Delay Example
NAND Gate Delay Example (1)

- No velocity saturation and $\lambda = 0$
- Assume, $R_{N_{DN}}' = 1k\Omega \cdot \mu m$, $R_{P_{UP}}' = 2k\Omega \cdot \mu m$. $C_{GN}' = C_{GP}' = C_G' = 2fF/\mu m$, $C_{DN}' = C_{DP}' = C_D = 1fF/\mu m$
  - $R_{N12} = R_{N1} + R_{N2} = R_{N_{DN}}'/(W_{N1}/\lambda^*) = 1k\Omega \cdot \mu m/(2\mu m/2) = 1k\Omega$
  - $R_P = R_{P_{UP}}'/(W_P) = 2k\Omega \cdot \mu m/(3\mu m) = 0.67k\Omega$
  - $C_{LOAD} = C_{self} + C_{gate}$
    - $C_{self} = C_D'*(3 + 2) \mu m$ (sharing) or $C_D'*(6 + 2) \mu m$ (not sharing)
    - $C_{gate} = C_G'*(12 + 6) \mu m$
  - $t_{NANDpull_{up}} = 0.69*0.67k*(41f) = 18.8ps$
  - $t_{NANDpull_{dn}} = 0.69*1k*(41f) = 28.3ps$
NAND Gate Delay Example

- With velocity saturation $E_cL = 0.3$ and $\lambda = 0.5$
- Calculate $\chi = W_0/W_{N1} = 0.57$
  - $R_{N12} = R_{N_{DN}}/(\chi W_{N1}) = 1\Omega \cdot \mu m / (0.57 \cdot 2\mu m) = 0.88k\Omega$
  - $R_P$ and $C_{LOAD}$ doesn’t change.
  - $t_{NANDpull_{dn}} = 0.69 \cdot 0.88k \cdot (41f) = 24.9ps$
Fan-In and Fan-Out

- There are several definitions for the same terms.
- **Fan-In**
  - The number of inputs
  - An indication of the input load that the gate presents to a predecessor gate.
    - Because the series stack is roughly the number of inputs
    - Later we will use Logical Effort to embed this concept.
- **Fan-Out**
  - The number of gates driven by the gate
  - An indication of the capacitive loading of a gate
    - Depends on the type of gate.
  - Typically normalize the loading to the gate capacitance of an inverter with equal drive strength as the gate.
    - $\text{FO} = \frac{C_{\text{LOAD}}}{C_{\text{INV}}}$, where $C_{\text{INV}} = C_G(W_P+W_N)$ and $R_{\text{INV}} = R_{\text{PULL UP/DN}}$ of the logic gate
Logical Effort
Logical Effort

- Delay = \((0.69) \, R_{gate}(C_{load} + C_{self}) = (0.69) \, (R_{gate}C_{load} + R_{gate}C_{self})\)
- Logical Effort basic equation: \(d = f + p\)
  - \(d = \text{delay}/\tau_o\)
  - \(\tau_o = 0.69 \, R_0C_0\)
  - \(f = \text{effort delay (also fanout)}\)
  - \(p = \text{parasitic delay}\).
- \(d = \text{Delay}/\tau = (R_{gate}C_{load} + R_{gate}C_{self})/ \, R_0C_0\)
  - Normalized to the delay of a FO-1 inverter (w/ no self load)
  - Set \(R_0 = R_{gate}\), \(d = \text{fanout + normalized parasitic}\).
- Key: \(d\) is a measure that is independent of process, voltage, temp.
Logical and Electrical Effort

- Instead of just \( d = f + p \), let \( f = gh \)
  - \( g = \) logical effort (of a gate)
    - Cost of implementing logic
  - \( h = \) electrical effort
    - Cost of driving a load.

