Lecture 14: Clock Synthesis

Matthew Guthaus
Professor
UCSC, Computer Science & Engineering
http://vlsida.soe.ucsc.edu
mrg@ucsc.edu

Today's Lecture

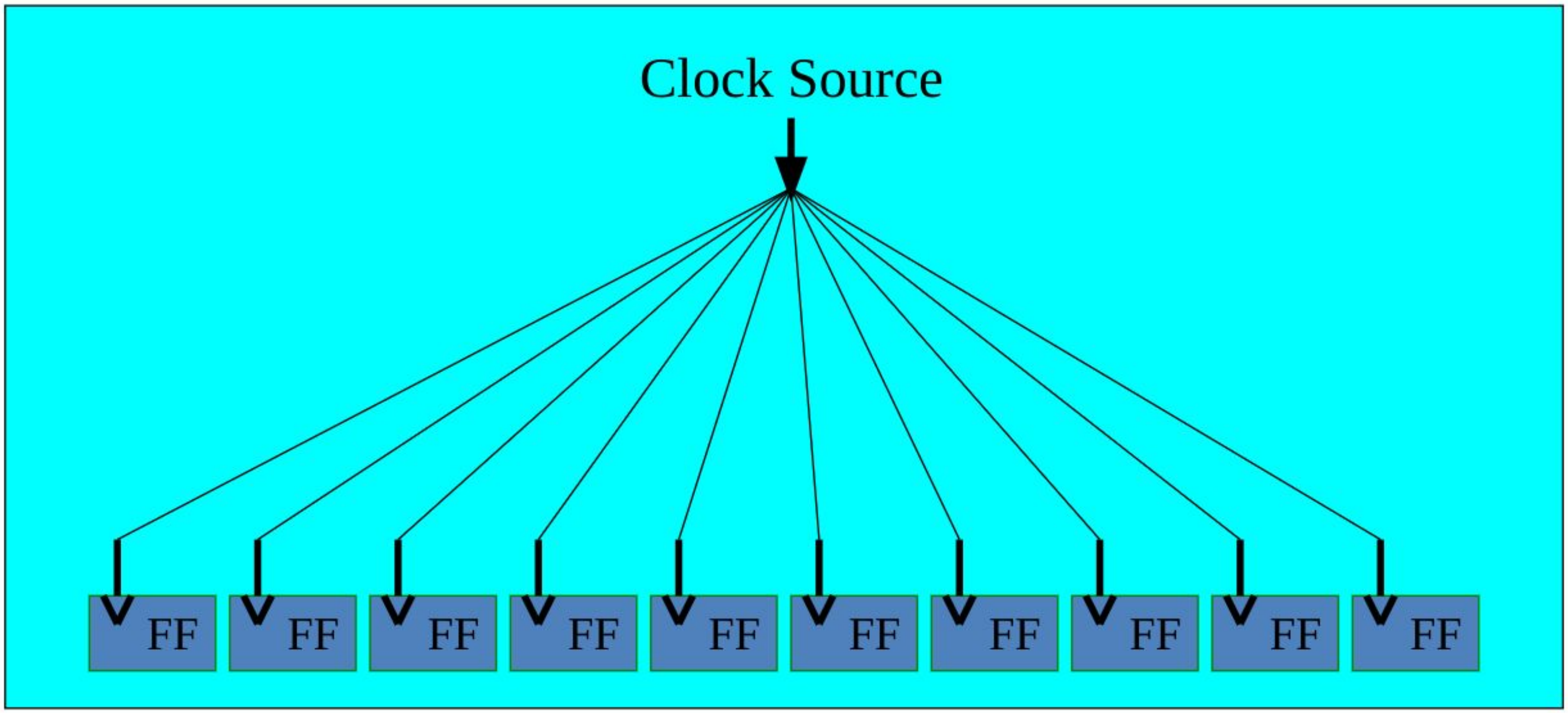
- Clock Metrics
- Local vs Global Clock Skew
- Common Path Pessimism Removal (CPPR)
- Types of Clocks
  - H-Tree
  - Balanced tree (DME)
  - Grid

Routing of Clock and Power Nets

- Different from other signal nets, clock and power are special routing problems
  - For clock nets, need to consider *clock skew* as well as delay.
  - For power nets, need to consider *current density* (IR drop)
  - => specialized routers for these nets.
- Automatic tools for ASICs
- Often manually routed and optimized for microprocessors, with help from automatic tools

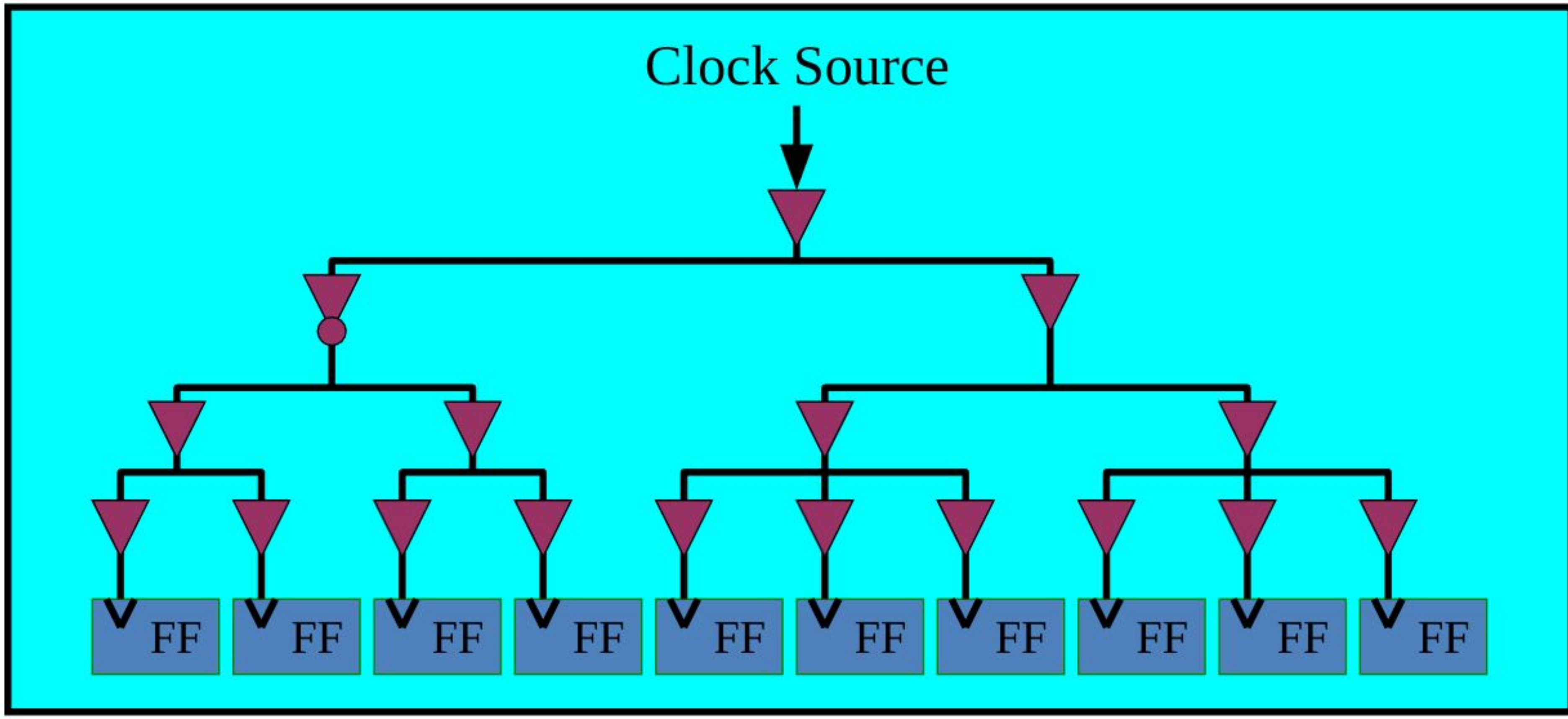
Clock trees

- A path from the clock source to clock sinks



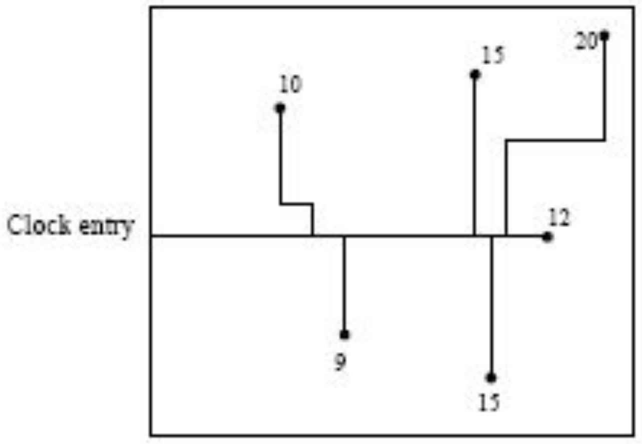
Clock trees

- A path from the clock source to clock sinks

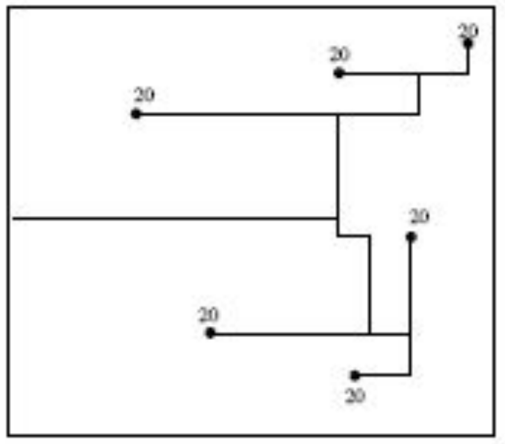


Global Clock Skew

- Clock skew is the maximum difference in the arrival time of a clock signal at any two different components.
- Easy to compute and you can ignore STA
  - Find minimum and maximum delay to clock sinks, subtract



