

CSE460: VLSI Design

Lecture 3

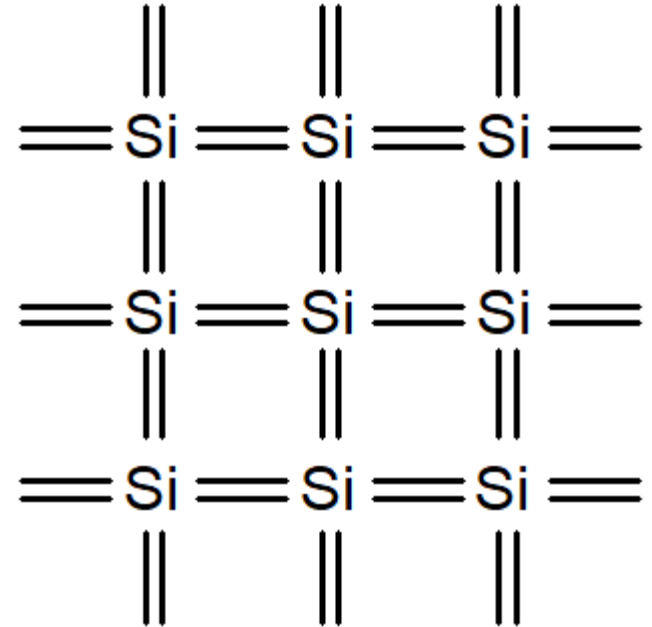
CMOS Logic

Contents

- Silicon lattice structure
- p-type and n-type semiconductors, p-n junction diode
- nMOS and pMOS transistors
- Transistors as switches
- Basic logic gate review
- CMOS gate structure
- Pull-up network (PUN) and pull-down network (PDN)
- Series and parallel PUN & PDN networks
- CMOS implementation of AOI, OAI and other logic gates

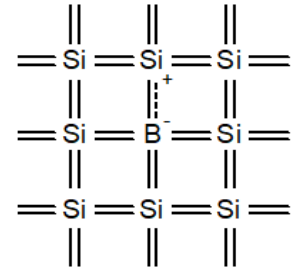
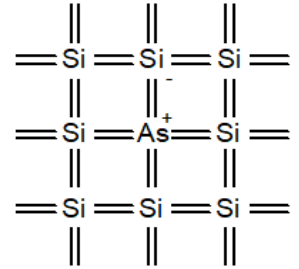
Silicon Lattice

- **Silicon is a semiconductor**
- Transistors are built on a silicon substrate
- Silicon is a **Group IV** material
 - Si (16) $\rightarrow 1s^2 2s^2 2p^6 3s^2 3p^2$
- Forms crystal lattice with covalent bonds to four neighbors
- Pure silicon has no free carriers and conducts poorly
- *How to increase its conductivity?*



Doped (Impure) Silicon

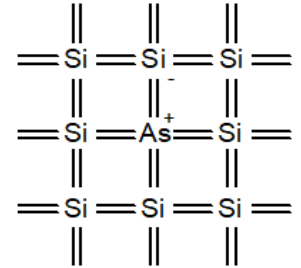
- *How to increase the conductivity of a pure semiconductor (usually Group IV materials)?*
- Answer: by adding impurity! (dopants)
- Adding dopants increases the conductivity
- **Group V** dopants: provides *free* “**electron**” to the pure semiconductor (Ex: Arsenic)
 - Called the n-type semiconductor afterwards
- **Group III** dopants: provides *free* “**holes**” to the pure semiconductor (Ex: Boron)
 - Called the p-type semiconductor afterwards



Concept Review

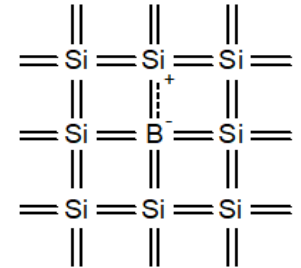
- n-type semiconductor

- Carrier: electrons
- *Net electrical charge of the material stays the same (**electrically neutral**)*
- Difference with pure semiconductor: it has more free electrons, even after forming bonds with the neighboring atoms
- Free electrons increase conductivity



