ECE 6473
Lecture 5
Date: 10/13/2015

Shaloo Rakheja
Assistant Professor
Electrical and Computer Engineering, NYU

Reading

- Power point and hand-written lecture notes posted on newclasses.nyu.edu

- Sections 6.2.2 from Digital Integrated Circuits by Jan M. Rabaey et al.

Homework # 3

- **Due on 10/19 without any exceptions. If the server is down, please email your homework to the TAs.**

Exam on 11/02

Content:

- Everything covered up to 10/26/2015
- 2-page cheat sheet versus open book ?

Combinational logic styles

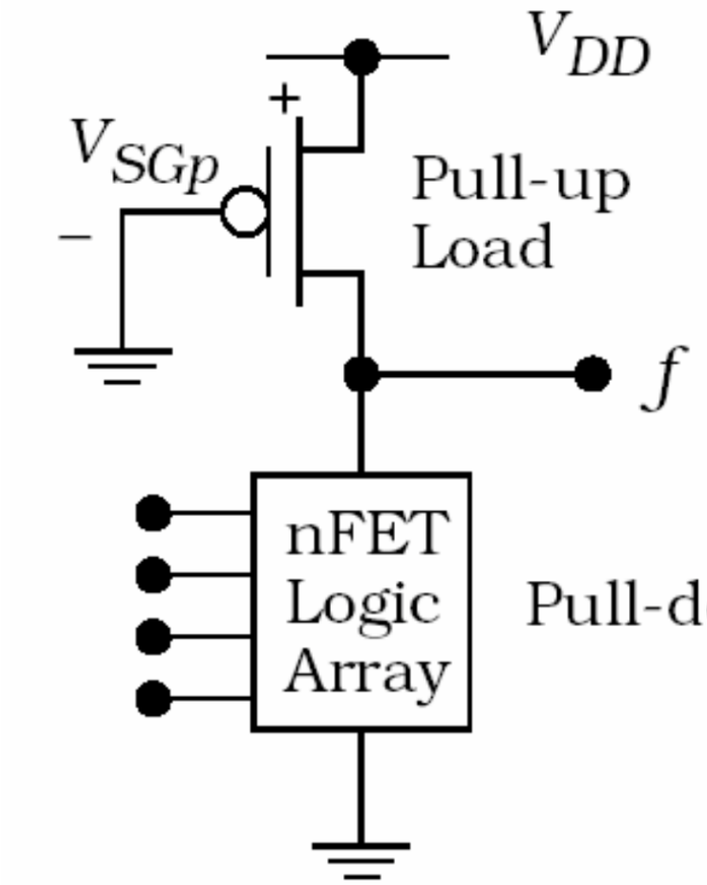
1. Pseudo-NMOS
2. Pass-transistor
3. Transmission gate
4. Dynamic logic

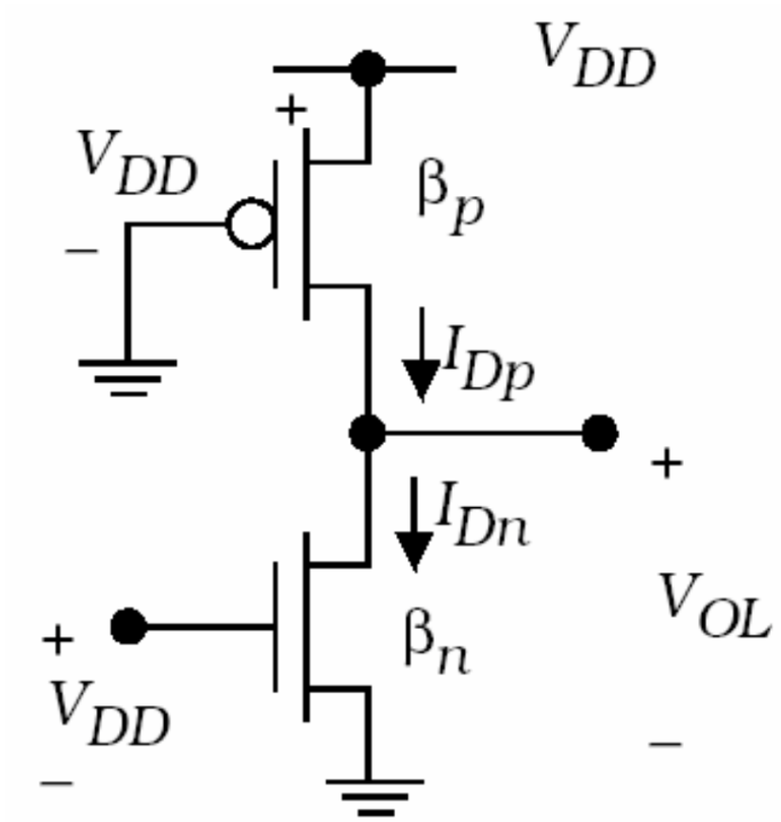
Power Dissipation

Static CMOS circuit: issues

- **Two devices (one NMOS and one PMOS) is required to implement each variables in a Boolean function**
  - To implement a function in with N variables 2N devices are required
  - Larger area

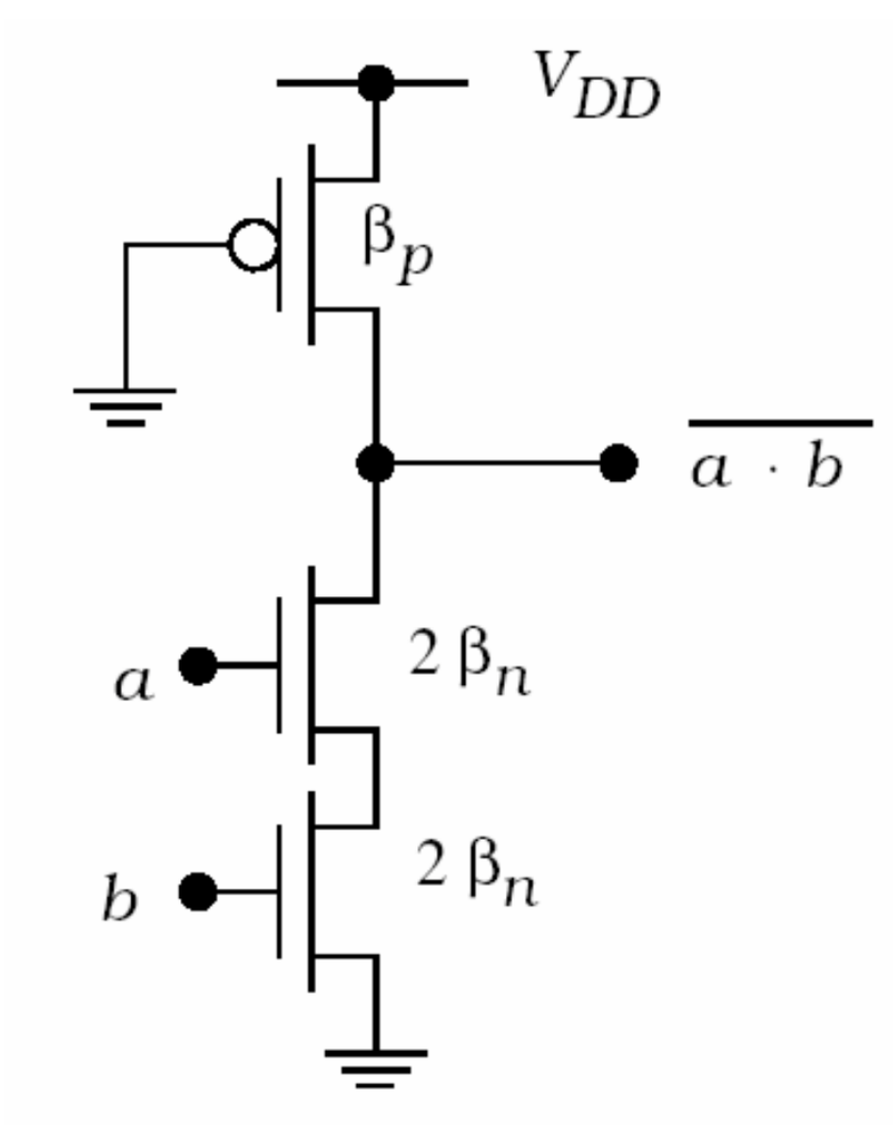
Ratioed logic



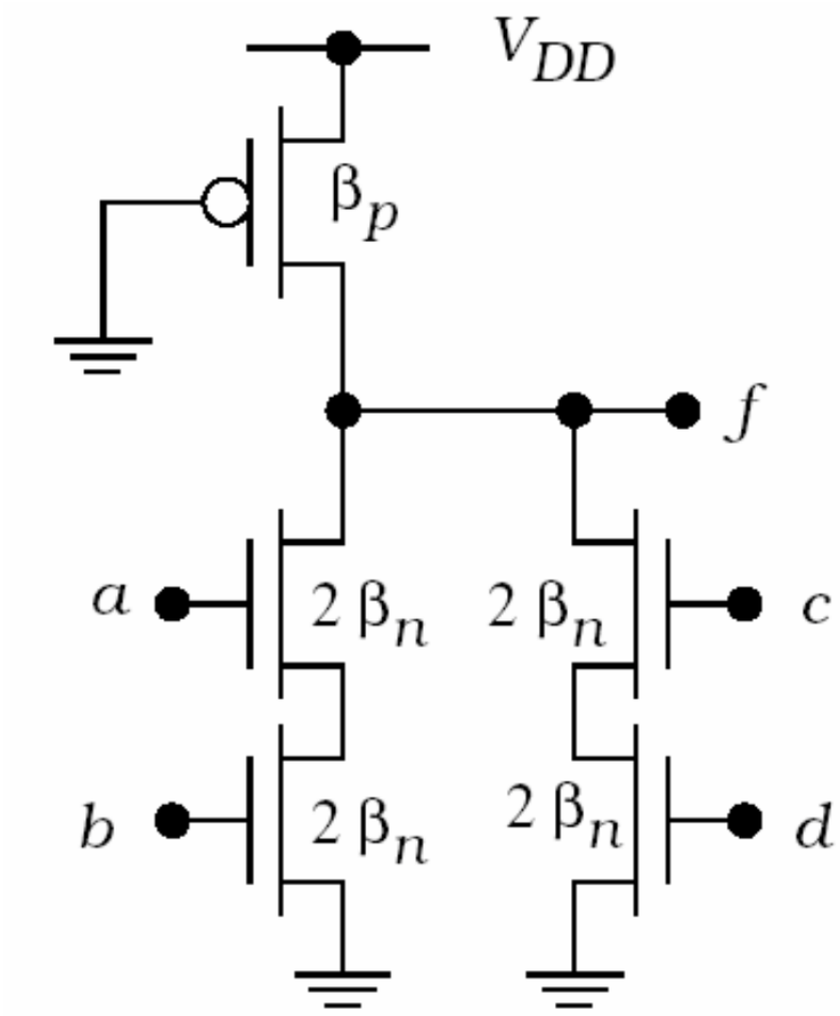


- **Less number of transistors**
- **V_{OL} > 0 and depends on the ratio of PFET to NFET size**

Ratioed logic



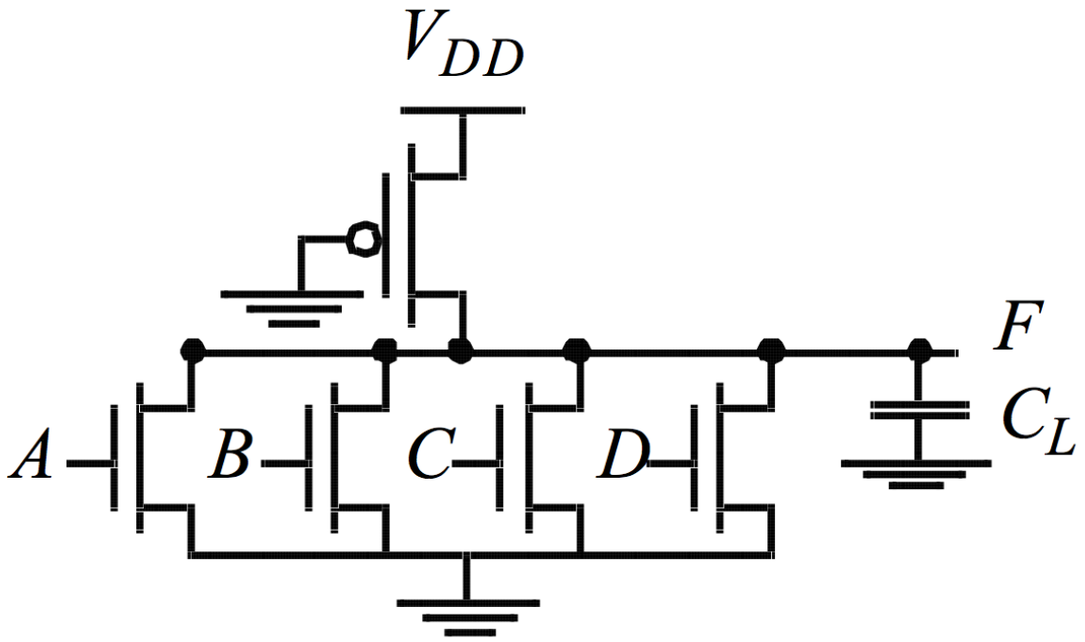
NAND Gate



AND-OR-INVERT

Pseudo-NMOS logic

$V_{OH} = V_{DD}$ as in case of CMOS Logic

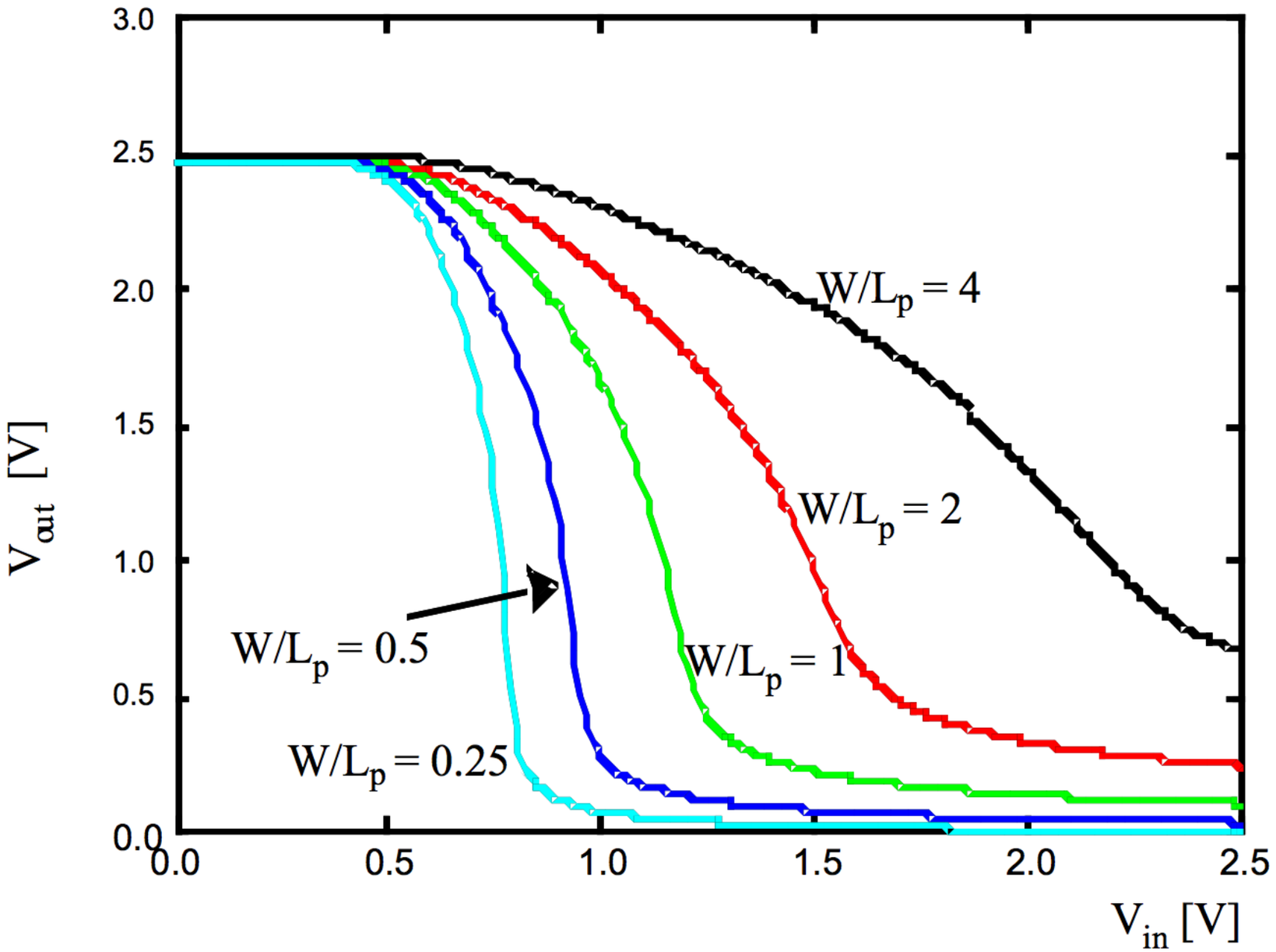


$$0.5\beta_p\left(V_{DD}-\left|V_{Tp}\right|\right)^2=\beta_n\left(V_{DD}-V_{Tn}-0.5V_{OL}\right)V_{OL}$$

For small V_{OL}

$$V_{OL}=0.5\left(\frac{W_p}{W_n}\frac{\mu_p}{\mu_n}\right)\left(\frac{\left(V_{DD}-\left|V_{Tp}\right|\right)^2}{V_{DD}-V_{Tn}}\right)$$

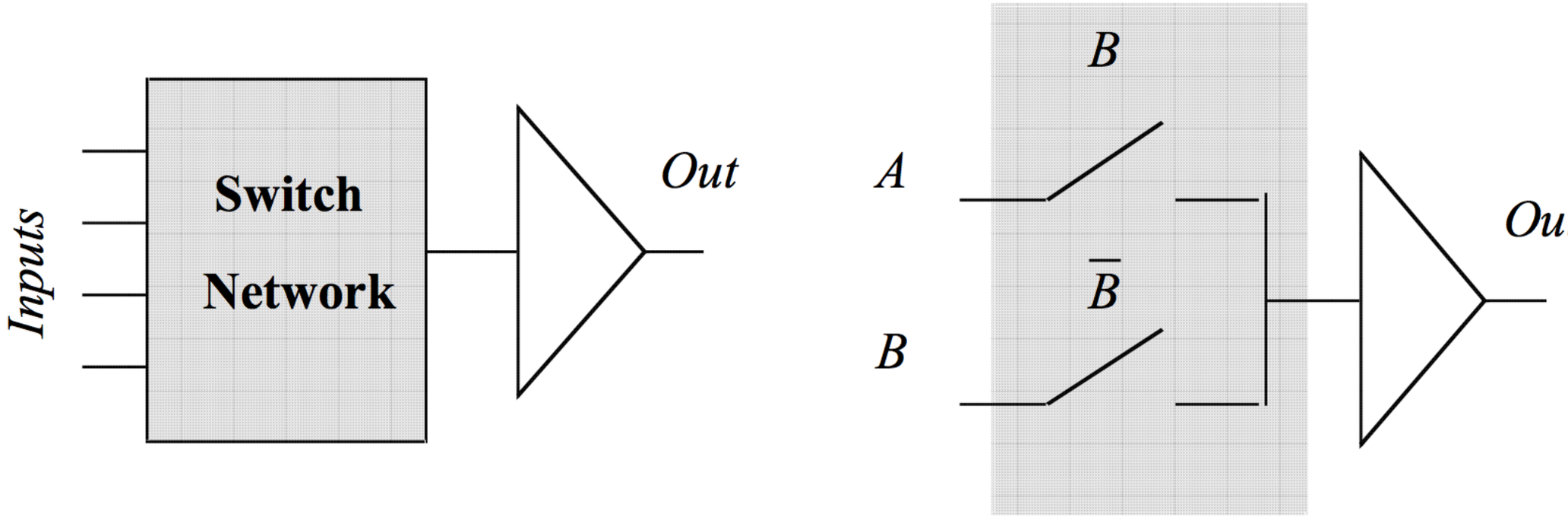
Pseudo-NMOS voltage transfer characteristic



