Model Diagnoser™

Ultimate Verification, Quality Assurance and Interactive Debugging for .Lib Model of Cell Library

Agenda

- Introduction
- Model Diagnoser Flow
- Setup/hold time diagnosis for function/noise violations
- Repairing library model from function/noise violations
- Inaccuracy diagnosis by library model comparison
- Function verification between views of library model
- CCS consistency check of library model
- Interactive Debugging
- The Conclusion

Legend's Products

- IP Library Verification/Characterization Products
  - Model Diagnoser™: *Cell Library QA, Diagnosis and Debugging*
  - Charflo-Cell!™ : *Automatic Cell/IO Library Characterization*
  - Charflo-Memory!™: *Automatic Memory Characterization*
- Circuit Simulation Products
  - MSIM®: *Accurate-Spice Simulator for Analog/RF/Mixed-Signal IC and IP, LCD, and PCB/IBIS/Package*
  - PCB Design Manager: *Integrated Schematic & Simulation Environment with Test Bench Automation*
  - Turbo-MSIM™: *Fast-Spice Simulator*

The Problems

Standard /IO Cell Library Modeling

- The .Lib model of standard / IO cell library may be
  - Incorrectly modeled or characterized
  - Inappropriately applied at new PVTs
- Need QA process to assure the .Lib timing model
  - No functional failures shall be resulted
  - Noise on output pins shall be within the margin
  - Timing, power and noise models are valid in accuracy
  - No over-excessive timings degrading speed seriously
- Need repairing .Lib model to meet the QA criteria

The Solutions

Model Diagnoser^{TM} Functions

- Setup/hold time diagnosis for function/noise violations
- Repairing library model from function/noise violations
- Inaccuracy diagnosis by library model comparison
- Function verification between views of library model
- CCS consistency check of library model
- Interactive Debugging

The Necessities

Model Diagnoser on top of Characterization

- Use ultimate validation, different from characterization
  - Verify setup/hold time by directly plugging into final simulation, instead of bi-section with error tolerances
- Check internal nodes, not covered by characterization
  - Examine glitches and noise strengths on internal nodes, instead of 'output pins only' by simulator's bi-section
- Trust but verify possible errors in characterization from
  - Characterization tools
  - Simulation tools
  - Human mistakes and manual settings

TSMC’s Selection

Model Diagnoser^{TM}

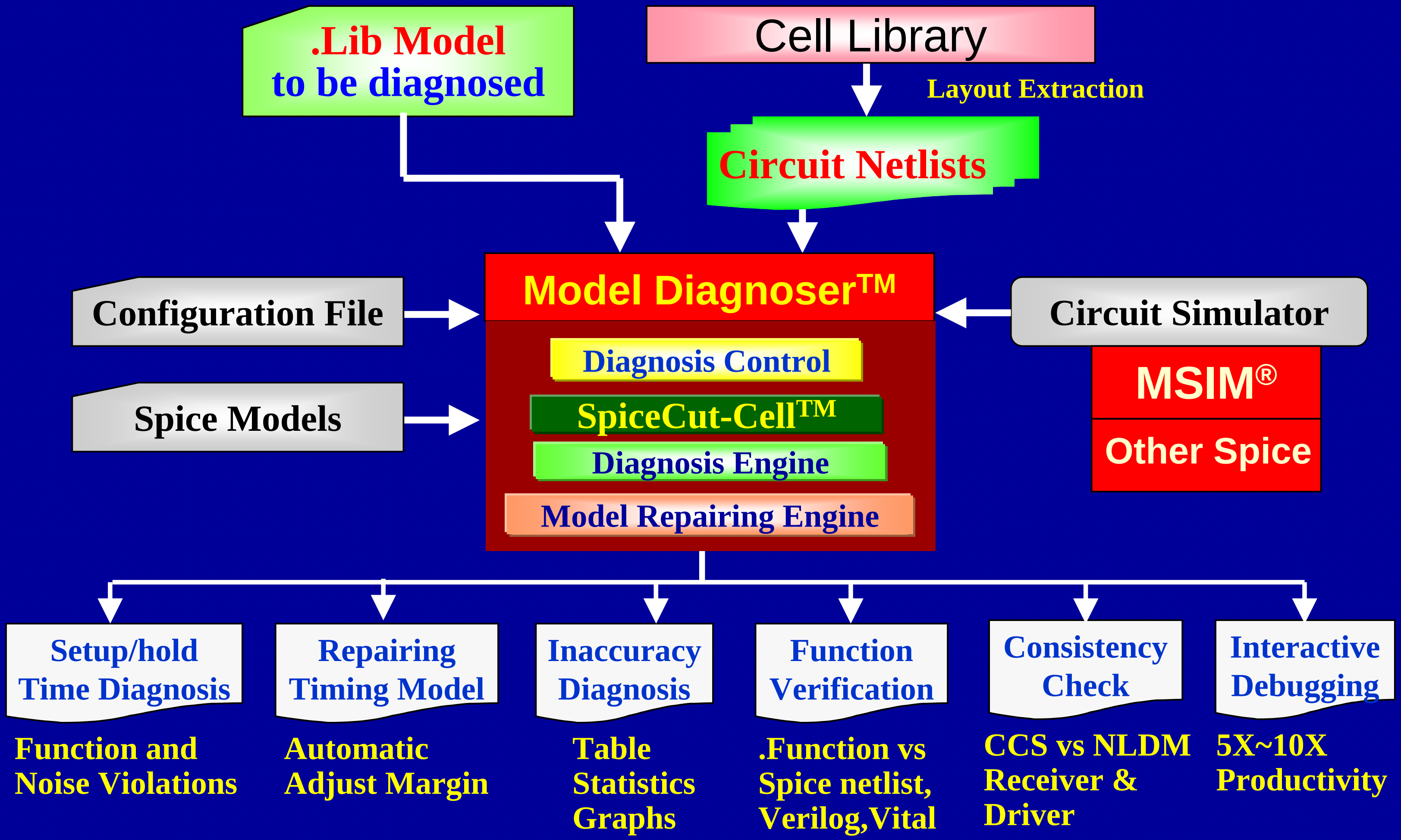
- TSMC Selects Legend’s Model Diagnoser for Standard Cell Library Quality Assurance

  http://www.legenddesign.com/BW/060909.shtml

- *"Legend’s Model Diagnoser can help locate the functional issues in the .lib models of TSMC 90nm and 65nm standard cell libraries. We are satisfied with the tool results, and continue to work with Legend to ensure our library quality for advanced nanometer technologies,"*

  said Tom Quan, deputy director of design service marketing at TSMC.

