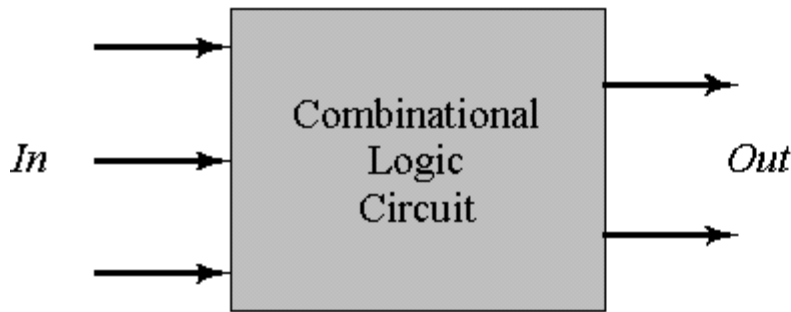


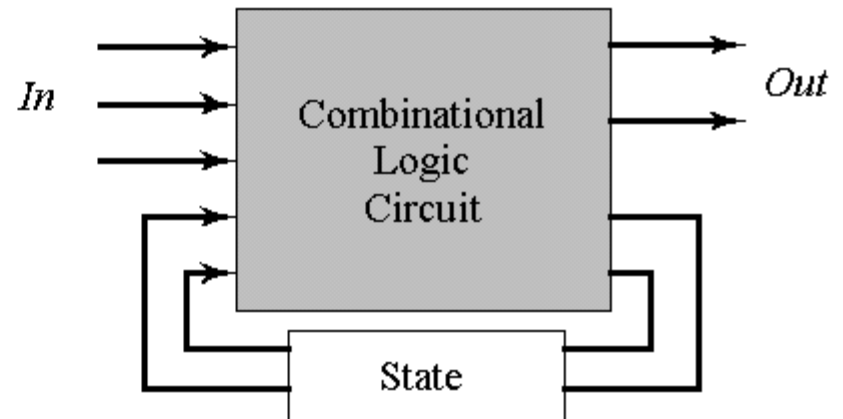
Designing Combinational Logic Circuits

Combinational vs. Sequential Logic



Combinational

$$\text{Output} = f(\text{In})$$



Sequential

$$\text{Output} = f(\text{In}, \text{Previous In})$$

Static CMOS Circuit

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).

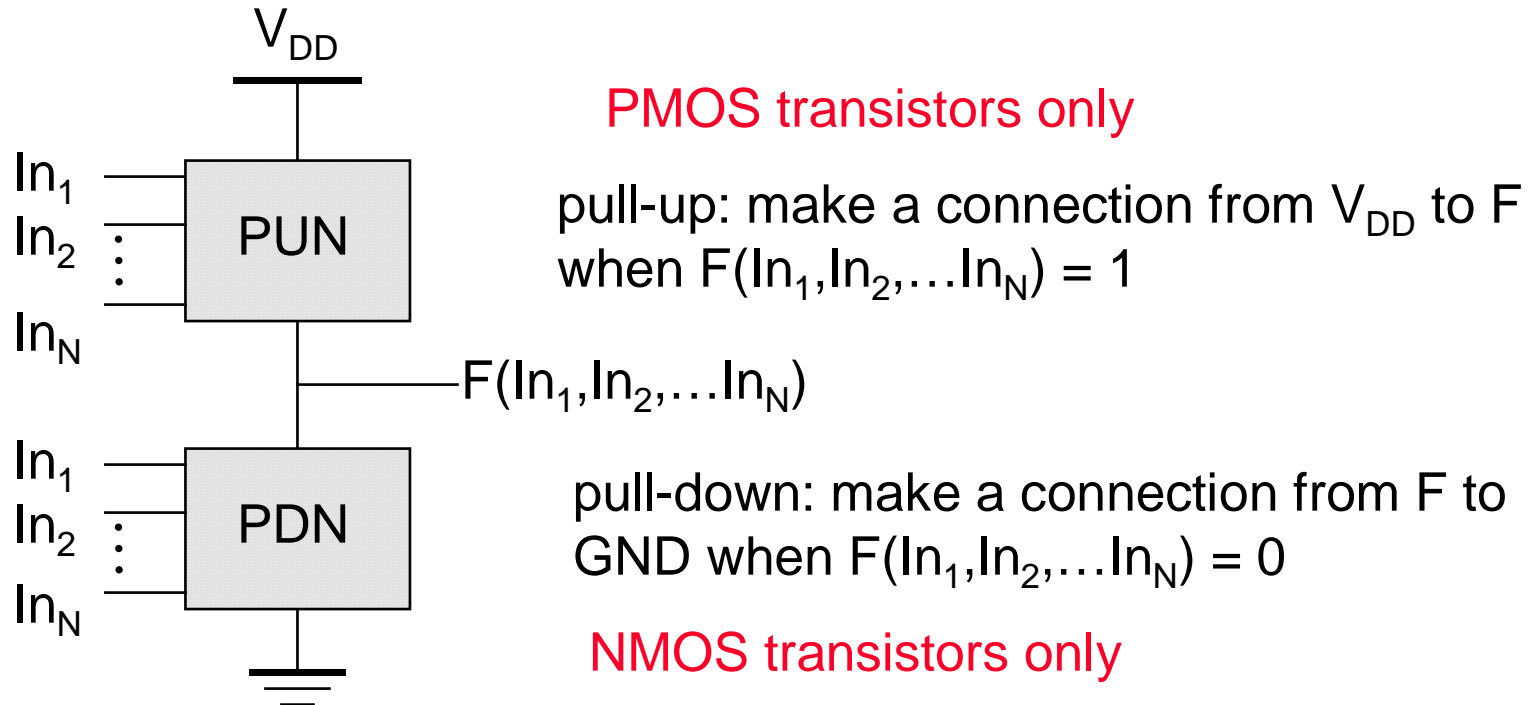
This is in contrast to the *dynamic* circuit class, which relies on temporary storage of signal values on the capacitance of high impedance circuit nodes.

CMOS Circuit Styles

- ❑ **Static complementary** CMOS - except during switching, output connected to either V_{DD} or GND via a low-resistance path
 - λ high noise margins
 - full rail to rail swing
 - V_{OH} and V_{OL} are at V_{DD} and GND, respectively
 - λ low output impedance, high input impedance
 - λ no steady state path between V_{DD} and GND (**no** static power consumption)
 - λ delay a function of load capacitance and transistor resistance
 - λ comparable rise and fall times (under the appropriate transistor sizing conditions)
- ❑ **Dynamic** CMOS - relies on temporary storage of signal values on the capacitance of high-impedance circuit nodes
 - λ simpler, faster gates
 - λ increased sensitivity to noise

Static Complementary CMOS

- Pull-up network (PUN) and pull-down network (PDN)

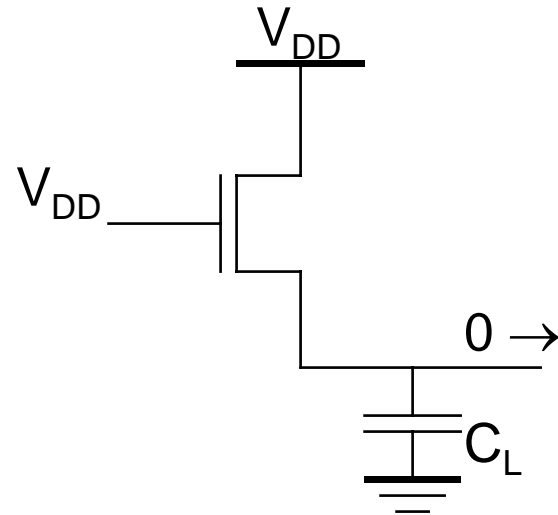
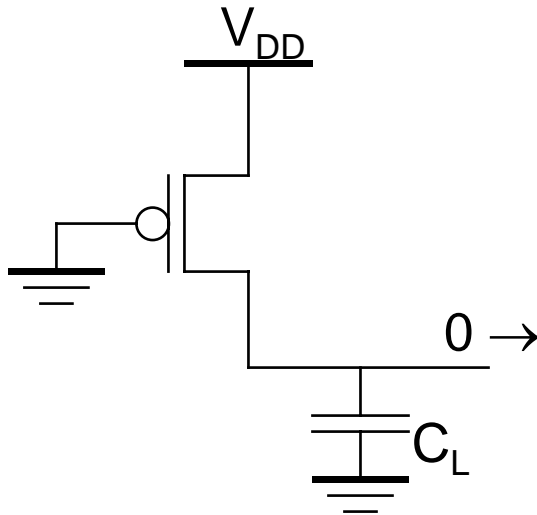


PUN and PDN are **dual** logic networks

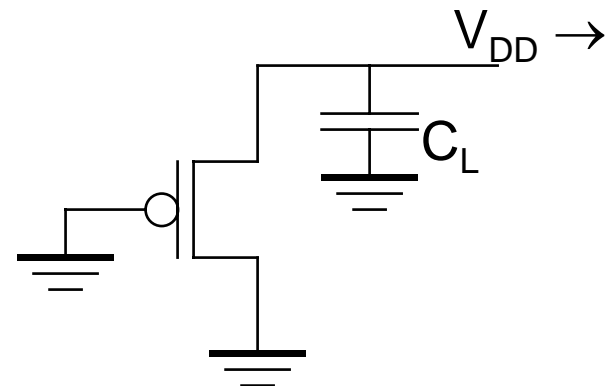
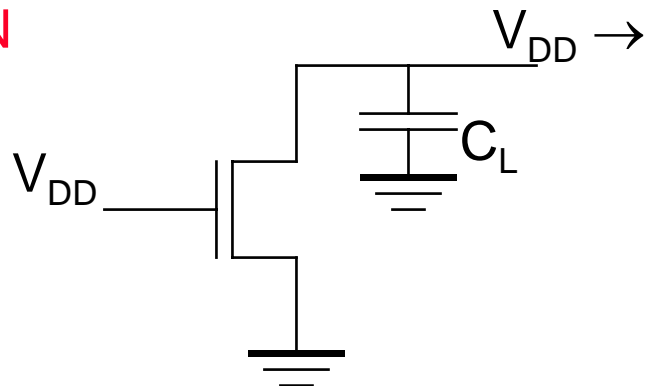
One and only one of the networks is conducting in steady state

Threshold Drops

PUN

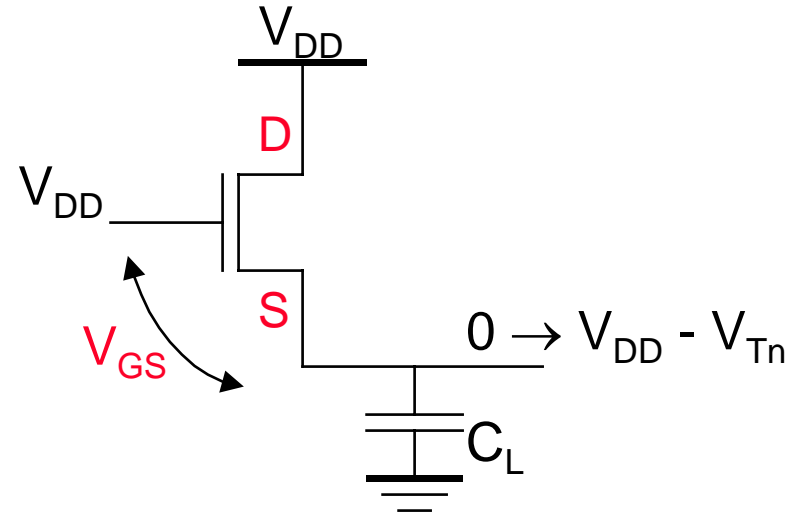
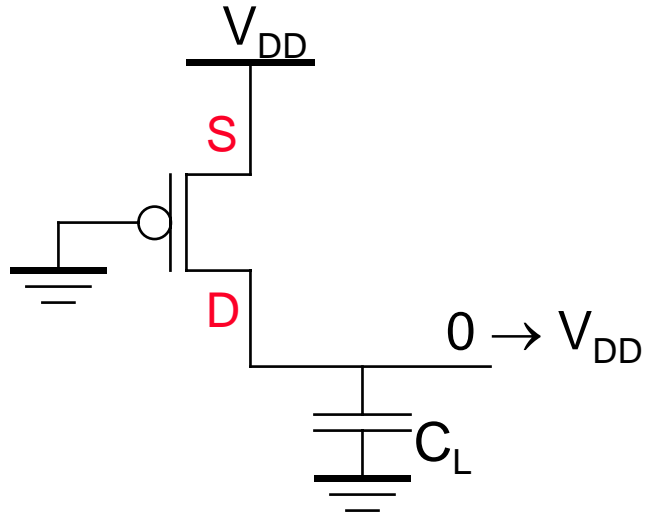


PDN

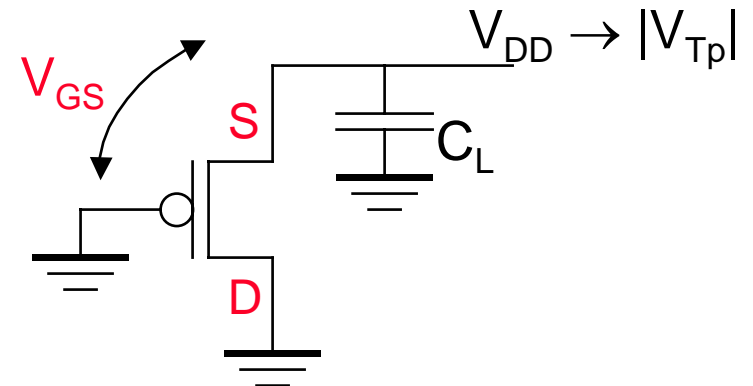
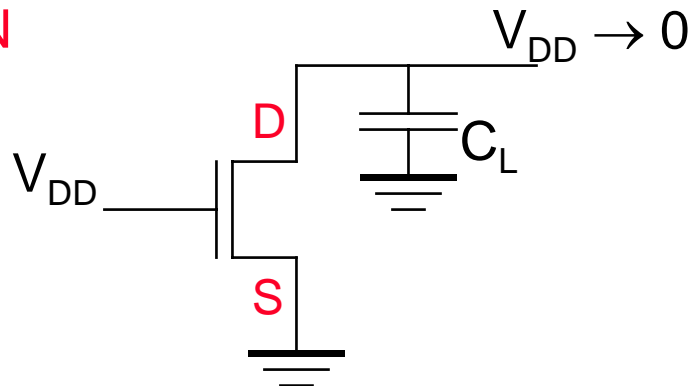


Threshold Drops

PUN



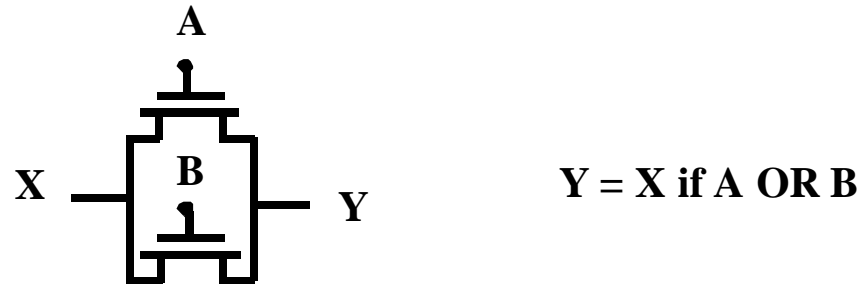
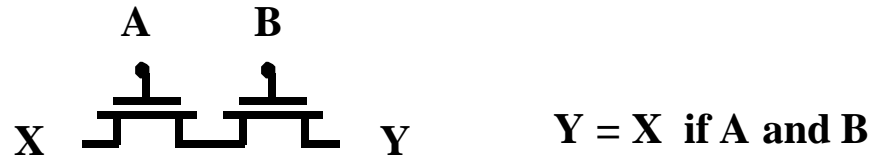
PDN



NMOS Transistors in Series/Parallel Connection

Transistors can be thought as a switch controlled by its gate signal

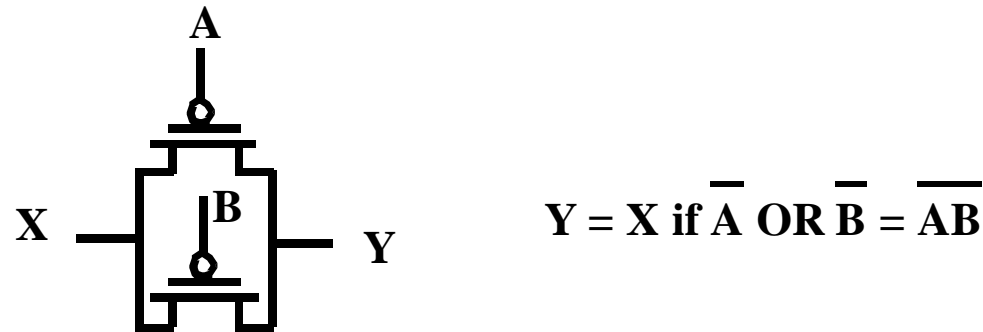
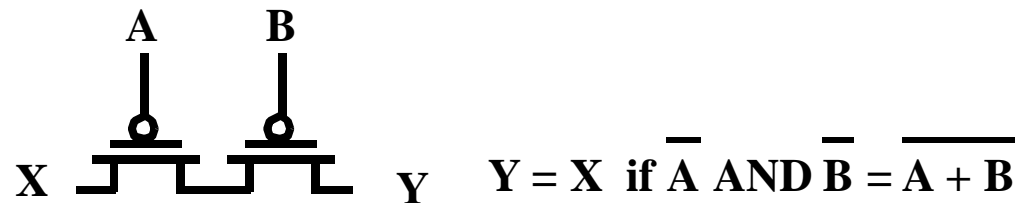
NMOS switch closes when switch control input is high



NMOS Transistors pass a “strong” 0 but a “weak” 1

PMOS Transistors in Series/Parallel Connection

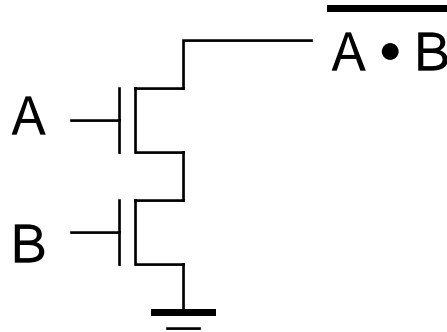
PMOS switch closes when switch control input is low



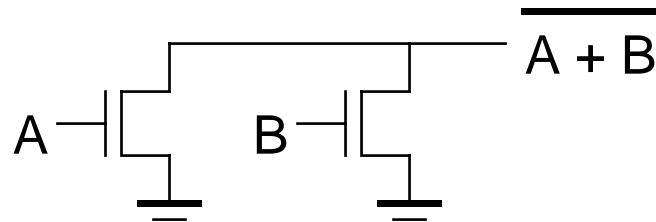
PMOS Transistors pass a “strong” 1 but a “weak” 0

Construction of PDN

- NMOS devices in **series** implement a NAND function



- NMOS devices in **parallel** implement a NOR function



Dual PUN and PDN

- PUN and PDN are dual networks

λ DeMorgan's theorems

$$\overline{A + B} = \overline{A} \cdot \overline{B}$$

$$\overline{A \cdot B} = \overline{A} + \overline{B}$$

λ a **parallel** connection of transistors in the PUN corresponds to a **series** connection of the PDN

- Complementary gate is naturally **inverting** (NAND, NOR, XNOR)
- Number of transistors for an N-input logic gate is **2N**

