CMOS Inverter

Additional Slides

Vishal Saxena

ECE, Boise State University

Oct 21, 2010
## Inverter Operation Regions

<table>
<thead>
<tr>
<th>Region</th>
<th>NMOS</th>
<th>PMOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Cutoff</td>
<td>Triode</td>
</tr>
<tr>
<td>B</td>
<td>Saturation</td>
<td>Triode</td>
</tr>
<tr>
<td>C</td>
<td>Saturation</td>
<td>Saturation</td>
</tr>
<tr>
<td>D</td>
<td>Triode</td>
<td>Saturation</td>
</tr>
<tr>
<td>E</td>
<td>Triode</td>
<td>Cutoff</td>
</tr>
</tbody>
</table>

![Inverter diagram](image)
Noise Margin

- How much noise can a gate input see before it does not recognize the output?
  - Noise margins of a digital gate indicate how well it will perform with noisy input.

![Noise Margin Diagram]

- Logical High Output Range
  - Logical High Input Range
  - Indeterminate Region
  - Logical Low Output Range
  - Logical Low Input Range
Noise Margin

- How much noise can a gate input see before it does not recognize the output?
- Noise margins of a digital gate indicate how well it will perform with noisy input

![Diagram of inverter with noise margin regions]

Output Characteristics

- Logical High Output Range
- Logical Low Output Range

Input Characteristics

- Indeterminate Region
- Logical High Input Range
- Logical Low Input Range
Noise Margin

- $NM_H = V_{IH} - V_{OH}$
  - **HIGH** noise margin

- $NM_L = V_{IL} - V_{OL}$
  - **LOW** noise margin

- $V_{IH} =$ minimum **HIGH** input voltage
- $V_{IL} =$ maximum **LOW** input voltage
- $V_{OH} =$ minimum **HIGH** output voltage
- $V_{OL} =$ maximum **LOW** output voltage
Noise Margin

- $NM_H = V_{IH} - V_{OH}$
  - HIGH noise margin

- $NM_L = V_{IL} - V_{OL}$
  - LOW noise margin

- $V_{IH} = \text{minimum HIGH input voltage}$
- $V_{IL} = \text{maximum LOW input voltage}$
- $V_{OH} = \text{minimum HIGH output voltage}$
- $V_{OL} = \text{maximum LOW output voltage}$
Noise Margin

- $NM_H = V_{IH} - V_{OH}$
  - HIGH noise margin
- $NM_L = V_{IL} - V_{OL}$
  - LOW noise margin

- $V_{IH} = \text{minimum HIGH input voltage}$
- $V_{IL} = \text{maximum LOW input voltage}$
- $V_{OH} = \text{minimum HIGH output voltage}$
- $V_{OL} = \text{maximum LOW output voltage}$
Noise Margin

- $N_{MH} = V_{IH} - V_{OH}$
  - HIGH noise margin
- $N_{ML} = V_{IL} - V_{OL}$
  - LOW noise margin
- $V_{IH} = \text{minimum HIGH input voltage}$
- $V_{IL} = \text{maximum LOW input voltage}$
- $V_{OH} = \text{minimum HIGH output voltage}$
- $V_{OL} = \text{maximum LOW output voltage}$
Noise Margin

- $NM_H = V_{IH} - V_{OH}$
  - HIGH noise margin
- $NM_L = V_{IL} - V_{OL}$
  - LOW noise margin
- $V_{IH} =$ minimum HIGH input voltage
- $V_{IL} =$ maximum LOW input voltage
- $V_{OH} =$ minimum HIGH output voltage
- $V_{OL} =$ maximum LOW output voltage
Noise Margin

- \( NM_H = V_{IH} - V_{OH} \)
  - HIGH noise margin
- \( NM_L = V_{IL} - V_{OL} \)
  - LOW noise margin
- \( V_{IH} = \) minimum HIGH input voltage
- \( V_{IL} = \) maximum LOW input voltage
- \( V_{OH} = \) minimum HIGH output voltage
- \( V_{OL} = \) maximum LOW output voltage
To maximize noise margins, select logic levels at

- unity gain point of DC transfer characteristics
To maximize noise margins, select logic levels at the unity gain point of DC transfer characteristics.
To maximize noise margins, select logic levels at

- unity gain point of DC transfer characteristics
- regenerate the logic levels (gain>1)
To maximize noise margins, select logic levels at

- unity gain point of DC transfer characteristics
  - regenerate the logic levels (gain > 1)
To maximize noise margins, select logic levels at

- unity gain point of DC transfer characteristics
- regenerate the logic levels (gain>1)
If $\frac{\beta_n}{\beta_p} \neq 1$, inverter’s switching point ($V_{SP}$) will move from the ideal value of $\frac{V_{DD}}{2}$

- called **skewed** gate
Inverter Layout

- Two styles for laying out an inverter
- Power and ground routed on metal-1 using standard frame

---

Vishal Saxena | CMOS Inverter
Two styles for laying out an inverter

Power and ground routed on metal-1 using standard frame
Latch-up

- Fast voltage pulses can feed-through the C1 or C2 and turn on the parasitic BJT
- If any of the BJT is turned on, it creates a positive feedback loop
- Eventually both the BJTs are turned fully on and the circuit is stuck in that state (undesired)

Cross-sectional view of an inverter showing parasitic bipolar transistors and resistors

Schematic for understanding latch-up

Vishal Saxena | CMOS Inverter
Latch-up

- Fast voltage pulses can feed-through the C1 or C2 and turn on the parasitic BJT.
- If any of the BJT is turned on, it creates a positive feedback loop.
  - Eventually both the BJTs are turned fully on and the circuit is stuck in that state (undesired).

