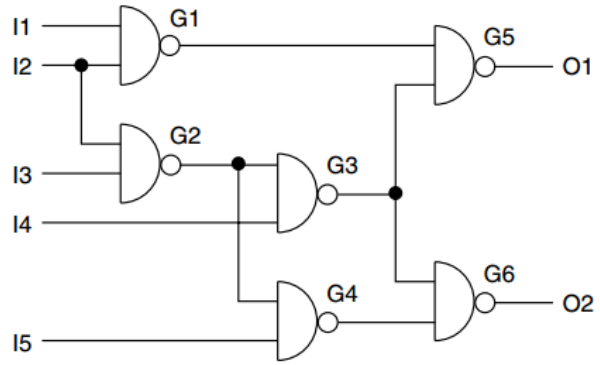
Static timing analysis

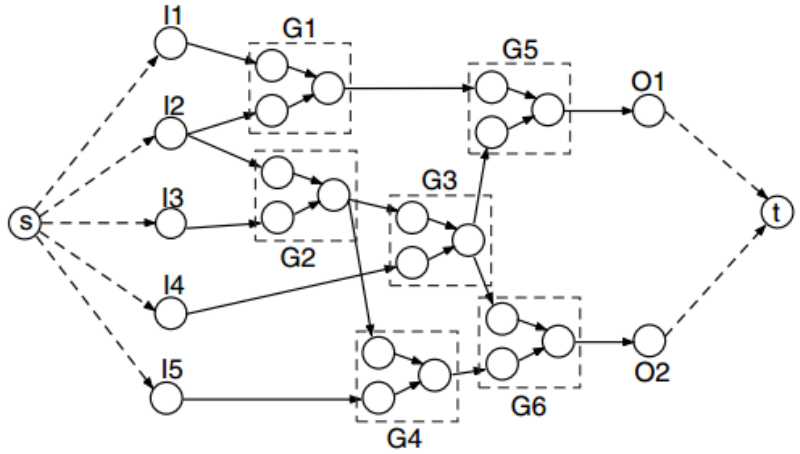
VLSI CAD

Compiled by Oleg Venger

Representation of combinational circuits

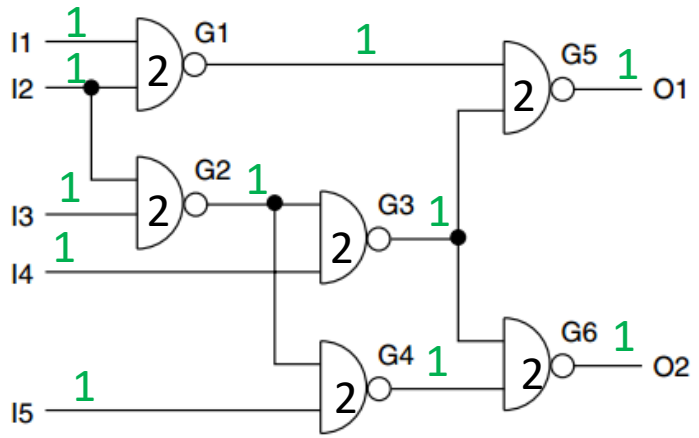
- Timing graph: $G(V, E)$ where
- V includes all terminals of cells
- E has two types of elements:
  - Timing arcs:
    - Combinational cell:
      - from each input to output – propagation delay
    - Sequential cell:
      - From clock input to output(s) – propagation delay
      - From clock input to data input – setup, hold checks
  - Interconnect delays: connect cell output to the inputs of its fanout gates