Summary: pseudo-NMOS logic

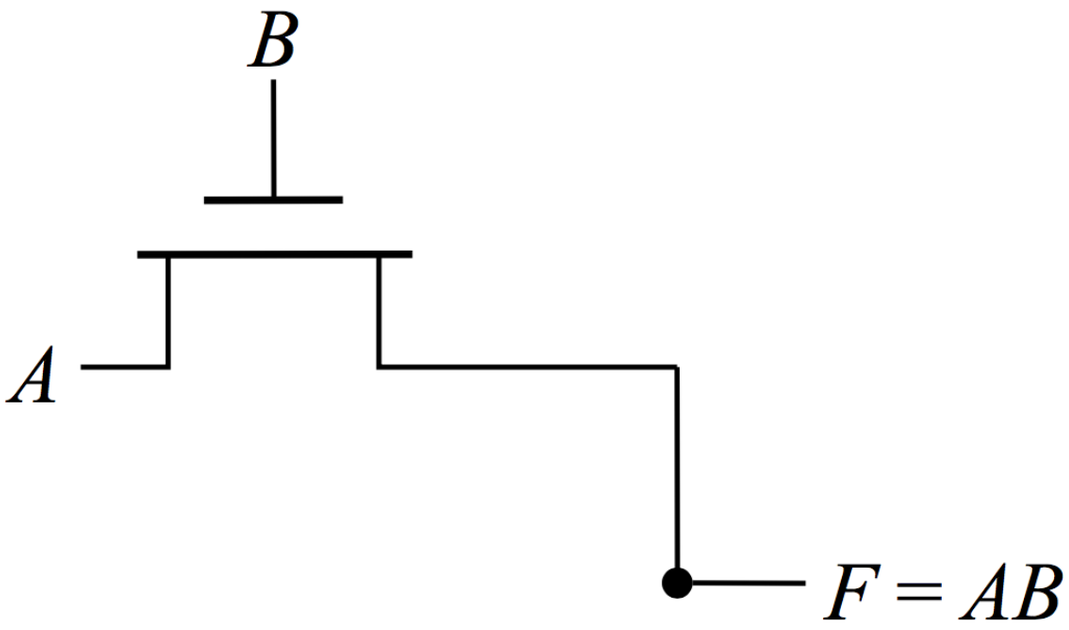
- For n-inputs, (n+1) transistors are needed —> n NMOS transistors and 1 PMOS transistor —> **area advantage.**
- It is inverting logic (just like CMOS).
- $V_{OH} = V_{DD}$ but $V_{OL} \neq 0$.
- **PMOS is always ON. Therefore, static power dissipation.**
- If (Wp/Wn) increases —> V_{OL} and V_M increase.

Pass-transistor logic

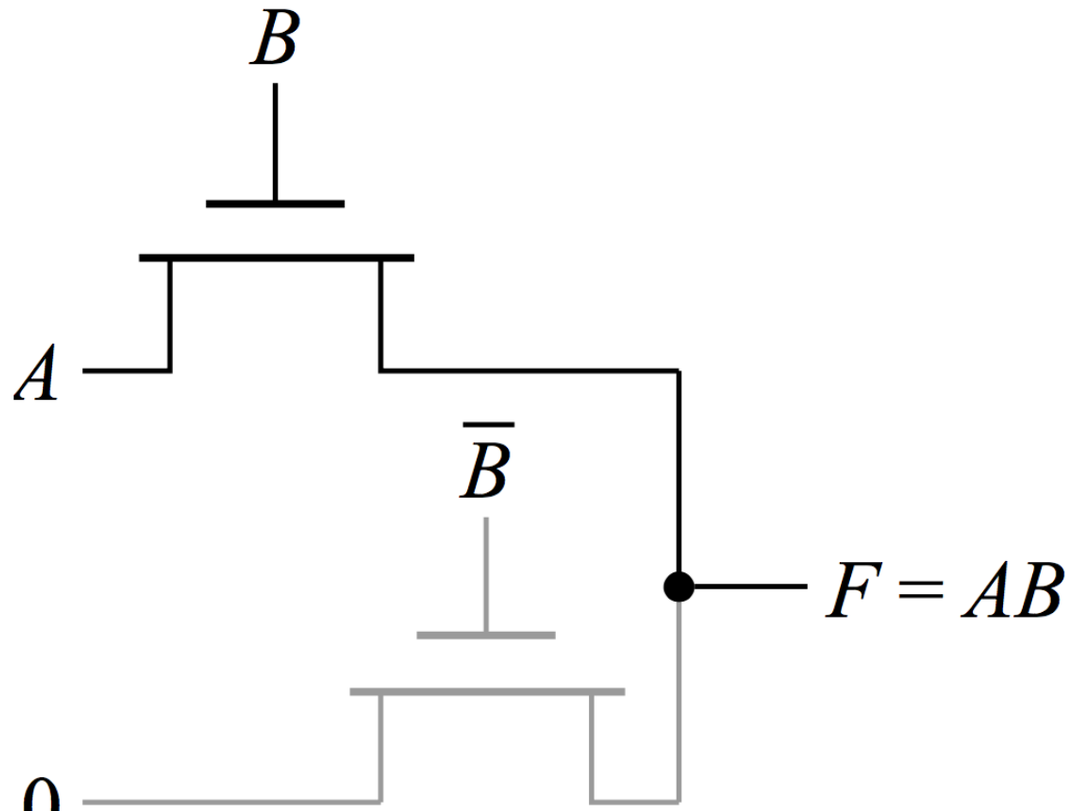


- **N transistors**
- **No static consumption**

Pass-transistor logic: AND gate



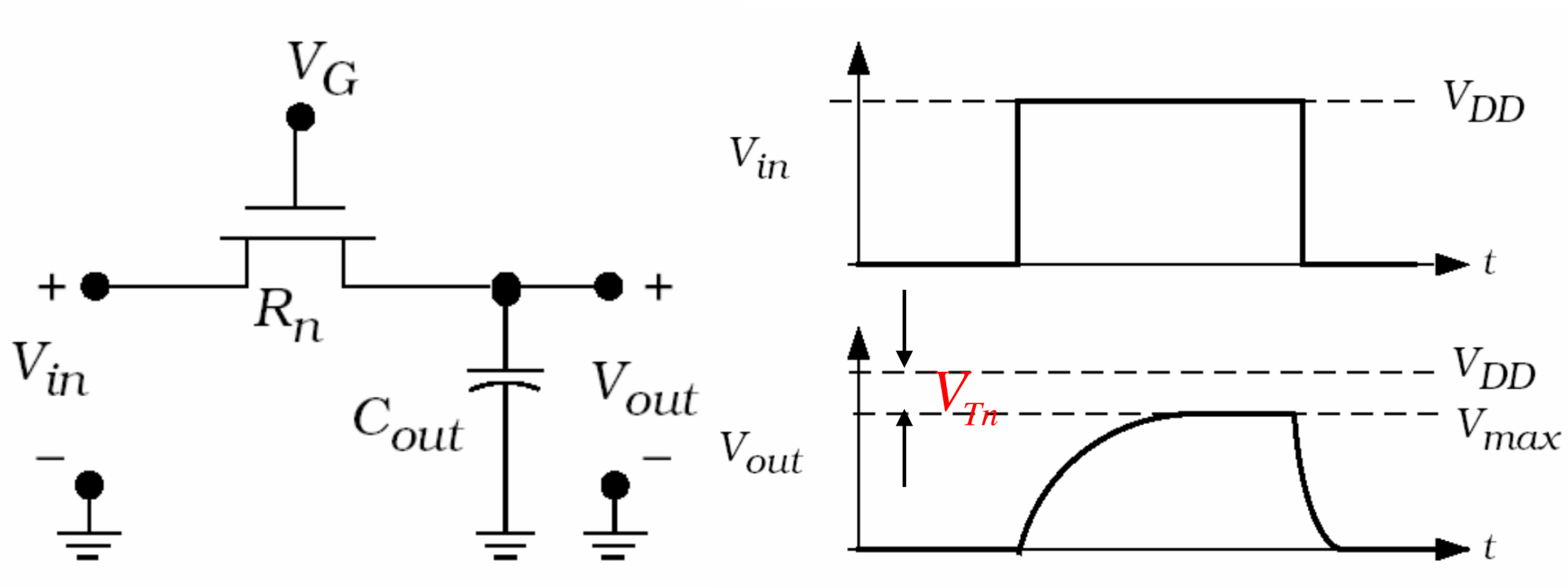
F is floating when B=0



F is 0 when B=0

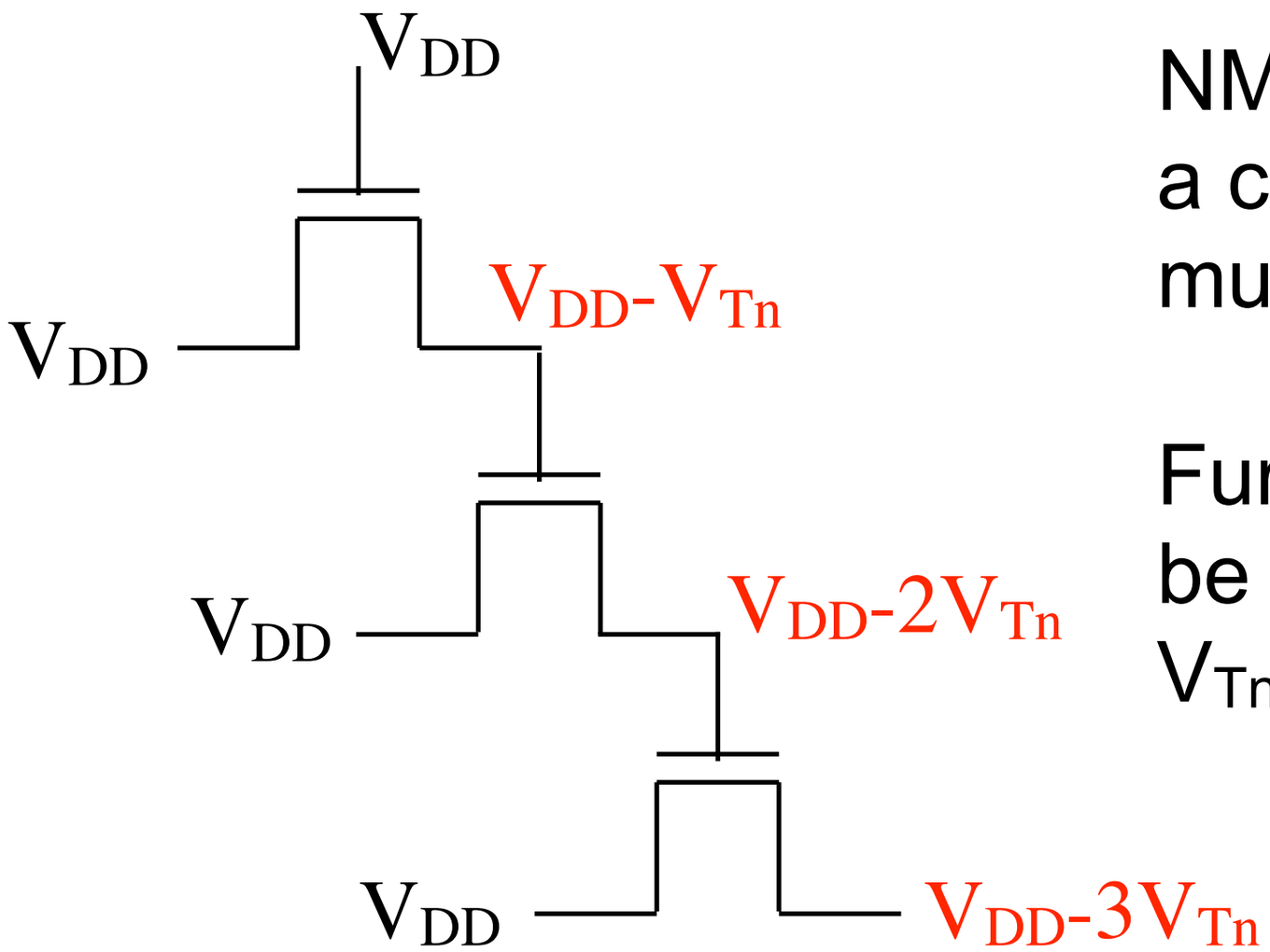
Note that this logic is non-inverting.

NMOS pass transistor logic output



Note the output cannot go all the way up to VDD because of threshold voltage drop on the NMOS.

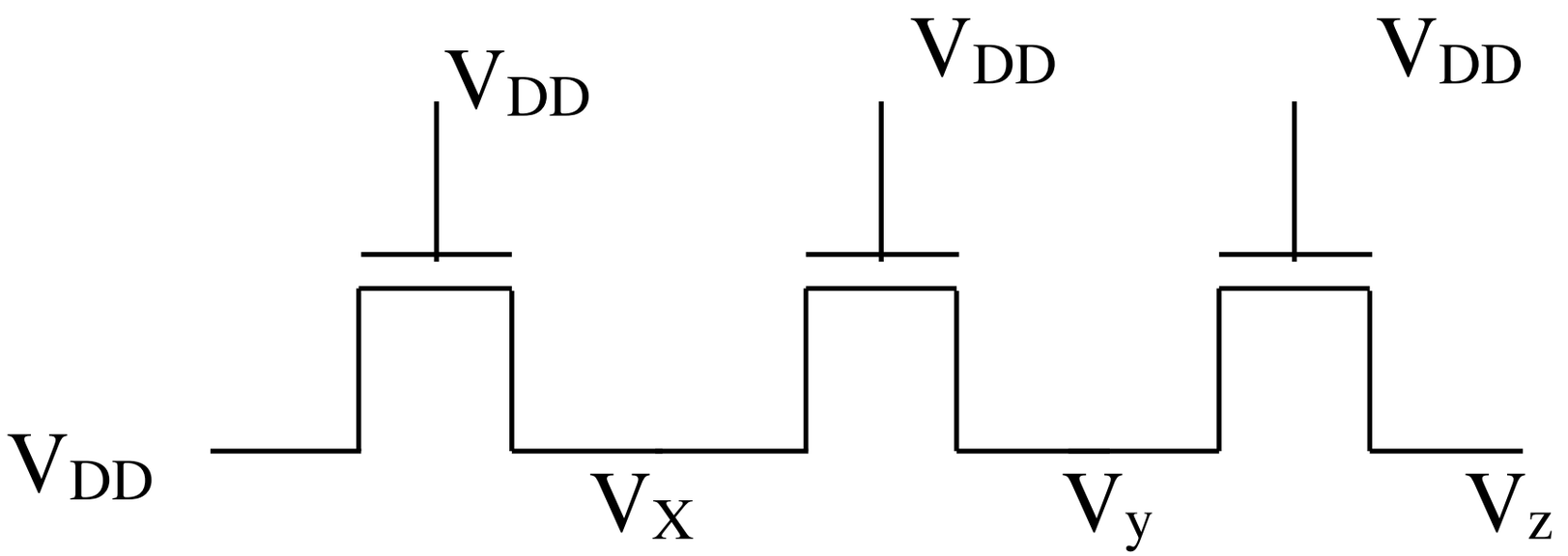
NMOS pass transistor logic output



NMOS pass transistors in a cascade leads to multiple V_{Tn} drops.

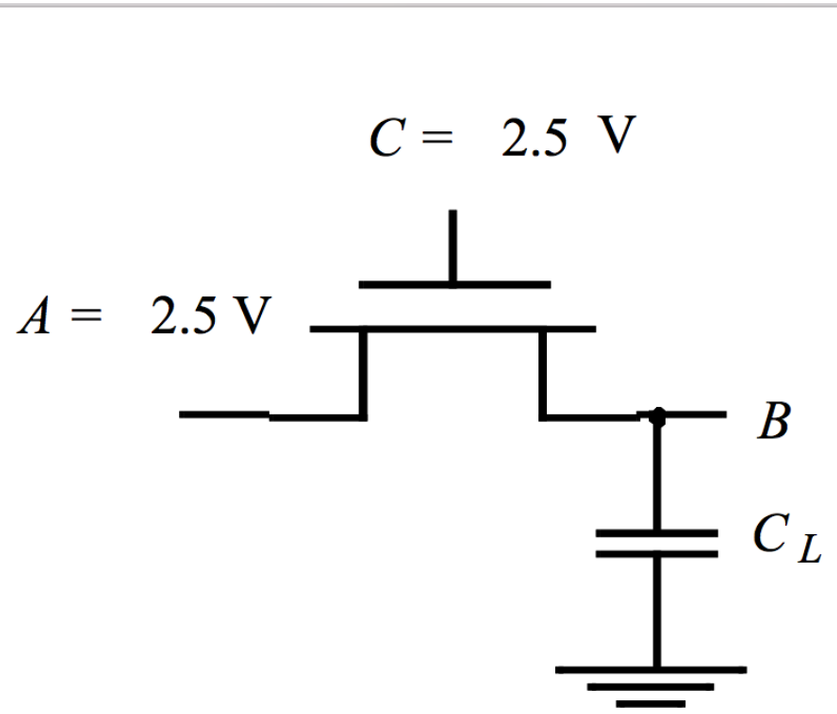
Further, each NMOS will be under body effect, so V_{Tn} will be high.

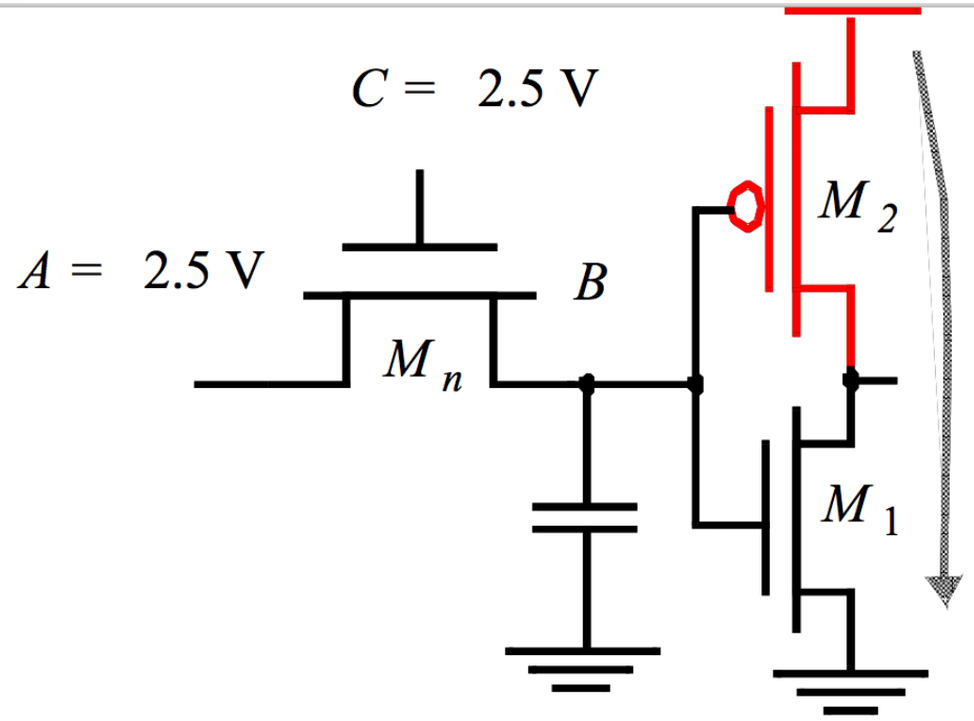
NMOS pass transistor logic output



What is Vx, Vy, Vz ?

