Digital IC-Project and Verification

Static Timing Analysis (STA)

Liang Liu & Joachim Rodrigues

Outline

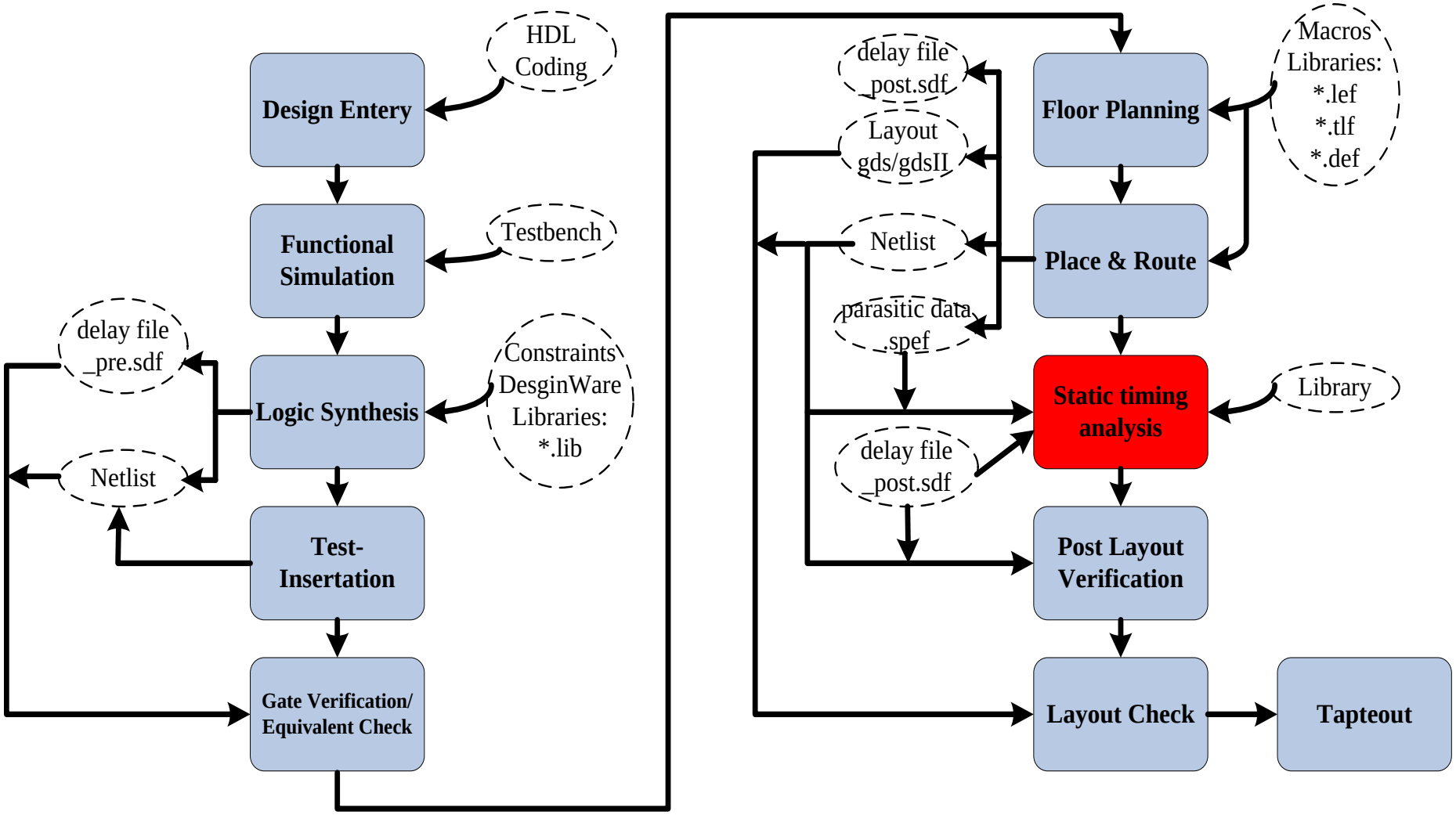
- STA & PrimeTime Overview
- STA Using PrimeTime
  - Basic Concepts
  - PrimeTime Flow
- Suggestions

Static Timing Analysis

- What's STA
  - STA is a method of validating the timing performance of a design by checking all possible paths for timing violations.

- Different with dynamical timing analysis
  - ***Full coverage***: removes the possibility that not all critical paths are identified
  - ***Higher speed***: especially for large complex designs
  - Slightly **pessimistic estimation**: e.g., wire load model

