CMOS Combinational Logic

Delay
- RC Models
- RC Delay
- Logical Effort

Energy
Area

RC Models

- $C_{SB}$ capacitors do not actually switch, so ignore.
- Lump $C_{DBP} + C_{DBN}$ since from well to constant nodes.
- Lump $C_{GSP} + C_{GSO}$ since from well to constant nodes.
- Assume PMOS mobility 2x worse than NMOS mobility.
Define \( C \) = gate capacitance of minimum sized NMOS

Define \( R \) = effective resistance of minimum sized NMOS

Define \( k \) = how much wider a transistor is compared to NM MOS
RC DECAY: INVERTER

\[ V_{out}(t) = V_{DD} e^{-t/RC} \]

Let \( t = RC \),

\[ \frac{V}{I} = \frac{Q}{I} = \frac{Q}{Q/s} = \text{seconds} \]

\[ t_{PD} = \text{propagation delay, time until } V_{out} = \frac{V_{DD}}{2} \]

\[ = \ln(2) \cdot R \cdot C_1 \quad \text{let } R' = \ln(2) \cdot R \]

\[ = R' \cdot C_1 = 2RC \]

We usually just assume effective resistance is scaled by \( \ln(2) \)

\[ t_{PD} = 2RC \]

\[ V_{out} = \frac{V_{DD}}{2} = V_{DD} e^{-t/\tau} \]

\[ \frac{1}{\tau} \frac{V_{DD}}{2} = e^{-t/\tau} \]

\[ \ln\left(\frac{1}{2}\right) = -\frac{t}{\tau} \]

\[ t = -\tau \ln\left(\frac{1}{2}\right) = \tau \ln(2) \]

\[ \text{Oscilator's low } V = IT \]

\[ \text{settling time } = \tau \]

\[ \text{DEF of } C \text{ is Q/V} \]
RC DELAY: 2-input NAND

\[ Y = \frac{1}{g} \]

\( g = \) output gain

\[ A \rightarrow Y \]

\[ B \rightarrow Y \]

\[ r = \text{input resistance} \]

\( C \) is also charged up to VDD!

**COMPLIQUED 2nd ORDER MODEL**

**APPROXIMATION**

\[ \tau = \tau_1 + \tau_2 = RC_1 + (R + R)C_2 \]

\[ = RC + (2R)(3C) \]

\[ = RC + 6RC = 7RC \text{ (3.5x slower than inverter)} \]

Best when one \( \tau \) much larger than other \( \tau \)

Even if \( \tau_1 = \tau_2 \), error < 1.570

**GENERALIZE TO ELmore DELAY**

\[ t_{PD} = \sum_{i} R_{ij} C_i \]

\[ \tau_{PD} = RC + 2RC + 3RC \]

\[ = 6RC \]
Elmore Delay of Trees

Delay of path from $x$ to $y$ is impacted by branch to $z$

Delay of path from $x$ to $z$ is impacted by branch to $y$

For path $x \rightarrow y$ we also lump $C_2 + C_3$ and use summed resistance, $R_o + R_i$

Similarly for path $x \rightarrow z$ we lump $C_1$ and use summed resistance, $R_o + R_i$

This extra term factors in delay of "Branch"

$$t_{pd, xy} = R_o C_o + (R_o + R_i + R_e) C_1 + (R_o + R_i)(C_2 + C_3)$$
$$= RC + 3RC + 4RC$$
$$= 8RC$$

Delay due to extra branch

$$t_{pd, xz} = R_o C_o + (R_o + R_i + R_2) C_2 + (R_o + R_i)(R_o + R_i + R_2) C_3 + (R_o + R_i) C_i$$
$$= RC + 3RC + 4RC + 2RC$$
$$= 10RC$$

Delay due to extra branch
RISE/FALL TIMES: INVERTER

From earlier, \( t_{PD,1 \rightarrow 0} = 2RC \), unequal rise/fall times.

\[
\frac{1}{3} \cdot 2R \quad \frac{1}{3} \cdot 2R
\]

\[
\frac{1}{3} \cdot 2R \quad \frac{1}{3} \cdot 2R
\]

For equal rise/fall times, the effective resistance of pullup must equal effective resistance of pulldown.

If we assume PMOS mobility 2x work that of NMOS, then PMOS must be 2x size of NMOS in an inverter for equal rise/fall times.

RISE/FALL TIMES: OUTPUT NAND

Size transistors so worst case effective resistance equal in both pull up and pull down networks.

Assumes worst case where only a single PMOS is pulling up output node.
\[ t_{PD, 1 \rightarrow 0} = RC + (R + R)3C = 3RC \]

\[ t_{PD, 0 \rightarrow 1} = (R + R)3C = 3RC \]

\[ t_{PD, 0 \rightarrow 1} = 2RC + 2RC = 4RC \]

\[ t_{PD, 0 \rightarrow 1} = 4RC \]

\[ t_{PD} = GRC \]

\[ t_{PD} = GRC \]

\[ t_{PD} = GRC \]

\[ t_{PD} = GRC \]

\[ t_{PD} = GRC \]

So delay depends on internal capacitance and the order in which inputs arrive on which inputs change.
RISE / FALL TIMES: Z INPUT NODE

$t_{pd, 1 \rightarrow 0}$
$A = 0 \rightarrow 1$
$B = 0 \rightarrow 1$

$t_{pd, 1 \rightarrow 0}$
$A = 0$
$B = 0 \rightarrow 1$

$t_{pd, 1 \rightarrow 0}$
$A = 0 \rightarrow 1$
$B = 0$

$t_{pd, 0 \rightarrow 1}$
$A = 0$
$B = 1 \rightarrow 0$

$t_{pd, 0 \rightarrow 1}$
$A = 1 \rightarrow 0$
$B = 0$

$t_{pd} = \frac{R}{C} G C$
$t_{pd} = \frac{3 R}{C} G C$

$t_{pd} = G R C + R \frac{1}{C}$
$t_{pd} = G R C + 4 R C$
$t_{pd} = 10 R C$
$t_{pd} = 6 G R C$

$t_{pd} = \frac{R}{C} \frac{1}{C} + \frac{R}{C} \frac{1}{C} G C$
$t_{pd} = 2 R C + 6 G R C$
$t_{pd} = 8 G R C$
$t_{pd} = 6 G R C$
<table>
<thead>
<tr>
<th>Gate</th>
<th>Worst</th>
<th>Next</th>
<th>Worst</th>
<th>Best</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Inv</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>NAND</strong></td>
<td>No Internal C &amp; D</td>
<td>GRC</td>
<td>GRC</td>
<td>GRC</td>
</tr>
<tr>
<td></td>
<td>1 Internal C &amp; D</td>
<td>TEC</td>
<td>GRC</td>
<td>BRC</td>
</tr>
<tr>
<td><strong>Nor</strong></td>
<td>No Internal C &amp; D</td>
<td>GRC</td>
<td>SNC</td>
<td>GRC</td>
</tr>
<tr>
<td></td>
<td>1 Internal C &amp; D</td>
<td>10EC</td>
<td>3RC</td>
<td>BRC</td>
</tr>
</tbody>
</table>

Is this a fair comparison? No, we are not normalizing anything across these gates. Need to either normalize:

1) **Input Gate Cap** (i.e., load on previous gate).
2) **Drive Strength** (i.e., effective resistance).