Complementary CMOS Logic Style

- **PUN is the DUAL of PDN**
(can be shown using DeMorgan's Theorem's)

$$\overline{A + B} = \bar{A}\bar{B}$$

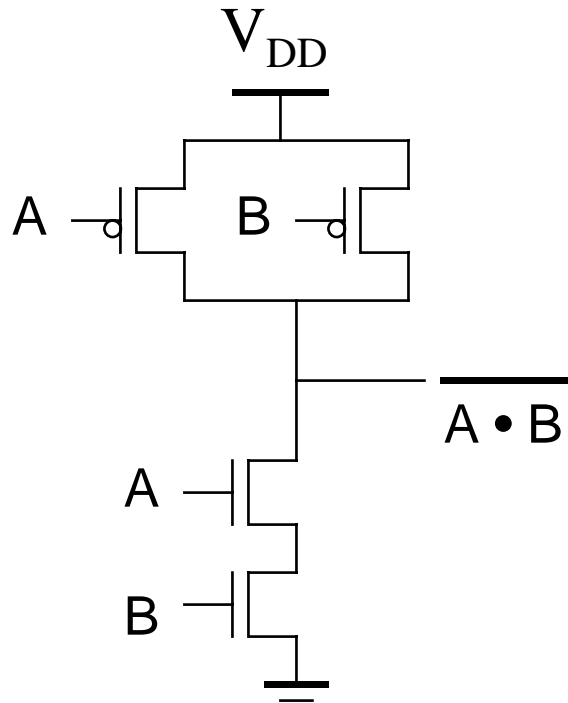
$$\overline{\bar{A}\bar{B}} = A + B$$

- **The complementary gate is inverting**

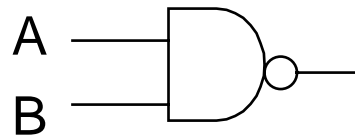


$$\text{AND} = \text{NAND} + \text{INV}$$

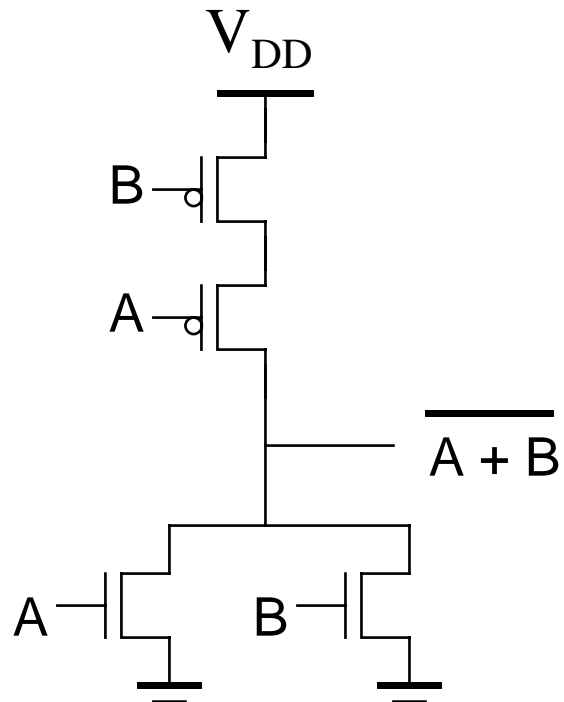
CMOS NAND



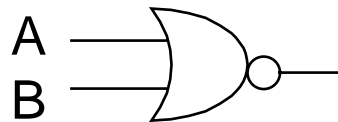
A	B	F
0	0	1
0	1	1
1	0	1
1	1	0



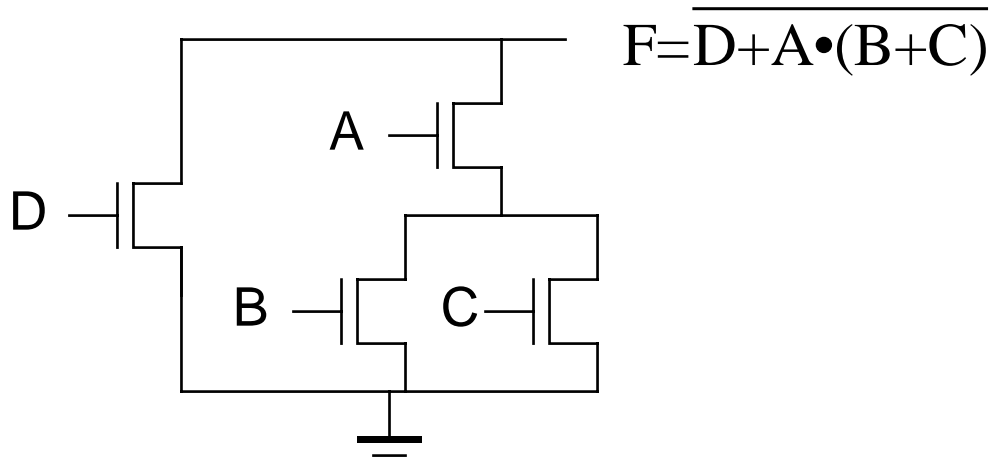
CMOS NOR



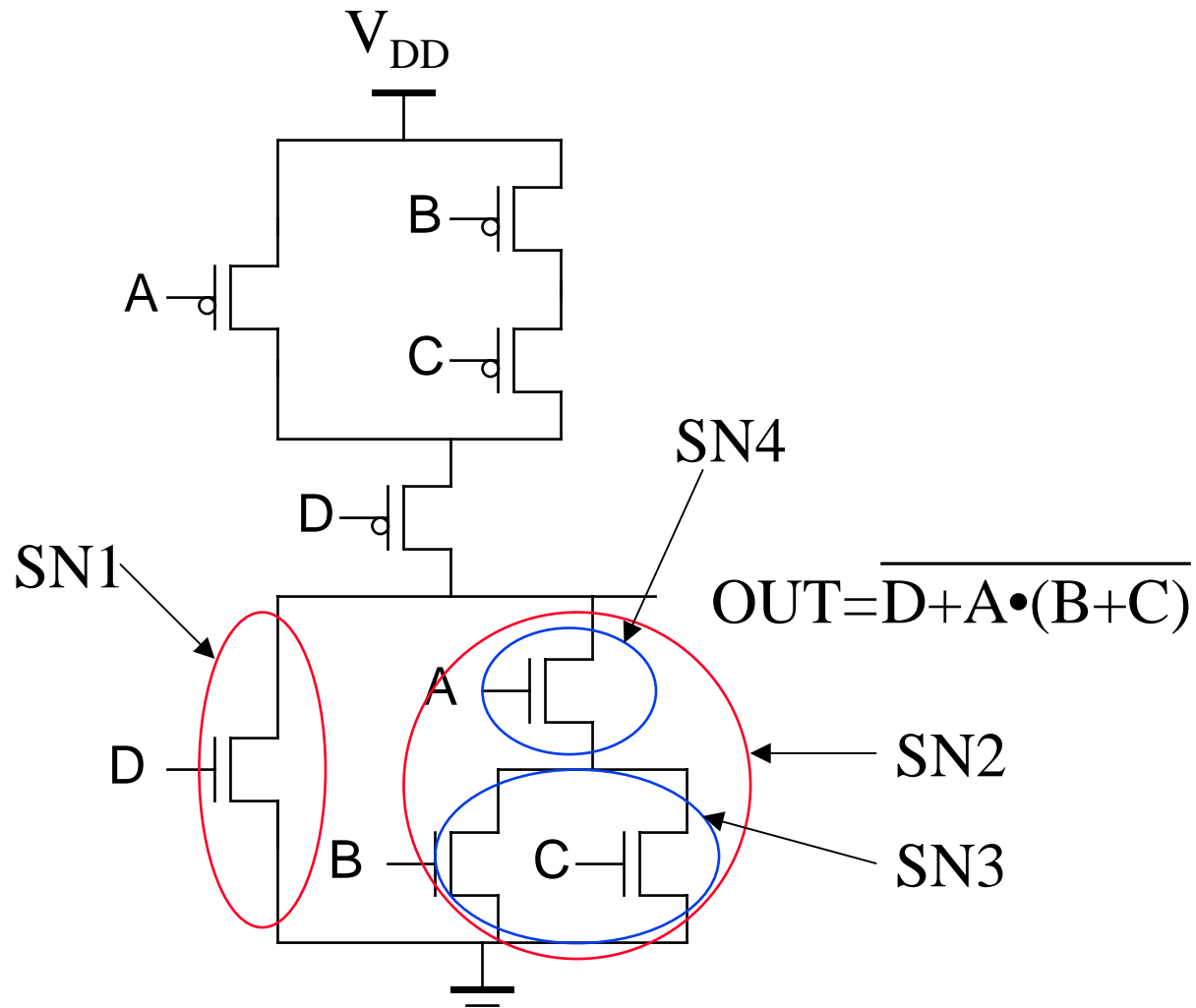
A	B	F
0	0	1
0	1	0
1	0	0
1	1	0



Complex CMOS Gate

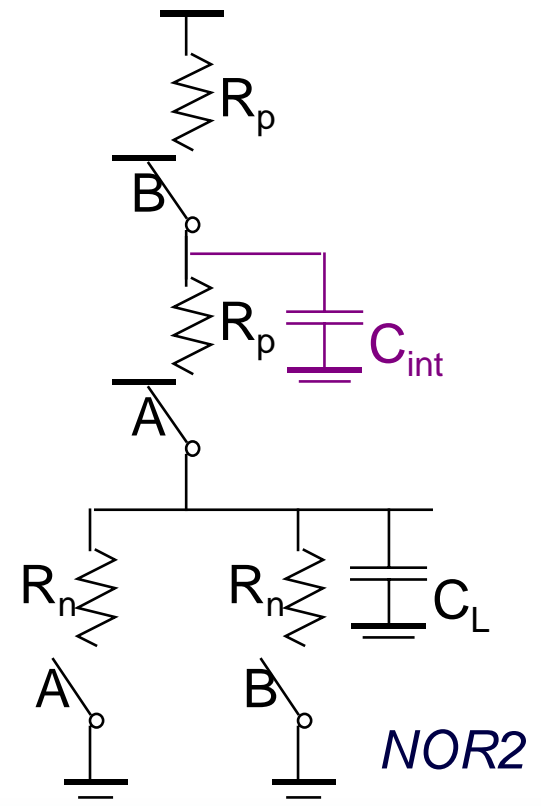
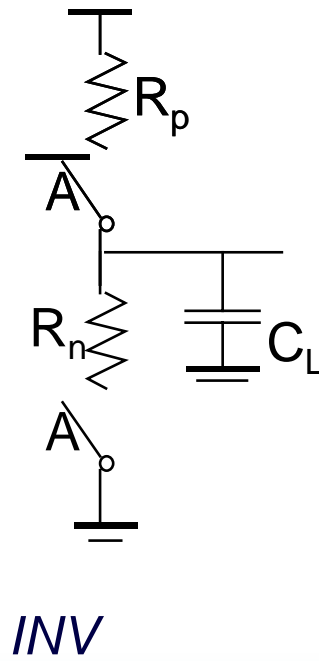
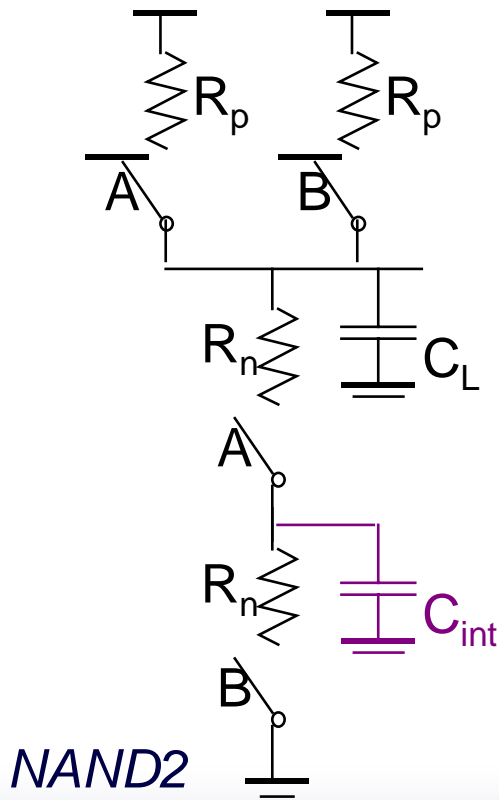
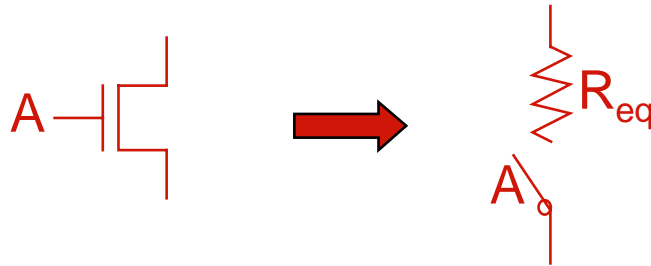


Complex CMOS Gate

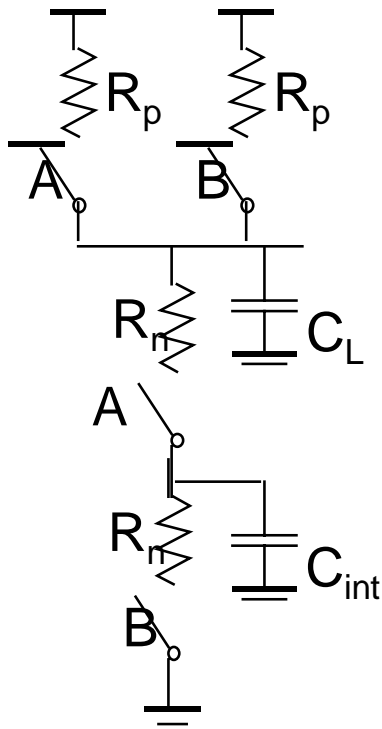


100

Switch Delay Model



Input Pattern Effects on Delay



- Delay is dependent on the **pattern** 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 $0.69 2R_n C_L$

Adding devices in series slows down the circuit, and devices must be made wider to avoid performance penalty.

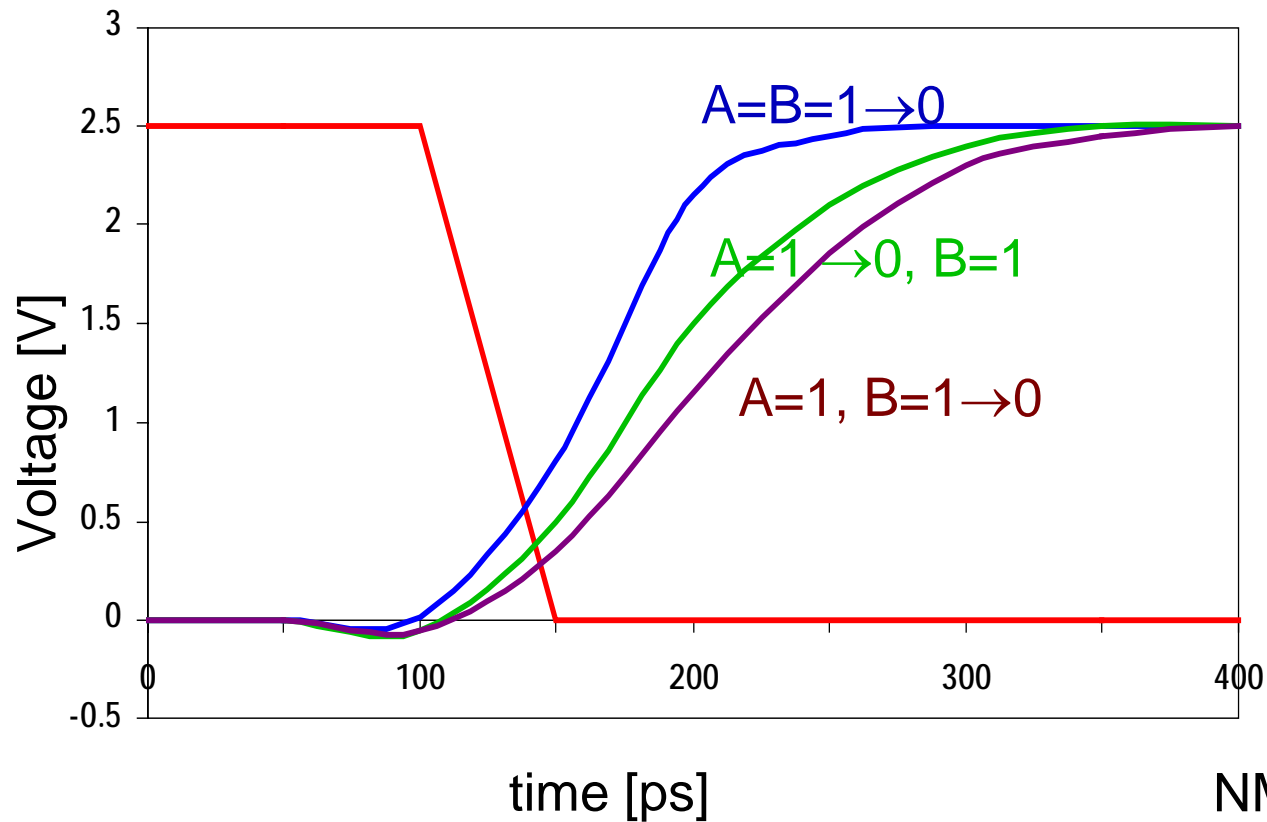
When sizing the transistors in a gate with multiple inputs, we should pick the combination of inputs that triggers the worst case conditions.

For the NAND gate to have the same pull-down delay (t_{phl}) with an inverter, the NMOS devices in the PDN stack must be made twice as wide (2.5 times, if velocity saturation is effective). PMOS devices can remain unchanged.

