EE2021
Devices and Circuits

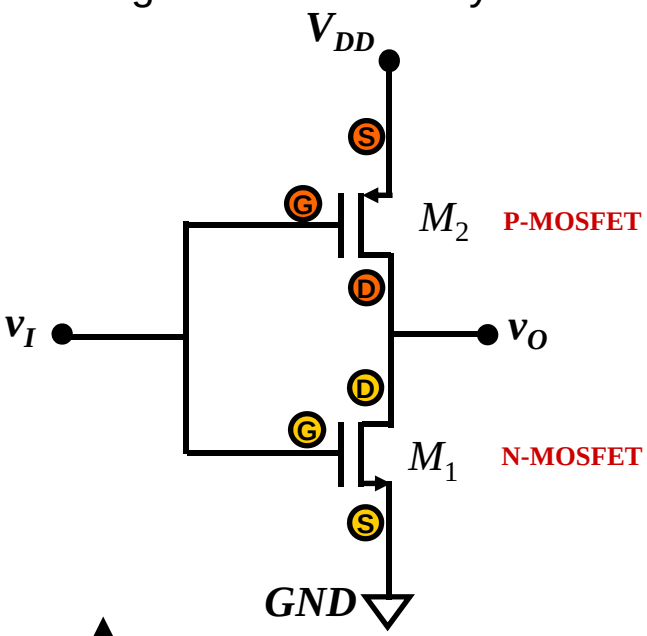
Inverter, Logic Synthesis

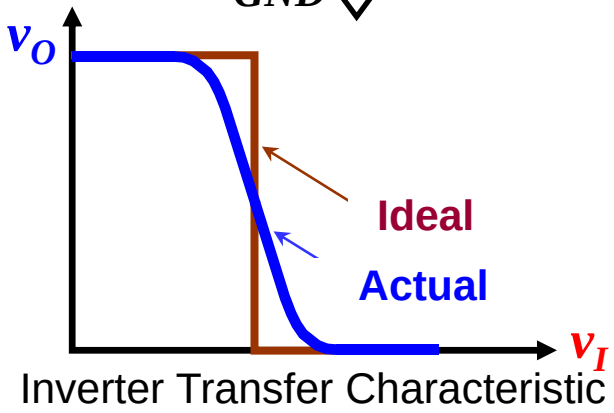
Lecture Outline

- **CMOS Inverter,**
  How to design CMOS symmetric inverter;
- **Logic Synthesis,**
  How to synthesize logic function using PUN and PDN.

CMOS Inverter Circuit

- The CMOS Inverter consists of a pair of Complementary MOSFETs (N-MOSFET & P-MOSFET).
- The source of each MOSFET is connected to its body, although not shown in the following circuit. The body effect is therefore eliminated.





Inverter Transfer Characteristic

We assume the following is true of the N-MOSFET and P-MOSFET device parameters, i.e. the devices are matched:

- The magnitudes of the threshold voltages are the same:

$$V_{THN} = |V_{THP}| = V_{TH}$$

V_{THP} (threshold voltage of the P-MOSFET) is negative

V_{THN} (threshold voltage of N-MOSFET) is positive

- We also assume that V_{TH} is less than $V_{DD}/2$.

- The values of conductance parameters are the same

$$K_n = K_p = K$$

Dimensions of N- and P- MOSFETs

- A symmetrical inverter transfer characteristic is obtained when the devices are designed to have equal K_n and K_p, where

Surface electron mobility (Field-effect electron mobility) in N-MOSFET

Ratio of transistor Width to Gate Length (Also known as sizing) for N-MOSFET

$$K_n = \frac{1}{2}\mu_n C_{ox}\left(\frac{W}{L}\right)_n$$

$$K_p = \frac{1}{2}\mu_p C_{ox}\left(\frac{W}{L}\right)_p$$

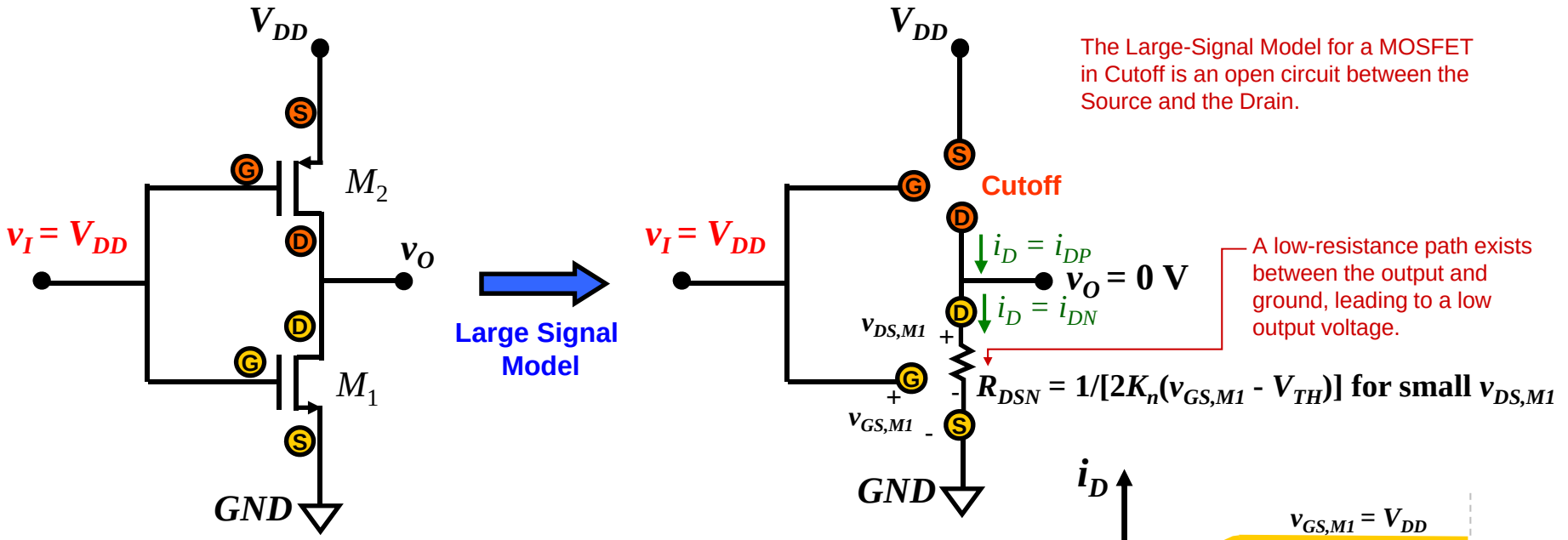
- The surface mobility μ_n of electrons in the N-MOSFET is larger than the surface mobility μ_p of holes in the P-MOSFET. $(\mu_n/\mu_p) \approx 2.5$.
- Therefore, to have $K_n = K_p$, the sizings of the transistors are such that:

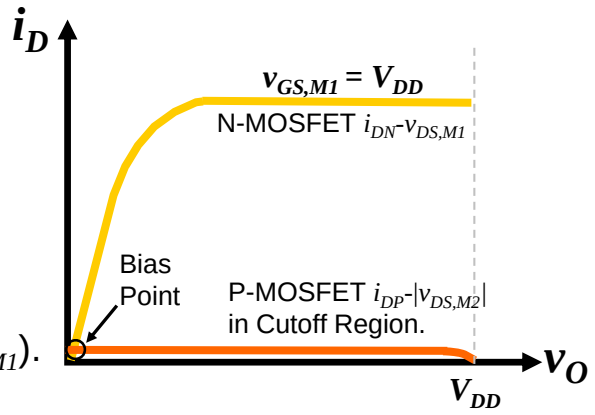
$$\left(\frac{W}{L}\right)_p = (\mu_n/\mu_p)\left(\frac{W}{L}\right)_n$$

- Usually, the N- and P-MOSFETs have the same gate length L, and the P-MOSFET is designed to have a 2.5 times larger width than the N-MOSFET.

Why do we prefer to have a symmetrical inverter?