Cross-sectional view of an inverter showing parasitic bipolar transistors and resistors.

Schematic for understanding latch-up.
Fast voltage pulses can feed-through the C1 or C2 and turn on the parasitic BJT.

If any of the BJT is turned on, it creates a positive feedback loop.

Eventually both the BJTs are turned fully on and the circuit is stuck in that state (undesired).

Cross-sectional view of an inverter showing parasitic bipolar transistors and resistors.

Schematic for understanding latch-up.
Latch-up prevention

- Reduce the well series resistances (RW1 and RW2) by using as many contacts as possible and closer to the inverter.
  - can also use guard ring structures
- Use slow rise and fall times in the logic
- Reduce drain areas to reduce C1 and C2

Cross-sectional view of an inverter showing parasitic bipolar transistors and resistors

Schematic for understanding latch-up
Latch-up prevention

- Reduce the well series resistances (RW1 and RW2) by using as many contacts as possible and closer to the inverter
  - can also use guard ring structures
- Use slow rise and fall times in the logic
- Reduce drain areas to reduce C1 and C2

Cross-sectional view of an inverter showing parasitic bipolar transistors and resistors

Schematic for understanding latch-up
Latch-up prevention

- Reduce the well series resistances (RW1 and RW2) by using as many contacts as possible and closer to the inverter.
- Can also use guard ring structures.
- Use slow rise and fall times in the logic.
- Reduce drain areas to reduce C1 and C2.

Cross-sectional view of an inverter showing parasitic bipolar transistors and resistors.

Schematic for understanding latch-up.
Latch-up prevention

- Reduce the well series resistances (RW1 and RW2) by using as many contacts as possible and closer to the inverter
  - can also use guard ring structures
- Use slow rise and fall times in the logic
- Reduce drain areas to reduce C1 and C2

Cross-sectional view of an inverter showing parasitic bipolar transistors and resistors

Schematic for understanding latch-up
In nm-CMOS, assuming that for equal drive strengths $W_p = 2W_n$

- effective switching resistance of PMOS & NMOS = $R$
- in MOSFETs switching model assume that $C_{in} = C_{out} = C$

- Propgataion delay ($d$) =
  $$t_{pLH} = t_{pHL} = 0.7 \times R(C_{outp} + C_{outn}) \triangleq 0.7 \times 3RC$$
  $$\Rightarrow \tau = 3RC$$

- Can express delay in a process-independent unit
  $$d = d_{abs}/0.7\tau$$
  $$d = 1$$ for an inverter with no load
- In nm-CMOS, assuming that for equal drive strengths $W_p = 2W_n$
  - effective switching resistance of PMOS & NMOS = $R$
  - in MOSFETs switching model assume that $C_{in} = C_{out} = C$

- Propagation delay ($d$) =
  $t_{pLH} = t_{pHL} = 0.7 \times R(C_{outp} + C_{outn}) \triangleq 0.7 \times 3RC$
  - $\Rightarrow \tau = 3RC$

- Can express delay in a process-independent unit
  - $d = d_{abs}/0.7\tau$
  - $d = 1$ for an inverter with no load
In nm-CMOS, assuming that for equal drive strengths $W_p = 2W_n$

- effective switching resistance of PMOS & NMOS = $R$
- in MOSFETs switching model assume that $C_{in} = C_{out} = C$

Propgataion delay ($d$) =

$t_{pLH} = t_{pHL} = 0.7 \times R(C_{outp} + C_{outn}) \triangleq 0.7 \times 3RC$

$\Rightarrow \tau = 3RC$

- Can express delay in a process-independent unit

- $d = d_{abs}/0.7\tau$
- $d = 1$ for an inverter with no load
In nm-CMOS, assuming that for equal drive strengths $W_p = 2W_n$
- effective switching resistance of PMOS & NMOS = $R$
- in MOSFETs switching model assume that $C_{in} = C_{out} = C$

Propgataion delay ($d$) =
\[ t_{p LH} = t_{p HL} = 0.7 \times R(C_{outp} + C_{outn}) \triangleq 0.7 \times 3RC \]
- $\Rightarrow \tau = 3RC$

Can express delay in a process-independent unit
- $d = d_{abs}/0.7\tau$
- $d = 1$ for an inverter with no load
In nm-CMOS, assuming that for equal drive strengths $W_p = 2W_n$

- effective switching resistance of PMOS & NMOS = $R$
- in MOSFETs switching model assume that $C_{in} = C_{out} = C$