**Effective Resistance of All 3 Gates**

- **Inv**: \( T \)
- **NAND**:
  - \( 2R \) — twice the effective resistance
  - \( R \) — so half the drive current
- **Nor**:
  - \( R \) — so half the drive current

Here are all three gates sized to have equal rise and fall times and the same drive strength.

\[
\begin{align*}
R_{\text{eff}} &= R \\
C_{\text{in}} &= 3C \\
C_{\text{c}} &= 3C \\
E_{\text{pp,164}} &= 3RC \\
E_{\text{pp,0-91}} &= 3RC
\end{align*}
\]
**LARGER GATES**

\[
\begin{align*}
-\frac{1}{3} & \quad \frac{1}{3} \\
\frac{1}{2} & \quad \frac{1}{2} \\
\frac{1}{2} & \\
\frac{1}{2} & \\
\frac{1}{4} & \\
\end{align*}
\]

**Worst Case**

\[
\begin{align*}
\tau_{po,1} & \rightarrow 0 = \frac{R}{K} KC + \left( \frac{R}{K} + \frac{R}{K} \right) 3.5KC = 7RC \\
\tau_{po,1} & \rightarrow 0 = \frac{2R}{K} 3.5KC + \left( \frac{2R}{K} \right) KC = 9RC
\end{align*}
\]

Same as before!

This is the parasitic delay, it is independent of size.

**LARGER LOADS**

\[
\begin{align*}
-\frac{1}{3} & \quad \frac{1}{3} \\
\frac{1}{2} & \quad \frac{1}{2} \\
\frac{1}{2} & \\
\frac{1}{2} & \\
\frac{1}{4} & \\
\end{align*}
\]
\[ t_{PD,1-\infty} \]

\[ \frac{R}{K} \cdot KC + \left( \frac{R}{K} + \frac{R}{K} \right) (3K + \beta) C \]

\[ RC + 2 \frac{R}{K} (3K + \beta) C \]

\[ RC + CRC + 2 \frac{R}{K} \beta C \]

\[ \frac{CRC + \frac{R}{K} CRC}{CRC} \]

Effort Delay: depends on complexity of gate, size of gate, and what it is driving.

Parasitic Delay: inherent delay when no load is attached, independent of sizing.

Note: Increasing \( \beta \), increases effort delay.

Increasing \( K \), decreases effort delay.

But will increase \( \beta \) of previous gate!
Deriving Linear Decay Mode (logical effort)

Abstract all CMOS logic gates as

\[ C_{W} = \alpha C_{T} \]

\[ R_{i} = R_{ui} = R_{di} = \frac{R_{T}}{\alpha} \]

\[ C_{pi} = \alpha C_{pt} \]

\[ t_{pd} = d_{abs} = \frac{R_{i}}{C_{W}} \left( \frac{C_{out} + C_{pi}}{C_{W}} \right) \]

\[ = \frac{R_{T}}{\alpha} \left( \frac{C_{out}}{C_{W}} \right) + \frac{R_{T}}{\alpha} \left( \alpha C_{pt} \right) \]

\[ = \frac{R_{T}}{\alpha} \left( \frac{C_{out}}{C_{W}} \right) + \frac{R_{T}}{\alpha} \left( \alpha C_{pt} \right) \]

\[ = R_{T} C_{T} \left( \frac{C_{out}}{C_{W}} \right) + R_{T} C_{pt} \]
\[ \alpha_{\text{ass}} = \frac{R_t C_t}{C_{\text{inv}}} \left( \frac{C_{\text{out}}}{C_{\text{inv}}} \right) + R_t C_{\text{pt}} \]

- Parasitic Delay
- Effort Delay
- \( \alpha \) is "infinite" in just \( C_{\text{inv}} \)

\[ (c + \tau) = \frac{R_{\text{inv}} C_{\text{inv}}}{R_{\text{inv}} C_{\text{inv}}} \]

- Delay units
- Logical effort
- Electrical effort
- Parasitic Delay

\[ \alpha_{\text{ass}} = \frac{R_{\text{inv}} C_{\text{inv}}}{R_{\text{inv}} C_{\text{inv}}} \left( \frac{R_t C_t}{R_{\text{inv}} C_{\text{inv}}} \right) \left( \frac{C_{\text{out}}}{C_{\text{inv}}} \right) + R_{\text{inv}} C_{\text{inv}} \left( \frac{R_t C_{\text{pt}}}{R_{\text{inv}} C_{\text{inv}}} \right) \]

- Complexity of gate topology
- Ratio of gate time constant to inv time constant
- How much worse is gate at producing current compared to inv with same input cap
- How much more input cap is necessary to drive same current as min. inv.

\[ \alpha_{\text{ass}} = \gamma (\chi + \rho) \]

\[ \alpha_{\text{ass}} = \tau' \alpha \]

- Name of load to input cap
- \( \tau' \) \( \text{cour} \uparrow \text{delay} \uparrow \) to reduce delay need to \( \uparrow C_{\text{inv}} \)

For basic template was same effective impedance
as minimum inverter pair.

\[ R_t = R_{\text{inv}} \]

So

\[ g = \frac{R_t C_t}{R_{\text{inv}} C_{\text{inv}}} = \frac{C_t}{C_{\text{inv}}} \]

- Ratio of diff cap/mi vs gate cap/mi
- Crudely assume \( 1 \) for inv

\[ P_{\text{inv}} = \frac{R_t C_{\text{pt}}}{R_{\text{inv}} C_{\text{inv}}} = \frac{C_{\text{pt}}}{C_{\text{inv}}} \]
**TEMPLATES**

\[
\begin{align*}
\frac{R_{eff}}{R} &= 3C \\
\frac{C_I}{C_{in}} &= 4C \\
\frac{R_{4C}}{R_{3C}} &= \frac{R_{4C}}{R_{3C}} \\
\frac{g}{g_{0}} &= \frac{1}{3} \\
\frac{R_{4C}}{R_{3C}} &= \frac{10}{3} \\
\frac{g}{g_{0}} &= 2
\end{align*}
\]

