LECTURE 3

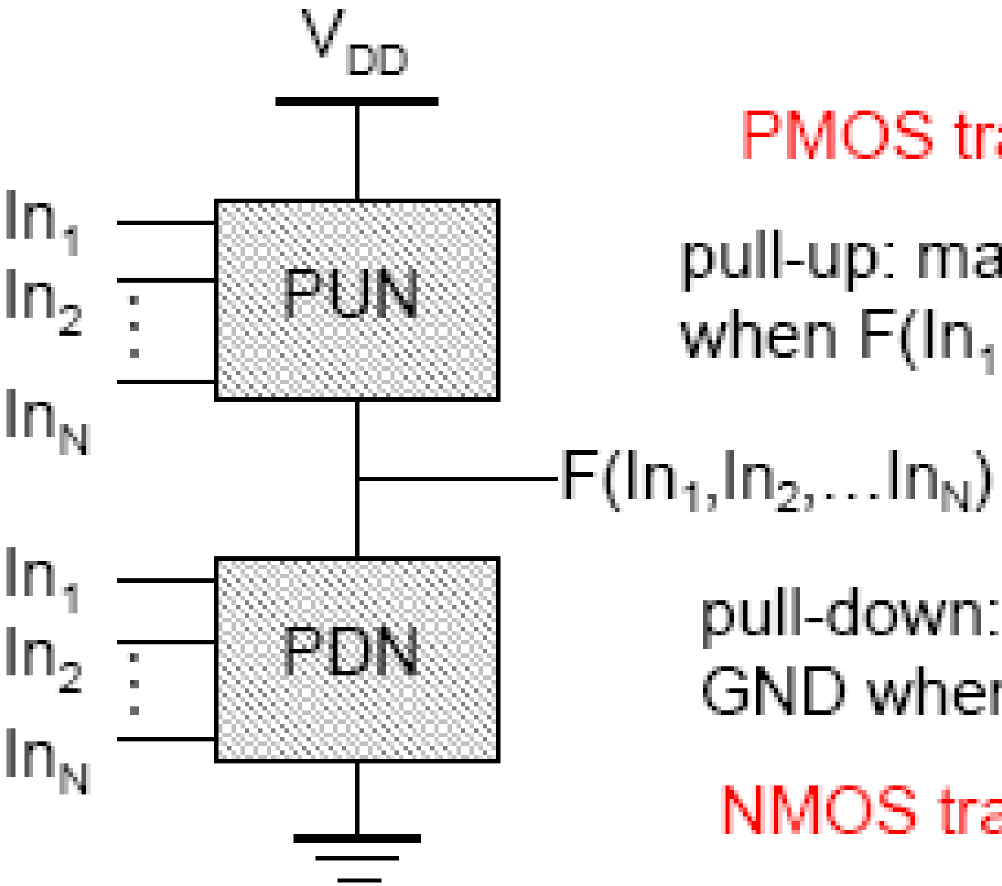
Static CMOS Logic

CMOS Circuit Styles

- Static complementary CMOS - except during switching, output connected to either VDD or GND via a low-resistance path
  - high noise margins
    - full rail to rail swing
    - VOH and VOL are at VDD and GND, respectively
  - low output impedance, high input impedance
  - no steady state path between VDD and GND (no static power consumption)
  - delay a function of load capacitance and transistor resistance
  - 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
  - simpler, faster gates
  - increased sensitivity to noise

Static Complementary CMOS

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



PMOS transistors only

pull-up: make a connection from V_{DD} to F when $F(In_1,In_2,\ldots In_N) = 1$

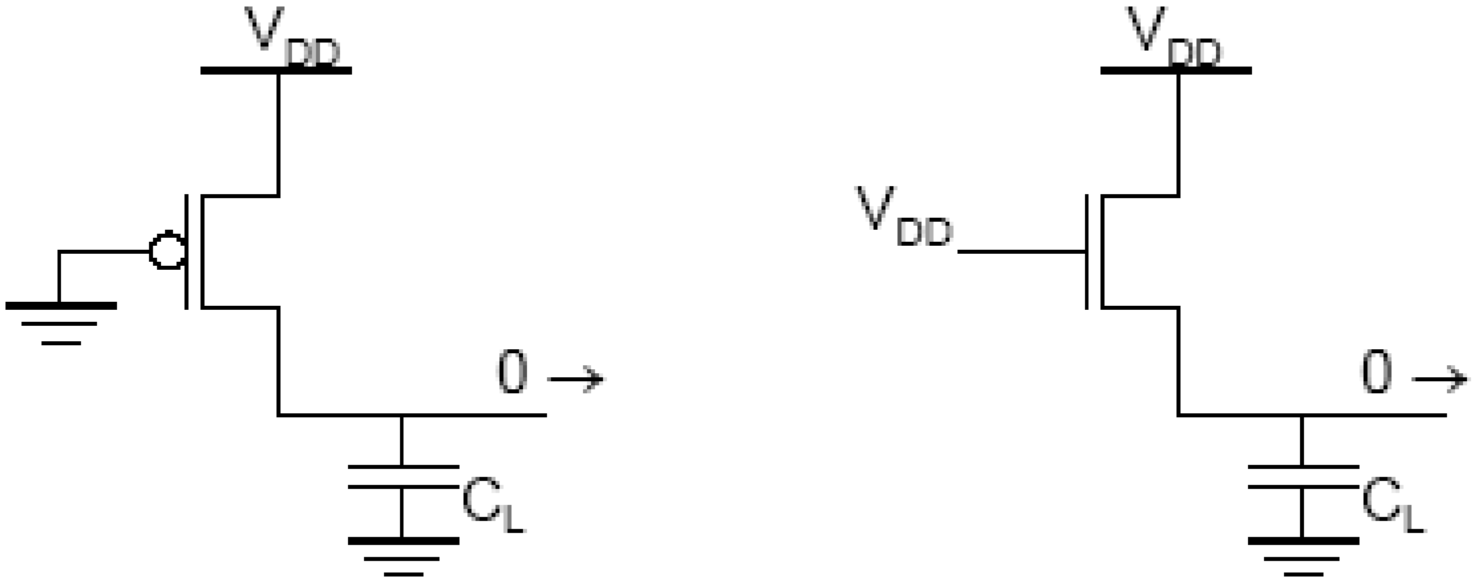
pull-down: make a connection from F to GND when $F(In_1,In_2,\ldots In_N) = 0$

NMOS transistors only

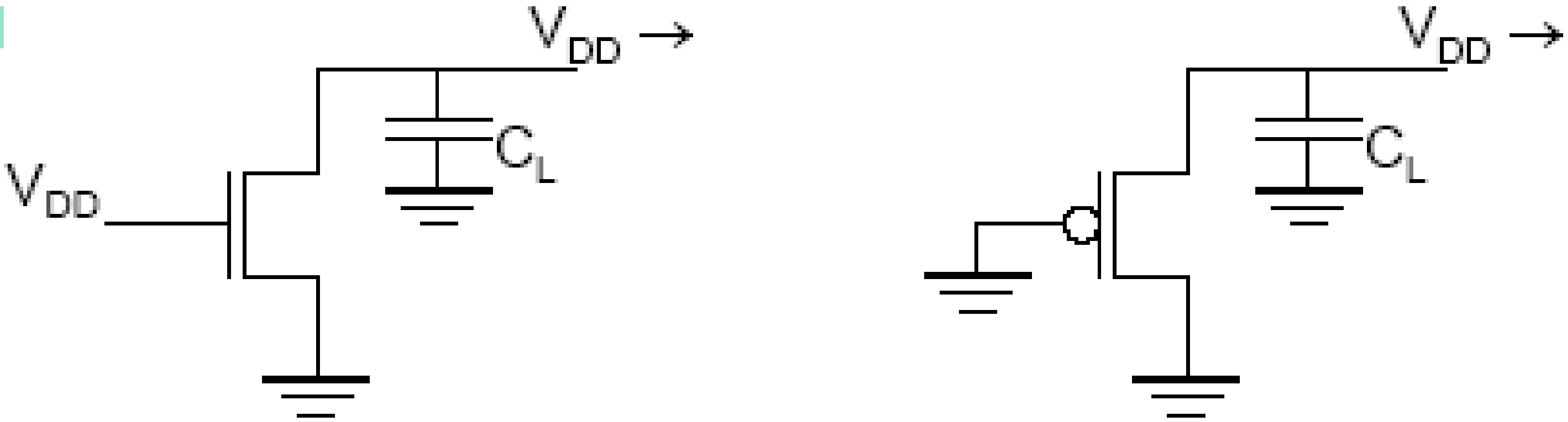
PUN and PDN are dual logic networks

Threshold Drops

PUN

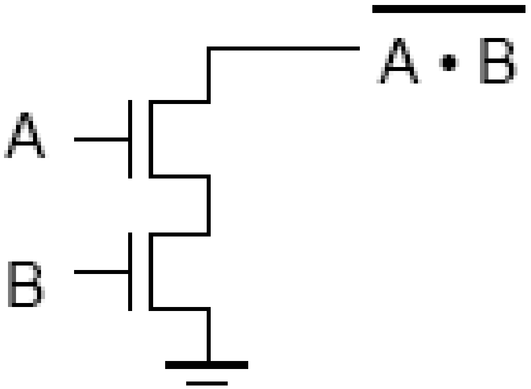


PDN

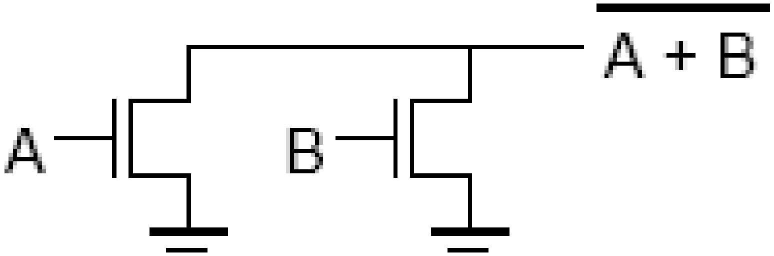


Construction of PDN

- 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
  - DeMorgan's theorems

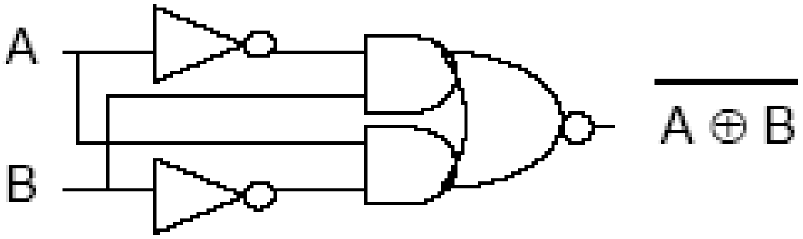
    $\overline{A + B} = \overline{A} \cdot \overline{B}$ [!(A + B) = !A • !B or !(A | B) = !A & !B]

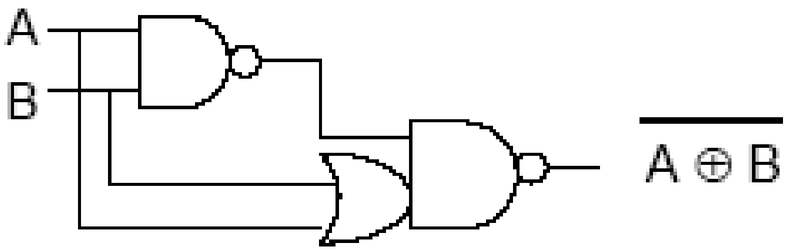
    $\overline{A \cdot B} = \overline{A} + \overline{B}$ [!(A • B) = !A + !B or !(A & B) = !A | !B]

  - a parallel connection of transistors in the PUN corresponds to a series connection of the PDN
- Complementary gate is naturally inverting (NAND, NOR, AOI, OAI)
- Number of transistors for an N-input logic gate is 2N

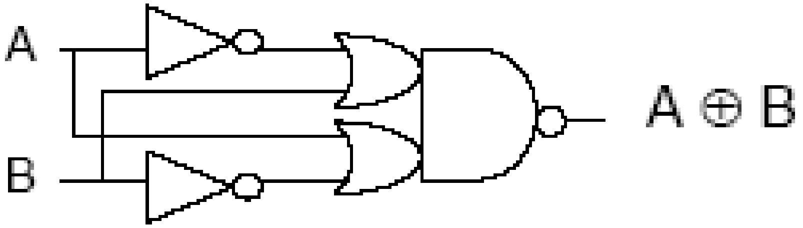
XNOR/XOR Implementation

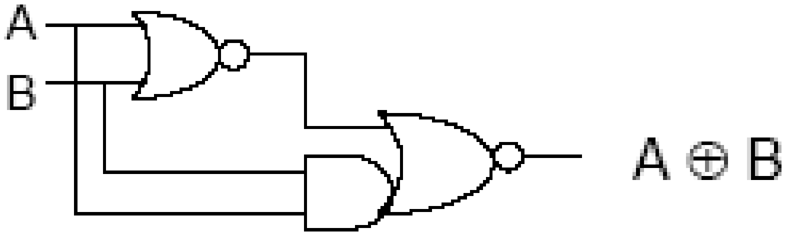
XNOR





XOR





- How many transistors in each?
- Can you create the stick transistor layout for the lower left circuit?

Combinational Logic Cells

- CMOS logic cells
  - AND-OR-INVERT (AOI)
  - OR-AND-INVERT(OAI)
- Example: AOI221

  Z = (A*B + C*D + E)’

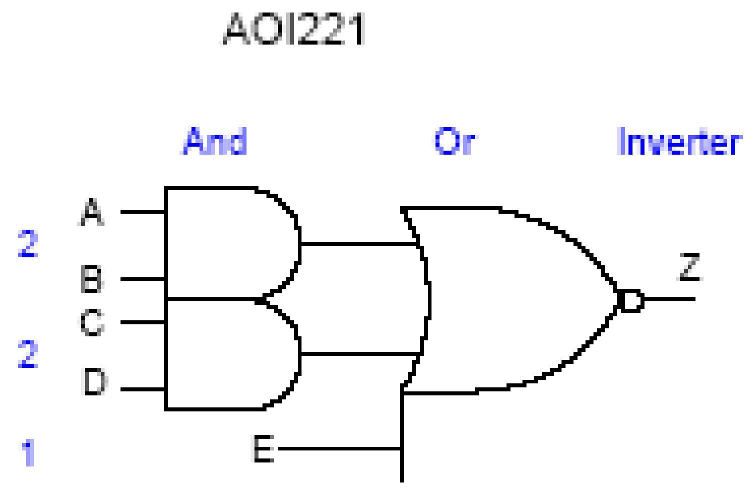
  Z = AOI221(A, B, C, D, E)

  Exercise: Construct this logic cell?
- Example: OAI321

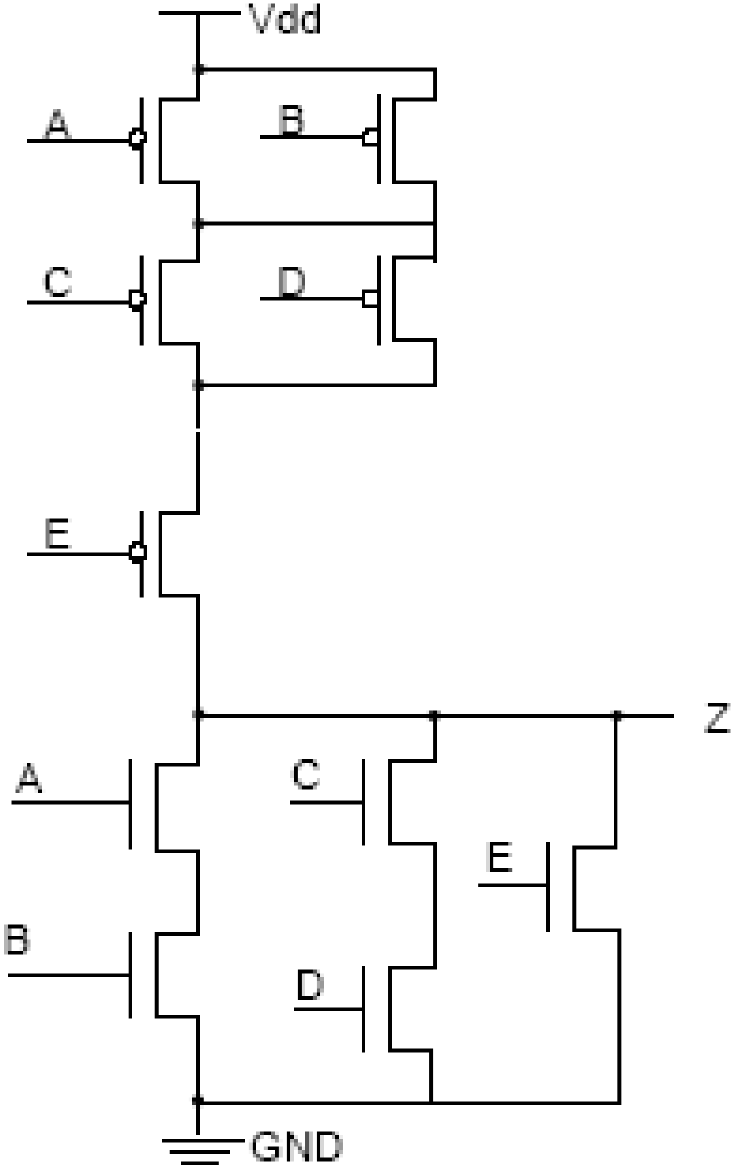
  Z = [(A+B+C)*(D+E)*F]’

  Z = OAI321(A, B, C, D, E, F)

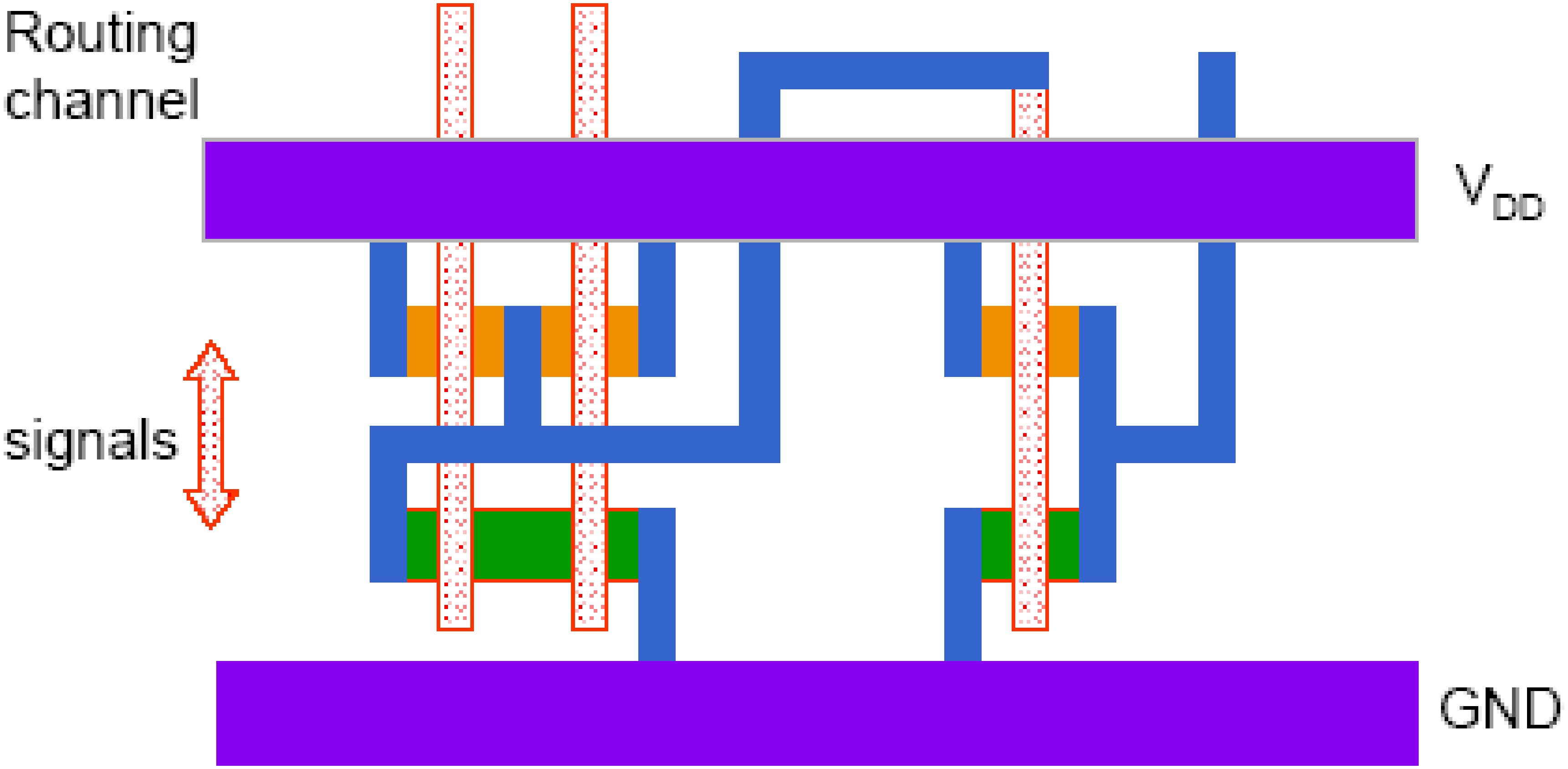
  Exercise: Construct this logic cell?



