



國立臺灣大學
National Taiwan University

A Quick Journey through Timing Analysis

Iris Hui-Ru Jiang

Department of Electrical Engineering

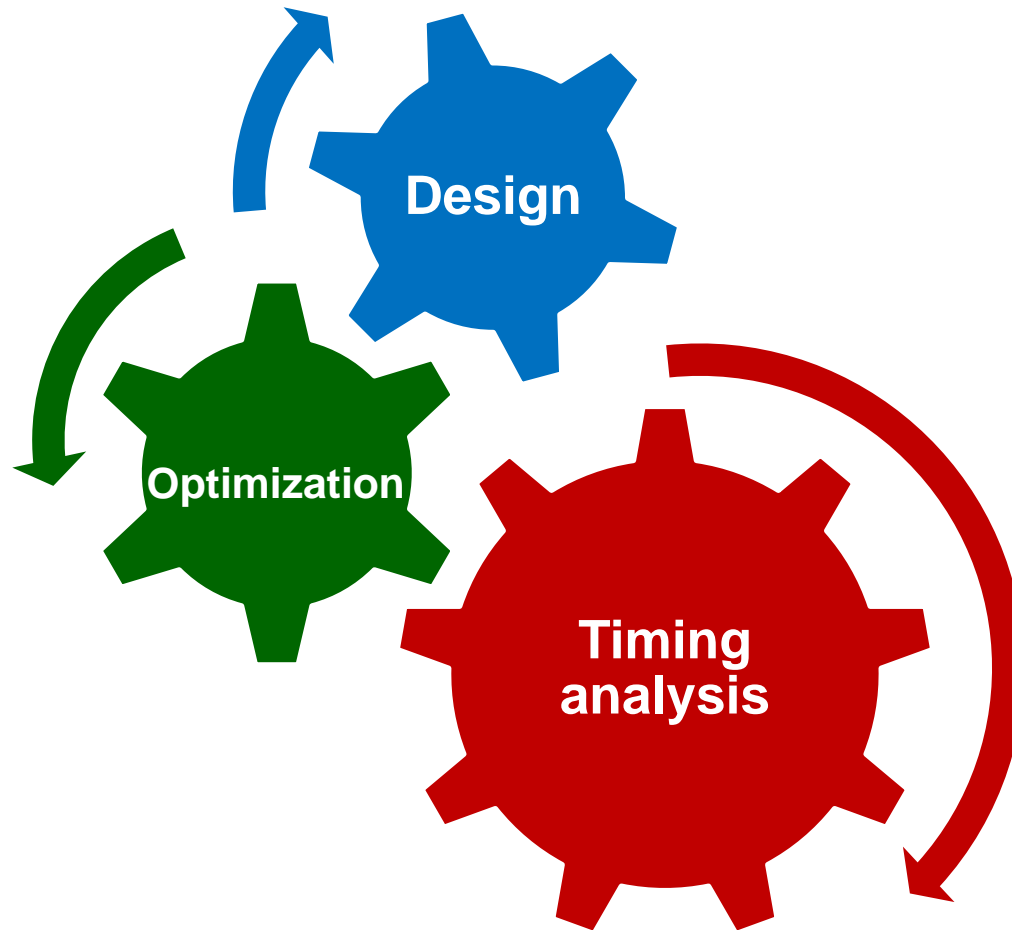
Graduate Institute of Electronics Engineering

National Taiwan University



Timing is Everything...

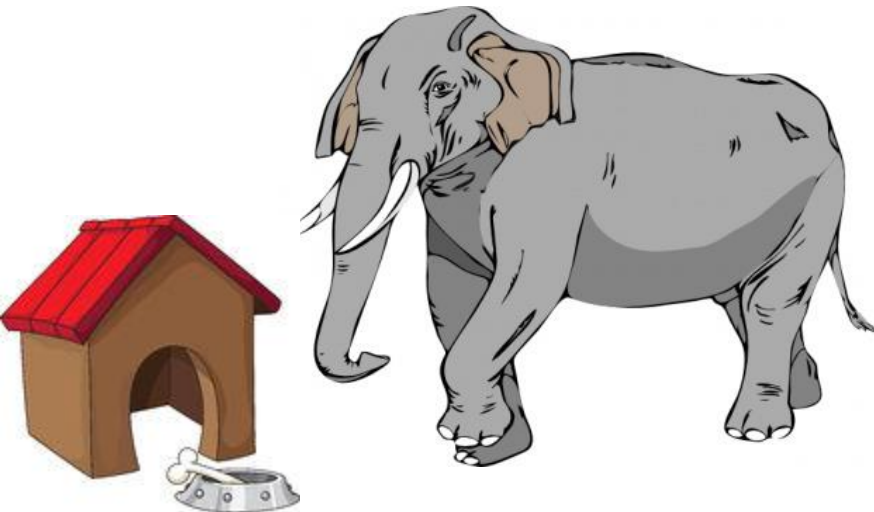
- Signals should arrive at the right place at the right time
- Timing analysis is essential in the modern IC design flow



Don't Put Timer into Other Engines!

