk1
set_clock_latency 3 -source CLK2
set_clock_uncertainty 3 -from CLK1 -to CLK2
```

File2:

```
create_clock -name CLK1 -period 10 -waveform {0 5} clk1
create_clock -name CLK2 -period 10 -waveform {0 5} clk2
set_load 3 in1
set_clock_transition 3 CLK1 -rise
set_input_transition 2 clk1
set_clock_uncertainty 4 -from CLK1 -to CLK2
```

In this example:

- [set_load](#) is not equivalent since the load values are different
- [set_clock_transition](#) is not equivalent because the `-rise` option is set in File 2 only
- [set_input_transition](#) is equivalent
- [set_clock_latency](#) are not equivalent since it is set only in File 1
- [set_clock_uncertainty](#) is not equivalent since the values are different

Timing Exception Constraints

The equivalence analysis is reported in the [Equivalence Report](#).

Example 1

Let the two SDC files reference and implement contain the following:

```
set_false_path -through AND1/Z (reference)
set_false_path -from A -to B (implement)
```

If only the timing paths that exist between A and B pass through AND1/Z, the timing exception is considered as equivalent.

The *Equiv_SDC* rule considers clock domains if there are clocks involved in the timing exception command and they reduce the timing path.

Example 2

Let the two SDC files reference and implement contain the following:

```
set_false_path -from A -to B (reference)
```

```
set_false_path -from CLK1 -to CLK2 (implement)
```

Consider that the design contains two flip-flops: A and B. If the flip-flop A is driven by the clock CLK1 and flip-flop B is driven by the clock CLK2, the timing exception commands are considered equivalent. However, if flip-flop A is driven by both clocks and flip-flop B is driven only by the clock CLK2, the following are equivalent:

```
set_false_path -from A -to B (reference)
```

```
set_false_path -from {CLK1 CLK2} -to CLK2 (implement)
```

or

```
set_false_path -from CLK1 -to CLK2 (reference)
```

```
set_false_path -from CLK1 -to CLK2 (implement)
```

If in an SDC file the [set_case_analysis](#) command is also given along with a timing exception command, the *Equiv_SDC* rule gives priority to the [set_case_analysis](#) constraint and ignores the timing exception command.

Example 3

Let the two SDC files reference and implement contain the following:

```
set_false_path -from A -to B (reference)
```

```
set_multicycle_path -from A -to B (reference)
```

```
set_false_path -from A -to B (implement)
```

In the above example, the timing path between A and B constrained with the [set_multicycle_path](#) constraint is ignored because the [set_false_path](#) constraint gets priority over the [set_multicycle_path](#) constraint.

Example 4

For the following specification, the *Equiv_SDC* rule reports a violation message because the timing exception is applied on the path which is not specified with clocks:

```
set_false_path -rise_from in1 -to o1 //where in1 & o1 are
input output ports
```

For the following specifications, the *Equiv_SDC* rule does not report a violation message because the timing exception is not applied on the path which is specified with clocks:

```
set_false_path -rise_from [get_clocks clk1] -to [get_clocks clk2]
set_false_path -rise_from [get_clocks clk1] -to [get_ports
o1] // clk1 is clock & o1 is output port
```

Example 5

Suppose you are performing an equivalence check between the following:

```
reference.sdc
```

```
set_false_path -from A -to [list B C]
```

```
Implement.sdc
```

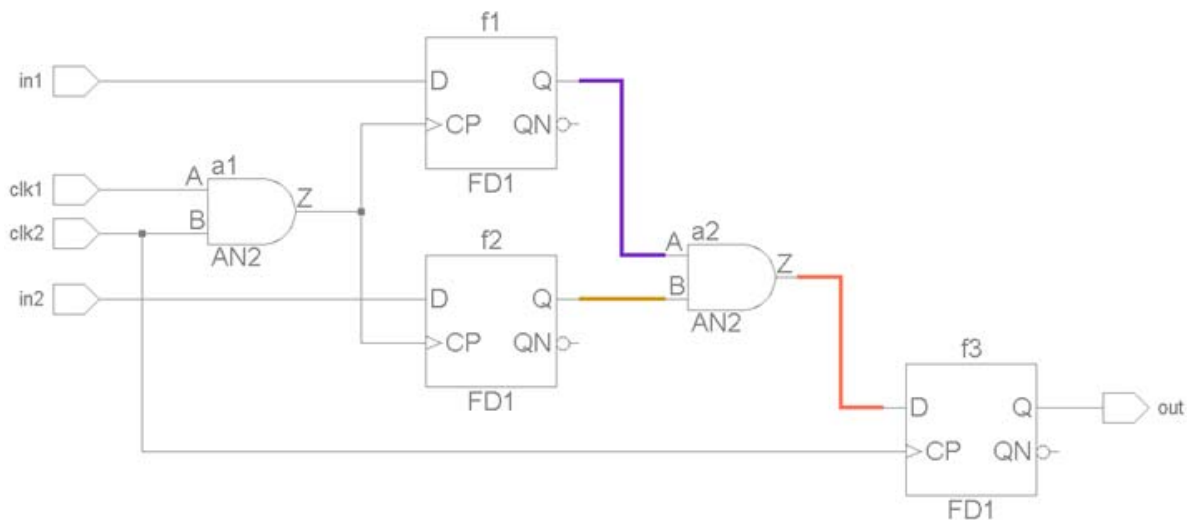
```
set_false_path -from A -to B
```

The path A to C mentioned in reference.sdc is not stated in the implement.sdc file. However, the path A to B is stated in both files. Therefore, the constraint mentioned in reference.sdc is are not equivalent to the constraint mentioned in implement.sdc.

The *Equiv_SDC* rule reports the results in the [Equivalence Report](#).

set_disable_timing equivalence

This example describes the equivalence check for [set_disable_timing](#). Consider the following design:

**Reference SDC:**

```
create_clock -name rC1 -period 10 [get_ports {clk1}]
create_clock -name rC2 -period 10 [get_ports {clk2}]
# The following constraint would disable timing-paths f1/CP-f3/D
# and f2/CP-f3/D
set_disable_timing a2/Z
```

Implement SDC:

```
create_clock -name iC1 -period 10 [get_ports {clk1}]
create_clock -name iC2 -period 10 [get_ports {clk2}]
# The following constraint would disable/falsify timing-paths
# f1/CP-f3/D
set_false_path -from f1/CP -to f3/D
# The following constraint would disable/falsify timing-paths
# f2/CP-f3/D
set_disable_timing a2/B
```

Constraints Management Rules

```
# The following constraint would disable clock path from clk1  
(iCl)
```

```
set_disable_timing -from A -to Z [ get_cells { a1 } ]
```

Both the modes are equivalent. In addition, the following specification in the Implement mode would not be reported because it is not a timing path, but is a clock path:

```
set_disable_timing -from A -to Z [ get_cells { a1 } ]
```

The *Equiv_SDC* rule reports the results in the *Equivalence Report*.

Default Severity Label

Warning

Rule Group

const_mgmt

Report and Related Files

- *Equivalence Report*
- *Clock-Mapping Report*

Equiv_SDC_Auto_Detect

Perform auto detection of equivalent ports and sequential elements between two designs

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

The *Equiv_SDC_Auto_Detect* rule performs auto detection of equivalent ports and sequential elements between two designs and generates a mapping file called *auto_equiv.txt*.

The *Equiv_SDC_Auto_Detect* rule automatically extracts the information of equivalent design objects. The rule also honors the specified information (using the *equiv_sdc_design_equivalence_file* parameter). In case, you have not specified the equivalence for some of the design objects, the rule infers equivalence for the missing design objects, besides honoring the equivalence of the design objects given in the equivalence file (mapping file).

In addition, as the rule reads the equivalence file, it examines the *equiv_sdc_design_ignore_prefix* parameter and appends the same to the pairs, it finds, automatically.

You can also provide hierarchical vs hierarchical mapping information through the *equiv_sdc_design_equivalence_hier_file* parameter. This rule automatically detects equivalent points and accordingly generates the mapping file. See *Detecting of Equivalent Design Objects*.

For more information, read the following topics:

- [Prerequisites](#)
- [Detecting of Equivalent Design Objects](#)
- [Parsing of the Equivalence File](#)
- [Enabling the One-Pass Flow](#)
- [Rule Exceptions](#)

Prerequisites

The rule runs only if the *sdc_data* constraint is specified for both designs.

Detecting of Equivalent Design Objects

The rule detects the equivalent design objects on the basis of the names of the design objects. It considers a design object of a reference design equivalent to a design object of an implement design, if the hierarchical names of the design objects are the same and the objects are referred in the SDC commands.

If you have provided hierarchical vs hierarchical mapping information through the `equiv_sdc_design_equivalence_hier_file` parameter, this rule automatically detects equivalent points and accordingly generates the mapping file.

For mapping, if the name of the object is different between implement and reference, this rule outputs these objects in a separate section in the `auto_equiv.txt` file as follows.

```
List of design objects in reference design with no
counterpart in implement design
```

```
-----
#Ref B1Pin Name(last) r:top/U_modA/B1/Z
```

```
List of design objects in implement design with no
counterpart in reference design
```

```
-----
#Impl B1Pin Name(last) i:top/B1/Z
```

If you provide the hierarchical mapping, such as:

```
r:top/U_modA/B1
```

```
i:top/B1
```

This rule outputs these objects in the same section:

```
List of Equivalent objects found on the basis of names
```

```
-----
#Ref B1Pin Name(last) r:top/U_modA/B1/Z
```

```
#Impl B1Pin Name(last) i:top/B1/Z
```

For an RTL design, the hierarchical name of inferred sequential cell have `_reg` extension appended to the register names. The rule attempts to identify the equivalence for the top-level ports, the sequential elements and other design objects that are referred to in the SDC commands. It also reports the design objects for which, it cannot infer an equivalence object on the basis of hierarchical names.

Only the following design objects are considered as valid equivalence design objects:

- Top level port
- Sequential cell or its pins
- Design object referred in the SDC constraints
- Lib cell/RTL primitive/Black box/Grey box or its pins

Parsing of the Equivalence File

The rule reads each line of the equivalence file and looks for `r:` and `i:` in the line. If any of the phrases is not found, it ignores the line. Otherwise, the part of line after each phrase is read and treated as the hierarchical name of the respective object.

Enabling the One-Pass Flow

In the one-pass flow, the *Equiv_SDC_Auto_Detect* and the *Equiv_SDC_Dual_Design* rules are run together. The mapping file generated by the *Equiv_SDC_Auto_Detect* is internally passed to the *Equiv_SDC_Dual_Design* rule.

To enable the one-pass flow, set the *equiv_sdc_enable_one_pass* to `yes`.

You can specify a script file for modifying the mapping file, which was generated by the *Equiv_SDC_Auto_Detect* rule. To apply the script file, set the *equiv_sdc_script_file* parameter to the script file path.

If the script file requires a side file, specify the side file by using the *equiv_sdc_script_side_file* parameter.

Rule Exceptions

The capability of automatic detection of the equivalent design objects will

give good results if it is known that the reference and the implement designs have many design objects of the same type with the same name. Two design objects are of the same type, if both are either top-level ports or sequential elements or if both are neither top-level ports nor sequential elements.

If the design object is a hierarchical pin, which is not constrained or a hierarchical instance of a lib cell/RTL primitive/Black box/Grey box, the object is considered invalid equivalence design objects. For design objects whose equivalent counterparts cannot be inferred, you need to review and update the information in the generated equivalence file. Such records are prefixed with a '#', you need to remove the '#' and update the information. However, if the generated file is fed without review to the [Equiv_SDC_Dual_Design](#) rule, the commented design objects (prefixed with a #) are ignored and not read from the equivalence file.

Parameter(s)

- [equiv_sdc_design_equivalence_file](#): Default value is none. Set the value to the design equivalence file that provides a report to map the ports, registers, and any intermediate design object used in the SDC files of one design to another. Do not use this parameter with the [equiv_sdc_design_equivalence_hier_file](#) parameter.
- [equiv_sdc_design_equivalence_hier_file](#): Default is none. Set this parameter to the path of the hierarchical mapping file. Do not use this parameter with the [equiv_sdc_design_equivalence_file](#) parameter.
- [equiv_sdc_design_ignore_prefix](#): Default is none. Set the value to the prefix of the design hierarchy in the equivalent design file that you want to ignore.
- [equiv_sdc_enable_one_pass](#): Default is no. Set this parameter to yes enable the one-pass flow for dual design equivalence.
- [equiv_sdc_script_file](#): Default is none. Set this parameter to the file path of the mapping script file for the one-pass flow.
- [equiv_sdc_script_side_file](#): Default is none. Set this parameter to the file path of the side file used in the one-pass flow.

Constraint(s)

None

Messages and Suggested Fix

Message 1

The following message indicates the generation of an equivalence file between the reference and implement designs:

[INFO] Equivalent design objects between designs "<ref-des>" (reference) and "<imp-des>" (implement) is output

Potential Issues

Not applicable

Consequences of Not Fixing

Not applicable

How to Debug and Fix

View the [auto_equiv.txt](#) and the [Equivalence Report](#).

See [Example 1](#) for more information.

Message 2

The SDC Commands are read by the rule through [sdc_data](#) SGDC constraints. When the mode of the [sdc_data](#) constraint is specified as `reference/implement`, the reference/implement SDC constraints are identified, respectively. If it is not specified for any of the designs, following FATAL violation is shown, and the rule exits:

[FATAL] Rule runs only if `sdc_data` constraint for both designs are given (missing for <des-name-list> design(s))

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

For Message 2, SpyGlass will exit without performing any further checks.

How to Debug and Fix

The rule expects reference and implement as `-mode` with two different [sdc_data](#) constraints. Therefore, review the SGDC file. If not specified, update the SGDC file and run the rule again.

Message 3

The following message appears when mapping errors are found in the

mapping file:

[WARNING] ERROR(s) were found in the mapping file, please check <mapping-file> for details.

Potential Issues

The message appears due to redundant mappings and invalid design object names.

Consequences of Not Fixing

Mappings that have errors are ignored.

How to Debug and Fix

Double-click the message to view the spreadsheet, which describes the errors with the mapping file. The spreadsheet rows are linked to the mapping file. Click a row to open the mapping file. The erroneous line in the mapping file is highlighted.

Remove the redundant mappings and the invalid design objects.

See [Example 2](#) for more information.

Example Code and/or Schematic

Example 1

For the following snippet, the *Equiv_SDC_Auto_Detect* rule reports a violation because the mode argument is not specified in the ref.sgcd file.

```
ref.sgcd
current_design top
sdc_data -file ref.sdc
```

```
imp.sgcd
current_design top1
sdc_data -file imp.sdc -mode implement
```

Example 2

In this example, there is an error in the mapping file. Consequently, the *Equiv_SDC_Auto_Detect* rule reports [Message 3](#).

Assume that the reference design does not contain any object called

top/wrongDesObj1 and the implement design does not contain any object called top/wrongDesObj2.

The equivalence file given is below.

```

1 r:top/in1
2 r:top/in1
3 i:top/out_reg
4
5 r:top/out_reg
6 i:top/out_reg
7 i:top/out_reg
8
9 r:top/wrongDesObj1
10 i:top/wrongDesObj2
11

```

The equivalence file contains a redundant reference mapping (at line 2), a redundant implement mapping (at line 7) and 2 invalid design objects at line 9 and 10.

After running the rule, the following message appears.



Double-click the violation message to view the spreadsheet, which describes the errors with the mapping file.

	B	C	D	E
	Line No	Reference Hierarchy	Implement Hierarchy	Error Message
1	2	r:top/in1	-	Redundant reference mapping.
2	7	-	t:top/out_reg	Redundant implement mapping.
3	9	r:top/wrongDesObj1	-	Reference design object not found in the design.
4	10	-	t:top/wrongDesObj2	Implement design object not found in the design.

Default Severity Label

Info/Fatal

Rule Group

const_mgmt

Report and Related Files

- [Equivalence Report](#)
- auto_equiv.txt

The rule reports equivalent design objects in the same format as the equivalence file in a file named, auto_equiv.txt, which is located at:

spyglass_spysch/const_mgmt/auto_equiv.txt

This file is saved as a tar.gz file if the file size is greater than 1 Gb.

The format of the generated file is kept the same as the equivalence file so that this file can be passed as an argument to the [equiv_sdc_design_equivalence_file](#) parameter for the consumption by [Equiv_SDC_Dual_Design](#) rule. The report contains the following 4 sections:

- a. Equivalent objects inherited from the [equiv_sdc_design_equivalence_file](#) parameter.
- a. Equivalent objects inferred based on hierarchical name matching
- a. Design objects in reference design not having equivalent counterpart in the implement design.
- a. Design objects in implement design not having equivalent counterpart in reference design.

A sample of auto_equiv.txt file is as follows:

```

1 List of Equivalent objects inherited from equivalence file
2 -----
3
4 Empty Section
5
6 List of Equivalent objects found on the basis of names
7 -----
8
9 Ref Port Name(last) r:top/in1
10 Impl Port Name(last) i:top2/in1
11
12 Ref Port Name(last) r:top/clk1
13 Impl Port Name(last) i:top2/clk1
14
15 Ref Port Name(last) r:top/clk2
16 Impl Port Name(last) i:top2/clk2
17
18 Ref Port Name(last) r:top/sel
19 Impl Port Name(last) i:top2/sel
20
21 Ref Port Name(last) r:top/out1
22 Impl Port Name(last) i:top2/out1
23
24 Ref DFF Name(last) r:top/b11/b21/b32/f1
25 Impl DFF Name(last) i:top2/b11/b21/f1
26
27 Ref DFF Name(last) r:top/b11/f1
28 Impl DFF Name(last) i:top2/b11/f1
29
30 Ref DFF Name(last) r:top/f1
31 Impl DFF Name(last) i:top2/f1
32
33
34 List of design objects in reference design with no counterpart in implement design
35 -----
36
37 Empty Section
38
39 List of design objects in implement design with no counterpart in reference design
40 -----
41
42 Empty Section

```

A format in which design object are reported in the file is as follows:

Reference object

Ref <object type> Name(Last) r:<obj-hier-name>

Implement object

Imp <object type> Name(Last) i:<obj-hier-name>

where, <object type> is the type of design object that may be

any of the following: Port: Port , Terminal: B1Pin, or Instance: DFF.

In case, any section of the report is empty, it is shown with an Empty Section message for clarity.

Equiv_SDC_Block

Performs equivalence between the top-level and block-level design units

When to Use

To determine whether SDC files that are used for block- and top-level designs are consistent and complete with respect to each other. This rule is applicable to the RTL, Pre-layout, Post-layout phases.

Description

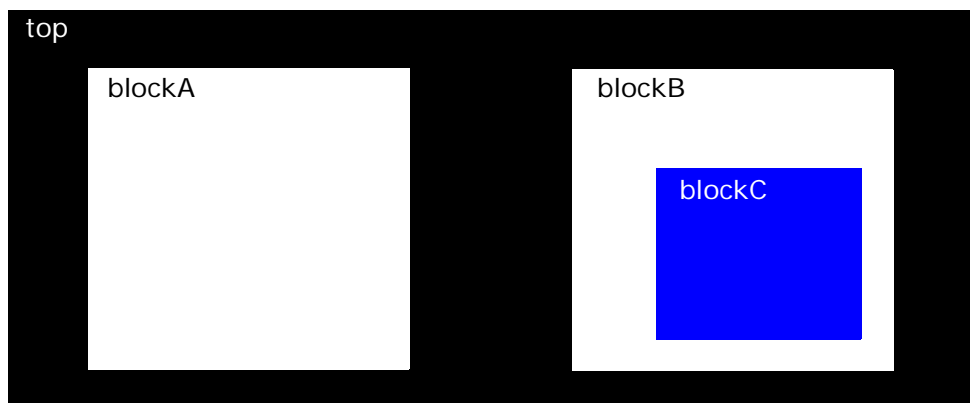
The *Equiv_SDC_Block* rule performs an equivalence check between the SDC files of the top-level and block-level design units.

Prerequisites

Specify the top-level SDC file with `-mode flatTop`. Specify the block level SDC file with `-mode block`.

Specifying the SGDC File for Equivalence Check

This section illustrates how to specify the SGDC file for performing an equivalence check for a design that has multiple hierarchies. The name of the top design unit is `top`, `blockA` and `blockB` are a block instances inside `top`. The block instance `blockC` is inside `blockB`. The design is illustrated as follows.



The SDC files that need to be compared are:

- design top top.sdc and block blockA.sdc
- design top top.sdc and block blockC.sdc

The SDC files for the top-level and block-level design units are specified as follows:

```
current_design top
sdc_data -file top.sdc -mode flatTop
block -name blockA
block -name blockC
```

```
current_design blockA
sdc_data -file blockA.sdc -mode block
```

```
current_design blockC
sdc_data -file blockC.sdc -mode block
```

In the SGDC file specified above, `flatTop` and `block` are the modes that differentiate the SDC files. Refer to the [-mode <mode>](#) section for more details.

If a skeleton SDC file, `skeleton.sdc`, is specified using the mode `skeleton`, the `skeleton.sdc` file overrides the constraints specified in the `block.sdc` file (Refer to the [Specifying the Skeleton SDC File to Check SDC Equivalence](#) for details).

The `Equiv_SDC_Block` rule checks for equivalence between the same commands as the `Equiv_SDC` rule. For example, `create_clock` is compared with `create_clock`. See the [Equiv_SDC](#) documentation for more information.

NOTE: *There are some commands for which equivalence is always performed before the commands specified by you. For details on those commands, refer to the [Understanding the Equivalence Flow](#) section.*

Parameter(s)

- *equiv_sdc_constraint_file*: Default is none, therefore an equivalence check is checked for all commands. To check for specific commands, state them in a file and set the value of this parameter to that file.
- *ignore_incremental_equivalence*: Default is no. Set the value to yes to check the equivalence of other commands even when the equivalence of commands with high precedence cannot be established.
- *tc_lvpc*: Default is 20. Set the value to any positive integer to limit the reporting of violations that have a similar reason. To report all violations, set the value to -1.
- *equiv_sdc_show_violations*: Default is no. Set the value to yes to make the messages visible in the message tree.
- *equiv_sdc_ambiguous_clock_file*: Default value is none. Set this parameter to the name of a file containing ambiguous clocks.
- *equiv_sdc_ignore_sca_for_top_block_equivalence*: Default is no. Set this parameter to yes to not display messages of type "specified only in top" for the top-block equivalence check in the *Equivalence Report*.
- *equiv_sdc_report_blocked_clocks*: Default is no. Set this parameter to yes if you want all the clocks, such as blocked clocks, to be reported in the *Equivalence Report*.
- *equiv_sdc_copy_sca_sdc*: Default is none. Set this parameter to the directory path where you want to store the SDC file, which contains the *set_case_analysis* propagated from Top.
- *equiv_sdc_import_sca_from_top*: By default, the value of this parameter is no. Set this parameter to yes to import constant values from the Top SDC file and apply them on block ports.
- *equiv_sdc_tolerance*: Default is 0. Set the value to any value between 0 and 100 to specify the degree of tolerance when checking for equivalence between commands.
- *equiv_sdc_clock_precision*: Default is 2 and the clock period comparison is till the second decimal place. You can set this parameter to any value between 1 and 4 to signify the decimal place for clock period comparison.

- *tc_regression_mode*: Default is 0 and the regression mode is disabled. This indicates that the timing exceptions spreadsheet contains all SDC and schematic back references. When you enable regression mode by setting this parameter to 1, SDC and schematic back references are not created in the timing exceptions spreadsheet.
- *tc_report_backref_all*: Default is 0 and only the first 10,000 rows have SDC and schematic back references. This saves runtime. Set this parameter to 1 to report all SDC and schematic back references in the timing exceptions spreadsheet.
- *tc_report_verbose*: Default is 1 and verbose reporting is enabled. Set this parameter to 0 to disable verbose reporting. When verbose reporting is disabled, equivalent timing exceptions are not reported.
- *equiv_sdc_check_fanin_fanout_for_clocks*: Default is no. This indicates that the fan-in, fan-out relationship is not checked during clock equivalence. Set this parameter to yes to consider fan-in, fan-out relationships during clock equivalence. See [Example: Impact on Clock Equivalence](#).
- *equiv_sdc_object_name_length*: Default is 12. This means that 12 characters are displayed in the worksheet title bar and any longer block instance name is truncated to 12 characters. Set the value to 0 to remove the length limit. Alternatively, set this parameter to a positive integer value greater than 2 to set the length.
- *equiv_sdc_report_type*: Default is both. This means that both equivalent and inequivalent constraints are reported. Set this parameter to *equiv_only* to report only equivalent constraints and *inequiv_only* to report inequivalent constraints only.
- *report_inactive_gen_clocks*: Default is yes and inactive generated clocks are removed from the design.
- *equiv_sdc_disambiguate_same_names_clock*: Default is no. When you set this parameter to yes, you need not specify the ambiguous clocks which have the same name, through the *equiv_sdc_ambiguous_clock_file* parameter.

Constraint(s)

None

Messages and Suggested Fix

The *Equiv_SDC_Block* rule generates the same violation messages as the *Equiv_SDC* rule. The violation messages generated by the *Equiv_SDC_Block* rule also contain the instance name and master name of the block. For debugging information, refer to the How to Debug sections of the *Equiv_SDC* rule.

The messages and how to debug that are unique to this rule are mentioned below.

- [Messages for Non-existence of SDC](#)
- [Messages for Constraints Non-equivalence](#)
- [Message for Report Generation](#)
- [Messages for Delay Paths from Top to Block](#)

Messages for Non-existence of SDC

If you do not specify the mode names (flatTop/block) for the SDC files, the *Equiv_SDC_Block* rule generates the following violation messages:

[FATAL] No schema with -mode "block" is provided for design "block"

[FATAL] No schema with -mode "flatTop" is provided for design "block"

[FATAL] No schema with -mode "block" and -mode "flatTop" is provided for design "block"

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

The rule will not perform any checks.

How to Debug and Fix

Specify the SDC file for top with -mode flatTop and for block with -mode block.

Messages for Constraints Non-equivalence

Message 1

The following messages appear when [set_clock_latency](#) set for a port or clock in both modes is not equivalent:

[WARNING] set_clock_latency (<source>) is not equivalent (block {BLOCK: <instance_name>} = "<value1>" should be greater than fl atTop {<top_name>} = "<value2>" for clock "<clk_name>".

[WARNING] set_clock_latency (network) is not equivalent (block {BLOCK: <instance_name>} = "<value1>" should be less than fl atTop {<top_name>} = "<value2>" for clock "clk_name").

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

There are two types of [set_clock_latency](#) constraints, source and network. Source latency caters to delay outside the block and network latency caters to delay inside the block. In a top vs. block hierarchy, block will always be inside top. Therefore, source delay outside the block has to be greater in number at block vs. top and network delay inside the block has to be lesser at block compared to top. If this relationship is not there, either of the values at top or block is incorrect. If you perform timing analysis with wrong numbers, it will not be correct. Either this would take longer to reach timing closure or even lead to chip failure, if left undetected.

How to Debug and Fix

The value of all the options (early, late, max, min, rise, fall) that are not equivalent in both modes. Update the non-matching options in the SDC file.

Message 2

The following message appears when a constant value (VSS and VDD) from top does not have a corresponding [set_case_analysis](#) at a block:

[WARNING] constant (VSS or VDD) value from top does not have corresponding set_case_analysis at block, impacting the endpoint <end-point-name>

Potential Issues

The violation message appears because the [set_case_analysis](#) constraint is

not applied in the fan-in of the reported object at block.

Consequences of Not Fixing

If [set_case_analysis](#) is not equivalent, the result of equivalence of other exceptions might be incorrect.

How to Debug and Fix

The schematic shows the path from VSS/VDD at top-level to block-level port. The RTL line causing the difference is highlighted.

Message 3

If the block-level port is being driven by or is generating the top-level sequential cell and [set_input_delay](#) or [set_output_delay](#) is not specified for the clocks that are sampling those sequential cells, the following message is reported:

```
[WARNING] <delay-type> for port "<port-name>" with clocks  
{<clk-name-list>} not set in block <block-name>
```

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

This is basically a case of missing [set_input_delay](#) or [set_output_delay](#). The block level SDC is incomplete. Timing analysis will not be complete and timing closure will take longer.

How to Debug and Fix

To view the top-level design for the block-level violation, perform the following steps:

1. Double-click the violation message.
2. Click **Incremental Schematic**.
3. Right-click the Incremental Schematic window, and click **Show Associated Data**. The Show Associated Data window appears.
4. Select the appropriate **Associated Messages** check boxes. The top-level design appears in the Incremental Schematic window.

NOTE: *The top-level design schematic does not label the clocks. To view the schematic with clocks labeled, run the [Show_Clock_Propagation](#) rule.*

The *Equiv_SDC_Block* rule does not generate a violation message in the

following cases:

Command	Reason
<i>set_clock_latency</i>	If the source latency of <i>block</i> is greater than flatTop and the network latency of block is less than flatTop.
<i>set_clock_transition</i>	The transition of <i>block</i> is less than or equal to flatTop
<i>set_input_transition</i>	A top-level input/inout port and block-level input/inout port are compared for the <i>set_input_transition</i> command equivalence only if both the ports are connected without any logic between them. Note: If the transition value at the block level is greater than the transition value at the top level, then the <i>Equiv_SDC_Block</i> rule will not generate a violation.
<i>set_input_delay/</i> <i>set_output_delay</i>	The delay value of <i>block</i> is greater than flatTop. If <i>set_false_path</i> is defined for the timing-path, missing <i>set_input_delay/set_output_delay</i> are not reported.
<i>set_load</i>	A top-level port and a block-level port are compared for <i>set_load</i> equivalence only if both the ports are connected without any logic between them. In addition, a top-level net with which the <i>set_load</i> command is set is compared for equivalence if the net is inside a block.
<i>set_max_delay/</i> <i>set_min_delay</i>	If the value of <i>set_max_delay/set_min_delay</i> specified at block is less than the value specified at top for a timing path.

Message for Report Generation

The following message appears for instance <instance-name> of block <block-name>:

[WARNING] Equivalence report is prepared for instance <instance-name> of block "<block-name>"

For information on debugging, refer to [How to Debug and Fix](#).

Messages for Delay Paths from Top to Block

The following messages appear when there exists a delay path from top-level clocks to the block ports. Refer to [Example 3](#).

Message 1

The following message appears when a delay path generated by one clock `<delay-path-clock>` in flatTop mode and by clocks `<block-clock-name>` in the block mode:

[WARNING] Delay paths generated by clock `<delay-path-clock>` in `<top-name>` are constrained by [clock|multiple clocks] `<block-clock-name>` in `<block-name>`

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

Only one clock is generating a delay path to the design object in *flatTop mode*, but multiple clocks waveforms are constraining that design object through IO delay in *block mode*. The SDC file highlights the lines causing the difference. Review the IO delays in the block mode and ensure that the clock of the delay paths matches exactly with the IO delays. Remove the IO delays that do not have a corresponding constraint at the top-level.

Message 2

The following message appears when a delay path generated by one clock `<delay-paths-clocks>` in flatTop mode but not in the block mode:

[WARNING] Delay paths generated by clock `<delay-paths-clocks>` in `<top-name>` are not constrained by any clock in `<block-name>`

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

Only one clock is generating a delay path to the design object in *flatTop mode*. It may be possible that IO delays are not specified in *block mode* for the design object. The SDC file highlights the lines causing the difference. Review the IO delays in the block mode and ensure that the clock of the delay paths matches corresponding with a matching IO delays. Add the IO delays in block to match the corresponding constraint at the top-level.

Message 3

The following message appears when IO delays constrained by clock `<block-clocks-names>` in block mode is constrained by delay paths generated by clock `<delay-paths-clocks>` in flatTop mode:

[WARNING] IO Delays constrained by [clock|multiple clocks] `<block-clocks-names>` in `<block-name>` are constrained by delay paths generated by clock `<delay-paths-clocks>` in `<top-name>`

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

In both modes, multiple clocks waveforms are constraining the design object. In this case, the mapping of clocks waveforms must be specified in the ambiguous clock-mapping file. Otherwise, the rule will not do equivalence of other constraints. The SDC file highlights the lines causing the difference.

Message 4

The following message appears when IO delays constrained by clock `<block-clocks-names>` in block mode is constrained by IO delays `<IOClocks-names>` and delay paths generated by clock `<delay-paths-clock>` in flatTop mode:

[WARNING] IO Delays constrained by [clock|multiple clocks] `<block-clocks-names>` in `<block-name>` are constrained by [clock|multiple clocks] `<IOClocks-names>` and delay paths generated by clock `<delay-paths-clocks>` in `<top-name>`

Refer to [Potential Issues](#) and [Consequences of Not Fixing](#).

How to Debug and Fix

In both modes, multiple clocks waveforms are constraining the design object. In this case, the mapping of clocks waveforms must be specified in the ambiguous clock-mapping file. Otherwise, the rule will not do equivalence of other constraints. The SDC file highlights the lines causing the difference.

Potential Issues

The violation message explicitly states potential issues.

Consequences of Not Fixing

If the clocks characteristics constraining the design object is not same, you are performing timing analysis for the same design using different

assumptions. Since only one of them could be valid, there is a chance that analysis is not appropriate and your chip may fail though your timing numbers will be met.

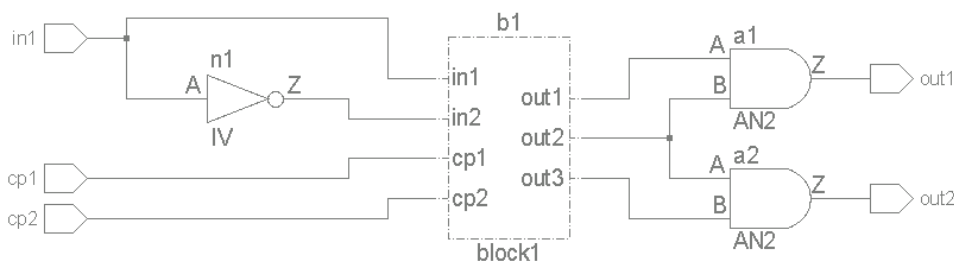
If the number of clocks constraining the design object is different, you are performing one set/mode of analysis in one case, while doing multi-mode analysis in the other case. From an equivalence perspective, these would not be equivalent. Therefore, modes that are left out in one case should be provided so that meaningful equivalence check can be done.

Example Code and/or Schematic

Example 1

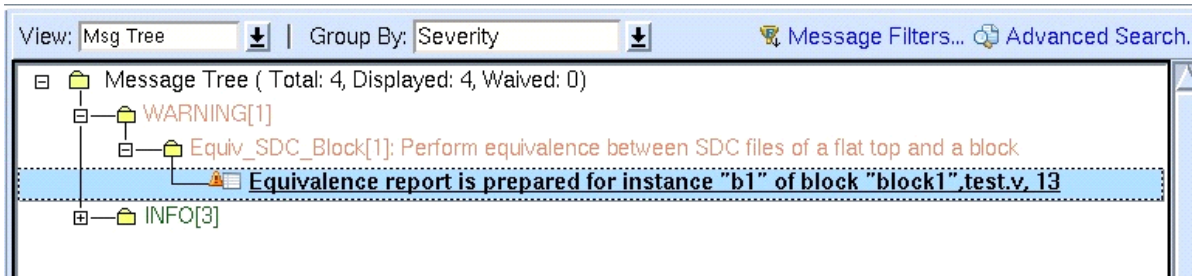
[View Test Case Files](#)

This example shows how to open the Equivalence report. The schematic of the top module is as follows:



Double-click the violation message to view the Equivalence Report.

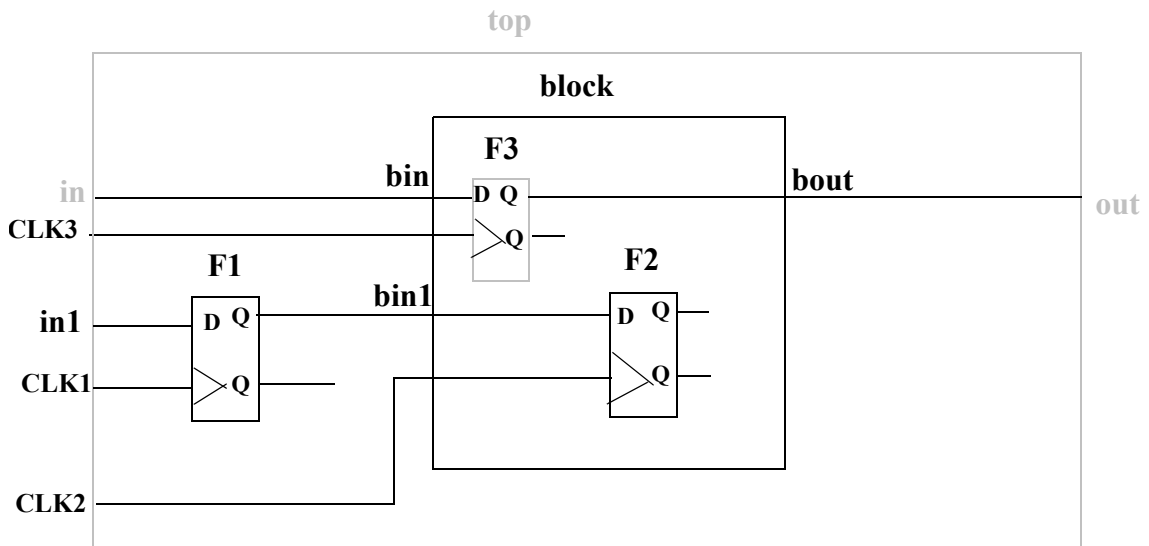
Constraints Management Rules



Example 2 Block Top Clock Equivalence Analysis

Test Case Files Not Available

Read the following example to understand the IO delay that should be present in the block with respect to the top-level SDC file:



In the above figure, the flip-flops F1, F2, and F3 are driven by clock CLK1, CLK2, and CLK3 respectively.

```
create_clock -name CLK1 -period <period> [get_port clk1]
create_clock -name CLK2 -period <period> [get_port clk2]
create_clock -name CLK3 -period <period> [get_port clk3]
```

In addition, assume that the following constraints are provided in the top-level SDC file:

```
create_clock -name VCLK1 -period <period>
create_clock -name VCLK2 -period <period>
set_input_delay <value> -clock VCLK1 {in}
set_output_delay <value> -clock VCLK2 {out}
```

The block-level SDC file is directly connected to bin (probably through some combinational logic). Therefore, an input delay should be present for the bin port with respect to clock VCLK1. Similarly, for the port bout an output delay should be present with respect to clock VCLK2. In addition, the values at the block-level should be greater than or equal to the values at the top level.

However, a timing path also exists from flip-flop F1 to flip-flop F2 through the port bin1. Therefore, there is a need for an input delay in for the port bin1 with respect to a virtual clock similar to CLK1.

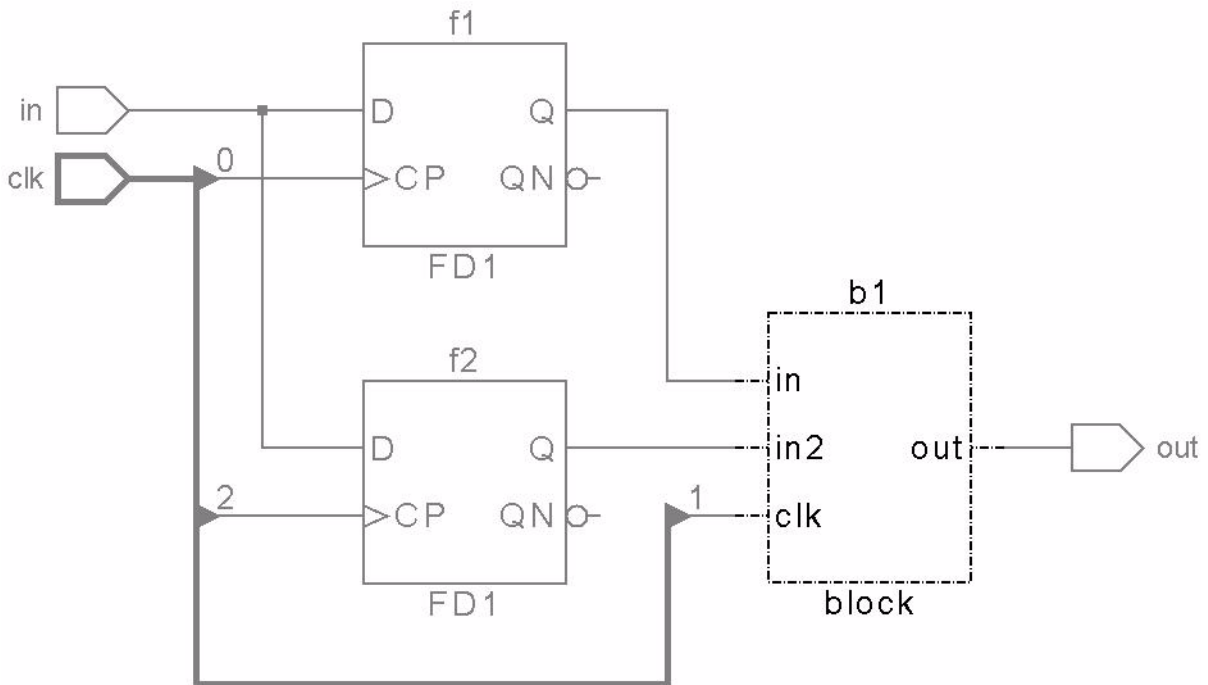
A similar case would have existed for the output delay had there been a sequential path through an output port of a block to a flip-flop outside the block. Therefore, a correct block-level SDC should be specified as follows:

```
create_clock -name BCLK2 -period <period> [get_port clk2]
create_clock -name BCLK3 -period <period> [get_port clk3]
create_clock -name VCLK1 -period <period>
create_clock -name VCLK2 -period <period>
create_clock -name CLK1 -period <period>
set_input_delay <value> -clock VCLK1 {bin}
set_output_delay <value> -clock VCLK2 {bout}
set_input_delay <value> -clock clk1 {bin1}
```

Example 3 Delay Paths Existence from Top-to-Block Port

Test Case Files Not Available

This example illustrates the clock inequivalence scenarios belonging to the block top equivalence.



The example above shows the schematic of top level module, block being b1. If a clock is specified at clk, there consequently exists sampling path from f1 to the block input in. Hence, while doing block top equivalence for clock, there must exist a input delay constraint at in. If there is not such a constraint, a clock not equivalence violation is shown.

Default Severity Label

Warning

Rule Group

const_mgmt

Report and Related Files

- [Equivalence Report](#)
- [Clock-Mapping Report](#)

Equiv_SDC_Dual_Design

Performs equivalence between two SDC files of two equivalent designs

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

The *Equiv_SDC_Dual_Design* rule performs an equivalence check between two SDC files of two equivalent designs. The

Review the following sections:

- [Prerequisites](#)
- [Order of Equivalence Check](#)
- [Rule Exceptions](#)

Prerequisites

Specify two SDC files are specified using `reference` and `implement` as the mode names in two separate SGDC files.

The strings for the `-mode <mode>` argument should appear under the two different designs. For example, if you want to perform an equivalence check between a file `file1.sdc` belonging to the design `Top1` and file `file2.sdc` belonging to its equivalent design `Top1`, then the SGDC constraint should be specified as follows:

```
reference.sgdc
current_design Top1
sdc_data -file file1.sdc -mode reference

implement.sgdc
current_design Top1
sdc_data -file file2.sdc -mode implement
```

If you do not specify the mode names (`reference/implement`) for the SDC files of the two designs, the *Equiv_SDC_Dual_Design* rule generates the following violation messages:

[FATAL] No schema with -mode "reference" is provided for design "top"

[FATAL] No schema with `-mode "implement"` is provided for design "top"

[FATAL] No schema with `-mode "reference"` and `-mode "implement"` is provided for design "top" and its equivalent design "top"

The design can either be a top-level design or a block-level design. The equivalent design file (mapping file) is specified using the [equiv_sdc_design_equivalence_file](#) parameter. You can specify one-to-many mapping in the mapping file, see [Specifying One-to-Many Mapping](#). It is mandatory to specify the equivalent design file (mapping file) in the [equiv_sdc_design_equivalence_file](#) parameter for the `Equiv_SDC_Dual_Design` rule to work properly.

NOTE: *To enable one-to-many mapping, set the environment variable "SDC_EQUIV_ONE_TO_MANY" to 1 using the "setenv SDC_EQUIV_ONE_TO_MANY 1" command in the shell, before running SpyGlass.*

Design specific options must be specified for both `reference` and `implement` designs, individually. Implement design options must be specified in the project file and reference design options must be specified in the `reference_design_sources` file. For example, to generate the `moresimple_csv` report for the `implement` design, specify the following option in the project file:

```
set_option report {moresimple_csv}
```

To generate the `moresimple_csv` report for the `reference` design, specify the following in the `reference_design_sources` file:

```
-report {moresimple_csv}
```

Order of Equivalence Check

The `Equiv_SDC_Dual_Design` rule performs an equivalence check between two SDC files incrementally in the following order:

1. `set_case_analysis`
2. `create_clock/create_generated_clock`
3. `set_input_delay/set_output_delay`
4. `set_clock_transition/set_input_transition/
set_clock_uncertainty/set_clock_latency/set_load`

5. `set_false_path/set_clock_groups/
set_multicycle_path/set_min_delay/set_max_delay/
set_disable_timing`

NOTE: *There are some commands for which an equivalence check is always performed before the commands specified by you. For details on those commands, refer to the [Understanding the Equivalence Flow](#) section.*

This rule considers two sets of clocks, one in each mode, that has the same overall effect on the design as equivalent. It supports one-to-many and many-to-many clock mapping for clock equivalence.

Refer to the [Example 1: One-to-Many Clock Mapping](#) and [Example 2: Many-to-Many Clock Mapping](#) sections for more information.

Use the `strict` parameter to report equivalent clocks based on the points of definition. When this parameter is set to `yes` and equivalent clocks have different points of definition, they are reported as not equivalent. However, in other types of equivalence, such as timing exceptions and clock transitions, the clocks would be treated as equivalent. Refer to the [Example 3: Implication of the Strict Parameter](#) section for more information.

Rule Exceptions

This rule has the follow exceptions:

- This rule should not be run together with any other rules.
- If you do not specify the mapping file for a design object, the object is not considered while doing equivalence for dual design.
- If you run the `Equiv_SDC_Dual_Design` rule with the value of the `equiv_sdc_ignore_unequivalent_design_obj` parameter set to `no`, the violations reported by this rule may not be valid.

Parameter(s)

- `equiv_sdc_design_equivalence_file`: Default value is `none`. Set the value to the design equivalence file that provides a report to map the ports, registers, and any intermediate design object used in the SDC files of one design to another.
- `equiv_sdc_ignore_unequivalent_design_obj`: Default is `yes`.

Sometimes, it is not possible to map all points (sequential elements and output ports) of one design with another. This may occur particularly if

one design is RTL and the other design is a netlist design. This may lead to incorrect violations as in certain cases, equivalence is established based on ports and registers. To avoid such violations, ensure the value of the *equiv_sdc_ignore_unequivalent_design_obj* parameter is set to *yes*.

For example, two clocks are considered equivalent if and only if both of them drive the same set of flip-flops with the same polarity. However, if a clock drives a flip-flop whose equivalent point is missing in the other design, then we cannot match the two sets of flip-flops driven by the two clocks which results in a violation.

This violation can be avoided either by providing the corresponding equivalent point in the other design or by not considering the end points whose equivalent points are missing by ensuring the value of the *equiv_sdc_ignore_unequivalent_design_obj* parameter to *yes*.

- *equiv_sdc_constraint_file*: Default is *none*. This means that the rule performs an equivalence check on all commands. Limit the check by setting the value to a file name, which contains the SDC commands that need to be checked for equivalence.
- *tc_lvpc*: Default is 20. Set the value of this parameter to any positive integer to limit the reporting of violations that have a similar reason. Default is 20. To report all violations, set the value to -1.
- *ignore_incremental_equivalence*: Default is *no*. Set the value to *yes* to check the equivalence of other commands even when the equivalence of commands with high precedence cannot be established.
- *equiv_sdc_ambiguous_clock_file*: Default value is *none*. Set this parameter to the name of a *file* containing ambiguous clocks.
- *equiv_sdc_check_fanin*: Default is *no*. Set this parameter to *yes* to check the fan-ins of objects provided in the mapping file and whether the mapping file is incomplete.
- *strict*: Default value is *no*. Set this parameter to *yes* report equivalent clocks as not equivalent based on the points of definition of the clocks.
- *equiv_sdc_tolerance*: Default is 0. Set the value to any value between 0 and 100 to specify the degree of tolerance when checking for equivalence between commands.
- *equiv_sdc_clock_precision*: Default is 2 and the clock period comparison is till the second decimal place. You can set this parameter to any value

between 1 and 4 to signify the decimal place for clock period comparison.

- *tc_regression_mode*: Default is 0 and the regression mode is disabled. This indicates that the timing exceptions spreadsheet contains all SDC and schematic back references. When you enable regression mode by setting this parameter to 1, SDC and schematic back references are not created in the timing exceptions spreadsheet.
- *tc_report_backref_all*: Default is 0 and only the first 10,000 rows have SDC and schematic back references. This saves runtime. Set this parameter to 1 to report all SDC and schematic back references in the timing exceptions spreadsheet.
- *tc_report_verbose*: Default is 1 and verbose reporting is enabled. Set this parameter to 0 to disable verbose reporting. When verbose reporting is disabled, equivalent timing exceptions are not reported.
- *equiv_sdc_fast_exceptions_equivalence*: Default is no. Set this parameter to yes to enable the new equivalence check algorithm. This algorithm provides runtime improvement for timing exception equivalence.
- *equiv_sdc_report_type*: Default is both. This means that both equivalent and inequivalent constraints are reported. Set this parameter to *equiv_only* to report only equivalent constraints and *inequiv_only* to report inequivalent constraints only.
- *equiv_sdc_disambiguate_same_names_clock*: Default is no. When you set this parameter to yes, you need not specify the ambiguous clocks which have the same name, through the *equiv_sdc_ambiguous_clock_file* parameter.

Constraint(s)

None

Messages and Suggested Fix

The *Equiv_SDC_Dual_Design* rule generates the same messages as the *Equiv_SDC* rule.

This rule also generates *Message 3* when there are errors in the mapping file.

In addition, this rule generates the following messages.

Message 1

The following message appears when the *ignore_incremental_equivalence* parameter is not set:

[WARNING] Report generated for objects specified in the mapping file but not found in the design

Message 2

The following message appears when the *ignore_incremental_equivalence* parameter is set:

[WARNING] Report generated for objects specified in the mapping file but not found in the design. Since the parameter *ignore_incremental_equivalence* is set, so all the mapping error violation are ignored hence the equivalence results may not be correct

Potential Issues

The violation messages explicitly state the potential issues.

Consequences of Not Fixing

There is a high probability of inaccurate results.

How to Debug and Fix

Review the reported objects and delete that object from the mapping file if it is not in the design.

Example Code and/or Schematic

This section contains the following examples:

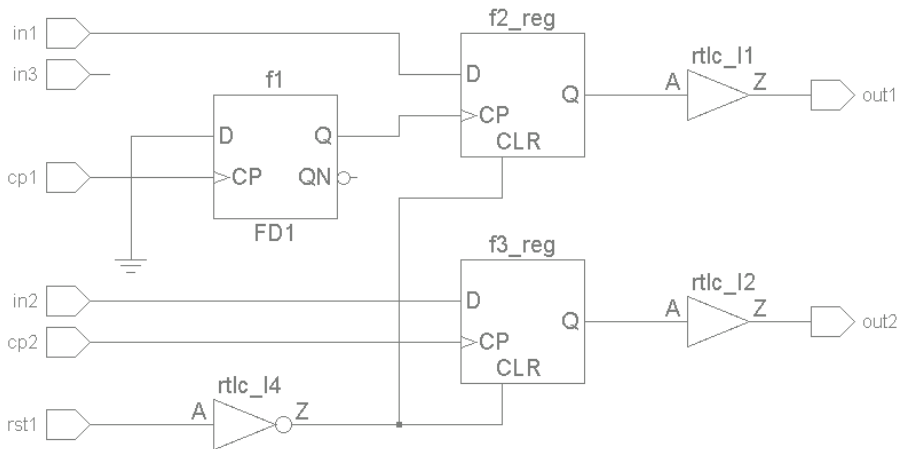
- [Example 1: Opening the Equivalence Report](#)
- [Example 2 Specifying the Design Files and Viewing Case Analysis and Clock Propagation Information](#)
- [Example 3: Implication of the Strict Parameter](#)

Example 1: Opening the Equivalence Report

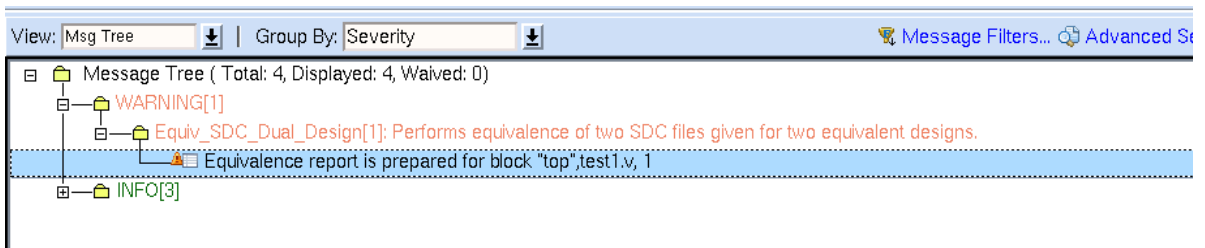
[View Test Case Files](#)

This example shows how to open the Equivalence report. The schematic of the top module is as follows:

Constraints Management Rules



Double-click the violation message to view the Equivalence Report.



Example 2 Specifying the Design Files and Viewing Case Analysis and Clock Propagation Information

Test Case Files Not Available

Design files for implement design are loaded in the GUI using the implement project file.

In the implement project file, reference design sources (containing RTL and

constraints) need to be loaded using:

```
set_goal_option reference_design_sources
```

Consider the following implement design project file:

```
##Data Import Section
read_file -type verilog test_netlist.v
##Common Options Section
set_option language_mode mixed
set_option projectwdir .
set_option projectcwd <path to project CWD>
set_option active_methodology <path to methodology>
set_option top <top design name>
##Goal Setup Section
current_methodology <path to current methodology>
current_goal Constraints/pre_layout/sdc_equiv_dual_design -
top top
read_file -type sgdc <path to implement SGDC>
set_goal_option reference_design_sources { <reference source
file name> }
set_parameter equiv_sdc_design_equivalence_file <path to
equivalence design file name>
```

The reference source file contains the Verilog sources, libs, and the SGDC file of the reference design.

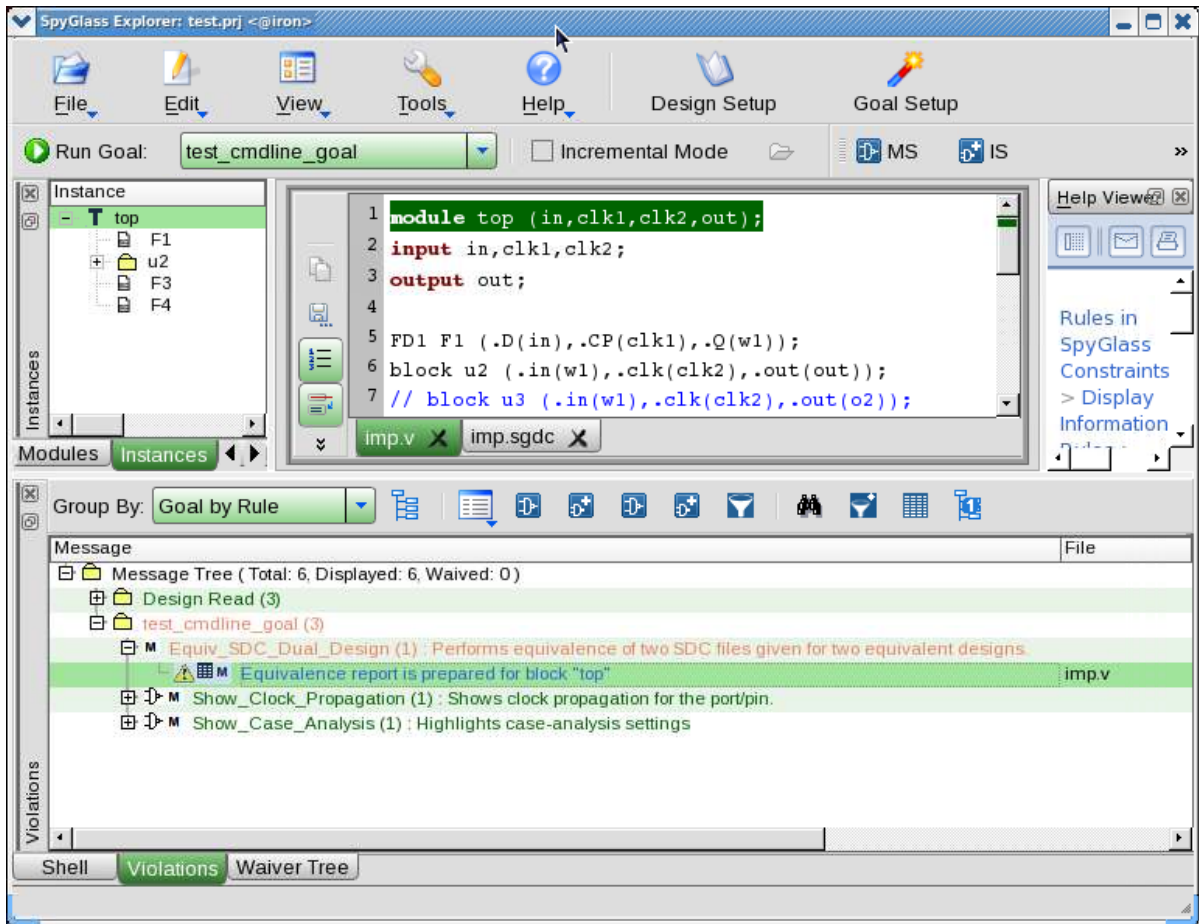
Example

```
-verilog test_rtl.v
-sglib mylib.sglib
-sgdc reference_constraints.sgdc
```

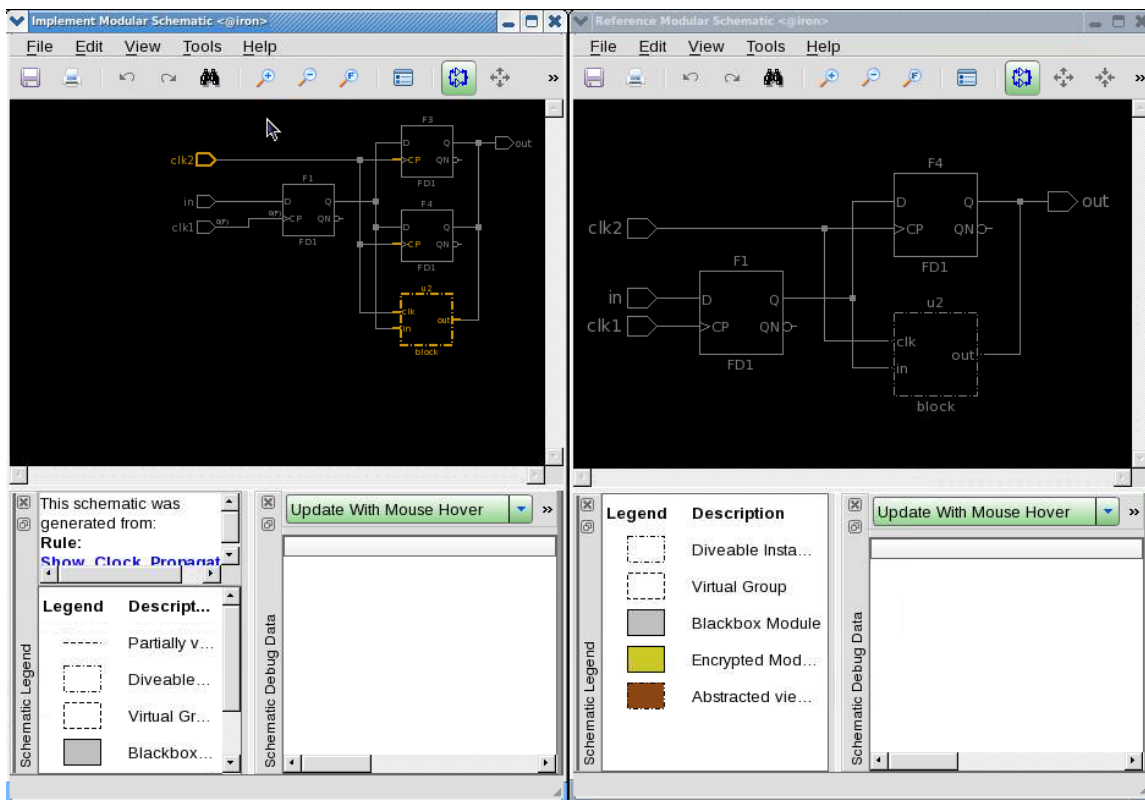
Schematic Details

The GUI contains separate schematic buttons for reference and implement design schematics (Modular and Incremental). The following figure shows the GUI with the Message Tree and the schematic buttons:

Constraints Management Rules



You can click the Implement Incremental/Modular Schematic and the Reference Incremental/Modular Schematic buttons to open and compare the schematic information of the two designs. The following figure shows the Implement Modular Schematic and the Reference Modular Schematic.



Viewing the Case Analysis and Clock Propagation Information in the Schematics

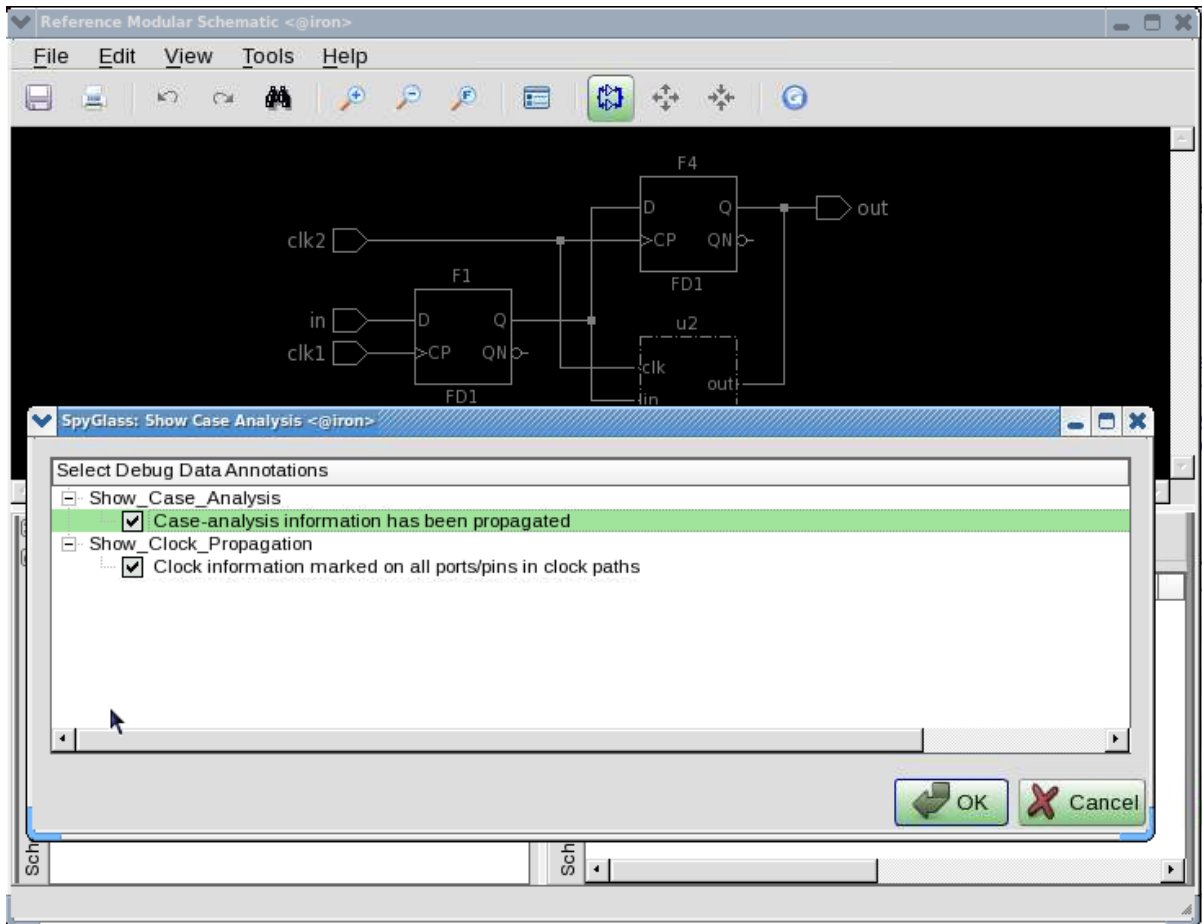
From both the modular and incremental schematics, you can view:

- The clock information marked on all ports/pins, in clock paths
- The case-analysis information

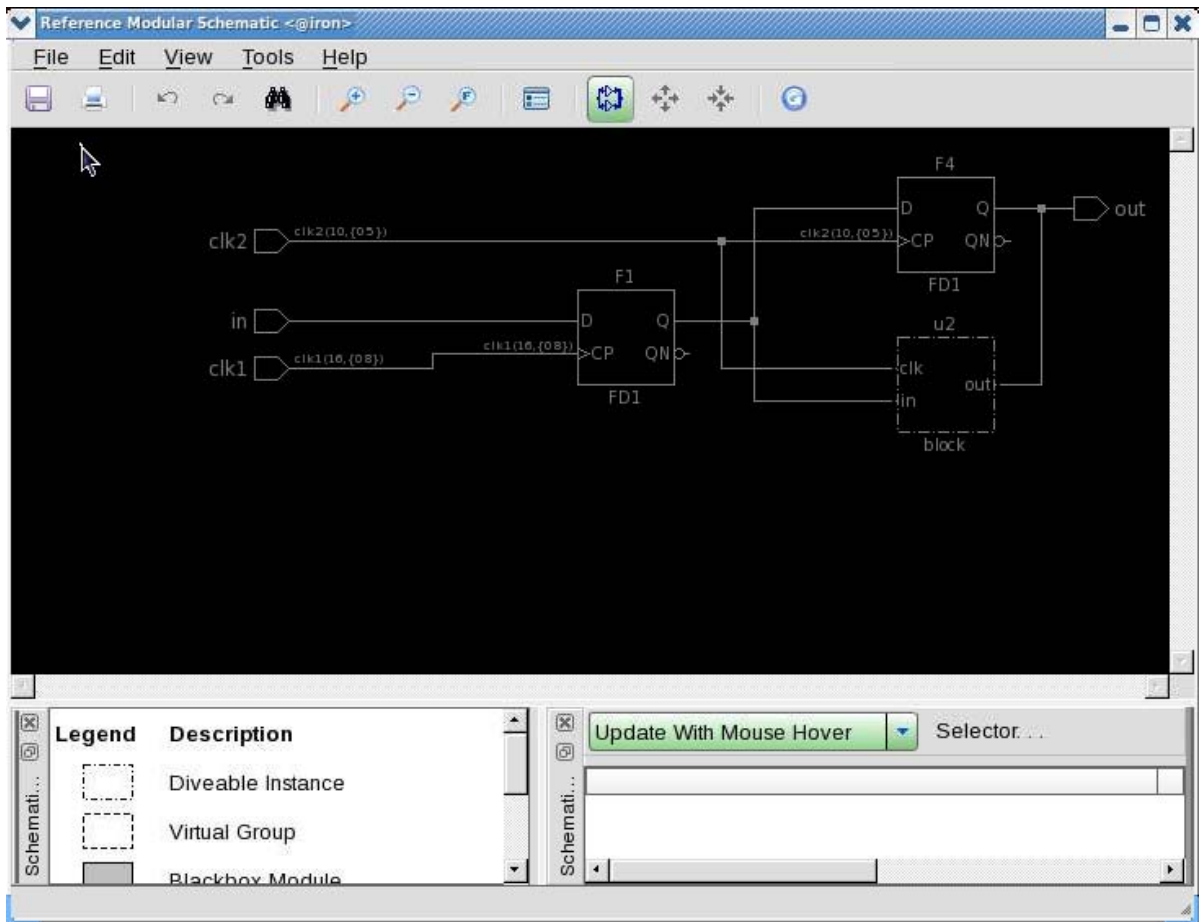
To do this, perform the following steps:

1. Open the schematic window.
2. Click the **Edit** menu and then the **Annotate Debug Data** option. The **SpyGlass: Show Case Analysis** window appears.

Constraints Management Rules



3. To view case analysis information, select the **Case-analysis information has been propagated** check box.
4. To view clock propagation, select the **Clock information marked on all ports/pins in clock paths** check box.
5. Click **OK**. The schematic is annotated with the case-analysis and clock information:



To remove the annotation, click the **Edit** menu and then the **Clear Debug Data** option.

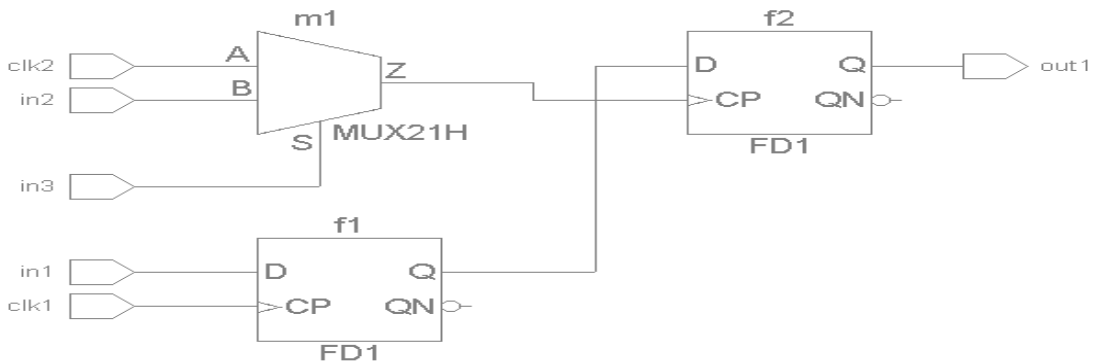
Example 3: Implication of the Strict Parameter

Test Case Files Not Available

Constraints Management Rules

This example illustrates the implication of setting the *strict* parameter to *yes*.

Before you understand the implication of setting the parameter to *yes*, let's review the default behavior. By default, this parameter is set to *no* and the points of definition of clock is not considered when determining their equivalence.



The constraints defined for reference and implement modes are:

Reference Mode Constraints

```
create_clock -name r1 -period 10 [get_ports {clk1}]
create_clock -name r2 -period 10 [get_ports {clk2}]
set_case_analysis 0 in3
```

Implement Mode Constraints

```
create_clock -name i1 -period 10 [get_ports {clk1}]
create_clock -name i2 -period 10 [get_pins {m1/Z}]
```

After running this rule, the following spreadsheet is generated. Notice that the reference mode clock *r2* and implement mode clock *i2* are reported as equivalent in the spreadsheet (second row).

When you set the *strict* parameter to *yes*, these clocks are reported as not equivalent because their points of definition are different. Though, the clocks are reported as not equivalent, the timing paths formed by these clocks are treated as equivalent and accordingly reported in other

equivalence checks.

	B	C	D	E	F	G
	Equivalent	Analysis	Clocks in reference mode	Clocks in implement mode	reference Design Object	Implement Design Object
1	Yes	Equivalent Clocks but differ in name	r1:default_waveform:positive	i1:default_waveform:positive	f1/CP	i1/CP
2	Yes	Equivalent Clocks but differ in name and the object on which it is defined.	r2:default_waveform:positive	i2:default_waveform:positive	f2/CP	i2/CP

Default Severity Label

Warning

Rule Group

const_mgmt

Report and Related Files

- [EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT](#): Lists all the flops/ports for which mapping is not specified.
- [Equivalence Report](#)
- [Clock-Mapping Report](#)

Equiv_SDC_Top

Performs equivalence between the top-level design unit

When to Use

Use this rule in the RTL, Pre-layout, and Post-layout phases.

Description

The *Equiv_SDC_Top* rule performs an equivalence check between the SDC files of the top-level design unit. The *Equiv_SDC_Top* rule considers all blocks under the top-level design unit as blackboxes.

Prerequisites

The SDC files are specified using `glueTop` or `flatTop` as the mode names as shown below.

```
current_design top
sdc_data -file top.sdc -mode flatTop
sdc_data -file glue_top.sdc -mode glueTop
block -name blockA
```

In the snippet above, the name of the top design unit is `top` and `blockA` is a block instance inside `top`. The two SDC files for the design `top` `top.sdc` and `glue_top.sdc` will be compared. In this example, the instance `blockA`, which is inside design unit `top`, will be considered as a blackbox. In addition, `flatTop` and `glueTop` are the modes that differentiate the two SDC files. Refer to the [-mode <mode>](#) section for more details.

If you do not specify the mode names (`glueTop/flatTop`) for the SDC files, then the *Equiv_SDC_Top* rule generates the following violation messages:

```
[FATAL] No schema with -mode "glueTop" is provided for design "top"
```

```
[FATAL] No schema with -mode "flatTop" is provided for design "top"
```

```
[FATAL] No schema with -mode "glueTop" and -mode "flatTop" is provided for design "top"
```

The *Equiv_SDC_Top* rule checks for equivalence between the same

commands as the *Equiv_SDC* rule.

NOTE: *There are some commands for which equivalence is always performed before the commands specified by you. For details on those commands, refer to the [Understanding the Equivalence Flow](#) section.*

Parameter(s)

- *equiv_sdc_constraint_file*: Default is none. This means that the rule performs an equivalence check on all commands. Limit the check by setting the value to a file name, which contains the SDC commands that need to be checked for equivalence.
- *tc_lvpc*: Default is 20. Set the value of this parameter to any positive integer to limit the reporting of violations that have a similar reason. Default is 20. To report all violations, set the value to -1.
- *ignore_incremental_equivalence*: Default is no. Set the value to yes to check the equivalence of other commands even when the equivalence of commands with high precedence cannot be established.
- *equiv_sdc_show_violations*: Default is no. Set the value to yes to make the messages visible in the message tree.
- *equiv_sdc_ambiguous_clock_file*: Default value is none. Set this parameter to the name of a file containing ambiguous clocks.
- *equiv_sdc_tolerance*: Default is 0. Set the value to any value between 0 and 100 to specify the degree of tolerance when checking for equivalence between commands.
- *equiv_sdc_clock_precision*: Default is 2 and the clock period comparison is till the second decimal place. You can set this parameter to any value between 1 and 4 to signify the decimal place for clock period comparison.

Constraint(s)

None

Messages and Suggested Fix

The *Equiv_SDC_Top* rule generates the same messages as the *Equiv_SDC* rule. However, the *Equiv_SDC_Top* rule does not generate a violation message for the *set_input_delay*/*set_output_delay* constraints if the delay

value of `flatTop` is 0.

The following message appears in glue logic, if a timing path that is referred to using IO delays is present only at the block-level SDC file and not at the top-level SDC file, :

```
[WARNING] <timing-path> for pin <pin-name> with clocks {<clock-name> [period: <value>, waveform {<value>}]} set only in flatTop
```

Where, *<timing-path>* can be [set_input_delay](#) or [set_output_delay](#).

Potential Issues

The violation message explicitly states the potential issues.

Consequences of Not Fixing

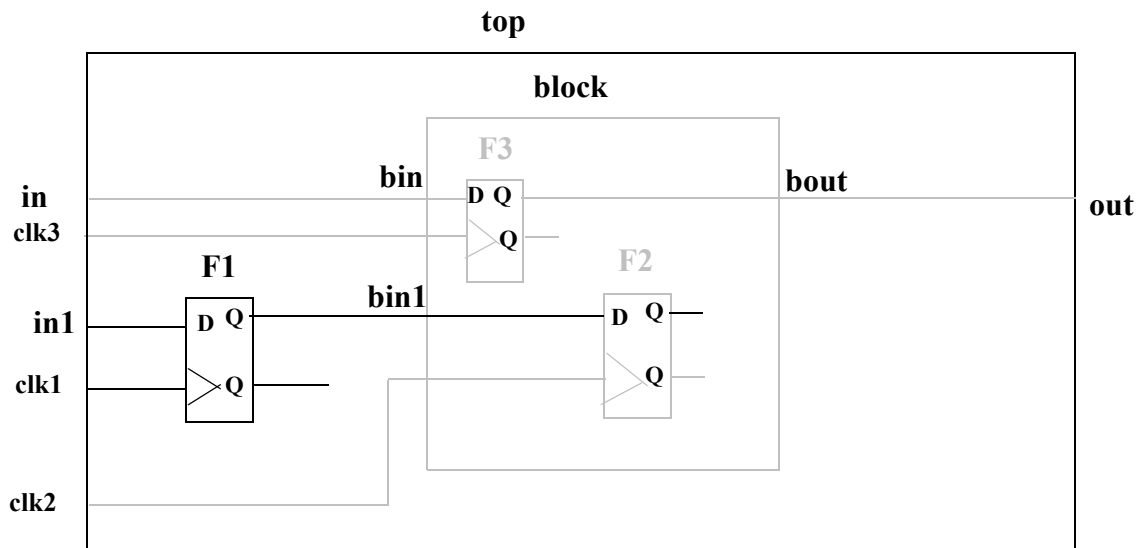
The designs are not equivalent with respect to the reported constraint.

How to Debug and Fix

Refer to the Example Code and/or Schematic section to understand the IO delay that should be present in `glueTop` with respect to the top-level SDC file.

Example Code and/or Schematic

Let us consider the following example to understand the IO delay that should be present in `glueTop` with respect to the top-level SDC file:



```
create_clock -name CLK1 -period <period> [get_port clk1]
create_clock -name CLK2 -period <period> [get_port clk2]
create_clock -name CLK3 -period <period> [get_port clk3]
```

In the above figure, let the flip-flops F1, F2, and F3 be driven by clock clk1, clk2, and clk3 respectively.

In addition, let us assume that the following constraints are provided in the top-level SDC file:

```
create_clock -name VCLK1 -period <period>
create_clock -name VCLK2 -period <period>
set_input_delay <value> -clock VCLK1 {in}
set_output_delay <value> -clock VCLK2 {out}
```

For the glue-level logic, the block module has just one interface. Therefore, the top-level input delays are retained as is. However, a timing path ends inside the block through the ports bin and bin1. Therefore, there should be an output delay present at the block boundaries. Similarly, a timing path originates from the flip-flop F3 inside the block, and therefore, an input delay should be present for block output port bout.

Therefore, a glue-level SDC should be defined as follows:

```
create_clock -name CLK1 -period <period> [get_port clk1]
create_clock -name CLK2 -period <period> [get_port clk2]
create_clock -name CLK3 -period <period> [get_port clk3]
create_clock -name VCLK1 -period <period>
create_clock -name VCLK2 -period <period>
```

```
set_input_delay <value> -clock VCLK1 {in}
set_output_delay <value> -clock VCLK2 {out}
set_output_delay <value> -clock CLK3 {block/bin}
set_output_delay <value> -clock CLK2 {block/bin1}
set_input_delay <value> -clock CLK3 {block/bout}
```

Default Severity Label

Warning

Rule Group

const_mgmt

Report and Related Files

- [Equivalence Report](#)
- [Clock-Mapping Report](#)

SDC_multimode_equiv

Performs equivalence checks between a merge-mode SDC and individual modes SDC of the same design

When to Use

Designs have a set of constraints for each mode of operation. For example, different functional modes, test mode etc. Designers try to optimize the design for each mode despite conflicting requirements.

All of the individual set of constraints of each functional mode are consolidated into a single SDC file which accounts for various modes.

- The Consolidated SDC file represents a mode (hypothetical), which covers all the timing scenarios of the individual modes
- Merged constraint is supposed to be pessimistic that it cannot under-constraint any path or design object any more than the individual modes

Use this rule to check the following:

- Mode merge SDC file is really pessimistic and not under-constraining
- Mode merge SDC file is not over constraining the design

Use this rule in the RTL, Pre-layout, and Post-layout phases.

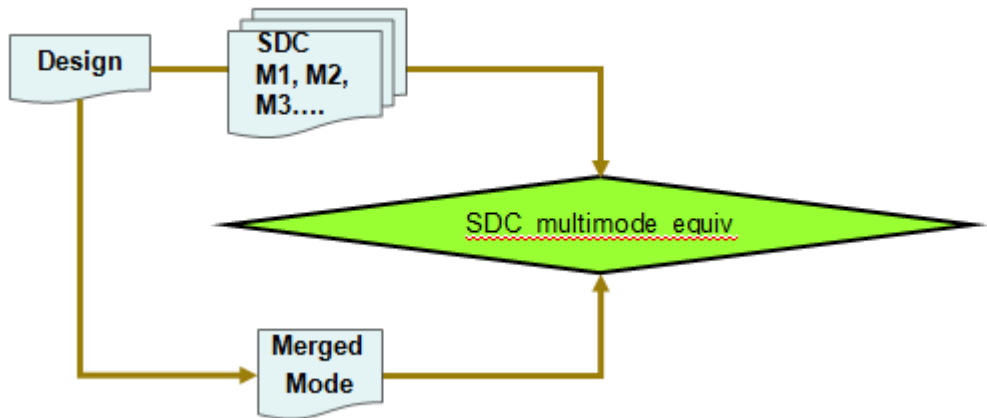
Description

The *SDC_multimode_equiv* rule performs equivalence between SDC of each of the modes and the mode merged SDC, as shown in the following diagram. By using this rule, you can verify that the merged mode covers all aspects of the individual mode.

To learn about merging multiple modes, see to [SDC_ModeMerge](#).

Review the following topics:

- [Prerequisites](#)
- [Order of Constraints Checking](#)
- [Rule Exceptions](#)



Prerequisites

the mode name for the consolidated mode that covers all the timing scenarios of the individual modes must be called merge-mode. The individual modes may have any name.

The following is an example of an SGDC specification:

```

current_design top
sdc_data -file merge_mode.sdc -mode merge-mode
sdc_data -file modeA.sdc -mode A
sdc_data -file modeB.sdc -mode B
  
```

Order of Constraints Checking

The rule performs the equivalence check between constraints of merge-mode and each individual mode incrementally in the following order:

```

set_case_analysis
create_clock/create_generated_clock
set_input_delay/set_output_delay
set_false_path/set_multicycle_path/set_max_delay/
set_min_delay
set_disable_timing
  
```

Checks for `set_case_analysis`, `create_clock/create_generated_clock`, `set_input_delay/set_output_delay` would always be performed but if merge-mode constraints under-constraint any path or design object any more than the individual mode, then this rule does not perform equivalence for `set_false_path/set_multicycle_path/set_max_delay/set_min_delay` and `set_disable_timing` further.

By default, the rule performs equivalence checks on all commands. However, you can use the [equiv_sdc_constraint_file](#) parameter to specify the commands for which equivalence check has to be performed. However, there are some commands for which equivalency is always performed before the commands mentioned in the file specified in the [equiv_sdc_constraint_file](#) parameter.

Rule Exceptions

This rule does not report a violation for timing paths that do not exist in both modes.

Parameter(s)

- [equiv_sdc_constraint_file](#): Default is none. This means that the rule performs an equivalence check on all commands. Limit the check by setting the value to a file name, which contains the SDC commands that need to be checked for equivalence.
- [tc_lvpc](#): Default is 20. Set the value of this parameter to any positive integer to limit the reporting of violations that have a similar reason. To report all violations, set the value to -1.
- [ignore_incremental_equivalence](#): Default is no. Set the value to yes to check the equivalence of other commands even when the equivalence of commands with high precedence cannot be established.
- [equiv_sdc_ignore_value](#): Default is no. Set this value to yes to consider the value between two commands while performing an equivalency check.
- [multimode_sdc_ambiguous_clock_file](#): Default is none. Set this parameter to the name of a file containing ambiguous clocks for each mode.
- [equiv_sdc_tolerance](#): Default is 0. Set the value to any value between 0 and 100 to specify the degree of tolerance when checking for equivalence between commands.

- *equiv_sdc_set_tolerance*: Default is no and the tolerance value is determined by the value of the *equiv_sdc_tolerance* parameter. Set this parameter to yes to use the delay/clock_period ratio as the tolerance value.
- *equiv_sdc_clock_precision*: Default is 2 and the clock period comparison is till the second decimal place. You can set this parameter to any value between 1 and 4 to signify the decimal place for clock period comparison.
- *tc_regression_mode*: Default is 0 and the regression mode is disabled. This indicates that the timing exceptions spreadsheet contains all SDC and schematic back references. When you enable regression mode by setting this parameter to 1, SDC and schematic back references are not created in the timing exceptions spreadsheet.
- *tc_report_backref_all*: Default is 0 and only the first 10,000 rows have SDC and schematic back references. This saves runtime. Set this parameter to 1 to report all SDC and schematic back references in the timing exceptions spreadsheet.
- *tc_report_verbose*: Default is 1 and verbose reporting is enabled. Set this parameter to 0 to disable verbose reporting. When verbose reporting is disabled, equivalent timing exceptions are not reported.

Constraint(s)

None

Messages and Suggested Fix

The following message appears after the multimode equivalence report is generated:

```
[INFO] Multi mode Equivalence report is prepared for block
"Block_Name"
```

Potential Issues

Not applicable

Consequences of Not Fixing

Not applicable

How to Debug and Fix

Double-click the violation message to view the report in Spreadsheet Viewer. A sample spreadsheet is shown as follows:

	Constraint	implement1	merge-mode versus implement2	implement3
1	Clocks	Equivalent(Over-Constrained)	Equivalent(Over-Constrained)	Inequivalent
5	Case Analysis	Equivalent(Over-Constrained)	Equivalent(Over-Constrained)	Inequivalent
9	IO Delays	Equivalent	Equivalent	Equivalent
13	Exceptions	Equivalent	Equivalent	Not Run
17	Disable Timing	Equivalent	Equivalent	Not Run

The report displays the equivalence status as follows:

- **Equivalent** (Represented in green color): Individual mode and merge-mode constraint is exactly equivalent
- **Inequivalent** (Represented in red color): Individual mode has certain constraint which is not there in merge-mode.
- **Equivalent (Over-Constrained)** (Represented in green color): Merge-mode has additional constraints compared to individual constraints

By clicking the hyperlinks, a new spreadsheet is displayed, which provides more details.

Example Code and/or Schematic

This example illustrates the multi_mode.csv spreadsheet. By expanding each category, you can get more information on the equivalence check. For example, you can determine the number of clocks, which are equivalent between merge-mode and the individual mode.

Constraints Management Rules

The screenshot shows a spreadsheet window titled "Multi_mode.csv" with a menu bar (View, Tools, Help) and a toolbar (Reload, Save, Save As, Print, Schematic(IS)). Below the toolbar is a search bar containing "Clocks". The main area is a table with columns for "Constraint", "implement1", "implement2", "merge-mode versus implement3", "implement4", and "implement5". The table contains several rows of data, including "IO Delays", "Exceptions", "Disable Timing", "Clocks", "Inequivalent Clocks", "Equivalent Clocks", "Clocks only in merge-mode", and "Case Analysis".

	Constraint	implement1	implement2	merge-mode versus implement3	implement4	implement5
9	IO Delays	Equivalent(Over-Constrained)	Equivalent(Over-Constrained)	Equivalent(Over-Constrained)	Equivalent	Equivalent
13	Exceptions	Inequivalent	Inequivalent	Not Run	Not Run	Equivalent
17	Disable Timing	Equivalent	Equivalent	Not Run	Not Run	Equivalent
1	Clocks	Equivalent(Over-Constrained)	Equivalent(Over-Constrained)	Inequivalent	Inequivalent	Equivalent(Over-Constrained)
3	Inequivalent Clocks	0	0	6	3	0
2	Equivalent Clocks	4	5	2	3	6
4	Clocks only in merge-mode	3	2	1	2	1
5	Case Analysis	Equivalent(Over-Constrained)	Equivalent(Over-Constrained)	Inequivalent	Inequivalent	Equivalent

In this spreadsheet, some of the information provided is:

- List of constraints compared
- Status of equivalence check
- Number of equivalent clocks in a specific mode.
- Number of clocks that are only in the merge-mode.
- Constraints that are not run because constraints were not equivalent in earlier constraints

Default Severity Label

Info

Rule Group

const_mgmt

Report and Related Files

- **multi_mode.csv**: See [Example Code and/or Schematic](#).

- *Equivalence Report*
- *Clock-Mapping Report*

SGDC Constraints

SpyGlass Design Constraints (SGDC) provides additional design information that is not apparent in an RTL.

In addition, you can restrict SpyGlass analysis to certain objects in a design by specifying these objects by using SGDC commands.

The following table lists the SGDC commands used by the SpyGlass Constraints solution:

<i>assume_path</i>	<i>clock</i>	<i>domain</i>
<i>mapped_pin_map</i>	<i>sdc_data</i>	<i>block</i>
<i>blocksize</i>	<i>clock_group</i>	<i>abstract_port</i>

SpyGlass Constraints Rule Parameters

Overview

This section provides detailed information on the SpyGlass Constraints solution rule parameters. You can set these parameters in both the Console GUI and Tcl by using the following syntax:

```
set_parameter <parameter_name> <parameter_value>
```

For more information on setting the parameters, refer to the *SpyGlass Tcl Interface User Guide* and *SpyGlass Explorer User Guide*.

NOTE: *Unless mentioned explicitly, all rule parameters are optional and the corresponding rules work in the default behavior mode as described.*

allow_clock_on_output_port

Allows create_clock on output port and relax the SDC_30 rule check.

Default is no and the SDC_30 rule check is performed. Set this parameter to yes to deactivate the check performed by the SDC_30 rule.

Refer to the SDC_30 rule documentation in the *SpyGlass Built-In Rules Reference Guide* for more information about the violation.

Used by	sdci_init_rule
Options	yes, no, ' '
Default	no
Example	
<i>Console/Tcl-based usage</i>	set_parameter allow_clock_on_output_port 1
<i>Usage in goal/source files</i>	-allow_clock_on_output_port=1

allow_drive

Specifies whether the [Inp_Trans01a](#) rule should flag use of the [set_drive](#) constraints (along with the [set_load](#) constraint).

By default, the [Inp_Trans01a](#) rule flags use of the [set_drive](#) constraints (along with the [set_load](#) constraint).

Used by	Inp_Trans01a
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	set_parameter allow_drive 1
<i>Usage in goal/source files</i>	-allow_drive=1

allow_driving_cell

Specifies whether the [Inp_Trans01](#) and [Inp_Trans01a](#) rules should flag use of the [set_driving_cell](#) constraints.

By default, the *Inp_Trans01* and *Inp_Trans01a* rules do not report use of the *set_driving_cell* constraint.

Used by	<i>Inp_Trans01a</i> , <i>Inp_Trans01</i>
Options	yes, 1; no, 0
Default	yes; however, if the <i>chip</i> parameter is defined then the default value of this parameter is no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter allow_driving_cell 1</code>
<i>Usage in goal/source files</i>	<code>-allow_driving_cell=1</code>

allow_input_transition

Specifies whether the *Inp_Trans01* and *Inp_Trans01a* rule should flag use of the *set_input_transition* constraints.

By default, the *Inp_Trans01* and *Inp_Trans01a* rule reports the use of the *set_input_transition* constraint.

Used by	<i>Inp_Trans01a</i> , <i>Inp_Trans01</i>
Options	yes, 1; no, 0
Default	no; however, if the <i>chip</i> parameter is defined then the default value of this parameter is yes
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter allow_input_transition 1</code>
<i>Usage in goal/source files</i>	<code>-allow_input_transition=1</code>

allowed_load_cells

Specifies the comma-separated name list of load cells for the `load_of` command for the [Methodology Rules](#) rule.

Used by	Methodology Rules
Options	comma-separated list of load cells
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter allowed_load_cells 'cell_1'</code>
<i>Usage in goal/source files</i>	<code>-allowed_load_cells='cell_1'</code>

cg14_prefix

Specifies the permitted virtual clock signal name prefix for [Clk_Gen14](#) rule. The [Clk_Gen14](#) rule checks whether each virtual clock is named as per the correct syntax.

Used by	Clk_Gen14
Options	list of string values
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter cg14_prefix 'v1,v2,v3'</code>
<i>Usage in goal/source files</i>	<code>-cg14_prefix='v1,v2,v3'</code>

cg14_suffix

Specifies the permitted virtual clock signal name suffix for the [Clk_Gen14](#) rule. The [Clk_Gen14](#) rule checks whether each virtual clock is named as per the correct syntax.

Used by	Clk_Gen14
Options	list of string values
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter cg14_suffix 'v1,v2,v3'</code>
<i>Usage in goal/source files</i>	<code>-cg14_suffix='v1,v2,v3'</code>

check_clock_group_violations

Activates and deactivates checks performed by the SDC_288 rule.

Default is `no`. This indicates that checks are not performed by the SDC_288 rule and consequently makes the constraints parsing faster. However, not performing checks of the SDC_288 rule can lead to erroneous `set_clock_groups` commands remaining in the database. Set this parameter to `yes` to perform the SDC_288 rule checks.

Refer to the SDC_288 rule documentation in the *SpyGlass Built-In Rules Reference Guide* for more information about the violation.

Used by	<code>sdc_init_rule</code>
Options	<code>yes, no, ' '</code>
Default	<code>no</code>
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter check_clock_group_violations yes</code>
<i>Usage in goal/source files</i>	<code>-check_clock_group_violations=yes</code>

chip

Specifies the name of the design unit which corresponds to the chip-level. The value specified by the `chip` parameter is read by the [SDCPARSE](#) rule and propagated to other rules.

You can specify only those design unit names with the `chip` parameter that are reported to be top-level design units by the `DetectTopDesignUnits` rule (SpyGlass Built-in rule).

The following example sets the top-level module `mytop` to correspond to the chip-level:

```
set_parameter chip 'mytop'
```

Used by	Block06 , Block11 , Clk_Gen07 , Inp_Del03a , Inp_Del03b , Inp_Del05 , Inp_Trans01 , Inp_Trans01a , Inp_Trans09 , Op_Del03a , Op_Del03b , SDC_Methodology62 , SDC_Methodology63 , SDCPARSE
Options	a string value
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter chip 'mytop'</code>
<i>Usage in goal/source files</i>	<code>-chip='mytop'</code>

clk_gen_module

Specifies the names of special modules such as clock generators to be ignored by the [SDC_Methodology01](#) rule while checking for constraints set on non-leaf instance pins.

By default, the rules checks all design units.

Used by	SDC_Methodology01
Options	list of string value, including wildcards

Default	none
Example	
<i>Console/Tcl-based usage</i>	<pre>set_parameter clk_gen_module 'mymod1,mymod2' or set_parameter clk_gen_module 'mymod*'</pre>
<i>Usage in goal/source files</i>	<code>-clk_gen_module='mytop'</code>

clk_gen01_generate_report

Specifies whether the [Clk_Gen01a](#) and [Clk_Gen01b](#) rules should generate reports. The report is generated when the value of this parameter is set to `yes`.

Used by	Clk_Gen01a , Clk_Gen01b
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<pre>set_parameter clk_gen01_generate_report yes</pre>
<i>Usage in goal/source files</i>	<code>-clk_gen01_generate_report=yes</code>

clk_gen09_trav_over_sequential_arc

Specifies whether the [Clk_Gen09](#) rule traverses through a sequential arc while checking whether the source clock is in the fan-in of the generated clock.

Default is `yes`. Set this parameter to `no` to stop the traversal at a sequential arc while checking whether the source clock is in the fan-in of the generated clock.

Used by	Clk_Gen09
Options	yes, 1; no, 0
Default	yes
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter clk_gen09_trav_over_sequential_arc yes</code>
<i>Usage in goal/source files</i>	<code>-clk_gen09_trav_over_sequential_arc=yes</code>

default_max_capacitance

Specifies the default maximum load limit for design objects for the [Load02](#) rule.

By default, the [Load02](#) rule flags those design objects that have a load more than 2.000000. The value specified with the `default_max_capacitance` parameter should be consistent with maximum limit for capacitance specified for the technology library.

Used by	Load02a , Load02b
Options	a positive float value
Default	2.0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter default_max_capacitance '5.6'</code>
<i>Usage in goal/source files</i>	<code>-default_max_capacitance='5.6'</code>

default_max_transition

Specifies the default maximum transition value for the [Inp_Trans04](#), [Clk_Trans03](#), [Clk_Trans05](#), and [SDC_Methodology21](#) rules.

By default, the rules flag those objects that have a transition value more than 5.000000. The [Inp_Trans04](#), [Clk_Trans03](#), and [Clk_Trans05](#) rules decide the maximum clock transition value as follows:

- The `max_transition` value set for the corresponding pin in the associated technology library file (`.lib` file)
- Otherwise, the `default_max_transition` value set for the parent cell's library in the associated technology library file (`.lib` file)
- Otherwise, maximum of the `default_max_transition` values of all other associated technology library files (`.lib` files), if any
- Otherwise, the value set by the `default_max_transition` parameter

Used by	Inp_Trans04 , Clk_Trans03 , Clk_Trans05 , SDC_Methodology21
Options	a positive float value
Default	5.0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter default_max_transition '5.6'</code>
<i>Usage in goal/source files</i>	<code>-default_max_transition='5.6'</code>

default_min_capacitance

Specifies the default minimum load limit for design objects for the [Load02](#) rule.

By default, the [Load02](#) rule reads the default minimum capacitance from the specified target gate library. If no such value is found, the value of the `default_min_capacitance` parameter is used. By default the `default_min_capacitance` parameter is set to 0.0.

Used by	Load02a , Load02b
Options	a positive float value

Default	0.0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter default_min_capacitance '5.6'</code>
<i>Usage in goal/source files</i>	<code>-default_min_capacitance='5.6'</code>

default_min_transition

Specifies the default minimum transition value for the [Inp_Trans04](#), [Clk_Trans03](#), and [Clk_Trans05](#) rules.

By default, these rules consider the minimum transition values specified for the pins in the corresponding library. When this value is not found for a pin in the library, the rules assume the minimum transition value for the pin to be 0 unless overridden by the `default_min_transition` parameter.

These rules decide the minimum clock transition value as follows:

- The `min_transition` value set for the corresponding pin in the associated technology library file (.lib file)
- Otherwise, the value set by the `default_min_transition` parameter

Used by	Inp_Trans04 , Clk_Trans03 , Clk_Trans05
Options	a positive float value
Default	0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter default_min_transition '0.5'</code>
<i>Usage in goal/source files</i>	<code>-default_min_transition='0.5'</code>

del03_allow_setdelay

Specifies the [Inp_Del03b](#) and [Op_Del03a](#) rules ignore those port-clock pairs for which a `set_max_delay` constraint is defined.

By default, the `del03_allow_setdelay` parameter is not set and these rules check such port-clock pairs also.

Used by	Inp_Del03b , Op_Del03a
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter del03_allow_setdelay 1</code>
<i>Usage in goal/source files</i>	<code>-del03_allow_setdelay=1</code>

drive_cells_list

Specifies the comma-separated name list of drive cells for the [set_driving_cell](#) and [get_lib_cells](#) commands for the [SDC_Methodology21](#) rule.

Used by	SDC_Methodology21
Options	comma-separated list of drive cells
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter drive_cells_list 'drive_cell_name'</code>
<i>Usage in goal/source files</i>	<code>-drive_cells_list='drive_cell_name'</code>

equiv_sdc_ambiguous_clock_file

Clocks having same characteristics, applied on equivalent design objects, reaching same set of flip-flops, and/or with input/output delay specified on same set of input/output ports cannot be resolved and are considered to be ambiguous.

Used by	Equiv_SDC , Equiv_SDC_Top , Equiv_SDC_Block , Equiv_SDC_Dual_Design
Options	file name
Default	None
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_ambiguous_clock_file 'prop.txt'</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_ambiguous_clock_file='prop.txt'</code>

To resolve the ambiguity between such clocks, you need to specify the ambiguous clocks in a file containing a pair of equivalent clocks and pass this file as a value in the [equiv_sdc_ambiguous_clock_file](#) parameter.

The detailed format for mapping clocks is as follows:

```
<clockname1>[:<waveform-type>] | <clockname1>:<waveform-
type>[:<polarity>] <clockname2>[:<waveform-
type>] | <clockname2>:<waveform-type>[:<polarity>]
```

Each line should have a list of clock pairs. The description of the *waveform-type* and *polarity* fields is as follows:

- *waveform-type* describes waveform of the clock. This field is optional. If not specified, 'default' waveform is assumed. The valid values are as follows:
 - a. default_waveform
 - b. rise_triggered_high_pulse
 - c. fall_triggered_high_pulse
 - d. rise_triggered_low_pulse
 - e. fall_triggered_low_pulse

The `default_waveform` value corresponds to the clock waveform in the SDC file. All other waveforms correspond to the waveform specified through the `pulse` option of the `set_clock_sense/set_sense` constraint.

- `polarity` describes whether the waveform is positive or negative. This field is optional. If not specified, positive polarity is used by default. The values can either be positive or negative.

NOTE: *You need not specify all the clocks in the file. You can specify only those clocks in the file that have been declared by SpyGlass as ambiguous.*

Read the following examples for more details.

Example 1 - Default Clock Mapping

The following file snippet implies that `default_waveform`, positive polarity of Clock1 maps to `default_waveform`, positive polarity of Clock2.

```
--file begin--
Clock1 Clock2
--file end--
```

The default values of waveform and polarity are used for Clock1. If default values of `waveform-type` and `polarity` are used for the Clock1, this line applies to all waveform-type and polarity combinations of Clock1.

The above file snippet is equivalent to the following mapping file:

```
--file begin--
Clock1:default:positive Clock2:default:positive
Clock1:default:negative Clock2:default:negative

Clock1:rise_triggered_high_pulse:positive Clock2:rise_triggered_high_pulse:positive
Clock1:rise_triggered_high_pulse:negative Clock2:rise_triggered_high_pulse:negative

Clock1:fall_triggered_high_pulse:positive Clock2:fall_triggered_high_pulse:positive
Clock1:fall_triggered_high_pulse:negative Clock2:fall_triggered_high_pulse:negative

Clock1:rise_triggered_low_pulse:positive Clock2:rise_triggered_low_pulse:positive
Clock1:rise_triggered_low_pulse:negative Clock2:rise_triggered_low_pulse:negative

Clock1:fall_triggered_low_pulse:positive Clock2:fall_triggered_low_pulse:positive
```

```
Clock1:fall_triggered_low_pulse:negative Clock2:fall_triggered_low_pulse:negative
--file end--
```

Example 2 - Partial Short Form Clock Mapping

The following file snippet implies that `rise_triggered_high_pulse`, positive polarity of Clock1 maps to `rise_triggered_high_pulse`, positive polarity of Clock2. Since, the value of the `waveform-type` is non-default, this line applies to only positive polarity of `rise_triggered_high_pulse` of Clock1 and not to any other waveform or polarity combination.

```
--file begin--
Clock1:rise_triggered_high_pulse Clock2:rise_triggered_high_pulse:negative
--file end--
```

The above file snippet is equivalent to the following mapping file:

```
--file begin--
Clock1:rise_triggered_high_pulse:positive Clock2:rise_triggered_high_pulse:negative
--file end--
```

Example 3 - Overlapping Clock Mapping

The following file snippet content implies there is an existing mapping for a waveform, polarity of Clock1 and another mapping is also defined for it. The latter mapping overrides the earlier mapping. Therefore, in the following file, Clock1 maps to Clock3.

```
--file begin--
Clock1 Clock2
Clock1 Clock3
--file end--
```

The above file snippet is equivalent to the following mapping file:

```
--file begin--
Clock1:default:positive Clock3:default:positive
Clock1:default:negative Clock3:default:negative

Clock1:rise_triggered_high_pulse:positive Clock3:rise_triggered_high_pulse:positive
Clock1:rise_triggered_high_pulse:negative Clock3:rise_triggered_high_pulse:negative
```

Overview

```
Clock1:fall_triggered_high_pulse:positive Clock3:fall_triggered_high_pulse:positive
Clock1:fall_triggered_high_pulse:negative Clock3:fall_triggered_high_pulse:negative
```

```
Clock1:rise_triggered_low_pulse:positive Clock3:rise_triggered_low_pulse:positive
Clock1:rise_triggered_low_pulse:negative Clock3:rise_triggered_low_pulse:negative
```

```
Clock1:fall_triggered_low_pulse:positive Clock3:fall_triggered_low_pulse:positive
Clock1:fall_triggered_low_pulse:negative Clock3:fall_triggered_low_pulse:negative
--file end--
```

Example 4 - Partial Overlapping Clock Mapping

The following file snippet implies there exists a mapping for a waveform, polarity of Clock1 and another mapping is defined for some other waveform, polarity combination of the same clock. In this case, the latter statement overrides the earlier mapping.