Background

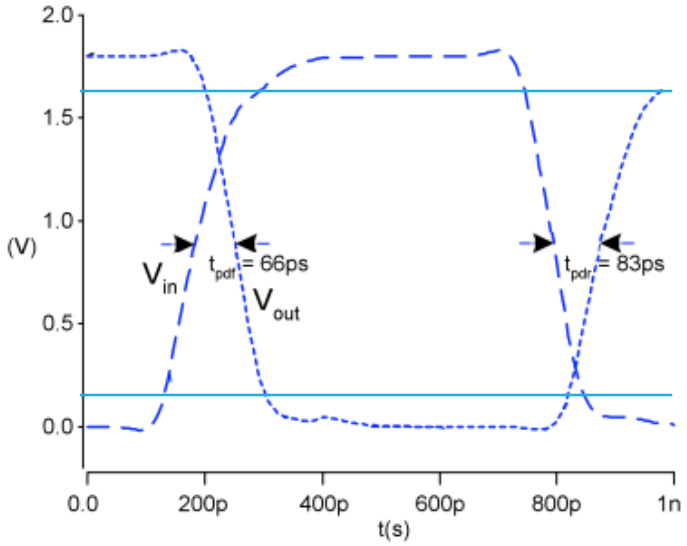
- Netlist is represented as Directed Acyclic Graph (DAG) – timing graph
- Delay values associated with edges of timing graph
- Timing information is associated with topological paths:
  - Start point
  - End point
  - Path delay
- Total path delay is the sum of stage delays (cell and net delays) along the path
- Maximum path delay through a combinational block defines the speed of the circuit



What is max path delay for this circuit?

Delay parameters

- t_r: rise transition time
  - From output crossing 0.1 VDD to 0.9 VDD
- t_f: fall transition time
  - From output crossing 0.9 VDD to 0.1 VDD
- t_{pdr}: rising propagation delay
  - From input crossing VDD/2 to rising output crossing VDD/2
- t_{pdf}: falling propagation delay
  - From input crossing VDD/2 to falling output crossing VDD/2
- Can t_{pd} be negative?



Inverter timing response

Gate delay models

- Standard cell delays are pre-characterized for various capacitive loads C_L and input transition time τ_{in}
- Some commonly used cell delay models:
  - Look-up tables with each entry in the table corresponding to the delay under different capacitive loads and different input transition times
  - $D = k_1 + C_L k_2$. Where k_1 - intrinsic delay. k_2 - drive strength
  - $D = \alpha\tau_{in} + \beta C_L + \gamma\tau_{in} C_L + \delta$
  - Etc
- After calculation, delay is scaled by the operating conditions:
  - $FinalDelay = CalculatedDelay * K_{voltage} * K_{temperature} * K_{process}$

Interconnect delay: wire load model

- Net delay results from the need for charging/discharging all the parasitics of a given net. It is a function of net capacitance and net resistance
- Wire load model (pre-layout)
  - Net length is a function of net fanout and block area
  - For a given block area, averages of R and C are estimated for different fanouts
  - Net delay is calculated simply as $R * C$

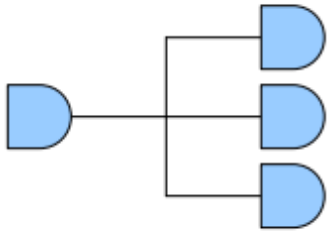
Capacitance as a function of fanout :

1	0.015
2	0.030
3	0.046

Resistance as a function of fanout :