Placement

Timing



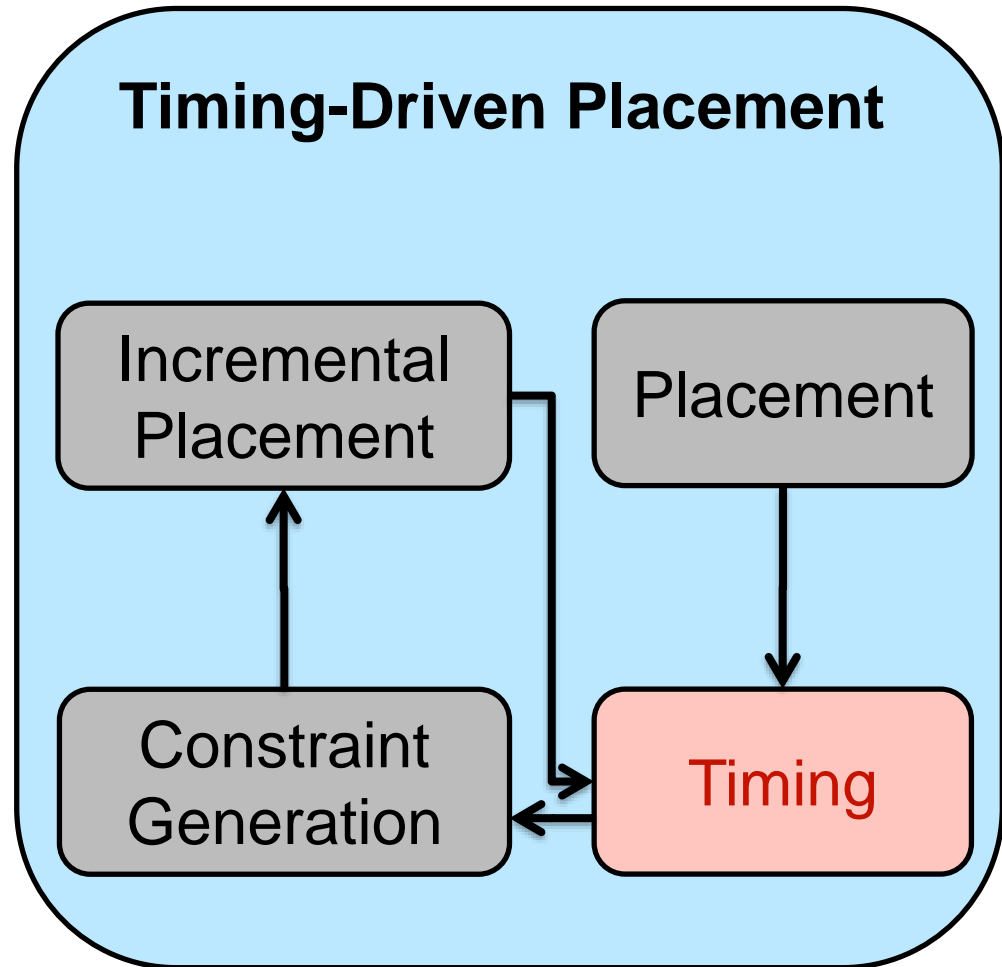
Timing-Driven Placement

Incremental Placement

Placement

Constraint Generation

Timing



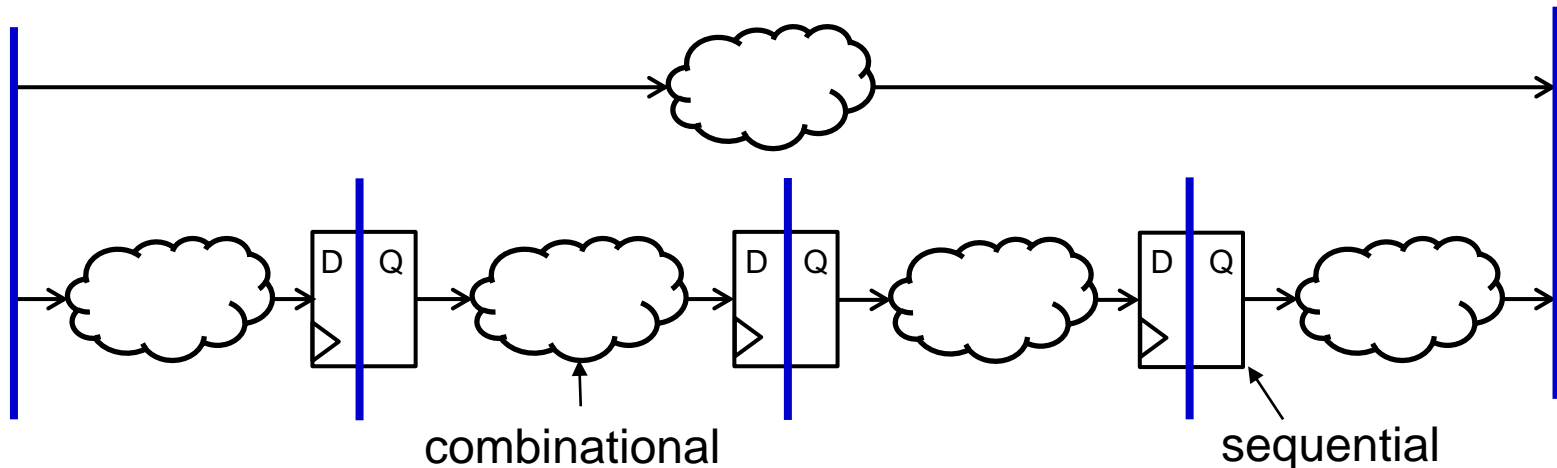
Timing Analysis

- Categories
 - **Functional timing analysis (FTA)**: concurrently consider input vectors and path sensitization to verify timing performance
 - Accurate but slow to achieve 100% coverage
 - **Static timing analysis (STA)**: separate timing from functionality, consider the **worst-case** performance, calculate delay based on pure structural analysis
 - Fast but pessimistic
 - Need false path information
- Because of its high scalability for large scale designs, **STA** is widely adopted in the modern IC design flow (industry standard)

