Digital IC Design

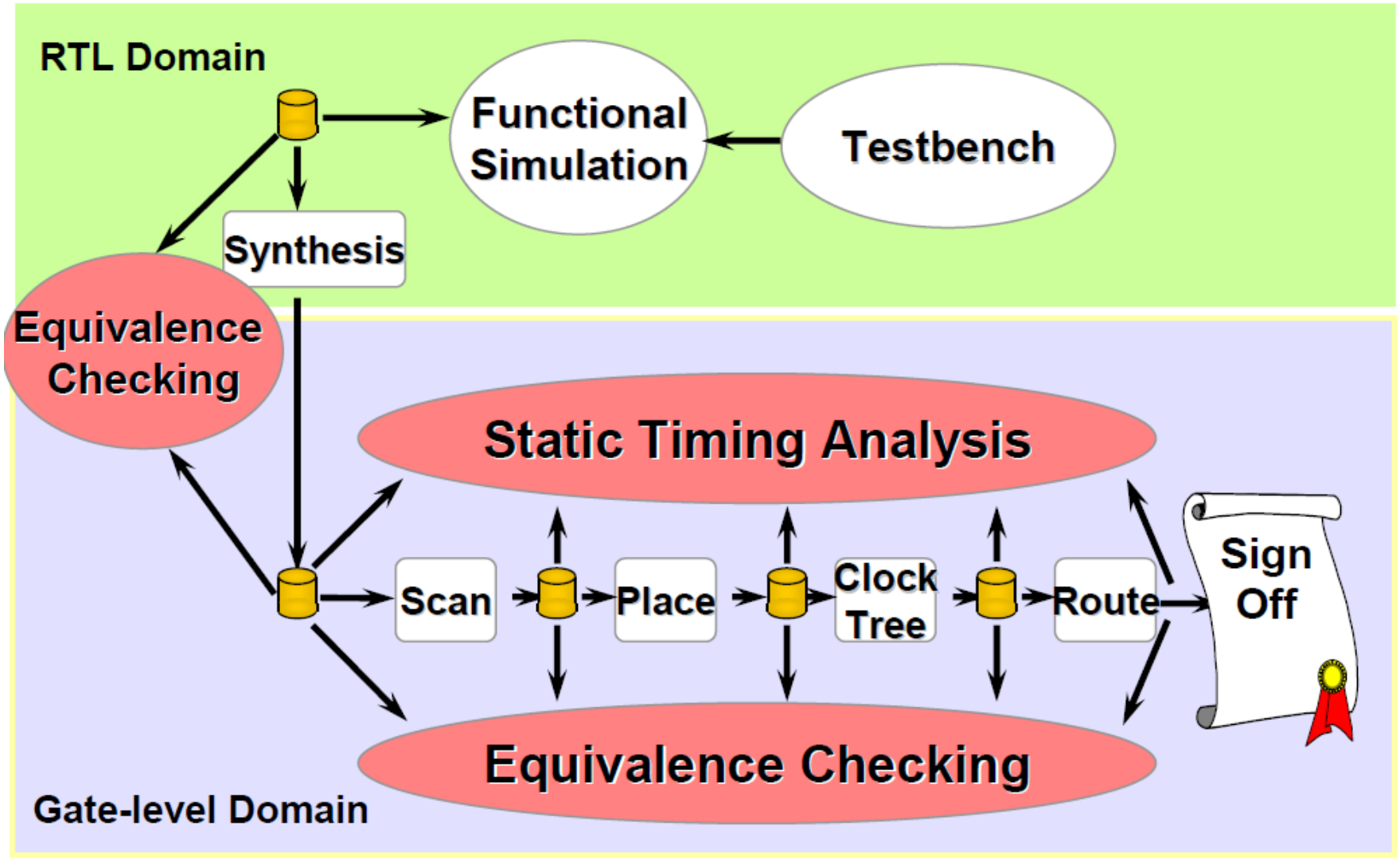
Lecture 11:
Static Timing Analysis

黃柏蒼 **Po-Tsang (Bug) Huang**
bughuang@nycu.edu.tw

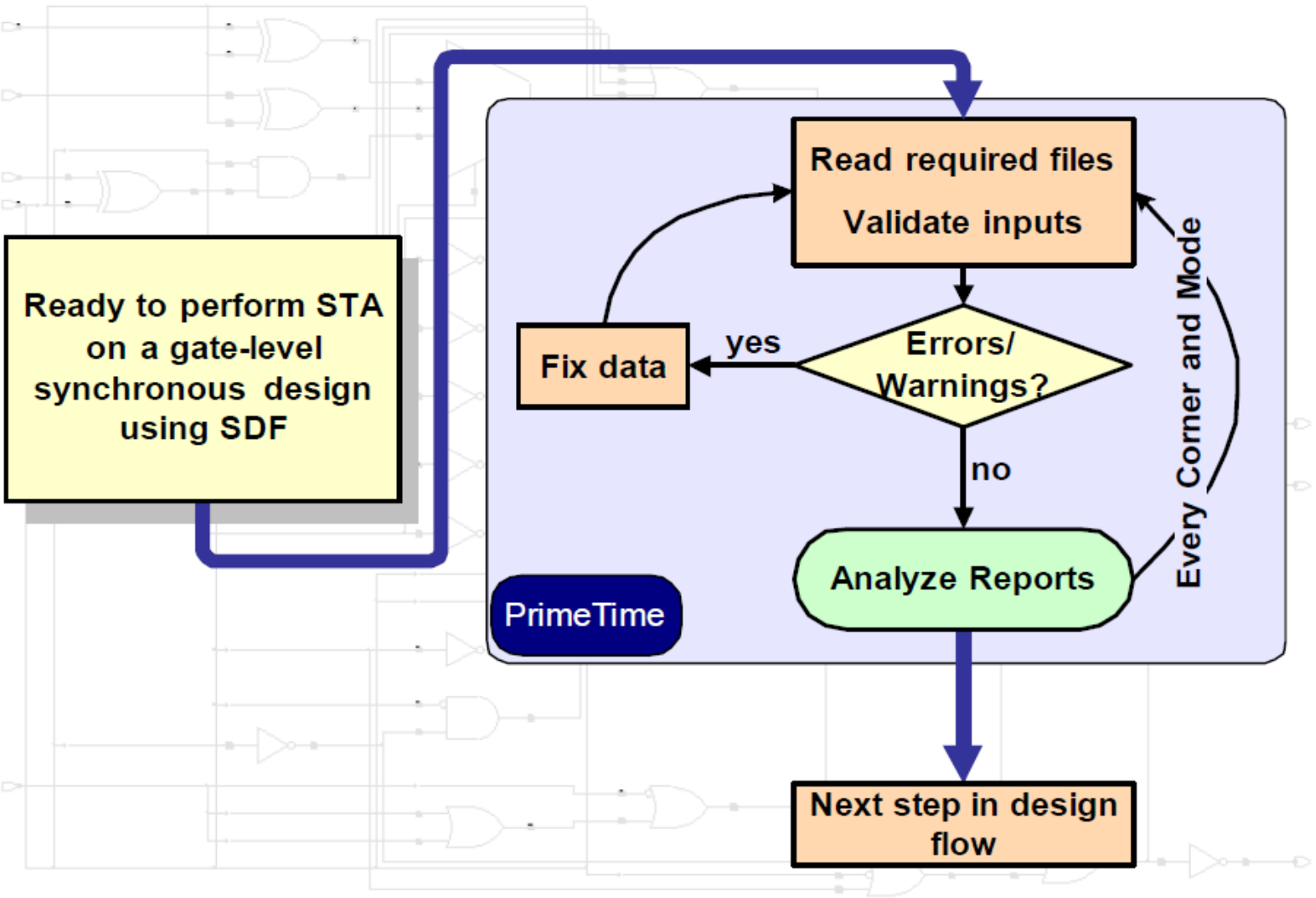
International College of Semiconductor Technology
National Chiao Tung Yang Ming University

