Digital Circuits

- Logic Effort
- Elmore Delay
- Power Dissipation
- Other Logic Styles
# Propagation Delay in Multiple-Levels of Logic with Stage Loading

**Asymmetric-sized gates**

<table>
<thead>
<tr>
<th>C&lt;sub&gt;IN/C&lt;sub&gt;REF</th>
<th>Equal Rise/Fall</th>
<th>Equal Rise/Fall (with OD)</th>
<th>Minimum Sized</th>
<th>Asymmetric OD (OD&lt;sub&gt;HL&lt;/sub&gt;, OD&lt;sub&gt;LH&lt;/sub&gt;)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inverter</td>
<td><em>od</em></td>
<td>OD</td>
<td>1/2</td>
<td>[OD&lt;sub&gt;HL&lt;/sub&gt; + 3 • OD&lt;sub&gt;LH&lt;/sub&gt;] / 4</td>
</tr>
<tr>
<td>NOR</td>
<td>3k+1/4</td>
<td>3k+1 • OD</td>
<td>1/2</td>
<td>[OD&lt;sub&gt;HL&lt;/sub&gt; + 3k • OD&lt;sub&gt;LH&lt;/sub&gt;] / 4</td>
</tr>
<tr>
<td>NAND</td>
<td>3+k/4</td>
<td>3+k • OD</td>
<td>1/2</td>
<td>k • OD&lt;sub&gt;HL&lt;/sub&gt; + 3 • OD&lt;sub&gt;LH&lt;/sub&gt; / 4</td>
</tr>
</tbody>
</table>

**Review from Last Time**

\[
T_{PROP} = T_{REF} \left( \frac{1}{2} \sum_{k=1}^{n} F_{i(k+1)} \left( \frac{1}{OD_{HL_k}} + \frac{1}{OD_{LH_k}} \right) \right)
\]
Review from Last Time

Propagation Delay in Multiple-Levels of Logic with Stage Loading

- Equal rise/fall (no overdrive)

- Equal rise/fall with overdrive

- Minimum Sized

- Asymmetric overdrive

- Combination of equal rise/fall, minimum size and overdrive
Review from Last Time

Optimal Driving of Capacitive Loads

\[ \theta_{\text{OPT}} = e \]

\[ n_{\text{OPT}} = \ln \left( \frac{C_L}{C_{\text{REF}}} \right) \]

\[ t_{\text{PROP}} = t_{\text{REF}} \left( \frac{\theta}{\ln(\theta)} \right) \left[ \ln \left( \frac{C_L}{C_{\text{REF}}} \right) \right] \]

\[ t_{\text{PROP}} = t_{\text{REF}} e \left[ \ln \left( \frac{C_L}{C_{\text{REF}}} \right) \right] = n\theta t_{\text{REF}} \]
Optimal Driving of Capacitive Loads

- Often termed a pad driver
- Often used to drive large internal busses as well
- Generally included in standard cells or in cell library
- Device sizes can become very large
- Odd number of stages will cause signal inversion but usually not a problem
Digital Circuit Design

- Hierarchical Design
- Basic Logic Gates
- Properties of Logic Families
- Characterization of CMOS Inverter
- Static CMOS Logic Gates
  - Ratio Logic
- Propagation Delay
  - Simple analytical models
    - FI/OD
    - Logical Effort
  - Elmore Delay
- Sizing of Gates
  - done
  - partial

- Propagation Delay with Multiple Levels of Logic
- Optimal driving of Large Capacitive Loads
  - Power Dissipation in Logic Circuits
  - Other Logic Styles
  - Array Logic
  - Ring Oscillators
Propagation Delay in “Logic Effort” approach

(Discussed in Chapter 4 of Text but definitions are not rigorous)

Propagation delay for equal rise/fall gates was derived to be

$$t_{PROP} = t_{REF} \sum_{k=1}^{n} \frac{F_{l(k+1)}}{OD_k}$$

Delay calculations with “logical effort” approach

Logical effort delay approach:

$$t_{PROP} = \sum_{k=1}^{n} f_k$$

(t_{REF} scaling factor not explicitly stated)

where $f_k$ is the “effort delay” of stage $k$

$$f_k = g_k h_k$$

g_k = \text{logical effort}

h_k = \text{electrical effort}
Propagation Delay in “Logic Effort” approach

\[ t_{\text{PROP}} = \sum_{k=1}^{n} f_k \]

\[ f_k = g_k h_k \]

\[ f_k = \text{“effort delay” of stage k} \]

\[ g_k = \text{logical effort} \]

\[ h_k = \text{electrical effort} \]

Logic Effort is the ratio of the input capacitance of a gate to the input capacitance of an inverter that can deliver the same output current.

