



PrimeTime Workshop

Student Guide

10-I-034-SSG-015 2018.06

Synopsys Customer Education Services
690 E. Middlefield Road
Mountain View, California 94043

Workshop Registration: <http://training.synopsys.com>

Copyright Notice and Proprietary Information

© 2018 Synopsys, Inc. All rights reserved. This software and documentation contain confidential and proprietary information that is the property of Synopsys, Inc. The software and documentation are furnished under a license agreement and may be used or copied only in accordance with the terms of the license agreement. No part of the software and documentation may be reproduced, transmitted, or translated, in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without prior written permission of Synopsys, Inc., or as expressly provided by the license agreement.

Destination Control Statement

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

Disclaimer

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

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at <http://www.synopsys.com/Company/Pages/Trademarks.aspx>.

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

Third-Party Links

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

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

Table of Contents

Unit i: Introduction to STA in PrimeTime

Facilities	i-2
Workshop Goal	i-3
Workshop Target Audience	i-4
Workshop Prerequisite Knowledge	i-5
Introductions	i-6
What is PrimeTime?	i-7
PrimeTime in the Implementation Flow	i-8
What is Static Timing Analysis (STA)?	i-9
STA is Path Based	i-10
STA is Constraint Driven.....	i-11
Signoff Considerations.....	i-12
Agenda : Flow and Methodology	i-13
Agenda :Best Practices	i-14
Agenda : Signoff Considerations (Continued).....	i-15
Icons Used in this Workshop	i-16

Unit 1: STA Concepts and Flow in PrimeTime

Agenda	1-1
Unit Objectives	1-2
PrimeTime Verifies a Number of Timing Checks.....	1-3
Summary Report of Timing Checks verified	1-4
STA is Path Based: Start and End Points of Timing Paths.....	1-5
Summary Report of Violating Path Types.....	1-6
Setup Timing (Max Delay) Analysis	1-7
Timing Report for Setup (Max Delay Analysis).....	1-8
Hold Timing (Min Delay) Analysis.....	1-9
Timing Report for Hold (Min Delay Analysis)	1-10
PrimeTime Inputs and Outputs	1-11
Timing Analysis Flow in PrimeTime	1-12
Step 1a: Load Design and Check.....	1-13
Step 1b: Load Libraries and Check.....	1-14
Step 2a: Read Parasitics	1-15
Step 2b: Check Parasitic Annotation	1-16
Step 3: Source Constraints and check for correctness	1-17
Confirming Constraint Completeness.....	1-18
Checking for Ignored Timing Exceptions.....	1-19
Recalling Timing Analysis Flow in PrimeTime	1-20
Step 4a: Update Timing	1-21
Coverage Analysis	1-22
Step 4b: Generate detailed Reports.....	1-23

Table of Contents

Step 6: Exit.....	1-27
Checking the Run Script Execution	1-28
PrimeTime Setup File	1-29
PrimeTime Run Script : Seed Script Using RMGEN	1-30
Reference Methodology (RMGEN) Generated Files	1-31
PrimeTime Run Script pt.tcl: Generated by RMGEN	1-32
Script syntax checking and running PrimeTime	1-33
Runtime and Memory Profiling for Monitoring Performance.....	1-34
Lab 1: Timing Analysis Flow	1-35
Appendix.....	1-36
Useful Commands for Lab when in pt_shell 1/2	1-37
Useful Commands when in pt_shell 2/2	1-38

Unit 2: Methodology: Qualifying Constraints

Agenda	2-1
Unit Objectives	2-2
Timing Analysis Flow in PrimeTime – Validating Constraints	2-3
Requirements for Complete STA.....	2-4
Constraint Warning: No Input Delay	2-5
Constraint Warning: No Output Delay	2-6
Constraint Warning: No Clock	2-7
Timing Checks: Vendor Specified and User Specified	2-8
Untested Timing Check: false_paths	2-9
Untested Timing Check: user_disabled	2-10
Untested Timing Check: constant_disabled.....	2-11
Untested Timing Check: no_paths.....	2-12
Test For Understanding.....	2-13
Introducing Tools of the Trade Job Aid.....	2-14
Example Design with a Constraint Issue	2-15
How Does all_fanin Work?	2-16
Another Start Point Example	2-17
Final Start Point Example	2-18
Further Details on Start Point	2-19
“Sketch” Cell Connections	2-20
Find the Master Clock for Defining Generated Clock.....	2-21
How Does get_attribute Work?	2-22
Find Disabled Timing Arcs Along Path.....	2-23
Identify Causes for Disabled Arcs	2-24
Find the User Specified Case Values.....	2-25
How Does report_case_propagation Work?	2-26
Dismiss Clock Gating Logic	2-27
Sketch a Schematic	2-28
Expand Sketch One Level At a Time	2-29
How Does all_fanout Work?	2-30

Table of Contents

Identifying Constraints from Timing Reports.....	2-31
Clock Latency Components.....	2-32
Pre Versus Post Clock Tree Synthesis (CTS).....	2-33
Specifying Clock jitter	2-34
Example: Cycle to Cycle Jitter	2-35
Example: Duty-Cycle Jitter.....	2-36
Example: No Jitter	2-37
Clock Uncertainty and Skew	2-38
Clock Constraints From Timing Report	2-39
Reporting Clock Source Latency and Clock Network Cells	2-40
Interface Paths: Input Ports.....	2-41
Input Delay Constraint From Timing Report.....	2-42
A Cleaner Solution – Constraining Interface Paths	2-43
Interface Paths: Output Ports	2-44
Output Delay Constraint From Timing Report.....	2-45
Constraining Output Port for Setup and Hold Requirements	2-46
Review of Unit Objectives.....	2-47
Lab 2: Constraints in a Timing Report	2-48
Appendix.....	2-49
Test For Understanding : Clock Constraints 1/2.....	2-50
Test For Understanding : Clock Constraints 2/2.....	2-51
Test For Understanding : Input Delay Constraints	2-52
Test For Understanding : Output Delay Constraints 1/2	2-53
Test For Understanding : Output Delay Constraints 2/2	2-54
Checking Constraints Applied on Ports.....	2-55

Unit 3: Methodology: Generating Reports

Agenda	3-1
Objectives	3-2
Timing Analysis Flow in PrimeTime – Generating Reports	3-3
Generating Reports Methodology – Summary Reports.....	3-4
Quality of Design: report_qor.....	3-5
Where are the Violations? : report_global_timing.....	3-6
Violations Sorted By Clock : report_constraint -all	3-7
Checks Exercised: report_analysis_coverage	3-8
Listing All Pins Having Setup Violations.....	3-9
Examples Using report_analysis_coverage	3-10
Generating Reports Methodology – Detailed Reports.....	3-11
Timing Path Details: Built From Timing Arcs	3-12
Common Types of Timing Arcs in Libraries.....	3-13
Rise/Fall Transitions Along a Timing Path	3-14
Generating Timing Reports Specifying Endpoint Transition	3-15
Data Arrival Time: Edge Sensitivity negative_unate	3-16
Data Arrival Time: Edge Sensitivity non_unate.....	3-17

Table of Contents

Reporting Library Arcs: Inverter	3-18
Reporting Library Arcs: Flip-Flop.....	3-19
Hierarchy in a Timing Path.....	3-20
Identify Hierarchy in a Timing Report	3-21
Default Behavior of report_timing.....	3-22
report_timing for each path group	3-23
The nworst vs. max_paths Options of report_timing.....	3-24
Exercise on nworst vs. max_Paths options	3-25
Reporting Multiple Paths to a Single Endpoint	3-26
Cover Design Reporting (report_timing –cover_design)	3-27
Reporting Summary Using report_timing	3-28
Another “Summary” Report Using report_timing	3-29
Filtering Paths Based by Slack Amount	3-30
Reporting Paths From, To, Through or Exclude.....	3-31
Clock Ports Versus Clock Objects.....	3-32
Recommendation – Being Specific.....	3-33
Example: Specifying Capturing Clock Edge	3-34
Example: Specifying Net Name Along Timing Path.....	3-35
Half Cycle Path: Clock Edges Used For Setup and Hold.....	3-36
Reporting a Half Cycle Path : Setup	3-37
Reporting A Half Cycle Path : Hold	3-38
Generating Reports Methodology – Delay Calculation.....	3-39
Including Input Pins in a Timing Report	3-40
NLDM Driver and Receiver Models – Limited Accuracy	3-41
Cell Delay Calculation Reporting with NLDM library	3-42
Net Delay Calculation Reporting with NLDM library	3-43
CCS Driver and Receiver Models – Most Accurate!.....	3-44
Cell Delay Calculation Report With CCS Library	3-45
Net Delay Calculation Report With CCS Library	3-46
report_timing != report_delay_calculation ?	3-47
report_delay_calculation For Min Delay	3-48
Review of Unit Objectives.....	3-49
Lab 3: Generate Reports - Control Which Paths Are Reported.....	3-50
Appendix 1	3-51
Enabling CCS delay calculation	3-52
Timing report with unannotated parasitics	3-53
Cell delay calculation using Library Table and Lumped load.....	3-54
Cell Delay Arcs For report_delay_calculation	3-55
Conditional Timing Arcs	3-56

Unit 4: Constraining Multiple Clocks

Agenda	4-1
Unit Objectives	4-2
Timing Analysis Flow in PrimeTime – Generating Reports	4-3

Table of Contents

Clocks and STA : Three Types of Clocks	4-4
What Are Primary Clocks?	4-5
Generated Clocks: Internally Derived Clocks	4-6
Generated Clocks: Source Latency	4-7
More Clocks - Source Synchronous Interface	4-8
Generated Clocks: Outgoing Clocks.....	4-9
Third Kind of Clock - Virtual Clocks	4-10
Use report_clock For All Clocks	4-11
How Many Clocks Are In Your Design?	4-12
Ignorable check_timing warning	4-13
Source Latency Calculation Reporting for Generated Clocks	4-14
Clocks and STA: Inter Clock Relationships	4-15
Timing Between (A)synchronous Clocks.....	4-16
Exhaustive Analysis != Intended Analysis	4-17
Interacting Clocks and Multiple STA Runs.....	4-18
Single Analysis with Multiple Clocks!	4-19
Different Example: Multiple Clocks.....	4-20
Test For Understanding.....	4-21
Asynchronous Clocks	4-22
Identify All Clock Crossings	4-23
How Do You Perform These Checks?.....	4-24
Generate Timing Reports Between Clocks.....	4-25
Interpret Timing Reports Between Clocks	4-26
Clock Edges used for Setup and Hold	4-27
Messages During Timing Updates.....	4-28
Timing Reports for Asynchronous Clocks	4-29
Reports for Unconstrained Paths	4-30
No Paths versus No Constrained Paths	4-31
Two General Guidelines	4-32
Lab 4: Getting to Know Your Clocks	4-33

Unit 5: Additional Checks and Constraints

Agenda	5-1
Unit Objectives	5-2
Timing Analysis Flow in PrimeTime – Generating Reports	5-3
Timing Checks Verified by STA	5-4
I. Asynchronous Clear/Reset Pins.....	5-5
Timing Report Recovery.....	5-6
Test For Understanding.....	5-7
Path Groups – Full Picture	5-8
II. Clock Gating Checks.....	5-9
Which Clock Edges Are Used?.....	5-10
Timing Report for Clock Gating.....	5-11
III. Data to Data Check	5-12

Table of Contents

Specify Data to Data Checks	5-13
Another Application for a Data-to-Data Check.....	5-14
IV. Clock Min Pulse Width	5-15
Summary Min Pulse Width Reports	5-16
Test For Understanding	5-17
Summary Report for All Timing Checks.....	5-18
1. Latch based analysis [Default = Old]	5-19
Latch with Zero Time Borrow : Path to L2/D	5-20
Latch w/ Zero Time Borrow : Path from L2/G.....	5-21
Latches with Time Borrow: Path to L2/D.....	5-22
Time Borrowing Information in Report [Default = Old].....	5-23
Latches with Time Borrow: Path from L2/D.....	5-24
Time Borrowed in Previous Stage is Now Returned [Old]	5-25
New Latch through Analysis	5-26
Old vs. New Latch Analysis : Differences.....	5-27
OLD: Data arrives before transparency: Path to Latch/D.....	5-28
NEW: Data arrives before transparency: Path to Latch/D.....	5-29
New: Data arrives during transparency: -to LAT/D	5-30
New: Data arrives after transparency: -to LAT/D	5-31
New: Early path -through LAT/D (Ideal Clock).....	5-32
New: Early path -through LAT/D (Propagated Clock).....	5-33
New: Data arrives during transparency: -through LAT/D	5-34
New: Recovery path -through LAT/D	5-35
2. Multicycle Paths.....	5-36
Default reported path is single cycle.....	5-37
Specifying Multicycle Path for Setup	5-38
Specifying Multicycle path for Hold (New data every 6 cycles).....	5-39
Reporting a multi cycle path with report_timing	5-40
3. Combinational Feedback Loops	5-41
STA and Combinational Loops	5-42
Issues with Combinational Loops.....	5-43
Which Arc is Broken by PrimeTime?.....	5-44
Examine Path with Broken Arc	5-45
4. Non-Unate Cells (Arcs) in clock paths	5-46
PTE-070 non-unateness and set_sense	5-47
Clock Used as Data.....	5-48
Review of Unit Objectives.....	5-49
Lab 5: Additional Checks and Constraints	5-50

Unit 6: Best Practices Debugging Reports

Agenda	6-1
Unit Objectives	6-2
Timing Analysis Flow in PrimeTime – Generating Reports	6-3
Do NOT start analysis with report_timing.....	6-4

Table of Contents

Much Easier With check_timing	6-5
Hints on Debugging report_timing	6-6
Test For Understanding.....	6-7
Is My Timing Report Incorrect/Questionable?	6-8
1. Incorrect Source Latency in Timing Report	6-9
1. Incorrect Source Latency in Timing Report : Debugging	6-10
2. Zero Source Latency for Generated Clock	6-11
2. The Generated Clock Specified	6-12
2. Unsatisfiable Generated Clock!	6-13
2. Zero Source Latency for Generated Clock : Solution.....	6-14
3. Sequential Loop in Source Latency Network	6-15
3. Sequential Loop in Source Latency Network : Debugging	6-16
4. Source Latency included with Input Delay.....	6-17
5. Zero CRPR.....	6-18
5. Zero CRPR : Solution	6-19
6. Zero CRPR with propagated clock	6-20
6. Zero CRPR with propagated clock : CRPR Threshold!	6-21
7. Incorrect CRPR Common Point.....	6-22
7. Incorrect CRPR Common Point in Reports	6-23
7. Incorrect CRPR Common Point: Cause.....	6-24
8. Zero CRP in Timing Report	6-25
8. Zero CRP in Timing Report: Clock to Data CRP	6-26
9. Unexpected Report with report_timing -slack_greater.....	6-27
9. Unexpected Report with report_timing -slack_greater.....	6-28
9. Unexpected Report with report_timing -slack_greater.....	6-29
10. PBA WNS report_qor/report_glob & report_timing Differ	6-30
10. PBA WNS report_qor/report_glob & report_timing Differ	6-31
10. PBA WNS report_qor/report_glob & report_timing Differ	6-32
11. Negative Cell Delay in Timing Report	6-33
11. Negative Cell Delay in Timing Report : The issue.....	6-34
12. No Paths in Timing Report	6-35
12. No Paths in Timing Report : Debugging the cause.....	6-36
12. No Paths in Timing Report : find_blocking_arcs	6-37
Review of Unit Objectives.....	6-38

Unit 7: Signoff: Path Based Analysis (PBA)

Agenda	7-1
Unit Objectives	7-2
Timing Analysis Flow in PrimeTime – Generating Reports	7-3
Fast Analysis != Final Analysis	7-4
Graph Based Analysis (GBA).....	7-5
Path-Based Analysis (PBA)	7-6
Why Computationally Prohibitive?	7-7
Why Use Path-Based Analysis (PBA)?	7-8

Table of Contents

Using Path-Based Analysis.....	7-9
PBA Modes: Path vs Exhaustive	7-10
Test For Understanding.....	7-11
exhaustive PBA mode: when does it stop.....	7-12
Exhaustive PBA: Performance Recommendation	7-13
Exhaustive PBA: Usage Recommendation	7-14
Recalculated Paths in Timing Reports.....	7-15
When to use: Path vs Exhaustive	7-16
Test For Understanding.....	7-17
Things to Remember When Using PBA:.....	7-18
report_delay_calculation Does Not Match report_timing -pba	7-19
Making report_delay_calculation Match PBA Delay.....	7-20
Is My Design Ready for PBA?	7-21
Path-Based Analysis : A Recommended Flow	7-22
Case-1: Not Suitable for Path Based Analysis	7-23
Case-2: Not Suitable for Path Based Analysis	7-24
Case-3: Ready for Path Based Analysis	7-25
Case-4: Must for Exhaustive Path Based Analysis	7-26
Derate Only PBA	7-27
Path Based Analysis: Summary	7-28
Lab 7: PBA	7-29

Unit 8: Signoff: Crosstalk Delay Analysis

Agenda	8-1
Unit Objectives	8-2
Timing Analysis Flow in PrimeTime – SI Delay Analysis	8-3
What is Crosstalk?	8-4
PrimeTime Inputs and Outputs	8-5
PrimeTime SI Delay Analysis Run Script	8-6
PrimeTime SI Terminology	8-7
What Is a Stage Delay?	8-8
What Are Timing Windows?	8-9
What Are Overlapping Timing Windows?	8-10
Partial Window Overlap is also considered!	8-11
What Is Crosstalk-Induced Delay?	8-12
Switching Bumps versus Delta Delay.....	8-13
Key Messages : SI Delay Analysis	8-14
Ensuring Correct Inputs for Crosstalk Analysis	8-15
Reading Parasitics – Complete and No Errors.....	8-16
Reading Parasitics – Background Processing	8-17
Resolve All Parasitic Warnings	8-18
Boundary (I/O) Drive and Load Constraint Requirements.....	8-19
Constraining Input Ports with Input Delays.....	8-20
(A)synchronous Clocks: Delta Delay Analysis	8-21

Table of Contents

Asynchronous Clocks	8-22
Logically Exclusive Clocks	8-23
Physically Exclusive Clocks	8-24
Clock Relationships – Summary (1/2).....	8-25
Clock Relationships – Summary (2/2).....	8-26
Constraining Static Nets – Case Analysis.....	8-27
PrimeTime SI - CRPR in Setup Analysis	8-28
PrimeTime SI - CRPR in Hold Analysis	8-29
update_timing Under the Hood.....	8-30
Composite Aggressor Mode	8-31
Timing Window Overlap : all_paths SI Window Alignment.....	8-32
Introducing all_path_edges Window Alignment (Max path)	8-33
Introducing all_path_edges Window Alignment (Min path).....	8-34
SI and Path-Based Analysis	8-35
Using report_delay_calculation -crosstalk.....	8-36
Review Unit Objectives	8-37
Lab 8: Perform Crosstalk Delay Analysis	8-38
Appendix: Electrical Filtering Details	8-39
What is Electrical Filtering?	8-40
Accumulated versus Individual Filtering.....	8-41
Step #1: Aggressor Sorting	8-42
Step #2: Combined Filtering.....	8-43
Voltage Bump Summing.....	8-44

Unit 9: Signal Integrity: Crosstalk Noise Analysis

Agenda	9-1
Unit Objectives	9-2
Timing Analysis Flow in PrimeTime – SI Noise Analysis.....	9-3
PTSI Noise Inputs and Outputs.....	9-4
PTSI Noise Run Script.....	9-5
What Is Crosstalk-Induced Noise?.....	9-6
Unsafe noise bumps : Crossing the Logic Thresholds.....	9-7
Example Functional Failures due to Noise	9-8
PrimeTime SI Noise Terminology	9-9
Noise Bump Modeling – Height and Width	9-10
The Size of a Noise Bump is Affected By	9-11
Noise Bumps Calculated in Two Regions (By Default).....	9-12
Noise Immunity Curve vs. Noise Margin Constraint	9-13
Reporting Noise Slack Using 3 Different Slack Types	9-14
Interpreting report_noise (1/2).....	9-15
Interpreting report_noise (2/2).....	9-16
Special Cases: No Slack and Slack Equal to “Infinity”	9-17
Beyond Rail Analysis	9-18
Noise Propagation.....	9-19

Table of Contents

Example Noise Report	9-20
Propagation of a Noise Violation.....	9-21
Accounting For Incomplete Library	9-22
Example – Injecting Noise.....	9-23
More Details from Noise Calculation Report	9-24
Adding Margin and Excluding Nets for Noise Analysis	9-25
Detecting Double Switching Due to Crosstalk	9-26
Introducing CCS Noise Model in Noise Analysis	9-27
Noise Bump Calculation with CCS Noise	9-28
POSITIVE Slack Reporting.....	9-29
Calculated Slack Reporting.....	9-30
Noise Immunity Curve Display in GUI.....	9-31
report_noise_calculation: Aggressors Section.....	9-32
report_noise_calculation: Constraint Section	9-33
Two Modes of Reporting with CCS Noise	9-34
Reporting Example : report_at_source	9-35
Reporting Example : report_at_endpoint.....	9-36
Reporting Violation Sources	9-37
Review Unit Objectives	9-38
Lab 9: Perform Crosstalk Noise Analysis.....	9-39
Appendix.....	9-40
Noise Analysis: Driver Weakening	9-41
CCS Noise Attributes on the Library Cell	9-42
Example: report_at_source	9-43
Example: report_at_endpoint.....	9-44

Unit 10: Correlation: POCV and AWP Analysis

Agenda	10-1
Unit Objectives	10-2
Timing Analysis Flow in PrimeTime -- Analysis.....	10-3
Traditional Accounting for On Chip Variation (OCV) Effects	10-4
Example: OCV Analysis with Timing Derate	10-5
Derating Factor in Timing Reports	10-6
CRP in a Timing Report	10-7
Timing Pessimism with OCV Analysis : TFU	10-8
AOCV Calculates Derating Based on Depth and Distance	10-9
Parametric On-Chip Variation Modeling of Random Variation.....	10-10
POCV Input Data: Sigma (σ) - Two Formats.....	10-11
POCV Path Calculation Example	10-12
POCV Input Data: Distance-based Derating	10-13
POCV Timing Report	10-14
Running PrimeTime with POCV using Side File	10-15
Running PrimeTime with POCV using LVF	10-16
The 3 commands needed for POCV Debugging.....	10-17

Table of Contents

Incr. Column and Statistical Adjustment	10-18
Statistical Graph Pessimism.....	10-19
Statistical Graph Pessimism in report_timing -var	10-20
POCV Summary	10-21
Introduction to Advanced Waveform Propagation (AWP)	10-22
Causes of Waveform Distortions	10-23
Long Tail Effect.....	10-24
Strong Backward Miller Effect.....	10-25
AWP Correlation With HSPICE.....	10-26
Advanced Waveform Propagation Visualization in the GUI	10-27
Using Advanced Waveform Propagation	10-28
AWP Summary	10-29
Review Unit Objectives	10-30
Appendix	10-31
Sources of Delay Variations	10-32
Random vs. Systemic Delay Variations.....	10-33
POCV Precedence Rules: POCV Side File vs. LVF	10-34
How is sigma_slack Calculated?	10-35
Transition Variation is Combined Into Cell Delay Variation.....	10-36
POCV coefficient from report_delay_calculation -derate	10-37
Specifying POCV guardband vs. scaling factor	10-38
Transition Variation in report_delay_calculation -derate	10-39
POCV LVF in report_delay_calculation -derate	10-40
POCV Constraint Variation in report_delay_calculation	10-41
POCV Constraint Variation in report_timing -variation	10-42
Example: Statistical Graph Pessimism	10-43

Unit 11: Timing Closure: ECO/What If Analysis

Agenda	11-1
Unit Objectives	11-2
Signoff-driven Physically Aware ECO Flow	11-3
Recommended ECO Guidance Flow in PrimeTime-SI.....	11-4
Power (and area) recovery: fix_eco_power	11-5
DRC and Noise Fixing: fix_eco_drc	11-6
ECO DRC fixing : Verbose mode	11-7
Iterating fix_eco_drc	11-8
Setup and Hold Time Fixing: fix_eco_timing 1/2	11-9
Setup and Hold Time Fixing: fix_eco_timing 2/2	11-10
Setup/Hold fixing in Path-Based Analysis	11-11
Leakage Recovery : fix_eco_power -pattern_priority	11-12
Reporting Vt Cell Usage Example.....	11-13
Handling Unconstrained Cells	11-14
Recommended ECO Guidance Command Flow : 1/2	11-15
Recommended ECO Guidance Command Flow : 2/2	11-16

Table of Contents

Creating the Change List: write_changes	11-17
Introducing Physically Aware ECO to Improve Fix Rates.....	11-18
Using Physically Aware ECO.....	11-19
open_site and occupied_site Physical Modes	11-20
Reading in Physical Constraint Data	11-21
Example Physically Aware ECO in PT (1/2)	11-22
Example Physically Aware ECO in PT (2/2)	11-23
Review of Unit Objectives.....	11-24
Appendix	11-25
Limiting changes during ECO	11-26
Restricting ECO fix with Path Control	11-27
Example Manual ECO – Size a Cell.....	11-28
Example Manual ECO – Insert a Buffer at a Driver.....	11-29
Example ECO – Insert a Buffer at a Load	11-30
Manually Fixing a Select Path: Setup	11-31
Manually Fixing a Select Path: Hold.....	11-32
What-If Analysis for Crosstalk	11-33
Other Manual Netlist Editing Commands.....	11-34
ECO Fixing Flow.....	11-35
Optional Tailoring: Library Names	11-36
Optional Tailoring: Buffer Names	11-37
Optional Guidance: Sizing Cells.....	11-38
Using User Synthesis Library Attributes	11-39
User Interface for Leakage Recovery	11-40

Unit 12: Large Data: DMSA and Hyperscale Analysis

Agenda	12-1
Unit Objectives	12-2
Timing Analysis Flow in PrimeTime	12-3
What Is Distributed Multi-Scenario Analysis (DMSA)?.....	12-4
Invoking a DMSA Master Process	12-5
Setting Up DMSA.....	12-6
Defining Scenario Data.....	12-7
Scenario Output Logs	12-8
remote_execute	12-9
Merged report_timing	12-10
Merged report_constraint.....	12-11
Merged report_constraint.....	12-12
DMSA Resources on SolvNET	12-13
Hyperscale Analysis Introduction.....	12-14
Hyperscale Top Level Context	12-15
Hyperscale Block Model.....	12-16
Transitioning to Hyperscale Analysis From	12-17
Flow #1: Flat Context flow -- Context Characterization	12-18

Table of Contents

Flow #1: Context Reporting.....	12-19
Flow #1: Block Level Analysis with Characterized Context.....	12-20
Flow #2: Bottom Up Flow with Hyperscale Models	12-21
Flow #2: Clock Mapping for Hyperscale Analysis.....	12-22
Example Clock Mapping Issue : Full Chip Analysis is OK	12-23
Example Clock Mapping Issue : Clock Mapping Failed!.....	12-24
Example Clock Mapping Issue : Unconstrained Block Path!.....	12-25
Example Clock Mapping Issue : Solution	12-26
Example Clock Mapping Issue : Block Level Analysis is OK.....	12-27
Flow #3: Top Down Flow with Constraint Extraction	12-28
Flow #3 Step-1: Run HyperScale Constraint Extractor	12-29
Flow #3 Step-2: Initial Block Level Analysis	12-30
Flow #3 Step-3: Top Level Analysis	12-31
Flow #3 Step-4: Finalize Block Level Analysis	12-32
Timing Report in HyperScale with Top Level Context.....	12-33
Hyperscale Training on SolvNET.....	12-34
Review of Unit Objectives.....	12-35
Appendix.....	12-36
Terminology : Master	12-37
Terminology : Slave.....	12-38
Terminology : Scenario.....	12-39
Terminology : Image.....	12-40
License Management	12-41
Machine Management.....	12-42
Setting Up DMSA: Configuring Licenses	12-43
Setting Up DMSA: Configuring Remote Processes	12-44
Setting Up DMSA: Starting the Farm.....	12-45
Setting Up DMSA: Defining Scenarios	12-46
Setting Up DMSA: Defining Scenarios Using Scripts	12-47
Setting Up DMSA: Defining Scenarios Using Scripts	12-48
Setting Up DMSA: Defining Scenarios Using Scripts	12-49
Setting Up DMSA: Defining Scenarios Using Scripts	12-50
Setting Up DMSA: Defining Scenarios Using Scripts	12-51
Setting Up DMSA: Defining Scenarios Using Sessions.....	12-52
DMSA Flow for ECO	12-53
HyperScale-driven ECO: Top Level ECO w/ Top Logic Only	12-54
HyperScale-driven ECO: Top Level ECO w/ Block Boundary	12-55
HyperScale-driven ECO : Context Driven Block ECO	12-56

Unit 13: Conclusion

Agenda	13-1
Workshop Goal	13-2
Timing Analysis Flow in PrimeTime	13-3
Following Day-1 units, You Can Now	13-4

Table of Contents

Following Day-2 units, You Can Now	13-5
Following Day-3 units, You Can Now	13-6
How to Download Lab Files (1/2)	13-7
How to Download Lab Files (2/2)	13-8

Unit CS: Customer Support

Synopsys Support Resources	CS-2
SolvNet Online Support.....	CS-3
SolvNet Registration.....	CS-4
Support Center	CS-5
Other Technical Sources	CS-6
Summary: Getting Support	CS-7

PrimeTime: Static Timing and SI Analysis

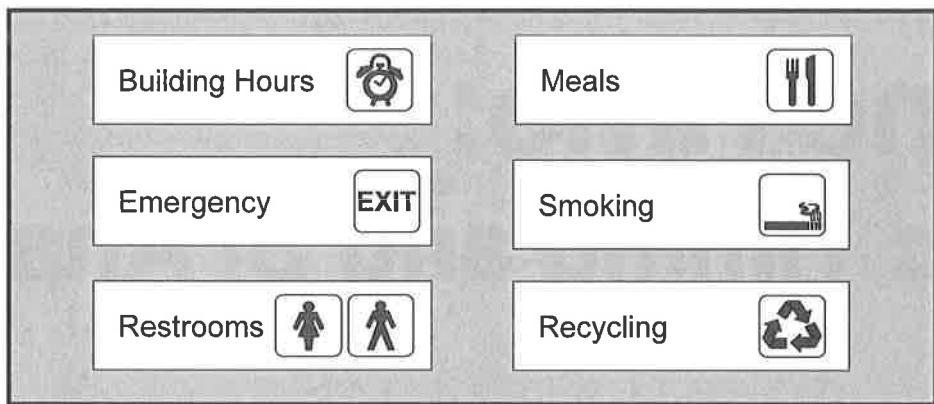
2018.06



Synopsys Customer Education Services
©2018 Synopsys, Inc. All Rights Reserved

Synopsys 104-034-SSG-015

Facilities



Please turn off or silence your cell phone

i-2

Workshop Goal

PRIMETIME

- Use *PrimeTime* to perform Static Timing Analysis (STA) and Signal Integrity (SI) analysis on a block or chip level design netlist by
 - Reading in design, library, parasitic data, and constraints
 - Debugging STA constraints before generating reports
 - Creating and Restoring saved sessions
 - Considering Signal Integrity (SI) impact due to Coupling Capacitances
 - Generating and Interpreting Reports for Summary, Timing and Noise
 - Accounting for On Chip [delay] Variations (OCV) using POCV technique
 - Using Path Based Analysis (PBA)
 - Enabling Advanced Waveform Propagation (AWP) for signoff accuracy

i- 3

POCV: Parametric On Chip Variation

Workshop Target Audience

ASIC Timing Signoff engineers who will be using
PrimeTime to perform Static Timing Analysis



i-4

Workshop Prerequisite Knowledge

- Prior experience with *PrimeTime* is not needed
- An understanding of basic digital ASIC design concepts is assumed, including:
 - Combinational and sequential logic functionality
 - Setup and hold timing
- The ability to work in a *Linux* environment, using a text editor such as *emacs*, *vi*, *pine*, is required for labs

i-5

Introductions

- Name
- Company
- Job Responsibilities
- Experience with Synopsys Tools or Flow
 - *PrimeTime, Design Compiler, IC Compiler (II), Star RC and/or NanoTime*
- Main Goal(s) and Expectations for this Course

i-6

PrimeTime is our gate level Static Timing Analysis (STA) tool

Design Compiler is our RTL Synthesis tool

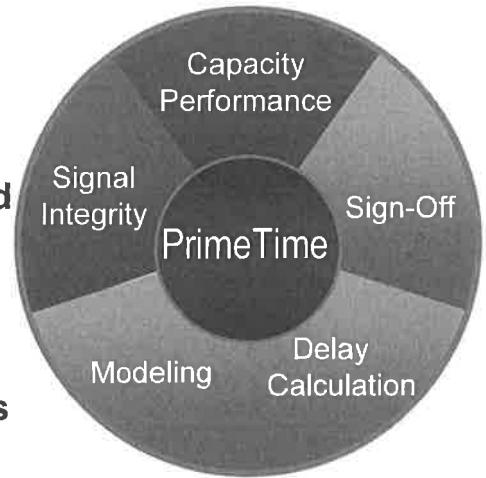
IC Compiler and *IC Compiler II* are our Place and Route (layout) tools

Star RC is our parasitic (RC) extraction tool

NanoTime is our transistor Level STA tool

What is PrimeTime?

- Performs Static Timing Analysis (STA)
- Has accurate Delay Calculator using SPEF
- Provides signoff accuracy using techniques as PBA, POCV and AWP
- Includes signal integrity analysis for Timing and Noise using CCS library models
- Can be used to create timing models such as QTM and ETM
- Enables Multi mode Multi corner DMSA analysis
- 50M+ Gate Capacity and Performance using (distributed) Hyperscale analysis capability

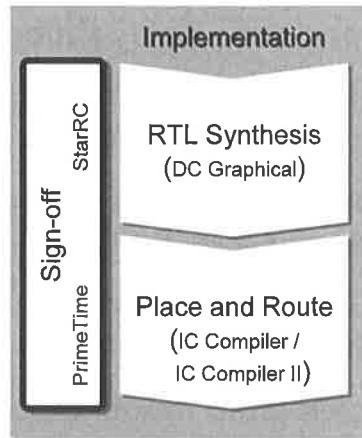


i-7

SPEF:	Standard Parasitic Exchange Format (IEEE Standard)
PBA:	Path Based Analysis
POCV:	Parametric On Chip Variation analysis
AWP:	Advanced Waveform Propagation
CCS:	Composite Current Source model for delay and noise (Liberty Standard)
QTM:	Quick Timing Model (a placeholder model when a block netlist is not ready)
ETM:	Extracted Timing Model (a netlist converted into a single library cell – shields details of an IP – can be used during synthesis and P&R as well as for STA)
DMSA:	Distributed Multi Scenario Analysis

PrimeTime in the Implementation Flow

- **Industry-standard STA and signoff**
- **Run at the gate level after design transformations**
 - Before handing off to manufacturing
 - Generally, between tools
 - Sometimes, between transformations within a tool



i- 8

What is Static Timing Analysis (STA)?

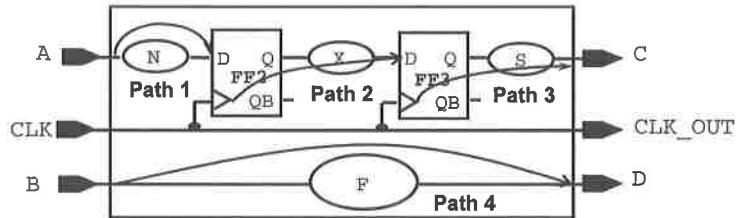
- **Static timing analysis:**

- **Verifies Timing**
 - ◆ verifies timing between synchronous clocks
 - ◆ does not verify functionality
- **Is Exhaustive**
 - ◆ uses formal, mathematical techniques instead of vectors
 - ◆ does not use dynamic logic simulation
- **Is Fast**
 - ◆ significantly faster than gate level simulation
 - ◆ orders of magnitude Faster than Tx level (spice) simulation

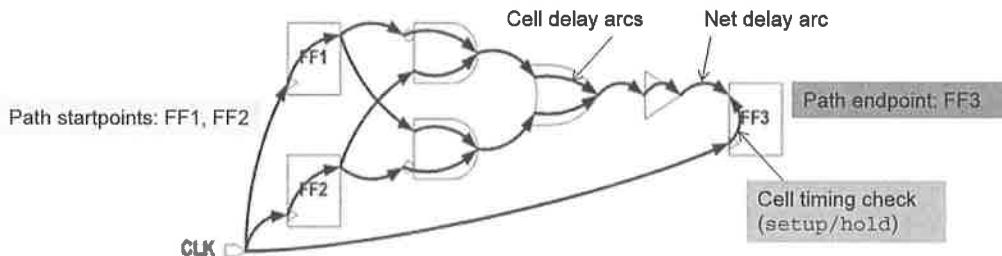
i-9

STA is Path Based

- STA identifies timing paths within design for analysis



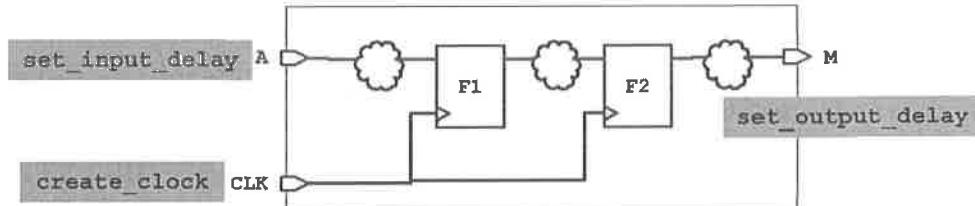
- Timing arcs within each path are calculated



i-10

STA is Constraint Driven

- STA in PT is constraint driven



- PT does not report a path, by default, that is not constrained for timing
- Incomplete or inaccurate constraints will lead to incorrect analysis and wasted runtime
- Applying, Interpreting, and Debugging constraints are covered in this workshop

i-11

Signoff Considerations

- **Analysis must incorporate crosstalk impact due to coupling capacitances between nets that can result in timing or functional failures**
 - PrimeTime-SI checks for above impact on the delay and noise
- **It's necessary to model random process variations accurately and efficiently**
 - POCV models timing as true statistical distribution (instead of min:max delays)
- **In advanced process nodes, design waveforms can significantly deviate from characterization waveforms**
 - AWP with CCS libraries can restore timing accuracy due to waveform distortions
- **Generating ECO guidance would be required for timing closure**
 - Physically Aware ECO considers physical implementation achievability for improved correlation
- **Design timing must be analyzed in multiple modes and PVT corners**
 - DMSA provides efficient unified analysis of multiple PrimeTime scenarios

i- 12

SI:	Signal Integrity
POCV :	Parametric On Chip Variation
AWP:	Advanced Waveform Propagation
ECO:	Engineering Change Order
DMSA:	Distributed Multi Scenario Analysis

Agenda : Flow and Methodology

DAY
1

i Introduction to STA in PrimeTime

1 STA Concepts and Flow in PrimeTime

2 Methodology: Qualifying Constraints

3 Methodology: Generating Reports

i- 13

Agenda : Best Practices

DAY
2

4 Constraining Multiple Clocks



5 Additional Checks and Constraints



6 Best Practices Debugging Reports

7 Signoff: Path Based Analysis (PBA)



i-14

Agenda : Signoff Considerations (Continued)

DAY
3

- | | | |
|----|--|---|
| 8 | Signal Integrity: Crosstalk Delay Analysis |  |
| 9 | Signal Integrity: Crosstalk Noise Analysis |  |
| 10 | Correlation: POCV and AWP Analysis |  |
| 11 | Timing Closure: ECO/What If Analysis |  |
| 12 | Large Data: DMSA and Hyperscale Analysis |  |
| 13 | Conclusion |  |

i- 15

Icons Used in this Workshop



Recommendation



Caution.
Non-intuitive tool behavior



Question



Avoid



Lab Exercise



Group Exercise

i- 16

Agenda

DAY
1

- i Introduction to STA in PrimeTime
- 1 STA Concepts and Flow in PrimeTime 
- 2 Methodology: Qualifying Constraints 
- 3 Methodology: Generating Reports 

Unit Objectives

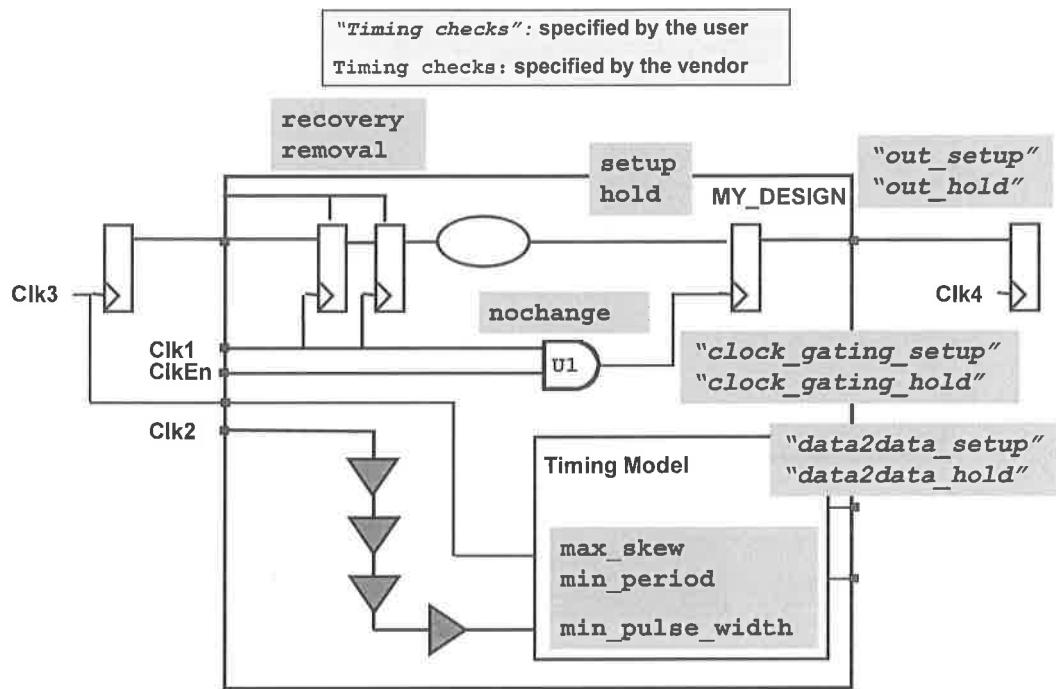


After completing this unit, you should be able to:

- State the purpose of using the following reports:
 - report_analysis_coverage
 - report_global_timing
 - report_timing
- List the steps of Timing Analysis flow in PrimeTime
- Verify what inputs have been read into PrimeTime
- State the purposes of *setup file* and a *run script*
- Enable monitoring runtime and memory used

1-2

PrimeTime Verifies a Number of Timing Checks



1-3

There are additional checks, which can be specified in the library, for example:

clock separation: This is a constraint for master/slave latches for a minimum required clock separation between two clocks to avoid the latch from becoming transparent.

nonsequential: This is similar to a data to data setup and hold check between two data pins.

A **constraint check** is one specified by the user as a constraint (e.g. set_output_delay).

A **library check** is one specified by the vendor in the library for that specific leaf cell (e.g. setup/hold).

A **library check is also** created when writing out an extracted model: nochange is clock gating setup and hold checks modeled as nonchange arcs on the extracted model.

Summary Report of Timing Checks verified

```
pt_shell> report_analysis_coverage
```

Type of Check	Total	Met	Violated	Untested
setup	6724	5366 (80%)	0 (0%)	1358 (20%)
hold	6732	5366 (80%)	0 (0%)	1366 (20%)
recovery	362	302 (83%)	0 (0%)	60 (17%)
removal	354	302 (85%)	0 (0%)	52 (15%)
min_pulse_width	4672	4310 (92%)	0 (0%)	362 (8%)
clock_gating_setup	65	65 (100%)	0 (0%)	0 (0%)
clock_gating_hold	65	65 (100%)	0 (0%)	0 (0%)
out_setup	138	138 (100%)	0 (0%)	0 (0%)
out_hold	138	74 (54%)	64 (46%)	0 (0%)
All Checks	19250	15988 (84%)	64 (0%)	3198 (16%)

There are 64 violations of
type out_hold in the design

1-4

STA is Path Based: Start and End Points of Timing Paths

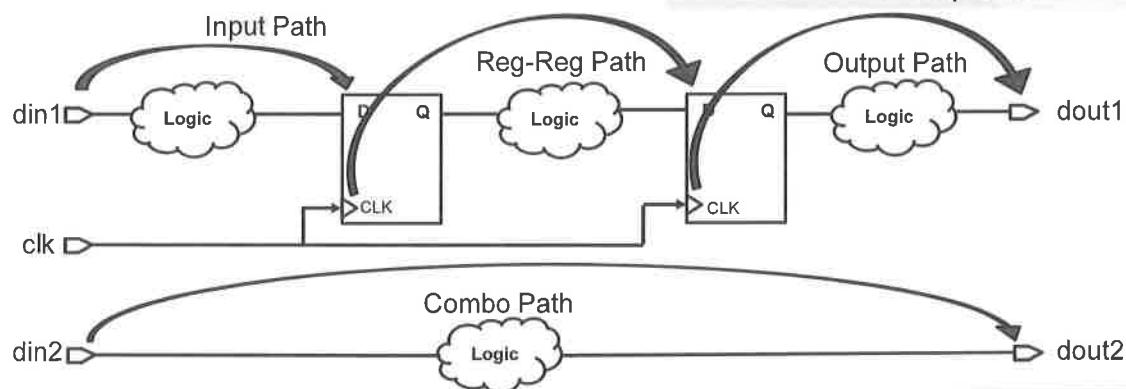
- **Timing path start points include:**

- Clock pins of registers
- Input ports

- **Timing path end points include:**

- All input pins of registers except clock pins
- Output ports

Path Type	Start Point	End Point
Input Path	Input Port	Register Non Clock pin
Reg-Reg Path	Register Clock pin	Register Non Clock pin
Output Path	Register Clock pin	Output port
Combo Path	Input Port	Output Port



1-5

Summary Report of Violating Path Types

Are violations internal or in the I/O?

```
pt_shell> report_global_timing
```

Four categories

Setup violations -- All groups					
	Total	reg->reg	reg->out	in->reg	in->out
WNS	-1.457	-1.457	-1.157	0.000	0.000
TNS	-15.859	-12.040	-3.819	0.000	0.000
NUM	28	19	9	0	0

Hold violations -- All groups

	Total	reg->reg	reg->out	in->reg	in->out
WNS	-0.437	-0.429	-0.370	-0.236	-0.437
TNS	-43.266	-31.155	-3.636	-2.577	-5.898
NUM	196	148	14	18	16

Worst,
Total, and
Number

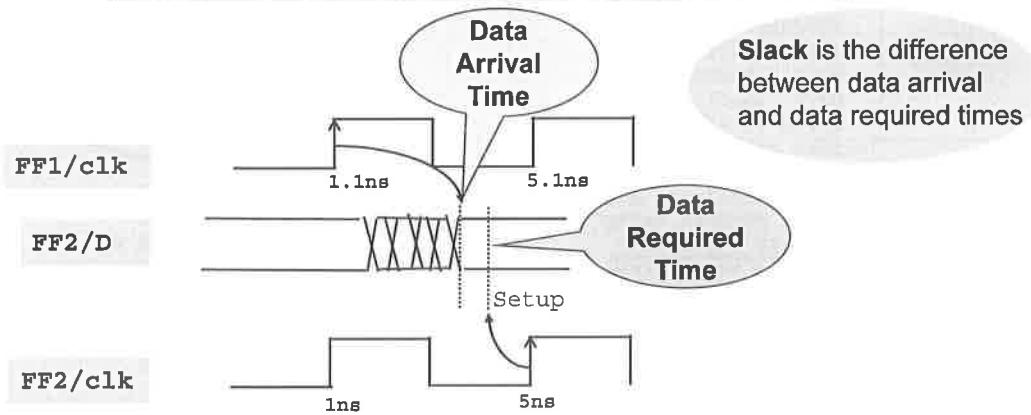
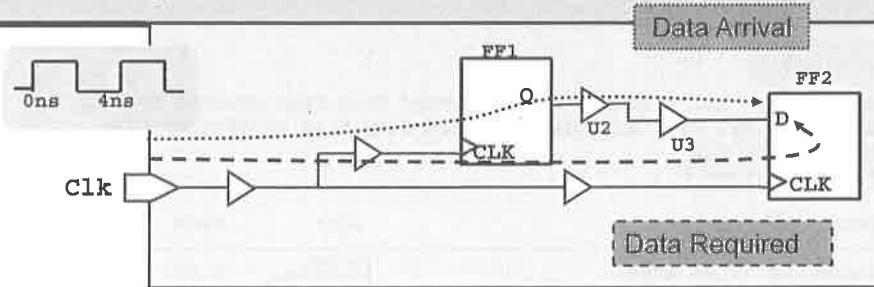
1-6

WNS: Worst Negative Slack (Largest violation)

TNS: Total Negative Slack (Sum of all the violations)

NUM: Number of violations

Setup Timing (Max Delay) Analysis



1-7

In PrimeTime, **positive slack** means the timing requirements are met; **negative slack** means there is a violation.

Timing Report for Setup (Max Delay Analysis)

report_timing

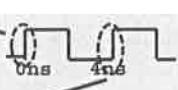
Header

Startpoint: FF1 (rising edge-triggered flip-flop clocked by Clk)
 Endpoint: FF2 (rising edge-triggered flip-flop clocked by Clk)
 Path Group: Clk
 Path Type: max

Data arrival

Point
 clock Clk (rise edge)
 clock network delay (propagated)
 FF1/CLK (fdef1a15)
 FF1/Q (fdef1a15)
 U2/Y (buf1a27)
 U3/Y (buf1a27)
 FF2/D (fdef1a15)
 ... data arrival time

Incr	Path
0.00	0.00
1.10 &	1.10
0.00	1.10 r
0.50 &	1.60 r
0.11 &	1.71 r
0.11 &	1.82 r
0.05 &	1.87 r
	1.87



Data required

clock Clk (rise edge)
 clock network delay (propagated)
 FF2/CLK (fdef1a15)
 library setup time
 ... data required time

Incr	Path
4.00	4.00
1.00 &	5.00
	5.00 r
-0.21	4.79
	4.79

Slack

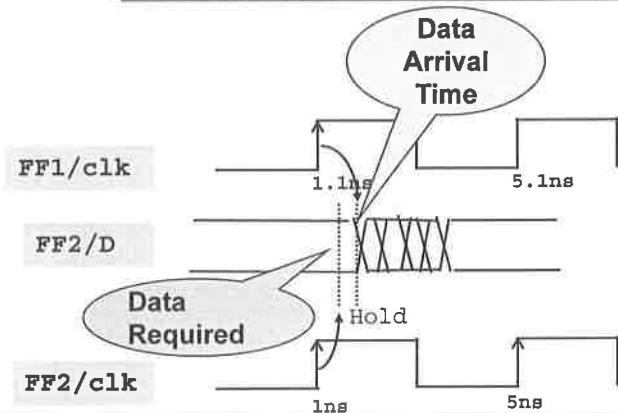
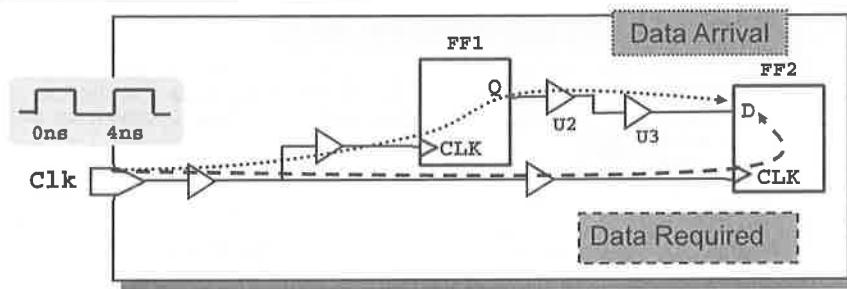
data required time
 data arrival time
 ... slack (MET)

Incr	Path
	4.79
	-1.87
	2.92

1-8

report_timing generates a timing report, which is a setup check report, by default (the -delay max option is implicit)

Hold Timing (Min Delay) Analysis



Slack is the difference between data arrival and required.

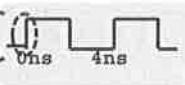
1-9

Timing Report for Hold (Min Delay Analysis)

report_timing -delay min

Startpoint: FF1 (rising edge-triggered flip-flop clocked by Clk)
 Endpoint: FF2 (rising edge-triggered flip-flop clocked by Clk)
 Path Group: Clk
 Path Type: min

Point	Incr	Path
<hr/>		
clock Clk (rise edge)	0.00	0.00
clock network delay (propagated)	1.10 &	1.10
FF1/CLK (fdef1a15)	0.00	1.10 r
FF1/Q (fdef1a15)	0.40 &	1.50 f
U2/Y (buf1a27)	0.05 &	1.55 f
U3/Y (buf1a27)	0.05 &	1.60 f
FF2/D (fdef1a15)	0.01 &	1.61 f
data arrival time		1.61
<hr/>		
clock Clk (rise edge)	0.00	0.00
clock network delay (propagated)	1.00 &	1.00
FF2/CLK (fdef1a15)		1.00 r
library hold time	0.10	1.10
data required time		1.10
<hr/>		
data required time		1.10
data arrival time		-1.61
<hr/>		
slack (MET)		0.51

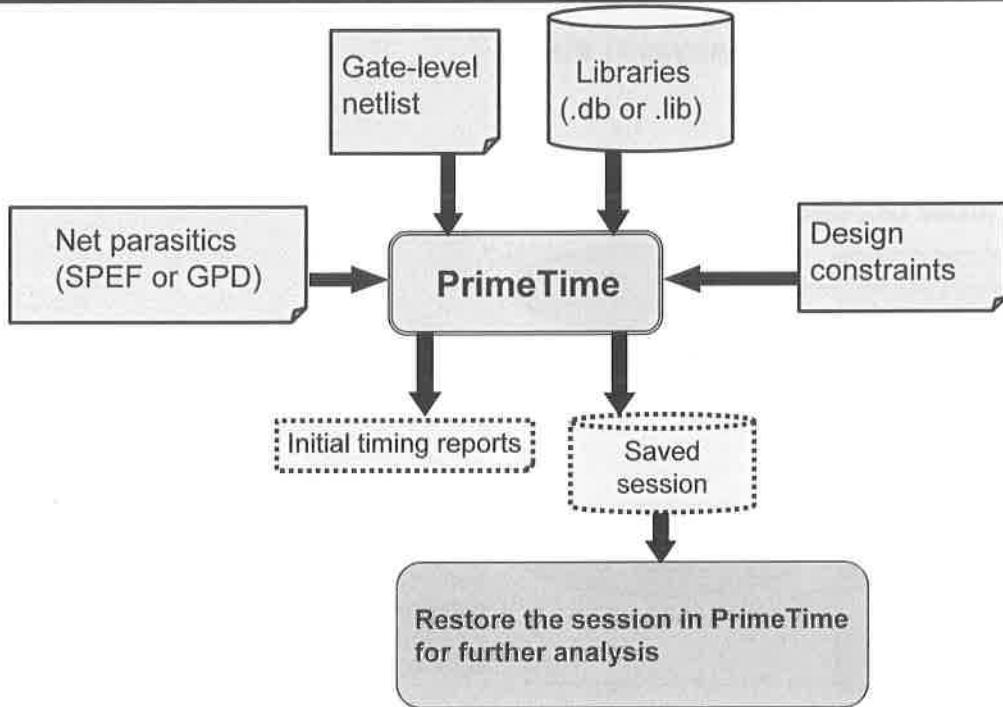


1-10

There are several clues that this is a hold time report.

Clauses that this is a hold report: path type min, library hold time, clock edges (same), data arrival is AFTER data require and the slack is positive

PrimeTime Inputs and Outputs



1-11

SPEF: IEEE Standard Parasitic Exchange Format (ASCII)

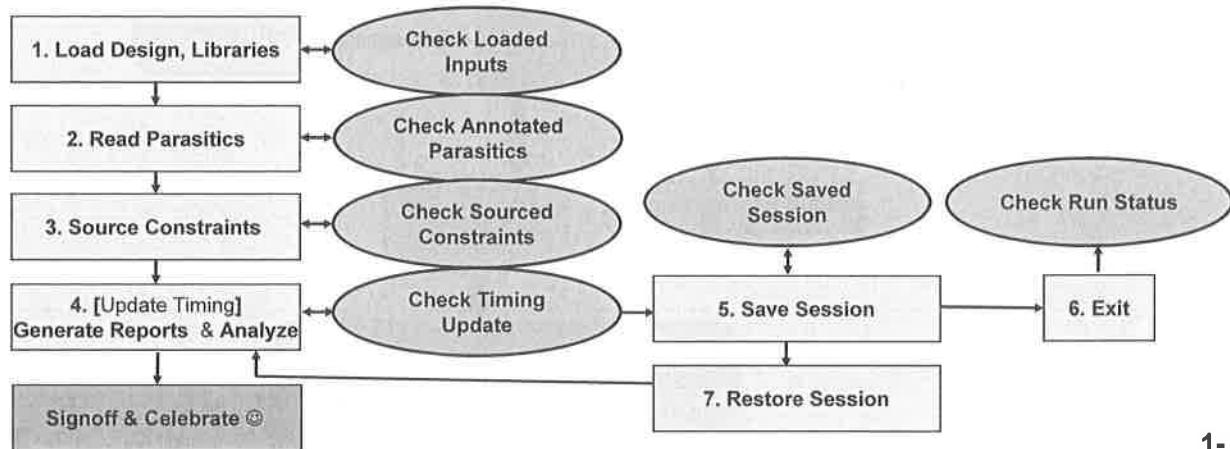
GPD: Synopsys Galaxy Parasitic Database (Binary)

Timing Analysis Flow in PrimeTime

- STA flow is divided into several steps

- Steps 1-3 read data and
- Analysis begins at Step-4

- The STA flow is repeated until signoff is achieved



1-12

STA: Static Timing Analysis

Step 1a: Load Design and Check

```
# Read gate level netlist; See also help read*
read_verilog orca_routed.v.gz
# Set the current design to be Top level design
current_design ORCA

# Check what is Current design
pt_shell> current_design
{"ORCA"}

# What Designs are loaded in memory?
pt_shell> get_designs *
{"ORCA"}


# Source File name of current_design
pt_shell> list_designs
Design Registry:
*L ORCA          /<path to>/orca_routed.v.gz:ORCA
```

1-13

```
pt_shell> list_designs
Design Registry:
*L ORCA          /u/ref/design_data/orca_routed.v.gz:ORCA
  * - Indicates that the design is the current design
  L - Indicates that the design is linked
  N - Indicates that the design is not in memory
  l - Indicates that the design is partially linked
```

PrimeTime User Guide recommends setting the two *_path variables before reading the design.

1. Specify the directories in which PrimeTime searches for designs, logic libraries, and other design data such as timing models. To do this, set the `search_path` variable. For example:

```
pt_shell> set_app_var search_path ". /abc/design /abc/libs"
```

PrimeTime searches the directories in the order that you specify.

2. Specify the libraries in which PrimeTime finds elements in the design hierarchy by setting the `link_path` variable. For example:

```
pt_shell> set_app_var link_path "* STDLIB.db"
```

The variable can contain an asterisk (*), library names, and file names. The asterisk instructs PrimeTime to search for a design in memory. PrimeTime searches libraries in the order that you specify. The main library is the first library in the link path.

Step 1b: Load Libraries and Check

```
set search_path ". . ./ref/libs . ./ref/design"
set link_path "* sc_max.db io_max.db"
link_design      ;# Load Libraries and Resolve References
```

```
# Are the Libraries loaded?
list_libs OR list_libraries OR get_libs
Library Registry:
* cb13fs120_tsmc_max    . ./ref/libs/sc_max.db:cb13fs120_tsmc_max
  cb13io320_tsmc_max    . ./ref/libs/io_max.db:cb13io320_tsmc_max

# Are Link Library DB Files followed by a "*"
pt_shell> printvar link_path
link_path      = "* sc_max.db io_max.db"

# Does search_path list start with a ". . ."
pt_shell> printvar search_path
search_path   = ". . ./ref/libs . ./ref/design"
```

1-14

```
pt_shell> link_design
Linking design ORCA...
Information: removing 24 unneeded designs..... (Lnk-034)
Information: 439 (81.75%) library cells are unused in library
cb13fs120_tsmc_max..... (Lnk-045)
Information: 82 (95.35%) library cells are unused in library
cb13io320_tsmc_max..... (Lnk-045)
Information: total 521 library cells are unused (LNK-046)
Design 'ORCA' was successfully linked.
Information: there are 20403 leaf cells, ports, hiers and 24967 nets in the
design (lnk-047)
1
pt_shell> list_libraries
Library Registry:
* cb13fs120_tsmc_max    . ./ref/libs/sc_max.db:cb13fs120_tsmc_max
  cb13io320_tsmc_max    . ./ref/libs/io_max.db:cb13io320_tsmc_max
* refers to the Main Library name; Library from the first DB file listed in
link_path variable
```

There are a few other commands as follows for checking. Refer to the man pages for more details

```
printvar link_create_black_boxes
printvar link_allow_design_mismatch
list_libs  [-only_used]
report_lib <Lib_Name>
```

Step 2a: Read Parasitics

```
# SPEF file  
read_parasitics -format SPEF flat.spef  
  
# GPD directory  
read_parasitics -format GPD GPD_dir  
  
# Hierarchical SPEF  
read_parasitics -path I_BlkB I_BlkB.spef.gz  
read_parasitics -path {U1 U2 U3} BlkC.spef.gz ;# MIM  
read_parasitics TOP.spef.gz
```

It's recommended that parasitics are read in using one of the two formats:

- Galaxy Parasitic Database (GPD)
- Standard Parasitic Exchange Format (SPEF)

1-15

GPD: Galaxy Parasitic Database (directory – Synopsys format - binary data)
SPEF: Standard Parasitic Exchange Format (IEEE format - ASCII data)
MIM: Multiply Instantiated Module

read_parasitics supports the following formats:

- Galaxy Parasitic Database (GPD)
- Standard Parasitic Exchange Format (SPEF)
- Detailed Standard Parasitic Format (DSPF)
- Reduced Standard Parasitic Format (RSPF)
- Milkyway (PARA)

Step 2b: Check Parasitic Annotation

Net Type	Single RC			C-R-C "pi"	Detailed RC	
	Total	Lumped	RC pi	RC network	Not Annotated	
Internal nets						
- Pin to pin nets	22581	0	0	22581	0	
- Driverless nets	24	0	0	0	24	
- Loadless nets	112	0	0	112	0	
Boundary/port nets						
- Pin to pin nets	71	0	0	0	71	
- Driverless nets	0	0	0	0	0	
- Loadless nets	0	0	0	0	0	
	22788	0	0	22693	95	

report annotated parasitics -list not annotated
1. I_ORCA_TOP/pclk (driver: pin I_ORCA_TOP/pclk) 2. I_ORCA_TOP/test_mem_clk (driver: pin I_ORCA_TOP/test_mem_clk)

1-16

1. read_parasitics command automatically invokes the report_annotated_parasitics that shows the report
2. Any unannotated pin-to-pin nets should be investigated
Driverless or loadless nets generally result from unused pins such as a flop qbar pin – it is alright for these to be unannotated.
3. report_annotated_parasitics –help; then choose from the options. Final answer:
report_annotated_parasitics –list_not_annotated -max_nets 100 (by default, it just lists the first ten unannotated nets).

Additionally, examine any PARA-, RC- Warnings during parasitic reading.

Step 3: Source Constraints and check for correctness

```
# SDC (.sdc) file  
read_sdc -echo $constraint_file  
  
# Tcl Constraints (Not an .sdc file)  
source -echo $constraint_file
```

Source constraints

```
# Are Constraints complete?  
check_timing -verbose  
  
# Examine Clock and Port constraints  
report_clock -skew -attribute  
report_port -verbose  
  
# Examine Design Constraints and Exceptions  
report_design  
report_exceptions -ignored  
report_case_analysis
```

Check for constraint completeness and correctness

1-17

SDC: Synopsys Design Constraints (file)

Tcl: Tool Command Language

Confirming Constraint Completeness

```
pt_shell> check_timing -verbose
Information: Checking 'no_input_delay'.
Information: Checking 'no_driving_cell'.
Information: Checking 'unconstrained_endpoints'.
Warning: There are 3 endpoints which are not constrained for maximum delay.
Endpoint
-----
| q_out      |
| DATR_reg/D |
| DATF_reg/D |
Information: Checking 'unexpandable_clocks'.
Information: Checking 'latch_fanout'.
Information: Checking 'no_clock'.
Information: Checking 'partial_input_delay'.
Information: Checking 'generic'.
Information: Checking 'loops'.
Information: Checking 'generated_clocks'.
Information: Checking 'pulse_clock_non_pulse_clock_merge'.
Information: Checking 'pll_configuration'.
0
```

1-18

Checking for Ignored Timing Exceptions

■ User-specified exceptions to PT single-cycle synchronous STA are

- False paths
- Multi-cycle paths
- set_max_delay and set_min_delay

■ Incorrectly-specified exceptions are ignored

```
pt_shell> report_exceptions -ignored
Reasons :   f  invalid start points
             t  invalid end points
             p  non-existent paths
             o  overridden paths

From      Through      To      Setup      Hold      Ignored
I_ORCA_TOP/I_SDRAM_IF/DQ_out_0_reg[0]/Q
*
           I_ORCA_TOP/I_SDRAM_IF/sd_mux_dq_out_0/I0
                           FALSE      FALSE      f
```

Why is this exception ignored?

1-19

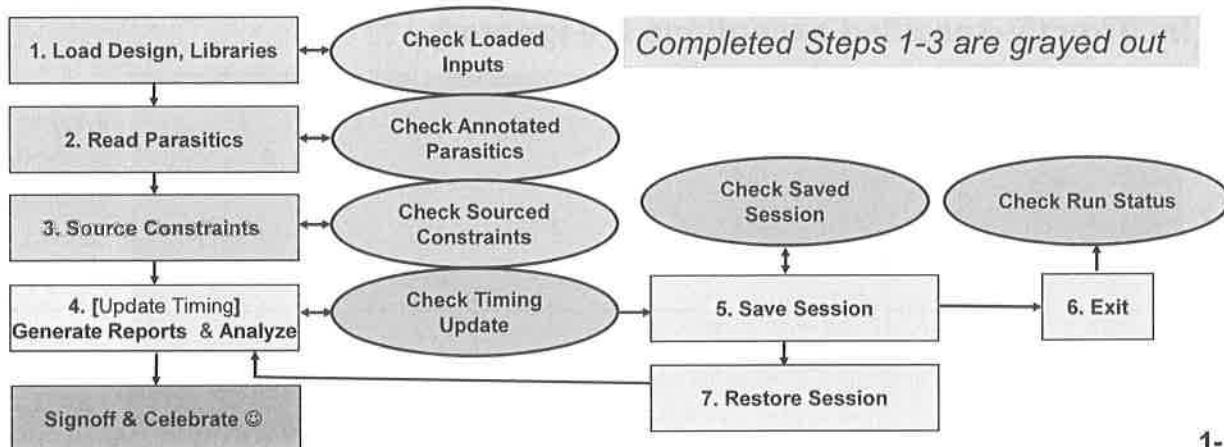
1. Invalid start point (the output of a flop is not a valid start point – valid start points are only ports, clock pin of a flop, and clock objects)

Recalling Timing Analysis Flow in PrimeTime

- STA flow is divided into several steps

- Steps 1-3 read data and
- Analysis begins at Step-4

- The STA flow is repeated until signoff is achieved



1- 20

STA: Static Timing Analysis

Step 4a: Update Timing

- Invoke an explicit Timing Update once [optionally -full]

```
update_timing [-full]
```

- Check Update Timing Results by generating summary reports

```
report_global_timing [-pba]  
report_qor [-pba]  
report_analysis_coverage  
report_constraint -all_violators [-pba]  
report_disable_timing
```

1-21

PBA: Path Based Analysis → This workshop topic is covered in a later module
Running the PBA option before save_session can avoid recalculation after restore_session

Coverage Analysis

report_analysis_coverage



Report : analysis_coverage
Design : ORCA

Use this as a sanity check.
That is, given the mode and corner being tested,
do I expect these number of untested checks.

Type of Check	Total	Met	Violated	Untested
setup	9629	3565 (37%)	23 (0%)	6041 (63%)
hold	9629	3535 (37%)	53 (1%)	6041 (63%)
recovery	1316	1210 (92%)	0 (0%)	106 (8%)
removal	1316	1210 (92%)	0 (0%)	106 (8%)
min_period	20	20 (100%)	0 (0%)	0 (0%)
min_pulse_width	7273	5957 (82%)	0 (0%)	1316 (18%)
clock_gating_setup	33	33 (100%)	0 (0%)	0 (0%)
clock_gating_hold	33	33 (100%)	0 (0%)	0 (0%)
out_setup	75	43 (57%)	32 (43%)	0 (0%)
out_hold	75	75 (100%)	0 (0%)	0 (0%)
All Checks	29399	15681 (53%)	108 (0%)	13610 (46%)

1-22

Step 4b: Generate detailed Reports

■ Generate Detailed Reports

Generate Timing Reports in GBA mode with applicable options

```
report_timing \
    -input_pins -path full_clock_expanded -nets \
    -delay min_max -group $grp -max_paths $max \
    -slack_less $positive_number \
    -exceptions all <other options>
```

Generate Timing Reports in PBA mode

```
report_timing -pba_mode path | exhaustive $recalculated_paths
```

■ Examine generated reports for:

- Timing Violations, Constraints and Exceptions applied

1-23

Step 5: Save Session

■ Save session

```
save_session $session_directory
```

■ Check for save session completeness

```
# Check if session was saved  
unix% ls -al $session_directory  
  
# Check version of PrimeTime used to save session  
unix% more $session_directory/Readme  
  
#Check for the libraries used  
unix% more $session_directory/lib_map
```

```
#Check if a saved session restores  
restore_session $session_directory
```

Can restore with the same
version that was used to
save the session

1-24

Saving the PrimeTime Session

```
./pt_scripts/pt.tcl  
# Run script for ORCA  
.*.  
# Save the session  
save_session [-include_libraries] orca_savesession  
quit
```

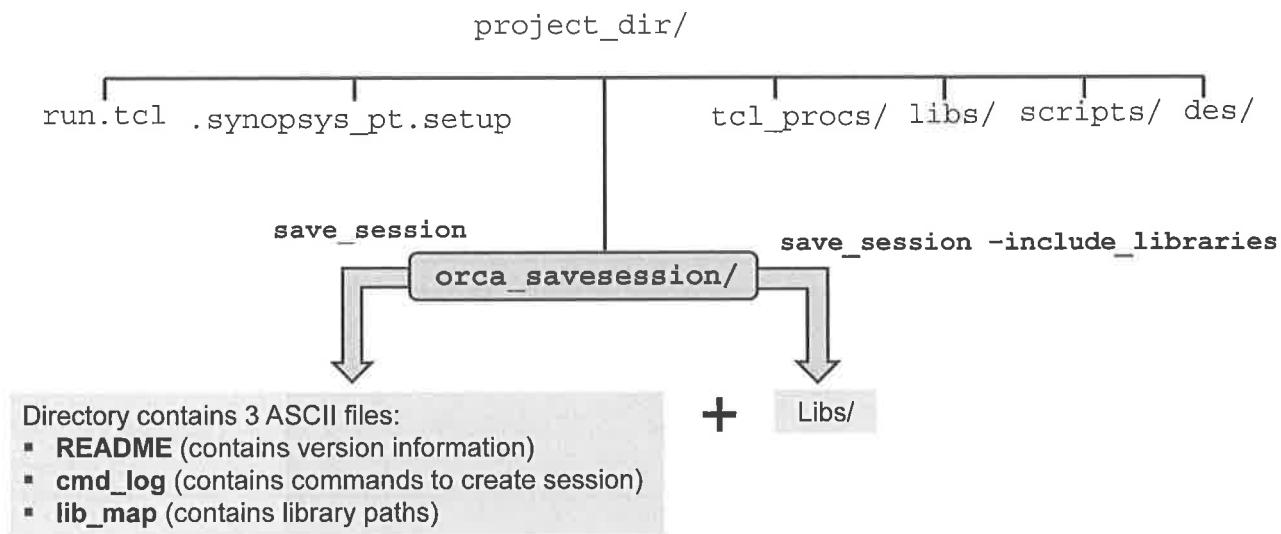
No timing update needed before
debugging a restored session – Fast!

Unix directory name

A saved session contains the following information for restore:

- Linked design and loaded libraries
- Clocks, timing exceptions, and other constraints
- Operating conditions
- Back-annotated SDF delays and parasitics
- Variable settings
- Netlist edits (insert_buffer, size_cell, swap_cell)
- Analysis data
- Cross-coupled delay data and noise data

The saved session directory



1-26

The **README** file contains the PrimeTime version that was used to save the session (The same version is needed to restore the session)

The **cmd_log**: the tool saves the session history into a file called the command log file. This file contains all commands executed during the session and serves as a record of your work. Later, you can repeat the whole session by running the file as a script with the source command.

The **lib_map** file may be modified for the location of library files if not using the `-include_libraries` option (i.e., when the libraries are not copied)

Step 6: Exit

- [Print summary of messages and] Exit primetime

```
pt_shell>
redirect -tee $exit_file {
    print_message_info      ;# Messages during run
    quit                    ;# quit or exit PT
}
```

- Examine \$exit_file for run summary, looking for:

- ◆ Number of Timing Updates
- ◆ CPU time/Memory
- ◆ Number and types of Errors/Warnings

Checking the Run Script Execution

■ Check the quality of the run

```
pt_shell> print_message_info

  Id      Severity      Limit      Occurrences      Suppressed
  -----
CMD-029  Warning       0          1          0
CMD-041  Information   0         53          0
PTSR-014 Information   0          4          0
DES-028  Information   0          1          0

Diagnostics summary: 4 warnings, 58 informationals
```

■ Check for runtime summary

```
pt_shell> quit
Timing updates: 3 (2 implicit, 1 explicit) (1 incremental, 2 full, 0
logical)
Noise updates: 0 (0 implicit, 0 explicit) (0 incremental, 0 full)
Maximum memory usage for this session: 691.07 MB
CPU usage for this session: 31 seconds
Elapsed time for this session: 77 seconds
Diagnostics summary: 4 warnings, 106 informationals
```

1-28

Explicit update : When update_timing or update_timing -full is executed manually

Implicit update : Timing update is automatically invoked when needed.

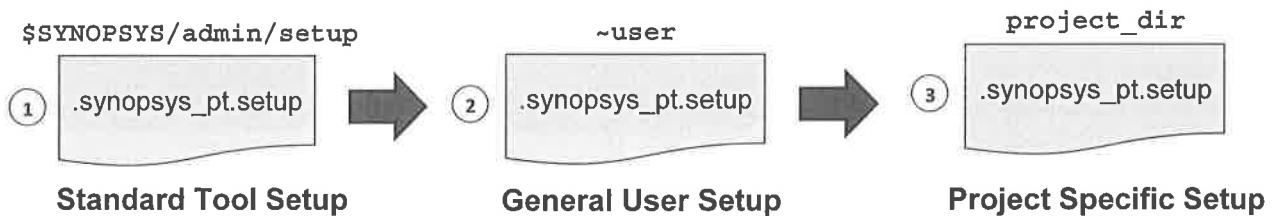
Refer to the Solvnet article for info

(<https://solvnet.synopsys.com/retrieve/011239.html>)

Logical update : Constants and clock identities are propagated, but no delay calculation is performed.

PrimeTime Setup File

- The name of the setup file is `.synopsys_pt.setup`
 - Use `ls -al` to check if this file is present
- Executed automatically when PrimeTime is invoked

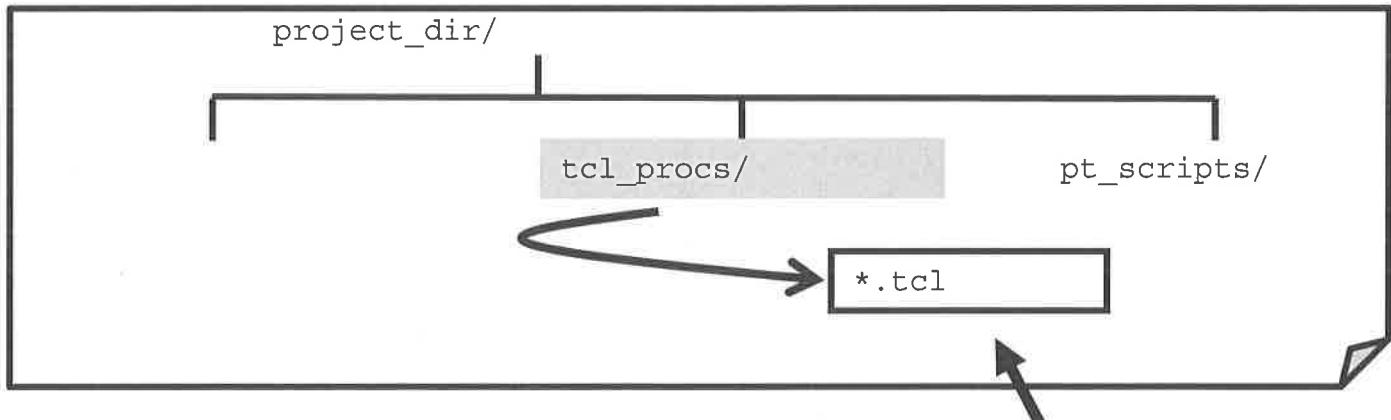


- Use `.synopsys_pt.setup` only for conveniences (Ex: aliases)
Settings affecting STA should be specified in file(s) sourced explicitly
 - See Notes for an example setup file from a user's project directory

1-29

```
# Project directory setup file
#####
# Alias Commands
alias h {history}
alias page_on {set_app_var sh_enable_page_mode true}
alias page_off {set_app_var sh_enable_page_mode false}

##### Other
history keep 200
```



```
foreach each_proc [glob -nocomplain ./tcl_procs/*.tcl] {
    source $each_proc}
```

PrimeTime Run Script : Seed Script Using RMGEN

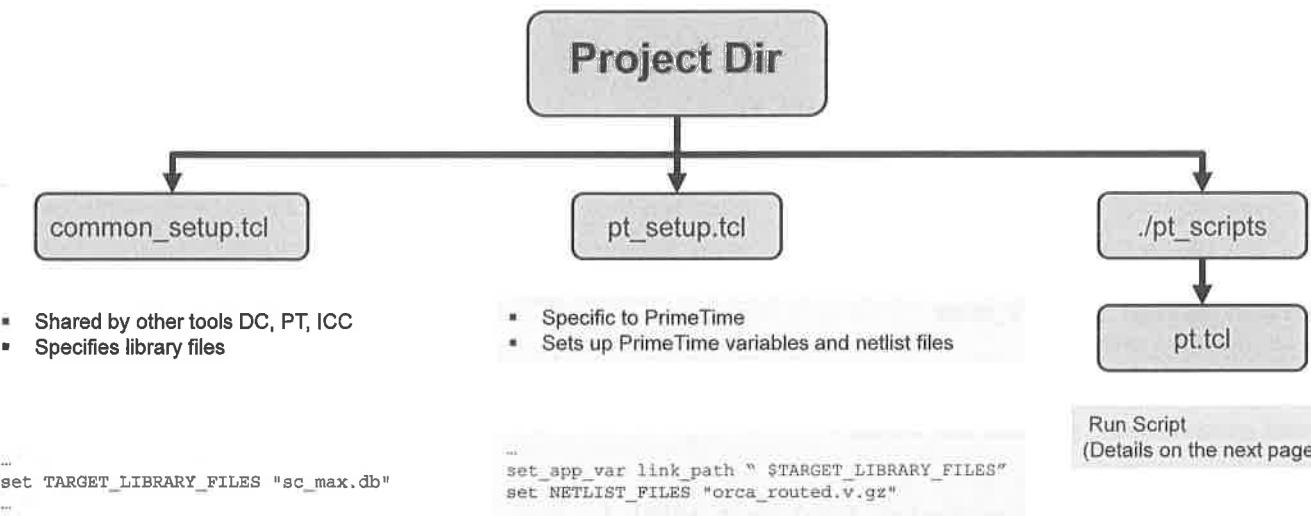
Tool Flow			
Flow	<input checked="" type="radio"/> PT	<input type="radio"/> PTSI	
Back Annotation Mode	<input checked="" type="radio"/> GPD	<input type="radio"/> SPEF	<input type="radio"/> SDF
HyperScale	Input Data		
Design Flow	Link to TetraMax	<input type="radio"/> TRUE	<input checked="" type="radio"/> FALSE
CCS Waveform Propagation	OVC Mode	<input checked="" type="radio"/> OFF	<input type="radio"/> AOCV
Distributed Multi-Scenario Mode	Timing Analysis		
Multivoltage Scaling Mode	CRPR Mode	<input type="radio"/> TRUE	<input checked="" type="radio"/> FALSE
UPF Mode	Derate Mode	<input type="radio"/> TRUE	<input checked="" type="radio"/> FALSE
Clock Gating and Threshold Voltage Group Reporting	Auto Clock Mux	<input type="radio"/> TRUE	<input checked="" type="radio"/> FALSE
Power Analysis	Output Data		
ECO Mode	Save Session	<input type="radio"/> TRUE	<input checked="" type="radio"/> FALSE
Fix Power ECO	Timing Models	<input checked="" type="radio"/> OFF	<input type="radio"/> GENERATION
Fix DRC ECO	Path-Based Analysis	<input type="radio"/> TRUE	<input checked="" type="radio"/> FALSE
Fix Timing ECO	Reports	<input checked="" type="radio"/> STANDARD	<input type="radio"/> ENHANCED
Fix Leakage ECO	Design Environment		
Constraint Analysis	Lynx Compatible	<input checked="" type="radio"/> Lynx	<input type="radio"/> RM / RM+

RMgen provides seed scripts/templates

<https://solvnet.synopsys.com/rmgen/>

Online tutorials and application notes about the RMgen utility are available on <https://solvnet.synopsys.com/rmgen/>: Click on “Product-Specific RM Tutorials”.

Reference Methodology (RMGEN) Generated Files



1-31

PrimeTime Run Script pt.tcl: Generated by RMGEN

```
source ./common_setup.tcl
source ./pt_setup.tcl
}
----- Source common_setup & pt_setup Tcl files

set_app_var sh_source uses_search_path true
set_app_var search_path "$search_path"
}
----- Define Search Path

set_app_var link_path "* $link_path"
read_verilog $NETLIST_FILES
current_design $DESIGN_NAME
link_design -verbose
}
----- Read netlist and link design

read_parasitics $PARASITIC_FILES
----- Back annotate Parasitics

if {[file extension $constraint_file] eq ".sdc"} {
    read_sdc -echo $constraint_file
} else {
    source -echo $constraint_file
}
}
----- Define Search Path
```

1- 32

Script syntax checking and running PrimeTime

■ Use PT's syntax checker: ptprocheck

- ◆ Checks for command syntax without invoking PT

```
Unix% ptprocheck my_script.tcl
Synopsys Tcl Syntax Checker - Version 1.0
...
my_script.tcl:23 (SnpsE-UnkOpt) Unknown
option
'create_clock -preiod'
create_clock -preiod 20 [get_port pclk]
```

■ Source PT Run script in batch mode

```
Unix% pt_shell -f run.tcl | tee -i run.log      (or)
```

```
Unix% pt_shell -f ./pt_scripts/pt.tcl | tee -i run.log
```

1-33

Running PT script interactively:

```
pt_shell> source run.tcl -echo -verbose
[OR]
pt_shell> redirect run.log {source run.tcl -echo -verbose}
```

Redirecting output of command execution:

```
pt_shell> redirect -tee check_timing.rpt {
            check_timing
            report_analysis_coverage }
```

Runtime and Memory Profiling for Monitoring Performance

Identify runtime and memory bottlenecks in your Tcl script

```
#Run script  
start_profile -output my_profile_dir  
#body of run script  
stop_profile / exit / quit
```



- tcl_profile_summary_latest.html
- tcl_profile_summary_latest.txt
- tcl_profile_sorted_by_cpu_time*.txt
- tcl_profile_sorted_by_delta_mem_*.txt
- tcl_profile_sorted_by_elapsed_time_*.txt
- tcl_profile_sorted_by_num_calls_*.txt
- tcl_stopwatch_*.txt

```
> more tcl_profile_summary_latest.txt
```

Reports:

Tcl Profiling. Text output sorted by:

CPU time : tcl_profile_sorted_by_cpu_time_25166.txt

Elapsed time : tcl_profile_sorted_by_elapsed_time_25166.txt

Delta Memory : tcl_profile_sorted_by_delta_mem_25166.txt

Number of calls : tcl_profile_sorted_by_num_calls_25166.txt

Tcl Profiling, Self-Only. Text output, sorted by:

CPU time : tcl_profile_self_only_sorted_by_cpu_time_25166.txt

Elapsed time : tcl_profile_self_only_sorted_by_elapsed_time_25166.txt

Delta Memory : tcl_profile_self_only_sorted_by_delta_mem_25166.txt

Number of calls : tcl_profile_self_only_sorted_by_num_calls_25166.txt

Stopwatch report (text): tcl_stopwatch_25166.txt

```
> more tcl_profile_sorted_by_cpu_time_25166.txt
```

Procedure/Command	Calls	Elapsed	CPU	Memory	Delta
<total>	1	22	14	139	
report_global_timing		1	9	9	96
link_design	1	3	2	24	
define_scaling_lib_group	2	1	1	0	
read_verilog	1	0	0	18	
read_sdc	1	0	0	0	

1-34

Lab 1: Timing Analysis Flow



45 minutes

Validate an existing PrimeTime session

Execute recommended STA flow

Analyze Setup and Hold Reports

1- 35

Appendix

Useful Commands for Lab when in pt_shell

1-36

Useful Commands for Lab when in pt_shell 1/2

- **Use history commands**

- **You do not have to type entire command/option names:**

- Use the <Tab> key to expand
- report_tim <Tab> -inp <Tab>
- report_ana -stat

- **Cut and paste long hierarchical names**

- **Within PrimeTime:**

```
pt_shell> help -verbose report_timing  
pt_shell> man link_path  
pt_shell> aa report ;# aa is a "grep" like utility
```

- **From a Unix shell in lab:**

```
unix% ptman report_timing ;# ptman is an alias
```

1-37

PrimeTime man pages are stored in standard Unix man page format. In lab, an alias has been set up to enable PrimeTime man pages in Unix (this is useful for Solaris operating systems).

unix% alias ptman “man -M <path>/doc/pt/man”



Reading man pages from Unix came from SolvNet article “How Can I Find Information on a Command or Variable I don't Know?”

Doc Id: 901270 **Last Modified:** 05/09/01



To download the “always ask” aaTcl procedure, refer to SolvNet article “Find Commands and Variables with a Single Command”.

Doc Id: 012959 **Last Modified:** 10/01/2004

Useful Commands when in pt_shell 2/2

■ Turn on page mode with lab aliases

- page_on and page_off
- Type q to quit from long reports and return to the PrimeTime prompt

■ Use command line editing

- Command, switch, file name completion with the Tab key
- Emacs (default) or vi key bindings

■ Use the view utility for long reports

- Opens a pop-up window with a scroll bar
- ```
pt_shell> view report_timing
```

1-38



To download the view utility, refer to SolvNet article “An easy method for viewing long reports in the shell interface”.

**Doc Id:** 014947 **Last Modified:** 01/25/2005

To modify the editing mode in PrimeTime, use the variable **sh\_line\_editing\_mode**. To view the current key bindings, use the command **sh\_list\_key\_bindings**. To enable line editing, the following variable must be set to true in the **.synopsys\_pt.setup** file - **sh\_enable\_line\_editing**.

Page mode is controlled with the variable **sh\_enable\_page\_mode**.

# Agenda

DAY

1

i Introduction to STA in PrimeTime

1 STA Concepts and Flow in PrimeTime



2 Methodology: Qualifying Constraints



3 Methodology: Generating Reports



## Unit Objectives



After completing this unit, you should be able to:

- Verify that the design is completely constrained using `check_timing`
- Find number of timing checks in the design and how many are exercised / not exercised using `report_analysis_coverage`
- Using a Job Aid of 8 commands and 1 custom procedure, debug the issues found above
- Identify the clock and interface constraints out of timing reports

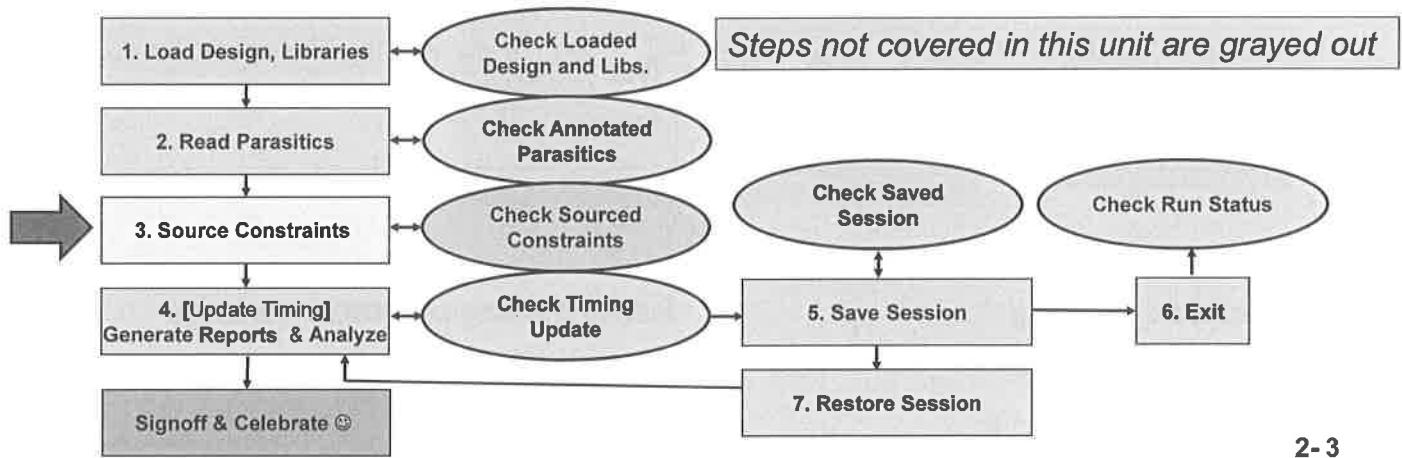
2-2

# Timing Analysis Flow in PrimeTime – Validating Constraints

- STA flow is divided into several steps

- Steps 1-3 read data and
- Analysis begins at Step-4

- The STA flow is repeated until signoff is achieved



2-3

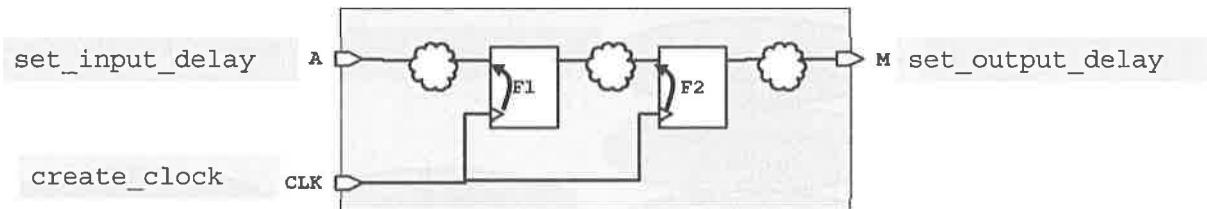
STA: Static Timing Analysis

## Requirements for Complete STA

### ■ For complete static timing analysis, you must:

- Completely constrain the design
- Exercise all needed timing checks

NOTE: Completeness != Correctness 



### ■ Two validation commands:

- `check_timing`
- `report_analysis_coverage`

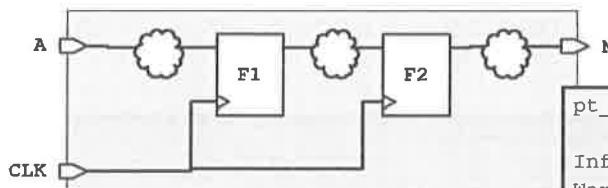
issues missing constraint warnings  
flags about untested timing checks

2-4

The complete list of checks performed by `check_timing` (bold indicates default checks):

```
clock_crossing
data_check_multiple_clock
data_check_no_clock
generated_clocks
generic
ideal_clocks
latch_fanout
latency_override
loops
ms_separation
multiple_clock
no_clock
no_driving_cell
no_input_delay
partial_input_delay
retain
signal_level
unconstrained_endpoints
unexpandable_clocks
```

# Constraint Warning: No Input Delay



Given this warning,  
you will want to find out:

```
pt_shell> check_timing -verbose
Information: Checking 'no_input_delay'.
Warning: There are 1 ports with no clock-relative input delay
specified. Since the variable 'timing_input_port_default_clock'
is 'true', a default input port clock will be assumed for these
ports.
Ports

A
```

NOTE: This warning is off by default, you will need to:  
`set_app_var timing_input_port_default_clock true`

What is the input port A connected to? (or is it unconnected?)

What paths are affected by this warning? (or is case analysis missing on input port?)

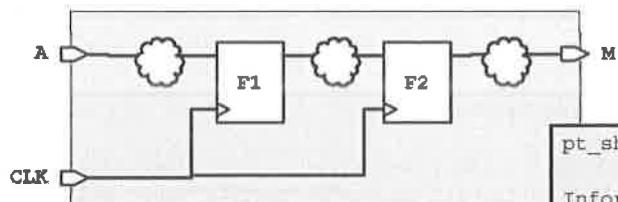
2-5

We need to check if we need to constrain the port:

Some possible reasons where user doesn't need to constrain an input port are:

- The port is supposed to drive a constant signal using a set\_case\_analysis e.g. A port driving the select switch of a MUX. Check if the case\_analysis setting has to be applied.
- If the path starting from the port will not be active in the current mode of analysis

## Constraint Warning: No Output Delay



```
pt_shell> check_timing -verbose
```

Information: Checking 'unconstrained\_endpoints'.  
Warning: There are 1 endpoints which are not constrained for maximum delay.

Endpoint

M

Given this warning,  
you will want to find out:

What is the output port M connected to?  
Register(s)? Input port(s)?

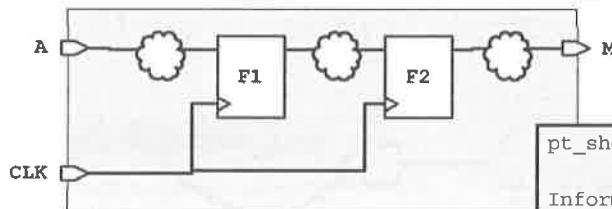
What paths are affected by this warning? (or is this an outgoing clock port?)

2-6

Check to see if endpoint must be constrained with output delay:

If the port is used to send an outgoing clock, then, we don't need to define an output delay on the port. Create the required generated clock and ignore the Warning.

## Constraint Warning: No Clock



```
pt_shell> check_timing -verbose
```

Information: Checking 'no\_clock'.

Warning: There are 2 register clock pins with no clock.

Clock Pin

-----

F1/CLK  
F2/CLK

Given this warning,  
you will want to find out:

Where should a clock be created to drive the affected ports? (The CLK port in this example)

2-7

Check to see if the clock signal is blocked in the fanout of clock port due to disabling of some arcs or user stopping the signal manually using set\_sense command.

```
pt_shell>set_sense -help
set_sense # Set sense on clock or data networks
[-type type] (Specify whether this is clock sense or data sense)

.
.

.
.

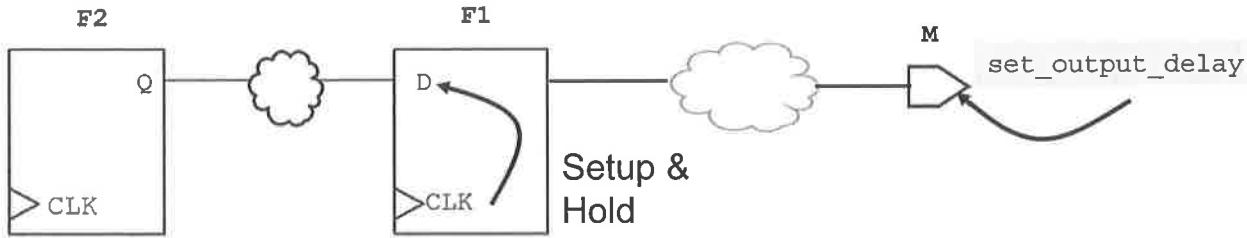
[-stop_propagation] (Stop propagation from specified pins)

.
.

.
.

pin_object_list (List of pins or cell timing-arcs)
```

## Timing Checks: Vendor Specified and User Specified



The command `report_analysis_coverage` checks each existing timing check in a design and if any is unexercised issues a warning and a reason.

Your job is to determine if the timing check is needed and then debug what is causing the timing check to be unexercised.

2-8

`report_analysis_coverage` checks for a variety of timing checks in the library and those from user constraints:

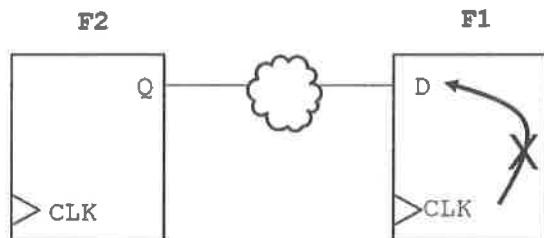
- recovery/removal
- setup/hold
- minimum\_pulse\_width
- min\_period
- out\_setup
- out\_hold
- and more ....

