Performance Characterization

- Delay analysis
- Transistor sizing
- Logical effort
- Power analysis

Delay Definitions

- \( t_{pdr} \): rising propagation delay
  - From input to rising output crossing \( V_{DD}/2 \)
- \( t_{pdf} \): falling propagation delay
  - From input to falling output crossing \( V_{DD}/2 \)
- \( t_{pd} \): average propagation delay
  - \( t_{pd} = \frac{t_{pdr} + t_{pdf}}{2} \)
- \( t_r \): rise time
  - From output crossing 0.2 \( V_{DD} \) to 0.8 \( V_{DD} \)
- \( t_f \): fall time
  - From output crossing 0.8 \( V_{DD} \) to 0.2 \( V_{DD} \)
Simulated Inverter Delay

- Solving differential equations by hand is too hard
- SPICE simulator solves the equations numerically
  - Uses more accurate I-V models too!
- But simulations take time to write

![Graph showing inverter delay](image)

Delay Estimation

- We would like to be able to easily estimate delay
  - Not as accurate as simulation
  - But easier to ask “What if?”
- The step response usually looks like a 1st order RC response with a decaying exponential.
- Use RC delay models to estimate delay
  - \( C \) = total capacitance on output node
  - Use effective resistance \( R \)
  - So that \( t_{pd} = RC \)
- Characterize transistors by finding their effective \( R \)
  - Depends on average current as gate switches
**RC Delay Models**

- Use equivalent circuits for MOS transistors
  - Ideal switch + capacitance and ON resistance
  - Unit nMOS has resistance R, capacitance C
  - Unit pMOS has resistance 2R, capacitance C
- Capacitance proportional to width
- Resistance inversely proportional to width

---

**Example: 3-input NAND**

- Sketch a 3-input NAND with transistor widths chosen to achieve effective rise and fall resistances equal to a unit inverter (R).
Example: 3-input NAND

- Sketch a 3-input NAND with transistor widths chosen to achieve effective rise and fall resistances equal to a unit inverter (R).
3-input NAND Caps

• Annotate the 3-input NAND gate with gate and diffusion capacitance.

[Diagram of a 3-input NAND gate with annotations for gate and diffusion capacitance]
3-input NAND Caps

- Annotate the 3-input NAND gate with gate and diffusion capacitance.

![3-input NAND gate diagram]

Elmore Delay

- ON transistors look like resistors
- Pullup or pulldown network modeled as RC ladder
- Elmore delay of RC ladder

\[ t_{pd} \approx \sum_{\text{nodes } i} R_{i-to-source} C_i \]

\[ = R_1 C_1 + (R_1 + R_2) C_2 + \ldots + (R_1 + R_2 + \ldots + R_N) C_N \]
Example: 2-input NAND

- Estimate worst-case rising and falling delay of 2-input NAND driving \( h \) identical gates.

![2-input NAND circuit diagram]

Example: 2-input NAND

- Estimate rising and falling propagation delays of a 2-input NAND driving \( h \) identical gates.

![2-input NAND circuit diagram]
Example: 2-input NAND

- Estimate rising and falling propagation delays of a 2-input NAND driving \( h \) identical gates.

\[
t_{pdr} = \frac{6 + 4h}{RC}
\]
Example: 2-input NAND

- Estimate rising and falling propagation delays of a 2-input NAND driving $h$ identical gates.

$$t_{pdf} = \frac{(6+4h)C}{2}$$
Example: 2-input NAND

- Estimate rising and falling propagation delays of a 2-input NAND driving \( h \) identical gates.

\[
t_{pdf} = \left( 2C \right) \left( \frac{R}{2} \right) + \left[ \left( 6 + 4h \right) C \right] \left( \frac{R}{2} + \frac{R}{2} \right)
= \left( 7 + 4h \right) RC
\]

Delay Components

- Delay has two parts
  - \textit{Parasitic delay}
    - 6 or 7 RC
    - Independent of load
  - \textit{Effort delay}
    - 4h RC
    - Proportional to load capacitance
Contamination Delay

• Best-case (contamination) delay can be substantially less than propagation delay.
• Ex: If both inputs fall simultaneously

\[ t_{cdr} = (3 + 2h)RC \]