國立陽明交通大學
NATIONAL YANG MING CHIAO TUNG UNIVERSITY

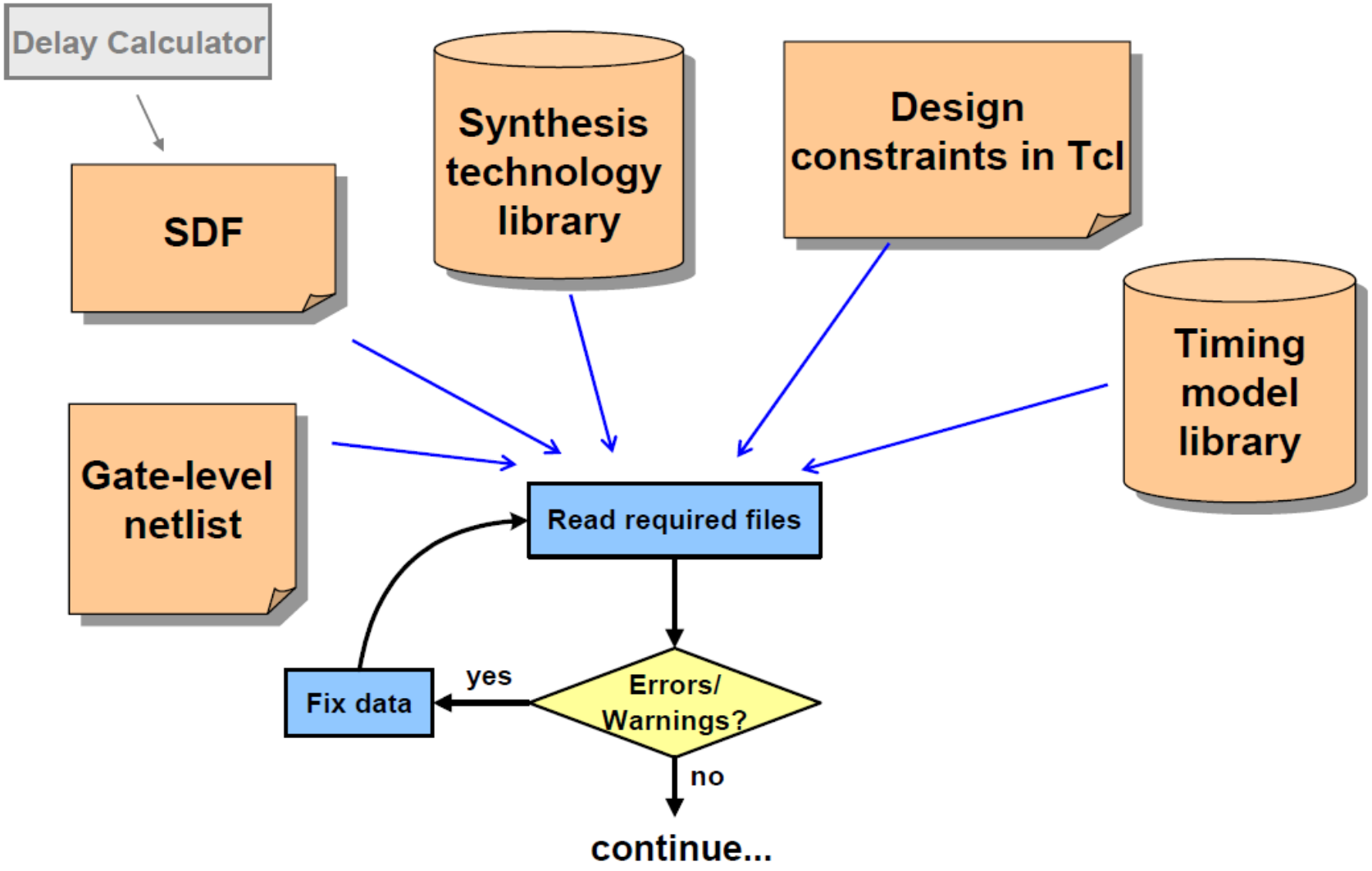
Static Verification Flow



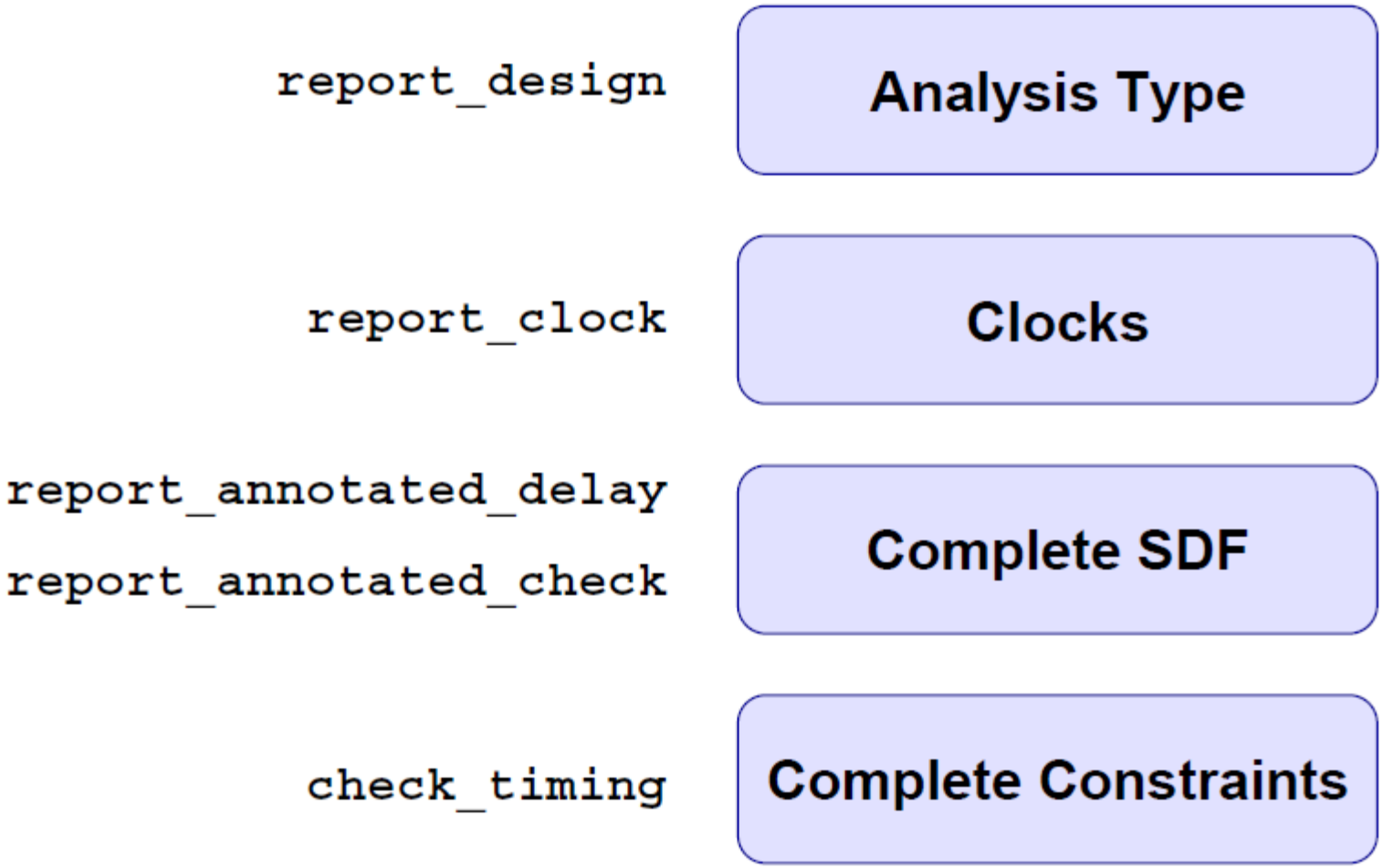
Static Timing Analysis Flow



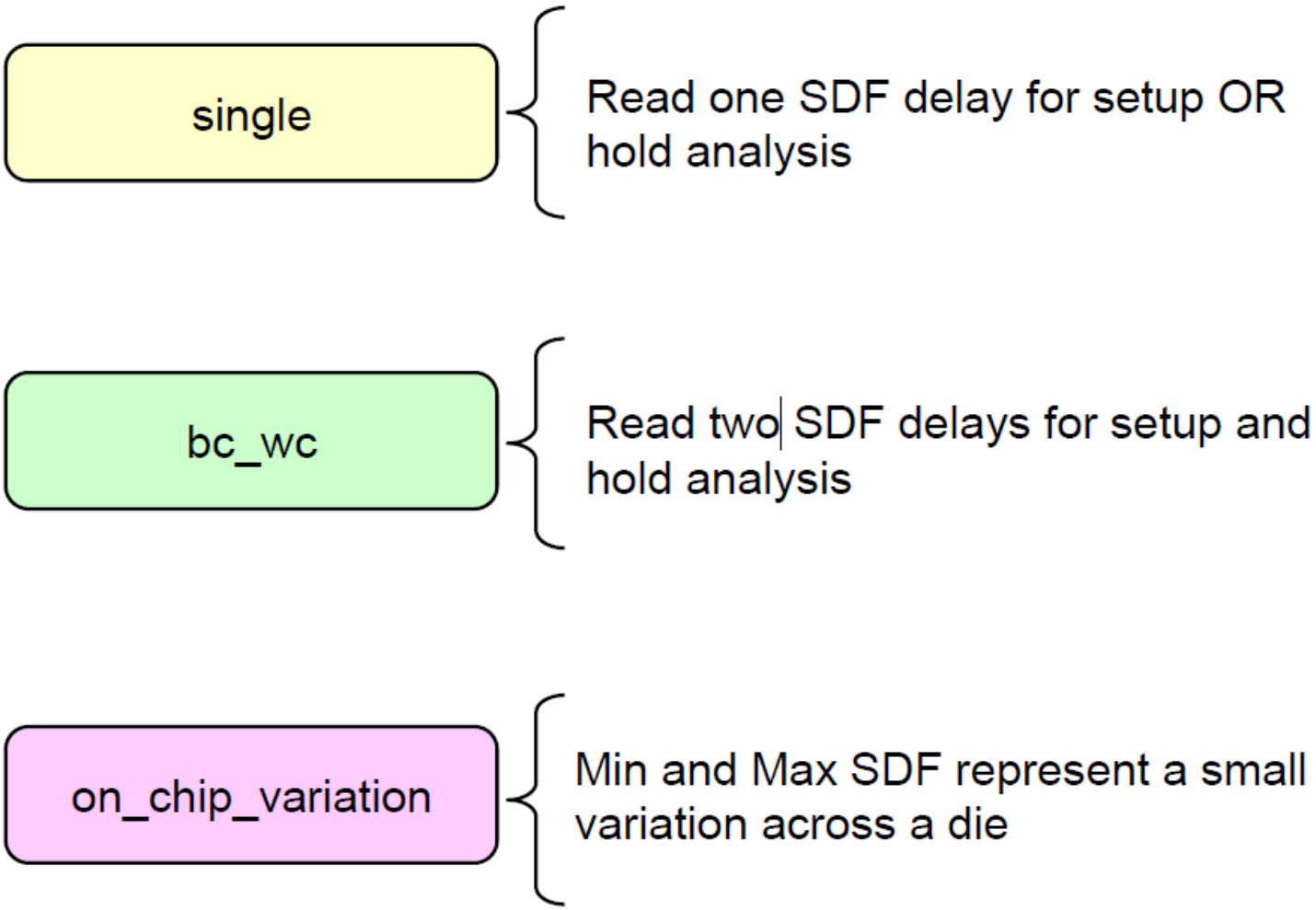
Required Input Files



Validate Complete and Correct Constraints



Three Types of Analysis



Report All Violations

- report_constraint –all_violators

```
max_delay/setup ('Clk1' group)

 Endpoint                                          Slack
 ---------------------------------------------------------------------
 B                                                 -0.50   (VIOLATED)

min_delay/hold ('Clk1' group)

 Endpoint                                           Slack
 ---------------------------------------------------------------------
 FF1/D0                                            -0.67   (VIOLATED)

sequential_clock_pulse_width

                     Required        Actual
 Pin                pulse width    pulse width       Slack
 ---------------------------------------------------------------------
 FF2/clk (high)         0.90           0.85          -0.05   (VIOLATED)
```

The Number of Violations

- report_analysis_coverage

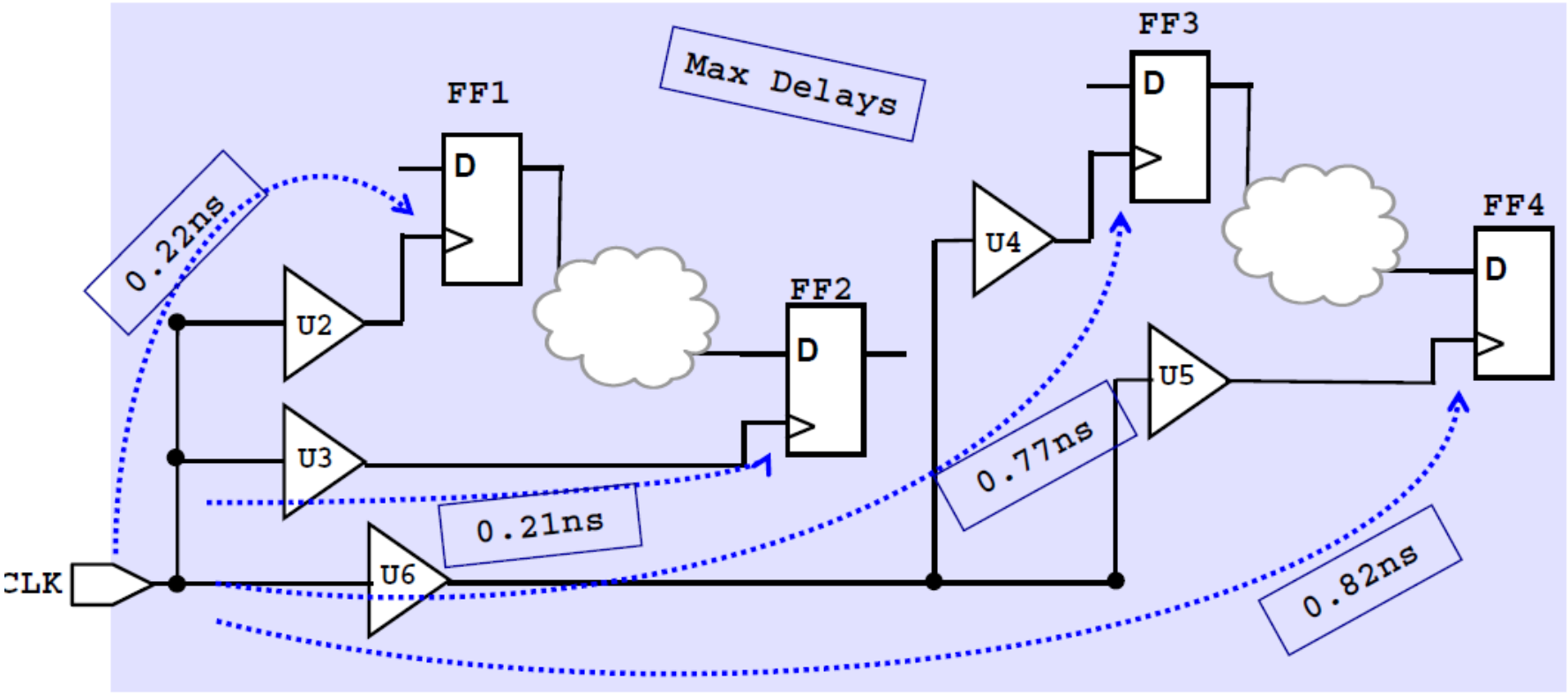
Type of Check	Total	Met	Violated	Untested
setup	6724	2366 (35%)	0 (0%)	4358 (65%)
hold	6732	2366 (35%)	0 (0%)	4366 (65%)
recovery	362	302 (83%)	0 (0%)	60 (17%)
removal	354	302 (85%)	0 (0%)	52 (15%)
min_pulse_width	4672	4310 (92%)	0 (0%)	362 (8%)
clock_gating_setup	65	65 (100%)	0 (0%)	0 (0%)
clock_gating_hold	65	65 (100%)	0 (0%)	0 (0%)
out_setup	138	138 (100%)	0 (0%)	0 (0%)
out_hold	138	74 (54%)	64 (46%)	0 (0%)
All Checks	19250	9988 (52%)	64 (0%)	9198 (48%)

More Details: Path Timing Reports

- **Default: Returns the worst path for max analysis for:**
  - Each clock
  - Recovery checks
  - Clock gating checks
- **Customize with MANY different switches:**
  - Setup versus hold reports
  - Increase the significant digits
  - Focus on specific paths
  - Increase the # of generated reports
  - Include net fanout
  - Expand the calculated clock network delay