Reasons for “untested” timing checks:

- false\_paths
- user\_disabled
- constant\_disabled
- no\_paths
- mode\_disabled:
- no\_endpoint\_clock:
- no\_startpoint\_clock:
- no\_constrained\_clock:
- no\_ref\_clock:
- no\_clock:
- unknown:

## Untested Timing Check: `false_paths`

`report_analysis_coverage` gives `false_paths` as the reason when a user specifies `set_false_path`, asynchronous or exclusive clock groups.



Given this warning,  
you will want to find out:

```
pt_shell> report_analysis_coverage -status_details untested \
 -check setup
```

| Constrained Pin | Related Pin | Check Type | Slack    | Reason      |
|-----------------|-------------|------------|----------|-------------|
| F1/D(rise)      | CLK(rise)   | setup      | untested | false_paths |

What clocks are involved in these timing paths?

What start point flops are connected to F1?

Which false path (clock group) command caused this?

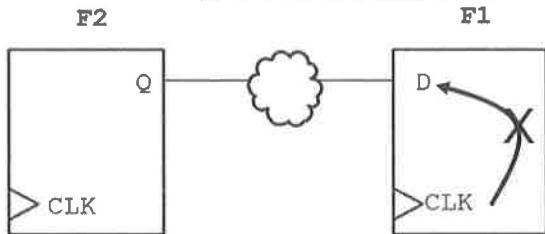
2-9

An example constraint that could cause the above warning:

`set_false_path -from F2/CLK -to F1/D`

## Untested Timing Check: user\_disabled

report\_analysis\_coverage gives user\_disabled as a reason when a user specifies set\_disable\_timing on a vendor specified timing check for a specific cell or on a library cell.



Given this warning,

you will want to find out:

```
pt_shell> report_analysis_coverage -status_details untested \
 -check setup
Constrained Related Check
Pin Pin Type Slack Reason

```

| Constrained Pin | Related Pin | Check Type | Slack    | Reason        |
|-----------------|-------------|------------|----------|---------------|
| F1/D(rise)      | CLK(rise)   | setup      | untested | user_disabled |

Is this timing check disabled for a specific cell (F1) or in the library?

2-10

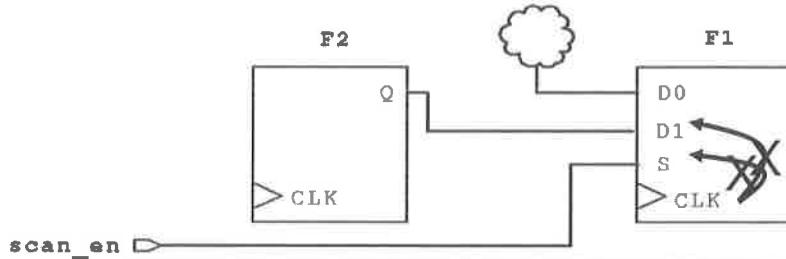
Two example constraints that could cause the above warning:

`set_disable_timing -from CLK -to D F1`

`set_disable_timing -from CLK -to D [get_lib_cell core_slow.db/fdesf2a15]`

## Untested Timing Check: constant\_disabled

report\_analysis\_coverage gives constant\_disabled as a reason for propagated constants due to a user specified set\_case\_analysis or to a signal tied high or low.



```
pt_shell> report_analysis_coverage -status_details untested \
 -check setup
```

| Constrained Pin | Related Pin | Check Type | Slack    | Reason            |
|-----------------|-------------|------------|----------|-------------------|
| <hr/>           |             |            |          |                   |
| F1/D1 (rise)    | CLK(rise)   | setup      | untested | constant_disabled |
| F1/S (rise)     | CLK(rise)   | setup      | untested | constant_disabled |

Given this warning,

you will want to find out:

Are disabled timing arcs due to a user specified case value or a signal tied high or low?

Which user specified case value is causing this specific disabled timing check?

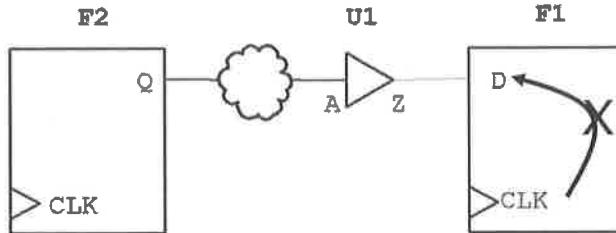
2-11

An example constraint that could cause the above warning:

`set_case_analysis 0 scan_en`

## Untested Timing Check: no\_paths

report\_analysis\_coverage gives no\_paths as a reason when there are no paths and thus no arrival times to a specific timing check.



```
pt_shell> report_analysis_coverage -status_details untested \
 -check setup
```

| Constrained Pin | Related Pin | Check Type | Slack    | Reason   |
|-----------------|-------------|------------|----------|----------|
| F1/D (rise)     | CLK(rise)   | setup      | untested | no_paths |

Given this warning,  
you will want to find out:

Are disabled timing arcs due to a user specified case value or a signal tied high or low?

Which user specified case value is causing this specific disabled timing check?

2-12

An example constraint that could cause the above warning:

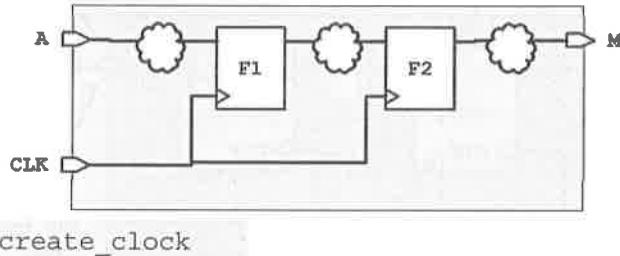
**set\_disable\_timing -from A -to Z U1**

# Test For Understanding



```
set_false_path -from [get_ports A]
```

```
set_input_delay
```



```
create_clock
```



If the constraints above are applied correctly, circle which of the following commands returns a warning.

check\_timing warns of  
constraint problems

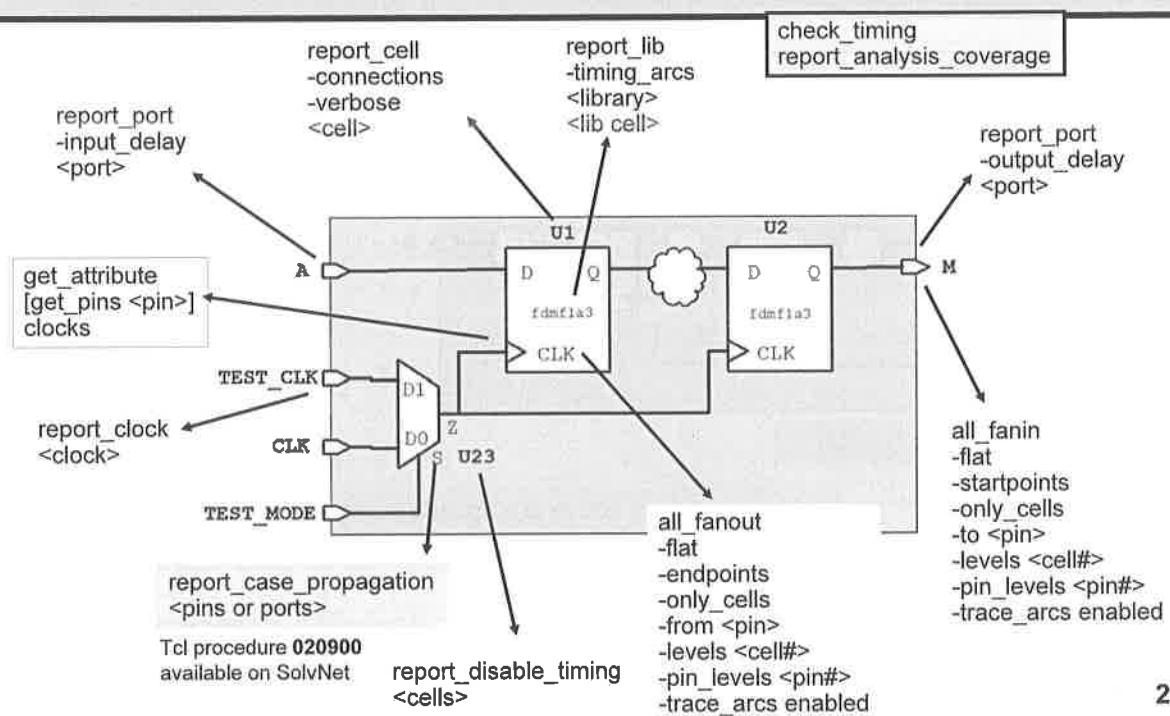
report\_analysis\_coverage  
warns of untested checks

2-13

- 1) check\_timing will warn about missing output delay constraint on Port M.  
2) report\_analysis\_coverage will warn about untested timing checks on F1 due to false\_path

Answer:

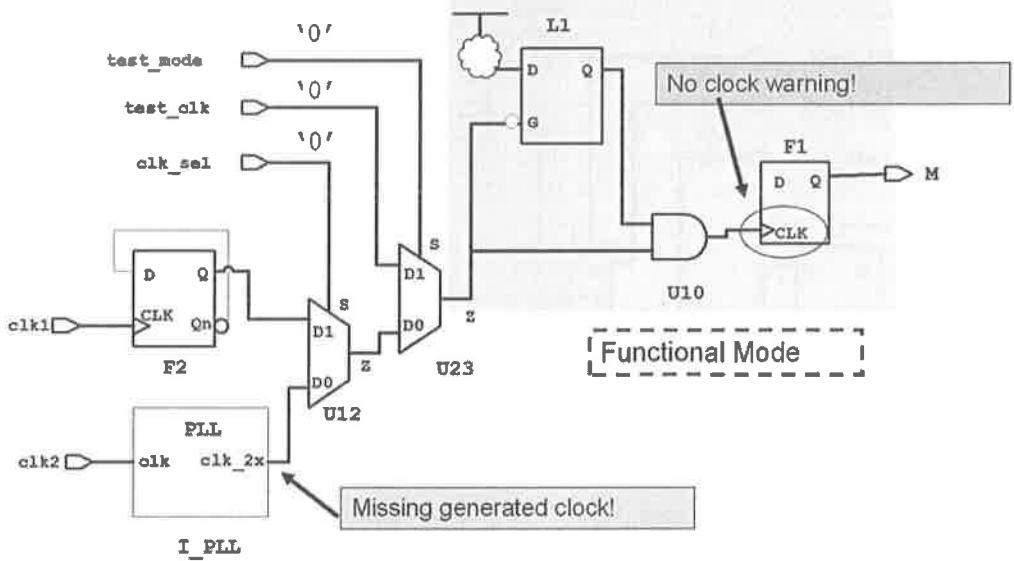
## Introducing Tools of the Trade Job Aid



2-14

For a copy of the Tcl procedure `report_case_propagation` and further reading, refer to SolvNet article "Tracing a constant back to its source", Doc ID: 020900 Last Modified: 31 May 2007

## Example Design with a Constraint Issue

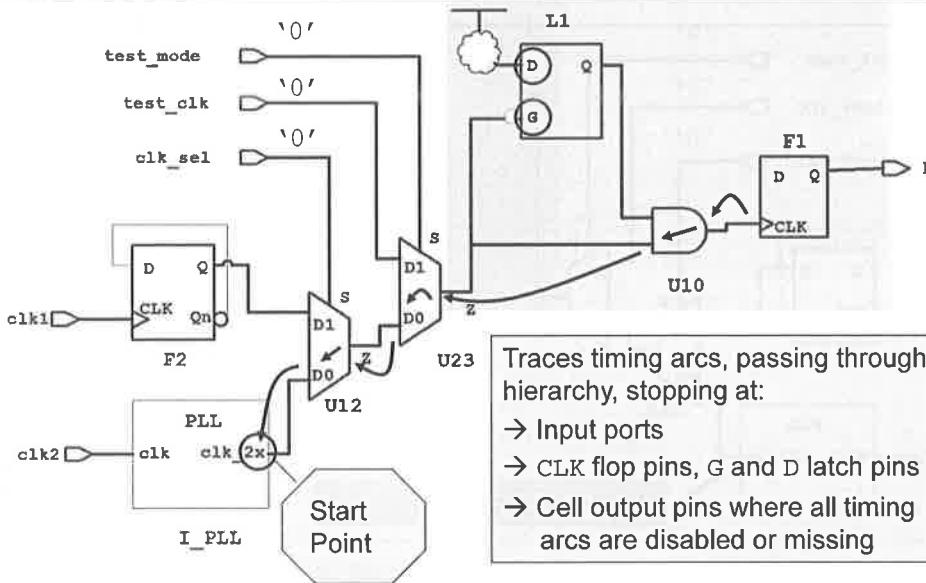


Use the following command to find the start point on the clock network:

```
all_fanin -startpoints -flat -to F1/CLK
```

2-15

## How Does all\_fanin Work?



2-16

```
pt_shell> all_fanin -help
```

```
all_fanin # Create a collection of pins/ports or cells in
```

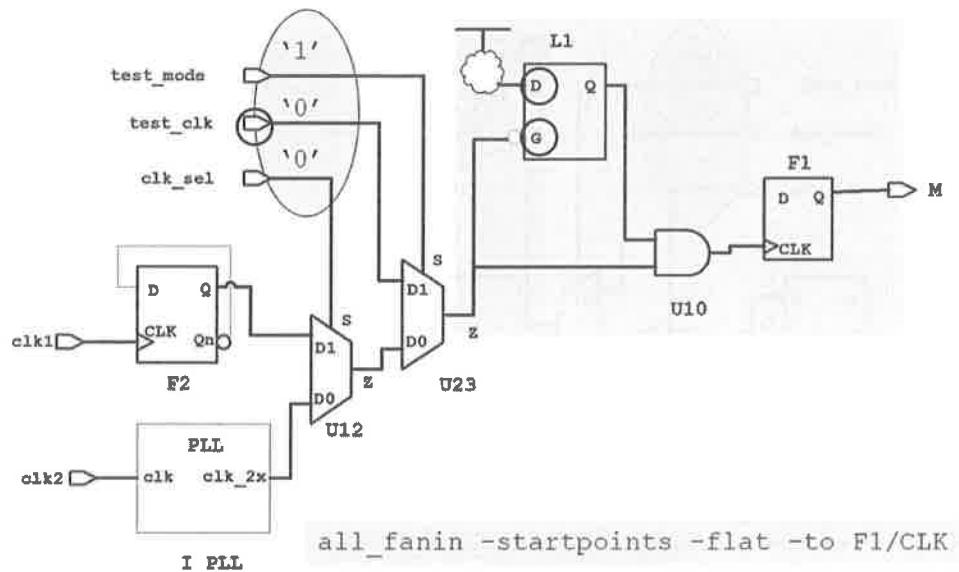
the

fanin of specified sinks

|                         |                                                                  |
|-------------------------|------------------------------------------------------------------|
| -to sink_list           | (List of sink pins, ports, or nets)                              |
| [-startpoints_only]     | (Find only the timing startpoints)                               |
| [-only_cells]           | (Return cells rather than pins)                                  |
| [-flat]                 | (Hierarchy is ignored)                                           |
| [-step_into_hierarchy]  | (Count levels inside sub-hierarchies)                            |
| [-levels cell_count]    | (Maximum number of cell levels to traverse:<br>Value >= 0)       |
| [-pin_levels pin_count] | (Maximum number of pin levels to traverse:<br>Value >= 0)        |
| [-trace_arcs arc_types] | (Type of network arcs to trace:<br>Values: timing, enabled, all) |

For a path constrained with case\_analysis, use report\_case\_propagation to see the values as they propagate along the fanin

## Another Start Point Example



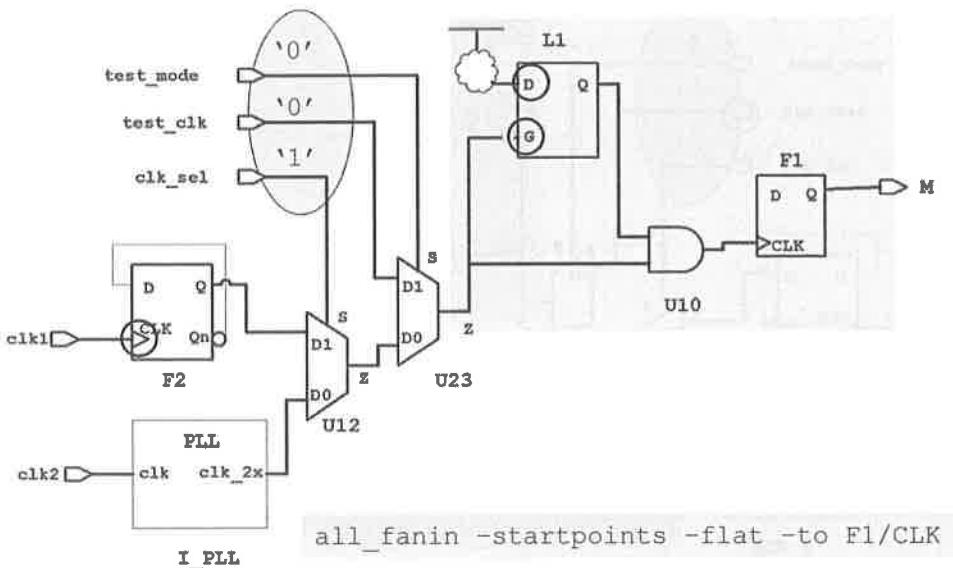
2-17

The command **all\_fanin**:

Traces timing arcs, passing through hierarchy, stopping at:

- Input ports
- CLK flop pins, G and D latch pins
- Cell output pins where all timing arcs are disabled or missing

## Final Start Point Example



2-18

The command `all_fanin`:

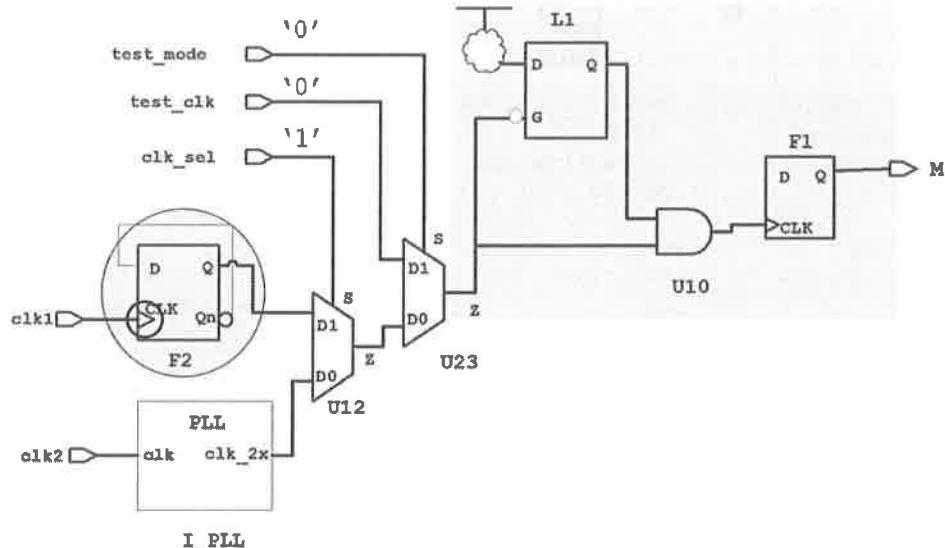
Traces timing arcs, passing through hierarchy, stopping at:

- Input ports
- CLK flop pins, G and D latch pins
- Cell output pins where all timing arcs are disabled or missing

## Further Details on Start Point

Use the following command to find the start point cell details:

```
report_cell -connections -verbose F2
```

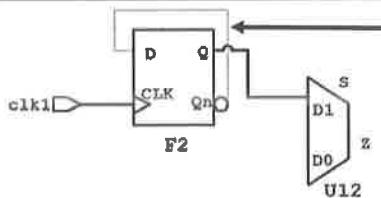


2-19

Note: You can also use the following

```
report_cell -conn -v [all_fanin -flat -only_cells -to F1/CLK -startpoints]
```

## “Sketch” Cell Connections



The sequential feedback net name  
n1 is in the report below.

```
pt_shell> report_cell -connections -verbose [get_cells F2]
Reference: fdmf2a3
Library: core_slow.db
Area: 67.74
```

| Input Pins  | Net  | Net Driver Pins | Driver Pin Type      |
|-------------|------|-----------------|----------------------|
| D           | n1   | F2/Qn           | Output Pin (fdmf2a3) |
| CLK         | clk1 | clk1            | Input Port           |
| Output Pins | Net  | Net Load Pins   | Load Pin Type        |
| Q           | n2   | U12/DI          | Input Pin (mx2a1)    |
| Qn          | n1   | F2/D            | Input Pin (fdmf2a3)  |

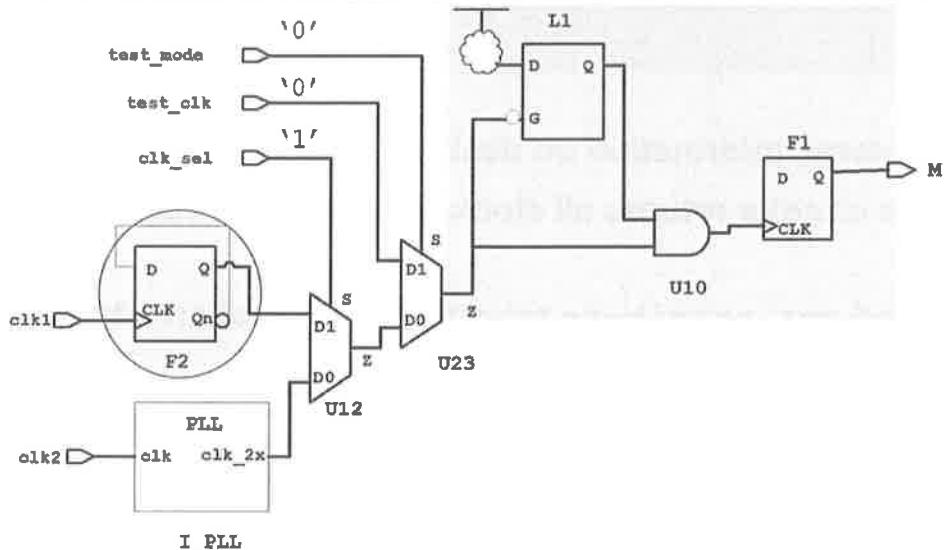
2-20

```
pt_shell> report_cell -help
report_cell
 [-connections] # Report cell info
 [-verbose] (Show cell connection info)
 [-nosplit] (Show all cell info)
 [-significant_digits digits] (Do not split lines if column overflows)
 [cell_list] (Number of digits to display:Range: 0 to 13)
 (List of cells)
```

## Find the Master Clock for Defining Generated Clock

Use the following command to identify the master clock name for the div-by 2 flop:

```
get_attribute [get_pins F2/CLK] clocks
```



2-21

## How Does get\_attribute Work?

Which clock(s) propagate to the clock pin of F2?

```
pt_shell> get_attribute [get_pins F2/CLK] clocks
{"Clk1"}
```

- PrimeTime stores information on design objects using attributes
- The attribute **clocks** returns all clocks that pass through the given pin or port
- The command **get\_attribute** returns the value of an attribute set on a single design object (e.g. a pin)
  - Use **foreach\_in\_collection** to loop through multiple pins

2-22

The application attribute **clocks** is set by PrimeTime on all pins in the clock network when the user issues a **create\_clock** or **create\_generated\_clock** command. The attribute is a collection which contains every clock passing through a given pin.

Another interesting attribute is called **clock\_network\_pins** and exists on clock objects. This attribute is a collection of all the pins in the clock network for a given clock.

```
pt_shell> get_attribute [get_clocks Clk1] clock_network_pins
→ {"F2/CLK"}
```

To gain more experience writing Tcl procedures and accessing design objects and their attributes, please refer to the following workshops offered by Customer Education.

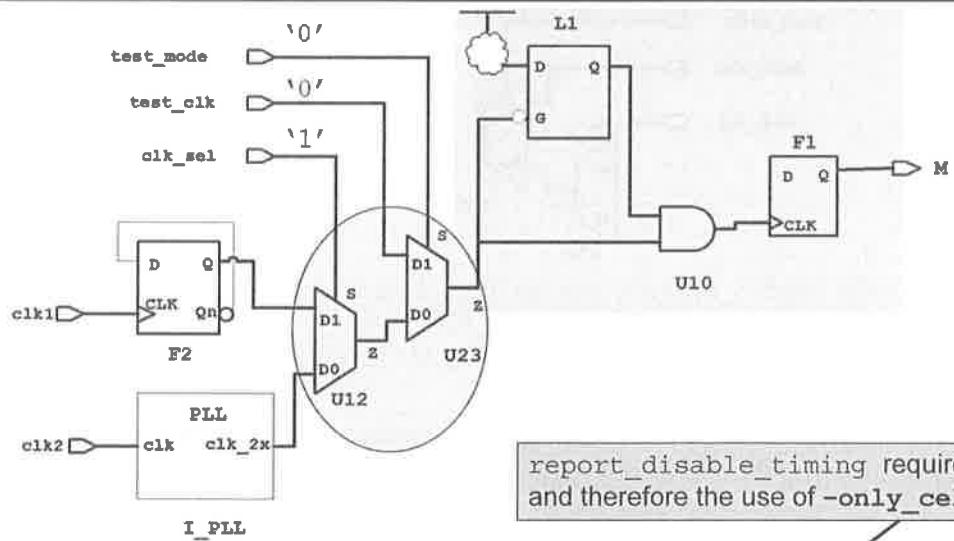
**Tcl 1: Becoming a Proficient User**

**Tcl 2: Creating High Impact Procedures**

**Tcl 3: Direct Access Through Collections and Attributes**



## Find Disabled Timing Arcs Along Path

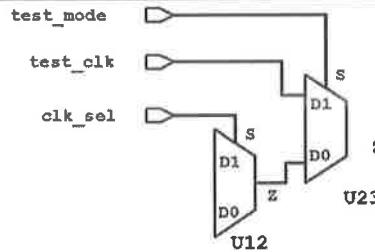


Use the following command to identify disabled arcs along the clock path:

```
report_disable_timing [all_fanin -only_cells -to F1/CLK -flat]
```

2-23

## Identify Causes for Disabled Arcs



## Reason for the disabled timing arcs

```
pt_shell> report_disable_timing [all_fanin -flat -only -to F1/CLK]
```

Flags :      c case-analysis  
              p propagated constant

| Cell or Port | From | To | Sense          | Flag | Reason |
|--------------|------|----|----------------|------|--------|
| U23          | S    | Z  | positive_unate | c    | S = 0  |
| U23          | S    | Z  | negative_unate | c    | S = 0  |
| U23          | D1   | Z  | positive_unate | c    | S = 0  |
| U12          | S    | Z  | positive_unate | c    | S = 1  |
| U12          | S    | Z  | negative_unate | c    | S = 1  |
| U12          | D0   | Z  | positive unate | c    | S = 1  |

2-24

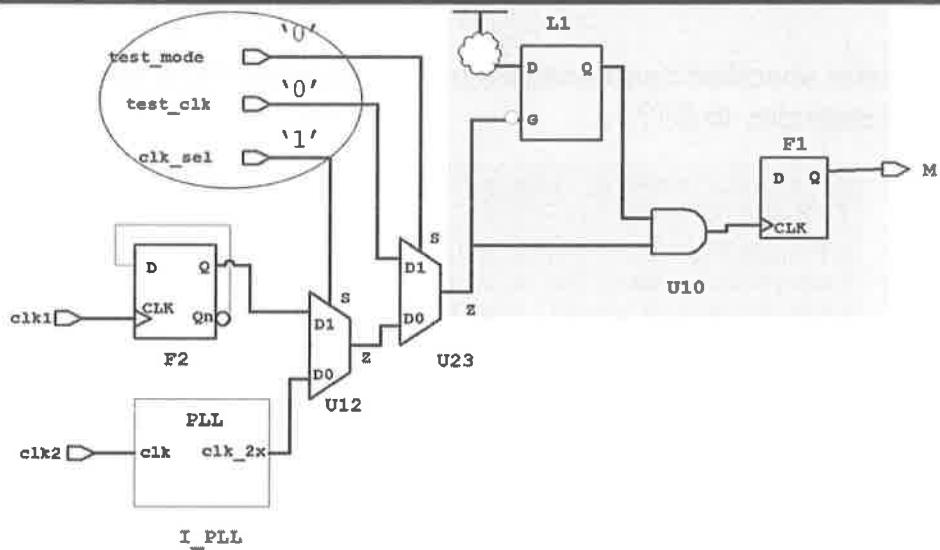
```
pt_shell> report_disable_timing -help
report_disable_timing # Report disabled timing arcs
 [-nosplit] (Do not split lines if column overflows)
 [object spec] (Show only disabled arcs for specified cells or ports)
```

The complete list of flags for `report_disable_timing` includes:

### Flags :

- c case-analysis
  - C Conditional arc
  - d default conditional arc
  - f false net-arc
  - l loop breaking
  - p propagated constant
  - u user-defined

## Find the User Specified Case Values



Use the following command to identify user specified case value:

```
report_case_propagation [get_pins U23/S]
```

2- 25

## How Does report\_case\_propagation Work?

Has a user specified case analysis been propagated to a select pin, for example, to S1?

```
pt_shell> report_case_propagation I_CLOCK_GEN/U21/S1
I_CLOCK_GEN/U21/S1 (core_slow.db/mx3a15) case=0
test_mode_iopad/C (io_slow.db/PDIDGZ) case=0
test_mode_iopad/PAD (io_slow.db/PDIDGZ) case=0
test_mode (in port) user-defined case=0
```

Traces the fanin back to the startpoint – If case analysis has been applied to both inputs of a gate, traces both branches

Confirm your finding

```
pt_shell> report_case_analysis
Pin name User case analysis value

test_mode 0
```

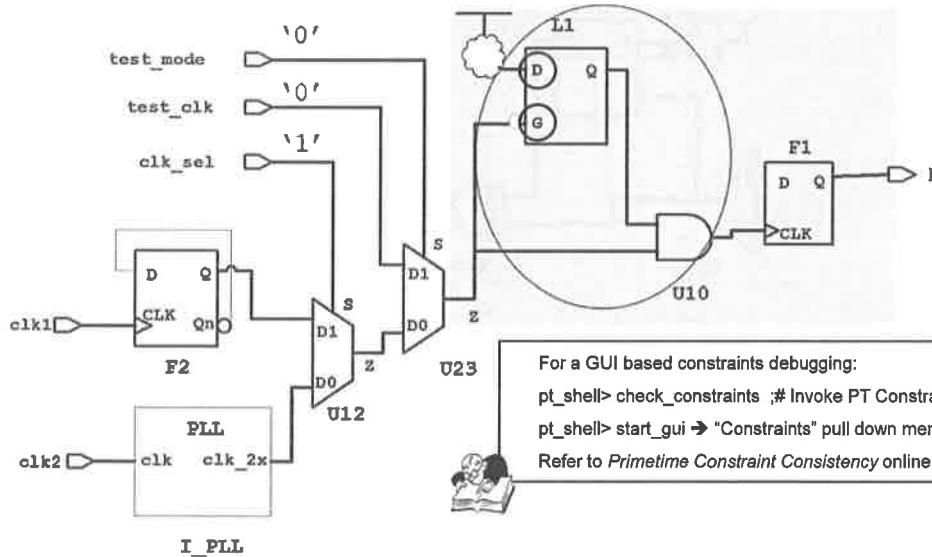
2-26

```
pt_shell> report_case_analysis -help
report_case_analysis # Report case analysis on ports and pins
[-all] (Report logic constant pins of the design)
[-nosplit] (Do not split lines if column overflows)
```

## Dismiss Clock Gating Logic

Use the following command that returns the input pins of L1:

```
all_fanin -start -flat -to F1/CLK
```

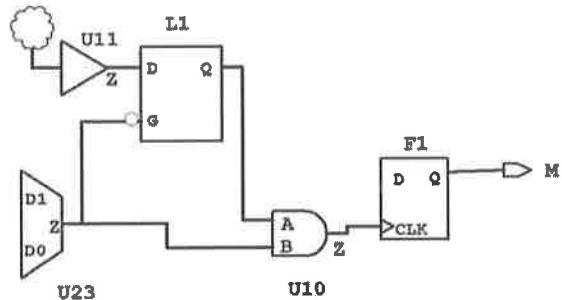


For a GUI based constraints debugging:  
pt\_shell> check\_constraints ;# Invoke PT Constraint Consistency  
pt\_shell> start\_gui ➔ "Constraints" pull down menu  
Refer to *Primetime Constraint Consistency* online Documentation

2-27

## Sketch a Schematic

```
report_cell -connections -verbose L1
```

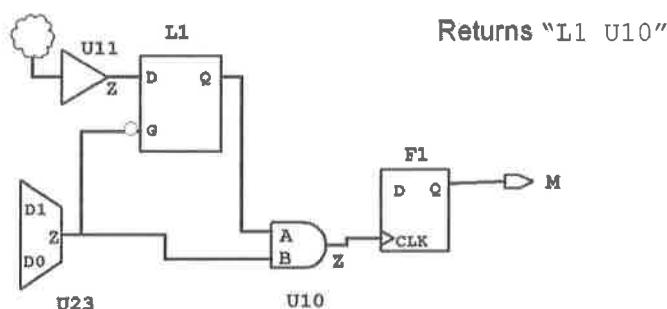


Identifies the pins of L1 plus the nets and pins in the immediate fanin and fanout of L1.

2-28

## Expand Sketch One Level At a Time

```
report_cell -connections -verbose \
 [all_fanout -levels 1 -flat -only -from L1/Q] ←
```



For a GUI based constraints debugging:

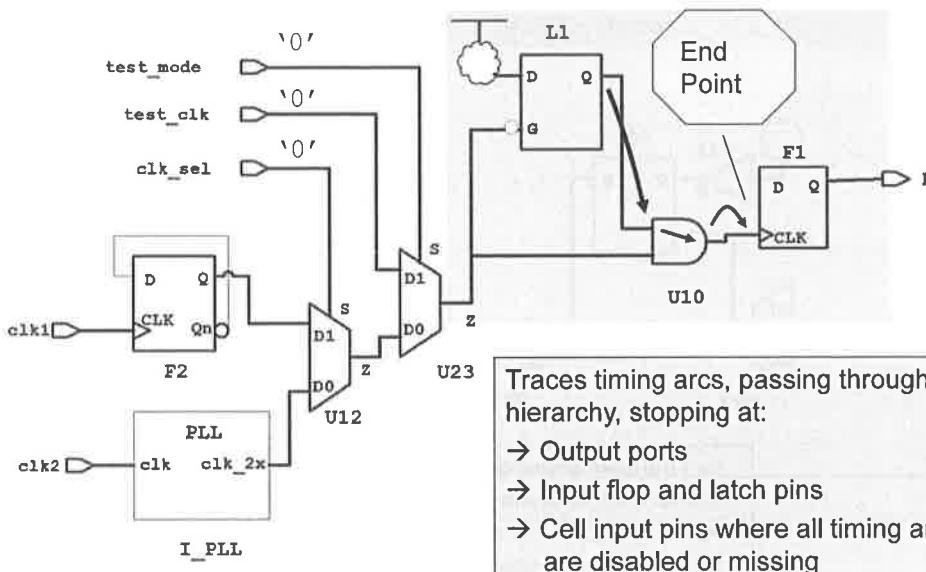
```
pt_shell> check_constraints ;# Invoke PT Constraint Consistency
pt_shell> start_gui ➔ "Constraints" pull down menu ➔ "View Constraint Checking Results"
Refer to Primetime Constraint Consistency online Documentation
```



2-29

## How Does all\_fanout Work?

all\_fanout -flat -endpoints -from L1/Q



2-30

pt\_shell> all\_fanout -help

all\_fanout # Create a collection of pins/ports or cells in the fanout of specified sources.

-from sink\_list  
-clock\_tree  
[-endpoints\_only]  
[-only\_cells]  
[-flat]  
[-step\_into\_hierarchy]  
[-levels level\_count]  
traverse:  
Value >= 0)

(List of source pins, ports, or nets)

(Return list of clock tree components)

(Find only the timing endpoints)

(Return cells rather than pins)

(Hierarchy is ignored)

(Count levels inside sub-hierarchies)

(Maximum number of cell levels to

[-pin\_levels pin\_count] (Maximum number of pin levels to traverse:  
Value >= 0)

[-trace\_arcs arc\_types]  
(Type of network arcs to trace:  
Values: timing, enabled, all)

To use all\_fanout on a path constrained by set\_case\_analysis, you must use the -trace\_arcs enabled option.

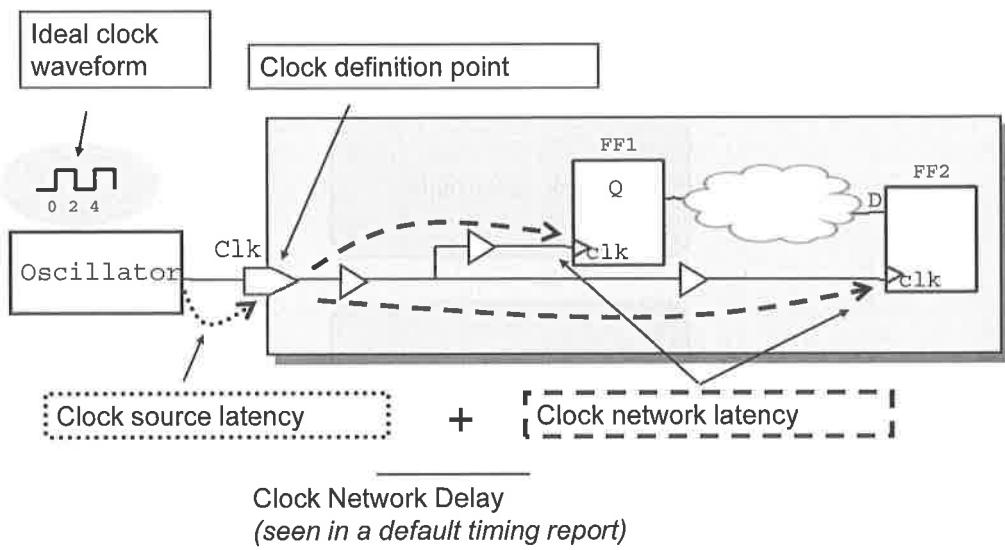
# Identifying Constraints from Timing Reports

Clock constraints

Interface constraints

2-31

# Clock Latency Components



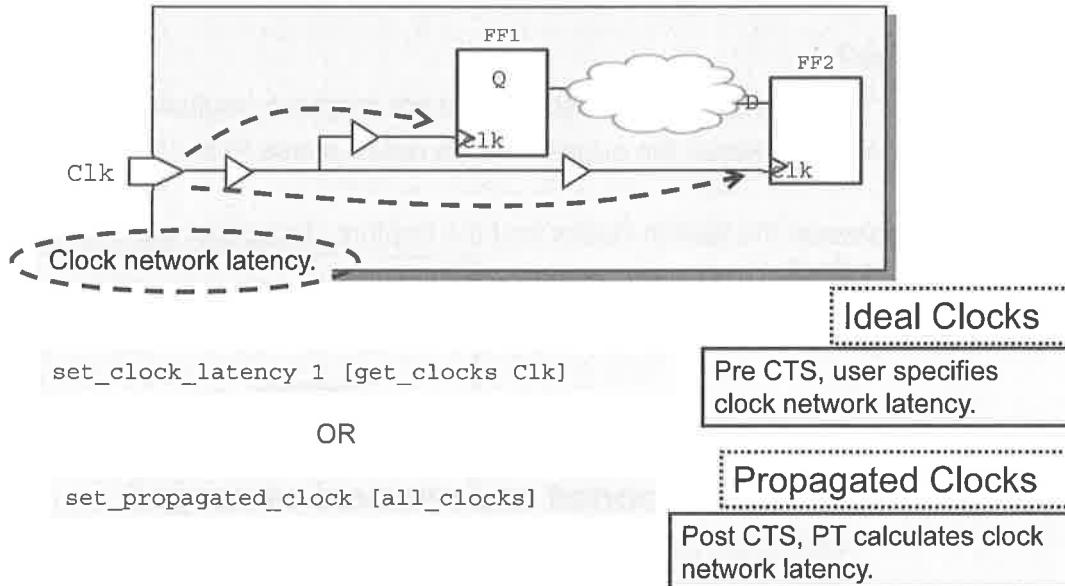
```
create_clock -period 4 [get_ports Clk]
set_clock_latency -source 2 [get_clocks Clk]
```

2-32

Example constraint commands used to create this clock:

```
create_clock -period 4 [get_ports Clk]
set_clock_latency -source 2 [get_clocks Clk]
set_clock_latency 1 [get_clocks Clk]; # For ideal clock network latency
```

## Pre Versus Post Clock Tree Synthesis (CTS)



2- 33

CTS

Clock tree synthesis

By default, clocks are created as ideal clocks. To constrain all clock as propagated, execute the following command:

```
set_propagated_clock [all_clocks]
```

### Ideal clocks

Used prior to clock tree synthesis (CTS). Min/max clock network latency and skew (or uncertainty) are estimated by you.

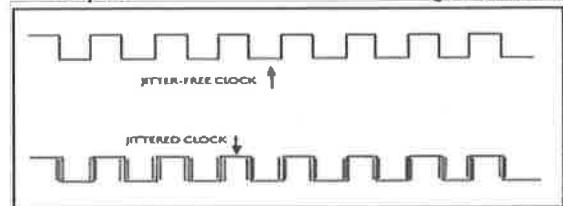
### Propagated clocks

Used after CTS. Min/max clock network latency and skew are calculated by PrimeTime.

# Specifying Clock jitter

## ■ What is Clock Jitter?

- Cycle-to-cycle jitter - Variation between the edges that are in-phase (multiple clock cycles apart)
- Duty-cycle jitter - Variation between the edges that are out-of-phase (0.5, 1.5, 2.5, ... cycles apart)
- Clock jitter is used between the launch clocks and the capture clocks that are both generated from the same master clock
- Clock jitter is only applied to the end points



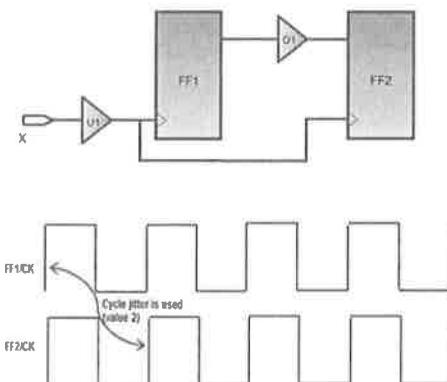
## ■ Clock jitter can be specified, reported and removed as needed

```
set_clock_jitter
report_clock_jitter
remove_clock_jitter
```

2- 34

## Example: Cycle to Cycle Jitter

```
create_clock x -name mclk -period 10
set_clock_jitter -cycle 2 -duty_cycle 3 -clock mclk
```



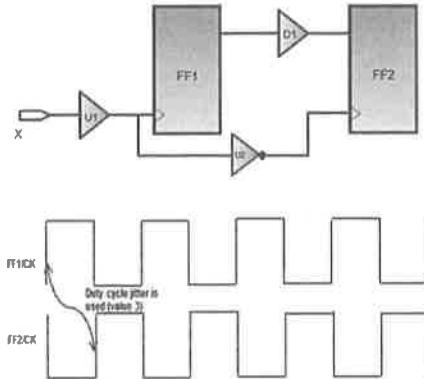
| Point                       | Index  | Path    |
|-----------------------------|--------|---------|
| clock mclk (rise edge)      | 0.00   | 0.00    |
| clock source latency        | 0.00   | 0.00    |
| x (in)                      | 0.00 z | 0.00 z  |
| U1/Y (BUF)                  | 0.09 z | 0.09 z  |
| FF1/CR (SDFF)               | 0.00 z | 0.09 z  |
| FF1/Q (SDFF)                | 0.11 z | 0.20 z  |
| U1/Y (BUF)                  | 0.08 z | 0.28 z  |
| FF2/CR (SDFF)               | 0.00 z | 0.28 z  |
| date arrival time           |        | 0.28    |
| clock mclk (rise edge)      | 10.00  | 10.00   |
| clock source latency        | 0.00   | 10.00   |
| x (in)                      | 0.00 z | 10.00 z |
| U1/Y (BUF)                  | 0.09 z | 10.09 z |
| FF1/CR (SDFF)               | 0.00 z | 10.09 z |
| clock convergence pessimism | 0.00   | 10.09   |
| cycle clock jitter          | -2.39  | 8.09    |
| library setup time          | 0.00   | 8.09    |
| data required time          |        | 8.09    |
| data arrival time           |        | -0.28   |
| blank (BUF)                 |        | 7.81    |

2- 35

In this example both flops receive the same clock senses, i.e., rise sense at launch and capture flops, these are in phase clock cycle. So cycle to cycle jitter is applied for max (setup) paths.

## Example: Duty-Cycle Jitter

```
create_clock x -name mclk -period 10
set_clock_jitter -cycle 2 -duty_cycle 3 -clock mclk
```



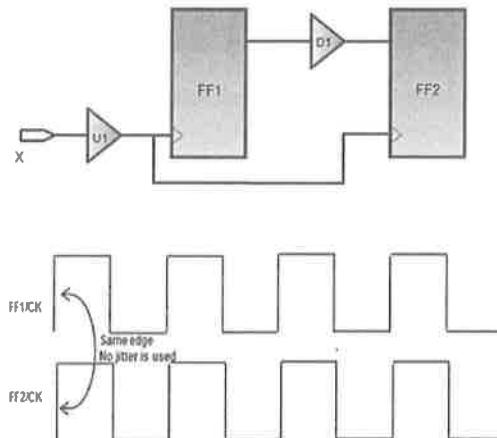
| Point                    | INCLK  | Path   |
|--------------------------|--------|--------|
| clock mclk (rise edge)   | 0.00   | 0.00   |
| clock source latency     | 0.00   | 0.00   |
| x (in)                   | 0.00 z | 0.00 z |
| U1/Y (BUF)               | 0.09 z | 0.09 z |
| FF1/CLK (SDFF)           | 0.00 z | 0.09 z |
| FF1/Q (SDFF)             | 0.11 z | 0.20 z |
| D1/Y (BUF)               | 0.08 z | 0.28 z |
| FF2/D (SDFF)             | 0.00 z | 0.28 z |
| data arrival time        |        | 0.28   |
| clock mclk' (rise edge)  | 5.00   | 5.00   |
| clock source latency     | 0.00   | 5.00   |
| x (in)                   | 0.00 z | 5.00 z |
| U1/Y (BUF)               | 0.08 z | 5.08 z |
| U2/Y (INV)               | 0.07 z | 5.15 z |
| FF2/CLK (SDFF)           | 0.00 z | 5.15 z |
| clock re-synchronization | 0.00   | 5.15   |
| duty_cycle_slack_jitter  | -1.00  | 5.15   |
| library setup time       | -0.01  | 2.14   |
| data required time       |        | 2.14   |
| data required time       |        | 2.14   |
| data arrival time        |        | -0.28  |
| slack (MET)              |        | 1.86   |

2-36

In this example, the launch edge is rising and the capture edge is falling. Therefore, duty cycle jitter is applied.

## Example: No Jitter

```
create_clock x -name mclk -period 10
set_clock_jitter -cycle 2 -duty_cycle 3 -clock mclk
```



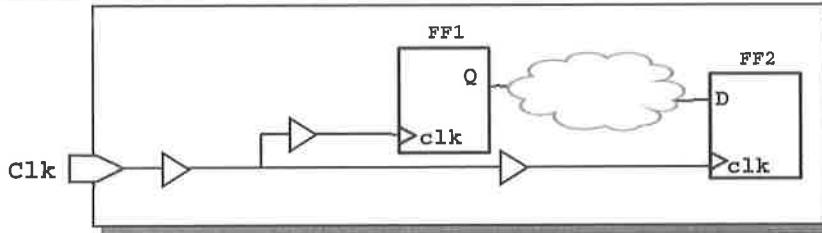
| Startpoint: FF1 (rising edge-triggered flip-flop clocked by mclk) |        |        |
|-------------------------------------------------------------------|--------|--------|
| Endpoint: FF2 (rising edge-triggered flip-flop clocked by mclk)   |        |        |
| Path Group: mclk                                                  |        |        |
| Path Type: min                                                    |        |        |
| Point                                                             | INCLK  | Path   |
| clock mclk (rise edge)                                            | 0.00   | 0.00   |
| clock source latency                                              | 0.00   | 0.00   |
| x (in)                                                            | 0.00 # | 0.00 r |
| U1/Y (BUF)                                                        | 0.09 # | 0.09 r |
| FF1/CN (SDFF)                                                     | 0.00 # | 0.09 r |
| FF1/Q (SDFF)                                                      | 0.10 # | 0.19 f |
| D1/Y (BUF)                                                        | 0.07 # | 0.26 f |
| FF2/D (SDFF)                                                      | 0.00 # | 0.26 f |
| data arrival time                                                 |        | 0.26   |
| clock mclk (rise edge)                                            | 0.00   | 0.00   |
| clock source latency                                              | 0.00   | 0.00   |
| x (in)                                                            | 0.00 # | 0.00 r |
| U1/Y (BUF)                                                        | 0.09 # | 0.09 r |
| FF2/CN (SDFF)                                                     | 0.00 # | 0.09 r |
| clock convergence pessimism                                       | 0.00   | 0.09   |
| library hold time                                                 | -0.01  | 0.08   |
| data required time                                                |        | 0.08   |
| data arrival time                                                 |        | -0.26  |
| slack (MET)                                                       |        | 0.18   |

2-37

In this example the same clock sense is reaching the launch and capture flops for zero cycle (hold timing) paths. Therefore, no jitter is applied.

# Clock Uncertainty and Skew

Clock uncertainty is another constraint specified by the designer.



```
set_clock_uncertainty 0.4 [get_clocks Clk]
```

Pre CTS

Clock Uncertainty = Clock skew + Margin

Post CTS

Clock Uncertainty = Margin

Traditional specifications have combined clock jitter into clock uncertainty. However, the correct way to apply clock jitter would be by using `set_clock_jitter`

Why is clock skew removed post CTS?

2-38

To specify clock uncertainty, use the following command:

```
set_clock_uncertainty 0.4 [get_clocks Clk]
```

PrimeTime is calculating the actual clock skew for every timing path. Pre-CTS, the clocks are ideal and the user must guess and specify the worst case skew which will be applied to every timing path. Why is clock skew removed post CTS - Post-CTS the clocks should be propagated and thus PrimeTime is calculating the actual clock skew for every timing path. Pre-CTS, the clocks are ideal and the user must guess and specify the worst case skew which will be applied to every timing path.

# Clock Constraints From Timing Report

Clock constraints are contained in the timing report.

| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 1.10 & | 1.10   |
| FF1/CLK (fdef1a15)               | 0.00   | 1.10 r |
| FF1/Q (fdef1a15)                 | 0.50 & | 1.60 r |
| U2/Y (buf1a27)                   | 0.11 & | 1.71 r |
| U3/Y (buf1a27)                   | 0.11 & | 1.82 r |
| FF2/D (fdef1a15)                 | 0.05 & | 1.87 r |
| data arrival time                |        | 1.87   |
| clock Clk (rise edge)            | 4.00   | 4.00   |
| clock network delay (propagated) | 1.00 & | 5.00   |
| clock uncertainty                | -0.40  | 4.60   |
| FF2/CLK (fdef1a15)               |        | 4.60 r |
| library setup time               | -0.21  | 4.39   |
| data required time               |        | 4.39   |
| -----                            |        |        |
| data required time               |        | 4.39   |
| data arrival time                |        | -1.87  |
| -----                            |        |        |
| slack (MET)                      |        | 2.52   |

2-39

Clock constraints are contained in this timing report – ideal clock waveform (clock period). Clock is propagated (not ideal) – that is, it is calculated, and therefore not a constraint – clock uncertainty – maybe there is clock source latency, but you cannot tell from the default report, which combines network latency (whether ideal or propagated) and source latency into a single number for network delay – to separate the two, do a report – timing – path full – clock – expanded

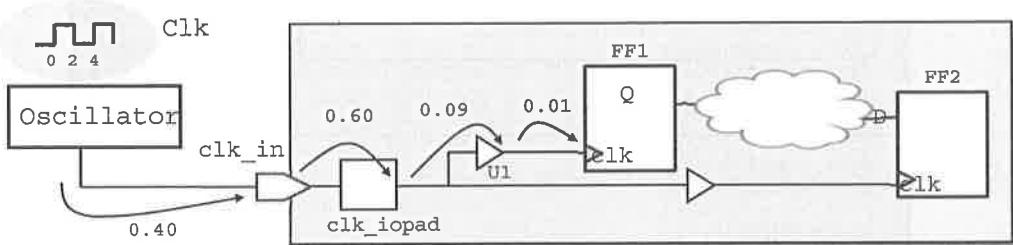
Methodology: Qualifying Constraints  
PrimeTime

## Reporting Clock Source Latency and Clock Network Cells

report\_timing -path full\_clock

| Point                                     | Incr   | Path   |
|-------------------------------------------|--------|--------|
| clock Clk (rise edge)                     | 0.00   | 0.00   |
| clock source latency                      | 0.40   | 0.40   |
| clk_in (in) <----> Clock definition point | 0.00   | 0.40 r |
| clk_iopad/PAD (pc3d01)                    | 0.60 & | 1.00 r |
| U1/Y (buf1a27)                            | 0.09 & | 1.09 r |
| FF1/CLK (fdef1a15)                        | 0.01 & | 1.10 r |
| FF1/Q (fdef1a15)                          | 0.40 & | 1.50 f |
| U2/Y (buf1a27)                            | 0.05 & | 1.55 f |
| U3/Y (buf1a27)                            | 0.05 & | 1.60 f |
| FF2/D (fdef1a15)                          | 0.01 & | 1.61 f |
| data arrival time                         |        | 1.61   |

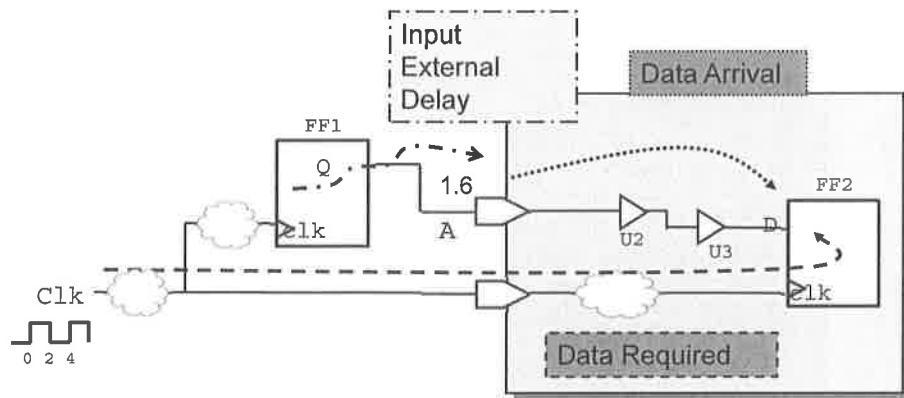
- Clock ideal waveform
- Clock source latency
- Clock network latency



2-40

## Interface Paths: Input Ports

Input delay is a constraint that represents the arrival times at the input ports of the design with respect to the launch clock.



```
set_input_delay 1.60 -max -clock Clk [get_ports A]
```

2-41

Specify the arrival time for setup to input ports with the following command:

```
set_input_delay 1.60 -max -clock Clk [get_ports A]
```

## Input Delay Constraint From Timing Report

The report contains  
Input delay constraint  
Input port name  
External start point clock

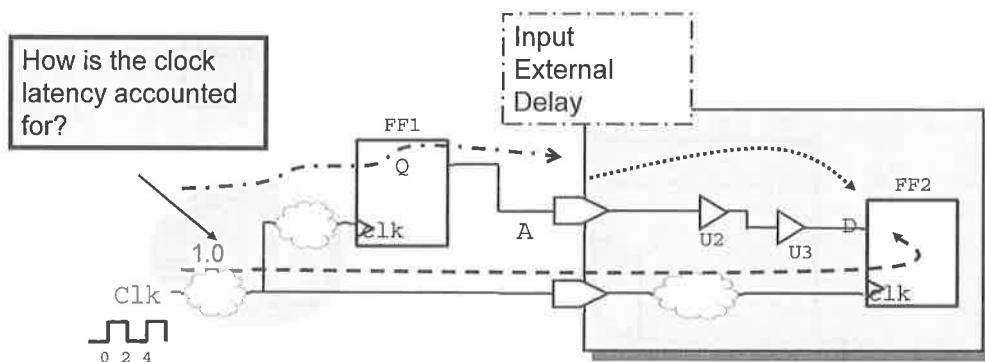
| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 0.00   | 0.00   |
| input external delay             | 1.60   | 1.60 r |
| A (in)                           | 0.00   | 1.60 r |
| U2/Y (buf1a27)                   | 0.11 & | 1.71 r |
| U3/Y (buf1a27)                   | 0.11 & | 1.82 r |
| FF2/D (fddef1a15)                | 0.05 & | 1.87 r |
| data arrival time                |        | 1.87   |
| clock Clk (rise edge)            | 4.00   | 4.00   |
| clock network delay (propagated) | 1.00 & | 5.00   |
| FF2/CLK (fddef1a15)              |        | 5.00 r |
| library setup time               | -0.21  | 4.79   |
| data required time               |        | 4.79   |
| data required time               |        | 4.79   |
| data arrival time                |        | -1.87  |
| slack (MET)                      |        | 2.92   |

2-42

External start point clock - Clk  
Input port name - A  
Input delay constraint - 1.6

Answers:

## A Cleaner Solution – Constraining Interface Paths



- One clean solution - include the clock latency as part of the input delay constraint

```
set_input_delay -network_latency_included \
 -source_latency_included -max 2.6 -clock clk [get_ports A]
```

- When debugging violations on interface paths – discuss how the clock latencies are being represented:  
Network latency defined on C1k is used by IO path if “ideal”  
– can not be used with “propagated” clocks

2-43

Specify the arrival time for setup to input ports with the source and network latency included.

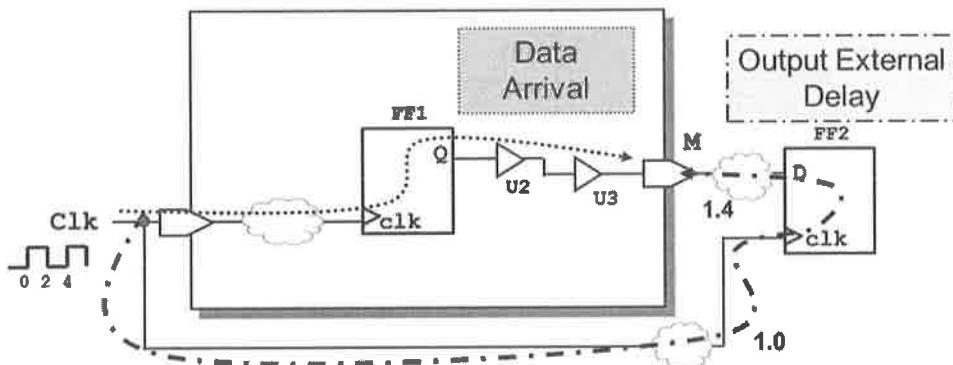
```
set_input_delay [expr {1.60 + 1.0}] -max -
 network_latency_included \
 -source_latency_included -clock Clk [get_ports A]
```

For setup analysis, omitting the clock latency for the launch clock will result in the input timing path being optimistically constrained (the path may falsely meet timing). For hold analysis, omitting the clock latency for the launch clock will result in the input timing path being pessimistically constrained (the path may falsely violate timing).

For output timing paths, the opposite will be true.

## Interface Paths: Output Ports

Output delay is a constraint that represents the setup and hold at the output ports of the design with respect to the capture clock.



```
set_output_delay 0.40 -max -clock Clk -network -source [get_ports M]
```

2-44

Specify the output external delay for setup with the following command:

```
set_output_delay [expr {1.4 - 1.0}] -max -clock Clk \
 -network_latency_included \
 -source_latency_included [get_ports M]
```

## Output Delay Constraint From Timing Report



The report contains  
Output delay constraint  
Output port name  
External end point clock

| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 1.10 & | 1.10   |
| FF1/CLK (fdef1a15)               | 0.00   | 1.10 r |
| FF1/Q (fdef1a15)                 | 0.50 & | 1.60 r |
| U2/Y (buf1a27)                   | 0.11 & | 1.71 r |
| U3/Y (buf1a27)                   | 0.11 & | 1.82 r |
| M (out)                          | 0.05 & | 1.87 r |
| data arrival time                |        | 1.87   |
| clock Clk (rise edge)            | 4.00   | 4.00   |
| clock network delay (propagated) | 0.00   | 4.00   |
| output external delay            | -0.40  | 3.60   |
| data required time               |        | 3.60   |
| data required time               |        | 3.60   |
| data arrival time                |        | -1.87  |
| slack (MET)                      |        | 1.73   |

2-45

External end point clock - Clk  
Output port name - M  
Output delay constraint - -0.40

ANSWERS:

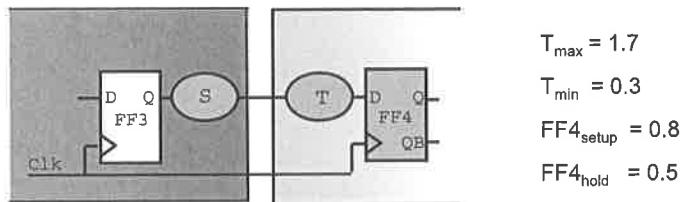
## Constraining Output Port for Setup and Hold Requirements

- **set\_output\_delay -max:**

- Describes the external setup time requirement on the output ports

- **set\_output\_delay -min:**

- Describes the external hold time requirement on the output ports



$$\text{Output Delay}_{\max} = T_{\max} + FF4_{\text{setup}}$$

$$\text{Output Delay}_{\min} = T_{\min} - FF4_{\text{hold}}$$

```
set_output_delay -max 2.5 -clock Clk [all_outputs]
set_output_delay -min -0.2 -clock Clk [all_outputs]
```

2-46

## Review of Unit Objectives



**Having completed this unit, you should be able to:**

- Verify that the design is completely constrained using `check_timing`
- Find number of timing checks in the design and how many are exercised / not exercised using `report_analysis_coverage`
- Using a Job Aid of 8 commands and 1 custom procedure, debug the issues found above
- Identify the clock and interface constraints out of timing reports

2-47

## Lab 2: Constraints in a Timing Report



30 minutes

Examine Constraint Completeness and  
Untested Timing Checks

Identify and interpret constraints in a timing report

2-48

## **Appendix**

### **Tests For Understanding: Finding Constraints From Reports**

**2-49**

## Test For Understanding : Clock Constraints 1/2



The clock network latency is:

Estimated by the user.  Calculated by PrimeTime.

The calculated clock skew of 0.10ns causes a:

Smaller positive slack.  Larger positive slack.

| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 1.10 & | 1.10   |
| FF1/CLK (fdef1a15)               | 0.00   | 1.10 r |
| FF1/Q (fdef1a15)                 | 0.40 & | 1.50 f |
| U2/Y (buf1a27)                   | 0.05 & | 1.55 f |
| U3/Y (buf1a27)                   | 0.05 & | 1.60 f |
| FF2/D (fdef1a15)                 | 0.01 & | 1.61 f |
| data arrival time                |        | 1.61   |
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 1.00 & | 1.00   |
| FF2/CLK (fdef1a15)               |        | 1.00 r |
| library hold time                | 0.10   | 1.10   |
| data required time               |        | 1.10   |
| data required time               |        | 1.10   |
| data arrival time                |        | -1.61  |
| slack (MET)                      | 0.51   |        |

2-50

the calculated clock skew of 0.10 ns causes a - larger positive slack

the clock network is - calculated by PT (Clock is propagated) - Source latency is supplied by user

Answers:

## Test For Understanding : Clock Constraints 2/2



The clock uncertainty:

Should contain skew.    Should NOT contain skew.

The clock network latency + clock source latency constraint is:

Can't tell from report.    1.10ns.

| Point                       | Incr   | Path   |
|-----------------------------|--------|--------|
| clock Clk (rise edge)       | 0.00   | 0.00   |
| clock network delay (ideal) | 1.10   | 1.10   |
| FF1/CLK (fdef1a15)          | 0.00   | 1.10 r |
| FF1/Q (fdef1a15)            | 0.50 & | 1.60 r |
| U2/Y (buf1a27)              | 0.11 & | 1.71 r |
| U3/Y (buf1a27)              | 0.11 & | 1.82 r |
| FF2/D (fdef1a15)            | 0.05 & | 1.87 r |
| data arrival time           |        | 1.87   |
| clock Clk (rise edge)       | 4.00   | 4.00   |
| clock network delay (ideal) | 1.10   | 5.10   |
| clock uncertainty           | -0.80  | 4.30   |
| FF2/CLK (fdef1a15)          |        | 4.30 r |
| library setup time          | -0.21  | 4.09   |
| data required time          |        | 4.09   |
| data required time          |        | 4.09   |
| data arrival time           |        | -1.87  |
| slack (MET)                 |        | 2.22   |

2-51

clocks.

the clock network latency and clock source latency constraint is – 1.10 ns. Note that the same clock network delay is applied to the launch and capture clock edges. This is what happens for ideal

the clock uncertainty – should contain skew

Answers:

# Test For Understanding : Input Delay Constraints



Explain:

Where must the external clock latency be included?

Why is there no “&” designation for the input external delay?

| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 0.00   | 0.00   |
| input external delay             | 2.60   | 2.60 r |
| A (in)                           | 0.00   | 2.60 r |
| U2/Y (buf1a27)                   | 0.11 & | 2.71 r |
| U3/Y (buf1a27)                   | 0.11 & | 2.82 r |
| FF2/D (fdef1a15)                 | 0.05 & | 2.87 r |
| data arrival time                |        | 2.87   |
| clock Clk (rise edge)            | 4.00   | 4.00   |
| clock network delay (propagated) | 1.00 & | 5.00   |
| FF2/CLK (fdef1a15)               |        | 5.00 r |
| library setup time               | -0.21  | 4.79   |
| data required time               |        | 4.79   |
| data arrival time                |        | -2.87  |
| slack (MET)                      |        | 1.92   |

2-52

Why is there no “&” designation for the input external delay – the input delay is a constraint – not included or derived from SP&F

where MUST the external clock latency be included – must be represented as part of the input delay constraint. The clock network latency is zero so it has not been represented there.

Answers:

## Test For Understanding : Output Delay Constraints 1/2



Circle the:  
Output delay constraint.  
Output port name.  
External end point clock.

| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 1.10 & | 1.10   |
| FF1/CLK (fdef1a15)               | 0.00   | 1.10 r |
| FF1/Q (fdef1a15)                 | 0.50 & | 1.60 r |
| U2/Y (buf1a27)                   | 0.11 & | 1.71 r |
| U3/Y (buf1a27)                   | 0.11 & | 1.82 r |
| M (out)                          | 0.05 & | 1.87 r |
| data arrival time                |        | 1.87   |
| clock Clk (rise edge)            | 4.00   | 4.00   |
| clock network delay (propagated) | 0.00   | 4.00   |
| output external delay            | -0.40  | 3.60   |
| data required time               |        | 3.60   |
| data required time               |        | 3.60   |
| data arrival time                |        | -1.87  |
| slack (MET)                      |        | 1.73   |

2- 53

External end point clock - Clk  
Output port name - M  
Output delay constraint - -0.40

ANSWERS:

## Test For Understanding : Output Delay Constraints 2/2



Explain:

- Why is there no “library setup time” in this report?
- Where must the external clock latency be included?
- What does the 0.05 incremental delay represent?

| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 1.10 & | 1.10   |
| FF1/CLK (fdef1a15)               | 0.00   | 1.10 r |
| FF1/Q (fdef1a15)                 | 0.50 & | 1.60 r |
| U2/Y (buf1a27)                   | 0.11 & | 1.71 r |
| U3/Y (buf1a27)                   | 0.11 & | 1.82 r |
| M (out)                          | 0.05 & | 1.87 r |
| data arrival time                |        | 1.87   |
|                                  |        |        |
| clock Clk (rise edge)            | 4.00   | 4.00   |
| clock network delay (propagated) | 0.00   | 4.00   |
| output external delay            | -0.40  | 3.60   |
| data required time               |        | 3.60   |
|                                  |        |        |
| data required time               |        | 3.60   |
| data arrival time                |        | -1.87  |
|                                  |        |        |
| slack (MET)                      |        | 1.73   |

2-54

output port.

What does the 0.05 incremental delay represent – the final net delay from the last buffer to the

constraint

network delay is zero the clock latency can only be represented as part of the output delay where must the external clock latency be included – same answer as before, because the clock

look this timing up in a library – it must come from constraints supplied by the user. This timing path. There is no capture flip flop and thus no library setup time. PrimeTime cannot why is there no ‘library setup time’ in this report – the output delay constraint is the setup/hold for

Answers:

## Checking Constraints Applied on Ports

```
pt_shell> report_port -input_delay -output_delay pad[0] ← inout port
 Input Delay
 Min Max Related Related
Input Port Rise Fall Rise Fall Clock Pin

pad[0] 2.00 2.00 8.00 8.00 PCI_CLK --
 Output Delay
 Min Max Related Related
Output Port Rise Fall Rise Fall Clock Pin

pad[0] -1.00 -1.00 4.00 4.00 PCI_CLK --
```

2-55

This page was intentionally left blank

# Agenda

DAY

1

i Introduction to STA in PrimeTime

1 STA Concepts and Flow in PrimeTime



2 Methodology: Qualifying Constraints



3 Methodology: Generating Reports



## Objectives



**After completing this lecture, you should be able to:**

- Generate summary information for violations sorted by slack, timing check or clock group
- Generate detailed information for cell, net delays and of the paths of interest
- Get explanation for cell and net delays reported

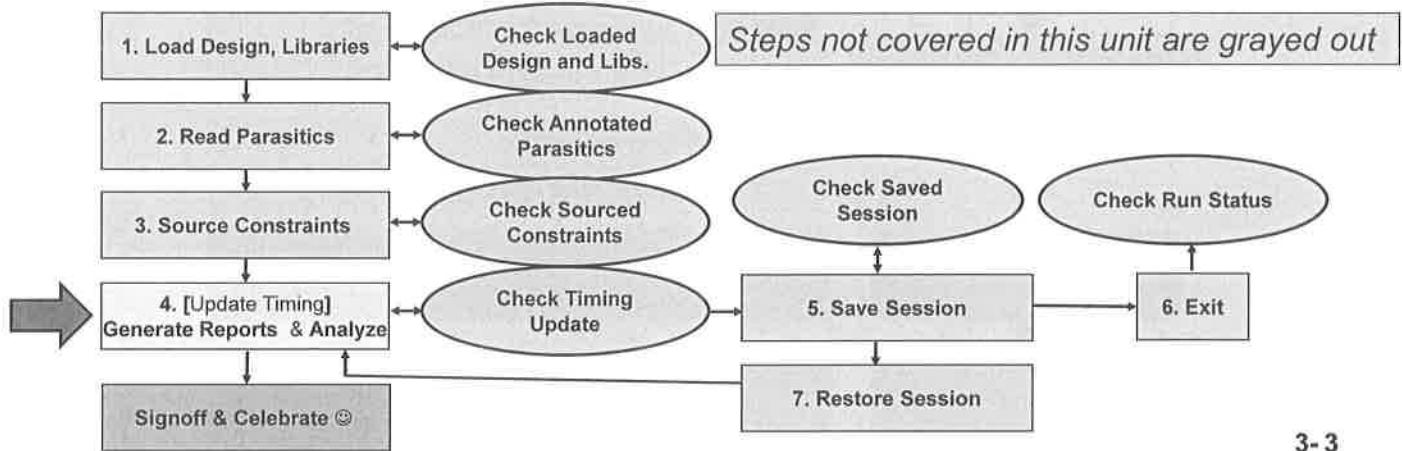
3-2

# Timing Analysis Flow in PrimeTime – Generating Reports

- STA flow is divided into several steps

- Steps 1-3 read data and
- Analysis begins at Step-4

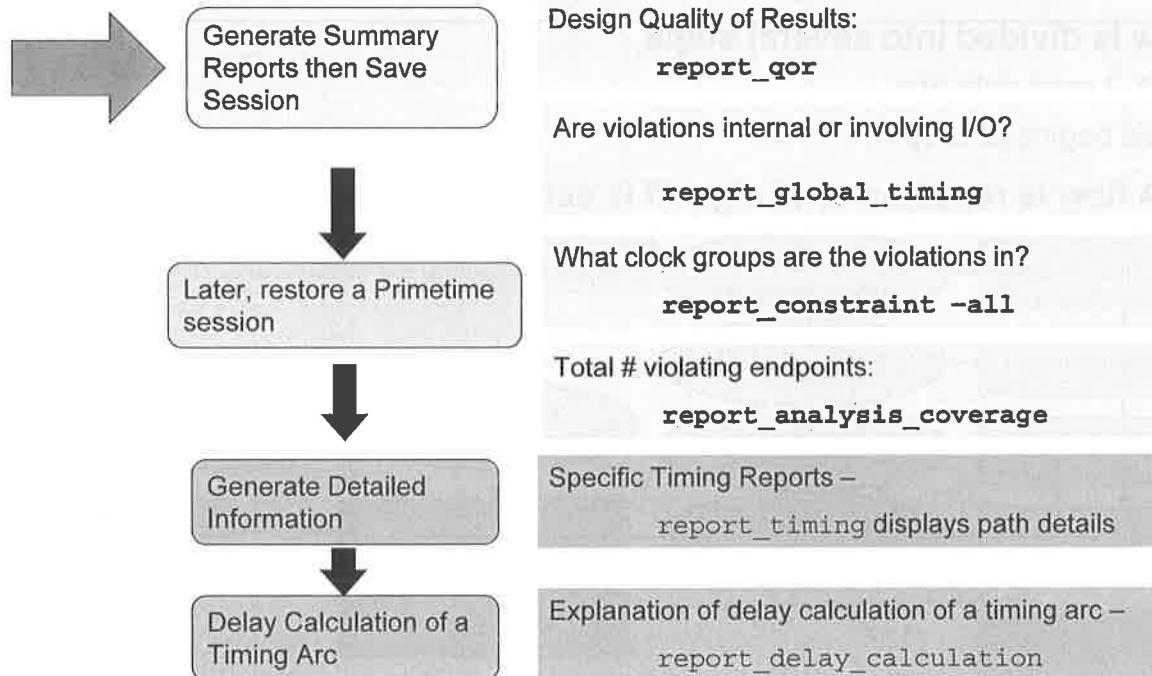
- The STA flow is repeated until signoff is achieved



3-3

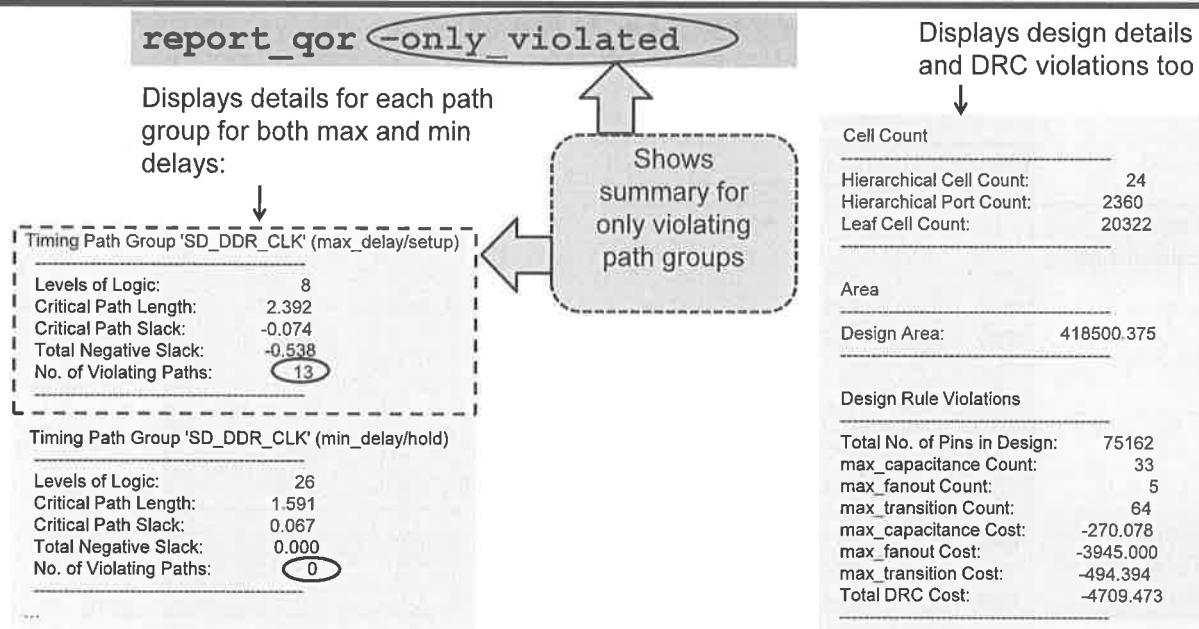
STA: Static Timing Analysis

## Generating Reports Methodology – Summary Reports



3-4

## Quality of Design: report\_qor



Two violations to the same endpoint but in different path groups will be reported as two different violations **3-5**

To report the maximum or minimum delay, use the report\_qor command with the -max or -min option

## Where are the Violations? : report\_global\_timing

Are violations internal or in the I/O?

report\_global\_timing

Four categories

Setup violations -- All groups

|     | Total   | reg->reg | reg->out | in->reg | in->out |
|-----|---------|----------|----------|---------|---------|
| WNS | -1.457  | -1.457   | -1.157   | 0.000   | 0.000   |
| TNS | -15.859 | -12.040  | -3.819   | 0.000   | 0.000   |
| NUM | 28      | 19       | 9        | 0       | 0       |

Hold violations -- All groups

|     | Total   | reg->reg | reg->out | in->reg | in->out |
|-----|---------|----------|----------|---------|---------|
| WNS | -0.437  | -0.429   | -0.370   | -0.236  | -0.437  |
| TNS | -43.266 | -31.155  | -3.636   | -2.577  | -5.898  |
| NUM | 196     | 148      | 14       | 18      | 16      |

By default, each violating endpoint is only counted one time and only one worst slack at each violating endpoint contributes to the total negative slack ( [Refer to Solvnet Article :1868374](#) )

3-6

## Violations Sorted By Clock : report\_constraint -all

Some reasons for organizing violations by clock domain:

1. Some clocks may be more critical than others
2. Clock slew or a "known" issue with a particular clock may be causing multiple violations

Lists all violating endpoints sorted by slack within each path group. Path groups are displayed in the alphabetical order

| pt_shell> report_constraint -all_violators -max_delay |        |            |
|-------------------------------------------------------|--------|------------|
| max_delay/setup ('SDRAM_CLK' group)                   |        |            |
| Endpoint                                              | Slack  |            |
| I_ORCA_TOP/I_SDRAM_IF/out_control_reg[15]/D           | -0.344 | (VIOLATED) |
| I_ORCA_TOP/I_SDRAM_IF/DQ_in_1_reg[6]/D                | -0.133 | (VIOLATED) |

| max_delay/setup ('SYS_2x_CLK' group)        |        |            |
|---------------------------------------------|--------|------------|
| Endpoint                                    | Slack  |            |
| I_ORCA_TOP/I_RISC_CORE/I_ALU/Zro_Flag_reg/D | -0.270 | (VIOLATED) |
| I_ORCA_TOP/I_RISC_CORE/PCint_reg[3]/D       | -0.266 | (VIOLATED) |

## Checks Exercised: report\_analysis\_coverage

Library  
Timing Checks

From  
set\_output\_delay

### report\_analysis\_coverage

| Type of Check   | Total | Met         | Violated  | Untested     |
|-----------------|-------|-------------|-----------|--------------|
| setup           | 9629  | 3500 ( 36%) | 88 ( 1%)  | 6041 ( 63%)  |
| hold            | 9629  | 3588 ( 37%) | 0 ( 0%)   | 6041 ( 63%)  |
| recovery        | 1316  | 1210 ( 92%) | 0 ( 0%)   | 106 ( 8%)    |
| removal         | 1316  | 1210 ( 92%) | 0 ( 0%)   | 106 ( 8%)    |
| min_period      | 20    | 10 ( 100%)  | 0 ( 0%)   | 0 ( 0%)      |
| min_pulse_width | 7273  | 6290 ( 82%) | 0 ( 0%)   | 1316 ( 18%)  |
| out_setup       | 68    | 33 ( 48%)   | 33 ( 49%) | 2 ( 3%)      |
| out_hold        | 68    | 66 ( 97%)   | 0 ( 0%)   | 2 ( 3%)      |
| All Checks      | 2931  | 1738 ( 59%) | 121 ( 0%) | 13614 ( 46%) |



You want details for these violations, sorted by slack.

3-8

- Reports library defined timing checks
- Also reports output setup and hold constraints (out\_setup and out\_hold)
- User defined constraints e.g. set\_max\_delay and set\_min\_delay are not reported
- Data to Data checks (set\_data\_check) are not reported

## Listing All Pins Having Setup Violations

```
report_analysis_coverage -status violated -check setup
```

| Type of Check | Total | Met         | Violated | Untested    |
|---------------|-------|-------------|----------|-------------|
| setup         | 9629  | 3500 ( 36%) | 88 ( 1%) | 6041 ( 63%) |
| All Checks    | 9629  | 3500 ( 36%) | 88 ( 1%) | 6041 ( 63%) |

| Constrained Pin                       | Related Pin | Check Type | Slack  |
|---------------------------------------|-------------|------------|--------|
| I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]/D | CP(rise)    | setup      | -0.719 |
| I_ORCA_TOP/I_BLENDER/s4_op2_reg[30]/D | CP(rise)    | setup      | -0.654 |
| I_ORCA_TOP/I_BLENDER/s4_op1_reg[31]/D | CP(rise)    | setup      | -0.483 |
| I_ORCA_TOP/I_BLENDER/s4_op2_reg[15]/D | CP(rise)    | setup      | -0.469 |
| I_ORCA_TOP/I_BLENDER/s4_op1_reg[30]/D | CP(rise)    | setup      | -0.374 |

Organized by slack!

Note: This command reports the number of violating END POINTS, not the number of VIOLATIONS – there could be multiple violations per endpoint!!

3-9

The above report is organized by slack – even if you include several timing checks together. For example, the command below will list all violations for setup and hold organized strictly by slack.

```
report_analysis_coverage -status violated -check "setup hold"
```

If you would like to sort by timing check first and then by slack, add the following switch.

```
report_analysis_coverage -status violated -check "setup hold" -sort check_type
```

## Examples Using report\_analysis\_coverage

1. Listing output port names having *untested out\_setup* checks

```
→ report_analysis_coverage -status untested -check "out_setup"
```

| Constrained Pin | Related Pin | Clock  | Check Type | Slack    | Reason   |
|-----------------|-------------|--------|------------|----------|----------|
| clock2out       |             | clock2 | out_setup  | untested | no_paths |
| clockout        |             | clock1 | out_setup  | untested | no_paths |

2. Listing output port names having *untested out\_setup* checks *excluding untested no\_paths*

```
→ report_analysis_coverage -status untested -check "out_setup" \
 -exclude_untested no_paths
```

| Type of Check | Total | Met       | Violated  | Untested |
|---------------|-------|-----------|-----------|----------|
| out_setup     | 66    | 33 ( 50%) | 33 ( 50%) | 0 ( 0%)  |
| All Checks    | 66    | 33 ( 50%) | 33 ( 50%) | 0 ( 0%)  |

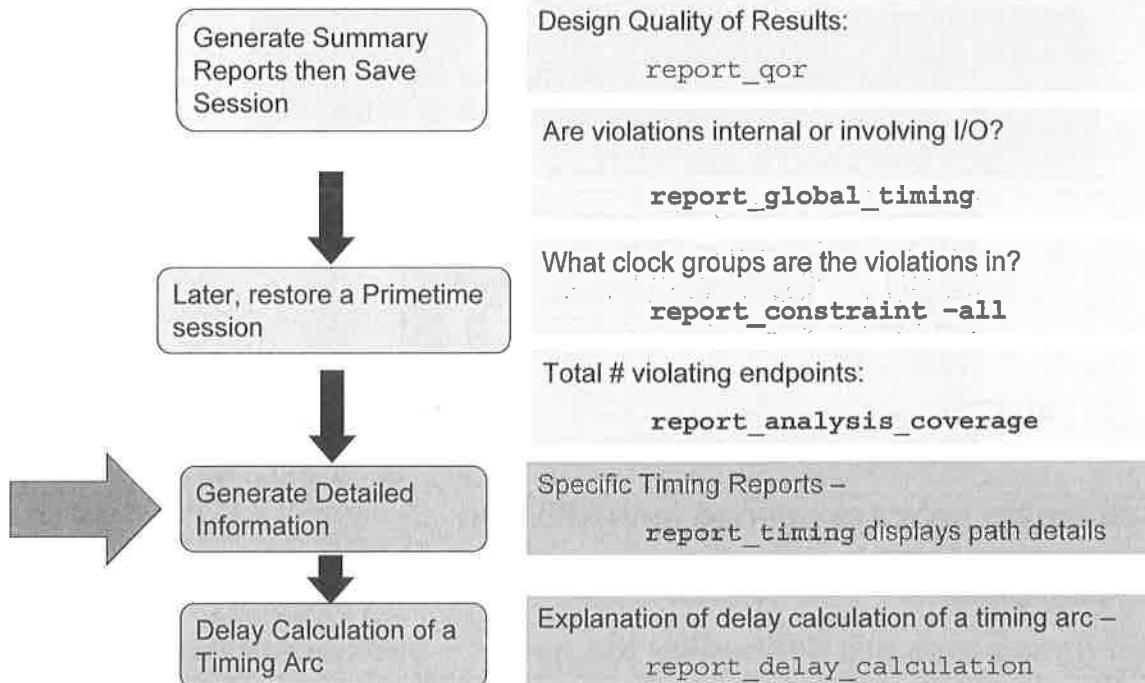
3-10

```
pt_shell> report_analysis_coverage -help
```

Usage:

```
report_analysis_coverage # Show coverage of timing checks
 [-status_details status_list]
 (Details for status of 'untested', 'violated', and/or 'met')
 [-check_type check_type_list]
 (Checks to include: 'setup', 'hold', 'recovery', 'removal', etc.)
 [-exclude_untested untested_reason_list]
 (Checks with untested status to exclude by reason: 'unknown',
 'no_clock', etc.)
 [-sort_by sort_method]
 (Method to sort the detailed list:
 Values: name, slack, check_type, check)
 [-significant_digits digits]
 (Number of digits to display: Range: 0 to 13)
 [-nosplit]
 (Do not split lines when columns overflow)
```

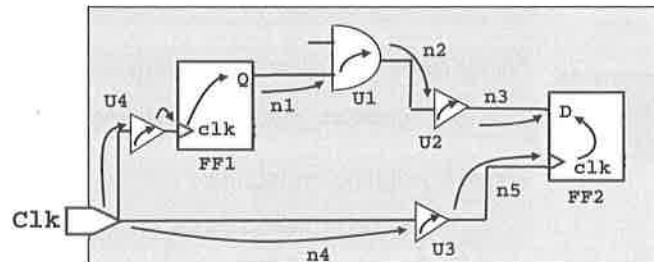
## Generating Reports Methodology – Detailed Reports



3-11

## Timing Path Details: Built From Timing Arcs

- PrimeTime assembles timing arcs into paths
- `report_timing` displays the arcs of paths



- Cell timing arcs are defined in the library:
  - Propagation Delay
  - Timing check
- Net timing arcs are defined by the netlist + back-annotated data

3-12

# Common Types of Timing Arcs in Libraries

Timing arc “unateness” specifies how the output changes relative to input transitions



## positive\_unate

Rising input -> rising output  
Falling input -> falling output



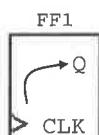
## positive\_unate

and



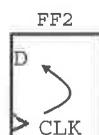
## negative\_unate

Rising input -> falling output  
Falling input -> rising output



## rising\_edge

Rising clock edge causes a transition on the output pin



## hold\_clk\_rise

## setup\_clk\_rise

Rising edge of the related (clock) pin used for setup and hold checks on clocked elements

Output transition depends on the state of other inputs as well

## Additional examples for “non unate” timing arcs:

- XNOR
- Select-to-output of MUX
- Enable-to-output of TRISTATE driver

3-13

Unate: the output transition is predictable based on the input transition causing it.

Positive unate is when the output changes in the same direction as the input that caused the change (for example, an AND gate). So, for an AND gate with A@0, B@1, when A changes 0 to 1, that is, a positive transition, the AND gate output will change 0->1 also. Same for a buffer, same for an OR gate.

Negative unate cells have the output switching in the opposite direction to the input which effected the change. Negative unate is the inversion of the input (for example, an inverter).

Non-unate: the output transition is not predictable; that is, the cell outputs could go either way depending on the other inputs (for example, an XOR or a MUX).

## Rise/Fall Transitions Along a Timing Path

| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 1.10   | 1.10   |
| FF1/CLK (fdef1a15)               | 0.00   | 1.10 r |
| FF1/Q (fdef1a15)                 | 0.40 & | 1.50 f |
| U2/Y (buf1a27)                   | 0.05 & | 1.55 f |
| U3/Y (buf1a27)                   | 0.05 & | 1.60 f |
| FF2/D (fdef1a15)                 | 0.01 & | 1.61 f |
| data arrival time                |        | 1.61   |
|                                  |        |        |
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 1.00   | 1.00   |
| FF2/CLK (fdef1a15)               |        | 1.00 r |
| library hold time                | 0.10   | 1.10   |
| data required time               |        | 1.10   |
|                                  |        |        |
| data required time               |        | 1.10   |
| data arrival time                |        | -1.61  |
|                                  |        |        |
| slack (MET)                      |        | 0.51   |



- FF1 rising edge flop => "r" at FF1/CLK
- FF1/Q can transition from 0->1 or 1->0
- Worst hold slack is reported with a "f" transition at FF1/Q

3-14

# Generating Timing Reports Specifying Endpoint Transition

`report_timing -to FF2/D -delay max_fall`

Startpoint: FF1 (rising edge-triggered flip-flop clocked by Clk)  
 Endpoint: FF2 (rising edge-triggered flip-flop clocked by Clk)  
 Path Group: Clk  
 Path Type: max

| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 1.10   | 1.10   |
| FF1/CLK (fdef1a15)               | 0.00   | 1.10 r |
| FF1/Q (fdef1a15)                 | 0.40 & | 1.50 f |
| U2/Y (bufla27)                   | 0.05 & | 1.55 f |
| U3/Y (bufla27)                   | 0.05 & | 1.60 f |
| FF2/D (fdef1a15)                 | 0.01 & | 1.61 f |
| data arrival time                |        | 1.61   |
|                                  |        |        |
| clock Clk (rise edge)            | 4.00   | 4.00   |
| clock network delay (propagated) | 1.00   | 5.00   |
| FF2/CLK (fdef1a15)               |        | 5.00 r |
| library setup time               | -0.15  | 4.85   |
| data required time               |        | 4.85   |
|                                  |        |        |
| data required time               |        | 4.85   |
| data arrival time                |        | -1.61  |
|                                  |        |        |
| slack (MET)                      |        | 3.24   |

4 options:

`-delay max_fall`  
`-delay max_rise`  
`-delay min_fall`  
`-delay min_rise`

`report_timing -to FF2/D`  
 generates a setup time report  
 having the worst slack [the  
 endpoint could be "r" = rising]

3-15

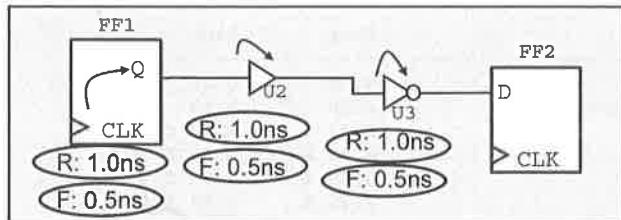
`pt_shell> report_timing -help`

```
[-delay_type delay_type]
(Type of path delay:
Values: max, min, min_max, max_rise,
max_fall, min_rise, min_fall)
```

## Data Arrival Time: Edge Sensitivity *negative\_unate*

R: Cell delay when output pin transitions from 0→1

F: Cell delay when output pin transitions from 1→0



The longest data arrival time is

3.0ns       2.5ns

The shortest data arrival time is

2.0ns       1.5ns

The total # of possible data arrival times for setup is

1       2       4

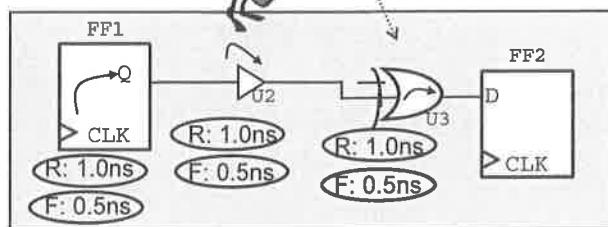
FF1 → U2 → U3  
R      R      F  
F      F      R

3-16

## Data Arrival Time: Edge Sensitivity *non unate*

R: Cell delay when output pin transitions from 0→1

F: Cell delay when output pin transitions from 1→0



The longest data arrival time

3.0ns

2.5ns

FF1 → U2 → U3

R R R

R R F

F F F

F F R

The shortest data arrival time

2.0ns

1.5ns

The total # of possible data arrival times for setup is

1

2

4

3-17

## Reporting Library Arcs: Inverter

```
pt_shell> report_lib -timing_arcs cb13fs120_tsmc_max inv0d1
 Arc Arc Pins
Lib Cell Attributes # Type/Sense From To

inv0d1 0 negative_unate I ZN
```

Diagram showing the command output with annotations:

- Library name: points to "cb13fs120\_tsmc\_max"
- Reference /lib\_cell name: points to "inv0d1"

Use the following commands to  
list all loaded libraries in  
PrimeTime memory

- list\_libs
- list\_libraries
- get\_libs

3-18

```
pt_shell> report_lib -help
Usage:
report_lib # Report library information
[-timing_arcs] (Show timing arc data for lib_cells)
[-nosplit] (Don't split lines if column overflows)
library (Name of a library in memory)
[lib_cell_list] (Show only these lib_cells)
```

If you have only the cell instance name, but not the library or reference name, use the following.

```
pt_shell> report_cell U7
Cell Reference Library Area
Attributes

U7 inv0d1 cb13fs120_tsmc_max
 0.7500

Total 1 cells 0.7500
```

## Reporting Library Arcs: Flip-Flop

| Lib    | Cell | Attributes | Arc # | Type/Sense             | From | To  | Arc Pins |
|--------|------|------------|-------|------------------------|------|-----|----------|
| sdcrql | s    |            | 0     | clock_pulse_width_high | CP   | CP  |          |
|        |      |            | 1     | clock_pulse_width_low  | CP   | CP  |          |
|        |      |            | 2     | recovery_rise_clk_rise | CP   | CDN |          |
|        |      |            | 3     | removal_rise_clk_rise  | CP   | CDN |          |
|        |      |            | 4     | hold_clk_rise          | CP   | D   |          |
|        |      |            | 5     | setup_clk_rise         | CP   | D   |          |
|        |      |            | 6     | hold_clk_rise          | CP   | SC  |          |
|        |      |            | 7     | setup_clk_rise         | CP   | SC  |          |
|        |      |            | 8     | hold_clk_rise          | CP   | SD  |          |
|        |      |            | 9     | setup_clk_rise         | CP   | SD  |          |
|        |      |            | 10    | rising_edge            | CP   | Q   |          |
|        |      |            | 11    | clock_pulse_width_low  | CDN  | CDN |          |
|        |      |            | 12    | clear_low              | CDN  | Q   |          |

3-19

The function of each pin is as follows:

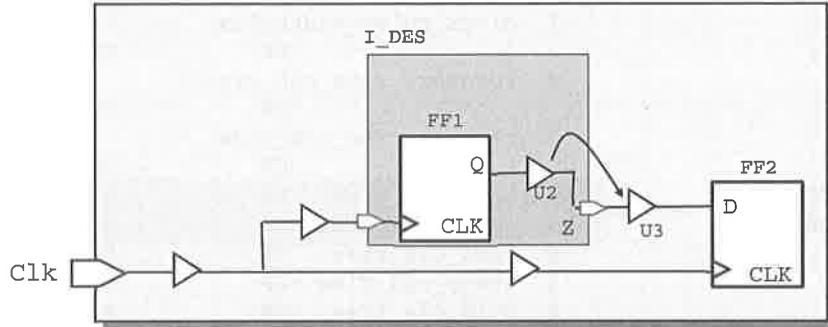
|            |                                  |
|------------|----------------------------------|
| <b>CDN</b> | Asynchronous clear, asserted low |
| <b>CP</b>  | Clock                            |
| <b>D</b>   | Data input                       |
| <b>SC</b>  | Scan enable input                |
| <b>SD</b>  | Scan chain input                 |
| <b>Q</b>   | Output pin                       |

## Hierarchy in a Timing Path

Hierarchy does not “block” STA.

Hierarchy does not “break” net timing arcs.

The hierarchy (pin I\_DES/Z) in path is reported.



The delay on the hierarchical pins in a timing report is 0.00 !!

3-20

Delay calculation in PrimeTime is done in stages. A stage consists of a driver, a receiver and the parasitic network between them.