(Extra capacitance introduced by widening is ignored here, which is not a good assumption)

Delay Dependence on Input Patterns

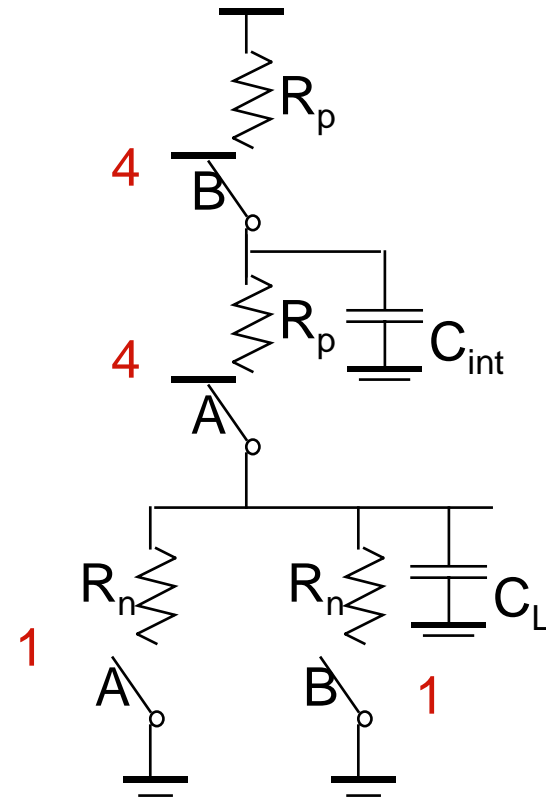
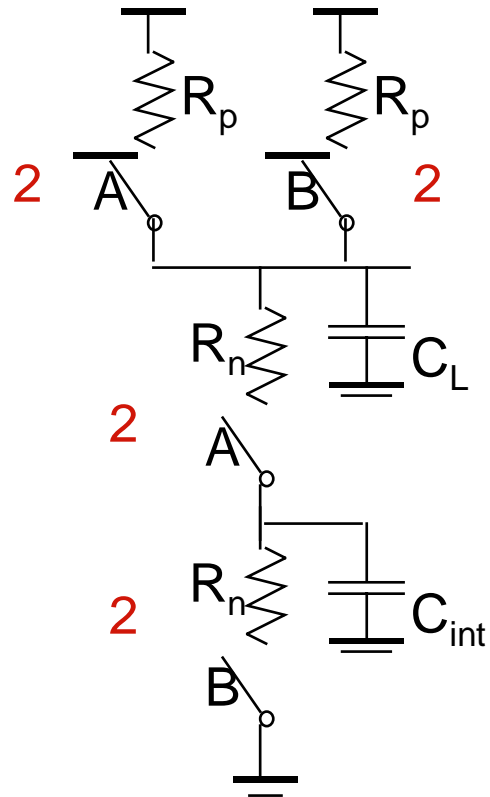
NAND Gate



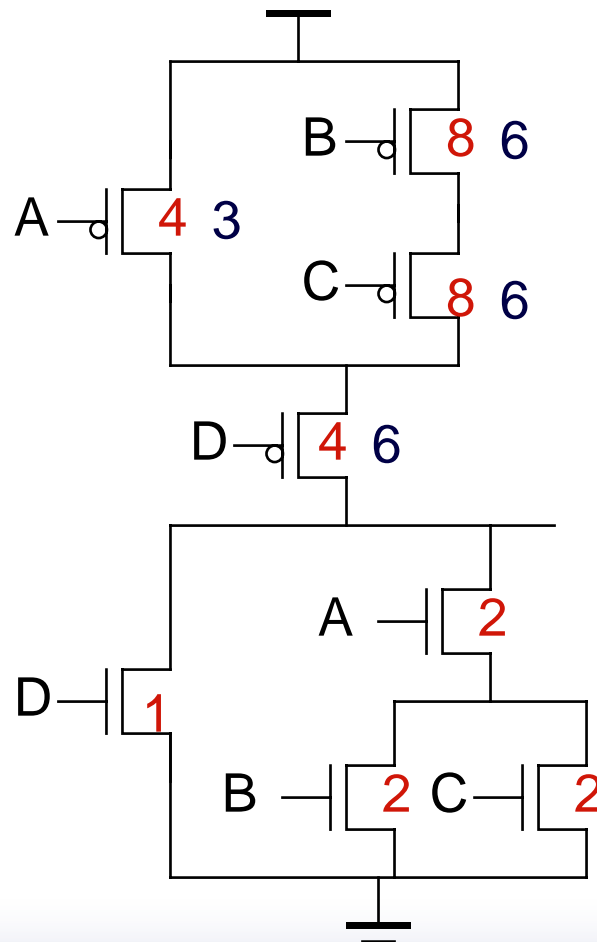
Input Data Pattern	Delay (psec)
$A=B=0 \rightarrow 1$	69
$A=1, B=0 \rightarrow 1$	62
$A=0 \rightarrow 1, B=1$	50
$A=B=1 \rightarrow 0$	35
$A=1, B=1 \rightarrow 0$	76
$A=1 \rightarrow 0, B=1$	57

NMOS = $0.5\mu\text{m}/0.25\mu\text{m}$
PMOS = $0.75\mu\text{m}/0.25\mu\text{m}$
 $C_L = 100\text{ fF}$

Transistor Sizing

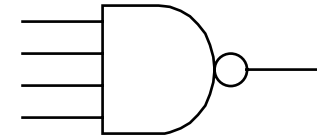
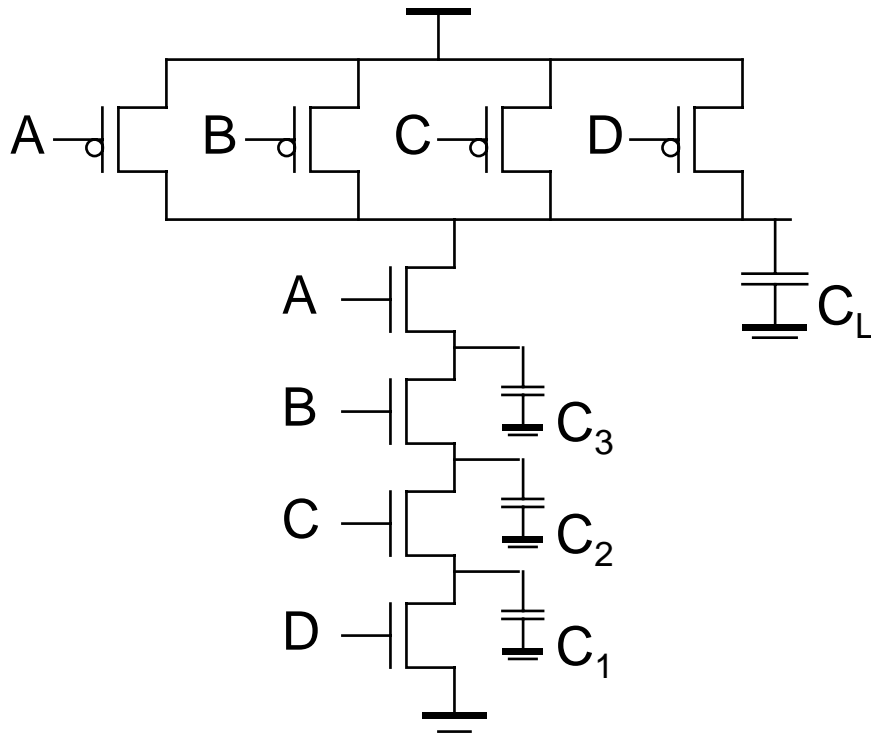


Transistor Sizing a Complex CMOS Gate



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

Fan-In Considerations



Distributed RC model
(Elmore delay), assuming all
NMOS devices of equal size

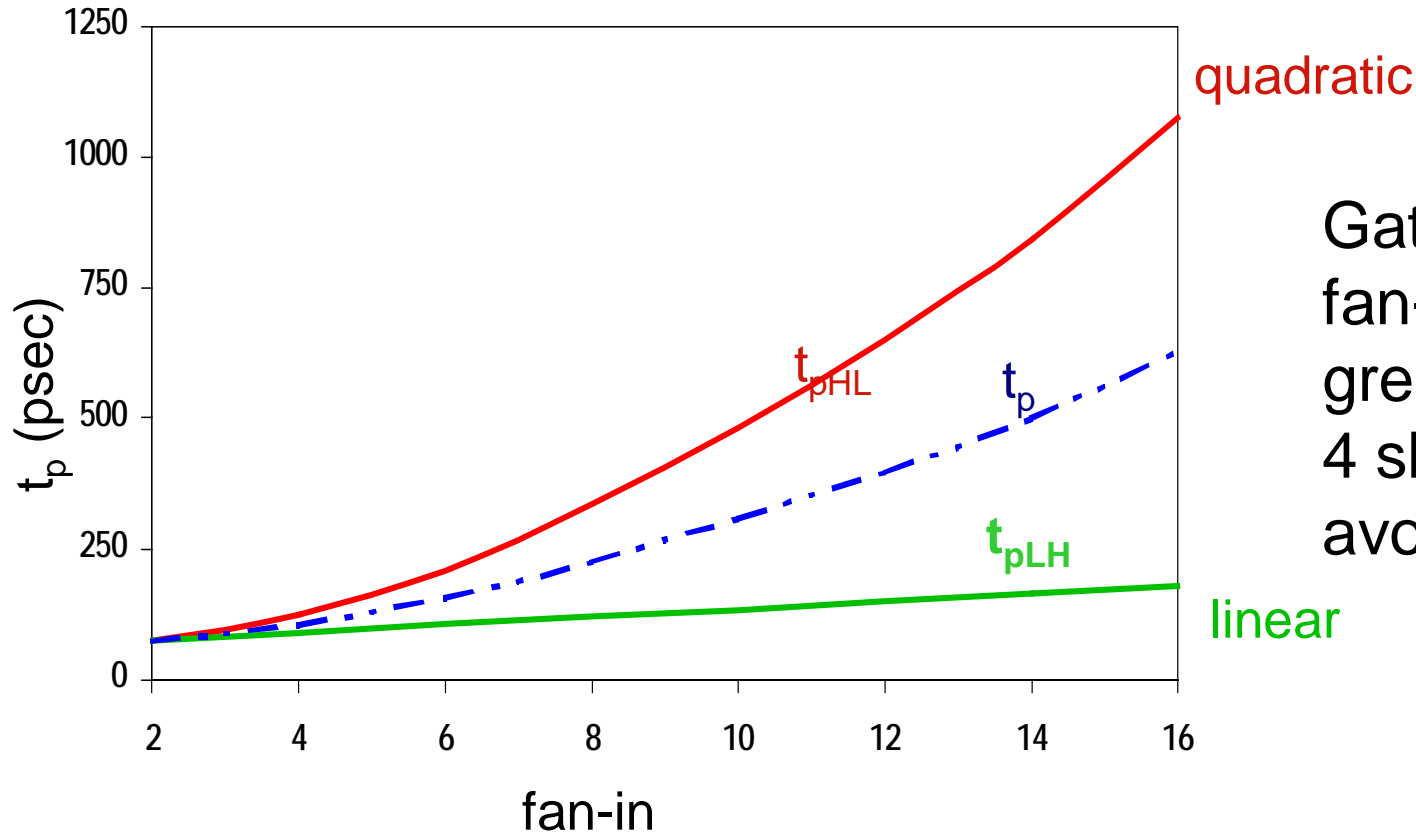
$$t_{pHL} = 0.69 R_{eqn}(C_1 + 2C_2 + 3C_3 + 4C_L)$$

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

$$t_{pHL} = 0.69 [R_1 C_1 + (R_1 + R_2) C_2 + (R_1 + R_2 + R_3) C_3 + (R_1 + R_2 + R_3 + R_4) C_L]$$

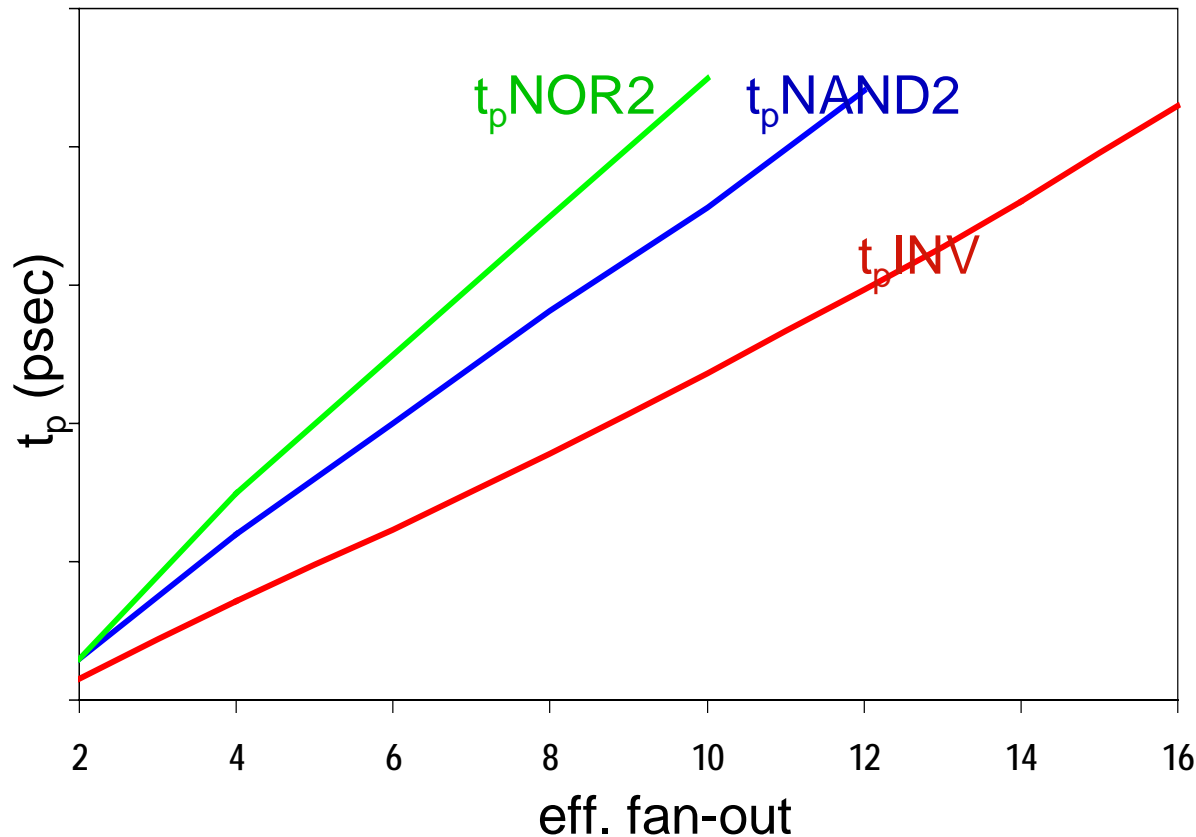
t_p of CMOS NAND as a Function of Fan-In

(Assuming a fixed fan out of one inverter)



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

t_p as a Function of Fan-Out



All gates have the same drive current.

Slope is a function of “driving strength”

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

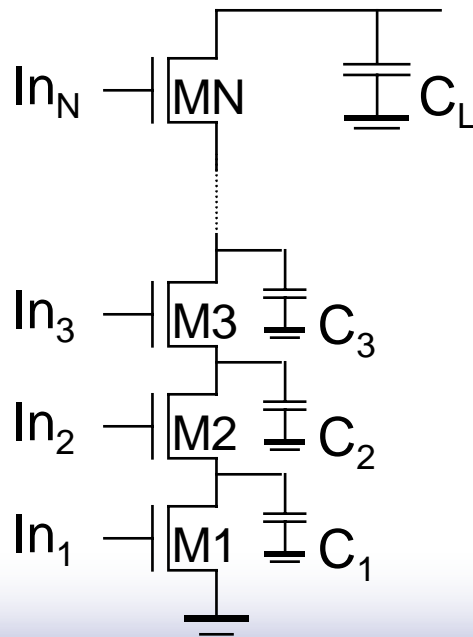
- Fan-in: **quadratic** due to increasing resistance and capacitance
- Fan-out: each additional fan-out gate adds **two** gate capacitances to C_L

Fast Complex Gates: Design Technique 1

□ Transistor sizing

- as long as fan-out capacitance dominates

□ Progressive sizing



Distributed RC line

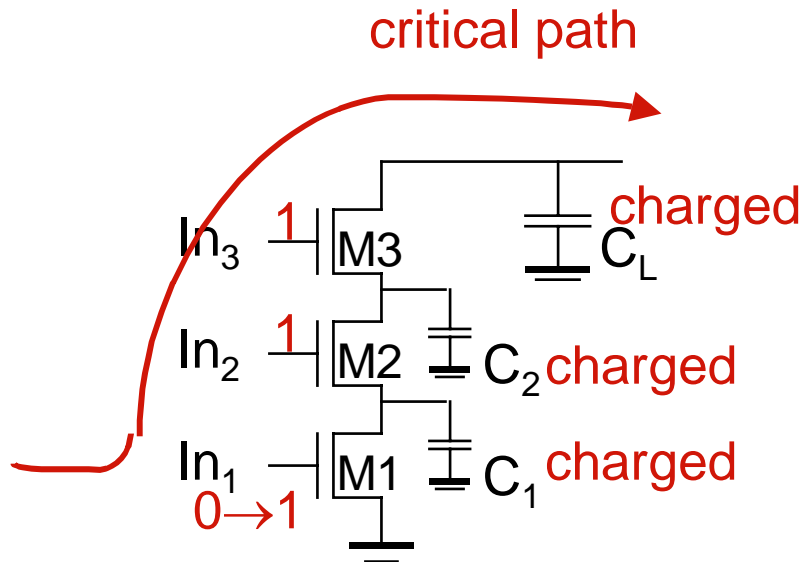
$M1 > M2 > M3 > \dots > MN$
(the fet closest to the
output is the smallest)

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

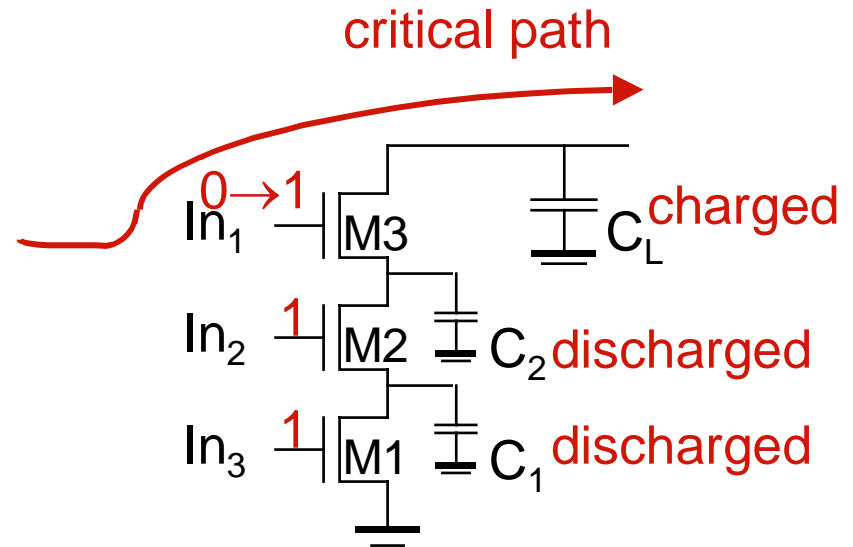
Fast Complex Gates: Design Technique 2

□ Transistor ordering

An input signal is called “critical” if it is the last signal to assume a stable value, the path which determines the ultimate speed is called “critical path”



delay determined by time to discharge C_L , C_1 and C_2



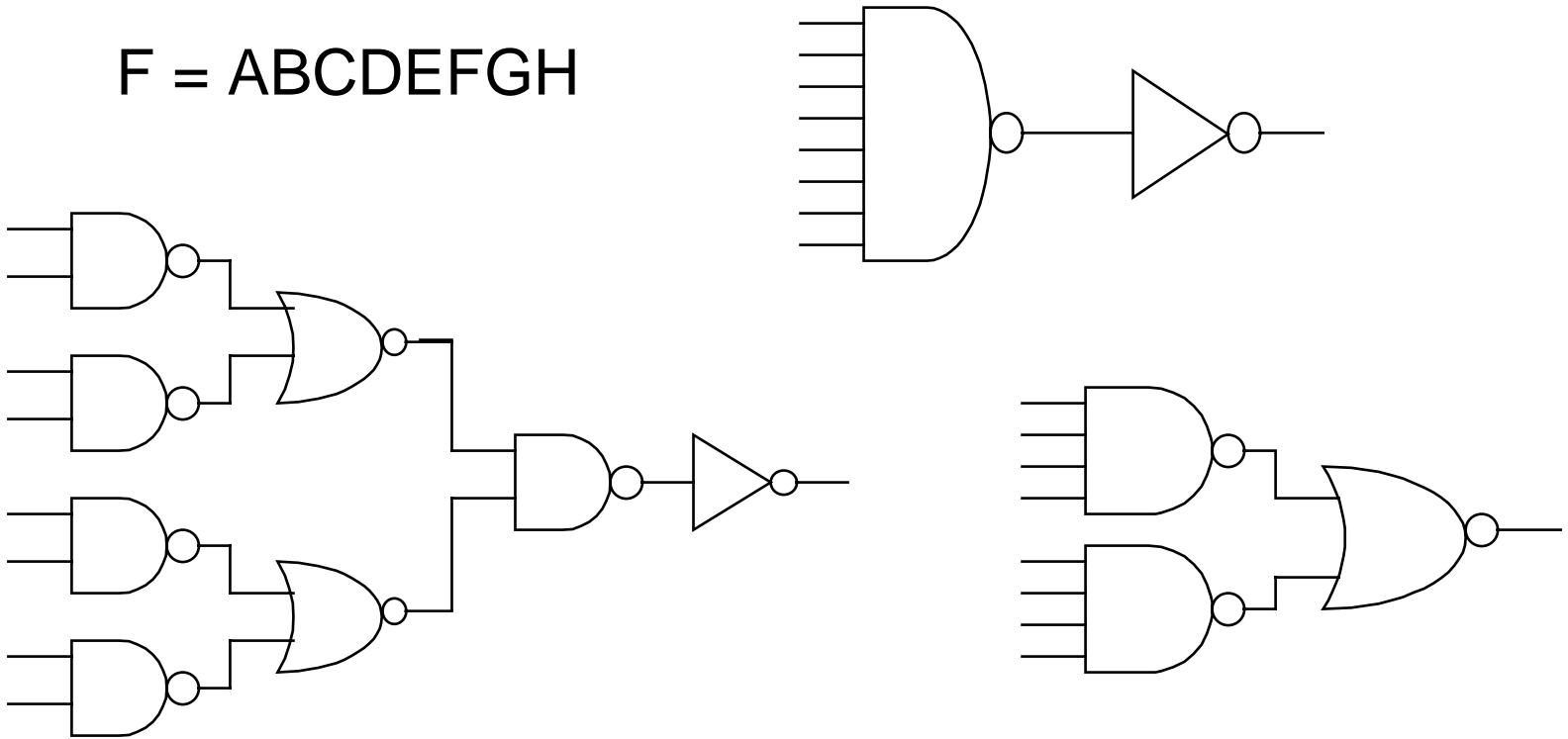
delay determined by time to discharge C_L

Fast Complex Gates:

Design Technique 3

□ Alternative logic structures

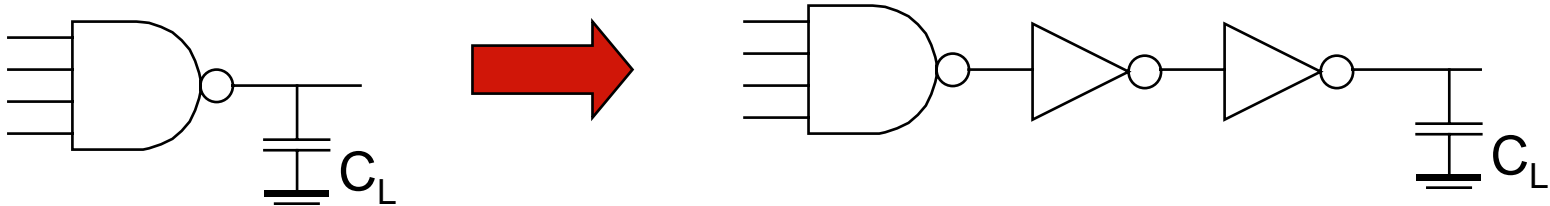
$$F = ABCDEFGH$$



Fast Complex Gates:

Design Technique 4

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



Logical Effort

$$t_p = t_{p0} (1 + C_{ext}/\gamma C_g) = t_{p0} (1 + f/\gamma)$$

$$t_p = t_{p0} (p + gf/\gamma)$$

p – ratio of the intrinsic (unloaded delays of the complex gate and the simple inverter

g – logical effort – how much more input capacitance a gate presents to deliver the same output current as an inverter (how much worse it is at producing output current than an inverter)

f – effective fanout (C_{ext}/C_g) (electrical effort)

Normalize everything to an inverter: $g_{inv}=1$, $p_{inv}=1$

$P = n$ for n input NAND and NOR gates

Assume $\gamma = 1$.