Static Timing Analysis

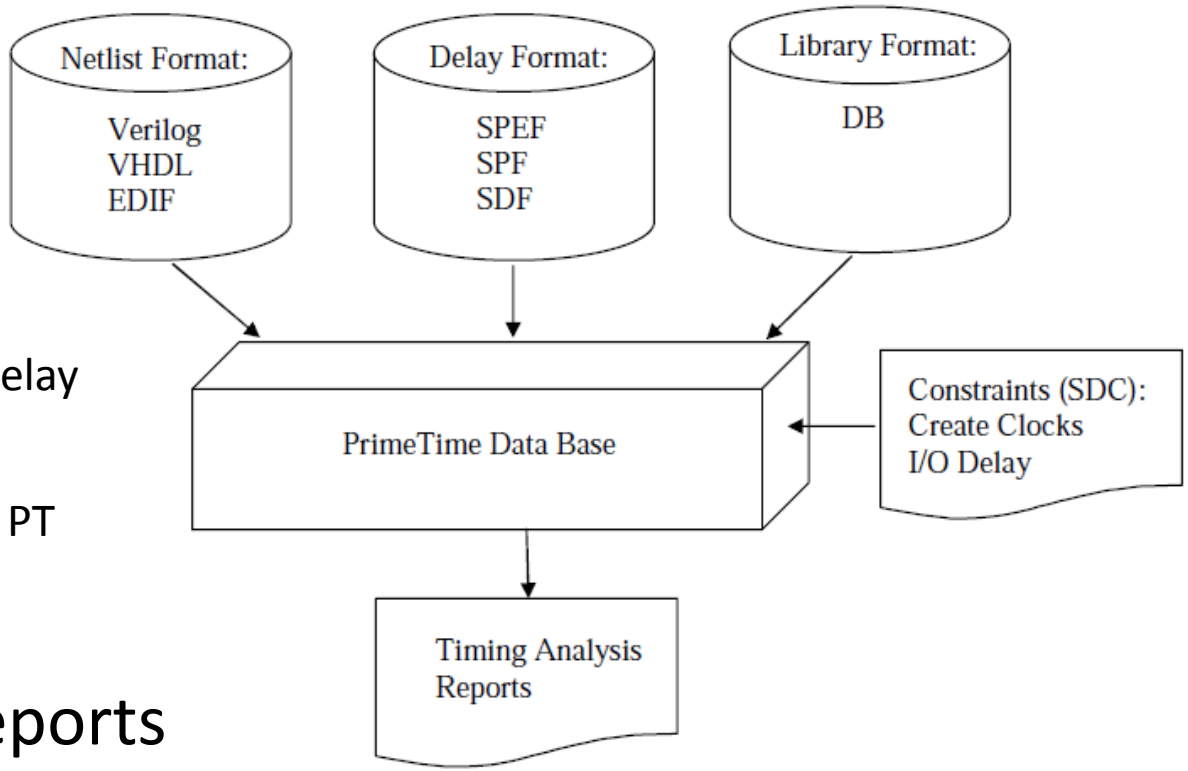


PrimeTime - Overview

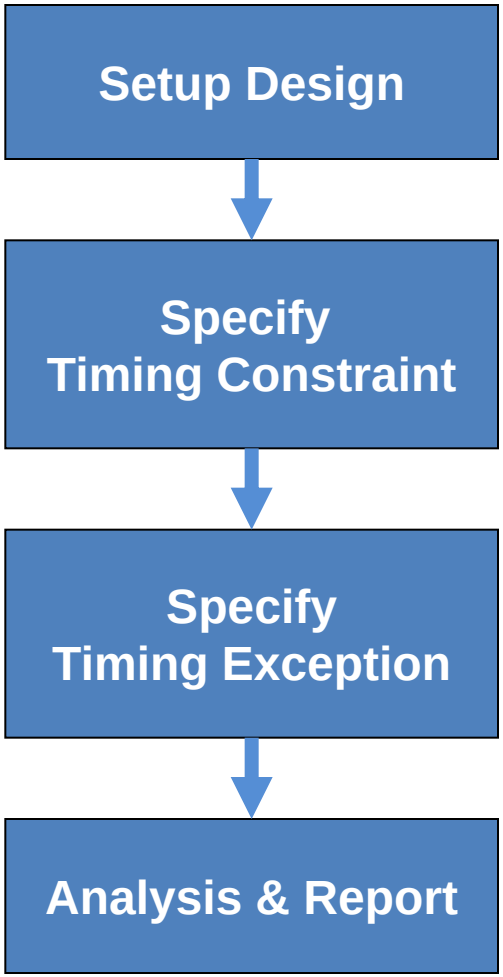
- PrimeTime is the ***Synopsys*** stand-alone full chip, gate-level static timing analyzer
- Widely-adopted in industry and academia, sign-off tools
- Controlled by Tool command language (TCL) compatible with DC

PrimeTime - Input/Output

- Inputs:
  - Netlist file
    - Verilog/VHDL/EDIF
  - Delay format:
    - SPEF/SPF/SDF
  - Database file (DB):
    - Determine the cell delay
  - SDC file:
    - Define the design to PT
- Outputs:
  - Timing Analysis Reports



PrimeTime - STA Flow



PrimeTime - Setup Design

- Set the search path and the link path

```
set search_path "lib path"
set link_library "* design.db"
set target_library "design.db"
```

- Read the design and the libraries

```
read_verilog top_level.v
current_design "top_level"
```

- Link the top design

```
link_design
```

PrimeTime - Timing Constraint

- Timing Violations
  - Setup violations happen when data changes less than t_{Setup} before the rising edge of the clock.
    - The ***maximum*** data path is used to check setup violations
  - Hold violations are similar to setup violations but data changes less than t_{Hold} after the rising edge of the clock.
    - The ***minimum*** data path is used to check hold violations

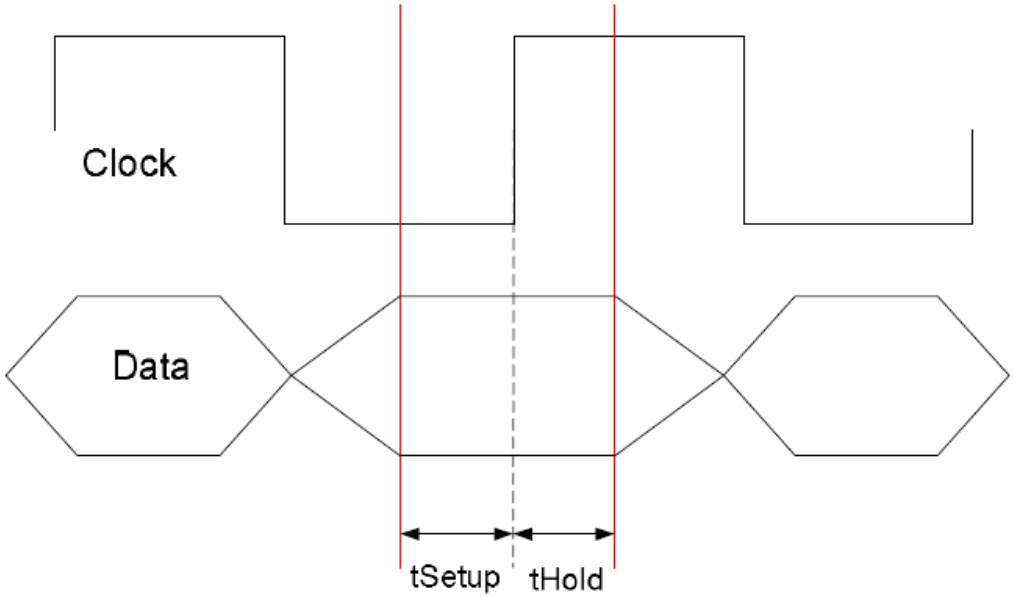
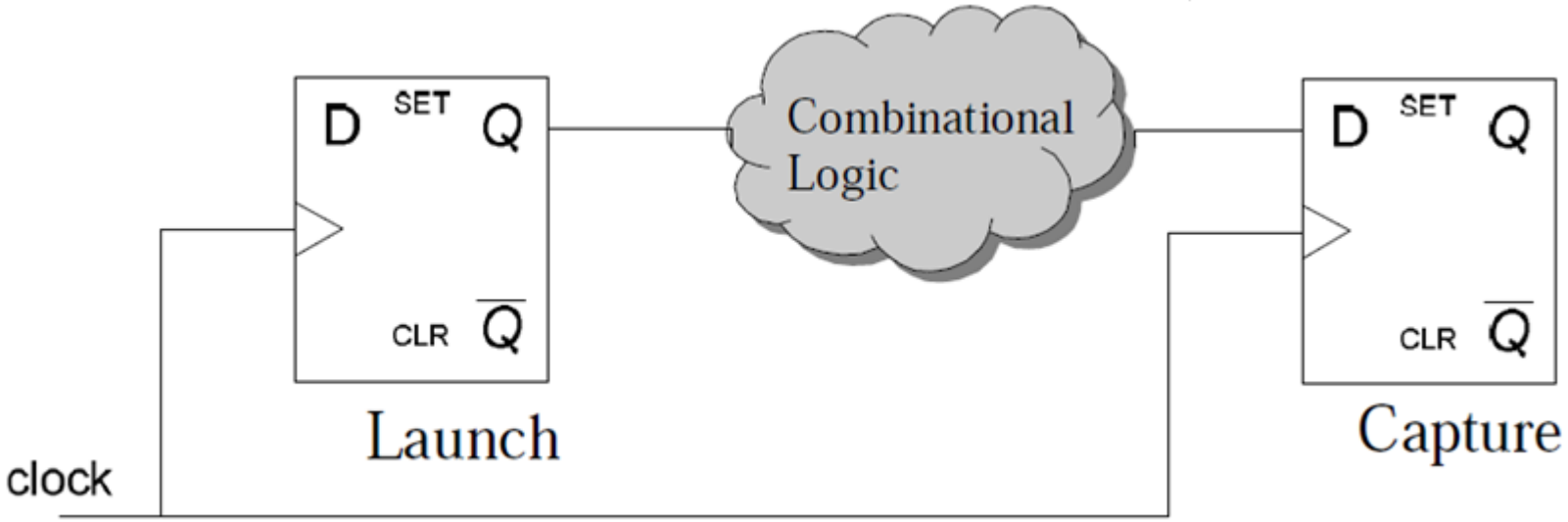


Figure from reference "PrimeTimeStatic Timing Analysis Tool", *George Michael*, 2006

PrimeTime - Timing Constraint

- Clock Period Constraint



$T_{Combinational\ logic} + FF_{lauch}(clk \rightarrow Q) < Clock\ Period - FF_{tSetup} - Clock\ Uncertainty$

$T_{Combinational\ logic} + FF_{lauch}(clk \rightarrow Q) > FF_{tHold} + Clock\ Uncertainty$