Model Diagnoser™ Flow



.Lib Model
to be diagnosed

Cell Library

Layout Extraction

Circuit Netlists

Configuration File

Spice Models

Model Diagnoser™

Diagnosis Control

SpiceCut-Cell™

Diagnosis Engine

Model Repairing Engine

Circuit Simulator

MSIM®

Other Spice

Setup/hold Time Diagnosis

Function and
Noise Violations

Repairing Timing Model

Automatic
Adjust Margin

Inaccuracy Diagnosis

Table
Statistics
Graphs

Function Verification

.Function vs
Spice netlist,
Verilog,Vital

Consistency Check

CCS vs NLDM
Receiver &
Driver

Interactive Debugging

5X~10X
Productivity

Setup/Hold Time Diagnosis

Function & Noise Check on Latch/Flip-flop

- Analyze the circuit of each cell by SpiceCut tool, and locate high-risk spots inside the cell.
- Build all possible state patterns based on functions and conditions in .Lib, and set up corresponding stimulus.
- Simulate the cell by applying setup/hold time, and min. pulse width from .Lib, with the stimulus built.
- Verify the simulation results for locating functional failures and noise violations (e.g. glitch)
- Locate over-excessive timings (e.g. min. clock width) to prevent from serious performance degrading

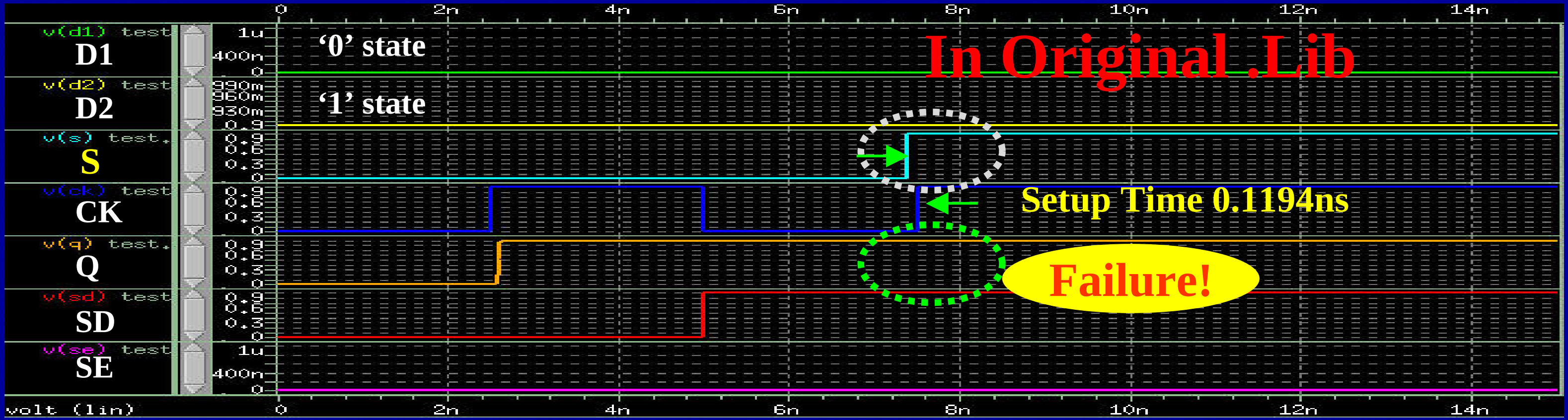
SpiceCut-Cell Functions

Circuit Analysis & Pattern Recognition

- Build circuit database of nodes and MOSFETs, and extract subcircuit configurations.
- Locate the high-risk nodes inside the cells to monitor for ensuring the modeling quality.
- Identify the measurable nodes inside the cells before tri-state output for complex I/O cell characterization.
- Perform pattern recognition over the circuits of standard cells, complex cells and customized cells.

‘Function’ Violation

Due to Insufficient Setup Time



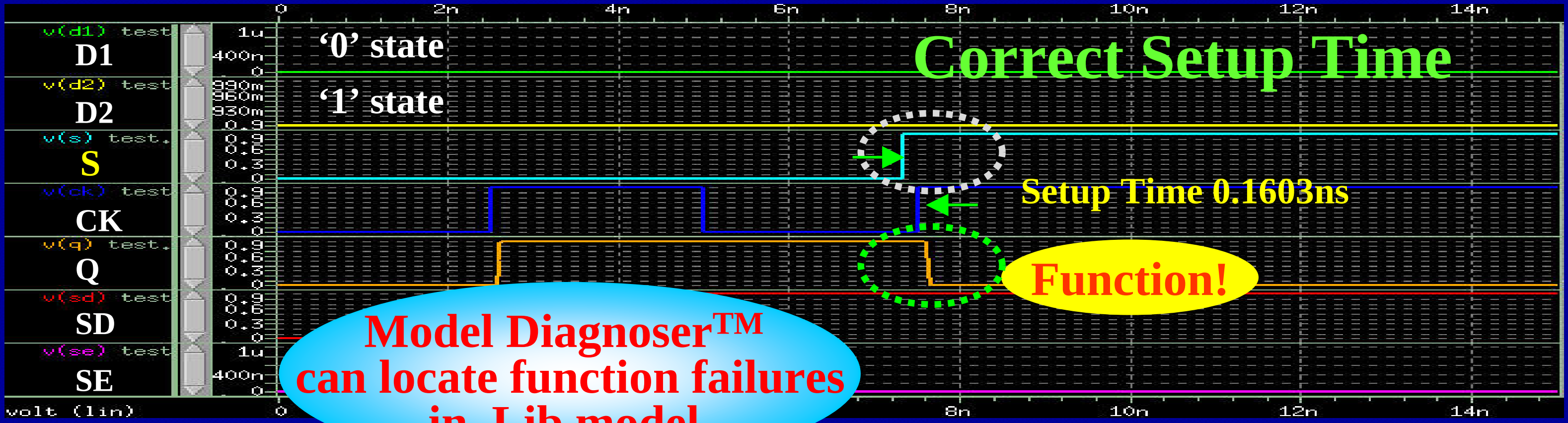
‘0’ state

‘1’ state

In Original .Lib

Setup Time 0.1194ns

Failure!



