EEC 118 Lecture #7: Designing with Logical Effort

Rajeevan Amirtharajah
University of California, Davis
Jeff Parkhurst
Intel Corporation
Announcements

• Lab 3 this week at lab section
• HW 3 due this Friday at 4 PM in box, Kemper 2131
• Quizzes will be handed back in lab section
Outline

- Review: CMOS Combinational Gate Design
- Finish Lecture 6 slides
- Logical Effort
- Combinational MOS Logic Circuits: Rabaey 6.1-6.2 (Kang & Leblebici, 7.1-7.4)
Acknowledgments

• Slides due to David Money Harris from E158: Introduction to CMOS VLSI Design at Harvey Mudd College
Review: Static CMOS

- Complementary pullup network (PUN) and pulldown network (PDN)
- Only one network is on at a time
- PUN: PMOS devices
  - Why? Pulls up to VDD.
- PDN: NMOS devices
  - Why? Pulls down to ground.
- PUN and PDN are *dual* networks
Review: Dual Networks

- Dual networks: parallel connection in PDN = series connection in PUN, vice-versa

- If CMOS gate implements logic function $F$:
  - PUN implements function $F$
  - PDN implements function $\overline{F} = F$

Example: NAND gate
Lecture 6: Logical Effort
Outline

- Logical Effort
- Delay in a Logic Gate
- Multistage Logic Networks
- Choosing the Best Number of Stages
- Example
- Summary
Introduction

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

- Logical effort is a method to make these decisions
  - Uses a simple model of delay
  - Allows back-of-the-envelope calculations
  - Helps make rapid comparisons between alternatives
  - Emphasizes remarkable symmetries
Example

- Ben Bitdiddle is the memory designer for the Motoroil 68W86, an embedded automotive processor. Help Ben design the decoder for a register file.

- Decoder specifications:
  - 16 word register file
  - Each word is 32 bits wide
  - Each bit presents load of 3 unit-sized transistors
  - True and complementary address inputs A[3:0]
  - Each input may drive 10 unit-sized transistors

- Ben needs to decide:
  - How many stages to use?
  - How large should each gate be?
  - How fast can decoder operate?
Delay in a Logic Gate

- Express delays in process-independent unit
- Delay has two components: \( d = f + p \)
  - \( f \): effort delay = \( gh \) (a.k.a. stage effort)
    - Again has two components
  - \( g \): logical effort
    - Measures relative ability of gate to deliver current
    - \( g \equiv 1 \) for inverter
  - \( h \): electrical effort = \( C_{out} / C_{in} \)
    - Ratio of output to input capacitance
    - Sometimes called fanout
  - \( p \): parasitic (intrinsic) delay
    - Represents delay of gate driving no load
    - Set by internal parasitic capacitance

\[ d = \frac{d_{abs}}{\tau} \]
\[ \tau = 3RC \approx 3 \text{ ps in 65 nm process} \]
\[ 60 \text{ ps in 0.6 \( \mu \)m process} \]
$d = f + p$

$= gh + p$
Computing Logical Effort

- **DEF:** Logical effort is the ratio of the input capacitance of a gate to the input capacitance of an inverter delivering the same output current.
- Measure from delay vs. fanout plots
- Or estimate by counting transistor widths

![Logic Diagrams]

- $C_{in} = 3, \quad g = 3/3$
- $C_{in} = 4, \quad g = 4/3$
- $C_{in} = 5, \quad g = 5/3$
## Catalog of Gates

- Logical effort of common 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>Tristate / mux</td>
<td>2</td>
</tr>
<tr>
<td>XOR, XNOR</td>
<td>4, 4</td>
</tr>
</tbody>
</table>
Catalog of Gates

- Parasitic delay of common gates
  - In multiples of $p_{\text{inv}} \approx 1$

<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>2</td>
</tr>
<tr>
<td>NOR</td>
<td>2</td>
</tr>
<tr>
<td>Tristate / mux</td>
<td>2</td>
</tr>
<tr>
<td>XOR, XNOR</td>
<td>4</td>
</tr>
</tbody>
</table>
**Example: Ring Oscillator**

- Estimate the frequency of an N-stage ring oscillator

![Ring Oscillator Diagram]

- Logical Effort: \( g = \)
- Electrical Effort: \( h = \)
- Parasitic Delay: \( p = \)
- Stage Delay: \( d = \)
- Frequency: \( f_{osc} = \)

31 stage ring oscillator in 0.6 μm process has frequency of ~ 200 MHz
Example: FO4 Inverter

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

Logical Effort: \( g = \)

Electrical Effort: \( h = \)

Parasitic Delay: \( p = \)

Stage Delay: \( d = \)

The FO4 delay is about 300 ps in a 0.6 \( \mu \)m process and 15 ps in a 65 nm process.
Multistage Logic Networks