CMOS Inverter: Input High



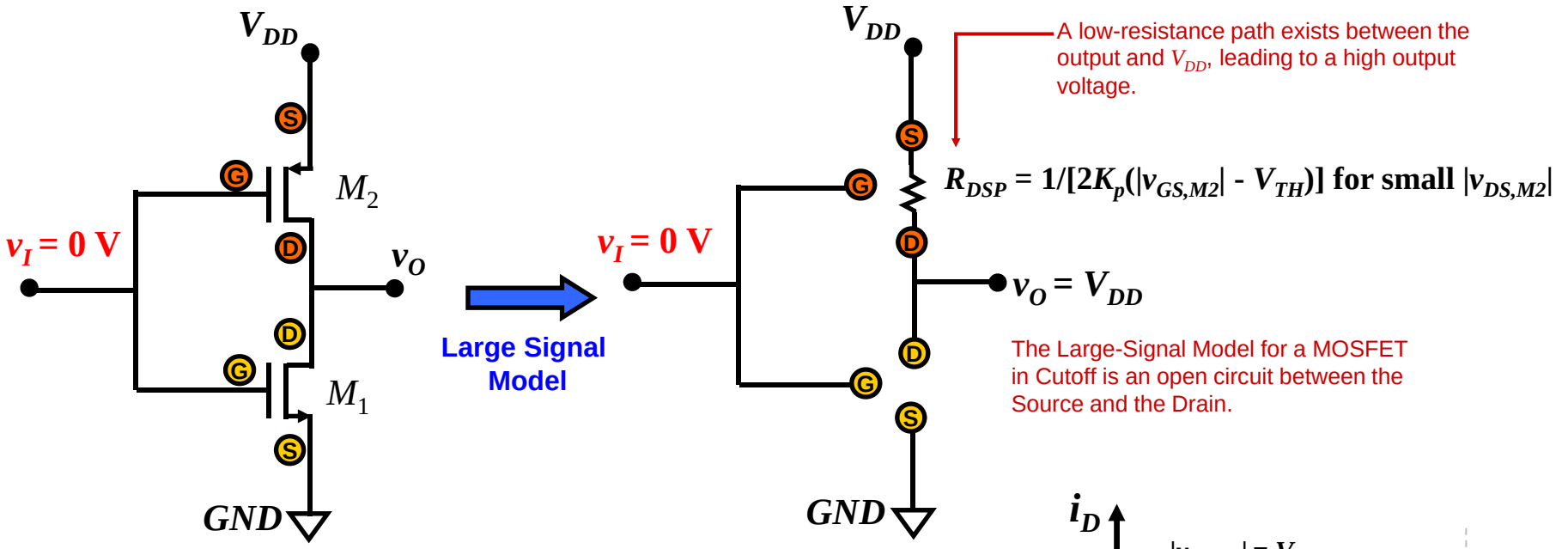


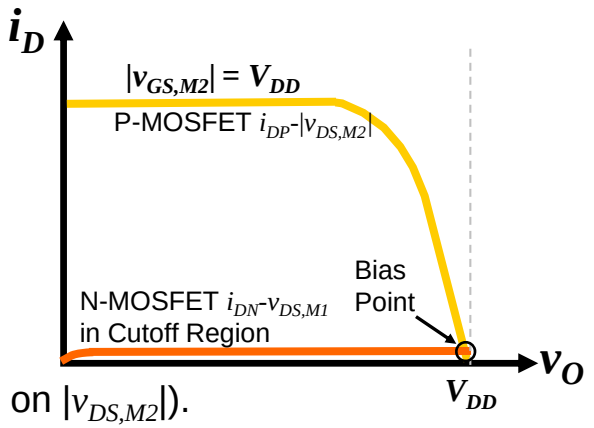
When $v_I = V_{DD}$,

- For P-MOSFET: $|v_{GS,M2}| = 0$, and $|v_{GS,M2}| < |V_{THP}| = V_{TH}$.
  P-MOSFET is in the **Cutoff** region.
- For N-MOSFET: $v_{GS,M1} = V_{DD} > V_{TH}$.
  N-MOSFET is either in **Linear** or **Saturation** region (depending on $v_{DS,M1}$).
  If there is no current i_D in the steady-state, the N-MOSFET will pull the output voltage all the way down to 0 V, and it will be in the Linear region.
- N-MOSFET pulls the output down to 0V (Pull-down Network, PDN)

Note:
- In logic, only large signal model is applied for transistor analysis.
- In amplifier, large signal model is applied for DC analysis whereas small signal model is applied for AC analysis.

CMOS Inverter: Input Low





When $v_I = 0$ V,

- For N-MOSFET: $v_{GS,M1} = 0$, and $v_{GS,M1} < V_{THN} = V_{TH}$.
  N-MOSFET is in the **Cutoff** region.
- For P-MOSFET: $|v_{GS,M2}| = V_{DD} > V_{TH}$.
  P-MOSFET is either in **Linear** or **Saturation** region (depending on $|v_{DS,M2}|$).
  If there is no current I_D in the steady-state, the P-MOSFET will pull the output voltage all the way up to V_{DD}, and it will be in the Linear region.
- P-MOSFET pulls the output up to V_{DD} (Pull-up Network, PUN)

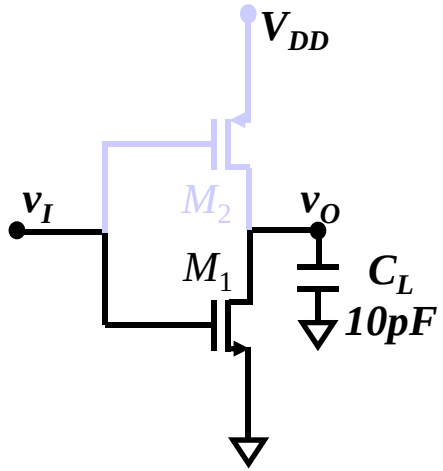
Propagation Delay

- Derived based on capacitor current. Assuming NMOS discharging current for t_{pHL}:

$$i_{DN} = C_L \frac{dv_o}{dt} = \begin{cases} K_n\left(V_{GS,M1} - V_{THN}\right)^2 & \textit{for saturation} \\ 2K_n\left[\left(V_{GS,M1} - V_{THN}\right)V_{DS,M1} - \frac{V_{DS,M1}^2}{2}\right] & \textit{for linear} \end{cases}$$

$$t_{pHL} = \sum dt = \sum \frac{C_L dv_o}{i_{DN}} \propto \frac{C_L}{K_n} \propto \frac{C_L}{\mu_n\left(\frac{W}{L}\right)_N}$$

Directly proportional to C_L, but inversely proportional to mobility μ_n and NMOS sizing $(W/L)_N$

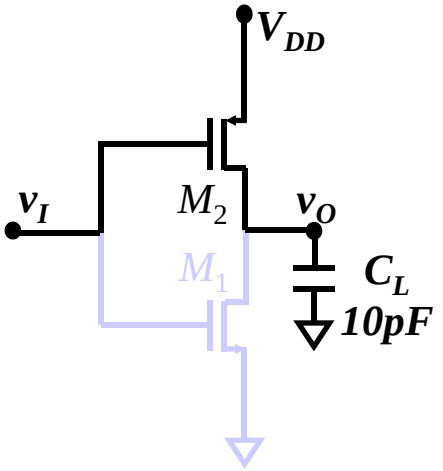


- Derived based on capacitor current. Assuming PMOS charging current for t_{pLH}:

$$i_{DP} = C_L \frac{dv_o}{dt} = \begin{cases} K_p\left(\left|V_{GS,M2}\right| - \left|V_{THP}\right|\right)^2 & \textit{for saturation} \\ 2K_p\left[\left(\left|V_{GS,M2}\right| - \left|V_{THP}\right|\right)\left|V_{DS,M2}\right| - \frac{V_{DS,M2}^2}{2}\right] & \textit{for linear} \end{cases}$$

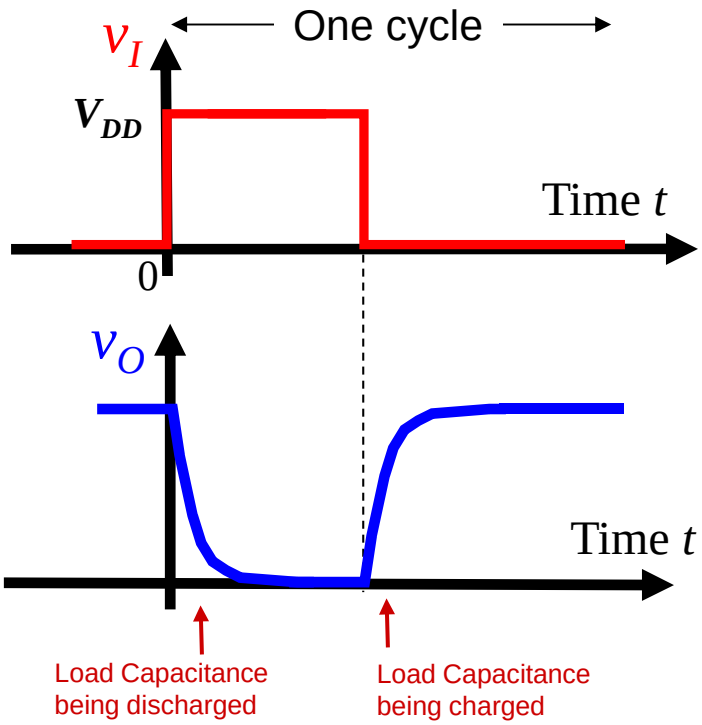
$$t_{pLH} = \sum dt = \sum \frac{C_L dv_o}{i_{DP}} \propto \frac{C_L}{K_p} \propto \frac{C_L}{\mu_p\left(\frac{W}{L}\right)_P}$$

Directly proportional to C_L, but inversely proportional to mobility μ_p and PMOS sizing $(W/L)_P$