Logical Effort

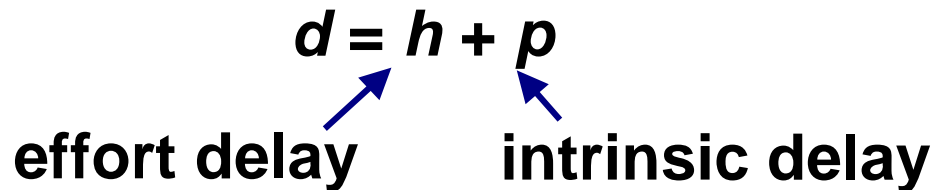
- ❑ Inverter has the smallest logical effort and intrinsic delay of all static CMOS gates
- ❑ Logical effort of a gate presents the ratio of its input capacitance to the inverter capacitance when sized to deliver the same current
- ❑ Logical effort increases with the gate complexity

Delay in a Logic Gate

Gate delay:

$$d = h + p$$

effort delay intrinsic delay



Effort delay (gate effort):

$$h = g f$$

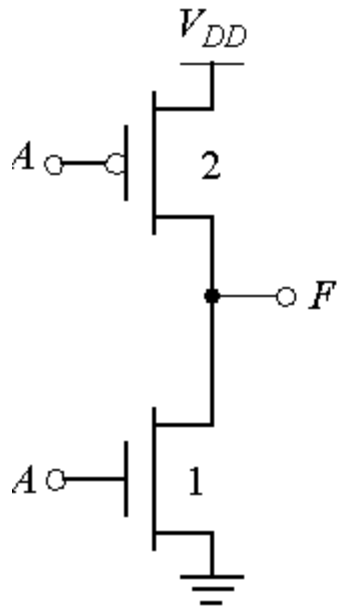
logical effort effective fanout = C_{out}/C_{in}



Logical effort is a function of topology, independent of sizing
Effective fanout (electrical effort) is a function of load/gate size

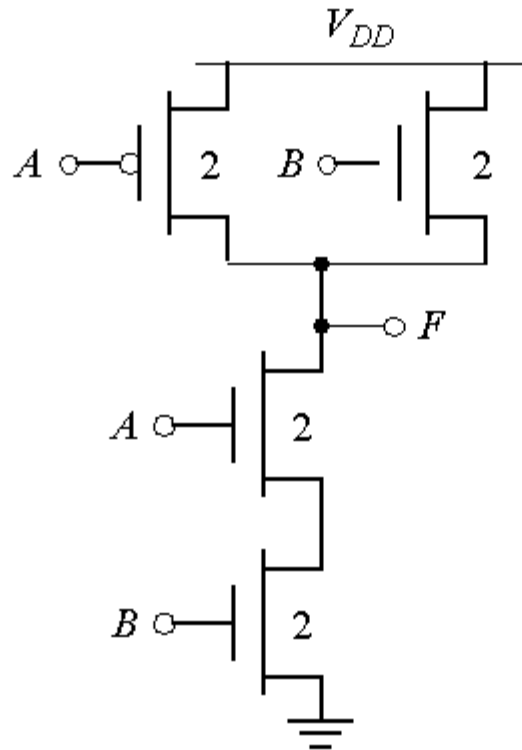
Logical Effort

Assuming PMOS/NMOS ratio of 2, the input capacitance of a minimum sized symmetrical inverter equals 3 times the gate capacitance of a minimum sized NMOS (called C_{unit})



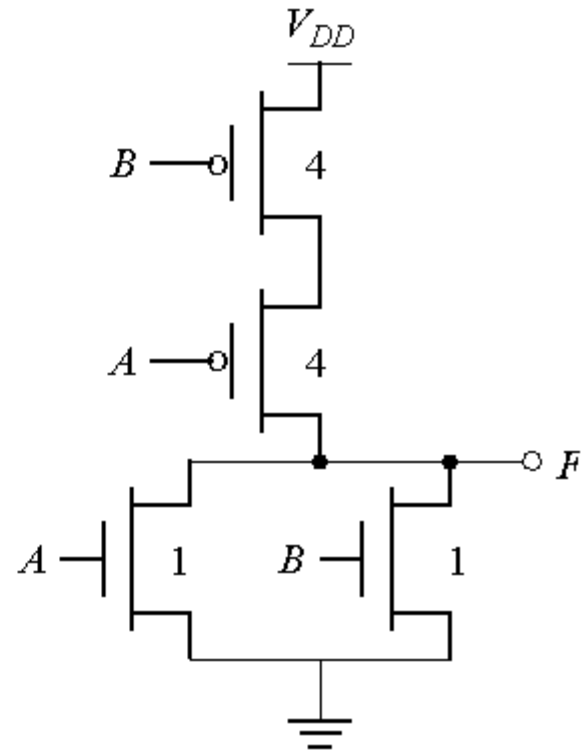
Inverter

$$g = 1$$
$$(3 C_{\text{unit}})$$



2-input NAND

$$g = 4/3$$
$$(4 C_{\text{unit}})$$



2-input NOR

$$g = 5/3$$
$$(5 C_{\text{unit}})$$

Intrinsic Delay Term, p

- ❑ The more involved the structure of the complex gate, the higher the intrinsic delay compared to an inverter

Gate Type	p
Inverter	1
n-input NAND	n
n-input NOR	n
n-way mux	$2n$
XOR, XNOR	$n 2^{n-1}$

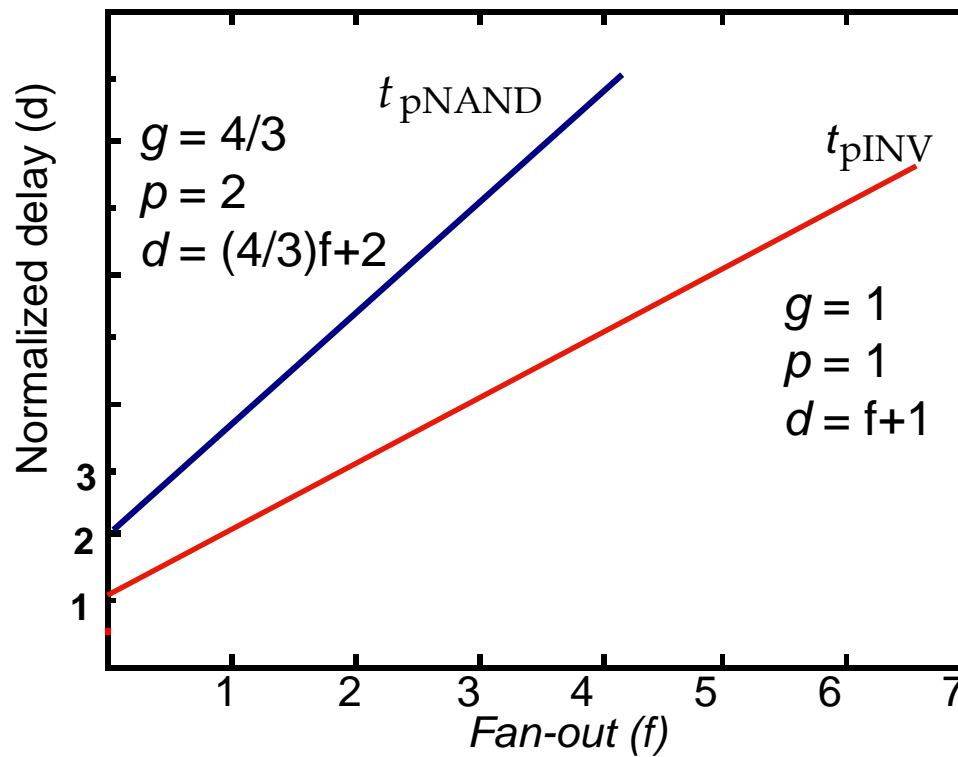
Ignoring second order effects such as internal node capacitances

Logical Effort

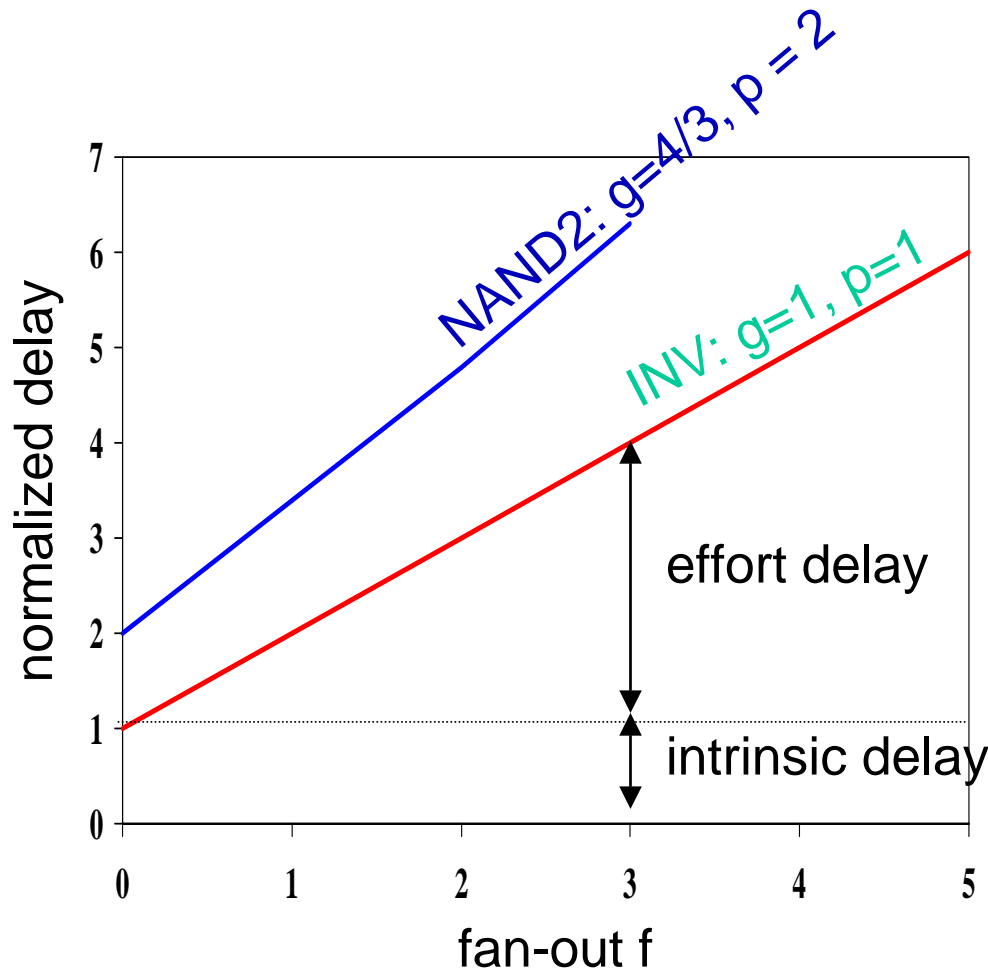
Gate Type	Number of Inputs			
	1	2	3	n
Inverter	1			
NAND		$4/3$	$5/3$	$(n + 2)/3$
NOR		$5/3$	$7/3$	$(2n + 1)/3$
Multiplexer		2	2	2
XOR		4	12	

For the same input capacitance, 2 input NAND and NOR gates have $4/3$ and $5/3$ less driving strength than the inverter

Logical Effort of Gates



Delay as a Function of Fan-Out



- The slope of the line is the logical effort of the gate ($d = p + fg$)
- The y-axis intercept is the intrinsic delay
- Can adjust the delay by adjusting the effective fan-out (by sizing) or by choosing a gate with a different logical effort
- Gate effort: $h = fg$

Multistage Networks

Total delay of a path:
$$t_p = \sum_{j=1}^N t_{p,j} = t_{p0} \sum_{j=1}^N \left(p_j + \frac{f_j g_j}{\gamma} \right)$$

Using a similar procedure with the sizing of the inverter chain (finding N-1 partial derivatives and equating to zero), we find that **each stage should bear the same gate effort**:

$$f_1 g_1 = f_2 g_2 = \dots = f_N g_N$$

Here we have some definitions:

Path effective fan-out (Path electrical effort): $F = C_L / C_{g1}$

Path logical effort: $G = g_1 g_2 \dots g_N$ $G = \prod_{i=1}^N g_i$

Branching effort of a logical gate on a path:

$$b = \frac{C_{on-path} + C_{off-path}}{C_{on-path}}$$

Path branching effort: $B = \prod_1^N b_i$

The path electrical effort can now be related to the electrical and branching efforts:

$$F = \prod_1^N \frac{f_i}{b_i} = \frac{\prod f_i}{B}$$

Finally the total path effort H can be defined:

$$H = \prod_{i=1}^N h_i = \prod_{i=1}^N g_i f_i = GFB$$

From here on, the analysis can be carried out as in the case of inverter chain. The gate effort that minimizes the path delay is:

$$h = \sqrt[N]{H}$$

And the minimum delay through the path is:

$$D = t_{p0} \left(\sum_{j=1}^N p_j + \frac{N \left(\sqrt[N]{H} \right)}{\gamma} \right)$$

We assume that a unit-size gate has a driving capability equal to a minimum-size inverter. This means that its input capacitance is g times larger than that of the reference inverter (C_{ref}). With s_1 the sizing factor of the first gate in the chain, the input capacitance of the chain can be written as:

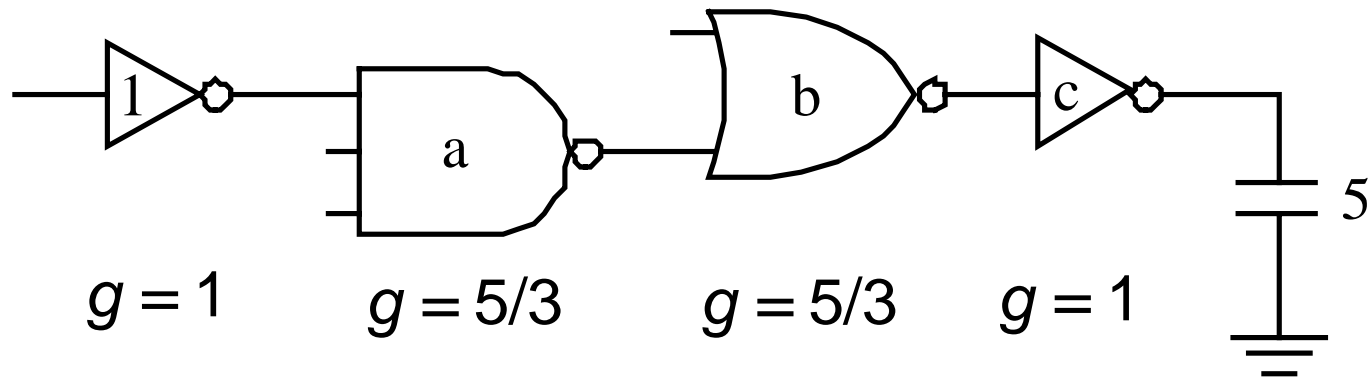
$$C_{g1} = g_1 s_1 C_{ref}$$

Including the branching effort, the input capacitance of gate 2 will be f_1/b_1 times larger:

$$g_2 s_2 C_{ref} = \left(\frac{f_1}{b_1} \right) g_1 s_1 C_{ref} \quad \text{For gate } i \text{ in the chain:}$$

$$s_i = \left(\frac{g_1 s_1}{g_i} \right) \prod_{j=1}^{i-1} \left(\frac{f_j}{b_j} \right)$$

Example: Optimize Path



Effective fanout, $F = 5$

$$G = 1 \times (5/3) \times (5/3) \times 1 = 25/9$$

$$H = G \text{FB (no branching)} = 125/9 = 13.9$$

$$h = \sqrt[4]{H} = 1.93$$

since $f_1 g_1 = f_2 g_2 = f_3 g_3 = f_4 g_4 = h$:

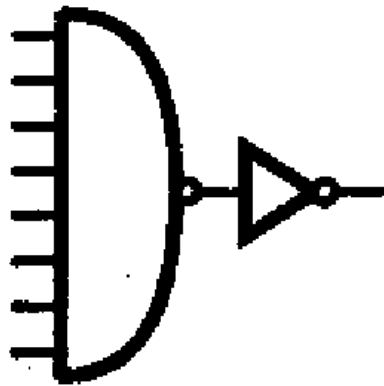
$$f_1 = 1.93, f_2 = 1.93 \times (3/5) = 1.16, f_3 = 1.16, f_4 = 1.93$$

$$a = s_2 = f_1 g_1 / g_2 = 1.16$$

$$b = s_3 = f_1 f_2 g_1 / g_3 = 1.34$$

$$c = s_4 = f_1 f_2 f_3 g_1 / g_4 = 2.60$$

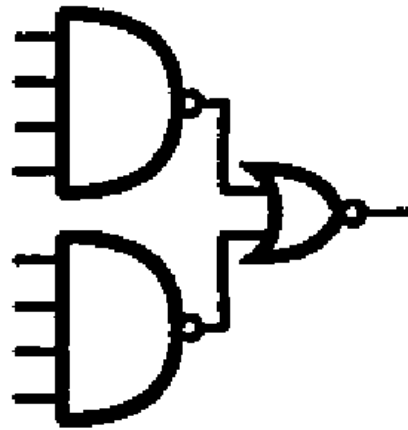
Path Logical Effort Variation With Restructuring (8 – input AND)



$$g=10/3 \quad g=1$$

(a)

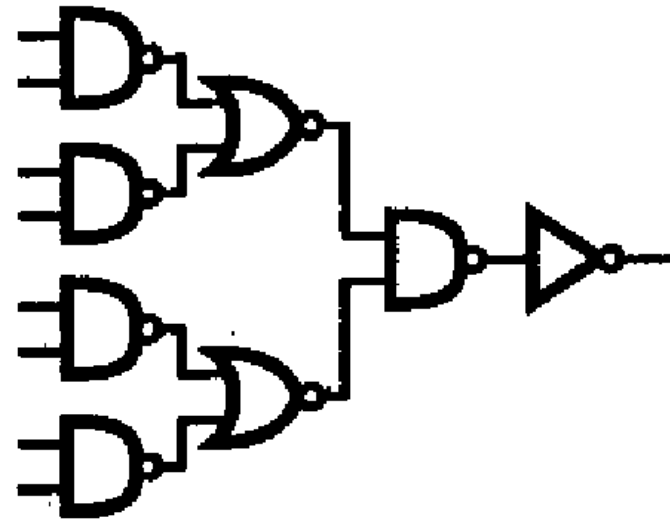
$$G=3.33$$



$$g=2 \quad g=5/3$$

(b)

$$G=3.33$$



$$g=4/3 \quad g=5/3 \quad g=4/3 \quad g=1$$

(c)

$$G=2.96$$

Dynamic Power Consumption is Data Dependent

❑ Switching activity, $P_{0 \rightarrow 1}$, has two components

λ A static component – function of the logic topology

λ A dynamic component – function of the timing behavior (glitching)

$$P_{dyn} = C_L V_{DD}^2 f_{0 \rightarrow 1} = C_L V_{DD}^2 P_{0 \rightarrow 1} f = C_{EFF} V_{DD}^2 f$$

2-input NOR Gate

A	B	Out
0	0	1
0	1	0
1	0	0
1	1	0

Static transition probability

$$P_{0 \rightarrow 1} = P_{out=0} \times P_{out=1} \\ = P_0 \times (1 - P_0)$$

With input **signal probabilities**

$$P_{A=1} = 1/2$$

$$P_{B=1} = 1/2$$

NOR static transition probability

$$= 3/4 \times 1/4 = 3/16$$

N_0 and N_1 are the number of zero and number of one entries in the output column of the truth table of the function.