**RECALL**  \( d = gh + p \)  **(LINEAR DELAY MODEL)**

**PLOT** \( d \) **as a function of** \( h \)

- **Nan:**  \( d = \frac{5}{3}h + 2 \)
- **NANO:**  \( d = \frac{4}{3}h + 2 \)
- **INV:**  \( d = h + 1 \)

\[ h = \frac{C_{in}}{C_{0}} \]

**PARASITIC DELAY**
MANY, MANY APPROXIMATIONS

- Elmore Delay
  - \( p = 1 \) for \( \mu \text{V} \) (i.e. \( 1 \mu \text{m diff cap} \approx 1 \mu \text{m gate cap} \))
  - Ignore internal parasitic cap
  - Equal rise and fall time
  - \( \mu_0 = \frac{1}{2} \mu_0 \)
- Ignore actual rise/fall times
- Ignore worst arrival time
- Ignore velocity saturation

Still reasonably good results when using logical effort for sizing even in modern technologies. Helps designers build the right intuition.
MULTISTAGE LOGIC NETWORK

\[ D = \sum d_i = \sum g_i h_i + \sum p_i \]

Key Questions:
- How should we size gates to minimize total delay?
- How should we change the topology to minimize delay?

Let's develop some metrics that are independent of sizing:

\[ G = \prod g_i \]
\[ H = \frac{C_{out}}{C_{in}} \]

So for above example:
\[ G = 1 \cdot \frac{5}{3} \cdot \frac{4}{3} \cdot 1 = \frac{20}{9} = 2.22 \]
\[ H = \frac{20c}{10c} = 2 \]

Path effort is the product of stage efforts:
\[ F = \prod f_i = \prod g_i h_i \]
So since stage effort $f = g \cdot h$, does PAM effort $F = G \cdot H$?

Consider simple example:

$$\begin{align*}
F &= T \cdot g \cdot h = 6 \cdot 2 \cdot 2 = 24 \\
G &= 1 \cdot 1 = 1 \\
H &= 5c / 5c = 1 \\
F &= T \cdot g \cdot h = 1 \times 5 \times 1 \times 5 = 5 \\
T &= g \cdot H = 1 \not= 36 \\
G \cdot H &= 1 \not= 36 \\
\text{So in this example } F &= 2G \cdot H \\
\text{we call this the "branching" effort}
\end{align*}$$

The key idea is some of the drive current is directed off the PAM we are analyzing. Recall resistance delay for trees.

Stage effort

<table>
<thead>
<tr>
<th>PAM effort</th>
<th>$T = T_i b_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b = \frac{C_{\text{offPam}} + C_{\text{onPam}}}{C_{\text{onPam}}}$</td>
<td></td>
</tr>
</tbody>
</table>

So in this case $b = \frac{15 + 15}{15} = 2$

So PAM effort

$$F = \prod T \cdot f_i = \prod T \cdot j_h = G \cdot T \cdot H$$

Note that PAM effort depends on circuit topology and loading of entire PAM but not size of transitions in network.

Also, note that PAM effort does not change if add or remove inverters!
Q1: How should we size gates to minimize total delay?

Start with path delay equation:

\[ D = \sum d_i = \sum_1^2 g_i h_i + \sum_1^2 p_i \]

Independent variables are \( h_i \)'s (i.e. internal gate swings). We want to choose \( h_i \)'s to minimize \( D \). So we can take the partial derivative of \( D \) w.r.t. \( h_i \)'s, set to zero, and solve for optimum \( h_i \)'s.

Consider two stage path:

\[ D = (g_1 h_1 + p_1) + (g_2 h_2 + p_2) \]

Note that \( h_1 \) and \( h_2 \) are constrained since \( c_1 \) and \( c_2 \) are given and input cap of gate 2 is output cap for gate 1.

\[ h_1 = \frac{c_2}{c_1}, \quad h_2 = \frac{c_3}{c_2} \]

\[ H = h_1 h_2 = \frac{c_3}{c_1}, \quad H \text{ is a constant since } c_1 \text{ and } c_3 \text{ given} \]

Substitute \( h_2 = H / h_1 \) into delay equation:

\[ D = (g_1 h_1 + p_1) + (\frac{g_2 H}{h_1} + p_2) \]
Take partial derivative with respect to only variable \( h_1 \),
\[
\frac{dD}{dh_1} = g_1 - \frac{g_2 h_1}{h_1^2} = 0
\]
\( g_1 h_1 = g_2 h_1 \)
\( g_1 h_1^2 = g_2 h_1 h_2 \)
\( g_1 h_1 = g_2 h_2 \)
\( f_1 = f_2 \)

Delay is minimized when stage effort is same in both stages!

This generalizes to fans with any number of stages and fans with branching effort.

Fastest design always equalizes effort in each stage.

For a general fan, optimal delay is:
\[
\hat{D} = N F^{1/2} + P
\]

\( \hat{D} \) = Min fan delay for optimal sizing

Method for optimal sizing

1. Calculate fan effort \( F = GBH \)
2. Calculate effort for each stage \( \hat{f} = F^{1/2} \)
3. Estimate min delay of opt sizing \( \hat{D} = N \hat{f} + P \)
4. Starting with last stage, work backwards sizing each gate
\[
\hat{f} = gh = g \left( \frac{C_{out}}{C_{in}} \right) \quad C_{in} = \frac{g}{\hat{f}} C_{out}
\]
Example of Optimal Sizing

\[ F = G \cdot D \cdot H = \left(1 + \frac{3}{4} \cdot \frac{4}{3} \cdot 1\right) \left(1 + \frac{1}{2} \cdot 2 \cdot 1\right) \left(\frac{20 \text{ C}}{10 \text{ C}}\right) \]
\[ = 2.22 \times 1 \times 2 = 8.89 \]

Step 1) \[ \hat{f} = F^{1/\omega} = 1.73 \]

Step 2) \[ \hat{D} = 4 \cdot (1.73) + (1 + 2 + 2 + 1) \]
\[ = 6.92 + 6 = 12.92 \]
\[ \hat{D}_{\text{loss}} = 12.92 \text{ C} \]

Step 4) \[ C_{\text{in}} = \frac{1}{1.73} \times 20 \text{ C} = 11.56 = 11.56 \text{ C} \]
\[ C_{\text{in}} = \frac{4/3}{1.73} \times 11.56 \text{ C} \times 2 = 17.81 \text{ C} \]
\[ C_{\text{in}} = \frac{5/3}{1.73} \times 17.81 \text{ C} = 17.16 \text{ C} \]
\[ C_{\text{in}} = \frac{1}{1.73} \times 17.16 \text{ C} = 9.92 \text{ C} \]

Due to branching! Close to given input cap of 10C.
This assumes we can size gates arbitrarily. In a full custom design, what if using standard cell lib?