```
--file begin--
Clock1 Clock2
Clock1:rise_triggered_high_pulse:positive Clock3
--file end--
```

The above file snippet is equivalent to the following mapping file:

```
--file begin--
Clock1:default:positive Clock2:default:positive
Clock1:default:negative Clock2:default:negative

Clock1:rise_triggered_high_pulse:positive Clock3:rise_triggered_high_pulse:positive
Clock1:rise_triggered_high_pulse:negative Clock2:rise_triggered_high_pulse:negative

Clock1:fall_triggered_high_pulse:positive Clock2:fall_triggered_high_pulse:positive
Clock1:fall_triggered_high_pulse:negative Clock2:fall_triggered_high_pulse:negative

Clock1:rise_triggered_low_pulse:positive Clock2:rise_triggered_low_pulse:positive
Clock1:rise_triggered_low_pulse:negative Clock2:rise_triggered_low_pulse:negative

Clock1:fall_triggered_low_pulse:positive Clock2:fall_triggered_low_pulse:positive
Clock1:fall_triggered_low_pulse:negative Clock2:fall_triggered_low_pulse:negative
--file end--
```

equiv_sdc_check_fanin

Checks fan-ins of the objects provided in the mapping file and whether the mapping file is incomplete.

Default is `no`. Set this parameter to `yes` to check the fan-ins of objects provided in the mapping file and whether the mapping file is incomplete.

Used by	Equiv_SDC_Dual_Design
Options	yes, 1; no, 0
Default	no
Example	
<i>Console-based usage</i>	<code>set_parameter equiv_sdc_check_fanin 1</code>
<i>Tcl-based usage</i>	<code>set_parameter equiv_sdc_check_fanin 1</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_check_fanin=1</code>

equiv_sdc_check_faninfanout_for_clocks

Checks fan-ins of the objects provided in the mapping file and whether the mapping file is incomplete.

Default is `no`. This indicates that the fan-in, fan-out relationship is not checked during clock equivalence. Set this parameter to `yes` to consider fan-in, fan-out relationships during clock equivalence. See [Example: Impact on Clock Equivalence](#).

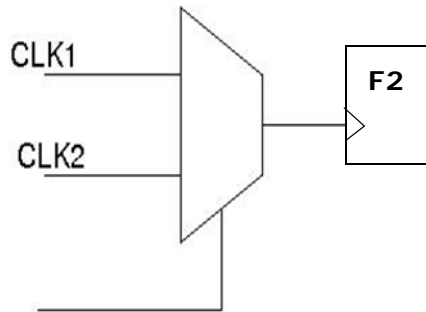
Used by	Equiv_SDC , Equiv_SDC_Block
Options	yes, 1; no, 0
Default	no

Example

<i>Console-based usage</i>	set_parameter equiv_sdc_check_fanin_fanout_for_clocks 1
<i>Tcl-based usage</i>	set_parameter equiv_sdc_check_fanin_fanout_for_clocks 1
<i>Usage in goal/ source files</i>	-equiv_sdc_check_fanin_fanout_for_clocks=1

Example: Impact on Clock Equivalence

To understand the impact on clock equivalence, consider the following design.



The SDC files are as follows:

M1.sdc

```
create_clock -name clk1 -period 10 clk1
create_clock -name clk2 -period 20 clk2
```

M2.sdc

```
create_clock -name clk1 -period 20 clk1
create_clock -name clk2 -period 10 clk2
```

After running the *Equiv_SDC* or *Equiv_SDC_Block*, the Equivalence Report is generated. The results reported for clock equivalence vary depending on the setting of this parameter.

If `equiv_sdc_check_fanin_fanout_for_clocks` is set to `no`, the following clock equivalence is reported in the Equivalence Report:

"Equivalent Clocks but differ in name and the object on which it is defined.",CLK1: period 20; CLK2: period 20,F2/CP,F2/CP

"Equivalent Clocks but differ in name and the object on which it is defined.",CLK2: period 10,CLK1: period 10,F2/CP,F2/CP

If `equiv_sdc_check_fanin_fanout_for_clocks` is set to `yes`, the following clock equivalence is reported in the Equivalence Report: :

"No equivalent clocks in implement mode.",CLK1: period 20,F2/CP,F2/CP

"No equivalent clocks in implement mode.",CLK2: period 10,F2/CP,F2/CP

"No equivalent clocks in reference mode.",CLK2: period 20,F2/CP,F2/CP

"No equivalent clocks in reference mode.",CLK1: period 10,F2/CP,F2/CP

equiv_sdc_clk_prefix

Specifies the clock name prefix for virtual clocks mapping in the Constraints Management rules. By default, no prefix is specified for virtual clocks.

Used by	<i>Equiv_SDC</i> , <i>Equiv_SDC_Block</i> , <i>Equiv_SDC_Dual_Design</i> , <i>Equiv_SDC_Top</i>
Options	list of string values
Default	unspecified
Example	

<i>Console-based usage</i>	<code>set_parameter equiv_sdc_clk_prefix v1,v2,v3</code>
<i>Tcl-based usage</i>	<code>set_parameter equiv_sdc_clk_prefix "v1,v2,v3"</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_clk_prefix=v1,v2,v3</code>

A virtual clock is associated with a real clock only when their characteristics are the same and the input/output delays specified by the virtual clock in one SDC file are also specified by the real clock in the other SDC file. However, you can directly associate a virtual clock to a real clock if a simple name mapping is present.

For example, if a real clock CLK associates to a virtual clock v_CLK, then by setting the value of the [equiv_sdc_clk_prefix](#) parameter to v_, you can directly map a real clock to a virtual clock.

equiv_sdc_clk_suffix

Specifies the clock name suffix for virtual clocks mapping in the Constraints Management rules. By default, no suffix is specified for virtual clocks.

Used by	Equiv_SDC , Equiv_SDC_Block , Equiv_SDC_Dual_Design , Equiv_SDC_Top
Options	list of string values
Default	unspecified
Example	
<i>Console-based/Tcl-based usage</i>	<code>set_parameter equiv_sdc_clk_suffix v1,v2,v3</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_clk_suffix=v1,v2,v3</code>

A virtual clock is associated with a real clock only when their characteristics are the same and the input/output delays specified by the virtual clock in one SDC file are also specified by the real clock in the other SDC file. However, you can directly associate a virtual clock to a real clock if a simple suffix-based name mapping is present.

For example, if a real clock CLK associates to a virtual clock CLK_IO, then by setting the value of the `equiv_sdc_clk_suffix` parameter to `_IO`, you can directly map a real clock to a virtual clock.

equiv_sdc_clock_precision

Specifies the clock period comparison to a specific number of decimal places in clock equivalence.

Default is 2 and the clock period comparison is till the second decimal place. You can set this parameter to any value between 1 and 4 to signify the decimal place for clock period comparison.

For example, suppose this parameter is set to 4. Clocks with period 1.2345 and 1.2346 are reported as **Inequivalent**. However, if this parameter is set to 3, these clocks are reported as **Equivalent**.

If the tolerance is set by `equiv_sdc_tolerance` parameter, the clock characteristics comparison is based only on tolerance value. Therefore, the `equiv_sdc_clock_precision` parameter does not have any impact on determining clock equivalence, if the `equiv_sdc_tolerance` parameter is set.

Used by	<i>Equiv_SDC, Equiv_SDC_Block, Equiv_SDC_Dual_Design, Equiv_SDC_Top, SDC_multimode_equiv</i>
Options	1, 2, 3, 4
Default	2
Example	
<i>Console-based/Tcl-based usage</i>	<code>set_parameter equiv_sdc_clock_precision 4</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_clock_precision=4</code>

equiv_sdc_constraint_file

Specifies the SDC commands that need to be checked for equivalence. The `equiv_sdc_constraint_file` parameter takes a file as its value.

Used by	<i>Equiv_SDC, Equiv_SDC_Top, Equiv_SDC_Block, Equiv_SDC_Dual_Design, SDC_multimode_equiv</i>
Options	file name
Default	none
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_constraint_file 'prop.txt'</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_constraint_file='prop.txt'</code>

The file should be in the following format:

<sdc-command-names>

The file can have one or more of the following commands.

TABLE 1 Supported SDC Commands

Command	Supported	Not supported
create_clock	•	
-period <i>period_value</i>	•	
[-name <i>clock_name</i>]	•	
[-waveform <i>edge_list</i>]	•	
[-add]	•	
[<i>port_pin_list</i>]	•	
create_generated_clock	•	

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported
<code>-source master_pin</code>	•	
<code>[-name clock_name]</code>	•	
<code>[-edges edge_list]</code>	•	
<code>[-divide_by factor]</code>	•	
<code>[-multiply_by factor]</code>	•	
<code>[-combinational]</code>	•	
<code>[-duty_cycle factor]</code>	•	
<code>[-invert]</code>	•	
<code>[-edge_shift shift_list]</code>	•	
<code>[-add]</code>	•	
<code>[-master_clock clock]</code>	•	
<code>[port_pin_list]</code>	•	
set_case_analysis	•	
<code>value_sca</code>	•	
<code>port_pin_list</code>	•	
set_clock_latency	•	
<code>delay</code>	•	
<code>object_list</code>	•	
<code>[-rise]</code>	•	
<code>[-fall]</code>	•	
<code>[-min]</code>	•	
<code>[-max]</code>	•	

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported
[-source]	•	
[-late]	•	
[-early]	•	
[-clock <i>clock</i>]		
set_clock_transition	•	
[-rise]	•	
[-fall]	•	
[-min]	•	
[-max]	•	
<i>transition</i>	•	
<i>clock_list</i>	•	
set_clock_uncertainty	•	
[-from <i>from_clock</i>]	•	
[-to <i>to_clock</i>]	•	
[-rise]	•	
[-rise_from <i>from_clock</i>]		•
[-rise_to <i>to_clock</i>]		•
[-fall]	•	
[-fall_from <i>from_clock</i>]		•
[-fall_to <i>to_clock</i>]		•
[-setup]	•	
[-hold]	•	

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported
<code>[-from_edge <i>edge_list</i>]</code>		•
<code>[-to_edge <i>edge_list</i>]</code>		•
<i>uncertainty</i>	•	
<code>[<i>object_list</i>]</code>	•	
set_dont_touch		
<code><i>object_list</i> [true false]</code>	•	
set_drive		
	•	
<code>[-rise -fall]</code>	•	
<code>[-min]</code>	•	
<code>[-max]</code>	•	
<i>resistance</i>	•	
<i>port_list</i>	•	
set_driving_cell		
	•	
<code>[-lib_cell <i>lib_cell_name</i>]</code>	•	
<code>[-rise]</code>	•	
<code>[-fall]</code>	•	
<code>[-max]</code>	•	
<code>[-min]</code>	•	
<code>[-library <i>lib_name</i>]</code>	•	
<code>[-pin <i>pin_name</i>]</code>	•	
<code>[-from_pin <i>from_pin_name</i>]</code>	•	

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported
<code>[-multiply_by <i>multiply_factor</i>]</code>	•	
<code>[-input_transition_rise <i>rise_time</i>]</code>	•	
<code>[-input_transition_fall <i>fall_time</i>]</code>	•	
<code>[-clock <i>clock_name</i>]</code>	•	
<code>[-clock_fall -clock <i>clock_name</i>]</code>	•	
<i>port_list</i>	•	
<code>[-dont_scale]</code>	•	
<code>[-no_design_rule]</code>	•	
set_false_path	•	
<code>[-setup]</code>	•	
<code>[-hold]</code>	•	
<code>[-rise]</code>	•	
<code>[-fall]</code>	•	
<code>[-from <i>from_list</i>]</code>	•	
<code>[-to <i>to_list</i>]</code>	•	
<code>[-through <i>through_list</i>]</code>	•	
<code>[-rise_from <i>from_list</i>]</code>		•
<code>[-fall_from <i>from_list</i>]</code>		•
<code>[-rise_to <i>to_list</i>]</code>	•	
<code>[-fall_to <i>to_list</i>]</code>	•	
<code>[-rise_through <i>through_list</i>]</code>		•
<code>[-fall_through <i>through_list</i>]</code>		•

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported
[-reset_path]		•
set_ideal_network		
[-dont_care_placement]		•
<i>port_pin_list</i>	•	
set_input_delay		
[-clock <i>clock_name</i>]	•	
[-clock_fall]	•	
[-level_sensitive]		•
[-rise]	•	
[-fall]	•	
[-max]	•	
[-min]	•	
[-add_delay]	•	
[-network_latency_included]		•
[-source_latency_included]		•
[-reference_pin <i>pin_name</i>]		•
<i>delay_value</i>	•	
<i>port_pin_list</i>	•	
set_input_transition		
[-rise]	•	
[-fall]	•	

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported
[-min]	•	
[-max]	•	
[-clock <i>clock_name</i>]	•	
[-clock_fall]	•	
<i>transition port_list</i>	•	
set_load	•	
[-min]	•	
[-max]	•	
[-fall]	•	
[-rise]	•	
[-subtract_pin_load]	•	
[-pin_load]	•	
[-wire_load]	•	
<i>value</i>	•	
<i>objects</i>	•	
set_max_capacitance	•	
[capacitance_value]	•	
[object_list]	•	
<i>value</i>	•	
<i>port_list</i>	•	

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported
set_min_delay	•	
[-rise]	•	
[-fall]	•	
[-from <i>from_list</i>]	•	
[-to <i>to_list</i>]	•	
[-through <i>through_list</i>]	•	
[-rise_from <i>from_list</i>]		•
[-fall_from <i>from_list</i>]		•
[-rise_to <i>to_list</i>]	•	
[-fall_to <i>to_list</i>]	•	
[-rise_through <i>through_list</i>]		•
[-fall_through <i>through_list</i>]		•
<i>delay_value</i>	•	
[-group_path <i>group_name</i>]		•
[-reset_path]		•
set_max_fanout	•	
<i>fanout_value</i>	•	
<i>object_list</i>	•	
set_max_transition	•	
<i>transition_value</i>	•	
<i>object_list</i>	•	

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported
<code>[-clock_path]</code>	•	
<code>[-data_path]</code>	•	
<code>[-rise]</code>	•	
<code>[-fall]</code>	•	
set_max_delay	•	
<code>[-rise]</code>	•	
<code>[-fall]</code>	•	
<code>[-from <i>from_list</i>]</code>	•	
<code>[-to <i>to_list</i>]</code>	•	
<code>[-through <i>through_list</i>]</code>	•	
<code>[-rise_from <i>from_list</i>]</code>		•
<code>[-fall_from <i>from_list</i>]</code>		•
<code>[-rise_to <i>to_list</i>]</code>	•	
<code>[-fall_to <i>to_list</i>]</code>	•	
<code>[-rise_through <i>through_list</i>]</code>		•
<code>[-fall_through <i>through_list</i>]</code>		•
<i>delay_value</i>	•	
<code>[-group_path <i>group_name</i>]</code>		•
<code>[-reset_path]</code>		•
set_multicycle_path	•	
<code>[-setup]</code>	•	

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported
<code>[-hold]</code>	•	
<code>[-rise]</code>	•	
<code>[-fall]</code>	•	
<code>[-start]</code>	•	
<code>[-end]</code>	•	
<code>[-from <i>from_list</i>]</code>	•	
<code>[-to <i>to_list</i>]</code>	•	
<code>[-through <i>through_list</i>]</code>	•	
<code>[-rise_from <i>from_list</i>]</code>		•
<code>[-fall_from <i>from_list</i>]</code>		•
<code>[-rise_to <i>to_list</i>]</code>	•	
<code>[-fall_to <i>to_list</i>]</code>	•	
<code>[-rise_through <i>through_list</i>]</code>		•
<code>[-fall_through <i>through_list</i>]</code>		•
<code><i>path_multiplier</i></code>	•	
<code>[-reset_path]</code>		•
set_output_delay	•	
<code>[-clock <i>clock_name</i>]</code>	•	
<code>[-clock_fall]</code>	•	
<code>[-level_sensitive]</code>		•
<code>[-rise]</code>	•	
<code>[-fall]</code>	•	

TABLE 1 Supported SDC Commands (Continued)

Command	Supported	Not supported
[-max]	•	
[-min]	•	
[-add_delay]	•	
[-network_latency_included]		•
[-source_latency_included]		•
[-reference_pin <i>pin_name</i>]		•
<i>delay_value</i>	•	
<i>port_pin_list</i>	•	
[-group_path <i>group_name</i>]		•
set_propagated_clock	•	
<i>object_list</i>	•	

If a [create_clock](#) or [create_generated_clock](#) command is specified in this file, then the [set_case_analysis](#), [create_clock](#) and [create_generated_clock](#) commands will be checked for equivalence.

If any of the [set_input_delay](#), [set_output_delay](#), [set_clock_latency](#), [set_clock_uncertainty](#), [set_clock_transition](#), [set_input_transition](#), or [set_load](#) commands are specified in the file, then the [set_case_analysis](#), [create_clock](#), and [create_generated_clock](#) and the specified commands will be checked for equivalence.

If any of the [set_false_path](#), [set_multicycle_path](#), or [set_max_delay/set_min_delay](#), commands are specified in the file, then the [set_case_analysis](#), [create_clock](#), [create_generated_clock](#), [set_input_delay](#), [set_output_delay](#) and the specified commands will be checked for equivalence.

If the `equiv_sdc_constraint_file` parameter is not specified, then

all commands will be checked for equivalence.

NOTE: *There are certain commands, such as [set_case_analysis](#) that are always checked for equivalency before the commands specified by you, as described in the [Understanding the Equivalence Flow](#) section. If these commands are not equivalent, the rules in the SpyGlass Constraints solution generate an error for these commands and equivalency check is not performed further. For example, if you have specified the `input delay` command in the `equiv_sdc_constraint_file` parameter, equivalency will first be performed for the [set_case_analysis](#) command. If the [set_case_analysis](#) commands are not equivalent then equivalency will not be performed further.*

equiv_sdc_copy_sca_sdc

Copies the SDC file containing the [set_case_analysis](#) propagated from Top to a directory.

The default value of this parameter is none. Set this parameter to the absolute directory path where you want to store the SDC file, which contains the [set_case_analysis](#) propagated from Top.

Prerequisite

To use this parameter, set the [equiv_sdc_import_sca_from_top](#) parameter to yes.

Used by	Equiv_SDC_Block
Options	an absolute directory path
Default	none
Example	
<i>Console-based usage</i>	<code>set_parameter equiv_sdc_copy_sca_sdc /home/somename</code>
<i>Tcl-based usage</i>	<code>set_parameter equiv_sdc_copy_sca_sdc /home/somename</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_copy_sca_sdc=/home/somename</code>

equiv_sdc_design_equivalence_file

Specifies the design equivalence file that provides a report to map the ports, registers, and any intermediate design object used in the SDC files of one design to another for the [Equiv_SDC_Dual_Design](#) and [Equiv_SDC_Auto_Detect](#) rules.

The [Equiv_SDC_Auto_Detect](#) rule generates the design equivalence file in the compressed format, if the file size is greater than 1 Gb. You can specify this file in the tar.gz compressed form.

Without the file, the [Equiv_SDC_Dual_Design](#) rule may not work correctly.

Used by	Equiv_SDC_Dual_Design , Equiv_SDC_Auto_Detect
Options	a string value
Default	none
Example	
<i>Console-based usage</i>	<code>set_parameter equiv_sdc_design_equivalence_file prop.txt</code>
<i>Tcl-based usage</i>	<code>set_parameter equiv_sdc_design_equivalence_file "prop.txt"</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_design_equivalence_file=prop.txt</code>

For more information on how to specify a mapping file, see the following sections:

- [Mapping File Structure](#)
- [Example of a Mapping File](#)
- [Specifying Non-Sequential Instances](#)
- [Specifying One-to-Many Mapping](#)

Mapping File Structure

SpyGlass parses the mapping file for 'r: <>' and 'i: <>' string patterns and will ignore other data, such as comment lines, in the file. In addition, the

file should be a UNIX text file.

To specify reference design object, the mapping file should contain the following pattern:

```
^(.*[ \t])?r:<hierarchical_name>
```

Where, the `hierarchical_name` should include the name of the design.

To specify implement design object, the mapping file should contain the following pattern:

```
^(.*[ \t])?i:<hierarchical_name>
```

Where, the `hierarchical_name` should include the name of the design.

It is not necessary for the `hierarchical_name` to include escape sequences, even if the `hierarchical_name` contains special characters.

To specify a comment in the mapping file, begin the line with the '#' character.

Example of a Mapping File

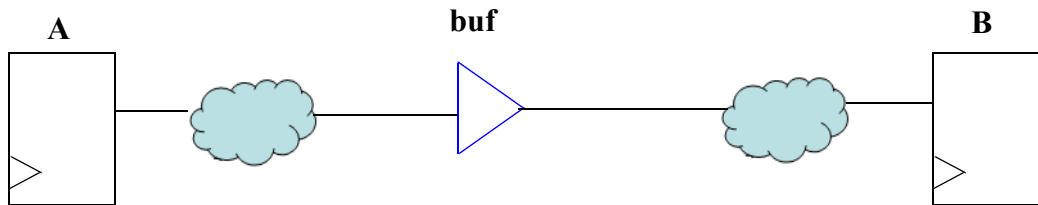
```
Ref DFF                Name(Last) r:top/out_reg
Impl DFF              Name(Last) i:top/out_reg

# Following two lines are commented and ignored
# Ref Port            Name(Last) r:top/in
# Impl Port           Name(Last) i:top/in

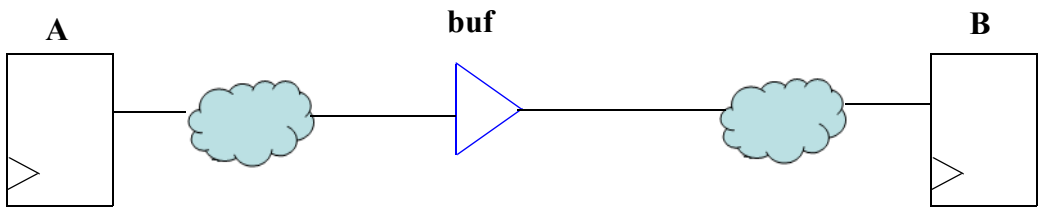
# We can have lines beginning with 'r:' or 'i:'
r:top/clock
i:top/clock
```

Specifying Non-Sequential Instances

At times, you may need to specify non-sequential instances in the mapping file. Consider the following example:



Reference Design



Implement Design

Let us assume that a `set_false_path` command is specified as follows:

```
set_false_path -from A -through buf -to B (reference)
```

```
set_false_path -from A -through buf -to B (implement)
```

If a mapping is provided for A and B points only, then the tool considers the instance `buf` of the reference design as different from the instance `buf` of the implement design. In such a case, the tool reports the following two messages:

A false path -from A -through buf -to B is present only in the reference design.

A false path -from A -through buf -to B is present only in the implement design.

If a mapping is provided for `buf`, then the `buf` instances of the reference and the implement designs are considered equivalent. In such a case, the tool considers the two commands as equivalent and does not report any

message. You can provide mapping of the buf instance as:

```
Ref DFF r: top/buf
```

```
Impl DFF i: top/buf
```

To map non-sequential instances for the two equivalent designs, you need to use the following syntax in the mapping file that needs to be passed as an argument to the *equiv_sdc_design_equivalence_file* parameter:

```
r:<the hierarchical path for the instance in the reference design>
```

```
i:<the hierarchical path for the instance in the implement design>
```

As shown below:

```
Ref BlPin r: top/MUX1/Z
```

```
Impl BlPin i: top/AND1/Z
```

Specifying One-to-Many Mapping

This section shows you how to specify the mapping file when one design object (port or pin) in reference design is equivalent to multiple objects in implement design.

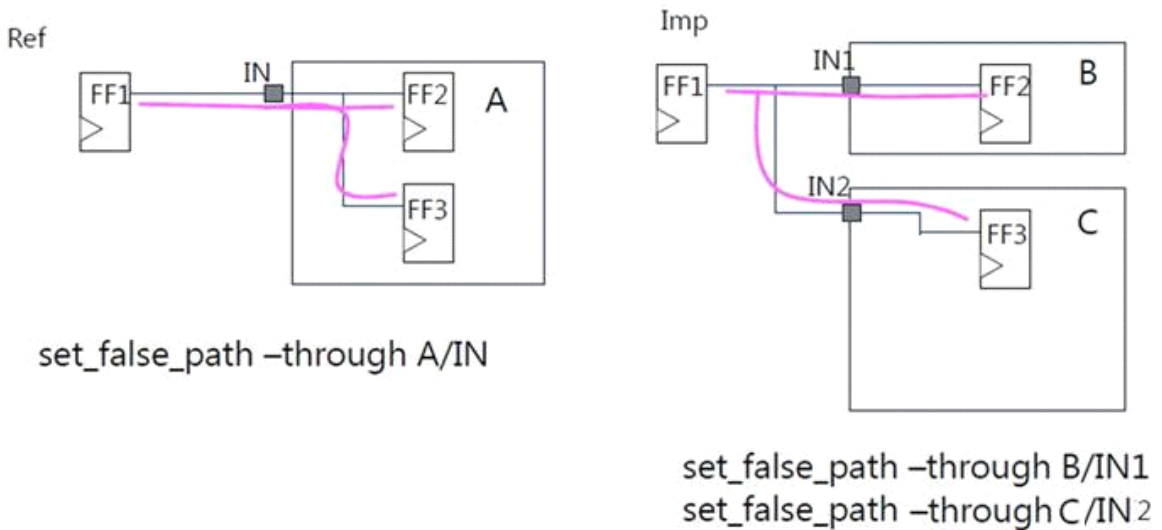
The syntax of the mapping file is as follows:

```
r: <ref port/pin>
```

```
i: <imp port/pin>, <imp port/pin2>, <imp port/pin3>, ...  
#multiple objects by ',' separated list
```

NOTE: *You can specify only ports and pins one-to-many mapping. Instances cannot be specified.*

For example, consider the following scenario, where A/IN is equivalent to B/IN1, C/IN2 pins, and timing exceptions given in reference design, should be equivalent to Exceptions given in implement design.



The mapping file would be specified as:

r: A/IN

i: B/IN1, C/IN2 #1-to-many mapping

NOTE: To enable one-to-many mapping, set the environment variable "SDC_EQUIV_ONE_TO_MANY" to 1 using the "setenv SDC_EQUIV_ONE_TO_MANY 1" command in the shell, before running SpyGlass.

equiv_sdc_design_equivalence_hier_file

Specifies the design equivalence file that provides a report to map the hierarchy used in the [Equiv_SDC_Auto_Detect](#) rule.

When you use this parameter, do not specify the [equiv_sdc_design_equivalence_file](#) parameter.

Default is none. Set this parameter to the path of the hierarchical mapping

file.

Used by	Equiv_SDC_Auto_Detect
Options	a string value
Default	none
Example	
<i>Console-based usage</i>	set_parameter equiv_sdc_design_equivalence_hier_file prop.txt
<i>Tcl-based usage</i>	set_parameter equiv_sdc_design_equivalence_hier_file "prop.txt"
<i>Usage in goal/source files</i>	-equiv_sdc_design_equivalence_hier_file=prop.txt

SpyGlass parses the mapping file for 'r: <>' and 'i: <>' string patterns and will ignore other data, such as comment lines, in the file. In addition, the file should be a UNIX text file.

To specify reference design object, the mapping file should contain the following pattern:

```
^(.*[ \t])?r:<hierarchical_name>
```

Where, the `hierarchical_name` should include the name of the design.

To specify implement design object, the mapping file should contain the following pattern:

```
^(.*[ \t])?i:<hierarchical_name>
```

Where, the `hierarchical_name` should include the name of the design. It is not necessary for the `hierarchical_name` to include escape sequences, even if the `hierarchical_name` contains special characters.

To specify a comment in the mapping file, begin the line with the '#' character.

Example of a Hierarchical Mapping File

```
Ref DFF          Name(Last) r:top/A/B
```

```

Impl DFF                Name(Last) i:top/A/B/C

# Following two lines are commented and ignored
# Ref Port              Name(Last) r:top/A
# Impl Port             Name(Last) i:top/B/C

```

equiv_sdc_design_ignore_prefix

At times, the formality report generated by tools may contain some prefix (like WORK) added to the port/register name. This prefix may not be inferred correctly by the equivalent design and therefore needs to be ignored. The `equiv_sdc_design_ignore_prefix` rule parameter is used by the [Equiv_SDC_Dual_Design](#) and [Equiv_SDC_Auto_Detect](#) rules to ignore the prefix of the design hierarchy in the equivalent design file.

Used by	Equiv_SDC_Dual_Design , Equiv_SDC_Auto_Detect
Options	a string value
Default	none
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_design_ignore_prefix '/WORK/'</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_design_ignore_prefix='/WORK/'</code>

Example:

In this example, for the Ref port, WORK will be ignored and it will match with the Impl port. The parameter is set as follows:

```
set_parameter equiv_sdc_design_ignore_prefix 'prop.txt'
```

The prop.txt file contains the following:

```
Ref Port      Name(Last) r:WORK/top/in1
Impl Port     Name(Last) i:top/in1
```

equiv_sdc_enable_one_pass

Enables the one-pass flow for dual-design equivalence.

Default is no. Set this parameter to yes enable the one-pass flow for dual design equivalence.

Used by	Equiv_SDC_Auto_Detect
Options	no, 0; yes, 1
Default	no
Example	
<i>Console/Tcl-based usage</i>	set_parameter equiv_sdc_enable_one_pass 1
<i>Usage in goal/source files</i>	-equiv_sdc_enable_one_pass=1

equiv_sdc_disambiguate_same_names_clock

Used to automatically resolve ambiguity of clocks on the basis of name.

Typically, to resolve the ambiguity between clocks, you need to specify the ambiguous clocks in a file containing a pair of equivalent clocks and pass this file as a value in the [equiv_sdc_ambiguous_clock_file](#) parameter.

When you set this parameter to yes, you need not specify the ambiguous clocks which have the same name, through the [equiv_sdc_ambiguous_clock_file](#) parameter.

By default, this parameter is set to no.

Used by	Equiv_SDC_Dual_Design , Equiv_SDC_Block , Equiv_SDC
Options	no, 0; yes, 1
Default	no
Example	
<i>Console/Tcl-based usage</i>	set_parameter equiv_sdc_disambiguate_same_names_clock 1
<i>Usage in goal/source files</i>	-equiv_sdc_disambiguate_same_names_clock=1

equiv_sdc_fast_exceptions_equivalence

Enables a new algorithm for [Equiv_SDC](#) and [Equiv_SDC_Dual_Design](#) timing exception equivalence.

Default is no. Set this parameter to yes to enable the new equivalence check algorithm. This algorithm provides runtime improvement for timing exception equivalence.

When you use this parameter, the following parameters are not supported:

- [equiv_sdc_check_fanin](#)
- [equiv_sdc_show_violations](#)
- [tc_lvpc](#)
- [tc_report_backref_all](#)
- [tc_report_verbos](#)
- [tc_regression_mode](#)

Used by	Equiv_SDC_Dual_Design , Equiv_SDC
Options	no, 0; yes, 1
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_fast_exceptions_equivalence 1</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_fast_exceptions_equivalence=1</code>

equiv_sdc_ignore_constant_pins_for_top_block_equivalence

Sometimes while doing the top block equivalence of `set_case_analysis`, you might want to ensure that the constant power pins (VSS/VDD) outside the block are not considered. In such a case, set this parameter to `yes`.

Used by	Equiv_SDC_Block , Equiv_SDC_Top
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_ignore_constant_pins_for_top_block_equivalence 1</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_ignore_constant_pins_for_top_block_equivalence=1</code>

equiv_sdc_ignore_unequivalent_design_obj

Sometimes the formality report might not be able to establish equivalence between all ports and registers. This may lead to incorrect violations as in certain cases, equivalence is established based on ports and registers.

The `equiv_sdc_ignore_unequivalent_design_obj` parameter specifies whether the [Equiv_SDC_Dual_Design](#) rule ignores those design objects whose equivalent design object is missing in the equivalence report.

Used by	Equiv_SDC_Dual_Design
---------	---------------------------------------

Options	yes, 1; no, 0
---------	---------------

Default	yes
---------	-----

Example

<i>Console/ Tcl-based usage</i>	<code>set_parameter equiv_sdc_ignore_unequivalent_design_obj 1</code>
---	---

<i>Usage in goal/ source files</i>	<code>-equiv_sdc_ignore_unequivalent_design_obj=1</code>
--	--

equiv_sdc_ignore_sca_for_top_block_equivalence

Controls whether the messages of type "specified only in top" are displayed in the [Equivalence Report](#) for top-block equivalence check.

By default, the value of this parameter is to `no` and the messages are waived. Set this parameter to `yes` to not display messages of type "specified only in top" when performing the top-block equivalence check.

Used by	Equiv_SDC_Block
---------	---------------------------------

Options	yes, 1; no, 0
---------	---------------

Default	no
Example	
<i>Console/ Tcl-based usage</i>	set_parameter equiv_sdc_ignore_sca_for_top_block_equivalence 1
<i>Usage in goal/ source files</i>	-equiv_sdc_ignore_sca_for_top_block_equivalence=1

equiv_sdc_ignore_value

Specifies whether to ignore the value between two commands while performing an equivalency check.

The `equiv_sdc_ignore_value` parameter only performs option-to-option comparison. For example, if a command has only the `-rise` option in reference and only `-fall` option in implement, the parameter does not have any effect on the command.

Used by	Equiv_SDC_Dual_Design , Equiv_SDC_Block , Equiv_SDC_Top , Equiv_SDC , SDC_multimode_equiv
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl- based usage</i>	set_parameter equiv_sdc_ignore_value 1
<i>Usage in goal/source files</i>	-equiv_sdc_ignore_value=1

equiv_sdc_import_sca_from_top

For many blocks (especially IPs) constant values are not provided in the Block SDCs per se as they are customizable from TOP depending on the mode of operation. In such scenarios, constant values gets propagated from Top-to-Block but as the same constant value is not specified in the Block SDC, the [Equiv_SDC_Block](#) rule reports a violation for [set_case_analysis](#) that exist only at Top.

The [equiv_sdc_ignore_sca_for_top_block_equivalence](#) parameter is used to waive of such violation messages but the parameter does not affect the propagation of other constraints accordingly.

By using the `equiv_sdc_import_sca_from_top` parameter, you can import such constant values from the Top SDC file and apply them on Block ports. Consequently, "specified only in top" type of [set_case_analysis](#) violations are reported as equivalent and the effect on other constraints are handled accordingly.

By default, the value of this parameter is `no`. Set this parameter to `yes` to import constant values from the Top SDC file and apply them on Block ports.

Used by	Equiv_SDC_Block
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_import_sca_from_top 1</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_import_sca_from_top=1</code>

Also, see [equiv_sdc_copy_sca_sdc](#).

equiv_sdc_object_name_length

Controls the length of block instance names shown in the worksheet title of

the generated spreadsheet.

Default is 12. This means that 12 characters are displayed in the worksheet title bar and any longer block instance name is truncated to 12 characters. Set the value to 0 to remove the length limit. Alternatively, set this parameter to a positive integer value greater than 2 to set the length.

Used by	Equiv_SDC_Block
Options	0; positive integer value greater than 2
Default	12
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_object_name_length 9</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_object_name_length=9</code>

equiv_sdc_report_blocked_clocks

Reports equivalence/inequivalence of clocks that are blocked.

Blocked clocks are the clocks that are not reaching any clock pin of a sequential element.

Since blocked clocks do not impact clock equivalence, by default SpyGlass Constraints does not report them in the [Equivalence Report](#). However, if you want all the clocks to be reported in the Equivalence Report, set this parameter to `yes`.

The incremental flow of equivalence checking is not impacted by the inequivalence of blocked clocks. The equivalence check for further stages is performed even if such blocked clocks are reported as inequivalent.

Used by	Equiv_SDC_Block , Equiv_SDC
Options	yes, 1; no, 0
Default	no

Example

<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_report_blocked_clocks 1</code>
--------------------------------	--

<i>Usage in goal/source files</i>	<code>-equiv_sdc_report_blocked_clocks=1</code>
-----------------------------------	---

equiv_sdc_report_type

Both equivalent and inequivalent SDC constraints are reported in the spreadsheets generated. Use this parameter to report only equivalent or only inequivalent constraints.

Default is `both`. This means that both equivalent and inequivalent constraints are reported. Set this parameter to `equiv_only` to report only equivalent constraints and `inequiv_only` to report inequivalent constraints only.

Used by	Equiv_SDC_Block , Equiv_SDC , Equiv_SDC_Dual_Design
---------	---

Options	<code>equiv_only</code> ; <code>inequiv_only</code> ; <code>both</code>
---------	---

Default	<code>both</code>
---------	-------------------

Example

<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_report_type inequiv_only</code>
--------------------------------	---

<i>Usage in goal/source files</i>	<code>-equiv_sdc_report_type=inequiv_only</code>
-----------------------------------	--

equiv_sdc_script_file

Specifies the mapping script file for the one-pass flow.

Default is `none`. Set this parameter to the file path of the mapping script file for the one-pass flow.

Used by	Equiv_SDC_Auto_Detect
Options	string value
Default	none
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_script_file 'sort_mapping_file.txt'</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_script_file='sort_mapping_file.txt'</code>

equiv_sdc_script_side_file

Specifies the side file, which is used as an input to the script file specified using the [equiv_sdc_script_file](#) parameter.

Default is none. Set this parameter to the file path of the side file used in the one-pass flow.

Used by	Equiv_SDC_Auto_Detect
Options	string value
Default	none
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_script_side_file 'sidefile.txt'</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_script_side_file='sidefile.txt'</code>

equiv_sdc_show_violations

Shows the violation messages generated by rules that check for equivalence. By default, the messages are suppressed and visible only by accessing the [Equivalence Report](#). Set the value to `yes` to make the messages visible in the message tree.

Used by	Equiv_SDC_Dual_Design , Equiv_SDC_Block , Equiv_SDC_Top , Equiv_SDC
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_show_violations 1</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_show_violations=1</code>

equiv_sdc_set_tolerance

Default is `no` and the tolerance value is determined by the value of the [equiv_sdc_tolerance](#) parameter. Set this parameter to `yes` to use the delay/clock_period ratio as the tolerance value.

Used by	SDC_multimode_equiv
Options	yes, 1; no, 0
Default	0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_set_tolerance yes</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_set_tolerance=yes</code>

equiv_sdc_tolerance

Two commands are considered equivalent if their values are exactly the same. However, you may want to specify a limit (tolerance value) within which the commands are considered equivalent.

The `equiv_sdc_tolerance` parameter is used to specify the tolerance value in percentage while checking for equivalency between two commands. If the difference in value between the two commands is within the tolerance value, then the rules in the SpyGlass Constraints solution do not report a violation and display the following message:

[INFO] Tolerance value of <tolerance-value> used for checking equivalence

Used by	Equiv_SDC_Dual_Design , Equiv_SDC_Block , Equiv_SDC_Top , Equiv_SDC , SDC_multimode_equiv
Options	float 0-100
Default	0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter equiv_sdc_tolerance 5</code>
<i>Usage in goal/source files</i>	<code>-equiv_sdc_tolerance=5</code>

Example

Consider the following SDC commands:

```
create_clock -name CLK1 -period 10 [get_ports {clk}]
```

```
create_clock -name CLK1 -period 10.5 [get_ports {clk}]
```

The two commands are not equivalent as their clock period is not the same. This should result in a violation. However, if you set the value of the `equiv_sdc_tolerance` parameter to 10, the violation is not reported.

gen_sdc_constraints_file

Specifies the file containing names of the constraints to be generated by the *SDC_GenerateIncr* rule.

By default, only the *set_case_analysis*, *create_clock*, and *create_generated_clock* constraints are enabled.

You can specify your own list of additional constraints to be generated by listing the required constraints in an ASCII file (one constraint name per line) and specifying it using the *gen_sdc_constraints_file* parameter.

You can comment a line in the file by placing “#” or “//” at the start of the line.

Used by	<i>SDC_GenerateIncr</i>
Options	a string value
Default	“gensdcConstraintsFile.txt” lying in the SpyGlass Constraints installation directory
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter gen_sdc_constraints_file 'prop.txt'</code>
<i>Usage in goal/source files</i>	<code>-gen_sdc_constraints_file='prop.txt'</code>

gen_sdc_merge_seed

Generates a merged SDC, which contains seed and generated constraints. When using a parameterized flow, it also generates a parameter file that contains the associated parameters.

Used by	<i>SDC_GenerateIncr</i>
Options	yes, 1; no, 0
Default	no

Example

<i>Console/Tcl-based usage</i>	<code>set_parameter gen_sdc_merge_seed 1</code>
--------------------------------	---

<i>Usage in goal/source files</i>	<code>-gen_sdc_merge_seed=1</code>
-----------------------------------	------------------------------------

gen_sdc_csv2sdc_tool

Invokes an internal tool which reads SDC constraints in CSV format and outputs the same in SDC format.

Used by	SDC_GenerateIncr
Options	a colon separated 2 string values, one defining CSV file path and other defining output SDC file path respectively.
Default	None

Example

<i>Console/Tcl-based usage</i>	<code>set_parameter gen_sdc_csv2sdc_tool 'my.csv:output.sdc'</code>
--------------------------------	---

<i>Usage in goal/source files</i>	<code>-gen_sdc_csv2sdc_tool='my.csv:output.sdc'</code>
-----------------------------------	--

gen_sdc_param_file

Specifies the file that contains the format for the parameters that are generated to be used by the [SDC_GenerateIncr](#) rule. The [SDC_GenerateIncr](#) rule then replaces the value of the constraints in the seed file with the appropriate parameters.

By default, the parameters are not generated.

Used by	SDC_GenerateIncr
Options	a string value
Default	"gensdcParamSdc.txt" lying in the SpyGlass Constraints installation directory
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter gen_sdc_param_file 'prop.txt'</code>
<i>Usage in goal/source files</i>	<code>-gen_sdc_param_file='prop.txt'</code>

The file that the `gen_sdc_param_file` parameter accepts as a value follows a naming convention which is described in the following section.

Naming Convention for the File

The format in which the parameter is described in the naming convention file depends on the constraint. A constraint may contain the following fields and the parameters for the constraint can be defined accordingly:

- **Source Fields:** These are fields that do not require any timing value. These fields can be used to construct the parameter names. For example, the design object field is a source field for many constraints.
- **Phase Fields:** These fields that determine the scope of a constraint. These fields can further be classified as follows:
 - **Constraints Phase Fields:** These fields describe the phase of the constraint. For example, `min/max/rise/fall` and so on. If the naming convention for a constraint, say `set_input_delay` is defined without mentioning its constraints phase fields, then a single parameter is generated for all the leaf-level phase combinations. For example, the leaf-level phase combination for the `set_input_delay` constraint are `-min -rise`, `-min -fall`, `-max -rise`, and `-max -fall`. If an existing constraint is not defined for all leaf-level phase combinations, then the naming convention does not match the constraint and the parameters are not generated.
 - **Field Phase Fields:** Some constraints phase fields may have sub-fields. This fields determines the sub-fields of the constraints phase field. For example, in `set_input_delay/set_output_delay` constraints,

`clock_fall` describes the clock edge. Therefore, it is a sub-field of `clock`.

NOTE: *In case, the naming convention is defined for only one of the possible values of the field, new constraints are generated with that option. However, if the naming convention does not refer to the field, new constraints are generated with the default value. For example, if the naming convention for `set_input_delay` refers to the `clock_fall` field, new constraints shall be generated with the value of `clock_fall` set to `yes`. If the naming convention does not refer to the `clock_fall` field, the new constraints are generated with `clock_fall` set to its default value `no`.*

■ **Value Fields:** These are the fields for which the parameter can be defined. Such fields require timing values from you. The parameter names for such fields are constructed using the Source and Phase fields. There are two types of such fields:

□ **Compulsory Value Fields:** These are mandatory fields for which timing values must be defined. For example, specifying the period and waveform is mandatory for the `create_clock` constraint and specifying the delay value is mandatory for the `set_input_delay` and `set_output_delay` constraints.

□ **Optional Value Fields:** These are optional fields for which the values may/may not be defined. For example, the `input_transition_rise` and `input_transition_fall` fields of the `set_driving_cell` constraint

■ **Related Value Fields:** There are some fields in the constraints that are mutually exclusive. This means that you can specify only one field out of the other fields. For example, the `divide_by`, `multiply_by` and `edge_list` fields are mutually exclusive in the `create_generated_clock` constraint.

NOTE: *In case, the naming convention for a constraint is not defined or if it is defined for multiple related value fields, a single value field among related value fields will be generated depending on the priority list defined.*

The following is an example of a naming convention file:

```
# Naming Convention for specifying set_input_delay/  
set_output_delay constraints  
  
set_input_delay %objectlist.0.objname_%clock.objname_id_mnr  
-min -rise
```

```

set_input_delay %objectlist.0.objname_%clock.objname_id_mnf
-min -fall

set_input_delay %objectlist.0.objname_%clock.objname_id_mxr
-max -rise

set_input_delay %objectlist.0.objname_%clock.objname_id_mxf
-max -fall

set_output_delay %objectlist.0.objname_%clock.objname_od_mnr
-min -rise

set_output_delay %objectlist.0.objname_%clock.objname_od_mnf
-min -fall

set_output_delay %objectlist.0.objname_%clock.objname_od_mxr
-max -rise

set_output_delay %objectlist.0.objname_%clock.objname_od_mxf
-max -fall

set_case_analysis %objectlist.0.objname_sca

```

NOTE: *The above example displays the naming convention for specifying the `set_input_delay` and `set_output_delay` constraints only. For details on naming convention for other constraints, refer to the `gensdcParamSdc.txt` file located at `<your-inst-directory>/SPYGLASS_HOME/constraints` directory.*

In the naming convention file:

- White spaces and lines beginning with a # and // are treated as comments and hence ignored
- The given naming convention file is assumed to be lexically, syntactically, and semantically valid

Support for set and source Tcl Commands

The SPNC format supports set and source TCL commands. You can define a global variable in the SPNC file by using the set command. The following example shows how you can use the defined variable as a parameter or in parameter value expressions in the SPNC file.

```

----SPNC Begin----
set global_var1 2.2

```

```
set global_var2 3.3
set custom_file_path $env(MY_ENV_VAR)
set_input_delay global_var1
set %objectlist.0.objname_%clock.objname_od [expr
$global_var1*$global_var2]
set_output_delay %objectlist.0.objname_%clock.objname_od
----SPNC End----
```

You can parameterize the value in the SPNC file by referring to an environment variable. In the above example, the `custom_file_path` variable is using an environment variable `MY_ENV_VAR`. For example, the `SPYGLASS_HOME` variable is usually set in the shell environment and you can this variable in the SPNC file.

When a variable with the name `period` is declared in the SPNC file, its value is set to the minimum period of all the real primary clocks in the seed file. However, if there are no real primary clocks, the value is not modified.

Support of Technology Variables

You can group related global variables into a file and then refer to them by using the `source` command in the SPNC file. For example, you can group `global_var1` and `global_var2` variables into a file called `global_vars.tcl`. The following example shows how you can refer to them in the SPNC file by using the `source` command.

```
source 'global_vars.tcl'
```

A group of variables may have different values based on the technology platform on which the design is fabricated. Suppose, you want to 'source' the technology file containing the variables and use them in the SPNC file. To enable this, the following 'special' variables are adopted in the SPNC file.

```
'tech', 'tech_files_location', 'valid_techs'
```

where, `<tech_files_location>` refers to location where technology files are kept, `<valid_techs>` are the comma-separated list of valid technologies, and `<tech>` refers to the technology to be used.

The technology file name convention is `'$tech_files_location/$tech.tcl'`. The following `source` command is generated into the generated SDC.

```
source <technology filename>
```

Support of Conditional Value Expressions

There is support for conditional value expressions. This support provided by the `?:` ternary operator. For example, in the following example, the SPNC specifies that the input delay generated parameter's value is dependent on the clock's period. If the clock period is less than 15, then the value should be \$a, else the value is \$b.

```
----SPNC Begin----
set a 3
set b 7
set %objectlist.0.objname_%clock.objname_id %clock_period <
15 ? $a : $b
set_input_delay %objectlist.0.objname_%clock.objname_id
----SPNC End----
```

gen_sdc_user_param_file

Overrides the default SPNC configuration as specified through the [gen_sdc_param_file](#) parameter.

Used by	SDC_GenerateIncr
Options	A string value
Default	none
Example	
<i>Console-based/Tcl-based usage</i>	set_parameter gen_sdc_user_param_file 'prop.txt'
<i>Usage in goal/source files</i>	-gen_sdc_user_param_file='prop.txt'

Example

Depending on how you use the [gen_sdc_param_file](#) parameter, it may

contain overriding/custom SPNC specifications. Suppose, IO delays are specified to be partitioned into min/max through the SPNC specification in [gen_sdc_param_file](#).

```
----SPNC Begin----
set_input_delay %objectlist.0.objname_%clock.objname_%opt_id
-min -opt mn
set_input_delay %objectlist.0.objname_%clock.objname_%opt_id
-max -opt mx
set_output_delay
%objectlist.0.objname_%clock.objname_%opt_od -min -opt mn
set_output_delay
%objectlist.0.objname_%clock.objname_%opt_od -max -opt mx
----SPNC End----
```

However, you may want to partition IO delays in terms of rise/fall and not change anything else like the parameter name conventions. The following example shows how you can specify an SPNC specification through [gen_sdc_user_param_file](#).

```
----SPNC Begin----
set_input_delay -rise -opt r
set_input_delay -fall -opt f
set_output_delay -rise -opt r
set_output_delay -fall -opt f
----SPNC End----
```

It will override the earlier specification and generate the final SPNC, as illustrated by the following example.

```
----SPNC Begin----
set_input_delay %objectlist.0.objname_%clock.objname_r_id -
rise
set_input_delay %objectlist.0.objname_%clock.objname_f_id -
fall
set_output_delay %objectlist.0.objname_%clock.objname_r_od -
```

```

rise
set_output_delay %objectlist.0.objname_%clock.objname_f_od -
fall
----SPNC End----

```

gen_c2cVerify

Set this parameter to `yes` to invoke the SpyGlass TXV product flow for formal verification of clock-to-clock paths.

Used by	SDC_GenerateIncr
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter gen_c2cVerify no</code>
<i>Usage in goal/source files</i>	<code>-gen_c2cVerify=no</code>

gen_sdc_phase

Specifies the design phase for which the [SDC_GenerateIncr](#) rule should generate the template constraints file.

By default, these ruled generate the template constraints file for the RTL design phase.

Used by	SDC_GenerateIncr
Options	RTL, PRELAYOUT, POSTLAYOUT
Default	RTL

Example

<i>Console/Tcl-based usage</i>	<code>set_parameter gen_sdc_phase 'POSTLAYOUT'</code>
<i>Usage in goal/source files</i>	<code>-gen_sdc_phase='POSTLAYOUT'</code>

ignore_incremental_equivalence

The equivalence of commands is done incrementally. If the equivalence of commands with high precedence cannot be established, then SpyGlass will not check other commands. The precedence is as follows:

1. `set_case_analysis`
2. `create_clock/create_generated_clock`
3. `set_input_delay/set_output_delay`

When the `ignore_incremental_equivalence` parameter is set to `yes`, SpyGlass also checks the equivalence of other commands even if equivalence of commands with high precedence cannot be established.

Used by	Equiv_SDC_Dual_Design , Equiv_SDC_Block , Equiv_SDC_Top , Equiv_SDC , SDC_multimode_equiv
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter ignore_incremental_equivalence 1</code>
<i>Usage in goal/source files</i>	<code>-ignore_incremental_equivalence=1</code>

For more information on checking equivalence, refer to the [Constraints Management](#) section.

ignore_io_if_fp

Controls the behavior of some rules regarding checks related to the [set_false_path](#) constraints.

If the `ignore_io_if_fp` parameter is set, only then the [set_false_path](#) constraints are considered, otherwise they are ignored.

NOTE: *If the `tc_ignore_te` parameter is set to `yes`, then the `ignore_io_if_fp` parameter is not considered.*

Example

Consider the [Inp_Trans01a](#) rule. Suppose, there is no transition specified for an input port IN that has a [set_false_path](#) constraint. Since the port IN has false path specified, there is no need to specify any transition for the port. However, the rule [Inp_Trans01a](#) will report for IN if the value of the `ignore_io_if_fp` parameter is set to `no`.

If the [set_false_path](#) constraint needs to be considered, then the value of the `ignore_io_if_fp` parameter should be set to `yes`.

Used by	Clk_Uncert03 , Inp_Del01a , Inp_Del01b , Inp_Del01c , Op_Del01a , Op_Del01b , Op_Del01c , Inp_Del02 , Op_Del02 , Inp_Del03a , Inp_Del03b , Op_Del03a , Op_Del03b , Inp_Del07 , Op_Del07 , Inp_Del07a , Op_Del07a , Inp_Trans01 , Inp_Trans01a , Clk_Gen05 , Combo_Paths01 , Combo_Paths03 , Combo_Paths06
Options	yes, no, 1, 0
Default	no
Default value in GuideWare2.0	yes
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter ignore_io_if_fp 1</code>
<i>Usage in goal/source files</i>	<code>-ignore_io_if_fp=1</code>

Assumptions/Limitations

SpyGlass processes the `set_false_path` and `set_multicycle_path` constraints under the following assumptions when the `ignore_io_if_fp` parameter is set:

1. SpyGlass processes a pair of start point and end point based on the specified constraints and NOT based on the actual topology.

For example, assume that there are two paths in the design starting from input port `in1` and ending at output port `out1` — `in1..pin1..out1` and `in1..pin2..out1`.

If you have specified the following (for only one path), all paths between input port `in1` and output port `out1` are assumed to be false paths:

```
...
set_false_path -from in1 -through pin1 -to out1
...
```

2. SpyGlass does not check for the existence of the specified paths.

For example, the following constraint has been specified for input port `in1` and output port `out1`:

```
...
set_false_path -from in1 -through pin1 -to out1
...
```

Even when the specified path does not exist in the design, all *actual* paths from input port `in1` to output port `out1` are assumed to be false paths.

3. SpyGlass does not check for completeness of these constraints with respect to the `-rise/-fall/-hold/-setup` options. If constraints with only a few options have been specified, it is assumed that constraints with the complete set of options have been specified. Therefore, the following specification is interpreted as if the specification is applicable for `-fall` and `-setup` options also:

```
set_multicycle_path -from in1 -rise -hold
```

In effect, the `-rise/-fall/-hold/-setup` options are ignored.

ignore_sdc_constraints_file

Causes the *SDC_GenerateIncr* rule to ignore all SDC Constraints specified in the seed SDC file except *set_case_analysis*, *create_clock*, and *create_generated_clock* commands.

By default, the following SDC commands, if any in the seed SDC file are read by the *SDC_GenerateIncr* rule:

<i>set_case_analysis</i>	<i>create_clock</i>
<i>create_generated_clock</i>	<i>set_input_delay</i>
<i>set_output_delay</i>	<i>set_clock_latency</i>
<i>set_clock_uncertainty</i>	<i>set_clock_transition</i>
<i>set_propagated_clock</i>	<i>set_input_transition</i>
<i>set_driving_cell</i>	<i>set_drive</i>
<i>set_load</i>	<i>set_max_delay/ set_min_delay</i>
<i>set_false_path</i>	

All other commands in the seed SDC file are ignored.

Used by	<i>SDC_GenerateIncr</i>
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter ignore_sdc_constraints_file 1</code>
<i>Usage in goal/source files</i>	<code>-ignore_sdc_constraints_file=1</code>

inp_delay_margin

Specifies the maximum delay margin between the clock period and input

delay as used by the [Inp_Del07a](#) rule.

By default, the [Inp_Del07a](#) rule flags those input ports where the input delay value is more than 80% of the corresponding clock period. When you set the `inp_delay_margin` parameter, the [Inp_Del07a](#) rule flags those input ports where the input delay is greater than the difference between the clock period and the `inp_delay_margin` parameter value.

By default, the `inp_delay_margin` parameter is set to 0.

The `inp_delay_margin` rule parameter value is assumed to be in the same unit as that as the clock period.

Refer to the [inp_percent](#) parameter because it is a related parameter.

Used by	Inp_Del07a
Options	a positive float value
Default	0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter inp_delay_margin '5'</code>
<i>Usage in goal/source files</i>	<code>-inp_delay_margin='5'</code>

inp_percent

Specifies the limit of clock period for the [Inp_Del07](#) rule.

Input delay should be less than or equal to $(\text{inp_percent}) * (\text{clock period})$. Default value for the `inp_percent` rule parameter is 80.

See the [inp_delay_margin](#) parameter also.

Used by	Inp_Del07
Options	a positive float value between 0 and 100
Default	80
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter inp_percent '75'</code>
<i>Usage in goal/source files</i>	<code>-inp_percent='75'</code>

inp_percent_max

Specifies the maximum percentage limit for the [Inp_Del07a](#) rule.

By default, the `inp_percent_max` parameter is set to 80 and the [Inp_Del07a](#) rule checks input delay relative to 80% of the clock period value as the maximum limit.

Used by	Inp_Del07a
Options	a positive float value between 0 and 100
Default	80
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter inp_percent_max '75'</code>
<i>Usage in goal/source files</i>	<code>-inp_percent_max='75'</code>

inp_percent_min

Specifies the minimum percentage limit for the [Inp_Del07a](#) rule.

By default, the `inp_percent_min` rule parameter is set to 20 and the [Inp_Del07a](#) rule checks input delay relative to 20% of the clock period value as the minimum limit.

Used by	Inp_Del07a
Options	a positive float value between 0 and 100
Default	20
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter inp_percent_min '25'</code>
<i>Usage in goal/source files</i>	<code>-inp_percent_min='25'</code>

library_clk_gen_naming

Use to switch the naming conventions for the clock names.

Default is `yes` and you can generate the clock names in accordance with other industry tools, i.e. the hierarchical name in the instance name or pin name format. Set this parameter to `no` to generate clock names as per the Liberty Reference Manual.

The following is a sample clock naming scheme in SpyGlass when the value of this parameter is `no`:

```
cell (namestring {
    generated_clock (namestring) { // This is the name of the
generated clock by default
        ...clock data...
    }
}
```

Used by	SDCPARSE
Options	yes, 1; no, 0
Default	1
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter library_clk_gen_naming 0</code>
<i>Usage in goal/source files</i>	<code>-library_clk_gen_naming=0</code>

io07_ports

Specifies the list of ports for the [Inp_Del07](#), [Inp_Del07a](#), [Op_Del07](#), and [Op_Del07a](#) rules.

By default, the value of the `io07_ports` parameter is `*`, that is, all ports are checked. You can specify the port names directly or by using wildcards.

Port names can contain wildcard (`*`) characters as in the following example:

```
set_parameter io07_ports 'din*,sel*'
```

Used by	Inp_Del07 , Inp_Del07a , Op_Del07a , Op_Del07
Options	list of string values
Default	*
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter io07_ports 'inp1,inp2,IP*'</code>
<i>Usage in goal/source files</i>	<code>-io07_ports='inp1,inp2,IP*'</code>

io07a_range

Sets the range-checking type as within or outside for the [Inp_Del07a](#) and [Op_Del07a](#) rules.

By default, the `io07a_range` parameter is set to `within` and the [Inp_Del07a](#) ([Op_Del07a](#)) rule flags input delay (or output delay) values that are not within the range `$inp_percent_min%` and `$inp_percent_max%` (`$op_percent_min%` and `$op_percent_max%`) of the clock value.

Set the `io07a_range` parameter to `outside` and the [Inp_Del07a](#) ([Op_Del07a](#)) rule flags input delay (or output delay) values that are not outside the range `$inp_percent_min%` and `$inp_percent_max%` (`$op_percent_min%` and `$op_percent_max%`) of the clock value.

Used by	Inp_Del07a , Op_Del07a
Options	<code>within</code> , <code>outside</code>
Default	<code>within</code>
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter io07a_range 'outside'</code>
<i>Usage in goal/source files</i>	<code>-io07a_range='outside'</code>

logic_level_min

Sets the minimum combinational logic level.

Used by	Block12
Options	a positive integer value
Default	0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter logic_level_min 4</code>
<i>Usage in goal/source files</i>	<code>-logic_level_min=4</code>

logic_level_max

Sets the maximum number of logic levels for the [Block05](#) and [SDC_Report03](#) rules.

By default, the value of the `logic_level_max` parameter is set to 200.

Used by	Block05
Options	a positive integer value
Default	200
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter logic_level_max '10'</code>
<i>Usage in goal/source files</i>	<code>-logic_level_max='10'</code>

multimode_sdc_ambiguous_clock_file

Clocks that have the same characteristics, applied on equivalent design objects, reaching the same set of flip-flops and/or with IO delay specified on the same set of IO ports cannot be resolved. You need to provide the information for each mode in a file containing a pair of equivalent clocks in each line separated by a space.

Used by	SDC_multimode_equiv
Options	string
Default	none
Example	

<i>Console/Tcl-based usage</i>	set_parameter multimode_sdc_ambiguous_clock_file 'prop.txt'
<i>Usage in goal/source files</i>	- multimode_sdc_ambiguous_clock_file='prop.txt'

To resolve the ambiguity between such clocks, you need to specify the ambiguous clocks in a file containing a pair of equivalent clocks and pass this file as a value in the [equiv_sdc_ambiguous_clock_file](#) parameter.

The detailed format for mapping clocks is as follows:

Mode <mode1-name>:

```
<clockname1>[:<waveform-type>] | <clockname1>:<waveform-type>[:<polarity>] <clockname2>[:<waveform-type>] | <clockname2>:<waveform-type>[:<polarity>]
```

Mode <mode2-name>:

```
<clockname1>[:<waveform-type>] | <clockname1>:<waveform-type>[:<polarity>] <clockname2>[:<waveform-type>] | <clockname2>:<waveform-type>[:<polarity>]
```

Each line should have a list of clock pairs. The description of the *waveform-type* and *polarity* fields is as follows:

- Mode state the mode name. This name is the name of the mode that needs to be compared with the merge-mode.
- *waveform-type* describes waveform of the clock. This field is optional. If not specified, 'default' waveform is assumed. The valid values are as follows:
 - a. default_waveform
 - b. rise_triggered_high_pulse
 - c. fall_triggered_high_pulse
 - d. rise_triggered_low_pulse
 - e. fall_triggered_low_pulse

The default_waveform value corresponds to the clock waveform in the SDC file. All other waveforms correspond to the waveform specified

through the pulse option of the [set_clock_sense/set_sense](#) constraint.

- *polarity* describes whether the waveform is positive or negative. This field is optional. If not specified, positive polarity is used by default. The values can either be positive or negative.

NOTE: *You need not specify all the clocks in the file. You can specify only those clocks in the file that have been declared by SpyGlass as ambiguous.*

Read the following examples for more details.

- [Example 1 - Default Clock Mapping](#)
- [Example 2 - Partial Short Form Clock Mapping](#)
- [Example 3 - Overlapping Clock Mapping](#)
- [Example 4 - Partial Overlapping Clock Mapping](#)

Example 1 - Default Clock Mapping

The following file snippet implies that default_waveform, positive polarity of Clock1 maps to default_waveform, positive polarity of Clock2.