Delay calculation is always performed between two physical pins in a design.

## Identify Hierarchy in a Timing Report

Hierarchy pin name → Design name

Look for zero delay  
Net + cell delay

| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 1.10   | 1.10   |
| I_DES/FF1/CLK (fdef1a15)         | 0.00   | 1.10 r |
| I_DES/FF1/Q (fdef1a15)           | 0.40 & | 1.50 f |
| I_DES/U2/Y (buf1a27)             | 0.05 & | 1.55 r |
| I_DES/Z (MYDES) ← Design name    | 0.00 & | 1.55 f |
| U3/Y (buf1a27)                   | 0.05 & | 1.60 f |
| FF2/D (fdef1a15)                 | 0.01 & | 1.61 f |
| data arrival time                |        | 1.61   |
|                                  |        |        |
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 1.00   | 1.00   |
| FF2/CLK (fdef1a15)               |        | 1.00 r |
| library hold time                | 0.10   | 1.10   |
| data required time               |        | 1.10   |
|                                  |        |        |
| data required time               |        | 1.10   |
| data arrival time                |        | -1.61  |
|                                  |        |        |
| slack (MET)                      | 0.51   |        |

report\_timing -include\_hierarchical\_pins

3-21

The output of the `report_timing` command does not show hierarchical pins by default. To show the hierarchical pins, use the `report_timing` command with the new `-include_hierarchical_pins` option. This behavior is consistent with the `-include_hierarchical_pins` option of the `get_timing_paths` command.

## Default Behavior of report\_timing

The command `report_timing`, by default generates one report with the worst slack in the entire design for setup time.

*Default options used are shown in the report header*

```
pt_shell> report_timing

Report : timing
-path_type full
-delay_type max
-max_paths 1
-sort_by slack
Design : ORCA
Version: O-2018.06

```

3-22

## report\_timing for each *path group*

To generate one report  
with the worst slack  
for each path group  
for setup time:

```
pt_shell> report_timing -group [get_path_group *]

Report : timing
-path_type full
-delay_type max
-max_paths 1
-group **async_default** **clock_gating_default** **default** PCI_CLK
SDRAM_CLK SD_DDR_CLK SD_DDR_CLKn SYS_2x_CLK SYS_CLK
-sort_by slack
Design : ORCA

```

One timing report per path group  
is generated

3- 23

report\_timing -group [get\_path\_group \*] is equivalent to report\_timing -sort\_by group

Path groups are created:

1. one for each capturing clock in the design (by default)
2. PT predefined path groups as:  
`**async_default**,  
**clock_gating_default**,  
**default**,  
none groups`
3. User defined path groups using `group_path`

## The nworst vs. max\_paths Options of report\_timing

**-nworst** : Consider these many paths per endpoint

- Defaults to 1
- If `-nworst > 1`
  - `max_paths = nworst`
  - `slack_lesser_than = 0`

Only violating paths are reported if using  
`nworst | max_paths > 0`

```
pt_shell> report_timing -nworst 3
```

```

Report : timing
-path_type full
-delay_type max
-nworst 3
-slack_lesser_than 0.00
-max_paths 3
-sort_by slack
Design : ORCA
Version: O-2018.06

```

```
pt_shell> report_timing -max_paths 4
```

```

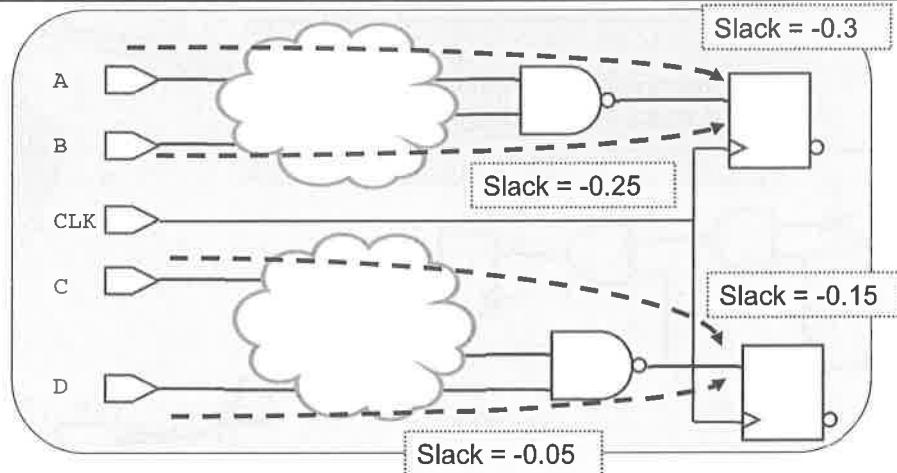
Report : timing
-path_type full
-delay_type max
-slack_lesser_than 0.00
-max_paths 4
-sort_by slack
Design : ORCA
Version: O-2018.06

```

**-max\_paths** : Display these many paths in the design

- Defaults to the nworst setting
- If `-max_paths > 1`
  - `slack_lesser_than = 0`

## Exercise on nworst vs. max\_Paths options



max\_paths > 1:  
▪ nworst = 1  
▪ slack\_less\_than = 0

report\_timing -max\_paths 2

report\_timing -nworst 2

nworst>1:  
▪ max\_paths = nworst  
▪ slack\_less\_than = 0

?

Circle the reported slacks

3- 25

- There are two endpoints in this design.
- Shown are the two worst slack values for each endpoint.

Answer:

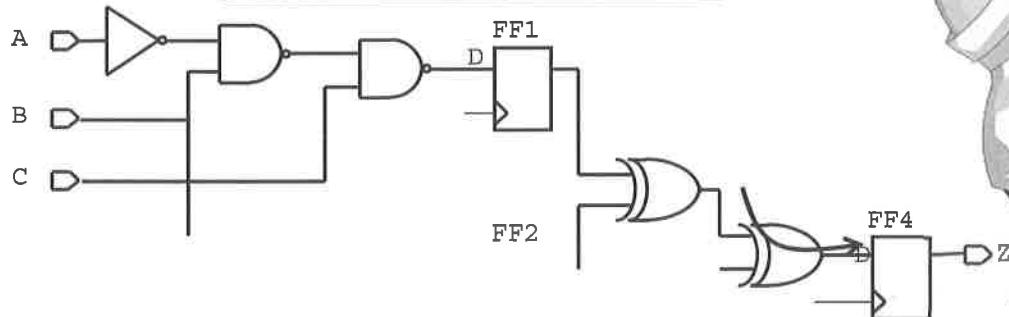
circle the reported slacks

-max\_paths 2 → -0.3 and -0.15; that is, consider the single worst path to each endpoint; from that collection, display the worst two

-nworst 2 → -0.3 and -0.25; that is, consider the two worst paths to each endpoint; starting with the worst endpoint, display the worst two paths.

## Reporting Multiple Paths to a Single Endpoint

FF4 has a violation. Report 3 timing paths to this end point for debugging.



`report_timing -to FF4/D -max_paths 3`



1 path is reported

`report_timing -to FF4/D -nworst 3`



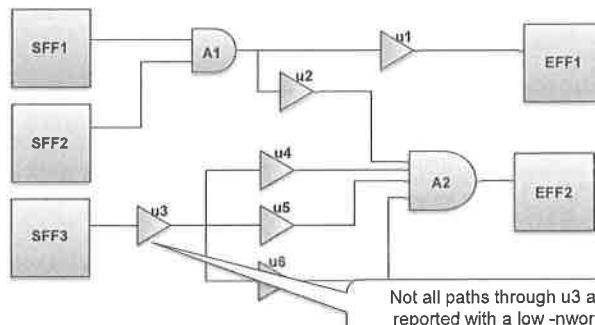
3 paths are reported

3-26

## Cover Design Reporting (`report_timing -cover_design`)

Need: To report the worst case path through every violating pin in the design

Assume all violating pins



| Path order           |
|----------------------|
| P1: SFF3 – u5 – EFF2 |
| P2: SFF3 – u4 – EFF2 |
| P3: SFF1 – u2 – EFF2 |
| P4: SFF3 – u6 – EFF2 |
| P5: SFF2 – u2 – EFF2 |
| P6: SFF1 – u1 – EFF1 |
| P7: SFF2 – u1 – EFF1 |

Order of slack  
↓

```
pt_shell> sizeof_collection [get_timing_paths -cover_design]
6
pt_shell> sizeof_collection [get_timing_paths -max_paths 6 -nworst 5]
6
```

Eliminates the need for guessing the correct max\_paths/nworst settings!

3- 27

Lets look at an example. Here we have 7 paths between, the 3 startpoint FFs and two endpoint FFs. The assumption here is all the paths have violating slack. The path order for these paths is as shown in the table.

For this path order, the `-cover_design` reports 6 paths which cover all the violating pins.

To use the `max_path`, `nworst` settings to get the same set of paths, you will have to use a high `nworst` value, 5 in this case. If the correct `max_paths`, `nworst` values are not used, some paths may be missed (lower settings) or additional paths may be included (higher settings)

## Reporting Summary Using report\_timing



You want the top 10 timing paths for hold captured by the falling edge of SDRAM\_CLK and reported as a summary.

```
report_timing -delay min -fall_to [get_clocks SDRAM_CLK] \
 -max 10 -path_end
```

| Endpoint                                         | Path Delay | Path Required | Slack  |
|--------------------------------------------------|------------|---------------|--------|
| I_ORCA_TOP/I_SDRAM_IF/DQ_in_1_reg[15]/D (sdnfb1) | 4.644 f    | 4.837         | -0.192 |
| I_ORCA_TOP/I_SDRAM_IF/DQ_in_1_reg[14]/D (sdnfb1) | 4.721 f    | 4.837         | -0.116 |
| I_ORCA_TOP/I_SDRAM_IF/DQ_in_1_reg[9]/D (sdnfb1)  | 4.721 f    | 4.837         | -0.115 |

3-28

## Another “Summary” Report Using `report_timing`

Example Reporting a summary of top 10 violating endpoints for setup for the clock `SYS_2x_CLK`.

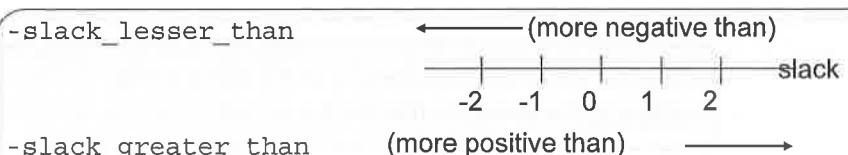
Displays the path startpoint, endpoint and slack

```
report_timing -path_summary -group SYS_2x_CLK -max_paths 10
```

| Startpoint                                                                 | Endpoint         | Slack         |
|----------------------------------------------------------------------------|------------------|---------------|
| u0_0/p0_iu0/r_reg_X_NERROR_CLK (SDFFARX2_HVT)                              | errorn (out)     | -12287598.000 |
| u_m/u_power_controller_top/p3_control_isolate_reg_q_reg/CLK (SDFFARX1_HVT) | data[21] (inout) | -3.113        |
| u_m/u_power_controller_top/p3_control_isolate_reg_q_reg/CLK (SDFFARX1_HVT) | sd[29] (inout)   | -3.106        |

## Filtering Paths Based by Slack Amount

report\_timing



```
report_timing <slack_greater_than -13 -slack_lesser_than -12> -max_paths 10 -path_type summary
```

```
Report : timing
-path_type summary
-delay_type max
-slack_lesser_than -12.00
-slack_greater_than -13.00
-max_paths 10
-sort_by slack
```

```
Design : top
```

```
Version: 0-2018.06
```

| Startpoint                                | Endpoint              | Slack  |
|-------------------------------------------|-----------------------|--------|
| eta/ain5_ff4/CP (SDFCNQD1BWP7T40P140EHVT) | data1_ff4/D (FD1QAFF) | -12.32 |
| eta/ain5_ff5/CP (SDFCNQD1BWP7T40P140EHVT) | data1_ff5/D (FD1QAFF) | -12.32 |
| eta/ain5_ff6/CP (SDFCNQD1BWP7T40P140EHVT) | data1_ff6/D (FD1QAFF) | -12.32 |

2011.12 default:  
slack\_lesser\_than 0

3-30

## Reporting Paths From, To, Through or Exclude

```
report_timing
 -from -rise_from -fall_from
 -to -rise_to -fall_to
 -through -rise_through -fall_through
 -exclude -rise_exclude -fall_exclude
```

```
report_timing -from A -through B -through C -to D
report_timing -from A -through {B C} -to D
report_timing -exclude [get_ports *]
```

Examples of  
Start points  
End points  
Through/Exclude points

Start: Non-clock Input ports; Clock pins or a clock object

End: Output ports; non-clock input pins of registers/latches, clock object

Through/Exclude: Any pin, port or net

Can also use the following "Shortcuts" for -to and -from:  
- Clock names (objects)  
- Register/latch cell names (not recommended!)

## Clock Ports Versus Clock Objects

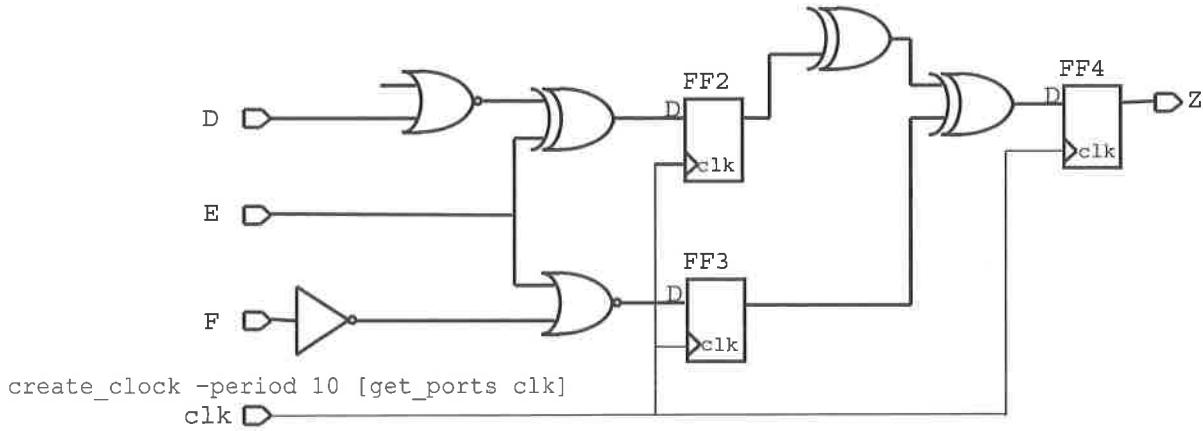
report\_timing -from [get\_ports clk] will return:

- Nothing – this is not a timing path.
- A single timing path from the clk port

report\_timing -from [get\_clocks clk] will return:

- Nothing – this is not a timing path.
- A single, worst timing path for setup launched by the clock clk.

Not a valid start point  
(port with a clock definition)



3-32

For more information on the clock network (e.g. clock skew, latency or transition times) for each clock domain individually or between clock domains, use the following command.

```
pt_shell> report_clock_timing -help
```

## Recommendation – Being Specific

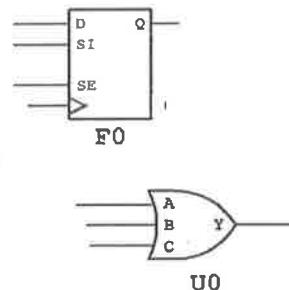


Use the specific pin names, not just the cell name.

Best runtime.

Get the intended timing paths.

```
report_timing -to (get_pins F0/D]
report_timing -through (get_pins U0/A]
```



Commands to report cell  
pin names:

```
report_cell -connections -verbose U0
get_pins -of_objects U0
report_lib -timing_arcs cb13fs120_tsmc_max or02d1
```

3-33

For further research, refer to SolvNet article “UITE-416 warning messages during report\_timing in U-2003.03”.

**Doc Id: 005471 Last Modified: 04/14/2003**

A few examples to find the pins of a cell.

```
pt_shell> report_cell -connections -verbose U7
Connections for cell 'U7':
 Reference: inv0d1
 Library: cb13fs120_tsmc_max
 Area: 0.75

 Input Pins Net Net Driver Pins Driver Pin Type

 I net_pad_en I_ORCA_TOP/I_PCI_CORE/pad_en_reg/Q
 Output Pin (sdcrq1)

 Output Pins Net Net Load Pins Load Pin Type

 ZN n43 U62/I Input Pin (inv0d1)
```

```
pt_shell> get_pins -of_objects U7
{ "U7/I", "U7/ZN" }
```

## Example: Specifying Capturing Clock Edge



A default timing report returns a path captured by the rising edge of CLK1.

You want the worst slack captured by the falling edge of CLK1.

When using clock objects, the rise and fall switches refer to the clock edge.

```
report_timing -fall_to [get_clocks CLK1]
```

↑  
Explicitly refer to a clock object named CLK1 (i.e. not a pin or a port).

3- 34

The above command will return a data path constrained by the falling edge of the clock `CLK1` with the worst slack for setup. The data path itself may have either a falling or rising transition at the end point.

For more information and practice with these types of commands, refer to the additional workshop:

**The Power of Tcl 3: Direct Access Through Collections and Attributes**



## Example: Specifying Net Name Along Timing Path



You want the worst slack through  
a bus. The net names are  
sd\_DQ\*.

Supported Wildcards :

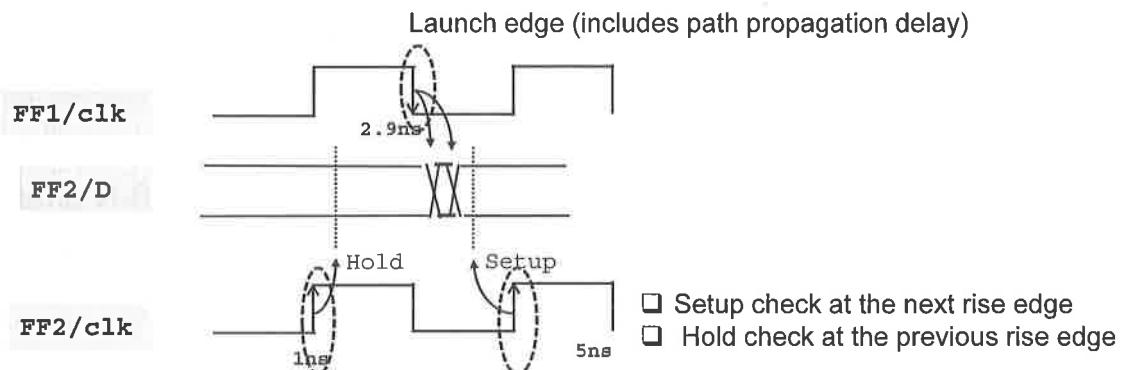
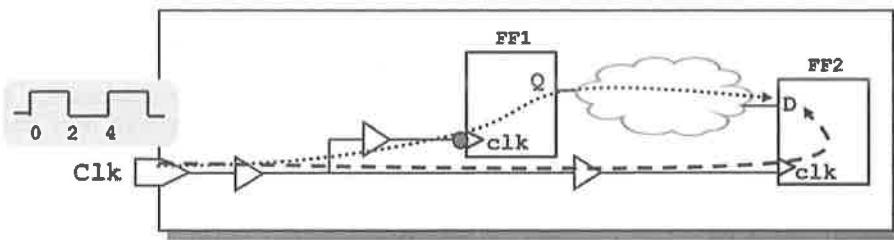
\* : matches any number of characters  
? : Matches exactly one character only

```
report_timing -through [get_nets sd_DQ*]
```

↑  
Represents 0-N characters

3- 35

## Half Cycle Path: Clock Edges Used For Setup and Hold



3-36

## Reporting a Half Cycle Path : Setup

```
report_timing -fall_from [get_clocks clk] -rise_to [get_clocks clk]
```

| Startpoint: FF1 (falling edge-triggered flip-flop clocked by Clk)<br>Endpoint: FF2 (rising edge-triggered flip-flop clocked by Clk)<br>Path Group: Clk<br>Path Type: max |        |        |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|--------|
| Point                                                                                                                                                                    | Incr   | Path   |
| clock Clk (fall edge)                                                                                                                                                    | 2.00   | 2.00   |
| clock network delay (propagated)                                                                                                                                         | 0.90 & | 2.90   |
| FF1/CLK (fdmf1a15)                                                                                                                                                       | 0.00   | 2.90 f |
| FF1/Q (fdmf1a15)                                                                                                                                                         | 0.50 & | 3.40 r |
| U2/Y (bufla27)                                                                                                                                                           | 0.11 & | 3.51 r |
| U3/Y (bufla27)                                                                                                                                                           | 0.11 & | 3.62 r |
| FF2/D (fdef1a15)                                                                                                                                                         | 0.05 & | 3.67 r |
| data arrival time                                                                                                                                                        |        | 3.67   |
| clock Clk (rise edge)                                                                                                                                                    | 4.00   | 4.00   |
| clock network delay (propagated)                                                                                                                                         | 1.00 & | 5.00   |
| FF2/CLK (fdef1a15)                                                                                                                                                       |        | 5.00 r |
| library setup time                                                                                                                                                       | -0.21  | 4.79   |
| data required time                                                                                                                                                       |        | 4.79   |
| data required time                                                                                                                                                       |        | 4.79   |
| data arrival time                                                                                                                                                        |        | -3.67  |
| slack (MET)                                                                                                                                                              |        | 1.12   |

Launch edge : fall  
Capture edge : rise

3-37

If a design has both positive and negative edge triggered flip flops, it's likely, that your design will have half cycle paths. A half-cycle path could be from a rising edge flip flop to a falling edge flip flop and vice versa.

## Reporting A Half Cycle Path : Hold

report\_timing -delay min -fall\_from ... -rise\_to ...

| Startpoint: FF1 (falling edge-triggered flip-flop clocked by Clk)<br>Endpoint: FF2 (rising edge-triggered flip-flop clocked by Clk)<br>Path Group: Clk<br>Path Type: min |        |        |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|--------|
| Point                                                                                                                                                                    | Incr   | Path   |
| clock Clk (fall edge)                                                                                                                                                    | 2.00   | 2.00   |
| clock network delay (propagated)                                                                                                                                         | 0.90 & | 2.90   |
| FF1/CLK (fdmfla15)                                                                                                                                                       | 0.00   | 2.90 f |
| FF1/Q (fdeflal15)                                                                                                                                                        | 0.40 & | 3.30 f |
| U2/Y (buflal27)                                                                                                                                                          | 0.05 & | 3.35 f |
| U3/Y (buflal27)                                                                                                                                                          | 0.05 & | 3.40 f |
| FF2/D (fdeflal15)                                                                                                                                                        | 0.01 & | 3.41 f |
| data arrival time                                                                                                                                                        |        | 3.41   |
| clock Clk (rise edge)                                                                                                                                                    | 0.00   | 0.00   |
| clock network delay (propagated)                                                                                                                                         | 1.00 & | 1.00   |
| FF2/CLK (fdeflal15)                                                                                                                                                      |        | 1.00 r |
| library hold time                                                                                                                                                        | 0.10   | 1.10   |
| data required time                                                                                                                                                       |        | 1.10   |
| data required time                                                                                                                                                       |        | 1.10   |
| data arrival time                                                                                                                                                        |        | -3.41  |
| slack (MET)                                                                                                                                                              |        | 2.31   |

Launch edge : fall  
Capture edge : rise

clock Clk (fall edge)

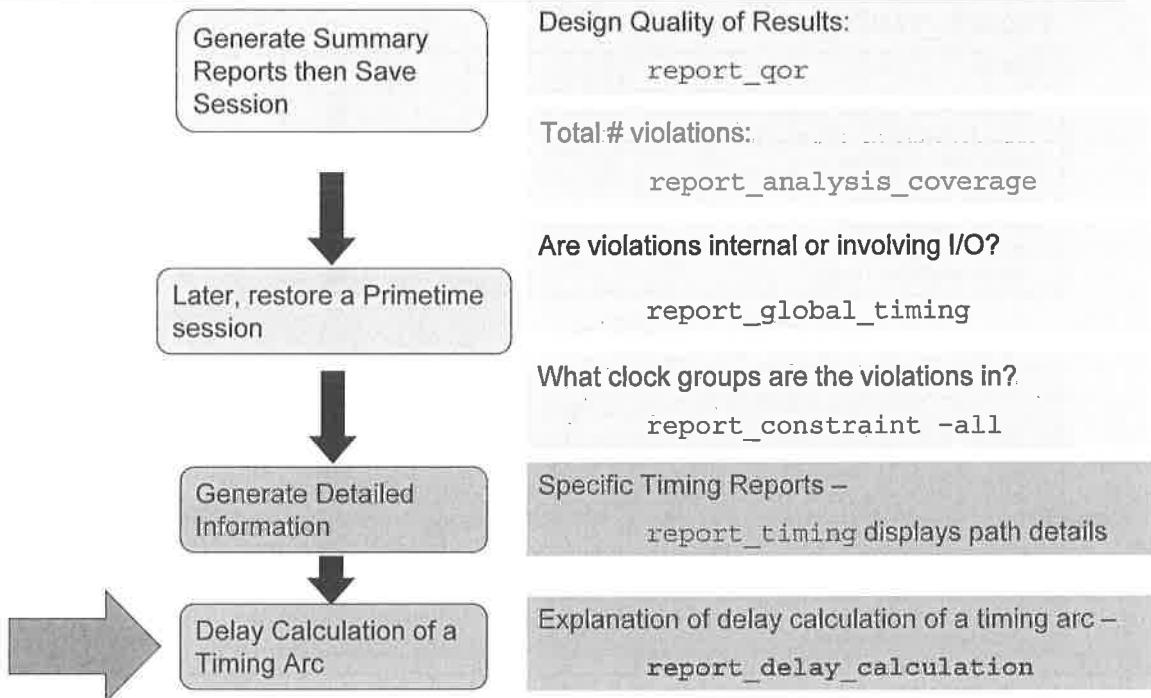
clock Clk (rise edge)

3-38

is 2 to 0

Answer: circle the clock edges – same old zero-cycle hold check – this time is falling to rising, so it

## Generating Reports Methodology – Delay Calculation



3- 39

## Including Input Pins in a Timing Report

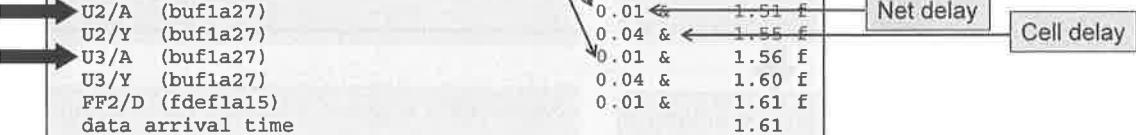
```
report_timing
```

| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 1.10   | 1.10   |
| FF1/CLK (fdef1a15)               | 0.00   | 1.10 r |
| FF1/Q (fdef1a15)                 | 0.40 & | 1.50 f |
| U2/Y (buf1a27)                   | 0.05 & | 1.55 f |
| U3/Y (buf1a27)                   | 0.05 & | 1.60 f |
| FF2/D (fdef1a15)                 | 0.01 & | 1.61 f |
| data arrival time                |        | 1.61   |

```
report_timing:-input_pins
```

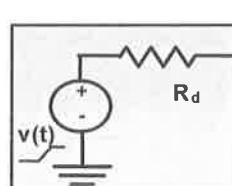
| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock Clk (rise edge)            | 0.00   | 0.00   |
| clock network delay (propagated) | 1.10   | 1.10   |
| FF1/CLK (fdef1a15)               | 0.00   | 1.10 r |
| FF1/Q (fdef1a15)                 | 0.40 & | 1.50 f |
| U2/A (buf1a27)                   | 0.01 & | 1.51 f |
| U2/Y (buf1a27)                   | 0.04 & | 1.55 f |
| U3/A (buf1a27)                   | 0.01 & | 1.56 f |
| U3/Y (buf1a27)                   | 0.04 & | 1.60 f |
| FF2/D (fdef1a15)                 | 0.01 & | 1.61 f |
| data arrival time                |        | 1.61   |

Input Pins

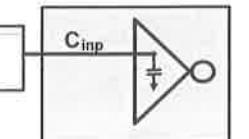
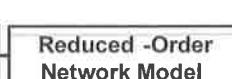


3-40

## NLDM Driver and Receiver Models – Limited Accuracy



NLDM Driver Model



NLDM Receiver Model

- Delay calculation can be performed on every timing arc of a cell
- Ramp voltage source  $V(t)$ , Fixed drive resistance ( $R_d$ )
- Accurate for most nets
- Limited accuracy for high impedance networks with strong drivers (RC-009)
- Cell receiver behavior is modeled as a single constant capacitance
- Doesn't model capacitance variation during transition (Miller Effect)

3- 41

NLDM: Non Linear Delay Model

## Cell Delay Calculation Reporting with NLDM library

```
report_delay_calculation -from I_ORCA_TOP/I_BLENDER/IU10961/B -to \
I_ORCA_TOP/I_BLENDER/IU10961/ZN ;# Delay from a cell input pin to its output pin
```

Main Library Units: 1ns 1pF 1kOhm

Library: 'tcbn65lpwc\_ccs'

Library name

Library Units: 1ns 1pF 1kOhm

Library Cell: 'GOAI21D1'

arc sense:

Positive or negative unate?

arc type:

negative\_unate  
cell

Cell delay arc

Rise Fall

-----

Input transition time = 0.078463 0.081563 (in library unit)

Effective capacitance = 0.002809 0.003360 (in pF)

Effective capacitance = 0.002809 0.003360 (in library unit)

Drive resistance = 5.662462 10.393124 (in Kohm)

Output transition time = 0.055211 0.099180 (in library unit)

Cell delay = 0.053615 0.081653 (in library unit)

Library units

Rise vs. Fall delay

| Point                                         | Cap | Trans | Incr    | Path     |
|-----------------------------------------------|-----|-------|---------|----------|
| I_ORCA_TOP/I_BLENDER/IU10961/B (GOAI21D1)     |     |       | 0.019 & | 7.714 r  |
| I_ORCA_TOP/I_BLENDER/IU10961/ZN (GOAI21D1) <- |     |       | 0.082 & | 7.796 f. |
| I_ORCA_TOP/I_BLENDER/U80/B (AOI21D1)          |     |       | 0.016 & | 7.812 f  |

3-42

## Net Delay Calculation Reporting with NLDM library

```
report_delay_calculation -from I_ORCA_TOP/I_BLENDER/IU10961/ZN -to \
I_ORCA_TOP/I_BLENDER/U80/B ;# From driver cell output pin to load cell input pin
```

Main Library Units: 1ns 1pF 1kOhm

arc sense: unate

arc type: **net**

**Net delay arc**

RC network on pin 'I\_ORCA\_TOP/I\_BLENDER/IU10961/ZN' :

**RC Network**

Number of elements = 3 Capacitances + 2 Resistances

Total capacitance = 0.004199 pF

Total capacitance = 0.004199 (in library unit)

Total resistance = 4.000010 Kohm

Rise Fall

**Rise vs. Fall delay**

Net delay = 0.018213 0.016164 (in library unit)

Transition time = 0.193072 0.111257 (in library unit)

From\_pin transition time = 0.188472 0.102686 (in library unit)

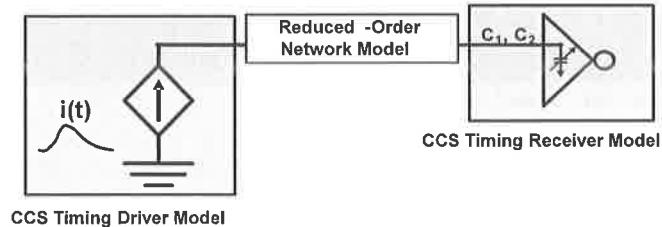
To\_pin transition time = 0.193072 0.111257 (in library unit)

Net slew degradation = 0.004600 0.008571 (in library unit)

| Point                                         | Cap | Trans | Incr    | Path     |
|-----------------------------------------------|-----|-------|---------|----------|
| I_ORCA_TOP/I_BLENDER/IU10961/B (GOAI21D1)     |     |       | 0.019 & | 7.714 r  |
| I_ORCA_TOP/I_BLENDER/IU10961/ZN (GOAI21D1) <- |     |       | 0.082 & | 7.796 f. |
| I_ORCA_TOP/I_BLENDER/U80/B (AOI21D1)          |     |       | 0.016 & | 7.812 f  |

3-43

## CCS Driver and Receiver Models – Most Accurate!



- Time varying nonlinear current source  $i(t)$
- Accurate for any net topology
- Accurate voltage scaling support
- Input capacitance modeled as two values ( $C_1$  and  $C_2$ )
- Models input capacitance variation during transition (Miller effect)
- $C_1$  and  $C_2$  vary with input slew, output load, rise/fall transition, state of the cell, and voltage

3-44

CCS: Composite Current Source (Library Model)

# Cell Delay Calculation Report With CCS Library

Main Library Units: 1ns 1pF 1kOhm

Library: 'tcbn65lpwc\_ccs'

Library Units: 1ns 1pF 1kOhm

Library Cell: 'GOAI21D1' ←

**Library name**

arc sense:

negative\_unate

arc type:

cell ←

**Cell delay arc**

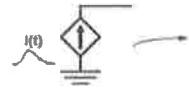
RC network on pin 'I\_ORCA\_TOP/I\_BLENDER/IU10961/ZN' :

Number of elements = 3 Capacitances + 2 Resistances

Total capacitance = 0.004337 pF

Total capacitance = 0.004337 (in library unit)

Total resistance = 4.000010 Kohm



Advanced driver-modeling used for rise and fall.

**Advanced driver modelling = CCS**

Rise Fall

| Point                                         | Cap        | Trans    | Incr              | Path    |
|-----------------------------------------------|------------|----------|-------------------|---------|
| I_ORCA_TOP/I_BLENDER/IU10961/B (GOAI21D1)     |            |          | 0.016 &           | 7.709 r |
| I_ORCA_TOP/I_BLENDER/IU10961/ZN (GOAI21D1) <- |            |          | 0.097 &           | 7.806 f |
| I_ORCA_TOP/I_BLENDER/U80/B (AOI21D1)          |            |          | 0.016 &           | 7.822 f |
| Output transition time                        |            |          |                   |         |
| Cell delay                                    | = 0.079413 | 0.096980 | (in library unit) |         |

**Fall delay**

3-45

CCS: Composite Current Source Library model (More accurate)

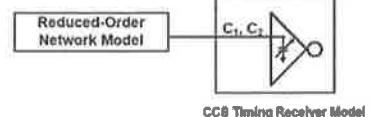
# Net Delay Calculation Report With CCS Library

| Point                                         | Cap | Trans | Incr    | Path    |
|-----------------------------------------------|-----|-------|---------|---------|
| I_ORCA_TOP/I_BLENDER/IU10961/B (GOAI21D1)     |     |       | 0.016 & | 7.709 r |
| I_ORCA_TOP/I_BLENDER/IU10961/ZN (GOAI21D1) <- |     |       | 0.097 & | 7.806 f |
| I_ORCA_TOP/I_BLENDER/U80/B (AOI21D1)          |     |       | 0.016 & | 7.822 f |

RC network on pin 'I\_ORCA\_TOP/I\_BLENDER/IU10961/ZN' :

Number of elements = 3 Capacitances + 2 Resistances  
 Total capacitance = 0.004337 pF  
 Total capacitance = 0.004337 (in library unit)  
 Total resistance = 4.000010 Kohm  
 Advanced receiver-modeling used for rise and fall.  
 Advanced Receiver Model

## Receiver model C1 and C2 = CCS



|                                         |                            |
|-----------------------------------------|----------------------------|
| Rise                                    | Fall                       |
| Receiver model capacitance 1 = 0.001096 | 0.001044 (in library unit) |
| Receiver model capacitance 2 = 0.001154 | 0.001335 (in library unit) |

## C1 and C2 receiver capacitances

Rise Fall  
 Net delay = 0.016839 0.016274 (in library unit)  
 Transition time = 0.176991 0.132863 (in library unit)  
 From\_pin transition time = 0.174309 0.125275 (in library unit)  
 To\_pin transition time = 0.176991 0.132863 (in library unit)  
 Net slew degradation = 0.002682 0.007588 (in library unit)

## Net delay

3-46

CCS: Composite Current Source Library model (More accurate)

## report\_timing != report\_delay\_calculation ?

|                                   |                                                                                                                                                                      |                                                                                                                                                                                                                      |
|-----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Capture path<br>(clock tree cell) | clock clk (rise edge)<br>clock source latency<br>clk (in)<br>U1/I (bufbd1)<br>U1/Z (bufbd1)<br><br>U4/CP (dfnrb1)<br>library setup time<br>data required time<br>... | 0.01<br>0.00<br>0.28<br>0.03<br><br>2.57<br>-112.07<br><br>5.00<br>0.00<br>0.00 &<br>0.14 &<br>0.05 &<br><br>1.28 &<br>-112.07<br><br>5.00<br>5.00<br>5.00 r<br>5.14 r<br>5.19 r<br><br>6.71 r<br>-105.36<br>-105.36 |
|                                   |                                                                                                                                                                      | Delays do not match!                                                                                                                                                                                                 |

```
pt_shell> report_delay_calculation -from U1/I -to U1/Z
 Rise Fall
-----+-----+
Input transition time = 0.301854 0.301854 (in library unit)
Effective capacitance = 0.004258 0.004258 (in pF)
Effective capacitance = 0.004258 0.004258 (in library unit)
Drive resistance = 9.589458 9.005477 (in Kohm)
Output transition time = 63.024223 59.173939 (in library unit)
Cell delay = 69.358299 67.699692 (in library unit) . . .
```



3-47

Some Other reasons :

1. **Set\_timing\_derate:** Report\_timing will reflect the derates applied in delays and arrival times, but report\_delay\_calculation does not
2. **Slew merging :** At each pin, PrimeTime selects the worst slew from all arriving slews to use for forward propagation. The slowest slew is used for max-delay timing, and the fastest slew is used for min-delay timing. In the timing report, worst slew is reported, while in the delay calculation report, we see the actual slew at the pin.

(<http://solvnet.synopsys.com/retrieve/023188.html>)

## report\_delay\_calculation For Min Delay

- By default report\_delay\_calculation considers worst\_slew at cell input to calculate the worst delay.
  - Use –min option to report fast delay through the cell

```
report_delay_calculation -from U1/I -to U1/Z -min
```

```
pt_shell> report_delay_calculation -from U1/I -to U1/Z -min
 Rise Fall

Input transition time = 0.284161 0.284161 (in library unit)
Effective capacitance = 0.004008 0.004008 (in pF)
Effective capacitance = 0.004008 0.004008 (in library unit)
Drive resistance = 0.005532 0.004863 (in Kohm)
Output transition time = 0.034227 0.030032 (in library unit)
Cell delay = 0.052004 0.044064 (in library unit)
```

**023188: When Can report\_timing and  
report\_delay\_calculation Differ?**

<http://solvnet.synopsys.com/retrieve/023188.html>

3-48

## Review of Unit Objectives



**After completing this lecture, you should be able to:**

- Generate summary information for violations sorted by slack, timing check or clock group
- Generate detailed information for cell, net delays and of the paths of interest
- Get explanation for cell and net delays reported

3-49

## Lab 3: Generate Reports - Control Which Paths Are Reported



45 minutes

Generate Summary Reports

Apply report\_timing switches to control which  
and how many paths are reported

Identify and report half cycle paths

3-50

## **Appendix 1**

**Additional Information on report\_delay\_calculation**

3- 51

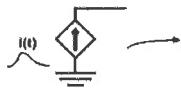
# Enabling CCS delay calculation

## ■ Pre-requisites

- The net has to be completely annotated with detailed parasitics (SPEF)

- The library should have

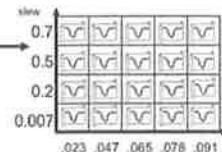
- Current based driver models.



CCS Timing Driver Model

- Advanced receiver capacitance models.

CCS Timing Driver Model



- Set the variables to advanced (Default)

```
set_app_var rc_driver_model_mode advanced
set_app_var rc_receiver_model_mode advanced
```

If CCS data is not present in the lib, NLDM data will be used for delay calculation

3-52

## Timing report with unannotated parasitics

```
Startpoint: a (input port)
Endpoint: z (output port)
Path Group: (none)
Path Type: max
```

| Point                                         | Incr    | Path         |
|-----------------------------------------------|---------|--------------|
| I_ORCA_TOP/I_BLENDER/IU1094/B2 (aoi21d1)      | 0.024 & | 7.465 f      |
| I_ORCA_TOP/I_BLENDER/IU1094/ZN (aoi21d1)      | 0.231 & | 7.695 r      |
| I_ORCA_TOP/I_BLENDER/IU10961/B (GOAI21D1)     | 0.019 & | 7.714 r      |
| I_ORCA_TOP/I_BLENDER/IU10961/ZN (GOAI21D1) <- | 0.051   | 7.765 f      |
| I_ORCA_TOP/I_BLENDER/U80/B (AOI21D1)          | 0.000   | 7.765 f. . . |

Why am I not seeing  
& for this cell arc?  
(WLM based delay  
calculation)

### Hint:

1. Check for any PARA messages in the log
2. Use `report_annotation_parasitics`
3. Use `get_attr [get_nets -of [get_pins \ I_ORCA_TOP/I_BLENDER/IU10961/ZN]] has_valid_parasitics`

# Cell delay calculation using Library Table and Lumped load

```
Main Library Units: ins 1pF 1kOhm
```

```
Library: 'tcbn65lpwc_ccs'
```

```
Library Units: ins 1pF 1kOhm
```

```
Library Cell: 'GOAI2ID1'
```

```
arc sense: negative_unate
```

```
arc type: cell
```

```
Units: ins 1pF 1kOhm
```

```
Rise Delay
```

```
cell delay = 0.0511044
```

```
Table is indexed by
```

```
(X) input_pin_transition = 0.081563
```

```
(Y) output_net_total_cap = 0.00213097
```

```
Relevant portion of lookup table:
```

| (Y) | 0.0018 | (X) | 0.0448 | (X) | 0.0928 | (Z) | 0.0374 | (Z) | 0.0536 | (Z) | 0.0443 | (Z) | 0.0607 |
|-----|--------|-----|--------|-----|--------|-----|--------|-----|--------|-----|--------|-----|--------|
| (Y) | 0.0036 |     |        |     |        |     |        |     |        |     |        |     |        |
| (Y) |        |     |        |     |        |     |        |     |        |     |        |     |        |

Total cap used  
for output load

Look Up Table from library

Since X and Y fall inside the library range,  
interpolation is performed for delay calculation

```
Z = A + B*X + C*Y + D*X*Y
```

```
A = 0.0156
```

```
B = 0.3333
```

```
C = 3.7296
```

```
D = 2.3148
```

```
Z = 0.0511044
```

```
scaling result for operating conditions
```

```
multiplying by 1 gives 0.0511044
```

3-54

what cause a non-zero CRP -- The most common source of CRP is on chip variation results. Less common are reconvergent paths that cause CRP even in bc\_wc mode (which can be caused both by structurally correct paths, as well as by incorrect constraints such as a missing case analysis setting!) The delay to the common point should be the same for the launch and capture path – yes



## Cell Delay Arcs For report\_delay\_calculation

- Delay calculation can be performed on every timing arc of a cell



How do I find timing arcs  
within a cell

Solution

```
foreach_in_collection col [get_timing_arcs -of_objects U3] {
 echo "[get_attribute $col sense] "
 "from pin:" \
 [get_object_name [get_attribute $col from_pin]] \
 "to pin:" \
 [get_object_name [get_attribute $col to_pin]]
}
```

```
setup_clk_rise from pin: U3/CP to pin: U3/D
hold_clk_rise from pin: U3/CP to pin: U3/D
clock_pulse_width_low from pin: U3/CP to pin: U3/CP
clock_pulse_width_high from pin: U3/CP to pin: U3/CP
rising_edge from pin: U3/CP to pin: U3/Q
rising_edge from pin: U3/CP to pin: U3/QN
```

Also recall: `report_lib -timing_arcs <Lib Name> <Cell Name>`

## Conditional Timing Arcs

| Lib Cell Attributes | # | Type/Sense     | From | To | When    |
|---------------------|---|----------------|------|----|---------|
| mx02d0              | 0 | positive_unate | I0   | Z  |         |
|                     | 1 | positive_unate | I1   | Z  |         |
|                     | 2 | positive_unate | S    | Z  | I0'*I1' |
|                     | 3 | negative_unate | S    | Z  | I0*I1'  |
|                     | 4 | negative_unate | S    | Z  |         |
|                     | 5 | positive_unate | S    | Z  |         |

Conditional arcs

- By default, `report_delay_calculation` performs delay calculation for each conditional arc unless a condition is not satisfied.

```
report_delay_calculation -from I_ORCA_TOP/I_BLENDER/U3610/C1 -to
I_ORCA_TOP/I_BLENDER/U3610/S Library: 'cb13fs120_tsmc_max'
arc sense: positive_unate
arc SDF condition: I0==1'b0&&I1==1'b1
arc type: cell
...
arc sense: negative_unate
arc SDF condition: I0==1'b1&&I1==1'b0
arc type: cell
```

Conditional arcs

# Agenda

DAY  
2

4 Constraining Multiple Clocks



5 Additional Checks and Constraints



6 Best Practices Debugging Reports



7 Signoff: Path Based Analysis (PBA)



## Unit Objectives



After completing this unit, you should be able to:

- **On an unfamiliar design, gather basic information about the design clocks:**
  - How many clocks
  - What type and where are the clocks defined
  - Which clocks are interacting

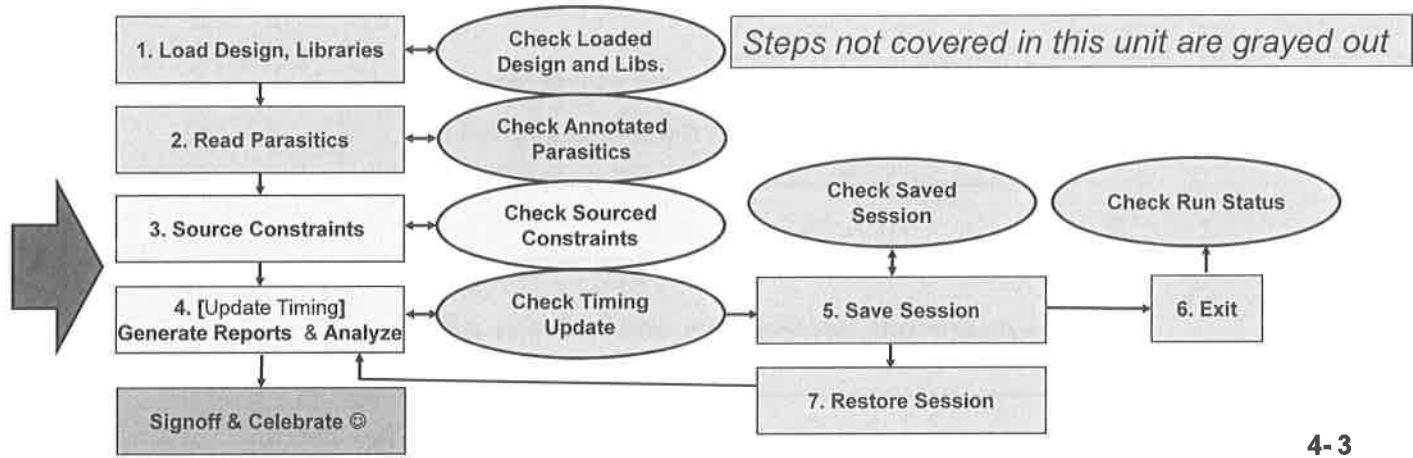
4-2

# Timing Analysis Flow in PrimeTime – Generating Reports

- STA flow is divided into several steps

- Steps 1-3 read data and
- Analysis begins at Step-4

- The STA flow is repeated until signoff is achieved



4-3

STA: Static Timing Analysis

## Clocks and STA : Three Types of Clocks

**STA is dictated largely by the design clocks.**

Faster, easier debugging of timing violations with  
familiarity of the design clocks!



**3 Types of Clocks: Primary, Generated and Virtual**

**Asynchronous, synchronous, and exclusive clocks**

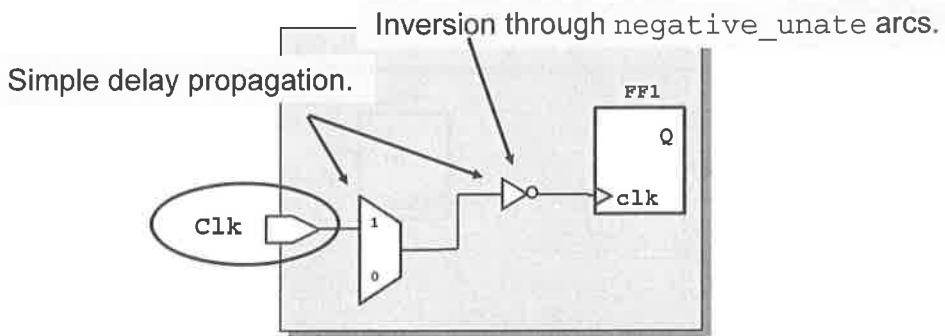
4-4

# What Are Primary Clocks?

- Primary clocks are typically created at input ports

```
create_clock -period 4 [get_ports Clk]
```

- Clocks are propagated through combinational logic



How are network latency/skew/transition determined when Clk is:



Ideal  
Propagated

4-5

```
Primary clock
create_clock -period 4 [get_ports Clk]
Propagate clocks post-CTS
set_propagated_clock [get_clocks Clk]
```

Primary clocks should be created at input ports and output pins of black boxes.

Never create clocks on hierarchy pins.

Creating clocks on hierarchy will cause problems when reading SDF. The net timing arc becomes segmented at the hierarchy and PrimeTime will be unable to annotate that net successfully.

See SolvNet article “**How can I avoid UITE-130 and UITE-136 warnings in my design?**”

Doc ID: 011059 Last Modified: 03/24/2004



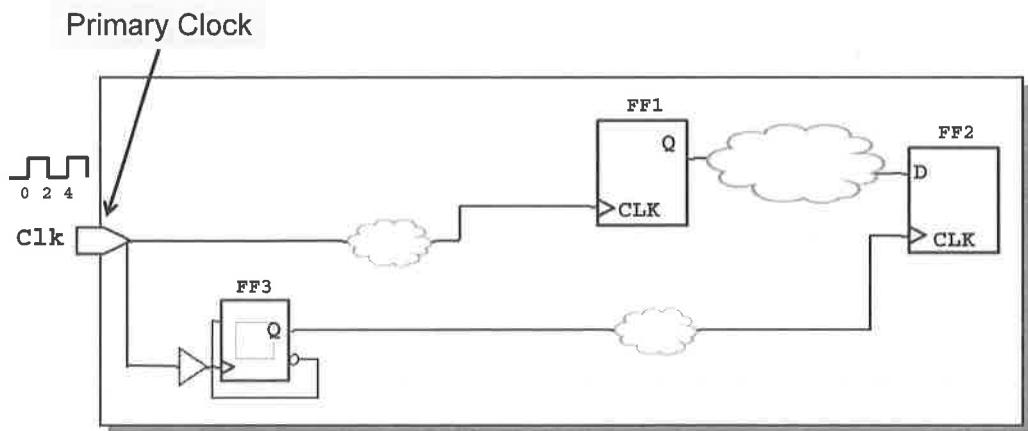
are provided only with set\_clock\_latency – propagated, PT calculates them  
describe what happens if clk is ideal or propagated – ideal delays through mux and inverter

Answer

## Generated Clocks: Internally Derived Clocks



Circle an internal pin where a “generated clock” should be defined.



```
create_generated_clock -divide_by 2 -name div_clk -source [get_ports Clk] FF3/Q
```

4-6

Generated clocks are generally created for waveform modifications of a primary clock (not including simple inversions). PrimeTime does not simulate a design and thus will not derive internally generated clocks automatically – these clocks must be created by the user and applied as a constraint.

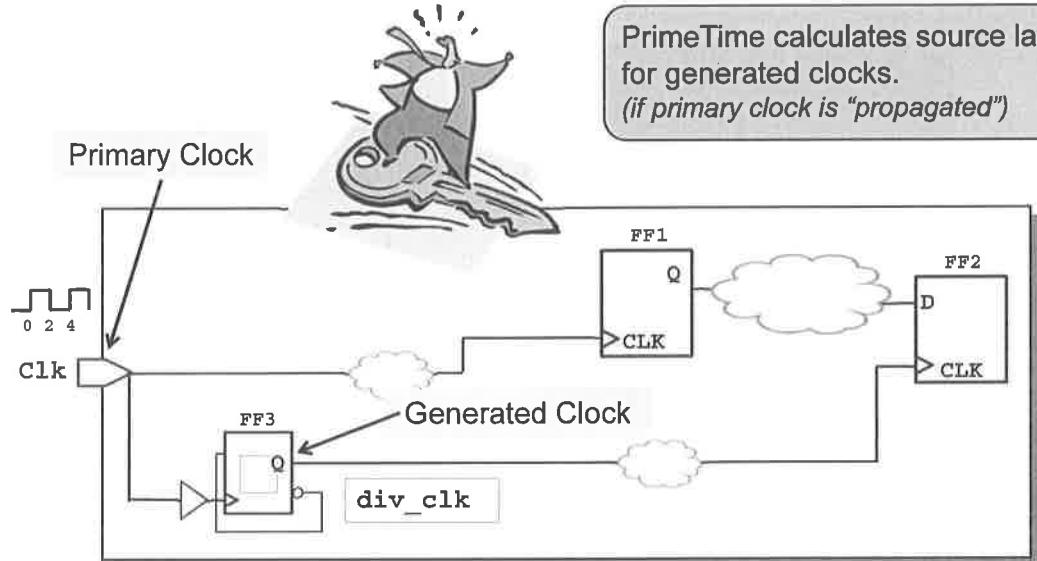
In lab, you will explore a GUI window called the “clock relationship tree” that shows primary clocks and the generated clocks created from them.

```
Primary clock
create_clock -period 4 [get_ports Clk]
Generated clock
create_generated_clock -divide_by 2 -name div_clk -source [get_ports Clk] FF3/Q
```

Circle an internal pin where a generated clock should be defined – ff3/Q

Answer:

## Generated Clocks: Source Latency



Draw the source latency of the generated clock `div_clk`.

4-7

If `div_clk` were created as a primary clock (using the `create_clock` command), the above statement will not be true. PrimeTime will not calculate the source latency of `div_clk` in this case.

The primary clock of the generated clock must be propagated for PrimeTime to calculate the source latency of the generated clock. If this is not true, PrimeTime will not calculate this source latency.

To see the calculated source latency, use

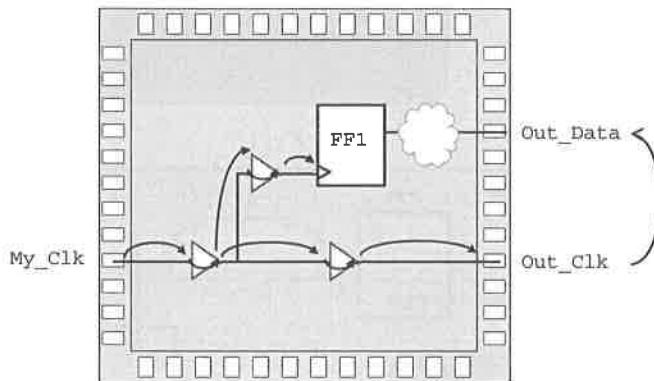
```
pt_shell> report_clock -skew
```

| Object      | Min Condition Source Latency |         |        |        | Max Condition Source Latency |         |        |        |
|-------------|------------------------------|---------|--------|--------|------------------------------|---------|--------|--------|
|             | Early_r                      | Early_f | Late_r | Late_f | Early_r                      | Early_f | Late_r | Late_f |
| SD_DDR_CLK  | 3.089                        | 2.495   | 3.089  | 2.495  | 3.482                        | 3.067   | 3.482  | 3.067  |
| SD_DDR_CLKn | 2.899                        | 3.540   | 2.899  | 3.540  | 3.722                        | 4.269   | 3.722  | 4.269  |
| SYS_2x_CLK  | 1.201                        | 0.603   | 1.201  | 0.603  | 1.228                        | 0.870   | 1.228  | 0.870  |

divide by flop – it would also include source latency constraints applied to CLK  
draw the source latency of the generated clock `div_clk` – from primary clock CLK to the Q pin of the

ANSWER:

## More Clocks - Source Synchronous Interface



The chip specification includes a min/max path requirement with respect to Out\_Clk.

The entire latency from the input port My\_Clk to the output port Out\_Clk is a component of the path requirement.

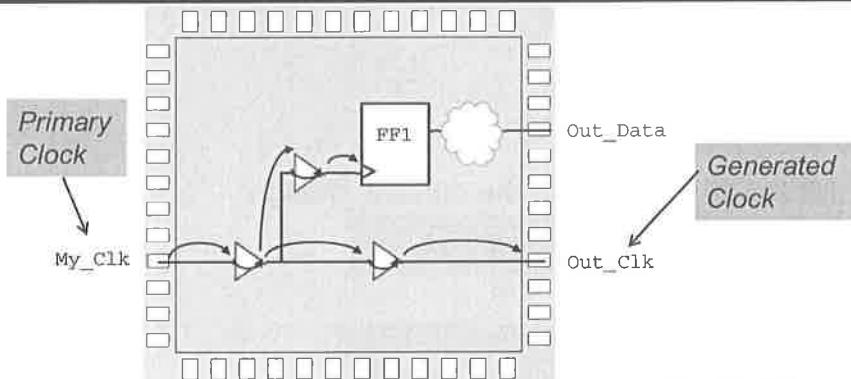
4- 8

**Source synchronous interfaces** refer to designs where the reference clock and data are sent from the transmitting device.

The outgoing clock example is derived from a SNUG paper by Paul Zimmer, San Jose 2001, “**Complex Clocking Situations using PrimeTime**”.



## Generated Clocks: Outgoing Clocks



PrimeTime calculates source latency for generated clocks. (if primary clock is propagated)

```
create_clock -name myclk -period 10 [get_ports My_Clk]
set_clock_latency -source 2 [get_clocks myclk]
set_propagated_clock [get_clocks myclk]
create_generated_clock -name Out_Clk \
 -source [get_ports My_Clk] \
 -divide_by 1 [get_ports Out_Clk]
set_output_delay -max 2 \
 -clock Out_Clk [get_ports Out_Data]
```

Draw the source latency of the generated clock Out\_Clk.

What happens if the clocks are ideal?

4-9

```
Numbers determined by chip specification
create_clock -name myclk -period 10 [get_ports My_Clk]
set_clock_latency -source 2 [get_clocks myclk]
set_propagated_clock [get_clocks myclk]
create_generated_clock -name Out_Clk -source [get_ports My_Clk] \
 -divide_by 1 [get_ports Out_Clk]
set_output_delay -max 2 -clock Out_Clk [get_ports Out_Data]
```

generated clocks. That is, the primary clock is not propagated.

What happens if the clocks are ideal – PT will not calculate source latency for

the output port Out\_Clk  
draw the source latency of the generated clock Out\_Clk – from the primary clock to

Answers:

## Third Kind of Clock - Virtual Clocks

### ■ Virtual clocks:

- Are clock objects without a source
- Do not clock sequential devices within the current\_design
- Serve as references for input or output delays

```
create_clock -period 5 -name vclk
set_input_delay -max 2 -clock vclk [get_ports in1]
set_output_delay -max 1 -clock vclk [get_ports out2]
```

If a virtual clock is propagated:



- PrimeTime calculates network latency.
- There is no network latency to calculate!

4-10

```
Create a virtual clock, vclk, for input and output delay constraints
create_clock -period 5 -name vclk
set_input_delay -max 2 -clock vclk [get_ports in1]
set_output_delay -max 1 -clock vclk [get_ports out2]
```

if a virtual clock is propagated – there is nothing to calculate

Answer:

## Use report\_clock For All Clocks



Circle the source (definition point) for each clock.

Identify the generated clocks.

Identify the virtual clocks (if any).

```
pt_shell> report_clock
Attributes:
 p - Propagated clock
 G - Generated clock
 I - Inactive clock
```



| Clock       | Period | Waveform   | Attrs | Sources   |
|-------------|--------|------------|-------|-----------|
| PCI_CLK     | 15.00  | {0 7.5}    | p     | {pclk}    |
| SDRAM_CLK   | 7.50   | {0 3.75}   | p     | {sdr_clk} |
| SD_DDR_CLK  | 7.50   | {0 3.75}   | p, G  | {sd_CK}   |
| SD_DDR_CLKn | 7.50   | {3.75 7.5} | p, G  | {sd_CKn}  |
| SYS_2x_CLK  | 4.00   | {0 2}      | p, G  |           |
| SYS_CLK     | 8.00   | {0 4}      | p     | {sys_clk} |

4-11

Circle the source for each clock – right column  
Identify the generated clocks – left column  
Identify the virtual clocks (if any) – none

Answers:

## How Many Clocks Are In Your Design?

```
pt_shell> sizeof_collection [all_clocks]
→ 6
```

all\_clocks includes all primary, generated and virtual clocks defined in your design.

```
pt_shell> sizeof_collection [get_generated_clocks *]
→ 3
```



How many reports will report\_timing generate by default?

4-12

Answer to question: One report

```
pt_shell> man all_clocks
```

### NAME

all\_clocks      Creates a collection of all clocks in the current design. You can assign these clocks to a variable or pass them into another command.

### DESCRIPTION

The all\_clocks command creates a collection of all clocks in the current design. If you do not define any clocks, the empty collection (empty string) is returned.

If you want only certain clocks, use get\_clocks to create a collection of clocks matching a specific pattern and optionally pass in filter criteria.

how many reports will report\_timing generate by default – 6, one for each clock

ANSWER:

## Ignorable check\_timing warning

```
pt_shell> check_timing -verbose
Information: Checking 'unconstrained_endpoints'.
Warning: There are 2 endpoints which are not constrained for
maximum delay.

Endpoint

sd_CK
sd_CKn

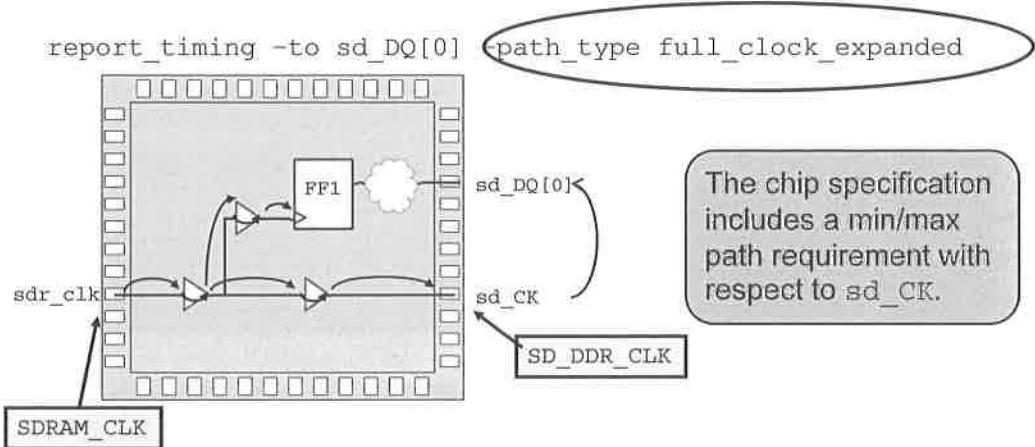
Information: Checking 'unexpandable_clocks'.
Information: Checking 'generic'.
```

**check\_timing** warns of output clock ports unconstrained for output delay – This warning can be ignored

4-13

## Source Latency Calculation Reporting for Generated Clocks

Use the full\_clock\_expanded option to report source latency details for generated clocks.



4-14

## Clocks and STA: Inter Clock Relationships

**STA is dictated largely by the design clocks.**

Faster, easier debugging of timing violations with  
familiarity of the design clocks!

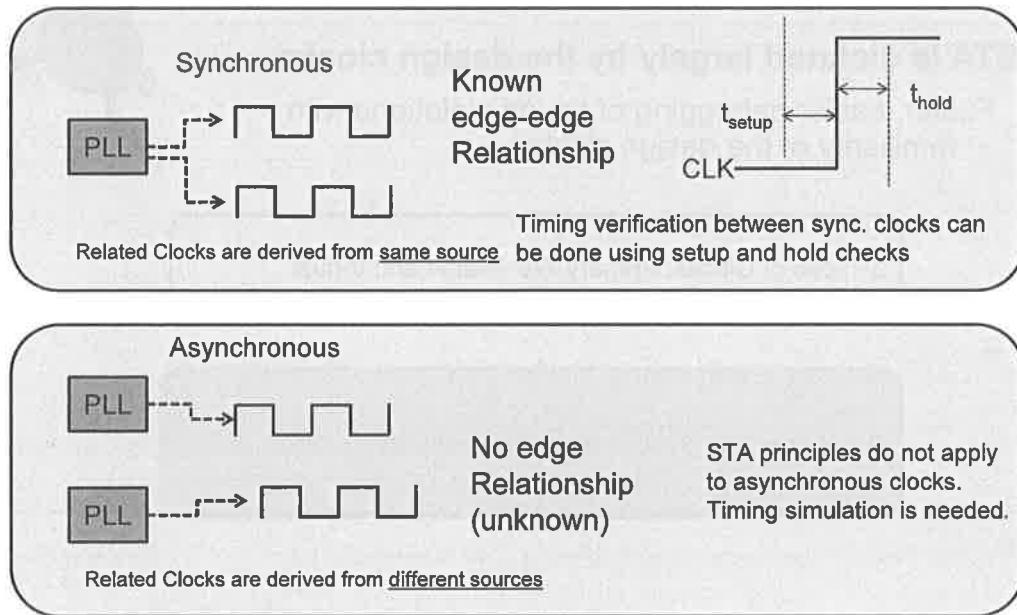


3 Types of Clocks: Primary, Generated and Virtual

Asynchronous, Synchronous, and Exclusive Clocks

4-15

## Timing Between (A)synchronous Clocks



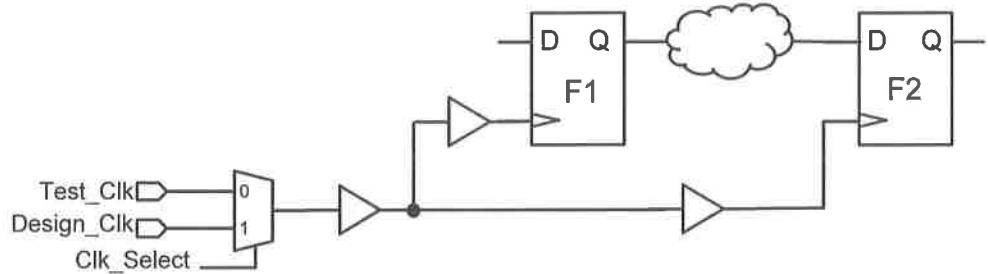
4-16

To specify clocks as asynchronous in PrimeTime, use either of the following choices.

```
Clk1 and Clk2 are asynchronous
set_false_path -from [get_clocks Clk1] -to [get_clocks Clk2]
set_false_path -from [get_clocks Clk2] -to [get_clocks Clk1]

A preferred method, which is also portable to PrimeTime-SI
set_clock_groups -asynchronous -group Clk1 -group Clk2
```

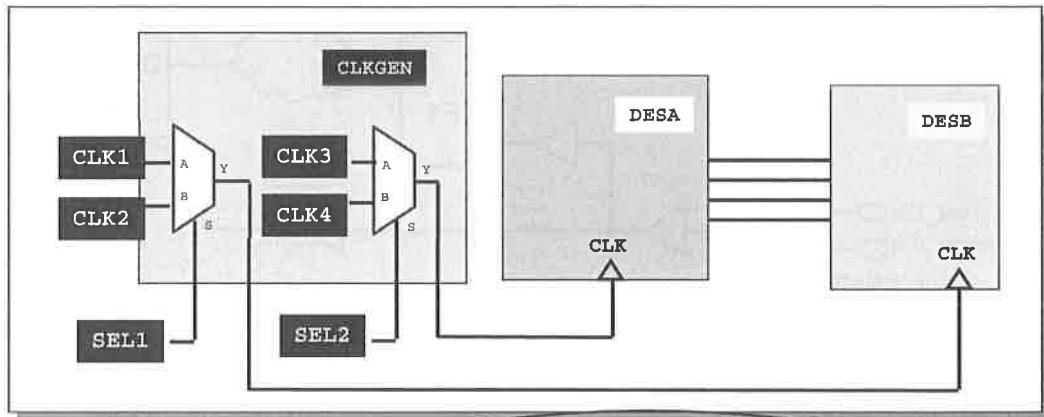
## Exhaustive Analysis != Intended Analysis



- STA is exhaustive [does not use vectors]
  - May occasionally lead to analysis that is unintended
  - For example, in the example above, the path between `Test_clk` and `Design_Clk` could be analyzed
- Clock relationships must, therefore, be defined correctly

4-17

## Interacting Clocks and Multiple STA Runs



One way to ensure timing is verified under all SEL1/SEL2 combinations:

4 STA runs per corner using  
set\_case\_analysis.

Alternatively, this can be done in 1 STA run ...

4-18

4 clock combinations:

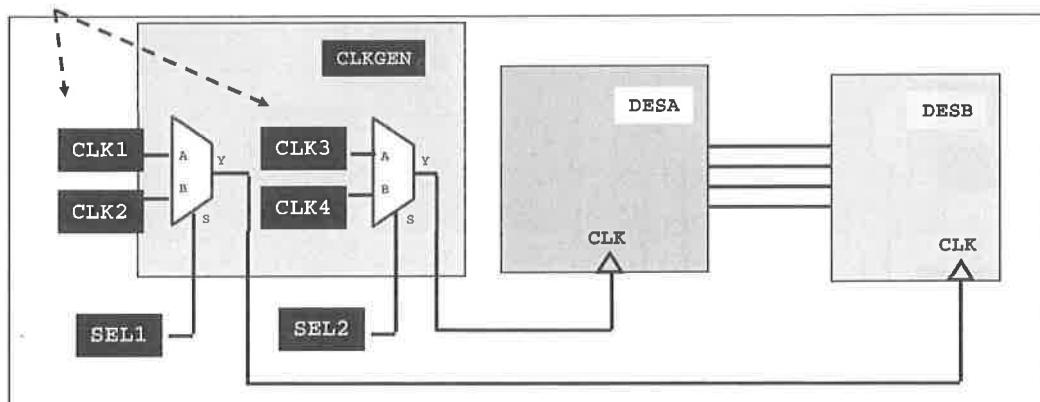
- |                    |                    |
|--------------------|--------------------|
| <b><u>DESA</u></b> | <b><u>DESB</u></b> |
| 1. CLK1            | CLK3               |
| 2. CLK1            | CLK4               |
| 3. CLK2            | CLK3               |
| 4. CLK2            | CLK4               |

Another way to think about this is this design will run with:

(CLK1 or CLK2) and (CLK3 or CLK4)

# Single Analysis with Multiple Clocks!

Step #1: Create all clocks.



Step #2: Specify exclusive clock groups.

```
set_clock_groups -logically_exclusive -group CLK1 -group CLK2
set_clock_groups -logically_exclusive -group CLK3 -group CLK4
Now perform a single analysis run with all clock combinations.
```

4-19

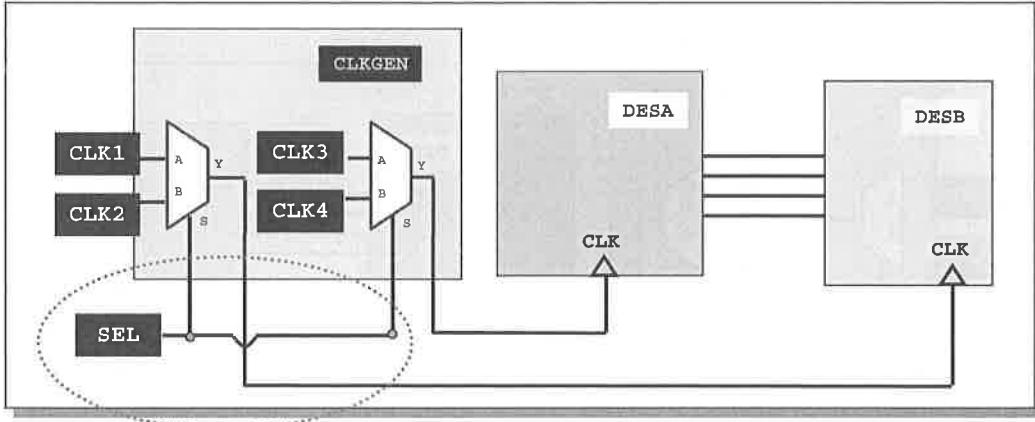
Exclusive clock groups can be logically or physically exclusive. PTSI treats them differently when handling possible cross-coupling. See the man page for `set_clock_groups`.

pt\_shell> man create\_clock

...  
-add

Specifies whether to add this clock to the existing clock or to overwrite. Use this option to capture the case where multiple clocks must be specified on the same source for simultaneous analysis with different clock waveforms. When you specify this option, you must also use the `-name` option.

## Different Example: Multiple Clocks



```
set_clock_groups -name logic_excl_grp -logically_exclusive \
 -group "CLK1 CLK3" \
 -group "CLK2 CLK4"
Now perform a single analysis run with all clock combinations.
```

4-20

### DESA

1. CLK1
2. CLK2

### DESB

1. CLK3
2. CLK4

Another way to think about this is this design will run with:  
(CLK1 and CLK3) or (CLK2 and CLK4)

```
pt_shell> report_clock ~group

Report : clock_groups
Design : MYDES

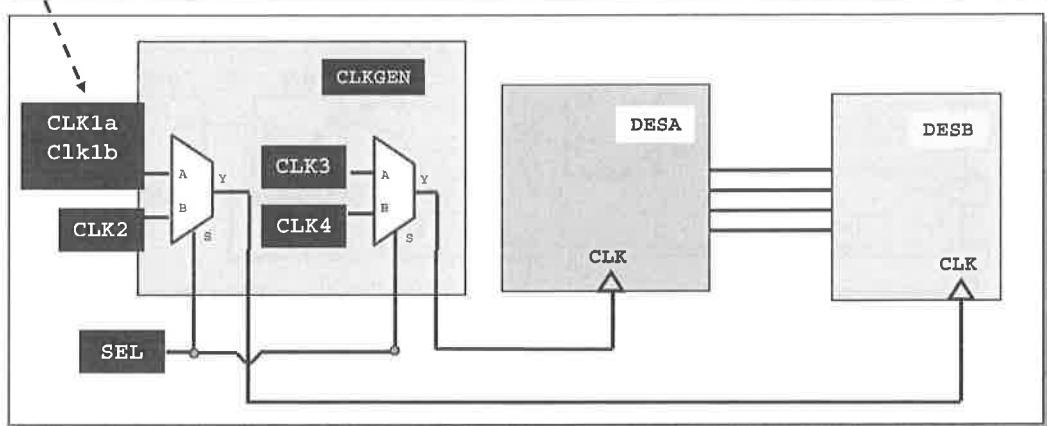
Active clocks:
CLK1 CLK2 CLK3 CLK4
```

```
Total exclusive groups: 1.
NAME : logic_excl_grp
 -group {CLK1 CLK3 }
 -group {CLK2 CLK4 }
```

# Test For Understanding

15 Minutes!

Port CLK1 is driven by a clock with a 10ns period and a range of possible duty cycles:  
→ 40/60 to 60/40, modeled as two clocks: CLK1a and CLK1b.



1. Write the commands to specify each end of the range of duty cycles for CLK1
2. Write the commands to specify the exclusive clock groups.

4-21

For question #2, assume the definition point for CLK1a and CLK1b is the port named **CLK1**; period is 10ns.

```
pt_shell> man create_clock
-add
```

Specifies whether to add this clock to the existing clock or to overwrite. Use this option to capture the case where multiple clocks must be specified on the same source for simultaneous analysis with different clock waveforms. When you specify this option, you must also use the -name option.

```
-waveform edge_list
```

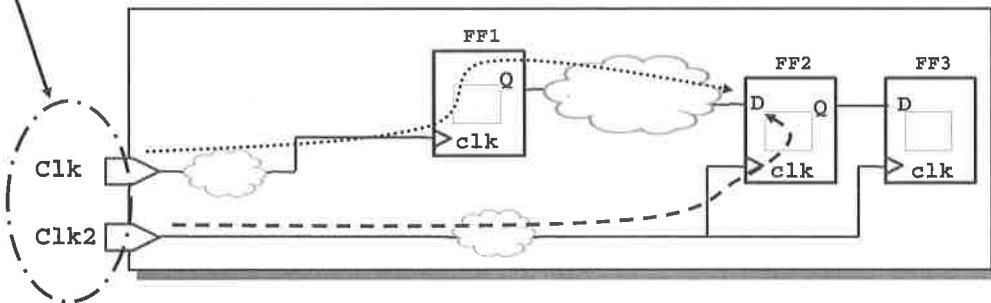
Specifies the rise and fall edge times of the clock waveforms of the clock, in library time units, over an entire clock period. The first time in the edge\_list is a rising transition, typically the first rising transition after time zero. If you do not specify this option, a default waveform is assumed, which has a rise edge of 0.0 and a fall edge of period\_value/2.

```
create_clock -name CLK1a -per 10 -w {0 4} [get_ports CLK1]
create_clock -name CLK1b -per 10 -w {0 6} [get_ports CLK1]
set_clock_groups -physically_exclusive_group "CLK1a CLK1b CLK3" -group "CLK2 CLK4"
set_clock_groups -physically_exclusive_group CLK1a -group CLK1b
```

# Asynchronous Clocks

Asynchronous  
clock groups

FF2 does not need to be checked for reliable data capture.



```
Specify asynchronous clock groups
set_clock_group -name async_group -asynchronous \
 -group Clk -group Clk2
```

4-22

```
pt_shell> report_clock -attributes -groups
```

Attributes:

- p - Propagated clock
- G - Generated clock
- I - Inactive clock

| Clock | Period | Waveform | Attrs | Sources |
|-------|--------|----------|-------|---------|
| Clk   | 4      | {0 2}    | p     | {Clk}   |
| Clk2  | 5      | {0 2.5}  | p     | {Clk2}  |

Total asynchronous groups: 1.  
NAME : async\_group  
-group {Clk}  
-group {Clk2 }

## Identify All Clock Crossings

```
pt_shell> check_timing -verbose -override clock_crossing
Information: There are 4 clocks having domains interacting.

* all paths are false paths
#
part of paths are false paths

From Clock Crossing Clocks

SYS_CLK SDRAM_CLK*, SYS_2x_CLK, PCI_CLK*
SDRAM_CLK SYS_CLK*
PCI_CLK SYS_CLK*
SYS_2x_CLK SDRAM_CLK*, SYS_CLK
```



Circle the one clock pair that has constrained clock crossings.  
Do any timing paths exist between PCI\_CLK and SDRAM\_CLK?

Yes       No

4-23

In lab, you will explore a GUI window called the “clock domain matrix” that shows the same information above, but in a graphical table format.

```
pt_shell> check_timing -help
check_timing # Show possible timing problems for design
[-verbose] (Show detailed information)
[-override_defaults check_list]
 (Check list to replace timing_check_defaults:
 Values: clock_crossing,
 data_check_multiple_clock,
 data_check_no_clock, generated_clocks,
 generic, latch_fanout, latency_override,
 loops, ms_separation, multiple_clock,
 no_clock, no_input_delay,
 partial_input_delay, ideal_clocks,
 no_driving_cell, unexpandable_clocks, retain,
 signal_level, unconstrained_endpoints)

[-include check_list]
 (Check list to add to timing_check_defaults:)

[-exclude check_list]
 (List of checks to subtract from timing_check_defaults:)
```

circle the one clock pair that has constrained clock crossings - SYS\_CLK and SYS\_2x\_CLK  
Do any timing paths exist between PCI\_CLK and SDRAM\_CLK - no

# How Do You Perform These Checks?

Interactively

```
pt_shell> restore_session orca_savesession
pt_shell> check_timing -v -override clock_crossing
```

Or

During initial run

```
lappend timing_check_defaults clock_crossing
Performs default checks + clock_crossing
check_timing -verbose
```

place this variable early  
in your run script

4-24

```
pt_shell> man timing_check_defaults
NAME
 timing_check_defaults
 Defines the default checks for the check_timing command.

TYPE
 list

DEFAULT
 generated_clocks generic latch_fanout loops no_clock
 no_input_delay unconstrained_endpoints

DESCRIPTION
 Defines the default checks to be performed when the check_timing command is
 executed without any options. The same default checks are also performed if
 the check_timing command is used with -include or -exclude options. The
 default check list defined by this variable can be overridden by either
 redefining it before check_timing is executed or using the -override_defaults
 option of the check_timing command.
```

If an undefined check is specified while redefining this variable, a warning  
will be issued by the next execution of the check\_timing command.

where will you place this variable – at top of run file, or in pt-setup.tcl

## Generate Timing Reports Between Clocks



Write the command to generate a timing report for a path launched by `Clk1` and captured by `Clk2`.



\_\_\_\_\_

4- 25

Write the command to generate a timing report from `Clk1` to `Clk2` – `report_timing -from [get_clocks Clk1] -to [get_clocks Clk2]`

ANSWER:

## Interpret Timing Reports Between Clocks



One by one, circle in the report below:

The launch and capture clocks.

The launch and capture clock edges.

| Point                            | Incr    | Path    |
|----------------------------------|---------|---------|
| clock Clk1 (rise edge)           | 8.00    | 8.00    |
| clock network delay (propagated) | 1.10 *  | 9.10    |
| FF1/CLK (fdef1a15)               | 0.00    | 9.10 r  |
| FF1/Q (fdef1a15)                 | 0.50 *  | 9.60 r  |
| U2/Y (buf1a27)                   | 0.11 *  | 9.71 r  |
| U3/Y (buf1a27)                   | 0.11 *  | 9.82 r  |
| FF2/D (fdef1a15)                 | 0.05 *  | 9.87 r  |
| data arrival time                |         | 9.87    |
| clock Clk2 (rise edge)           | 9.00    | 9.00    |
| clock network delay (propagated) | 1.00 *  | 10.00   |
| FF2/CLK (fdef1a15)               |         | 10.00 r |
| library setup time               | -0.21 * | 9.79    |
| data required time               |         | 9.79    |
| -----                            |         |         |
| data required time               |         | 9.79    |
| data arrival time                |         | -9.87   |
| -----                            |         |         |
| slack (VIOLATED)                 | -0.08   |         |

4-26

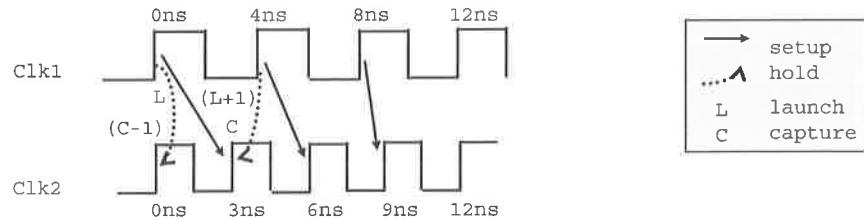
circle launch and capture clocks and clock edges – key – different clocks, edges are 8 and 9

ANSWERS:

## Clock Edges used for Setup and Hold

PT picks the most restrictive pair of edges for setup and for hold. It determines which edges to use as follows:

1. Evaluate waveforms over the smallest common base period
2. For each capture edge, find the closest setup launch edge. Call these the primary pairs
3. Out of the primary pairs, pick the most restrictive setup launch and capture edges
4. For each primary pair, draw two hold relationships: Launch to  $(\text{Capture} - 1)$ ;  $(\text{Launch} + 1)$  to Capture. From all of these hold relationships, pick the most restrictive



4-27

PrimeTime uses the ideal clock waveform (as reported in `report_clock`) to determine the appropriate clock edges for inter-clock analysis.

The most restrictive setup pair is from Clk1 8ns to Clk2 9ns.

The most restrictive hold pair is from Clk1 0ns to Clk2 0ns.

For further reading on single or multiple clock analysis, refer to PrimeTime User Guide: Fundamentals, unit 8 “**Timing Exceptions**” under [Single-Cycle \(Default\) Path Delay Constraints](#).



## Messages During Timing Updates

```
set_app_var timing_update_status_level high
```

```
Information: Related clock set 0 includes clock 'SYS_2x_CLK' with period 4.00. (PTE-064)
Information: Related clock set 0 includes clock 'SYS_CLK' with period 8.00. (PTE-064)
Information: Related clock set 0 has base period 8.00. (PTE-065)
Information: Expanding clock 'SYS_2x_CLK' to base period of 8.00 (old period was 4.00, added 2 edges). (PTE-016)
Information: Expanding clock 'SYS_CLK' to base period of 8.00 (old period was 8.00, added 0 edges). (PTE-016)
```



Circle the base period between the two clocks.

Large base periods can alert you of potential clock definition problems.

4-28

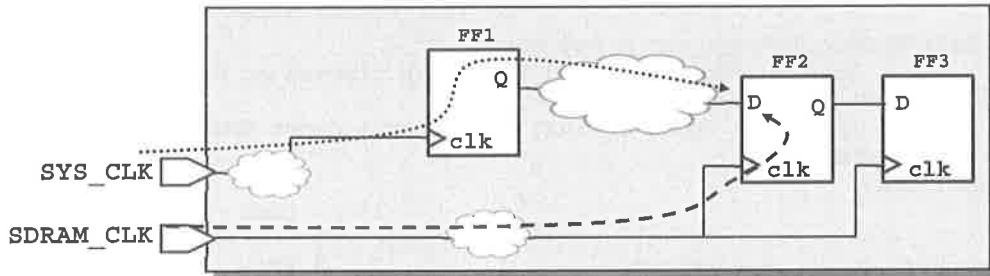
```
pt_shell> man PTE-064
```

### DESCRIPTION

Two clocks are related if there exists a true path between them. They can also be related if they are respectively related to a third clock. For example, if CLK1 is related to CLK2 and CLK2 is related to CLK3, then CLK1 is related to CLK3. That is, CLK1 is related to CLK3 even though there may not be a true path between them. Using the relatedness relation, PrimeTime partitions the clocks in the design into mutually exclusive (related) clock sets. For each clock set, PrimeTime assigns an integer identification number and computes a base period (over all the clocks in the set). This message says that this clock is a member of this clock set.

## Timing Reports for Asynchronous Clocks

FF2 does not need to be checked for reliable data capture.



```
pt_shell> report_timing -from [get_clocks SYS_CLK] \
 -to [get_clocks SDRAM_CLK]
```

No constrained paths.

4- 29

By default, PrimeTime will not generate timing reports for unconstrained timing paths (e.g. asynchronous clock domains).

For further information and examples, refer to SolvNet article “[Why are my unconstrained paths not reported in U2003.03](#)”.

Doc Id: 005347 Last Modified: 03/28/2003



## Reports for Unconstrained Paths



```
pt_shell> set_app_var timing_report_unconstrained_paths true
pt_shell> report_timing -from [get_clocks SYS_CLK] \
 -to [get_clocks SDRAM_CLK]
Startpoint: I_ORCA_TOP/sys_rst_n_buf_reg
 (rising edge-triggered flip-flop clocked by SYS_CLK)
Endpoint: I_ORCA_TOP/sync_reg[4]
 (recovery check against rising-edge clock SDRAM_CLK)
```

Path Group: (none)

Path Type: max

Point

Incr Path

|                                          |        |        |
|------------------------------------------|--------|--------|
| clock network delay (propagated)         | 2.82 * | 2.82   |
| I_ORCA_TOP/sys_rst_n_buf_reg/CP (sdcrq1) | 0.00   | 2.82 r |
| I_ORCA_TOP/sys_rst_n_buf_reg/Q (sdcrq1)  | 0.43 * | 3.25 f |
| I_ORCA_TOP/U93/Z (aor21d1)               | 1.08 * | 4.33 f |
| I_ORCA_TOP/U93ASThfnInst566/Z (bufbda)   | 0.35 * | 4.68 f |
| I_ORCA_TOP/U93ASThfnInst568/Z (bufbd1)   | 0.80 * | 5.48 f |
| I_ORCA_TOP/sync_reg[4]/CDN (sdcrq1)      | 0.01 * | 5.49 f |
| data arrival time                        |        | 5.49   |

(Path is unconstrained)

(Use it for debugging only then turn off – increases PT run times!)

4-30

```
pt_shell> man timing_report_unconstrained_paths
```

### NAME

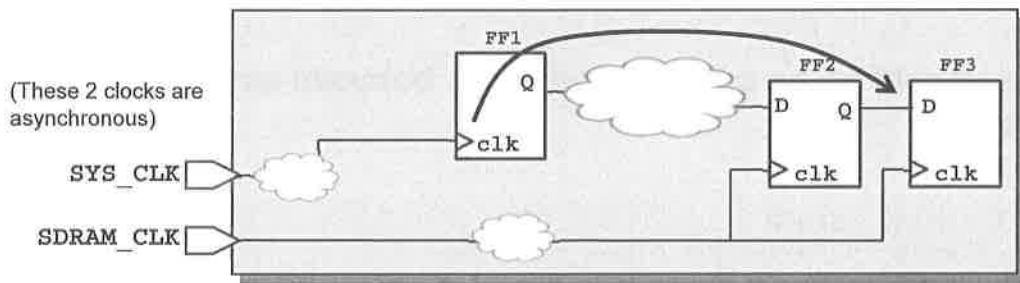
timing\_report\_unconstrained\_paths Specifies if PrimeTime searches for unconstrained paths or not when report\_timing or get\_timing\_paths command is used.

### DESCRIPTION

When this variable sets to FALSE (default behavior), the report\_timing and get\_timing\_paths commands search for constrained paths only. It runs much faster if there are lots of unconstrained paths in the design but you are not interested in these unconstrained paths.

For backward compatibility, you can set this variable to TRUE. The report\_timing and get\_timing\_paths commands will continue searching for unconstrained paths when constrained paths can not be found. Searching unconstrained paths may take longer run-time than expected as warning by UITE-413.

## No Paths versus No Constrained Paths



```
pt_shell> report_timing -from FF1/clk -to FF3/D
No constrained paths.
pt_shell> set_app_var timing_report_unconstrained_paths true
pt_shell> report_timing -from FF1/clk -to FF3/D
No Paths.
```

4- 31

## Two General Guidelines

- There should be no constrained paths between asynchronous clock domains

```
report_timing -group
check_timing -v -override clock_crossing
```

- For the best runtime, turn off reporting of unconstrained paths when generating a large number of reports (warning UITE-413)

```
alias unc_on {set_app_var timing_report_unconstrained_paths true}
alias unc_off {set_app_var timing_report_unconstrained_paths false}
```

4-32

pt\_shell> man UITE-413

### NAME

UIITE-413 (Warning) Searching unconstrained paths will take longer run-time than expected.

### DESCRIPTION

The tool will search for unconstrained paths when constrained paths can not be found, which will involve partial update timing, and takes longer time than searching constrained paths.

### WHAT NEXT

It is suggested that all paths in the design get constrained. The potential unconstrained paths may be caused by black boxes, no launching or capturing clocks, empty hierarchy, cells from IP instead of from library, etc. To check constraints in the design, use the check\_timing command. Specifying the to\_pin\_list can also reduce the number of endpoints to search and speed up the report timing process.

ANSWER: how will you validate this - check\_timing -v -override clock\_crossing

## Lab 4: Getting to Know Your Clocks



45 minutes

Apply the commands from lecture to gain familiarity with the design clocks

Use the GUI for another view of the clock relationships in a design

4- 33

This page was intentionally left blank

# Agenda

DAY  
2

4 Constraining Multiple Clocks



5 Additional Checks and Constraints



6 Best Practices Debugging Reports

7 Signoff: Path Based Analysis (PBA)



5- 1

# Unit Objectives



After completing this unit, you should be able to:

■ **Exercise the following timing checks during STA**

- I. Recovery, Removal
- II. Clock Gating Checks
- III. Data to Data Checks
- IV. Minimum Pulse Width Checks

■ **Analyze timing that involves**

1. Latches with Time borrowing
2. Multi cycle paths
3. Combinational feedback loops
4. Non unate cells along clock paths

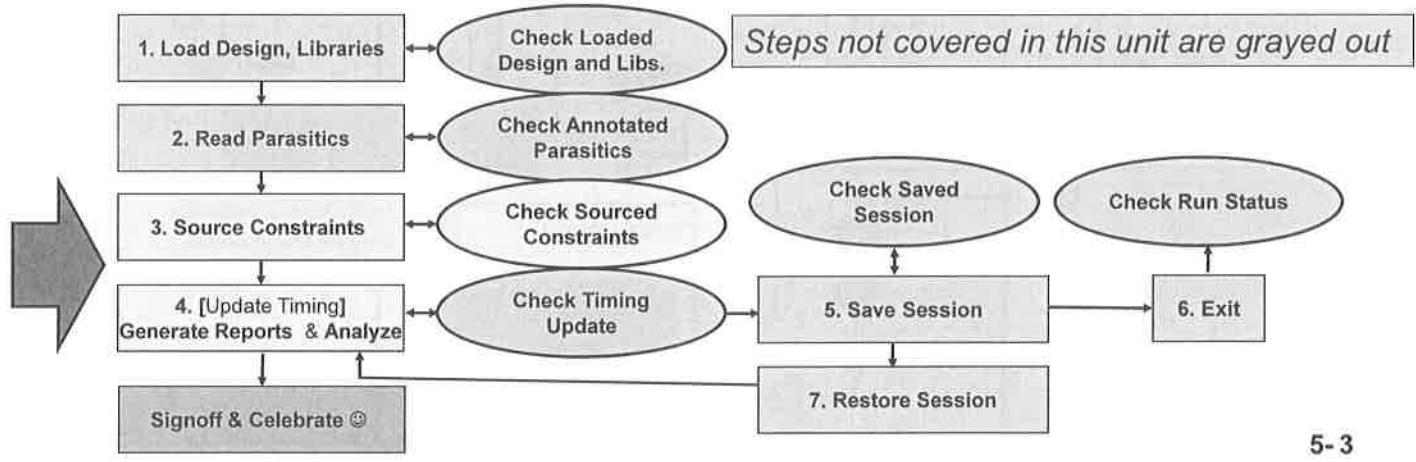
5- 2

# Timing Analysis Flow in PrimeTime – Generating Reports

- STA flow is divided into several steps

- Steps 1-3 read data and
- Analysis begins at Step-4

- The STA flow is repeated until signoff is achieved



5- 3

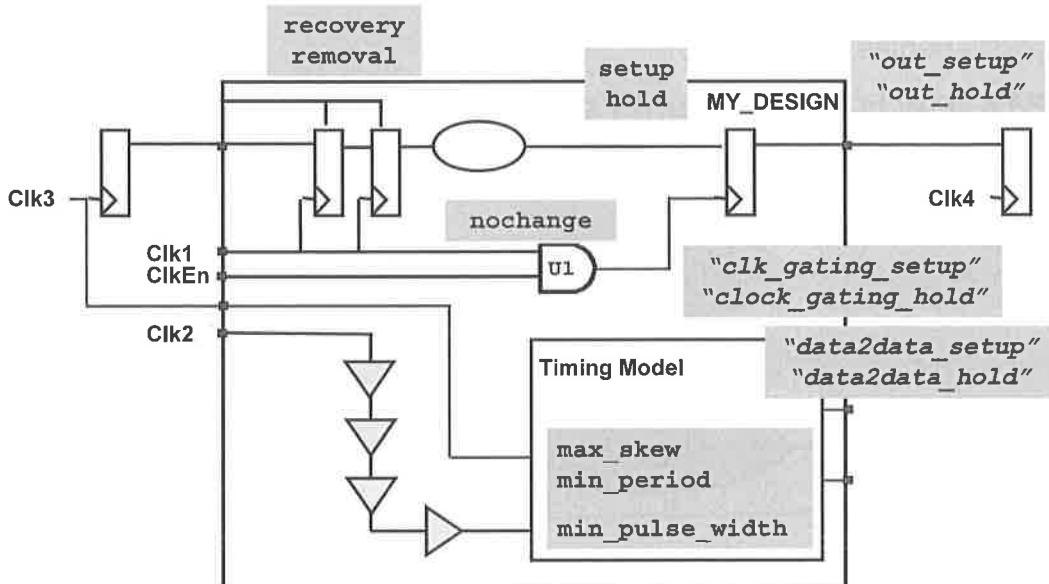
STA: Static Timing Analysis

## Timing Checks Verified by STA



Circle the timing checks already covered.

"Timing checks": specified by the user  
 Timing checks: specified by the vendor



5-4

There are three additional checks, which can be specified in the library:

**min period:** This is a design rule, which can specify a minimum clock period requirement.

**clock separation:** This is a constraint for master/slave latches for a minimum required clock separation between two clocks to avoid the latch from becoming transparent.