Propagation delay ($d$) =

$$t_{pLH} = t_{pHL} = 0.7 \times R(C_{outp} + C_{outn}) \triangleq 0.7 \times 3RC$$

$$\Rightarrow \tau = 3RC$$

Can express delay in a process-independent unit

- $d = d_{abs}/0.7\tau$
- $d = 1$ for an inverter with no load
Delay in a Logic Gate

- Can express delay in a process-independent unit
  \[ d = d_{\text{abs}} / \tau \]
  \[ \tau = 0.7 \times R(C_{\text{outp}} + C_{\text{outn}}) \triangleq 0.7 \times 3RC \]

- Delay has two components: \( d = f + p \)
  - \( f \): effort delay \( = g \cdot h \) (a.k.a. stage effort)
  - \( g \): logical effort
    - measures relative ability of gate to deliver current
    - \( g = 1 \) for inverter (baseline circuit)
  - \( h \): electrical effort \( = C_{\text{out}} / C_{\text{in}} \)
    - ratio of output to input capacitance
    - sometimes called fanout
  - \( p \): parasitic delay
    - represents delay of gate driving no load
      - set by internal parasitic capacitance

Vishal Saxena | CMOS Inverter
Delay in a Logic Gate

- Can express delay in a process-independent unit
  \[ d = \frac{d_{\text{abs}}}{\tau} \quad \tau = 0.7 \times R(C_{\text{outp}} + C_{\text{outn}}) \triangleq 0.7 \times 3RC \]

- Delay has two components: \( d = f + p \)
  - \( f \): effort delay \( = g \cdot h \) (a.k.a. stage effort)
    - again has two components:
      - \( g \): logical effort
        - measures relative ability of gate to deliver current
        - \( g = 1 \) for inverter (baseline circuit)
      - \( h \): electrical effort \( = \frac{C_{\text{out}}}{C_{\text{in}}} \)
        - ratio of output to input capacitance
        - sometimes called fanout
  - \( p \): parasitic delay
    - represents delay of gate driving no load
    - set by internal parasitic capacitance
Delay in a Logic Gate

- Can express delay in a process-independent unit
  \[ d = \frac{d_{abs}}{\tau} \]
  \[ \tau = 0.7 \times R(C_{outp} + C_{outn}) \triangleq 0.7 \times 3RC \]

- Delay has two components: \( d = f + p \)
  - \( f \): effort delay = \( g \cdot h \) (a.k.a. stage effort)
    - again has two components:
      - \( g \): logical effort
        - measures relative ability of gate to deliver current
        - \( g=1 \) for inverter (baseline circuit)
      - \( h \): electrical effort = \( \frac{C_{out}}{C_{in}} \)
        - ratio of output to input capacitance
        - sometimes called fanout
  - \( p \): parasitic delay
    - represents delay of gate driving no load
    - set by internal parasitic capacitance
Delay in a Logic Gate

- Can express delay in a process-independent unit
  - \( d = \frac{d_{abs}}{\tau} \)
  - \( \tau = 0.7 \times R(C_{outp} + C_{outn}) \triangleq 0.7 \times 3RC \)

- Delay has two components: \( d = f + p \)
  - \( f: \) effort delay \( = g \cdot h \) (a.k.a. stage effort)
    - again has two components:
      - \( g: \) logical effort
        - measures relative ability of gate to deliver current
        - \( g=1 \) for inverter (baseline circuit)
      - \( h: \) electrical effort \( = \frac{C_{out}}{C_{in}} \)
        - ratio of output to input capacitance
        - sometimes called fanout
  - \( p: \) parasitic delay
    - represents delay of gate driving no load
    - set by internal parasitic capacitance
Delay in a Logic Gate

- Can express delay in a process-independent unit
  
  - $d = \frac{d_{abs}}{\tau}$
  
  - $\tau = 0.7 \times R(C_{outp} + C_{outn}) \equiv 0.7 \times 3RC$

- Delay has two components: $d = f + p$

  - $f$: effort delay $= g \cdot h$ (a.k.a. stage effort)
    
    - again has two components:
      
      - $g$: logical effort
        
        - measures relative ability of gate to deliver current
        
        - $g=1$ for inverter (baseline circuit)
      
      - $h$: electrical effort $= C_{out} / C_{in}$
        
        - ratio of output to input capacitance
        
        - sometimes called fanout
  
  - $p$: parasitic delay
    
    - represents delay of gate driving no load
    
    - set by internal parasitic capacitance
Delay in a Logic Gate

- Can express delay in a process-independent unit
  - \( d = \frac{d_{\text{abs}}}{\tau} \)  \( \tau = 0.7 \times R(C_{\text{outp}} + C_{\text{outn}}) \cong 0.7 \times 3RC \)

- Delay has two components: \( d = f + p \)
  - \( f \): effort delay = \( g \cdot h \) (a.k.a. stage effort)
    - again has two components:
      - \( g \): logical effort
        - measures relative ability of gate to deliver current
        - \( g=1 \) for inverter (baseline circuit)
      - \( h \): electrical effort = \( C_{\text{out}} / C_{\text{in}} \)
        - ratio of output to input capacitance
        - sometimes called fanout
  - \( p \): parasitic delay
    - represents delay of gate driving no load
    - set by internal parasitic capacitance
Delay in a Logic Gate