Clock skew = 20 - 9 = 11 units



Clock skew = 0

Local Clock Skew

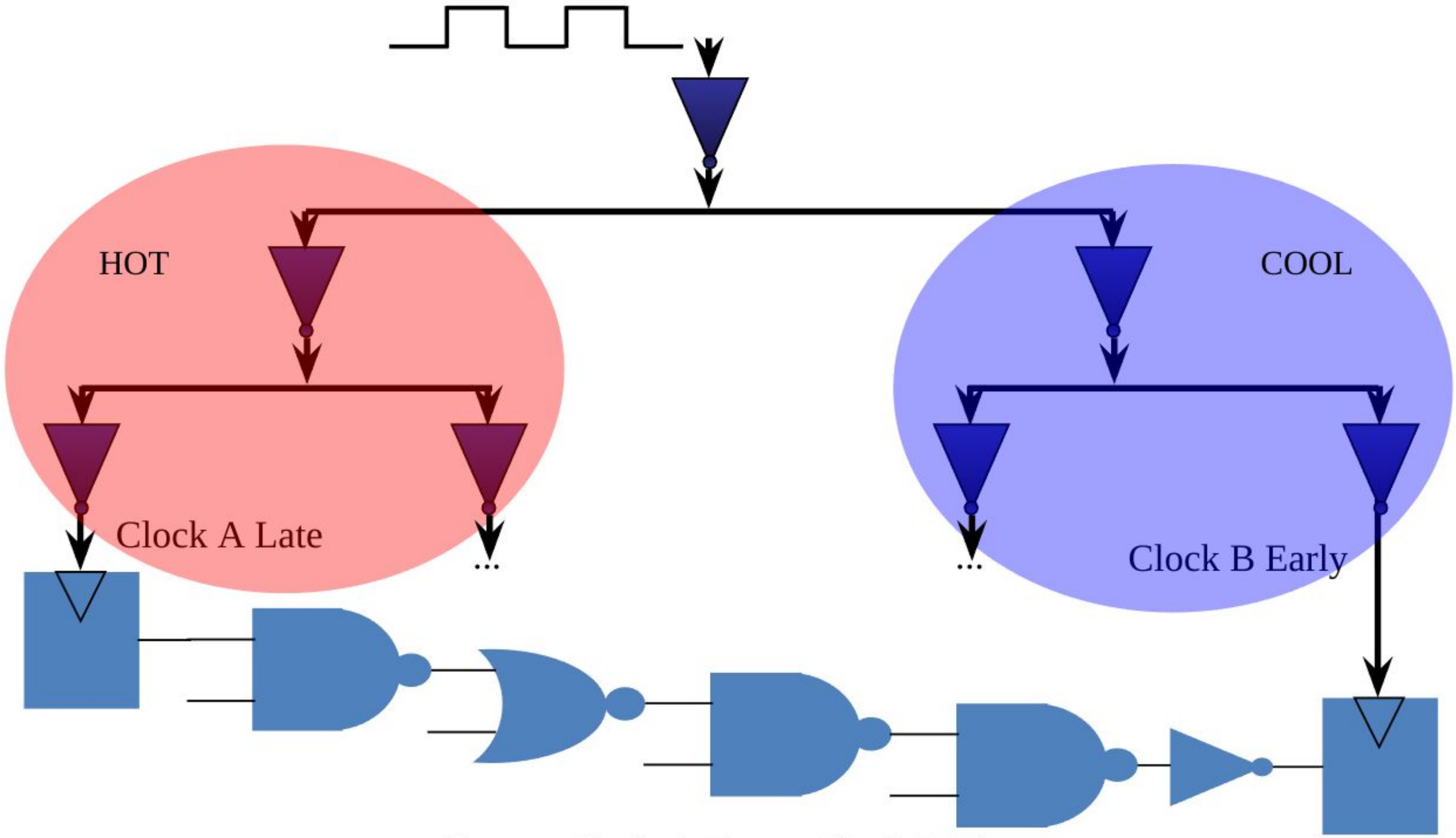
- For synchronized designs, data transfer between functional elements are synchronized by clock signals
- Clock period equation iff connected by a logic path:

$$\text{clock period} \geq t_d + t_{skew} + t_{su} + t_{ds}$$

- No path? False paths? Common path pessimism?

t_d:	Longest path through combinational logic
t_{skew}:	Clock skew
t_{su}:	Setup time of the synchronizing elements
t_{ds}:	Propagation delay within the synchronizing element

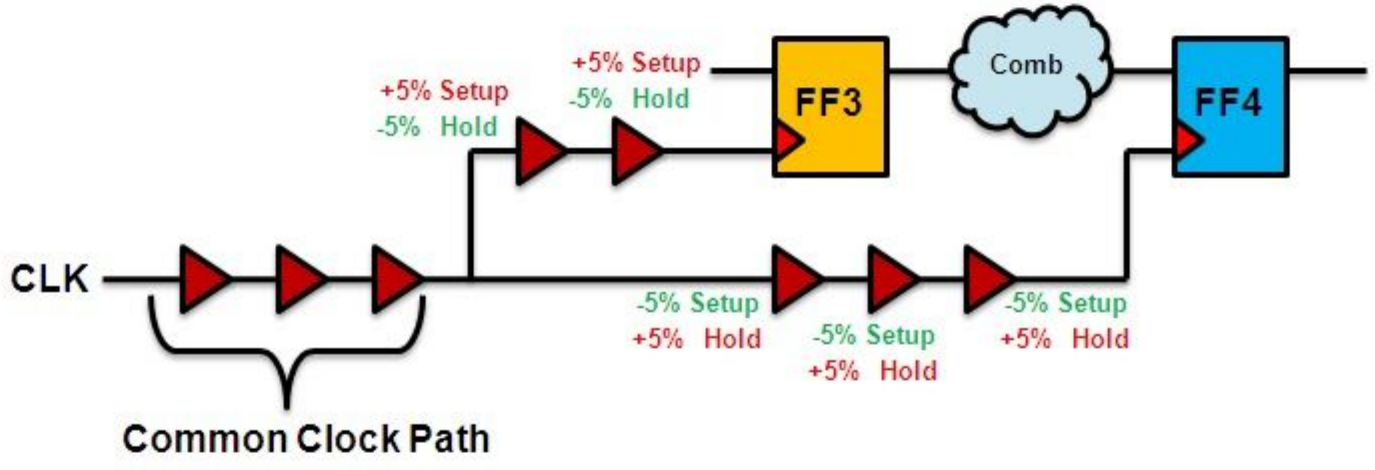
Local Clock Skew



Skew = Clock A Time - Clock B Time

Common Path Pessimism Removal (CPPR)

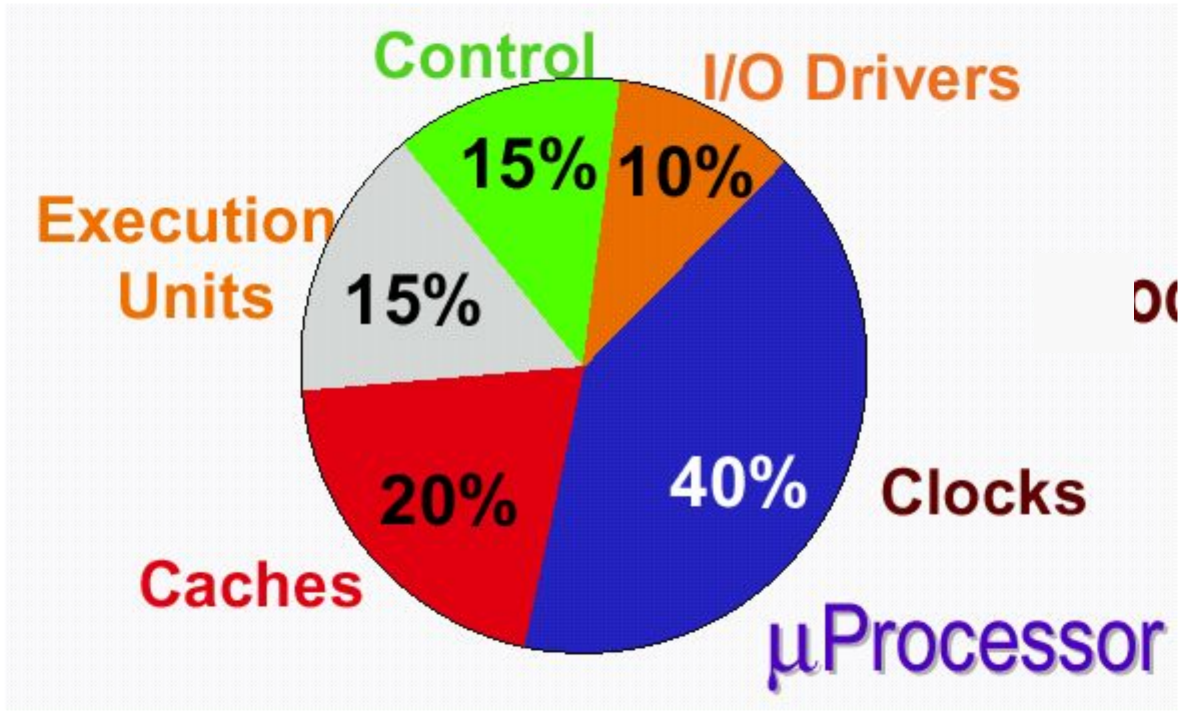
- A.K.A. Clock Reconvergence Pessmism Removal (CRPR)
- Deals with on-chip variation and "derating factors"
- Shared portion of clock can't be both sl