PrimeTime - Timing Constraint

- Clock Period Constraint (script)

```
create_clock -period 2 [get_ports clk_in]
```

define a clock with a frequency of 500 MHz or 2ns period in PrimeTime

```
set_clock_uncertainty # [get_clocks clk_in]
```

define delay between the clock branches (skew). For pre-layout

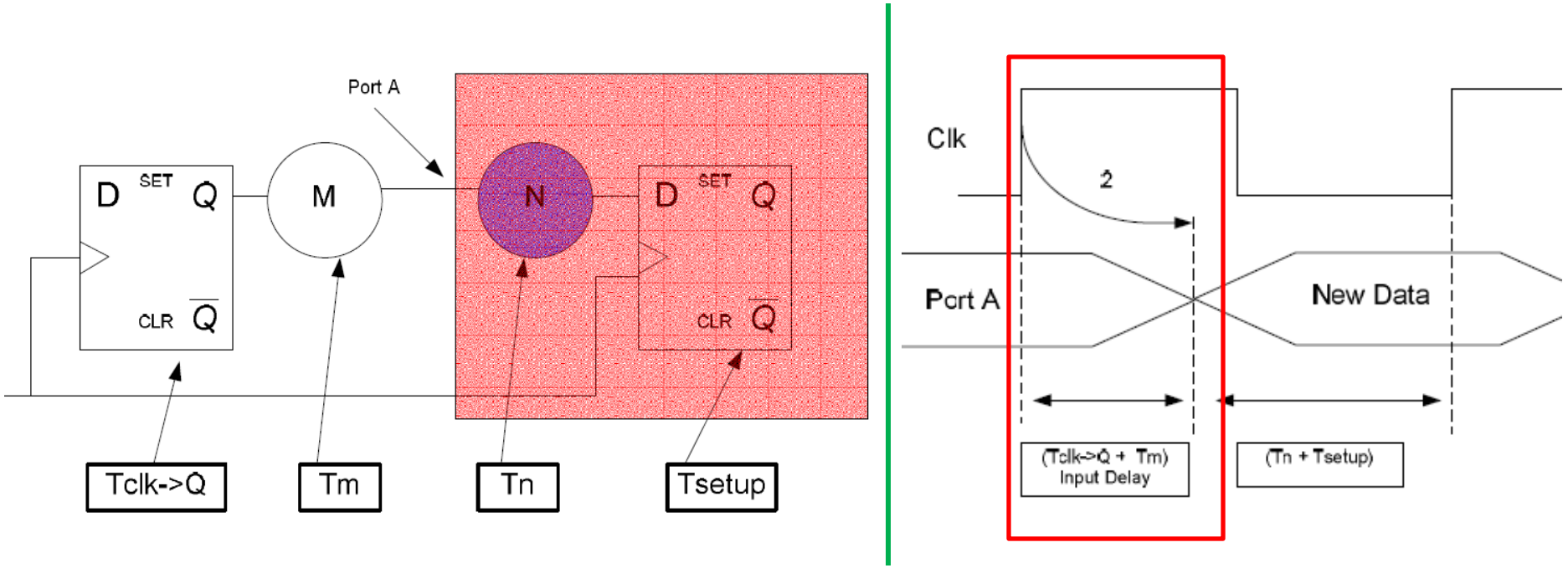
```
set_propagated_clock [all_clocks]
```

specifies that PrimeTime realized the latency for each clock path. This command should be used during post route analysis.

```
read_sdc top_level.sdc
```

PrimeTime - Timing Constraint

- Input Delay
  - Specify the delay of external logic driving current design



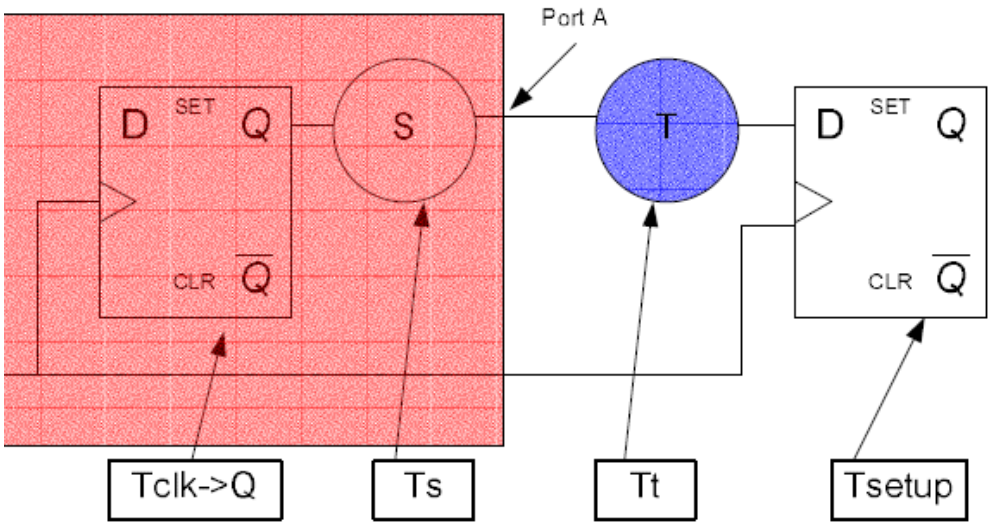
Script:

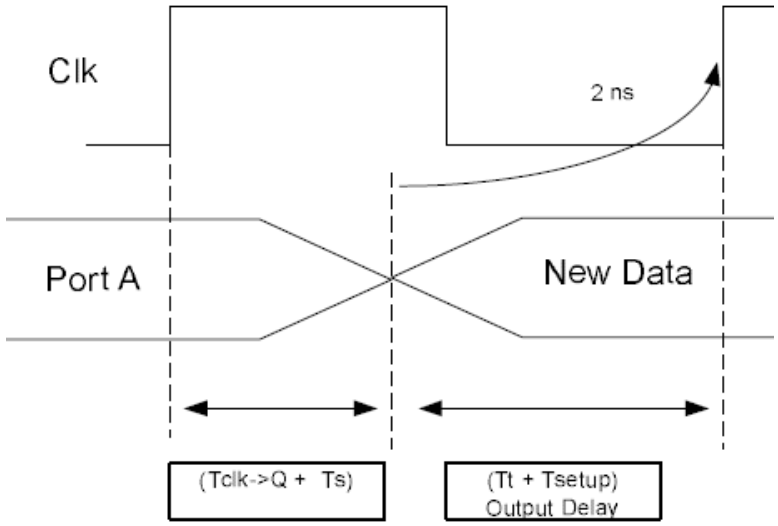
```
set_input_delay -clock clk_in -max #[get_ports i_*]
```

Figure from reference “PrimeTimeStatic Timing Analysis Tool”, *George Michael*, 2006

PrimeTime - Timing Constraint

- Output Delay
  - Specify the delay of external logic driven by current design





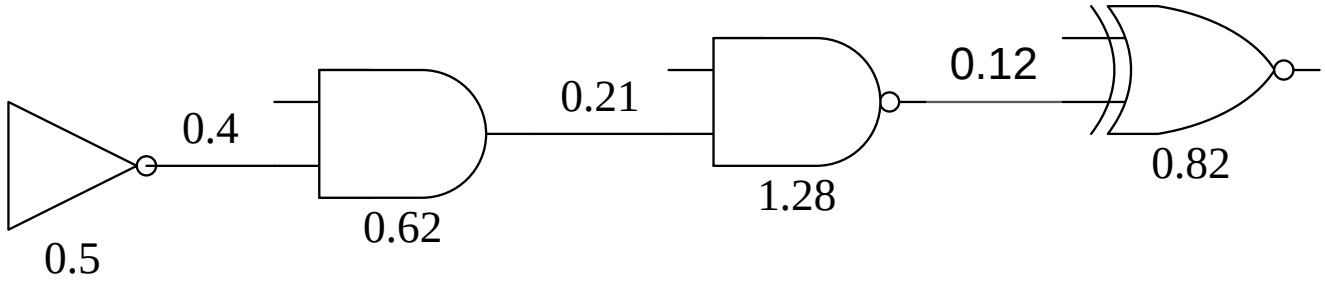
Script:

```
set_output_delay -clock clk_in -max #[get_ports O_*]
```