```
--file begin--
Clock1 Clock2
--file end--
```

The default values of waveform and polarity are used for Clock1. If default values of *waveform-type* and *polarity* are used for the Clock1, this line applies to all waveform-type and polarity combinations of Clock1.

The above file snippet is equivalent to the following mapping file:

```
--file begin--
Clock1:default:positive Clock2:default:positive
Clock1:default:negative Clock2:default:negative

Clock1:rise_triggered_high_pulse:positive Clock2:rise_triggered_high_pulse:positive
Clock1:rise_triggered_high_pulse:negative Clock2:rise_triggered_high_pulse:negative

Clock1:fall_triggered_high_pulse:positive Clock2:fall_triggered_high_pulse:positive
Clock1:fall_triggered_high_pulse:negative Clock2:fall_triggered_high_pulse:negative

Clock1:rise_triggered_low_pulse:positive Clock2:rise_triggered_low_pulse:positive
```

```
Clock1:rise_triggered_low_pulse:negative Clock2:rise_triggered_low_pulse:negative

Clock1:fall_triggered_low_pulse:positive Clock2:fall_triggered_low_pulse:positive
Clock1:fall_triggered_low_pulse:negative Clock2:fall_triggered_low_pulse:negative
--file end--
```

Example 2 - Partial Short Form Clock Mapping

The following file snippet implies that `rise_triggered_high_pulse`, positive polarity of Clock1 maps to `rise_triggered_high_pulse`, positive polarity of Clock2. Since, the value of the waveform-type is non-default, this line applies to only positive polarity of `rise_triggered_high_pulse` of Clock1 and not to any other waveform or polarity combination.

```
--file begin--
Clock1:rise_triggered_high_pulse Clock2:rise_triggered_high_pulse:negative
--file end--
```

The above file snippet is equivalent to the following mapping file:

```
--file begin--
Clock1:rise_triggered_high_pulse:positive Clock2:rise_triggered_high_pulse:negative
--file end--
```

Example 3 - Overlapping Clock Mapping

The following file snippet content implies there is an existing mapping for a waveform, polarity of Clock1 and another mapping is also defined for it. The latter mapping overrides the earlier mapping. Therefore, in the following file, Clock1 maps to Clock3.

```
--file begin--
Clock1 Clock2
Clock1 Clock3
--file end--
```

The above file snippet is equivalent to the following mapping file:

```
--file begin--
Clock1:default:positive Clock3:default:positive
Clock1:default:negative Clock3:default:negative
```

Overview

```

Clock1:rise_triggered_high_pulse:positive Clock3:rise_triggered_high_pulse:positive
Clock1:rise_triggered_high_pulse:negative Clock3:rise_triggered_high_pulse:negative

Clock1:fall_triggered_high_pulse:positive Clock3:fall_triggered_high_pulse:positive
Clock1:fall_triggered_high_pulse:negative Clock3:fall_triggered_high_pulse:negative

Clock1:rise_triggered_low_pulse:positive Clock3:rise_triggered_low_pulse:positive
Clock1:rise_triggered_low_pulse:negative Clock3:rise_triggered_low_pulse:negative

Clock1:fall_triggered_low_pulse:positive Clock3:fall_triggered_low_pulse:positive
Clock1:fall_triggered_low_pulse:negative Clock3:fall_triggered_low_pulse:negative
--file end--

```

Example 4 - Partial Overlapping Clock Mapping

The following file snippet implies there exists a mapping for a waveform, polarity of Clock1 and another mapping is defined for some other waveform, polarity combination of the same clock. In this case, the latter statement overrides the earlier mapping.

```

--file begin--
Clock1 Clock2
Clock1:rise_triggered_high_pulse:positive Clock3
--file end--

```

The above file snippet is equivalent to the following mapping file:

```

--file begin--
Clock1:default:positive Clock2:default:positive
Clock1:default:negative Clock2:default:negative

Clock1:rise_triggered_high_pulse:positive Clock3:rise_triggered_high_pulse:positive
Clock1:rise_triggered_high_pulse:negative Clock2:rise_triggered_high_pulse:negative

Clock1:fall_triggered_high_pulse:positive Clock2:fall_triggered_high_pulse:positive
Clock1:fall_triggered_high_pulse:negative Clock2:fall_triggered_high_pulse:negative

Clock1:rise_triggered_low_pulse:positive Clock2:rise_triggered_low_pulse:positive
Clock1:rise_triggered_low_pulse:negative Clock2:rise_triggered_low_pulse:negative

```

```
Clock1:fall_triggered_low_pulse:positive Clock2:fall_triggered_low_pulse:positive
Clock1:fall_triggered_low_pulse:negative Clock2:fall_triggered_low_pulse:negative
--file end--
```

num_falsepath_max

Specifies the maximum number of paths that can be specified in [set_false_path](#) commands for the [False_Path04a](#) rule.

By default, the value of the `num_falsepath_max` parameter is set to 20.

Used by	False_Path04a , False_Path04a
Options	a positive integer value
Default	20
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter num_falsepath_max '50'</code>
<i>Usage in goal/source files</i>	<code>-num_falsepath_max='50'</code>

num_mcpath_max

Specifies the maximum number of paths that can be specified in [set_multicycle_path](#) commands for the [MCP04a](#) rule.

By default, the value of the `num_mcpath_max` parameter is set to 20.

Used by	MCP04a , MCP04a
Options	a positive integer value

Default	20
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter num_mcpath_max '50'</code>
<i>Usage in goal/source files</i>	<code>-num_mcpath_max='50'</code>

op_delay_margin

Specifies the maximum delay margin between the clock period and output delay as used by the [Op_Del07a](#) rule.

By default, the [Op_Del07a](#) rule flags those output ports where the output delay value is more than 20% of the corresponding clock period. When you set the `op_delay_margin` parameter, the [Op_Del07a](#) rule flags those output ports where the output delay is greater than the difference between the clock period and the `op_delay_margin` parameter value.

By default, the `op_delay_margin` parameter is set to 0. The `op_delay_margin` parameter value is assumed to be in the same unit as that as the clock period.

Related Parameter: [op_percent](#)

Used by	Op_Del07a
Options	a positive float value
Default	0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter op_delay_margin '5'</code>
<i>Usage in goal/source files</i>	<code>-op_delay_margin='5'</code>

op_percent

Specifies the limit of clock period for the *Op_Del07* rule.

Output delay should be less than or equal to $(op_percent) * (clock\ period)$. Default value for the *op_percent* parameter is 20.

Related Parameter: *op_delay_margin*

Used by	<i>Op_Del07</i>
Options	a positive float value between 0 and 100
Default	20
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter op_percent '10'</code>
<i>Usage in goal/source files</i>	<code>-op_percent='10'</code>

op_percent_max

(Optional) Specifies the maximum percentage limit for the *Op_Del07a* rule.

By default, the *op_percent_max* parameter is set to 80 and the *Op_Del07a* rule checks input delay relative to 80% of the clock period value as the maximum limit.

Used by	<i>Op_Del07a</i>
Options	a positive float value between 0 and 100
Default	80
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter op_percent_max '75'</code>
<i>Usage in goal/source files</i>	<code>-op_percent_max='75'</code>

op_percent_min

Specifies the minimum percentage limit for the [Op_Del07a](#) rule.

By default, the `op_percent_min` rule parameter is set to 20 and the [Op_Del07a](#) rule checks input delay relative to 20% of the clock period value as the minimum limit.

Used by	Op_Del07a
Options	a positive float value between 0 and 100
Default	20
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter op_percent_min '25'</code>
<i>Usage in goal/source files</i>	<code>-op_percent_min='25'</code>

report_inactive_gen_clocks

Used to remove inactive generated clock from the design. There are several sources of inactive generated clocks, such as generated clocks that have source pin, which is blocked by the [set_case_analysis](#) constraint in the fanin.

Default is `yes` and inactive generated clocks are removed from the design.

Used by	Equiv_SDC_Block , Equiv_SDC
Options	yes, 1; no, 0
Default	1
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter report_inactive_gen_clocks 0</code>
<i>Usage in goal/source files</i>	<code>-report_inactive_gen_clocks=0</code>

For more information on checking equivalence, refer to the [Constraints Management](#) section.

sdc_consist_ambiguous_clock_file

Clocks having same characteristics, applied on equivalent design objects, reaching same set of flip-flops, and/or with input/output delay specified on same set of input/output ports cannot be resolved and are considered to be ambiguous.

Used by	Equiv_SDC
Options	file name
Default	None
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter sdc_consist_ambiguous_clock_file 'prop.txt'</code>
<i>Usage in goal/source files</i>	<code>-sdc_consist_ambiguous_clock_file='prop.txt'</code>

To resolve the ambiguity between such clocks, you need to specify the ambiguous clocks in a file containing a pair of equivalent clocks and pass this file through the `sdc_consist_ambiguous_clock_file` parameter.

The detailed format for mapping clocks is as follows:

```
<clockname1>[:<waveform-type>] | <clockname1>:<waveform-  
type>[:<polarity>] <clockname2>[:<waveform-  
type>] | <clockname2>:<waveform-type>[:<polarity>]
```

Each line should have a list of clock pairs. The description of the *waveform-type* and *polarity* fields is as follows:

- *waveform-type* describes waveform of the clock. This field is optional. If not specified, 'default' waveform is assumed. The valid values are as follows:
 - a. default_waveform
 - b. rise_triggered_high_pulse
 - c. fall_triggered_high_pulse
 - d. rise_triggered_low_pulse
 - e. fall_triggered_low_pulse

The default_waveform value corresponds to the clock waveform in the SDC file. All other waveforms correspond to the waveform specified through the pulse option of the [set_clock_sense/set_sense](#) constraint.

- *polarity* describes whether the waveform is positive or negative. This field is optional. If not specified, positive polarity is used by default. The values can either be positive or negative.

NOTE: *You need not specify all the clocks in the file. You can specify only those clocks in the file that have been declared by SpyGlass as ambiguous.*

Read the following examples for more details:

- [Example 1 - Default Clock Mapping](#)
- [Example 2 - Partial Short Form Clock Mapping](#)
- [Example 3 - Overlapping Clock Mapping](#)
- [Example 4 - Partial Overlapping Clock Mapping](#)

Example 1 - Default Clock Mapping

The following file snippet implies that default_waveform, positive polarity of Clock1 maps to default_waveform, positive polarity of Clock2.

```
--file begin--
Clock1 Clock2
--file end--
```

The default values of waveform and polarity are used for Clock1. If default values of *waveform-type* and *polarity* are used for the Clock1, this line applies to all waveform-type and polarity combinations of Clock1.

The above file snippet is equivalent to the following mapping file:

```
--file begin--
Clock1:default:positive Clock2:default:positive
Clock1:default:negative Clock2:default:negative

Clock1:rise_triggered_high_pulse:positive Clock2:rise_triggered_high_pulse:positive
Clock1:rise_triggered_high_pulse:negative Clock2:rise_triggered_high_pulse:negative

Clock1:fall_triggered_high_pulse:positive Clock2:fall_triggered_high_pulse:positive
Clock1:fall_triggered_high_pulse:negative Clock2:fall_triggered_high_pulse:negative

Clock1:rise_triggered_low_pulse:positive Clock2:rise_triggered_low_pulse:positive
Clock1:rise_triggered_low_pulse:negative Clock2:rise_triggered_low_pulse:negative

Clock1:fall_triggered_low_pulse:positive Clock2:fall_triggered_low_pulse:positive
Clock1:fall_triggered_low_pulse:negative Clock2:fall_triggered_low_pulse:negative
--file end--
```

Example 2 - Partial Short Form Clock Mapping

The following file snippet implies that *rise_triggered_high_pulse*, positive polarity of Clock1 maps to *rise_triggered_high_pulse*, positive polarity of Clock2. Since, the value of the waveform-type is non-default, this line applies to only positive polarity of *rise_triggered_high_pulse* of Clock1 and not to any other waveform or polarity combination.

```
--file begin--
Clock1:rise_triggered_high_pulse Clock2:rise_triggered_high_pulse:negative
--file end--
```

The above file snippet is equivalent to the following mapping file:

```
--file begin--
Clock1:rise_triggered_high_pulse:positive Clock2:rise_triggered_high_pulse:negative
--file end--
```

Example 3 - Overlapping Clock Mapping

The following file snippet content implies there is an existing mapping for a waveform, polarity of Clock1 and another mapping is also defined for it. The latter mapping overrides the earlier mapping. Therefore, in the following file, Clock1 maps to Clock3.

```
--file begin--
Clock1 Clock2
Clock1 Clock3
--file end--
```

The above file snippet is equivalent to the following mapping file:

```
--file begin--
Clock1:default:positive Clock3:default:positive
Clock1:default:negative Clock3:default:negative

Clock1:rise_triggered_high_pulse:positive Clock3:rise_triggered_high_pulse:positive
Clock1:rise_triggered_high_pulse:negative Clock3:rise_triggered_high_pulse:negative

Clock1:fall_triggered_high_pulse:positive Clock3:fall_triggered_high_pulse:positive
Clock1:fall_triggered_high_pulse:negative Clock3:fall_triggered_high_pulse:negative

Clock1:rise_triggered_low_pulse:positive Clock3:rise_triggered_low_pulse:positive
Clock1:rise_triggered_low_pulse:negative Clock3:rise_triggered_low_pulse:negative

Clock1:fall_triggered_low_pulse:positive Clock3:fall_triggered_low_pulse:positive
Clock1:fall_triggered_low_pulse:negative Clock3:fall_triggered_low_pulse:negative
--file end--
```

Example 4 - Partial Overlapping Clock Mapping

The following file snippet implies there exists a mapping for a waveform, polarity of Clock1 and another mapping is defined for some other waveform, polarity combination of the same clock. In this case, the latter statement overrides the earlier mapping.

```
--file begin--  
Clock1 Clock2  
Clock1:rise_triggered_high_pulse:positive Clock3  
--file end--
```

The above file snippet is equivalent to the following mapping file:

```
--file begin--  
Clock1:default:positive Clock2:default:positive  
Clock1:default:negative Clock2:default:negative  
  
Clock1:rise_triggered_high_pulse:positive Clock3:rise_triggered_high_pulse:positive  
Clock1:rise_triggered_high_pulse:negative Clock2:rise_triggered_high_pulse:negative  
  
Clock1:fall_triggered_high_pulse:positive Clock2:fall_triggered_high_pulse:positive  
Clock1:fall_triggered_high_pulse:negative Clock2:fall_triggered_high_pulse:negative  
  
Clock1:rise_triggered_low_pulse:positive Clock2:rise_triggered_low_pulse:positive  
Clock1:rise_triggered_low_pulse:negative Clock2:rise_triggered_low_pulse:negative  
  
Clock1:fall_triggered_low_pulse:positive Clock2:fall_triggered_low_pulse:positive  
Clock1:fall_triggered_low_pulse:negative Clock2:fall_triggered_low_pulse:negative  
--file end--
```

sdconsis_check_fanin_fanout_for_clocks

Checks fanins of objects provided in the mapping file and whether the mapping file is incomplete.

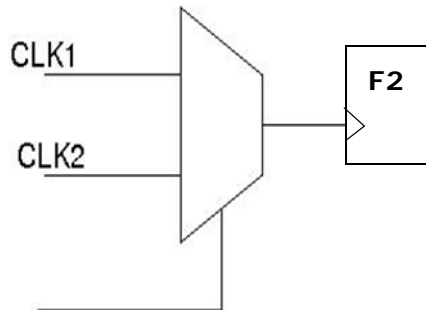
Default is no. This indicates that the fanin, fanout relationship is not

checked during clock equivalence. Set this parameter to `yes` to consider fan-in, fan-out relationships during clock equivalence. See [Example: Impact on Clock Equivalence](#).

Used by	Equiv_SDC
Options	yes, 1; no, 0
Default	no
Example	
<i>Console-based usage</i>	set_parameter sdc_consist_check_fanin_fanout_for_clocks 1
<i>Tcl-based usage</i>	set_parameter sdc_consist_check_fanin_fanout_for_clocks 1
<i>Usage in goal/ source files</i>	-sdc_consist_check_fanin_fanout_for_clocks=1

Example: Impact on Clock Equivalence

To understand the impact on clock equivalence, consider the following design.



The SDC files are as follows:

M1.sdc

```
create_clock -name clk1 -period 10 clk1
```

```
create_clock -name clk2 -period 20 clk2
```

M2.sdc

```
create_clock -name clk1 -period 20 clk1  
create_clock -name clk2 -period 10 clk2
```

After running the *Equiv_SDC* rule, the Equivalence Report is generated. The results reported for clock equivalence vary depending on the setting of this parameter.

If `sdc_consist_check_faninfor_fanout_for_clocks` is set to no, the following clock equivalence is reported in the Equivalence Report:

"Equivalent Clocks but differ in name and the object on which it is defined.",CLK1: period 20; CLK2: period 20,F2/CP,F2/CP

"Equivalent Clocks but differ in name and the object on which it is defined.",CLK2: period 10,CLK1: period 10,F2/CP,F2/CP

If `sdc_consist_check_faninfor_fanout_for_clocks` is set to yes, the following clock equivalence is reported in the Equivalence Report: :

"No equivalent clocks in implement mode.",CLK1: period 20,F2/CP,F2/CP

"No equivalent clocks in implement mode.",CLK2: period 10,F2/CP,F2/CP

"No equivalent clocks in reference mode.",CLK2: period 20,F2/CP,F2/CP

"No equivalent clocks in reference mode.",CLK1: period 10,F2/CP,F2/CP

sdc_consist_clock_precision

Specifies the clock period comparison to a specific number of decimal places in clock equivalence.

Default is 2 and the clock period comparison is till the second decimal place. You can set this parameter to any value between 1 and 4 to signify the decimal place for clock period comparison.

For example, suppose this parameter is set to 4. Clocks with period 1.2345 and 1.2346 are reported as **Inequivalent**. However, if this parameter is set to 3, these clocks are reported as **Equivalent**.

If the tolerance is set by [sdc_consis_tolerance](#) parameter, the clock characteristics comparison is based only on tolerance value. Therefore, the `sdc_consis_clock_precision` parameter does not have any impact on determining clock equivalence, if the [sdc_consis_tolerance](#) parameter is set.

Used by	Equiv_SDC
Options	1, 2, 3, 4
Default	2
Example	
<i>Console-based/Tcl-based usage</i>	<code>set_parameter sdc_consis_clock_precision 4</code>
<i>Usage in goal/source files</i>	<code>-sdc_consis_clock_precision=4</code>

sdc_consis_disambiguate_same_named_clocks

Used to automatically resolve ambiguity of clocks on the basis of name.

Typically, to resolve the ambiguity between clocks, you need to specify the ambiguous clocks in a file containing a pair of equivalent clocks and pass this file as a value in the [sdc_consis_ambiguous_clock_file](#) parameter.

When you set this parameter to `yes`, you need not specify the ambiguous clocks which have the same name, through the [sdc_consis_ambiguous_clock_file](#) parameter.

By default, this parameter is set to `no`.

Used by	Equiv_SDC
Options	no, 0; yes, 1
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter sdc_consisis_disambiguate_same_named_clock 1</code>
<i>Usage in goal/source files</i>	<code>-sdc_consisis_disambiguate_same_named_clock=1</code>

sdc_consisis_object_name_length

Controls the length of block instance names shown in the worksheet title of the generated spreadsheet.

Default is 12. This means that 12 characters are displayed in the worksheet title bar and any longer block instance name is truncated to 12 characters. Set the value to 0 to remove the length limit. Alternatively, set this parameter to a positive integer value greater than 2 to set the length.

Used by	Equiv_SDC
Options	0; positive integer value greater than 2
Default	12
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter sdc_consisis_object_name_length 9</code>
<i>Usage in goal/source files</i>	<code>-sdc_consisis_object_name_length=9</code>

sdc_consis_report_blocked_clocks

Reports equivalence/inequivalence of clocks that are blocked.

Blocked clocks are the clocks that are not reaching any clock pin of a sequential element.

Since blocked clocks do not impact clock equivalence, by default SpyGlass Constraints does not report them in the [Equivalence Report](#). However, if you want all the clocks to be reported in the Equivalence Report, set this parameter to `yes`.

The incremental flow of equivalence checking is not impacted by the inequivalence of blocked clocks. The equivalence check for further stages is performed even if such blocked clocks are reported as inequivalent.

Used by	Equiv_SDC
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter sdc_consis_report_blocked_clocks 1</code>
<i>Usage in goal/source files</i>	<code>-sdc_consis_report_blocked_clocks=1</code>

sdc_consis_report_inactive_gen_clocks

Used to remove inactive generated clock from the design. There are several sources of inactive generated clocks, such as generated clocks that have source pin, which is blocked by the [set_case_analysis](#) constraint in the fanin.

Default is `yes` and inactive generated clocks are removed from the design.

Used by	Equiv_SDC
Options	yes, 1; no, 0
Default	1
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter sdc_consist_report_inactive_gen_clocks 0</code>
<i>Usage in goal/source files</i>	<code>-sdc_consist_report_inactive_gen_clocks=0</code>

For more information on checking equivalence, refer to the [Constraints Management](#) section.

sdc_consist_report_type

Both equivalent and inequivalent SDC constraints are reported in the spreadsheets generated. Use this parameter to report only equivalent or only inequivalent constraints.

Default is `both`. This means that both equivalent and inequivalent constraints are reported. Set this parameter to `equiv_only` to report only equivalent constraints and `inequiv_only` to report inequivalent constraints only.

Used by	Equiv_SDC
Options	<code>equiv_only</code> ; <code>inequiv_only</code> ; <code>both</code>
Default	<code>both</code>
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter sdc_consist_report_type inequiv_only</code>
<i>Usage in goal/source files</i>	<code>-sdc_consist_report_type=inequiv_only</code>

sdc_consist_show_violations

Shows the violation messages generated by rules that check for equivalence. By default, the messages are suppressed and visible only by accessing the [Equivalence Report](#). Set the value to `yes` to make the messages visible in the message tree.

Used by	Equiv_SDC
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter sdc_consist_show_violations 1</code>
<i>Usage in goal/source files</i>	<code>-sdc_consist_show_violations=1</code>

sdc_consist_tolerance

Two commands are considered equivalent if their values are exactly the same. However, you may want to specify a limit (tolerance value) within which the commands are considered equivalent.

The `sdc_consist_tolerance` parameter is used to specify the tolerance value in percentage while checking for equivalency between two commands. If the difference in value between the two commands is within the tolerance value, then the rules in the SpyGlass Constraints solution do not report a violation and display the following message:

[INFO] Tolerance value of <tolerance-value> used for checking

equivalence

Used by	Equiv_SDC
Options	float 0-100
Default	0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter sdc_consist_tolerance 5</code>
<i>Usage in goal/source files</i>	<code>-sdc_consist_tolerance=5</code>

Example

Consider the following SDC commands:

```
create_clock -name CLK1 -period 10 [get_ports {clk}]
create_clock -name CLK1 -period 10.5 [get_ports {clk}]
```

The two commands are not equivalent as their clock period is not the same. This should result in a violation. However, if you set the value of the `sdc_consist_tolerance` parameter to 10, the violation is not reported.

SDC_MergeBlocks_deglitch_cell

Specifies the deglitching cell for the [SDC_MergeBlocks](#) rule.

The specified deglitching cell is considered a combinational gate while searching for clocks. The specified deglitching cell should be a library cell and must be hard-instantiated in the design. Deglitching cell will not be inferred for RTL designs.

Used by	SDC_MergeBlocks
Options	a string value
Default	unspecified

Example	
<i>Console/Tcl-based usage</i>	set_parameter SDC_MergeBlocks_deglitch_cell 'mycell'
<i>Usage in goal/source files</i>	-SDC_MergeBlocks_deglitch_cell='mycell'

SDC_MergeBlocks_deglitch_cell_clock

Specifies clock pin of the deglitching cell specified using the [SDC_MergeBlocks_deglitch_cell](#) parameter for the [SDC_MergeBlocks](#) rule.

The `SDC_MergeBlocks_deglitch_cell_clock` parameter is considered while searching for clocks. Only the fan-in of the specified clock pin is searched when the deglitching cell is reached.

Used by	SDC_MergeBlocks
Options	a string value
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	set_parameter SDC_MergeBlocks_deglitch_cell_clock 'mycell'
<i>Usage in goal/source files</i>	-SDC_MergeBlocks_deglitch_cell_clock='mycell'

SDC_DnStrm02_control_point

Specifies the `control_point` option for the `set_clock_gating_style` command in the SDC files for the [SDC_DnStrm02](#) rule.

The [SDC_DnStrm02](#) rule uses the value of the `SDC_DnStrm02_control_point` parameter to verify that the user-specified gating style is adhered to in the design.

Used by	SDC_DnStrm02
Options	before, after, none, unspecified
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter SDC_DnStrm02_control_point 'before'</code>
<i>Usage in goal/source files</i>	<code>-SDC_DnStrm02_control_point='before'</code>

SDC_DnStrm02_control_signal

Specifies the `control_signal` option for the `set_clock_gating_style` command in the SDC files for the [SDC_DnStrm02](#) rule.

The `SDC_DnStrm02` rule uses the value of the `SDC_DnStrm02_control_signal` parameter to verify that the user-specified gating style is adhered to in the design.

Used by	SDC_DnStrm02
Options	scan_enable, test_mode, unspecified
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter SDC_DnStrm02_control_signal 'scan_enable'</code>
<i>Usage in goal/source files</i>	<code>-SDC_DnStrm02_control_signal='scan_enable'</code>

SDC_DnStrm02_pedge_logic

Specifies the values of the `positive_edge_logic` option for the `set_clock_gating_style` command for the [SDC_DnStrm02](#) rule.

The [SDC_DnStrm02](#) rule uses the value of the `SDC_DnStrm02_pedge_logic` parameter to verify that the user-specified gating style is adhered to in the design.

You may specify more than one `positive_edge_logic` styles as a comma-separated lists separated by a space character as shown in the table.

Used by	SDC_DnStrm02
Options	list of string values
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter SDC_DnStrm02_pedge_logic {and,buf and,buf,inv and}</code>
<i>Usage in goal/source files</i>	<code>-SDC_DnStrm02_pedge_logic='and,buf and,buf,inv and'</code>

SDC_DnStrm02_seq_cell

Specifies the sequential cell parameter type for `set_clock_gating_style` command in SDC files for the [SDC_DnStrm02](#) rule.

Used by	SDC_DnStrm02
Options	latch, none, unspecified
Default	unspecified
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter SDC_DnStrm02_seq_cell 'latch'</code>
<i>Usage in goal/source files</i>	<code>-SDC_DnStrm02_seq_cell='latch'</code>

SDC_Methodology01_allow_block_pins

Specifies whether the [SDC_Methodology01](#) rule flags block-level logical pin names used in commands.

By default, the [SDC_Methodology01](#) rule flags block-level logical pin names used in commands.

Used by	SDC_Methodology01
Options	yes, 1; no, 0
Default	no

Example

<i>Console/Tcl-based usage</i>	<code>set_parameter SDC_Methodology01_allow_block_pins 1</code>
<i>Usage in goal/source files</i>	<code>-SDC_Methodology01_allow_block_pins=1</code>

SDC_Methodology01_commands_list

Specifies an ASCII file that contains the list of constraints (one constraint per line) to be checked by the [SDC_Methodology01](#) rule. By default, the [SDC_Methodology01](#) rule checks all supported commands.

Used by	SDC_Methodology01
Options	a string value
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter SDC_Methodology01_commands_list 'constr_list'</code>
<i>Usage in goal/source files</i>	<code>-SDC_Methodology01_commands_list='constr_list'</code>

SDC_Methodology02_forbid_constraints

Specifies an ASCII file that contains the list of forbidden constraints with its options, if any (one constraint per line) for the [SDC_Methodology02](#) rule.

For example, if you want to indicate [set_max_time_borrow](#) as a forbidden constraint, write `set_max_time_borrow` in the ASCII file.

For example, the file may contain as follows:

```
current_design
set_load
set_resistance
```

Then, the [SDC_Methodology02](#) rule reports use of the corresponding constraints in the specified SDC files.

NOTE: *You cannot specify the constraints options; only the commands/constraints can be specified.*

Used by	SDC_Methodology02
Options	a string value
Default	no
Example	

<i>Console/Tcl-based usage</i>	set_parameter SDC_Methodology02_forbid_constraints 'my_const_file'
<i>Usage in goal/source files</i>	- SDC_Methodology02_forbid_constraints='my_const_file'

SDC_DnStrm04_template

Specifies the file containing the forbidden constraint names for the [SDC_DnStrm04](#) rule.

The [SDC_DnStrm04](#) rule flags constraints not understood by tools other than Synopsys Design Compiler or PrimeTime.

The following standard tool-specific forbidden constraints list files are installed with SpyGlass:

- Astro
- Monterey

You can specify any of the standard tool-specific forbidden constraints list files as in the following example:

```
set_parameter SDC_DnStrm04_template='Monterey'
```

You can also create your own file (say myFCfile) containing list of constraints and specify it as follows:

```
set_parameter SDC_DnStrm04_template='myFcfile'
```

The template file should be an ASCII file containing the list of forbidden constraints, one constraint and its arguments, if any per line. Then, the [SDC_DnStrm04](#) rule flags use of the corresponding constraints in the specified SDC files.

NOTE: *You cannot specify the constraints options alone. You need to specify the options along with the commands/constraints. However, you can specify the commands/constraints without the options.*

The template file is searched in the following directories in the same order:

- Current working directory
- Directory specified with the environment variable `SPYGLASS_LOCAL`
- The SpyGlass Constraints solution installation directory (normally the `SPYGLASS_HOME/policies/constraints` directory)

Used by	SDC_DnStrm04
Options	Astro, Monterey or an ASCII test file
Default	Astro
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter SDC_DnStrm04_template 'Monterey'</code>
<i>Usage in goal/source files</i>	<code>-SDC_DnStrm04_template='Monterey'</code>

SDC_DnStrm04a_template

Specifies the file containing the names of the commands that are valid for the downstream tools as checked by the [SDC_DnStrm04a](#) rule.

The [SDC_DnStrm04a](#) rule reports use of those commands that are potentially problematic for downstream tools.

Used by	SDC_DnStrm04a
Options	SdcCommands or an ASCII test file.
Default	SdcCommands
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter SDC_DnStrm04a_template 'mySdcCommands'</code>
<i>Usage in goal/source files</i>	<code>-SDC_DnStrm04a_template='mySdcCommands'</code>

By default, the [SDC_DnStrm04a](#) flags use of commands other than the

commands listed in the `<your-inst-dir>/SPYGLASS_HOME/policies/constraints/SdcCommands` file.

TABLE 2 Default List of Allowed Commands (Continued)

all_outputs	create_generated_clock	echo
find	get_cells	get_lib_cell
get_lib_pin	get_net	get_pin
get_ports	if	proc
set_case_analysis	set_clock_latency	set_clock_uncertainty
set_driving_cell	set_hierarchy_separato r	set_ideal_transition
set_load	set_logic_zero	set_max_delay
set_max_transition	set_multicycle_path	set_port_fanout_number
set_resistance	set_wire_load_model	set_timing_derate
set_min_porosity	unsupress_message	all_clocks
all_registers	current_design	expr
for	get_attribute	get_clock
get_lib_pins	get_nets	get_pins
global	lappend	puts
set_clock_gating_check	set_clock_sense	set_disable_timing
set_fanout_load	set_ideal_latency	set_input_delay
set_logic_dc	set_max_area	set_max_fanout
set_min_capacitance	set_operating_conditio ns	set_units
set_wire_load_min_bloc k_size	set_wire_load_selectio n_group	set_max_dynamic_powe r
while	all_inputs	create_clock
current_instance	filter_collection	foreach
get_cell	get_clocks	get_lib_cells
get_libs	get_port	group_path

list	redirect	set
set_clock_groups	set_clock_transition	set_drive
set_false_path	set_ideal_network	set_input_transition
set_logic_one	set_max_capacitance	set_max_time_borrow
set_min_delay	set_output_delay	set_propagated_clock
set_wire_load_mode	set_data_check	set_max_leakage_power
supress_message		

You can create your own file (say myFcfile) containing list of allowed commands and specify it as follows:

```
set_parameter SDC_DnStrm04a_template 'myFcfile'
```

The template file should be an ASCII file containing the list of allowed commands, one command per line. For example, the file may contain as follows:

```
get_cells
set_case_analysis
while
```

Then, the [SDC_DnStrm04a](#) rule flags use of all other commands in the specified SDC files.

NOTE: *You cannot specify the constraints options; only the command names/constraint names can be specified.*

The template file is searched in the following directories in the same order:

- Current working directory
- Directory specified with the environment variable SPYGLASS_LOCAL
- The SpyGlass Constraints solution installation directory (normally the SPYGLASS_HOME/policies/constraints directory)

Causes SpyGlass to perform as described in the following table:

Feature	With pt rule parameter set to no	Default Mode
<ul style="list-style-type: none"> • • <code>set_signal_type</code> command • <code>@name</code> attribute of <code>filter_collection</code> command 	Supported	Invalid
<ul style="list-style-type: none"> • <code>-clock</code> and <code>-clock_fall</code> arguments of the <code>set_input_transition</code> command • <code>-class</code> argument of the <code>get_attribute</code> command • <code>direction</code> attribute of <code>filter_collection</code> command • <code>-from_edge</code>, <code>-to_edge</code>, <code>-fall_from</code>, <code>-rise_from</code>, <code>-fall_to</code>, and <code>-rise_to</code> arguments of the <code>set_clock_uncertainty</code> command 	Invalid	Supported
<code>pin_direction</code> attribute of <code>filter</code> commands	Allowed on ports	Not allowed on ports
<code>set_clock_gating_check</code> command	Not allowed on ports	Allowed on ports also
<code>-master_clock</code> and <code>-add</code> arguments of the <code>create_generated_clock</code> command	One or both can be specified	Both must be specified
<code>-source</code> argument of the <code>create_generated_clock</code> command	Source clock must be defined on the specified pin	Source clock can be in the fan-in of the specified pin
<code>object_list</code> of the <code>set_propagated_clock</code> command	Clock must be defined on each object	Clock can be defined in the fan-out of the object

Feature	With pt rule parameter set to no	Default Mode
get_* commands used without <i>pattern</i> (for example [get_cells])	Interpreted as get all (for example, same as [get_cells *])	Error condition
set_max_delay/set_min_delay commands	Leaf-level pins are not valid as timing points (error condition)	Leaf-level pins are valid as timing points (warning condition)

The behavior of the `pt` parameter with respect to the [tc_overconstrained_factor](#) parameter is as follows:

pt	Behavior	Default Mode
yes/ supermode	Normal	pt
no	Normal	dc
Used by	Clk_Gen26 , Clk_Trans13 , Clk_Uncert01 , SDCPARSE , Clk_Lat01	
Options	yes, 1; no, 0; supermode, "	
	There are certain commands that allow the names of the design objects to be changed. Supermode works in the same manner as when <code>pt</code> is set to <code>yes</code> , but permits the name changes.	
Default	yes	
Example		
<i>Console/Tcl-based usage</i>	<code>set_parameter pt no</code>	
<i>Usage in goal/source files</i>	<code>-pt=no</code>	

scan_insert

Specifies whether testability has been inserted in the design. By default, the `scan_insert` parameter is set to `no`, that is, there is no test insertion. To imply test insertion, set the value to `yes`.

Used by	Test_Rules01 , Test_Rules02 , Test_Rules03 , Test_Rules05 , Test_Rules06
Options	yes, no
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter scan_insert 'yes'</code>
<i>Usage in goal/source files</i>	<code>-scan_insert='yes'</code>

scan_shift

Specifies the mode to be used by the [Test_Rules05](#) and [Test_Rules06](#) rules.

By default, the `scan_shift` parameter is not set and the [Test_Rules05](#) and [Test_Rules06](#) rules are not run.

To run the [Test_Rules05](#) rule, set the `scan_shift` parameter to any valid mode name, as specified with the `-mode` argument of the [sdc_data](#) constraint.

To run the [Test_Rules06](#) rule, set the `scan_shift` parameter to any value other a mode name, as specified with the `-mode` argument of the [sdc_data](#) constraint.

Used by	Test_Rules05 , Test_Rules06
Options	a string value
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter scan_shift 'enable'</code>
<i>Usage in goal/source files</i>	<code>-scan_shift='enable'</code>

schematic_opt

Specifies whether the related rules generate all possible highlight information (requires larger memory and longer runtime) or generate optimized highlight information.

Currently, the `schematic_opt` parameter is applicable for the [TE_Conflict01](#) rule only.

When the `schematic_opt` parameter is set to `yes` (default), this rule generates highlight information for only few representative paths only. This process takes less memory and is faster.

However, when the `schematic_opt` parameter is set to `no`, this rule generates highlight information for all possible paths. This process may consume a large amount of memory and increase runtime depending on the design.

These rules generate messages for all possible paths in both cases.

Used by	TE_Conflict01
Options	yes, 1; no, 0
Default	yes
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter schematic_opt no</code>
<i>Usage in goal/source files</i>	<code>-schematic_opt=no</code>

strict

Specifies whether the different rules perform the checking in `strict` mode.

Used by	Const_Struct04a , Const_Struct05 , Clk_Gen02 , Clk_Lat05 , Clk_Gen23 , MCP05 , Clk_Gen33 , False_Path01 , MCP01 , TE_Consis01 , TE_Consis02 Equiv_SDC_Dual_Design
Options	yes, 1, no, 0, or a comma/space-separated list of rule names
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter strict 'yes'</code> <code>set_parameter strict 'MCP05'</code>
<i>Usage in goal/source files</i>	<code>-strict='yes'</code> <code>-strict='MCP05'</code>

tc_allow_async_pins

Specifies whether or not the tool considers asynchronous pins of sequential elements for computing sampling clocks information of input delays. The

default value of the `tc_allow_async_pins` parameter is `no`. This means that the asynchronous pins are not considered.

Used by	Inp_Del01a , Inp_Del01b , Inp_Del01c , Inp_Del03a , Inp_Del03b , and SDC_GenerateIncr rules
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_allow_async_pins yes</code>
<i>Usage in goal/source files</i>	<code>-tc_allow_async_pins=yes</code>

tc_at_speed_testing_mode

Specifies whether a design is run in the functional mode or in the at-speed test mode. The default value of the rule parameter is `no` which means that the design is run in the functional mode.

Set the value of the parameter to `yes`, to specify that the design is run in the at-speed test mode.

Used by	Inp_Del01a , Inp_Del01b , Inp_Del01c , and SDC_Methodology30 rules
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_at_speed_testing_mode yes</code>
<i>Usage in goal/source files</i>	<code>-tc_at_speed_testing_mode=yes</code>

tc_allow_design_obj

Specifies whether design objects are also allowed in the object list of the [set_clock_uncertainty/set_clock_latency](#) commands as checked by the [Clk_Uncert01](#) and [Clk_Lat01](#) rules:

If the value of the `tc_allow_design_obj` rule parameter is set to `yes`, then the [Clk_Uncert01](#) and [Clk_Lat01](#) rules can set the [set_clock_uncertainty/set_clock_latency](#) constraints on the design objects.

Used by	Clk_Uncert01 , Clk_Lat01
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_allow_design_obj yes</code>
<i>Usage in goal/source files</i>	<code>-tc_allow_design_obj=yes</code>

tc_allow_mm_or_io

Specifies the commands checked by the [Combo_Paths01](#) rule.

By default, the [Combo_Paths01](#) rule checks for both [set_input_delay/set_output_delay](#) constraints and the [set_max_delay/set_min_delay](#) constraints.

You can set the `tc_allow_mm_or_io` parameter to `io` to have the [Combo_Paths01](#) rule check for the [set_input_delay/set_output_delay](#) constraints only. Set the `tc_allow_mm_or_io` parameter to `mm` to have the [Combo_Paths01](#) rule check for the [set_max_delay/set_min_delay](#) constraints only.

Used by	Combo_Paths01
Options	both, io, mm
Default	both
Example	
<i>Console/Tcl-based usage</i>	set_parameter tc_allow_mm_or_io 'io'
<i>Usage in goal/source files</i>	-tc_allow_mm_or_io='io'

tc_bus_merge

Specifies whether or not the constraints generated by the [SDC_GenerateIncr](#) rule on bus-bits should be merged.

Used by	SDC_GenerateIncr
Options	yes, 1; no, 0
Default	yes
Example	
<i>Console/Tcl-based usage</i>	set_parameter tc_bus_merge 0
<i>Usage in goal/source files</i>	-tc_bus_merge=0

tc_check_for_multiple_clock_fanin

Specifies whether the [Clk_Gen01a](#) rule checks for multiple unconstrained clock inputs if any input is constrained.

By default, the [Clk_Gen01a](#) rule reports violation for any input that is not constrained. When you set this parameter to `no`, the [Clk_Gen01a](#) rule does not report violations for clock inputs that are not constrained, if there is one input that is constrained.

Used by	Clk_Gen01a
Options	yes, 1; no, 0
Default	yes
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_check_for_multiple_clock_fanin 0</code>
<i>Usage in goal/source files</i>	<code>-tc_check_for_multiple_clock_fanin=0</code>

tc_checkmcp_report_sorted_clock

Specifies whether or not the [mcp_domain_<integer> Report](#) generated by the [CheckMCP](#) rule displays the instance names sorted with respect to clocks.

Used by	CheckMCP
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_checkmcp_report_sorted_clock 1</code>
<i>Usage in goal/source files</i>	<code>-tc_checkmcp_report_sorted_clock=1</code>

tc_clk_compat

Specifies the behavior of the rules (mentioned in the table below) for clock domain interaction when a clock reaches the data pin of a flip-flop triggered by another clock.

By default, the `tc_clk_compat` parameter is not set and these rules do not consider it to be a clock domain interaction when a clock reaches the data pin of a flip-flop triggered by another clock.

Set the `tc_clk_compat` parameter to `yes` to have these rules consider such cases to be of clock domain interaction.

Used by	Clk_Uncert03 , False_Path07 , Clk_Gen05 , False_Path08 , DomainAnalysis , DomainInfo , DomainError , Domain_SGDC_Consis , Clk_Uncert07 , Clk_Uncert08 , and Clk_Lat03
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_clk_compat 1</code>
<i>Usage in goal/source files</i>	<code>-tc_clk_compat=1</code>

tc_ignore_clk_outputs

Used to control whether the [Clk_Gen02](#) rule should report clocks which terminate on output ports.

Default is `no` and the [Clk_Gen02](#) rule reports clocks which terminate on output ports. Set the value to `yes` to ignore clocks that terminate on output ports.

Used by	Clk_Gen02
Options	yes, 1; no, 0

Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_clk_gen02_ignore_outports 1</code>
<i>Usage in goal/source files</i>	<code>-tc_clk_gen02_ignore_outports=1</code>

For more information on the impact of the [tc_ignore_clk_outports](#) parameter, see [Example 3](#).

tc_clk_register

Specifies whether clock ports are registered.

Used by	Block11
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_clk_register 1</code>
<i>Usage in goal/source files</i>	<code>-tc_clk_register=1</code>

tc_clock_fanout_limit

Specifies the maximum fan-out count for the [High_Fan03a](#) rule.

The [High_Fan03a](#) rule flags clock nets that have a higher fan-out count than the value specified using the `tc_clock_fanout_limit` parameter. By

default, the value of the `tc_clock_fanout_limit` parameter is set to 200.

Used by	High_Fan03a
Options	a positive integer value
Default	200
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_clock_fanout_limit '10'</code>
<i>Usage in goal/source files</i>	<code>-tc_clock_fanout_limit='10'</code>

NOTE: The `tc_clock_fanout_limit` parameter only considers nets that are either clock, reset, supply, or scanenable nets.

tc_combo_check

Specifies that the [Block11](#) rule should check for existence of combinational path between block inputs and outputs.

By default, the [Block11](#) rule does not check for such paths.

Used by	Block11
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_combo_check 1</code>
<i>Usage in goal/source files</i>	<code>-tc_combo_check=1</code>

tc_combopaths03_clk_multiplier

The `tc_combopaths03_clk_multiplier` rule parameter is used by the [Combo_Paths03](#) rule.

If a [set_multicycle_path](#) constraint is set across ports, then the `Combo_Paths03` rule compares the sum of input delay and the output delay with the product of the multi-cycle path multiplier value, the clock period, and the value of the `tc_combopaths03_clk_multiplier` parameter.

Used by	Combo_Paths03
Options	a positive float value
Default	1
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_combopaths03_clk_multiplier_check 3</code>
<i>Usage in goal/source files</i>	<code>-tc_combopaths03_clk_multiplier_check=3</code>

tc_compare_cmd_file

Specifies the file that has pairs of timing exception commands, which are checked by the [TE_Conflict01](#) rule.

The timing exception commands supported are [set_false_path](#), [set_max_delay](#), [set_min_delay](#), [set_clock_groups](#), and [set_multicycle_path](#).

Each timing exception command in a pair should be separated by ":" (colon). For example, the following is a valid pair specification.

```
set_false_path:set_multicycle_path
```

If you specify an empty file, all timing exception commands are compared.

NOTE: [set_max_delay](#) and [set_min_delay](#) is not a valid pair

Used by	TE_Conflict01
Options	file
Default	<p>none</p> <ul style="list-style-type: none"> - The following overlapping timing exception commands are checked: - <code>set_clock_groups</code> with <code>set_clock_groups</code> - <code>set_false_path</code> with <code>set_false_path</code> - <code>set_multicycle_path</code> with <code>set_multicycle_path</code> - <code>set_max_delay</code> with <code>set_max_delay</code> - <code>set_min_delay</code> with <code>set_min_delay</code>
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_compare_cmd_file cmdfile</code>
<i>Usage in goal/source files</i>	<code>-tc_compare_cmd_file=cmdfile</code>

tc_comments_cmd_file

Specifies the file that has the name of constraints, which are checked by the [SDC_Methodology70](#), for the `comment` option.

The file can contain the following constraint names:

- [create_clock](#)
- [create_generated_clock](#)
- `group_path`
- [set_clock_groups](#)
- [set_false_path](#)
- [set_max_delay/set_min_delay](#)
- [set_multicycle_path](#)

All other constraints are ignored because they do not support the `comment` option.

Used by	SDC_Methodology70
Options	file
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_comments_cmd_file cmdfile</code>
<i>Usage in goal/source files</i>	<code>-tc_comments_cmd_file=cmdfile</code>

tc_ct17_clock_trans

Causes the [Clk_Trans17](#) rule process the [set_clock_transition](#) constraint in different manners.

When the `tc_ct17_clock_trans` parameter is set to ignore (default), the [Clk_Trans17](#) rule ignores [set_clock_transition](#) constraints, if any.

When the `tc_ct17_clock_trans` parameter is set to disallow, the [Clk_Trans17](#) rule flags [set_clock_transition](#) constraints, if present.

When the `tc_ct17_clock_trans` parameter is set to mandatory, the [Clk_Trans17](#) rule flags [set_clock_transition](#) constraints, if missing.

Used by	Clk_Trans17
Options	disallow, mandatory, ignore
Default	ignore
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ct17_clock_trans 'disallow'</code>
<i>Usage in goal/source files</i>	<code>-tc_ct17_clock_trans='disallow'</code>

tc_ct17_drive

Causes the *Clk_Trans17* rule process the *set_drive* constraint in different manners.

When the *tc_ct17_drive* parameter is set to *ignore* (default), the *Clk_Trans17* rule ignores *set_drive* constraints, if any.

When the *tc_ct17_drive* parameter is set to *disallow*, the *Clk_Trans17* rule flags *set_drive* constraints, if present.

When the *tc_ct17_drive* parameter is set to *mandatory*, the *Clk_Trans17* rule flags *set_drive* constraints, if missing.

Used by	<i>Clk_Trans17</i>
Options	<i>disallow</i> , <i>mandatory</i> , <i>ignore</i>
Default	<i>ignore</i>
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ct17_drive 'disallow'</code>
<i>Usage in goal/source files</i>	<code>-tc_ct17_drive='disallow'</code>

tc_ct17_driving_cell

Causes the *Clk_Trans17* rule process the *set_driving_cell* constraint in different manners.

When the *tc_ct17_driving_cell* parameter is set to *ignore* (default), the *Clk_Trans17* rule ignores *set_driving_cell* constraints, if any.

When the *tc_ct17_driving_cell* parameter is set to *disallow*, the *Clk_Trans17* rule flags *set_driving_cell* constraints, if present.

When the *tc_ct17_driving_cell* parameter is set to *mandatory*, the *Clk_Trans17* rule flags *set_driving_cell* constraints, if missing.

Used by	Clk_Trans17
Options	disallow, mandatory, ignore
Default	ignore
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ct17_driving_cell 'mandatory'</code>
<i>Usage in goal/source files</i>	<code>-tc_ct17_driving_cell='mandatory'</code>

tc_ct17_input_trans

Causes the [Clk_Trans17](#) rule process the [set_input_transition](#) constraint in different manners.

When the `tc_ct17_input_trans` parameter is set to ignore (default), the [Clk_Trans17](#) rule ignores [set_input_transition](#) constraints, if any.

When the `tc_ct17_input_trans` parameter is set to disallow, the [Clk_Trans17](#) rule reports [set_input_transition](#) constraints, if present.

When the `tc_ct17_input_trans` parameter is set to mandatory, the [Clk_Trans17](#) rule reports [set_input_transition](#) constraints, if missing.

Used by	Clk_Trans17
Options	disallow, mandatory, ignore
Default	ignore
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ct17_input_trans 'disallow'</code>
<i>Usage in goal/source files</i>	<code>-tc_ct17_input_trans='disallow'</code>

tc_c2c_max_cycles

Sets the maximum value of the number of cycles in which two clocks can be synchronized.

If two clocks have a relationship such that their clock periods can be synchronized in N cycles and their duty cycles are same, they will be assumed to be synchronous provided neither [set_false_path](#) nor [set_clock_groups](#) is specified.

The value of this parameter sets the maximum value of N.

Used by	DomainInfo , DomainError , and DomainAnalysis
Options	any positive integer
Default	50
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_c2c_max_cycles 100</code>
<i>Usage in goal/source files</i>	<code>-tc_c2c_max_cycles=100</code>

tc_disable_caching

Use this parameter to disable caching of auxiliary commands.

By default, SpyGlass executes the first command and uses it in the second command because both commands are the same. For example, in the following SDC specification, both commands are parsed, but the result of the first `get_pins` SDC command is used in the second specification, instead of searching again. This is the default behavior.

