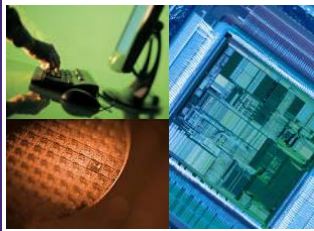


Static Timing Analysis with PrimeTime & PT-SI

Lecture 9



SYNOPSYS
Predictable Success

Agenda

- *Introduction*
- Basic Flow
- Advanced Analysis
- Graphical User Interface (GUI)
- Back Annotation
- Modeling
- Usability
- Signal Integrity Analysis (PT-SI)
- Summary

What is STA?

- *Static Timing Analysis*

- (STA) is a method of determining if a circuit meets timing constraints without simulating clock cycles

- How it works:

- Identify timing startpoints and endpoints
 - Input/output ports, registers/latches
- Trace delays through timing paths from startpoints to endpoints
- Compare path delays to clock period to see if constraints are met
- Also check hold violations (races) and transition violations (from library design rules)

- STA is:

- Exhaustive
- Fast
- Not dependent on simulation vectors

© 2007 Synopsys, Inc. (3) CONFIDENTIAL

SYNOPSYS
Predictable Success

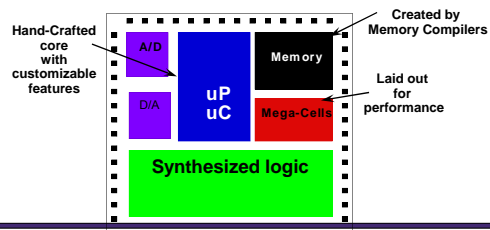
What is PrimeTime?

A Full Chip Gate-Level Static Timing Analyzer

- Static Timing Sign-Off Tool
- Fast and memory efficient
- Delay calculator
- Custom block modeling solution
- Advanced analysis functionality
- 350+ Vendor libraries, and timing consistency with Synthesis

Ease of Adoption

- Same design database as Design Compiler (DC)
- Same libraries as DC
- Same commands as DC



© 2007 Synopsys, Inc. (4) CONFIDENTIAL

SYNOPSYS
Predictable Success

What is PT-SI ?

- PrimeTime - Signal Integrity (PT-SI)
 - Extends PT's timing analysis to include signal integrity effects, including:
 - Crosstalk delay
 - Noise bumps (glitches)
 - IR drop
 - SI effects analyzed in regular PrimeTime environment and included in standard timing reports
- Benefits
 - Fast, accurate analysis of multi-million-gate designs
 - Much faster than traditional SPICE analysis
 - Can perform simultaneous Min and Max analysis

© 2007 Synopsys, Inc. (5) CONFIDENTIAL

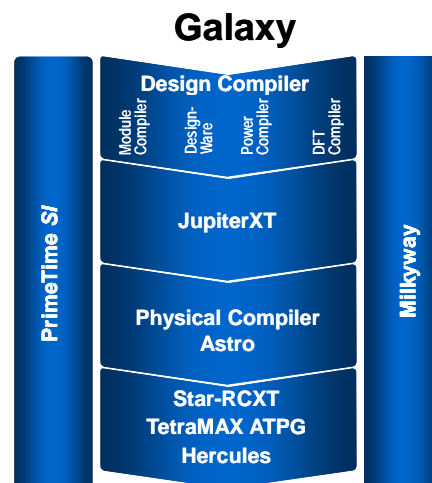
SYNOPSYS
Predictable Success

Synopsys' Galaxy™ Design Platform

Industry-Standard STA & SI Sign-Off

PrimeTime & PrimeTime SI

- Golden static timing & signal integrity standard for Galaxy
- Accurate timing and SI analysis throughout implementation flow
- Industry standard timing and SI signoff



© 2007 Synopsys, Inc. (6) CONFIDENTIAL

SYNOPSYS
Predictable Success

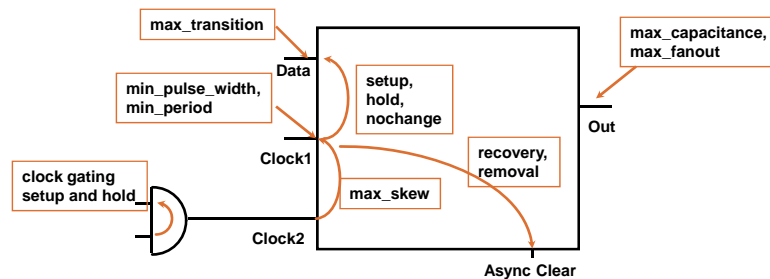
Agenda

- Introduction
- **Basic Flow**
- Advanced Analysis
- Graphical User Interface (GUI)
- Back Annotation
- Modeling
- Usability
- Signal Integrity Analysis (PT-SI)
- Summary

© 2007 Synopsys, Inc. (7) CONFIDENTIAL

SYNOPSYS
Predictable Success

Comprehensive Timing Checks



- Many types of timing and design rule checks
 - Timing checks can be delay calculated or SDF annotated
 - Setup and hold can depend on $Slew_{data}$, $Slew_{clock}$, and Cap_Q
 - Design rule checks consistent with DC, PC, and Astro

© 2007 Synopsys, Inc. (8) CONFIDENTIAL

SYNOPSYS
Predictable Success

Basic Timing Analysis Flow Using PrimeTime

Set Up Design Environment, Read & Link Design

- Search path, link path
- Read designs, libraries, then link
- Set operating conditions, wireload models, input drives & output loads

Specify Timing Constraints

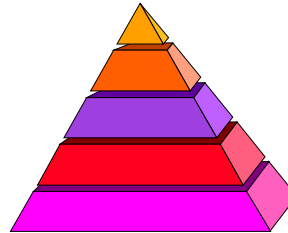
- Clock period/waveform/uncertainty/latency
- Input/output delays

Specify Timing Exceptions

- Multicycle paths
- False paths
- Min/max delays, segmentation, disabled arcs

Perform Timing Analysis, Create Reports

- Check timing constraints
- Generate timing and constraint reports
- Generate bottleneck and coverage analysis reports

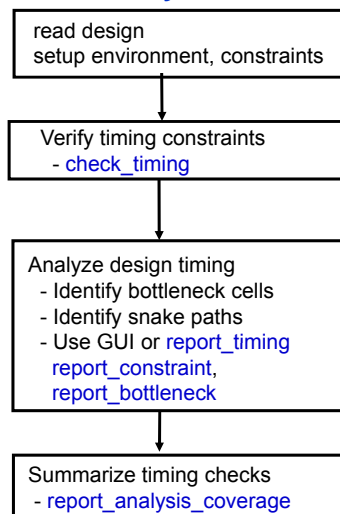


© 2007 Synopsys, Inc. (9) CONFIDENTIAL

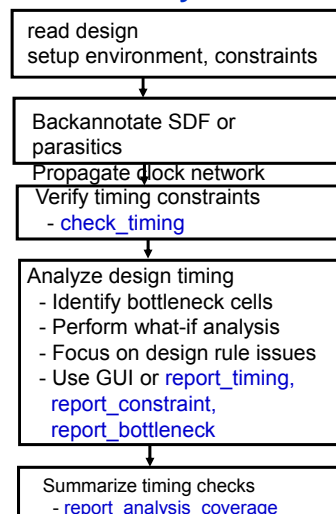
SYNOPSYS
Predictable Success

Pre- and Post-Layout Timing Flows

Pre-layout



Post-layout



© 2007 Synopsys, Inc. (10) CONFIDENTIAL

SYNOPSYS
Predictable Success

Set Up Environment, Read & Link Design

- **Definitions**

- search_path variable specifies where to search for design and library data
- link_path variable specifies which design and library data to be loaded during linking (link_design)
- link_design command resolves all design references

- **Example**

```
pt_shell> set search_path ". ./lib"
pt_shell> set link_path "* pt_lib.db"
pt_shell> read_ddc design.ddc
pt_shell> link_design
```



Note: Designs can be read in DDC, Verilog, VHDL, Milkyway, or .db formats (read_ddc, read_verilog, read_vhdl, read_milkyway, read_db)

Important : Linking a new design causes all constraints and back annotated data to be removed from the currently linked design, and removes all but the top level design from memory

© 2007 Synopsys, Inc. (11) CONFIDENTIAL

SYNOPSYS
Predictable Success

Set Operating Conditions, Wireload Models, Input Drives & Output Loads

- Set the operating conditions
- Set wireload models and mode (pre-layout only)
- Specify input drives and output loads

- **Example:**

```
pt_shell> set_operating_conditions -library pt_lib \
          -analysis_type on_chip_variation WCCOM
pt_shell> set_wire_load_mode top
pt_shell> set_wire_load_model -library pt_lib -name 05x05 -min
pt_shell> set_wire_load_model -library pt_lib -name 20x20 -max
pt_shell> set_driving_cell -lib_cell OR2 [all_inputs]
pt_shell> set_driving_cell -lib_cell IV9 [get_ports {asy_* rstN*}]
pt_shell> set_load 0.5 [all_outputs]
pt_shell> set_load 0.1 [remove_from_collection [all_inputs] CLK]
```

© 2007 Synopsys, Inc. (12) CONFIDENTIAL

SYNOPSYS
Predictable Success

Specify Timing Constraints: Clocks (1)

- **Example:**

- » Set up the basic timing constraints for the design. Start with the clock information.

```
pt_shell> create_clock -name CLK -period 30 [get_port CLOCK]
pt_shell> set_clock_uncertainty 0.5 [all_clocks]
pt_shell> set_clock_latency -min 3.5 [get_clocks CLK]
pt_shell> set_clock_latency -rise 5.5 [get_clocks CLK]
pt_shell> set_clock_transition -rise 0.25 [get_clocks CLK]
pt_shell> set_clock_transition -fall 0.3 [get_clocks CLK]
```

- For post layout clock tree :

```
set_propagated_clock <clock_object_list>
```

or

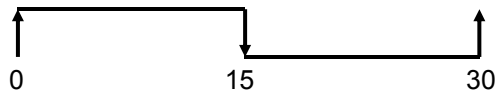
```
set timing_all_clocks_propagated true
```

© 2007 Synopsys, Inc. (13) CONFIDENTIAL

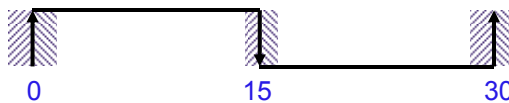
SYNOPSYS
Predictable Success

Specify Timing Constraints: Clocks (2)

Reference clock waveform



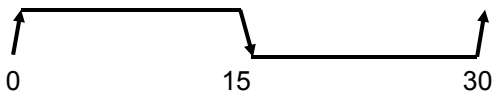
Reference clock waveform with uncertainty



Reference clock waveform with latency



Reference clock waveform with transition



Reference clock waveform with uncertainty, latency, and transition



© 2007 Synopsys, Inc. (14) CONFIDENTIAL

SYNOPSYS
Predictable Success

Specify Timing Constraints: Clocks (3) Modeling Off-chip Latency

- You can specify external latency:

```
pt_shell> set_clock_latency -source 1.7 CLOCK
```

The clock is delayed by 1.7 units before it gets to the clock definition point

- You can model clock jitter*:

```
pt_shell> set_clock_latency -source -rise -early 0.1 GENCLK
pt_shell> set_clock_latency -source -fall -early 0.1 GENCLK
pt_shell> set_clock_latency -source -rise -late 0.2 GENCLK
pt_shell> set_clock_latency -source -fall -late 0.3 GENCLK
```



* Note: Symmetrical clock jitter can also be specified with set_clock_uncertainty

© 2007 Synopsys, Inc. (15) CONFIDENTIAL

SYNOPSYS
Predictable Success

Specify Timing Constraints: Input & Output Delays

- Specify signal arrival/required times at **all** ports relative to clocks with:

```
set_input_delay
set_output_delay
```

Examples:

```
pt_shell> set_input_delay 5.0 -clock ClkA [all_inputs]
pt_shell> set_input_delay 2.5 -clock ClkB [get_ports input2]
pt_shell> set_input_delay 0.5 -clock ClkA -add_delay input2
pt_shell> set_output_delay 3.0 -clock ClkA [all_outputs]
```

Note: For bidirectional ports, use both `set_input_delay` & `set_output_delay`.

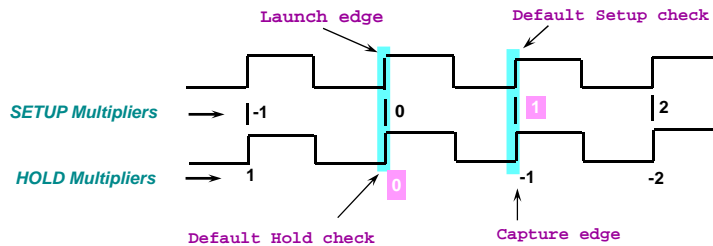
© 2007 Synopsys, Inc. (16) CONFIDENTIAL

SYNOPSYS
Predictable Success

Setup and Hold Multipliers (1)

Setup multiplier represents the maximum number of clock cycles allowed for a signal to traverse a given path.

Setup multiplier = 1 is the default (one cycle after launch)
Hold multiplier = 0 is the default (one cycle before capture edge)



• Note moving setup check to clock edge x the hold will be checked at x-1 clock edge.

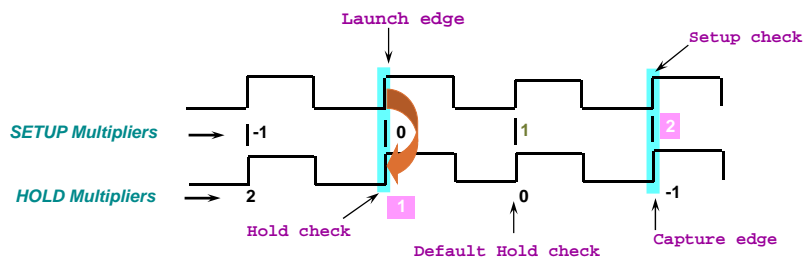
© 2007 Synopsys, Inc. (17) CONFIDENTIAL

SYNOPSYS
Predictable Success

Setup and Hold Multipliers (2)

A positive hold multiplier represents the number of clock cycles before the default hold check.

```
set_multicycle_path -setup <setup multiplier> -hold <hold multiplier>
```



```
set_multicycle_path -setup 2 -hold 1 -from <from_list> -to <to_list>
```

Setup multiplier = 2 (two cycles after launch)
Hold multiplier = 1 (two cycles before capture edge)

© 2007 Synopsys, Inc. (18) CONFIDENTIAL

SYNOPSYS
Predictable Success

Specify Timing Exceptions

What Are They?

- Exceptions to default single-cycle timing
 - False paths
 - Multi-cycle Paths
 - User-defined max_delay, min_delay
- Exceptions can have from/through/to objects
- -through exceptions allow ANDing and ORing of exceptions
 - `set_false_path -through {A B C}`
Path must travel through points **A or B or C** to be considered false
 - `set_multicycle_path 2 -through {A B} -through {C}`
Path must travel through points **(A or B) and C** to be a multi-cycle path

© 2007 Synopsys, Inc. (19) CONFIDENTIAL

SYNOPSYS
Predictable Success

Specify Timing Exceptions

Avoid Overuse of Wildcards!

- Large numbers of timing exceptions increases memory usage and timing analysis runtimes **SUBSTANTIALLY**.
- Example:
 - » Set false paths between 32-bit registers Reg_A and Reg_B in asynchronous clock domains clocked by clkA and clkB

```
pt_shell> set_false_path -from Reg_A*/Q -to Reg_B*/D
pt_shell> set_false_path -from Reg_B*/Q -to Reg_A*/D
```

This creates and stores **2*32*32 = 2048 false path** exceptions! (BAD)

or

```
pt_shell> set_false_path -from clkA -to clkB
pt_shell> set_false_path -from clkB -to clkA
```

This creates and stores **2 false path** exceptions! (GOOD)



SolvNet Doc Id: 902120

PrimeTime Memory Issues With Timing Exceptions

© 2007 Synopsys, Inc. (20) CONFIDENTIAL

SYNOPSYS
Predictable Success

Specify Timing Exceptions Single -through Pin

- What looks like a single exception through a pin is usually many exceptions from the start-points that fan-in to the pin to the end-points that fan-out from the pin.

Bad:

```
pt_shell> set_false_path -through [get_pins ADDER/CI]
```

- Use `set_disable_timing` to disable ALL timing paths through the pin. Disabling the paths this way means that PrimeTime does NOT have to store information about the multiple exceptions.

Good:

```
pt_shell> set_disable_timing [get_pins ADDER/CI]
```

© 2007 Synopsys, Inc. (21) CONFIDENTIAL

SYNOPSYS
Predictable Success

Exception Analysis & Compaction

- `report_exceptions -ignored`
- `transform_exceptions`
- Removes ignored timing exceptions from memory
- Useful with legacy constraints
- Options: `-from`, `-to`, `-rise_from`, `-rise_to`, `-fall_from`, `-fall_to`, `-rise_through`, `-fall_through`, `-dry_run`, `-verbose`, `-flatten`

```
pt_shell> report_exceptions -ignored
pt_shell> transform_exceptions
pt_shell> write_sdc -out clean_constraints
pt_shell> report_exceptions -ignored
```









© 2007 Synopsys, Inc. (22) CONFIDENTIAL

SYNOPSYS
Predictable Success

Perform Timing Analysis

Check Constraints

- To identify problems with design or assertions before you spend time on detailed reports
 - » Use `check_timing [-verbose]`
 - » Review man page for list of default checks
 - » `timing_check_defaults` list of timing checks performed by `check_timing`
- (Not all available checks are run by default)

 Unconstrained endpoints?
 Combinational loops?
 Missing clock definitions?
 Multiple clock fanin?
 Latch fanout problems?
 Generated clocks consistent?
 Ignored timing exceptions?
 Ports with missing input delay?
 ...

© 2007 Synopsys, Inc. (23) CONFIDENTIAL

SYNOPSYS
Predictable Success

Perform Timing Analysis

Report Coverage

- To summarize timing checks met, violated, untested:
 - Determine if your analysis is complete
 - Show untested checks and reason why not tested
 - Use `report_analysis_coverage -status_details {untested}`

```
pt_shell> report_analysis_coverage
```

Type of Check	Total	Met	Violated	Untested
-----	-----	-----	-----	-----
setup	5	2 (40%)	2 (40%)	1 (20%)
hold	5	4 (80%)	0 (0%)	1 (20%)
recovery	2	0 (0%)	2 (100%)	0 (0%)
min_period	1	1 (100%)	0 (0%)	0 (0%)
min_pulse_width	2	2 (100%)	0 (0%)	0 (0%)
-----	-----	-----	-----	-----
All Checks	15	9 (60%)	4 (27%)	2 (13%)
...				