Figure from reference “PrimeTimeStatic Timing Analysis Tool”, *George Michael*, 2006

PrimeTime - Path Delay Calculation

- path delay = cell delay + net delay



Path Delay = 0.5+0.4+0.62+0.21+1.28+0.12+0.82=3.95 ns

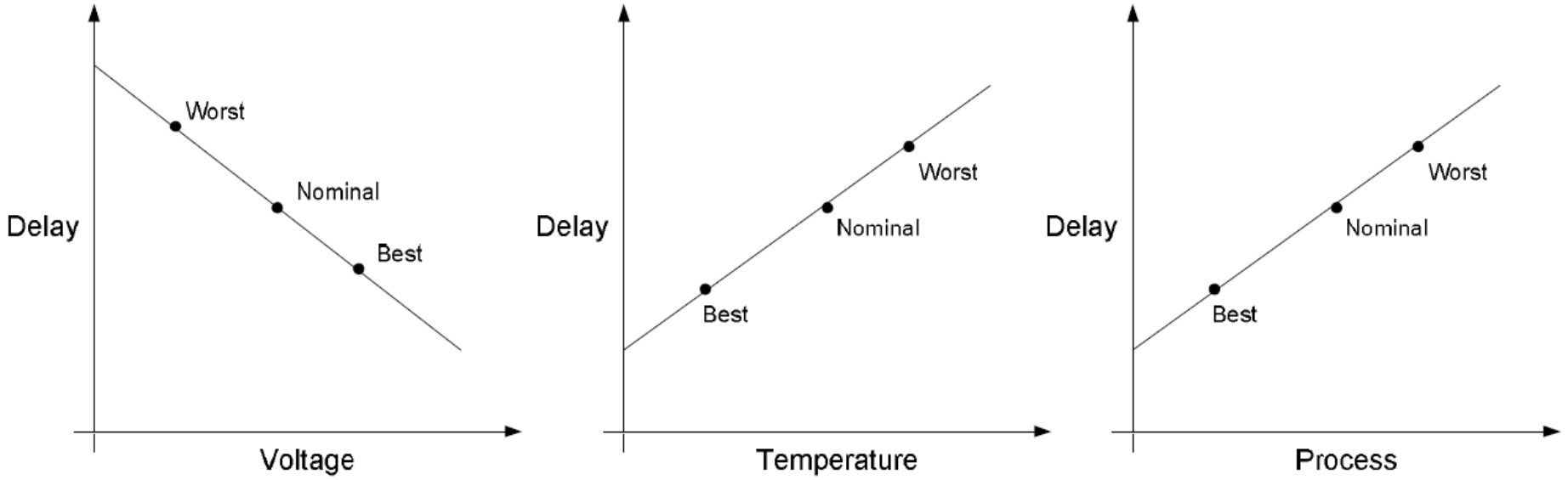
- Cell delay is stored in files called Synopsys database files or db files. Database files are read into PrimeTime by the link_path variable
- Net delay is stored in sdf file (post-layout) or calculated by PrimeTime by an internal delay calculator (pre-layout).

Script:

```
set link_library "*.db"
read_parasitics -format SPEF top_level.spef.gz
read_sdf top_level.sdf
```

PrimeTime - Working Condition

- Best Case v.s. Worst Case



  - Use the worst case delay when testing for setup violations
  - Use the best case delay when testing for hold violations

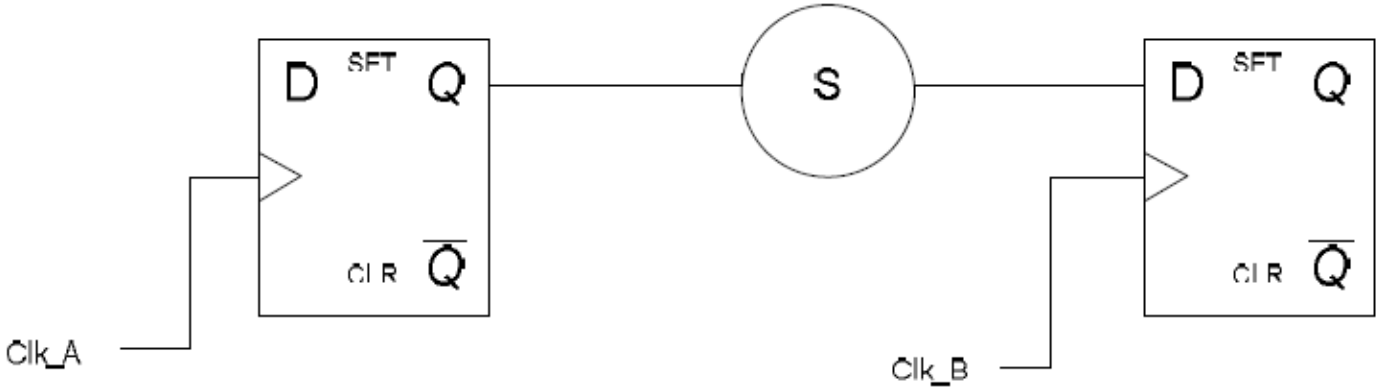
Script:

```
set_operating_conditions <worst/best-case>
```

Operating condition is defined in library

PrimeTime - Timing Exception

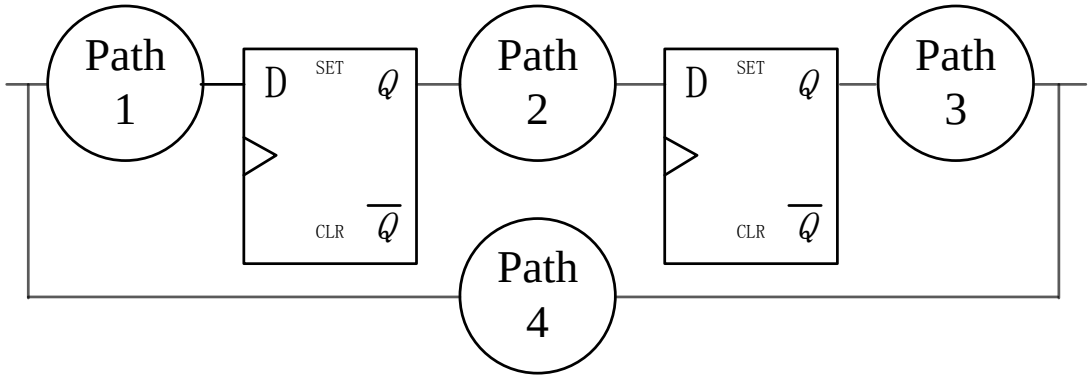
- False Path
  - paths in a design were a designer would not want the timing arcs to be calculated
    - Paths not relevant to functional operation of the circuit
    - paths which are impossible to exercise
    - Paths cross different clock domains




```
set_false_path -from clk_a -to clk_b
set_false_path -from clk_b -to clk_a
```

PrimeTime - Generating Reports

- Report Timing
  - To reduce the size and complexity of the PrimeTime reports, it is recommended to break the design into groups



P1: input to reg
P2: reg to reg
P3: reg to output
P4: input to output

```
report_timing   -from [all_registers -clock_pins]
                      [all_inputs]
                -to   [all_registers -data_pins]
                      [all_outputs]
```

PrimeTime - Generating Reports

- Report Timing (continued)

`report_timing -from -to`

If this commands is not used PrimeTime will default to the longest path (critical path) in the design

`-path full_path`

This option reports not only the data path but the launching and capturing clock path. `Set_propagated_clocks` must be set for this option to properly report the clock paths.

`-delay {max|min}`

max: PrimeTime reports setup time/min: PrimeTime reports hold time

`-max_paths`

This variable states the total number of paths to be reported per group. The default is one.

PrimeTime - Generating Reports

- Report Violation

```
report_constraints -all_violators
```

This command generates a summary of all paths that are violation setup and hold times as well as and any cells that violation a design rule such as fanout, capacitance, and transition. Viewing this one report will tell you if changes will need to be made to your design.