Dynamic Power Dissipation

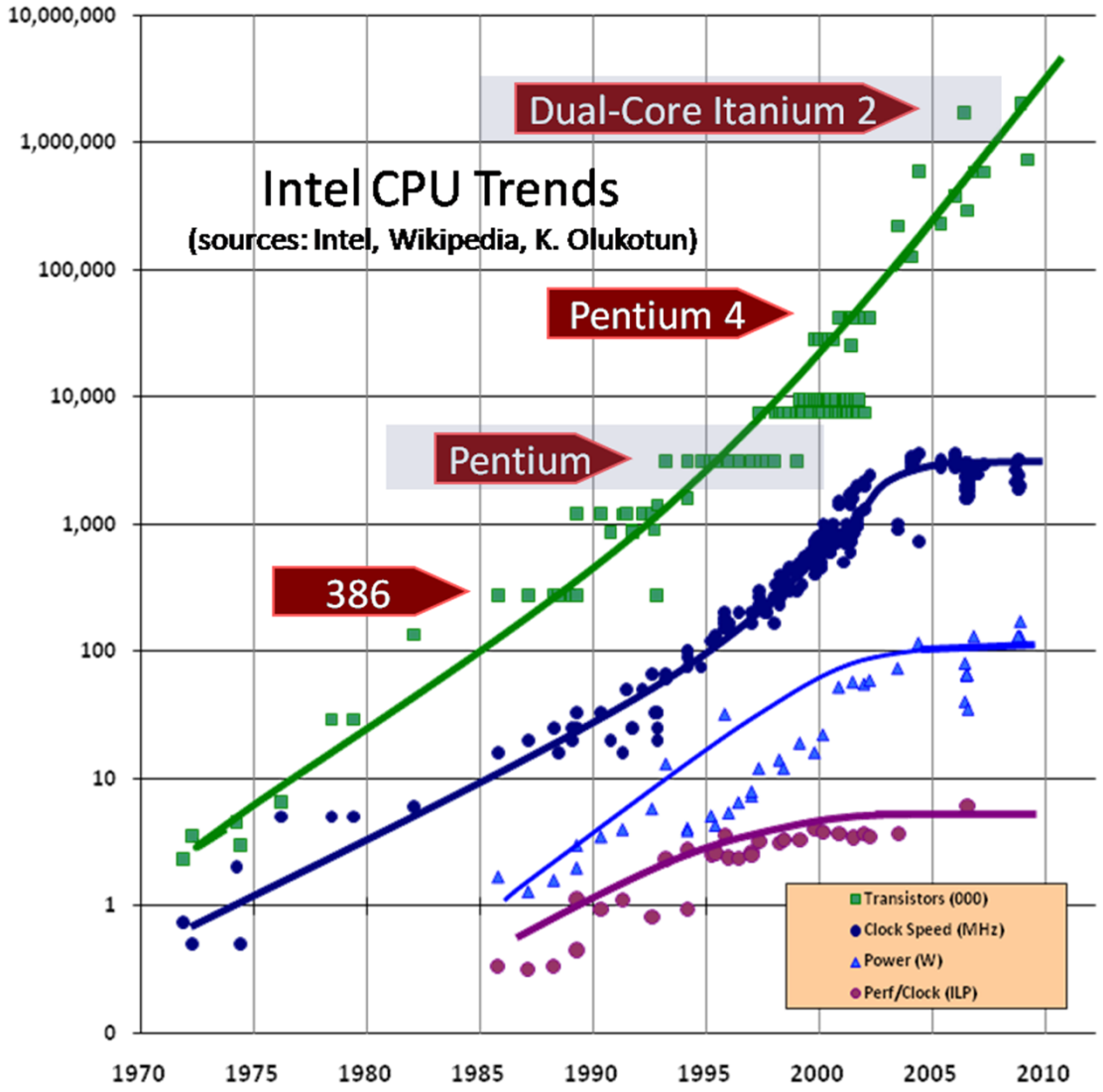
- Consider v_I as being a periodic square wave with frequency f.
- At time = 0^-, $v_O = V_{DD}$, and the energy stored on the capacitor is $(C_L V_{DD}^2)/2$.
- When v_I goes to high at $t = 0$, the N-MOSFET discharges the capacitor. The energy $(C_L V_{DD}^2)/2$ is removed from C_L and dissipated by the N-MOSFET.
- In the other half-cycle, v_I goes to zero, and the P-MOSFET charges C_L so that v_O goes from zero to V_{DD}.
- During this charging process, $(C_L V_{DD}^2)/2$ is dissipated in the P-MOSFET.
- The total energy dissipated per cycle is $C_L V_{DD}^2$. There are f cycles per second, and the energy dissipated per second is $fC_L V_{DD}^2$.
- One advantage of static CMOS logic is that power is only consumed during switching. Very little power consumed during the static mode due to leakage. (Leakage power can be significant for advanced technology with trillion transistors)



Dynamic Power Dissipation = $fC_L V_{DD}^2$

A larger V_{DD} would lead to a larger Dynamic Power Dissipation

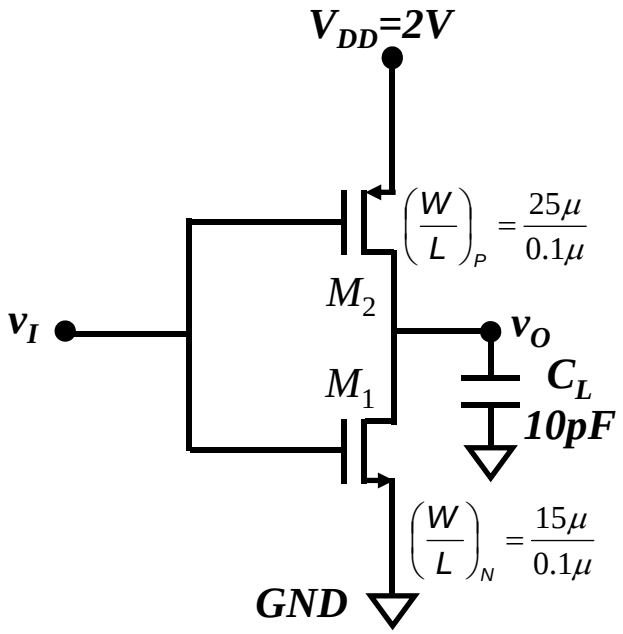
CPU Trend



A. P. Chandrakasan, A. Burstein, and R. W. Brodersen, “A low-power chipset for a portable multimedia applications,” ISSCC Dig. Tech. Papers, pp. 82-83, Feb. 1994.

Example

- Estimate the ratio of t_{pHL} and t_{pLH}, and dynamic power dissipation with input frequency of 100MHz.



$\mu_n = 2.5\mu_p$

- Applying slide 5-7 for propagation delay,

$$\frac{t_{pHL}}{t_{pLH}} = \frac{\frac{1}{\mu_n\left(\frac{W}{L}\right)_N}}{\frac{1}{\mu_p\left(\frac{W}{L}\right)_P}} = \frac{\mu_p\left(\frac{25}{0.1}\right)}{\mu_n\left(\frac{15}{0.1}\right)} = 0.67$$

- Applying slide 5-8 for dynamic power,

$$\begin{aligned} Power &= f \times C_L \times V_{DD}^2 \\ &= 100MHz \times 10pF \times 2^2 \\ &= 4mW \end{aligned}$$

CMOS Logic Gate: Basic Features

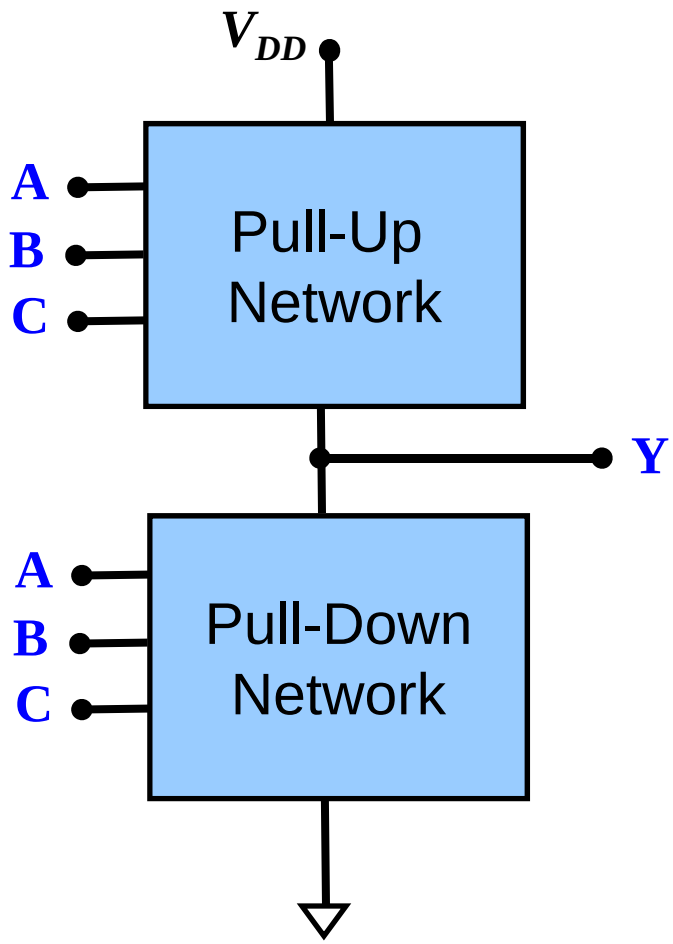
A CMOS Logic Gate Circuit is an extension or generalization of the CMOS inverter. It consists of 2 networks:

- **Pull-Up Network (PUN)** comprising PMOS transistors.
- **Pull-Down Network (PDN)** comprising NMOS transistors.

The Pull-Up Network (PUN) and the Pull-Down Network (PDN) are controlled by input variables (e.g. A, B, C) in a complementary fashion.

