Administrative Stuff

- **PROJECT 1 (Start early!)**
  - You will not be able to finish in 1 week!

- **LABS**
  - This week: Finish any remaining SW labs
  - Next week: Project help in labs

- **HOMEWORKS**
  - Due date for Hw6 = Mon Oct 24 (EE142 Midterm)
  - No new homework this week
Important Changes

- **NEW OH**
  - Thursday, 1-3pm
  - Starting this week

- **BI-WEEKLY REVIEW**
  - One hour sessions every other Thursday
  - Starts this week (Time & Location TBD)

Schedule

- **Last lecture:**
  - CMOS logic gates

- **Today:**
  - Project launch
  - CMOS Design optimization
  - Logical effort
Switch Delay Model

**NAND-2**
- \( A \rightarrow B \)
- \( R_p \), \( R_n \), \( C_{int} \)
- \( C_L \)

**Inverter**
- \( A \rightarrow A' \)
- \( R_p \), \( C_{int} \)
- \( C_L \)

**NOR-2**
- \( A \rightarrow B \)
- \( R_p \), \( R_n \), \( C_{int} \)
- \( C_L \)

---

Input Pattern Effects on Delay

- **Delay is dependent on the pattern of inputs**
- **Low-to-high transition**
  - both inputs go low
    - delay is \( 0.69 \frac{R_p}{2} \) \( C_L \)
  - one input goes low
    - delay is \( 0.69 \) \( R_p \) \( C_L \)
- **High-to-low transition**
  - both inputs go high
    - delay is \( 0.69 \) \( 2R_n \) \( C_L \)
Delay Dependence on Input Patterns

<table>
<thead>
<tr>
<th>Input Data Pattern</th>
<th>Delay (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A=B=0→1</td>
<td>69</td>
</tr>
<tr>
<td>B=1, A=0→1</td>
<td>62</td>
</tr>
<tr>
<td>B= 0→1, A=1</td>
<td>50</td>
</tr>
<tr>
<td>A=B=1→0</td>
<td>35</td>
</tr>
<tr>
<td>B=1, A=1→0</td>
<td>76</td>
</tr>
<tr>
<td>B=1→0, A=1</td>
<td>57</td>
</tr>
</tbody>
</table>

NMOS = 0.5μm/0.25 μm
PMOS = 0.75μm/0.25 μm
C_L = 100 fF

Transistor Sizing

EE141 7

EE141 8
Sizing of a Complex CMOS Gate

\[ \text{OUT} = D + A \cdot (B + C) \]

Fan-In Considerations

\[ t_{\text{phL}} = 0.69 \ R_{\text{eqn}} (C_1 + 2C_2 + 3C_3 + 4C_L) \]

Propagation delay deteriorates rapidly as a function of fan-in – quadratically in the worst case.
$t_p$ as a Function of Fan-In

- Gates with fan-in > 4 should be avoided

$tp$ as a Function of Fan-Out

- All gates have the same drive current
- Slope is a function of “driving strength”
$t_p$ as a Function of Fan-In and Fan-Out

- Fan-in: quadratic due to increasing resistance and capacitance
- Fan-out: each additional fan-out gate adds two gate capacitances to $C_L$

$$t_p = a_1 FI + a_2 FI^2 + a_3 FO$$

Fast Complex Gates: Design Technique 1

- Progressive transistor sizing
  - as long as fan-out capacitance dominates

Distributed RC line

$M_1 > M_2 > M_3 > \ldots > M_N$

(the FET closest to the output is the smallest)

Can reduce delay by more than 20%;
Be careful: input loading, junction caps, decreasing gains as technology shrinks
Fast Complex Gates: Design Technique 2

**Transistor ordering**

- Critical path
  - In3 → M3 → C_L
  - In2 → M2 → C_2
  - In1 → M1 → C_1
  - delay determined by time to discharge C_L, C_1 and C_2

- Critical path
  - In1 → M1 → C_1
  - In2 → M2 → C_2
  - In3 → M3 → C_L
  - delay determined by time to discharge C_L

Fast Complex Gates: Design Technique 3

**Alternative logic structures**

\[ F = ABCDEFGH \]
Fast Complex Gates: Design Technique 4

- Isolating fan-in from fan-out using buffer insertion

\[ t_{pHL} = 0.69 \left( \frac{3}{4} \frac{C_L \ V_{DD}}{I_{DSATn}} \right) \]
\[ = 0.69 \left( \frac{3}{4} \frac{C_L \ V_{swing}}{I_{DSATn}} \right) \]

- linear reduction in delay
- also reduces power consumption

Fast Complex Gates: Design Technique 5

- Reducing the voltage swing

\[ t_{pHL} = 0.69 \left( \frac{3}{4} \frac{C_L \ V_{DD}}{I_{DSATn}} \right) \]
\[ = 0.69 \left( \frac{3}{4} \frac{C_L \ V_{swing}}{I_{DSATn}} \right) \]

- But the following gate is much slower!
  - can use of “sense amplifiers” on the receiving end to restore the signal level (memory design)
Logical Effort

Introduction

Chip designers face wide array of choices
- What is the best circuit topology for a function?
- How large should transistors be?
- How many stages of logic give least delay?