1	0.012
2	0.016
3	0.020

For fanout = 3
Net delay = 0.046 * 0.020

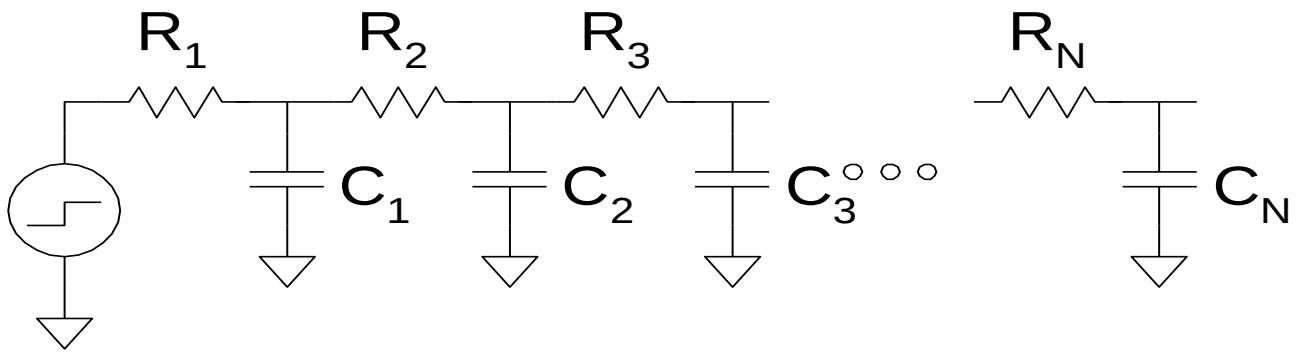


Interconnect delay: Elmore model

- Pullup or pulldown network modeled as *RC ladder*
- Elmore delay of RC ladder:
  - $\tau_{DN} = \sum_{k=1}^{N} C_k \left(\sum_{j=1}^{k} R_k\right) = \sum_{k=1}^{N} CkR = RC\frac{N(N+1)}{2} = rcL^2\frac{N+1}{2N} \xrightarrow[N\to\infty]{} \frac{rcL^2}{2}$

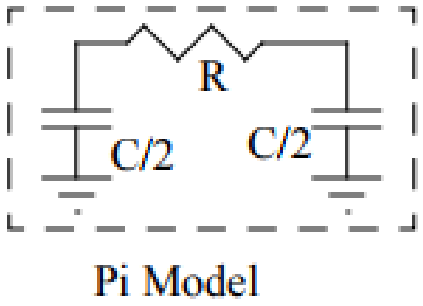
$$R = \frac{r * L}{N}$$

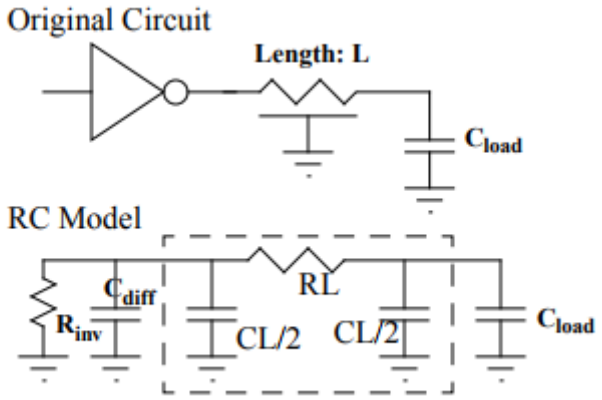
$$C = \frac{c * L}{N}$$



Multiple segment nets

- The distributed RC delay can be modeled by breaking up a wire into one or more segments and using a lumped π-model for each segment.