Assume standard cell lib with following gates

\[ \text{INV} X_1, \text{INV} X_2, \text{INV} X_3, \text{INV} X_4, \text{INV} X_5, \text{INV} X_6, \text{NAND} X_1, \text{NAND} X_2, \text{NAND} X_4, \text{NOR} X_1, \text{NOR} X_2, \text{NOR} X_4 \]

What does \( X_1, X_2, X_4 \) mean? \( X_2 \) means twice the drive strength of an \( X_1 \) inverter. So \( \text{NAND} X_2 \) means \( \alpha = 2 \)

\[
\begin{aligned}
\text{INV} X_1, & \\
\text{INV} X_2, & \text{NAND} X_1, \\
\text{INV} X_1, & \\
\text{INV} X_2, & \\
\text{NAND} X_1, & \\
\end{aligned}
\]

Assume we have determined optimal sizing is \( C_{\text{inv}} \), how do we figure out which cell to use?

Remember \( g = \frac{V_{th} + C_+}{R_{\text{inv}} C_{\text{inv}}} \) if we assume \( R_{\text{th}} = R_{\text{inv}} \)

\[
g = \frac{C_+}{C_{\text{inv}}} \quad \Rightarrow \quad C_+ = g C_{\text{inv}}
\]

And \( C_{\text{inv}} = \alpha C_+ \)

So \( C_{\text{inv}} = \alpha g C_{\text{inv}} \)

\[ \alpha = \frac{C_{\text{inv}}}{g C_{\text{inv}}} \quad \text{since} \quad C_{\text{inv}} = \frac{1}{g} C_{\text{inv}} \]
Given output $C_m$ from before, what is $\alpha$?

<table>
<thead>
<tr>
<th>$C_m$</th>
<th>$g$</th>
<th>$C_m/(1.3)$</th>
<th>$\alpha$</th>
<th>gate?</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.56 C</td>
<td>1</td>
<td>11.56/(1.3)</td>
<td>3.85</td>
<td>INV x 4</td>
</tr>
<tr>
<td>17.01 C</td>
<td>4/3</td>
<td>17.01/(4/3)</td>
<td>4.95</td>
<td>NA x 4</td>
</tr>
<tr>
<td>17.16 C</td>
<td>5/3</td>
<td>17.16/(5/3)</td>
<td>3.55</td>
<td>N0 x 4</td>
</tr>
<tr>
<td>9.92 C</td>
<td>1</td>
<td>9.92/(1.3)</td>
<td>3.3</td>
<td>INV x 3</td>
</tr>
</tbody>
</table>

Recalculate actual delay given these gates

First calculate actual $C_m$ for each 50 cell gate

<table>
<thead>
<tr>
<th>gate</th>
<th>$C_m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>INV x 4</td>
<td>$C_m = \alpha g 3C = 4.1.3C = 12C$</td>
</tr>
<tr>
<td>N0 x 4</td>
<td>$C_m = \alpha g 3C = 4.4/3 3C = 16C$</td>
</tr>
<tr>
<td>NA x 4</td>
<td>$C_m = \alpha g 3C = 4.5/3 3C = 20C$</td>
</tr>
<tr>
<td>INV x 3</td>
<td>$C_m = \alpha g 3C = 8.1.3C = 9C$</td>
</tr>
</tbody>
</table>

Now use RAM decay equation

$$D = \sum g + \sum p$$

$$= (1 \cdot \frac{12}{16}) + (4/3 \cdot \frac{12 \times 2}{16}) + (2 \cdot \frac{16}{20}) + (1.3) + (1 + 2 + 2 + 1)$$

$$= 1.67 + 2 + 1.33 + 2.22 + 6$$

$$= 12.22 + 6$$

$$= 18.22$$

Compare with actual which is 17.92 off by 2.3%.
Q2: How should we change topology to minimize delay?

Assume we want to implement an eight input AND gate. Calculate minimum delay assuming optimal sizing for following three topologies assuming \( H = 1 \) and \( H = 12 \).

<table>
<thead>
<tr>
<th>topology</th>
<th>( nF^{1/4} )</th>
<th>( \hat{D} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>8-NAND</td>
<td>3.65</td>
<td>9</td>
</tr>
<tr>
<td>2 \times 4-NAND</td>
<td>3.65</td>
<td>9.65</td>
</tr>
<tr>
<td>4 \times 2-NAND</td>
<td>5.25</td>
<td>12.25</td>
</tr>
</tbody>
</table>

\( \hat{D} \) values:
- \( H = 1 \): 12.64, 9, 21.64
- \( H = 12 \): 9.77, 7, 16.77
Optimal number of stages using invariance?

\[ \hat{D} = N \frac{1}{N} + N \hat{D}_{\text{inv}} \]

\[ \frac{d\hat{D}}{dN} = \frac{1}{N} - \frac{1}{N} \ln \left( \frac{1}{N} \right) + \hat{D}_{\text{inv}} = 0 \]

\[ \hat{D}_{\text{inv}} = 0 \]

\[ \frac{1}{N} - \frac{1}{N} \ln \left( \frac{1}{N} \right) = 0 \]

\[ \ln \left( \frac{1}{N} \right) = 1 \]

\[ \frac{1}{N} = e \]

or in other words, \( \hat{d} = e \)

So if we assume \( \hat{D}_{\text{inv}} = 0 \), optimal number of stages results in a stage effort of \( e \) for every stage. Since \( g = 1 \) for an invariance this means \( h = 2.718 \) for every stage.

If \( \hat{D}_{\text{inv}} = 1 \) then we need to solve following nonlinear eq:

\[ \frac{1}{N} - \frac{1}{N} \ln \left( \frac{1}{N} \right) + 1 = 0 \]

let \( \rho = \frac{1}{N} \) when \( \hat{N} \) is optimal

\[ 1 + \rho (1 - \ln \rho) = 0 \]

we can find numerically that \( \rho \approx 3.59 \)

So optimal number of stages results in stage effort of 3.59 when we take into account parasitics. We can roughly approximate 3.59 to be 4.
\[ F^{\frac{1}{3}} \approx 4 \]

\[ \log(F^{\frac{1}{3}}) = \log(4) \]

\[ \frac{1}{3} \log(F) = \log(4) \]

\[ \hat{N} = \frac{\log(F)}{\log(4)} = \log_4(F) \]

So optimal number of stages for inverter chain is roughly:

\[ \hat{N} = \log_4(F) \]

Since \( G=1 \) and \( B=1 \) for inverter chain,

\[ \hat{N} = \log_4(H) \]

Not too bad of an estimate even for realistic values of \( G \) and \( B \) and \( H \) that are not inverters.
Logical effort can help guide us in determining how to size gates and choose a topology to minimize delay but it has many limitations.