```
get_pins *
get_pins *
```

However, if this parameter is set to yes, both `get_pins` SDC commands

are parsed, but each `get_pins` SDC command is searched for again.

Used by	SDCPARSE
Options	yes, 1; no, 0; " "
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_disable_caching 1</code>
<i>Usage in goal/source files</i>	<code>-tc_disable_caching=1</code>

tc_domain_mode

Specifies the mode for inferring domains. Each mode gets values from

different constraints and follows a priority order to resolve conflicts.

Used by	DomainInfo , DomainError , and DomainAnalysis
Options	<p>STA, TC_STA, STRICT, SGDC</p> <p>STA: If this mode is selected, the tool infers the domains from the SDC constraints. The clocks must be constrained with one of the set_clock_groups, set_false_path, set_clock_uncertainty, or Master-Generated relationship. The priority, starting from highest to lowest, of the different constraints will be as follows:</p> <ul style="list-style-type: none"> * set_clock_group (ASYNC) - [highest priority]: Intergroup relation * set_false_path (ASYNC) * set_clock_uncertainty (SYNC) * Master - Generated (SYNC) * Virtual clock relation (SYNC) * set_clock_group (SYNC): Intragroup relation <p>TC_STA: If this mode is selected, the tool infers the domains from the SDC constraints. In this mode, clock crossings are honored. The clocks must be constrained with set_clock_groups, set_false_path, set_clock_uncertainty, Master-Generated, or Harmonic relationship. The priority, starting from highest to lowest, of the different constraints is as follows:</p> <ul style="list-style-type: none"> * Master - Generated (SYNC) * set_clock_group (Intergroup ASYNC) * set_false_path (ASYNC) * set_clock_uncertainty (SYNC) * Harmonic relation (SYNC) * No-Constraint (having Clk Crossing - Sync) * Virtual clock relation (SYNC) <p>STRICT: Specify the set_clock_groups constraint between the clocks which are ASYNC. Clocks specified in the same group will be assumed as SYNC. Parent-generated clocks will be assumed to be in same domain unless over-ridden by the user using SCG command.</p> <p>SGDC: In this mode, if clock domains are defined in SGDC, the domains would be inferred from SGDC. The domain of generated clocks is inferred based on the domain of the parent clock, if not defined explicitly in the SGDC. Child generated clocks inherit the domain of the parent clock. However, a parent clock does not get the domain of a child clock. Domain defined in the clock_group SGDC command gets precedence over the domain of the parent clock.</p>
Default	STA
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_domain_mode TC_STA</code>
<i>Usage in goal/source files</i>	<code>-tc_domain_mode=TC_STA</code>

TABLE 3 Report generated as per the `tc_domain_mode` parameter option.

Generate Report	Set <code>tc_domain_mode</code> Value			
	TC_STA	STA	STRICT	SGDC
<i>Domain Matrix Report</i>	Yes	Yes	Yes	Yes
<i>Domain Cyclic Errors Report</i>	Yes	Yes	No	No
<i>Missing Domains Report</i>	Yes	Yes	Yes	Yes

tc_drc_spec_file

Specifies a template file containing limiting values for DRC constraints, such as [set_load](#), [set_input_transition](#), [set_max_capacitance](#), [set_max_transition](#), [set_clock_transition](#), [set_clock_uncertainty](#), [set_clock_latency](#), and [set_driving_cell](#).

The template file contents should follow the Tcl syntax. The filename is determined by the [SDC_Methodology72](#) rule by using the values of this parameter and the [tc_tech](#) parameters, as shown below.

```
<tc_drc_spec_file><tc_tech>
```

Used by	SDC_Methodology72
Options	file path
Default	none
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_drc_spec_file /home/user/spec_files/myspec</code>
<i>Usage in goal/source files</i>	<code>-tc_drc_spec_file=/home/user/spec_files/myspec</code>

tc_enable_preset_clear_arcs

Enables preset and clear arcs of sequential cells.

By default, timing exceptions are not propagated through clear and preset arcs of sequential cells. However, if you set the [tc_enable_preset_clear_arcs](#) parameter to *yes*, clear and preset pins are not valid end points and timing exceptions are propagated through clear and preset arcs of sequential cells. This support is added only for net list designs with library cells that have "preset-to-q" and "clear-to-q" arcs. It will not work in RTL designs.

Used by	False_Path01
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_enable_preset_clear_arcs 1</code>
<i>Usage in goal/source files</i>	<code>-tc_enable_preset_clear_arcs=1</code>

tc_high_fan_nets_file

Specifies the file that contains a list of all RTL net names for checking by the [High_Fan16](#) rule.

Used by	High_Fan16
Options	file
Default	none
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter tc_high_fan_nets_file cmdfile</code>
<i>Usage in goal/source files</i>	<code>-tc_high_fan_nets_file=cmdfile</code>

tc_honor_virtual_clk

Default is no. Set this parameter to yes to honor the `set_input_delay/set_output_delay` defined with virtual clocks that do not have any matching real clock in the SDC file.

Do not use the `tc_honor_virtual_clk` parameter with the [chip](#) parameter.

Used by	Inp_Del03a , Inp_Del03b , Op_Del03a , Op_Del03b
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_honor_virtual_clk 1</code>
<i>Usage in goal/source files</i>	<code>-tc_honor_virtual_clk=1</code>

tc_express_check

Enables certain options to improve run time. By setting this parameter to yes, the following options are enabled:

- `no_sgdc_check`

- **nosch**: Use this command to suppress generation of schematic data by rules so that no schematic highlighting is available in SpyGlass GUI.
- **noispy**: This option is the same as the **nosch** command. With the **noispy** command, the module schematic is not available in the SpyGlass GUI.
- **noreport**: Use this command to suppress report generation.
- **disallow_view_delete**: Use this command to disable large design processing mode.

Used by	All rules in the SpyGlass Constraints solution
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_express_check 1</code>
<i>Usage in goal/source files</i>	<code>-tc_express_check=1</code>

tc_enable_param_sdc_flow

Generates parameters for the timing values in the SDC template based on the format specified by the [gen_sdc_param_file](#) parameter.

Used by	SDC_GenerateIncr
Options	yes, 1; no, 0
Default	no
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter tc_enable_param_sdc_flow 1</code>
<i>Usage in goal/source files</i>	<code>-tc_enable_param_sdc_flow=1</code>

tc_ignore_async_ports

Specifies the asynchronous ports that should be ignored.

By default, the asynchronous ports are not ignored. To ignore specific asynchronous ports, specify them in a file and set this parameter to that file.

You can specify regular expressions and vector points, such as `in[0:9]`, in the file. In addition, you can include comments by inserting `#` at the start of the line containing the comment. The SpyGlass Constraints solution ignores all lines starting with `#`.

Used by	Inp_Del01 , Inp_Del03 , Op_Del01 , Op_Del03
Options	file name
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ignore_async_ports 'asyncPorts.txt'</code>
<i>Usage in goal/source files</i>	<code>-tc_ignore_async_ports='asyncPorts.txt'</code>

tc_fanout_limit

Specifies the maximum fan-out count for the [High_Fan01](#), [High_Fan01a](#), [High_Fan03b](#), [High_Fan11](#), and [SDC_GenerateIncr](#) rules.

The *High_Fan01*, *High_Fan01a*, and *High_Fan03b* and *SDC_GenerateIncr* rules flag nets that have a higher fan-out count than the value specified by the `tc_fanout_limit` parameter. The *High_Fan11* rule flags *set_max_fanout* constraints that set a value more than the value specified by the `tc_fanout_limit` parameter. By default, the value of the `tc_fanout_limit` parameter is set to 200.

Used by	<i>High_Fan01</i> , <i>High_Fan01a</i> , <i>High_Fan03b</i> , <i>High_Fan11</i> , <i>SDC_GenerateIncr</i>
Options	a positive integer value
Default	200
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_fanout_limit '10'</code>
<i>Usage in goal/source files</i>	<code>-tc_fanout_limit='10'</code>

NOTE: The `tc_fanout_limit` parameter considers all nets.

tc_ignore_block_path

Specifies whether or not the *CheckMCP* rule has considered the *set_case_analysis*/*set_input_delay*/*set_output_delay* constraints while identifying the clock relationship.

Used by	<i>CheckMCP</i>
Options	yes, 1; no, 0
Default	yes
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ignore_block_path 0</code>
<i>Usage in goal/source files</i>	<code>-tc_ignore_block_path=0</code>

tc_ignore_if_clk_op_port

Controls whether an unconstrained output port, which has a clock propagating to it, is reported

Default is `no` and unconstrained output ports, which have a clock propagating to them, are reported. Set this parameter to `yes` to not report such output ports.

Used by	Op_Del01a , Op_Del01b , Op_Del01c
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ignore_if_clk_op_port 1</code>
<i>Usage in goal/source files</i>	<code>-tc_ignore_if_clk_op_port=1</code>

tc_ignore_io_if_minmax_delay

Controls whether the [Inp_Del01b](#) and [Op_Del01b](#) rules report missing IO delay, when [set_max_delay](#) or [set_min_delay](#) is specified, even if the port is connected to a register.

Default is `no` and the rules report missing IO delays. Set this parameter to

yes to ignore missing IO delays, when [set_max_delay](#) or [set_min_delay](#) is specified, even if the port is connected to a register.

Used by	Inp_Del01b , Op_Del01b
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ignore_io_if_minmax_delay 1</code>
<i>Usage in goal/source files</i>	<code>-tc_ignore_io_if_minmax_delay=1</code>

tc_ignore_iodelay_if_maxdelay

Overrides the [set_max_delay](#) constraint over IO delays.

Default is no and the [Combo_Paths03](#) rule reports a violation when the sum of input and output delays is greater than the clock period where the value of the [set_max_delay](#) constraint is greater than the clock period. Set this parameter to yes to not report such violations.

Used by	Combo_Paths03
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ignore_iodelay_if_maxdelay 1</code>
<i>Usage in goal/source files</i>	<code>-tc_ignore_iodelay_if_maxdelay=1</code>

tc_ignore_latch_enable

Specifies whether the [Clk_Gen01a](#), [Clk_Gen01b](#), and [Clk_Gen33](#) rules ignore the control/enable pins of latches present in the design.

Used by	Clk_Gen01a , Clk_Gen01b , Clk_Gen33
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	set_parameter tc_ignore_latch_enable 1
<i>Usage in goal/source files</i>	-tc_ignore_latch_enable=1

tc_ignore_libcells

Default is no and the [SDC_Methodology01](#) rule reports constraints specified on hierarchical pins of library cells. Set the value to yes to not report constraints specified on hierarchical pins of library cells.

For more information, refer to [Example 2: Disabling Reporting of Constraints Defined on Hierarchical Pins of Library Cells](#).

Used by	SDC_Methodology01
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	set_parameter tc_ignore_libcells 1
<i>Usage in goal/source files</i>	-tc_ignore_libcells=1

tc_ignore_nets

Specifies the comma-separated name list of nets to be ignored by the [High_Fan01a](#) rule.

You can use wildcard characters while specifying net names. By default, the [High_Fan01a](#) rule checks all nets in the design.

Used by	High_Fan01a
Options	A String Value
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ignore_nets "gen_clk"</code>
<i>Usage in goal/source files</i>	<code>-tc_ignore_nets="gen_clk"</code>

tc_ignore_min

Causes the related rules to ignore missing `-min` option in SDC command (in worst-case SDC file) whose completeness is being checked by the respective rule.

The worst-case SDC file is specified using the `-corner worst` in the [sdc_data](#) constraint.

Used by	Clk_Lat06 , Clk_Trans09 , Clk_Trans15 , High_Fan04 , High_Fan05 , Inp_Del04 , Inp_Trans03 , Inp_Trans04 , Op_Del07a
Options	yes, 1; no, 0
Default	no
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ignore_min 1</code>
<i>Usage in goal/source files</i>	<code>-tc_ignore_min=1</code>

tc_ignore_missing_minmax_delay

Controls the behavior of the [Combo_Paths02](#) rule for checks related to [set_min_delay](#) and [set_max_delay](#).

Default is no and the [Combo_Paths02](#) rule reports a violation when a [set_max_delay](#) is specified without a corresponding [set_min_delay](#), and vice-versa. Set this parameter to yes to not perform this check.

Used by	Combo_Paths02
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ignore_missing_minmax_delay 1</code>
<i>Usage in goal/source files</i>	<code>-tc_ignore_missing_minmax_delay=1</code>

tc_ignored_commands

Specifies the file containing the list of SDC commands (one command per line) to be ignored by the [SDCPARSE](#) rule.

For example, valid commands can be specified like [create_clock](#) which will effectively ignore [create_clock](#), i.e. remove all the clocks from the design.

SDC non-supported commands, such as `read_verilog`, can be specified so that there are no invalid command errors while parsing this command in

the SDC file.

Used by	SDCPARSE
Options	fileName
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ignored_commands 'fileName'</code>
<i>Usage in goal/source files</i>	<code>-tc_ignored_commands=' fileName'</code>

Related parameter: [tc_stop_parsing_ignored_commands](#)

tc_ignore_modes

Specifies mode names that are ignored when merging modes.

Used by	SDC_ModeMerge
Options	Name of modes
Default	none
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ignore_modes "mode1 mode2 mode3"</code>
<i>Usage in goal/source files</i>	<code>-tc_ignore_modes="mode1 mode2 mode3"</code>

tc_ignore_scg

Controls whether the [set_clock_groups](#) are ignored by the [Clk_Gen06](#) rule.

Default is no. Set this parameter to yes to ignore [set_clock_groups](#) and to report the corresponding violation message.

Used by	Clk_Gen06
Options	yes, 1; no, 0
Default	0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_ignore_scg 1</code>
<i>Usage in goal/source files</i>	<code>-tc_ignore_scg=1</code>

tc_ignore_te

Controls the behavior of some rule regarding checks related to [set_false_path](#) and [set_multicycle_path](#) settings.

If the `tc_ignore_te` parameter is set to `yes`, both [set_false_path](#) and [set_multicycle_path](#) constraints are ignored. However, if the value of the `tc_ignore_te` parameter is set to `no`, the [set_multicycle_path](#) constraints are considered but the [set_false_path](#) constraints are considered only if the value of the [ignore_io_if_fp](#) parameter is set to `yes`.

Used by	Clk_Uncert03 , Inp_Del01a , Inp_Del01b , Inp_Del01c , Op_Del01a , Op_Del01b , Op_Del01c , Inp_Del02 , Op_Del02 , Inp_Del03a , Inp_Del03b , Op_Del03a , Op_Del03b , Inp_Del07 , Inp_Del07a , Op_Del07 , Op_Del07a , Inp_Trans01 , Inp_Trans01a , Clk_Gen05 , Combo_Paths01 , Combo_Paths03 , Combo_Paths06
Options	yes, no, 1, 0
Default	yes
Default value in GuideWare2.0	no
Example	

<i>Console/Tcl-based usage</i>	set_parameter tc_ignore_te 1
<i>Usage in goal/source files</i>	-tc_ignore_te=1

tc_inter_block_delay

Specifies the interconnect delay as a factor of the clock period.

Used by	IO_Consis04
Options	float less than 1
Default	0
Example	
<i>Console/Tcl-based usage</i>	set_parameter tc_inter_block_delay 0.5
<i>Usage in goal/source files</i>	-tc_inter_block_delay=0.5

tc_io_delay_bus_margin

Specifies the maximum percentage by which an input/output delay of a bit of a bus can vary relative to the mean bus delay.

Used by	Inp_Del10 , Op_Del09
Options	positive float value
Default	0
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter tc_io_delay_bus_margin 10</code>
<i>Usage in goal/source files</i>	<code>-tc_io_delay_bus_margin=10</code>

tc_io_reg

Specifies the type of block ports to be checked by the [Block11](#) rule.

You can set the `tc_io_reg` parameter to the following values:

Value	For checking...
input	Only block inputs
output (default)	Only block outputs
both	Both block inputs and block outputs

Used by	Block11
Options	input, output, both
Default	output
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_io_reg "both"</code>
<i>Usage in goal/source files</i>	<code>-tc_io_reg="both"</code>

tc_lvpc

Specifies the limit for reporting number of violations per command.

A single timing exception command can constrain many paths in a design. Therefore, a large number of violations, which have a similar reason, for that command could be reported.

To report a limited number of violations, set the `tc_lvpc` parameter to the number of violations you want reported.

Used by	Equiv_SDC , Equiv_SDC_Block , Equiv_SDC_Top , Equiv_SDC_Dual_Design , SDC_multimode_equiv
Options	integer value: Report the number of violations you want to view -1: Report all the violations
Default	20
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_lvpc 50</code>
<i>Usage in goal/source files</i>	<code>-tc_lvpc=50</code>

tc_modemerge_backref_info

Controls the debugging information provided in the output merge SDC file generated by the [SDC_ModeMerge](#) rule.

By default, this parameter is set to 1 and back reference information is provided as comments for each of the merged SDC constraints being generated. Set this parameter to 0 to not provide the debugging information.

Used by	SDC_ModeMerge
Options	0, no; 1, yes
Default	yes
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter tc_modemerge_backref_info 0</code>
<i>Usage in goal/source files</i>	<code>-tc_modemerge_backref_info=0</code>

tc_num_load_max

Specifies the maximum number of [set_load](#) constraints that can be provided for design objects.

Used by	Load04
Options	any positive integer
Default	0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_num_load_max 5</code>
<i>Usage in goal/source files</i>	<code>-tc_num_load_max=5</code>

tc_num_fp_max

Specifies the maximum number of [set_false_path](#) constraints that can be provided in the SDC file.

Used by	False_Path04b
Options	any positive integer
Default	0
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter tc_num_fp_max 5</code>
<i>Usage in goal/source files</i>	<code>-tc_num_fp_max=5</code>

tc_num_mcp_max

Specifies the maximum number of [set_multicycle_path](#) constraints that can be provided in the SDC file.

Used by	MCP04b
Options	any positive integer
Default	0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_num_mcp_max 5</code>
<i>Usage in goal/source files</i>	<code>-tc_num_mcp_max=5</code>

tc_opt01_port_des

Specifies the scope of the [High_Fan12](#), [SDC_Methodology62](#), and [SDC_Methodology63](#) rules. These rules flag ports or design depending upon the value of the `tc_opt01_port_des` parameter.

Used by	High_Fan12 , SDC_Methodology62 , SDC_Methodology63 , SDC_GenerateIncr
Options	port, design, both, design_or_port
Default	both

Example

<i>Console/Tcl-based usage</i>	<code>set_parameter tc_opt01_port_des port</code>
<i>Usage in goal/source files</i>	<code>-tc_opt01_port_des=port</code>

tc_overconstrained_factor

Specifies the over-constrained limit as a factor of the clock period. By default, the parameter is set to 0.8. If the timing-path starts/ends at the top-level port, the delay specified is also considered in delay computation.

Used by	IO_Consis04
---------	-----------------------------

Options	float less than 1
---------	-------------------

Default	0.8
---------	-----

Example

<i>Console/Tcl-based usage</i>	<code>set_parameter tc_overconstrained_factor 0.5</code>
<i>Usage in goal/source files</i>	<code>-tc_overconstrained_factor=0.5</code>

tc_regression_mode

Enables regression mode of operation for faster execution of certain rules.

Regression mode is useful in running regressions on large design databases because it reduces runtime and requires less memory. This is achieved by not creating SDC and schematic back references in the timing exceptions spreadsheet.

Default is 0 and the regression mode is disabled. This indicates that the timing exceptions spreadsheet contains all SDC and schematic back references. When you enable regression mode by setting this parameter to

1, SDC and schematic back references are not created in the timing exceptions spreadsheet.

The `tc_regression_mode` parameter value takes precedence over the `tc_report_verbose` and `tc_report_backref_all` parameters. Therefore, if the `tc_regression_mode` parameter is set to 1, both `tc_report_verbose` and `tc_report_backref_all` parameters are ignored.

Used by	Equiv_SDC , Equiv_SDC_Block , Equiv_SDC_Dual_Design , SDC_ModeMerge , SDC_Coverage , SDC_multimode_equiv
Options	0, no; 1, yes
Default	0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_regression_mode 1</code>
<i>Usage in goal/source files</i>	<code>-tc_regression_mode=1</code>

See also [tc_report_verbose](#) and [tc_report_backref_all](#).

tc_report_backref_all

Enables the timing exceptions spreadsheet to report all SDC and schematic back references.

Default is 0 and only the first 10,000 rows have SDC and schematic back references. This saves runtime. Set this parameter to 1 to report all SDC and schematic back references in the timing exceptions spreadsheet.

The `tc_regression_mode` parameter value takes precedence over the `tc_report_backref_all` parameter. Therefore, if the `tc_regression_mode` parameter is set to 1, the `tc_report_backref_all` parameter is ignored.

Used by	Equiv_SDC , Equiv_SDC_Block , Equiv_SDC_Dual_Design , SDC_ModeMerge , SDC_Coverage , SDC_multimode_equiv
Options	0, no; 1, yes
Default	0
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_report_backref_all 1</code>
<i>Usage in goal/source files</i>	<code>-tc_report_backref_all=1</code>

See also [tc_report_verbose](#) and [tc_regression_mode](#).

tc_report_clks_start_end_points

Controls whether the start and/or end points corresponding to the clocks will be reported. This parameter is useful when clocks are specified in the from and/or points of the constraint.

If this parameter is set to `no`, the start and/or end points corresponding to the clocks are not reported.

Used by	CheckMCP
Options	yes, 1; no, 0
Default	yes
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_report_clks_start_end_points no</code>
<i>Usage in goal/source files</i>	<code>-tc_report_clks_start_end_points=no</code>

tc_report_csv_messages

Controls whether the messages reported by specific rules in the SpyGlass Constraints solution be presented in CSV format and therefore can be read through Spreadsheet Viewer.

Currently, this parameter supports only the [SDC_Methodology01](#) rule. To understand the impact of setting this parameter, refer to [Example 2: Disabling Reporting of Constraints Defined on Hierarchical Pins of Library Cells](#).

Used by	SDC_Methodology01
Options	SDC_Methodology01
Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	set_parameter tc_report_CSV_messages SDC_Methodology01
<i>Usage in goal/source files</i>	-tc_report_CSV_messages=SDC_Methodology01

tc_report_unconnected_points

Controls the number of unconnected points that are reported for exceptions specified in the SDC file.

Default is 5. This indicates that a maximum of 5 unconnected points are reported for exceptions specified in the SDC file. You can set this parameter to an integer between 0 and 1000 to increase or decrease this limit. For example, if you set this parameter to 20, up to 20 unconnected points are reported in the generated CSV file.

Used by	False_Path01 , MCP01 , TE_Consis01 , TE_Consis02
Options	0 to 1000
Default	5

Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_report_unconnected_points 20</code>
<i>Usage in goal/source files</i>	<code>-tc_report_unconnected_points=20</code>

tc_report_verbose

Enables verbose reporting in the timing exceptions spreadsheet.

When verbose reporting is enabled, both equivalent and inequivalent timing exceptions are reported in the timing exceptions spreadsheet. Similarly, the spreadsheet generated by the [SDC_ModeMerge](#) rule shows ignored timing exceptions. Verbose reporting is enabled by default.

Default is 1 and verbose reporting is enabled. Set this parameter to 0 to disable verbose reporting. When verbose reporting is disabled, equivalent timing exceptions are not reported. Similarly, the spreadsheet generated by the [SDC_ModeMerge](#) rule does not show ignored timing exceptions.

The [tc_regression_mode](#) parameter value takes precedence over the `tc_report_verbose` parameter. Therefore, if the `tc_regression_mode` parameter is set to 1, the `tc_report_verbose` parameter value is ignored.

Used by	Equiv_SDC , Equiv_SDC_Block , Equiv_SDC_Dual_Design , SDC_ModeMerge , SDC_multimode_equiv
Options	0, no; 1, yes
Default	1
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_report_verbose 0</code>
<i>Usage in goal/source files</i>	<code>-tc_report_verbose=0</code>

See also [tc_regression_mode](#) and [tc_report_backref_all](#).

tc_setup_hold

Specifies the behavior of related rules regarding checks for the `-setup` and `-hold` options.

You can set the value of the `tc_setup_hold` parameter to `setup` (to check for the `-setup` option only), `hold` (to check for the `-hold` option only), or `both` (to check for both `-setup` and `-hold` options).

Used by	Clk_Uncert02b , Clk_Uncert02c , Clk_Uncert05 , False_Path07 , MCP05 , SDC_Methodology06 , SDC_Methodology16
Options	setup, hold, both
Default	both
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_setup_hold {setup}</code>
<i>Usage in goal/source files</i>	<code>-tc_setup_hold="setup"</code>

tc_show_all_unconstrained_flops

Specifies whether all unconstrained clock pins of a sequential cell be dumped in the CSV report.

Used by	Clk_Gen33
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_show_all_unconstrained_flops 1</code>
<i>Usage in goal/source files</i>	<code>-tc_show_all_unconstrained_flops=1</code>

tc_show_sca_on_terminals

Displays the [set_case_analysis](#) attributes on nets and terminals.

Default is no and the schematic displays the `set_case_analysis` attributes only on nets. Set this parameter to yes to display the `set_case_analysis` attributes on nets and terminals.

Used by	Show_Case_Analysis
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_show_sca_on_terminals yes</code>
<i>Usage in goal/source files</i>	<code>-tc_show_sca_on_terminals=yes</code>

tc_similar_clocks_tolerance_levels

Sets the tolerance percentage that is used to compare clock waveforms

when identifying similar clocks when merging modes.

By default, the value of this parameter is set to 10. This indicates that clock waveforms that are within a 10% range are merged while retaining the fastest clock.

Used by	SDC_ModeMerge
Options	0.0 to 100
Default	10
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_similar_clock_tolerance_levels 20</code>
<i>Usage in goal/source files</i>	<code>-tc_similar_clock_tolerance_levels=20</code>

tc_solver_run_time

With the `tc_solver_run_time` parameter, you can set the runtime limit of the formal engine for functional analysis of generated clocks.

The default runtime is set to 2 hours per [sdc_data](#).

The time does not include activities, such as the time taken for analysis, elaboration, synthesis, flattening, and path generation.

Used by	Clk_Gen23a
Options	a positive integer with s for second, h for hours, and m for minutes
Default	2h
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter txv_solver_run_time 10h</code>
<i>Usage in goal/source files</i>	<code>-txv_solver_run_time=10h</code>

NOTE: *You cannot specify a combination of hours, minutes, and seconds while specifying the time.*

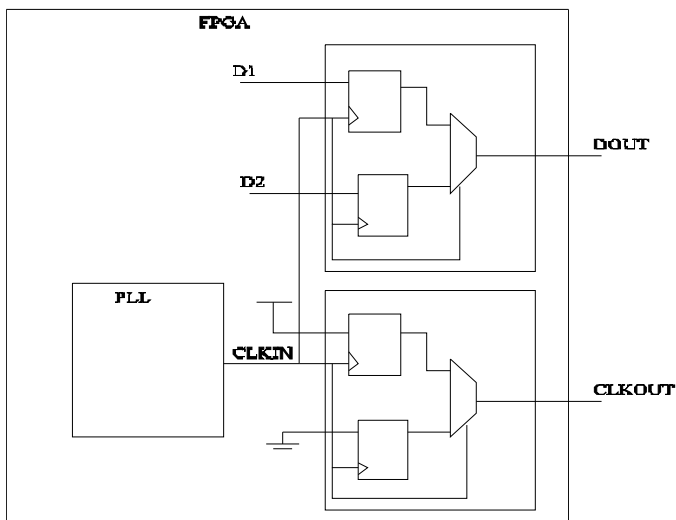
tc_source_syn_clks

If this parameter is set to yes, dependent rules will support the [Source Synchronous Interfaces](#) feature. This impacts generated clocks (divide_by 1 only) applied at inout/output ports.

Used by	<i>Clk_Gen02, Clk_Gen08, Op_Del09</i>
Options	yes, no, 1, 0
Default	yes
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_source_syn_clks 0</code>
<i>Usage in goal/source files</i>	<code>-tc_source_syn_clks=0</code>

Source Synchronous Interfaces

The source synchronous clocks are used to source and sink data for the same clock. The source synchronous structure is as follows.



This setup is used at the source side of data and a similar setup is at sink side of data. The CLKIN clock is generated at the out of the PLL and the CLKOUT clock is identical to the CLKIN clock. The motive of CLKOUT is to capture the DOUT data for CLKOUT, i.e. CLKIN.

As PLL generates clocks of various phases, but the generated clock should be identical as CLKIN. The constraints are applied on ports as follows:

```
create_clock -name CLK -period 20 c
create_generated_clock -name CLKIN -source c -divide_by 2 f1/Q
create_generated_clock -name CLKOUT -source f1/Q -divide_by 1
CLKOUT
set_output_delay 2.0 -clock CLKOUT DOUT
CLKIN and CLKOUT can be generated through the create\_generated\_clock.
CLKOUT is generated for source clock CLKIN with either divide_by=1 or
multiply_by=1 or combinational option. For example:
create_generated_clock -name CLKOUT -source CLKIN -divide_by
1 CLKOUT
create_generated_clock -name CLKOUT -source CLKIN -
multiply_by 1 CLKOUT
```



```
create_generated_clock -name CLKOUT -source CLKIN -
combinational CLKOUT
```

This means the path from CLKIN to CLKOUT must be combinational only. In the illustration, the path, CLKIN-MUX_SEL-CLKOUT, is combinational. Path from CLKIN to CLKOUT should satisfy the following conditions

1. Path polarity must be same as the polarity of the path from CLKIN to f2/CP.
2. Path must not have any sequential elements.
3. Path can have any number of combinational elements.

tc_start_end_max_count

Specifies the maximum limit for the [High_Fan16](#) rule.

By default, the `tc_start_end_max_count` parameter is set to 100 and the [High_Fan16](#) rule reports nets specified using the `tc_high_fan_nets_file` parameter or nets whose immediate fan-out count is greater than the limit specified using the `tc_fanout_limit` parameter when the total number of start-points (end-points) in the fan-in (fan-out) of the net is more than 100.

You can set the `tc_start_end_max_count` parameter to any positive integer number.

Used by	High_Fan16
Options	positive integer
Default	100
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_start_end_max_count 50</code>
<i>Usage in goal/source files</i>	<code>-tc_start_end_max_count=50</code>

tc_start_end_point_details

Specifies the level of details in the CSV report generated by the [High_Fan16](#) rule.

The report has the following fields:

- Name of the high fan-out net
- Actual count of start-points (or end-points)
- Name list of start-points (or end-points) only when `tc_start_end_point_details` parameter is set to `yes`

Used by	High_Fan16
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_start_end_point_detail 1</code>
<i>Usage in goal/source files</i>	<code>-tc_start_end_point_detail=1</code>

tc_stop_parsing_ignored_commands

Use this parameter to not parse any nested SDC commands, which are specified inside SDC commands passed through the [tc_ignored_commands](#) parameter.

By using the [tc_ignored_commands](#) parameter, you can specify SDC commands that should not be parsed. However, all nested SDC commands are parsed. For example, suppose you have specified `set_false_path` as an SDC command that should not be parsed and you have the following SDC specification. In this specification, the `get_ports` and `get_pins` commands are parsed.

```
set_false_path -from [get_ports dont] -to [get_pins no/
doesnotexist/D]
```

If you set the `tc_stop_parsing_ignored_commands` parameter to `yes`, SpyGlass stops parsing such nested SDC commands.

Used by	SDCPARSE
Options	Yes, 1; No, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_stop_parsing_ignored_commands 1</code>
<i>Usage in goal/source files</i>	<code>-tc_stop_parsing_ignored_commands=1</code>

Related parameter: [tc_ignored_commands](#)

tc_te_conflict_fast_run

Use this parameter to not generate a schematic after running the [TE_Conflict01](#) rule. By not generating the schematic, the rule runtime is faster.

Default is `no` and the [TE_Conflict01](#) rule generates a schematic. Set this parameter to `yes` to not generate a schematic. This results in a faster rule runtime.

If the [schematic_opt](#) parameter is set to `no`, the `tc_te_conflict_fast_run` parameter does not have any effect.

Used by	TE_Conflict01
Options	yes, 1; no, 0
Default	no
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter tc_te_conflict_fast_run 1</code>
<i>Usage in goal/source files</i>	<code>-tc_te_conflict_fast_run=1</code>

tc_tech

Specifies a template file containing limiting values for DRC constraints, such as [set_load](#), [set_input_transition](#), [set_max_capacitance](#), [set_max_transition](#), [set_clock_transition](#), [set_clock_uncertainty](#), [set_clock_latency](#), and [set_driving_cell](#).

The template file contents should follow the Tcl syntax. The filename is determined by the [SDC_Methodology72](#) rule by using the values of this parameter and the tc_tech parameters, as shown below.

```
<tc_drc_spec_file><tc_tech>
```

Used by	SDC_Methodology72
Options	string value
Default	none
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_tech 40</code>
<i>Usage in goal/source files</i>	<code>-tc_tech=40</code>

tc_violate_ports

Used by [Clk_Gen29](#) to violate for converging paths that are reaching to an output port.

Default is no. Set this parameter to yes to report converging paths that

are reaching an output port.

Used by	Clk_Gen29
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_violate_ports 1</code>
<i>Usage in goal/source files</i>	<code>-tc_violate_ports=1</code>

tc_virtual_clock_prefixes

Specifies the clock name prefix for virtual clock names for the domain flow. With this parameter, you can specify a list of prefixes as part of a clock name in the SDC file. By default, no prefix is specified for virtual clocks.

Example

Suppose, there is a real clock C1 in the design and a virtual clock v_C1. The constraints specification is as follows:

```
create_clock -name C1 -period 10 clk1
create_clock -name v_C1 -period 14.1
```

If the `tc_virtual_clock_prefixes` parameter is set to "v_", the virtual clock v_C1 is assumed to be related to C1.

If no conflict arises on assuming synchronous relation between C1 and v_C1, the clocks are assigned the same domain and are assumed synchronous to each other.

Used by	DomainAnalysis , DomainError , DomainInfo
Options	list of string values

Default	unspecified
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_virtual_clock_prefixes 'v1,v2,v3'</code>
<i>Usage in goal/source files</i>	<code>-tc_virtual_clock_prefixes='v1,v2,v3'</code>

tc_virtual_clock_suffixes

Specifies the clock name suffix for virtual clock names for the domain flow. With this parameter, you can specify a list of suffixes as part of a clock name in the SDC file. By default, no suffix is specified for virtual clocks.

Example

Suppose, there is a real clock C1 in the design and a virtual clock C1_v. The constraints specification is as follows:

```
create_clock -name C1 -period 10 clk1
create_clock -name C1_v -period 14.1
```

If the `tc_virtual_clock_suffixes` parameter is set to `"_v"`, the virtual clock C1_v is assumed to be related to C1.

If no conflict arises on assuming synchronous relation between C1 and C1_v, the clocks are assigned the same domain and are assumed synchronous to each other.

Used by	DomainAnalysis , DomainError , DomainInfo
Options	list of string values
Default	unspecified
Example	

<i>Console/Tcl-based usage</i>	<code>set_parameter tc_virtual_clock_suffices 'v1,v2,v3'</code>
<i>Usage in goal/source files</i>	<code>-tc_virtual_clock_suffices='v1,v2,v3'</code>

tc_waive

Causes the *Clk_Uncert02c* rule to ignore *set_clock_uncertainty* constraints with the `-hold` option specified with zero value.

By default, the `tc_waive` rule parameter is set to `none` and the *Clk_Uncert02c* rule flags *set_clock_uncertainty* constraints with the `-hold` option specified with zero value when the *tc_setup_hold* is set to `hold` or `both`.

Used by	<i>Clk_Uncert02c</i>
Options	<code>zero_hold</code> , <code>none</code>
Default	<code>none</code>
Example	
<i>Console/Tcl-based usage</i>	<code>set_parameter tc_waive 'zero_hold'</code>
<i>Usage in goal/source files</i>	<code>-tc_waive='zero_hold'</code>

tc_waive_const_struct05

This parameter is used by *Const_Struct05* to waive off violations for certain pairs of conflicting constraints. Possible pairs are *set_input_transition* &

[set_clock_transition](#), [set_input_transition](#) & [set_driving_cell](#), and [create_clock](#) and [create_generated_clock](#).

Used by	Const_Struct05
Options	none, list of pairs (set_input_transition : set_clock_transition , set_driving_cell : set_input_transition , create_clock : create_generated_clock) separated by comma in any order
Default	none
Example	
<i>Console/Tcl-based usage</i>	<pre>set_parameter tc_waive_const_struct05 'set_input_transition: set_clock_transition, set_driving_cell: set_input_transition, create_clock: create_generated_clock'</pre>
<i>Usage in goal/source files</i>	<pre>-tc_waive_const_struct05='set_input_transition: set_clock_transition, set_driving_cell: set_input_transition, create_clock: create_generated_clock''</pre>

tc_xbuf_ignore_clks

Specifies whether the XBuf01 rule ignores the [set_multicycle_path](#) commands that have clock(s) specified in the `-from` or `-to` list. By default, all such commands are ignored.

Used by	False Path Rules
Options	yes, 1; no, 0
Default	yes
Example	

<i>Console/Tcl-based usage</i>	set_parameter tc_xbuf_ignore_clks 0
<i>Usage in goal/source files</i>	-tc_xbuf_ignore_clks=0

tcdecompile

Specifies to decompile the constraints files' information.

When the `tcdecompile` parameter is specified, the [SDCPARSE](#) rule writes its constraints information (as read/understood by SpyGlass) into a file named `TcdecompiledInfo` that shows the expanded commands as well as the file name and line number of the original SDC constraints file.

An excerpt of the `TcdecompiledInfo` file is as follows:

```
#ideal.sdc@@28@@
sg_set_ideal_network port_pin_list {A1/in1}
#ideal.sdc@@29@@
sg_set_ideal_network port_pin_list {A1/rtlc_I2/Z A1/rtlc_I2/
in1 A1/rtlc_I2/in2}
```

The `tcdecompile` parameter is a debug parameter and not required to be specified in a normal run. It could be useful only when you want to know whether SpyGlass was able to parse the constraints passed to it.

NOTE: *The expanded command might refer to internally (internal to SpyGlass) generated names also.*

Used by	SDCPARSE
Options	yes, 1; no, 0
Default	no
Example	

<i>Console/Tcl-based usage</i>	set_parameter tcdecompile 1
<i>Usage in goal/source files</i>	-tcdecompile=1

verbose

Reduces the number of violations count for the [I/O Rules](#) rule. If the `verbose` parameter is set to `no`, the [I/O Rules](#) rule reports a single violation from each clock source to all clock pins present in its fan-out. If the `verbose` parameter is set to `yes`, then the [I/O Rules](#) rule reports a violation for each violating path from clock source to clock pin.

In addition, the [Block11](#) rule reports the logic levels for an unregistered port.

Used by	I/O Rules , Block11
Options	yes, 1; no, 0
Default	no
Example	
<i>Console/Tcl-based usage</i>	set_parameter verbose 1
<i>Usage in goal/source files</i>	-verbose=1

Other Rule Parameters

There are other rule parameters related to delay estimator (the [Block05](#) rule). Please refer the respective rule description for details.

SpyGlass Constraints Reports

The SpyGlass Constraints solution generates rules and product-specific reports that can be opened from the **Reports** menu in the Console GUI.

The SpyGlass Constraints solution reports object hierarchical names using forward slash (/) as hierarchical separator instead of dot (.). Therefore, the hierarchical names are reported in a format consistent with SDC files.

The following table lists the reports that can be generated.

TABLE 4 Type of reports generated by the SpyGlass Constraints solution

Report Category	Reports Used For
<i>Timing Constraints Information Reports</i>	View basic information on <i>set_disable_timing</i> , unparsed constraints, and ports that are registered in a design.
<i>Timing Constraints Domain Reports</i>	View domain information and cyclic dependencies.
<i>Timing Analysis Consolidated Report</i>	Intuitively debug problems related to constraints related to case analysis, clock, I/O, and timing exceptions. Since this report provides a wealth of information, the issues are surfaced gradually to enable systematic debugging. The issues and constraints are directly linked back to the SDC file and, where applicable, a schematic view is generated.
<i>Timing Constraints Coverage Report</i>	Discover design objects that are not covered or are partially covered. The issues and constraints are directly linked back to the SDC file and, where applicable, a schematic view is generated.
<i>Timing Exception Reports</i>	Identify timing exception issues.

TABLE 4 Type of reports generated by the SpyGlass Constraints solution

Report Category	Reports Used For
<i>Constraints Equivalence Reports</i>	Verify that the SDC files for a design are functionally equivalent. In addition, use these reports to ensure that the block-level constraints are correct in the context of the chip (block-to-top analysis).
<i>Miscellaneous Reports</i>	Debug issues related to various constraints related to latency, clock paths, dont use, and clock uncertainty.

Timing Constraints Information Reports

The following reports provide data on the basic timing constraints of a design.

TABLE 5 Reports that show basic timing constraints information

Purpose	Report	Generated by	Spreadsheet Viewer Support
Lists disabled timing points set by set_disable_timing	tc_report_disable_timing Report Report Name tc_report_disable_timing	Disable_Timing 01	No
Lists unparsed and unpopulated commands.	tc_unparsed_commands Report Report Name tc_unparsed_commands	SDCPARSE	No
View list of registered input/output/inout ports	tc_block11_info Report Report Name tc_block11_info	Block11	No

To view other reports that are generated by the SpyGlass Constraints solution, refer to [SpyGlass Constraints Reports](#).

tc_report_disable_timing Report

The `tc_report_disable_timing` report displays the list of timing points that have been disabled by setting the [set_disable_timing](#) constraint.

For more information, review the following sections:

- [Generate the tc_report_disable_timing Report](#)
- [View the tc_report_disable_timing Report](#)
- [Type of Information in the tc_report_disable_timing Report](#)
- [Interpret the tc_report_disable_timing Report](#)
- [Related Reports](#)

Generate the tc_report_disable_timing Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** Not applicable
- **Run Rule:**
 - [Disable_Timing01](#) — [Reports objects specified in set_disable_timing](#)
 - Parameters: None
- **Generated by Default:** No, update project file with:

```
set_option report tc_report_disable_timing
```

View the tc_report_disable_timing Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the tc_report_disable_timing Report

The `tc_report_disable_timing` report lists disable timing points set by [set_disable_timing](#). The `tc_report_disable_timing` report contains the following

information:

- **Port/Cell:** The port/cell where disable timing is set
- **From:** The start point of the disable timing path.
- **To:** The end points of the disable timing path.
- **SDC File:** The SDC file name.
- **Line Number:** The line number of the *set_disable_timing* command in the SDC file.

Interpret the tc_report_disable_timing Report

A sample report is a follows.

```
#####
# Total Number of Generated Messages : 2
# Number of Waived Messages : 0
# Number of Reported Messages : 2
#
# Number of Overlimit Messages : 0
#####
#####
Port/Cell      From      To      SDC File      Line Number
-----
```

Related Reports

[Timing Constraints Information Reports](#)

Timing Constraints Information Reports

tc_unparsed_commands Report

The `tc_unparsed_commands` report displays the list of unparsed and unpopulated commands.

For more information, review the following sections:

- [Generate the `tc_unparsed_commands` Report](#)
- [View the `tc_unparsed_commands` Report](#)
- [Type of Information in the `tc_unparsed_commands` Report](#)
- [Interpret the `tc_unparsed_commands` Report](#)
- [Related Reports](#)

Generate the `tc_unparsed_commands` Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** Not applicable
- **Run Rule:**
 - SDCPARSE* — *Reads the SDC and SGDC files for running rules in the SpyGlass Constraints solution*
 - Parameters (Optional):
 - tc_ignored_commands: Default is unspecified. Set the value to the file name that contains the list of SDC commands, which the SDCPARSE rule should ignore while parsing the SDC files.*

View the `tc_unparsed_commands` Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the `tc_unparsed_commands` Report

The `tc_unparsed_commands` report contains the following information:

- Invalid SDC commands
- Unpopulated commands that are syntactically correct
- Unparsed commands

Interpret the `tc_unparsed_commands` Report

A sample report is as follows.

```
#####  
Number of commands syntactically parsed but not used by  
SpyGlass Constraints = 0  
Number of commands ignored by SpyGlass Constraints = 0  
Number of commands not understood by SpyGlass Constraints = 0  
Number of invalid commands in SpyGlass Constraints = 0  
Number of commands populated = 1  
#####
```

Related Reports

[Timing Constraints Information Reports](#)

tc_block11_info Report

The tc_block11_info report displays the list of registered input, output, or inout ports in the design.

For more information, review the following sections:

- [Generate the tc_block11_info Report](#)
- [View the tc_block11_info Report](#)
- [Type of Information in the tc_block11_info Report](#)
- [Interpret the tc_block11_info Report](#)
- [Related Reports](#)

Generate the tc_block11_info Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** Not applicable
- **Run Rule:**
 - [Block11](#) — *Reports block outputs, inouts, or inputs that are not registered*
 - Parameters (Optional):
 - tc_combo_check: Default is no. Set the value to yes to report cases when a combinational path exists between a port and any other port.*

View the tc_block11_info Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from spyglass_constraints/reports.

Type of Information in the tc_block11_info Report

Since the [Block11](#) rule reports unregistered ports, the tc_block11_info report lists ports that are not reported by the [Block11](#) rule.

Timing Constraints Information Reports

A port is assumed to be a registered port when:

- A sequential cell is found when traversing from the port
- The port is a clock port or is in the transitive fan-out of a clock
- The port is blocked (due to case analysis settings or connected to ground)
- The port is a hanging port
- The inout port is used as input only

The `tc_block11_info` report contains information about port names, port type, and the reason for which the ports are not reported.

Interpret the `tc_block11_info` Report

A sample report is shown as follows. Also, refer to the [Example Code and/or Schematic](#) section of the [Block11](#) rule to understand when this rule reports violation messages and how false path is supported.

