# EEC 118 Lecture #8: CMOS Logic Transient Characteristics

**Rajeevan Amirtharajah**
**University of California, Davis**
**Jeff Parkhurst**
**Intel Corporation**

# Announcements

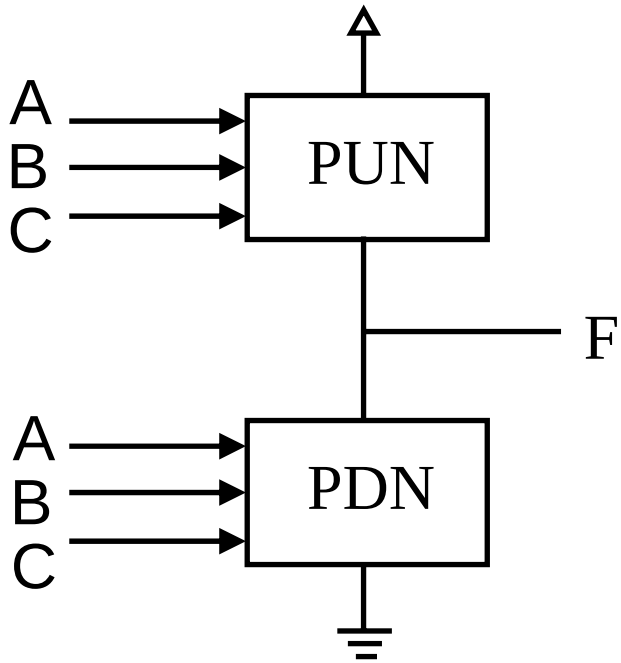
- **Quiz 2 on Monday, April 26**
- **Midterm on Monday, May 3**
  - Covers material through Lecture (Monday 4/26)
- **HW4 due Friday, 4PM in box, Kemper 2131**
- **Lab 3, Part 2 report due next week**

# Outline

- **Review: Static CMOS Logic**
- **Finish equivalent inverter discussion**
- **Combinational MOS Logic Circuits: Rabaey 6.1-6.2, 7.1-7.3 (Kang & Leblebici, 7.1-7.4)**

# Review: Static CMOS

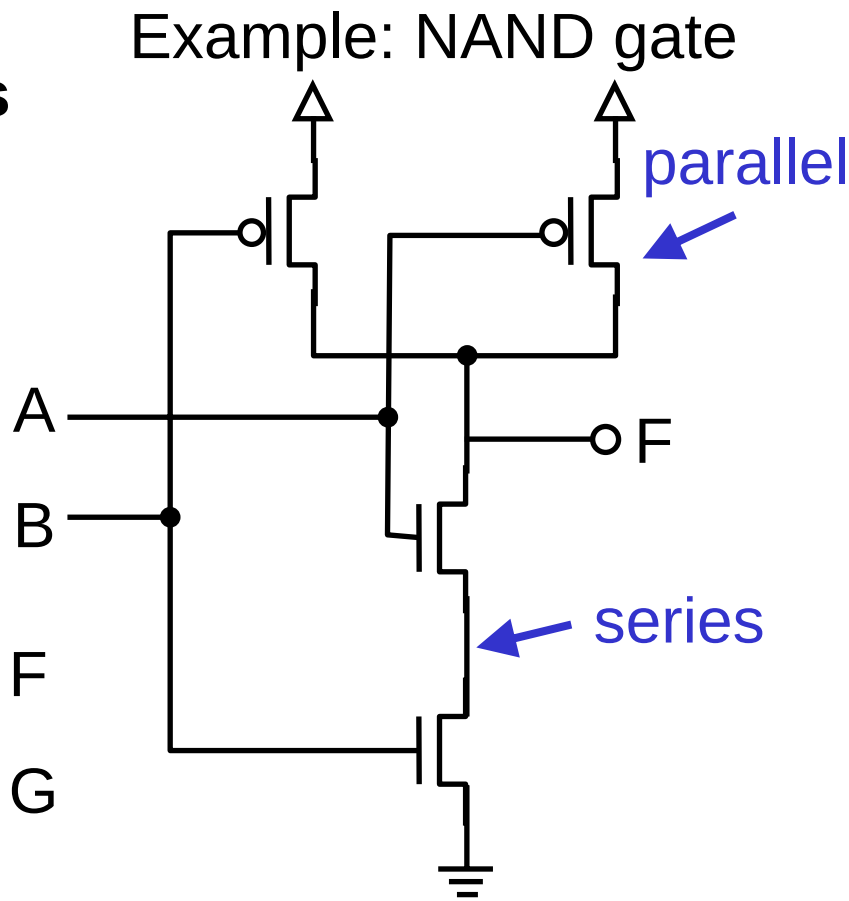
- **Complementary pullup network (PUN) and pulldown network (PDN)**
- **Only <u>one</u> network is on at a time**
- **PUN: PMOS devices**
  - Why? $V_{OH} = V_{DD}$
- **PDN: NMOS devices**
  - Why? $V_{OL} = 0$ V
- **PUN and PDN are *dual* networks**



# Review: Dual Networks

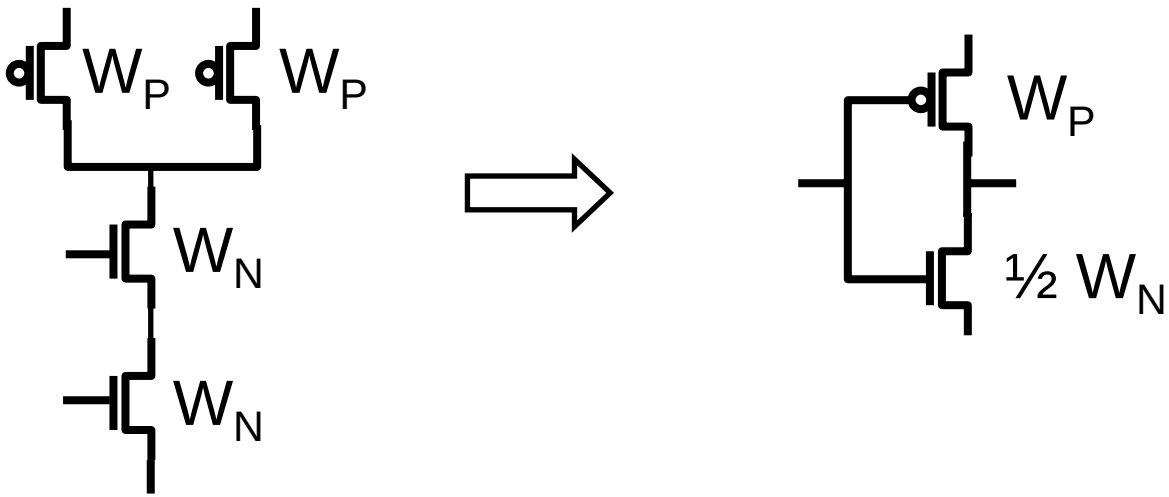
- **Dual networks: parallel connection in PDN = series connection in PUN, vice-versa**