- \( f = \frac{R_{\text{gate}}}{R_0} \frac{C_{\text{load}}}{C_0}, \ p = \frac{R_{\text{gate}}}{R_0} \frac{C_{\text{self}}}{C_0} \)
  - Let \( R_0 = R_{\text{inv}} \), where \( R_{\text{inv}} = R_{\text{gate}}, \ C_0 = C_{\text{inv}} \)
  - \( p = \frac{C_{\text{self}}}{C_{\text{inv}}} , \ f = \frac{C_{\text{in}}}{C_{\text{in}}} \frac{C_{\text{load}}}{C_{\text{in}}} C_{\text{inv}} \)
    - \( C_{\text{in}} \) is the gate’s input capacitance (for the particular input)
  - \( g = \frac{C_{\text{in}}}{C_{\text{inv}}} \)
    - Each gate (and each input of every gate) has different values.
  - \( h = \frac{C_{\text{load}}}{C_{\text{in}}} \)
    - Output to input capacitance ratio.
Computing Logical Effort: $g$

$g$ is an unitless inherent characteristic of the gate
- Not a function of size of the gate.
- It is a function of the construction of the gate (connection and relative size between transistors)
- An indication of the “cost” of implementing the function.

Procedure:
1. Choose an input.
2. Find total device width driven by that input.
3. Find $W_P$, the pull-up device width of an inverter (single device) that has equivalent drive strength as a gate’s pull-up of that input.
4. For the reference inverter with Equal Rise/Fall Resistance, $\beta \sim \mu$, with $W_P$ from Step 3, determine the total gate widths of the inverter.
5. Divide Step 2 by Step 4 to determine $g_{up}$.
6. Repeat Steps 3-5 for pull-down device for $g_{down}$.
- The two $g$’s would only be different if $\beta$ of the gate does not have equal Rise/Fall resistances.
Assume not velocity saturated and $\lambda=0$

For a NOR gate
- Let reference $\beta = \mu = 3$

Reference inverter
- $W_P:W_N = 6:2$
- $C_{G_{\text{INV}}} = 8$

NOR gate input capacitance
- $C_{G_{\text{NOR}}} = 14$
- Logical Effort = 7/4
Meaning of g in terms of $R_N$ or $R_P$

- The g is a way to de-rate the $R_{eq}$ of a gate from that of an inverter with equal gate capacitance.
- Write
  - $R_{eq} = \tau / (0.69C_{INV}) = \tau \times (g/0.69C_{IN})$.
- If $\tau = (0.69)R_{INV}C_{IN}$ where $R_{INV}$ is the resistance of an inverter with input capacitance = $C_{IN}$,
  - $R_{eq} = R_{INV} \times (g)$ where g is the de-rating factor.
Calculating Parasitic Effort: $p$

$p = \frac{R_{\text{gate}} C_{\text{self}}}{R_{\text{INV}} C_{\text{inv}}} \text{ (where } R_{\text{gate}} = R_{\text{INV}})$

Estimate if

- Know $C_D'$ versus $C_G'$
- Assume no sharing of diffusion (or sharing).

Example: assume $C_D' = 0.5C_G' = 0.5C_O$

- For an inverter $C_{\text{self}}/C_{\text{inv}} = p_{\text{INV}} = 0.5$
  - Higher $C_{S/D}/C_G$ results in larger $p$ (penalizing delay more).

NOR Gate

- $C_{S/D_{\text{NOR}}} = 12 + 2(\text{shared})$
- $= 14 = 7C_O$
- $C_{\text{INV}} = 8C_O$
- $p_{\text{NOR}} = 7/8 \approx 2p_{\text{INV}}$
Asymmetric Examples

- If NOR sizing for PMOS is different.
  - Reference Inverter is 6:2.
    - $C_{inA} = 10$, $C_{inB} = 26$
    - $g_A = (10/8=5/4)$ not equal to $g_B = (26/8=13/4)$
  - B input is much worse... for a reason.

- For more complex gates:
  - Reference Inverter is 9:3.
    - $g_{EDC} = 21/12=7/4$, $g_{A'B'} = 30/12 = 5/2$

The larger $g$ is bad.
More Effort to do the logic
Logical Effort when Accounting for Non-Idealities

- Assume that $\beta=2$ for reference inverter
- The equivalent PMOS (of inverter)
  - $W_{INV_P}/W_{P1} = 0.71$
  - $W_{INV_P} = 6\times0.71 = 4.2$, $W_{INV_N} = W_{INV_P}/\beta$
  - $C_{INV} = C_G'(6.3)$
  - $g_A = 8.1/6.3 = 1.3$, $g_B = 14.1/6.3 = 2.2$
- Note that with velocity saturation, the logical effort for series stack is lower.