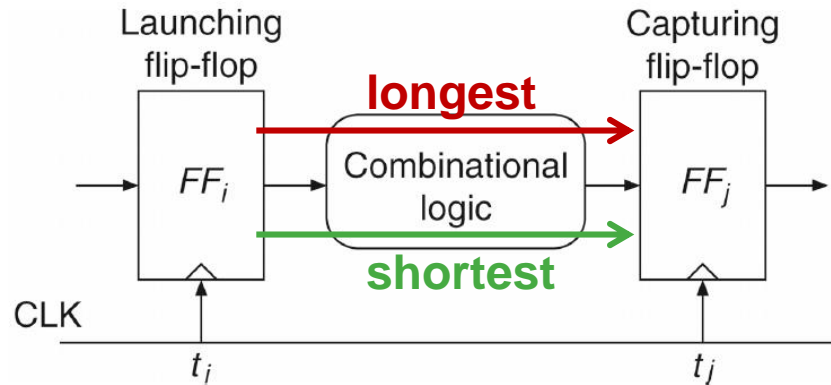


STA SOP

1. Divide a design into timing windows based on flip-flops/latches and I/Os
2. Construct a timing graph (directed acyclic graph) for each window
3. Calculate the circuit delay
4. Check timing constraints

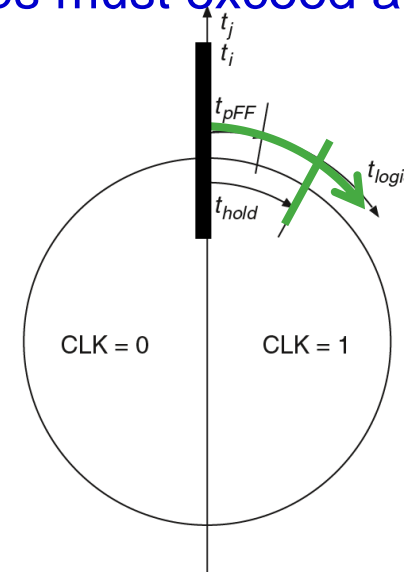
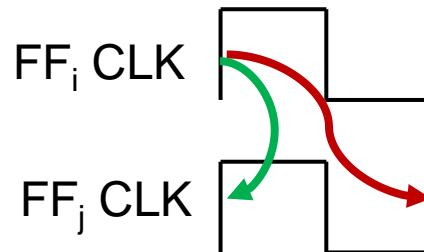
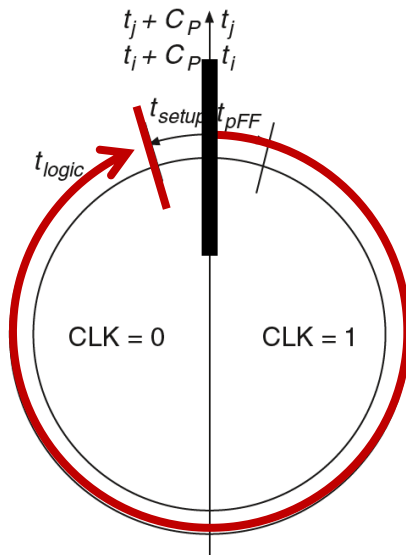


Timing Check



– **Setup time:** Delay between 2 flops should be less than 1 cycle

– **Hold time:** Delay between 2 flops must exceed a threshold



How to Perform STA?

- Path-based STA (PBA)
 - Test one path at one time
 - Provide detailed critical path information
 - Number of paths grows exponentially with circuit size
- Block-based STA (a.k.a. graph-based STA, GBA)
 - Propagate only the worst-case timing information
 - Most efficient, fastest and lowest memory
 - Hard to retrieve other timing-critical paths



The Simplest Timing Analysis Engine

Setup time

- Block-based: Topological ordering + longest/shortest path
 - Cycles in combinational are not allowed

- Timing graph:

- Vertex: gate/IO
- Edge: wire

- Arrival time

- Forward

- Required time

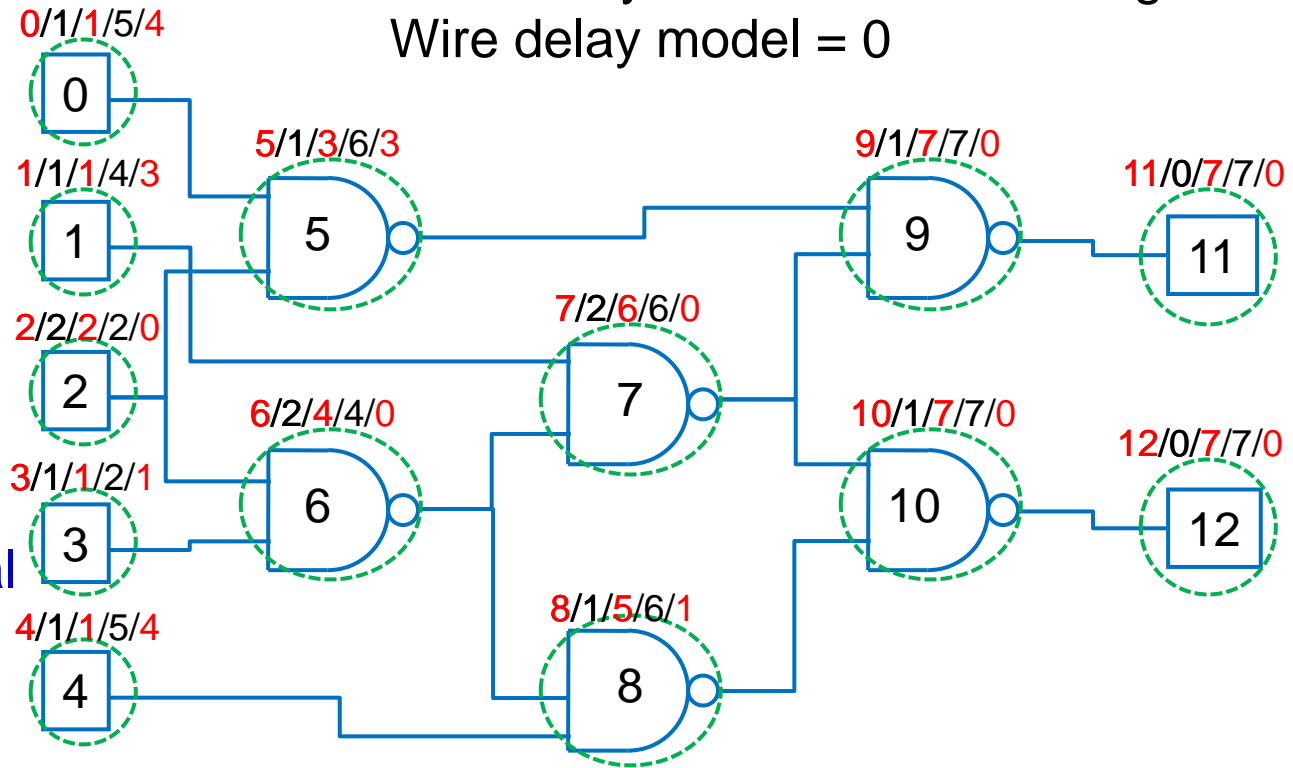
- Backward

- Slack

- Required-arrival

Gate delay model = # of fanout gates

Wire delay model = 0



Order/delay/arrival/required/slack



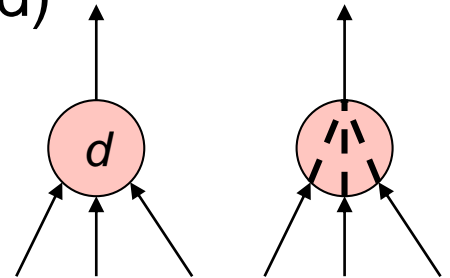
Other STA Ingredients

- Delay models
 - Gate/cell delay
 - Wire/interconnect
 - Environment constraints
 - Operating conditions
 - Wire load model
 - Design rule constraints
 - Timing constraints
 - Input
 - Output
 - Clock waveform
 - Timing exceptions
 - Multicycle paths
 - False paths
- .lib**
Liberty
(cell timing library)
- .sdc**
Synopsys Design Constraint
(timing constraints)
-



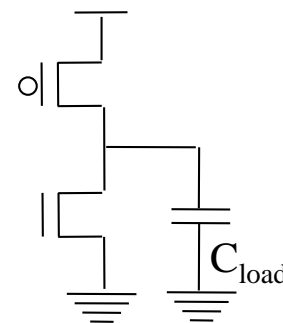
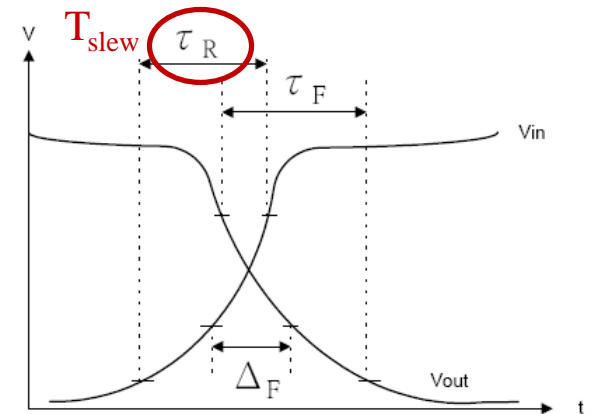
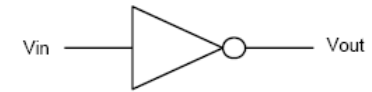
Gate Delay Models

- Constant delay model (popular for front-end)
 - Constant gate delay, or pin-to-pin gate delay
 - Not accurate
- Fanout delay model (popular for front-end)
 - Gate delay considering fanout load (#fanouts)
 - Slightly more accurate
- Linear delay model (abstract model used in academia)
 - More accurate than fanout delay model
- Library delay model (industry)
 - Tabular delay data given in the cell library (SPICE)
 - Pin-to-pin
 - Determine delay from input slew and output load
 - Table look-up + interpolation/extrapolation
 - Accurate

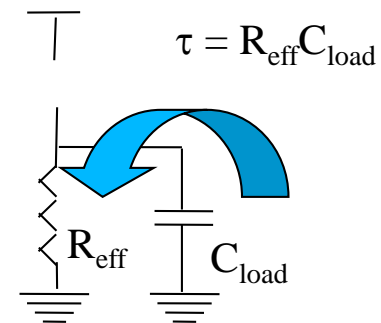


Gate Delay

- The delay of a gate depends on:
 - 1. **Input slew**
 - Slew = transition time
 - Slower transistor switching \Rightarrow longer delay and longer output slew
 - 2. **Output load**
 - Capacitive loading \propto charge needed to swing the output voltage
 - Due to interconnect and logic fanout
- Propagation delay: 50%-50%



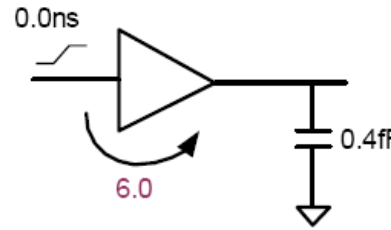
An inverter



e.g. output 1 \rightarrow 0



Timing Library



Input Transition (ns)	Total Output Load (fF)			
	0.2	0.3	0.4	0.5
0	3	4.5	6	7
0.1	5	8	10.7	13

Cell Delay (ps)

- Timing library contains all relevant information about each standard cell
 - E.g., pin direction, clock, pin capacitance, etc.
- **Delay** (fastest, slowest, and often typical) and **output slew** are encoded for each input-to-output path and each pair of transition directions (rise/fall)
- Values are typically represented as 2D look-up tables (of input slew and output load) based on SPICE simulation
 - (Linear) interpolation/extrapolation is used