‘0’ state

‘1’ state

Correct Setup Time

Setup Time 0.1603ns

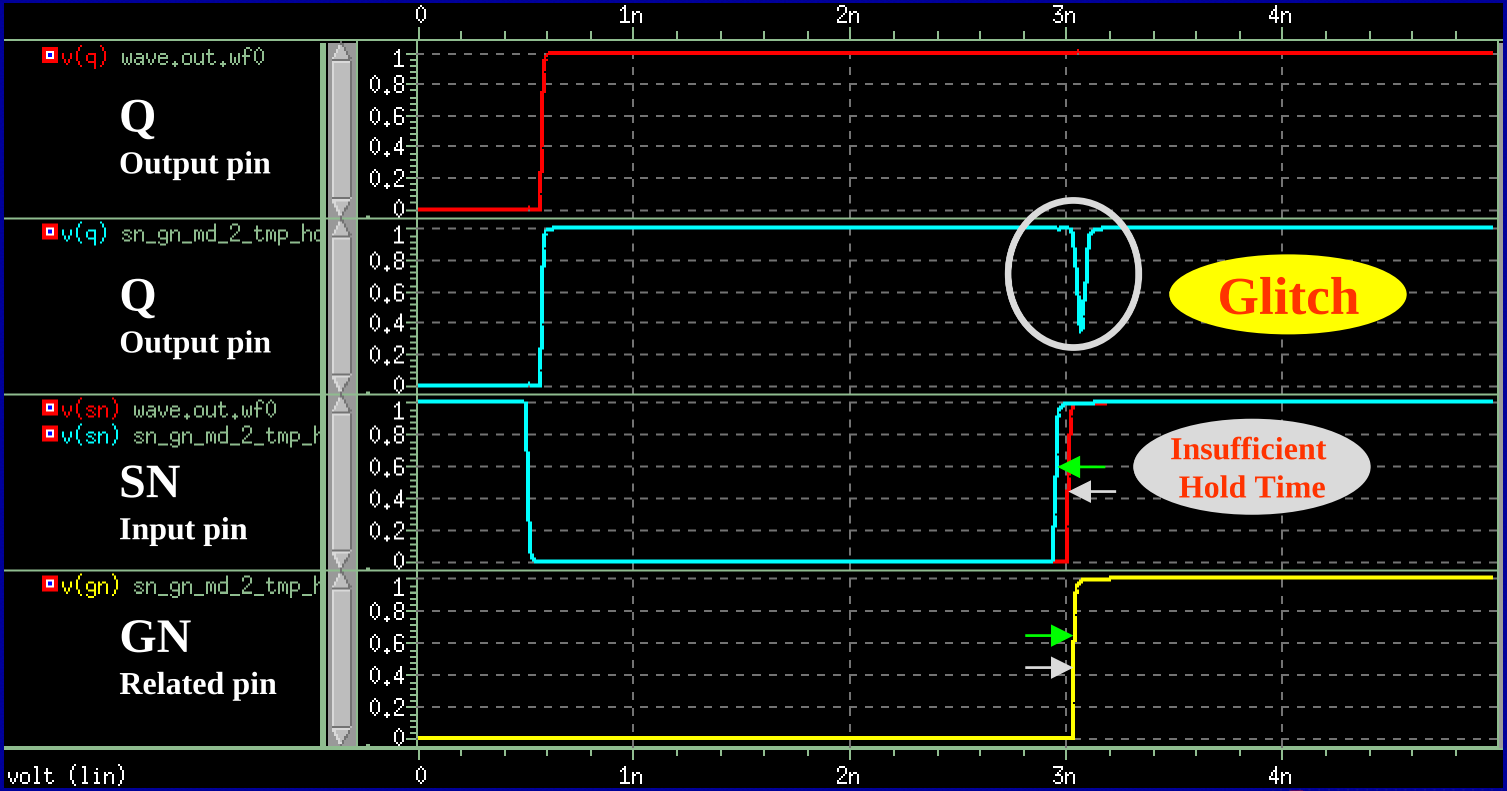
Function!

Model Diagnoser^{TM} can locate function failures in .Lib model.

Output-Pin 'Glitch' Violation

Due to Insufficient Hold Time

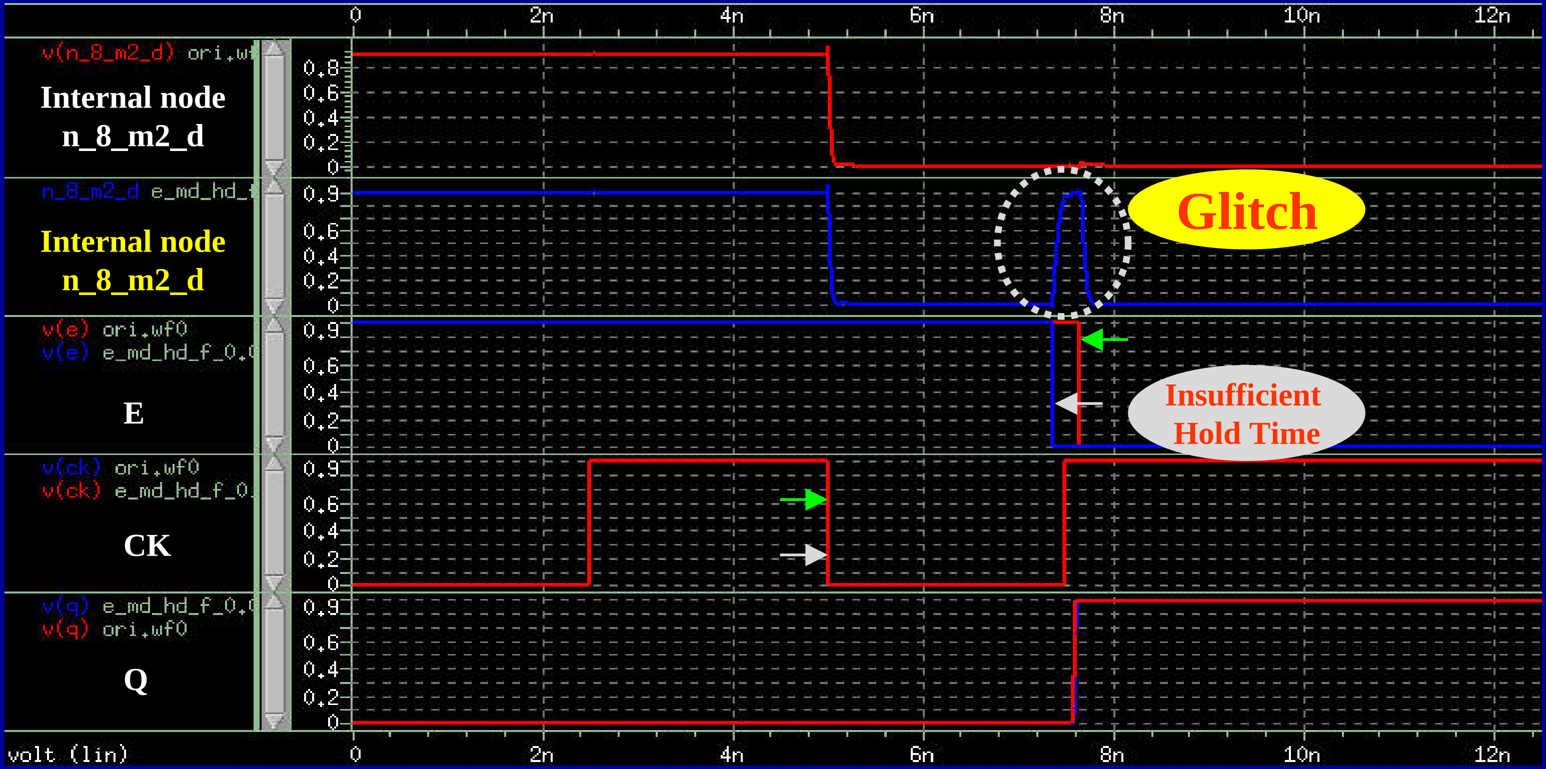
- The glitch is of 66% Vdd with the width 37ps.