- Logical effort generalizes to multistage networks
- **Path Logical Effort**
  \[ G = \prod g_i \]
- **Path Electrical Effort**
  \[ H = \frac{C_{\text{out-path}}}{C_{\text{in-path}}} \]
- **Path Effort**
  \[ F = \prod f_i = \prod g_i h_i \]

\[ \begin{align*}
g_1 &= 1 \\
h_1 &= x/10 \\
g_2 &= 5/3 \\
h_2 &= y/x \\
g_3 &= 4/3 \\
h_3 &= z/y \\
g_4 &= 1 \\
h_4 &= 20/z \\
10 &\quad x &\quad y &\quad z &\quad 20
\end{align*} \]
Multistage Logic Networks

- Logical effort generalizes to multistage networks
- **Path Logical Effort** \( G = \prod g_i \)
- **Path Electrical Effort** \( H = \frac{C_{out\text{-path}}}{C_{in\text{-path}}} \)
- **Path Effort** \( F = \prod f_i = \prod g_i h_i \)

- Can we write \( F = GH? \)
Paths that Branch

- No! Consider paths that branch:

\[
\begin{align*}
G &= \frac{15 + 15}{5} = 6 \\
H &= \frac{90}{15} = 6 \\
GH &= 18h \\
h_1 &= \frac{90}{15} = 6 \\
h_2 &= \frac{90}{15} = 6 \\
F &= gh_1h_2 = 36 \neq 2GH \\
\end{align*}
\]
Introduce branching effort
- Accounts for branching between stages in path

\[ b = \frac{C_{on\ path} + C_{off\ path}}{C_{on\ path}} \]

\[ B = \prod b_i \]

Note:
\[ \prod h_i = BH \]

Now we compute the path effort
- \( F = GBH \)
Multistage Delays

- Path Effort Delay
  \[ D_F = \sum f_i \]

- Path Parasitic Delay
  \[ P = \sum p_i \]

- Path Delay
  \[ D = \sum d_i = D_F + P \]
Designing Fast Circuits

\[ D = \sum d_i = D_F + P \]

- Delay is smallest when each stage bears same effort

\[ \hat{f} = g_i h_i = F^{1/N} \]

- Thus minimum delay of N stage path is

- This is a key result of logical effort
  - Find fastest possible delay
  - Doesn’t require calculating gate sizes
Gate Sizes

- How wide should the gates be for least delay?

\[ \hat{f} = gh = g \frac{C_{\text{out}}}{C_{\text{in}}} \]

\[ \Rightarrow C_{\text{in}_i} = \frac{g_i C_{\text{out}_i}}{\hat{f}} \]

- Working backward, apply capacitance transformation to find input capacitance of each gate given load it drives.

- Check work by verifying input cap spec is met.
Example: 3-stage path

- Select gate sizes $x$ and $y$ for least delay from A to B
Example: 3-stage path

Logical Effort \[ G = \]
Electrical Effort \[ H = \]
Branching Effort \[ B = \]
Path Effort \[ F = \]
Best Stage Effort \[ f = \]
Parasitic Delay \[ P = \]
Delay \[ D = \]
Example: 3-stage path

- Work backward for sizes

\[ y = 45 \times \left( \frac{5}{3} \right) / 5 = 15 \]
\[ x = (15 \times 2) \times \left( \frac{5}{3} \right) / 5 = 10 \]
Best Number of Stages

- How many stages should a path use?
  - Minimizing number of stages is not always fastest
- Example: drive 64-bit datapath with unit inverter

\[ D = \frac{N}{f} + P \]

<table>
<thead>
<tr>
<th>N</th>
<th>f</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>64</td>
<td>65</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>18</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>2.8</td>
<td>15.3</td>
</tr>
</tbody>
</table>

Initial Driver

```
1  
8  
4  
2.8
```

Datapath Load

```
64  
64  
64  
64
```
Consider adding inverters to end of path

- How many give least delay?

\[ D = N F \frac{1}{N} + \sum_{i=1}^{n_1} p_i + \left( N - n_1 \right) p_{\text{inv}} \]

\[ \frac{\partial D}{\partial N} = -F \frac{1}{N} \ln F \frac{1}{N} + F \frac{1}{N} + p_{\text{inv}} = 0 \]

Define best stage effort \( \rho = F \frac{1}{N} \)

\[ p_{\text{inv}} + \rho \left( 1 - \ln \rho \right) = 0 \]
Best Stage Effort

- $p_{\text{inv}} + \rho \left(1 - \ln \rho \right) = 0$ has no closed-form solution

- Neglecting parasitics ($p_{\text{inv}} = 0$), we find $\rho = 2.718 \, (e)$
- For $p_{\text{inv}} = 1$, solve numerically for $\rho = 3.59$
Sensitivity Analysis

How sensitive is delay to using exactly the best number of stages?

- $2.4 < \rho < 6$ gives delay within 15% of optimal
  - We can be sloppy!
  - I like $\rho = 4$
Example, Revisited

- Ben Bitdiddle is the memory designer for the Motoroil 68W86, an embedded automotive processor. Help Ben design the decoder for a register file.