To deal with more complicated scenarios we can also write the delay equations for each gate in system and minimize the latest arrival time.

**Example**

Let's write our linear delay equation as a function of $\alpha$.

\[
\begin{align*}
d &= gh + p \\
g &= \frac{P L C_T}{\eta_m c_{in}} \\
c_{in} &= \alpha^2 C_+ \\
c_+ &= \frac{c_w}{\alpha}
\end{align*}
\]
Now write delay equations for each stage:

\[
\begin{align*}
\alpha & = \begin{cases} 1 & \text{if } \alpha_i \leq 3 \frac{g}{s} \text{ and } 3 \frac{g}{s} \leq \alpha_i \leq 3 \frac{d}{s} \\ 1 & \text{if } \alpha_i \leq 3 \frac{g}{s} \text{ and } 3 \frac{g}{s} \leq \alpha_i \leq 3 \frac{d}{s} \\ 1 & \text{if } \alpha_i \leq 3 \frac{g}{s} \text{ and } 3 \frac{g}{s} \leq \alpha_i \leq 3 \frac{d}{s} \\
\end{cases} \\
g & = \begin{cases} 1 & \text{if } g / s \leq t / s \text{ and } t / s \leq 1 \\
t / s & \text{if } g / s \leq t / s \text{ and } t / s \leq 1 \\ 1 & \text{if } g / s \leq t / s \text{ and } t / s \leq 1 \\
\end{cases} \\
c_{\omega} & = 3g \alpha_d = \begin{cases} 3 & \text{if } 3g \alpha_d \leq 3 \frac{d}{s} \\ 4d & \text{if } 3g \alpha_d \leq 3 \frac{d}{s} \\ 7d & \text{if } 3g \alpha_d \leq 3 \frac{d}{s} \\
\end{cases} \\
p & = \begin{cases} 1 & \text{if } 3 \frac{d}{s} \leq p \text{ and } p \leq 3 \\ 2 & \text{if } 3 \frac{d}{s} \leq p \text{ and } p \leq 3 \\ 3 & \text{if } 3 \frac{d}{s} \leq p \text{ and } p \leq 3 \\
\end{cases}
\end{align*}
\]

\[
\begin{align*}
d_0 & = \frac{(4d + 7d_2)}{3d_1} + 1 = \frac{4d_1}{3} + \frac{7}{3} d_2 + 1 \\
d_1 & = \frac{(7d_2)}{3d_1} + 2 = \frac{7d_2}{3d_1} + 2 \\
d_2 & = \frac{(70 + 3d_3)}{3d_2} + 3 = \frac{10}{d_2} + \frac{d_3}{d_2} + 3 \\
d_3 & = \frac{(96)}{3d_3} + 1 = \frac{12}{d_3} + 1
\end{align*}
\]

**Arrival Times**

\[
\begin{align*}
t_0 & = d_0 \\
t_1 & = \max \left( t_0, d_{\omega} \right) + d_1 \\
t_2 & = \max \left( t_0, t_1 \right) + d_2 \\
t_3 & = t_2 + d_3 \\
t_4 & = \max \left( t_0, t_1 \right) + d_2 + d_3 \\
& = \max \left( d_0, \max \left( d_0, d_{\omega} \right) + d_1 \right) + d_2 + d_3
\end{align*}
\]
\( t_3 = \max (d_0, \max (d_0, d_1) + d_1) + d_2 + d_3 \)

Minimize \( t_3 \) subject to above constraints with \( d_1, d_2, d_3 \) as the independent variables.

Actually in summary, we really want to minimize area (or energy) subject to constraint on \( t_3 \).

So we could craft optimization problem to be minimize sum of \( d_1, d_2, d_3 \) (prox for area) subject to constraint:

\[ t_{\text{calc}} > \max \left( d_0, \max (d_0, d_{1\text{calc}}) + d_{1\text{calc}} \right) + d_2 + d_3 \]

Clock-phase constraint
Energy

Energy is a measure of work.

Power is the rate at which work is done.

E-field
Test charge of large electrical potential energy
Small electrical potential energy

Electric potential energy: Capacity for doing work which arises from position of charge in E-field (Joules)

Electric potential: Electric potential energy per unit charge (Volts, \(1V = 1J/C\), \(\Delta V = \Delta E/Q\))

Current: Rate at which charge flows past position (Amperes, \(1A = 1C/s\), \(I = Q/\Delta t\))

Power: Rate at which electric energy is supplied or consumed (Watts, \(1W = 1J/s\), \(P = \Delta E/\Delta t\), \(P = \frac{\Delta V \cdot Q}{Q/\Delta t} = VI\))

\[ E = \int_0^T P(t) \, dt \]
Energy Stored on Cap

\[ E = \int_{0}^{\infty} p(t) dt = \int_{0}^{\infty} v(t) i(t) dt \]

Energy on Capacitor

\[ = \int_{0}^{\infty} v(t) \frac{dQ}{dt} dt = \int_{0}^{\infty} v(t) \frac{C}{dt} dv \]

\[ = C \int_{0}^{\infty} v^{2} dv = \frac{1}{2} C V_{dd}^{2} \]

\[ \int_{0}^{\infty} dV = \frac{1}{2} V_{dd}^{2} \]

So on \( t \rightarrow 0 \), input transition \( \frac{1}{2} C V_{dd}^{2} \) stored on capacitor + this energy is released on \( 0 \rightarrow 1 \) transition.

Energy delivered from Power Supply

\[ E_{supply} = \int_{0}^{\infty} p(t) dt = \int_{0}^{\infty} v_{dd} i(t) dt \]

\[ = v_{dd} \int_{0}^{\infty} \frac{dQ}{dt} dt = v_{dd} \int_{0}^{\infty} C \frac{dv}{dt} dt \]

\[ = C V_{dd} \int_{0}^{\infty} v_{dd} dv = C V_{dd}^{2} \]

\[ \int_{0}^{\infty} dv = \infty \]

Not every dissipated as heat in PMOS

Some will dissipated as heat in NMOS

On Avg each bit transition requires \( \frac{1}{2} C V_{dd}^{2} \)

\(-1 \rightarrow 0 \) uses \( C V_{dd}^{2} \)

\(-0 \rightarrow 1 \) uses \( \phi \)

\[ E_{transition} = \frac{1}{2} C V_{dd}^{2} \]

Probability of bit toggle
Power

\[ P_{tot} = P_{switching} + P_{static} \]

\[ = \alpha f \frac{1}{2} C V_{dd}^2 + V_{dd} I_{off} \]

