



PrimeTime 1

Workshop

Student Guide

10-I-034-SSG-005 2006.06

Synopsys Customer Education Services
700 East Middlefield Road
Mountain View, California 94043

Workshop Registration: **1-800-793-3448**

www.synopsys.com

Copyright Notice and Proprietary Information

Copyright © 2006 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.

Right to Copy Documentation

The license agreement with Synopsys permits licensee to make copies of the documentation for its internal use only. Each copy shall include all copyrights, trademarks, service marks, and proprietary rights notices, if any. Licensee must assign sequential numbers to all copies. These copies shall contain the following legend on the cover page:

"This document is duplicated with the permission of Synopsys, Inc., for the exclusive use of _____ and its employees. This is copy number _____. "

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.

Registered Trademarks (®)

Synopsys, AMPS, Arcadia, C Level Design, C2HDL, C2V, C2VHDL, Cadabra, Calaveras Algorithm, CATS, CRITIC, CSim, Design Compiler, DesignPower, DesignWare, EPIC, Formality, HSIM, HSPICE, Hypermodel, iN-Phase, in-Sync, Leda, MAST, Meta, Meta-Software, ModelTools, NanoSim, OpenVera, PathMill, Photolynx, Physical Compiler, PowerMill, PrimeTime, RailMill, RapidScript, Saber, SiVL, SNUG, SolvNet, Superlog, System Compiler, TetraMAX, TimeMill, TMA, VCS, Vera, and Virtual Stepper are registered trademarks of Synopsys, Inc.

Trademarks (™)

Active Parasitics, AFGen, Apollo, Apollo II, Apollo-DPII, Apollo-GA, ApolloGAI, Astro, Astro-Rail, Astro-Xtalk, Aurora, AvanTestchip, AvanWaves, BCView, Behavioral Compiler, BOA, BRT, Cedar, ChipPlanner, Circuit Analysis, Columbia, Columbia-CE, Comet 3D, Cosmos, CosmosEnterprise, CosmosLE, CosmosScope, CosmosSE, Cyclelink, Davinci, DC Expert, DC Professional, DC Ultra, DC Ultra Plus, Design Advisor, Design Analyzer, Design Vision, DesignerHDL, DesignTime, DFM-Workbench, Direct RTL, Direct Silicon Access, Discovery, DW8051, DWPCI, Dynamic-Macromodeling, Dynamic Model Switcher, ECL Compiler, ECO Compiler, EDAnavigator, Encore, Encore PQ, Evaccess, ExpressModel, Floorplan Manager, Formal Model Checker, FoundryModel, FPGA Compiler II, FPGA Express, Frame Compiler, Galaxy, Gatran, HANEX, HDL Advisor, HDL Compiler, Hercules, Hercules-Explorer, Hercules-II, Hierarchical Optimization Technology, High Performance Option, HotPlace, HSIMplus, HSPICE-Link, iN-Tandem, Integrator, Interactive Waveform Viewer, i-Virtual Stepper, Jupiter, Jupiter-DP, JupiterXT, JupiterXT-ASIC, JVXtreme, Liberty, Libra-Passport, Library Compiler, Libra-Visa, Magellan, Mars, Mars-Rail, Mars-Xtalk, Medici, Metacapture, Metacircuit, Metamanager, Metamixsim, Milkyway, ModelSource, Module Compiler, MS-3200, MS-3400, Nova Product Family, Nova-ExploreRTL, Nova-Trans, Nova-VeriLint, Nova-VHDLint, Optimum Silicon, Orion_ec, Parasitic View, Passport, Planet, Planet-PL, Planet-RTL, Polaris, Polaris-CBS, Polaris-MT, Power Compiler, PowerCODE, PowerGate, ProFPGA, ProGen, Prospector, Protocol Compiler, PSMGen, Raphael, Raphael-NES, RoadRunner, RTL Analyzer, Saturn, ScanBand, Schematic Compiler, Scirocco, Scirocco-i, Shadow Debugger, Silicon Blueprint, Silicon Early Access, SinglePass-SoC, Smart Extraction, SmartLicense, SmartModel Library, Softwire, Source-Level Design, Star, Star-DC, Star-MS, Star-MTB, Star-Power, Star-Rail, Star-RC, Star-RCXT, Star-Sim, Star-SimXT, Star-Time, Star-XP, SWIFT, Taurus, TimeSlice, TimeTracker, Timing Annotator, TopoPlace, TopoRoute, Trace-On-Demand, True-Hspice, TSUPREM-4, TymeWare, VCS Express, VCSI, Venus, Verification Portal, VFormal, VHDL Compiler, VHDL System Simulator, VirSim, and VMC are trademarks of Synopsys, Inc.

Service Marks (SM)

MAP-in, SVP Café, and TAP-in are service marks of Synopsys, Inc.

SystemC is a trademark of the Open SystemC Initiative and is used under license. ARM and AMBA are registered trademarks of ARM Limited. All other product or company names may be trademarks of their respective owners.

Document Order Number: 10-I-034-SSG-005

PrimeTime 1 Workshop Student Guide

Table of Contents

Unit i: Introduction & Overview

Introductions	i-2
Facilities.....	i-3
Course Materials	i-4
Workshop Goal	i-5
What is PrimeTime?	i-6
PrimeTime Inputs and Outputs	i-7
Workshop Assumptions	i-8
Workshop Objectives.....	i-9
Share Your Goals.....	i-10
Curriculum Flow	i-11
Agenda	i-12
Icons Used in this Workshop	i-15

Unit 1: Does Your Design Meet Timing?

Objectives	1-2
Primetime in the Implementation Flow	1-3
What is Static Timing Analysis?.....	1-4
PrimeTime Timing Analysis Flow.....	1-5
Does Your Design Meet Timing?	1-6
Are You Finished?	1-7
Timing Verification of Synchronous Designs.....	1-8
Define Setup and Hold.....	1-9
Static Timing Verification of FF2: Setup	1-10
PrimeTime Terminology.....	1-11
Four Sections in a Timing Report.....	1-12
The Header.....	1-13
Data Arrival Section	1-14
Data Required Section	1-15
Summary - Slack	1-16
Static Timing Verification of FF2: Hold.....	1-17
Which Edges are Used in a Timing Report?.....	1-18
PrimeTime Terminology.....	1-19
Example Hold Timing Report.....	1-20
Test For Understanding.....	1-21
Negedge Triggered Registers: Setup Time	1-22
What About Hold Time?.....	1-23
Which Edges are Used in a Timing Report?.....	1-24
Timing Report for Hold	1-25
Lab 1: Does Your Design Meet Timing?.....	1-26

Table of Contents

Lab 1 Wrap Up: Restoring a Session	1-27
Lab 1 Wrap Up: Turn On and Off Page Mode.....	1-28
Lab 1 Wrap Up: Time Units Based on Library	1-29
Lab 1 Wrap Up: Focus Timing Reports.....	1-30
Lab 1 Wrap-Up: Timing Models	1-31
Lab 1 Wrap-Up: Example Timing Report	1-32

Unit 2: Objects, Attributes, Collections

Objectives	2-2
Some Key Background Information	2-3
Objects in the PrimeTime Database.....	2-4
Accessing Objects	2-5
Returning Objects as a Collection.....	2-6
Accessing Attributes on an Object.....	2-7
Filtering with Expressions and Wildcards	2-8
Accessing Objects Connected to Objects.....	2-9
For more information on Tcl:	2-10
Lab 2: PrimeTime Objects	2-11

Unit 3: Constraints in a Timing Report

Objectives	3-2
2 Steps BEFORE Addressing a Violation.....	3-3
Clocks and Clock Constraints.....	3-4
Clock Latency Components	3-5
Pre Versus Post Clock Tree Synthesis (CTS)	3-6
Clock Uncertainty and Skew.....	3-7
Test For Understanding 1/3.....	3-8
Show All Cells on Clock Network.....	3-11
Timing Report with Schematic	3-12
Clocks and Clock Constraints.....	3-13
Start and End Points.....	3-14
Generate Timing Reports	3-15
Interface Paths: Input Ports	3-16
Test For Understanding.....	3-17
Gotcha – Constraining Interface Paths.....	3-18
Test For Understanding.....	3-19
Interface Paths: Output Ports	3-20
Test For Understanding 1/2.....	3-21
Test For Understanding 2/2.....	3-22
Constrain Output Paths for Hold.....	3-23

Table of Contents

Constrain for Minimum Delay: Output Paths	3-24
Calculation of Output Delay Time.....	3-25
Summary of Example Constraints	3-26
Lab 3: Constraints in a Timing Report.....	3-27
Lab 3 Wrap Up: Constraints on Ports	3-28
Lab 3 Wrap Up: Recall the Methodology	3-29

Unit 4: Timing Arcs in a Timing Report

Objectives	4-2
What Are Timing Arcs?.....	4-3
What is SDF (Standard Delay Format)?	4-4
Group Exercise.....	4-5
Include Input Pins in a Timing Report.....	4-6
Let's Talk About Transitions	4-7
Cell Delay – Rise and Fall	4-8
Cell Timing Checks: Rise and Fall	4-9
Rise and Fall in a Timing Report.....	4-10
Test for Understanding 1/2	4-11
Test for Understanding 2/2	4-12
Data Arrival Time: Edge Sensitivity 1/2.....	4-13
Data Arrival Time: Edge Sensitivity 2/2.....	4-14
Default Timing Reports – Worst Slack.....	4-15
Generate Worst Slack Timing Report.....	4-16
Generate Timing Reports for Rise or Fall.....	4-17
Common Types of Timing Arcs	4-18
Report Library Arcs: Inverter.....	4-19
Report Library Arcs: Flip-Flop	4-20
Last Topic - Hierarchy in a Timing Path.....	4-21
Identify Hierarchy in a Timing Report.....	4-22
Lab 4: Timing Arcs in a Timing Report	4-23
Lab 4 Wrap Up.....	4-24
Appendix.....	4-25
Tri-State Timing Arcs: Bus Contention	4-26
Tri-State Timing Arcs: Floating Bus	4-27

Unit 5: Control Which Paths are Reported

Objectives	5-2
Default Behavior of report_timing.....	5-3
Explore and Debug – Generate More Reports	5-4
Control Which and How Many Reports	5-5

Table of Contents

How Many Paths Would You Like Reported?	5-6
Setup: Exercise on Nworst and Max_Paths	5-7
Exercise on Nworst and Max_Paths	5-8
Report Many Paths to a Single Endpoint.....	5-9
Test For Understanding.....	5-10
Filter Further Using More Switches.....	5-11
Control Which and How Many Reports	5-12
From, To or Through	5-13
Clock Ports versus Clock Objects.....	5-14
Recommendation – Be Specific.....	5-15
Example #1	5-16
Explore a Rising Data Transition.....	5-17
Add More Switches.....	5-18
Example #2	5-19
Required - Use get_clocks	5-20
Example #3	5-21
Use * as a Wildcard	5-22
Useful Commands for Navigation	5-23
You Would Like To Know More!	5-24
Final Exercise.....	5-25
What Is Returned?.....	5-26
Lab 5: Control Which Paths are Reported	5-27
Lab 5 Wrap Up : Tcl Scripting.....	5-28
Lab 5 Wrap Up: Half Clock Cycle Paths	5-29

Unit 6: Summary Reports + Wrap Up

Unit 1 – 5 Review (1/2)	6-2
Unit 1-5 Review (2/2)	6-3
Objectives	6-4
Generating Reports – A Methodology	6-5
Recall - Does Your Design Meet Timing?	6-6
Apply Appropriate Switches	6-7
Switches - report_analysis_coverage	6-8
Test For Understanding.....	6-9
Report All Violations Sorted By Clock	6-10
Use report_constraint –all_violators	6-11
What If	6-12
Generate Summary report_timing.....	6-13
Test For Understanding.....	6-14
Find Resolutions for Fixing Violations.....	6-15
Example - Load Isolation or Upsizing	6-16
Large Delay Associated with Large Fanout	6-17

Table of Contents

Reporting Alternative Library Cells.....	6-18
Generate Timing Reports for Each Fanout	6-19
Identify Bottleneck Cells in the Design	6-20
Last Topic – Significant Digits	6-21
Set an Application Variable Instead!	6-22
Group Exercise.....	6-23
Lab 6: Generate Summary Reports	6-24
Lab 6 Wrap Up.....	6-25

Unit 7: Create a Setup File and Run Script

Unit Objectives	7-2
PrimeTime Inputs and Outputs	7-3
Create a .synopsys_pt.setup File	7-4
Example Setup File	7-5
Source Multiple Tcl Procedures.....	7-6
Setup Files in Other Locations.....	7-7
Components of a Master Run Script.....	7-8
Read Application Variables & Design Netlist	7-9
After Reading, Designs Must Be Linked	7-10
Identify All Libraries and Read All Designs	7-11
The Star in link_path.....	7-12
Read SDF	7-13
Test For Understanding 1/2.....	7-14
Test For Understanding 2/2.....	7-15
Components of a Master Run Script.....	7-16
Summary: Reading 6 PrimeTime Inputs.....	7-17
What is Your SolvNet ID?	7-18
Lab 7: Create a Setup File and Run File	7-19
Lab 7 Wrap Up: Questions On SolvNet / SNUG ?	7-20
Lab 7 Wrap Up: Browsing SolvNet.....	7-21
Lab 7 Wrap-Up: PTE-029 Warning	7-22
Lab 7 Wrap Up.....	7-23

Unit 8: Validate a Run Script

Unit Objectives	8-2
Components of a Master Run Script.....	8-3
Checking the Run Script	8-4
Identify Warnings in Sourced Files.....	8-5
Linking May be Incomplete!.....	8-6
Preventing Black Boxes	8-7

Table of Contents

Is the SDF Complete?	8-8
What Is Reported?.....	8-9
Validate Constraints are Complete	8-10
To Explore Port Details	8-11
Debug and Fix All Warnings	8-12
Test For Understanding.....	8-13
Examine the combined state of the inputs	8-14
Save the PrimeTime Session.....	8-15
Sharing Saved Sessions.....	8-16
Required Files and Directories.....	8-17
Improving Performance	8-18
Timing Updates.....	8-19
How Many Timing Updates Occurred?	8-20
Identify Where Timing Updates Occur.....	8-21
Create Metrics for Run Script	8-22
What Does ‘stopwatch’ Output Look Like?	8-23
Three Useful Commands and One Variable	8-24
Test For Understanding.....	8-25
Lab 8: Validate your design; Enhance a run script	8-26
Lab 8 Wrap Up.....	8-27
Lab 8 Wrap Up: Redirecting Reports	8-28
Lab 8 Wrap Up: Searching the Log Files.....	8-29
Lab 8 Wrap Up: Missing Information.....	8-30
Appendix	8-31
Guidelines to improve runtime and memory	8-32

Unit 9: Getting to Know Your Clocks

Review of Run Scripts	9-2
Share A Work Application.....	9-3
Unit Objectives	9-4
Clocks are STA	9-5
Three Types of Clocks	9-6
What Are Master Clocks?.....	9-7
Generated Clocks: Internally Derived Clocks.....	9-8
Generated Clocks: Source Latency	9-9
More Clocks - Source Synchronous Interface.....	9-10
Generated Clocks: Outgoing Clocks.....	9-11
Third Kind of Clock - Virtual Clocks	9-12
Use report_clock For All Clocks	9-13
How Many Clocks Are In Your Design?.....	9-14
A Useful Tcl Procedure.....	9-15
Test For Understanding 1/2.....	9-16

Table of Contents

Test For Understanding 2/2.....	9-17
Generate a Timing Report	9-18
Test for Understanding.....	9-19
Break	9-20
Asynchronous, Synchronous, & Exclusive Clocks.....	9-21
Synchronous versus Asynchronous Clocks	9-22
Interacting Clocks and Multiple STA Runs.....	9-23
Single Analysis with Multiple Clocks!	9-24
Different Example: Multiple Clocks.....	9-25
Test for Understanding.....	9-26
Asynchronous Clocks	9-27
Identify All Clock Crossings.....	9-28
How Do You Perform These Checks?.....	9-29
Generate Timing Reports Between Clocks.....	9-30
Interpret Timing Reports Between Clocks.....	9-31
Which Edges for Setup and Hold.....	9-32
Messages During Timing Updates.....	9-33
Timing Reports for Asynchronous Clocks.....	9-34
Reports for Unconstrained Paths	9-35
No Paths versus No Constrained Paths	9-36
General Guidelines.....	9-37
Lab 9: Getting to Know Your Clocks	9-38
Lab 9 Wrap Up: Propagated Clocks	9-39

Unit 10: Analysis Type and Back Annotation

Unit Objectives	10-2
Topics in this module.....	10-3
Back Annotation Files.....	10-4
Back Annotation for the PT Delay Calculator	10-5
How PrimeTime Calculates Cell Delay	10-6
Do Parasitic Files Supply All Needed Data?	10-7
Reading Parasitic RCs.....	10-8
Examples of Missing Parasitic RC Data.....	10-9
Long Runtime Reading Parasitics	10-10
SDF: Using Third-Party Delay Calculators	10-11
What Affects SDF Generation?	10-12
Effects of Slew	10-13
PrimeTime Slew Propagation and Cell Delays.....	10-14
Launch versus Capture Path – Use Which Slew?	10-15
Which Slew Goes with which Analysis Type?	10-16
Setup Optimism in the bc_wc Mode.....	10-17
Hold Optimism in the bc_wc Mode.....	10-18

Table of Contents

On Chip Variation: Single Library.....	10-19
On-Chip Variation: Multiple Libraries, Derating	10-20
To Support OCV	10-21
How Many Back Annotation Files?.....	10-22
How Many Constraint Files?	10-23
How Many OCV Runs?	10-24
# STA Runs is Equal to?.....	10-25
Break	10-26
Summary – Analysis Type	10-27
Clock Reconvergence Pessimism (CRP)	10-28
Removal of CRP from STA (CRPR)	10-29
Backannotation: Why use Derating Factors?.....	10-30
Back Annotation: Applying Derating Factors.....	10-31
Global Derating versus Specific Derating.....	10-32
Are these OCV corners?	10-33
How Will You Validate SDF?	10-34
Open the SDF File and Take a Look!	10-35
Pay Attention to Output from read_sdf	10-36
Validate Design Analysis Type.....	10-37
Completeness of SDF	10-38
Reporting Design Rule Violations	10-39
Summary: Why use Parasitics instead of SDF?	10-40
Useful Commands for Lab	10-41
Lab 10: SDF and Analysis Type	10-42
Lab 10 Wrap Up: Boundary Nets.....	10-43
Lab 10 Wrap Up: Missing SDF Annotations.....	10-44
Lab 10 Wrap Up: CRPR	10-45

Unit 11: Additional Checks and Constraints

Review of Units 9 - 10	11-2
Unit Objectives	11-3
Additional Timing Checks.....	11-4
Timing Checks Verified by STA	11-5
Asynchronous Clear/Reset Pins	11-6
Timing Report Recovery.....	11-7
Recall Path Groups – Full Picture.....	11-8
Test for Understanding.....	11-9
Clock Min Pulse Width.....	11-10
Summary Min Pulse Width Reports	11-11
Report for Min Pulse Width.....	11-12
Summary Report for All Timing Checks	11-13
Latches and Time Borrow	11-14

Table of Contents

What About Latches?.....	11-15
Latches with Zero Time Borrow	11-16
Each Path is Independent	11-17
Latches with Time Borrow: Setup	11-18
Max Allowable Time Borrow	11-19
Paths Are No Longer Independent.....	11-20
Account For Borrowing in Previous Stage	11-21
Break	11-22
Multicycle Paths.....	11-23
What Are Multicycle Paths?	11-24
By Default, What Does PT Report?.....	11-25
How do you Guide PrimeTime?	11-26
Multicycle Paths.....	11-27
Where Should the Hold Check Be?	11-28
PrimeTime Does Hold Checks Relative to Setup	11-29
Set the Proper Hold Constraint	11-30
Validate Hold Multipliers	11-31
Ignored Timing Exceptions.....	11-32
Path-Specific Timing Exceptions	11-33
Combinational Feedback Loops.....	11-34
STA and Combinational Loops	11-35
Will PrimeTime Generate a Warning?	11-36
Gotcha's with Combinational Loops	11-37
Identify the Valid Timing Path	11-38
For This Example, Turn On Dynamic Loop	11-39
Loop-breaking Details.....	11-40
Non-Unate Clock Path Logic	11-41
Propagating Both Senses of the Clock	11-42
PTE-070 non-unateness and set_clock_sense.....	11-43
What is the Next Step?.....	11-44
Clock Used as Data.....	11-45
Libraries for Deep Sub-Micron Design.....	11-46
CCS Supported Throughout Galaxy	11-47
CCS Availability: Library Vendors.....	11-48
Lab 11: Additional Checks and Constraints	11-49
Lab Wrap Up: Warnings in ORCA.....	11-50
Lab Wrap Up: Informational Messages	11-51

Table of Contents

Unit 12: Conclusion

Workshop Goal	12-2
After Units 1-6, You Can Now	12-3
After Units 7-9, You Can Now	12-4
After Units 10-11, You Can Now	12-5
Share A Work Application.....	12-6
Course Evaluation.....	12-7
Thank You!	12-8
Appendix.....	12-9
How to Download Lab Files (1/4)	12-10

Customer Support

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



PrimeTime 1

2006.06

Synopsys Customer Education Services
© 2006 Synopsys, Inc. All Rights Reserved

Synopsys 10-I-034-SSG-005

Introductions

- Name
- Company
- Job Responsibilities
- EDA Experience
- Main Goal(s) and Expectations for this Course

i-2



EDA = Electronic Design Automation

Facilities



Please turn off cell phones and pagers

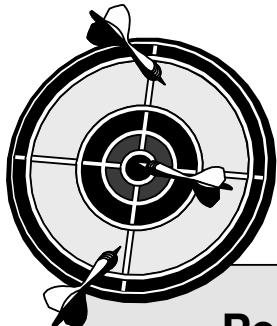
i-3

Course Materials

- **Student Workbook**
- **Lab Book**
- **Job Aid**
- **Course Evaluations (online)**

i-4

Workshop Goal



**Perform STA using PrimeTime.
Generate and interpret timing reports.
Create a clean, optimized run script.
Debug your timing violations.**

i-5

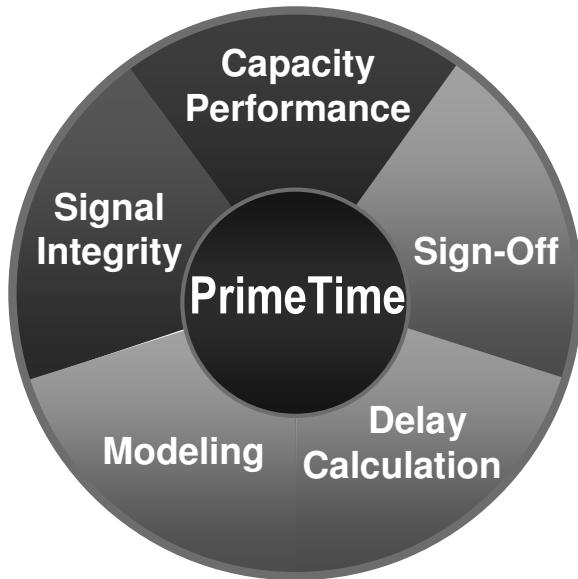


STA

Static Timing Analysis

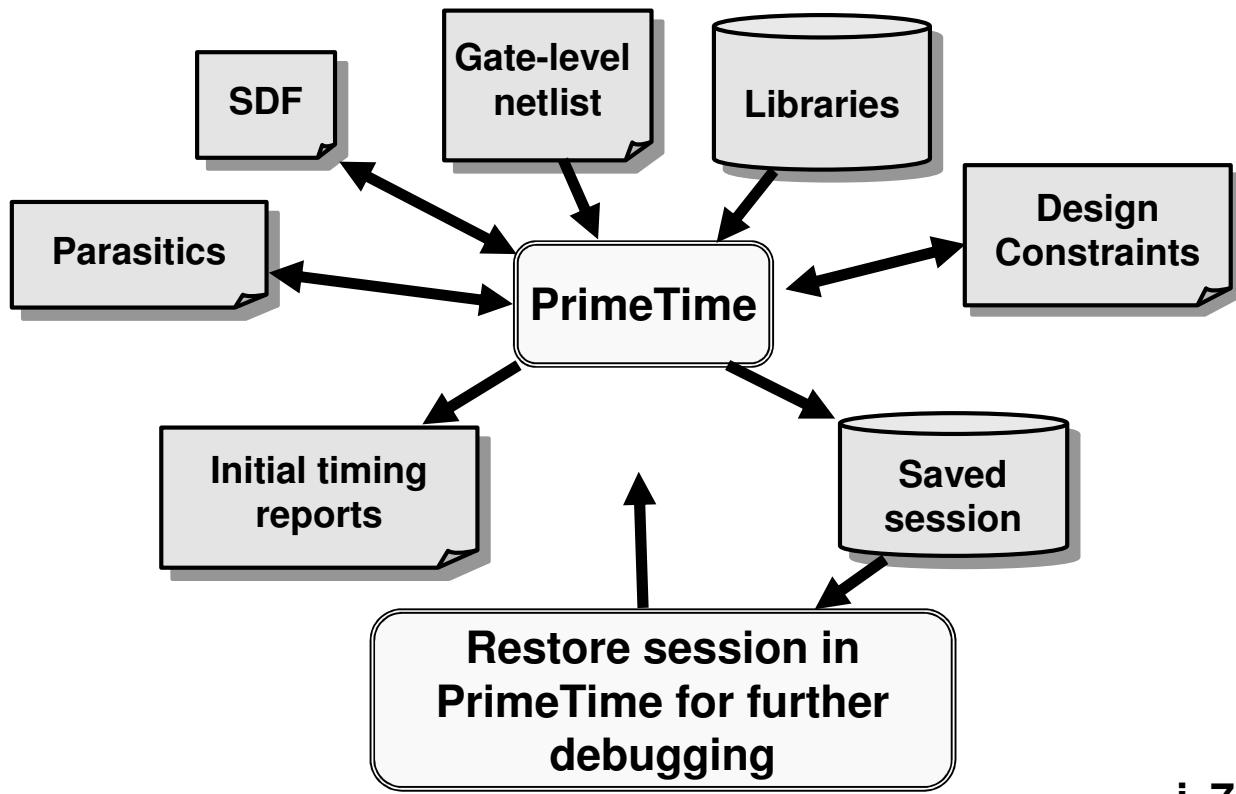
What is PrimeTime?

- Accurate Delay Calculator
- Performs Hierarchical STA
- Can include Signal Integrity Analysis
- 50M+ Gate Capacity and Performance
- #1 in ASIC Vendor SignOff
- Easy to adopt and use



i- 6

PrimeTime Inputs and Outputs



i-7

Workshop Assumptions

- **All input files are provided:**
 - Including the design constraints
 - ◆ Workshop covers constraint concepts for interpreting and debugging your design violations!
- **STA is performed using SDF or parasitic RCs:**
 - PrimeTime is accepted as a golden delay calculator
 - ◆ Workshop will focus on STA, not delay calculation
 - ◆ Skills learned in this workshop are essential for both SDF as well as parasitic based STA!
 - ◆ Parasitic flow details are provided, but labs and most examples use SDF

i-8

Workshop Objectives

Generate and interpret timing reports.

Units 1, 2, 3

**Generate custom timing information.
Explore pins, ports, and clocks.**

Units 4, 5, 6

**Create a clean, optimized run
script and supporting files.**

Units 7, 8

Explore design clocks.

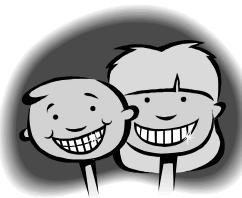
Unit 9

**Apply back-annotation files, on-chip
variation, and additional constraints.**

Units 10,11

i-9

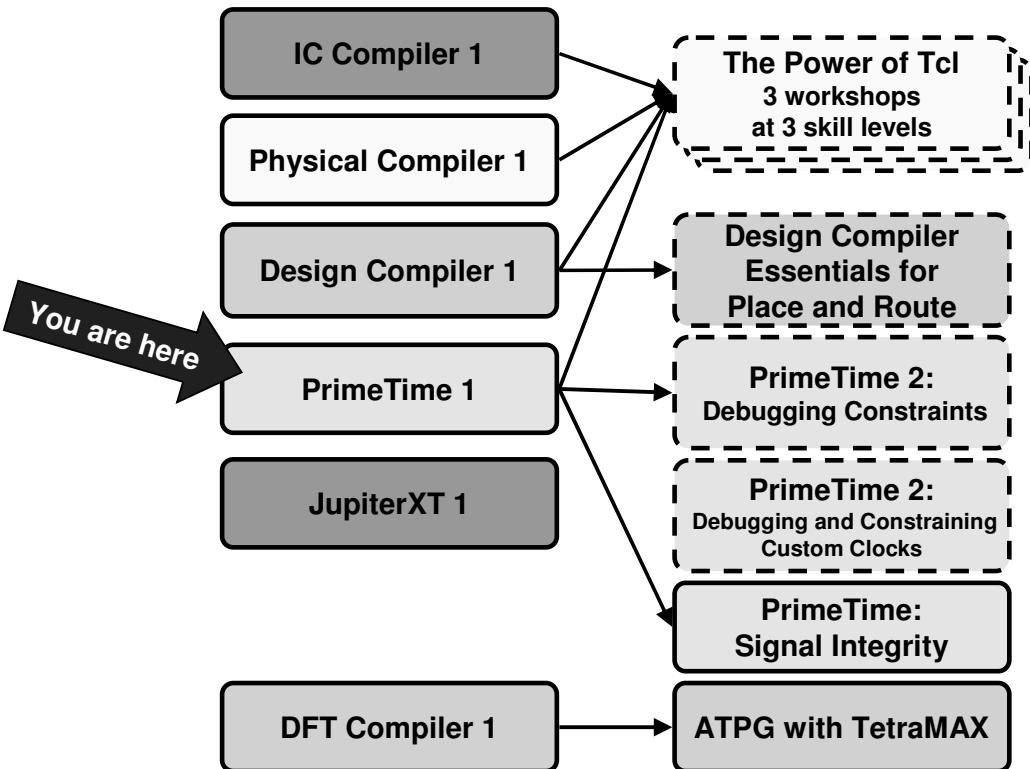
Share Your Goals



**Please share work problems you
are hoping to resolve with the skills
learned in this workshop.**

i-10

Curriculum Flow



i-11

Synopsys Customer Education Services offers workshops in two formats: The “classic” workshops, delivered at one of our centers, and the virtual classes, that are offered conveniently over the web. Both flavors are delivered *live* by expert Synopsys instructors. The Tcl workshops are also being offered as OnDemand playback training for FREE! Visit the following link to sign up:

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

Classic

Virtual classroom

Agenda

**DAY
1**

1 Does Your Design Meet Timing?



2 Objects, Attributes, Collections



3 Constraints in a Timing Report



4 Timing Arcs in a Timing Report



5 Control Which Paths are Reported



i-12

Agenda

**DAY
2**

6 Summary Reports



7 Create a Setup File and Run Script



8 Validate a Run Script



9 Getting to Know Your Clocks



i-13

Agenda

**DAY
3**

10 Analysis Type and Back Annotation



11 Additional Checks and Constraints



12 Conclusion

CS Customer Support

i-14

Icons Used in this Workshop



Lab Exercise



Caution



Recommendation



Definition of Acronyms



For Further Reference



Fill in workbook



**“Under the Hood”
Information**



Group Exercise

i-15

This page was intentionally left blank.

Agenda

**DAY
1**

1 Does Your Design Meet Timing?



2 Objects, Attributes, Collections



3 Constraints in a Timing Report



4 Timing Arcs in a Timing Report



5 Control Which Paths are Reported



Objectives



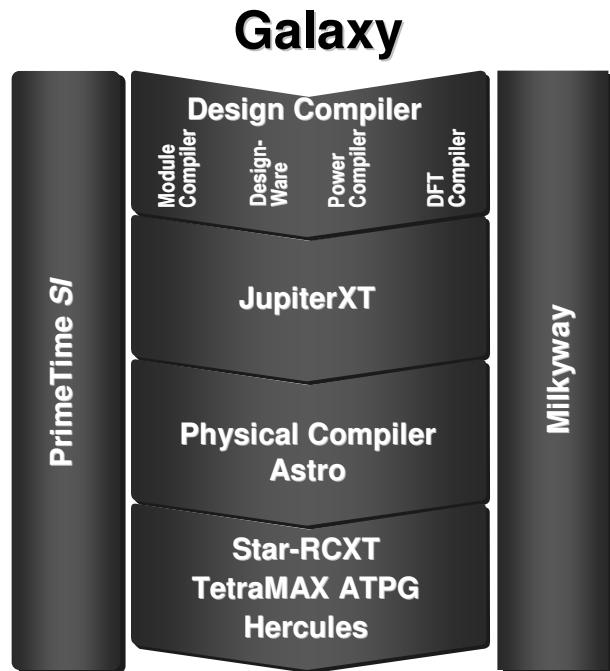
After completing this lecture, you should be able to:

- Interpret a summary report for design violations
- Interpret a specific timing report for setup and hold

1-2

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



1-3

What is Static Timing Analysis?

■ Static timing analysis:

- Verifies timing, not functionality
- Is exhaustive
 - ◆ Uses formal, mathematical techniques instead of vectors
 - ◆ Does not use dynamic logic simulation
- Is fast

1-4

PrimeTime Timing Analysis Flow

Units
7, 8

1. Setup Design Environment

- Search path, link path
- Read designs, libraries, then link
- Set OC, wire load, port load/drive/transition

Unit 10

2. Source Timing Assertions and Exceptions

- Clock period/waveform/uncertainty/latency
- Input/output delays
- Multi-cycle paths
- False paths
- Min/max delays, disabled arcs

Unit 11

Unit 9

3. Perform Analysis, Create Reports

- Check timing
- Timing reports and constraint reports
- Bottleneck and coverage analysis reports

Units 4, 5, 6

Units 1, 2, 3

4. Explore and Debug

- Clocks

1-5

Does Your Design Meet Timing?

```
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%)

Met + Violated + Untested = 100%



Circle the number of violations above.

1-6

64 violations are reported by report_analysis_coverage

Are You Finished?



**When PrimeTime was run it revealed
64 violations in the design.**

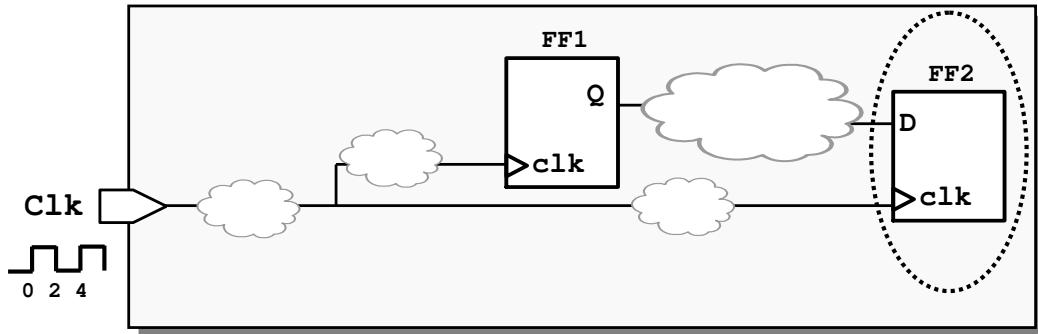
■ What else is there?

- Are the violations real?
- Can you explain warnings in the log files?
- What are your suggestions for resolution?
- You have a special situation – what are the issues?

1-7

Timing Verification of Synchronous Designs

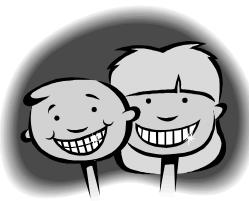
All “registers” must reliably capture data at the desired clock edges.



1-8

The term “registers” as used above includes all clocked devices, such as timing models or constrained output ports.

Define Setup and Hold



Setup and hold are “timing checks”.

You will learn additional timing checks on day 3.

Define:



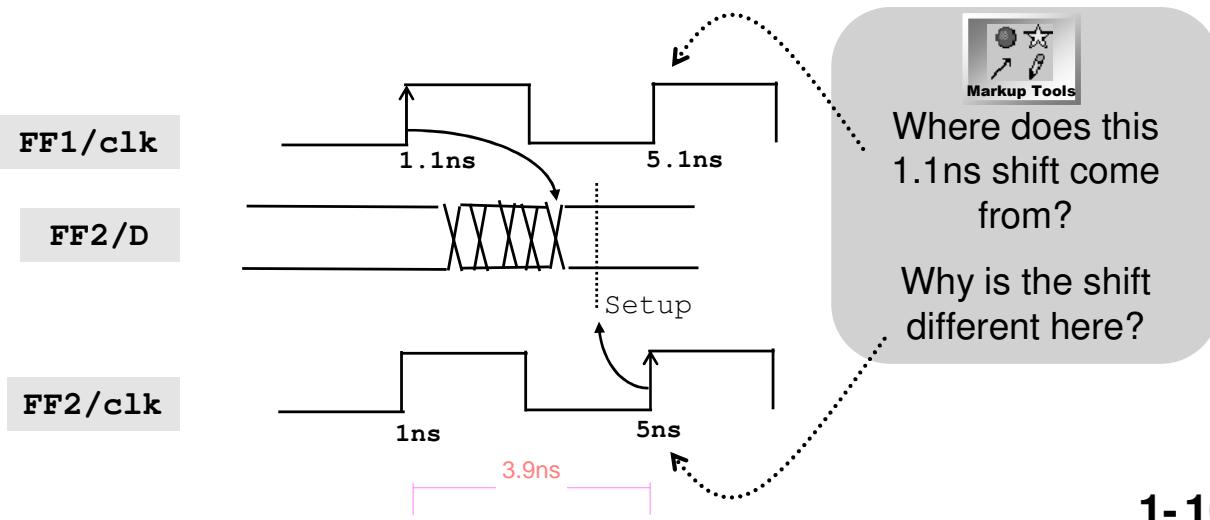
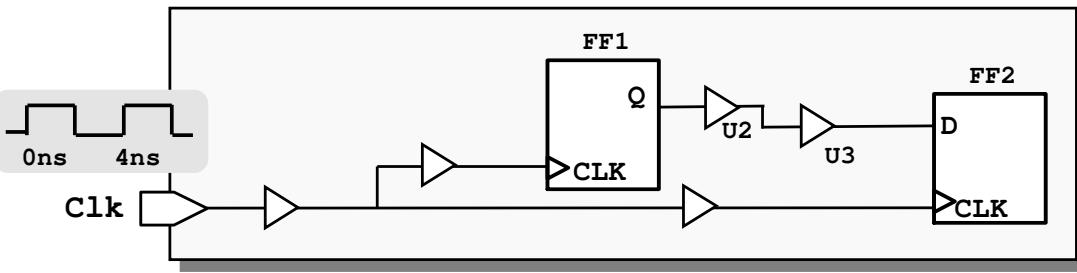
Setup

Hold

setup : the minimum time to be stable for latching before the clock edge
hold : the minimum time to be stable for latching after the clock edge

1-9

Static Timing Verification of FF2: Setup

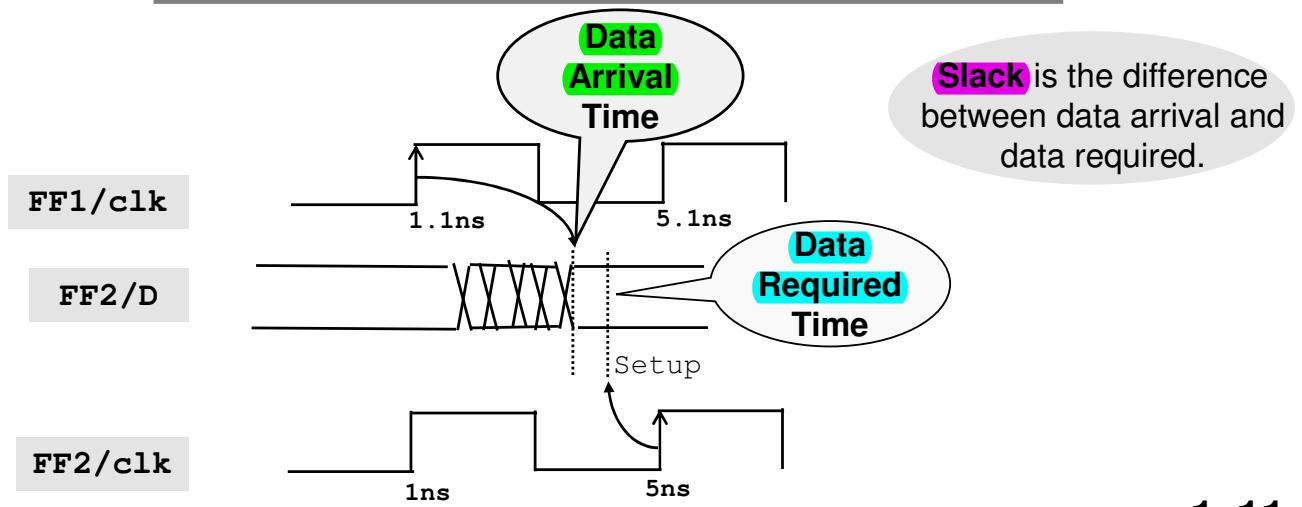
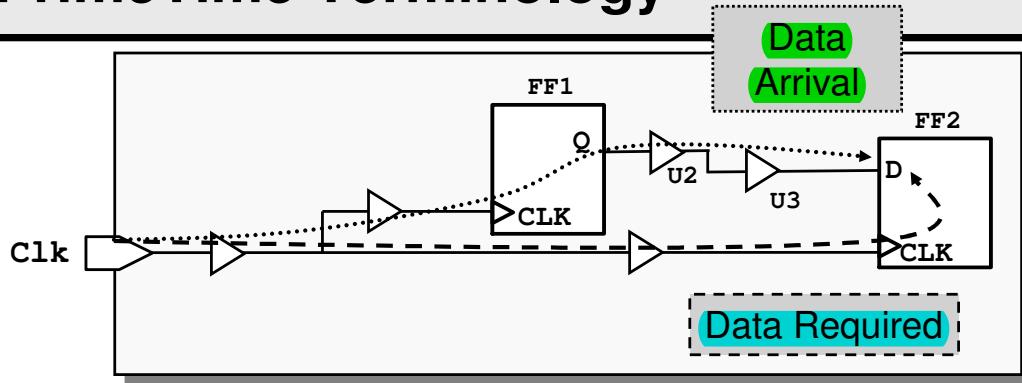


1-10

As an example, master clocks are created using the constraint shown below.

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

PrimeTime Terminology



1-11

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

Four Sections in a Timing Report

report_timing

report_timing -delay max ;# setup checking

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	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

Data required

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

Slack

data required time	4.79	
data arrival time	-1.87	
slack (MET)	2.92	

1-12

In PrimeTime, generate this report using the `report_timing -delay max` command. The switch `-delay max` indicates that the timing report is for `setup` (which is the default).

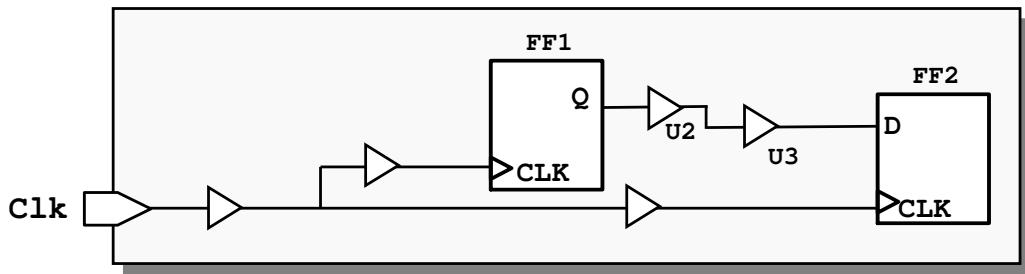
The Header

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

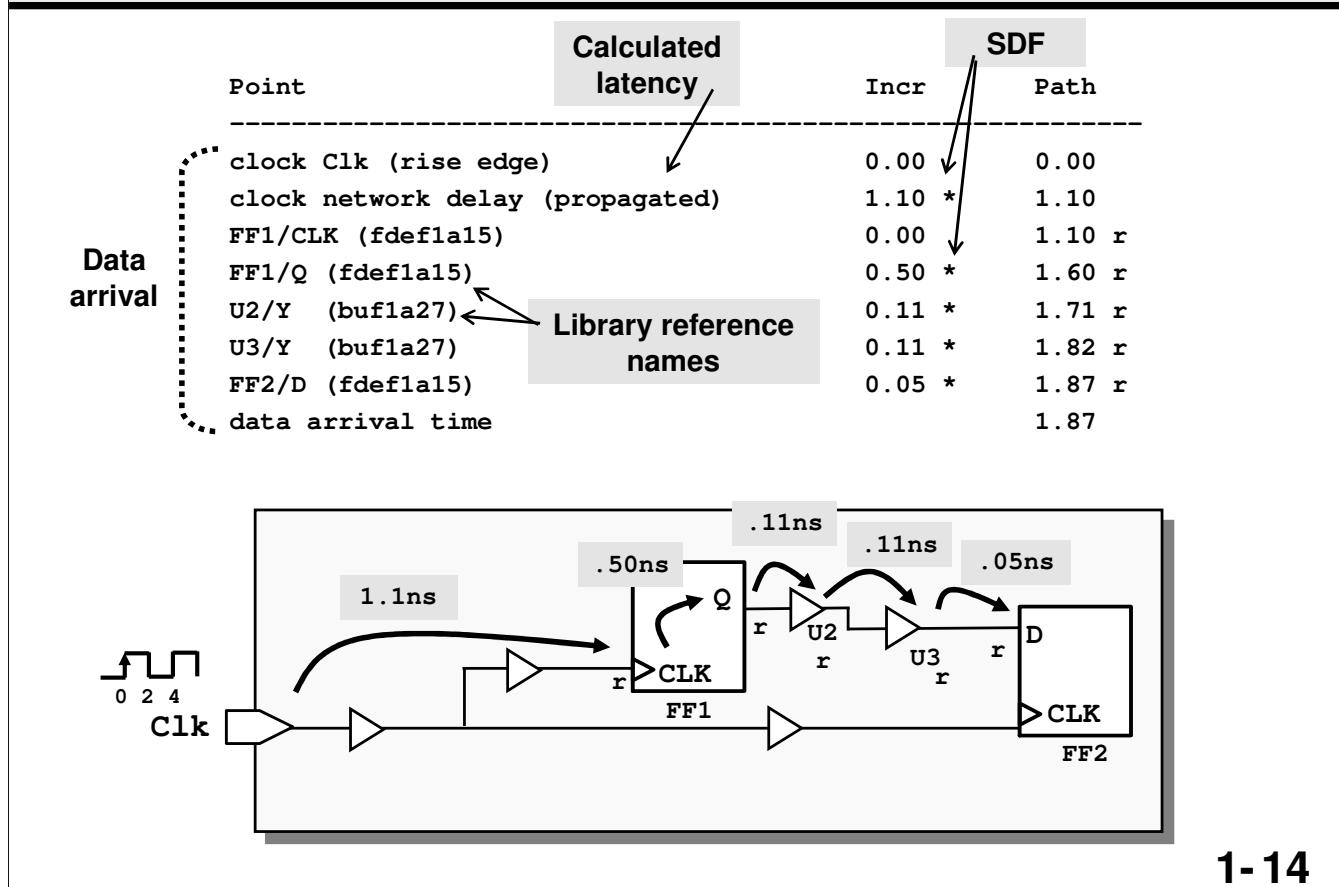
Capture clock

Report is for setup



1-13

Data Arrival Section



1-14

pt_shell> man report_timing

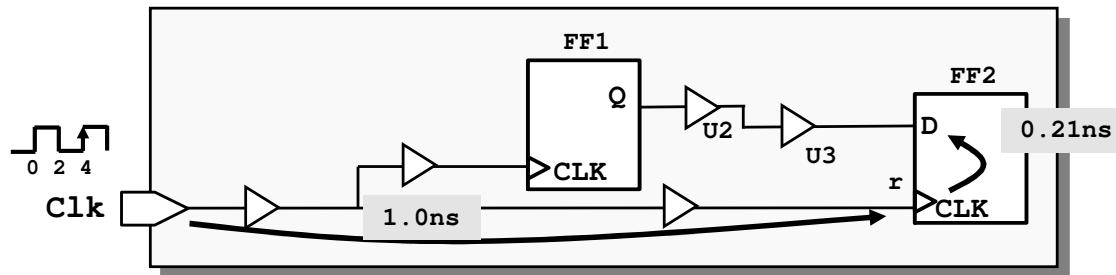
. . .

Symbol	Annotation
-----	-----
H	Hybrid annotation
*	SDF back-annotation
&	RC network back-annotation
\$	RC pi back-annotation
+	Lumped RC
<none>	Wire load model or none

Data Required Section

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
FF2/CLK (fdef1a15)		5.00 r
library setup time	-0.21 *	4.79
Data required		
data required time		4.79

SDF



1-15

Summary - Slack

report_timing

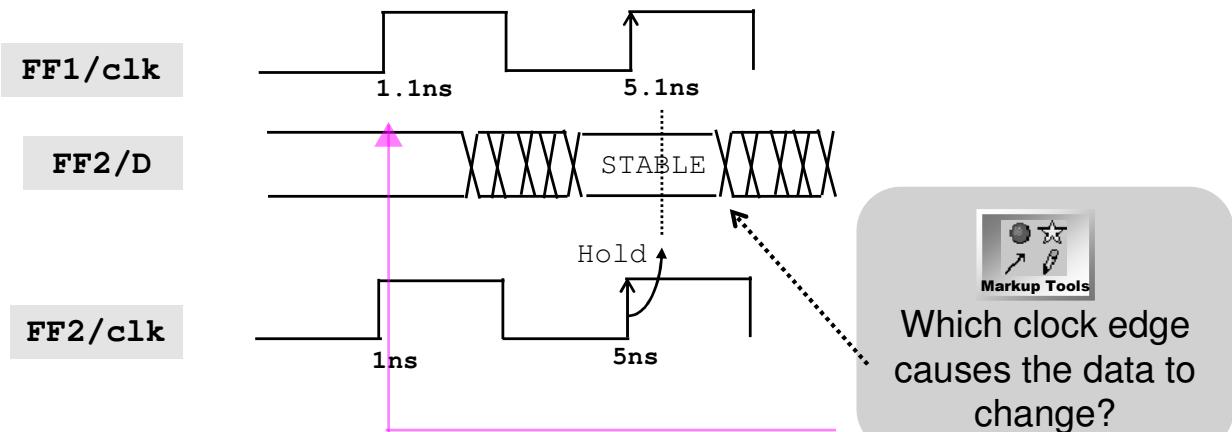
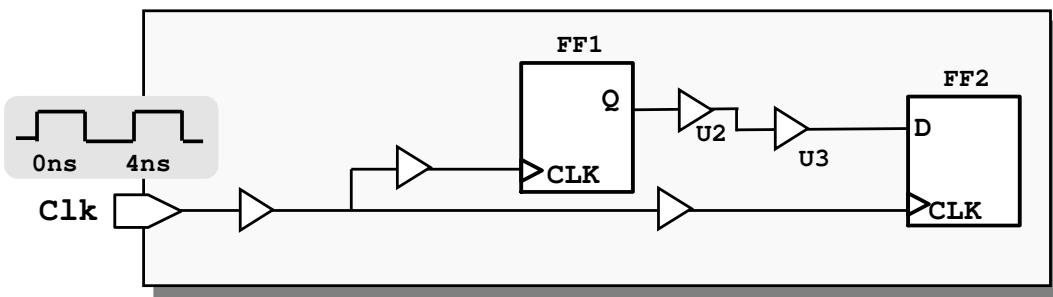
Startpoint: FF1 (rising edge-triggered flip-flop clocked by Clk)
Endpoint: FF2 (rising edge-triggered flip-flop clocked by Clk)
Path Group: Clk
Path Type: max

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)	0.50 *	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
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		-1.87
slack (MET)		2.92

Slack

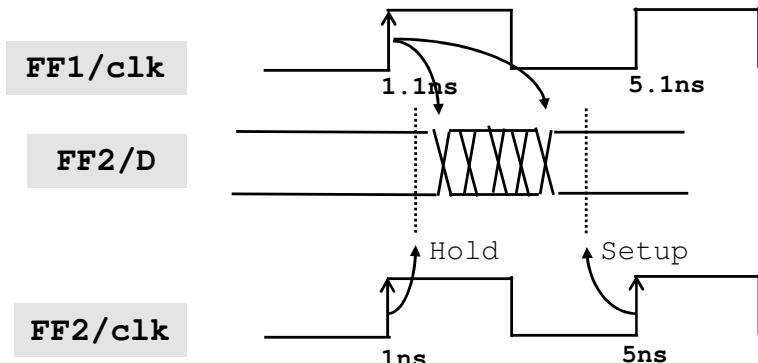
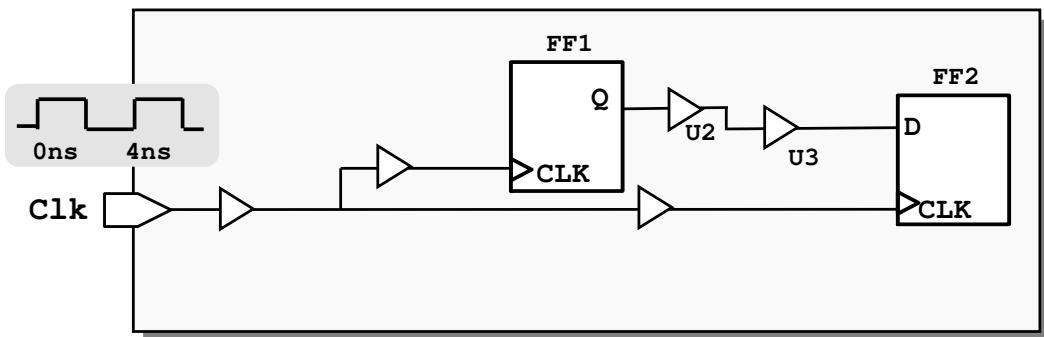
1-16

Static Timing Verification of FF2: Hold



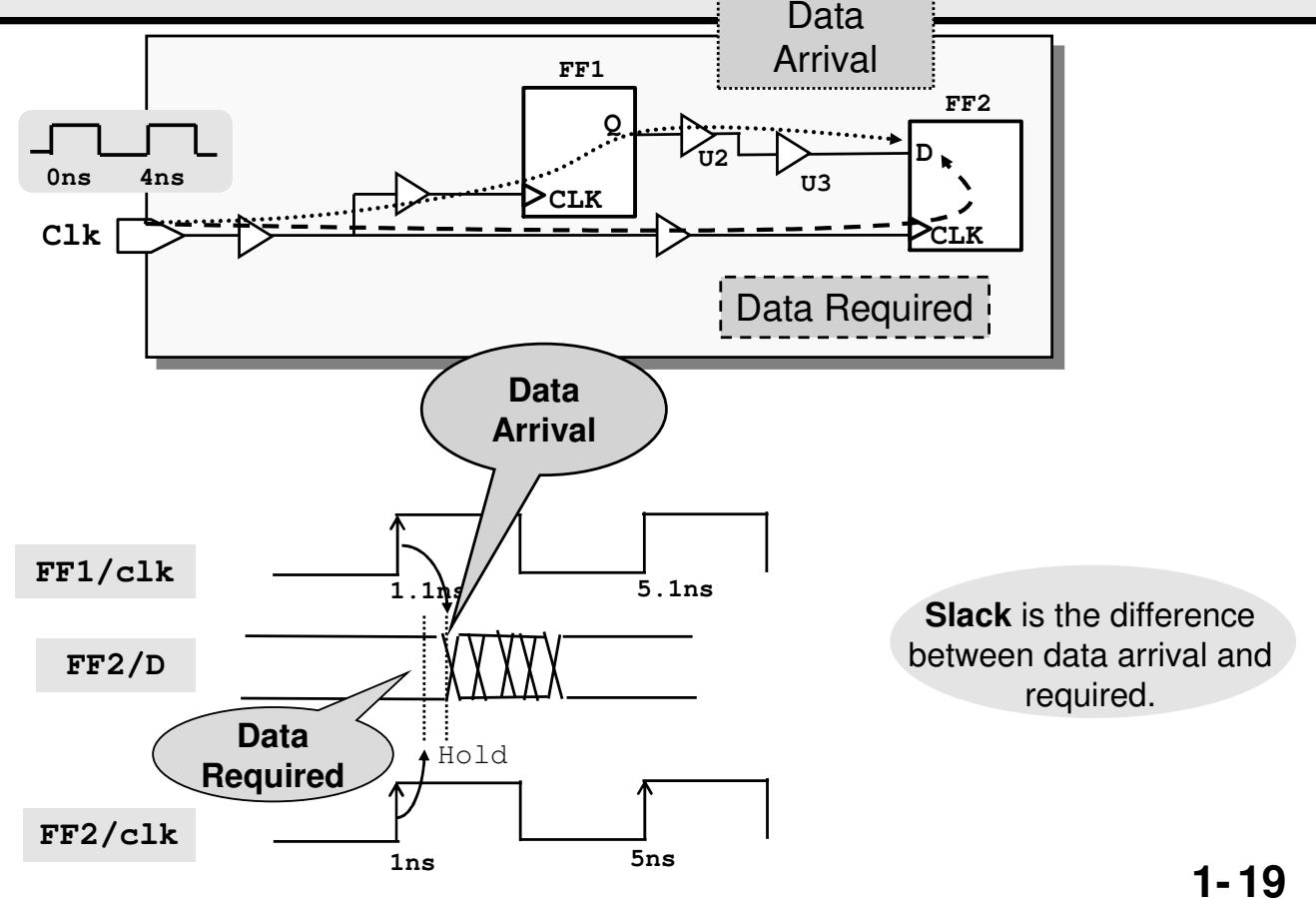
1-17

Which Edges are Used in a Timing Report?



1-18

PrimeTime Terminology



1-19

Example Hold Timing Report



Identify at least 3 clues that this is a hold report.

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
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	

1-20

In PrimeTime, generate this report using the `report_timing -delay min` command.

Test For Understanding



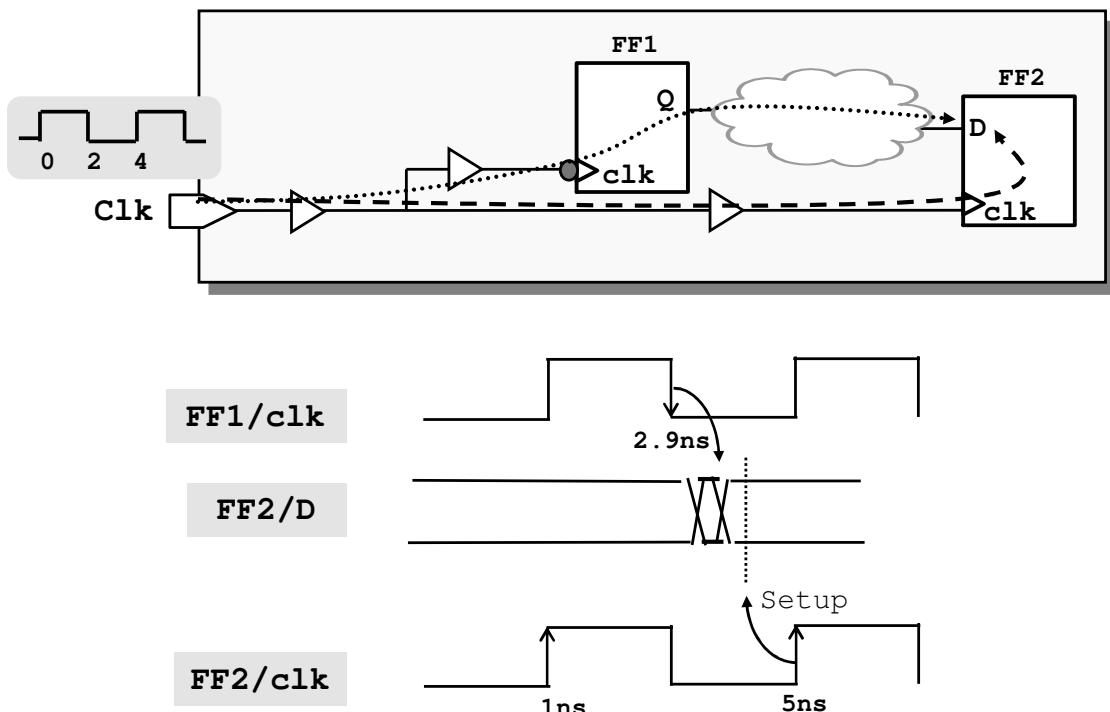
What is
different?

Startpoint: FF1 (falling 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 (fall edge)	0 -> 2	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 (buf1a27)	0.11 *	3.51 r
U3/Y (buf1a27)	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

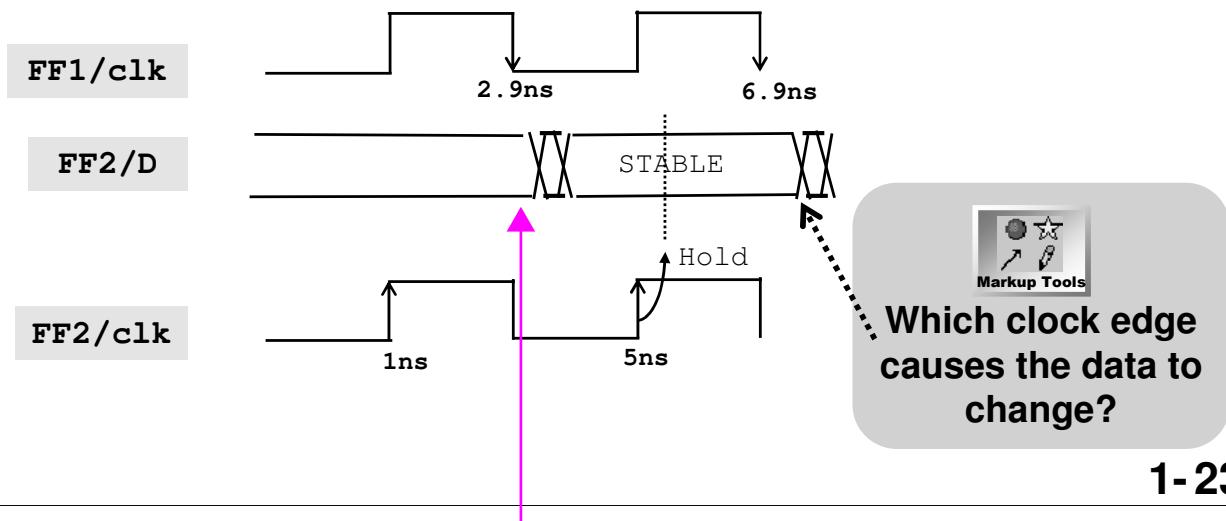
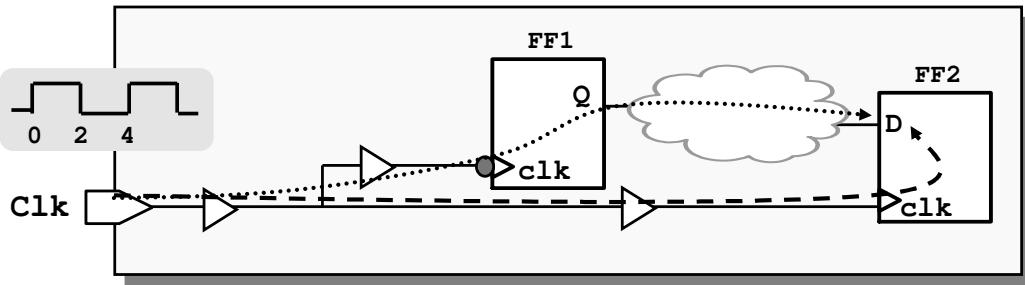
1-21

Negedge Triggered Registers: Setup Time



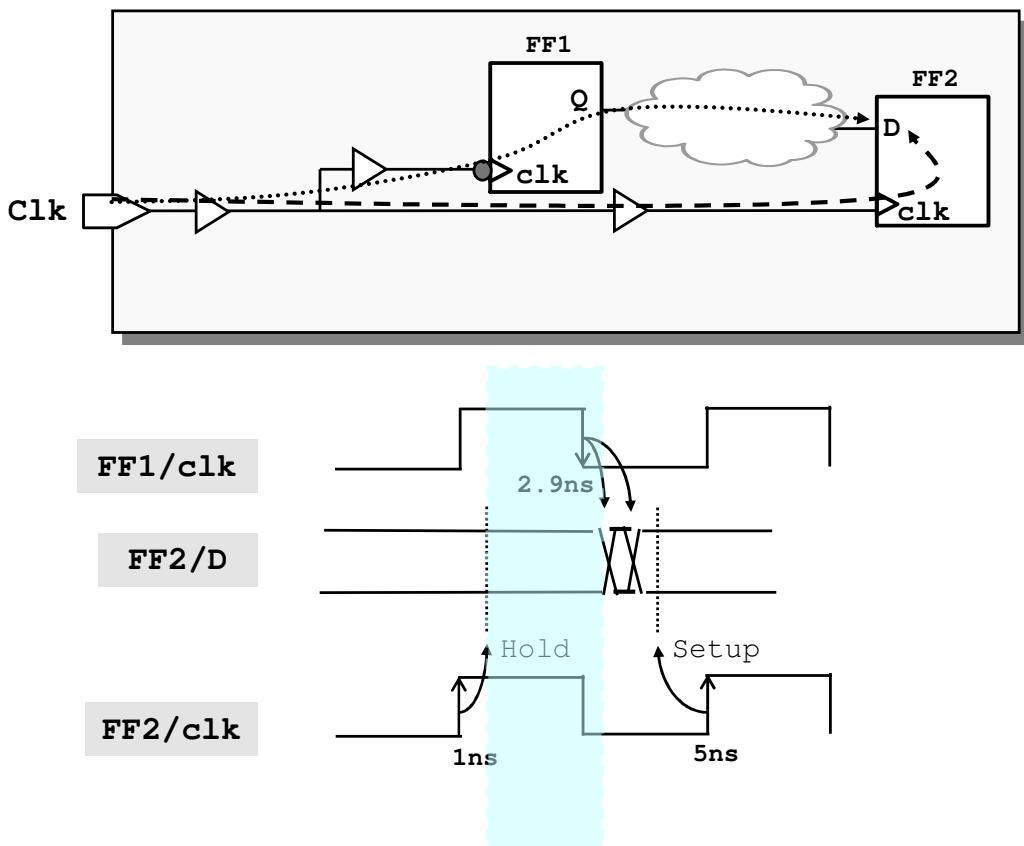
1-22

What About Hold Time?



1-23

Which Edges are Used in a Timing Report?



1-24

Timing Report for Hold



Circle the
clock
edges.

Startpoint: FF1 (falling 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
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 (fdef1a15)	0.40 *	3.30 f
U2/Y (buf1a27)	0.05 *	3.35 f
U3/Y (buf1a27)	0.05 *	3.40 f
FF2/D (fdef1a15)	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 (fdef1a15)		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

1-25

Lab 1: Does Your Design Meet Timing?



30 minutes



10 Minute BREAK
30 Minute LAB

The objective is to:

- Restore a PrimeTime session
- Execute and interpret a timing report for setup and hold
- Learn helpful commands when using PrimeTime

1-26

Lab 1 Wrap Up: Restoring a Session