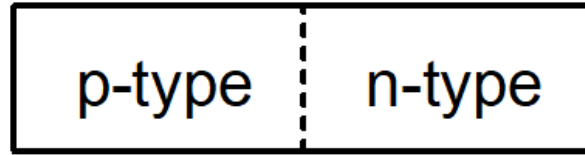
- p-type semiconductor

- Carrier: holes (absence of electrons)
- *Net electrical charge of the material stays the same (**electrically neutral**)*
- Difference with pure semiconductor: it has empty electron spaces (called holes) in certain bonds with the neighbouring atoms
- Electrons from neighbouring bonds try to fill up the empty spaces by creating a flow of electrons, hence increasing the conductivity



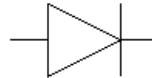
Concept Review

- A junction between p-type and n-type semiconductor forms a diode.
- Current flows only in one direction
- p-type to n-type, ~~n-type to p-type~~



anode

cathode

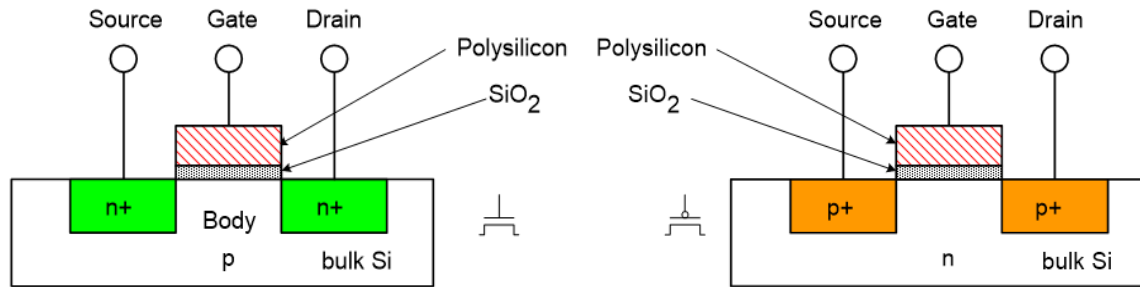


Electrical Voltage \Leftrightarrow Logic Conversion

- **V_{DD} = Logical High (1)**
- **GND = Logical Low (0)**
- $V_{DD} = 5\text{ V}, 3.3\text{ V}, 2.5\text{ V}, 1.8\text{ V}, 1.5\text{ V}, 1.2\text{ V}, 1.0\text{ V}, \dots$ (electrical voltage)
- GND = 0 V (electrical voltage)
- V_{DD} has decreased in modern processes
 - High V_{DD} would damage modern tiny transistors
 - Lower V_{DD} saves power

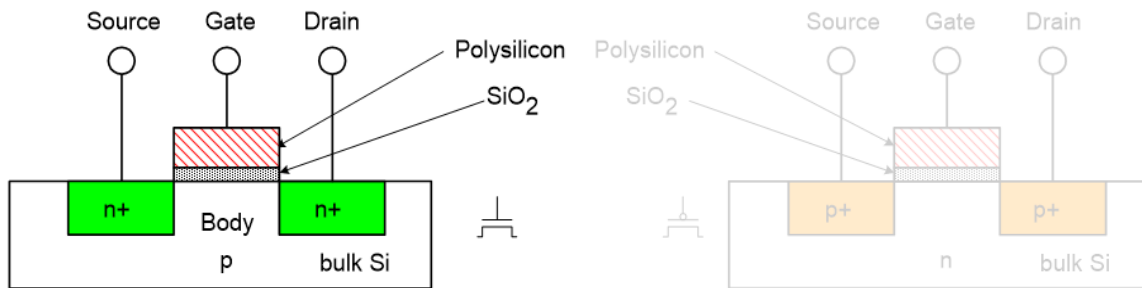
MOS Transistor

- Four terminals: gate, source, drain, body
 - gate, source, body -> conductors, SiO_2 (oxide) -> insulator
 - **source and drain are identical**, can be used interchangeably, no difference in construction
- Body is tied to the source: **functionally a 3 terminal device**
- Depending on the *voltage applied* to the gate terminal, source and drain are either electrically **connected** or **disconnected**