```
#####
#      Total Number of Generated Messages :          11
#      Number of Waived Messages          :           0
#      Number of Reported Messages       :          11
#
#      Number of Overlimit Messages      :           0
#####
##      PortName      PortType      Reason
#####
REPORT FOR DESIGN :: add3bits.str AND SCHEMA LINE NO :: 6
*****
PortName      PortType      Reason
+++++
REPORT FOR DESIGN :: mult1.beh AND SCHEMA LINE NO :: 10
*****
```

PortName	PortType	Reason
+++++		
REPORT FOR DESIGN :: mult.str AND SCHEMA LINE NO :: 2		

PortName	PortType	Reason
+++++		

Related Reports

[*Timing Constraints Information Reports*](#)

Timing Constraints Domain Reports

The following table lists the reports that can help you locate timing constraints domain issues in a design.

TABLE 6 Reports for identifying domain relationships between clocks

Purpose	Report	Generated by	Spreadsheet Viewer Support
Determine the synchronous/asynchronous relationship with other clocks	Domain Matrix Report	DomainError	Yes
Determine clocks that have a cyclic dependency	Domain Cyclic Errors Report	DomainError	Yes
Determine clocks for which no domain could be inferred	Missing Domains Report	DomainError	Yes

Apart from these reports, the SpyGlass TXV solution provides intensive coverage of reports useful for resolving issues related to domain and timing exception verification. Refer to the *SpyGlass TXV Rules Reference Guide*.

To view other reports that are generated by the SpyGlass Constraints solution, refer to [SpyGlass Constraints Reports](#).

Domain Matrix Report

The Domain Matrix report displays domains assigned to clocks based on the SDC constraints specified. In addition, this report provides the synchronous/asynchronous relationship with other clocks. By interpreting the Domain Matrix report, you can verify the results of other violation messages, which use the domain assignment shown in this report.

For more information, review the following sections:

- [Generate the Domain Matrix Report](#)
- [View the Domain Matrix Report](#)
- [Type of Information Reported in the Domain Matrix Report](#)
- [Interpret the Domain Matrix Report](#)
- [Related Reports](#)

Generate the Domain Matrix Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** `sdc_audit`
- **Run Rule:**
 - [DomainError](#) — *Reports clock domain errors extracted from SDC commands*
 - Parameters (optional)
 - tc_domain_mode: Default is STA. Set the value to specify the mode for inferring domains. Other possible values are: TC_STA, STRICT, and SGDC.*

View the Domain Matrix Report

After generation, the following message appears:

[INFO] Domain Matrix for design "<design-name>", based on the individual constraints defined in the SDC as inferred in the TC_STA mode, <file-name>, <line no.>

You can view the report by one of the following methods:

- **Spreadsheet Viewer (Recommended):** In the Console GUI Message Tree, double-click the message.

Timing Constraints Domain Reports

- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information Reported in the Domain Matrix Report

The Domain Matrix Report contains information about the clock-to-clock relationship (synchronous/asynchronous) inferred based on the constraints defined in the SDC file.

The report contains the following information:

- **Clock Name:** Represents the name of the clock that is being used as a baseline.
- **Domain:** Represents the domain name.
- **Filename Line:** Displays the SDC file and line number that is used for inference.

The Domain Matrix Report is shown as follows:

	A	B	C	D	E	F
	Clock Name	Domain	Filename:Line	C1	C2	C3
1	C1	d1	top_sdc_1	NA	S(GEN,HMR)	S(GEN,HMR)
2	C2	d1	top_sdc_2	S(GEN)	NA	S(SCG,HMR)
3	C3	d1	top_sdc_3	S(GEN)	S(SCG)	NA

Messages: Displayed: 3 Total: 3

This report contains mnemonics. To interpret the mnemonics, refer to [Mnemonics in the Domains Spreadsheets](#).

Interpret the Domain Matrix Report

Refer to the following examples:

- [Example 3 - Domain Matrix](#)
- [Example 4 - Domain Matrix and Missing Domains](#)

Related Reports

[*Timing Constraints Domain Reports*](#)

Domain Cyclic Errors Report

The Domain Cyclic Dependency Errors report displays information about the clocks, which are in a cyclic dependency, while inferring the domains. Use this report for a holistic view of clocks with the constraints specified in the SDC file and the clock relationship, synchronous or asynchronous, assumed while inferring domains.

For more information, review the following sections:

- [Generate the Domain Cyclic Errors Report](#)
- [View the Domain Cyclic Errors Report](#)
- [Type of Information in the Domain Cyclic Errors Report](#)
- [Related Reports](#)

Generate the Domain Cyclic Errors Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** `sdc_audit`
 - **Run Rule:**
 - Rule Name: [DomainError](#)
 - Rule Title: [Reports clock domain errors extracted from SDC commands](#)
 - Parameters (Mandatory)
 - [tc_domain_mode](#): Default is `STA`. Set the value to specify the mode for inferring domains. Other possible values are: `TC_STA`, `STRICT`, and `SGDC`.
- This report is not generated when the value of the `tc_domain_mode` parameter is set to `STRICT` or `SGDC`.

View the Domain Cyclic Errors Report

After generation, the following message appears:

[ERROR] For design "<design-name>", <no. of dependencies> cyclic dependency exists between the clocks, same domain is assumed. Report is generated, <file-name>, <line no.>

You can view the report by one of the following methods:

- **Spreadsheet Viewer (Recommended):** In the Console GUI Message Tree, double-click the message.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

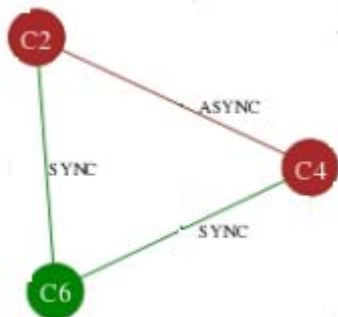
Type of Information in the Domain Cyclic Errors Report

This section describes the information in the Domain Cyclic Errors report. This report contains information about the clocks which resulted in conflicts while inferring the domains. There are two types of conflicts: Cyclic Error or Dual Relation Conflict.

Cyclic Error

Cyclic Errors are applicable only to STA and TC_STA modes.

In the following figure, the clock relationship is a cyclic error because C2 and C4 have a direct asynchronous relation and both are synchronous to C6:



Dual Relation Conflict

Dual Relation Conflict is applicable only to the STA mode.

A Dual Relation Conflict occurs when there is a `set_false_path` defined from one clock to another, while there is a `set_clock_uncertainty` defined in the reverse direction, as shown by the following SDC constraints specification:

```
set_false_path -from C1 -to C2
```

```
set_clock_uncertainty 1.0 -from C2 -to C1
```

Domain Cyclic Errors Spreadsheet Report

The report contains the following information:

- **Conflicting Clock Pair:** Shows clocks defined as asynchronous directly but also have a synchronous relation (direct/indirect). Click the cells in this column to cross-probe the conflicting clock pair in the SDC file.
- **Related Clocks:** Shows all clocks related to the conflicting pair of clocks by synchronous relation.

After you click a cell in this column, the **FSM Viewer** button appears in the toolbar. In addition, the related clocks are highlighted in the SDC file.

Click the **FSM Viewer** button to view the graphical relationship of the related clocks. Each node on the graph depicts a clock. For quick identification, conflicting and related clocks are shown in different colors. In addition, the asynchronous and synchronous relationship is shown in different colors. The graphical representation appears only for Cyclic Errors.

- **Type of Conflict:** Shows the type of conflict. For Dual Relation Conflict, click the hyperlink to cross-probe the conflicting constraints in the SDC file.

	A	B	C
	Conflicting Clock Pair	Related Clocks	Type
1	C5 C3	C1 C10 C11 C12 C13 C14 C15 C2 C3 C4 C5 C6 C7 C8 C9	Dual relation Conflict
2	C1 C13	C1 C10 C11 C12 C13 C14 C15 C2 C3 C4 C5 C6 C7 C8 C9	Cyclic error
3	C2 C4	C1 C10 C11 C12 C13 C14 C15 C2 C3 C4 C5 C6 C7 C8 C9	Cyclic error

The screenshot shows the 'Fsm Viewer' window with a graph of clock domains. Nodes are labeled C1 through C15. Nodes C1, C2, C3, C4, C13, and C14 are colored red, indicating they are conflicting clocks. All other nodes (C10, C11, C12, C15, C5, C6, C7, C8, C9) are colored green, indicating they are related clocks. Edges between nodes are labeled either 'SYNC' (green lines) or 'ASYNC' (red lines). A legend on the right side of the window defines the colors and line styles: a red circle for 'conflicting clocks', a green circle for 'related clocks', a red line for 'ASYNC', and a green line for 'SYNC'. The log at the bottom of the window shows 'Node C12'.

Related Reports

[Timing Constraints Domain Reports](#)

Missing Domains Report

The Missing Domains report displays information about the clocks for which no domain could be inferred. The clocks listed in this report may impact the violation for other domain dependent checks.

For more information, review the following sections:

- [Generate the Missing Domains Report](#)
- [View the Missing Domains Report](#)
- [Type of Information in the Missing Domains Report](#)
- [Interpret the Missing Domains Report](#)
- [Related Reports](#)

Generate the Missing Domains Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** `sdc_audit`
- **Run Rule:**
 - DomainError* — *Reports clock domain errors extracted from SDC commands*
 - Parameters (Mandatory)
 - tc_domain_mode: Default is STA. Set the value to specify the mode for inferring domains. Other possible values are: TC_STA, STRICT, and SGDC.*

View the Missing Domains Report

After generation, the following message appears:

[ERROR] Missing domain for <no. of clocks> clock(s), <file name>, <line no.>

You can view the report by one of the following methods:

- **Spreadsheet Viewer (Recommended):** In the Console GUI Message Tree, double-click the message.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the Missing Domains Report

The Missing Domains report displays clocks for which no domain could be inferred. The report contains the following information:

- **List of clocks with no domains:** Displays the clocks for which a domain could not be inferred or SGDC clocks, which could not be mapped to a corresponding SDC clock.
- **Filename Line:** Displays the SDC file name and the line number of the [create_clock](#) or [create_generated_clock](#) constraint that are used to define the clock.

The Missing Domain report is shown as follows.

	A	B	C
	List of clocks with no domains	Filename:Line	Reason for NoDomain
1	C4	../top.sdc:4	No Constraint specified
2	C5	../top.sdc:5	No Constraint specified
3	C6	../top.sdc:6	No Constraint specified

In the SGDC mode, the clock constraints, which do not have a corresponding SDC clock, are reported explicitly as an SGDC clock. For example, if C7 is an SGDC clock that has no corresponding SDC clock, it would be reported in the **List of clocks with no domains** column as follows:

C7 (SGDC)

Interpret the Missing Domains Report

Refer to the following example:

- [Example 4 - Domain Matrix and Missing Domains](#)

Related Reports

[Timing Constraints Domain Reports](#)

Timing Analysis Consolidated Report

The Timing Analysis Consolidated report helps you identify issues related to the following: `set_case_analysis` (case analysis), Clocks, I/O Delays, Timing Exceptions.

For more information, review the following sections:

- [Generate the Timing Analysis Consolidated Report](#)
- [View the Timing Analysis Consolidated Report](#)
- [Type of Information in the Timing Analysis Consolidated Report](#)
- [Interpret the Timing Analysis Consolidated Report](#)
- [Related Reports](#)

Generate the Timing Analysis Consolidated Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** Not applicable
- **Run Rule:**
 - [Cons_SDC_Report](#) — *Generates clock domain information from SDC commands*
 - Parameters: None

View the Timing Analysis Consolidated Report

After generation, the following message appears:

[INFO] Consolidated report generated for block <block-name>, schema line num : <line-num>

You can view the report by one of the following methods:

- **Spreadsheet Viewer (Recommended):** In the Console GUI Message Tree, double-click the message.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the Timing Analysis Consolidated Report

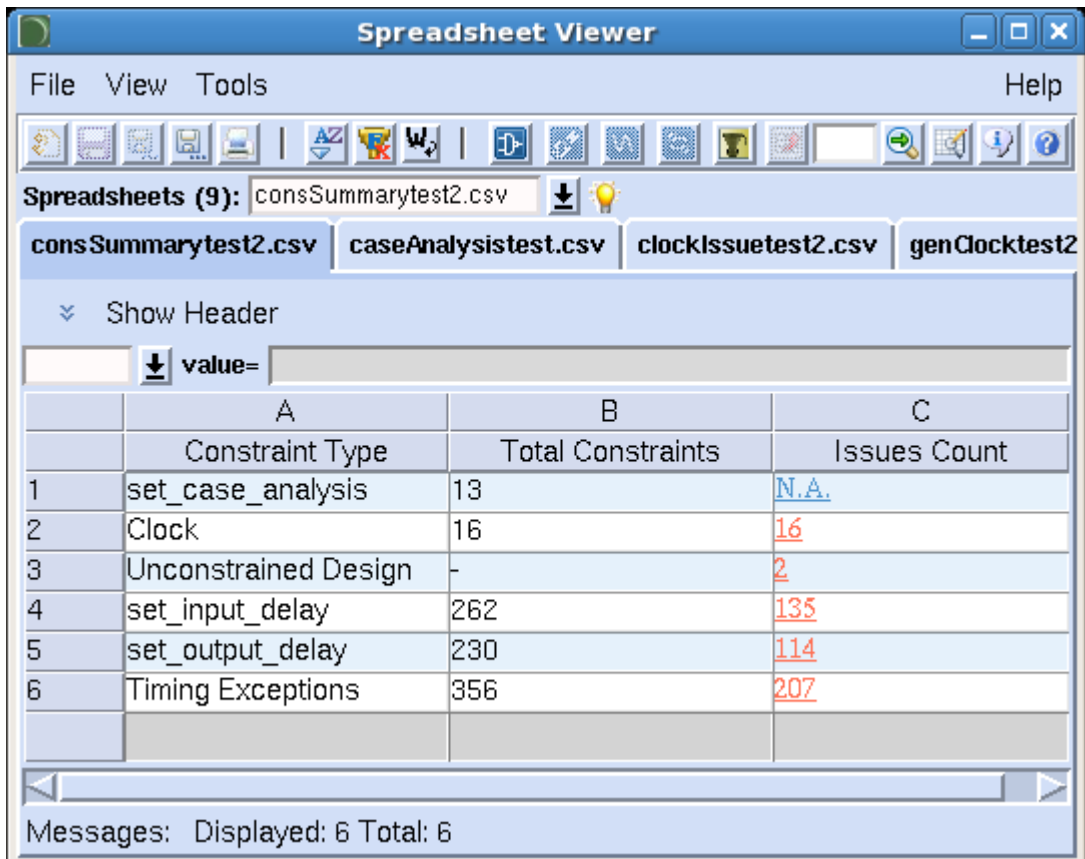
The following worksheets are part of the Timing Analysis Consolidated report:

- **consSummary<block-name><sdcounter>.csv**: Provides the summary and the index page for the report
- **caseAnalysis<block-name>.csv**: Contains the case analysis values for the block <block-name> across all SDC modes
- **clockIssue<block-name><sdcounter>.csv**: Contains data related to primary clocks that have issues
- **genClock<block-name><sdcounter>.csv**: Contains data related to generated clocks that have issues
- **clock<block-name><sdcounter>.csv**: Contains data generated for clocks
- **UnConsClocksPin<block-name><sdcounter>.csv**: Contains list of unconstrained pins and the probable clock sources
- **inputDelay<block-name><sdcounter>.csv**: Contains data generated for input delays.
- **outputDelay<block-name><sdcounter>.csv**: Contains data generated for output delays
- **timingExcp<block-name><sdcounter>.csv**: Contains data generated for timing exceptions

Interpret the Timing Analysis Consolidated Report

The Timing Analysis Consolidated report enables you to efficiently and effectively debug issues related to constraints specification in a design. You can begin to debug, by viewing the **Summary** tab (consSummary<block-name><sdcounter>.csv).

A sample report is shown as follows:



The screenshot shows the 'Spreadsheet Viewer' application window. The title bar reads 'Spreadsheet Viewer'. The menu bar includes 'File', 'View', 'Tools', and 'Help'. Below the menu bar is a toolbar with various icons. The main area shows a list of spreadsheets: 'consSummarytest2.csv', 'caseAnalysistest.csv', 'clockIssuetest2.csv', and 'genClocktest2'. The 'consSummarytest2.csv' tab is active. A 'Show Header' checkbox is checked. Below the spreadsheet, there is a search bar with a dropdown arrow and the text 'value='.

	A	B	C
	Constraint Type	Total Constraints	Issues Count
1	set_case_analysis	13	N.A.
2	Clock	16	16
3	Unconstrained Design	-	2
4	set_input_delay	262	135
5	set_output_delay	230	114
6	Timing Exceptions	356	207

Messages: Displayed: 6 Total: 6

In the screenshot above, the **Summary** tab displays the constraint type, total number of instances of each constraint type, and the number of issues for each constraint type. When a constraint has an issue, the count of issues appears in red font under the **Issues Count** heading. The count of issues are linked to the more information that can help you debug and resolve the issue.

For example, for **262 set_input_delay** constraints, **135** issues have been reported. To get details on these issues, click **135**. Spreadsheet Viewer displays the tab that displays the input delay worksheet (inputDelay<block-name><sdccounter>.csv), as shown. All issues are in red font and are back referenced to enable you to resolve the issue.

	A	B	C	D	E	F	G	H	I	J
	Design Object	Object Type	Delay Value	Associated Clocks	Overwritten	Unequal Delay	Missing Clocks	Incorrect Clocks	Delay Options	Issues
1	in1[4]	port	87(clk5)	clk5	Yes	0:7	clk7	clk5	delay > clock period	5
2	in4[1]	port	4(clk2)	clk2	Yes	0:3	clk3 clk7	clk2	-	4
3	in3[4]	port	87(clk5)	clk5	Yes	0:7	clk3	clk5	delay > clock period	5
4	in0[1]	port	9(clk5)	9(clk clk5 clk7 clk0 clk3	No	0:7	clk2	clk0 clk3 clk4 clk5	delay > clock period	4
5	in1[3]	port	9(clk5)	9(clk clk5 clk7 clk0 clk1	No				d	4
6	in3[3]	port	9(clk5)	9(clk clk5 clk7 clk0 clk1	No				d	4
7	in3[7]	port	9(clk5)	9(clk clk5 clk7 clk0 clk3	No	0:7	clk2	clk0 clk3 clk4 clk5	delay > clock period	4
8	in4[2]	port	8(clk2)	clk2	Yes	0:3	clk1 clk3 clk7	-	delay > clock period	4
9	in0[0]	port	9(clk5)	9(clk clk5 clk7 clk0 clk3	No	0:7	clk1 clk2	clk0 clk4 clk5 clk6	delay > clock period	4
10	in0[4]	port	87(clk5)	clk5	Yes	0:7	clk1 clk2	clk5	delay > clock period	5
11	in1[6]	port	9(clk5)	9(clk clk5 clk7 clk0 clk3	No	0:7	clk1 clk2	clk0 clk4 clk5 clk6	delay > clock period	4
12	in2[5]	port	9(clk5)	9(clk clk5 clk7 clk0 clk3	No	0:7	clk1 clk2	clk0 clk4 clk5 clk6	delay > clock period	4
13	in0[5]	port	9(clk5)	9(clk clk5 clk7 clk0 clk3	No	0:7	clk1	clk0 clk4 clk5 clk6	delay > clock period	4
14	in0[7]	port	9(clk5)	9(clk clk5 clk7 clk0 clk3	No	0:7	clk1	clk0 clk4 clk5 clk6	delay > clock period	4
15	in1[2]	port	9(clk5)	9(clk clk5 clk7 clk0 clk3	No	0:7	clk1	clk0 clk4 clk5 clk6	delay > clock period	4
16	in2[4]	port	87(clk5)	clk5	Yes	0:7	clk1	clk5	delay > clock period	5
17	in2[7]	port	9(clk5)	9(clk clk5 clk7 clk0 clk3	No	0:7	clk1	clk0 clk4 clk5 clk6	delay > clock period	4

Messages: Displayed: 40 Total: 40

Similarly, you can explore issues with other constraint types.

Related Reports

[SpyGlass Constraints Reports](#)

Timing Constraints Coverage Report

The *Timing Constraints Coverage* report displays a snapshot of timing constraints coverage status of the design in all modes. By viewing this report, you can also identify the design objects (registers and ports) that are not covered in all modes or are totally uncovered (not covered in any mode) for a design. In addition, this report provides information on uncovered and covered timing paths.

For more information, review the following sections:

- [Generate the Timing Constraints Coverage Report](#)
- [View the Timing Constraints Coverage Report](#)
- [Type of Information in the Timing Constraints Coverage Report](#)
- [Interpret the Timing Constraints Coverage Report](#)
- [Related Reports](#)

Generate the Timing Constraints Coverage Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** `sdc_audit`
- **Run Rule:**
 - SDC_Coverage* — *Compute design coverage and report uncovered design objects and timing paths*
 - Parameters: None

View the Timing Constraints Coverage Report

After generation, the following message appears:

[INFO] Timing coverage report for design/block <name> prepared

You can view the report by one of the following methods:

- **Spreadsheet Viewer (Recommended):** In the Console GUI Message Tree, double-click the message.
- **Reports Menu:** View from the Reports menu in the Console GUI. The report is displayed in `.rpt` format.

- **Report Location:** Open the report directly from `spyglass_constraints/reports`. The report is displayed in `.csv` format.

Type of Information in the Timing Constraints Coverage Report

This report has the following worksheets:

- [Timing Constraints Coverage Summary Worksheet](#)
- [Timing Constraints Ports Coverage Worksheet](#)
- [Timing Constraints Registers Coverage Worksheet](#)
- [Timing Constraints Timing Paths Coverage Worksheet](#)

The format of the Timing Constraints Coverage report for multimode analysis is the same as for single-mode analysis.


Timing Constraints Coverage Summary Worksheet

The `<design-name>CovSummary.csv` file displays the SDC coverage percentage for each mode and the SDC coverage percentage of the design considering all modes together. In addition, the number of uncovered timing paths are displayed.

For ports, the SDC coverage percentage for the complete design, considering all modes together, and the SDC coverage percentage for each mode in the design is reported.

For registers, the SDC coverage percentage for the complete design of each mode and the SDC coverage percentage of each block instance for each mode is listed. SDC Coverage percentage of All Modes Together is shown for the complete design and each block instance. The following information is available:

- **Design Object Type:** The design object type (Ports and Registers) and the name of the first-level blocks in the design
- **Percentage Coverage (All Modes Together):** The SDC Coverage percentage of the design/block in all modes together
- **Percentage Coverage (Mode <mode-name>):** The SDC Coverage percentage of the design/block in mode `<mode-name>`

To view the percentage coverage of submodules, click the  icon to expand the design hierarchy. Click each submodule to open the [Timing Constraints Registers Coverage Worksheet](#) at the point where the SDC

Timing Constraints Coverage Report

coverage details for that module are reported.

When you click the Ports (design-name) and Registers (design-name), the [Timing Constraints Ports Coverage Worksheet](#) and [Timing Constraints Registers Coverage Worksheet](#) are opened. In addition, clicking the submodule name in the expanded form, opens the Timing Constraints Registers Coverage worksheet at the point where the SDC coverage details for that module are listed.

The Timing Constraints Coverage Summary worksheet is shown as follows.

The screenshot shows a 'Spreadsheet Viewer' window with a menu bar (File, View, Tools, Help) and a toolbar. Below the toolbar are four tabs: 'top_CovSummary.csv', 'top_uncovPorts.csv', 'top_uncovRegister.csv', and 'top_timing_paths.csv'. The 'top_CovSummary.csv' tab is active, showing a table with the following data:

	A	B	C	D	E
	Design Object Type	Percentage Coverage (All Modes Together)	Percentage Coverage (mode default)	Percentage Coverage (mode b)	Percentage Coverage (mode c)
1	Ports (top)	37.68%	21.74%	17.39%	17.39%
2	Registers (top)	19.05%	9.52%	19.05%	19.05%
	2 Timing Paths are UNCOVERED				

At the bottom of the spreadsheet, a status bar indicates: 'Messages: Grouped: 2 Displayed: 5 Total: 5'.

Timing Constraints Ports Coverage Worksheet

The `<design-name>_uncovPorts.csv` file displays the SDC Coverage information of all the ports of the design for each mode. In addition, the the SDC Coverage percentage in all modes (All Modes Together) is also shown. The following information is available:

- **Port:** Shows the name of the port
- **Type:** Shows the type of the port, such as input, output, or inout

- **Coverage (All Modes Together):** Shows if the port is covered in at least one mode
- **Coverage (mode <mode-name>):** Shows the SDC Coverage status of the port in mode <mode-name>

The Timing Constraints Ports Coverage worksheet is shown as follows.

Spreadsheet Viewer

File View Tools Help

top_CovSummary.csv top_uncovPorts.csv top_uncovRegister.csv

Show Header

value=

	A	B	C	D	E	F
	Port	Type	Coverage(All modes together)	Coverage (mode default)	Coverage (mode b)	Coverage (mode c)
1	in1[0]	Input	Yes	Covered: Set Case Analysis	Uncovered: Hanging Port	Uncovered: Hanging Port
2	in1[1]	Input	Yes	Covered: Set Case Analysis	Uncovered: Hanging Object,Const/Blocked Pin found,No Input Delay	Uncovered: Hanging Object,Const/Blocked Pin found,No Input Delay
3	in1[2]	Input	Yes	Covered: Set Case Analysis	Uncovered: No Input Delay	Uncovered: No Input Delay
4	in1[3]	Input	Yes	Covered: Set Case Analysis	Uncovered: Const/Blocked Pin found,No Input Delay	Uncovered: Const/Blocked Pin found.No Input Delay

Messages: Displayed: 32 Total: 32

Timing Constraints Registers Coverage Worksheet


The <design-name>_uncovRegister.csv file displays the SDC Coverage percentage of all the registers in each mode and for all modes together. The information is shown for the complete design and is classified based on the design hierarchy.

The following information is available:

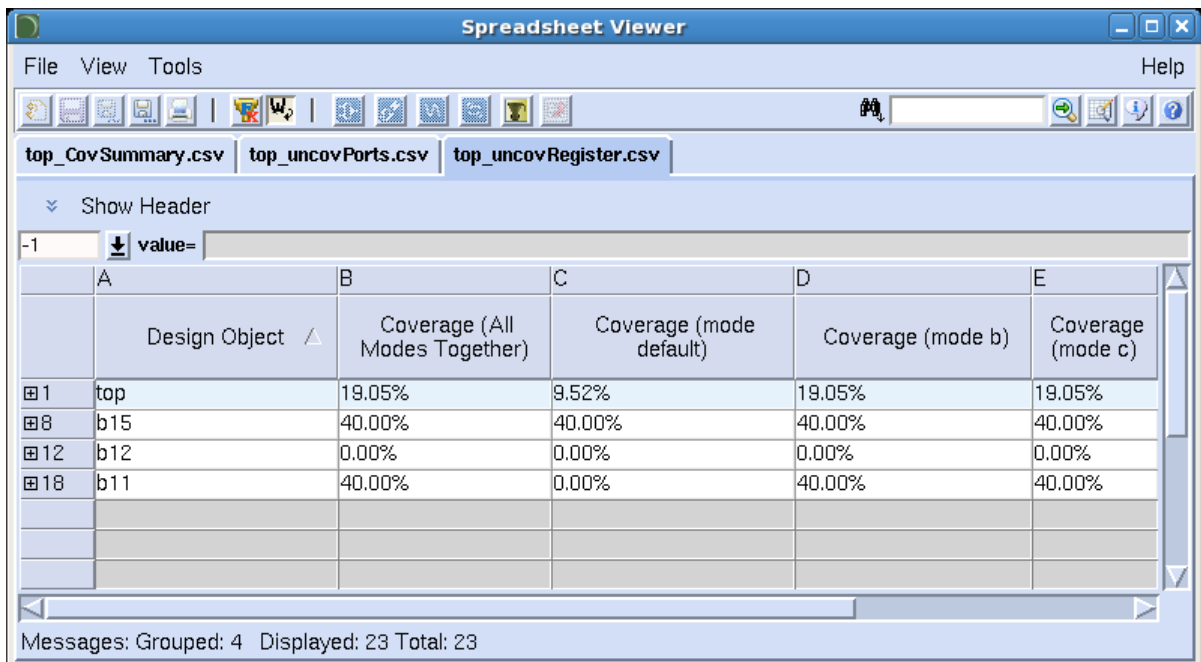
- **Design Object:** Shows the name of the clock pin.

Timing Constraints Coverage Report

- **Coverage (All Modes Together):** Shows if the clock pin is covered in at least one mode.
- **Coverage (mode <mode-name>):** Shows the SDC Coverage status of the clock pin in mode <mode-name>.

To view the detailed coverage status of the registers in a module, click the  icon to expand the submodule name.

The Timing Constraints Registers Coverage worksheet is shown as follows.



The screenshot shows a 'Spreadsheet Viewer' window with a menu bar (File, View, Tools, Help) and a toolbar. The active file is 'top_uncovRegister.csv'. The table below shows the coverage data for various design objects.

	A	B	C	D	E
	Design Object	Coverage (All Modes Together)	Coverage (mode default)	Coverage (mode b)	Coverage (mode c)
1	top	19.05%	9.52%	19.05%	19.05%
8	b15	40.00%	40.00%	40.00%	40.00%
12	b12	0.00%	0.00%	0.00%	0.00%
18	b11	40.00%	0.00%	40.00%	40.00%

Messages: Grouped: 4 Displayed: 23 Total: 23

Timing Constraints Timing Paths Coverage Worksheet

The <design-name>_timing_paths.csv file reports the timing paths that are uncovered in all the modes or partially covered in some of the modes. A timing path can be falsified using [set_false_path](#) or [set_clock_groups](#).

The information is shown for the complete design and is classified based on the design hierarchy.

The following information is available:

- **Timing Path:** Shows the timing path covered or uncovered.
- **Coverage (All Modes Together):** Shows UNCOVERED if the timing path is falsified in all modes. Shows COVERED if the timing path is not falsified in at least one mode.
- **Reason:** Displays the reason for coverage status.
- **Exception <ModeName>:** Displays the [set_false_path](#) or [set_clock_groups](#) constraint specification in the mode, if specified.

The Timing Constraints Timing Paths Coverage worksheet is shown as follows.

	B	C	D	E	F	G
	Timing Path	Coverage (All Modes Together)	Reason	Exception (mode1)	Exception (mode2)	Exception (mode3)
1	f0/CP->f1/D	UNCOVERED	Path falsified in all the modes.	set_clock_groups -group { clk11 } -group { clk2 }	set_false_path -from k1 -to k2	set_false_path -from k1 -to k2
2	-from in3 -to f4/D	UNCOVERED	Path falsified in all the modes.	set_false_path -from in3	set_false_path -from in3 -to f4/D	set_false_path -to f4/D
3	-from f5/CP	COVERED	NO Timing Exception in atleast one mode 'mode2'.	set_false_path -from f5/CP	-	-

Messages: Displayed: 3 Total: 3

Interpret the Timing Constraints Coverage Report

You can use the *Timing Constraints Coverage* report to determine design objects, such as ports and registers, that are not covered and timing paths, which are falsified. In Spreadsheet Viewer, all partially covered or uncovered ports and registers are linked back to the SDC, SGDC, or RTL file. For example, if a register is not getting a clock, it is linked back to the:

- Corresponding violation message generated by the [Clk_Gen01](#) rule
- Corresponding line in the SDC file

This enables efficient debugging and resolution. Similarly, all falsified timing paths are linked back to the [set_false_path](#) or [set_clock_groups](#) constraints specification.

In this section, an example is used to illustrate how this report can be used to investigate partially covered or totally uncovered ports and registers in a design. In addition, the timing paths, which are falsified in one or more modes are discussed.

Review the following sections:

- [View Summary of Timing Constraints Coverage](#)
- [Discover Partially Covered or Totally Uncovered Ports](#)
- [Discover Partially Covered or Totally Uncovered Registers](#)
- [Discover Uncovered and Covered Timing Paths](#)

View Summary of Timing Constraints Coverage

When you open the report in Spreadsheet Viewer, the [Timing Constraints Coverage Summary Worksheet](#) is displayed, as shown in the following screenshot.

Spreadsheet Viewer

File View Tools Help

top_CovSummary.csv top_uncovPorts.csv top_uncovRegister.csv top_timing_paths.csv


Show Header

value=

	A	B	C	D	E
	Design Object Type	Percentage Coverage (All Modes Together)	Percentage Coverage (mode default)	Percentage Coverage (mode b)	Percentage Coverage (mode c)
1	Ports (top)	37.68%	21.74%	17.39%	17.39%
2	Registers (top)	19.05%	9.52%	19.05%	19.05%
	2 Timing Paths are UNCOVERED				

Messages: Grouped: 2 Displayed: 5 Total: 5

This worksheet displays the percentage coverage for ports and registers.

- **Ports:** This worksheet shows that the percentage coverage of ports for All Modes Together is 37.68%. The modes that have the least coverage are Mode b and Mode c, both have 17.39%. To investigate the coverage in these modes, click the **Ports (top)** link located in cell A1. This link opens the *Timing Constraints Ports Coverage* worksheet. See to [Discover Partially Covered or Totally Uncovered Ports](#) for details.
- **Registers:** Since registers information is classified per the design hierarchy, click the  icon to view the coverage of registers in each submodule, as shown in the following worksheet. See to [Discover Partially Covered or Totally Uncovered Registers](#) for details.
- **Timing Paths:** In this worksheet, two uncovered timing paths are reported. Click the cell for more details. See to [Discover Uncovered and Covered Timing Paths](#) for details.

Timing Constraints Coverage Report

Spreadsheet Viewer

File View Tools Help

top_CovSummary.csv top_uncovPorts.csv top_uncovRegister.csv

Show Header

-1 value=

	A	B	C	D	E
	Design Object Type	Percentage Coverage (All Modes Together)	Percentage Coverage (mode default)	Percentage Coverage (mode b)	Percentage Coverage (mode c)
1	Ports (top)	37.68%	21.74%	17.39%	17.39%
2	Registers (top)	19.05%	9.52%	19.05%	19.05%
3	top.b11	40.00%	0.00%	40.00%	40.00%
4	top.b12	0.00%	0.00%	0.00%	0.00%
5	top.b15	40.00%	40.00%	40.00%	40.00%

Messages: Grouped: 2 Displayed: 5 Total: 5

The worksheet shows that the default mode has the least coverage for all submodules and the **top.b15** submodule is covered equally in all modes.

To view the coverage details of each submodule, click the module name. For example, click **top.b15** located in cell A5 to open the *Timing Constraints Registers* worksheet at the point where the SDC coverage details for that module are reported. Refer to [Discover Partially Covered or Totally Uncovered Registers](#) for details.

Discover Partially Covered or Totally Uncovered Ports

To view the uncovered ports or partially covered ports, click the **<design-name>uncovPorts.csv** tab or click the **Ports** link from the Timing Constraints Coverage Summary worksheet. The Timing Constraints Ports Coverage worksheet is displayed, as shown.

	A	B	C	D	E	F
	Port	Type	Coverage(All modes together)	Coverage (mode default)	Coverage (mode b)	Coverage (mode c)
1	in1[0]	Input	Yes	Covered: Set Case Analysis	Uncovered: Hanging Port	Uncovered: Hanging Port
2	in1[1]	Input	Yes	Covered: Set Case Analysis	Uncovered: Hanging Object,Const/Blocked Pin found,No Input Delay	Uncovered: Hanging Object,Const/Blocked Pin found,No Input Delay
3	in1[2]	Input	Yes	Covered: Set Case Analysis	Uncovered: No Input Delay	Uncovered: No Input Delay
4	in1[3]	Input	Yes	Covered: Set Case Analysis	Uncovered: Const/Blocked Pin found,No Input Delay	Uncovered: Const/Blocked Pin found,No Input Delay

Messages: Displayed: 32 Total: 32

Apart from providing coverage details of each port, this worksheet enables you to debug further by displaying all issues in red font. For example, you can investigate the coverage of the **in1[2]** port (cell A3) in Mode b by reading the coverage status, **Uncovered: No Input Delay**, displayed in cell E3. By clicking this cell, the corresponding:

- violation message is displayed in the Console GUI message tree,
- SDC line and/or incremental schematic is displayed.

Update the SDC file accordingly.

You can verify the covering constraints of a port in a mode by clicking the cell that contains the covered status displayed in green font (Refer to column D).

Discover Partially Covered or Totally Uncovered Registers

To view the uncovered registers or partially covered registers from the *Timing Constraints Coverage Summary Worksheet*, click the:

- **<design-name>uncovRegister.csv** tab or the **Registers** link: This shows a collapsed view of each submodule in the *Timing Constraints Registers Coverage* worksheet.
- Click the submodule name from the *Timing Constraints Coverage Summary* worksheet: This navigates you directly to the line where the coverage status of the submodule is detailed.

The screenshot shows a 'Spreadsheet Viewer' window with the following data table:

	A	B	C	D	E
	Design Object	Coverage (All Modes Together)	Coverage (mode default)	Coverage (mode b)	Coverage (mode c)
1	top	19.05%	9.52%	19.05%	19.05%
8	b15	40.00%	40.00%	40.00%	40.00%
9	b15/b22/f2/CP	No	Uncovered: Hanging Object	Uncovered: Hanging Object	Uncovered: Hanging Object
10	b15/b22/f1/CP	No	Uncovered: No Clock Constraint	Uncovered: No Clock Constraint	Uncovered: No Clock Constraint
11	b15/b21/f2/CP	No	Uncovered: Data Pin Constant	Uncovered: Hanging Object	Uncovered: Hanging Object
12	b12	0.00%	0.00%	0.00%	0.00%
18	b11	40.00%	0.00%	40.00%	40.00%

Messages: Grouped: 4 Displayed: 23 Total: 23

In the worksheet displayed above, the expanded view of submodule **b15** is shown.

Apart from providing coverage details of each register, this worksheet enables you to debug further by displaying all issues in red font.

For example, you can investigate the coverage of **b15/b22/f1/CP** register (cell A10) in Mode c by reading the coverage status, **Uncovered: No Clock Constraint**, displayed in cell E10. By clicking this cell, the corresponding:

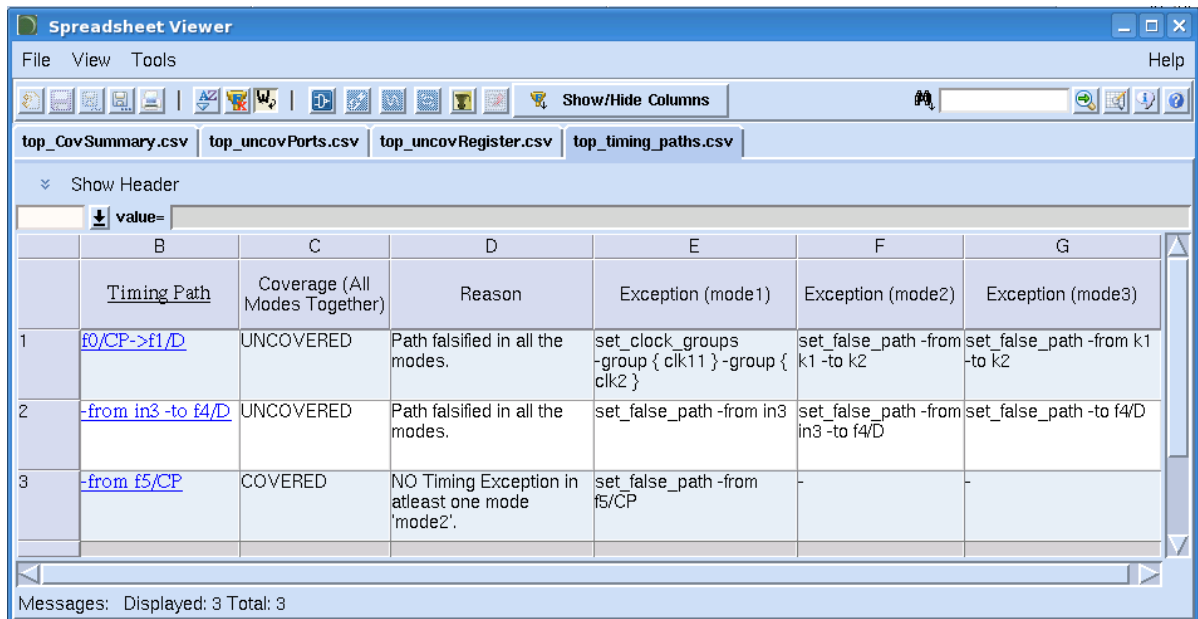
- violation message is displayed in the Console GUI message tree,
- SDC line and/or incremental schematic is displayed.

Update the SDC file accordingly.

You can verify the covering constraints of a register in a mode by clicking the cell that contains the covered status displayed in green font.

Discover Uncovered and Covered Timing Paths

To view the uncovered and covered timing paths, from the *Timing Constraints Coverage Summary* Worksheet, click the **<design-name>_timing_paths.csv** tab or the **Timing Paths** cell in the summary worksheet



Timing Constraints Coverage Report

This worksheet displays the following timing paths which are falsified in all modes as **UNCOVERED**: f0/CP - f1/D and -from in3 -to f4/D. The timing path, -from f5/CP, is falsified in one mode, but not in all modes. Therefore, it is reported as **COVERED**.

Related Reports

[SpyGlass Constraints Reports](#)

Timing Exception Reports

The following table lists the reports that can help you identify timing exception issues for a design.

TABLE 7 Reports for finding problems with timing exceptions

Task	Report	Generated by	Spreadsheet Viewer Support
View the unconnected points specified in set_false_path	False_Path01 Report Report Name False_Path01	False_Path01	No
View the unconnected points specified in -from/-through/-to of set_multicycle_path	MCP01 Report Report Name MCP01	MCP01	No
View unconnected points specified in -from/-through/-to of set_max_delay	TE_Consis01 Report Report Name TE_Consis01	TE_Consis01	No
View the unconnected points specified in -from/-through/-to of set_min_delay	TE_Consis02 Report Report Name TE_Consis02	TE_Consis02	No

To view other reports that are generated by the SpyGlass Constraints solution, refer to [SpyGlass Constraints Reports](#).

False_Path01 Report

The False_Path01 report displays the list of all points specified in the [set_false_path](#) commands that are not connected.

For more information, review the following sections:

- [Generate the False_Path01 Report](#)
- [View the False_Path01 Report](#)
- [Type of Information in the False_Path01 Report](#)
- [Interpreting the False_Path01 Report](#)
- [Related Reports](#)

Generate the False_Path01 Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** `sdc_exception_struct`
- **Run Rule:**
 - [False_Path01](#) — *Identifies false path reference points that are not connected*
 - Parameters: None

View the False_Path01 Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the False_Path01 Report

This report lists the following information:

- SDC File line number containing the [set_false_path](#) specification.
- SDC File containing the [set_false_path](#) specification.

- List of points for each level in the command that are not connected to any points in next or previous level.

Interpreting the False_Path01 Report

Refer to the following example:

- *Example Code and/or Schematic*

A sample report is as follows.

```
#####
```

```
REPORT DETAILS
```

```
#####
```

```
Report Purpose:
```

```
Prints list of all unconnected points specified in
set_false_path commands.
```

```
Points specified with -from, -through, and -to fields
should be connected to the next or previous level.
```

```
Examples:
```

1. -from {A B C} -to {X Y Z}
 - from {A B C} -through {X Y Z}
 - through {A B C} -through {X Y Z}
 - through {A B C} -to {X Y Z}

```
In all the above cases, A/B/C should be connected to
at least one of X/Y/Z.
```

```
Also, X/Y/Z should be connected to at least one of
A/B/C.
```

2. -from {A B C} -through {P Q R} -to {X Y Z}
 - from {A B C} -through {P Q R} -through {X Y Z}
 - through {A B C} -through {P Q R} -to {X Y Z}

```
In all the above cases, A/B/C should be connected to
at least one of P/Q/R.
```

Timing Exception Reports

Also, P/Q/R should be connected to at least one of A/B/C and at least one of X/Y/Z.

Also, X/Y/Z should be connected to at least one of P/Q/R.

```
3. -from {A B C} -through {P Q R} -through {U V W} -to
   {X Y Z}
```

```
-from {A B C} -through {P Q R} -through {U V W} -
through {X Y Z}
```

```
-through {A B C} -through {P Q R} -through {U V W} -
to {X Y Z}
```

In all the above cases, A/B/C should be connected to at least one of P/Q/R.

Also, P/Q/R should be connected to at least one of A/B/C and at least one of U/V/W.

Also, U/V/W should be connected to at least one of P/Q/R and at least one of X/Y/Z.

Also, X/Y/Z should be connected to at least one of U/V/W.

Report Format:

Lists the following information:

A. SDC File's line number of the `set_false_path` command
 B. SDC File containing the command
 C. List of points for each level in the command that are not connected to any points in next or previous level

```
#####
#===END OF REPORT===
```

Related Reports

[Timing Exception Reports](#)

MCP01 Report

The MCP01 report lists the points, which are not connected, but specified in `-from/-through/-to` arguments of `set_multicycle_path` commands.

For more information, review the following sections:

- [Generate the MCP01 Report](#)
- [View the MCP01 Report](#)
- [Type of Information in the MCP01 Report](#)
- [Interpreting the MCP01 Report](#)
- [Related Reports](#)

Generate the MCP01 Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** `sdc_exception_struct`
- **Run Rule:**
 - MCP01 — Identifies multi-cycle path reference points that are not connected*
 - Parameters: None

View the MCP01 Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the MCP01 Report

This report lists the following information:

- SDC File line number containing the `set_multicycle_path` specification.
- SDC File containing the `set_multicycle_path` specification.

- List of points for each level in the command that are not connected to any points in the next or previous level.

Interpreting the MCP01 Report

Refer to the following example:

- *Example Code and/or Schematic*

A sample report is as follows.

```
#####
```

```
REPORT DETAILS
```

```
#####
```

Report Purpose:

Prints list of all unconnected points specified in set_multicycle_path commands.

Points specified with -from, -through, and -to fields should be connected to the next or previous level.

Examples:

- from {A B C} -to {X Y Z}
 - from {A B C} -through {X Y Z}
 - through {A B C} -through {X Y Z}
 - through {A B C} -to {X Y Z}

In all the above cases, A/B/C should be connected to at least one of X/Y/Z.

Also, X/Y/Z should be connected to at least one of A/B/C.

- from {A B C} -through {P Q R} -to {X Y Z}
 - from {A B C} -through {P Q R} -through {X Y Z}
 - through {A B C} -through {P Q R} -to {X Y Z}

In all the above cases, A/B/C should be connected to at least one of P/Q/R.

Also, P/Q/R should be connected to at least one of A/B/C and at least one of X/Y/Z.

Also, X/Y/Z should be connected to at least one of P/Q/R.

3. -from {A B C} -through {P Q R} -through {U V W} -to {X Y Z}

-from {A B C} -through {P Q R} -through {U V W} -through {X Y Z}

-through {A B C} -through {P Q R} -through {U V W} -to {X Y Z}

In all the above cases, A/B/C should be connected to at least one of P/Q/R.

Also, P/Q/R should be connected to at least one of A/B/C and at least one of U/V/W.

Also, U/V/W should be connected to at least one of P/Q/R and at least one of X/Y/Z.

Also, X/Y/Z should be connected to at least one of U/V/W.

Report Format:

Lists the following information:

A. SDC File's line number of the set_multicycle_path command

B. SDC File containing the command

C. List of points for each level in the command that are not connected to any points in next or previous level

#####

===END OF REPORT===

Timing Exception Reports

Related Reports

[Timing Exception Reports](#)

TE_Consis01 Report

The TE_Consis01 report lists the points, which are not connected, but specified in `-from/-through/-to` arguments of the [set_max_delay](#) commands.

For more information, review the following sections:

- [Generate the TE_Consis01 Report](#)
- [View the TE_Consis01 Report](#)
- [Type of Information in the TE_Consis01 Report](#)
- [Interpreting the TE_Consis01 Report](#)
- [Related Reports](#)

Generate the TE_Consis01 Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** Not applicable
- **Run Rule:**
 - [TE_Consis01](#) — *Reports set_max_delay reference points that are not connected*
 - Parameters: None

View the TE_Consis01 Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the TE_Consis01 Report

This report lists the following information:

- SDC File line number containing the [set_multicycle_path](#) specification.

- SDC File containing the *set_multicycle_path* specification.
- List of points for each level in the command that are not connected to any points in the next or previous level.

Interpreting the TE_Consis01 Report

Refer to the following example:

- *Example Code and/or Schematic*

A sample report is as follows.

```
#####
                        R E P O R T   D E T A I L S
#####
```

Report Purpose:

Prints list of all unconnected points specified in `set_max_delay` commands.

Points specified with `-from`, `-through`, and `-to` fields should be connected to the next or previous level.

Examples:

1. `-from {A B C} -to {X Y Z}`
`-from {A B C} -through {X Y Z}`
`-through {A B C} -through {X Y Z}`
`-through {A B C} -to {X Y Z}`

In all the above cases, A/B/C should be connected to at least one of X/Y/Z.

Also, X/Y/Z should be connected to at least one of A/B/C.

2. `-from {A B C} -through {P Q R} -to {X Y Z}`
`-from {A B C} -through {P Q R} -through {X Y Z}`
`-through {A B C} -through {P Q R} -to {X Y Z}`

In all the above cases, A/B/C should be connected to

at least one of P/Q/R.

Also, P/Q/R should be connected to at least one of A/B/C and at least one of X/Y/Z.

Also, X/Y/Z should be connected to at least one of P/Q/R.

3. -from {A B C} -through {P Q R} -through {U V W} -to {X Y Z}

-from {A B C} -through {P Q R} -through {U V W} -through {X Y Z}

-through {A B C} -through {P Q R} -through {U V W} -to {X Y Z}

In all the above cases, A/B/C should be connected to at least one of P/Q/R.

Also, P/Q/R should be connected to at least one of A/B/C and at least one of U/V/W.

Also, U/V/W should be connected to at least one of P/Q/R and at least one of X/Y/Z.

Also, X/Y/Z should be connected to at least one of U/V/W.

Report Format:

Lists the following information:

- A. SDC File's line number of the set_max_delay command
- B. SDC File containing the command
- C. List of points for each level in the command that are not connected to any points in next or previous level

```
#####
===END OF REPORT===
```

Timing Exception Reports

Related Reports

[*Timing Exception Reports*](#)

TE_Consis02 Report

The TE_Consis02 report lists the points, which are not connected, but specified in `-from/-through/-to` arguments of the [set_min_delay](#) commands.

For more information, review the following sections:

- [Generate the TE_Consis02 Report](#)
- [View the TE_Consis02 Report](#)
- [Type of Information in the TE_Consis02 Report](#)
- [Interpreting the TE_Consis02 Report](#)
- [Related Reports](#)

Generate the TE_Consis02 Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** Not applicable
- **Run Rule:**
 - [TE_Consis02](#) — [Reports unconnected set_min_delay reference points](#)
 - Parameters: None

View the TE_Consis02 Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the TE_Consis02 Report

This report lists the following information:

- SDC File line number containing the [set_min_delay](#) specification.
- SDC File containing the [set_min_delay](#) specification.

- List of points for each level in the command that are not connected to any points in next or previous level.

Interpreting the TE_Consis02 Report

Refer to the following example:

- *Example Code and/or Schematic*

A sample report is as follows.

```
#####
```

```
REPORT DETAILS
```

```
#####
```

Report Purpose:

Prints list of all unconnected points specified in set_min_delay commands.

Points specified with -from, -through, and -to fields should be connected to the next or previous level.

Examples:

1. -from {A B C} -to {X Y Z}
 - from {A B C} -through {X Y Z}
 - through {A B C} -through {X Y Z}
 - through {A B C} -to {X Y Z}

In all the above cases, A/B/C should be connected to at least one of X/Y/Z.

Also, X/Y/Z should be connected to at least one of A/B/C.

2. -from {A B C} -through {P Q R} -to {X Y Z}
 - from {A B C} -through {P Q R} -through {X Y Z}
 - through {A B C} -through {P Q R} -to {X Y Z}

In all the above cases, A/B/C should be connected to at least one of P/Q/R.

Also, P/Q/R should be connected to at least one of A/B/C and at least one of X/Y/Z.

Also, X/Y/Z should be connected to at least one of P/Q/R.

```
3. -from {A B C} -through {P Q R} -through {U V W} -to
   {X Y Z}
```

```
-from {A B C} -through {P Q R} -through {U V W} -
through {X Y Z}
```

```
-through {A B C} -through {P Q R} -through {U V W} -
to {X Y Z}
```

In all the above cases, A/B/C should be connected to at least one of P/Q/R.

Also, P/Q/R should be connected to at least one of A/B/C and at least one of U/V/W.

Also, U/V/W should be connected to at least one of P/Q/R and at least one of X/Y/Z.

Also, X/Y/Z should be connected to at least one of U/V/W.

Report Format:

Lists the following information:

A. SDC File's line number of the set_min_delay command

B. SDC File containing the command

C. List of points for each level in the command that are not connected to any points in next or previous level

```
#####
```

```
===END OF REPORT===
```

Related Reports

[Timing Exception Reports](#)

Constraints Equivalence Reports

The following table lists the reports that can help you check the equivalence of SDC constraints in two modes.

TABLE 8 Reports that are generated by equivalence checking rules.

Purpose	Report	Generated by	Spreadsheet Viewer Support
View information about the matched and ambiguous clocks specified in reference and implement	<i>Clock-Mapping Report</i> Report Name Clock-Mapping	<i>Equiv_SDC, Equiv_SDC_Block, Equiv_SDC_Dual_Design, or Equiv_SDC_Top</i>	No
View information about the sequential elements (registers) and ports for which the mapping information is not provided in the mapping file	<i>EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT</i> Report Name EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT	<i>Equiv_SDC_Dual_Design</i>	No
Analyze the equivalence between SDC constraints in the two modes	<i>Equivalence Report</i>	<i>Equiv_SDC, Equiv_SDC_Block, Equiv_SDC_Dual_Design, Equiv_SDC_Top</i>	Yes

To view other reports that are generated by the SpyGlass Constraints solution, refer to [SpyGlass Constraints Reports](#).

Clock-Mapping Report

The Clock-Mapping report contains information about the equivalence between clocks based on delay in reference and implement. Use this report to verify the automatic resolution of real and virtual clocks when multiple clocks have the same characteristics. In addition, use this report to identify the list of clocks that have some uncertainty in matching.

For more information, review the following sections:

- [Generate the Clock-Mapping Report](#)
- [View the Clock-Mapping Report](#)
- [Type of Information in the Clock-Mapping Report](#)
- [Interpret the Clock-Mapping Report](#)
- [Related Reports](#)

Generate the Clock-Mapping Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goals:** `sdc_equiv`, `sdc_hier_equiv`, or `sdc_dual_design_equiv`
- **Run Rule:**
 - Rule Name and Title (any one of the following rules):
 - ◆ [Equiv_SDC](#) — *Performs equivalence between two SDC files*
 - ◆ [Equiv_SDC_Block](#) — *Performs equivalence between the top-level and block-level design units*
 - ◆ [Equiv_SDC_Dual_Design](#) — *Performs equivalence between two SDC files of two equivalent designs, or*
 - ◆ [Equiv_SDC_Top](#) — *Performs equivalence between the top-level design unit*
 - Parameters (Optional):
 - ◆ [equiv_sdc_ambiguous_clock_file](#): *Default value is none. Set this parameter to the name of a file containing ambiguous clocks.*

View the Clock-Mapping Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the Clock-Mapping Report

The Clock-Mapping Report lists the following information:

- List of matched real clocks
- List of matched virtual clocks
- List of ambiguous virtual clocks
- List of matched clocks with no flip-flop in fan-out

Interpret the Clock-Mapping Report

Refer to the following section:

- [Specifying the Skeleton SDC File to Check SDC Equivalence](#)

A sample report is shown below.