© 2007 Synopsys, Inc. (24) CONFIDENTIAL

SYNOPSYS
Predictable Success

Perform Timing Analysis Generate Reports

- **Summarize timing results:**
 - » Use **report_constraint** <options>
 - Note: -all_violators -verbose option may cause long run times. Use -max_delay or -min_delay to focus on setup or hold violations.
- **Display worst violations:**
 - » Use **report_timing** <options>
 - Note: large values for -nworst or -max_paths option may cause long run times.
 - Unconstrained paths are not listed by default
timing_report_unconstrained_paths = "false"
- **Report bottleneck cells contributing to multiple violations:**
 - » Use **report_bottleneck** <options>

© 2007 Synopsys, Inc. (25) CONFIDENTIAL

SYNOPSYS
Predictable Success

report_timing -exclude *Improves Flexibility of Analysis*

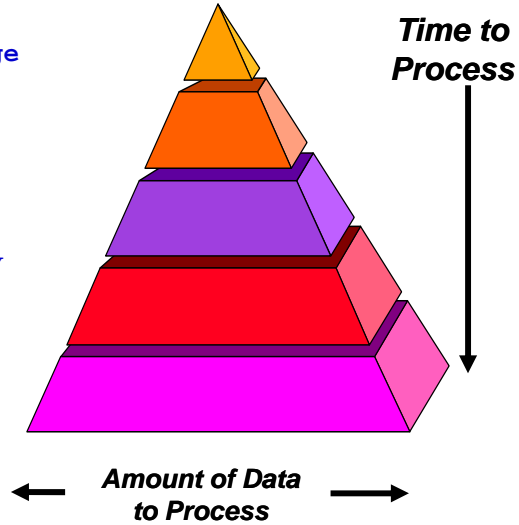
- New report_timing -exclude option
 - Allows designers to report paths NOT through a given pin or sub-block
 - Excludes all paths from/through/to specified pins
 - Enhances usability and flexibility for path analysis
- Example:
 - report_timing -exclude object_list
 - get_timing_paths -exclude object_list
 - object_list: a list of pins, ports, cells, or nets
 - Also supported: -rise_exclude, -fall_exclude

© 2007 Synopsys, Inc. (26) CONFIDENTIAL

SYNOPSYS
Predictable Success

Command Runtime Usage

```
check_timing
report_analysis_coverage
report_bottleneck
report_constraint
report_constraint
  -all_violators
  -max_delay -min_delay
report_constraint
  -all_violators
  -verbose
report_timing
  -nworst 100
```



© 2007 Synopsys, Inc. (27) CONFIDENTIAL

SYNOPSYS
Predictable Success

Agenda

- Introduction
- Basic Flow
- **Advanced Analysis**
- Graphical User Interface (GUI)
- Back Annotation
- Modeling
- Usability
- Signal Integrity Analysis (PT-SI)
- Summary

© 2007 Synopsys, Inc. (28) CONFIDENTIAL

SYNOPSYS
Predictable Success

Advanced Timing Analysis Features in PrimeTime

- Analysis Modes
- Data to Data Checks
- Case Analysis
- Multiple Clocks per Register
- Non-Unate Clock Network
- Minimum Pulse Width Checks
- Derived Clocks
- Clock Gating Checks
- Netlist Editing
- PT to Astro ECO Flow
- report_clock_timing
- Clock Reconvergence Pessimism
- Worst-Arrival Slew Propagation
- Path-Based Analysis
- Debugging Delay Calculation

© 2007 Synopsys, Inc. (29) CONFIDENTIAL

SYNOPSYS
Predictable Success

Analysis Mode

- PrimeTime performs analysis based upon a [Single operating condition, BC_WC, and On Chip Variation \(OCV\)](#)
- From 2004.12 onwards, OCV is the default operating mode
- OCV provides the most accurate analysis
 - Properly handles slew and delay variations
- Single & BC_WC mode could be optimistic for delay calculation
- Use report_design to see which analysis mode is being used



SolvNet Doc Id: 013762

What is the difference between single, bc_wc, and on_chip_variation analysis mode

© 2007 Synopsys, Inc. (30) CONFIDENTIAL

SYNOPSYS
Predictable Success

Analysis Mode Summary

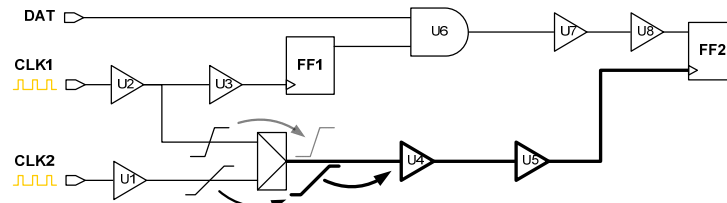
Operating Conditions	Number of OC	Slew Propagation	Timing Analysis
Single OC	1: OC	max slew @ OC	setup @ OC: <ul style="list-style-type: none"> • max data @ OC • min clock @ OC hold @ OC: <ul style="list-style-type: none"> • min data @ OC • max clock @ OC
Min-max: 2 simultaneous BC and WC	2: BC and WC	min slew @ BC max slew @ WC	setup @ WC: <ul style="list-style-type: none"> • max data @ WC • min clock @ WC hold @ BC: <ul style="list-style-type: none"> • min data @ BC • max clock @ BC
on-chip PVT Variation	2: OC - Δ_{\min} OC + Δ_{\max}	min slew @ OC - Δ_{\min} max slew @ OC + Δ_{\max}	setup @ OC: <ul style="list-style-type: none"> • max data @ OC + Δ_{\max} • min clock @ OC - Δ_{\min} hold @ OC: <ul style="list-style-type: none"> • min data @ OC - Δ_{\min} • max clock @ OC + Δ_{\max}

© 2007 Synopsys, Inc. (31) CONFIDENTIAL

SYNOPSYS
Predictable Success

Setup Optimism in the single Mode

- Consider the following setup path:



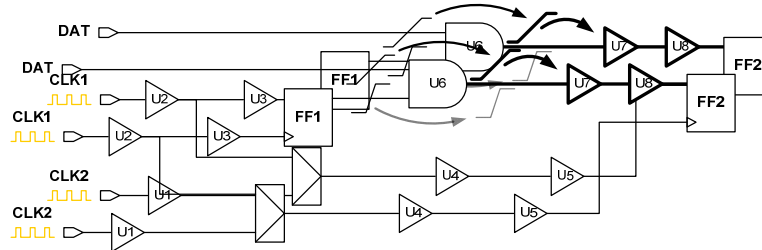
- Slow slew is propagated from mux output
- Slow delays computed for {U4 U5}
- But fastest possible setup capture leg is needed
 - Setup analysis is optimistic for paths captured by CLK1!

© 2007 Synopsys, Inc. (32) CONFIDENTIAL

SYNOPSYS
Predictable Success

Hold Optimism in the single Mode

- Consider the following hold path:



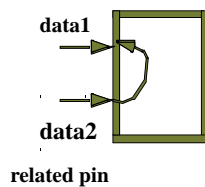
- Slow slew is propagated from U6/Z
- Slow delays computed for {U7 U8}
- But fastest possible launch leg is needed
 - Hold analysis is optimistic for paths launched by FF1!

© 2007 Synopsys, Inc. (33) CONFIDENTIAL

SYNOPSYS
Predictable Success

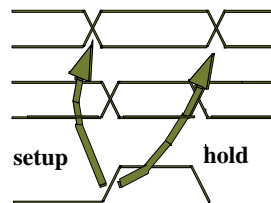
Data to Data checks

Constrained pin



data1

data2



setup

hold

- Data checks are checks defined between two signals in the form of setup and hold arcs
- Commands
 - set_data_check** - set data to data checks between any two pins
 - remove_data_check** - specifies the data check to be removed
- Used for: bus skew checks, handshaking interfaces, self timed circuits, etc

© 2007 Synopsys, Inc. (34) CONFIDENTIAL

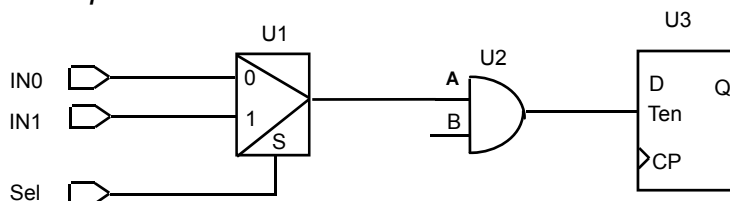
SYNOPSYS
Predictable Success

Case Analysis (1)

- **Concept:**

- Set logic constants ('1' or '0') or logic transitions ('rise' or 'fall') on pins or ports of the design.
- These constants are forwarded throughout the logic, enabling or disabling timing arcs as appropriate.
- Syntax: `set_case_analysis 0|1 port_or_pin_list`

- **Example:**



- » If port Sel is set to a constant '0', only the IN0 to U3 path will be traced.
- » If pin U2/B is set to a constant 0, no paths to U3/Ten will be traced.

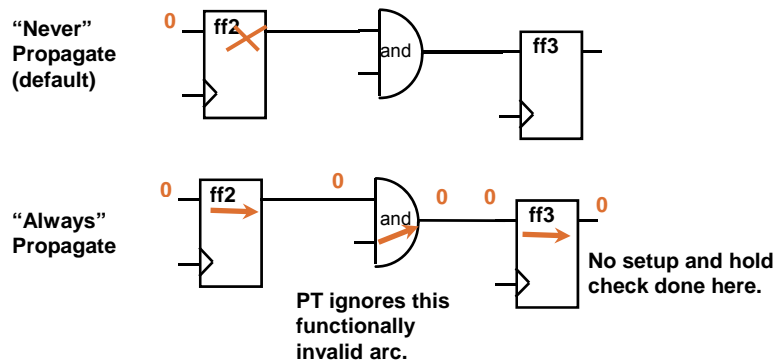
© 2007 Synopsys, Inc. (35) CONFIDENTIAL

SYNOPSYS
Predictable Success

Case Analysis (2)

`case_analysis_sequential_propagation` variable:

- Determines whether case analysis is propagated across sequential cells.
- Default is "never" -- which disables constant propagation.
- Global variable, applies to design, not instance.

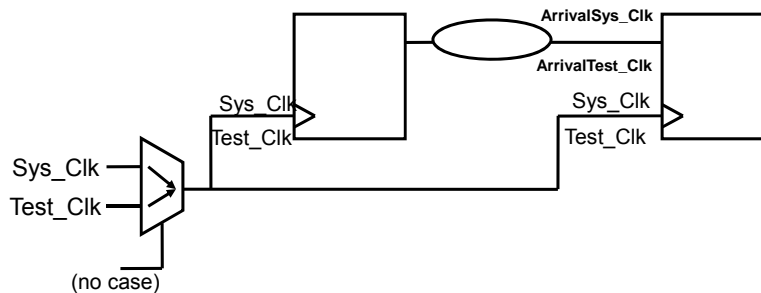


© 2007 Synopsys, Inc. (36) CONFIDENTIAL

SYNOPSYS
Predictable Success

Multiple Clocks Per Register

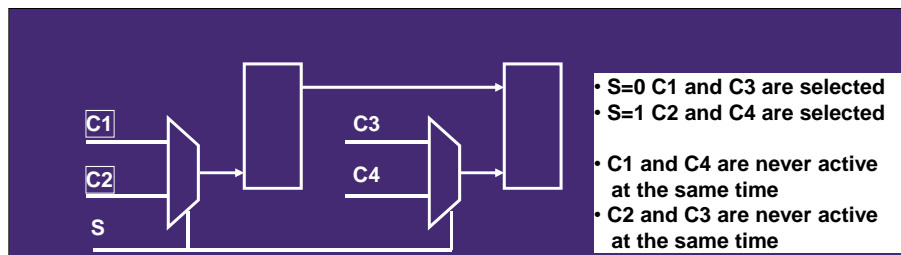
- PrimeTime does enable:
 - Simultaneous analysis of multiple clocks per register
 - Analysis of all clocks in one run; no need to set case analysis
 - Control specific combinations by using `set_false_path`, `set_clock_groups`, `set_active_clocks` and `exclusive clocks`.



© 2007 Synopsys, Inc. (37) CONFIDENTIAL

SYNOPSYS
Predictable Success

Exclusive Clock Groups



- Multiple Clock Propagation
 - `set_false_path` manually between mutually exclusive clocks and do the analysis
- Multiple Clock Propagation with Active Clock groups
 - Set C1, C3 and C2, C4 to be exclusive clocks
 - Analyze C1 and C3 only

```

set_clock_groups -logically_exclusive -name E1 -group {c1 c3}
                  -group {c2 c4}
set_active_clocks {c1, c3}
      
```

© 2007 Synopsys, Inc. (38) CONFIDENTIAL

SYNOPSYS
Predictable Success

Clock Groups

- Set active clocks to be analyzed together
 - `set_active_clocks <clock_list>`
- Specify clock group information
 - `set_clock_groups [-logically_exclusive/-asynchronous] [-name <name>] -group <clock_list>`
- Remove the clock groups
 - `remove_clock_groups [-logically_exclusive/-asynchronous] [-name <name_list>] [-all]`
- Report clock groups set
 - `report_clock [-groups]`

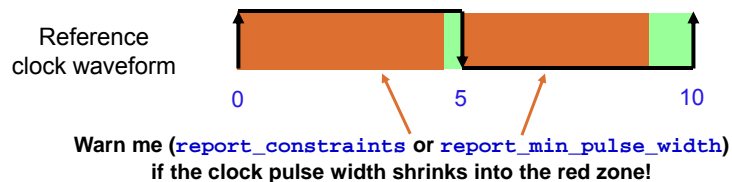
© 2007 Synopsys, Inc. (39) CONFIDENTIAL

SYNOPSYS
Predictable Success

Minimum Pulse Width Check

- You can specify a minimum pulse width on clocks, pins, cells, or all clocks in a design
 - Separate values can be specified for active low and active high phase of pulse
- Minimum pulse width checks can be back-annotated from SDF

```
set_min_pulse_width -high 4.5 [all_clocks]
set_min_pulse_width -low 4.0 [all_clocks]
```



© 2007 Synopsys, Inc. (40) CONFIDENTIAL

SYNOPSYS
Predictable Success

Specifying Generated (Derived) Clocks

- Generated clocks are created within the design and are derived from a source clock
 - Examples: clock dividers, clock multipliers, edge relationships
- To specify a clock derived from an existing clock:
 - `create_generated_clock`

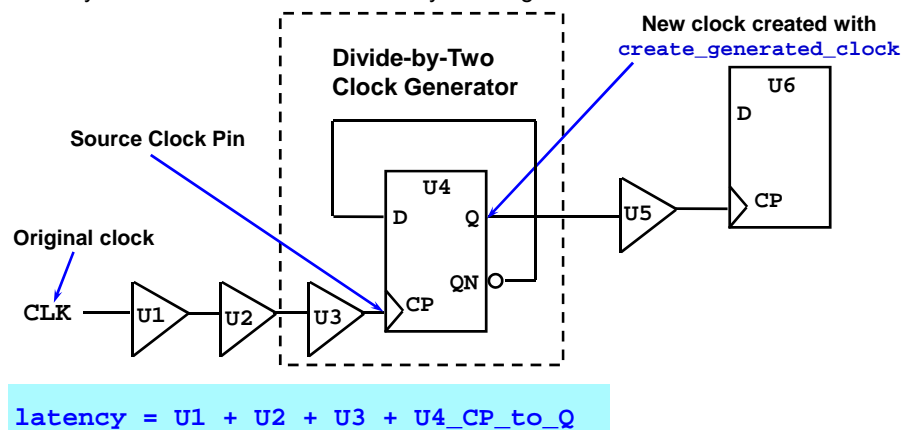
```
pt_shell> create_generated_clock -divide_by 2 \  
        -source sys_clk [get_pins FF1/Q]  
pt_shell> create_generated_clock -multiply_by 3 \  
        -source sys_clock [get_pins div3/Q]  
pt_shell> create_generated_clock -edges {1 3 5} \  
        -source sys_clock [get_pins DIV_CLK]
```

© 2007 Synopsys, Inc. (41) CONFIDENTIAL

SYNOPSYS
Predictable Success

Automatic Calculation of Generated Clock Latency

- PrimeTime will automatically calculate the latency (delay) from the source clock to the generated clock if the source clock is propagated and you have not set a source latency on the generated clock.

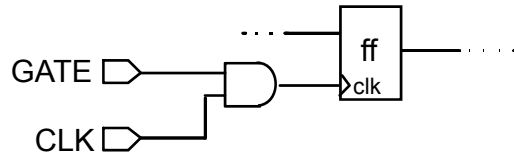


© 2007 Synopsys, Inc. (42) CONFIDENTIAL

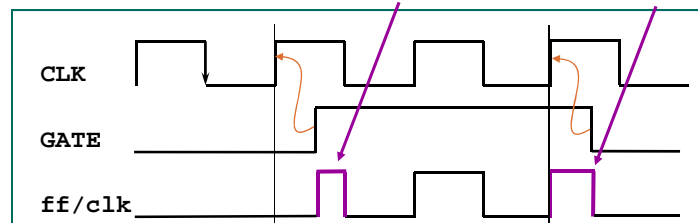
SYNOPSYS
Predictable Success

Clock Gating Checks (1)

- Gated clocks can cause glitches or “clipped” clocks if gating signal changes during the “active” portion of the clock.



Glitches due to late arrival time of GATE



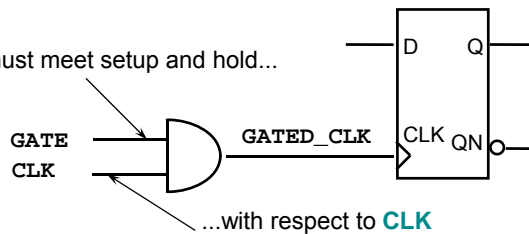
© 2007 Synopsys, Inc. (43) CONFIDENTIAL

SYNOPSYS
Predictable Success

Clock Gating Checks (2)

- PrimeTime enables you to specify setup and hold requirements for gated clocks
 - Invoked automatically if gating logic exists, default values for setup and hold requirements are set to “0”

GATE must meet setup and hold...