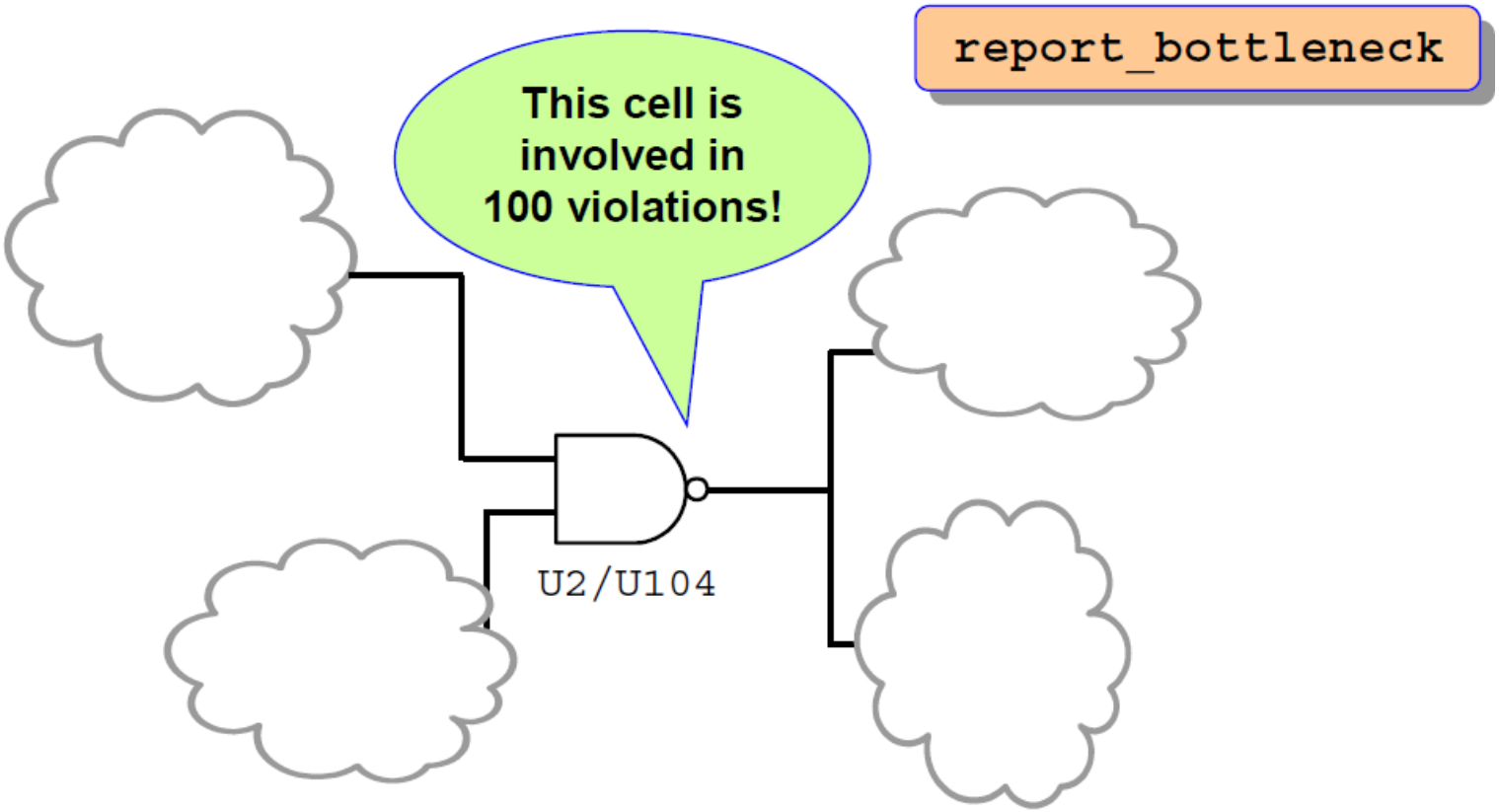
Clock Network Reports

- **report_clock_timing –type skew**
  - For each clock, report *REAL* skew



Bottleneck Analysis

- **Identify cells involved in multiple violations**
  - Use the results to determine cells to buffer or upsize



Specify Timing Assertions

- ***Example:***
  - Set up the basic timing assertions 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 -max 5.5 [get_clocks CLK]
pt_shell> set_clock_transition -min 0.25 [get_clocks CLK]
pt_shell> set_clock_transition -max 0.3 [get_clocks CLK]
```

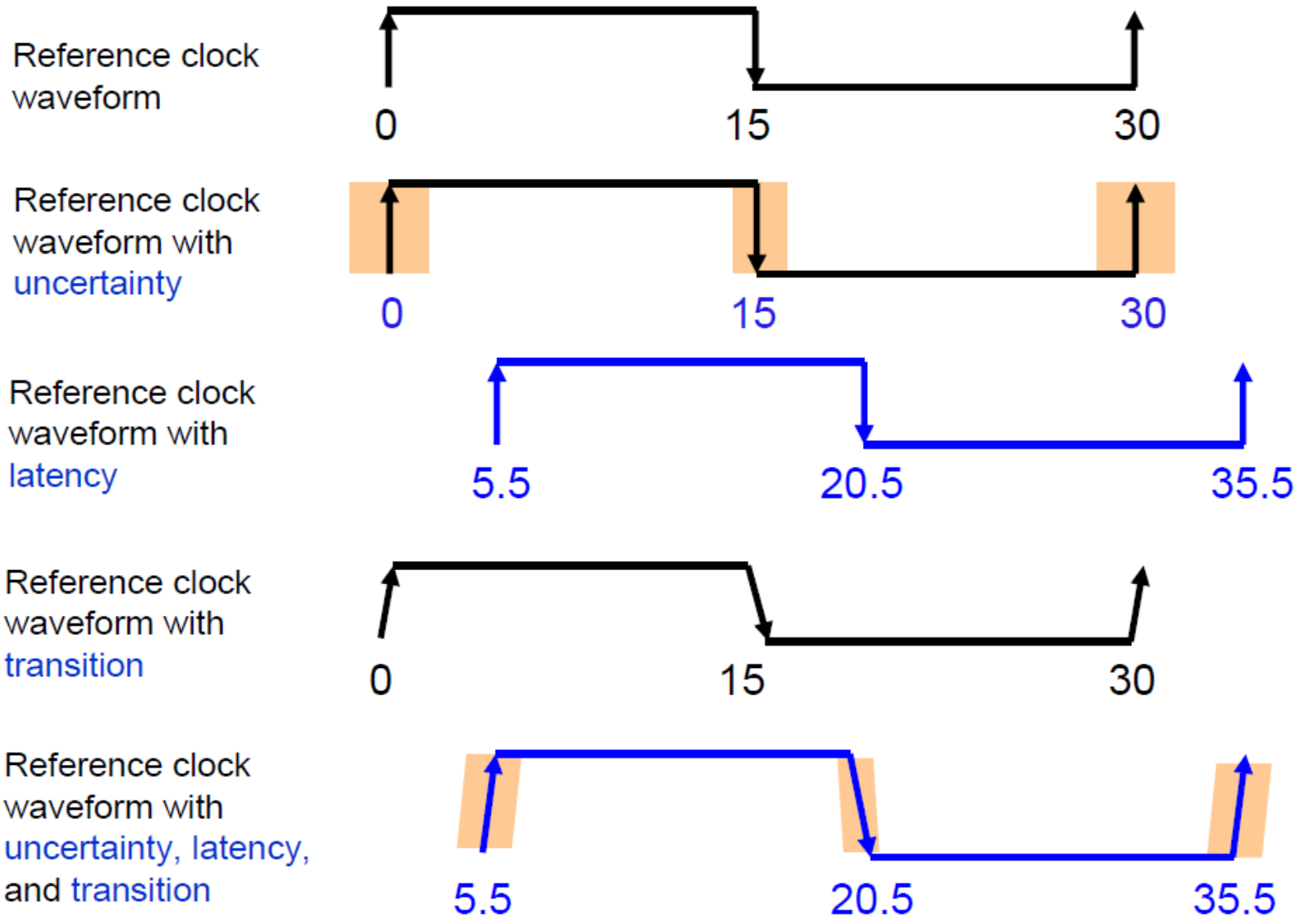
  - For post layout clock tree:

```
set_propagated_clock <clock_object_list>
```

or

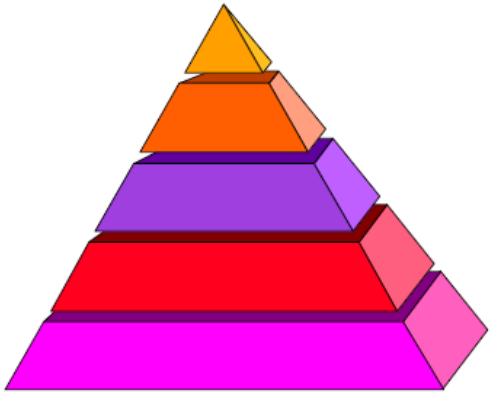
```
set timing_all_clocks_propagated true
```

Specify Timing Assertions



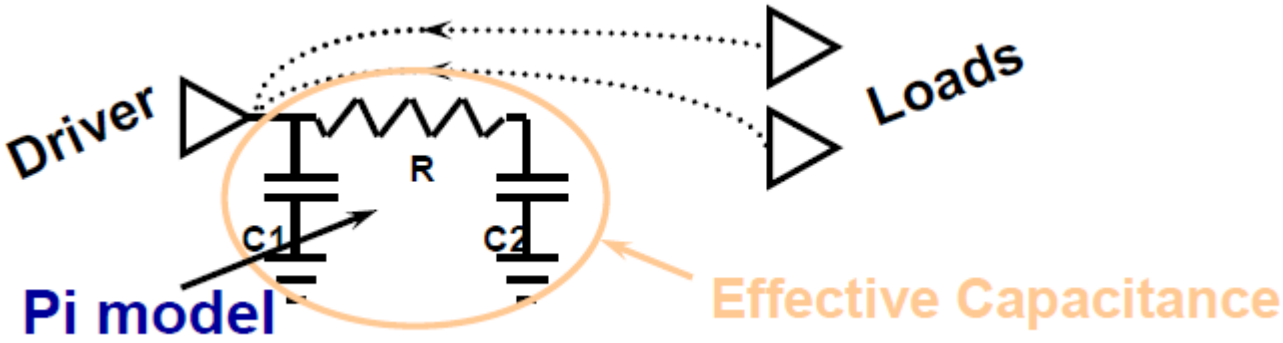
Advanced Timing Analysis

- Analysis Modes
- Data to Data Checks
- Case Analysis
- Multiple Clocks per Register
- Minimum Pulse Width Checks
- Derived Clocks
- Clock Gating Checks
- Netlist Editing
- Report_clock_timing
- Clock Reconvergence Pessimism
- Worst-Arrival Slew Propagation
- Debugging Delay Calculation

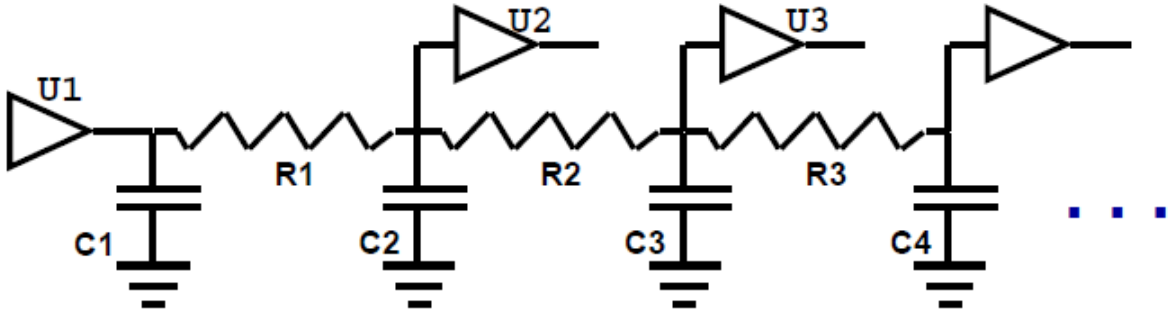


Parasitics Reduced and Distributed Parasitic Files

- *Reduced* format annotates an RC pi model, and computes the effective capacitance.

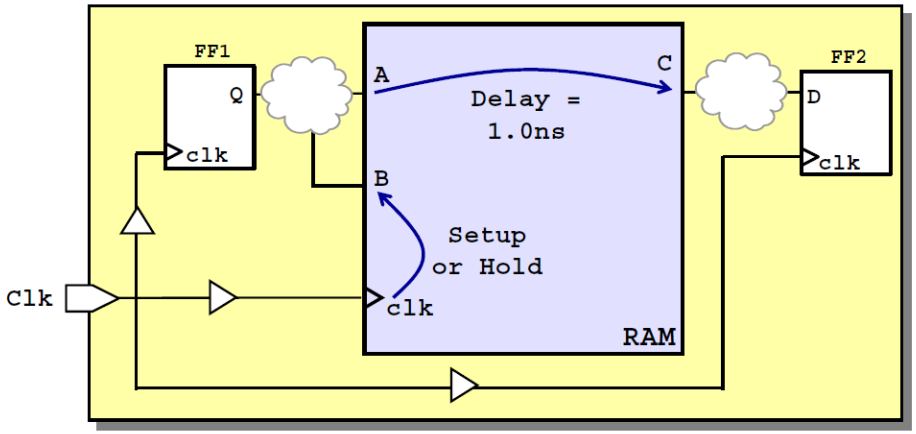


- *Distributed* format enables PrimeTime to annotate each physical segment of the routed netlist (most accurate form of RC backannotation)



PrimeTime Timing Models Support

- 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)**
  - **Stamp Model**
- **Timing models are cells with many timing arcs:**
  - "Flip-flop" with setup and hold timing checks
  - "Delay cell" included along the data arrival time



Timing Model Usage Scenario

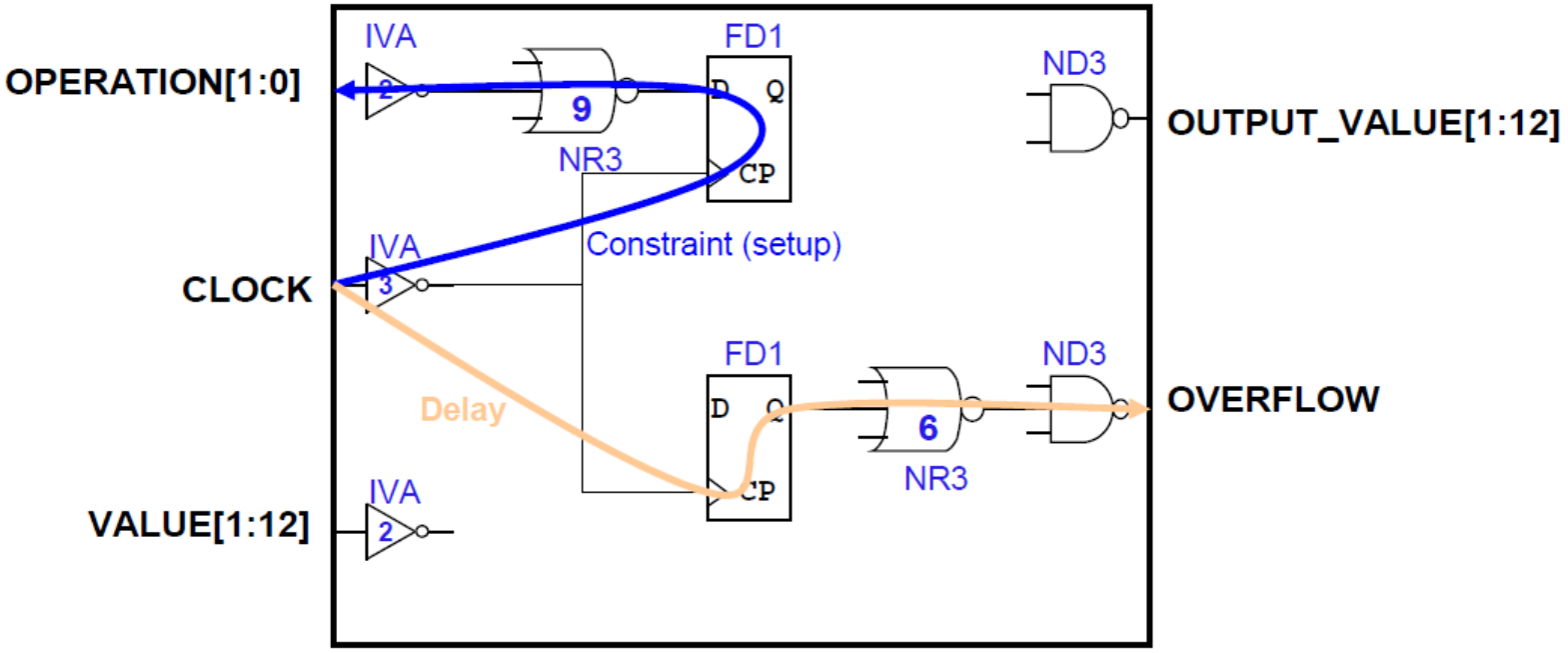
Usage Scenario	Appropriate Model
Top-Down Design	Quick Timing Models
IP Reuse	ETMs
Interface to non-STA and 3rd party tools	ETMs
Synthesis Tasks	ILMs / ETMs
Chip-Level STA	ILMs
Memory and Datapath	Stamp Models

Quick Timing Models (QTMs)

- 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 specs generated with a lot less effort**
  - **apply chip level timing constraints and time the whole design**
  - **discover violators up front**

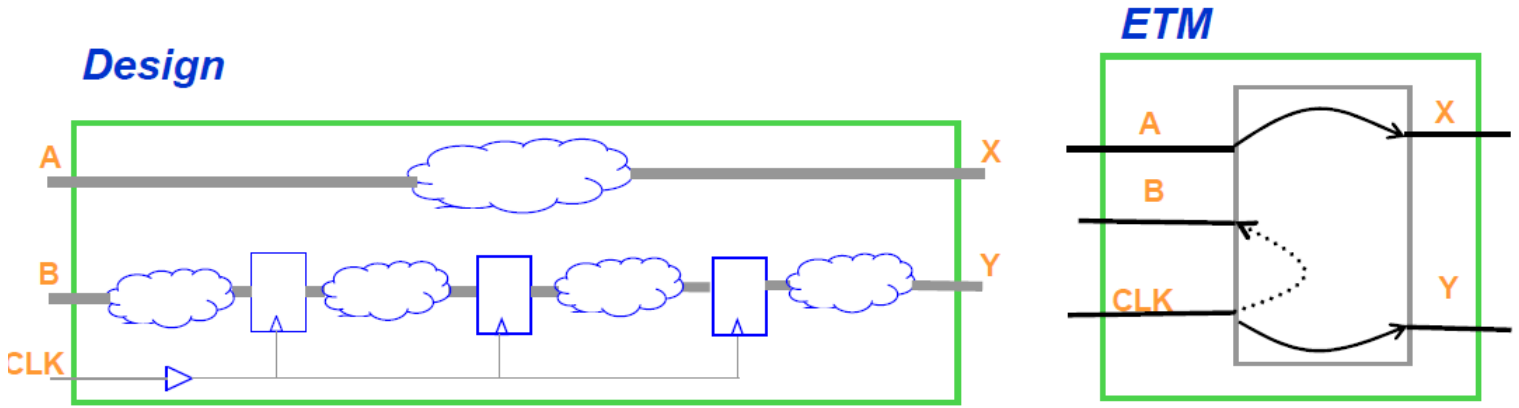
