Speed Optimization via Gate Sizing

- Gate sizing basics
  - P:N ratio
  - Complex gates
  - Velocity saturation
  - Tapering

- Developing intuition
  - Number of stages vs. fanout
  - Popular inverter chain example

- Formal approach: logical effort

- Sizing optimization for speed
Basic Gate Sizing Relationships

- Rise and fall delays are determined by the pull-up and pull-down “strength”
  - Besides the dimensions, strength depends on $\mu$, $C_{OX}$, $V_T$
  - PMOS is weaker because of lower $\mu_P$
    * Larger P network than N network

- Increasing size of gate can reduce delay
  - Inverse ($1/W$) relationship with resistance (and hence delay)
  - BUT it can slow down the gate driving it
    * Proportional ($W$) relationship with Capacitance. So be careful!

P:N Ratio for “Equal” Rise and Fall Delay

- Good to have roughly equal delays for different transitions
  - Don’t need to worry about a worst-case sequence
  - Size P’s to compensate for mobility
    * $C_{OX}$, $V_T$, $L$ are roughly the same.
    
    $R_{DR} \propto 1/I \propto 1/\mu W$

  - Make the Pull-up and Pull-down resistances equal
    * $R_P/R_N = 1 = \mu_P W_P/\mu_N W_N = k\beta, k = \text{mobility ratio, } \beta = P:N \text{ ratio}$
    * $W_P/W_N = \mu_N/\mu_P$

- Approximately the same as making $V_{THL} = V_{DD}/2$

- Easy for an inverter
  - What about more complex gates?
Complex Gate Sizing

- N-stack series devices need N times lower resistance
  - N×Width
- Make worst case strength of each path equal
  - Multi-input transition can result in stronger network
- Long series stacking is VERY bad

Ex: $\beta = 2$

Accounting for Velocity Saturation

- Series stacking is actually less velocity saturated
  - If we use $R_{\text{no_stack}} = \frac{4}{3}R_{\text{stack}}$
  - Adjust the single device size to account for velocity saturation

Ex: $\beta = 2$
**P:N Ratio for Minimum Delay**

- Delay of an inverter chain (2 inverters) to include \( t_{PLH} \) & \( t_{PHL} \).
  
  ![Diagram of two inverters](image)

- Let \( R_{PDRV} \sim R_{O'}/W_{P}\mu_{P} \), \( R_{NDRV} \sim R_{O'}/W_{N}\mu_{N} \), \( C_{G} \sim C_{O}(1+W_{P}/W_{N}) \).

- \( t_{PD} = t_{D1} + t_{D2} = R_{O'}(1/W_{P}\mu_{P} + 1/W_{N}\mu_{N}) \cdot C_{O}(1+W_{P}/W_{N}) \).

- \( \tau_{N}(1+1/k\beta)(1+\beta) \).

- Min\( (t_{PD}); dt_{PD}/d\beta = 0 = \tau_{N}(1-k/\beta^{2}) \).

- So \( \beta = W_{P}/W_{N} = \sqrt{\mu_{N}/\mu_{P}} \).

*Intuition is that since NMOS has more drive for a given size, it is better to use more NMOS.*

---

**Plot of P:N Ratio to Delay**

- Normalized Delay to that of an Inverter driving another inverter with 4x the size (fanout of 4).
  
  - Where \( \mu \) = mobility ratio, and \( \beta \) is P:N ratio.
  
  - Curve is relatively flat so not a strong delay tradeoff.

![Plot of P:N Ratio to Delay](image)
Tapering

- One observation from Elmore delay is that capacitance closer to the v-source has less effect on delay:
  - \[ \tau_{\text{delay}} = R_1(C_1) + (R_1 + R_2)(C_2) \]
    - C_1 has less effect on delay than C_2

- So taper stacked devices to speed them up:
  - Make the bottom ones bigger
    - R_1 (many occurrences) has less resistance
    - C_3 (multiplying larger R) has smaller capacitance

In reality, tapering doesn’t win as much because layout is less compact when stacking unequal sized transistors (causing more C)