Electrical Effort is the ratio of the gate load capacitance to the input capacitance of a gate.
Propagation Delay in “Logic Effort” approach

\[ t_{PROP} = \sum_{k=1}^{n} f_k \quad f_k = g_k h_k \]

Logic Effort \((g)\) is the ratio of the input capacitance of a gate to the input capacitance of an inverter that can deliver the same output current.

Electrical Effort \((h)\) is the ratio of the gate load capacitance to the input capacitance of a gate.

\[ g_k = \frac{C_{IN_k}}{C_{REF} \cdot OD_k} \quad h_k = \frac{C_{REF} \cdot F_l_{k+1}}{C_{IN_k}} \]
Propagation Delay in “Logic Effort” approach

\[ t_{\text{PROP}} = \sum_{k=1}^{n} f_k \]

\[ f_k = g_k h_k \]

\[ g_k = \frac{C_{\text{IN}k}}{C_{\text{REF}} \cdot \text{OD}_k} \]

\[ h_k = \frac{C_{\text{REF}} \cdot F_{l(k+1)}}{C_{\text{IN}k}} \]

\[ f_k = \left( \frac{C_{\text{IN}k}}{C_{\text{REF}} \cdot \text{OD}_k} \right) \left( \frac{C_{\text{REF}} \cdot F_{l(k+1)}}{C_{\text{IN}k}} \right) \]

\[ f_k = \frac{F_{l(k+1)}}{\text{OD}_k} \]

\[ t_{\text{PROP}} = \sum_{k=1}^{n} f_k = \sum_{k=1}^{n} g_k h_k = \sum_{k=1}^{n} \frac{F_{l(k+1)}}{\text{OD}_k} \]
Propagation Delay in “Logic Effort” approach

\[ t_{\text{PROP}} = \sum_{k=1}^{n} f_k = \sum_{k=1}^{n} g_k h_k = \sum_{k=1}^{n} \frac{F_{l(k+1)}}{OD_k} \]

- Note with the exception of the \( t_{\text{REF}} \) scaling factor, this expression is identical to what we have derived previously.

- Probably more tedious to use the “Logical Effort” approach.

- Extensions to asymmetric overdrive factors may not be trivial.

- Extensions to include parasitics may be tedious as well.

- Logical Effort is widely used throughout the industry.
Digital Circuit Design

- Hierarchical Design
- Basic Logic Gates
- Properties of Logic Families
- Characterization of CMOS Inverter
- Static CMOS Logic Gates
  - Ratio Logic
- Propagation Delay
  - Simple analytical models
    - FI/OD
    - Logical Effort
- Elmore Delay
- Sizing of Gates

- Propagation Delay with Multiple Levels of Logic
- Optimal driving of Large Capacitive Loads
  - Power Dissipation in Logic Circuits
  - Other Logic Styles
  - Array Logic
  - Ring Oscillators

- done
- partial
Elmore Delay Calculations

- Interconnects have a distributed resistance and a distributed capacitance
  - Often modeled as resistance/unit length and capacitance per unit length

- These delay the propagation of the signal

- Effectively a transmission line
  - Analysis is really complicated

- Can have much more complicated geometries
Elmore Delay Calculations

Can have much more complicated geometries
Elmore Delay Calculations

For $x_1 < x_2 < x_3$

\[ V_{\text{IN}}(t) = \begin{cases} 0 & t = 0 \\ V & t > 0 \end{cases} \]

\[ V(x_1)(t) = \begin{cases} 0 & t = 0 \\ V(x_1) & t > 0 \end{cases} \]

\[ V(x_2)(t) = \begin{cases} 0 & t = 0 \\ V(x_2) & t > 0 \end{cases} \]

\[ V(x_3)(t) = \begin{cases} 0 & t = 0 \\ V(x_3) & t > 0 \end{cases} \]
Elmore Delay Calculations

A lumped element model of transmission line

Even this lumped model is 4-th order and a closed-form solution is very tedious!

Need a quick (and reasonably good) approximation to the delay of a delay line!!
Elmore Delay Calculations

It can be shown that this is a reasonably good approximation to the actual delay.

Numbering is critical (resistors and capacitors numbered from input to output).

As stated, only applies to this specific structure.