Quick Timing Models

- 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 or Stamp format



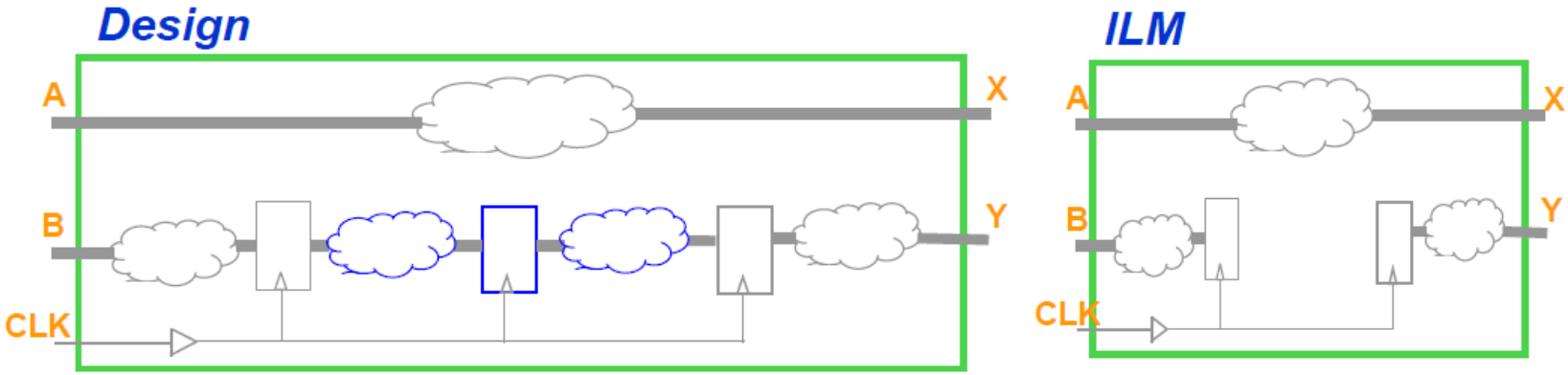
Extracted Timing Models (ETM)

- 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



Interface Logic Models (ILM)

- 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

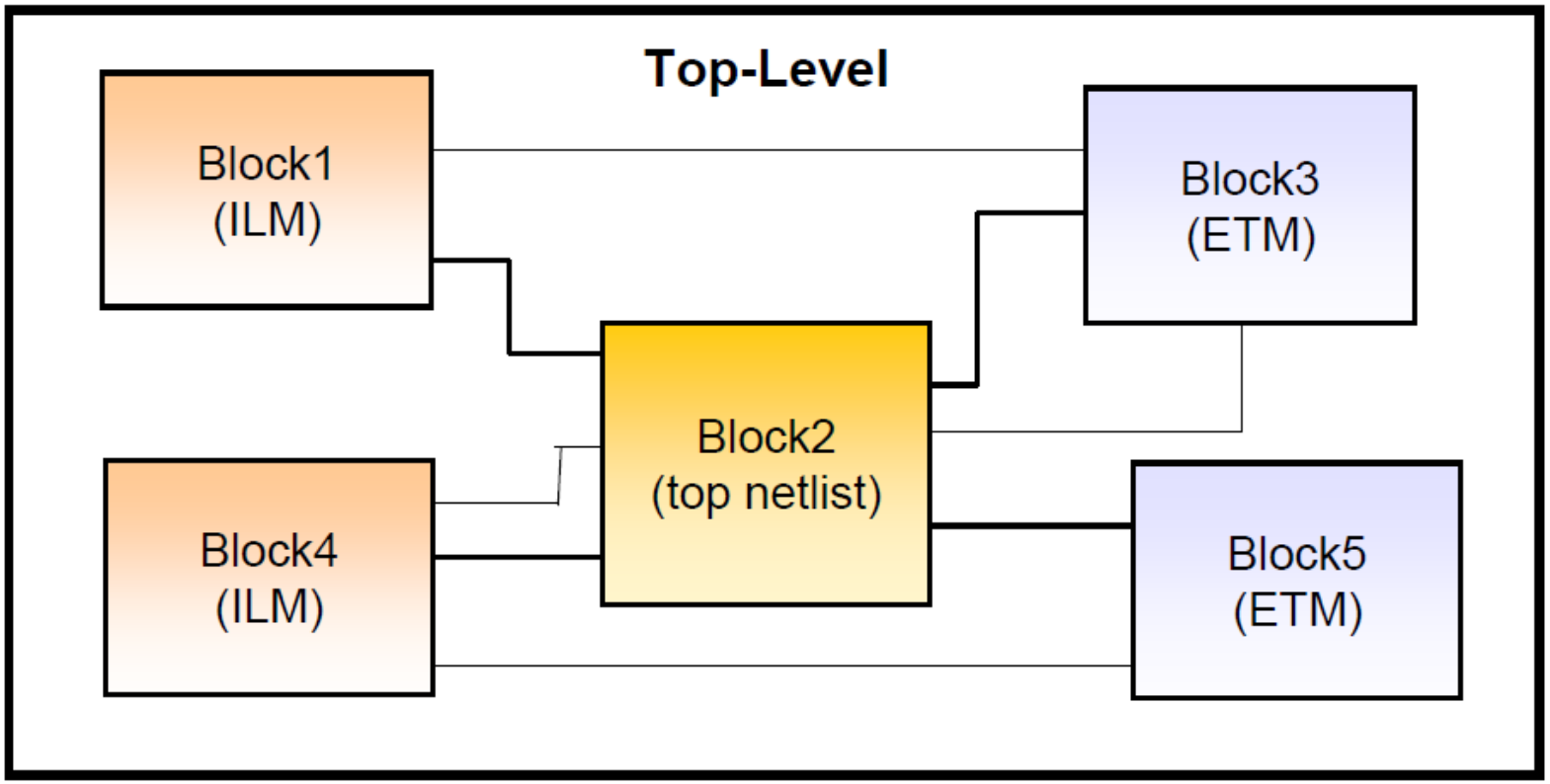


Stamp Modeling

- Generally created for transistor-level designs, where there is no gate-level netlist. Stamp timing models are usually created by core or technology vendors, as a compiled db.
- **Capabilities include the ability to model:**
  - pin-to-pin timing arcs
  - setup and hold data
  - pin capacitance and drive
  - mode information
  - tri-state outputs
  - internally generated clocks
- **Stamp models co-exist with the Library Compiler**
  - **.lib models**

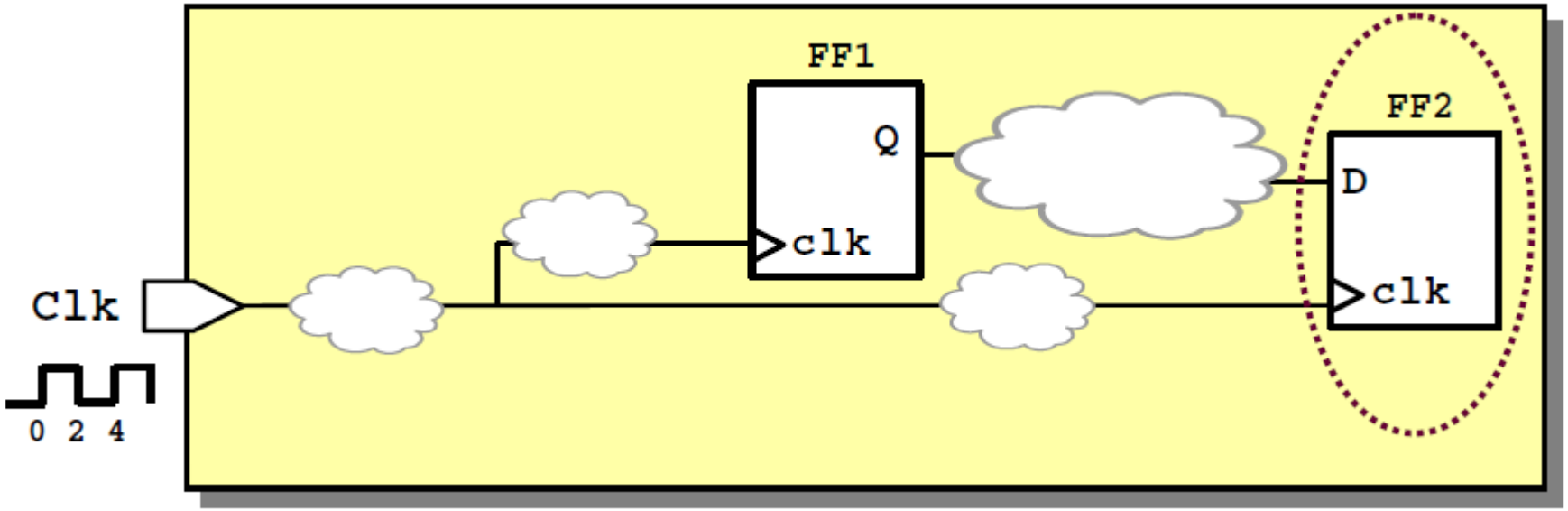
Chip-Level Verification using Models

- Using ILMs and ETMs to address capacity and timing issues in multimillion gate design

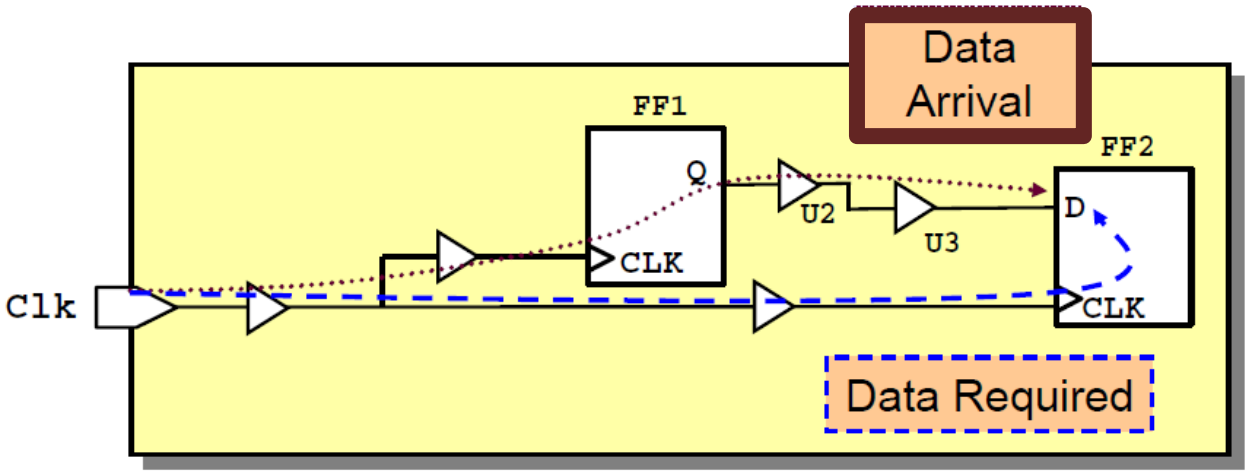


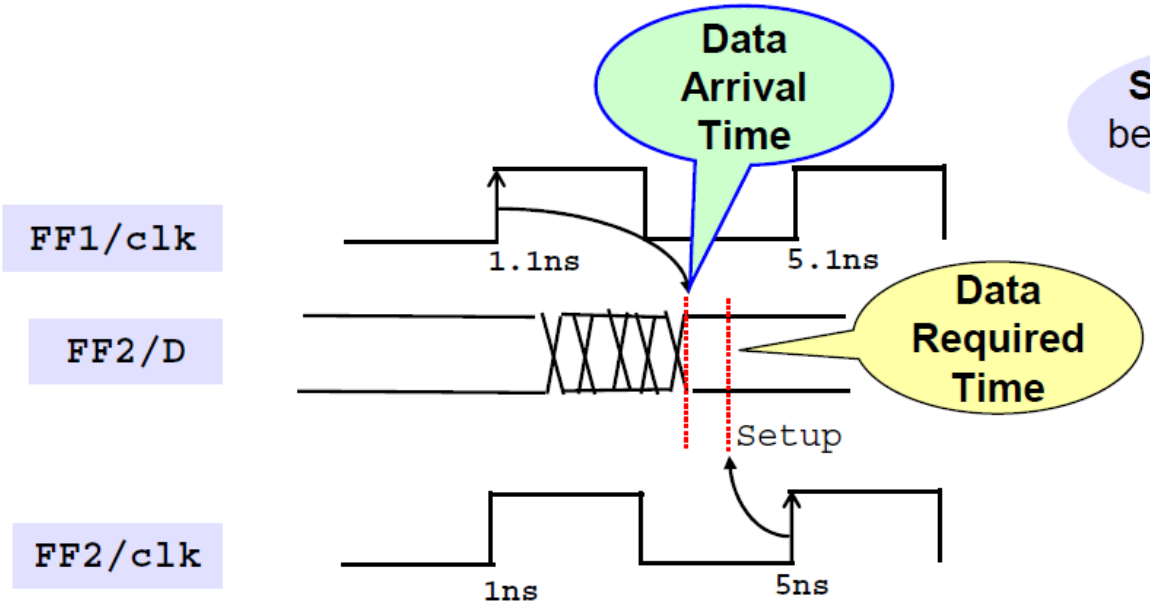
Timing Verification of Synchronous Designs

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



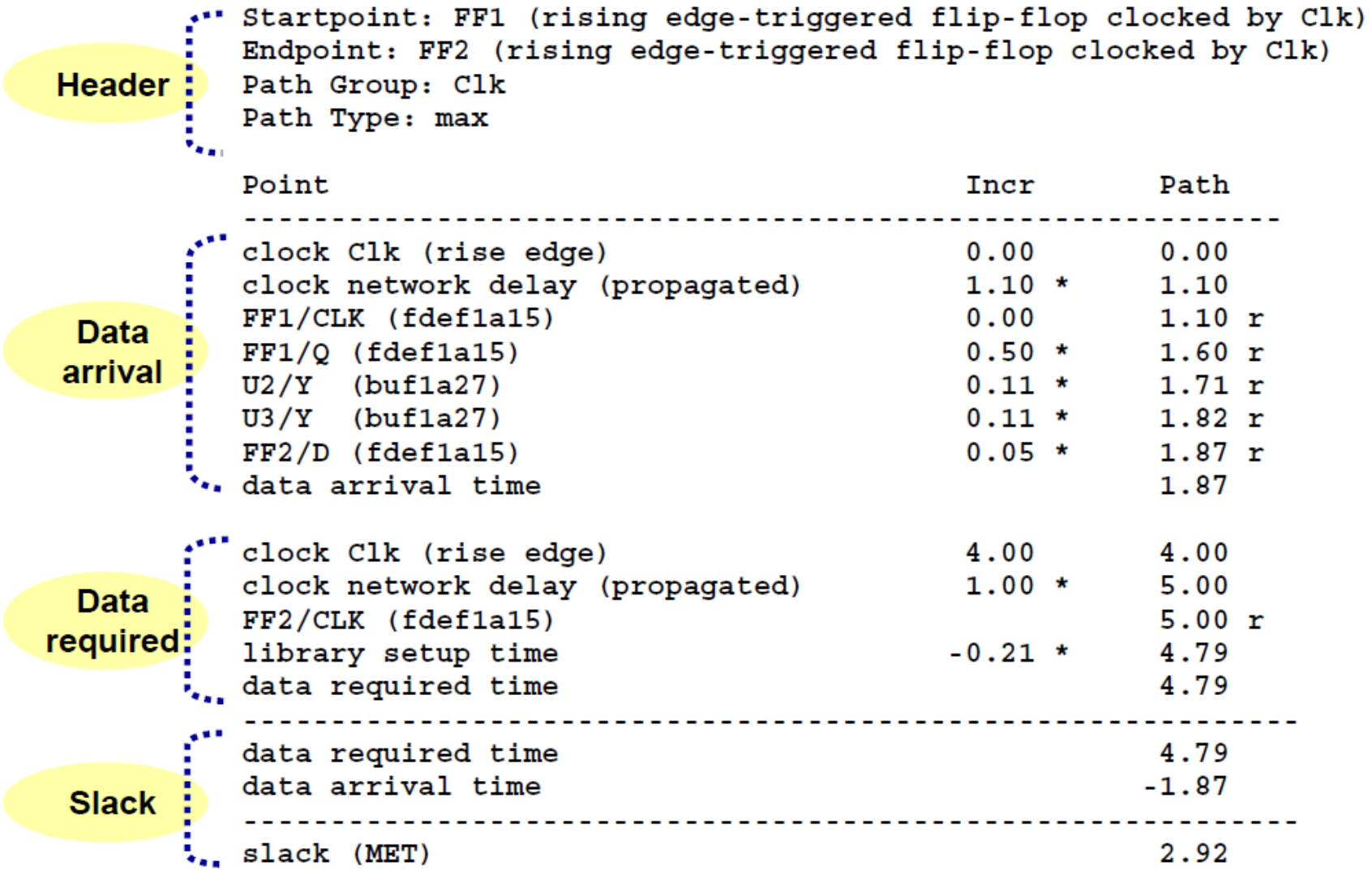
Arrival Time and Required Time



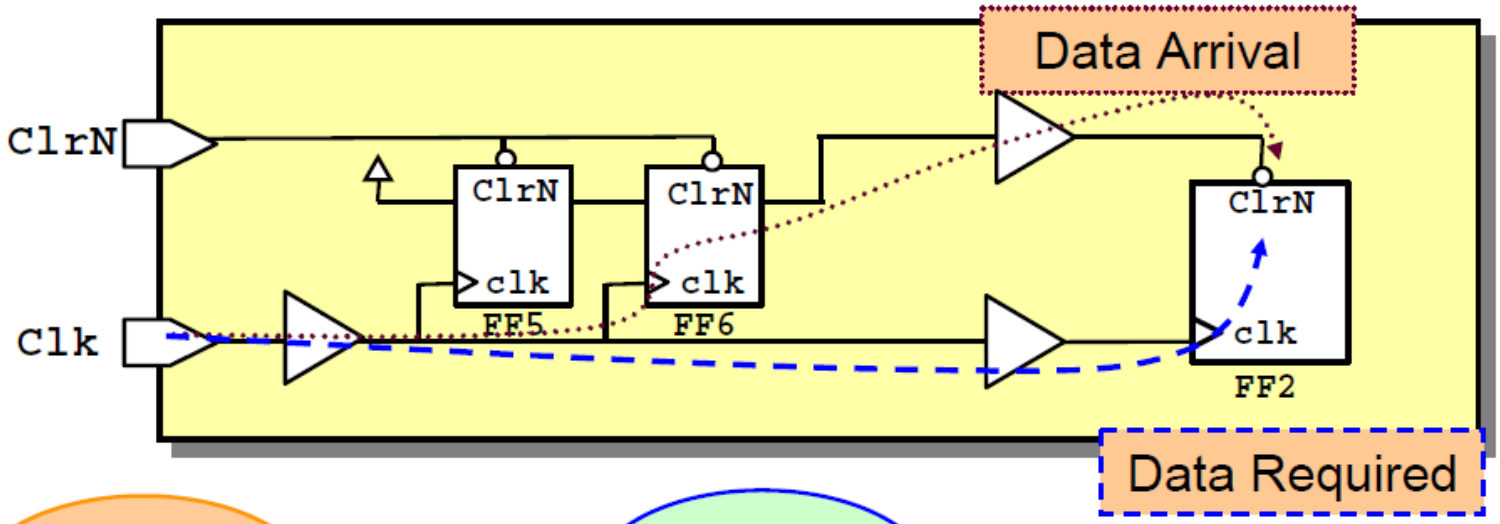


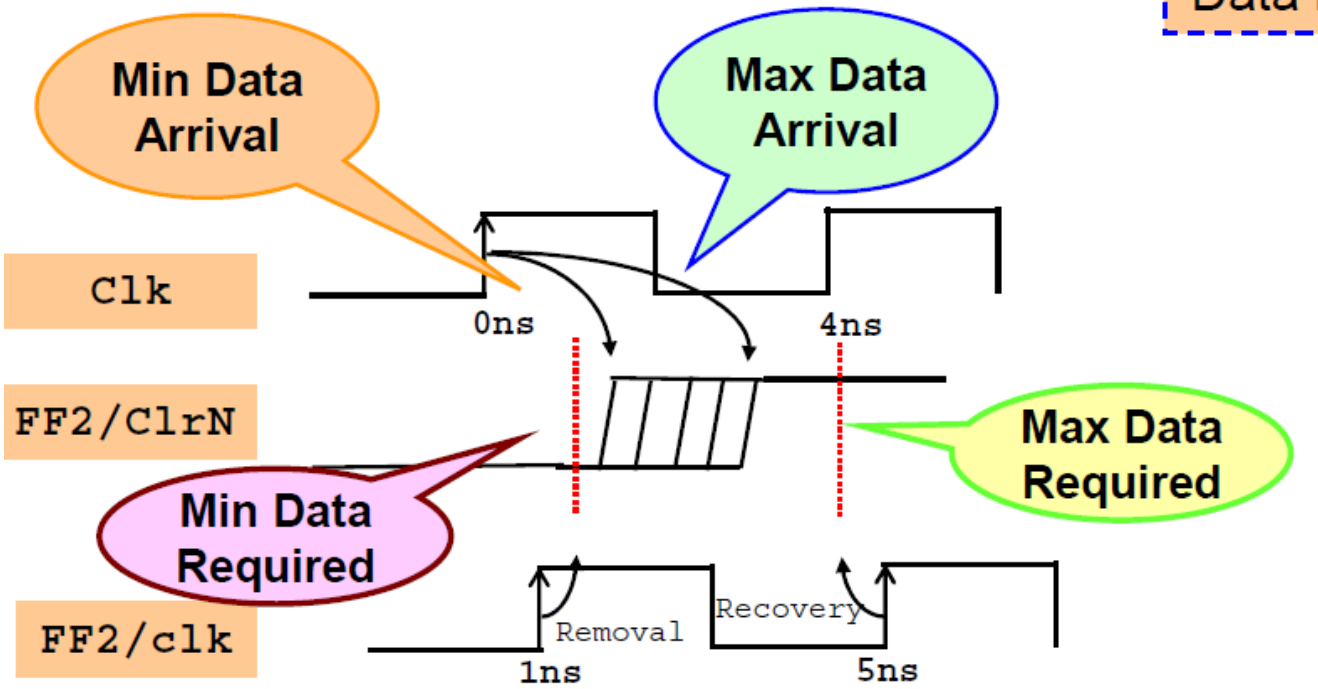
Slack is the difference between data arrival and data required.

Four Sections in a Timing Report



Asynchronous Clear/Reset Pins



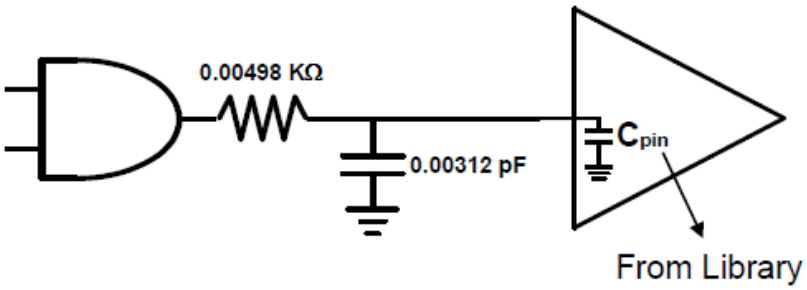


Estimating Rnet and Cnet Pre-layout

- Extraction data of already routed designs are used to build a lookup table called the wire load model
- WLM is based on the statistical estimates of R and C based on “Net Fanout”

