10.3 CMOS Logic Gate Circuits

Reading Assignment: *pp. 963-974*

Q: Can't we build a more **complex** digital device than a simple digital inverter?

A:

HO: CMOS Device Structure

Q:

A: HO: Synthesis of CMOS Gates

HO: Examples of CMOS Logic Gates

Example: CMOS Logic Gate Synthesis

Example: Another CMOS Logic Gate Synthesis

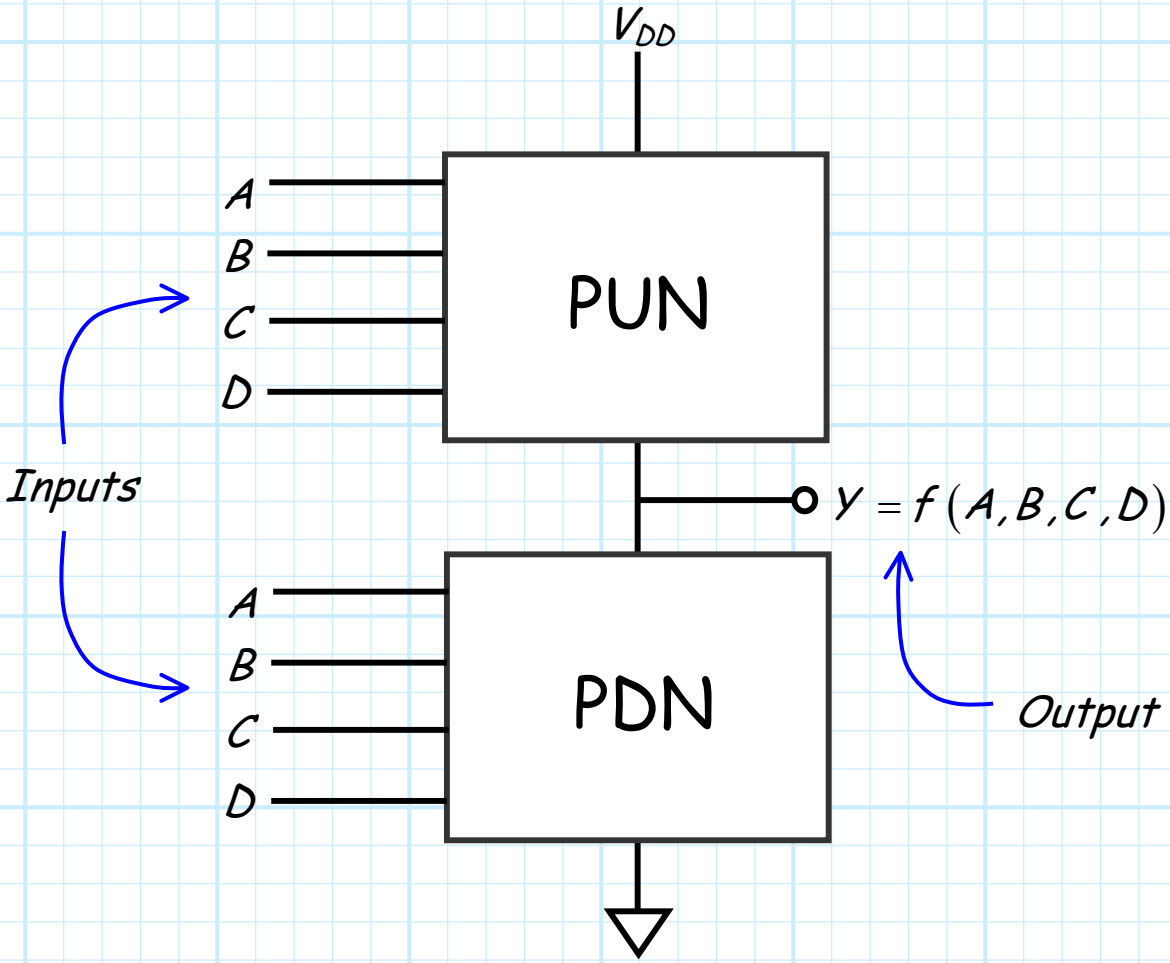
CMOS Device Structure

For every CMOS device, there are essentially **two** separate circuits:

1) *The Pull-Up Network*

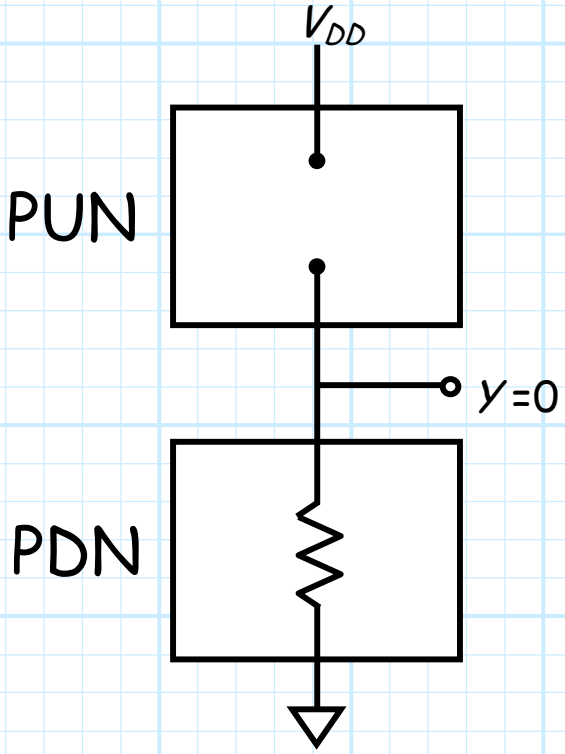
2) *The Pull-Down Network*

The basic **CMOS structure** is:



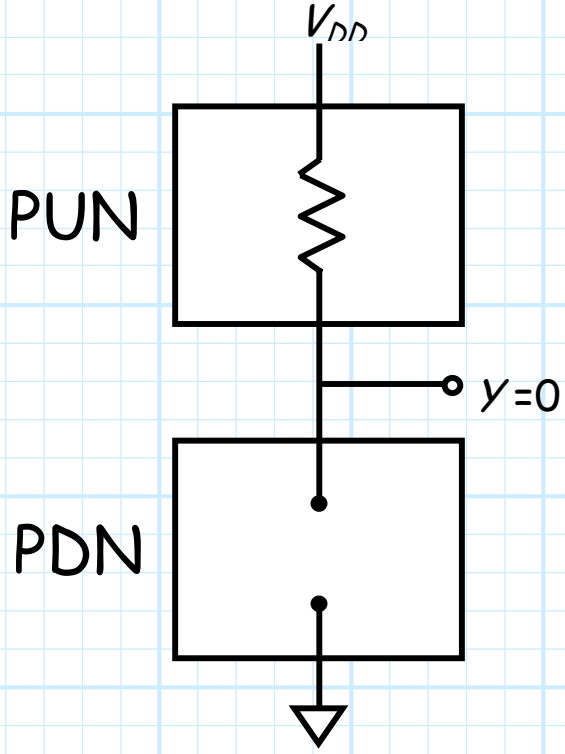
A CMOS logic gate **must** be in one of **two** states!

State 1: PUN is **open** and PDN is **conducting**.