NMOS pass transistor logic: static power consumption



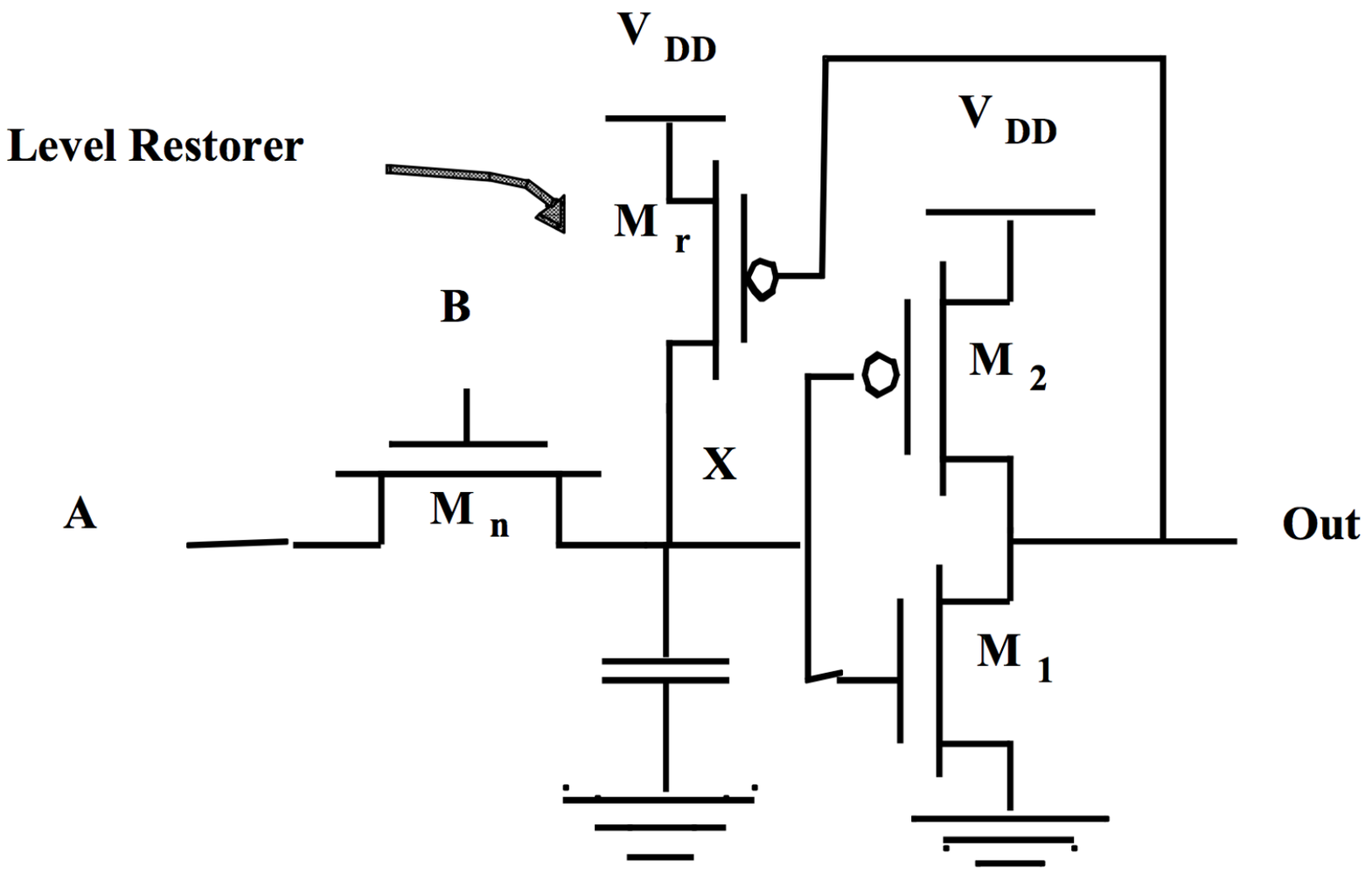


V_B does not pull up to 2.5V, but 2.5V -V_{TN}

Threshold voltage loss causes static power consumption

NMOS has higher threshold than PMOS (body effect)

NMOS pass transistor logic: **level restorer**

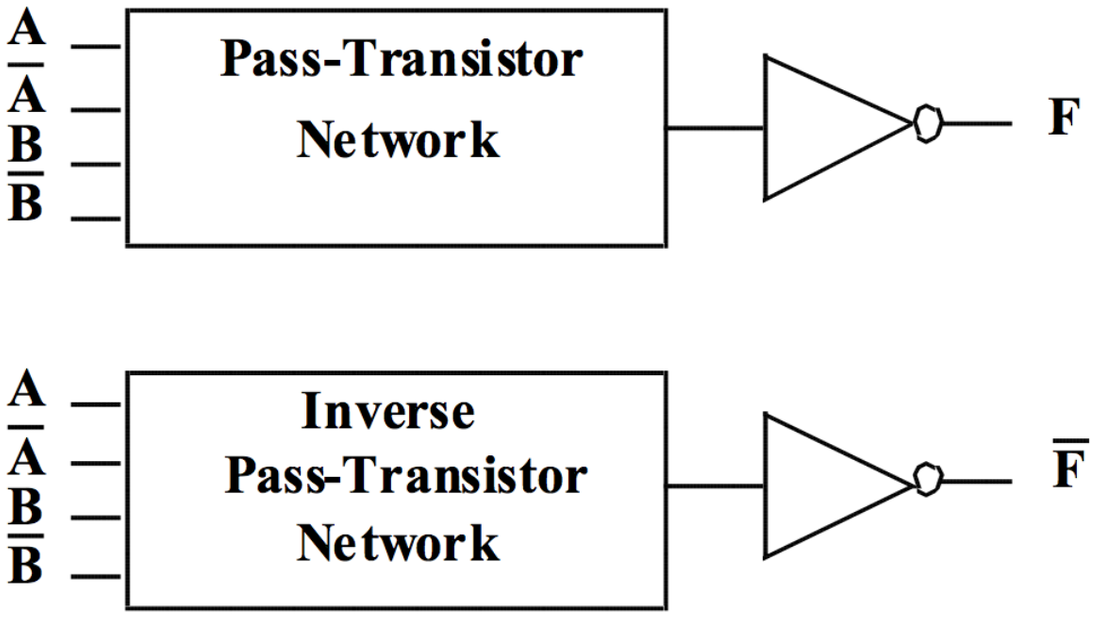


- **Advantage: Full Swing**
- **Restorer adds capacitance, takes away pull down current at X**
- **Ratio problem**

Summary: NMOS only switch

- Pass transistor logic uses fewer transistors to implement logic.
- NMOS only switch always has a V_{Tn} drop.
- The implementation is **NON-inverting.**
- Avoid feeding the output of NMOS only switch as a gate input to another NMOS only switch. This leads to multiple V_{Tn} drops across the cascade.
- **Static power consumption occurs in the fan-out stage.**
- **Dynamic power dissipation will be reduced due to the lower voltage swing at the output node.**

Complimentary pass transistor logic



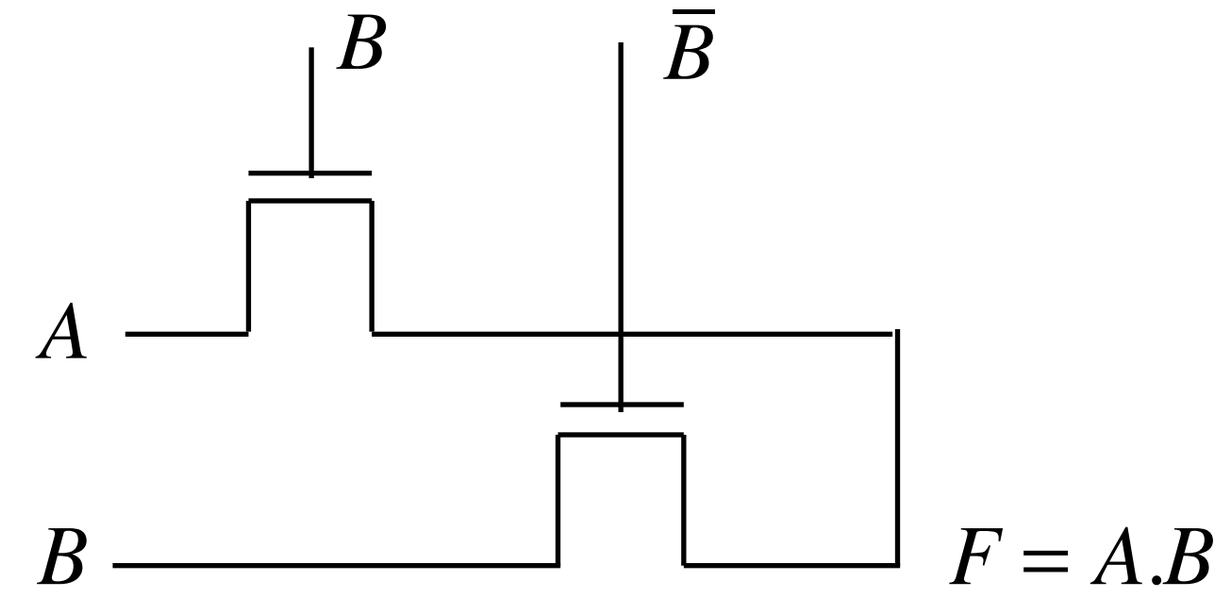
When F is '0' we need to connect it to a 'voltage source' to avoid it from floating.
It still passes a weak '1'

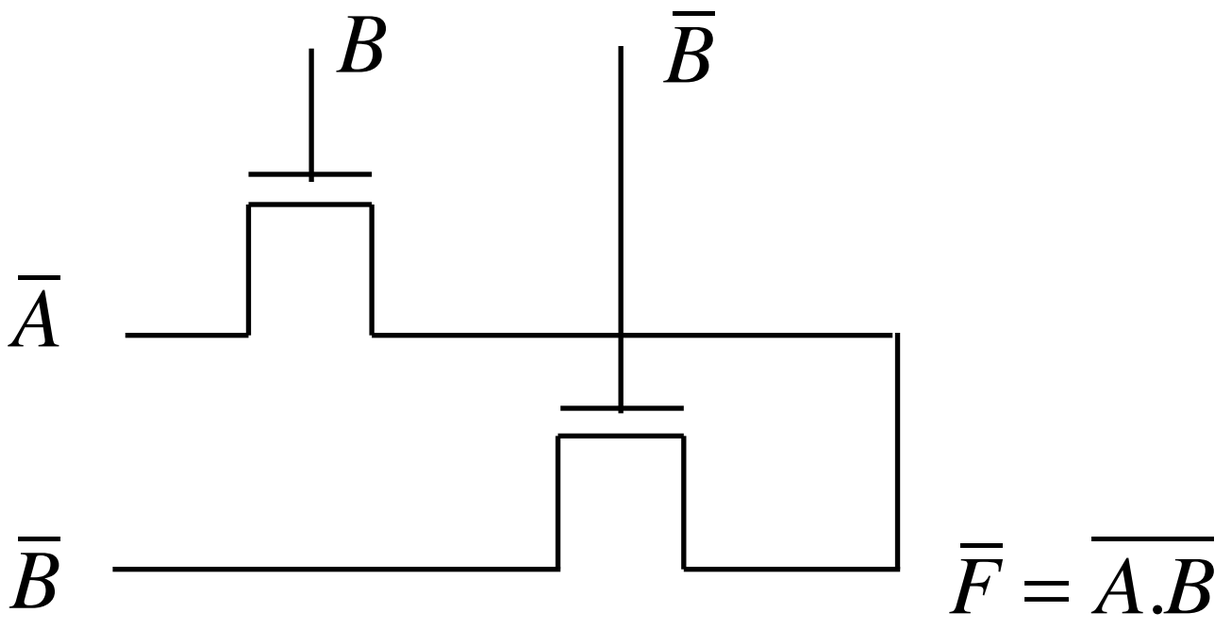
Circuits are differential, complementary inputs and outputs are always available.

Static gate, since the output nodes are always connected to ground or VDD —> better noise resilience.

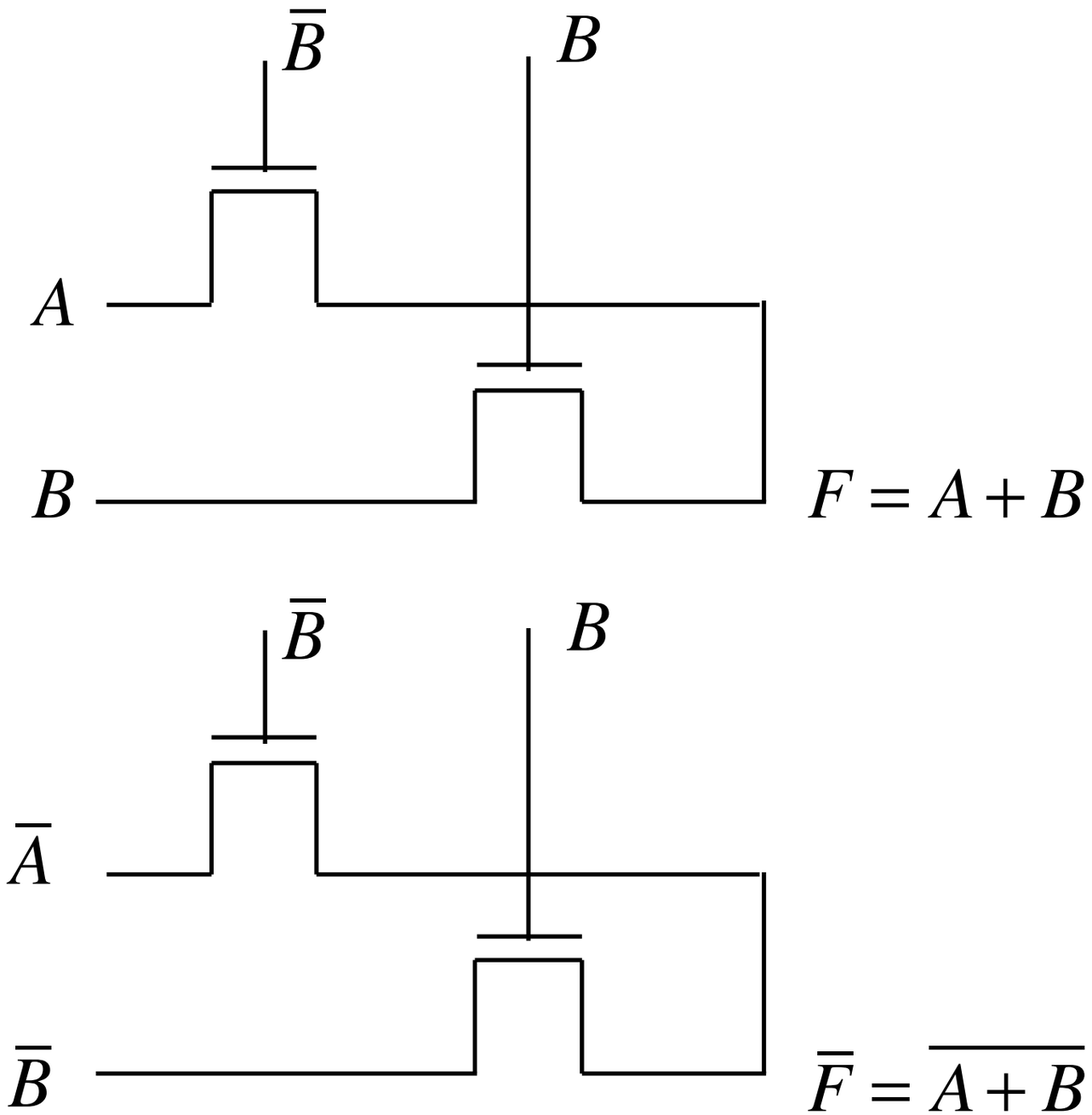
Design is very modular —> All gates use the same topology

Complimentary pass transistor logic: AND/NAND



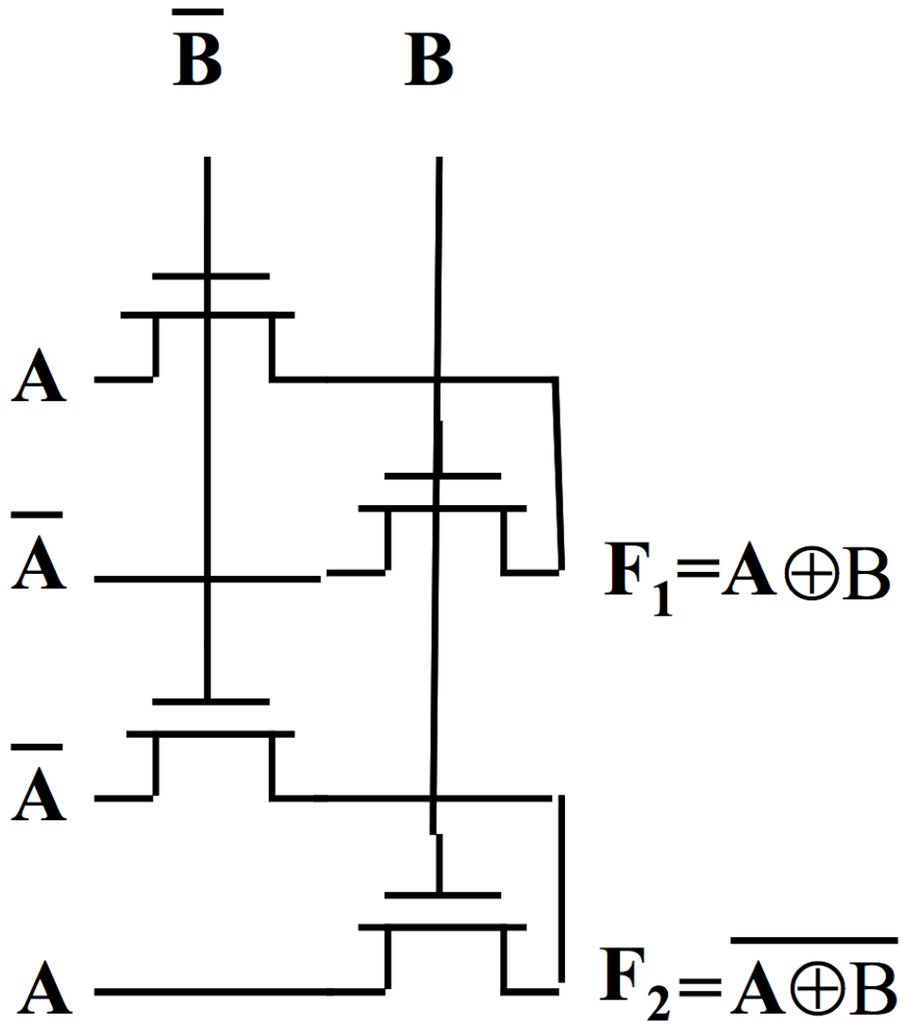


Complimentary pass transistor logic: OR/NOR



Complimentary pass transistor logic: XOR/XNOR

How many transistors does a CMOS implementation of XOR gate require?



