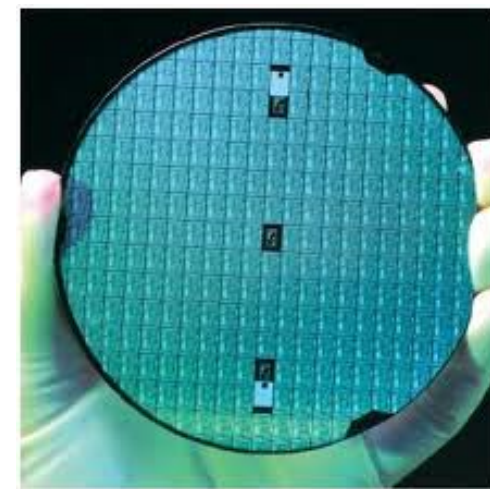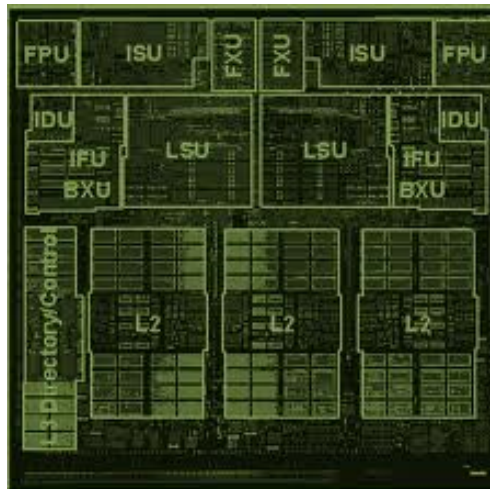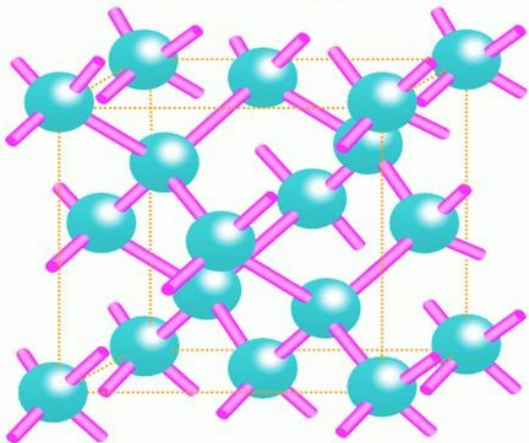Structure of silicon crystal

# *ECE 122A*
# *VLSI Principles*

## *Lecture 12*

Prof. Kaustav Banerjee
Electrical and Computer Engineering
University of California, Santa Barbara
*E-mail: kaustav@ece.ucsb.edu*

# Designing Combinational Logic Circuits

Kaustav Banerjee

# *Static Vs Dynamic Circuits*

**Static Circuits**

At every point in time (except during the switching transients) each **gate output is connected to either** $V_{DD}$ or $V_{ss}$ via a low-resistive path.

The outputs of the gates **assume at all times the value of the Boolean function**, implemented by the circuit (ignoring, once again, the transient effects during switching periods).

This is in contrast to the *dynamic* circuit class, which relies on temporary storage of signal values on the capacitance of high impedance circuit nodes.

*Resulting gate is simpler and faster, but increased sensitivity to noise….*

Kaustav Banerjee

# *Static and Dynamic CMOS Circuit Families*

- ❑ Static:
  - Complementary CMOS
    - (robustness, low power, large fan-in expensive in terms of area and performance)
  - Ratioed Logic: pseudo-NMOS, differential cascode voltage switch logic (DCVSL)
    - (simple and fast at the expense of reduced NM and static power)
  - Pass-Transistor Logic (Transmission Gate)
    - attractive for specific circuits: MUX, XOR-dominated logic such as Adders)
- ❑ Dynamic:
  - good for fast and complex gates, design process is harder due to parasitic effects, leakage puts an upper limit on the operating frequency of the circuit
    - Domino Logic
    - np-CMOS

*Which style is best?*

*……depends on ease of design, performance, power, area and robustness.*

# *Complementary CMOS Logic*

❑ Full rail-to-rail swing; high noise margins

❑ Logic levels not dependent upon the relative device sizes; ratioless

❑ Always a path to Vdd or Gnd in steady state; low output impedance

❑ Extremely high input resistance; nearly zero steady-state input current

❑ No direct path steady state between power and ground; no static power dissipation

❑ Propagation delay function of load capacitance and resistance of transistors

Kaustav Banerjee

# *Static Complementary CMOS*
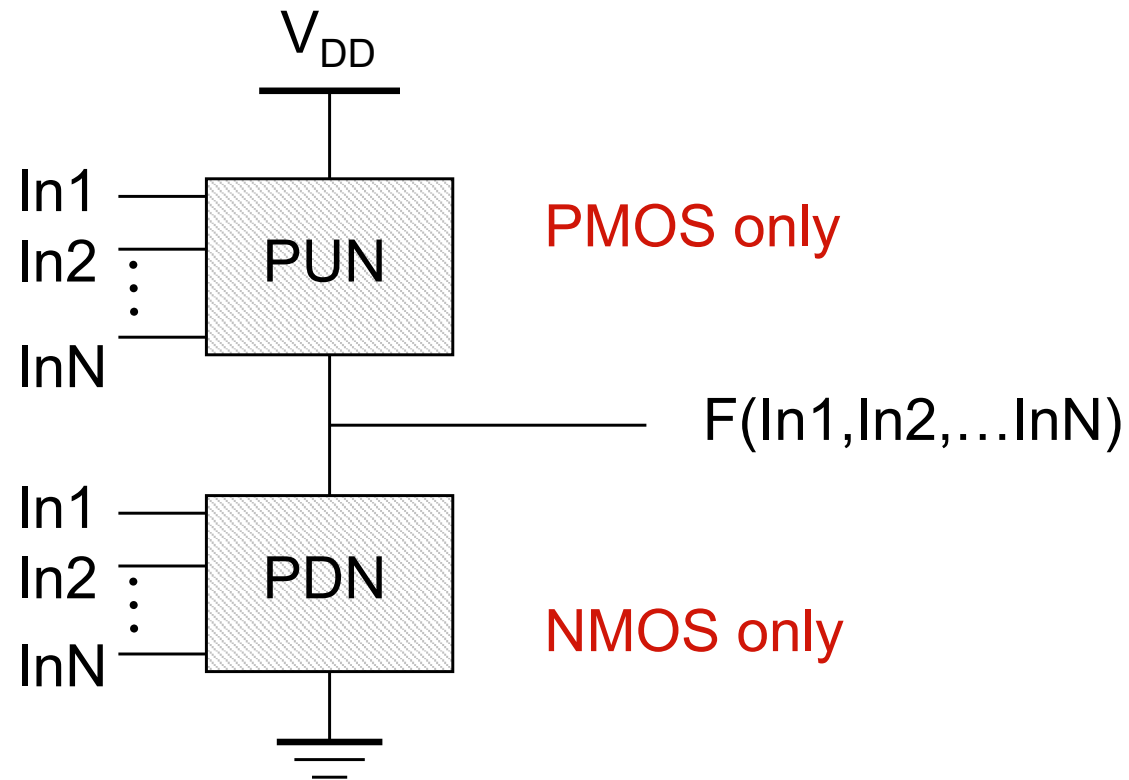


PUN and PDN are dual logic networks

Kaustav Banerjee

# *Dual Networks*

Example: NAND gate

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

❑ If CMOS gate implements logic function F:

- PUN implements function $\overline{\overline{F}}$

- PDN implements function G = F



parallel

A

B

F

series

Kaustav Banerjee
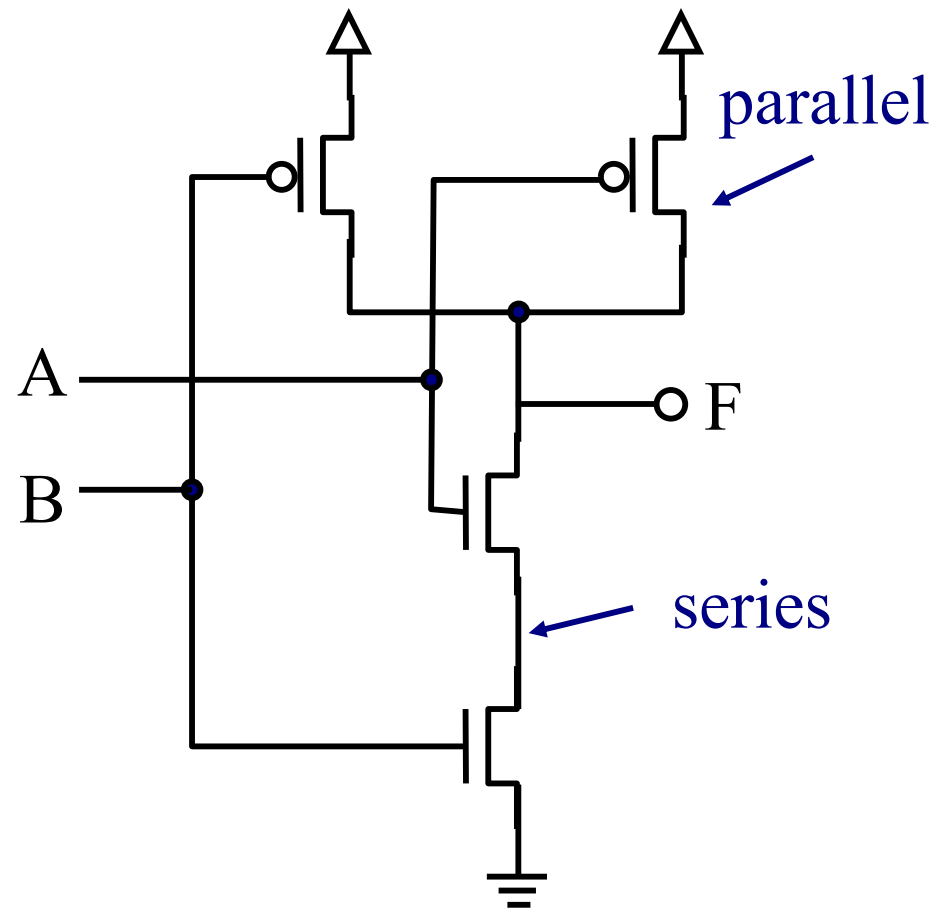
# Key Properties of Complementary CMOS Gates: Snapshot