- Can express delay in a process-independent unit
  \[ d = \frac{d_{\text{abs}}}{\tau} \quad \tau = 0.7 \times R(C_{\text{outp}} + C_{\text{outn}}) \triangleq 0.7 \times 3RC \]

- Delay has two components: \( d = f + p \)
  - \( f \): effort delay = \( g \cdot h \) (a.k.a. stage effort)
    - again has two components:
      - \( g \): logical effort
        - measures relative ability of gate to deliver current
        - \( g = 1 \) for inverter (baseline circuit)
      - \( h \): electrical effort = \( C_{\text{out}} / C_{\text{in}} \)
        - ratio of output to input capacitance
        - sometimes called fanout
  - \( p \): parasitic delay
    - represents delay of gate driving no load
    - set by internal parasitic capacitance
Can express delay in a process-independent unit

\[ d = \frac{d_{abs}}{\tau} \quad \tau = 0.7 \times R(C_{outp} + C_{outn}) \triangleq 0.7 \times 3RC \]

Delay has two components: \( d = f + p \)

- **f**: effort delay = \( g \cdot h \) (a.k.a. stage effort)
  - again has two components:
    - **g**: logical effort
      - measures relative ability of gate to deliver current
      - \( g = 1 \) for inverter (baseline circuit)
    - **h**: electrical effort = \( \frac{C_{out}}{C_{in}} \)
      - ratio of output to input capacitance
      - sometimes called fanout

- **p**: parasitic delay
  - represents delay of gate driving no load
  - set by internal parasitic capacitance
Delay in a Logic Gate

- Can express delay in a process-independent unit
  \[ d = \frac{d_{abs}}{\tau} \quad \tau = 0.7 \times R(C_{outp} + C_{outn}) \geq 0.7 \times 3RC \]

- Delay has two components: \( d = f + p \)

  - \( f \): effort delay = \( g \cdot h \) (a.k.a. stage effort)
    - again has two components:
      - \( g \): logical effort
        - measures relative ability of gate to deliver current
        - \( g=1 \) for inverter (baseline circuit)
      - \( h \): electrical effort = \( C_{out} / C_{in} \)
        - ratio of output to input capacitance
        - sometimes called fanout

  - \( p \): parasitic delay
    - represents delay of gate driving no load
      - set by internal parasitic capacitance
Delay in a Logic Gate

- Can express delay in a process-independent unit
  \[ d = \frac{d_{\text{abs}}}{\tau} \]
  \[ \tau = 0.7 \times R(C_{\text{outp}} + C_{\text{outn}}) \triangleq 0.7 \times 3RC \]

- Delay has two components: \( d = f + p \)
  - \( f \): effort delay = \( g \cdot h \) (a.k.a. stage effort)
    - again has two components:
      - \( g \): logical effort
        - measures relative ability of gate to deliver current
        - \( g = 1 \) for inverter (baseline circuit)
      - \( h \): electrical effort = \( \frac{C_{\text{out}}}{C_{\text{in}}} \)
        - ratio of output to input capacitance
        - sometimes called fanout
  - \( p \): parasitic delay
    - represents delay of gate driving no load
      - set by internal parasitic capacitance
Delay in a Logic Gate

- Can express delay in a process-independent unit
  - \( d = \frac{d_{abs}}{\tau} \quad \tau = 0.7 \times R(C_{outp} + C_{outn}) \triangleq 0.7 \times 3RC \)

- Delay has two components: \( d = f + p \)
  - \( f \): effort delay \( = g \cdot h \) (a.k.a. stage effort)
    - again has two components:
      - \( g \): logical effort
        - measures relative ability of gate to deliver current
        - \( g=1 \) for inverter (baseline circuit)
      - \( h \): electrical effort \( = \frac{C_{out}}{C_{in}} \)
        - ratio of output to input capacitance
        - sometimes called fanout
  - \( p \): parasitic delay
    - represents delay of gate driving no load
    - set by internal parasitic capacitance
Delay Plots

\[ d = f + p \]
\[ h = \frac{C_{out}}{C_{in}} \]
\[ d = g + 1 \]

Note: In these slides it is assumed that the MOSFET capacitance model is \( C_{in} = C_{out} = C \), and that \( W_p = 2W_n \) for equal drive strengths for the PMOS and NMOS in the inverter.
Delay Plots

- Delay: $d = f + p$
- $= gh + p$

Note: In these slides it is assumed that the MOSFET capacitance model is $C_{in} = C_{out} = C$, and that $W_p = 2W_n$ for equal drive strengths for the PMOS and NMOS in the inverter.
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 300 ps in 0.5 \( \mu \)m process 15 ps in a 65 nm process
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 300 ps in 0.5 \( \mu \)m process 15 ps in a 65 nm process
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 300 ps in 0.5 \( \mu \)m process 15 ps in a 65 nm process
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 300 ps in 0.5 μm process 15 ps in a 65 nm process
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 300 ps in 0.5 \( \mu \)m process 15 ps in a 65 nm process
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 300 ps in 0.5 $\mu$m process 15 ps in a 65 nm process
Multistage Logic Circuits