Net Fanout	Resistance KΩ	Capacitance pF
1	0.00498	0.00312
2	0.01295	0.00812
3	0.02092	0.01312
4	0.02888	0.01811

Wire Load Model (RC)

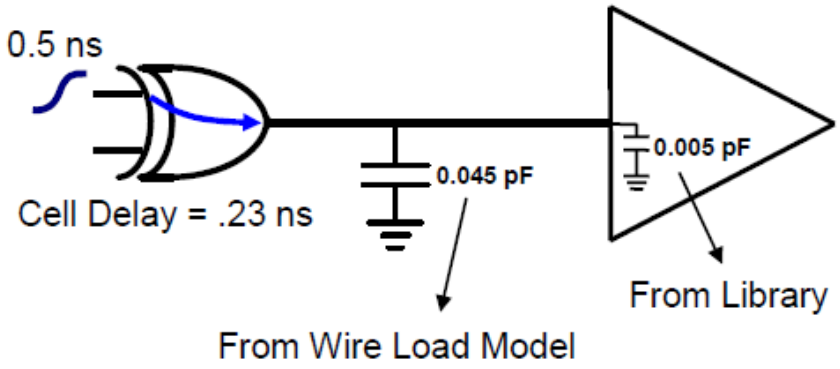


Estimated RCs are represented as wire load model

Cell Delay Calculation

- Cell delays are calculated from a Non Linear Delay Model (NLDM) table in the technology library
- Tables are indexed by input transition and total output load for each gate

Cell Delay = f (Input Transition Time, Output Load)



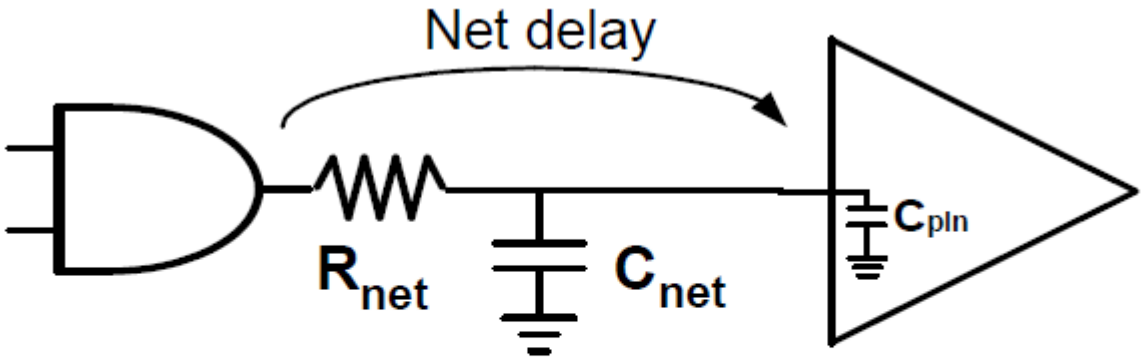
SPICE

Input Trans (ns) \ Output Load (pF)	.005	.05	.10	.15
0.0	.1	.15	.2	.25
0.5	.15	.23	.3	.38
1.0	.25	.4	.55	.75

Cell Delay (ns)

Net Delay Calculation

- Net delay is the "time-of-flight" due to the net's RC
- Net's RC is obtained from wire load model for pre-layout design



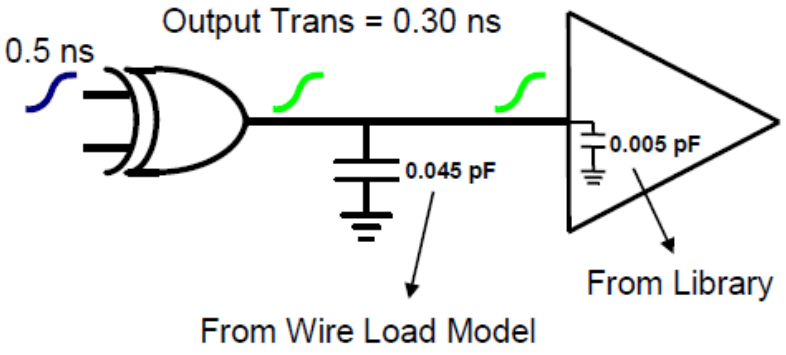
$$\text{Net Delay} = f\,(R_{net},\ C_{net} + C_{pin})$$

Post-layout RCs are extracted as a parasitic file.

Output Transition Calculation

- There is another NLDM table in the library to calculate output transition
- Output transition of a cell becomes the input transition of the next cell down the chain

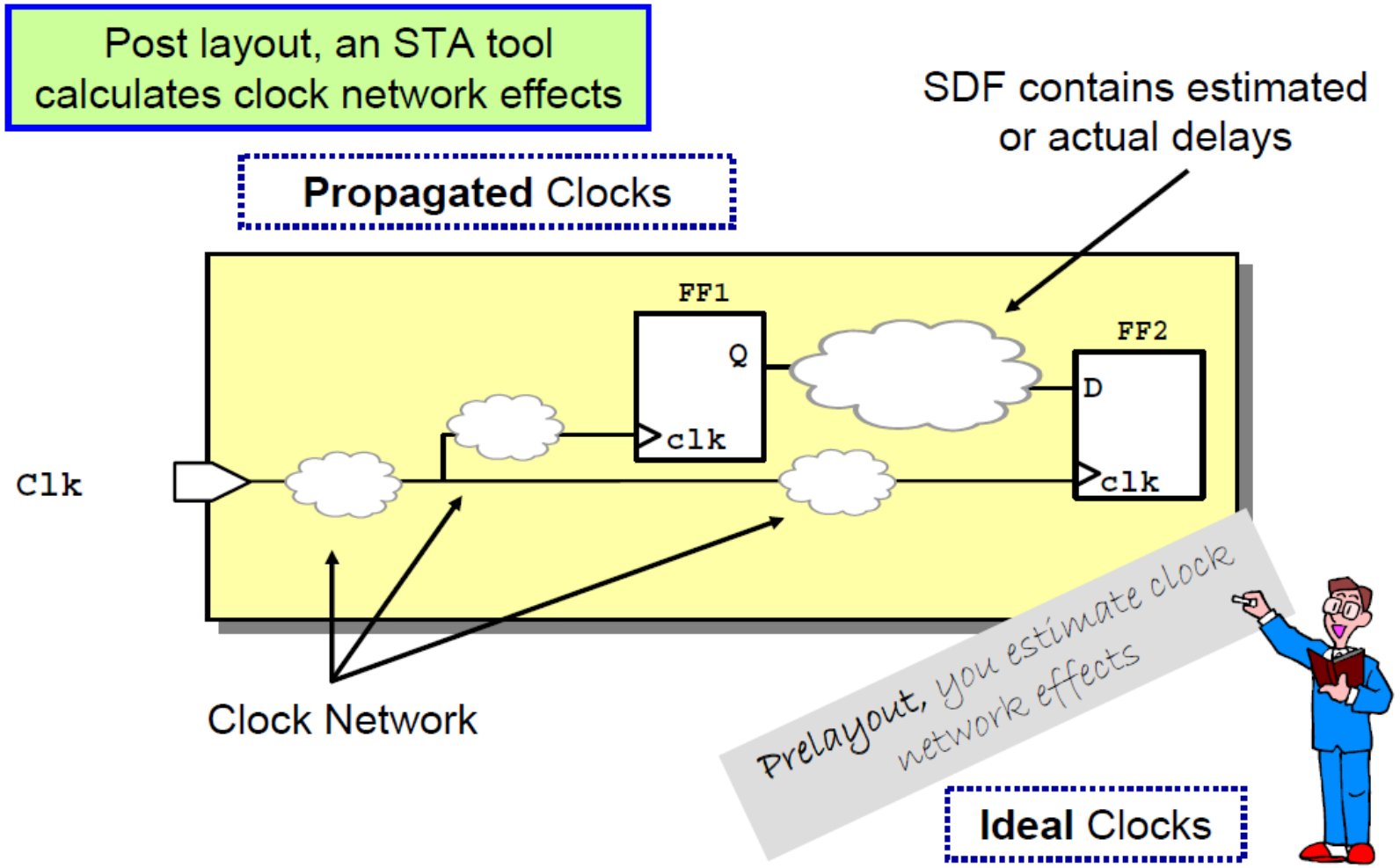
Output Transition = f (Input Transition Time, Output Load)



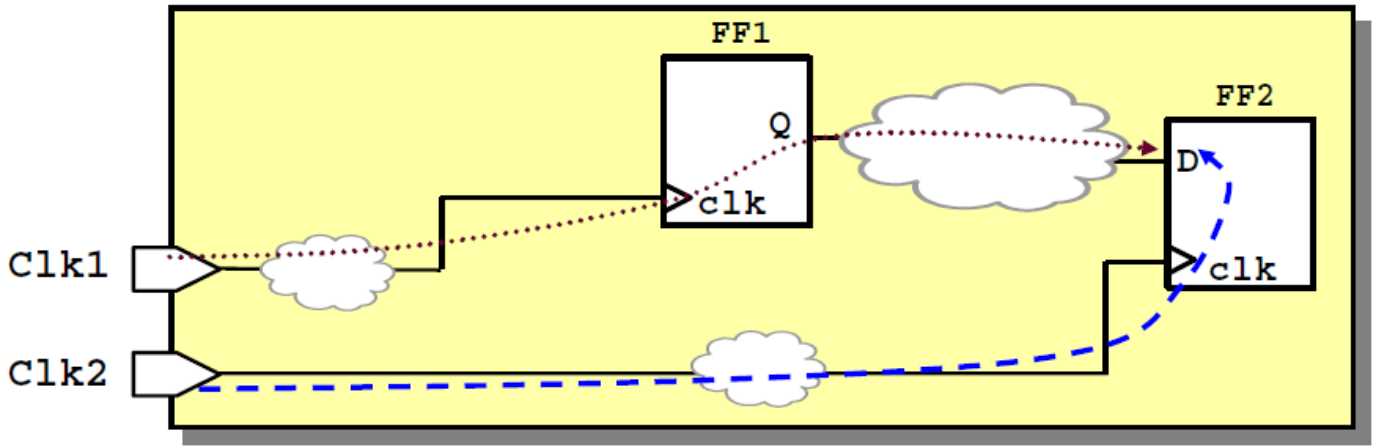
SPICE	Output Load (pF)			
Input Trans (ns)	.005	.05	.10	.15
0.00	0.10	0.20	0.37	0.60
0.50	0.18	0.30	0.49	0.80
1.00	0.25	0.40	0.62	1.00

Output Transition (ns)

What About Pre and Post Layout STA?

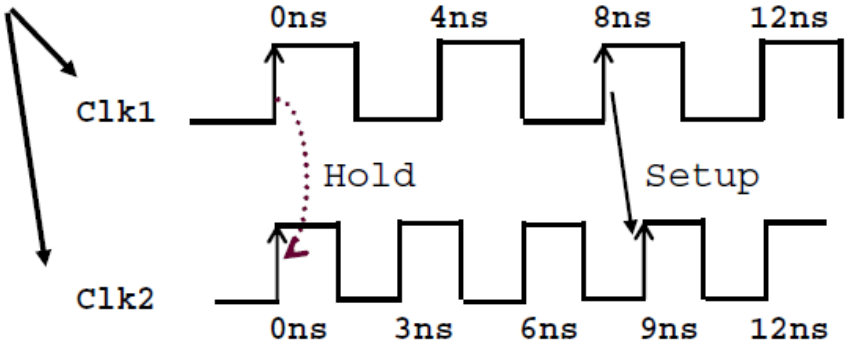


What About Multi-Frequency Clocks?



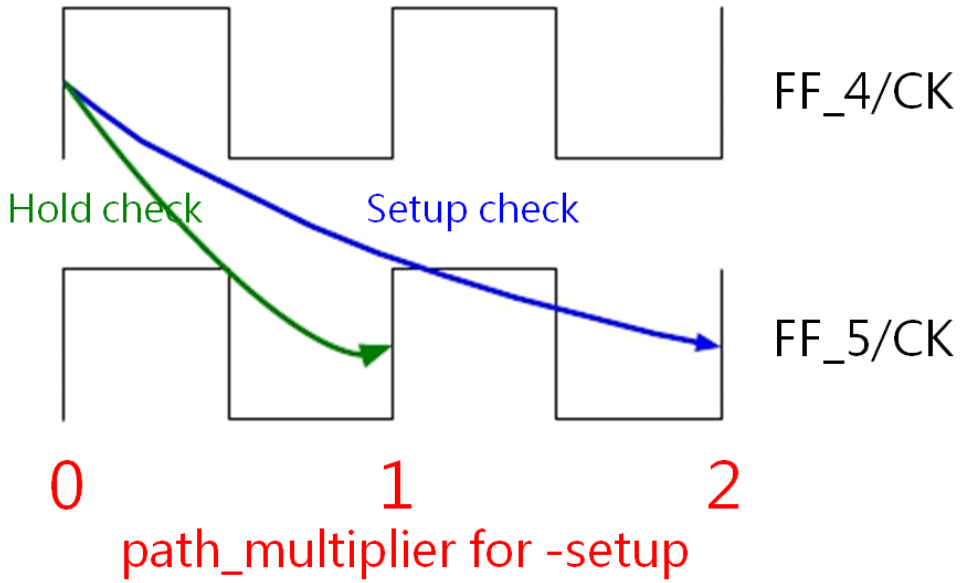
Create both clocks

Base Period is from 0ns to 12ns



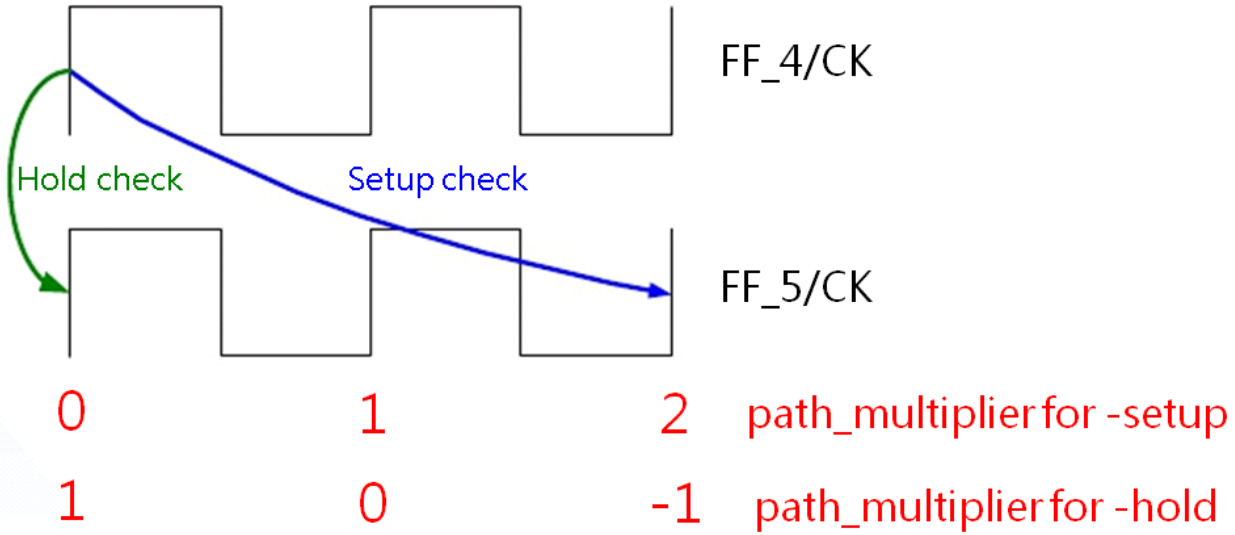