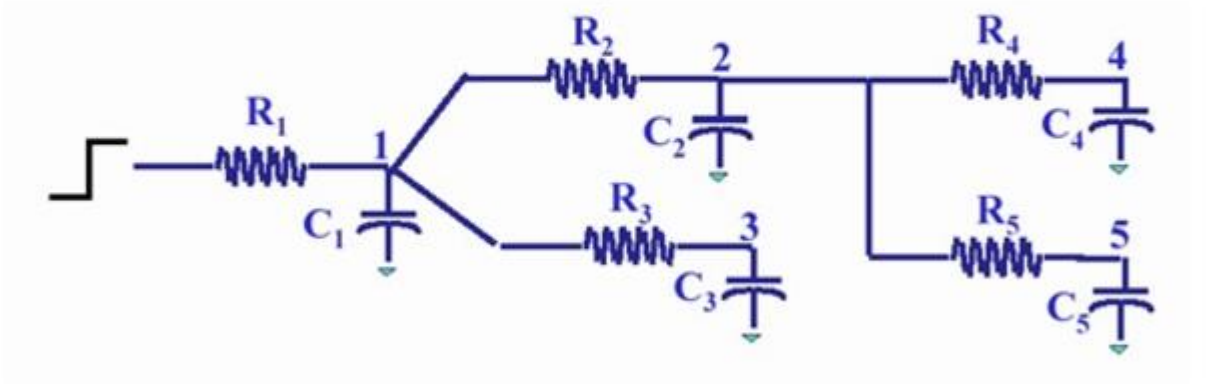




Elmore delay: multiple segment nets

- **Resistance-oriented Formula:**

$$T_{delay} = \sum_{i \text{ on path}} R_i C_{downstream,i}$$



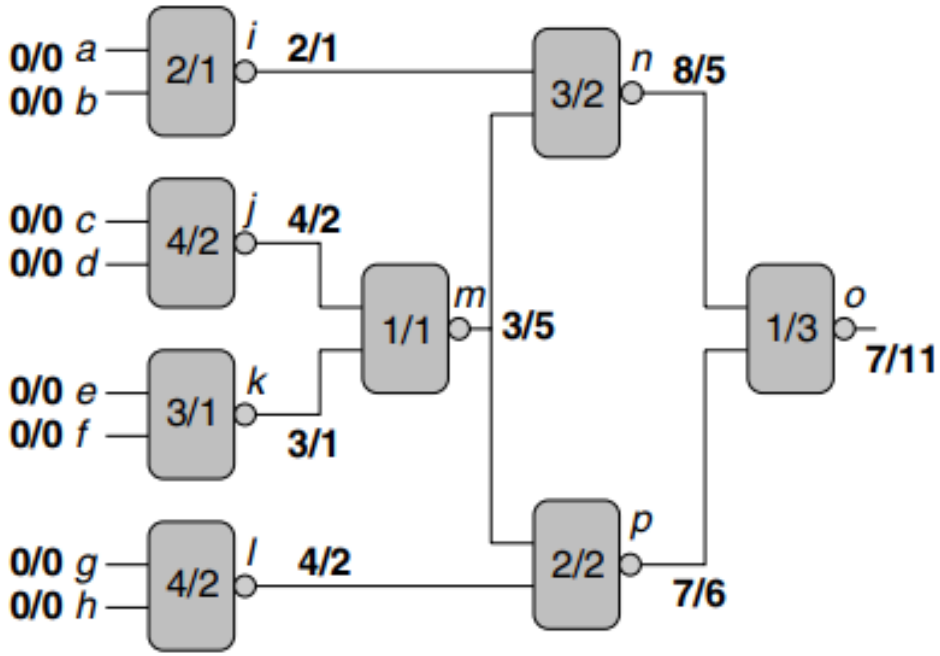
$$T_{delay,4}=R_1(C_1+C_2+C_3+C_4+C_5)+R_2(C_2+C_4+C_5)+R_4C_4$$

Critical path and slack

- **Arrival time** of a signal is the time elapsed for a signal to arrive at a certain point.
- **Required arrival time** is the latest time at which a signal can arrive without making the clock cycle longer than desired.
- $slack(v) = Req(v) - Arr(v)$
- **Critical path**, defined as the path between an input and an output with the minimal slack
- Slack is constant along the critical path

Arrival time propagation

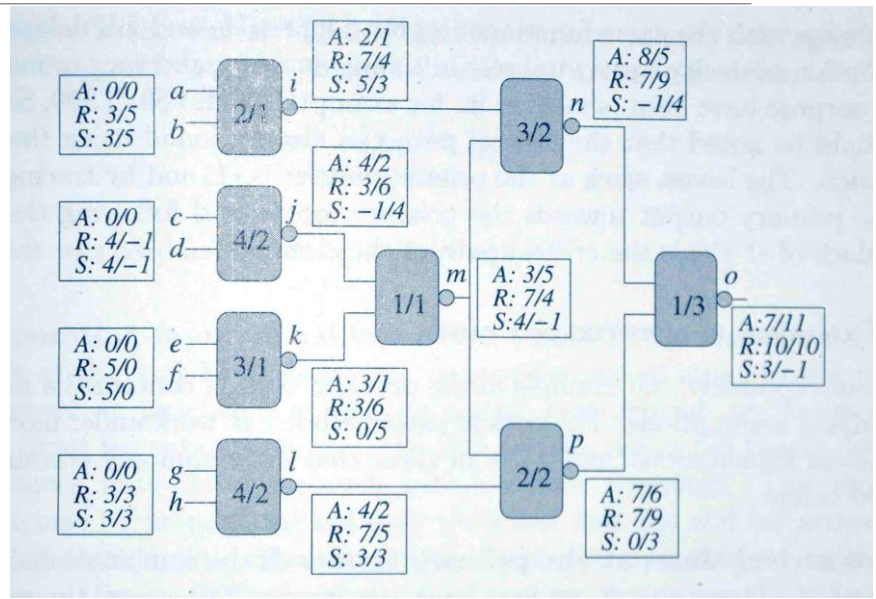
1. **for** each vertex v **:**
   - $v.num\_ready\_inputs$ = 0
   - if v is primary input **:** add v into $To\_Compute$ vertex queue
2. **while** $To\_Compute$ queue is not empty **:**
   - Get v from $To\_Compute$ queue
   - **if** v is primary input : $Arrmax(v) = 0$
   - **else** :
     - **for** each input edge (u, v) **:**
       - $Arrmax(v) = \max(Arrmax(v), Arrmax(u) + d(u, v))$
     - **for** each output edge (v, w) **:**
       - $w.num\_ready\_inputs$ += 1
       - **if** *w.num_ready_inputs == w.num_inputs*:
         - add vertex w into $To\_Compute$ queue