- Decoder specifications:
  - 16 word register file
  - Each word is 32 bits wide
  - Each bit presents load of 3 unit-sized transistors
  - True and complementary address inputs \( A[3:0] \)
  - Each input may drive 10 unit-sized transistors

- Ben needs to decide:
  - How many stages to use?
  - How large should each gate be?
  - How fast can decoder operate?
Number of Stages

- Decoder effort is mainly electrical and branching
  - Electrical Effort: \( H = \frac{32 \times 3}{10} = 9.6 \)
  - Branching Effort: \( B = 8 \)

- If we neglect logical effort (assume \( G = 1 \))
  - Path Effort: \( F = \)

  Number of Stages: \( N = \)

- Try a \( \_\_\_\_\_ \)-stage design
Gate Sizes & Delay

Logical Effort: \( G = \)
Path Effort: \( F = \)
Stage Effort: \( \hat{f} = \)
Path Delay: \( D = \)
Gate sizes: \( z = \) \( y = \)

96 units of wordline capacitance
Comparison

- Compare many alternatives with a spreadsheet
- \( D = N (76.8 \ G)^{1/N} + P \)

<table>
<thead>
<tr>
<th>Design</th>
<th>N</th>
<th>G</th>
<th>P</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOR4</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>234</td>
</tr>
<tr>
<td>NAND4-INV</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>29.8</td>
</tr>
<tr>
<td>NAND2-NOR2</td>
<td>2</td>
<td>20/9</td>
<td>4</td>
<td>30.1</td>
</tr>
<tr>
<td>INV-NAND4-INV</td>
<td>3</td>
<td>2</td>
<td>6</td>
<td>22.1</td>
</tr>
<tr>
<td>NAND4-INV-INV-INV</td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>21.1</td>
</tr>
<tr>
<td>NAND2-NOR2-INV-INV-INV</td>
<td>4</td>
<td>20/9</td>
<td>6</td>
<td>20.5</td>
</tr>
<tr>
<td>NAND2-INN-V-NAND2-INN</td>
<td>4</td>
<td>16/9</td>
<td>6</td>
<td>19.7</td>
</tr>
<tr>
<td>INV-NAND2-INN-NAND2-INN</td>
<td>5</td>
<td>16/9</td>
<td>7</td>
<td>20.4</td>
</tr>
<tr>
<td>NAND2-INN-V-NAND2-INN-INN</td>
<td>6</td>
<td>16/9</td>
<td>8</td>
<td>21.6</td>
</tr>
</tbody>
</table>
# Review of Definitions

<table>
<thead>
<tr>
<th>Term</th>
<th>Stage</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>number of stages</td>
<td>1</td>
<td>$N$</td>
</tr>
<tr>
<td>logical effort</td>
<td>$g$</td>
<td>$G = \prod g_i$</td>
</tr>
<tr>
<td>electrical effort</td>
<td>$h = \frac{C_{\text{out}}}{C_{\text{in}}}$</td>
<td>$H = \frac{C_{\text{out-path}}}{C_{\text{in-path}}}$</td>
</tr>
<tr>
<td>branching effort</td>
<td>$b = \frac{C_{\text{on-path}}+C_{\text{off-path}}}{C_{\text{on-path}}}$</td>
<td>$B = \prod b_i$</td>
</tr>
<tr>
<td>effort</td>
<td>$f = gh$</td>
<td>$F = GBH$</td>
</tr>
<tr>
<td>effort delay</td>
<td>$f$</td>
<td>$D_F = \sum f_i$</td>
</tr>
<tr>
<td>parasitic delay</td>
<td>$p$</td>
<td>$P = \sum p_i$</td>
</tr>
<tr>
<td>delay</td>
<td>$d = f + p$</td>
<td>$D = \sum d_i = D_F + P$</td>
</tr>
</tbody>
</table>
**Method of Logical Effort**

1) Compute path effort  \( F = GBH \)
2) Estimate best number of stages  \( N = \log_4 F \)
3) Sketch path with \( N \) stages
4) Estimate least delay  \( D = NF^{\frac{1}{N}} + P \)
5) Determine best stage effort  \( \hat{f} = F^{\frac{1}{N}} \)
6) Find gate sizes  

\[
C_{in_i} = \frac{g_i C_{out_i}}{\hat{f}}
\]
Limits of Logical Effort

- Chicken and egg problem
  - Need path to compute G
  - But don’t know number of stages without G
- Simplistic delay model
  - Neglects input rise time effects
- Interconnect
  - Iteration required in designs with wire
- Maximum speed only
  - Not minimum area/power for constrained delay
Logical effort is useful for thinking of delay in circuits
- Numeric logical effort characterizes gates
- NANDs are faster than NORs in CMOS
- Paths are fastest when effort delays are ~4
- Path delay is weakly sensitive to stages, sizes
- But using fewer stages doesn’t mean faster paths
- Delay of path is about $\log_4 F$ FO4 inverter delays
- Inverters and NAND2 best for driving large caps

Provides language for discussing fast circuits
- But requires practice to master
Next Topic: Sequential Logic

• Basic sequential circuits in CMOS
  – RS latches, transparent latches, flip-flops
  – Alternative sequential element topologies
  – Pipelining