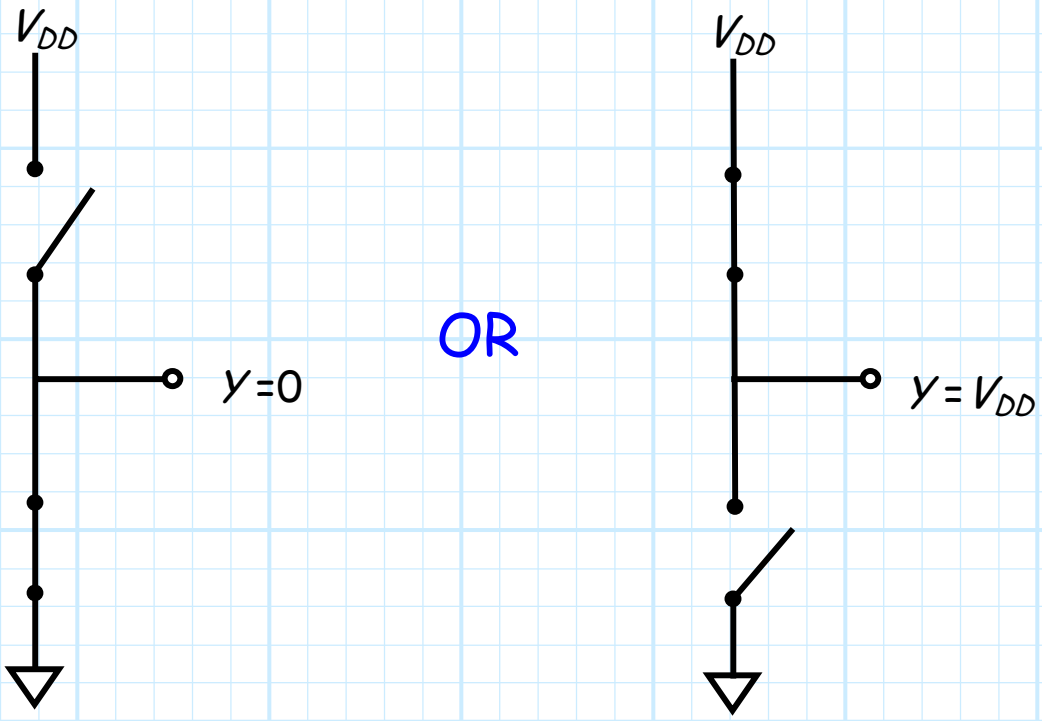
In this state, the output is **LOW** (i.e., $Y=0$).

State 2: PUN is **conducting** and PDN is **open**.



In this state, the output is **HIGH** (i.e., $Y=1$).

Thus, the PUN and the PDN essentially act as **switches**, connecting the output to **either** V_{DD} or to ground:

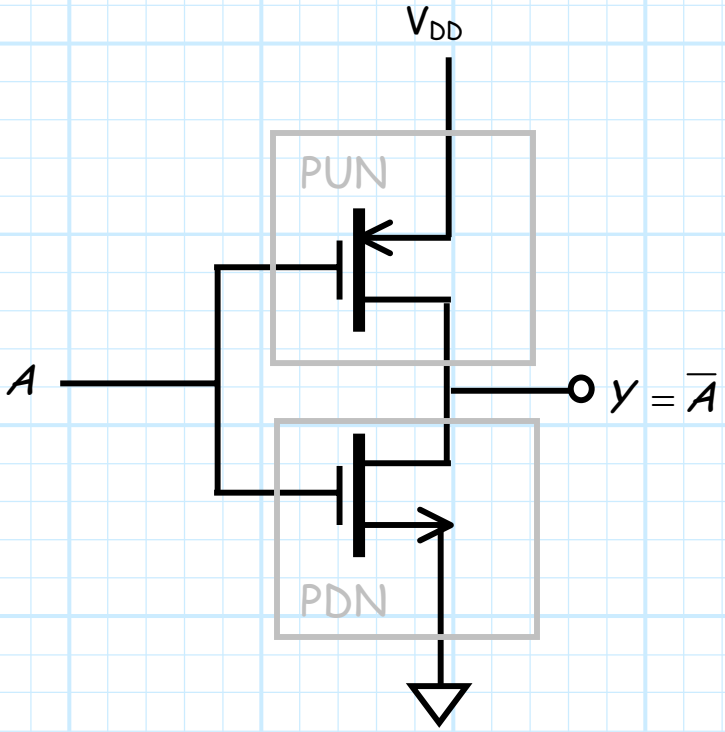


* Note that the key to proper operation is that **one** switch must be closed, while the **other** must be open.

* **Both** switches closed or **both** switches open would cause an **ambiguous** digital output!

* To prevent this from occurring, the PDN and PUN must be **complementary** circuits.

For example, consider the **CMOS inverter**:



For more **complex** digital CMOS gates (e.g., a 4-input OR gate), we find:

1) The PUN will consist of **multiple** inputs, therefore requires a circuit with **multiple PMOS** transistors.

2) The PDN will consist of **multiple** inputs, therefore requires a circuit with **multiple NMOS** transistors.

Synthesis of CMOS Gates

Let's consider the design synthesis of CMOS gates by considering the design synthesis of PUN and PDN separately.

PDN Design Synthesis

1. If the PDN is **conducting**, then the **output** will be **low**. Thus, we must find a Boolean expression for the **complemented output** $\overline{Y}$.

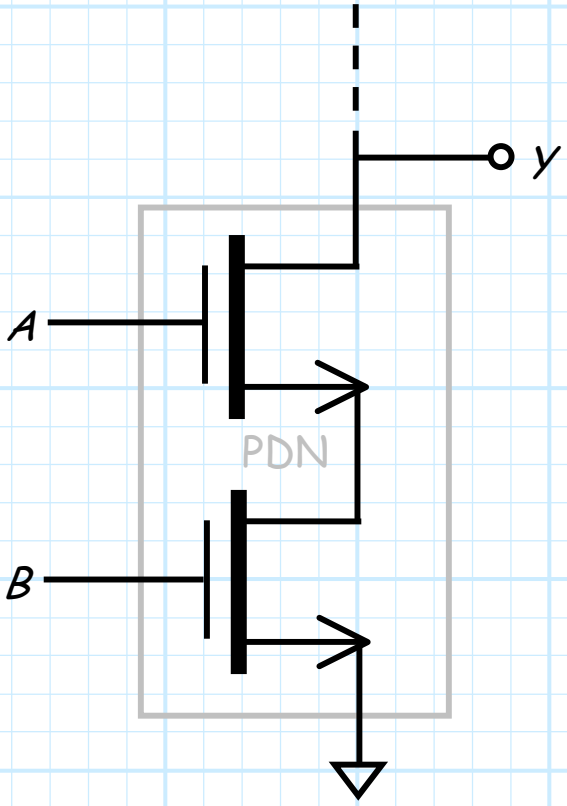
In turn, the PDN can only be conducting **if one or more** of the NMOS devices are **conducting**—and **NMOS** devices will be conducting (i.e., **triode** mode) when the **inputs are high** ($V_{GSN} = V_{DD}$).

Thus, we **must** express $\overline{Y}$ in terms of **un-complemented inputs** A, B, C, etc (i.e., $\overline{Y} = f(A,B,C)$).

e.g., → $\overline{Y} = A + BC$

This step may test our **Boolean algebraic** skills!

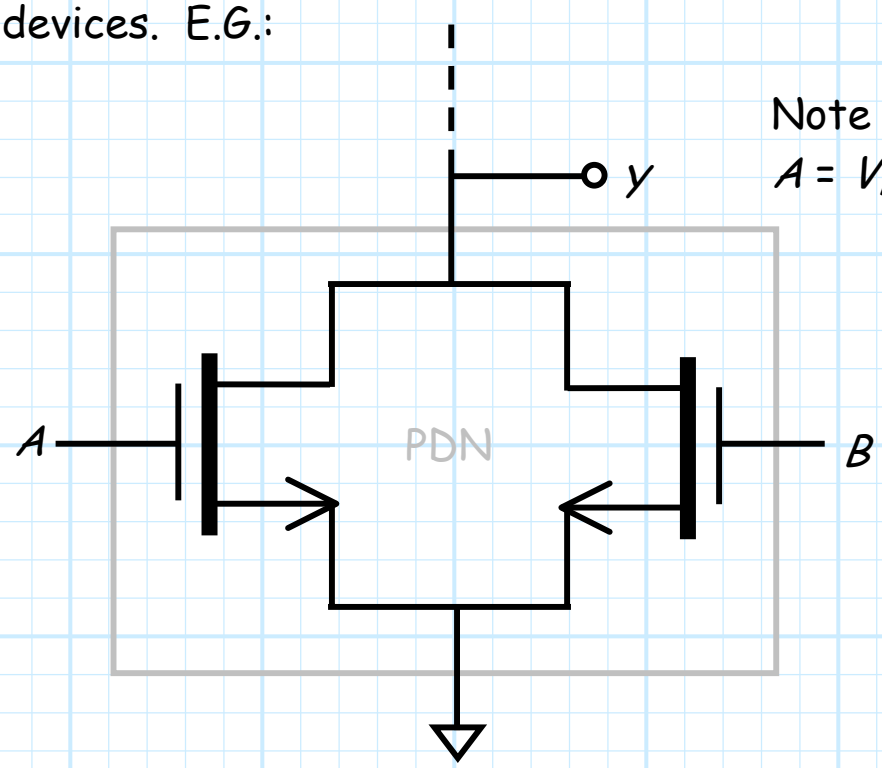
2. Then, we realize **AND** operations in $\overline{Y} = f(A,B,C)$ with **series NMOS** devices. E.G.:



Note that Y=0 if **both** $A = V_{DD}$ **AND** $B = V_{DD}$.

$$\therefore \quad \overline{Y} = AB$$

3. Likewise, we realize **OR** operations with **parallel NMOS** devices. E.G.:



Note that Y=0 if **either** $A = V_{DD}$ **OR** $B = V_{DD}$.

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

PUN Design Synthesis

1. If the PUN is **conducting**, then the **output** will be **high**. Thus, we must find a Boolean expression for the **un-complemented output** Y.

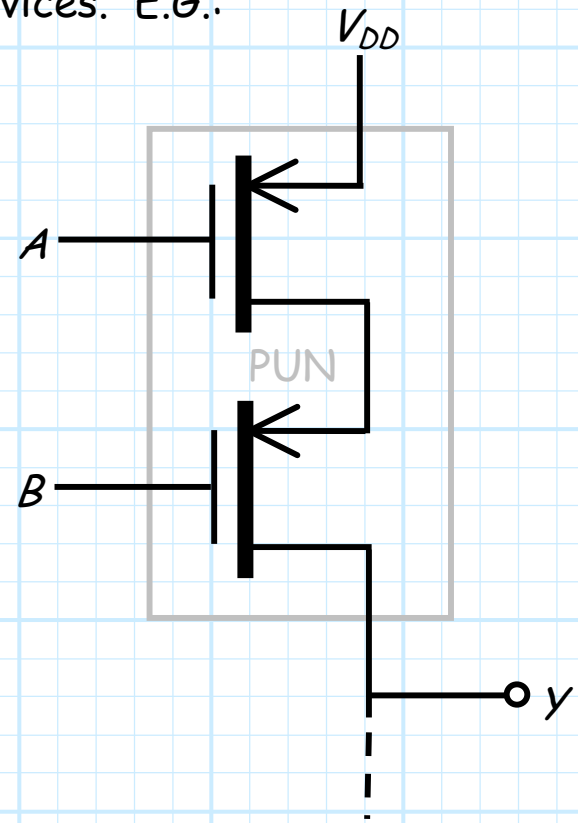
In turn, the PUN can only be conducting **if one or more** of the PMOS devices are **conducting**—and **PMOS** devices will be conducting (i.e., **triode** mode) when the **inputs are low** (V_{GSP}= $-V_{DD}$).

Thus, we **must** express Y in terms of complemented inputs $\overline{A}, \overline{B}, \overline{C}$, etc (i.e., $Y = f\left(\overline{A}, \overline{B}, \overline{C}\right)$).

e.g., → $Y = \overline{A} + \overline{B}\,\overline{C}$

This step may test our **Boolean algebraic** skills!

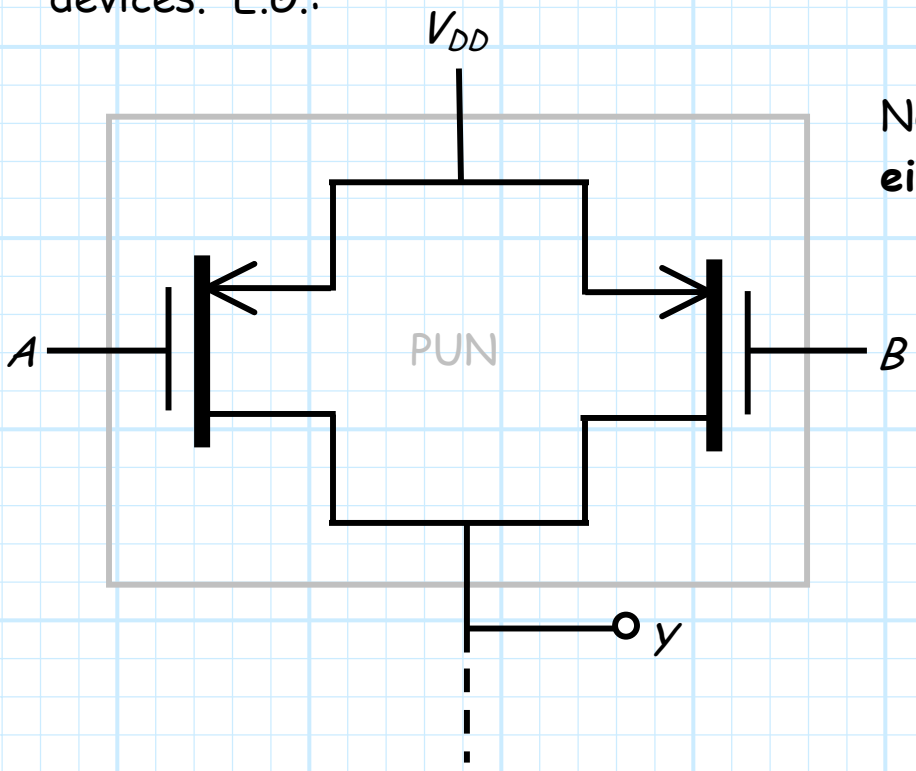
2. Then, we realize **AND** operations with **series PMOS** devices. E.G.:



Note that $Y = V_{DD}$ if **both** $A = 0$ **AND** $B = 0$.

$$\therefore \quad Y = \overline{A}\,\overline{B}$$

3. Likewise, we realize **OR** operations with **parallel PMOS** devices. E.G.:

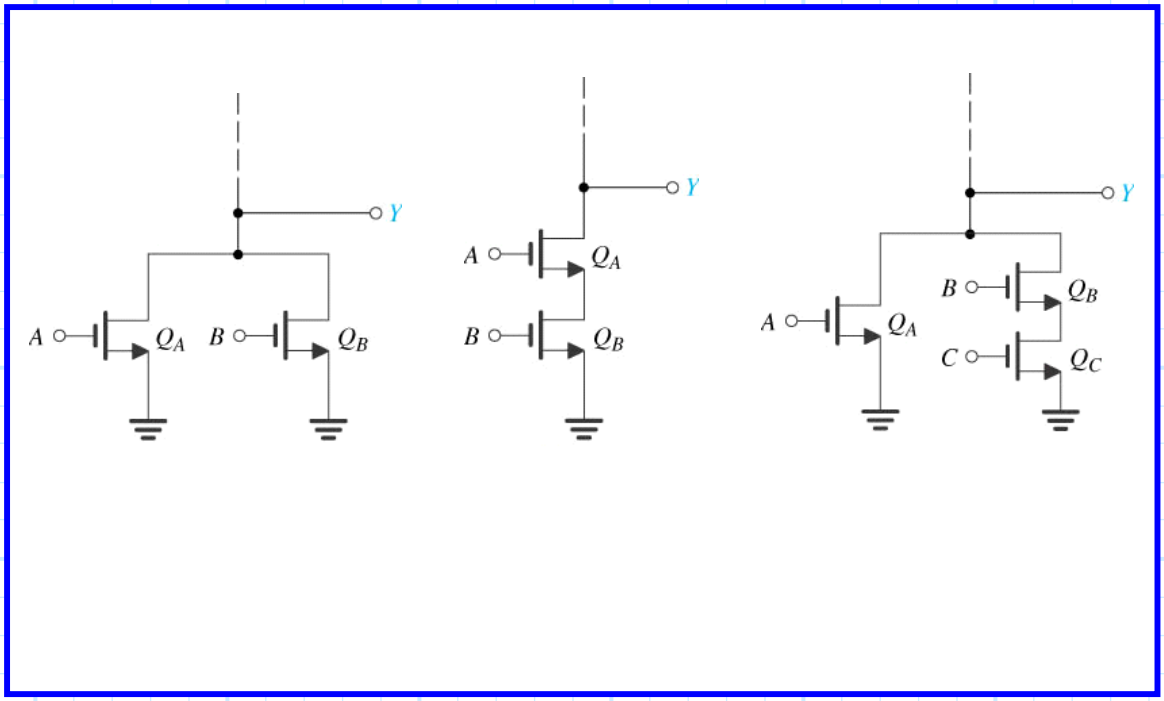


Note that $Y = V_{DD}$ if **either** $A = 0$ **OR** $B = 0$.