Internal 'Glitch' Violation

Due to Insufficient Hold Time

- The glitch is of 96% Vdd with the width 0.303ns.



Technology Leader in IP Characterization and IC/PCB Simulation

Legend Design Technology

Run-Statistics Examples

Function & Noise Check on Latch/Flip-flop

- Diagnosing one cell library normally takes 2~4 hours

Cell Library	Number of Latch & FlipFlop Cells	Add Margin 0.05ns/ 0.1ns/ 0.2ns		Violation-Free Margin to add
		Function Violation	Glitch* Violation	
40nm (857 cells)	**158**	**46/ 4/ 2**	**0/ 0/ 0**	**0.25ns**
55nm (1084 cells)	**294**	**2/ 0/ 0**	**15/ 5/ 1**	**0.22ns**
65nm (814 cells)	**228**	**28/ 0/ 0**	**13/ 0/ 0**	**0.1ns**
90nm (837 cells)	**251**	**76/ 71/ 10**	**130/ 126/ 1**	**0.3ns**

		Add Margin 0.5ns/ 1.0ns/ 1.5ns		Violation-Free Margin to add
0.18um (594 cells)	**145**	**51/ 2/ 1**	**0/ 0/ 0**	**2.4ns**
0.35um (588 cells)	**139**	**97/ 45/ 20**	**0/ 0/ 0**	**1.7ns**

* Report 'Glitch Violation' only when (1) glitch_height > 40% of Vdd (2) glitch_width > 20ps (3) glitch_height_in_V * glitch_width_in_ps > 20 V-ps

Repairing Library Model

To Prevent from Function/Noise Violations

- For setup/hold time & min pulse width in .Lib model, the margin shall be automatically adjusted to ensure
  - No functional violations
  - No glitch violations on internal nodes and output pins
  - No over-excessive timings degrading performance seriously
- Margin increment can be by values or by percent
- Margin adjustment for tabular setup/hold time can be determined by the worst corner, by four extreme corners and the central, or by all entries of that table.