...with respect to **CLK**

```
set_clock_gating_check -setup 0.25 -hold 0.1 CLK
```

- Clock gating checks are active by default. Disable with:
`set_timing_disable_clock_gating_checks true`






Note: Default value of clock gating setup and hold checks is 0.0

© 2007 Synopsys, Inc. (44) CONFIDENTIAL

SYNOPSYS
Predictable Success

Netlist Editing ("What if") (1)

- "What If" analysis allows minor editing of the netlist
 - Not intended to replace IPO!
- Commands for netlist/timing modification:

 `size_cell`
 `insert_buffer/remove_buffer`
 `create_cell/remove_cell`
 `create_net/remove_net`
 `swap_cell`

© 2007 Synopsys, Inc. (45) CONFIDENTIAL

SYNOPSYS
Predictable Success

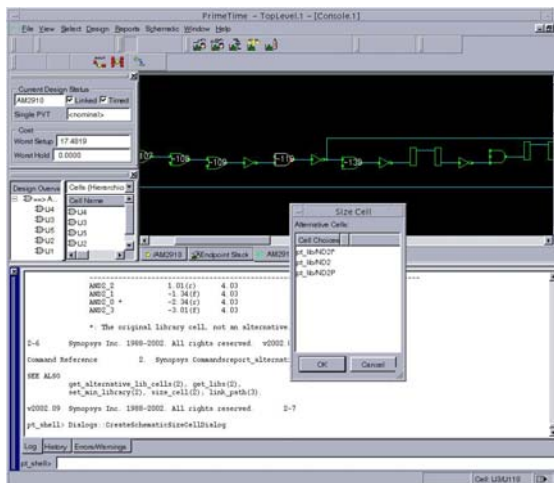
Netlist Editing (2) What-if Analysis

To list cells in GUI:

- 1) Highlight cell (select)
- 2) Schem → size cell

In pt_shell:

`get_alternative_lib_cells`
`report_alternative_lib_cells`



Enables guided cell sizing for timing optimization

- explore alternatives
- commit to a change

© 2007 Synopsys, Inc. (46) CONFIDENTIAL

SYNOPSYS
Predictable Success

Netlist Editing (3)

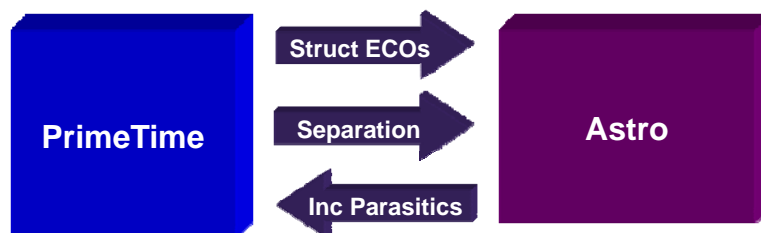
- **Change made to a multiply instantiated design:**
 - PT "auto-uniquifies" the design
 - Only blocks that require it will be auto-uniquified
- **How do I save the changes?**
- **Use `write_changes` command**
 - Writes a text-based description of the changes, or
 - Writes a DC shell Tcl command script, or
 - Writes a PT command file to recreate the changes

© 2007 Synopsys, Inc. (47) CONFIDENTIAL

SYNOPSYS
Predictable Success

PrimeTime to Astro ECO Flow (version 2005.06)

- Many improvements have been made to enable a seamless ECO flow between PrimeTime and Astro



© 2007 Synopsys, Inc. (48) CONFIDENTIAL

SYNOPSYS
Predictable Success

Direct Output of Astro ECO Files

- PrimeTime now supports direct output of netlist changes as Astro ECO files
 - structural changes in HECO format
 - coupling separations in scheme format

```
write_astro_changes # Write design changes for Astro
-format type        (Astro format to write:
                    Values: heco, scheme)
[-dont_merge_changes] (Don't merge structural changes)
file_name           (Output file for design changes)
```

© 2007 Synopsys, Inc. (49) CONFIDENTIAL

SYNOPSYS
Predictable Success

Direct Output of Astro ECO Files: Structural ECO Changes

- HECO file is read into Astro with:
auHECOByFile <lib> <cell> <HECO_file>
 - Cell does not need to be open
 - If cell is open, it is closed
- Then reload cell and perform:
 - ECO/ECO Place – Design ECO
 - ECO/ECO Route – Design ECO



© 2007 Synopsys, Inc. (50) CONFIDENTIAL

SYNOPSYS
Predictable Success

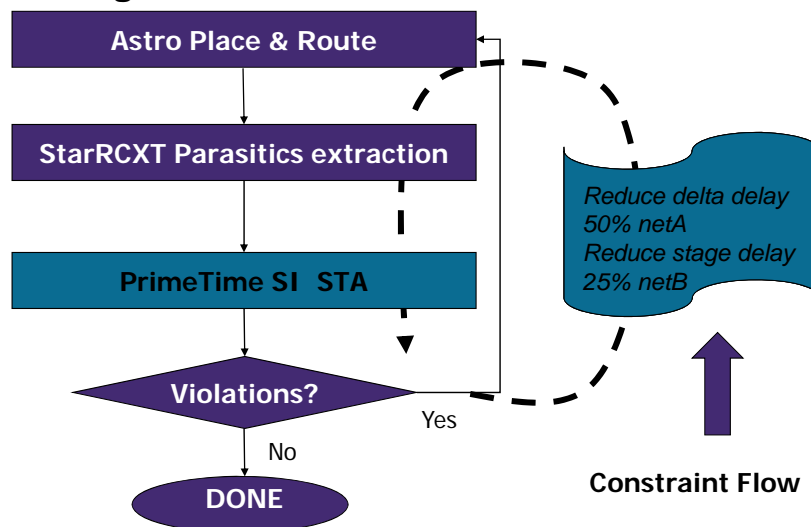
Direct Output of Astro ECO Files: Coupling Separations

- Coupling separations are written as Astro scheme commands
 - Only "effective" aggression couplings are considered
 - Avoids writing unneeded or ineffective separation directives
 - Separation directives are written in both directions
 - i.e. – "Separate n1 from n2,
Separate n2 from n1"
 - Allows maximum possible router freedom
 - Directives are not written which move clock nets
 - Directives are still written to move nets away from clock nets

© 2007 Synopsys, Inc. (51) CONFIDENTIAL

SYNOPSYS
Predictable Success

Binary ECO Fixing Flow starting version 2005.12



© 2007 Synopsys, Inc. (52) CONFIDENTIAL

SYNOPSYS
Predictable Success

Binary Constraint-based Flow

PTSI directly identifies problem nets & provides constraints with current & target timing to Astro

- PrimeTime SI identifies optimal set of nets to reduce delay/noise in order to achieve timing/SI closure
- PrimeTime SI generates constraints for Astro, which contain current & target timing for these nets.
- Astro determines what changes to make, based on sign-off timing.
- Goal – simplification!!!!



SolvNet Doc Id: 017980

PrimeTime SI / Astro Constraint ECO Flow - Application Note and FAQ

© 2007 Synopsys, Inc. (53) CONFIDENTIAL

SYNOPSYS
Predictable Success

Usage Model

```
pt_shell> create_eco_astro_constraints -help
```

Usage:

```
create_eco_astro_constraints # Create ECO constraints for Astro to fix crosstalk delay, timing and noise
```

```
[-output file_name] (Name of the output constraint file)
```

```
[-constraint_type constraint_type_list]
```

(Specifies one or multiple constraint types of delta_delay, stage_delay and noise.

(default = {delta_delay stage_delay}))

```
[-delay_type max|min|min_max] (Type of path delay to fix (default = max))
```

```
[-effort_level low|high] (Effort level)
```

```
[-validate] (Performs what-if analysis by back-annotating constraints and update_timing)
```

```
[-group path_group_name] (Path group)
```

```
[-max_constraints number_of_constraints] (maximum number of constraints (default = 1000))
```

```
[-max_iteration number_of_iterations] (Maximum number of iterations to create constraints (default = 3))
```

```
[-verbose] (Verbose mode)
```

```
[object_list] (Collection of nets or paths)
```

© 2007 Synopsys, Inc. (54) CONFIDENTIAL

SYNOPSYS
Predictable Success

Usage - Example

```
create_eco_astro_constraints -output
./const/test1.cst
Collecting 100 max SI bottleneck nets to analyze
...
Collected worst 1215 timing paths to analyze...
Timing Constraint Generation
=====
Starting to create constraints to reduce delays ...
Getting paths with violations ...

Warning: More than 10000 paths with violations.
Warning: Getting the worst 10000 paths ...

Analyzing path topology ...
Processing 10000 paths ...
  10 percent completed ...
  20 percent completed ...
  30 percent completed ...
  40 percent completed ...
  50 percent completed ...
  60 percent completed ...
  70 percent completed ...
  80 percent completed ...
  90 percent completed ...
 100 percent completed ...
```

```
Creating delta delay constraints ...
Processing 10000 paths ...
  10 percent completed ...
  20 percent completed ...
  30 percent completed ...
  40 percent completed ...
  50 percent completed ...
  60 percent completed ...
  70 percent completed ...
  80 percent completed ...
  90 percent completed ...
 100 percent completed ...
```

```
Creating stage delay constraints ...
```

Constraint Creation Summary

```
-----
Elapsed CPU time              707
Number of examined paths with violations
10000
Number of delta delay constraints      9
Number of stage delay constraints     54

Writing constraint file ./const/test1.cst ...
```

© 2007 Synopsys, Inc. (55) CONFIDENTIAL

SYNOPSYS
Predictable Success

Clock Timing Reporting (1)

- PrimeTime is capable of analyzing clock latency, clock skew (including inter-clock skew) and clock transition attributes of a clock network after CTS automatically.
- Command:
 - `report_clock_timing \`
 `-type <skew, interclock_skew, latency, transition,`
 `summary> \`

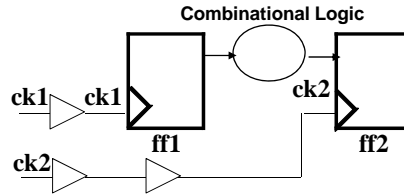
© 2007 Synopsys, Inc. (56) CONFIDENTIAL

SYNOPSYS
Predictable Success

Clock Timing Reporting (2) Inter-Clock Skew

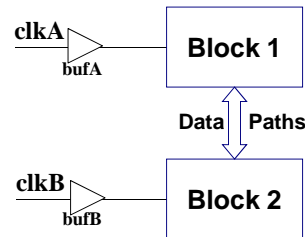
- To find setup skew between registers ff1 and ff2

```
report_clock_timing -from
ff1/CK -to ff2/CK -
setup -type
interclock_skew
```



- To find 5 largest setup skews between blocks 1 and 2

```
report_clock_timing -from
clkA -to clkB -setup -
nworst 5 -type
interclock_skew
```



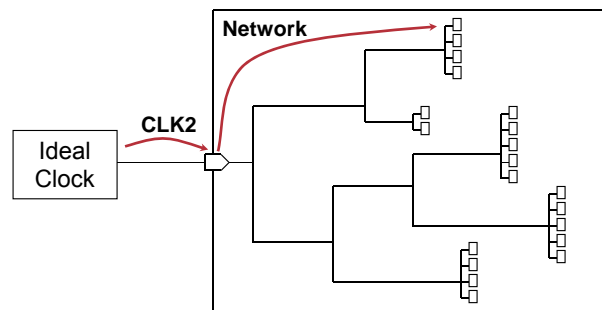
© 2007 Synopsys, Inc. (57) CONFIDENTIAL

SYNOPSYS
Predictable Success

Clock Timing Reporting (3) Clock Latency

- Find 5 worst launching rise latencies in the clock network CLK2 for hold paths

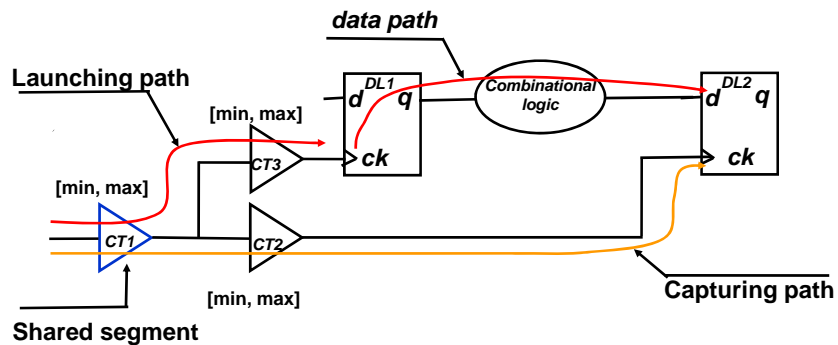
```
report_clock_timing -clock CLK2 -hold -rise
-launch -nworst 5 -type latency
```



© 2007 Synopsys, Inc. (58) CONFIDENTIAL

SYNOPSYS
Predictable Success

What is Clock Reconvergence Pessimism?



- Timing inaccuracy due to different delays (min & max) used for a shared segment
- Correction of timing inaccuracy is called CRPR (CRP Removal)



SolvNet Doc Id: 002064

What is Clock Reconvergence Pessimism Removal (CRPR)?

© 2007 Synopsys, Inc. (59) CONFIDENTIAL

SYNOPSYS
Predictable Success

Enabling CRPR

- To enable CRPR, set the following variable prior beginning timing analysis:
 >> set **timing_remove_clock_reconvergence_pessimism** true
- By default, CRPR analysis is disabled
- Using CRPR analysis results in a more accurate (less pessimistic) analysis, but causes an increase in runtime and memory usage

© 2007 Synopsys, Inc. (60) CONFIDENTIAL

SYNOPSYS
Predictable Success

CRPR report

- `report_crpr` command
- User can see the CRPR calculation data such as
 - Common Point
 - CRPR Value for Rise and Fall
- User must supply a `-from` and `-to` option between the two sequential elements, e.g.
 - `pt_shell> report_crpr -from dff1/CP -to dff2/CP \`
`-clock CLK -type setup`



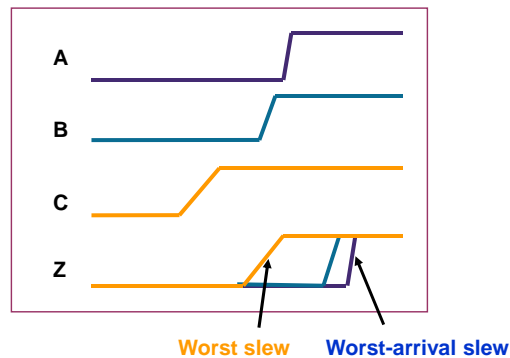
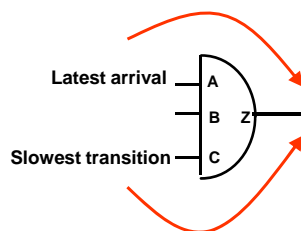
SolvNet Doc Id: 005511

PrimeTime Clock Reconvergence Pessimism Removal (CRPR) App Note

© 2007 Synopsys, Inc. (61) CONFIDENTIAL

SYNOPSYS
Predictable Success

Worst-Arrival Slew Propagation



- Slew corresponding to latest arrival at Z
- Con: Can be optimistic depending on slew sensitivity of fanout logic
- Pro: Accurate slew applies to critical path per end point only
- To enable worst-arrival slew propagation:

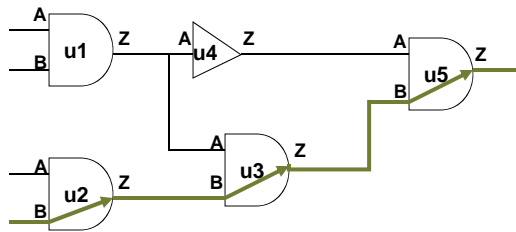

```
>> set timing_slew_propagation_mode [worst_slew or worst_arrival]
```

© 2007 Synopsys, Inc. (62) CONFIDENTIAL

SYNOPSYS
Predictable Success

Path-Based Analysis

- Provides reduced pessimism
- Re-calculate cell and net delays on user-selected paths
 - Uses path-specific arrival times & slews
 - Re-calculates corresponding clock path
 - Re-calculates setup/hold constraint arcs
 - Uses revised (victim) timing edge



© 2007 Synopsys, Inc. (63) CONFIDENTIAL

SYNOPSYS
Predictable Success

Path-Based Analysis Usage

- Add `-recalculate` to `get_timing_paths` or `report_timing` commands:

```
get_timing_paths -nworst 5 -max_paths 50 -recalculate
```

- Or `report_timing` directly with PBA on specified paths:

```
report_timing -nworst 5 -max_paths 50 -recalculate
```

© 2007 Synopsys, Inc. (64) CONFIDENTIAL

SYNOPSYS
Predictable Success

Path-Based Analysis Usage

- Recalculated results are for reporting purposes only, and cannot be saved in the database
 - Not available for use by other commands
- Path Based Analysis
 - Recommended for end of the design process
 - Not intended for use with large number of violating paths
- Fast Reporting Procedure for Path Based Analysis
 - report_recalculated_paths
 - See SolvNet Id: 019538

A Fast (But Non-Exhaustive) Path-Based Analysis Search in Tcl



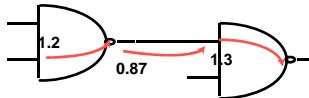
SolvNet Doc Id: 012134

Accurate Signoff analysis with PrimeTime's path-based analysis

© 2007 Synopsys, Inc. (65) CONFIDENTIAL

SYNOPSYS
Predictable Success

Delay Calculation



- **Delay Calculation determines the time to propagate a transition across a net or cell timing arc.**
- **Pre-layout**
 - Synopsys delay model from library : Nonlinear Delay, etc.
 - Wire load models for net estimation
- **Post-layout**
 - Accurate analysis with detailed parasitics:
 - "Adaptive Arnold" algorithm is used to compute net delay & slew
 - SDF from vendor delay calculator can be used directly
 - CCS Models for leading edge technologies

© 2007 Synopsys, Inc. (66) CONFIDENTIAL

SYNOPSYS
Predictable Success

Delay Calculation - Example

Partial output of `report_timing -net -input_pins:`

How is this delay calculated?

```
...
U2/n442 (net)                2
U2/U100/A (EO)                0.02    54.37 f
U2/U100/Z (EO)                2.61    56.99 f
U2/n309 (net)                1
U2/U62/B (AN2)                0.01    56.99 f
U2/U62/Z (AN2)                1.91    58.90 f
U2/OUTPUT40[2] (net)        1
U2/OUTPUT_reg[2]/D (FD1)     0.01    58.91 f
data arrival time              58.91

clock CLOCK (rise edge)      30.00    30.00
clock network delay (ideal)   5.50    35.50
clock uncertainty             -0.50    35.00
U2/OUTPUT_reg[2]/CP (FD1)    35.00 r
library setup time           -0.80    34.20
data required time            34.20

-----
data required time            34.20
data arrival time             -58.91
-----
slack (VIOLATED)              -24.71
```

© 2007 Synopsys, Inc. (67) CONFIDENTIAL

SYNOPSYS
Predictable Success

Delay Calculation - Example

`pt_shell> report_delay_calculation -from U2/U62/B -to U2/U62/Z`

```
*****
Report : delay_calculation
Design : AM2910
Version: 2000.11
Date   : Thu Jun 3 18:01:20 1999
*****

Library: 'pt_lib'
Library Cell: 'AN2'

positive_unate arc:
Units: lns 0.0001pF 10000kOhm
Input net transition times:  Dt_rise = 0.339297,
                             Dt_fall = 0.151962

Rise Delay computation:
rise_intrinsic              1.11463 +
rise_slope * Dt_rise        0 * 0.339297 +
rise_resistance * (pin_cap + wire_cap) / driver_count
0.335086 * (1 + 0.00215) / 1
-----
Total                        1.45044
```

```
Fall Delay computation:
fall_intrinsic              1.78806 +
fall_slope * Dt_fall        0 * 0.151962 +
fall_resistance * (pin_cap + wire_cap) / driver_count
0.121448 * (1 + 0.00215) / 1
-----
Total                        1.90977

Rise delay = 1.45044
Fall delay = 1.90977

rise_resistance * (pin_cap + wire_cap) / driver_count
0.335086 * (1 + 0.00215) / 1
= 0.335807

fall_resistance * (pin_cap + wire_cap) / driver_count
0.121448 * (1 + 0.00215) / 1
= 0.12171

Rise transition = 0.335807
Fall transition = 0.12171
```

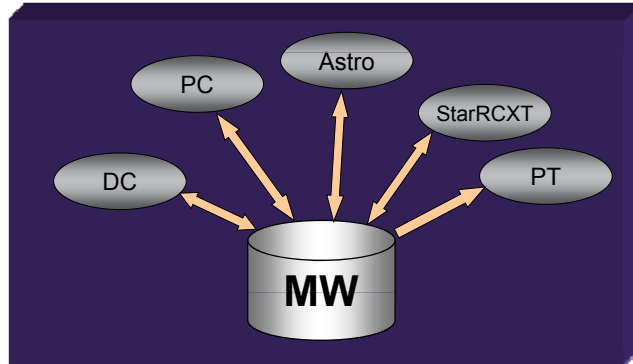
© 2007 Synopsys, Inc. (68) CONFIDENTIAL

SYNOPSYS
Predictable Success

Milkyway Support

Overview

- Improve the integration with the Galaxy Platform



© 2007 Synopsys, Inc. (69) CONFIDENTIAL

SYNOPSYS
Predictable Success

Milkyway Support

Usage

- PrimeTime script for MilkyWay access **must** include the following:
 - Specify the `search_path` to include the path to Milkyway reference library directories

```
set search_path " . ./mw_db"
```
 - Specify the `link_path` to include Milkyway reference libraries

```
set link_path " * lib_name.db"
```
 - Use `read_milkyway` command to read in the CEL view

```
read_milkyway TOP_CELL -library ./mw_db
```
- After reading the design from MilkyWay, the Parasitics can be read from the MilkyWay PARA view:
 - `read_parasitics -format PARA`