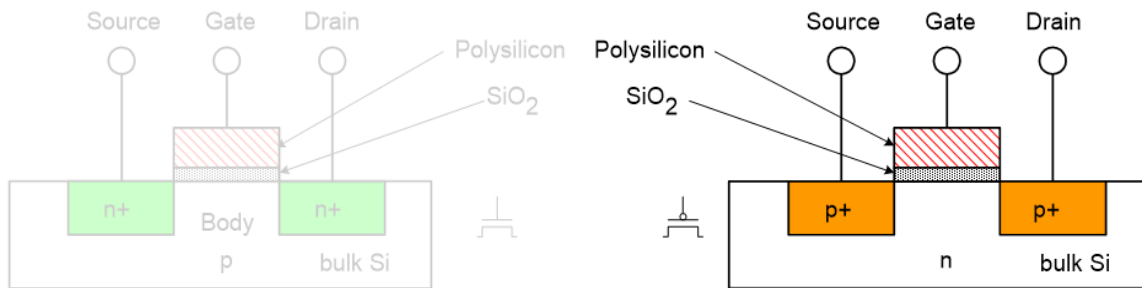
nMOS Transistor

- **Low voltage (GND)** applied to the gate: TRANSISTOR is **OFF**
 - source and drain are **disconnected** (no channel between them)
 - **no current flows**, transistor is **OFF**
- **High voltage (V_{DD})** applied to the gate: TRANSISTOR is **ON**
 - source and drain are **connected** by an *n-channel*, right under the gate oxide (SiO_2)
 - **current flows from drain to source**, transistor is **ON**
 - details on the working mechanism: later in this course!



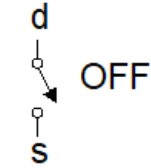
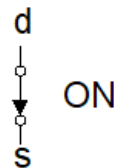
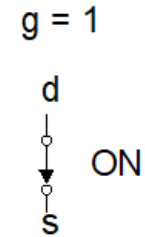
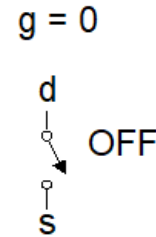
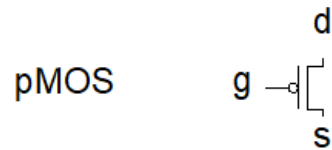
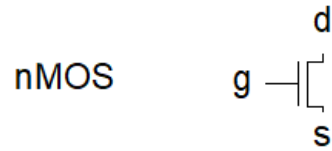
pMOS Transistor

- **Low voltage (GND)** applied to the gate: TRANSISTOR is **ON**
 - source and drain are **connected** by a *p-channel*, right under the gate oxide (SiO_2)
 - **current flows from source to drain (opposite direction of nMOS)**, transistor is **ON**
 - details on the working mechanism: later in this course!
- **High voltage (V_{DD})** applied to the gate: TRANSISTOR is **OFF**
 - source and drain are **disconnected** (no channel between them)
 - **no current flows**, transistor is **OFF**

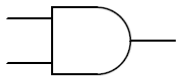


MOS Transistors as Switches

- Electrically controlled switches
- Voltage at gate controls path from source to drain