AOI221



Standard Cell Layout Methodology

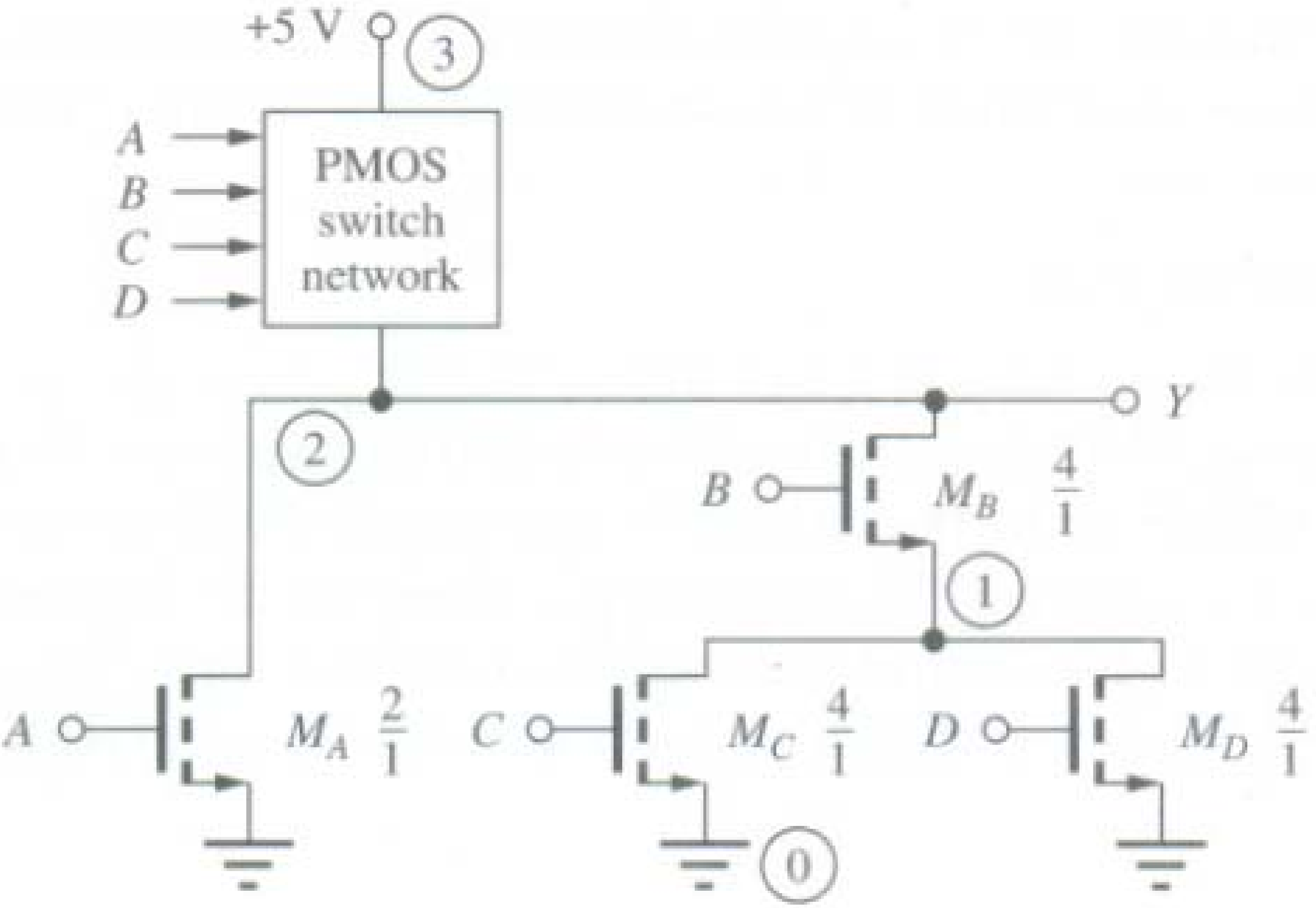


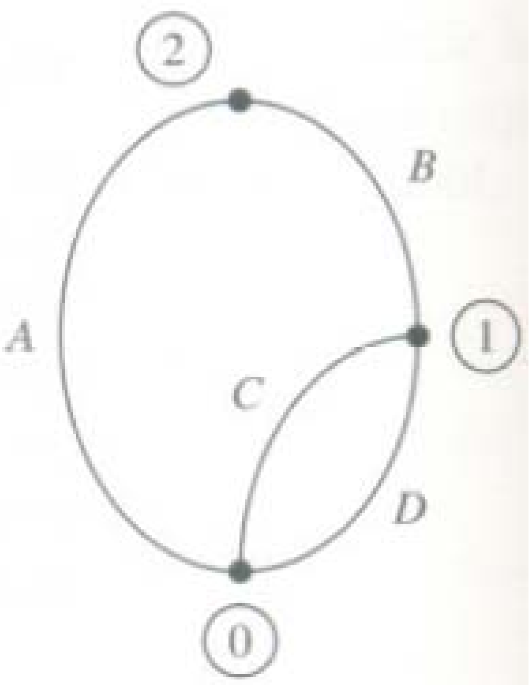
What logic function is this?

COMPLEX GATES IN CMOS

Graphical approach to find the PMOS network

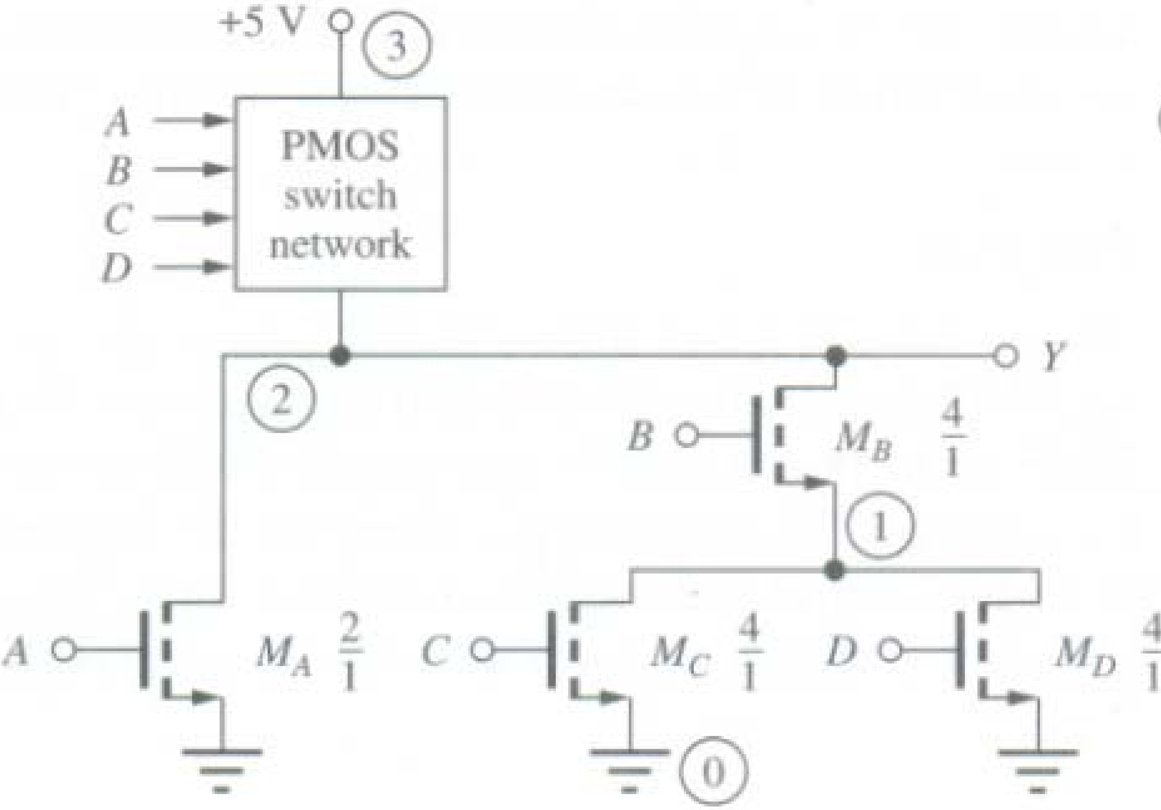
Implement the logic function $Y = \overline{A + BC + BD}$ or $\overline{Y} = A + BC + BD$

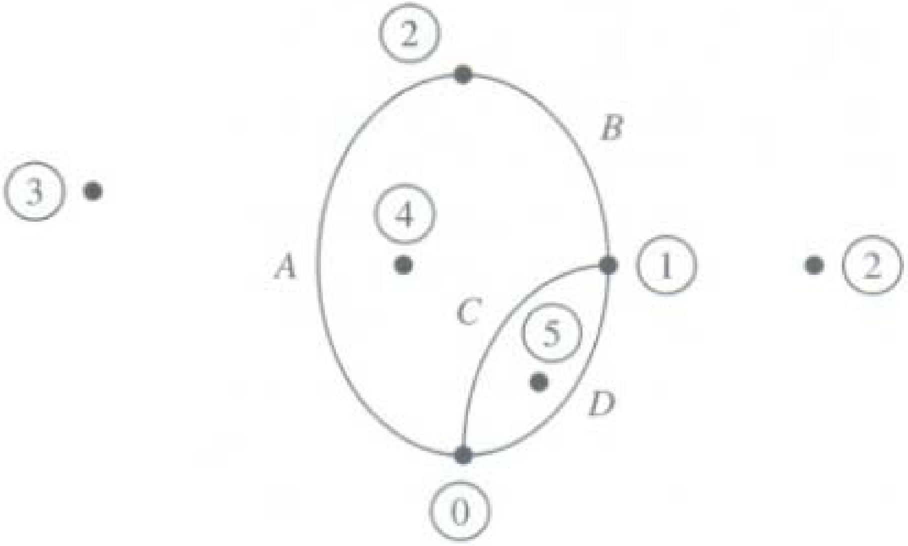




Each NMOS transistor is represented by an arc connecting the source and drain nodes of the transistors and is labeled with the logical input variable.

Graphical approach to find the PMOS network

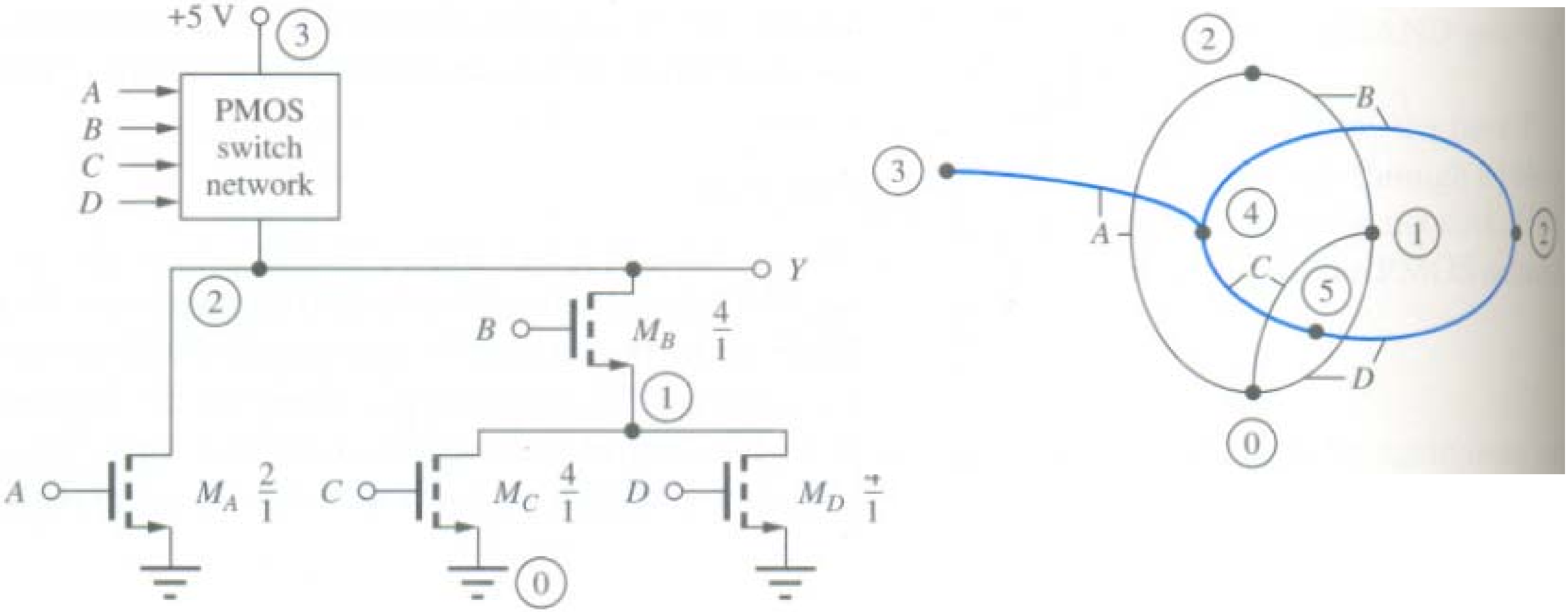




➢ Placing a new node inside of every enclosed path

➢ Two exterior nodes are needed: one representing the output and one representing V_{DD}

Graphical approach to find the PMOS network

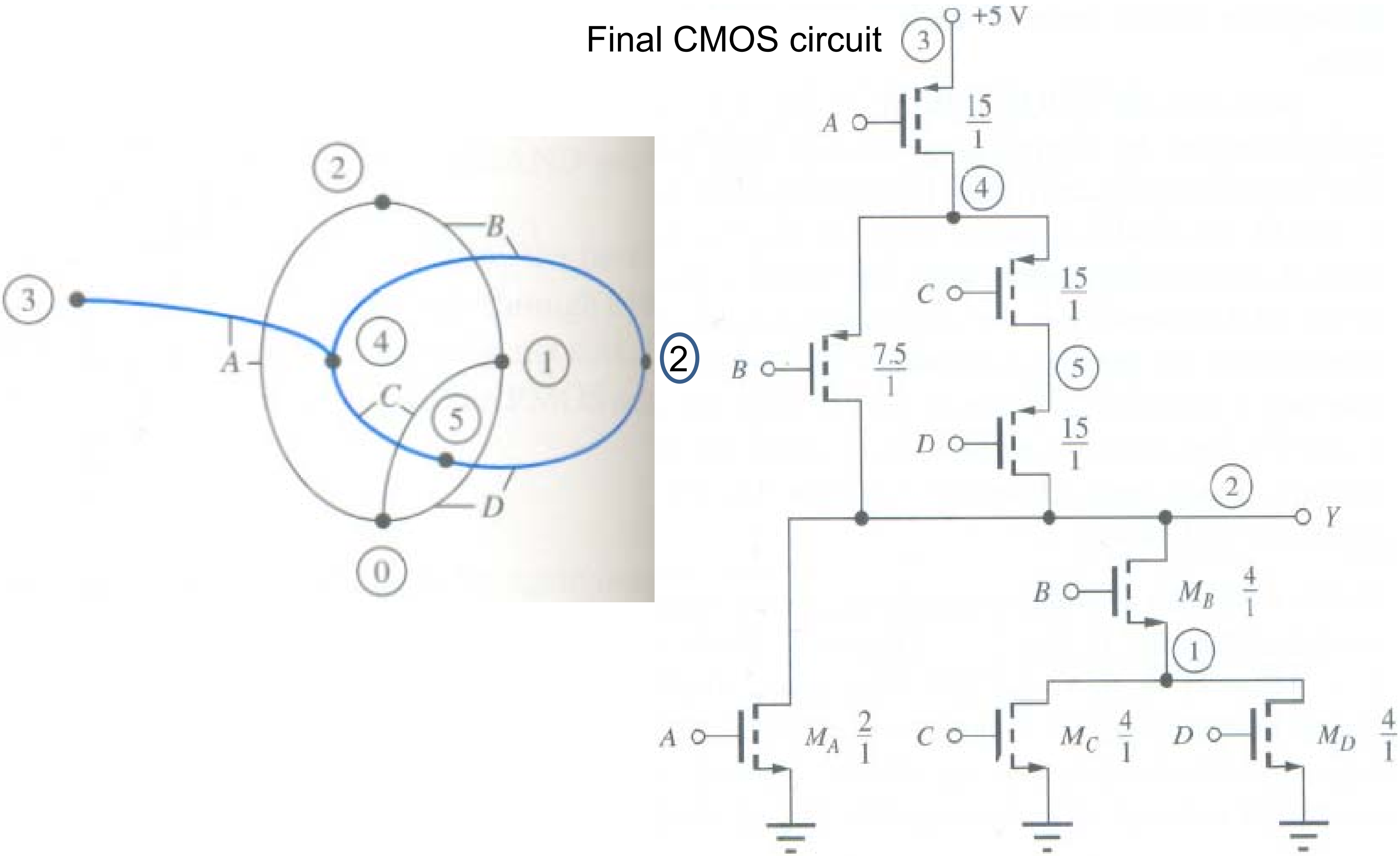


New arcs cut through the NMOS arcs and connect the pairs of nodes that are separated by the NMOS arcs →

→ has the same logic label as the NMOS arc that is intersected

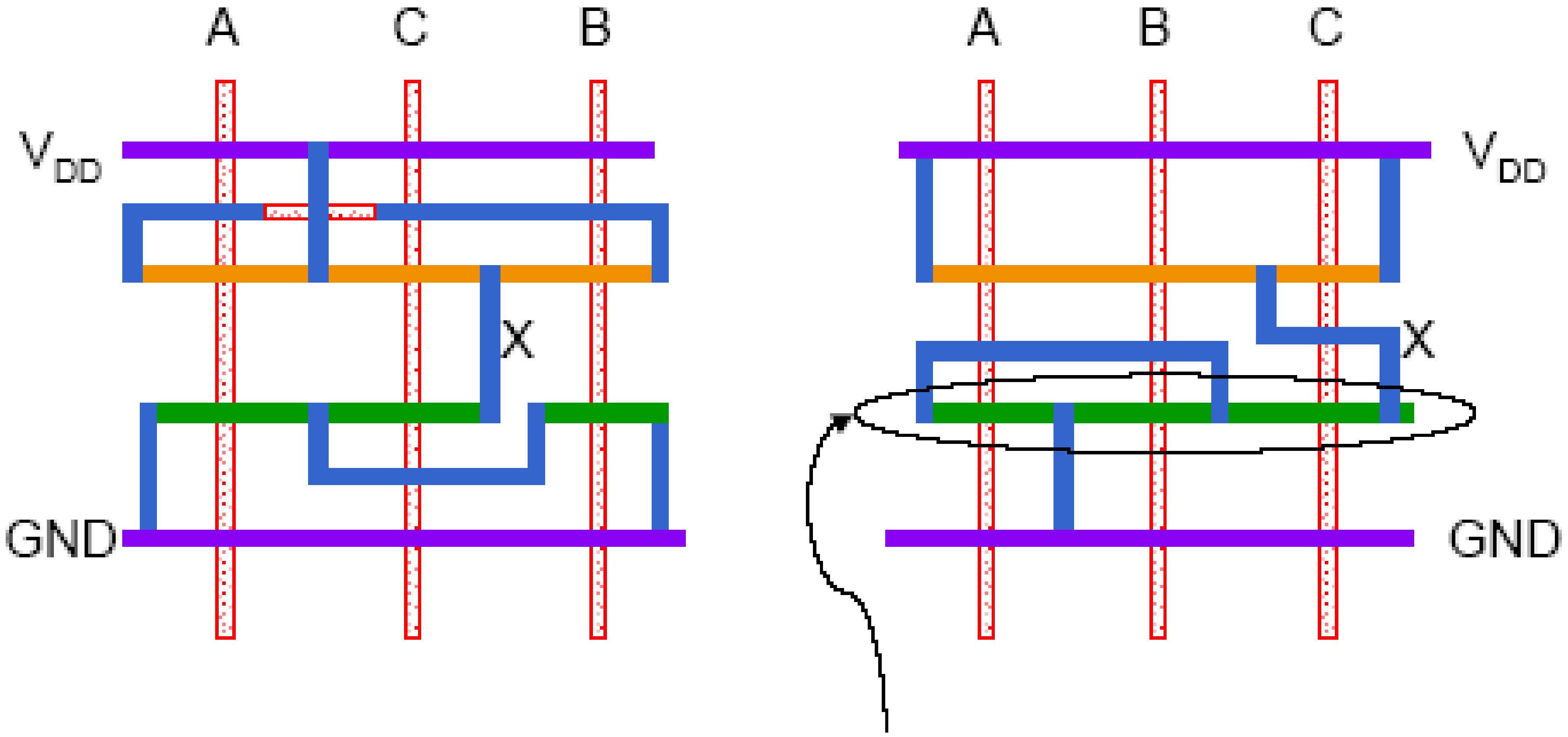
→ minimum PMOS logic network → only one PMOS transistor per logic input.

Graphical approach to find the PMOS network


Final CMOS circuit

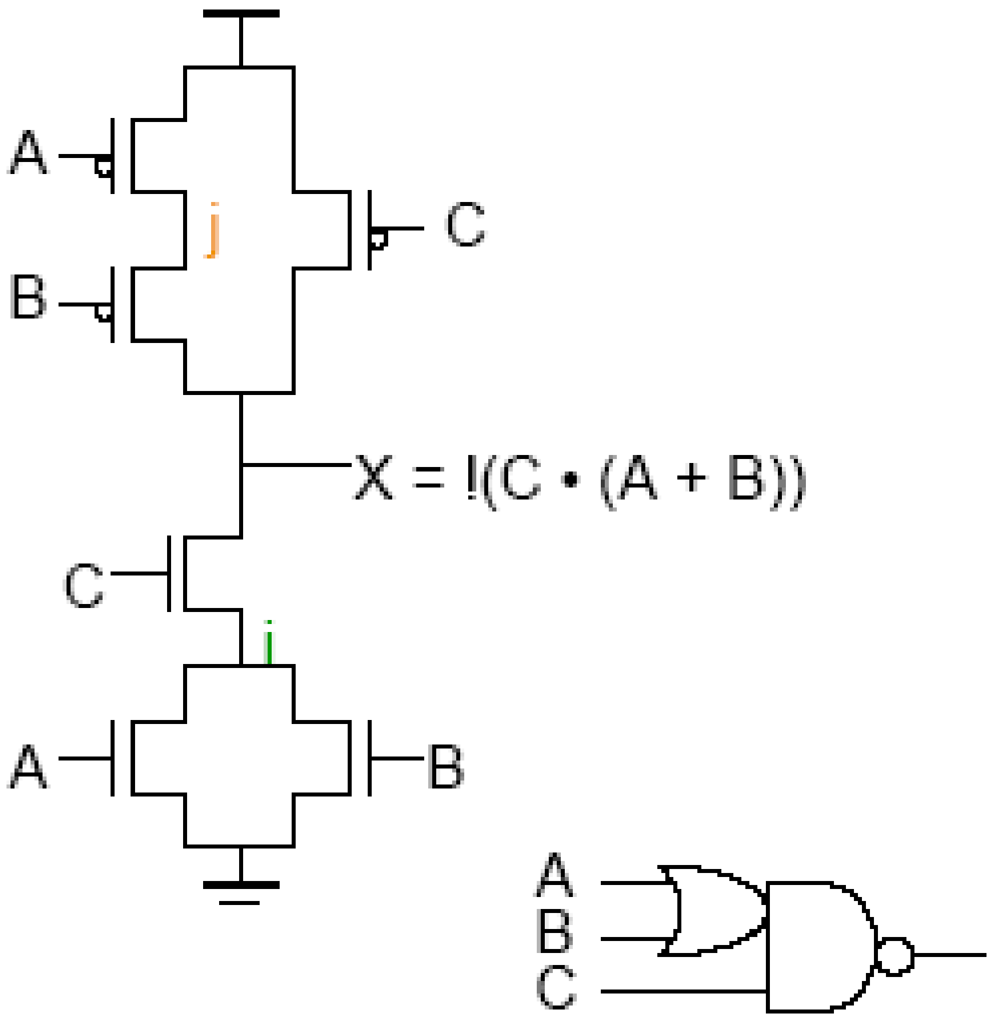
Two Stick Layouts

of !(C • (A + B))



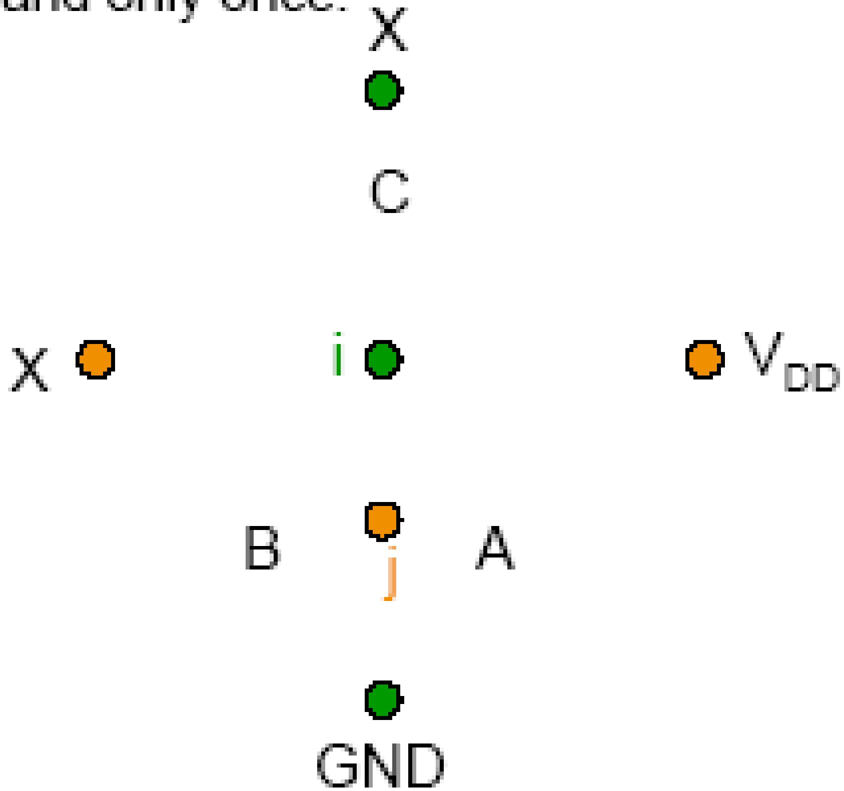
uninterrupted diffusion strip

OAI21 Logic Graph



Consistent Euler Path

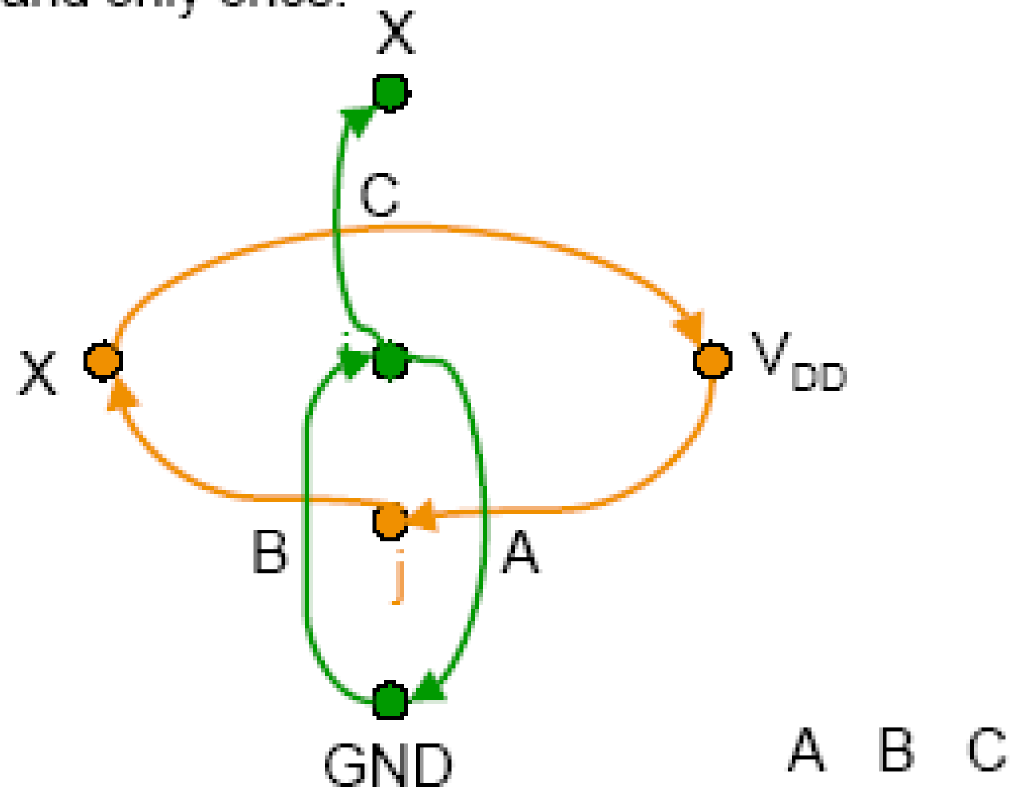
- An uninterrupted diffusion strip is possible only if there exists a Euler path in the logic graph
  - Euler path: a path through all nodes in the graph such that each edge is visited once and only once.