© 2007 Synopsys, Inc. (70) CONFIDENTIAL

SYNOPSYS
Predictable Success

Agenda

- Introduction
- Basic Flow
- Advanced Analysis
- **Graphical User Interface (GUI)**
- Back Annotation
- Modeling
- Usability
- Signal Integrity Analysis (PT-SI)
- Summary

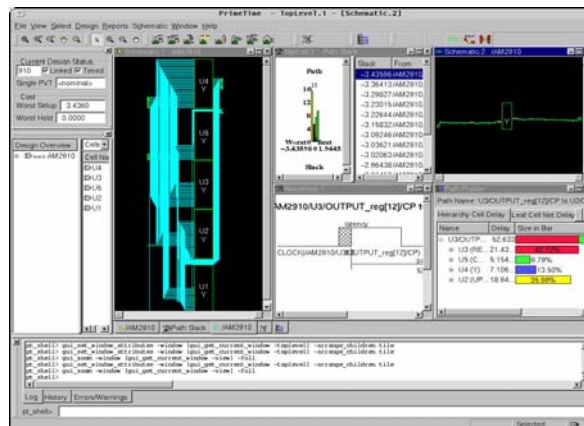
© 2007 Synopsys, Inc. (71) CONFIDENTIAL

SYNOPTSYS
Predictable Success

PrimeTime Graphical User Interface

✓ The PrimeTime GUI consists of the following windows:

- » Main Console
- » Hierarchy Browser
- » Histogram
- » Path Profiler
- » Schematic
- » Report
- » Waveform



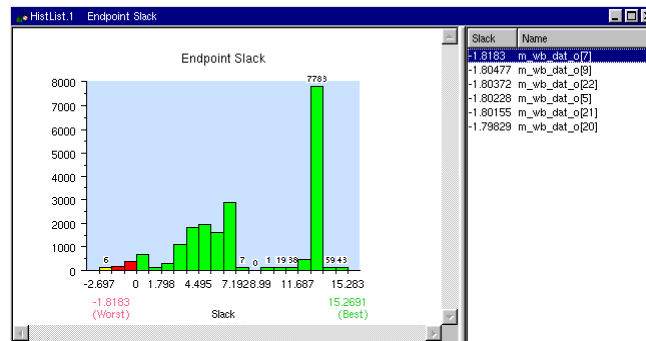
✓ To run the GUI:

```
unix> primetime  
or  
pt_shell> start_gui
```

© 2007 Synopsys, Inc. (72) CONFIDENTIAL

SYNOPTSYS
Predictable Success

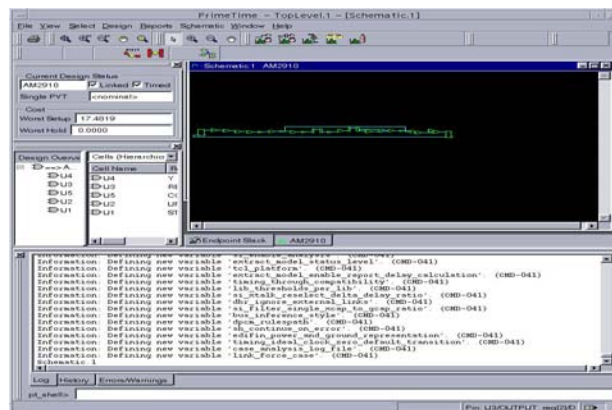
- Endpoint slack histogram shows distribution of endpoints by slack
 - Violating bins shown in red
 - Selected bins shown in yellow
 - Multiple bins can be selected by holding down the Ctrl key
 - Items in selected bins are shown in rightmost pane



© 2007 Synopsys, Inc. (73) CONFIDENTIAL

SYNOPSIS
Predictable Success

- Select path
- Schematic→
- New



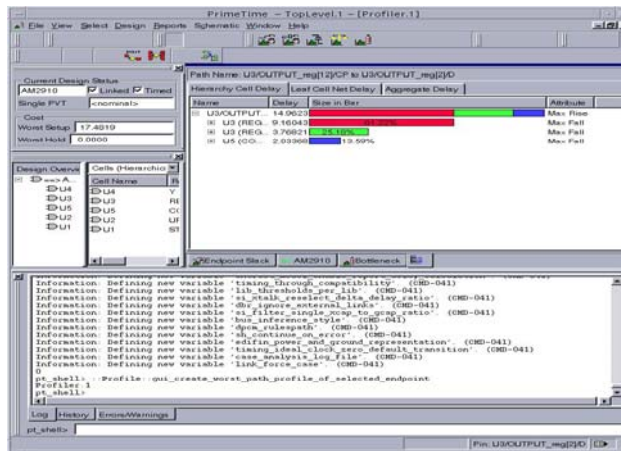
- Shows all cells, nets and hierarchy crossings for a single timing path
- Optional annotations show slack, capacitance, transition, etc
- Capable of adding fanin or fanout cones of selected pins or nets

© 2007 Synopsys, Inc. (74) CONFIDENTIAL

SYNOPSIS
Predictable Success

Path Profiler - Hierarchical Path Analysis

- Select path
- Right hand mouse click
- Profile



- Shows relative contribution of hierarchical cells to path delay
- Can push into hierarchy

© 2007 Synopsys, Inc. (75) CONFIDENTIAL

SYNOPSYS
Predictable Success

Path Inspector

- The Path Inspector schematics can highlight portions of a path:

- clock launch
- clock capture
- data path



- CRPR common point can also be shown in red

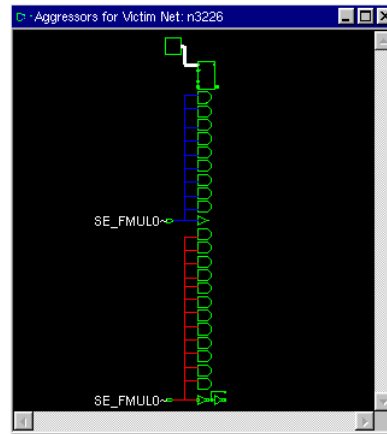
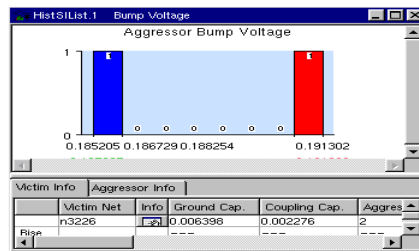


© 2007 Synopsys, Inc. (76) CONFIDENTIAL

SYNOPSYS
Predictable Success

Crosstalk Aggressor Schematic

- Displays victim and aggressors with associated drivers and loads
- Color-coded histogram of aggressor nets also displayed

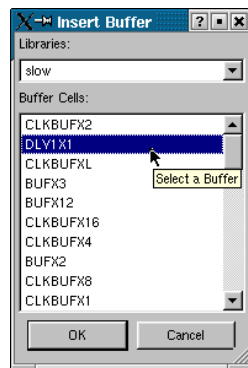


© 2007 Synopsys, Inc. (77) CONFIDENTIAL

SYNOPSYS
Predictable Success

What-If ECO Changes

- Cell resizing, buffer insertion can be performed from the GUI
 - Schematic -> Size Cell – resizes selected cell
 - Schematic -> Insert Buffer – adds buffer at selected input/output pin(s)
- Cell resizing dialog allows new slack to be computed for each potential replacement library cell



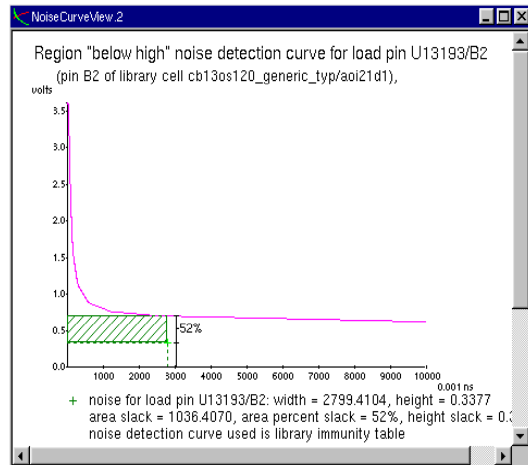
Library Cell Name	Area	Timing Model Type	Slack
slow/NOR4X1*	19.958401	default	Not Calculated
slow/NOR4X2	29.937599	default	Not Calculated
slow/NOR4XL	19.958401	default	Not Calculated
slow/NOR4X4	56.548801	default	Not Calculated

© 2007 Synopsys, Inc. (78) CONFIDENTIAL

SYNOPSYS
Predictable Success

Noise Immunity Curve

- Area slack for noise bump is displayed



© 2007 Synopsys, Inc. (79) CONFIDENTIAL

SYNOPSYS
Predictable Success

SI Bottleneck Explorer

- Nets of interest are shown in the top
- Double-click on a net in the top (or click "Explore") to show coupled nets in the bottom

Coupling Analysis Dialog

Net Name	Cost Value	Min Delta	Min Slack	Min Trans	Max Delta	Max Slack	Max Trans	Driver Ty
wishbone/add_589/carry[1]	0.02	-0.01	0.67	0.14	0.02	2.30	0.13	AN2D0
wishbone/add_589/carry[2]	-0.00	0.00	0.69	0.09	0.00	2.30	0.09	AN2D0
wishbone/394/carry[2]	0.00	-0.00	0.65	0.10	0.00	2.27	0.10	AN2D0
wishbone/394/carry[3]	0.01	-0.01	0.65	0.11	0.01	2.27	0.11	AN2D0
wishbone/394/carry[4]	0.01	-0.00	0.65	0.11	0.01	2.27	0.11	AN2D0

Aggressors Coupled to Victim 'wishbone/add_589/carry[1]'

Net Name	Bump Height	Min Delta	Min Slack	Min Trans	Max Delta	Max Slack	Max Trans	Driver Type
wishbone/N768	0.06	-0.01	0.35	0.10	0.01	2.38	0.10	EDFCNQD1
n44	0.03	-0.00	-0.19	0.19	0.00	1.48	0.19	CKAN2D0
wishbone/Rx8DAddress[2]	0.02	-0.00	0.48	0.11	0.01	2.64	0.11	EDFCNQD1

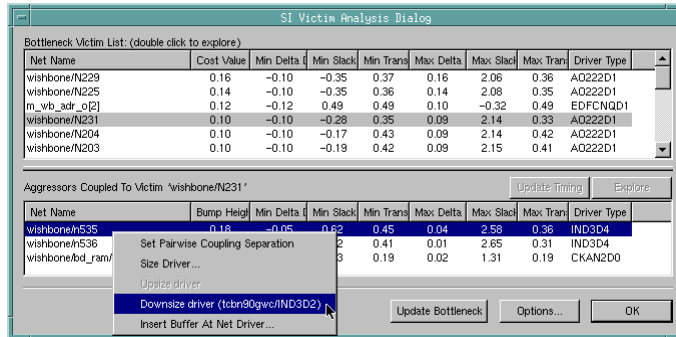
Options... OK

© 2007 Synopsys, Inc. (80) CONFIDENTIAL

SYNOPSYS
Predictable Success

SI Bottleneck Explorer (2)

- Right-click on nets in both sections for useful ECO operations
 - click "Update Timing" to update currently shown nets
 - useful for evaluating effects of ECO operations
 - click "Update Bottleneck" to regenerate bottleneck net list



© 2007 Synopsys, Inc. (81) CONFIDENTIAL

SYNOPSYS
Predictable Success

Agenda

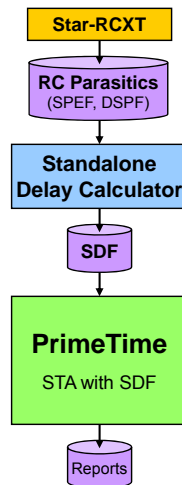
- Introduction
- Basic Flow
- Advanced Analysis
- Graphical User Interface (GUI)
- **Back Annotation**
- Modeling
- Usability
- Signal Integrity Analysis (PT-SI)
- Summary

© 2007 Synopsys, Inc. (82) CONFIDENTIAL

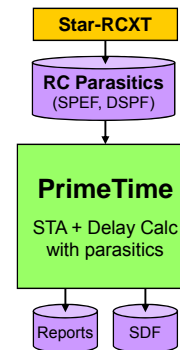
SYNOPSYS
Predictable Success

Back Annotation: SDF, Parasitics

SDF-based Flow



Parasitic Flow



Advantages:

- Integration gives higher accuracy
- Simpler flow
- Single tool
- Zero cost

PrimeTime's Delay Calculation allows post-layout STA to be done with RC parasitics

© 2007 Synopsys, Inc. (83) CONFIDENTIAL

SYNOPSYS
Predictable Success

Back-Annotation - SDF

- PrimeTime can read instance-specific pin-to-pin leaf cell and net timing from SDF files.

```
pt_shell> read_sdf <your_sdf_file>
```

- Remove_annotated_delay supports removing a list of nets

```
pt_shell> remove_annotated_delay <list_nets>
```

- PT supports SDF forward annotation and reporting

```
pt_shell> write_sdf <your_sdf_file>
pt_shell> write_sdf_constraints
pt_shell> report_annotated_delay
```

- PT supports SDF version 1.0, 2.0, 2.1, 3.0

© 2007 Synopsys, Inc. (84) CONFIDENTIAL

SYNOPSYS
Predictable Success

Back-Annotation- Parasitics

- PrimeTime can read the following types of parasitics files:
 - » RSPF (Reduced Standard Parasitic Format)
 - » DSPF (Detailed Standard Parasitic Format)
 - » SPEF (Standard Parasitic Exchange Format)
 - » SBPF (Synopsys Binary Parasitics Format)

```
pt_shell> read_parasitics <parasitic_file>
```

- Formats detected automatically.
- remove_annotated_parasitics supports removing a list of nets or ALL nets

```
pt_shell> remove_annotated_parasitics <list_nets>
```

- Use report_annotated_parasitics to check on completeness of annotation

```
pt_shell> report_annotated_parasitics <list_nets>
```

© 2007 Synopsys, Inc. (85) CONFIDENTIAL

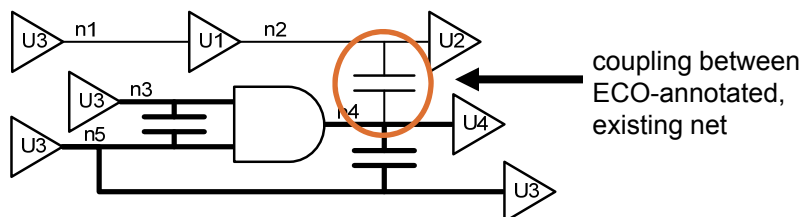
SYNOPSYS
Predictable Success

Incremental Parasitics Support

- Incremental coupling parasitics files from Star-RCXT are now supported with:

```
read_parasitics -eco
```

- Subset of coupled nets annotated by parasitics file
- Allows cross-couplings at boundary of ECO parasitics annotations to be properly managed



© 2007 Synopsys, Inc. (86) CONFIDENTIAL

SYNOPSYS
Predictable Success

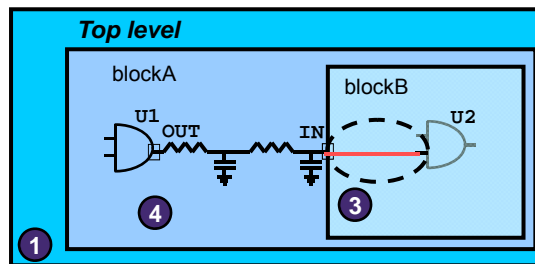
Completing Partial net parasitics:

Selecting a Wire Load Model for Incomplete Nets

- `read_parasitics` incomplete_parasitics.spef
- `complete_net_parasitics -complete_with wlm`
- `complete_net_parasitics -complete_with zero` (DEFAULT)



- 1 Top Level Wire Load
- 2 Default Library Wire Load
- 3 Hierarchical Wire Load
- 4 Wire load for highest level of hierarchy the net traverses



- Top: Use 1 if it exists, else use 2
- Enclosed: Use 3 if the net is fully inside the hierarchy, else use 4
- No wire load model: Use $R=C=0$

Segmented mode
not supported

© 2007 Synopsys, Inc. (87) CONFIDENTIAL

SYNOPSYS
Predictable Success

Scaling Parasitics

- Allows modeling of effects on parasitics due to temperature or other variations
 - Run extraction once at nominal PVT
- Parasitic values can be scaled on a net specific basis, or global scaling factor can be specified
- `scale_parasitics` is not affected by operating conditions
- `scale_parasitics` operates immediately to modify the parasitics in memory
- Different scaling factors can be specified for:
 - Resistance
 - Ground capacitance
 - Coupling capacitance

© 2007 Synopsys, Inc. (88) CONFIDENTIAL

SYNOPSYS
Predictable Success

Ground Capacitance and Resistance Scaling

- Scale factor is a positive floating point number > 0

```
scale_parasitics -ground 1.3 [get_nets n1]
scale_parasitics -resistance 1.2 [get_nets n1]
```

- For nets scaled multiple times or if global scaling is done, last scaling command is applied to the original value

```
scale_parasitics -ground 2.0

# Assume original ground cap is 1pf. scale all nets by factor
of 2. Then cap on all nets will be 2pf.

scale_parasitics -ground 3.0 [get_nets n1]

# scale net n1 by factor of 3. Now cap on all nets (except n1)
will be 2pf. cap on net n1 will be 3pf, NOT 6pf
```



SolvNet Doc Id: 011591

How can I model temperature effects on my detailed parasitics?