Note: # transistors, sometimes just # load transistors, but won't be factor of \( \frac{1}{2} \)
Compare Energy

Need to find total switched cap in worst case.

\[
\begin{align*}
C_{\text{inv},g} &= \frac{1}{1.8} \times 10 = 5.6 \\
C_{\text{nao},g} &= \frac{515}{1.8} \times 10 = 283.3 \\
C_{\text{nao},g} &= \frac{2}{1.8} \times 9.3 = 10.3 \\
C_{\text{tot},g} &= C_{\text{inv},g} + 8 \times C_{\text{nao},g} \\
&= 5.6 + 8 \times 10.3 \\
&= 88.8 \text{ C}
\end{align*}
\]

To determine parasitic cap need to understand how gate cap is distributed across transistors.
INVERTER

\[
\begin{align*}
\text{Input} & \quad C_{\text{inv},g} = 5.6 \\
\text{Output} & \quad C_{\text{inv},p} = 5.6
\end{align*}
\]

B INPUT NAND GATE

\[
\begin{align*}
\text{Input} & \quad C_{\text{nan},g} = 10.4 \\
\text{Output} & \quad C_{\text{nan},p} = 24.96
\end{align*}
\]

2 INPUT NOR GATE

\[
\begin{align*}
\text{Input} & \quad C_{\text{nor},g} = 9.3 \\
\text{Output} & \quad C_{\text{nor},p} = 11.16
\end{align*}
\]

4 INPUT NAND GATE

\[
\begin{align*}
\text{Input} & \quad C_{\text{nan},g} = 10.3 \\
\text{Output} & \quad C_{\text{nan},p} = 20.58
\end{align*}
\]
\[ E = \frac{1}{2} C V_{dd}^2 \]

Assume \( d = 0.1 \) and \( V_{dd} = 1 \) V for both, so only

For 8-input NAND topology

\[ C_{\text{tot}} = C_{\text{tot, } g} + C_{\text{tot, } p} = 88.8 + (5.6 + 24.96) = 119.36 \text{ nF} \]

For 4-input NAND topology

\[ C_{\text{tot}} = C_{\text{tot, } g} + C_{\text{tot, } p} = 101 + (11.16 + 20.58) = 132.74 \text{ nF} \]

So second topology requires \( \approx 10\% \) more energy in the worst case where all capacitance is switched. This ignores the energy required for switching the output load.

These energy estimates are in units of \( \text{fJ} \) (IEC, the gate cap for a minimum VLSI transition). What if we want to estimate the absolute energy in joules?

Page 312, Table 8.5 in VLSI Harris

\[ C_{\text{g}} \text{ in IBM 90nm} \approx 1-2 \text{ fF/mm} \quad V_{dd} = 1 \text{ V} \]

Assume \( W_{\text{min}} \approx 9 \lambda \)

\[ W = 9 \lambda \text{ too small, so let's assume } 45 \AA \]

\[ \lambda = 45 \text{ nm} \]

\[ W_{\text{min}} = 9 \cdot 45 = 405 \text{ nm} = 0.45 \text{ mm} \]

Since \( C_{\text{g}} \approx 1-2 \text{ fF/mm} \) and \( W_{\text{min}} \approx 0.45 \text{ mm} \), the \( C = 0.5 \text{ fF} \)

For first topology (assume \( d = 0.1 \))

\[ E = \frac{1}{2} C V_{dd}^2 = \frac{1}{2} (120 \text{ C}) \frac{0.5 \text{ fF}}{C} \text{ (1V)}^2 = 3 \text{ fJ} \]

\[ P = d \frac{1}{2} C V_{dd}^2 = (0.5 \times 10^{-9}) (30 \times 10^{-15}) = 1.5 \mu \text{W} \]
Activity factors

- Previous example just used fixed $d = 0.1$ for all nodes.
- More accurate to track activity factors than topology.

- Assume inputs have completely random data

\[
P_i = \text{Prob node is on}
\]
\[
\bar{P}_i = 1 - P_i = \text{Prob node is off}
\]
\[
\alpha = \bar{P}_i \bar{P}_c + P_i P_c = \sum_i \bar{P}_i \bar{P}_c
\]
\[
\alpha' = \frac{\bar{P}_i P_c}{P_c} = 0.25
\]
\[
\alpha' = \frac{1}{2} \alpha
\]

NAND 2
\[
\alpha_{out} = \bar{P}_{out} P_{out} = (P_A P_D)(1 - P_A P_D)
\]
NAND B
\[
\alpha_{out} = (P^B)(1 - P^B) = (0.0039)(0.996) = 0.0039
\]
Assume $P_A = P_B = ... = P$
NAND 2

\[ \alpha_{out} = \overline{P_a P_b} = (P_a P_b)(1 - P_a P_b) \]

Output is zero.

Output is one if both inputs are one.

Output of NAND 2 is zero if both inputs are one.

Output of NAND 2 is one if any other combination is true.

Assume inputs are random data.

\[ P_a = 0.5 \quad P_b = 0.5 \]

\[ \alpha_{out}' = (P_a P_b)(1 - P_a P_b) = (0.5 \times 0.5)(1 - 0.5 \times 0.5) \]

\[ = (0.25)(1 - 0.25) = 0.25 \times 0.75 = 0.1875 \]

NAND B

\[ \alpha_{out}' = \overline{P_a P_b} = (P_a \overline{P_b})(1 - P_a \overline{P_b}) \]

Output is zero if all B inputs are one.

Output is one if any other combination is true.

\[ \alpha_{out}' = (P_a \overline{P_b})(1 - P_a \overline{P_b}) = (0.5 \overline{P_b})(1 - 0.5 \overline{P_b}) \]

\[ = (0.5 \times 0.49)(1 - 0.5 \times 0.49) = 0.0039 \times 0.496 \]

\[ = 0.0039 \]
<table>
<thead>
<tr>
<th>Vendor</th>
<th>Orbit</th>
<th>HP</th>
<th>AMI</th>
<th>AMI</th>
<th>TSMC</th>
<th>TSMC</th>
<th>TSMC</th>
<th>IBM</th>
<th>IBM</th>
<th>IBM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>MOSIS</td>
<td>MOSIS</td>
<td>MOSIS</td>
<td>MOSIS</td>
<td>MOSIS</td>
<td>MOSIS</td>
<td>MOSIS</td>
<td>IBM</td>
<td>IBM</td>
<td>IBM</td>
</tr>
<tr>
<td>Feature Size $f$</td>
<td>nm</td>
<td>2000</td>
<td>800</td>
<td>600</td>
<td>600</td>
<td>350</td>
<td>250</td>
<td>180</td>
<td>130</td>
<td>90</td>
</tr>
<tr>
<td>$V_{DD}$</td>
<td>V</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3.3</td>
<td>3.3</td>
<td>2.5</td>
<td>1.8</td>
<td>1.2</td>
<td>1.0</td>
</tr>
</tbody>
</table>