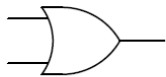


Basic logic gates



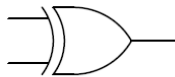
AND

A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1



OR

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1



XOR

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0



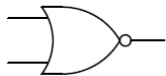
NOT

Input	Output
0	1
1	0



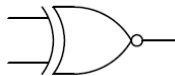
NAND

A	B	Output
0	0	1
0	1	1
1	0	1
1	1	0



NOR

A	B	Output
0	0	1
0	1	0
1	0	0
1	1	0

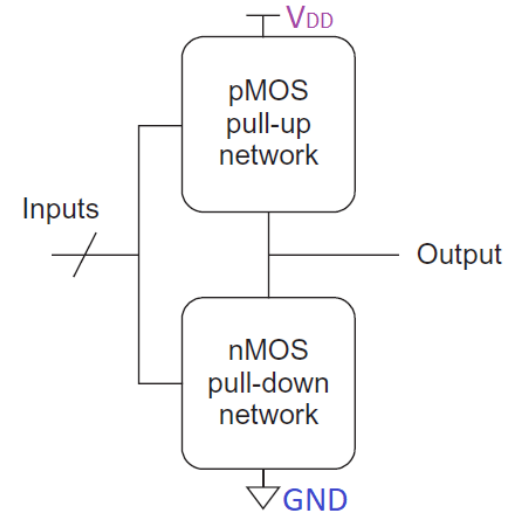


XNOR

A	B	Output
0	0	1
0	1	0
1	0	0
1	1	1

CMOS

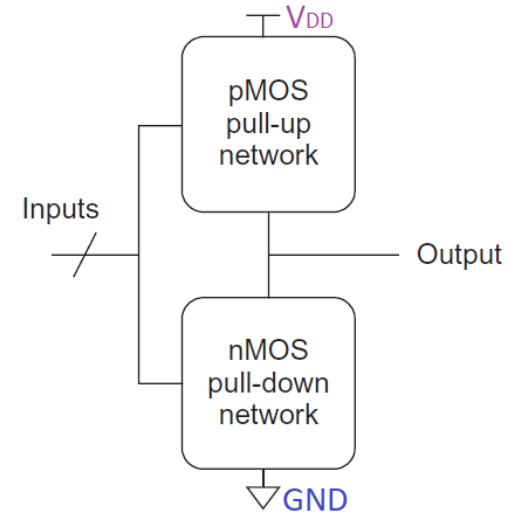
- CMOS can be used to build both simple and compound logic gates
- Consists of 2 networks
 - pMOS pull-up network
 - nMOS pull-down network
- pMOS & nMOS both used in the same structure -> CMOS



CMOS

- Depending on inputs the networks can be ON or OFF
- Network **ON**: Network is *conducting* electricity
- Network **OFF**: Network is *not conducting* electricity
- Depending on the inputs, 4 possible combinations of the 2 networks are possible:

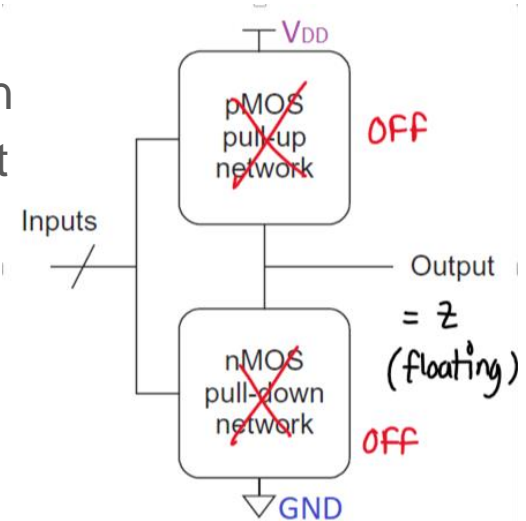
	Pull-up OFF	Pull-up ON
Pull-down OFF	Output = Z (float)	Output = 1 (High)
Pull-down ON	Output = 0 (Low)	Output = X (indeterminate)



CMOS gate structure

- Pull-up: **OFF**
- Pull-down: **OFF**
- Output is very high impedance/Z/floating/no connection
- Useful when we want to detach some part of our circuit

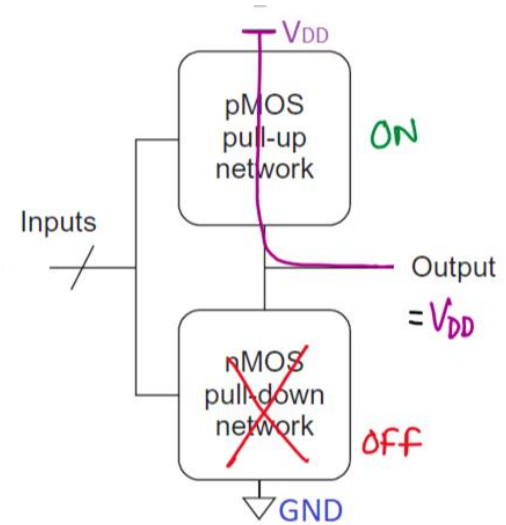
	Pull-up OFF	Pull-up ON
Pull-down OFF	Output = Z (float)	Output = 1 (High)
Pull-down ON	Output = 0 (Low)	Output = X (indeterminate)



CMOS gate structure

- Pull-up: **ON**
- Pull-down: **OFF**
- Output is *pulled-up* to V_{DD}
- Useful when we want to output **logical high (1)**