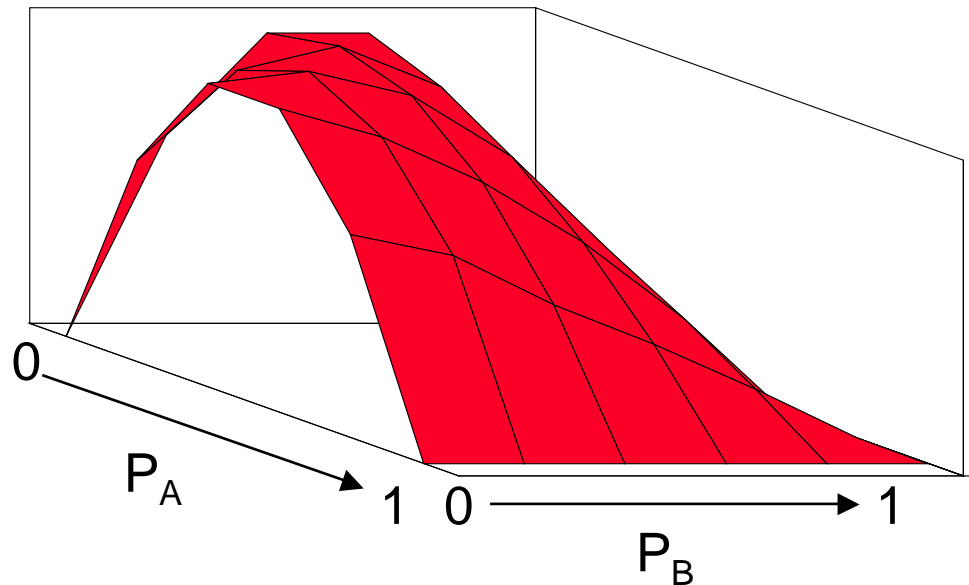
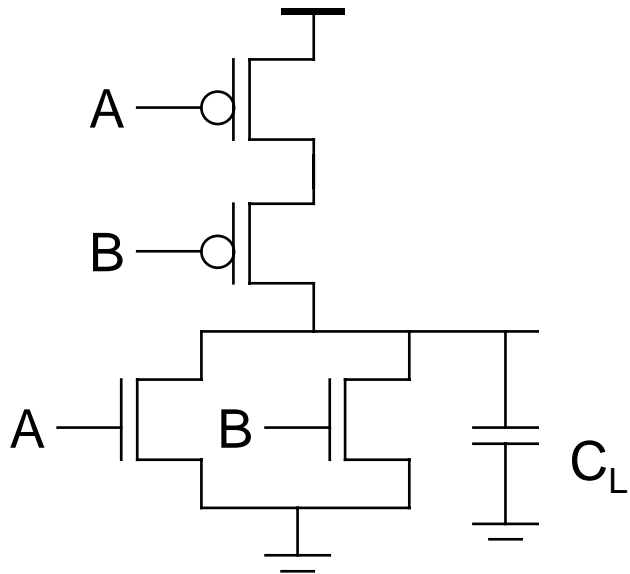
Assuming that the N inputs are independent and uniformly distributed (the 2^N possible states are equally likely):

$$P_{0 \rightarrow 1} = \frac{N_0}{2^N} \frac{N_1}{2^N} = \frac{N_0 (2^N - N_0)}{2^{2N}}$$

NOR Gate Transition Probabilities

- Switching activity is a strong function of the input signal statistics

λ P_A and P_B are the probabilities that inputs A and B are one

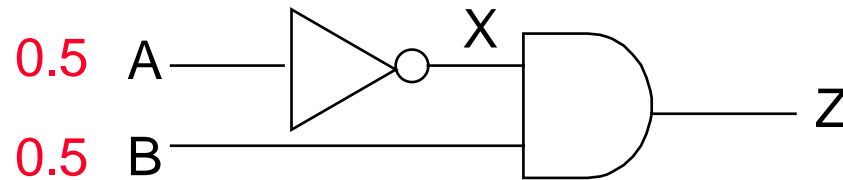


$$P_1 = (1-P_A)(1-P_B)$$

$$P_{0 \rightarrow 1} = P_0 \times P_1 = (1-(1-P_A)(1-P_B)) (1-P_A)(1-P_B)$$

Transition Probabilities for Some Basic Gates

	$P_{0 \rightarrow 1} = P_{\text{out}=0} \times P_{\text{out}=1}$
NOR	$(1 - (1 - P_A)(1 - P_B)) \times (1 - P_A)(1 - P_B)$
OR	$(1 - P_A)(1 - P_B) \times (1 - (1 - P_A)(1 - P_B))$
NAND	$P_A P_B \times (1 - P_A P_B)$
AND	$(1 - P_A P_B) \times P_A P_B$
XOR	$(1 - (P_A + P_B - 2P_A P_B)) \times (P_A + P_B - 2P_A P_B)$



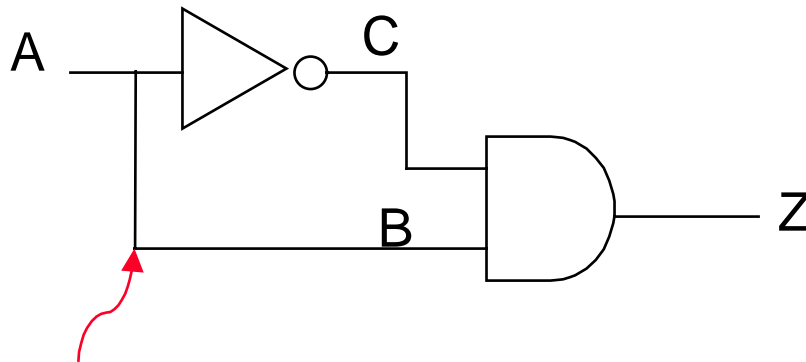
For X: $P_{0 \rightarrow 1} = P_0 \times P_1 = (1 - P_A) P_A$
 $= 0.5 \times 0.5 = 0.25$

For Z: $P_{0 \rightarrow 1} = P_0 \times P_1 = (1 - P_X P_B) P_X P_B$
 $= (1 - (0.5 \times 0.5)) \times (0.5 \times 0.5) = 3/16$

Inter-signal Correlations

- ❑ Determining switching activity is complicated by the fact that signals exhibit correlation in space and time

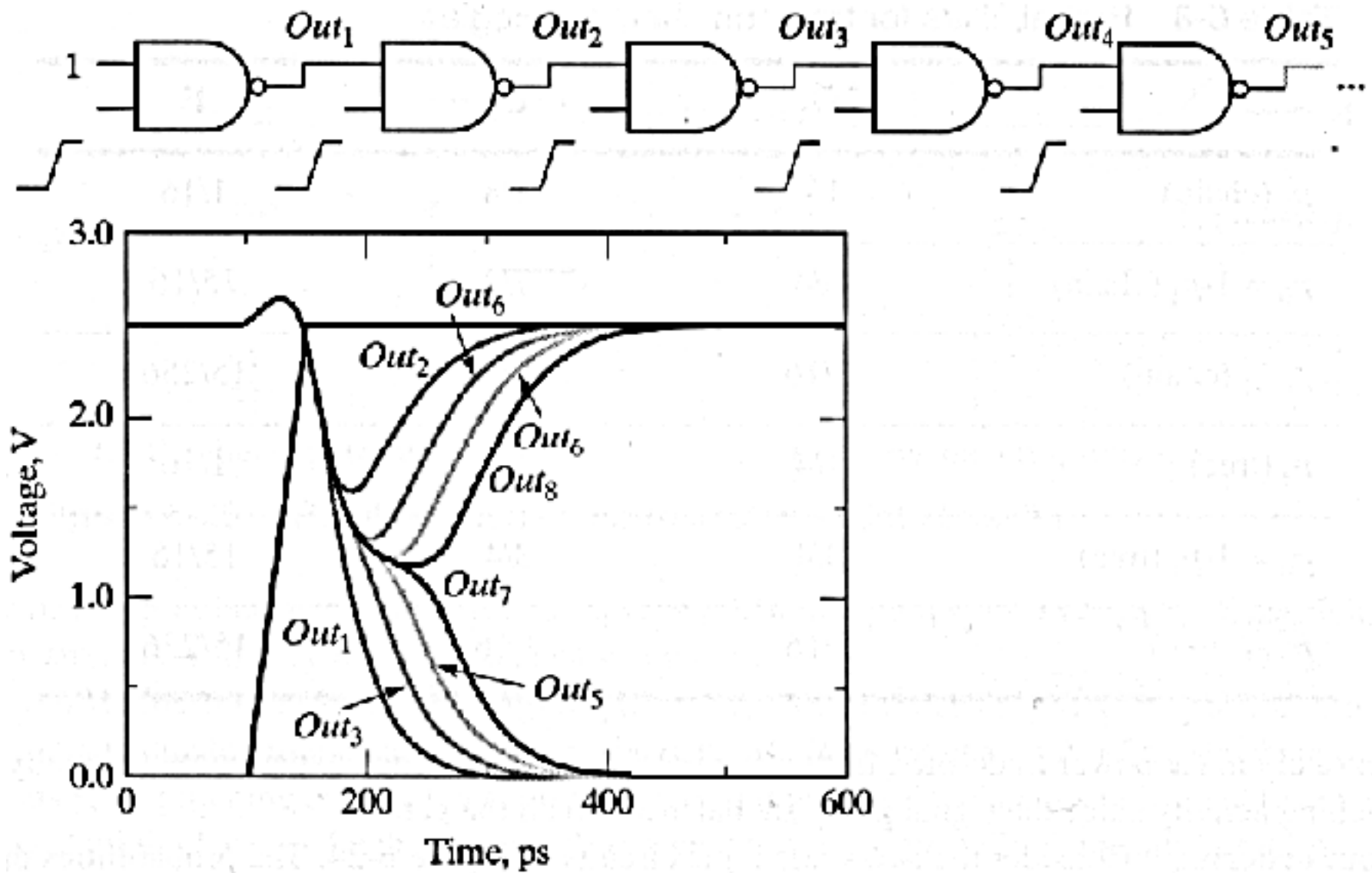
If B is not connected to A: $P(Z=1) = P(B=1, C=1) = 1/4$



Reconvergent fan-out

$$P(Z=1) = P(C=1|B=1) \cdot P(B=1) = 0$$

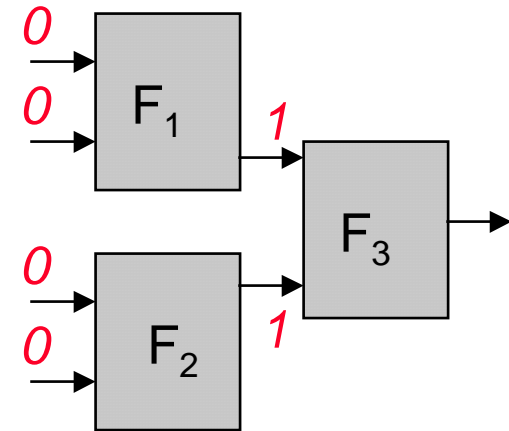
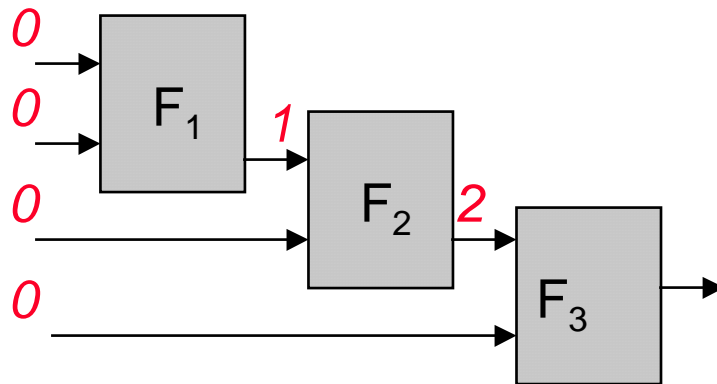
- ❑ Have to use **conditional probabilities**



Glitching in a chain of NAND gates

Balanced Delay Paths to Reduce Glitching

- ❑ Glitching is due to a mismatch in the path lengths in the logic network; if all input signals of a gate change simultaneously, no glitching occurs

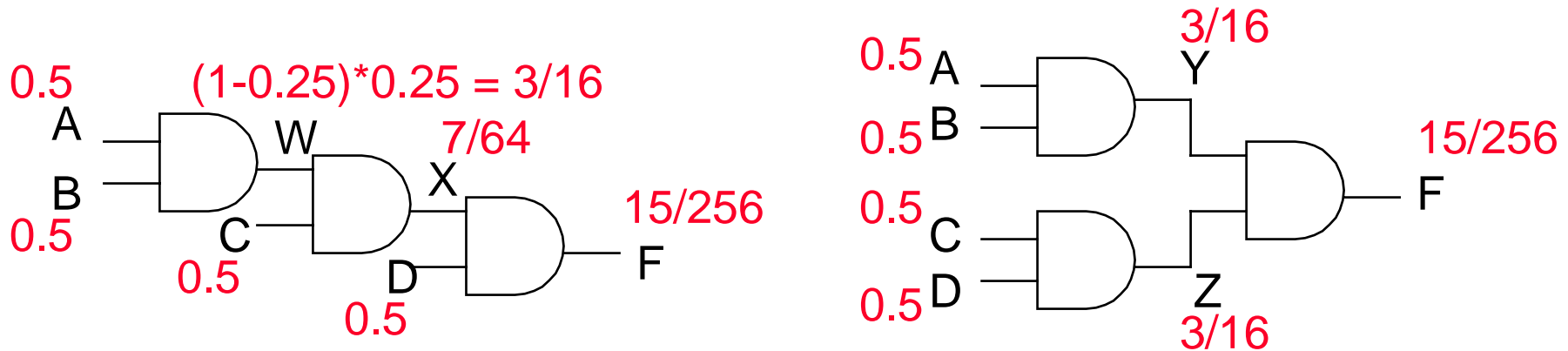


So equalize the lengths of timing paths through logic

Logic Restructuring

- Logic restructuring: changing the topology of a logic network to reduce transitions

$$\text{AND: } P_{0 \rightarrow 1} = P_0 \times P_1 = (1 - P_A P_B) \times P_A P_B$$



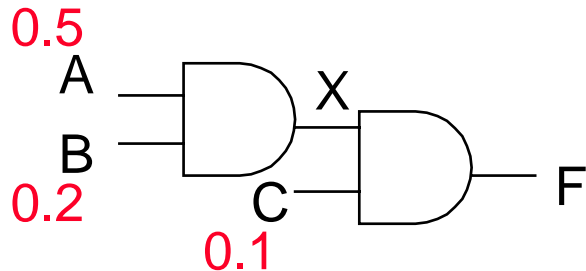
Chain implementation has a lower overall switching activity than the tree implementation for random inputs

Ignores glitching effects

Input Ordering

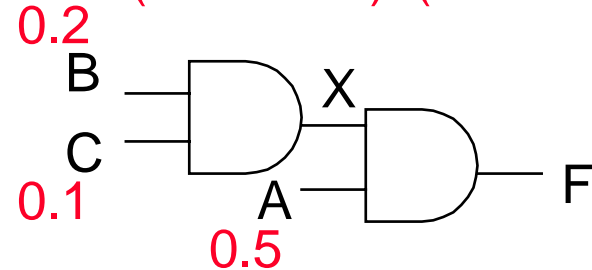
$$P_X =$$

$$(1 - 0.5 \times 0.2) \times (0.5 \times 0.2) = 0.09$$



$$P_X =$$

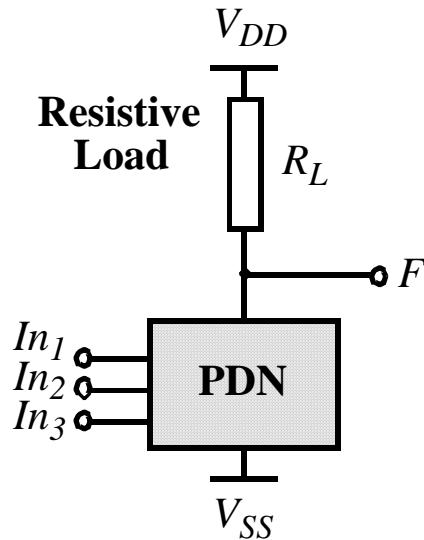
$$(1 - 0.2 \times 0.1) \times (0.2 \times 0.1) = 0.0196$$



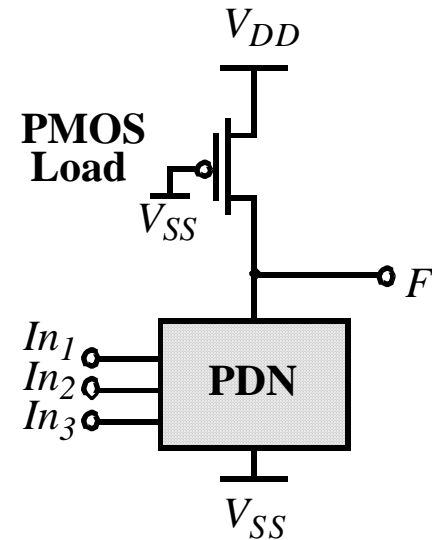
Beneficial to postpone the introduction of signals with a **high** transition rate (signals with signal probability close to 0.5)

Time – multiplexing resources may significantly effect switching activity!