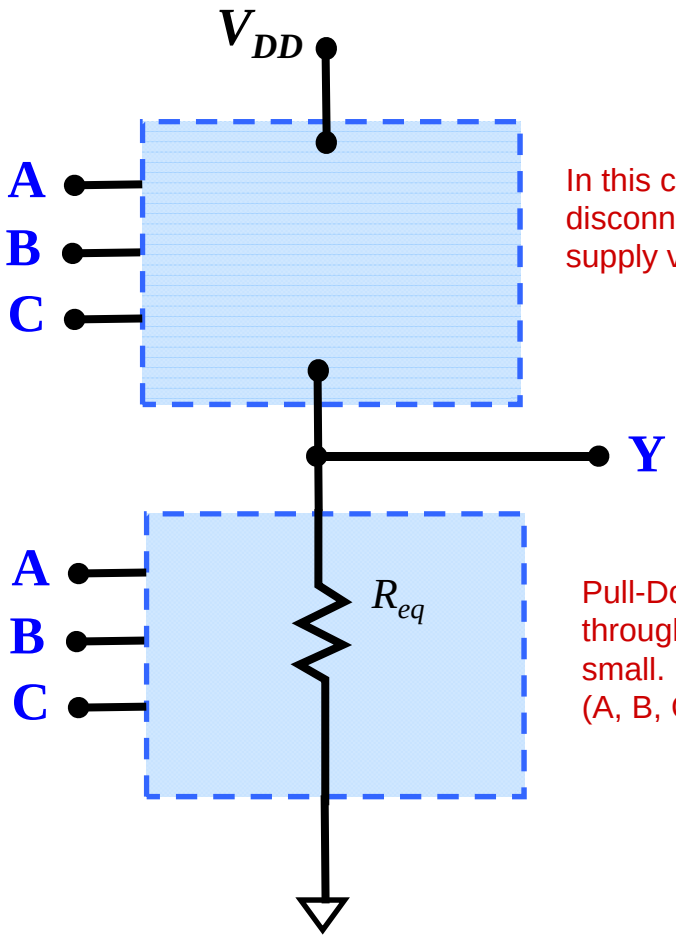
PUN and PDN are Complementary, i.e. for any given combination of input variables, either PUN pulls the output voltage to V_{DD} or PDN pulls the output voltage to ground.

An Example of a 3-input Logic Gate.



Activation of Pull-Down Network (PDN)

- In this example of a 3-input Logic Gate, the PDN will conduct for all input combinations that require a low output (Y = 0), and will then pull the output voltage down to ground, causing a zero voltage to appear at the output.
- At the same time, the PUN will be turned off.
- No direct DC path will exist between V_{DD} and ground in the entire circuit.

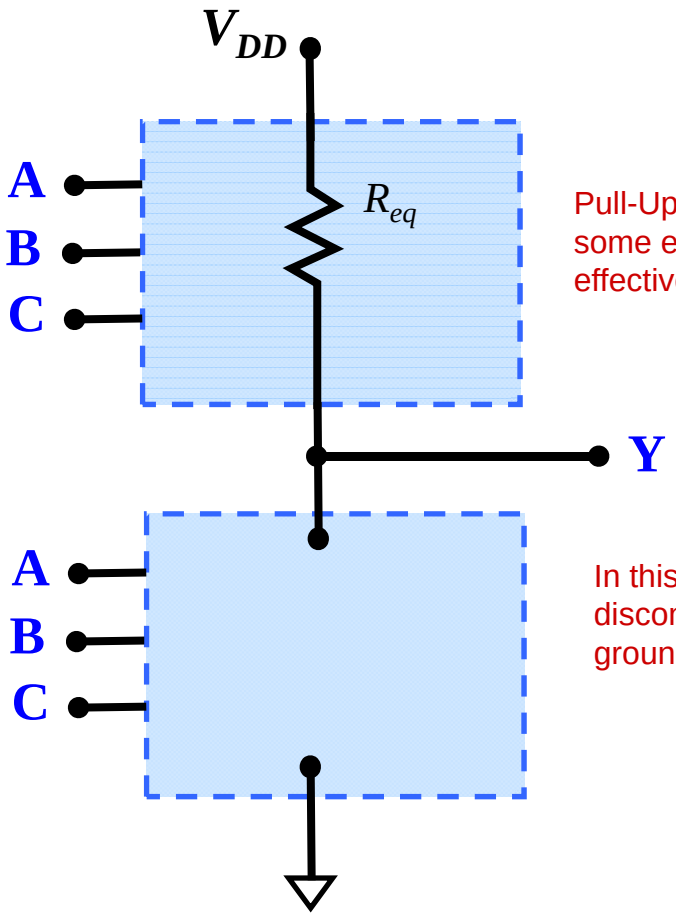


In this case, the Pull-Up Network disconnects the Output Node from the supply voltage V_{DD}.

Pull-Down Network connects the Output Node to ground through some effective or equivalent resistance R_{eq} which is small. The equivalent resistance is a function of the logic states (A, B, C).

Activation of Pull-Up Network (PUN)

- All input combinations that call for a high output (Y = 1) will cause PUN to conduct, and the PUN will then pull the output voltage up to supply voltage V_{DD}, establishing $v_Y = V_{DD}$.
- Simultaneously, the PDN will be cut off.
- No direct DC path will exist between V_{DD} and ground in the circuit.

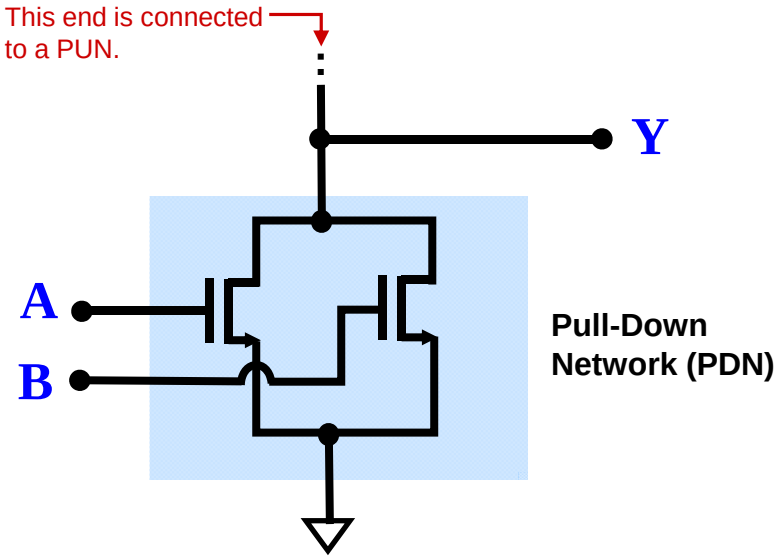


Pull-Up Network connects the Output Node to V_{DD} through some effective or equivalent resistance R_{eq} which is small. The effective resistance is a function of the logic states (A, B, C).

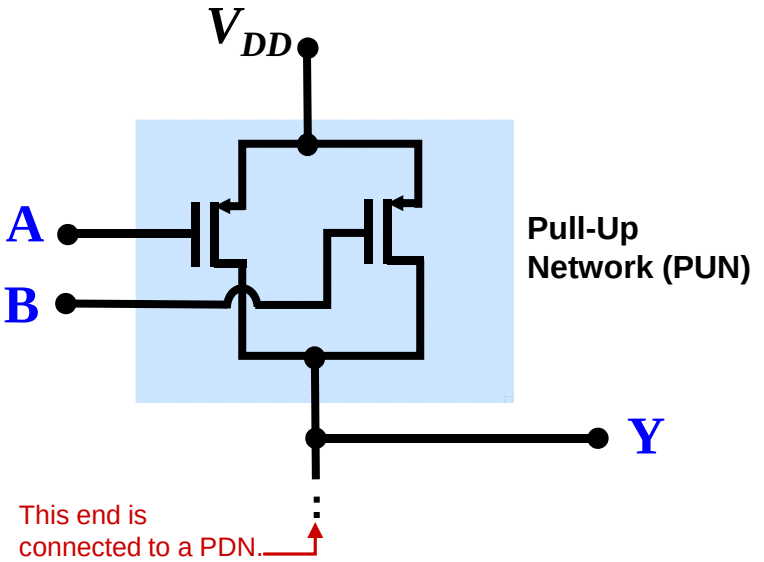
In this case, the Pull-Down Network disconnects the Output Node from the ground.

How PUN and PDN are activated

- **PDN** comprises NMOS transistors, and an NMOS transistor conducts when the signal at its gate is a high voltage, i.e. $V_{GS} > V_{THN}$. The PDN is activated when the inputs are high.



- **PUN** comprises PMOS transistors, and a PMOS transistor conducts when the signal at its gate is a low voltage, i.e. $|V_{GS}| > |V_{THP}|$. The PUN is activated when the inputs are low.

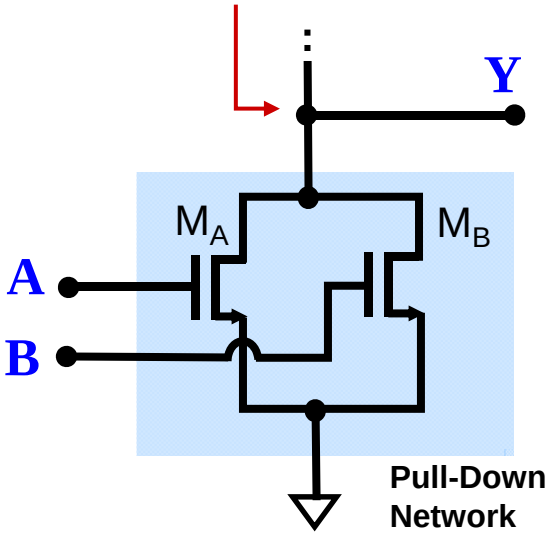


Logic Implemented by PDN

The PDN ultilizes

- NMOS Transistors in parallel to form an NOR function.
- NMOS Transistors in Series to form an NAND function.