Delay of an N-Input Function: AND Example

- Series stacking results in larger devices without improving drive strength:
  - Greater self-loading capacitance
  - We expect that with large number of inputs, it is no longer better to build bigger gates

- Comparison (approximate):
  - 1 N-input NAND gate driving an inverter
  - 2 N/2-input NAND gates driving a NOR gate (to combine)
  - Drive the same output load

Let’s analyze building blocks: NAND, NOR, INV
Delay of an N-Input NAND

- Assume
  - \( C_{\text{GNM}} = C_{\text{DNM}} = C_0 fF/\mu m \)
  - NMOS Resistance = \( R_0 \Omega/\mu m \)
- NAND, NMOS size is \( NW_N/f \)
  - For \( N \) inputs, \( R_1 = R_2 = \ldots = R_O/(NW_N/f) \)
  - \( C_1 = C_2 = \ldots = C_N = NC_0W_N/f \)

\[
t_{\text{pNAND}} = \sum_{i=1}^{N-1} iR_0C_i + R_0C_0(1 + NC_0/2f) + C_{\text{gate}}fW_N.
\]

Let \( \beta = 2 \) (\( t_{\text{sh}} = t_{\text{sh}} \) for simplified analysis),

\[
t_{\text{pNAND}} = R_0C_0 \left( \frac{N(N-1)}{2} + N \right) + R_0C_0 \left( 2N + \frac{C_{\text{gate}}f}{C_0W_N} \right).
\]

Delay of Inverter and NOR

- Inverter
  - \( R_{\text{INV}} = R_0/W_N \)
  - \( C_{\text{INV}} = C_{\text{DIFF}} + C_{\text{GATE}} = C_0(W_n(1+\beta) + f W_n(1+\beta)) \)
  - For \( \beta = 2 \), \( t_{\text{INV}} = R_0C_0(3+3f) \)
  - \( C_{\text{gate,INV}} = C_0(W_n(1+\beta)) \) – Input capacitance of inverter

- NOR2
  - \( R_{\text{NOR}} = R_0/W_N \)
  - \( C_{\text{NOR}} = C_{\text{DIFF}} + C_{\text{GATE}} = C_0(W_n(2+2\beta) + f W_n(1+\beta)) \)
  - For \( \beta = 2 \), \( t_{\text{NOR}} = R_0C_0(6+3f) \)
  - \( C_{\text{gate,NOR}} = C_0(W_n(1+2\beta)) \) – Input capacitance of NOR2
Comparison

**N-input NAND and Inverter**

\[ t_{p_{\text{NAND}}} = R_0 C_0 \left( \frac{N(N-1)}{2} + N \right) + R_0 C_0 \left( 2N + \frac{C_{\text{G}} f}{C_{\text{b}} W_N} \right) \]

\[ t_{p_{\text{NAND}}} = R_0 C_0 \left( \frac{N(N-1)}{2} + N \right) + R_0 C_0 \left( 2N + 3f \right) \]

\[ t_{p_1} = R_0 C_0 \left( \frac{N(N-1)}{2} + 3N + 3f \right) + R_0 C_0 \left( 3f + 3 \right) \]

**N/2-input NAND and NOR**

\[ t_{p_{\text{NAND}}} = R_0 C_0 \left( \frac{N(N/2-1)}{4} + \frac{3N}{2} + \frac{C_{\text{G}} f}{C_{\text{b}} W_N} \right) \]

\[ t_{p_{\text{NAND}}} = R_0 C_0 \left( \frac{N(N/2-1)}{4} + \frac{3N}{2} + 5f \right) \]

\[ t_{p_2} = R_0 C_0 \left( \frac{N(N/2-1)}{4} + \frac{3N}{2} + 5f \right) + R_0 C_0 \left( 3f + 3 \right) \]

Crossover at \( N = \frac{1}{2} \) with \( f = 4 \)
(Nota that the comparison has unequal input C)

---

**Table of Comparison**

- **\( N = 4 \)**
  - \( t_{p_1} = 21 + 6f \) (45 for \( f = 4 \))
  - \( t_{p_2} = 13 + 8f \) (45 for \( f = 4 \))