Ratioed Logic



(a) resistive load

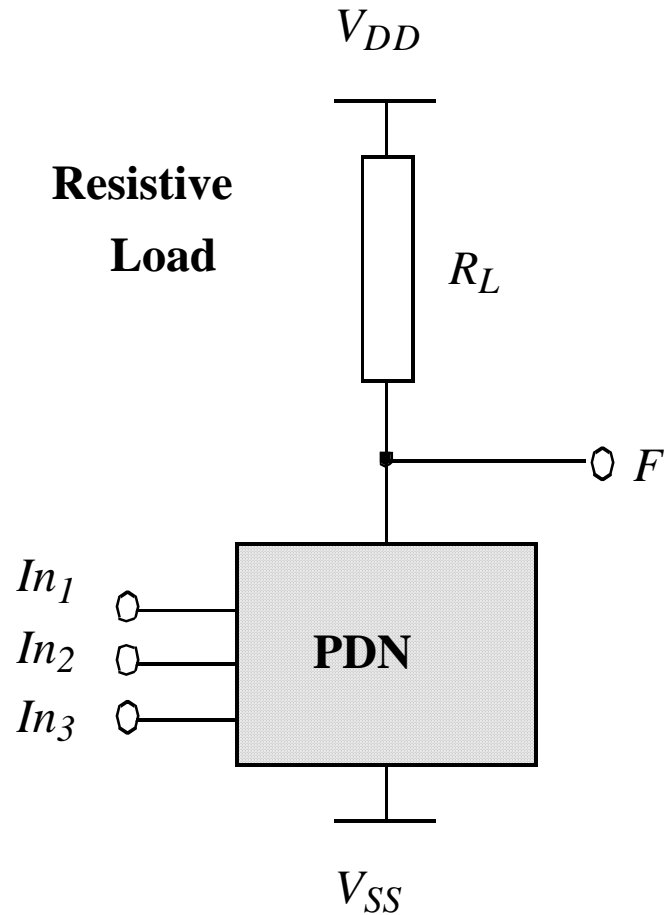


(c) pseudo-NMOS

Nominal $V_{OH} = V_{DD}$ but nominal $V_{OL} \neq 0$

Goal: to reduce the number of devices over complementary CMOS
(At the cost of reduced robustness and extra power)

Ratioed Logic

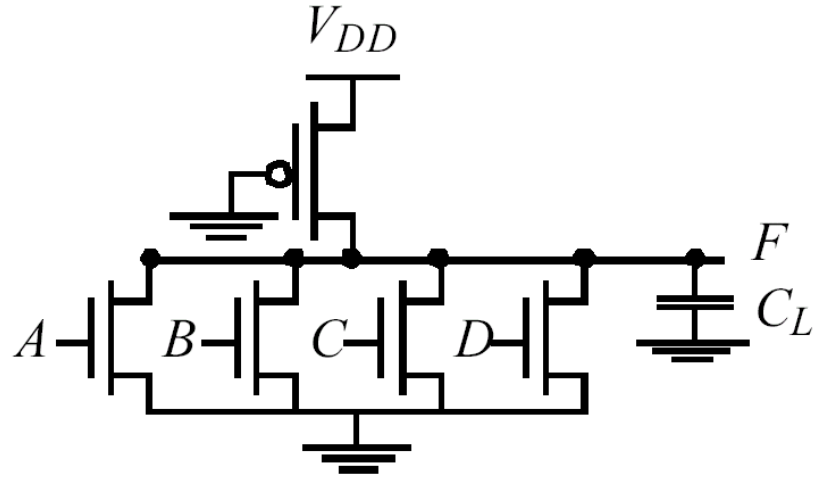


- **N transistors + Load**
- $V_{OH} = V_{DD}$
- $V_{OL} = \frac{R_{PN}}{R_{PN} + R_L} V_{DD}$
- **Assymetrical response**
- **Static power consumption**
- $t_{pL} = 0.69 R_L C_L$

Pseudo-NMOS

Pseudo-NMOS NOR Gate

Smaller area & Load but
Static power dissipation!!!



To obtain V_{OL} , equate the currents at load (PMOS) and driver (NMOS), when $V_{in} = V_{DD}$. It is reasonable to assume NMOS at linear (output should be close to 0V), PMOS at saturation

$$k_n \left((V_{DD} - V_{Tn}) V_{OL} - \frac{V_{OL}^2}{2} \right) + k_p \left((-V_{DD} - V_{Tp}) V_{DSATp} - \frac{V_{DSATp}^2}{2} \right) = 0$$

Assuming that $V_{OL} \ll (V_{DD} - V_T)$, $V_{DSATp} \ll (V_{DD} - V_T)$

and $|V_{Tn}| = |V_{Tp}|$

$$V_{OL} \approx \frac{k_p (V_{DD} + V_{Tp}) V_{DSATp}}{k_n (V_{DD} - V_{Tn})} \approx \frac{\mu_p W_p}{\mu_n W_n} V_{DSATp}$$

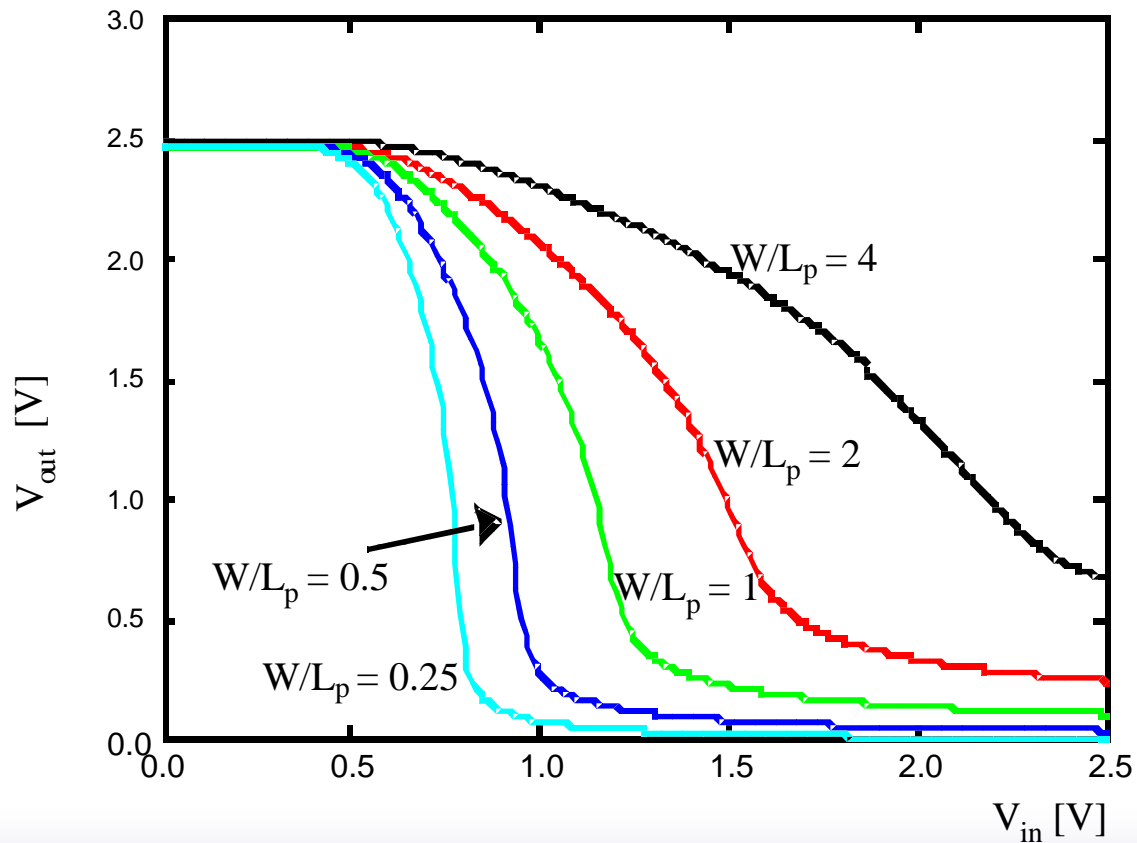
In order to make V_{OL} small, PMOS must be sized much smaller than NMOS. But this has negative effect on the propagation delay. A major disadvantage of the pseudo-NMOS is the static power dissipation when the output is low

$$P_{static} = V_{DD} I_{low} \approx V_{DD} \left| k_p \left((-V_{DD} - V_{Tp}) V_{DSATp} - \frac{V_{DSATp}^2}{2} \right) \right|$$

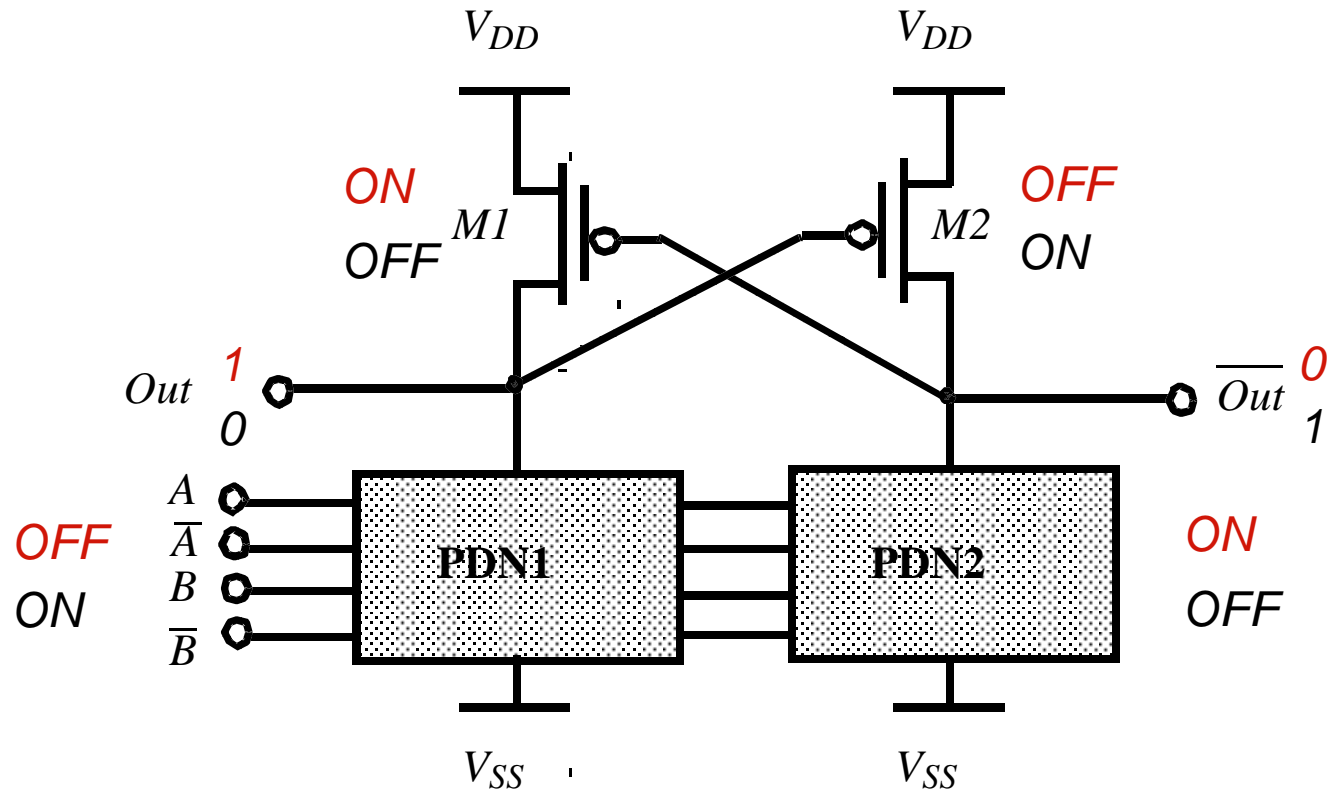
Pseudo-NMOS VTC

INVERTER

W/L_p	V_{oL}	P (μW)	t_{plh} (ps)
4	0.69	564	14
2	0.27	298	56
1	0.13	160	123
0.5	0.06	80	268
0.25	0.03	41	569

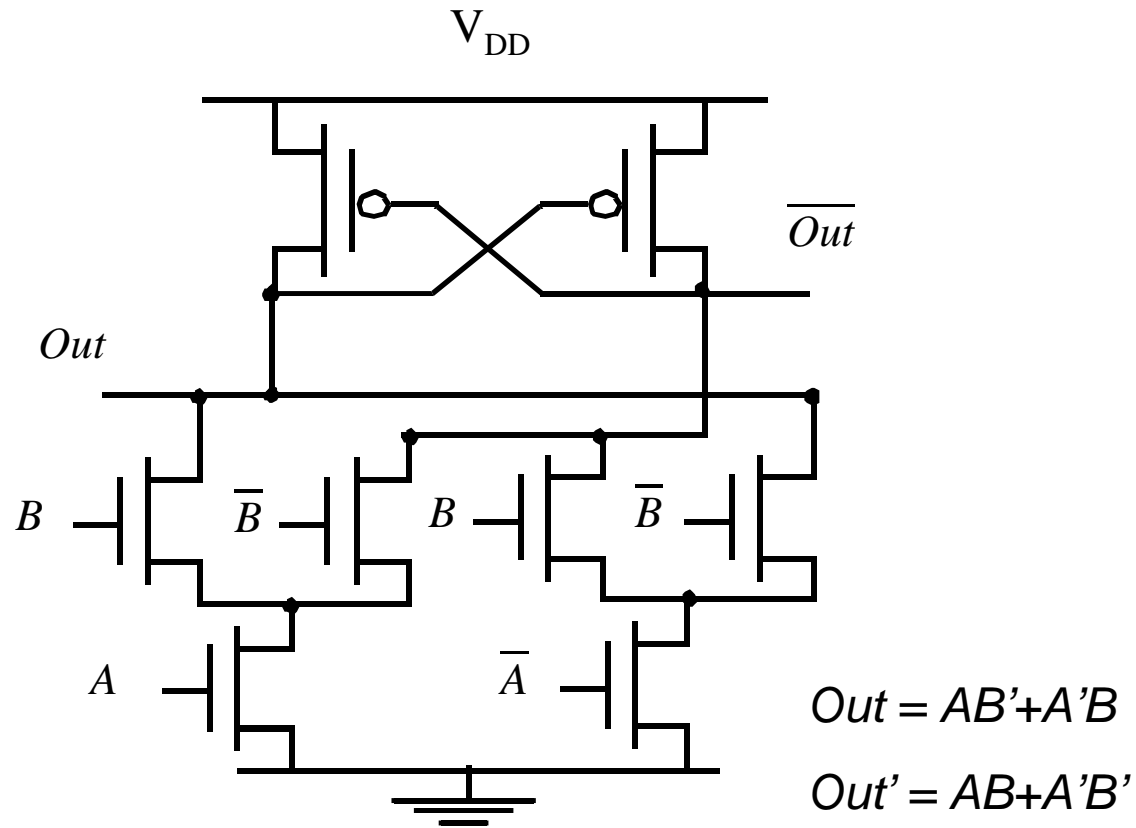


Improved Loads (Differential Logic & Positive Feedback)

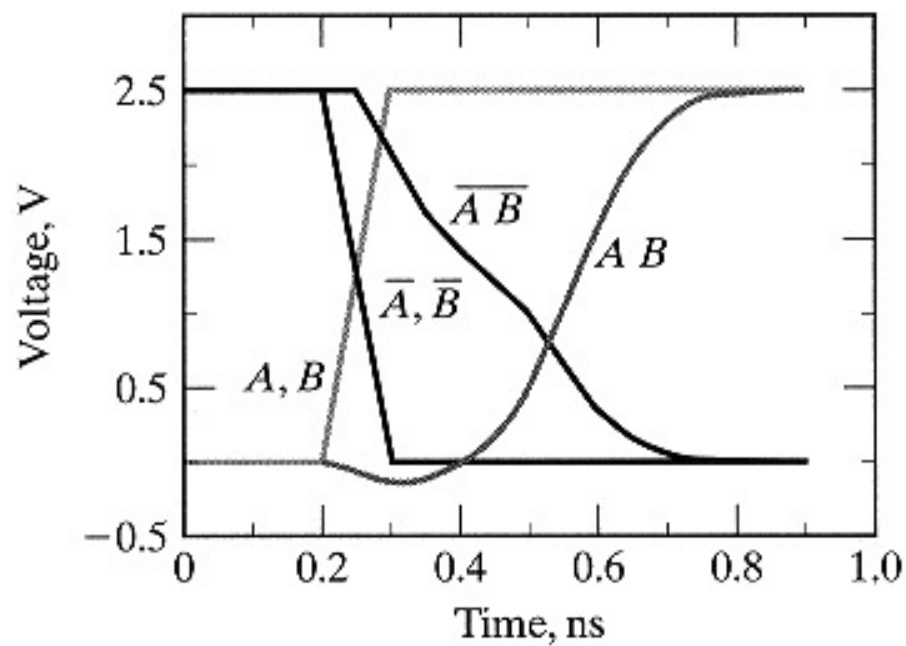
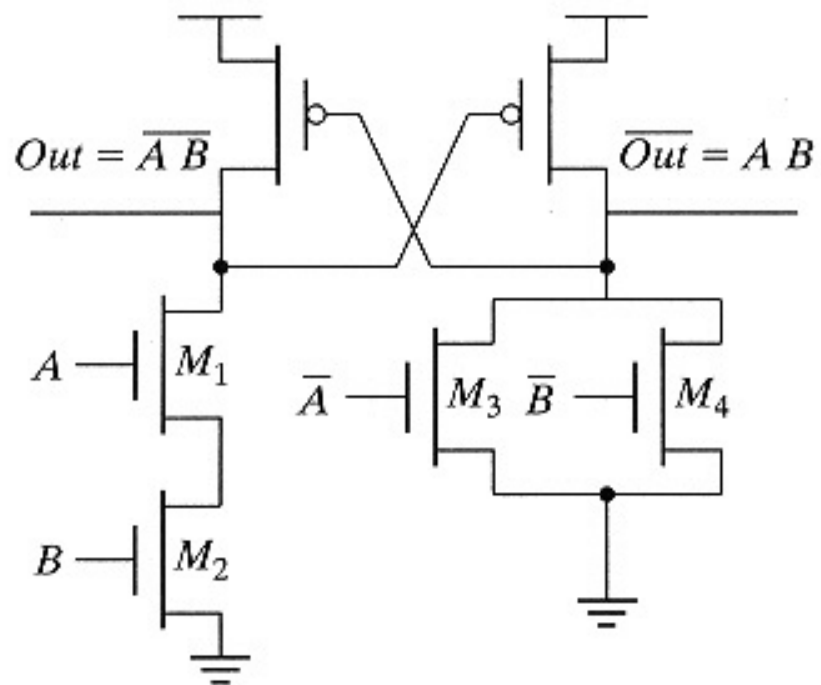


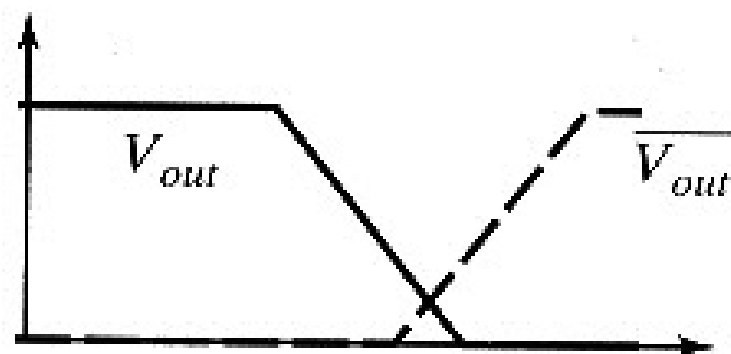
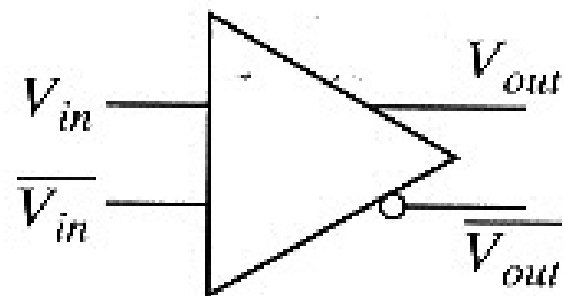
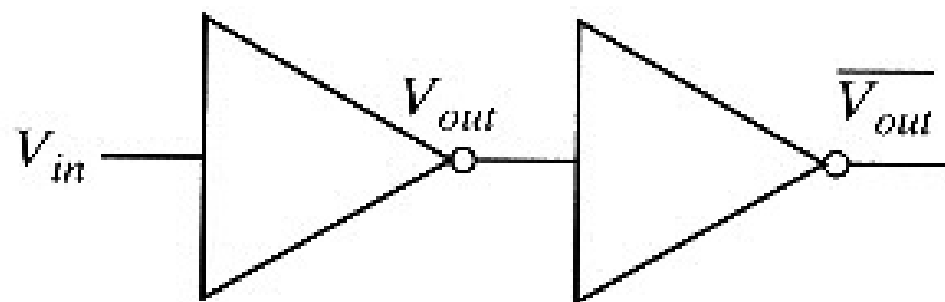
Differential Cascode Voltage Switch Logic (DCVSL)