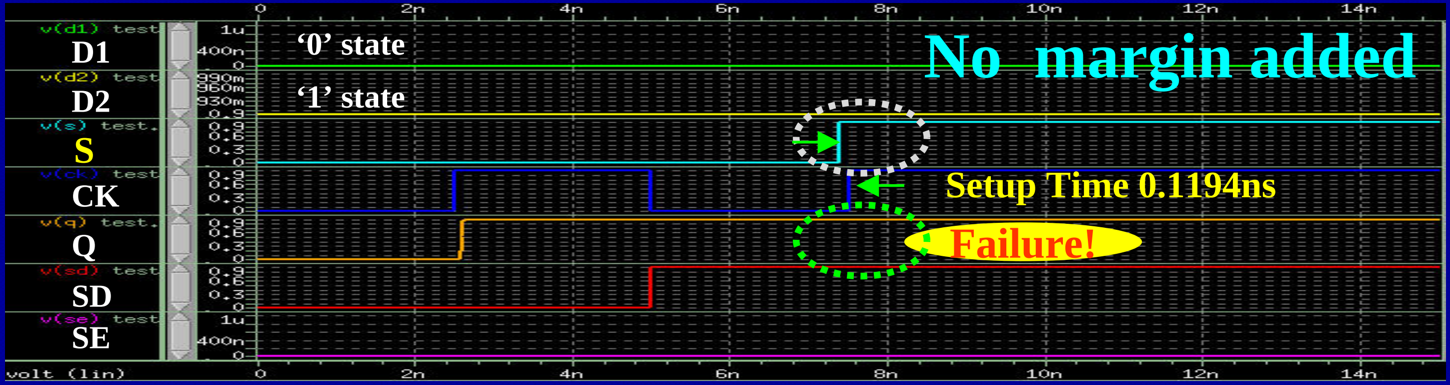
Repair .Lib Model

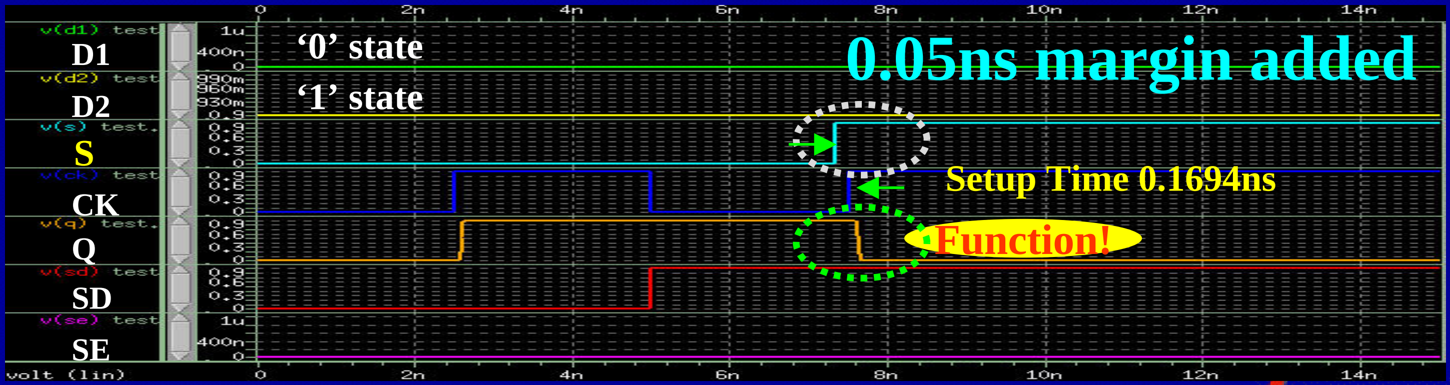
Add Margin 0.05ns to Setup Time

Cell SDFMQM8NA: **Pin 'S' Setup_Rise**

State Pattern

se=0 sd=1 d1=0 **s=0** d2=1 **state(Q)=1**
se=0 sd=1 d1=0 **s=1** d2=1 **state(Q)=0**





Correct ‘Excessive Spike’ in .Lib

To Prevent from Serious Speed Degrading

- ‘Excessive Spike’ entry in the table of setup/hold time & min pulse width in .Lib model could cause
  - Difficult to keep monotonic table as required by PrimeTime
  - Large performance sacrifice (e.g. due to min clock width)
- Simulate the cell by applying the ‘reduced’ setup/hold time & min pulse width, i.e. adding ‘negative’ margin.
- At the various negative margins, verify the simulation results for locating functional ‘success’ without noise violations. Then, report the necessary corrections.

Inaccuracy Diagnosis

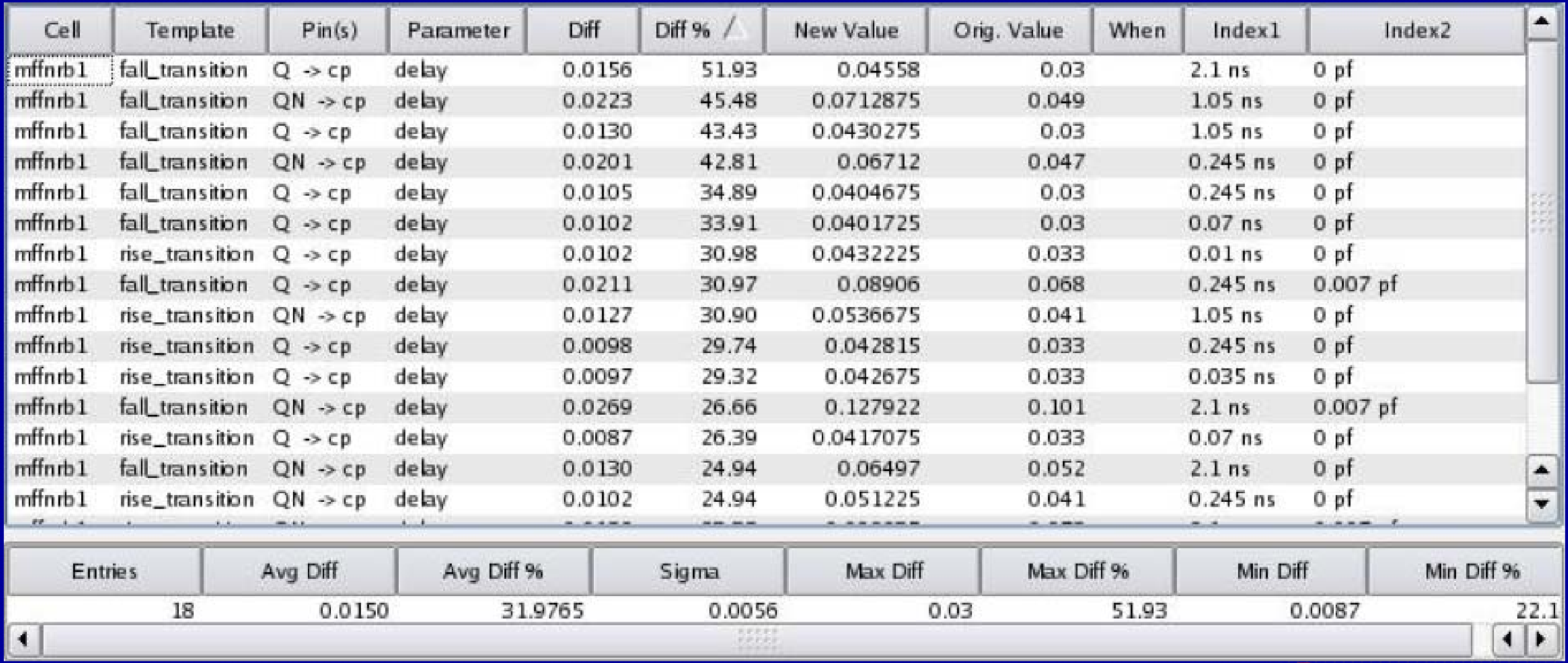
By Library Model Comparison

- Diagnose the inaccuracy of target library by comparing its .Lib model with the re-characterized one by using Model Diagnoser.
- The difference between the tables of parameters will be represented by
  - Table report
  - Statistical report
  - 2D Graphical report
  - 3D Graphical report

Inaccuracy Report

Sorted Timing-Arc and Statistical Report

- Statistical report can be used for measuring overall quality of target library, at cell or parameter level.