Logical Effort is a method to answer these Qs
- Uses simple delay model
- Back of the envelope calculations

Who cares about LE?
- Circuit designers who waste time in simulate/tweak loop
- High-speed logic designers need to know where time is going in their logic
- CAD designers need to understand circuits to build better tools

Courtesy: D. Harris, HMC
Logical Effort Formalism (1/4)

- Gate delay we used up to now:

\[
\text{Delay} = 0.69 \cdot R_{\text{drive}} \cdot C_{\text{intrinsic}} \cdot \left( 1 + \frac{C_{\text{out}}}{\gamma \cdot C_{\text{in}}} \right) = t_{p0} \cdot \left( 1 + \frac{C_{\text{out}}}{\gamma \cdot C_{\text{in}}} \right)
\]

- Another way to write this formula is:

\[
\text{Delay} = 0.69 \cdot R_{\text{drive}} \cdot C_{g_{\text{ gate}}} \cdot \left( \gamma + \frac{C_{\text{out}}}{C_{\text{in}}} \right) = \tau_{\text{gate}} \cdot \left( \gamma + \frac{C_{\text{out}}}{C_{\text{in}}} \right)
\]

Logical Effort Formalism (2/4)

- In this example, the total delay is:

\[
\text{Delay} = \tau_{\text{SAND}} \cdot \left( \gamma_{\text{SAND}} + \frac{C_{j+1}}{C_j} \right) + \tau_{\text{INV}} \cdot \left( \gamma_{\text{INV}} + \frac{C_{j+2}}{C_{j+1}} \right) + \tau_{\text{NOR}} \cdot \left( \gamma_{\text{NOR}} + \frac{C_{j+3}}{C_{j+2}} \right)
\]

- Normalized to the intrinsic time constant of INV:

\[
\frac{\text{Delay}}{\tau_{\text{INV}}} = \frac{\tau_{\text{SAND}}}{\tau_{\text{INV}}} \cdot \left( \gamma_{\text{SAND}} + \frac{C_{j+1}}{C_j} \right) + \frac{\tau_{\text{INV}}}{\tau_{\text{INV}}} \cdot \left( \gamma_{\text{INV}} + \frac{C_{j+2}}{C_{j+1}} \right) + \frac{\tau_{\text{NOR}}}{\tau_{\text{INV}}} \cdot \left( \gamma_{\text{NOR}} + \frac{C_{j+3}}{C_{j+2}} \right)
\]

Courtesy: B. Murmann, Stanford
Logical Effort Formalism (3/4)

Since it is hard to fit on the back of an envelope, we define new symbols:

\[
\frac{\text{Delay}}{\tau_{\text{INV}}} = \frac{\tau_{\text{NAND}}}{\tau_{\text{INV}}} \left( \frac{C_{j+1}}{C_j} + \gamma_{\text{NAND}} \right) + \frac{\tau_{\text{INV}}}{\tau_{\text{INV}}} \left( \frac{C_{j+2}}{C_{j+1}} + \gamma_{\text{INV}} \right) + \frac{\tau_{\text{NOR}}}{\tau_{\text{INV}}} \left( \frac{C_{j+3}}{C_{j+2}} + \gamma_{\text{NOR}} \right)
\]

\[D = (LE_{\text{NAND}} \cdot FO_j + P_{\text{NAND}}) + (LE_{\text{INV}} \cdot FO_{j+1} + P_{\text{INV}}) + (LE_{\text{NOR}} \cdot FO_{j+2} + P_{\text{NOR}})\]

Logical Effort \quad Parasitic Delay

"Electrical Effort"

Courtesy: B. Murmann, Stanford

Logical Effort Formalism (4/4)

More nomenclature:

- \( D_{\text{gate}} = LE \cdot FO + P = \text{Effort Delay} + \text{Parasitic Delay} \)

Some options to find LE of a logic gate:

- Set \( R_{\text{drive}} \) equal, then compare \( C_{\text{in}} \)
- Set \( C_{\text{in}} \) equal, then compare \( R_{\text{drive}} \)
- Or simply compare R and C ratio from first principles:

\[
LE_{\text{gate}} = \frac{\tau_{\text{gate}}}{\tau_{\text{INV}}} = \frac{(R_{\text{drive}} \cdot C_{\text{in}})_{\text{gate}}}{(R_{\text{drive}} \cdot C_{\text{in}})_{\text{INV}}}
\]

Courtesy: B. Murmann, Stanford
Calculating Logical Effort

**DEF:** Logical effort is the ratio of the input capacitance to the input capacitance of an inverter delivering the same output current.