**High noise margins**:

$V_{OH}$ and $V_{OL}$ are at $V_{DD}$ and $GND$, respectively.
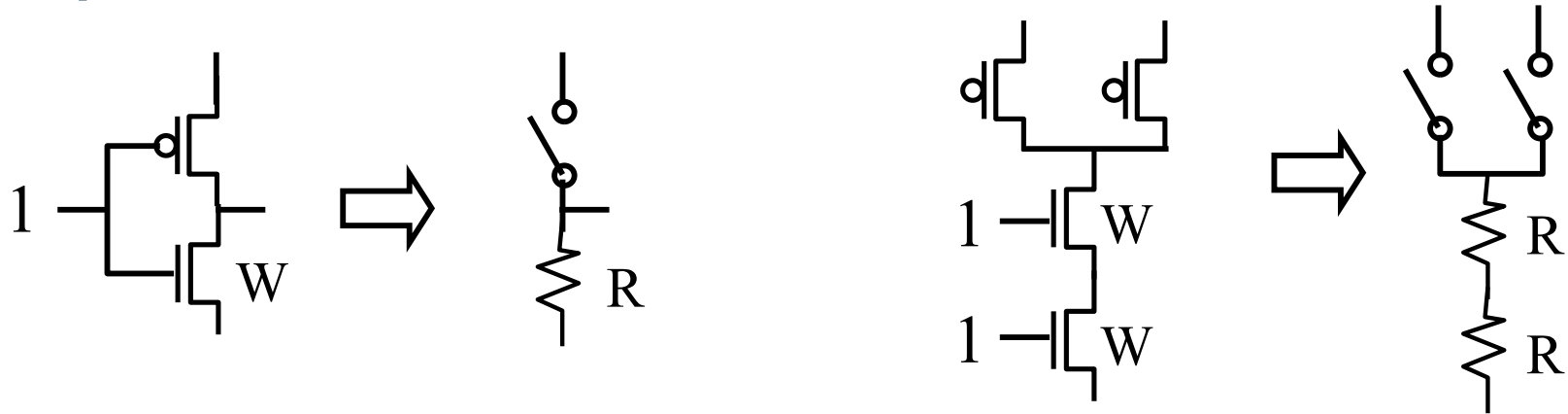
**No static power consumption**:

There never exists a direct path between $V_{DD}$ and $V_{SS}$ ($GND$) in steady-state mode.

**Comparable rise and fall times:**

(under appropriate sizing conditions)

Kaustav Banerjee

# *Analysis of CMOS gates*

❑ Represent "on" transistors as resistors



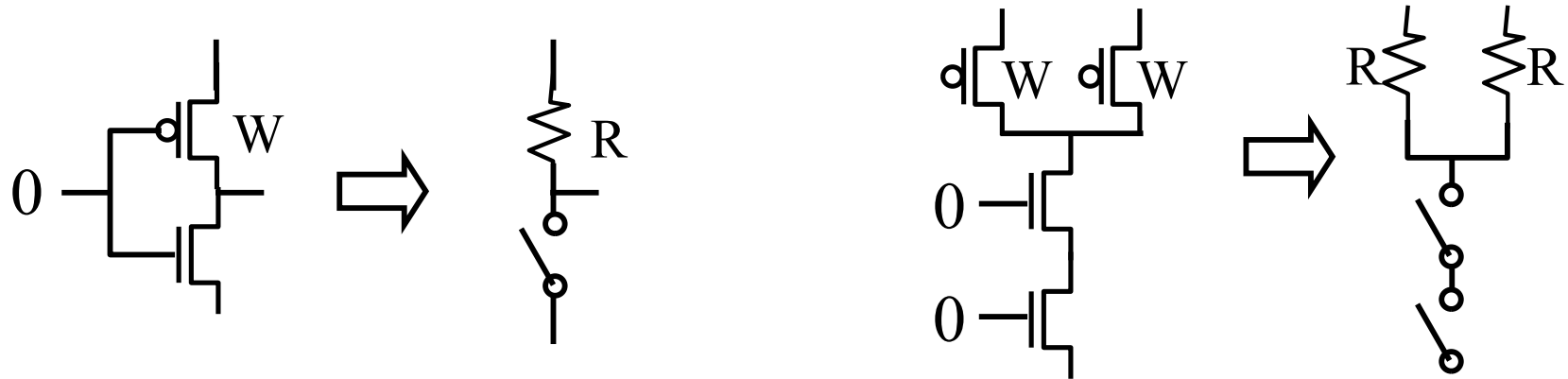- Transistors in series → resistances in series
  - Effective resistance = 2R
  - Effective width = ½ W

*Note: 1) As transistor width increases, its ON resistance (R) decreases*

*2) If transistor channel length increases, R increases*

# *Analysis of CMOS Gates, contd…*

- Represent "on" transistors as resistors



- Transistors in parallel $\rightarrow$ resistances in parallel
  - Effective resistance = ½ R
  - Effective width = 2W

Kaustav Banerjee

# *Equivalent Inverter*

- CMOS gates: many paths to Vcc and Gnd
  - Multiple values for $V_M$, $V_{IL}$, $V_{OL}$, etc
  - Different delays for each input combination
- Equivalent inverter
  - Represent each gate as an inverter with appropriate device width
  - Include only transistors which are ON or switching
  - Calculate $V_M$, delays, etc using inverter equations

Kaustav Banerjee

# *CMOS gates: equivalent inverter*

❑ Represent complex gate as inverter for delay estimation

❑ Use worst-case delays

❑ Example: NAND gate

- ▪ Worst-case (slowest) pull-up: only 1 PMOS "on"
- ▪ Pull-down: both NMOS "on"

Kaustav Banerjee
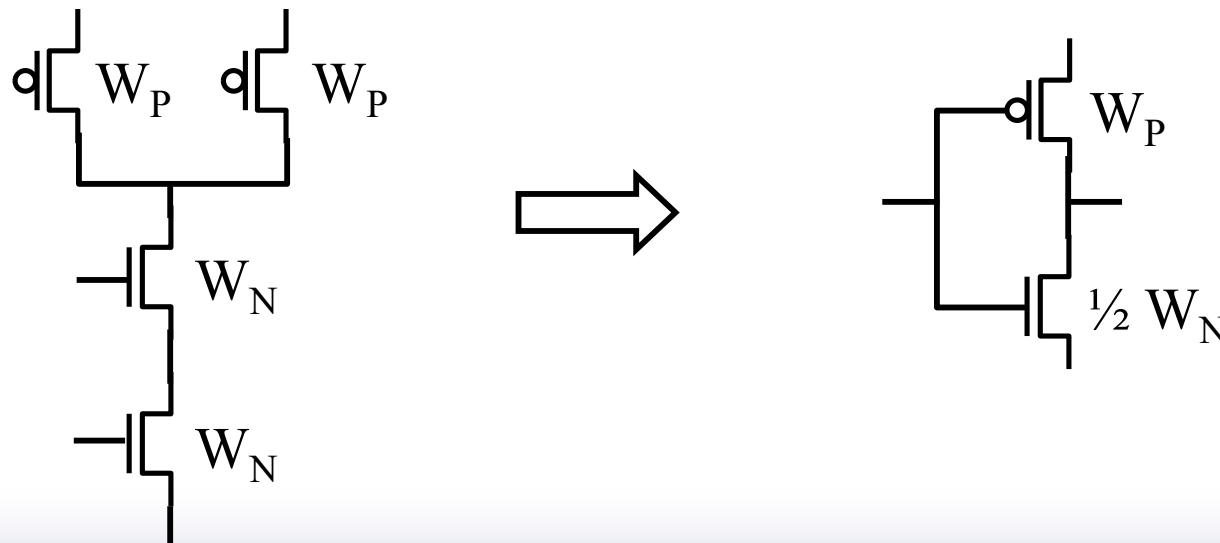
# *Equivalent Inverter: $V_M$*
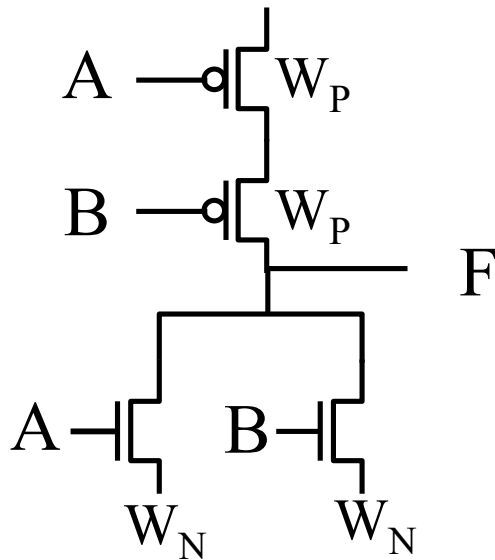
❑ Example: 2-input NAND gate threshold $V_M$
   Three possibilities:

   ▪ A & B switch together

   ▪ A switches alone

   ▪ B switches alone
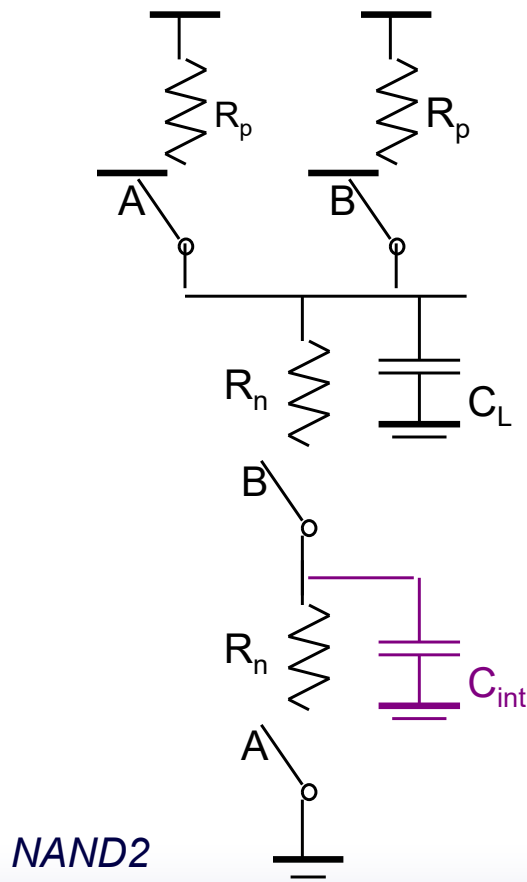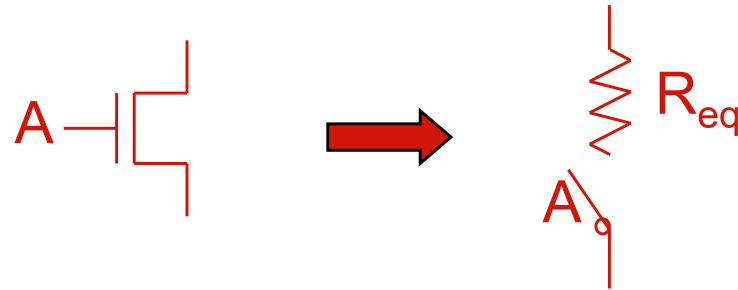
   *Hint: the VTC curve is data dependent*
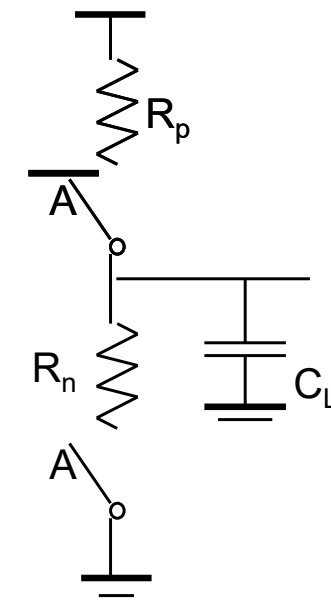
❑ What is equivalent inverter for each case?

Kaustav Banerjee

# *Example: NOR gate*



- ❏ Find threshold voltage $V_M$ when both inputs switch simultaneously

- ❏ Two methods:
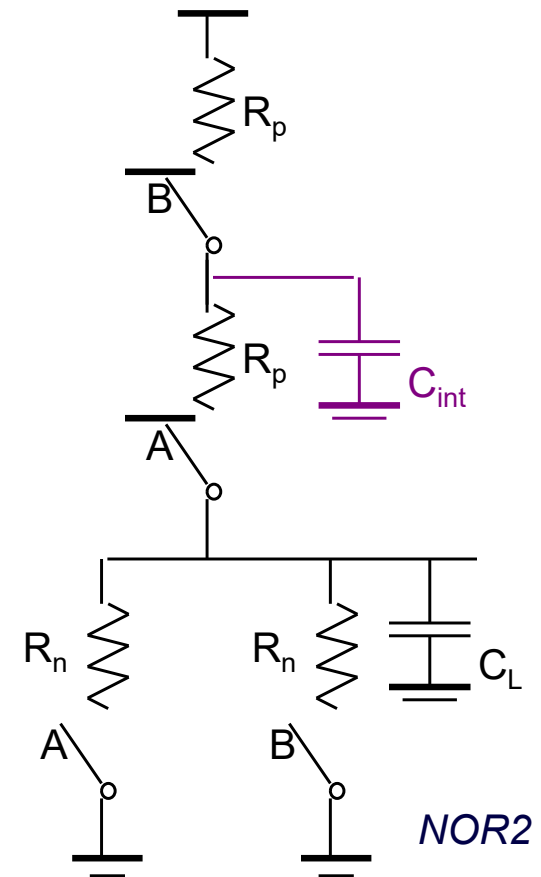  - Transistor equations
  - Equivalent inverter

Kaustav Banerjee

# Switch Delay Model



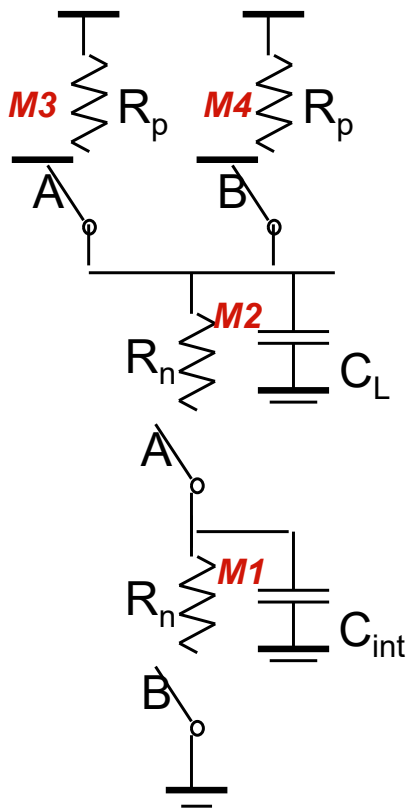NAND2

INV

NOR2

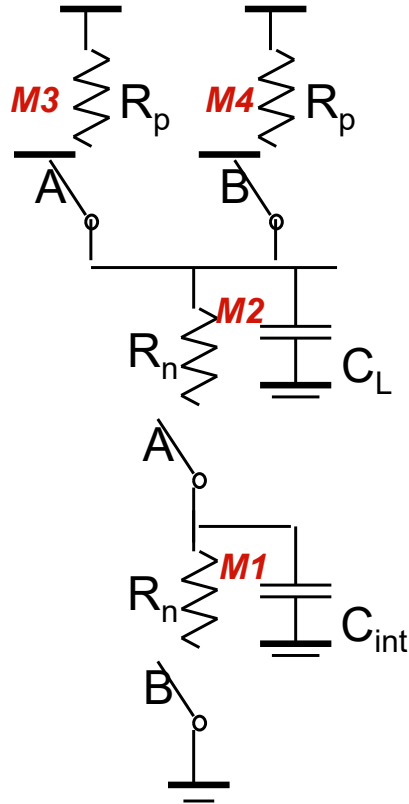# *Input Pattern Effects on Delay*



**2-input NAND**

- ❑ Delay is dependent on the <span style="color:red">pattern</span> of inputs
- ❑ Low to high transition
  - ▪ both inputs go low
    - – delay is $0.69\ R_p/2\ C_L$
  - ▪ one input goes low
    - – delay is $0.69\ R_p\ C_L$
- ❑ High to low transition
  - ▪ both inputs go high
    - – delay is $0.69\ 2R_n\ C_L$