- For a single poly strip for every input signal, the Euler paths in the PUN and PDN must be consistent (the same)

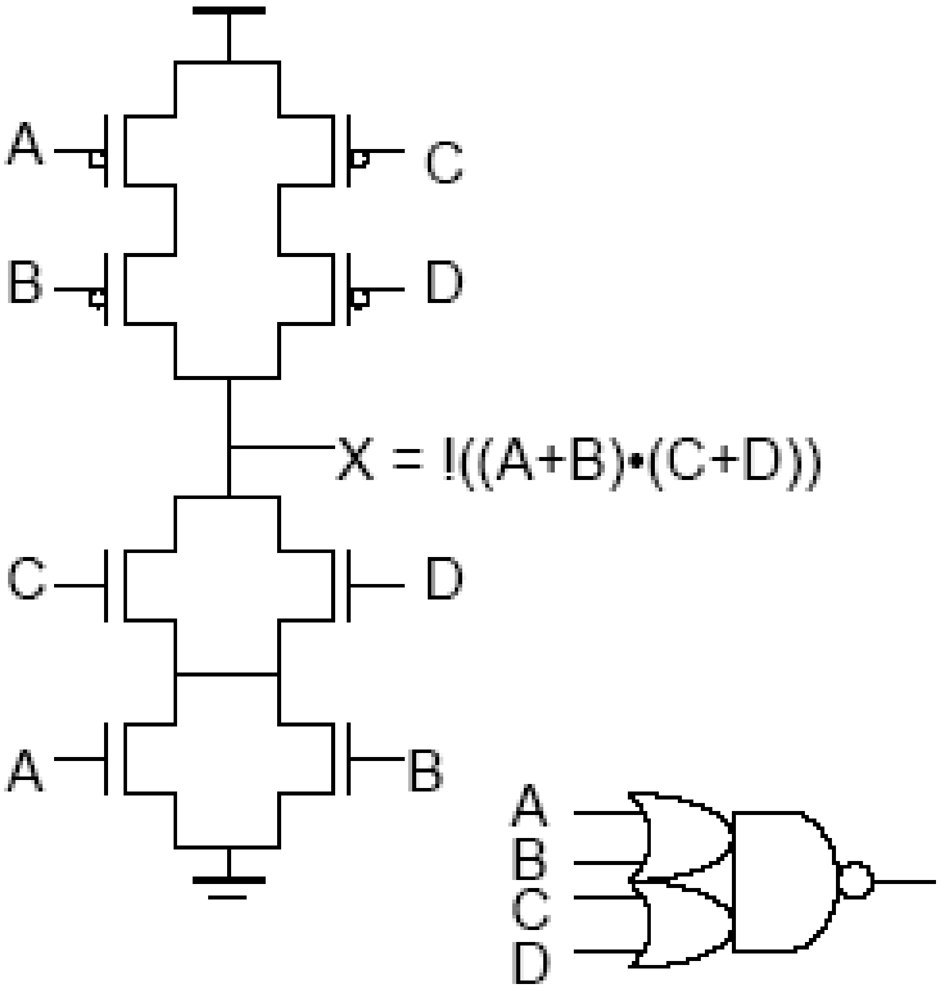
Consistent Euler Path

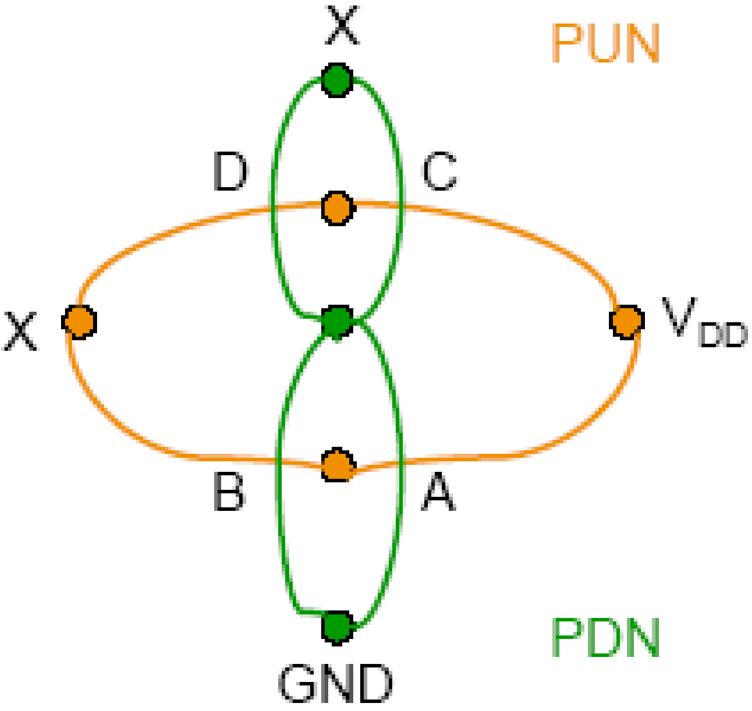
- An uninterrupted diffusion strip is possible only if there exists a Euler path in the logic graph
  - Euler path: a path through all nodes in the graph such that each edge is visited once and only once.



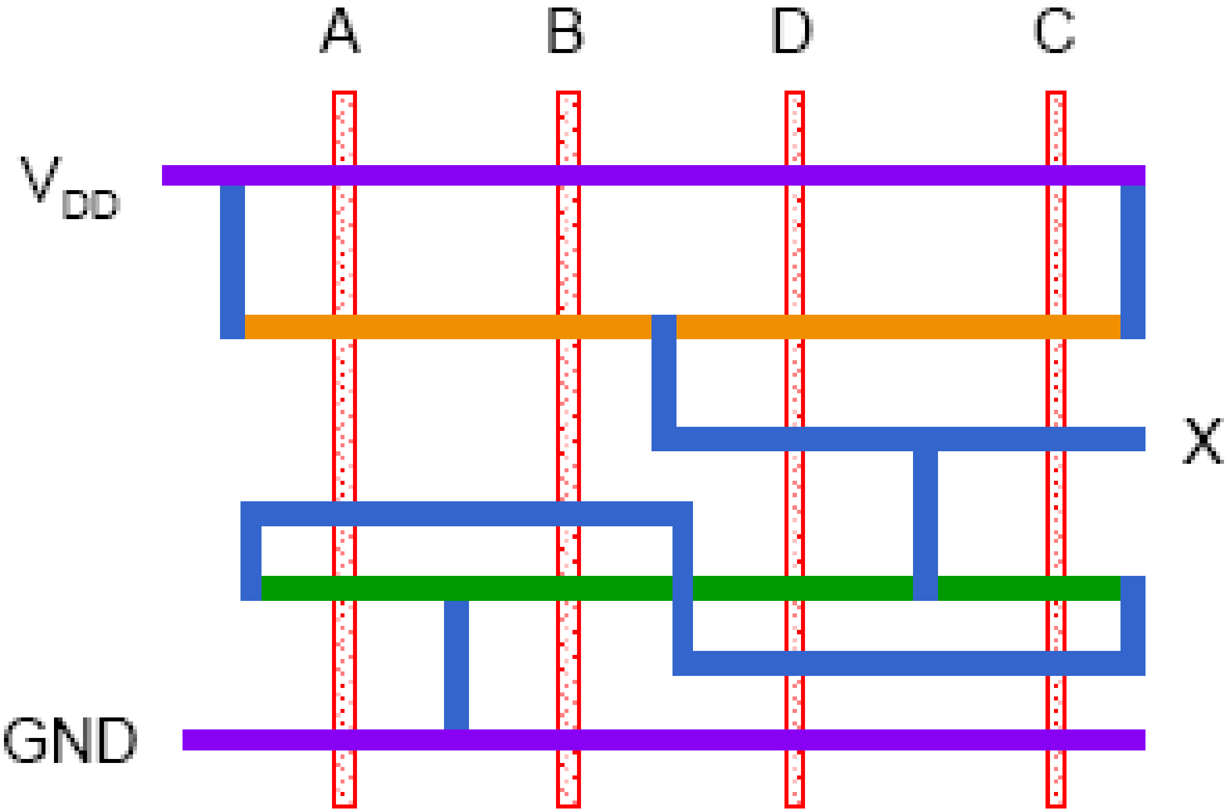
- For a single poly strip for every input signal, the Euler paths in the PUN and PDN must be consistent (the same)

OAI22 Logic Graph



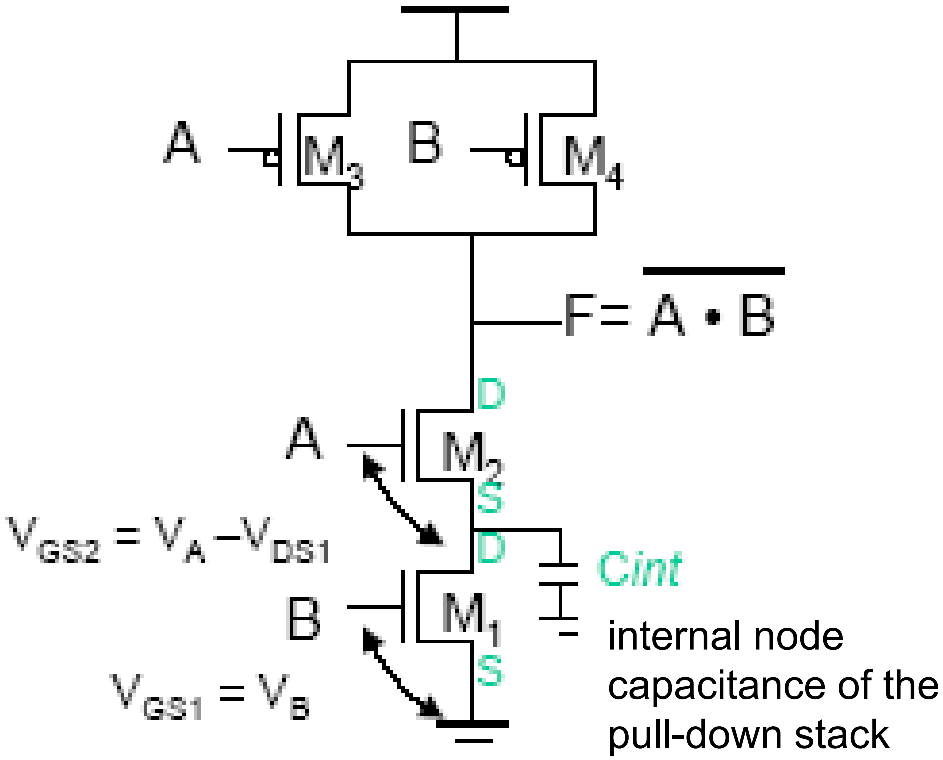


OAI22 Layout

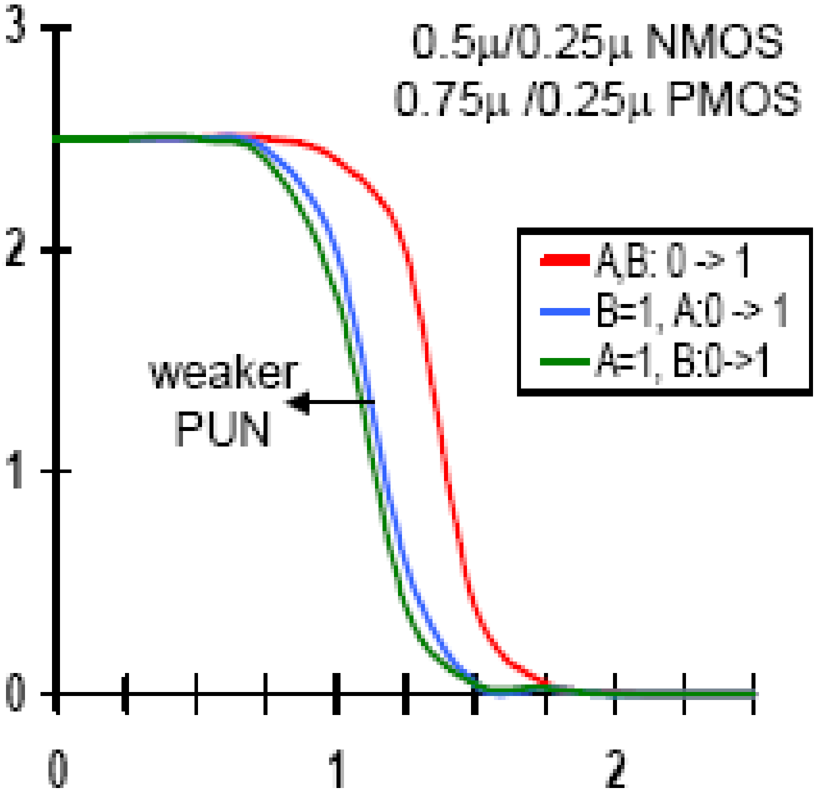


What‘s the logic function ?

VTC is Data-Dependent



internal node capacitance of the pull-down stack

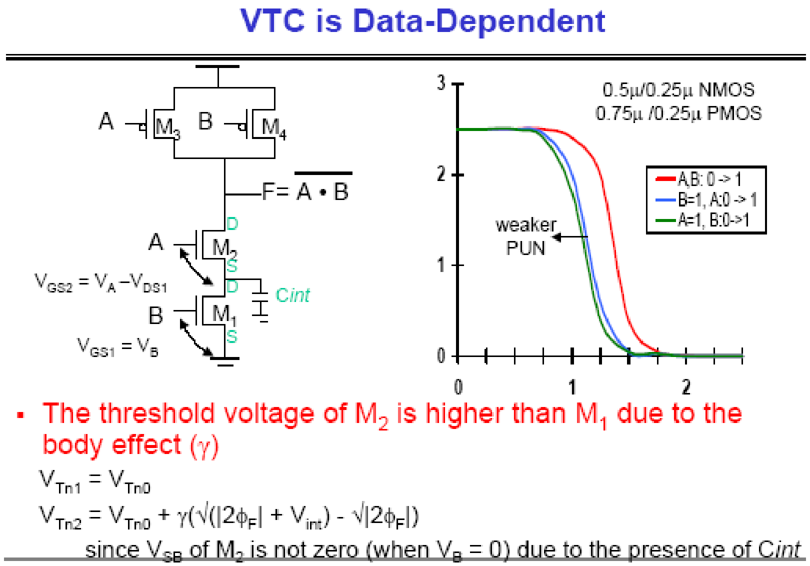


- The threshold voltage of M_2 is higher than M_1 due to the body effect (γ)

$V_{Tn1} = V_{Tn0}$

$V_{Tn2} = V_{Tn0} + \gamma(\sqrt{(|2\phi_F| + V_{int})} - \sqrt{|2\phi_F|})$

since V_{SB} of M_2 is not zero (when $V_B = 0$) due to the presence of *Cint*



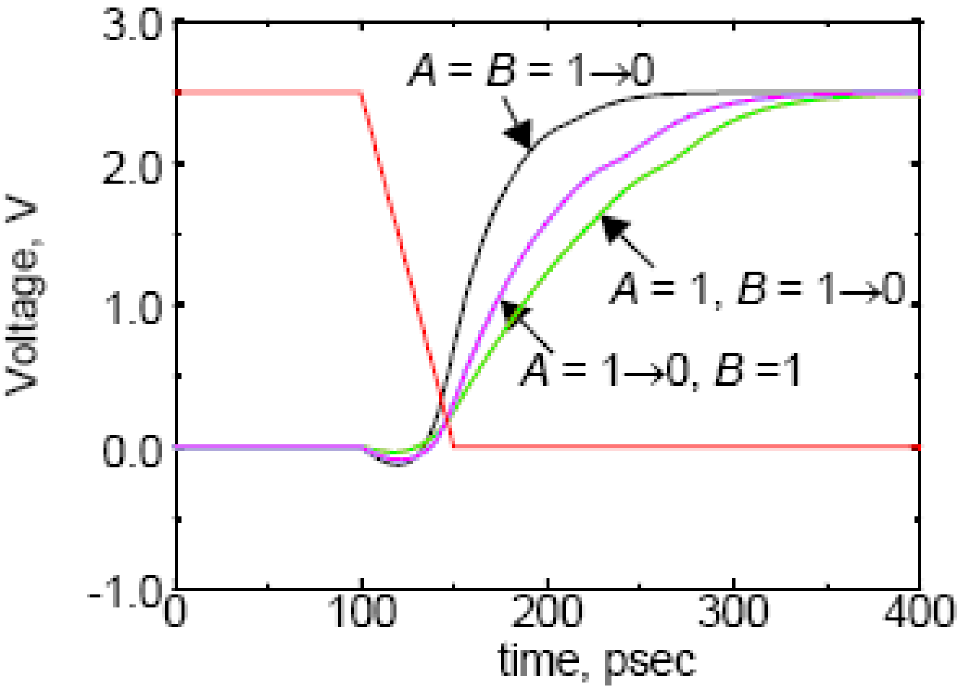
VTC is Data-Dependent

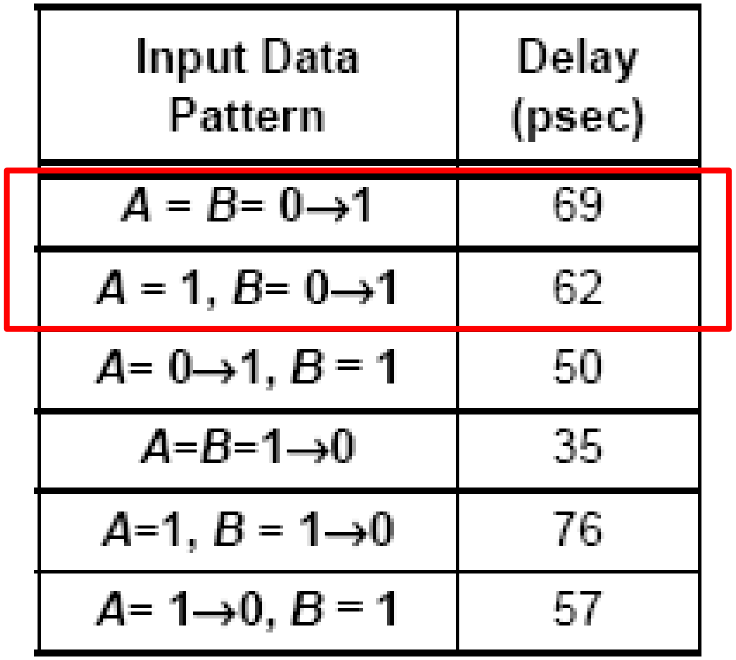
- The threshold voltage of M_2 is higher than M_1 due to the body effect (γ)

$V_{Tn1} = V_{Tn0}$

$V_{Tn2} = V_{Tn0} + \gamma(\sqrt{|2\phi_F| + V_{int}} - \sqrt{|2\phi_F|})$

since V_{SB} of M_2 is not zero (when $V_B = 0$) due to the presence of *Cint*





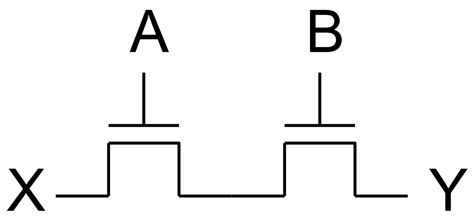
Input Data Pattern	Delay (psec)
A = *B*= 0→1	69
A = 1, *B*= 0→1	62
A= 0→1, *B* = 1	50
A=*B*=1→0	35
A=1, *B* = 1→0	76
A= 1→0, *B* = 1	57

the case where both inputs transition go low (*A = B = 1 →0) results in a smaller* delay

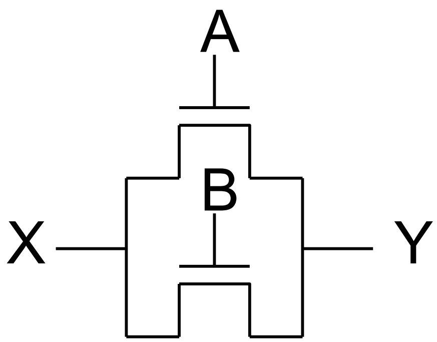
PASS TRANSISTOR LOGIC

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

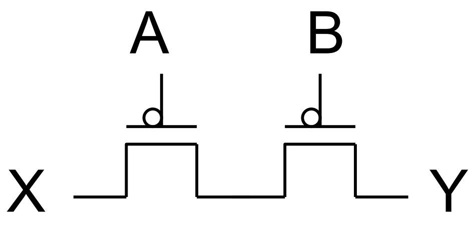


X = Y if A or B

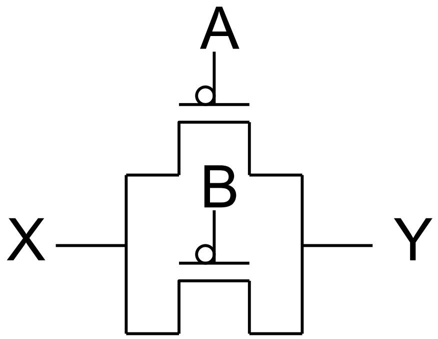
- Remember - NMOS transistors pass a strong 0 but a weak 1