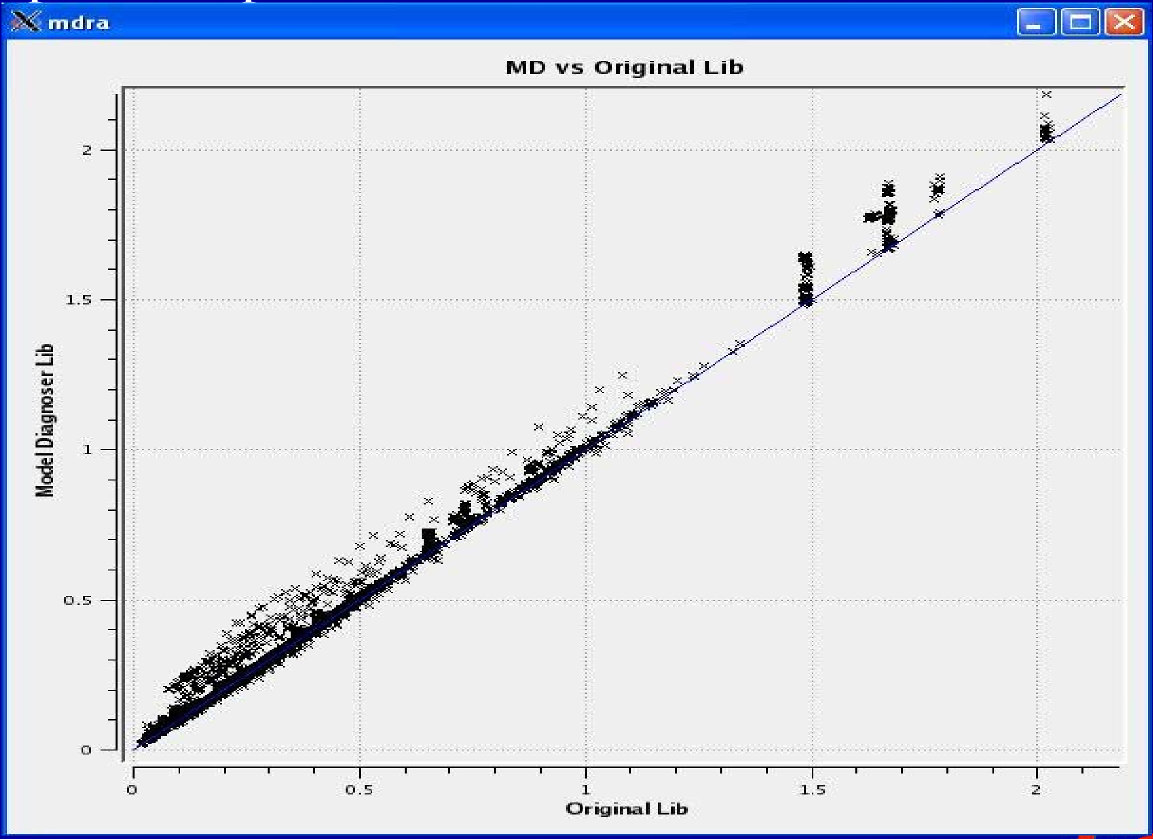
Cell	Template	Pin(s)	Parameter	Diff	Diff %	New Value	Orig. Value	When	Index1	Index2
mffnrb1	fall_transition	Q -> cp	delay	0.0156	51.93	0.04558	0.03		2.1 ns	0 pf
mffnrb1	fall_transition	QN -> cp	delay	0.0223	45.48	0.0712875	0.049		1.05 ns	0 pf
mffnrb1	fall_transition	Q -> cp	delay	0.0130	43.43	0.0430275	0.03		1.05 ns	0 pf
mffnrb1	fall_transition	QN -> cp	delay	0.0201	42.81	0.06712	0.047		0.245 ns	0 pf
mffnrb1	fall_transition	Q -> cp	delay	0.0105	34.89	0.0404675	0.03		0.245 ns	0 pf
mffnrb1	fall_transition	Q -> cp	delay	0.0102	33.91	0.0401725	0.03		0.07 ns	0 pf
mffnrb1	rise_transition	Q -> cp	delay	0.0102	30.98	0.0432225	0.033		0.01 ns	0 pf
mffnrb1	fall_transition	Q -> cp	delay	0.0211	30.97	0.08906	0.068		0.245 ns	0.007 pf
mffnrb1	rise_transition	QN -> cp	delay	0.0127	30.90	0.0536675	0.041		1.05 ns	0 pf
mffnrb1	rise_transition	Q -> cp	delay	0.0098	29.74	0.042815	0.033		0.245 ns	0 pf
mffnrb1	rise_transition	Q -> cp	delay	0.0097	29.32	0.042675	0.033		0.035 ns	0 pf
mffnrb1	fall_transition	QN -> cp	delay	0.0269	26.66	0.127922	0.101		2.1 ns	0.007 pf
mffnrb1	rise_transition	Q -> cp	delay	0.0087	26.39	0.0417075	0.033		0.07 ns	0 pf
mffnrb1	fall_transition	QN -> cp	delay	0.0130	24.94	0.06497	0.052		2.1 ns	0 pf
mffnrb1	rise_transition	QN -> cp	delay	0.0102	24.94	0.051225	0.041		0.245 ns	0 pf

Entries	Avg Diff	Avg Diff %	Sigma	Max Diff	Max Diff %	Min Diff	Min Diff %
18	0.0150	31.9765	0.0056	0.03	51.93	0.0087	22.1

2D/3D Graphic Report

Based on Statistics of Cell & Parameter

- 2D Graphic Report



Function Verification

Between Views of Library Models

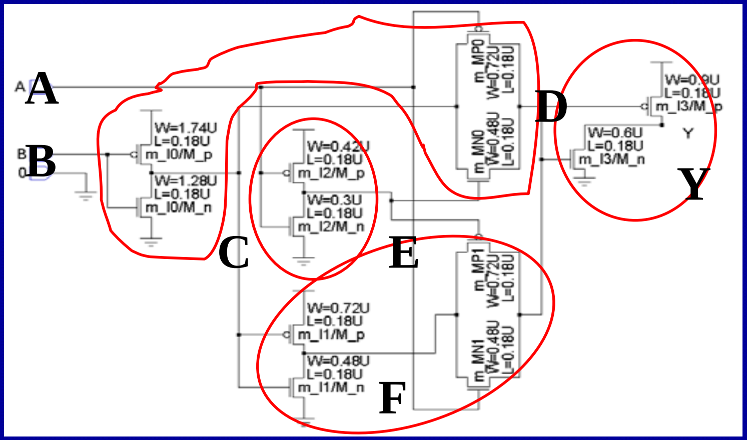
- Check the consistency between function description (i.e. .FUNCTION statements) in .Lib and
  - Spice netlist at transistor level
  - Verilog descriptions
  - *Vital descriptions
- Report the difference of their derived state patterns