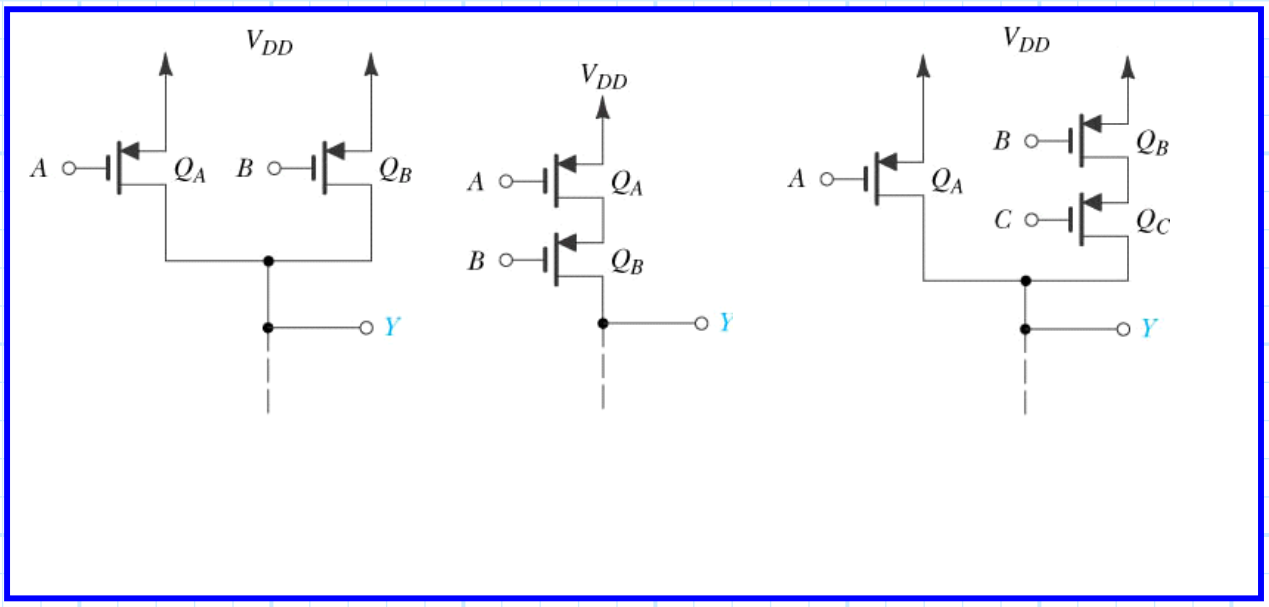
$$\therefore \quad Y = \overline{A} + \overline{B}$$

Examples of CMOS Logic Gates

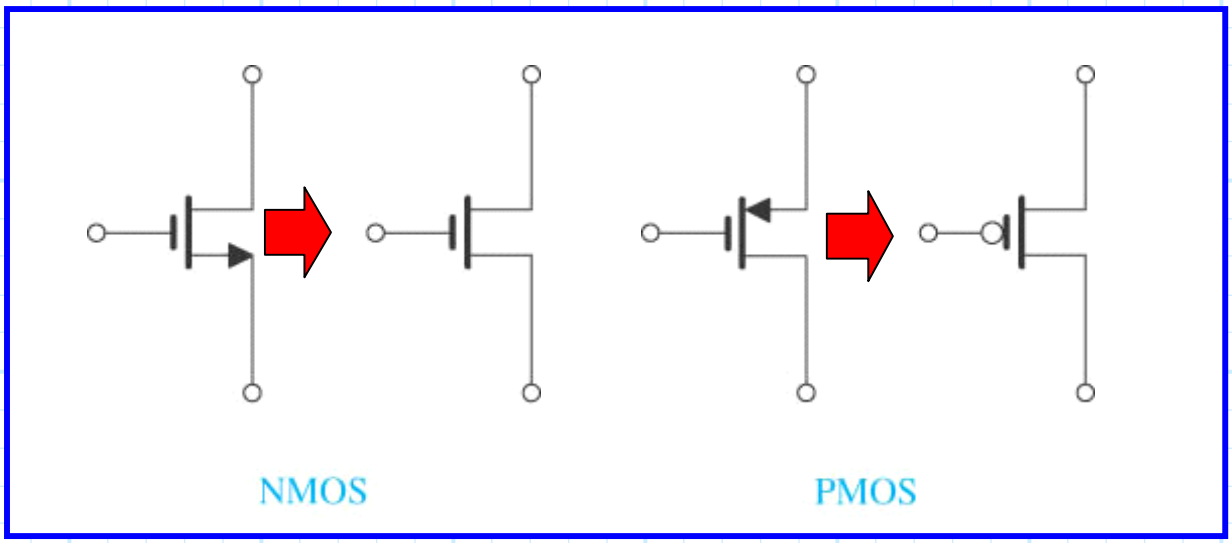
See if you can determine the **Boolean expression** that describes these **pull-down networks**:



See now if you can determine the **Boolean algebraic expression** for these **pull-up networks**:



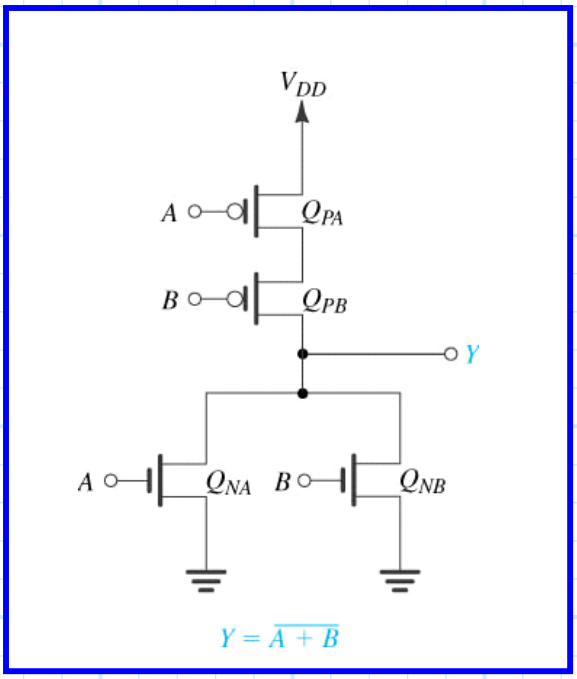
Now, we will make a simplifying **change of symbols**:

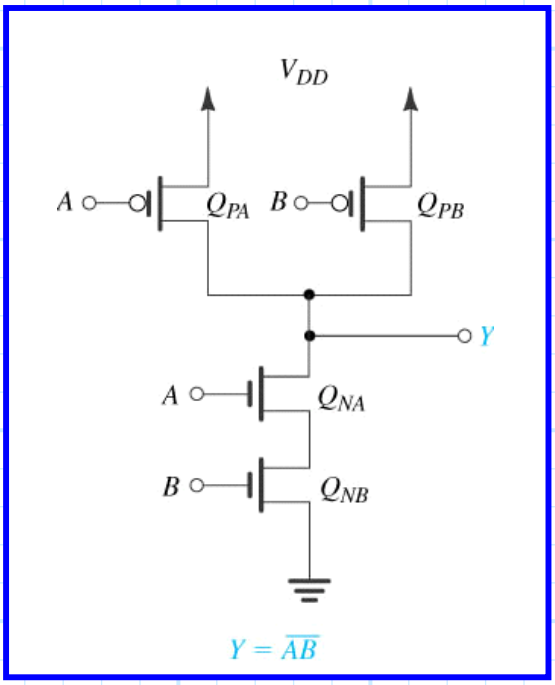


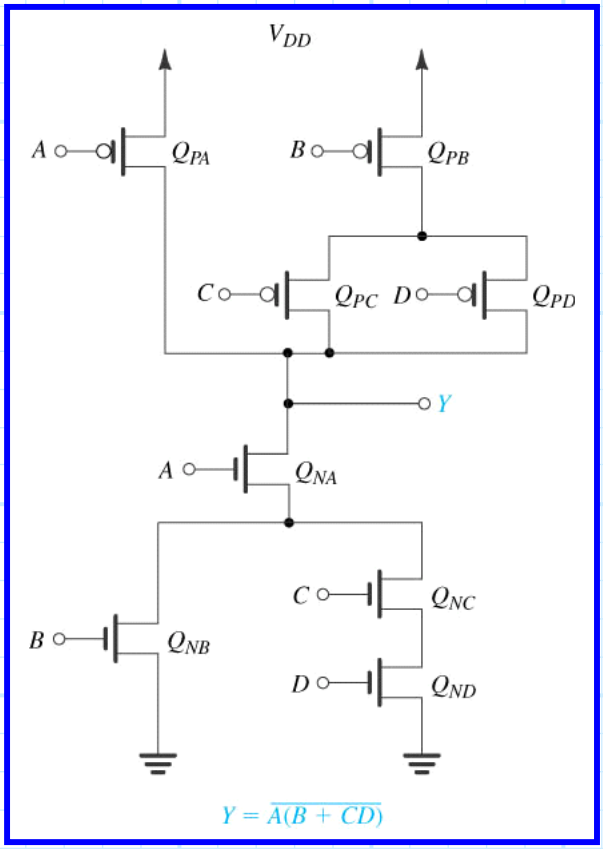
Effectively, these symbols represent the fact that we are now considering MOSFETs as **switches**, which can be placed either in an **open** state or a **conducting** state.

Note there are **two** kinds of "switches"—the ones that conduct when the input is high (i.e., **NMOS**) and ones that conduct when the input is low (i.e., **PMOS**).

And now consider **these** logic gates:







Note the PUN and the PDN for each of these circuits have **equivalent Boolean expressions** (make sure **you** see this!).

Example: CMOS Logic Gate Synthesis

Problem: **Design** a CMOS digital circuit that realizes the Boolean function:

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

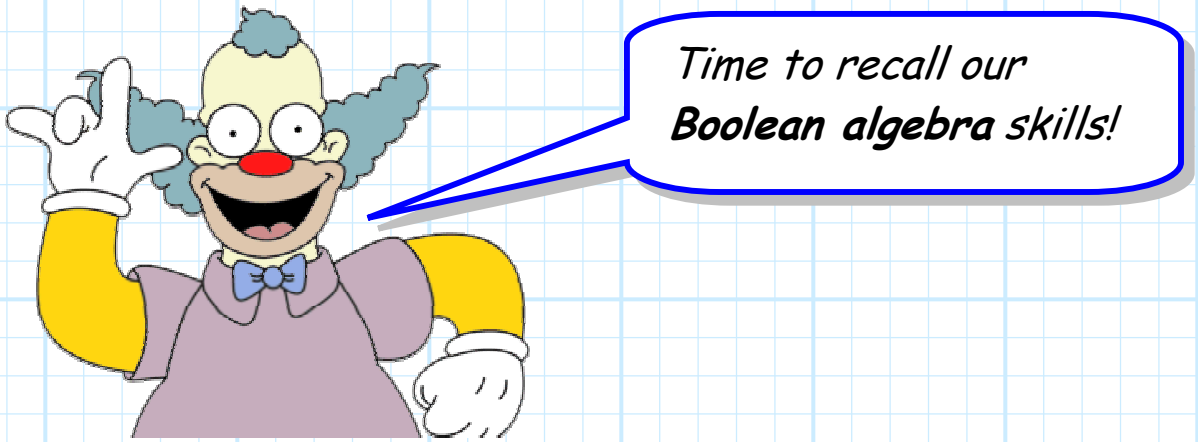
Solution: Follow the steps of the **design synthesis** handout!

Step1: Design the PDN

First, we must **rewrite** the Boolean function as:

$$\overline{Y} = f(A, B, C)$$

In other words, write the **complemented output** in terms of **un-complemented inputs**.



We must first complement this equation, and then apply **DeMorgan's Theorem** (several times!).

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

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

$$= \left(\overline{\overline{A + B}}\right)\left(\overline{\overline{A}\,\overline{C}}\right)$$