PMOS Transistors in Series/Parallel

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



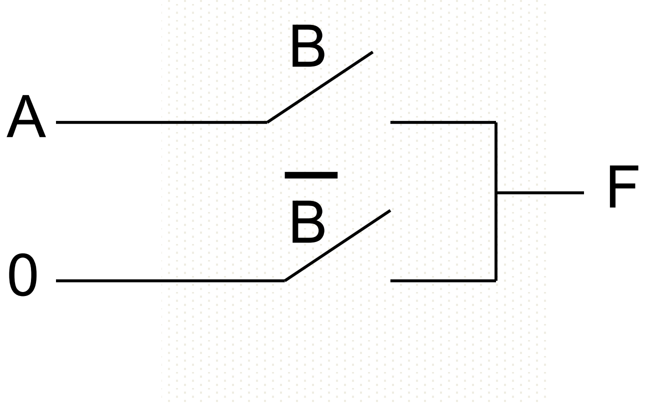
$X = Y \text{ if } \overline{A} \text{ and } \overline{B} = \overline{A + B}$

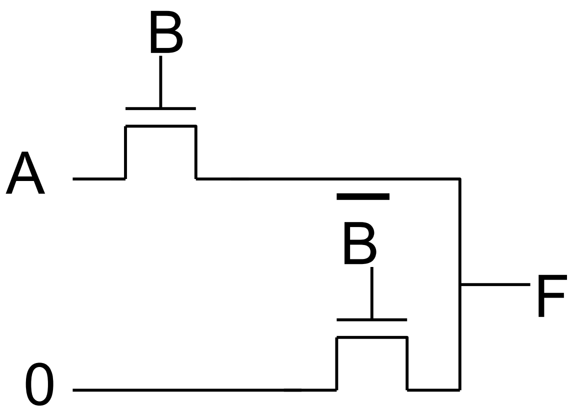


$X = Y \text{ if } \overline{A} \text{ or } \overline{B} = \overline{A \bullet B}$

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

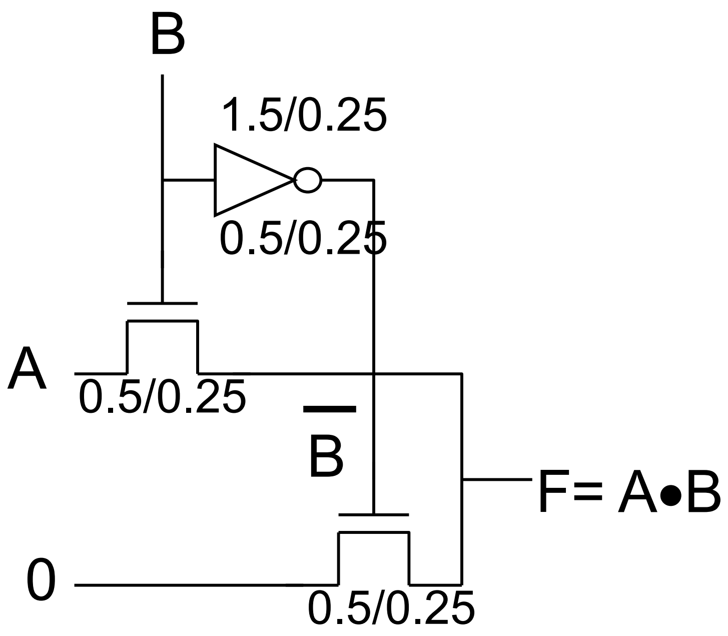
Pass Transistor (PT) Logic

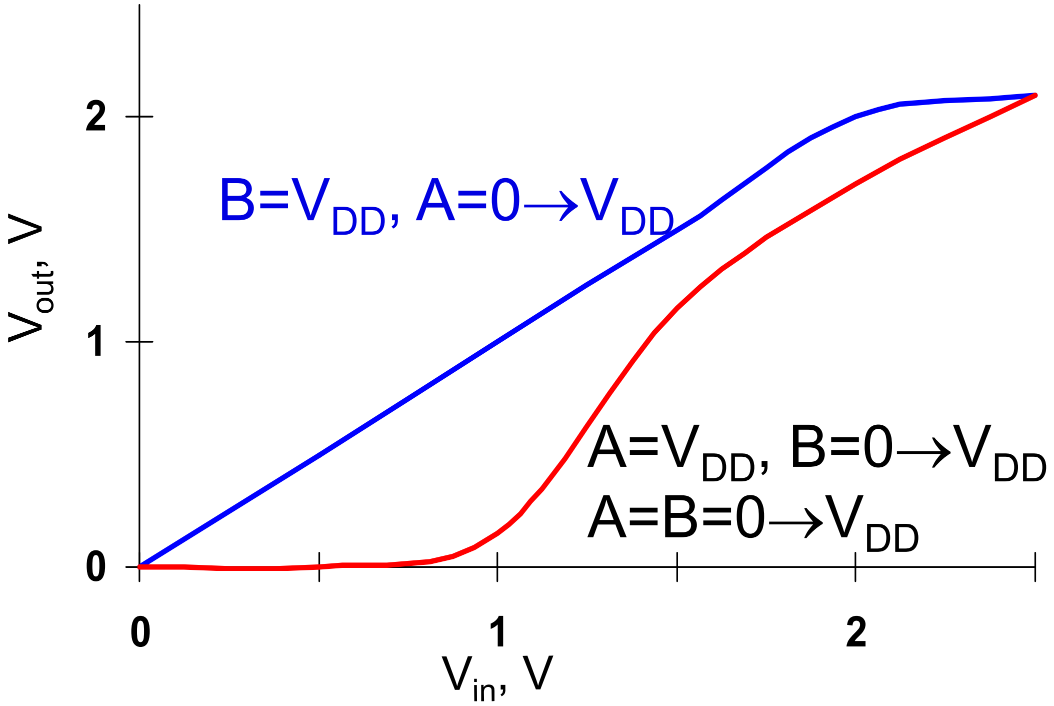




- Gate is static – a low-impedance path exists to both supply rails under all circumstances
- N transistors instead of 2N
- No static power consumption
- Ratioless
- Bidirectional (versus unidirectional)

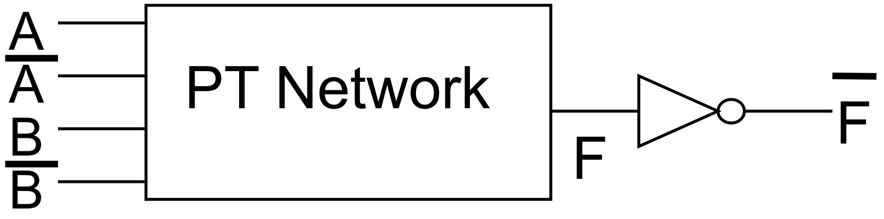
VTC of PT AND Gate

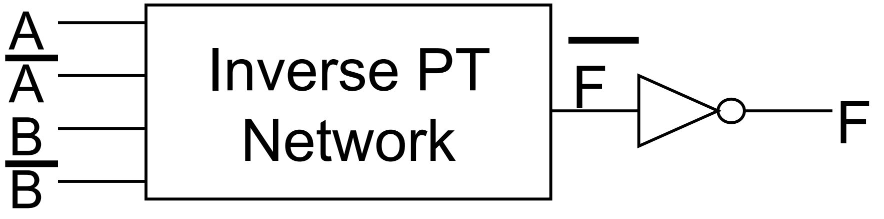


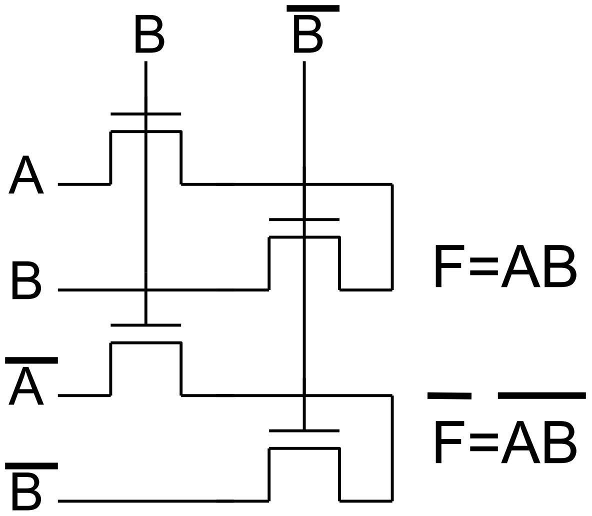


- Pure PT logic is not regenerative - the signal gradually degrades after passing through a number of PTs (can fix with static CMOS inverter insertion)

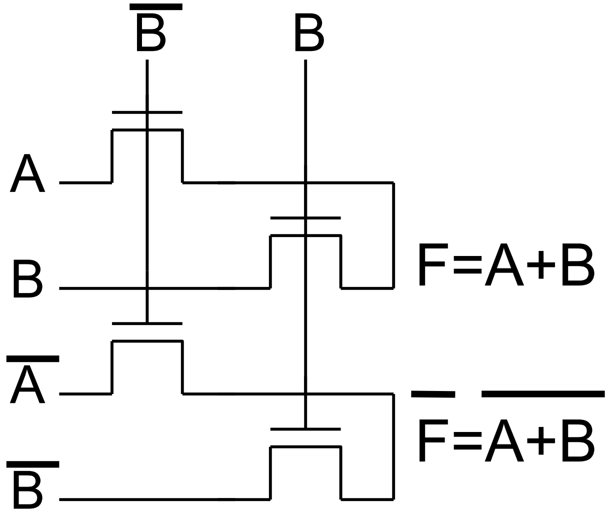
Differential PT Logic (CPL)



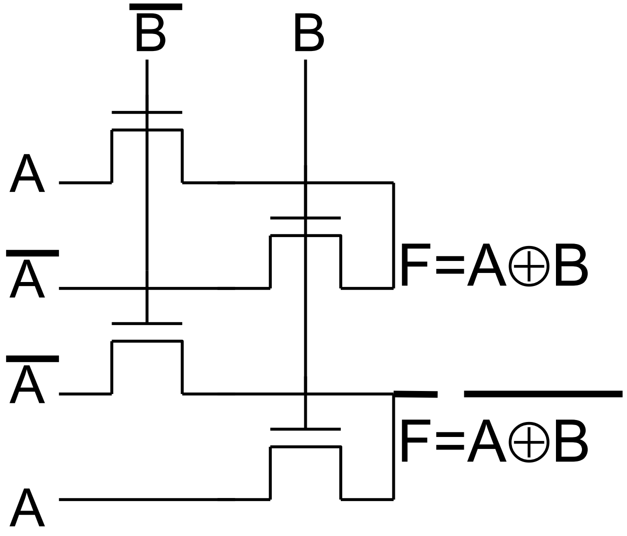




AND/NAND



OR/NOR



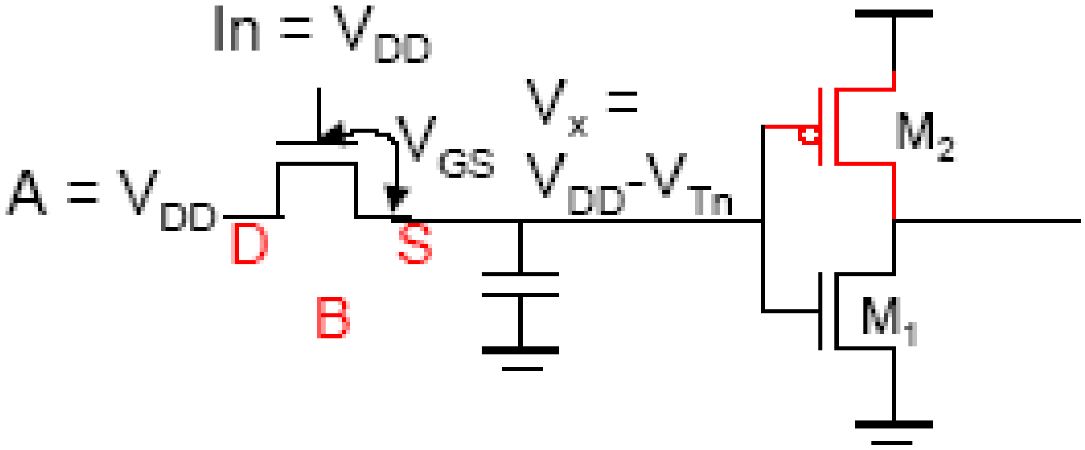
XOR/XNOR

Why not do the same with all pfets??

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

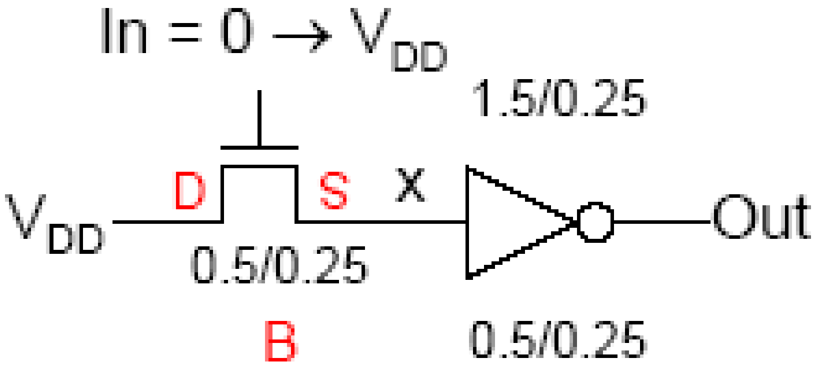
NMOS Only PT Driving an Inverter

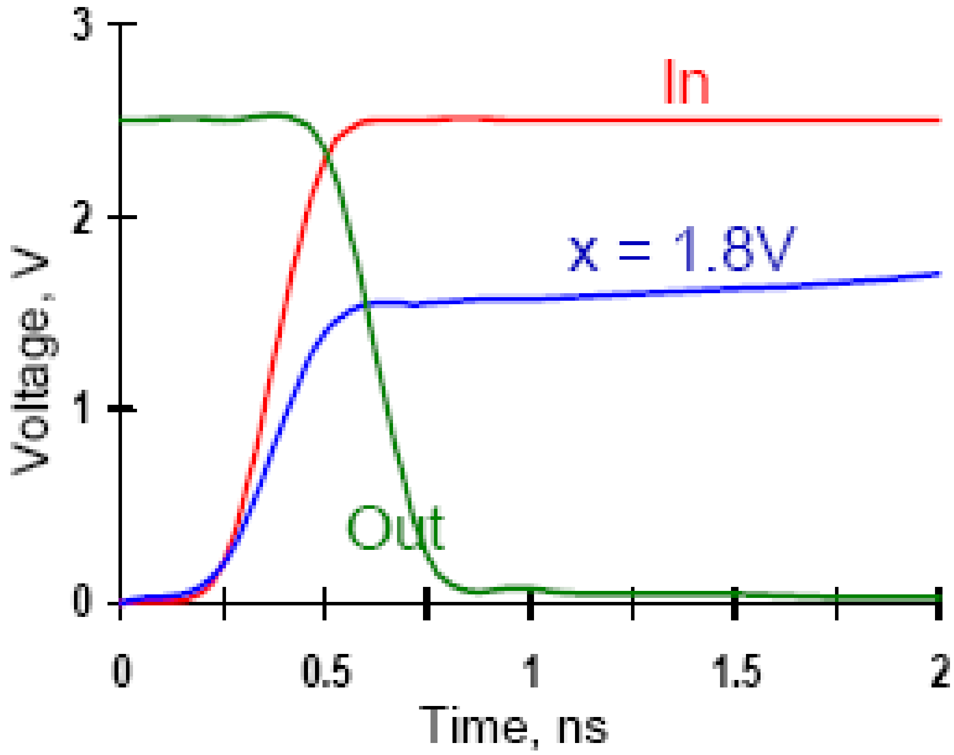


- V_x does not pull up to V_{DD}, but $V_{DD} - V_{Tn}$

- 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

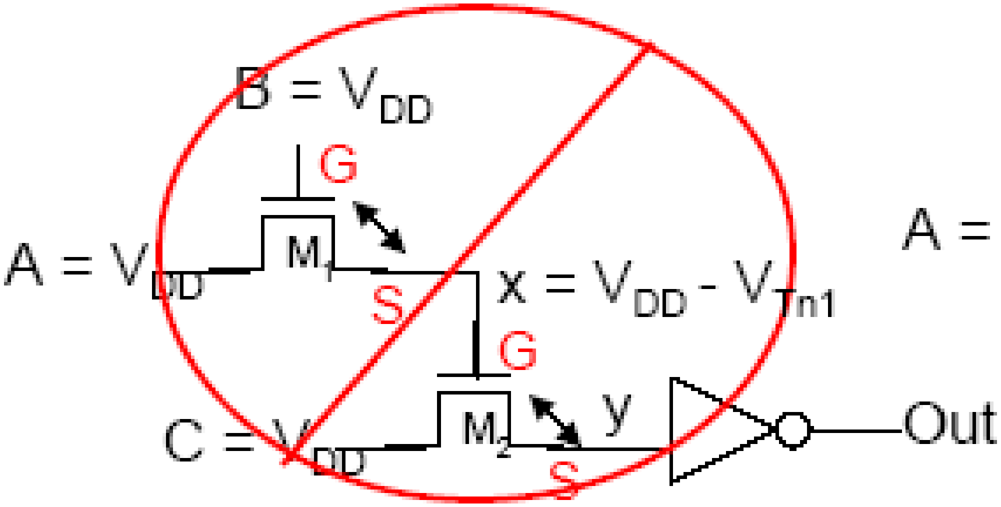


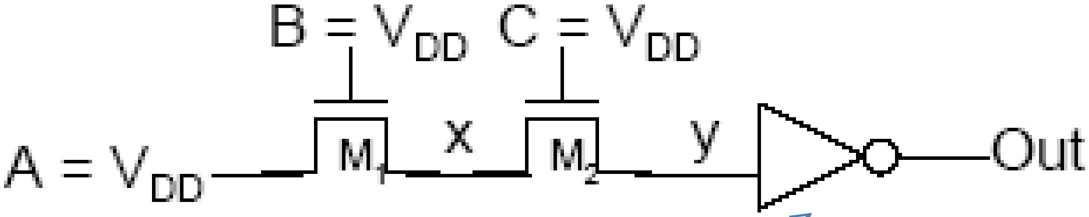


- Body effect – large V_{SB} at x - when pulling high (B is tied to GND and S charged up close to V_{DD})
- So the voltage drop is even worse

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

Cascaded NMOS Only PTs



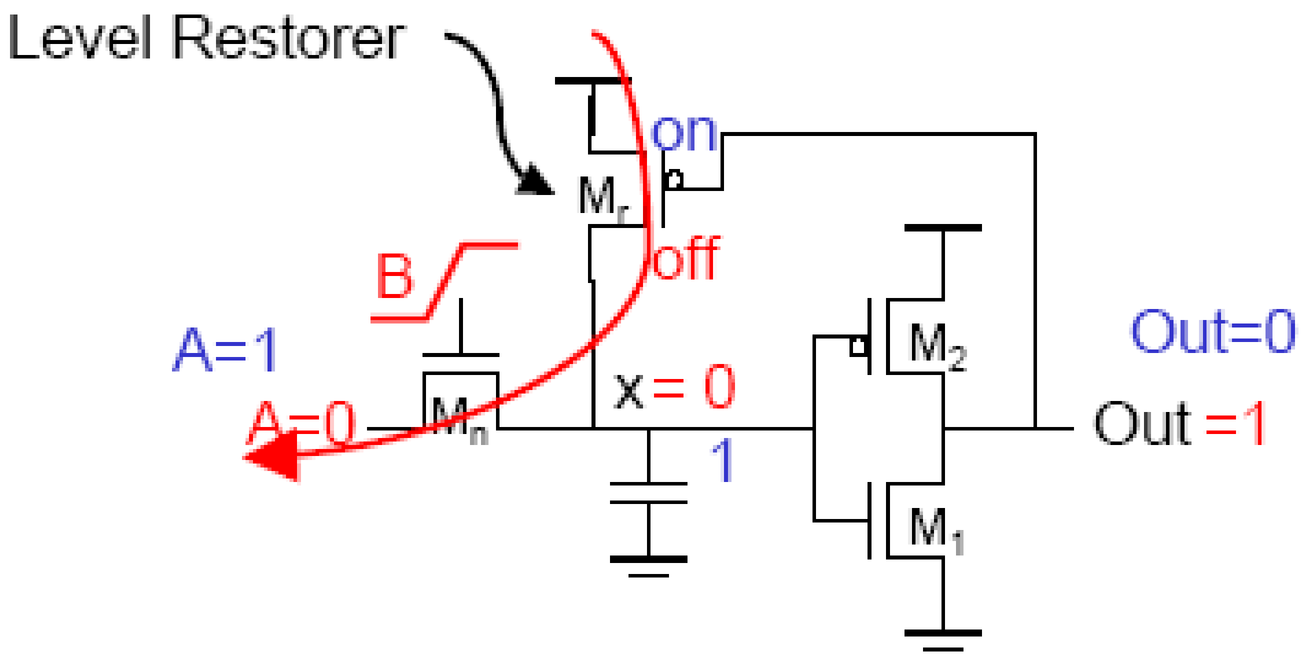


Swing on y = $V_{DD} - V_{Tn1} - V_{Tn2}$

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

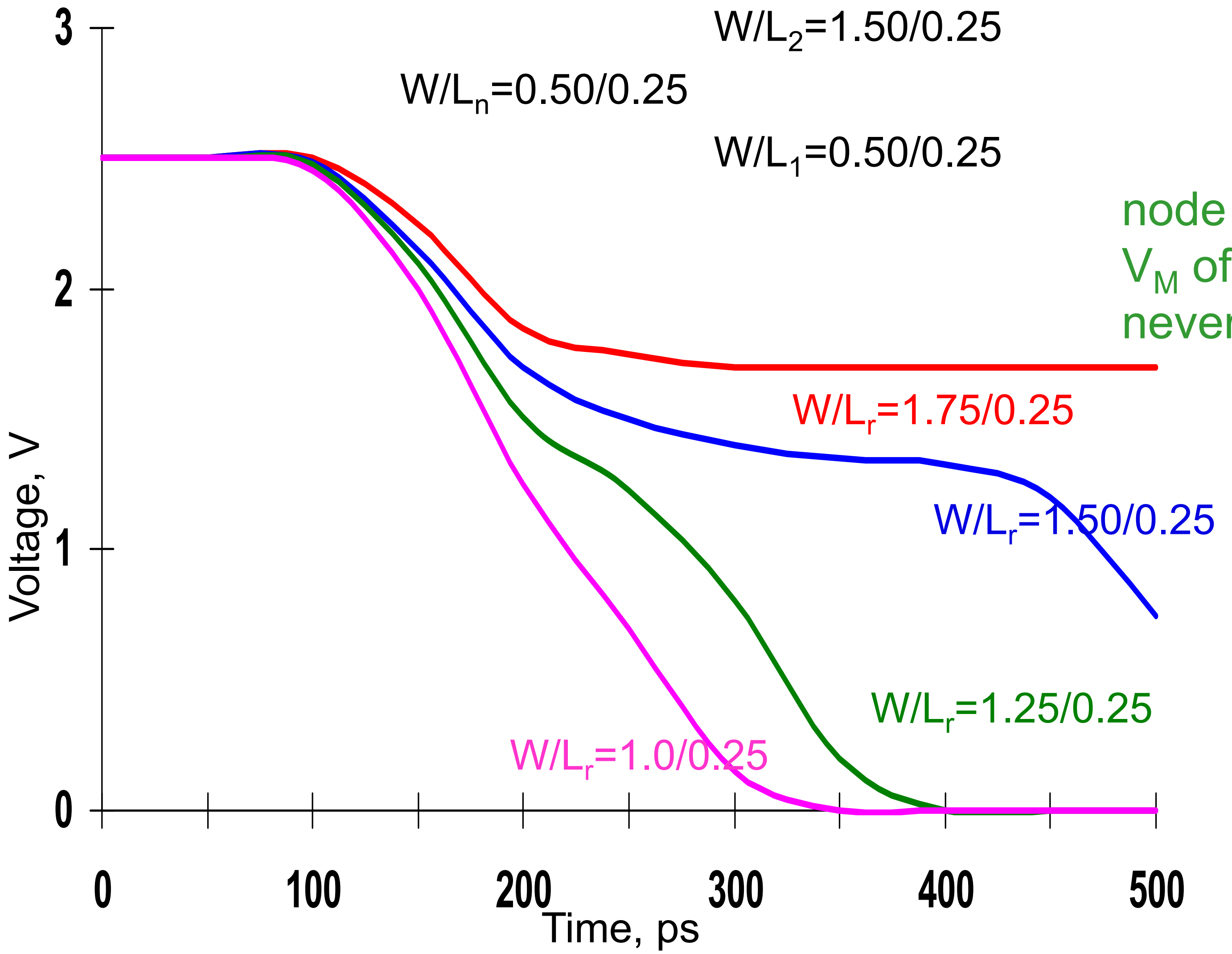
- Pass transistor gates should never be cascaded as on the left
- Logic on the right suffers from static power dissipation and reduced noise margins