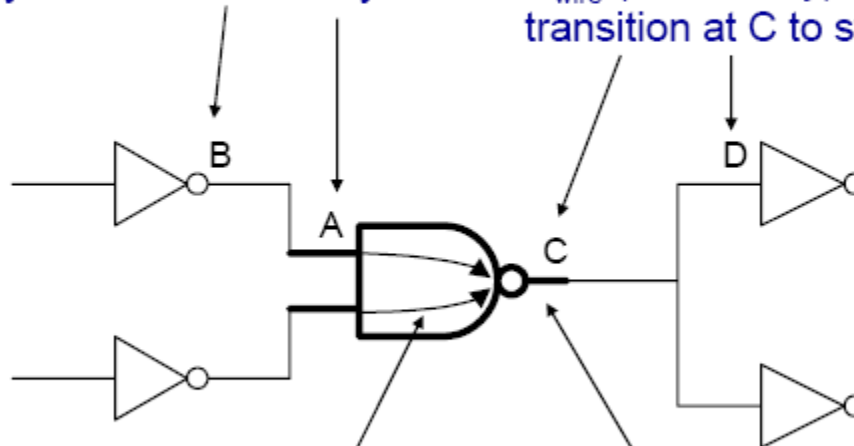
Linear Delay Model (1/2)

Gate delay Wire delay

$$\text{Delay} = D_{\text{slope}} + D_{\text{intrinsic}} + D_{\text{transition}} + D_{\text{wire}}$$

D_{slope} (Slope delay): delay at input A caused by the transition delay at B

D_{wire} (Wire delay): time from state transition at C to state transition at D



$D_{\text{intrinsic}}$ (Intrinsic delay): incurred from cell input to cell output

$D_{\text{transition}}$ (Transition delay): output pin loading, output pin drive



Linear Delay Model (2/2)

- Delay = $D_{\text{slope}} + D_{\text{intrinsic}} + D_{\text{transition}} + D_{\text{wire}}$
- Slope delay
 - The delay due to a slow logic transition at the input pin
 - $D_{\text{slope}} = D_{\text{Tprevious}} * S$ ($D_{\text{Tprevious}}$: previous-stage transition; S: slope sensitivity)
- Intrinsic delay
 - The built-in delay, fixed
- Transition delay
 - Output resistance times load
 - $D_{\text{transition}} = R_{\text{drive}} * (C_{\text{pin}} + C_{\text{wire}})$
- Wire delay
 - The time to propagate a logic transition through an interconnect network
 - $D_{\text{wire}} = R_{\text{wire}} * (C_{\text{pin}} + C_{\text{wire}})$

Downstream capacitance depends on interconnect model



Wire/Interconnect Delay

- Wire load model (front-end)
 - Unit fanout delay model
 - Incorporate an additional delay for each fanout
 - RC model (best-case, worst-case, balanced-case)
 - Calculate delay according to physical information (distance, loading, etc.)
- Wire model (back-end)
 - RC network extracted from routing
 - Global routing: Steiner tree
 - Detailed routing: RC extraction
 - SPEF (Standard Parasitic Exchange Format, .spef) (Industry!)



Wire/Interconnect Delay

- Wire load model (front-end)
 - Unit fanout delay model
 - Incorporate an additional delay for each fanout
 - RC model (best-case, worst-case, balanced-case)
 - Calculate delay according to physical information (distance, loading, etc.)
- Wire model (back-end)
 - RC network extracted from routing
 - Global routing: Steiner tree
 - Detailed routing: RC extraction
 - SPEF (Standard Parasitic Exchange Format, .spef) (Industry!)

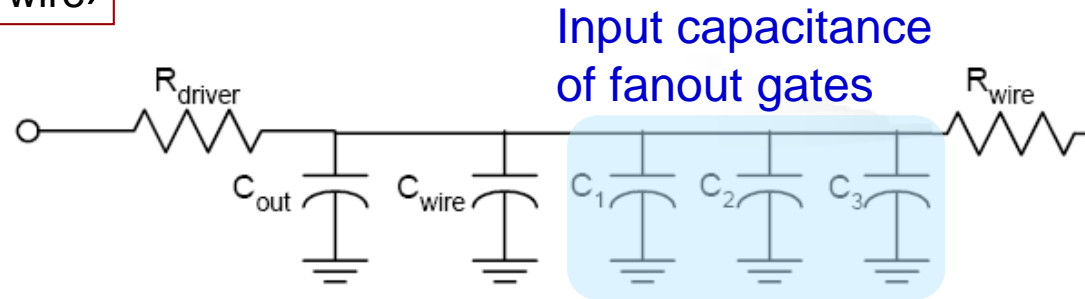


Front-End RC Models

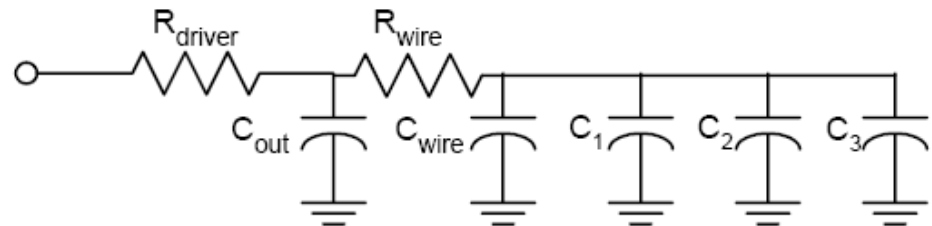
Downstream capacitance depends on RC network