- **\( N = 6 \)**
  - \( t_{p_1} = 36 + 6f \) (60 for \( f = 4 \))
  - \( t_{p_2} = 21 + 8f \) (53 for \( f = 4 \))

- **\( N = 8 \)**
  - \( t_{p_1} = 55 + 6f \) (79 for \( f = 4 \))
  - \( t_{p_2} = 30 + 8f \) (62 for \( f = 4 \))

*It does not make sense in delay to build large-fan-In static CMOS gates of fan-in greater than 4!*
Transmission Gate Sizing

- Attempt to make a T-gate have equal pull-up and pull-down resistance

- P:N ratio of k is not good for delay:
  - NMOS still has some significant pull-up strength (even if not all the way to Vdd)
  - PMOS has some pull-down (but very weak)

- Using some common numbers
  - \( R_{N,DOWN} = R_O k \mu m \), \( R_{N,UP} = 2 R_O k \mu m \) (2x penalty for weak transition)
  - \( R_{P,UP} = 2.5 R_O k \mu m \), \( R_{P,DOWN} = 5 R_O k \mu m \) (2x penalty for weak transition)
  - Let's try \( W_P = W_N \)
    - Parallel Up, \( R_{T,G,UP} = R_{N,UP} || R_{P,UP} = 1.1 R_O \)
    - Parallel Down, \( R_{T,G,DN} = R_{N,DOWN} || R_{P,DOWN} = 0.83 R_O \)
  - So, using \( W_P/W_N = 1 \) is fairly reasonable
  - Actual size may depend on the process technology

Delay Analysis (So Far) – Summary

- The capacitance and resistance of the devices determine the performance of the circuit

- Elmore Delay approximation gives initial insight into design
  - Step response, does not account for signal slopes

- The sizing of the transistors (a first glimpse)
  - Determines the logical threshold
  - Determines the drive strength of the gate as well as the load it presents to the preceding gate which effects the delay
  - Determines the capacitance and hence power dissipated by the gate

- Large fan-in gates imply large self-loading and gate loading to the preceding gate
  - Better to split into 2 gates when fan-in is greater than 4
Speed Optimization via Gate Sizing

- Gate sizing basics
  - P:N ratio
  - Complex gates
  - Velocity saturation
  - Tapering

- Developing intuition
  - Number of stages vs. fanout
  - Popular inverter chain example

- Formal approach: logical effort

- Sizing optimization for speed

Problem Statement

- Given:
  - An arbitrary logical function
  - A given implementation

- How do we decide the relative size of each gate?

- Constraints
  - $C_{out}$ (load), $C_{in}$ (load presented to input), and maximum delay
Simplified Problem: Buffering

- Assume
  - $\beta$ (P:N ratio) = $\mu$ (mobility ratio).
  - $I_{PSAT} = I_{NSAT}$
  - $R_0 = $ Pull down for NMOS with size $W_0$ or PMOS with size $\beta W_0$
  - $C_0 = $ Gate capacitance of N+PMOS of size $W_0$, $\beta W_0$
  - * Ignore Source/Drain & Wire Capacitance
  - $\tau_0 = R_0 C_0$

- Goal: sizes each of the $N$ stages for minimum delay.
  - Delay for stage 1: $\alpha_1 C_0 R_0 = \alpha_1 \tau_0$
  - Delay for stage 2: $\alpha_2 C_0 R_0 / \alpha_1 = \frac{\alpha_2}{\alpha_1} \tau_0$

Optimal Fanout

- Fanout of each stage of the inverter chain
  - Stage 1 = $\alpha_1$, Stage 2 = $\frac{\alpha_2}{\alpha_1}$

- 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 $C_{out} = C_0 \alpha_0^N$

- Total Delay = Sum (Delay of stage 1-N)
  - Delay = $\tau_0 N \alpha_0$

- Since $C_{in} = C_0$ (remember: both $C_{in}$ and $C_{out}$ are given)
  - $\frac{C_{out}}{C_{in}} = \frac{\alpha_0 N}{\alpha_0^N}$

- For a given $N$, what is the optimal $\alpha_0$
  - $\alpha_0 = \sqrt[2N]{\frac{C_{out}}{C_{in}}}$
Optimum Number of Stages