$$= (A + B)\left(\overline{\overline{A}} + \overline{\overline{C}}\right)$$

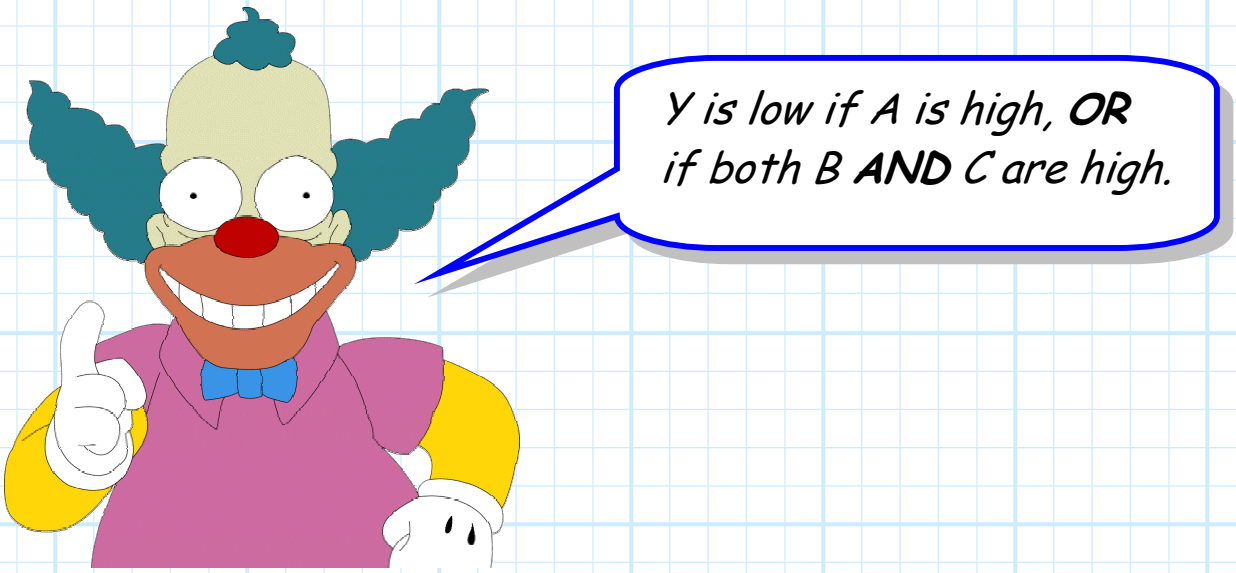
$$= (A + B)(A + C)$$

$$= AA + AC + BA + BC$$

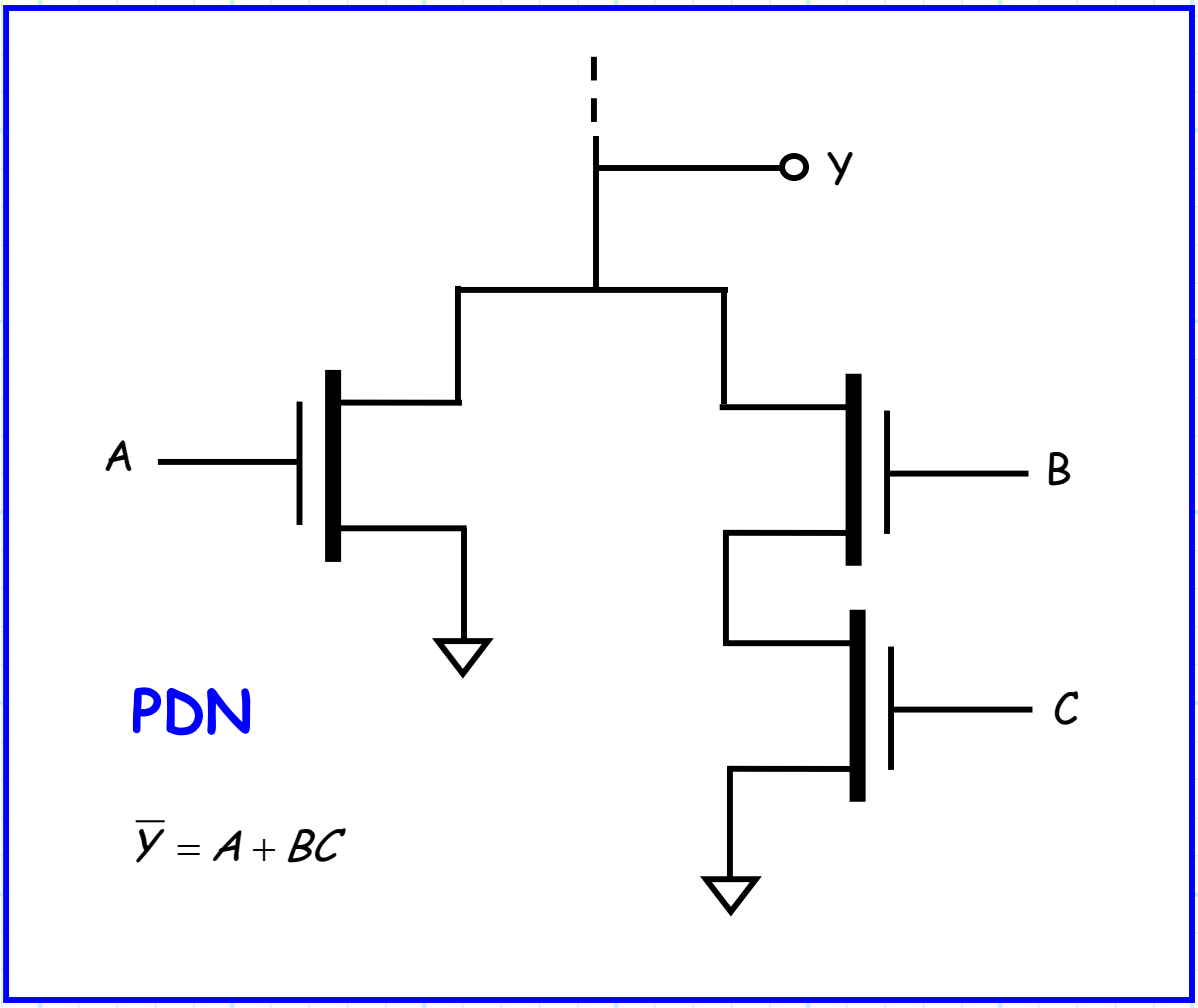
$$= A(A + B + C) + BC$$

$$= A + BC$$

Logically, this result says:



We can thus realize this logic with the following **NMOS** PDN:



Step2: Design the PUN

First, we must **rewrite** the Boolean function as:

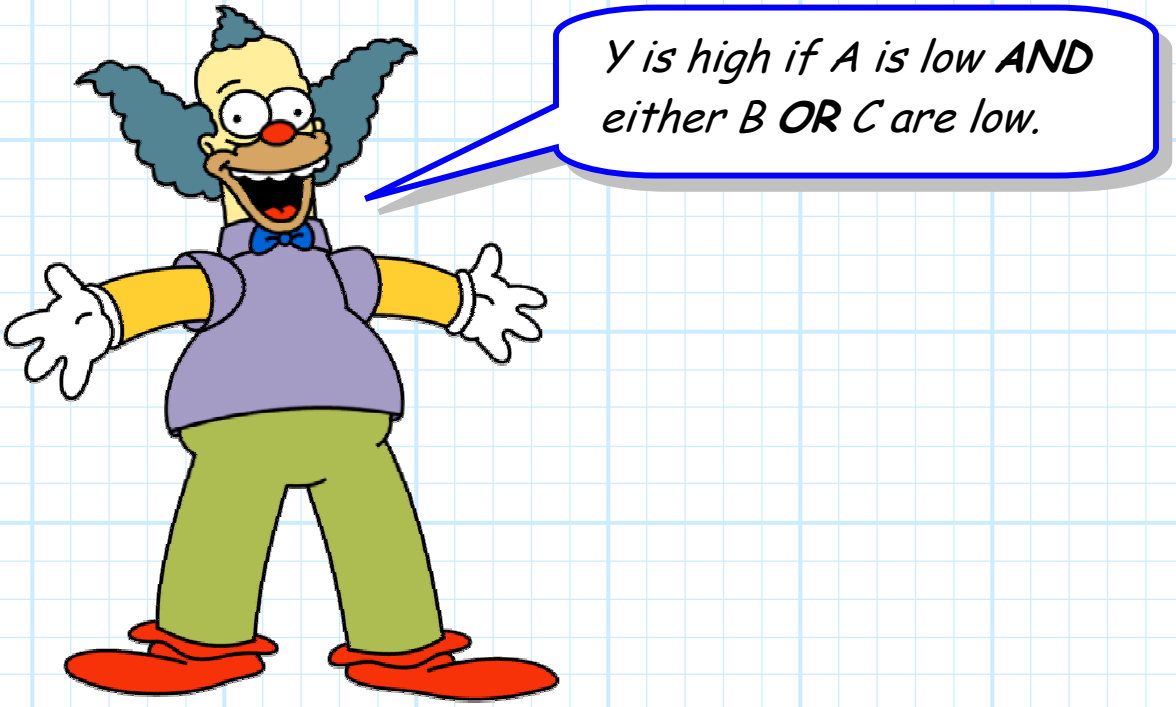
$$Y = f\left(\overline{A}, \overline{B}, \overline{C}\right)$$

In other words, write the **un-complemented output** in terms of **complemented inputs**.