Max delay propagation for rise/fall delay (circuit with inverting gates)

Required time propagation

1. **for** each vertex v :
   - *v.num_ready_outputs = 0*
   - if v is primary output : add v into $To\_Compute$ vertex queue
2. **while** $To\_Compute$ queue is not empty :
   - get v from $To\_Compute$ queue
   - **if** v is primary output : $Reqmax(v) = T_{clock} - T_{setup}$
   - **else** :
     - **for** each output edge (v, u) :
       - $Reqmax(v) = \min(Reqmax(v), Reqmax(u) - d(v, u))$
     - **for** each input edge (w, v) :
       - *v.num_ready_outputs += 1*
       - **if** *v.num_ready_outputs == v.num_outputs* :
         - add vertex w into $To\_Compute$ queue



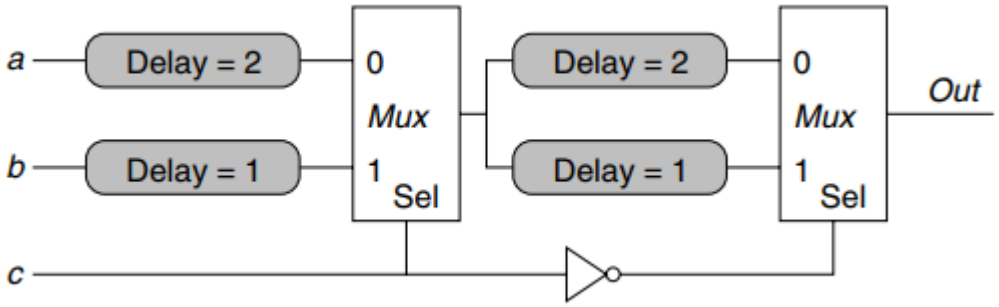
Required time and slack

Extensions of basic algorithm

- Non-zero arrival times at the primary inputs
  - For instance, in hierarchical models
- Minimum delay calculations
  - Propagate earliest arrival time instead of latest
- Minmax delay calculations
  - If the gate delay is specified in terms of an interval $[d_{min}, d_{max}]$
- Non-inverting gates
  - Rise[fall] transition at the output affected by a rise[fall] transition at the input
- Incorporating input signal transition times
  - Propagate arrival time/ transition time pairs
  - Which of two pairs to propagate if $at_1 > at_2$ and $tt_1 < tt_2$?

False paths

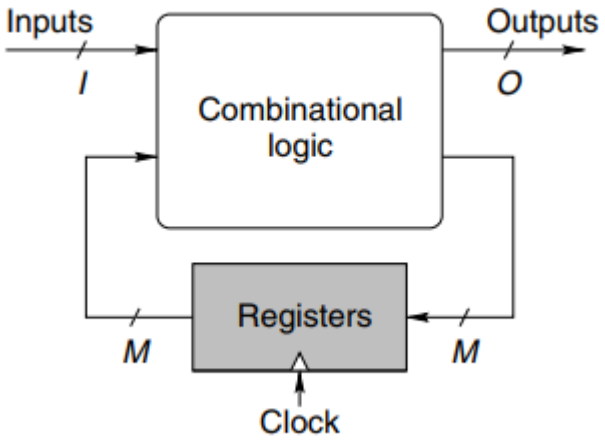
- Static analysis is fast but leads to **false paths**
- Path of length 4 can never be sensitized (events on this path are not propagated to the output)
  - In fact it can be dynamically sensitizable
- Approaches:
  - Throw out user-specified “non-sensitizable” (false) paths
  - Circuit delay = Length of longest path
    - Too pessimistic



