REVIEW: 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: \( d = 5 \)

Logical Effort:

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

REVIEW: 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
  \[ F = \prod f_i = \prod g_i h_i \]

REVIEW: Branching Effort

- Introduce branching effort
  - Accounts for branching between stages in path
  \[ b = \frac{C_{\text{on-path}} + C_{\text{off-path}}}{C_{\text{on-path}}} \]
  \[ B = \prod b_i \]
  Note: \( \prod b_i = BH \)
- Now we compute the path effort
  - \( F = GBH \)

REVIEW: Multistage Delays

- Path Effort Delay
  \[ D_p = \sum l_i \]
- Path Parasitic Delay
  \[ P = \sum p_i \]
- Path Delay
  \[ D = \sum d_i = D_p + P \]
REVIEW: Designing Fast Circuits

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

- Delay is smallest when each stage bears same effort
  \[ \hat{f} = \hat{g} h_i = F_F \]
- Thus minimum delay of N stage path is
  \[ D = NF_F + P \]
- This is a key result of logical effort
  - Find fastest possible delay
  - Doesn’t require calculating gate sizes

REVIEW: Gate Sizes

- How wide should the gates be for least delay?
  \[ \hat{f} = gh = g \frac{C}{C_{out}} \]
  \[ \Rightarrow C_{in} = g \frac{C_{out}}{\hat{f}} \]
- 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.

REVIEW: 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

\[
\begin{array}{c|c|c|c|c|c}
\text{Initial Driver} & \text{Datapath Load} & \text{N} & \text{f} & \text{D} \\
\hline
\text{1} & \text{64} & 1234 & 65 & 234 \\
\text{2} & \text{8} & 18 & 15 & 15.3 \\
\text{3} & \text{4} & 15 & 7.8 & 2.8 \\
\text{4} & \text{2.8} & 3 & 7.8 & 2.8 \\
\end{array}
\]

Best Stage Effort

- has no closed-form solution
  \[ p_{inv} + \rho (1 - \ln \rho) = 0 \]
- Neglecting parasitics (\( p_{inv} = 0 \)), we find \( \rho = 2.718 \) (e)
- For \( p_{inv} = 1 \), 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: $F = GBH = 76.8$

  Number of Stages: $N = \log_4 F = 3.1$

  • Try a 3-stage design

Gate Sizes & Delay

Logical Effort: $G = 1$
Path Effort: $F = GBH = 154$
Stage Effort: $\hat{f} = 5.36$
Path Delay: $D = 22.1$
Gate sizes: $z = \frac{96 \times 15.36}{5} = 18$

$\hat{y} = \frac{18 \times 25.36}{6.7}$
### 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>2</td>
<td>4</td>
<td>30.1</td>
</tr>
<tr>
<td>INV/NAND4-INV</td>
<td>2</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</td>
<td>4</td>
<td>2</td>
<td>9</td>
<td>20.5</td>
</tr>
<tr>
<td>INV/NAND2-NAND2-INV</td>
<td>4</td>
<td>16/9</td>
<td>6</td>
<td>19.7</td>
</tr>
<tr>
<td>INV-NAND2-NAND2-INV</td>
<td>5</td>
<td>16/9</td>
<td>7</td>
<td>20.4</td>
</tr>
<tr>
<td>NAND2-NAND2-NAND2-INV</td>
<td>8</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>f</td>
<td>N</td>
</tr>
<tr>
<td>logical effort</td>
<td>( g )</td>
<td>( \sum</td>
</tr>
<tr>
<td>electrical effort</td>
<td>( h )</td>
<td>( \sum</td>
</tr>
<tr>
<td>branching effort</td>
<td>( b )</td>
<td>( \sum</td>
</tr>
<tr>
<td>effort</td>
<td>( f )</td>
<td>( \sum f )</td>
</tr>
<tr>
<td>effort delay</td>
<td>( f )</td>
<td>( \sum f )</td>
</tr>
<tr>
<td>parasitic delay</td>
<td>( p )</td>
<td>( \sum p )</td>
</tr>
<tr>
<td>delay</td>
<td>( d = f + p )</td>
<td>( D = \sum D = D_g + P )</td>
</tr>
</tbody>
</table>

### Method of Logical Effort

1) Compute path effort 
2) Estimate best number of stages 
3) Sketch path with N stages 
4) Estimate least delay 
5) Determine best stage effort 
6) Find gate sizes

\[
F = GBH \\
N = \log_4 F \\
D = NF^* + P \\
f = F^* \\
C_a = \frac{g_C w}{f}
\]

### 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

### Summary

- 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 \( \log_F FO4 \) inverter delays
  - Inverters and NAND2 best for driving large caps
- Provides language for discussing fast circuits
  - But requires practice to master

### Wires
Outline

• Introduction
• Wire Resistance
• Wire Capacitance
• Wire RC Delay
• Crosstalk
• Wire Engineering
• Repeaters