**LE Catalog of Gates**

<table>
<thead>
<tr>
<th>Gate type</th>
<th>Number of inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>inverter</td>
<td>1</td>
</tr>
<tr>
<td>NAND</td>
<td>4/3</td>
</tr>
<tr>
<td>NOR</td>
<td>5/3</td>
</tr>
<tr>
<td>multiplexer</td>
<td>2</td>
</tr>
<tr>
<td>Muller C</td>
<td>2</td>
</tr>
<tr>
<td>xor (parity)</td>
<td>4</td>
</tr>
<tr>
<td>majority</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gate type</th>
<th>Parasitic delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>inverter</td>
<td>$P_{in}$</td>
</tr>
<tr>
<td>n-input NAND</td>
<td>$P_{n NAND}$</td>
</tr>
<tr>
<td>n-input NOR</td>
<td>$P_{n NOR}$</td>
</tr>
<tr>
<td>n-way multiplexer</td>
<td>$2P_{n multiplexer}$</td>
</tr>
<tr>
<td>n-input Muller C</td>
<td>$2P_{n Muller C}$</td>
</tr>
<tr>
<td>xor, xnor</td>
<td>$4P_{xor}$</td>
</tr>
<tr>
<td>3-input majority</td>
<td>$6P_{3 majority}$</td>
</tr>
</tbody>
</table>

Source: “Logical Effort,” I. Sutherland, B. Sproull, D. Harris (Morgan-Kaufmann 1999)
**Warm-up Example 1**

Estimate the frequency of an $N$-stage ring oscillator:

Logical Effort: \[ LE = 1 \]
Electrical Effort: \[ FO = \frac{C_{out}}{C_{in}} = 1 \]
Parasitic Delay: \[ P = p_{inv} = 1 \]
Stage Delay: \[ D = LE \cdot FO + P = 2 \]
OSC Frequency: \[ F_{osc} = \frac{1}{2ND} = \frac{1}{4N\tau} \]
Warm-up Example 2

Estimate the delay of a fanout-of-4 (FO4) inverter:

Logical Effort: \( \text{LE} = 1 \)

Electrical Effort: \( \text{FO} = \frac{C_{\text{out}}}{C_{\text{in}}} = 4 \)

Parasitic Delay: \( P = p_{\text{inv}} = 1 \)

Stage Delay: \( D = \text{LE} \cdot \text{FO} + P = 5 \)

---

Multistage Networks

\[ \text{Delay} = \sum_{i=1}^{N} \left( P_i + \text{LE}_i \cdot \text{FO}_i \right) \]

Stage effort: \( \text{SE}_i = \text{LE}_i \cdot \text{FO}_i \)

Path electrical effort: \( \text{FO}_{\text{path}} = \frac{C_{\text{out}}}{C_{\text{in}}} \)

Path logical effort: \( \text{LE}_{\text{path}} = \text{LE}_1 \cdot \text{LE}_2 \cdots \text{LE}_N \)

Branching effort: \( B_{\text{path}} = b_1 b_2 \cdots b_N \)

Path effort: \( \text{LE}_{\text{path}} \cdot \text{FO}_{\text{path}} \cdot B_{\text{path}} \)

Path delay \( D = \sum D_i = \sum P_i + \sum \text{FO}_i \cdot \text{LE}_i \)
Optimum Effort per Stage

When each stage bears the same effort:

\[ SE^N = \prod LE \cdot FO = \text{PathEffort} \]

\[ SE^* = \sqrt[N]{\text{PathEffort}} \]

Effective fanout of each stage:

\[ FO_i = \frac{SE^*}{LE_i} \]

Minimum path delay

\[ D_{\text{min}} = \sum (LE_i \cdot FO_i + P_i) = N \cdot SE^* + P \]

Gate Sizing Problem

- We will use the LE of the gate to help find the correct sizes
  - We know that the LE·FO for each gate should be the same