Technology Trends: Power

- Heavily pipelined designs –> more latches, more capacitive load for clock
- Larger chips –> more wirelength needed to cover the entire die
- Complexity –> more functionality and devices means more clocked elements
- Dynamic logic –> more clocked elements

Clocks: Power-Hungry



$$P = \alpha \, C \, V_{dd}^{2} \, f$$

Not only is the clock capacitance large, it switches every cycle!

Clock Distribution Metric: Area

- Clock networks consume silicon area (clock drivers, PLL, etc.) and routing area
- Routing area is most vital
- Top-level metals are used to reduce RC delays
  - These levels are precious resources (unscaled)
  - Power routing, clock routing, key global signals
- By minimizing area used, we also reduce wiring capacitance & power
- Typical #'s: Intel Itanium – 4% of M4/5 used in clock routing

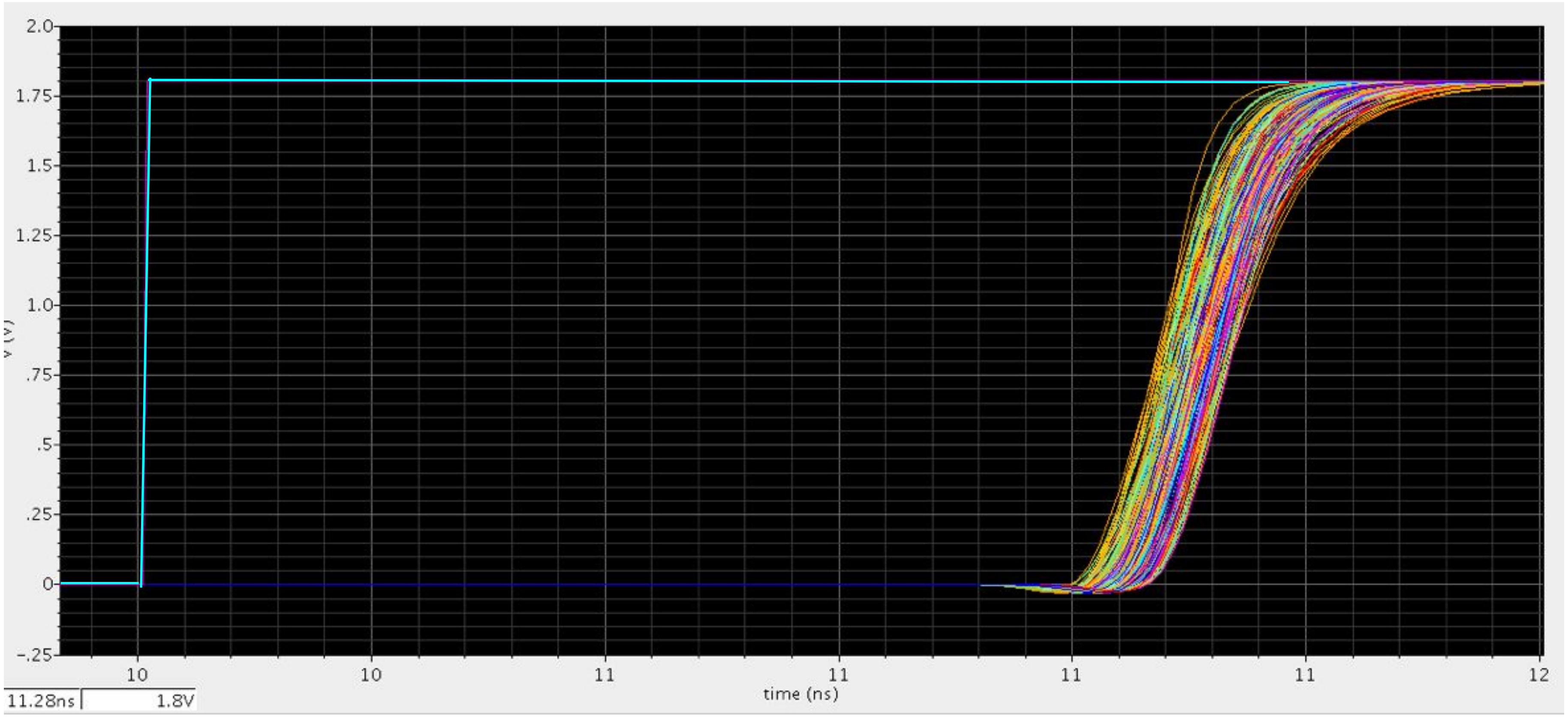
Clock Distribution Metric: Slew Rates

- To maintain signal integrity and latch performance, minimum slew rates are required
- Too slow – clock is more susceptible to noise, latches are slowed down, eats into timing budget
- Too fast – burning too much power, overdesigned network, enhanced ground bounce
- Rule-of-thumb: Trise and Tfall of clock are each between 10-20% of clock period (10% - aggressive target)
  - 1 GHz clock; Trise = Tfall = 100-200ps

Clock Distribution Metric: Insertion Delay

- The delay from the root of the tree to the leaves (sinks)
  - More insertion delay usually means more power because more levels
- What does insertion delay do to variation?
  - Global variation?
  - Local variation?

Skew vs Insertion Delay

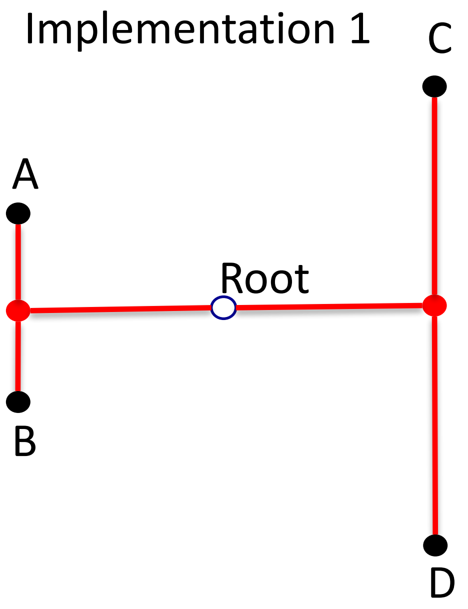


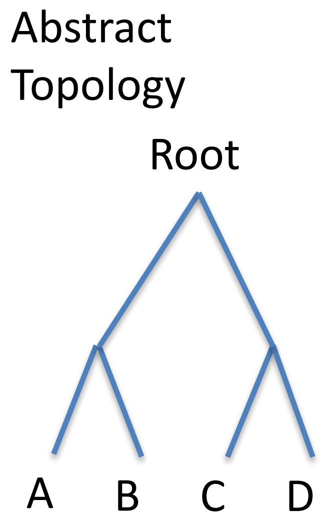
The Clock Routing Problem

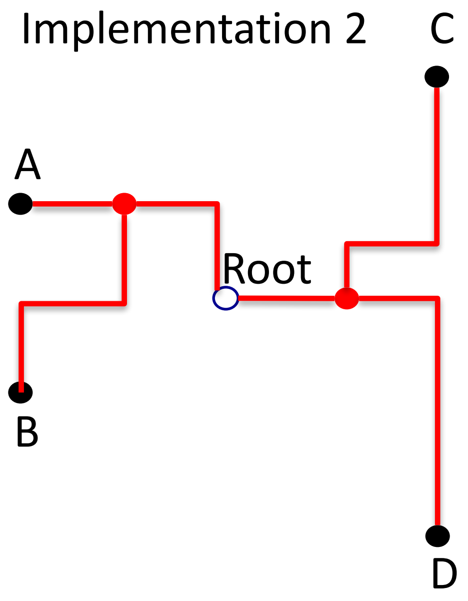
- Clock signal are generated externally (e.g., by PLL)
- Given a source and *n* sinks.
- Connect all sinks to the source by an interconnect network (tree or non-tree) so as to minimize:
  - Clock Skew = $\max_{i,j} |t_i - t_j|$
  - Delay = $\max_i t_i$
  - Power
  - Slew variation
  - Total wirelength
  - Noise and coupling effects
  - Etc

Tree Topology

- Tree topology is merging order of tree
  - Found by clustering or partitioning
- Does not represent actual wires