DCVSL Example

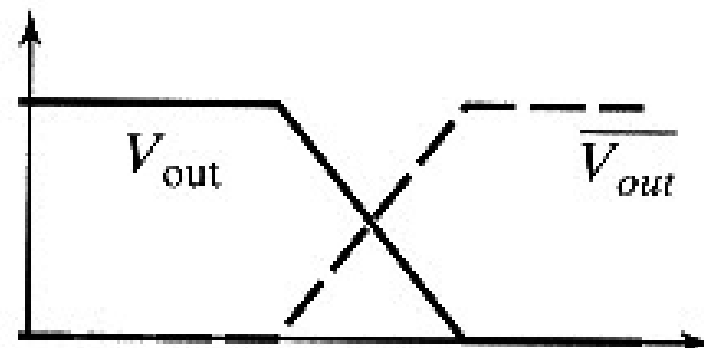


XOR – XNOR Gate





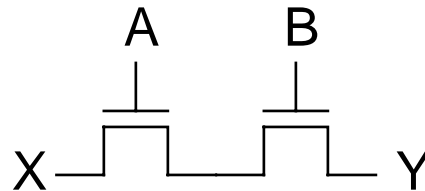
(a) Single ended



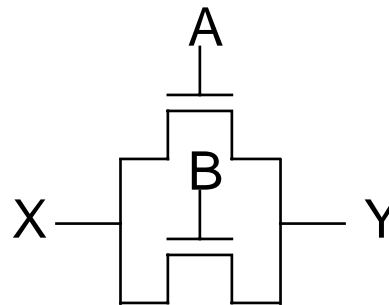
(b) Differential

NMOS Transistors in Series/Parallel

- ❑ Primary inputs drive both gate and source/drain terminals
- ❑ NMOS switch closes when the gate input is high



$X = Y$ if A and B

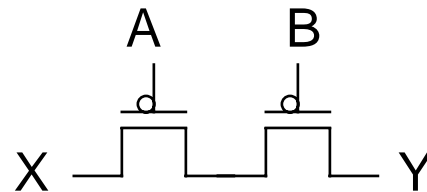


$X = Y$ if A or B

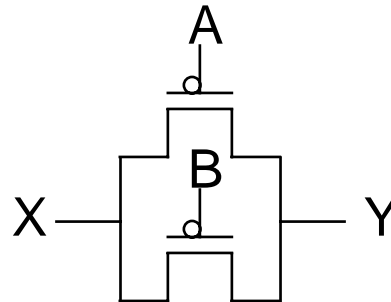
- ❑ Remember - NMOS transistors pass a **strong** 0 but a **weak** 1

PMOS Transistors in Series/Parallel

- ❑ Primary inputs drive both gate and source/drain terminals
- ❑ PMOS switch closes when the gate input is low



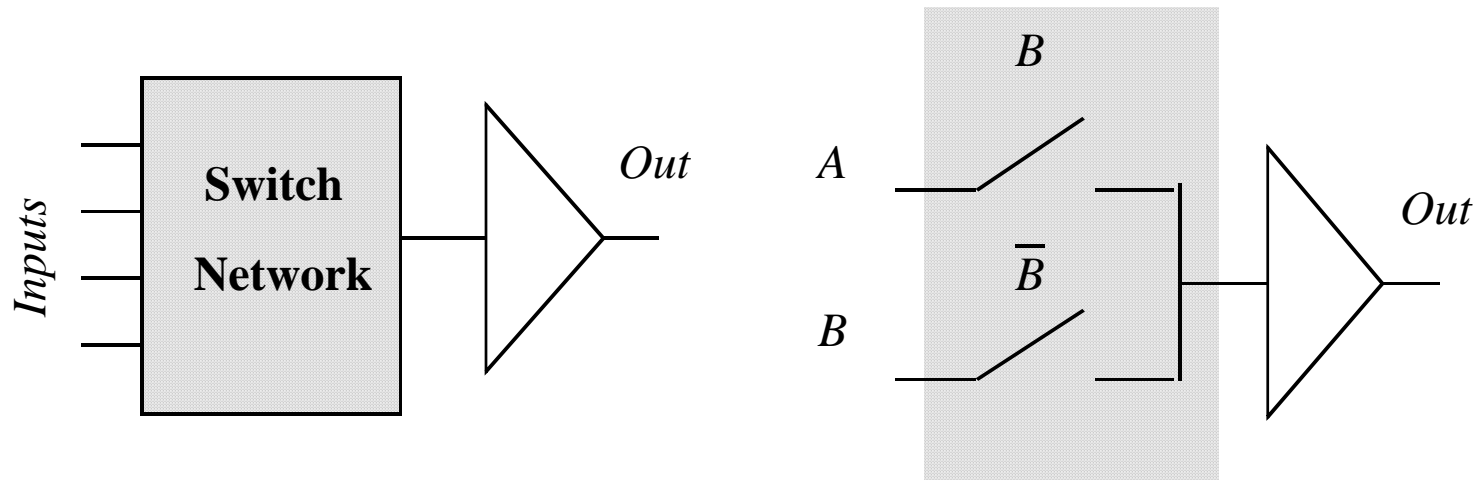
$$X = Y \text{ if } \overline{A} \text{ and } \overline{B} = \overline{A + B}$$



$$X = Y \text{ if } \overline{A} \text{ or } \overline{B} = \overline{A \bullet B}$$

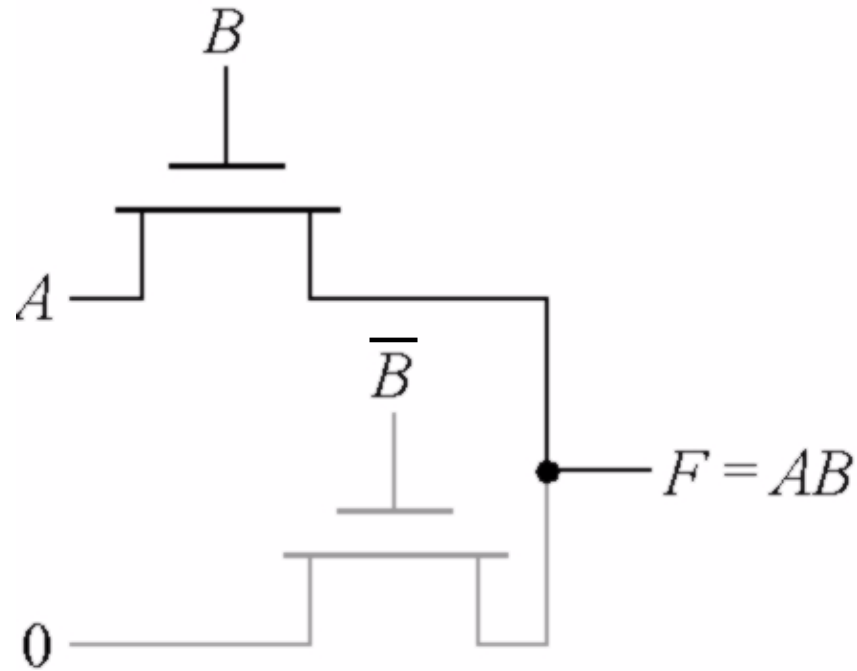
- ❑ Remember - PMOS transistors pass a **strong** 1 but a **weak** 0

Pass-Transistor Logic

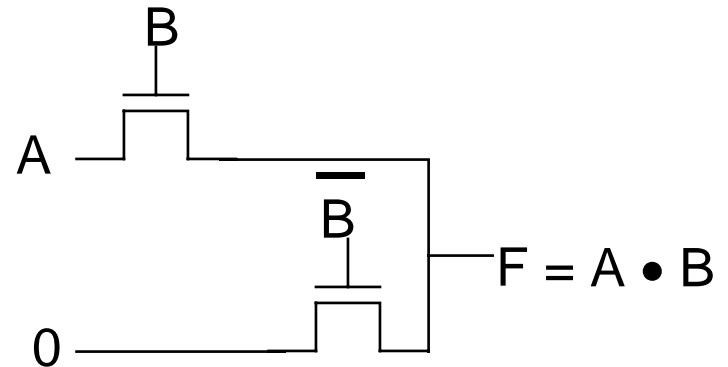
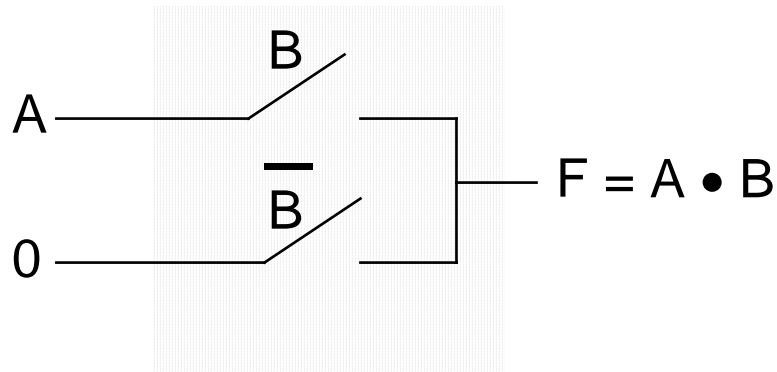


- **N transistors**
- **No static consumption**

Example: AND Gate

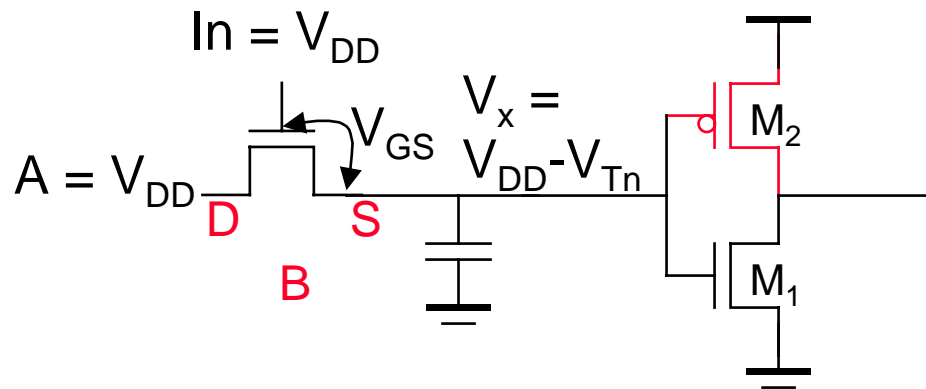


Pass Transistor (PT) Logic



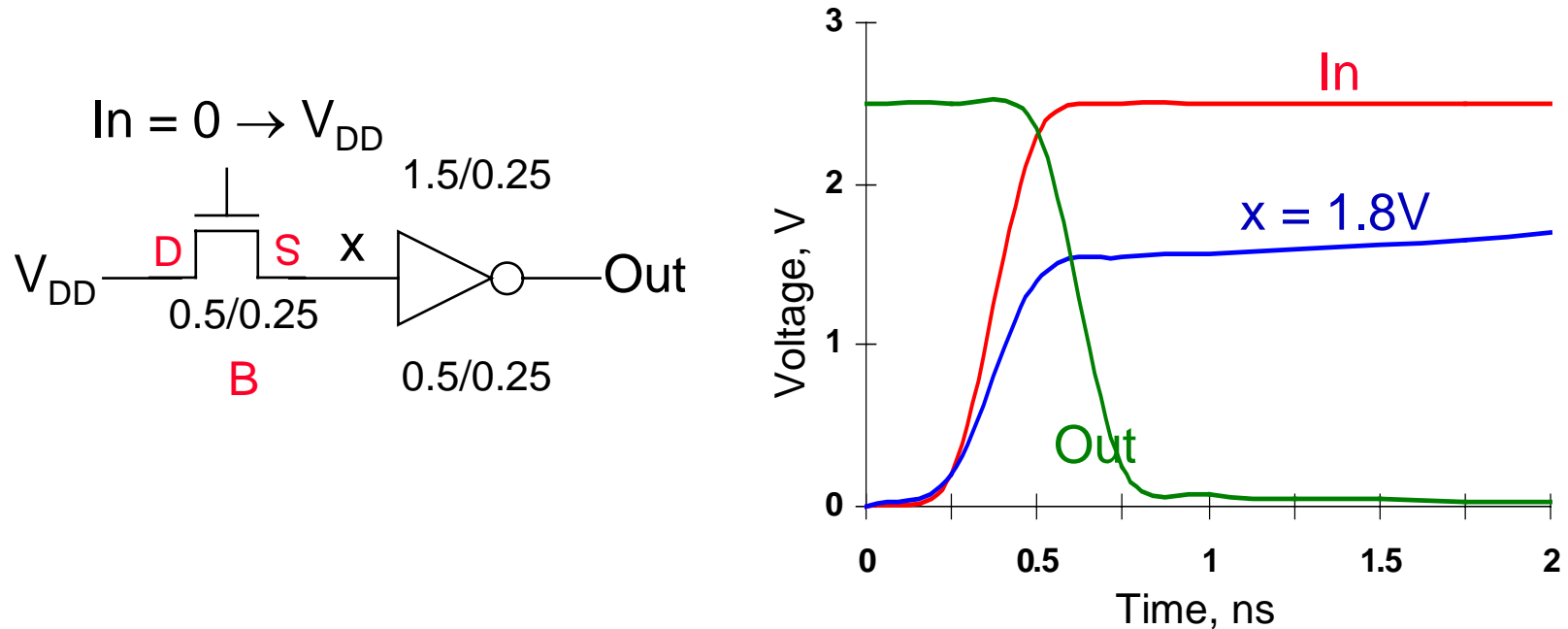
- ❑ Gate is static – a low-impedance path exists to both supply rails under all circumstances
- ❑ N transistors instead of $2N$
- ❑ No static power consumption
- ❑ Ratioless
- ❑ Bidirectional (versus unidirectional)

NMOS Only PT Driving an Inverter



- ❑ V_x does not pull up to V_{DD} , but $V_{DD} - V_{Tn}$
- ❑ Threshold voltage drop causes static power consumption (M_2 may be weakly conducting forming a path from V_{DD} to GND)
- ❑ Notice V_{Tn} increases of pass transistor due to **body effect** (V_{SB})

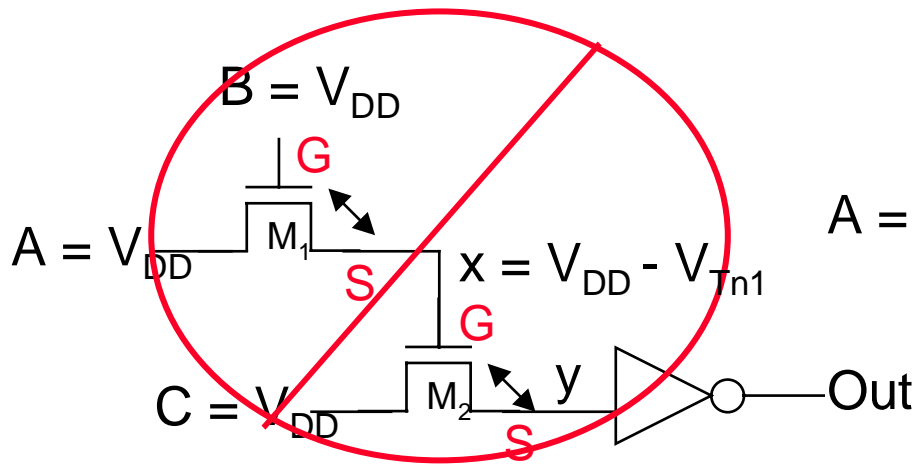
Voltage Swing of PT Driving an Inverter



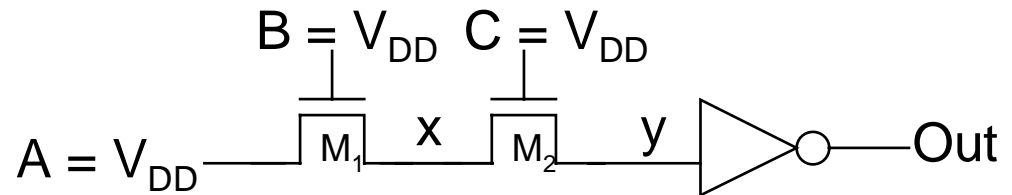
- ❑ **Body effect** – large V_{SB} at x - when pulling high (B is tied to GND and S charged up close to V_{DD})
- ❑ So the voltage drop is even worse

$$V_x = V_{DD} - (V_{Tn0} + \gamma(\sqrt{|2\phi_f| + V_x} - \sqrt{|2\phi_f|}))$$

Cascaded NMOS Only PTs



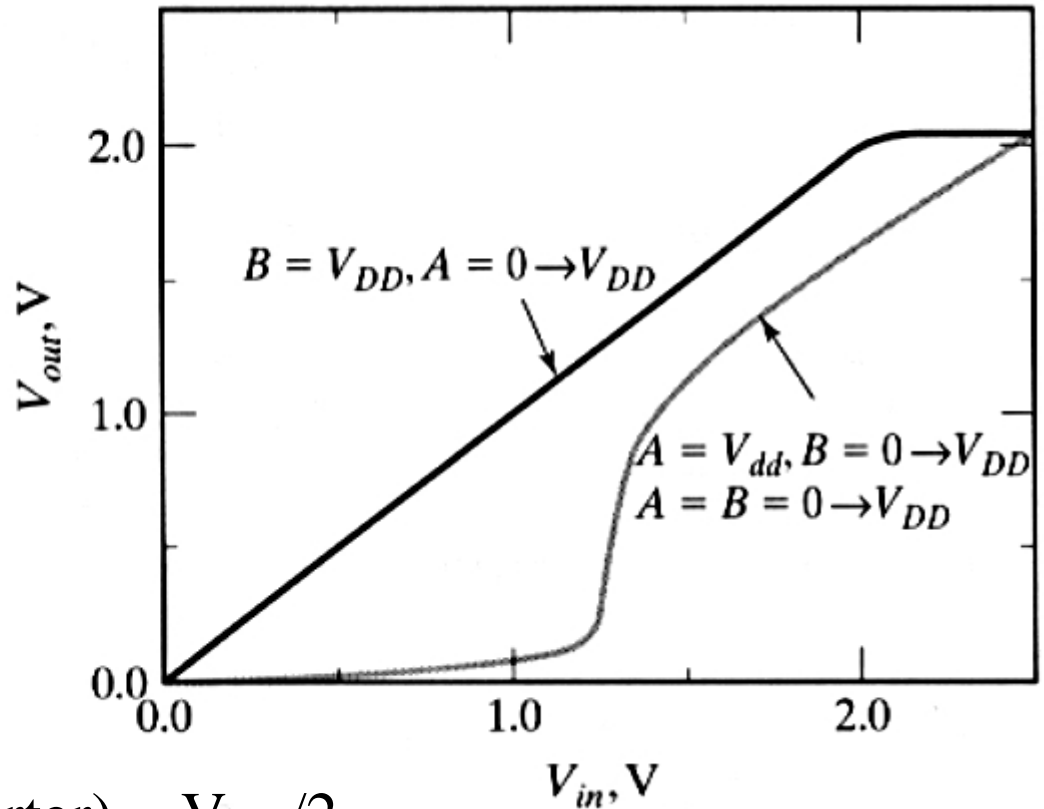
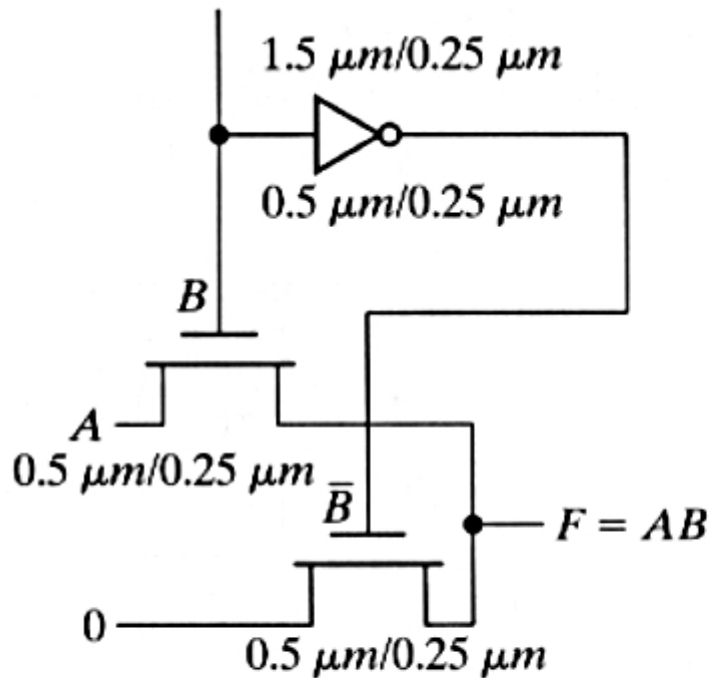
Swing on $y = V_{DD} - V_{Tn1} - V_{Tn2}$