```
pt_shell -x "restore_session orca_savesession"
```

■ Restore a previously saved PrimeTime session:

- Reads the design data, libraries and constraints for further timing analysis
- Does not restore the PrimeTime “state” (e.g. command history)

■ Requirements:

- Use the same PrimeTime version as that used to save the session
 - ◆ The platforms used for saving and restoring the session can be different (Solaris, HP, Linux, 32-bit, 64-bit)

1-27



For more information, refer to the **PrimeTime User Guide: Fundamentals** version 2006.06 in Unit 2, “Starting and Using PrimeTime: Analysis Flow in PrimeTime” under [Saving and Restoring PrimeTime Sessions](#).

Lab 1 Wrap Up: Turn On and Off Page Mode

PrimeTime application variable

ON

```
pt_shell> set sh_enable_page_mode true
```

OFF

```
pt_shell> set sh_enable_page_mode false
```

More tomorrow!

```
alias page_on {set sh_enable_page_mode true}
```

View a man page for page mode.



- man page_on
- man sh_enable_page_mode

1-28

```
pt_shell> man sh_enable_page_mode
```

NAME

sh_enable_page_mode

Displays long reports one page at a time
(similar to the UNIX more command).

TYPE

Boolean

DEFAULT

false

DESCRIPTION

This variable displays long reports one page at a time when set to true (similar to the UNIX more command). Consult the man pages for various commands that generate reports to see if this variable affects them. By default, it is set to false.

To determine the current value of this variable, use
printvar sh_enable_page_mode.

Lab 1 Wrap Up: Time Units Based on Library

Time units are determined by the main technology library.

```
pt_shell> list_lib  
# This is a Tcl comment  
# Use copy and paste to avoid mistyping the lib name  
pt_shell> report_lib cb13fs120_tsmc_max  
*****  
Report : library  
Library: cb13fs120_tsmc_max  
Version: V-2004.06  
Date   : Tue Aug  3 15:25:57 2004  
*****
```

Time Unit
Capacitance Unit

: 1 ns

: 1 pF

1-29

Lab 1 Wrap Up: Focus Timing Reports

The command `report_timing`
generates one report
with the worst slack
for each path group (e.g. name of the capture clock)
for setup time.

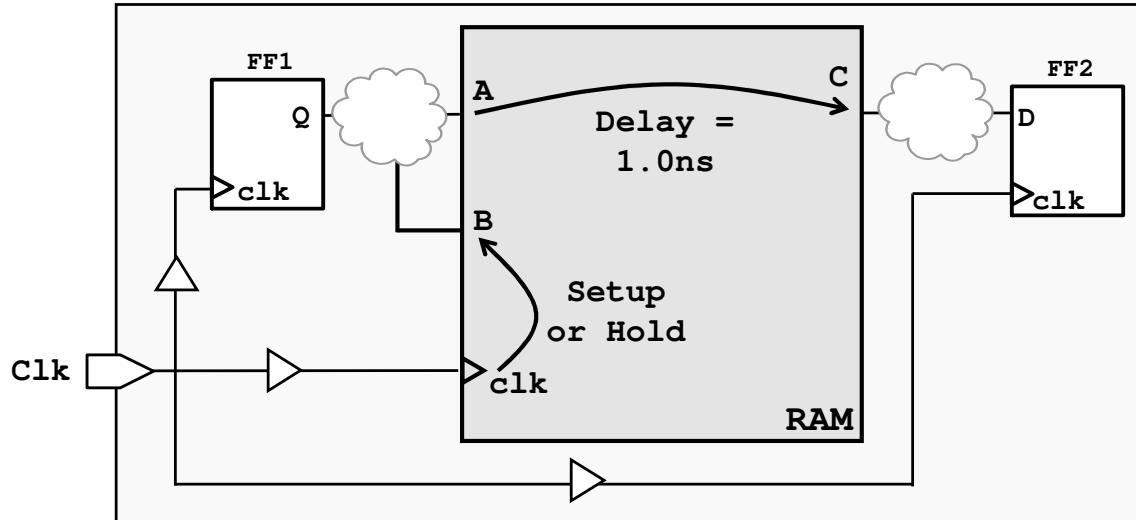
```
pt_shell> report_clock  
pt_shell> report_timing -group SYS_CLK
```

1-30

Lab 1 Wrap-Up: Timing Models

- Timing models are cells with many timing arcs:

- “Flip-flop” with setup and hold timing checks
- “Delay cell” included along the data arrival time



1-31

Lab 1 Wrap-Up: Example Timing Report

The timing model **ram32x32** looks like?



- A delay cell
- A flip-flop

Point	Incr	Path

clock SYS_CLK (rise edge)	0.000	0.000
clock network delay (propagated)	2.713 *	2.713
I_ORCA_TOP/I_PCI_WRITE_FIFO/count_int_reg[0]1/CP (sdcrql)	0.000	2.713 r
I_ORCA_TOP/I_PCI_WRITE_FIFO/count_int_reg[0]1/Q (sdcrql)	0.678 *	3.390 r
I_ORCA_TOP/I_PCI_WRITE_FIFO/PCI_WFIFO_RAM/A1[0] (ram32x32)	0.008 *	3.398 r
data arrival time		3.398

clock SYS_CLK (rise edge)	0.000	0.000
clock network delay (propagated)	2.711 *	2.711
I_ORCA_TOP/I_PCI_WRITE_FIFO/PCI_WFIFO_RAM/CE1 (ram32x32)		2.711 r
library hold time	0.282 *	2.992
data required time		2.992

data required time		2.992
data arrival time		-3.398

slack (MET)		0.406

1-32

Agenda

**DAY
1**

1 Does Your Design Meet Timing?



2 Objects, Attributes, Collections



3 Constraints in a Timing Report



4 Timing Arcs in a Timing Report



5 Control Which Paths are Reported



Objectives



**Timing Reports reflect your netlist and constraints.
To apply constraints, it is necessary to access netlist
objects.**

After completing this lecture, you should be able to:

- **List 5 objects defined in your netlist**
- **List 1 object that must be defined as a constraint**
- **Describe how to access objects**
- **Describe how to return objects for use by another command**

Examples of constraints are in the next module

2-2

Some Key Background Information

■ Key Concepts

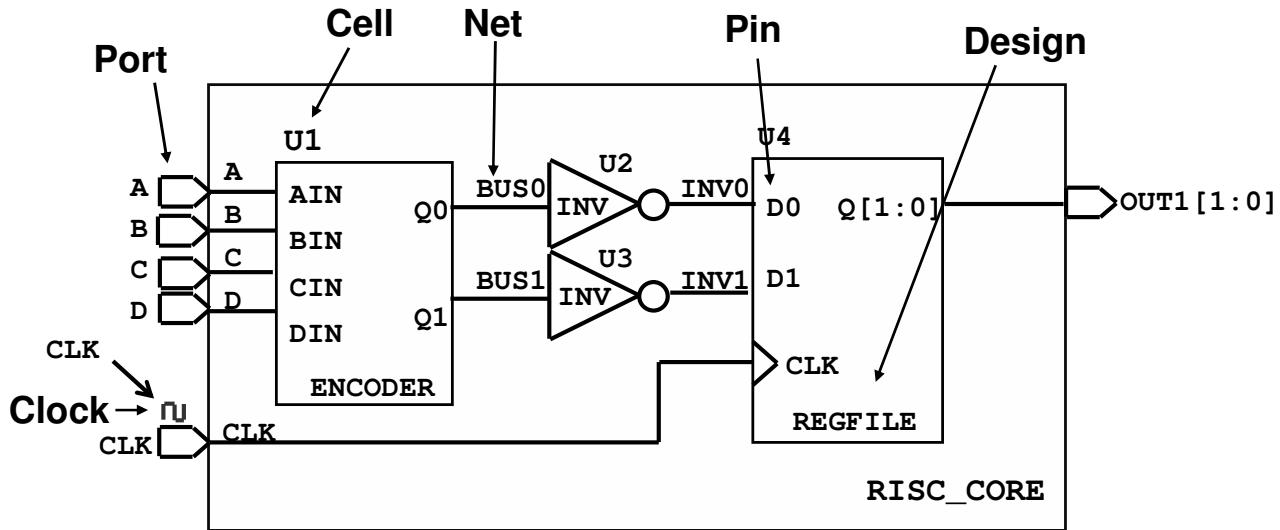
- Objects ← 6 objects are CCDPPN
- Attributes
- Collections ← a collection of an object

■ Additional: A little more power

- Accessing attributes on objects
- Expressions and wildcards
- Accessing objects connected to objects

2-3

Objects in the PrimeTime Database



The netlist defines 5 of the objects; you define the 6th:

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

Stored as an attribute on the clock object. To see all attributes on the clock object:

```
list_attributes -application -class clock
```

2-4

CCDPPN

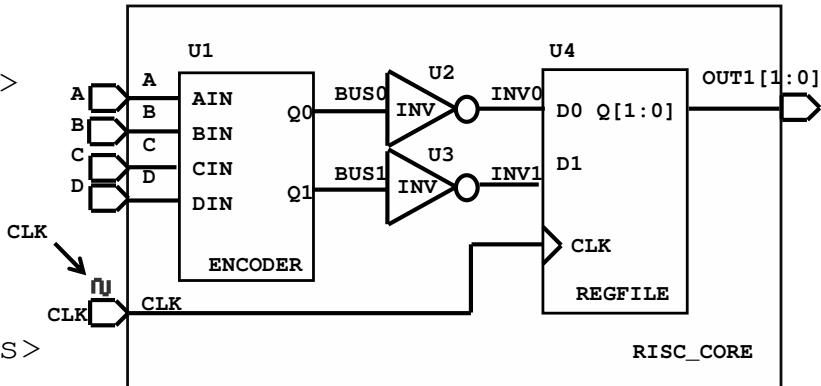
1. Cell
2. Clock
3. Design
4. Pin
5. Port
6. Net

Accessing Objects

The 'get' and the 'all' commands
return sets of objects called a 'collection'

a collection of an object

```
get_cells <patterns>
get_clocks <patterns>
get_nets <patterns>
get_pins <patterns>
get_ports <patterns>
get_designs <patterns>
all_instances <design_name>
```



How do you access all nets named 'CLK'?



If you refer to an object without using a 'get' command; for example, `create_clock -period 5 CLK`; where is the clock object created?

2-5

Returning Objects as a Collection

- ***Return a collection of ports to create_clock***

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

Square brackets
nest commands

- ***Return a collection of clocks and store in a variable***

```
set my_clocks [get_clocks SYS*]
```

- ***Display collections stored in a variable***

```
query_objects $my_clocks  
{ "SYS_CLK", "SYS_2x_CLK" }
```

\$ is required to
dereference this
variable

```
printvar my_clocks  
my_clocks = "_sel1342"
```

collections are groups
of objects with
attached attributes,
and cannot be
displayed as if they
were simple lists

2-6

The display produced by query or printvar is not the value returned by the command, is not a list, and will show a limited number of collection member names. The maximum number is the value of the **collection_result_display_limit** variable and may be set by the user (default 100).

Accessing Attributes on an Object

- What attributes are on the clock object?

```
pt_shell> list_attributes -application -class clock
...
period           clock   float    A
propagated_clock clock   boolean   A
sources          clock   collection A
...
...
```

- What is the value of a particular attribute?

```
pt_shell> get_attribute [get_clocks SYS_CLK] period
8.000000
```

Or

```
report_attribute -application [get_clocks SYS_CLK]
```

2-7

Filtering with Expressions and Wildcards

- Return all clocks whose period is less than or equal to 8

```
get_clocks -filter "period <= 8"  
{ "SYS_CLK", "SDRAM_CLK", "SD_DDR_CLK",  
  "SD_DDR_CLKn", "SYS_2x_CLK" }
```

- Return all cells whose reference name begins with 'mx'

```
get_cells -filter "ref_name =~ mx*"  
{ "U149", "U150", "U145", "U146", "U147",  
  "U148" }
```

Special operators required for wildcards

- Relational operators: ==, !=, >, <, >=, <=, =~, !~

2-8

Accessing Objects Connected to Objects

```
pt_shell> get_cells *CLOCK*
```

```
{ "I_CLOCK_GEN" }
```



- Return the nets connected to the I_CLOCK_GEN cell

```
pt_shell> get_nets -of_objects I_CLOCK_GEN
```

```
{ "net_sys_clk", "net_sdram_clk", "net_pcclk", ... }
```



- Return the cells connected to the net_sys_clk net

```
pt_shell> get_cells -of_objects net_sys_clk
```

```
{ "sys_clk_iopad", "I_CLOCK_GEN" }
```

2-9

For more information on Tcl:

- Courses
 - **Tcl 1: Becoming a Proficient User**
 - **Tcl 2: Creating Procedures**
 - **Tcl 3: Collections & Attributes**
- Books
 - **Tcl and the Tk Toolkit by John K. Ousterhout**
 - **Practical Programming in Tcl & Tk by Brent B. Welch**
- Web
 - **www.tcl.tk**
 - **wiki.tcl.tk**

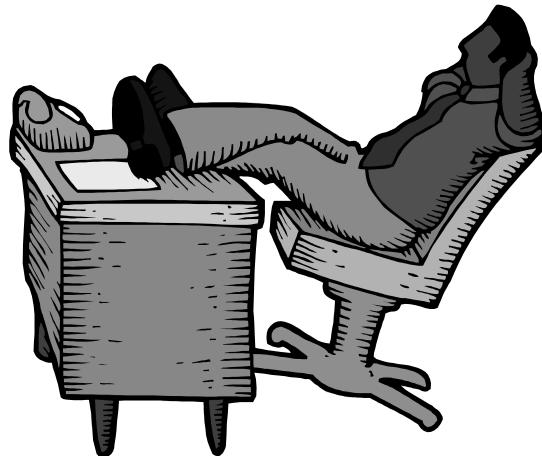
2-10

The Tcl courses above are available in the Solvnet Training Centra, at:
<http://solvnet.synopsys.com/training>

Lab 2: PrimeTime Objects



20 minutes



The objective is to:

■ **Traverse a design from port to clock generator**

- Examine PrimeTime Objects
- Use the data base commands for accessing those objects

2-11

This page was intentionally left blank.

Agenda

**DAY
1**

1 Does Your Design Meet Timing?



2 Objects, Attributes, Collections



3 Constraints in a Timing Report



4 Timing Arcs in a Timing Report



5 Control Which Paths are Reported



Objectives



After completing this lecture, you should be able to:

- Identify constraints in a timing report
- Interpret reports for interface paths

3-2

2 Steps BEFORE Addressing a Violation

① Is this the correct path?



Which details will you use
to determine whether the
reported path is for the
path of interest?

② Are the constraints and SDF
correct?

Ensure that the violation is
real before looking for
resolution!

Determine a possible resolution
for fixing the violation.



3-3

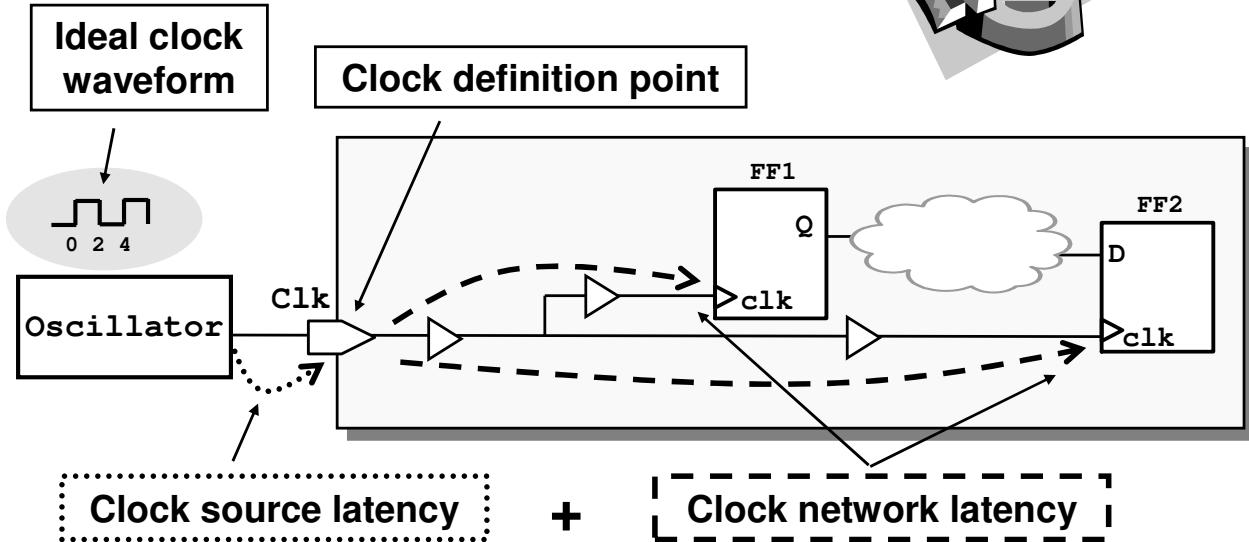
Clocks and Clock Constraints

**Clocks and clock
constraints**

Interface timing paths

3-4

Clock Latency Components



Clock Network Delay
(seen in a default timing report)

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

3-5

Example constraint commands used to create this clock:

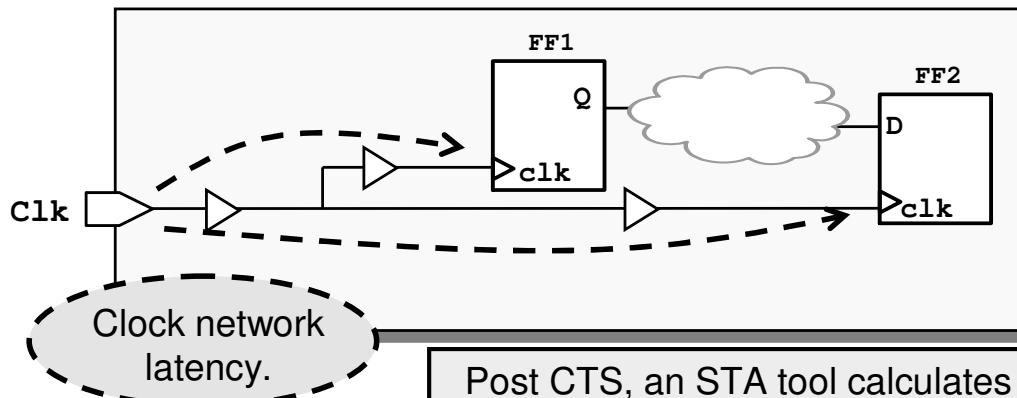
```
create_clock -period 4 [get_ports Clk]  
set_clock_latency -source 2 [get_clocks Clk]  
set_clock_latency 1 [get_clocks Clk]; # For ideal clock network latency
```

Pre Versus Post Clock Tree Synthesis (CTS)

Pre CTS, the user specifies clock network latency.

```
set_clock_latency 1 [get_clocks Clk]
```

Ideal Clocks



Post CTS, an STA tool calculates clock network latency.

```
set_propagated_clock [all_clocks]
```

Propagated Clocks

3-6



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

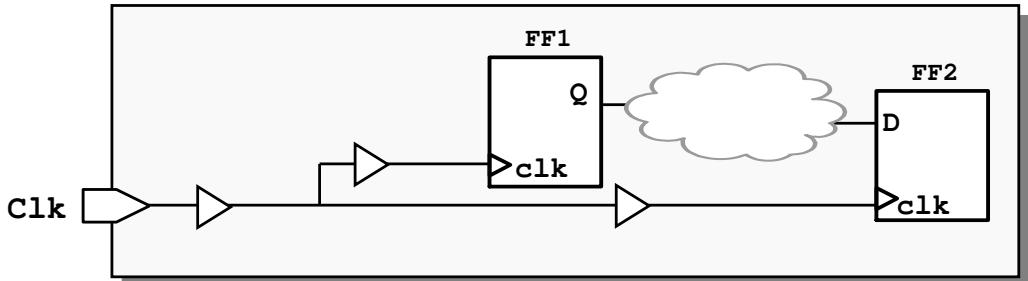
Used prior to clock tree synthesis (CTS). Min/max clock network latency and skew (or uncertainty) are estimated by you.

Propagated clocks

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

Clock Uncertainty and Skew

Clock uncertainty is another constraint specified by you.



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

Pre CTS

Clock Uncertainty = Clock skew + Clock jitter + Margin

Post CTS

Clock Uncertainty = Clock jitter + Margin



Why is clock skew removed post CTS?

3-7

To specify clock uncertainty, use the following command:

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

Test For Understanding 1/3



Circle the clock constraints in this 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 (bufla27)	0.11 *	1.71 r
U3/Y (bufla27)	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

3-8

Test for Understanding 2/3



The clock network delay 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

3-9

* is from SDF

Test for Understanding 3/3



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

3-10

Show All Cells on Clock Network



Circle the difference in the 2nd report below.

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 -path full_clock

Point	Incr	Path
clock Clk (rise edge)	0.00	0.00
clock source latency	0.00	0.00
clk_in (in)	0.00	0.00 r
clk_iopad/PAD (pc3d01)	1.00 *	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

3-11

Timing Report with Schematic

report_timing -path full_clock

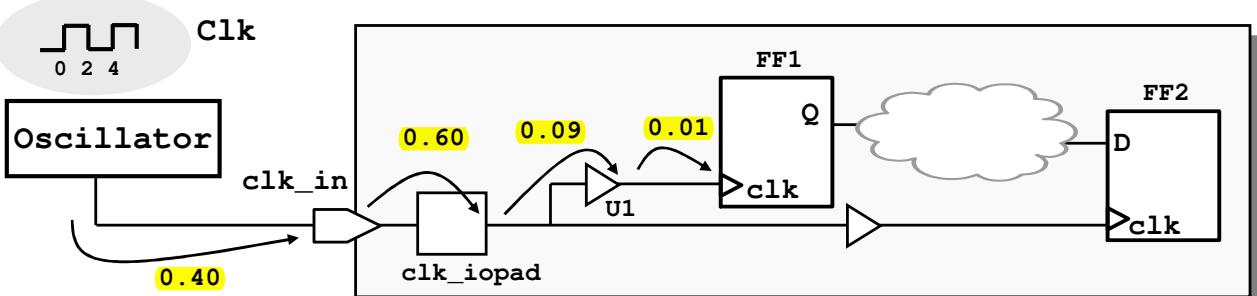
Point	Incr	Path
clock Clk (rise edge)	0.00	0.00
clock source latency	0.40	0.40
clk_in (in) <.....	0.00	0.40 r
clk_iopad/PAD (pc3d01)	0.60 *	1.00 r
U1/Y (buf1a27)	0.09 *	1.09 r
FF1/CLK (fdef1a15)	0.01 *	1.10 r
FF1/Q (fdef1a15)	0.40 *	1.50 f
U2/Y (buf1a27)	0.05 *	1.55 f
U3/Y (buf1a27)	0.05 *	1.60 f
FF2/D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

Clock definition point

Clock ideal waveform

Clock source latency

Clock network latency



3-12

Clocks and Clock Constraints

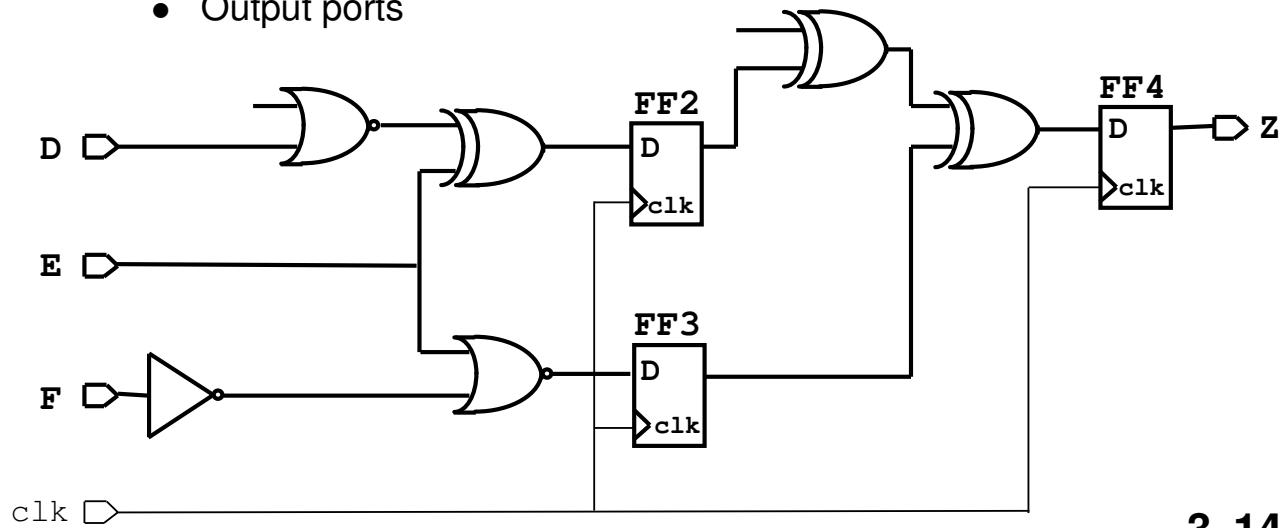
**Clocks and clock
constraints**

Interface timing paths

3-13

Start and End Points

- Timing path start points include:
 - Clock pins of flip-flops
 - Input ports
- Timing path end points include:
 - All input pins of flip-flops except clock pins
 - Output ports



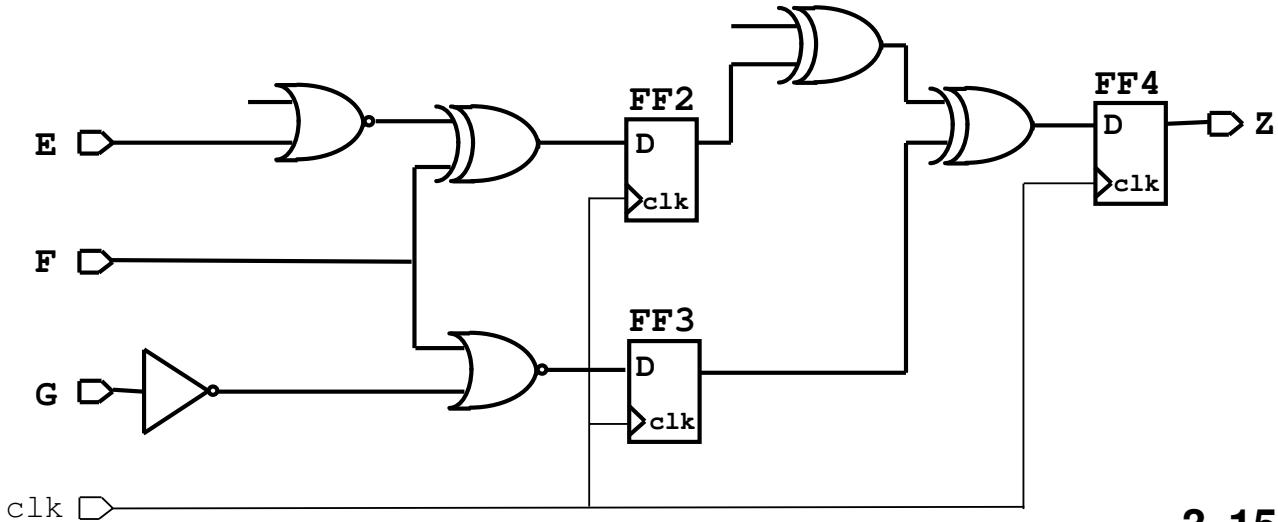
3-14

Generate Timing Reports

Draw the timing path reported.



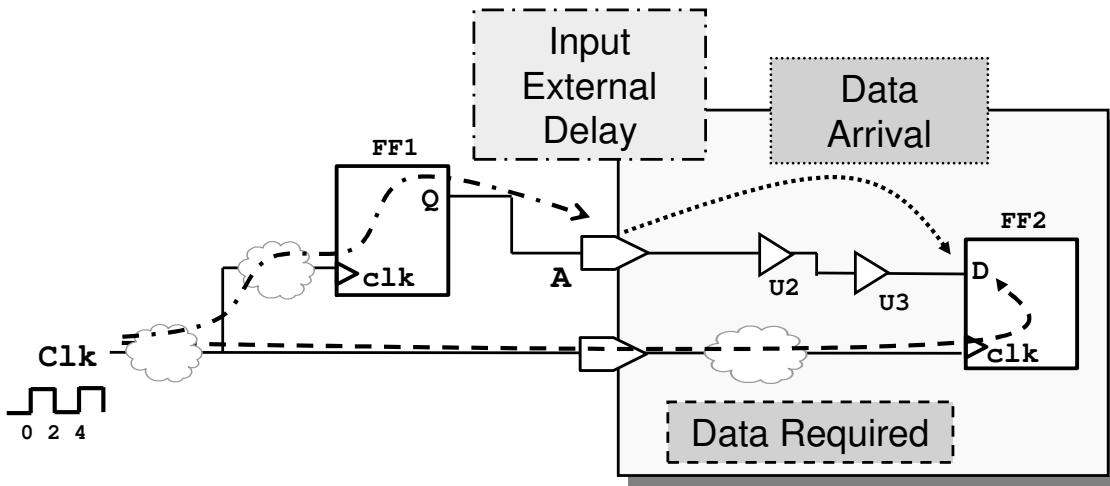
```
report_timing -from E  
report_timing -to Z  
report_timing -from G -to FF4/D
```



3-15

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.

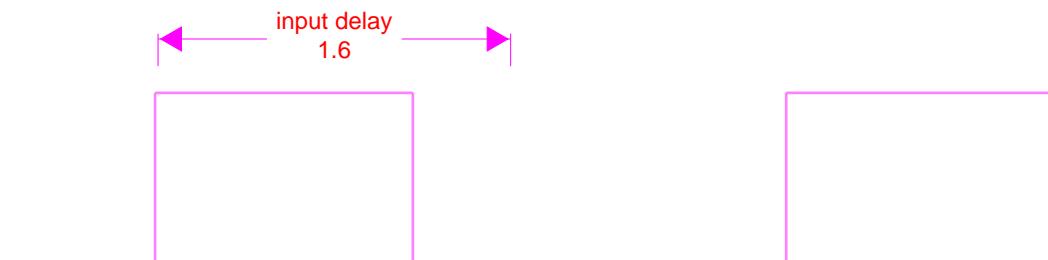


```
set_input_delay 1.60 -max -clock Clk [get_ports A]
```

3-16

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

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



Test For Understanding



Circle the:



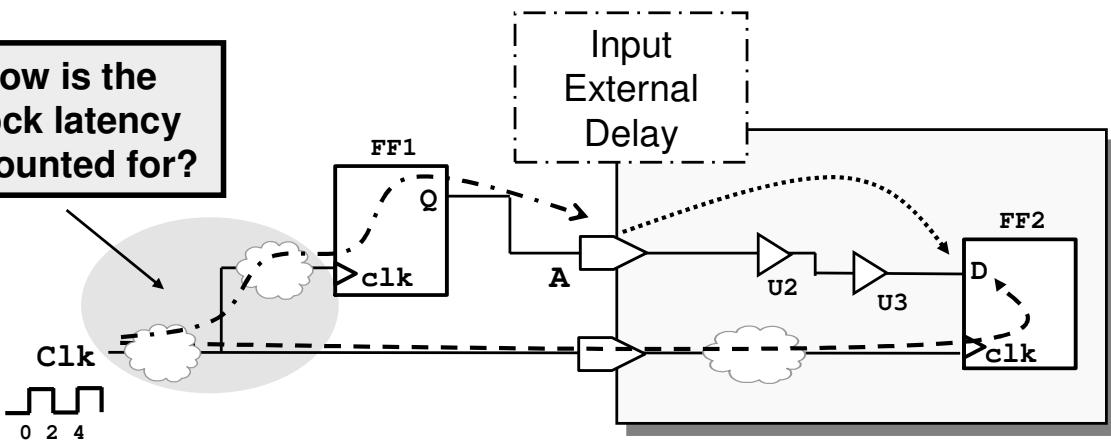
- Input delay constraint.
- Input port name.
- External start point clock.

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 (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
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		-1.87
slack (MET)		2.92

3-17

Gotcha – Constraining Interface Paths

How is the clock latency accounted for?



```
set_input_delay -network_latency_included \
    -source_latency_included -max 1.6 -clock Clk [get_ports A]
```

- One clean solution - include the clock latency as part of the input delay constraint
- When debugging violations on interface paths – discuss how the clock latencies are being represented:
 - Additional examples on day 3 of this workshop

3-18

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

```
set_input_delay 1.60 -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.

Test For Understanding



Explain:

Where must the external clock latency be included?

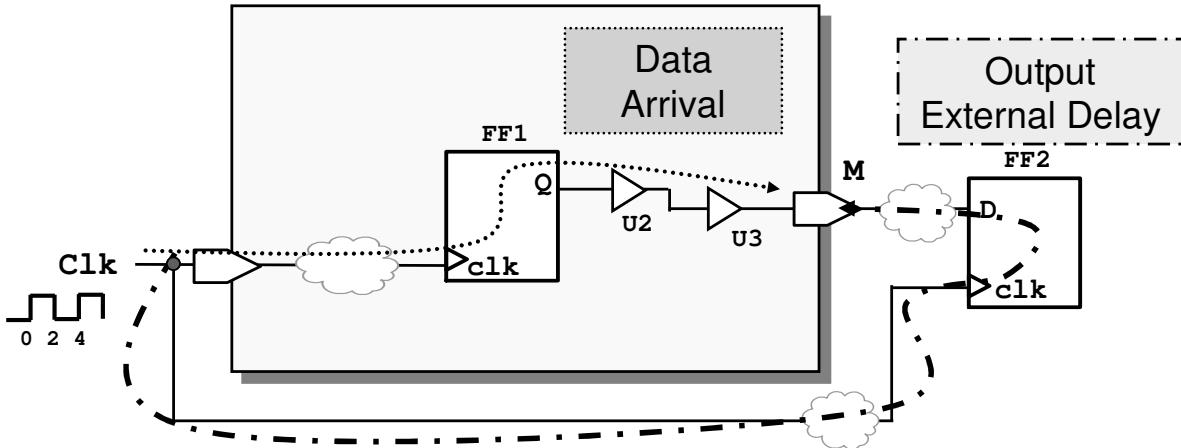
Why is there no “**” designation for the input external delay?

Point	Incr	Path
clock Clk (rise edge)	0.00	0.00
clock network delay (propagated)	0.00	0.00
input external delay	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 (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
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		-1.87
slack (MET)		2.92

3-19

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.