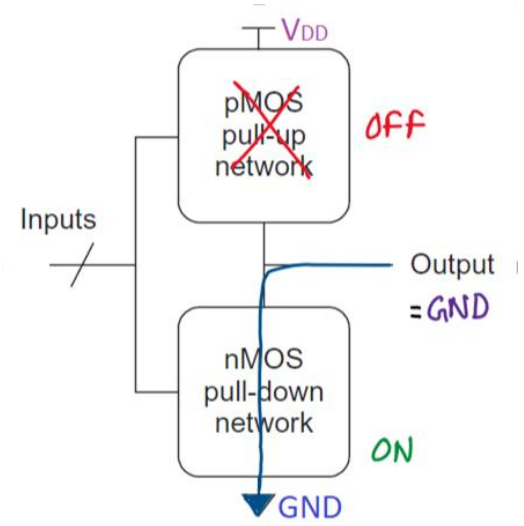
	Pull-up OFF	Pull-up ON
Pull-down OFF	Output = Z (float)	Output = 1 (High)
Pull-down ON	Output = 0 (Low)	Output = X (indeterminate)



CMOS gate structure

- Pull-up: **OFF**
- Pull-down: **ON**
- Output is *pulled-down* to **GND**
- Useful when we want to output **logical low (0)**

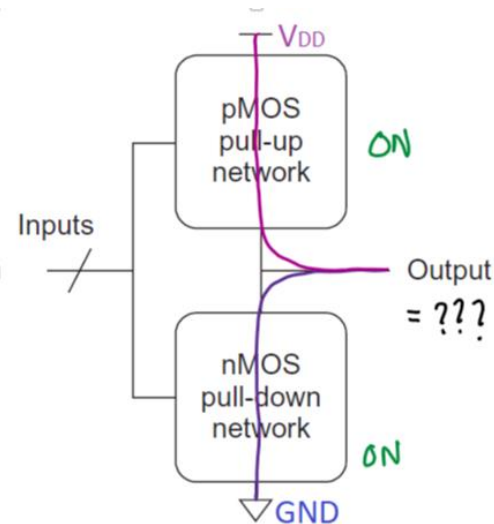
	Pull-up OFF	Pull-up ON
Pull-down OFF	Output = Z (float)	Output = 1 (High)
Pull-down ON	Output = 0 (Low)	Output = X (indeterminate)



CMOS gate structure

- Pull-up: ON
- Pull-down: ON
- Output is indeterminate
 - somewhere between V_{DD} and GND , we don't know for sure
- Usually a faulty condition

	Pull-up OFF	Pull-up ON
Pull-down OFF	Output = Z (float)	Output = 1 (High)
Pull-down ON	Output = 0 (Low)	Output = X (indeterminate)

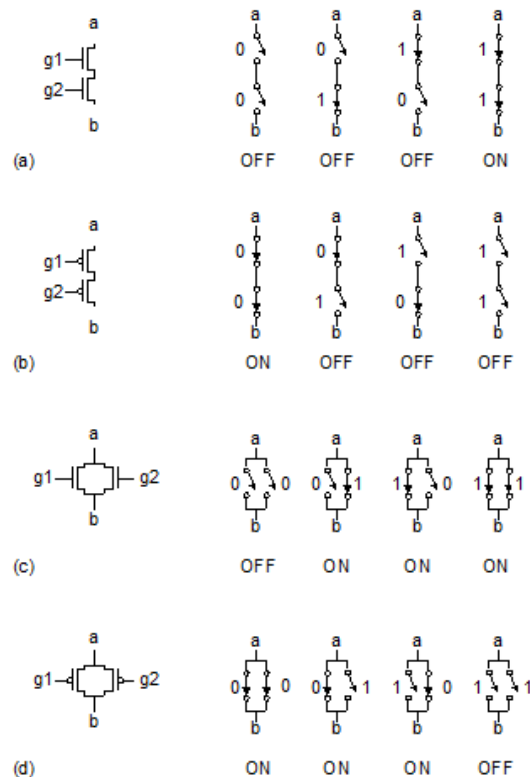


Pull-up & Pull-down Networks

- Simplest pull-up network: a single pMOS
- Complex pull-up network can be build from different series and parallel combinations of pMOS transistors
- Simplest pull-down network: a single nMOS
- Complex pull-down network can be build from different series and parallel combinations of nMOS transistors