- This leads to functional timing analysis for false paths

Sequential circuits

- For STA purpose it is possible to represent a sequential circuit as combinational logic with additional inputs and outputs
  - storage elements outputs are considered as primary inputs (PI) to combinational logic
  - storage element inputs are considered as primary outputs (PO) of combinational logic
- STA depends on type of registers
  - Level-clocked latches
  - Edge-triggered flip-flops

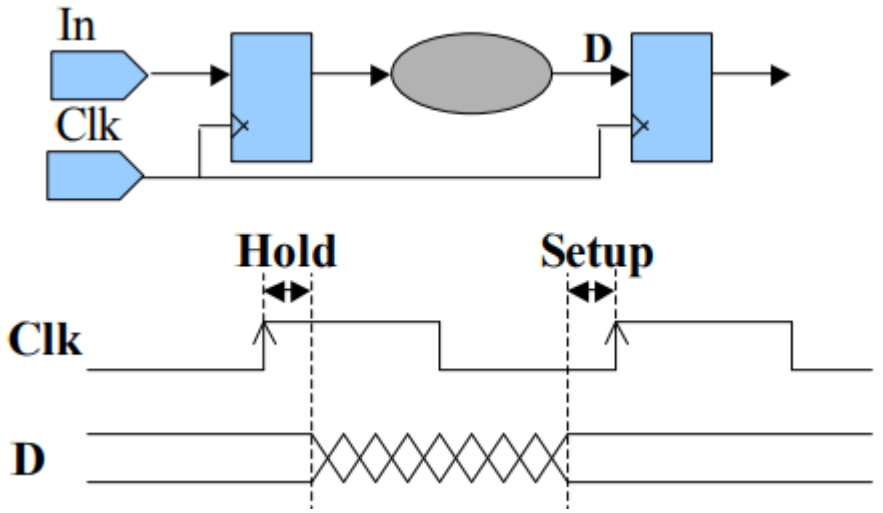


Clock skew

- Clock signals do not arrive at all the registers at the same time. This is referred to as ‘skew’
- Conventional approach to clock tree synthesis tries to minimize the skew
- Playing with the clock skew might be beneficial:
  - May change the required arrival time for late or early signals
  - Deliberate clock skew is used to reduce dynamic transient current that happens during register switching and could create a significant voltage drop on the power distribution network

Setup and hold times (flip-flop)

- Setup time: amount of time data must be stable at the data pin of flip-flop before the clock capturing edge
- Hold time: amount of time data must remain stable at the data pin of flip-flop after the clock capturing edge



STA Constraints

- Main path constraints include:
  - Setup time (Input-to-Reg and Reg-to-Reg paths)
  - Hold time (Input-to-Reg and Reg-to-Reg paths)
  - Input arrival time (Input-to-Reg and Input-to-Output paths)
  - Output required arrival time (Reg-to-Output and Input-to-Output paths)
- Electrical constraints: min and max limits for:
  - Net / Port capacitance
  - Net Transition times

Edge-triggered circuits

- The data input of a register must receive data at least T_{setup} units before the active edge of clock
- The data input of a register must be kept stable for a time of T_{hold} so that data is stored correctly in the flip-flop
- Each register has delay between the time the clock and the data are both available and when it is latched (clock-to-Q delay)
- For timing verification of edge-triggered circuits, it is possible to decompose the circuit into combinational blocks and verify the validity of constraints on each such block independently

Level-clocked circuits

- More complex analysis: combinational blocks are not insulated from each other and multicycle path are possible
- Offer more flexibility both in terms of minimum clock period and minimum amount of memory elements required

Time borrowing

- Level-clocked circuits allow time borrowing i.e. permitting the logic to automatically borrow time from next cycle, thereby reducing the time available for data to arrive for the following cycle