![NOR Diagram]
# A Catalog of Gates

<table>
<thead>
<tr>
<th>Gate Type</th>
<th>g for Different number of inputs</th>
<th>Parasitic delay</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Inverter</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>NAND</td>
<td>4/3</td>
<td>5/3</td>
</tr>
<tr>
<td>NOR</td>
<td>5/3</td>
<td>7/3</td>
</tr>
<tr>
<td>Multiplexer</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>XOR,XNOR</td>
<td>4</td>
<td>12</td>
</tr>
</tbody>
</table>

- Let reference $\beta = \mu = 2$
- Mux is tri-state inverters shorted together.
- XOR assumes that input is bundled (a,a’)
- $p_{INV} \sim 1$
- $p_{GATE}$ in this table does not include intermediate nodes.
Example: Ring Oscillator

Assume an N-stage inverter (odd-N) that feedback to itself.
- Toggles 0-1-0-1… with a rate that depends on the delay of the chain.
  - \( T_{\text{LOW}} = T_{\text{HIGH}} = T_{\text{DELAY}} \)

Approximate the frequency.
- \( g_{\text{INV}} = 1, \quad p_{\text{INV}} = 1, \quad h = 1 \)
- Stage delay, \( d = gh + p = 2 \).
- Delay of chain = \( N \times d = 2N \) (normalized)
- Cycle time = \( 4N \tau_0 \) (\( \tau_0 \) is the FO-1 delay of inverter w/o self-load)
- Frequency = \( 1/(4N \tau_0) \)
Multi-Stage Sizing
Simplified Problem: Buffering

Assume
- \( \beta \) (PNrat) = \( \mu \) (mobility ratio).
- \( I_{PSAT} = I_{NSAT} \)
- \( R_o = \text{Resistance for NMOS with size } W_0 \text{ or PMOS with size } \beta W_0 \)
- \( C_o = \text{Gate load capacitance of N+PMOS of size } W_0 + \beta W_0 \)
- \( C_D = \text{Self load capacitance of N+PMOS of size } W_0 + \beta W_0 \)
- \( \tau_o = (0.69)R_o(C_o) \)
- \( \tau_{INV} = (0.69)R_o(\alpha C_o + C_D) = \tau_o(\alpha + p_{inv}); d_{inv} = \alpha + p_{inv} \)

Determine the sizes of each of the N stages for minimum delay.
- \( d_1 = \alpha_1 + p_{inv} \)
- \( d_2 = \alpha_2 / \alpha_1 + p_{inv} \)
Optimum Fanout, $f$

- Fanout of each stage of the inverter chain
  - Stage 1 = $\alpha_1$, Stage 2 = $\alpha_2/\alpha_1$
- Assuming that the fanout of each stage is equal, $\alpha_0$
  - Let $\alpha_1 = \alpha_0$, $\alpha_2 = \alpha_0^2$, $\alpha_3 = \alpha_0^3$
  - Let $C_{out} = C_{in} \alpha_0^N$ so $C_{out}/C_{in} = \alpha_0^N$
- Total Delay = Sum (Delay of stage 1-N)
  - $d = N(\alpha_0 + p_{INV})$
- For a given $N$, the optimum $\alpha_0$

\[
\alpha_0 = N \sqrt[2N]{C_{out}/C_{in}}
\]
Optimum Number of Stages

For an arbitrary N

\[ N = \frac{\ln\left(\frac{C_{out}}{C_{in}}\right)}{\ln(\alpha_0)} \]

\[ d = N(\alpha_0 + p) \]

For \( p \approx 1-2 \),
- Optimal \( \alpha_0 \) or \( f \approx 4 \)
- Fan-Out=4 is a decent place to start for sizing consecutive stages.
Multi-stage Random Logic

- Usually analyze the critical path (a given input to output)
- Delay of a multi-stage network is the sum of delay of each stage.
  - Path Effort Delay \( D_F = \sum_i f_i \)
  - Path Parasitic Delay \( P = \sum_i p_i \)
  - Total Path Delay \( D = \sum_i d_i = D_F + P \)

\[
\begin{align*}
W_P:W_N &= 2:2 \\
g &= 4/3 \\
p &= 2
\end{align*}
\]

\[
\begin{align*}
W_P:W_N &= 6:6 \\
g &= 5/3 \\
p &= 2
\end{align*}
\]

\[
\begin{align*}
W_P:W_N &= 24:6 \\
g &= 4/3 \\
p &= 2
\end{align*}
\]