* Vital verification will be released in Q3, 2010

Extract Cell Functions

Directly from Spice Circuit Netlist

- Standard cells are normally by static designs. Their subcircuits and functions are quite straightforward.
- SpiceCut-Cell can partition/recognize those subcircuit patterns, and extract their corresponding functions.
- Example: Exclusive-OR (XOR) Cell



$C = -B;\quad F = -C = B;\quad E = -A;$

$D = C * E + F * (-E)$

$\quad = (-B) * (-A) + B * A$

$Y = (-A) * (-B) + A * B$

$\mathbf{Y = A \,\hat{}\, B}$

Verify Functions in .Lib

Against Functions Extracted from Circuits

- Extract the logic functions directly from cell circuit netlist by SpiceCut-Cell
- Validate the FUNCTION statement of each timing arc in the existed .Lib model, by comparing with those circuit-extracted ones.
- Confirm the specifications of related_pin, timing_sense, when, and sdf_cond etc. in the existed .Lib model.

CCS Model Check

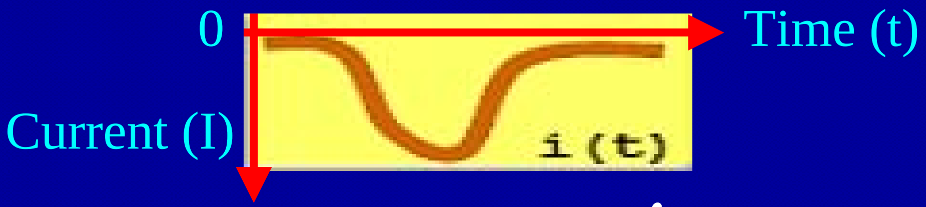
- CCS Receiver Model Consistency Check

  Compare CCS’ C1 and C2 with NLDM input capacitance

- CCS Driver Model Consistency Check

  Compare CCS waveform’s peak time with NLDM delay

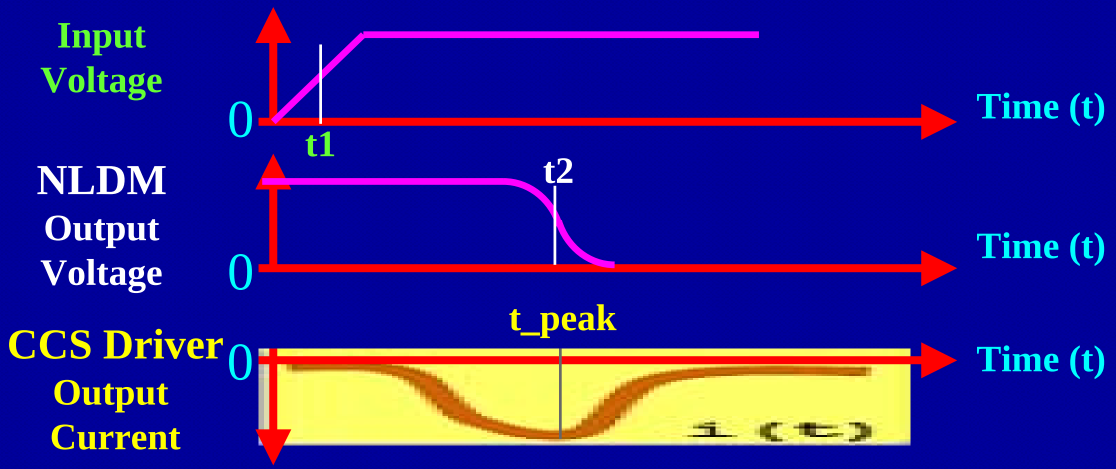
- CCS Driver Model Integration Check



- $I = C * dV/dt \quad \Rightarrow \quad \int I\, dt / C = Vdd$
- Difference between $\int I\, dt / C$ and Vdd need be < 5%

CCS Driver Model

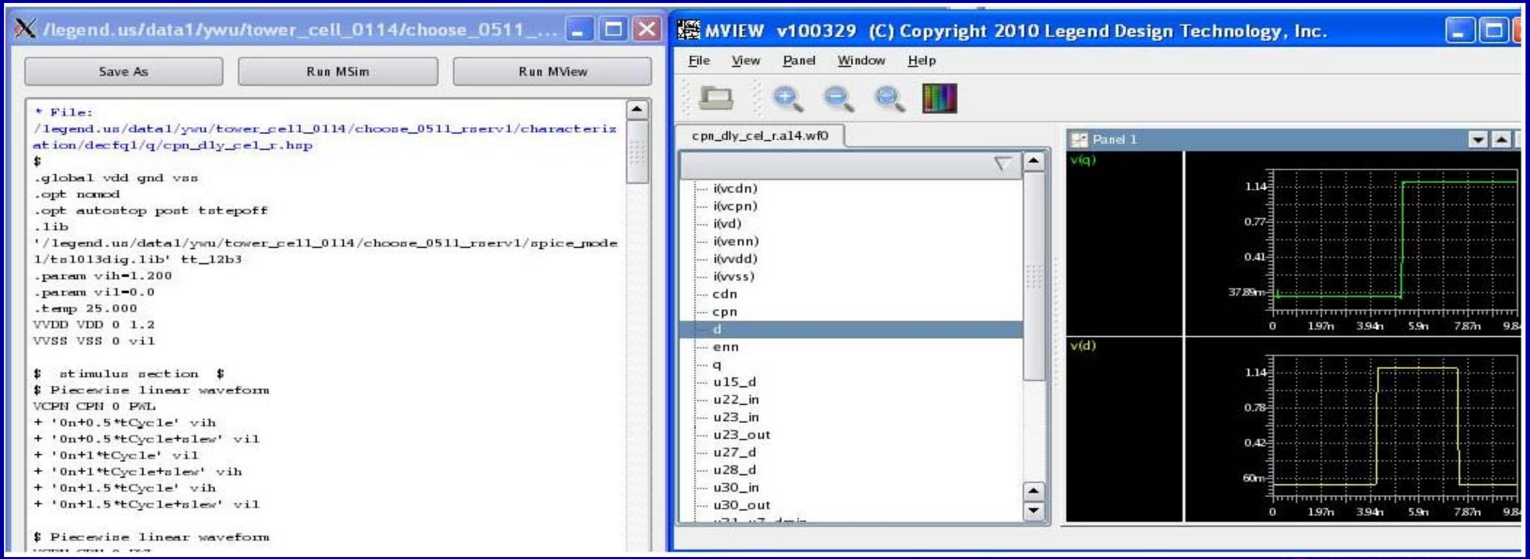
CCS vs NLDM Consistency Check



- NLDM_delay = t2 - t1
- CCS_delay from driver model = t_peak - t1
- Consistency can be checked by comparing CCS_delay with NLDM_delay, and reporting the difference.