- $D_{\text{wire}} = R_{\text{wire}} * (C_{\text{pin}} + C_{\text{wire}})$

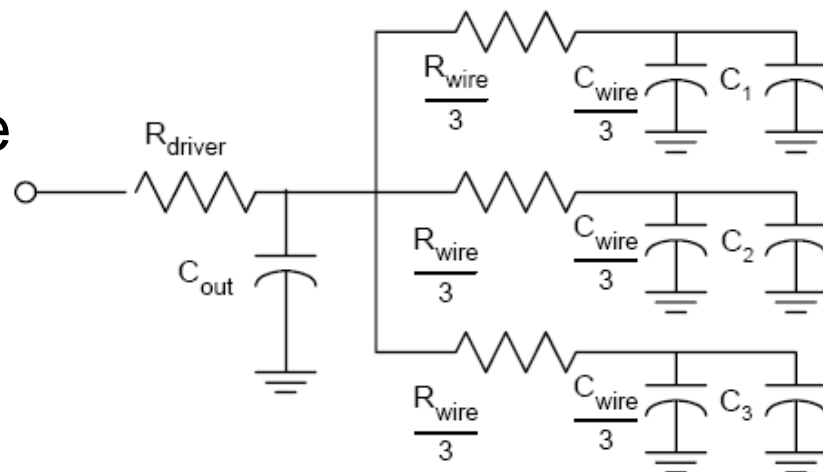
- Best-case RC tree
 - Wire delay = 0



- Worst-case RC tree



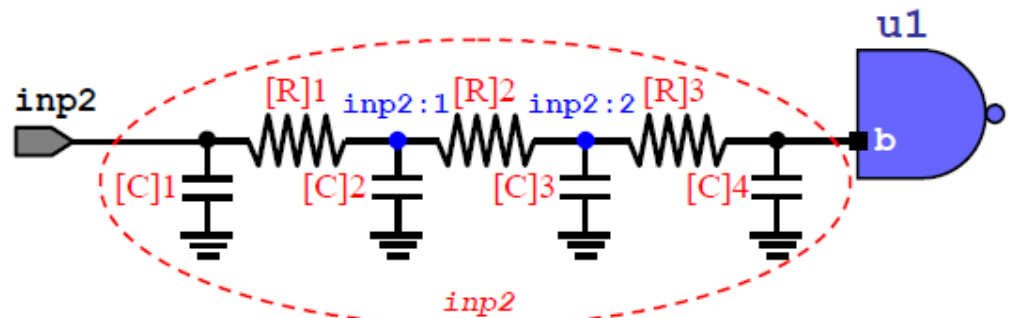
- Balanced-case RC tree



Back-End Wire Model

- Extract an RC network from routing
- SPEF (Standard Parasitic Exchange Format) (Industry!)

```
01. *D_NET inp2 2.0
02. *CONN
03. *P inp2 I
04. *I u1:b I
05. *CAP
06. 1 inp2 0.2
07. 2 inp2:1 0.5
08. 3 inp2:2 0.4
09. 4 u1:b 0.9
10. *RES
11. 1 inp2 inp2:1 1.4
12. 2 inp2:1 inp2:2 1.5
13. 3 inp2:2 u1:b 1.6
14. *END
```



Wire Delay Calculation

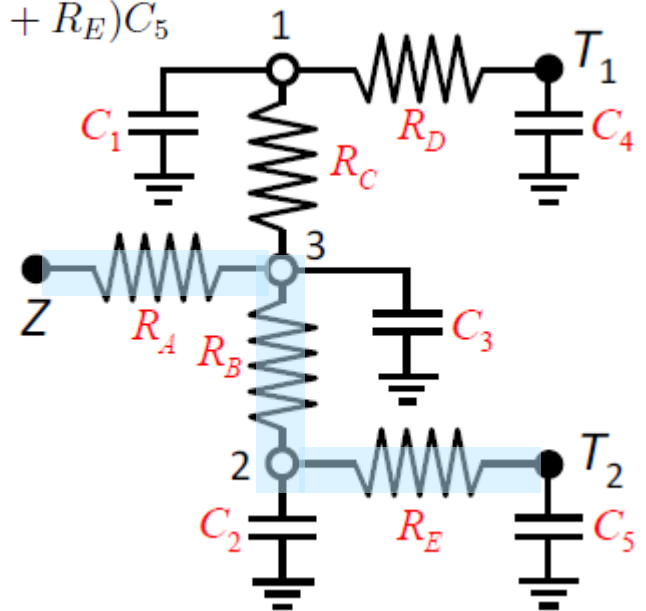
- Elmore delay model
- Delay

$$\begin{aligned}d_{T_2} &= R_A C_1 + R_A C_3 + R_A C_4 + (R_A + R_B) C_2 + (R_A + R_B + R_E) C_5 \\ &= R_A (C_1 + C_3 + C_4) + (R_A + R_B) C_2 + (R_A + R_B + R_E) C_5\end{aligned}$$

- Slew

$$s_{oT} \approx \sqrt{s_i^2 + 2\beta_T - d_T^2}$$

- s_i : input slew
- β_T : second moment of the input response
- d_T : Elmore delay



W.C. Elmore, "The Transient Analysis of Damped Linear Networks with Particular Regard to Wideband Amplifiers," J. Applied Physics, vol. 19(1), 1948.