Diffusion Capacitance

• We assumed contacted diffusion on every s / d.
• Good layout minimizes diffusion area
• Ex: NAND3 layout shares one diffusion contact
  – Reduces output capacitance by 2C
  – Merged uncontacted diffusion might help too
Layout Comparison

- Which layout is better?

\[
\begin{array}{c}
V_{DD} \\
A & B \\
GND \\
\end{array}
\]

\[
\begin{array}{c}
V_{DD} \\
A & B \\
GND \\
\end{array}
\]

Resizing the Inverter

Minimum-sized transistor:
\[W = 3\lambda, \ L = 2\lambda\]

To get equal rise and fall times,
\[\beta_n = \beta_p \Rightarrow W_p = 3W_n,\]
assuming that electron mobility is three times that of holes
\[W_p = 9\lambda\]

Sometimes the function being implemented makes resizing unnecessary!
Analyzing the NAND Gate

\[ \beta_{n, \text{eff}} = \frac{1}{\frac{1}{\beta_{n1}} + \frac{1}{\beta_{n2}} + \frac{1}{\beta_{n3}}} \]

Resistances are in series (conductances are in parallel)

If \( \beta_{n1} = \beta_{n2} = \beta_{n3} = \beta_n \) then \( \beta_{n, \text{eff}} = \beta_n / 3 \)

• Pull-down circuit has three times resistance, one-third times the conductance

Why not consider resistances in parallel?

For pull-up, only one transistor has to be on, \( \beta_{p, \text{eff}} = \min\{\beta_{p1}, \beta_{p2}, \beta_{p3}\} \)

If \( \beta_{p1} = \beta_{p2} = \beta_{p3} = \beta_p \) then \( \beta_{p, \text{eff}} = \beta_p \) \( \Rightarrow \) no resizing is necessary

Analyzing the NOR Gate

\[ \beta_{p, \text{eff}} = \frac{1}{\frac{1}{\beta_{p1}} + \frac{1}{\beta_{p2}} + \frac{1}{\beta_{p3}}} \]

Resistances are in series (conductances are in parallel)

If \( \beta_{p1} = \beta_{p2} = \beta_{p3} = \beta_p \) then \( \beta_{p, \text{eff}} = \beta_p / 3 \)

• Pull-up circuit has three times resistance, one-third times the conductance

For pull-down, only one transistor has to be on, \( \beta_{n, \text{eff}} = \min\{\beta_{n1}, \beta_{n2}, \beta_{n3}\} \)

If \( \beta_{n1} = \beta_{n2} = \beta_{n3} = \beta_n = 3\beta_p \) then \( \beta_{n, \text{eff}} = 9\beta_{p, \text{eff}} \) \( \Rightarrow \) considerable resizing is necessary

\[ W_p = 9W_n \]
Effect of Series Transistors

Transistor resizing example

Resize the pull-up transistors to make pull-up times equal

After resizing:
- a: $2\beta_p$
- b: $2\beta_p$
- c: $\beta_p$

Effect of Series Transistors

Resize the pull-up transistors to make pull-up times equal.
Transistor Placement (Series Stack)

How to order transistors in a series stack?

Body effect: $\delta V_t \propto \sqrt{V_{sb}}$

- At time $t = 0$, $a=b=c=0$, $f=1$, capacitances are charged.
- Ideally $V_{ta} = V_{tb} = V_{tc} \approx 0.8V$
- However, $V_{ta} > V_{tb} > V_{tc}$ because of body effect.

- If $a$, $b$, $c$ become 1 at the same time, which transistor will switch on first?
  - $t_c$ will switch on first ($V_{sb}$ for $t_c$ is zero), $C_c$ will discharge, pulling $V_{sb}$ for $t_b$ to zero.
  - If signals arrive at different times, how should the transistors be ordered?
  - Design strategy: place latest arriving signal nearest to output-early signals will discharge internal nodes.