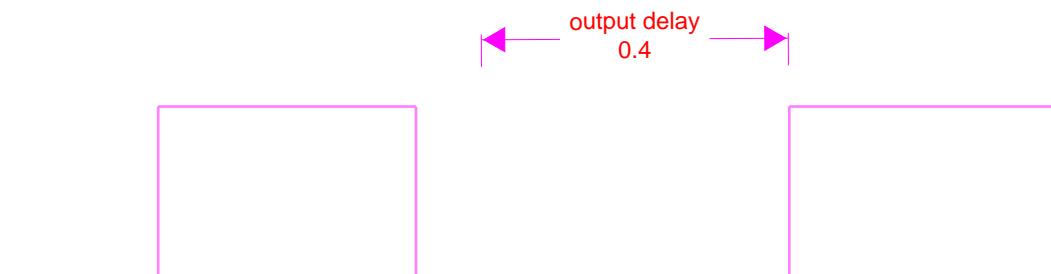


```
set_output_delay 0.40 -max -clock Clk \
    -network -source [get_ports M]
```

3-20

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

```
set_output_delay 0.40 -max -clock Clk \
    -network_latency_included \
    -source_latency_included [get_ports M]
```



Test For Understanding 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

3-21

Test For Understanding 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

3-22

Constrain Output Paths for Hold



Circle output delay constraints.

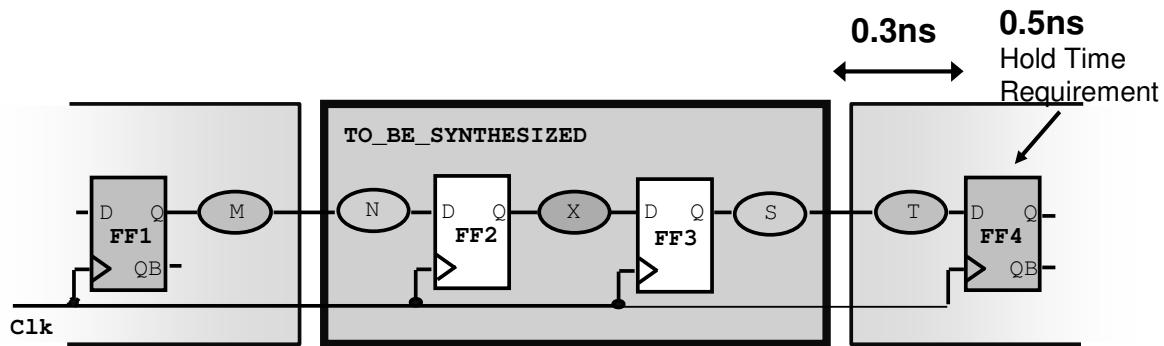
```
pt_shell> report_analysis_coverage
```

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%)

- Validate that the outputs are constrained for setup and hold

3-23

Constrain for Minimum Delay: Output Paths



■ `set_output_delay -min:`

- Describes the minimum time requirement of the external logic on the output ports

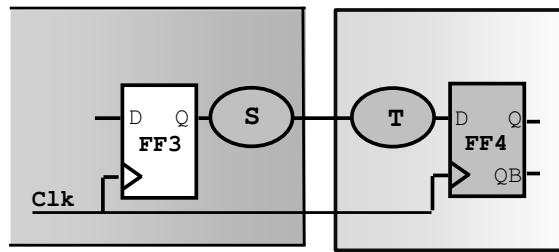


If FF has $T_{HOLD} = 0.5\text{ns}$ and $T_T = 0.3\text{ns}$:
What is the min `output_delay`? _____

3-24

If logic cloud T uses 0.3 ns, you need to have a minimum delay for FF3 (Clk→Q) plus logic cloud S of 0.2 ns in order not to violate the hold time requirement on FF4.

Calculation of Output Delay Time



$$T_{\max} = 1.7$$

$$T_{\min} = 0.3$$

$$FF4_{\text{setup}} = 0.8$$

$$FF4_{\text{hold}} = 0.5$$

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

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

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

3-25

Summary of Example Constraints

- **Clock Specification**

```
create_clock -period 4 [get_ports Clk]
set_clock_latency -source 2 [get_clocks Clk]
set_clock_latency 1 [get_clocks Clk] ;# ideal OR
set_propagated_clock [all_clocks] ; # propagated
set_clock_uncertainty 0.4 [get_clocks Clk]
```

- **Input Arrival Time**

```
set_input_delay 1.60 -max -network \
               -source -clock Clk [get_ports A]
```

- **Output Setup Time**

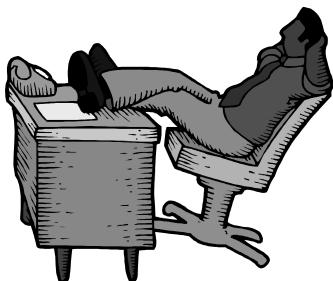
```
set_output_delay 0.40 -max -clock Clk \
                -network_latency_included \
                -source_latency_included [get_ports M]
set_output_delay -min -0.2 -clock Clk [all_outputs]
```

3-26

Lab 3: Constraints in a Timing Report



30 minutes



10 Minute BREAK
30 Minute LAB

The objective is to identify and interpret constraints in a timing report.

3-27

Lab 3 Wrap Up: Constraints on Ports

inout port

```
pt_shell> report_port -input_delay -output_delay pad[0]
```

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

Output Delay

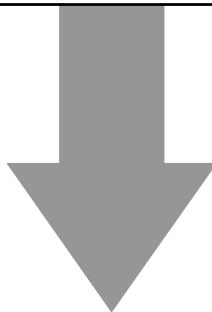
Output Delay						
Output Port	Min	Max	Related Clock	Related Pin		
Rise	Fall	Rise	Fall	Clock	Pin	
pad[0]	-1.00	-1.00	4.00	4.00	PCI_CLK	--

3-28

Lab 3 Wrap Up: Recall the Methodology

① Is this the correct path?

② Are the constraints and SDF
correct?



Determine a possible resolution
for fixing the violation.

3-29

This page was intentionally left blank.

Agenda

**DAY
1**

1 Does Your Design Meet Timing?



2 Objects, Attributes, Collections



3 Constraints in a Timing Report



4 Timing Arcs in a Timing Report



5 Control Which Paths are Reported



Objectives



After completing this lecture, you should be able to:

- Interpret data and clock transitions in a timing report
- Split net and cell timing arc delays in a timing report
- Generate a timing report for a specific data transition at the end point
- Generate a report for library timing arcs

4-2

What Are Timing Arcs?

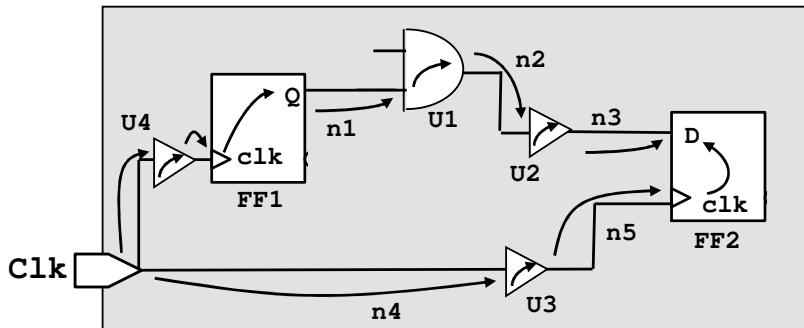
Circle one of each type of timing arcs:



Net delay

Cell delay

Flip-flop setup/hold



■ Cell timing arcs are defined in the library:

- Delay
- Timing check

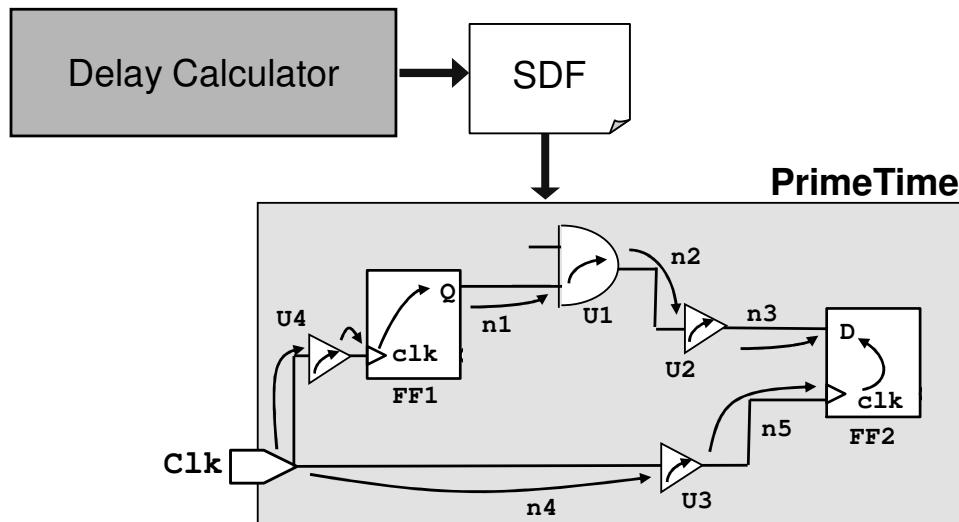
■ Net timing arcs are defined by the netlist

4-3

What is SDF (Standard Delay Format)?

- **SDF contains delays for every timing arc:**

- This includes flip-flop setup and hold!
- Does not contain net RC information



4-4



SDF (Open Verilog International) Standard Delay Format

SDF contains: Net and cell delays; Timing checks; Conditional arcs (A leaf cell timing arc could have different delays depending on the state (high or low) of other input pins on the same leaf cell). PrimeTime accepts SDF v1.0, v2.0, v2.1 and v3.0.

SDF does not contain parasitic RCs or cell transition time as a discernable number:

- You can not perform design rule analysis.
- You can not report correct capacitance or transition information in timing reports.

PrimeTime will perform delay calculation if SDF is not provided. Other Synopsys tools which will generate SDF include Astro and Physical Compiler.

Negative SDF delays are accepted by PrimeTime. Negative SDF delays may warrant attention if they are the result of delay calculations performed outside the cell characterization range.



For further reading, refer to PrimeTime User Guide: Advanced Timing Analysis, unit 6
“SDF Back-Annotation.”

Group Exercise



What does 0.05 represent below?

- A net SDF delay plus a cell SDF delay.
- Just a cell delay.

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

4-5

pt_shell> man report_timing

Symbol	Annotation
-----	-----
H	Hybrid annotation
*	SDF back-annotation
&	RC network back-annotation
\$	RC pi back-annotation
+	Lumped RC
<none>	Wire-load model or none

Include 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

4-6

Let's Talk About Transitions

r: rise
f: fall

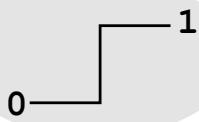
Point	Incr	Path
clock Clk (rise edge)	0.00	0.00
clock network delay (propagated)	1.10 *	1.10
FF1/CLK (fdef1a15)	0.00	1.10 r
FF1/Q (fdef1a15)	0.40 *	1.50 f
U2/Y (buf1a27)	0.05 *	1.55 f
U3/Y (buf1a27)	0.05 *	1.60 f
FF2/D (fdef1a15)	0.01 *	1.61 f
data arrival time		1.61
clock Clk (rise edge)	0.00	0.00
clock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



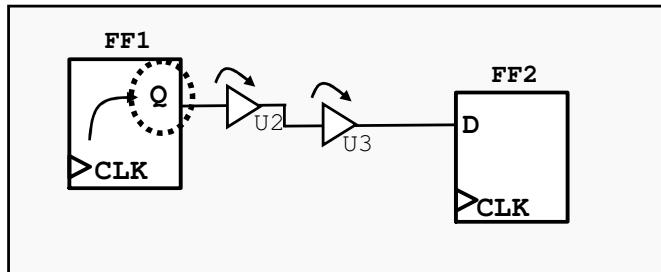
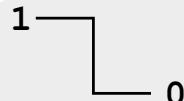
4-7

Cell Delay – Rise and Fall

The data at FF1/Q could transition from



OR



The cell delay from CLK to Q of FF1 is typically:

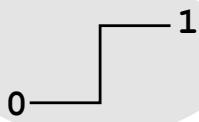


- Different for a rise vs. a fall transition.
- The same for a rise vs. fall transition.

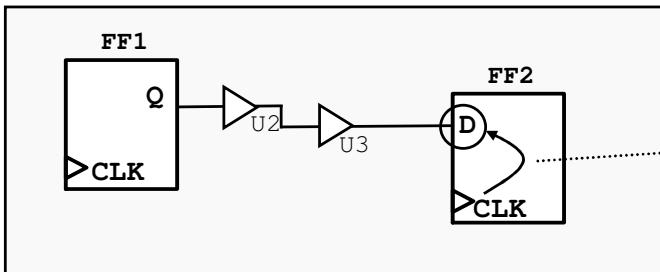
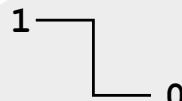
4-8

Cell Timing Checks: Rise and Fall

The data at FF2/D could transition from



OR



Setup or hold timing checks

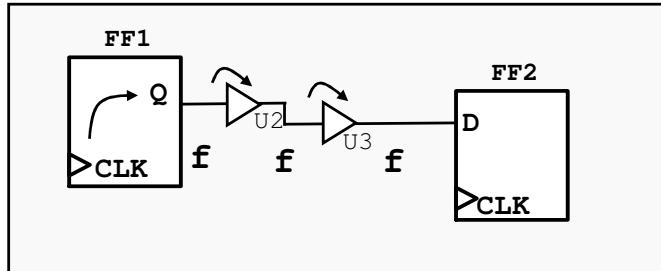
The setup or hold time for FF2:



- Can be different for a rise versus a fall transition.
- Are always the same for a rise versus fall transition.

4-9

Rise and Fall in a Timing Report



The “r” and “f” reports the transition used at the corresponding pin.

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

4-10

Test for Understanding 1/2



An “r” is shown on this line because?



- FF1 is a rising edge triggered flip-flop.
 This causes the worst arrival time.

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

4-11

Test for Understanding 2/2



Markup Tools

The transition at the data pin of the end point flip-flop:

- Affects the value of the library hold time.
- Has nothing to do with the library hold time.

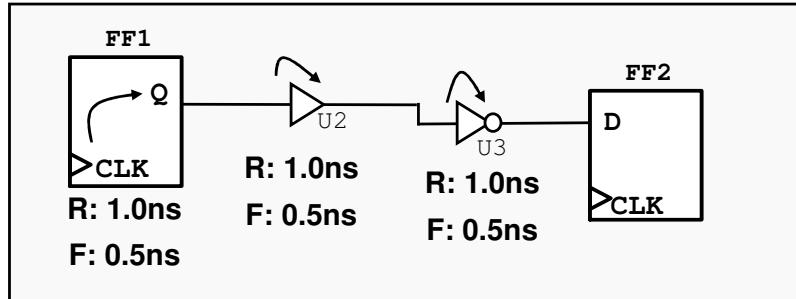
Point	Incr	Path
clock Clk (rise edge)	0.00	0.00
clock network delay (propagated)	1.10 *	1.10
FF1/CLK (fdef1a15)	0.00	1.10 r
FF1/Q (fdef1a15)	0.40 *	1.50 f
U2/Y (buf1a27)	0.05 *	1.55 f
U3/Y (buf1a27)	0.05 *	1.60 f
FF2/D (fdef1a15)	0.01 *	1.61 f
data arrival time		1.61
clock Clk (rise edge)	0.00	0.00
clock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

4-12

Data Arrival Time: Edge Sensitivity 1/2

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

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



What is the longest data arrival time?

3.0ns 2.5ns

What is the shortest data arrival time?

2.0ns 1.5ns

The total # of possible data arrival times for setup is?

1 2 4

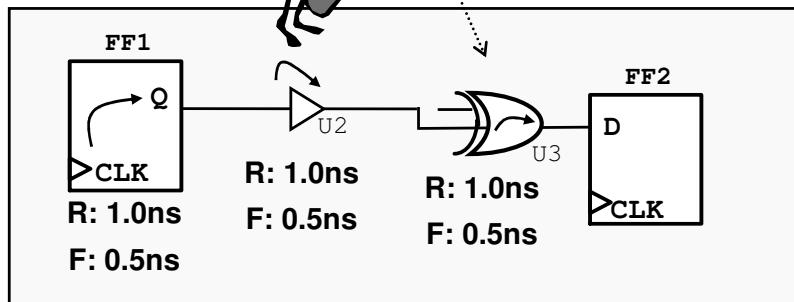
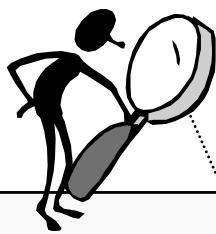


4-13

Data Arrival Time: Edge Sensitivity 2/2

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

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



What is the longest data arrival time?

- 3.0ns 2.5ns

What is the shortest data arrival time?

- 2.0ns 1.5ns

The total # of possible data arrival times for setup is?

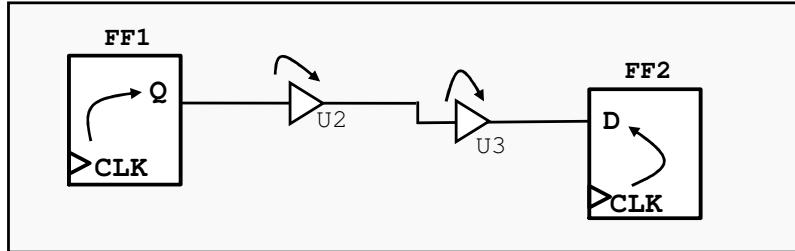
- 1 2 4



4-14

Default Timing Reports – Worst Slack

By default, `report_timing` generates a report for the worst slack for setup time.



Which will always result in the worst slack for setup time?



- The longest data arrival time.
- The shortest data arrival time.
- It is unclear as the slack is a function of data arrival time as well as data required time.

Example in lab.

4-15

Generate Worst Slack Timing Report

```
report_timing -to FF2/D
```

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)	0.50 *	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
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		-1.87
slack (MET)		2.92

The worst slack

r: 0.21 setup time
f: 0.15 setup time

4-16

Generate Timing Reports for Rise or Fall

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																													
<table><thead><tr><th>Point</th><th>Incr</th><th>Path</th></tr></thead><tbody><tr><td>clock Clk (rise edge)</td><td>0.00</td><td>0.00</td></tr><tr><td>clock network delay (propagated)</td><td>1.10 *</td><td>1.10</td></tr><tr><td>FF1/CLK (fdef1a15)</td><td>0.00</td><td>1.10 r</td></tr><tr><td>FF1/Q (fdef1a15)</td><td>0.40 *</td><td>1.50 f</td></tr><tr><td>U2/Y (buf1a27)</td><td>0.05 *</td><td>1.55 f</td></tr><tr><td>U3/Y (buf1a27)</td><td>0.05 *</td><td>1.60 f</td></tr><tr><td>FF2/D (fdef1a15)</td><td>0.01 *</td><td>1.61 f</td></tr><tr><td>data arrival time</td><td></td><td>1.61</td></tr></tbody></table>			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
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																											
<table><tbody><tr><td>clock Clk (rise edge)</td><td>4.00</td><td>4.00</td></tr><tr><td>clock network delay (propagated)</td><td>1.00 *</td><td>5.00</td></tr><tr><td>FF2/CLK (fdef1a15)</td><td></td><td>5.00 r</td></tr><tr><td>library setup time</td><td>-0.15 *</td><td>4.85</td></tr><tr><td>data required time</td><td></td><td>4.85</td></tr></tbody></table>			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												
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																											
<table><tbody><tr><td>data required time</td><td></td><td>4.85</td></tr><tr><td>data arrival time</td><td></td><td>-1.61</td></tr></tbody></table>			data required time		4.85	data arrival time		-1.61																					
data required time		4.85																											
data arrival time		-1.61																											
<table><tbody><tr><td>slack (MET)</td><td>r: 0.21 setup time f: 0.15 setup time</td><td>3.24</td></tr></tbody></table>			slack (MET)	r: 0.21 setup time f: 0.15 setup time	3.24																								
slack (MET)	r: 0.21 setup time f: 0.15 setup time	3.24																											

4-17

Larger positive slack

```
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)
```

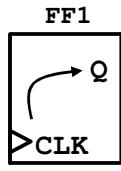
Common Types of Timing Arcs



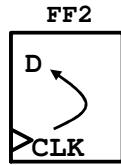
positive_unate



negative_unate



rising_edge



hold_clk_rise

setup_clk_rise



positive_unate

and

negative_unate



List more examples of
“non unate” timing arcs?

4-18

Report Library Arcs: Inverter

Reference name
Library name

```
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
```



**What command will list
all libraries in
PrimeTime memory?**

4-19

```
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
```

Report Library Arcs: Flip-Flop

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

4-20

The function of each pin is as follows:

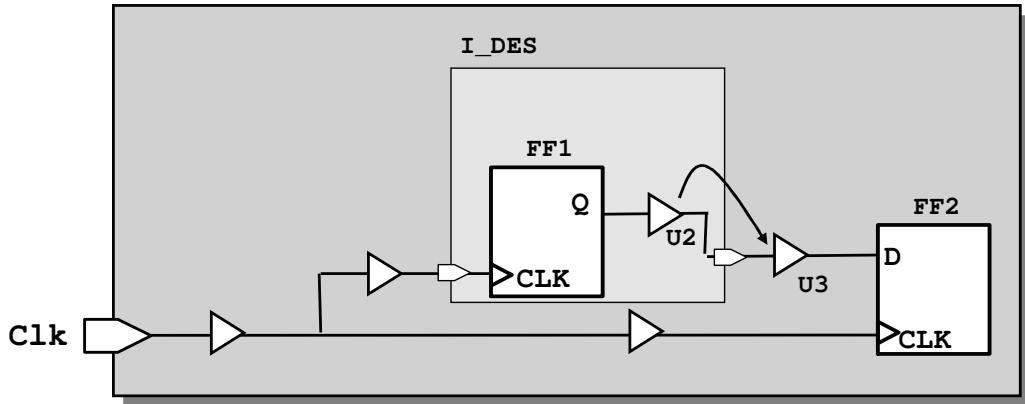
- CDN** Asynchronous clear, asserted low
- CP** Clock
- D** Data input
- SC** Scan enable input
- SD** Scan chain input
- Q** Output pin

Last Topic - Hierarchy in a Timing Path

Hierarchy does not “block” STA.

Hierarchy does not “break” net timing arcs.

The hierarchy along a path is reflected in the timing report.



What delay do you expect to see for
hierarchical pins in a timing report?

4-21

Identify Hierarchy in a Timing Report

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 f
I_DES/A (MYDES)	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

Hierarchy pin name → **I_DES/A (MYDES)**

Look for zero delay → **0.00 ***

Design name → **I_DES/A (MYDES)**

Net + cell delay

4-22

Lab 4: Timing Arcs in a Timing Report



40 minutes



10 Minute BREAK
40 Minute LAB

The objective is to:

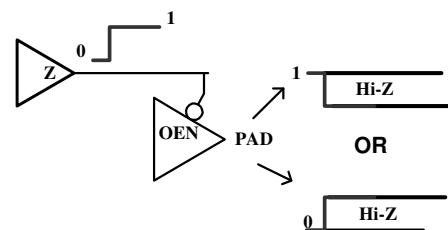
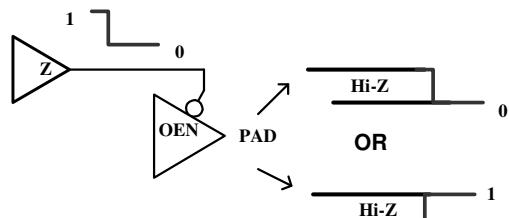
- Interpret timing arcs and transitions in a timing report
- Generate a timing report with a specific data transitions at the end point
- Generate a timing report with input pins included
- Generate a report for library timing arcs

4-23

Lab 4 Wrap Up

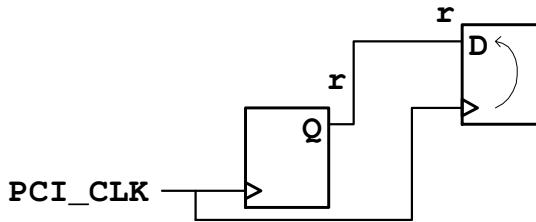
Generate a report with
input pins and library
timing arcs.

Non unate timing arc
from OEN > PAD



Generate a report using
fall delays.

Data arrival time calculated with rise
transitions results in a worse slack for hold!



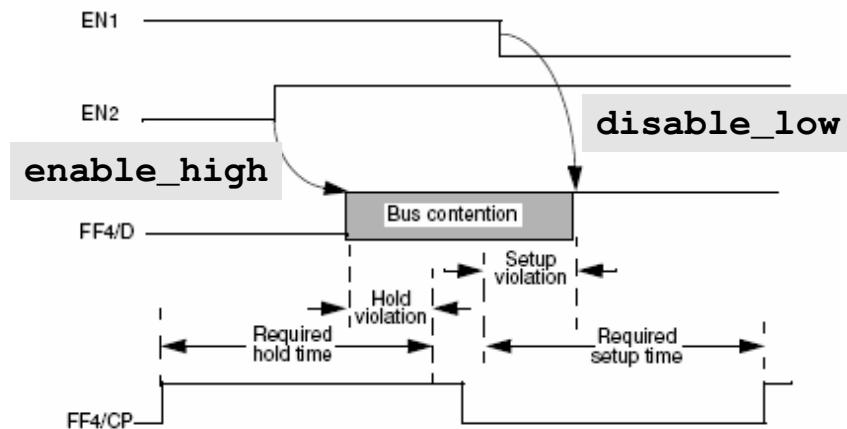
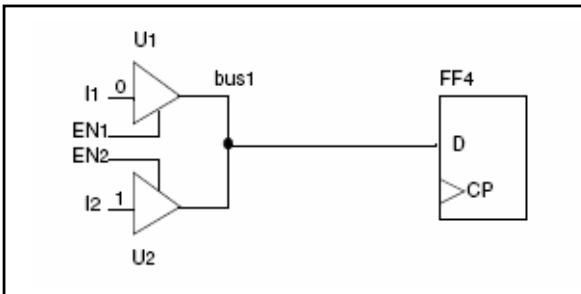
4-24

Appendix

Bus Contention

Floating Bus

Tri-State Timing Arcs: Bus Contention

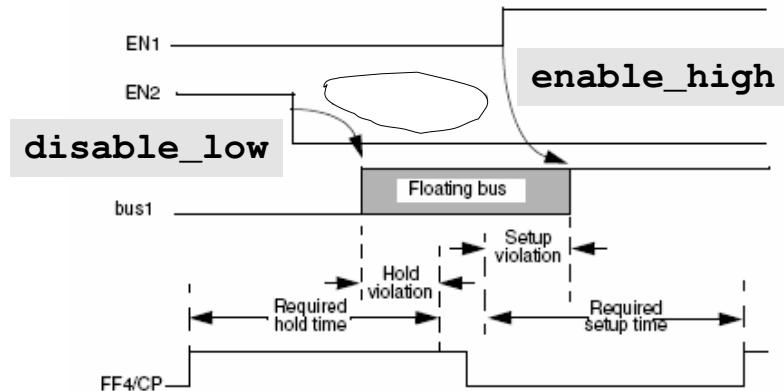
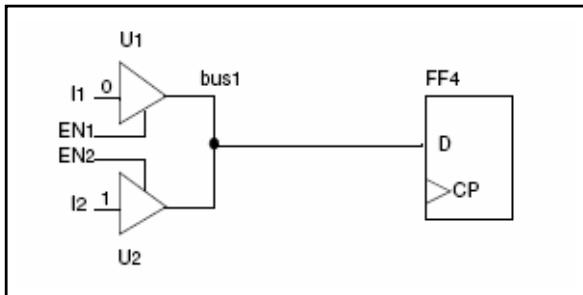


4-26



For further explanation of the drawings above, refer to the PrimeTime User Guide Advanced Timing Analysis, unit 5 “Advanced Analysis Techniques” under Three State Bus Analysis.

Tri-State Timing Arcs: Floating Bus



4-27



For further explanation of the drawings above, refer to the PrimeTime User Guide Advanced Timing Analysis, unit 5 “**Advanced Analysis Techniques**” under **Three State Bus Analysis**.

This page was intentionally left blank.

Agenda

**DAY
1**

1 Does Your Design Meet Timing?



2 Objects, Attributes, Collections



3 Constraints in a Timing Report



4 Timing Arcs in a Timing Report



5 Control Which Paths are Reported



Objectives



After completing this lecture, you should be able to:

- **Apply the appropriate switches for `report_timing` to control which and how many paths are reported**
- **Apply netlist navigation commands to explore pins, ports and clocks of interest**

5-2

Default Behavior of report_timing

The command `report_timing`
generates one report
with the worst slack
for each path group (e.g. capture clock name)
for setup time.



For any given design, the number of paths reported by the default `report_timing` is equal to:

- 1.
- The number of unique “capture” clocks in the design.

5-3

Explore and Debug – Generate More Reports

`report_timing`

Explore and debug violations
through a larger window
(i.e. more timing reports).

Gain insight on specific design
issues with positive margin.

External processing of
many reports.

5-4

Control Which and How Many Reports

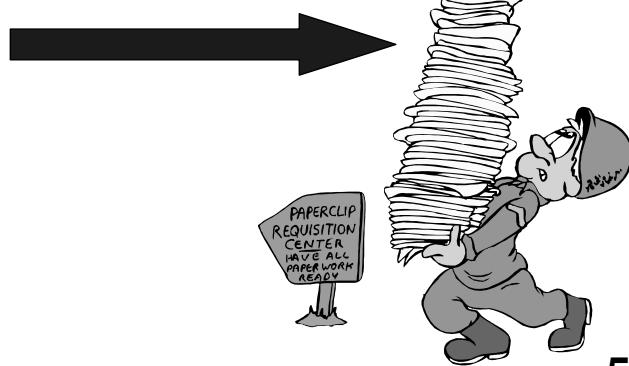


What switch controls if a setup or hold report is generated?

Reports for hold versus setup

More reports

To/From/Through specific pins



5-5

How Many Paths Would You Like Reported?

`report_timing`

`-max_paths` Number of paths to report for each path group.

`-nworst` Consider this number of paths per endpoint.

(Default values are 1.)

5-6

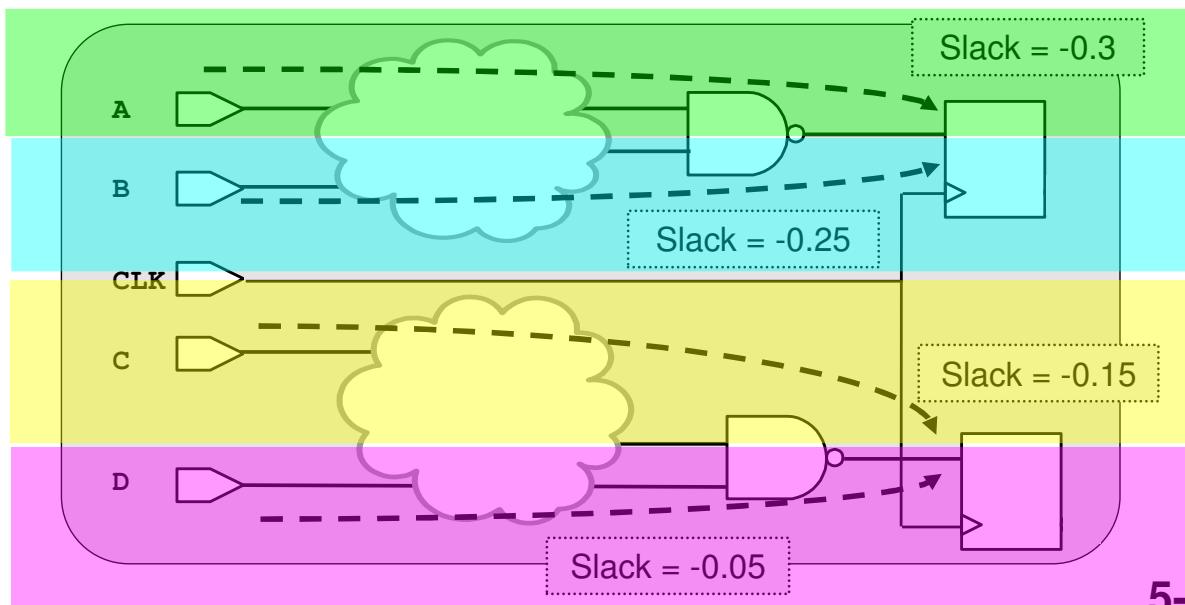
Setup: Exercise on Nworst and Max_Paths

Given

There are two endpoints in this design.

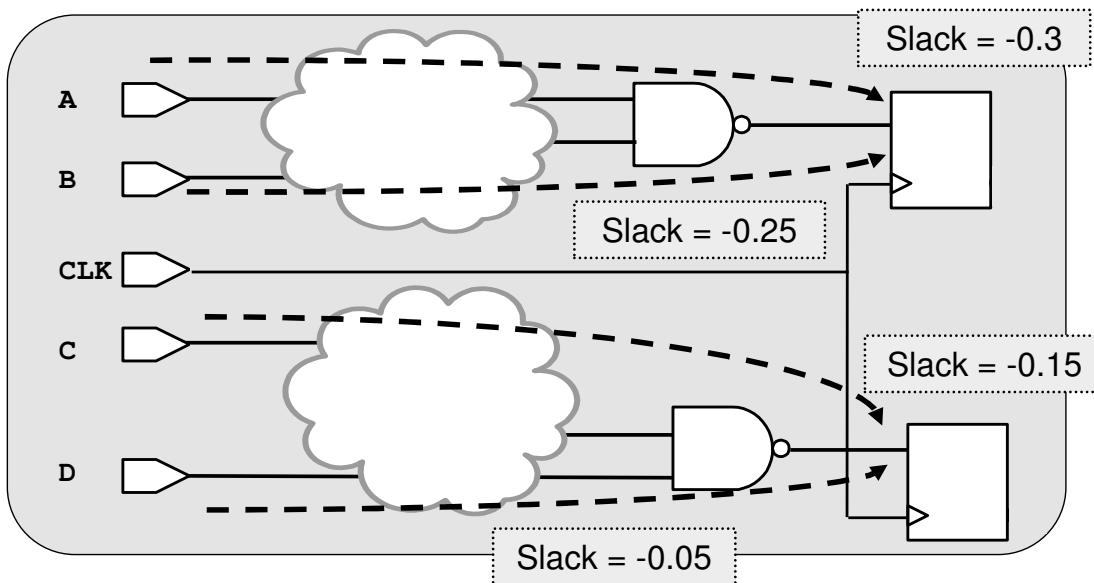
Shown are the two worst slack values for each endpoint.

There is one defined clock.



5-7

Exercise on Nworst and Max_Paths



```
report_timing -max_paths 2 -nworst 1
```

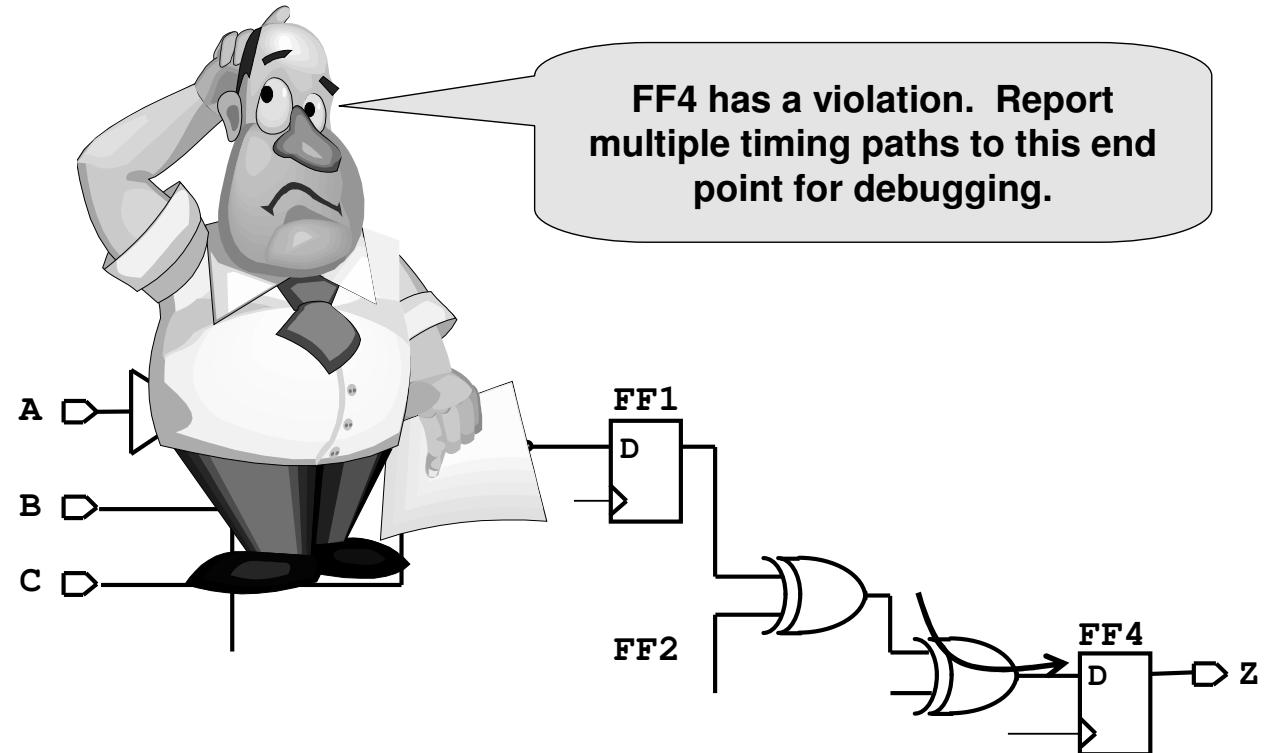
```
report_timing -max_paths 2 -nworst 2
```



Circle the
reported slacks.

5-8

Report Many Paths to a Single Endpoint



5-9

Test For Understanding

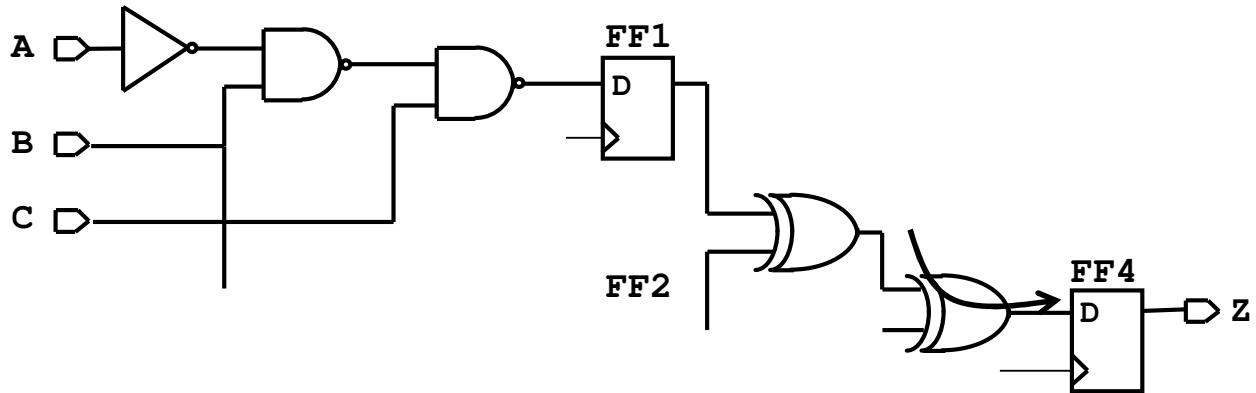


```
report_timing -max_paths 4 -to FF4/D
```

How many paths will be reported?



- 4 reports with the worst slack.
- 1 (only one path per endpoint will be considered).



5-10

Filter Further Using More Switches

`report_timing`

`-max_paths` Number of paths to report for each path group.

`-nworst` Considering this number of paths per endpoint.

`-group` For this path group (e.g. capture clock name).

`-slack_lesser_than` ← (more negative than)



`-slack_greater_than` (more positive than) →

5-11

Control Which and How Many Reports

Reports for setup
versus hold

More reports

To/From/Through
specific pins



5-12

From, To or Through . . .

`report_timing`

`-to -rise_to -fall_to`

`-from -rise_from -fall_from`

`-through -rise_through -fall_through`

`-exclude -rise_exclude -fall_exclude`

List examples of



Start points

End points

Through points

5-13

The three switches for excluding nets, cells, pins or ports when generating a timing report were new in v2004.12.

Clock Ports versus Clock Objects

report_timing -from [get_ports clk] will return:

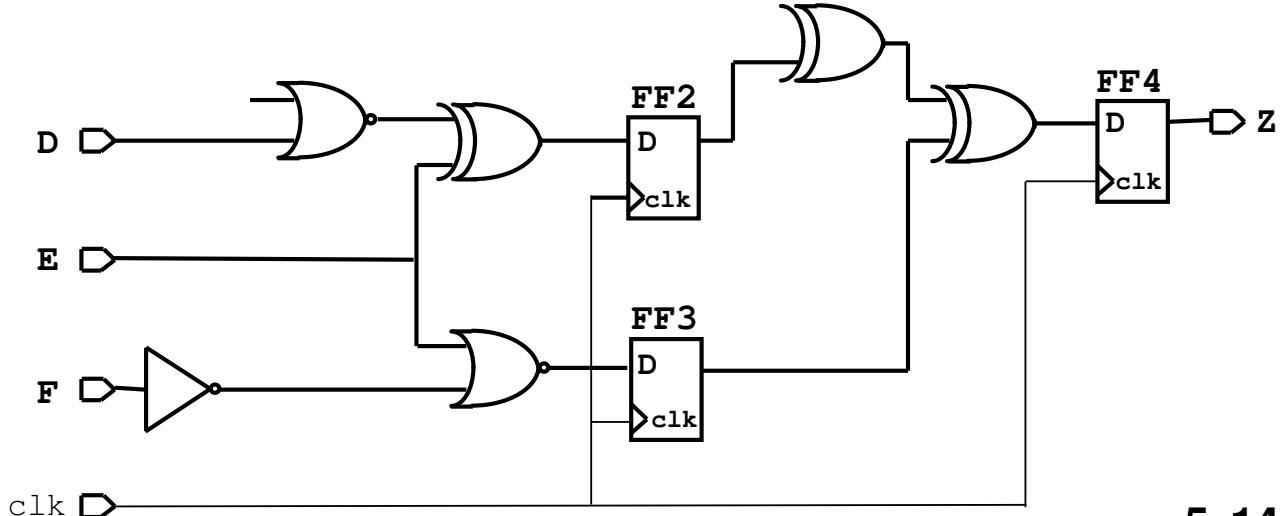
- Nothing – this is not a timing path.
- I do not know.

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.



Markup Tools

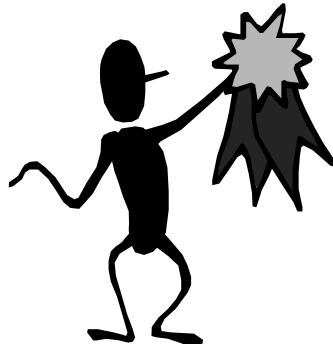


5-14

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
report_clock_timing # Report clock timing info
[-clock clock_list]          (Clock network(s) of interest)
[-from_clock from_clock_list] (From clock network(s) of interest)
[-to_clock to_clock_list]     (To clock network(s) of interest)
[-from from_list]            (From pins in the clock network)
[-to to_list]                (To pins in the clock network)
[-verbose]                   (Display the report in verbose format)
-type report_type            (Generate this type of report: Values: skew,
                               interclock_skew, latency, transition, summary)
[-slack_lesser_than slack_upper_limit] (Filter skew entries)
[-lesser_than upper_limit]      (Filter entries less than this)
[-greater_than lower_limit]    (Filter entries greater than this)
[-nworst worst_entries]        (List N worst entries: Value >= 1)
[-rise]                       (Show entries with rising transition)
[-fall]                        (Show entries with falling transition)
[-capture]                     (Show entries with capture transition)
[-launch]                      (Show entries with launch transition)
[-setup]                       (Display entries for setup paths between latches)
[-hold]                        (Display entries for hold paths between latches)
[-include_uncertainty_in_skew]  (Include uncertainty as a skew component)
. . .
```

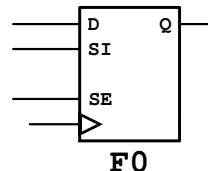
Recommendation – Be Specific



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

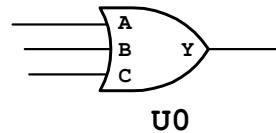
Best runtime.

Get the intended timing paths.



```
report_timing -to F0/D
```

```
report_timing -through U0/A
```



Which command(s) will report cell pin names?

5-15



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

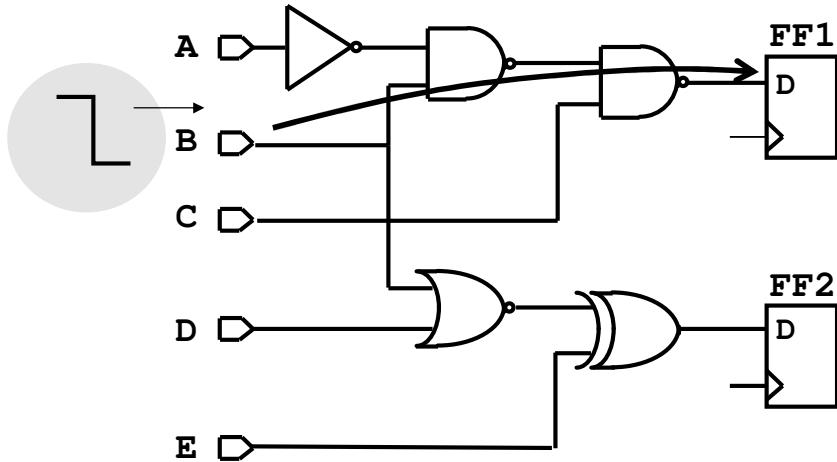
  Input Pins          Net             Net Driver Pins   Driver Pin Type
  -----              -----           -----           -----
  I                  net_pad_en     I_ORCA_TOP/I_PCI_CORE/pad_en_reg/Q
                                Output Pin (sdcrq1)

  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 #1

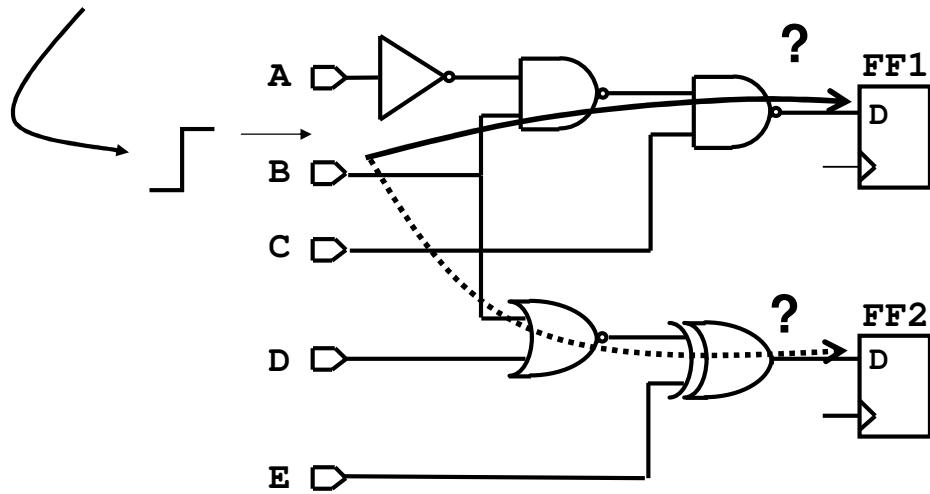
FF1 has the worst slack in the design for setup.



5-16

Explore a Rising Data Transition

```
report_timing -rise_from B
```



Which path will be reported?

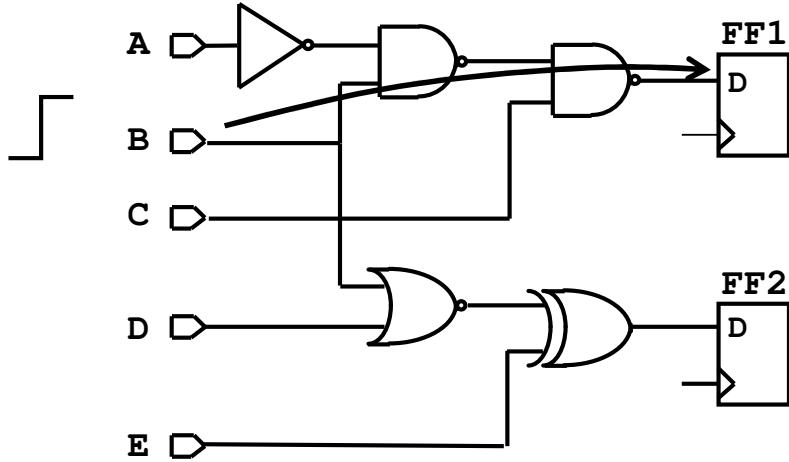


- The same path as before, to FF1.
- Whichever path offers the worst slack.

5-17

Add More Switches

```
report_timing -rise_from B -to FF1/D
```



How will you validate the generated report is for the intended path?

5-18

Example #2



The default timing report is to the rising edge of CLK1.

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

5-19

Required - Use `get_clocks`

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

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

Embed commands.

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

5-20

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 #3



You want the worst slack
through a bus. The net
names are `sd_DQ*`.

5-21

Use * as a Wildcard

Represents 0-N characters

report_timing -through sd_DQ*

How many paths will be reported?



- 1 with the worst slack.
- 1 for each unique capture clock.

5-22

Useful Commands for Navigation

All the start point pins to a specific end point?

```
all_fanin -flat -start -to FF1/D
```

Find all input ports constrained by a clock?

```
all_inputs -clock CLK1
```

Find all pin names for a specific cell?

```
get_pins -of_object U1
```

Find all output ports?

```
all_outputs
```

Find the data pin names of all latches?

```
all_registers -level_sensitive -data_pins
```

Find the net connected to a pin?

```
all_connected U1/Z
```



How do you pass the results directly to report_timing?

5-23



For more information and practice with these types of commands, refer to two additional workshops:

The Power of Tcl 3: Direct Access Through Collections and Attributes
PrimeTime: Advanced STA Constraint Debugging

You Would Like To Know More!



How do you find ALL the get_* and all_* commands?



5-24

```
pt_shell> help all_*
all_clocks          # Create a collection of all clocks in design
all_connected       # Create a collection of objects connected to another
all_fanin           # Create a collection of pins/ports or cells in the fanin
all_fanout          # Create a collection of pins/ports or cells in the fanout
all_inputs          # Create a collection of all input ports in design
all_instances        # Create a collection of all instances of a design
all_outputs         # Create a collection of all output ports in design
all_registers        # Create a collection of register cells or pins
pt_shell> help get_*
get_alternative_lib_cells # Create a collection of alternative library cells
get_cells            # Create a collection of cells
get_clocks           # Create a collection of clocks
get_designs          # Create a collection of designs
get_generated_clocks # Create a collection of generated clocks
get_lib_cells        # Create a collection of library cells
get_lib_pins         # Create a collection of library cell pins
get_lib_timing_arcs # Create a collection of library timing arcs
get_libs              # Create a collection of libraries
get_nets              # Create a collection of nets
get_path_groups      # Create a collection of path groups
get_pins              # Create a collection of pins
get_ports             # Create a collection of ports
get_timing_arcs      # Create a collection of timing arcs
get_timing_paths     # Create a collection of timing paths
```

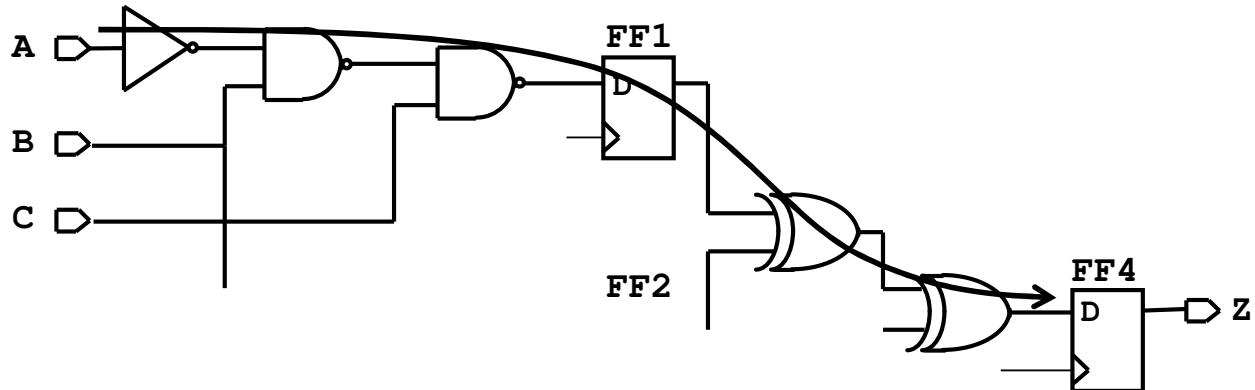
Final Exercise

```
report_timing -from A -to FF4/D
```

How many paths will be reported?



- 1 with the worst slack.
- 0 – no single timing path between these points.



5-25

What Is Returned?