Kaustav Banerjee
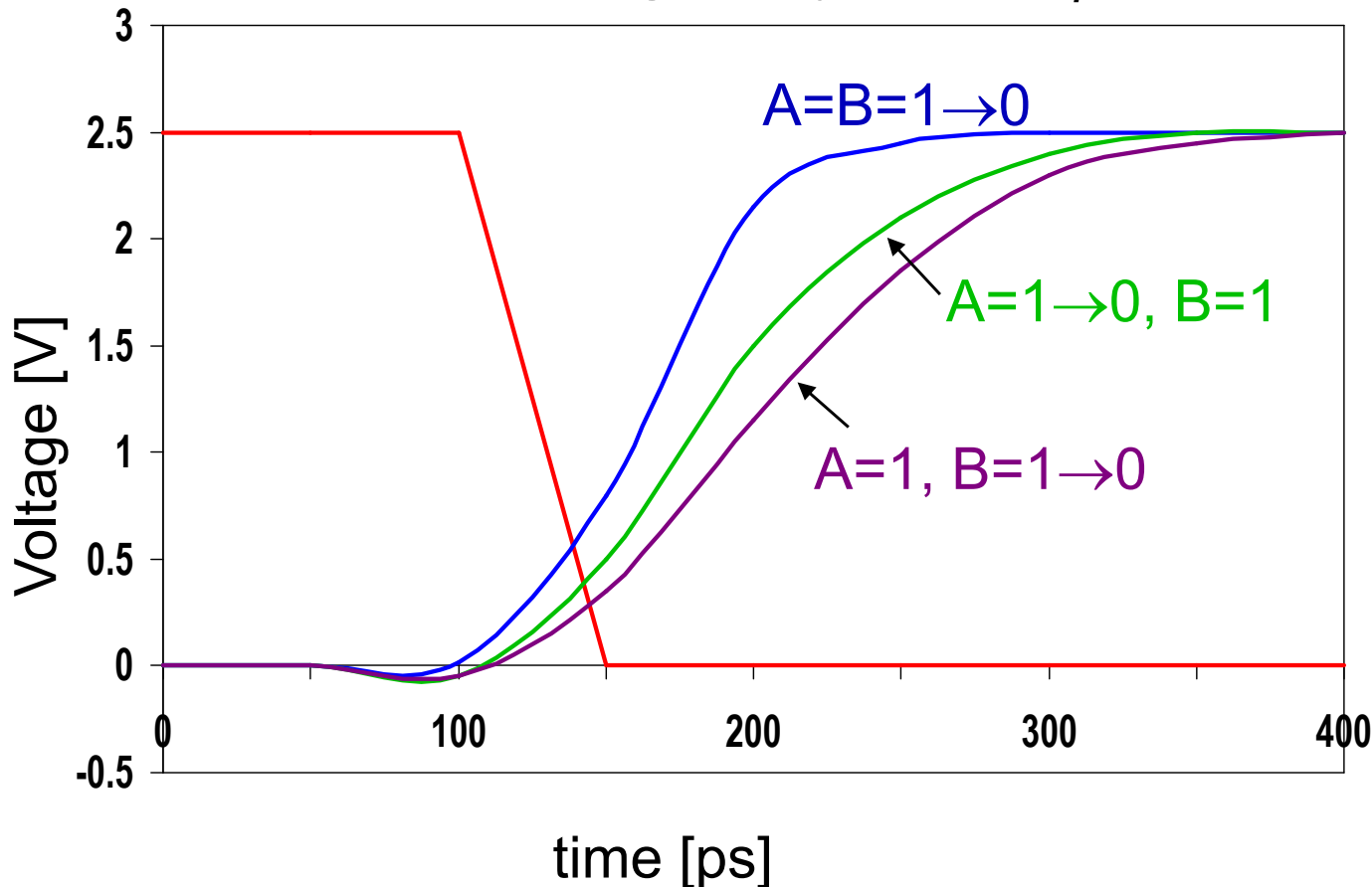
# *Delay Dependence on Input Patterns*



**2-input NAND**

- ❑ **High to Low Delay** depends on the initial state of the internal nodes
- ❑ Example: when both inputs transition from 0 to 1, it is important to know the state of the internal node (between M1 and M2)
- ❑ **Worst case**: when internal node is initially charged up to $V_{DD} - V_{tn}$. This can be ensured by pulsing the A input from 1-(0-1), while B only makes the 0-1 transition.

*Bottom Line: Delay estimation can be a fairly complex affair and requires careful consideration of the internal node caps. and data patterns*

Kaustav Banerjee

# Delay Dependence on Input Patterns

Simulated **Low to High** delay for a 2-input NAND



| Input Data Pattern | Delay (psec) |
|---|---|
| A=B=0→1 | 69 |
| A=1, B=0→1 | 62 |
| A= 0→1, B=1 | 50 |
| A=B=1→0 | 35 |
| A=1, B=1→0 | 76 |
| A= 1→0, B=1 | 57 |

*Note: worst case L-H delay depends on which input (A or B) is driven low (due to internal node cap of PDN stack: source of M2)*

NMOS = 0.5μm/0.25 μm
PMOS = 0.75μm/0.25 μm
$C_L$ = 100 fF

Kaustav Banerjee

# *Revisit Transistor Sizing....*

❏ Sizing for switching threshold
  ▪ All inputs switch together

❏ Sizing for delay
  ▪ Find worst-case input combination

❏ Find equivalent inverter, use inverter analysis to set device sizes

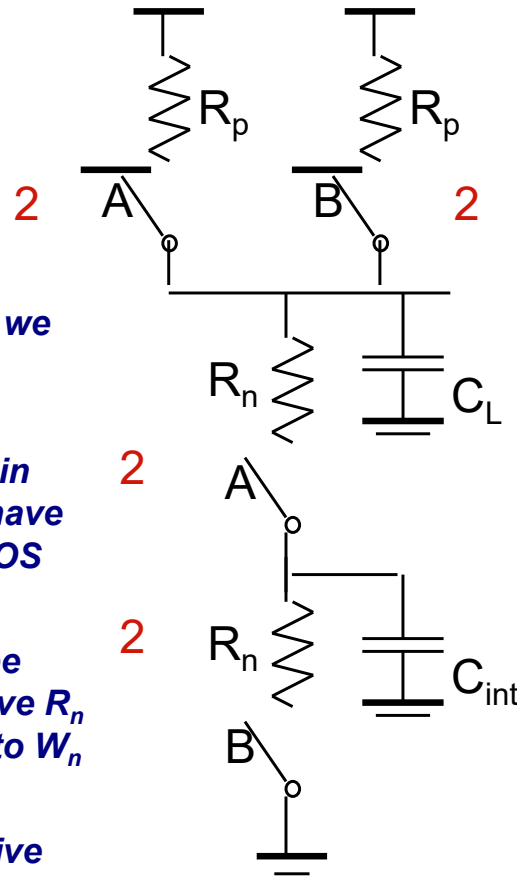Kaustav Banerjee

# *Recall Transistor Sizing….*

**What sizing will lead to a 2:1 equivalent inverter?**

*For effective $R_p = R_n$, we need $W_p = 2W_n$*

*Since two NMOS are in series, each should have $R_n/2$, hence each NMOS size, $W_n = 2$*

*Each PMOS should be such that $R_p$ = effective $R_n$ (which corresponds to $W_n = 1$)*
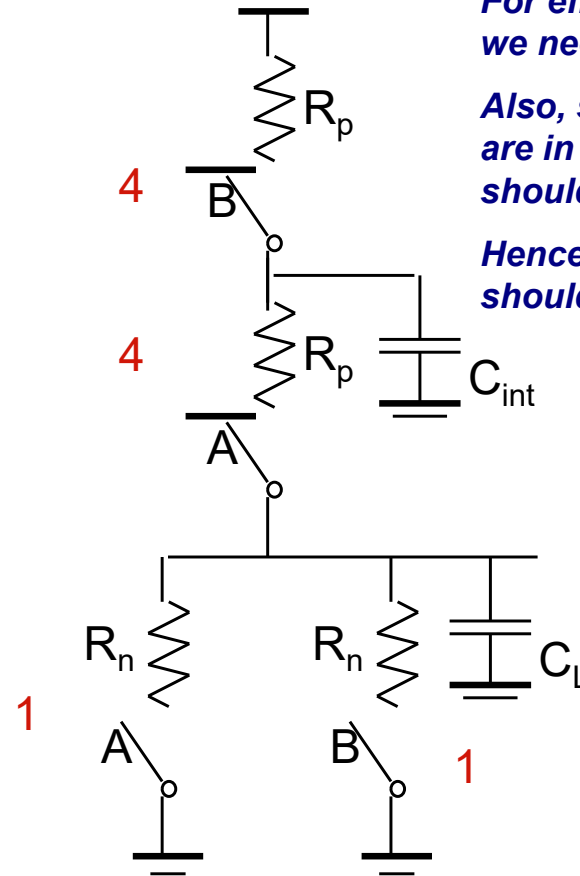
*Hence, $W_p = 2$ (effective $W_n$) = 2*

*For effective $R_p = R_n$, we need $W_p = 2W_n$*

*Also, since two PMOS are in series, each should have $R_p/2$*

*Hence, $W_p$ of each should be $4W_n$*

*Avoid stacking PMOS transistors in series….*



*2-input NAND*

*2-input NOR*

Kaustav Banerjee

# *Example: complex gate (1)*

Design CMOS gate for this truth table:

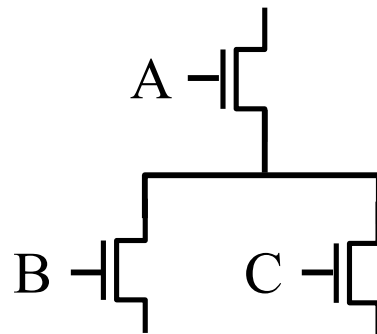| A | B | C | F |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

$$F = \overline{A \cdot (B+C)}$$

Kaustav Banerjee

# *Example: complex CMOS gate (1)*

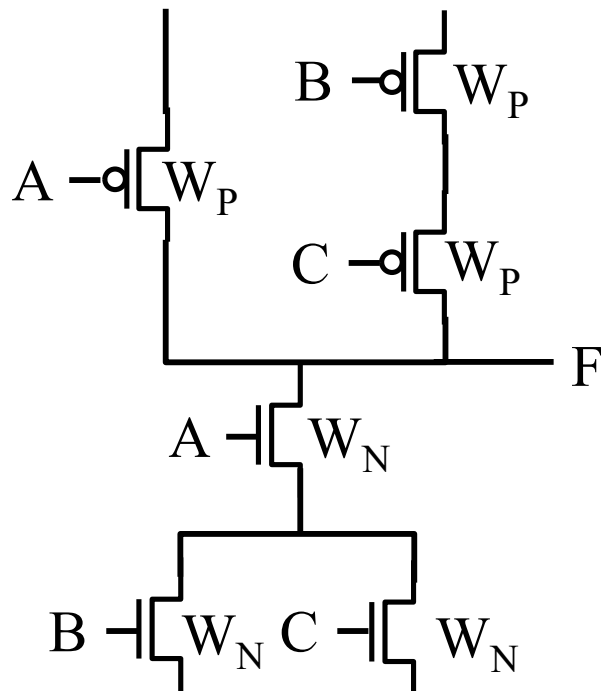Design CMOS gate for this logic function:

$$F = \overline{A \cdot (B+C)}$$

1. Find NMOS pulldown network diagram:

$$G = \overline{F} = A \cdot (B+C)$$

Kaustav Banerjee

# *Example: complex CMOS gate (1)*
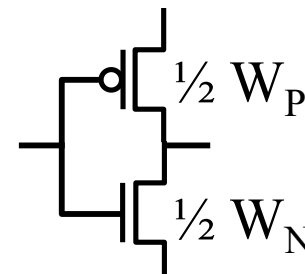
## Completed gate:
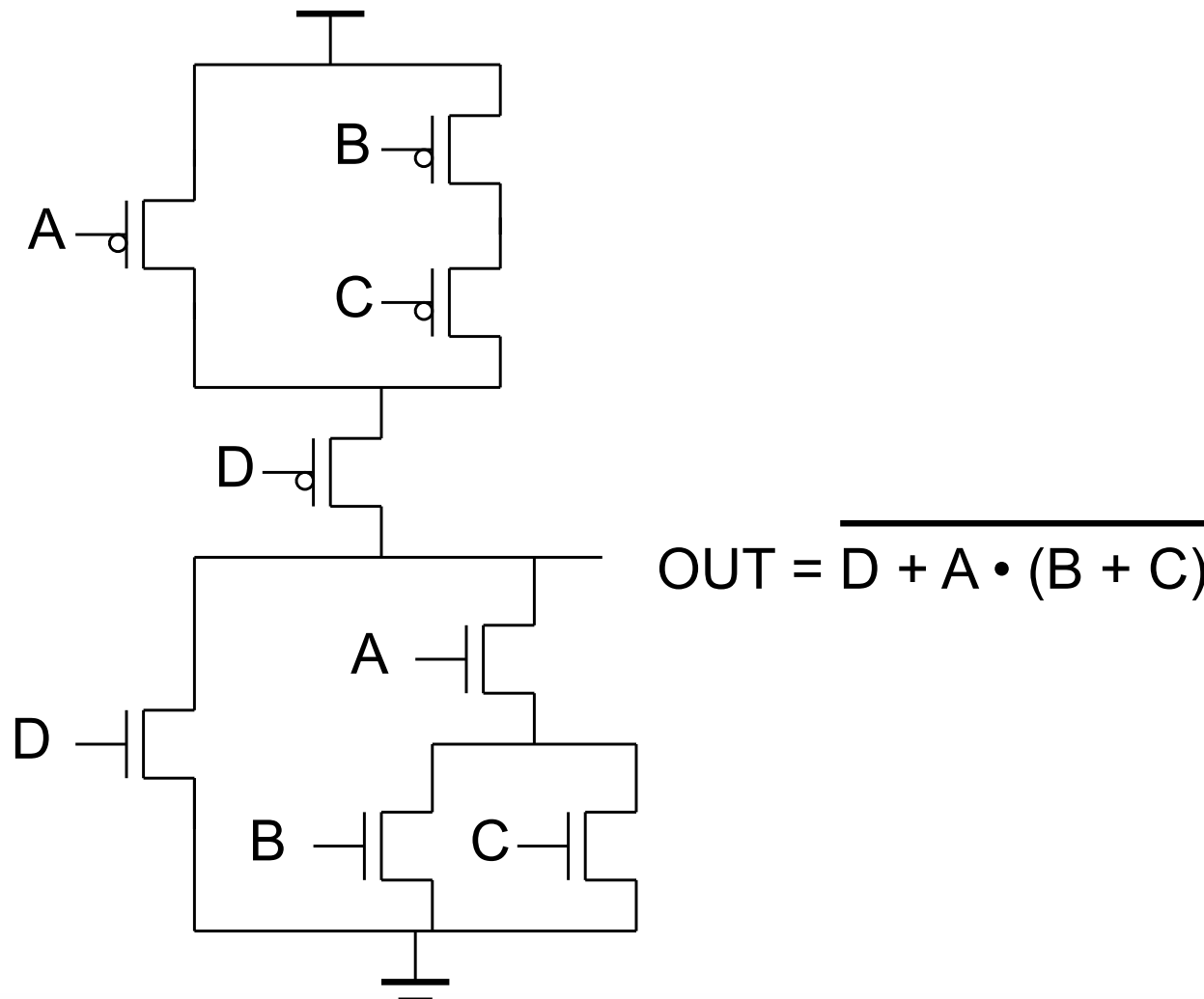


- What is worse-case pullup delay?
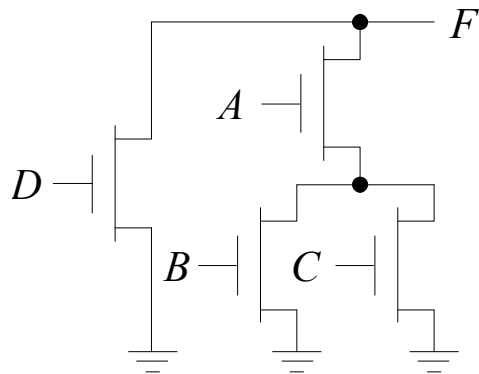
  BC

- What is worse-case pulldown delay?
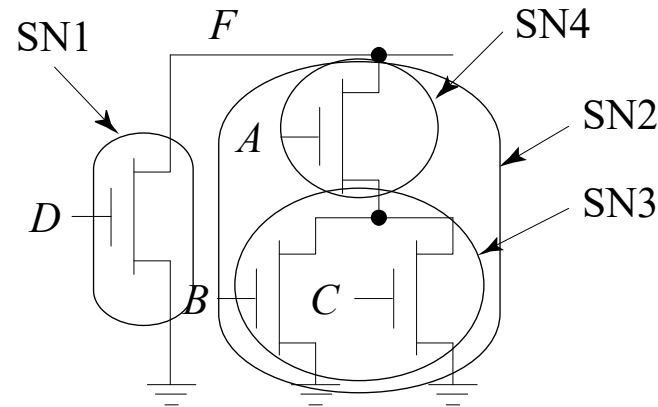
  AB or AC

- Effective inverter for delay calculation:

# *Example: complex CMOS gate (2)*



$$OUT = \overline{D + A \cdot (B + C)}$$
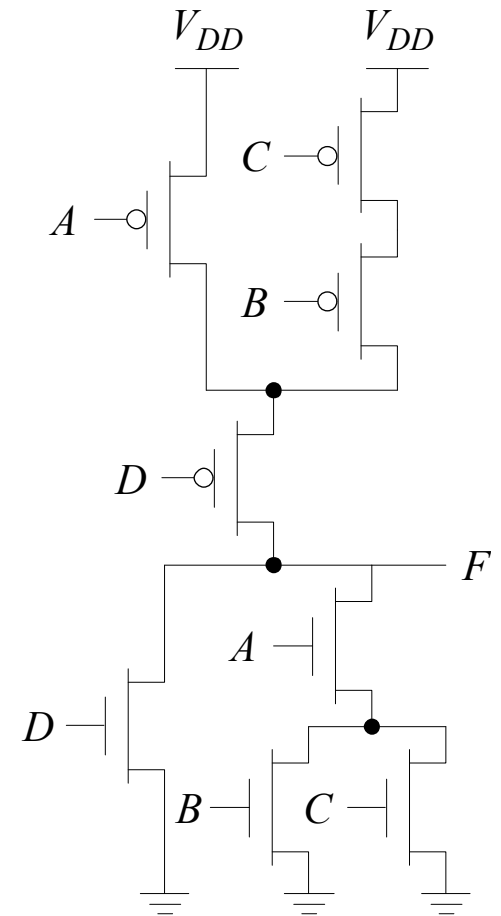
Kaustav Banerjee