- 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 \( F = \prod f_i = \prod g_i h_i \)
- For a single path (no branching): \( F = G \cdot H \)

```
g_1 = 1
h_1 = x/10
g_2 = 5/3
h_2 = y/x
g_3 = 4/3
h_3 = z/y
g_4 = 1
h_4 = 20/z
```
- 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 \( F = \prod f_i = \prod g_i h_i \)
- For a single path (no branching): \( F = G \cdot H \)
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 \[ F = \prod f_i = \prod g_i h_i \]

For a single path (no branching): \[ F = G \cdot H \]
Multistage Logic Circuits

- 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 \( F = \prod f_i = \prod g_i h_i \)
- For a single path (no branching): \( F = G \cdot H \)
Multistage Logic Circuits

- Logical effort generalizes to multistage networks
- Path Logical Effort \( G = \prod g_i \)
- Path Electrical Effort \( H = \frac{C_{out\text{-path}}}{C_{in\text{-path}}} \)
- Path Effort \( F = \prod f_i = \prod g_i h_i \)
- For a single path (no branching): \( F = G \cdot H \)
Multistage Delays

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

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

- $D = \sum d_i = D_F + P$
- Delay is smallest when each stage bears same effort
  - $\hat{f} = g_i h_i = F_1^\frac{1}{N}$
- Thus minimum delay of N stage path is
  - $D = NF_1^\frac{1}{N} + P$
- 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?
  - \( \hat{f} = gh = g \frac{C_{out}}{C_{in}} \)
  - \( C_{in} = g_i \frac{C_{out}}{\hat{f}} \)

- Working backward, apply capacitance transformation to find input capacitance of each gate given load it drives.

- Check work by verifying input cap spec is met.
Gate Sizes

■ How wide should the gates be for least delay?

\[ \hat{f} = gh = g \frac{C_{out}}{C_{in}} \]

\[ C_{in_i} = \frac{g_i C_{out}}{\hat{f}} \]

■ Working backward, apply capacitance transformation to find input capacitance of each gate given load it drives.

■ Check work by verifying input cap spec is met.
Gate Sizes

How wide should the gates be for least delay?

\[ \hat{f} = gh = g \frac{C_{\text{out}}}{C_{\text{in}}} \]

\[ C_{\text{in}} = \frac{g_i C_{\text{out}}}{\hat{f}} \]

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.
Gate Sizes

- How wide should the gates be for least delay?
  - \( \hat{f} = gh = g \frac{C_{out}}{C_{in}} \)
  - \( C_{in_i} = \frac{g_i C_{out}}{\hat{f}} \)

- Working backward, apply capacitance transformation to find input capacitance of each gate given load it drives.

- Check work by verifying input cap spec is met.
Gate Sizes

- How wide should the gates be for least delay?
  - \( \hat{f} = gh = g \frac{C_{out}}{C_{in}} \)
  - \( C_{in_i} = \frac{g_i C_{out}}{\hat{f}} \)

- Working backward, apply capacitance transformation to find input capacitance of each gate given load it drives.

- Check work by verifying input cap spec is met.
Buffer Design: Best Number of Stages

- How many stages should a buffer use?
  - Minimizing number of stages is not always fastest
  - Example: drive $64 \times C$ load with unit inverter
    - $D = NF^{\frac{1}{2}} + P$
    - $N = N(64)^{\frac{1}{2}} + N$
Buffer Design: Best Number of Stages

- How many stages should a buffer use?
  - Minimizing number of stages is not always fastest
- Example: drive $64 \times C$ load with unit inverter
  - $D = NF^{\frac{1}{2}} + P$
  - $D = N(64)^{\frac{1}{2}} + N$

![Diagram showing buffer stages and load](image)
Buffer Design: Best Number of Stages

- How many stages should a buffer use?
  - Minimizing number of stages is not always fastest
- Example: drive $64 \times C$ load with unit inverter
  - $D = NF^{1/N} + P$
  - $= N(64)^{1/N} + N$
Buffer Design: Best Number of Stages

- How many stages should a buffer use?
  - Minimizing number of stages is not always fastest
- Example: drive $64 \times C$ load with unit inverter
  - $D = NF^{\frac{1}{N}} + P$
  - $= N(64)^{\frac{1}{N}} + N$
Buffer Design: Best Number of Stages

- How many stages should a buffer use?
  - Minimizing number of stages is not always fastest
- Example: drive $64 \times C$ load with unit inverter
  - $D = NF^{\frac{1}{N}} + P$
  - $= N(64)^{\frac{1}{N}} + N$
Buffer Design: Best Number of Stages

- How many stages should a buffer use?
  - Minimizing number of stages is not always fastest

- Example: drive $64 \times C$ load with unit inverter
  - $D = NF^{\frac{1}{N}} + P$
  - $= N(64)^{\frac{1}{N}} + N$
How many stages should a buffer use?