**nonsequential:** This is similar to a data to data setup and hold check between two data pins.

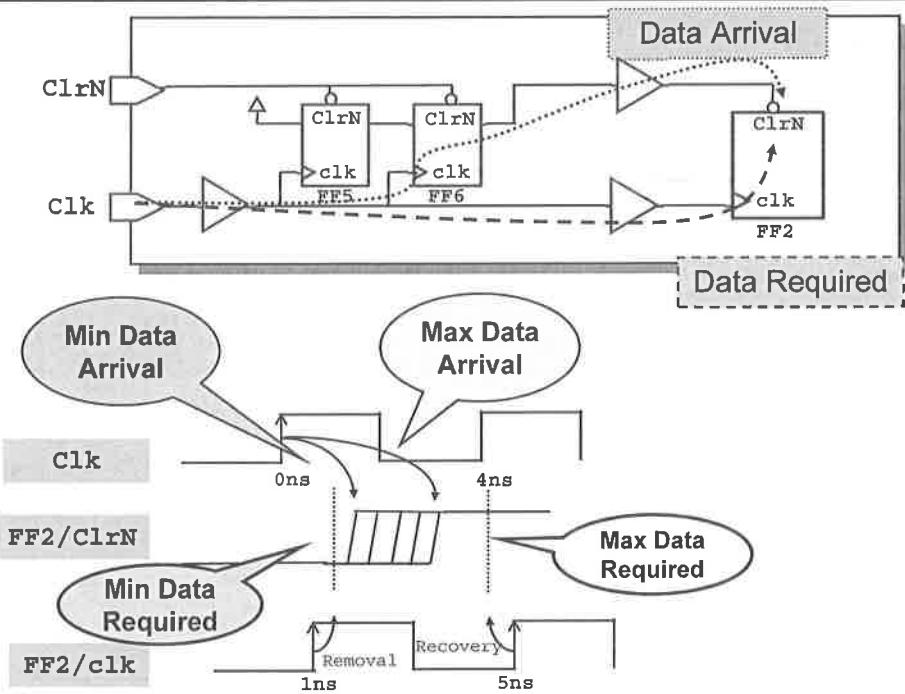
**max skew:** The maximum time allowed between two clock signals for the rising or falling edges. Clock uncertainty is not taken into account for this check, only the calculated actual skew.

A **constraint check** is one specified by the user as a constraint (e.g. set\_output\_delay).

A **library check** is one specified by the vendor in the library for that specific leaf cell (e.g. setup/hold).

**Specified when writing out an extracted model:** nochange is clock gating setup and hold checks modeled as nonchange arcs on the extracted model.

## I. Asynchronous Clear/Reset Pins



5-5

The assertion of asynchronous reset/clear pins on flip-flops is considered asynchronous and is not validated for timing by STA tools.

### Recovery:

The minimum amount of time required for an asynchronous control signal to become inactive and a subsequent active edge of the clock, (this is similar to a setup check).

### Removal:

The minimum amount of time required between a clock edge that occurs while an asynchronous control signal is active and the subsequent removal of the asynchronous control signal, (this is similar to a hold check).

The reset synchronizing circuitry was outlined at SNUG San Jose 2002, Clifford E. Cummings and Don Mills “Synchronous Resets? Asynchronous Resets? I am so confused! How will I ever know which to use?”



# Timing Report Recovery



One by one, circle the:

Path group.

Timing path endpoint.

Data transition at the endpoint.

Library recovery time.

Startpoint: I\_ORCA\_TOP/I\_RESET\_BLOCK/sys\_2x\_rst\_n\_buf\_reg  
(rising edge-triggered flip-flop clocked by SYS\_2x\_CLK)

Endpoint: I\_ORCA\_TOP/I\_RISC\_CORE/I\_ALU/Neg\_Flag\_reg  
(recovery check against rising-edge clock SYS\_2x\_CLK)

Path Group: \*\*async\_default\*\*

Path Type: max

| Point                                                     | Incr    | Path             |
|-----------------------------------------------------------|---------|------------------|
| clock SYS_2x_CLK (rise edge)                              | 0.000   | 0.000            |
| clock network delay (propagated)                          | 2.846 & | 2.846            |
| I_ORCA_TOP/I_RESET_BLOCK/sys_2x_rst_n_buf_reg/CP (sdcrq1) | 0.000   | 2.846 r          |
| I_ORCA_TOP/I_RISC_CORE/I_ALU/Neg_Flag_reg/CDN (sdcrb1)    | 0.073 & | 3.974 r<br>3.974 |
| data arrival time                                         |         |                  |
| clock SYS_2x_CLK (rise edge)                              | 4.000   | 4.000            |
| clock network delay (propagated)                          | 2.833 & | 6.833            |
| I_ORCA_TOP/I_RISC_CORE/I_ALU/Neg_Flag_reg/CP (sdcrb1)     | 0.128   | 6.833 r<br>6.962 |
| library recovery time                                     |         |                  |
| data required time                                        |         |                  |
| data required time                                        |         | 6.962            |
| data arrival time                                         |         | -3.974           |
| slack (MET)                                               |         | 2.988            |

Adding recovery time???

5-6

- Path group (\*\*async\_default\*\*), timing path endpoint (async data pin), data transition at the endpoint (r on CDN), library recovery time
- We're actually subtracting a negative recovery time!

ANSWERS:

## Test For Understanding



```
report_timing -delay max_fall -to FF2/ClrN
```

What is the result?



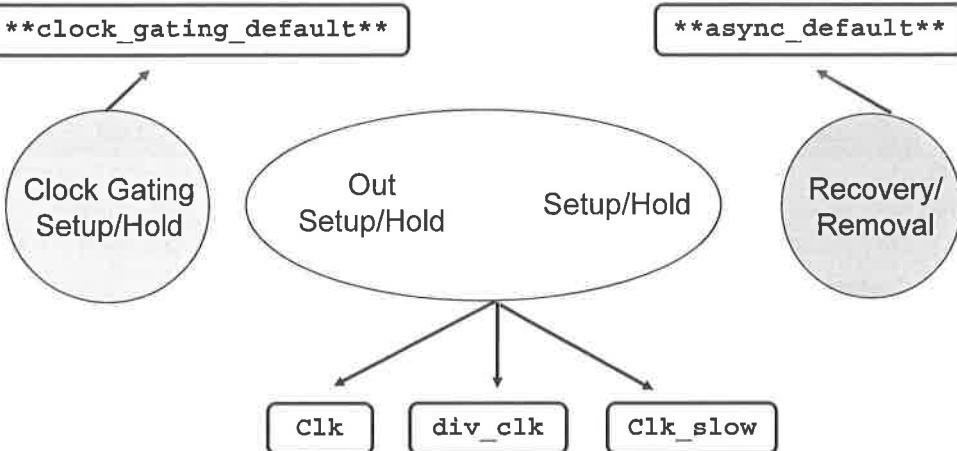
- A recovery report
- A removal report
- No constrained paths

5-7

what is the result – no constrained paths

ANSWERS:

## Path Groups – Full Picture



*Timing paths grouped by the capture clock.*



Write the command to generate  
a single timing report for removal.

5-8

Two additional path groups:

**\*\*default\*\***: This path group contains timing paths not constrained by a clock but constrained with `set_max_delay`/`set_min_delay`.

Example:

An output timing path constrained with `set_max_delay` instead of `set_output_delay`.

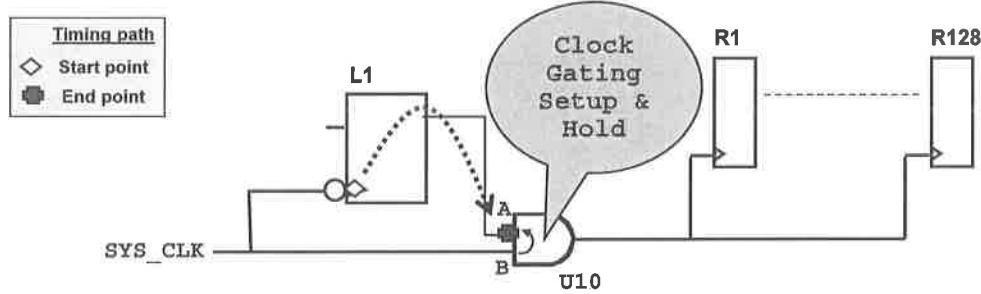
**<none>**: This path group contains timing paths which are unconstrained.

| pt_shell> report_path_group |        |       |      |         |          |
|-----------------------------|--------|-------|------|---------|----------|
| Critical                    |        |       |      |         |          |
| Path_Group                  | Weight | Range | From | Through | To       |
| <hr/>                       |        |       |      |         |          |
| **clock_gating_default**    | 1.00   | 0.00  | -    | -       | -        |
| **default**                 | 1.00   | 0.00  | -    | -       | -        |
| Clk                         | 1.00   | 0.00  | *    | *       | Clk      |
| Clk_slow                    | 1.00   | 0.00  | *    | *       | Clk_slow |
| div_clk                     | 1.00   | 0.00  | *    | *       | div_clk  |

Write the command to generate a single timing report for removal – `report_timing_group`  
`*async_default* -delay min`

Answer:

## II. Clock Gating Checks



The clock enable must arrive at U10/A when:



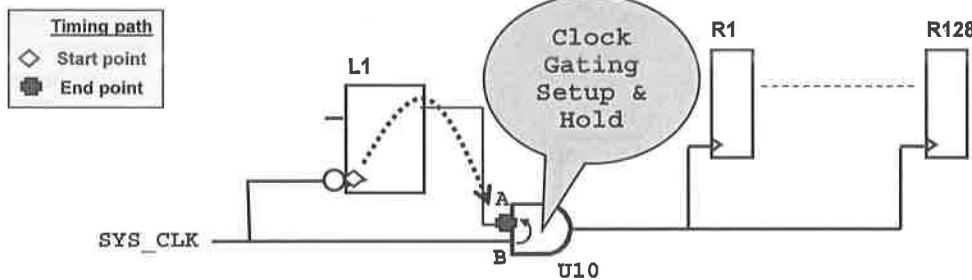
- The clock is low.
- The clock is high.

5-9

the clock enable must arrive at U10/A when the clock is low

ANSWER:

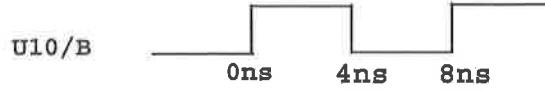
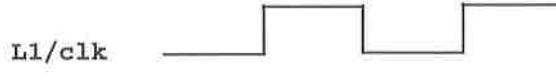
## Which Clock Edges Are Used?



Circle the clock edges for:



Setup.  
Hold.



0ns      4ns      8ns

5-10

circle the clock edges for setup and hold - 4 (A is active low to 8 (B is active high ; 4 to 4

Answer:

# Timing Report for Clock Gating

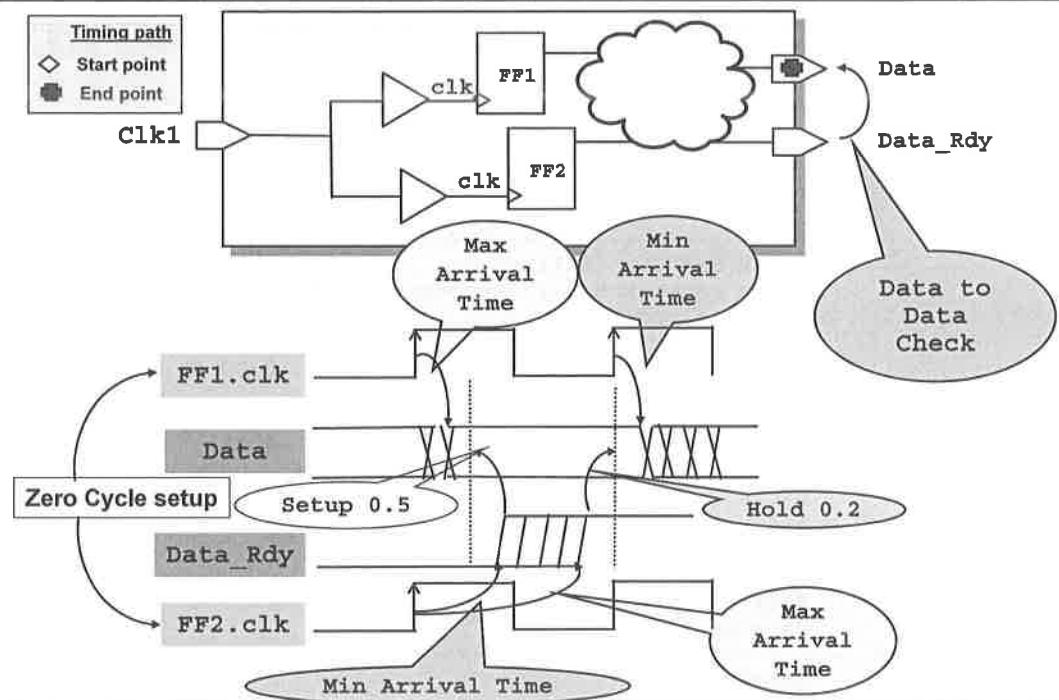
```
set_clock_gating_check -setup 0.20 [get_clocks SYS_CLK]
report_timing -to [get_pins U10/A]
```

Startpoint: L1 (negative level-sensitive latch clocked by SYS\_CLK)  
 Endpoint: U10 (rising clock gating-check end-point clocked by SYS\_CLK)  
 Path Group: \*\*clock\_gating\_default\*\*  
 Path Type: max

|                                  | Timing path | Path group | Clock edges |
|----------------------------------|-------------|------------|-------------|
| clock SYS_CLK (fall edge)        | 4.00        | 4.00       |             |
| clock network delay (propagated) | 2.37 &      | 6.37       |             |
| L1/EN (slnlq1)                   | 0.00        | 6.37 f     |             |
| L1/Q (slnlq1)                    | 0.62 &      | 6.99 f     |             |
| U10/A (ora21d4)                  | 0.00 &      | 6.99 f     |             |
| data arrival time                |             | 6.99       |             |
| clock SYS_CLK (rise edge)        | 8.00        | 8.00       |             |
| clock network delay (propagated) | 2.09 &      | 10.09      |             |
| U10/B (ora21d4)                  |             | 10.09 r    |             |
| clock gating setup time          | -0.20       |            |             |
| data required time               |             | 9.89       |             |
| Setup time constraint            |             | 9.89       |             |
| data required time               |             | -6.99      |             |
| data arrival time                |             |            |             |
| slack (MET)                      |             | 2.89       |             |

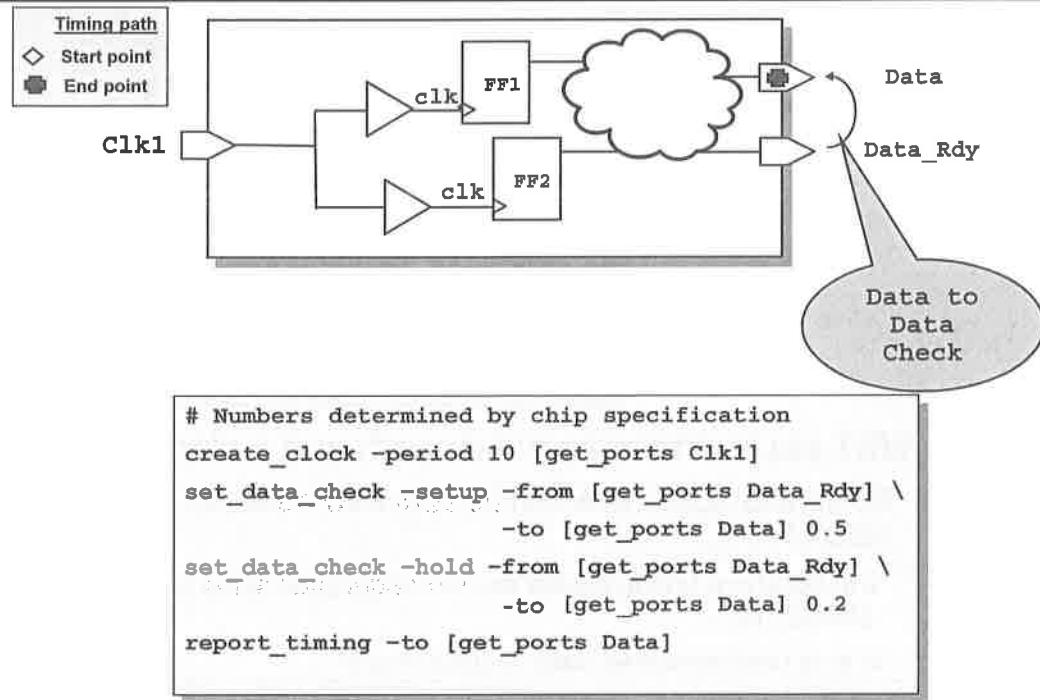
5-11

### III. Data to Data Check



5-12

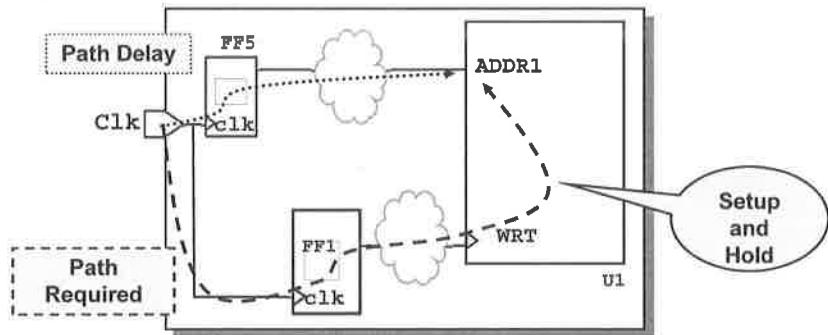
## Specify Data to Data Checks



5-13

## Another Application for a Data-to-Data Check

### Synchronous Memory Used Asynchronously

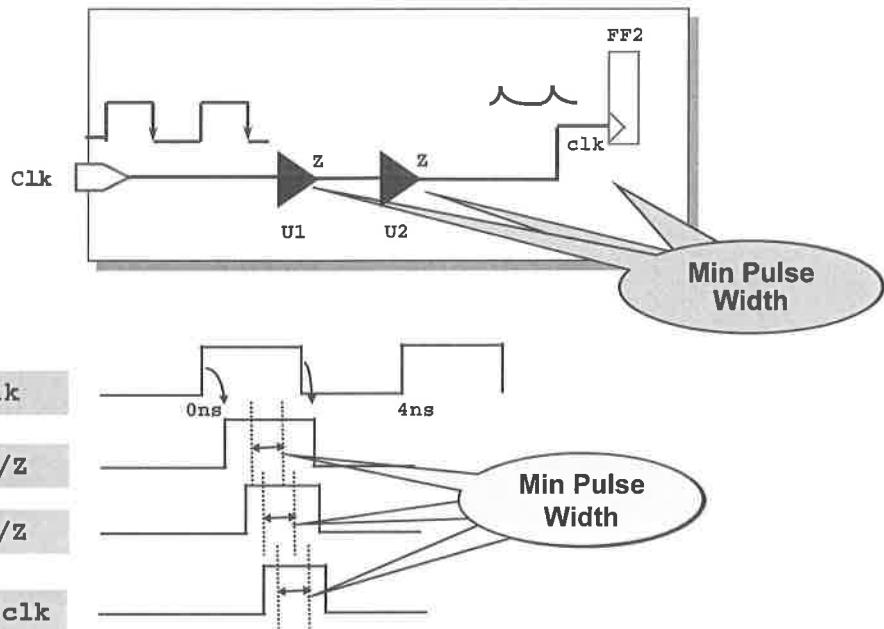


- **WRT pin is driven by a data path, not a clock signal.**

- Setup and hold check can be specified between WRT and ADDR1
  - ◆ in library or timing model as *non-sequential setup and hold checks* (or)
  - ◆ as a user-specified data-to-data check

5-14

## IV. Clock Min Pulse Width



5-15

This check can either be specified in the technology library for each appropriate cell, or you can specify a check with `set_min_pulse_width`.

```
pt_shell> set_min_pulse_width -help
set_min_pulse_width # Specify min pulse width checks
 [-low] (Set only min pulse width low)
 [-high] (Set only min pulse width high)
 value (Minimum pulse width: Value >= 0)
 [object_list] (List of clocks, pins or cells in the design)
```

### Minimum Clock Pulse Width:

The minimum allowable time for the high or low phase of the clock.

## Summary Min Pulse Width Reports

```
pt_shell> report_min_pulse_width

Report : min pulse width
-path_type summary ← Default path_type

sequential_clock_pulse_width

 Required Actual
Pin pulse width pulse width Slack

I_ORCA_TOP/I_RISC_CORE/I_REG_FILE/REG_FILE_A_RAM/CE1 (low)
 0.377 0.337 -0.040 (VIOLATED)
I_ORCA_TOP/I_RISC_CORE/I_REG_FILE/REG_FILE_A_RAM/CE2 (low)
 0.377 0.337 -0.040 (VIOLATED)
I_ORCA_TOP/I_RISC_CORE/I_REG_FILE/REG_FILE_B_RAM/CE1 (low)
 0.377 0.337 -0.040 (VIOLATED)
```

```
pt_shell> report_min_pulse_width -path_type short →
```

5-16

# Test For Understanding



## One by one, circle:

The constrained clock pin. Required pulse width.



This report is for a pulse width:  high  low

What is the largest contributor to the failing pulse width?

network delay  uncertainty

| Point                                                | Incr   | Path    |
|------------------------------------------------------|--------|---------|
| clock SYS_2x_CLK (fall edge)                         | 2.000  | 2.000   |
| clock network delay (ideal)                          | 1.014  | 3.014 f |
| I_ORCA_TOP/I_RISC_CORE/I_REG_FILE/REG_FILE_A_RAM/CE1 | 0.000  | 3.014 f |
| open edge clock latency                              |        | 3.014   |
| clock SYS_2x_CLK (rise edge)                         | 4.000  | 4.000   |
| clock network delay (ideal)                          | 0.851  | 4.851 r |
| I_ORCA_TOP/I_RISC_CORE/I_REG_FILE/REG_FILE_A_RAM/CE1 | 0.000  | 4.851 r |
| clock uncertainty                                    | -1.500 | 3.351   |
| close edge clock latency                             |        | 3.351   |
| required pulse width (low)                           | 0.377  |         |
| actual pulse width                                   | 0.337  |         |
| slack (VIOLATED)                                     | -0.040 | 5-17    |

pt\_shell> report\_min\_pulse\_width -help

Usage:

```
report_min_pulse_width # Report min pulse width check
[-all_violators] (Show only min pulse width violations)
[-significant_digits digits]
 (Number of digits to display:
 Range: 0 to 13)
[-nosplit] (Do not split lines when columns overflow)
[-path_type format] (Format for path report:
 Values: full_clock, full_clock_expanded,
 short, summary)
[-input_pins] (Show input pins in path)
[-derate] (Show derate factors)
[port_pin_list]
 (List of objects to check min pulse width
with.)
```

circle the constrained clock pin - CE1 - low - that is, goes low at 3.014, and stays low until 3.351, for a pulse width of 0.337 – uncertainty of 1.5 is the largest contributor

ANSWER:

# Summary Report for All Timing Checks



Circle the new checks just learned.



Which command  
generates this report?

| Type of Check      | Total | Met          | Violated | Untested     |
|--------------------|-------|--------------|----------|--------------|
| setup              | 9629  | 3500 ( 36%)  | 88 ( 1%) | 6041 ( 63%)  |
| hold               | 9629  | 3588 ( 37%)  | 0 ( 0%)  | 6041 ( 63%)  |
| recovery           | 1316  | 1210 ( 92%)  | 0 ( 0%)  | 106 ( 8%)    |
| removal            | 1316  | 1210 ( 92%)  | 0 ( 0%)  | 106 ( 8%)    |
| min_period         | 20    | 20 (100%)    | 0 ( 0%)  | 0 ( 0%)      |
| min_pulse_width    | 7273  | 5957 ( 82%)  | 0 ( 0%)  | 1316 ( 18%)  |
| clock_gating_setup | 33    | 33 (100%)    | 0 ( 0%)  | 0 ( 0%)      |
| clock_gating_hold  | 33    | 33 (100%)    | 0 ( 0%)  | 0 ( 0%)      |
| out_setup          | 75    | 75 (100%)    | 0 ( 0%)  | 0 ( 0%)      |
| out_hold           | 75    | 75 (100%)    | 0 ( 0%)  | 0 ( 0%)      |
| All Checks         | 29333 | 15635 ( 53%) | 88 ( 0%) | 13610 ( 46%) |

5-18

The complete list of checks that are reported by `report_analysis_coverage`:

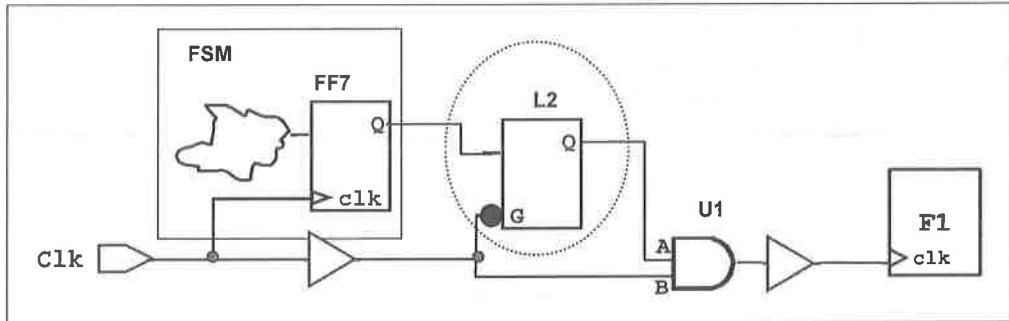
- setup/hold
- recovery/removal
- nochange
- min\_period
- min\_pulse\_width
- clock\_separation
- clock\_gating\_setup/clock\_gating\_hold
- max\_skew
- out\_setup/out\_hold

There is also a `report_min_period` command.

What command generates this report – `report_analysis_coverage`  
circle checks just learned – recovery, removal, min\_pulse\_width, out\_setup and out\_hold

ANSWER:

## 1. Latch based analysis [Default = Old]



- 1. Latches are level sensitive sequential devices
- 2. A Latch has a  $D \rightarrow Q$  timing arc, in addition to  $G \rightarrow Q$ 
  - a. The  $D \rightarrow Q$  arc is used if time borrowing is involved
  - b. The  $G \rightarrow Q$  arc is used with zero time borrowing [Flop like timing]

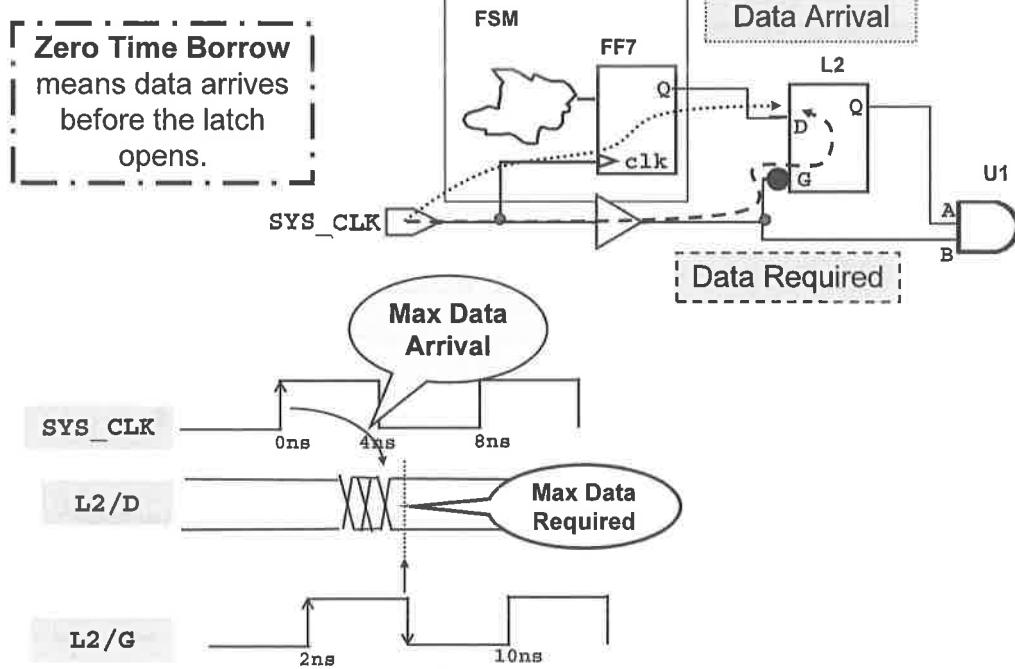
5-19

This is a common clock gating implementation by Power Compiler.

For further reading, refer to PrimeTime User Guide: Advanced Timing Analysis, in unit 5 “Advanced Analysis Techniques” under [Time Borrowing in Latch-Based Designs](#).



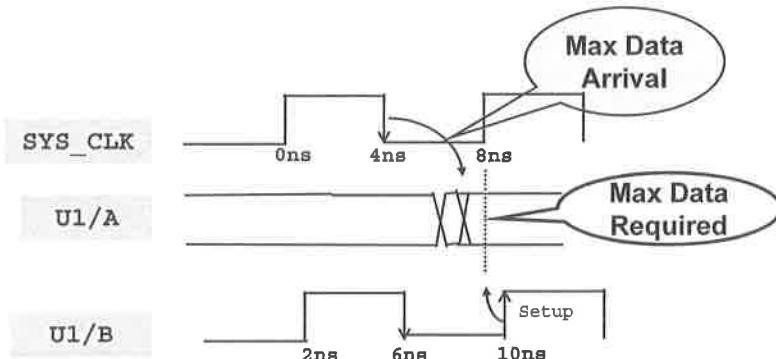
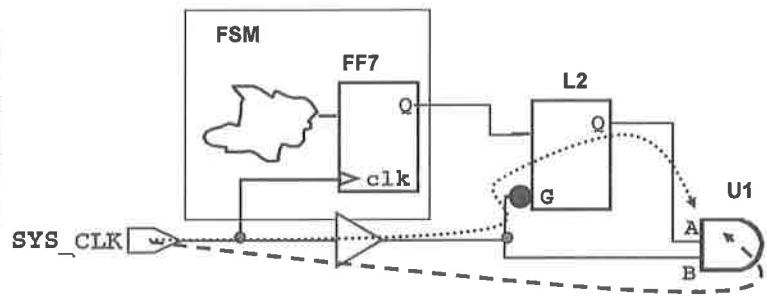
## Latch with Zero Time Borrow : Path to L2/D



5-20

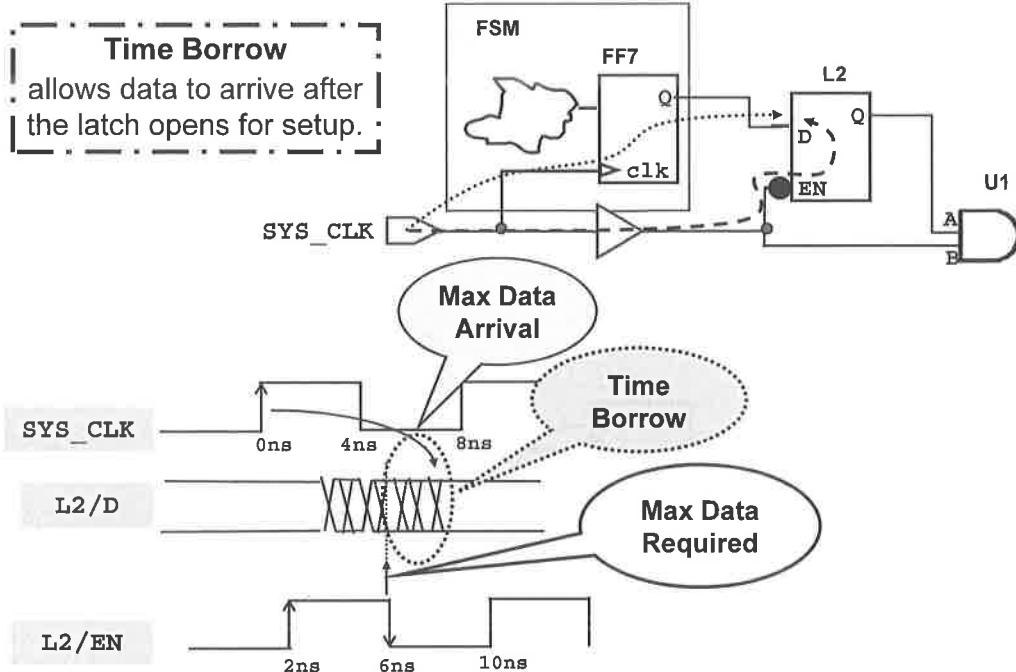
## Latch w/ Zero Time Borrow : Path from L2/G

**Zero Time Borrow**  
also means that the paths to L2/D and from L2/G are independent.



5-21

## Latches with Time Borrow: Path to L2/D



Commands and variables associated with latches and time borrowing.

```
pt_shell> aa borrow

Commands *****
remove_max_time_borrow # Remove time borrow limit for latches
set_max_time_borrow # Limit time borrowing for latches

Variables *****
timing_allow_short_path_borrowing = "false"
timing_early_launch_at_borrowing_latches = "true"
timing_include_available_borrow_in_slack = "false"

pt_shell> aa uncertainty

Commands *****

Variables *****
timing_propagate_interclock_uncertainty = "false"
```

## Time Borrowing Information in Report [Default = Old]

The time borrowed

Slack

Setup time of latch

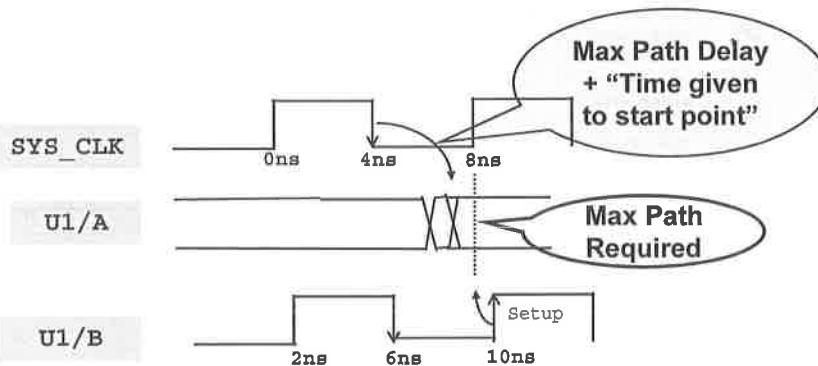
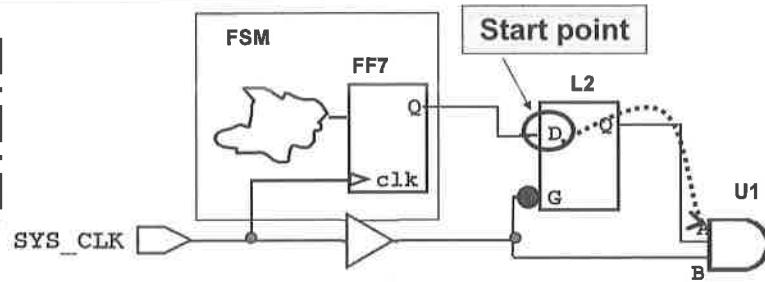
Max allowable time borrow

| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock SYS_CLK (rise edge)        | 0.00   | 0.00   |
| clock network delay (propagated) | 2.82 * | 2.82   |
| I_FSM/FF7/CP (sdcrb1)            | 0.00   | 2.82 r |
| I_FSM/FF7/Q (sdcrb1)             | 4.40 * | 7.22 f |
| I_FSM/blender_clk_en (FSM)       | 0.00 * | 7.22 f |
| I2/D (slnlq1)                    | 0.00 * | 7.22 f |
| data arrival time                |        | 7.22   |
|                                  |        |        |
| clock SYS_CLK (fall edge)        | 4.00   | 4.00   |
| clock network delay (propagated) | 2.37 * | 6.37   |
| I2/EN (slnlq1)                   |        | 6.37 f |
| time borrowed from endpoint      | 0.85   | 7.22   |
| data required time               |        | 7.22   |
|                                  |        |        |
| data required time               |        | 7.22   |
| data arrival time                |        | -7.22  |
|                                  |        |        |
| slack (MET)                      | 0.00   |        |
|                                  |        |        |
| Time Borrowing Information       |        |        |
| SYS_CLK nominal pulse width      | 4.00   |        |
| clock latency difference         | 0.45   |        |
| library setup time               | -0.16  |        |
|                                  |        |        |
| max time borrow                  | 4.30   |        |
| actual time borrow               | 0.85   |        |

5-23

## Latches with Time Borrow: Path from L2/D

Time Borrow also means that the paths to and from L2/D are no longer independent.



5-24

Generate a timing report for both stages when a latch is experiencing time borrow.

`report_timing -trace_latch_borrow -to U1/A`

## Time Borrowed in Previous Stage is Now Returned [Old]

From the report:

The time borrowed in previous stage returned = 0.85

Start point of timing path = L2/D pin

D to Q time = 0.64

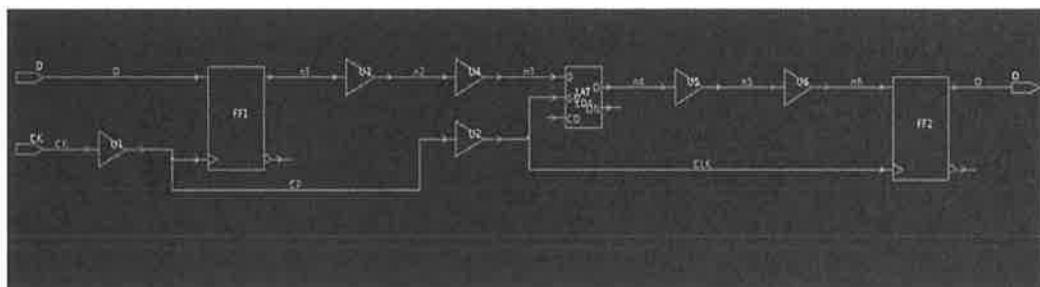
| Point                            | Incr   | Path    |
|----------------------------------|--------|---------|
| clock SYS_CLK (fall edge)        | 4.00   | 4.00    |
| clock network delay (propagated) | 2.37 * | 6.37    |
| time given to startpoint         | 0.85   | 7.22    |
| L2/D (slnlq1)                    | 0.00   | 7.22 f  |
| L2/Q (slnlq1)                    | 0.64 * | 7.87 f  |
| U1/A (ora21d4)                   | 0.00 * | 7.87 f  |
| data arrival time                |        | 7.87    |
| clock SYS_CLK (rise edge)        | 8.00   | 8.00    |
| clock network delay (propagated) | 2.09 * | 10.09   |
| U1/B (ora21d4)                   |        | 10.09 r |
| clock gating setup time          | -0.20  | 9.89    |
| data required time               |        | 9.89    |
| data arrival time                |        | -7.87   |
| slack (MET)                      |        | 2.02    |

5-25

## New Latch through Analysis

### Latches are either end points or through points

- Each latch data pin can be an endpoint causing latch clock pin to be startpoint [using Latch Clk → Q timing arc]
- Each latch data pin can be a through point causing the latch to function as a combinational delay cell [using Latch D → Q timing arc]
- The latch data pins are no longer valid startpoints [unlike old latch analysis]
- Allows improved CRPR, Leakage, and ECO fixing
- Checks out a *PrimeTime-ADV* (and *PrimeTime-SI*) license features



5-26

'PrimeTime-SI' is a pre-requisite of the 'PrimeTime-ADV' license.

## Old vs. New Latch Analysis : Differences

- Old Latch analysis behavior [Default]:

```
set_app_var timing_enable_through_paths false
```

- Latch D pin can be a start or an end point
  - NOT a through point
- Latch timing is performed w.r.t the transparency open edge

- New Latch analysis behavior:

```
set_app_var timing_enable_through_paths true
```

- Latch D pin can be a through or an end point, but,
  - NOT a start point
- Latch timing is performed w.r.t the transparency close edge

Refer to SolvNet Article: *PrimeTime ADV Next-Generation Latch Analysis Training* SolvNET Doc Id: 1540266 for more details



## OLD: Data arrives before transparency: Path to Latch/D

| Startpoint: FF1 (rising edge-triggered flip-flop clocked by CK) |        |         |
|-----------------------------------------------------------------|--------|---------|
| Endpoint: LAT (negative level-sensitive latch clocked by CK)    |        |         |
| Path Group: UK                                                  |        |         |
| Path Type: max                                                  |        |         |
| Point                                                           | Incr   | Path    |
| clock CK (rise edge)                                            | 0.000  | 0.000   |
| clock network delay (ideal)                                     | 0.030  | 0.030   |
| FF1/OP (FD1)                                                    | 0.000  | 0.000 F |
| FF1/O (FD1)                                                     | 1.579  | 1.579 F |
| U3/A (B11)                                                      | 0.000  | 1.579 F |
| U3/Z (B11)                                                      | 1.116  | 2.695 F |
| U4/W (B11)                                                      | 0.000  | 2.695 F |
| U4/Z (B11)                                                      | 1.104  | 3.799 F |
| LAT/D (LD4)                                                     | 0.000  | 3.799 F |
| data arrival time                                               |        | 3.799   |
| clock CK (fall edge)                                            | 5.000  | 5.000   |
| clock network delay (ideal)                                     | 0.000  | 5.000   |
| clock convergence pessimism                                     | 0.000  | 5.000   |
| LAT/DN (LD4)                                                    |        | 5.000 F |
| time borrowed from endpoint                                     | 0.000  | 5.000   |
| data required time                                              |        | 5.000   |
| data required time                                              | 5.000  |         |
| data arrival time                                               | -3.799 |         |
| slack (NET)                                                     |        | 1.201   |
| Time Borrowing Information                                      |        |         |
| CK nominal pulse width                                          | 5.000  |         |
| library setup time                                              | -0.500 |         |
| max time borrow                                                 | 4.500  |         |
| actual time borrow                                              | 0.000  |         |

- Latch is timed against the OPEN edge of the transparency window
- Contains Time borrowing Information
- (Actual) Time borrowed from end point is 0
- Positive slack

5-28

Old Latch analysis behavior

Data arrives at LAT/D before transparency open edge

## NEW: Data arrives before transparency: Path to Latch/D

| Startpoint: FFI (rising edge-triggered flip-flop clocked by CK) |        |          |
|-----------------------------------------------------------------|--------|----------|
| Endpoint: LAT (negative level-sensitive latch clocked by CK)    |        |          |
| Path Group: Of                                                  |        |          |
| Path Type: max                                                  |        |          |
| Point                                                           | Incr   | Path     |
| clock (K) (rise edge)                                           | 0,000  | 0,000    |
| clock network delay (ideal)                                     | 0,000  | 0,000    |
| FFI/CP (FFI)                                                    | 0,000  | 0,000 r  |
| FFL/D (FFI)                                                     | 1,579  | 1,579 f  |
| U3/H (B1)                                                       | 0,000  | 1,579 f  |
| U3/Z (B1)                                                       | 1,116  | 2,695 f  |
| U4/H (B1)                                                       | 0,000  | 2,695 f  |
| U4/Z (B1)                                                       | 1,104  | 3,799 f  |
| LAT/D (LAT)                                                     | 0,000  | 3,799 f  |
| data arrival time                                               |        | 3,799    |
| clock (K) (fall edge)                                           |        | 5,000    |
| clock latency                                                   | 0,000  | 5,000    |
| transparency open edge                                          |        | 5,000    |
| clock (K) (rise edge)                                           | 10,000 | 10,000   |
| clock network delay (ideal)                                     | 0,000  | 10,000   |
| clock recorvergence pessimism                                   | 0,000  | 10,000   |
| LAT/ON (LAT)                                                    |        | 10,000 r |
| library setup time                                              | -0,500 | 9,500    |
| transparency close edge                                         |        | 9,500    |
| data required time                                              |        | 9,500    |
| data required time                                              |        | 9,500    |
| data arrival time                                               |        | -3,799   |
| slack (MET)                                                     |        | 5,701    |

- Latch is timed against the CLOSE edge of the transparency window
- Does NOT contain time borrowing information
- Slack is more positive

5-29

```
set_app_var timing_enable_through_paths true
```

New Latch analysis behavior

Data arrives at LAT/D before transparency open edge

## New: Data arrives during transparency: -to LAT/D

Startpoint: FF1 (rising edge-triggered flip-flop clocked by CK)  
 Endpoint: LAT (negative level-sensitive latch clocked by CK)  
 Path Group: CK  
 Path Type: max

| Point                          | Incr     | Path          |
|--------------------------------|----------|---------------|
| clock CK (rise edge)           | 0,0000   | 0,0000        |
| clock network delay (ideal)    | 0,0000   | 0,0000        |
| FF1/CP (FDI)                   | 0,0000   | 0,0000 r      |
| FF1/O (BII)                    | 1,6732   | 1,6732 r      |
| U3/A (BII)                     | 0,0000   | 1,6732 r      |
| U3/Z (BII)                     | 0,5000 * | 2,1732 r      |
| U4/R (BII)                     | 0,0000   | 2,1732 r      |
| U4/Z (BII)                     | 0,5000 * | 2,6732 r      |
| LAT/GN (LBH)                   | 0,0000   | 2,6732        |
| <b>data arrival time</b>       |          | <b>2,6732</b> |
| clock CK (fall edge)           | 2,0000   |               |
| clock latency                  | 0,0000   | 2,0000        |
| <b>transparency open edge</b>  |          | <b>2,0000</b> |
| clock CK (rise edge)           | 4,0000   | 4,0000        |
| clock network delay (ideal)    | 0,0000   | 4,0000        |
| clock reconvergence pessimism  | 0,0000   | 4,0000        |
| LAT/GN (LBH)                   | 0,5000   | 4,0000 r      |
| <b>transparency close edge</b> |          | <b>3,5000</b> |
| data required time             | 3,5000   |               |
| data required time             | 3,5000   |               |
| data arrival time              | -2,6732  |               |
| slack (MET)                    | 0,8268   |               |

- **Data arrival to LAT/D is in between transparency open and close edges**
- **Flop to latch path is timed against close edge**
- **Slack is positive**

## New: Data arrives after transparency: -to LAT/D

| Startpoint: FF1 (rising edge-triggered flip-flop clocked by CK) |         |            |
|-----------------------------------------------------------------|---------|------------|
| Endpoint: LAT (negative level-sensitive latch clocked by CK)    |         |            |
| Path Group: CK                                                  |         |            |
| Path Type: max                                                  |         |            |
| Point                                                           | Incr    | Path       |
| clock CK (rise edge)                                            | 0.0000  | 0.0000     |
| clock network delay (ideal)                                     | 0.0000  | 0.0000     |
| FF1/CP (FD1)                                                    | 0.0000  | 0.0000 r   |
| FF1/Q (FD1)                                                     | 1.6732  | 1.6732 r   |
| U3/Y (B11)                                                      | 0.0000  | 1.6732 r   |
| U3/Z (B11)                                                      | 1.5000  | * 3.1732 r |
| U4/Y (B11)                                                      | 0.0000  | 3.1732 r   |
| U4/Z (B11)                                                      | 1.5000  | * 4.6732 r |
| LAT/D (LD4)                                                     | 0.0000  | 4.6732 r   |
| data arrival time                                               |         | 4.6732     |
| clock CK (fall edge)                                            |         | 2.0000     |
| clock latency                                                   | 0.0000  | 2.0000     |
| transparency open edge                                          |         | 2.0000     |
| clock CK (rise edge)                                            | 4.0000  | 4.0000     |
| clock network delay (ideal)                                     | 0.0000  | 4.0000     |
| clock reconvergence pessimism                                   | 0.0000  | 4.0000     |
| LAT/GN (LD4)                                                    |         | 4.0000 r   |
| library setup time                                              | -0.5000 | 3.5000     |
| transparency close edge                                         |         | 3.5000     |
| data required time                                              |         | 3.5000     |
| data required time                                              |         | 3.5000     |
| data arrival time                                               |         | -4.6732    |
| slack (VIOLATED)                                                |         | -1.1732    |

- Data arrival to LAT/D is after transparency close edge
- Slack is negative

## New: Early path - through LAT/D (Ideal Clock)

|                                                                 |         |          |
|-----------------------------------------------------------------|---------|----------|
| Startpoint: FF1 (rising edge-triggered flip-flop clocked by CK) |         |          |
| Endpoint: FF2 (rising edge-triggered flip-flop clocked by CK)   |         |          |
| Path Group: CK                                                  |         |          |
| Path Type: max                                                  |         |          |
| Point                                                           | Incr    | Path     |
| clock CK (rise edge)                                            | 0,0000  | 0,0000   |
| clock network delay (ideal)                                     | 0,0000  | 0,0000   |
| FF1/CP (FD1) <-                                                 | 0,0000  | 0,0000 r |
| U3/A (B11)                                                      | 0,1000  | 0,1000 f |
| U3/Z (B11)                                                      | 0,1000  | 0,2000 f |
| U4/A (B11)                                                      | 0,0000  | 0,2000 f |
| U4/Z (B11)                                                      | 0,1000  | 0,3000 f |
| LAT/D (LD4)                                                     | 0,0000  | 0,3000 f |
| Transparency window #1 (early)                                  |         |          |
| clock CK (fall edge)                                            |         | 2,0000   |
| clock latency                                                   | 0,0000  | 2,0000   |
| transparency open edge                                          |         | 2,0000   |
| clock CK (rise edge)                                            |         | 4,0000   |
| clock latency                                                   | 0,0000  | 4,0000   |
| library setup time                                              | -0,5000 | 3,5000   |
| transparency close edge                                         |         | 3,5000   |
| available borrow at through pin                                 | 3,2000  |          |
| LAT/D (LD4)                                                     | 0,0000  | 0,3000 f |
| LAT/Z (LD4)                                                     | 1,8272  | 1,8272 f |
| U5/A (B11)                                                      | 0,0000  | 1,8272 f |
| U6/Z (B11)                                                      | 1,1160  | 2,9432 f |
| U6/A (B11)                                                      | 0,0000  | 2,9432 f |
| U6/Z (B11)                                                      | 1,1040  | 4,0472 f |
| FF2/CP (FD1)                                                    | 0,0000  | 4,0472 f |
| data arrival time                                               |         | 4,0472   |
| clock CK (rise edge)                                            | 4,0000  | 4,0000   |
| clock network delay (ideal)                                     | 0,0000  | 4,0000   |
| clock reconvergence pessimism                                   | 0,0000  | 4,0000   |
| FF2/CP (FD1)                                                    | 0,0000  | 4,0000 r |
| library setup time                                              | -0,8000 | 3,2000   |
| data required time                                              |         | 3,2000   |
| data required time                                              |         | 3,2000   |
| data arrival time                                               |         | -4,0472  |
| slack (VIOLATED)                                                |         | -0,8472  |

- LAT/D is a through pin
- Clock is ideal
- Timing path is in between flop to flop through latch
- Report contains open and close edges of Transparency window [early path]
- Arrival at LAT/Q is sum of data arrival at LAT/D and D-Q delay of latch
- Keyword “early” indicates a “what-if” analysis

5-32

Since data arrives before the transparency period of latch, this path is also called early path.

report\_timing -inp -from FF1/CP -to FF2/D

## New: Early path - through LAT/D (Propagated Clock)

| Startpoint: FF1 (rising edge-triggered flip-flop clocked by CK)<br>Endpoint: FF2 (rising edge-triggered flip-flop clocked by CK)<br>Last common pin: CK<br>Path Group: CK<br>Path Type: max |         |            |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|------------|
| Point                                                                                                                                                                                       | Inter   | Path       |
| clock CK (rise edge)                                                                                                                                                                        | 0.0000  | 0.0000     |
| clock network delay (propagated)                                                                                                                                                            | 0.5795  | 0.5795     |
| FF1/Q (FD1) <-                                                                                                                                                                              | 0.0000  | 0.5795 f   |
| U5/A (B11)                                                                                                                                                                                  | 0.1000  | = 0.6795 f |
| U5/Z (B11)                                                                                                                                                                                  | 0.1000  | = 0.7795 f |
| U4/R (B11)                                                                                                                                                                                  | 0.0000  | = 0.7795 f |
| U4/Z (B11)                                                                                                                                                                                  | 0.1000  | = 0.8795 f |
| LAT/D (LD4)                                                                                                                                                                                 | 0.0000  | = 0.8795 f |
| <b>transparency window #1 (early)</b>                                                                                                                                                       |         |            |
| clock CK (fall edge)                                                                                                                                                                        | 2.0000  |            |
| clock latency                                                                                                                                                                               | 2.2280  | 4.2280     |
| transparency open edge                                                                                                                                                                      |         | 4.2280     |
| clock CK (rise edge)                                                                                                                                                                        | 4.0000  |            |
| clock latency                                                                                                                                                                               | 1.1413  | 5.1413     |
| library setup time                                                                                                                                                                          | -0.5000 | 4.6413     |
| transparency close edge                                                                                                                                                                     |         | 4.6413     |
| available borrow at through pin                                                                                                                                                             | 3.7618  |            |
| LAT/D (LD4)                                                                                                                                                                                 | 0.0000  | 0.8795 f   |
| LAT/Q (LD4)                                                                                                                                                                                 | 1.5272  | 2.4067 f   |
| U5/A (B11)                                                                                                                                                                                  | 0.0000  | 2.4067 f   |
| U5/Z (B11)                                                                                                                                                                                  | 1.1160  | 3.5227 f   |
| U6/A (B11)                                                                                                                                                                                  | 0.0000  | 3.5227 f   |
| U6/Z (B11)                                                                                                                                                                                  | 1.1160  | 4.3327 f   |
| FF2/D (FD1)                                                                                                                                                                                 | 0.0000  | 4.3327 f   |
| data arrival time                                                                                                                                                                           |         | 4.6267     |
| clock CK (rise edge)                                                                                                                                                                        | 4.0000  | 4.0000     |
| clock network delay (propagated)                                                                                                                                                            | 1.1413  | 5.1413     |
| clock reconvergence pessimism                                                                                                                                                               | 0.0000  | 5.1413     |
| FF2/D (FD1)                                                                                                                                                                                 |         | 5.1413 r   |
| library setup time                                                                                                                                                                          | -0.8000 | 4.3413     |
| data required time                                                                                                                                                                          |         | 4.3413     |
| data required time                                                                                                                                                                          | 4.3413  |            |
| data arrival time                                                                                                                                                                           |         | -4.6267    |
| slack (VIOLATED)                                                                                                                                                                            |         | -0.2854    |

- LAT/D is a through pin
- Clock is propagated. The slack has improved
- Transparency window shows different latency for open and close edges.
- Use report\_clock\_timing -clock \$clk -skew -verb to understand latency calculation at each edge
- Keyword “early” indicates a “what-if” analysis

5- 33

We can skip the early paths from reporting by setting the below variable

```
set timing_report_skip_early_paths_at_intermediate_latches true
```

Default value is false.

When enabled, for the kind of paths described in the above slide, the latch is no longer a through path. the FF1 through LAT to FF2 path wont be returned by report\_timing or get\_timing\_paths

## New: Data arrives during transparency: -through LAT/D

| Startpoint: FF1 (rising edge-triggered flip-flop clocked by CK)<br>Endpoint: FF2 (rising edge-triggered flip-flop clocked by CK)<br>Path Group: CK<br>Path Type: Max |          |          |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------|----------|
| Point                                                                                                                                                                | Incr     | Path     |
| clock CK (rise edge)                                                                                                                                                 | 0.0000   | 0.0000   |
| clock network delay (ideal)                                                                                                                                          | 0.0000   | 0.0000   |
| FF1/CP (FD1) <-                                                                                                                                                      | 0.0000   | 0.0000 r |
| U3/W (BIT)                                                                                                                                                           | 1.5792   | 1.5792 f |
| U3/Z (BIT)                                                                                                                                                           | 0.0000   | 1.5792 f |
| U4/W (BIT)                                                                                                                                                           | 0.5000 * | 2.0792 f |
| U4/Z (BIT)                                                                                                                                                           | 0.0000   | 2.0792 f |
| U4/W (BIT)                                                                                                                                                           | 0.5000 * | 2.5792 f |
| LAT/D (LD4)                                                                                                                                                          | 0.0000   | 2.5792 f |
| transparency window #1                                                                                                                                               |          |          |
| clock CK (fall edge)                                                                                                                                                 | 2.0000   |          |
| clock latency                                                                                                                                                        | 0.0000   | 2.0000   |
| transparency open edge                                                                                                                                               |          | 2.0000   |
| clock CK (rise edge)                                                                                                                                                 | 4.0000   |          |
| clock latency                                                                                                                                                        | 0.0000   | 4.0000   |
| library setup time                                                                                                                                                   | -0.5000  | 3.5000   |
| transparency close edge                                                                                                                                              |          | 3.5000   |
| available borrow at through pin                                                                                                                                      | 0.9208   |          |
| LAT/D (LD4)                                                                                                                                                          | 0.0000   | 2.5792 f |
| LAT/D (LD4)                                                                                                                                                          | 1.5272   | 4.1064 f |
| U5/W (BIT)                                                                                                                                                           | 0.0000   | 4.1064 f |
| U5/Z (BIT)                                                                                                                                                           | 1.1160   | 5.2224 f |
| U6/W (BIT)                                                                                                                                                           | 0.0000   | 5.2224 f |
| U6/Z (BIT)                                                                                                                                                           | 1.1040   | 6.3264 f |
| FF2/CP (FD1)                                                                                                                                                         | 0.0000   | 6.3264 f |
| data arrival time                                                                                                                                                    |          | 6.3264   |
| clock CK (rise edge)                                                                                                                                                 | 4.0000   | 4.0000   |
| clock network delay (ideal)                                                                                                                                          | 0.0000   | 4.0000   |
| clock reconvergence pessimism                                                                                                                                        | 0.0000   | 4.0000   |
| FF2/CP (FD1)                                                                                                                                                         |          | 4.0000 r |
| library setup time                                                                                                                                                   | -0.8000  | 3.2000   |
| data required time                                                                                                                                                   |          | 3.2000   |
| data required time                                                                                                                                                   | 3.2000   |          |
| data arrival time                                                                                                                                                    | -6.3264  |          |
| slack (VIOLATED)                                                                                                                                                     | -3.1264  |          |

- LAT/D is a through pin
- Clock is ideal
- Report contains open and close edges of Transparency window
- Arrival at LAT/Q is the sum of data arrival at LAT/D and D-Q delay of latch
- Timing violation

5-34

## New: Recovery path - through LAT/D

| Startpoint: FF1 (rising edge-triggered flip-flop clocked by CK) |          |          |
|-----------------------------------------------------------------|----------|----------|
| Endpoint: FF2 (rising edge-triggered flip-flop clocked by CK)   |          |          |
| Path Group: CK                                                  |          |          |
| Path Type: max                                                  |          |          |
| Point                                                           | Incr     | Path     |
| clock CK (rise edge)                                            | 0.0000   | 0.0000   |
| clock network delay (ideal)                                     | 0.0000   | 0.0000   |
| FF1/CP (FD1) <-                                                 | 0.0000   | 0.0000 r |
| U3/Z (B1I)                                                      | 1.5792   | 1.5792 f |
| U3/A (B1I)                                                      | 0.0000   | 1.5792 f |
| U3/Z (B1I)                                                      | 1.5000 * | 3.0792 f |
| U4/Z (B1I)                                                      | 0.0000   | 3.0792 f |
| U4/Z (B1I)                                                      | 1.5000 * | 4.5792 f |
| LAT/D (LD4)                                                     | 0.0000   | 4.5792 f |
| <b>transparency window #1 (missed)</b>                          |          |          |
| clock CK (fall edge)                                            | 2.0000   |          |
| clock latency                                                   | 0.0000   | 2.0000   |
| transparency open edge                                          |          | 2.0000   |
| clock CK (rise edge)                                            | 4.0000   |          |
| clock latency                                                   | 0.0000   | 4.0000   |
| library setup time                                              | -0.5000  | 3.5000   |
| transparency close edge                                         |          | 3.5000   |
| LAT/D (LD4)                                                     | 4.5792   |          |
| adjustment to close edge                                        | -1.0792  | 3.5000   |
| LAT/D (LD4)                                                     | 0.0000   | 3.5000 f |
| LAT/D (LD4)                                                     | 1.5272   | 5.0272 f |
| U5/A (B1I)                                                      | 0.0000   | 5.0272 f |
| U5/Z (B1I)                                                      | 1.1160   | 6.1432 f |
| U6/A (B1I)                                                      | 0.0000   | 6.1432 f |
| U6/Z (B1I)                                                      | 1.1040   | 7.2472 f |
| FF2/CP (FD1)                                                    | 0.0000   | 7.2472 f |
| data arrival time                                               |          | 7.2472   |
| clock CK (rise edge)                                            | 4.0000   | 4.0000   |
| clock network delay (ideal)                                     | 0.0000   | 4.0000   |
| clock convergence pessimism                                     | 0.0000   | 4.0000   |
| FF2/CP (FD1)                                                    | 4.0000   | 4.0000 r |
| library setup time                                              | -0.8000  | 3.2000   |
| data required time                                              |          | 3.2000   |
| data required time                                              |          | 3.2000   |
| data arrival time                                               |          | -7.2472  |
| slack (VIOLATED)                                                |          | -4.0472  |

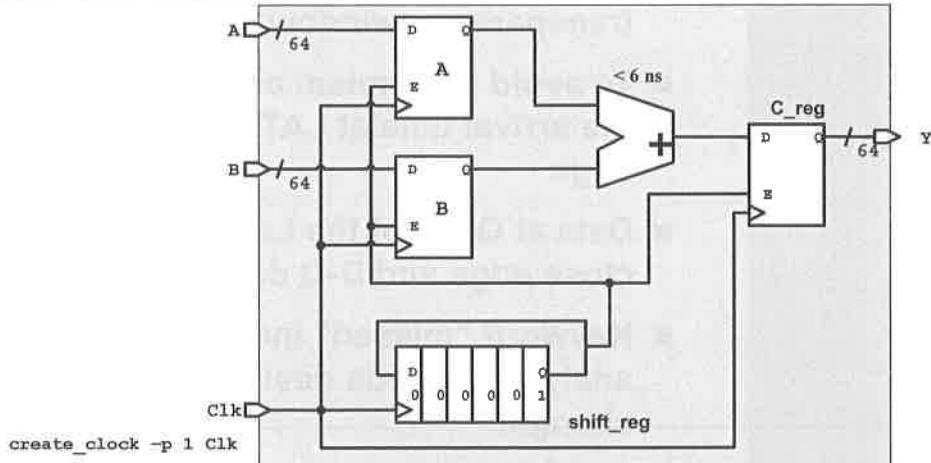
- This is a recovery path, since the transparency window is missed
- To avoid pessimism at downstream logic, the arrival time at LAT/D is limited to close edge
- Data at Q pin of the Latch is the sum of close edge and D-Q delay of the latch.
- Keyword “missed” indicates a “what-if” analysis – Needs design and/or constraint change!

5-35

The path is called recovery path. The word ‘missed’ here indicates that. And there’s also an ‘is\_recovered’ path attribute you can query.

## 2. Multicycle Paths

Circle the multicycle paths.



PrimeTime does not automatically  
identifies multicycle paths!

5-36

circle the multicycle paths – through the adder

ANSWERS:

## Default reported path is single cycle

| Point                                                 | Incr    | Path    |
|-------------------------------------------------------|---------|---------|
| clock SYS_2x_CLK (rise edge)                          | 0.000   | 0.000   |
| clock network delay (propagated)                      | 2.850 & | 2.850   |
| I_INSTRN_LAT/Crnt_Instrn_1_reg[26]/CP (sdnrb1)        | 0.000   | 2.850 r |
| .....                                                 |         |         |
| I_RISC_CORE/I_ALU/Zro_Flag_reg/D (sdcrb1)             | 0.000 & | 6.917 r |
| data arrival time                                     |         | 6.917   |
| clock SYS_2x_CLK (rise edge)                          | 1.000   | 1.000   |
| clock network delay (propagated)                      | 2.838 & | 3.838   |
| I_ORCA_TOP/I_RISC_CORE/I_ALU/Zro_Flag_reg/CP (sdcrb1) |         | 3.838 r |
| library setup time                                    | -0.191  | 3.647   |
| data required time                                    |         | 3.647   |
| .....                                                 |         |         |
| data required time                                    |         | 3.647   |
| data arrival time                                     |         | -6.917  |
| .....                                                 |         |         |
| slack (VIOLATED)                                      |         | -3.270  |

5-37

## Specifying Multicycle Path for Setup

```
set_multicycle_path 6 -from reg[26]/CP -to reg/D
report_exceptions
```

```

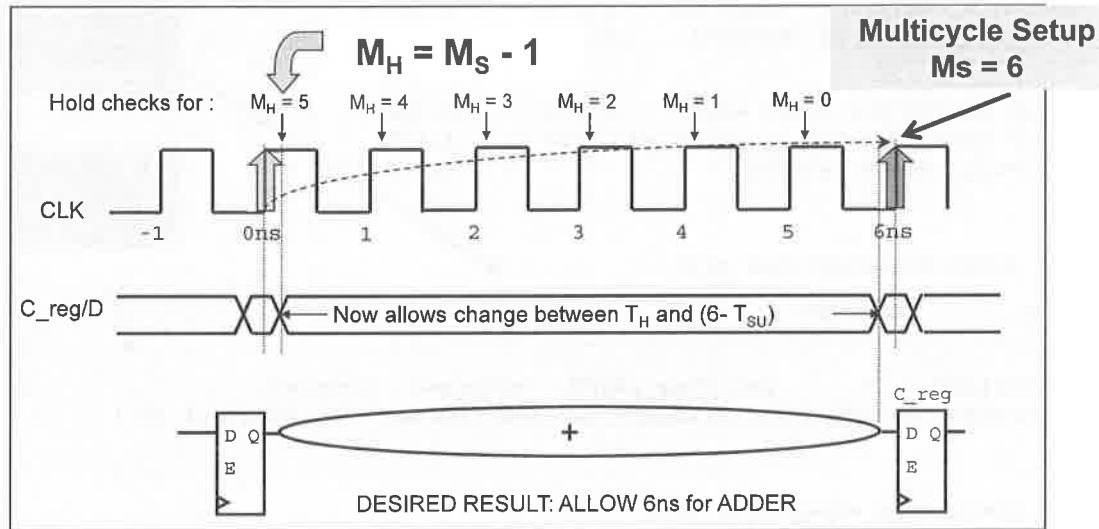
From Through To Setup Hold
reg[26]/CP * reg/D cycles=6
```

| Point                        | Incr | Path  |
|------------------------------|------|-------|
| clock SYS_2x_CLK (rise edge) | 0.00 | 0.00  |
| .....                        |      | 6.92  |
| clock SYS_2x_CLK (rise edge) | 6.00 | 6.00  |
| .....                        |      | 8.65  |
| data required time           |      | 8.65  |
| data arrival time            |      | -6.92 |
| -----                        |      |       |
| slack (MET)                  |      | 1.73  |

5-38

## Specifying Multicycle path for Hold (New data every 6 cycles)

```
set_multicycle_path -setup 6 -to [get_pins "*reg[*]/D"]
set_multicycle_path -hold 5 -to [get_pins "*reg[*]/D"]
```



5-39

MH stands for Hold Multiplier, MS for Setup Multiplier. The Setup multiplier counts up with increasing clock cycles, the Hold multiplier counts up with decreasing cycles. The origin (0) for the Hold Multiplier is always at the Setup Multiplier – 1 position.

## Reporting a multi cycle path with report\_timing

```
report_timing -exceptions all -from *reg[26]/CP -to *reg/D
```

```
clock SYS_2x_CLK (rise edge) 0.000 0.000
clock network delay (propagated) 2.850 & 2.850
Instrn_1_reg[26]/CP (sdnrb1) 0.000 2.850 r
Instrn_1_reg[26]/Q (sdnrb1) 0.699 & 3.549 f
...
```

Exceptions (multicycle path, false path, max/min delay) affecting reported path.

```
clock SYS_2x_CLK (rise edge) 6.000 6.000
clock network delay (propagated) 2.838 & 8.838
Zro_Flag_reg/CP (sdcrb1) 6.838 r
```

Displays file name and line number setting the exception

The dominant exceptions are:

| From | Through | To | Setup | Hold |
|------|---------|----|-------|------|
|------|---------|----|-------|------|

```
reg[26]/CP * Zro_Flag_reg/D cycles=6 cycles=5
[location = /remote/training/projects/PrimeTime_1/example.tcl:23]
```

The overridden exceptions are:

None

5-40

Arguments to -exceptions are 'all', 'dominant', and 'overridden'

In order for PrimeTime to display source file names and line numbers for timing exceptions, you must tell PrimeTime to save that information BEFORE you source in any exceptions, and BEFORE any exceptions are read in with the pt\_shell -f command.

Tell PrimeTime to save the exceptions information as follows:

```
set sdc_save_source_file_information true
```

The source file information is used by both the report\_exceptions and the report\_timing -exceptions commands.

When will a dominant false path exception appear in a timing report - B

ANSWER:

### 3. Combinational Feedback Loops

One of the default checks!

```
pt_shell> check_timing -verbose
Information: Checking 'loops'.
Warning: There is 1 timing loop in the design.

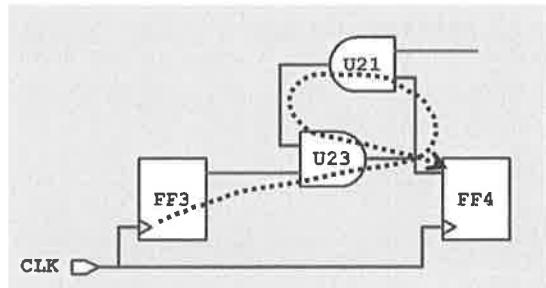
Pin (timing loop #1)

U23/Y
U21/B
U21/Y
U23/A
U23/Y

```

## STA and Combinational Loops

- Combinational loops are NOT verifiable using STA
- PrimeTime will disable a single timing arc to break this loop



How will you verify timing  
of the combo loop?

5-42

You must verify the timing through the combo loop using a dynamic simulation tool.

If the timing arc that should be broken is known – it is recommended that you specify this arc using **set\_disable\_timing** and to not depend on PrimeTime to automatically break an arc. This will guarantee consistency between STA tools.



```
pt_shell> set_disable_timing -help
set_disable_timing # Disable timing arcs
 [-from from_pin_name] (From pin on cell)
 [-to to_pin_name] (To pin on cell)
 object_list (List of cells or pins, ports,
 lib-cells, or lib-
 pins)
```

dynamic loop breaking feature  
how will you verify timing of the combo loop – use **set\_disable\_timing** or turn on PTs  
Answer:

## Issues with Combinational Loops

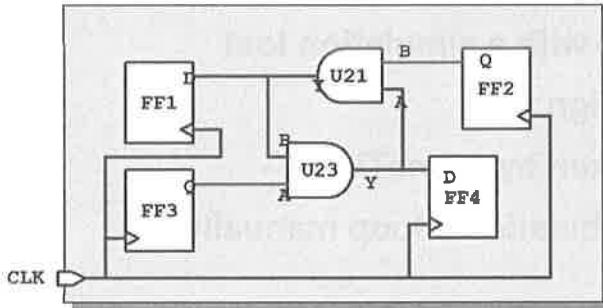
- Verify timing of combo loop with a simulation tool
- Inform the owner of the design
- Examine the timing arc broken by PrimeTime
- If necessary, break the combinational loop manually



Circle the command to find the disabled timing arc broken by PrimeTime:

```
report_disable_timing "U21 U23"
report_disable_timing "U23/Y"
```

## Which Arc is Broken by PrimeTime?



Interpret the report below and then indicate the disabled timing arc in the schematic.

```
pt_shell> report_disable_timing "U21 U23"
```

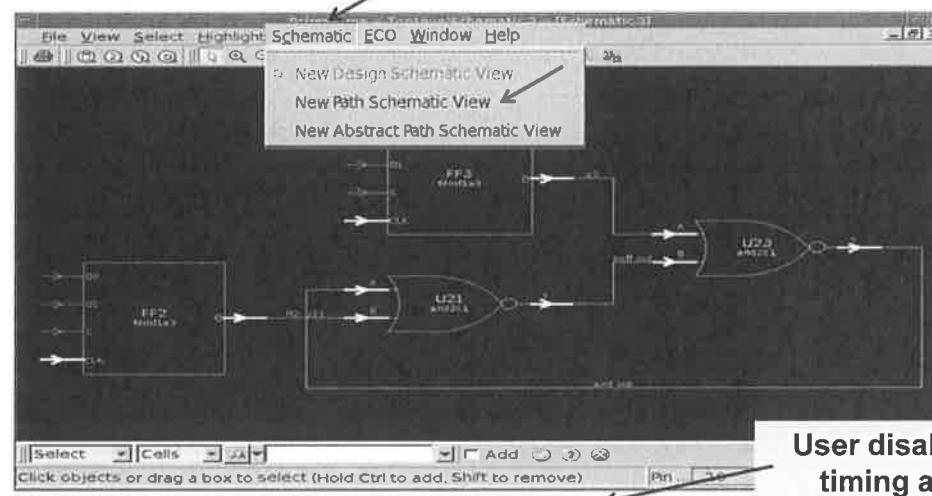
```
Flags : 1 loop breaking
```

| Cell or Port | From | To | Sense | Flag | Reason |
|--------------|------|----|-------|------|--------|
| U23          | B    | Y  | *     | 1    |        |

5-44

## Examine Path with Broken Arc

```
pt_shell> all_fanin -to U23/B -flat -trace_arcs all
{ "U23/B", "U21/Y", "U21/A", "U21/B", "U23/Y", "FF2/Q", "U23/A", "FF2/CLK", "FF3/Q", "FF3/CLK" }
pt_shell> change_selection [all_fanin -to U23/B -flat -trace_arcs all]
pt_shell> start_gui
```



5-45

If possible, break the loop by choosing a cell that does not break any other valid timing path.

## 4. Non-Unate Cells (Arcs) in clock paths

Information: A non-unate path in clock network detected.  
Propagating both inverting and noninverting senses of clock  
'CLK\_TEST' from pin 'AND/Z'. (PTE-070)

A non-unate path is one where:

- The edge sensitivity is unclear in the library.
- The edge sensitivity is both positive and negative unate.

By default, PrimeTime  
propagates both senses

5-46

pt\_shell> man PTE-070

### NAME

PTE-070 (information) A non-unate path in clock network detected. Propagating both inverting and noninverting senses of clock '%s' from pin '%s'.

### DESCRIPTION

The clock tree for the specified clock contains non-unate paths. Clock networks normally do not contain cells such as exclusive-OR gates which have non-unate behavior. This implies that both inverting and noninverting clock waveforms reach the specified pin. This is an informational message that such a pin has been detected and that PrimeTime is propagating both senses of clock.

### WHAT NEXT

If you want to control the sense of the clock used through this pin, use the command 'set\_clock\_sense'.

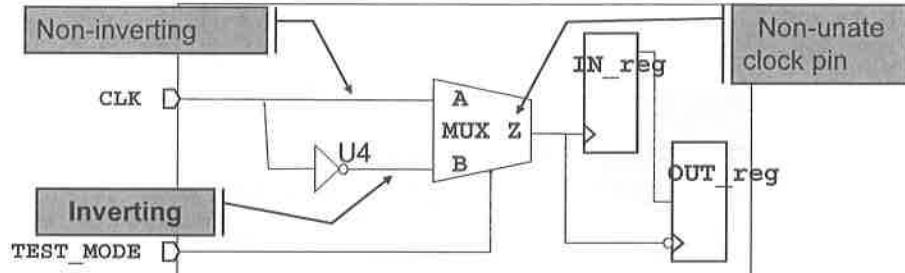
happens in real life

Under what solutions will this solution not work - you will get half-cycle failures that will not

Explains the solution PT will employ - it will waveforms starting with each edge

a non-unate path is one where - the edge sensitivity is both positive and negative unate

## PTE-070 non-unateness and set\_sense



Information: A non-unate path in clock network detected.  
Propagating both inverting and noninverting senses  
of clock 'CLK' from pin 'MUX/Z'. (PTE-070)

In this example, a setup check uses:  
slowest (inverting) path for launch  
fastest (non-inverting) path for capture

You can restrict the clock sense with one  
of these set\_sense commands:

← pessimistic  
`set_sense -positive MUX/Z ;# From MUX/Z,  
propagate the non-inverting (pin A) path`

`set_sense -negative MUX/Z ;# From MUX/Z,  
propagate the inverting (pin B) path`

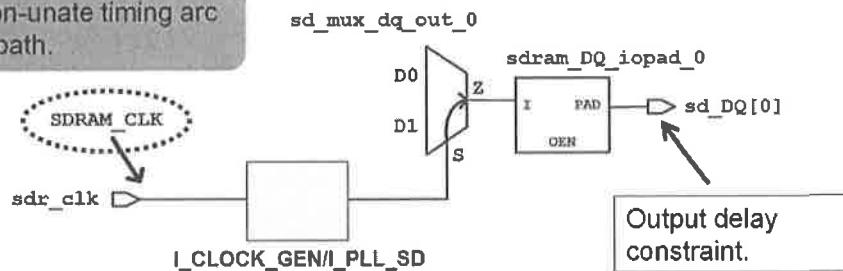
5-47

set\_case\_analysis can also be used to restrict the clock sense

Use remove\_sense to undo a set\_sense command

## Clock Used as Data

Circle the non-unate timing arc in the clock path.



PrimeTime propagates both senses when the clock is used as data

5-48

circle the non-unate timing arc in the clock path – mux s to Z arc

ANSWER:

## Review of Unit Objectives



**Having completed this unit, you should be able to:**

■ **Exercise the following timing checks during STA**

- I. Recovery, Removal
- II. Clock Gating Checks
- III. Data to Data Checks
- IV. Minimum Pulse Width Checks

■ **Analyze timing that involves**

1. Latches with Time borrowing
2. Multi cycle paths
3. Combinational feedback loops
4. Non unate cells along clock paths

## Lab 5: Additional Checks and Constraints



30 minutes

Apply user specified annotated delays to explore  
time borrowing with latches



Debug PTE-070 messages

5-50

# Agenda

DAY

2

4 Constraining Multiple Clocks



5 Additional Checks and Constraints



6 Best Practices Debugging Reports

7 Signoff: Path Based Analysis (PBA)



6-1

## Unit Objectives



**Having Completed this unit, you should be able to:**

- Identify key pieces of a timing report showing constraint issues
- Debug Timing Reports when the findings are incorrect/questionable

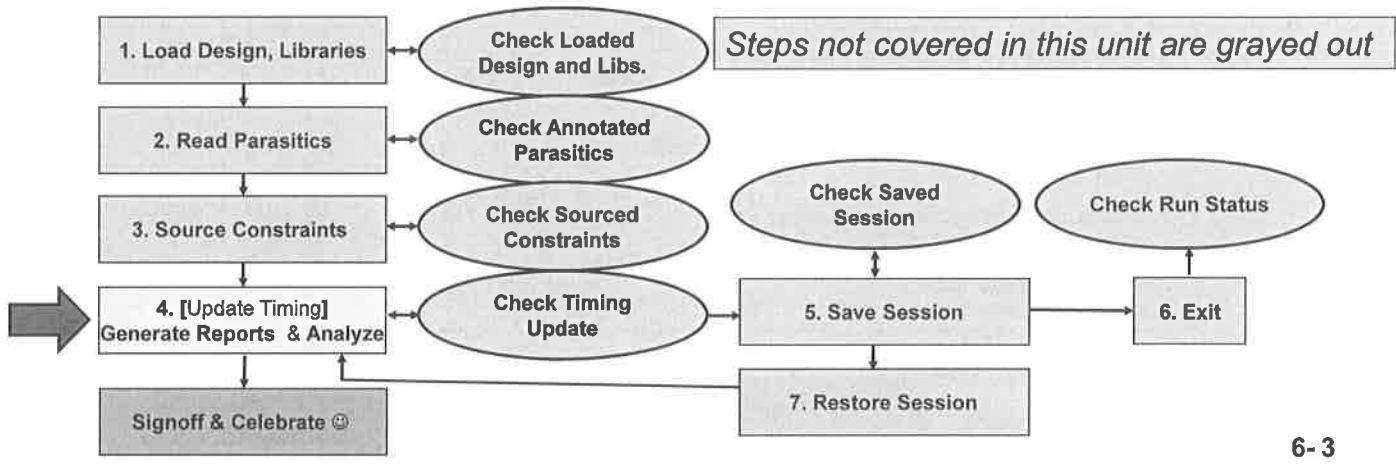
6-2

# Timing Analysis Flow in PrimeTime – Generating Reports

- STA flow is divided into several steps

- Steps 1-3 read data and
- Analysis begins at Step-4

- The STA flow is repeated until signoff is achieved



6-3

STA: Static Timing Analysis

## Do NOT start analysis with report\_timing

| Startpoint: pc_be[0] (input port)                                                                      | Incr    | Path    |
|--------------------------------------------------------------------------------------------------------|---------|---------|
| Endpoint: I_ORCA_TOP/I_PCI_READ_FIFO/U981<br>(rising edge-triggered flip-flop clocked by test_PCI_CLK) |         |         |
| Path Group: test_PCI_CLK                                                                               |         |         |
| Path Type: max                                                                                         |         |         |
| Point                                                                                                  |         |         |
| clock (input port clock) (rise edge)                                                                   | 0.00    | 0.00    |
| input external delay                                                                                   | 45.00   | 45.00 r |
| pc_be[0] (inout)                                                                                       | 0.00    | 45.00 r |
| pc_be_iopad_0/PAD (PCI66DGZ)                                                                           | 0.00 &  | 45.00 r |
| pc_be_iopad_0/C (PCI66DGZ)                                                                             | 0.95 &  | 45.95 r |
| I_ORCA_TOP/pc_be_in[0] (ORCA_TOP)                                                                      | 0.00 &  | 45.95 r |
| I_ORCA_TOP/I_PCI_CORE/pc_be_in[0] (PCI_CORE)                                                           | 0.00 &  | 45.95 r |
| I_ORCA_TOP/I_PCI_CORE/U492/Y (invla27)                                                                 | 0.07 &  | 46.02 f |
| .                                                                                                      |         |         |
| I_ORCA_TOP/I_PCI_READ_FIFO/U981/D0 (fdmf1a6)                                                           | 0.00 &  | 48.83 r |
| data arrival time                                                                                      |         | 48.83   |
| clock test_PCI_CLK (rise edge)                                                                         | 45.00   | 45.00   |
| clock network delay (propagated)                                                                       | 3.84 &  | 48.84   |
| I_ORCA_TOP/I_PCI_READ_FIFO/U981/CLK (fdmf1a6)                                                          |         | 48.84 r |
| library setup time                                                                                     | -0.18 & | 48.66   |
| data required time                                                                                     |         | 48.66   |
| data required time                                                                                     |         | 48.66   |
| data arrival time                                                                                      |         | -48.83  |
| slack (VIOLATED)                                                                                       |         | -0.17   |

Is this violation real?



6-4

## Much Easier With check\_timing

```
pt_shell> set_app_var timing_input_port_default_clock true
pt_shell> check_timing -verbose
Information: Checking 'no_input_delay'.
Warning: There are 14 ports with no clock-relative input delay
specified.
Since the variable 'timing_input_port_default_clock' is 'true',
a default input port clock will be assumed for these ports.

Ports

pc_be[0]
pc_be[1]
pc_be[2]
pc_be[3]
```

6-5

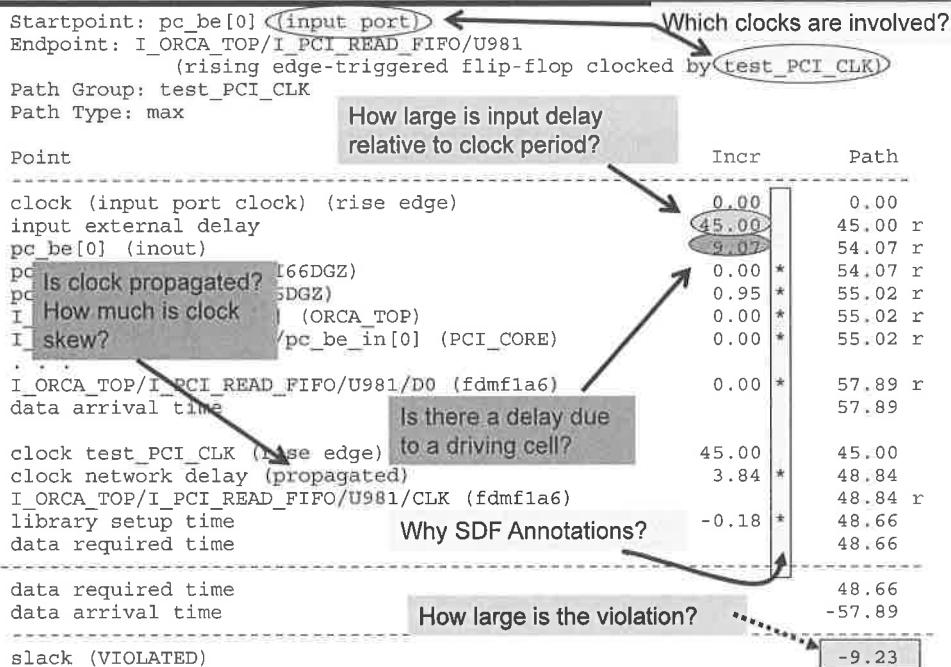
An example constraint that could cause the above warning and timing report on the previous page:

```
Command is missing a reference clock and thus is incorrectly specified:
set_input_delay 45 [get_ports pc_be*]
```

```
pt_shell> report_port -input_delay [get_ports pc_be*]
 Input Delay
 Min Max Related
Input Port Rise Fall Rise Fall Clock

pc_be[3] 45.00 45.00 45.00 45.00 --
pc_be[2] 45.00 45.00 45.00 45.00 --
pc_be[1] 45.00 45.00 45.00 45.00 --
pc_be[0] 45.00 45.00 45.00 45.00 --
```

## Hints on Debugging report\_timing



6-6

To separate net and cell delays use:

**report\_timing -input\_pins**

# Test For Understanding



Circle the pieces in the timing report that could indicate constraint problems.

Startpoint: I\_ORCA\_TOP/I\_BLENDER/latched\_clk\_en\_reg  
(negative level-sensitive latch clocked by test\_SYS\_CLK)  
Endpoint: I\_ORCA\_TOP/I\_BLENDER/S\_5  
(rising clock gating-check end-point clocked by SYS\_CLK)  
Path Group: \*\*clock\_gating\_default\*\*  
Path Type: max

| Point                                              | Incr    | Path     |
|----------------------------------------------------|---------|----------|
| clock test_SYS_CLK (fall edge)                     | 55.00   | 55.00    |
| clock network delay (propagated)                   | 3.82 *  | 58.82    |
| time given to startpoint                           | 85.05   | 143.87   |
| I_ORCA_TOP/I_BLENDER/latched_clk_en_reg/D (ldflb3) | 0.00    | 143.87 f |
| I_ORCA_TOP/I_BLENDER/latched_clk_en_reg/Q (ldflb3) | 0.29 *  | 144.16 f |
| I_ORCA_TOP/I_BLENDER/U1036/Y (or2a15)              | 0.31 *  | 144.46 f |
| I_ORCA_TOP/I_BLENDER/S_5/B (and2a15)               | 0.00 *  | 144.46 f |
| data arrival time                                  |         | 144.46   |
| clock SYS_CLK (rise edge)                          | 60.00   | 60.00    |
| clock network delay (propagated)                   | 2.25 *  | 62.25    |
| I_ORCA_TOP/I_BLENDER/S_5/A (and2a15)               |         | 62.25 r  |
| clock gating setup time                            | -0.20   | 62.05    |
| data required time                                 |         | 62.05    |
| data required time                                 | 62.05   |          |
| data arrival time                                  | -144.46 |          |
| slack (VIOLATED)                                   | -82.41  |          |

6-7

For more details on “time given to startpoint” with latches and time borrowing:

**report\_timing -trace\_latch\_borrow**

## Is My Timing Report Incorrect/Questionable?



6- 8

# 1. Incorrect Source Latency in Timing Report

- Why is the clock source latency zero even after applying clock source latency of 0.2?

```
set_clock_latency 0.2 [all_clocks] -source
```

| Point                 | Incr     | Path         |
|-----------------------|----------|--------------|
| clock clk (rise edge) | 0.000000 | 0.000000     |
| clock source latency  | 0.000000 | 0.000000     |
| CLK (in)              | 0.000000 | & 0.000000 r |
| CLKB1/Z (buff_test)   | 0.036749 | & 0.036749 r |
| CLKL1/Z (buff_test)   | 0.050674 | & 0.087424 r |
| CLKL2/Z (buff_test)   | 0.047189 | & 0.134613 r |
| CLKL3/Z (buff_test)   | 0.048370 | & 0.182983 r |
| CLKL4/Z (buff_test)   | 0.043574 | & 0.226556 r |
| FF1/CP (ffp_test)     | 0.000011 | & 0.225567 r |
| FF1/Q (ffp_test)      | 0.098763 | & 0.325330 r |
| U1/Z (buff_test)      | 0.053300 | & 0.378630 r |
| FF5/D (ffp_test)      | 0.000071 | & 0.378701 r |
| data arrival time     |          | 0.378701     |

- PTE-040 Warning:

Warning: Source Latency defined on pin/port 'CLK' will overwrite the clock source latency for clock 'clk'.  
(PTE-040)

## 1. Incorrect Source Latency in Timing Report : Debugging

- If clock source latency is defined for both a clock and its port (source pin), the source latency for the port takes precedence, being more specific.

- report\_clock -skew**

| Object | Min Condition Source Latency |          |          |          | Max Condition Source Latency |          |          |          |
|--------|------------------------------|----------|----------|----------|------------------------------|----------|----------|----------|
|        | Early_r                      | Early_f  | Late_r   | Late_f   | Early_r                      | Early_f  | Late_r   | Late_f   |
| clk    | 0.200000                     | 0.200000 | 0.200000 | 0.200000 | 0.200000                     | 0.200000 | 0.200000 | 0.200000 |
| CLK    | -                            | -        | -        | -        | -                            | -        | -        | -        |

Clock name: clk  
Clock port: CLK

- write\_sdc**

```
#####
Clock Related Information
#####
create_clock -name clk -period 1 -waveform { 0 0.5 } [get_ports {CLK}]
set_propagated_clock [get_clocks {clk}]
set_clock_latency -source -max 0.2 [get_clocks {clk}]
set_clock_latency -source -min 0.2 [get_clocks {clk}]
#####
Clock Latency Infomation
#####
set_clock_latency -source -max 0 [get_ports {CLK}]
set_clock_latency -source -min 0 [get_ports {CLK}]
#####
```

6-10

## 2. Zero Source Latency for Generated Clock

- Why is the clock source latency for the generated clock zero?