```
pt_shell> report_timing -from A -to FF4/D
```

No constrained paths.



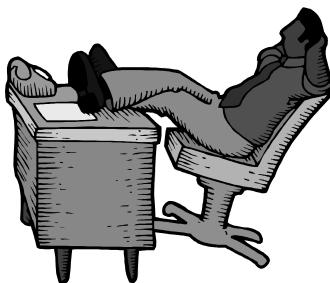
Use the job aid to write the command to find all possible end points from the input port A?

5-26

Lab 5: Control Which Paths are Reported



30 minutes



10 Minute BREAK
30 Minute LAB

The objective is to:

- **Apply report_timing switches to control which and how many paths are reported**
- **Apply netlist navigation commands**

5-27

Lab 5 Wrap Up : Tcl Scripting

In the answer section to lab 5 . . .

```
# The backslash is a line continuation character
pt_shell> report_timing -delay min\(
    -to I_ORCA_TOP/I_BLENDER/latched_clk_en_reg/D
```

```
# When using PrimeTime interactively - abbreviate
# command names or switches by typing enough letters to
# distinguish from other commands or switches
pt_shell> report_timing -slack_less 0
```

5-28

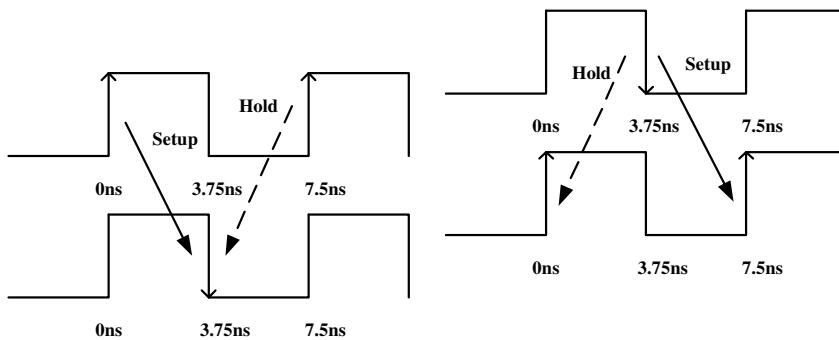
Lab 5 Wrap Up: Half Clock Cycle Paths

```
pt_shell> report_clock SDRAM_CLK
```

Attributes:

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

Clock	Period	Waveform	Attrs	Sources
SDRAM_CLK	7.50	{0 3.75}	p	{sdr_clk}



```
report_timing -delay min_max
```

5-29

This page was intentionally left blank.

Agenda

**DAY
2**

6 Summary Reports



7 Create a Setup File and Run Script



8 Validate a Run Script



9 Getting to Know Your Clocks



Unit 1 – 5 Review (1/2)



Match the questions with the correct answer.

1. Report the total # of violations?
2. Find help on report_timing?
3. Display names of all output ports?
4. Change the number of paths considered for each end point when generating timing reports?
5. A path group name is?
6. Line continuation character?
7. Embed a command?

- A. \
- B. report_analysis_coverage
- C. restore_session unix_dir
- D. report_timing -help;
man report_timing
- E. #
- F. report_timing -nworst
- G. report_timing -max_paths
- H. get_pins -of_object cell
- I. []
- J. all_outputs
- K. Same as the capture clock name

6-2

Unit 1-5 Review (2/2)

- (1) Identify hierarchy in a timing report by:
 - A. Hierarchy is not shown in a default timing report.
 - B. Looking for zero delays in the incremental column.
- (2) A negative output delay constraint for hold will impose a:
 - A. Positive hold requirement on the output port.
 - B. A negative hold requirement on the output port.
- (3) `report_timing -input_pins`:
 - A. Adds input pins to a default timing report (and splits net and cell delays).
 - B. Shows net names and fanout in a default timing report.
- (4) `report_timing -max_paths 4 -nworst 4 -to FF1/D`:
 - A. Generates 16 hold reports to this end point.
 - B. Generates 4 setup reports to this end point.
- (5) List timing path start and end points.
- (6) List the differences between ideal and propagated clocks.

6-3

Objectives

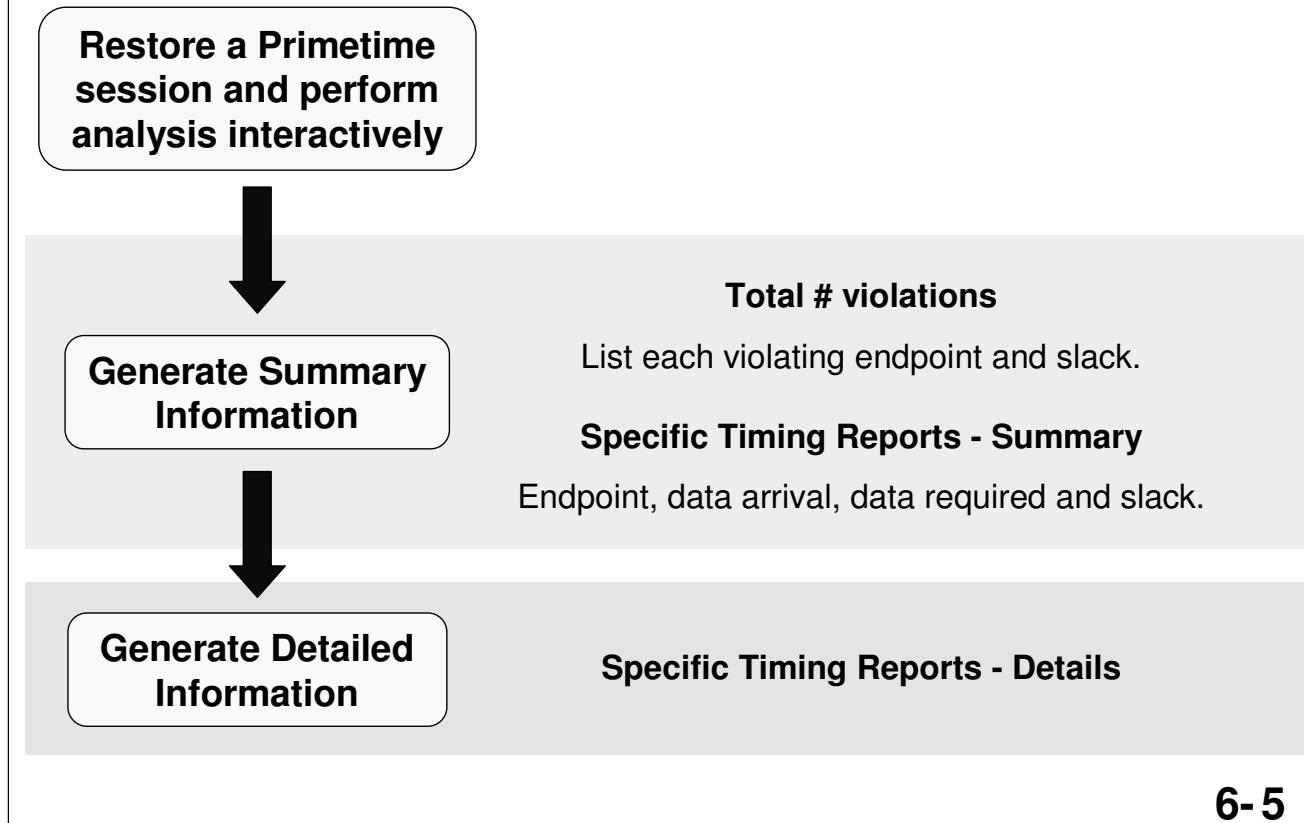


After completing this lecture, you should be able to:

- **Generate summary information for violations sorted by slack, timing check or clock group**
- **Apply one solution for finding a resolution to a timing violation**

6-4

Generating Reports – A Methodology



6-5

Recall - Does Your Design Meet Timing?

```
pt_shell> 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	20 (100%)	0 (0%)	0 (0%)
min_pulse_width	7273	5957 (82%)	0 (0%)	1316 (18%)
out_setup	75	(100%)	0 (0%)	0 (0%)
out_hold	75	(100%)	0 (0%)	0 (0%)
All Checks	29333	53%	88 (0%)	13610 (46%)



You want details for these violations, sorted by slack.

6-6

Apply Appropriate Switches

```
pt_shell> 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!

6-7

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
```

Switches - report_analysis_coverage

report_analysis_coverage

-status violated	Report details on violations.
-------------------------	--------------------------------------

-check {setup hold}	Default includes all timing checks – or focus to specific timing checks of interest.
----------------------------	---

-sort_by slack	Sort by slack first (default).
-----------------------	---------------------------------------

-sort_by check_type	Sort by type of timing check and then by slack.
----------------------------	--

6-8

```
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)
```

Test For Understanding



1. List all violations sorted by slack?



2. List all setup and hold violations sorted by slack?



3. List all setup and hold violations sorted by timing
check first and then sorted by slack?



```
.....  
report_analysis_coverage  
    -status  
    -check  
    -sort_by  
.....
```



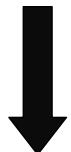
6-9

Report All Violations Sorted By Clock

List all violations, sorted by slack

If desired, sorted by timing checks

```
report_analysis_coverage
```



List all violations, sorted by
clock domain

```
report_constraint \  
-all_violators
```



What are some reasons to organize
violations by clock domain?

6-10

Use report_constraint -all_violators



The violations below are organized by:

- Slack.
- Slack within each path group.

```
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)

6-11

What If . . .



There are hundreds of violations!

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

6-12

Generate Summary report_timing

```
pt_shell> 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_out_1_reg[11]/D (sepfq1)	4.557 r*	4.134	0.423
I_ORCA_TOP/I_SDRAM_IF/DQ_out_1_reg[4]/D (sepfq1)	4.547 r*	4.123	0.424
I_ORCA_TOP/I_SDRAM_IF/DQ_out_1_reg[9]/D (sepfq1)	4.551 r*	4.126	0.425

Sorted by path group
and then by slack!

How many path groups are reported above?



- 1
 Cannot tell!

6-13

Test For Understanding



**Write the command to report a summary
of 10 violations for setup for the clock
SYS_2x_CLK.**



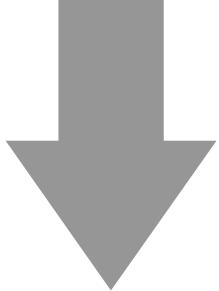
6-14

Find Resolutions for Fixing Violations

✓① Is this the correct path?

✓② Are the constraints and SDF correct?

Determine a possible resolution for fixing the violation.



6-15

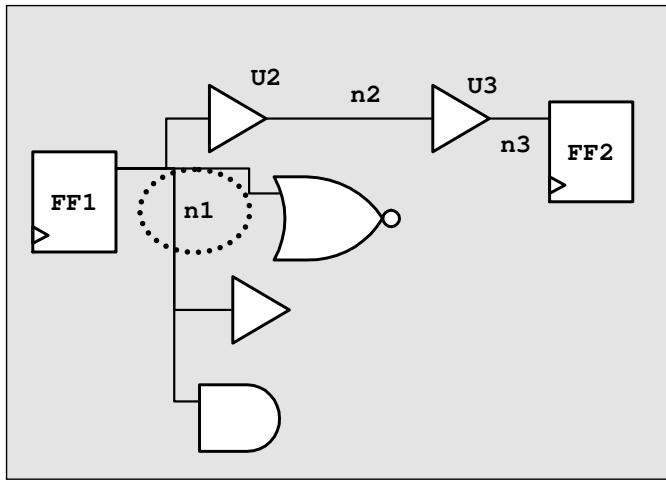
Example - Load Isolation or Upsizing

Violation!

Large delays
associated with
large net fanout?



Load isolation or
upsizing cells.



Step #1

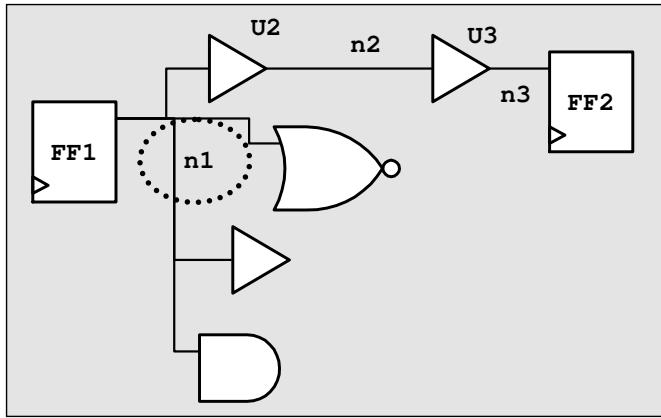
Include net fanout
in timing reports.

6-16

Large Delay Associated with Large Fanout

report_timing(-nets)

Point	Fanout	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)		2..40..*	3.50 r
n1 (net)	4		
U2/Y (buf1a27)		0.15 *	3.65 r
n2 (net)	1		
U3/Y (buf1a27)		0.05 *	3.70 r
n3 (net)	1		
FF2/D (fdef1a15)		0.01 *	3.71 r
data arrival time			3.71



Step #2
Can FF1 be
upsized?

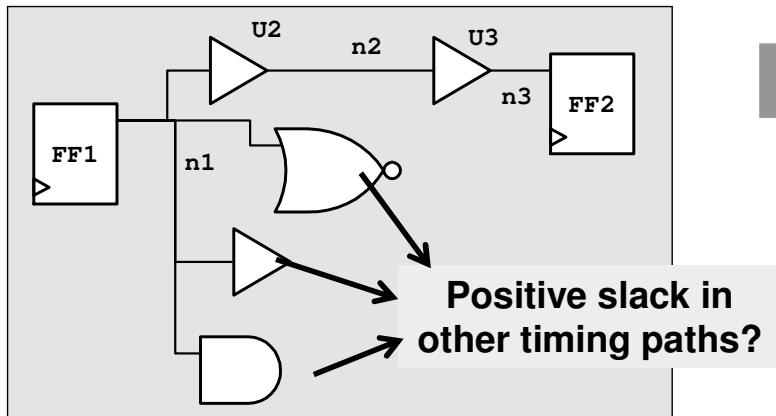
6-17

Reporting Alternative Library Cells

```
pt_shell> report_cell FF1
```

Cell	Reference	Library	Area	Attributes
FF1	fdef1a15	my_lib	7.750	n
Total 1 cells				7.750

```
pt_shell> get_alternative_lib_cell -lib my_lib FF1
{"fdef1a5", "fdef1a30"}
```



Step #3

Can the violating path be isolated?

6-18

```
pt_shell> man get_alternative_lib_cells
```

```
...
```

DESCRIPTION

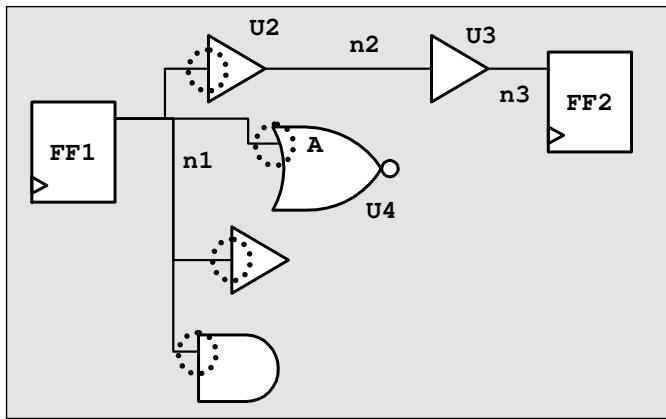
The `get_alternative_lib_cells` command creates a collection of library cells that can be used to replace or "size" a specified cell in the current design. A library cell is included in the collection only if it satisfies the same criteria used by the `size_cell` command to determine whether a library cell can be used to replace a cell in the current design.

When the `-libraries` option is omitted, the command traverses the link path to find libraries. In this case, if a library is found with a min library relationship (from `set_min_library`), the min library is automatically included. When the `-libraries` option is used, only the libraries passed in are considered.

To see a comparison of the slack and design cost within a set of alternative library cells, use the `report_alternative_lib_cells` command.

Generate Timing Reports for Each Fanout

```
# Report all net driver and load pins  
pt_shell> report_net -connections -verbose n1  
. . .  
# Repeat for each additional pin  
pt_shell> report_timing -through U4/A
```



Insert a buffer on paths with positive slack!

See SolvNet article in student notes!

6-19



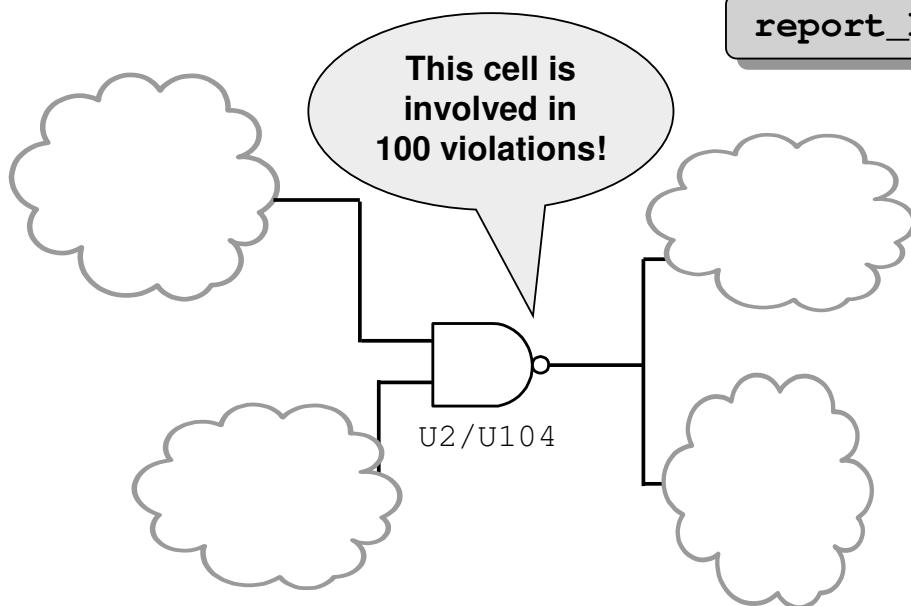
For additional details plus a useful Tcl procedure, refer to SolvNet article “What are some alternatives to resizing a heavily-loaded cell in PrimeTime”.

Doc Id: 012395 Last Modified: 04/03/2005

Identify Bottleneck Cells in the Design

Identify cells involved in multiple violations.

Use the results to determine cells to buffer or upsize.



6-20

```
pt_shell> report_bottleneck -cost_type fanout_endpoint_cost
*****
Report : bottleneck
-cost_type fanout_endpoint_cost
-max_cells 20
-nworst_paths 100
Design : AM2910
Version: 2001.08-SP1
Date   : Thu Oct 18 16:36:09 2001.08
*****
Bottleneck Cost = Number of violating paths through cell
Cell           Reference      Bottleneck Cost
-----
U2/U104          EO          100.00
U2/U70          AN2          100.00
U3/U109          OR2P         100.00
U3/U110          ND2I         100.00
U3/U130          ND2I         100.00
U3/U148          IV           100.00
U3/U150          IV           100.00
U3/U152          IV           100.00
```

Last Topic – Significant Digits

`report_timing`

`-significant_digits 3`

Increase the significant digits for this specific report (default 2)



Would you prefer to control the significant digits by default for many reports?

6-21

Set an Application Variable Instead!

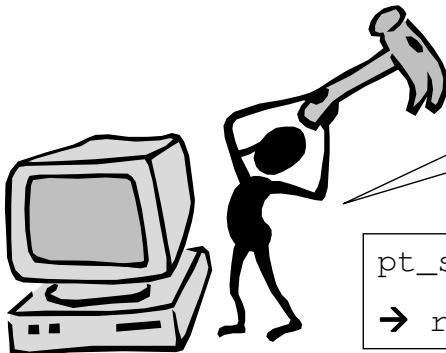
PrimeTime application variable

`report_default_significant_digits`

Specify the default digits for many reports to 4:



- `report_default_significant_digits = 4`
- `set report_default_significant_digits 4`



It is difficult to remember all these long variable names!

```
pt_shell> printvar *sign*
→ report_default_significant_digits
```

6-22

Group Exercise



Describe the difference in what the two commands below return?

```
pt_shell> printvar *report*
```

```
pt_shell> help *report*
```

6-23

Lab 6: Generate Summary Reports



15 minutes



10 Minute BREAK

15 Minute LAB

The objective is to:

- **Generate summary reports**

6-24

Lab 6 Wrap Up

List the top 5 violations for setup – endpoint and slack.

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

List the clock domains with violations.

```
report_constraint -all
```



List the worst slack to each bit and identify the largest margin.

```
report_timing -path end -max_paths 16 -to sd_DQ*
```

6-25

This page was intentionally left blank.

Agenda

**DAY
2**

6 Summary Reports



7 Create a Setup File and Run Script



8 Validate a Run Script



9 Getting to Know Your Clocks



Unit Objectives

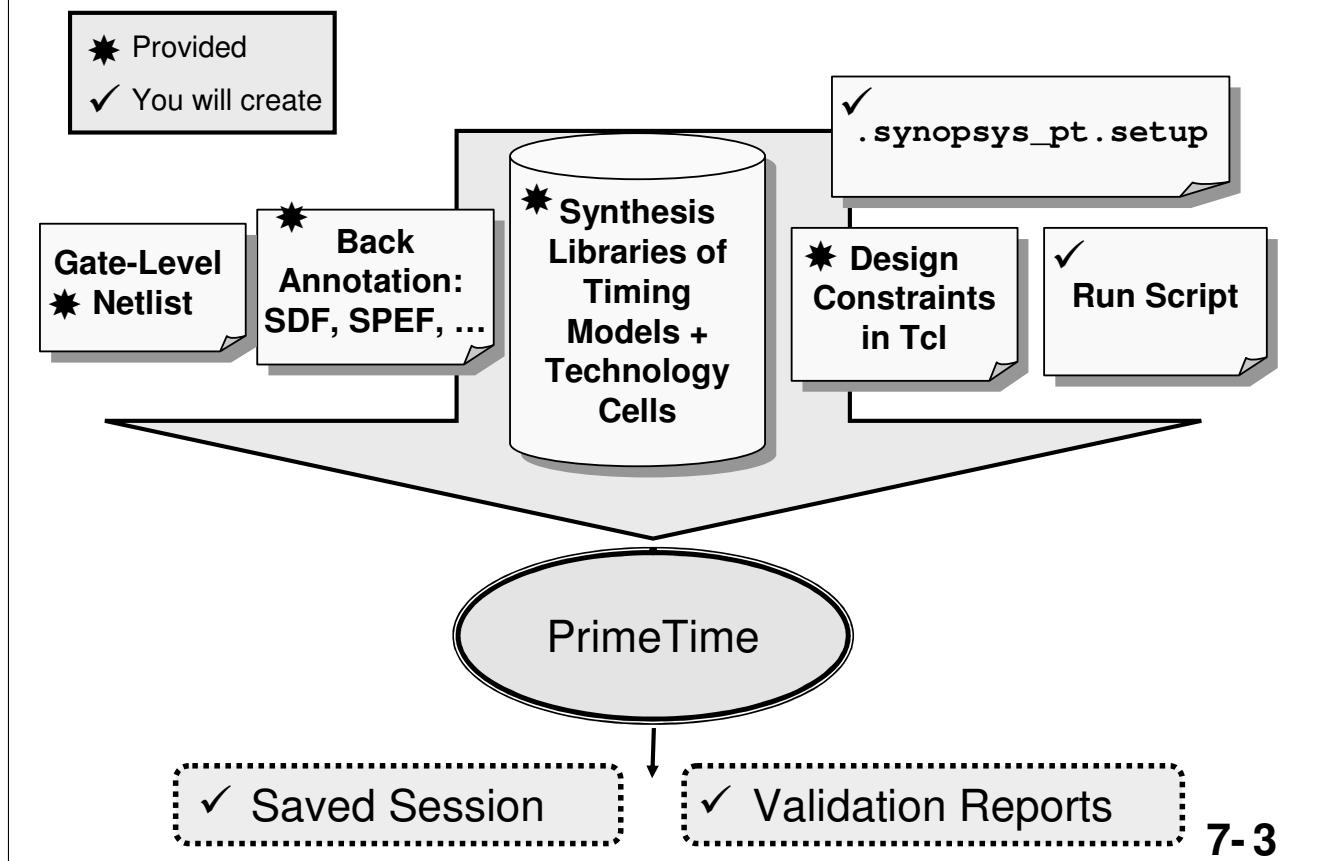


After completing this unit, you should be able to:

- **State 6 PrimeTime inputs, 2 outputs**
- **Create a setup file that includes**
 - Aliases
 - Suppressed messages
 - Useful Tcl procedures
 - ◆ Including one you download from SolvNET
- **Create a run script that**
 - Reads the 6 PrimeTime inputs

7-2

PrimeTime Inputs and Outputs



7-3

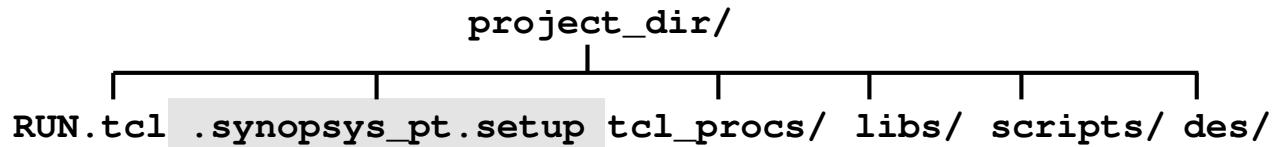
Synthesis technology libraries must be compiled to Synopsys db format.

PrimeTime does not use simulation or physical libraries.

Gate-level netlists can be in EDIF, Verilog, VHDL or Synopsys compiled db, ddc or Milkyway formats.

In this course, the back annotation we will focus on involves SDF files – on the last day, we will also talk about parasitic (SPEF) files

Create a .synopsys_pt.setup File



- You create this file
- The name of the file must be `.synopsys_pt.setup`
- Executed automatically every time PrimeTime is invoked from the project directory



How do you list all contents of a Unix directory, including the setup file:

- `ls`
- `ls -a`

7-4

Example Setup File

.synopsys_pt.setup

```
# ORCA setup file
##### Alias Commands
alias h {history}
alias page_on {set sh_enable_page_mode true}
alias page_off {set sh_enable_page_mode false}
alias _____ {report_timing}
alias _____ {report_timing -delay min}

##### Suppress Messages
# Informational message when restoring a session
suppress_message PTSR-004

##### Other
history keep 200
set sh_enable_line_editing true
```



7-5

By default, only 20 commands are kept in the history list → history keep 200

NO: “alias q quit” → Hitting “q” once too often when trying to quit from page mode may cause PrimeTime to exit!

OK: “**unalias q**”

So as to make it obvious what is suppressed, some project teams put all suppress_message commands in a single ‘suppress message’ file sourced by the run file, rather than in the setup file.

Suppressing Multiple Messages: **suppress_message PTSR-004**

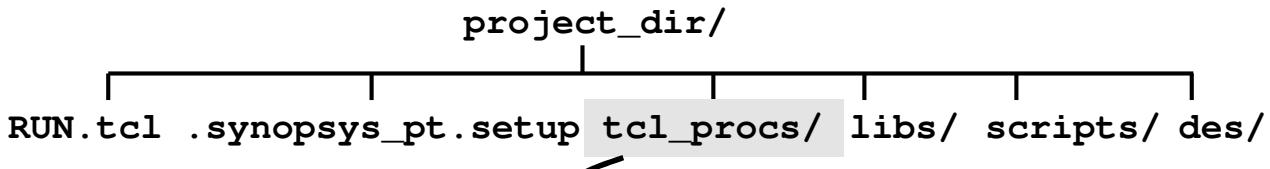
suppress_message ENV-003 (OR) suppress_message {PTSR-004 ENV-003}

```
pt_shell> man sh_enable_line_editing
NAME
    sh_enable_line_editing
        Enables the command line editing
        capabilities in PrimeTime.

DEFAULT
    false

DESCRIPTION
    If set to true it enables advanced Unix like shell capabilities. This
variable needs to be set in .synopsys_pt.setup file to take effect.
```

Source Multiple Tcl Procedures

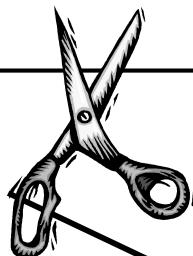


.synopsys_pt.setup

```
# ORCA setup file
#####
history keep 200
foreach each_proc [glob -nocomplain ./tcl_procs/*.tcl] \
    {source $each_proc}
```

no error generated if an empty list is returned

Copy and paste!



7-6



Refer to SolvNet article "How to automatically source Tcl procs".

Doc Id: 001688 **Last Modified:** 09/25/2002.

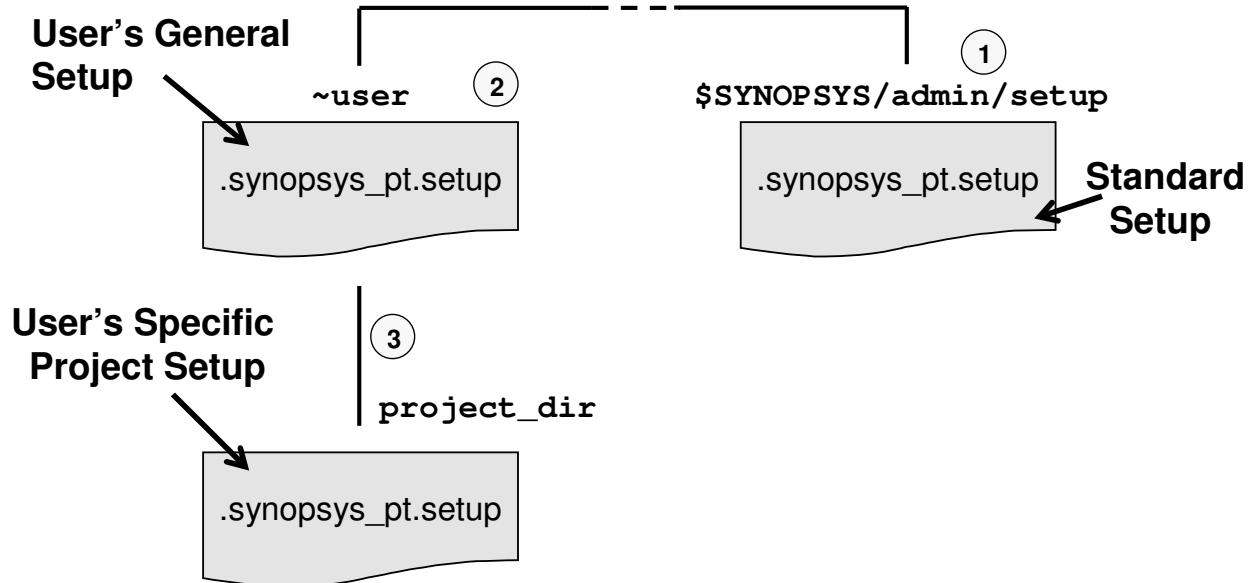
```
pt_shell> man glob
NAME
    glob - Return names of files that match patterns
SYNOPSIS
    glob ?switches? pattern ?pattern ...?
DESCRIPTION
    This command performs file name ``globbing'' in a fashion similar to the
    csh shell. It returns a list of the files whose names match any of the
    pattern arguments.
```

If the initial arguments to glob start with - then they are treated as switches. The following switches are currently supported:

-nocomplain

Allows an empty list to be returned without error;
without this switch an error is returned if the result
list would be empty.

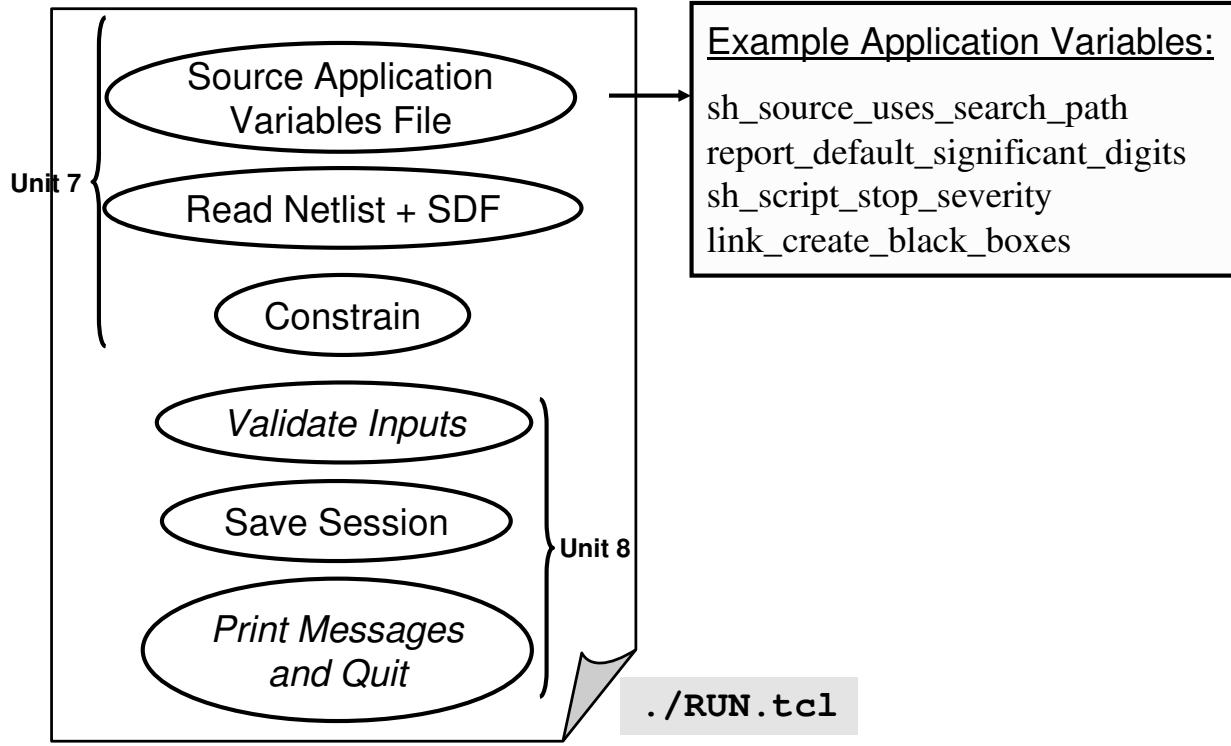
Setup Files in Other Locations



7-7

Components of a Master Run Script

`pt_shell -f RUN.tcl | tee -i run.log`



7-8

Example Application Variable settings:

```

set sh_source_uses_search_path true      ← when using "source"
set report_default_significant_digits 4
set sh_script_stop_severity E
set link_create_black_boxes false        ← don't create black box when a reference is not reso
  
```

Read Application Variables & Design Netlist

```
# Run script for ORCA  
  
set search_path { . ./scripts ./libs ./des}  
  
source ./scripts/variables.tcl  
  
# Read all gate-level design files  
read_verilog my_full_chip.v
```

./RUN.tcl

■ Read all netlist files:

- Order is unimportant
- PrimeTime supports many formats (e.g. VHDL, ddc, EDIF)



Where will PrimeTime search for
the netlist files for reading?

7-9

Some users prefer placing `search_path` variable in the `.synopsys_pt.setup` file

To read multiple netlist files, use the following two examples.

```
# With a single read command  
pt_shell> read_verilog "A.v B.v Top.v"  
# With several read commands  
pt_shell> read_verilog A.v  
pt_shell> read_verilog B.v  
pt_shell> read_verilog Top.v  
  
pt_shell> ss read_  
***** Commands *****  
read_db # Read a db file  
read_ddc # Read a ddc format design file  
read_edif # Read an EDIF file  
read_file # Read a netlist or library file  
read_lib # Read a synopsys .lib file  
read_parasitics # Backannotate RC parasitics  
read_sdc # Read a Synopsys Design Constraints format script  
read_sdf # Backannotate delays from SDF  
read_verilog # Read a Verilog file  
read_vhdl # Read a VHDL file  
***** Variables *****  
read_parasitics_load_locations = "false"
```

After Reading, Designs Must Be Linked

./RUN.tcl

```
# Run script for ORCA
set search_path { . ./scripts ./libs ./des }
source ./scripts/variables.tcl
# Read all gate-level design files
read_verilog my_full_chip.v
```

link_design ORCA

- Resolves all instantiations in a design
- Sets specified design (top level) for analysis:
 - The current_design
- Reads all libraries into PrimeTime



List 3 “categories” of what could be instantiated in your design netlist?

7-10

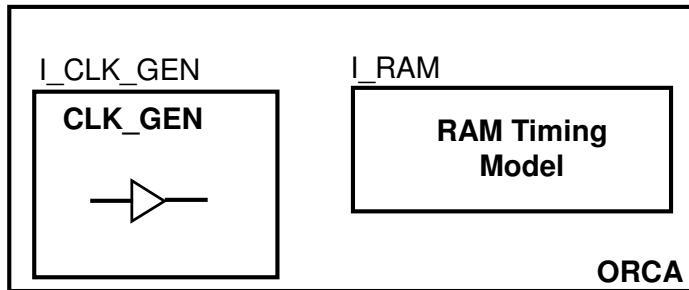
PrimeTime is optimized to work on only one linked design in memory at a time.

If you link a design, all other designs are first unlinked. Unlinking throws away all constraints and SDF for that particular netlist.

Another way, to link a design in PrimeTime is:

```
pt_shell> current_design ORCA
pt_shell> link_design
```

Identify All Libraries and Read All Designs



./RUN.tcl

Run script for ORCA
set search_path { . ./scripts ./libs ./des }

lappend link_path tech_lib.db RAM_lib.db

source ./scripts/variables.tcl
Read all gate-level design files
read_verilog my_full_chip.v

link_design ORCA

① Include all libraries

② Read all designs and subdesigns

③ Set the top-level design

7-11

pt_shell> **man link_path**
DEFAULT

*

DESCRIPTION

Specifies a list of libraries, design files, and library files used during linking. The link_design command looks at those files and tries to resolve references in the order of specified files.

The link_path variable can contain three different types of elements: "*", a library name, or a file name.

The "*" entry in the value of this variable indicates that link_design should search all the designs loaded in pt_shell while trying to resolve references. Designs are searched in the order in which they were read.

For elements other than "*", pt_shell searches for a library that has already been loaded. If that search fails, pt_shell searches for a filename using the search_path variable.

The default value of link_path is "*". To determine the current value of this variable, type printvar link_path or echo \$link_path.

pt_shell> **man link_library**

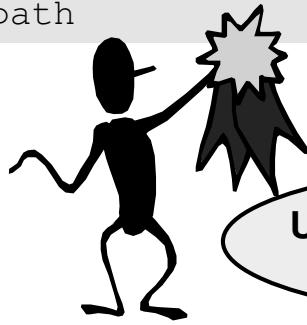
NAME

link_library This is a synonym for the link_path variable.

The Star in link_path

If missing, PrimeTime will not link to designs read in to PrimeTime memory to resolve subdesigns!

```
pt_shell> printvar link_path  
link_path = "*" ←  
  
pt_shell> lappend link_path tech_lib.db RAM_lib.db  
  
pt_shell> printvar link_path  
link_path = "* tech_lib.db RAM_lib.db"
```



Use lappend to avoid losing the default * in link_path.

7-12

If you use the **set** command to modify the **link_path** variable – you must remember to include the * yourself.

```
set link_path "* tech_lib.db RAM_lib.db"
```

The “main library” will be the first library in the **link_path** variable that has time units. This determines the time units used during the PrimeTime session.

```
pt_shell> man lappend  
NAME  
lappend - Append list elements onto a variable  
SYNOPSIS  
lappend varName ?value value value ...?  
DESCRIPTION  
This command treats the variable given by varName as a list and appends each of the value arguments to that list as a separate element, with spaces between elements. If varName does not exist, it is created as a list with elements given by the value arguments. Lappend is similar to append except that the values are appended as list elements rather than raw text. This command provides a relatively efficient way to build up large lists. For example, ``lappend a $b'' is much more efficient than ``set a [concat $a [list $b]]'' when $a is long.
```

Read SDF

```
# Run script for ORCA
set search_path {.. ./scripts ./libs ./des}
lappend link_path tech_lib.db RAM_lib.db
source ./scripts/variables.tcl

# Read all gate-level design files
read_verilog my_full_chip.v
link_design ORCA
```

./RUN.tcl

read_sdf -analysis_type on_chip_variation orca.sdf.gz

- Set the design into **on_chip_variation mode**:
 - Recommended for accuracy
 - Details later in the course: Analysis Type
- **Read gzip files directly**

7-13

More information regarding **on_chip_variation** mode in unit 10.

```
pt_shell> read_sdf -help
[-path path_name]      (Path prefix to add to SDF instances)
[-load_delay load_delay_type] (Delay include cell delay: Values: net, cell)
[-analysis_type type]  (Analysis_type:
                         Values: single, bc_wc, on_chip_variation)
[-min_file min_fname] (Read min delays from SDF)
[-max_file max_fname] (Read max delays from SDF)
[-type sdf_delay_type] (SDF delay type for delays (default: sdf_max):
                         Values: sdf_min, sdf_typ, sdf_max)
[-min_type min_delay_type]
                         (SDF delay type for min delays (default: sdf_min):
                          Values: sdf_min, sdf_typ, sdf_max)
[-max_type max_delay_type]
                         (SDF delay type for max delays (default: sdf_max):
                          Values: sdf_min, sdf_typ, sdf_max)
[-cond_use use_type]   (Take the min or max value among COND delays:
                         Values: min, max, min_max)
[-syntax_only]         (Parse SDF for syntax errors, no delay annotation)
[-strip_path path_name] (Path prefix to strip from SDF instances)
[-quiet]                (Do not report annotated delays and checks)
[-worst]                (Annotate only worst delays)
[file_name]             (Name of SDF file)
```

Test For Understanding 1/2



Circle 4 problems in this run script.

```
# Run script for ORCA  
  
set search_path {.. ./scripts ./libs ./des}  
  
set link_path {tech_lib.db RAM_lib.db}  
  
source ./scripts/variables.tcl  
  
# Read all gate-level design files  
  
read_verilog orca.v  
  
read_sdf orca.sdf.gz  
  
link_design
```



./RUN.tcl

7-14

Test For Understanding 2/2



Circle the command that will read libraries into PrimeTime.
Where will PrimeTime search for these library files?

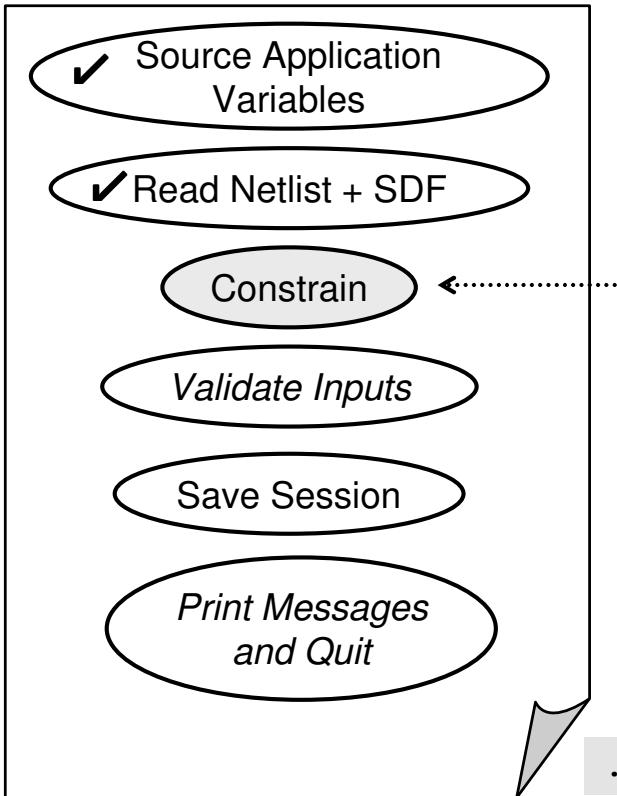
./RUN.tcl

```
# Run script for ORCA
set search_path { . ./scripts ./libs ./des }
lappend link_path tech_lib.db RAM_lib.db
source ./scripts/variables.tcl
# Read all gate-level design files
read_verilog orca.v
link_design ORCA
read_sdf -analysis_type on_chip_variation orca.sdf.gz
```



7-15

Components of a Master Run Script



What command will you use to execute the constraint file?

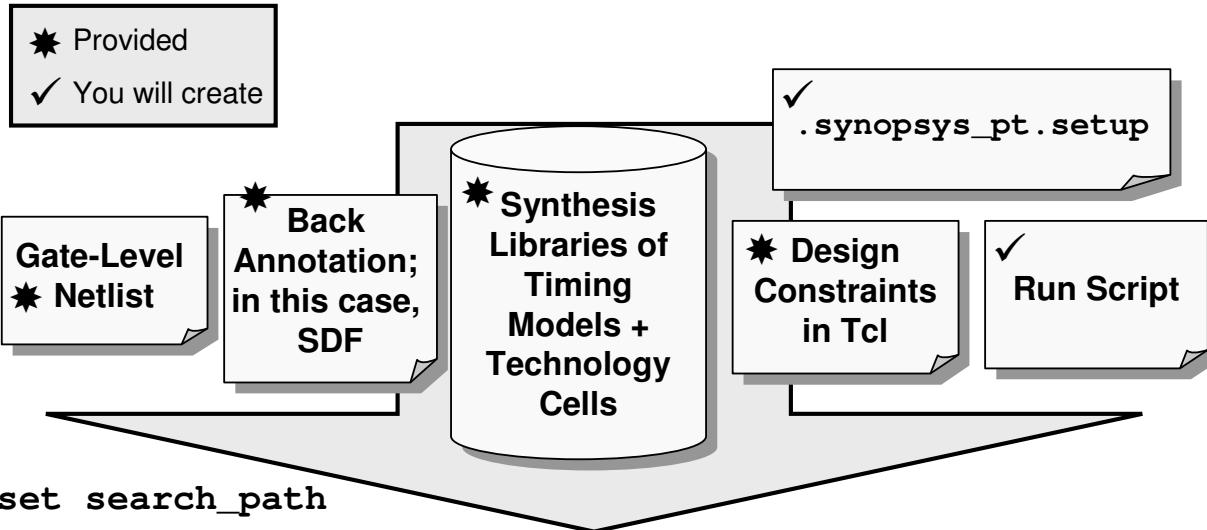
What options to this command are helpful for debugging?



`./RUN.tcl`

7-16

Summary: Reading 6 PrimeTime Inputs



```
set search_path  
lappend link_path  
read_verilog  
link_design  
read_sdf  
source
```



What commands are needed to read each input?

What input is NOT read in by any of these six commands?

7-17

Synthesis technology libraries must be compiled to Synopsys db format.

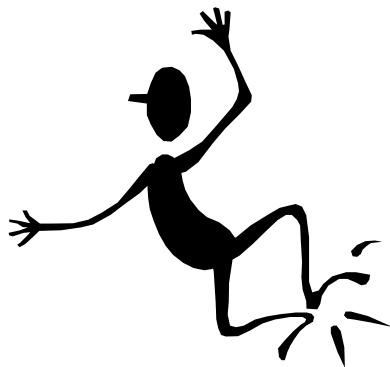
PrimeTime does not use simulation or physical libraries.

Gate-level netlists can be in EDIF, Verilog, VHDL or Synopsys compiled db, ddc or Milkyway formats.

What is Your SolvNet ID?



I know my
SolvNet ID!



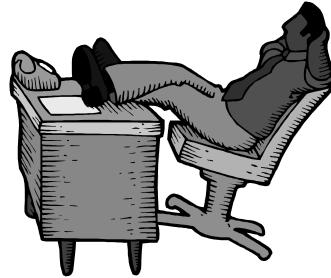
If you do not have a SolvNet ID, speak with
the instructor.

7-18

Lab 7: Create a Setup File and Run File



90 minutes



The objective is to:

- **Create and test a setup file that includes aliases and custom procedures**
- **Use SolvNet to download a Tcl procedure**
- **Create a Run script**

7-19

Lab 7 Wrap Up: Questions On SolvNet / SNUG ?

Search SolvNet and download the aaTcl procedure.

Search SNUG and download a paper on PrimeTime updates.

SYNOPSYS® SolvNet

Release Notes | Script Library | Scout Newsletter | Software and Installation | Support Resources | CheckPoint Weekly Poll | Bulletin | Industry News | Bookmarks

Expiration length of a License_key for a TSL has changed | New SupportResources Channel Offers Easy Access to Support | New Articles | Support Resources | Tool Support and Product Issues | Software and Installation | Documentation and Media | Issues with SolvNet Website | Go to URL: []

What detailed chip router do you use? (radio buttons: Cadence, Avant!, Other) | More: Read more New Articles... | Getting the Most out of the New Verilog 2000 Standard | Updated Jul 18, 2001 9:21 AM | Do We Need Another Spice? Yes, And Here's Why | Electronic Design - A new Spice with integrated logic primitives that performs the switch-level simulation is a better approach | Alcatel to Use 0.13-micron process technology for next-generation ICs | AFX News Limited - Alcatel to be one of the first semiconductor companies in the world to adopt the 0.13-micron process for chip production. | Intel Eases Gloom by Beating Estimates | Tech Web - Analysts applauded the strong performance in an unforgiving market. | Applied Materials unveils equipment for smaller, faster chips | Associated Press - The new technology will squeeze more circuits onto

SYNOPSYS® SolvNet

Back Forward Reload Home Search Netscape Print Security Stop Bookmarks Netscape http://solvnet.synopsys.com/DesktopServer?action=contentProvider=wFrontProvider

synonet marsha synopsys symon excite cku Accue Insight

MY PROFILE EDIT HELP LOGOUT

Search Solv-It

Search for: Search File

Try Advanced Search | Browse by Categories | Search Documentation | Search Synopsys IP

New Articles

- How to Read in a Parameterized Design for Synthesis
- Slew Propagation From Pin Other than on Critical Path
- Applying set auto ideal nets
- Config DAC in Customer Survey
- Howto Add a Prefix to a Block and All its Subblocks Throughout the Hierarchy

Support Resources

- Solv-It Advanced Search
- Enter A Call to the Support Center
- Loud Modeling Support
- VC3 Support | VERA Support

Software and Installation

- Smartkeys | Electronic Software Transfer

Documentation and Media

- Documentation on the Web | MediaDocs

Issues with SolvNet Website

- SolvNet Help | SolvNet Feedback

General Support Information

- More About Support

Bulletin

need support?