- Minimizing number of stages is not always fastest

Example: drive $64 \times C$ load with unit inverter

- $D = NF^{1/N} + P$
- $= N(64)^{1/N} + N$
Buffer Design: Best Number of Stages

- How many stages should a buffer use?
  - Minimizing number of stages is not always fastest

- Example: drive $64 \times C$ load with unit inverter
  - $D = NF^{1/N} + P$
  - $= N(64)^{1/N} + N$
Buffer Design: Best Number of Stages

- How many stages should a buffer use?
  - Minimizing number of stages is not always fastest
- Example: drive $64 \times C$ load with unit inverter
  - $D = NF^\frac{1}{N} + P$
  - $= N(64)^\frac{1}{N} + N$
Buffer Design: Best Number of Stages

- How many stages should a buffer use?
- Minimizing number of stages is not always fastest
- Example: drive $64 \times C$ load with unit inverter

$$D = NF \frac{1}{N} + P$$
$$= N(64) \frac{1}{N} + N$$

<table>
<thead>
<tr>
<th>N</th>
<th>f</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>64</td>
<td>65</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>18</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>2.8</td>
<td>15.3</td>
</tr>
</tbody>
</table>
Buffer Design: Best Number of Stages

- How many stages should a buffer use?
  - Minimizing number of stages is not always fastest
- Example: drive $64 \times C$ load with unit inverter
  - $D = NF^{\frac{1}{N}} + P$
  - $= N(64)^{\frac{1}{N}} + N$
Derivation

- How many inverters in a buffer give the least delay?
  - For $N$ inverters: $D = NF^\frac{1}{N} + N \cdot p_{inv}$
    - $p_{inv}$ is the parasitic delay of the inverter, $F$ is the path effort
    - Path Effort: $F = G \cdot H = \frac{C_{out}}{C_{in}}$
  - Minimize delay: $\frac{\partial D}{\partial N} = -F^\frac{1}{N} \cdot ln \left( F^\frac{1}{N} \right) + F^\frac{1}{N} + p_{inv} = 0$
  - Define best stage effort $\rho = F^\frac{1}{N}$
    - $p_{inv} + \rho(1 - ln\rho) = 0$
How many inverters in a buffer give the least delay?

For \( N \) inverters: \( D = NF^{\frac{1}{N}} + N \cdot p_{inv} \)

- \( p_{inv} \) is the parasitic delay of the inverter, \( F \) is the path effort
- Path Effort: \( F = G \cdot H = \frac{C_{out}}{C_{in1}} \)

Minimize delay: \( \frac{\partial D}{\partial N} = -F^{\frac{1}{N}} \cdot \ln \left( F^{\frac{1}{N}} \right) + F^{\frac{1}{N}} + p_{inv} = 0 \)

Define best stage effort: \( \rho = F^{\frac{1}{N}} \)

- \( p_{inv} + \rho(1 - \ln \rho) = 0 \)
How many inverters in a buffer give the least delay?

For $N$ inverters: $D = NF^{1/N} + N \cdot p_{inv}$

- $p_{inv}$ is the parasitic delay of the inverter, $F$ is the path effort.
- Path Effort: $F = G \cdot H = \frac{C_{out}}{C_{in1}}$

Minimize delay: $\frac{\partial D}{\partial N} = -F^{1/N} \cdot \ln \left( F^{1/N} \right) + F^{1/N} + p_{inv} = 0$

Define best stage effort: $\rho = F^{1/N}$

$\quad p_{inv} + \rho (1 - \ln \rho) = 0$
How many inverters in a buffer give the least delay?

For $N$ inverters: $D = NF^{\frac{1}{N}} + N \cdot p_{inv}$

- $p_{inv}$ is the parasitic delay of the inverter, $F$ is the path effort
- Path Effort: $F = G \cdot H = \frac{C_{out}}{C_{in1}}$

Minimize delay: $\frac{\partial D}{\partial N} = -\frac{1}{N} F^{\frac{1}{N}} \cdot \ln \left( F^{\frac{1}{N}} \right) + F^{\frac{1}{N}} + p_{inv} = 0$

Define best stage effort $\rho = F^{\frac{1}{N}}$

- $p_{inv} + \rho (1 - \ln \rho) = 0$
Derivation

- How many inverters in a buffer give the least delay?
- For $N$ inverters: $D = NF^{\frac{1}{N}} + N \cdot p_{inv}$
  - $p_{inv}$ is the parasitic delay of the inverter, $F$ is the path effort
  - Path Effort: $F = G \cdot H = \frac{C_{out}}{C_{in1}}$

- Minimize delay: $\frac{\partial D}{\partial N} = -F^{\frac{1}{N}} \cdot ln \left( F^{\frac{1}{N}} \right) + F^{\frac{1}{N}} + p_{inv} = 0$

- Define best stage effort: $\rho = F^{\frac{1}{N}}$
  - $p_{inv} + \rho(1 - ln\rho) = 0$
How many inverters in a buffer give the least delay?