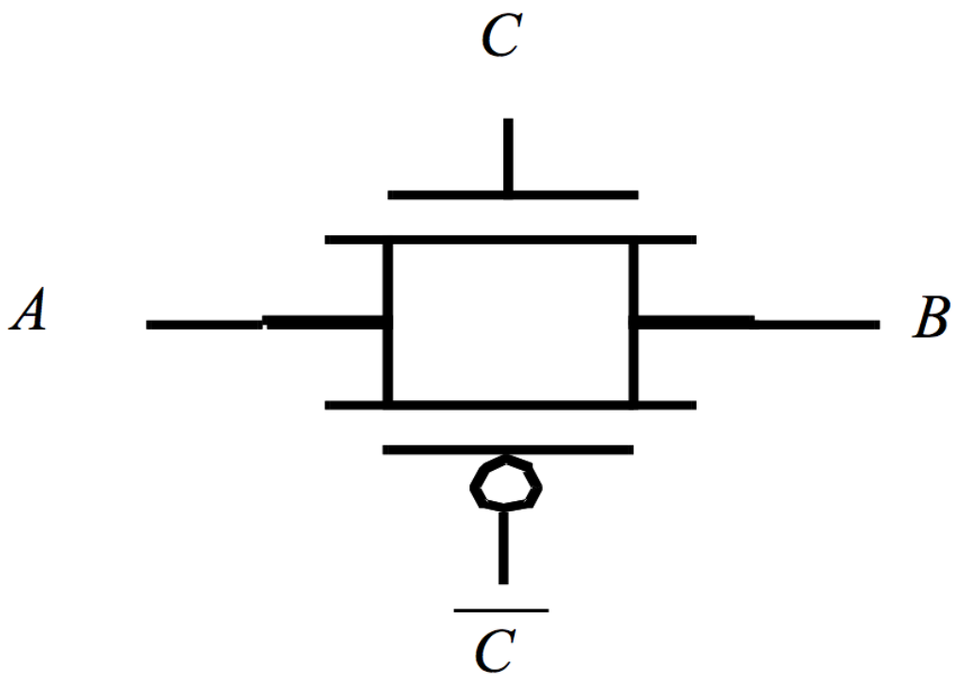
XOR/XNOR

Complimentary pass transistor logic

- Complimentary pass transistor logic is modular in design.
- Resilient to noise as the outputs are always connected to VDD or Ground.
- Both inverting and non-inverting outputs are available simultaneously.
- **However, there is still the problem of threshold voltage drop.**
- **Static power consumption occurs in the fan-out stage.**

Transmission gate logic overcomes the threshold voltage drop problem.

Transmission Gate

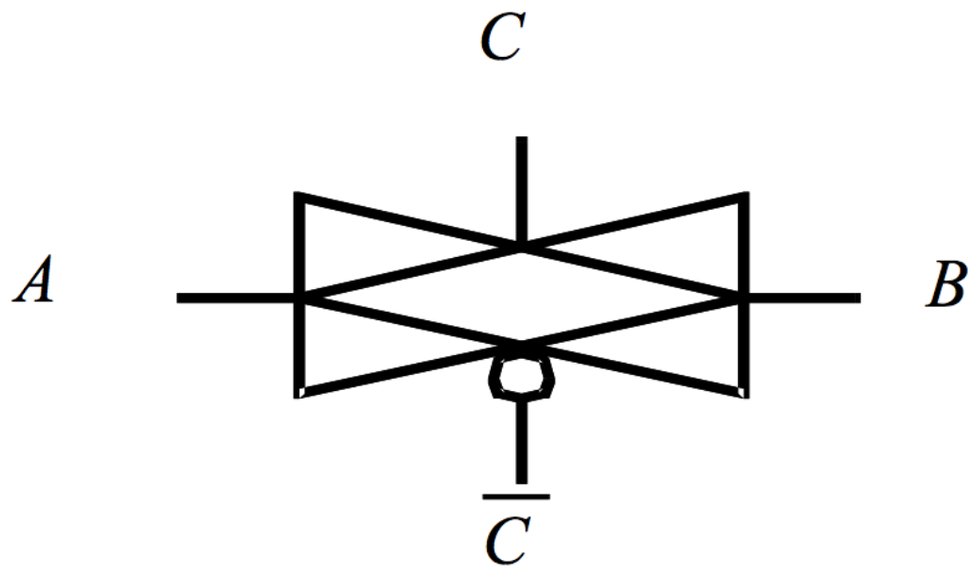


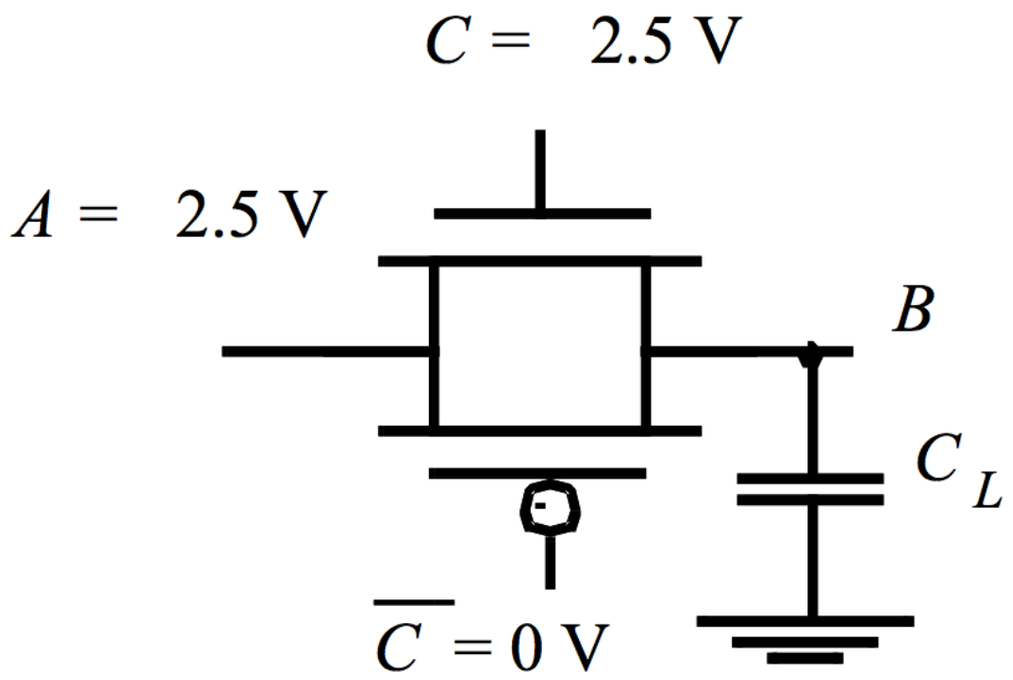
PMOS passes A=1
NMOS passes A=0

Place NMOS in parallel with PMOS.

Bidirectional switch controlled by the signal C.

Transmission Gate





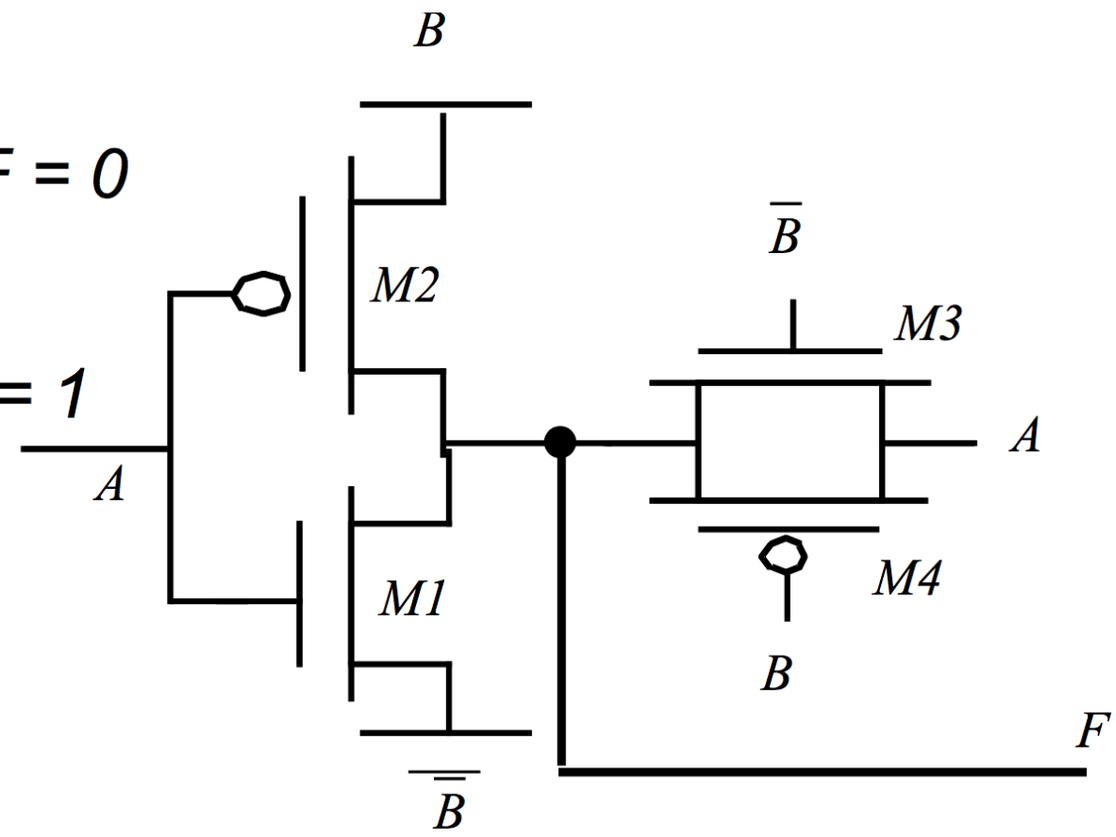
Transmission Gate (TG): XOR implementation

A = 0, B=0
=> M3/M4 ON => F= A => F = 0

A = 1, B=0
⇒M3/M4 ON => F= A => F = 1

A = 0, B=1
⇒M2/M1 acts as inverter
⇒ F= $\overline{A}$ => F = 1

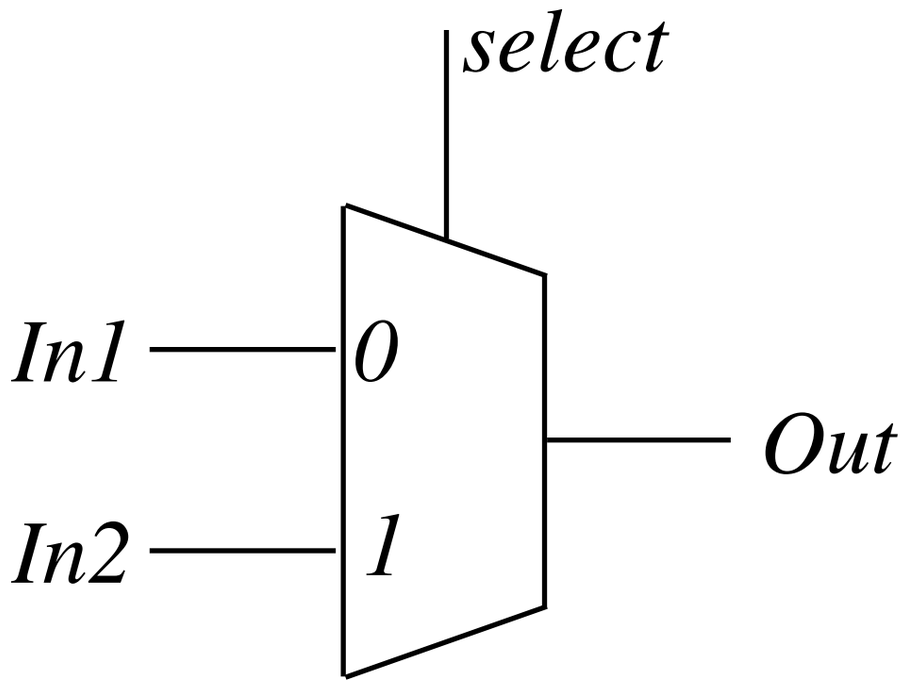
A = 1, B=1
⇒M2/M1 acts as inverter
⇒ F= $\overline{A}$ => F = 0



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

Transmission Gate (TG) is particularly attractive for multiplexing inputs

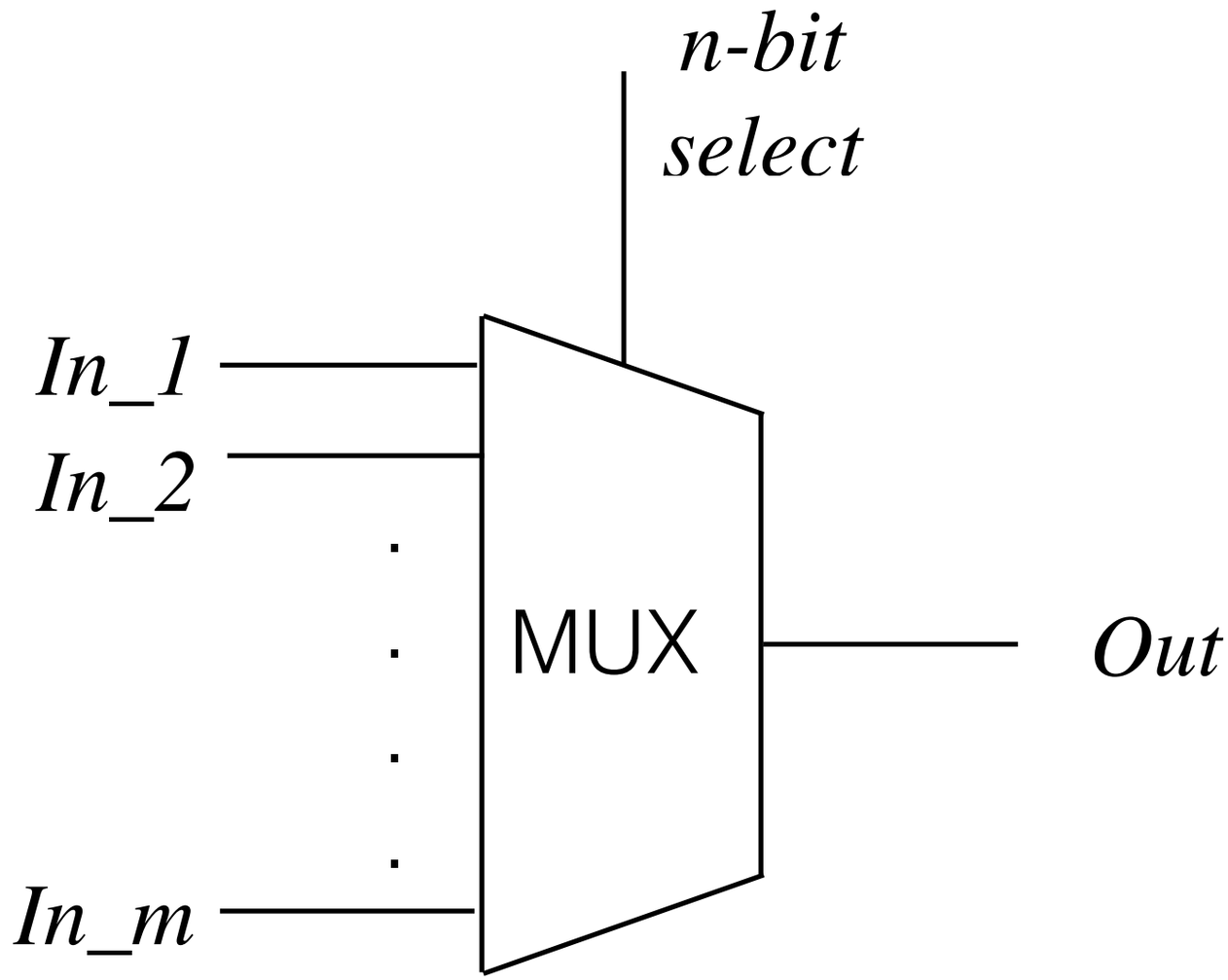
Multiplexer/Multiplexor/MUX



Role of a MUX is to connect one of the inputs to the outputs using a select signal.

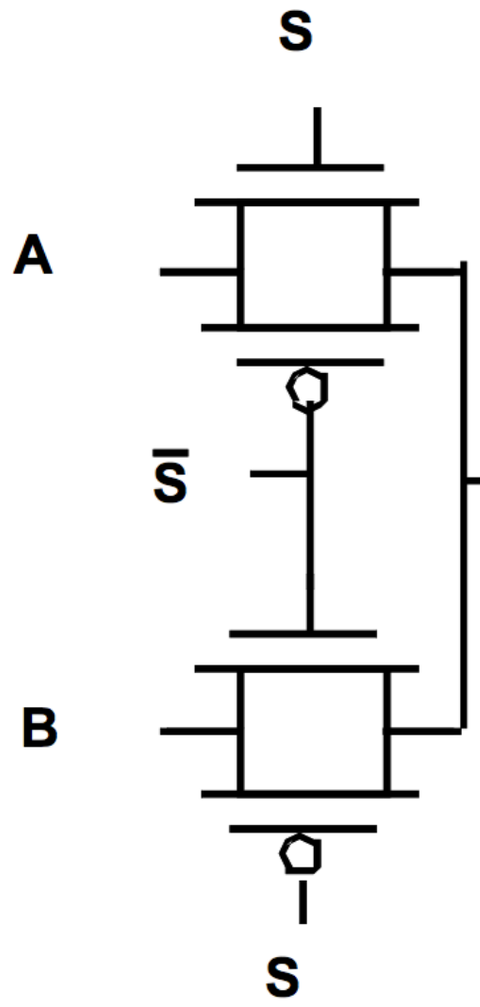
If select = 0, Out = In1
If select = 1, Out = In2

Multiplexer/Multiplexor/MUX



$$For\ m\ signals\ to\ be\ MUXED, n-bit\ select\ signal\ is\ required$$
$$n = \log_2(m)$$

TG- MULTIPLEXER (MUX)