Solution 1: Level Restorer



- Full swing on x (due to Level Restorer) so no static power consumption by inverter
- No static backward current path through Level Restorer and PT since Restorer is only active when A is high
- For correct operation M_r must be sized correctly (ratioed)

Transient Level Restorer Circuit Response

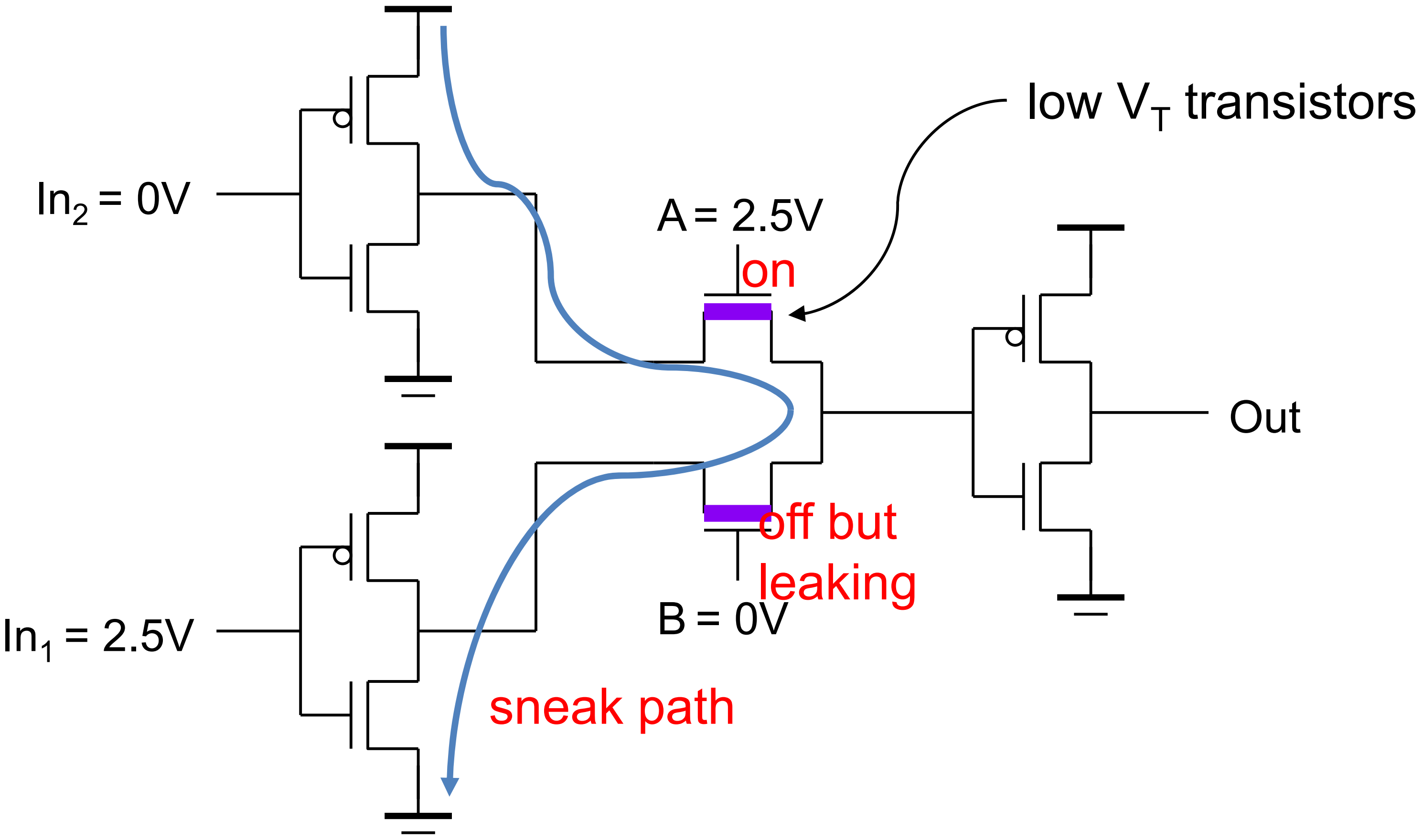


node x never goes below V_M of inverter so output never switches

- Restorer has speed and power impacts: increases the capacitance at x, slowing down the gate; increases t_r (but decreases t_f)

Solution 2: Multiple V_T Transistors

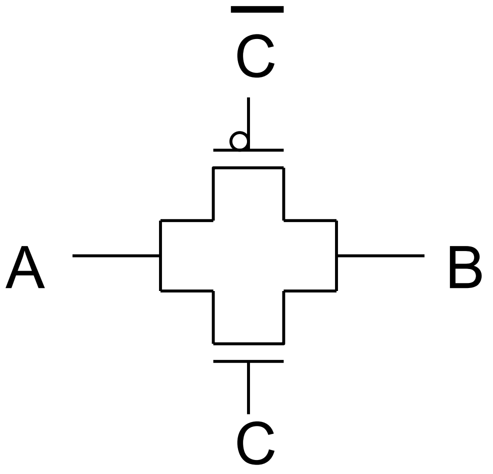
- Technology solution: Use (near) **zero V_T devices** for the NMOS PTs to eliminate *most* of the threshold drop (body effect still in force preventing full swing to V_{DD})

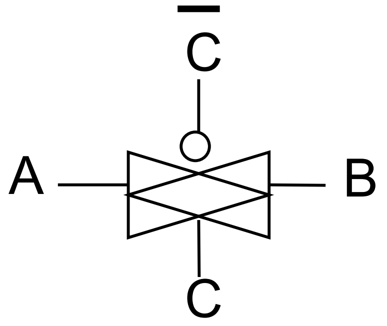


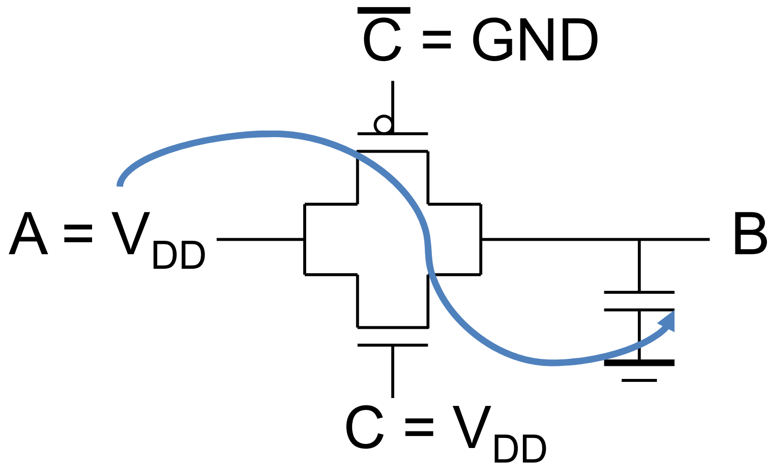
❑ **Impacts static power** consumption due to subthreshold currents flowing through the PTs (even if V_{GS} is below V_T)

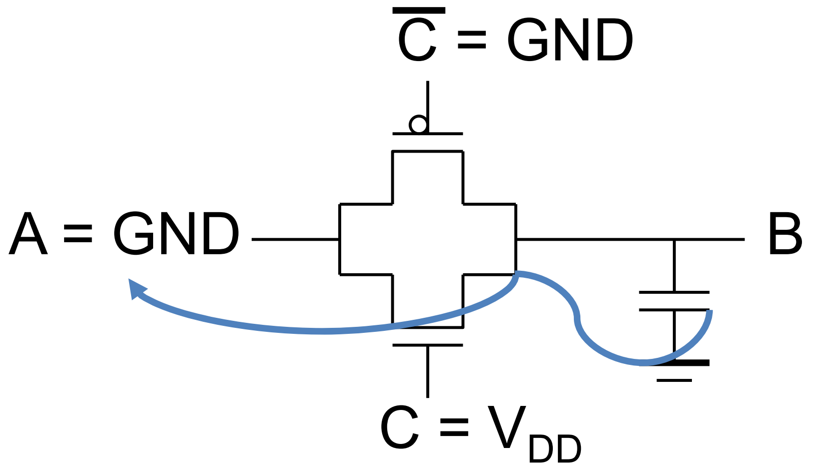
Solution 3: Transmission Gates (TGs)

- Most widely used solution



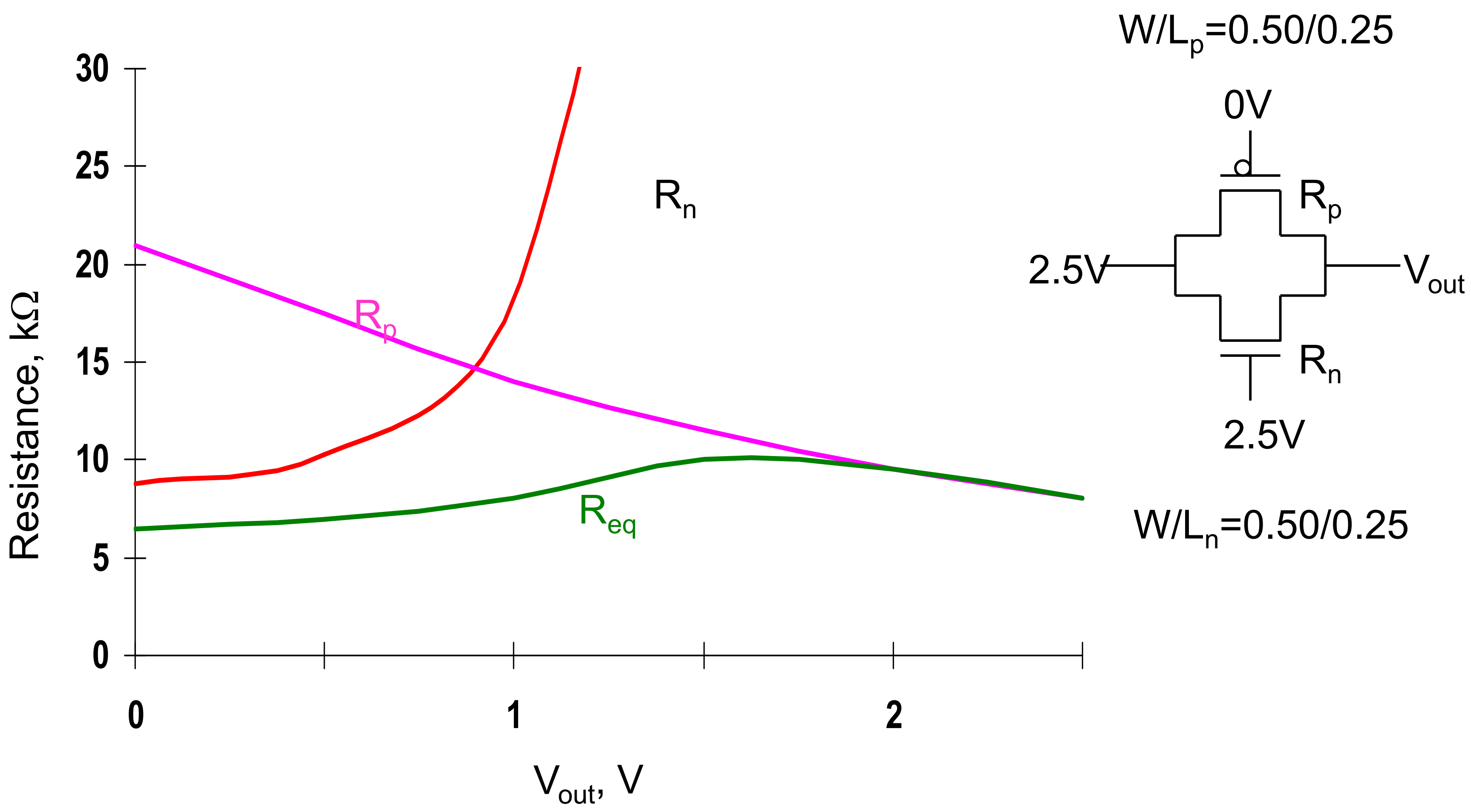






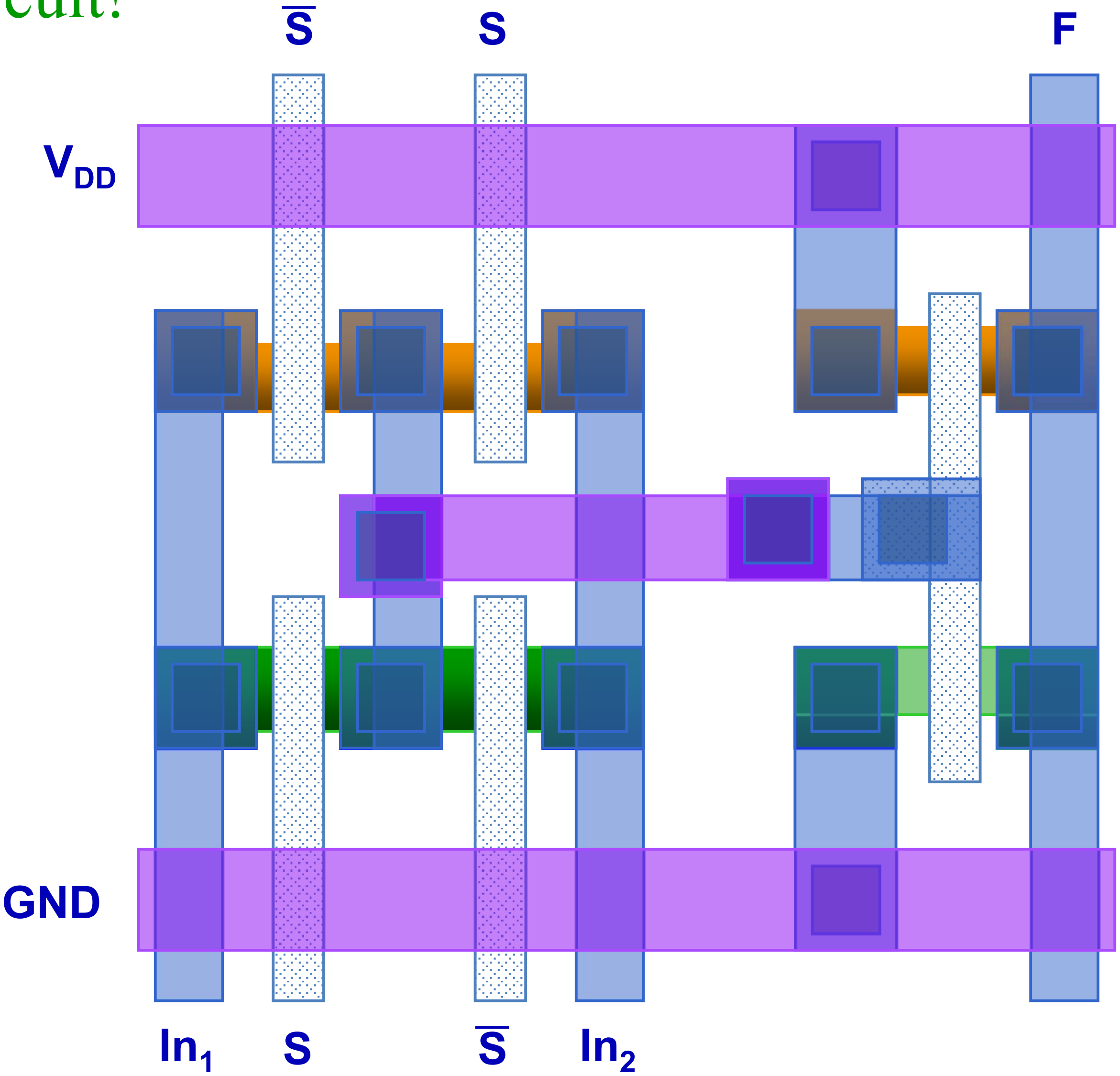
- Full swing *bidirectional* switch controlled by the gate signal C, A = B if C = 1

Resistance of TG

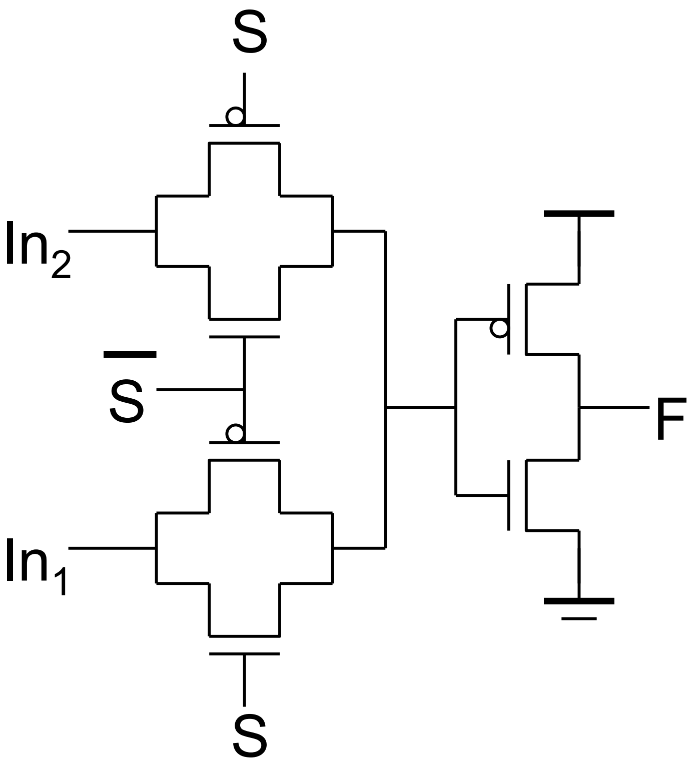


TG Multiplexer

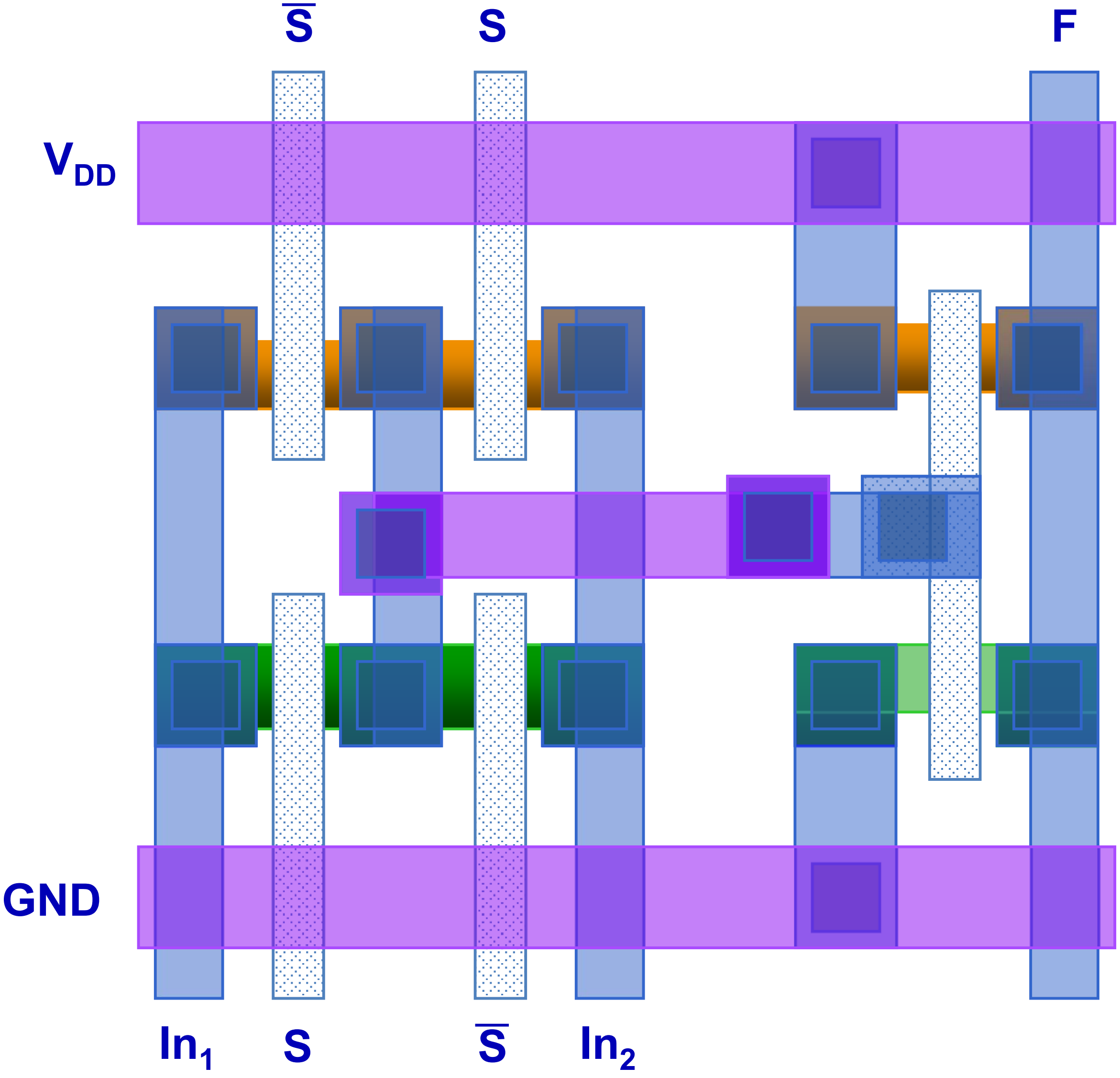
Draw the transistor level circuit!