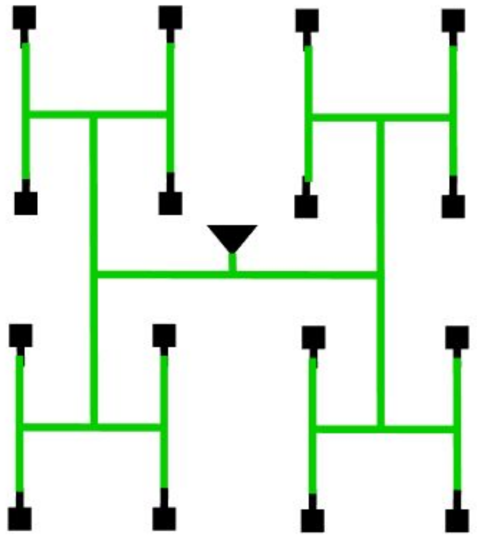




Clock Design Considerations

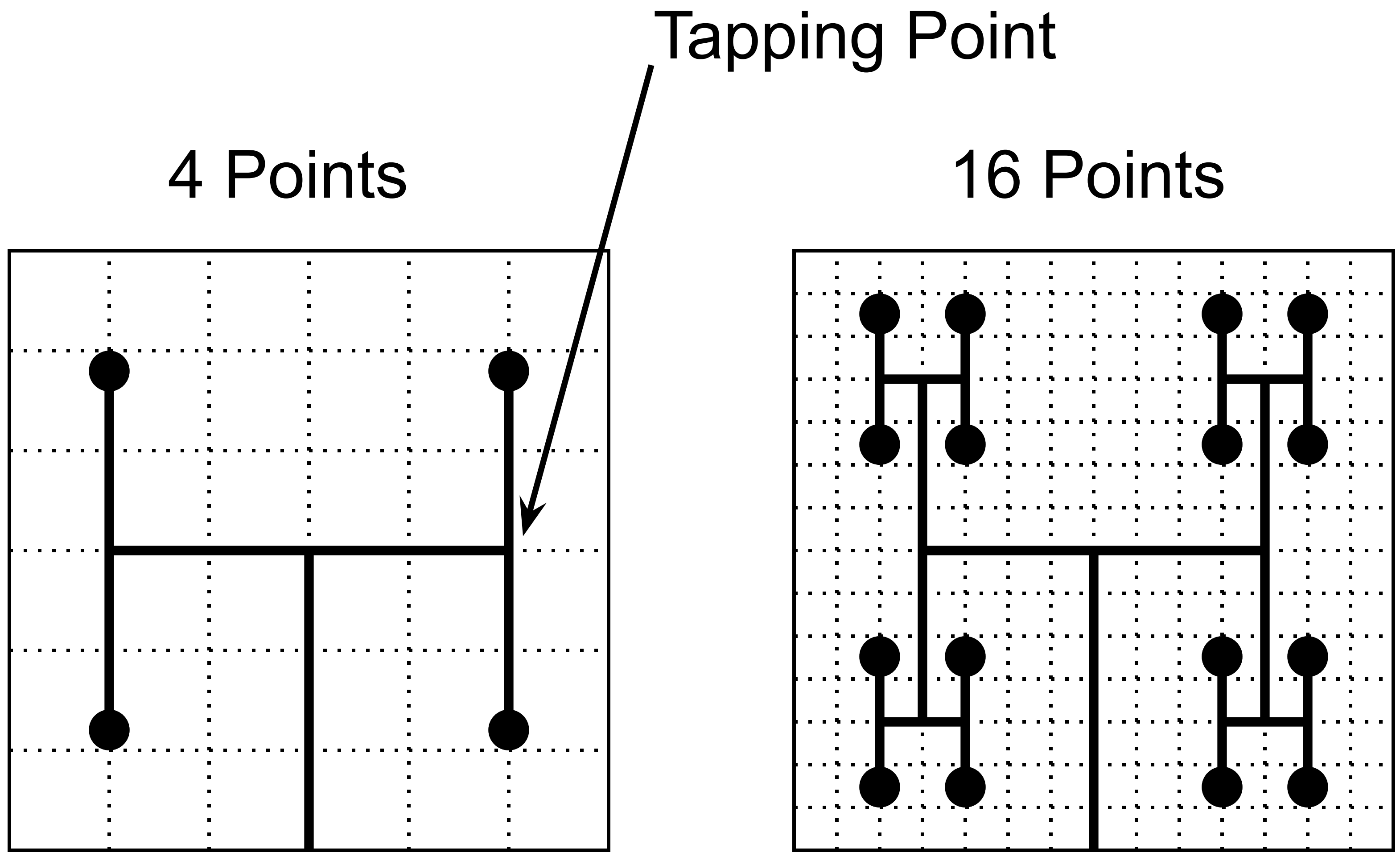
- Clock signal is global in nature, so clock nets are usually very big
  - Significant interconnect capacitance and resistance
- So what are the techniques?
  - Routing
    - Clock tree versus clock mesh (non-tree or grid)
    - Balance skew and total wire length
  - Buffer insertion
    - Clock buffers to reduce clock skew, delay, and distortion in waveform.
  - Wire sizing
    - To further tune the clock tree/mesh

H-Trees



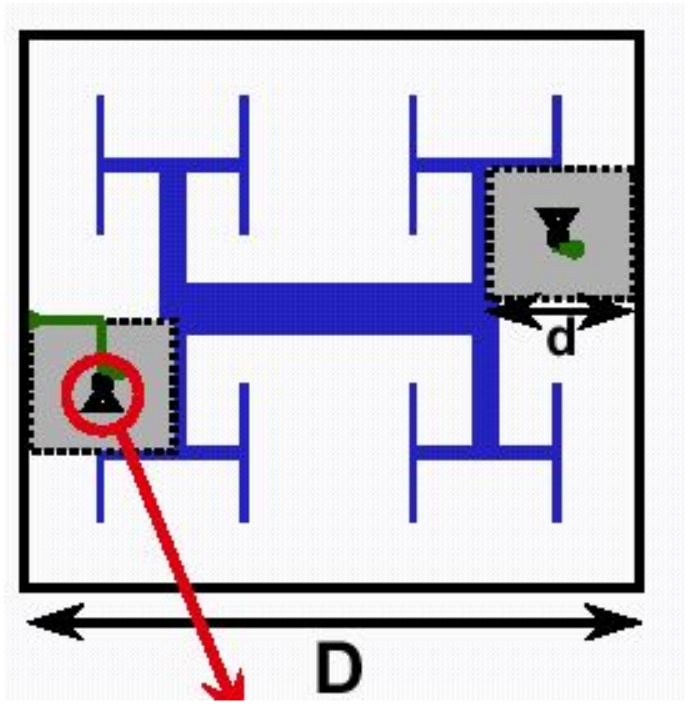
- Uniform (H-Tree)
  - Fairly robust, symmetric
  - Recursive formulation
  - Not minimal wirelength/power

H-Tree Clock Routing



Original H-tree (Bakoglu)

- One large central driver
- Recursive H-style structure to match wirelengths
- Halve wire width at branching points to reduce reflections

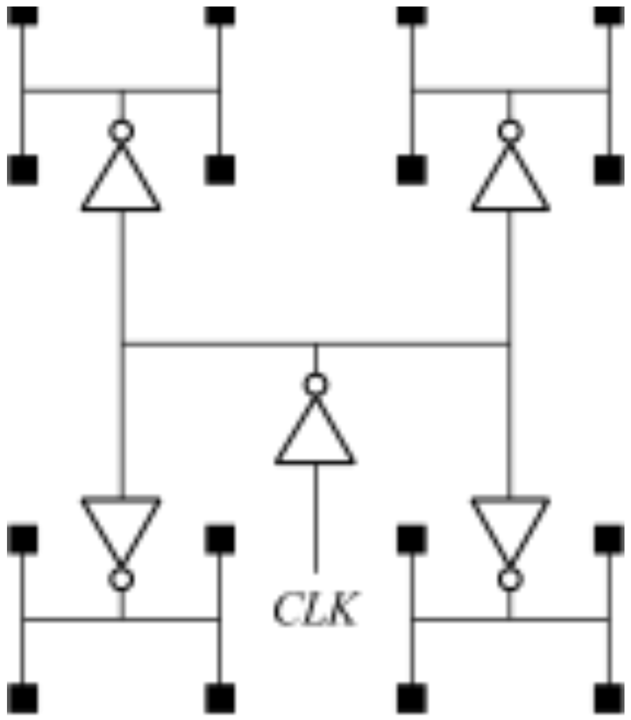


H-Tree Problems

- Drawback to original tree concept
  - Slew degradation along long RC paths
  - Unrealistically large central driver
  - Non-uniform load distribution
- Inherently non-scalable (wire resistance skyrockets)
- Solution to some problems
  - Introduce intermediate buffers along the way
  - Specifically at branching points