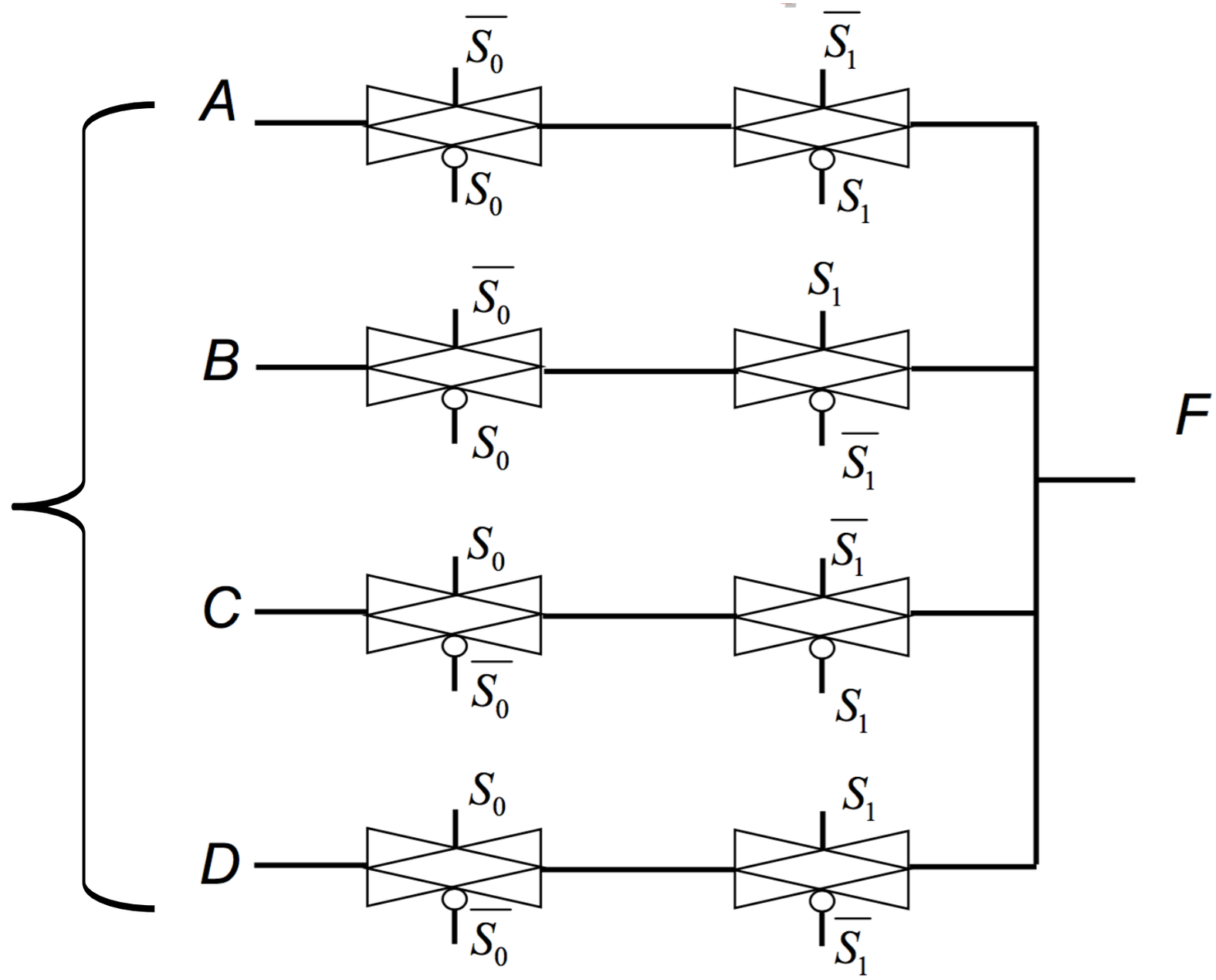


$$F = A.S + B.\overline{S}$$

In this example:
if S = 1, F = A
if S = 0, F = B

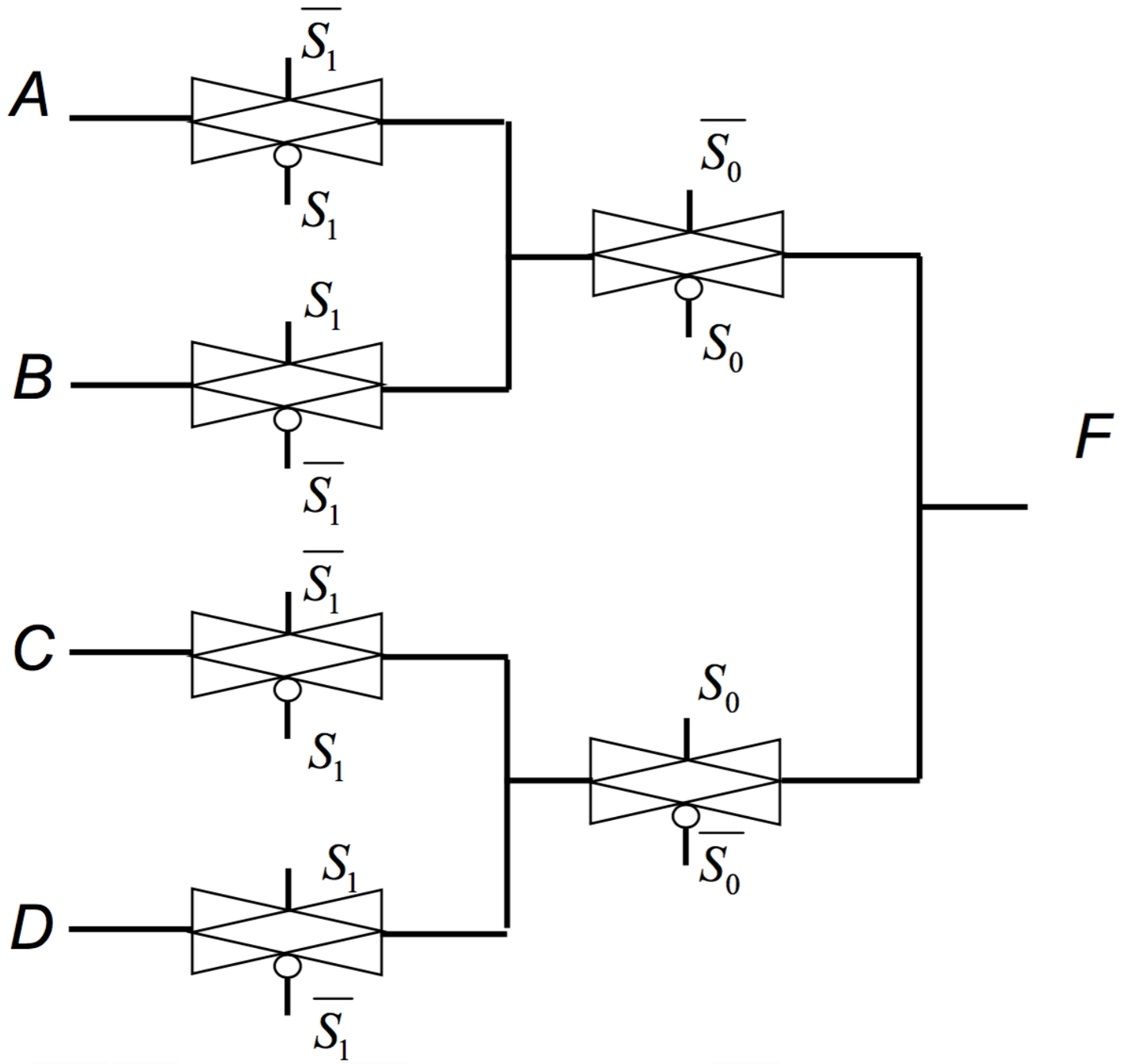
TG-MUX (4 input signals)

For multiplexing FOUR signals (A, B, C, D) we will need TWO select signals: S0 and S1



$$F = A.\overline{S_0}\,\overline{S_1} + B.\overline{S_0}S_1 + C.S_0\overline{S_1} + D.S_0S_1$$

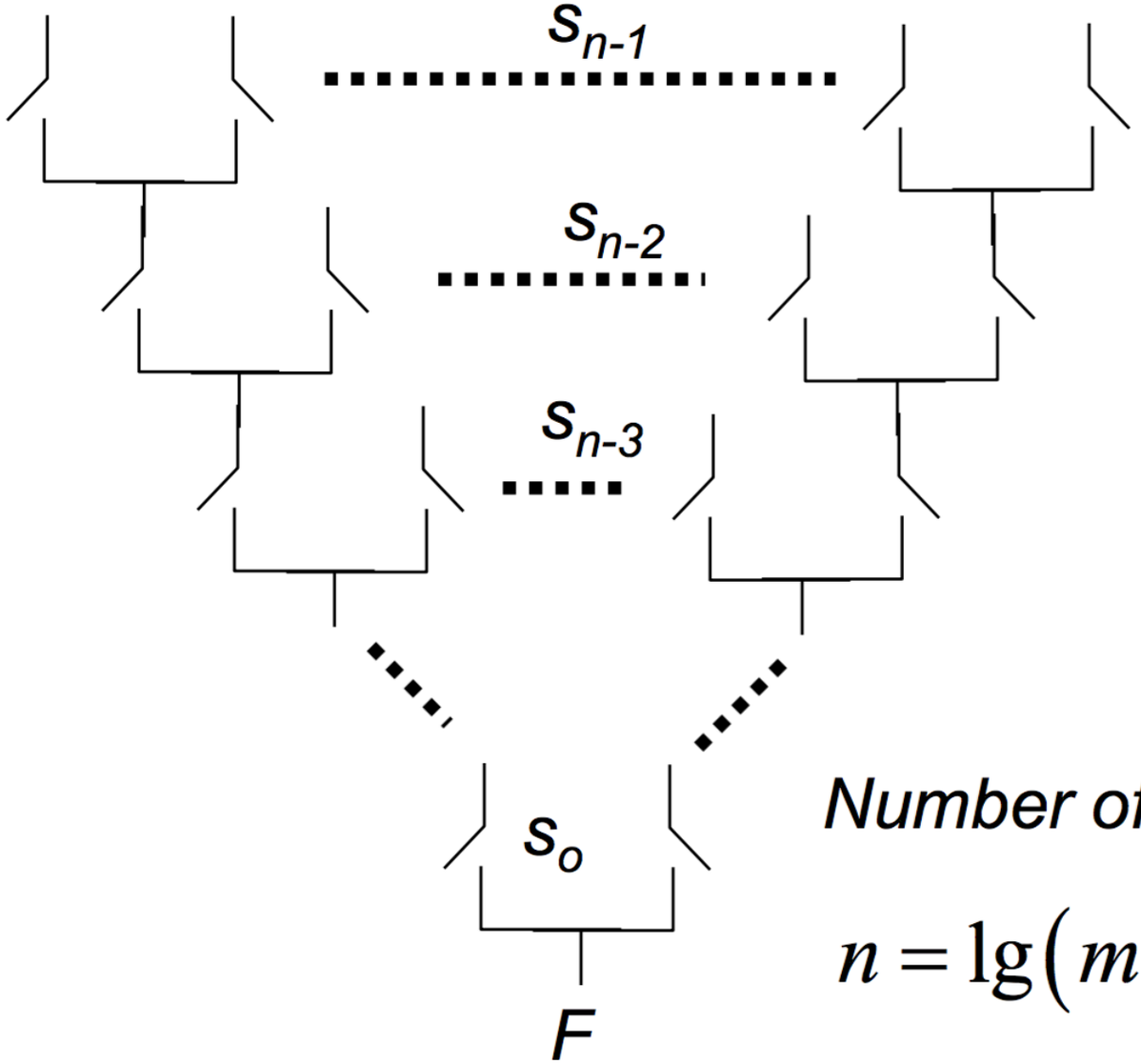
TG-MUX (4 input signals)



$$F = A.\overline{S_0}\,\overline{S_1} + B.\overline{S_0}S_1 + C.S_0\overline{S_1} + D.S_0S_1$$

Transmission Gate MUX (n-input signals)

$$F = A_1.\overline{S_0}\ldots\overline{S_{n-1}} + \ldots + A_{m-1}.S_0\ldots S_{n-1}$$



m = number of signals needs to be MUXed

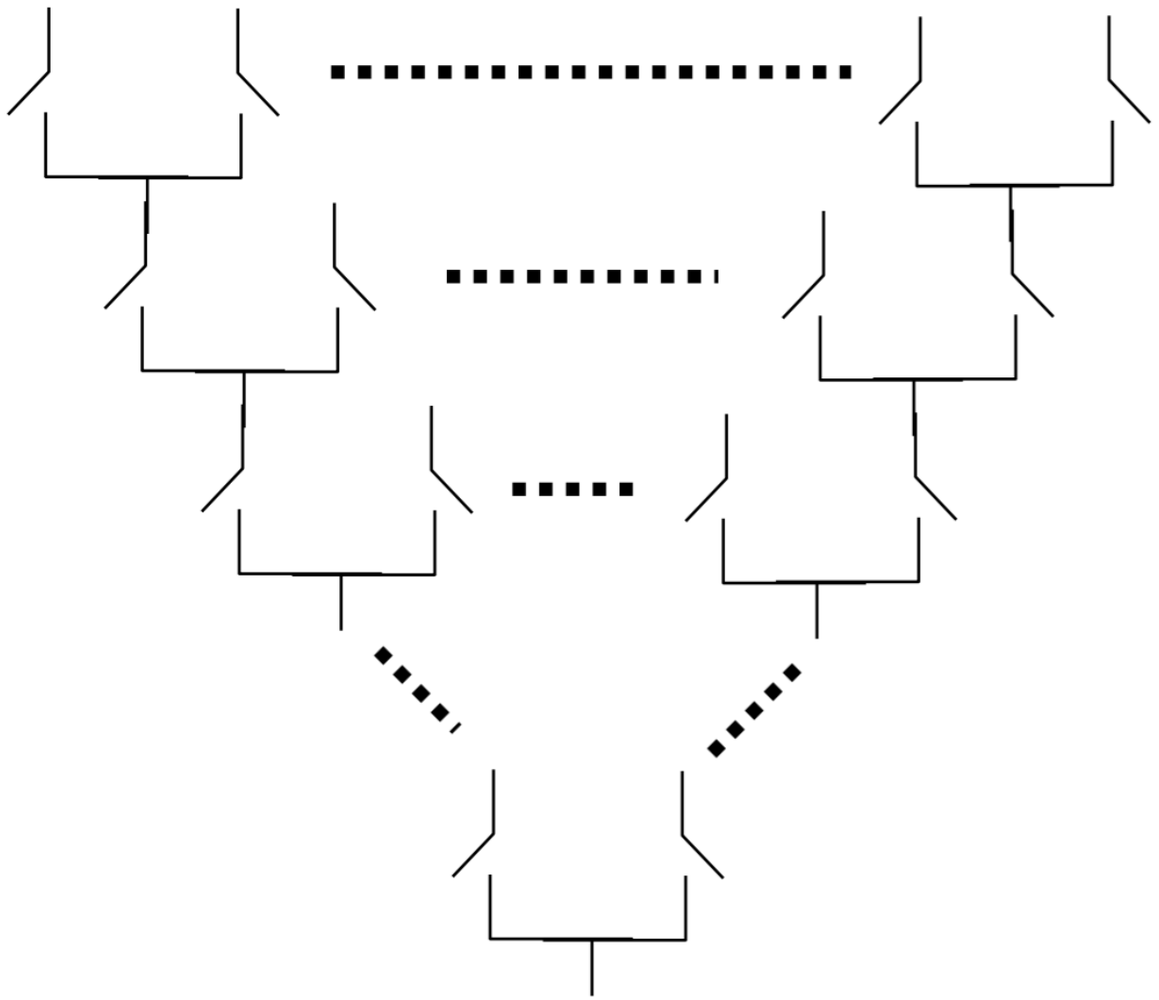
n = number of select signals

$$n = \lg(m) = \lceil \log_2(m) \rceil$$

Number of series connected TG

$$n = \lg(m) = \lceil \log_2(m) \rceil$$

Transmission Gate MUX propagation delay



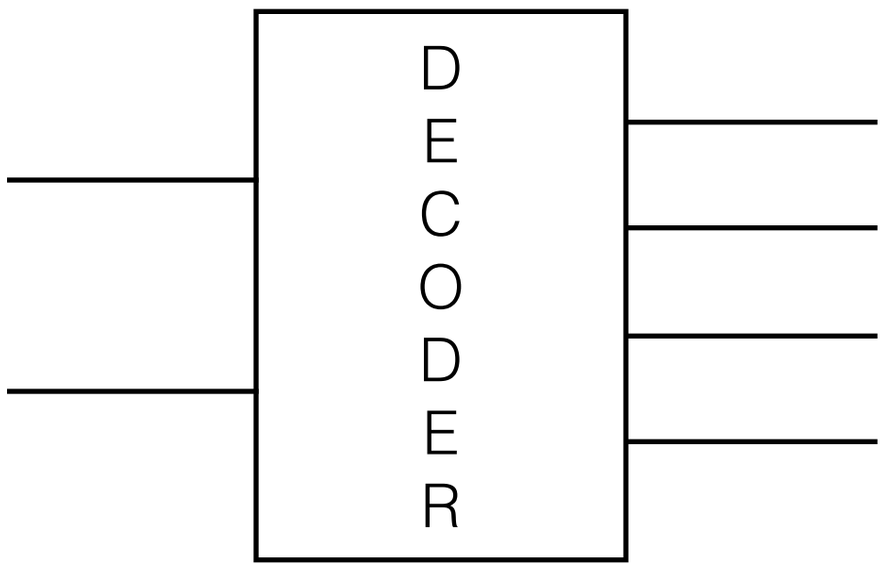
$$Propgation\ delay \propto (\#\ selcet\ signals)^2 \propto n^2$$

Transmission Gate MUX propagation delay

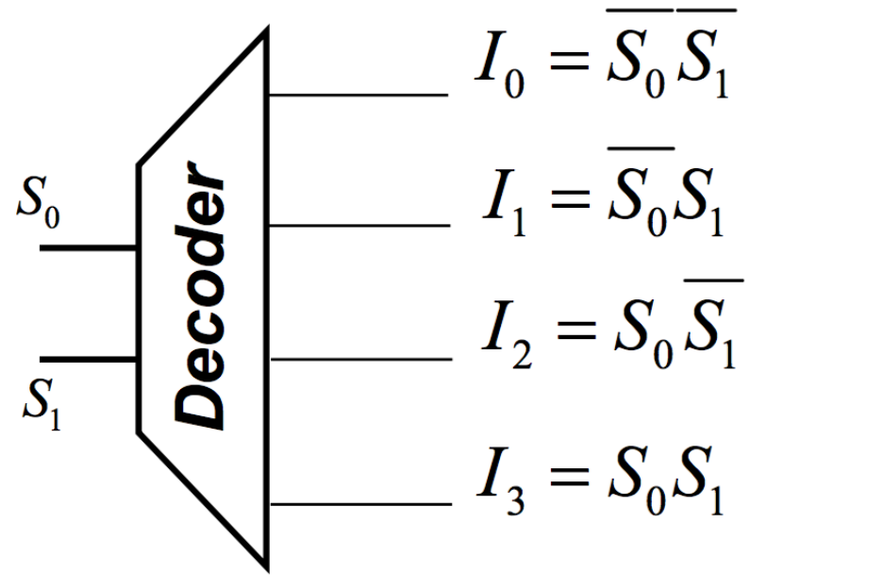
Is there another way of implementing MUX?

$$F = A.\overline{S}_0\overline{S}_1 + B.\overline{S}_0 S_1 + C.S_0\overline{S}_1 + D.S_0 S_1$$

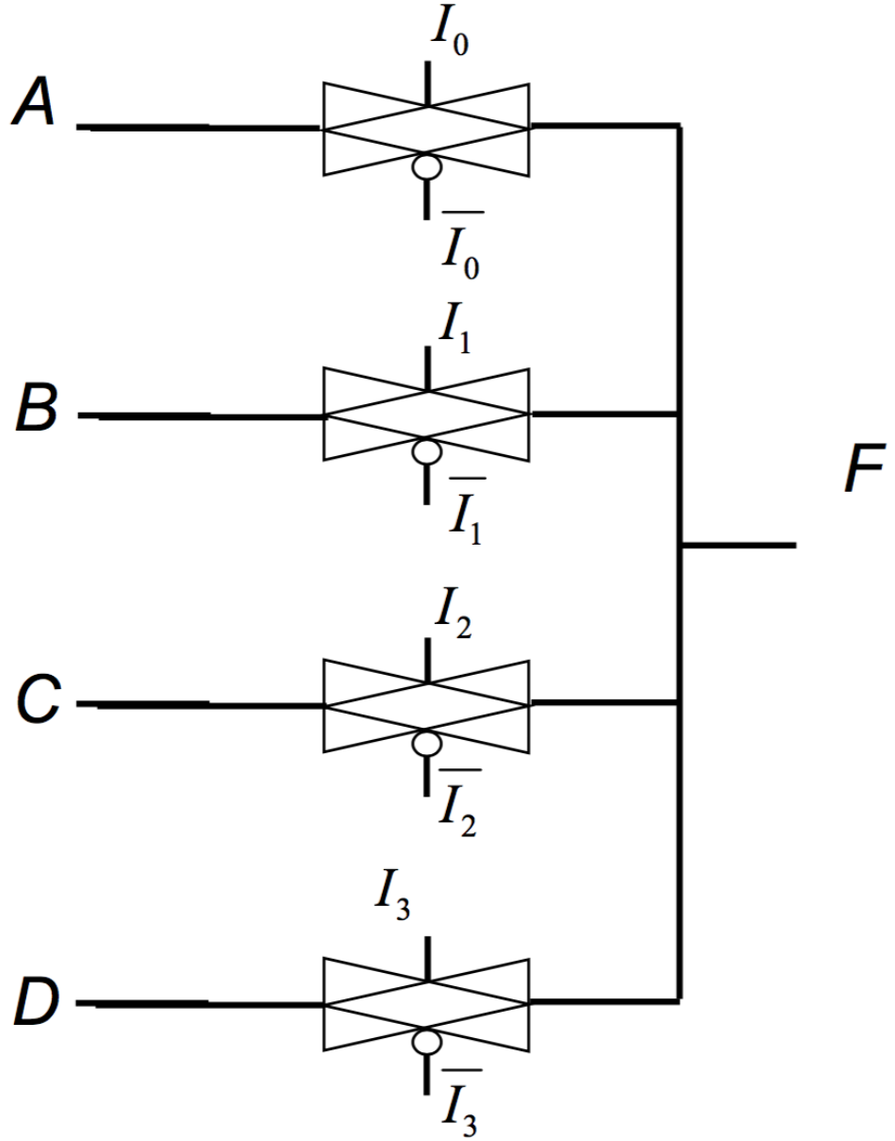
Pre-generate the four signals from a **DECODER** circuit



