Physical Design of Digital Integrated Circuits  
(EN0291 S40)

Sherief Reda  
Division of Engineering, Brown University  
Fall 2006
Lecture 08: Interconnect Trees

• Introduction to Graphs and Trees
• Minimum Spanning Trees
• Steiner Minimum Trees
• Delay of Interconnect trees
• Clock Trees
  – H-trees
  – Zero-skew trees
  – Prescribed-skew trees
Graphs

• A graph is a pair of sets $G=\langle V, E \rangle$
  – $V$: set of vertices/nodes (representing logic gates/standard cells)
  – $E$: set of edges (representing interconnects)

• A path is a sequence of edges where no vertex more than once; a cycle is a path that starts and terminates at the same node

• Two nodes $u$ and $v$ are connected in $G$ if there exists a path between them in $G$

• A graph is connected if there exists a path between every pair of nodes
Hypergraphs

• A graph is a pair of sets $H=(V, E)$
  – $V$: set of vertices/nodes (representing logic gates/standard cells)
  – $E$: set of hyperedges (representing interconnects)

• A hyperedge, $e$, is an edge that connects two or more nodes
  – Ideal for modeling an interconnect where the output of one node is connected to many inputs
  – *Degree* of hyperedge $= \text{number of nodes that belong to it}$
Trees

- A tree is a connected graph with a small twist:
  - There exists *only one path* between every pair of nodes \(\rightarrow\) no cycles

- We can define weights for every edge in a graph
  - Reflect cost of traversal/using a certain edge
Relationship between paths, trees, graphs and hypergraphs
Lecture 08: Interconnect Trees

• Introduction to Graphs and Trees
• Minimum Spanning Trees
• Steiner Minimum Trees
• Delay of Interconnect Trees
• Clock Trees
  – H-trees
  – Zero-skew trees
  – Prescribed-skew trees
Minimum Spanning Tree

- The spanning tree of a graph $G=(V, E)$ is a tree that has
  - same nodes as $G$
  - smallest set of edges needed to connect everything together

- If edges have weights
  The spanning tree that has the minimum total weight is called
  *Minimum Spanning Tree (MST)*
How to generate a spanning tree from a given spanning tree?

Add an edge that belong to the graph but not in the current tree

- forms a cycle
- delete an edge from the cycle
How to find a MST?

There are three algorithms: Kruskal / Prim / Boruvka

- Sort all edges ascendingly according to their weight/length: (a, e, d, c, f, b)
- Initialize the tree to be empty
- Following the sorted list:
  - add edges to the tree as long as they do not form a cycle

Complexity: $O(|E| \log |E|)$

Why is this algorithm correct?
Steiner Minimum Tree

Find a minimum spanning tree that spans all nodes without help of other nodes → minimum spanning tree with help of a subset of nodes → Steiner Minimum Tree (SMT)

Finding the SMT is much harder than finding the MST

SMT is an NP-hard problem
Rectilinear Steiner Minimum Tree

<table>
<thead>
<tr>
<th>Points in 2-D plane</th>
<th>MST</th>
<th>SMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Steiner point</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- What are the candidate Steiner points in a rectilinear SMT problem?

Steiner points always lie on the Hanan grid
How to find a “good” SMT?

1. Construct a minimum spanning tree without using any Steiner points
2. Until there is no possible improvement in WL
   1. Consider each Steiner point \( x \)
      1. Calculate the MST consisting of the current points plus \( x \)
      2. Keep track of the best point \( x_b \)
   2. Add the best Steiner point \( x_b \) to the tree
How to calculate delay of Steiner trees?

Elmore Delay (see Lecture03):

- Delay from \( a \) → \( b \): \( R_1(C_1+C_2+C_3+C_4+C_5)+R_3C_3 \)
- Delay from \( a \) → \( c \): \( R_1(C_1+C_2+C_3+C_4+C_5)+R_3C_3 \)
- Delay from \( a \) → \( e \): \( R_1(C_1+C_2+C_3+C_4+C_5)+R_4(C_4+C_5) \)
- Delay from \( a \) → \( f \): \( R_1(C_1+C_2+C_3+C_4+C_5)+R_4(C_4+C_5)+R_5C_5 \)
Lecture 08: Interconnect Trees

- Introduction to Graphs and Trees
- Minimum Spanning Trees
- Steiner Minimum Trees
- Delay of Interconnect trees

- Clock Trees
  - Meshes
  - H-Trees
  - Zero-Skew Trees
  - Useful skew
  - Impact of process variations
Objectives of Clock Tree Design

• **Function:**
  Generate a clock signal; distribute to all FFs to synchronize all events in a chip, i.e., establish an order for different events

• **Objectives in design:**
  - Minimize clock period → improves performance
  - Manage skew → no setup/hold time violations (see lecture04).
  - Maintain signal integrity (minimize slew rate)
  - Minimize metal demand and power consumption
  - Minimize jitter (uncertainty in clock arrival time)

• Skew can be helpful or harmful (lecture04), but jitter is always harmful
Clock Tree Synthesis in the P & R flow

Using Cadence Encounter

placed circuit

clock tree

location of all FFs

clock tree + all routes

s38417
Components of Clock Trees

- PLL
- Reference clock signal
- CPU clock
- Buffer
- Macro/IP
- FF
Unplanned skew potentially reduces performance

- $s_i$ and $s_j$ skew of clock at FF $i$ and $j$
- $d_{\text{min}} \leq d \leq d_{\text{max}}$
- $s_i + \Delta + d_{\text{max}} \leq s_j + P - T_s$
  \[ \rightarrow d_{\text{max}} \leq P - T_s - \Delta + s_j - s_i \] (setup time)
- $s_i + \Delta + d_{\text{min}} \geq T_h + s_j$
  \[ \rightarrow d_{\text{min}} \geq T_h - \Delta + s_j - s_i \] (hold time)

[From lecture04]
Clock meshes

Advantages:
• Low skew values
• Larger current values
• Tolerant to process variations

Disadvantages:
• Consume large metal resources
• Consume huge power
Clock Trees: H-Trees

- Simple and regular to construct
- Require very regular placement of FFs (which is unlikely to happen)
- Buffers at the same level has to be matched identically
Clock Trees: Method of Means/Medians

- Bisect regions according to the median
- Join parts using their center of mass (mean)
- **Advantages**: simple to understand and yields good results → not always zero skew
Clock Trees: Zero-Skew Trees

Using the Deferred Merged Embedding (DME) technique

**Advantages:** zero-skew at minimum wirelength

→ requires an input tree topology
Useful clock skew

- Zero clock skew ($s_i = 0$ & $s_j = 0$) $\Rightarrow$ clock period = 10ns, $f_{\text{max}} = 100$MHz
- $S_i = -1$ $s_j = 0$ $\Rightarrow$ clock period = 9ns, $f_{\text{max}} = 111$MHz (no timing violations)
- $S_i = -2$ $s_j = 0$ $\Rightarrow$ clock period = 8ns, $f_{\text{max}} = 125$MHz (no timing violations)
- Introducing skew also helps minimize the simultaneous switching of FFs $\Rightarrow$ less load on the P/G network

How to modify the previous clock tree method (DME) to produce prescribed clock skew?
Clock tree gating

- Shuts down clock from a logic region → no more switching
  → saves power
Impact of Process Variability

Clock skew induced by process variations in Pentium 4
[Kommandur/Bannerjee]

Reasons:
Variations in gate lengths, threshold, supply voltage, temperature, and interconnect dimensions
Next lecture

• Guest lecture by Prof. Alex Zaslavsky on future devices and how it can impact circuit design
• Project presentations + reports