M_A will conduct when A is high ($v_A = V_{DD}$) and will then pull the output voltage down to ground ($v_Y = 0$ V, Y = 0). Similarly, when M_B conducts, $v_Y = 0$ V, Y = 0.

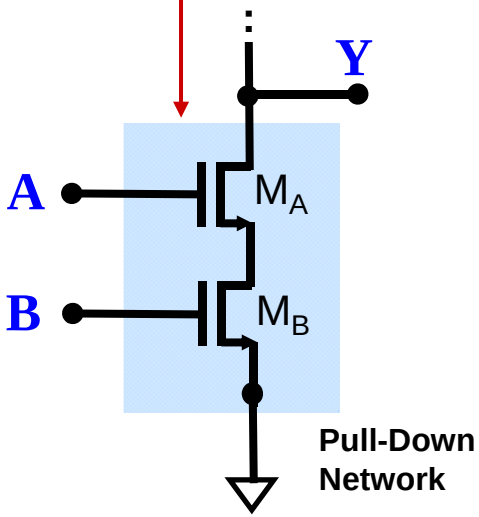


$\overline{Y} = A + B$

$Y = \overline{A + B}$

See that when A or B is high, Y will be low.

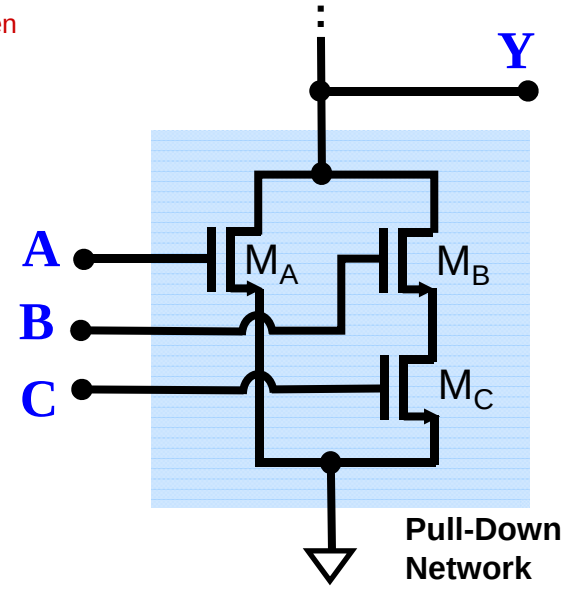
See the PDN will conduct only when both M_A and M_B conduct, i.e. only when A and B both high simultaneously.



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

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

See that when A and B are both high, Y will be low.



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

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

See that when A is high or when B and C are both high, Y will be low.

For constructing the PDN, it is useful to obtain the Boolean expression for $\overline{Y}$ in terms of the uncomplemented logic variables.

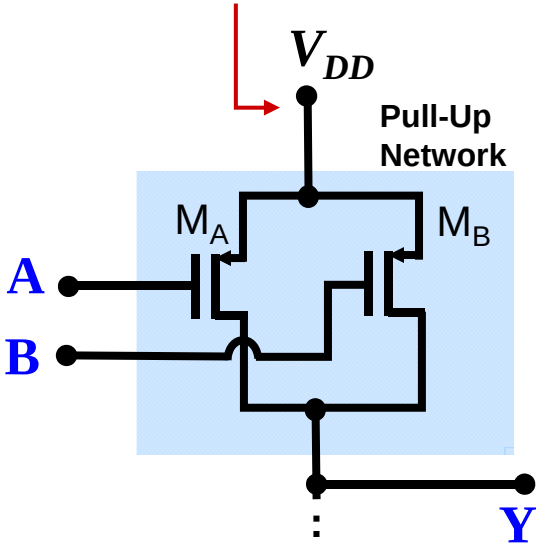
Logic Implemented by PUN

The PUN ultilizes

PMOS only turns on when input is low

- PMOS Transistors in parallel to form an OR function with inverted input.
- PMOS Transistors in Series to form an AND function with inverted input.

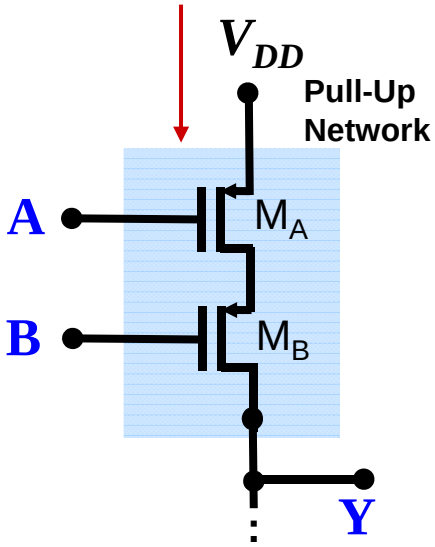
M_A will conduct when A is low ($v_A = 0$ V) and will then pull the output voltage up to V_{DD} ($v_Y = V_{DD}$, Y = 1). Similarly, when M_B conducts, $v_Y = V_{DD}$, Y = 1.



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

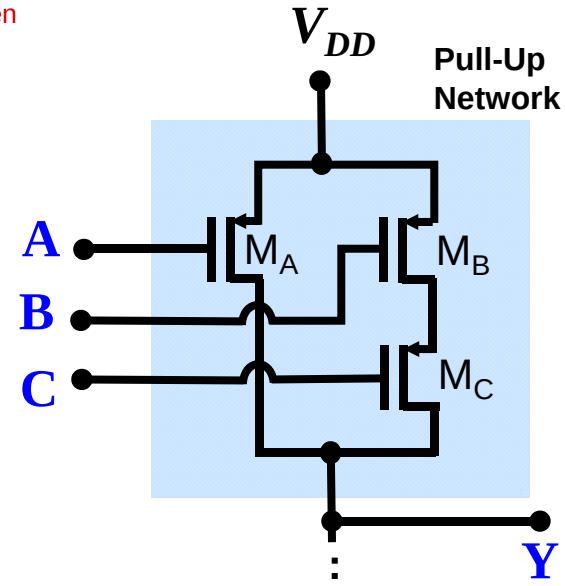
See that when A or B is low, Y will be high.

See the PUN will conduct only when both M_A and M_B conduct, i.e. only when A and B both low simultaneously.



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

See that when A and B are both low, Y will be high.



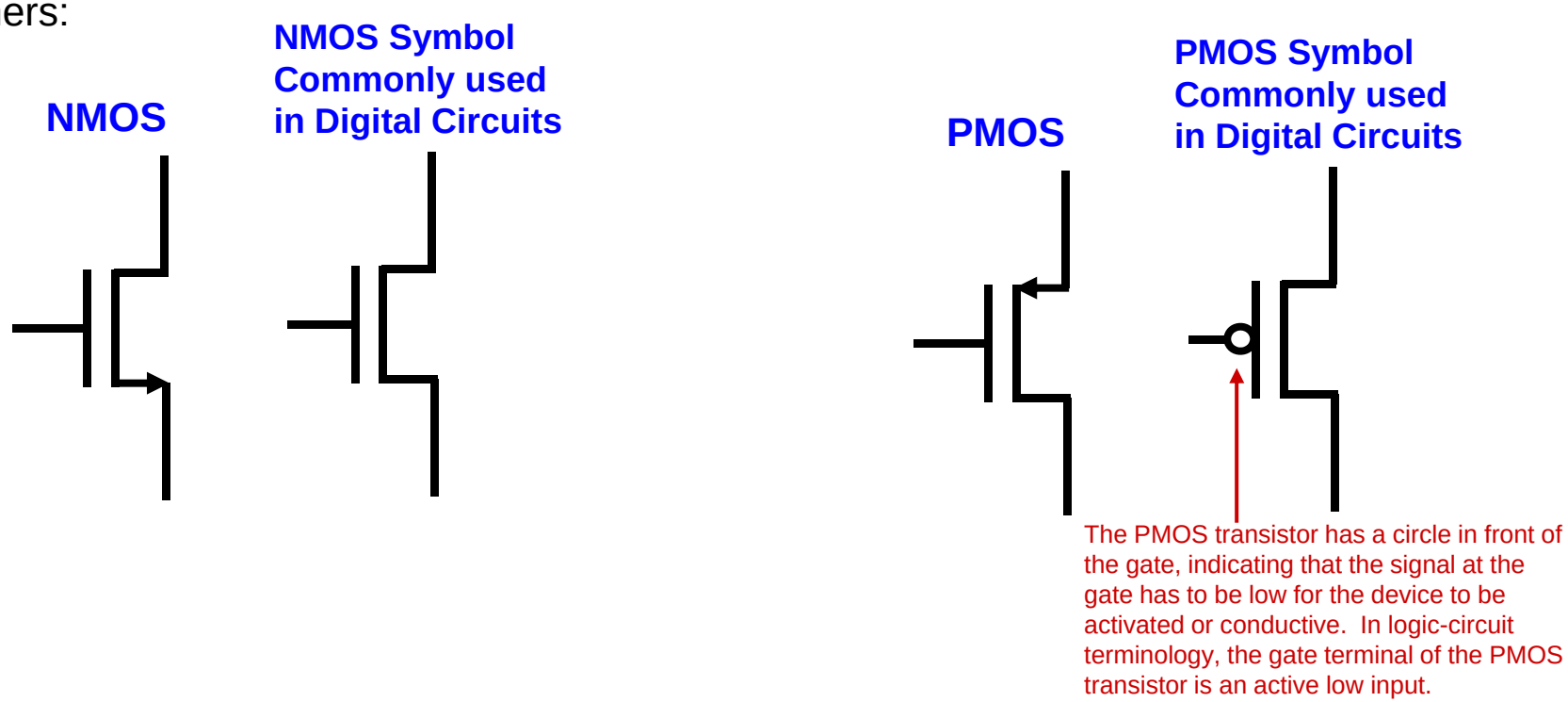
$Y = \overline{A} + \overline{B} \cdot \overline{C}$

See that when A is low or when B and C are both low, Y will be high.