Introduction

• Chips are mostly made of wires called interconnect
  – In stick diagram, wires set size
  – Transistors are little things under the wires
  – Many layers of wires
• Wires are as important as transistors
  – Speed
  – Power
  – Noise
• Alternating layers run orthogonally

Wire Geometry

• Pitch = w + s
• Aspect ratio: AR = t/w
  – Old processes had AR ≪ 1
  – Modern processes have AR ≈ 2
    • Pack in many skinny wires

Wire Resistance

\[ p = \text{resistivity (}\Omega \text{m)} \]

\[ R = \frac{\rho}{l} \frac{t}{w} \]
Wire Resistance

- $\rho = \text{resistivity (}\Omega\text{m})$
- $R = \frac{\rho l}{W} = \frac{1}{\rho W}$
- $R_{\square} = \text{sheet resistance (}\Omega/\square\text{)}$
  - $\square$ is a dimensionless unit(!)
- Count number of squares
  - $R = R_{\square} \times (\# \text{ of squares})$

Sheet Resistance

- Typical sheet resistances in 180 nm process

<table>
<thead>
<tr>
<th>Layer</th>
<th>Sheet Resistance ($\Omega/\square$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diffusion (silicided)</td>
<td>3-19</td>
</tr>
<tr>
<td>Diffusion (no silicide)</td>
<td>50-200</td>
</tr>
<tr>
<td>Polysilicon (silicided)</td>
<td>3-19</td>
</tr>
<tr>
<td>Polysilicon (no silicide)</td>
<td>50-400</td>
</tr>
<tr>
<td>Metal 1</td>
<td>0.08</td>
</tr>
<tr>
<td>Metal 2</td>
<td>0.05</td>
</tr>
<tr>
<td>Metal 3</td>
<td>0.05</td>
</tr>
<tr>
<td>Metal 4</td>
<td>0.03</td>
</tr>
<tr>
<td>Metal 5</td>
<td>0.02</td>
</tr>
<tr>
<td>Metal 6</td>
<td>0.02</td>
</tr>
</tbody>
</table>

Contacts Resistance

- Contacts and vias also have 2-20 $\Omega$
- Use many contacts for lower $R$
  - Many small contacts for current crowding around periphery

Capacitance Trends

- Parallel plate equation: $C = \varepsilon A/d$
  - Wires are not parallel plates, but obey trends
  - Increasing area ($W, t$) increases capacitance
  - Increasing distance ($s, h$) decreases capacitance
- Dielectric constant
  - $\varepsilon = k\varepsilon_0$
    - $\varepsilon_0 = 8.85 \times 10^{-14} \text{ F/cm}$
    - $k = 3.9$ for SiO$_2$
- Processes are starting to use low-k dielectrics
  - $k = 3$ (or less) as dielectrics use air pockets

Choice of Metals

- Until 180 nm generation, most wires were aluminum
- Modern processes often use copper
  - Cu atoms diffuse into silicon and damage FETs
  - Must be surrounded by a diffusion barrier

<table>
<thead>
<tr>
<th>Metal</th>
<th>Bulk Resistivity ($\Omega*cm$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Silver (Ag)</td>
<td>1.6</td>
</tr>
<tr>
<td>Copper (Cu)</td>
<td>1.7</td>
</tr>
<tr>
<td>Gold (Au)</td>
<td>2.2</td>
</tr>
<tr>
<td>Aluminum (Al)</td>
<td>2.8</td>
</tr>
<tr>
<td>Tungsten (W)</td>
<td>5.3</td>
</tr>
<tr>
<td>Molybdenum (Mo)</td>
<td>5.3</td>
</tr>
</tbody>
</table>
M2 Capacitance Data

- Typical wires have ~ 0.2 fF/μm
  - Compare to 2 fF/μm for gate capacitance

Diffusion & Polysilicon

- Diffusion capacitance is very high (about 2 fF/μm)
  - Comparable to gate capacitance
  - Diffusion also has high resistance
  - Avoid using diffusion runners for wires!
- Polysilicon has lower C but high R
  - Use for transistor gates
  - Occasionally for very short wires between gates

Lumped Element Models

- Wires are a distributed system
  - Approximate with lumped element models

  - 3-segment π-model is accurate to 3% in simulation
  - L-model needs 100 segments for same accuracy!
  - Use single segment π-model for Elmore delay

Example

- Metal2 wire in 180 nm process
  - 5 mm long
  - 0.32 μm wide

  - Construct a 3-segment π-model
    - \( R_Ω = 0.05 \, \text{Ω} \)
    - \( C_{\text{permicron}} = 0.2 \, \text{fF/μm} \)

  - \( R = 781 \, \text{Ω} \)
  - \( C = 1 \, \text{pF} \)

  \[
  \begin{align*}
  R & = 260 \, \text{Ω} \\
  C & = 167 \, \text{fF} \\
  \end{align*}
  \]