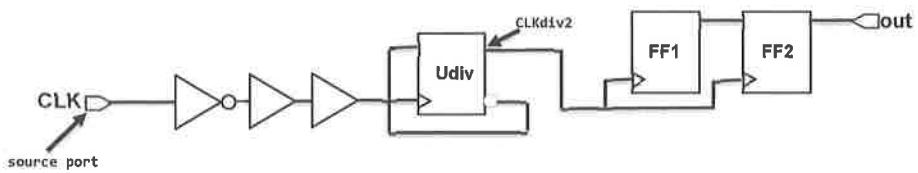
| Point                     | Incr    | Path      |
|---------------------------|---------|-----------|
| clock CLKdiv2 (rise edge) | 0.00000 | 0.00000   |
| clock source latency      | 0.00000 | 0.00000   |
| Udiv/Q (FD2)              | 0.00000 | 0.00000 r |
| FF1/Q (FD2)               | 0.00000 | 0.00000 r |
| FF1/Q (FD2)               | 1.42230 | 1.42230 f |
| FF2/D (FD2)               | 0.00000 | 1.42230 f |
| data arrival time         |         | 1.42230   |

- UITE-461 Error

```
Error: Generated clock 'CLKdiv2 with source pin Udiv/Q' 'rise_edge' is not satisfiable; zero
source latency will be used. (UITE-461)
Error: Generated clock 'CLKdiv2 with source pin Udiv/Q' 'fall_edge' is not satisfiable; zero
source latency will be used. (UITE-461)
```

## 2. The Generated Clock Specified

- The design is:



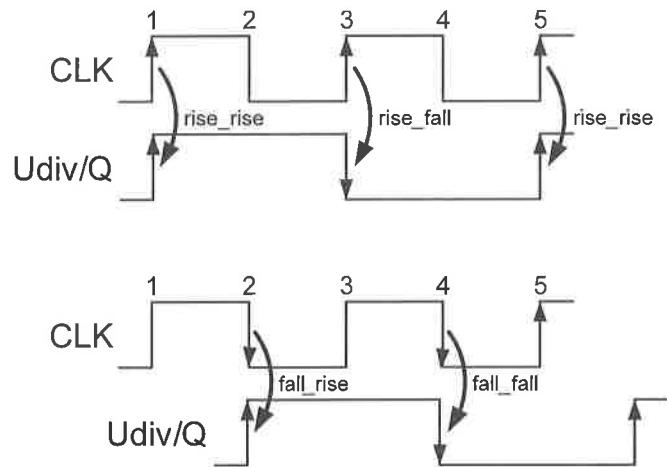
- The generated clock with `-divide_by 2` is specified at Udiv/Q as below:

```
create_generated_clock -name CLKdiv2 -divide_by 2 -source [get_ports CLK] [get_pins Udiv/Q]
```

6-12

## 2. Unsatisfiable Generated Clock!

- Gen Clock: Generated clock specification assumes a **rise\_rise** and **rise\_fall** edge relationship
  - This edge relationship cannot exist as there is an inversion in the clock path
- Design: The falling edges at the input port CLK result in rising or falling edges at the output of the dividing flip-flop
  - That means that only **fall\_rise** and **fall\_fall** relationship can exist
  - The design cannot provide the required propagation paths for this generated clock specification, zero source latency is used

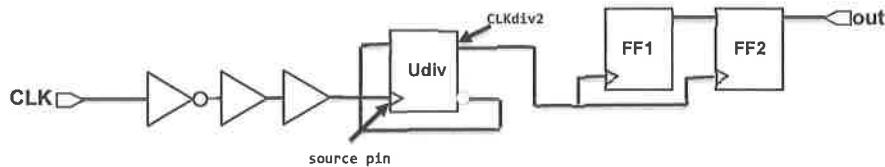


6-13

## 2. Zero Source Latency for Generated Clock : Solution

- Modify the source pin of the generated clock specification

```
create_generated_clock -name CLKdiv2 -divide_by 2 [-source [get_pins Udiv/CP]] [get_pins Udiv/Q]
```



| Point                               | Incr           | Path             |
|-------------------------------------|----------------|------------------|
| clock CLKdiv2 (rise edge)           | 2.00000        | 2.00000          |
| clock CLK (source latency)          | 0.00000        | 2.00000          |
| <b>CLK (in)</b>                     | <b>0.00000</b> | <b>2.00000 E</b> |
| U2/Z (IV)                           | 0.95720        | 2.95720 E        |
| U3/Z (B1I)                          | 0.57360        | 3.53080 E        |
| U4/Z (B1I)                          | 0.55590        | 4.08670 E        |
| <b>Udiv/Q (FD2) (gclock source)</b> | <b>1.48160</b> | <b>5.56830 E</b> |
| FF1/CP (FD2)                        | 0.00000        | 5.56830 E        |
| FF1/Q (FD2)                         | 1.42230        | 6.99060 E        |
| FF2/D (FD2)                         | 0.00000        | 6.99060 E        |
| data arrival time                   |                | 6.99060          |

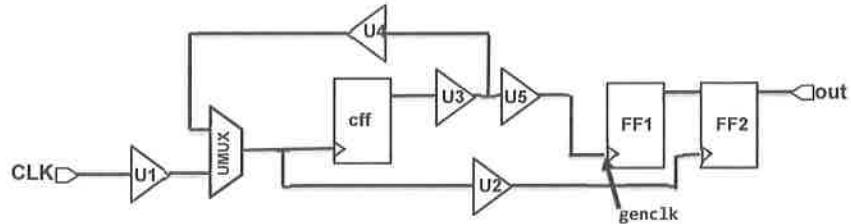
**020373: Why am I Getting UITE-461  
Messages and Zero Source Latency?**

<http://solvnet.synopsys.com/retrieve/020373.html>

6- 14

### 3. Sequential Loop in Source Latency Network

- Loops are detected in generated source latency network - the set of paths between the source pin of its master clock and the source pin of the generated clock
  - This scenario occurs when generated clock has not been correctly specified



#### ■ UITE-461 Warning

Warning: Loop(s) were detected in the generated source latency network of clock genclk. PrimeTime is prevented from computing the source latency of this clock with complete accuracy. (PTE-103)

### 3. Sequential Loop in Source Latency Network : Debugging

```
check_timing -override_defaults generated_clocks -verbose
```

```
pt_shell> check_timing -verbose -override_defaults generated_clocks
Warning: Loop(s) were detected in the generated source latency network of clock genclk. PrimeTime is
prevented from computing the source latency of this clock with complete accuracy. (PTE-103)
Example source network loop for clock genclk:
(# denotes clock source; * denotes sequential pin)

U3/Z
U4/A
U4/Z
UMUX/B
UMUX/Z
cff/CP *
cff/Q
U3/A
U3/Z
```

- Manually break the loop within the generated clock network.

```
set_sense -stop_propagation -clocks <master_clock_name> [get_pins U4/Z] ;# OR
set_disable_timing -from A -to Z [get_cells U4]
```

6- 16

## 4. Source Latency included with Input Delay

- The source latency is applied but the clock network delay remains zero?

```
set_clock_latency -source -early 1.1 CLK0
```

```
Startpoint: DIN_1 (input port clocked by CLK0)
Endpoint: F1 (rising edge-triggered flip-flop clocked by CLK0)
Path Group: CLK0
Path Type: min

Point Incr Path
-----+-----+-----+
clock CLK0 (rise edge) 0.00 0.00
clock network delay (propagated) 0.00 0.00
input external delay 0.40 0.40 f
DIN_1 (in) 0.00 0.40 f
AN1/Y (andd_test) 0.17 0.57 f
F1/DATA (ffp_test) 0.00 0.57 f
data arrival time 0.00 0.57 f
```

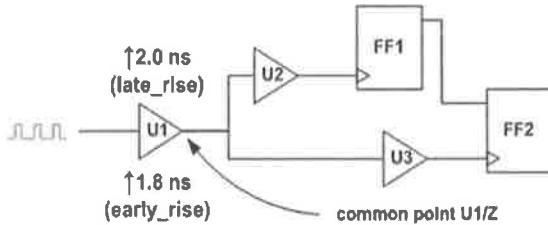
Clue: Startpoint is  
an input port with  
input delay

- write\_sdc

```
#####
External Delay Information
#####
set_input_delay 0.5 -clock [get_clocks {CLK0}] -max -source_latency_included [get_ports {DIN_1}]
set_input_delay 0.4 -clock [get_clocks {CLK0}] -min -source_latency_included [get_ports {DIN_1}]
```

## 5. Zero CRPR

### ■ Why is the CRPR zero?



```

Report : clock
Design : top

```

#### Attributes:

p - Propagated clock  
G - Generated clock  
I - Inactive clock

| Clock | Period | Waveform | Attrs | Sources |
|-------|--------|----------|-------|---------|
| clock | 10.00  | {0 5}    |       | {clock} |

```
Startpoint: ff1 (rising edge-triggered flip-flop clocked by
clock')
Endpoint: ff2 (rising edge-triggered flip-flop clocked by clock')
Path Group: clock
Path Type: max
```

| Point                         | Fanout | Incr      | Path      |
|-------------------------------|--------|-----------|-----------|
| clock clock' (rise edge)      |        | 5.0000    | 5.0000    |
| clock network delay (ideal)   |        | 0.0000    | 5.0000    |
| ff1/CP (FD1)                  |        | 0.0000    | 5.0000 f  |
| ff1/Q (FD1)                   |        | 1.5790 *  | 6.5790 f  |
| int1 (net)                    | 1      |           |           |
| inv1/A (B4I)                  |        | 0.0000 *  | 6.5790 f  |
| inv1/Z (B4I)                  |        | 0.3830 *  | 6.9620 r  |
| int2 (net)                    | 1      |           |           |
| ff2/D (FD1)                   |        | 0.0000 *  | 6.9620 r  |
| data arrival time             |        |           | 6.9620    |
| clock clock' (rise edge)      |        | 15.0000   | 15.0000   |
| clock network delay (ideal)   |        | 0.0000    | 15.0000   |
| clock reconvergence pessimism |        | 0.0000    | 15.0000   |
| ff2/CP (FD1)                  |        |           | 15.0000 r |
| library setup time            |        | -0.8000 * | 14.2000   |
| data required time            |        |           | 14.2000   |
| data required time            |        |           | 14.2000   |
| data arrival time             |        |           | -6.9620   |
| slack (MET)                   |        |           | 7.2380    |

6-18

## 5. Zero CRPR : Solution

### ■ Clock was ideal! – Upon propagating the clock

- `set_propagated_clock [get_clock clock]`

```

Report : clock
Design : top

Attributes:
 p - Propagated clock
 G - Generated clock
 I - Inactive clock
Clock Period Waveform Attrs Sources

clock 10.00 {0 5} p {clock}
```

| Startpoint: ff1 (rising edge-triggered flip-flop clocked by clock') |           |          |         |
|---------------------------------------------------------------------|-----------|----------|---------|
| Endpoint: ff2 (rising edge-triggered flip-flop clocked by clock')   |           |          |         |
| Path Group: clock                                                   |           |          |         |
| Path Type: max                                                      |           |          |         |
| Point                                                               | Fanout    | Incr     | Path    |
| clock clock' (rise edge)                                            | 5.0000    | 5.0000   |         |
| clock network delay (propagated)                                    | 3.0000    | 8.0000   |         |
| ff1/CP (FDI)                                                        | 0.0000    | 0.0000 r |         |
| ff1/Q (FDI)                                                         | 1.5790 *  | 9.5790 f |         |
| int1 (net)                                                          | 1         |          |         |
| inv1/A (B4I)                                                        | 0.0000 *  | 9.5790 f |         |
| inv1/Z (B4I)                                                        | 0.3830 *  | 9.9620 r |         |
| int2 (net)                                                          | 1         |          |         |
| ff2/D (FDI)                                                         | 0.0000 *  | 9.9620 r |         |
| data arrival time                                                   |           |          | 9.9620  |
| clock clock' (rise edge)                                            | 15.0000   | 15.0000  |         |
| clock network delay (propagated)                                    | 1.0000    | 16.0000  |         |
| clock reconvergence pessimism                                       | 2.0000    | 18.0000  |         |
| ff2/CP (FDI)                                                        | 18.0000 r |          |         |
| library setup time                                                  | -0.8000 * | 17.2000  |         |
| data required time                                                  |           |          | 17.2000 |
| data required time                                                  |           |          | 17.2000 |
| data arrival time                                                   |           |          | -9.9620 |
| slack (MET)                                                         |           |          | 7.2380  |

## 6. Zero CRPR with propagated clock

- Clock is propagated, CRPR is zero?
- report\_crpr shows non zero value!

```
pt_shell> report_crpr -from FF1/CP -to FF5/CP
Startpoint: FF1 (rising edge-triggered flip-flop clocked by clk)
Endpoint: FF5 (rising edge-triggered flip-flop clocked by clk)

Common Point: CLKB1/Z
Common Clock: clk
Launching edge at common point: RISING
Capturing edge at common point: RISING
CRPR threshold: 0.005

Arrival Times
-----|-----|-----|-----|-----|-----|
Rise Early Late CRP
-----|-----|-----|-----|-----|-----|
0.036234 0.036749 0.000515
Fall 0.036771 0.037048 0.000277
-----|-----|-----|-----|-----|-----|
Selection Details
-----|-----|-----|-----|-----|-----|
Edge Match: Match, using rise CRP
-----|-----|-----|-----|-----|-----|
clock reconvergence pessimism 0.000515
-----|-----|-----|-----|-----|-----|
Information: The CRP value reported by report_timing and report_clock_timing will
be -0.000000. (UITE-468)
```

| Startpoint: FF1 (rising edge-triggered flip-flop clocked by clk)<br>Endpoint: FF5 (rising edge-triggered flip-flop clocked by clk)<br>Last common pin: CLK<br>Path Group: clk<br>Path Type: max |           |              |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------|--------------|
| Point                                                                                                                                                                                           | Incr      | Path         |
| clock clk (rise edge)                                                                                                                                                                           | 0.000000  | 0.000000     |
| clock source latency                                                                                                                                                                            | 0.000000  | 0.000000     |
| CLK (in)                                                                                                                                                                                        | 0.000000  | & 0.000000 r |
| CLKB1/Z (buff_test)                                                                                                                                                                             | 0.036749  | & 0.036749 r |
| CLKL1/Z (buff_test)                                                                                                                                                                             | 0.050674  | & 0.087424 r |
| CLKL2/Z (buff_test)                                                                                                                                                                             | 0.047189  | & 0.134613 r |
| CLKL3/Z (buff_test)                                                                                                                                                                             | 0.048370  | & 0.182983 r |
| CLKL4/Z (buff_test)                                                                                                                                                                             | 0.043574  | & 0.226556 r |
| FF1/CP (ffp_test)                                                                                                                                                                               | 0.000011  | & 0.226567 r |
| FF1/Q (ffp_test)                                                                                                                                                                                | 0.098763  | & 0.325330 r |
| U1/Z (buff_test)                                                                                                                                                                                | 0.053300  | & 0.378630 r |
| FF5/D (ffp_test)                                                                                                                                                                                | 0.000071  | & 0.378701 r |
| data arrival time                                                                                                                                                                               |           | 0.378701     |
| clock clk (rise edge)                                                                                                                                                                           | 1.000000  | 1.000000     |
| clock source latency                                                                                                                                                                            | 0.000000  | 1.000000     |
| CLK (in)                                                                                                                                                                                        | 0.000000  | & 1.000000 r |
| CLKB1/Z (buff_test)                                                                                                                                                                             | 0.036234  | & 1.036234 r |
| CLKCS/Z (buff_test)                                                                                                                                                                             | 0.042594  | & 1.078826 r |
| FF5/CP (ffp_test)                                                                                                                                                                               | 0.000010  | & 1.078839 r |
| clock reconvergence pessimism                                                                                                                                                                   | 0.000000  | 1.078839     |
| library setup time                                                                                                                                                                              | -0.024407 | 1.054431     |
| data required time                                                                                                                                                                              |           | 1.054431     |
| ----- ----- ----- ----- ----- -----                                                                                                                                                             |           |              |
| data required time                                                                                                                                                                              |           | 1.054431     |
| data arrival time                                                                                                                                                                               |           | -0.378701    |
| ----- ----- ----- ----- ----- -----                                                                                                                                                             |           |              |
| slack (MET)                                                                                                                                                                                     |           | 0.675730     |

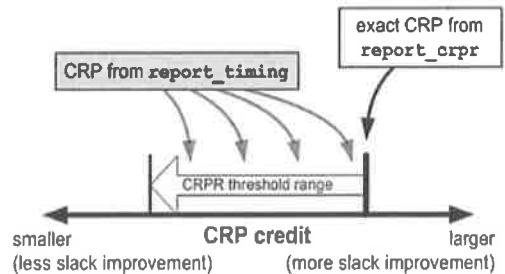
6-20

## 6. Zero CRPR with propagated clock : CRPR Threshold!

### ■ Why does report\_timing CRPR value differ from the report\_crpr?

- The value chosen during update\_timing will either be the exact value, or within the threshold distance on the conservative (smaller) side. This is applied in report\_timing output
- report\_crpr does a single register-to-register analysis, therefore it produces the exact CRPR value for the given path
- CRPR Threshold Variable → Default 5 ps

```
set timing_crpr_threshold_ps 5
```



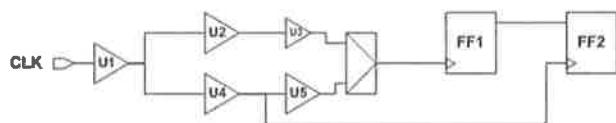
### 019502: Why Does report\_crpr Differ From the CRPR Value in report\_timing?

<http://solvnet.synopsys.com/retrieve/019502.html>

6-21

## 7. Incorrect CRPR Common Point

- Why is my common point incorrect?



- PrimeTime reports U1/Z as the common point. Shouldn't U4/Z be the common point?
- Let's call these common points as:
  - Visual Common point: U4/Z
  - Topological Common point: U1/Z

6-22

## 7. Incorrect CRPR Common Point in Reports

### ■ Incorrect CRPR common points from report\_crpr and report\_timing

```
report_crpr -from FF1/CP -to FF2/CP
 Startpoint: FF1 (rising edge-triggered flip-flop clocked by CLK)
 Endpoint: FF2 (rising edge-triggered flip-flop clocked by CLK)

 Common Point: U1/Z
 Common Clock: CLK
 Launching edge at common point: RISING
 Capturing edge at common point: RISING
 CRPR threshold: 0.005

 Arrival Times Early Late CRP

 Rise 0.05 0.08 0.03
 Fall 0.07 0.10 0.03

 Selection Details
 Edge Match: Match, using rise CRP

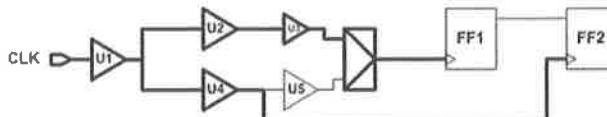
 clock reconvergence pessimism 0.03
```

| Startpoint: FF1 (rising edge-triggered flip-flop clocked by CLK) |             |               |
|------------------------------------------------------------------|-------------|---------------|
| Endpoint: FF2 (rising edge-triggered flip-flop clocked by CLK)   |             |               |
| Last common pin: U1/Z                                            |             |               |
| Point                                                            | Incr        | Path          |
| clock CLK (rise edge)                                            | 0.00        | 0.00          |
| clock source latency                                             | 0.00        | 0.00          |
| CLK (in)                                                         | 0.00        | 0.00 r        |
| <b>U1/Z (BUFPD1)</b>                                             | <b>0.08</b> | <b>0.08 r</b> |
| <b>U4/Z (BUFPD1)</b>                                             | <b>0.10</b> | <b>0.18 r</b> |
| U5/Z (BUFPD1)                                                    | 0.09        | 0.27 r        |
| UMUX/Z (OR2DI)                                                   | 0.11        | 0.38 r        |
| FF1/CP (DFD1)                                                    | 0.00        | 0.38 r        |
| FF1/Q (DFD1)                                                     | 0.28        | 0.66 f        |
| FF2/D (DFD1)                                                     | 0.00        | 0.66 f        |
| data arrival time                                                |             | 0.66          |
| clock CLK (rise edge)                                            | 1.00        | 1.00          |
| clock source latency                                             | 0.00        | 1.00          |
| CLK (in)                                                         | 0.00        | 1.00 r        |
| <b>U1/Z (BUFPD1)</b>                                             | <b>0.05</b> | <b>1.05 r</b> |
| <b>U4/Z (BUFPD1)</b>                                             | <b>0.07</b> | <b>1.12 r</b> |
| FF2/CP (DFD1)                                                    | 0.00        | 1.12 r        |
| clock reconvergence pessimism                                    | 0.03        | 1.15          |
| library setup time                                               | -0.01       | 1.14          |
| data required time                                               |             | 1.14          |
| data required time                                               |             | 1.14          |
| data arrival time                                                |             | -0.66         |
| slack (MET)                                                      |             | 0.48          |

6-23

## 7. Incorrect CRPR Common Point: Cause

- There exists another path that has a common point as U1/Z. This is called the topological common point since it's the common point which is valid for all possible paths.



- Refer to the 2 Tcl procedures on SolvNET to analyze this issue

**017503: Why Is My CRPR Common Point Incorrect?**

<http://solvnet.synopsys.com/retrieve/017503.html>

| Point                         | Incr        | Path          |
|-------------------------------|-------------|---------------|
| clock CLK (rise edge)         | 0.00        | 0.00          |
| clock source latency          | 0.00        | 0.00          |
| CLK (in)                      | 0.00        | 0.00 r        |
| <b>U1/Z (BUFPD1)</b>          | <b>0.08</b> | <b>0.08 r</b> |
| U2/Z (BUFPD1)                 | 0.09        | 0.17 r        |
| data arrival time             |             | 0.64          |
| clock CLK (rise edge)         | 1.00        | 1.00          |
| clock source latency          | 0.00        | 1.00          |
| CLK (in)                      | 0.00        | 1.00 r        |
| <b>U1/Z (BUFPD1)</b>          | <b>0.05</b> | <b>1.05 r</b> |
| U4/Z (BUFPD1)                 | 0.07        | 1.12 r        |
| FF2/CP (DFD1)                 | 0.00        | 1.12 r        |
| clock reconvergence pessimism | 0.03        | 1.15          |
| library setup time            | -0.01       | 1.14          |
| data required time            |             | 1.14          |
| data required time            |             | 1.14          |
| data arrival time             |             | -0.64         |
| slack (MET)                   |             | 0.50          |

6-24

## 8. Zero CRP in Timing Report

### ■ Why is the CRPR value zero in the timing report?

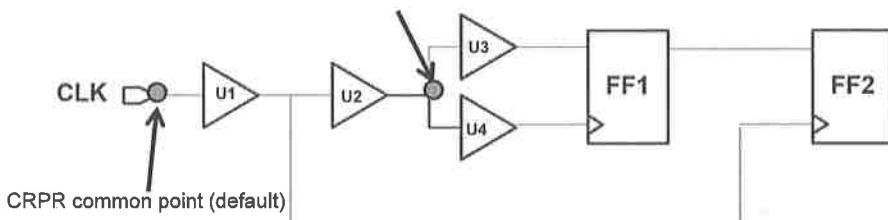
- Note that the startpoint is listed as the clock source

| Point                         | Incr      | Path       |
|-------------------------------|-----------|------------|
| clock clk (rise edge)         | 0.000000  | 0.000000   |
| clock source latency          | 0.000000  | 0.000000   |
| CLK (in)                      | 0.000000  | 0.000000 r |
| U1/Z (B1I)                    | 0.695400  | 0.695400 r |
| U2/Z (B1I)                    | 0.736640  | 1.412040 r |
| U3/Z (B1I)                    | 0.667080  | 2.079120 r |
| FPL/D (FD1)                   | 0.000000  | 2.079120 r |
| data arrival time             |           | 2.079120   |
|                               |           |            |
| clock clk (rise edge)         | 2.000000  | 2.000000   |
| clock source latency          | 0.000000  | 2.000000   |
| CLK (in)                      | 0.000000  | 2.000000 r |
| U1/Z (B1I)                    | 0.463600  | 2.463600 r |
| U2/Z (B1I)                    | 0.477760  | 2.941360 r |
| U4/Z (B1I)                    | 0.444720  | 3.386080 r |
| FPL/CP (FD1)                  | 0.000000  | 3.386080 r |
| clock reconvergence pessimism | 0.000000  | 3.386080   |
| library setup time            | -0.800000 | 2.586080   |
| data required time            |           | 3.586080   |
|                               |           |            |
| data required time            |           | 2.586080   |
| data arrival time             |           | -2.079120  |
|                               |           |            |
| slack (MET)                   |           | 0.506960   |

6-25

## 8. Zero CRP in Timing Report: Clock to Data CRP

- When a startpoint is listed as a clock source instead of a register, this is an indication that the clock is being directly captured as a data signal
- PrimeTime processes these clock-as-data paths by enabling the variable `timing_crpr_remove_clock_to_data_crp`
  - Default : False CRPR common point, when `timing_crpr_remove_clock_to_data_crp = true`



- Once above variable is set to true, a non zero CRP value is reported!

6-26

## 9. Unexpected Report with report\_timing -slack\_greater

- When I use the `report_timing` command with the `-slack_greater_than 0` option, PrimeTime reports that there are no paths.
- However, if I increase `nworst` value, PrimeTime reports the paths, Why?

```
pt_shell> report_timing -path end -slack_lesser_than infinity -slack_greater_than 0

Report : timing
 -path_type end
 -delay_type max
 -slack_lesser_than inf
 -slack_greater_than 0.000000
 -max_paths 1
 -sort_by slack
Design : test

No paths with slack less than inf and slack greater than 0.000000.

1
```

## 9. Unexpected Report with report\_timing -slack\_greater

- When nworst is increased, PrimeTime reports paths with slack > zero

| report_timing -path end -slack_lesser_than infinity -nworst 100 |            |               |          |           |
|-----------------------------------------------------------------|------------|---------------|----------|-----------|
| Endpoint                                                        | Path Delay | Path Required | CRP      | Slack     |
| FF10/D (ffp_test)                                               | 8.249595 r | 3.739289      | 0.145760 | -4.364546 |
| FF10/D (ffp_test)                                               | 7.892798 f | 3.626665      | 0.145760 | -4.120373 |
| FF6/D (ffp_test)                                                | 3.827941 r | 2.177457      | 0.152269 | -1.498215 |
| FF6/D (ffp_test)                                                | 3.492724 f | 2.065217      | 0.152269 | -1.275239 |
| FF7/D (ffp_test)                                                | 3.070989 r | 1.991879      | 0.186084 | -0.892946 |
| FF2/D (ffp_test)                                                | 2.846067 f | 1.882285      | 0.186084 | -0.777699 |
| FF12/D (ffp_test)                                               | 2.372742 f | 1.929467      | 0.145760 | -0.297515 |
| FF12/D (ffp_test)                                               | 2.327717 r | 2.044358      | 0.145760 | -0.137599 |
| FF8/D (ffp_test)                                                | 1.798550 f | 1.727310      | 0.152269 | 0.081029  |
| FF4/D (ffp_test)                                                | 1.814732 f | 1.763503      | 0.186084 | 0.134855  |
| FF8/D (ffp_test)                                                | 1.752961 r | 1.842148      | 0.152269 | 0.241456  |
| FF4/D (ffp_test)                                                | 1.767062 r | 1.876445      | 0.186084 | 0.295467  |
| FF7/D (ffp_test)                                                | 0.000000 f | 0.737619      | 0.000000 | 0.737618  |
| FF3/D (ffp_test)                                                | 0.000000 f | 0.776320      | 0.000000 | 0.776320  |
| FF7/D (ffp_test)                                                | 0.000000 r | 0.850523      | 0.000000 | 0.850523  |
| FF3/D (ffp_test)                                                | 0.000000 r | 0.888804      | 0.000000 | 0.888804  |
| FF11/D (ffp_test)                                               | 0.000000 f | 0.938916      | 0.000000 | 0.938916  |
| FF1/D (ffp_test)                                                | 0.000000 f | 0.987504      | 0.000000 | 0.987503  |
| FF11/D (ffp_test)                                               | 0.000000 r | 1.051831      | 0.000000 | 1.051831  |
| FF1/D (ffp_test)                                                | 0.000000 r | 1.079730      | 0.000000 | 1.079730  |
| FF11/D (ffp_test)                                               | 0.000000 r | 1.166207      | 0.000000 | 1.166207  |
| FF5/D (ffp_test)                                                | 0.000000 f | 1.258171      | 0.000000 | 1.258170  |
| FF9/D (ffp_test)                                                | 0.000000 f | 2.723541      | 0.000000 | 2.723541  |
| FF9/D (ffp_test)                                                | 0.000000 r | 2.815475      | 0.000000 | 2.815474  |

| report_timing -path end -slack_lesser_than infinity -nworst 100<br>-slack_greater_than 0 |            |               |          |          |
|------------------------------------------------------------------------------------------|------------|---------------|----------|----------|
| Endpoint                                                                                 | Path Delay | Path Required | CRP      | Slack    |
| FF8/D (ffp_test)                                                                         | 1.798550 f | 1.727310      | 0.152269 | 0.081029 |
| FF4/D (ffp_test)                                                                         | 1.814732 f | 1.763503      | 0.186084 | 0.134855 |
| FF8/D (ffp_test)                                                                         | 1.752961 r | 1.842148      | 0.152269 | 0.241456 |
| FF4/D (ffp_test)                                                                         | 1.767062 r | 1.876445      | 0.186084 | 0.295467 |
| FF7/D (ffp_test)                                                                         | 0.000000 f | 0.737619      | 0.000000 | 0.737618 |
| FF3/D (ffp_test)                                                                         | 0.000000 f | 0.776320      | 0.000000 | 0.776320 |
| FF7/D (ffp_test)                                                                         | 0.000000 r | 0.850523      | 0.000000 | 0.850523 |
| FF3/D (ffp_test)                                                                         | 0.000000 r | 0.888804      | 0.000000 | 0.888804 |
| FF11/D (ffp_test)                                                                        | 0.000000 f | 0.938916      | 0.000000 | 0.938916 |
| FF1/D (ffp_test)                                                                         | 0.000000 f | 0.987504      | 0.000000 | 0.987503 |
| FF11/D (ffp_test)                                                                        | 0.000000 r | 1.051831      | 0.000000 | 1.051831 |
| FF1/D (ffp_test)                                                                         | 0.000000 r | 1.079730      | 0.000000 | 1.079730 |
| FF5/D (ffp_test)                                                                         | 0.000000 f | 1.166207      | 0.000000 | 1.166207 |
| FF5/D (ffp_test)                                                                         | 0.000000 r | 1.258171      | 0.000000 | 1.258170 |
| FF9/D (ffp_test)                                                                         | 0.000000 f | 2.723541      | 0.000000 | 2.723541 |
| FF9/D (ffp_test)                                                                         | 0.000000 r | 2.815475      | 0.000000 | 2.815474 |

6-28

## 9. Unexpected Report with report\_timing -slack\_greater

### ■ The -slack\_greater\_than option is a post-processing filter.

- Step1: PrimeTime gathers the specified paths without the -slack\_greater\_than option
- Step2: Applies the filtering for slack greater than 0

```
report_timing -path end -slack_lesser_than infinity
Endpoint Path Delay Path Required CRP Slack

```

| Endpoint          | Path Delay | Path Required | CRP      | Slack     |
|-------------------|------------|---------------|----------|-----------|
| FF10/D (ffp_test) | 8.249595 r | 3.739289      | 0.145760 | -4.364546 |



```
report_timing -path end -slack_lesser_than infinity -slack_greater_than 0
No paths with slack less than inf and slack greater than 0.000000.
```

## 10. PBA WNS report\_qor/report\_glob & report\_timing Differ

- The path mode PBA WNS is different in timing report compared to report\_global\_timing and report\_qor. Why?

```
report_timing -pba path -path end -sig 6
Endpoint Path Delay Path Required CRP Slack
EP4/D (FD2) 37.323143 r 37.119068 0.000000 -0.204075

report_global_timing -pba path -sig 6
Setup violations
Total reg->reg in->reg reg->out in->out
WNS -0.215172 -0.215172 0.000000 0.000000 0.000000
TNS -0.722958 -0.722958 0.000000 0.000000 0.000000
NUM 4 4 0 0 0

report_qor -pba path -sig 6
Timing Path Group 'clk' (max_delay/setup)
Levels of Logic: 15
Critical Path Length: 3.108688
Critical Path Slack: -0.215172
Total Negative Slack: -0.722958
No. of Violating Paths: 4
```

6-30

## 10. PBA WNS report\_qor/report\_glob & report\_timing Differ

- `report_timing -pba_mode path` with `-max_paths/nworst 1` considers the worst GBA endpoint and recalculate a path to this endpoint

| Endpoint    | Path Delay  | Path Required | CRP      | Slack     |
|-------------|-------------|---------------|----------|-----------|
| EP4/D (FD2) | 37.365784 r | 37.105492     | 0.000000 | -0.260292 |



| Endpoint    | Path Delay  | Path Required | CRP      | Slack     |
|-------------|-------------|---------------|----------|-----------|
| EP4/D (FD2) | 37.323143 r | 37.119068     | 0.000000 | -0.204075 |

| GBA            | PBA path       |  |  |
|----------------|----------------|--|--|
| EP4 -0.260292  | EP1 -0.215172  |  |  |
| EP8 -0.257202  | EP2 -0.209812  |  |  |
| EP7 -0.256588  | EP3 -0.204216  |  |  |
| EP3 -0.247246  | EP4 -0.204075  |  |  |
| EP1 -0.228786  | EP5 -0.17746   |  |  |
| EP2 -0.214481  | EP6 -0.165691  |  |  |
| EP5 -0.200218  | EP7 -0.161419  |  |  |
| EP10 -0.200169 | EP8 -0.152912  |  |  |
| EP6 -0.18856   | EP9 -0.133823  |  |  |
| EP9 -0.165573  | EP10 -0.126579 |  |  |

- In this case, EP4/D is chosen by `report_timing -pba_mode path` as it is the worst GBA endpoint
- When the path gathering options (`-max_paths/nworst`) are increased, we get the EP1/D as the worst path in PBA path mode

6-31

## 10. PBA WNS report\_qor/report\_glob & report\_timing Differ

- The report\_global\_timing and report\_qor commands consider all endpoints so they find the real worst pba endpoint EP1/D

```
report_global_timing -pba path -sig 6
```

Setup violations

|     | Total     | reg->reg  | in->reg  | reg->out | in->out  |
|-----|-----------|-----------|----------|----------|----------|
| WNS | -0.215172 | -0.215172 | 0.000000 | 0.000000 | 0.000000 |
| TNS | -0.722958 | -0.722958 | 0.000000 | 0.000000 | 0.000000 |
| NUM | 4         | 4         | 0        | 0        | 0        |

```
report_qor -pba path -sig 6
```

Timing Path Group 'clk' (max\_delay/setup)

| Levels of Logic:        | 15        |
|-------------------------|-----------|
| Critical Path Length:   | 3.108688  |
| Critical Path Slack:    | -0.215172 |
| Total Negative Slack:   | -0.722958 |
| No. of Violating Paths: | 4         |

| GBA  |           | PBA path |                  |
|------|-----------|----------|------------------|
| EP4  | -0.260292 | EP1      | <b>-0.215172</b> |
| EP8  | -0.257202 | EP2      | -0.209812        |
| EP7  | -0.256588 | EP3      | -0.204216        |
| EP3  | -0.247246 | EP4      | -0.204075        |
| EP1  | -0.228786 | EP5      | -0.17746         |
| EP2  | -0.214481 | EP6      | -0.165691        |
| EP5  | -0.200218 | EP7      | -0.161419        |
| EP10 | -0.200169 | EP8      | -0.152912        |
| EP6  | -0.18856  | EP9      | -0.133823        |
| EP9  | -0.165573 | EP10     | -0.126579        |

6-32

## 11. Negative Cell Delay in Timing Report

- Why am I seeing negative cell delays? CCS Timing libraries are being used

| Rise                              | Fall      | CCS               |
|-----------------------------------|-----------|-------------------|
| Input transition time = 0.380636  | 0.363515  | (in library unit) |
| Effective capacitance = 0.004279  | 0.004271  | (in pF)           |
| Effective capacitance = 0.004279  | 0.004271  | (in library unit) |
| Output transition time = 0.055042 | 0.044331  | (in library unit) |
| Cell delay = 0.052895             | -0.005472 | (in library unit) |

- The issue is not seen when using NLDM driver model

| Rise                              | Fall     | NLDM              |
|-----------------------------------|----------|-------------------|
| Input transition time = 0.380636  | 0.363515 | (in library unit) |
| Effective capacitance = 0.004869  | 0.004899 | (in pF)           |
| Effective capacitance = 0.004869  | 0.004899 | (in library unit) |
| Drive resistance = 0.001000       | 0.001000 | (in Kohm)         |
| Output transition time = 0.058206 | 0.048634 | (in library unit) |
| Cell delay = 0.053883             | 0.008379 | (in library unit) |

## 11. Negative Cell Delay in Timing Report : The issue

### ■ CCS delay calculation derived from table results in negative delay

- Very large input transition and very small load
- Output transition is much sharper than input transition

| Rise                              | Fall                        | CCS |
|-----------------------------------|-----------------------------|-----|
| Input transition time = 0.380636  | 0.363515 (in library unit)  |     |
| Effective capacitance = 0.004279  | 0.004271 (in pF)            |     |
| Effective capacitance = 0.004279  | 0.004271 (in library unit)  |     |
| Output transition time = 0.055042 | 0.044331 (in library unit)  |     |
| Cell delay = 0.052895             | -0.005472 (in library unit) |     |

### ■ Perform library consistency checks in Library Compiler

- The Library Compiler check\_library tests revealed several mismatches and all with large falling transitions and small load
- The issues will need to be addressed in characterization

6-34

## 12. No Paths in Timing Report

- Why am I getting “No Paths” in the timing report? I can see a path in the schematic

```
set timing_report_unconstrained_paths true
report_timing -from FF1/CP -to FF2/D -path full_clock_expanded -exceptions all

Report : timing
 -path_type full_clock_expanded
 -delay_type max
 -max_paths 1
 -sort_by slack
Design : test

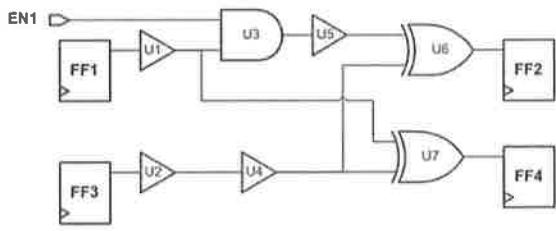
No Paths.

1
```

- The unconstrained path reporting is enabled
- The timing path is checked for any exception setting

## 12. No Paths in Timing Report : Debugging the cause

- There is definitely a topological combinational path between the pins FF1/CP and FF2/D
- Using the `-trace_arcs all` option of the `all_fanin` and `all_fanout` commands:

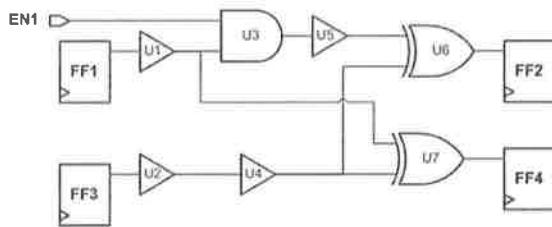


```
pt_shell> set fanout [all_fanout -flat -from FF1/CP -trace_arcs all]
{"FF1/CP", "FF1/Q", "FF1/QN", "U1/A", "U1/Z", "U3/B", "U7/A2",
 "U3/Z", "U7/Z", "U5/A", "FF4/D", "U5/Z", "U6/A2", "U6/Z", "FF2/D"}
pt_shell> set fanin [all_fanin -flat -to FF2/D -trace_arcs all]
{"FF2/D", "U6/Z", "U6/A1", "U6/A2", "U4/Z", "U5/Z", "U4/A", "U5/A",
 "U2/Z", "U3/Z", "U2/A", "U3/A", "U3/B", "FF3/Q", "EN1", "U1/Z",
 "FF3/CP", "U1/A", "FF1/Q", "FF1/CP"}
pt_shell> set intersection [intersect $fanin $fanout]
{"FF2/D", "U6/Z", "U6/A2", "U5/Z", "U5/A", "U3/Z", "U3/B", "U1/Z",
 "U1/A", "FF1/Q", "FF1/CP"}
```

- Next step is to find out where is the path blocked and what is causing it

## 12. No Paths in Timing Report : `find_blocking_arcs`

- The `find_blocking_arcs` (Solvnet article 030124) shows that the path is blocked starting with the cell U3
  - EN1 port is set to zero!



```
find_blocking_arcs -from FF1/CP -to FF2/D -include_net_arcs
Fanin from 'FF1/CP' blocked at timing arcs:
 U3/Z
 ->U5/A

Unreachable timing arcs:
 U5/Z
 ->U6/A2

 U5/A
 ->U5/Z

Fanout to 'FF2/D' blocked at timing arcs:
 U6/A2
 ->U6/Z
```

The proc reported first arc in the path that is not reachable

```
pt_shell> get_pins -of [get_cells {U3 U5}] -filter {defined(case_value)}
{"U3/A", "U3/Y", "U5/I", "U5/Y"}
pt_shell> report_case_propagation [get_pins U3/A]
...
EN1 (in port) user-defined case=0
```

6-37

```
pt_shell> find_blocking_arcs -help
```

Usage:

|                                  |                                       |
|----------------------------------|---------------------------------------|
| <code>find_blocking_arcs</code>  | # find blocking arcs between two pins |
| <code>-from_pin pin_name</code>  | (from pin of interest)                |
| <code>-to_pin pin_name</code>    | (to pin of interest)                  |
| <code>[-include_net_arcs]</code> | (include net arcs in report)          |

## Review of Unit Objectives



**Having Completed this unit, you should be able to:**

- Identify key pieces of a timing report showing constraint issues
- Debug Timing Reports when the findings are incorrect/questionable

6- 38

# Agenda

DAY  
2

4 Constraining Multiple Clocks



5 Additional Checks and Constraints



6 Best Practices Debugging Reports

7 Signoff: Path Based Analysis (PBA)



7-1

## Unit Objectives



After completing this unit, you should be able to:

- Achieve timing accuracy by reducing pessimism using Path Based Analysis (PBA)
- Differentiate path vs. exhaustive PBA modes
- Identify situations in which PBA is not suitable / suitable / a must
- Discuss the purpose behind using Derate only exhaustive PBA and how to use it

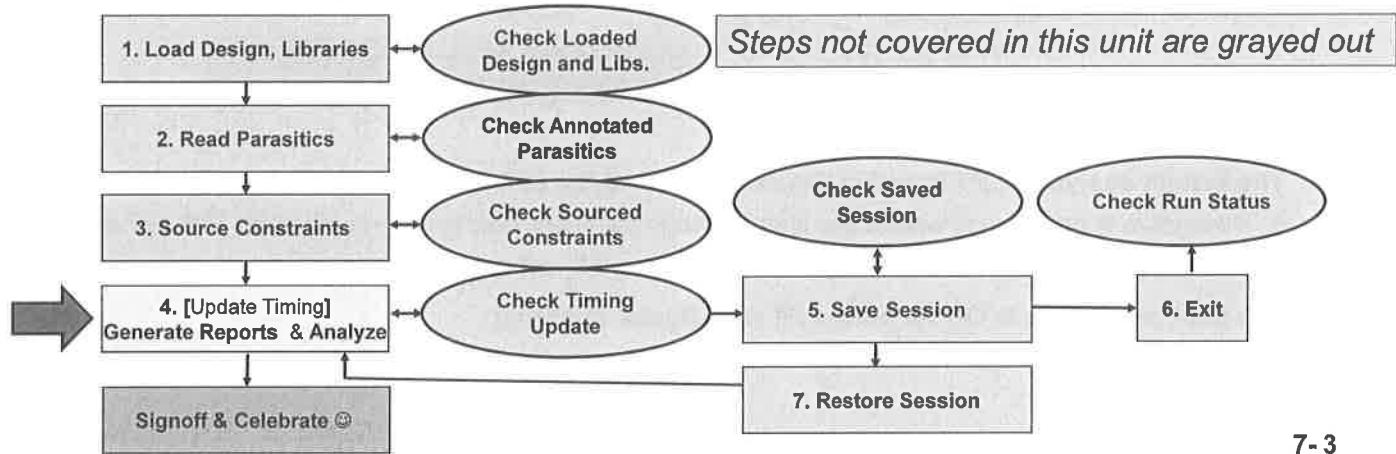
7-2

# Timing Analysis Flow in PrimeTime – Generating Reports

- STA flow is divided into several steps

- Steps 1-3 read data and
- Analysis begins at Step-4

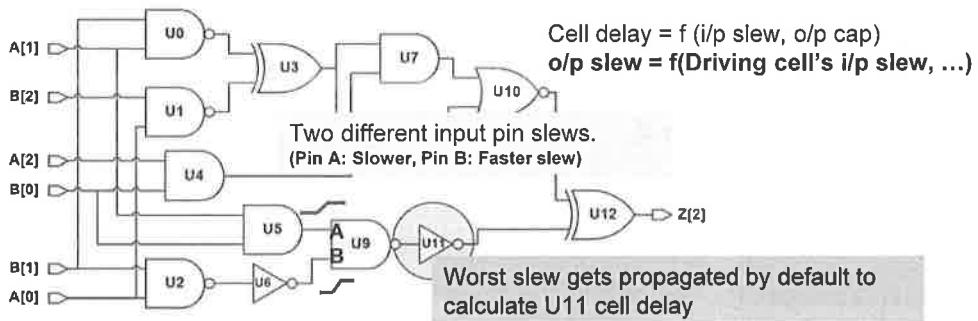
- The STA flow is repeated until signoff is achieved



7-3

STA: Static Timing Analysis

# Fast Analysis != Final Analysis

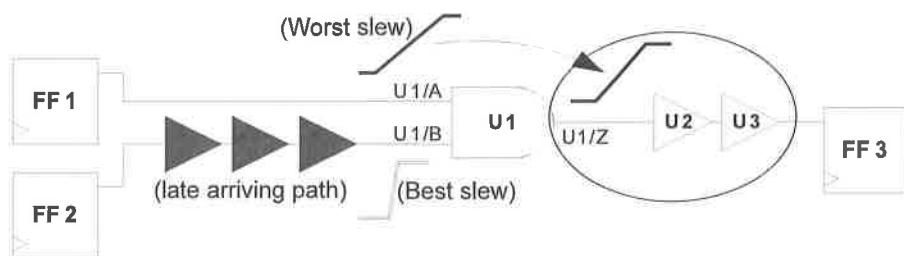


- STA is fast [doesn't require simulation]
- The default analysis in PT is called Graph based analysis (GBA)
  - Analysis is faster -- Uses worst case slew (through pin U9/A) propagation to calculate U11 cell delay
  - Analyzed path through pin U9/B is pessimistic due to U11 cell delay
- The GBA pessimism is OK for earlier PT runs [faster analysis]
  - When approaching signoff, this pessimism has to be removed by using path based analysis (PBA)

7-4

## Graph Based Analysis (GBA)

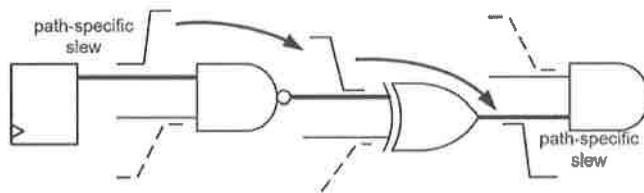
- The default analysis in PT is known as GBA which uses the worst input slew of a cell to calculate that cell's output transition
  - The calculated output slew will be used to calculate the delays of downstream cells
  - This can lead to pessimism in the analysis of certain paths, ex: when FF2-FF3 path is analyzed using the U1/A pin slew.



7-5

## Path-Based Analysis (PBA)

- **PBA performs path-specific slew propagation**
  - computes path-specific cell and net delays
  - computes path-specific slew degradation across nets
  - then, re-computes resulting path slack
- **It's computationally prohibitive to apply PBA technique on paths of the entire design**
  - PBA can be applied on user-specified timing path(s), ex: on a handful of violating timing paths

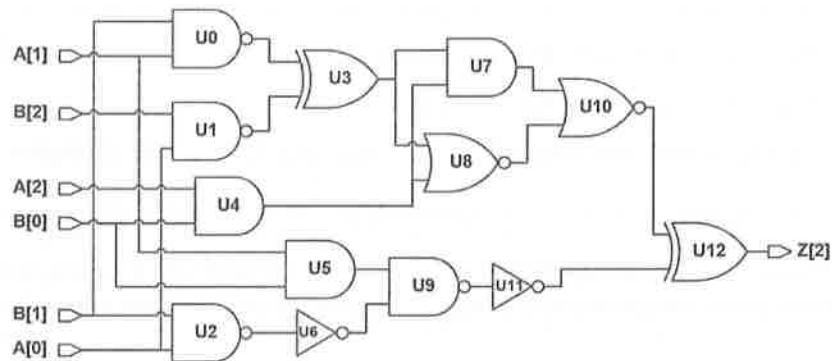


7-6

In an AOCV (advanced on-chip variation) flow, path-based analysis recalculates both slew AND path depth (see the PrimeTime and PrimeTime SI User Guide in Solvnet).

## Why Computationally Prohibitive?

- In the following logic cone (with only 5 levels of logic), there are 320 unique paths to Z[2]
  - Non-unate, conditional arcs, reconvergent paths are considered
  - There are 320 output slews and arc delay values (to compute and store) for the XOR gate U12 alone!



7-7

## Why Use Path-Based Analysis (PBA)?

- **Graph-based analysis (GBA) has some inherent pessimism**

- Conservative slew merging approach
- Conservative AOCV depth calculation
- SI window alignment analysis

- **Path-based analysis reduces the pessimism seen in GBA mode**

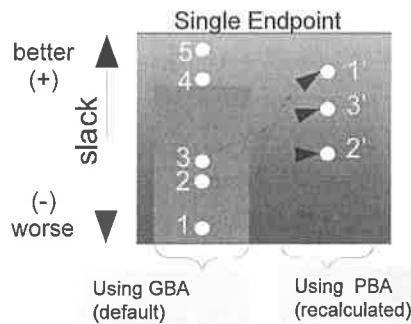
- Checks whether an endpoint that violates timing in GBA can meet timing
- Can waive timing violations seen in GBA
- Useful especially when signing off many high performance designs

*Note:*

- An endpoint can be marked as timing clean, only if all paths to it have positive slack
- An endpoint violates timing if any of the paths to it have negative slack

7-8

# Using Path-Based Analysis



```
recalculate timing path using PBA:
report_timing -pba_mode path ...
get_timing_paths -pba_mode path ...

OR
report_timing -pba_mode exhaustive ...
get_timing_paths -pba_mode exhaustive ...
```

- PBA recalculates timing paths starting with the worst (#1) of GBA analyzed paths
- Each path may see different amounts of slack improvements
- PBA slack can never be worse than GBA's
- There are two PBA modes:
  - path
  - exhaustive

7-9

As of the latest version of PrimeTime, several commands support the `pba_mode` option (using `apropos pba_mode`):

```
fix_eco_timing
get_timing_paths
report_constraint
report_global_timing
report_qor
report_timing
```

## PBA Modes: Path vs Exhaustive

### ■ Path:

- `nworst` and `max_paths` determine, based on GBA based slacks, which paths are going to be reported
- Reports the recalculated (PBA) delays/slacks of those exact same paths

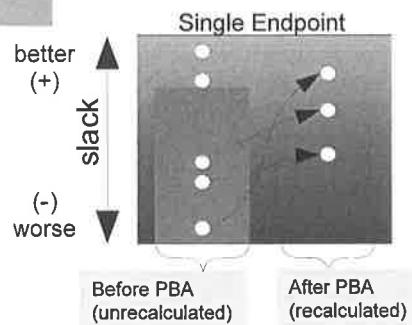
Fast, but does not guarantee that the reported paths are indeed paths having the worst slacks.

Useful for getting an idea of how much improvement PBA might bring.

Slower – wait until closer to sign-off.

### ■ Exhaustive:

- `nworst` and `max_paths` only determines how many paths will be reported, not which ones
- Timing analysis recalculates as many paths as needed (up to 25K per each endpoint), to ensure that the reported paths are indeed the paths with the worst recalculated slacks



7-10

## Test For Understanding



- 1. If there are 2 timing violations to a given endpoint in the design using GBA, how many paths are recalculated by PBA using:**

report\_timing -to \$endpoint -pba\_mode path

- a. 2
- b. 1
- c. Can't say

- 2. The recalculated path returned is the worst violating path ?**

- a. Yes
- b. No
- c. Can't say

7-11

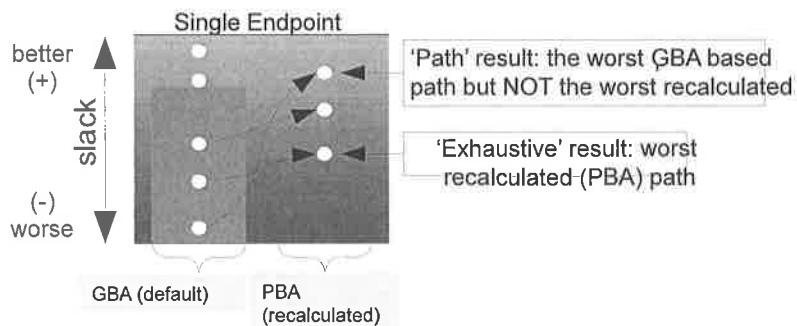
Answer : 1 b. 1 (same as nworst default = 1)

2. c. Can't say

## exhaustive PBA mode: when does it stop

### ■ During “exhaustive” mode analysis:

- After recalculating the worst before-PBA path, PrimeTime
  1. Looks for the next unrecalculated path with the worst slack
  2. Compares the unrecalculated path slack to all of the worst *nworst* recalculated paths.
  3. If the unrecalculated slack is worse, it too is recalculated, and so on, up to the default of 25000 paths per endpoint.



7-12

Remember that when a path is recalculated, the slack can only remain the same or improve. PBA cannot make slack worse.

## Exhaustive PBA: Performance Recommendation

- **Exhaustive PBA runtime depends upon**
  - Amount of slack improvement versus GBA
  - No. of the paths present near critical slack value
- **Use exhaustive PBA toward final timing signoff stages**
- **Use default -nworst with exhaustive PBA**
  - If an endpoint fails after exhaustive PBA, no need to report multiple paths for the same endpoint
- **Use default (or lower) setting for -slack\_lesser\_than**
  - PBA is used to waive failing GBA endpoints
  - Generating data for passing endpoints increases the runtime

7-13

## Exhaustive PBA: Usage Recommendation

### ■ Using `pba_exhaustive_endpoint_path_limit` variable

- This variable restricts the exhaustive path search by limiting the number of paths recalculated at any endpoint. Issues UITE-480 once the limit is reached
- Default is 25000
- Set it to a smaller value when exhaustive PBA runtime is high
- Recommended setting is 1000 (for limiting runtime)

**Warning:** The exhaustive path-based recalculation limit of 1000 has been exceeded at endpoint 'ff2/DATA' and group 'all'. The worst PBA slack found at this endpoint was -0.112729. The worst GBA slack of the remaining paths not recalculated is -0.123319. It is known that no PBA paths exist with worse slack than this worst GBA slack at this endpoint for this group. (UIITE-480)

Increase the limit only for analyzing the endpoints that show positive PBA slack and negative GBA slack in *UIITE-480* message

- No UITE-480 warnings indicate the result is exhaustive

7-14

For more information about UITE-480, refer to SolvNet 019597 : "How Does the Exhaustive Path-Based Search Work?"

```
set pba_exhaustive_endpoint_path_limit 25000 (default)
```

## Recalculated Paths in Timing Reports

pba\_mode path recalculates then reports nworst/max\_paths paths

```
pt_shell> report_timing -path_end -to I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]/D \
-nosplit nworst 4 -max_paths 4 -pba path
```

| Endpoint                                       | Path     | Delay  | Path Required | CRP    | Slack |
|------------------------------------------------|----------|--------|---------------|--------|-------|
| I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]/D (sdnrb1) | 12.002 f | 10.547 | 0.470         | -0.985 |       |
| I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]/D (sdnrb1) | 11.993 f | 10.547 | 0.470         | -0.976 |       |
| I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]/D (sdnrb1) | 11.987 f | 10.547 | 0.470         | -0.970 |       |
| I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]/D (sdnrb1) | 11.977 f | 10.547 | 0.470         | -0.960 |       |

pba\_mode exhaustive recalculates as many paths as needed to report nworst/max\_paths paths

```
pt_shell> report_timing -path_end -to I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]/D \
-nosplit nworst 4 -max_paths 4 -pba exhaustive
```

Information: Recalculated 6 paths and returned 4 paths. The maximum number of paths recalculated at an endpoint was 6.

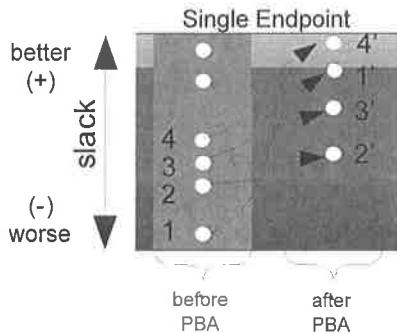
| Endpoint                                       | Path     | Delay  | Path Required | CRP    | Slack |
|------------------------------------------------|----------|--------|---------------|--------|-------|
| I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]/D (sdnrb1) | 12.002 f | 10.547 | 0.470         | -0.985 |       |
| I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]/D (sdnrb1) | 11.993 f | 10.547 | 0.470         | -0.976 |       |
| I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]/D (sdnrb1) | 11.987 f | 10.547 | 0.470         | -0.970 |       |
| I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]/D (sdnrb1) | 11.981 f | 10.547 | 0.470         | -0.964 |       |

## When to use: Path vs Exhaustive

- **path: for exploring – to get a quick idea of what can be recovered**
  - Guide it with -through, -groups, -to, etc.
- **exhaustive: for signoff – recover that last bit of pessimism**
  - Run when number of violating endpoints is ‘reasonable’ (<50K).
  - To determine how many failing endpoints exist to be recalculated, use
    - ◆ report\_global\_timing (or)
    - ◆ report\_qor -summary

7-16

## Test For Understanding



1. report\_timing -nworst 2 -pba\_mode path
  - a) Which paths are recalculated?
  - b) Are those the worst paths after recalculation?
2. report\_timing -nworst 2 -pba\_mode exhaustive
  - a) How many / Which paths are recalculated?
  - b) How many / Which paths are reported?

7-17

1a. Paths 1 and 2. -pba\_mode path recalculates nworst ‘before PBA’ paths, starting with the worst.

1b. No – paths 2 and 3 are the worst (2’ and 3’) after recalculation

2a. 4 – paths 1, 2, 3, and 4. -pba\_mode exhaustive recalculates up to 25000 paths per endpoint, until it has identified nworst recalculated paths to consider per endpoint. In this case, after recalculating the worst three ‘before PBA’ paths, it has identified only one ‘after PBA’ path that is worse than the fourth worst ‘pre-PBA’ path. So, it must recalculate the fourth path. It now stops recalculating because no ‘before-PBA’ path is worse than the worst two ‘after PBA’ paths: 2’ and 3’.

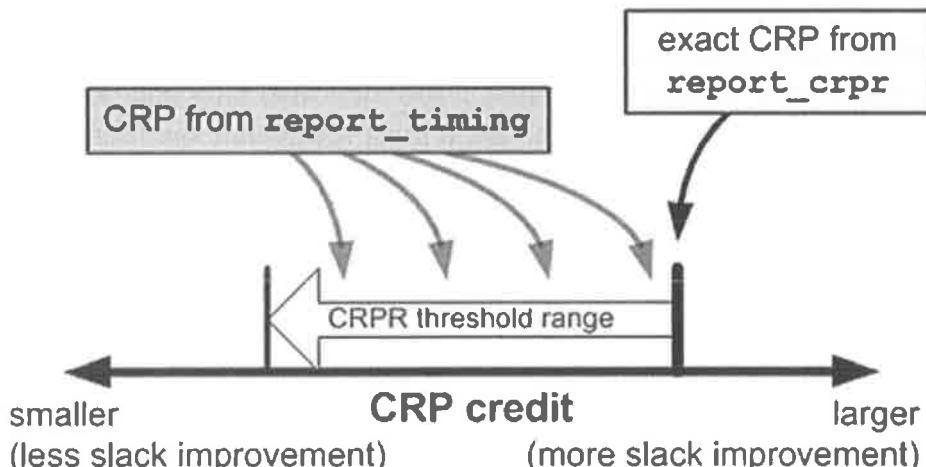
2b. 2 -- by default, max\_paths takes the same value as nworst, so PrimeTime reports 2 paths per design: in this case, 2’ and 3’.

## Things to Remember When Using PBA:

- Parasitic back annotation files are required for PBA
- With CRPR enabled (the default)
  - PrimeTime re-computes exact CRP values for recalculated paths
  - Effective `timing_crpr_threshold_ps` variable is set to 0 automatically [default is 5] for exact CRP calculation (See Notes)
    - ◆ i.e., CRP value in `report_timing` will match that of `report_crpr`
- PBA is also supported by the following reports
  - `report_constraint`
  - `report_global_timing`
  - `report_qor`
  - `fix_eco_timing`

7-18

`Timing_crpr_threshold_ps` is amount of pessimism that clock reconvergence pessimism removal (CRPR) is allowed to leave in the report..



## report\_delay\_calculation Does Not Match report\_timing -pba



```

Report : timing
 collection of 1 path(s)
 -input_pins
 -transition_time
 -pba_mode path

I_ORCA_TOP/I_BLENDER/IU4272/CI (ad01d0) 0.3165 0.0443 & 2.9221 f
I_ORCA_TOP/I_BLENDER/IU4272/S (ad01d0) 0.2508 0.3846 & 3.3068 f
I_ORCA_TOP/I_BLENDER/U3610/CI (ad01d0) 0.2885 0.0444 & 3.3512 f
I_ORCA_TOP/I_BLENDER/U3610/S (ad01d0) 0.2324 0.3747 & 3.7259 f
```

The slews at the input pin are different!

Cell delays are different!

```
report_delay_calculation -from I_ORCA_TOP/I_BLENDER/U3610/CI -to
I_ORCA_TOP/I_BLENDER/U3610/S
Input transition time = 0.308650 0.312457 (in library unit)
Effective capacitance = 0.007689 0.007625 (in pF)
Effective capacitance = 0.007689 0.007625 (in library unit)
Drive resistance = 6.274639 6.111027 (in Kohm)
Output transition time = 0.238395 0.232450 (in library unit)
Cell delay = 0.376172 0.379031 (in library unit)
```

## Making report\_delay\_calculation Match PBA Delay

- report\_delay\_calculation is not supported in PBA mode!
- Solution: Annotate the correct transition amount from the timing report



```

Report : timing
....
-pba_mode path

I_ORCA_TOP/I_BLENDER/IU4272/CI (ad01d0) 0.3165 0.0443 & 2.9221 f
I_ORCA_TOP/I_BLENDER/IU4272/S (ad01d0) 0.2508 0.3846 & 3.3068 f
I_ORCA_TOP/I_BLENDER/U3610/CI (ad01d0) 0.2885 0.0444 & 3.3512 f
I_ORCA_TOP/I_BLENDER/U3610/S (ad01d0) 0.2324 0.3747 & 3.7259 f

set_annotated_transition 0.2885 -fall
I_ORCA_TOP/I_BLENDER/U3610/CI Matching Cell delays!

report_delay_calculation -from I_ORCA_TOP/I_BLENDER/U3610/CI -to
I_ORCA_TOP/I_BLENDER/U3610/S
Rise Fall

Input transition time = 0.308650 0.288500 (in library unit)
Effective capacitance = 0.007689 0.007625 (in pF)
Effective capacitance = 0.007689 0.007625 (in library unit)
Drive resistance = 6.274639 6.111027 (in Kohm)
Output transition time = 0.238385 0.232450 (in library unit)
Cell delay = 0.376172 0.374719 (in library unit)
```

7-20

To see how specific timing arc delays are calculated (whether net or cell delays), use the command report\_delay\_calculation.

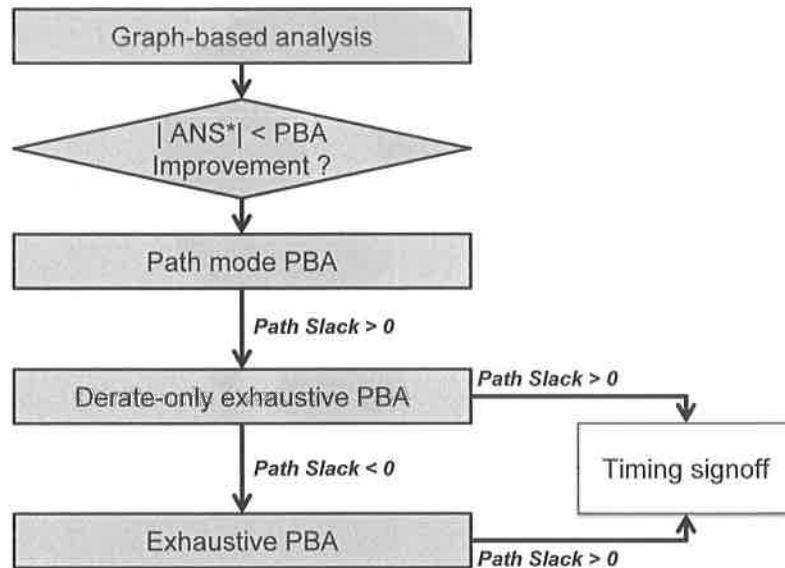
```
pt_shell> set_annotated_delay -help
-cell (Backannotate cell delays)
-net (Backannotate net delays)
[-rise] (Backannotate rise delay)
[-fall] (Backannotate fall delay)
[-min] (Backannotate min delay)
[-max] (Backannotate max delay)
[-load_delay load_delay_type]
 (Cell delay includes load delay:
 Values: net, cell)
[-from from_pins] (From pins of the timing delay)
[-to to_pins] (To pins of the timing delay)
[-cond sdf_expression] (Condition for backannotation)
[-increment] (Increment the current delay)
[-delta_only] (Annotate a delta delay over computed delay)
[-worst] (Keep worst value - NOT SUPPORTED)
delay_value (Backannotated delay value)
```

## Is My Design Ready for PBA?



7-21

## Path-Based Analysis : A Recommended Flow



\*ANS: Average Negative Slack = TNS / No. of Violating Endpoints

7-22

## Case-1: Not Suitable for Path Based Analysis

- Endpoints have unrealistic delay or slack
- Example-1: Over-constrained path: `path_slack > path_length`

```
Timing Path Group 'core'

Critical Path Length: 1.3110
Critical Path Slack: -2.0130
```

- Possibly missing multicycle path or false path exceptions?
- Example-2: Short paths (Paths involving I/O or Macros)

```
Timing Path Group 'outputs'

Levels of Logic: 2.0000
Critical Path Slack: -0.1530
```

- Unrealistic I/O delay? Missing multicycle path or false path exceptions?

7-23

## Case-2: Not Suitable for Path Based Analysis

- Endpoints have very large GBA delays
- Example-1: Large stage delays: path\_length / levels >> average cell delay

```
Timing Path Group 'Lclk'

Levels of Logic: 2
Critical Path Length: 1.2119
```

- Check for large net delay or large load annotations
- May need to ECO fix paths using GBA slack

- Example-2: Skewed WNS/TNS: |TNS| / violating paths no.<< |WNS|

```
Timing Path Group 'clk'

Critical Path Slack: -0.2442
Total Negative Slack: -6.7087
No. of Violating Paths: 2884
```

- Only few large violating paths which needs to be fixed with GBA slack

7-24

Examine if large net delay could be due to large fanout and if it's an issue with place and route.

## Case-3: Ready for Path Based Analysis

- Endpoints have reasonable GBA slacks

- Example-1: Reasonable GBA slack per endpoint

|                          |         |
|--------------------------|---------|
| Timing Path Group 'Lclk' |         |
| Critical Path Slack:     | -0.2123 |
| Total Negative Slack:    | -282.23 |
| No. of Violating Paths:  | 3412    |

- $|TNS| / \text{violating paths no.} < \text{PBA slack improvement}$
- Suitable for *path mode* PBA run

- Example-2: Skewed WNS/TNS:  $|TNS| / \text{violating paths no.} \ll |WNS|$

|                         |         |
|-------------------------|---------|
| Timing Path Group 'clk' |         |
| Critical Path Slack:    | -0.2442 |
| Total Negative Slack:   | -6.7087 |
| No. of Violating Paths: | 2884    |

- Endpoints with smaller GBA violations: good for exhaustive PBA runs

7-25

Note that in Example-2, there are only a few large violating paths, that are not suitable for PBA – Recall Example-2 in Case-2

## Case-4: Must for Exhaustive Path Based Analysis

- Endpoints with positive *path mode* GBA slacks
- Example: Design meets timing with *path mode* PBA

```
report_qor -pba_mode path

Timing Path Group 'Mclk'

Critical Path Slack: 0.0000
Total Negative Slack: 0.0000
No. of Violating Paths: 0
```

- Exhaustive PBA must be run at this stage, to report the real WNS and TNS of the design

```
report_qor -pba_mode exhaustive

Timing Path Group 'Mclk'

Critical Path Slack: -0.0112
Total Negative Slack: -0.0189
No. of Violating Paths: 3
```

7-26

## Derate Only PBA

- Enabled by `set_app_var pba_derate_only_mode true`
  - Only path deratings are re-evaluated during PBA
  - Slew recalculation is not performed
  - Provides most of the slack improvement in lesser runtime
  - Also supported in AOCV, POCV, and SMVA flows
- Path-specific slew propagation using Exhaustive PBA further improves slack
  - Runtime can be longer
  - Use slew recalculation only if an endpoint shows negative slack in derate-only PBA run

Set `pba_derate_only_mode` to `true` for best runtime in *exhaustive* path-based analysis

7-27

AOCV: Advanced On Chip Variation Analysis

POCV: Parametric On Chip Variation Analysis

SMVA: Simultaneous Multi Voltage Analysis

## Path Based Analysis: Summary

- ❑ Use *path mode* PBA for most of your analysis runs  
`-pba_mode path`
- ❑ Switch to *derate-only exhaustive PBA* only when path mode PBA shows positive slack  
`set_app_var pba_derate_only_mode true`  
`-pba_mode exhaustive`
- ❑ Use *slew recalculation in exhaustive PBA* only for the endpoints which fail in *derate-only mode*  
`set_app_var pba_derate_only_mode false`  
`-pba_mode exhaustive`
- ❑ Limit runtime for *Exhaustive PBA* by controlling paths per end point  
`set_app_var pba_exhaustive_endpoint_path_limit 1000`  
`-pba_mode exhaustive`

7-28

## Lab 7: PBA



30 minutes

Run PBA in *path* and *exhaustive* modes

**012134 : Accurate Signoff Analysis with  
Path-Based Analysis in PrimeTime**

<http://solvnet.synopsys.com/retrieve/012134.html>

7-29

This page was intentionally left blank

# Agenda

DAY  
**3**

**8** Signal Integrity: Crosstalk Delay Analysis



**9** Signal Integrity: Crosstalk Noise Analysis



**10** Correlation: POCV and AWP Analysis

**11** Timing Closure: ECO/What If Analysis

**12** Large Data: DMSA and Hyperscale Analysis

**13** Conclusion

## Unit Objectives



After completing this unit, you should be able to:

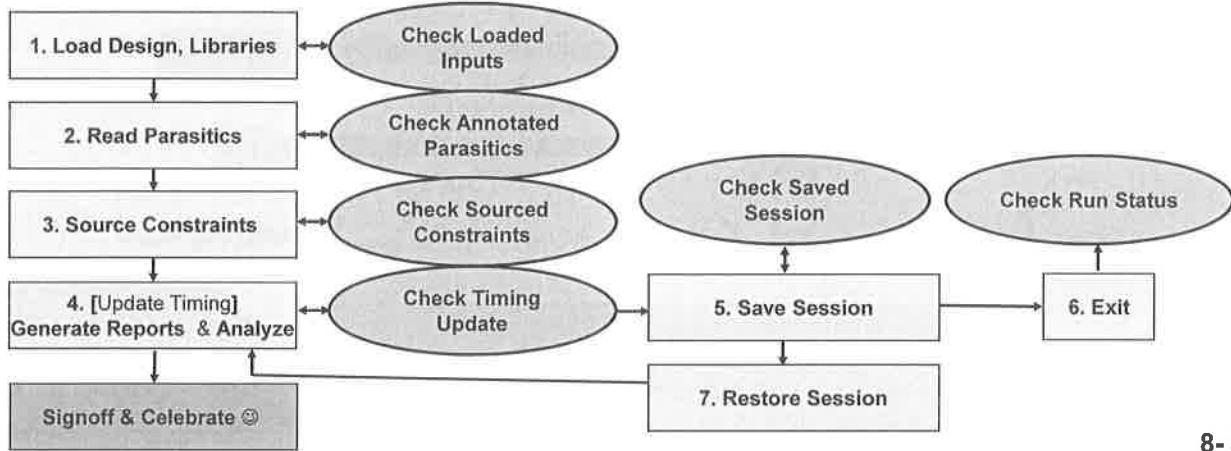
- Describe how crosstalk impacts *setup* and *hold* requirements
- List 3 requirements for performing crosstalk delay analysis in *PrimeTime-SI*
- State the impact of incorrectly or incompletely constrained clocks and I/Os on the crosstalk delay analysis
- Provide 2 mechanisms using which `update_timing` is controlled when performing crosstalk delay analysis
- Use `report_delay_calculation` to find crosstalk information

# Timing Analysis Flow in PrimeTime – SI Delay Analysis

## ■ STA flow is divided into several steps

- Steps 1-3 read data and
- Analysis begins at Step-4

## ■ The STA flow is repeated until signoff is achieved

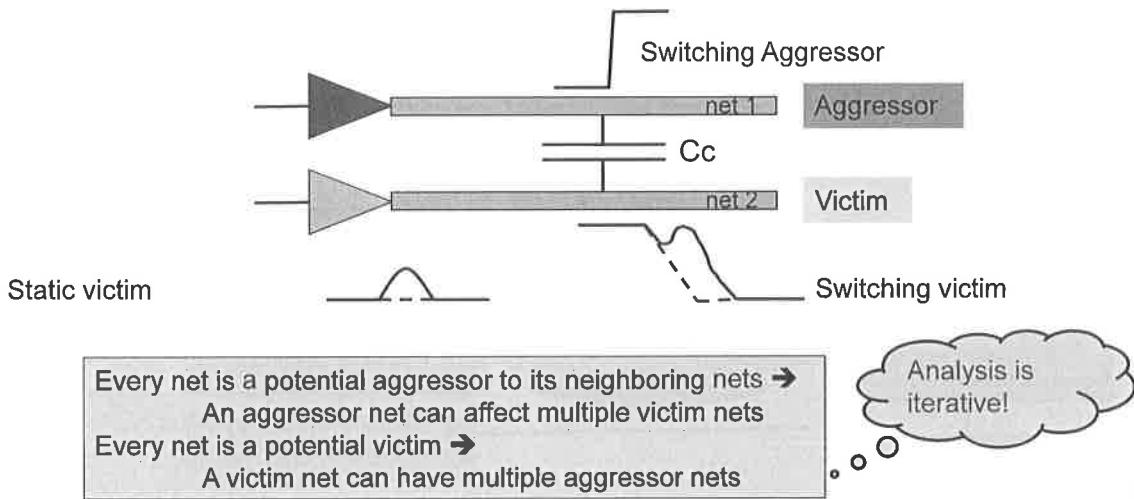


8-3

STA: Static Timing Analysis

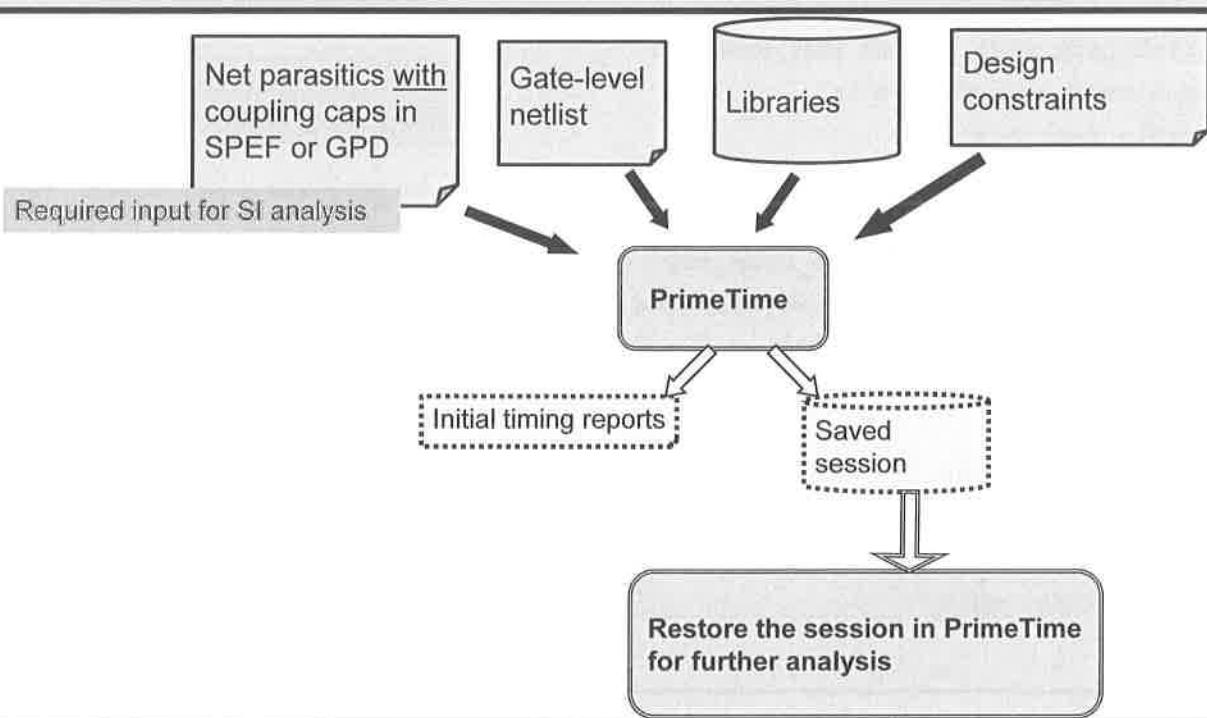
## What is Crosstalk?

Crosstalk is the transfer of a voltage transition from multiple switching nets (aggressors) to another static or switching net (victim) through a coupling capacitance ( $C_c$ ).



8-4

## PrimeTime Inputs and Outputs



8-5

SPEF: IEEE Standard Parasitic Exchange Format (ASCII)

GPD: Synopsys Galaxy Parasitic Database (Binary)

## PrimeTime SI Delay Analysis Run Script

```
lappend link_path tech.lib.db RAM_model.db
source application_variables.tcl
set si_enable_analysis true

read_verilog ORCA.v; current_design ORCA; link_design
read_parasitics -keep_capacitive_coupling -format GPD gpd_dir
read_parasitics -keep_capacitive_coupling -format spef ORCA.spef.gz
source func_maxPVT_constraints.tcl
update_timing
check_timing;
report_analysis_coverage
report_timing -crosstalk
save_session func_maxPVT_savesession
print_message_info; quit
```

SI specific variable settings / options are shown in **bold**.

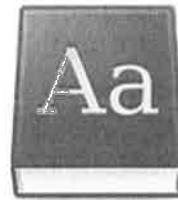
./RUN.tcl

8-6

PrimeTime SI supports only SPEF or compressed SPED (gzip) or GPD (Galaxy Parasitic Data) parasitic formats.

## PrimeTime SI Terminology

These words and concepts will show up in PrimeTime reports



Stage delay

Overlapping timing windows

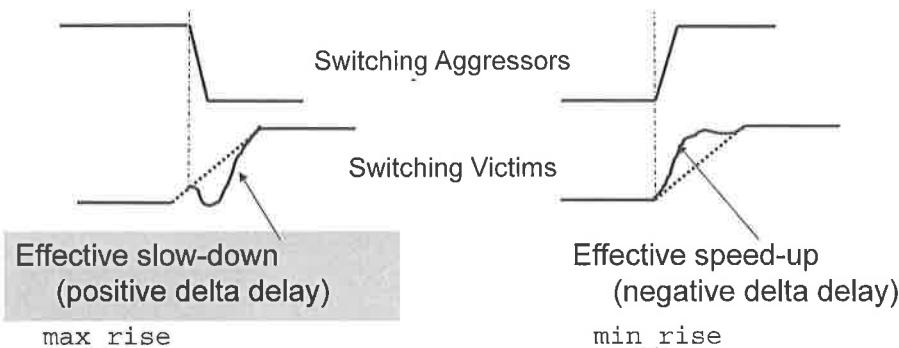
Delta delay

Delta slew

Switching bumps

## What Is a Stage Delay?

- A victim's stage delay consists of
  - The cell + net delay, calculated together
  - 4 numbers - min/max rise/fall delays
- A stage delay is used to
  - Determine each stage arrival windows
  - Calculate data arrival and required time for setup/hold analysis
- A stage delay is calculated considering the worst case effects

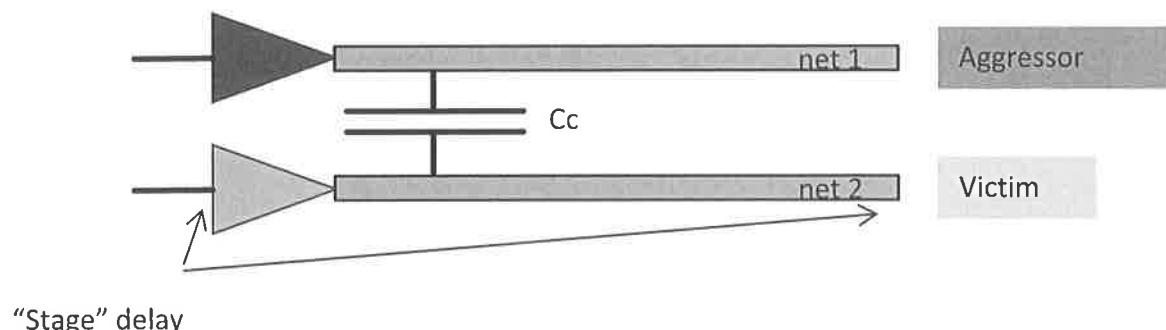


8-8

The STA results using PrimeTime SI will be the same or more conservative than the STA results using regular PrimeTime.

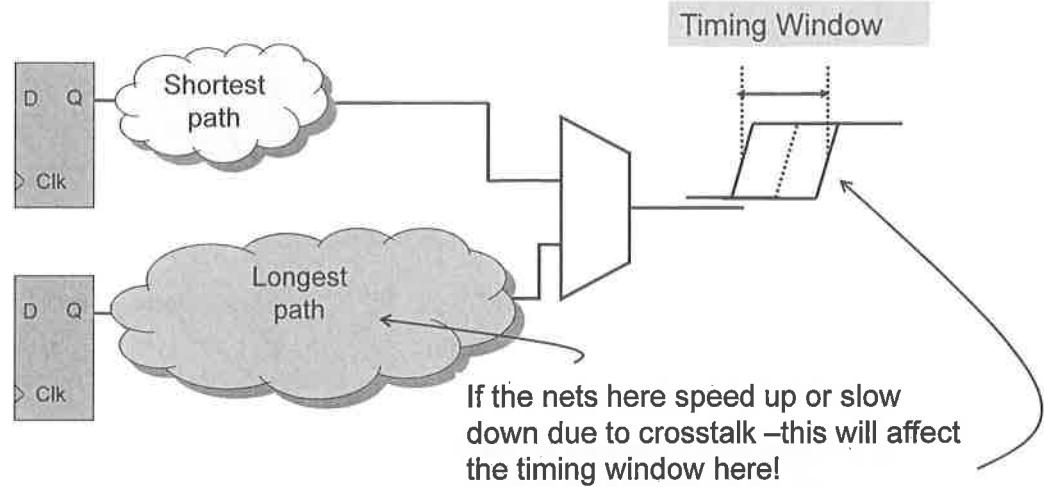
In addition to assuming the worst case rise/fall transition combination between victim and aggressor nets, PrimeTime SI does the following to calculate each stage delay.

- The sharpest transition is used for each aggressor net
- The slower transition is used for the victim net
- If the timing windows overlap between victim and aggressor, the assumption is made that the aggressor will switch at the earliest or the latest arrival times of the victim such that the min and max stage delay of the victim is made worse.



## What Are Timing Windows?

- Each stage has a timing window, defined by the shortest and longest path delays leading up to that net

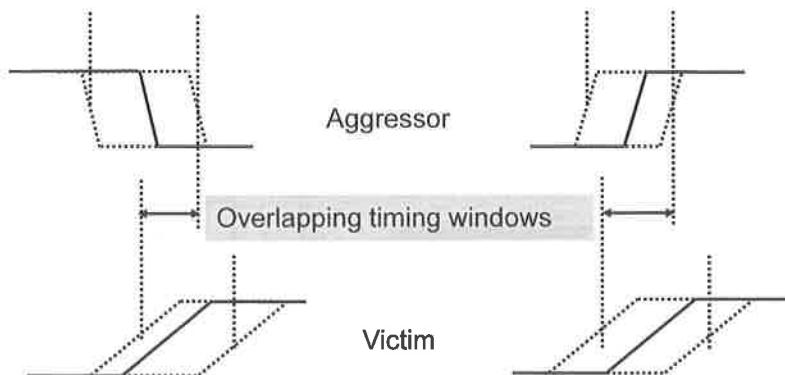


8-9

Timing windows are affected by short and long paths, slew propagation, crosstalk effects. Timing windows are important in determining overlap between victim and aggressor nets, AND they are used to determine the path delay for setup and hold analysis.

## What Are Overlapping Timing Windows?

- Victim nets that switch around the same time as aggressor nets are said to have “overlapping timing windows”.

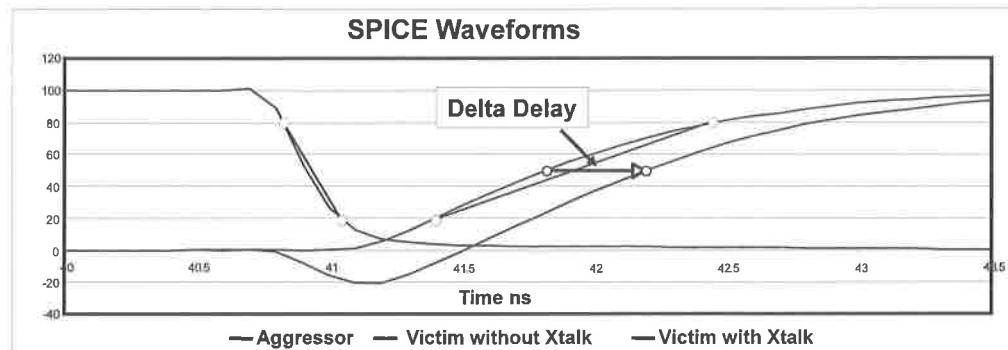


If an aggressor causes a voltage bump on a victim without any window overlap, it is NOT analyzed as crosstalk delay!

8-10

## Partial Window Overlap is also considered!

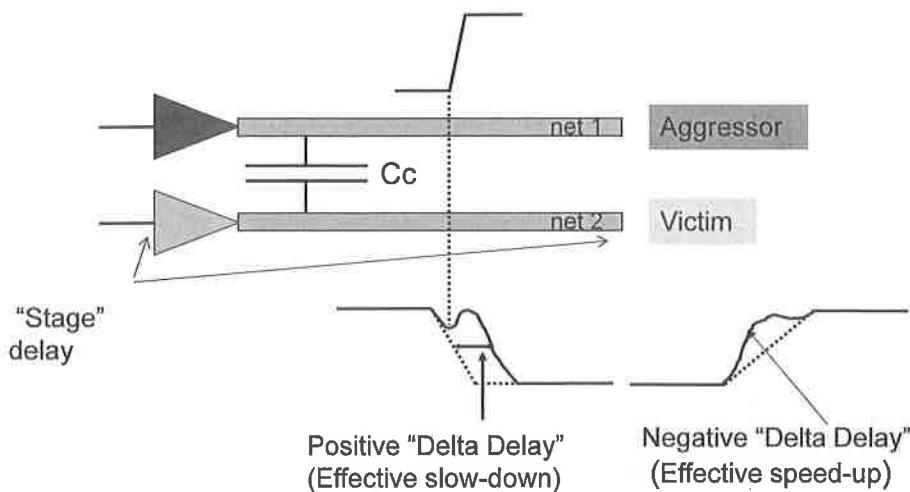
- If victim and aggressor arrival windows do not overlap at the net driver input, delta delay effects of waveforms beyond slew trip points may still be calculated
  - Called partial overlap
  - Can not use min/max arrival times to determine timing windows – too simplistic!



8-11

## What Is Crosstalk-Induced Delay?

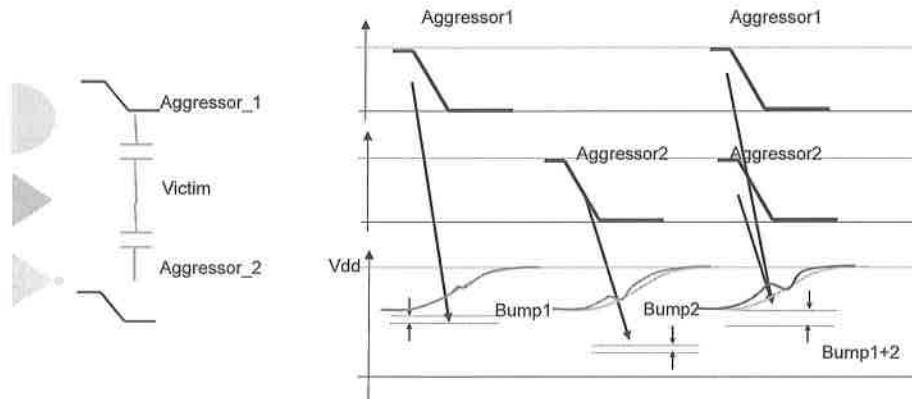
- Switching aggressor/victim nets with overlapping timing windows can cause crosstalk-induced delay and slew on victim nets.



8-12

## Switching Bumps versus Delta Delay

- A “switching bump” is expressed as a ratio of VDD
  - It is used to calculate delta delay on a victim
  - It is influenced by drive strengths, size of the coupling capacitance and the net resistance



8-13

A switching bump is not the same thing as a noise bump. A noise bump is measured when the victim is in a steady state (either held at zero or Vdd) whereas a switching bump is measured when the victim is transitioning.

## Key Messages : SI Delay Analysis

Delta delay is caused by coupling capacitances between switching aggressor and switching victim nets:

- 1) Positive delta delay causes potential setup violation and
- 2) Negative delta delay causes potential hold violation

**Positive (Negative) delta delay is caused by aggressor and victim nets switching in the opposite (same) direction**

Aggressors that are coupled to a victim net, but do not have overlapping timing windows, **do not contribute to the calculated delta delay: The non-overlapping aggressor nets are “screened” from analysis**

A switching bump is not the same thing as delta delay:

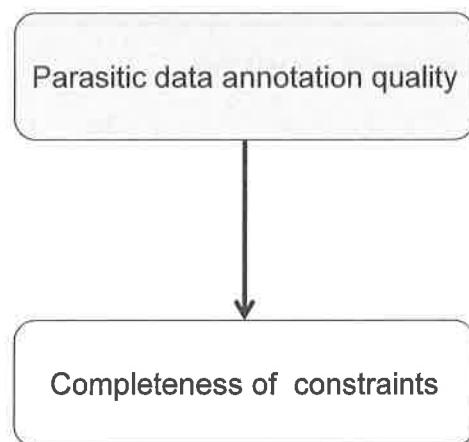
- 1) Switching bump is expressed as a ratio of VDD.
- 2) The switching bump is used to determine delta delay which is in time units

8- 14

## Ensuring Correct Inputs for Crosstalk Analysis



- For an accurate crosstalk analysis, inputs must be complete and correct
  - Any error(s) while reading inputs must be analyzed/fixed



8-15

## Reading Parasitics – Complete and No Errors

```
pt_shell> read_parasitics -keep_capacitive_coupling clock_gen.spf

Error: Driver pin 'I342/Y' is missing in the RC annotation for net 'N556'.
Ignoring the incomplete RC annotation. (PARA-006)

Executing command 'report_annotated_parasitics':

Net Type | Total | Lumped | RC pi | RC network | Coupled network | Not Annotated |
-----+-----+-----+-----+-----+-----+
Internal nets | 4932 | 0 | 0 | 369 | 4560 | 3 |
- Driverless nets | 0 | 0 | 0 | 0 | 0 | 0 |

Boundary/port nets | 105 | 0 | 0 | 0 | 105 | 0 |
- Driverless nets | 0 | 0 | 0 | 0 | 0 | 0 |

| 5037 | 0 | 0 | 369 | 4665 | 3 |
```

8-16

To find more information on missing RC annotations, use the following switch.

**report\_annotated\_parasitics -list\_not\_annotated**

```
pt_shell> report_annotated_parasitics -help
```

Usage:

```
report_annotated_parasitics # Report on annotated RC parasitics
[-check] (Check the integrity of RC networks)
[-internal_nets] (Report on internal nets)
[-boundary_nets] (Report on port nets)
[-max_nets num] (Max number of nets reported for the list
 options)
[-list_annotated] (List net drivers which are backannotated)
[-list_not_annotated] (List net drivers which are not backannotated)
[-constant_arcs] (Keep a separate count for constant nets)
[net_list] (Specific nets to report)
```

## Reading Parasitics – Background Processing

- Whenever `set_host_options -max_core` is set to greater than 1 (default is 4), parasitic annotation is launched as a background process
- **Recommendation: read parasitics prior to reading constraints**
  - Parasitics will run in background while constraints are being applied.
- **When you issue the `read_parasitics` command it will return a status flag value “1” immediately**
  - This means the command was accepted and is running
- **When annotation is complete, the `parasitic_command.log` file will appear in your working directory**
  - Contains the details of the annotation success

8-17

## Resolve All Parasitic Warnings

- **Use SPEF (or GPD) with coupling capacitances**
- **Before loading parasitics, syntax check:**
  - `read_parasitics -syntax_only`
- **RC-\*, PARA-\* or DES-\* warnings indicate delay calculation, parasitic, design or library issues**
  - These warnings may indicate that STA results will be conservative or incorrect
  - “Ok” results in regular PrimeTime may be more conservative in PrimeTime SI

Use `print_message_info` command that lists all warnings that occurred during run (even if they are suppressed)

8-18

If you did not save the log file or the results from `print_message_info`, there is one more approach you can use to find nets with RC-0\* warnings. The command `report_delay_calculation` issues these warnings again if asked to generate a report for a given net.

## Boundary (I/O) Drive and Load Constraint Requirements

- **Use driving cells (`set_driving_cell`) for both data and clock input ports**
  - Allows PrimeTime to create a driver model for accurate stage delay calculation
- **Do not use `set_input_transition` or leave the ports unconstrained without a drive**
  - Without a driving cell, input stage (Cell + Net) will be missing
  - Will result in inaccurate delta delay calculations

```
set_driving_cell -lib_cell buffd1 -max [all_inputs]
set_driving_cell -lib_cell buffd7 -min [all_inputs]
set_load -max $max [all_outputs]
set_load -min $min [all_outputs]
set_clock_latency -early $min -source [get_clock clk]
set_clock_latency -late $max -source [get_clock clk]
```

8-19

# Constraining Input Ports with Input Delays

## ■ Specify min and max input delays accurately

- Values are used to define arrival windows on input ports

1 set\_input\_delay -max 3 -clock CLK [get\_ports A]

2

set\_input\_delay 3 -clock CLK [get\_ports A]

set\_input\_delay -max 3 -clock CLK [get\_ports A]  
set\_input\_delay -min 2 -clock CLK [get\_ports A]

3

Circle the correct set  
of constraints

Zero arrival windows will give

- Optimistic results.
- Conservative results

8-20

Clocks should also have accurate min/max early/late source latencies defined. This will define the arrival windows for the clock nets (equivalent to min/max input delays on launch paths).

pt\_shell> man set\_input\_delay

• • •

-max

Specifies that delay\_value refers to the longest path. If you do not specify -max or -min, maximum and minimum input delays are assumed to be equal.

-min

Specifies that delay\_value refers to the shortest path. If you do not specify -max or -min, maximum and minimum input delays are assumed to be equal.

The third set of constraints is the correct one. The first set only specifies max input delay, the min will be zero which is pessimistic on the timing window. The second will still result in a zero timing window as the min/max input delay will be equivalent!

Zero timing windows are optimistic! Will not see potential crosstalk affects (aggressors will be screened because they do not overlap).

## (A)synchronous Clocks: Delta Delay Analysis

Aggressor/victim nets associated with asynchronous clocks should consider:

- No delta delay analysis.
- Delta delay analysis with “infinite” arrival windows.**
- Delta delay analysis only when arrival windows overlap.

Aggressor/victim nets associated with synchronous clocks should consider:

- No delta delay analysis.
- Delta delay analysis with “infinite” arrival windows.
- Delta delay analysis only when arrival windows overlap.**

8-21

### Infinite arrival windows

All aggressor transitions are combined to calculate the worst case delta delay and slew on the victim, irrespective of timing. For asynchronous clocks, there is no fixed relationship between the clocks and thus PrimeTime SI assumes the transitions can occur at any time.

### Synchronous clock analysis

All victim/aggressor pairs associated with synchronous clocks will be analyzed for crosstalk when arrival windows overlap. The overlapping timing window analysis will be done over the smallest common base period for the synchronous clock group.

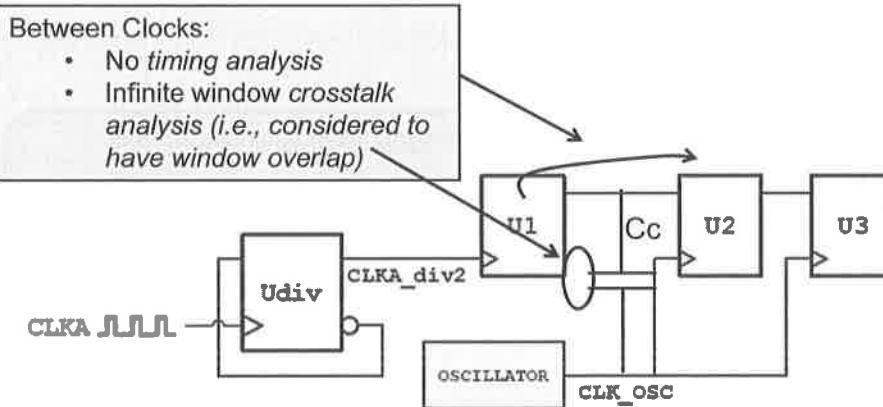
Asynchronous clocks should result in infinite window overlap as the clock edges can occur at any time (this relates not only to the clock paths themselves but data paths that are related to asynchronous clocks).

Synchronous clocks should result in crosstalk delays when windows overlap.

## Asynchronous Clocks

- Asynchronous clocks do not have functional paths between them
- They are analyzed *conservatively* causing delta delay due to crosstalk

```
set_clock_groups -asynchronous -group {CLKA CLKA_div2} -group CLK_OSC
```



8-22

The command **set\_clock\_groups** is an SDC (Synopsys Design Constraints) command

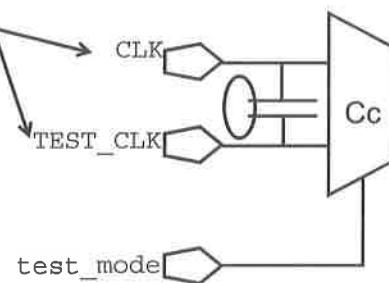
## Logically Exclusive Clocks

- Logically Exclusive clocks do not have functional paths between them
- They are analyzed for *possibly* causing delta delay due to crosstalk

```
set_clock_groups -logically_exclusive -group CLK -group TEST_CLK
```

Between clocks:

- No *timing analysis*
- Crosstalk calculated using overlapping timing windows



8-23

The command `set_clock_groups` is supported by SDC (Synopsys Design Constraints).

## Physically Exclusive Clocks

- Physically Exclusive clocks do not co-exist on the design simultaneously to be affected by functional paths or crosstalk between them

Two clocks on a single port

```
create_clock -name CLK_FUNC [get_ports CLK] -period 6
create_clock -name CLK_TEST [get_ports CLK] -period 120 -add
set_clock_groups -physically_exclusive -group CLK_FUNC -group CLK_TEST
```

Between CLK\_FUNC and CLK\_TEST:

- No *Timing Analysis*
- No *crosstalk analysis*

8-24

The command **set\_clock\_groups** is supported by **write\_sdc**.

## Clock Relationships – Summary (1/2)

### ■ Use clock groups to define clock relationships

- Timing paths between the clock groups are ignored for STA
- Crosstalk calculations on victim/aggressor nets associated with
  - ◆ Asynchronous clocks -- infinite window
  - ◆ Logically exclusive clocks – overlapping timing windows
  - ◆ Physically exclusive – no crosstalk

Invisible to you:  
no information  
messages

### ■ Do not use `set_false_path` for asynchronous clocks

- PrimeTime SI assumes false paths are synchronous
  - ◆ It extends the clock periods, trying to find a common base period
  - ◆ Delta delay is calculated only for overlapping timing windows in that extended clock period
- Analysis is optimistic!

### ■ *Continued...*

8-25

If all clocks are asynchronous to each other, use the following script to constrain them.

```
foreach_in_collection clk [all_clocks] {
 set_clock_groups -async -group $clk
}
```

## Clock Relationships – Summary (2/2)

### ■ Do not use both set\_false\_path and clock groups

- set\_clock\_groups overwrites already defined set\_false\_path

```
set_false_path -from Clk1 -to Clk2
set_false_path -from Clk2 -to Clk1
set_clock_groups -asynchronous -group Clk1 -group Clk2 → Clock group wins!
```

- An error message is issued if set\_false\_path is applied between clocks already defined using clock groups [exclusive/asynchronous]

```
set_clock_groups -asynchronous -group Clk1 -group Clk2
set_false_path -from Clk1 -to Clk2
set_false_path -from Clk2 -to Clk1 → Error!
```

### ■ Do not over constrain synchronous clocks (by changing clock period)

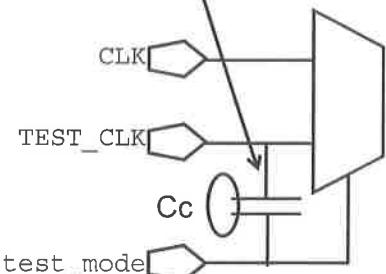
- If clock relationship shifts, may produce new or hide real timing window overlaps
- Use Clock uncertainty to model timing margin

## Constraining Static Nets – Case Analysis

### ■ Constrain all static nets in the design using case analysis

- Static nets cannot be an aggressor nor victim for crosstalk delay analysis
- This will turn off regular STA on that specific timing path
- This will turn off the ability for that net (and all nets through which the case value propagates) to be an aggressor

No delta delay analysis  
When doing separate runs for each mode  
`set_case_analysis 0 test_mode`



When merging modes into a single run  
`set_si_delay_analysis -exclude -victims/-aggressors [get_nets test_mode]`

8-27

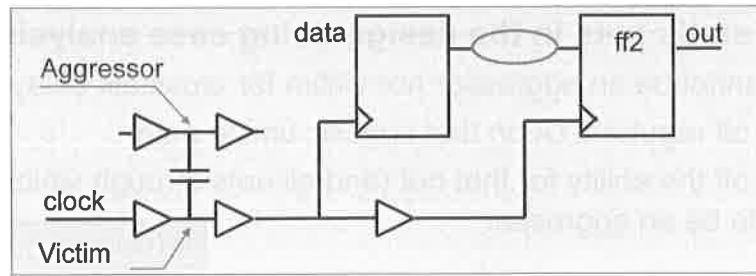
If you are doing separately-constrained runs for functional and test, your static nets should already be constrained.

If you are merging constraints into a single master file, the above may be more difficult, and you may have to resort to `set_si_delay_analysis -exclude`.

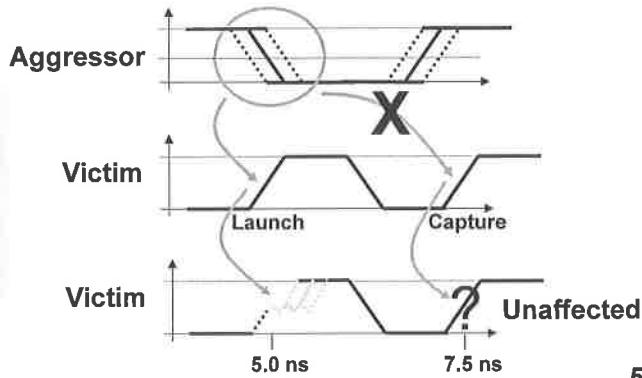
```
set_si_delay_analysis
[-ignore_arrival inets]
[-exclude]
[-victims vnets]
[-aggressors anets]
[-rise]
[-fall]
[-min]
[-max]
-exclude
```

Indicates that nets specified as a *vnets* or *anets* variable are to be excluded from the crosstalk analysis as victim nets or aggressor nets, respectively. You cannot use this option with the `-ignore_arrival` option. When both the `-victims vnets` and `-aggressors anets` options are applied, all cross capacitances between the *vnets* and *anets* variables are excluded, when *vnets* are victims and *anets* are aggressors.