- For an arbitrary $N$
  \[
  N = \frac{\ln \left( \frac{C_{\text{out}}}{C_{\text{in}}} \right)}{\ln(\alpha_0)}
  \]

\[
\frac{d \text{Delay}}{d \alpha_0} = \frac{d}{d \alpha_0} \left[ \tau_0 \frac{\alpha_0}{N \ln(\alpha_0)} \ln \left( \frac{C_{\text{out}}}{C_{\text{in}}} \right) \right] = 0
\]

\[
\frac{d \text{Delay}}{d N} = \sqrt{\frac{C_{\text{out}}}{C_{\text{in}}}} - \sqrt{\frac{C_{\text{out}}}{C_{\text{in}}} \cdot \frac{\ln \left( \frac{C_{\text{out}}}{C_{\text{in}}} \right)}{N}}
\]

- Min Delay
  \[\alpha_0 = e\]

Optimum buffer fanout is $e$ (2.718) when the self-loading is neglected

Constant Fanout Per Stage?

- Intuition: what if we increase the size of 1 stage by $(1+\Delta)$
  - $R_{\text{drv}}$ reduces $1/(1+\Delta)$
  - $C_{\text{load}}$ (previous stage) increases by $(1+\Delta)$
  - Delay is summed and $R$ reduces less quickly than $C$ increases
    * So delay would increase if we deviate.

- Mathematically:
  - Delay = $\tau_0(\alpha_1 + \alpha_2/\alpha_1 + \alpha_3/\alpha_2 + \alpha_4/\alpha_3 + \ldots)$
  - $d\text{Delay}/d\alpha_1 = 0$
    * $d\text{Delay}/d\alpha_1 = \tau_0(1 - \alpha_2/\alpha_1^2)$
    * So $\alpha_1^2 = \alpha_2$
  - $d\text{Delay}/d\alpha_2 = 0$
    * $d\text{Delay}/d\alpha_2 = \tau_0(1/\alpha_1 - \alpha_3/\alpha_2^2)$
    * So $\alpha_2^2 = \alpha_1 \alpha_3$, thus $\alpha_1^3 = \alpha_3$
Optimal Buffering with Self Loading

- **Intuition: without self loading**
  - Delay decreases proportionally with decreasing the number of stages
  - But increasing fanout increases delay proportionally
  - The two are equal at the optimum # of stages and fanout

- **Intuition: with self loading**
  - Increasing fanout no longer increases delay proportionally
    * Delay = \( R_0 (\alpha C_0 + C_{sd}) \)
  - New optimum number of stages would be less and fanout is bigger

- All equations remain the same except Delay
  \[
  \frac{d \text{Delay}}{d \alpha_0} = \frac{d}{d \alpha_0} \left[ \tau_0 \left( \frac{C_{sd}}{C_0} + \frac{C_{sd}}{C_0} \right) \ln \left( \frac{C_{sd}}{C_0} \right) \right] = 0
  \]
  \[
  \alpha_0 = e^{1+p/\alpha C_0} \Rightarrow \quad p = \frac{C_{sd}}{C_0}
  \]

Optimal Buffering as \( fn \) (Self-Loading)

- The optimum changes with self loading
- A reasonable number to use for optimal delay is fanout of 4

![Graph showing delay versus fanout with different equations labeled for each fanout value]
Buffer Optimization for Energy-Delay

- Optimizing for Energy (Power) doesn’t make sense because the optimum will be the smallest possible device size
- Instead optimize for the best Energy-Delay tradeoff
  - Assuming constant fanout

![Graph showing Energy-Delay versus Fanout]

Assuming FO is constant, $\alpha_0$

Results in larger FO...