© 2007 Synopsys, Inc. (89) CONFIDENTIAL

SYNOPSYS
Predictable Success

Direct Read & Write of gzip files

- Allows users to reduce disk space and network overhead
 - gzip compresses some files to 1/10 or better.
 - PrimeTime can read gzip files directly; no need for intermediate files or pipes.
 - Netlists, constraints, parasitics, scripts,.....
 - To write gzip files directly from PrimeTime, use **-compress** option.

```
pt_shell> write_[script/sdf/sdc] -compress \
          <output_file>
```

© 2007 Synopsys, Inc. (90) CONFIDENTIAL

SYNOPSYS
Predictable Success

Back-Annotation

- Question: ***Can parasitics and SDF be back-annotated at the same time?***
- Answer: **Yes!**
 - SDF delays always override RC calculations.
 - Parasitics are used to compute transition time.
 - Parasitics are used to find max_transition and max_capacitance design rule violations.
- report_timing tags various symbols to indicate type of back-annotation information

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

© 2007 Synopsys, Inc. (91) CONFIDENTIAL

SYNOPSYS
Predictable Success

Agenda

- Introduction
- Basic Flow
- Advanced Analysis
- Graphical User Interface (GUI)
- Back Annotation
- Modeling
- Usability
- Signal Integrity Analysis (PT-SI)
- Summary

© 2007 Synopsys, Inc. (92) CONFIDENTIAL

SYNOPSYS
Predictable Success

PrimeTime Timing Models Support

PrimeTime offers the following timing models to address STA needs for IP, large hierarchical designs, and custom design:

- Quick Timing Model (QTM)
- Extracted Timing Model (ETM)
- Interface Logic Model (ILM)
- Liberty (.lib)
 - Modeling custom blocks/cells



See "PrimeTime Modeling User Guide" in Synopsys Online Documentation (SOLD)

© 2007 Synopsys, Inc. (93) CONFIDENTIAL

SYNOPSYS
Predictable Success

Timing Model Usage Scenarios in PrimeTime

Usage Scenario	Appropriate Model
Top-Down Design	Quick Timing Models
IP Reuse	ETMs
Interface to non-STA and 3rd party tools	ETMs
Synthesis Tasks	ILMs / ETMs
Chip-Level STA	ILMs
Memory, Datapath, Analog	Liberty (.lib) format

© 2007 Synopsys, Inc. (94) CONFIDENTIAL

SYNOPSYS
Predictable Success

- Provide means to quickly and easily create a timing model of an unfinished block for performing timing analysis
- Should later be replaced with gate-level netlists or equivalent models
- Created with PrimeTime commands - *no compiling needed!*
- Can contain:
 - Port specs for the block
 - Setup and hold constraints for inputs
 - Clock-to-output delays
 - Input-to-output delays
- Benefits
 - accurate block specs generated with a lot less effort
 - apply chip level timing constraints and time the whole design
 - allow chip level checks/flows before all blocks complete
 - discover violators in the context of the full design earlier

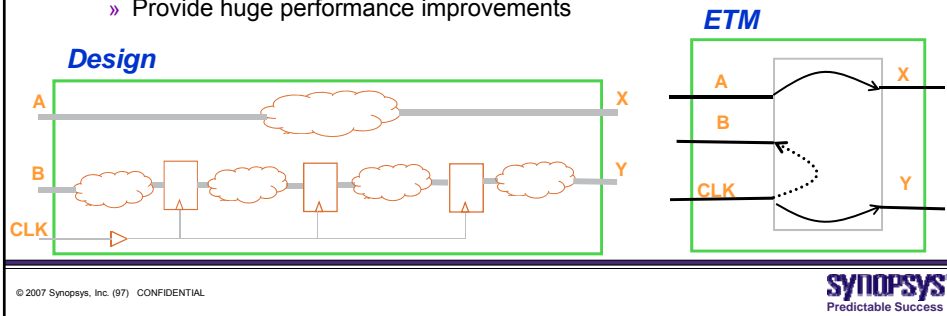
SYNOPSIS
Predictable Success

- QTM is a set of interactive PrimeTime commands - not a language
- Like all PrimeTime commands, QTM can be saved in a script
- QTM model can be saved in db format

SYNOPSIS
Predictable Success

Extracted Timing Models (ETMs)

- Enable IP Reuse and interchange of timing models between EDA tools
- Compact black-box timing models
 - » contain timing arcs between external pins
 - » Internal pins only for generated/internal clocks
 - » models written out in Stamp, .lib ,or db formats
 - » context independent
 - » Exceptions and latches supported
 - » Provide huge performance improvements

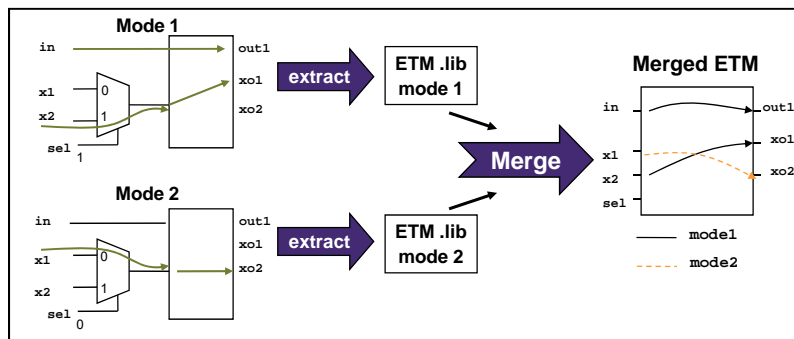


© 2007 Synopsys, Inc. (97) CONFIDENTIAL

SYNOPSYS
Predictable Success

ETM .lib Model Merging

- `merge_model` enhanced to support .lib ETMs
 - Merges multiple ETMs (different modes) to a single ETM model
- Fewer ETMs to manage

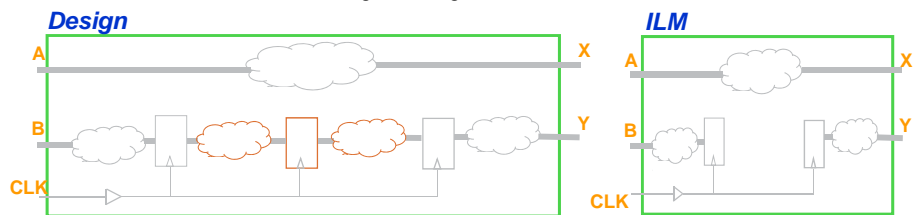


© 2007 Synopsys, Inc. (98) CONFIDENTIAL

SYNOPSYS
Predictable Success

Interface Logic Models (ILMs)

- **Enable Hierarchical STA**
 - Reduce memory and CPU usage for chip-level analysis
 - Offer big netlist reduction if block IOs are registered
 - Back-annotation and constraint files for interface logic are written out along with netlist
- **Benefits:**
 - High accuracy because interface logic is not abstracted
 - Fast model generation time
 - Context independent
 - Can change load, drive, operating conditions, parasitics, SDF, constraints without re-generating the model



© 2007 Synopsys, Inc. (99) CONFIDENTIAL

SYNOPSYS
Predictable Success

Interface Logic Models (ILMs)

- ILMs can be used in SDF and parasitics based flows

```
pt_shell> write_ilm [sdf/parasitics] <output_file>
```

- Support for Hierarchical SI analysis

```
pt_shell> create_ilm -include {xtalk_pins}
```

- Support for Model Validation

```
pt_shell> compare_interface_timing <ref_file> \
    <cmp_file> -slack 0.2 -include slack
```

© 2007 Synopsys, Inc. (100) CONFIDENTIAL

SYNOPSYS
Predictable Success

Agenda

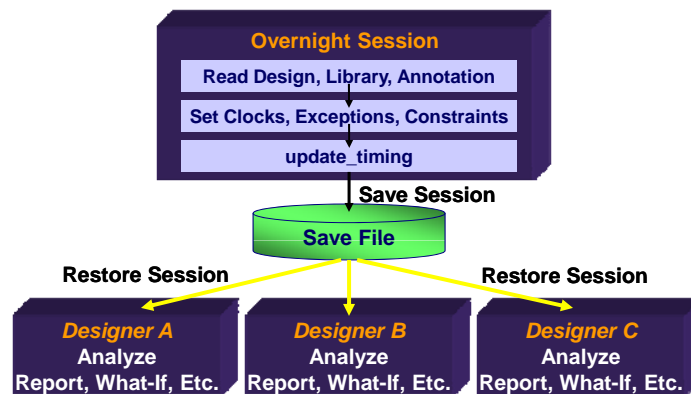
- Introduction
- Basic Flow
- Advanced Analysis
- Graphical User Interface (GUI)
- Back Annotation
- Modeling
- *Usability*
- Signal Integrity Analysis (PT-SI)
- Summary

© 2007 Synopsys, Inc. (101) CONFIDENTIAL

SYNOPSYS
Predictable Success

Save and Restore

- Save and restore PrimeTime session after timing update
 - Enables concurrent and quicker analysis



© 2007 Synopsys, Inc. (102) CONFIDENTIAL

SYNOPSYS
Predictable Success

Save and Restore Usage

- `save_session <session_dir_name> [-replace] \`
`[-include noise]`
 - The saved data is saved to the specified directory.
 - Data may be saved to an existing directory with the `[-replace]` option
 - Saved data will be in binary files.
 - Information will be kept to check for user tampering of files.
 - Noise is only saved if option is given.
- `restore_session <session_dir_name>`

© 2007 Synopsys, Inc. (103) CONFIDENTIAL

SYNOPSYS
Predictable Success

Today's Challenge

- Managing multiple STA/SI runs is common
 - **Multiple Modes * Multiple Corners = Multiple STA Runs**

Issues

- Complex and time-consuming to manage
- Difficult and cumbersome to analyze output
 - Multiple reports from multiple runs

Timing Verification Effort Increasing !

© 2007 Synopsys, Inc. (104) CONFIDENTIAL

SYNOPSYS
Predictable Success

Multi-Scenario Analysis

- Powerful new technology
- Tremendous productivity improvement for multiple mode & corner analysis
- Easy management of multiple jobs
- **Merged reporting**
 - Helps designers identify worst timing violations across many scenarios
- Delivers Full Accuracy, No Compromises

© 2007 Synopsys, Inc. (105) CONFIDENTIAL

SYNOPSYS
Predictable Success

Scenarios

- A multi-scenario analysis is comprised of a number of *scenarios*
 - Scenarios are the building block of multi-scenario analysis
 - A scenario is a single PrimeTime or PrimeTime SI analysis
- Normally, a scenario analyzes a specific operating mode at a specific corner

© 2007 Synopsys, Inc. (106) CONFIDENTIAL

SYNOPSYS
Predictable Success

Scenarios

- Consider a design with four operating modes, analyzed at three PVT corners
 - This would result in 12 scenarios as shown below

	Functional mode	Scan shift mode	Scan capture mode	JTAG mode
worst-case (125°C, 1.35V)	Scenario #1	Scenario #2	Scenario #3	Scenario #4
typical (25°C, 1.50V)	Scenario #5	Scenario #6	Scenario #7	Scenario #8
best-case (0°C, 1.65V)	Scenario #9	Scenario #10	Scenario #11	Scenario #12

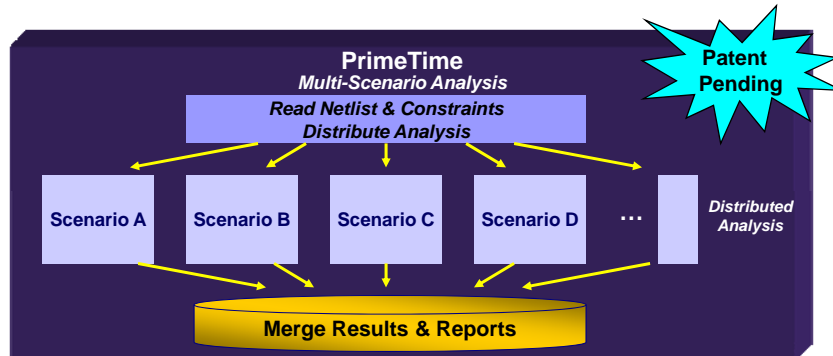
© 2007 Synopsys, Inc. (107) CONFIDENTIAL

SYNOPSYS
Predictable Success

Distributed Multi-Scenario Analysis

Analysis for Multiple Modes & Corners

- Automatically manages & distributes scenarios across CPUs
- Merged reporting across multiple scenarios
 - Identify & debug violations across many scenarios



© 2007 Synopsys, Inc. (108) CONFIDENTIAL

SYNOPSYS
Predictable Success

Invoking a Master Process

- A master process is invoked with the `-multi_scenario` switch:
 - `pt_shell -multi_scenario`
- A master `pt_shell` is different than a normal `pt_shell`
 - different commands
 - different warnings/errors
 - different manpages
 - different access variable (`set_distributed_variables`)



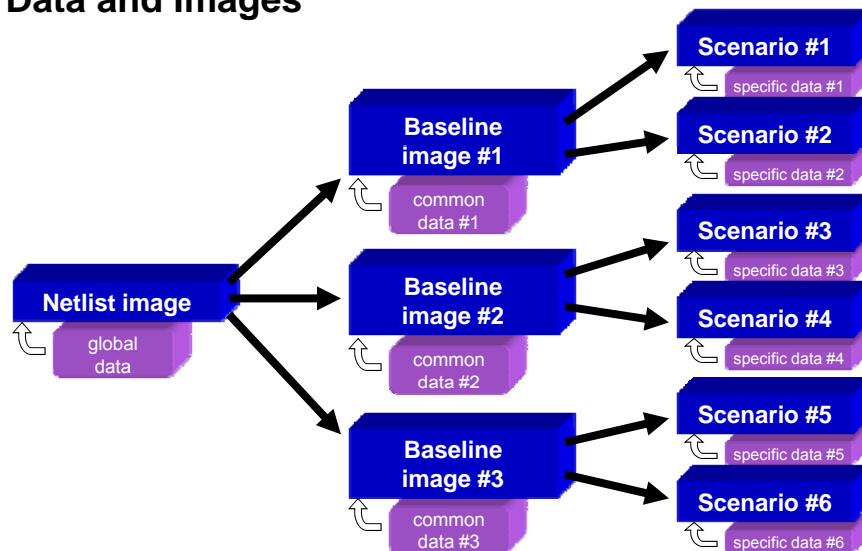
SolvNet Doc Id: 014511

Example Distributed Multi-Scenario Analysis (DMSA) Design

© 2007 Synopsys, Inc. (109) CONFIDENTIAL

SYNOPSYS
Predictable Success

Defining Scenario Data Data and Images



© 2007 Synopsys, Inc. (110) CONFIDENTIAL

SYNOPSYS
Predictable Success

Merged `report_timing`

- Paths from different scenarios can be merged into a single reporting file and duplicates can be removed if they are:
 - Same path group
 - Same sequence of pins along data portion of the path
 - Same transition directions at every pin along data path
 - Same launch clock
 - Same capture clock
 - Same constraint type

```
Startpoint: m_wb_ack_i (input port clocked by wb_clk_i)
Endpoint: core/wishbone/IncrTxPointer_reg
          (rising edge-triggered flip-flop clocked by wb_clk_i)
Path Group: inputs
Path Type: max
Scenario: func_wc func_bc
Max Data Paths Derating Factor : 1.00
Min Clock Paths Derating Factor : 0.95
Max Clock Paths Derating Factor : 1.00
```

© 2007 Synopsys, Inc. (111) CONFIDENTIAL

SYNOPSYS
Predictable Success

Current Limitations

- Tcl procedures are not saved by image swapping or sharing
 - Procedures should be defined in `.synopsys_pt.setup` so they are available to all remote processes
- Limited to 64 remote slave processes
- License count can be subsequently increased, but not decreased
 - This will be implemented in a future release
- Multiple multi-scenarios have to be run in different directories
 - logfiles, images in working directory would overlap

© 2007 Synopsys, Inc. (112) CONFIDENTIAL

SYNOPSYS
Predictable Success

Command Line Editing

Improves Usability

- Enables users to edit commands in pt_shell line
 - Similar to tcsh or ksh
 - Keystroke editing of command line (emacs or vi)
 - Up and Down arrow for history recall
 - Auto completion (Tab)
 - Commands, Options, File Names
- New variables
 - `sh_enable_line_editing` (true/false, default: false)
 - `sh_line_editing_mode` (emacs/vi, default: emacs)
- Limitations
 - Does not expand aliases

© 2007 Synopsys, Inc. (113) CONFIDENTIAL

SYNOPSYS
Predictable Success

Agenda

- Introduction
- Basic Flow
- Advanced Analysis
- Graphical User Interface (GUI)
- Back Annotation
- Modeling
- Usability
- *Signal Integrity Analysis (PT-SI)*
- Summary

© 2007 Synopsys, Inc. (114) CONFIDENTIAL

SYNOPSYS
Predictable Success

PrimeTime® SI

Extending PrimeTime with Signal Integrity Effects

PrimeTime SI

Static Timing
and
Signal Integrity
Sign-Off

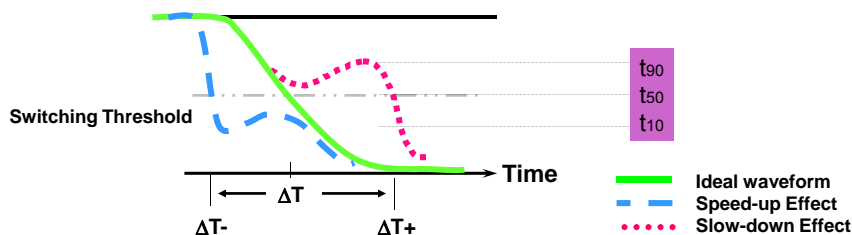
- Comprehensive SI Sign-Off Solution
 - Crosstalk Delay and Noise
 - Liberty Noise Libraries
- Industry-leading Performance/Capacity
 - Hierarchical SI
 - Multi-million gates in hours
- Integrated into Galaxy Design Platform
 - Common Delay Calculator
 - Astro ECO Flow

© 2007 Synopsys, Inc. (115) CONFIDENTIAL

SYNOPSYS
Predictable Success

Crosstalk Delay Impacts Timing

- Timing Failures
 - Crosstalk affects interconnect delay of victim net
 - Nets changing in the same direction: speed up
 - Nets changing in opposite directions: slow down
- Delay and Slew Effects
 - Modify the propagation and slew trip points



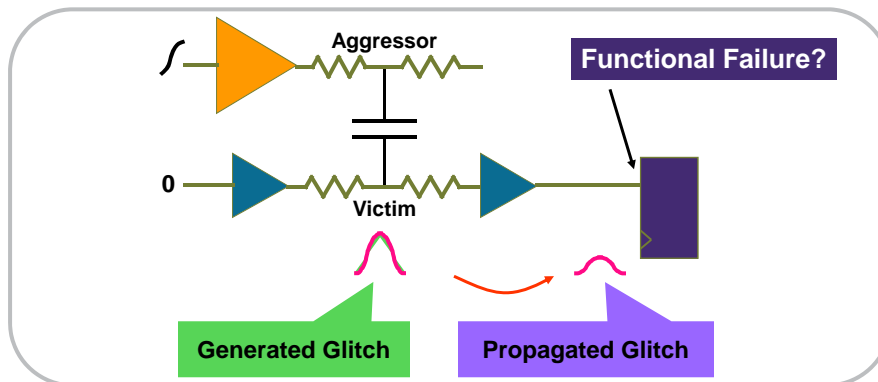
**Crosstalk analysis is required
to prevent timing failures**

© 2007 Synopsys, Inc. (116) CONFIDENTIAL

SYNOPSYS
Predictable Success

Crosstalk Noise (Glitch)

- Functional Failures
 - Crosstalk noise (glitch) can result in wrong values being captured at registers

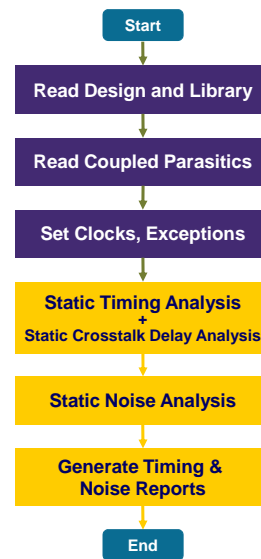
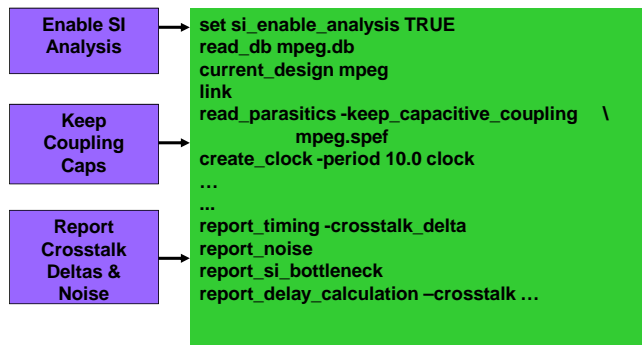