## PrimeTime SI - CRPR in Setup Analysis



**CRP due to delta delays on different clock edges (e.g. setup analysis) is NOT removed**

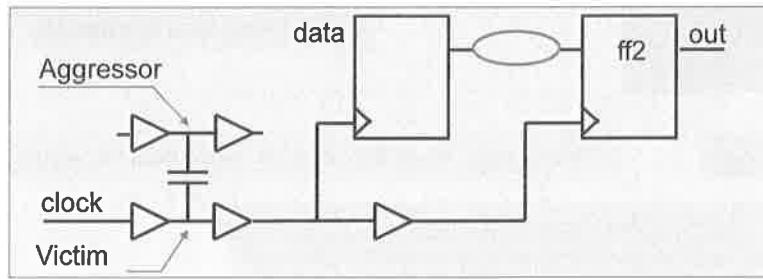


5

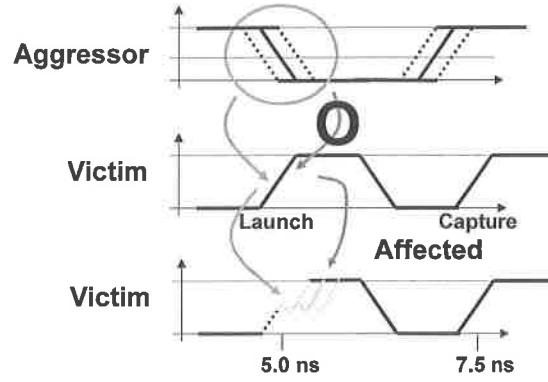
8-28

Answer: NO, CRP due to crosstalk should not be removed. Non-crosstalk related CRP should be removed. This is because crosstalk can affect different clock edges in different ways and thus subsequent clock edges can have different delays to the common point.

## PrimeTime SI - CRPR in Hold Analysis



**CRP due to delta delays on the same clock edge (e.g. hold analysis) is removed**



6

8-29

Crosstalk can not affect the same clock edge in two different ways.

## update\_timing Under the Hood

### Crosstalk Delay Calculation Flow:

update\_timing:

set\_app\_var si\_xtalk\_composite\_aggr\_mode statistical

#### Electrical Filtering:

Screen out aggressor nets with small bumps

#### Delay Calculation with infinite windows

#### Delay Calculation with window overlap

### User Controllable

set\_app\_var si\_xtalk\_delay\_analysis\_mode all\_path\_edges

8-30

All optional user steps are executed prior to update\_timing.

### si\_xtalk\_composite\_aggr\_mode

Specifies the composite aggressor mode for crosstalk delay.

**disabled** (default) - Disables the composite aggressor feature. In the **disabled** mode, PrimeTime SI uses its original flow with composite aggressor completely off to calculate the crosstalk delay.

**statistical** - Causes PrimeTime SI to calculate crosstalk by using the statistical composite aggressor flow. The **statistical** composite aggressor mode reduces the pessimism for crosstalk delay analysis by reducing the effect of composite aggressor.

### si\_xtalk\_delay\_analysis\_mode

Specifies the arrival window alignment mode for crosstalk delay. In the **all\_paths** (default) alignment mode, PrimeTime SI considers the largest possible crosstalk delta delay for the given victim & aggressor arrival windows. This analyzes all paths going through the victim net with different arrival times conservatively. This is the default and traditional way PrimeTime SI calculates delta delay. In the **all\_path\_edges** alignment mode, PrimeTime SI considers all paths that pass through a victim by keeping track of the individual leading edges of those paths. This mode is less pessimistic than the **all\_paths** mode, while at the same time is safe for signoff.

## Composite Aggressor Mode

### ■ In UDSM technologies (65nm and below)

- Trend is for larger numbers of very small aggressors
- They may fall below electrical filtering threshold and may be screened from analysis

### ■ Two approaches for addressing this:

- Adjust down the electrical filtering ratios
  - ◆ This is effective but very expensive in run time
- Use Composite Aggressor Mode
  - ◆ Lumps the filtered aggressors together into a single “composite aggressor” which is then statistically modelled and included in victim SI analysis

Simply adding all composite aggressors assumes they all switch at once, which is overly pessimistic. Statistical is more realistic and less pessimistic.

```
set_app_var si_xtalk_composite_aggr_mode statistical
```

8- 31

For further details refer to:

Solvnet Article #030997 “How does composite aggressor filtering work in PrimeTime SI?”

PT User Guide

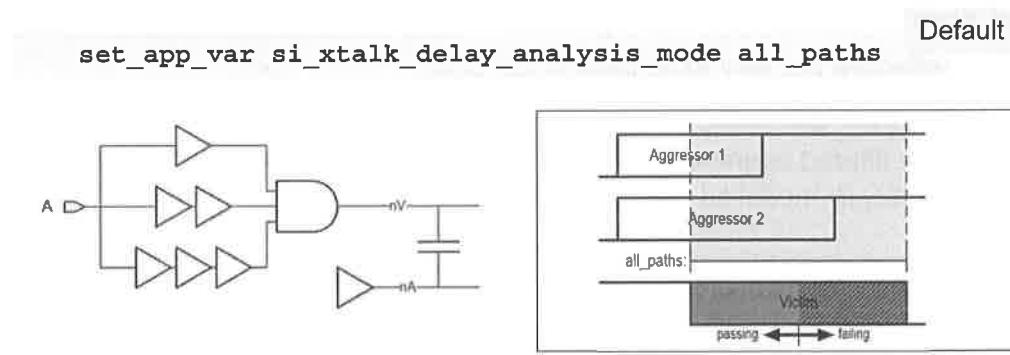
[https://solvnet.synopsys.com/dow\\_retrieve/latest/dg/ptolh/Content/ni/ptug/ni/ptug.pdf](https://solvnet.synopsys.com/dow_retrieve/latest/dg/ptolh/Content/ni/ptug/ni/ptug.pdf)

Search for: “Crosstalk Analysis with Composite Aggressors”

## Timing Window Overlap : all\_paths SI Window Alignment

### ■ By default, PrimeTime SI uses a window alignment mode called “all\_paths”

- Computes delta delays using the entire arrival window of the victim net
- Arrival window is bounded by
  - ◆ earliest possible minimum delay arrival
  - ◆ latest possible maximum delay arrival
- The resultant window is used to calculate delta delay for both min and max paths



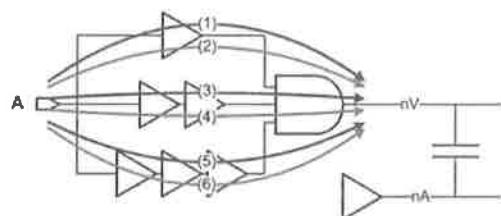
8-32

In the **all\_paths** alignment mode, PrimeTime SI considers the largest possible crosstalk delta delay for the given victim & aggressor arrival windows. This analyzes all paths going through the victim net with different arrival times conservatively. This is the default and traditional way PrimeTime SI calculates delta delay.

## Introducing all\_path\_edges Window Alignment (Max path)

### Max Path

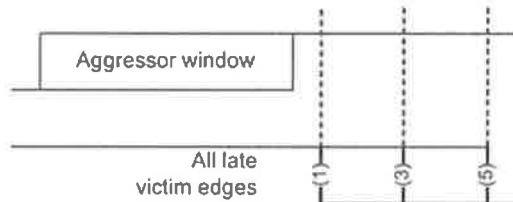
- Use victim edge arrivals instead of the entire arrival window



In the example, the edge arrivals are:

- 1) Max delay arrival through top path
- 2) Min delay arrival through top path
- 3) Max delay arrival through middle path
- 4) Min delay arrival through middle path
- 5) Max delay arrival through bottom path
- 6) Min delay arrival through bottom path

- For the max path, delta delay calculation uses edges 1, 3, and 5



- No arriving maximum delay edge overlaps the aggressor window

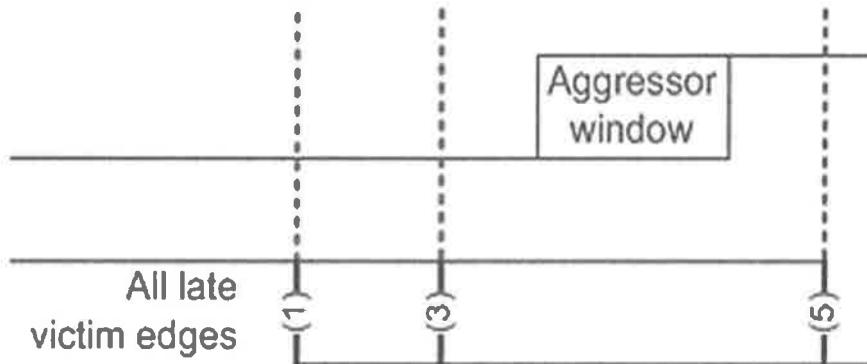
- Thus delta delay = ZERO !

```
set_app_var si_xtalk_delay_analysis_mode all_path_edges
```

8-33

In the **all\_path\_edges** alignment mode, PrimeTime SI considers all paths that pass through a victim by keeping track of the individual leading edges of those paths. This mode is less pessimistic than the **all\_paths** mode, while at the same time is safe for signoff.

Sometimes an aggressor window can exist between multiple victim edges without any overlap. In such cases, computed delta delay will be ZERO

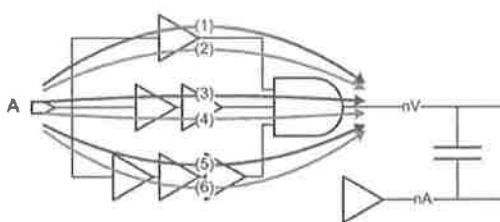


Should I use **all\_path\_edges** all the time?

Yes – the small increase in runtime (up to 10%) is worth it!

## Introducing all\_path\_edges Window Alignment (Min path)

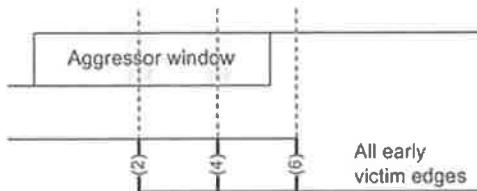
### Min Path



In the example, the edge arrivals are:

- 1) Max delay arrival through top path
- 2) Min delay arrival through top path
- 3) Max delay arrival through middle path
- 4) Min delay arrival through middle path
- 5) Max delay arrival through bottom path
- 6) Min delay arrival through bottom path

- For the min path, delay delta calculation uses edges 2, 4 and 6

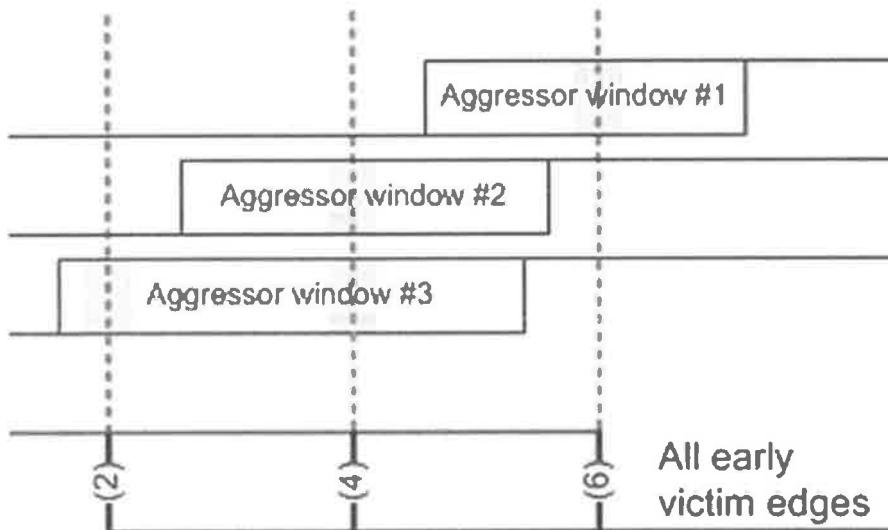


- Here there are edges occurring during the aggressor window
- The delta delay value is non-zero

```
set_app_var si_xtalk_delay_analysis_mode all_path_edges
```

8-34

If multiple victim edges align with aggressor windows, the worst delta delay across all the overlapping victim edge combinations is used for all paths through the net

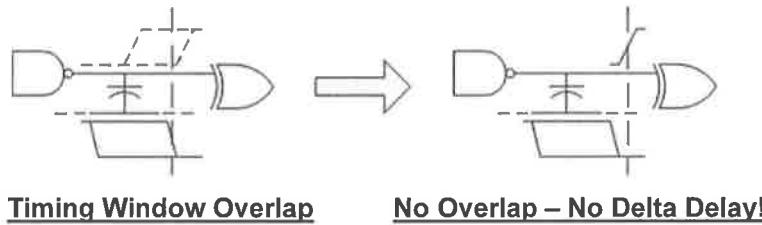


Should I use all\_path\_edges all the time?

Yes – the small increase in runtime (up to 10%) is worth it!

## SI and Path-Based Analysis

- Because path-based analysis considers only a single timing path, this eliminates the need to propagate a timing window down this path
  - SI effects calculated on a single edge for the victim nets
    - ◆ Arrival windows still used for aggressors from last timing update
  - SI effects are further reduced as the actual path-specific slews are propagated and not the worst case slew
    - ◆ Slower slews are more susceptible to crosstalk delay effects



## Using report\_delay\_calculation -crosstalk

```
pt_shell> report_delay_calculation -from U1407/ZN -to U1416/A1 -crosstalk
```

...

Reporting for Crosstalk:  
 Victim net name: n6472  
 Number of aggressors: 4  
 Number of effective (non-filtered) aggressors: 2

...

| Victim Net | Coupling Cap | Driver Lib Cell | Clocks |
|------------|--------------|-----------------|--------|
| n6472      | 0.006352     | nd02d2          | clk    |

Sorted by  
bump height



| Aggressor Net | Coupling Cap | Driver Lib Cell | Clocks | Attributes | Switching Bump (ratio of VDD) |
|---------------|--------------|-----------------|--------|------------|-------------------------------|
| n6482         | 0.001511     | nd02d1          | clk    | A          | 0.032472                      |
| n7286         | 0.000831     | an02d1          | clk    | A          | 0.026429                      |

Victim is falling:

| Victim Net    | Coupling Cap | Driver Lib Cell | Clocks | Attributes | Switching Bump (ratio of VDD) |
|---------------|--------------|-----------------|--------|------------|-------------------------------|
| n6472         | 0.006352     | nd02d2          | clk    |            |                               |
| Aggressor Net | Coupling Cap | Driver Lib Cell | Clocks |            |                               |
| n6482         | 0.001511     | nd02d1          | clk    | A          | 0.032167                      |
| n7286         | 0.000831     | an02d1          | clk    | A          | 0.024979                      |

If not active,  
why?

Large Cc?

Downsize? Upsize?

8-36

The report from report\_delay\_calculation is large. Much detail has been removed from this slide so that it will fit on the page.

## Review Unit Objectives



After completing this unit, you should be able to:

- Describe how crosstalk impacts *setup* and *hold* requirements
- List 3 requirements for performing crosstalk delay analysis in *PrimeTime-SI*
- State the impact of incorrectly or incompletely constrained clocks and I/Os on the crosstalk delay analysis
- Provide 2 mechanisms using which `update_timing` is controlled when performing crosstalk delay analysis
- Use `report_delay_calculation` to find crosstalk information

## Lab 8: Perform Crosstalk Delay Analysis



45 minutes

Execute a run script for Crosstalk delay analysis

Invoke the shell and generate reports

Invoke the GUI and generate reports

8-38

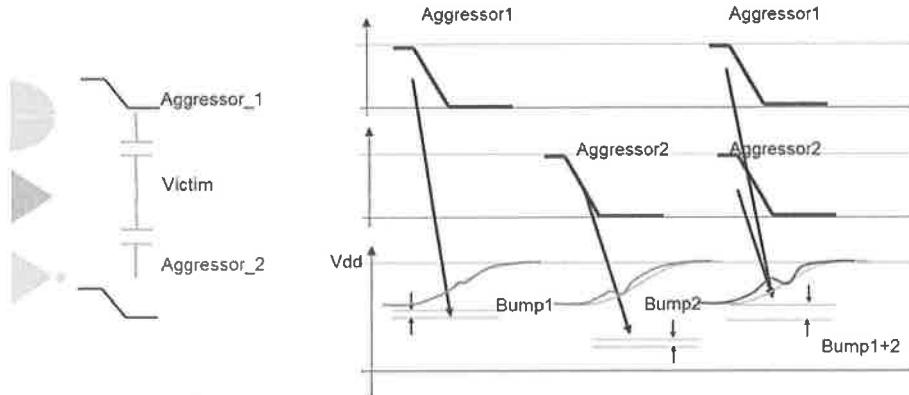
## **Appendix: Electrical Filtering Details**

**8-39**

# What is Electrical Filtering?

- To achieve accurate results in a reasonable amount of time, aggressor nets that have too small an effect on the final results are filtered

- Filtered aggressors are no longer considered for crosstalk delay analysis of that victim
- Filtered aggressors may still be used in the analysis of a different victim
- Filtered aggressors are still used in statistical composite aggressor mode



8-40

It is generally recommended to use the default settings for the following variables. Further details on electrical filtering is available in the appendix of this unit.

```
pt_shell> man si_filter_accum_aggr_noise_peak_ratio
```

NAME  
Specifies the threshold for the accumulated voltage bumps introduced by aggressors at a victim node, divided by Vcc, below which aggressor nets can be filtered out during electrical filtering.  
DEFAULT 0.03

```
pt_shell> man si_filter_per_aggr_noise_peak_ratio
```

NAME  
Specifies the threshold for the voltage bump introduced by an aggressor at a victim node, divided by Vcc, below which the aggressor net can be filtered out during electrical filtering.  
DEFAULT 0.01

Review the man page of `si_xtalk_composite_aggr_mode` to learn more about composite aggressors.

## Accumulated versus Individual Filtering

- An aggressor is filtered if it meets the criteria for accumulated or individual filtering
- Filtering method accounts for large number of small aggressor
  - Aggressors are sorted based on switching bump
  - Sorted bumps are summed, starting at smallest
  - Aggressors filtered whose bump sums are below threshold

```
set_app_var si_filter_accum_aggr_noise_peak_ratio 0.03
```

- Switching bump of each aggressor examined

- Aggressors filtered whose bump height is below threshold

```
set_app_var si_filter_per_aggr_noise_peak_ratio 0.01
```

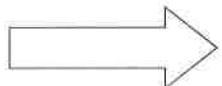
## Step #1: Aggressor Sorting

Unsorted

| Aggr | Ratio of Vcc |
|------|--------------|
| 1    | 0.0012       |
| 2    | 0.0007       |
| 3    | 0.0029       |
| 4    | 0.0213       |
| 5    | 0.0111       |
| 6    | 0.0034       |
| 7    | 0.0105       |

Sorted

| Aggr | Ratio of Vcc |
|------|--------------|
| 2    | 0.0007       |
| 1    | 0.0012       |
| 3    | 0.0029       |
| 6    | 0.0034       |
| 7    | 0.0105       |
| 5    | 0.0111       |
| 4    | 0.0213       |



8- 42

## Step #2: Combined Filtering

Sorted

| Aggr | Ratio of Vcc |
|------|--------------|
| 2    | 0.0007       |
| 1    | 0.0012       |
| 3    | 0.0029       |
| 6    | 0.0034       |
| 7    | 0.0105       |
| 5    | 0.0111       |
| 4    | 0.0213       |

- Individual aggressor filtering = 0.01
- Nets filtered individually still considered for accumulated aggressor filtering
- Accumulated aggressor filtering = 0.03

Filtered by individual filtering.

Filtered by accumulated filtering.

Only aggressor net that is kept.

## Voltage Bump Summing

### Sorted

| Aggr | Ratio of Vcc |
|------|--------------|
| 2    | 0.0007       |
| 1    | 0.0012       |
| 3    | 0.0029       |
| 6    | 0.0034       |
| 7    | 0.0105       |
| 5    | 0.0111       |
| 4    | 0.0213       |

**Sum bump heights, starting with smallest  
Filter all nets whose sum is < 0.03**

1. SUM(2) = 0.0007
2. SUM(2, 1) = 0.0019
3. SUM (2, 1, 3) = 0.0048
4. SUM (2, 1, 3, 6) = 0.0082
5. SUM (2, 1, 3, 6, 7) = 0.0187
6. SUM (2, 1, 3, 6, 7, 5) = 0.0298
7. SUM (ALL) = 0.0511

8-44

# Agenda

DAY

3

8 Signal Integrity: Crosstalk Delay Analysis



9 Signal Integrity: Crosstalk Noise Analysis



10 Correlation: POCV and AWP Analysis

11 Timing Closure: ECO/What If Analysis

12 Large Data: DMSA and Hyperscale Analysis

13 Conclusion

## Unit Objectives



After completing this unit, you should be able to:

- Modify an existing PrimeTime run script to perform crosstalk noise analysis
- Generate and interpret the key reports in the shell as well as the GUI to identify violations due to crosstalk noise
- Describe the advantages of using CCS Noise library
- Differentiate *report\_at\_source* vs. *report\_at\_endpoint* when using CCS Noise library

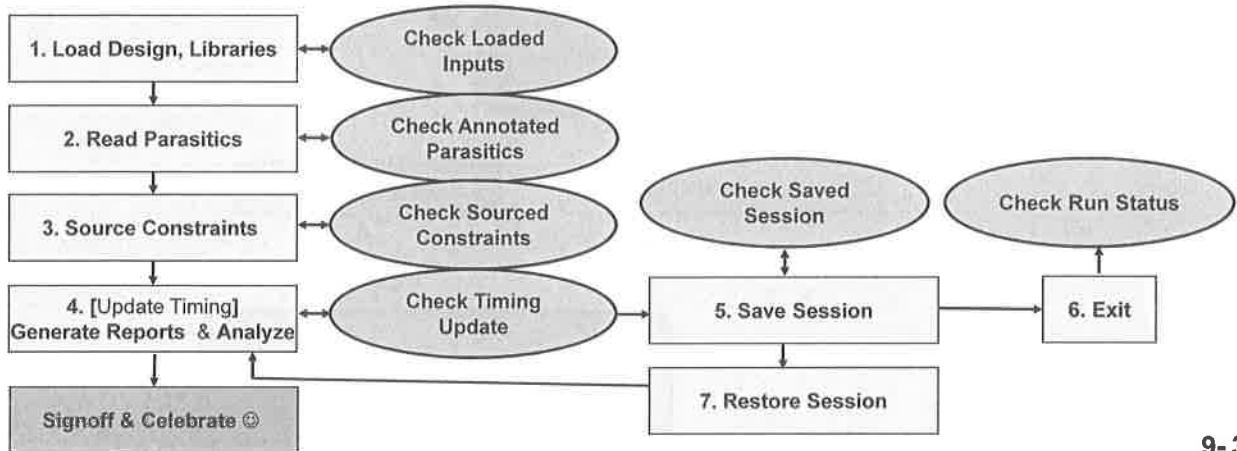
9-2

# Timing Analysis Flow in PrimeTime – SI Noise Analysis

## ■ STA flow is divided into several steps

- Steps 1-3 read data and
- Analysis begins at Step-4

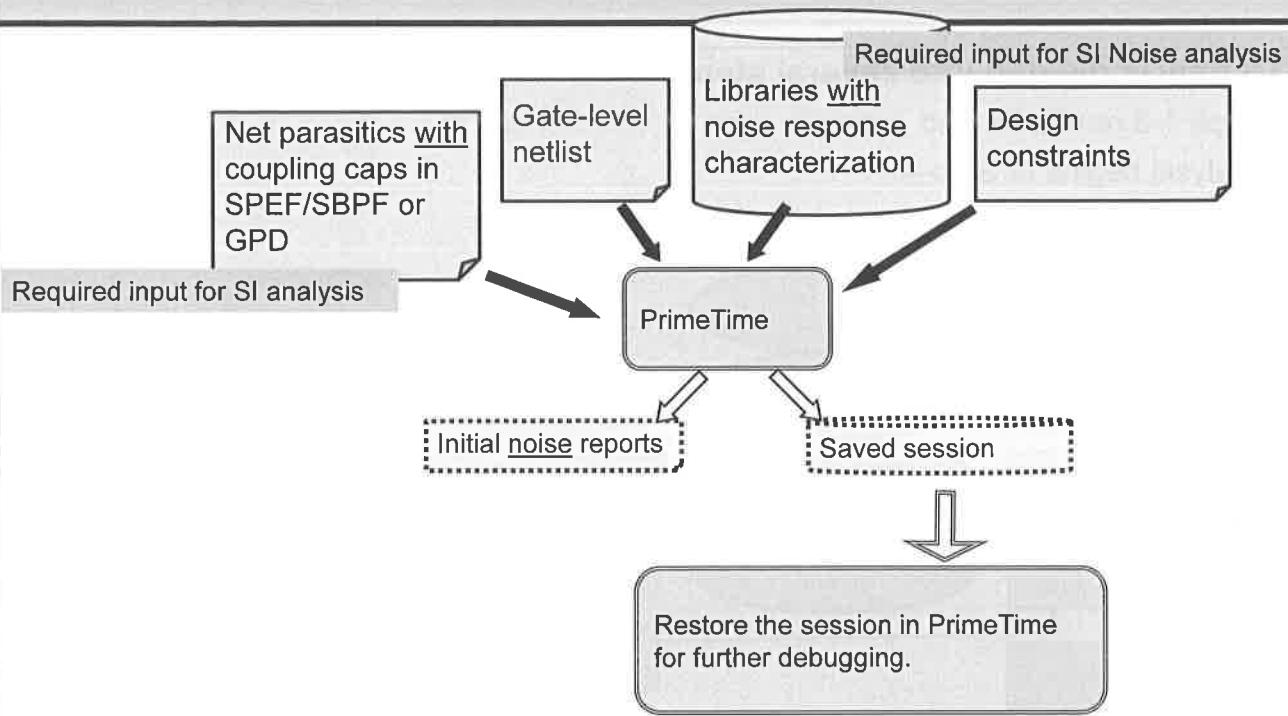
## ■ The STA flow is repeated until signoff is achieved



9-3

STA: Static Timing Analysis

## PTSI Noise Inputs and Outputs



9-4

When generating reports for noise – it will be apparent whether your library has noise response characterizations. For a Tcl procedure to validate your library without actually performing noise analysis, refer to SolvNet article Checking Consistency of Pin-Based CCS Noise Models

**Doc Id:** 2285479 <https://solvnet.synopsys.com/retrieve/2285479.html>

For a high quality noise calculation, please note that you will need either NLDM noise data (older technology) or CCS noise data (preferred) in the library.

To check if a library is fully characterized for noise analysis (Steady State R, propagated noise, noise immunity curve), the Library Screener utility can be helpful. Please contact your friendly SNPS Application Consultant for details.

3 differences – they are all underlined ☺

## PTSI Noise Run Script

```
lappend link_path tech.lib.db RAM_model.db
source application_variables.tcl
set si_enable_analysis true

read_verilog ORCA.v; current_design ORCA
link_design

read_parasitics -keep_capacitive_coupling -format GPD gpd_dir
or read_parasitics -keep_capacitive_coupling -format spef ORCA.spef
source func_maxPVT_constraints.tcl

check_timing
report_timing -crosstalk
report_noise -all_violators

save_session func_maxPVT_savesession
print_message_info; quit
```

With noise response characterization!

./RUN.tcl

9-5

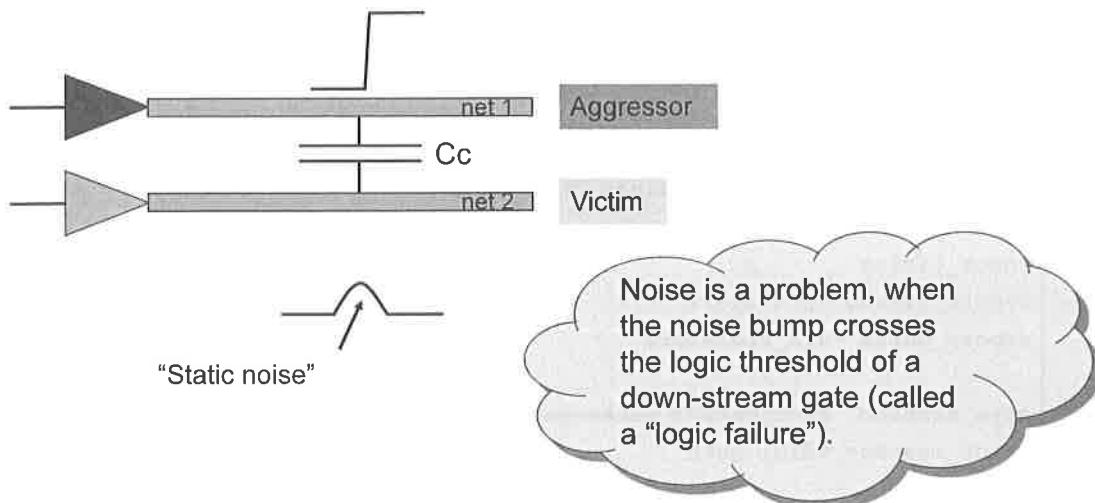
pt\_shell> report\_noise -help

Usage:

```
report_noise # Report noise analysis information
 [-above] (Report for the above rails)
 [-below] (Report for the below rails)
 [-low] (Report for the low rails)
 [-high] (Report for the high rails)
 [-nworst_pins pin_count] (Report only the nworst pins)
 [-significant_digits digits] (Number of digits to display)
 [-verbose] (Show verbose information)
 [-nosplit] (Do not split lines when columns overflow)
 [-slack_lesser_than float] (Only report slacks less than this
value)
 [-slack_type slack_type] (Set the type of slack:
 Values: height, area,
area_percent)
 [-all_violators] (Show only violating pins)
 [-data_pins] (Report for register data pins)
 [-clock_pins] (Report for register clock pins)
 [-async_pins] (Report for asynchronous pins)
 [object_list] (List of pins or ports)
```

# What Is Crosstalk-Induced Noise?

Switching aggressor nets can create crosstalk-induced noise on static victim nets, also called "static noise".



9-6

The term "noise" in electronic design generally means any undesirable deviation in voltage of a net that ought to have a constant voltage, such as a power supply or ground line. In CMOS circuits, this includes data signals being held constant at logic one or logic zero.

There are many different causes of noise such as charge storage effects at P-N junctions, power supply noise, and substrate noise. However, the dominant noise effect in deep-submicron CMOS circuits is crosstalk noise between physically adjacent logic nets. For this reason, PrimeTime SI concentrates on the analysis of crosstalk noise between these nets.

In static noise analysis, a "logic failure" is an incorrect logic value on a net resulting from the propagation of a noise bump from an input to an output of a cell.

There are two possible strategies for repairing logic failures:

- Fix logic failures that are latched and ignore logic glitches that are not latched.
- Fix all logic failures at the source, whether or not they appear to be latched.

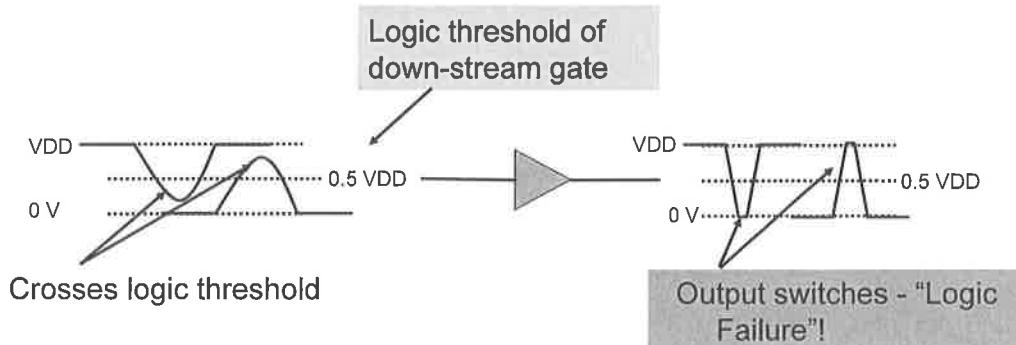
The "fix latched logic failure" strategy requires less fixing to be done, but risks the quality of the finished product. The safer, recommended strategy is to fix all logic failures at the source, which has the following advantages:

- It positively identifies and fixes the actual source of the failure (not just the latched result).
- It is less susceptible to process variation.
- It can detect failures caused by outside-rail noise (above high and below low) as well as inside-rail noise.
- It works effectively with dynamic (charge-storage) logic.

PrimeTime SI static noise analysis is based on this recommended strategy.

## Unsafe noise bumps : Crossing the Logic Thresholds

If the noise bump crosses the logic thresholds of the downstream gate, the output of that gate may generate a logic swing → Potential problem!



This is called “between-the-rails” noise bumps (below\_high and above\_low).

If the noise bump does not cross the logic threshold of the downstream gate, there is no negative effect – the noise bump is considered safe!

9-7

PrimeTime SI concentrates on four types of noise bumps caused by aggressor net transitions: above low, below low, above high, and below high. Bumps between the two rail voltages (above low and below high) can cause logic failure, if they exceed the logic thresholds of the technology. Bumps outside of the range between the two rail voltages (below low and above high) are also important because they can forward-bias pass gates at the inputs of flip-flops and latches, allowing incorrect values to be latched.

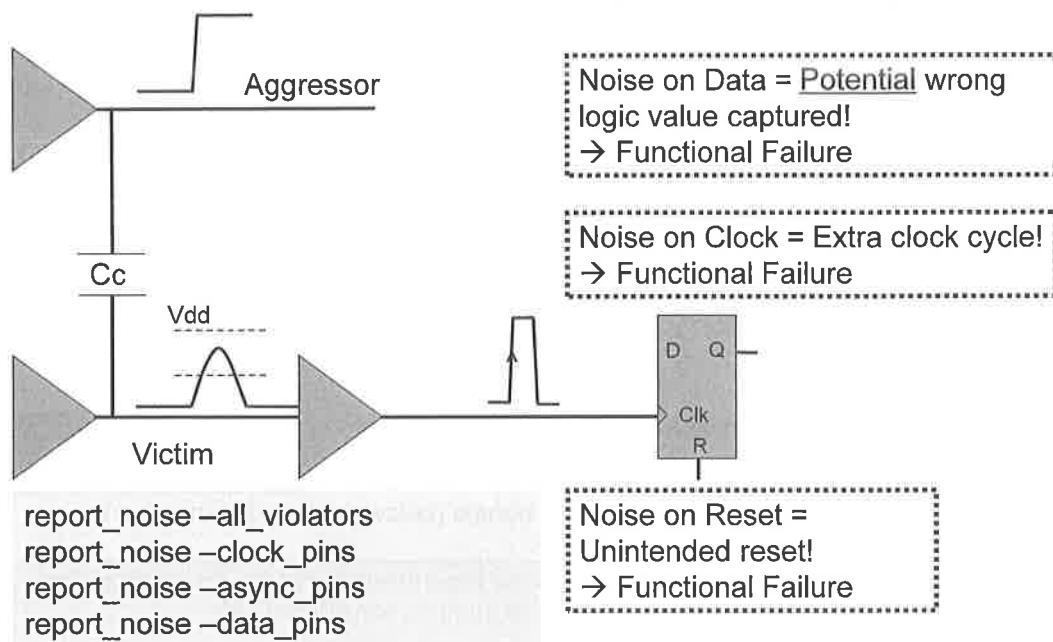
By default, PrimeTime SI only considers “between-the-rails” noise bumps.

Not all libraries will have the problem of over the power rail bump failures. Only libraries that have latches/flops without any buffers to the input of the pass gate. If the input to a nmos pass gate falls below the vt of the device, it can turn on the device and corrupt the internal node.

Most libraries have the sequential devices buffered, so they will be protected against above high or below low noise.

Performing analysis of “beyond-rail” noise bumps is discussed later in the unit.

## Example Functional Failures due to Noise



9-8

A logic failure does not necessarily result in functional failure of the device. PrimeTime SI will not check that the logic failure is actually latched. If the incorrect value becomes correct again before the capture edge occurs, the device will work properly. PrimeTime SI noise analysis is based on the safe, recommended strategy is to fix all logic failures at the source, whether or not they appear to be latched.

If the load pin of the victim net is the data pin of a sequential cell, a special check is performed to find the worst bump height that can occur in the capture region between the setup and hold edges. If no bump can be found in this capture region, the total noise bump height is set to zero.

For more information on this feature, please refer to SolvNet article:  
“How are the aggressor bumps summed in report\_noise\_calculation”.

**Doc Id: 014387 Last Modified: 2/19/2010**

## PrimeTime SI Noise Terminology

These words and concepts  
will show up in PrimeTime  
reports.

Height and width of a  
“noise bump”

Steady-state I-V  
characteristics

Noise Immunity Curve

Noise Margin

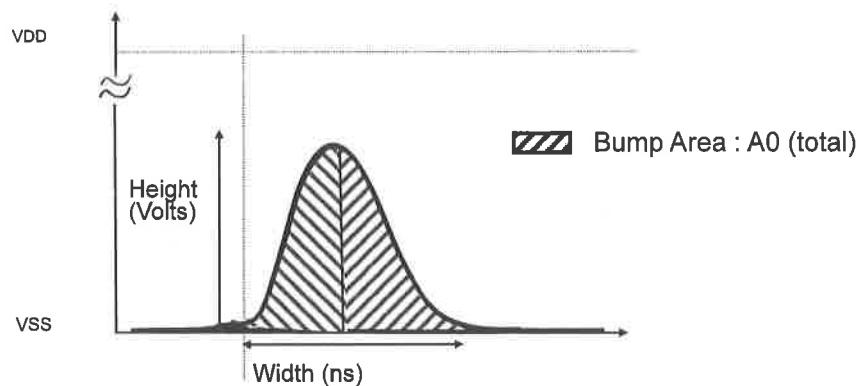
Noise “slack”

“Above low” and “below high”

9-9



## Noise Bump Modeling – Height and Width



### ■ Two important numbers characterize a noise bump

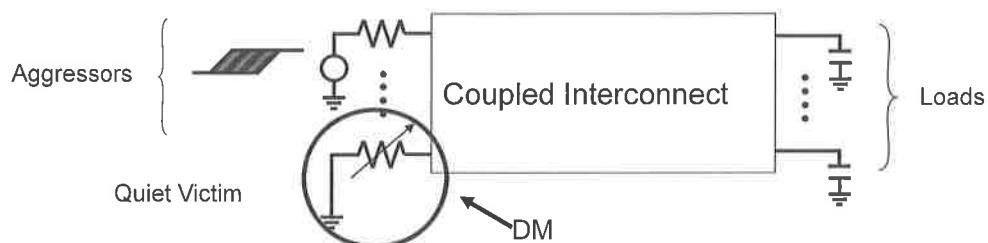
- Height (library voltage units)
- Width (library time units)
  - ◆ Derived from the height and area of the noise bump
  - ◆  $\text{Width} = (2 * A_0) / \text{Height}$

9-10

A third number also used to characterize a noise bump is called time-peak-ratio, which is the time-to-peak divided by the total bump width. A time-peak-ratio of 0.5 indicates a symmetrical bump with equal rising and falling times.

## The Size of a Noise Bump is Affected By ...

- Coupling capacitance
- Drive characteristics of all “active” aggressors
  - Aggressors may be screened if their timing windows do not overlap, their bumps are too small or due to logical correlation
- Steady-state I-V characteristics of the victim driver output
  - For more accurate results, noise response characterization should be specified in the library



9-11

In the absence of library-specified or command-specified I-V characteristics, PrimeTime uses an estimated linear resistance calculated from the output delay, output slew, and NMOS/PMOS transistor threshold voltages. The output delay and slew are specified in the library. By default, PrimeTime assumes an NMOS and PMOS threshold voltage of 0.2 times the rail-to-rail voltage. It is recommended to use `set_noise_lib_pin` where noise library is missing. Sets an equivalent noise library pin for a driver or load.

The screening of aggressor nets due to bump size and logical correlation are controlled by the same variables as that for crosstalk delay calculations.

`si_filter_accum_aggr_noise_peak_ratio` (default = 0.03)  
`si_filter_per_aggr_noise_peak_ratio` (default = 0.01)  
`si_analysis_logical_correlation_mode` (default = true)

See the following SolvNet articles:

How does composite aggressor filtering work in PrimeTime SI?

**Doc Id:** 030997 <https://solvnet.synopsys.com/retrieve/030997.html>

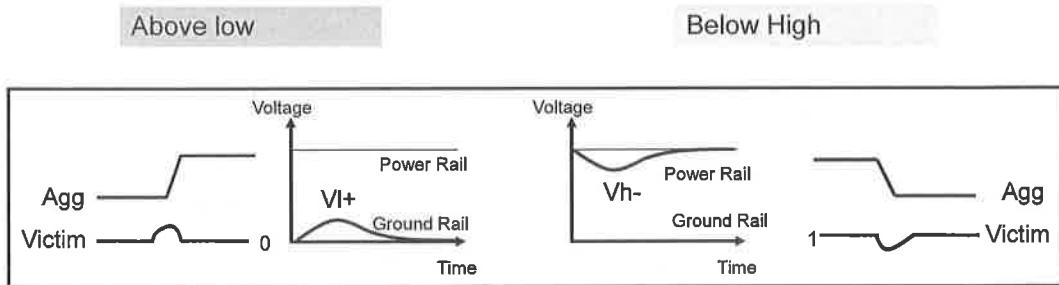
Logical Correlation Analysis on Victim and Aggressor Pairs With the Same Sense

**Doc Id:** 1464951 <https://solvnet.synopsys.com/retrieve/1464951.html>

Understanding `set_noise_parameters` Interaction with PrimeTime Noise Variables

**Doc Id:** 028853 <https://solvnet.synopsys.com/retrieve/028853.html>

## Noise Bumps Calculated in Two Regions (By Default)



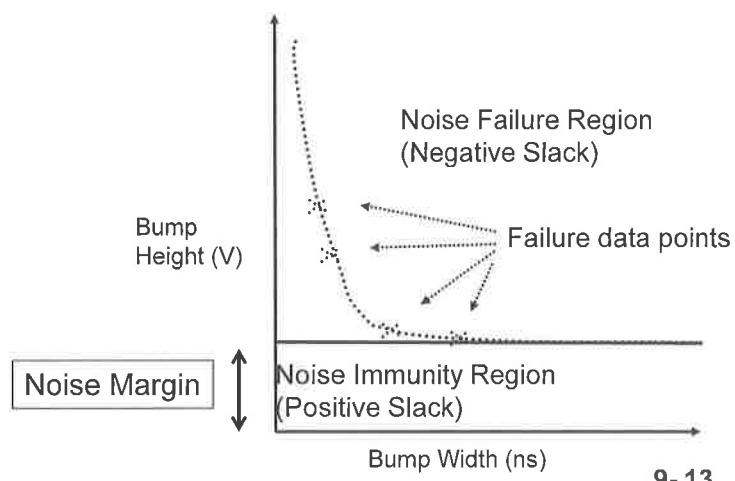
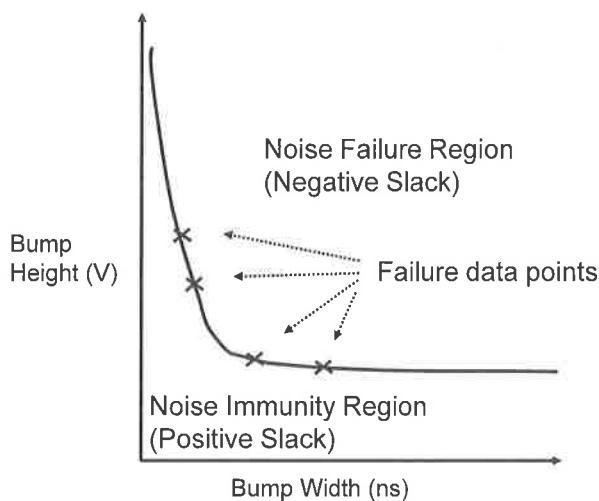
```
pt_shell> report_noise -above -low
pt_shell> report_noise -below -high
```

9-12

## Noise Immunity Curve vs. Noise Margin Constraint

Noise Immunity Curve describes the amount of noise that can be tolerated on the input causing a logic failure at the output.

Noise Margin is a simpler, faster, but, more conservative method to describe the amount of noise that can be tolerated on the input causing a logic failure at the output.



9-13

PrimeTime SI uses the library-specified or command-specified (`set_noise_immunity_curve`) noise immunity at cell inputs to determine whether noise failures occur and the amount of noise slack at each cell input. The user specified noise immunity using the command will override that specified in the library. The `report_noise_calculation` command shows whether noise immunity information was taken from the library or annotated by this command.

For high, narrow noise bumps, using height-only noise margins is pessimistic because it treats some bumps as noise failures that would otherwise pass with the immunity curve model. However, for wide noise bumps, using noise margins gives the same results as using noise immunity curves.

PrimeTime SI uses the library-specified DC noise margins or command-specified (`set_noise_margin`) noise margin at cell inputs to determine whether noise failures occur and the amount of noise slack at each cell input.

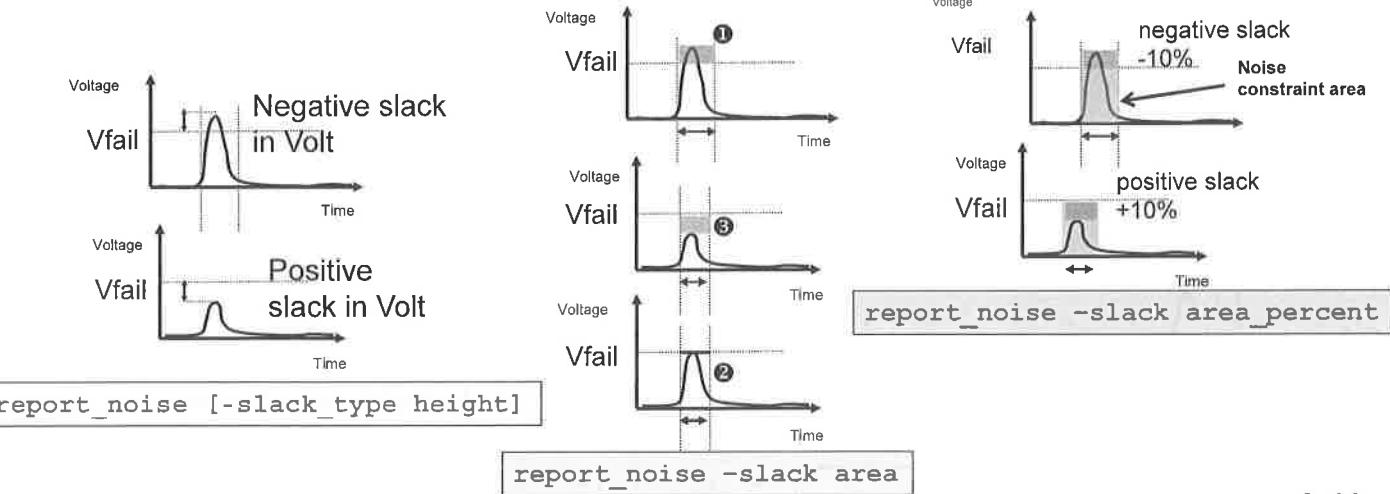
In case of conflict between different methods, PrimeTime SI uses noise immunity specifications in the following precedence order:

- PrimeTime SI command-specified noise immunity curves (`set_noise_immunity_curve`)
- PrimeTime SI command-specified bump height noise margins (`set_noise_margin`)
- Library-specified per-arc noise immunity tables
- Library-specified per-pin noise immunity curves
- Library-specified DC noise margins (VOL, VOH, VIL, VIH)

# Reporting Noise Slack Using 3 Different Slack Types

## ■ Noise slack can be reported using 3 different methods

- The default is height slack



9-14

By default, report\_noise reports the slack in height (voltage) units. Slack is equal to the voltage margin (height of the curve (NIC) above the data point)

With the report\_noise -slack\_type area, slack is equal to the voltage margin (height of the curve above/below the allowed height) multiplied by the noise bump width.

Noise slack is an area value and the unit is, for example, mV·ns

With the report\_noise -slack\_type area\_percent, slack is given in percentage – the area slack divided by the noise constraint area.

The noise constraint area is the bump width multiplied by the allowed height → This is more meaningful when gauging “how far we are from the constraint”

## Interpreting report\_noise (1/2)



```
pt_shell> report_noise -slack_type area_percent

slack type: area_percent

noise_region: above_low
pin name (net name) width height slack

U1/A1 (n32) 1420.099365 0.289573 46.59%

noise_region: below_high
pin name (net name) width height slack

U16/A2 (n50) 5191.086914 0.360771 42.73%
```

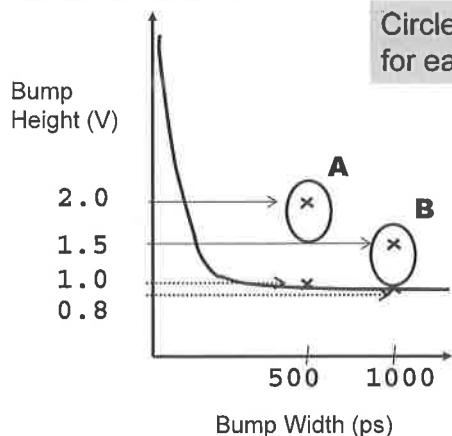
- (1) Is there a noise violation?
- (2) Describe what is meant by above\_low?
- (3) What are the units for width and for height?
- (4) Describe what is an "area\_percent" slack type?

9-15

- (1) No, a violation would be a negative slack
- (2) A violation when the aggressor is switching from 0 -> 1 and the victim is steady state at 0.
- (3) Width is in library time units and height is in library voltage units
- (4) Constraint Height - Height of noise bump / Constraint Height

Answers

## Interpreting report\_noise (2/2)



Circled below are the “worst” negative slack for each slack

|   | Area Slack | Percent Area Slack | Height Slack |
|---|------------|--------------------|--------------|
| A | -500       | (-100%)            | -1           |
| B | -700       | ~-90%              | -0.7         |

- (1) Describe which slack type(s) give a better idea of “how far off” the noise bump is from the constraint?
- (2) Will the three methods of describing slack always report the same “worst” nets in the same order?

9-16

- (1) This is open to discussion – Percentage is the best indication of how far off the curve the slack is. Height doesn’t seem too bad either. Area is a really weird number – and gives fairly different results from the other two.
- (2) No! It’s not simply three different ways to report the same thing. You will probably get different results in a different order.

Answer

## Special Cases: No Slack and Slack Equal to “Infinity”

- If no constraint is available, slack is not calculated

```
pt_shell> report_noise [all_outputs]
noise_region: above_low
pin name (net name) width height slack

V4Y[25] (V4Y[25]) 23792.4238 0.0891 (circled)
```

- If the noise calculated for a victim is zero, slack is infinity

```
pt_shell> report_noise -clock_pins
noise_region: above_low
pin name (net name) width height slack

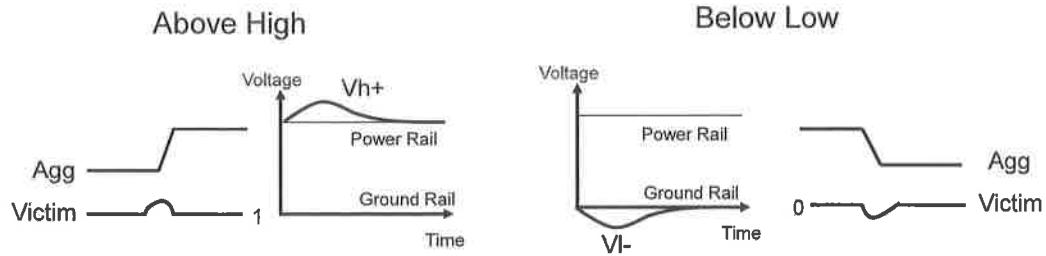
LOCKUP/EN (n2) 0.0000 0.0000 INFINITY (circled)
```

9-17

A user can supply noise constraints on any port, pin or library pin using the commands **set\_noise\_margin** or **set\_noise\_immunity\_curve**.

```
pt_shell> set_noise_margin 0.5 [all_outputs]
```

## Beyond Rail Analysis



- Bumps below low and above high are also important because they can forward-bias pass gates at the inputs of flip-flops and latches, allowing incorrect values to be latched
  - Off by default

```
Set before the first noise update
set_noise_parameters -include_beyond_rails
report_noise
```

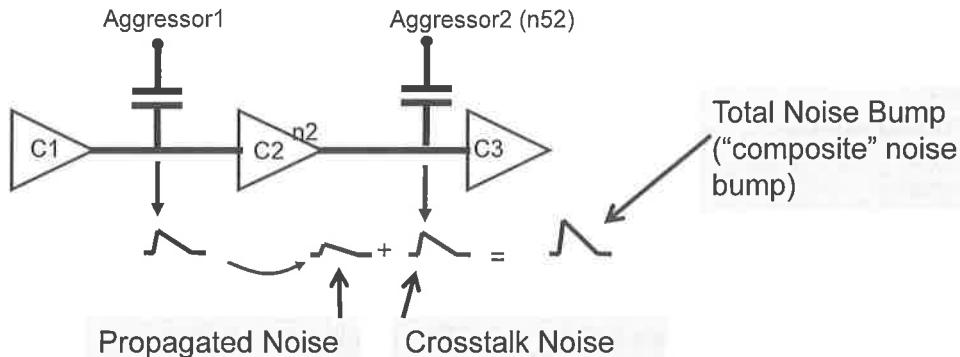
9-18

# Noise Propagation

- A noise bump at a cell input may cause a noise bump at the cell output

- This effect is called noise propagation (off by default)
- For more accurate results, propagation characteristics should be specified in the library

f(noise width, noise height, output loading)



9-19

## User-Defined Noise

In some cases, propagated noise exists in the design but cannot be calculated. For example, the library might not include noise propagation characteristics for a cell, perhaps because the cell is an extracted timing model or a black-box model. For these cases, you can specify a noise bump explicitly on any pin or port in the design by using a PrimeTime SI command, `set_input_noise`. This is called user-defined noise. User-defined noise can either override or add to any propagated noise for the applicable pin or port.

## Example Noise Report

```
Set before first noise update
set_noise_parameters -enable_propagation
report_noise -verbose
```

```
slack type: area
```

```
noise_region: above_low
pin name (net name) width height slack
```

```

```

```
C3/A (n2)
```

```
Aggressors:
```

```
n52
```

```
Propagated:
```

```
C2/Z
```

```
Total:
```

```
2.089932 0.042543
```

```
2.089932 0.042543
```

Offer one reason for  
no reported  
propagated noise?

Circle the following in the report above.

The victim net

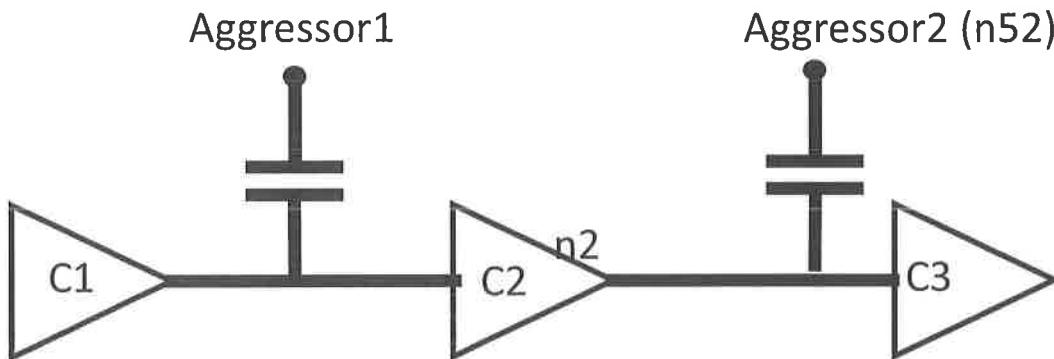
The drive pin on the victim net

The effective aggressor net(s)

The propagated noise

9-20

The command **report\_noise** lists “effective” aggressor nets (i.e. it does not limit the report to “active” aggressors only). This is done because in some cases it may not be sufficient to fix active aggressors only.



Answers

- (1) The victim net is n2
- (2) The one effective aggressor is n52
- (3) The drive pin in C2/Z
- (4) The propagated noise is zero. One reason is the library may not be characterized or contain this information. And of course – the feature must be turned on (as it is in this example).

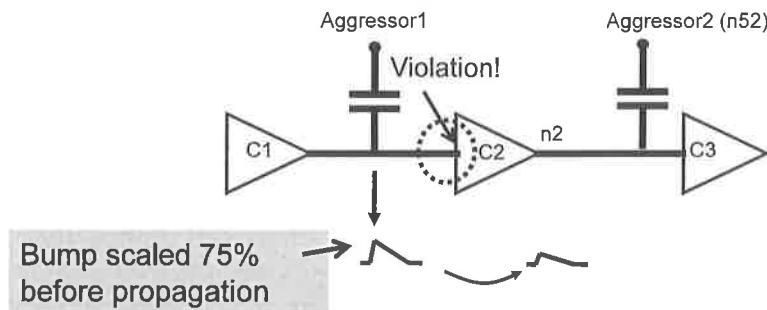
# Propagation of a Noise Violation

## ■ If the previous noise bump is failing

- In report\_at\_source noise analysis mode [To be soon discussed]
- The failing bump's height is scaled down to its constraint height multiplied by 75%
- This is propagated through the cell using the cell's noise propagation characteristics into the following stage

## ■ Allows noise analysis to continue for fanout nets

- Assumption - repair process fixes violation with a 25% margin



9-21

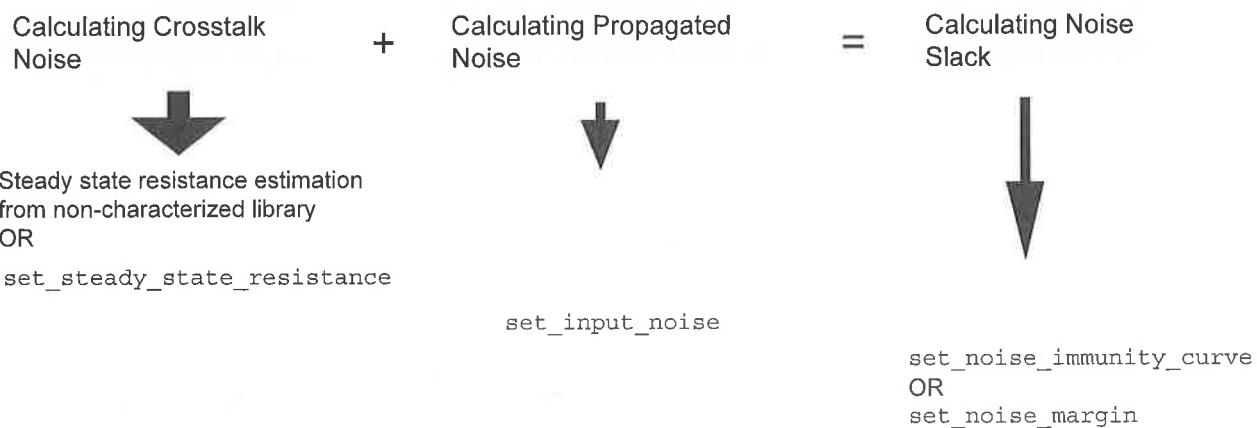
The scaling of noise violations during propagation is 75% by default and is controlled by the application variable `si_noise_limit_propagation_ratio` when in `report_at_source` noise analysis mode.

For more information, refer to SolvNet article:

“What is the `si_noise_limit_propagation_ratio` variable used for?”.

**Doc Id:** 013301 **Last Modified:** 03/20/2014

# Accounting For Incomplete Library



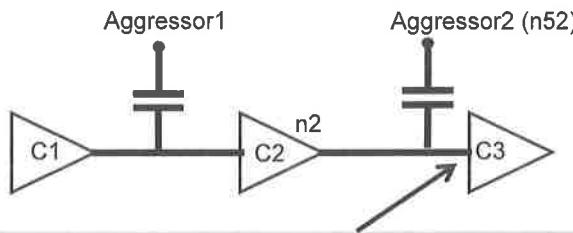
See SolvNet article in student notes

9-22

In addition to using noise characteristics from libraries, PrimeTime allows you to manually set or alter noise characteristics in blocks or cells in a design. Manual noise characteristic setting is beneficial in cases when you want to set noise behavior that is different than what is already defined in the library or when the particular blocks or cells do not have defined noise characteristics. For example, these methods can be used on blocks (i.e. custom transistor blocks that have not been characterized for noise, memories or other black boxes in your design) as well as on library cells with missing or incomplete noise information.

For more information, please refer to SolvNet article  
“PrimeTime SI Noise Analysis With Missing or Incomplete Noise Library Data”.  
**Doc Id:** 013888 **Last Modified:** 01/18/2010

## Example – Injecting Noise



```
pt_shell> set_input_noise -height 0.1 -width 1 C3/A
```

```
pt_shell> report_noise -verbose C3/A
```

```
slack type: area
```

```
noise_region: above_low
```

| pin name (net name) | width | height | slack |
|---------------------|-------|--------|-------|
|---------------------|-------|--------|-------|

```
C3/A (n2)
```

```
Aggressors:
```

```
n52
```

```
Propagated:
```

```
C2/Z
```

```
Total:
```



```
→ 2.089932 0.042543
```

```
→ 1.000000 0.100000
```

```
→ 1.325298 0.142543 3.667597
```

9-23

### set\_input\_noise

Sets a noise bump for a pin or port.

This example specifies a noise bump height of 0.58, width of 1.70 for pin B of cell U1 in the current design.

```
pt_shell> set_input_noise -above -low -width 1.70 -height 0.58 [get_pins U1/B]
```

## More Details from Noise Calculation Report

| pt_shell> report_noise_calculation from C2/Z -to C3/A |                 |                 |                 |           |            |
|-------------------------------------------------------|-----------------|-----------------|-----------------|-----------|------------|
|                                                       | Height          | Width           | Area            | Aggressor | Attributes |
| <b>Aggressors:</b>                                    |                 |                 |                 |           |            |
| n52                                                   | 0.042543        | 2.089932        | 0.044456        |           | A D        |
| <b>Propagated:</b>                                    |                 |                 |                 |           |            |
| C2/Z                                                  | 0.100000        | 1.000000        | 0.050000        |           |            |
| <b>Total:</b>                                         | <b>0.142543</b> | <b>1.325298</b> | <b>0.094456</b> |           |            |

Derived Width =  $2 * \text{Area} / \text{Height}$

Area =  $\text{Width} * \text{Height} / 2$

9-24

The aggressor attributes for the above report are as follows.

Attributes:

- A - aggressor is active
- I - aggressor has infinite window
- S - aggressor is screened due to small bump height
- L - aggressor is screened due to logical correlation
- E - aggressor is screened due to user exclusion
- D - aggressor is analyzed with detailed engine

## Adding Margin and Excluding Nets for Noise Analysis

- Allows you to add margin on the calculated noise bump size and slack

```
set_noise_derate
```

- Exclude victim and/or aggressor nets from noise analysis
- Set specific aggressor nets as infinite arrival windows

```
set_si_noise_analysis \
 -exclude my_net \
 -ignore_arrival my_io_nets
```

9-25

```
pt_shell> set_noise_derate -help
```

Usage:

```
set_noise_derate # Set parameters for noise derate
 [-above] (Set the derate above the rails)
 [-below] (Set the derate below the rails)
 [-low] (Set the derate for the low rail)
 [-high] (Set the derate for the high rail)
 [-height_offset float] (Set height offset value)
 [-height_factor float] (Set height scale factor value)
 [-width_factor float] (Set width scale factor value)
 [object_list] (List of input pins or output ports)
```

```
pt_shell> set_si_noise_analysis -help
```

Usage:

```
set_si_noise_analysis # Set noise coupling information on nets
 -ignore_arrival inets (Set infinite window for nets)
 -exclude (Exclude nets for crosstalk noise)
 [-victims vnets] (List of nets as victim)
 [-aggressors anets] (List of nets as aggressor)
 [-above] (For above rail analysis)
 [-below] (For below rail analysis)
 [-high] (For victim high rail)
 [-low] (For victim low rail)
```

# Detecting Double Switching Due to Crosstalk

## ■ Default noise detection

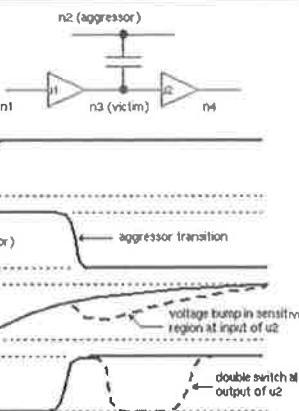
- Detects noise on **static** victim nets

## ■ Double Switching Detection

- Detects noise on **dynamic** victim nets, which can cause

Chip  
killers

- ◆ False clocking
- ◆ Double clocking
- ◆ Combinational glitch propagation



## ■ Requires CCS noise libraries

```
set si_xtalk_double_switching_mode
clock_network
update_timing
report_si_double_switching
```

9-26

`si_xtalk_double_switching_mode` can be set to `full_design` or to `clock_network`

Double switching is not detected by `update_noise`; instead, it requires `update_timing`

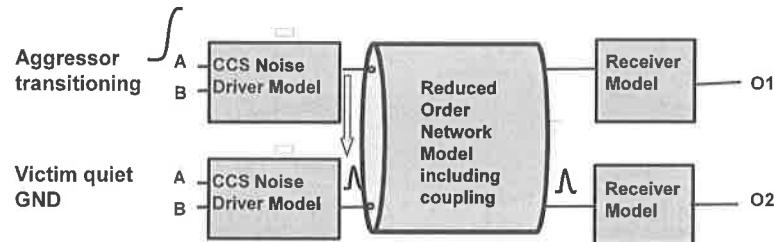
## Report\_si\_double\_switching

\*\*\*\*\*

| Victim Net    | Switching Direction | Actual Bump Height | Required Bump Height | Double Switching Slack |
|---------------|---------------------|--------------------|----------------------|------------------------|
| ASTfhNet1     | max_rise            | 0.413813           | 0.200000             | -0.213813 (VIOLATING)  |
| fdm1_macl[17] | max_rise            | 0.376398           | 0.218734             | -0.157664 (VIOLATING)  |

## Introducing CCS Noise Model in Noise Analysis

- The victim's driver and aggressor's driver are replaced by the CCS Noise driver models
- The parasitics network is replaced by a Reduced Order Model.



- Receiver model comes from the Timing library
- Input noise immunity threshold is calculated on the fly.
  - Unlike NLDM Noise library, noise immunity table is not stored in CCS Noise library.

9-27

Noise analysis using CCS model computes exact *glitch waveform* at the load pin.

Noise bump waveform is a non-linear function of injected noise, propagated noise components, not a linear superposition of noise from different aggressors.

The Noise analysis uses both static and dynamic response of the gate in a FAST SPICE-like fashion.

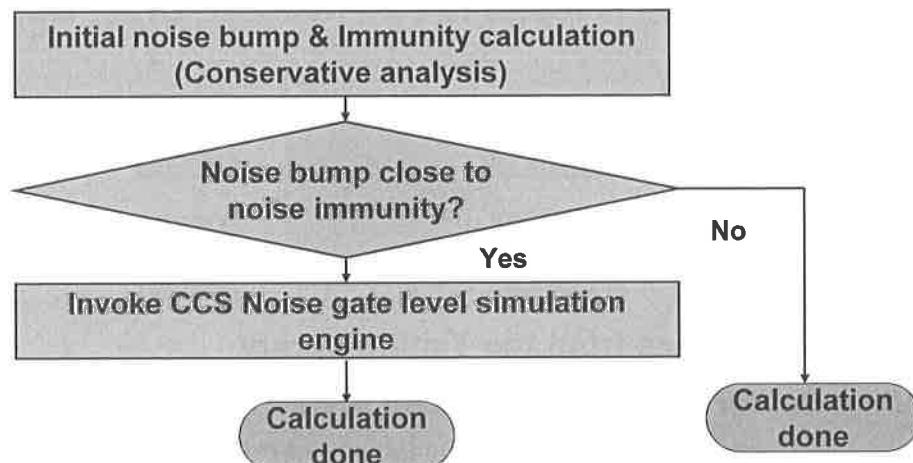
Noise bump is computed at the input of the load. Effect of noise injection, propagation and driver weakening are analyzed concurrently.

Contribution of propagated noise bump to the total noise bump can not be separated.

Total noise bump is not numerical summation of contribution from individual aggressors and propagated noise. Noise bump analysis is done on the entire electrical cluster.

## Noise Bump Calculation with CCS Noise

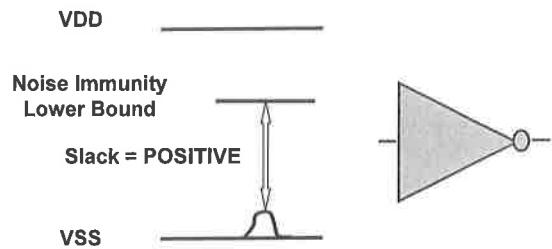
- `update_noise` invokes noise analysis



9- 28

## POSITIVE Slack Reporting

- The calculation engine quickly estimates the noise immunity in a conservative fashion. If this estimate is far above the noise bump, detailed immunity calculation is skipped
- Slack is reported as “POSITIVE”



| pin name (net name) | width    | height   | slack    |
|---------------------|----------|----------|----------|
| <hr/>               |          |          |          |
| buf5/A (I2)         |          |          |          |
| Aggressors:         |          |          |          |
| I3                  | 3.981111 | 0.157197 |          |
| I1                  | 2.282906 | 0.281733 |          |
| Total:              | 3.252171 | 0.387783 | POSITIVE |

9-29

Accurate noise immunity calculation is not required if it is way above the noise bump. This behavior can not be overridden by the user. Because accurate calculation can always be forced by report\_noise\_calculation command, and this does not increase update\_noise command runtime.

## Calculated Slack Reporting

- When the noise bump is close to the estimated immunity, then the detailed immunity calculation is performed
- Actual slack is reported

| pin name (net name) | width    | height   | slack    |
|---------------------|----------|----------|----------|
| <hr/>               |          |          |          |
| buf5/A (I2)         |          |          |          |
| Aggressors:         |          |          |          |
| I3                  | 2.355216 | 0.298266 |          |
| I1                  | 2.351451 | 0.300110 |          |
| Total:              | 2.417497 | 0.607778 | 0.041656 |

Slack  
computed  
(close to 0)

| pin name (net name) | width    | height   | slack     |
|---------------------|----------|----------|-----------|
| <hr/>               |          |          |           |
| buf5/A (I2)         |          |          |           |
| Aggressors:         |          |          |           |
| I3                  | 2.619492 | 0.278514 |           |
| I1                  | 2.966029 | 0.356814 |           |
| Total:              | 2.860501 | 0.651009 | -0.115716 |

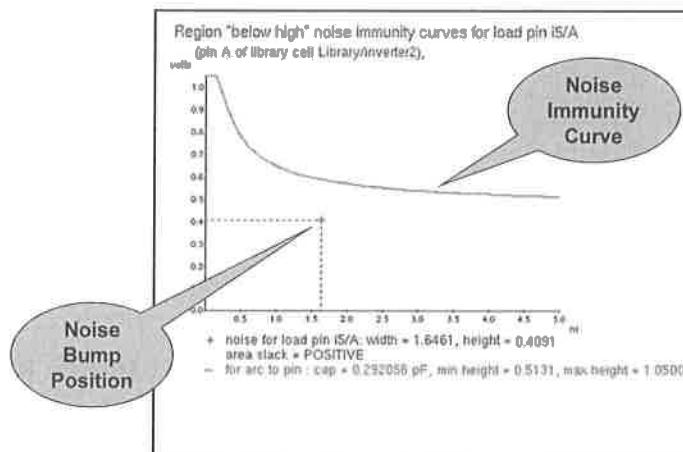
Slack  
computed  
(violating)

9-30

With on-the-fly immunity calculation, any noise waveform can be accurately handled. To save runtime, PT would use detailed immunity calculation only when really required.

## Noise Immunity Curve Display in GUI

- Noise Immunity Curve calculated for an input pin can be displayed in the GUI
- This also displays the position of the noise bump



9-31

## report\_noise\_calculation: Aggressors Section

- G : CCS gate level simulation engine has been used on this significant noise bump as more accuracy is required.

|             | Height   | Width    | Area     | Aggressor Attributes |
|-------------|----------|----------|----------|----------------------|
| <hr/>       |          |          |          |                      |
| Aggressors: |          |          |          |                      |
| I3          | 0.298266 | 2.355216 | 0.351240 | A I D                |
| I1          | 0.300110 | 2.351451 | 0.352847 | A I D                |
| Total:      | 0.607778 | 2.417497 | 0.734651 | G                    |

Estimated contribution from individual aggressors

CCS Gate level simulation used on this bump

- The noise bump height/width reported for individual aggressors is only an estimation.
- To calculate the total noise, these numbers are not used. Instead CCS gate level simulation engine is used, which calculates the total noise bump in a “single shot”.

9-32

The contribution from individual aggressors is only an estimate. CCS noise calculation looks at the electrical system of aggressors and victim as a whole.

## report\_noise\_calculation: Constraint Section

| Constraint type: library CCS noise immunity |          |                       |
|---------------------------------------------|----------|-----------------------|
|                                             | Height   | Area                  |
| Required                                    | 0.625009 | (0.625009 * 2.417497) |
| Actual                                      | 0.607778 | (0.607778 * 2.417497) |
| Slack                                       | 0.017231 | 0.041656              |

CCS noise immunity on input pin  
Area=height\*width

- **Actual:** Total noise bump calculated at the load
- **Required:** The noise immunity bump
- **Slack = Required - Actual**
  - This report shows both Height & Area type slack.

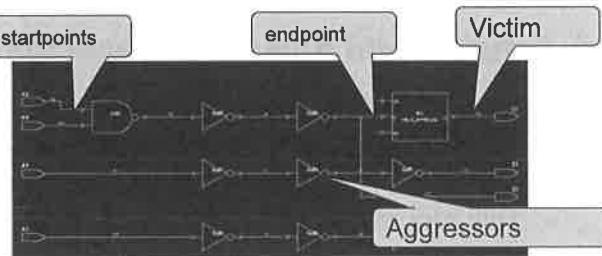
## Two Modes of Reporting with CCS Noise

### ■ report\_at\_source shows violations

- ◆ Report a potential source of violation
- ◆ At every load pin where noise is higher than noise immunity, violation is reported
- ◆ Fix the potential source of violation

### ■ report\_at\_endpoint shows failures

- ◆ Report where a failure happens
- ◆ Noise bump is propagated until it reaches an endpoint.
- ◆ Report the source of the failure
- ◆ Fix only the source of failure



9-34

Noise report depends on the reporting mode chosen

Violation vs failure: report\_at\_source shows violations, report\_at\_endpoint shows failures

A noise startpoint can be any of the following:

- An output pin of a flip-flop
- An output pin of a level-sensitive latch
- An input port
- An output pin of a multi-stage cell (a cell with multiple levels of transistor stages, such as a flip-flop or macro)

A noise endpoint can be any of the following:

- A data, clock, or asynchronous pin of a flip-flop
- An input pin of a level-sensitive latch
- An output port
- An input pin of a multi-stage cell
- Any combinational logic pin where the noise exceeds a threshold of 75 percent of VDD

Startpoint is where the node from where the noise analysis starts, and propagates through cells and terminates at a noise endpoint.

Input of a cell is endpoint generally implies the output is a startpoint.

## Reporting Example : report\_at\_source

```
set_noise_parameters -analysis_mode report_at_source
report_noise -nworst_pins 8
```

| noise_region: below_high |       |        |       |
|--------------------------|-------|--------|-------|
| pin name (net name)      | width | height | slack |
| nd2/B (A4)               | 10.00 | 0.65   | -2.02 |
| buf7/A (J1)              | 7.77  | 0.75   | -1.63 |
| buf5/A (I2)              | 2.33  | 0.30   | -0.47 |
| buf8/A (J3)              | 8.97  | 0.51   | -0.11 |
| buf4/A (I1)              | 3.34  | 0.83   | -0.08 |
| buf6/A (I3)              | 7.55  | 0.41   | 0.21  |
| ff1/D (Z2)               | 1.71  | 0.05   | 0.97  |
| buf9/A (J3)              | 4.86  | 0.07   | 2.14  |

| noise_region: above_low |       |        |       |
|-------------------------|-------|--------|-------|
| pin name (net name)     | width | height | slack |
| sd2/B (A4)              | 10.00 | 0.65   | -1.45 |
| buf4/A (I1)             | 1.79  | 0.69   | 0.28  |
| buf6/A (I3)             | 1.07  | 0.25   | 0.32  |
| buf7/A (J1)             | 5.31  | 0.46   | 0.43  |
| buf9/A (J3)             | 1.01  | 0.04   | 0.61  |
| ff1/D (Z2)              | 5.78  | 0.03   | 2.18  |
| buf8/A (J2)             | 9.35  | 0.18   | 3.05  |
| buf5/A (I2)             | 7.09  | 0.08   | 3.41  |

6 violations

9-35

## Reporting Example : report\_at\_endpoint

```
set_noise_parameters -analysis_mode report_at_endpoint
report_noise -nworst_pins 8
```

```

Report : noise
-nworst_pins 8

analysis mode: report_at_endpoint
slack type: area

noise_region: above_low
pin name (net name) width height slack

Z1 (Z1) 6.16 1.02 -2.96
Z2 (Z2) 6.16 1.02 POSITIVE
Z3 (Z3) 5.59 0.57 POSITIVE
Z4 (Z4) 0.00 0.00

noise_region: below_high
pin name (net name) width height slack

Z1 (Z1) 3.53 0.91 -1.20
Z2 (Z2) 3.53 0.91 POSITIVE
Z3 (Z3) 2.78 0.38 POSITIVE
Z4 (Z4) 0.00 0.00
```

Failing endpoint

9-36

## Reporting Violation Sources

```
report_noiseViolationSource
```

```

Report : noise violation sources

```

```
analysis mode: report_at_endpoint
slack type : area
```

```
endpoint noise region: above_low
violation source width height slack endpoint
```

```

nd2/B 10.00 0.65 -2.02 ff1/D
```

```
endpoint noise region: below_high
```

```
violation source width height slack endpoint
```

```

nd2/B 10.00 0.65 -1.45 ff1/D
```

Source of  
the problem

failing  
endpoint

```
getNoiseViolationSource
{ "nd2/B" }
```

9-37

report\_noiseViolationSource reports the source (the first load input to have the negative slack)  
getNoiseViolationSource returns a collection of the source pins

## Review Unit Objectives



After completing this unit, you should be able to:

- Modify an existing PrimeTime run script to perform crosstalk noise analysis
- Generate and interpret the key reports in the shell as well as the GUI to identify violations due to crosstalk noise
- Describe the advantages of using CCS Noise library
- Differentiate *report\_at\_source* vs. *report\_at\_endpoint* when using CCS Noise library

9-38

## Lab 9: Perform Crosstalk Noise Analysis



45 minutes

Execute a run script for Crosstalk noise analysis

Invoke the shell and generate reports

Invoke the GUI and generate reports

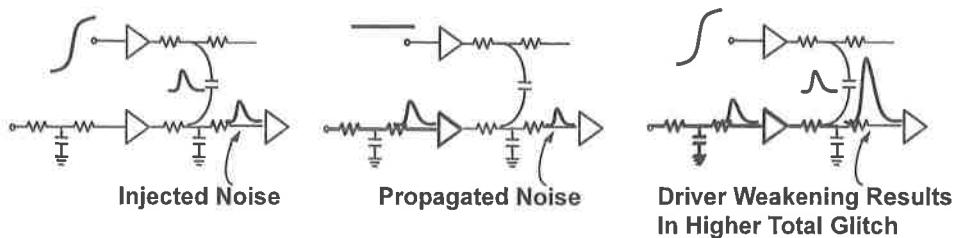
9-39

## **Appendix**

### **Additional Information on CCS Noise**

**9-40**

## Noise Analysis: Driver Weakening



- When injected noise is combined with propagated noise, a nonlinear effect can be observed that “weakens” the victim driver.
- The combined effects result in a higher noise glitch.
- More predominant in 65-nm and below technologies.
- CCS Noise model handles driver weakening effect

9-41

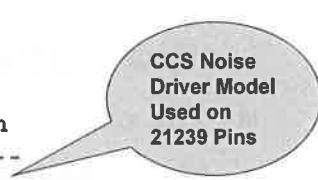
Noise bump waveform is a non-linear function of injected noise, propagated noise components, not a linear superposition of noise from different aggressors.

## CCS Noise Attributes on the Library Cell

- Boolean attributes indicating the presence of CCS Noise data on the pin or timing arc of a library cell.
  - has\_ccs\_noise\_below\_high
  - has\_ccs\_noise\_above\_low
  - has\_ccs\_noise\_above\_high
  - has\_ccs\_noise\_below\_low
- check\_noise command can be used to verify whether CCS Noise data is present on all the pins in the design.

```
check_noise -include noise_driver
Noise driver check:
Noise driver type above_low below_high

CCS noise 21239 21239
library IV curve 0 0
library resistance 0 0
none 13 13
```



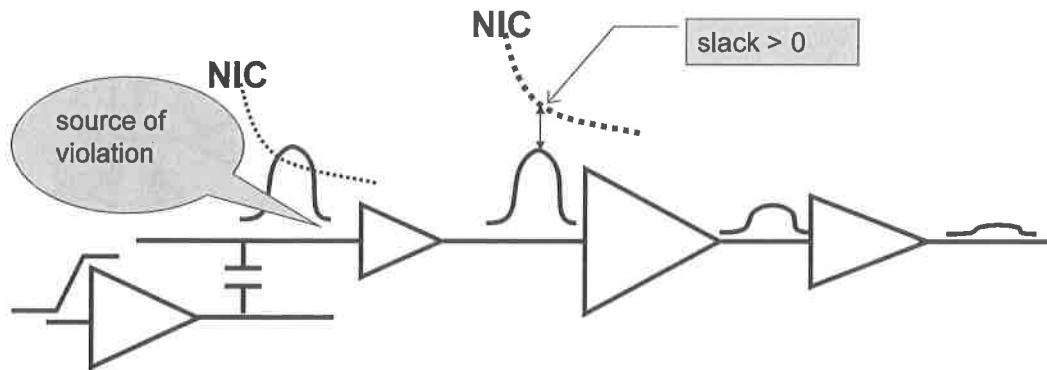
9-42

These attributes check for the presence of CCS Noise information on the pin or timing arc of the library cell.

check\_noise may be preferred over querying attributes if user cares about what is used in the design as opposed to what is in the library.

## Example: report\_at\_source

- At every load pin where noise is higher than noise immunity, violation is reported
  - This may attenuate in the next stages and may not cause functional failure.
- BUT...
  - Noise bumps can increase dynamic power consumption.
  - They can cause timing failures on the paths they are coupled to.



9-43

NIC: Noise Immunity Curve

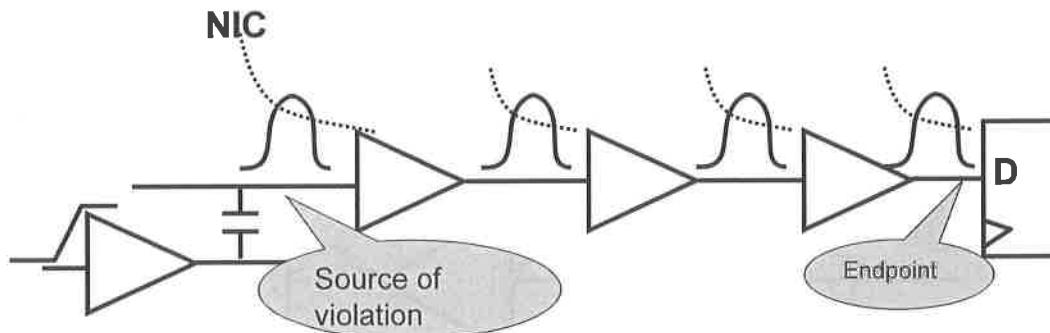
Why use report\_at\_source?

For clean design. Power

Can cause timing failures on the coupled path. They may not have overlapping timing windows, and this may not be caught during timing analysis.

## Example: report\_at\_endpoint

- Noise bump is propagated until it reaches an endpoint.
  - If noise at the endpoint is greater than noise immunity,
    - ◆ Report the endpoint as a violation.
    - ◆ Report the source of violation: First load pin with negative noise slack along the propagation path.
- Indicates possible functional failure i.e incorrect logic value clocked.



9-44

Why use report\_at\_endpoint?

Fix if the concern is latching of incorrect value by sequential elements.

# Agenda

DAY

3

8 Signal Integrity: Crosstalk Delay Analysis



9 Signal Integrity: Crosstalk Noise Analysis



10 Correlation: POCV and AWP Analysis

11 Timing Closure: ECO/What If Analysis

12 Large Data: DMSA and Hyperscale Analysis

13 Conclusion

10-1

## Unit Objectives



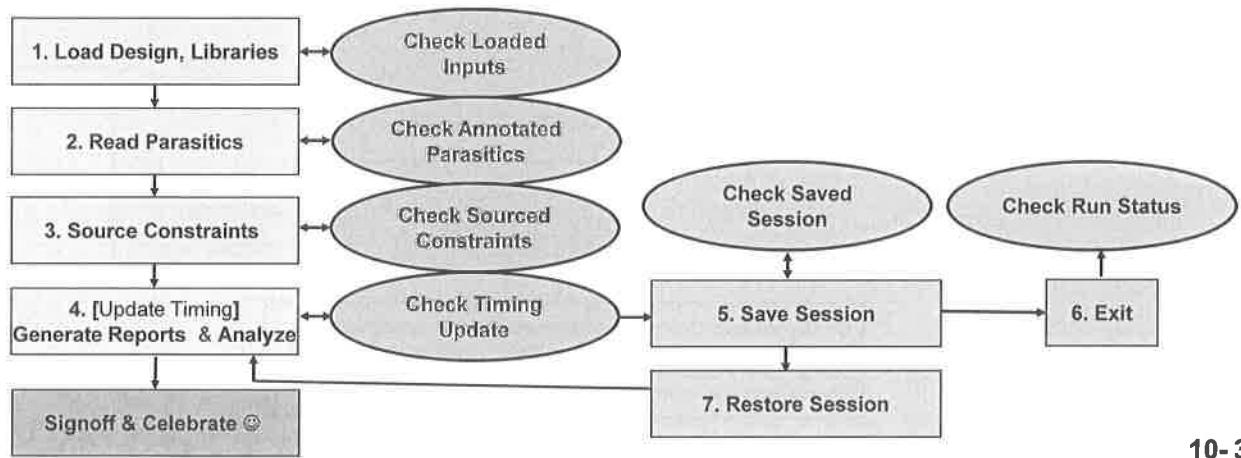
After completing this unit, you should be able to:

- Account for on-chip delay variations using Parametric OCV (POCV) analysis with
  - Side File
  - Data in Liberty Variation Format (LVF)
- Enable Advanced Waveform Propagation (AWP) to address waveform distortions due to
  - Long Tail effect and
  - Strong backward Miller effect

10-2

## Timing Analysis Flow in PrimeTime -- Analysis

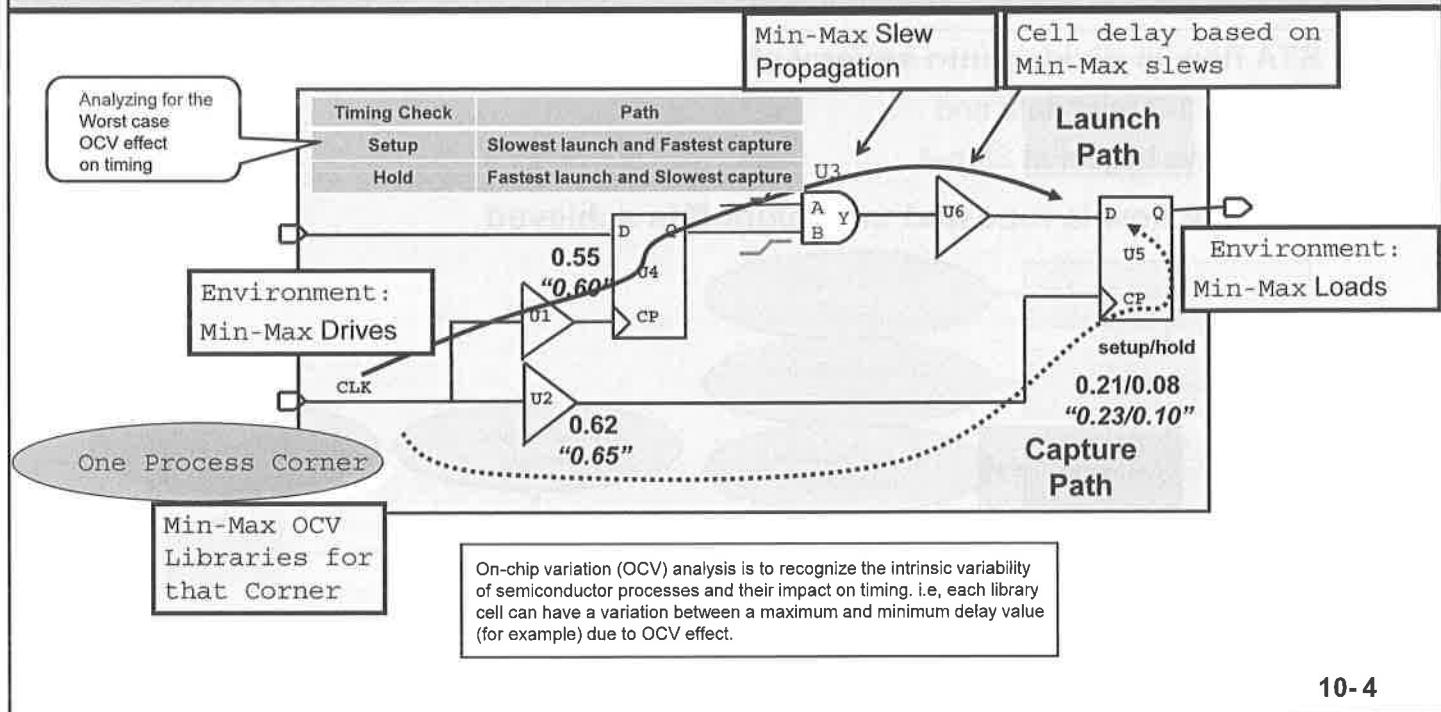
- STA flow is divided into several steps
  - Steps 1-3 read data and
  - Analysis begins at Step-4
- The STA flow is repeated until signoff is achieved



10-3

STA: Static Timing Analysis

# Traditional Accounting for On Chip Variation (OCV) Effects



10-4

On-chip variation (OCV) is a recognition of the intrinsic variability of semiconductor processes and their impact on timing.

Setup analysis is done using slowest launch and fastest capture  
Hold analysis is done using fastest launch and slowest capture

$$\text{Launch Path} = U1 + U4 + U3 + U6$$

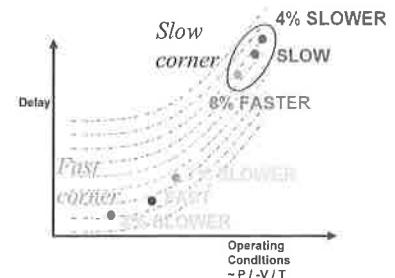
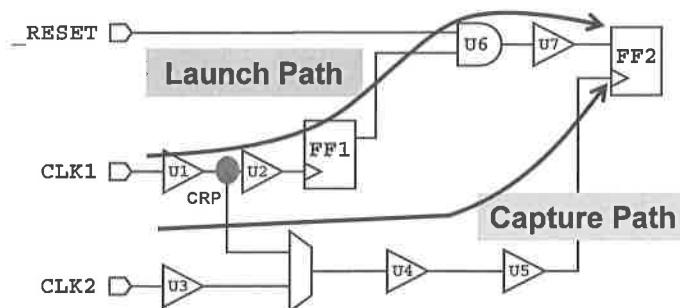
$$\text{Capture Path} = U2 + U5 \text{ _Setup/Hold}$$

If you provide library data for two corners, OCV will use both, increasing the difference between launch and capture path delays, causing failures that would never occur in real life on a single chip – so, with OCV, you should provide library data for only one corner.

## Example: OCV Analysis with Timing Derate

```
In the SLOW PVT corner:
set_timing_derate -early 0.92
set_timing_derate -late 1.04
```

Models an 8% speed-up and a 4% slow-down from the SLOW PVT corner



| Setup                                                   | Hold                                                   |
|---------------------------------------------------------|--------------------------------------------------------|
| Launch uses LATE derate:<br>1.04 x SLOW corner delays   | Launch uses EARLY derate:<br>0.92 x SLOW corner delays |
| Capture uses EARLY derate:<br>0.92 x SLOW corner delays | Capture uses LATE derate:<br>1.04 x SLOW corner delays |

10-5

Most library providers supply you with one library per PVT corner, for example the SLOW corner library and the FAST corner library; They do not provide multiple libraries per corner to model *OCV* differences. It is therefore up to the user to model the effects of on-chip variation using the `set_timing_derate` command. This allows launch and capture path delays to be appropriately derated for setup or hold timing analysis and optimization.

*OCV* analysis with late/early derating is pessimistic, but it is “safe” (you won’t miss any real violations). It is pessimistic because it assumes that ALL the launch cells are in one corner (the late-derated corner, for setup timing), while ALL the capture cells are in a different corner (the early-derated corner, for setup timing). In reality, you will rarely have such a “worst-case” situation. *AOCV*, or *Advanced OCV*, and *POCV* or *Parametric OCV* are other available techniques to reduce this pessimism and achieve more realistic timing analysis and optimization.

## Derating Factor in Timing Reports

```
Startpoint: I_ORCA_TOP/I_BLENDER/s3_op2_reg[18]
(rising edge-triggered flip-flop clocked by SYS_CLK)
Endpoint: I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]
(rising edge-triggered flip-flop clocked by SYS_CLK)
```

```
Path Group: SYS_CLK
```

```
Path Type: max
```

```
Min Clock Path: Derating Factor : 0.900
```

```
...
```

```
report_timing
```

| Point                      | Incr    | Path    |
|----------------------------|---------|---------|
| clock SYS_CLK (rise edge)  | 8.000   | 8.000   |
| clock source latency       | 0.000   | 8.000   |
| sys_clk (in)               | 0.000   | 8.000 r |
| sys_clk_iopad/PAD (pc3d01) | 0.042   | 8.042 r |
| sys_clk_iopad/CIN (pc3d01) | 0.687 & | 8.729 r |

```
report_timing -derate
```

| Point                      | Derate | Incr    | Path    |
|----------------------------|--------|---------|---------|
| clock SYS_CLK (rise edge)  | 8.000  | 8.000   |         |
| clock source latency       | 0.000  | 8.000   |         |
| sys_clk (in)               | 0.000  | 8.000 r |         |
| sys_clk_iopad/PAD (pc3d01) | 0.900  | 0.042   | 8.042 r |
| sys_clk_iopad/CIN (pc3d01) | 0.900  | 0.687 & | 8.729 r |

Capture  
Clock

10-6

Path type is max (a setup check). Therefore, the fastest (min) path is the capture path, shown above. This min path is derated by .9

## CRP in a Timing Report

Defaults

Use -path\_type full\_clock or full\_clock\_expanded to examine the clock tree containing the paths through the last common pin

```
pt_shell> report_timing
Report : timing
-path_type full
-delay_type max
-max_paths 1
-sort_by slack
Design : ORCA

Startpoint: I_ORCA_TOP/I_BLENDER/s3_op2_reg[18]
(rising edge-triggered flip-flop clocked by SYS_CLK)
Endpoint: I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]
(rising edge-triggered flip-flop clocked by SYS_CLK)
Last common pin: I_ORCA_TOP/I_BLENDER/bufbdfG6B2I8/Z
Path Group: SYS_CLK
Path Type: max
Min Clock Paths Derating Factor : 0.900

Point Incr Path

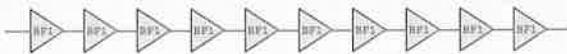
...
I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]/D (sdnrb1) 8.242 6 12.004 f.
data arrival time 12.004
...
clock SYS_CLK (rise edge) 8.000
clock network delay (propagated) 2.736 10.736
clock reconvergence pessimism 0.470 11.206
I_ORCA_TOP/I_BLENDER/s4_op2_reg[31]/CP (sdnrb1) 11.206 r
library setup time -0.189 11.017
data required time 11.017
```

10-7

# Timing Pessimism with OCV Analysis : TFU



The "min-max" OCV delay range of BF1 in the SLOW corner is 90-100 ps:



1. Assume the above OCV is modelled by early/late derating:  
If the launch and capture clock paths each contain ten BF1 buffers,  
setup timing analysis assumes:
  - a. 900 ps for launch path; 1000 ps for capture path
  - b. 1000 ps for launch path; 900 ps for capture path
2. On actual silicon, the more realistic total delay for ten BF1 buffers will be between 900ps and 1000ps. Therefore, OCV timing analysis is:
  - a. Optimistic
  - b. Pessimistic
3. The closer the cells are physically placed to each other:
  - a. The larger their systemic V/T variations can be
  - b. The smaller their systemic V/T variations can be
4. Since process (P) variation is random, its effects are smaller in a path with:
  - a. 30 BF1 cells
  - b. 3 BF1 cells

10-8

"deeper" paths (more cells along a path).

essentially cancelling out the "random" P variations. This makes the path delay more predictable on associated with "random" process (P) variations will be to the "mean" or average path delay.