- **If CMOS gate implements logic function F:**
  - PUN implements function F
  - PDN implements function G = $\overline{F}$



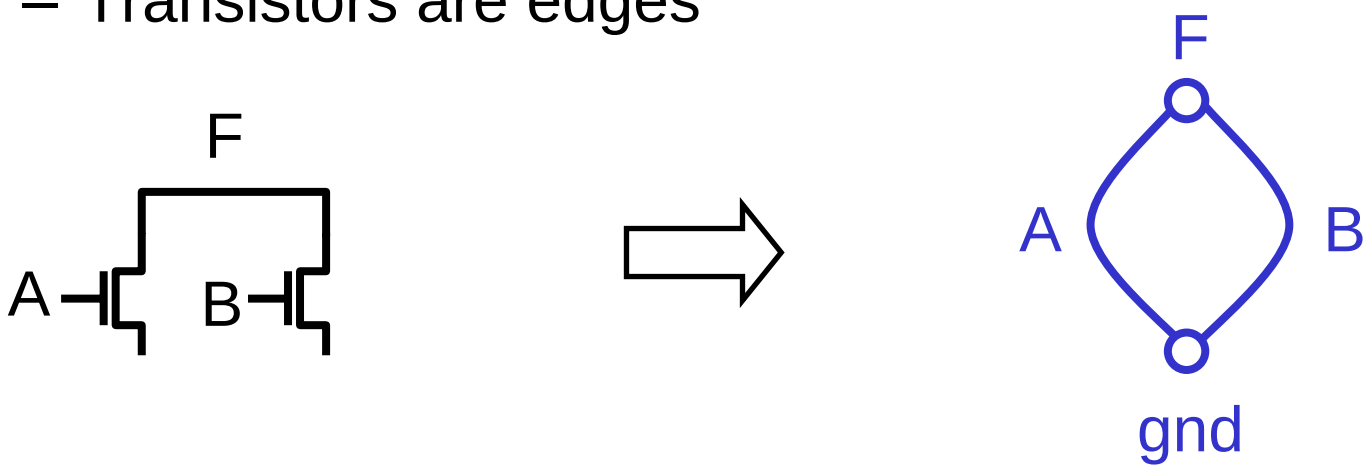
# Review: Equivalent Inverter

- **Represent complex gate as inverter for delay estimation, VTC analysis**
- **Use worse-case conditions for delays**
- **Example: NAND gate**
  - Worse-case (slowest) pull-up: only 1 PMOS “on”
  - Pull-down: both NMOS “on”



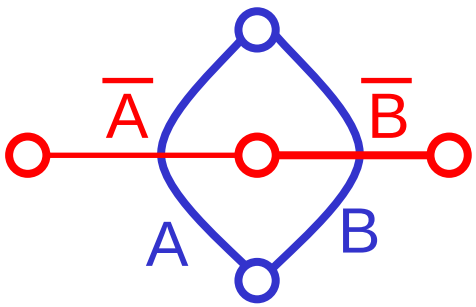
# Graph-Based Dual Network

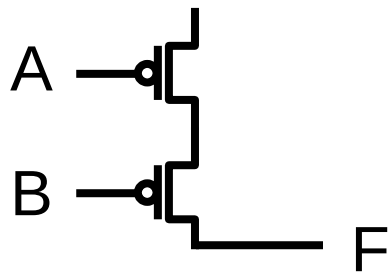
- **Use graph theory to help design gates**
  - Mostly implemented in CAD tools
- **Draw network for PUN or PDN**
  - Circuit nodes are vertices
  - Transistors are edges



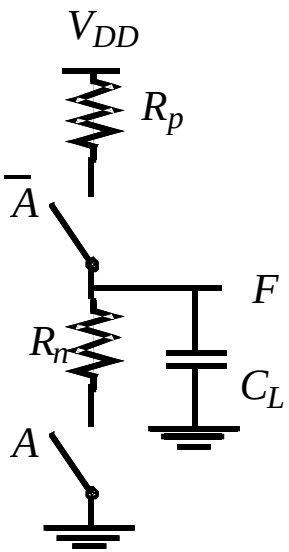
# Graph-Based Dual Network (2)

- **To derive dual network:**
  - Create new node in each enclosed region of graph
  - Draw new edge intersecting each original edge
  - Edge is controlled by inverted input

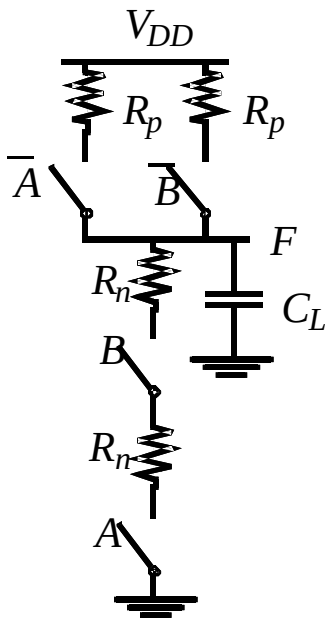


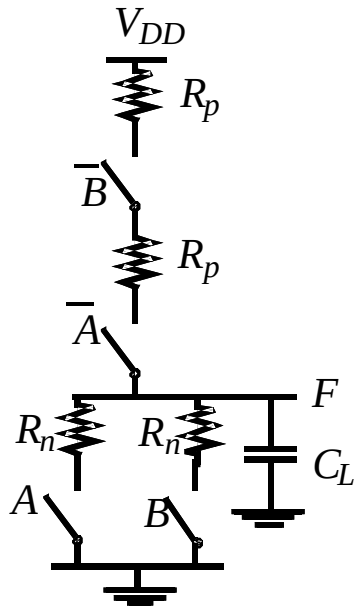
# Propagation Delay Analysis - The Switch Model


