6. Logical Effort

Jacob Abraham

Department of Electrical and Computer Engineering
The University of Texas at Austin

VLSI Design
Fall 2020

September 15, 2020

Review: See Additional Notes Posted

Calculate the Elmore delay from C to F in the circuit. The widths of the pass transistors are shown, and the inverters have minimum-sized.

Use the Elmore delay approximation to find the worst-case rise and fall delays at output F for the following circuit. The gate sizes of the transistors are shown in the figure. Assume NO sharing of diffusion regions, and the worst-case conditions for the initial charge on a node.
Example: Delay with Different Input Sequences

Find the delays for the given input transitions (gate sizes shown in figure)

Assumptions: diffusion capacitance is equal to the gate capacitance, the resistance of an nMOS transistor with unit width is $R$ and the resistance of a pMOS transistor with width 2 is also $R$, and NO sharing of diffusion regions

Off-path capacitances can contribute to delay, and if a node does not need to be charged (or discharged), its capacitance can be ignored

$$ABCD = 0101 \rightarrow ABCD = 1101$$
$$ABCD = 1111 \rightarrow ABCD = 0111$$
$$ABCD = 1010 \rightarrow ABCD = 1101$$

Delay with Different Input Sequence, Cont’d

Look at the charges on the nodes at the end of the first input of the sequence; only the capacitances of the nodes which would change with the second vector need to be considered

$$ABCD = 0101 \rightarrow$$
$$ABCD = 1101;$$
Delay = 36RC

$$ABCD = 1111 \rightarrow$$
$$ABCD = 0111;$$
Delay = 16RC

$$ABCD = 1010 \rightarrow$$
$$ABCD = 1101;$$
Delay = 43RC
Delay Components

Delay has two parts

Parasitic Delay
- 6 or 7 RC
- Independent of Load

Effort Delay
- 4h RC
- Proportional to load capacitance

Contamination Delay

Minimum (Contamination) Delay
- Best-case (contamination) delay can be substantially less than propagation delay
- Example: If both inputs fall simultaneously
- Important for "hold time" (will see later in the course)

\[ t_{cdr} = (3 + 2h)RC \]
Introduction to Logical Effort

Chip designers have to 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 one 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
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

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

Express delay in a process-independent unit

\[ d = \frac{d_{\text{abs}}}{\tau} \]

\[ \tau = 3RC \]

\[ \approx 12 \text{ ps in 180 nm process} \]
\[ 40 \text{ ps in 0.6 \ } \mu\text{m process} \]

Delay has two components: \( d = f + p \)

Effort delay, \( f = gh \) (stage effort)

\( g \): Logical Effort
Measures relative ability of gate to deliver current
\( g \equiv 1 \) for inverter

\( h \): Electrical Effort
Ratio of output to input capacities, sometimes called fanout effort

Parasitic delay, \( p \)
- Represents delay of gate driving no load
- Set by internal parasitic capacitance

Delay Plots

\[ d = f + p \]

\[ = gh + p \]

What about NOR2?
Computing Logical Effort

- 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]

<table>
<thead>
<tr>
<th>Gate type</th>
<th>Number of inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inverter</td>
<td>1</td>
</tr>
<tr>
<td>NAND</td>
<td>4/3 5/3 6/3 (n+2)/3</td>
</tr>
<tr>
<td>NOR</td>
<td>5/3 7/3 9/3 (2n+1)/3</td>
</tr>
<tr>
<td>Tristate/Mux</td>
<td>2 2 2 2</td>
</tr>
<tr>
<td>XOR, XNOR</td>
<td>4,4 6,12,6 8,16,16,8</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>Inverter</td>
<td>1</td>
</tr>
<tr>
<td>NAND</td>
<td>2 3 4 n</td>
</tr>
<tr>
<td>NOR</td>
<td>2 3 4 n</td>
</tr>
<tr>
<td>Tristate/Mux</td>
<td>2 4 6 8 2n</td>
</tr>
<tr>
<td>XOR, XNOR</td>
<td>4 6 8</td>
</tr>
</tbody>
</table>

Example: Ring Oscillator

Estimate the frequency of an N-stage ring oscillator

Logical Effort: $g = 1$
Electrical Effort: $h = 1$
Parasitic Delay: $p = 1$
Stage Delay: $d = 2$
Frequency: $f_{\text{osc}} = 1/(2 \cdot N \cdot d) = 1/4N$

31 stage ring oscillator in 0.6 $\mu$m process has frequency of $\sim 200$ MHz
**Example: FO4 Inverter**

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

Logical Effort: \( g = 1 \)
Electrical Effort: \( h = 4 \)
Parasitic Delay: \( p = 1 \)
Stage Delay: \( d = 5 \)

The FO4 delay is about:
- 200 ps in a 0.6\( \mu \)m process
- 60 ps in a 180 nm process
- \( f/3 \) ns in a \( f \) \( \mu \)m process
- \( f/3 \) ps in a \( f \) nm process

**Example Problem**

A particular technology node has a FO4 delay of 9 ps. How many minimum size (2:1) inverters need to be included in a ring oscillator so that the frequency is close to 7.3 GHz?