CheckPoint Archive

CheckPoint Weekly Poll

What detailed chip router do you use?

- a. Cadence
- b. Avant!
- c. Other

Submit

CheckPoint Archive

Industry News

Updated Jul 18, 2001 9:21 AM

- Do We Need Another Spice? Yes, And Here's Why
- Electronic Design - A new Spice with integrated logic primitives that performs the switch-level simulation is a better approach
- Alcatel to Use 0.13-micron process technology for next-generation ICs
- AFX News Limited - Alcatel to be one of the first semiconductor companies in the world to adopt the 0.13-micron process for chip production.
- Intel Eases Gloom by Beating Estimates
- Tech Web - Analysts applauded the strong performance in an unforgiving market.
- Applied Materials unveils equipment for smaller, faster chips
- Associated Press - The new technology will squeeze more circuits onto

Bookmarks

Go to URL: []

Synopsys

Designsphere Online Design | HelloBrain Intellectual Capital Exchange | Registrar Console

7-20

Lab 7 Wrap Up: Browsing SolvNet

SYNOPSYS® SolvNet

MY PROFILE | PREFERENCES

Browse SolvNet by Product

[Browse Home](#) | [Search Within this Category](#) | [Custom Search](#) | [Help](#)

PrimeTime (580)

- [Application Note \(17\)](#)
- [Q&A \(511\)](#)
- [STAR \(40\)](#)
- [Script \(12\)](#)

Browse
Sort by date
Learn!



7-21

Lab 7 Wrap-Up: PTE-029 Warning



Did you see
this warning?

./ORCA_EW.log

Id	Limit	Occurrences	Suppressed
<hr/>			
CMD-029	0	1	1

The above messages could be:



- Errors, warnings or informationals.
- Warnings or informationals only – scripts terminate on errors.

What is the next step to debug
this message?

7-22

Lab 7 Wrap Up

Identify a message, research, and then suppress it.

```
suppress_message CMD-029  
unalias q  
unsuppress_message CMD-029
```

Id	Limit	I	Occurrences	Suppressed
CMD-005	0	1	1	0
CMD-029	0	1	1	1
Diagnostics summary: 1 error				

Use to debug the setup file.

```
source -echo -verbose .synopsys_pt.setup
```

7-23

```
pt_shell> print_message_info -help  
Usage:  
  print_message_info  # Print information about messages  
    [-ids id_list]      (List of message ids to report)  
    [-summary]          (Summarize diagnostics)
```

The following will print all messages starting with the letters CMD*.

```
pt_shell> print_message_info -ids CMD*
```

This page was intentionally left blank.

Agenda

**DAY
2**

6 Summary Reports



7 Create a Setup File and Run Script



8 Validate a Run Script



9 Getting to Know Your Clocks



Unit Objectives



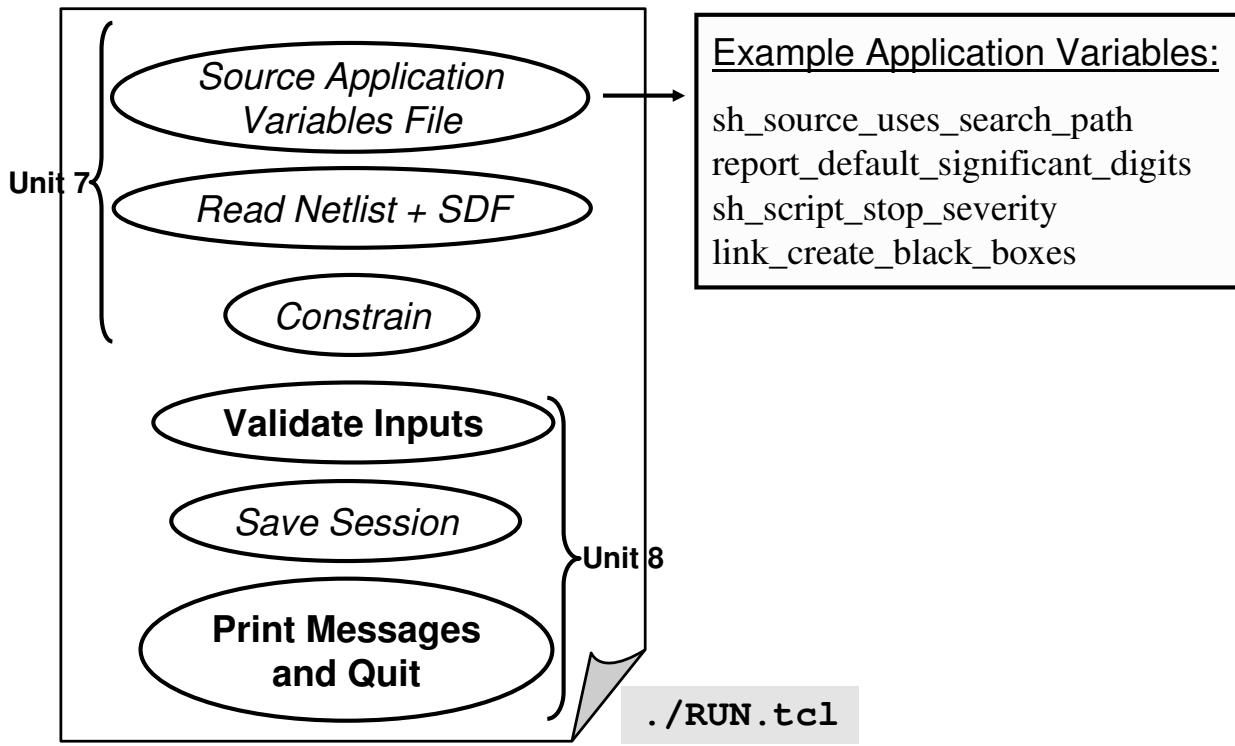
After completing this unit, you should be able to:

- **Create a run script that**
 - Validates design data
 - Monitors runtime, memory and timing updates
- **Save a session and share it with others**

8-2

Components of a Master Run Script

`pt_shell -f RUN.tcl | tee -i run.log`



8-3

Example Application Variable settings:

```
set sh_source_uses_search_path true
set report_default_significant_digits 4
set sh_script_stop_severity E
set link_create_black_boxes false
```

Checking the Run Script

■ Before executing

```
you need to install it !!!  
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]  
      ^
```

■ After executing

```
pt_shell> redirect -tee -append ./EW.log {print_message_info}  
Id          Limit  I  Occurrences  Suppressed  
-----  
CMD-005          0        1          0  
CMD-029          0        1          1  
Diagnostics summary: 1 error
```

8- 4

For information on how to install ptprocheck: Solvnet article “How to Check or Debug Tcl Scripts for Design Compiler or PrimeTime”; Doc ID 901288

Identify Warnings in Sourced Files

```
# Echo each command and command result  
source -echo -verbose ./scripts/constraints.tcl
```

OR

```
# Identify line numbers for warnings  
set sh_source_emits_line_numbers W  
set sh_continue_on_error true ← stop "execution" when a error is occured!!!  
source ./scripts/constraints.tcl
```

Information: Errors and/or warnings occurred at or before
line 5 in script './scripts/constraints.tcl'. (CMD-082)

8-5

Unless you set sh_continue_on_error to true,
the behavior is to stop execution when an error is
encountered.

```
pt_shell> man sh_source_emits_line_numbers
```

...

DESCRIPTION

When a script is executed with the source command, error and warning messages can be emitted from any command within the script. Using the sh_source_emits_line_numbers variable, you can help isolate where errors and warnings are occurring.

This variable can be set to "none", "W", or "E". When set to "E", the generation of one or more error messages by a command will cause a CMD-082 informational message to be issued when the command completes, giving the name of the script and the line number of the command. Similarly, when sh_source_emits_line_numbers is set to "W", the generation of one or more warning or error messages will cause a the CMD-082 message.

Linking May be Incomplete!

./run.log



```
link_design ORCA
```

Linking design ORCA...

Warning: Unable to resolve reference to 'CLKMUL' in 'ORCA'.
(LNK-005)

Warning: Unable to resolve reference to 'PLL' in 'ORCA'.
(LNK-005)

Creating black box for I_CLOCK_GEN/I_CLKMUL/CLKMUL...

Creating black box for I_CLOCK_GEN/I_PLL_PCI/PLL...

Creating black box for I_CLOCK_GEN/I_PLL_SD/PLL...

. . .

Design 'ORCA' was successfully linked.

1



o-6

Preventing Black Boxes

- By default PT creates black boxes for missing references
 - A Black box is a cell having no timing arcs
- If there are missing references, it's preferable to resolve or use a timing model instead
- To prevent PrimeTime from creating black boxes:

```
# Default is true  
set link_create_black_boxes false
```



Where will you place this variable?

8-7

```
pt_shell> man link_create_black_boxes
```

DESCRIPTION

When true (the default), the linker automatically converts each unresolved reference into a black box, which is essentially an empty cell with no timing arcs. The result is a completely linked design on which analysis can be performed.

When false, unresolved references remain unresolved and most analysis commands cannot function.

At the end of an unsuccessful link – you will see the following message indicating that the link failed (rather than completed successfully with black boxes).

```
Information: Design 'CLOCK_GEN' was not successfully linked:  
29 unresolved references. (LNK-003)
```

0

Is the SDF Complete?

```
# Beginning of ./RUN.tcl  
file delete ./EW.log  
  
# Read all SDF files quietly  
read_sdf -quiet -analysis_type on_chip_variation orca.sdf.gz  
# Report missing SDF delays or timing checks  
redirect -append ./EW.log \  
    {report_annotated_delay; report_annotated_check}  
  
# End of ./RUN.tcl
```

./RUN.tcl

8-8

```
pt_shell> report_annotated_delay -help  
report_annotated_delay # Report backannotated delays  
    [-cell]           (Report on cell delays)  
    [-net]            (Report on internal net delays)  
    [-from_in_ports] (Report on nets starting at input ports)  
    [-to_out_ports]  (Report on net delays ending at output ports)  
    [-max_line num]  (Max number of lines for the -list_* options)  
    [-list_annotated] (List timing arcs which are backannotated)  
    [-list_not_annotated] (List timing arcs which are not backannotated)  
    [-constant_arcs]  (Keep a separate count for constant arcs)  
pt_shell> report_annotated_check -help  
report_annotated_check # Report backannotated timing checks  
    [-setup]          (Report setup timing checks)  
    [-hold]           (Report hold timing checks)  
    [-recovery]        (Report recovery timing checks)  
    [-removal]         (Report removal timing checks)  
    [-nochange]        (Report nochange timing checks)  
    [-width]           (Report width timing checks)  
    [-period]          (Report period timing checks)  
    [-max_skew]        (Report max skew timing checks)  
    [-clock_separation] (Report clock separation timing checks)  
    [-max_line num]   (Max number of line for the -list_* options)  
    [-list_annotated]  (List the check arcs which are backannotated)  
    [-list_not_annotated] (List the check arcs which are not backannotated)  
    [-constant_arcs]  (Keep a separate count for constant arcs)
```

What Is Reported?

This design has

 Complete SDF.

Missing SDF annotations.

Delay type	Total	Annotated	NOT Annotated
cell arcs	154424	154424	0
cell arcs (unconnected)	68	68	0
internal net arcs	52781	52781	0
net arcs from primary inputs	54	54	0
net arcs to primary outputs	61	61	0
	207388	207388	0

Timing check type	Total	Annotated	NOT Annotated
cell setup arcs	9629	9629	0
cell hold arcs	9629	9629	0
cell recovery arcs	1316	1316	0
cell removal arcs	1316	1316	0
cell min pulse width arcs	7273	7273	0
cell min period arcs	20	20	0
	29183	29183	0

8-9

pt_shell> **redirect -help**

Usage:

```
redirect          # Redirect output of a command to a file
      [-append]    (Append output to the file)
      [-tee]        (Tee output to the current output stream)
      [-file]       (Output to a file (default))
      [-variable]   (Output to a variable)
target           (Name of file/variable target for redirect)
command_string  (Command to redirect. Should be in braces {}.)
```

Validate Constraints are Complete

```
./RUN.tcl
```

```
# Run script for ORCA  
# After reading the design and SDF  
source -echo -verbose ORCA_constraints.tcl
```

```
redirect -tee -append ./EW.log {check_timing}
```

- Executes 10 default checks on design constraints
- To display the checks
 - printvar timing_check_defaults
- To display the ports/pins having potential problems
 - check_timing -verbose

8-10

The list of default checks performed by `check_timing` is

```
generated_clocks  
generic  
latch_fanout  
loops  
no_clock  
no_input_delay  
partial_input_delay  
unconstrained_endpoints  
unexpandable_clocks  
no_driving_cell
```

To Explore Port Details

```
pt_shell> report_port -input_delay pm66en
          Input Delay
          Min           Max       Related Related
Input Port   Rise   Fall    Rise   Fall   Clock   Pin
-----
pm66en      2.00   2.00   8.00   8.00  PCI_CLK   --
```

```
pt_shell> report_port -help
Usage:
  report_port          # Report port info
  [-verbose]           (Show all port info)
  [-design_rule]       (Only port design rule info)
  [-drive]             (Only port drive info)
  [-input_delay]        (Only port input delay info)
  [-output_delay]      (Only port output delay info)
  ...
```

8-11

Debug and Fix All Warnings



**It is just a warning,
ignore it!**

```
pt_shell> check_timing
Information: Checking 'no_clock'.
Information: Checking 'no_input_delay'.
Information: Checking 'partial_input_delay'.
Information: Checking 'no_driving_cell'.
Information: Checking 'unconstrained_endpoints'.
Warning: There are 63 endpoints which are not constrained
for maximum delay.
Information: Checking 'unexpandable_clocks'.
. . .
```

check_timing -verbose

8-12

The default checks verified by **check_timing** include:

generated_clocks; generic; latch_fanout; loops; no_clock; no_input_delay; partial_input_delay; unconstrained_endpoints; unexpandable_clocks; no_driving_cell.

To this you can add:

clock_crossing; data_check_multiple_clock; data_check_no_clock; latency_override; ms_separation; multiple_clock; retain; signal_level.

```
pt_shell> man check_timing
[-verbose]
[-significant_digits]
[-override_defaults check_list]
    Overrides the checks in timing_check_defaults using check_list.
[-include check_list]
    Adds the checks listed in check_list to the checks in
    timing_check_defaults.
[-exclude check_list]
    Subtracts the checks listed in check_list from the checks in
    timing_check_defaults.
```

Test For Understanding



**Circle 2 problems
in this run script.**

```
# Run script for ORCA

# After reading the design netlist and SDF
redirect -tee -append ./EW.log {print_message_info}
redirect -tee -append ./EW.log {check_timing}
source -echo -verbose ORCA_constraints.tcl

# Generate initial reports
redirect -tee -append ./RUN.sta {report_analysis_coverage}
save_session orca_savesession
quit
```

./RUN.tcl

8-13

Examine the combined state of the inputs

- **Timing verification depends on**
 - Design
 - Constraints
 - ◆ Constraint file
 - ◆ Back annotation file (SDF or SPEF)
 - Settings
 - ◆ Application variables
- **Check the combined state of the inputs with a timing verification summary**
 - `report_analysis_coverage`
 - ◆ Summarizes how many of each type of violation exist in your design

8- 14

Save the PrimeTime Session

```
./RUN.tcl
```

```
# Run script for ORCA
. . .
# Save the session
save_session -replace orca_savesession

quit
```

Unix directory name



- Saves all required data to the named Unix directory:

- If the directory exists, it will be deleted and overwritten with the new data for the current session
- If switch is omitted and the Unix directory exists, this command will generate an error and terminate!
- A saved session can be restored only using the same version of PT

8-15

Use -replace for repeated, successful execution of your run script.

Use “restore_session” command to restore this session later

```
pt_shell> man save_session
. . .
ARGUMENTS
dir_name
    Specifies the name of a directory to save the session in. If the named
    directory does not exist, save_session will try to create it. If it
    already exists, save_session will issue error message and stop, unless the
    -replace option is given.

-replace
    If specified, it indicates that if any file or directory with the same
    name as the specified already exists in the file system, it will be
    deleted and then overwritten with the saved data for the current session.
    Use -replace with great caution, because it may remove all the existing
    data, including all files and subdirectories, already present in the
    target directory.
```

DESCRIPTION

This command creates a repository of data to save the current PT session to. The name of the directory to save the session is a required argument. If the directory does not exist, a new one will be created and the data for the session will be written into the directory. If the directory exists, it is an error unless the -replace option is given. The -replace option will cause an existing file or directory to be removed and replaced by the data of the current session to be saved.

Sharing Saved Sessions



You executed an initial PrimeTime run, validated it, and saved the session.



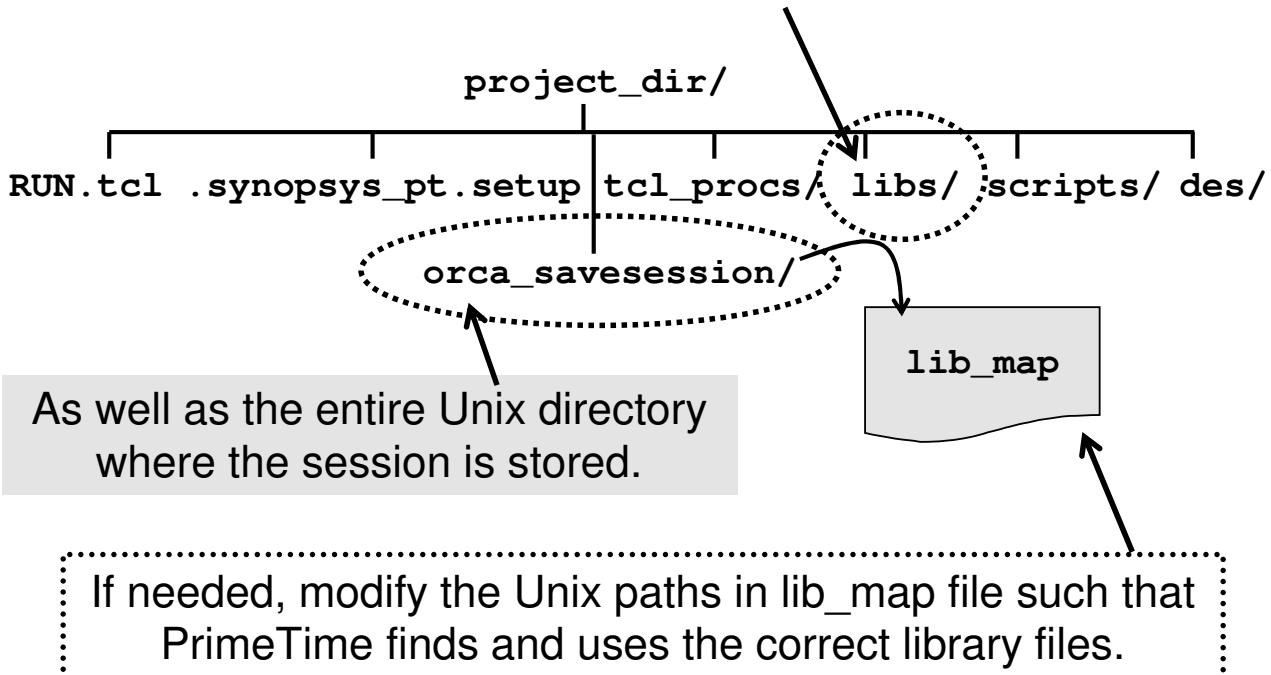
Joe has some time to help debug the design violations.

How will you share the saved session with your team members?

8-16

Required Files and Directories

Joe requires all the library files.



8-17

Improving Performance

■ You have

- Validated reading your PrimeTime inputs
- Saved your session for a future restore
 - ◆ By you
 - ◆ Or by others

■ What else can be done to the run script?

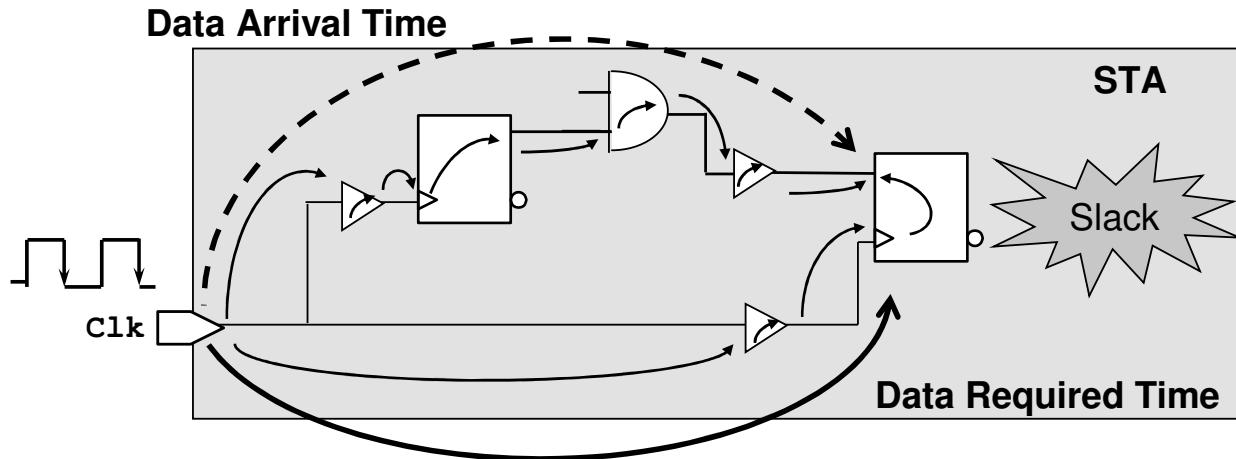
- Enhance performance

8-18

Timing Updates

PrimeTime must perform a timing update to perform STA:

- A timing update refers to delay and slack calculations
- A timing update occurs automatically when needed
- Timing update is not needed following a restore_session

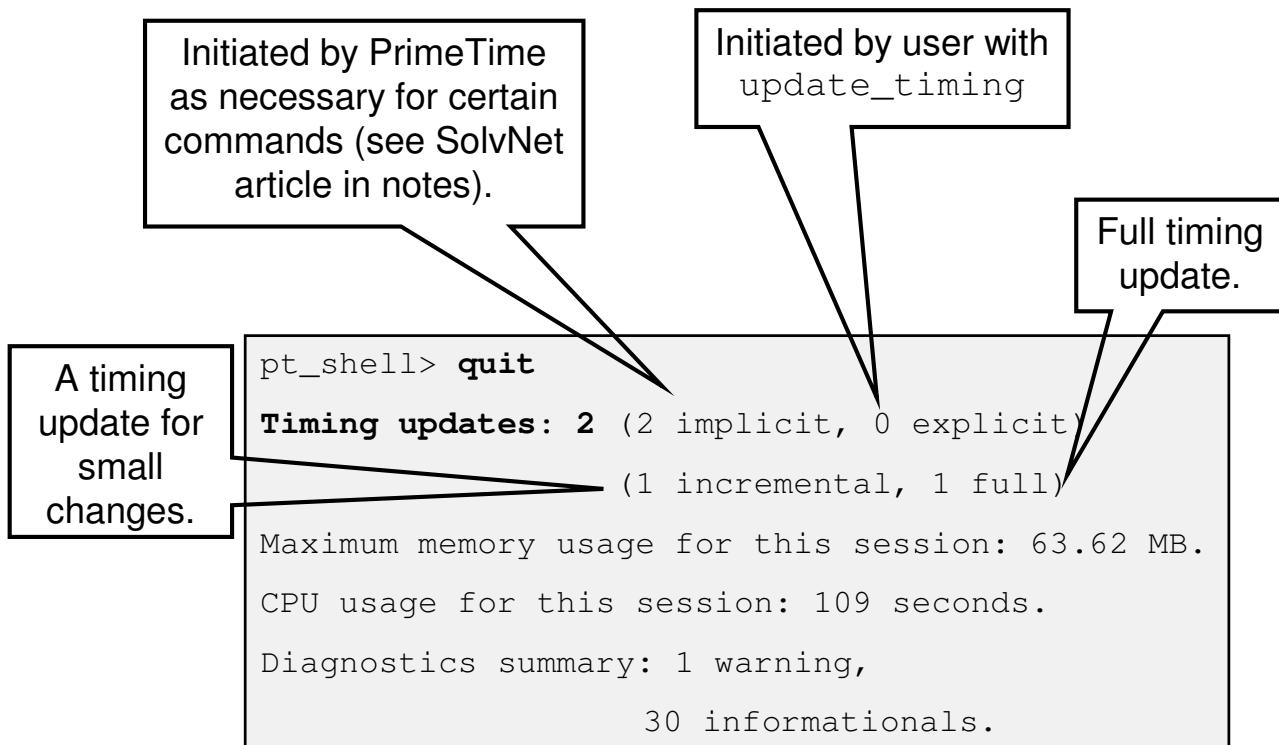


8-19



If you are in a situation where you are “fixing” or changing constraints as you are performing timing analysis, PrimeTime will automatically perform an incremental timing update if appropriate. Use the latest versions of PrimeTime because this capability has been improving. Fast timing updates are abandoned with warning code **PTE-018**.

How Many Timing Updates Occurred?



An incremental timing update is a lot faster than a full update !!

8-20



For more information, refer to SolvNet article “**What commands trigger an implicit timing update in PrimeTime**”

Doc Id: 011239 **Last Modified:** 03/04/2005

pt_shell> **man update_timing**

ARGUMENTS

-full

Indicates that the entire timing analysis is to be performed from the beginning. The default is to perform an incremental analysis, which updates only out-of-date information and runs more quickly.

DESCRIPTION

Updates timing for the current design. Timing is also automatically updated by commands that need the information, such as report_timing. This command explicitly prepares the design for further analysis.

Identify Where Timing Updates Occur

```
pt_shell> set timing_update_status_level high ←  
pt_shell> read_sdf -analysis_type bc_wc ORCA.sdf.gz  
pt_shell> source ORCA_constraints.tcl  
pt_shell> check_timing  
Information: Updating design - Started (UIITE-214)  
 . . .  
Information: Updating design - Completed (UIITE-214)
```

8-21

See Solvnet article 011239 “What commands trigger an implicit timing update in PrimeTime?”

Create Metrics for Run Script

```
# Track cpu time and peak memory usage at every major step

redirect ./metrics.log {stopwatch "START"}

read_verilog ORCA.v

link_design ORCA

redirect -append ./metrics.log {stopwatch "READ + LINK"}

read_sdf -analysis_type on_chip_variation ORCA.sdf.gz

redirect -append ./metrics.log {stopwatch "SDF"}

. . .
```

./RUN.tcl

Get the Tcl procedure **stopwatch** by:



- Contacting the support center.
- Searching on SolvNet.

8-22



The Tcl procedure stopwatch is available in the SolvNet article “**stopwatch.tcl - Measure Memory/Runtime Easily**”

Doc Id: 903482 **Last Modified:** 06/30/2005

If you would like to look at the body of a Tcl procedure while in PrimeTime, execute the following.

```
pt_shell> info body stopwatch
```

For more information and practice with writing Tcl procedures, refer to two additional workshops:

The Power of Tcl 1: Become a Proficient Tcl User

The Power of Tcl 2: Creating High Impact Procedures

The Power of Tcl 3: Collections and Attributes



What Does ‘stopwatch’ Output Look Like?



Circle two steps consuming the most memory and runtime.

`./metrics.log`

```
Information: Time used since beginning of process is 13. (START)
Information: Memory used since beginning of process is 6312k. (START)
Information: Time used since last stopwatch is 13. (READ + LINK)
Information: Memory used since last stopwatch is 38912k. (READ + LINK)
Information: Time used since last stopwatch is 66. (SDF)
Information: Memory used since last stopwatch is 0k. (SDF)
Information: Time used since last stopwatch is 114. (CONSTRAINTS)
Information: Memory used since last stopwatch is 2360k. (CONSTRAINTS)
Information: Time used since last stopwatch is 0. (check_timing)
Information: Memory used since last stopwatch is 0k. (check_timing)
Information: Time used since last stopwatch is 6. (Initial Reports)
Information: Memory used since last stopwatch is 0k. (Initial Reports)
Information: Time used since last stopwatch is 13. (Save Session)
Information: Memory used since last stopwatch is 5928k. (Save Session)
```

8-23

Search for the **last non-zero** “Memory used since last stopwatch” line, this will be the command or step which consumed the most memory during the run.

Three Useful Commands and One Variable

cputime

Total CPU time in seconds.

mem

Peak memory usage in kbytes.

quit

Total # of timing updates.

timing_update_status_level

UITE-214 message for each timing update.



8-24

```
pt_shell> man timing_update_status_level
```

DESCRIPTION

Controls the number of progress messages displayed during the timing update process. Allowed values are none (the default), low, medium, or high.

When set to none, no messages are displayed. When set to low, medium, or high, the progress of the timing update is reported for an explicit update (using the update_timing command) or for an implicit update invoked by another command (for example, report_timing) that forces a timing update. The number of messages varies based on the value of the variable . . .

Test For Understanding



Describe the contents of:



metrics.log

run.log

ORCA_EW.log

PrimeTime



./metrics.log

./run.log

./ORCA_EW.log

Saved Session

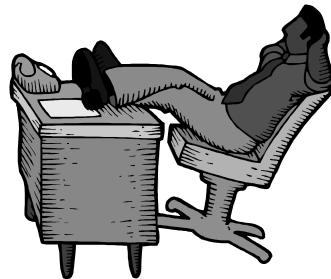
Initial Reports

8-25

Lab 8: Validate your design; Enhance a run script



60 minutes



The objective is to:

- **Validate your design data**
- **Generate runtime metrics for ORCA**
- **Enhance your run script**

8- 26

Lab 8 Wrap Up

Use to debug the setup or run file.

```
source -echo -verbose .synopsys_pt.setup  
source -echo -verbose RUN.tcl
```

List all messages that have occurred in this PrimeTime session.

Including suppressed messages!

```
print_message_info
```

8-27

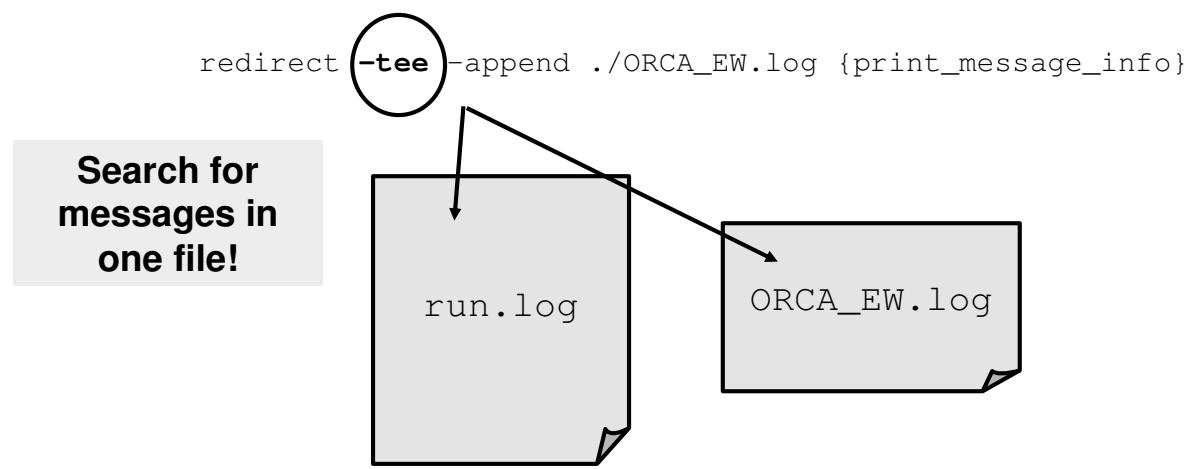
```
pt_shell> print_message_info -help  
Usage:  
  print_message_info  # Print information about messages  
    [-ids id_list]      (List of message ids to report)  
    [-summary]          (Summarize diagnostics)
```

The following will print all messages starting with the letters CMD*.

```
pt_shell> print_message_info -ids CMD*
```

Lab 8 Wrap Up: Redirecting Reports

- The redirect command **redirects all output**:
 - The command output (e.g. reports)
 - All messages (i.e. errors, warnings or informationals)
- This information will not be included in `run.log`



8-28

Lab 8 Wrap Up: Searching the Log Files



Where are the timing update occurring?

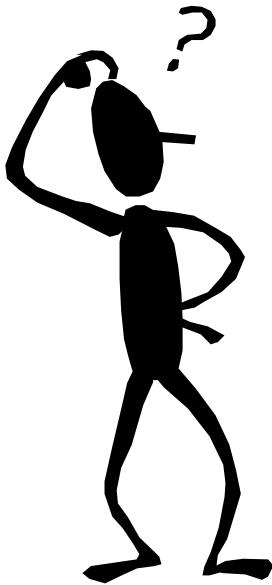
```
source orca_pt_constraints.tcl
Information: Updating design - Started (UITE-214)
Information: Updating design - Propagating Constants (UITE-214)
.
.
.
Information: Updating design - Completed (UITE-214)
Information: Errors and/or warnings occurred at or before line 36 in
    script 'PT_1_2006.06/lab8_*/scripts/orca_pt_other.tcl'. (CMD-082)
Information: Updating design - Started (UITE-214)
Information: Updating design - Calculating delays (UITE-214)
Information: Updating design - Calculating delays 10%... (UITE-214)
Information: Updating design - Calculating delays 100%... (UITE-214)
.
.
.
Information: Updating design - Calculating slacks (max type) for groups
(UITE-214)
Information: Updating design - Calculating slacks (min type) for groups
(UITE-214)
Information: Updating design - Completed (UITE-214)
Information: Updating design - Started (UITE-214)
```

8-29

Lab 8 Wrap Up: Missing Information

```
./run.log
```

```
Timing updates: 3 (2 implicit, 1 explicit) (1 incremental, 2 full)
```



I can only find two in `run.log` –
where did the last one go?

Keep all messages in `./run.log`



Markup Tools

redirect -tee

Don't use redirect.

8-30

Appendix

Guidelines to improve runtime and memory

- Run PT on the best machine (unix or linux)
- Use the latest PT version
- Reduce the # of Timing updates
- Use 64 bit mode only if necessary
- Avoid using exec or sh commands

8-32

Agenda

**DAY
2**

6 Summary Reports



7 Create a Setup File and Run Script



8 Validate a Run Script



9 Getting to Know Your Clocks

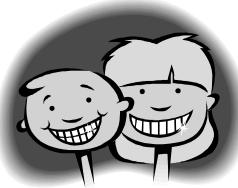


Review of Run Scripts

- 1) You know that no errors occurred during execution of the run script because:**
 - A. You searched through `run.log` for the word Error.
 - B. The script executed completely (i.e. did not terminate prematurely).
- 2) `print_message_info`:**
 - A. Used to generate man pages for specific messages.
 - B. Lists all messages (errors, warnings or informational) that have occurred.
- 3) `quit`:**
 - A. Place early in the run script to drastically improve runtime.
 - B. Reports the number and type of timing updates that have occurred.

9-2

Share A Work Application



**Please share an application at work
where you can apply one of the
scripting skills learned.**

9-3

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

9-4

Clocks are STA



STA is dictated largely by the design clocks.

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

9-5

Three Types of Clocks

3 Types of Clocks

**Asynchronous, synchronous,
and exclusive clocks**

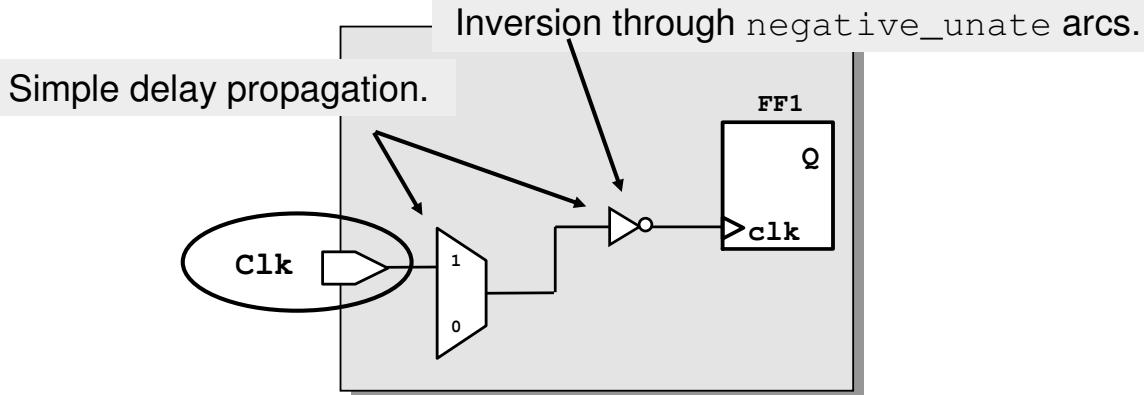
9-6

What Are Master Clocks?

- Master clocks are typically created at input ports

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

- Clocks are propagated through combinational logic



Describe what happens if Clk is:



Ideal
Propagated

9-7

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

Master 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?**”

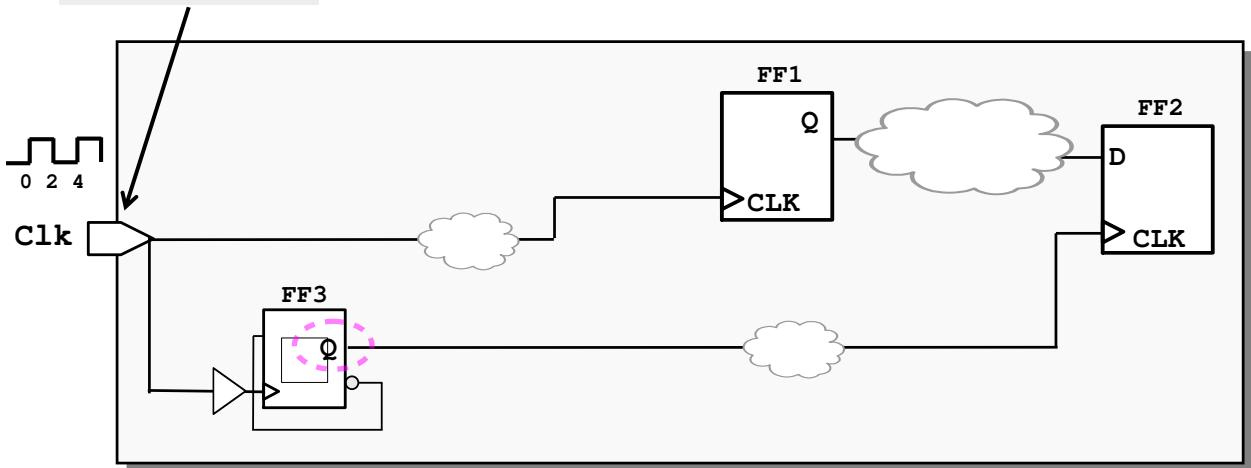
Doc ID: 011059 Last Modified: 03/24/2004

Generated Clocks: Internally Derived Clocks



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

Master Clock



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

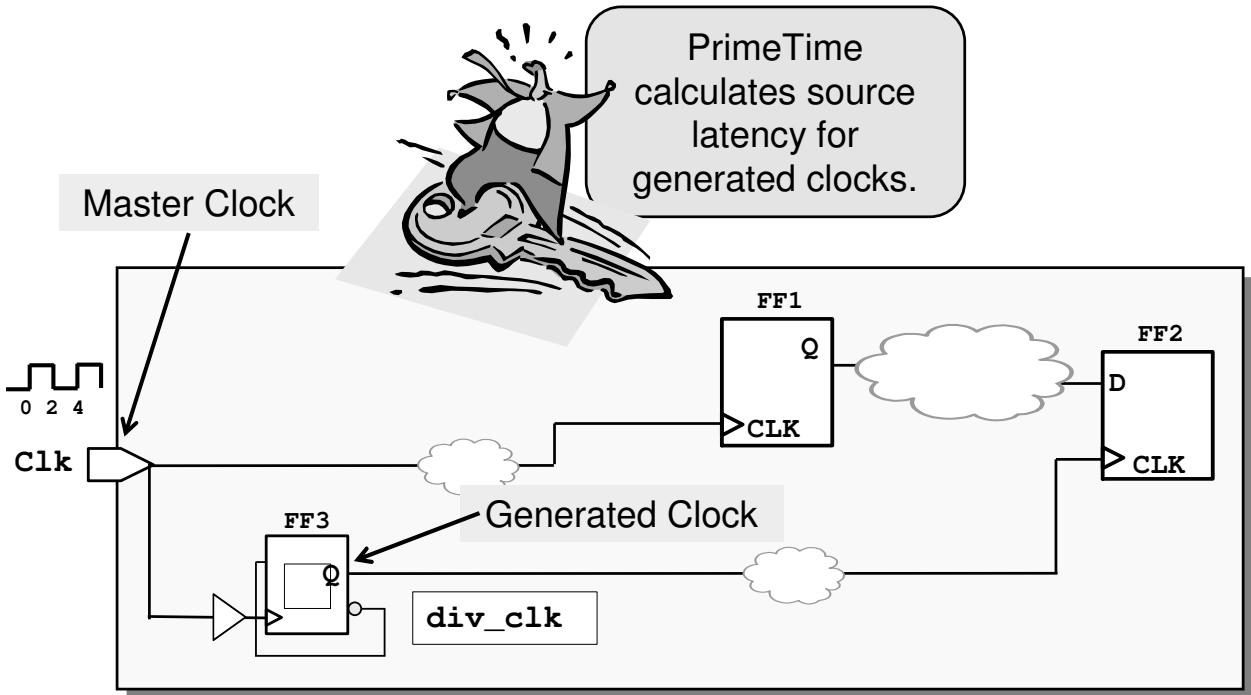
9-8

Generated clocks are generally created for waveform modifications of a master 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 master clocks and the generated clocks created from them.

```
# Master 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
```

Generated Clocks: Source Latency



9-9

If `div_clk` were created as a master 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.

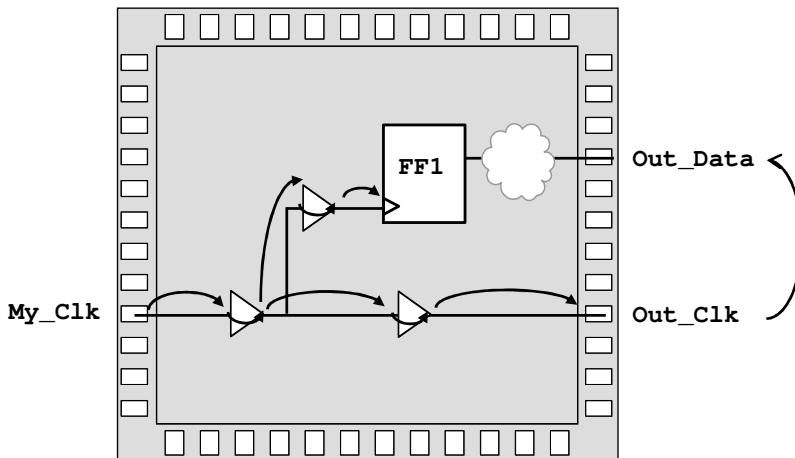
The master clock of the generated clock must be propagated for PrimeTime to calculate the source latency of the generated clock. If this is not true, PrimeTime will not calculate this source latency.

To see the calculated source latency, use

```
pt_shell> report_clock -skew
Min Condition Source Latency          Max Condition Source Latency
-----
```

Object	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

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.

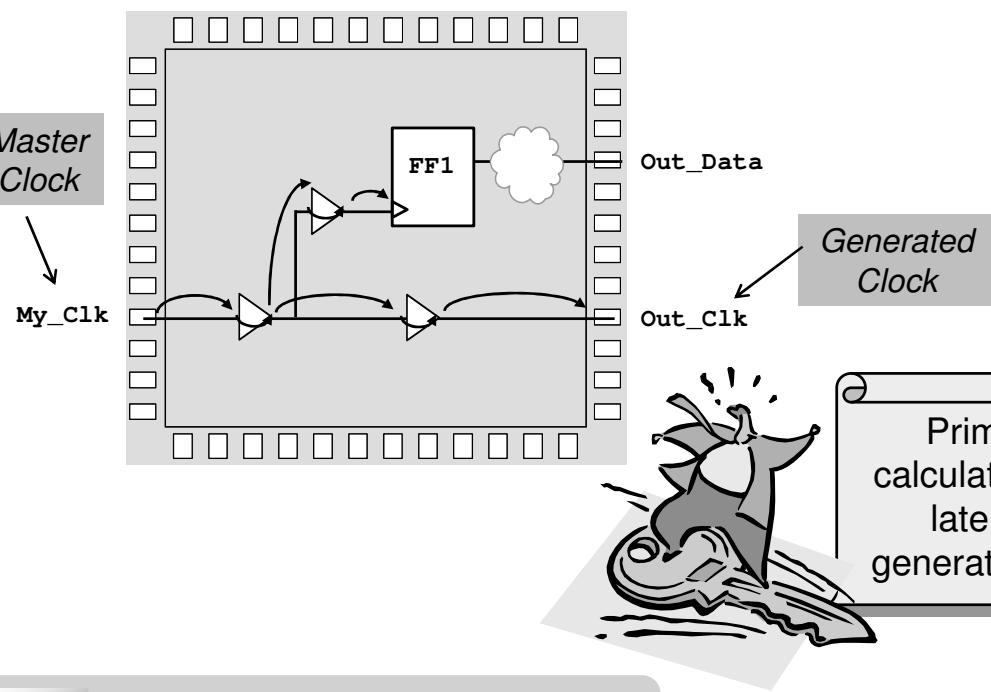
9-10

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



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



What happens if the clocks are ideal?

9-11

```
# 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]
```

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!

9-12

```
# 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]
```

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

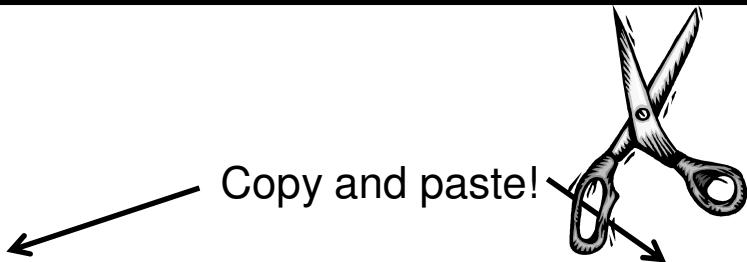
I - Inactive clock



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	{I_CLOCK_GEN/I_CLKMUL/CLK_2X}
SYS_CLK	8.00	{0 4}	p	

9-13

How Many Clocks Are In Your Design?



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

All clocks includes all master, 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?

9-14

```
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.

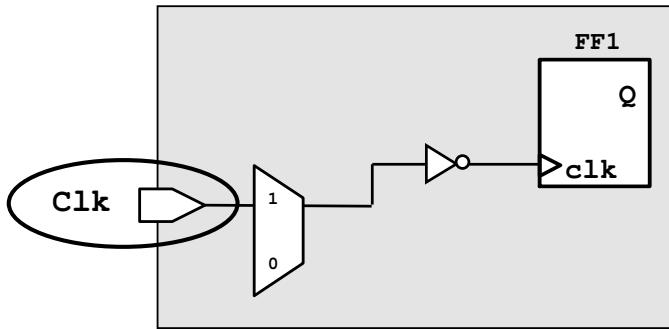
A Useful Tcl Procedure

A Tcl procedure that reports all ports with clocks defined.

```
pt_shell> rpt_clock_ports
```

Port Name	Direction	Clock Name	Is Generated

Clk	in	myclk	false



9-15



For a similar Tcl procedure (get_clock_ports), refer to SolvNet article “**Script to find all clock ports in a design**”.

Doc Id: 012739 **Last Modified:** 09/10/2004

If you would like to look at the body of a Tcl procedure while in PrimeTime, execute the following.

```
pt_shell> info body rpt_clock_ports
```

Test For Understanding 1/2



```
pt_shell> rpt_clock_ports
```

Port Name	Direction	Clock Name	Is Generated
<hr/>			
pclk	in	PCI_CLK	false
sys_clk	in	SYS_CLK	false
sdr_clk	in	SDRAM_CLK	false
sd_CK	out	SD_DDR_CLK	true
sd_CKn	out	SD_DDR_CLKn	true



Offer one reason for generated clocks
to be created on output ports?

9-16

Test For Understanding 2/2



```
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'.



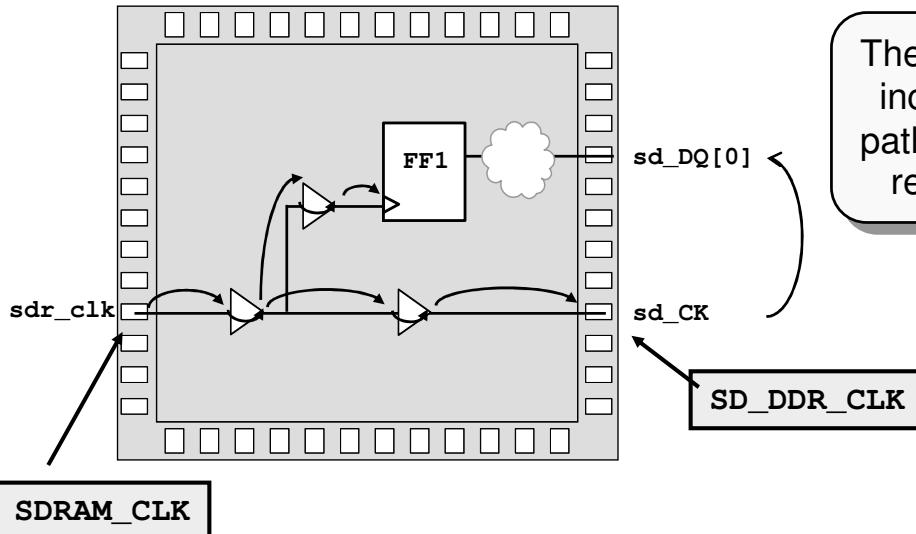
Continuing the example from the previous page, can the warning above be ignored?

9-17

Generate a Timing Report

For calculated source latency details of generated clocks.

```
report_timing -to sd_DQ[0] -path_type full_clock_expanded
```



The chip specification includes a min/max path requirement with respect to **sd_ck**.