(a) Inverter




(b) 2-input NAND



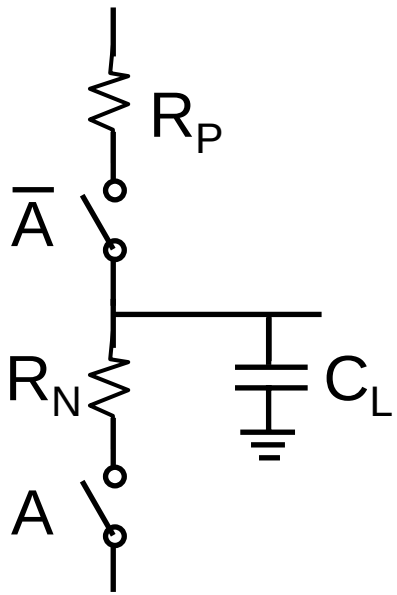

(c) 2-input NOR

$$t_p = 0.69\ R_{on}\ C_L$$

**(assuming that $C_L$ dominates!)**

# Switch Level Model

- **Model transistors as switches with series resistance**
- **Resistance $R_{on}$ = average resistance for a transition**
- **Capacitance $C_L$ = average load capacitance for a transition (same as we analyzed for transient inverter delays)**



# What is the Value of $R_{on}$?

- *Depends strongly on the operating region*
- *For hand analysis use a fixed value of R which it the average of the two end points of the transition*
- *Similar to the previous approach of averaging currents*

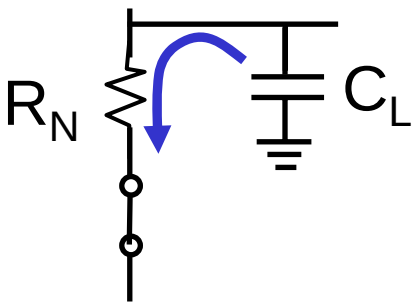
***EXAMPLE: For $t_{pHL}$ for an inverter, the $R_{on}$ is:***

$$R_{on} = \frac{1}{2}(R_{NMOS}(V_{out} = V_{DD}) + R_{NMOS}(V_{out} = V_{DD}/2))$$

$$= \frac{1}{2}\left(\left(\frac{V_{DS}}{I_D}\right)_{V_{out} = V_{DD}} + \left(\frac{V_{DS}}{I_D}\right)_{V_{out} = V_{DD}/2}\right)$$

# Switch Level Model Delays

**Delay estimation using switch-level model (for general RC circuit):**




$$I = C\frac{dV}{dt} \quad \rightarrow \quad dt = \frac{C}{I}dV$$

$$I = \frac{V}{R} \quad \rightarrow \quad dt = \frac{RC}{V}dV$$

$$t_1 - t_0 = t_p = \int_{V_0}^{V_1} \frac{RC}{V}dV$$

$$t_p = RC\left[\ln(V_1) - \ln(V_0)\right] = RC\ln\left(\frac{V_1}{V_0}\right)$$

# Switch Level Model RC Delays

- **For fall delay $t_{phl}$, $V_0=V_{DD}$, $V_1=V_{DD}/2$**

$$t_p = RC\ln\left(\frac{V_1}{V_0}\right) = RC\ln\left(\frac{\frac{1}{2}V_{DD}}{V_{DD}}\right)$$

$$t_p = RC\ln(0.5)$$

$$t_{phl} = 0.69R_nC_L$$

$$t_{plh} = 0.69R_pC_L$$

← Standard RC-delay equations from literature

# Numerical Examples

- **Example resistances for 1.2 μm CMOS**

$V_{DD} = 5V$, $W/Leff=1$ ($W/L_{eff}=2$ *is a minimum sized device 1.8μm/0.9μm*)
$L_{eff} = 1.2\mu m - 2(.15\mu m) = 0.9\mu m$

$R_n(W/L_{eff}= 2) = (5\text{ V} / 0.46\text{ mA} + 2.5\text{ V} / 0.29\text{ mA}) / 2 = 9.7\text{ k}\Omega$ (for $t_{pHL}$)

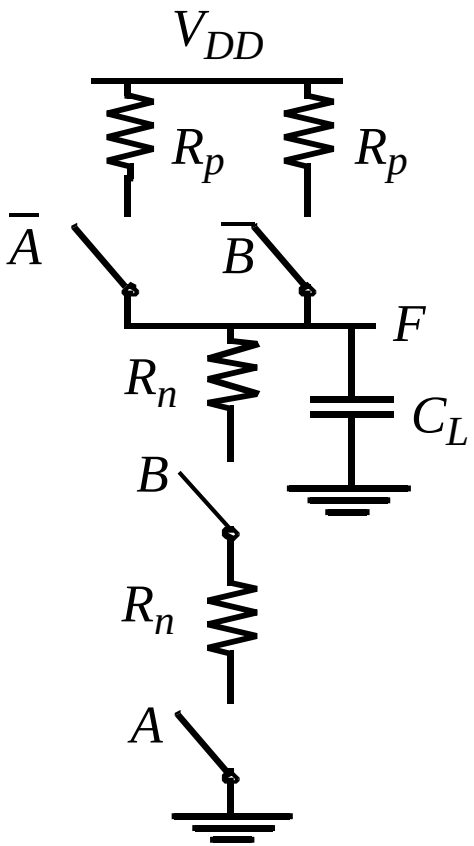
$R_n(W/L_{eff}= 1) = 9.7 * 2 = 19.4\text{ k}\Omega$ (for $t_{pHL}$)