Interactive Debugger

- ◆ Click the selected violation or timing-arc, and the windows of circuit stimulus/netlist and waveform viewer will automatically pop up.
- ◆ Users can Edit/Save, Simulate and View Waveforms.



```
* File:
/legend.us/data1/ywu/tower_cell_0114/choose_0511_rserv1/characteriz
ation/decfq1/q/cpn_dly_cel_r.hsp
$
.global vdd gnd vss
.opt nomod
.opt autostop post tstepoff
.lib
'/legend.us/data1/ywu/tower_cell_0114/choose_0511_rserv1/spice_mode
l/ts1013dig.lib' tt_12b3
.param vih=1.200
.param vil=0.0
.temp 25.000
VVDD VDD 0 1.2
VVSS VSS 0 vil

$  stimulus section  $
$ Piecewise linear waveform
VCPN CPN 0 PWL
+ '0n+0.5*tCycle' vih
+ '0n+0.5*tCycle+slew' vil
+ '0n+1*tCycle' vil
+ '0n+1*tCycle+slew' vih
+ '0n+1.5*tCycle' vih
+ '0n+1.5*tCycle+slew' vil

$ Piecewise linear waveform
```

Conclusion

Model Diagnoser$^{\text{TM}}$

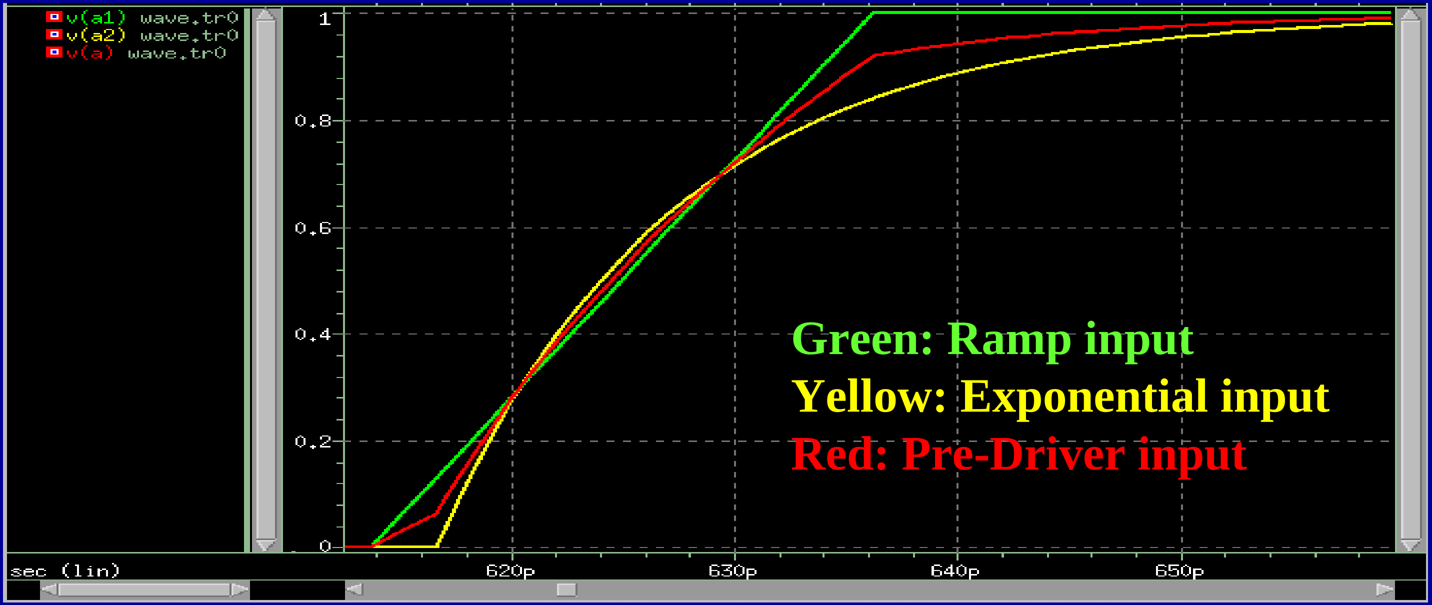
- Enable quality assurance of cell library .Lib models which are critical for SoC designs.
- Quickly locate the function, noise, timing and power violations in the cell library .Lib model at any PVT.
- Repair .Lib model of latch/flip-flop by automatically adjusting the margins for production yields.
- Easy to use with Interactive Debugger by GUI, and programmable configurations.
- Fully proven for production flow.

Appendix

Pre-Driver Input Waveform

For 45nm Cell Characterization

- Pre-driver method is analogous to taking the output of a PWL source and passing it through a low-pass filter.
- Model Diagnoser supports both ramp and pre-driver input



Green: Ramp input

Yellow: Exponential input

Red: Pre-Driver input

Criteria for ‘Glitch’ Violation

Noise Check on Latch/Flip-flop

- Command Format for criteria of Glitch violation

  Glitch 0.6 50 30

  Report Glitch violation when

  (glitch_height > 60% of Vdd) AND

  (glitch_width > 50ps) AND

  (glitch_height_in_V * glitch_width_in_ps > 30 V-ps)

- Default values for criteria of Glitch violation

  Glitch 0.3 10 10

Legend’s Patents

Model Diagnoser$^{\text{TM}}$

- United States Patent 7231336

  "Glitch and metastability checks using signal characteristics“

- United States Patent 7131088

  "Reliability based characterization using bisection"

- United States Patent 7203918

  "Delay and signal integrity check and characterization“

- United States Patent 6112022

  "Method for simulating ULSI/VLSI circuit designs"