PrimeTime - Generating Reports

- Report Clock Timing

```
report_clock_timing -type skew -verbose
```

This command will report clock skew, the difference between the longest and shortest clock insertion time, and allow the design to evaluate whether or not the clock tree must be re-synthesized. This is a powerful command can save the designer from numerous timing closure spins.

PrimeTime - setup Reports

Point	Incr	Path	
clock tck (rise edge)	0.00	0.00	
clock network delay (ideal)	0.00	0.00	
input external delay	15.00	15.00	r
tdi (in)	0.00	15.00	r
pads/tdi (pads)	0.00	15.00	r
pads/tdi_pad/Z (PAD1X)	1.32	16.32	r
pads/tdi_signal (pads)	0.00	16.32	r
ir_block/tdi (ir_block)	0.00	16.32	r
ir_block/U1/Z (AND2D4)	0.28	16.60	r
ir_block/U2/ZN (INV0D2)	0.33	16.93	f
ir_block/U1234/Z (OR2D0)	1.82	18.75	f
ir_block/U156/ZN(NOR3D2)	1.05	19.80	r
ir_block/ir_reg0/D (DFF1X)	0.00	19.80	r
data arrival time		19.80	

clock tck (rise edge)	30.00	30.00	
clock network delay (ideal)	2.50	32.50	
ir_block/ir_reg0/CP (DFF1X)		32.50	r
library setup time	–0.76	31.74	
data required time		31.74	
data required time		31.74	
data arrival time		–19.80	
slack (MET)		11.94	

PrimeTime - hold Reports

Point	Incr	Path	
clock tck (rise edge)	0.00	0.00	
clock network delay (propagated)	1.92	1.92	
state_block/st_reg9/CP (DFF1X)	0.00	1.92	r
state_block/st_reg9/Q (DFF1X)	0.18	2.10	r
state_block/U15/Z (BUFF4X)	0.04*	2.14	r
state_block/bp_reg2/D (DFF1X)	0.06*	2.20	r
data arrival time		2.20	
clock tck (rise edge)	0.00	0.00	
clock network delay (propagated)	1.54	1.54	
state_block/bp_reg2/CP (DFF1X)		1.54	r
library hold time	0.50	2.04	
data required time		2.04	
data required time		2.04	
data arrival time		–2.20	
slack (MET)		0.16	

Some suggestions

- Notes/comments are even more important

```
//-------------------------------------------------
// Design      : CARRIER SENSE
// File Name   : CARRIER_SENSE.v
// Purpose     : Model of CARRIER SENSE process in PCS (IEEE Std 802.3)
// Limitation  : none
// Errors      : none known
// Include Files: none
// Author      : Liang Liu, liang.liu@eit.lth.se, Lund University
// Simulator   : ModelSim 6.5
//-------------------------------------------------
// Revision List:
//+-----------+------------------+----------------+------------------------------+
//| Version  | Author           | Date           | Changes                      |
//+-----------+------------------+----------------+------------------------------+
//|1.0       |     Liang Liu    | 2001/08/03  | original created             |
//| 1.0      |     Liang Liu    | 2002/01/04  |disable TX_EN to CRS      |
//|          |                  |                |in repeater mode          |
//+-----------+------------------+------ ----------+------------------------------+
```

Some suggestions

- Name the files/signals

File name:

module file starts with “m_”, test bench starts with “tb_”

e.g., netlist name “syn_” for DC out, “pr_” for Encounter out

Signals:

inputs starts with “i_”

outputs starts with “o_”

clocks starts with “clk_”

resets starts with “rst_”

register out put ends with “_r”

low-valid signal ends with “_n”, e.g., rst_n

Some suggestions

- Pre-/Post layout design

Getting experienced by comparing pre- and post- layout design

Set reasonable timing margin to avoid **LARGE** loop in design flow, e.g.,

- clock_uncertainty : justify the value with post-layout report
- clock_period: post-layout period= pre-layout period+margin, depending on process technology

Meet timing requirement as early as possible

- keep in mind the delay information when design the circuits, e.g., pipeline schedule, parallel, et. al.
- set reasonable constraint for synthesis
- optimize the design at early stage of P&R

Sources

- http://www.eit.lth.se/index.php?ciuid=647&coursepage=3553
- SolvNet (support from Synopsys) https://solvnet.synopsys.com/ documents (user guide), online case, et.al.
- Google
- Man *command*

To start PrimeTime

- Change to the folder where you want to run PrimeTime, and execute

inittde dicp13 (more info @: www.eit.lth.se/cadsys/far130lnx.html)

- Initializes the environment and copies some setup files (if required)

- CAD tools initialization script creates several directories

  (good directory structure for project management)

  - /Desktop/project_name
  - netlists (.v, .sdf, .spef, .sdc)
  - reports (setup.rpt, hold.rpt, violate.rpt, skew.rpt)
  - scripts (.tcl, .run)
  - Readme.txt
- Execute ***pt_shell -64bit*** in the same terminal as inittde was executed
- Start_gui

Lab

- http://www.eit.lth.se/index.php?ciuid=647&coursepage=3553
- Download provided input-files for PrimeTime (readme.txt)
- Fill the template *medianfilter_pt_timing.tcl* & run PrimeTime

Digital IC-Project and Verification

Power Analysis

Liang Liu & Joachim Rodrigues

Outline

- Power Dissipation
- Power Analysis Using PrimeTime PX
  - Power analysis requirement
  - PrimeTime PX Flow
- General PrimeTime PX Script

Power Dissipation

CMOS Power = static power + dynamic power

- Static Power: $V*I_{leak}$
  - source-to-drain sub-threshold ***leakage current***
  - depend on voltage, temperature, transistor state …

- Dynamic Power: switching power + internal power
  - ***switching power*** = $½*(C_{int}+C_{load})*V^2*f$
  - ***short-circuits power*** = $V*I_{sc}$
  - ***f***: state transition rate/I_{sc}: short-circuits current/**C_{load}**: total load capacitance/$C_{int:}$ internal capacitance