$R_p(W/L_{eff}= 6) = (5\text{ V} / 0.57\text{ mA} + 2.5\text{ V} / 0.24\text{ mA}) / 2 = 9.6\text{ k}\Omega$ (for $t_{pLH}$)

$R_p(W/L_{eff}= 1) = 9.6 * 6 = 57.6\text{ k}\Omega$ (for $t_{pLH}$)

**SOLVE RC NETWORK TO DETERMINE DELAYS**

# Analysis of Propagation Delay




2-input NAND

1. Assume $R_n$=$R_p$= resistance of minimum sized NMOS inverter

2. Determine "Worst Case Input" transition (Delay depends on input values)

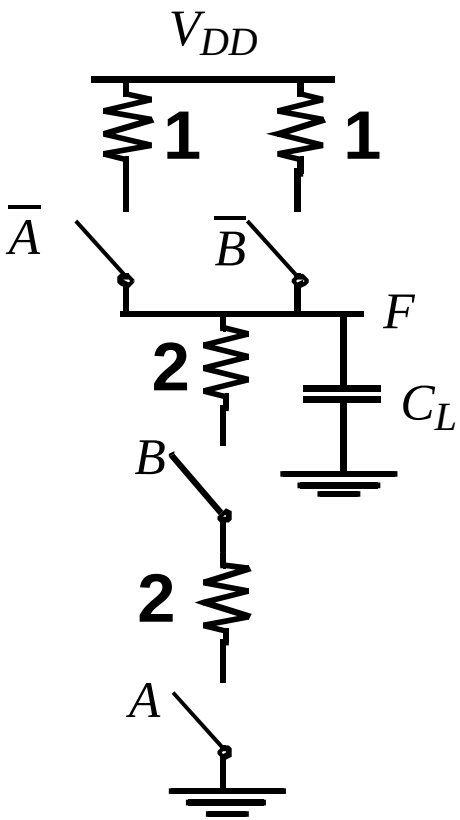
3. Example: $t_{pLH}$ for 2input NAND
   - Worst case when only ONE PMOS Pulls up the output node
   - For 2 PMOS devices in parallel, the resistance is lower

$$t_{pLH} = 0.69 R_p C_L$$

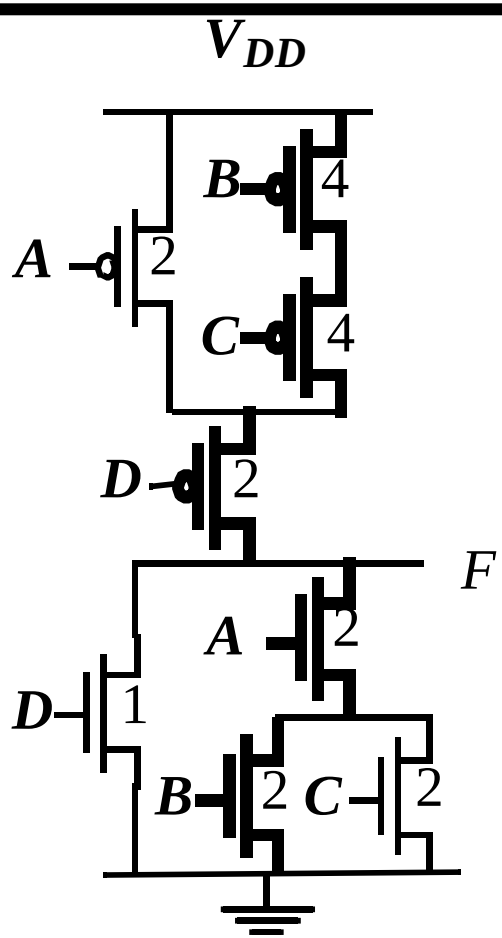
4. Example: $t_{pHL}$ for 2input NAND
   - Worst case : TWO NMOS in series