- \( f_1 = 4 \)
- \( f_2 = 3.33 \)
- \( f_3 = 3.33 \)
- \( D_F = 10.66, P = 6 \)
- \( D = 16.66 \) (gate delays)
Total Effort

- A path can have a total effort
  - $G = \text{Path Logical Effort}$
  - $H = \text{Path Electrical Effort}$
  - $F = \text{Path Effort}$
  - Consider path effort as fanout. Total fanout should be a product of all fanouts.
  - Consider the path as a single “gate”

Calculations
- $G = (4/3)^2(5/3) = 3$
- $H = 60/4 = 15$
- $F = 44.4; \quad F \neq GH \text{ (yes for this case)}$

\[
G_{PATH} = \prod_{i=1}^{N} g_i
\]
\[
H_{PATH} = \frac{C_{OUT}}{C_{IN}}
\]
\[
F_{PATH} = \prod_{i=1}^{N} f_i
\]

- $f_1 = 4$
- $f_2 = 3.33$
- $f_3 = 3.33$
Branching

What if circuits branch?
- $G_{\text{PATH}}$ is the same = 2.96
- $H_{\text{PATH}}$ is the same = 15
- $F_{\text{PATH}}$ differs.
  - $h_1 = 24/4 = 6$, $h_2 = 60/12 = 5$, $h_3 = 2$
  - $F_{\text{PATH}} = 4/3 * 6 * 4/3 * 5 * 5/3 * 2 = 177.8$
- $F$ is no longer GH.
Branching Effort

We can fix that by introducing one more type of effort.
- Branching Effort, B, such that \( F = BGH \).

Define

\[
B = \prod_i b_i
\]

- \( b_i = \frac{C_{ON-PATH} + C_{OFF-PATH}}{C_{ON-PATH}} \)
- \( C_{ON-PATH} = \) capacitance of gate that is on the path in question.
- \( C_{OFF-PATH} = \) capacitance of gates/wire not on the path.
- Or consider as \( C_{TOTAL}/C_{USEFUL} \)

\[
b_1 = \frac{24}{12} = 2
\]

\[
b_2 = \frac{60}{30} = 2
\]

\[
b_3 = 1 \text{ (no branch)}
\]

\[
B = 4
\]

\[
BGH = 177.8 \text{ equals } F
\]
Optimum Delay

- Given the number of stages, $N$
  - Optimum effort is simply
    \[ f_{opt} = \sqrt[4]{N} \sqrt{BGH} \]

- If the number of stages is small resulting in large $N$
  - Inverters can be inserted to increase $N$.
  - $f_{opt}$ for arbitrary $N$?
    - Same problem as in buffering.
    - Optimum depends on $p_{AVG}$ just like the buffer optimization earlier.
  - Optimal $N$ is such that $f_{opt} \approx 4$ per stage.

\[
D = Nf_{opt} + \sum_i p_i = N(f_{opt} + p_{AVG})
\]

\[
Np_{AVG} = \sum_i p_i
\]

\[
\frac{dD}{df} = 0 = \frac{d[\ln(F)]}{\ln(f_{opt})} \frac{(f_{opt} + p_{AVG})}{df}
\]

Same as for Inverters
Determining Gate Sizes

Once the path effort is determined, it is quite easy to determine the appropriate gate sizes.