Transistor Placement

Primary inputs (change simultaneously)
Some Design Guidelines

- Use NAND gates (instead of NOR) wherever possible
- Placed inverters (buffers) at high fanout nodes to improve drive capability
- Avoid use of NOR completely in high-speed circuits: \( A_1 + A_2 + \ldots + A_n = A_1 \cdot A_2 \cdot \ldots \cdot A_n \)

Some Design Guidelines

- Use limited fan-in (<10): high fan-in \( \Rightarrow \) long series stacks
- Use minimum-sized gates on high fan-out nodes: minimize load presented to driving gate
Logical Effort

- Chip designers face a bewildering array of choices:
  - What is the best circuit topology for a function?
  - How many stages of logic give least delay?
  - How wide should the transistors be?

- Logical effort is a method to make these decisions:
  - Uses a simple model of delay
  - Allows back-of-the-envelope calculations
  - Helps make rapid comparisons between alternatives
  - Emphasizes remarkable symmetries

Example

- 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?
Delay in a Logic Gate

- Express delays in process-independent unit
  \[ d = \frac{d_{abs}}{\tau} \]

  \[ \tau = 3RC \]

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

- Delay has two components
  \[ d = f + p \]
Delay in a Logic Gate

• Express delays in process-independent unit
  \[ d = \frac{d_{\text{abs}}}{\tau} \]

• Delay has two components
  \[ d = f + p \]

• Effort delay \( f = gh \) (a.k.a. stage effort)
  – Again has two components
  – \( g \): logical effort
    – Measures relative ability of gate to deliver current
    – \( g = 1 \) for inverter
Delay in a Logic Gate

- Express delays in process-independent unit
  \[ d = \frac{d_{abs}}{\tau} \]

- Delay has two components
  \[ d = f + p \]

- Effort delay \( f = gh \) (a.k.a. stage effort)
  - Again has two components

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

\[ d = f + p = gh + p \]

- What about NOR2?

\[ g = \frac{1}{3} \]
\[ p = 1 \]
\[ d = h + 1 \]

Effort Delay: \( f \)
Parasitic Delay: \( p \)

Effort Delay: \( f = (4/3)h + 2 \)
Computing Logical Effort

- DEF: Logical effort is the ratio of the input capacitance of a gate to the input capacitance of an inverter delivering the same output current.
- Measure from delay vs. fanout plots
- Or estimate by counting transistor widths

![Diagrams of gates with logical effort calculations]

Catalog of Gates

- Logical effort of common gates

<table>
<thead>
<tr>
<th>Gate type</th>
<th>Number of inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Inverter</td>
<td>1</td>
</tr>
<tr>
<td>NAND</td>
<td>4/3</td>
</tr>
<tr>
<td>NOR</td>
<td>5/3</td>
</tr>
<tr>
<td>Tristate / mux</td>
<td>2</td>
</tr>
</tbody>
</table>
Catalog of Gates

- Parasitic delay of common gates
  - In multiples of $p_{\text{inv}} \approx 1$

<table>
<thead>
<tr>
<th>Gate type</th>
<th>Number of inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Inverter</td>
<td>1</td>
</tr>
<tr>
<td>NAND</td>
<td>2 3 4 n</td>
</tr>
<tr>
<td>NOR</td>
<td>2 3 4 n</td>
</tr>
<tr>
<td>Tristate / mux</td>
<td>2 4 6 8 2n</td>
</tr>
<tr>
<td>XOR, XNOR</td>
<td>4 6 8</td>
</tr>
</tbody>
</table>

Example: Ring Oscillator

- Estimate the frequency of an N-stage ring oscillator

Logical Effort: $g = \phantom{$}
Electrical Effort: $h = \phantom{$}
Parasitic Delay: $p = \phantom{$}
Stage Delay: $d = \phantom{$}
Frequency: $f_{\text{osc}} = \phantom{$}$
Example: Ring Oscillator

- Estimate the frequency of an N-stage ring oscillator

![Ring Oscillator Diagram]

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

31 stage ring oscillator in 0.6 \( \mu \)m process has frequency of \(~ 200\) MHz

Example: FO4 Inverter

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

![FO4 Inverter Diagram]

Logical Effort: \( g = \)
Electrical Effort: \( h = \)
Parasitic Delay: \( p = \)
Stage Delay: \( d = \)
**Example: FO4 Inverter**