9-18

Test for Understanding



Use `-path_type full_clock_expanded` with virtual clocks:

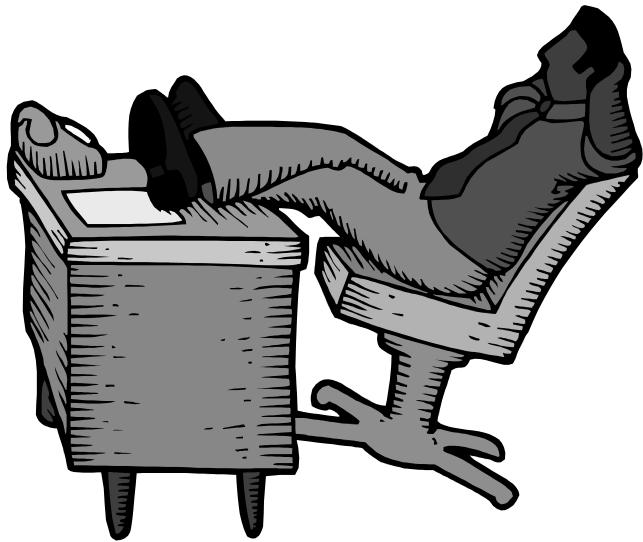


- To see the calculated source latency.
- This will not provide additional information.



9-19

Break



9-20

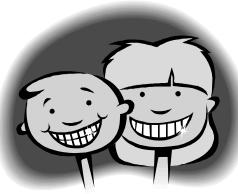
Asynchronous, Synchronous, & Exclusive Clocks

3 Types of Clocks

**Asynchronous, Synchronous,
and Exclusive Clocks**

9-21

Synchronous versus Asynchronous Clocks



Define:



Synchronous clocks

Asynchronous clocks



Use STA to verify timing between synchronous clocks?

Yes No

Use STA to verify timing between asynchronous clocks?

Yes No

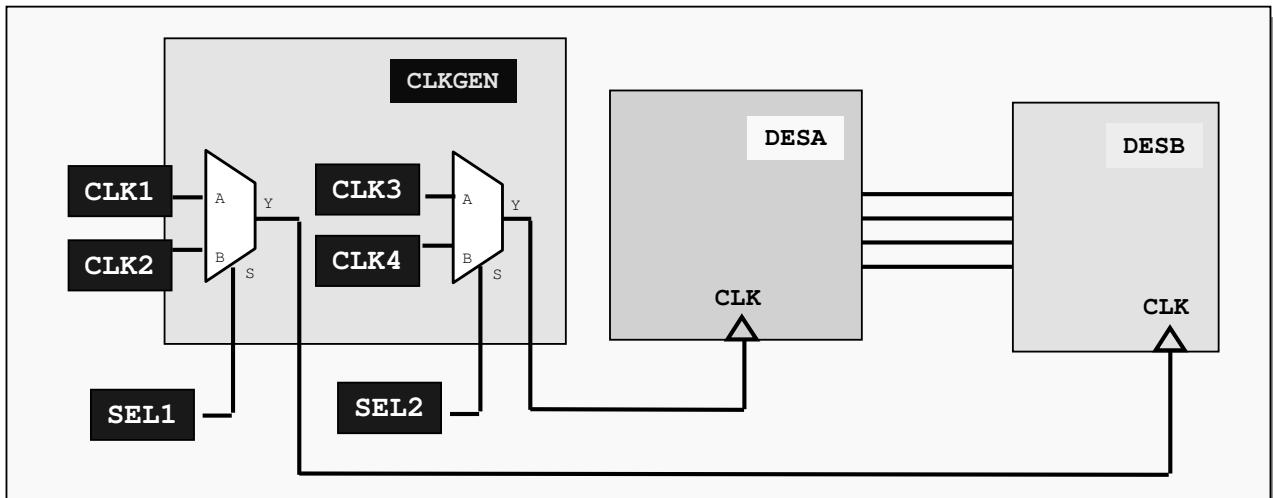
9-22

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

Interacting Clocks and Multiple STA Runs



4 clock combinations over 2 analysis types means 8 STA runs!

9-23

4 clock combinations:

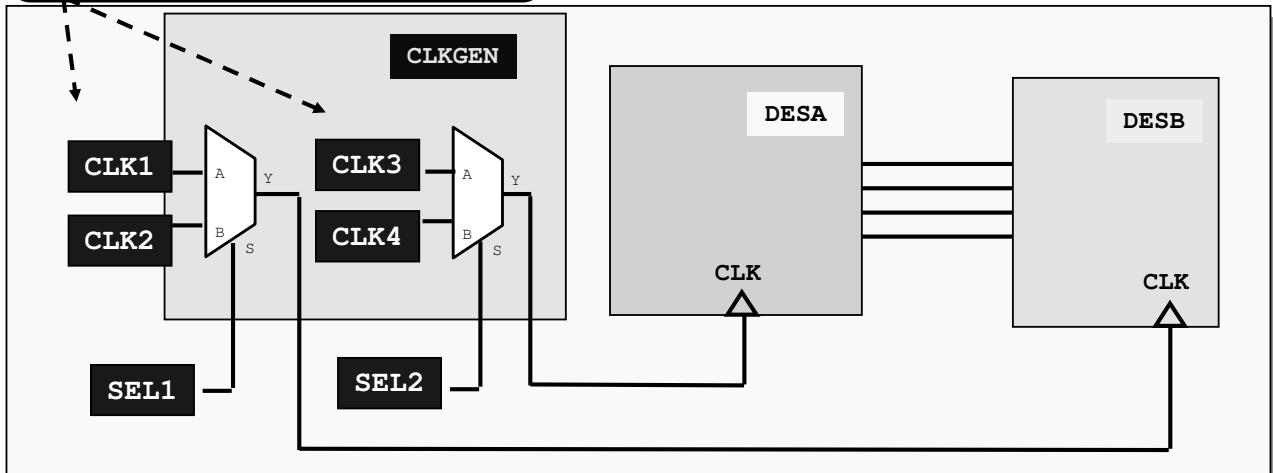
- | | |
|--------------------|--------------------|
| <u>DESA</u> | <u>DESB</u> |
| 1. CLK1 | CLK3 |
| 2. CLK1 | CLK4 |
| 3. CLK2 | CLK3 |
| 4. CLK2 | CLK4 |

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.
→ Use `-add` if necessary.



Step #2: Specify exclusive clock groups.

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

9-24

```
pt_shell> man create_clock
```

```
...
```

```
-add
```

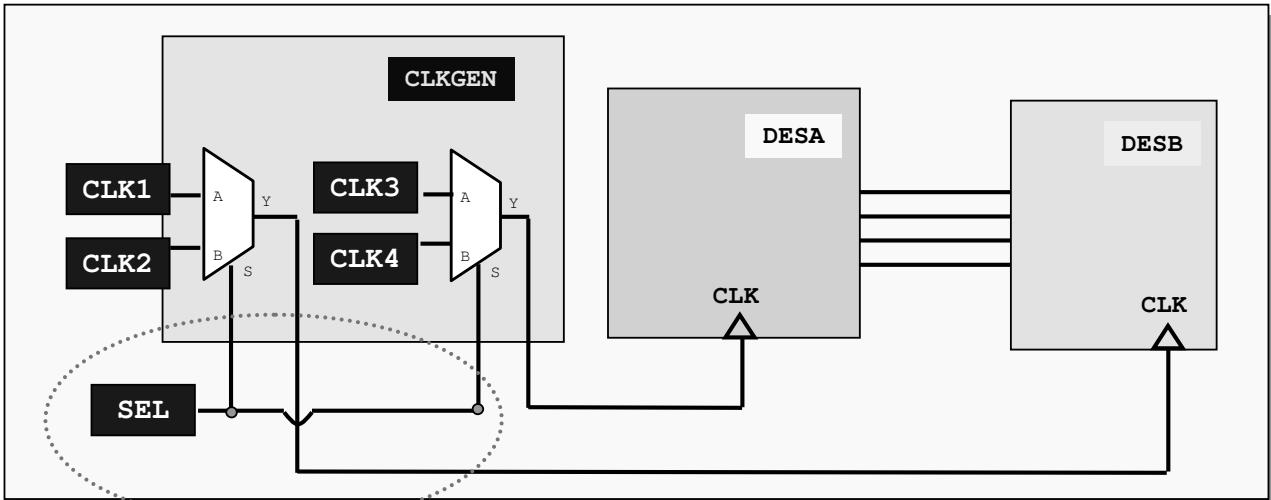
Specifies whether to add this clock to the existing clock or to overwrite. Use this option to capture the case where multiple clocks must be specified on the same source for simultaneous analysis with different clock waveforms. When you specify this option, you must also use the `-name` option.

Multiple clocks per register was new in v2002.03. Clock groups was new in T-2002.09.



Prior to v2002.03, if more than one clock drove a single register, PT only analyzed the last clock created.

Different Example: Multiple Clocks



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

9-25

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
Version: T-2002.09-2
Date   : Fri Nov  8 21:52:29 2002
*****
Active clocks:
  CLK1 CLK2 CLK3 CLK4

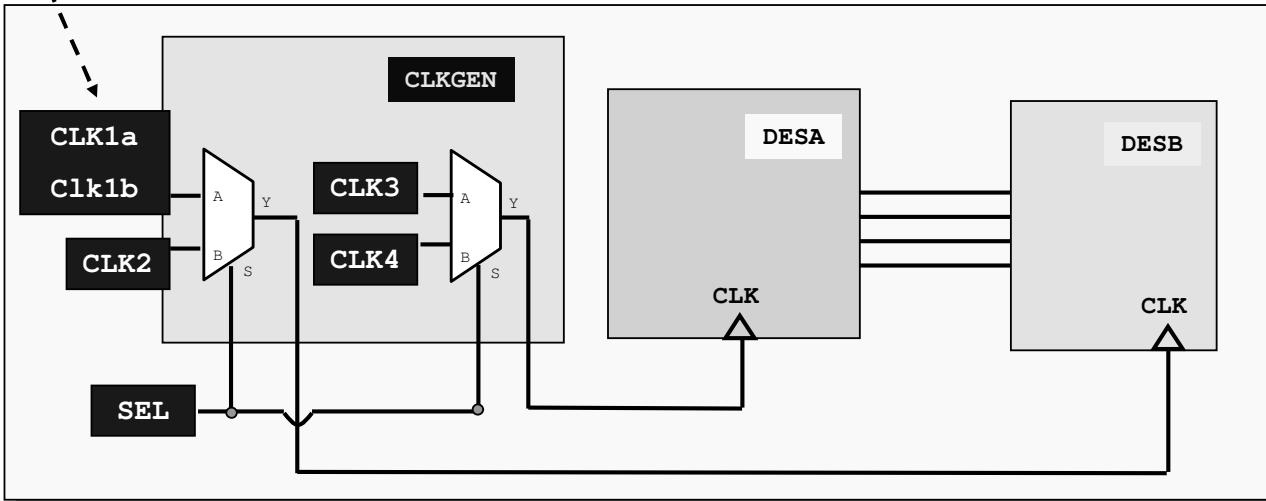
Total exclusive groups: 1.
NAME : SEL
      -group {CLK1 CLK3 }
      -group {CLK2 CLK4 }
```

Test for Understanding

15 Minutes!



CLK1 has a range of possible duty cycles.
→ 40/60 to 60/40



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

9-26

For question #2, assume the definition point for CLK1a and CLK1b is the port named **CLK1**, and the 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
```

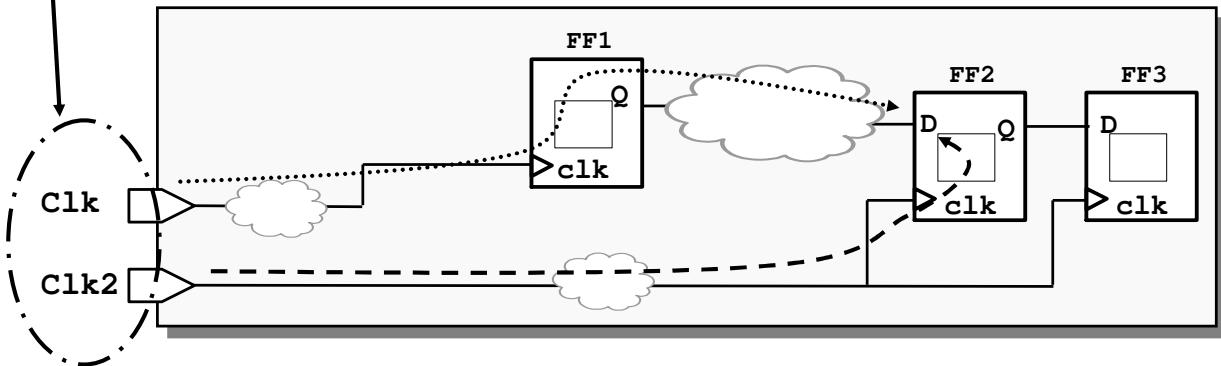
Specifies the rise and fall edge times of the clock waveforms of the clock, in library time units, over an entire clock period. The first time in the edge_list is a rising transition, typically the first rising transition after time zero.

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.

Asynchronous Clocks

Asynchronous
clock groups

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



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

9-27

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

Attributes:

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

Clock	Period	Waveform	Attrs	Sources
<hr/>				
Clk	4	{0 2}	p	{Clk}
Clk2	5	{0 2.5}	p	{Clk2}

Total asynchronous groups: 1.

```
NAME : my_asynch_clocks
      -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

9-28

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: )
```

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
```



Where will you
place this variable?

9-29

```
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.
```

Generate Timing Reports Between Clocks



Write the command to generate a timing report between C1k1 and C1k2.



9-30

Interpret Timing Reports Between Clocks

One by one, circle in the report below:



The launch and capture clocks.

The launch and capture clock edges.

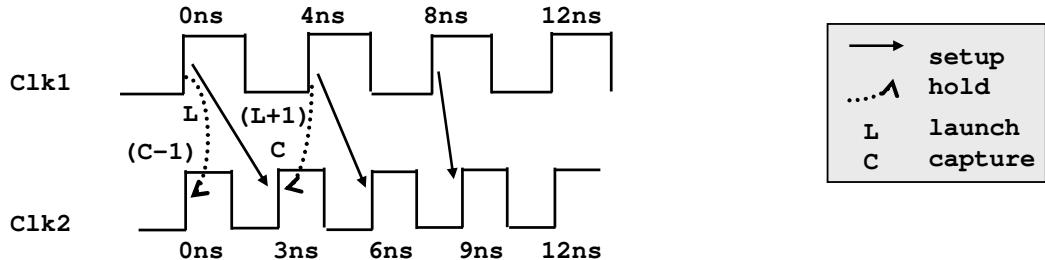
Point	Incr	Path
clock Clk1 (rise edge)	8.00	8.00
clock network delay (propagated)	1.10 *	9.10
FF1/CLK (fdef1a15)	0.00	9.10 r
FF1/Q (fdef1a15)	0.50 *	9.60 r
U2/Y (buf1a27)	0.11 *	9.71 r
U3/Y (buf1a27)	0.11 *	9.82 r
FF2/D (fdef1a15)	0.05 *	9.87 r
data arrival time		9.87
clock Clk2 (rise edge)	9.00	9.00
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

9-31

Which Edges for Setup and Hold

How PrimeTime finds clock edges for multi-frequency clocks.

1. Evaluate waveforms over 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 (Capture – 1); (Launch + 1) to Capture. From all of these hold relationships, pick the most restrictive.



9-32

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 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.**



**Are these messages you
would suppress and why?**

9-33

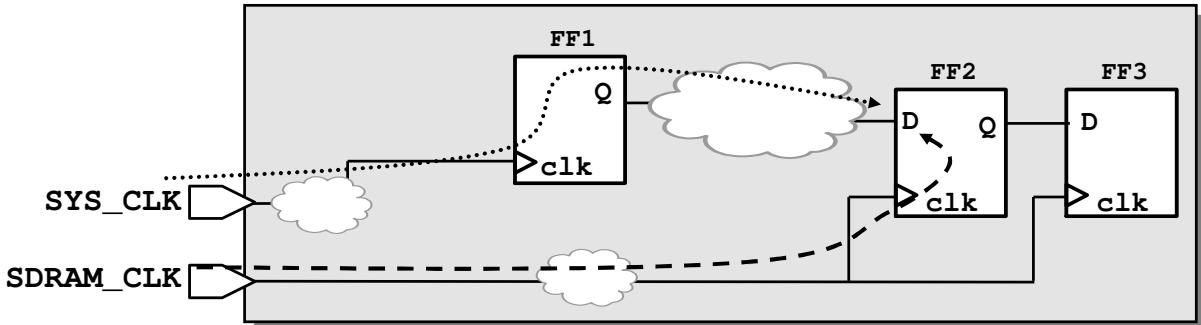
```
pt_shell> man PTE-064
```

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.

9-34

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 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)

9-35

```
pt_shell> man timing_report_unconstrained_paths
```

NAME

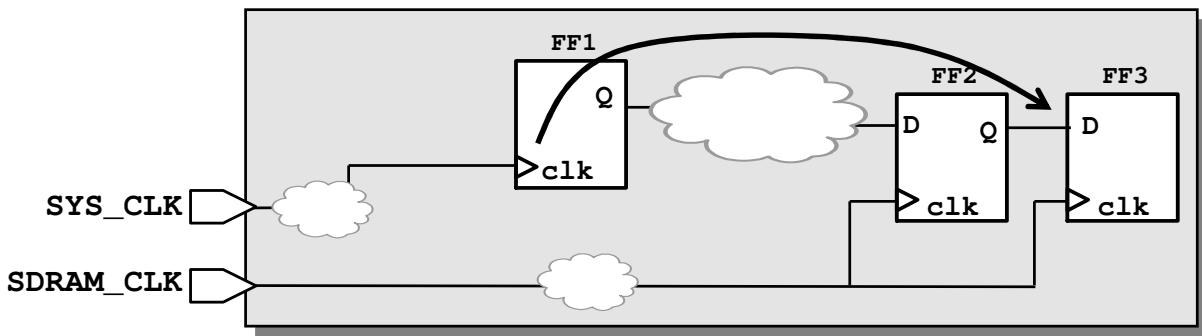
timing_report_unconstrained_paths Specifies if PrimeTime searches for unconstrained paths or not when report_timing or get_timing_paths command is used.

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

9-36

General Guidelines

- There should be no constrained paths between asynchronous clock domains



How will you validate this?

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



```
alias _____ {set timing_report_unconstrained_paths true}  
alias _____ {set timing_report_unconstrained_paths false}
```

9-37

pt_shell> **man UITE-413**

NAME

UITE-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.

Lab 9: Getting to Know Your Clocks



60 minutes



10 Minute BREAK

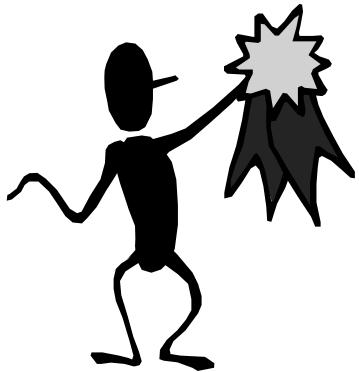
45 Minute LAB

The objective is to:

- **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**

9-38

Lab 9 Wrap Up: Propagated Clocks



All subsequently created clocks are propagated by default.

```
set timing_all_clocks_propagated true
```

After clocks are created, propagate them.

```
set_propagated_clock [all_clocks]
```

Flag ideal clocks in a design.

```
lappend timing_check_defaults ideal_clocks  
check_timing
```

9-39

This page was intentionally left blank.

Agenda

**DAY
3**

10 Analysis Type and Back Annotation



11 Additional Checks and Constraints



12 Conclusion

CS Customer Support

Unit Objectives



After completing this unit, you should be able to:

- **Describe two kinds of back annotation files**
 - Parasitics
 - SDF
- **Describe two methods for calculating path delay**
 - One “traditional” but inaccurate method
 - One safe method recommended for all designs
- **Describe how to apply and the benefits of the types of annotation files**

10-2

Topics in this module

1. Parasitic back annotation
2. Delay (SDF) back annotation
3. Slew effects
4. Controlling slew affects with analysis mode
5. Supporting analysis mode with libraries, back annotation files, and constraints
6. Accuracy: Removing reconvergent clock pessimism; Derating delays
7. Validating an SDF file
8. Back annotation-enabled checks – Design Rule Checking
9. Advantages of parasitics over SDF

10-3

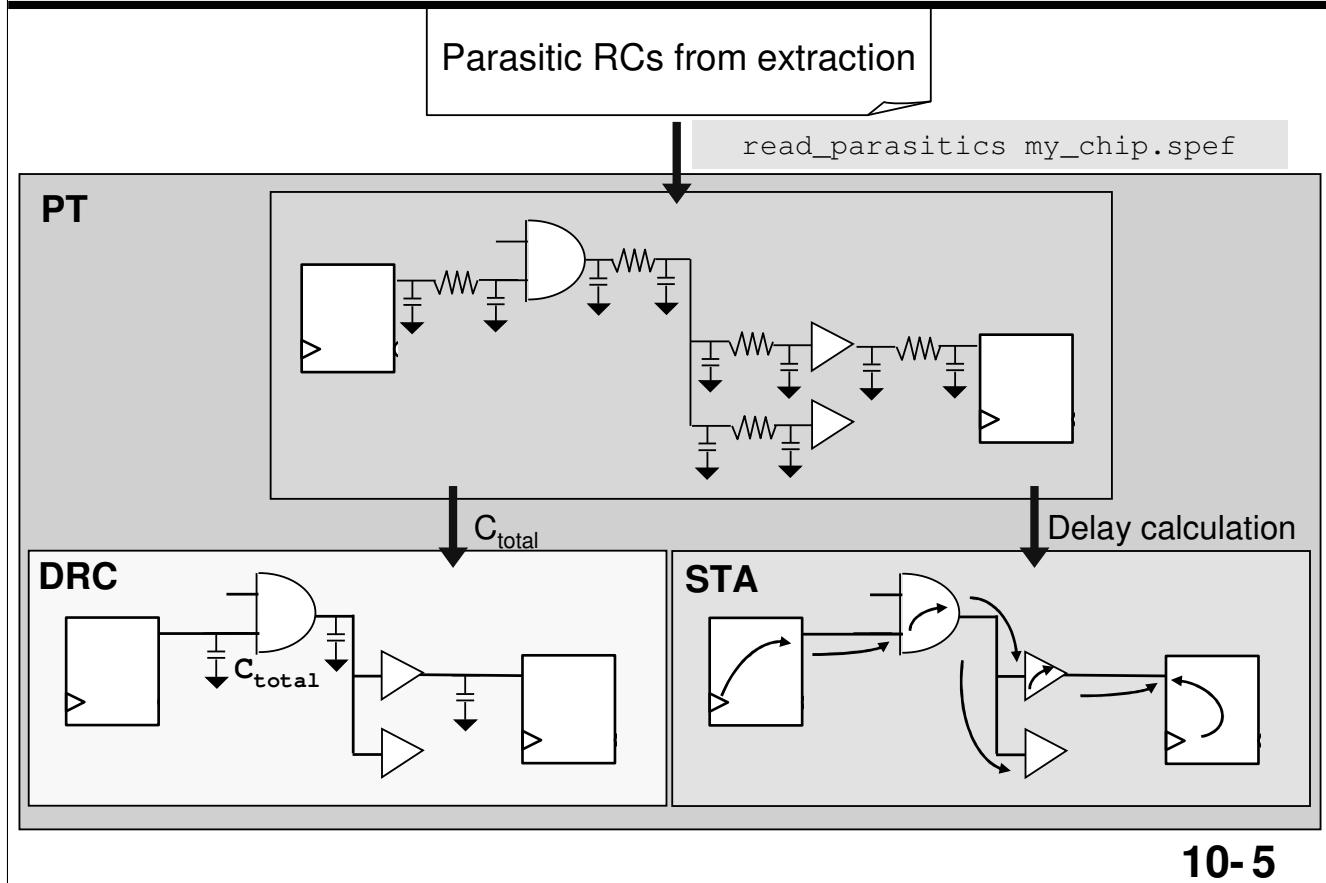
Back Annotation Files

- To perform post-layout STA, you back annotate data into PrimeTime with either a SPEF or an SDF file:
 - SPEF
 - ◆ From a layout extraction tool
 - ◆ Contains net R and C values
 - ◆ PrimeTime calculates ‘signoff quality’ net and cell delays and transition times based on these values
 - SDF
 - ◆ From a third party delay calculator, or from PrimeTime
 - ◆ Contains calculated net and cell delays
 - ◆ Primetime performs timing checks based on these delays

10- 4

PrimeTime can write out an SDF file for use in subsequent runs or for consumption by other tools. Increasingly, PrimeTime save and restore capability is replacing the long-time flow of reading parasitics into PrimeTime, then writing out SDF for use in subsequent runs to save run time.

Back Annotation for the PT Delay Calculator



10-5



DRC Design Rule Checks

PT accepts the following file formats for parasitic RCs:

- Cadence Design Systems Reduced Standard Parasitic Format (**RSPF**) and Detailed Standard Parasitic Format (**DSPF**).
- IEEE 1481 Standard Parasitic Exchange Format (**SPEF**).
- Synopsys Binary Parasitic Format (**SBPF**), which is written by PT-SI and STAR-RCXT.

Some examples of DRCs: `max_transition` and `max_capacitance`.

Cell Delay = Intrinsic Delay + Output Slew = $f(\text{Input Slew}, C_{out})$

Net Delay = $f(\text{Net C}, \text{Pin C}, \text{Net R})$

Both cell and net delays may also be scaled by an operating condition.

How PrimeTime Calculates Cell Delay

- Cell delay is a function of output loading and input slew
- Output loading and input slew affect the cell delay and output slew
- Output slew becomes the next cell's input slew

Example:			
Output load = 0.05 pF		Input slew = 0.5 ns	
Cell Delay = .23 ns		Output slew = 0.3 ns	

SPICE		Output Load (pF)			
		.005	.05	.10	.15
Input Trans (ns)	0.00	.1	.15	.2	.25
	0.50	.15	.23	.3	.38
	1.00	.25	.4	.55	.75

Cell Delay (ns)

SPICE		Output Load (pF)			
		.005	.05	.10	.15
Input Trans (ns)	0.00	0.10	0.20	0.37	0.60
	0.50	0.18	0.30	0.49	0.80
	1.00	0.25	0.40	0.62	1.00

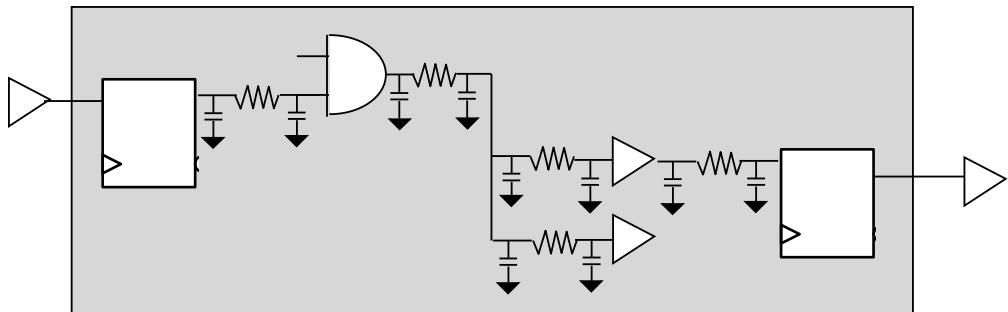
Output Slew (ns)

10-6

Non-linear delay models use Spice-derived timing at several input_transition and output_load points. Points not found in the tables are linearly interpolated. Note the presence of two tables: one for cell delay, and another for output transition (true rise/fall time). Since the output transition is used as the input transition at the next cell, it does affect the next cell's delay. Input transition also affects output transition in this model, which is used as the input transition to the following cell, so a slow transition on one cell can affect the delay of several following cells.

See the *Library Compiler User Guide* for more details.

Do Parasitic Files Supply All Needed Data?



■ What else do you need to supply?

- External drive cells for inputs

```
set_driving_cell -lib_cell mpad [get_ports A]
```

- External loads for outputs

```
set_load 5 [get_ports O1]
```

10-7

At the module level, set_driving_cell is required. At the chip level, if the driving chip is in your library, set_driving_cell still is a good choice – if the driving chip is not in your library, you will need set_input_transition.

Reading Parasitic RCs

```
pt_shell> read_parasitics clock_gen.spf.gz
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':
...
      |       |       |       |       |
Pin type | Total | RC pi | network | Not | Annotated |
-----+-----+-----+-----+-----+
internal net driver | 2214 | 0 | 2117 | 97 |
-----+-----+-----+-----+
design input port | 38 | 0 | 38 | 0 |
-----+-----+-----+-----+
      | 2252 | 0 | 2155 | 97 |

pt_shell> man PARA-006

WHAT NEXT
    Examine the SPF or SPEF file to determine why the
    specified RC network is incomplete...
```

10-8



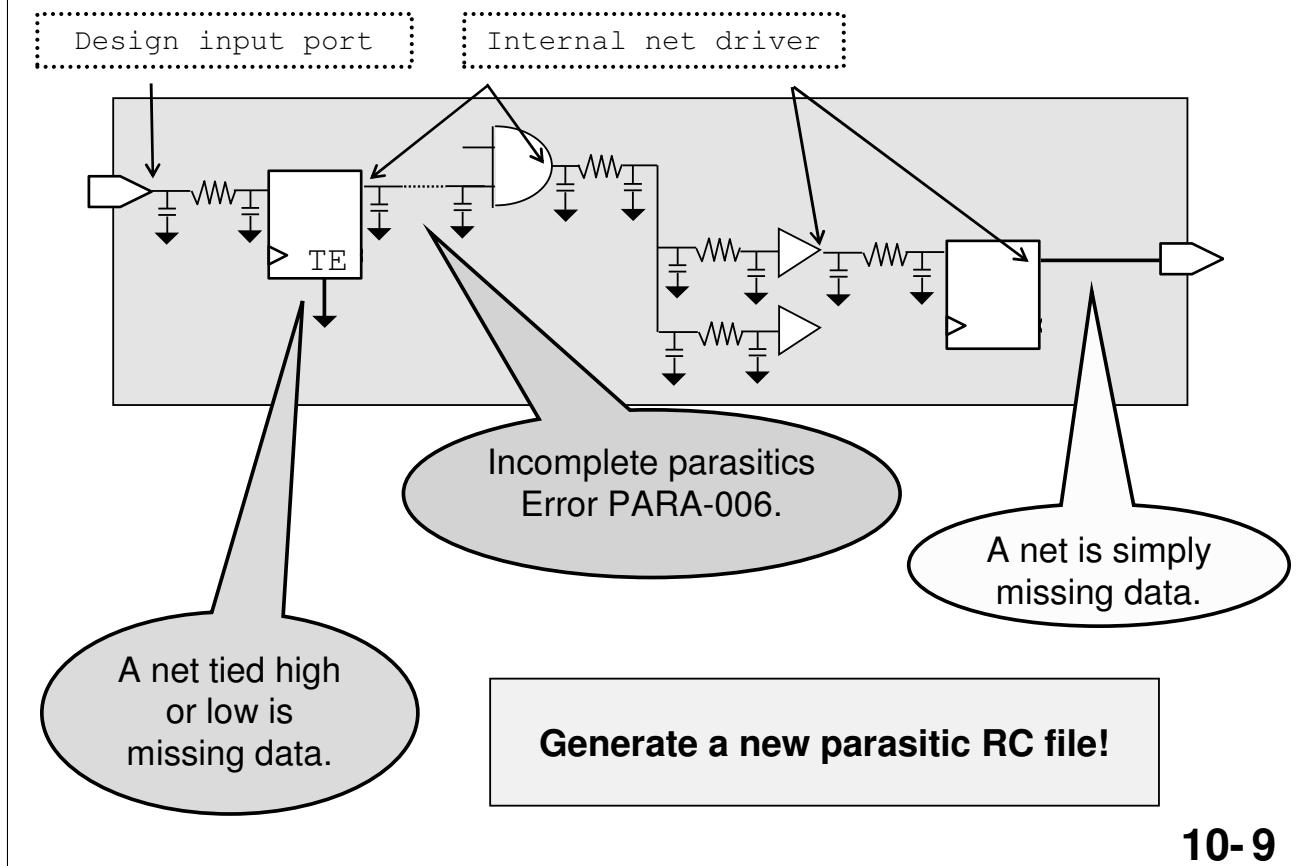
RC Resistance and Capacitance
RC Pi A reduced parasitic RC format
RC Network A detailed parasitic RC format

One parasitic RC file can contain a mixture of reduced or detailed formats. Each individual net is described in either a reduced or detailed format (not both!).



For further reading, refer to the online Documentation, PrimeTime User Guide:
Advanced Timing Analysis, unit 7 “**Parasitic Back-Annotation.**”

Examples of Missing Parasitic RC Data



10-9

Additional debugging hints:

- The parasitic file and the design netlist must match exactly
- Check to see if the missing parasitic RCs occur on nets tied high or low using,
`report_annotated_parasitics -list_not_annotated -constant_arcs`
- Report a summary for all nets missing parasitic RCs + any error messages,
`report_annotated_parasitics -check`
- For all nets missing parasitic RCs, PT will use wire load models to determine net RCs for STA and DRC

To remove parasitic RCs, use:

`remove_annotated_parasitics`

`pt_shell> man PARA-006`

. . .

DESCRIPTION:

You receive this message if `report_annotated_parasitics` detects an incompletely back-annotated RC network in the current design. The message informs you that the RC network annotation on the specified net is being ignored because it is incomplete; the specified pin is not physically connected to the RC network annotation.

Long Runtime Reading Parasitics

■ Use Synopsys binary parasitics



■ If reading multiple parasitic files

- Read from bottom up
- Suppress complaints about incomplete parasitics at the module level

```
read_parasitics -quiet -increment -path I_ALU I_ALU.spf.gz
read_parasitics -quiet -increment -path I_DATA_PATH \
    I_DATA_PATH.spf.gz
read_parasitics -increment TOP.spf.gz
```

Read top-level last

10-10



SBPF

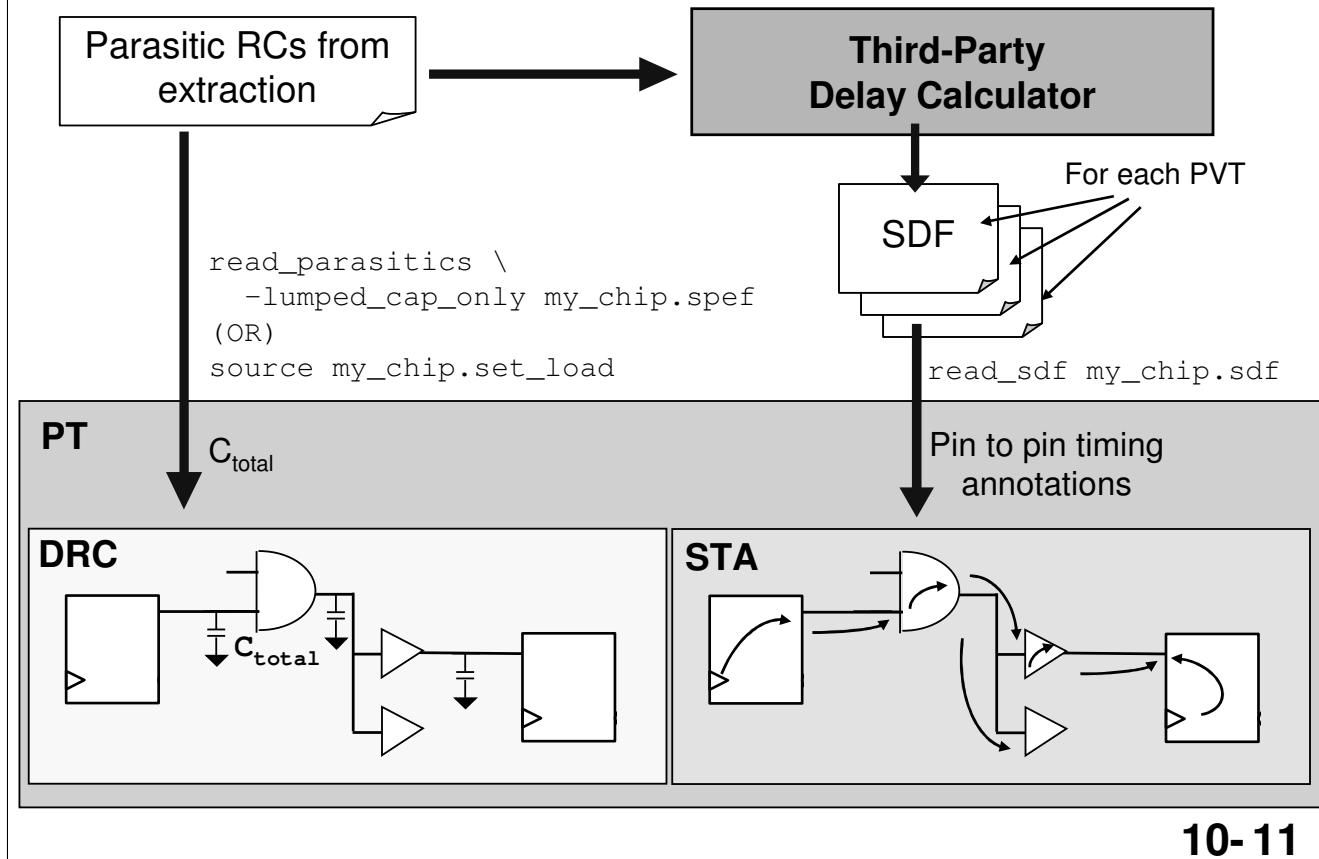
Synopsys Binary Parasitic Format, produced by PT-SI and Star-RCXT. Introduced in U-2003.03, SBPF supports hierarchical STA.

```
pt_shell> read_parasitics -format SBPF My_Chip.sbpf; # Read binary parasitics
pt_shell> man write_parasitics
```

DESCRIPTION

The write_parasitics command writes all parasitics annotated on the current design to the file specified in the file_name argument. write_parasitics supports the Synopsys Binary Parasitics format (SBPF) and requires a **PrimeTime-SI license**. Binary parasitics are very useful for analysis iterations. You can read parasitics in one format (for example, SPEF), then use write_parasitics to output the parasitics in SBPF format, **which loads much faster than ASCII formats and is much more compact**.

SDF: Using Third-Party Delay Calculators



10-11



SDF (Open Verilog International) Standard Delay Format.

SDF contains: Net and cell delays; Timing checks (e.g. Setup, hold, recovery and removal); Conditional arcs (A leaf cell timing arc could have different delays depending on the state (high or low) of other input pins on the same leaf cell. Conditional arcs became supported by PT in v1999.10). PT accepts SDF v1.0 - v2.1 and v3.0.

SDF does not contain parasitic RCs or cell transition time as a discernable number:

- You can not perform design rule analysis
 - You can not report capacitance or transition information in `report_timing`
- For these two reasons, read lumped C information.

What Affects SDF Generation?



Circle 4 things that affect
SDF generation.

Net parasitics

Drive and load
on ports

Clock Period

How slews are
propagated

Input and
output delays

PVT Corner

10-12

Answer: Drive and load, the operating conditions, slew propagation and net parasitics



PVT Process, Voltage and Temperature

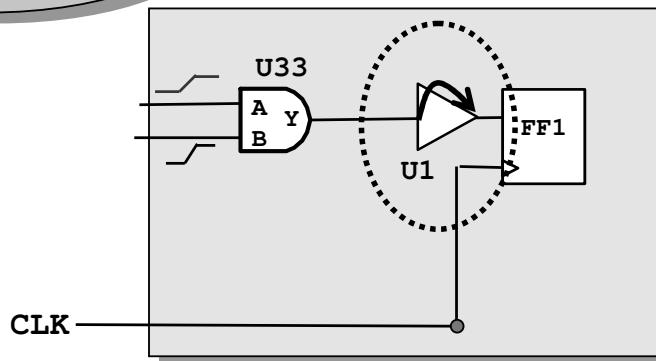
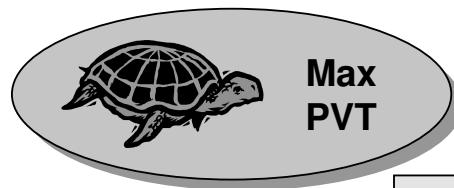
Effects of Slew



In a single PVT corner, the same cell delay timing arc may have:



- 2 different delays based on slow/fast slew propagation.
- 1 delay only.



10-13

Answer: 2 different delays

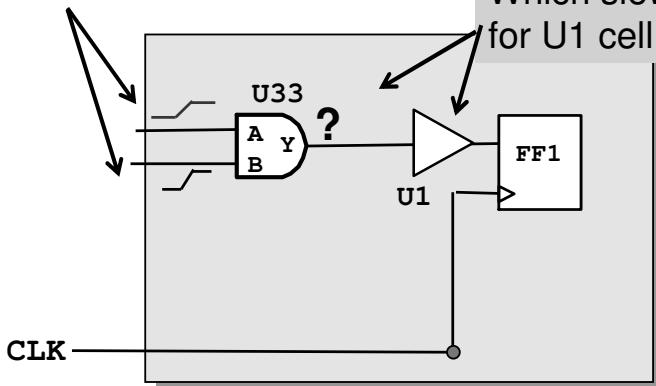
(2 delays are due to slow and fast propagation, 2 different delays)

Cell delay is a function of input slew which is determined by the slew propagation from the previous cell.

PrimeTime Slew Propagation and Cell Delays

Two different input pin slews.

Which slew gets propagated
for U1 cell delay calculation?



- Ideally, the specific slew for each path under analysis is used:

- Computationally prohibitive for the entire design



If path-specific slew propagation is too time consuming, what are the alternatives?

10-14

Answer: An alternative is to propagate the worst slew for one analysis and the best slew for another analysis – 2 analyses instead of too many.

Propagate both the slow slews and fast slews alongside each other consistently. This will provide the fastest and slowest possible timing bounds. The reality during operation is somewhere in between.

Another example of a slew merge point is a load pin of a multi-driven net.

Using the worst and best slew for propagation at slew merge points for delay calculation is called “**worst_slew**” propagation mode in PrimeTime.



For further reading, refer to Online Documentation, PrimeTime User Guide: Fundamentals, Unit 7 “**Timing Analysis Conditions**” under Slew Propagation.



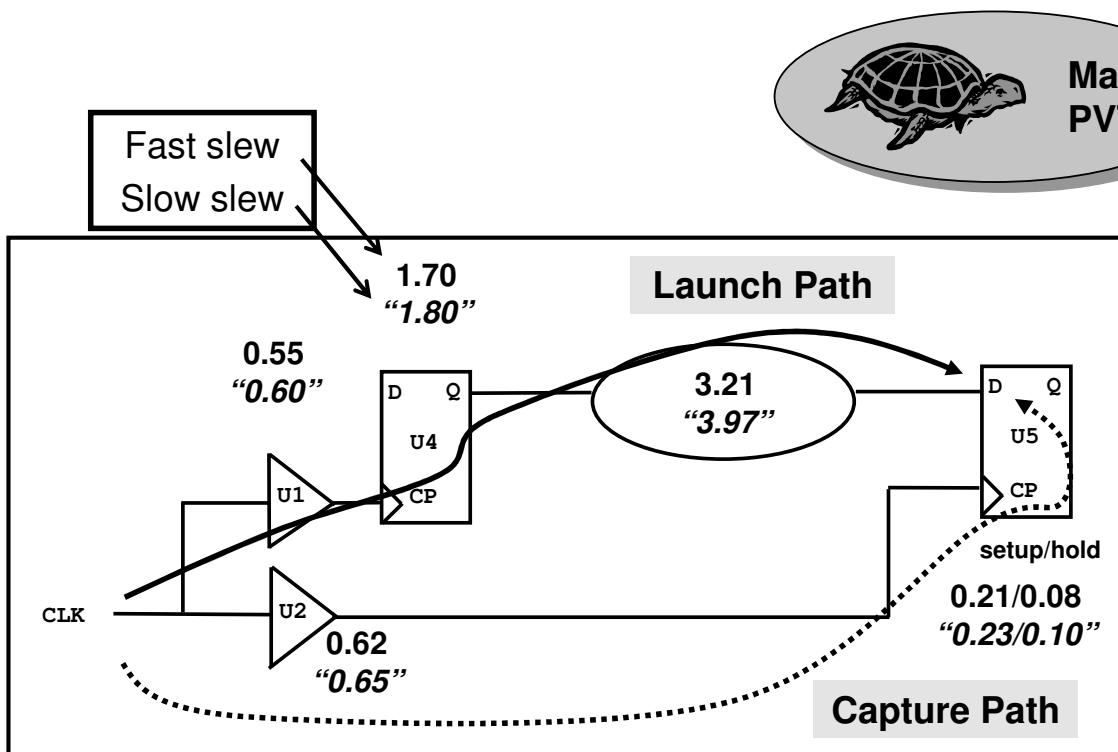
PrimeTime supports path specific analysis and slew propagation. This was a new feature in v2004.06. Please refer to the PrimeTime User Guide: Advanced Timing Analysis, Unit 5 “**Advanced Analysis Techniques**” under Path Specific Timing Analysis.



For more information on the default behavior of PrimeTime and slew propagation, refer to SolvNet “**Has the default behavior for the timing_propagate_single_condition_min_slew variable changed in PrimeTime W-2004.12?**”.

Doc Id: 014472 Last Modified: 12/10/2004

Launch versus Capture Path – Use Which Slew?

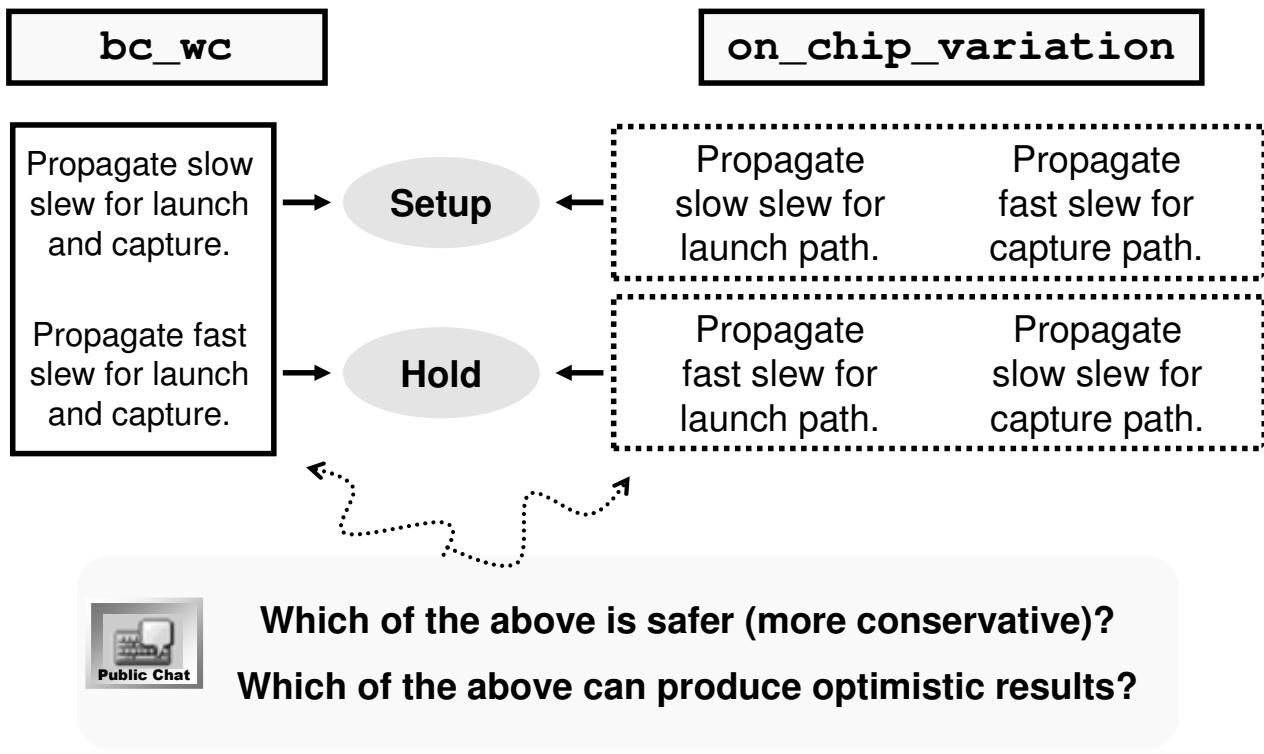


10-15

Launch Path = U1 + U4 + Logic Cloud

Capture Path = U2 + U5_Setup/Hold

Which Slew Goes with which Analysis Type?

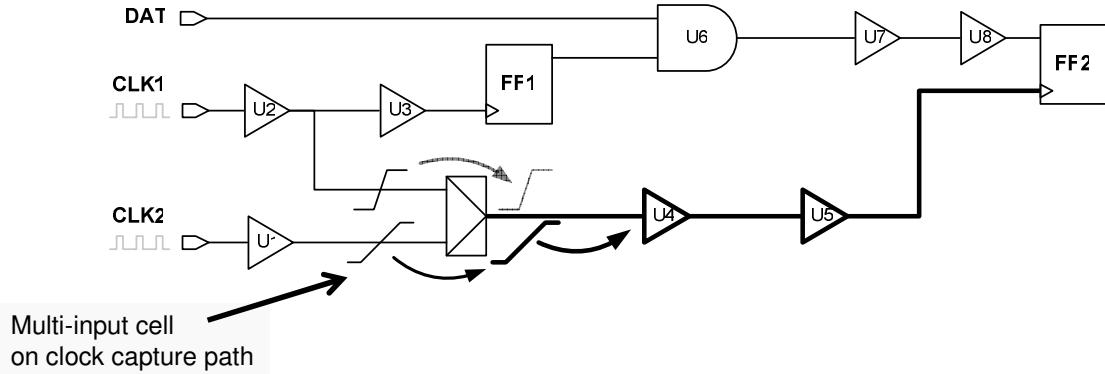


10-16

bc_wc – best case worst case

Setup Optimism in the bc_wc Mode

- Since ‘worst case’ uses all slow delays, setup checks experience optimism in the following case:

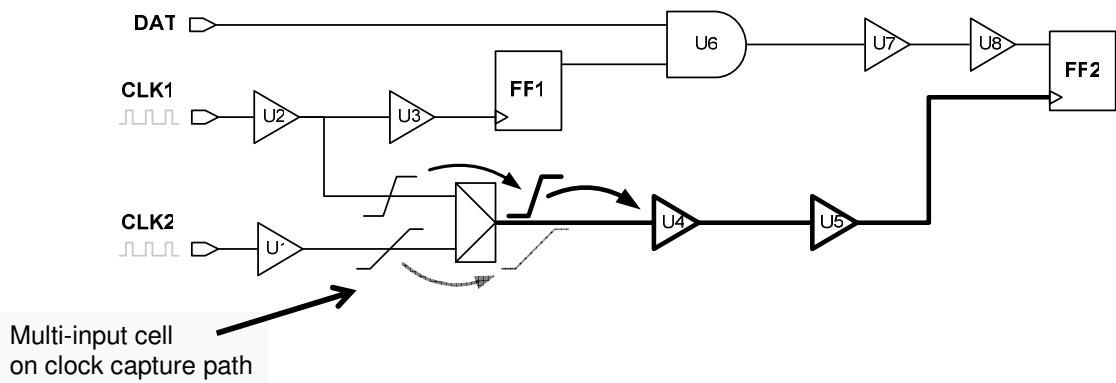


- Slow slew is propagated from mux output
- Slow delays computed for {U4 U5}
- But fastest possible capture path is needed
 - Setup analysis is optimistic for paths captured by CLK1!

10-17

Hold Optimism in the bc_wc Mode

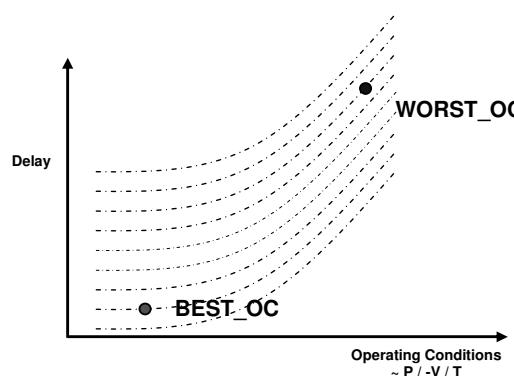
- Since 'best case' uses all fast delays, hold checks experience optimism in the following case:



- Fast slew is propagated from mux output
- Fast delays computed for {U4 U5}
- But slowest possible capture path is needed
 - Hold analysis is optimistic for paths captured by CLK2!

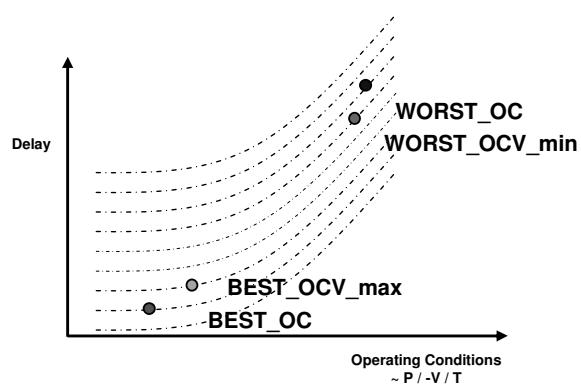
10-18

On Chip Variation: Single Library



Most optimistic OCV: Setup and hold for each corner