FO4 delay = 15\( RC \) = 9ps
Stage delay = 6\( RC \) = 3.6ps

\[
f = \frac{1}{2 \times N \times d} \implies \]

\[
N = \frac{1}{2 \times f \times d} = \frac{1}{2 \times 7.3 \times 3.6 \times 10^{-3}} = 19.05
\]

Number of inverters = 19
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, \( F = \prod f_i = \prod g_i h_i \)

Consider Paths that Branch

\( G = 1 \)
\( H = \frac{90}{5} = 18 \)
\( GH = 18 \)
\( h_1 = \frac{15 + 15}{5} = 6 \)
\( h_2 = \frac{90}{15} = 6 \)
\( F = g_1 g_2 h_1 h_2 = 36 = 2GH \)

Can we write \( F = GH \) in general?
Branching Effort and Multistage Delays

**Branching Effort** accounts for branching between stages in path

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

\[ B = \prod b_i \quad \text{(Note: } \prod h_i = BH \text{)} \]

Now, 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

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

Thus, the minimum delay of an \( N \)-stage path is

\[ D = NF^\frac{1}{N} + P \]

- 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 the least delay?

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

\[ \Rightarrow C_{in_i} = \frac{g_i C_{out_i}}{\hat{f}} \]

- Working backward, apply capacitance transformation to find input capacitance of each gate given load it drives
- Check work by verifying input capacitance specification is met

Example: 3-stage Path

Select gate sizes \( x \) and \( y \) for least delay from \( A \) to \( B \)
Example: 3-stage Path, Cont’d

Logical Effort \[ G = \left(\frac{4}{3}\right)\left(\frac{5}{3}\right)\left(\frac{5}{3}\right) = \frac{100}{27} \]
Electrical Effort \[ H = \frac{45}{8} \]
Branching Effort \[ B = 3 \times 2 = 6 \]
Path Effort \[ F = GBH = 125 \]
Best Stage Effort \[ \hat{f} = \sqrt[3]{F} = 5 \]
Parasitic Delay \[ P = 2 + 3 + 2 = 7 \]
Delay \[ D = 3 \times 5 + 7 = 22 = 4.4 \text{ FO4} \]

Work backward for sizes
\[ y = 45 \times \frac{5}{3} / 5 = 15 \]
\[ x = (15 \times 2) \times \frac{5}{3} / 5 = 10 \]
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

\[
D = NF^{\frac{1}{N}} + P
\]
\[
D = N(64)^{\frac{1}{N}} + N
\]

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
- For example, use \(\rho = 4\)
**Decoder Example: Number of Stages**

- 16 word, (32 bit) register file
- 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

**Find: number of stages, sizes of gates, speed**

- Decoder effort is mainly electrical and branching
  - Electrical Effort: H = (32*3)/10 = 9.6
  - Branching Effort: B = 8
- If we neglect logical effort (assume G = 1)
  - Path Effort: F = GBH = 76.8
- Number of Stages: N = log₄F = 3.1
  - Try a 3-stage design

---

**Decoder: Gate Sizes and Delay**

- Logical Effort: G = 1 * 6/3 * 1 = 2
- Path Effort: F = GBH = 154
- Stage Effort:  \( \hat{f} = F^{3/2} = 5.36 \)
- Path Delay: \( D = 3\hat{f} + 1 + 4 + 1 = 22.1 \)
- Gate sizes: \( z = 96*1/5.36 = 18 \)
- Gate sizes: \( y = 18*2/5.36 = 6.7 \)
Decoder: Comparison

Compare many alternatives with a spreadsheet

<table>
<thead>
<tr>
<th>Design</th>
<th>N</th>
<th>G</th>
<th>P</th>
<th>D</th>
</tr>
</thead>
<tbody>
<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-NAND2-NAND2-INV</td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>21.1</td>
</tr>
<tr>
<td>NAND2-NOR2-INV-INV</td>
<td>4</td>
<td>20/9</td>
<td>6</td>
<td>20.5</td>
</tr>
<tr>
<td>INV-NAND2-INV-NAND2-INV-INV</td>
<td>5</td>
<td>16/9</td>
<td>7</td>
<td>20.4</td>
</tr>
<tr>
<td>NAND2-INV-NAND2-INV-INV-INV</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>$h = \frac{C_{out}}{C_{in}}$</td>
<td>$H = \frac{C_{out-path}}{C_{in-path}}$</td>
</tr>
<tr>
<td>Branching effort</td>
<td>$b = \frac{C_{on-path} + C_{off-path}}{C_{on-path}}$</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 \( C_{in} = \frac{g_i C_{out}}{f} \)

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 significant wires
- Maximum speed only
- Not minimum area/power for constrained delay

Summary of Logical Effort

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 \( \sim 4 \)
- 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
Decoder: Comparison

Compare many alternatives with a spreadsheet

<table>
<thead>
<tr>
<th>Design</th>
<th>N</th>
<th>G</th>
<th>P</th>
<th>D</th>
</tr>
</thead>
<tbody>
<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-INV-NAND2-INV</td>
<td>4</td>
<td>16/9</td>
<td>6</td>
<td>19.7</td>
</tr>
<tr>
<td>INV-NAND2-INV-NAND2-INV</td>
<td>5</td>
<td>16/9</td>
<td>7</td>
<td>20.4</td>
</tr>
<tr>
<td>NAND2-INV-NAND2-INV-INV-INV</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_{out}}{C_{in}} )</td>
<td>H = ( \frac{C_{out-path}}{C_{in-path}} )</td>
</tr>
<tr>
<td>Branching effort</td>
<td>b = ( \frac{C_{on-path} + C_{off-path}}{C_{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>

Department of Electrical and Computer Engineering, The University of Texas at Austin
J. A. Abraham, September 15, 2020
**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 \( C_{in} = g \cdot C_{out} \)

**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 significant wires
- Maximum speed only
  - Not minimum area/power for constrained delay

**Summary of Logical Effort**

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 \( \sim 4 \)
- 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