Swing on $y = V_{DD} - V_{Tn2}$

- λ Pass transistor gates should **never** be cascaded as on the left
- λ Logic on the right suffers from static power dissipation and reduced noise margins

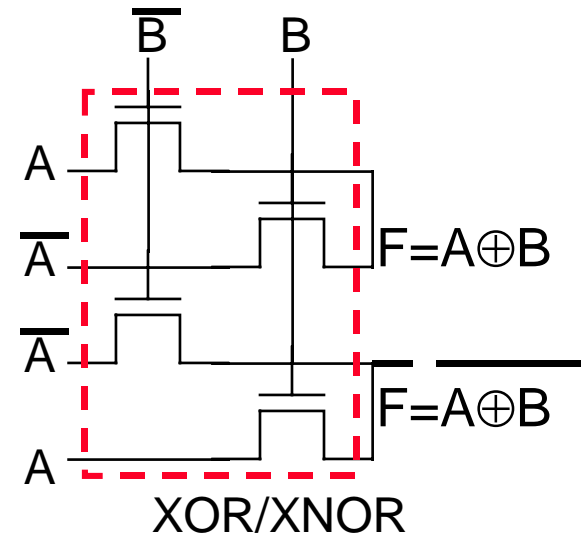
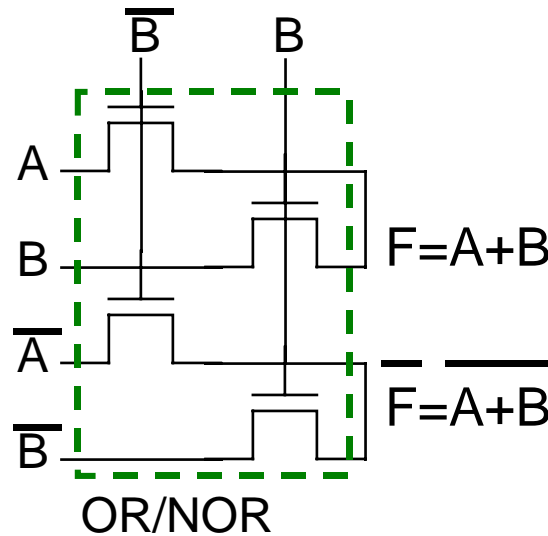
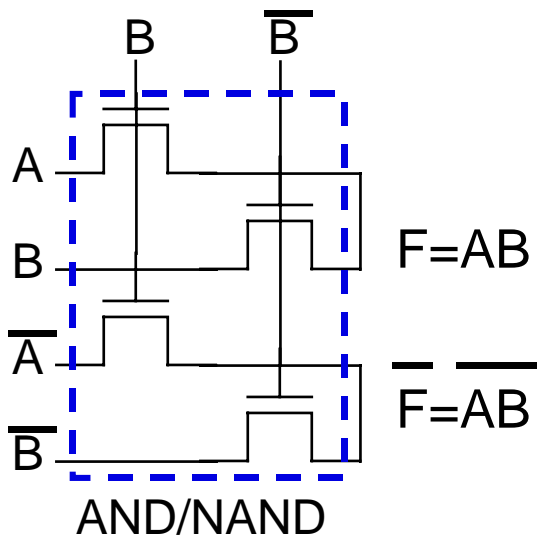
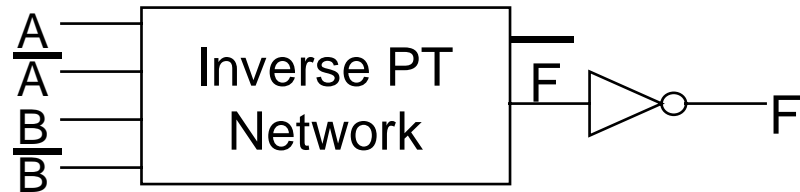
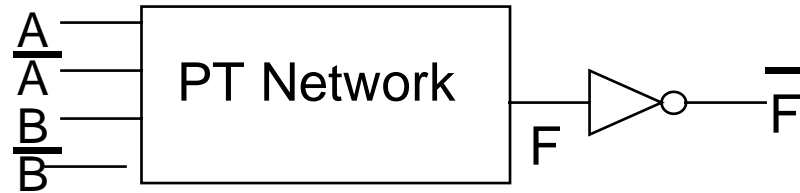
VTC of the pass transistor AND gate



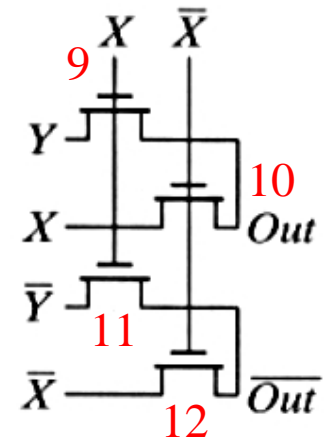
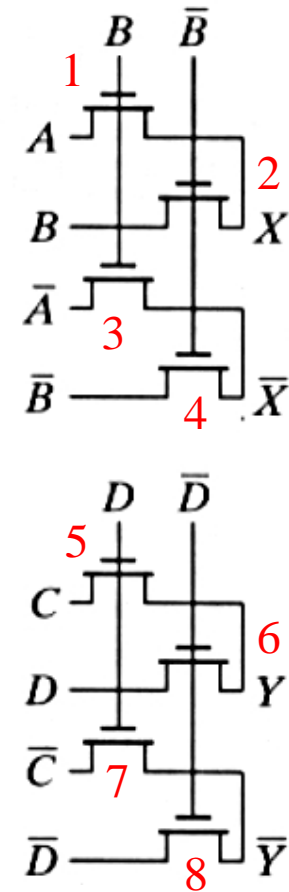
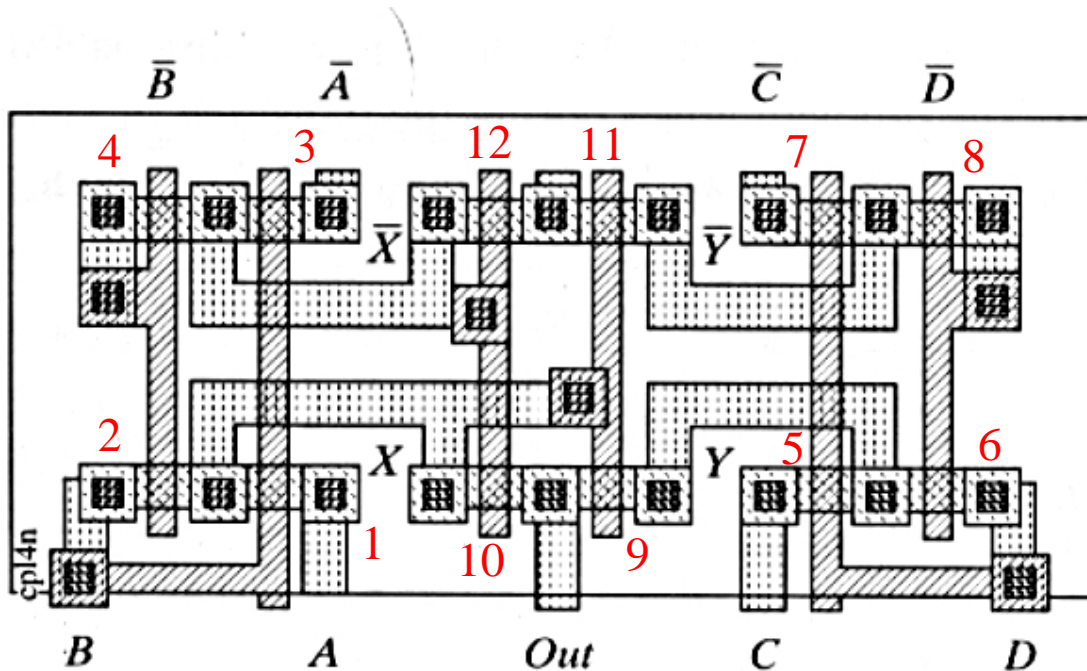
Assuming V_M (of the inverter) = $V_{DD}/2$

Pure pass-transistor gate is not regenerative

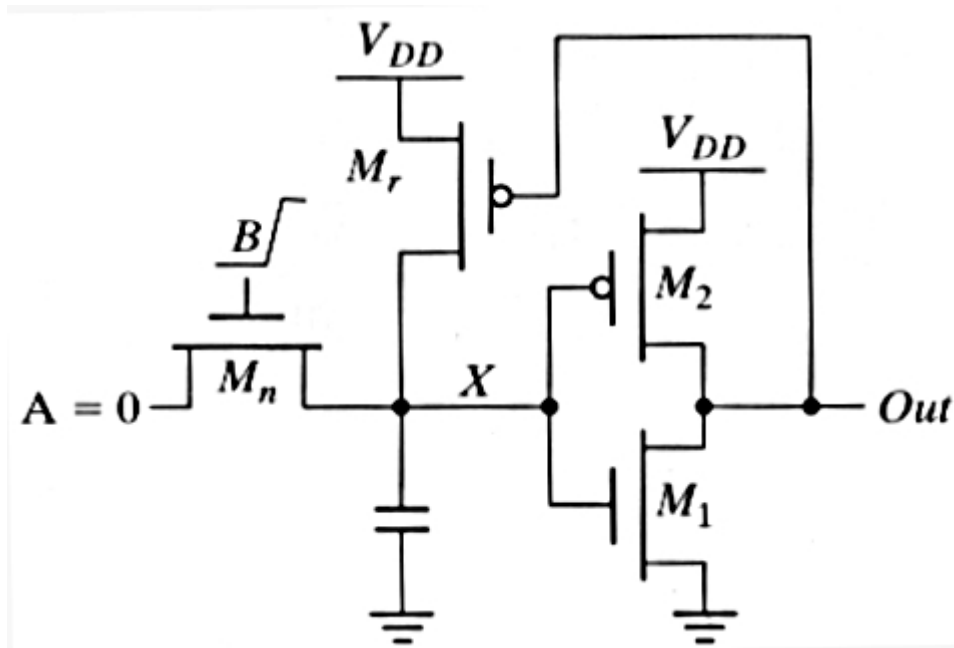
Differential PT Logic (CPL)



4-input AND/NAND gate in CPL



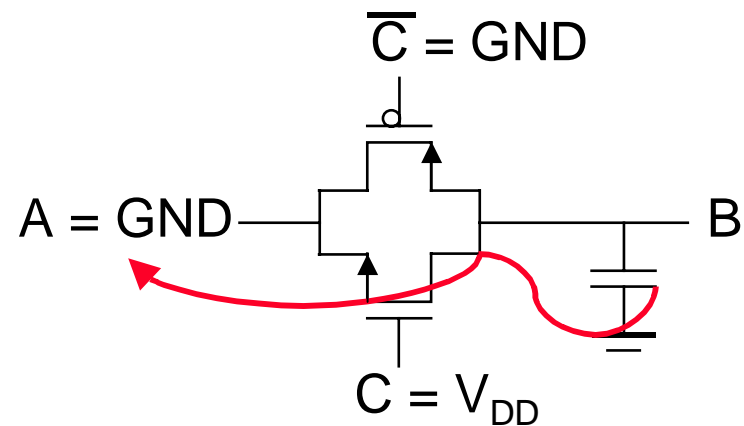
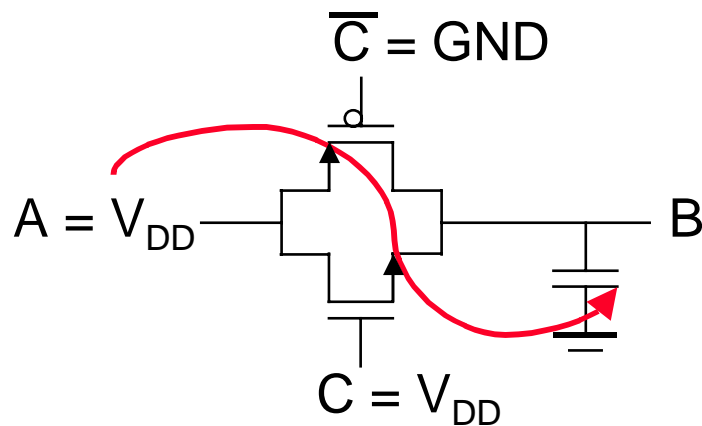
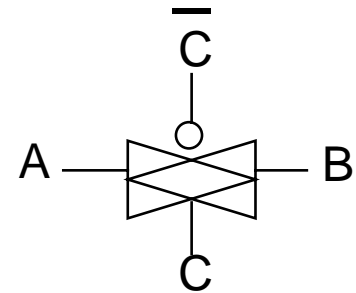
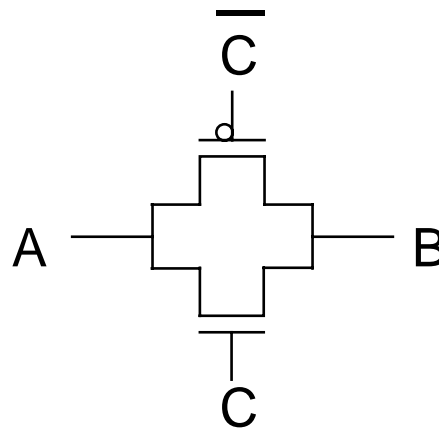
LEVEL RESTORATION



When A goes high, all nodes are either 0 or V_{DD} . To pull node X down, M_n need to be stronger than M_r . That requires careful transistor sizing (if R_r is too small with respect to R_n , it is impossible to bring V_x below the switching threshold of the inverter).

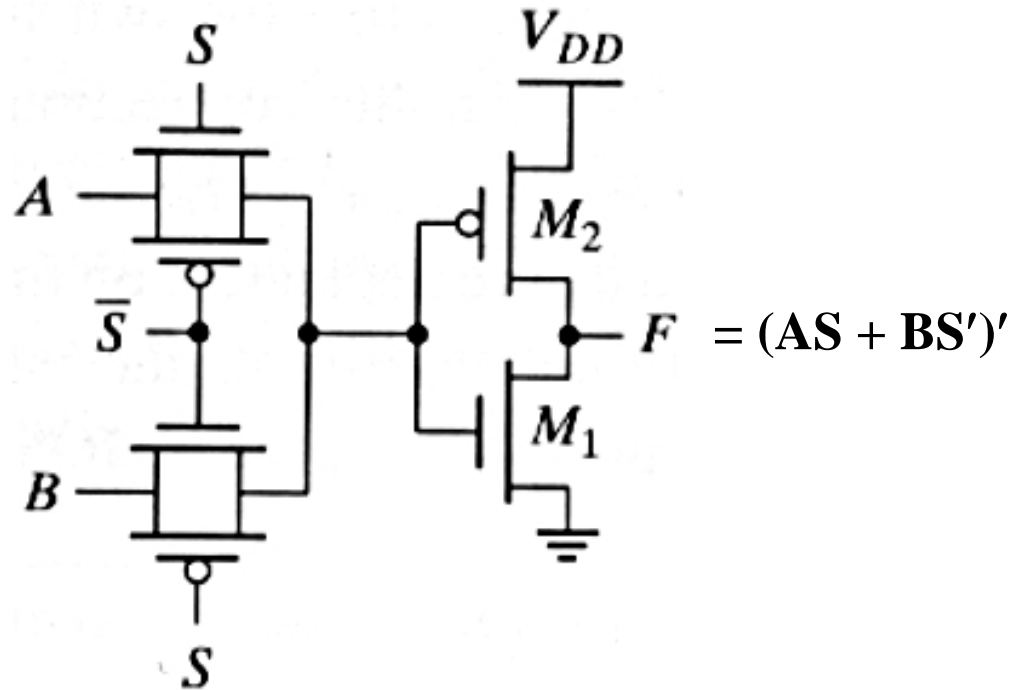
Transmission Gates (TGs)

- Most widely used solution for level restoration

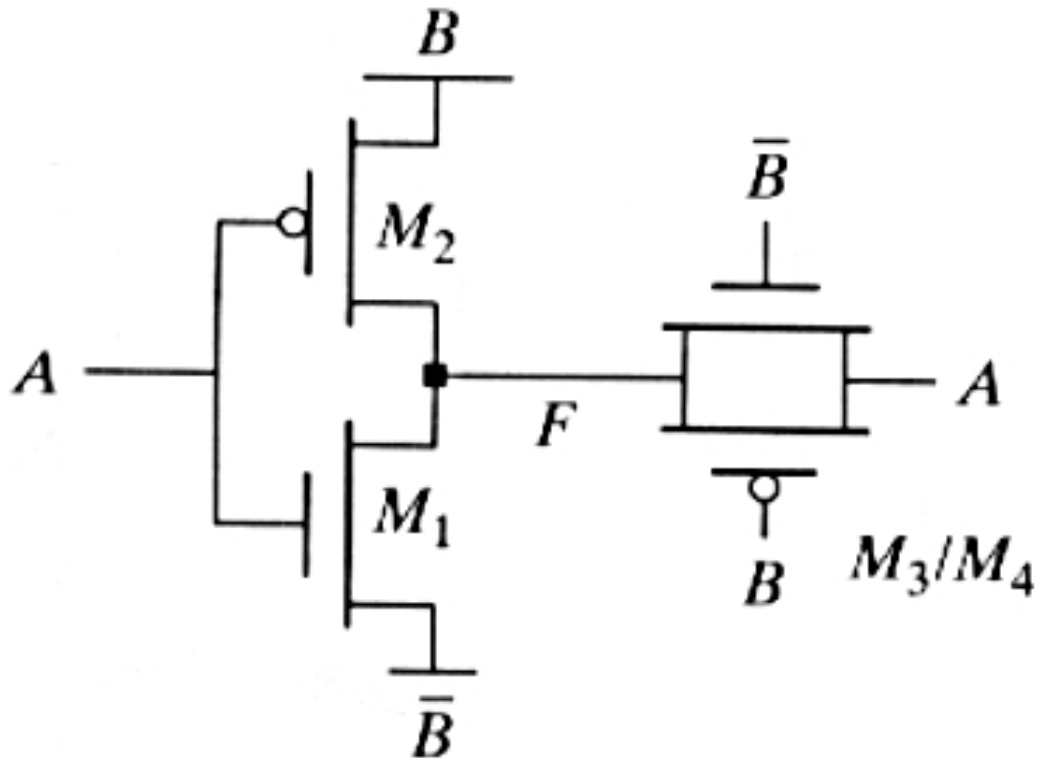


- Full swing** *bidirectional* switch controlled by the gate signal C , $A = B$ if $C = 1$

Transmission gate 2-input inverting multiplexer



Transmission gate XOR



$$F = A'B + AB'$$

CPL Properties

- ❑ **Differential** so complementary data inputs and outputs are always available (so don't need extra inverters)
- ❑ Still static, since the output defining nodes are always tied to V_{DD} or GND through a low resistance path
- ❑ Design is **modular**; all gates use the same topology, only the inputs are permuted.
- ❑ Simple XOR makes it attractive for structures like **adders**
- ❑ Fast (assuming number of transistors in series is small)
- ❑ Additional routing overhead for complementary signals
- ❑ Still have static power dissipation problems