```
set_operating_conditions  
-analysis_type  
  on_chip_variation  
-max_lib WORST_LIB  
-max WORST_OC  
-min_lib WORST_LIB  
-min WORST_OC
```

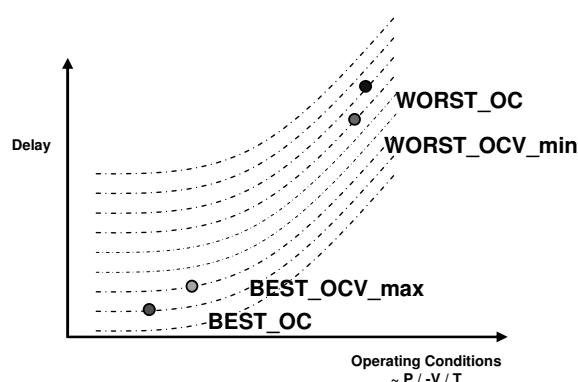


Pessimistic but safe: Setup and hold in each corner: calculated OCV OCs in library

```
set_operating_conditions  
-analysis_type  
  on_chip_variation  
-max_lib WORST_LIB  
-max WORST_OC  
-min_lib WORST_LIB  
-min WORST_OCV_min
```

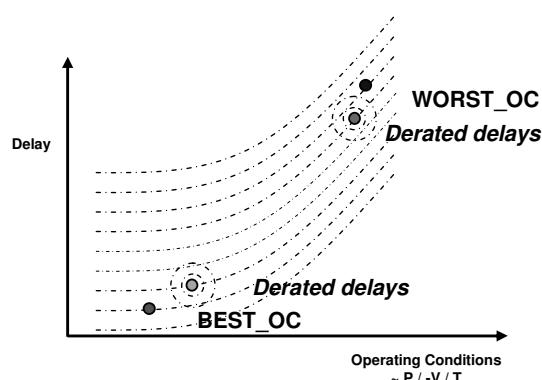
10-19

On-Chip Variation: Multiple Libraries, Derating



Most accurate: Setup and hold in each corner: separate calculated OCV libraries

```
set_operating_conditions
  -analysis_type
    on_chip_variation
  -max_lib WORST_LIB
  -max WORST_OC
  -min_lib WORST_LIB_OCV_min
  -min WORST_OCV_min
```



Pessimistic but safe: Setup and hold in each corner: user-specified derating

```
set_operating_conditions
  -analysis_type
    on_chip_variation
  -max_lib WORST_LIB
  -max WORST_OC
  -min_lib WORST_LIB
  -min WORST_OC
set_timing_derate -early ...
```

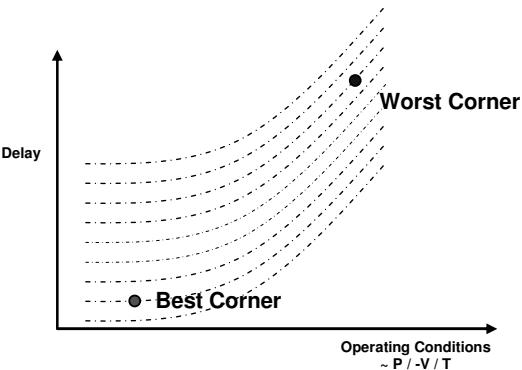
10-20

To Support OCV...

- To support recommended analysis of

- Setup and hold in each corner

recommended

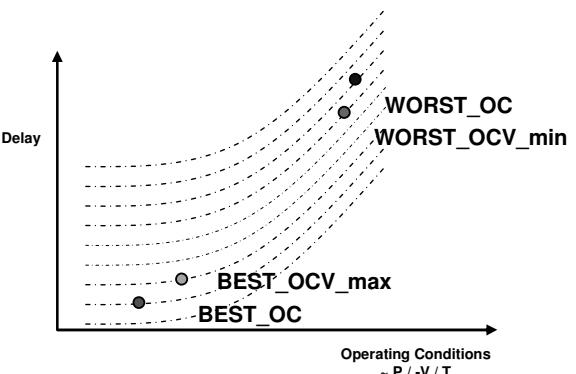


- How many do you need?

- Back annotation files
- Constraints

- Logistically

- How many runs?



10-21

How Many Back Annotation Files?

■ Parasitics – R + C values

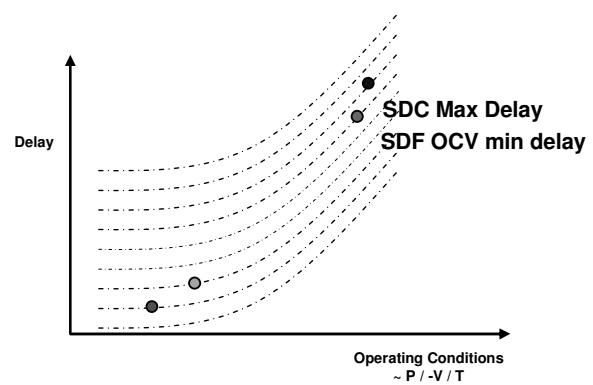
- One SPEF file for all corners
 - ◆ Primetime calculates delays for best case and worst case slews
- Multiple SPEF files
 - ◆ High cost
 - Extraction tool runtime
 - Multiple PT runs
 - ◆ Benefit generally measured in ps

← Excellent – for most designs

← More granular – for aggressive designs

■ SDF – Cell and Net delay

- min::max delays for each corner
 - ◆ Either one file with OCV min::max delays
 - ◆ Or two files representing a single corner
 - One with max ocv delays
 - One with min ocv delays



10-22

How Many Constraint Files?

Latency in OCV constraint file: one per corner

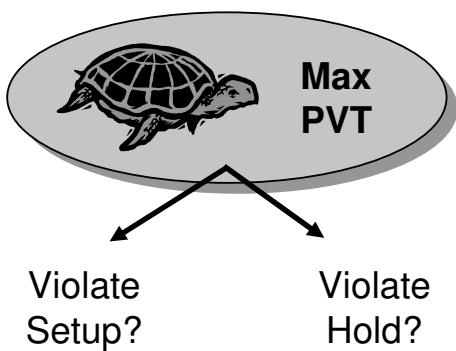
```
# worst case constraint file  
  
set_clock_latency -source -early 2.8 CLK  
set_clock_latency -source -late 3.2 CLK
```

```
# best case constraint file  
  
set_clock_latency -source -early 0.8 CLK  
set_clock_latency -source -late 1.2 CLK
```

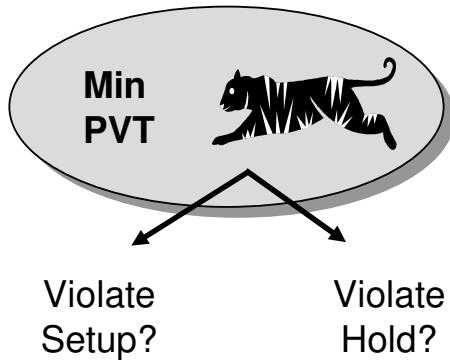
10-23

How Many OCV Runs?

Slowest Delays



Fastest Delays



 Offer one reason to perform setup and hold in both corners.

10-24

Answer: The clock skew can be much worst under slow PVT. The clock skew has a huge impact on hold. Therefore – you should do hold under slow PVT corner.

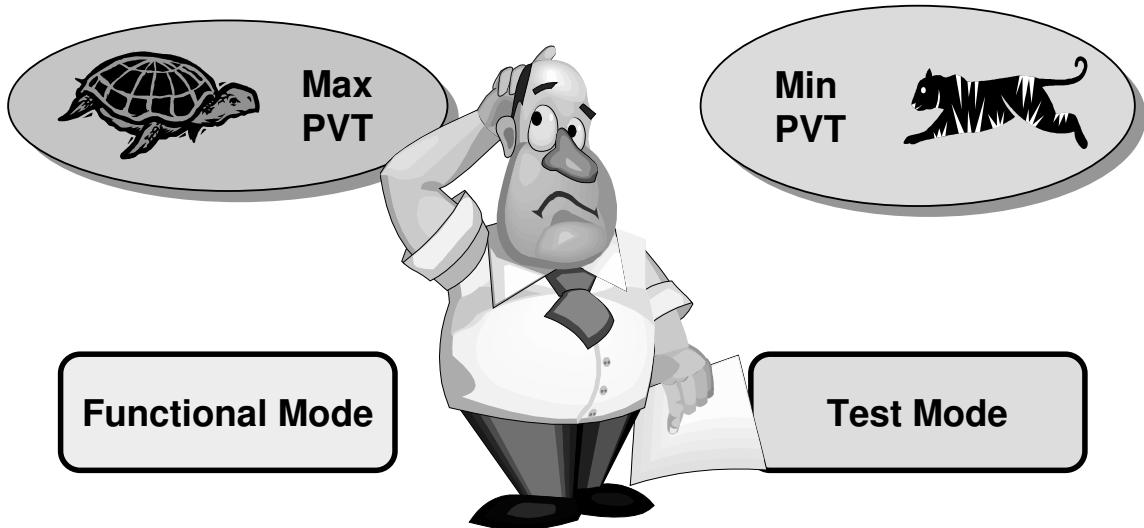


PVT Process Voltage and Temperature

PVT refers to a variation in cell delays and cell pin capacitances due to fabrication process variance, power supply voltage variations, and temperature (which could vary because of power consumption, ambient temperature, package type or cooling method).

Performing setup and hold checks under both PVT extreme corners is sometimes referred to as “four-corner analysis”.

STA Runs is Equal to?



How are different modes represented?

You will have to perform:



- 2 separate STA runs.
- 4 separate STA runs.

10-25

The # of STA runs = # Modes x # PVT corners.

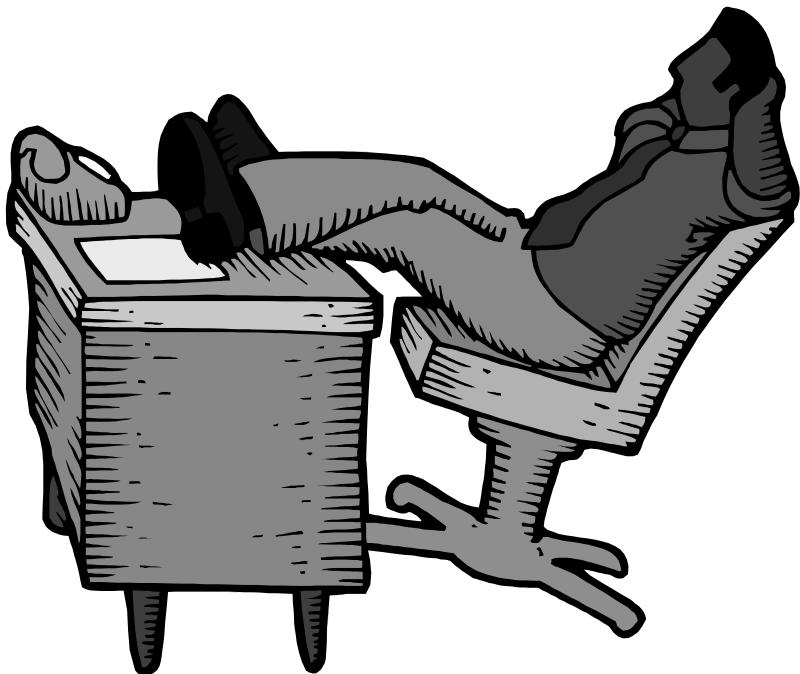
Answer – using different design constraints (and constraint files)

Answer – 4 runs. Functional and test under both slow and fast PVT.

Examples of constraints in test mode while the chip is a device under test (DUT):

- Tester clock period and clock source
- Model tester skew on the input ports
- Different timing exceptions
- Different setup/hold on the output ports
- The scan chain is exercised in test mode (but not in functional mode)

Break



10-26

Summary – Analysis Type

Default since 2004.12

- **Avoid optimistic results**
 - Use on_chip_variation

```
set_operating_conditions -analysis_type on_chip_variation  
report_design ;# to show current operating condition
```

- **Remove unrealistic clock pessimism caused by the two slews propagated in on-chip variation mode.**

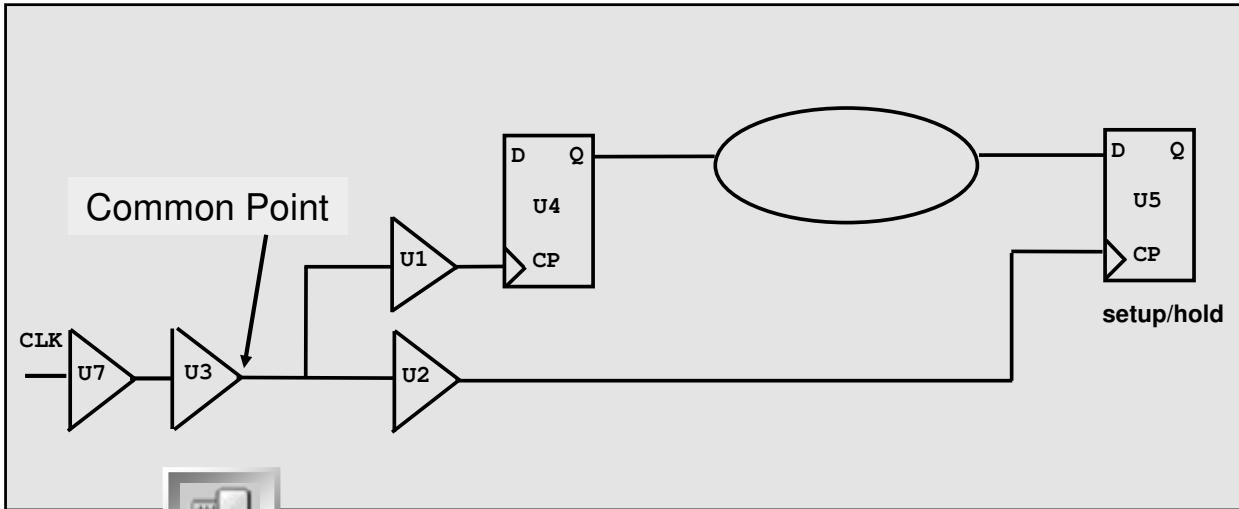
Turn on CRPR. 

10-27

Clock Reconvergence Pessimism (CRP)

CRP =

Latest arrival time to common point – Earliest arrival time to common point.



What causes a non-zero CRP?

The delay to the common point should be the same for the launch and capture path (i.e. CRP == 0).

10- 28

Common Point

If the same clock drives both the launching and capture flop, the launch and capture clock path will share a common subpath before branching. The output pin of the last common cell on this subpath is referred to as the common point.



For further reference, read the SolvNet “**PrimeTime Clock Reconvergence Pessimism Removal (CRPR) Application Note**”

Doc Id: 005511 **Last Modified:** 01/27/2004

If the **transition at the common point** differs between the launch and capture clock path (one is rise and one is fall), PrimeTime uses:

CRP = Smaller of {

(Latest RISE arrival time to common point – Earliest RISE arrival time to common point)

Or

(Latest FALL arrival time to common point – Earliest FALL arrival time to common point)

}

You can control this behavior with the variable `timing_clock_reconvergence_pessimism`.

Removal of CRP from STA (CRPR)

- By default, PrimeTime does not remove CRP
- When performing on-chip variation, set the variable below to true to remove CRP from STA:
 - Especially important for final sign-off when pushing design performance
 - Expect memory and runtime increase when using CRPR

```
set timing_remove_clock_reconvergence_pessimism true
```



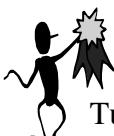
Where will you place this variable?

10-29

Recommendation on setting the variable `timing_crpr_threshold_ps`?

 By default, this variable is set to 20ps (the units are always in ps regardless of the library units). Choose a value (depending on your technology and vendor requirements) for a good balance between accuracy and runtime equivalent to one-half of a gate delay plus net delay of a typical stage in the clock network. Use a higher value early in the design cycle and a smaller value later in the design cycle.



 Turn off the variable `timing_early_launch_at_borrowing_latches` if CRPR is turned on for accurate analysis with latches experiencing time borrow.

The CRPR threshold variable affects STA with `report_timing`. To view the actual CRP and perform analysis on specific timing paths, use the following.

```
pt_shell> report_crpr -help;      # Reports the CRP calculation
      -from from_latch_clock_pin  (Clock pin of the launching reg)
      -to to_latch_clock_pin     (Clock pin of the capturing reg)
      -type setup/hold           (setup or hold check)
```



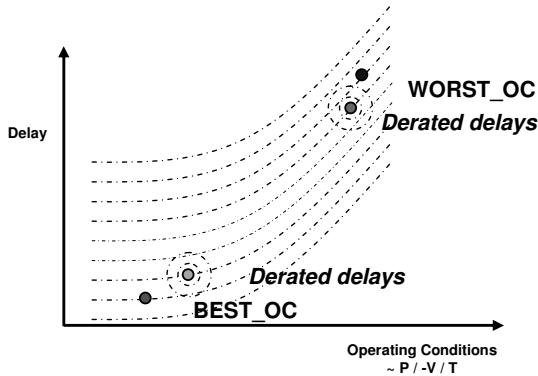
For information on removing CRP from paths that start at clock sources and a useful Tcl procedure, refer to SolvNet article “**How to determine CRPR credit in paths from clock sources**”.

Doc Id: 010459 **Last Modified:** 01/12/2005

Backannotation: Why use Derating Factors?

- Some PrimeTime users apply derating factors to account for effects such as:

- Static IR drop
- Temperature variation
- Transistor level effects
- On-die process variation
 - ◆ Chemical process
 - ◆ Mechanical process
 - ◆ Aging



The actual factors used must come from your own research!

10-30

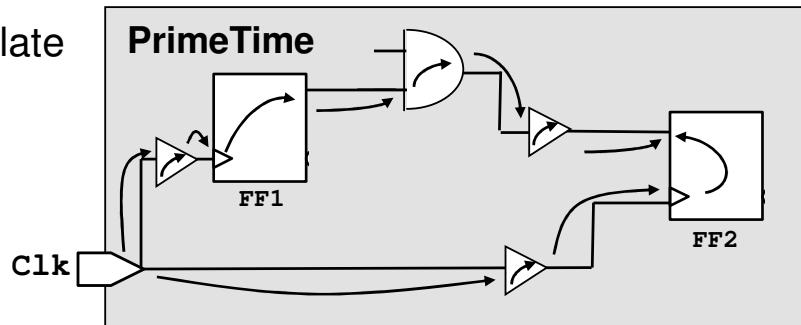


For further research, take a look at the SNUG paper by Denis Bzowy “**LOCV: Location-Based On-Chip Variation**” presented at Europe 2004.

```
pt_shell> set_timing_derate -help
Usage:
set_timing_derate      # Capture delay derating factor
[-cell_delay]           (Specify derate factor for cell delays)
[-cell_check]           (Specify derate factor for cell timing checks)
[-net_delay]            (Specify derate factor for net delays)
[-data]                 (Specify derate factor for data paths)
[-clock]                (Specify derate factor is for clock paths)
[-early]                (Specify early derate factor)
[-late]                 (Specify late derate factor)
[derate_value]          (Derate factor)
[object_list]           (List of cells or nets)
```

Back Annotation: Applying Derating Factors

- Derating factors will scale the delay values
- PrimeTime allows for global or specific derating:
 - Cells versus nets
 - Launch path versus capture path
 - Timing checks
 - Library cells
 - Early versus late



10-31

Derating factors are applied after slew is chosen

Global Derating versus Specific Derating

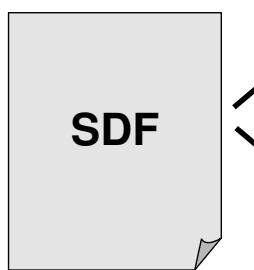
```
Startpoint: FF1 (rising edge-triggered fl Clk)
Endpoint: FF2 (rising edge-triggered flip flop Clk)
Path Group: Clk
Path Type: max
Max Data Paths Derating Factor : 1.100
Point           Incr      Path
-----  
clock Clk (rise edge)      0.00      0.00
clock network delay (propagated) 1.10 *   1.10
FF1/CLK (fdef1a15)        0.00      1.10 r
FF1/Q (fdef1a15)          0.50 *   1.60 r
U2/Y (buf1a27)            0.11 *   1.71 r
. . .
```

```
report_timing -derate
```

Point	Derate	Incr	Path
clock Clk (rise edge)	0.00	0.00	0.00
clock source latency	0.00	0.00	0.00
Clk (in)	0.00	0.00	0.00 r
clk_iopad/I (pc3d01)	1.00	0.05 *	0.05 r
clk_iopad/PAD (pc3d01)	1.00	0.95 *	1.00 r
U1/A (buf1a27)	1.00	0.01 *	1.01 r
U1/Y (buf1a27)	1.00	0.08 *	1.09 r
FF1/CLK (fdef1a15)	1.00	0.01 *	1.10 r
FF1/Q (fdef1a15)	1.10	0.40 *	1.50 f
U2/A (buf1a27)	1.10	0.01 *	1.51 f
. . .			

10-32

Are these OCV corners?



Meaningless!

What do these numbers represent?

Which PVT corner (or the same)?

How is the slew propagated?



Why should you not read and use one SDF number for both setup and hold?

10-33

How Will You Validate SDF?

With UDSM designs, there is much less margin for error!



You must understand what your SDF represents, how it was generated and how it should be applied during STA.

How will you validate the SDF information?

10-34

Open the SDF File and Take a Look!

Circle the:



Operating conditions used to generate the SDF.
Time units (which should match the library units).

```
(DELAYFILE
(SDFVERSION "OVI 2.1")
(DESIGN "ORCA")
(DATE "Mon Aug 25 22:57:18 2003")
(VENDOR "CLKMUL_lib PLL_lib core_slow.db io_slow.db")
(PROGRAM "Synopsys PrimeTime")
(VERSION "U-2003.03-SP1")
(DIVIDER '/')
// OPERATING CONDITION
"fast_0_1.98_BCT:::slow_125_1.62_WCT"
(VOLTAGE 1.98::1.62)
(PROCESS "1.000000::1.000000")
(TEMPERATURE 0.00::125.00)
(TIMESCALE 1ns)
```

orca.sdf



10-35

If the operating conditions are not noted in the SDF file, a conversation with the vendor may be needed to find out the conditions under which the SDF was generated.

Pay Attention to Output from read_sdf



Circle the:

Operating conditions used to generate the SDF.
Analysis type.

```
*****  
Report : read_sdf /.../design_data/orca_routed.sdf.gz  
        -load_delay cell  
        -analysis_type on_chip_variation  
        -min_type sdf_min  
        -max_type sdf_max  
Design : ORCA  
Version: V-2004.06-1  
Date   : Tue Sep 14 14:15:38 2004  
*****
```



**In which file will you find
this output logged?**

```
0 error(s)  
Number of annotated cell delay arcs : 364468  
Number of annotated net delay arcs  : 52896  
Number of annotated timing checks  : 41168  
Number of annotated constraints   : 7273  
TEMPERATURE: 125.00 (min) 125.00 (max)  
VOLTAGE     : 1.08 (min) 1.08 (max)  
PROCESS      : 1.200000 (min) 1.200000 (max)
```

10-36

Validate Design Analysis Type



Circle the analysis type below.



```
pt_shell> report_design
```

Design Attribute	Value
<hr/>	
Operating Conditions:	
analysis_type	single
operating_condition_max_name	slow_125_1.62
process_max	1
temperature_max	125
voltage_max	1.62
tree_type_max	balanced_case

10-37

The third analysis type is called **single**. This analysis type should not be used for the following reasons:

- PrimeTime will not allow you to apply constraints for hold time under single analysis type.
- With single analysis type – only a single set of delays may be applied to the design.
 - A single set of delays that represent a single PVT corner and a single slew propagation (for example, the slow slews are propagated) will be optimistic for hold analysis (or vice versa, if the fast slews are propagated this will be optimistic for setup analysis).

Completeness of SDF



In verifying the completeness of your SDF, what is the difference between the following two commands?

```
report_annotated_delay  
report_annotated_check
```

10-38

Reporting Design Rule Violations

```
read_parasitics clock_gen.spf.gz ← Parasitic flow  
or  
read_parasitics -lumped_cap_only my_chip.spf  
(OR)  
source my_chip.set_load ← SDF flow
```

report_constraint

Constraint	Cost	
max_delay/setup	3.5363	(VIOLATED)
min_delay/hold	0.0000	(MET)
recovery	0.0000	(MET)
removal	0.0000	(MET)
clock_gating_setup	0.0000	(MET)
clock_gating_hold	0.0000	(MET)
sequential_clock_pulse_width	0.0000	(MET)
min_period	0.0000	(MET)
max_capacitance	1.0015	(VIOLATED)
max_transition	1.3426	(VIOLATED)
max_fanout	0.0000	(MET)

10-39

Summary: Why use Parasitics instead of SDF?

■ Parasitics

- Accurate – Use BOTH of PrimeTime's strengths:
 - ◆ ‘signoff quality’ delay calculation
 - ◆ Static timing analysis
- Necessary for Signal Integrity Analysis
- Enables Design Rule Checking
- Responsive to change
 - ◆ Size cell
 - ◆ Constraint or mode changes
- Sign-off (at close of design cycle)

For speed, use `save_session` and `restore_session`

■ SDF

- Fast – uses pre-calculated delays
- Not responsive to change
- Often used early in design cycle

10-40

Useful Commands for Lab

report_annotated_delay	Report existing SDF values
report_annotated_check	Report existing timing checks

set_annotated_delay	Overwrite existing SDF with a new delay (discarding what was initially there).
----------------------------	---

remove_annotated_delay	Remove existing SDF or user specified delays or timing checks.
remove_annotated_check	

set_load	If SDF is missing, force net parasitic for specific nets.
set_resistance	

10-41

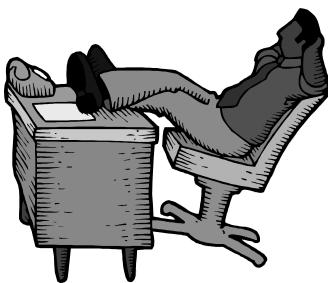
To see how specific timing arc delays are calculated (whether net or cell delays), use the command **report_delay_calculation**.

```
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)
```

Lab 10: SDF and Analysis Type



30 minutes



10 Minute BREAK
30 Minute LAB

The objective is to:

- **Debug a hold violation using your understanding of the SDF**

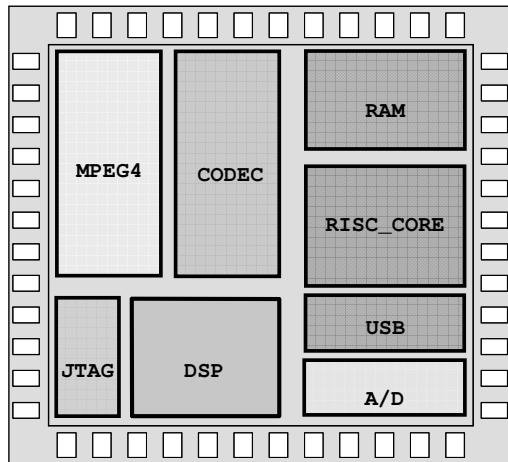
10-42

Lab 10 Wrap Up: Boundary Nets

Boundary nets do not represent anything on the physical chip.

The delays for these nets should be zero.

See SNUG paper in notes.



10-43



For a discussion of the example used in lab, refer to the SNUG article by Paul Zimmer
“**Complex Clocking Situations Using PrimeTime**” under the subsection named
“Phantom delays on input and output ports” presented at San Jose 2001.

Lab 10 Wrap Up: Missing SDF Annotations

- PrimeTime does not assume a zero delay for nets or cells missing SDF!
 - PrimeTime will calculate delays using either provided net parasitics or WLMs
- Timing reports have different annotations for the dominant annotations on that timing arc

Symbol	Annotation
-----	-----
H	Hybrid annotation
*	SDF back-annotation
&	RC network back-annotation
\$	RC pi back-annotation
+	Lumped RC
<none>	Wire-load model or none

10-44



WLM

Wire Load Model

pt_shell> **man report_timing**

...

The symbol is indicative only of the **dominant annotation**, and you should note that this might not have been used to calculate the observed delay. For example, assume a back-annotated RC network exists on a net, but that the network calculation fails to converge for the driver cell delay. In this case, you are alerted, and a lumped RC model is used instead; but, during report timing the "&" symbol appears anyway. To see how the pin-to-pin delay is actually calculated, use the `report_delay_calculation` command.

For more information on how a timing arc delay is being calculated, use the following command:

pt_shell> **report_delay_calculation -help**

```
report_delay_calculation # Report delay calc details for cell or net arc
  [-min]           (Report minimum delay calculation)
  [-max]           (Report maximum delay calculation - default)
  [-from from_pin] [-to to_pin]      (From and to pin)
  [-of_objects objects] (Report delay calculation for collect. of timing arcs.)
  [-nosplit]        (Do not split lines if column overflows - NYI)
  [-from_rise_transition float]     (Pass from rise transition)
  [-from_fall_transition float]     (Pass from fall transition)
  [-thresholds]       (Show library thresholds)
  [-crosstalk]        (Shows crosstalk information)
```

Lab 10 Wrap Up: CRPR



Is CRPR turned on?

```
pt_shell> aa reconvergence
***** Commands *****
Information: No commands matched '*reconv*'. (CMD-040)
***** Variables *****
timing_clock_reconvergence_pessimism = "normal"
timing_remove_clock_reconvergence_pessimism = "true"
```

report_timing -to sd_DQ[7] -delay min

clock SD_DDR_CLK (fall edge)	3.7500	3.7500
clock network delay (propagated)	3.0208	6.7708
clock reconvergence pessimism	0.0000	6.7708
output external delay	0.1000	6.8708
data required time		6.8708

data required time		6.8708
data arrival time		-6.6657

slack (VIOLATED)		-0.2051

10-45

This page was intentionally left blank.

Agenda

**DAY
3**

10 Analysis Type and Back Annotation



11 Additional Checks and Constraints



12 Conclusion

CS Customer Support

Review of Units 9 - 10



List the three types of clocks.



Why are generated clocks created on output ports and the benefit gained?



When will you use bc_wc analysis type versus on_chip_variation?

11-2

Unit Objectives



After completing this unit, you should be able to:

- **Address additional constraints and timing checks while performing STA on a design**
- **Recommend a timing modeling technology that meets the needs of deep sub-micron design**

11-3

Additional Timing Checks

Min Pulse Width Checks

Recovery and Removal

Latches and Time Borrow

Multicycle Paths

Combinational Feedback Loops

Non-unate Cells in Clock Path

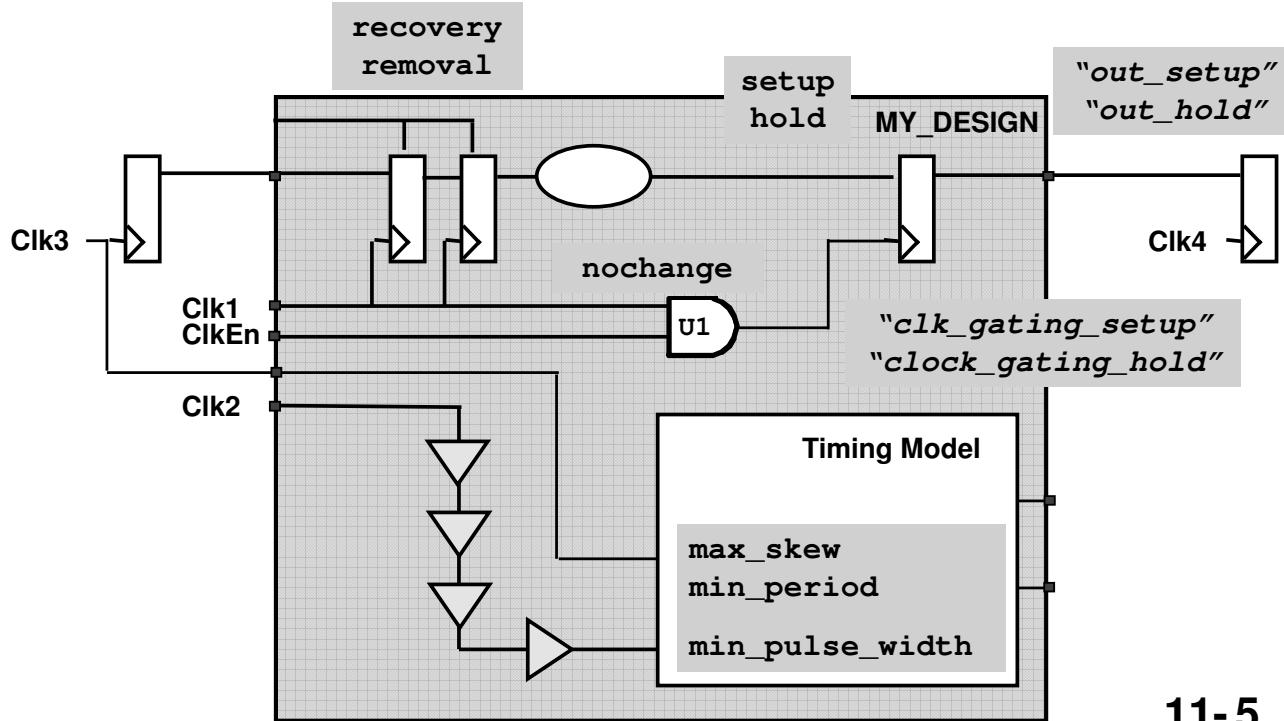
11-4

Timing Checks Verified by STA



Check the timing checks already covered.

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



11-5

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.

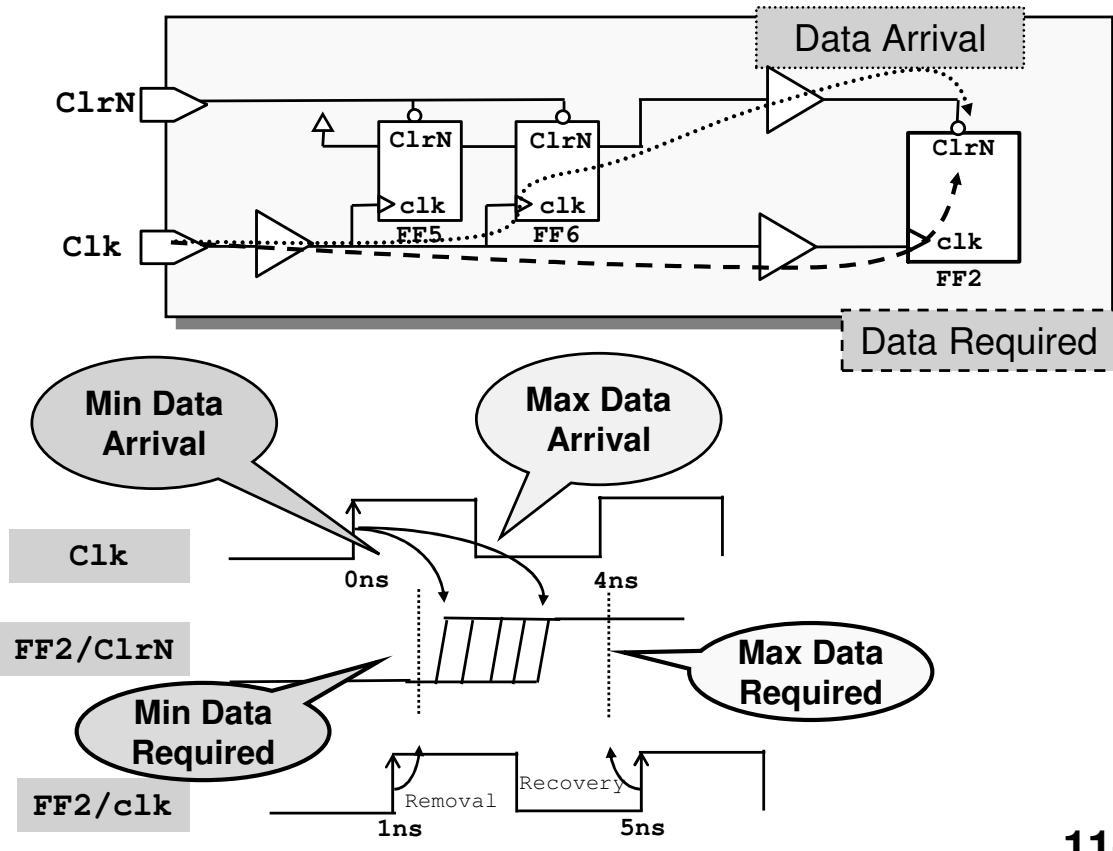
nonsequential: This is similar to a data to data setup and hold check between two data pins.

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`).

Asynchronous Clear/Reset Pins



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.

Data transition at the endpoint.

Timing path 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)

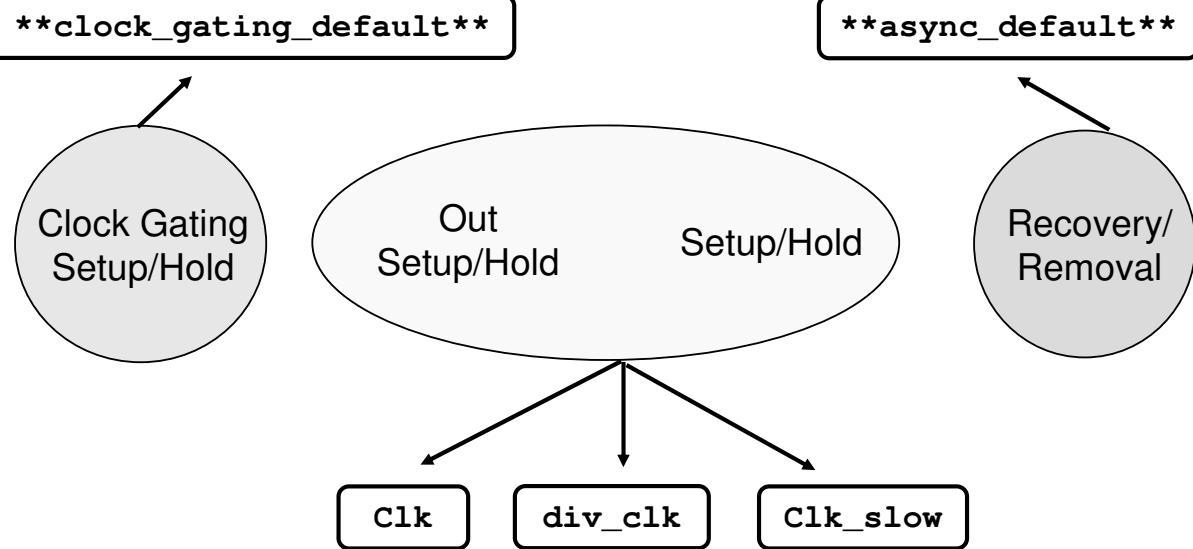
Path Group: **async_default**

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 (sdcrql1)	0.000	2.846 r
.	.	.
I_ORCA_TOP/I_RISC_CORE/I_ALU/Neg_Flag_reg/CDN (sdcrb1)	0.073 *	3.974 r
data arrival time		3.974
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)		6.833 r
library recovery time	0.128 *	6.962
data required time		6.962
data required time		6.962
data arrival time		-3.974
slack (MET)	2.988	

11-7

Recall Path Groups – Full Picture



Timing paths grouped by the capture clock.



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

11-8

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
```

Path_Group	Weight	Critical	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

Test for Understanding



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

What is the result?



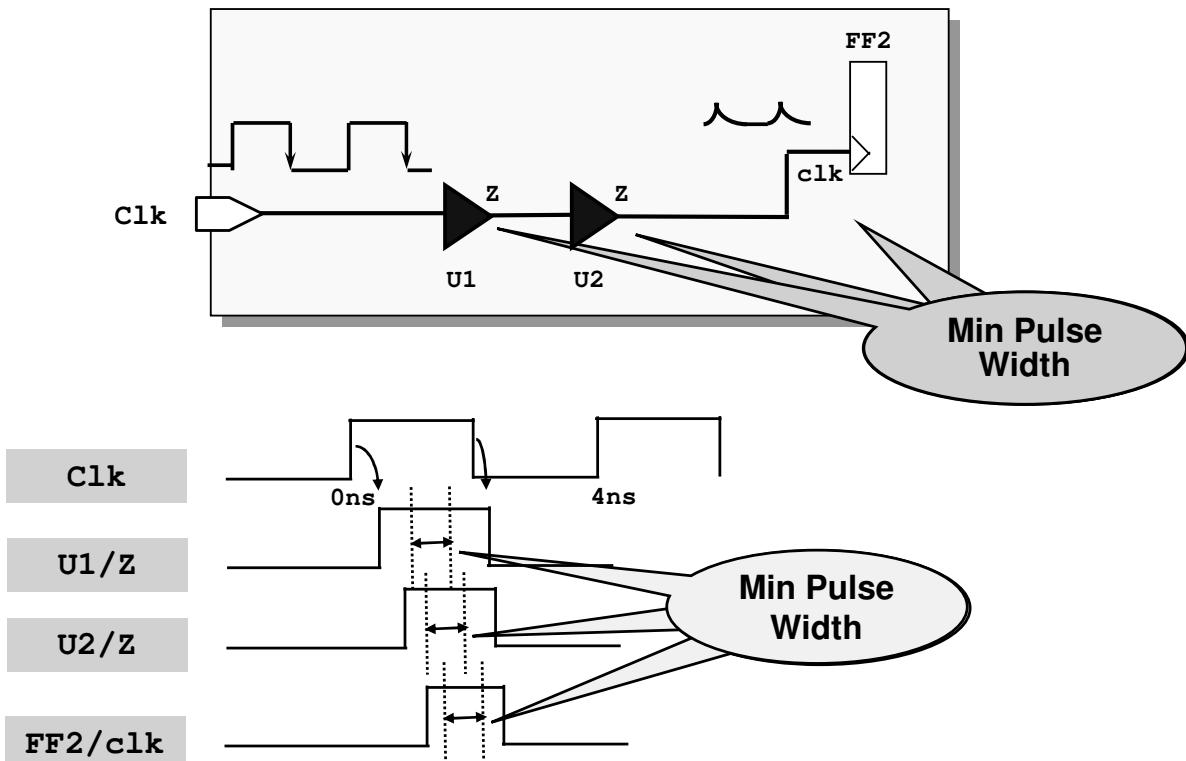
- No constrained paths.
- Will generate a timing report.



What is one solution if you are interested in this timing report?