For constructing the PUN, it is useful to look at the Boolean expression for Y in terms of the complemented logic variables.

Circuit Symbols for NMOS & PMOS

- Here, we introduce alternative circuit symbols for NMOS and PMOS transistors.
- The following symbols are almost universally used for MOS transistors by Digital-Circuit designers:



- In transistor symbols used in Digital Circuits, there is no indication of which terminal is the source and which terminal is the drain.
- This is not a problem, as for an NMOS transistor, the drain is the terminal with a higher voltage (I_D flows from drain to source), and for a PMOS transistor, the source is the terminal with a higher voltage (I_D flows from source to drain).

Two-Input NOR Gate

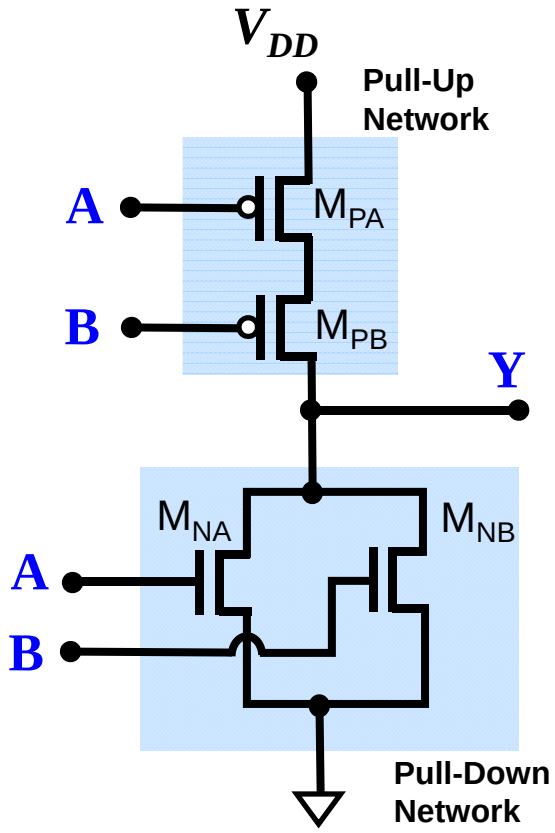
Consider a CMOS logic gate that realizes the two-input NOR function:

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

The implementation is as follows:

Note: DeMorgan's Law for Boolean algebra:

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



Y is to be high (PUN conducting) when A is low and B is low. See that when A and B are both low, the PDN does not conduct.

Y is to be low (PDN conducting) when either A is high or B is high. See that when either A or B is high, the PUN does not conduct.

There will be no combinations of A and B where the PUN and the PDN are both conductive or both non-conductive. The PDN and PUN are complementary, one of them is conductive, and the other is non-conductive.

Two-Input NAND Gate

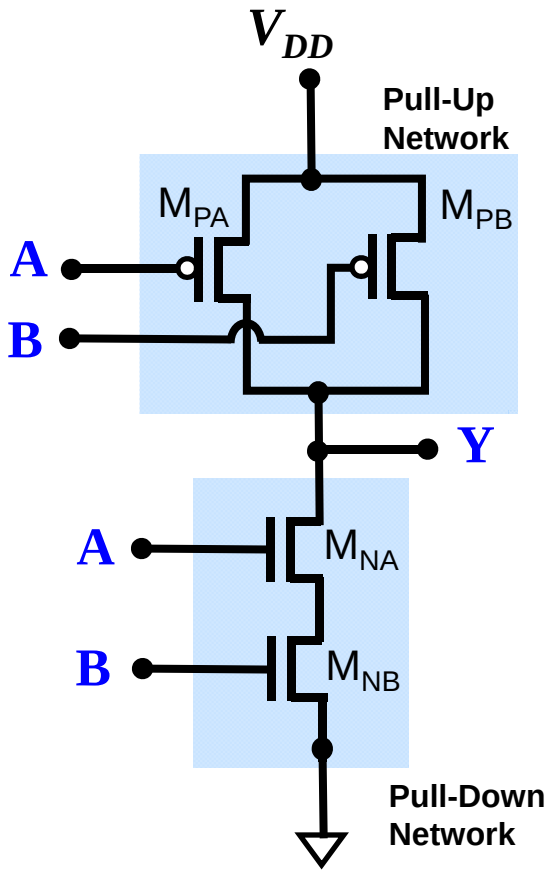
Consider a CMOS logic gate that realizes the two-input NAND function:

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

Note: DeMorgan's Law for Boolean algebra:

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

The implementation is as follows:



Y is to be high (PUN conducting) when A is low or B is low. See that when A or B are low, the PDN does not conduct.

Y is to be low (PDN conducting) when both A and B are high. See that when A and B are both high, the PUN does not conduct.

We see that there will be no combinations of A and B where both PUN and PDN are conductive or non-conductive. The PDN and PUN are complementary, one of them is conductive, and the other is non-conductive.

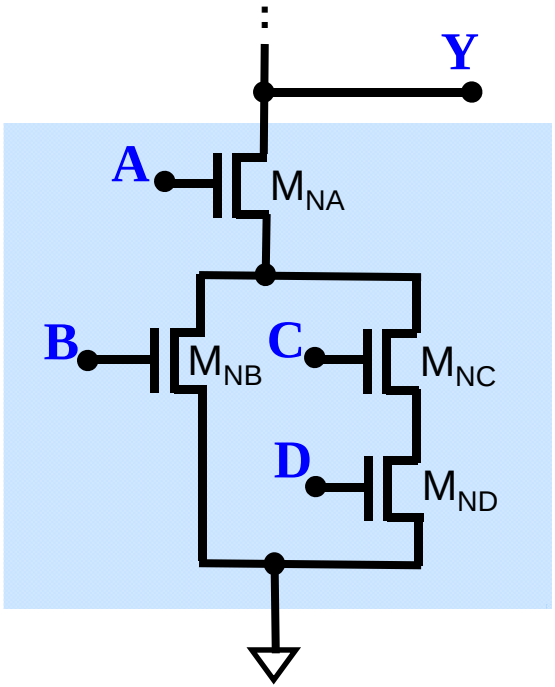
Example: A Complex Logic Gate

Consider a more complex logic function: $Y = \overline{A(B + CD)}$

which can be written as $\overline{Y} = A(B + CD)$

See that Y is to be low when A is high and simultaneously either B or C and D are both high.

By inspecting the above expression, we can draw the following Pull-Down Network (PDN):



Pull-Down Network

$\overline{Y} = A(B + CD)$ ← A and (B+CD) have same level of hierarchy

$(B + CD)$ ← B and CD have same level of hierarchy

CD ← C and D have same level of hierarchy

The PDN is most directly obtained or synthesized by expressing $\overline{Y}$ in terms of the uncomplemented logic variables. If complemented variables appear in this expression, additional inverters will be required to generate the complemented variables.

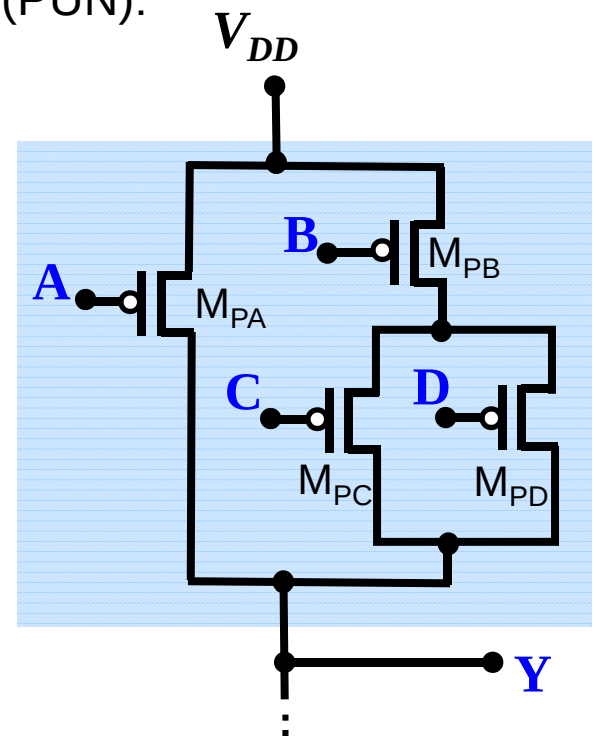
Construction of PUN

The PUN is most directly synthesized by expressing Y in terms of the complemented logic variables. If uncomplemented variables appear in this expression, additional inverters will be required to generate the uncomplemented variables.