```
*****
Clock-mapping report for Reference {<ref_design_name>} and Implement
{<imp_design_name>}
=====
List of matched real clocks (reference file:line {reference clock-
object} {implement clock-object} {reference clock-name:clock-
wvformtype:polarity} {implement clock-name:clock-wvformtype:polarity}
implement file:line)
=====
ref.sdc 1 C1 C1 clk1:default_waveform:positive
clk11:default_waveform:positive imp.sdc 1
ref.sdc 2 C2 C2 clk11:default_waveform:positive
```

```
clk21:default_waveform:positive imp.sdc 2
```

```
=====
```

```
List of matched virtual clocks {reference file:line reference clock-
name} {implement clock-name} implement file:line)
```

```
=====
```

```
ref.sdc 3 vclk1 vclk11 imp.sdc 3
```

```
ref.sdc 4 vclk11 vclk21 imp.sdc 4
```

```
List of ambiguous real clocks ({list of reference file-name and clock-
name } {list of implement file-name and clock-name}):
```

```
=====
```

```
{ref.sdc 1 clk1, ref.sdc 2 clk2} {imp.sdc 1 clk11, imp.sdc 2 clk21}
```

```
List of ambiguous virtual clocks ({list of reference file-name and
clock-name } {list of implement file-name and clock-name}):
```

```
=====
```

```
{ref.sdc 1 vclk1, ref.sdc 2 vclk2} {imp.sdc 1 vclk11, imp.sdc 2 vclk21}
```

```
List of matched clocks (reference file:line {reference clock-object}
{implement clock-object} {reference clock-name:clock-
wvformtype:polarity} {implement clock-name:clock-wvformtype:polarity}
implement file:line) with no flip-flop in fan-out:
```

```
=====
```

```
ref.sdc 1 C1 C1 clk1:default_waveform:positive
```

```
clk11:default_waveform:positive imp.sdc 1
```

```
List of matched clocks after Name Mapping (reference file:line
{reference} {implement} implement file:line):
```

```
=====
```

```
ref.sdc 6 V1 V2 imp.sdc 7
```

Related Reports

[Constraints Equivalence Reports](#)

EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT

This report contains the list of sequential elements (registers) and ports for which mapping information is not provided in the mapping file.

For more information, review the following sections:

- [Generate the EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT](#)
- [View the EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT](#)
- [Type of Information in the EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT](#)
- [Interpret the EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT](#)
- [Related Reports](#)

Generate the EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goals:** `sdc_dual_design_equiv`
- **Run Rule:**
 - Rule Name and Title:
 - ◆ [Equiv_SDC_Dual_Design](#) — *Performs equivalence between two SDC files of two equivalent designs*
 - Parameters (Mandatory):
 - ◆ [equiv_sdc_design_equivalence_file](#): *Default value is none. Set the value to the design equivalence file that provides a report to map the ports, registers, and any intermediate design object used in the SDC files of one design to another.*

View the EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT

After generation, the following messages appear:

[WARNING] Report generated for objects specified in the mapping file but not found in the design

[WARNING] Report generated for objects specified in the mapping file but not found in the design. Since the parameter `ignore_incremental_equivalence` is set, so all the mapping error violation are ignored hence the equivalence results may not be correct

You can view the report by one of the following methods:

- **Spreadsheet Viewer (Recommended)**: In the Console GUI Message Tree, double-click either of the messages listed above.
- **Report Location**: Open the report directly from `spyglass_constraints/reports`.

Type of Information in the EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT

The Clock-Mapping Report lists the following information:

- List of ports/sequential end points in reference whose equivalent design object is not specified.
- List of ports/sequential end points in implement whose equivalent design object is not specified.

Interpret the EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT

Refer to the following section:

- [Example Code and/or Schematic](#)

A sample report is shown below.

List of ports/sequential end points in reference whose equivalent design object is not specified

```
in1
F1
in2..
```

List of ports/sequential end points in implement whose equivalent design object is not specified

-----in1

```
F1
```

Constraints Equivalence Reports

in2..

Related Reports

[Constraints Equivalence Reports](#)

Equivalence Report

You can use this report to analyze the equivalence between SDC constraints in the two modes.

For more information, review the following sections:

- [Generate the Equivalence Report](#)
- [View the Equivalence Report](#)
- [Type of Information in the Equivalence Report](#)
- [Interpret the Equivalence Report](#)
- [Related Reports](#)

NOTE: For information on performing equivalence between SDC of each of the modes and the mode merged SDC, see [SDC_multimode_equiv](#).

Generate the Equivalence Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** `sdc_equiv`, `sdc_hier_equiv`, or `sdc_dual_design_equiv`
- **Run Rule:**
 - Rule Name and Title (any one of the following rules):
 - ◆ [Equiv_SDC](#) — *Performs equivalence between two SDC files*
 - ◆ [Equiv_SDC_Block](#) — *Performs equivalence between the top-level and block-level design units*
 - ◆ [Equiv_SDC_Dual_Design](#) — *Performs equivalence between two SDC files of two equivalent designs, or*
 - ◆ [Equiv_SDC_Top](#) — *Performs equivalence between the top-level design unit*
 - Parameters (Mandatory):
 - ◆ [equiv_sdc_ignore_unequivalent_design_obj](#): *Default value is yes. This indicates that the [Equiv_SDC_Dual_Design](#) rule ignores the design object that does not have an equivalent design object in the equivalence report. Set this parameter to no if you want the [Equiv_SDC_Dual_Design](#) rule not to ignore such design objects.*

View the Equivalence Report

After generation, the following messages appear:

[WARNING] Equivalence report is prepared for block "<block-name>"

[WARNING] Equivalence report is prepared for instance <instance-name> of block "<block-name>"

You can view the report by one of the following methods:

- **Spreadsheet Viewer (Recommended)**: In the Console GUI Message Tree, double-click either of the above messages.
- **Report Location**: Open the report directly from `spyglass_constraints/reports`.

Type of Information in the Equivalence Report

You can view this report by using the Spreadsheet Viewer. Each constraint for which equivalence is being performed is displayed as a separate tab in Spreadsheet Viewer. Each tab has the **Equivalent** and **Analysis** columns. The other column headers may vary with the mode and the type of constraint.

Interpret the Equivalence Report

This section describes the key points pertaining to:

- *set_case_analysis Equivalence*
- *Clock Equivalence*
- *Input/Output Delay Equivalence*
- *Timing Exceptions Equivalence*

set_case_analysis Equivalence

For *set_case_analysis* constraints, exact equivalence cannot be performed. Therefore, the equivalent end-point/timing-path are not reported. The equivalence report does not show equivalent commands in reference and implement SDC files.

The following is reported in the Equivalent column:

- **Yes:** Constraints are equivalent.
- **No:** Constraints are not equivalent.

A sample report is as follows:

	B	C	D	E	F	G	H	I
	Equivalent	Analysis	Constant Values in block	Constant Values in flatTop	Impact Point in block	Impact Point in flatTop	Constant Value/Clock Reaching the End-point in block	Constant Value/Clock Reaching the End-point in flatTop
1	No	Constant Values defined only in flatTop		in3:0,in4:0	out3,out4	out3,out4	X	0
4	No	Constant Values defined only in block	in7:1,in8:1		out7,out8	out7,out8	1	X
7	No	Constant Values defined in both the modes	in5:1,in6:1	in5:0,in6:0	out5,out6	out5,out6	1	0

Messages: Grouped: 3 Displayed: 9 Total: 9

To not display the display messages of type **specified only in flatTop** (Row 1 in the figure above) for the top-block equivalence check in the *Equivalence Report*, set the [equiv_sdc_ignore_sca_for_top_block_equivalence](#) parameter to `yes`.

For more information, refer to the following sections:

- [Performing Equivalence between set_case_analysis Constraints](#)
- [Example Code and/or Schematic](#)

Clock Equivalence

For clocks, the following is reported:

- **Yes:** Clocks are equivalent.

Constraints Equivalence Reports

- **No:** Clocks are not equivalent.
- **Ambiguous:** Equal number of multiple clocks are in both the modes, such that the exact one-to-one clock mapping is not possible. This status means that the exact clock equivalence was not established. The equivalence does not stop in the presence of ambiguous or equivalent clocks. The ambiguity is resolved for further stages of equivalence by random assignment. To resolve the ambiguity, use the [equiv_sdc_ambiguous_clock_file](#) parameter.

A sample Equivalence report of clock equivalence is as follows:

	B	C	D	E	F	G
	Equivalent	Analysis	Clocks in reference mode	Clocks in implement mode	reference Design Object	implement Design Object
B1	Yes	Clocks are Equivalent	r1:default_wavemode:positive	i1:default_wavemode:positive i2:default_wavemode:positive i3:default_wavemode:positive	1/CP,2/CP,3/CP	1/CP,2/CP,3/CP
2	Yes	implement clock is equivalent to reference clock but differ in name and the object on which it is defined.	r1:default_wavemode:positive	i1:default_wavemode:positive	1/CP	1/CP
3	Yes	implement clock is equivalent to reference clock but differ in name and the object on which it is defined.	r1:default_wavemode:positive	i2:default_wavemode:positive	2/CP	2/CP
4	Yes	implement clock is equivalent to reference clock but differ in name and the object on which it is defined.	r1:default_wavemode:positive	i3:default_wavemode:positive	3/CP	3/CP

Messages: Grouped: 1 Displayed: 4 Total: 4

For more information, refer to the following sections:

- [Performing Equivalence between Clock Constraints](#)
- [Example 1: One-to-Many Clock Mapping](#)
- [Example 2: Many-to-Many Clock Mapping](#)

Input/Output Delay Equivalence

Clock equivalence is established on the basis of I/O delay specified at this stage for both real and virtual clocks. This information along with clock relationship established in clock equivalence is used for further equivalence checks.

For I/O delay equivalence, the following is reported:

- **Yes:** The I/O delays are equivalent.
- **No:** The I/O delays are not equivalent.
- **Redundant:** Multiple I/O delays have been specified at the same port and the clocks are equivalent (within the mode). Therefore, delays specified with reference to the clock pairs other than the first pair are considered redundant and the clock relationship is not established for these pairs.

A sample report is as follows:

	B		D	E	F	G	H	I	J
	Equivalent	Analysis	Delay Type	reference object	implement object	reference clock	implement clock	reference values	implement values
1	No	Inequivalent Delay	set_input_delay set_output_delay	in1	in1	r2 r1	i4 i3 i1 i2		
6	No	Inequivalent Delay	set_input_delay set_output_delay	in2	in2	r2 r1 r3	i4 i3 i1 i2		

Messages: Grouped: 2 Displayed: 11 Total: 11

For more information, refer to the following sections:

- [Performing Equivalence between Input/Output Delays](#)

Constraints Equivalence Reports

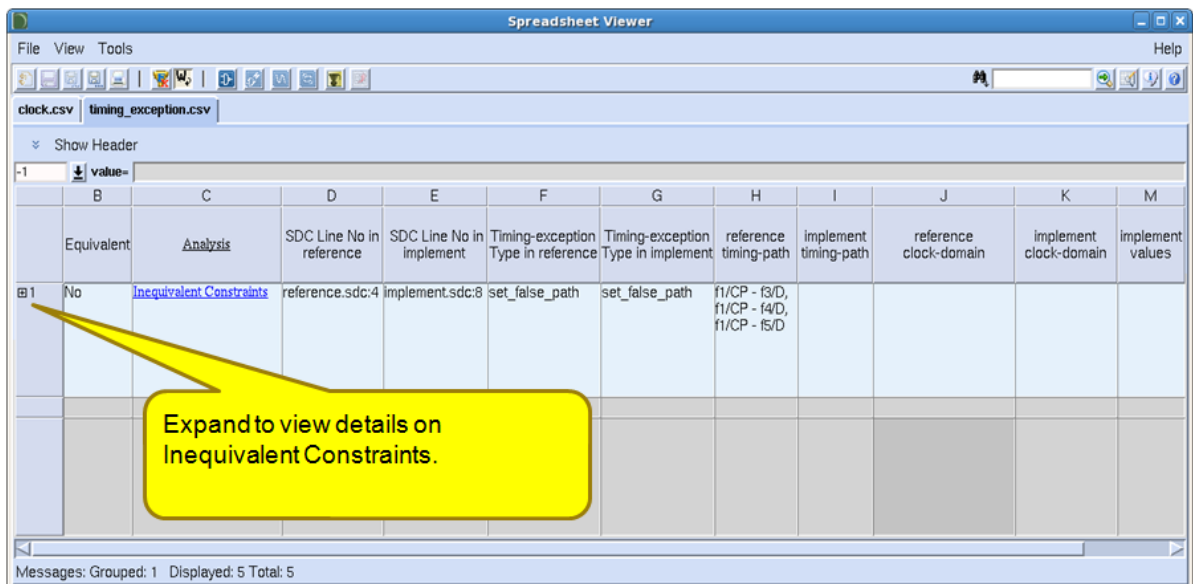
- [Example Code and/or Schematic](#)

Timing Exceptions Equivalence

For timing exceptions equivalence, the following is reported:

- **Yes**: Constraint is fully equivalent.
- **No**: Constraint is not equivalent.

A sample report is as follows:



	B	C	D	E	F	G	H	I	J	K	M
	Equivalent	Analysis	SDC Line No in reference	SDC Line No in implement	Timing-exception Type in reference	Timing-exception Type in implement	reference timing-path	implement timing-path	reference clock-domain	implement clock-domain	implement values
1	No	Inequivalent Constraints	reference.sdc:4	implement.sdc:8	set_false_path	set_false_path	f1/CP - f3/D, f1/CP - f4/D, f1/CP - f5/D				

Messages: Grouped: 1 Displayed: 5 Total: 5

The following shows the expanded view. Though there is a constraint that has an equivalent status of **Yes**, there are some constraints that are not equivalent. Therefore, the overall equivalent status is **No**.

	B	C	E	F	G	H	I	J	K	M	
	Equivalent	Inequivalent Constraints	SDC Line No in reference	SDC Line No in Implement	Timing-exception Type in reference	Timing-exception Type in Implement	reference timing-path	Implement timing-path	reference clock-domain	Implement clock-domain	implement values
1	No	Inequivalent Constraints	reference.sdc:4	implement.sdc:8	set_false_path	set_false_path	f1/CP - f3/D, f1/CP - f4/D, f1/CP - f5/D				
2	No	Specified only in reference	reference.sdc:4		set_false_path		f1/CP - f3/D		(from clock r1 with positive transitioning, to clock r2 with positive transitioning)		
3	No	Specified only in reference	reference.sdc:4		set_false_path		f1/CP - f4/D		(from clock r1 with positive transitioning, to clock r2 with positive transitioning)		
4	No	Specified only in reference	reference.sdc:4		set_false_path		f1/CP - f5/D		(from clock r1 with positive transitioning, to clock r2 with positive transitioning)		
5	Yes	Constraint is equivalent in both the modes	reference.sdc:4	implement.sdc:8	set_false_path	set_false_path					

Messages: Grouped: 1 Displayed: 5 Total: 5

For more information, refer to the following sections:

- [Performing Equivalence between Timing Exceptions](#)
- [Example Code and/or Schematic](#)

For all other constraints, the equivalent and non equivalent constraints are reported.

Related Reports

[Constraints Equivalence Reports](#)

Miscellaneous Reports

The following table lists the reports that can help you identify various issues related to constraints in a design.

TABLE 9 Miscellaneous Reports

Task	Report	Generated by	Spreadsheet Viewer Support
Lists modules with the <i>set_dont_touch</i> constraint	<i>tc_dont_touch_info Report</i>	<i>SDC_Report 03</i>	No
	Report Name tc_dont_touch_info		
Lists cells with the <i>set_dont_use</i> or <i>dont_use</i> attribute set	<i>tc_dont_use_info Report</i>	<i>SDC_Report 04</i>	No
	Report Name tc_dont_use_info		
Determine CDC that have different clocks	<i>clk_domain_false_path Report</i>	<i>False_Path0 7</i>	No
	Report Name clk_domain_false_path		
Identify domain crossing synchronous clocks that do not have <i>set_clock_uncertainty</i>	<i>sync_clock_uncertainty Report</i>	<i>Clk_Uncert0 3</i>	No
	Report Name sync_clock_uncertainty		

TABLE 9 Miscellaneous Reports

Task	Report	Generated by	Spreadsheet Viewer Support
Identify sequential cells that have clock pins driven by the reported clock net	tc_clk_gen01a_info Report	Clk_Gen01a	No
	Report Name tc_clk_gen01a_info		
Identify all sequential cells that have clock pins driven by the reported constant or hanging clock net	tc_clk_gen01b_info Report	Clk_Gen01b	No
	Report Name tc_clk_gen01b_info		
View all generated clocks and source clocks	tc_clock_info Report	SDC_Report 01	No
	Report Name tc_clock_info		
View the sum of source and network latencies of clocks	tc_latency_info Report	Clk_Lat03	No
	Report Name tc_latency_info		

TABLE 9 Miscellaneous Reports

Task	Report	Generated by	Spreadsheet Viewer Support
Identify clock paths comprising the feedback path	<i>SDC_Methodology31 Report</i>	<i>SDC_Methodology31</i>	No
	Report Name SDC_Methodology31		
Determine the domain relationships between clocks used by <i>set_multicycle_path</i>	<i>mcp_domain_<integer> Report</i>	<i>CheckMCP</i>	No
	Report Name mcp_domain_<integer>		

To view other reports that are generated by the SpyGlass Constraints solution, refer to [SpyGlass Constraints Reports](#).

tc_dont_touch_info Report

The tc_dont_touch_info report displays a list of modules with the [set_dont_touch](#) constraint.

For more information, review the following sections:

- [Generate the tc_dont_touch_info Report](#)
- [View the tc_dont_touch_info Report](#)
- [Type of Information in the tc_dont_touch_info Report](#)
- [Interpret the tc_dont_touch_info Report](#)
- [Related Reports](#)

Generate the tc_dont_touch_info Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** Not applicable
- **Run Rule:**
 - [SDC_Report03](#) — *Prints the list of dont touch module*
 - Parameters: None

View the tc_dont_touch_info Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the tc_dont_touch_info Report

The tc_dont_touch_info report contains the following information:

- Module name
- SDC file name and the line number

A sample report is shown as follows.

Miscellaneous Reports

```

#####
#   Total Number of Generated Messages :           3
#   Number of Waived Messages          :           0
#   Number of Reported Messages        :           3
#
#   Number of Overlimit Messages       :           0
#
#####
#####
# Purpose:
#   Reports list of module with set_dont_touch
#
# Format:
#   Gives the following information:
#       - module name
#       - SDC file name and line number
#####
-----
Module name      Constraints File name  Line number
-----
mylatch         set_dont_touch1.sdc      2

top             set_dont_touch.sdc       2
mylatch1       set_dont_touch.sdc       3
#####

```

Interpret the tc_dont_touch_info Report

In the following report snippet, the `mylatch`, `top`, and `mylatch1` modules have the `set_dont_touch` constraint specified. The report also states the SDC file name and line number that states the `set_dont_touch` command.

```
#####  
-----  
Module name          Constraints File name  Line number  
-----  
mylatch              set_dont_touch1.sdc    2  
  
top                  set_dont_touch.sdc     2  
mylatch1             set_dont_touch.sdc     3  
#####
```

Related Reports

[Miscellaneous Reports](#)

tc_dont_use_info Report

The `tc_dont_use_info` report displays a list of cells with the `set_dont_use` or `dont_use` attribute set.

For more information, review the following sections:

- [Generate the `tc_dont_use_info` Report](#)
- [View the `tc_dont_use_info` Report](#)
- [Type of Information in the `tc_dont_use_info` Report](#)
- [Interpret the `tc_dont_use_info` Report](#)
- [Related Reports](#)

Generate the tc_dont_use_info Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** Not applicable
- **Run Rule:**
 - [SDC_Report04 — Reports instances of the `dont_use` attribute set on a design object](#)
 - Parameters: None

View the tc_dont_use_info Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the tc_dont_use_info Report

The `tc_dont_use_info` report contains the following information:

- The name of the cell and the library if the `dont_use` attribute is set on the cell

- The cell name, library name, the SDC file name, and the line number for the `set_dont_use` command.

Interpret the `tc_dont_use_info` Report

The following information is presented in this report:

- From the `lsi_10k` library, the following cells have the `dont_use` attribute set: `AN2`, `OR2`, `AN2P`, `B1I`, `LS1`, and `LS1P`.
- The `set_dont_use` command instances in the `myand` and `top` blocks.

A sample report is as follows.

```
#####
#      Total Number of Generated Messages :          7
#      Number of Waived Messages          :          0
#      Number of Reported Messages        :          7
#
#      Number of Overlimit Messages       :          0
#
#####
#####
#
# Purpose:
#       Reports list of cells with set_dont_use or dont_use
#       attribute set
#
# Format:
#       Gives the following information:
#       - cell name and library name if dont_use attribute
#       is set on cell
#       - cell name, library name, SDC file name and line
```

Miscellaneous Reports

```

number for
    set_dont_use command
#####
From library:

```

```

-----
Cell Name      Library Name
-----
AN2            lsi_10k
OR2            lsi_10k
AN2P           lsi_10k
B1I            lsi_10k
LS1            lsi_10k
LS1P           lsi_10k

```

Don't use cell for block myand:

```

-----
Cell Name  Library Name  Constraints  File name  Line Number
-----
AN2        lsi_10k        a.sdc        1

```

Don't use cell for block top:

```

-----
Cell Name  Library Name  Constraints  File name  Line Number
-----
OR2        lsi_10k.db:lsi_10k  test11.sdc  1

```

Related Reports

[Miscellaneous Reports](#)

clk_domain_false_path Report

The `clk_domain_false_path` report displays clock domain crossings that have occurred for different clocks and whether a [set_false_path](#) constraint has been specified.

For more information, review the following sections:

- [Generate the clk_domain_false_path Report](#)
- [View the clk_domain_false_path Report](#)
- [Type of Information in the clk_domain_false_path Report](#)
- [Interpret the clk_domain_false_path Report](#)
- [Related Reports](#)

Generate the clk_domain_false_path Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** Not applicable
- **Run Rule:**
 - [False_Path07](#) — *Reports when set_false_path is not specified between two connected asynchronous clock domains*
 - Parameters: None

View the clk_domain_false_path Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the clk_domain_false_path Report

The `clk_domain_false_path` report contains the following information:

- List of source and destination clocks

Miscellaneous Reports

- Whether the *set_false_path* and *set_multicycle_path* constraints (both directions) have been set with both the *-setup* and *-hold* options for each pair of clocks of different domains.

Interpret the *clk_domain_false_path* Report

A sample report is shown as follows.

```
#####
#      Total Number of Generated Messages :          4
#      Number of Waived Messages          :          0
#      Number of Reported Messages        :          4
#      Number of Overlimit Messages       :          0
#####
# Purpose:
#       Reports whether set_false_path is specified between domain
#       crossing asynchronous clocks
# Format:
#       This report lists the following information :
#       - List of source and destination clocks
#       - Whether set_false_path or set_multicycle_path constraints (both
#       directions) has been set for each pair of clocks of different domains
#       with both the options (-setup and -hold).
# Presumption:
#       - All pairs of clocks specified in report are asynchronous
#       - All asynchronous clocks are crossing domain
#####
design/block : top
sdc_data -file falsepath.sdc -corner worst -mode func -level rtl
source clock destination clock false path specified multicycle specified
-----
```

Clk1	GClk2	no	no
Clk2	GClk1	no	no
Clk3	GClk2	no	no

Related Reports

[Miscellaneous Reports](#)

sync_clock_uncertainty Report

The `sync_clock_uncertainty` report displays information about the existence of a path between synchronous clocks and whether the [set_clock_uncertainty](#) constraint has been specified between them.

For more information, review the following sections:

- [Generate the sync_clock_uncertainty Report](#)
- [View the sync_clock_uncertainty Report](#)
- [Type of Information in the sync_clock_uncertainty Report](#)
- [Interpret the sync_clock_uncertainty Report](#)
- [Related Reports](#)

Generate the sync_clock_uncertainty Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** `sdc_check`
- **Run Rule:**
 - [Clk_Uncert03](#) — *Reports inter-clock uncertainty that has not been defined between synchronous clocks*
 - Parameters: None

View the sync_clock_uncertainty Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the sync_clock_uncertainty Report

The `sync_clock_uncertainty` report displays information in the following situations:

- Two crossing clocks are synchronous and uncertainty is specified
- Two crossing clocks are synchronous and *set_false_path/set_clock_groups* is specified between the clocks

The `sync_clock_uncertainty` report does not display information for two clocks or two synchronous clocks that are not crossing.

This report contains the following information:

- List of source and destination clocks
- Whether a *set_clock_uncertainty* (both directions) constraint has been set for each pair of clocks

Interpret the `sync_clock_uncertainty` Report

A sample report is as shown below.

```
#####
#      Total Number of Generated Messages :          2
#      Number of Waived Messages          :          0
#      Number of Reported Messages        :          2
#
#      Number of Overlimit Messages       :          0
#
#####
#####
# Purpose:
#       Reports whether set_clock_uncertainty is specified
#       between domain crossing
#       synchronous clocks
#
# Format:
#   This report lists the following information :
#   - List of source and destination clocks
```

Miscellaneous Reports

```

#       - Whether a set_clock_uncertainty (both directions)
has been set for each
        pair of clocks
# Presumption:
#       - All pairs of clocks specified in report are
synchronous
#       - All synchronous clocks pair are crossing domain
#####
Design: top (top-level)
design/block : top
sdc_data -file clkuncert.sdc -corner worst -mode func -level
rtl
source clock          destination clock    clock uncertainty
specified
-----
Clk1                  Clk3                  no

```

Related Reports

[Miscellaneous Reports](#)

tc_clk_gen01a_info Report

The tc_clk_gen01a_info report displays a list of all sequential cells that have clock pins driven by the reported clock net.

For more information, review the following sections:

- [Generate the tc_clk_gen01a_info Report](#)
- [View the tc_clk_gen01a_info Report](#)
- [Type of Information in the tc_clk_gen01a_info Report](#)
- [Interpret the tc_clk_gen01a_info Report](#)
- [Related Reports](#)

Generate the tc_clk_gen01a_info Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** sdc_check
- **Run Rule:**
 - Clk_Gen01a* — *Identifies sequential elements that are not constrained by a clock*
 - Parameters: None

View the tc_clk_gen01a_info Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from spyglass_constraints/reports.

Type of Information in the tc_clk_gen01a_info Report

The tc_clk_gen01a_info report is divided into the following parts:

Part A

This part contains a list of all unconstrained flip-flops in any mode of

operation.

Part B

This part contains the following:

- Reported clock net name
- List of unconstrained sequential cells driven by the clock net
- List of already constrained (by some user-specified clock) sequential cells driven by the clock net

Interpret the tc_clk_gen01a_info Report

A sample report is as follows.

```
#####
#      Total Number of Generated Messages :          3
#      Number of Waived Messages          :          0
#      Number of Reported Messages       :          3
#
#      Number of Overlimit Messages      :          0
#####
# Purpose:
#      Prints list of all sequential cells whose clock pin is
#      driven by reported
#      clock net
#
# Format:
#      Gives the following information :
#      A. List of all unconstrained flip-flop in any mode
#      of operation
#      B. - Reported clock net name
#      - List of unconstrained sequential cells driven
#      by the clock net
```

```
#           - List of already constrained (by some user
specified clock)
           sequential cells driven by the clock net
#####
A. Flop not constraints in any mode of operation:

1. Unconstraints flip-flop for design/block 'top':
    I1
    I2

B. Clock not driven by a clock constraint:
  1. Clock Net 'clk' of design/block 'top':
    a). Unconstrained 1 sequential cells:
        I1

  2. Clock Net 'I1/QT[1]' of design/block 'top':
    a). Unconstrained 1 sequential cells:
        I2
```

Related Reports

[Miscellaneous Reports](#)

tc_clk_gen01b_info Report

The tc_clk_gen01b_info report displays a list of sequential cells that have clock pins driven by the reported constant or hanging clock net.

For more information, review the following sections:

- [Generate the tc_clk_gen01b_info Report](#)
- [View the tc_clk_gen01b_info Report](#)
- [Type of Information in the tc_clk_gen01b_info Report](#)
- [Interpret the tc_clk_gen01b_info Report](#)
- [Related Reports](#)

Generate the tc_clk_gen01b_info Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** sdc_check
- **Run Rule:**
 - [Clk_Gen01b](#) — *Clock driven by a constant value or hanging*
 - Parameters: None

View the tc_clk_gen01b_info Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from spyglass_constraints/reports.

Type of Information in the tc_clk_gen01b_info Report

The tc_clk_gen01b_info report has two parts.

Part A

This part contains a list of all unconstrained flops in any mode of operation.

Part B

This part contains the following:

- Reported clock net name
- List of unconstrained sequential cells driven by the clock net
- List of already constrained, by some user-specified clock, sequential cells driven by the clock net

Interpret the tc_clk_gen01b_info Report

A sample report is as follows.

```
#####
#      Total Number of Generated Messages :          1
#      Number of Waived Messages           :          0
#      Number of Reported Messages         :          1
#      Number of Overlimit Messages        :          0
#
#####
# Purpose:
#      Prints list of all sequential cells whose clock pin is
#      driven by reported constant or hanging clock net
#
# Format:
#      Gives the following information :
#          A. List of all unconstrained flip-flop in any mode
#          of operation
#          B. - Reported clock net name
#              - List of unconstrained sequential cells
#              driven by the clock net
#              - List of already constrained (by some user
#              specified clock)
```

sequential cells driven by the clock net

#####

A. Flop not constraints in any mode of operation:

B. Clock not driven by a clock constraint:

Related Reports

[Miscellaneous Reports](#)

tc_clock_info Report

The `tc_clock_info` report displays a list of generated clocks and source clocks. For more information, review the following sections:

- [Generate the `tc_clock_info` Report](#)
- [View the `tc_clock_info` Report](#)
- [Type of Information in the `tc_clock_info` Report](#)
- [Interpret the `tc_clock_info` Report](#)
- [Related Reports](#)

Generate the tc_clock_info Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** `sdc_audit` or `sdc_redundancy`
- **Run Rule:**
 - [SDC_Report01](#) — *Reports a constrained clock that is not used as a clock*
 - Parameters: None

View the tc_clock_info Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the tc_clock_info Report

The `tc_clock_info` report contains the following information:

- **Source clock v/s Generated clock**

If a source clock ([create_clock](#) or [create_generated_clock](#)) is defined at the source of a [create_generated_clock](#), both clocks are assumed to be related and shown in the report. However, if [create_clock](#) is not defined at the

source of any *create_generated_clock*, the field related to the *create_generated_clock* is blank.

■ Source clock v/s Virtual clock

If the characteristics of the source clock (*create_clock* only) matches with the characteristics of the virtual clock, both clocks are assumed to be related and shown in the report. If the characteristics of the source clock do not match with the virtual clock, the virtual clock field is blank.

■ Port v/s clock

In the Port v/s clock table, the port can be an input/output/inout port and the clock can be a *create_clock*/*create_generated_clock*/virtual clock.

In the above Port v/s Clock table:

- The first entry shows that a *set_input_delay* constraint is applied on the input port data1 with reference to clock clk1
- The second entry shows that no *set_input_delay* constraint is applied on the input port data2.
- The third entry shows that the clock clk1 is applied on the input port clka
- The fourth entry shows that the clock gclk1 is applied on the output port OP1.
- The fifth entry shows that a *set_output_delay* constraint is applied on the output port OP2 with reference to clock gclk2

Interpret the tc_clock_info Report

A sample report is shown as follows.

```
#####
#      Total Number of Generated Messages :          5
#      Number of Waived Messages          :          0
#      Number of Reported Messages       :          5
#
#      Number of Overlimit Messages      :          0
```

```

#
#####
#
# Purpose:
#   Prints table of Generated clocks and source clocks
#
# Format:
#   Gives the following information :
#     - Source Clock Vs Generated Clock
#     - Source Clock Vs Virtual Clock
#     - Ports, its direction and the clock with respect
to which the
#       delay is specified
#####
*****

Design : top
sdc_data options:  -corner = worst  mode = func
constraint files:  clkinfo.sdc
*****

    Source Clock Vs Generated Clock
+++++
Source Clock          Generated Clock
=====
clk1                  gclk3
                     gclk1
                     gclk2
gclk3                 gclk5
gclk4                 --

```

Miscellaneous Reports

```

gclk1                gclk4
gclk2                --
gclk5                --

```

```

-----
Source Clock Vs Virtual Clock

```

```

+++++
Source Clock          Virtual Clock
=====
clk1                  --

```

```

-----
Port Vs Clock

```

```

+++++
PortName              Type              ClockName
=====
data1                 input              clk1
data2                 input              --
clka[0:5] (clock port) input              clk1
OP1 (clock port)     output            gclk1
OP1                   output            gclk2

```

Related Reports

[Miscellaneous Reports](#)

tc_latency_info Report

The `tc_latency_info` report displays the sum of source and network latencies of clocks.

For more information, review the following sections:

- [Generate the `tc_latency_info` Report](#)
- [View the `tc_latency_info` Report](#)
- [Type of Information in the `tc_latency_info` Report](#)
- [Interpret the `tc_latency_info` Report](#)
- [Related Reports](#)

Generate the `tc_latency_info` Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** Not applicable
- **Run Rule:**
 - `Clk_Lat03` — [Reports synchronous clocks that have different sum of network and source latency](#)
 - Parameters: None

View the `tc_latency_info` Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the `tc_latency_info` Report

The `tc_latency_info` report contains information about all clocks, arranged according to the clock domain, along with the sum of source latency and network latency for various options (`min-max`, `rise-fall`, `early-late`).

Interpret the tc_latency_info Report

A sample reports is as shown below.

```
#####
#      Total Number of Generated Messages :          2
#      Number of Waived Messages          :          0
#      Number of Reported Messages        :          2
#
#      Number of Overlimit Messages        :          0
#
#####
```

Design: top

Domain	Condition
source_latency+latency	
Clk1 Clk2 11.00	min-rise-early 12.00
Clk1 Clk2 11.00	min-rise-late 12.00
Clk1 Clk2 11.00	min-fall-early 12.00
Clk1 Clk2 11.00	min-fall-late 12.00
Clk1 Clk2 11.00	max-rise-early 12.00
Clk1 Clk2 11.00	max-rise-late 12.00
Clk1 Clk2 11.00	max-fall-early 12.00
Clk1 Clk2 11.00	max-fall-late 12.00

Domain	Condition
source_latency+latency	
Clk2 Clk1 12.00	min-rise-early 11.00
Clk2 Clk1 12.00	min-rise-late 11.00
Clk2 Clk1 12.00	min-fall-early 11.00
Clk2 Clk1 12.00	min-fall-late 11.00
Clk2 Clk1 12.00	max-rise-early 11.00
Clk2 Clk1 12.00	max-rise-late 11.00
Clk2 Clk1 12.00	max-fall-early 11.00
Clk2 Clk1 12.00	max-fall-late 11.00
#####	

Related Reports

[Miscellaneous Reports](#)

SDC_Methodology31 Report

The SDC_Methodology31 report displays clock paths that have a feedback path.

For more information, review the following sections:

- [Generate the SDC_Methodology31 Report](#)
- [View the SDC_Methodology31 Report](#)
- [Type of Information in the SDC_Methodology31 Report](#)
- [Interpret the SDC_Methodology31 Report](#)
- [Related Reports](#)

Generate the SDC_Methodology31 Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** Not applicable
- **Run Rule:**
 - [SDC_Methodology31](#) — *Reports feedback loops comprising of clock paths*
 - Parameters: None

View the SDC_Methodology31 Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the SDC_Methodology31 Report

The SDC_Methodology31 report comprises of tables. Each table contains the clocks and objects present in each loop. The columns in each table are:

- **Clocks:** List of clocks present in the loop. Contains a comma-separated list of clock names that are defined in the SDC file that constitutes the feedback paths.
- **Design Objects:** List of objects present in loop. Last object in the list signifies the repetition of first object. Contains a list of pins found in the feedback path. The pins are separated by a '-'.

NOTE: *Pins of instances inferred by SpyGlass from RTL are not displayed.*

Interpret the SDC_Methodology31 Report

A sample reports is as follows.

```
#####
Clocks                Design objects
                      f1/CP -
                      f1/Q -
                      a1/A -
                      a1/Z -
                      f2/CP -
C1, C2, C3           f2/Q -
                      a2/A -
                      a2/Z -
                      f3/CP -
                      f3/Q -
                      f1/CP -
#####
```

Related Reports

[Miscellaneous Reports](#)

mcp_domain_<integer> Report

The `mcp_domain_<integer>` report displays the domain relationships between clocks used in the `set_multicycle_path` constraint.

For more information, review the following sections:

- [Generate the mcp_domain_<integer> Report](#)
- [View the mcp_domain_<integer> Report](#)
- [Type of Information in the mcp_domain_<integer> Report](#)
- [Interpret the mcp_domain_<integer> Report](#)
- [Related Reports](#)

Generate the mcp_domain_<integer> Report

You can generate this report by one of the following methods:

- **Run GuideWare2.0 Goal:** Not applicable
- **Run Rule:**
 - `CheckMCP` — *Prints domain information between clocks used in multicycle path commands*
 - Parameters (Optional):
 - ◆ `tc_checkmcp_report_sorted_clock`: *Default value is no. Set the value to yes, and the `mcp_domain_<integer>.rpt` report displays the instance names sorted with respect to clocks.*
 - ◆ `tc_ignore_block_path`: *Default is yes. Set the value to no so that the `set_case_analysis`, input delays, and/or output delays are also considered.*

NOTE: The `CheckMCP` rule uses the SpyGlass TXV license.

View the mcp_domain_<integer> Report

You can view the report by one of the following methods:

- **Console GUI:** Access the Reports menu.
- **Report Location:** Open the report directly from `spyglass_constraints/reports`.

Type of Information in the `mcp_domain_<integer>` Report

The `mcp_domain_<integer>` report displays the domain relationships between clocks used in the [set_multicycle_path](#) constraint. Black boxes found while performing the fan-in and/or fan-out traversal are reported.

Interpret the `mcp_domain_<integer>` Report

The format of the `mcp_domain_<integer>` report is as follows:

```
-----  
Report for schema defined at File:test.sgdc,Line:2  
Total number of MCPs found: 1  
-----
```

```
MCP at line 4 of file test.sdc -  
Source clocks: CLK1 CLK2  
Destination clocks: CLK1  
Source CLK1, Destination CLK1,  
Source CLK2, Destination CLK1, Synchronous
```

```
Starting Flop test.a_reg on domain CLK1 CLK2  
Ending Flop test.b_reg on domain CLK1  
-----
```

If the value of the [tc_checkmcp_report_sorted_clock](#) parameter is set to `yes`, the report displays the instance names sorted with respect to clocks as shown below.

```
-----  
Report for schema defined at File:test.sgdc,Line:2  
Total number of MCPs found: 1  
-----
```

```
MCP at line 7 of file test.sdc -  
Source clocks: CLK1 CLK2  
Destination clocks: CLK1  
Source CLK1, Destination CLK1,  
Source CLK2, Destination CLK1, Synchronous
```

Miscellaneous Reports

```
Starting Points
    { Flop { test.a_reg } } on domain CLK1
    { Flop { test.a_reg } } on domain CLK2
End Points
    { Flop { test.b_reg } } on domain CLK1
```

In case the proc information is provided, the following appears in the report:

MCP at line 7 of file test.sdc (The proc is defined at line <line number> of file <sdc file name>).

The format does not change for other information, such as source clocks, destination clocks, starting points, and end points.

Related Reports

[Miscellaneous Reports](#)

Appendix: SDC Constraints

SDC Constraints shows the syntax and description of the SDC Constraints.

NOTE: *SpyGlass Constraints does not read any SGDC constraint that has a corresponding SDC constraint.*

create_clock	set_clock_gating_check	set_clock_uncertainty
create_generated_clock	set_clock_groups	set_data_check
set	set_clock_latency	set_disable_timing
set_annotated_transition	set_clock_sense	set_dont_touch
set_case_analysis	set_clock_transition	set_drive
set_driving_cell	set_false_path	set_fanout_load
set_ideal_latency	set_ideal_net	set_ideal_network
set_ideal_transition	set_input_delay	set_input_transition
set_load	set_logic_dc	set_logic_one
set_logic_zero	set_max_area	set_max_capacitance
set_max_fanout	set_max_delay	set_min_delay
set_max_dynamic_power	set_max_leakage_power	set_max_time_borrow
set_max_transition	set_min_capacitance	set_min_transition

<i>set_multicycle_path</i>	<i>set_operating_conditions</i>	<i>set_output_delay</i>
<i>set_port_fanout_number</i>	<i>set_propagated_clock</i>	<i>set_resistance</i>
<i>set_timing_derate</i>	<i>set_wire_load_model</i>	<i>set_wire_load_selection_group</i>

set_sense

NOTE: Support for the *set_sense* constraint is provided for all Equivalence rules and most of the other constraints rules, except the *SDC_GenerateIncr* rule.

`create_clock`

create_clock

Syntax

The syntax to specify the `create_clock` constraint is as follows:

```
create_clock
  [-name clock_name]
  [clock_sources]
  [-period value]
  [-waveform edge_list]
  [-add]
  [-comment]
```

Arguments

-name clock_name

Specifies the name of the clock being created. If no name is specified, it takes the name of the first clock source specified with the `clock_sources` argument.

This option is required when you do not specify any clock source that creates a virtual clock or if you are using `-add` argument.

clock_sources

Specifies a list of pins or ports on which to apply the clock being created. If you do not specify `-clock_sources`, you must use `-name` argument that creates a virtual clock.

-period value

Specifies the period of the clock waveform in library time units.

-waveform edge_list

Defines the rise and fall transition of the clock edges. The list contains even number of elements of increasing times that model rise and the fall edge. If no waveform is specified, the clock is assumed to have a 50% duty cycle.

-add

This argument is used when multiple clocks are specified on the same source. While using this argument, you must also specify the `-name` argument.

-comment

This argument is used to document comments so that you can use the information about the origin of individual constraints. In addition, you can use the history of their change in the design flow.

Examples

Consider the following example:

```
create_clock -name clk1 [get_ports clk] -period 10 -waveform  
{5 7}
```

Here, we are creating a clock on port `clk`, with period of 10 library time units. The clock has a rising edge at 5 and falling edge at 7.

`create_generated_clock`

create_generated_clock

Syntax

The syntax to specify the `create_generated_clock` constraint is as follows:

```
create_generated_clock
  [-name clock_name]
  -source master_pin
  [-edges edge_list]
  [-divide_by divide_factor]
  [-multiply_by multiply_factor]
  [-combinational]
  [ -duty_cycle factor]
  [-invert]
  [-edge_shift shift_list]
  [-add]
  [-master_clock clock]
  [port_pin_list]
  [-comment]
```

Arguments

-name clock_name

Specifies the name of the clock being generated. If no name is specified, it takes the name of the first clock source specified with the `-source` argument.

clock_sources

Specifies the master clock source (clock source pin in the design) from which the clock waveform is to be derived. Note that the actual delays (latency) for the generated clock are computed using its own source pins and not the master pin.

-edges edge_list

Specifies a list of integers that represents edges of the source clock that form the edges of the generated clock. Like the clock definition, the edges are interpreted as alternating rising and falling edges. The edge list must have an odd number of elements (minimum of 3) and each edge must be not less than its previous edge.

-divide_by divide_factor

Specifies the frequency division factor.

-multiply_by multiply_factor

Specifies the frequency multiplication factor.

-combinational

Helps to model generated clocks at output of combination elements like a mux. Therefore, the source latency paths will not flow through sequential element clock pins or the source pins of other generated clocks.

-invert

Helps to model an inverted generated clock signal.

-edge_shift edge_shift_list

Specifies a list of floating point numbers that represents the amount of shift, in library time units, that the specified edges are to be moved by to create the final generated clock waveform. The number of edge shifts specified must be equal to the number of edges specified. The values can be positive or negative.

-add

This argument is used when multiple generated clocks are specified on the same source. While using this argument, you must also specify the *-name* and *-master_clock* arguments.

-master_clock clock

Specifies the master clock to be used for a generated clock, if multiple

create_generated_clock

clocks fan into the master pin. If you specify this option, you must also use the `-add` argument.

port_pin_list

Specifies a list of pins or ports on which to apply this generated clock.

-comment

This argument is used to document comments so that you can use the information about the origin of individual constraints. In addition, you can use the history of their change in the design flow.

Examples

Consider the following example:

```
create_generated_clock -name gclk1 -source clk1 -divide_by 2  
[get_pins FF1/Q]
```

Here, we are creating a generated clock whose period is twice as long as the source clock.

```
create_generated_clock -name gclk2 -source clk1 -edges { 1 3  
5 } [get_pins FF1/Q]
```

Here, we are creating a generated clock with first rising edge at 1, falling edge at 3 and next rising edge at 5.

set

Syntax

The syntax to specify the set constraint is as follows:

```
set
    variable_name variable_value
```

Arguments

variable_name variable_value

Specify the variable name and the variable value that needs to be set.

Examples

Consider the following example:

```
set sdc_version 1.3
```

Here, the SDC version is set to 1.3.

`set_annotated_transition`

set_annotated_transition

Syntax

The syntax to specify the `set_annotated_transition` constraint is as follows:

```
set_annotated_transition
  [ -rise ]
  [ -fall ]
  [ -min ]
  [ -max ]
  [ -delta_only ]
  slew_value
  pin_list
```

Arguments

-rise/fall

Declares that `slew_value` denotes the data rise or fall transition time.

-min/-max

Declares that `slew_value` denotes the minimum or maximum transition time. This is valid only if the design is in min-max mode.

-delta_only

Declares that `slew_value` denotes a delta transition time, which is added to the transition time computed by delay calculation.

slew_value

Defines the transition time of the pins or ports.

pin_list

Defines pins or ports that are annotated with the `slew_value` transition

time.

Examples

Consider the following example:

```
set_annotated_transition -rise 0.3 [get_pins U11/U22/U33/Z]  
set_annotated_transition -fall 0.5 [get_pins U11/U22/U33/Z]
```

Here, a rising transition time of 0.3 units and a falling transition time of 0.5 units on the Z input pin of the U11/U22/U33 cell is annotated.

set_case_analysis

set_case_analysis

Syntax

The syntax to specify the `set_case_analysis` constraint is as follows:

```
set_case_analysis
  value_sca
  port_pin_list
```

Arguments

value_sca

Sets a constant logic value or a transition to be assigned to the given pin or port. The valid constant values are 0, 1, zero, and one. The transition values are rising, falling, rise, and fall.

port_pin_list

Specifies the list of ports or pins on which the specified constant value or transition is to be applied.

Examples

Consider the following example:

```
set_case_analysis 1 IN1
```

Here, port IN1 is set to constant logic value of 1.

```
set_case_analysis falling {u1/u2/p1}
```

Here, pin p1 of instance u1/u2 is considered only for falling transition.

set_clock_gating_check

Syntax

The syntax to specify the `set_clock_gating_check` constraint is as follows:

```
set_clock_gating_check
  [ -setup setup_value ]
  [ -hold hold_value ]
  [ -rise | -fall ]
  [ -high | -low ]
  object_list
```

Arguments

-setup setup_value

Defines the clock gating setup time.

-hold hold_value

Defines the clock gating hold time.

-rise

Denotes that only rising delays are constrained.

-fall

Denotes that only falling delays are constrained.

-high

Denotes that only the high level of the clock is checked.

-low

Denotes that only the low level of the clock is checked

`set_clock_gating_check`

object_list

Declares clocks, ports, pins, or cells in the current design for which the clock gating check is applied.

Examples

Consider the following example:

```
set_clock_gating_check -setup 0.4 -hold 0.6 [get_clocks CK11]
```

Here, a setup time of 0.4 and a hold time of 0.6 for all gates in the clock network of the CK11 clock is specified.

set_clock_groups

Syntax

The syntax to specify the `set_clock_groups` constraint is as follows:

```
set_clock_groups
  [-physically_exclusive |
   -logically_exclusive |
   -asynchronous]
  [-allow_paths]
  [-name name]
  [-group clock_list]
  [-comment]
```

NOTE: *The `-physically_exclusive`, `-logically_exclusive`, and `-asynchronous` arguments are mutually exclusive.*

Arguments

-physically_exclusive

Used for clock groups that cannot co-exist together in a design, physically. For example, multiple clocks defined on the same pin are physically exclusive with each other.

NOTE: *While inferring domain, this argument is considered only in the TC_STA mode. You can specify the mode by setting the [tc_domain_mode](#) parameter.*

-logically_exclusive

Used for clock groups that do not have any functional paths between them, but may have coupling interactions with each other. For example, two clocks being selected by a mux.

NOTE: *While inferring domain, this argument is considered only in the TC_STA mode. You can specify the mode by setting the [tc_domain_mode](#) parameter.*

-asynchronous

Used for clock groups that have no phase relationship with each other.

set_clock_groups

NOTE: *While inferring domain, this argument is considered in all domain modes.*

-allow_paths

Enables the timing analysis between specified asynchronous clock groups.

-name name

Specifies a name for the clock grouping to be created.

-group clock_list

Specifies a list of clocks that is used to establish exclusive or asynchronous relationship with clocks of another group.

-comment

This argument is used to document comments so that you can use the information about the origin of individual constraints. In addition, you can use the history of their change in the design flow.

Examples

Consider the following example:

```
set_clock_groups -asynchronous -group {CLK1 CLK2}
```

Here, CLK1 and CLK2 are asynchronous to all other clocks in the design

```
set_clock_groups -asynchronous -group {CLK1 CLK2} -group  
{CLK3 CLK4}
```

Here, CLK1 and CLK2 are asynchronous to CLK3 and CLK4. However, no relationship is implied for CLK1 and CLK2 pair or CLK3 and CLK4 pair.

set_clock_latency

Syntax

The syntax to specify the `set_clock_latency` constraint is as follows:

```
set_clock_latency
  [ -rise ]
  [ -fall ]
  [ -min ]
  [ -max ]
  [ -source ]
  [ -late ]
  [ -early ]
  [ -clock clock_list ]
  delay
  object_list
```

Arguments

-rise/-fall

Specifies clock rise/fall latency.

-min/-max

Specifies clock latency for the minimum/maximum operating condition.

-source

Specifies clock source latency.

-late/-early

Specifies clock late/early source latency.

-clock clock_list

Specifies the associated clocks that contribute to the network latency on

`set_clock_latency`

ports and pins specified by the `object_list`.

delay

Floating point number in library time units representing the latency

object_list

Specifies clocks, ports, or pins.

Examples

Consider the following example:

```
set_clock_latency 1.1 [get_clocks CLK1]
```

Here, the rise and fall latency of CLK1 is set to 1.1.

```
set_clock_latency -rise 1.1 [get_clocks CLK1]
```

```
set_clock_latency -fall 1.2 [get_clocks CLK1]
```

Here, the rise latency of CLK1 is set to 1.1 and fall is set to 1.2.

set_clock_sense

Syntax

The syntax to specify the `set_clock_sense` constraint is as follows:

```
set_clock_sense
  [ -positive ]
  [ -negative ]
  [ -stop_propagation ]
  [ -logical_stop_propagation ]
  [ -pulse pulse_type ]
  [ -clocks clock_list ]
  object_list
```

Arguments

-positive/-negative

Denotes positive or negative unateness applied to all pins in the `object_list` as per the clock source.

If you use the `-positive` argument, you cannot use the `-negative` or `-pulse` arguments. Similarly, if you use the `-negative` option, you cannot use the `-positive` or `-pulse` options.

-stop_propagation

Denotes that clock-as-clock and clock-as-data propagation is stopped for the clocks specified in the `clock_list` from the pins or cell timing arcs declared in the `object_list`.

-logical_stop_propagation

This argument is used only by the `set_clock_sense` constraint. It is not used by the `set_sense` constraint.

Denotes that only clocks-as-clock propagation is stopped for specified clocks-as-clock in the `clock_list` from the pins or cell timing arcs

`set_clock_sense`

declared in the `object_list`.

-pulse pulse_type

Defines the pulse clock type that is applied to all pins in the `object_list` with reference to the clock source. The possible values for the `pulse_type` argument are:

- `rise_triggered_high_pulse`
- `fall_triggered_high_pulse`
- `rise_triggered_low_pulse`
- `fall_triggered_low_pulse`

If you use the `-pulse` argument, you cannot use the `-positive` or `-negative` arguments.

-clock clock_list

Defines clock objects that are applied with the given unateness that is located on all pin objects in the `object_list`. If you do not specify the `-clocks` option, all clocks passing through the given pin objects are considered.

object_list

List of pins, ports, or cell timing arcs with specified unateness to propagate. The timing arcs object can be used with the `-stop_propagation` option only.

Examples

Consider the following example:

```
set_clock_sense -positive -clocks [get_clocks CLK11] XOR/Z
```

Here, a positive unateness for the XOR/Z pin with respect to CLK11 is specified.

Stopping Clock Propagation

You can use the `-stop_propagation` argument of the `set_clock_sense` command to stop the propagation of a clock.

The following is the syntax of the `set_clock_sense` command with the `-stop_propagation` argument:

```
set_clock_sense -stop_propagation [-clocks
list_of_clocks][list_of_pins]
```

This means that the propagation of clocks mentioned in the clock list (`list_of_clocks`) will not proceed beyond the mentioned pins (`list_of_pins`).

NOTE: The `set_clock_sense` command also takes other arguments for propagation. Here, we are discussing the `-stop_propagation` argument only.

Example

Suppose a design contains three clocks: CLK1, CLK2, and CLK3. Let the clock CLK1 go to pins p1, p2, p3, and p4. In addition, let the clock CLK2 go to pins p3 and p4, and the clock CLK3 go to pin p3.

The effect of the `set_clock_sense` command on propagation of clocks is described in the table below:

Scenario	Expected Behavior
<code>set_clock_sense -stop_propagation -clock {CLK1} {p1 p2}</code>	The clock CLK1 will not propagate beyond pins p1 and p2
<code>set_clock_sense -stop_propagation {p3}</code>	No clock will propagate beyond pin p3
<code>set_clock_sense -stop_propagation -clock {CLK1 CLK2} p4</code>	The clocks CLK1 and CLK2 will not propagate beyond pin p4

The `set_clock_sense` command with the `-stop_propagation` argument affects the following rules:

<code>Clk_Gen01a</code>	<code>Clk_Gen01b</code>	<code>Clk_Gen10</code>	<code>Check_Timing02</code>
<code>Clk_Gen02</code>	<code>Clk_Gen05</code>	<code>Clk_Lat03</code>	<code>Clk_Uncert03</code>
<code>DomainAnalysis</code>	<code>DomainInfo</code>	<code>False_Path07</code>	<code>False_Path08</code>
<code>Inp_Del03a</code>	<code>Inp_Del03b</code>	<code>Op_Del03a</code>	<code>Op_Del04</code>

set_clock_sense

<i>Clk_Gen06</i>	<i>Disable_Timing 02</i>	<i>False_Path01</i>	<i>MCP01</i>
<i>TE_Consis02</i>	<i>Constraints Management Rules</i>	<i>I/O Rules</i>	<i>Inp_Del01a</i>
<i>Inp_Del01b</i>	<i>Op_Del01a</i>	<i>Op_Del01b</i>	<i>Clk_Gen24</i>
<i>Block05</i>	<i>Clk_Uncert07</i>	<i>DomainError</i>	<i>False_Path03</i>
<i>False_Path04a</i>	<i>MCP03</i>	<i>MCP04a</i>	<i>Show_Clock_Propagati on</i>

set_sense

Syntax

The syntax to specify the `set_sense` constraint is as follows:

```
set_sense
  [ -type clock | data ]
  [ -positive ]
  [ -negative ]
  [ -stop_propagation ]
  [ -pulse pulse_type ]
  [ -clocks clock_list ]
  object_list
```

NOTE: Support for the `set_sense` constraint is provided for all Equivalence rules and most of the other constraints rules, except the [SDC_GenerateIncr](#) rule.

Arguments

-type clock | data

Specifies whether the sense is applied to clock or data networks. If you do not specify this option, the default is clock. If you specify data, you must also use the `-stop_propagation` and `-clocks` options.

-positive/-negative

Denotes positive or negative unateness applied to all pins in the `object_list` as per the clock source.

If you use the `-positive` argument, you cannot use the `-negative` or `-pulse` arguments. Similarly, if you use the `-negative` option, you cannot use the `-positive` or `-pulse` options.

-stop_propagation

Denotes that clock-as-clock and clock-as-data propagation is stopped for the clocks specified in the `clock_list` from the pins or cell timing arcs declared in the `object_list`.

`set_sense`

-pulse pulse_type

Defines the pulse clock type that is applied to all pins in the `object_list` with reference to the clock source. The possible values for the `pulse_type` argument are:

- `rise_triggered_high_pulse`
- `fall_triggered_high_pulse`
- `rise_triggered_low_pulse`
- `fall_triggered_low_pulse`

If you use the `-pulse` argument, you cannot use the `-positive` or `-negative` arguments.

-clock clock_list

Defines clock objects that are applied with the given unateness that is located on all pin objects in the `object_list`. If you do not specify the `-clocks` option, all clocks passing through the given pin objects are considered.

object_list

List of pins, ports, or cell timing arcs with specified unateness to propagate. The timing arcs object can be used with the `-stop_propagation` option only.

Examples

The following example specifies a positive unateness for the XOR/Z pin with respect to the CLK1 clock.

```
set_sense -positive -clocks [get_clocks CLK1] XOR/Z
```

The following example specifies negative unateness for the MUX/Z pin for all clocks.

```
set_sense -negative MUX/Z
```

set_clock_transition

Syntax

The syntax to specify the `set_clock_transition` constraint is as follows:

```
set_clock_transition
  [ -rise ]
  [ -fall ]
  [ -min ]
  [ -max ]
  transition
  clock_list
```

Arguments

-rise/-fall

Specifies the rising/falling clock transition time.

-min/-max

Specifies clock transition time for the minimum/maximum operating condition.

transition

Floating point number in library time units representing the transition.

clock_list

Specifies the list of clocks on which transition is being set.

Examples

Consider the following example:

```
set_clock_transition 1.1 [get_clocks CLK1]
```

Here, the rise and fall transition of CLK1 is set to 1.1.

`set_clock_transition`

```
set_clock_transition -rise 1.1 [get_clocks CLK1]
```

```
set_clock_transition -fall 1.2 [get_clocks CLK1]
```

Here, the rise transition of CLK1 is set to 1.1 and fall transition is set to 1.2.

set_clock_uncertainty

Syntax

The syntax to specify the `set_clock_uncertainty` constraint is as follows:

```
set_clock_uncertainty
  [-from from_clock]
  [-to to_clock]
  [-rise]
  [-rise_from from_clock]
  [-rise_to to_clock]
  [-fall]
  [-fall_from from_clock]
  [-fall_to to_clock]
  [-setup]
  [-hold]
  uncertainty
  [object_list]
```

Arguments

-from/-rise_from/-fall_from from_clock
-to/-rise_to/-fall_to to_clock

These arguments are used to specify interclock uncertainty for source and destination clocks. You must specify either the pair of `-from/-rise_from/-fall_from` and `-to/-rise_to/-fall_to`, or `object_list`. The `rise_from/fall_from` applies only to the rising/falling edge of the source clock. Similarly, `rise_to/fall_to` applies only to the rising/falling edge of the destination clock.

-rise/-fall

Indicates that uncertainty is applied to only the rising/falling edge of the destination clock only. By default, the uncertainty applies to both rising and

`set_clock_uncertainty`

falling edges. It has the same effect as `-rise_to` and `-fall_to`

-setup/-hold

Specifies that uncertainty is applied only to setup/hold checks. By default, uncertainty applies to both setup and hold checks.

uncertainty

Floating point number in library time units representing the uncertainty value

object_list

Specifies clocks, ports, or pins on which uncertainty is applied. This cannot be used with `-from` or `-to` options.

Examples

Consider the following example:

```
set_clock_uncertainty 1.1 -from [get_clocks CLK1] -to
[get_clocks CLK2]
```

Here, the interclock uncertainty between CLK1 and CLK2 is set to 1.1

```
set_clock_uncertainty 1.1 -fall_from [get_clocks CLK1] -
rise_to [get_clocks CLK2]
```

Here, interclock uncertainty between CLK1 and CLK2 is set to 1.1 for the specific edges (i.e. fall for CLK1 and rise for CLK2).