11-9

Clock Min Pulse Width



11-10

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
sequential_clock_pulse_width

          Required      Actual
Pin           pulse width    pulse width    Slack
-----
I_ORCA_TOP/I_RISC_CORE/I_REG_FILE/REG_FILE_B_RAM/CE2 (high)
              0.419        1.587        1.169  (MET)
I_ORCA_TOP/I_RISC_CORE/I_REG_FILE/REG_FILE_B_RAM/CE1 (high)
              0.419        1.587        1.169  (MET)
I_ORCA_TOP/I_RISC_CORE/I_REG_FILE/REG_FILE_A_RAM/CE1 (high)
              0.419        1.589        1.170  (MET)
```

```
pt_shell> report_min_pulse_width -verbose \
I_ORCA_TOP/I_RISC_CORE/I_REG_FILE/REG_FILE_B_RAM/CE2
```



11-11

Report for Min Pulse Width

One by one, circle:



The constrained clock pin. Required pulse width.

This report is for a pulse width:

high

low

Pin: I_ORCA_TOP/I_RISC_CORE/I_REG_FILE/REG_FILE_B_RAM/CE2

Check: sequential_clock_pulse_width

Point	Incr	Path
clock SYS_2x_CLK (rise edge)	0.000	0.000
clock network delay (propagated)	2.833	2.833 r
I_ORCA_TOP/I_RISC_CORE/I_REG_FILE/REG_FILE_B_RAM/CE2	0.000	2.833 r
open edge arrival time		2.833
clock SYS_2x_CLK (fall edge)	2.000	2.000
clock network delay (propagated)	2.421	4.421 f
I_ORCA_TOP/I_RISC_CORE/I_REG_FILE/REG_FILE_B_RAM/CE2	0.000	4.421 f
close edge arrival time		4.421
required pulse width (high)	0.419	*
actual pulse width	1.587	
slack (MET)	1.169	

11-12

pt_shell> **report_min_pulse_width -help**

Usage:

```
report_min_pulse_width # Report min pulse width check
  [-significant_digits digits] (Number of digits to display: Range: 0 to 13)
  [-nosplit]           (Do not split lines when columns overflow)
  [-verbose]            (Print verbose report, default is summary only)
  [port_pin_list]       (List of objects to check min pulse width.)
```

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%)
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%)

11-13

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

Latches and Time Borrow

Min Pulse Width Checks

Recovery and Removal

Latches and Time Borrow

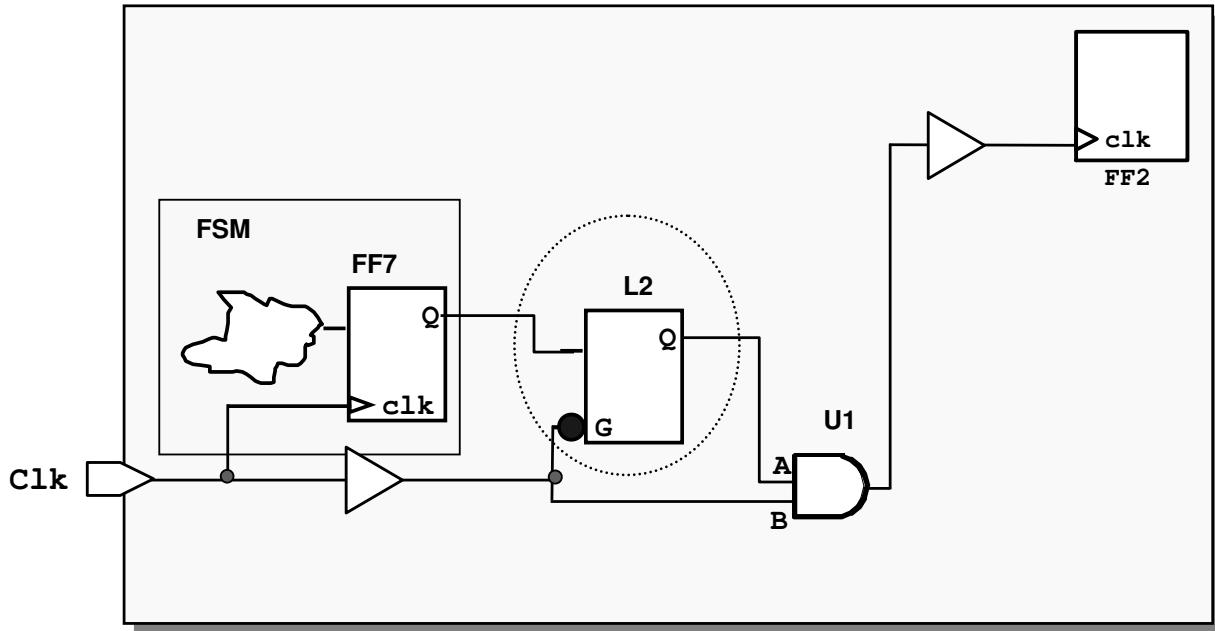
Multicycle Paths

Combinational Feedback Loops

Non-unate Cells in Clock Path

11-14

What About Latches?



11-15

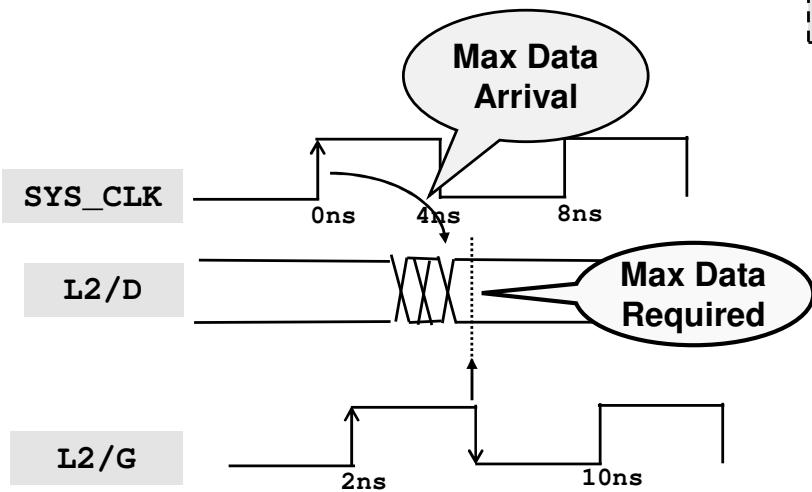
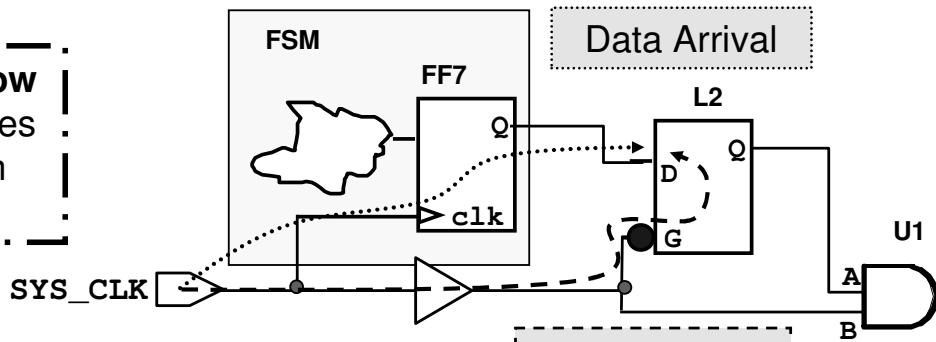
This is a common clock gating implementation by Power Compiler.



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

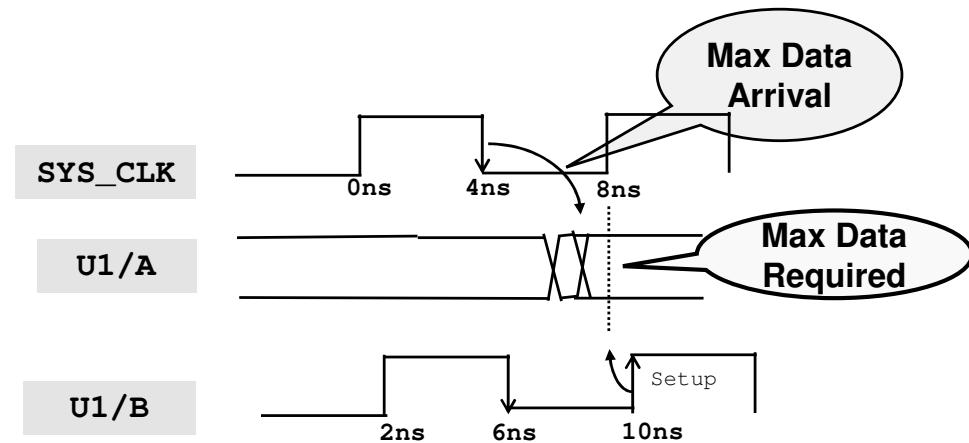
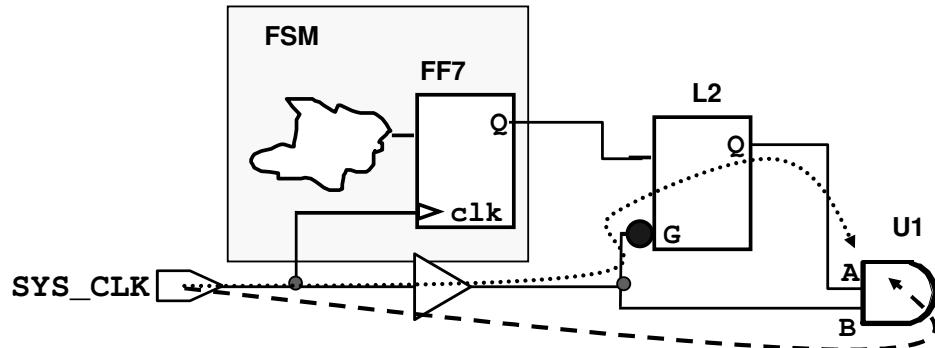
Latches with Zero Time Borrow

- Zero Time Borrow means data arrives before the latch opens.



11-16

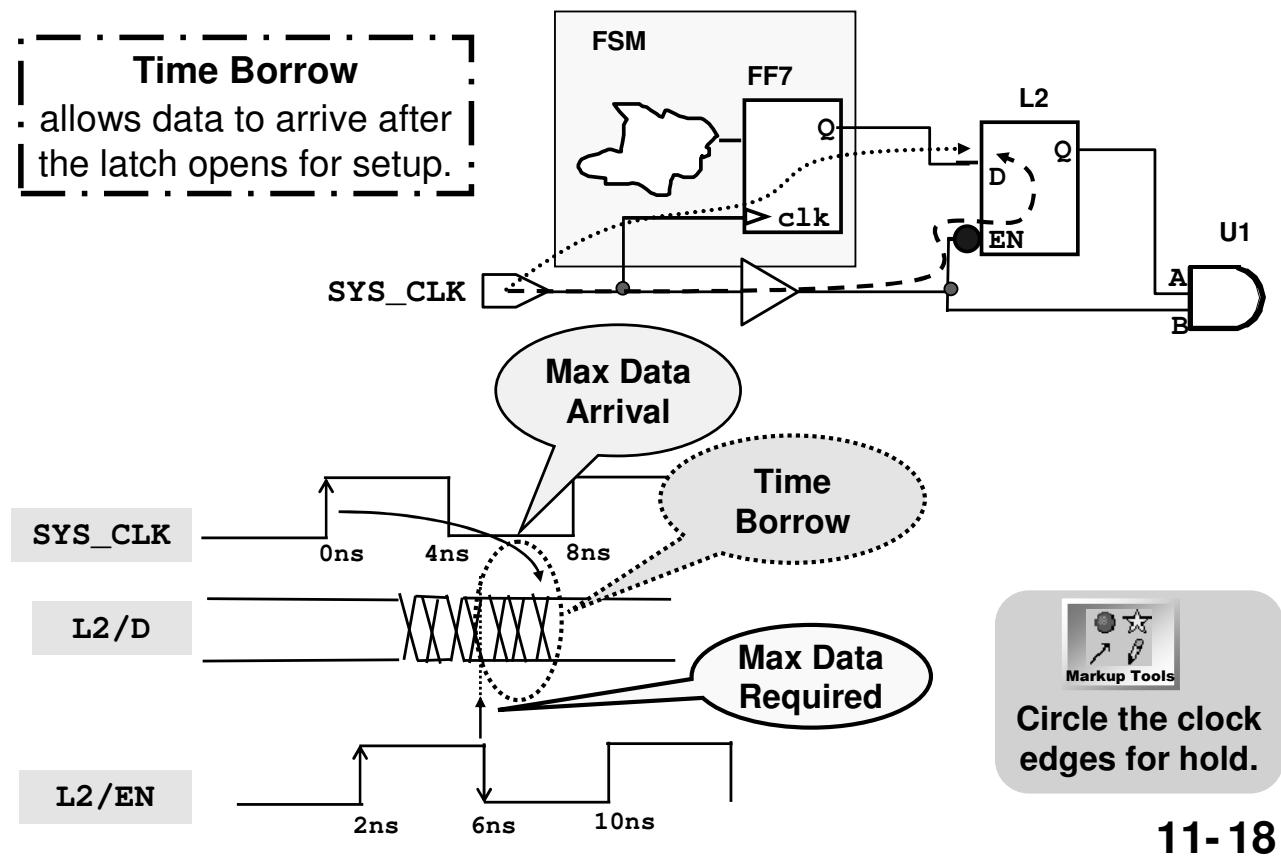
Each Path is Independent



Circle the clock edges for hold.

11-17

Latches with Time Borrow: Setup



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"
```

Max Allowable Time Borrow



One by one, circle:

The time borrowed

Slack

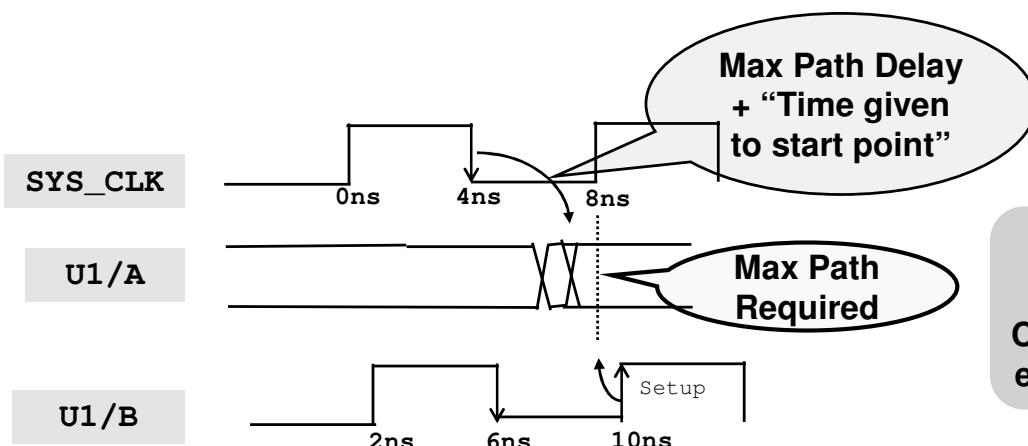
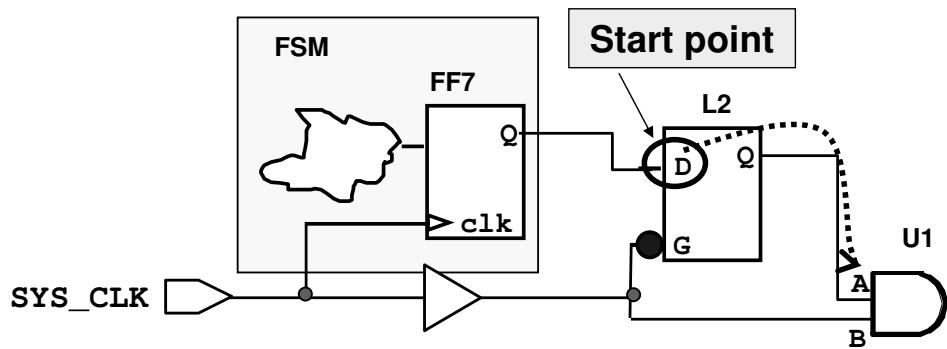
Max allowable time borrow

Setup time of latch

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
L2/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
L2/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	
<hr/>		
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	

11-19

Paths Are No Longer Independent



Circle the clock edges for hold.

11-20

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

```
report_timing -trace_latch_borrow -to U1/A
```

Account For Borrowing in Previous Stage



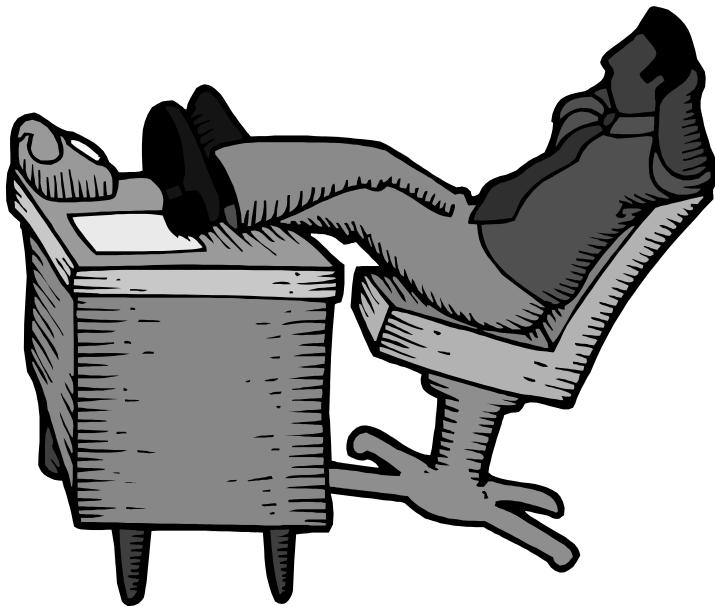
One by one, circle:

The time given to previous stage
Start point of timing path

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 required time		9.89
data arrival time		-7.87
slack (MET)		2.02

11-21

Break



11-22

Multicycle Paths

Min Pulse Width Checks

Recovery and Removal

Latches and Time Borrow

Multicycle Paths

Combinational Feedback Loops

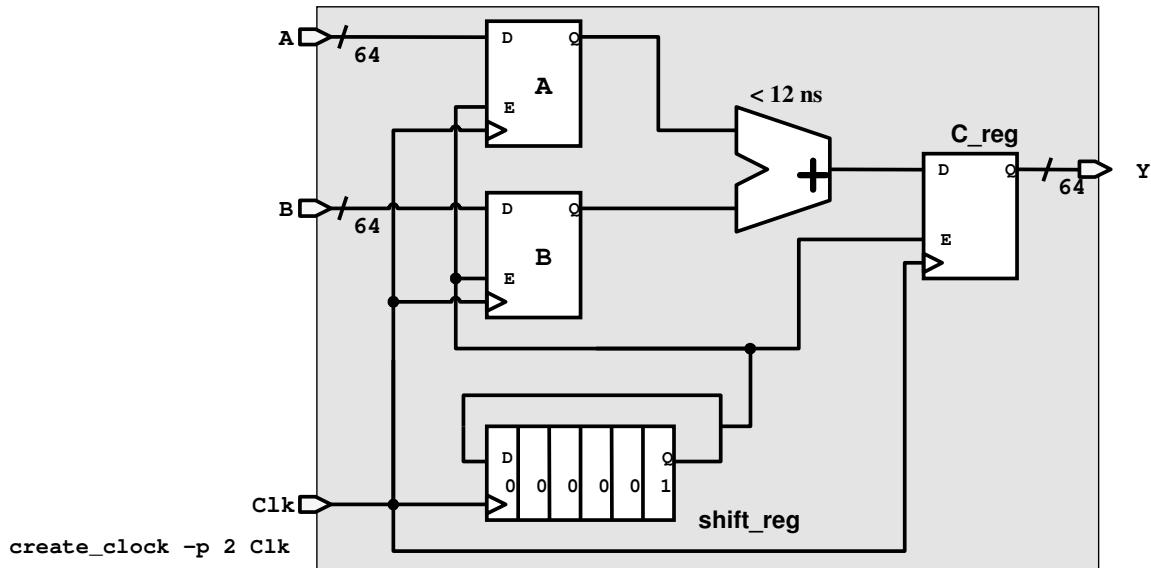
Non-unate Cells in Clock Path

11-23

What Are Multicycle Paths?



Circle the multicycle paths.



PrimeTime automatically identifies multicycle paths!

11-24

By Default, What Does PT Report?

Point	Incr	Path
<hr/>		
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)	2.000	2.000
clock network delay (propagated)	2.838 *	4.838
I_ORCA_TOP/I_RISC_CORE/I_ALU/Zro_Flag_reg/CP (sdcrb1)		4.838 r
library setup time	-0.191 *	4.647
data required time		4.647
data required time		4.647
data arrival time		-6.917
<hr/>		
slack (VIOLATED)		-2.270

11-25

How do you Guide PrimeTime?

```
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
```

11-26

Multicycle Paths



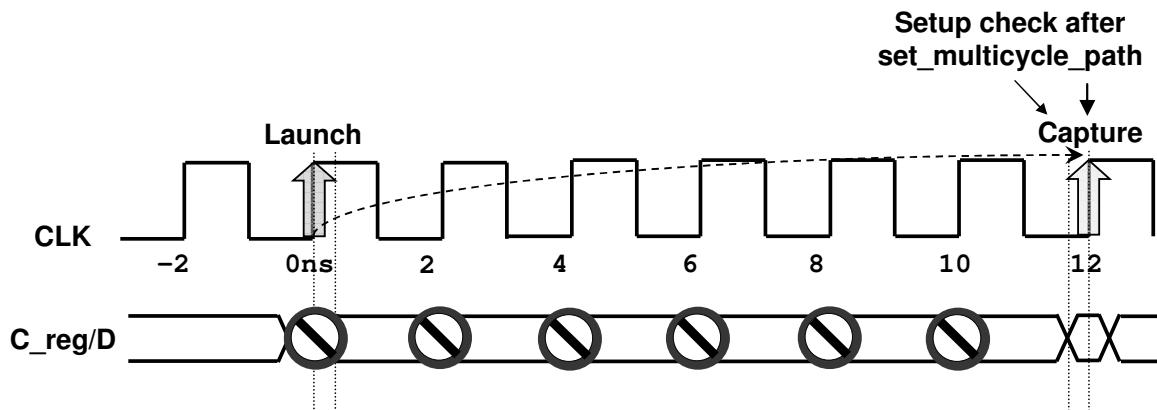
Circle the indication this is a multicycle path.

There is no indication!

Point	Incr	Path
clock SYS_2x_CLK (rise edge)	0.00	0.00
clock network delay (propagated)	2.85 *	2.85
Crnt_Instrn_reg[26]/CP (sdnrb1)	0.00	2.85 r
Crnt_Instrn_reg[26]/Q (sdnrb1)	0.70 *	3.55 f
...		
Zro_Flag_reg/D (sdcrb1)	3.37 *	6.92 r
data arrival time		6.92
clock SYS_2x_CLK (rise edge)	12.00	12.00
clock network delay (propagated)	2.84 *	14.84
Zro_Flag_reg/CP (sdcrb1)		14.84 r
library setup time	-0.19 *	14.65
data required time		14.65
data required time		14.65
data arrival time		-6.92
slack (MET)		7.73

11-27

Where Should the Hold Check Be?



Change could occur near **any** clock edge
causing metastability!



Circle the clock edges for hold
assuming the flip-flops are:

Active on every clock edge.
Active on every sixth clock edge.

11-28

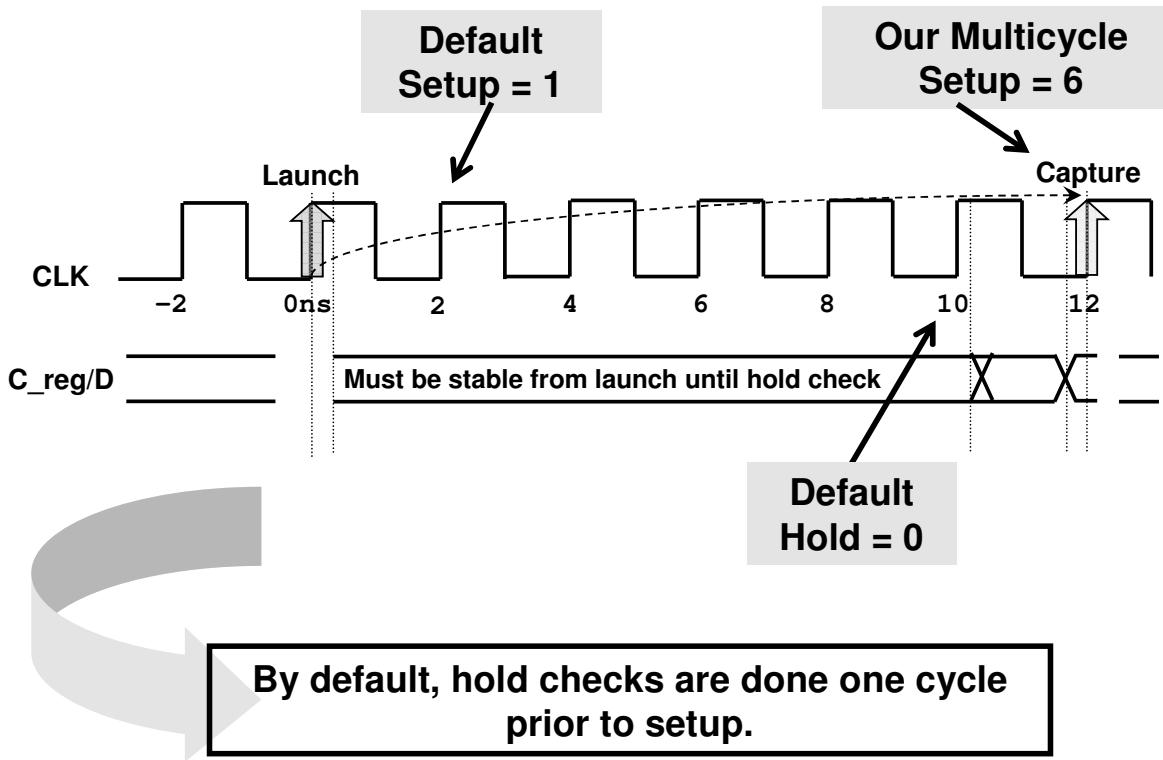
PrimeTime will perform the setup analysis on edge 6, i.e. at 12 ns. This will allow the adder's logic to have a delay of (12 – setup_time – uncertainty).

Specify the above multicycle constraint for setup as follows.

```
set_multicycle_path -setup 6 -to C_reg/D
```

```
pt_shell> set_multicycle_path -help
set_multicycle_path # Define multicycle path
[-setup]           (Only setup multiplier is set)
[-hold]            (Only hold multiplier is set)
[-rise]             (Multiplier valid for rising delays on path endpoint)
[-fall]             (Multiplier valid for falling delays on path endpoint)
[-start]            (Multiplier measured against path startpoint)
[-end]              (Multiplier measured against path endpoint)
[-reset_path]       (Reset this path before setting multicycle)
[-from from_list]   (List of path startpoints or clocks)
[-rise_from rise_from_list]
                   (Apply to paths rising from the list of startpoints or clocks)
[-fall_from fall_from_list]
                   (Apply to paths falling from the list of startpoints or clocks)
[-to to_list]        (List of path endpoints or clocks)
[-through through_list] (List of through pins, cells or nets)
path_multiplier      (Number of cycles)
```

PrimeTime Does Hold Checks Relative to Setup



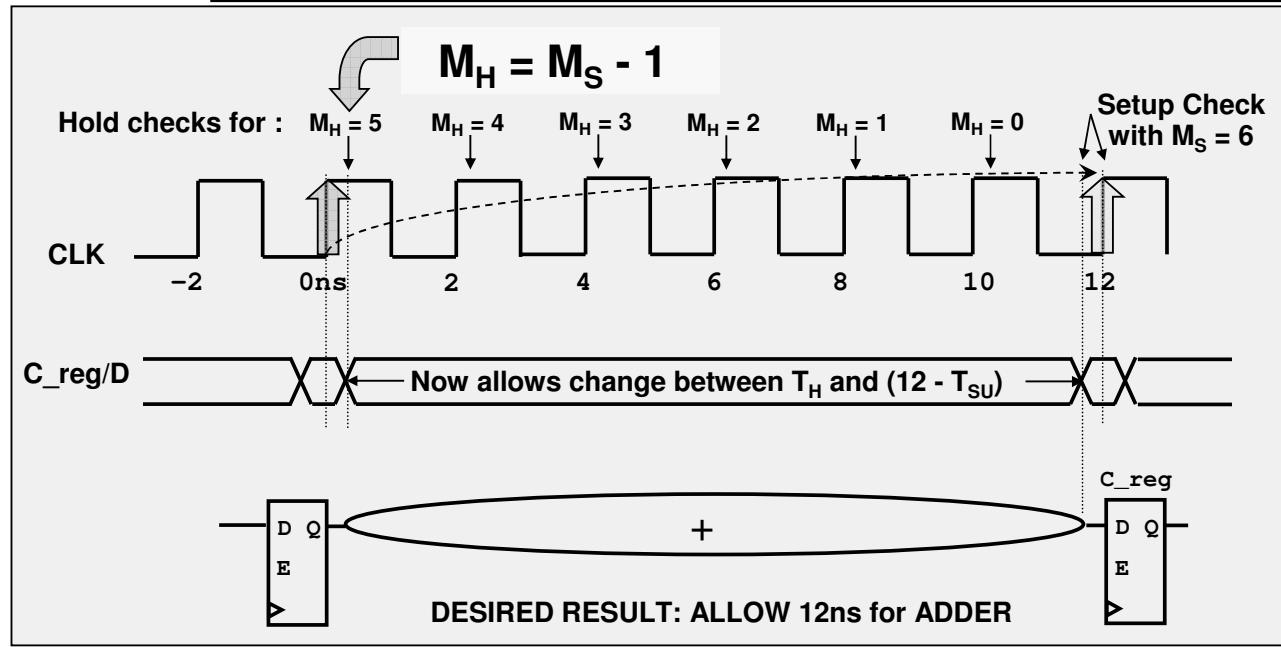
11-29

Set the above constraint for hold time with either of the following:

```
set_multicycle_path -hold 0 -to Creg/D
set_multicycle_path -hold 5 -to Creg/D
```

Set the Proper Hold Constraint

```
set_multicycle_path -setup 6 -to [get_pins "*reg[*]/D"]
set_multicycle_path -hold 5 -to [get_pins "*reg[*]/D"]
```



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.

Validate Hold Multipliers



Circle the setup multiplier.



What is the hold multiplier?



Is this consistent with what we specified?

```
pt_shell> report_exceptions
```

Reasons : f invalid start points
t invalid end points
p non-existent paths
o overridden paths

From	Through	To	Setup	Hold	Ignored
*	*	Zro_Flag_reg/D	cycles=6	cycles=5	

11-31

Ignored Timing Exceptions

Timing exceptions (e.g. multicycle paths) that were applied to the design but ignored by PrimeTime.

```
pt_shell> report_exceptions -ignored
```

Ideally, this report should return nothing!



There are ignored multicycle path exceptions! This indicates:

- There are missing multicycle path constraints.
- There may be missing multicycle path constraints.

11-32

Path-Specific Timing Exceptions

```
Pt_shell> 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 *
Instrn_1_reg[26]/CP (sdnrb1) 0.000 2.850 r
Instrn_1_reg[26]/Q (sdnrb1) 0.699 * 3.549 f
...
clock SYS_2x_CLK (rise edge) 12.000 12.000
clock network delay (propagated) 2.838 *
Zro_Flag_reg/CP (sdcrb1) 14.838 r
...
```

The dominant exceptions are:

From	Through	To	Setup	Hold

reg[26]/CP	*	Zro_Flag_reg/D	cycles=6	cycles=5

The overridden exceptions are:

None

2006.06 feature:
For the reported path, names the exceptions (multicycle path, false path, max/min delay) affecting it.

- When will a dominant false path exception appear in a timing report?
- a. Never
 - b. When you explicitly specify a path that has been ‘false pathed’.

11-33

Arguments to –exceptions are ‘all’, ‘dominant’, and ‘overridden’

Combinational Feedback Loops

Min Pulse Width Checks

Recovery and Removal

Latches and Time Borrow

Multicycle Paths

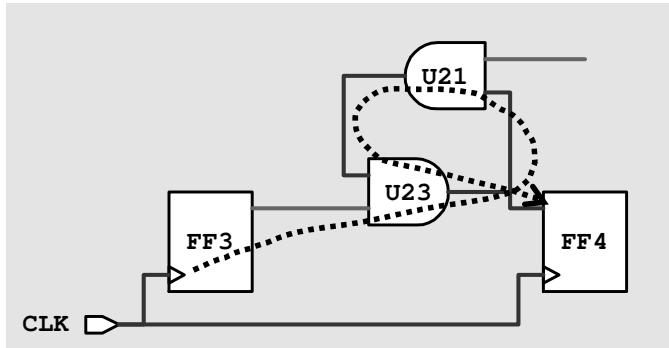
Combinational Feedback Loops

Non-unate Cells in Clock Path

11-34

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?

11-35

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)
```

Will PrimeTime Generate a Warning?

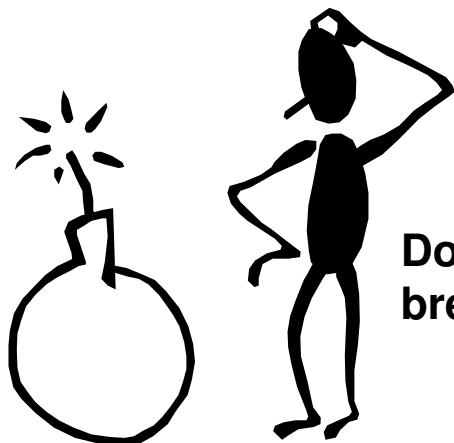
One of the default checks!

```
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
-----
```

11-36

Gotcha's with Combinational Loops



Does the disabled timing arc also break a valid timing path?

A valid timing path has no combinational feedback loop.

If yes – turn on dynamic loop breaking.

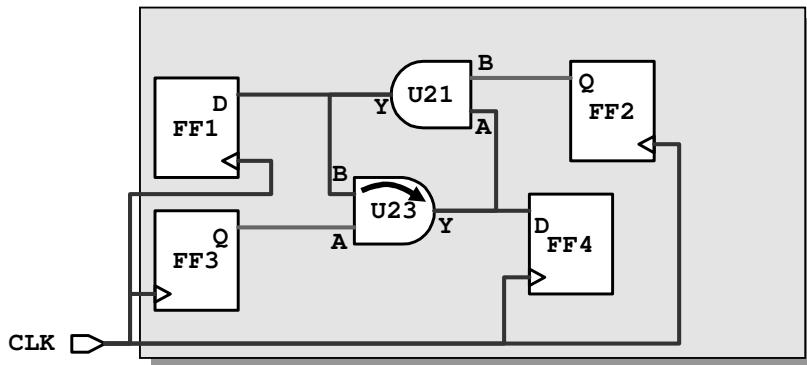
If no - ignore the `check_timing` warning.

Do not turn on dynamic loop breaking because it is not required and this allows for the best PrimeTime runtime.

11-37

Identify the Valid Timing Path

→ Timing arc disabled by PrimeTime to break the combinational loop.



Identify the valid timing path is also broken!

11-38

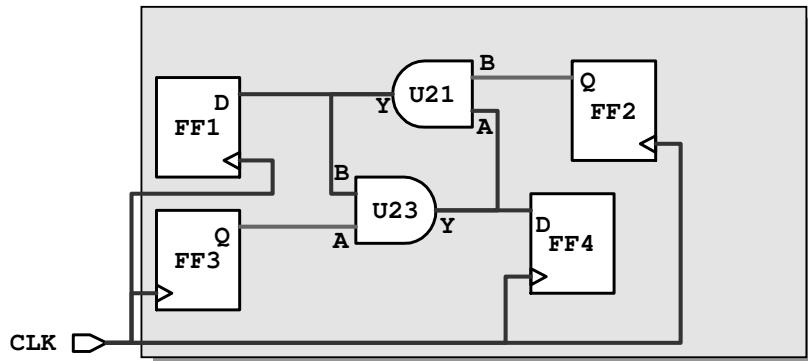
For This Example, Turn On Dynamic Loop

```
set timing_dynamic_loop_breaking true
```



**PrimeTime breaks the combinational loop
without breaking valid timing paths!**

This solution may increase runtime when generating
timing reports.



11-39

```
pt_shell> man timing_dynamic_loop_breaking
```

NAME

```
timing_dynamic_loop_breaking
Enable or disable the dynamic breaking
of combinational feedback loops.
```

DEFAULT

```
false
```

DESCRIPTION

When true, enables dynamic loop breaking. When false (the default), dynamic loop breaking is disabled.

By default, PrimeTime handles loops by identifying the loop and disabling one of the timing arcs of the loop. In some cases, this approach can result in some real paths not being reported in PrimeTime, because they are broken by the disabled arcs used to break loops. Enabling dynamic loop breaking guarantees that no timing arc is disabled to break a loop and that all valid paths of the design are reported.

If the design or the search space for report_timing is large, or if the loops are complex, setting this variable may increase the runtime (or memory) for report_timing significantly.

Loop-breaking Details

```
pt_shell> report_disable_timing
Cell or Port           From      To      Sense      Flag  Reason
-----
U23                   B          Z          *          1

pt_shell> report_timing -through U23/B
No constrained paths.

pt_shell> set timing_dynamic_loop_breaking true

pt_shell> report_timing -through U23/B
Point                  Incr      Path
clock clk (rise edge) 0.00      0.00
clock network delay (ideal) 0.00      0.00
FF2/CP (FD1)           0.00      0.00 r
FF2/Q (FD1)            1.44      1.44 f
U21/Z (AN2)            0.90      2.35 f
U23/B (AN2) <-          0.00      2.35 f
U23/Z (AN2)            0.00      2.35 f
FF4/D (FD1)            0.00      2.35 f
data arrival time       2.35
```

11-40

Non-Unate Clock Path Logic

Min Pulse Width Checks

Recovery and Removal

Latches and Time Borrow

Multicycle Paths

Combinational Feedback Loops

Non-unate Clock Path Logic

11-41

Propagating Both Senses of the Clock

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.



**Explain the solution
PrimeTime will employ.**



**Under what circumstances will
this solution will not work?**

11-42

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'.

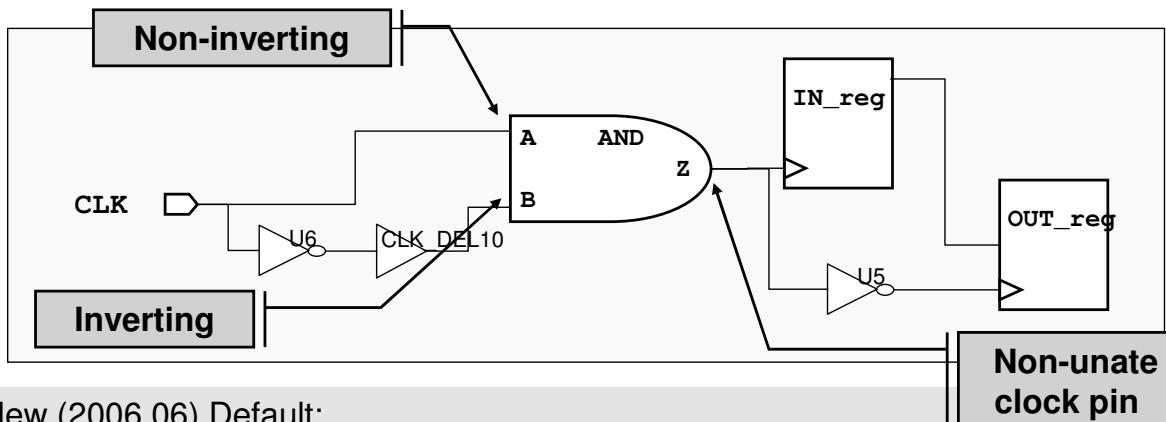
DESCRIPTION

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'.

PTE-070 non-unateness and set_clock_sense



New (2006.06) Default:

```
set timing_non_unate_clock_compatibility false  
PT now propagates BOTH senses of the clock from AND/Z
```

Former behavior was to propagate ONLY positive unateness
PT issues message information message PTE-070

You can restrict the clock sense with:

```
set_clock_sense -positive AND/Z ;# From AND/Z,  
propagate the non-inverting (pin A) path
```

```
set_clock_sense -negative AND/Z ;# From AND/Z,  
propagate the inverting (pin B) path
```

11-43

What is the Next Step?

What next?

Yes – PrimeTime is propagating both edges, and is doing timing checks based on the most pessimistic of these two edges.

No – negative unate or positive unate is the desired sense.

Is a case analysis value missing that will determine the proper sensitivity?

Create a new clock at this pin.

Or

NEW 2006.06 – Primetime propagates both senses unless you tell it which one you want.

`set_clock_sense -positive`
`set_clock_sense -negative`

Actually, the clock is being used as a data path . . .

11-44



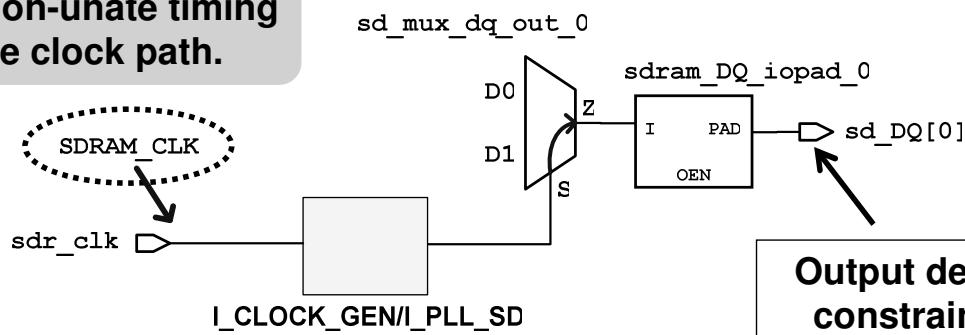
For another reference, take a look at SolvNet article “**What does PTE-012 mean?**”.

Doc Id: 013007 **Last Modified:** 10/01/2004

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



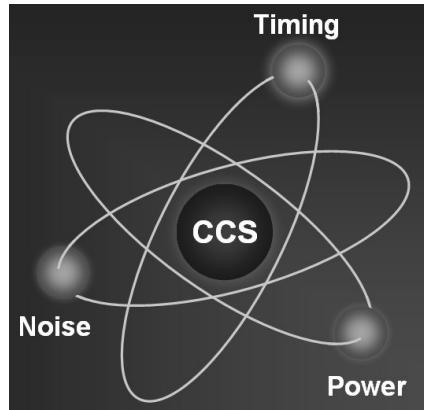
Test this out in lab!

11-45

Libraries for Deep Sub-Micron Design

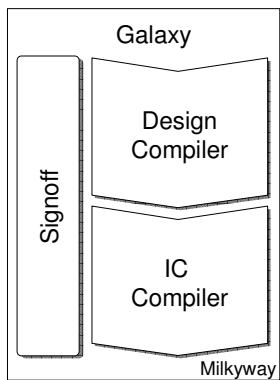
- For 90 nm and smaller, designers are moving to CCS (Composite Current Source) models

- Current-based approach that more accurately models
 - ◆ Timing, Noise, Power
 - High impedance interconnect
 - Miller effect
 - Dynamic IR-drop
 - Multi-voltage
 - Temperature Inversion



11-46

CCS Supported Throughout Galaxy



CCS Support			
Tools	Timing	Noise	Power
NanoChar	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
PrimeTime®	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
IC Compiler	<input checked="" type="checkbox"/>	2007.03	<input checked="" type="checkbox"/>
Design Compiler®	2007.03	-	<input checked="" type="checkbox"/>
PrimeRail	-	-	<input checked="" type="checkbox"/>

= Supported Today

11-47

CCS Availability: Library Vendors

	CCS timin	CCS noise	CCS power	Technology Nodes
Artisan ARM® Physical IP	g <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q1'07	CCS timing: 90nm & below CCS noise: 130nm & below CCS power: 65nm & below
tsmc	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1 st half 2007	CCS timing: 65nm & below CCS noise: 65nm & below CCS power: 65nm & below
UMC	Dec. 2006	Dec. 2006	1 st half 2007	CCS timing: 65nm & below CCS noise: 65nm & below CCS power: 65nm & below
VIRAGE LOGIC Accelerating Silicon Success	Q1'07	Q1'0 7	1 st half 2007	CCS timing: 65nm & below CCS noise: 90nm & below CCS power: 65nm & below



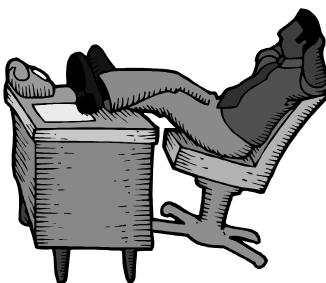
= Supported Today

11-48

Lab 11: Additional Checks and Constraints



30 minutes



10 Minute BREAK
30 Minute LAB

The objective is to:

- **Apply user specified annotated delays to explore time borrowing with latches**
- **Debug PTE-070 messages**

11-49

Lab Wrap Up: Warnings in ORCA

The message ID	A short description	Which command caused the warning
✓ PTE-003	There are disabled timing arcs in this design.	The command that initiates a timing update
✓ PTE-070	There is non-unate logic in the clock path.	"
✓ PTE-060	There are cells in the clock path for which PrimeTime could not infer clock gating checks.	"

11-50

Lab Wrap Up: Informational Messages

The message ID	A short description
✓ CMD-029	The unalias command tried to remove an alias that does not exist.
✓ ENV-003	Using automatic wire load models, this is unimportant in a full SDF flow.
✓ PTE-016	A clock's base period has been expanded.
✓ PTE-017	PrimeTime is inferring clock gating checks in the design.
✓ PTE-018	PrimeTime has abandoned fast timing updates.
✓ PTSR-002	Removing an existing directory before saving a new session. This message will only occur if you execute the run script more than once.

11-51

This page was intentionally left blank.

Agenda

**DAY
3**

10 Analysis Type and Back Annotation



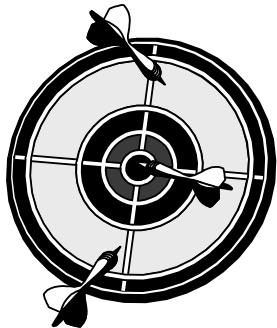
11 Additional Checks and Constraints



12 Conclusion

CS Customer Support

Workshop Goal



**Perform STA using PrimeTime.
Generate and interpret timing reports.
Create a clean, optimized run script.
Debug your timing violations.**

12-2

After Units 1-6, You Can Now

- Identify problems by interpreting `report_timing` reports
- Debug and explore further by controlling the output of `report_timing` reports



After Units 7-9, You Can Now

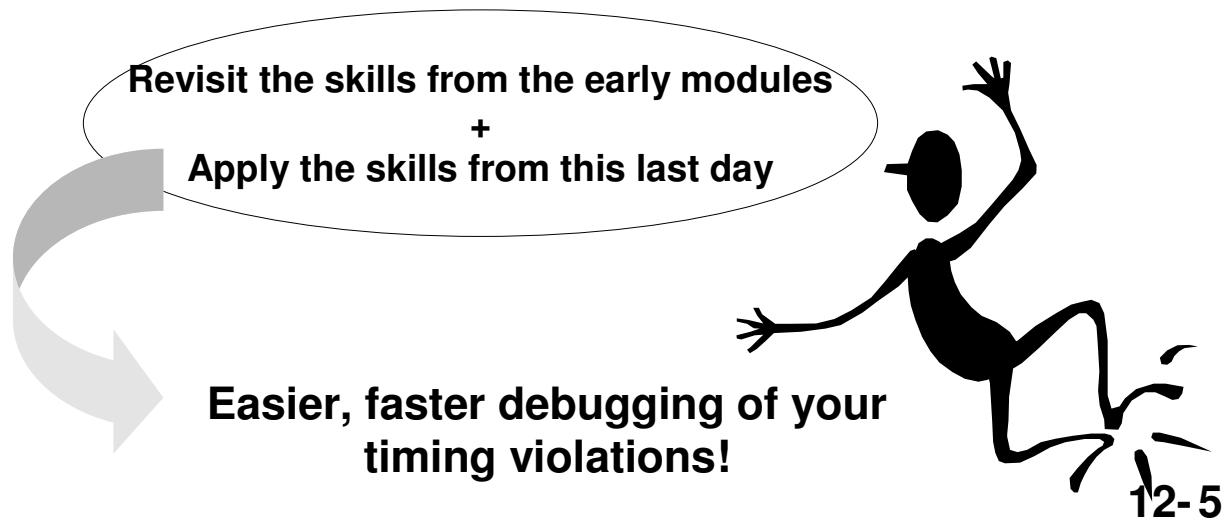
- Setup your PrimeTime environment with useful Tcl procedures and aliases
- Using a straight forward and simple methodology, create and debug a run script
- Create runtime metrics, the key to improving PrimeTime runtime
- Gather information about clocks in an unfamiliar design
- Use two essential sources of information - SolvNet and SNUG



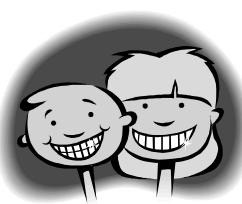
12-4

After Units 10-11, You Can Now

- Apply back annotation and on chip variation
- Apply the min and max SDF delays correctly
- Address additional constraints and timing checks



Share A Work Application



Please share an application at work
where you can apply one of the
skills learned today.

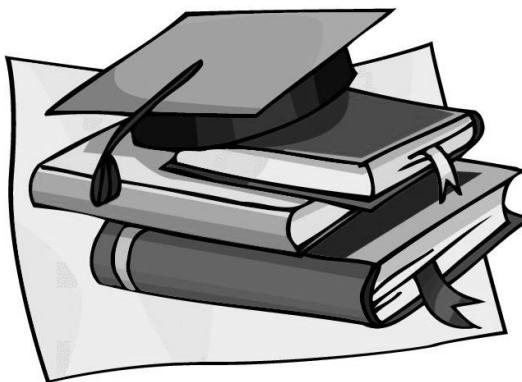
12-6

Course Evaluation

- **Synopsys needs your feedback to improve our courses**
- **You will receive an email in a couple of days, please follow the instructions to fill in the course evaluation!**

12-7

Thank You!



12-8

Appendix

Download Workshop Labs

How to Download Lab Files (1/4)

From the Synopsys home page select “Education & Support”

Then Select
Customer Education



12-10

How to Download Lab Files (2/4)

Then Select
Download Labs

Synopsys® Education & Support

Customer Education Services - North America

Synopsys Customer Education Services offers you the training that will help you maximize your investment in Synopsys tools. Our courses range from introductory level, which focus on tool operations, to advanced design methodology that includes lectures by Synopsys tool experts combined with hands-on labs to enhance your understanding of key concepts. Courses cover all aspects of high-level design including verification, synthesis, physical design, system design, IP creation and test.

Synopsys training options include: Public Courses, Private Courses, Online Learning, Virtual Classroom, live online training and Engineer Re-Employment Program (requires pre-qualification). You can find detailed information on the [Course Options](#) page.

Find a Course

Jump To...
Title Contains:
Location:
Limit the Date Range: From Month Year To Month Year
Search

Highlights

- NEW Virtual Classroom
- Special Pricing: SystemVerilog with VCS for Verilog Users

Resources

- [Download Lab Materials](#)
SolNet ID required
- [Receive the North America Course Summary via email](#)

Regional Training Centers Course Calendar

Done

12-11

How to Download Lab Files (3/4)

After logging in through SolvNet, you will see:

The screenshot shows a Microsoft Internet Explorer window with the title "Electronic Software Transfer (EST) for Lab Files - Microsoft Internet Explorer". The address bar contains the URL <https://solvnet.synopsys.com/retrieve/002471.html>. The page header includes the Synopsys logo and navigation links like "solvnet home | synopsys.com", "MY PROFILE | PREFERENCES | CONTENT | LAYOUT | FEEDBACK | HELP | LOGOUT". Below the header, there is a document summary:
Doc Id: 002471 Product: Not Product Specific
Last Modified: 06/15/2004 [Print-Friendly Version](#) [Email Article](#)
Rate this article: Not Useful Somewhat Useful Useful Very Useful A large arrow on the right side of the page points downwards, with the text "Scroll down to your course" written vertically next to it. At the bottom right of the page area, the number "12-12" is displayed.

Workshop Description	Software Version	FTP Product Directory
Advanced Chip Synthesis	2001.08	Labs_ADVCHIP Download
Advanced Verilog	6.0	Labs_ADVVER Download
		Labs ADVVHDL

12-12

How to Download Lab Files (4/4)

Locate Course Title

Click to Download Labs

README file will walk you through decompression and installation steps.

Course Title	Version	Download Links
Physical Compiler	2004.12-SP5	Labs_PC1_2004.12-SP5 Download
Power Compiler	2004.06	Labs_PWC_2004.06 Download
Power of Tcl 1: Becoming a Proficient Tcl User	2004.12	Labs_TCL1_2004.12 Download
Power of Tcl 2: Creating High-Impact Procedures	2004.12	Labs_TCL2_2004.12 Download
Power of Tcl 3: Direct Access through Collections and Attributes	2004.12	
PrimePower	2004.06	Labs_PP_2004.06
PrimeTime 1	2006.06	
PrimeTime: Advanced Static Timing Analysis Constraint Debugging	2004.12	
PrimeTime: Signal Integrity	2004.12	

12-13

This page was intentionally left blank.



Customer Support

© 2006 Synopsys, Inc. All Rights Reserved

20050516

Synopsys Support Resources

1. Build Your Expertise: Customer Education Services

- www.synopsys.com
 - ◆ Workshop schedule and registration
 - ◆ Download materials (*SolvNet id required*)

2. Empower Yourself: solvnet.synopsys.com

- Online technical information and access to support resources
- Documentation & Media

3. Access Synopsys Experts: Support Center



The screenshot shows the Synopsys Education & Support website. At the top, there's a navigation bar with links for 'PRODUCTS & SOLUTIONS', 'PROFESSIONAL SERVICES', 'EDUCATION & SUPPORT', 'CORPORATE', and 'PARTNERS'. Below the navigation, there's a banner featuring a person working at a computer with the text 'Synopsys Launches Virtual Classroom'. The main content area is divided into three columns: 'Customer Education' (with links to SolvNet Online Support, Worldwide Support Centers, and SNUG), 'SolvNet' (with links to SolvNet Home, Documentation & Media, Software & Installation, and Support Center Locator), and 'Support Center' (with links to Enter A Call to Support, Worldwide Support Centers, and Platforms & Releases). On the left, there's a sidebar with an 'IMPORTANT' section titled 'Information about Linux Migration'.

CS-2

SolvNet Online Support Offers

- Immediate access to the latest technical information
- Thousands of expert-authored articles, Q&As, scripts and tool tips
- Enter-a-call online Support Center access
- Release information
- Online documentation
- License keys
- Electronic software downloads
- Synopsys announcements (latest tool, event and product information)

The screenshot shows the SolvNet Online Support website. At the top, there's a navigation bar with links for 'MY PROFILE', 'PREFERENCES', 'CONTENT', 'LAYOUT', 'FEEDBACK', 'HELP', and 'LOGOUT'. Below the navigation is a search bar with a 'search' button. The main content area has several sections:

- Main Navigation:** Includes links for Home, Enter A Call - Tool Support, Support Case Status, ViewConnect, Software and Installation (Release Library, SmartKeys, Installation Guide, Licensing QuickStart, Electronic Software Transfer), Documentation and Media (Documentation on the Web, MediaDocs, Release Notes), and SNUG - Synopsys Users Group.
- Announcements:** Features a link to 'Synopsys Linux Support Migrating to RHEL 3.0'.
- Support Resources:** Includes links for 'Enter A Call to the Support Center', 'View and Update Cases', and 'View STARs (Defects)'.
- Featured Article:** Headed 'Q&A on Synopsys Migration to Red Hat Enterprise Linux 3', it discusses the migration from Red Hat Linux 7.2 to Red Hat Enterprise Linux version 3, providing answers to important questions.
- New & Updated Articles:** A list of recent articles including:
 - TetraMax Diagnosis FAQ (11-25-2004) Updated!
 - Conditional statements during scan shifting (11-24-2004)
 - How to create a timing model from an empty Verilog module (11-24-2004) Updated!
 - Why am I getting a NOISE-001 warning on my noise-characterized Library? (11-24-2004)
 - Do I Need to Back-Annotate Loads in Primetime if I Am Using Standard Delay Format Files? (11-24-2004) Updated!
 - Studio fails to load lacking libdb.so.3.0 on RHEL 3.0 WS (11-24-2004)
- Bulletin:** Includes a link to 'SI Closure white paper Professional Services'.
- IMPORTANT:** Includes a link to 'Information about Linux Migration'.

CS-3

SolvNet Registration is Easy

- 1. Go to solvnet.synopsys.com/ProcessRegistration**
- 2. Pick a username and password.**
- 3. You will need your “Site ID” on the following page.**
- 4. Authorization typically takes just a few minutes.**

New User Registration

Your Corporate Email

Select a Username (minimum 4 characters; a-z(lowercase only), 0-9)

Select a Password (minimum 4 characters; a-z, 0-9)

Re-enter Password

I am 18 or older.

By completing the registration fields and clicking on the "Submit" button, you are agreeing to the terms of the [Privacy Policy](#). If you have any questions, please contact [support@synopsys.com](#).

New User Registration

Important: Please Read Before Registering

To access to all Synopsys Online Services, you **must** provide an Active Site ID in the field below. These services include:

- * SolvNet Knowledge Base of self-help documents
- * Online Product Documentation
- * SmartKey License Retrieval
- * Electronic Software Downloads
- * ViewSupport and Support Case Tracker - view status of open support cases

Synopsys Site ID Add Another Site

Next

[Registration Help](#)

CS-4

Support Center: AE-based Support

- **Industry seasoned Application Engineers:**
 - 50% of the support staff has > 5 years applied experience
 - Many tool specialist AEs with > 12 years industry experience
 - Access to internal support resources
- **Great wealth of applied knowledge:**
 - Service >2000 issues per month
- **Remote access and debug via ViewConnect**
- **Contact us:**
 - Web: *Enter A Call* from solvnet.synopsys.com
 - See <http://www.synopsys.com/support>
for local support resources

Fastest access

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):**

- www.snug-universal.org

CS-6

Summary: Getting Support

- Customer Education Services
- SolvNet
- Support Center
- SNUG

CS-7

This page was intentionally left blank.