© 2007 Synopsys, Inc. (117) CONFIDENTIAL

SYNOPSYS
Predictable Success

PrimeTime® SI/Flow

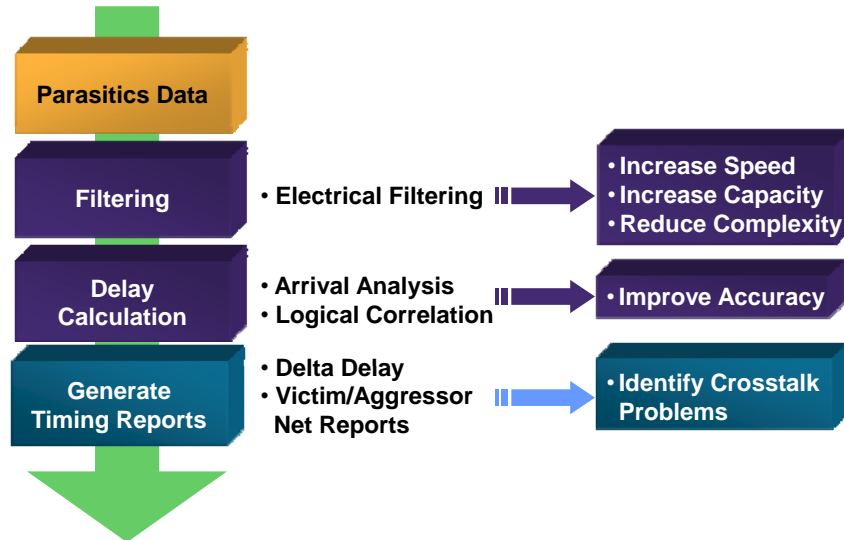
- Standard PrimeTime Flow
 - Same commands, libraries, Tcl scripts
- Outputs
 - Timing reports with crosstalk delta delays
 - Noise reports
 - SDF output
 - SPICE output



© 2007 Synopsys, Inc. (118) CONFIDENTIAL

SYNOPSYS
Predictable Success

PrimeTime® SI: Crosstalk Delay



© 2007 Synopsys, Inc. (119) CONFIDENTIAL

SYNOPSYS
Predictable Success

Timing Reports with Delta Delays

```

Path Group: clock
Path Type: max
Point          Fanout  Delta  Incr  Path
-----
clock clock (rise edge)          0.00  0.00
clock network delay (ideal)      0.00  0.00
hostif_0/host_address_regx3lx/CP (FD2S) 0.00  0.00 r
hostif_0/host_address_regx3lx/Q (FD2S)  0.47 & 0.47 f
hostif_0/U1569/D (NR4X05)          0.03  0.03 & 0.81 f
hostif_0/U1569/Z (NR4X05)          0.51 & 1.32 r
hostif_0/n5966 (net)              1
hostif_0/U1440/F (ND8P)            0.25  0.25 & 1.57 r
hostif_0/U1994/Z (ND3A)            0.24 & 6.20 r
hostif_0/n6400 (net)              1
hostif_0/taplink_tx_data_regx4x/D (FD2S) 0.02  0.02 & 6.22 r
clock clock (rise edge)          5.00  5.00
clock network delay (ideal)      0.00  5.00
hostif_0/taplink_tx_data_regx4x/CP (FD2S) 5.00 r
library setup time                -0.39  4.61
-----
data required time                4.61
data arrival time                 -6.22
-----
slack (VIOLATED)                 -1.61
  
```

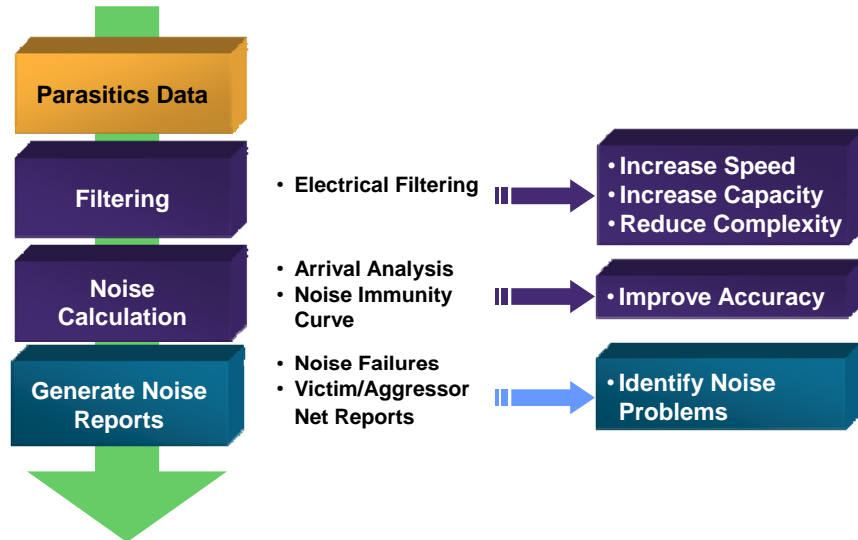
Delta Delay Column
(-crosstalk_delta)

Delay Introduced
by Crosstalk

© 2007 Synopsys, Inc. (120) CONFIDENTIAL

SYNOPSYS
Predictable Success

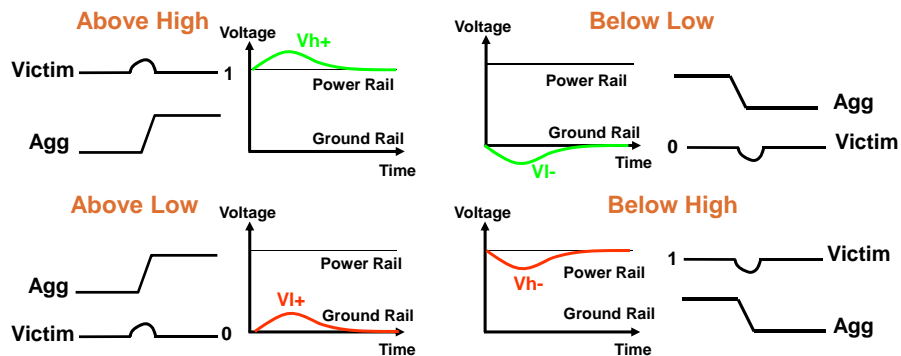
PrimeTime® SI: Crosstalk Noise



© 2007 Synopsys, Inc. (121) CONFIDENTIAL

SYNOPSYS
Predictable Success

Four Regions of Noise

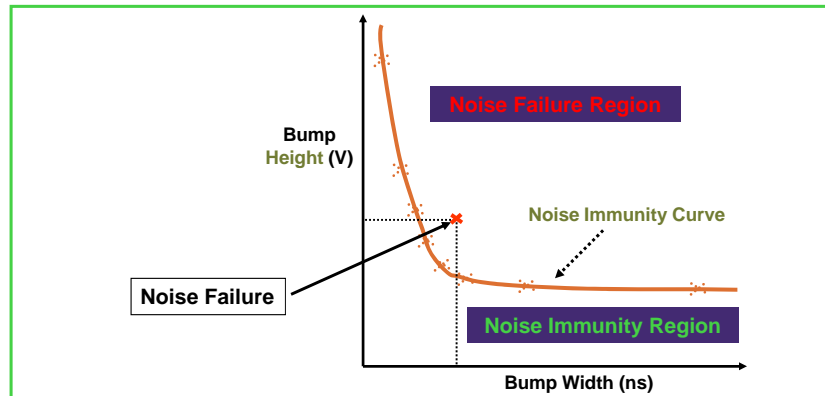


- PrimeTime SI analyzes all 4 regions
 - Bumps above low and below high can cause logic failure
 - Bumps below low and above high can forward-bias pass gates at the inputs of flip-flops and latches, allowing incorrect values to be latched

© 2007 Synopsys, Inc. (122) CONFIDENTIAL

SYNOPSYS
Predictable Success

Noise Immunity Curve (NIC)



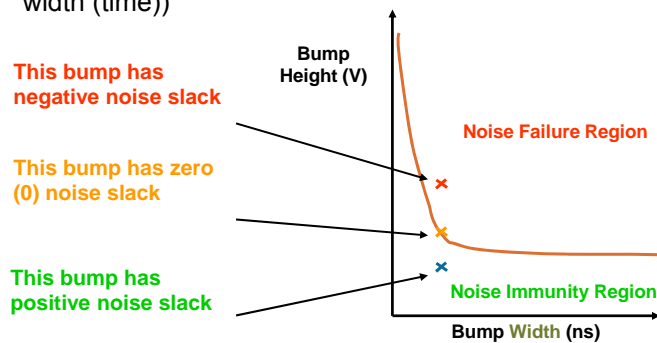
- Noise failure occurs when height and width of calculated noise bump falls into noise failure region
 - Noise immunity curve, in libraries characterized for noise, defines failure/immunity regions (or energy required to fail)

© 2007 Synopsys, Inc. (123) CONFIDENTIAL

SYNOPSYS
Predictable Success

Noise Slack

- PrimeTime SI reports noise slack
 - Noise slack is a metric to determine how far from passing or failing noise immunity constraint
 - Can be reported as height (voltage) or area (height (voltage) * width (time))



© 2007 Synopsys, Inc. (124) CONFIDENTIAL

SYNOPSYS
Predictable Success

Noise Report

report_noise -above -verbose -nosplit

```
*****
Report : noise
        -verbose
        -above
Design : cpu_core
Version: U-2003.03
Date   : Thu Feb 27 09:35:41 2003
*****
```

slack type: area

noise_region: above_low pin name (net name)	width	height	slack
rx_snt10g_frm78_prot_1/prot_rx_3/data_out_reg[27]/D (N43563)			
Aggressors:			
rst_prot_rx78_1_3_INBUF_103	0.80	0.00	
rx_snt10g_frm78_prot_1/prot_rx_3/poly_seq[4]	1.09	0.00	
rx_snt10g_frm78_prot_1/n30613_6	0.47	0.00	
Propagated:			
rx_snt10g_frm78_prot_1/U32709_C1/Z	0.38	0.87	
Total:	0.38	0.87	-0.12

Victim Net &
Cell Input Pin

Multiple
Aggressor Nets

Bump Width, Height,
& Noise Slack

© 2007 Synopsys, Inc. (125) CONFIDENTIAL

SYNOPSYS
Predictable Success

Enhanced SI Alignment Analysis Modes

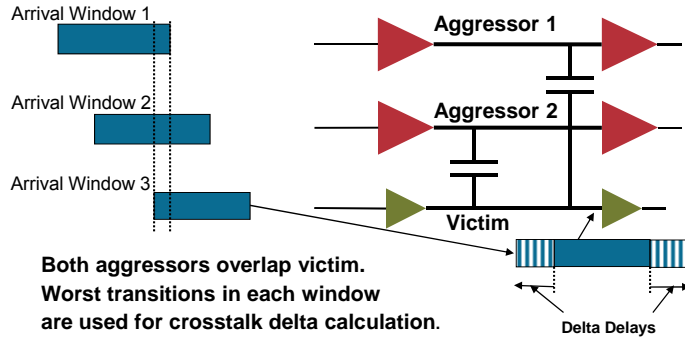
- PrimeTime® SI now has three usage modes:
 - all_paths** : (default) crosstalk for all paths. Same as pre PrimeTime® SI 2005.06 behavior.
 - all_violating_paths** : The crosstalk delta delay applies to all violating paths as well as the worst path.
 - worst_path** : enables calculating crosstalk for the worst arrival path.
- A new SI variable **si_xtalk_delay_analysis_mode** has been added to control these modes
 - set si_xtalk_delay_analysis_mode all_paths**
 - set si_xtalk_delay_analysis_mode all_violating_paths**
 - set si_xtalk_delay_analysis_mode worst_path**
 - Changing this variable will trigger a full **update_timing**

© 2007 Synopsys, Inc. (126) CONFIDENTIAL

SYNOPSYS
Predictable Success

Enhanced SI Alignment Analysis

all_paths (default) Mode



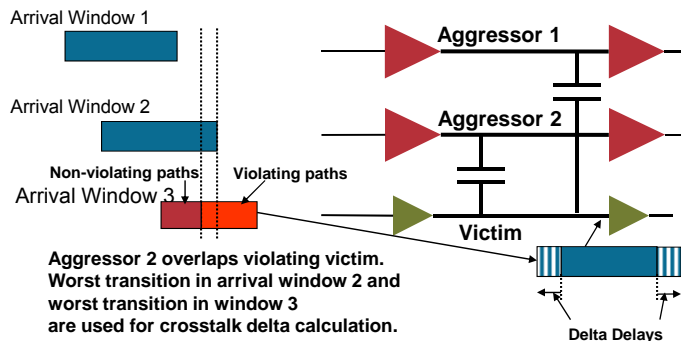
- For max analysis all victim paths will have delta delay and delta trans added.
- For min analysis (not shown in diagram) all paths will have delta delay and delta trans subtracted.

© 2007 Synopsys, Inc. (127) CONFIDENTIAL

SYNOPSYS
Predictable Success

Enhanced SI Alignment Analysis

all_violating_paths Mode



- Aggressor and Victim slews selected and propagated are same as *all_paths* mode.
- Delta delay may be smaller than *all_paths* mode.
- For max analysis all victim paths will have delta delay and delta trans added unless the failing victim region does not overlap any aggressor window.
- For min analysis (not shown in diagram) all paths will have delta delay and delta trans subtracted unless the failing victim region does not overlap any aggressor window.
- For unconstrained paths, clock paths or victim windows with no violators, the *worst_path* mode (see following slides) is used for alignment analysis.

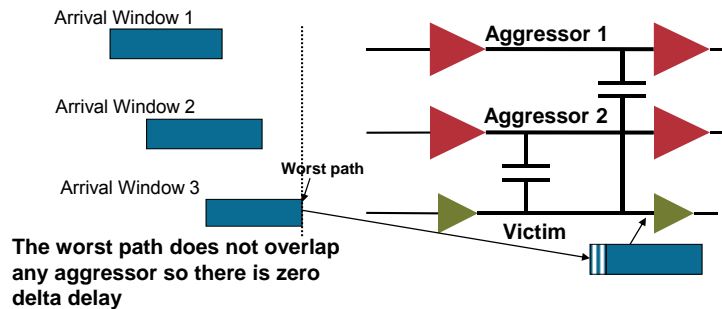
© 2007 Synopsys, Inc. (128) CONFIDENTIAL

SYNOPSYS
Predictable Success

Enhanced SI Alignment Analysis

worst_path Mode

Example



- All max paths through victim net will have a zero delta delay.
 - Only the worst path is guaranteed to be pessimistic
 - All sub-critical paths will have 0 delta delays added to them

© 2007 Synopsys, Inc. (129) CONFIDENTIAL

SYNOPSYS
Predictable Success

Incorporation into the Design Flow

- Suggested usage flow for the various modes:
 - Early phase of physical design when SI aware implementation tool does most fixing
 - PTSI : crosstalk delay for *all_paths*.
 - Late phase of physical design (ECO driven fixing and constraints are well defined)
 - PTSI : crosstalk for *all_violating_paths*.
 - Sign Off Runs : (to improve run time)
 - PTSI : crosstalk for *worst_paths* only.
 - Path based analysis could be used in combination with this mode.



SolvNet Doc Id: 015943

Frequently asked questions about PrimeTime SI's enhanced alignment modes

© 2007 Synopsys, Inc. (130) CONFIDENTIAL

SYNOPSYS
Predictable Success

PrimeTime SI/Enhanced Alignment Summary

- In **all_paths** mode, all paths through the victim net will have an accurate or pessimistic delta delay.
- In **all_violating_paths** mode, only violating paths through the victim net will have an accurate or pessimistic delta delay.
- In **worst_path** mode, only the worst violating path through the victim will have an accurate delta delay.

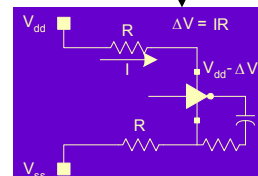
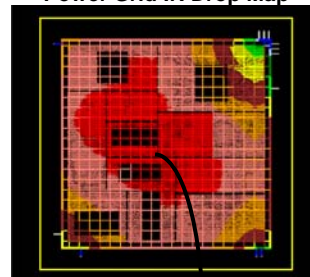
© 2007 Synopsys, Inc. (131) CONFIDENTIAL

SYNOPSYS
Predictable Success

IR (Voltage) Drop Delay Analysis

- IR Drop
 - Voltage drop on power grid
- Impact
 - Reduces supply voltages at cells
 - Changes delay across chip
 - 10% voltage drop reduces chip performance by 7-9%
- **Analysis** required to detect and prevent failures

Power Grid IR Drop Map



Source: Vivek, et.al., DAC 1998

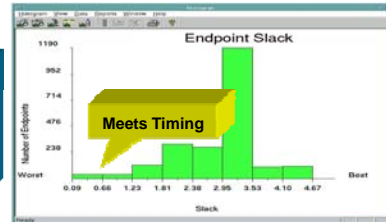
© 2007 Synopsys, Inc. (132) CONFIDENTIAL

SYNOPSYS
Predictable Success

Timing Effects of IR Drop

STA without IR Drop

PrimeTime

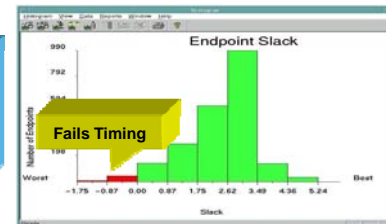


STA with IR Drop

Power Network Analysis
(Astro-Rail, PrimeRail)

Instance IR drop data

PrimeTime SI with IR Drop



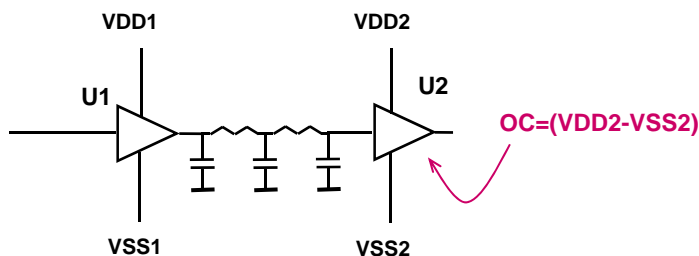
© 2007 Synopsys, Inc. (133) CONFIDENTIAL

SYNOPSYS
Predictable Success

IR Drop Delay Analysis using NLDM

- NLDM: k_{factor} are used
 - If IR Drop < 10% VDD, studies have shown that linear scaling with the k_{factor} is sufficient.

$$\text{Scaled Delay} = \text{Delay} \times (1 + K_{\text{volt}} (\Delta V))$$



© 2007 Synopsys, Inc. (134) CONFIDENTIAL

SYNOPSYS
Predictable Success

IR Drop Delay Analysis

- Single Voltage cell example:

```
set_rail_voltage -min -rail_value 5.8 u3
set_rail_voltage -max -rail_value 4.1 u3
set_rail_voltage -rail_value 3 u3
```

- Multiple Voltages cell example:

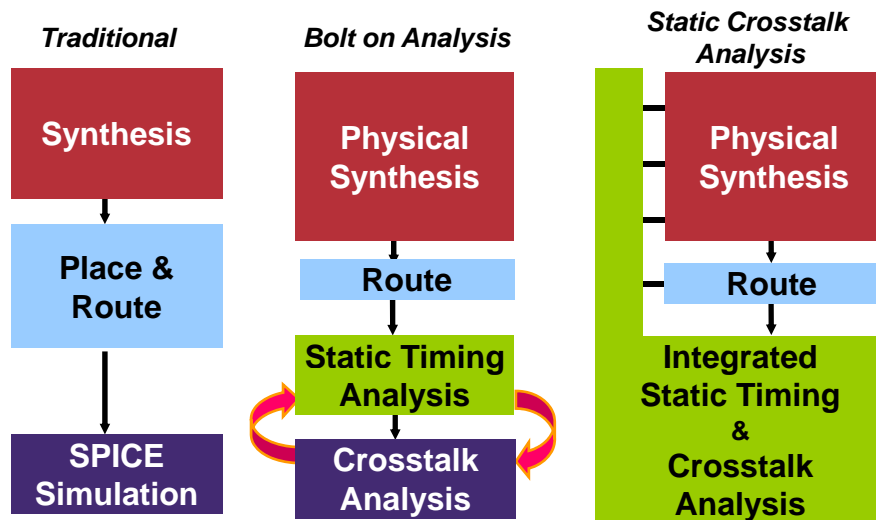
```
set_rail_voltage -min -rail_list {VDD0 1.4 VDD1 0.9 \
                                VDD2 3.5 VDD3 5.2} u3

set_rail_voltage -max -rail_list {VDD0 1.1 VDD1 0.7 \
                                VDD2 3.1 VDD3 4.8} u3
```

© 2007 Synopsys, Inc. (135) CONFIDENTIAL

SYNOPSYS
Predictable Success

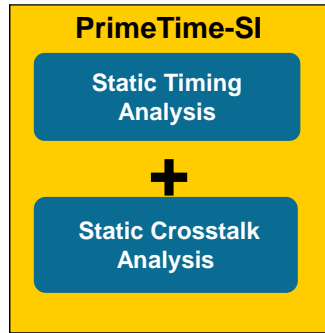
State of Timing Signoff Solutions



© 2007 Synopsys, Inc. (136) CONFIDENTIAL

SYNOPSYS
Predictable Success

Key PrimeTime Advantages



STA Technology with Crosstalk

- Based on proven STA technology
- Crosstalk analysis throughout the design process
- Integration with implementation flow
- Timing accurate
- Low cost adoption
- Multi-million gate capacity & performance

© 2007 Synopsys, Inc. (137) CONFIDENTIAL

SYNOPSYS
Predictable Success

Agenda

- Introduction
- Basic Flow
- Advanced Analysis
- Graphical User Interface (GUI)
- Back Annotation
- Modeling
- Usability
- Signal Integrity Analysis (PT-SI)
- *Summary*

© 2007 Synopsys, Inc. (138) CONFIDENTIAL

SYNOPSYS
Predictable Success

PrimeTime Summary

- Optimized for **fast**, **memory-efficient**, **chip-level** timing analysis
- Offers **advanced analysis** features for “system-on-a-chip” ASICs
- Offers **modeling** features for chip-level analysis
- Fully **integrated** with synthesis
- Integrated static timing with **signal Integrity analysis** (PT-SI)

© 2007 Synopsys, Inc. (139) CONFIDENTIAL

SYNOPSYS
Predictable Success

PrimeTime Documentation

- Synopsys On-line Documentation (**SOLD**):
 - PrimeTime User Guide: Fundamentals
 - PrimeTime User Guide: Advanced Timing Analysis
 - PrimeTime User Guide: Distributed Multiscenario Analysis
 - PrimeTime Modeling User Guide
 - PrimeTime SI User Guide
 - PrimeTime Tutorial
 - Using the Synopsys Design Constraints Format Application Note
- Tutorial Design:
 - [\\${SYNOPSYS}/doc/pt/tutorial](#)
- SolvNet (<https://solvnet.synopsys.com>)
 - Hundreds of technical articles on PrimeTime
 - Some useful SolvNet articles:
 - 1: Analyzing Configurable Clocks in PrimeTime (Methodology-35.html)
 - 2: PrimeTime Script to Help Assess Clock Tree (Synthesis-128.html)
 - 3: Guidelines for Static Timing Analysis (Methodology-89.html)
 - 4: Tcl script to report the slack distribution of a design (SYNTH-481797.html)
 - PrimeTime Application Notes now available on SolvNet
 - Search for “PrimeTime Application Note” (please see next slide)

© 2007 Synopsys, Inc. (140) CONFIDENTIAL

SYNOPSYS
Predictable Success

Application Notes (1)

Core Timing

- Location-Aware On-Chip Variation
- Accurate Signoff Using Path-Based Analysis
- What is the difference between single, bc_wc, and on_chip_variation analysis modes?
- Pulse-Generated Clock Creation with Non-Increasing Clock Edges
- PrimeTime Clock Reconvergence Pessimism Removal (CRPR)
- Transparent Latch Enhancements in PrimeTime
- Timing Model Guidelines for Latch Recognition
- Timing path analysis procedure for PrimeTime and PrimeTime SI
- Extensive Application Note on Clocks in PrimeTime
- Memory Usage/CPU Hints

Hierarchical STA

- Automated Timing Model Generation
- Hierarchical Static Timing Analysis Using ETMs
- Hierarchical Static Timing Analysis Using Interface Logic Models
- Performing arrival-tolerant block-level Analysis in PrimeTime SI

© 2007 Synopsys, Inc. (141) CONFIDENTIAL

SYNOPSYS
Predictable Success

Application Notes (2)

RC Delay Calculation

- Composite Current Source (CCS) Delay Calculation
- PrimeTime RC Delay Calculator Troubleshooting Guidelines
- Library Qualification Guidelines for PrimeTime
- Reference [1] - Understanding Delay Calculation with Detailed Parasitics in PrimeTime
- Reference [3] - Using PathMill to Characterize Library Cells
- Reference [4] -Constructing Better NLDs to Improve Timing Closure
- Delay Calculation White Paper
- How does PrimeTime translate between different trip points and voltages?
- Liberty Screener

PrimeTime SI

- PrimeTime SI and Astro-Xtalk Repair Flow for SI Timing Closure
- PrimeTime SI Noise Analysis With Missing or Incomplete Noise Library Data
- What is the si_noise_limit_propagation_ratio variable used for?

© 2007 Synopsys, Inc. (142) CONFIDENTIAL

SYNOPSYS
Predictable Success

Appendices

- Appendix A: [Reference Material](#)
- Appendix B: [TCL Primer](#)
- Appendix C: [ILM Flow](#)

Appendix A

Reference Material

PrimeTime Feature list

Advanced Timing Analysis

- Comprehensive timing checks
- On-Chip PVT Variation, clk re-convergence, Min/Max analysis, dynamic loop breaking
- Multi-phase and multi-frequency designs
- Time borrowing for latches, hold control
- Case and mode, plus constant propagation
- Check_timing, update_timing
- Automatic false path elimination
- Bus contention and float analysis
- Analysis coverage
- Pre-layout and post-layout flows

SoC Modeling and Extraction

- ILM for Hierarchical STA
- ETM for IP cores
- Quick Timing Model for top-down design

Signal Integrity Analysis

- Crosstalk Delay
- Crosstalk Noise (Glitch)

Std. Input and Output Formats

- .ddc, .db, Verilog, VHDL
- SDF, RSPF, DSPF, SPEF, SBPF
- SDF Path Constraints
- Verilog, VHDL
- Wire Load Models
- Synopsys Design Constraints (SDC)

Graphical User Interface

- Schematic, Profiler, Histogram, Waveform
- Visualize budgets and bottlenecks
- Linux (32- & 64-bit), Sun OS, Solaris, IBM

Timing Assertions and Exceptions

- Clocks : latency, skew, generated clocks
- Input and Output delay, port drive and load

Shell Interface

- Tcl-based, command help, attributes, collections, filtering, custom reporting

© 2007 Synopsys, Inc. (145) CONFIDENTIAL

SYNOPSYS
Predictable Success

Getting Help on Commands

- Use a wildcard to find a command:

```
pt_shell> help *clock
create_clock      # Create a clock object
create_generated_clock # Create a generated clock
object
remove_clock      # Remove a clock object
remove_generated_clock # Remove a generated_clock
object
```

- Use man pages to find detailed description of a command:

```
pt_shell> man create_clock

NAME
    create_clock    Creates a clock object.

SYNTAX
    string create_clock -period period_value
                        [-name clock_name] [-waveform edge_list]
                        [sources]

ARGUMENTS
    -period period_value
```

© 2007 Synopsys, Inc. (146) CONFIDENTIAL

SYNOPSYS
Predictable Success

Summary - File Formats

- PrimeTime reads:
 - Library : .db
 - Netlist : .ddc, Verilog, VHDL, Milkyway, .db
 - Timing Models : QTM, ETM, ILM
 - Annotated Delays : SDF
 - Detailed or Reduced Parasitics : SPEF, RSPF, DSPF
 - Synpsys Binary Parasitic Format: SBPF
- PrimeTime writes:
 - Scripts: pt_shell, dc_shell
 - Constraints: SDC
 - Timing Models: QTM, ETM, ILM
 - Annotated Delays : SDF
 - Path Constraints for Place & Route : SDF

© 2007 Synopsys, Inc. (147) CONFIDENTIAL

SYNOPSYS
Predictable Success

Appendix B

TCL Primer

© 2007 Synopsys, Inc. (148) CONFIDENTIAL

SYNOPSYS
Predictable Success

What is Tool Command Language (Tcl)?

- Scripting language created by John K. Ousterhout
- Offers many of powerful “C” style features
- Easy to use like a shell script
- References:



- *Tcl and the Tk Toolkit*, John K. Ousterhout
- *Practical Programming in Tcl and Tk*, Brent B. Welch
- <http://www.scriptics.com/>

Using Tcl in PrimeTime

- **Concept:**
 - The PrimeTime command language is based upon the Tool Command Language (Tcl) standard. Most emerging tools from Synopsys will be based upon Tcl for consistency.
 - While most of the timing commands and command options are identical to DC, the Tcl control language is quite different.
 - Tcl provides more consistency, power, and flexibility than the DC shell language.
 - There is a translator utility (transcript) that will convert DC timing scripts to the PrimeTime Tcl-based format (covered later in the tutorial).
 - Tcl contains a large amount of list, string and array processing functions.
 - Although Tcl is used in PrimeTime, Tk (the graphics toolkit) is not.
 - Recommend the following books to learn the details of Tcl:
 - “*Practical Programming in Tcl and Tk*” by Brent B. Welch, *Prentice-Hall*
 - “*Tcl and the Tk Toolkit*” by John K. Ousterhout, *Addison-Wesley*

Using Tcl in PrimeTime - Variables

- Using Variables:
 - Variables are set using the Tcl `set` command. PrimeTime will also support '=', but this is not standard Tcl.
 - Variables are dereferenced using '\$' before the variable name (like UNIX `cs`)
 - Variables are typed on demand, and can hold strings, lists, numeric values, etc.
 - Remove a variable using `unset varname`
 - Print variables using `printvar varname` or `echo $varname`
 - Arrays are supported - use `varname(index)`
 - Array indices can be strings - supports *associative arrays*.

© 2007 Synopsys, Inc. (151) CONFIDENTIAL

SYNOPSYS
Predictable Success

Using Tcl in PrimeTime - Variables

- **Example:**

```
pt_shell> set clock_period 10
10
pt_shell> echo $clock_period
10
pt_shell> echo "clock_period =" $clock_period
clock_period = 10
pt_shell> printvar clock_period
clock_period          = "10"
pt_shell> # A pound sign (#) indicates a comment
pt_shell> set index 1
1
pt_shell> set arr($index) "This is a string"
This is a string
pt_shell> echo $arr($index)
This is a string
pt_shell> set arr($index) 5
5
pt_shell> echo $arr($index)
5
```

© 2007 Synopsys, Inc. (152) CONFIDENTIAL

SYNOPSYS
Predictable Success

Using Tcl in PrimeTime Nesting Commands

- **Example:**

```
pt_shell> create_clock -per 10 [get_ports CLOCK]
1
pt_shell> set design_clock [get_clocks CLOCK]
_sel7
pt_shell> set mydelay 5
5
pt_shell> set_input_delay $mydelay -clock $design_clock [all_inputs]
1
pt_shell> query_objects [all_outputs]
{"INTERRUPT_DRIVER_ENABLE", "MAPPING_ROM_ENABLE", "OVERFLOW",
"PIPELINE_ENABLE", "Y_OUTPUT[10]", "Y_OUTPUT[11]", "Y_OUTPUT[12]",
"Y_OUTPUT[1]", "Y_OUTPUT[2]", "Y_OUTPUT[3]", "Y_OUTPUT[4]",
"Y_OUTPUT[5]", "Y_OUTPUT[6]", "Y_OUTPUT[7]", "Y_OUTPUT[8]",
"Y_OUTPUT[9]"}
```

Using Tcl in PrimeTime - Quoting

- **Example:**

```
pt_shell> echo "The value of mydelay is $mydelay"
The value of mydelay is 5
pt_shell> echo {The value of mydelay is $mydelay}
The value of mydelay is $mydelay
pt_shell> echo "5 + 4 is [expr 5+4]"
5 + 4 is 9
pt_shell> echo {5 + 4 is [expr 5+4]}
5 + 4 is [expr 5+4]
```

- **Note:**

- » Arithmetic expressions are evaluated using the **expr** command.

Objects and Collections

- **Concept:**

- Designs consist of objects:
 - Designs, Cells, Ports, Pins, Nets, Clocks
- PrimeTime has a set of collection commands that can select these objects - `get_objtype`, where `objtype` is one of the above.
- The collection commands are analogous to the DC `find` command



▪ **Note:** Collection commands return a collection handle, **NOT** a list or a string of the items! This is different from DC.

- The collection handle is a pointer to the selected items.
- Collection handles are much faster and more efficient to process than maintaining a list in memory.
- Collections must be saved, or they are discarded.
- To display the contents of a collection, use `query_objects`.

Objects and Collection

- **Example:**

```
pt_shell> help get_*; # Check out the collection commands
get_attribute      # Get the value of an attribute
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
get_lib_cells      # Create a collection of library cells
get_lib_pins       # Create a collection of library cell
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_qtm_ports      # Create a collection of QTM ports
get_timing_paths   # Create a collection of timing paths
get_unix_variable  # Procedure

pt_shell> set data_ports [get_port D[*]]
pt_shell> query $data_ports; # Can abbr. query_objects!
{"D[10]", "D[11]", "D[12]", "D[1]", "D[2]", "D[3]", "D[4]",
"D[5]", "D[6]", "D[7]", "D[8]", "D[9]}"
```

Objects and Collection

- Other important collection commands:

all_clocks - Get all clocks in the design (excluding generated clocks)

all_inputs - Get all input ports in the design

all_outputs - Get all output ports in the design

all_instances - Get all instances in the design

all_registers - Get all register cells in the design

all_connected - Get all connections for a pin, port, or net

© 2007 Synopsys, Inc. (157) CONFIDENTIAL

SYNOPSYS
Predictable Success

Objects and Collection

- *Example:*

```
pt_shell> foreach_in_collection outpin [get_port Y_OUTPUT[*]] {  
? set maxcap [get_attribute $outpin wire_capacitance_max]  
? set pinname [get_attribute $outpin full_name]  
? echo "Max capacitance of port $pinname is $maxcap"  
? }  
Max capacitance of port Y_OUTPUT[10] is 0.000000  
Max capacitance of port Y_OUTPUT[11] is 0.000000  
Max capacitance of port Y_OUTPUT[12] is 0.000000  
Max capacitance of port Y_OUTPUT[1] is 0.000000  
Max capacitance of port Y_OUTPUT[2] is 0.000000  
Max capacitance of port Y_OUTPUT[3] is 0.000000
```

- When entering a command, the “?” prompt indicates a continuation - PrimeTime is waiting for the rest of the command. This generally occurs when opening a string or statement with a quote (“), brace({) or square bracket([)

© 2007 Synopsys, Inc. (158) CONFIDENTIAL

SYNOPSYS
Predictable Success

Objects and Collection

- **Example:**

```
pt_shell> set all_but_clock [remove_from_collection \
[all_inputs] [get_port CLOCK]]
_sel19
pt_shell> query $all_but_clock
{"CARRY_IN", "CONDITION_CODE", "CONDITION_CODE_ENABLE",
"D[10]", "D[11]", "D[12]", "D[1]", "D[2]", "D[3]", "D[4]",
"D[5]", "D[6]", "D[7]", "D[8]", "D[9]", "INSTRUCTION[0]",
"INSTRUCTION[1]", "INSTRUCTION[2]", "INSTRUCTION[3]", "RELOAD"}
pt_shell> set_input_delay 5 -clock CLOCK $all_but_clock
1
pt_shell> set all_ports [add_to_collection [all_inputs] \
[all_outputs]]
_sel22
```

Objects and Collections



compare_collections

» Allows you to compare two collections



copy_collection

» Copy one collection into another



index_collection

» Obtain the 'n'th index into a collection (start from 0)

Examples:

```
pt_shell> compare_collection $col1 $col2; # '0' indicates equal
-1
pt_shell> set col2 [copy_collection $col1]
pt_shell> query_objects $col1
{"A", "B", "C", "D"}
pt_shell> query_objects [index_collection $col1 2]; # Starts at 0
{"C"}
```


Filtering Collections

- **Concept:**

- Filtering allows you to qualify your collections
- Similar to `filter` in `dc_shell`
- You can filter “on the fly” as collection is done with the `-filter` option of most collection commands
- You can also filter an existing collection using the `filter_collection` command
- `filter_collection` returns a new collection, or an empty string if no objects match your criteria
- It is more efficient to use `-filter`, as it filters as collection is occurring. This is useful for large collections (many thousands of items.)
- Filter expressions can include these operators:
`==, !=, >, <, >=, <=, =~`

Filtering Collections

- **Example:**

```
# List all the ND2, ND2I, ND3, ND4P, etc. cells in the hier
pt_shell> query [get_cells * -hier -filter "ref_name =~ ND*"]
{"U2/U113", "U2/U73", "U2/U78", "U2/U82", "U2/U84", "U2/U86", ...
pt_shell> set all_nets [get_nets * -hier]
_sel26
pt_shell> query [filter_collection $all_nets \
    "pin_capacitance_max > 20.0"]
{"CLOCK", "U1/CLOCK", "U2/CLOCK", "U3/CLOCK"}
pt_shell> query [get_lib_cell pt_lib/* -filter \
    "is_sequential == true"]
{"pt_lib/FD1", "pt_lib/FD1S", "pt_lib/FD2", "pt_lib/FD2S",
"pt_lib/FD4", "pt_lib/FD4S", "pt_lib/LD1"}
```

Working with Attributes

- **Example:**

```
pt_shell> list_attributes -application; # Good way to see all
attrs
pt_shell> get_attribute [get_clock CLOCK] period
pt_shell> get_attribute [get_design AM2910] temperature_max
Warning: Attribute 'temperature_max' does not exist on design
'AM2910' (ATTR-3)
pt_shell> set_operating_conditions -library pt_lib WCCOM
1
pt_shell> get_attribute [get_design AM2910] temperature_max
125.000000
pt_shell> get_attribute [get_design AM2910] voltage_max
4.500000
pt_shell> define_user_attribute pt_visited -type boolean \
-classes {cell net port}
pt_shell> set_user_attribute -class net {CLOCK} pt_visited true
Set attribute 'pt_visited' on 'CLOCK'
pt_shell> report_attribute [get_net CLOCK]
```

© 2007 Synopsys, Inc. (163) CONFIDENTIAL

SYNOPSYS
Predictable Success

Flexible Timing Reports

- Allows the user to define and create her own timing reports!
- Allows access to lower-level timing data!
- Timing report created by the user using power and flexibility of Tcl
- Interface to timing data through basic PrimeTime commands:
 - `select_timing_path -group -from -to -nworst ...`
 - `get_attribute attribute_name`
 - `select_path_group group_name` (optional)
- `select_timing_path` allows you to select one or more paths for further processing
- `get_attribute` then allows you to get a particular attribute value from the current timing path or path point
- `select_path_group` allows you to select one or more path groups

© 2007 Synopsys, Inc. (164) CONFIDENTIAL

SYNOPSYS
Predictable Success

Flexible Timing Reports

- Attributes you can obtain from `get_attribute` include:
 - Path points (a point along the selected path)
 - Slack
 - Arrival time
 - Required time
 - Min or max
 - Start/endpoint
 - Path type
 - Path group
 - Start/endpoint clock
 - Clock latency
 - Much more!
- We have written an identical `report_timing` command using flexible timing reports!