Power Dissipation

CMOS Power = static power + dynamic power

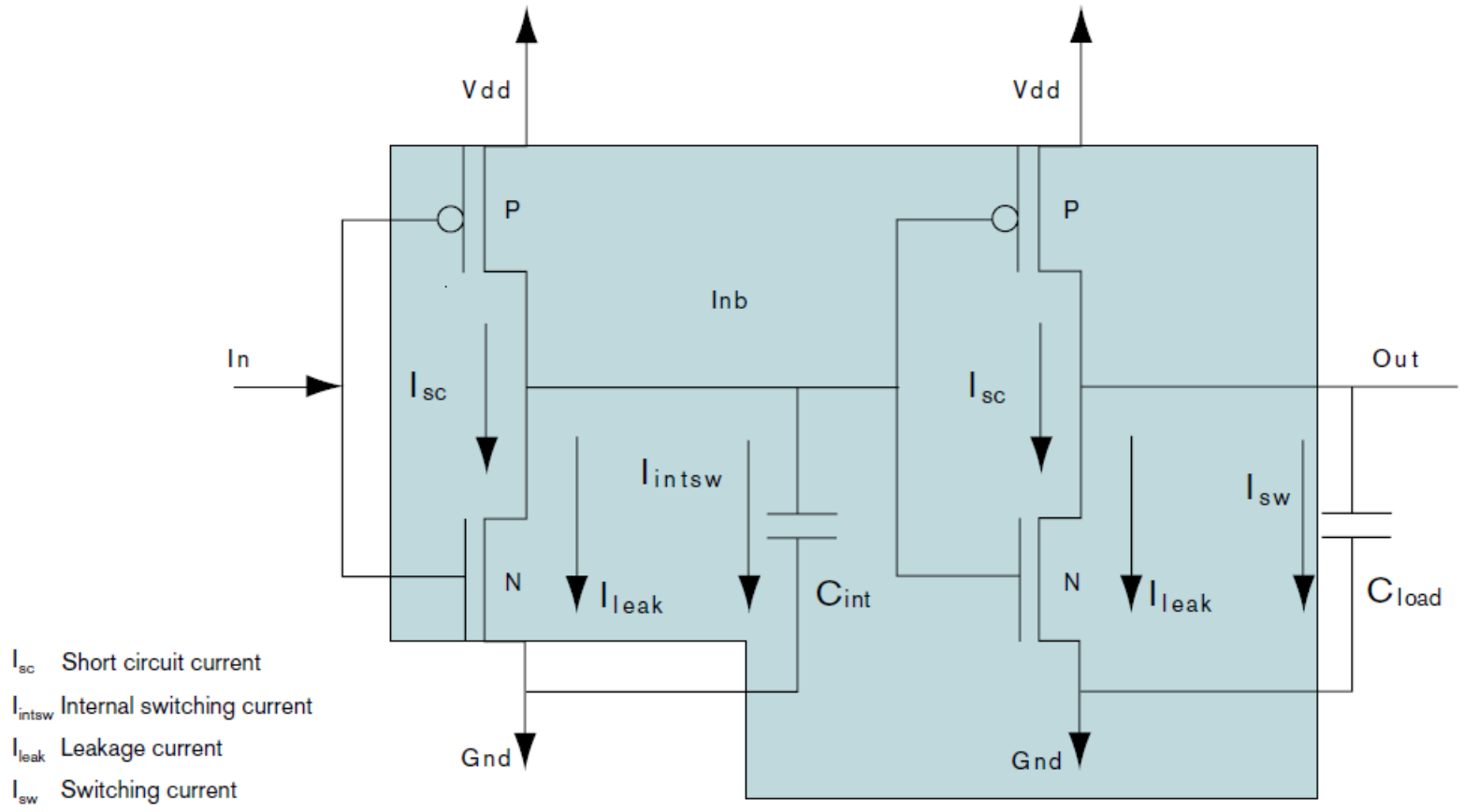
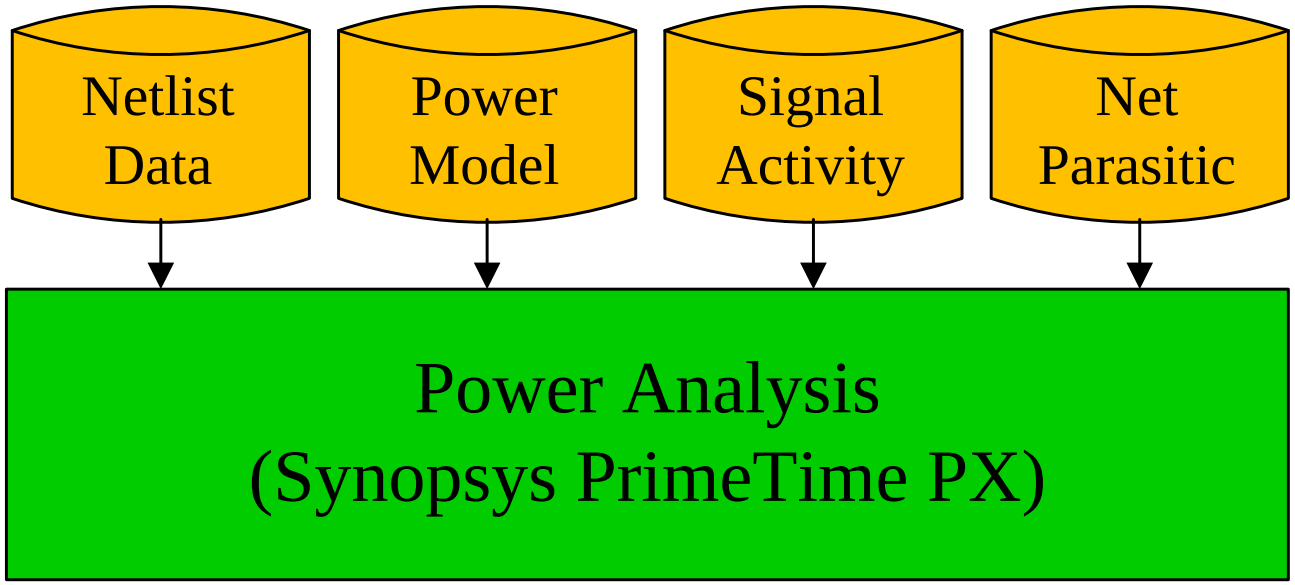


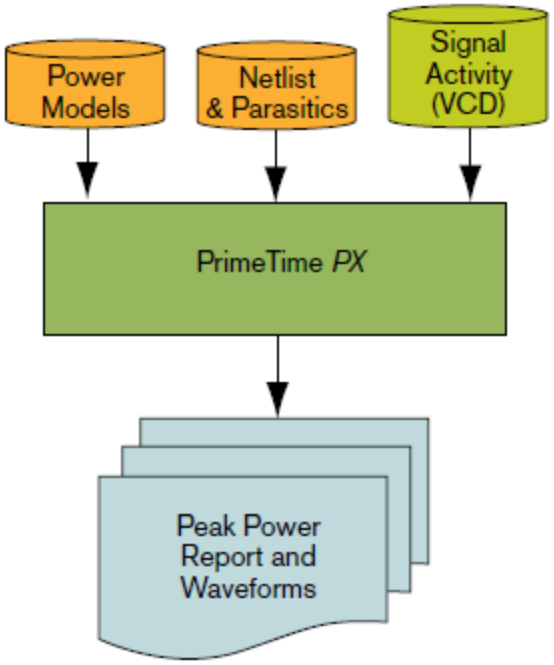
Figure from “Expanding the Synopsys PrimeTime Solution with Power Analysis”, Synopsis, inc

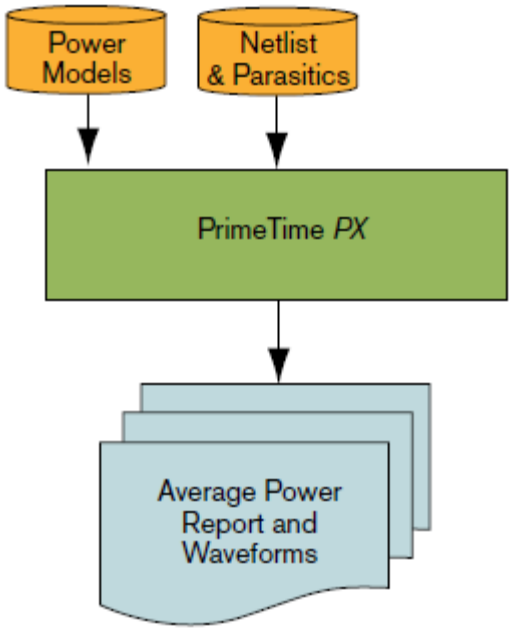
Power Analysis Requirement



- Netlist: PT PX accepts gate-level netlist only
- Power model: cell models which specify both the static and dynamic power consumption internal to the cell.
- Signal activity: ***VCD*** (Value Change Dump) or SAIF (Switching Activity Interchange Format) file from post-layout simulation
- Net parasitic: SPEF (Standard Parasitic Exchange Format) file