$$t_{pHL} = 0.69(2R_n) C_L$$

# Design for Worst Case



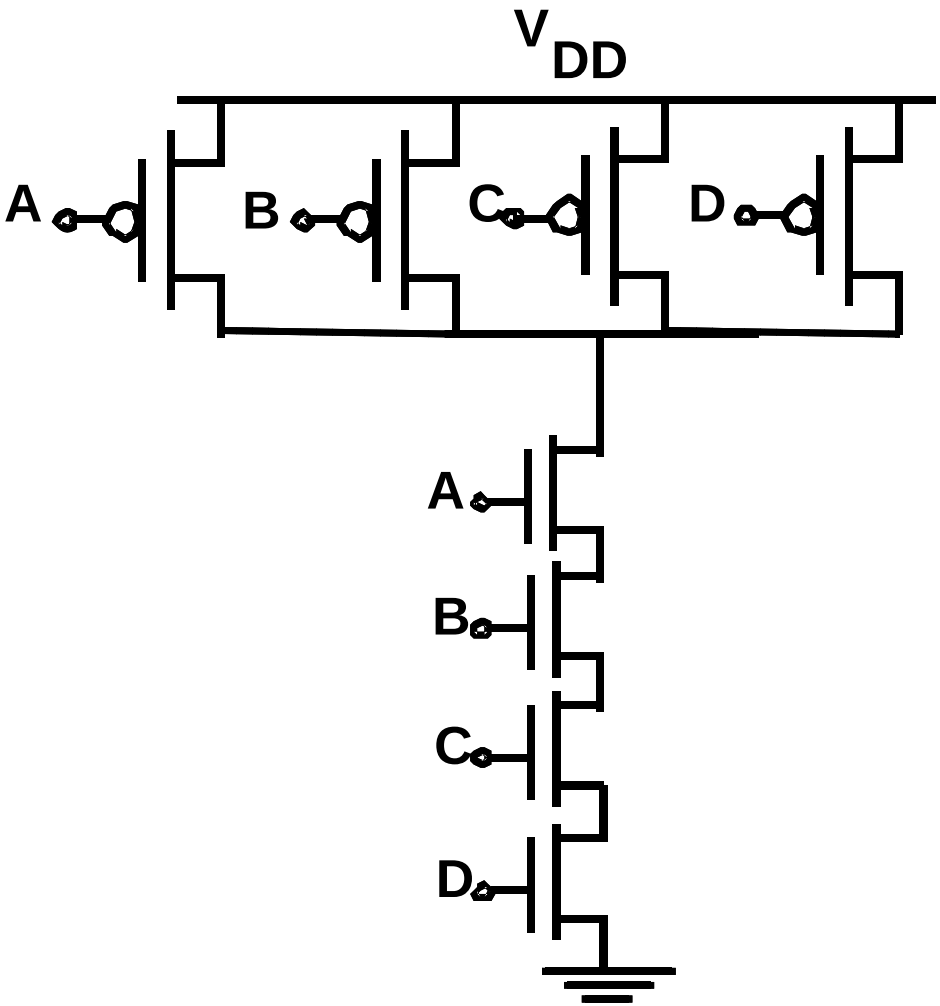

**NAND Gate**




**Complex Gate**

Here it is assumed that $R_p = R_n$

# Fan-In and Fan-Out




**Fan-Out**

Number of logic gates connected to output

(2 FET gate capacitances per fan-out)

**Fan-In**

Number of logical inputs

Quadratic delay term due to:

1. Resistance increasing
2. Capacitance increasing

for $t_{pHL}$ (series NMOS)

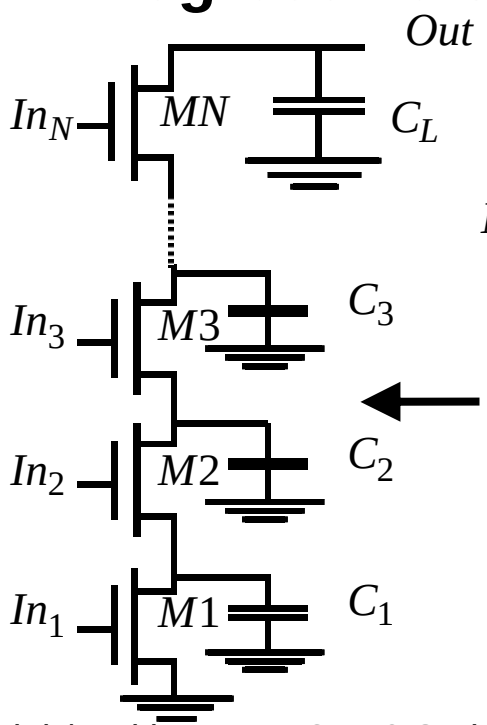
$t_p$ proportional to $a_1FI + a_2FI^2 + a_3FO$