# *Example: complex CMOS gate (2)*



(a) pull-down network

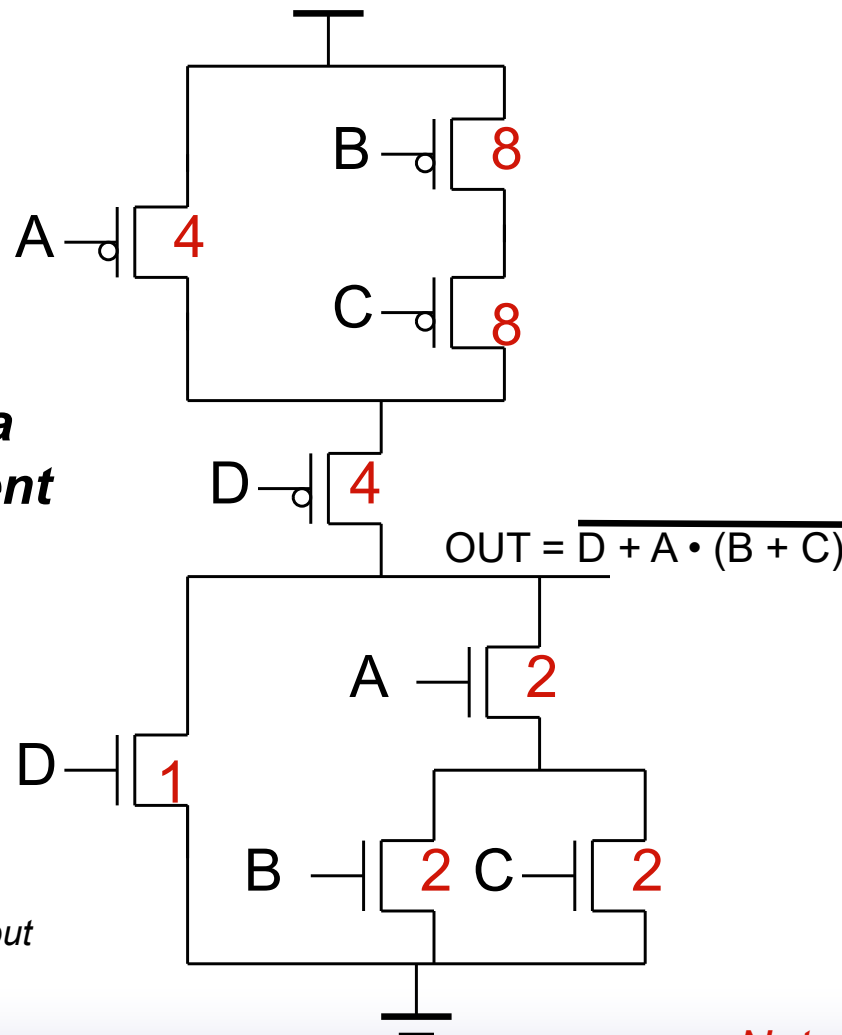(b) Deriving the pull-up network hierarchically by identifying sub-nets

(c) complete gate

*Find the equivalent inverter….*

# Transistor Sizing a Complex CMOS Gate

**What sizing will lead to a 2:1 equivalent inverter?**

OUT = $\overline{D + A \cdot (B + C)}$

*Note: A= 3 and B=C=D=6 will also give a 2:1 inverter but that will give higher output parasitic capacitance (due to larger D)*

1. Start with a transistor in the PDN, that is (preferably) isolated (i.e., it can pull down the output node on its own)----D in this case

2. Assign a minimum size to this transistor (W=1 for D)

3. Now identify other paths in the PDN that can also discharge the output node: AB and AC in this case.

4. Size A, B, and C such that: each path will be electrically equivalent to the path through D (i.e., with same resistance), hence A, B and C should have W=2

5. Repeat the same for the PUN, keeping in mind that i) effective resistance of any path must equal the effective resistance of any path in the PDN, and ii) PMOS transistors will have to be sized (twice as large in this case) appropriately.

*Note:  here we ignored internal caps.*

Kaustav Banerjee

# *CMOS gate design: summary*

❑ Designing a CMOS gate:

- Find pulldown NMOS network from logic function or by inspection

- Find pullup PMOS network
  - By inspection
  - Using logic function
  - Using dual network approach

- Size transistors using equivalent inverter
  - Find worst-case pullup and pulldown paths
  - Size to meet rise/fall or threshold requirements

Kaustav Banerjee
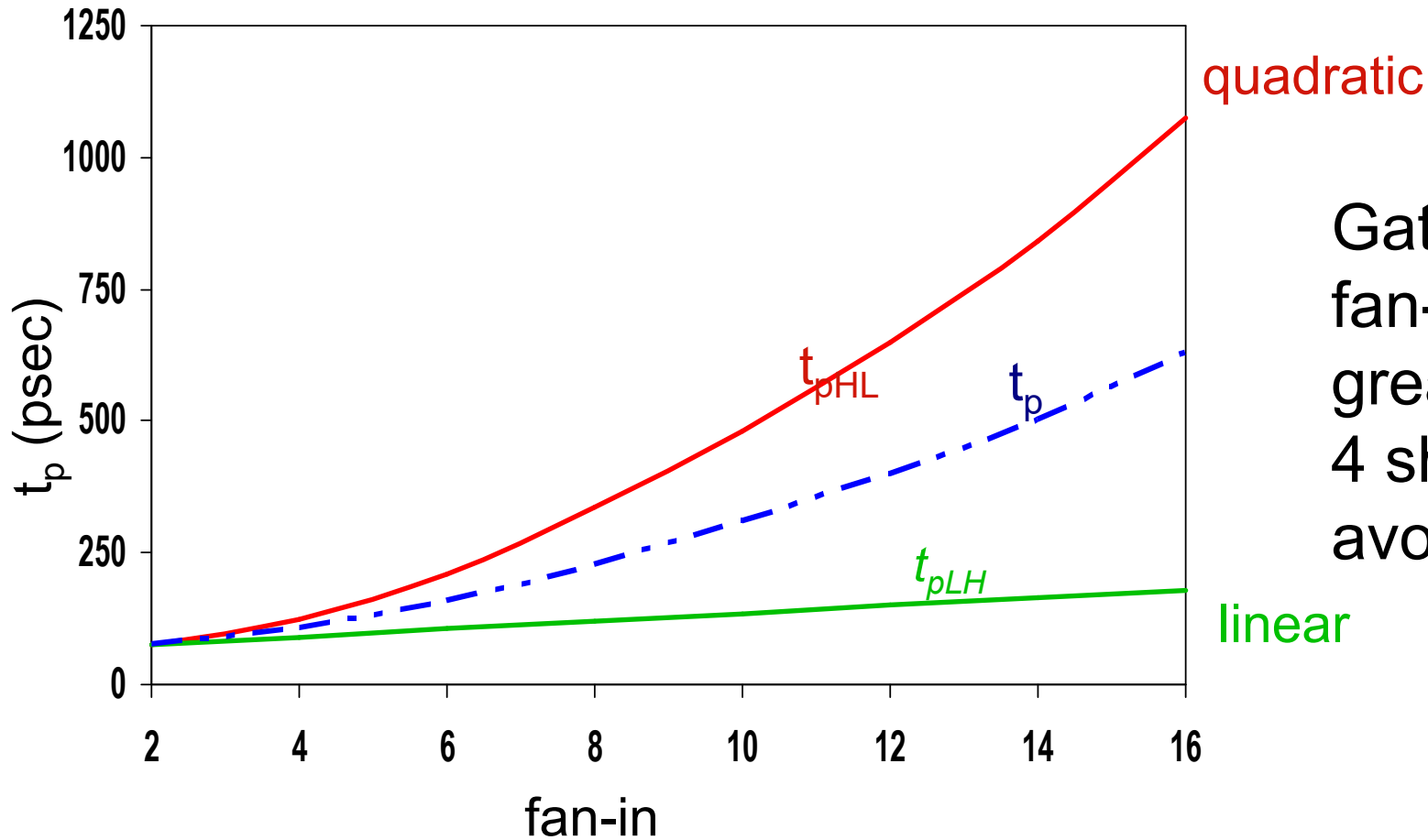
# *Fan-In Considerations*
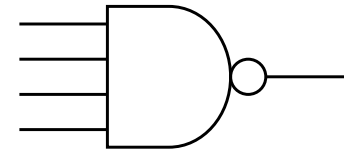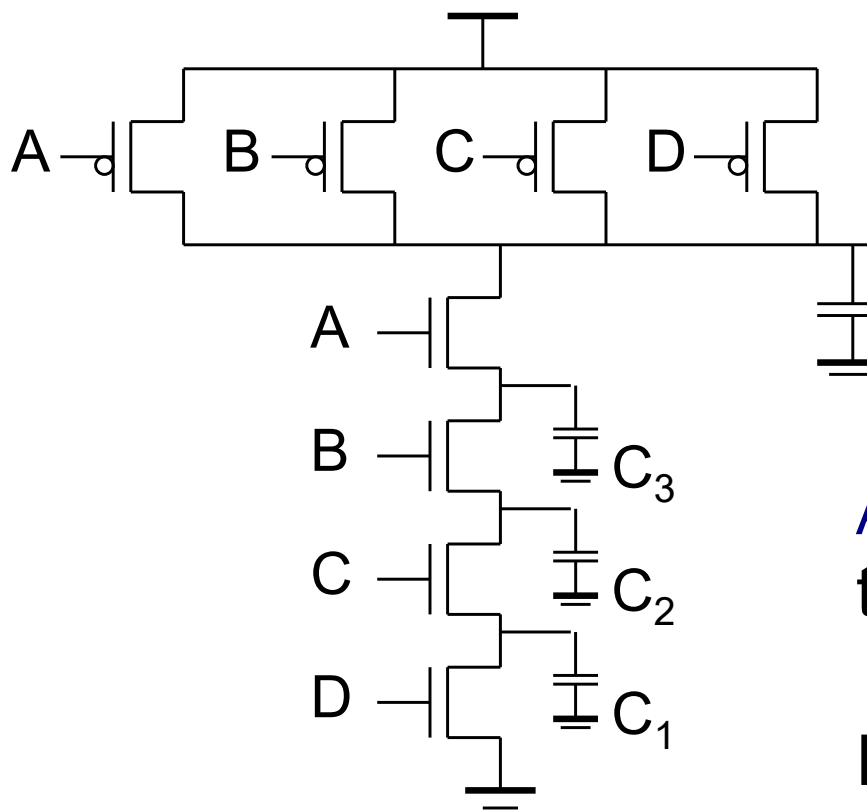