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

Logical Effort: \( g = 1 \)

Electrical Effort: \( h = 4 \)

Parasitic Delay: \( p = 1 \)

Stage Delay: \( d = 5 \)

The FO4 delay is about
- 200 ps in a 0.6 \( \mu \)m process
- 60 ps in a 180 nm process
- 1/3 ns in an \( f \) \( \mu \)m process

**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** \( F = \prod f_i = \prod g_i h_i \)
Multistage Logic Networks

- Logical effort generalizes to multistage networks
- *Path Logical Effort* \( G = \prod g_i \)
- *Path Electrical Effort* \( H = \frac{C_{out-path}}{C_{in-path}} \)
- *Path Effort* \( F = \prod f_i = \prod g_i h_i \)

- Can we write \( F = GH \)?

Paths that Branch

- No! Consider paths that branch:

\[
\begin{align*}
G &= \\
H &= \\
GH &= \\
h_1 &= \\
h_2 &= \\
F &= GH?
\end{align*}
\]
Paths that Branch

- No! Consider paths that branch:

\[ G = 1 \]
\[ H = 90 / 5 = 18 \]
\[ GH = 18 \]
\[ h_1 = (15 + 15) / 5 = 6 \]
\[ h_2 = 90 / 15 = 6 \]
\[ F = g_1 g_2 h_1 h_2 = 36 = 2GH \]

Branching Effort

- Introduce branching effort
  - Accounts for branching between stages in path
  \[ b = \frac{C_{\text{on path}}}{C_{\text{on path}}} + \frac{C_{\text{off path}}}{C_{\text{on path}}} \]

\[ B = \prod b_i \]

- Now we compute the path effort
  - \[ F = GBH \]

Note:
\[ \prod h_i = BH \]
Multistage Delays

- Path Effort Delay \( D_F = \sum f_i \)
- Path Parasitic Delay \( P = \sum p_i \)
- Path Delay \( D = \sum d_i = D_F + P \)

Designing Fast Circuits

\( D = \sum d_i = D_F + P \)

- Delay is smallest when each stage bears same effort
  \( \hat{f} = g_i h_i = F^{\frac{1}{N}} \)
- Thus minimum delay of N stage path is
  \( D = NF^{\frac{1}{N}} + P \)
- This is a key result of logical effort
  - Find fastest possible delay
  - Doesn’t require calculating gate sizes
Gate Sizes

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

- 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.

Example: 3-stage path

- Select gate sizes x and y for least delay from A to B
Example: 3-stage path

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

• Work backward for sizes

\[ y = 45 \times \frac{5}{3} / 5 = 15 \]
\[ x = (15 \times 2) \times \frac{5}{3} / 5 = 10 \]
Best Number of Stages

• How many stages should a path use?
  – Minimizing number of stages is not always fastest

• Example: drive 64-bit datapath with unit inverter

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

\[
= N(64)^{1/N} + N
\]
Derivation

- Consider adding inverters to end of path
  - How many give least delay?

\[ D = NF^{F_{1}/N} + \sum_{i=1}^{n_1} p_i + (N - n_1) p_{inv} \]

\[ \frac{\partial D}{\partial N} = -F^{F_{1}/N} \ln F^{F_{1}/N} + F^{F_{1}/N} + p_{inv} = 0 \]

- Define best stage effort \( \rho = F^{F_{1}/N} \)

\[ p_{inv} + \rho (1 - \ln \rho) = 0 \]

Best Stage Effort

\[ p_{inv} + \rho (1 - \ln \rho) = 0 \]

- has no closed-form solution

- Neglecting parasitics (\( p_{inv} = 0 \)), we find \( \rho = 2.718 \) (e)
- For \( p_{inv} = 1 \), solve numerically for \( \rho = 3.59 \)
Review of Definitions