Buffered H-tree

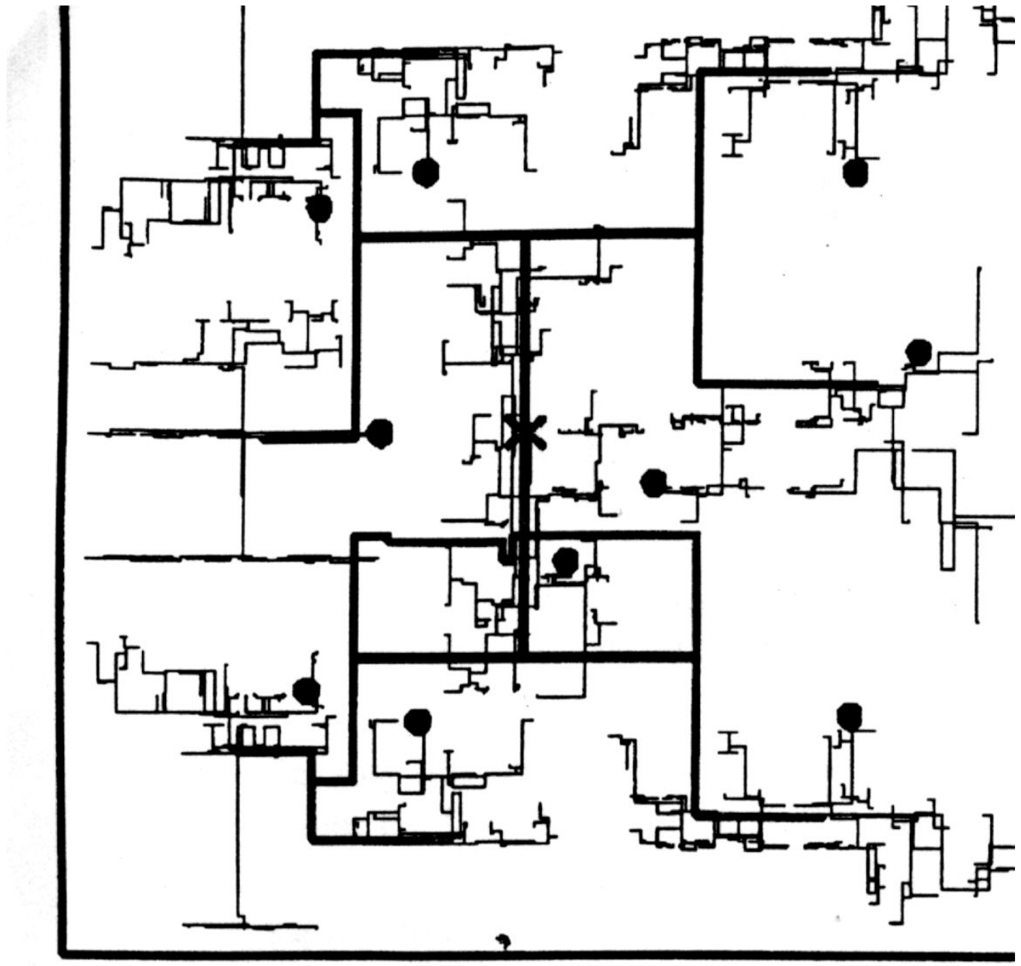
- Advantages
  - Ideally zero-skew
  - Can be low power (depending on skew requirements)
  - Low area (silicon and wiring)
  - CAD tool friendly (regular)
- Disadvantages
  - Sensitive to process variations
  - Local clocking loads are inherently non-uniform
  - Clock sinks are not really uniform!



H-tree Algorithm

- Minimize skew by making interconnections to subunits equal in length
  - Regular pattern
  - The skew is 0 assuming delay is directly proportional to wirelength
    - Is this always the case?
- Can be used when terminals are evenly distributed
  - However, this is never the case in practice (due to blockage, and so on)
  - So strict (pure) H-trees are rarely used
  - However, still popular for top-level clock network design
  - Cons: too costly to be used everywhere

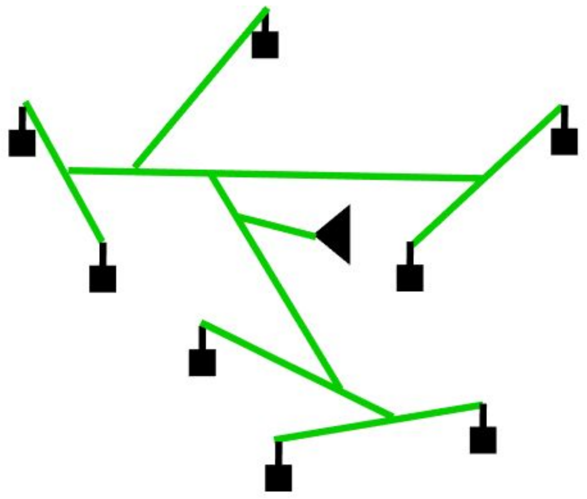
Realistic H-tree

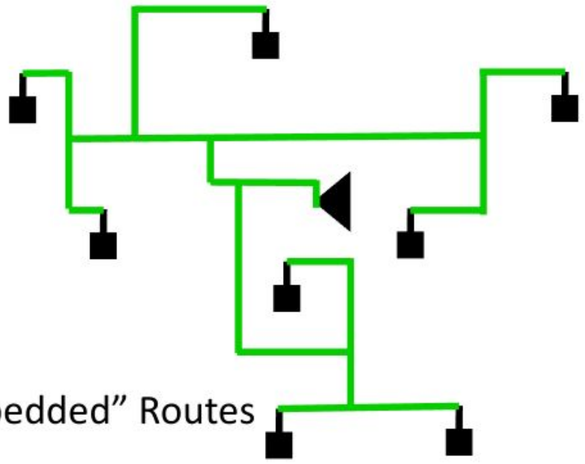


[Restle98]

Balanced Trees

- Advantages
  - Minimum wire length (Very power efficient)
  - Very low nominal skew
  - Can do useful skew!
- Disadvantages
  - Sensitive to process parameters
  - Difficult to deal with blockages

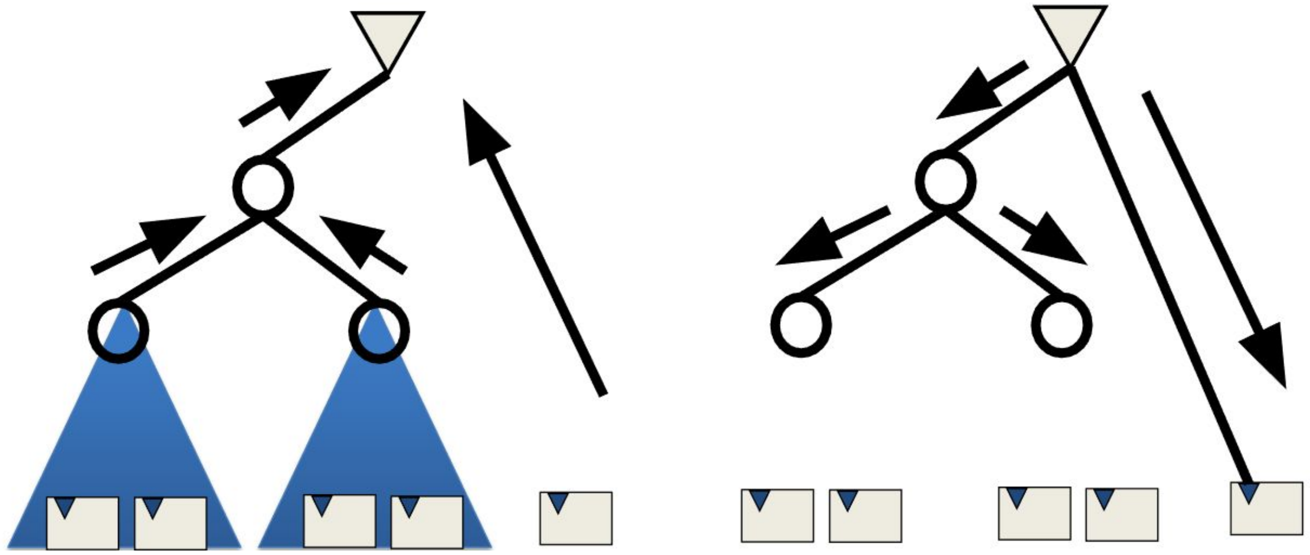




“Embedded” Routes

Top-Down vs. Bottom-Up

- Bottom-Up can use local skew because it knows paths
- Top-Down has a global physical view so knows wire lengths and locations of tree roots above it