Question 4: The more cells there are in a timing path, the closer the total path delay component to the cells, which also means less derating.

Question 3: The closer the cells are to each other, the less "systemic" V/T variation exists between cells), and the maximum distance between the cells along the timing path.

Questions 3 and 4 introduce the concepts that AOCM are based on: Different, more realistic derating factors can be applied to cells as a function of the depth of the timing path (number of cells), and the delays will be somewhere between these extremes (Question 2), not as pessimistic as OCV predicts.

Questions 1 and 2 show how OCV is pessimistic by nature: OCV will assume 1000 ps for the launch path and 900 ps for the capture path, for setup timing (Question 1), but in reality the launch/capture path delays will be something between these extremes (Question 2), not as pessimistic as OCV predicts.

This page serves as a "lead-in" or introduction to the next slides about AOCV and POCV.

Answers: 1 b, 2 b, 3 b, 4 a

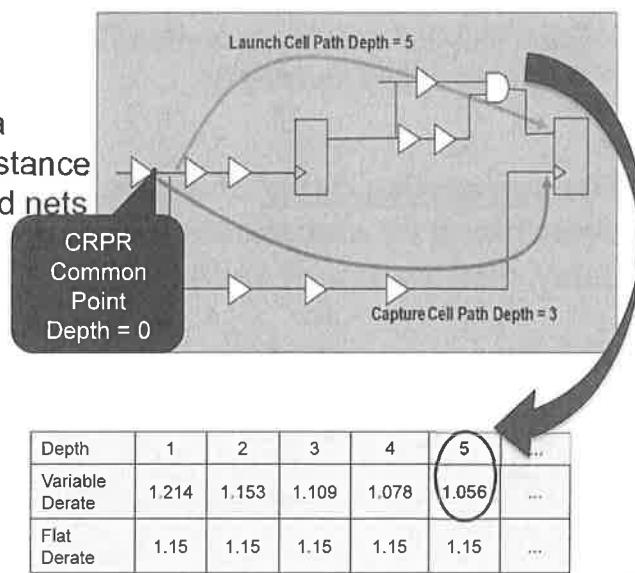
# AOCV Calculates Derating Based on Depth and Distance

**Advanced On-Chip Variation** provides more realistic variable derating by taking path depth and distance into account

- Random P variation modeled as a function of logical path depth
- Systemic V/T variation modeled as a function of the maximum physical distance between related timing path cells and nets

AOCV depth+distance variation table

```
version: 1.0
object_type: lib_cell
delay_type: cell
rf_type: rise
derate_type: late
object_spec: lib/MUX2_1
depth: 1 2 3 4 5 ...
distance: 100 500
table: 1.214 1.153 1.109 1.078 1.056 ...
 1.271 1.201 1.145 1.121 1.100 ...
```



10-9

AOCV is enabled by

`set timing aocvm_enable_analysis true`

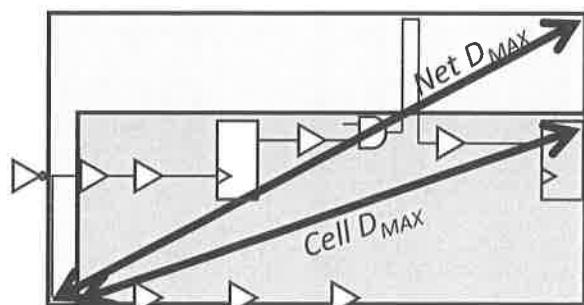
AOCV reduces the level of pessimism in timing analysis by careful modeling of the factors mentioned above: A single derate value is determined and applied to all cells along the launch path, if appropriate. Optionally, a different single derate factor can be applied to all the nets along that same path. Similarly, derate factors for cells and nets are determined and applied to capture paths, if appropriate. The cell and net derate factors vary based on the “depth” of the path (total number of cells along the path), as well as the maximum distance between the cells along the path (see example below). The larger the depth, and/or the smaller the maximum distance, the closer the derate factor should get to 1.0

The example below shows how the maximum distance is calculated for a specific launch/capture sequential timing path:

The maximum cell distance ( $Cell D_{MAX}$ ) is the length of the diagonal of the inner rectangle, which encompasses the placement of all the related launch and capture path cells.

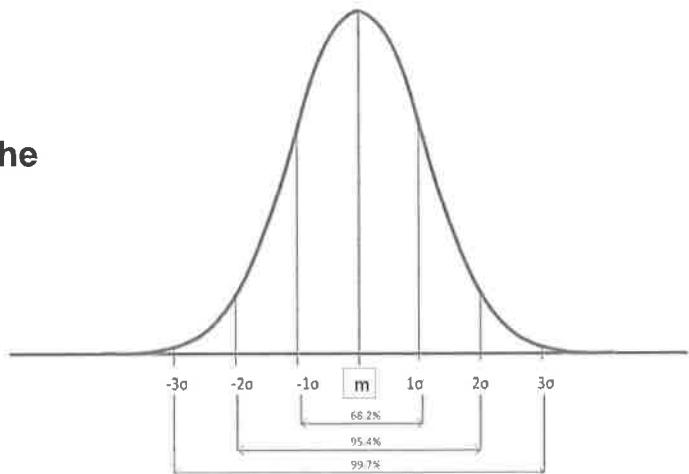
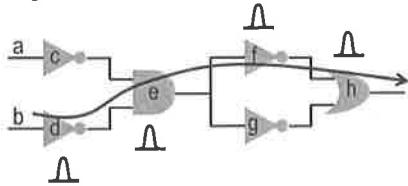
The maximum net distance ( $Net D_{MAX}$ ) is the length of the diagonal of the outer rectangle, which encompasses the routing of all the related launch and capture path nets.

Depth and distance information can be provided in the same table as shown above, or carried in two separate tables, one for depth, one for distance.



# Parametric On-Chip Variation Modeling of Random Variation

- POCV does not rely on path depths to model random variation – instead uses a **Gaussian Distribution** model for individual cell delays
  - Each cell has a nominal or *mean* ( $m$ ) delay and a *standard deviation* ( $\sigma$ ) delay value ( $\sigma$  is also called sensitivity)
- The **cumulative delay** of a path is determined by statistically adding the delay distribution of each stage



- Eliminates GBA-based pessimism due to random variations

10-10

Statistically combining the delay distribution of each stage is more accurate than simply adding the worst-case value from each stage. The resulting delay and slack values are more realistic and less pessimistic than values calculated by simple min-max addition.

## POCV Input Data: Sigma ( $\sigma$ ) - Two Formats

### POCV single coefficient file

```
version : 4.0
ocvm_type : pocvm
object_type: lib_cell
rf_type : rise fall
delay_type : cell
derate_type : late
object_spec: lib28nm/invx*
coefficient: 0.05
```

### Liberty Variation Format (LVF) in standard cell liberty libraries

```
ocv_sigma_cell_rise ("delay_template_4x4") {
 sigma_type : "late";
 index_1("0.0088, 0.0264, 0.0608, 0.1296");
 index_2("0.001, 0.0024, 0.0052, 0.0108");
 values("0.000476, 0.000677, 0.001075, 0.001870",
 "0.000651, 0.000901, 0.001303, 0.002081",
 "0.000840, 0.001166, 0.001714, 0.002558",
 "0.001115, 0.001520, 0.002193, 0.003317");
}
```

- The POCV Coefficient is characterized at a particular input transition and output load per library cell ( $\sigma = C * m$ )
  - No delay variation ( $\sigma$ ) dependency to input transition and output load

- LVF: Cell delay variation or  $\sigma$  (*time units*) is modeled as a function of input transition and output load per timing arc
  - For finer geometries (  $\leq 16\text{nm}$  ) and low voltage, delay variation(s) can strongly depend on the input slew and output load

10-11

SiliconSmart supports POCV LVF characterization. SiliconSmart® is Synopsys' library characterization solution.

The cell delay variation in LVF is modeled as a function of input transition and output load per timing arc. POCV table indexes are recommended (not required) to be the same as NLDM 2-D table lookup.

The POCV coefficient can be reported as follows:

```
pt_shell> report_ocvm -type pocvm [get_cell U1]
```

**POCV coefficient: 0.0500**

| Name | Object | Process | Voltage | Sense | Path  | Delay | Inherited  |
|------|--------|---------|---------|-------|-------|-------|------------|
| U1   | cell   | early   | *       | rise  | clock | cell  | lib1/INV16 |
| U1   | cell   | early   | *       | fall  | clock | cell  | lib1/INV16 |
| U1   | cell   | early   | *       | rise  | data  | cell  | lib1/INV16 |
| U1   | cell   | early   | *       | fall  | data  | cell  | lib1/INV16 |

**POCV coefficient: 0.0500**

| Name | Object | Process | Voltage | Sense | Path  | Delay | Inherited  |
|------|--------|---------|---------|-------|-------|-------|------------|
| U1   | cell   | late    | *       | rise  | clock | cell  | lib1/INV16 |
| U1   | cell   | late    | *       | fall  | clock | cell  | lib1/INV16 |
| U1   | cell   | late    | *       | rise  | data  | cell  | lib1/INV16 |
| U1   | cell   | late    | *       | fall  | data  | cell  | lib1/INV16 |

# POCV Path Calculation Example

Both cells have POCVM coefficient of 0.05 or 5%



Nominal or mean delay ( $m$ ) = 60ps      Nominal or mean delay ( $m$ ) = 80ps  
 Delay sigma ( $\sigma$ ) =  $0.05 \times 60 = 3\text{ps}$       Delay sigma ( $\sigma$ ) =  $0.05 \times 80 = 4\text{ps}$

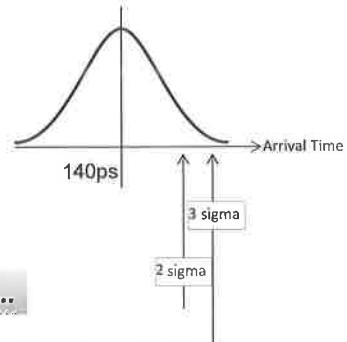
$$\sigma_{path}^2 = sensit\_cell\_A^2 + sensit\_cell\_B^2 + sensit\_cell\_C^2 + \dots$$

$$\text{Path delay mean, } m_{path} = 60 + 80 = 140\text{ps}$$

$$\text{Path delay sigma, } \sigma_{path} = \sqrt{3^2 + 4^2} = 5\text{ps}$$

$$\text{Late path arrival at } 2\sigma = 140 + 2 \times 5 = 150\text{ps}$$

$$\text{Late path arrival at } 3\sigma = 140 + 3 \times 5 = 155\text{ps}$$



Timing analysis uses 3 *sigmas*, by default:

$$\text{Data Arrival or Required}_{late} = m_{path} + 3 * \sigma_{path}$$

$$\text{Data Arrival or Required}_{early} = m_{path} - 3 * \sigma_{path}$$

$$Slack = m_{slack} - 3 * \sigma_{slack}$$

10-12

Path delay *mean*:

$$m_{path} = m_1 + m_2 + m_3 + \dots$$

Path delay *sigma*:

$$\sigma_{path} = \sqrt{\sigma_1^2 + \sigma_2^2 + \sigma_3^2 + \dots}$$

Slack *mean*, setup:

$$m_{slack, setup} = m_{data\ arrival} - m_{data\ required}$$

Slack *mean*, hold:

$$m_{slack, hold} = m_{data\ required} - m_{data\ arrival}$$

$$\sigma_{slack} = \sqrt{\sigma_{data\ arrival}^2 \pm \sigma_{data\ required}^2}$$

Slack *sigma*:

(+) if  $\sigma_{data\ required}$  is a positive value

(-) if  $\sigma_{data\ required}$  is a negative value

See Appendix for more details on slack *sigma* calculation including CRPR.

## POCV Input Data: Distance-based Derating

- Systemic or distance-based (V/T) derating factors can also be taken into account in POCV, and can also be specified in two formats:
  - A side file
  - Liberty Variation Format (LVF) in the logic libraries

POCV distance table (side file)

```
version : 4.0
ocvm_type : pocvm
object_type : design
rf_type : rise fall
delay_type : cell
derate_type : late
object_spec:
distance : 1000 10000 50000 100000 500000
table : 1.004 1.018 1.036 1.055 1.088
```

POCV distance table (LVF)

```
ocv_derate(aocvm_distance_design) {
 ocvm_derate_factors(aocvm_distance_template) {
 rf_type : rise fall ;
 derate_type: late ;
 path_type: clock_and_data;
 index_1("1000, 10000, 50000, 100000, 500000");
 values("1.004, 1.018, 1.036, 1.055, 1.088");
 }
}
```

10-13

The unit of distance is defined by the technology file: If unitLengthName = micron and lengthPrecision = 1000, the unit of length is 0.001 microns, or 1nm.

Timing analysis will interpolate derating values for distances in between the listed distances. If the distance is larger than the largest listed distance in the table, the last derate value is used (no extrapolation is done).

Using the example side file table above: If the distance of a cell is 30,000 nm, its derate value will be 1.027 (mid-way between 1.018 and 1.036). If the distance is 500,000 or larger, the derate value will be 1.088.

# POCV Timing Report

| Point                         | Corner Column<br>Mean +/- 3 * Sensit |      |        | Cumulative Mean & Sensit Column |        |         |
|-------------------------------|--------------------------------------|------|--------|---------------------------------|--------|---------|
|                               | Mean                                 | Incr | Sensit | Corner                          | Value  | Path    |
| clock clk (rise edge)         | 0.00                                 | 0.00 | 0.00   | 0.00                            | 0.00   | 0.00    |
| clock network delay (ideal)   | 0.00                                 | 0.00 | 0.00   | 0.00                            | 0.00   | 0.00    |
| input external delay          | 0.00                                 | 0.00 | 0.00   | 0.00                            | 0.00   | 0.00    |
| in1 (in)                      | 0.00                                 | 0.00 | 0.00   | 0.00                            | 0.00   | 0.00    |
| b1/Z (INV)                    | 1.00                                 | 3.00 | 0.00   | 10.00                           | 10.00  | 10.00 r |
| b3/Z (INV)                    | 1.00                                 | 2.00 | 0.00   | 7.00                            | 2.82   | 12.82 r |
| out1 (out)                    | 0.00                                 | 0.00 | 0.00   | 0.00                            | 0.00   | 12.82 r |
| data arrival time             | 0.00                                 | 0.00 | 0.00   | 0.00                            | 0.00   | 12.82   |
| clock clk (rise edge)         | 0.00                                 | 0.00 | 0.00   | 2.00                            | 2.00   | 2.00    |
| clock network delay (ideal)   | 0.00                                 | 0.00 | 0.00   | 0.00                            | 0.00   | 2.00    |
| clock reconvergence pessimism | 0.00                                 | 0.00 | 0.08   | 0.00                            | 0.00   | 2.00    |
| clock uncertainty             | 0.00                                 | 0.00 | 0.00   | -0.20                           | 0.00   | 1.80    |
| output external delay         | 0.00                                 | 0.00 | 0.00   | -10.00                          | -8.20  | -8.20   |
| data required time            | 0.00                                 | 0.00 | 0.00   | 0.00                            | -8.20  | -8.20   |
| data required time            | 0.00                                 | 0.00 | 0.00   | 0.00                            | 2.00   | -12.82  |
| data arrival time             | 0.00                                 | 0.00 | 0.00   | 0.00                            | 0.00   | 0.00    |
| statistical adjustment        | 0.00                                 | 0.00 | 0.00   | 0.00                            | 0.00   | 0.00    |
| slack (VIOLATED)              | 0.00                                 | 0.00 | 0.00   | 0.00                            | 0.00   | 0.00    |
|                               |                                      |      |        | 0.0                             | -10.20 | 3.61    |
|                               |                                      |      |        |                                 |        | -21.69  |
|                               |                                      |      |        |                                 |        | -21.69  |

$1.0 + 3 * 2.0 = 7.0$

$2.82 = 12.82 - 10.00$

$\sqrt{2.0^2 + 3.0^2} = 3.61$

10-14

Sensit = Sigma

# Running PrimeTime with POCV using Side File

Read and link design

Back-annotate parasitics

Apply design constraints

Timing analysis

Generate reports

```
set link_path { * timing.db }
read_verilog ...
link_design
set read_parasitics_load_locations true
read_parasitics ...
read_sdccreate_clock ...
set_timing_derate -early 0.80
set_timing_derate -late 1.20
read_ocvm pocv.table
read_ocvm distance.table
report_ocvm -type pocvm
set timing_pocvm_enable_analysis true
update_timing
report_constraint ...
report_analysis_coverage ...
report_clock_timing ...
report_timing -variation ...
```

Retain coordinate data \*

\* only for distance based derates

Read in POCV side file and distance tables (optional)

Report POCV information

Enable POCV analysis

Report paths with derating values

10-15

# Running PrimeTime with POCV using LVF

Read and link design

Back-annotate parasitics

Apply design constraints

Timing analysis

Generate reports

```
set link_path { * pocv_lvf.db }
read_verilog ...
link_design
set read_parasitics_load_locations true
read_parasitics ...

read_sdccreate_clock ...
set_timing_derate -early 0.80
set_timing_derate -late 1.20
read_ocvm pocv.table
read_ocvm distance.table
report_ocvm -type pocvm
set timing_pocvm_enable_analysis true
update_timing
report_constraint ...
report_analysis_coverage ...
report_clock_timing ...
report_timing -variation ...
```

Read in libraries with LVF slew-load tables

Retain coordinate data \*

\* only for distance based derates

Read in POCV side file and distance tables (optional)

Report POCV information

Enable POCV analysis

Report paths with derating values

10-16

Transition variation is supported in LVF format

Improved accuracy in Monte Carlo correlation

Has no effect if POCV side file is used as an input for delay variation

Transition variation is folded into cell delay variation

Reporting commands still report nominal transition. With and without transition variation, reported delay variation will be different

Based on input transition and output load, output transition sigma will be calculated using LVF tables in the library

Output transition sigma will be propagated to the input of receiver cell

Transition variation is off by default. To enable it

```
pt_shell> set_app_var timing_enable_slew_variation true
```

Constraint variation is supported in LVF format

Setup and hold

Recovery and removal

Non-sequential setup and hold

Nochange setup and hold

Clock-gating check

Using input transitions of constraint pin and related pin, constraint sigma will be calculated using LVF data in the library

```
report_ocvm -type pocvm [get_lib_timing_arcs -from lib/FF/CLK -to lib/FF/D]
```

Constraint sigma is RSS'ed into sigma of data required time

Constraint variation is off by default. To enable it

```
pt_shell> set_app_var timing_enable_constraint_variation true
```

## The 3 commands needed for POCV Debugging

- POCV debugging needs the following 3 reporting commands and options:

```
pt_shell> report_timing -variation
pt_shell> report_delay_calculation -derate
pt_shell> report_ocvm -type pocvm
```

- All reporting commands correlate information

- Mean and sigma from GBA `report_timing -variation` should match `report_delay_calculation -derate`
- Sigma or coefficient from `report_delay_calculation -derate` should be in sync with `report_ocvm -type pocvm`

## Incr. Column and Statistical Adjustment

- Incr column is calculated as

```
Incr = Path_Arrival(current) - Path_Arrival(previous)
Path_Arrival = Path_Arrival(mean) ± K × Path_Arrival(sigma)

(+) max mode
(-) min mode
K = timing_pocvm_report_sigma
```

- Statistical adjustment is added at the end to show slack adjustment (if any)

```
statistical adjustment = slack ± unadjusted slack
slack = mean_slack - K * sigma_slack
unadjusted_slack = required time - arrival time

(+) hold path
(-) setup path
K = timing_pocvm_report_sigma
```

|                                            |     |        |      |        |
|--------------------------------------------|-----|--------|------|--------|
| statistical adjustment<br>slack (VIOLATED) | 0.0 | -10.20 | 3.61 | -21.69 |
|--------------------------------------------|-----|--------|------|--------|

10-18

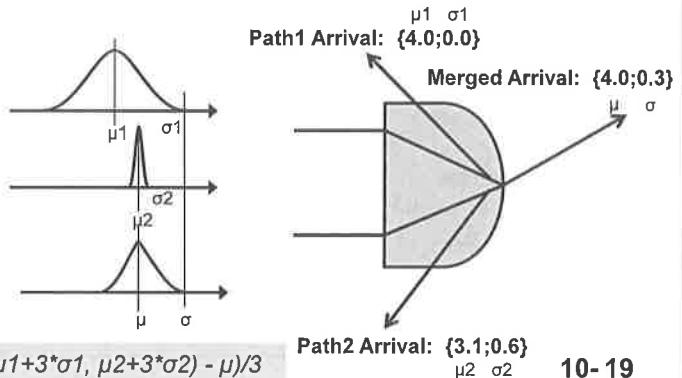
# Statistical Graph Pessimism

- A statistical graph pessimism is added into GBA report\_timing since statistical MAX operation is employed in POCV analysis

|                             |      |       |       |       |      |      |         |
|-----------------------------|------|-------|-------|-------|------|------|---------|
| in1 (in)                    | 0.00 | 0.00  | 0.00  | 0.00  | 0.00 | 0.00 | 0.00 r  |
| b1/Z (NR2)                  | 1.00 | 3.00  | 10.00 | 10.00 | 1.00 | 3.00 | 10.00 r |
| statistical graph pessimism | 3.00 | -2.24 |       | 0.00  | 4.00 | 2.00 | 10.00   |
| b2/Z (NR2)                  | 1.00 | 2.00  | 7.00  | 3.49  | 5.00 | 2.83 | 13.49 r |

## POCV MAX operator:

- $\mu = \text{MAX}(\mu_1, \mu_2)$ 
    - MAX of mean values
  - $\sigma = (\text{MAX}(\mu_1 + k^* \sigma_1, \mu_2 + k^* \sigma_2) - \mu) / k$ 
    - Derived from corner values  $\mu_1$
    - Where  $k = \text{timing_pocvm_corner_sigma}$
    - Default value  $k = 3$
- $\mu = \text{MAX}(\mu_1, \mu_2); \sigma = (\text{MAX}(\mu_1 + 3^* \sigma_1, \mu_2 + 3^* \sigma_2) - \mu) / 3$



10-19

Due to statistical MAX operation in POCV analysis

Graph pin slack attribute not found even with very high -nworst in report\_timing  
GBA path slack may not be the worst at -nworst 1 when compared to -nworst > 1

With statistical graph pessimism added into report\_timing GBA

Slack from report\_timing -nworst 1 (vs. -nworst > 1) will be the worst slack  
GBA report\_timing slack will match endpoint pin slack attribute

## Statistical Graph Pessimism in report\_timing -var

| Point                       | Incr                                               |        |        |        | Path                                              |        |         |
|-----------------------------|----------------------------------------------------|--------|--------|--------|---------------------------------------------------|--------|---------|
|                             | Mean                                               | Sensit | Corner | Value  | Mean                                              | Sensit | Value   |
| clock clk (rise edge)       | 0.00                                               |        |        | 0.00   | 0.00                                              |        | 0.00    |
| clock network delay (ideal) | 0.00                                               | 0.00   | 0.00   | 0.00   | 0.00                                              | 0.00   | 0.00    |
| input external delay        | 0.00                                               |        |        | 0.00   | 0.00                                              |        | 0.00 r  |
| in1 (in)                    | 0.00                                               | 0.00   | 0.00   | 0.00   | 0.00                                              | 0.00   | 0.00 r  |
| b1/Z (NR2)                  | 1.00                                               | 3.00   | 10.00  | 10.00  | 1.00                                              | 3.00   | 10.00 r |
| statistical graph pessimism | 3.00                                               | -2.24  |        | 0.00   | 4.00                                              | 2.00   | 10.00   |
| b2/Z (NR2)                  | 1.00                                               | 2.00   | 7.00   | 3.49   | 5.00                                              | 2.83   | 13.49 r |
| statistical graph pessimism | 5.00                                               | -2.58  |        | 0.00   | 10.00                                             | 1.16   | 13.49   |
| out1 (out)                  | 0.00                                               | 0.00   | 0.00   | 0.00   | 10.00                                             | 1.16   | 13.49 r |
| data arrival time           | $\sqrt{2.83^2 + X^2} = 1.16 \Rightarrow X = -2.58$ |        |        |        |                                                   |        | 13.49   |
| clock clk (rise edge)       | 2.00                                               |        |        | 2.00   | 2.00                                              |        | 2.00    |
|                             |                                                    |        |        |        | $\sqrt{3.0^2 + X^2} = 2.00 \Rightarrow X = -2.24$ |        |         |
| data required time          |                                                    |        |        | 2.00   | 0.00                                              |        | 2.00    |
| data required time          |                                                    |        |        | 2.00   | 0.00                                              |        | 2.00    |
| data arrival time           |                                                    |        |        | -10.00 | 1.16                                              |        | -13.49  |
| statistical adjustment      | 0.00                                               |        |        |        |                                                   |        | -11.49  |
| slack (VIOLATED)            |                                                    |        |        | -8.00  | 1.16                                              |        | -11.49  |

10-20

MAX calculation:

In order to know the values of (4,1) and (10, 1) used, you need to generate timing report through the other input(s) of b1 and b2 cells respectively.

Statistical Graph Pessimism Caveats :

`report_timing -from/-through -to <endpoint> vs. report_timing -to <endpoint>`

The consequence of POCV graph merging pessimism is that a timing report of a subgraph to an endpoint (i.e., `report_timing -from/-through -to <endpoint>`) has potentially less graph merging pessimism than a timing report to an endpoint (i.e., `report_timing -to <endpoint>` or endpoint pin slack)

Therefore, with POCV, we can expect that `report_timing -to <endpoint>` (or endpoint pin slack) will bound but may not necessarily match `report_timing -from/-through -to <endpoint>`

## POCV Summary

- **POCV methodology addresses random variation**

- Targeted for 16nm and below
- Works with distance-based derating (for spatial variation)
- Support constraint variation

- **The advantage of using POCV**

- Improved accuracy at advanced node
- Tighter the gap between graph-based and path-based analysis
- Faster incremental update timing and ECO fixing
- Tighter correlation between IC Compiler and PrimeTime

- **POCV Application Note** <https://solvnet.synopsys.com/retrieve/1521063.html>

- **Liberty User Guide** [http://www.opensourceliberty.org/resources\\_ccs.html#1](http://www.opensourceliberty.org/resources_ccs.html#1)

10-21

## Introduction to Advanced Waveform Propagation (AWP)

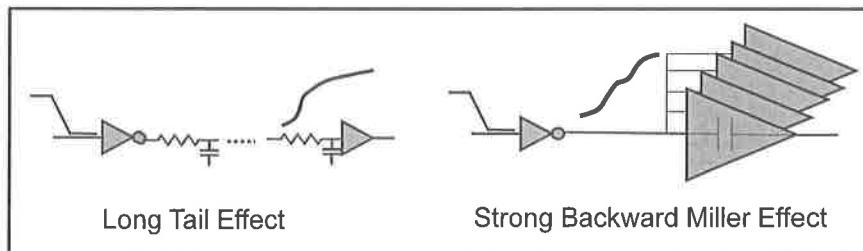
- In advanced process nodes, effects which distort design waveforms are becoming more prominent
- Accuracy of static timing analysis can degrade when the real design waveform deviates significantly from the characterization waveform
- Advanced Waveform Propagation technology is introduced to improve timing accuracy in the presence of waveform distortions

|                | 28nm                                                                              | 16nm                                                                               |
|----------------|-----------------------------------------------------------------------------------|------------------------------------------------------------------------------------|
| Miller effect  | 1X                                                                                | 2X                                                                                 |
| Wire R         | 1X                                                                                | 5X                                                                                 |
| VDD            | 0.85V                                                                             | 0.5V                                                                               |
| Waveform shape |  |  |

10-22

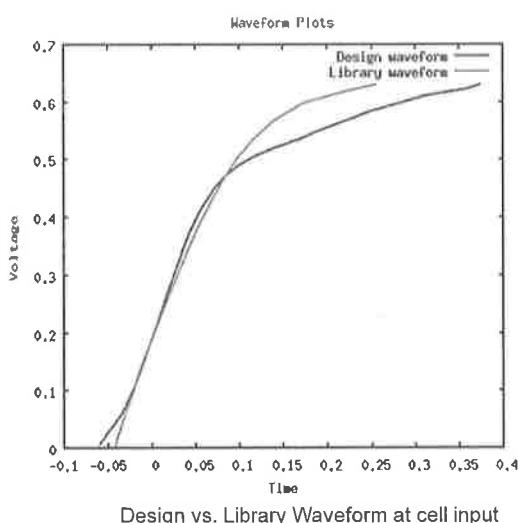
## Causes of Waveform Distortions

- Examples of waveform distortions occurring more frequently at smaller geometries:
  - Long Tail Effect
  - Strong Backward Miller Effect
- Cells are more sensitive to these distortions when  $V_{th}/VDD$  ratio is large.
  - Lower VDD makes high  $V_{th}$  cells more susceptible to these issues



10-23

## Long Tail Effect



- Increased metal resistance at smaller process nodes causes transition tails to increase
- Alters the switching behavior of the receiver thereby impacting path delay accuracy
- Optimizing the characterization waveform is not sufficient:
  - The distortion is stage-specific

10-24

At smaller nodes, the interconnect resistance is increasing dramatically, and therefore the switching waveform tends to “tail off” much more than before.

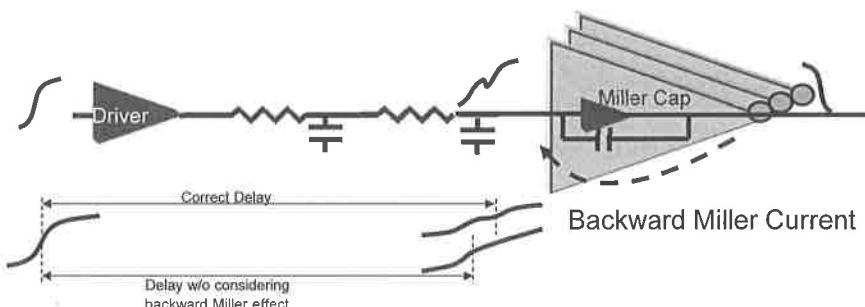
Remember that STA is dependent upon how accurate the library data is, based upon the waveform which is used during characterization.

In the example, in green is the waveform used during characterization, and in red is the actual waveform observed at the input in of the cell. While the waveforms match quite closely all the way through the upper slew trip point, the tail in the design waveform slows down and takes much longer to reach VDD. This waveform tail does impact the response of the circuit and therefore impacts path delay.

Since this type of distortion is stage specific, simply adjusting the characterization waveform would not be representative of the typical design operation.

## Strong Backward Miller Effect

- The Miller capacitance from receiver output to input distorts the waveform at the receiver input
  - Observed most often on high-fanout nets
- Both stage delay and path delay can be impacted
- Characterization optimization is not sufficient
  - Amount of distortion is stage-specific



10-25

At smaller nodes, the interconnect resistance is increasing dramatically, and therefore the switching waveform tends to “tail off” much more than before.

Remember that STA is dependent upon how accurate the library data is, based upon the waveform which is used during characterization.

In the example, in green is the waveform used during characterization, and in red is the actual waveform observed at the input in of the cell. While the waveforms match quite closely all the way through the upper slew trip point, the tail in the design waveform slows down and takes much longer to reach VDD. This waveform tail does impact the response of the circuit and therefore impacts path delay.

Since this type of distortion is stage specific, simply adjusting the characterization waveform would not be representative of the typical design operation.

# AWP Correlation With HSPICE

## PrimeTime Report:

### Without AWP:

| Point          | Fanout | Cap Trans | Incr    | Path    | Trans | Incr    | Path    |
|----------------|--------|-----------|---------|---------|-------|---------|---------|
| cell_u1/a      |        | 0.097     | 0.030 & | 2.702 f | 0.097 | 0.029 & | 2.702 f |
| cell_u1/nz     |        | 0.063     | 0.070 & | 2.772 r | 0.073 | 0.072 & | 2.774 r |
| net_a (net) 26 | 0.2966 |           |         |         |       |         |         |
| cell_u2/a      |        | 0.288     | 0.158 & | 2.930 r | 0.406 | 0.161 & | 2.933 r |
| cell_u2/z      |        | 0.372     | 0.344 & | 3.274 r | 0.443 | 0.385 & | 3.318   |
| net_b (net) 15 | 0.1679 |           |         |         |       |         |         |
| cell_u3/a      |        | 0.392     | 0.021 & | 3.296 r | 0.476 | 0.025 & | 3.343 r |

HSPICE delay 632ps Without AWP 594ps

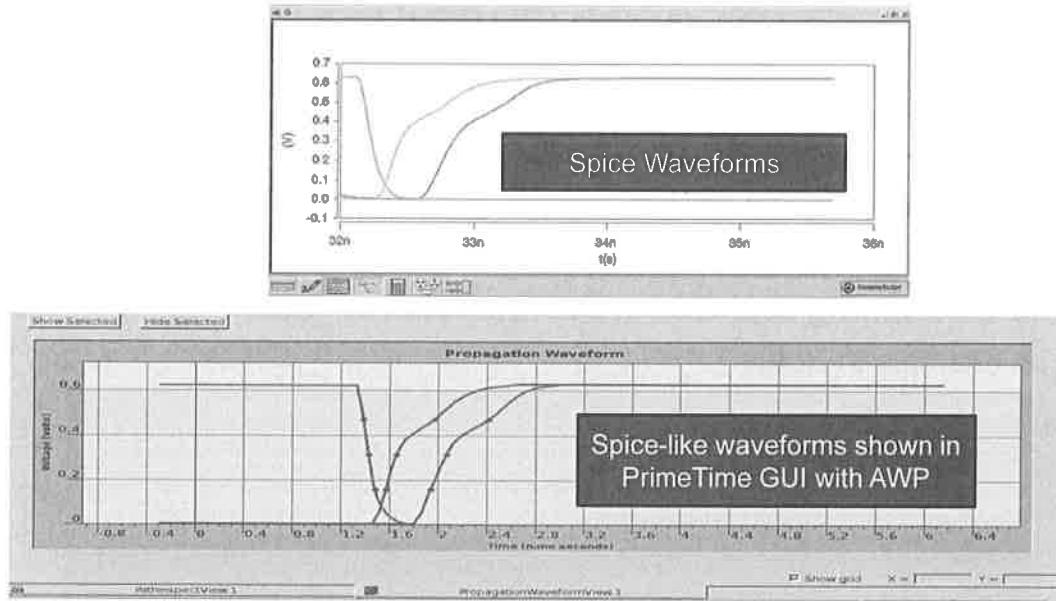
### With AWP:

With AWP 641ps

Improved correlation to HSPICE

10-26

## Advanced Waveform Propagation Visualization in the GUI



10- 27

## POCV Precedence Rules: POCV Side File vs. LVF

- PrimeTime allows to apply both – POCV side file and the library with POCV LVF at the same time

- If both POCV formats are applied on the same library cell, the side file with POCV single coefficient takes precedence by default

- The behavior can be controlled by the variable:

```
pt_shell> set_app_var timing_pocvm_precedence
```

- **file** (the default) - POCV side file takes precedence over LVF data available in library
- **library** - POCV LVF data takes precedence over POCV side file
- **lib\_cell\_in\_file** – POCV side file takes precedence over LVF data if it is applied on the same library cell

10-34

```
pt_shell> report_ocvm -type pocvm \
 [get_lib_timing_arcs -from lib1/INVD1/I -to lib1/INVD1/ZN]
```

```
min rise table
Sense / Type: negative_unate
 Slew
Load 0.0010000 0.0024000 0.0052000 0.0108000 0.0221000 0.0445000

0.0088000 0.0004760 0.0006770 0.0010750 0.0018700 0.0034380 0.0066260
0.0264000 0.0006510 0.0009010 0.0013030 0.0020810 0.0036780 0.0068180
0.0608000 0.0008400 0.0011660 0.0017140 0.0025580 0.0041120 0.0072490
0.1296000 0.0011150 0.0015200 0.0021930 0.0033170 0.0050870 0.0081530
0.2672000 0.0015210 0.0020330 0.0028830 0.0042420 0.0065220 0.0100720
0.5424000 0.0021550 0.0027930 0.0038530 0.0055630 0.0084240 0.0129550
1.0936000 0.0032040 0.0039770 0.0053210 0.0075150 0.0109600 0.0165820
```

```
min fall table
```

## How is sigma\_slack Calculated?

$\text{sigma\_slack} = \sqrt{(\text{sigma\_arrival}^2 + \text{sigma\_required}^2)}$

(+) if sigma\_required is positive value

(-) if sigma\_required is negative value

where

$\text{sigma\_required}^2 = \text{sigma\_capture\_path}^2 - \text{sigma\_crpr}^2$

where

$\text{sigma\_crpr}^2 = \text{sigma\_common\_launch\_path}^2 + \text{sigma\_common\_capture\_path}^2$

# Transition Variation is Combined Into Cell Delay Variation

- Cell delay sigma induced by input transition sigma is calculated as below

$$\Delta\sigma(S\sigma) = (D(Snom+N^*S\sigma) - D(Snom))/N$$

where

|                         |                                                                |
|-------------------------|----------------------------------------------------------------|
| $\Delta\sigma(S\sigma)$ | additional variation on cell delay due to transition variation |
| $D(Snom)$               | cell delay @ mean slew                                         |
| $D(Snom+N^*S\sigma)$    | cell delay @ mean_slew + N * slew_sigma                        |
| N                       | corner sigma timing_pocvm_corner_sigma                         |

- The combined sigma of cell delay sigma and additional sigma induced by input transition variation is calculated using a proprietary model which assumes cell delay and input transition are correlated

- To see library transition variation table and how it's used:

- report\_ocvm -type pocvm
- report\_delay\_calculation -derate

10-36

```
pt_shell> report_ocvm -type pocvm \
 [get_lib_timing_arcs -from lib/INVD1/I -to lib/INVD1/Z]
min rise table
Sense / Type: negative_unate
Delay:
Load | Slew
-----+-----
0.0088000 | 0.0004760 0.0006770 0.0010750 0.0018700 0.0034380 0.0066260
0.0264000 | 0.0006510 0.0009010 0.0013030 0.0020810 0.0036780 0.0068180
......
1.0936000 | 0.0032040 0.0039770 0.0053210 0.0075150 0.0109600 0.0165820
Transition:
Load | Slew
-----+-----
Slew | 0.0010000 0.0024000 0.0052000 0.0108000 0.0221000 0.0445000
0.0088000 | 0.0004474 0.0011155 0.0014607 0.0027899 0.0060622 0.0327056
......
1.0936000 | 0.0130970 0.0099494 0.0046566 0.0036081 0.0052415 0.0588888
```

## POCV coefficient from report\_delay\_calculation -derate

```
pt_shell> report_delay_calculation -from U1/I -to U1/ZN -derate
Advanced driver-modeling used for rise and fall.

 Rise Fall

Input transition time = 0.011600 0.011600 (in library unit)
Effective capacitance = 0.002000 0.002000 (in pF)
Effective capacitance = 0.002000 0.002000 (in library unit)
Output transition time = 0.007237 0.006342 (in library unit)
Cell delay = 0.008886 0.009559 (in library unit)

POCVM coefficient = 0.050000 0.050000
POCVM coef scale factor = 1.000000 1.000000
POCVM distance derate = 1.000000 1.000000
POCVM guardband = 1.020000 1.020000
Incremental derate = 0.000000 0.000000
Cell delay derated = 0.009064 0.009750 (in library unit)
Cell delay sigma = 0.000453 0.000488 (in library unit)

Cell delay derated = "Cell delay" * ("POCVM guardband" * "POCVM distance derate" + "Incremental derate")
Cell delay sigma = "Cell delay" * ("POCVM guardband" * "POCVM coefficient" * "POCVM coef scale factor")
```

```
POCV coefficient for random variation
version : 4.0
ocvm_type : pocvm
object_type: lib_cell
rf_type : rise fall
delay_type : cell
derate_type: early
object_spec: lib28nm/invx*
coefficient: 0.05
```

10-37

Here's how the output of report\_delay\_calculation –derate looks like using POCV single coefficient.

As you can see, in this example POCV coefficient 5% is applied for both rise and fall delay.

# Specifying POCV guardband vs. scaling factor

|                                                    |                                               |
|----------------------------------------------------|-----------------------------------------------|
| <code>set_timing_derate</code>                     |                                               |
| <code>-pocvm_guardband value</code>                | Scales both the nominal and POCV coefficients |
| <code>-pocvm_coefficient_scale_factor value</code> | Scales the POCV coefficient only              |

The POCVM guardband and POCVM coefficient scaling factor are used by the `report_delay_calculation -derate` command.

```
pt_shell> set_timing_derate -cell_delay -late -clock 1.02 -pocvm_guardband
pt_shell> set_timing_derate -cell_delay -early -clock 0.98 -pocvm_guardband
pt_shell> set_timing_derate -cell_delay -late -data 1.05 -pocvm_coefficient_scale_factor
pt_shell> set_timing_derate -cell_delay -early -data 1.03 -pocvm_coefficient_scale_factor
```

10-38

These are the new POCV options added to `set_timing_derate`.

`-pocvm_guardband` is similar to AOCV guardband and scales both nominal and POCV coefficients.

`-pocvm_coefficient_scale_factor` scales only the POCV coefficient.

`-pocvm_guardband`

Applies the derating on top of parametric on-chip variation (POCV) analysis. The specified derating factor applies to both the mean and standard deviation of the statistical delay distribution. This option has an effect only when the `timing_pocvm_enable_analysis` variable is set to true, and only affects the timing arcs analyzed by POCV analysis. The `-pocvm_guardband` option cannot be used with the `-dynamic` option.

`-pocvm_coefficient_scale_factor`

Applies the derating to the variation coefficient C of the statistical delay distribution in POCV analysis. This effectively scales the sigma without affecting the mean. This option has an effect only when the `timing_pocvm_enable_analysis` variable is set to true, and only affects the sigma of timing arcs analyzed by POCV analysis. The `-pocvm_coefficient_scale_factor` option cannot be used with the `-dynamic` option.

To report the settings for AOCV guardband, POCV guardband, or POCV coefficient scaling, use the `report_timing_derate` command with the `-aocvm_guardband`, `-pocvm_guardband`, or `-pocvm_coefficient_scale_factor` option.

## Transition Variation in report\_delay\_calculation -derate

```
pt_shell> report_delay_calculation -from I/I -to I/ZN -derate -min

Cell delay = 0.090344 0.093774 (in library unit)

POCVM delay sigma = 0.004517 0.004689
POCVM delay sigma induced by input transition sigma = -0.003557 -0.003525
POCVM combined delay sigma = 0.005750 0.005866
POCVM output transition sigma = 0.006414 0.005396
POCVM input transition sigma = 0.011627 0.010163

POCVM coef scale factor = 0.970000 0.970000
POCVM distance derate = 0.879554 0.879554
POCVM guardband = 0.980000 0.980000
Incremental derate = -0.040000 -0.040000
Cell delay derated = 0.074260 0.077079 (in library unit)
Cell delay sigma = 0.005468 0.005576 (in library unit)
```

10-39

# POCV LVF in report\_delay\_calculation -derate

```
pt_shell> report_delay_calculation -from I/I -to I/ZN -derate
Advanced driver-modeling used for rise and fall.

 Rise Fall

Input transition time = 0.011600 0.011600 (in library unit)
Effective capacitance = 0.002000 0.002000 (in pF)
Effective capacitance = 0.002000 0.002000 (in library unit)
Drive resistance = 0.001000 0.001000 (in Kohm)
Output transition time = 0.016620 0.011003 (in library unit)

Cell delay = 0.013041 0.010004 (in library unit)
POCVM delay sigma = 0.000652 0.000500
POCVM coef scale factor = 1.000000 1.000000
POCVM distance derate = 1.000000 1.000000
POCVM guardband = 1.020000 1.020000
Incremental derate = 0.000000 0.000000
Cell delay derated = 0.013302 0.010204 (in library unit)
Cell delay sigma = 0.000665 0.000510 (in library unit)
```

```
ocv_sigma_well_size ("delay_template_Te77") {
 sigma_type = "late";
 index = 10;
 values = {0.000354, 0.0024, 0.0498, 0.1294, 0.2472, 0.5424, 1.0934};
 index_r = {0.001, 0.0024, 0.0052, 0.0106, 0.0221, 0.0445, 0.0957};
 values_r = {0.000476, 0.000477, 0.001075, 0.001970, 0.002430, 0.004682, 0.012927,
 0.008451, 0.009811, 0.001303, 0.002081, 0.003478, 0.006818, 0.013147,
 0.000940, 0.001186, 0.001714, 0.002558, 0.004112, 0.007242, 0.013319,
 0.000115, 0.001520, 0.002893, 0.005271, 0.010532, 0.021077, 0.044287,
 0.000159, 0.002783, 0.005383, 0.009494, 0.012955, 0.026171,
 0.003294, 0.003977, 0.005321, 0.007318, 0.010969, 0.016582, 0.023796};
```

10-40

## POCV Constraint Variation in report\_delay\_calculation

```
pt_shell> report_delay_calculation -from FF1/CLK -to FF1/D
Fall Delay Sigma
cell delay sigma = 0.0163316
Table is indexed by
(X) related_pin_transition = 0.0193678
(Y) constrained_pin_transition = 0.0164327
Relevant portion of lookup table:
(X) 0.0049 (X) 0.0531
(Y) 0.0049 (Z) 0.0137 (Z) 0.0321
(Y) 0.0531 (Z) 0.0011 (Z) 0.0215
Z = A + B*X + C*Y + D*X*Y
A = 0.0131 B = 0.3776
C = -0.2651 D = 0.8605
Z = 0.0163316
scaling result for operating conditions
multiplying by 1 gives 0.0163316
Cell Delay Sigma
Fall: 0.0163316
report_delay_calculation -from FF1/CLK -to FF1/D
```

10-41

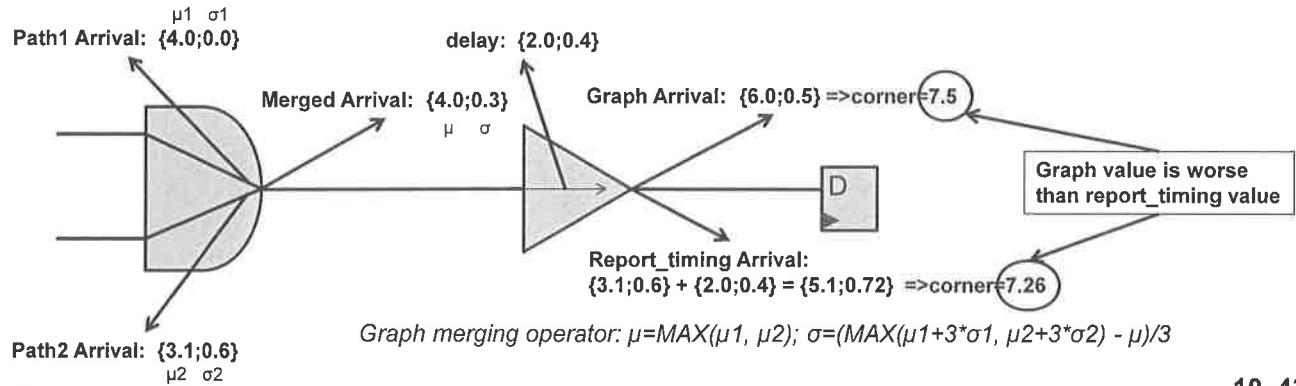
# POCV Constraint Variation in report\_timing -variation

| Point                                     | Mean   | Sensit | Incr   | Path    |
|-------------------------------------------|--------|--------|--------|---------|
| <hr/>                                     |        |        |        |         |
| clock CLK (rise edge)                     |        |        | 0.000  | 0.000   |
| clock network delay (propagated)          |        |        | 0.104  | 0.104   |
| Uff1/CP (DFD4BWP)                         | 0.000  | 0.000  | 0.000  | 0.104 r |
| report_timing -variation -sign 3 -nosplit |        |        |        |         |
| data arrival time                         |        |        |        | 0.235   |
| clock CLK (rise edge)                     |        |        | 0.000  | 0.000   |
| clock network delay (propagated)          |        |        | 0.203  | 0.203   |
| clock reconvergence pessimism             | -0.039 | -0.009 | -0.052 | 0.150   |
| clock uncertainty                         |        |        | 0.100  | 0.250   |
| Uff2/CP (DFD4BWP)                         |        |        |        | 0.250 r |
| library hold time                         | 0.016  | 0.016  | 0.060  | 0.310   |
| data required time                        | 0.267  | 0.016  |        | 0.310   |
| <hr/>                                     |        |        |        |         |
| data required time                        | 0.267  | 0.016  |        | 0.310   |
| data arrival time                         | -0.247 | 0.008  |        | -0.235  |
| <hr/>                                     |        |        |        |         |
| slack (VIOLATED)                          | -0.020 | 0.018  |        | -0.075  |

10-42

## Example: Statistical Graph Pessimism

- Graph pin slack from update\_timing includes merging pessimism of statistical max operation
- Slack from report\_timing does not include any merging
  - A statistical graph pessimism is added to make sure that report\_timing -nworst 1 slack is equivalent to graph pin slack



10-43

This page was intentionally left blank

# Agenda

DAY  
3

- |    |                                            |  |
|----|--------------------------------------------|--|
| 8  | Signal Integrity: Crosstalk Delay Analysis |  |
| 9  | Signal Integrity: Crosstalk Noise Analysis |  |
| 10 | Correlation: POCV and AWP Analysis         |  |
| 11 | Timing Closure: ECO/What If Analysis       |  |
| 12 | Large Data: DMSA and Hyperscale Analysis   |  |
| 13 | Conclusion                                 |  |

11-1

## Unit Objectives

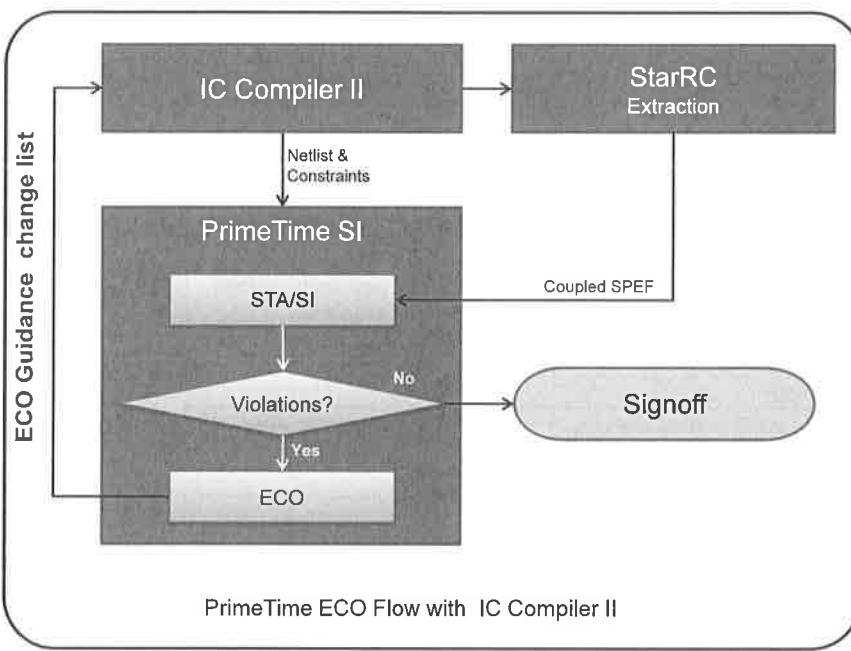


After completing this unit, you should be able to:

- **Describe our recommended ECO flow that automates:**
  1. Reducing dynamic and leakage power consumption and
  2. Fixing of DRC, Noise, Setup and Hold Violations
- **Write out an ECO change list for implementing the changes**
- **List additional input requirements for performing physically aware ECO**
- **Differentiate the *open\_site* vs. *occupied\_site* modes during physically aware ECO**

11-2

# Signoff-driven Physically Aware ECO Flow



## ■ PrimeTime inputs:

- Netlist (Verilog)
- Timing constraints (SDC)
- Power Intent (UPF), if applicable
- Layout (DEF) + Floorplan Constraints (Tcl)
- Ref Library + Tech Info (LEF)
- RC Parasitics with coordinates (SPEF)
- Standard cell spacing rules for advanced technologies (Encrypted Tcl)

## ■ PrimeTime output:

- ASCII ECO file with coordinates

11-3

*StarRC* can read *NDM*.

## Recommended ECO Guidance Flow in PrimeTime-SI

| Recommended ECO guidance |                                                                                                                                                                        | Power recovery                                                                      | DRC Fixing                              | Noise and Timing Fixing               | Final leakage recovery                          |
|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|-----------------------------------------|---------------------------------------|-------------------------------------------------|
| Fix mechanism            | Sizing                                                                                                                                                                 | DRC and noise use buffer insertion and sizing                                       | Setup uses sizing                       | Hold uses buffer insertion and sizing | Sizing                                          |
| Precedence rules         | Does not introduce new timing or DRC violations                                                                                                                        | DRC and noise alter setup and hold slack as needed (fix noise after DRC and timing) | Setup honors DRC, alters hold if needed | Hold honors setup slack and DRC       | Does not introduce new timing or DRC violations |
| Flow options             | <ul style="list-style-type: none"> <li>Perform ECOs that result in more changes</li> <li>Implement the changes in IC Compiler before next PrimeTime ECO run</li> </ul> |                                                                                     |                                         |                                       | Final step after timing closure                 |
| Highlights               | <ul style="list-style-type: none"> <li>Supports full-chip STA, SI, AOCV GBA and PBA flows</li> <li>Fixes across all scenarios</li> </ul>                               |                                                                                     |                                         |                                       |                                                 |
| QoR                      | No negative impact                                                                                                                                                     | 95%                                                                                 | 70%                                     | 95%                                   | No negative impact                              |

11-4

In *PrimeTime*, we recommend that ECO fixing is performed in the above order. The reason *power recovery* is first is because this step generally downsizes cells: That can make additional room, which subsequent DRC, noise and timing fixing can take advantage of. After timing fixing, final leakage recovery can be performed to take advantage of any remaining DRC and timing slack.

The *precedence rules* listed above reflect the order in which we recommend the violations be fixed (after initial *power recovery*). For instance, DRC and noise fixing have the freedom to alter timing as needed; Fixing setup violations honors DRC/noise, but can impact hold violations if needed, and so on.

Despite the recommended order, sometimes the choice of which type of violations to fix first may depend on the overall ECO flow. For instance, if the design has significantly more hold violations than setup violations, it may make sense to fix the hold violations first, since the impact of all the changes on implementation would be greater. Once these changes are implemented, the user could come back into *PrimeTime* ECO to reassess what should be fixed next.

The techniques available to *PrimeTime* for all Eco guidance or fixing (DRC, noise, timing, power) are cell sizing and buffer insertion.

You can take advantage of additional `fix_eco_xxx` command options, for example:  
`fix_eco_timing -setup_margin` controls the *slack* or *margin* for fixing setup timing; The margin is 0 by default; A positive value specifies over-fixing by that amount, and a negative value specifies under-fixing.

Regardless of the type of violation being fixed, there is a rich array of features and advanced SI analysis techniques to draw upon for ECO guidance for both single and multiple scenario analysis.

## Power (and area) recovery: fix\_eco\_power

- Performs cell area/internal power recovery driven by downsizing or swapping cells having positive setup slacks or by removing redundant buffers with positive hold slacks
  - The analysis is based on PrimeTime-SI's sign off timing

```
fix_eco_power
[-cell_type cell_types]
[-setup_margin margin]
[-attribute attribute_name]
[-verbose]
```

11-5

## DRC and Noise Fixing: fix\_eco\_drc

- **fix\_eco\_drc** fixes DRC and Noise violations using sizing and buffer insertion
  - max\_transition
  - max\_capacitance
  - max\_fanout
  - noise
- **Cells with dont\_touch will not be resized**
  - set\_dont\_touch on hierarchical blocks will propagate downwards to subblocks
    - ◆ Buffers will not be inserted on dont\_touch nets, or at dont\_touch cells
    - ◆ Library cells with dont\_use will not be used for sizing or buffer insertion
- **write\_changes command generates ASCII list for buffer insertion (insert\_buffer) and cell upsizing/downsizing (size\_cell)**

11-6

```
fix_eco_drc -type <max_transition|max_capacitance|max_fanout|noise>
[-method fixing_method_type]
[-buffer_list list]
[-physical_mode
 none|open_site|occupied_site|freeze_silicon]
[-hold_margin margin] [-setup_margin margin]
[-verbose]
```

The **fix\_eco\_drc** command attempts to improve or remove all DRC and noise violations, minimize impact on timing and area and minimize the number of ECO changes.

## ECO DRC fixing : Verbose mode

- Shows additional information during the ECO fixing process

- Supported by all the fix\_eco\_drc/timing/power -verbose commands
- `set eco_report_unfixed_reason_max_endpoints 50`
- Displays unfixable violations with the reasons
- These reasons will guide the user to the next steps

```
pt_shell> set_app_var eco_report_unfixed_reason_max_endpoints 100; # default 0
pt_shell> fix_eco_drc -type max_transition -method size_cell -verbose

Unfixable violations:
 A - There are available lib cells outside area limit
 C - The violation is in clock network
 I - Buffer insertion with given lib cells cannot fix the violation
 P - Driver cell of the violation is a port
 Q - Driver cell of the violation is a sequential cell
 S - Cell sizing with alternative lib cells cannot fix the violation
 T - Timing margin is too tight to fix the violation
 V - Driver cell of the violation is dont_touched

Violation Reasons

U63ASTipoInst494/I S
I_ORCA_TOP/I_BLENDER/s3_op2_reg[18]/Q Q
```

11-7

## Iterating fix\_eco\_drc

- To address any remaining DRC violations, try the following steps
  - Re-execute ECO DRC fixing process to see if fixing rate can be improved
  - Analyze the unfixable reasons and re-execute the fixing process
    - ◆ Example: Add insert\_buffer method

```
pt_shell> fix_eco_drc -type max_transition -method size_cell -verbose
Violation
```

Reasons

I\_ORCA\_TOP/I\_BLENDER/s3\_op2\_reg[18]/Q

Q

Remaining Violations:

| Violation Type             | Count |
|----------------------------|-------|
| Total remaining violations | 1     |
| Unfixable violations       | 1     |

```
pt_shell> fix_eco_drc -type max_transition \
-method [size cell][insert buffer]} \
-buffer_list {buf4 buf6 buf8}
```

Remaining Violations:

| Violation Type             | Count |
|----------------------------|-------|
| Total remaining violations | 0     |
| Unfixable violations       | 0     |

11-8

## Setup and Hold Time Fixing: fix\_eco\_timing 1/2

- **fix\_eco\_timing fixes timing violations**
  - setup
  - hold
- **Cells with dont\_touch will not be resized**
  - Buffers will not be inserted on dont\_touch nets, or at dont\_touch cells
  - Library cells with dont\_use will not be used for sizing or buffer insertion
- **write\_changes command generates ASCII list for buffer insertion (insert\_buffer) and cell upsizing/downsizing (size\_cell)**

11-9

fix\_eco\_timing command honors dont\_touch attributes in the netlist. During setup fixing it will not resize the cell that has dont\_touch attribute. During hold fixing, it checks the net and/or cell has this attributes before buffer insertion and exclude them if they have.

PrimeTime also supports dont\_touch propagation to sub-hierarchical block or cell. If the block is set dont\_touch, then it propagates downwards. Fix\_eco\_timing honors this downstream dont\_touch behaviors.

## Setup and Hold Time Fixing: fix\_eco\_timing 2/2

### ■ fix\_eco\_timing fixes setup timing violations

```
fix_eco_timing -type setup
```

- Uses cell sizing method
- Setup fixing doesn't cause DRC violations

### ■ fix\_eco\_timing fixes hold timing violations

```
fix_eco_timing -type hold -buffer_list <buffer_list> \
 -methods {size_cell insert_buffer}
```

- Uses cell sizing and buffer insertion methods
- Choose a subset of buffers to minimize runtime
- Hold fixing does not cause DRC or setup violations

11-10

## Setup/Hold fixing in Path-Based Analysis

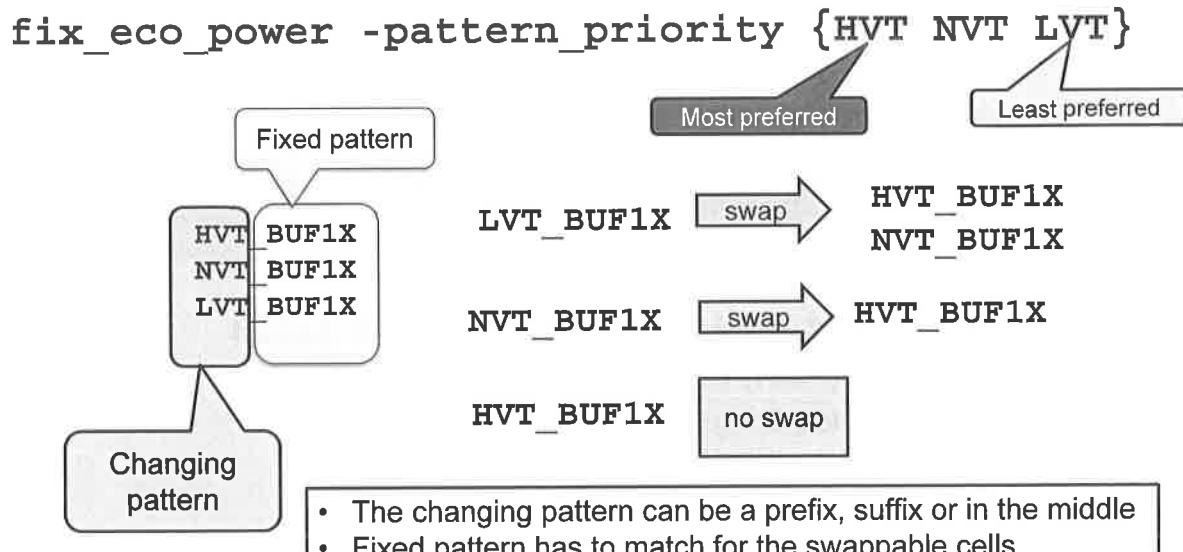
To remove pessimism in setup/hold fixing

```
fix_eco_timing
 -pba_mode <path | exhaustive>
```

- Fix setup & hold violations using path-based analysis (PBA) timing

- Setup/Hold fixing will use PBA timing
- Path for early ECO fixing
- Exhaustive for sign off fixing

## Leakage Recovery : fix\_eco\_power -pattern\_priority



11-12

## Reporting Vt Cell Usage Example

```
report_cell_usage -pattern_priority {HVT LVT}
```

Before :

| Pattern | Combinational cell | Sequential cell | Total          |
|---------|--------------------|-----------------|----------------|
| HVT     | 82081 ( 2%)        | 1078 ( 0%)      | 83159 ( 2%)    |
| LVT     | 3908613 ( 83%)     | 696435 ( 15%)   | 4605048 ( 98%) |
| Others  | 4115 ( 0%)         | 532 ( 0%)       | 4647 ( 0%)     |

More LVT, Fewer HVT cells

After :

| Pattern | Combinational cell | Sequential cell | Total          |
|---------|--------------------|-----------------|----------------|
| HVT     | 3678276 ( 78%)     | 3491 ( 0%)      | 3681767 ( 78%) |
| LVT     | 312418 ( 7%)       | 694022 ( 15%)   | 1006440 ( 21%) |
| Others  | 4115 ( 0%)         | 532 ( 0%)       | 4647 ( 0%)     |

More HVT, Fewer LVT cells

11-13

Here is an example of the “report\_cell\_usage”

This example shows pattern swap from “HVT” to “LVT”

As you can see in the after cell usage report, “HVT” high Vt cell increased from 2% to 78% of the design, which will save leakage power.

## Handling Unconstrained Cells

- In some designs some portions of the logic may be unconstrained for timing because:
  - Not all scenarios are used in the leakage recovery run
  - False paths are present
- Users can prevent cells which are unconstrained from being swapped by using the following variable :
  - set eco\_power\_exclude\_unconstrained\_cells true
    - ◆ Default false
- When this variable is set to true, unconstrained cells will not be considered for swapping in leakage recovery
  - Essentially treated as dont\_touch

11-14

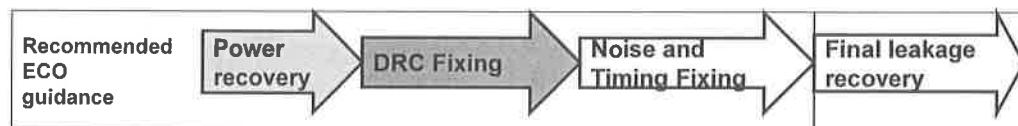
How does leakage recovery handle unconstrained cells? By default uncontained cells are swapped.

But in case you want to prevent this behavior, you can set the application variable

`set eco_leakage_exclude_unconstrained_cells true`

The unconstrained cells would be treated essentially as "dont\_touch"

## Recommended ECO Guidance Command Flow : 1/2

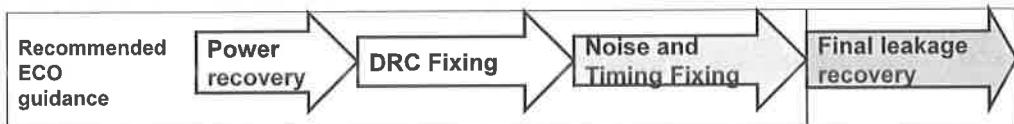


```
report_power
fix_eco_power
report_power

alias report_drcViolations "report_constraint -all_violators \
-max_transition -max_capacitance -max_fanout"
report_drcViolations
set buflist "bufbd1 buffd1 bufbd2"
fix_eco_drc -type max_transition -buffer_list $buflist -verbose
fix_eco_drc -type max_cap -buffer_list $buflist -verbose
fix_eco_drc -type max_fanout -buffer_list $buflist -verbose
report_drcViolations
```

11-15

## Recommended ECO Guidance Command Flow : 2/2



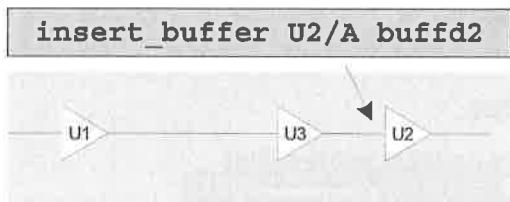
```
report_noise
fix_eco_drc -type noise -verbose
report_noise

report_analysis_coverage
fix_eco_timing -type setup
report_analysis_coverage
fix_eco_timing -type hold -buffer_list $buflist
report_analysis_coverage

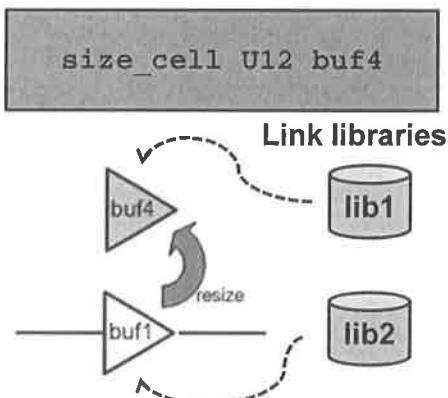
report_power
fix_eco_power -pattern_priority {HVT NVT LVT}
report_power
```

11-16

## Creating the Change List: write\_changes



- **write\_changes** creates the change list file containing commands
  - `insert_buffer`
  - `size_cell`



```
write_changes -format icc2tcl \
 -output mychange.tcl
write_changes -format text \
 -output mychangehistory.txt
```

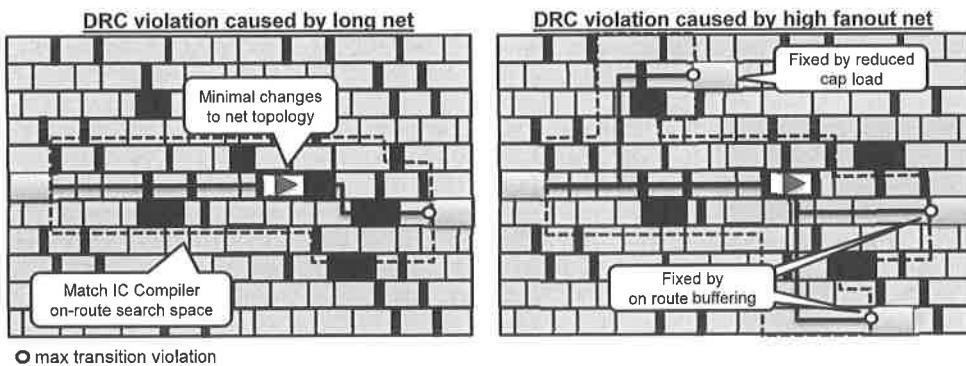
11-17

`insert_buffer` and `size_cell` are the commands used by `fix_eco_timing` to insert buffers and resize cells – They are also commands that appear in the file of changes generated by the `write_changes` command.

# Introducing Physically Aware ECO to Improve Fix Rates

## ■ PrimeTime finds space on route to insert buffer

- Effectively fixes DRC violations caused by long or high fanout nets
- Accurately models the on-route buffer by splitting parasitics



Utilize "add\_buffer\_on\_route" for IC Compiler implementation

11-18

*ICC legalizes buffer placement with minimum physical impact*

# Using Physically Aware ECO

## ■ Use option for `fix_eco_drc`/`fix_eco_timing` commands

“`-physical_mode`”

- Available for single scenario or DMSA

## ■ Physical Inputs Needed:

- DEF (Design Exchange Format) for physical layout
- LEF (Library Exchange Format) for basic library information

## ■ To read in the required physical information

`set_eco_options`

- ◆ Specifies location of necessary physical information input LEF/DEF files
- ◆ Supports reading multiple LEF files
- Effectively fixes DRC violations caused by long or high fanout nets
- Accurately models the on-route buffer by splitting parasitics

11-19

## `set_eco_options`

Specifies options for ECO commands such as `fix_eco_timing`, `fix_eco_drc`, and `fix_eco_power`.

### SYNTAX

```
status set_eco_options
[-physical_tech_lib_path file_name_list]
[-physical_lib_path file_name_list]
[-physical_design_path file_name_list]
[-physical_constraint_file file_name]
[-physical_lib_constraint_file file_name_list]
[-log_file file_name]
[-mim_group cell_list]
[-filler_cell_names list]
[-programmable_spare_cell_names list]
[-drc_setup_margin margin] [-drc_hold_margin margin] [-timing_setup_margin margin]
[-timing_hold_margin margin] [-power_setup_margin margin] [-power_hold_margin margin]
```

## **open\_site and occupied\_site Physical Modes**

### ■ Open site mode

```
fix_eco_drc/fix_eco_timing -physical_mode open_site
```

- Buffer insertion and cell sizing only when open sites available
- Minimizes disturbance to physical layout
- Targeted for final stage ECO

### ■ Occupied site mode

```
fix_eco_drc/fix_eco_timing -physical_mode occupied_site
```

- Buffer insertion and cell sizing allowed to overlap existing cells
- Provides tool the flexibility to address all the violations
- ECO implementation may see more disturbance

11-20

# Reading in Physical Constraint Data

## ■ Voltage areas and/or supplemental blockages

|                                                    |                                                                                                                                      |
|----------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| <b>set_eco_options</b>                             | Specify physical information                                                                                                         |
| <b>-physical_lib_path &lt;file_name_list&gt;</b>   | Specifies path to LEF files                                                                                                          |
| <b>-physical_design_path &lt;file_name&gt;</b>     | Specifies path to DEF file                                                                                                           |
| <b>-log_file &lt;my_log_file&gt;</b>               | Specifies log file to capture output of reading the physical files                                                                   |
| <b>-physical_constraint_file &lt;file_name&gt;</b> | Specifies the physical constraints file with<br><code>create_placement_blockage</code> and <code>create_voltage_area</code> commands |

```
create_voltage_area -name p0 -coordinate { 45.000 2073.280 1161.230 2776.000 45.000 2776.000 2641.665 3751.860 } \
-guard_band_x 5 -guard_band_y 5 [get_cells "u0_0/BLOCK"]

create_voltage_area -name p1 -coordinate { 45.000 45.760 2641.665 1019.200 45.000 1019.200 1161.230 1721.920 } \
-guard_band_x 5 -guard_band_y 5 [get_cells "u0_1/BLOCK"]

create_voltage_area -name p2 -coordinate { 4622.100 2073.280 5738.330 2776.000 3141.665 2776.000 5738.330 3751.860 } \
-guard_band_x 5 -guard_band_y 5 [get_cells "u0_2/BLOCK"]

create_placement_blockage -name Block1 -bbox { 200.0 200.0 300.0 300.0 }
```

11-21

## Example Physically Aware ECO in PT (1/2)

```
read_verilog ORCA_TOP.v; link_design ORCA_TOP
set_app_var read_parasitics_load_locations true
read_parasitics -format SPEF -keep_capacitive_coupling test.SPEF
load_upf design.upf
read_sdc design.sdc

Update timing with pin slack and arrival info
set_app_var timing_save_pin_arrival_and_slack true
update_timing -full

Configure Physical Information needed for ECO along with Voltage Areas
set_eco_options -physical_tech_lib_path tech.lef \
-physical_lib_path $lef_files \
-physical_lib_constraint_file lib_spacing_rules.tcl.enc \
-physical_design_path ./design.def.gz \
-physical_constraint_file va.tcl \
-log_file ./lef_def.log

Identify violations
report_constraints -all -max_cap -max_tran; report_noise; report_timing; report_power
```

SPEF with location for on route buffering

LEF/DEF files read in during fix\_eco\_timing

11-22

If the design contains filler cells, you can have PrimeTime ignore them. Make sure you first identify the filler cell masters:

```
set_eco_options -filler_cell_names "FILLER1 FILLER2 FILLER4 ..."
```

Then set the following variable:

```
set_app_var eco_allow_filler_cells_as_open_sites true
```

## Example Physically Aware ECO in PT (2/2)

```
Power recovery
fix_eco_power -methods {size_cell | remove_buffer}

Fix DRC violations
set buffer_list { BUF_1 BUF_2 ... BUF_N}
fix_eco_drc -type max_tran -physical_mode open_site -buffer_list $buffer_list
fix_eco_drc -type max_cap -physical_mode open_site -buffer_list $buffer_list

Fix noise violations
fix_eco_drc -type noise ...

Fix setup and hold violations
fix_eco_timing -type setup ...
fix_eco_timing -type hold ...

Write changes
write_changes -format icctcl -output pt_eco.tcl
.....
Implement PT ECOs in ICC II; Repeat above steps as necessary.
After timing closure, return to PT for final leakage power recovery
fix_eco_power ...
write_changes -format icctcl -output pt_eco_fslr.tcl
```

Physical mode:  
open\_site |  
occupied\_site |  
freeze\_silicon

11-23

The `-physical_mode` option of the `fix_eco*` commands has the following settings:

- `none` (the default) - Does not use physical data while fixing violations.
- `open_site` - Sizes a cell if there is enough room available around the cell and inserts a buffer if there is an empty site. This mode avoids moving existing cells; it places cells only in empty sites.
- `occupied_site` - *PrimeTime* considers layout density in the cell neighborhood and can place or resize a cell overlapping existing neighbor cells when the local placement density can accommodate the change. This mode places or resizes a cell even if no empty site is available; to implement the change list, you need a place-and-route tool to move existing cells to create room for the new or sized cell.
- `freeze_silicon` - This mode is meant for use with ICC II only. It inserts a buffer directly on top of a base layer, matching programmable spare cells specified by `set_eco_options`. This buffer will appear in the changelist file as an `insert_buffer` or an `add_buffer_on_route` command output with location. The exact location of the buffer in the programmable spare cell will be guided by the layout and the net topology. Each such buffer will have a corresponding ICC II `map_freeze_silicon` command output in the changelist file as well. This command will be used by ICC II to program the spare cell into the mapped buffer. When the programmable spare cell is wider than the buffer, the `map_freeze_silicon` command will backfill the left-over space inside it with base layers that match smaller programmable spare cells. *PrimeTime* `freeze_silicon` physical ECO guidance will ensure that when such a backfill is required, the buffer location leaves room for the left over space to fit one or more smaller programmable spare cells specified by `set_eco_options`.

Consider using the `occupied_site` physical mode early in the ECO flow, while you're expecting to perform multiple iterations: This allows the most flexibility for fixing violations, relying on ICC II to make room, as needed. Once you are close to timing closure and tape-out, you can minimize any additional changes to the design by switching to `open_site`.

The `-format icctcl` option of with `write_changes` is applicable to both *ICC* and *ICC II*.

## Review of Unit Objectives



**Having completed this unit, you should be able to:**

- **Describe our recommended ECO flow that automates:**
  1. Reducing dynamic and leakage power consumption and
  2. Fixing of DRC, Noise, Setup and Hold Violations
- **Write out an ECO change list for implementing the changes**
- **List additional input requirements for performing physically aware ECO**
- **Differentiate the *open\_site* vs. *occupied\_site* modes during physically aware ECO**

11-24

## **Appendix**

### **Additional Information on ECO What if Analysis**

11-25

## Limiting changes during ECO

### ■ **fix\_eco\_timing -type setup**

- Resizes using library cells in the linked libraries
- Minimize change to netlist
  - Limit the cell size increase for better closure with P&R
    - set eco\_alternative\_area\_ratio\_threshold <float>
    - Default = 0 (unlimited area increase)
    - 1 = in-place optimization (change cell within existing area limit)
    - 2 = good value in general. It allows area increase by 2X the original cell size
  - set eco\_alternative\_cell\_attribute\_restrictions
  - Specifies lib\_cell attributes used to restrict cell sizing

```
pt_shell> define_user_attribute -classes lib_cell -type string family
pt_shell> set eco_alternative_cell_attribute_restrictions "area family"
```

11-26

## Restricting ECO fix with Path Control

```
fix_eco_timing
[-slack_lesser_than slack_limit float]
[-slack_greater_than slack_limit float]
[-group group]
[-from from_list]
[-to to_list]
[-current_library]
[-setup margin setup_margin_value]
[-hold_margin hold_margin_value]
```

- Path selection options are similar to those in `report_timing`
- Control fixing scope
  - `-group` supports lists and wildcards
  - `-slack_greater_than` can be used to avoid spending time on large failures caused by misconstrained paths
  - `-setup_margin/hold_margin` can be used to fix setup and hold with positive finite margin.
  - `-current_library` can be used to limit the search of alternative cells to current cells library only.

11-27

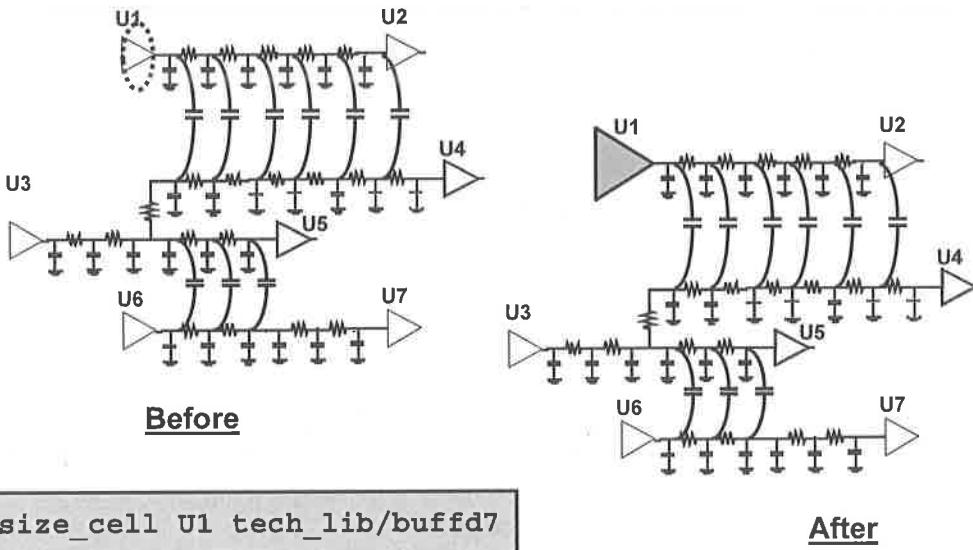
By default, `fix_eco_timing` addresses all violations in the design. Using these options user can specify more narrow path sets for the ECO.

`-group` to limit the fixing for the timing paths belong to groups.

`-slack_lesser/greater`

`-to` or `-from` to focus the fixing to violating paths to endpoint or startpoint.

## Example Manual ECO – Size a Cell



11-28

pt shell> man size\_cell

### ARGUMENTS

`cell_list` Specifies a list of leaf cells to be relinked. Each cell must be in scope (at or below the current instance).

`lib_cell` Specifies the name of the library cell to which the specified cells are to be linked.

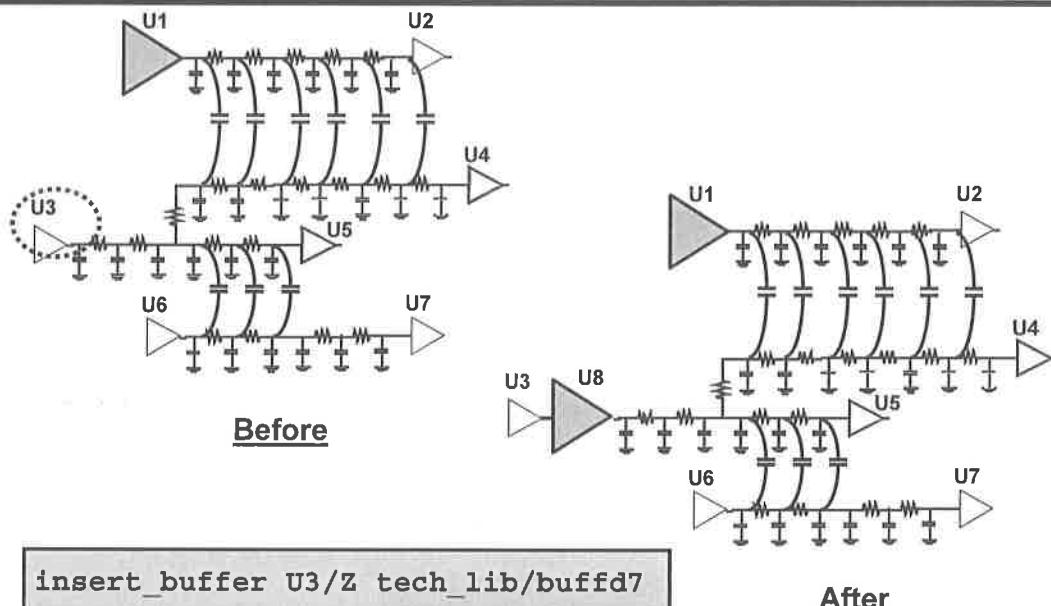
### DESCRIPTION

The `size cell` command changes the drive strength (or other properties) of a leaf cell by relinking it to a new library cell that has the required properties. Like all other netlist editing commands, for `size cell` to succeed, all of its arguments must succeed. If any arguments fail, the netlist remains unchanged. `size cell` returns a 1 if successful and a 0 if unsuccessful. Each cell in `cell_list` must be in scope; that is, at or below the current instance.

The `lib cell` that is being swapped in must conform to the following restrictions:

- o `lib_cell` must be functionally compatible with the current library cell to which the cells in `cell_list` are linked.
- o `lib_cell` cannot be the same as the current library cell of any of the cells in `cell_list`.
- o `lib_cell` must have the same pin-count and pin directions as the current library cell of the cells in `cell_list`.
- o Currently, the pins of `lib_cell` must be in the same order as the current library cell of the cells in `cell_list`.

## Example Manual ECO – Insert a Buffer at a Driver

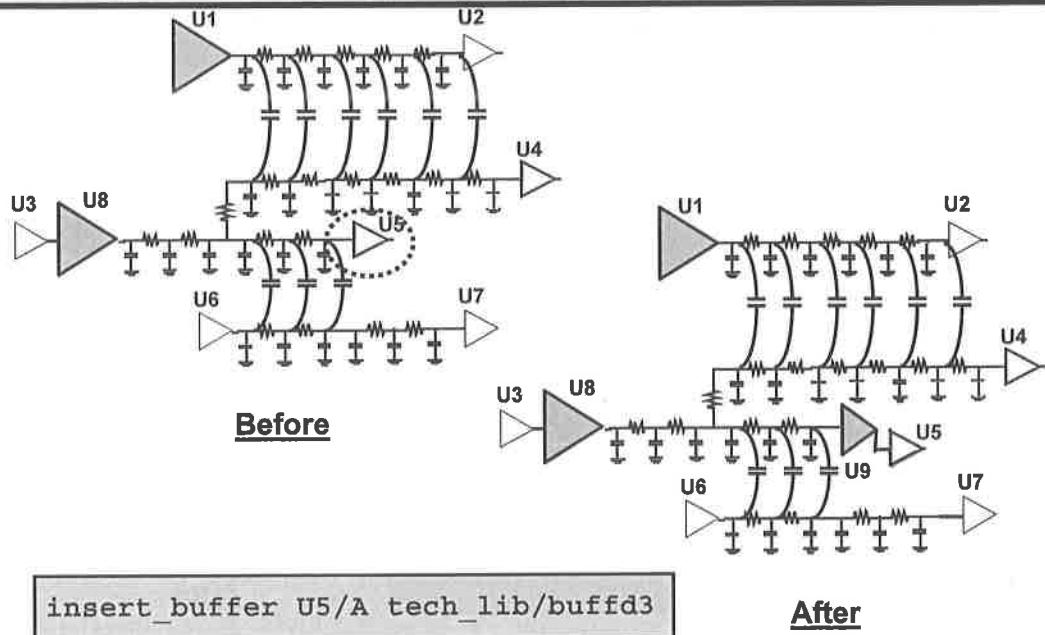


11-29

```
pt_shell> insert_buffer -help
Usage:
insert_buffer # Insert a buffer at pin(s)
 [-inverter_pair] (Insert a pair of inverting lib_cell)
 pin_or_port_list (List of pins or ports to buffer)
 lib_cell (Library cell to use as buffer)
```

```
pt_shell> remove_buffer -help
Usage:
remove_buffer # Remove buffers from the current design
 cell_list (List of buffers to remove)
```

## Example ECO – Insert a Buffer at a Load



11-30

## Manually Fixing a Select Path: Setup

**FAST!**  
**No timing update!**

```
pt_shell> report_timing
s4_mul_189_mult_mult_U1070/ZN (invbd4) 0.1268 & 2.3433 r
s4_mul_189_mult_mult_U1070ASTipolInst106/Z (bufbd1) 0.4108 & 2.7541 r
s4_mul_189_mult_mult_U115/ZN (nr02d0) 0.4249 & 3.1790 f
```

**Problem Cell**

```
pt_shell> estimate_eco -max s4_mul_189_mult_mult_U1070ASTipolInst106 -sort_by slack
```

| delay type : max_rise     | lib cell | area   | stage_delay | arrival | slack   |
|---------------------------|----------|--------|-------------|---------|---------|
| cb13fs120_tsmc_max/bufbdk |          | 6.2500 | 0.2532      | 2.6091  | 0.3265  |
| cb13fs120_tsmc_max/dl04d4 |          | 5.0000 | 7.3420      | 9.6943  | -6.8097 |

| delay type : max_fall     | lib cell | area   | stage_delay | arrival | slack   |
|---------------------------|----------|--------|-------------|---------|---------|
| cb13fs120_tsmc_max/bufbdk |          | 6.2500 | 0.2249      | 2.5389  | 0.4785  |
| cb13fs120_tsmc_max/dl04d4 |          | 5.0000 | 7.2236      | 9.5342  | -6.5681 |

```
pt_shell> size_cell I_ORCA_TOP/I_RISC_CORE/I_ALU/U391 cb13fs120_tsmc_max/bufbdk
pt_shell> update_timing
```

11-31

## Manually Fixing a Select Path: Hold

```
pt_shell> report_timing -delay_type min
clock SDRAM_CLK (rise edge) 0.0000 0.0000
clock network delay (ideal) 0.0000 0.0000
I_ORCA_TOP/I_SDRAM_READ_FIFO/count_int_reg[5]1/CP (sdcrq1) 0.0000 0.0000 r
I_ORCA_TOP/I_SDRAM_READ_FIFO/count_int_reg[5]1/Q (sdcrq1) 0.3428 & 0.3428 f
I_ORCA_TOP/I_SDRAM_READ_FIFO/SD_RFIFO_RAM/A1[5] (ram32x64) 0.0147 & 0.3576 f
data arrival time

pt_shell> estimate_eco-type insert_buffer \
 -lib_cells [get_alternative_lib_cells cb13fs120_tsmc_max/bufbd2] \
 I_ORCA_TOP/I_SDRAM_READ_FIFO/SD_RFIFO_RAM/A1[5] Path
 EndPoint

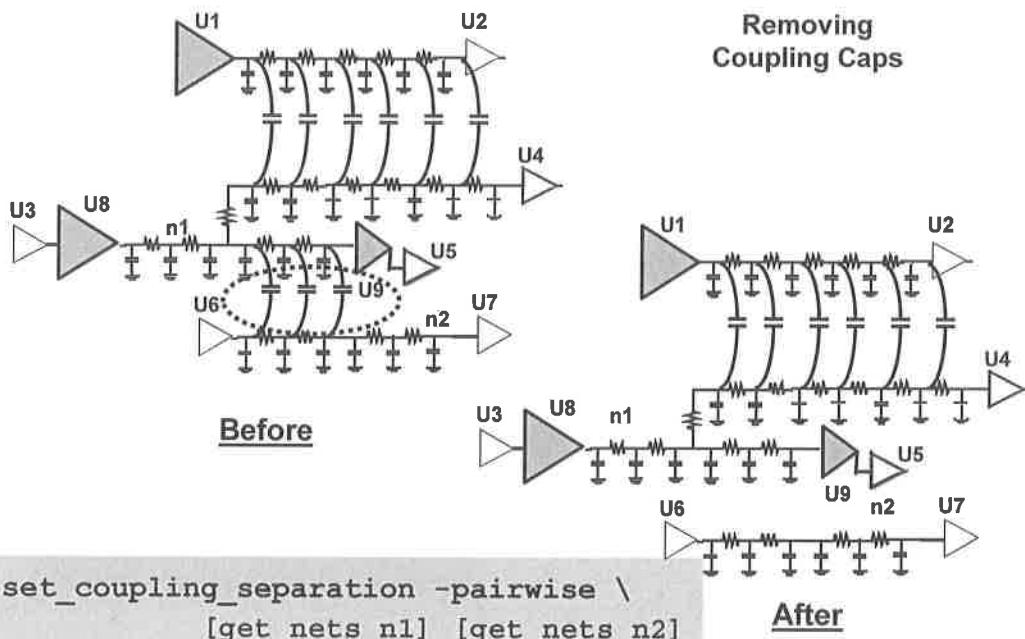
delay type : max_rise
lib cell area stage_delay arrival slack
cb13io320_tsmc_max/pt3o01 3426.1848 3.7382 3.7382 3.4990
cb13fs120_tsmc_max/bufbdk 6.2500 0.5925 0.5925 6.9533
cb13fs120_tsmc_max/dl04d4 5.0000 7.5219 7.5219 0.0082
.

.

pt_shell> insert_buffer I_ORCA_TOP/I_SDRAM_READ_FIFO/SD_RFIFO_RAM/A1[5] \
 cb13fs120_tsmc_max/bufbdk
pt_shell> update_timing
pt_shell> report_timing
Iterate if necessary
```

11- 32

## What-If Analysis for Crosstalk



11-33

```
pt_shell> remove_coupling_separation -help
Usage:
remove_coupling_separation # Remove separation constraint on nets
 [-pairwise pnets] (List of nets that have pairwise constraints with nets)
 nets (List of nets)
```

## Other Manual Netlist Editing Commands

report\_lib  
get\_alternative\_lib\_cells

size\_cell  
create\_cell  
rename\_cell  
remove\_cell  
swap\_cell

create\_net  
connect\_net  
disconnect\_net  
rename\_net  
remove\_net

insert\_buffer  
remove\_buffer

set\_coupling\_separation  
remove\_coupling\_separation

11-34

See the `write_changes` man page for lists of which of these commands can be written out for ICC.

## ECO Fixing Flow

1. Power recovery by cell sizing and buffer removal
2. Fix DRC/Noise violations by inserting buffers or resizing cells.
3. Fix setup violations by resizing cells
4. Fix hold violations by inserting buffers and/or resizing cells
5. Leakage recovery by threshold voltage swapping
6. Write change list for implementation tool

| Change List                                                                                                                            |                                                                                                        |
|----------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------|
| Text output: History of "What-if" Tests                                                                                                | Tcl output: Compressed Result of "What-if Tests"                                                       |
| <pre>insert_buffer U1/A BUFX1 size_cell U3 BUFX2 size_cell U3 BUFX4 remove_buffer U3 insert_buffer U2/A BUFX1 size_cell U4 BUFX2</pre> | <p>U3 is inserted,<br/>resized twice,<br/>and removed</p> <p>U4 is inserted,<br/>then resized once</p> |

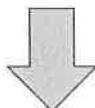
11-35

## Optional Tailoring: Library Names

- To handle cases where ICC/ICClI links against different logical library names than PrimeTime
- Have PrimeTime to remove lib name that prepends lib\_cell references

```
insert_buffer [get_pins {U2320/Y}] slow/BUFX4 \
 -new_net_names {net1} -new_cell_names {U2}
```

Default  
change\_list



Specify before  
modifying netlist

```
set eco_write_changes_prepend_libname_to_libcell false
fix_eco_timing -type setup
```

```
insert_buffer [get_pins {U2320/Y}] BUFX4 \
 -new_net_names {net1} -new_cell_names {U2}
```

Resulting  
change\_list

11-36

## Optional Tailoring: Buffer Names

- Hold fixing inserts cells and nets into netlist
- To specify prefixes for ECO net and cell names:

```
pt_shell> set eco_instance_name_prefix {Ueco2_}
Ueco2_
pt_shell> insert_buffer U2/Y slow/BUFX2
Information: Inserted 'Ueco2_1' at 'U2/Y'. (NED-046)
{"Ueco2_1"}
```

**Specify before modifying netlist  
or running fix\_eco commands**

```
pt_shell> set eco_net_name_prefix {NETeco2_}
NETeco2_
pt_shell> insert_buffer U2/Y slow/BUFX2
pt_shell> Information: Inserted 'U1' at 'U2/Y'. (NED-046)
 {"U1"}
pt_shell> all_connected [get_pins U1/Z]
 {"NETeco2_1"}
```

11-37

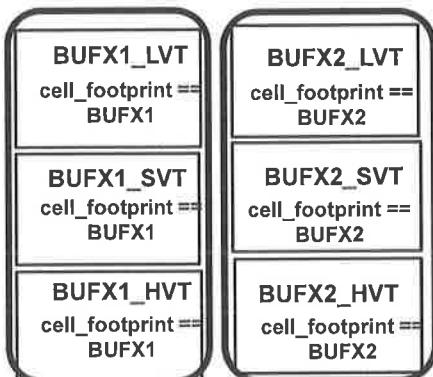
Insert\_buffer is the command used by fix\_eco\_timing to insert a buffer – it is also the command that appears in the file of changes generated by the write\_changes command.

## Optional Guidance: Sizing Cells

- Both manual and automatic fixes require selecting an “alternative cell” when resizing
- Restrict alternatives to same attribute value as ‘problem cell’

```
set eco_alternative_cell_attribute_restrictions{attr_list}
```

```
set eco_alternative_cell_attribute_restrictions "cell_footprint"
get_alternative_lib_cells slow/BUFX1_LVT
```



What cells will be returned by the above example?

- a. All logically-equivalent cells with the attribute “cell\_footprint”  
b. All logically-equivalent cells having the same “cell\_footprint” value (BUFX1) as BUFX1\_LVT

11- 38

Answer: b. All logically\_equivalent cells having the same ‘cell\_footprint’ value (BUFX1) as BUFX1\_LVT

# Using User Synthesis Library Attributes

```
pt_shell> set eco_alternative_cell_attribute_restrictions "cell_footprint"
Error: The variable eco_alternative_cell_attribute_restrictions cannot be
set to 'cell_footprint' (NED-048)
Error: can't set "eco_alternative_cell_attribute_restrictions":
 Use error_info for more info. (CMD-013)
```

```
pt_shell> list_attributes -application -class lib_cell
always_on lib_cell boolean A
area lib_cell float A
base_name lib_cell string A
disable_timing lib_cell boolean A
```

**Problem:**  
**The attribute is missing!**

```
define_user_attribute -import -class lib_cell -type string cell_footprint
link
```

## Solution

```
pt_shell> set eco_alternative_cell_attribute_restrictions "cell_footprint"
cell_footprint ←
```

```
pt_shell> list_attributes -application -class lib_cell
always_on lib_cell boolean A
area lib_cell float A
base_name lib_cell string A
cell_footprint lib_cell string U,I
```

**The attribute is now available**

11-39

What do you do if the cell\_footprint is not in your library?