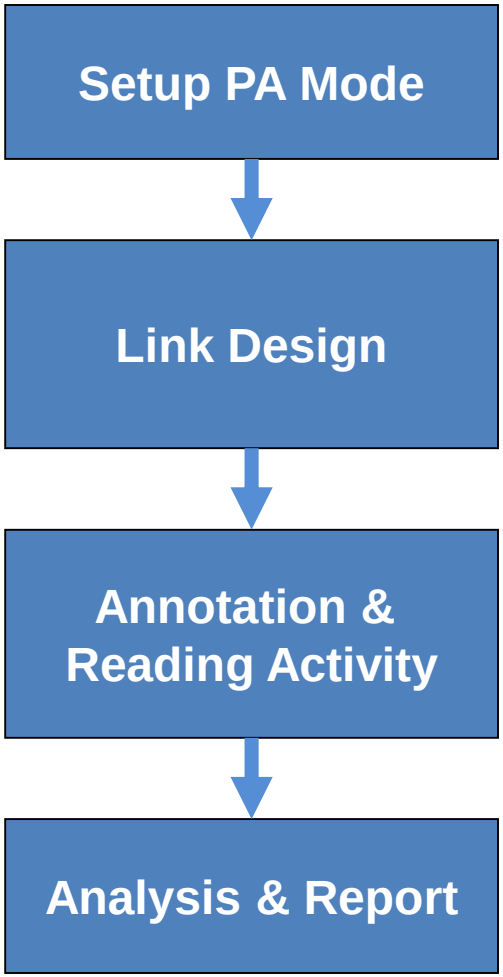
Power Analysis Modes





- **Average-Power Analysis**: the tool performs vector-free power analysis by using the default toggle rate. Fast but not accurate
- **Time-Based Power Analysis:** all the factors contributing to power consumption are supported in an accurate form. Peak and average power can be calculated, and detailed, time-based waveforms can be generated.

PrimeTime PX Flow



PrimeTime PX - Setup PA mode

- Set the Power Analysis Mode

```
set power_enable_analysis TRUE
set power_analysis_mode time_based/averaged
```

PrimeTime PX – Link design

- Set the search path and the link path

```
set search_path “lib path”
set link_library “*top_design.db”
set target_library “top_design.db”
```

- Read the design and the libraries

```
read_verilog top_level.v
current_design "top_level“
```

- Link the top design

```
link_design
```

PrimeTime PX – annotation & activity

- Annotate parasitic

```
read_parasitics top_level.spef
```

- Read switching activities

```
read_vcd -strip_path tb_top_design/u_top_design
./netlists/top_design.vcd
```

`-strip_path` option isolates the switching activity related to the module of our focus and annotates the design with that activity

PrimeTime PX – annotation

- Requirement
  - >**90% covering rate** is required for accurate power analysis

```
================================================================================
Summary:
Total number of nets = 115256
Number of annotated nets = 115256 (100.00%)
Total number of leaf cells = 97939
Number of fully annotated leaf cells = 97939 (100.00%)
================================================================================
```

PrimeTime PX – power analysis

- Power analysis

```
check_power
update_power
```

- Report power

```
report_power -verbose –hierarchy > power.rpt
```

PrimeTime PX – report

```
****************************************
Report : Time Based Power
Design : m_top_siso_detector_la
Version: F-2011.12-SP1
Date   : Wed Feb 13 09:19:10 2013
****************************************


 Attributes
 ----------
    i  -  Including register clock pin internal power
    u  -  User defined power group
```

Power Group	Internal Power	Switching Power	Leakage Power	Total Power (%)	Attrs
io_pad	0.0000	0.0000	0.0000	0.0000 (0.00%)	
memory	0.0000	0.0000	0.0000	0.0000 (0.00%)	
black_box	0.0000	0.0000	0.0000	0.0000 (0.00%)	
clock_network	0.0137	4.982e-03	3.116e-05	0.0187 (13.76%)	
register	3.029e-03	1.298e-03	8.082e-04	5.136e-03 (3.79%)	
combinational	0.0518	0.0557	4.337e-03	0.1118 (82.45%)	
sequential	0.0000	0.0000	0.0000	0.0000 (0.00%)	

```
Net Switching Power  =    0.0620   (45.73%)
Cell Internal Power  =    0.0684   (50.45%)
Cell Leakage Power   = 5.176e-03   ( 3.82%)
                         ---------
Total Power          =    0.1356  (100.00%)

X Transition Power   =    0.0000
Glitching Power      = 7.229e-04

Peak Power           =    0.6758
Peak Time            =   172.303
```

Lab

- http://www.eit.lth.se/index.php?ciuid=647&coursepage=3553
- Download provided input-files for PrimeTime PX (readme.txt)
- Fill the template *medianfilter_pt_power.tcl* & run PrimeTime