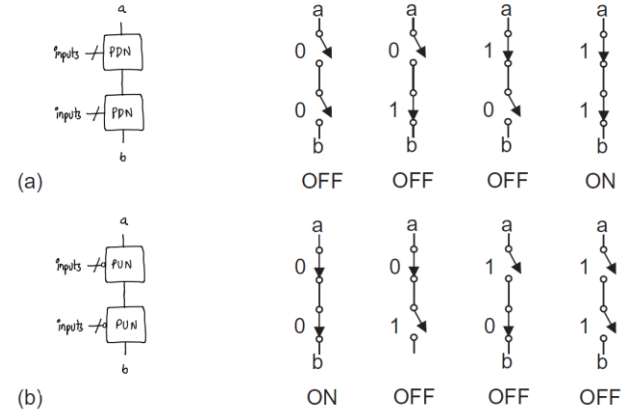
Series and Parallel Networks: *Simple* case

- Pull-up network: a single pMOS
 - High voltage at gate input: **OFF**
 - Low voltage at gate input: **ON**
- Pull-down network: a single nMOS
 - High voltage at gate input: **ON**
 - Low voltage at gate input: **OFF**
- *Series* combination of networks
 - *Every network* must be **ON** to make the **whole network conducting** (figure: a, b)
- *Parallel* combination of networks
 - *At least one network* must be **ON** to make the **whole network conducting** (figure: c, d)



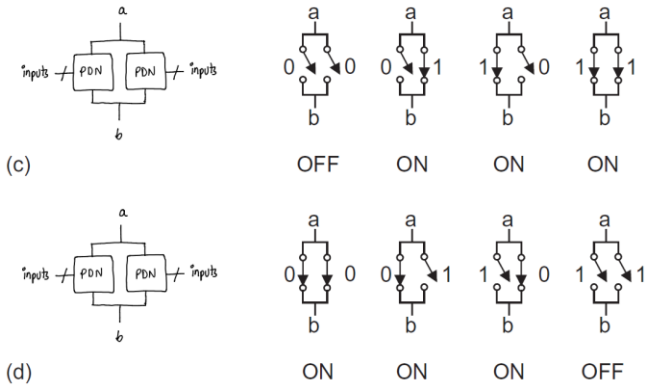
Series and Parallel Networks: *General* case 1

- Pull-up network: multiple pMOS transistors
 - *series* combination
 - Low voltage at all gate inputs: **ON**
 - *parallel* combination
 - Low voltage at minimum one gate input: **ON**
- Pull-down network: multiple nMOS transistors
 - *series* combination
 - High voltage at all gate inputs: **ON**
 - *parallel* combination
 - High voltage at minimum one gate input: **ON**
- *Series* combination of such networks
 - *Every network* must be **ON** to make the **whole network conducting**



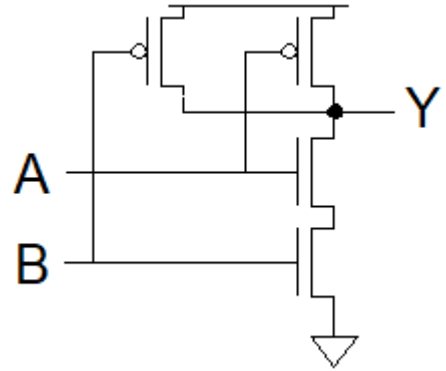
Series and Parallel Networks: **General** case 2

- Pull-up network: multiple pMOS transistors
 - *series* combination
 - Low voltage at all gate inputs: **ON**
 - *parallel* combination
 - Low voltage at minimum one gate input: **ON**
- Pull-down network: multiple nMOS transistors
 - *series* combination
 - High voltage at all gate inputs: **ON**
 - *parallel* combination
 - High voltage at minimum one gate input: **ON**
- *Parallel* combination of such networks
 - *At least one network* must be **ON** to make the **whole network conducting**



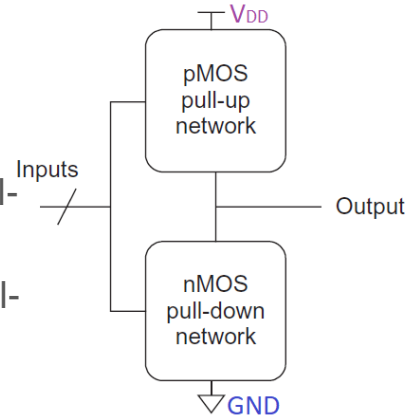
Conduction Complement

- Rule of Conduction Complements
 - Pull-up network is complement of pull-down (and vice versa)
 - Parallel -> series, series -> parallel
- Example: CMOS NAND-2 gate
 - Pull-down: 2 nMOS in **series** connection with inputs A & B
 - Pull-up: 2 pMOS in **parallel** (compliment of series) connection with the same inputs A & B
- Conclusion: for any CMOS logic gate, we just need to build only one network (usually the pull-down, because its easier) and the conduction complement rule will take care of the other one!