**Gates**

| $C_g$ (delay) | fF/μm | 1.77 | 1.67 | 1.55 | 1.48 | 1.90 | 2.30 | 1.67 | 1.04 | 0.97 | 0.80 |
| $C_g$ (power) | fF/μm | 2.24 | 1.70 | 1.83 | 1.76 | 2.20 | 2.92 | 2.06 | 1.34 | 1.23 | 1.07 |
| FO4 Inv. Delay | ps   | 856 | 297  | 230  | 312  | 210  | 153  | 75.6 | 45.9 | 37.2 | 17.2 |

**nMOS**

| $C_d$ (isolated) | fF/μm | 1.19 | 1.11 | 1.14 | 1.21 | 1.63 | 1.88 | 1.22 | 0.94 | 0.89 | 0.76 |
| $C_d$ (shared)  | fF/μm | 1.62 | 1.43 | 1.41 | 1.50 | 2.04 | 2.60 | 1.56 | 1.28 | 1.17 | 1.28 |
| $C_d$ (merged)  | fF/μm | 1.48 | 1.36 | 1.19 | 1.24 | 1.60 | 2.16 | 1.41 | 1.40 | 1.51 | 1.20 |
| $R_n$ (single)  | kΩ · μm | 30.3 | 10.1 | 9.19 | 11.9 | 5.73 | 4.02 | 2.69 | 2.54 | 2.35 | 1.34 |
| $R_n$ (series)  | kΩ · μm | 22.1 | 6.95 | 6.28 | 8.59 | 4.01 | 3.10 | 2.00 | 1.93 | 1.81 | 1.13 |
| $V_{th}$ (const.) | V | 0.65 | 0.65 | 0.70 | 0.70 | 0.59 | 0.48 | 0.41 | 0.32 | 0.32 | 0.31 |
| $V_{th}$ (linear ext.) | V | 0.65 | 0.75 | 0.76 | 0.76 | 0.67 | 0.57 | 0.53 | 0.43 | 0.43 | 0.43 |
| $I_{out}$      | μA/μm | 152 | 380  | 387  | 216  | 160  | 150  | 75.6 | 45.9 | 37.2 | 17.2 |
| $I_{eff}$      | μA/μm | 2.26 | 9.36 | 2.21 | 1.45 | 6.57 | 56.3 | 93.9 | 1720 | 4000 | 33400 |
| $I_{gate}$     | μA/μm | n/a | n/a  | n/a  | n/a  | n/a  | n/a  | n/a  | n/a  | 1.22 | 3620 |

**pMOS**

| $C_d$ (isolated) | fF/μm | 1.42 | 1.17 | 1.31 | 1.42 | 1.89 | 2.07 | 1.24 | 0.94 | 0.74 | 0.73 |
| $C_d$ (shared)  | fF/μm | 1.92 | 1.62 | 1.73 | 1.86 | 2.37 | 2.89 | 1.79 | 1.56 | 1.25 | 1.25 |
| $C_d$ (merged)  | fF/μm | 1.52 | 1.23 | 1.35 | 1.43 | 1.83 | 2.40 | 1.56 | 1.41 | 1.16 | 1.18 |
| $R_p$ (single)  | kΩ · μm | 67.1 | 26.7 | 19.9 | 29.6 | 16.1 | 8.93 | 6.51 | 6.39 | 5.47 | 2.87 |
| $R_p$ (series)  | kΩ · μm | 53.9 | 21.4 | 15.4 | 23.6 | 13.3 | 6.91 | 5.41 | 5.48 | 4.92 | 2.42 |
| $V_{th}$ (const.) | V | 0.72 | 0.91 | 0.90 | 0.90 | 0.83 | 0.46 | 0.43 | 0.33 | 0.35 | 0.33 |
| $V_{th}$ (linear ext.) | V | 0.71 | 0.94 | 0.93 | 0.93 | 0.88 | 0.52 | 0.51 | 0.42 | 0.43 | 0.42 |
| $I_{out}$      | μA/μm | 70.5 | 154  | 215  | 99.0 | 181  | 245  | 228  | 177  | 187  | 360 |
| $I_{eff}$      | μA/μm | 2.18 | 1.57 | 2.08 | 1.38 | 2.06 | 30.1 | 25.2 | 1330 | 2780 | 19500 |
| $I_{gate}$     | μA/μm | n/a | n/a  | n/a  | n/a  | n/a  | n/a  | n/a  | 0.06 | 1210 | 2770 |

The gate capacitance for delay held steady near 2 fF/μm for many generations, as scaling theory would predict, but abruptly dropped after the 180 nm generation. The gate capacitance for power is slightly higher than that for delay as discussed in Section 8.4.3.

The FO4 inverter delay has steadily improved with feature size as constant field scaling predicts. It fits our rule from Section 4.4.3 of one third to one half of the effective channel length, when delay is measured in picoseconds and length in nanometers.

Diffusion capacitance of an isolated contacted source or drain has been 1–2 fF/μm for both nMOS and pMOS transistors over many generations. The capacitance of a shared contacted diffusion region is slightly higher because it has more area and includes two gate overlaps. The capacitance of the merged diffusion reflects two gate overlaps but a smaller diffusion area. Half the capacitance of the shared and merged diffusions is allocated to each of the transistors connected to the diffusion region.
During lecture today, I mentioned that _adding_ inverters can sometimes _reduce_ the path delay. This might seem counter intuitive based on what you learned in ECE 2300. In ECE 2300, gates had a _constant_ delay. So every inverter might always have a delay of 1 \( \tau \), and every NAND2 gate might always have a delay of 2 \( \tau \). In fact, we used a similar simplification when estimating the critical path in ECE 4750. If we assume a constant delay model, then adding a pair of inverters would indeed _always_ slow down the path delay. Adding a pair of inverters would simply increase the total propagation delay.

Based on what we have learned in ECE 5745 so far, it should be clear that the constant delay model is a significant oversimplification. The delay of a gate depends on many things including its size, the load capacitance at the output, when inputs arrive, the rise/fall time of the inputs, layout details, etc. Our RC modeling and method of logical effort use a _linear_ delay model which is a little more reasonable than a _constant_ delay model (but of course is still a significant simplification!). So the delay of a gate is:

\[
d = gh + p
\]

The logical effort \((g)\) and the parasitic delay \((p)\) depend only on the template, while the electrical effort \((h)\) depends on both the size of the gate \((C_{\text{in}})\) and the load capacitance at the output \((C_{\text{out}})\).