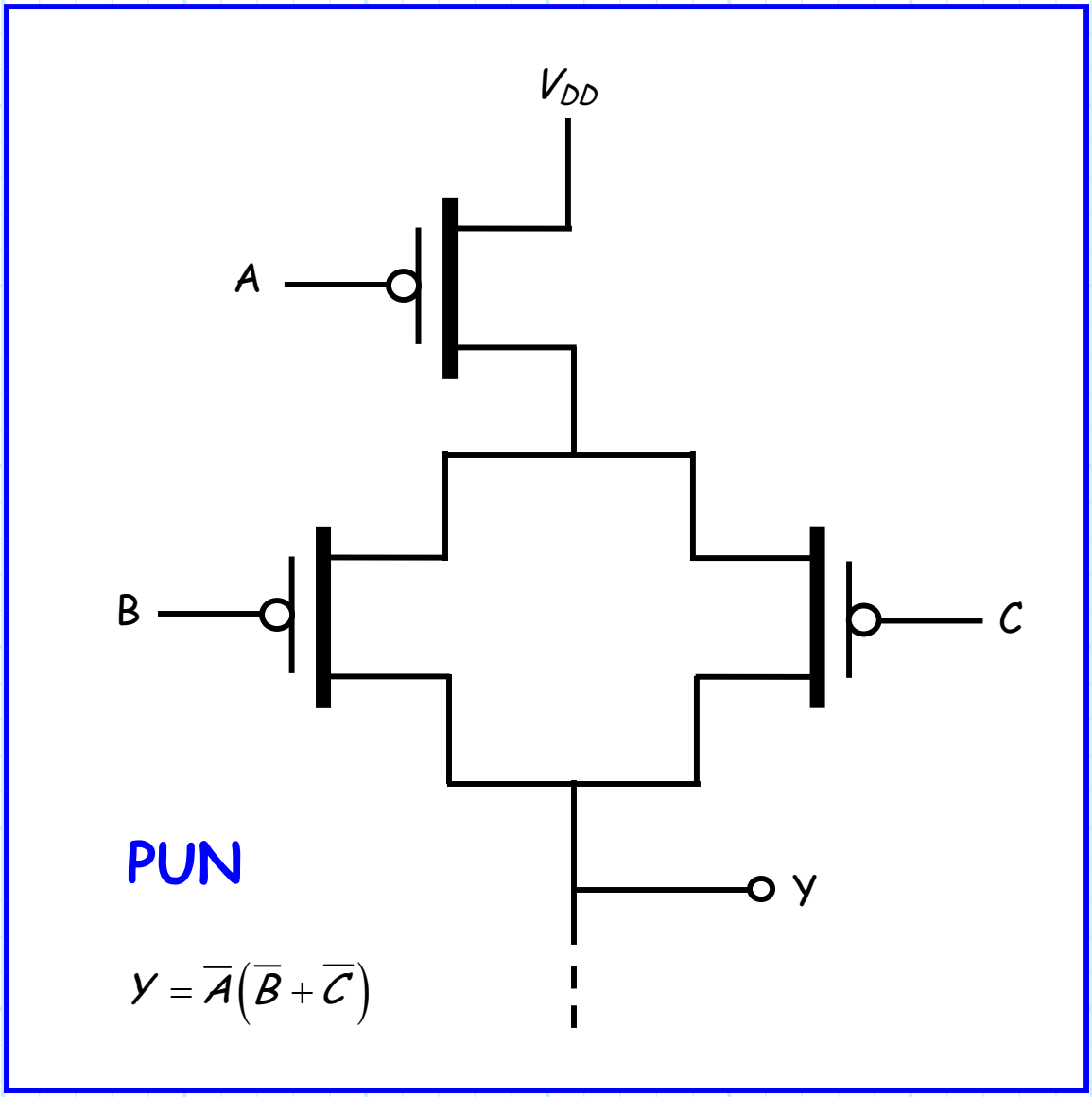
Again, using **DeMorgan's Theorem**:

$$\begin{aligned} Y &= \overline{A + B} + \overline{A}\,\overline{C} \\ &= \overline{A}\,\overline{B} + \overline{A}\,\overline{C} \\ &= \overline{A}\left(\overline{B} + \overline{C}\right) \end{aligned}$$

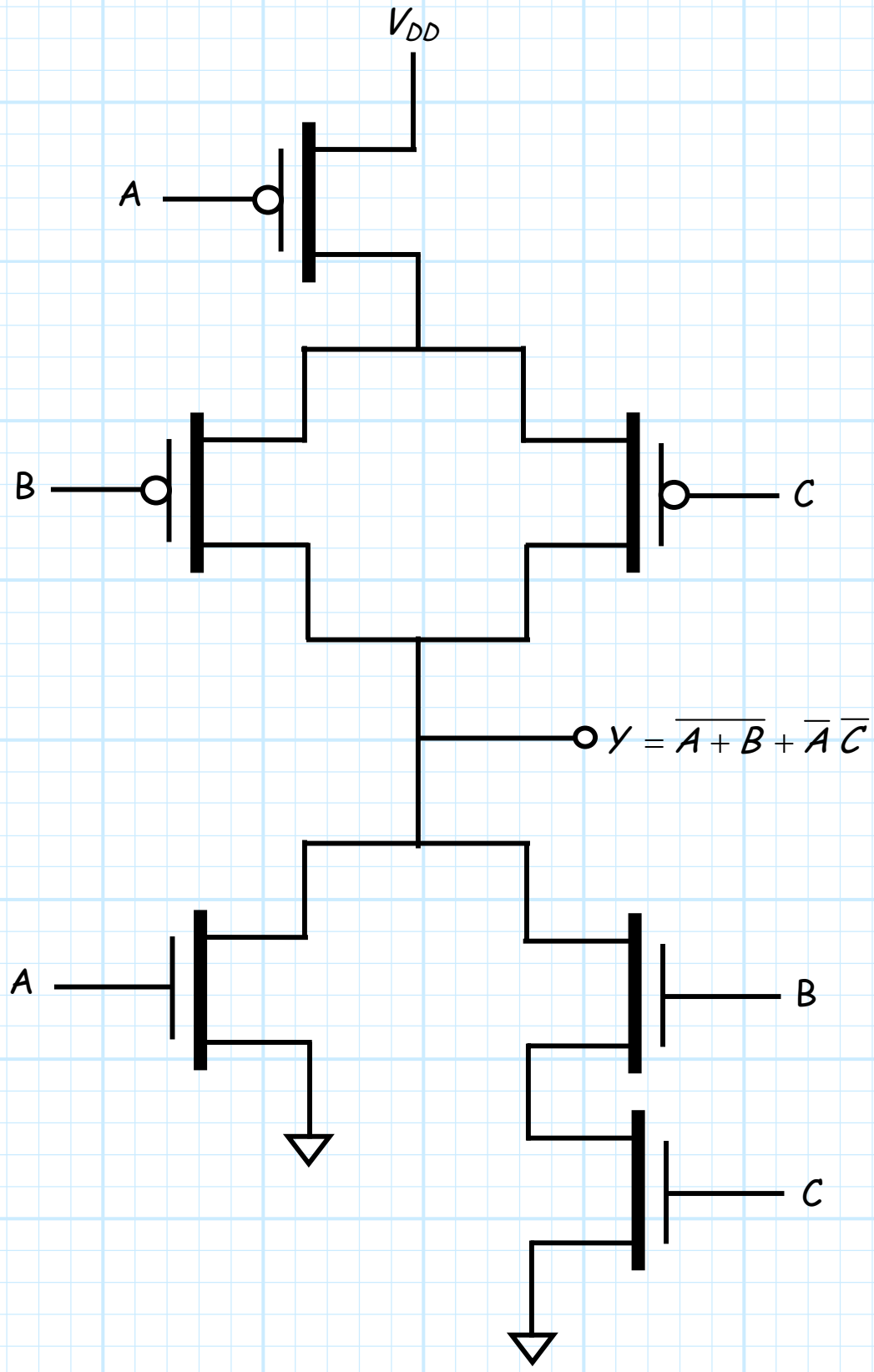
Logically, this result says:



We can thus realize this logic with the following **PMOS** PUN:



Thus, the **entire** CMOS realization is:


V_{DD}
A
B
C
$Y = \overline{A + B + \overline{A}\,\overline{C}}$
A
B
C

Example: Another CMOS Logic Gate Synthesis

Now let's design a gate that realizes this Boolean algebraic expression:

$$Y = \left(\overline{A} + \overline{B}\right)C$$

Step 1: *Design PDN*

First, let's rewrite Boolean expression as $\overline{Y} = f(A,B,C)$:

$$Y = \left(\overline{A} + \overline{B}\right)C$$

$$\overline{Y} = \overline{\left(\overline{A} + \overline{B}\right)C}$$

$$\overline{Y} = \left(\overline{\overline{A} + \overline{B}}\right) + \overline{C}$$

$$\overline{Y} = AB + \overline{C}$$

Q: *Yikes! We cannot write this expression explicitly in terms of **uncomplemented** inputs A, B, and C ! The input C appears as $\overline{C}$ in the expression. What do we do **now**?*

A: An easy problem to solve! We can essentially make a substitution of variables:

$$C' = \overline{C}$$

And thus:

$$\overline{Y} = AB + C'$$

Therefore, the inputs to this logic gate should be A, B, and C' (i.e, A, B, and the complement of C).