```
set_clock_uncertainty -setup 0.6 [get_clocks CLK1]
```

```
set_clock_uncertainty -hold 0.4 [get_clocks CLK1]
```

Here, setup and hold uncertainty for CLK1 is set to 0.6 and 0.4, respectively.

set_data_check

Syntax

The syntax to specify the `set_data_check` constraint is as follows:

```
set_data_check
  { -from from_object | -rise_from from_object |
    -fall_from from_object }
  { -to to_object | -rise_to to_object |
    -fall_to to_object }
  [ -setup | -hold ]
  [-clock clock_object]
  [check_value]
```

Arguments

-from from_object/-rise_from from_object/-fall_from from_object

Use the `from from_object` argument to define a pin or port in the current design as the related pin of the data-to-data check to be set. The check is for both rising and falling delays.

Use the `rise_from` and `fall_from` arguments only for rising or falling delays at the related pin.

-to to_object/-rise_to to_object/-fall_to to_object

Use the `to to_object` argument to define a pin or port in the current design as the constrained pin of the data-to-data check to be set. The check is for both rising and falling delays.

Use the `rise_to` and `fall_to` arguments apply only to rising or falling delays at the constrained path.

`set_data_check`

-setup/-hold

Declares that the data check value is for setup or hold analysis only. If you neither specify `setup` nor `hold`, both setup and hold is applied to the data check value.

-clock clock_object

Defines the name of a single clock that is used for the data check.

check_value

Defines the value of the setup and/or hold time for the check.

Examples

Consider the following example:

```
set_data_check -rise_from and1/Z -to and1/Y 0.6
```

Here, a data-to-data check from `and1/Z` to `and1/Y` as per the rising edge of signal at `and1/Z`, using a setup and hold time of 0.6 is created.

set_disable_timing

Syntax

The syntax to specify the `set_disable_timing` constraint is as follows:

```
set_disable_timing
  [-from from_pin_name]
  [-to to_pin_name]
  [object_list]
```

Arguments

-from from_pin_name

Specifies the source pin from which the timing arcs are to be disabled for the cell specified in the `object_list`.

-to to_pin_name

Specifies the destination pin to which the timing arcs are to be disabled for the cell specified in the `object_list`.

object_list

Specifies the list of cells to be disabled.

Examples

Consider the following example:

```
set_disable_timing -from A -to Z [get_cells AND]
```

Here, the timing arch from A to Z for AND cell is disabled.

set_dont_touch

set_dont_touch

Syntax

The syntax to specify the `set_dont_touch` constraint is as follows:

```
set_dont_touch  
  [ true | false ]  
  object_list
```

Arguments

true | false

Specifies the value of the `dont_touch` attribute. The default is `true`.

object_list

Specifies a list of cells, nets, references, designs, and library cells on which the `dont_touch` attribute is to be set.

Examples

Consider the following example:

```
set_dont_touch find (net, N1) false
```

Here, `set_dont_touch` is set to `false` for the `N1` net. This indicates that the net be modified during compile.

set_drive

Syntax

The syntax to specify the `set_drive` constraint is as follows:

```
set_drive
  [ -rise ]
  [ -fall ]
  [ -min ]
  [ -max ]
  resistance_value port_list
```

Arguments

-rise/-fall

Denotes that `resistance_value` is used to drive the ports only for rising or falling.

-min/-max

Denotes that the `resistance_value` is the maximum or minimum resistance. This is only applicable if the design is in min-max mode.

resistance_value port_list

Declares a non-negative port drive resistance value, which is greater than or equal to 0, for the ports in the `port_list`. The units must match those in the technology library.

Examples

Consider the following example:

```
set_drive 7 [all_inputs]
```

Here, the drive resistance is set to 7 on all input ports in the current design.

set_driving_cell

set_driving_cell

Syntax

The syntax to specify the `set_driving_cell` constraint is as follows:

```
set_driving_cell
  [-lib_cell lib_cell_name]
  [ -rise ]
  [ -fall ]
  [ -min ]
  [ -max ]
  [ -library lib_name ]
  [ -pin pin_name ]
  [ -from_pin from_pin_name ]
  [ -multiply_by factor ]
  [ -dont_scale ]
  [ -no_design_rule ]
  [ -input_transition_rise rtrans ]
  [ -input_transition_fall ftrans ]
  [ -clock clock_name ]
  [ -clock_fall ]
  port_list
```

Arguments

-lib_cell lib_cell_name

Defines the library cell name that is used to drive the ports.

-rise/-fall

Denotes the `driving_cell` information for the rising or falling port transition.

-min/-max

Denotes the `driving_cell` information for minimum or maximum

analysis. These arguments are applicable only in min-max mode.

-library lib_name

Declares the library name that contains the `library_cell_name` value.

-pin pin_name

Declares the output pin that is used to drive.

-from_pin from_pin_name

Declares an input pin on the cell to ensure that the drive of the timing arc from this pin to the specified pin is used.

-multiply_by_factor

Denotes that the calculated transition time is multiplied by the specified multiplier.

-dont_scale

Denotes that, according to the current operating conditions, the timing analyzer is not to scale the drive capability of the ports. Operating condition scaling is prevented.

-no_design_rule

Declares that the design rules are not be transferred from the driving cell to the port.

-input_transition_rise rtrans

Declares the input rising transition time that is associated with the `from_pin` option.

-input_transition_fall ftrans

Declares the input falling transition time that is associated with the `from_pin_name` value.

`set_driving_cell`

-clock clock_name

Declares the driving cell relative to the specified clock.

-clock_fall

Declares that the driving cell is relative to the falling edge of the clock. By default, it is relative to the rising edge.

port_list

Declares input or inout port names in the current design on which the driving cell information is set.

Examples

Consider the following example:

```
set_driving_cell -lib_cell AND22 {IN11}
```

Here, the AND22 library cell is associated with the IN1 port.

set_false_path

Syntax

The syntax to specify the `set_false_path` constraint is as follows:

```
set_false_path
  [-setup]
  [-hold]
  [-rise]
  [-fall]
  [-reset_path]
  [-from from_list]
  [-to to_list]
  [-through through_list]
  [-rise_from from_list]
  [-fall_from from_list]
  [-rise_to to_list]
  [-fall_to to_list]
  [-rise_through through_list]
  [-fall_through through_list]
  [-comment]
```

Arguments

-setup/-hold

Indicates that setup (maximum)/hold (minimum) paths are to be marked as false. This will disable setup/hold checking for these path during timing analysis.

-rise/-fall

Indicates that rising/falling delays for end points to be marked as false. These are equivalent to `rise_to` and `fall_to` arguments.

`set_false_path`

`-reset_path`

Removes the existing false path setting.

`-from/-rise_from/-fall_from from_list`

Specifies a list of objects that act as the start point for the false path. A valid start point is a clock, a primary input or inout port, a sequential cell, a clock pin of a sequential cell, a data pin of a level-sensitive latch, or a pin that has input delay specified. If a clock is specified, all registers and primary inputs related to the clock are considered as start points. The `rise_from` and `fall_from` are used only for rising/falling path of the start point.

`-to/-rise_to/-fall_to to_list`

Specifies a list of objects that act as endpoint for the false path. A valid endpoint is a clock, a primary output or inout port, a sequential cell, a data pin of a sequential cell, or a pin that has output delay specified. The `rise_to` and `fall_to` are used only for rising/falling path of the endpoint.

`-through/-rise_through/-fall_through through_list`

Specifies a list of pins, ports, cells, or nets that are to be disabled. The `rise_through` and `fall_through` are used only for rising/falling transition on the through points.

For example, consider the following specification:

```
set_false_path
  -through [get_pins A1 A2] -through [get_pins B1 B2]
```

The pins within one `-through` are logically ORed and then logically ANDed with the next `-through` option. Therefore, this specification is describing a path that goes through one of the following options:

- A1 and B1
- A1 and B2
- A2 and B1
- A2 and B2

-comment

This argument is used to document comments so that you can use the information about the origin of individual constraints. In addition, you can use the history of their change in the design flow.

Examples

Consider the following example:

```
set_false_path -from clk
```

Here, all registers and primary inputs related to `clk` clock are considered as start points.

```
set_false_path -from FF1 -to FF2
```

Here, all paths from `FF1` to `FF2` are disabled.

set_fanout_load

set_fanout_load

Syntax

The syntax to specify the `set_fanout_load` constraint is as follows:

```
set_fanout_load
  value
  port_list
```

Arguments

value

Specifies load value in units consistent with the technology library.

port_list

Specifies the list of output ports whose fan-out load is constrained.

Examples

Consider the following example:

```
set_fanout_load 2.5 [get_port OUT1]
```

Here, the fan-out load on port OUT1 is set to 2.5.

```
set_fanout_load 1.2 [get_ports "OUT*"]
```

Here, the fan-out load on ports matching OUT* are set to 1.2

set_ideal_latency

Syntax

The syntax to specify the `set_ideal_latency` constraint is as follows:

```
set_ideal_latency
  [ -rise ]
  [ -fall ]
  [ -min ]
  [ -max ]
  value
  object_list
```

Arguments

-rise/-fall

Declares that the `value` denotes the rise or fall latency time.

-min/-max

Declares that the `value` denotes the minimum or maximum latency time.

value

Defines an ideal latency value on leaf cell pins or top-level ports in an ideal network.

object_list

Declares leaf cell pins and top-level ports on which ideal latency is set.

Examples

Consider the following example:

```
set_ideal_latency 1.4 -rise {X Y Z}
set_ideal_latency 1.1 -fall {X Y Z}
```

set_ideal_latency

Here, a rise latency of 1 . 4 and a fall latency of 1 . 1 is specified for the X, Y, and Z ports.

set_ideal_net

Syntax

The syntax to specify the `set_ideal_net` constraint is as follows:

```
set_ideal_net  
  net_list
```

Arguments

net_list

Declares nets on which `ideal_net` attribute is to be set. Make sure the nets declared are visible from the current design.

Examples

Consider the following example:

```
current_design CLK_GEN123  
set_ideal_net find(net)
```

Here, the `ideal_net` attribute is set on nets in the CLK_GEN123 design.

set_ideal_network

set_ideal_network

Syntax

The syntax to specify the `set_ideal_network` constraint is as follows:

```
set_ideal_network
  [ -no_propagate ]
  object_list
```

Arguments

-no_propagate

Denotes that the ideal network is not propagated through leaf cells.

object_list

Defines objects that are sources of an ideal network.

Examples

Consider the following example:

```
current_design {ZZTOP}
set_ideal_network {IN11 IN22}
```

Here, an ideal network on ports is created in the ZZTOP design.

In the following example, an ideal network is created on the netZ multi-driven net

```
set_ideal_network -no_propagate {netZ}
```

set_ideal_transition

Syntax

The syntax to specify the `set_ideal_transition` constraint is as follows:

```
set_ideal_transition
  [ -rise ]
  [ -fall ]
  [ -min ]
  [ -max ]
  value
  object_list
```

Arguments

-rise/-fall

Declares that the `value` option denotes the rise or fall transition time.

-min/-max

Declares that the `value` option denotes the minimum or maximum transition time.

value

Defines an ideal transition value on leaf cell pins or top-level ports in an ideal network.

object_list

Declares leaf cell pins and top-level ports on which ideal transition is set.

Examples

Consider the following example:

```
set_ideal_transition 1.4 -rise {Z Y X}
```

set_ideal_transition

```
set_ideal_transition 1.1 -fall {Z Y X}
```

Here, a rise transition of 1.4 and a fall transition of 1.1 is specified for the Z, Y, and X ports.

set_input_delay

Syntax

The syntax to specify the `set_input_delay` constraint is as follows:

```
set_input_delay
  [ -clock clock_name ]
  [ -clock_fall ]
  [ -level_sensitive ]
  [ -rise ]
  [ -fall ]
  [ -min ]
  [ -max ]
  [ -add_delay ]
  delay_value
  port_pin_list
```

Arguments

-clock clock_name

Specifies the related clock of the port or pin on which the input delay is to be applied.

-clock_fall

Specifies the delay relative to the falling edge of the clock. While using this argument, you must also specify the `-clock` argument.

-level_sensitive

Specifies that the source of the delay is a level-sensitive latch.

-rise/-fall

Specifies the `delay_value` on account of a rising/falling transition on the specified port. When you do not specify these arguments, rise and fall delay are assumed to be same.

`set_input_delay`

-min/-max

Specifies `delay_value` on the shortest/longest path to the port/pin.

-add_delay

Used when delay on a port is required to be specified with respect to multiple clocks. This happens when multiple paths from different clocks lead to an input port.

delay

Floating point number in library time units representing the delay.

port_pin_list

Specifies a list of input port or internal pin names in the current design to which `delay_value` is assigned.

Examples

Consider the following example:

```
set_input_delay 1.0 -clock clk1 {IN1}
```

```
set_input_delay 1.7 -clock clk2-add_delay { IN1 }
```

Here, the input delay on IN1 is set with respect to two clocks `clk1` and `clk2`.

```
set_input_delay 1.0 { IN1 IN2 }
```

Here, delay is specified for a combinational design.

set_input_transition

Syntax

The syntax to specify the `set_input_transition` constraint is as follows:

```
set_input_transition
  [ -rise ]
  [ -fall ]
  [ -min ]
  [ -max ]
  transition
  port_list
```

Arguments

-rise/-fall

Specifies the rising/falling input transition time.

-min/-max

Specifies input transition time for the minimum/maximum operating condition.

transition

Floating point number in library time units representing the transition.

port_list

Specifies the list of ports on which transition is being set.

Examples

Consider the following example:

```
set_input_transistion 1.1 [get_ports "IN*"]
```

Here, the rise and fall transition on ports matching `IN1*` is set to `1.1`.

`set_input_transition`

```
set_input_transition -rise 1.1 [get_ports IN1]
```

```
set_input_transition -fall 1.2 [get_ports IN1]
```

Here, the rise transition of IN1 is set to 1.1 and fall transition is set to 1.2.

set_load

Syntax

The syntax to specify the `set_load` constraint is as follows:

```
set_load
  [ -rise ]
  [ -fall ]
  [ -min ]
  [ -max ]
  [ -subtract_pin_load ]
  [ -pin_load ]
  [ -wire_load ]
  capacitance
  objects
```

Arguments

-rise/-fall

Declares that the capacitance is rise or fall. These arguments are only applicable to ports.

-min/-max

Declares the minimum or maximum capacitance. These arguments are only valid to designs in the min-max mode.

-subtract_pin_load

Denotes that the current pin capacitances of the specified net are subtracted from capacitance before the net capacitance value is set.

-pin_load

Denotes that the specified capacitance is a pin capacitance. This argument is only applicable to ports.

`set_load`

-wire_load

Denotes that the specified capacitance is a wire capacitance. This argument is only applicable to ports.

capacitance

Defines the capacitance value set on the ports and nets in objects. The units are those used in the technology library.

objects

Declares ports and nets in the current design on which the capacitance is set.

Examples

Consider the following example:

```
set_load 4 XYZ
```

Here, a capacitance of 4 units on the XYZ port.

In the following example, a wire capacitance of 7 units is set on the XYZ port

```
set_load -wire_load 7 XYZ
```

set_logic_dc

Syntax

The syntax to specify the `set_logic_dc` constraint is as follows:

```
set_logic_dc  
port_list
```

Arguments

port_list

Defines input port names in the current design that are driven by logic dont care.

Examples

Consider the following example:

```
set_logic_dc XYZ
```

Here, the XYZ input port is driven by dont care.

set_logic_one

set_logic_one

Syntax

The syntax to specify the `set_logic_one` constraint is as follows:

```
set_logic_one  
  port_list
```

Arguments

port_list

Defines input port names in the current design that are to be driven by logic 1.

Examples

Consider the following example:

```
set_logic_one XYZ
```

Here, the XYZ input port is driven by logic 1.

set_logic_zero

Syntax

The syntax to specify the `set_logic_zero` constraint is as follows:

```
set_logic_zero  
port_list
```

Arguments

port_list

Defines input port names in the current design that are to be driven by logic 0.

Examples

Consider the following example:

```
set_logic_zero XYZ
```

Here, the XYZ input port is driven by logic 0.

set_max_area

set_max_area

Syntax

The syntax to specify the `set_max_area` constraint is as follows:

```
set_max_area  
    area_value
```

Arguments

area_value

Defines the `max_area` attribute value. This value must be greater than equal to zero and should use the same the units as in the technology library used during optimization.

Examples

Consider the following example:

```
set_max_area 150.0
```

Here, the target area for the current design is set to 150.

set_max_capacitance

Syntax

The syntax to specify the `set_max_capacitance` constraint is as follows:

```
set_max_capacitance
  [ -rise ]
  [ -fall ]
  [ -clock_path ]
  [ -data_path ]
  objects_list
```

Arguments

-rise/-fall

Declares the rising or falling capacitance for all selected pins.

-clock_path

Declares all pins that are in the network of the clocks in the `object_list`.

-data_path

Declares all pins that are in the data paths launched by the specific clocks in the `object_list`.

objects_list

Declares pins, ports, clocks or designs that need to have the maximum capacitance set.

Examples

Consider the following example:

```
set_max_capacitance 4.0 [get_ports "OUT*"]
```

`set_max_capacitance`

Here, a maximum capacitance limit of 4 . 0 units is set on the OUT* ports.

In the following example, a maximum capacitance limit of 1 . 5 units is set on all pins of the clock network of CLK11.

```
set_max_capacitance 1.5 [get_clocks CLK11] -clock_path
```

set_max_fanout

Syntax

The syntax to specify the `set_max_fanout` constraint is as follows:

```
set_max_fanout
  fanout_value
  object_list
```

Arguments

fanout_value

Defines the maximum fan-out load in library fan-out units.

objects_list

Defines input ports or designs where the maximum fan-out needs to be set.

Examples

Consider the following example:

```
set_max_fanout 1.5 [ge_ports "IN*"]
```

Here, a maximum fan-out limit of 1.5 units is set on ports `IN*`.

In the following example, the default maximum fan-out limit of 8.0 units is set on the `current_design`.

```
set_max_fanout 8.0 [current_design]
```

set_max_delay/set_min_delay

set_max_delay/set_min_delay

set_max_delay

set_min_delay

Syntax

The syntax to specify the set_max_delay/set_min_delay constraint is as follows:

```
set_max_delay / set_min_delay
  [-rise]
  [-fall]
  [-reset_path]
  [-from from_list]
  [-to to_list]
  [-through through_list]
  [-rise_from from_list]
  [-fall_from from_list]
  [-rise_to to_list]
  [-fall_to to_list]
  [-rise_through through_list]
  [-fall_through through_list]
  delay_value
  [-comment]
```

Arguments

-rise/-fall

Indicates that rising/falling path delays to be constrained.

-reset_path

Removes the existing max/min delay constraints.

-from/-rise_from/-fall_from from_list

Specifies a list of objects that act as the start point. A valid start point is a clock, a primary input or inout port, a sequential cell, a clock pin of a sequential cell, a data pin of a level-sensitive latch, or a pin that has input delay specified. If a clock is specified, all registers and primary inputs related to the clock are considered as start points. The `rise_from` and `fall_from` are used only for rising/falling path of the start point.

-to/-rise_to/-fall_to to_list

Specifies a list of objects that act as endpoint. A valid endpoint is a clock, a primary output or inout port, a sequential cell, a data pin of a sequential cell, or a pin that has output delay specified. The `rise_to` and `fall_to` are used only for rising/falling path of the end point.

-through/-rise_through/-fall_through through_list

Specifies a list of pins, ports, cells, or nets through which paths must pass for max or min. The `rise_through` and `fall_through` are used only for rising/falling transition on the through points.

delay_value

Floating point number that specifies the max/min delay on the timing path specified using `-from/-through/-to` options.

-comment

This argument is used to document comments so that you can use the information about the origin of individual constraints. In addition, you can use the history of their change in the design flow.

Examples

Consider the following example:

```
set_min_delay 2.0 -from {FF1} -to {FF2}
```

```
set_max_delay 6.0 -from {FF1} -to {FF2}
```

Here, all paths from FF1 to FF2 must have a delay more than 2.0 time units (Min) and less than 6.0 time units (Max).

set_max_dynamic_power

set_max_dynamic_power

Syntax

The syntax to specify the `set_max_dynamic_power` constraint is as follows:

```
set_max_dynamic_power  
    max_dynamic_power  
    [GW | MW | KW | W | mW | uW | nW | pW | fW | aW]
```

Arguments

max_dynamic_power

Defines the target dynamic power of the current design.

[*GW* | *MW* | *KW* | *W* | *mW* | *uW* | *nW* | *pW* | *fW* | *aW*]

Declares the power dimension.

Examples

Consider the following example:

```
set_max_dynamic_power 0.1
```

Here, the target power is set to 0.1 for the `current_design`.

set_max_leakage_power

Syntax

The syntax to specify the `set_max_leakage_power` constraint is as follows:

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

Arguments

leakage_power

Defines the target leakage power of the current design.

[*GW* | *MW* | *KW* | *W* | *mW* | *uW* | *nW* | *pW* | *fW* | *aW*]

Declares the power dimension.

Examples

Consider the following example:

```
set_max_leakage_power 0.1
```

Here, the target power is set to 0.1 for the `current_design`.

`set_max_time_borrow`

set_max_time_borrow

Syntax

The syntax to specify the `set_max_time_borrow` constraint is as follows:

```
set_max_time_borrow
  value
  object_list
```

Arguments

value

Defines the `max_time_borrow` attribute value.

objects_list

Defines clocks, latch cells, data pins, or clock (enable) pins in the `current_design` for which time borrowing is to be limited to `value`.

Examples

Consider the following example:

```
set_max_time_borrow 8.0 { latch11aZ latch22fG }
```

Here, time borrowing on `latch11aZ` and `latch22fG` is restricted to 8.0 units.

In the following example, no time borrowing takes place on `latch1`.

```
set_max_time_borrow 0.0 { latch1 }
```

set_max_transition

Syntax

The syntax to specify the `set_max_transition` constraint is as follows:

```
set_max_transition
  [ -rise ]
  [ -fall ]
  [ -clock_path ]
  [ -data_path ]
  transition_value
  object_list
```

Arguments

-rise/-fall

Defines the rising or falling transition for all selected pins.

-clock_path

Declares all pins in the network of the clocks in the `object_list`.

-data_path

Declares all pins in the data paths launched by the clocks in the `object_list`.

transition_value

Defines the transition limit, which is the maximum transition time in library time units.

object_list

Declares pins, ports, clocks or designs where the maximum transition is set.

set_max_transition

Examples

Consider the following example:

```
set_max_transition 3.0 [get_ports "OUT*"]
```

Here, a maximum transition limit of 3.0 units is set on the OUT* ports.

In the following example, a maximum transition limit of 5.0 units is set on all pins of the clock network of CLK1.

```
set_max_transition 5.0 [get_clocks CLK1] -clock_path
```

set_min_capacitance

Syntax

The syntax to specify the `set_min_capacitance` constraint is as follows:

```
set_min_capacitance
  capacitance_value
  object_list
```

Arguments

capacitance_value

Defines the minimum total capacitance.

objects_list

Declares input or inout ports or designs where the minimum capacitance is set.

Examples

Consider the following example:

```
set_min_capacitance 2.1 [get_ports "IN*"]
```

Here, a minimum capacitance limit of 2.1 units is set on the IN* ports.

In the following example, the default minimum capacitance limit of 0.4 units is set on the `current_design`.

```
set_min_capacitance 0.4 [current_design]
```

set_min_transition

set_min_transition

Syntax

The syntax to specify the `set_min_transition` constraint is as follows:

```
set_min_transition
  [ -rise ]
  [ -fall ]
  [ -clock_path ]
  [ -data_path ]
  transition_value
  object_list
```

Arguments

-rise/-fall

Declares the rising or falling transition for the pins selected.

-clock_path

Declares all pins that are in the network of the clocks in the `object_list`.

-data_path

Declares all pins that are in the data paths launched by the clocks in the `object_list`.

transition_value

Defines the minimum transition limit in library time units.

object_list

Defines clocks, pins, ports, or designs on which to set the minimum transition.

Examples

Consider the following example:

```
set_min_transition 3.0 [get_ports "OUT*"]
```

Here, a minimum transition limit of 3.0 units is set on the OUT* ports.

In the following example, a minimum transition limit of 1.0 unit is set on all pins of the clock network of CLK1.

```
set_max_transition 1.0 [get_clocks CLK1] -clock_path
```

`set_multicycle_path`

set_multicycle_path

Syntax

The syntax to specify the `set_multicycle_path` constraint is as follows:

```
set_multicycle_path
  [-setup]
  [-hold]
  [-rise]
  [-fall]
  [-start]
  [-end]
  [-reset_path]
  [-from from_list]
  [-to to_list]
  [-through through_list]
  [-rise_from from_list]
  [-fall_from from_list]
  [-rise_to to_list]
  [-fall_to to_list]
  [-rise_through through_list]
  [-fall_through through_list]
  path_multiplier
  [-comment]
```

Arguments

-setup/-hold

Indicates that setup (maximum)/hold (minimum) paths to be used as a multi-cycle paths specified using the `path_multiplier`.

-rise/-fall

Indicates that rising/falling delays for end points to be considered for multi-cycle path analysis. These are equivalent to `rise_to` and `fall_to`

arguments.

-reset_path

Removes the existing multi-cycle path setting.

-from/-rise_from/-fall_from from_list

Specifies a list of objects that act as the start point for the multi-cycle path. A valid start point is a clock, a primary input or inout port, a sequential cell, a clock pin of a sequential cell, a data pin of a level-sensitive latch, or a pin that has input delay specified. If a clock is specified, all registers and primary inputs related to the clock are considered as start points. The *rise_from* and *fall_from* are used only for rising/falling path of the start point.

-to/-rise_to/-fall_to to_list

Specifies a list of objects that act as endpoint for the multi-cycle path. A valid endpoint is a clock, a primary output or inout port, a sequential cell, a data pin of a sequential cell, or a pin that has output delay specified. The *rise_to* and *fall_to* are used only for rising/falling path of the end point.

-through/-rise_through/-fall_through through_list

Specifies a list of pins, ports, cells, or nets that are to be disabled. The *rise_through* and *fall_through* are used only for rising/falling transition on the through points.

For example, consider the following specification:

```
set_multicycle_path
  -through [get_pins A1 A2] -through [get_pins B1 B2]
```

The pins within one *-through* are logically ORed and then logically ANDed with the next *-through* option. Therefore, this specification is describing a path that goes through one of the following options:

- A1 and B1
- A1 and B2
- A2 and B1

`set_multicycle_path`

■ A2 and B2

path_multiplier

Integer that specifies the number of cycles path must have for setup or hold, relative to the start point or end point clock, before data is required at the endpoint. For example, specifying a `path_multiplier` of 2 for setup implies a 2-cycle data path.

-comment

This argument is used to document comments so that you can use the information about the origin of individual constraints. In addition, you can use the history of their change in the design flow.

Examples

Consider the following example:

```
set_multicycle_path 2 -from FF1 -to FF2
```

Here, paths from FF1 to FF2 is a 2 cycle path.

set_operating_conditions

Syntax

The syntax to specify the `set_operating_conditions` constraint is as follows:

```
set_operating_conditions
  [-analysis_type single | bc_wc | on_chip_variation]
  [-library lib]
  [condition]
  [-min min_condition]
  [-max max_condition]
  [-min_library min_lib]
  [-max_library max_lib]
  [-object_list objects]
```

Arguments

-analysis_type single | bc_wc | on_chip_variation

These mutually-exclusive arguments declare the usage of the operating conditions.

- `single`: Declares that only one operating condition is used.
- `bc_wc`: Declares that the min and max operating conditions are two extreme operating conditions. Setup violations are checked for the maximum operating condition, while hold violations are checked for the minimum operating condition.
- `on_chip_variation`: Declares that the minimum and maximum operating conditions denote the lower and upper bounds of the maximum variation of operating conditions on the chip.

-library lib

Defines the library that contains definitions of the environmental characteristics. It can either be a library name or a collection object.

`set_operating_conditions`

condition

Defines the name of the single operating condition.

-min min_condition

Defines the minimum operating condition for checking minimum delays and hold violations. Use `max` with this argument.

-max max_condition

Defines the maximum operating condition for checking maximum delays and setup violations. Use `min` with this argument.

-min_library min_lib/-max_library max_lib

Defines the library that contains the `min_condition` or `max_condition`. It can either be a library name or a collection object. In addition, make sure you use both `min` and `max` arguments.

-object_list objects

Defines the cells or ports on which to set operating conditions. If you do not use this option, operating conditions are set on the design. This option cannot be used with the `analysis_type` option.

Examples

Consider the following example:

```
set_operating_conditions WCIND11
```

Here, the `WCIND11` operating condition located in the `my_lib.db` library is used to set the operating conditions to `WCIND11` if the `link_path` is `my_lib.db`.

set_output_delay

Syntax

The syntax to specify the `set_output_delay` constraint is as follows:

```
set_output_delay
  [ -clock clock_name ]
  [ -clock_fall ]
  [ -level_sensitive ]
  [ -rise ]
  [ -fall ]
  [ -min ]
  [ -max ]
  [ -add_delay ]
  delay_value
  port_pin_list
```

Arguments

-clock clock_name

Specifies the related clock of the port or pin on which the output delay is to be applied.

-clock_fall

Specifies the delay relative to the falling edge of the clock. While using this argument, you must also specify the `-clock` argument.

-level_sensitive

Specifies that the source of the delay is a level-sensitive latch.

-rise/-fall

Specifies the `delay_value` on account of a rising/falling transition on the specified port. When you do not specify these arguments, rise and fall delay are assumed to be same.

`set_output_delay`

-min/-max

Specifies `delay_value` on the shortest/longest path to the port/pin.

-add_delay

Used when delay on a port is required to be specified with respect to multiple clocks. This happens when multiple paths from output port lead to different clocks.

delay_value

Floating point number in library time units representing the delay.

port_pin_list

Specifies a list of output port or internal pin names in the current design to which `delay_value` is assigned.

Examples

Consider the following example:

```
set_output_delay 1.0 -clock clk1 {OUT1}
```

```
set_output_delay 1.7 -clock clk2 -add_delay {OUT1}
```

Here, the output delay on OUT1 is set with respect to two clocks `clk1` and `clk2`.

```
set_output_delay 1.0 { O1 O2 }
```

Here, delay is specified for a combinational design.

set_port_fanout_number

Syntax

The syntax to specify the `set_port_fanout_number` constraint is as follows:

```
set_port_fanout_number
  [ -min ]
  [ -max ]
  fanout_number
  port_list
```

Arguments

-min/-max

Defines the minimum or maximum condition value.

fanout_number

Declares the number of external fan-out points, which is between 0 to 100000.

port_pin_list

Declares ports, which can be either a collection of ports or a pattern that matches ports on the current design.

Examples

Consider the following example:

```
set_wire_load_model 40x40 [get_ports OUT11*]
set_port_fanout_number 3 [get_ports OUT11*]
```

Here, the external wire load model for ports `OUT11*` is set to `40x40` and an external fan-out number of 3 is specified.

`set_propagated_clock`

set_propagated_clock

Syntax

The syntax to specify the `set_propagated_clock` constraint is as follows:

```
set_propagated_clock  
  object_list
```

Arguments

object_list

Specifies a list of clocks, ports and pins. Clocks on these objects are propagated to their transitive fan-out to actually "compute" the latency through the network.

Examples

Consider the following example:

```
set_propagated_clock [get_clocks CLK1]
```

Here, the latency of CLK1 should be "computed" based on the delays of the various elements found in the clock network, as the clock is being propagated.

set_resistance

Syntax

The syntax to specify the `set_resistance` constraint is as follows:

```
set_resistance
  [ -min ]
  [ -max ]
  resistance_value
  object_list
```

Arguments

-min/-max

Declares that the `resistance_value` is minimum or maximum.

resistance_value

Defines the resistance value for nets in the `object_list`.

object_list

Defines nets on which the `resistance_value` is set.

Examples

Consider the following example:

```
set_resistance 300 {z,y}
```

Here, a resistance of 300 units is set to the `z` and `y` nets.

set_timing_derate

set_timing_derate

Syntax

The syntax to specify the `set_timing_derate` constraint is as follows:

```
set_timing_derate
  -early | -late
  [ -rise ] [ -fall ]
  [ -clock ] [ -data ]
  [ -cell_delay ] [ -cell_check ] [ -net_delay ]
  [ -static ] [ -dynamic ]
  [ -scalar ] [ -variation ]
  derate_value
  object_list
```

Arguments

-early/-late

Use either early or late. The early argument indicates that the derate value be applied on the shortest paths, while the late argument indicates that the derate value be applied on the longest paths.

-rise/-fall

Denotes that the derate value is applicable to rise or fall delays.

-clock/-data

Denotes that the derate factor is applicable to the clock or data path.

-cell_delay/-cell_check/-net_delay

Denotes that derating factor is applicable to cell delays, cell timing checks, or net delays. The `cell_delay` and `net_delay` arguments cannot be used with the `cell_check` argument.

-static/-dynamic

Use with the `net_delay` option to specify that the derate factor be applied to the non-delta portion of net delays only (static) or to the dynamic component of delays only (dynamic).

-scalar/-variation

Denotes that the derate factor be applied to deterministic delays (scalar) or statistical delays (variation).

derate_value

Declares the timing derate value applicable to delays as a scalar multiplicative factor. This argument is a float value.

object_list

Declares a list of cells, library cells, or nets to which the derate factor is applicable.

Examples

You can set derating for constraints by using the `set_timing_derate` command, as shown in the following:

```
set_timing_derate -late 1.3 -cell_check  
set_timing_derate -early 0.8 -cell_check
```

`set_wire_load_model`

set_wire_load_model

Syntax

The syntax to specify the `set_wire_load_model` constraint is as follows:

```
set_wire_load_model
  -name model_name
  [-library lib_spec]
  [ -min ]
  [ -max ]
  [ object_list ]
```

Arguments

-name model_name

Defines the wire load model name.

-library lib_spec

Declares the library in which to search for the `model_name`. If you do not specify this argument, libraries in the `link_path` are searched.

-min/-max

Declares that the model is for minimum or maximum conditions.

object_list

Declares designs, ports, or hierarchical cells.

Examples

Consider the following example:

```
current_design TOP
set_wire_load_model -name LION
```

```
current_instance u11
set_wire_load_model -name TIGER
current_instance u22
set_wire_load_model -name LEOPARD
```

Here, a wire load model of LION is specified on the top level design, a model of TIGER on the u11 hierarchical cell, and a model of LEOPARD on the u11/u22 hierarchical cell.

`set_wire_load_selection_group`

set_wire_load_selection_group

Syntax

The syntax to specify the `set_wire_load_selection_group` constraint is as follows:

```
set_wire_load_selection_group
  selection_group_name
  [-library lib_spec]
  [ -min ]
  [ -max ]
  [ object_list ]
```

Arguments

selection_group_name

Defines the selection group name.

-library lib_spec

Declares the library in which to search for the `selection_group_name`.

-min/-max

Declares that the selection group is for minimum or maximum conditions.

object_list

Declares hierarchical cells or designs.

Examples

Consider the following example:

```
set_wire_load_selection_group wselgrp1 -library tech1_lib
```

Here, the `wselgrp1` selection group from the `tech1_lib` library is

applied to the `current_design`.

List of Topics

About This Book	29
Abstract View Rules	1250
allow_clock_on_output_port	1401
allow_drive	1402
allow_driving_cell	1402
allowed_load_cells	1403
allow_input_transition	1403
Block Rules	137
cg14_prefix	1404
cg14_suffix	1404
check_clock_group_violations	1405
Checking Equivalence between the Same SDC Commands	132
chip	1406
clk_domain_false_path Report	1636
clk_gen01_generate_report	1407
clk_gen09_trav_over_sequential_arc	1407
clk_gen_module	1406
Clock Consistency Rules	168
Clock Generation Rules	171
Clock Group Rules	500
Clock Latency Rules	321
Clock Rules	167
Clock Transition Rules	359
Clock Uncertainty Rules	398
Clock-Mapping Report	1614
Combinational Paths Rules	601
Comparing STA and TC_STA Domain Modes	80
Constraints Equivalence Reports	1613
Constraints Generation	1003
Constraints Generation	92
Constraints Management Rules	120
Constraints Management Rules	1277
Constraints Management	120
Constraints Verification	88
Contents of This Book	30
create_clock	1663
create_generated_clock	1665
default_max_capacitance	1408
default_max_transition	1408

List of Topics

default_min_capacitance	1409
default_min_transition	1410
Defining Technology Library Information.....	84
Defining the Scope of Analysis.....	37
del03_allow_setdelay	1411
Display Information Rules	521
Domain Cyclic Errors Report.....	1575
Domain Matrix Report	1572
Domain Rules.....	530
Dont Touch Rules	442
Down Streaming Rules	570
drive_cells_list	1411
Equivalence Report	1620
equiv_sdc_ambiguous_clock_file.....	1412
equiv_sdc_check_fanin.....	1416
equiv_sdc_check_faninfor_clocks	1416
equiv_sdc_clk_prefix.....	1418
equiv_sdc_clk_suffix	1419
equiv_sdc_clock_precision	1420
equiv_sdc_constraint_file	1421
equiv_sdc_copy_sca_sdc	1432
equiv_sdc_design_equivalence_file	1433
equiv_sdc_design_equivalence_hier_file	1437
equiv_sdc_design_ignore_prefix	1439
equiv_sdc_disambiguate_same_names_clock	1440
equiv_sdc_enable_one_pass	1440
equiv_sdc_fast_exceptions_equivalence	1441
equiv_sdc_ignore_constant_pins_for_top_block_equivalence.....	1442
equiv_sdc_ignore_sca_for_top_block_equivalence	1443
equiv_sdc_ignore_unequivalent_design_obj	1443
equiv_sdc_ignore_value	1444
equiv_sdc_import_sca_from_top	1445
EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT.....	1617
equiv_sdc_object_name_length.....	1445
equiv_sdc_report_blocked_clocks	1446
equiv_sdc_report_type.....	1447
equiv_sdc_script_file.....	1447
equiv_sdc_script_side_file	1448
equiv_sdc_set_tolerance	1449
equiv_sdc_show_violations	1449
equiv_sdc_tolerance	1450

Example 1 - Default Clock Mapping	1479
Example 2 - Partial Short Form Clock Mapping.....	1480
Example 3 - Overlapping Clock Mapping	1481
Example 4 - Partial Overlapping Clock Mapping.....	1482
Example of a Mapping File	1434
Example: Impact on Clock Equivalence.....	1483
Examples of SDC Constraints Prioritization.....	38
Exception Type Priority.....	37
False Path Rules.....	1159
False_Path01 Report	1599
gen_c2cVerify	1459
Generate the clk_domain_false_path Report	1636
Generate the Clock-Mapping Report	1614
Generate the Domain Cyclic Errors Report	1575
Generate the Domain Matrix Report	1572
Generate the Equivalence Report	1620
Generate the EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT	1617
Generate the False_Path01 Report.....	1599
Generate the MCP01 Report.....	1602
Generate the mcp_domain_<integer> Report	1657
Generate the Missing Domains Report	1579
Generate the SDC_Methodology31 Report	1655
Generate the sync_clock_uncertainty Report	1639
Generate the tc_block11_info Report	1568
Generate the tc_clk_gen01a_info Report	1642
Generate the tc_clk_gen01b_info Report	1645
Generate the tc_clock_info Report	1648
Generate the tc_dont_touch_info Report	1630
Generate the tc_dont_use_info Report	1633
Generate the tc_latency_info Report	1652
Generate the tc_report_disable_timing Report.....	1563
Generate the tc_unparsed_commands Report	1566
Generate the TE_Consis01 Report.....	1606
Generate the TE_Consis02 Report.....	1610
Generate the Timing Analysis Consolidated Report	1581
Generate the Timing Constraints Coverage Report	1585
gen_sdc_constraints_file	1451
gen_sdc_csv2sdc_tool.....	1452
gen_sdc_merge_seed	1451
gen_sdc_param_file.....	1452
gen_sdc_phase	1459

List of Topics

gen_sdc_user_param_file	1457
Handling Black Boxes	86
Handling Technology Library Cell Pin Names	85
High Fan-out Rules	452
I/O Rules	600
Identifying Missing Domains and Cyclic Dependencies	82
ignore_incremental_equivalence	1460
ignore_io_if_fp	1461
ignore_sdc_constraints_file	1463
Incrementally Generating SDC	92
Inferring Clock Domain Information	77
Inferring Clock Domains When No Domain is Specified	76
inp_delay_margin	1463
inp_percent	1464
inp_percent_max	1465
inp_percent_min	1465
Input Delay Rules	627
Input Transition Rules	714
Input/Output Consistency Rules	749
Interpret the clk_domain_false_path Report	1637
Interpret the Clock-Mapping Report	1615
Interpret the Domain Matrix Report	1573
Interpret the Equivalence Report	1621
Interpret the EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT	1618
Interpret the mcp_domain_<integer> Report	1658
Interpret the Missing Domains Report	1580
Interpret the SDC_Methodology31 Report	1656
Interpret the sync_clock_uncertainty Report	1640
Interpret the tc_block11_info Report	1569
Interpret the tc_clk_gen01a_info Report	1643
Interpret the tc_clk_gen01b_info Report	1646
Interpret the tc_clock_info Report	1649
Interpret the tc_dont_touch_info Report	1632
Interpret the tc_dont_use_info Report	1634
Interpret the tc_latency_info Report	1653
Interpret the tc_report_disable_timing Report	1564
Interpret the tc_unparsed_commands Report	1567
Interpret the Timing Analysis Consolidated Report	1582
Interpret the Timing Constraints Coverage Report	1590
Interpreting the False_Path01 Report	1600
Interpreting the MCP01 Report	1603

Interpreting the TE_Consis01 Report.....	1607
Interpreting the TE_Consis02 Report.....	1611
io07a_range	1467
io07_ports.....	1467
library_clk_gen_naming	1466
List of Supported SDC Constraints for Equivalence	123
Load Rules	850
logic_level_max	1469
logic_level_min	1468
Mapping File Structure	1433
MCP01 Report.....	1602
mcp_domain_<integer> Report.....	1657
Merge SDC Files	1263
Merging Multiple Modes for SDC Files	112
Merging SDC Files from Lower-level Blocks	114
Methodology Rules	869
Miscellaneous Reports.....	1627
Miscellaneous Rules	989
Missing Domains Report	1579
Mnemonics in the Domains Spreadsheets	555
Multi-Cycle Path Rules.....	1197
multimode_sdc_ambiguous_clock_file	1469
num_falsepath_max	1474
num_mcp_path_max.....	1474
op_delay_margin	1475
op_percent	1476
op_percent_max	1476
op_percent_min	1477
Order of Reading SDC Data.....	40
Other Rule Parameters	1559
Other Rules	999
Other Timing Exception Rules.....	1230
Output Delay Rules.....	770
Overview.....	1401
Path Specification Priority	38
pt.....	1500
Related Reports	1564
Related Reports	1567
Related Reports	1570
Related Reports	1574
Related Reports	1578

List of Topics

Related Reports.....	1580
Related Reports.....	1584
Related Reports.....	1597
Related Reports.....	1601
Related Reports.....	1605
Related Reports.....	1609
Related Reports.....	1612
Related Reports.....	1616
Related Reports.....	1619
Related Reports.....	1626
Related Reports.....	1632
Related Reports.....	1635
Related Reports.....	1638
Related Reports.....	1641
Related Reports.....	1644
Related Reports.....	1647
Related Reports.....	1651
Related Reports.....	1654
Related Reports.....	1656
Related Reports.....	1659
report_inactive_gen_clocks.....	1477
Reporting Rules.....	1041
scan_insert.....	1503
scan_shift.....	1503
schematic_opt.....	1504
sdc_consistent_ambiguous_clock_file.....	1478
sdc_consistent_check_faninout_for_clocks.....	1482
sdc_consistent_clock_precision.....	1484
sdc_consistent_disambiguate_same_named_clocks.....	1485
sdc_consistent_object_name_length.....	1486
sdc_consistent_report_blocked_clocks.....	1487
sdc_consistent_report_inactive_gen_clocks.....	1487
sdc_consistent_report_type.....	1488
sdc_consistent_show_violations.....	1489
sdc_consistent_tolerance.....	1489
SDC_DnStrm02_control_point.....	1491
SDC_DnStrm02_control_signal.....	1492
SDC_DnStrm02_pedge_logic.....	1493
SDC_DnStrm02_seq_cell.....	1493
SDC_DnStrm04a_template.....	1497

SDC_DnStrm04_template.....	1496
SDC_MergeBlocks_deglitch_cell.....	1490
SDC_MergeBlocks_deglitch_cell_clock.....	1491
SDC_Methodology01_allow_block_pins.....	1494
SDC_Methodology01_commands_list.....	1494
SDC_Methodology02_forbid_constraints.....	1495
SDC_Methodology31_Report.....	1655
set.....	1668
set_annotated_transition.....	1669
set_case_analysis.....	1671
set_clock_gating_check.....	1672
set_clock_groups.....	1674
set_clock_latency.....	1676
set_clock_sense.....	1678
set_clock_transition.....	1684
set_clock_uncertainty.....	1686
set_data_check.....	1688
set_disable_timing.....	1690
set_dont_touch.....	1691
set_drive.....	1692
set_driving_cell.....	1693
set_false_path.....	1696
set_fanout_load.....	1699
set_ideal_latency.....	1700
set_ideal_net.....	1702
set_ideal_network.....	1703
set_ideal_transition.....	1704
set_input_delay.....	1706
set_input_transition.....	1708
set_load.....	1710
set_logic_dc.....	1712
set_logic_one.....	1713
set_logic_zero.....	1714
set_max_area.....	1715
set_max_capacitance.....	1716
set_max_delay.....	1719
set_max_delay/set_min_delay.....	1719
set_max_dynamic_power.....	1721
set_max_fanout.....	1718
set_max_leakage_power.....	1722
set_max_time_borrow.....	1723

List of Topics

set_max_transition	1724
set_min_capacitance	1726
set_min_delay	1719
set_min_transition	1727
set_multicycle_path	1729
set_operating_conditions	1732
set_output_delay	1734
set_port_fanout_number	1736
set_propagated_clock	1737
set_resistance	1738
set_sense	1682
set_timing_derate	1739
Setting the Domain Mode	82
Setup Rules	1053
set_wire_load_model	1741
set_wire_load_selection_group	1743
Source Synchronous Interfaces	1547
Specifying Clocks for the SDC_GenerateIncr Rule	110
Specifying Domains	77
Specifying Non-Sequential Instances	1434
Specifying One-to-Many Mapping	1436
Specifying the SDC File to Check SDC Equivalence	124
Specifying the Skeleton SDC File to Check SDC Equivalence	126
SpyGlass Design Constraints File Structure Rules	1078
Stopping Clock Propagation	1680
strict	1505
Supported Attributes List	74
Supported Non-SDC Constraints	75
Supported SDC Constraints	40
sync_clock_uncertainty Report	1639
tc_allow_async_pins	1505
tc_allow_design_obj	1507
tc_allow_mm_or_io	1507
tc_at_speed_testing_mode	1506
tc_block11_info Report	1568
tc_bus_merge	1508
tc_c2c_max_cycles	1518
tc_check_for_multiple_clock_fanin	1508
tc_checkmcp_report_sorted_clock	1509
tc_clk_compat	1510
tc_clk_gen01a_info Report	1642

tc_clk_gen01b_info Report	1645
tc_clk_register	1511
tc_clock_fanout_limit	1511
tc_clock_info Report	1648
tc_combo_check	1512
tc_combopaths03_clk_multiplier.....	1513
tc_comments_cmd_file	1514
tc_compare_cmd_file.....	1513
tc_ct17_clock_trans.....	1515
tc_ct17_drive	1516
tc_ct17_driving_cell.....	1516
tc_ct17_input_trans.....	1517
tcdecompile	1557
tc_disable_caching	1518
tc_domain_mode.....	1519
tc_dont_touch_info Report.....	1630
tc_dont_use_info Report	1633
tc_drc_spec_file	1521
tc_enable_param_sdc_flow.....	1524
tc_enable_preset_clear_arcs.....	1522
tc_express_check.....	1523
tc_fanout_limit.....	1525
tc_high_fan_nets_file.....	1522
tc_honor_virtual_clk	1523
tc_ignore_async_ports	1525
tc_ignore_block_path.....	1526
tc_ignore_clk_outports.....	1510
tc_ignored_commands	1531
tc_ignore_if_clk_op_port	1527
tc_ignore_iodelay_if_maxdelay	1528
tc_ignore_io_if_minmax_delay.....	1527
tc_ignore_latch_enable	1529
tc_ignore_libcells.....	1529
tc_ignore_min.....	1530
tc_ignore_missing_minmax_delay	1531
tc_ignore_modes.....	1532
tc_ignore_nets.....	1530
tc_ignore_scg	1532
tc_ignore_te	1533
tc_inter_block_delay.....	1534
tc_io_delay_bus_margin.....	1534

List of Topics

tc_io_reg.....	1535
tc_latency_info Report	1652
tc_lvpc.....	1535
tc_modemerge_backref_info	1536
tc_num_fp_max	1537
tc_num_load_max	1537
tc_num_mcp_max	1538
tc_opt01_port_des	1538
tc_overconstrained_factor.....	1539
tc_regression_mode.....	1539
tc_report_backref_all	1540
tc_report_clks_start_end_points.....	1541
tc_report_csv_messages	1542
tc_report_disable_timing Report.....	1563
tc_report_unconnected_points	1542
tc_report_verbose	1543
tc_setup_hold	1544
tc_show_all_unconstrained_flops.....	1544
tc_show_sca_on_terminals	1545
tc_similar_clocks_tolerance_levels.....	1545
tc_solver_run_time.....	1546
tc_source_syn_clks.....	1547
tc_start_end_max_count	1549
tc_start_end_point_details.....	1550
tc_stop_parsing_ignored_commands.....	1550
tc_tech	1552
tc_te_conflict_fast_run.....	1551
tc_unparsed_commands Report.....	1566
tc_violate_ports	1552
tc_virtual_clock_prefixes	1553
tc_virtual_clock_suffixes.....	1554
tc_waive	1555
tc_waive_const_struct05	1555
tc_xbuf_ignore_clks	1556
TE_Consis01 Report	1606
TE_Consis02 Report	1610
Test Rules	1114
Timing Analysis Consolidated Report	1581
Timing Checking Rules	1142
Timing Constraints Coverage Report.....	1585
Timing Constraints Domain Reports.....	1571

Timing Constraints Information Reports.....	1562
Timing Coverage Rules.....	1149
Timing Exception Reports	1598
Timing Exception Rules	1155
Timing Exception Verification	119
Type of Information in the clk_domain_false_path Report	1636
Type of Information in the Clock-Mapping Report.....	1615
Type of Information in the Domain Cyclic Errors Report	1576
Type of Information in the Equivalence Report.....	1621
Type of Information in the EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT ..	1618
Type of Information in the False_Path01 Report	1599
Type of Information in the MCP01 Report.....	1602
Type of Information in the mcp_domain_<integer> Report	1658
Type of Information in the Missing Domains Report.....	1580
Type of Information in the SDC_Methodology31 Report	1655
Type of Information in the sync_clock_uncertainty Report.....	1639
Type of Information in the tc_block11_info Report	1568
Type of Information in the tc_clk_gen01a_info Report	1642
Type of Information in the tc_clk_gen01b_info Report	1645
Type of Information in the tc_clock_info Report	1648
Type of Information in the tc_dont_touch_info Report	1630
Type of Information in the tc_dont_use_info Report	1633
Type of Information in the tc_latency_info Report	1652
Type of Information in the tc_report_disable_timing Report	1563
Type of Information in the tc_unparsed_commands Report	1566
Type of Information in the TE_Consis01 Report	1606
Type of Information in the TE_Consis02 Report	1610
Type of Information in the Timing Analysis Consolidated Report.....	1582
Type of Information in the Timing Constraints Coverage Report	1586
Type of Information Reported in the Domain Matrix Report	1573
Typographical Conventions	31
Understanding Equivalence of SDC Files	121
Understanding the Equivalence Flow	123
Use Model for Inferring Domain	76
Using SDC Autofix	115
verbose.....	1558
View the clk_domain_false_path Report	1636
View the Clock-Mapping Report	1615
View the Domain Cyclic Errors Report.....	1575
View the Domain Matrix Report	1572
View the Equivalence Report	1621

List of Topics

View the EQUIV_SDC_NON_EQUIVALENT_DESIGN_REPORT	1617
View the False_Path01 Report	1599
View the MCP01 Report	1602
View the mcp_domain_<integer> Report.....	1657
View the Missing Domains Report	1579
View the SDC_Methodology31 Report.....	1655
View the sync_clock_uncertainty Report	1639
View the tc_block11_info Report.....	1568
View the tc_clk_gen01a_info Report.....	1642
View the tc_clk_gen01b_info Report.....	1645
View the tc_clock_info Report.....	1648
View the tc_dont_touch_info Report.....	1630
View the tc_dont_use_info Report.....	1633
View the tc_latency_info Report.....	1652
View the tc_report_disable_timing Report	1563
View the tc_unparsed_commands Report.....	1566
View the TE_Consis01 Report	1606
View the TE_Consis02 Report	1610
View the Timing Analysis Consolidated Report	1581
View the Timing Constraints Coverage Report.....	1585
xBuf	1156