Wire RC Delay

- Estimate the delay of a 10x inverter driving a 2x inverter at the end of the 5mm wire from the previous example.
  - \( R = 2.5 \, \text{kΩ/μm} \) for gates
  - Unit inverter: 0.36 μm nMOS, 0.72 μm pMOS

  \[
  t_{pd} = \]
Wire RC Delay

- Estimate the delay of a 10x inverter driving a 2x inverter at the end of the 5mm wire from the previous example.
  - $R = 2.5 \, \text{k} \, \Omega$ for gates
  - Unit inverter: $0.36 \, \mu \text{m} \, n\text{MOS}, \ 0.72 \, \mu \text{m} \, p\text{MOS}$
  - $t_{pd} = 1.1 \, \text{ns}$

\[
\begin{array}{c}
\text{Driver} \\
690 \, \Omega \\
\text{Wire} \\
500 \, \text{fF} \\
\text{Load} \\
4 \, \text{fF}
\end{array}
\]

Crosstalk

- A capacitor does not like to change its voltage instantaneously.
- A wire has high capacitance to its neighbor.
  - When the neighbor switches from 1->0 or 0->1, the wire tends to switch too.
  - Called capacitive coupling or crosstalk.
- Crosstalk effects
  - Noise on nonswitching wires
  - Increased delay on switching wires

Crosstalk Delay

- Assume layers above and below on average are quiet
  - Second terminal of capacitor can be ignored
  - Model as $C_{\text{gnd}} = C_{\text{top}} + C_{\text{bot}}$
- Effective $C_{\text{adj}}$ depends on behavior of neighbors
  - Miller effect

\[
\begin{array}{c|c|c|c}
\text{B} & \Delta V & C_{\text{effAdj}} & \text{MCF} \\
\hline
\text{Constant} & & & 1 \\
\text{Switching with A} & & & 0 \\
\text{Switching opposite A} & & & 2
\end{array}
\]

Crosstalk Noise

- Crosstalk causes noise on nonswitching wires
- If victim is floating:
  - model as capacitive voltage divider

\[
\Delta V_{\text{victim}} = \frac{C_{\text{adj}}}{C_{\text{gnd}} + C_{\text{adj}}} \Delta V_{\text{aggressor}}
\]

Driven Victims

- Usually victim is driven by a gate that fights noise
  - Noise depends on relative resistances
  - Victim driver is in linear region, agg. in saturation
  - If sizes are same, $R_{\text{aggressor}} = 2-4 \times R_{\text{victim}}$

\[
\begin{align*}
\Delta V_{\text{victim}} &= \frac{C_{\text{adj}}}{C_{\text{gnd}} + C_{\text{adj}}} \Delta V_{\text{aggressor}} \\
\tau_{\text{victim}} &= R_{\text{victim}} \left( C_{\text{gnd}} + C_{\text{adj}} \right) \\
\tau_{\text{aggressor}} &= R_{\text{aggressor}} \left( C_{\text{gnd}} + C_{\text{adj}} \right)
\end{align*}
\]
**Coupling Waveforms**

- Simulated coupling for $C_{adj} = C_{victm}$

**Noise Implications**

- So what if we have noise?
  - If the noise is less than the noise margin, nothing happens
  - Static CMOS logic will eventually settle to correct output even if disturbed by large noise spikes
    - But glitches cause extra delay
    - Also cause extra power from false transitions
  - Dynamic logic never recovers from glitches
  - Memories and other sensitive circuits also can produce the wrong answer

**Wire Engineering**

- Goal: achieve delay, area, power goals with acceptable noise
- Degrees of freedom:
  - Width
  - Spacing
  - Layer
  - Shielding
Repeaters

- R and C are proportional to $l$
- RC delay is proportional to $l^2$
  - Unacceptably great for long wires

Repeaters

- R and C are proportional to $l$
- RC delay is proportional to $l^2$
  - Unacceptably great for long wires
- Break long wires into N shorter segments
  - Drive each one with an inverter or buffer

Repeaters Design

- How many repeaters should we use?
- How large should each one be?
- Equivalent Circuit
  - Wire length $l/N$
  - Wire Capacitance $C_w/N$, Resistance $R_w/N$
  - Inverter width $W$ (nMOS = $W$, pMOS = 2$W$)
  - Gate Capacitance $C_g W$, Resistance $R/W$

Repeaters Design

- How many repeaters should we use?
- How large should each one be?
- Equivalent Circuit
  - Wire length $l$
  - Wire Capacitance $C_w$, Resistance $R_w l$
  - Inverter width $W$ (nMOS = $W$, pMOS = 2$W$)
  - Gate Capacitance $C_g W$, Resistance $R/W$

Repeaters Results

- Write equation for Elmore Delay
  - Differentiate with respect to $W$ and $N$
  - Set equal to 0, solve

  \[
  \frac{l}{N} = \frac{2 R C'}{\sqrt{R_w C_w}}
  \]

  \[
  \frac{l_{req}}{l} = \left(2 + \sqrt{2}\right) \sqrt{R C' R_w C_w} = \text{~60-80 ps/mm in 180 nm process}
  \]

  \[
  W' = \frac{R C_w}{R C'}
  \]