Elmore delay:

\[ t_{PD} = \sum_{i=1}^{n} \left( C_i \sum_{j=1}^{i} R_j \right) \]
Elmore Delay Calculations

Elmore delay: \[ t_{PD} = \sum_{i=1}^{n} \left( C_i \sum_{j=1}^{i} R_j \right) \]

• Note error in text on Page 161 of first edition of WH

\[ t_{pd} = \sum_{i} R_{n-i} C_i = \sum_{i=1}^{N} C_i \sum_{j=i}^{i} R_j \]

• Not detailed definition on Page 150 of second edition of WH
Elmore delay is a simple approximation to the delay through an RC network in an electronic system. It is often used in applications such as logic synthesis, delay calculation, static timing analysis, placement and routing, since it is simple to compute (especially in tree structured networks, which are the vast majority of signal nets within ICs) and is reasonably accurate. Even where it is not accurate, it is usually faithful, in the sense that reducing the Elmore delay will almost always reduce the true delay, so it is still useful in optimization.

Elmore Delay Calculations

**Example:**

![Circuit Diagram](image)

**Elmore delay:**

\[
 t_{PD} = \sum_{i=1}^{4} \left( C_i \sum_{j=1}^{i} R_j \right)
\]

where

\[
 t_i = C_i \sum_{j=1}^{i} R_j \quad j = 1, 2, 3, 4
\]

**What is really happening?**

- Creating 4 first-order circuits
- Delay to \( V_1 \), \( V_2 \), \( V_3 \) and \( V_4 \) calculated separately by considering capacitors one at a time and assuming others are 0
Elmore Delay Calculations

Extensions:

Lumped Network Model:
Elmore Delay Calculations

Extensions:

1. Create a lumped element model

2. Create a path from input to output
Elmore Delay Calculations

Extensions:

3. Renumber elements along path from input to output and neglect off-path elements.

4. Use Elmore Delay equation for elements on this RC network.

\[ t_{PD} = \sum_{i=1}^{4} \left( C_i \sum_{j=1}^{i} R_j \right) \]
Elmore Delay Calculations

How is a resistive load handled?
Elmore Delay Calculations

Example with resistive load:

Elmore delay:

\[ t_{PD} = \sum_{i=1}^{4} C_i \sum_{j=1}^{i} R_j \]

where

\[ t_{PD} = \sum_{i=1}^{4} (t_i) \]

\[ t_i = C_i \sum_{j=1}^{i} R_j \quad j = 1, 2, 3 \]

\[ t_4 = C_4 \left( \sum_{j=1}^{4} R_j \right) / R_5 \]
Elmore Delay Calculations

With resistive load:

\[
t = C R C R \left/ R \right.
\]

Simple Elmore delay:

\[
t_{PD} = \sum_{i=1}^{n-1} \left( C_i \sum_{j=1}^{i} R_j \right) + C_n \left( \sum_{j=1}^{n} R_j \right) \left/ R_L \right.
\]

Actually, \( R_L \) affects all of the delays and a modestly better but modestly more complicated delay model is often used.
Elmore Delay Calculations

Example with resistive load (modestly better model):

Elmore delay:

\[ t_1 = \left( R_1 \parallel \left[ R_2 + R_3 + R_4 + R_5 \right] \right) C_1 \]

\[ t_2 = \left( \left[ R_1 + R_2 \right] \parallel \left[ R_3 + R_4 + R_5 \right] \right) C_2 \]

\[ t_3 = \left( \left[ R_1 + R_2 + R_3 \right] \parallel \left[ R_4 + R_5 \right] \right) C_2 \]

\[ t_4 = \left( \left( R_1 + R_2 + R_3 + R_4 \right) \parallel R_5 \right) C_2 \]

where

\[ t_{PD} = \sum_{i=1}^{4} t_i \]

\[ t_i = C_i \left( \sum_{j=1}^{i} R_j \right) \parallel \left[ \sum_{j=i+1}^{5} R_j \right] \quad j = 1, 2, 3, 4 \]
Elmore Delay Calculations

With resistive load (modestly better model):

\[ t_{PD} = \sum_{i=1}^{n} C_i \left( \sum_{j=1}^{i} R_j \right) \parallel \left[ \sum_{j=i+1}^{n+1} R_j \right] \]
Elmore Delay Calculations

How are the number of stages chosen?

- For hand analysis, keep number of stages small (maybe 3 or 4 for simple delay line) if possible.
- If “faithfulness” is important, should keep the number of stages per unit length constant.
End of Lecture 42