Zero Skew Algorithms

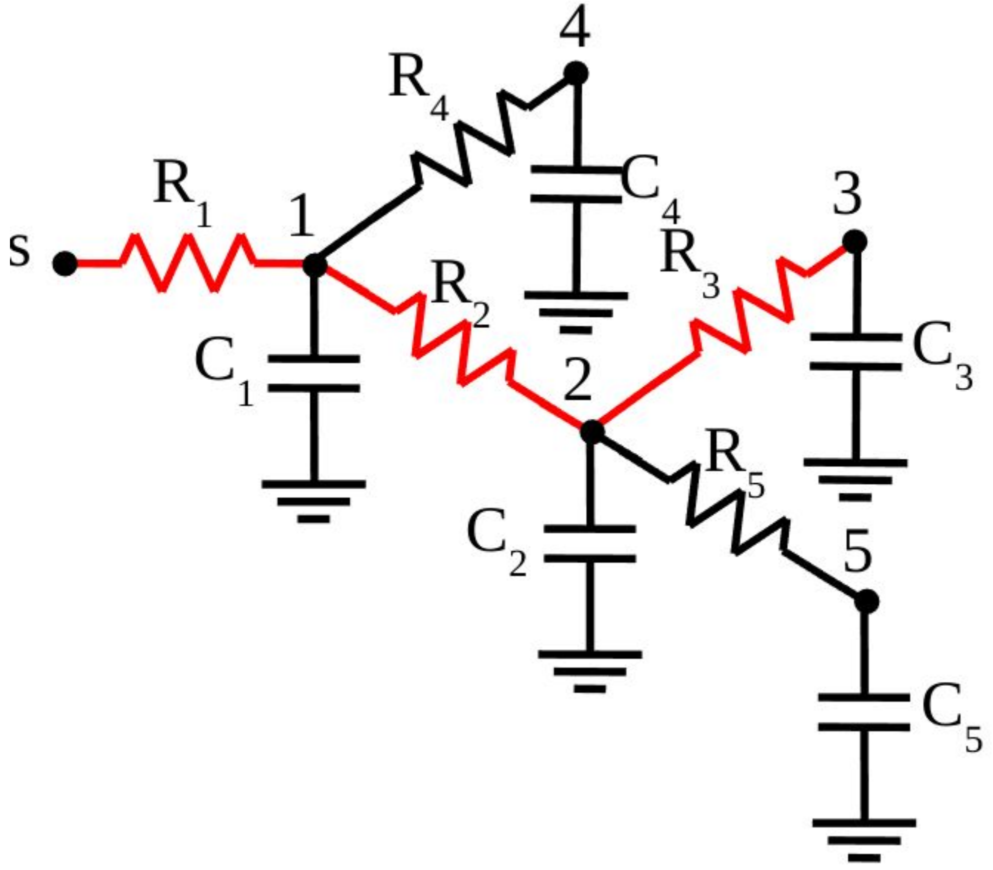
- Independently proposed by several groups
  - Edahiro, NEC Res Dev, 1991
  - Chao et al, DAC'92
  - Boese and Kahng, ASIC'92
- Use Elmore delay to compute delay
- Guarantees zero skew
  - Can easily to extended for zero skew or bounded skew
- Minimizes wire length, but not done very well
  - Lots of follow up works to minimize total wire length while maintaining zero skew
  - DME and its extensions

Deferred Merge Embedding

- As its name implies, DME defers the merging as late as possible, to make sure minimal wire length cost for merging
- DME needs an abstract routing topology as the input
- It has a bottom-up phase followed by a top-down process (sound familiar?)

Skew Calculation

- Elmore delay is not accurate, but high fidelity.
- Fast for optimization.
- How to model slew?



$$\text{Skew} = \max_{i \in (3,4,5)} \tau_i - \min_{i \in (3,4,5)} \tau_i$$

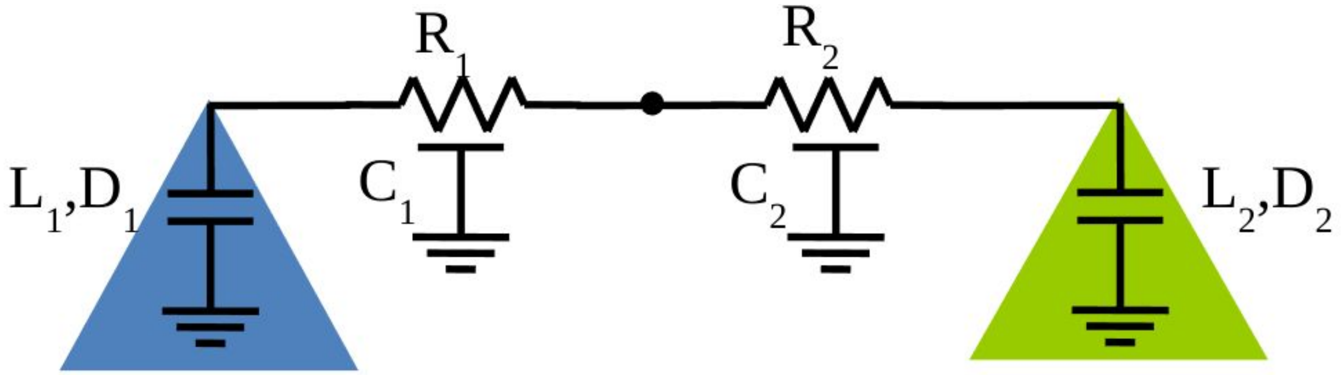
[Elmore, J. App. Physics 1948]

Zero-Skew Step

- Divide distance (D) to equalize Elmore delay
- Assume wires have equal unit R and C
- One equation, one unknown (x)
- Extra wire jogs may be needed

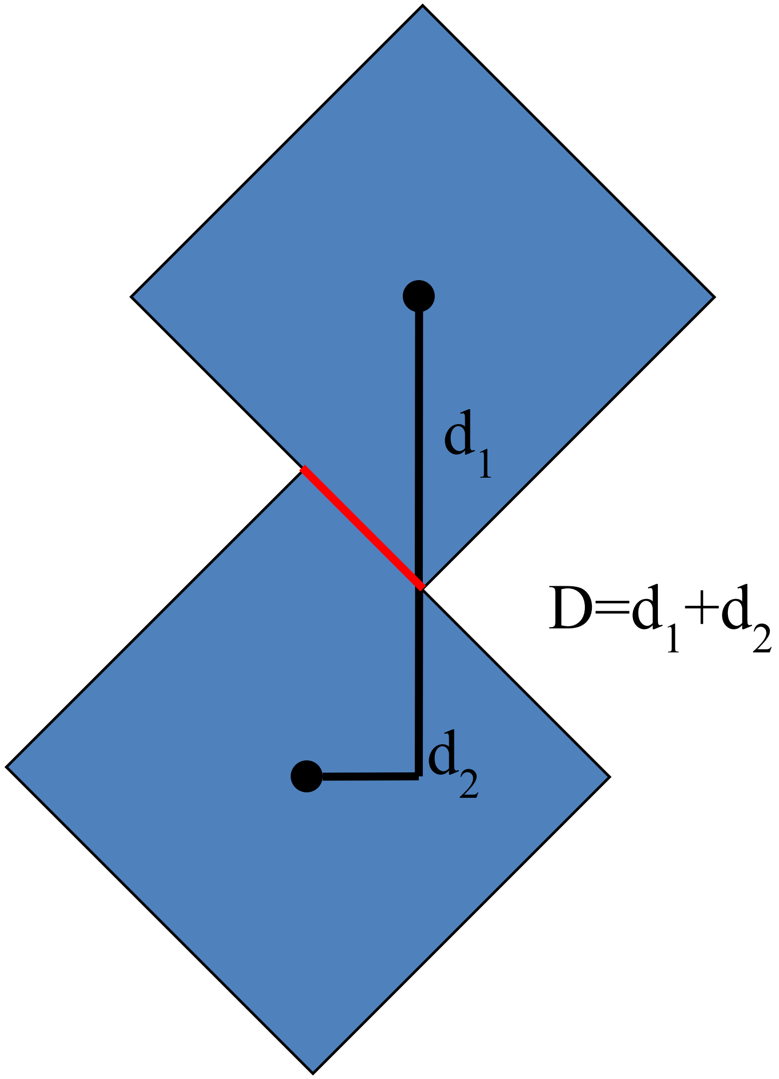
$$\boxed{R_1(C_1/2+L_1)+D_1=R_2(C_2/2+L_2)+D_2}$$

$$D=d_1+d_2 \rightarrow \begin{matrix} d_1=xD \\ d_2=(1-x)D \end{matrix} \rightarrow \begin{matrix} R_1=Rd_1 & R_2=Rd_2 \\ C_1=Cd_1 & C_2=Cd_2 \end{matrix} \rightarrow x?$$