# Fast Complex Gates - Design Techniques

- **Increase Transistor Sizing:**

  **Works as long as Fan-out capacitance dominates self capacitance (S/D cap increases with increased width)**

- **Progressive Sizing:**




$$M1 > M2 > M3 > MN$$

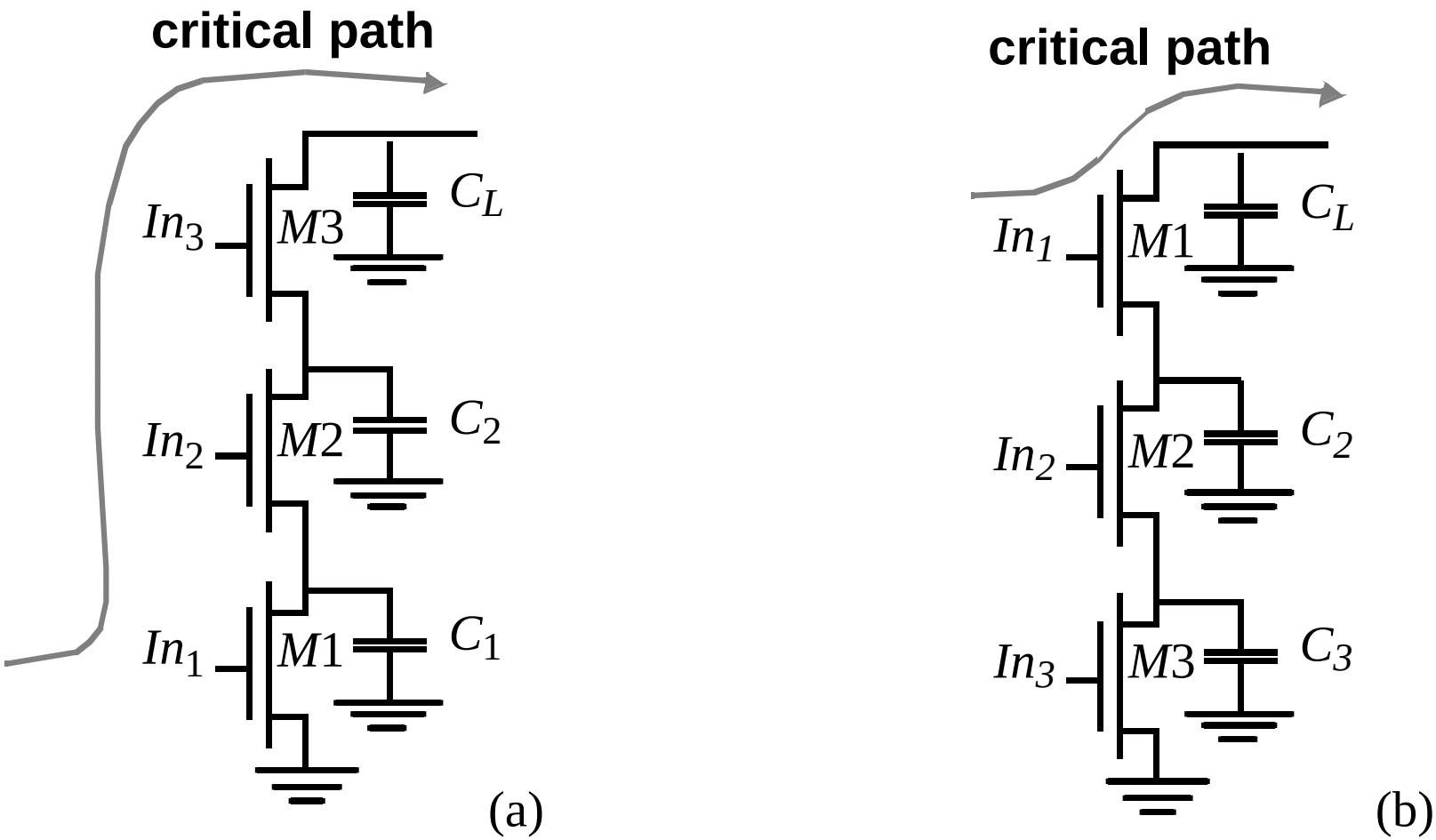
**Distributed RC-line**

**Can Reduce Delay by more than 30%!**

# Fast Complex Gates - Design Techniques (2)

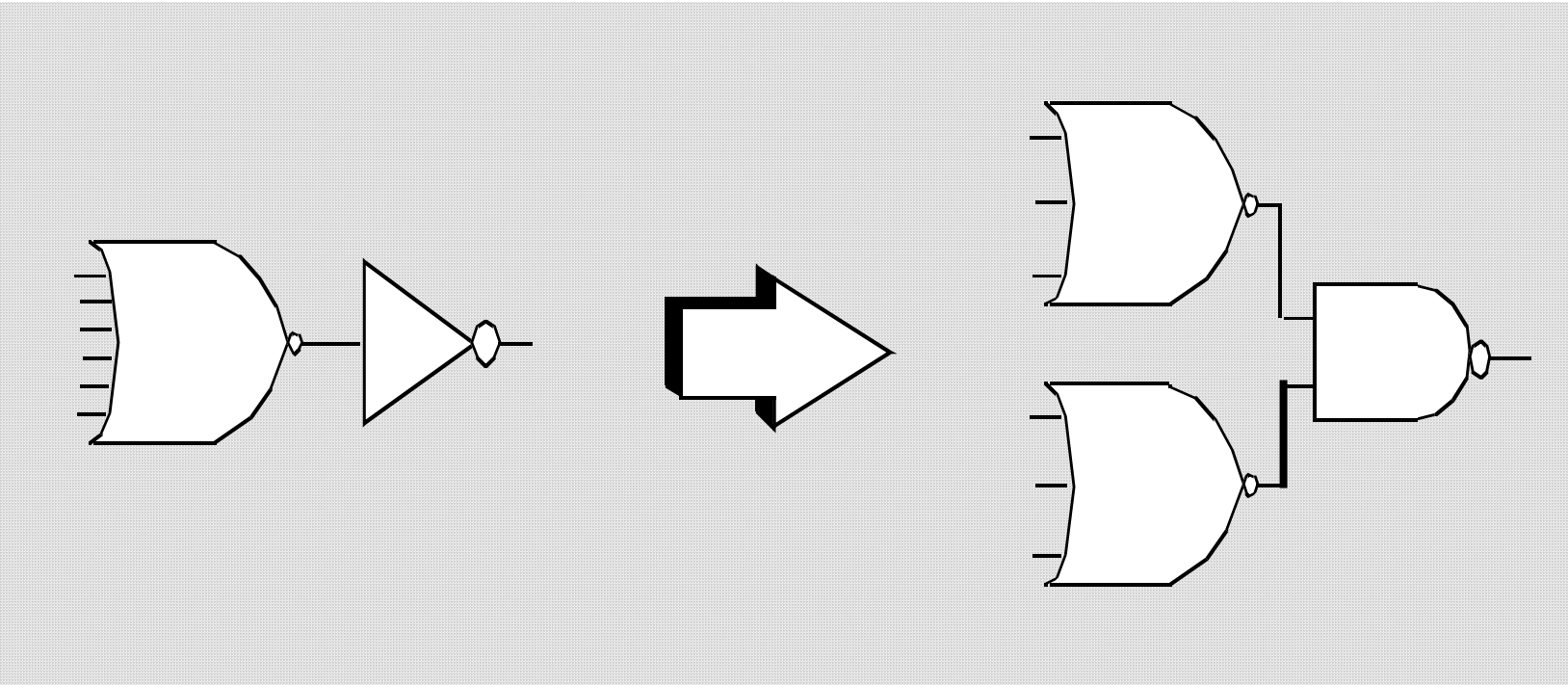