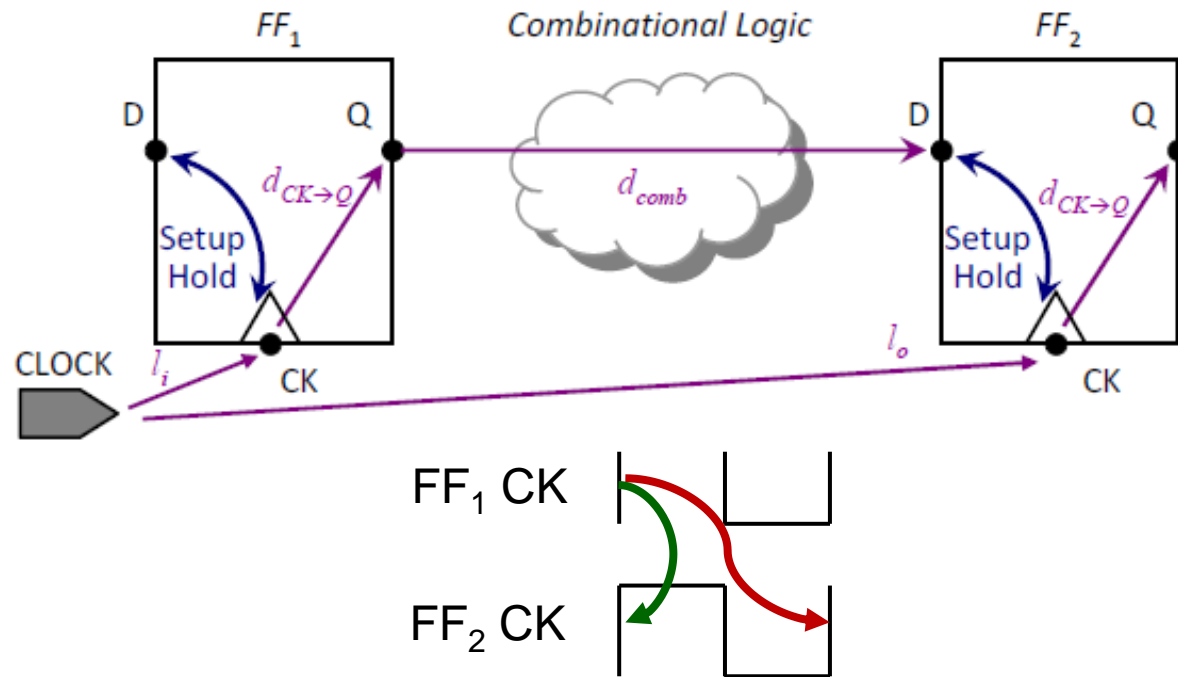
C. V. Kashyap, C. J. Alpert, F. Liu and A. Devgan, "Closed-form Expressions for Extending Step Delay and Slew Metrics to Ramp Inputs for RC Trees", IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems, 23(4)(2004), pp. 509-516.

Timing Propagation (Setup Time)

- **Arrival time** can be computed in the **topological order** from inputs to outputs
 - Arrival time of a primary input is given
 - When a node is visited, its output arrival time is:
the max of its fanin arrival times + its own gate delay
- **Required time** can be computed in the **reverse topological order** from outputs to inputs
 - Required time of a primary output is given or specified due to clock cycle constraint
 - When a node is visited, its input required time is:
the min of its fanout required times – its own gate delay
- **Slack** = required time – arrival time
 - Timing flexibility margin (positive: good; negative: bad)



Setup and Hold Constraints



Setup

$$at_D^{late} = l_i^{late} + d_{CK \rightarrow Q} + d_{comb}^{late}$$

$$rat_{setup} = rat_D^{late} = \textcircled{T} + l_o^{early} - t_{setup}$$

$$slack^{late} = rat^{late} - at^{late}$$

Hold

$$at_D^{early} = l_i^{early} + d_{CK \rightarrow Q} + d_{comb}^{early}$$

$$rat_{hold} = rat_D^{early} = l_o^{late} + t_{hold}$$

$$slack^{early} = at^{early} - rat^{early}$$



Environment Constraints

- Operating conditions
- Wire load model
- Design rule constraints



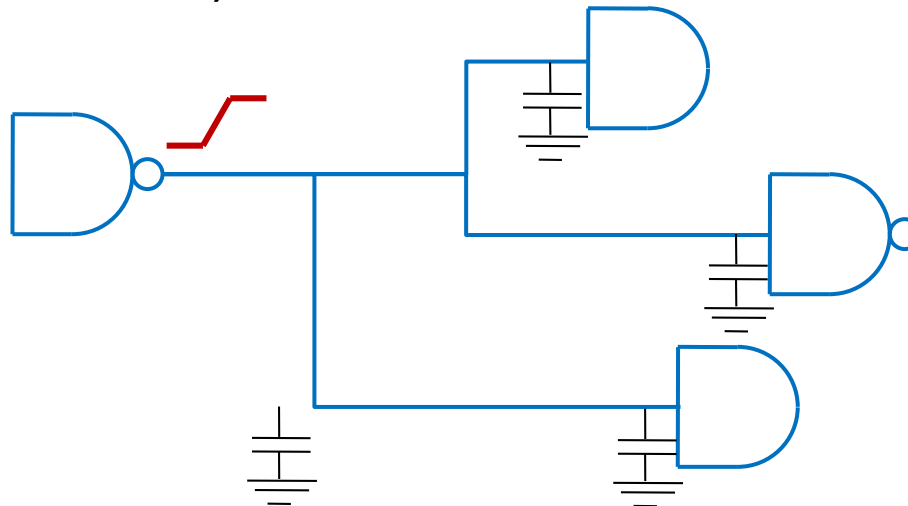
Operating Conditions

- Cases: best, typical, worst
- Process variation
 - Treated as a straight percentage variation in any performance calculation
- Voltage variation
 - Speed increased with higher voltage
- Temperature variation
 - Delay increased with higher temperature
- Implemented by setting different **derating** values