FO = 5 is pretty reasonable

Issue with Optimal Energy-Delay

- Constant fanout is not a good assumption
- Intuition:
  - Reduce a lot of power by reducing the size of the final driver
    - Large fanout at the last stage
  - Reduce fanout of prior stages to compensate
- Example: $C_{in}=1$, $C_{out}=1000$
  - Equal Fanout Result:
    - 4 stages
  - Unequal FO (tapered FO)

<table>
<thead>
<tr>
<th></th>
<th>Stage 1</th>
<th>Stage 2</th>
<th>Stage 3</th>
<th>Stage 4</th>
<th>EDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equal FO = 5.62</td>
<td>1</td>
<td>5.62</td>
<td>31.6</td>
<td>177.8</td>
<td>3220</td>
</tr>
<tr>
<td>Unequal FO</td>
<td>1</td>
<td>4.8 (4.8)</td>
<td>23.1 (4.9)</td>
<td>124.5 (5.4)</td>
<td>31100</td>
</tr>
</tbody>
</table>
Ultimately, We will get Here (~Lecture 8)

- Energy-Delay Optimization
  - Gate size, Supply Voltage, Threshold Voltage

![Energy-Delay Optimization Diagram]

Application of Fanout to Logic?

- When logic needs to drive a large capacitive load:
  - Fanout ~ 4
  - What is fanout?
    - Effective Load capacitance driven by the Gate
      - Normalized by the capacitance of the equivalent inverter
  - Example:
    - NAND gate \( W_P=5, W_N=5 \) driving 5 equal NAND gates
    - Equivalent Inverter: \( W_P=5, W_N=2.5 \); Total Gate width = 7.5
    - Total Load Gate Width = \( 5 \times 10 = 50 \)
      - Fanout = 6.6
    - Try to reorganize logic and add inverters so fanout ~4
- When logic has large N so each stage drives small fanouts:
  - Delay is logic limited so reduce N
  - Balance Fanout so that they are equal
- OK, but not very systematic…
Speed Optimization via Gate Sizing

- Gate sizing basics
  - P:N ratio
  - Complex gates
  - Velocity saturation
  - Tapering

- Developing intuition
  - Number of stages vs. fanout
  - Popular inverter chain example

- Formal approach: logical effort

- Sizing optimization for speed

Concept of Logical Effort

- Instead of running lots of simulations
  - Simplified: (almost) back-of-envelope calculations of delay

- Basic concept:
  - Delay = \( R_{gate}(C_{load} + C_{self}) = R_{gate}C_{load} + R_{gate}C_{self} \)
  - Logical Effort basic equation: \( d = f + p \)
    - \( d \) is the delay (normalized)
    - \( f \) is known as the effort delay
    - \( p \) is known as the 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 (no self load)
    - With \( R_0 = R_{gate} \), \( d \) = fanout + normalized parasitic
    - So \( f \) is essentially equivalent to fanout
    - \( d \) is a measure that is independent of process, voltage, temp
The Logical Effort Way of Thinking...

- Gate delay we used up to now:
  \[ \text{Delay} = 0.69 R_{\text{gate}} (C_{\text{parasitic}} + C_{\text{out}}) \]

- Another way to write this formula is:
  \[ \text{Delay} = 0.69 R_{\text{gate}} C_{\text{in, gate}} \left( \frac{C_{\text{parasitic}}}{C_{\text{in, gate}}} + \frac{C_{\text{out}}}{C_{\text{in, gate}}} \right) \]

  \[ \text{Delay} = \tau_{\text{gate}} \left( \gamma_{\text{gate}} + \frac{C_{\text{out}}}{C_{\text{in, gate}}} \right) \]

Now Normalize the Delay

Strategy: normalize to the time constant of an inverter

- Approach 1: normalize to fictitious “technology time constant”
  \[ \frac{\text{Delay}}{\tau_{\text{INV}}} = \frac{\tau_{\text{gate}}}{\tau_{\text{INV}}} \left( \gamma_{\text{gate}} + \frac{C_{\text{out}}}{C_{\text{in, gate}}} \right) \]

- Approach 2: normalize to intrinsic delay of inverter
  \[ \frac{\text{Delay}}{\tau_{\text{\theta,INV}}} = \frac{\tau_{\text{gate}}}{\tau_{\text{\theta,INV}}} \left( \gamma_{\text{gate}} + \frac{C_{\text{out}}}{C_{\text{in, gate}}} \right) \]