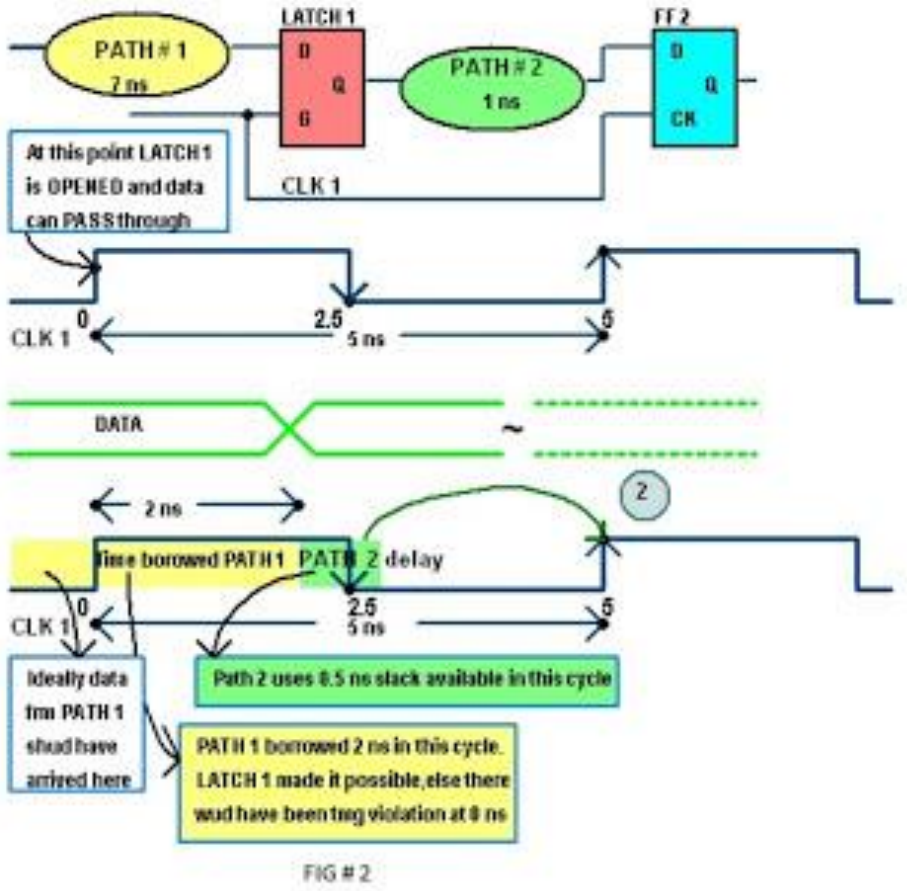


FIG # 2

Statistical STA (SSTA)

- Under true operating conditions, device parameters are perturbed from their nominal values due to various type of variations
- SSTA is an attempt to account for variation while doing timing analysis
- SSTA treats delays not as fixed numbers, but as probability density functions (PDFs), taking the statistical distribution of parametric variations into consideration while analyzing the circuit

Variations classification

- Process variations: result from perturbation in the fabrication process leading to deviations from the nominal values of parameters such as effective channel length, dopant concentration, or interlayer dielectric thickness
  - Inter-die: affect all the devices on the same chip in the same way
  - Intra-die: correspond to variability within single chip
    - Random variations: depict random behavior that can be characterized in terms of distribution. The distribution may either be explicit (built from fabrication line measurements), or implicit (e.g. Gaussian)
    - Systematic variations: show predictable variational trends across the chip, and are caused by known physical phenomena during manufacturing
- Environmental variations: due to changes in the operating environment such as temperature or voltage

Summary

- Static timing analysis applied at various phases of the design
- Linear complexity w.r.t. circuit size
- The closer to final layout the more accurate timing models used
- Accurate timing analysis is very important for circuit optimization

References

- Electronic Design Automation for Integrated Circuits Handbook. EDA for IC Implementation, Circuit Design, and Process Technology. Chapter 6 by Sachin S. Sapatnekar
- Zhou, Shuo, Static timing analysis in VLSI design
- Christiano Forzan. Introduction to Static Timing Analysis
- J.A.N. Rabaey, Digital Integrated Circuits
- N. Weste, D. Harris, CMOS VLSI Design