For \( N \) inverters: \( D = NF^{1/N} + N \cdot p_{\text{inv}} \)

- \( p_{\text{inv}} \) is the parasitic delay of the inverter, \( F \) is the path effort
- Path Effort: \( F = G \cdot H = \frac{C_{\text{out}}}{C_{\text{in1}}} \)

Minimize delay: \( \frac{\partial D}{\partial N} = -F^{1/N} \cdot \ln \left( F^{1/N} \right) + F^{1/N} + p_{\text{inv}} = 0 \)

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

- \( p_{\text{inv}} + \rho(1 - \ln \rho) = 0 \)
Derivation

How many inverters in a buffer give the least delay?

For \( N \) inverters: \( D = NF^\frac{1}{N} + N \cdot p_{\text{inv}} \)

- \( p_{\text{inv}} \) is the parasitic delay of the inverter, \( F \) is the path effort
- Path Effort: \( F = G \cdot H = \frac{C_{\text{out}}}{C_{\text{in}1}} \)

Minimize delay: \( \frac{\partial D}{\partial N} = -F^\frac{1}{N} \cdot \ln \left( F^\frac{1}{N} \right) + F^\frac{1}{N} + p_{\text{inv}} = 0 \)

Define best stage effort \( \rho = F^\frac{1}{N} \)

- \( p_{\text{inv}} + \rho(1 - \ln \rho) = 0 \)
Best Stage Effort

- \( p_{\text{inv}} + \rho(1 - \ln\rho) = 0 \) has no closed form solution
- Neglecting parasitics \( (p_{\text{inv}} = 0) \) we find \( \rho = e = 2.718 \)
- For \( p_{\text{inv}} = 1 \), numerical solution yields \( \rho = 3.59 \)
- Least delay for:
  - stage effort (or fan-out) equal to \( \rho = F^{\frac{1}{N}} = 4 \)
  - and when using \( \hat{N} = \log_{\rho} F \)
    \[
    \hat{N} = \log_{4} F = \log_{4} \left( \frac{C_{\text{out}}}{C_{\text{in}}} \right)
    \]
- Rule of thumb: Fan-out of 4 (FO4) stage effort results in fastest path
Best Stage Effort

- $p\text{\textsubscript{inv}} + \rho(1 - ln\rho) = 0$ has no closed form solution
- Neglecting parasitics ($p\text{\textsubscript{inv}} = 0$) we find $\rho = e = 2.718$
- For $p\text{\textsubscript{inv}} = 1$, numerical solution yields $\rho = 3.59$
- Least delay for:
  - stage effort (or fan-out) equal to $\rho = F^{1/2} = 4$
  - and when using $\hat{N} = log_{\rho}F$
  - $\hat{N} = log_{4}F = log_{4}\left(\frac{C_{out}}{C_{in}}\right)$
- Rule of thumb: Fan-out of 4 (FO4) stage effort results in fastest path
Best Stage Effort

- \( p_{inv} + \rho (1 - \ln \rho) = 0 \) has no closed form solution
- Neglecting parasitics (\( p_{inv} = 0 \)) we find \( \rho = e = 2.718 \)
- For \( p_{inv} = 1 \), numerical solution yields \( \rho = 3.59 \)
- Least delay for:
  - stage effort (or fan-out) equal to \( \rho = F^{\frac{1}{2}} = 4 \)
  - and when using \( \hat{N} = \log_{\rho} F \)
  - \( \hat{N} = \log_{4} F = \log_{4} \left( \frac{C_{out}}{C_{in}} \right) \)
- Rule of thumb: Fan-out of 4 (FO4) stage effort results in fastest path
Best Stage Effort

- \( p_{inv} + \rho (1 - \ln \rho) = 0 \) has no closed form solution
- Neglecting parasitics (\( p_{inv} = 0 \)) we find \( \rho = e = 2.718 \)
- For \( p_{inv} = 1 \), numerical solution yields \( \rho = 3.59 \)
- Least delay for:
  - stage effort (or fan-out) equal to \( \rho = F \frac{1}{\hat{N}} = 4 \)
  - and when using \( \hat{N} = \log_\rho F \)
  - \( = \log_4 F = \log_4 \left( \frac{C_{out}}{C_{in1}} \right) \)
- Rule of thumb: Fan-out of 4 (FO4) stage effort results in fastest path
Best Stage Effort

- $p_{inv} + \rho(1 - \ln \rho) = 0$ has no closed form solution
- Neglecting parasitics ($p_{inv} = 0$) we find $\rho = e = 2.718$
- For $p_{inv} = 1$, numerical solution yields $\rho = 3.59$
- Least delay for:
  - stage effort (or fan-out) equal to $\rho = F^{\frac{1}{N}} = 4$
  - and when using $\hat{N} = \log_{\rho} F$
  - $= \log_4 F = \log_4 \left( \frac{C_{out}}{C_{in1}} \right)$
- Rule of thumb: Fan-out of 4 (FO4) stage effort results in fastest path
Best Stage Effort