Multicycle Path of Single Clock Domain

- **Specify multiple path**
  - Modify only setup time check edge
    - set_multicycle_path 2 –from FF_4/CK –to FF_5/D
  - Hold time check edge will be changed too!!



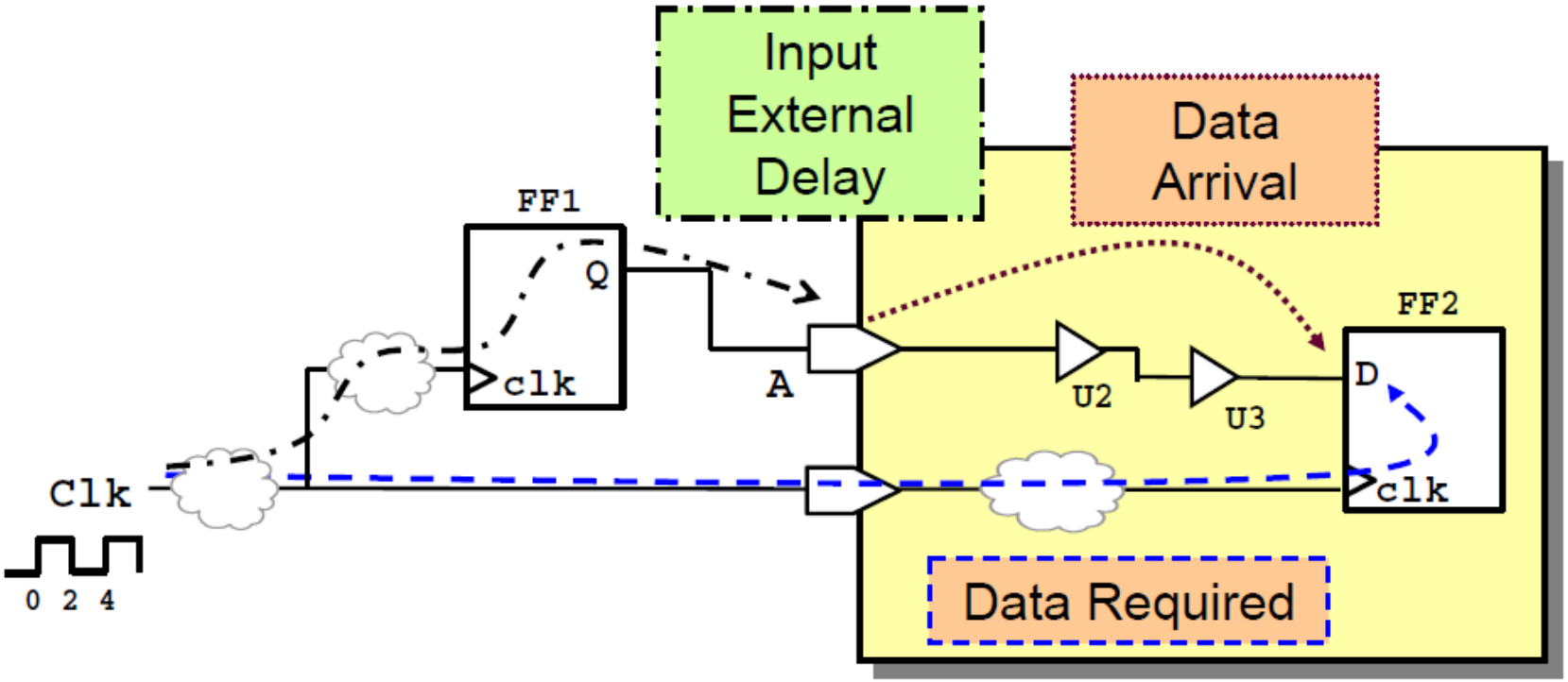
Multicycle Path of Single Clock Domain

- **Specify multiple path**
  - Modify both setup time and hold time check edge
    - set_multicycle_path 2 –from FF_4/CK –to FF_5/D
      set_multicycle_path 1 –hold –from FF_4/CK –to FF_5/D
  - Timing check edge is what we desire



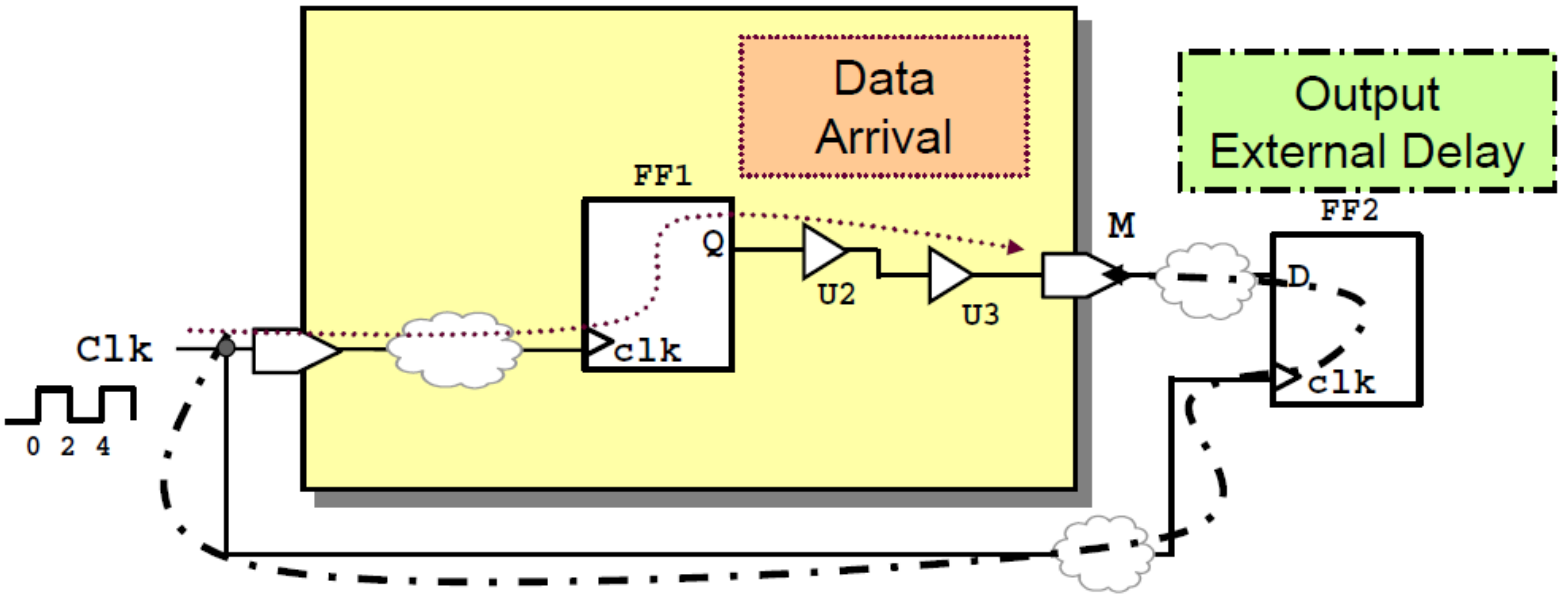
What About Interface Paths: Input Ports?

You specify the arrival times at the input ports of the design.

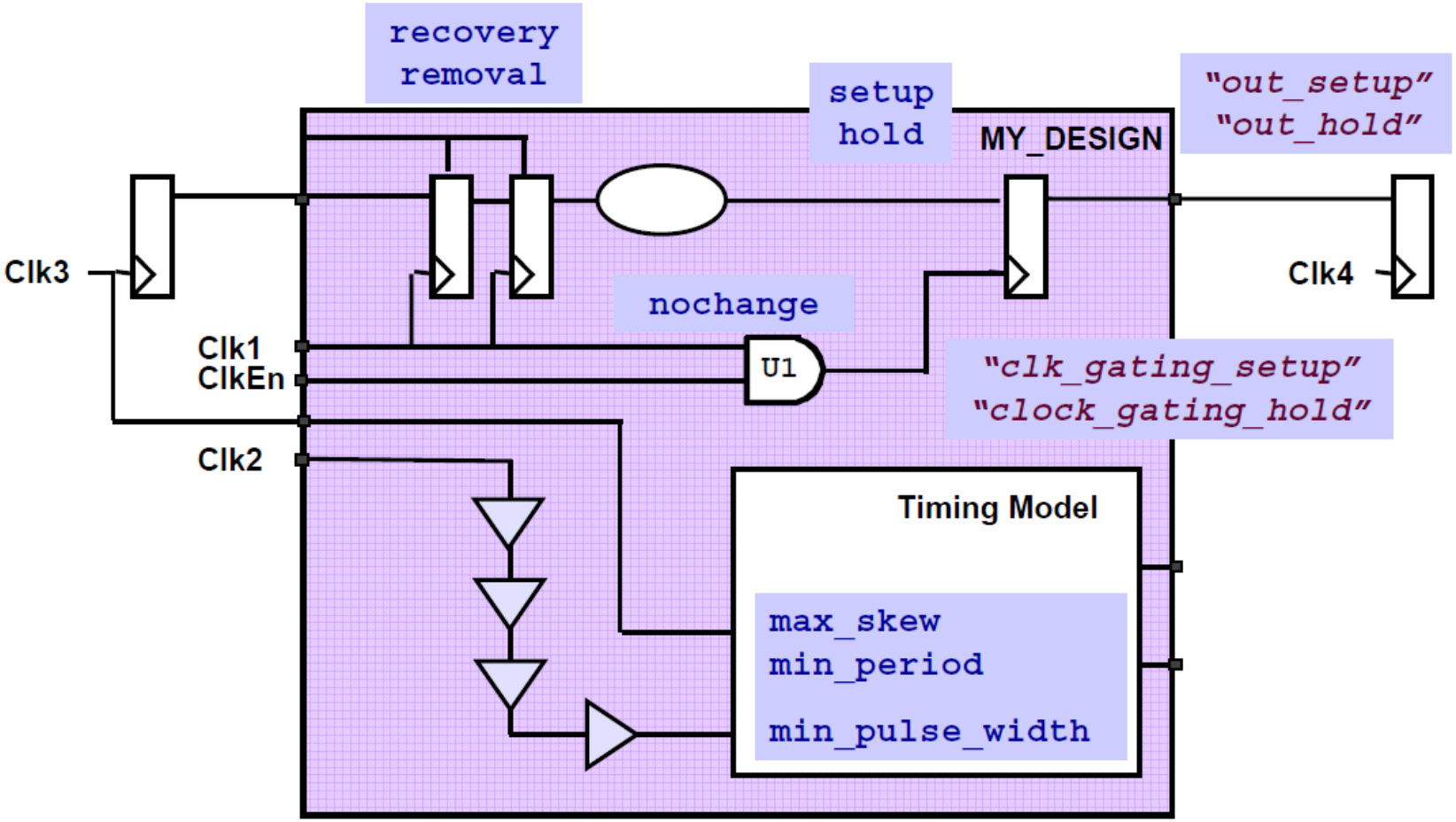


What About Interface Paths: Output Ports?

You specify the path required time at the output ports of the design.



Other Timing Checks Verified by STA

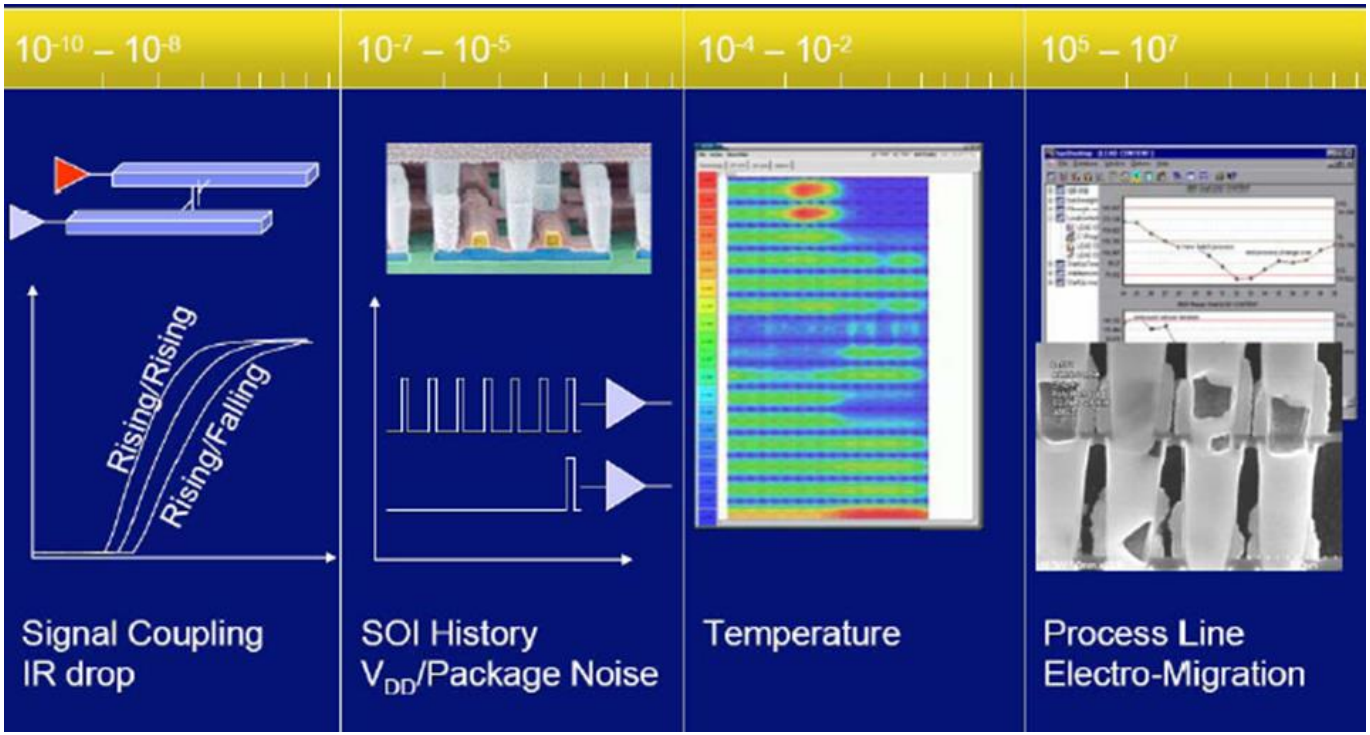


"Timing checks": specified by the user

Timing checks: specified by the vendor

Delay Variations

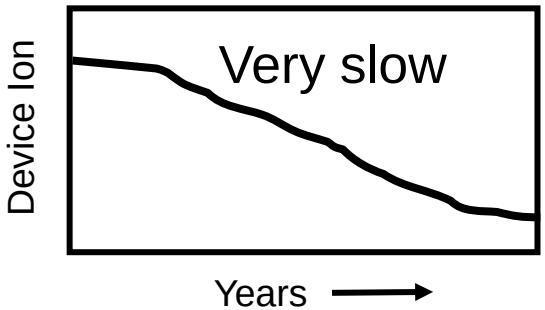
- Given corner data below, which combinations are expected to lead to worst and best gate delays?
  - **Process, voltage, temperature variations**



P, V, T Variations

Process

- Die-to-die variation
- Within-die variation
- Static for each die



Time dependent

Degradation

Aging

NBTI

Voltage

- Chip activity change
- Current delivery—RLC
- Dynamic: ns to 10-100us
- Within-die variation

Temperature

- Activity & ambient change
- Dynamic: 100-1000us
- Within-die variation

Multi-Mode Multi-Corner

- Corner: defined as a set of libraries characterized for process, voltage and temperature variations.
  - Corners are not dependent on functional settings