<table>
<thead>
<tr>
<th>Term</th>
<th>Stage</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>number of stages</td>
<td>1</td>
<td>N</td>
</tr>
<tr>
<td>logical effort</td>
<td>$g$</td>
<td>$G = \prod g_i$</td>
</tr>
<tr>
<td>electrical effort</td>
<td>$h = \frac{C_{in}}{t_{on}}$</td>
<td>$H = \frac{C_{on-path}}{t_{on}}$</td>
</tr>
<tr>
<td>branching effort</td>
<td>$b = \frac{C_{on-path} \cdot C_{off-path}}{t_{on-path}}$</td>
<td>$B = \prod b_i$</td>
</tr>
<tr>
<td>effort</td>
<td>$f = gh$</td>
<td>$F = GBH$</td>
</tr>
<tr>
<td>effort delay</td>
<td>$f$</td>
<td>$D_F = \sum f_i$</td>
</tr>
<tr>
<td>parasitic delay</td>
<td>$p$</td>
<td>$P = \sum p_i$</td>
</tr>
<tr>
<td>delay</td>
<td>$d = f + p$</td>
<td>$D = \sum d_i = D_F + P$</td>
</tr>
</tbody>
</table>

Method of Logical Effort

1) Compute path effort \( F = GBH \)
2) Estimate best number of stages \( N = \log_4 F \)
3) Sketch path with N stages
4) Estimate least delay \( D = NF^{\frac{1}{2}} + P \)
5) Determine best stage effort \( \hat{f} = F^{\frac{1}{2}} \)
6) Find gate sizes

\[ C_{in} = \frac{g^* C_{out}}{\hat{f}} \]
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 \( \log_4 F \) FO4 inverter delays
  - Inverters and NAND2 best for driving large caps
- Provides language for discussing fast circuits
  - But requires practice to master
**Power and Energy**

- Power is drawn from a voltage source attached to the $V_{DD}$ pin(s) of a chip.
  \[ P(t) = i_{DD}(t)V_{DD} \]
- Instantaneous Power:
  \[ E = \int_0^T P(t) dt = \int_0^T i_{DD}(t)V_{DD} dt \]
- Energy:
  \[ P_{avg} = \frac{E}{T} = \frac{1}{T} \int_0^T i_{DD}(t)V_{DD} dt \]
- Average Power:

**Dynamic Power**

- Dynamic power is required to charge and discharge load capacitances when transistors switch.
- One cycle involves a rising and falling output.
- On rising output, charge $Q = CV_{DD}$ is required
- On falling output, charge is dumped to GND
- This repeats $T_{fsw}$ times over an interval of $T$
Dynamic Power Cont.

\[ P_{\text{dynamic}} = \int_0^T i_{DD}(t) V_{DD} dt \]

\[ = \frac{V_{DD}}{T} \int_0^T i_{DD}(t) dt \]

\[ = \frac{V_{DD}}{T} \left[ T f_{\text{sw}} C V_{DD} \right] \]

\[ = C V_{DD}^2 f_{\text{sw}} \]
Activity Factor

- Suppose the system clock frequency = f
- Let $f_{sw} = \alpha f$, where $\alpha =$ activity factor
  - If the signal is a clock, $\alpha = 1$
  - If the signal switches once per cycle, $\alpha = \frac{1}{2}$
  - Dynamic gates:
    - Switch either 0 or 2 times per cycle, $\alpha = \frac{1}{2}$
  - Static gates:
    - Depends on design, but typically $\alpha = 0.1$

- Dynamic power: $P_{\text{dynamic}} = \alpha CV_{DD}^2 f$

Short Circuit Current

- When transistors switch, both nMOS and pMOS networks may be momentarily ON at once
- Leads to a blip of “short circuit” current.
- < 10% of dynamic power if rise/fall times are comparable for input and output
Example

- 200 Mtransistor chip
  - 20M logic transistors
    - Average width: 12 λ
  - 180M memory transistors
    - Average width: 4 λ
  - 1.2 V 100 nm process
  - $C_g = 2 \text{fF/μm}$

Dynamic Example

- Static CMOS logic gates: activity factor = 0.1
- Memory arrays: activity factor = 0.05 (many banks!)