- \( p_{inv} + \rho(1 - \ln \rho) = 0 \) has no closed form solution
- Neglecting parasitics (\( p_{inv} = 0 \)) we find \( \rho = e = 2.718 \)
- For \( p_{inv} = 1 \), numerical solution yields \( \rho = 3.59 \)
- Least delay for:
  - stage effort (or fan-out) equal to \( \rho = F \frac{1}{\hat{N}} = 4 \)
  - and when using \( \hat{N} = \log_{\rho} F \)
    \[
    = \log_4 F = \log_4 \left( \frac{C_{out}}{C_{in1}} \right)
    \]
- Rule of thumb: Fan-out of 4 (FO4) stage effort results in fastest path
Best Stage Effort

- \( p_{inv} + \rho (1 - \ln \rho) = 0 \) has no closed form solution
- Neglecting parasitics (\( p_{inv} = 0 \)) we find \( \rho = e = 2.718 \)
- For \( p_{inv} = 1 \), numerical solution yields \( \rho = 3.59 \)
- Least delay for:
  - stage effort (or fan-out) equal to \( \rho = F \frac{1}{\hat{N}} = 4 \)
  - and when using \( \hat{N} = \log_{F} F \)
  - \( = \log_{4} F = \log_{4} \left( \frac{C_{out}}{C_{in1}} \right) \)
- Rule of thumb: Fan-out of 4 (FO4) stage effort results in fastest path
Best Stage Effort

- \( p_{inv} + \rho(1 - \ln\rho) = 0 \) has no closed form solution
- Neglecting parasitics \( (p_{inv} = 0) \) we find \( \rho = e = 2.718 \)
- For \( p_{inv} = 1 \), numerical solution yields \( \rho = 3.59 \)
- Least delay for:
  - stage effort (or fan-out) equal to \( \rho = F^{\frac{1}{N}} = 4 \)
  - and when using \( \hat{N} = \log_{\rho} F \)
  - \( = \log_4 F = \log_4 \left( \frac{C_{out}}{C_{in1}} \right) \)
- Rule of thumb: Fan-out of 4 (FO4) stage effort results in fastest path
Sensitivity Analysis

- How sensitive is delay to using exactly the best number of stages?
  - $2.4 < \rho < 6$ gives delay within 15% of optimal
  - We can be sloppy!
  - Common standard is $\rho = 4$

<table>
<thead>
<tr>
<th>$N/\hat{N}$</th>
<th>$D(N)/D(\hat{N})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>1.51</td>
</tr>
<tr>
<td>0.7</td>
<td>1.15</td>
</tr>
<tr>
<td>1.0</td>
<td>1.26</td>
</tr>
<tr>
<td>1.4</td>
<td></td>
</tr>
<tr>
<td>2.0</td>
<td></td>
</tr>
</tbody>
</table>

Graph showing $D(N)/D(\hat{N})$ for different values of $N/\hat{N}$.
How sensitive is delay to using exactly the best number of stages?

2.4 < \rho < 6 gives delay within 15% of optimal

- we can be sloppy!
- Common standard is \rho = 4
How sensitive is delay to using exactly the best number of stages?

- $2.4 < \rho < 6$ gives delay within 15% of optimal

- we can be sloppy!

- Common standard is $\rho = 4$
How sensitive is delay to using exactly the best number of stages?

- $2.4 < \rho < 6$ gives delay within 15% of optimal
- we can be sloppy!
- Common standard is $\rho = 4$
Note that for the buffer design problem: $G = B = 1$, $g_i = 1$, and $F = H = \frac{C_{out}}{C_{in_1}}$. 
Total transistor area can be roughly estimated as
\[ A = A_1 \sum_{i=0}^{N-1} (\hat{f})^N \], where \( A_1 \) is the area of the first inverter.

The area can be minimized for a specified delay \((D_0)\) by optimizing the following set of constraints

\[
\begin{align*}
\text{minimize} & \quad \frac{(\hat{f})^N - 1}{\hat{f} - 1} \\
\text{for} \quad & \quad D = P + N\hat{f} \leq D_0
\end{align*}
\]

A fan-out of 8 can be used as a good trade-off to reduce layout area when designing large buffers.
Minimizing Layout Area?

- Total transistor area can be roughly estimated as
  \[ A = A_1 \sum_{i=0}^{N-1} (\hat{f})^N \], where \( A_1 \) is the area of the first inverter.

- The area can be minimized for a specified delay \( (D_0) \) by optimizing the following set of constraints

  \[
  \text{minimize } \frac{(\hat{f})^N - 1}{\hat{f} - 1}
  \]

  for \( D = P + N\hat{f} \leq D_0 \)

- A fan-out of 8 can be used as a good trade-off to reduce layout area when designing large buffers.
Minimizing Layout Area?

- Total transistor area can be roughly estimated as
  \[ A = A_1 \sum_{i=0}^{N-1} (\hat{f})^i \], where \( A_1 \) is the area of the first inverter.

- The area can be minimized for a specified delay \( D_0 \) by optimizing the following set of constraints
  \[
  \text{minimize } \frac{(\hat{f})^N - 1}{\hat{f} - 1}
  \]
  for \( D = P + N\hat{f} \leq D_0 \)

- A fan-out of 8 can be used as a good trade-off to reduce layout area when designing large buffers.