  - To capture variations in the manufacturing process, along with expected variations in the ***voltage and temperature*** of the environment in which the design will operate.
- Mode: defined by a unique set of clocks, supply voltages, and timing constraints in similar operating conditions.
  - It can also have annotation data, such as SDF or parasitic files.

MCMM (or MMMC) Optimization

- MCMM optimization is useful for designs that can operate in many modes such as test mode, low-power active mode, stand-by mode and so on.
- Used along with specification of power intent in the Unified Power Format (UPF), it serves as the key enabling technology for performing dynamic voltage and frequency scaling (DVFS) design realization

Timing Margins

- The setup and hold times must be analyzed simultaneously for different combinations of library models, voltages, and interconnect (RC) corners.

	Single Core Design			Core + 1 Island				Core + 2 Islands				
	Lib	Core	RC	Lib	Core	Vdd1	RC	Lib	Core	Vdd1	Vdd2	RC
Setup1	Max	1.2	Max	Max	1.2	0.9	Max	Max	1.2	0.9	0.9	Max
Setup2	Max	1.2	Min	Max	1.2	0.9	Min	Max	1.2	0.9	0.9	Min
Hold1	Min	1.8	Min	Min	1.8	1.5	Min	Min	1.8	1.5	1.5	Min
Hold2	Min	1.8	Max	Min	1.8	1.5	Max	Min	1.8	1.5	1.5	Max
Setup1	—	—	—	Max	1.2	0	Max	Max	1.2	0	1.2	Max
Setup2	—	—	—	Max	1.2	0	Min	Max	1.2	0	1.2	Min
Hold1	—	—	—	Min	1.8	0	Min	Min	1.8	0	1.8	Min
Hold2	—	—	—	Min	1.8	0	Max	Min	1.8	0	1.8	Max
Setup1	—	—	—	—	—	—	—	Max	1.2	0.9	1.2	Max
Setup2	—	—	—	—	—	—	—	Max	1.2	0.9	1.2	Min
Hold1	—	—	—	—	—	—	—	Min	1.8	1.5	1.8	Min
Hold2	—	—	—	—	—	—	—	Min	1.8	1.5	1.8	Max
Setup1	—	—	—	—	—	—	—	Max	1.2	0	0.9	Max
Setup2	—	—	—	—	—	—	—	Max	1.2	0	0.9	Min
Hold1	—	—	—	—	—	—	—	Min	1.8	0	1.5	Min
Hold2	—	—	—	—	—	—	—	Min	1.8	0	1.5	Max