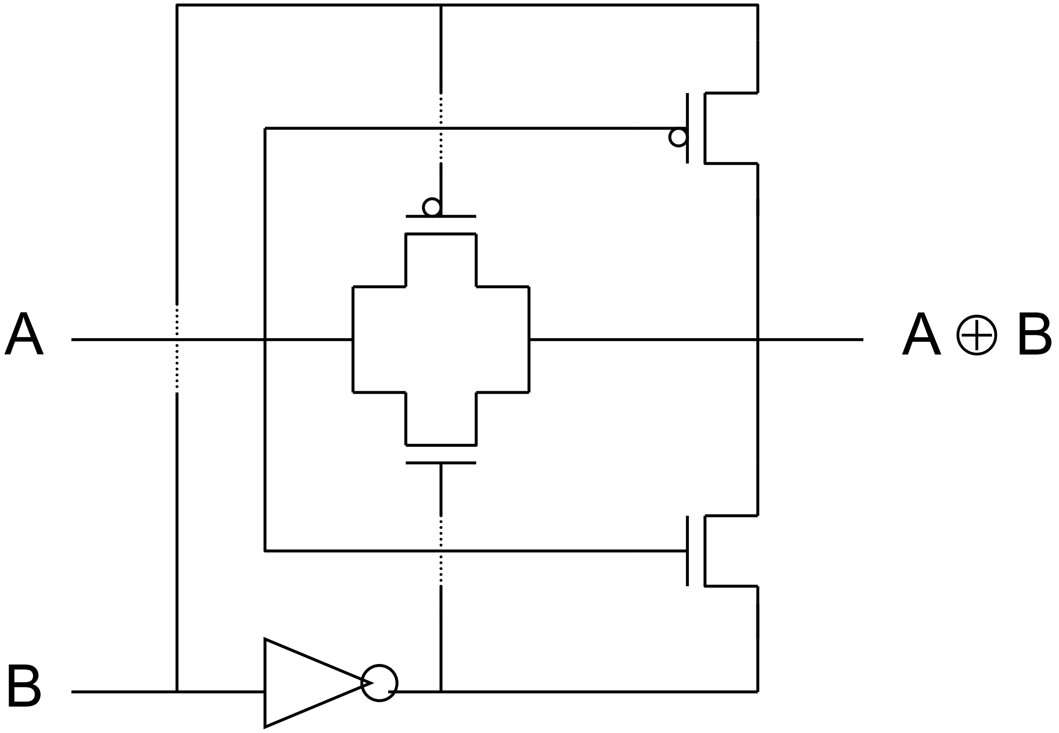
TG Multiplexer



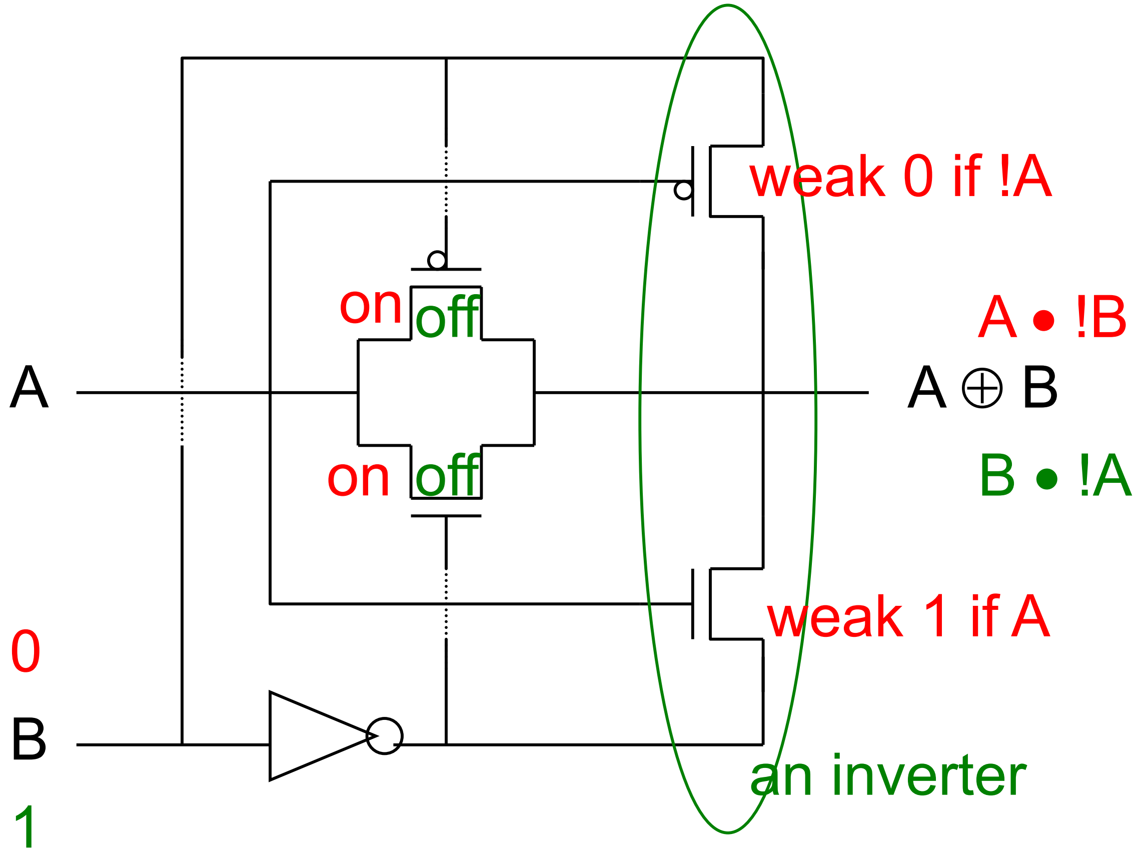
$F = !(In_1 \bullet S + In_2 \bullet \overline{S})$



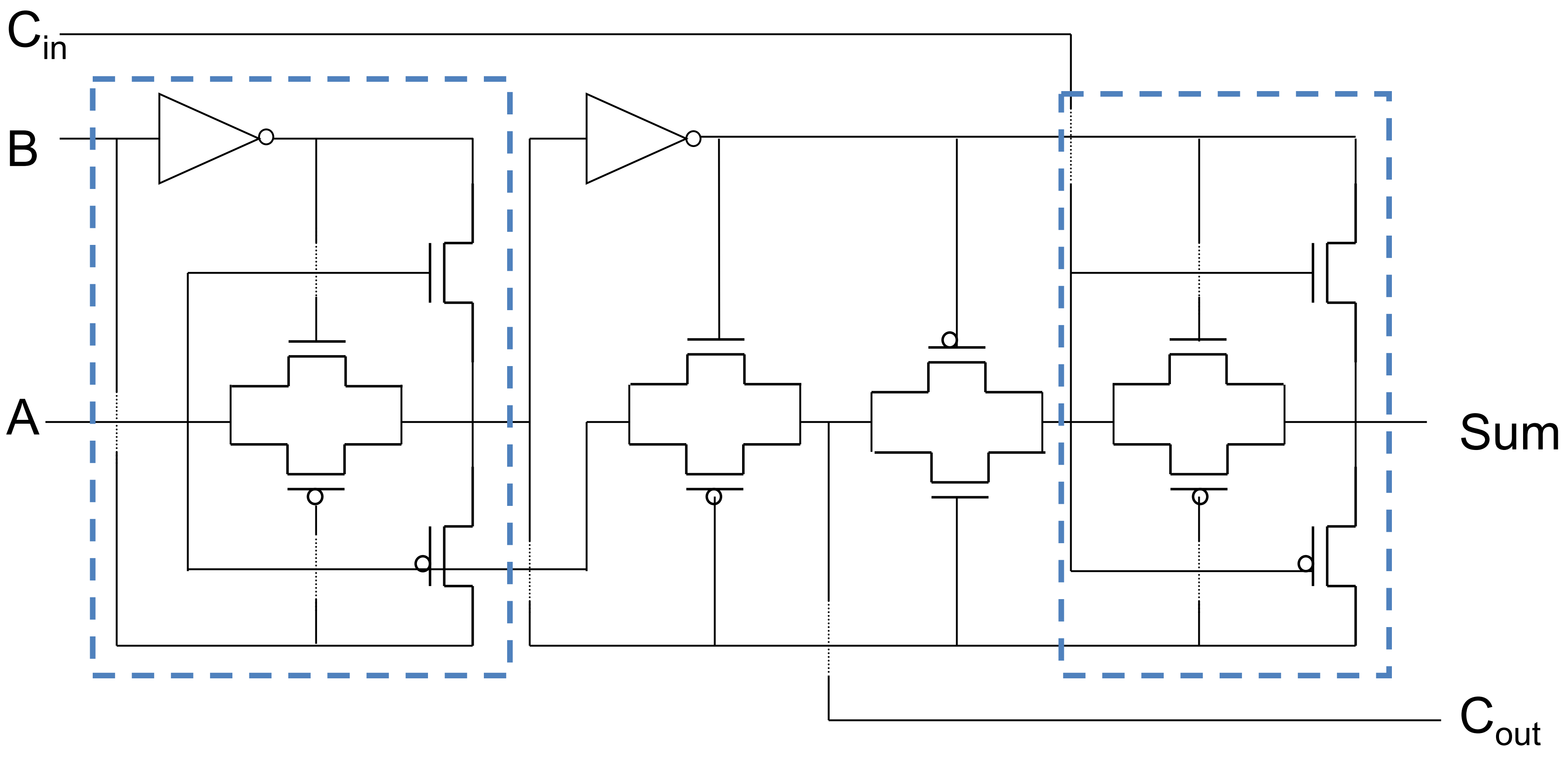
Transmission Gate XOR



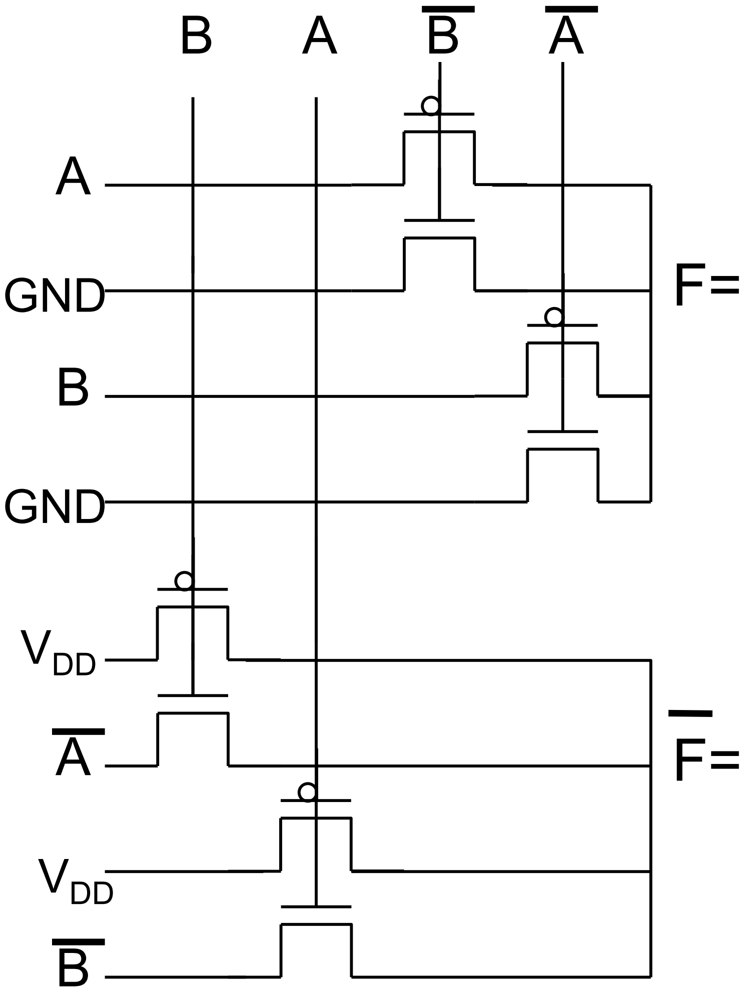
Transmission Gate XOR



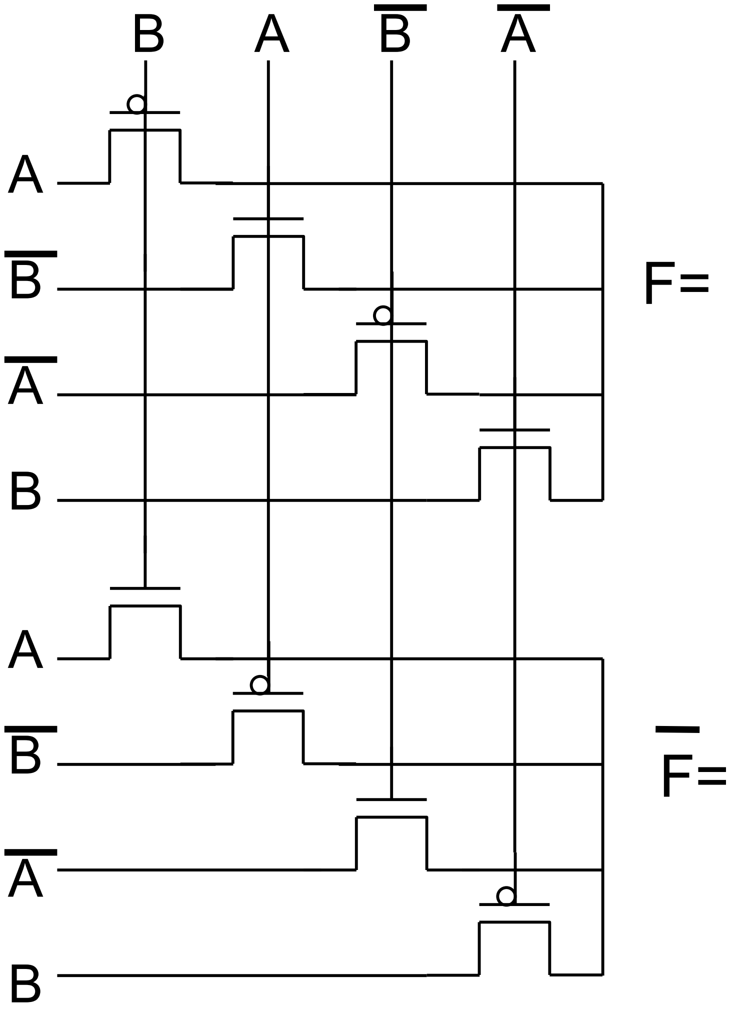
TG Full Adder



Differential TG Logic (DPL)

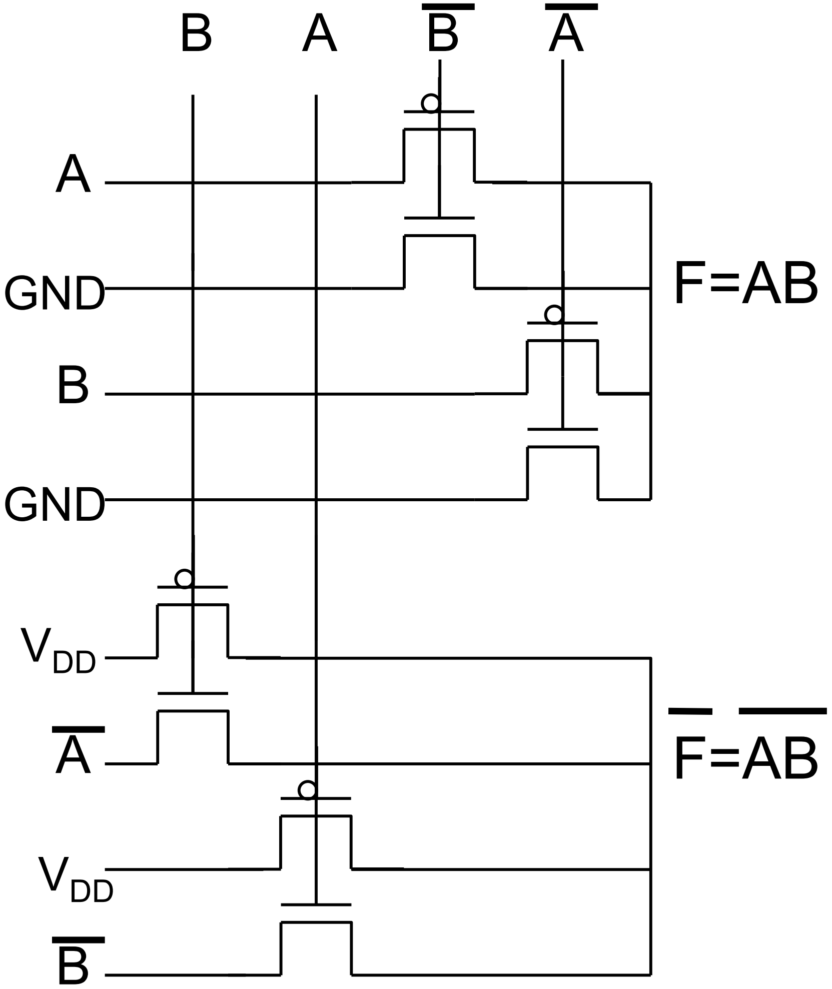


AND/NAND

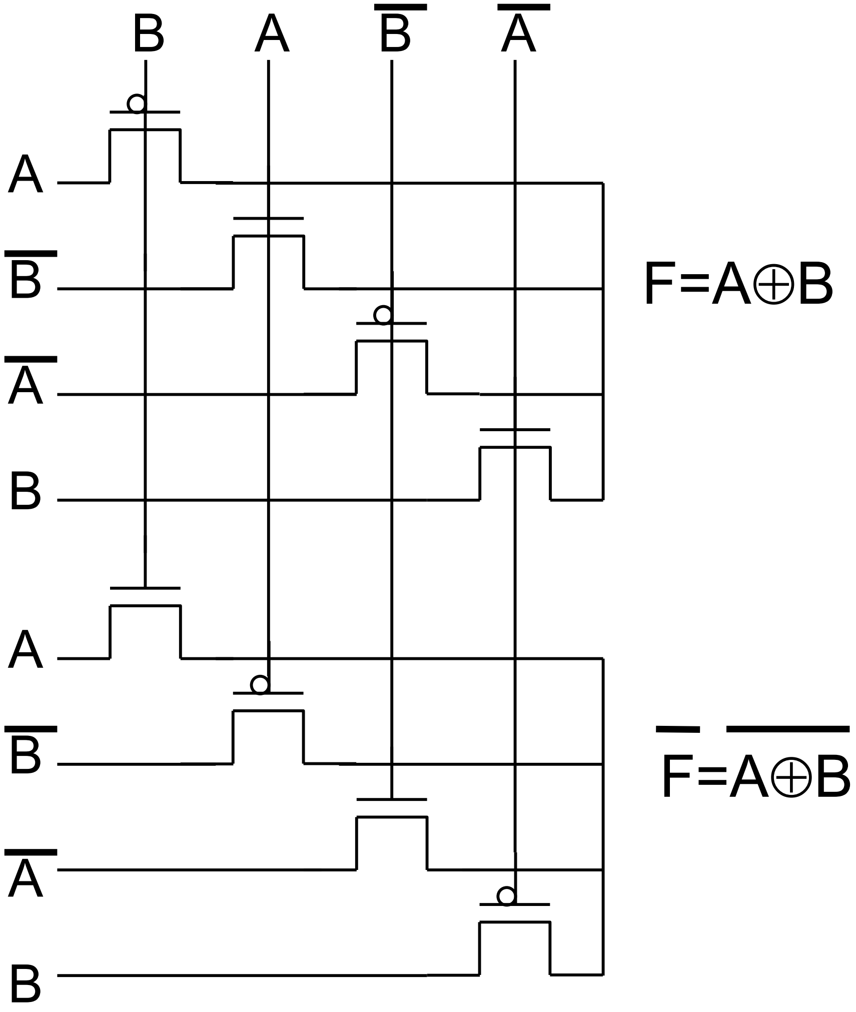


XOR/XNOR

Differential TG Logic (DPL)



AND/NAND



XOR/XNOR

VLSI Design I
Delay Estimation

Review: CMOS Circuit Styles

- Static complementary CMOS - except during switching, output connected to either VDD or GND via a low-resistance path
  - high noise margins
    - full rail to rail swing
    - VOH and VOL are at VDD and GND, respectively
  - low output impedance, high input impedance
  - no steady state path between VDD and GND (no static power consumption)
  - delay a function of load capacitance and transistor resistance
  - 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
  - simpler, faster gates
  - increased sensitivity to noise

Transient Response

- *DC analysis* tells us V_{out} if V_{in} is constant
- *Transient analysis* tells us $V_{out}(t)$ if $V_{in}(t)$ changes
  - Requires solving differential equations
- Input is usually considered to be a step or ramp
  - From 0 to V_{DD} or vice versa

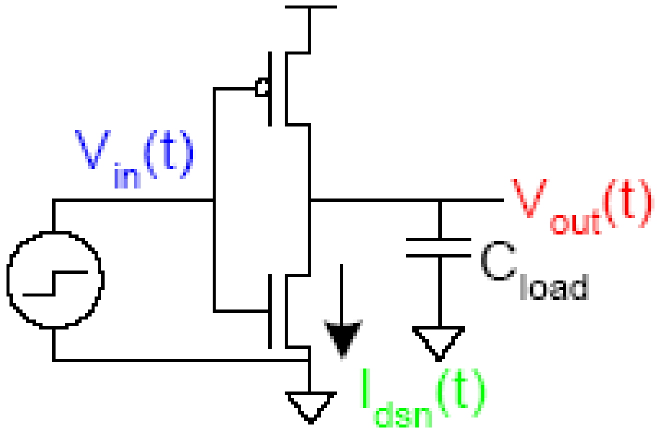
Inverter Step Response

- Ex: find step response of inverter driving load cap

$$V_{in}(t) = u(t - t_0)V_{DD}$$

$$V_{out}(t < t_0) = V_{DD}$$

$$\frac{dV_{out}(t)}{dt} =$$



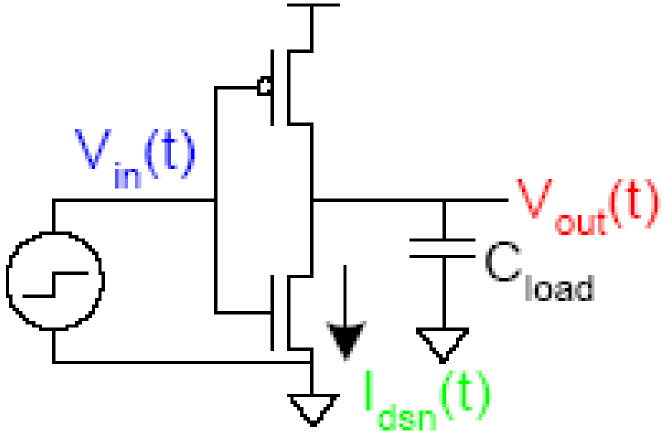
Inverter Step Response

- Ex: find step response of inverter driving load cap

$$V_{in}(t) = u(t - t_0)V_{DD}$$

$$V_{out}(t < t_0) = V_{DD}$$

$$\frac{dV_{out}(t)}{dt} = -\frac{I_{dsn}(t)}{C_{load}}$$



$$I_{dsn}(t) = \begin{cases} & t \le t_0 \\ & V_{out} > V_{DD} - V_t \\ & V_{out} < V_{DD} - V_t \end{cases}$$