- Estimate dynamic power consumption per MHz. Neglect wire capacitance and short-circuit current.
Dynamic Example

- Static CMOS logic gates: activity factor = 0.1
- Memory arrays: activity factor = 0.05 (many banks!)
- Estimate dynamic power consumption per MHz. Neglect wire capacitance.

\[
C_{\text{logic}} = (20 \times 10^6)(12\lambda)(0.05\mu m / \lambda)(2fF / \mu m) = 24nF
\]

\[
C_{\text{mem}} = (180 \times 10^6)(4\lambda)(0.05\mu m / \lambda)(2fF / \mu m) = 72nF
\]

\[
P_{\text{dynamic}} = \left[0.1C_{\text{logic}} + 0.05C_{\text{mem}}\right](1.2)^2 f = 8.6 \text{ mW/MHz}
\]

Static Power

- Static power is consumed even when chip is quiescent.
  - Ratioed circuits burn power in fight between ON transistors
  - Leakage draws power from nominally OFF devices

\[
I_{ds} = I_{ds0}e^{\frac{V_g-V_T}{V_{tr}} \left[1-e^{-V_{ds}/V_{tr}}\right]}
\]

\[
V_c = V_{t0} - \eta V'_{ds} + \gamma \left(\sqrt{\phi_s + V_{sb}} - \sqrt{\phi_s}\right)
\]
Ratio Example

• The chip contains a 32 word x 48 bit ROM
  – Uses pseudo-nMOS decoder and bitline pullups
  – On average, one wordline and 24 bitlines are high

• Find static power drawn by the ROM
  – $\beta = 75 \, \mu A/V^2$
  – $V_{tp} = -0.4V$

Solution:

$I_{\text{pull-up}} = \beta \left( V_{DD} - |\bar{V}_{in}| \right)^2 = 24 \mu A$

$P_{\text{pull-up}} = V_{DD} I_{\text{pull-up}} = 29 \mu W$

$P_{\text{static}} = (31 + 24)P_{\text{pull-up}} = 1.6 \, mW$
Leakage Example

- The process has two threshold voltages and two oxide thicknesses.
- Subthreshold leakage:
  - 20 nA/µm for low $V_t$
  - 0.02 nA/µm for high $V_t$
- Gate leakage:
  - 3 nA/µm for thin oxide
  - 0.002 nA/µm for thick oxide
- Memories use low-leakage transistors everywhere
- Gates use low-leakage transistors on 80% of logic

Leakage Example Cont.

- Estimate static power:
Leakage Example Cont.

- Estimate static power:
  - High leakage: \((20 \times 10^6)(0.2)(12 \lambda)(0.05 \mu m / \lambda) = 2.4 \times 10^6 \mu m\)
  - Low leakage: \((20 \times 10^6)(0.8)(12 \lambda)(0.05 \mu m / \lambda) + (180 \times 10^6)(4 \lambda)(0.05 \mu m / \lambda) = 45.6 \times 10^6 \mu m\)

\[
I_{\text{static}} = \left(2.4 \times 10^6 \mu m\right)\left[(20nA / \mu m)/2 + (3nA / \mu m)\right] + \\
\left(45.6 \times 10^6 \mu m\right)\left[(0.02nA / \mu m)/2 + (0.002nA / \mu m)\right]
\]

\[
= 32mA
\]

\[
P_{\text{static}} = I_{\text{static}}V_{\text{DD}} = 38mW
\]

- If no low leakage devices, \(P_{\text{static}} = 749 \text{ mW}(!)\)
Low Power Design

• Reduce dynamic power
  – $\alpha$:
  – $C$:
  – $V_{DD}$:
  – $f$:
• Reduce static power

Low Power Design

• Reduce dynamic power
  – $\alpha$: clock gating, sleep mode
  – $C$:
  – $V_{DD}$:
  – $f$:
• Reduce static power
Low Power Design

• Reduce dynamic power
  – $\alpha$: clock gating, sleep mode
  – $C$: small transistors (esp. on clock), short wires
  – $V_{DD}$:
  – $f$:
• Reduce static power
Low Power Design

- Reduce dynamic power
  - $\alpha$: clock gating, sleep mode
  - $C$: small transistors (esp. on clock), short wires
  - $V_{DD}$: lowest suitable voltage
  - $f$: lowest suitable frequency
- Reduce static power
  - Selectively use ratioed circuits
  - Selectively use low $V_t$ devices
  - Leakage reduction:
    stacked devices, body bias, low temperature