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

See that Y is to be high
1. When A is low OR
2. When B is low AND (C is low OR D is low).

By inspecting the above expression, we can draw the following Pull-Up Network (PUN):



Pull-Up Network

$$Y = \underbrace{\overline{A}}_{} + \underbrace{\overline{B} \cdot \underbrace{(\overline{C} + \overline{D})}_{}}_{}$$

$\overline{A}$ and $\overline{B} \cdot (\overline{C} + \overline{D})$ have same level of hierarchy

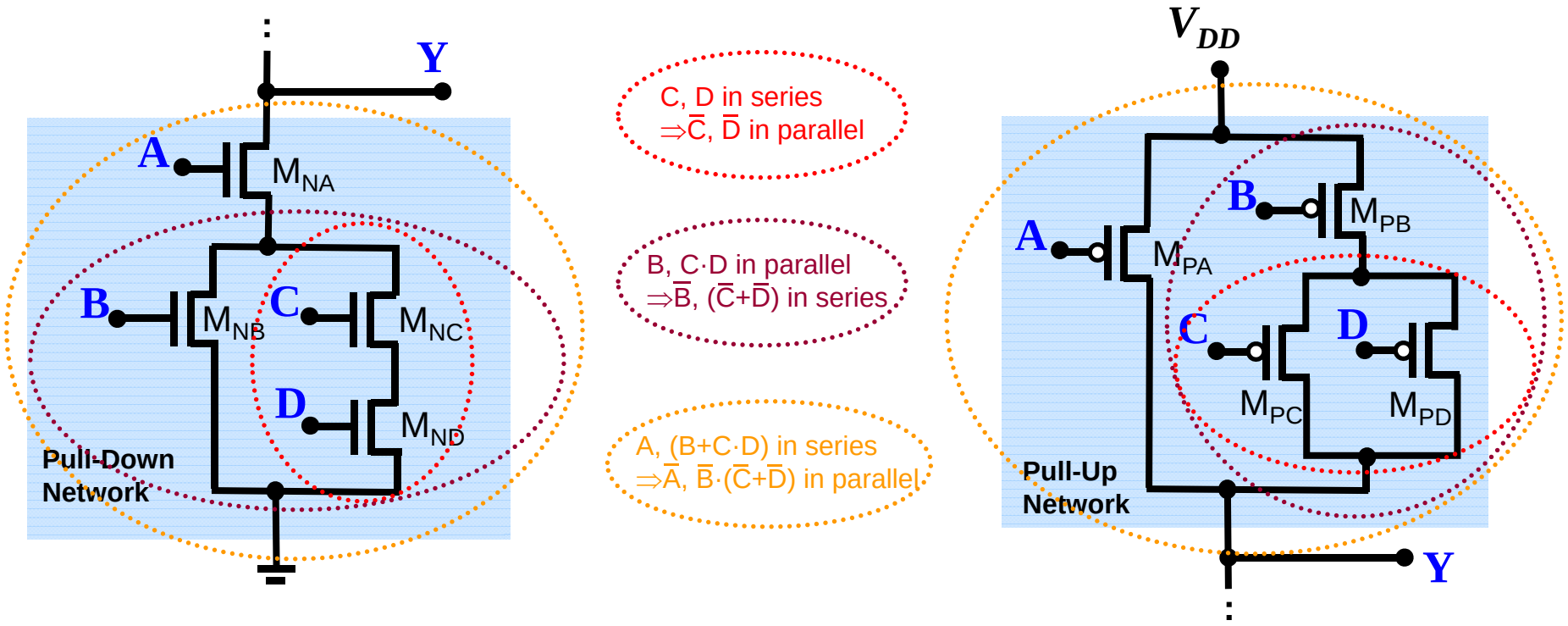
$$\underbrace{\overline{B}}_{} \cdot \underbrace{(\overline{C} + \overline{D})}_{}$$

$\overline{B}$ and $(\overline{C} + \overline{D})$ have same level of hierarchy

$$\underbrace{\overline{C}}_{} + \underbrace{\overline{D}}_{}$$

$\overline{C}$ and $\overline{D}$ have same level of hierarchy

Construction of PUN by Duality

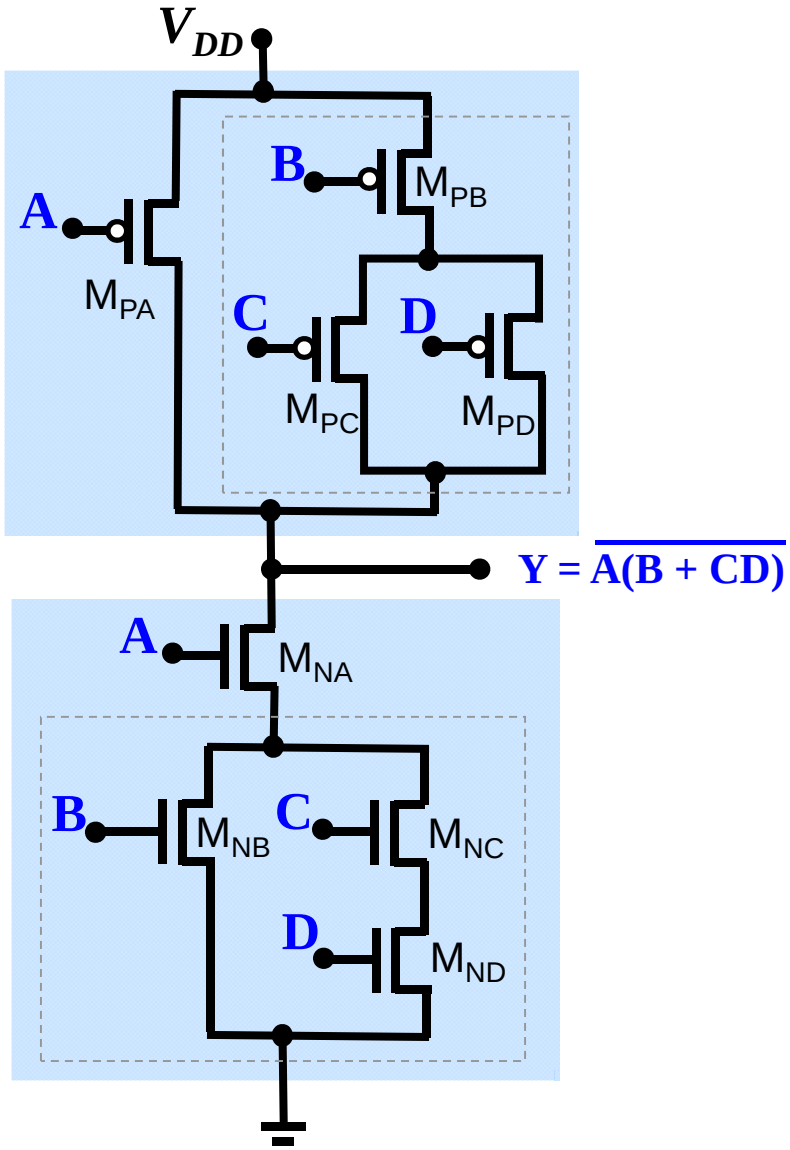


Rule of thumb

1) NMOS in series ⇒ PMOS in parallel
2) NMOS in parallel ⇒ PMOS in series
3) Vice versa for PMOS to NMOS conversion

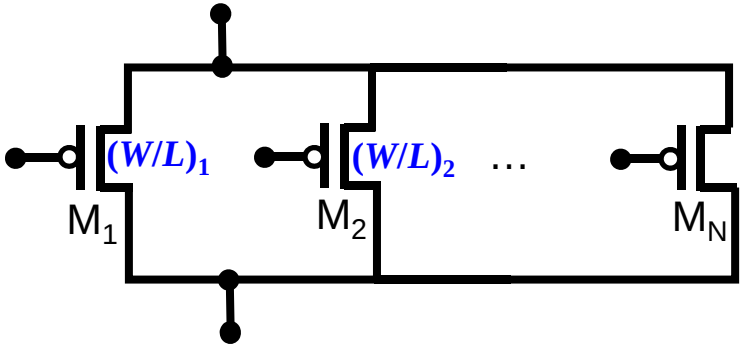
Construction of Logic Gate

Finally, we connect the PUN and PDN as follows:



Dynamic Performance: Switching Speed

- Rule of thumb for both NMOS and PMOS transistors in parallel:



Each switch is like a transistor with $R_{Qi} \propto 1/(W/L)_i$
Resistor in parallel $\Rightarrow R_{eq}=1/[1/R_{Q1}+\ldots+1/R_{QN}]$
$\Rightarrow R_{eq} \propto 1/[(W/L)_1+\ldots+(W/L)_N]=1/[(W/L)_{eq}]$

Worst Case
$\Rightarrow t_{pLH}$ is longest

Best Case
$\Rightarrow t_{pLH}$ is shortest

$(W/L)_{eq} = (W/L)_I$
where $(W/L)_I$ is the smallest
for $I \in \{1, 2, \ldots, N\}$

As propagation delay is inversely proportional to sizing, the smaller the sizing the longer it is. Also, there should only be one transistor pulling the output to high.