Transmission Gate MUX

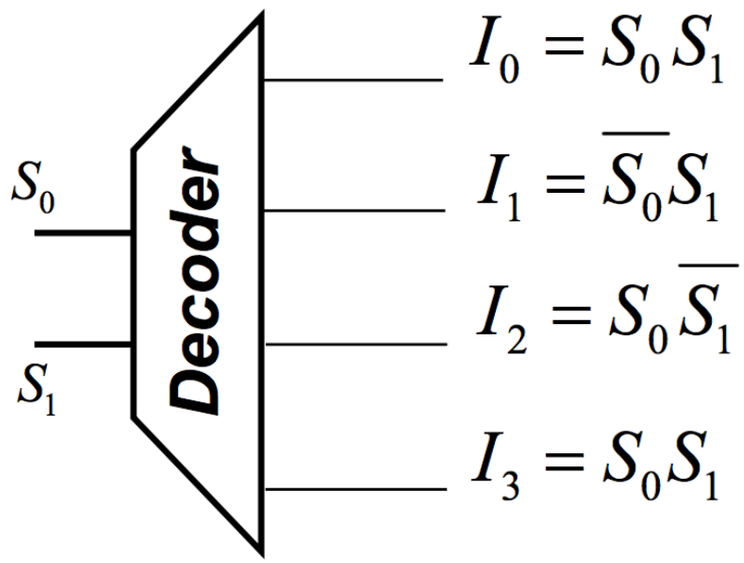


S0	S1	I0	I1	I2	I3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

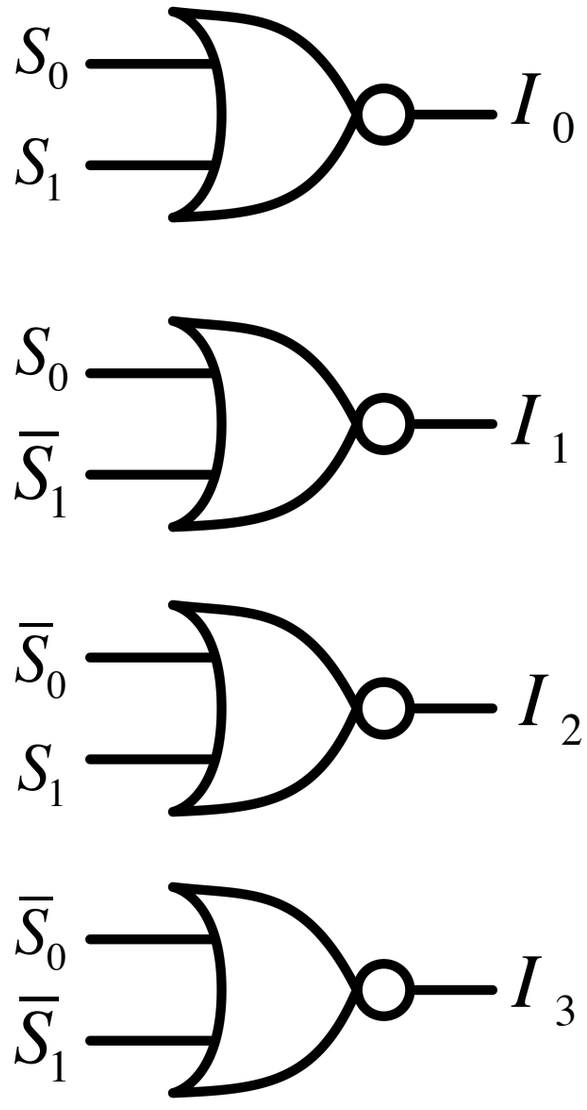


$$F = A.\overline{S_0}\,\overline{S_1} + B.\overline{S_0} S_1 + C.S_0\,\overline{S_1} +_1 + D.S_0 S_1$$

Decoding

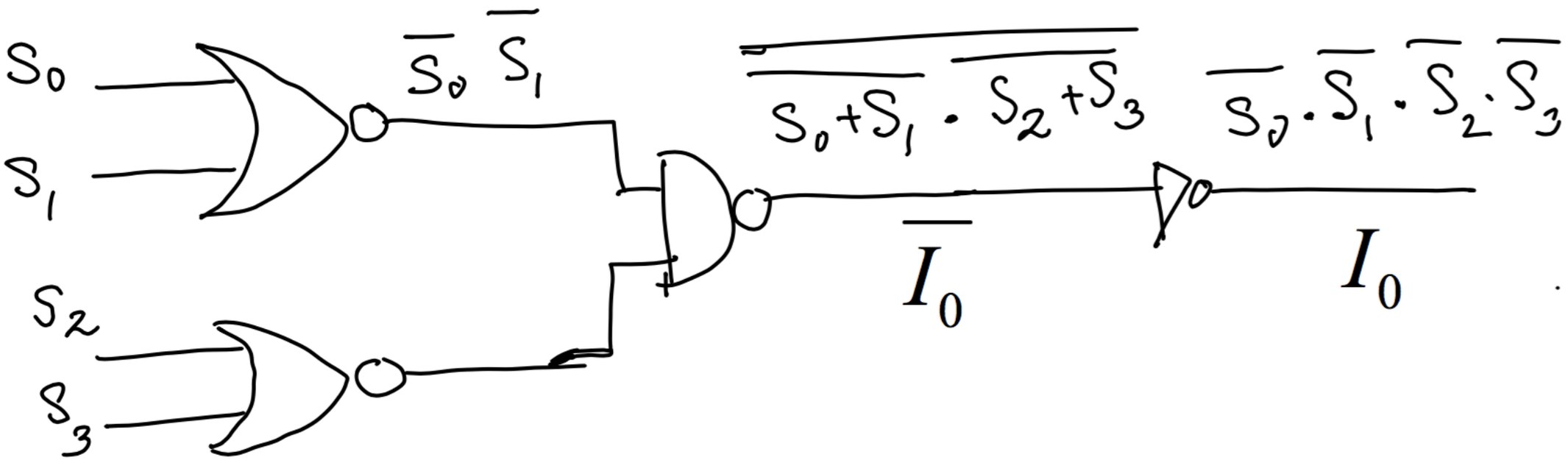


S0	S1	I0	I1	I2	I3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1



Decoding large # of signals

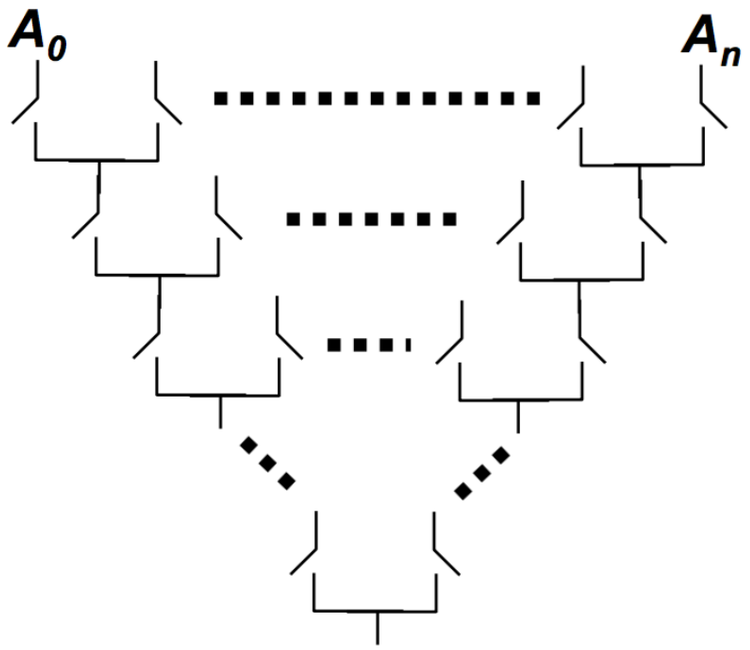
$$I_0 = \overline{S_0 S_1 S_3 S_4} = ?$$



 # gates in critical path =2
(the inverter needs not to be considered as we need both I_0 and its complement)

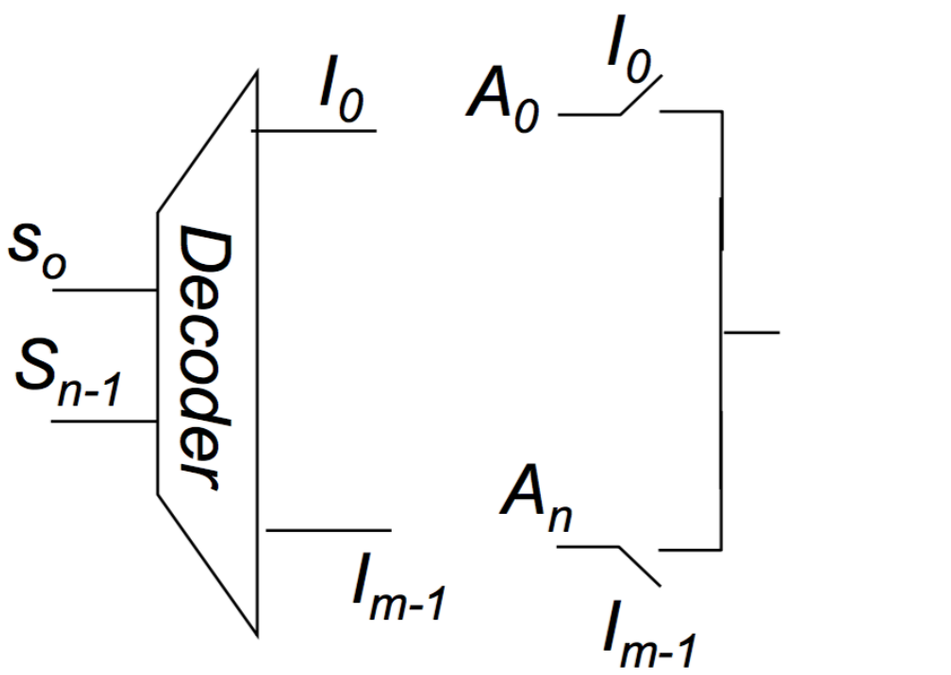
How to design a TG MUX?

Option 1: Embed the decoding logic in the TG Tree



Pros: smaller # of TXs
Cons: delay can be very large for large number of inputs

Option 2: a separate decoder and one level of TG Tree

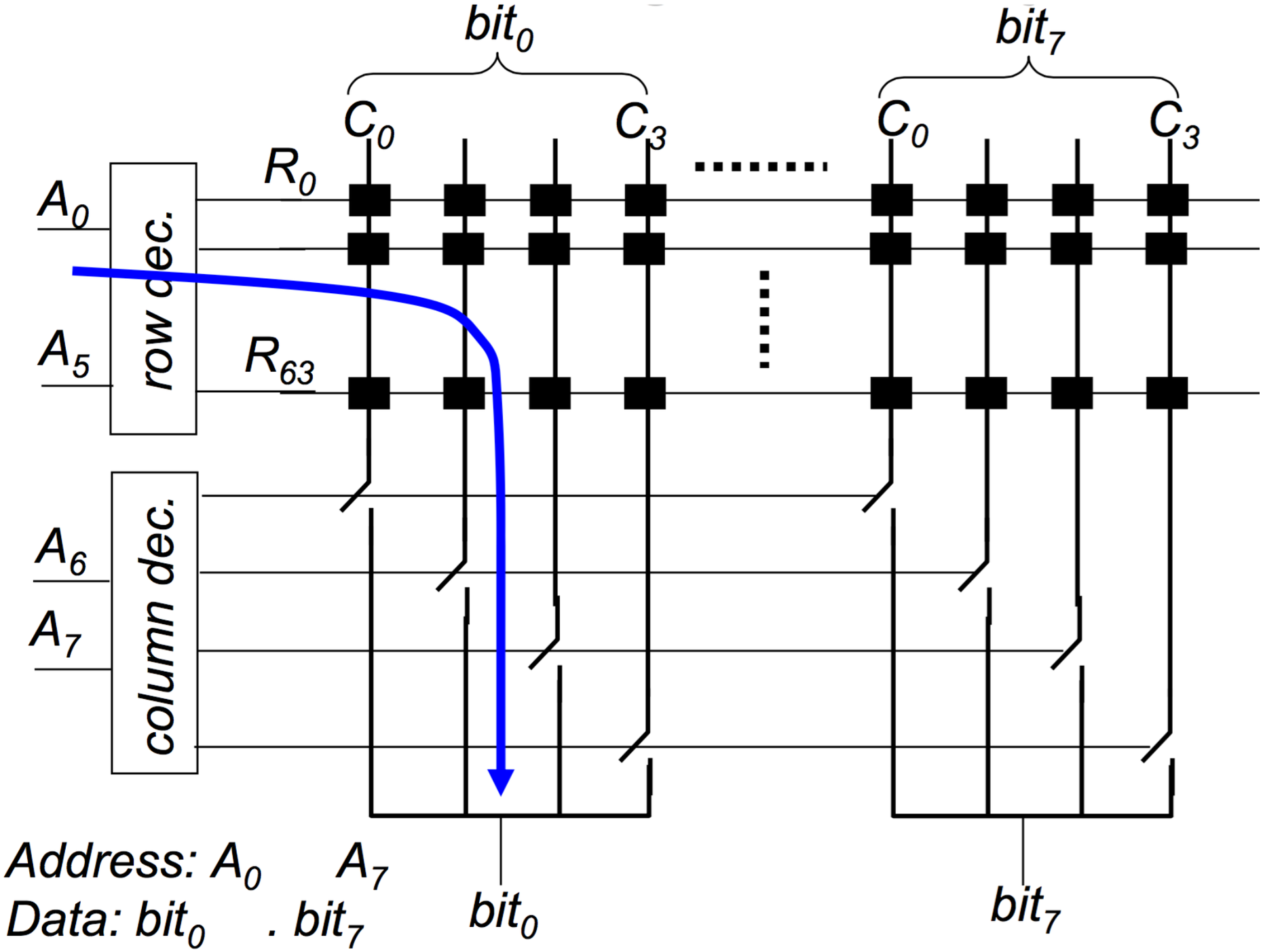


Pros: smaller delay through TG path (can be smaller overall delay)
Smaller TG-area
Cons: Larger total area

How to design a TG MUX? Suggestions

- ***For MUX-ing more than 4 signals, decoding first normally helps***
- ***If area is first priority a TG-chain can be better***
- ***If you know the select signals (s_0, , s_{n-1}) are available earlier than the input signals, decode them first and then use one level of MUX Tree. This will help to hide the decoding delay***
- ***Key for designing column multiplexer in memory arrays***

Column Circuit Design for Memory

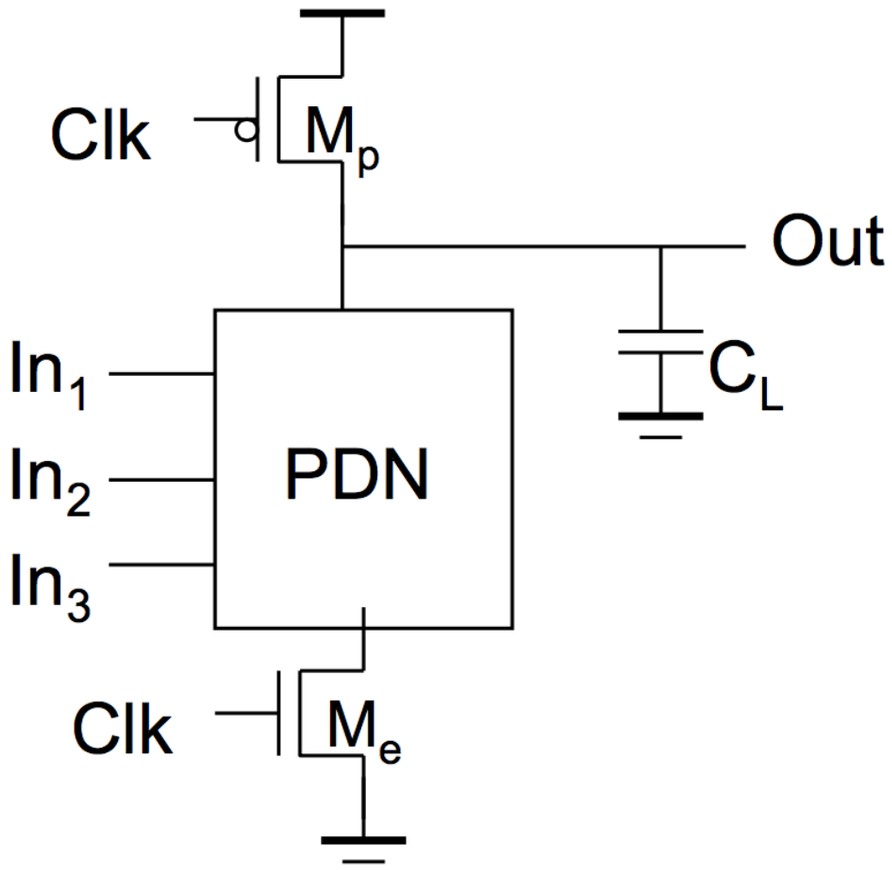


Dynamic Logic

Dynamic versus static

- In static circuits at every point in time (except when switching) the output is connected to either GND or V_{DD} via a low resistance path.
  - fan-in of *n* requires 2*n* (*n* N-type + *n* P-type) devices
- Dynamic circuits rely on the temporary storage of signal values on the capacitance of high impedance nodes.
  - requires on *n* + 2 (*n*+1 N-type + 1 P-type) transistors

Dynamic gate



Dynamic gates use a "Clock" signal and there are two phases of operation:

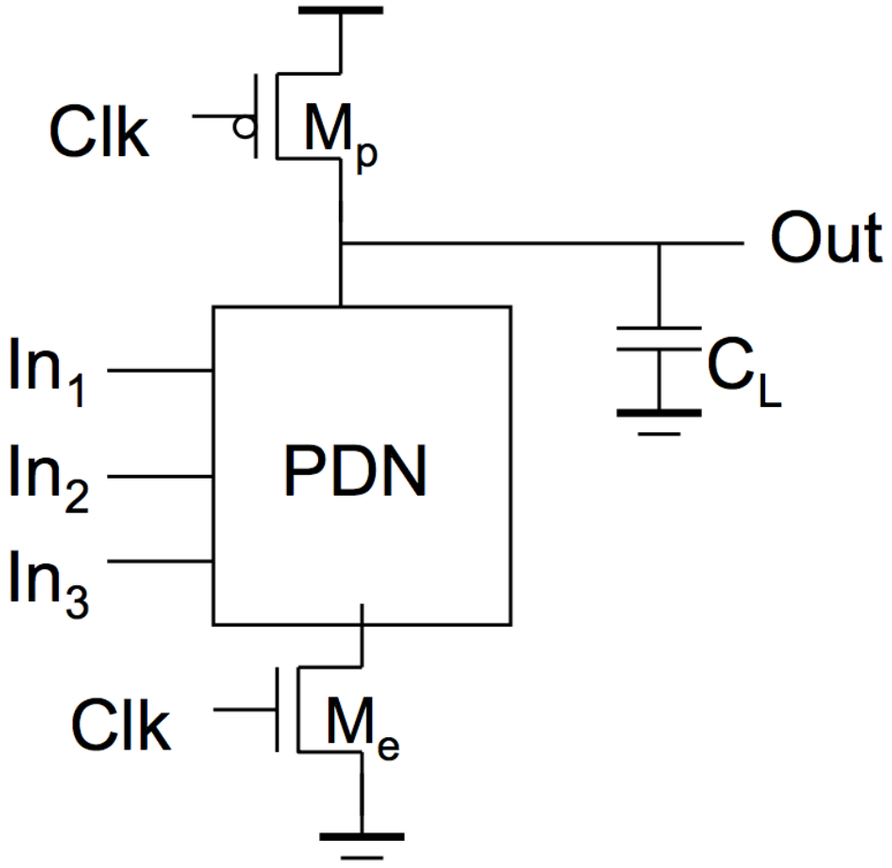
a. **PRECHARGE**
b. **EVALUATE**

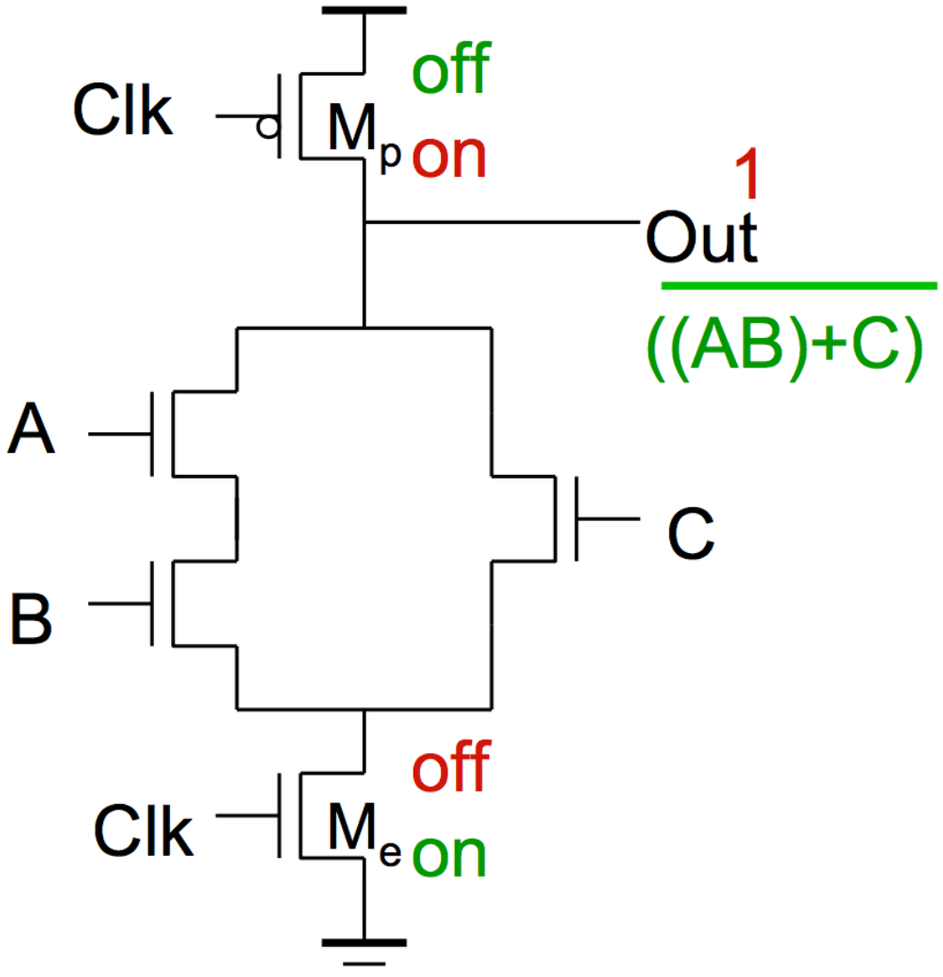
Two phase operation

Precharge (Clk = 0) ⟶ **Mp is ON, Me is OFF**

Evaluate (Clk = 1) ⟶ **Mp is OFF, Me is ON**

Dynamic gate





Two phase operation

Precharge (Clk = 0)

Evaluate (Clk = 1)

Conditions on output

- Once the output of a dynamic gate is discharged, it cannot be charged again until the next precharge operation.
- Inputs to the gate can make at most one transition during evaluation.

- Output can be in the high impedance state during and after evaluation (PDN off), state is stored on C_L

Properties of dynamic gates

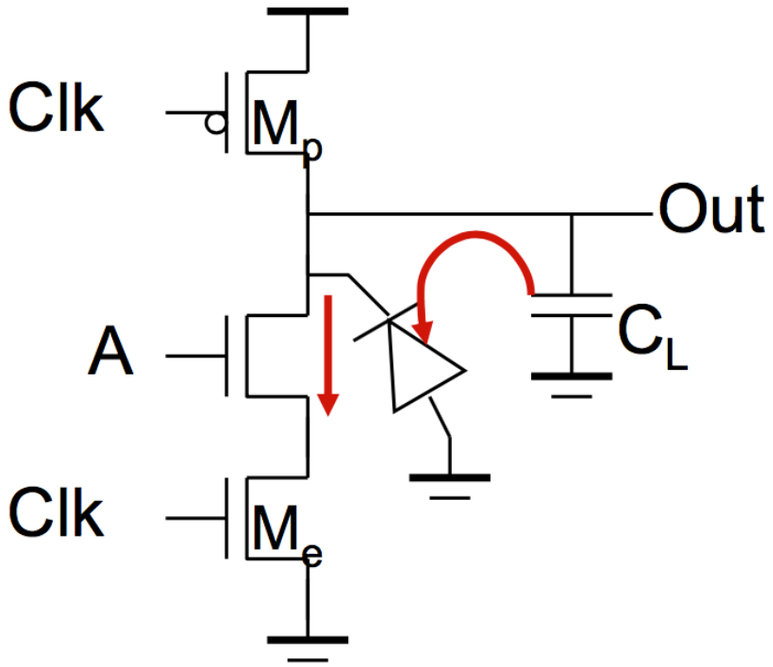
- Logic function is implemented by the PDN only
  - number of transistors is N + 2 (versus 2N for static complementary CMOS)
- Full swing outputs (V_{OL} = GND and V_{OH} = V_{DD})
- Non-ratioed - sizing of the devices does not affect the logic levels
- Faster switching speeds
  - reduced load capacitance due to lower input capacitance (C_{in})
  - reduced load capacitance due to smaller output loading (Cout)
  - no I_{sc}, so all the current provided by PDN goes into discharging C_L

Properties of dynamic gates

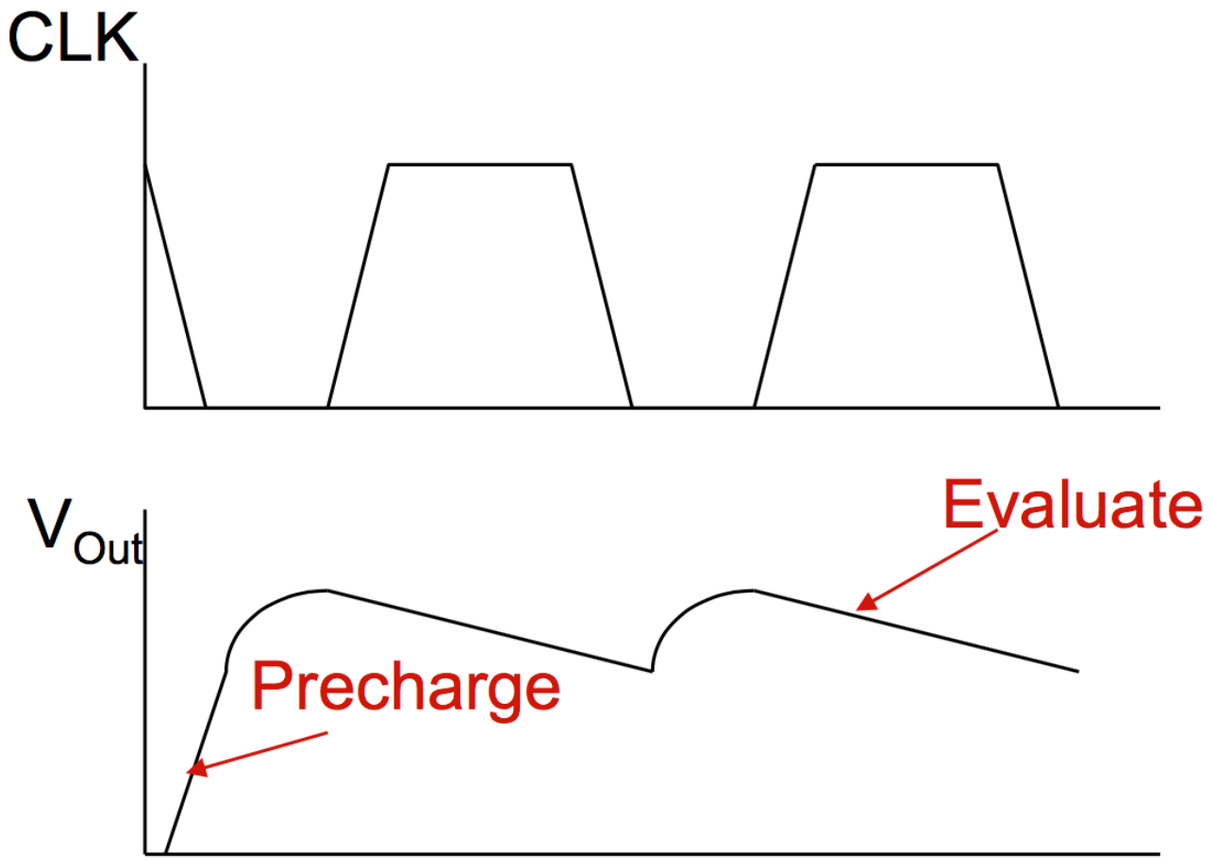
- Overall power dissipation usually higher than static CMOS
  - no static current path ever exists between V_{DD} and GND (including P_{sc})
  - no glitching
  - higher transition probabilities
    - extra load on Clk
- PDN starts to work as soon as the input signals exceed V_{Tn}, so V_M, V_{IH} and V_{IL} equal to V_{Tn}
  - low noise margin (NM_L)
- Needs a precharge/evaluate clock

Issues in dynamic logic

Issue 1: Charge leakage



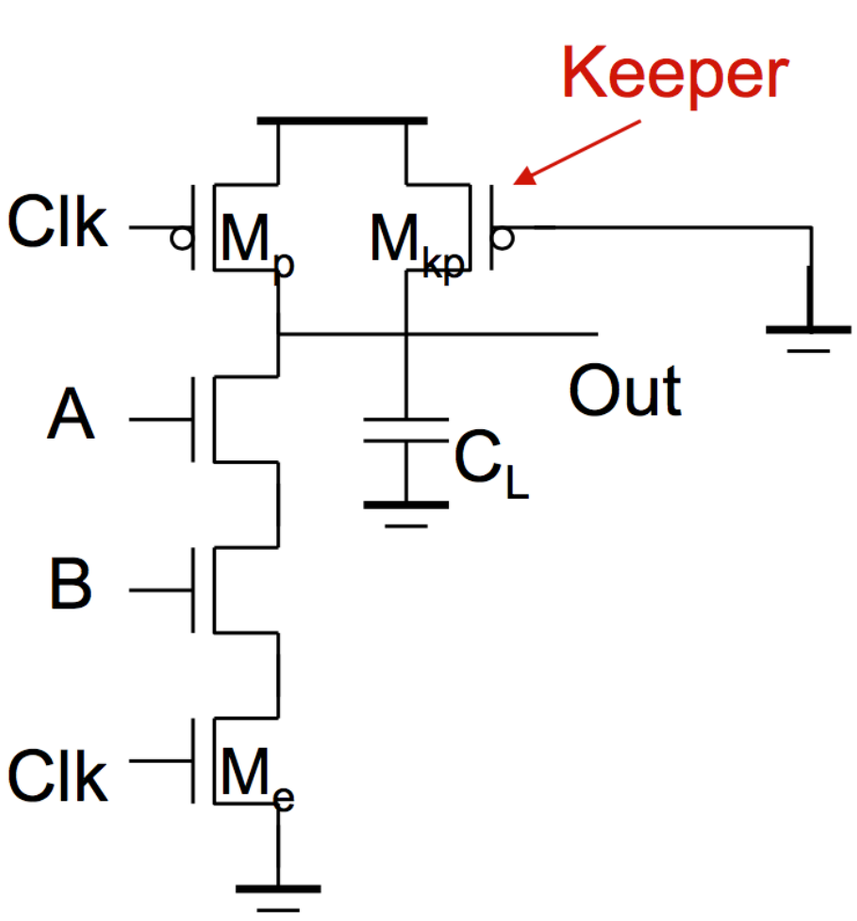
Leakage sources

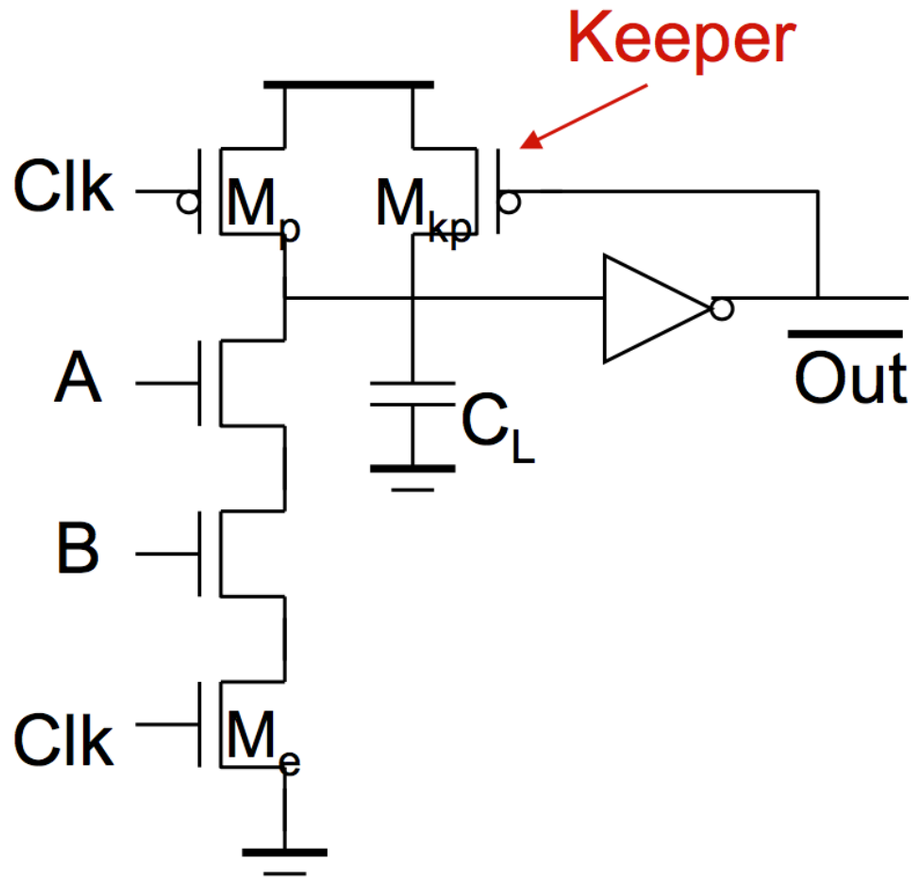


Dominant component is subthreshold current

Solution to charge leakage

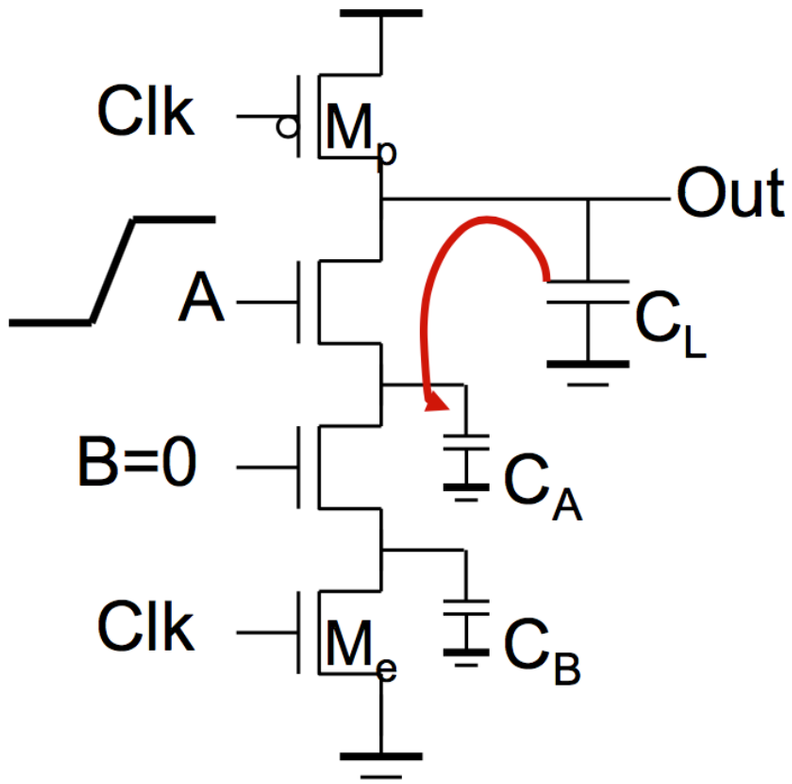
Solution: Use a keeper transistor





Issues in dynamic logic

Issue 2: Charge sharing



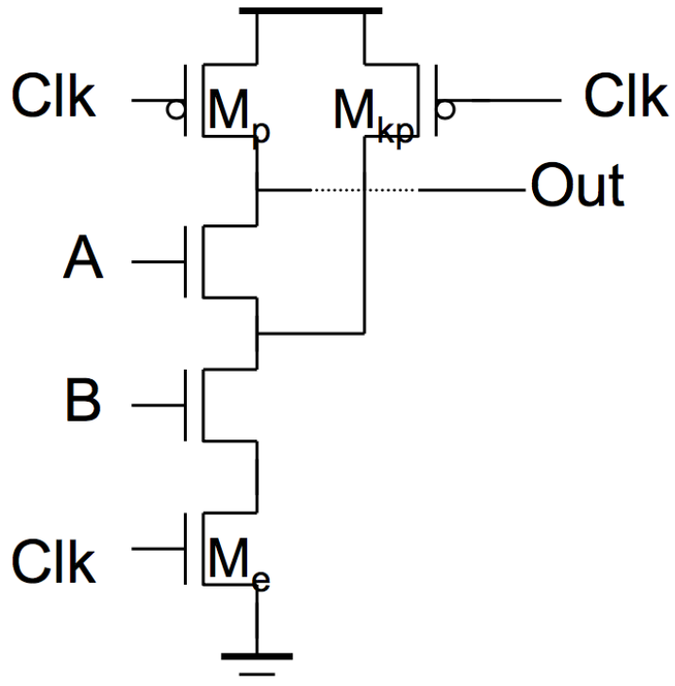
Assume, C_A initially discharged and C_L fully charged.