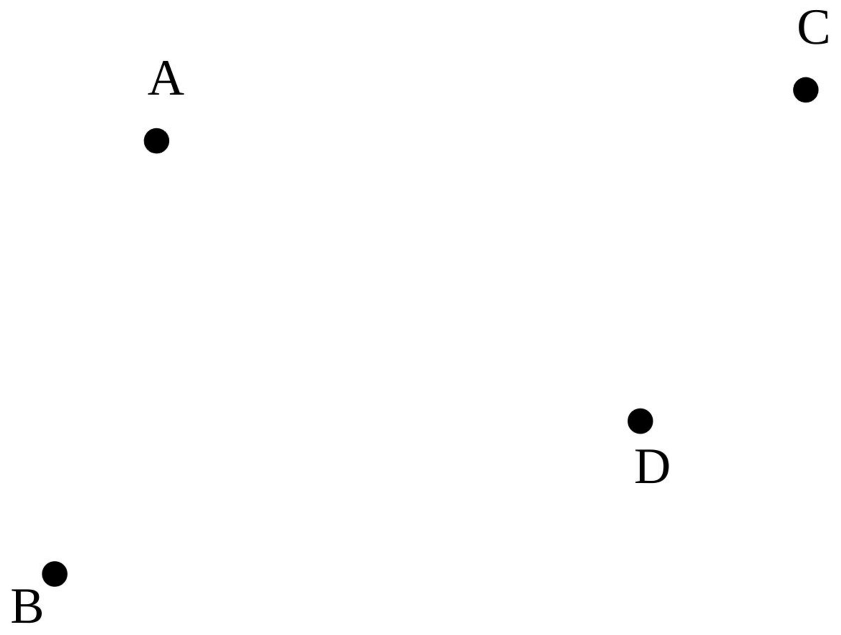


Deferred Merge Embedding

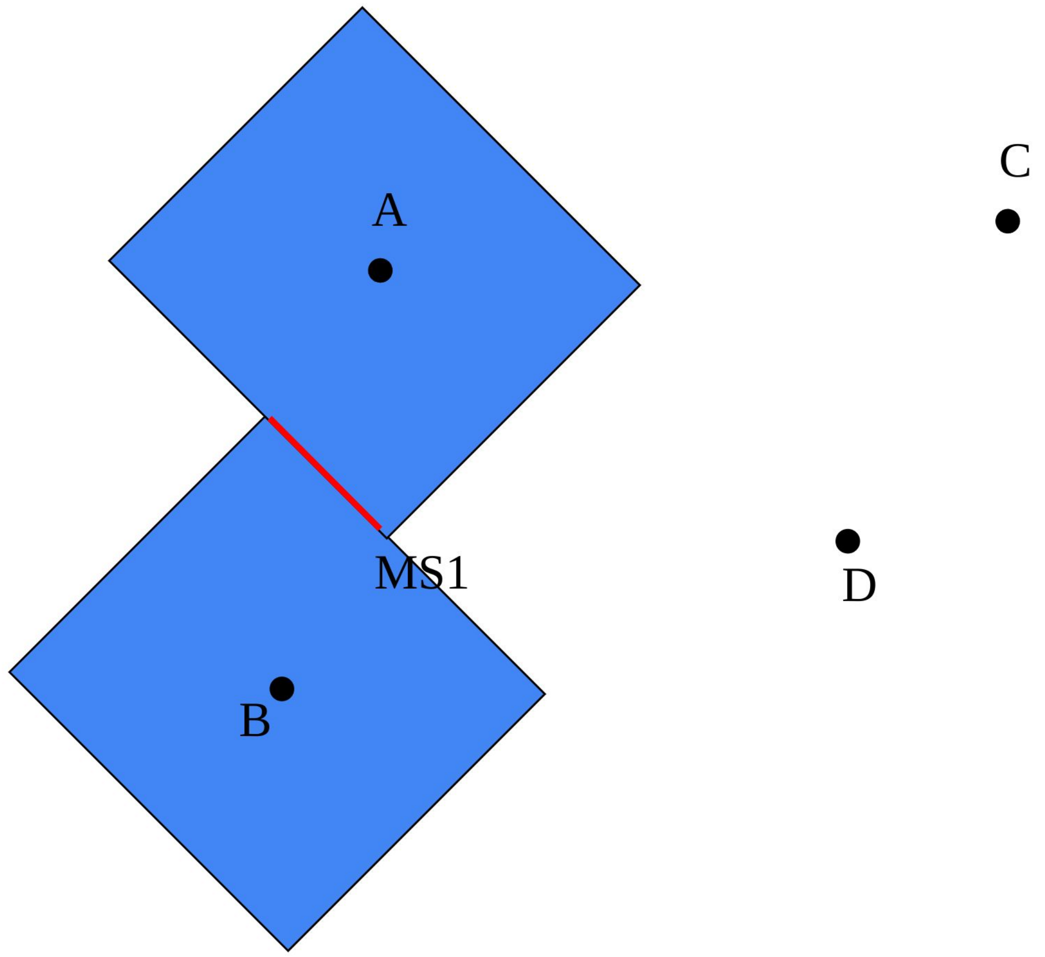
- Topology is fixed
- Equidistant points from a point or segment form a diamond
- Intersection of two diamonds is a Manhattan arc (+1 or -1 slope line)
- Set of zero skew points form a Merging Segment (MS)