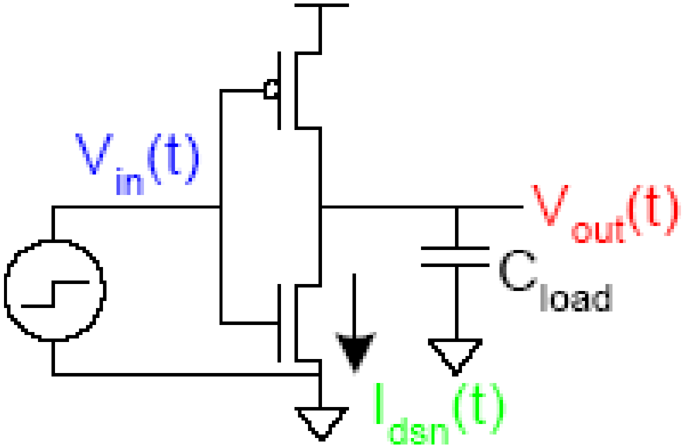
Inverter Step Response

- Ex: find step response of inverter driving load cap

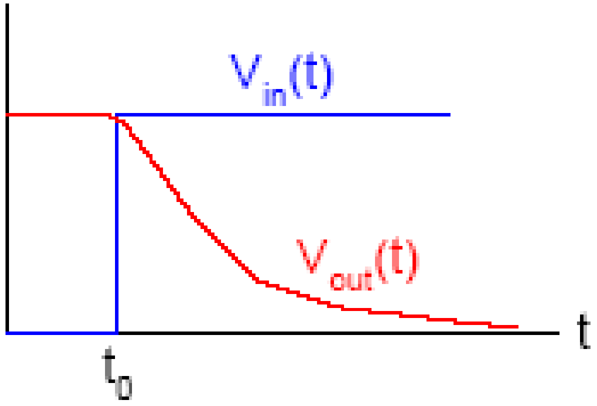
$$V_{in}(t) = u(t - t_0)V_{DD}$$

$$V_{out}(t < t_0) = V_{DD}$$

$$\frac{dV_{out}(t)}{dt} = -\frac{I_{dsn}(t)}{C_{load}}$$



$$I_{dsn}(t) = \begin{cases} 0 & t \le t_0 \\ \frac{\beta}{2}\left(V_{DD} - V\right)^2 & V_{out} > V_{DD} - V_t \\ \beta\left(V_{DD} - V_t - \frac{V_{out}(t)}{2}\right)V_{out}(t) & V_{out} < V_{DD} - V_t \end{cases}$$

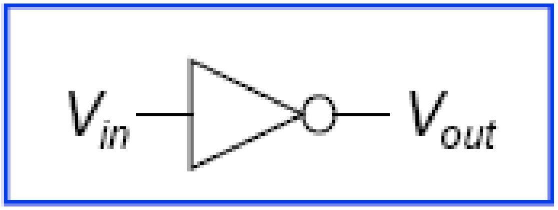


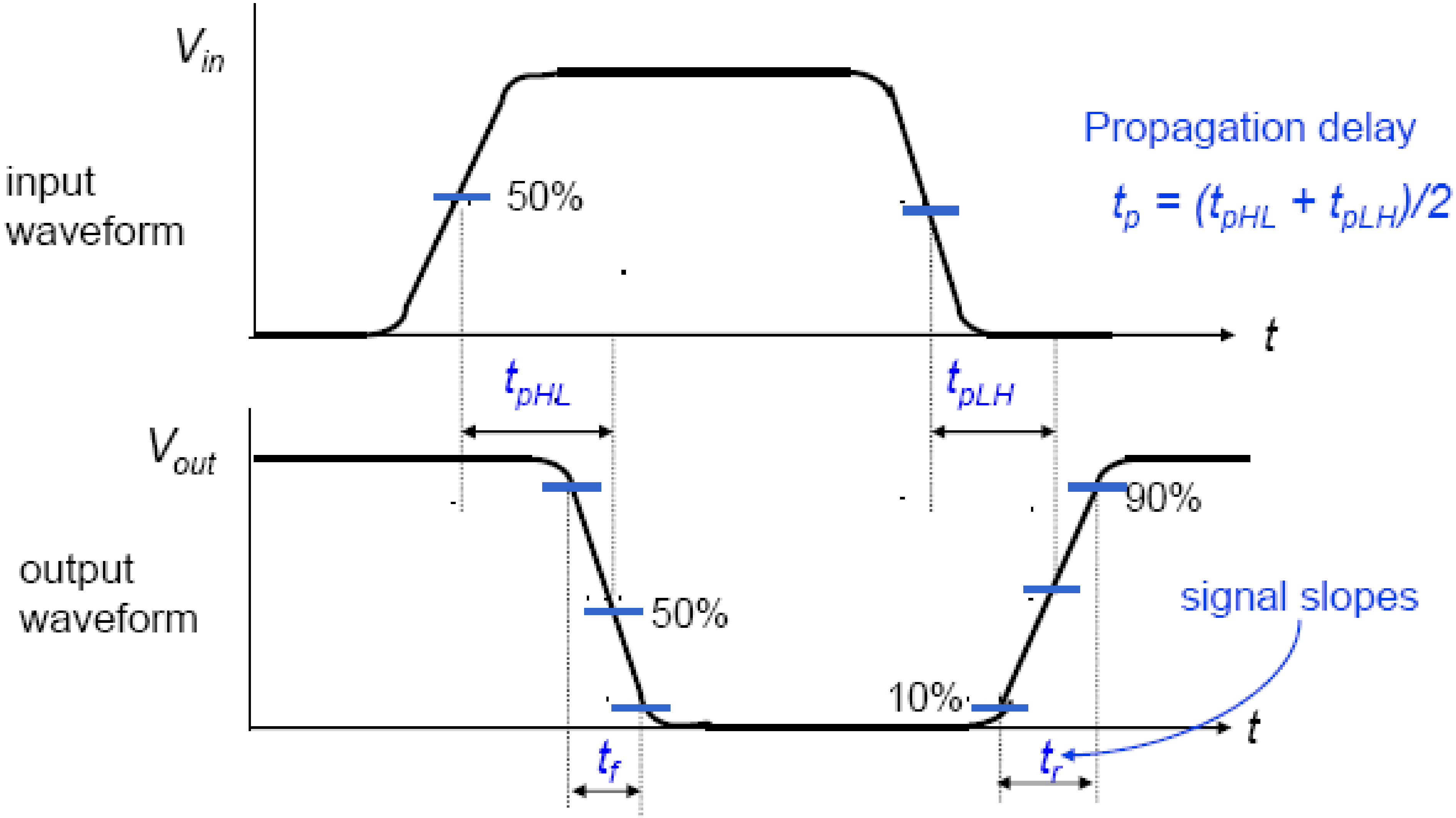
$$\beta = k'_n \frac{W}{L} \qquad k'_n = \mu_n C_{ox} = \frac{\mu_n \varepsilon_{ox}}{t_{ox}}$$

Delay Definitions

- $\mathbf{t_{pdr}}$: *rising propagation delay*
  - From input to rising output crossing $V_{DD}/2$
- $\mathbf{t_{pdf}}$: *falling propagation delay*
  - From input to falling output crossing $V_{DD}/2$
- $\mathbf{t_{pd}}$: *average propagation delay*
  - $t_{pd} = (t_{pdr} + t_{pdf})/2$
- $\mathbf{t_{r}}$: *rise time*
  - From output crossing 0.2 V_{DD} to 0.8 V_{DD}
- $\mathbf{t_{f}}$: *fall time*
  - From output crossing 0.8 V_{DD} to 0.2 V_{DD}

Delay Definitions





Propagation delay

$t_p = (t_{pHL} + t_{pLH})/2$

signal slopes

Delay Definitions

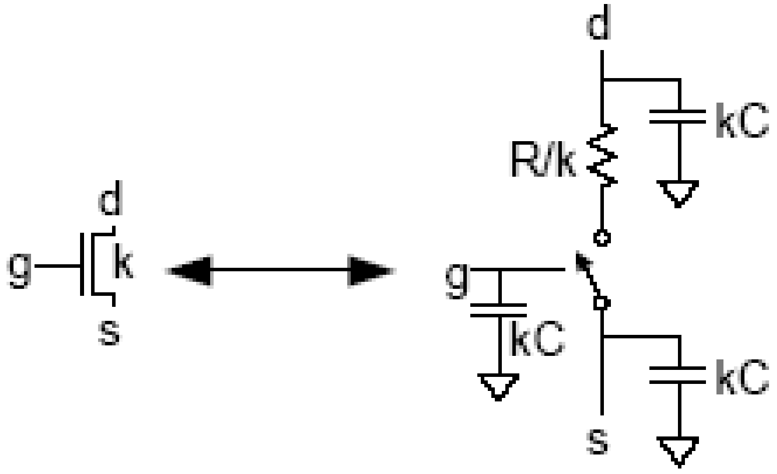
- $\mathbf{t_{cdr}}$: *rising contamination delay*
  - From input to rising output crossing $V_{DD}/2$
- $\mathbf{t_{cdf}}$: *falling contamination delay*
  - From input to falling output crossing $V_{DD}/2$
- $\mathbf{t_{cd}}$: *average contamination delay*
  - $t_{pd} = (t_{cdr} + t_{cdf})/2$

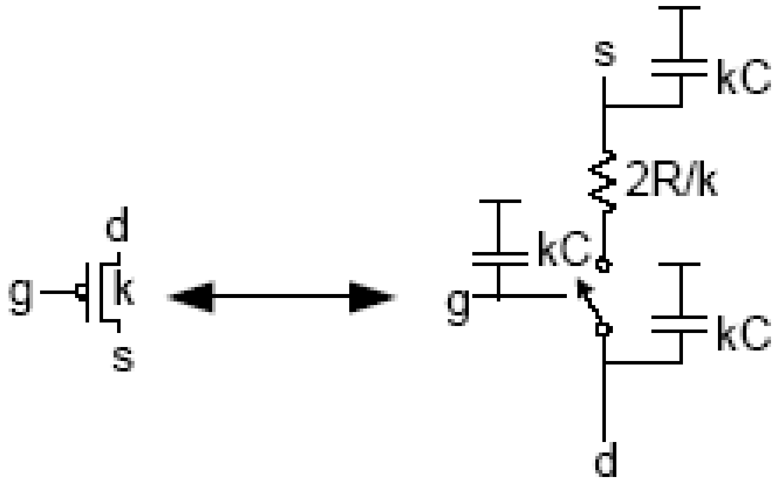
Delay Estimation

- We would like to be able to easily estimate delay
  - Not as accurate as simulation
  - But easier to ask "What if?"
- The step response usually looks like a 1st order RC response with a decaying exponential.
- Use RC delay models to estimate delay
  - C = total capacitance on output node
  - Use *effective resistance* R
  - So that $t_{pd} = RC$
- Characterize transistors by finding their effective R
  - Depends on average current as gate switches

RC Delay Models

- Use equivalent circuits for MOS transistors
  - Ideal switch + capacitance and ON resistance
  - Unit nMOS has resistance R, capacitance C
  - Unit pMOS has resistance 2R, capacitance C
- Capacitance proportional to width
- Resistance inversely proportional to width



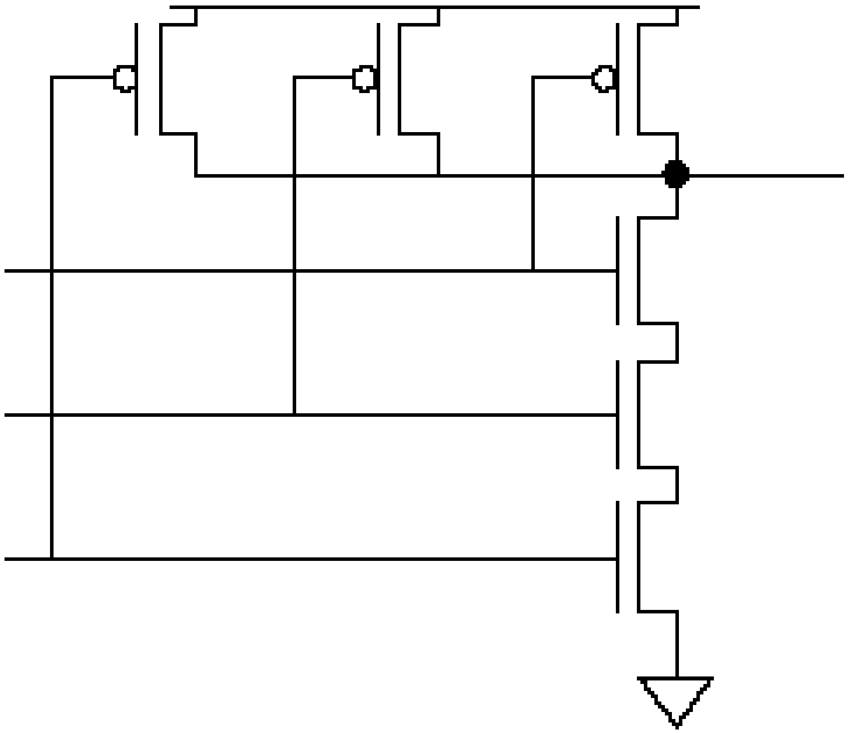


Example: 3-input NAND

- Sketch a 3-input NAND with transistor widths chosen to achieve effective rise and fall resistances equal to a unit inverter (R).

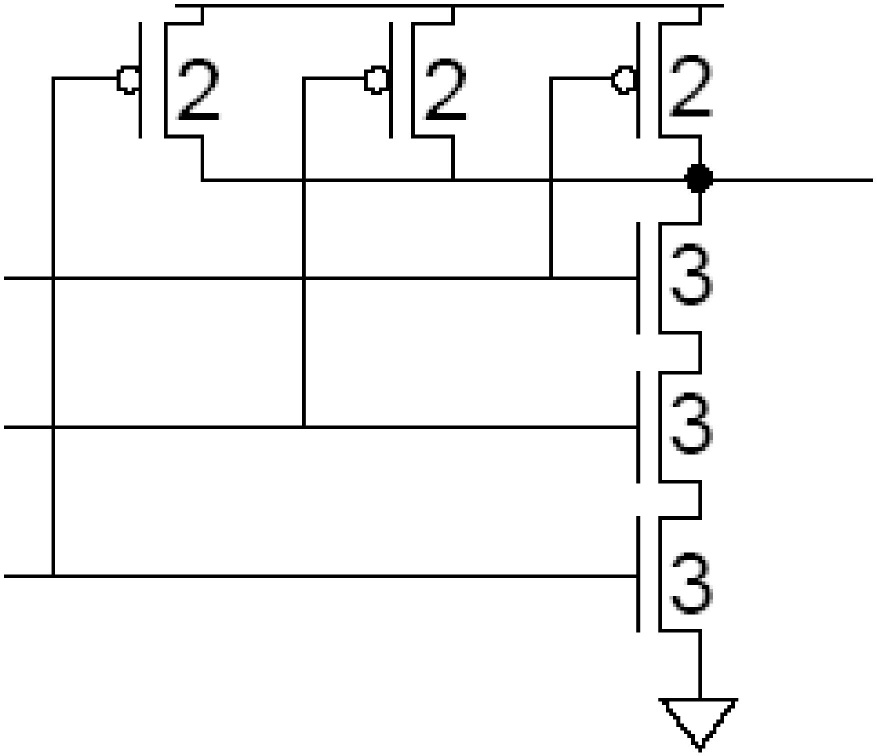
Example: 3-input NAND

- Sketch a 3-input NAND with transistor widths chosen to achieve effective rise and fall resistances equal to a unit inverter (R).



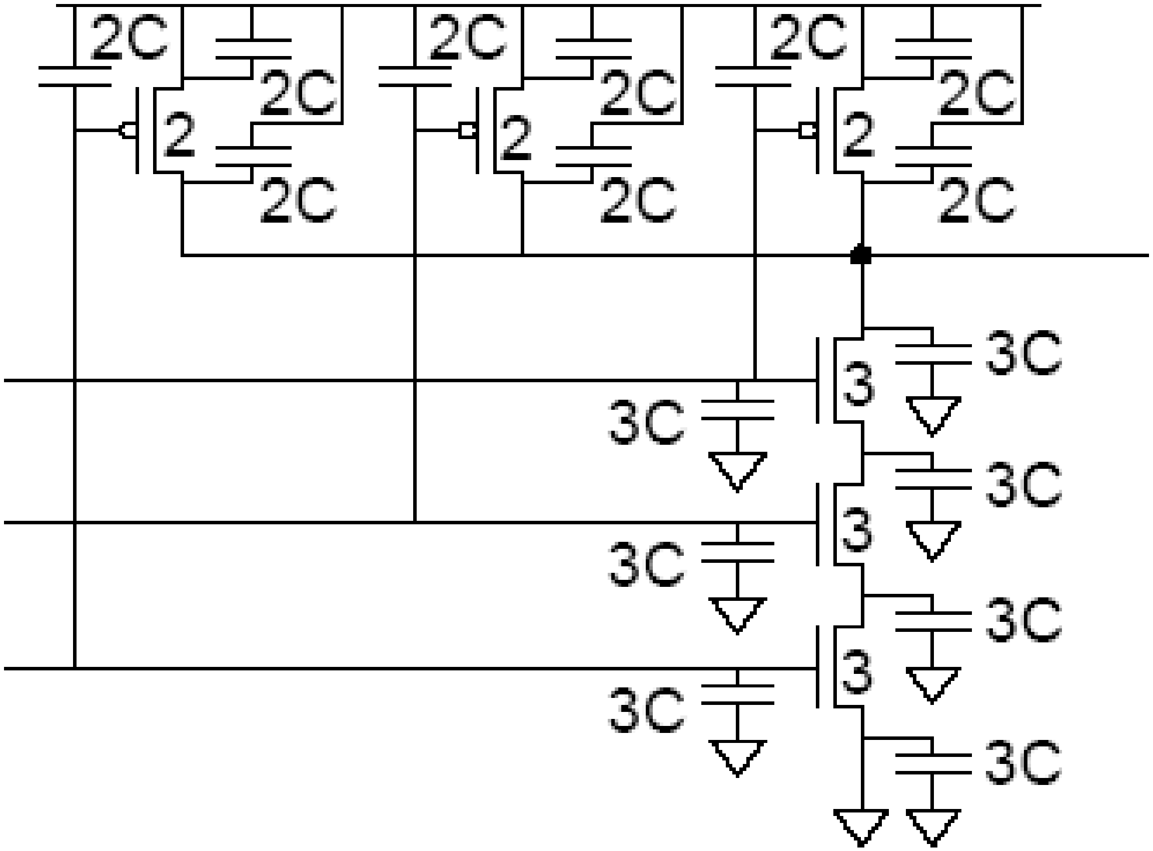
Example: 3-input NAND

- Sketch a 3-input NAND with transistor widths chosen to achieve effective rise and fall resistances equal to a unit inverter (R).



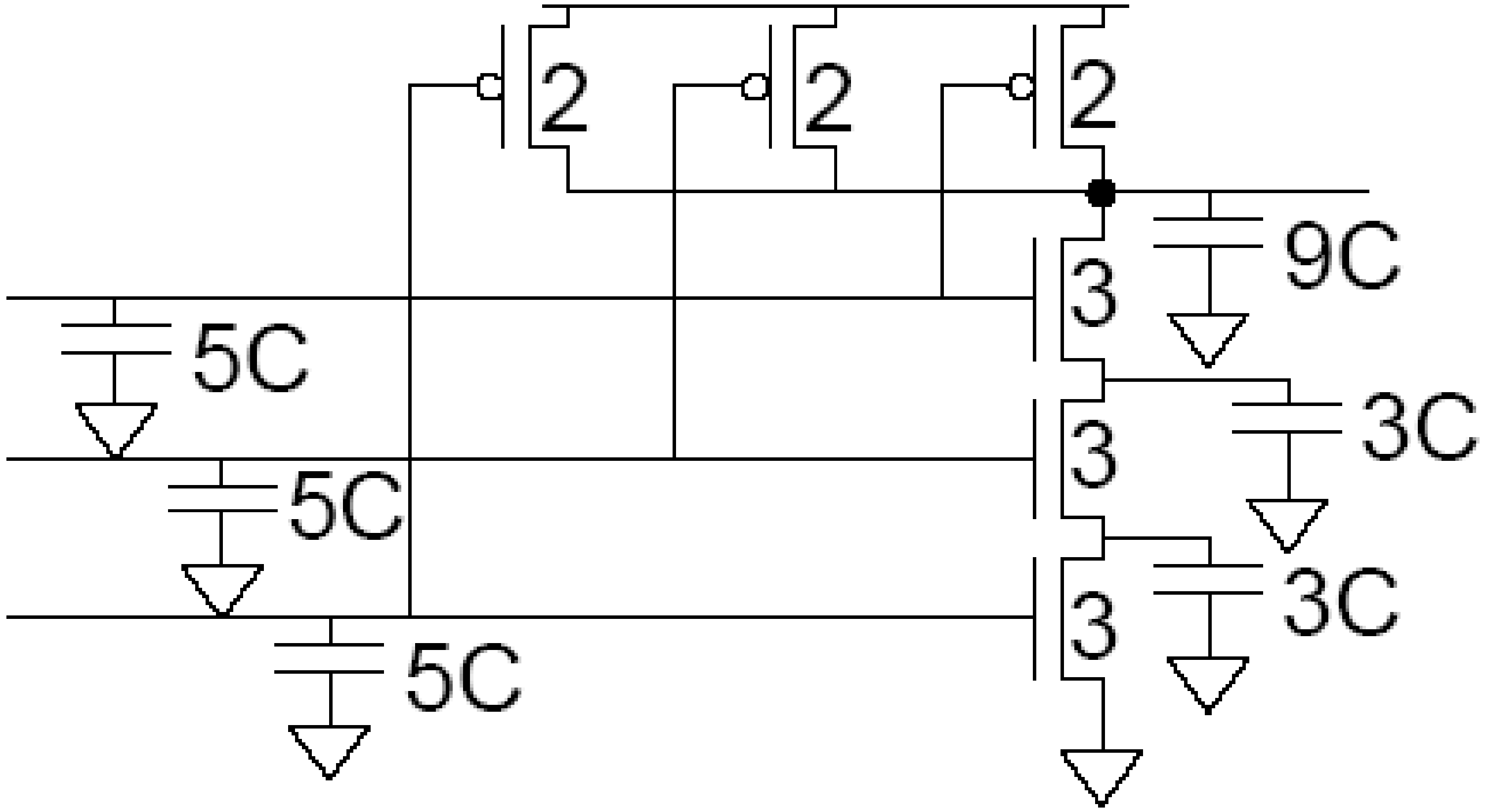
3-input NAND Caps

- Annotate the 3-input NAND gate with gate and diffusion capacitance.



3-input NAND Caps

- Annotate the 3-input NAND gate with gate and diffusion capacitance.