Both formulations exist in the literature
We use approach 1 from the original logical theory
Doesn’t really matter – it’s just a constant
Normalized Delay

**Strategy:** normalize to a time constant of an inverter
- Approach 1: normalize to fictitious “technology time constant”

\[
\frac{\text{Delay}}{\tau_{\text{INV}}} = \frac{\tau_{\text{gate}}}{\tau_{\text{INV}}} \left( \gamma_{\text{gate}} + \frac{C_{\text{out}}}{C_{\text{in,gate}}} \right)
\]

- Normalized delay: \( d = g (\gamma_{\text{gate}} + h) \)
- Even simpler: \( d = g \cdot h + p \)

Logical effort terms
- Logical effort \((g)\)  Electrical fanout \((h)\)  Parasitic delay \((p)\)

\[
g = \frac{R_{\text{gate}} \cdot C_{\text{in,gate}}}{R_{\text{INV}} \cdot C_{\text{in,INV}}} \quad h = \frac{C_{\text{out}}}{C_{\text{in,gate}}} \quad p = \frac{C_{\text{par,gate}}}{C_{\text{par,INV}}} \cdot \gamma_{\text{INV}}
\]

Intuition
- Logical effort \((g)\)
  - \(R_{\text{on}}\) ratio for equal \(C_{\text{in}}\)
  - \(C_{\text{in}}\) ratio for equal \(R_{\text{on}}\)
- Electrical fanout \((h)\)
  - \(C_{\text{out}} / C_{\text{in}}\) ratio (gate cap only, diffusion counts in the \(p\) term)
- Parasitic delay \((p)\)
  - Ratio of parasitic capacitances for equal \(R_{\text{on}}\)

\[
p = g \cdot \gamma_{\text{gate}} = \frac{RC_{\text{in,gate}}}{RC_{\text{in,INV}}} \cdot \frac{C_{\text{par,gate}}}{C_{\text{par,INV}}} = \frac{C_{\text{par,gate}}}{C_{\text{par,INV}}} \cdot \gamma_{\text{INV}}
\]
Calibrating the Model

- The values for $g$ and $p$ can be extracted from simulation
  - Because, $d = g*h + p$
  - Simulating the delay of the gate for different loads
    - Drive itself with different multiplication factor
  - Extract $\tau$ using inverter with no self-loading ($AS, AD, PS, PD = 0$)
  - Vary the inputs (and rise/fall) for different $g$ and $p$

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.

  - $f = \frac{R_{\text{gate}} C_{\text{load}}}{R_0 C_0}$, $p = \frac{R_{\text{gate}} C_{\text{self}}}{R_0 C_0}$
  - Let $R_0 = R_{\text{inv}}$, where $R_{\text{inv}} = R_{\text{gate}}$ and $C_0 = C_{\text{inv}}$
  - $p = \frac{C_{\text{self}}}{C_{\text{inv}}}$, $f = \frac{C_{\text{in}} C_{\text{load}}}{C_{\text{in}} C_{\text{inv}}}$
    * $C_{\text{in}}$ 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.
**Typical Simulation Data (\*)**

\( (*) \text{ assumes } \gamma_{INV} = 1 \)

\[
\text{Normalized delay: } \frac{d}{2} = \frac{g}{g} + \frac{p}{p} = \frac{d}{d} + \frac{h}{h} + 1
\]

\[

g = \frac{4}{3} \\
p = 2 \\
d = (\frac{4}{3})h + 2
\]

\[

g = 1 \\
p = 1 \\
d = h + 1
\]

Electrical effort: \( g = \frac{C_{out}}{C_{in}} \)

\[ d_{gate} = gh + p = \text{effort delay} + \text{parasitic delay} \]

---

**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 a single device that has equivalent drive strength as a gate’s pull-up of that input.
  4. For a reference inverter with Equal Rise/Fall, \( \beta = \mu \), with \( W_p \) from Step 3, determine the total gate widths of the inverter devices.
  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 gate is not \( \mu \).
**Calculating Logical Effort – Simple Example**

**DEF:** Logical effort is the ratio of the input capacitance to the input capacitance of an inverter delivering the same output current.