Distributed RC model
(Elmore delay)

Assuming all NMOS are equal size

$$t_{pHL} = 0.69 \, R_{eqn}(C_1 + 2C_2 + 3C_3 + 4C_L)$$

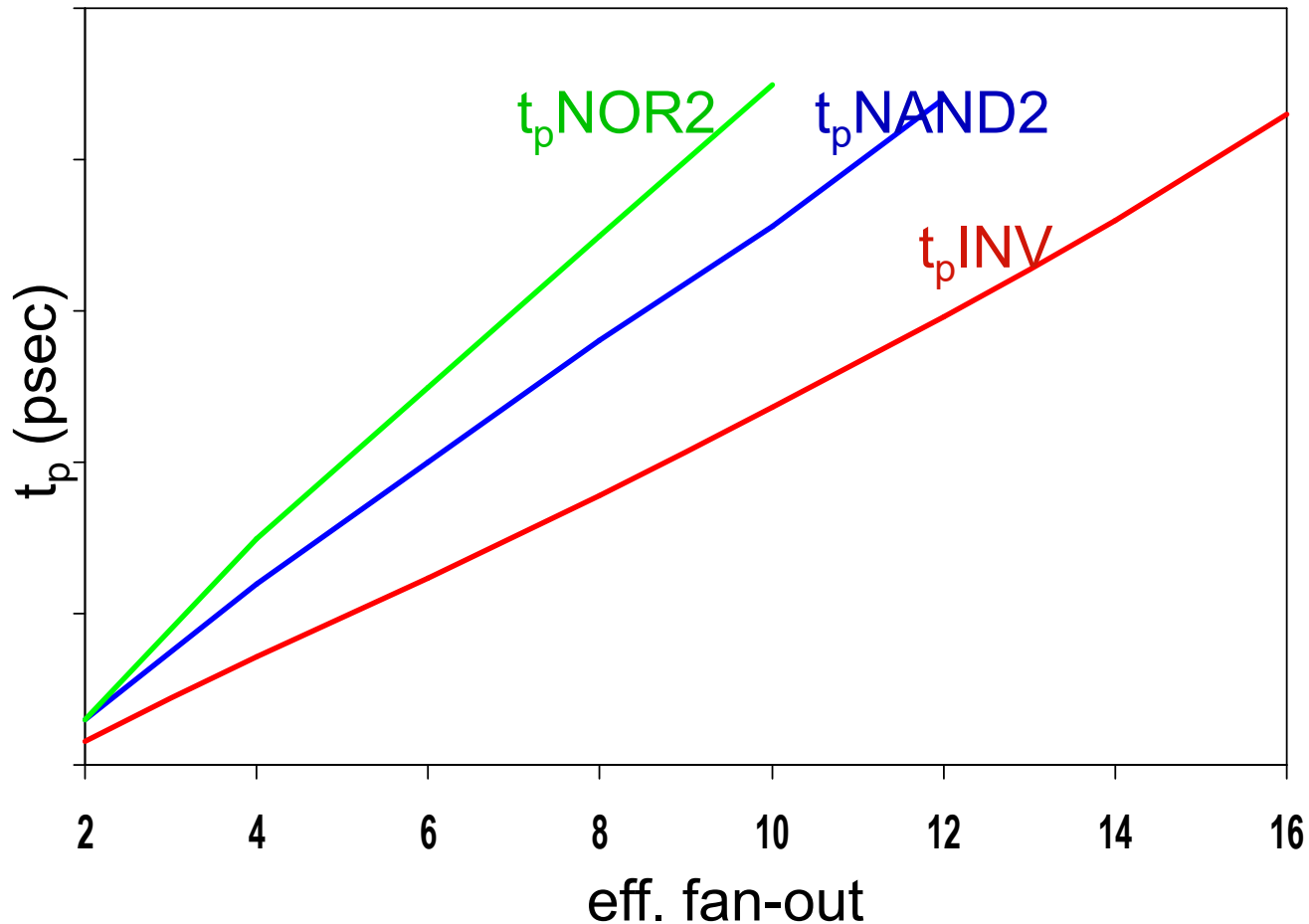Propagation delay deteriorates rapidly as a function of fan-in – quadratically in the worst case.

Kaustav Banerjee

# $t_p$ as a Function of Fan-In



Gates with a fan-in greater than 4 should be avoided.

*$t_{pLH}$ increases linearly with fan-in…..*

# *t$_p$ as a Function of Fan-Out*



All gates have the same drive current.

Slope is a function of "driving strength"

**Note: i) these lines are not going through the origin, ii) NAND and NOR have same intrinsic delay….**

Kaustav Banerjee

# $t_p$ as a Function of Fan-In and Fan-Out

- Fan-in: quadratic due to increasing resistance and capacitance
- Fan-out: each additional fan-out gate adds two gate capacitances to $C_L$

$$t_p = a_1 FI + a_2 FI^2 + a_3 FO$$

Kaustav Banerjee

# *Common Static CMOS Gate Topologies*
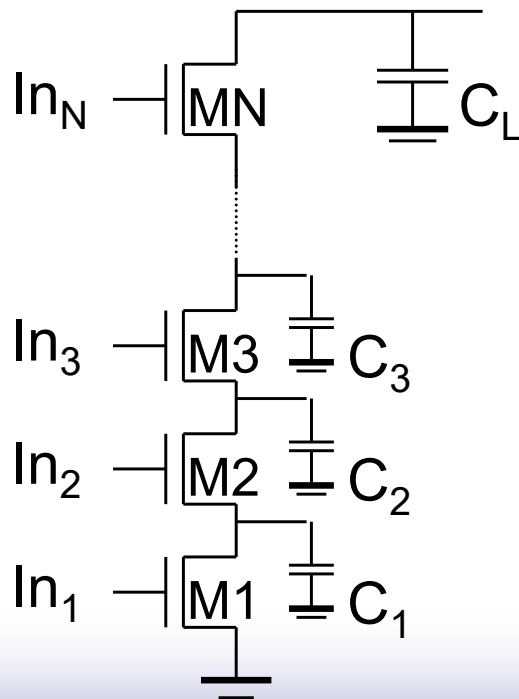
❑ And-Or-Invert (AOI)

- Sum of products Boolean function
- Parallel branches of series connected NMOS

❑ Or-And-Invert (OAI)

- Product of sums Boolean function
- Series connection of sets of parallel NMOS

Kaustav Banerjee

# Fast Complex Gates: Design Technique 1

❑ Transistor sizing

▪ as long as fan-out capacitance dominates

❑ Progressive sizing (not so simple to layout!)
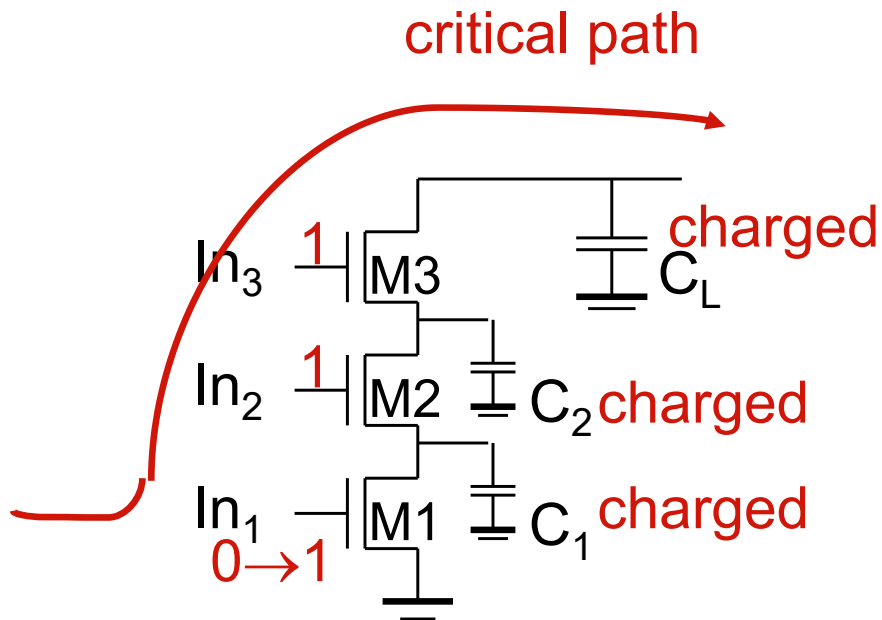
Distributed RC line