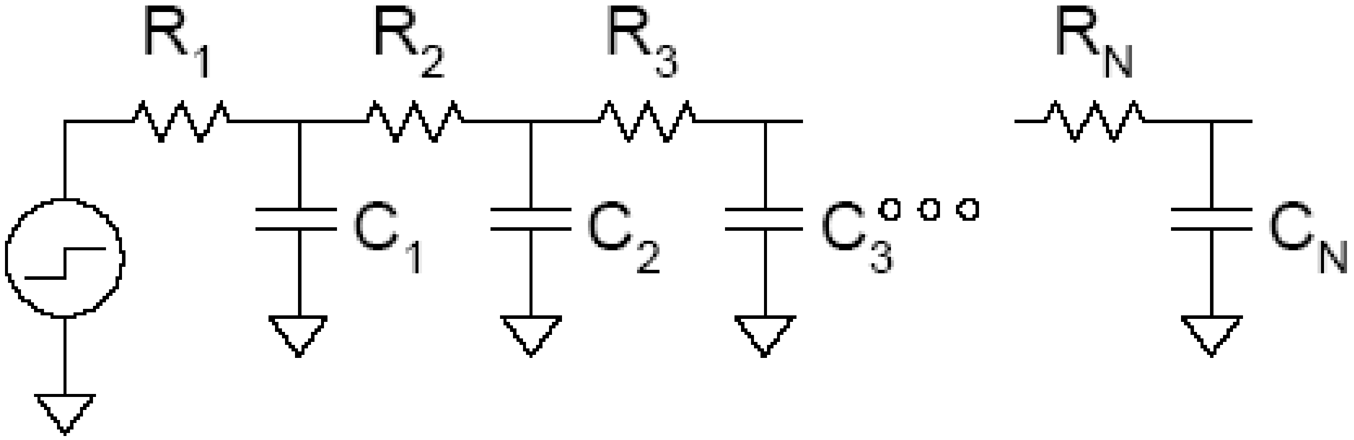


Elmore Delay

- ON transistors look like resistors
- Pullup or pulldown network modeled as *RC ladder*
- Elmore delay of RC ladder

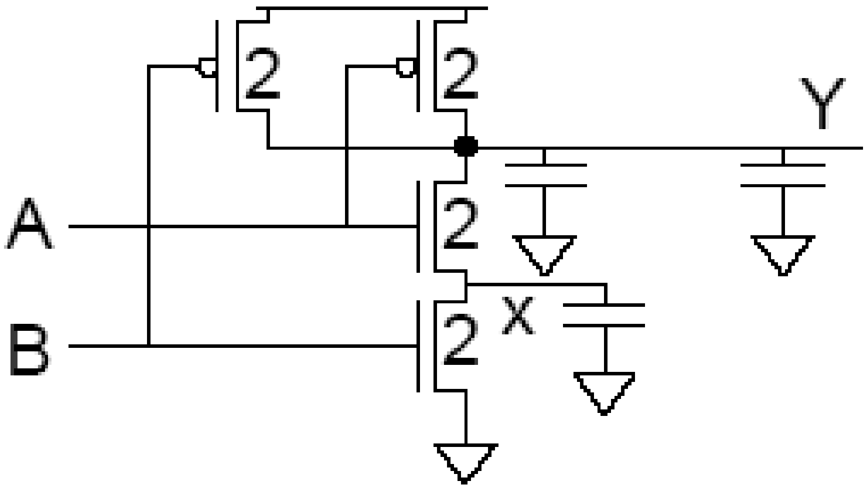
$$t_{pd} \approx \sum_{\text{nodes } i} R_{i-to-source} C_i$$

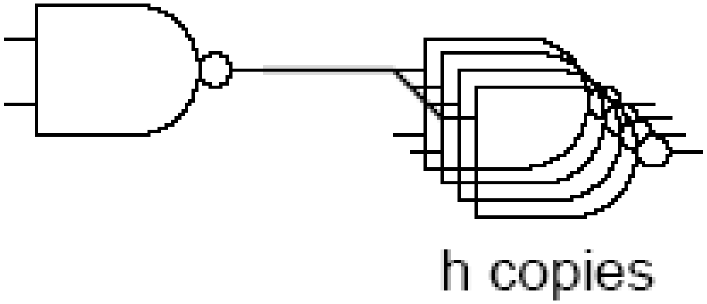
$$= R_1 C_1 + \left(R_1 + R_2\right) C_2 + \ldots + \left(R_1 + R_2 + \ldots + R_N\right) C_N$$



Example: 2-input NAND

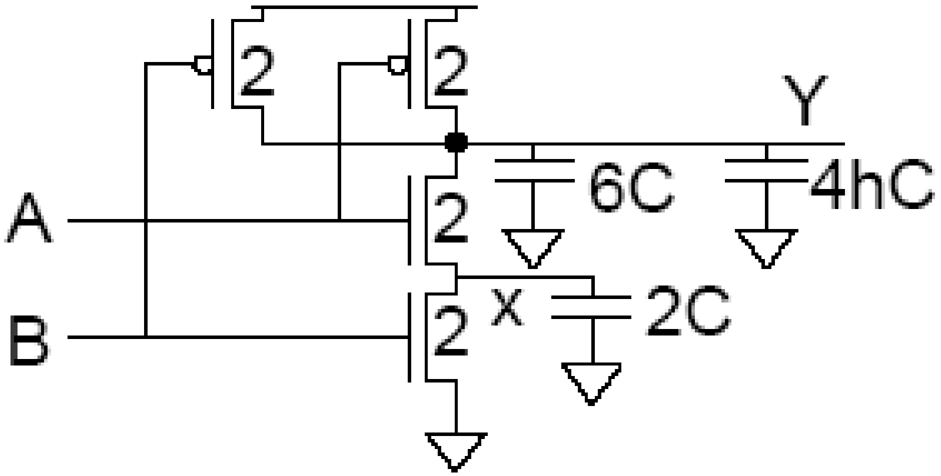
- Estimate worst-case rising and falling delay of 2-input NAND driving *h* identical gates.

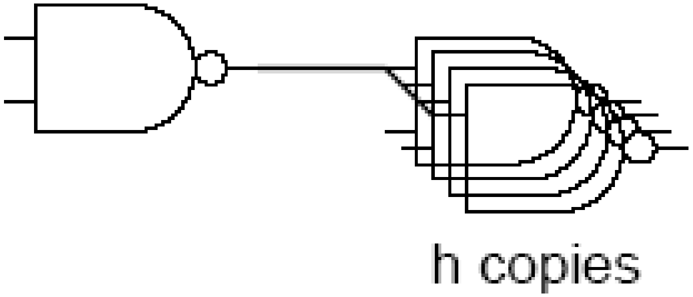




Example: 2-input NAND

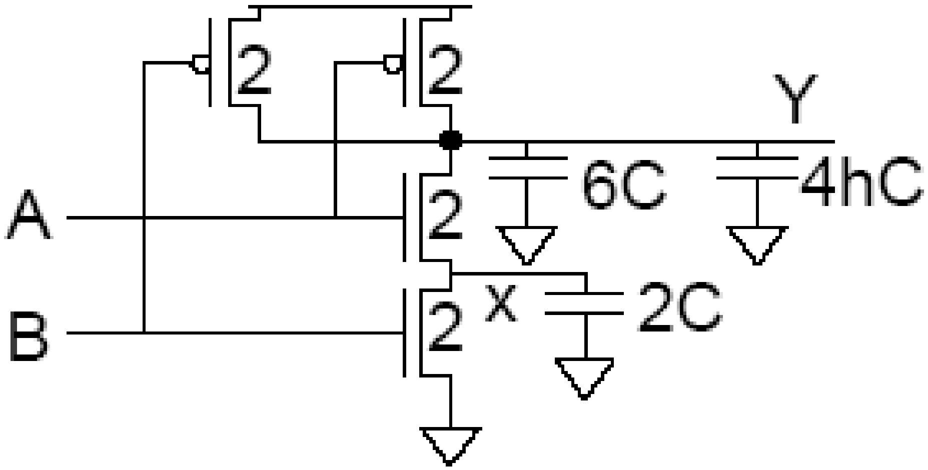
- Estimate rising and falling propagation delays of a 2-input NAND driving *h* identical gates.

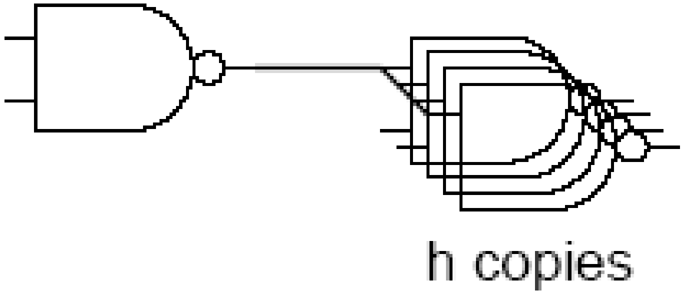


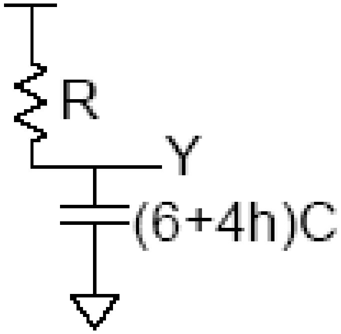


Example: 2-input NAND

- Estimate rising and falling propagation delays of a 2-input NAND driving *h* identical gates.



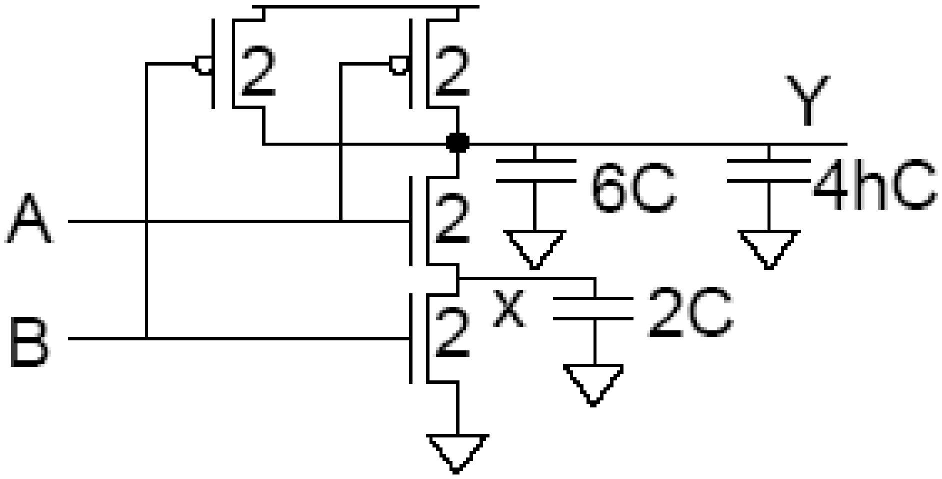


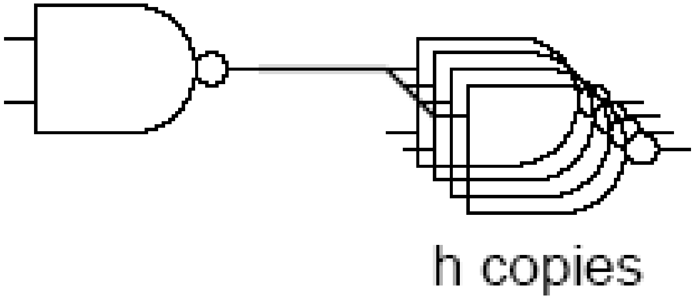


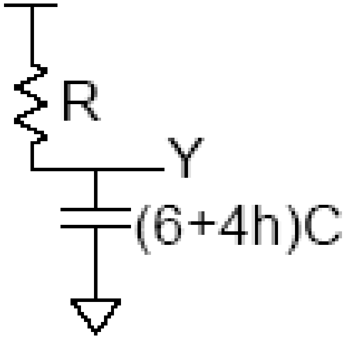
$t_{pdr} =$

Example: 2-input NAND

- Estimate rising and falling propagation delays of a 2-input NAND driving *h* identical gates.



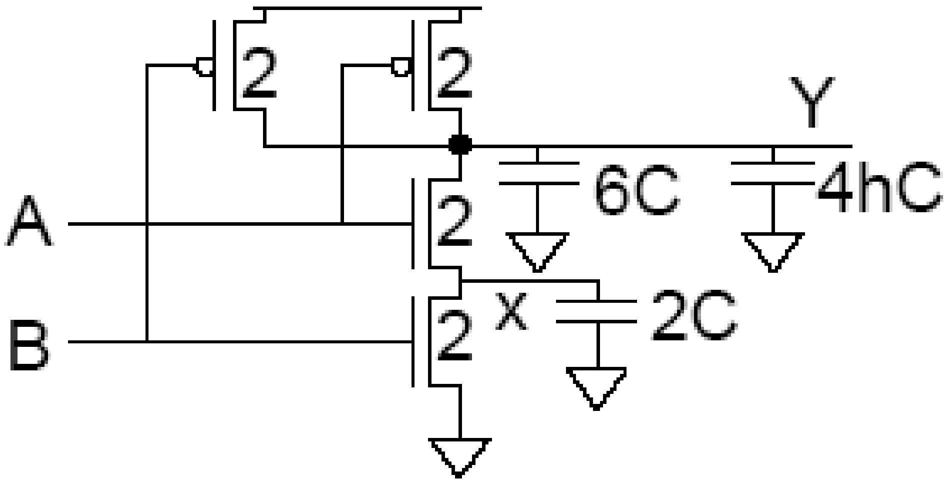


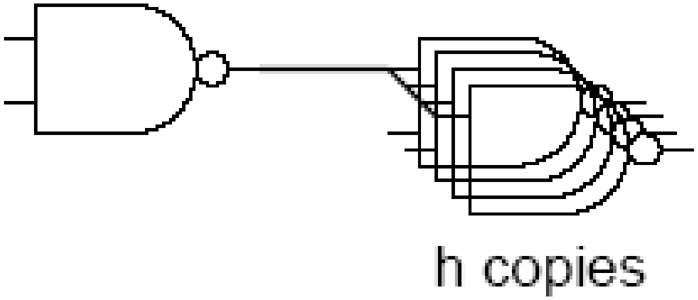


$$t_{pdr} = (6 + 4h)RC$$

Example: 2-input NAND

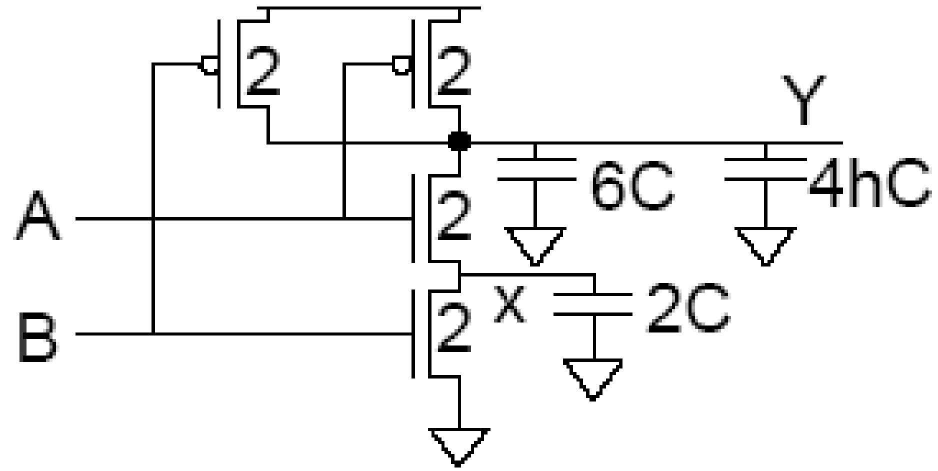
- Estimate rising and falling propagation delays of a 2-input NAND driving *h* identical gates.

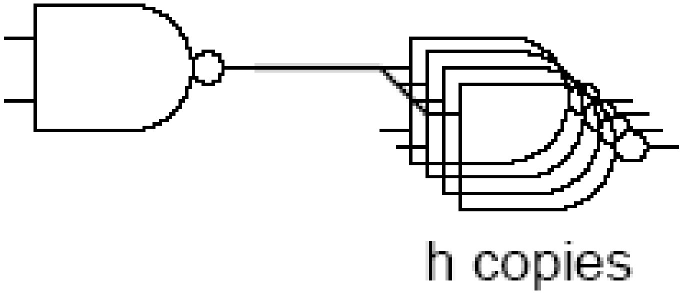


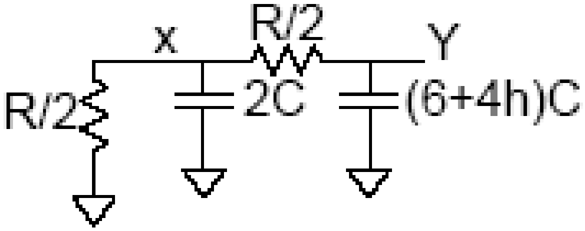


Example: 2-input NAND

- Estimate rising and falling propagation delays of a 2-input NAND driving *h* identical gates.



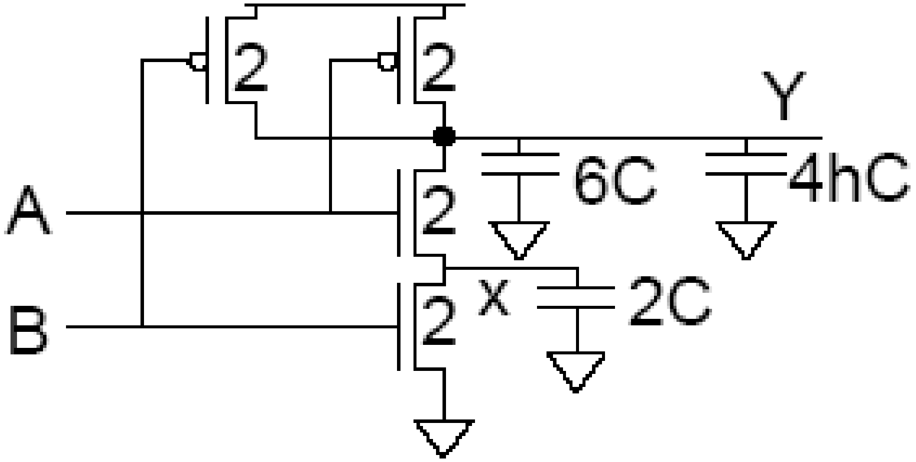


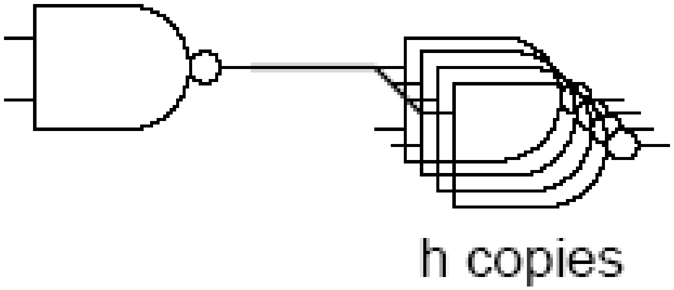


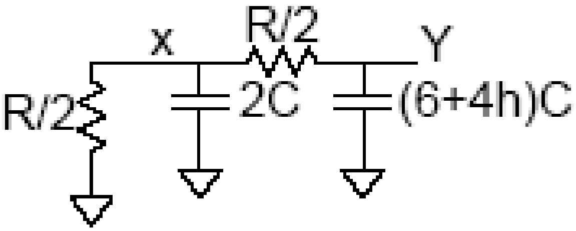
$$t_{pdf} =$$

Example: 2-input NAND

- Estimate rising and falling propagation delays of a 2-input NAND driving *h* identical gates.







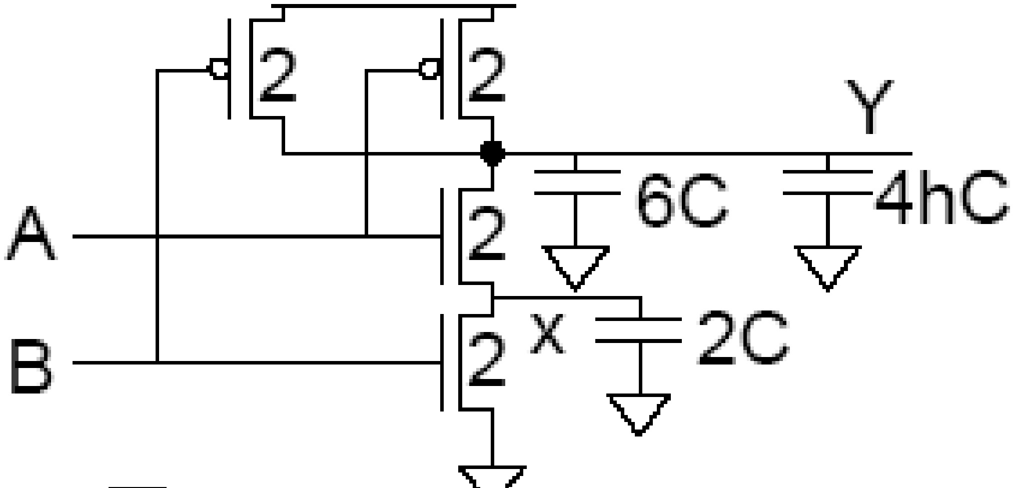
$$t_{pdf} = \left(2C\right)\left(\tfrac{R}{2}\right) + \left[\left(6+4h\right)C\right]\left(\tfrac{R}{2}+\tfrac{R}{2}\right)$$
$$= \left(7+4h\right)RC$$

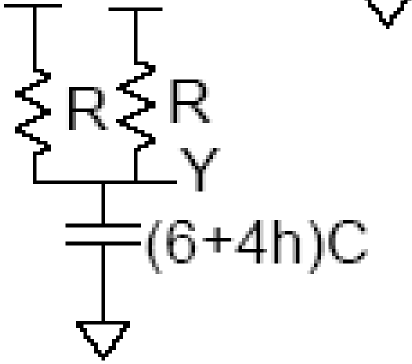
Delay Components

- Delay has two parts
  - *Parasitic delay*
    - 6 or 7 RC
    - Independent of load
  - *Effort delay*
    - 4h RC
    - Proportional to load capacitance

Contamination Delay

- Best-case (contamination) delay can be substantially less than propagation delay.
- Ex: If both inputs fall simultaneously

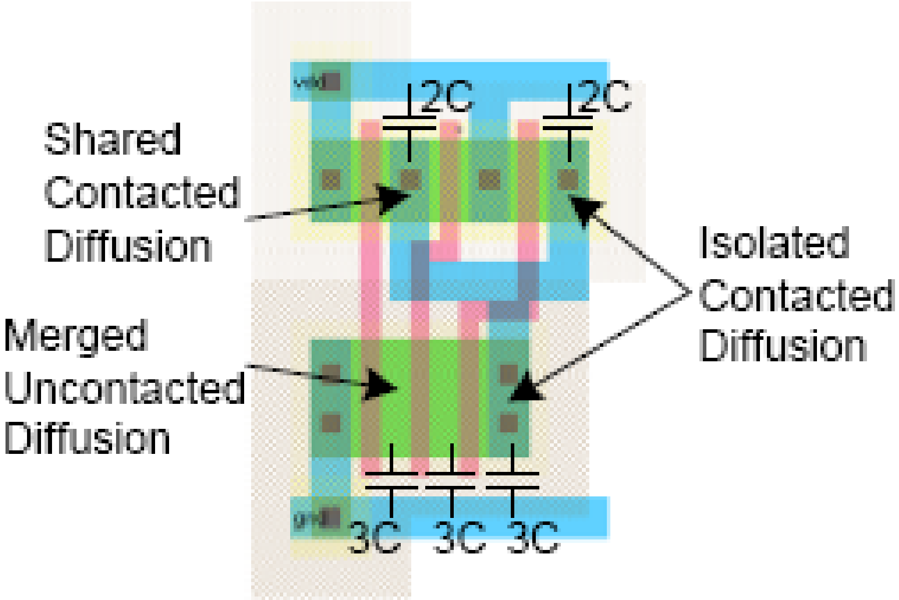


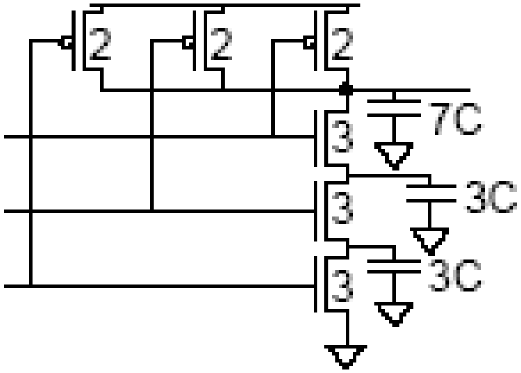


$$t_{cdr} = \left(3 + 2h\right) RC$$

Diffusion Capacitance

- we assumed contacted diffusion on every s / d.
- Good layout minimizes diffusion area
- Ex: NAND3 layout shares one diffusion contact
  - Reduces output capacitance by 2C
  - Merged uncontacted diffusion might help too


vdd
2C
2C
Shared Contacted Diffusion
Isolated Contacted Diffusion
Merged Uncontacted Diffusion
gnd
3C
3C
3C



Layout Comparison

- Which layout is better?