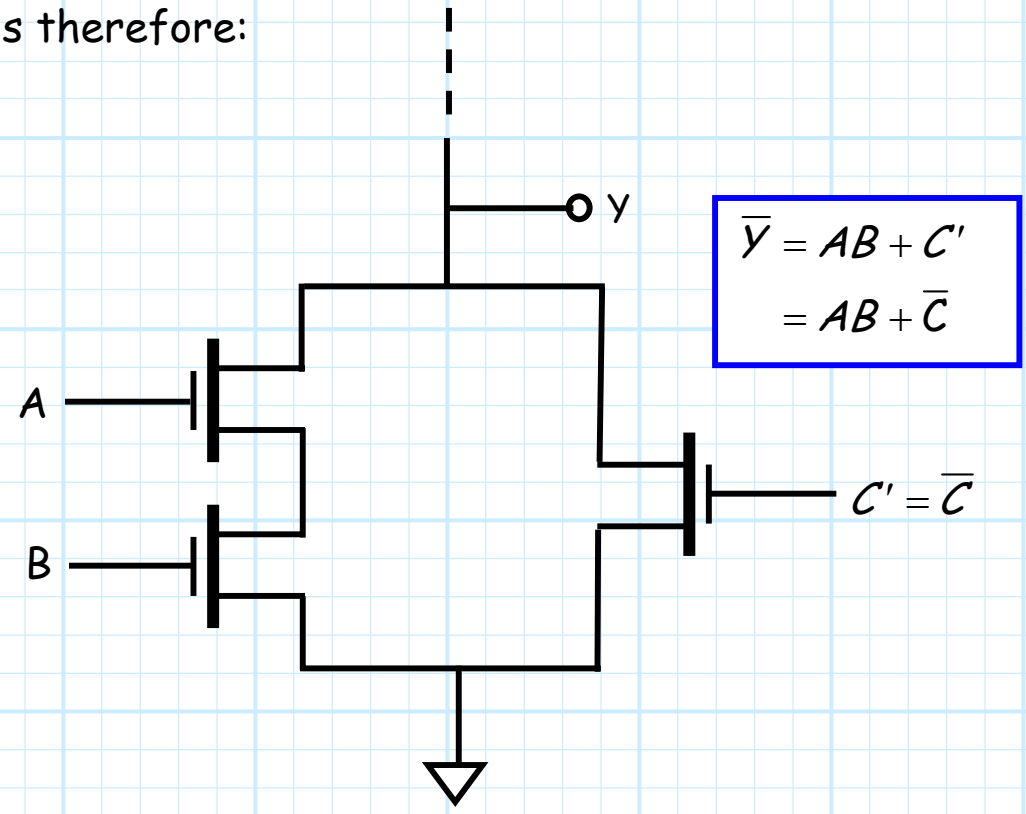
Note that this Boolean expression "says" that:

"The ouput is low if either,A AND B are both high, OR C' is **high**"

Of course another way of "saying" this is:

"The output is low if either A AND B are both high, OR C is **low**"

The PDN is therefore:



Step 2: *Design the PUN*

Note we have as similar problem as before—the expression for Y **cannot** explicitly be written in terms of complemented inputs $\overline{A}$, $\overline{B}$, and $\overline{C}$:

$$Y = \left(\overline{A} + \overline{B}\right)C$$

Note we can again solve this problem by using the same substitution of variable C:

$$C' = \overline{C}$$
$$\overline{C'} = C$$

Therefore:

$$Y = \left(\overline{A} + \overline{B}\right)\overline{C'}$$
$$= \left(\overline{A} + \overline{B}\right)C$$

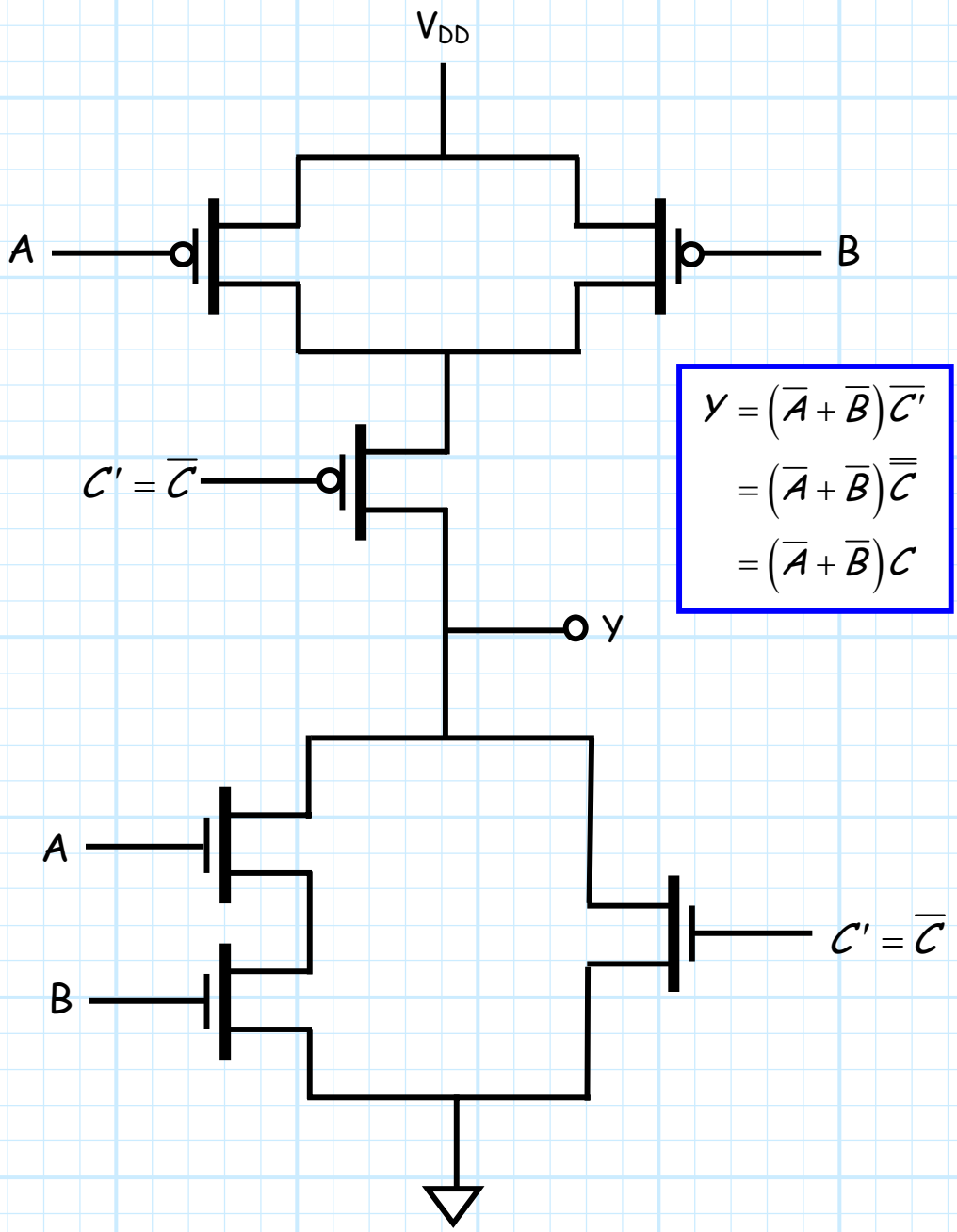
This expression "says" that:

"The output will be high if, either A OR B are low, AND C' is **low**"

Which is equivalent to saying:

"The output will be high if, either A OR B are low, AND C is **high**"

The CMOS digital logic device is therefore:



$$Y = \left(\overline{A} + \overline{B}\right)\overline{C'}$$
$$= \left(\overline{A} + \overline{B}\right)\overline{\overline{C}}$$
$$= \left(\overline{A} + \overline{B}\right)C$$