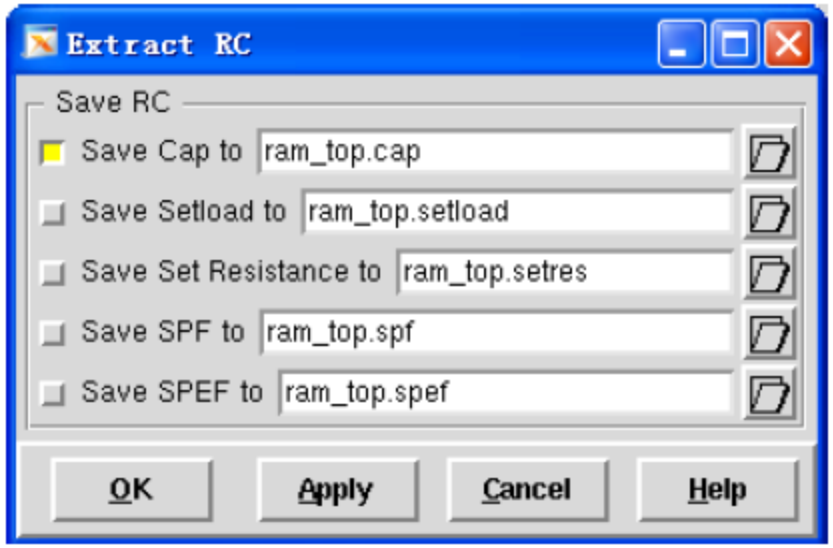
PrimeTime PX – spef

- Output Parasiticsin SoC Encounter

@ shell

```
rcOut -spef ./netlists/top_level.spef
```

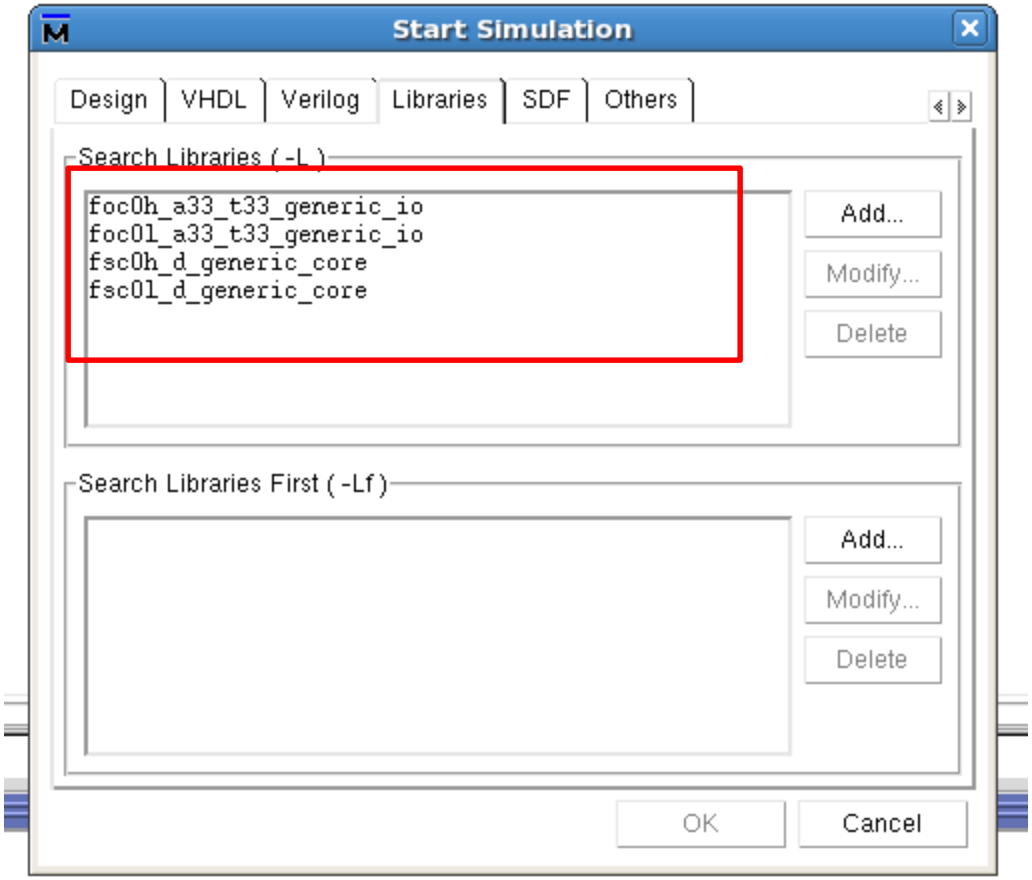
@ GUI: Timing->Extract RC



PrimeTime PX – Post_layout Sim

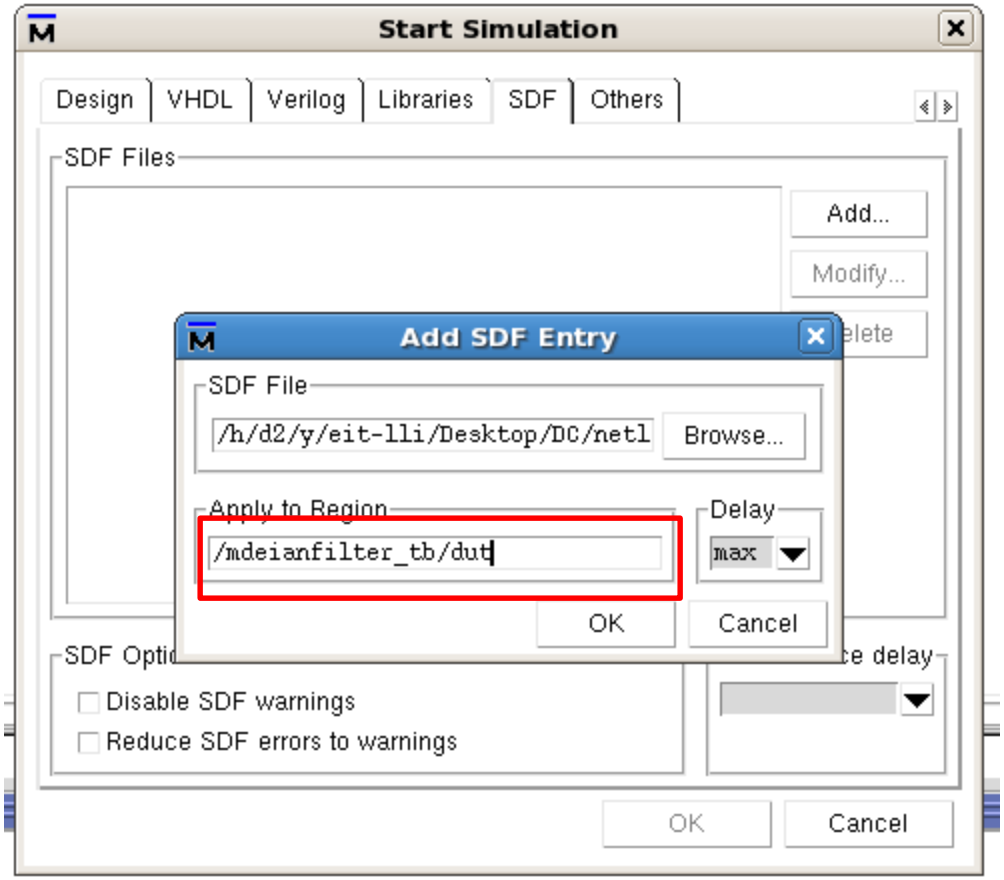
Add Library (mem lib uses behavior model)

Simulate->Start Simulation



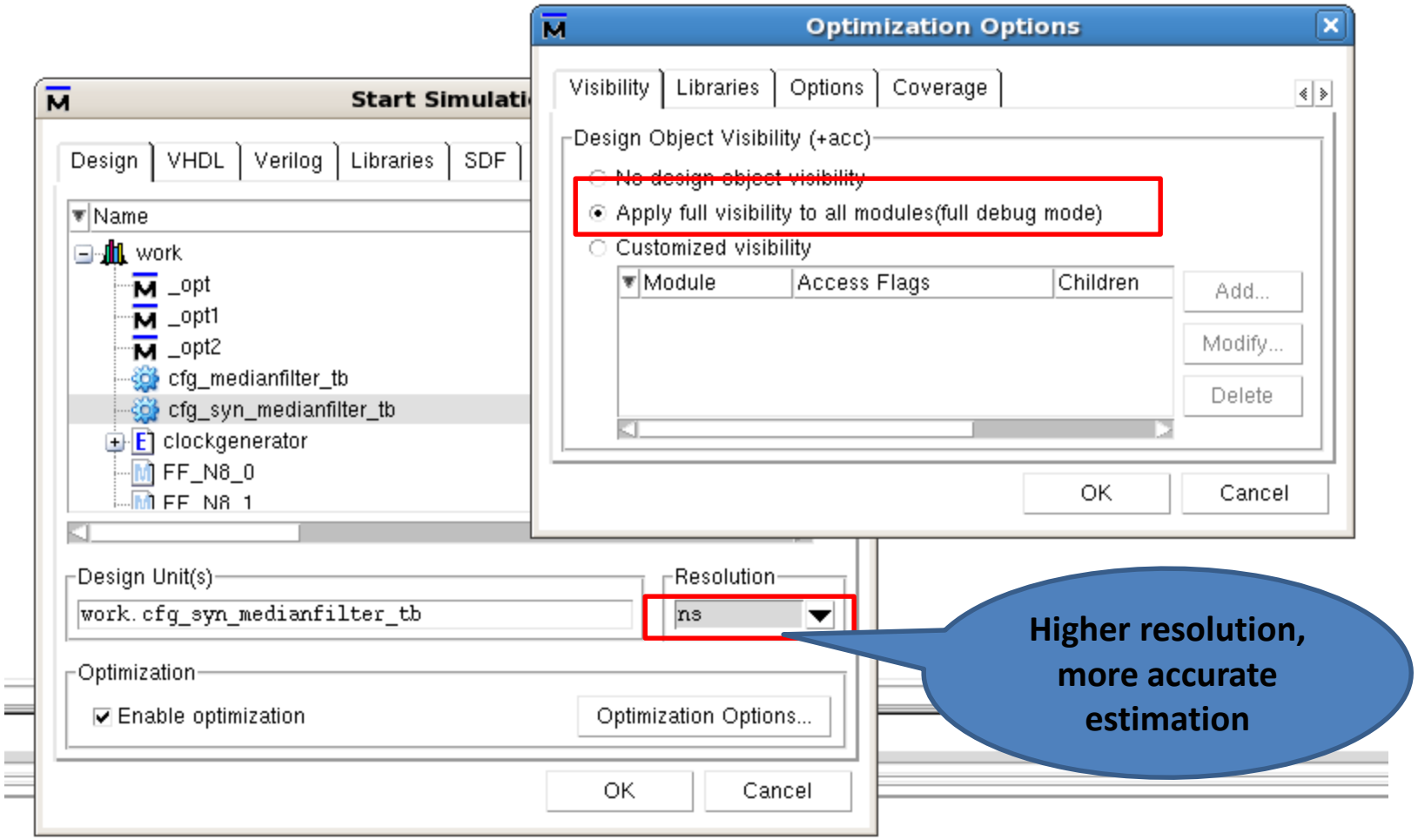
PrimeTime PX – Post_layout Sim

Annotate SDF



PrimeTime PX – Post_layout Sim

Optimization setup



PrimeTime PX – Post_layout Sim

Dump VCD File

vcd file ./nestlists/medianfilter.vcd

vcd add –r /medianfilter_tb/dut/*

run –all