![Inverter Diagram](image)

- **Inverter:**
  - $C_{in} = 3$
  - $LE = 1 \text{ (def)}$

- **NAND2:**
  - $C_{in} = 4$
  - $LE = 4/3$

- **NOR2:**
  - $C_{in} = 5$
  - $LE = 5/3$

**Another Example**

- **Common assumptions**
  - $C_{gate} \propto \text{Device Width}$
  - $R_{gate} \propto \text{Device Width}$

- **For a NOR gate**
  - $\beta = \mu = 3$
  - Units are not so important

- **Equivalent inverter**
  - $W_P:W_N = 6:2$
  - $C_{G,\text{INV}} = 8$

- **NOR gate input capacitance**
  - $C_{G,NOR} = 14$
  - Logical Effort = 7/4

*Caveat: don’t get confused with absolute transistor sizing!*
Calculating Parasitic Delay – Simple E.g.

**DEF:** Parasitic delay is the ratio of intrinsic capacitance at the output and intrinsic capacitance at the output of an equivalent inverter.

![Diagram showing an inverter, NAND2, and NOR2 circuits with capacitance values and parasitic delay calculations.]}

Calculating Parasitic Delay: \( p \)

- Typically given since it depends on \( C_{\text{diffusion}} \) of a gate.
- Example: assume \( C_{\text{S/D}} = 0.5C_G = 0.5C_o \)
  - For an inverter \( C_{\text{self}}/C_{\text{inv}} = p_{\text{INV}} = 0.5 \)
    - Higher \( C_{\text{S/D}}/C_G \) results in larger \( p \) (penalizing delay more).
    - \( C_{\text{S/D}}/C_G \) is often close to 1.

![Diagram showing two NOR gates with capacitance values and parasitic delay calculations.]}

- **NOR Gate**
  - \( C_{\text{S/DNOR}} = 12 + 2 + 2 = 16 \)
  - \( C_{\text{INV}} = 4C_o \)
  - \( p_{\text{NOR}} = 2(2p_{\text{INV}}) \)

*Caveat:* \( \gamma_{\text{INV}} \) is included in \( p_{\text{INV}} \) (it is not always 1)!
Calculating \( p \) Including Series Stacking

- What about the intermediate nodes?
  - One way to account for them is to use an “effective” \( p \).
  - For example: NOR pull up of B input
    
    \[
    R_{\text{NOR}} = 2 \times R_{\text{PMOS}}.
    \]
    
    \[
    \text{Delay} = \left( \frac{R_{\text{NOR}}}{2} \right) \times C_1 + R_{\text{NOR}} \times C_2 + R_{\text{NOR}} \times C_{\text{load}}
    \]
    
    \[
    p_{\text{BUP}} = \frac{\left( \frac{R_{\text{NOR}}}{2} \right) \times C_1 + R_{\text{NOR}} \times C_2}{R_{\text{inv}} \times C_{\text{inv}}} \quad \text{(where } R_{\text{inv}} = R_{\text{gate}})\]
    
    \[
    p_{\text{BUP}} = \frac{C_1}{2} + \frac{C_2}{C_{\text{inv}}}
    \]

Using \( C_{\text{SOB}} = 0.5 C_0 \)

\( C_1 = 6C_0 \) (shared)

\( C_2 = 8C_0 \)

\( p_{\text{BUP}} = 11/4 \)

Note: this increased accuracy requires different \( p \)'s for different input AND pull up/down.

Simplify by ignoring these nodes (unless otherwise specified)

Generalize N-input NAND

- Output load = 3N
  - N size-2 PMOS = 2N
  - 1 size-N NMOS = N

- Intermediate load = N (shared)

- Total pull down delay
  
  \[
  T = R(3N C_0) + \sum_{i=1}^{N-1} \left( iR/N \right) \times N C_0
  \]
  
  \[
  d \text{ (norm)} = 3N + (N^2/2 - N/2)
  \]
  
  \[
  p = (N^2/2 - N/2)
  \]
  
  - Proportional to \( N^2 \)!!!