Let’s look in more detail at the example we were discussing in lecture to demonstrate how _adding_ inverters can sometimes _reduce_ the path delay. Assume after synthesis we have the following two-gate path:

```
--|NAND
|  |NAND---I>o----.---
--|NAND          |
  4C              --- 1000C
NAND2X1 INVX4  ---  
   |  
  V
```

So we have a X1 two-input NAND gate (NAND2X1) and a X4 inverter driving a load of 1000\(C\). The synthesis tool optimized the design assuming the inverter was driving a modest load, but after place-and-route, it turned out that the inverter has to drive a cross-chip global wire and thus a very large fixed capacitance.

What is the delay of this two-gate path?

\[
D = \left( g_0h_0 + g_1h_1 \right) + \left( p_0 + p_1 \right)
= \left( \frac{4}{3} \times 12/4 + 1 \times 1000/12 \right) + \left( 2 + 1 \right)
= \left( 4 + 83.3 \right) + 3
= 90.3 \tau
\]

Recall that the minimum delay will occur when the stage effort is equal across all stages. Notice that the stage effort of the two stages is not even close to being equal which suggests this sizing is suboptimal.

The place-and-route tool can potentially reduce the path delay using "buffer resizing". So let’s assume the tool wants to increase the size of the inverter. Let’s use logical effort to figure out the optimal sizing.

\[
F = GHB = \frac{4}{3} \times \frac{1000}{4} \times 1 = 333
\]
\[
f' = F'^{(1/N)} = (333)^{(1/2)} = 18.25
\]
\[
D' = N*F'^{(1/N)} + P = 2*18.25 + \left( 2 + 1 \right) = 39.5 \tau
\]
Note that I am using \( f' \) instead of \( f \) "hat" and \( D' \) instead of \( D \) "hat". The path delay is significantly lower if we can resize the inverter. Let’s figure out how large the final inverter needs to be to achieve this optimal delay.

\[
C_{in,1} = \frac{(g/f')}{C_{load}} = \frac{(1/18.25)}{1000} = 54.8C
\]

That is a pretty big inverter! The inverter’s NMOS would be 18.27 times the minimum width and the inverter’s PMOS would be 36.53 times the minimum width. Assume our standard cell library has an INVX1, INVX2, INVX4, INVX8, INVX16, INVX32, and INVX64. Let’s choose the INVX16 for the final inverter (which is a little smaller than the optimal full-custom sizing).

\[
\begin{align*}
\text{--} & \quad \text{NAND} \\
\text{--} & \quad \text{NAND---I\rightarrow o----I\rightarrow o----I\rightarrow o----I\rightarrow o----.---} \\
\text{--} & \quad \text{NAND} \\
4C & \quad \text{--- 1000C} \\
\text{NAND2X1} & \quad \text{INVX16 ---} \\
\text{V} &
\end{align*}
\]

What is the new delay of the new path?

\[
D = (\frac{g_0*h_0 + g_1*h_1}{p_0 + p_1}) + (\frac{p_0 + p_1 + p_2 + p_3}{2 + 1}) = (16 + 20.8) + 3 = 39.8 \tau
\]

The delay using the standard cell is a little slower and the stage effort is not exactly balanced, but buffer resizing does still significantly reduce the path delay.

The place-and-route tool can potentially further reduce the path delay using "buffer insertion". Let’s quickly estimate the optimal number of stages.

\[
\log_4(F) = \log_4(333) = 4.2
\]

So the rough estimate of the optimal number of stages is 4, but we are only using two stages. Let’s add two INVX1 gates at the end of the path to see if that helps.

\[
\begin{align*}
\text{--} & \quad \text{NAND} \\
\text{--} & \quad \text{NAND---I\rightarrow o----I\rightarrow o----I\rightarrow o----I\rightarrow o----.---} \\
\text{--} & \quad \text{NAND} \\
4C & \quad \text{--- 1000C} \\
\text{NAND2X1} & \quad \text{INVX16 INVX1 INVX1 ---} \\
\text{V} &
\end{align*}
\]

\[
D = (\frac{g_0*h_0 + g_1*h_1 + g_2*h_2 + g_3*h_3}{p_0 + p_1 + p_2 + p_3}) + (\frac{p_0 + p_1 + p_2 + p_3}{2 + 1 + 1 + 1}) = (16 + 0.0625 + 1 + 333.3) + 5 = 355 \tau
\]

Yeow -- this is a bad idea. The delay is 9x worse! Instead of driving the large load capacitance with an INVX16, now we are driving this large load capacitance with an INVX1. Very bad idea. What if we add two more INVX16 gates at the end of the path?
This is better than the original design, but slower than the optimized two-gate design with buffer resizing. The key is that we don’t want to add more inverters. We want to add more inverters and then properly resize the gates to ensure we are balancing the stage efforts appropriately. We can just use the method of logical effort to find the optimal delay and the optimal sizing.

\[ F = GHB = \frac{4}{3} \times \frac{1000}{4} \times 1 = 333 \]

\[ f' = F^{(1/N)} = (333)^{(1/4)} = 4.27 \]

\[ D' = N \times F^{(1/N)} + P = 4 \times 4.27 + (2 + 1 + 1 + 1) = 22.1 \text{ tau} \]

So we have further reduced the delay by using a combination of buffer insertion and buffer resizing. Let’s figure out how large each inverter needs to be to achieve this optimal delay.

\[ C_{in,3} = (g/f') \times C_{load} = (1/4.27) \times 1000 = 234C \]

\[ C_{in,2} = (g/f') \times C_{in,3} = (1/4.27) \times 234 = 55C \]

\[ C_{in,1} = (g/f') \times C_{in,2} = (1/4.27) \times 55 = 13C \]

Yeow -- that is a big final inverter! Our INVX64 has a C_in of 192C so that will be the best we can do. Let’s size our inverters as follows:

\[ D = (g0*h0 + g1*h1 + g2*h2 + g3*h3) + (p0 + p1 + p2 + p3) \]

\[ = (4/3 \times 48/4 + 1 \times 96/48 + 1 \times 192/96 + 1 \times 1000/192) + (2 + 1 + 1 + 1) \]

\[ = (16 + 2 + 2 + 5.2) + 5 \]

\[ = 27.2 \text{ tau} \]

The original design without buffer insertion/resizing had a delay of 90.3 tau while the new design with buffer insertion/resizing has a delay of only 27.2, and improvement of 3.3x! So clearly _adding_ inverters can _reduce_ the path delay, but (as we saw with some of our counter-examples) this is only true if you properly size the gates!