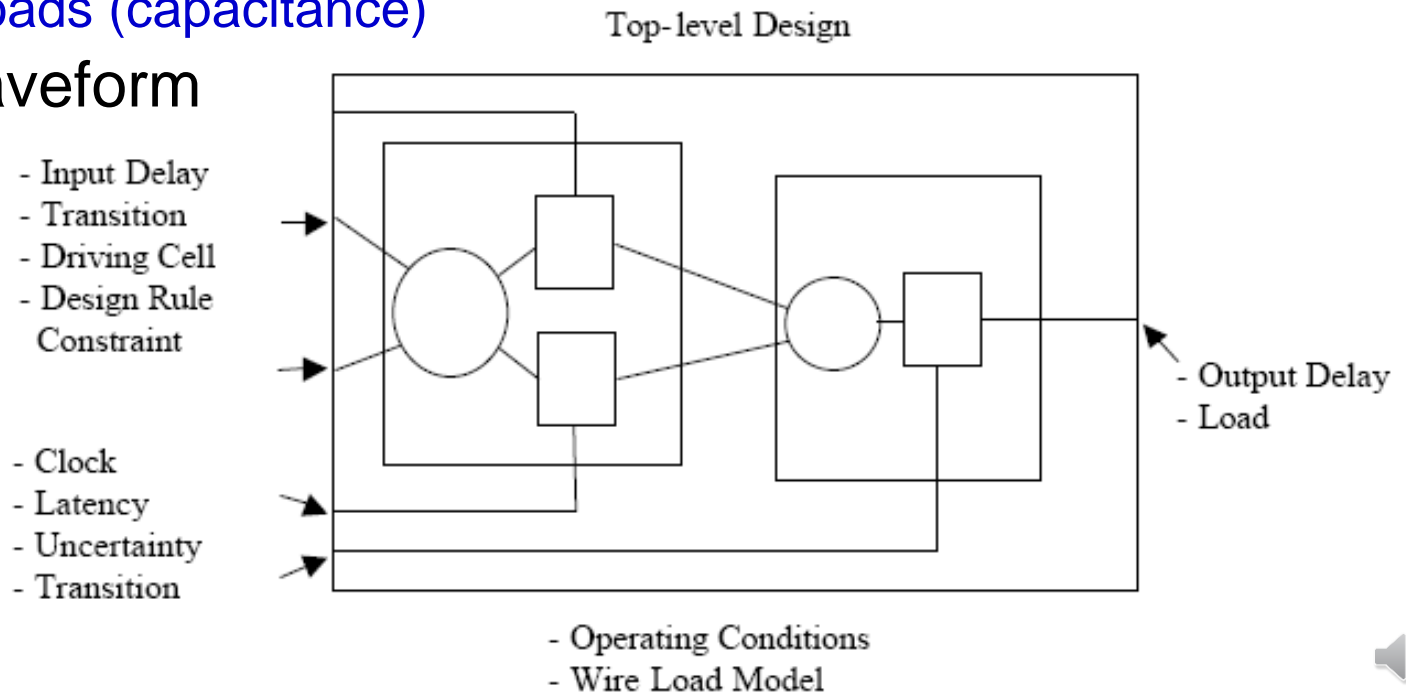
Design Rule Constraints

- \neq design rules for layout editing or mask generation
 - Width, spacing, enclosure
- Max transition time: max transition time of an output pin
- Max capacitance: max capacitance of an output pin
- Max fanout: max number of fanout allowed for an output pin (soft constraint)



Timing Constraints

- Inputs:
 - Arrival times (input delays)
 - Input drives (resistance)
- Output
 - Required times (output delays)
 - Output loads (capacitance)
- Clock waveform



Timing Exceptions

- Override default single-cycle timing behavior
 - Multicycle path
 - Override the default setup & hold relations
 - False path
 - Prune timing paths which are never be exercised functionally





Welcome to the real world!



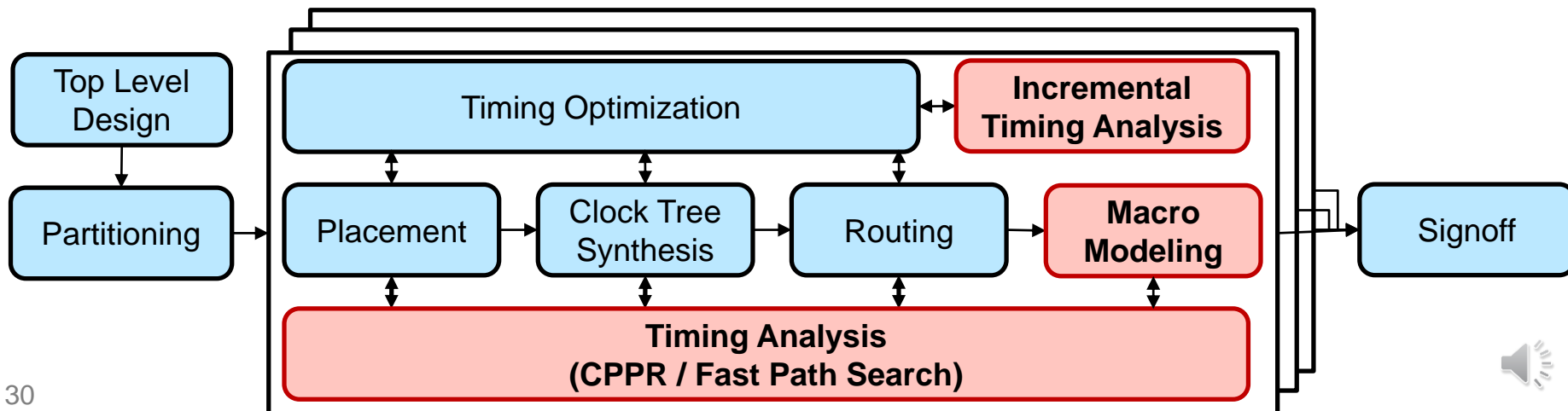
Challenges

- On chip variations: **P**rocess, **V**oltage, **T**emperature
 - Remove artificial pessimism
- Wire delay/slew calculation
- New cell delay model
- All critical path information is important!
 - K-most vs. the most critical path
- Repeated timing optimization operations
- Hierarchical design for large-scale integration
 - Hierarchical STA
- Key concerns: Accuracy, runtime/memory and scalability
 - Golden: SPICE



Key Techniques for Modern Timers

- On chip variations: **P**rocess, **V**oltage, **T**emperature
 - Common path pessimism removal (CPPR)
- All critical path information is important!
 - Fast path search
- Repeated timing optimization operations
 - Incremental timing analysis
- Hierarchical design paradigm for large-scale integration
 - Compact and accurate timing macro modeling for hierarchical STA



What We Have Covered Today...

- We have reviewed STA ingredients
- Recent research endeavors
 - Common path pessimism removal
 - Wire timing prediction
 - Fast path search
 - Fast incremental timing analysis
 - Compact and accurate timing macro modeling
 - Current source delay model
- Future directions:
 - Multi-corner multi-mode
 - Signal integrity
 - Voltage awareness
 - Statistical timing analysis



Thank you!