- **Transistor Ordering**

  **Place last arriving input closest to output node**



# Fast Complex Gates - Design Techniques (3)

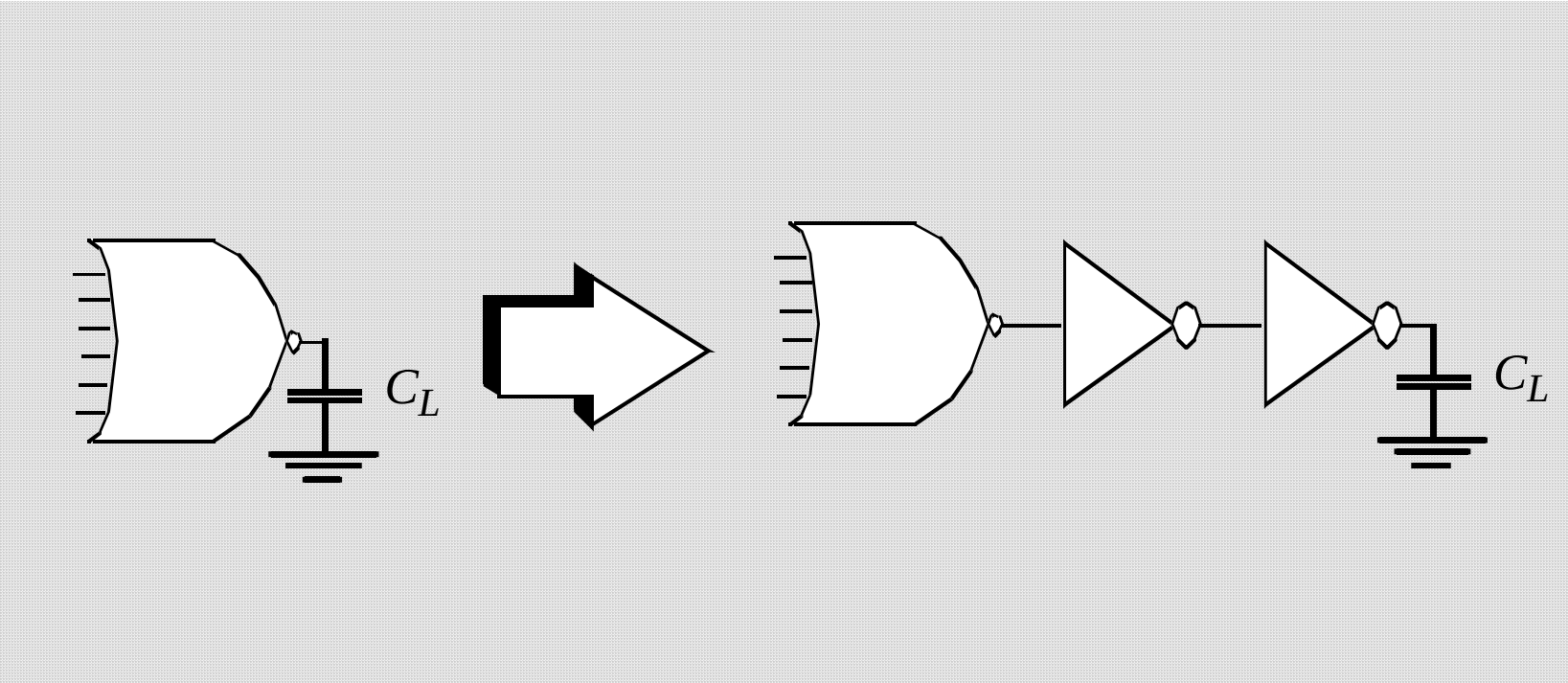
- **Improved Logic Design**



Note Fan-Out capacitance is the same, but Fan-In resistance lower for input gates (fewer series FETs)

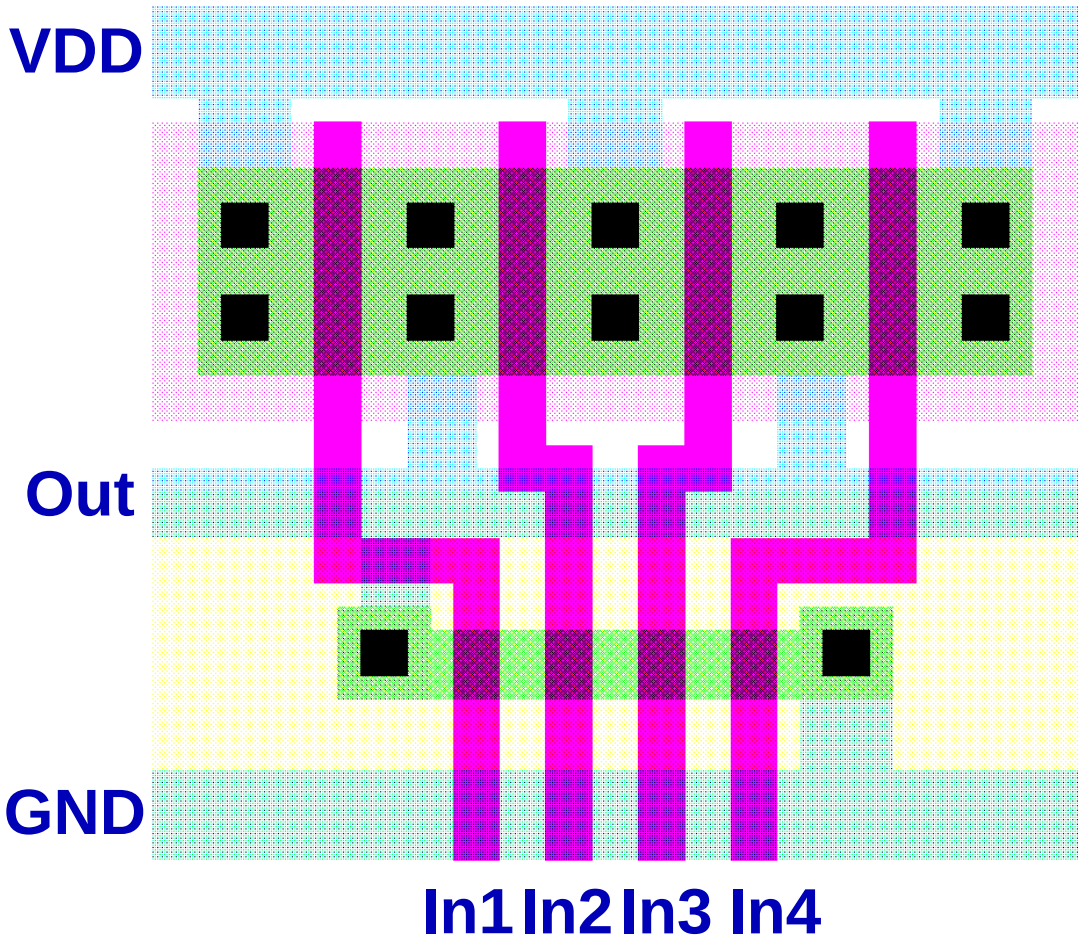
# Fast Complex Gates - Design Techniques (4)