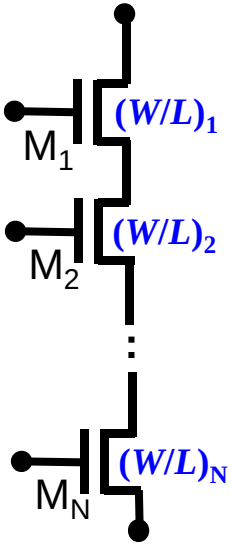
$(W/L)_{eq} = (W/L)_1 + (W/L)_2 + \ldots + (W/L)_N$

As propagation delay is inversely proportional to sizing, the larger the sizing, the shorter it is. Therefore, all transistors should be turned on and pulling the output to high.

Same concept applies for NMOS, but the NMOS is pulling the output to low.

Dynamic Performance: Switching Speed

- Rule of thumb for both NMOS and PMOS transistors in series:



Worst Case=Best Case
i.e. there is only one case where all transistors turn on and pull output to low

$$(W/L)_{eq} = \left[\frac{1}{(W/L)_1} + \frac{1}{(W/L)_2} + \ldots + \frac{1}{(W/L)_N} \right]^{-1}$$

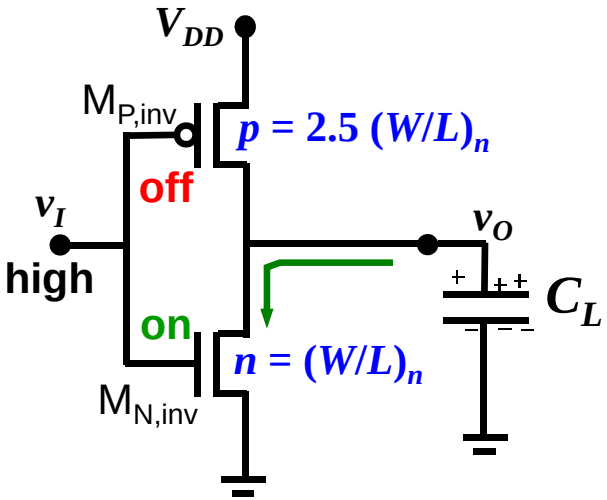
When transistors are in series, it is making the current more difficult to go through as there are so many transistors along the way⇒high resistance⇒smaller sizing.
Transistors in series reduce the equivalent transistor sizing and result in longer propagation delay.

Same concept applies for PMOS, but the PMOS is pulling the output to high.

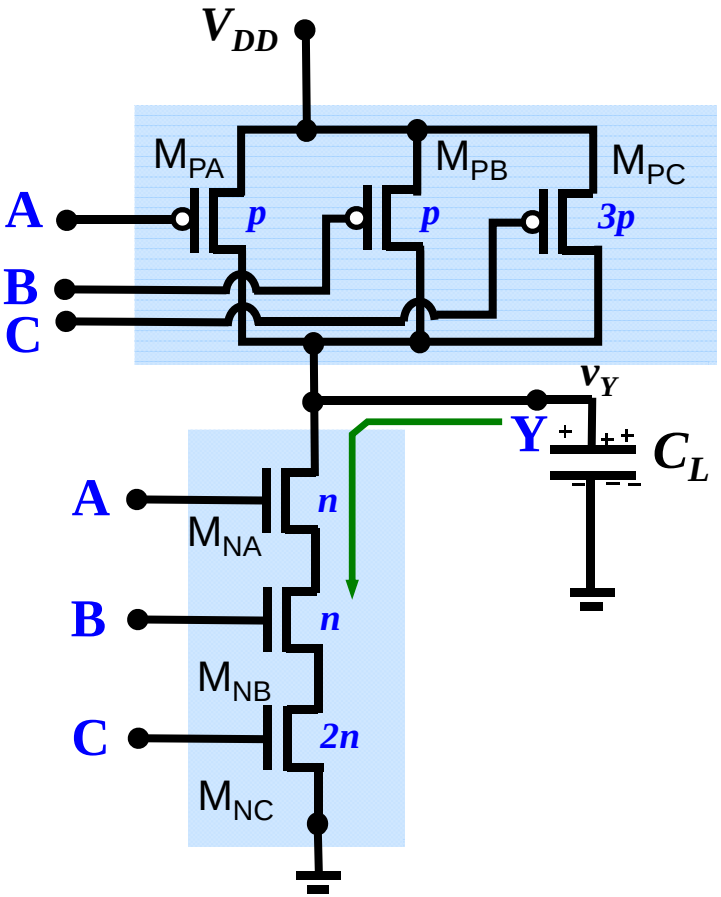
Each switch is like a transistor with
$R_{Qi} \propto 1/(W/L)_i$
Resistor in series⇒$R_{eq}=R_{Q1}+\ldots+R_{QN}$
⇒$R_{eq} \propto 1/[(W/L)_1]+\ldots+1/[(W/L)_N]=1/[(W/L)_{eq}]$

Example: 3 Input NAND Gate

- We would like to find out the worst case and best case t_{pLH} and t_{pHL} of 3 input NAND gate compared to a symmetric inverter.



p and n are just sizing constant to simplify the writing of W/L



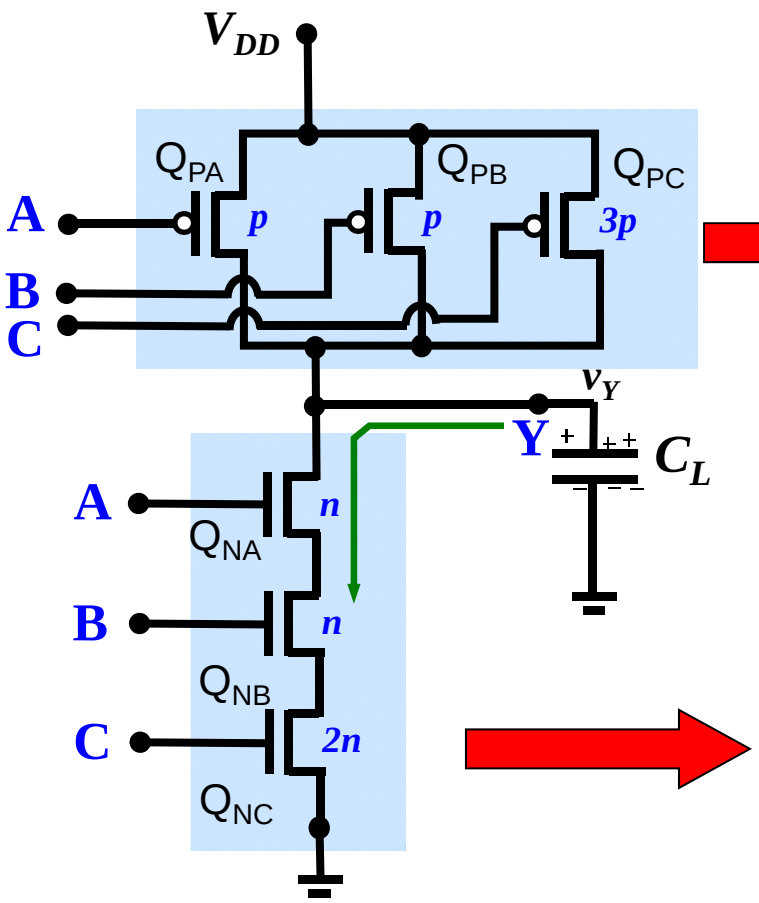
Inverter propagation delay

Proportionality constants for PMOS and NMOS

$$t_{pLH,inv} = \frac{k_1}{\mu_p p} \qquad t_{pHL,inv} = \frac{k_2}{\mu_n n}$$

Example: 3 Inputs NAND Gate

- We would like to find out the worst case and best case t_{pLH} and t_{pHL} of 3 inputs NAND gate compared to a symmetric inverter.



- For best case t_{pLH}, all transistors turn on.
- As they are in parallel, the $(W/L)_{eq}$=p+p+3p=5p

$$t_{pLH,NAND3,best} = \frac{k_1}{\mu_p 5p} \Rightarrow t_{pLH,NAND3,best} = \frac{1}{5} t_{pLH,INV}$$

- For worst case t_{pLH}, only one transistor with smallest size turns on.
- $(W/L)_{eq}$=p

$$t_{pLH,NAND3,worst} = \frac{k_1}{\mu_p p} \Rightarrow t_{pLH,NAND3,worst} = t_{pLH,INV}$$

- There is only one case for t_{pHL}, i.e. all transistors turn on.
- $(W/L)_{eq}=(1/n+1/n+1/2n)^{-1}=2n/5$

$$t_{pHL,NAND3} = \frac{k_2}{\mu_n \frac{2n}{5}} \Rightarrow t_{pHL,NAND3} = \frac{5}{2} t_{pHL,INV}$$

Lecture Summary

- Introduced CMOS inverter circuit.
- Calculate inverter propagation delay.
- Discussed Pull-Up Network (PUN) and Pull-Down Network (PDN) and their Complementary Functions.
- Discussed how to derive the propagation delay of complex gates in terms of symmetric inverter.

Reading Assignment

- **Reading: Reference Book (Sedra & Smith)**

  Chapter 9, pp. 901 – 921. (Logic)

  Chapter 4, pp. 426 – 429.