© 2007 Synopsys, Inc. (165) CONFIDENTIAL

SYNOPSYS
Predictable Success

Flexible Timing Reports

- Basic algorithm for writing a flexible timing report:

```
# Strongly suggest descriptive header!  
echo header information  
# Loop through desired timing paths  
foreach_selection path [select_timing_path args... paths...] {  
# Get desired timing data (via attributes) for selected path  
  set attr1 [get_attribute $path attr1_name]  
  set attr2 [get_attribute $path attr2_name]  
  ...  
  format timing report using $attrs  
}
```

Warning! Be sure flexible timing code is debugged and tested before relying on it for critical analysis! Synopsys cannot guarantee that user-created reports are accurate and complete. Always get the slack of the path to be sure.



© 2007 Synopsys, Inc. (166) CONFIDENTIAL

SYNOPSYS
Predictable Success

Flexible Timing Reports

- Example showing total negative and positive slack:

```
# Show Total Negative Slack, Total Positive Slack, Worst Negative Slack for any design
proc slackmaster {} {
  set design_tns 0
  set design_wns 100000
  set design_tps 0
  foreach_selection group [select_path_group *] {
    set group_tns 0
    set group_wns 100000
    set group_tps 0
    foreach_selection path [select_timing_path -nworst 10000 -group $group] {
      set slack [get_attribute $path slack]
      if {$slack < $group_wns} {
        set group_wns $slack
        if {$slack < $design_wns}
          set design_wns $slack
      }
    }
    if {$slack < 0.0} {
      set group_tns [expr $group_tns + $slack]
    } else {
      set group_tps [expr $group_tps + $slack]
    } # path
    set design_tns [expr $design_tns + $group_tns]
    set design_tps [expr $design_tps + $group_tps]
  } # group
  echo [format "WNS : %g\nTNS : %g\nTPS : %g" $design_wns $design_tns $design_tps]
}
```

© 2007 Synopsys, Inc. (167) CONFIDENTIAL

SYNOPSYS
Predictable Success

Paging Output from PrimeTime

- **Concept:**
 - » By default, output (reports, man pages, etc.) scrolls continuously from PrimeTime. You can cause PrimeTime to page the output, requiring the space bar to output each page. This is accomplished by setting the PrimeTime variable `sh_enable_page_mode` to "true".
- **Example:**
 - » Set this variable. Type:

```
pt_shell> set sh_enable_page_mode "true"
true
```

- If you wish the output to always be paged each time you use PrimeTime, set the `sh_enable_page_mode` variable in your `.synopsys_pt.setup` file.
- To view all the PrimeTime variables, type `printvar`.

© 2007 Synopsys, Inc. (168) CONFIDENTIAL

SYNOPSYS
Predictable Success

Appendix C

ILM Flow

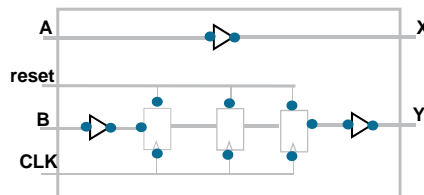
© 2007 Synopsys, Inc. (169) CONFIDENTIAL

SYNOPSYS
Predictable Success

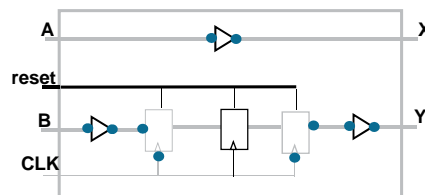
Identifying Interface Logic

- Identify the interface logic of a block
 - `identify_interface_logic` command tags interface logic in current design
- To ignore chip-level signals that fanout to internal registers
 - use `-ignore_ports` option

Attribute "is_interface_logic_pin"
is set to true on interface logic pins



```
pt_shell> identify_interface_logic
```



```
pt_shell> identify_interface_logic \  
? -ignore_ports {reset}
```

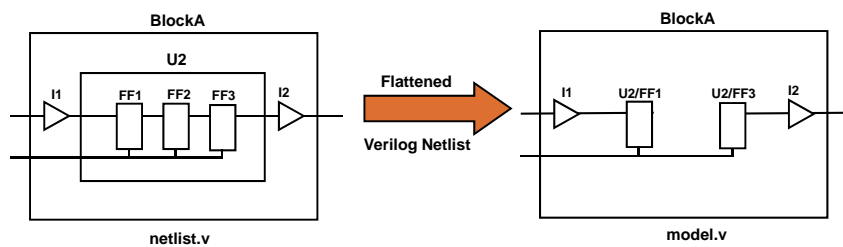
© 2007 Synopsys, Inc. (170) CONFIDENTIAL

SYNOPSYS
Predictable Success

Writing ILM netlist

- `write_ilm_netlist` command writes **flattened** verilog netlist for the ILM
 - Use `-include_all_net_pins` option, for the flow that requires delay calculation in PrimeTime

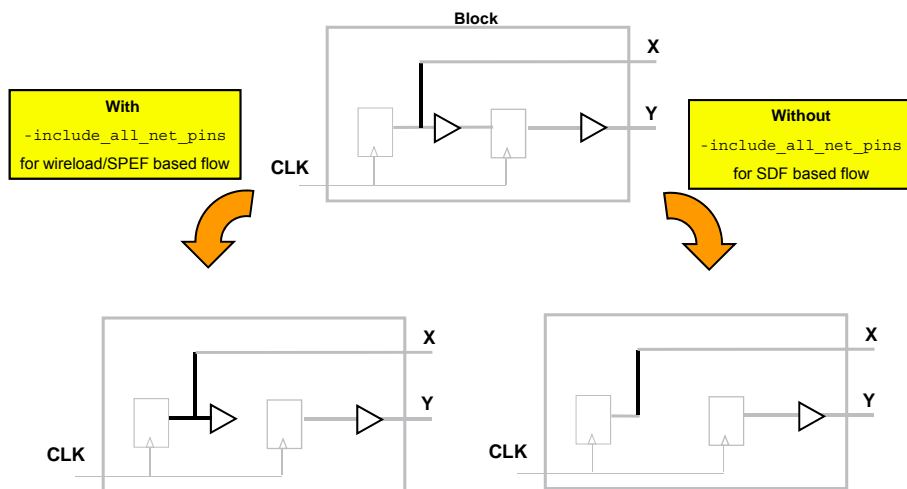
pt_shell> `write_ilm_netlist -include_all_net_pins model.v`



© 2007 Synopsys, Inc. (171) CONFIDENTIAL

SYNOPSYS
Predictable Success

Writing ILM netlist (cont.)



© 2007 Synopsys, Inc. (172) CONFIDENTIAL

SYNOPSYS
Predictable Success

Writing ILM scripts

- `write_ilm_script` command writes out script containing assertions & exceptions for the ILM

- Writing ILM script for **model verification**:

```
pt_shell> write_ilm_script model_verify.pt
```

- Writing ILM script for **top level timing analysis**:

```
pt_shell> write_ilm_script -instance model_analysis.pt
```

NOTE: When the `-instance` option is used with `write_ilm_script`, the boundary information for the block (`set_input_delay` and `set_output_delay`) is not included. Also, all the ports of the block are referenced with the `get_pin` command.

Writing ILM SDF/SPEF

- To write out SDF for the ILM:

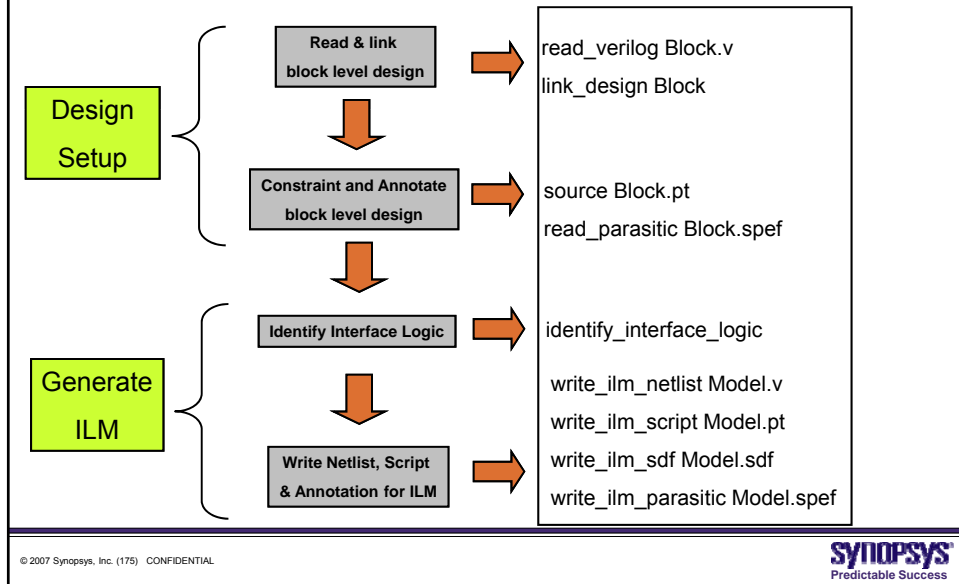
```
pt_shell> write_ilm_sdf model.sdf
```

- To write out parasitics for the ILM:

```
pt_shell> write_ilm_parasitics -input_port model.spef
```

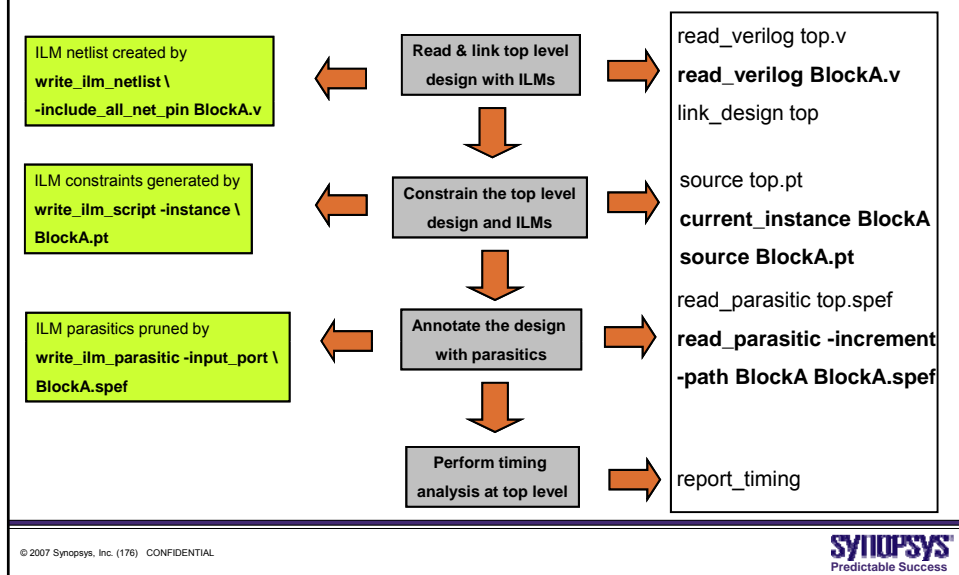
Note: PrimeTime writes out parasitics in SPEF format.

Generating ILMs

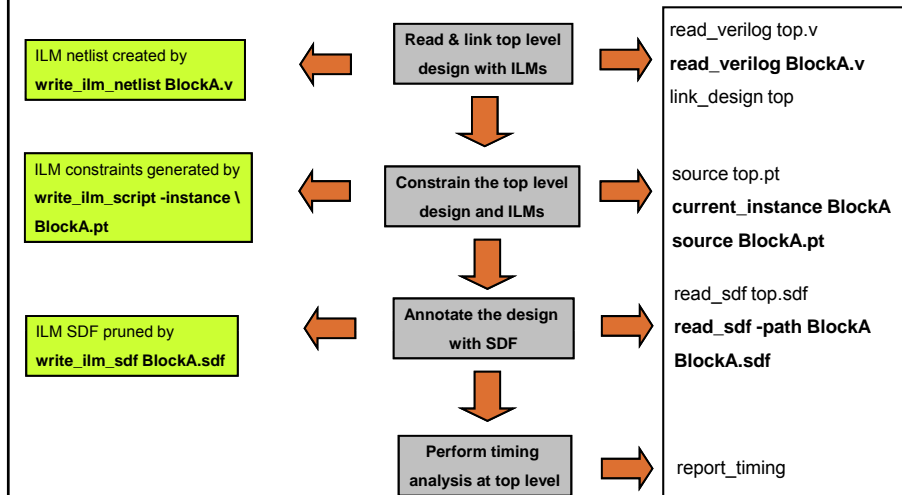


Hierarchical STA using ILMs

Parasitic Flow



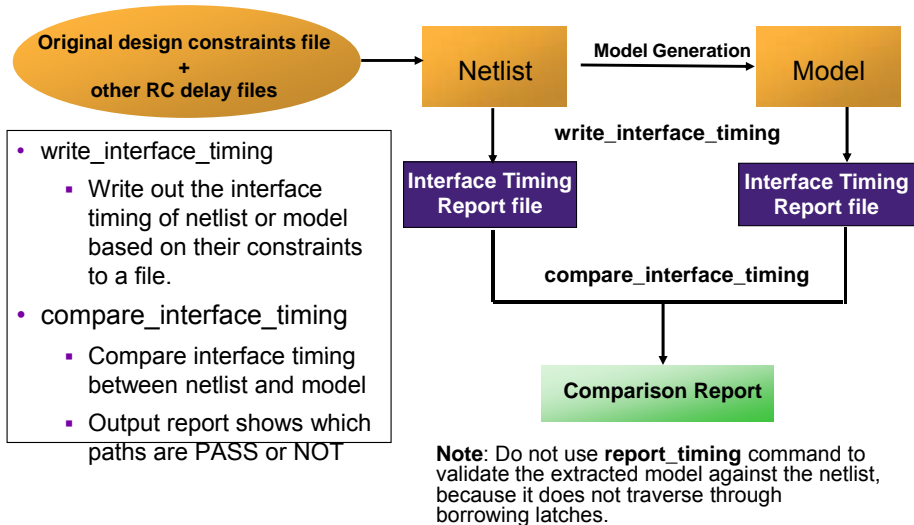
Hierarchical STA using ILMs SDF Flow



© 2007 Synopsys, Inc. (177) CONFIDENTIAL

SYNOPSYS
Predictable Success

ILM Validation



© 2007 Synopsys, Inc. (178) CONFIDENTIAL

SYNOPSYS
Predictable Success

Compressed ILMs

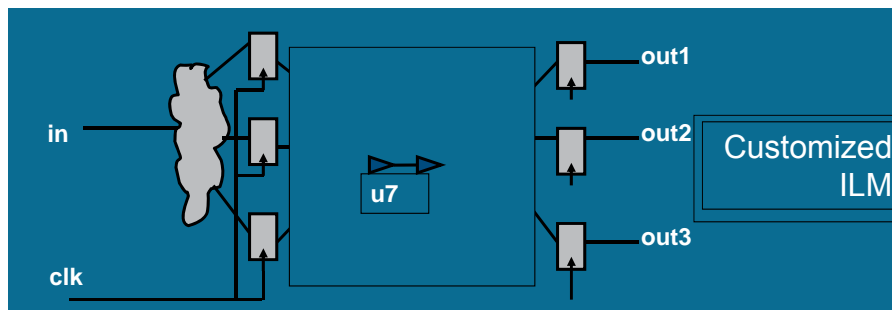
- Eliminate the logic that is not in the critical paths for the ILM in the current context
- New option with 'identify_interface_logic'
 - '-critical_pins' keeps only the interface logic associated with the critical paths
 - '-include_all_net_pins' option is recommended to be used with '-identify_interface_logic' when '-critical_pins' option for accuracy improvement
- All other ILM commands, write_ilm_(netlist/script parasitics/SDF) are unchanged
 - writes out the netlist/scripts/parasitics based on the identified pins like previous versions
- Disadvantage: ILM is context dependent with respect to top-level design from which it is created

© 2007 Synopsys, Inc. (179) CONFIDENTIAL

SYNOPSYS
Predictable Success

User control of ILM

- Sometimes user may want to retain portion of the netlist which is not part of the interface logic
- New option `-include_pins`
- User can only add pins



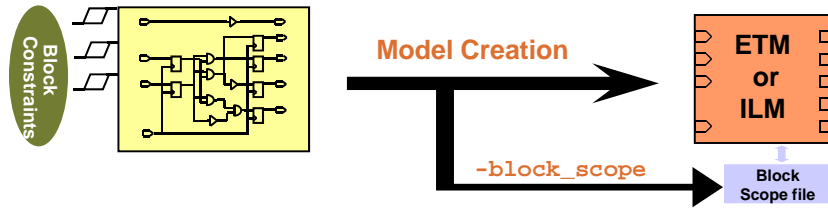
```
pt_shell> identify_interface_logic -include_pins {u7/Z} -include_all_net_pins
```

© 2007 Synopsys, Inc. (180) CONFIDENTIAL

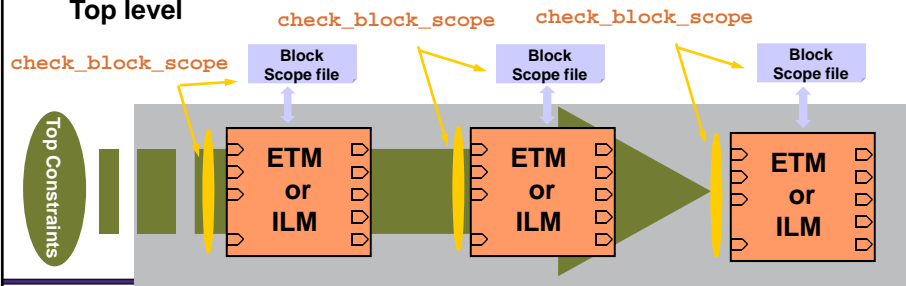
SYNOPSYS
Predictable Success

Checking the scope of a model

Block level



Top level



© 2007 Synopsys, Inc. (181) CONFIDENTIAL

SYNOPSYS
Predictable Success