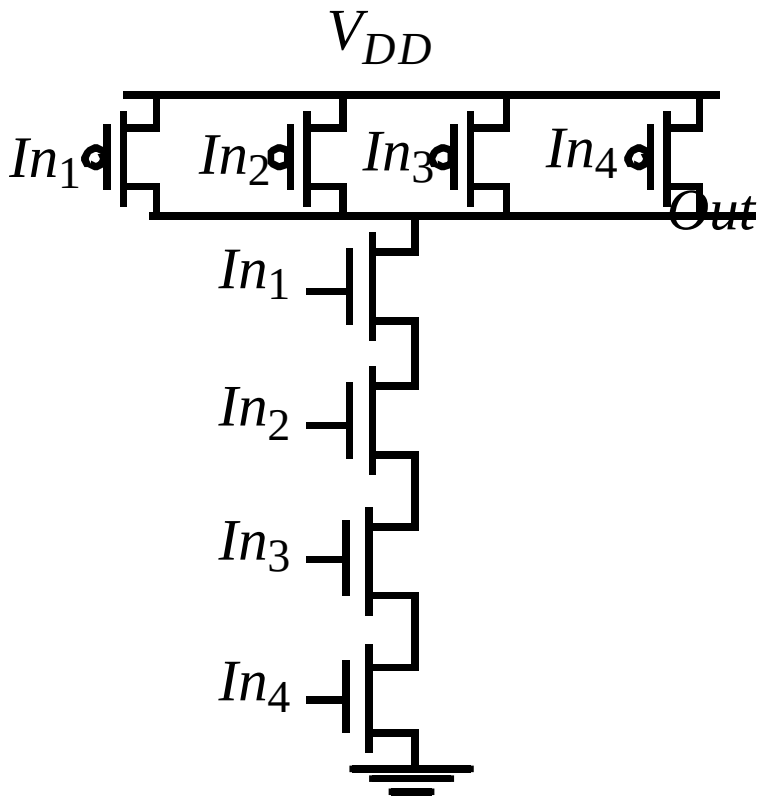
- **Buffering: Isolate Fan-in from Fan-out**



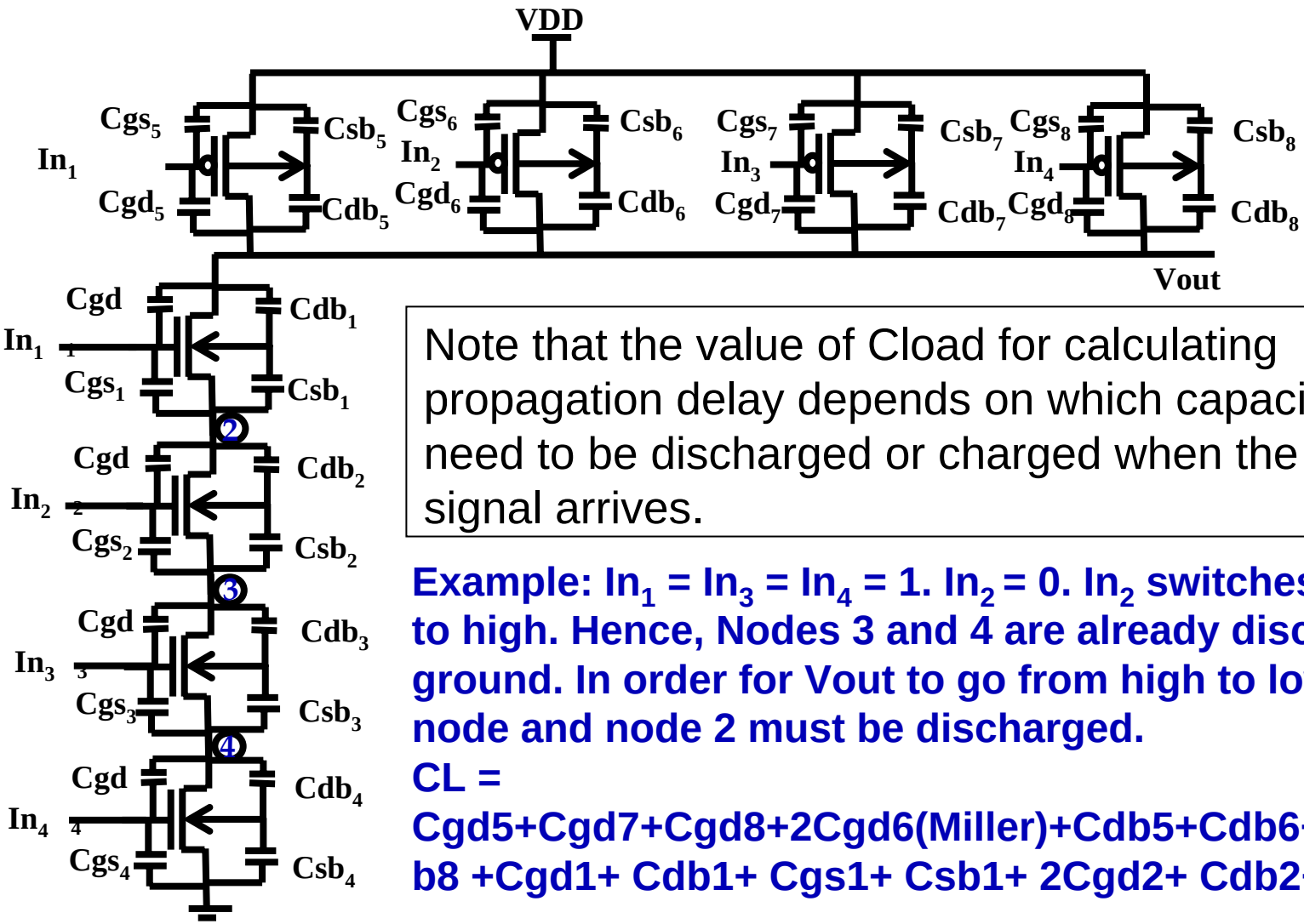

Keeps high fan-in resistance isolated from large capacitive load $C_L$

# 4 Input NAND Gate



# Capacitances in a 4 input NAND Gate




Note that the value of Cload for calculating propagation delay depends on which capacitances need to be discharged or charged when the critical signal arrives.

**Example: $In_1 = In_3 = In_4 = 1$. $In_2 = 0$. $In_2$ switches from low to high. Hence, Nodes 3 and 4 are already discharged to ground. In order for Vout to go from high to low… Vout node and node 2 must be discharged.**

**CL = Cgd5+Cgd7+Cgd8+2Cgd6(Miller)+Cdb5+Cdb6+Cdb7+Cdb8 +Cgd1+ Cdb1+ Cgs1+ Csb1+ 2Cgd2+ Cdb2+ Cw**

# Next Topic: Sequential Logic

- **Basic sequential circuits in CMOS**
  - RS latches, transparent latches, flip-flops
  - Alternative sequential element topologies
  - Pipelining