A makes a 0 -> 1 transition

Charge stored originally on C_L is redistributed (shared) over C_L and C_A leading to reduced robustness

Issues in dynamic logic

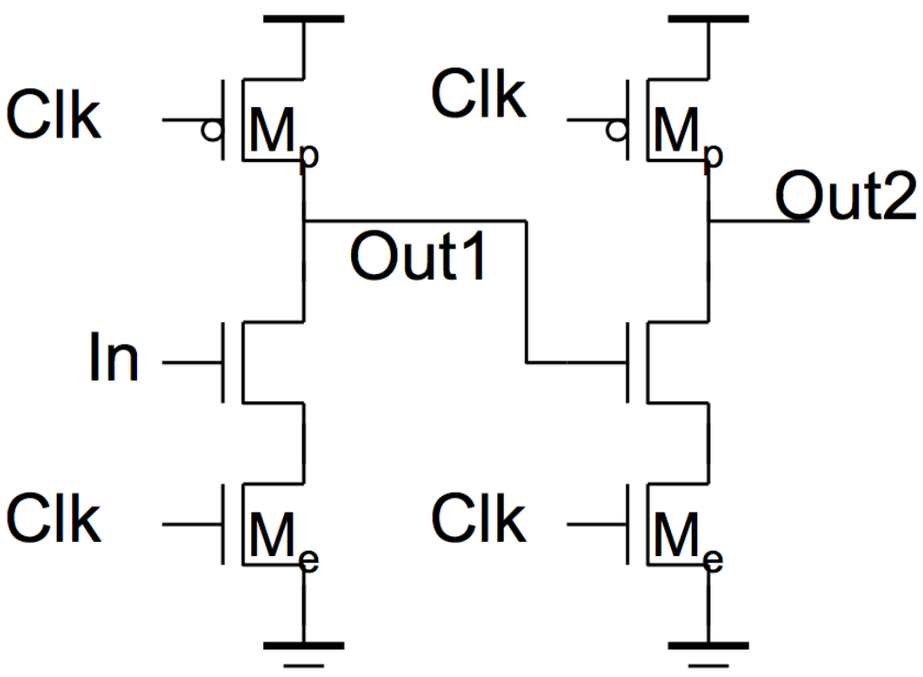
Solution: Charge redistribution

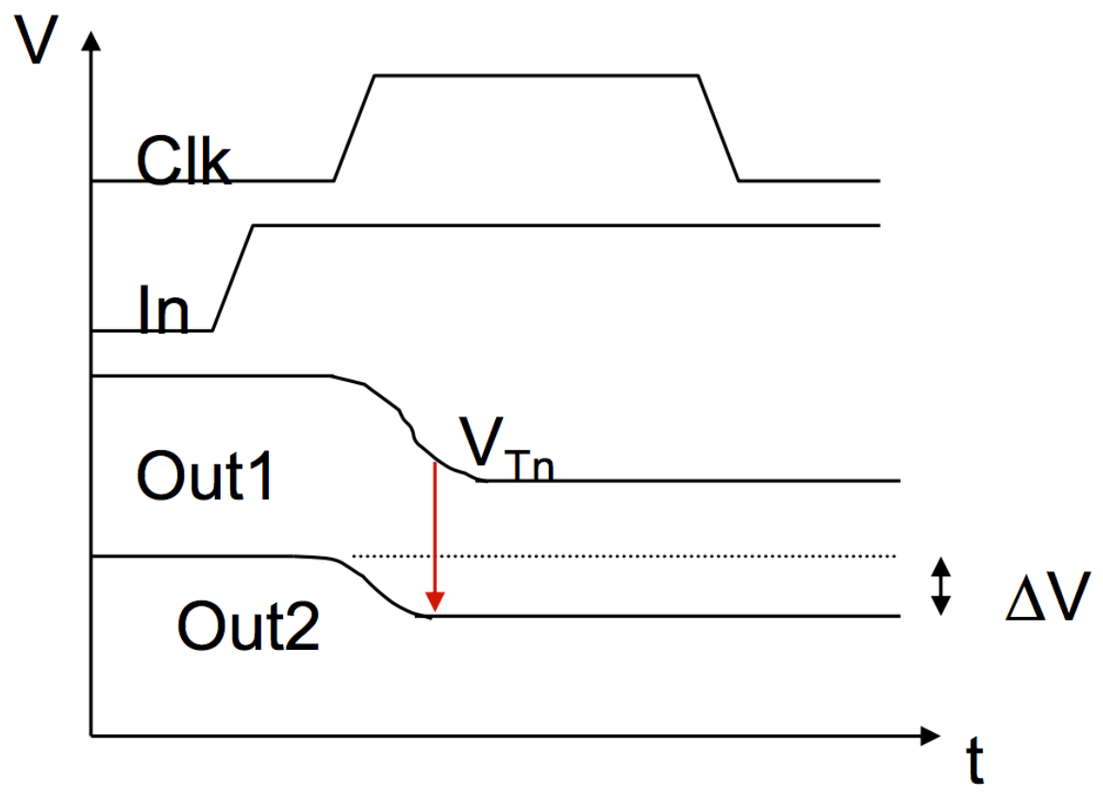


Precharge internal nodes using a clock-driven transistor (at the cost of increased area and power)

Issues in dynamic logic

Issue 3: Cascading





Ideally, Out2 should be same as In !!

But Out2 discharges until Out1 reaches below its threshold voltage to turn the pull down off in the output stage.

How to choose a logic style?

- **Static CMOS:**
  - **robust, ease of implementation and design automation.**
  - **Complex gates requires larger area and has lower speed**
- **Pseudo-NMOS:**
  - **Simple and fast**
  - **Reduced robustness and static power**
- **Pass transistor/Transmission gate:**
  - **Attractive for implementation of specific circuits: XOR, Adders, Multiplexers**
- **Dynamic CMOS:**
  - **High-performance and area efficient for complex gates**
  - **Reduced robustness, complex design process, and higher power dissipation**
- **Current Trend: Increased use of Static CMOS**
  - **Use of EDA tools for logic optimization**
  - **More suitable for voltage scaling as CMOS is more robust to noise and noise does not scale with voltage**

Power Dissipation

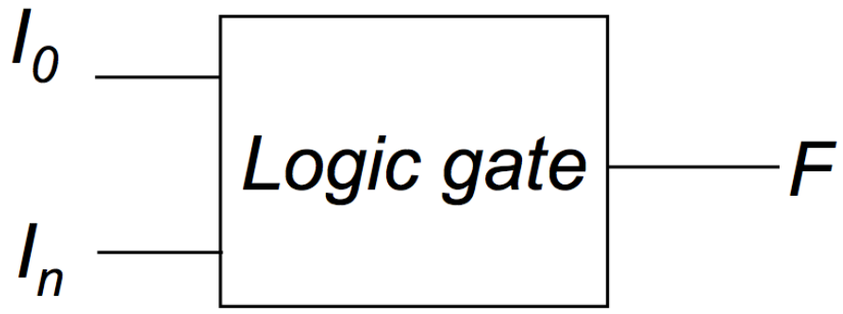
Node transition activity and power

$$P_{avg} = \alpha_{0 \to 1} \bullet C_L \bullet V_{dd}^2 \bullet f_{clk}$$

Power dissipation depends on the switching activity $\alpha_{0 \to 1}$

switching activity $\alpha_{0 \to 1}$: probability of a 0 to 1 transition at output.

Static CMOS gates



- **$0 \rightarrow 1$ transistion => F was '0' before the transition and becomes '1' after the transition**

$$\alpha_{0\rightarrow 1} = p_0 p_1 = p_0 \left(1 - p_0\right)$$

If inputs are non-correlated and uniformly distributed

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

N_0 = number of '0' entries for F in the truth table
N_1 = number of '1' entries for F in the truth table

Example: 2-input NOR gate

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

- **Assume**
  - **inputs are uniformly distributed - i.e. 00, 01,10, 11 has equal probability of occurrences**
  - **Only one input transition is possible in one clock cycle**

$$\alpha_{0\to 1} = \frac{N_0\left(2^2 - N_0\right)}{2^4} = \frac{3(4-3)}{16} = \frac{3}{16}$$

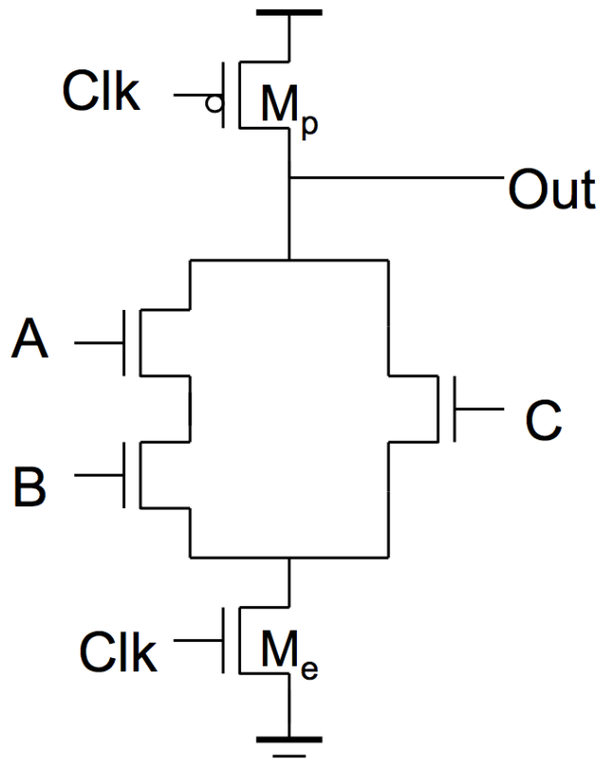
Example: 2-input NAND gate

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

- **Assume**
  - **inputs are uniformly distributed - i.e. 00, 01,10, 11 has equal probability of occurrences**
  - **Only one input transition is possible in one clock cycle**

$$\alpha_{0\to1} = \frac{N_0\left(2^2 - N_0\right)}{2^4} = \frac{1(4-1)}{16} = \frac{3}{16}$$

What happens for dynamic logic ?



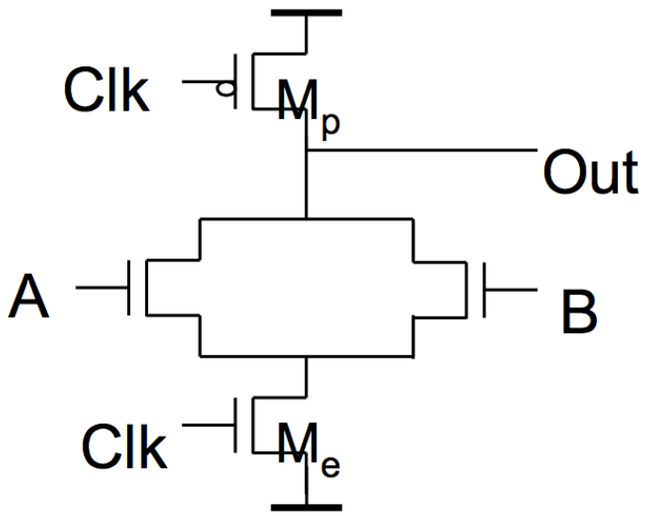
- If Output was evaluated to ‘0’ it will be precharged by the clock irrespective of the inputs
- A 0→1 transition at the output occurs as long as a output was evaluated to ‘0’ in the evaluation cycle.

$$\alpha_{0\to1} = p_0 > p_0 p_1$$

Dynamic logic has larger activity at the output node compared to staic CMOS

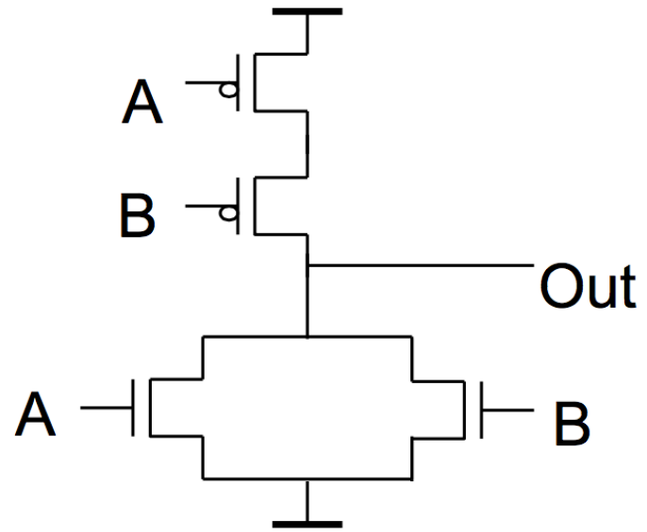
=> larger power dissipation in dynamic logic

Example of a NOR2 gate power dissipation



Dynamic Logic:
Out 0: (01, 10, 11)
$p_0 = 3/4$

Activity:
$\alpha_{0\to1} = p_0 = 0.75 = 3/4$



Static CMOS:
$p_0 = 3/4$
$p_1 = 1/4$

Activity:
$\alpha_{0\to1} = p_0 p_1 = 3/4 \times 1/4 = 3/16$

Signal probability

- Each input has a finite probability of being ‘1’
- Continue assuming that signals are non-correlated

Consider the 2-Input NOR Gates

Output is ‘1’ if both the signals are ‘0’

$$\Pr\{F='1'\} = \Pr\{A=0 \text{ and } B=0\} = \Pr\{A=0\}\Pr\{B=0\}$$

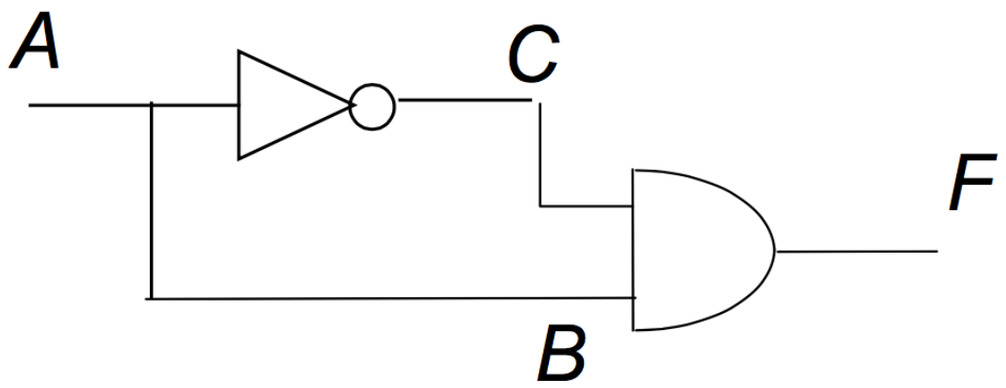
$$\Rightarrow p_1 = (1-p_a)(1-p_b)$$

$$\Rightarrow \alpha_{0\to1} = p_0 p_1 = \left[1-(1-p_a)(1-p_b)\right](1-p_a)(1-p_b)$$

What will be for NAND2?

$$p_0 = p_a p_b \Rightarrow \alpha_{0\to1} = p_0 p_1 = p_a p_b \left[1-p_a p_b\right]$$

Be careful: re-convergent fan-out



We are interested in the activity at node F

$$\Pr\{F='1'\} = \Pr\{B=1 \text{ and } C=1\} \neq \Pr\{B=1\}\Pr\{C=1\}$$ ***Why ?***

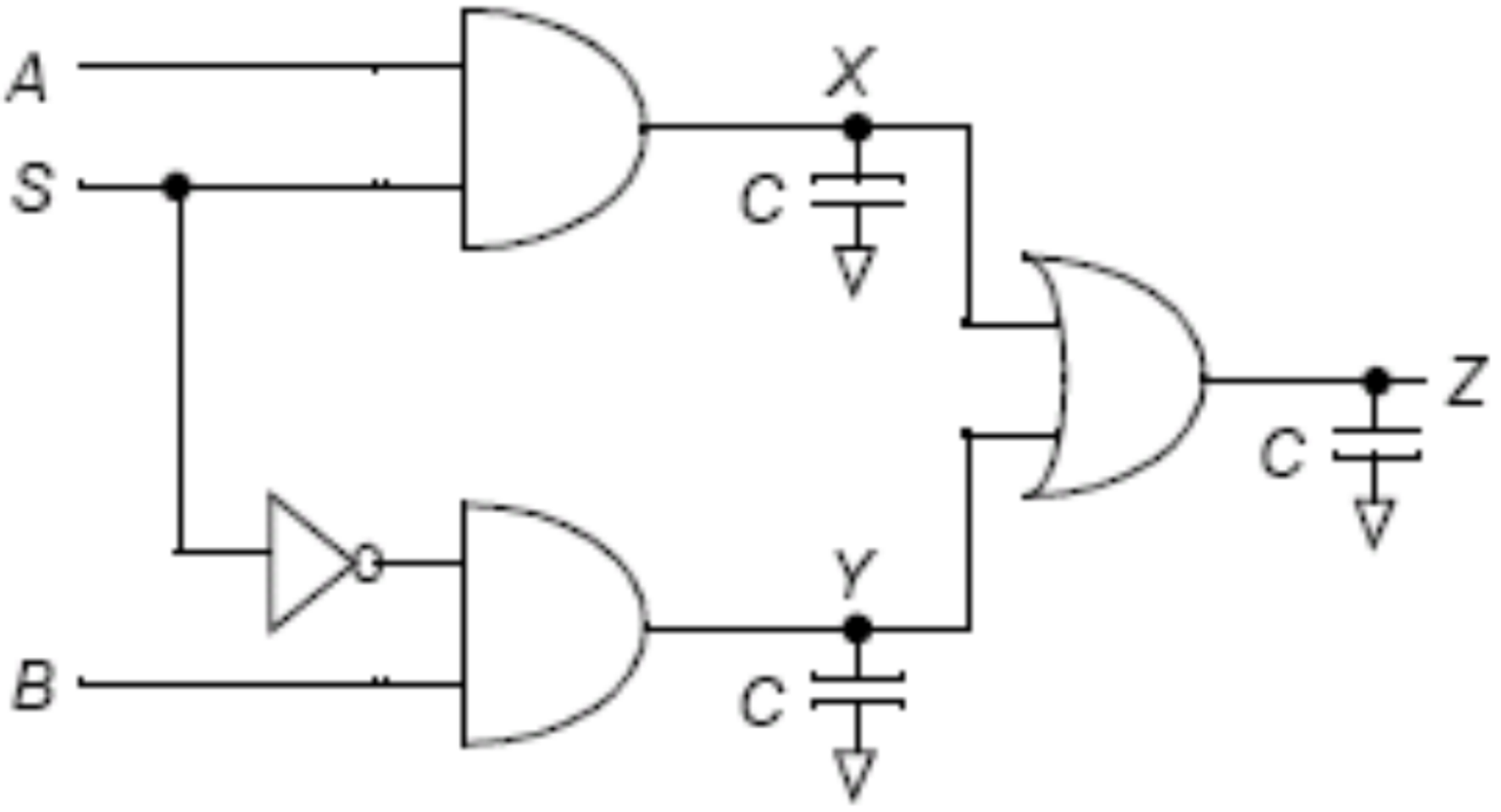
B and C are correlated

$$\Pr\{F='1'\} = \Pr\{B=1 \text{ and } C=1\} = \Pr\{C=1/B=1\}\Pr\{B=1\} = 0$$

$$\alpha_{0\to 1} = p_0 p_1 = 0$$

Re-convergent Fanout

A more complex problem



Class on Oct. 19, 2015

- Power dissipation
- Sequential elements
- Introduction to static random-access memory (SRAM)

Class on Oct. 26, 2015

- Project discussion
    - choose team members, groups of 4.
- Recap of materials for exam