- Start from the output of a path ($C_{out_i}$).
- Work backwards to the input
  - Check your work if the input is the same as the specification.
  - Assuming each unit $W$ has capacitance of unit $C$

\[
C_{in_i} = \frac{C_{out_i} g_i}{f_{opt}}
\]

Example:

- $C_{in3} = 60 \times \frac{5}{3} \times \frac{1}{5.6} = 17.9$
- $C_{in2} = b_2 \times 17.9 \times \frac{4}{3} \times \frac{1}{5.6} = 8.5$
- $C_{in1} = b_1 \times 8.5 \times \frac{4}{3} \times \frac{1}{5.6} = 4$
- $W_{P3} = 4/5 \times 17.9 = 14.3$
- $W_{P2} = \frac{1}{2} \times 8.5 = 4.25$

Diagram:

- Gate 2 (dup) $g = \frac{4}{3}$, $p = 2$
- Gate 3 (dup) $g = \frac{5}{3}$, $p = 2$
- $C_{in} = 4$
- $C_{out} = 60$
Example: 2-Stage; 8-Input AND

- Assume symmetric 8-input AND function.
  - \( \beta=\mu=2 \), width of \( 3W_o \) is unit \( C \)
  - \( g_{\text{NAND}}=10/3 \), \( p_{\text{NAND}}=8 \) for an 8-input NAND
  - \( g_{\text{INV}}=1 \), \( p_{\text{INV}}=1 \)
  - \( C_{\text{out}}=100 \), \( C_{\text{in}}=1 \)

- Logical Effort
  - \( G=10/3 \), \( B=1 \), \( H=100 \)
  - \( F=333.3 \)
  - For 2 stages, \( f_{\text{opt}}=18.3 \)
    - \( h_2=18.3 \), \( C_{\text{in}2}=5.5 \), \( W_P=11W_o \)
    - \( h_1=5.5 \), \( C_{\text{in}1}=1 \), \( W_P=0.6W_o \)
    - Delay = 36.6+9=45.6
  - For 3 stages, \( f_{\text{opt}}=7 \)
    - Delay = 31
  - For 4 stages, \( f_{\text{opt}}=4.3 \) (2 extra inverters), \textbf{Delay = 28.2}
  - For 5 stages, \( f_{\text{opt}}=3.19 \) (closer to optimal of 3.6), Delay=28
  - For 6 stages, \( f_{\text{opt}}=2.63 \) (below optimal of ~3.6), Delay=28.8
Many ways to implement this same function.

Use a tree of fewer input AND gates.
- \((A_0A_1)(A_2A_3)\ldots\)
- If multiple ANDs (as in a memory decoder), then partial results can be shared.
Example: 2-input Implementation

- Assuming that $3W_0$ has capacitance of unit C
- $F = \frac{4}{3^3} \times 100 = 235$
  - $f_{opt} = 2.48$
  - $D = 6 \times 2.48 + 3 \times 2 + 3 \times 1 = 14.88 + 9 = 23.88$ (better than 8-NAND)
- $G_7: C_{in7} = 100g_{inv}/f = 40. W_P = 80W_0$
- $G_6: C_{in6} = C_{in7}g_{nand}/f = 21.5. W_P = 32W_0$
- $G_5: C_{in5} = C_{in6}g_{inv}/f = 8.7. W_P = 17.4W_0$
- $G_4: C_{in4} = C_{in5}g_{nand}/f = 4.7. W_P = 7W_0$
- $G_3: C_{in3} = C_{in4}g_{inv}/f = 1.88. W_P = 3.8W_0$
- Double check $G_2: C_{in} = C_{in3}g_{nand}/f = 1. W_P = 1.5W_0$
Design Methodology

1. Draw network
2. Buffer non-critical paths with minimum-sized gates.
   • Minimize loading on critical path.
   • Simplifies sizing of non-critical path.
3. Estimate total effort along each path (without branching)
4. Verify that the number of stages is appropriate.
   • Add inverters if $f_{opt} > 5$
5. Assign branch ratio of each branch.
   • Estimate based on the ratio of the Effort of the paths.
   • Ignore paths that have little effect (i.e. min-sized).
   • Include wire capacitances
6. Compute delays for the design (include parasitic delay)
   • Adjust branching ratios (especially with wire capacitance).
   • Repeat if necessary until delay meets specification.
7. Reoptimize logic network if $f_{opt}$ is small. (Return to step 3)
Deviations from Optimal

What is the impact on delay if the choice of number of stages or sizing is not perfect?

Figure 3.7 — The relative delay compared to the best possible, as a function of $s$, the size error of a stage. Assumes $p_{inv} = 1$.

Figure 3.5 — The delay relative to the best possible, as a function of the relative error in the number of stages used, $N/N$. Assumes $p_{inv} = 1$. 

UCLA