$M1 > M2 > M3 > \ldots > MN$
(the transistor closest to the output is the smallest: has biggest R)

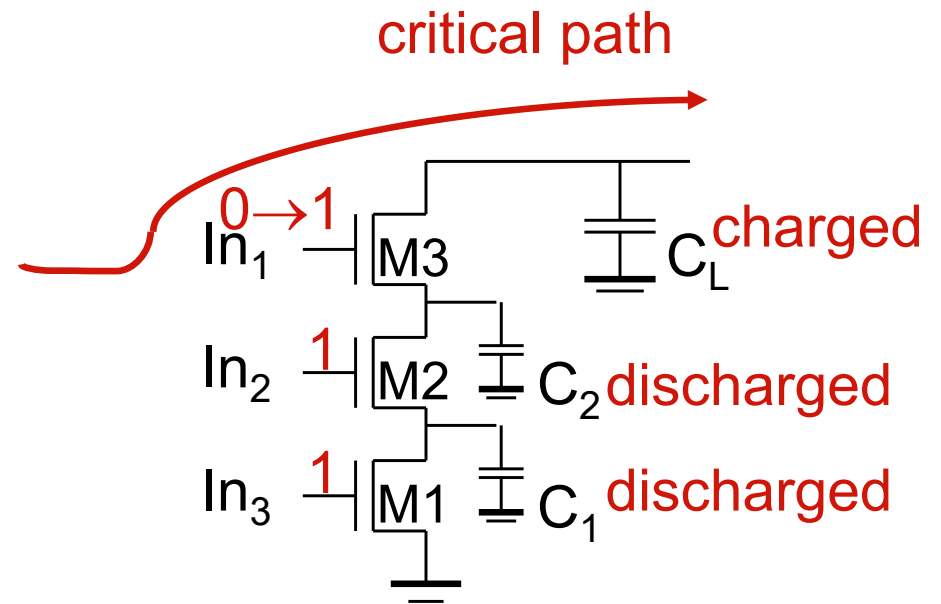Can reduce delay by more than 20%; decreasing gains as technology shrinks

Kaustav Banerjee

# *Fast Complex Gates: Design Technique 2*

❑ Transistor ordering

*Critical path is determined by the slowest arriving signal: In1*

critical path

$In_3$  1  M3          $C_L$ charged

$In_2$  1  M2  $C_2$ charged

$In_1$  M1  $C_1$ charged
$0 \rightarrow 1$

delay determined by time to discharge $C_L$, $C_1$ and $C_2$

critical path

$In_1$ $0 \rightarrow 1$  M3          $C_L$ charged

$In_2$  1  M2  $C_2$ discharged

$In_3$  1  M1  $C_1$ discharged

delay determined by time to discharge $C_L$

Kaustav Banerjee

# Fast Complex Gates: Design Technique 3
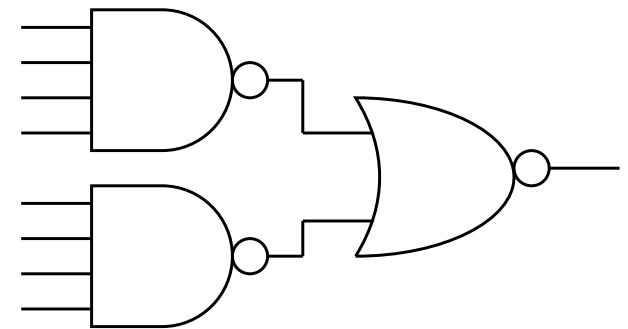
❑ Alternative logic structures

F = ABCDEFGH

*8-input AND*
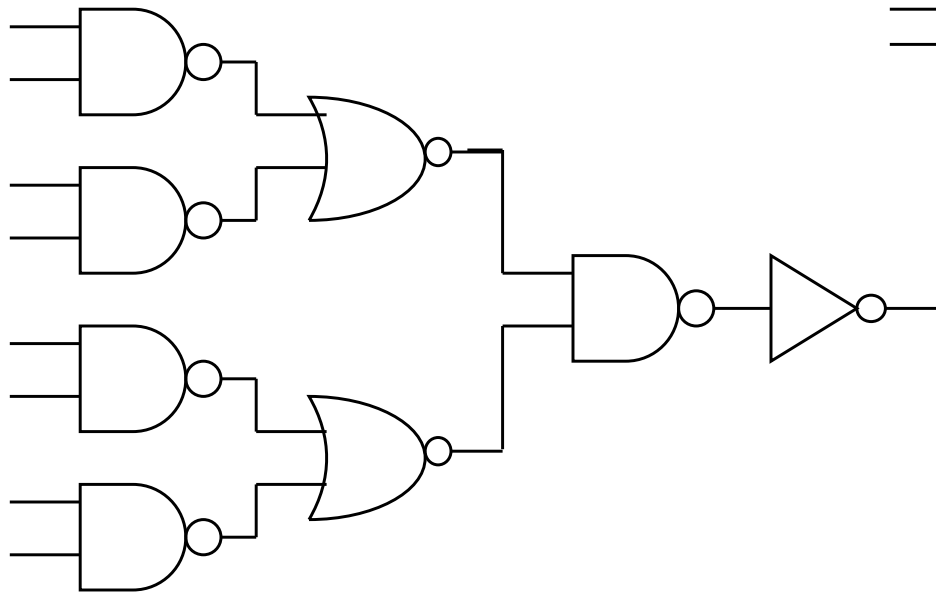
*sloooowww….*

*Using De Morgan's Law…*
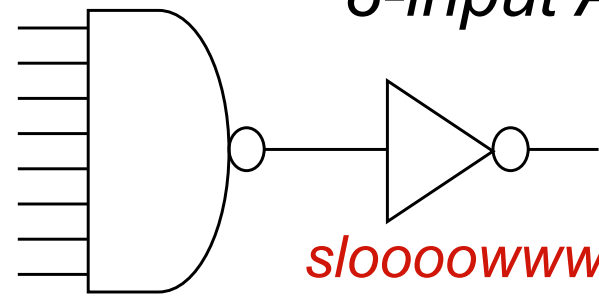
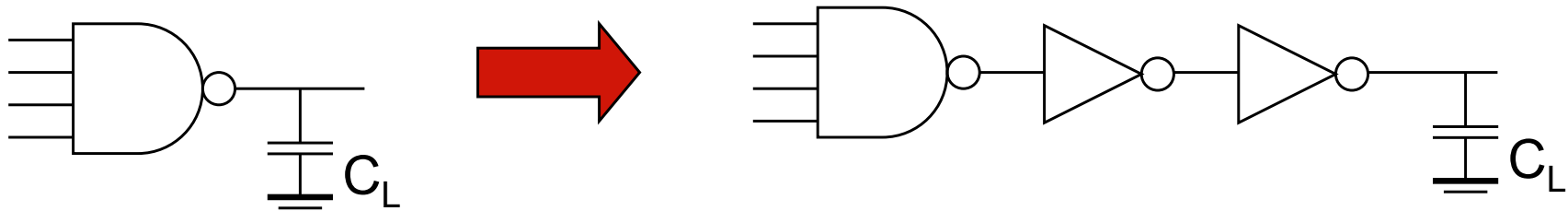# *Fast Complex Gates: Design Technique 4*

❑ Isolating fan-in from fan-out using buffer insertion

# Fast Complex Gates: Design Technique 5

❏ Reducing the voltage swing

$$t_{pHL} = 0.69 \, (3/4 \, (C_L \, V_{DD})/ \, I_{DSATn} \, )$$

$$= 0.69 \, (3/4 \, (C_L \, V_{swing})/ \, I_{DSATn} \, )$$

- linear reduction in delay
- also reduces power consumption

❏ But the following gate is much slower!

❏ Or requires use of "sense amplifiers" on the receiving end to restore the signal level (memory design)

Kaustav Banerjee

# *Revise CMOS gate layout....*

❑ Goal: minimum area

❑ Method

- Minimize diffusion breaks (reduces capacitance on internal nodes)
- Align transistors with common gates above each other in layout (minimizes poly length)
- Group PMOS and NMOS transistors together

❑ Approach:

- Use Euler path method to find ordering of transistors in layout

Kaustav Banerjee