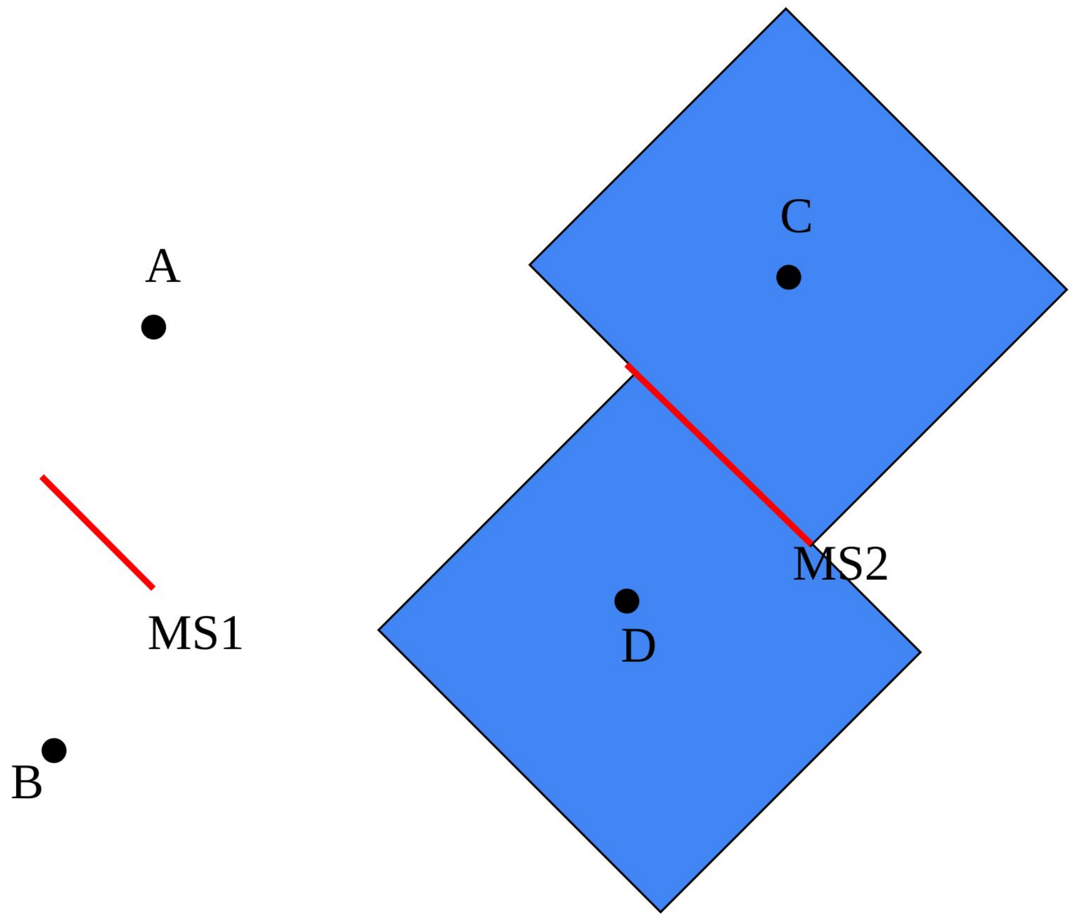
DME Bottom Up



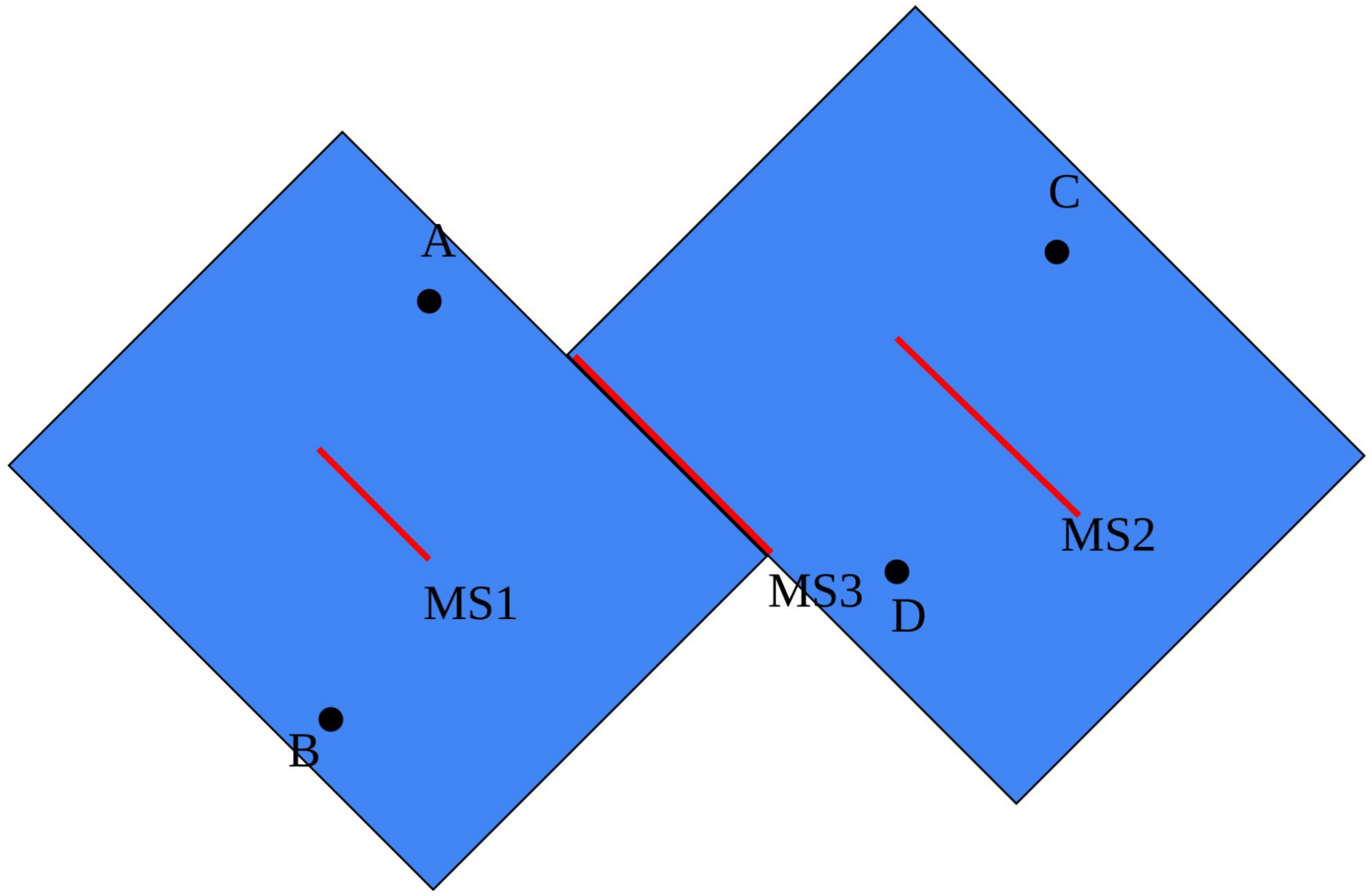
DME Bottom Up



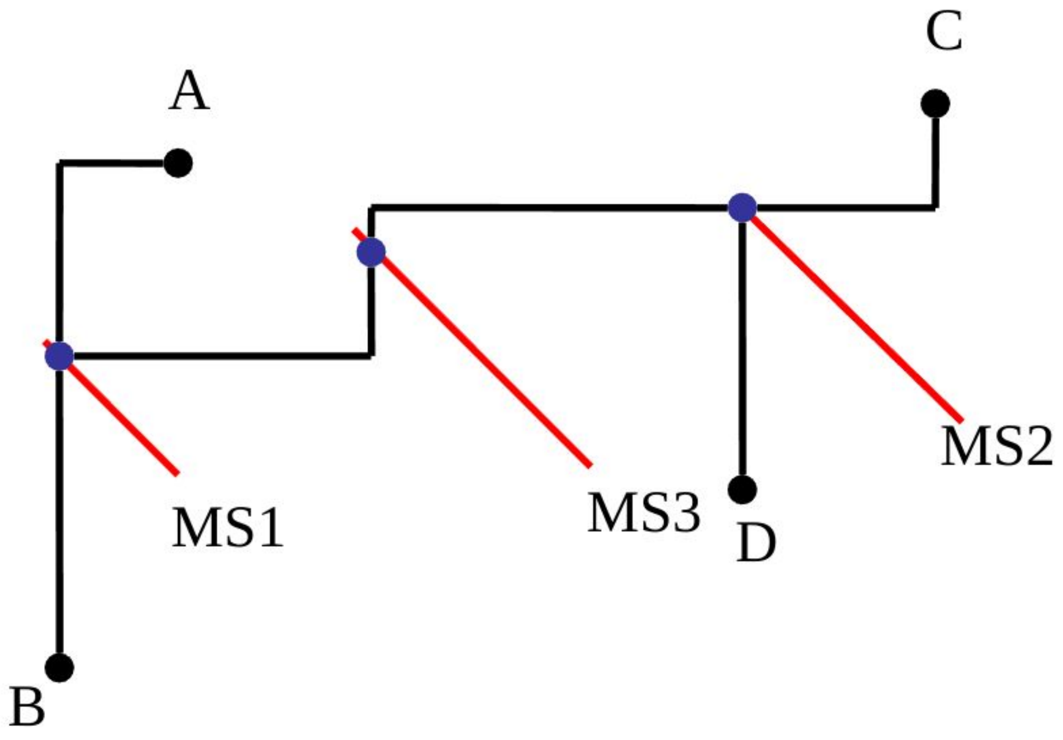
DME Bottom Up



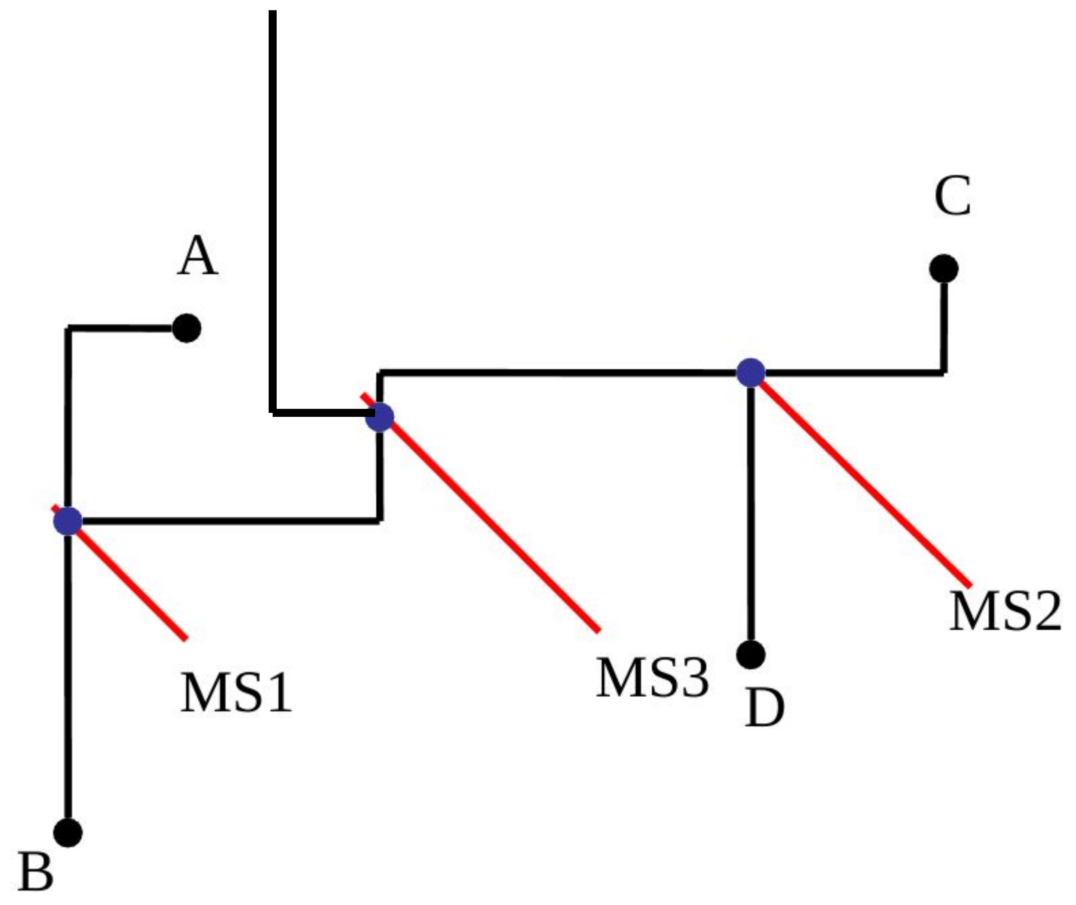
DME Bottom Up



DME Top-Down



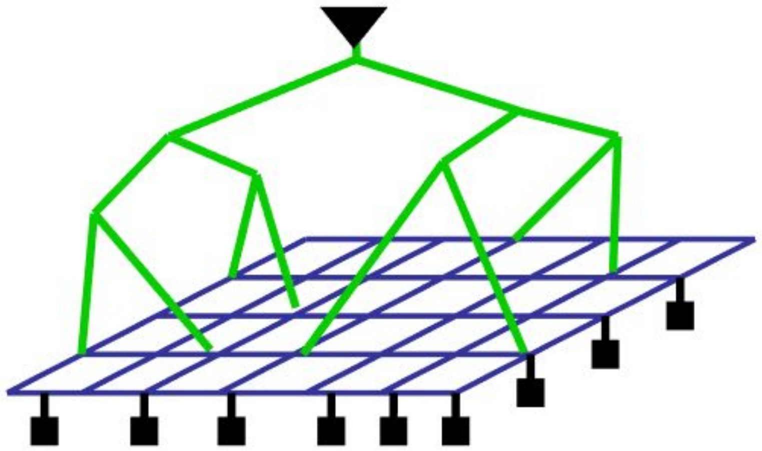
Root Location Doesn’t matter



What about buffers?

- Buffers in DME trees can be considered during zero skew calculation.
  - Use output capacitance limit while merging up
  - Add uniform levels and size after routing the tree
- How to legalize the buffer placement?
  - Moving the buffer will change the wires

Non-Trees: Meshes (Grids)



- Grids [Anderson et al ISSCC'06, Golden et al ISSCC'06]
  - Used by IBM (Power4) and AMD (Hammer)
  - Low variation, but huge power overhead
- In Power4, clock network consumes 70% of total power!

Next Lecture

- SRAM/DRAM video