- Before we do the math, we need to set a convention:
  - What does a gate size of “2” mean?
  - For an inverter, it is simple
    - It has twice the C and \( \frac{1}{2} \) the R of an inverter of size “1”
  - For a gate, you have two options:
    - Can define it to mean it has twice the C of an inverter OR
    - Can define it to mean it has \( \frac{1}{2} \) the R

- We assume the size is a measure of the input capacitance
  - A size 2 gate has twice the \( C_{\text{in}} \) of the inverter

Courtesy: B. Murmann, Stanford
Gate Sizing Example (1/3)

First Compute Path Effort

\[
\text{PathEffort} = \prod LE \cdot \text{FO}
\]

\[
= \left( \frac{x}{10} \right) \times \left( \frac{y}{3} \right) \times \left( \frac{z}{y} \right) \times \left( \frac{20}{z} \right) = \frac{400}{9}
\]

The optimal stage effort is:

\[
SE^* = LE \cdot \text{FO} = \left( \frac{400}{9} \right)^{1/4} = 1.45
\]

From: David Harris

Gate Sizing Example (2/3)

We can now size the gates, since for all of them:

\[
C_{in} = LE \cdot \frac{C_{out}}{SE^*}
\]

We have:

\[
x = \frac{5}{3} \cdot \frac{1.45}{1.45} = 14.5
\]

\[
y = \frac{4}{3} \cdot \frac{z}{1.45} = \frac{20}{1.45} = 13.8
\]

\[
y = \frac{4}{3} \cdot \frac{20}{1.45} = 12.7
\]

\[
x = \frac{5}{3} \cdot \frac{1.45}{1.45} = 10
\]

\[
x = \frac{5}{3} \cdot \frac{1.45}{1.45} = 10
\]
Gate Sizing Example (2/3)

\[ D = 4SE^* + \sum P = 4 \cdot 1.45 + (1 + 2 + 2 + 1) = 11.8 \]

Add Branching Effort

Branching effort:

\[ b = \frac{C_{\text{on-path}} + C_{\text{off-path}}}{C_{\text{on-path}}} \]
Branching Example 1

\[
\begin{align*}
LE &= 1 \\
FO &= 90/5 = 18 \\
PE &= 18 \text{ (wrong!)} \\
SE_1 &= (15+15)/5 = 6 \\
SE_2 &= 90/15 = 6 \\
PE &= 36, \text{ not } 18!
\end{align*}
\]

Introduce new kind of effort to account for branching:

- **Branching Effort**: 
  \[
  b = \frac{C_{\text{on-path}} + C_{\text{off-path}}}{C_{\text{on-path}}}
  \]

- **Path Branching Effort**: 
  \[
  B = \prod b_i
  \]

Now we can compute the path effort:

- **Path Effort**: 
  \[
  PE = \prod LE \cdot FO \cdot B
  \]

Branching Example 2

Select gate sizes \(y\) and \(z\) to minimize delay from \(A\) to \(B\)

- **Logical Effort**: 
  \(LE = (4/3)^3\)

- **Electrical Effort**: 
  \(FO = C_{\text{out}}/C_{\text{in}} = 9\)

- **Branching Effort**: 
  \(B = 2 \cdot 3 = 6\)

- **Path Effort**: 
  \(PE = \prod LE \cdot FO \cdot B = 128\)

- **Best Stage Effort**: 
  \(SE = PE^{1/3} \approx 5\)

- **Delay**: 
  \(D = 3 \cdot 5 + 3 \cdot 2 = 21\)

Work backward for sizes:

- \(z = \frac{9C \cdot (4/3)}{5} = 2.4C\)
- \(y = \frac{3z \cdot (4/3)}{5} = 1.9C\)
Multi-level Logic: What is Best?

\[
LE = \frac{10}{3} \\
(3.33)
\]

\[
LE = \frac{10}{3} \\
(3.33)
\]

\[
LE = \frac{80}{27} \\
(2.96)
\]

Handling Wires & Fixed Loads

\[
Delay = \sum_{i=1}^{N} \left( P_i + LE_i \cdot (FO_i + W_i) \right)
\]
Logical Effort “Design Flow”

- Compute the path effort: \( \text{Path Effort} = \prod LE \cdot FO \cdot B \)

- Find the best number of stages: \( N^* \sim \log_4(\text{PathEffort}) \)

- Compute the stage effort: \( SE^* = (\text{PathEffort})^{1/N} \)

- Working from either end, determine gate sizes:

\[
C_{in} = LE \cdot B \cdot \frac{C_{out}}{SE^*}
\]

Reference: Sutherland, Sproull, Harris, “Logical Effort,” (Morgan-Kaufmann 1999)

Next Lecture

- Ratioed Logic

- Pass-Transistor Logic