## User Interface for Leakage Recovery

|                                                |                                                                                              |
|------------------------------------------------|----------------------------------------------------------------------------------------------|
| <b>fix_eco_power</b>                           | Improves leakage quality by swapping library cells                                           |
| <b>-pattern_priority list</b>                  | Specifies the library cell name patterns in the order of highest priority to lowest priority |
| <b>[-cell_type combinational   sequential]</b> | Specifies combinational or sequential cell type for optimization, by default both are done   |
| <b>[-attribute string]</b>                     | Specifies the library cell attribute for attribute-based swapping                            |
| <b>[-setup_margin]</b>                         | Control timing margin for leakage recovery                                                   |
| <b>[-verbose]</b>                              | Shows detailed timing information                                                            |
| <b>report_cell_usage</b>                       | Reports the cell usage                                                                       |
| <b>-pattern_priority list</b>                  | Specifies the Pattern priority list                                                          |
| <b>[-attribute string]</b>                     | Specifies the library cell attribute                                                         |
| <b>[-alternative_lib_cells]</b>                | Shows alternative library cells                                                              |
| <b>[-show_others]</b>                          | Shows other cells that don't match patterns                                                  |

11-40

# Agenda

DAY

3

8 Signal Integrity: Crosstalk Delay Analysis



9 Signal Integrity: Crosstalk Noise Analysis



10 Correlation: POCV and AWP Analysis

11 Timing Closure: ECO/What If Analysis

12 Large Data: DMSA and Hyperscale Analysis

13 Conclusion

12-1

## Unit Objectives



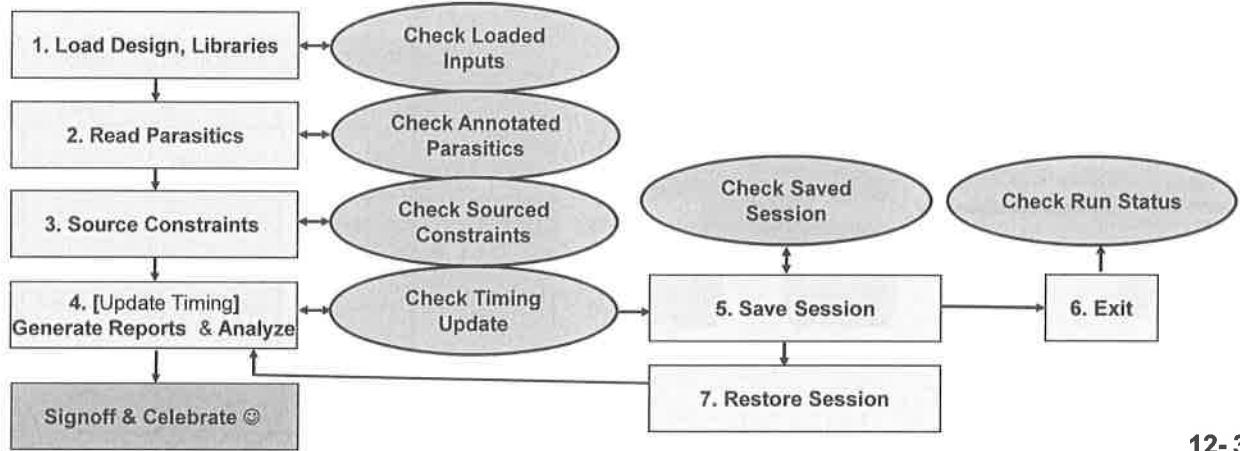
After completing this unit, you should be able to:

- **Setup and Invoke PT in DMSA mode**
- **Generate Merged DMSA reports**
- **Using Top level context, generate a Hyperscale block model**
- **Describe using the Hyperscale analysis flow to perform**
  - Hierarchical STA
  - ETM/ILM Replacement flow and
  - Top down flow

12-2

# Timing Analysis Flow in PrimeTime

- STA flow is divided into several steps
  - Steps 1-3 read data and
  - Analysis begins at Step-4
- The STA flow is repeated until signoff is achieved

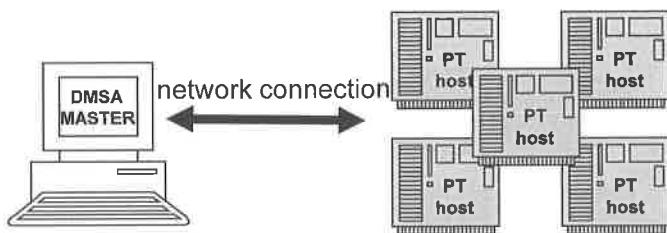


12-3

STA: Static Timing Analysis

# What Is Distributed Multi-Scenario Analysis (DMSA)?

- DMSA provides efficient unified analysis of multiple PrimeTime analyses (or scenarios) from a single *master* PrimeTime
  - We can work with multiple analyses as easily as with a single PrimeTime analysis run!



|                              | Functional mode | Scan shift mode | Scan capture mode | JTAG mode    |
|------------------------------|-----------------|-----------------|-------------------|--------------|
| worst-case<br>(125°C, 1.35V) | Scenario #1     | Scenario #2     | Scenario #3       | Scenario #4  |
| typical<br>(25°C, 1.50V)     | Scenario #5     | Scenario #6     | Scenario #7       | Scenario #8  |
| best-case<br>(0°C, 1.65V)    | Scenario #9     | Scenario #10    | Scenario #11      | Scenario #12 |

12-4

The goal of DMSA is to allow us to work with multiple analyses as easily as we work with a single PrimeTime analysis run.

The two primary causes of multiple STA analyses are multiple operating modes, and multiple analysis corners:

A design typically has multiple operating *modes*

Multiple functional modes

Multiple test modes

A design must reliably operate across

its specified range of voltage/temperature (VT) conditions

the range of possible process/interconnect (P) conditions

A specific set of conditions (environmental VT and manufacturing P) is called a *corner*

Consider a design with four operating modes, analyzed at three PVT corners

This would result in 12 scenarios as shown below

## Invoking a DMSA Master Process

- A master process is invoked with the `-multi_scenario` switch:

```
pt_shell -multi_scenario
```

- A master `pt_shell` is different than a normal `pt_shell`

- different commands
- different warnings/errors
- different manpages

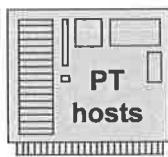
12-5

The `pt_shell_mode` variable specifies the current PrimeTime operating mode

```
determine which mode we are in
switch $pt_shell_mode {
 primetime {
 echo "Normal PrimeTime session"
 }
 primetime_master {
 echo "Multi-scenario master session"
 }
 primetime_slave {
 echo "Multi-scenario slave session"
 }
}
```

## Setting Up DMSA

- To set up a DMSA run, three key components must be configured:
  - Number of licenses
  - Number of remote (host) processes
  - Scenario definitions
- These are discussed in detail in the DMSA resources on SolvNET (some details are shown in *Appendix* at the end of this unit)



12- 6

To set up a DMSA run, three key components must be configured. We need to tell DMSA how many licenses it can use, how many remote host processes it can use, and how the different scenarios are defined. Let's take a look at how each of these are configured.

## Defining Scenario Data

- To define scenarios, three types of data must be provided

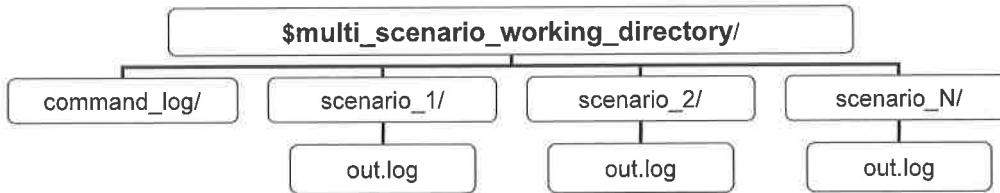
- *global* data
  - ◆ Data common to all scenarios
  - ◆ Includes netlist, global variables/settings, etc.
- *common* data
  - ◆ Data which can be used to group together scenarios
  - ◆ Runtime-intensive actions are typically put here, such as SPEF or SDF annotation files
- *specific* data
  - ◆ Scenario-specific data such as scenario specific variable settings

```
Specifying Common Data and Specific Data
foreach corner {bc tc wc} {
 foreach mode {func jtag atpg_capture atpg_shift} {
 create_scenario \
 -name ${mode}_${corner} \
 -common_data "lib_${corner}.tcl read_${corner}_spef.tcl" \
 -specific_data "cons_${mode}.tcl"
 }
}
```

12-7

## Scenario Output Logs

- During distributed analysis, each scenario's output is stored in the scenario's "out.log" file:



- The working directory defaults to "." (current directory)
- These logfiles can be used to check for errors/warnings or to debug problems

12-8

Once the distributed analysis is running, each scenario's output is stored in the scenario's "out.log" file in the multi-scenario working directory. The working directory defaults to a directory named "work" in your current directory. It would be overwhelming and confusing if all scenarios were simultaneously dumping their output to the terminal. You can use these scenario log files to check for warnings and errors or to debug problems.

## **remote\_execute**

- **remote\_execute can be used to directly execute any valid PrimeTime commands at the scenarios:**

```
pt_shell> remote_execute {set_false_path -from [all_inputs] }
1
pt_shell> remote_execute {
? set si_xtalk_delay_analysis_mode all_path_edges
? update_timing
?
}
1
pt_shell> report_timing -slack_lesser_than 0
```

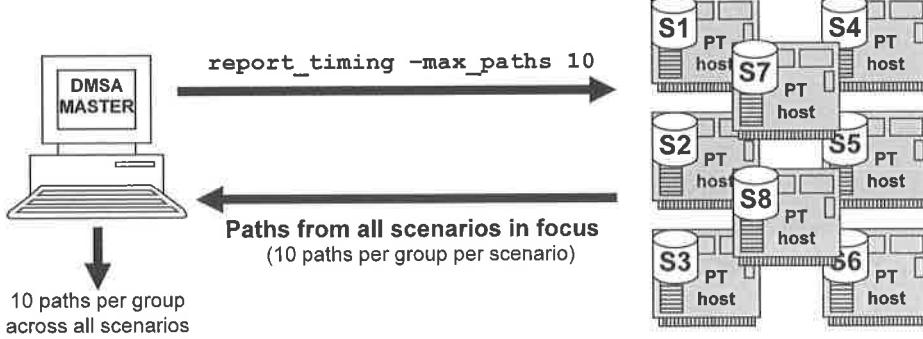
- **By default, remote\_execute output goes only to the separate scenario log files**
  - Use "**remote\_execute -verbose**" to see the output on-screen

12-9

In addition to reporting, there is a `remote_execute` command which can be used to directly execute any valid PrimeTime commands at the scenarios. For example, maybe we want to apply a false path from all inputs to disable some interface paths in our scenarios. Or maybe we want to set the SI enhanced alignment mode to `all_violating_paths` and force an `update_timing`. Just about anything we can do in normal PrimeTime, we can tell the scenarios to do directly with the `remote_execute` command.

## Merged report\_timing

- DMSA automatically merges the results across all scenarios, saving us from sifting through hundreds of nearly identical reports
  - Topological duplicates are automatically removed
- To see the 10 worst timing paths in each path group across all scenarios, we simply issue:



12-10

In DMSA, if we wanted to see the ten worst timing paths in each path group but across all scenarios, we simply type the same command we always would – “`report_timing -max_paths 10`”. The command is passed to the scenarios and they all provide their results back to the master. The master then merges all of this information together, showing us the worst 10 paths per path group, across all scenarios. During the merging, topological duplicates are automatically removed. What this means is that if multiple scenarios have the same timing path through the same exact set of cells and pins, only the path with the worst slack is kept.

An example merged summary report from DMSA

```
pt_shell> report_timing -path_type end -max_paths 10 -delay min
```

| Endpoint<br>Scenario                    | Path Delay | Path Required | Slack |    |
|-----------------------------------------|------------|---------------|-------|----|
| Busy_IRQ_sync2_reg/D (DFQD1)            | 0.68 f     | 0.75          | -0.07 | wc |
| RxAbortSync4_reg/D (DFCNQD1)            | 0.72 f     | 0.78          | -0.06 | wc |
| ethreg1/SetTxCIrq_sync3_reg/D (DFCNQD1) | 0.41 f     | 0.47          | -0.05 | bc |
| SendControlFrame_sync3_reg/D (DFCNQD1)  | 0.57 f     | 0.62          | -0.05 | tc |
| WriteRxDataToFifoSync3_reg/D (DFCNQD1)  | 0.58 f     | 0.63          | -0.05 | tc |
| miim1/RStat_q2_reg/D (DFCND1)           | 0.41 f     | 0.46          | -0.05 | bc |
| SyncRxStartFrm_q_reg/D (DFCND1)         | 0.73 f     | 0.78          | -0.05 | wc |
| Busy_IRQ_sync3_reg/D (DFQD1)            | 0.70 f     | 0.74          | -0.05 | wc |
| WriteRxDataToFifoSync2_reg/D (DFCND1)   | 0.58 f     | 0.63          | -0.05 | tc |
| miim1/EndBusy_reg/D (DFCND1)            | 0.41 f     | 0.46          | -0.05 | wc |

## Merged report\_constraint

- Merged report\_constraint shows the worst constraint behavior across all scenarios
  - A merged max\_transition DRC report is shown below:

| Pin                        | Scenario | Required Transition | Actual Transition | Slack  |            |
|----------------------------|----------|---------------------|-------------------|--------|------------|
| RxStatusInLatched_reg[2]/E | wc       | 1.001               | 1.026             | -0.025 | (VIOLATED) |
| LatchedRxLength_reg[11]/E  | wc       | 1.001               | 1.026             | -0.025 | (VIOLATED) |
| RxStatusInLatched_reg[5]/E | wc       | 1.001               | 1.025             | -0.024 | (VIOLATED) |
| rx_fifo/U201/B2            | bc       | 0.554               | 0.571             | -0.017 | (VIOLATED) |
| rx_fifo/U197/B2            | bc       | 0.554               | 0.571             | -0.017 | (VIOLATED) |
| rx_fifo/U210/B2            | bc       | 0.554               | 0.570             | -0.016 | (VIOLATED) |
| rx_fifo/U202/B2            | bc       | 0.554               | 0.570             | -0.016 | (VIOLATED) |
| LatchedRxLength_reg[5]/E   | tc       | 0.801               | 0.794             | -0.007 | (VIOLATED) |
| LoadRxStatus_reg/Q         | tc       | 0.801               | 0.794             | -0.007 | (VIOLATED) |

12-11

A merged report\_constraint command is also available which shows the worst constraint behavior across all scenarios. Report\_constraint has many different types of reports. In the example below, we can see a merged max\_transition DRC report. Again, the report looks just like a normal max\_transition report, except that we can now see the scenario where each violation exists.

## Merged report\_analysis\_coverage

- Merged report\_analysis\_coverage shows whether constraint arcs are met, violated or untested across all scenarios

```
Scenarios: func_min, jtag_min, atpg_capture_min, atpg_shift_min, func_typ,
 jtag_typ, atpg_capture_typ, atpg_shift_typ, func_max, jtag_max,
 atpg_capture_max, atpg_shift_max
```

| Type of Check   | Total  | Met           | Violated   | Untested   |
|-----------------|--------|---------------|------------|------------|
| setup           | 42186  | 42178 (100%)  | 0 ( 0%)    | 8 ( 0%)    |
| hold            | 42186  | 40790 ( 97%)  | 1388 ( 3%) | 8 ( 0%)    |
| recovery        | 1284   | 1282 (100%)   | 0 ( 0%)    | 2 ( 0%)    |
| removal         | 1284   | 1282 (100%)   | 0 ( 0%)    | 2 ( 0%)    |
| min_pulse_width | 23190  | 21906 ( 94%)  | 0 ( 0%)    | 1284 ( 6%) |
| out_setup       | 348    | 270 ( 78%)    | 78 ( 22%)  | 0 ( 0%)    |
| out_hold        | 348    | 348 (100%)    | 0 ( 0%)    | 0 ( 0%)    |
| All Checks      | 110826 | 108056 ( 98%) | 1466 ( 1%) | 1304 ( 1%) |

12-12

The merged report\_analysis\_coverage command shows whether constraint arcs such as setup and hold arcs are met, violated or untested across all scenarios. The merged report\_analysis\_coverage summary report looks like a normal report, except that the percentages represent the met, violated and untested check coverages across all scenarios.

All of the reporting commands can be invoked with remote\_execute in the slaves

|                          |                                    |
|--------------------------|------------------------------------|
| report_analysis_coverage | # Shows coverage of timing checks  |
| report_clock_timing      | # Report clock timing info         |
| report_constraint        | # Report constraint evaluations    |
| report_global_timing     | # Show top level timing summary    |
| report_power             | # Display power report             |
| report_si_bottleneck     | # Report crosstalk bottleneck nets |
| report_timing            | # Report timing paths              |

## DMSA Resources on SolvNET

**014511: "Example Distributed  
Multi-Scenario Analysis Design"**

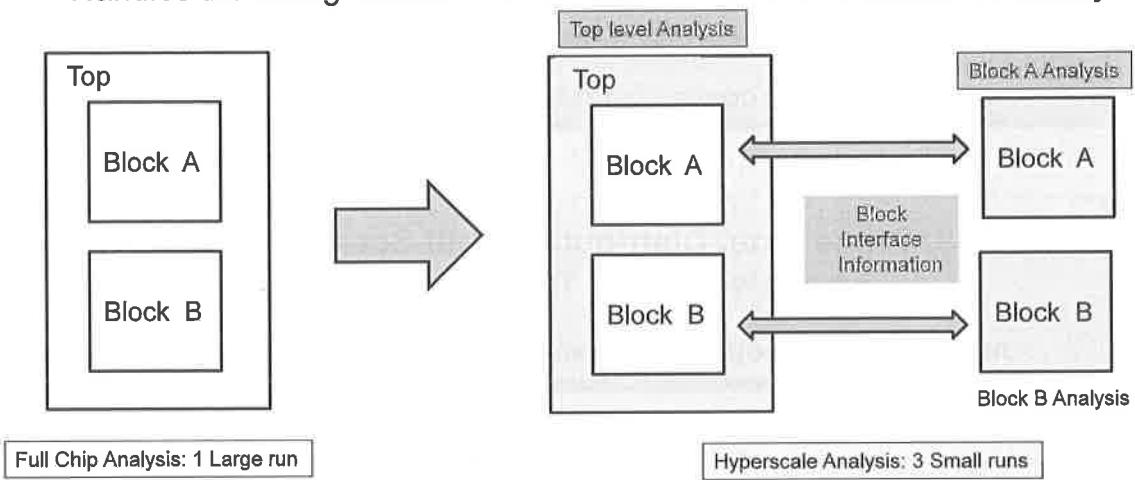
<http://solvnet.synopsys.com/retrieve/014511.html>

**031169: PrimeTime: Distributed Multi-Scenario  
Analysis (DMSA) Training**

<http://solvnet.synopsys.com/retrieve/031169.html>

## Hyperscale Analysis Introduction

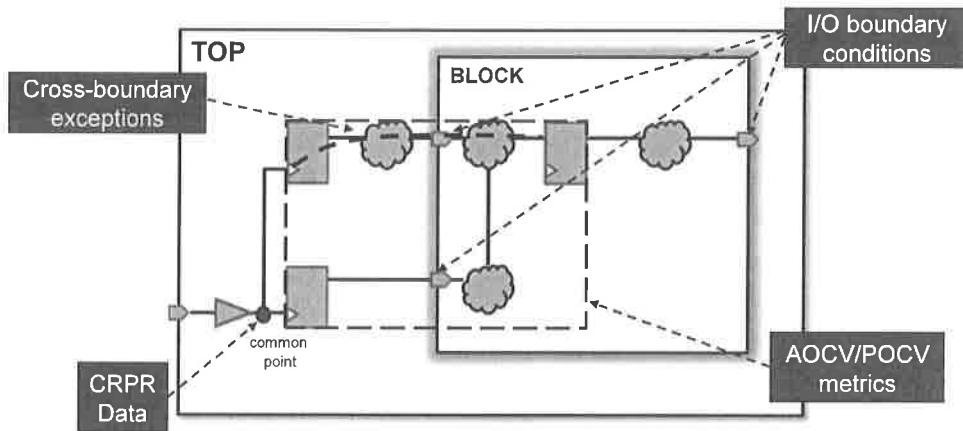
- HyperScale hierarchical analysis method analyzes the block-level and top-level portions of the design using separate runs
  - Handles the timing interfaces across hierarchical boundaries accurately



12-14

# Hyperscale Top Level Context

- **HyperScale context captures actual timing information at block boundary**
  - Input/output boundary conditions
  - Non-SDC information such as CRPR, POCV, cross-block exceptions etc.



12-15

HyperScale context captures the actual timing information at the block boundary which enables accurate block level analysis consistent with flat. In other words, it allows you to have an accurate full chip view at the block level.

It contains I/O boundary conditions, CRPR data, cross-boundary exceptions, as well as AOCV/POCV metrics.

## Multiply Instantiated Model (MIM) Analysis

If a design contains multiply instantiated modules (MIMs), you can optionally run just one analysis of the module and use the same results for multiple instances of that block at the top level

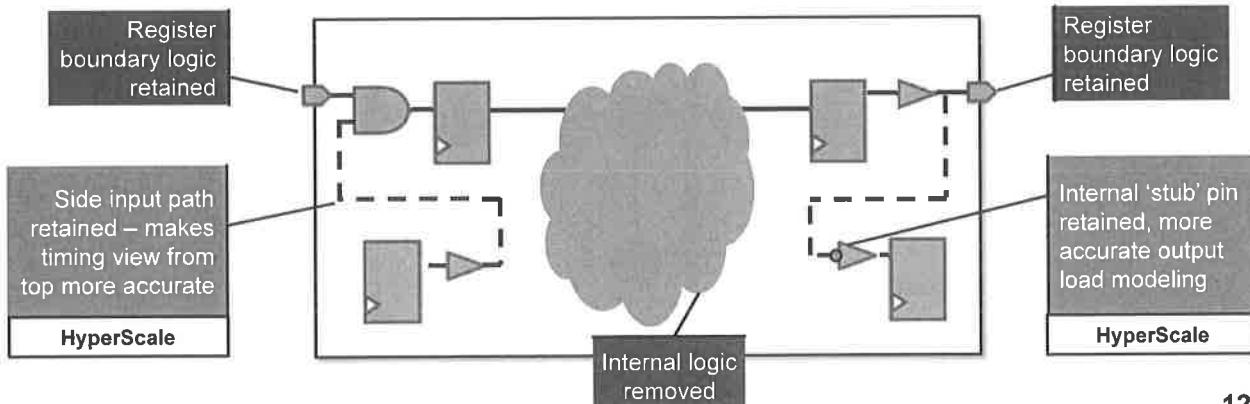
Where different constraints apply to different instances of the module, the tool “merges” the constraints, choosing the most restrictive constraints among all instances, so that the single block-level analysis results can be used for all instances of the module. An instance in an unusual and distinct environment can be handled separately, if needed.

For ECO fixing, the merged context can be applied to all MIM instances so fixing is performed only once for all the instances.

# Hyperscale Block Model

## ■ Conceptually similar, yet, more accurate than Interface Logic Model (ILM)

- Register to register paths are removed
  - ◆ model is significantly more compact than block netlist – analysis is efficient
- Side input path and stub pin are retained – analysis is accurate



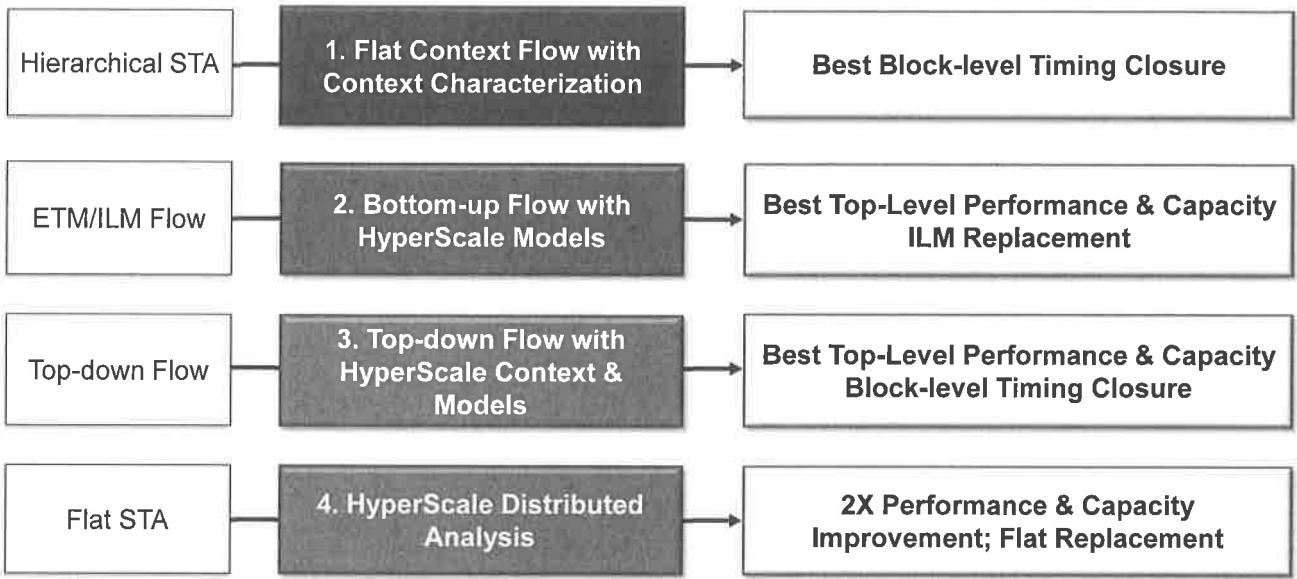
12-16

### An Example creating a Hyperscale model

```
set_app_var hier_enable_analysis true # Enable hierarchical analysis
read_verilog block_A.v # Read in the block
link_design BLK_A

...
update_timing # Run timing analysis
update_noise # Run noise analysis if applicable
...
write_hier_data $blkModel # Write out hierarchical data
```

# Transitioning to Hyperscale Analysis From ...



We shall discuss the first 3 flows in this workshop.

12-17

You can easily transition to HyperScale regardless of the flow you are currently using for sign-off.

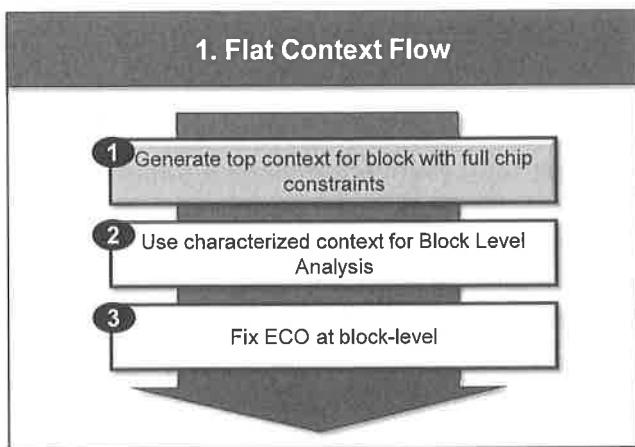
If you are presently doing flat sign-off, you can take advantage of HyperScale's automated partitioning and distribution technology to perform your analysis faster with smaller memory footprint.

If your design has already driven you to hierarchical implementation such as ETM/ILM flow, you can use HyperScale's bottom-up single-pass flow. This gives you faster TAT with improved accuracy.

An existing HyperScale bottom up block closure flow can be extended to take advantage of the top context. The top context enables the block to have a full chip view. And with this flow, you get to close your blocks faster without potentially leaving performance on the table – in contrast to an ETM/ILM flow.

The engine behind HyperScale's partitioning and distribution technology allows you to move between flows. It enables you to efficiently perform analysis on all or just part of the design.

## Flow #1: Flat Context flow -- Context Characterization



```
read_verilog ...
link_design
read_parasitics ...
read_sdc

characterize_context -block {BLKA}

update_timing
report_context -nosplit [get_cells BLKA_I1]

write_context -format gbc {BLKA} \
 -output BLKA_context

write_context -format ptsh {BLKA} \
 -output BLKA_cons.tcl
```

12-18

## Flow #1: Context Reporting

```
pt_shell> report_context [get_cell BLKA_I1]

Design : top/BLKA_I1 (BLKA)

Clocks Waveforms:
Attributes:
 p - propagated_clock
 v - virtual clock

Clock Orig_Name Period Waveform Attrs
Sources

CLK_1 CLK_1 80.00 {0 2.5} p {CLK_1}
CLK_2 CLK_2 80.00 {0 2.5} p, v {}

pt_shell> report_context

Design : BLKA

Clocks Waveforms:
Attributes:
 p - propagated_clock
 v - virtual clock

Clock Orig_Name Period Waveform Attrs
Sources

CLK_1 CLK_1 80.00 {0 2.5} p {CLK_1}
CLK_2 CLK_2 80.00 {0 2.5} p, v {}
```

Reporting binary context of  
block at top level

Reporting binary context of  
block at block level

12-19

## Flow #1: Block Level Analysis with Characterized Context

```
read_verilog ...
link_design
read_parasitics ...
source BLKA_cons.tcl
set_dont_override [get_ports input_[1]]
read_context ./BLKA_context
update_timing
report_context -nosplit
get_attribute [current_design] is_context_loaded

get_attribute [get_port input_[1]]\\
is_user_dont_override

get_attribute [current_design] context_timestamp
```

2 Use characterized context for Block Level Analysis

Apply block-level constraints before read\_context

Optional: Prevent read\_context from overriding the original user-defined constraints on specified ports

Read binary context before update\_timing

Report context info

Check if context loaded successfully

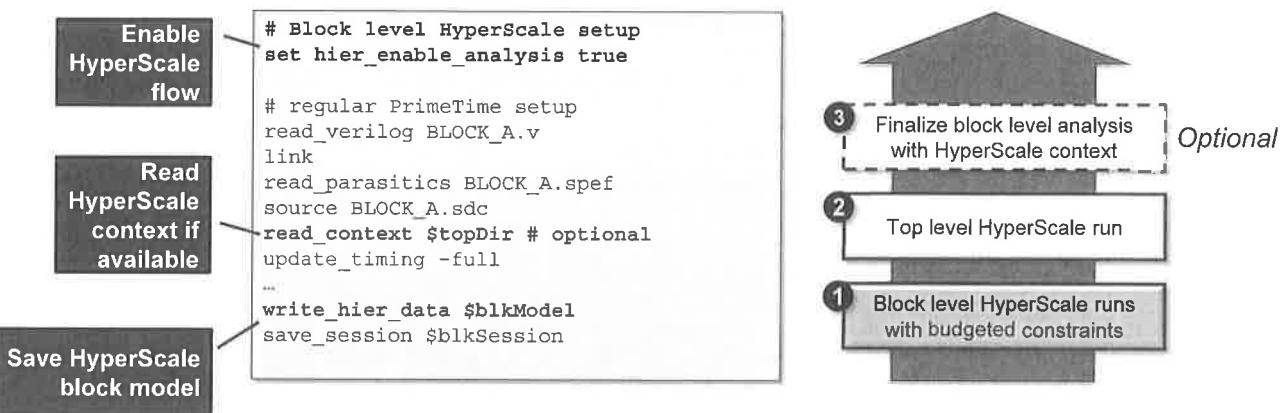
Check if user-specified override is set on port

Print context creation time

- To ensure block & top-level constraint consistency, write\_context ptsh format output can be used instead of original block-level constraints
- If using original constraints, clock mapping issue may be found and need to be resolved (covered later)

12- 20

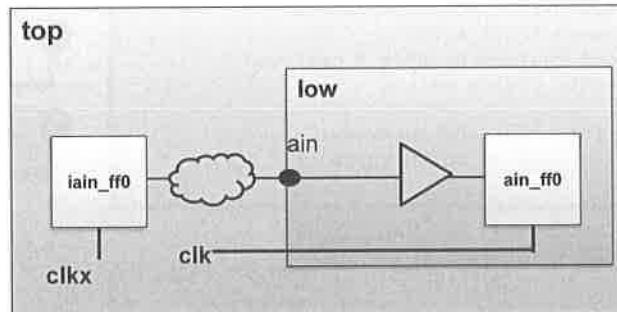
## Flow #2: Bottom Up Flow with Hyperscale Models



12-21

## Flow #2: Clock Mapping for Hyperscale Analysis

- HyperScale does timing analysis on partial paths when a complete path is split across different hierarchies
- Clocks are used to automatically stitch paths and perform accurate timing analysis across hierarchies



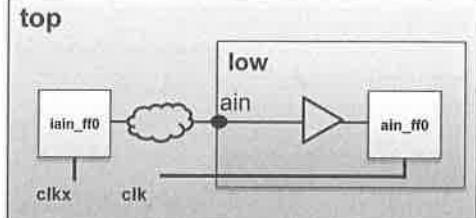
12-22

## Example Clock Mapping Issue : Full Chip Analysis is OK

```
report_timing -through low_1/ain[0]
```

Startpoint: iain\_ff0 (rising edge-triggered flip-flop clocked by clkx)  
 Endpoint: low\_1/ain\_ff0 (rising edge-triggered flip-flop clocked by clk)  
 Path Group: clk  
 Path Type: max

| Point                            | Incr  | Path   |
|----------------------------------|-------|--------|
| clock clkx (rise edge)           | 0.00  | 0.00   |
| clock network delay (propagated) | 0.00  | 0.00   |
| iain_ff0/CP (FD1)                | 0.00  | 0.00 f |
| iain_ff0/Q (FD1)                 | 0.09  | 0.09 f |
| ain_buf1/A (BUF)                 | 0.00  | 0.09 f |
| ain_buf1/Z (BUF)                 | 0.04  | 0.13 f |
| low_1/ain[0] (low) <-            | 0.00  | 0.13   |
| low_1/lobuf0/A (BUF)             | 0.00  | 0.13 f |
| low_1/lobuf0/Z (BUF)             | 0.04  | 0.17 f |
| low_1/ain_ff0/D (FD3)            | 0.00  | 0.17 f |
| data arrival time                |       | 0.17   |
|                                  |       |        |
| clock clk (rise edge)            | 5.00  | 5.00   |
| clock network delay (propagated) | 0.30  | 5.30   |
| clock reconvergence pessimism    | 0.00  | 5.30   |
| low_1/ain_ff0/CP (FD3)           |       | 5.30 f |
| library setup time               | -0.14 | 5.17   |
| data required time               |       | 5.17   |
|                                  |       |        |
| data required time               |       | 5.17   |
| data arrival time                |       | 5.17   |
|                                  |       |        |
| slack (NET)                      |       | 5.00   |



- From the full chip timing report, the arrival at the “ain” port of the block is associated with clock “clkx”
- This is as expected

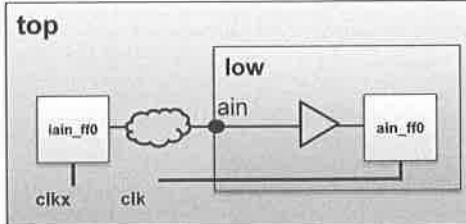
## Example Clock Mapping Issue : Clock Mapping Failed!

```
report_clock -map
```

```

Report : clock_map
Design : top
...

Instance Top level clock Block level
clock clkx clk
low_1 clkx (N/A)
clk clk clock
```



### ■ Top Level clock mapping failed!

12-24

PrimeTime offers several utilities to ensure block/top clock definition consistency and facilitate an efficient bottom-up flow

To show the clocks that are relevant to the given hierarchical cells (blocks)

```
report_clock -cells
```

To Show the map between top level clock and block level clock

```
report_clock -map
```

To define the mapping between top and block level clocks

```
set_clock_map
```

## Example Clock Mapping Issue : Unconstrained Block Path!

```
report_timing -through ain[0] -exceptions all
```

```
Startpoint: ain[0] (input port)
Endpoint: ain_ff0 (rising edge-triggered flip-flop clocked by clock)
Path Group: (none)
Path Type: max
Point Incr Path

input external delay 0.00 0.00 f
ain[0] (in) <- 0.00 @ 0.00 f
1obuf0/A (BUF) 0.00 @ 0.00 f
1obuf0/Z (BUF) 0.04 0.04 f
ain_ff0/D (FD3) 0.00 0.04 f
data arrival time 0.04

```

(Path is unconstrained)

The dominant exceptions are:  
None

The overridden exceptions are:  
None

The unconstrained reasons (except for false path) are:

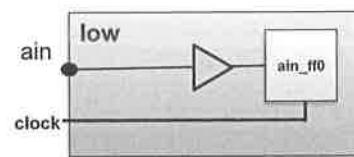
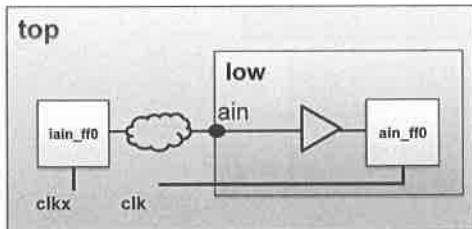
| Reason          | Startpoint | Endpoint |
|-----------------|------------|----------|
| no_launch_clock | ain[0]     | -        |

HyperScale  
annotations

The top level context  
cannot be annotated at the  
"ain" port of the block,  
causing the path to become  
unconstrained

12-25

## Example Clock Mapping Issue : Solution



```
Virtual clock to fix clock mapping issue
create_clock -period 10 -name clkx
```

```

Report : clock_map
Design : top

Instance Top level clock Block level clock
low_1 clkx clkx
 clk clock
```

Creating a virtual clock with the same name for the block level analysis resolves the clock mapping issue between block and top – Clock mapping is now established

12-26

## Example Clock Mapping Issue : Block Level Analysis is OK

```
report_timing -from ain[0]
```

| Point                            | Incr   | Path   |
|----------------------------------|--------|--------|
| clock clkx (rise edge)           | 0.00   | 0.00   |
| clock network delay (ideal)      | 0.00   | 0.00   |
| input external delay             | 0.11 @ | 0.13 f |
| ain[0] (in) <-                   | 0.00 @ | 0.13 f |
| 1obuf0/A (BUF)                   | 0.00 @ | 0.13 f |
| 1obuf0/Z (BUF)                   | 0.04   | 0.17 f |
| ain_ff0/D (FD3)                  | 0.00   | 0.17 f |
| data arrival time                |        | 0.17   |
| clock clock (rise edge)          | 5.00   | 5.00   |
| clock network delay (propagated) | 0.27   | 5.27   |
| clock reconvergence pessimism    | 0.00   | 5.27   |
| ain_ff0/CP (FD3)                 | -0.14  | 5.13   |
| library setup time               |        | 5.13   |
| data required time               |        | 5.13   |
| data required time               | 5.13   |        |
| data arrival time                | -0.17  |        |
| block (NET)                      | 4.97   |        |

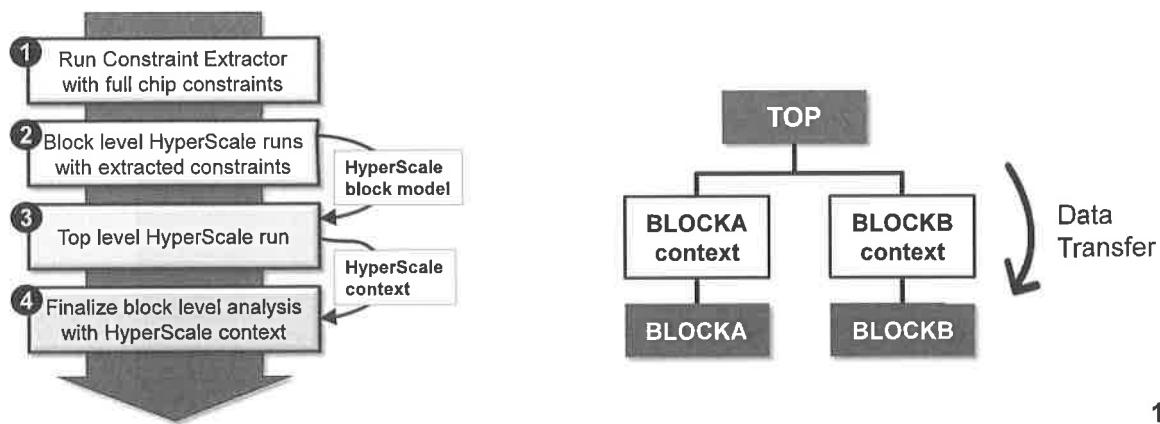
Top level context is now properly annotated

HyperScale annotations

Path is now constrained

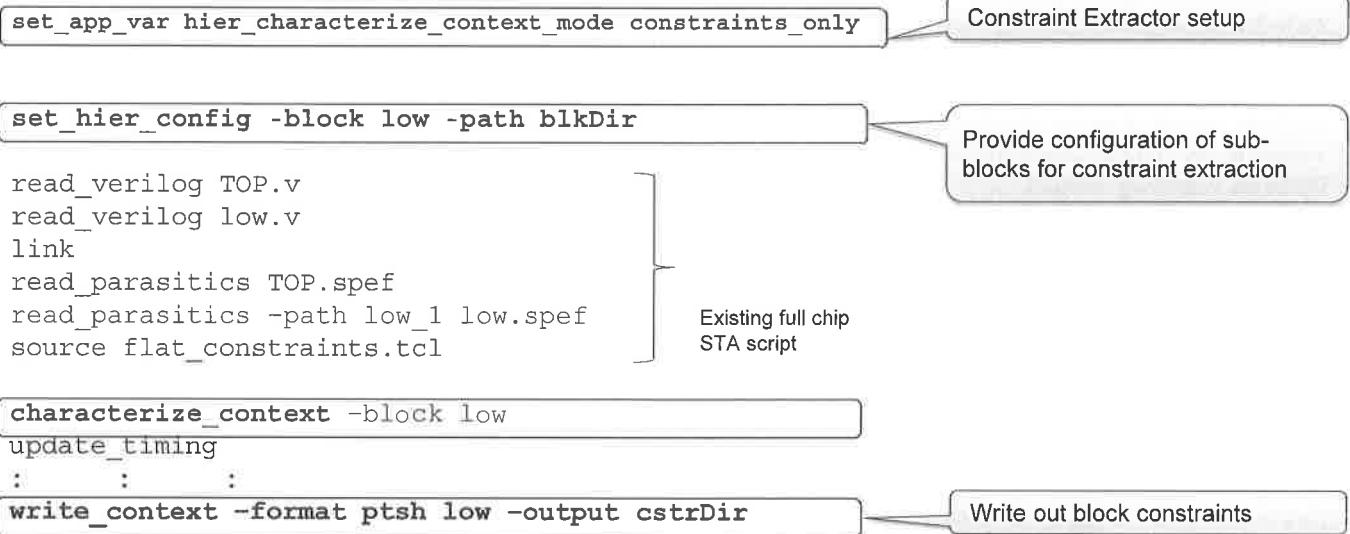
## Flow #3: Top Down Flow with Constraint Extraction

- Top down flow is recommended when full chip constraints are available / the reference
  - Enables the fastest timing closure
  - block level constraints are guaranteed to be consistent with full chip constraints

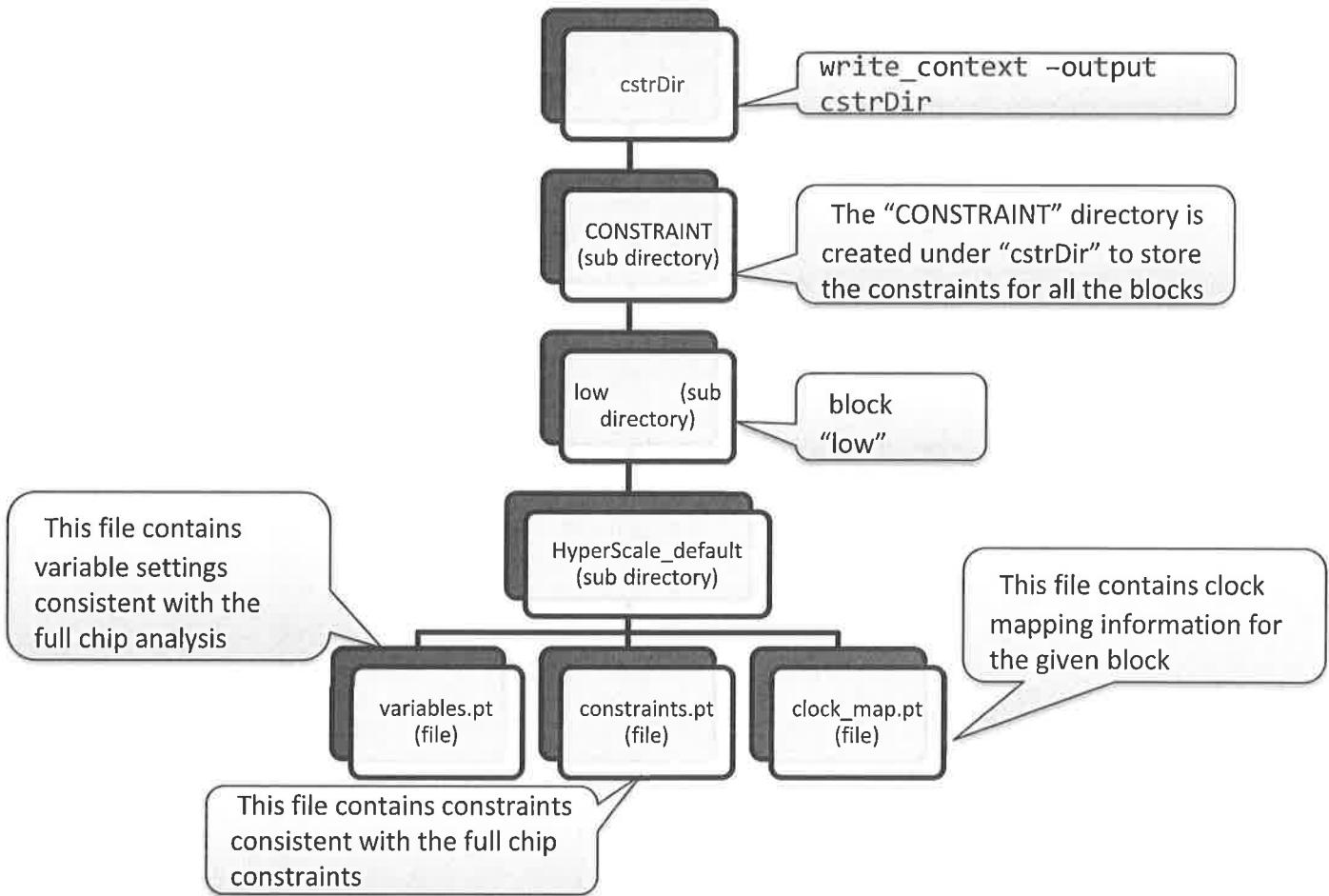


12-28

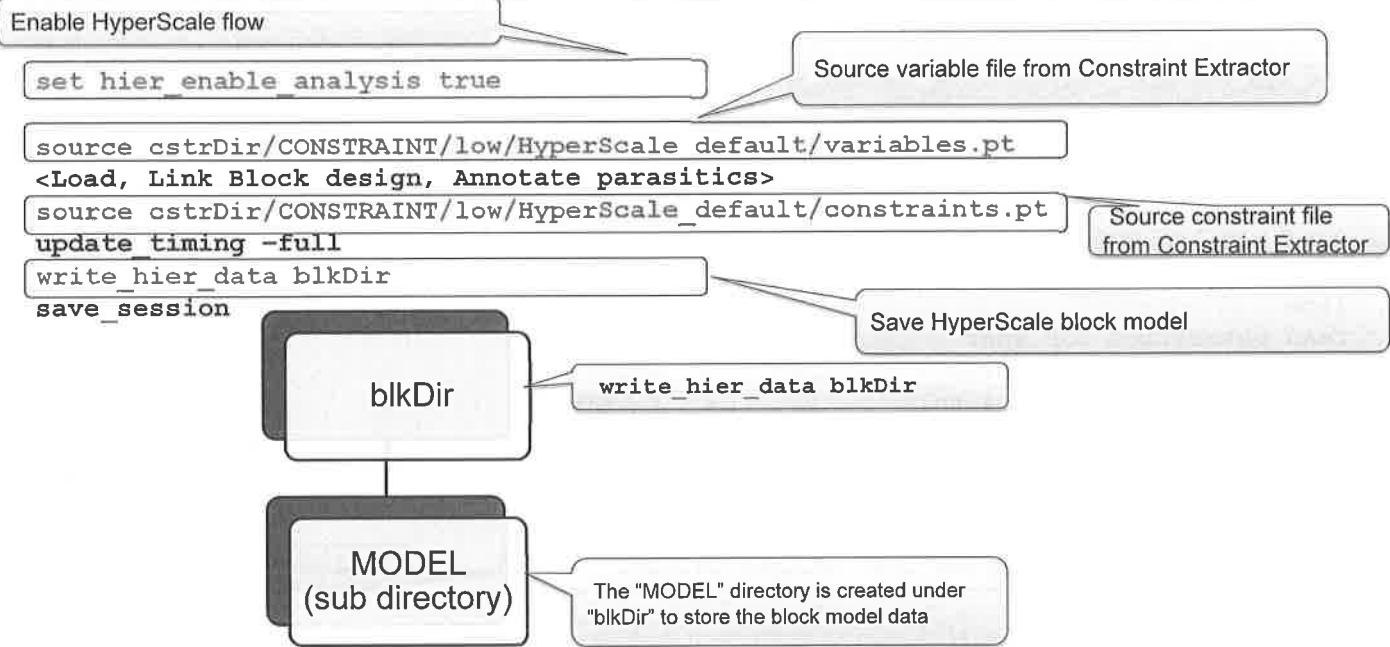
## Flow #3 Step-1: Run HyperScale Constraint Extractor



12-29



## Flow #3 Step-2: Initial Block Level Analysis



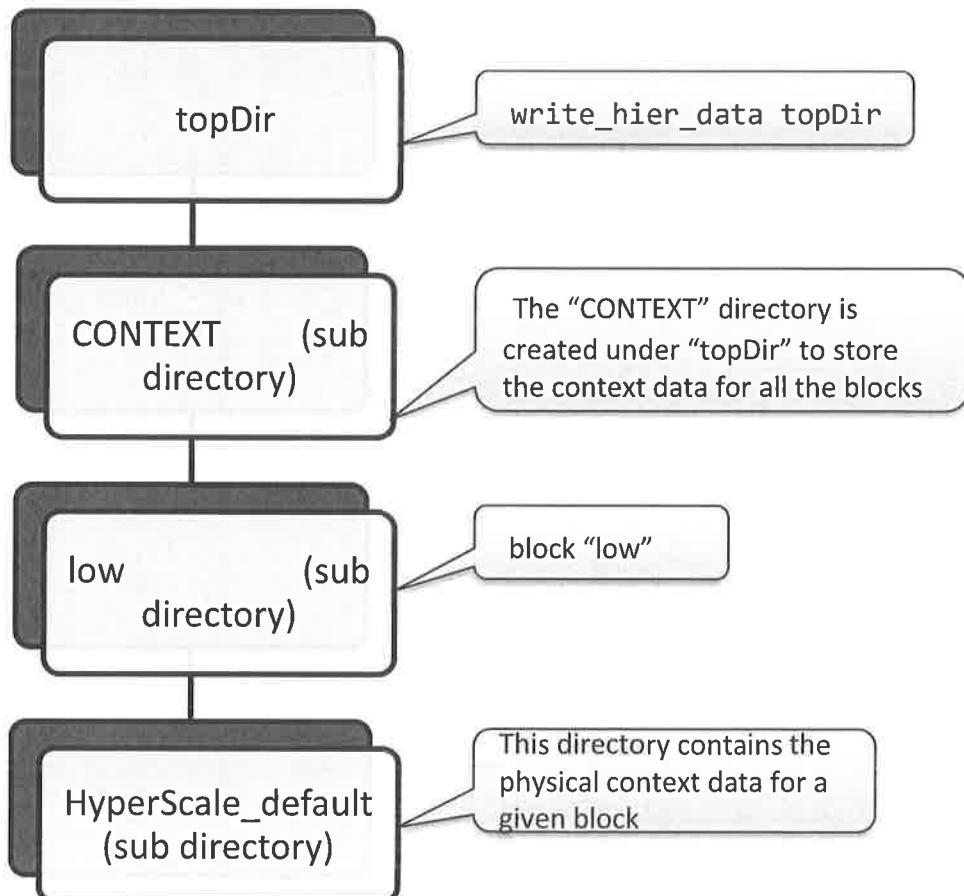
12-30

## Flow #3 Step-3: Top Level Analysis

```
set hier_enable_analysis true Enable HyperScale flow
set hier_config -block low -path blkDir Provide configuration of sub-blocks
 that are used in top level
read_verilog TOP.v
read_verilog low.v
link
read_parasitics TOP.spf
read_parasitics -path low_1 low.spf
source flat_constraints.tcl
update_timing -full
: : :
write_hier_data topDir Generate context data
save_session
```

Full chip STA script

12- 31



## Flow #3 Step-4: Finalize Block Level Analysis

```
set hier_enable_analysis true Enable HyperScale flow
source cstrDir/CONSTRAINT/low/HyperScale_default/variables.pt
read_verilog low.v
link
read_parasitics low.spf
source cstrDir/CONSTRAINT/low/HyperScale_default/constraints.pt
read_context topDir
update_timing -full
: : :
: : : Load top level context
Source constraint file from Constraint Extractor
write_hier_data blkDir2
save_session
Save new HyperScale block data if needed
```

12- 32

## Timing Report in HyperScale with Top Level Context

```
report_timing -from data[1]
```

```
Startpoint: data[1] (input port clocked by clock)
Endpoint: ffil (rising edge-triggered flip-flop clocked by clock)
Path Group: clock
Path Type: max

Point Incr Path
-----+-----+-----+
clock clock (rise edge) 0.00 0.00
clock network delay (propagated) 0.00 0.00
input external delay 0.34 @ 0.34 r
data[1] (in) 0.00 @ 0.34 x
and1/A (AND2) 0.00 @ 0.34 x
and1/Z (AND2) 0.06 & 0.41 r
ff11/D (FD3) 0.00 & 0.41 r
data arrival time 0.41
 0.41

clock clock (rise edge) 5.00 5.00
clock source latency 0.25 @ 5.25
clock (in) 0.00 @ 5.25 r
cbufi/A (BUF) 0.00 @ 5.25 r
cbufi/2 (BUF) 0.11 & 5.38 r
ff11/CP (FD3) 0.02 & 5.40 r
clock reconvergence pessimism 0.00 5.40
clock uncertainty -0.05 5.35
library setup time -0.15 5.20
data required time 0.00 5.20
 5.20
data required time 5.20
data arrival time 0.41
 0.41

slack (MET) 4.79
```

- HyperScale annotations added to report\_timing (@ sign)
- IO constraints & values overwritten by top-level context
- Top level context enables accurate full chip timing view at the block

12-33

## Hyperscale Training on SolvNET

**2617311: PrimeTime M-2016.12 HyperScale Training**

<http://solvnet.synopsys.com/retrieve/2617311.html>

**2815130: PrimeTime N-2017.12 Update Training**

→ **Module-3: Hierarchical Analysis**

<http://solvnet.synopsys.com/retrieve/2815130.html>

12- 34

## Review of Unit Objectives



**Having completed this unit, you should be able to:**

- **Setup and Invoke PT in DMSA mode**
- **Generate Merged DMSA reports**
- **Using Top level context, generate a hyperscale block model**
- **Describe using the hyperscale analysis flow to perform**
  - Hierarchical STA
  - ETM/ILM Replacement flow and
  - Top down flow

## **Appendix**

**Additional Information on DMSA**

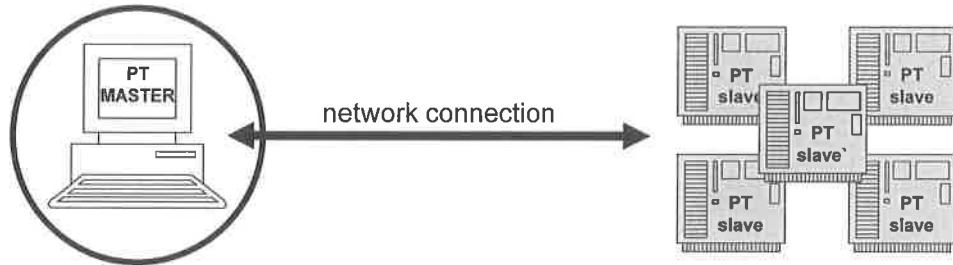
**Additional Information on Hyperscale Analysis**

**12- 36**

## Terminology : Master

### ■ What is a *master*?

- It is a special PrimeTime process running in a master mode which controls the slave processes
- It allocates license and machine resources for the slaves, issues commands to the slaves, and collects their data
- It provides the user with a single interface that allows analysis across all scenarios



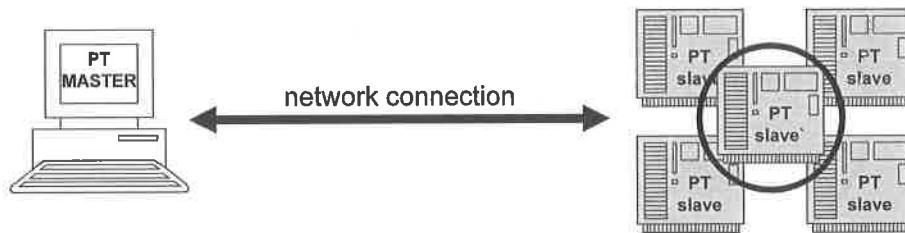
12- 37

The master session can be run interactively or as a batch process.

## Terminology : Slave

### ■ What is a slave?

- It is a remote PrimeTime or PrimeTime SI process that is controlled by the master
  - ◆ Therefore, a slave is also referred to as a *remote process*
- It is created by the master, and is given tasks to complete by the master
- At any given time, a slave process can be analyzing only a single scenario



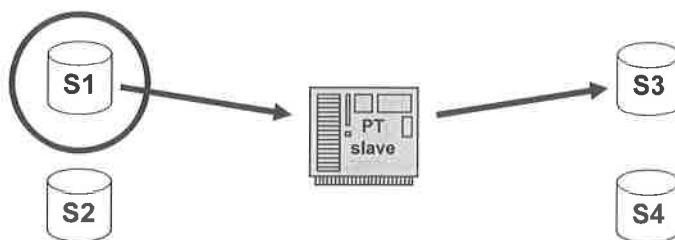
12- 38

Slaves are never run directly by the user; they are always started and controlled by the master. Slaves are never interactive.

## Terminology : Scenario

### ■ What is a *scenario*?

- Think of a scenario as a single PrimeTime or PrimeTime SI analysis that you would normally run
- The master's job is to take one or more scenarios, and run them on the remote processes
- Scenarios can be swapped from a slave process's memory to disk by the master if needed



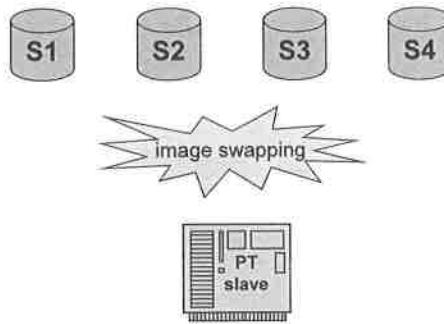
12- 39

A scenario is basically any PrimeTime or PrimeTime SI run. You could have a mix of scenarios, some of which are SI runs, and some of which are non-SI runs. Anything can be changed across the scenarios – constraints, PrimeTime variable settings, SI filtering settings, clocks, annotation files, etc. The only restriction is that they all share the same netlist.

## Terminology : Image

### ■ What is an *image*?

- An image is a special `save_session` format for scenarios
  - ◆ Scenario images allows scenarios to be swapped to disk
- You cannot do a `restore_session` on these automatically-generated scenario images



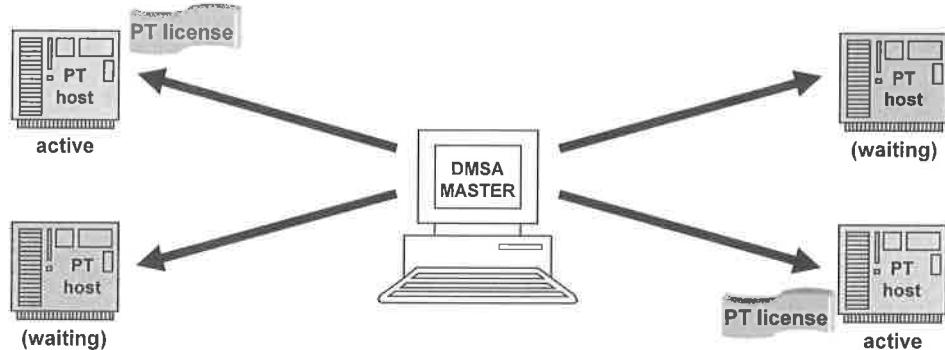
12-40

The images used by multi-scenario analysis are similar to those created by `save_session/restore_session`, but the format is tailored to multi-scenario. They cannot be manually restored in a normal PrimeTime by the user. However, the user can still save a normal session from one or more scenarios.

# License Management

## ■ When (# licenses) < (# remote host processes):

- The master distributes licenses to the remote host
- When a task finishes at a remote host, the license is freed to be immediately used by another host
- Remote host processes are not terminated to free license



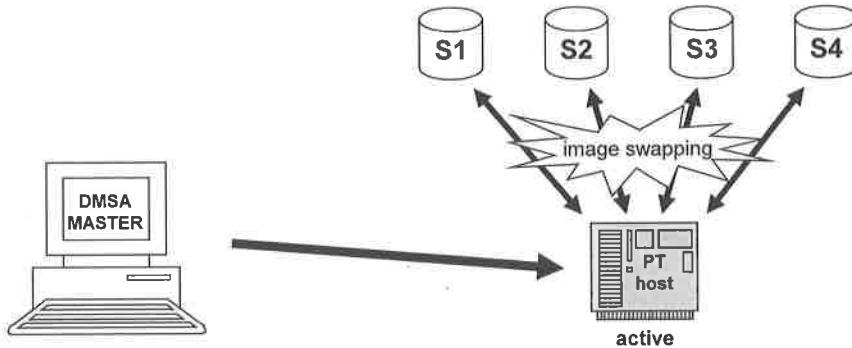
12- 41

Now that we've defined our licenses, compute hosts and scenarios, how do these all relate to each other? Let's look at what happens when you have fewer licenses than host processes. In DMSA, the master acquires all the licenses from the license server, up to the maximum you've allowed it, then distributes the licenses to the remote hosts. In this particular case, we clearly don't have enough licenses to allow all host processes to be active at once. When DMSA needs to perform some task, it will enable as many host processes as it can with the licenses it has. The rest of the host processes will be queued up waiting for a license. As each active host process completes its task, the master takes back that license and gives it to the next host process in the waiting queue. This process of reallocating the license from one host to another takes a fraction of a second, pretty much in zero time. The important thing to realize here is that we do not need to terminate a PrimeTime process because it has no available license. This is a major differentiator between normal PrimeTime and DMSA. With normal PrimeTime, there is no concept of putting it to sleep to free up the license – you have to quit out of it complete, losing your analysis unless it's saved to a session. But in DMSA, the master can move the licenses around in zero-time as needed, waking and sleeping each process as needed, and allowing everything to remain in memory. Effectively, the licenses are time-multiplexed across the scenarios with almost no switching overhead.

# Machine Management

## ■ When (# remote processes) < (# scenarios):

- Scenario images must be swapped to disk to execute the task across all scenarios
  - ◆ This is expensive in terms of disk activity and runtime
- This should be avoided unless absolutely necessary!

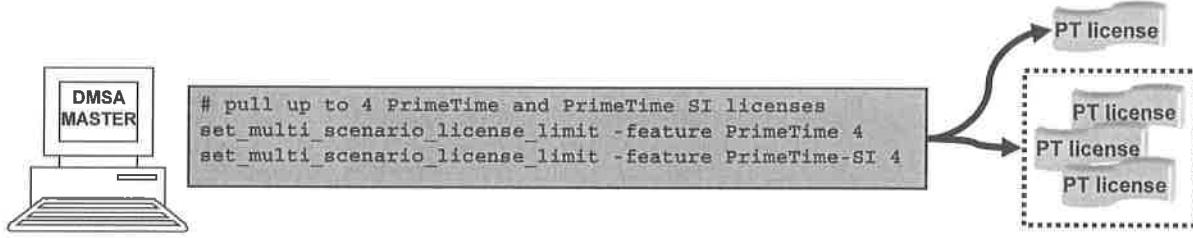


12- 42

Another situation is when you have fewer remote host processes than scenarios. In this case, it's simply impossible for every scenario to be loaded at the same time across the available host processes. In this case, when DMSA needs to perform some task, some scenario images must be swapped to disk to execute the task across all scenarios. This is obviously expensive in terms of disk activity and runtime. For each and every individual command you issue like report\_timing or report\_constraint, you will end up having to wait for scenarios to be saved and restored to allow the scenario images to cycle through the limited number of available hosts. Clearly this should be avoided unless absolutely necessary!

# Setting Up DMSA: Configuring Licenses

- **Specify DMSA the upper limit for the number of licenses to use**
  - A minimum of one is needed to do any work
  - Additional licenses are pulled from the license server as needed
  - The master does not require an extra license
- **In the example below:**
  - Allow DMSA up to 4 PrimeTime and PrimeTime-SI licenses

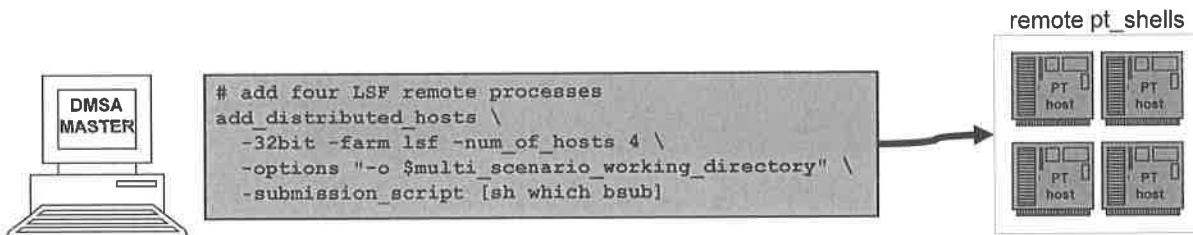


12- 43

To configure licenses in DMSA, we give it the upper limit for the number of licenses to use. A minimum of one license is needed to do any work, of course. But additional licenses are only pulled from the license server as needed. We could give DMSA a hundred licenses, but if it only needs three licenses for three scenarios, it will only pull three licenses. The master PrimeTime process does not require an additional license; you only need licenses up to the number of scenarios you wish to analyze. In the example below, we allow DMSA up to four PrimeTime and PrimeTime-SI licenses. DMSA will immediately check out one license to get up and running, then can pull up to three from the license server more as needed.

# Setting Up DMSA: Configuring Remote Processes

- Provide DMSA information on the remote processes to invoke
  - Compute farms are supported (LSF, GRD, proprietary)
  - Discrete named machines can be given
  - Multiple commands can be issued (and they are additive)
  - A mix of platforms (linux, sparc, 32/64-bit, etc.) can be used
- In the example below:
  - We ask DMSA to invoke 4 processes on an LSF farm

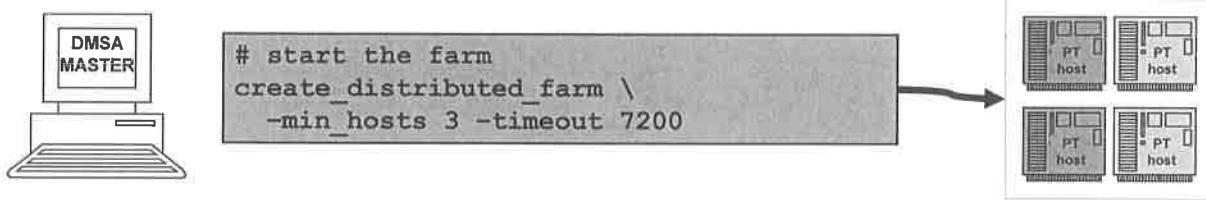


12- 44

Next, we give DMSA information on the remote processes we allow it to invoke. Compute farms are supported such as LSF and Sun Grid, and even proprietary in-house compute farms. Or you can give it a set of discrete machines by hostname. Multiple add\_distributed\_hosts commands can be given, and they are additive so that all commands are considered as desired host processes. The machines don't need to be the same, you can provide a mix of platforms if you want, such as linux and sparc, 32-bit and 64-bit, and so on. In the example below, we ask DMSA to invoke four processes on an LSF farm. I'd like to point out here that add\_distributed\_hosts does not actually invoke these remote processes, it's only giving DMSA information about the processes.

## Setting Up DMSA: Starting the Farm

- Next, using `create_distributed_farm` launch the remote host processes
- There are two options to control this:
  - `-timeout T` (defaults to 6 hours)
    - ◆ Wait T seconds, then proceed with available hosts
    - ◆ If nothing is available after the timeout, the analysis will error out due to lack of remote hosts
  - `-min_hosts N` (defaults to all hosts)
    - ◆ If N hosts become available, proceed immediately



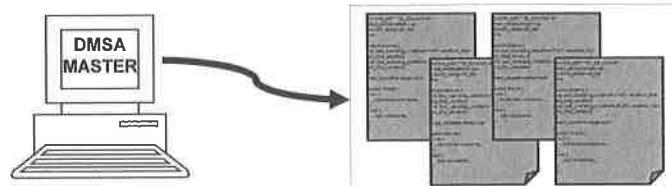
12-45

Once we've defined all the hosts processes, we use the `create_distributed_farm` command to actually launch all the remote host processes. There are two options available to control this. You can specify the “`-timeout`” option which controls how long the command will wait for the hosts to start up. By default, it will wait for six hours. Once the timeout expires, the analysis will proceed with whatever hosts are available. If no hosts are available, the analysis will eventually error out due to a lack of remote hosts. The section option is the “`-min_hosts`” option. By default, the command waits for all hosts to start up, but we can specify that once N hosts have started up, we should end the timeout and proceed. In both cases, if any additional hosts become available as the analysis is running, they will be dynamically added to the analysis.

## Setting Up DMSA: Defining Scenarios

- The analysis **scenarios** are defined in one of two ways:

- Providing PrimeTime scripts which load/constrain the analyses:



- Providing existing PrimeTime saved sessions:

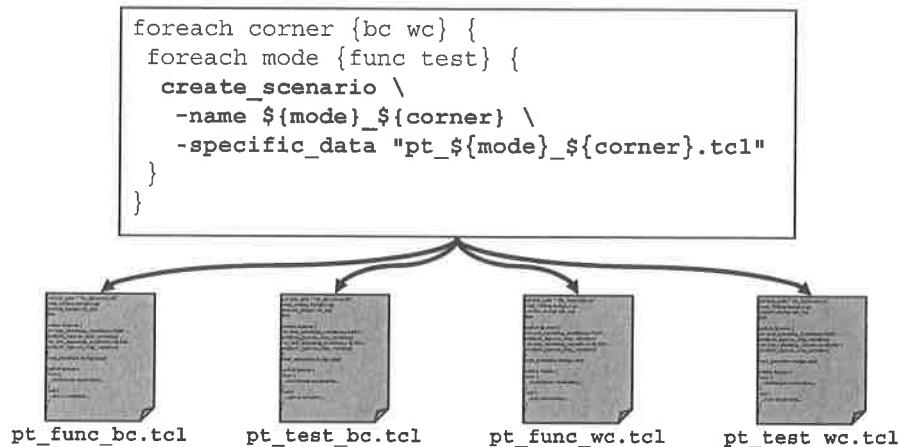


12- 46

At this point, we've launched all of our remote computing host processes. Now let's talk about how we define the analysis scenarios. The scenarios are defined in one of two ways – by providing PrimeTime analysis scripts which load and constrain the analyses, or by providing existing PrimeTime saved sessions to use as scenarios. If desired, we can even use both scenario definitions methods together.

## Setting Up DMSA: Defining Scenarios Using Scripts

- If you have a different script for each scenario, just point to the proper script for that scenario definition



12-47

The simplest case would be where you have a different script for each scenario. For example, you might have a separate script for your best-case functional run, your best-case test mode run, your worst-case functional run, and your worst-case test mode run. To define these four scenarios, we'd just use a double nested loop to iterate through all four possible combinations of mode and corner, create a scenario named according to the mode and corner variables, and point to the corresponding script file. PrimeTime will simply run each script to bring up that scenario and perform the analysis.

## Setting Up DMSA: Defining Scenarios Using Scripts

- The more common case is a single analysis script using variables to control which mode/corner is analyzed

- Below, mode and corner control the analysis

```
set link_path "* lib ${corner}.db"
read_verilog design.v.gz "pt_analysis.tcl"
current_design eth_top
link

switch $corner {
 bc {set_operating_conditions FAST -analysis_type on_chip_variation}
 wc {set_operating_conditions SLOW -analysis_type on_chip_variation}
}
read_parasitics design.sbps

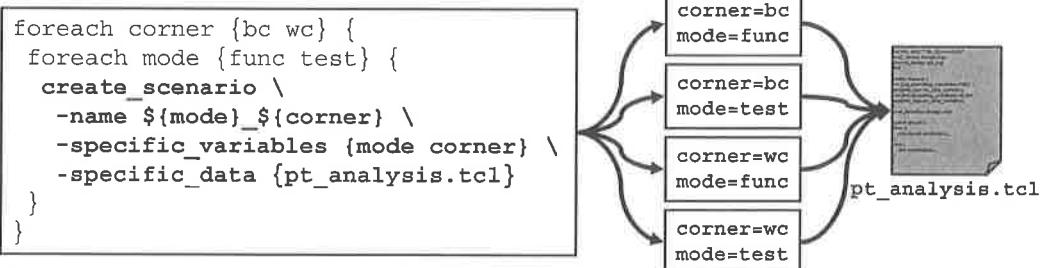
switch $mode {
 func {
 ...functional constraints...
 }
 test {
 ...test constraints...
 }
}
```

12- 48

Most of the time, you don't actually have a separate script for every different mode and corner combination. The more common case is where you have a single analysis script using variables to control which mode and corner is analyzed. In the analysis script below, you can see that the corner library controls which library should be used to link the design, and which operation condition should be used for the analysis. It also uses the mode variable to decide which SDC file to apply which constrain that operating mode.

# Setting Up DMSA: Defining Scenarios Using Scripts

- In this case, we simply vary these variables across their different combinations and create scenarios:



- The `-specific_variables` option “pushes” a variable’s current value at the master into the scenario
  - ◆ How does this work?

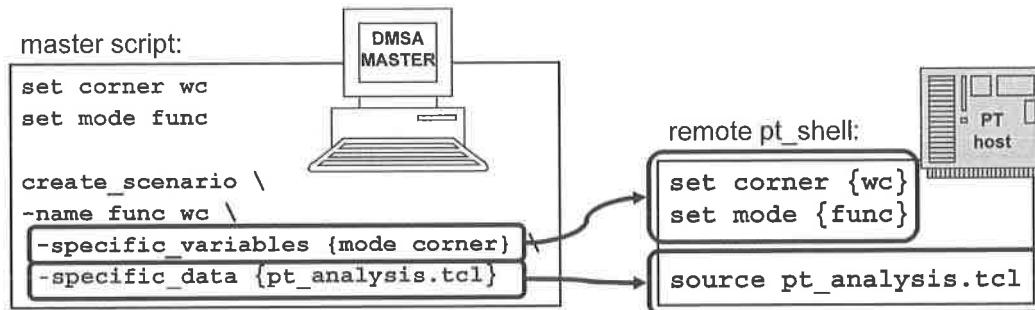
12-49

In this case, we simply vary these variables across their different combinations and create the scenarios as shown here. Again, we loop through all four combinations and create the scenarios named after the mode and corner. However this time, we specify this additional “`-specific_variables`” option to push the mode and corner variables down into the scenario, then point to the common analysis script. By defining scenarios in this way, each scenario gets the specific variable settings which go with that scenario definition, and those variable values will get used when the scenario is analyzed. The `-specific_variables` option actually pushes a variable’s current value at the master into the scenario. How does this work?

# Setting Up DMSA: Defining Scenarios Using Scripts

- Using `-specific_variables` in a scenario definition can be thought of as:

- invoking a PrimeTime session
- setting the specified variables
- sourcing the scenario script



12-50

Using `-specific_variables` in a scenario definition can be thought of as:

- \* first, invoking a PrimeTime session
- \* then, setting the specified values from that scenario definition in that PrimeTime session before you've sourced any scripts
- \* then finally, sourcing whatever scripts are a part of the scenario definitions

In this example, you can see that this scenario definition for functional worst-case would basically set the corner variable to wc, set the mode variable to func, then source the analysis script specified in the scenario definition.

## Setting Up DMSA: Defining Scenarios Using Scripts

- Environmental shell variables can be passed to the scenarios using Tcl's `env()` array
  - In Tcl, `env(foo)` represents environmental variable `foo`
  - Simply include the desired environmental variable entries in the variable list

master script:

```
foreach corner {bc wc} {
 foreach mode {func test} {
 create_scenario \
 -name ${mode}_${corner} \
 -specific_variables {mode corner env(DESIGNDIR)} \
 -specific_data {pt_analysis.tcl}
 }
}
```

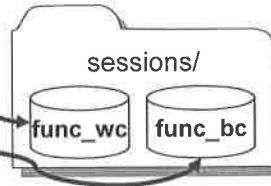
12-51

Sometimes, scripts use environmental variables from the UNIX or linux invoking shell to control where design files exist or where report files should be generated. In these cases, environmental shell variables can be passed from the master to the scenarios using Tcl's `env()` array. In Tcl, `env(foo)` represents the environmental variable "foo." Since the master was invoked by the user, the master will have the desired environmental variables. However, the remote processes invoked on the compute farm may not have them defined since they are fresh processes. To push the environmental variables down into the remote host processes, simply include the desired environmental variable array entries in the `-specific_variables` list.

## Setting Up DMSA: Defining Scenarios Using Sessions

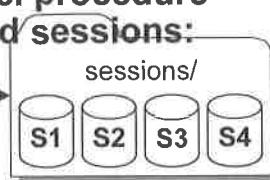
- You can also define scenarios using saved sessions:

```
create_scenario \
 -name func_wc -image sessions/func_wc
create_scenario \
 -name func_bc -image sessions/func_bc
```



- SolvNet article 018039 provides a Tcl procedure to easily restore a directory of saved sessions:

```
restore_dmsa_session sessions/
```



12- 52

We've seen how you can define scenarios using analysis scripts. You can also define scenarios using saved sessions. Just issue the `create_scenario` command with the scenario name, and point to a saved session directory. DMSA will restore each session and use it for that scenario's analysis. If you have a directory of saved sessions available, SolvNet article 018039 provides a Tcl procedure to easily restore a directory of saved sessions, automatically naming each scenario after its session directory name.

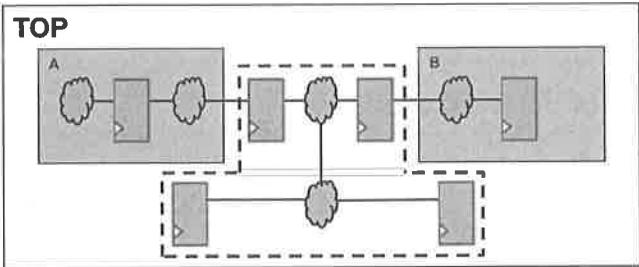
## DMSA Flow for ECO

- Different scenarios can be individually run into *saved sessions*
  - This avoids the issue of disparity in runtime of different scenarios
- DMSA ECO is run by restoring the *save\_sessions* data

```
foreach corner {bc wc} {
 foreach mode {func test} {
 create_scenario -image <scenario>
 }
}
report_global_timing
fix_eco_timing ...
report_global_timing
```

12-53

## HyperScale-driven ECO: Top Level ECO w/ Top Logic Only



### Approach

- ECO focused on top level logic only
  - `set_dont_touch` on blocks

### HyperScale advantage

- Allows targeted ECO without disturbing blocks – faster than full-chip ECO with smaller resources

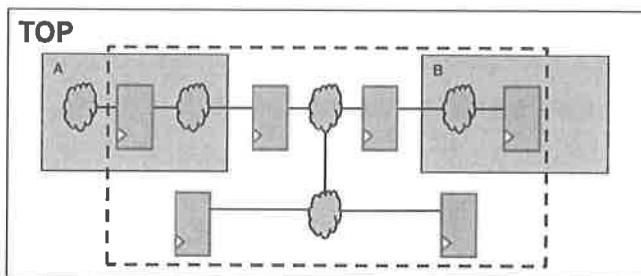
### HyperScale

```
restore_session HS_session
set_dont_touch {inst_A inst_B}
fix_eco_timing -type setup
write_changes -format icctcl \
-output pt_eco.tcl
```

Focus ECO on top  
level logic only

12- 54

## HyperScale-driven ECO: Top Level ECO w/ Block Boundary



HyperScale

```
restore_session HS_session
set_dont_touch {inst_A inst_B}
fix_eco_timing -type setup
write_changes -format icctcl \
-output pt_eco.tcl
```

### Approach

- Uses top level HyperScale context to drive block level ECO closure – with visibility into block level boundary logic
  - ECO can happen on block interface paths

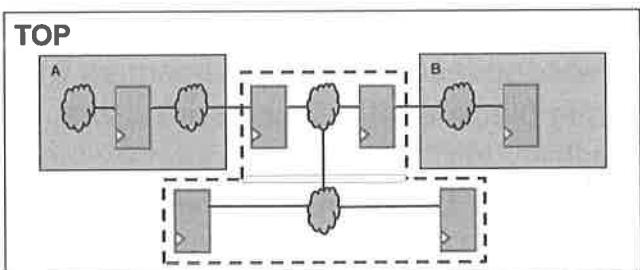
### HyperScale advantage

- Provides greater flexibility to apply ECO to complete block boundary path, while retaining HyperScale runtime/memory advantages over full chip ECO

Block interface  
paths are included  
by default

12-55

## HyperScale-driven ECO : Context Driven Block ECO



### HyperScale

```
restore_session HS_session

set_dont_touch {inst_A inst_B}

fix_eco_timing -type setup

write_changes -format icccl \
-output pt_eco.tcl
```

### Approach

- ECO focused on top level logic only
  - set\_dont\_touch on blocks

### HyperScale advantage

- Allows targeted ECO without disturbing blocks – faster than full-chip ECO with smaller resources

Focus ECO on top level logic only

12- 56

# Agenda

**DAY  
3**

**8 Signal Integrity: Crosstalk Delay Analysis**



**9 Signal Integrity: Crosstalk Noise Analysis**



**10 Correlation: POCV and AWP Analysis**

**11 Timing Closure: ECO/What If Analysis**

**12 Large Data: DMSA and Hyperscale Analysis**

**13 Conclusion**

13- 1

## Workshop Goal

# PRIMETIME

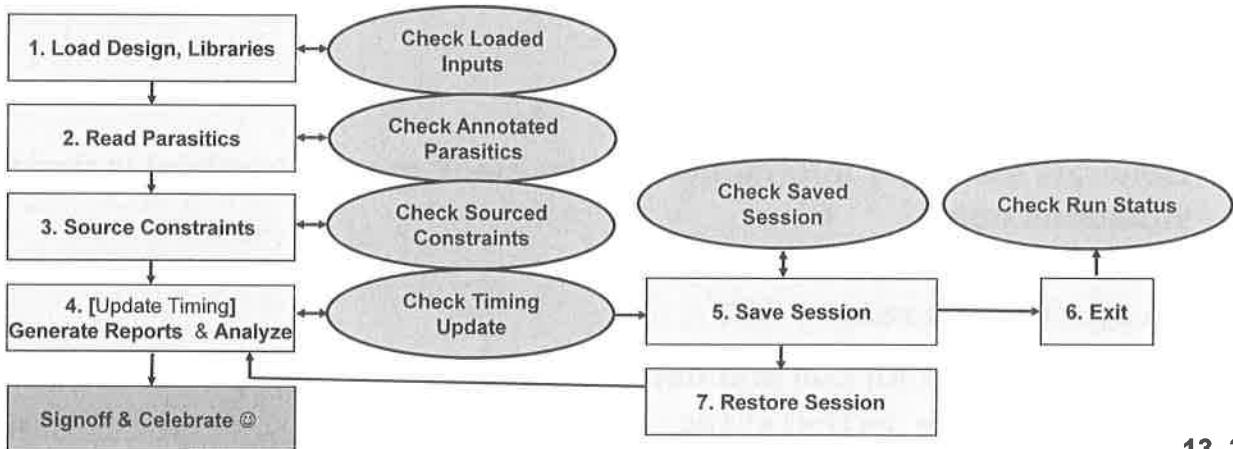
- Use *PrimeTime* to perform Static Timing Analysis (STA) and Signal Integrity (SI) analysis on a block or chip level design netlist by
  - Reading in design, library, parasitic data, and constraints
  - Debugging STA constraints before generating reports
  - Creating and Restoring saved sessions
  - Considering Signal Integrity (SI) impact due to Coupling Capacitances
  - Generating and Interpreting Reports for Summary, Timing and Noise
  - Accounting for On Chip [delay] Variations (OCV) using POCV technique
  - Using Path Based Analysis (PBA)
  - Enabling Advanced Waveform Propagation (AWP) for signoff accuracy

13-2

POCV: Parametric On Chip Variation

## Timing Analysis Flow in PrimeTime

- STA flow is divided into several steps
  - Steps 1-3 read data and
  - Analysis begins at Step-4
- The STA flow is repeated until signoff is achieved



13-3

STA: Static Timing Analysis

## Following Day-1 units, You Can Now

- State the purpose of using the following reports:
  - `report_global_timing`
  - `report_timing`
- Use the following commands to identify incomplete constraints and unexercised timing checks
  - `check_timing`
  - `report_analysis_coverage`
- Generate summary information for the clock groups involved in timing violations using
  - `report_qor -only_violated`
  - `report_constraint -all`
- Get explanation for cell and net delays using
  - `report_delay_calculation`

13-4

## Following Day-2 units, You Can Now

- Constrain multiple clocks that are
  - Logically Exclusive, Physically Exclusive and Asynchronous
- Analyze Timing paths involving latches, multi cycle paths and combinational feedback loops
- Exercise Minimum Pulse width, Recovery/Removal, clock gating and data to data checks
- Use Path Based Analysis (PBA) differentiating the 3 modes:
  - Path, Exhaustive and Derate Only
- Debug Timing reports containing unexpected or questionable
  - Source Latency
  - CRPR Credit or common point
  - Discrepancy with other reports as `report_delay_calculation`

13- 5

## Following Day-3 units, You Can Now

- **Perform Crosstalk Delay Analysis**

- Verify Constraint Correctness
- Use `all_path_edges` window alignment mode

- **Perform Crosstalk Noise Analysis**

- Generate Noise reports
- Differentiate the `report_at_source` vs. `report_at_endpoint` when using CCS Noise library

- **Interpret POCV reports using Side file and LVF input data**

- **Enable AWP as a technique for correlation with SPICE**

- **Perform (Physically Aware) ECO What-if analysis**

- **Discuss the STA roles of DMSA and Hyperscale analysis**

13- 6

# How to Download Lab Files (1/2)

After logging in through SolvNet, search for the article on  
“EST for Lab Files”, Doc ID: 002471

The screenshot shows a SolvNet article page for "EST for Lab Files" (Doc ID: 002471). The page includes a navigation bar with links to Documentation, Support, Downloads, Training, Methodology, and My Profile. Below the navigation is a breadcrumb trail: HOME > SOLVNET ARTICLE. The main content area displays the article title "Electronic Software Transfer (EST) for Lab Files". It includes details such as Doc Id: 002471, Product: Not Product Specific, Last Modified: 06/28/2018, and Average User Rating: ★★★★☆ (66). There are also links to Save Article, Tag Article, Print, and Email. A note states: "Welcome to the Customer Education EST Information page. This page has been created for workshop attendees looking to download the lab files of their recent workshop and/or for Customer System Administrators who need to download our lab files in preparation for a workshop." Another note says: "These are lab script files only, they do not contain course presentation material or executable pieces of Synopsys code." To the right of the main content are two sections: "SAVED ARTICLES" and "TAGS". A callout box with an arrow points down to the "SAVED ARTICLES" section, containing the text "Scroll down to your course".

| Workshop Description | Software Version | FTP Product Directory                       |
|----------------------|------------------|---------------------------------------------|
| Design Compiler      | 2016.03-SP5      | <a href="#">Lab_DC_2016.03-SP5 Download</a> |
| Design Compiler      | 2016.12-SP3      | <a href="#">Lab_DC_2016.12-SP3 Download</a> |
| Design Compiler      | 2017.09-SP4      | <a href="#">Lab_DC_2017.09-SP4 Download</a> |

13-7

## How to Download Lab Files (2/2)

|                                               |                |                                              |
|-----------------------------------------------|----------------|----------------------------------------------|
| IC Compiler II SoC DP                         | 2016.03-SP5    | Labs_JCC_II_SoC_DP_2016.03-SP5<br>Download   |
| IC Compiler II SoC DP                         | 2016.12-SP3    | Labs_ICC_II_SoC_DP_2016.12-SP3<br>Download   |
| IC Compiler II SoC DP                         | 2018.06        | Labs_ICC_II_SoC_DP_2018.06<br>Download       |
| IC Validator Beginning User                   | 2016.06        | Labs_ICV_BEGINNING_USERS_2016.06<br>Download |
| IC Validator Beginning User                   | 2017.06        | Labs_ICV_BEGINNING_USERS_2017.06<br>Download |
| IC Validator Runset                           | 2016.06        | Labs_ICV_RUNSET_2016.06<br>Download          |
| IC Validator Runset                           | 2017.06        | Labs_ICV_RUNSET_2017.06<br>Download          |
| Low Power Flow HLD (Frontend)                 | 2013.06        | Labs_LPF_HLD_2013.06<br>Download             |
| Low Power Flow PnR (Backend)                  | 2014.09        | Labs_LPF_PnR_2014.09<br>Download             |
| Power-Aware Verification with VCS-NLP and UPF | 2016.06-SP2    | Labs_NLP_2016.06-SP2<br>Download             |
| Power-Aware Verification with VCS-NLP and UPF | 2017.03        | Labs_NLP_2017.03<br>Download                 |
| PrimeTime                                     | <b>2018.06</b> | Labs_PT_2018.06<br>Download                  |

1.  
Locate  
Course  
Title

3.  
After Downloading,  
Follow the README file  
for the lab installation steps.

2.  
Click  
Download

13-8



# Customer Support

© 2018 Synopsys, Inc. All Rights Reserved

20181001

# Synopsys Support Resources

## ■ Build a solid foundation:

Hands-on training for Synopsys tools and methodologies

<https://synopsys.com/support/training.html>

- Workshop Schedule and Registration
- Download Labs  
(SolvNet ID required)

## ■ Drill down to areas of interest:

SolvNet online support

<https://solvnet.synopsys.com>

- Online technical information and access to support resources
- Documentation & Media

## ■ Ask an Expert:

Synopsys Support Center

<https://onlinecase.synopsys.com>

The screenshot shows the Synopsys Training & Education website. At the top, there's a banner with a photo of a person working at a computer. Below the banner, a section titled "Ready to get started?" encourages users to browse the training curriculum by product. Another section, "TRAINING COURSES", lists various courses: eLearning FreeView, Physical Implementation, RTL Synthesis, Signoff, Verification, OVM/TGAD, and Helpful Links. On the right side, there's a sidebar with links for Contact Us, Products, eLearning (with sub-links for FreeView, SystemVerilog, Courses, and Schedule), and Helpful Links (with sub-links for Course Catalog, Course Options, Training Center, Log In, SolvNet, and 2018 EDS Training Weeks).

<https://www.synopsys.com/support/training.html>

CS - 2

# SolvNet Online Support

- Immediate access to the latest technical information
- Product Update Training
- Methodology Training
- Thousands of expert-authored articles, Q&As, scripts and tool tips
- [Open a Support Center Case](#)
- Release information
- Online documentation
- License keys
- Electronic software downloads
- Synopsys announcements (latest tool, event and product information)

The screenshot shows the SolvNet Online Support homepage. At the top, there's a navigation bar with links for Documentation, Support, Downloads, Training, Methodology, and My Profile. Below the navigation is a search bar with dropdown options for "SELECT FILE", "Articles and Documentation", and "Advanced". A sidebar on the right contains links for "SOLN Session Recordings", "Former Atrenta Products", "Laker-Verdi Support Forum", and "CODE V, LightTools, RSoft, and LucidShape Users". There are also sections for "OPEN A SUPPORT CASE", "GLOBAL SUPPORT CENTER", "SYNOPSYS TRAINING", "FONDSYNTESIS.COM", and "LAKER VERDI FORUM". The main content area displays a list of articles under "New Updated Articles" and "Saved Articles". Some visible article titles include:

- How to Find Board Data Files and Import It to a Pin Path File [04-17-2017]
- How to Check for Missing Generated Clock Definitions on Divider Circuits [04-17-2017]
- Understanding Timing Exceptions and Masking Behavior [04-17-2017]
- [Short Video] Reverse Debug with Verdi [04-17-2017]
- Migrating From VCS/VCS-MX 2-2014.12 to VCS/VCS-MX 2-2015.09 [04-17-2017]
- Applying Filters on the Excitation Manager [04-16-2017]
- How to Prevent Unwanted Clock Pulses During Transition Delay Testing? [04-12-2017]
- How to Select a Portion of a Pattern Set and Evaluate the New Test Coverage [04-12-2017]
- How Do I Find the Expected Value of a ATPG Pattern? [04-12-2017]
- Effects of Display Options on Time-Based Power Analysis Results Due to VCS Delay Options [04-12-2017]
- TetraMAX Does Not Honor False Path in Routed EBD [04-12-2017]
- Synopsys' MultiVoltage Analysis (MVA) [04-12-2017]
- Script to Find the Distance Between Two Objects [04-11-2017]
- Highlighting Routed Clock Net Shapes That Violate the Nondshift Width [04-11-2017]
- Querying 45-Degree Routes [04-11-2017]
- Specifying a Package to the VCS Command so that the Source Code Is Active In Verdi [04-11-2017]
- How Can I Specify Per-Partition Scan and clock\_gating Scan Enable Signals? [04-10-2017]
- Calculating the Total Asset Number for the Different Sections in the Statistics Tab [04-09-2017]
- Using Simultaneous Multivoltage Analysis (SMVA) with ECOs [04-07-2017]
- Extremely Small Net Delay Reported for a High Capacitance Net [04-07-2017]
- The Checked Out License Feature for Debugging or Loading AMS Designs in Verdi [04-07-2017]
- Disabling and Enabling Messages in the Question Form [04-06-2017]
- The `netlist -by_writing` Command Generates a Large Number of Unnecessary Netlist Filling Commands [04-06-2017]

<https://solvnet.synopsys.com>

CS - 3

# SolvNet Registration

1. Go to SolvNet page:
  - <https://solvnet.synopsys.com/>
2. Click on:
  - "Sign Up for an Account"
3. Pick a username and password.
4. You will need your "Site ID"
  - For Information on how to find your Site ID, select the "Synopsys Site ID" link
5. Authorization typically takes just a few minutes.

The image displays three sequential screenshots of the Synopsys New User Registration process:

- Screenshot 1:** Shows the initial registration screen with fields for First Name, Last Name, and Password, and a "SIGN UP FOR AN ACCOUNT" button.
- Screenshot 2:** Shows the registration screen after entering information, with fields for First Name, Last Name, Password, and Re-enter Password, along with checkboxes for "I am 18 or older" and "Please Share".
- Screenshot 3:** Shows the final step where the user is prompted to enter their "Business Site ID" before clicking "Submit".

<https://solvnet.synopsys.com/ProcessRegistration> CS -4

# Support Center

## ■ Industry seasoned Application Engineers:

- 50% of the support staff has >5 years applied experience
- Many tool specialist AEs with >12 years industry experience
- Engineers located worldwide

## ■ Great wealth of applied knowledge:

- Service >2000 issues per month

## ■ Remote access, and interactive debug, available via WebEx

Contact us:  
Open a support case

The screenshot shows the Synopsys Global Support Centers page. At the top right, there's a navigation bar with links for PRODUCTS, SOLUTIONS, SERVICES, COMMUNITY, ABOUT US, and SUPPORT. Below the navigation is a search bar and a "Global Support Centers" link. The main content area features a large photo of a woman in a lab coat. To her right, a section titled "Expert Support is Just a Click Away" explains how Synopsys' network of global support resources helps keep design on schedule. It includes links for "North America", "Europe / Israel Central Europe", and "Asia Pacific". Each region section contains contact information: North America (800-245-8005, 7:00 am - 5:30 pm PT), Europe (Open a case online), and Asia Pacific (Open a case online). On the right side, there are several sidebar links: "CUSTOMER SUPPORT", "CODE V, Firing Tools, Executive Edition", "VIEW YOUR PRODUCTS", "VIEW YOUR CASES", "RELEASE NOTIFICATIONS", "UPDATE DOWNLOAD KITS", and "Related items".

<http://www.synopsys.com/Support/GlobalSupportCenters>

CS - 5

## Other Technical Sources

- **Application Consultants (ACs):**

- Tool and methodology pre-sales support
- Contact your Sales Account Manager for more information

- **Synopsys Professional Services (SPS) Consultants:**

- Available for in-depth, on-site, dedicated, custom consulting
- Contact your Sales Account Manager for more details

- **SNUG (Synopsys Users Group):**

<https://www.synopsys.com/community/snug.html>

CS - 6

# Summary: Getting Support

## ■ Customer Training

<https://www.synopsys.com/support/training.html>

- Register for a Class
- Download Labs

## ■ SolvNet

<https://solvnet.synopsys.com>

- Tool Documentation and Support Articles
- Product Update and Methodology Information / Training
- Open a Support Case (Support Center)

## ■ Other Technical Resources

- Synopsys Users Group (SNUG)
- Application Consultants
- Synopsys Professional Services

CS - 7

This page was intentionally left blank.