- This is bad news for large series stacking
  
  - Even worse for PMOS (NOR)
  
  - Reality is even worse since \( C_{\text{GS}} \) makes each intermediate node capacitance > \( NC_0 \)
A Catalog of Gates

<table>
<thead>
<tr>
<th>Gate Type</th>
<th>g for Different 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 7/3 (n+2)/3</td>
</tr>
<tr>
<td>NOR</td>
<td>5/3 7/3 9/3 11/3 (2n+1)/3</td>
</tr>
<tr>
<td>Multiplexer</td>
<td>2 2 2 2 2</td>
</tr>
<tr>
<td>XOR,XNOR</td>
<td>4 12 32</td>
</tr>
</tbody>
</table>

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

Gate Type | Parasitic delay
--- | ---
Inverter | $p_{\text{inv}}$
N-input NAND | $n p_{\text{inv}}$
N-input NOR | $n p_{\text{inv}}$
N-way Multiplexer | $2 n p_{\text{inv}}$
2-input XOR,XNOR (sym) | $n 2^{n-1} p_{\text{inv}}$

Example #1: Ring Oscillator

Estimate the frequency of an N-stage ring oscillator:

- Logical Effort: $g = 1$
- Electrical Effort: $h = C_{\text{out}}/C_{\text{in}} = 1$ (gpdk090: $t_{\text{stage}} = 13\text{ps (TT)}$
- Parasitic Delay: $p = p_{\text{inv}} = 1$
- Stage Delay: $d = g \cdot h + p = 2$
- OSC Frequency: $f_{\text{osc}} = \frac{1}{2 N d \tau} = \frac{1}{4 N \tau}$
Example #2: Fanout-of-4 Inverter

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

Logical Effort:
\[ g = \frac{C_{\text{out}}}{C_{\text{in}}} = 4 \]

Electrical Effort:
\[ h = \frac{p_{\text{inv}}}{C_{\text{in}}} = \frac{1}{4} \]

Parasitic Delay:
\[ p = p_{\text{inv}} = 1 \]

Stage Delay:
\[ d = g \cdot h + p = 5 \]

Example #3: Gate Delays

- Delay of the path from A to B where \( \beta = \mu = 2 \) and \( p_{\text{inv}} = 1 \)
  - \( g_{G1} = \frac{4}{3}, p_{G1} = 2, C_{\text{IN,G1}} = 8 \)
  - \( g_{G2} = \frac{5}{3}, p_{G2} = 2, C_{\text{IN,G2}} = 15 \)
  - \( g_{G3} = 4, p_{G3} = 4, C_{\text{IN,G3}} = 30 \)
  - \( C_{\text{IN,G4}} = 15 \)
  - \( h_{G1} = \frac{C_{\text{IN,G2}} + C_{\text{IN,G3}}}{C_{\text{IN,G1}}} = 5.625, h_{G2} = \frac{C_{\text{IN,G4}}}{C_{\text{IN,G2}}} = 1 \)
  - \( d_{G1} = g_{G1} h_{G1} + p_{G1} = 9.5 \)
  - \( d_{G2} = g_{G2} h_{G2} + p_{G2} = 3.66 \)
  - Delay = 13.16
  - Normalized

\[ W_p:W_n = 10:5 \]

\[ W_p:W_n = 20:10 \]

\[ W_p:W_n = 4:4 \]
Summary

- Delay and/or power of a logic network depend significantly on the relative sizes of logic gates (not transistors within a gate).
- Inverter buffering is a simple example of the analysis:
  - The analysis leads to ~FO-4 as being optimal fanout for driving larger capacitive loads.
- To generalize analysis of delay, we introduce logical effort:
  - Delay normalized by inverter delay, \( d = gh + p \)
  - \( g \) and \( p \) are characteristics of a logic gate that depends on its structure and does not depend on gate size.
    * May have different \( g \)'s and \( p \)'s for different inputs and pull-up / pull-down
    * Simplify by using \( g_{AVG} \) and ignoring \( C \)'s of intermediate nodes
  - Once a table of \( g \)'s and \( p \)'s are created for the catalog of gates, delay can be calculated quickly and easily.
- Next we will look at how to size a network instead of just analyzing it.