Building An Inverting CMOS Logic Gate

1. Build the **pull-down network** using nMOS transistors
 - a. **series** nMOS transistor networks for inputs/signals that are “**AND**”ed
 - b. **parallel** nMOS transistors for inputs/signals that are “**OR**”ed
2. Build the **pull-up network** using pMOS transistors
 - a. Using the conduction complement rule
 - b. Parallel networks in pull-down network will be redrawn in series for pull-up network
 - c. Series networks in pull-down network will be redrawn in parallel for pull-up network
3. Connect the pull-up network (PUN) and the pull-down network (PDN) in series between V_{DD} and GND

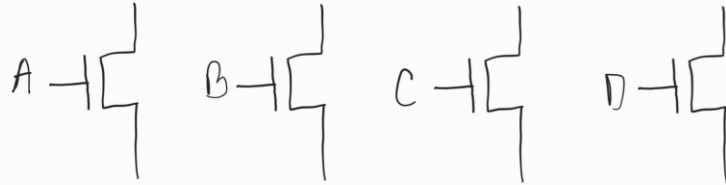


An Example: Building the Pull-down Network

- **series nMOS** transistor networks for inputs/signals that are “**AND**”ed together
- **parallel nMOS** transistors for inputs/signals that are “**OR**”ed together
 - Example: consider the inverting function, $Y = [(A+BC)D]'$
 - Step 1: Individual PD networks for A, B, C, D inputs (simple case)
 - 4 single nMOS transistors
 - Step 2: PD network for **BC** signal
 - B & C ANDed in **BC**: B & C PD networks will be in **series**
 - Step 3: PD network for **(A+BC)** signal
 - A and BC ORed (**A+BC**): A & BC PD networks will be in **parallel**
 - Step 4: PD network for **(A+BC)D** signal
 - (A+BC) and D ANDed in **(A+BC)D**: (A+BC) & D PD networks will be in **series**
 - (The term “signal” is used to describe an intermediate value which are generated from inputs. Here A,B,C,D are inputs but A+BC or BC are referred to as signal)

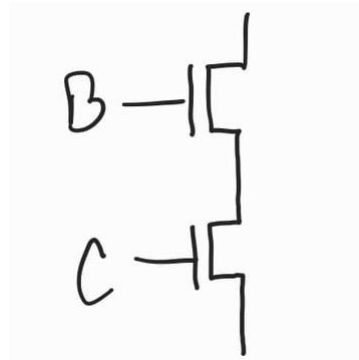
An Example: Building the Pull-down Network

- $Y = [(A+BC)D]'$
 - Step 1: Individual PD networks for A, B, C, D inputs (simple case)
 - 4 single nMOS transistors



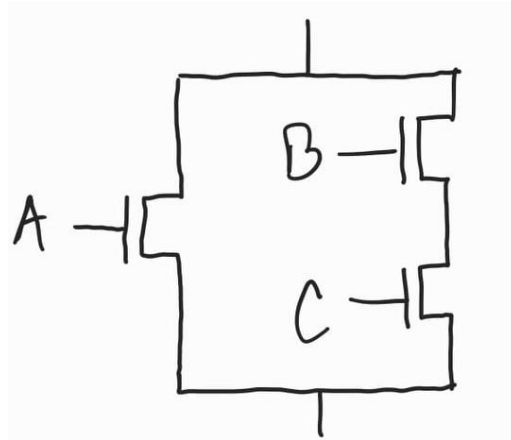
An Example: Building the Pull-down Network

- $Y = [(A+BC)D]'$
 - Step 2: PD network for **BC** signal
 - B & C ANDed in **BC**: B & C PD networks will be in **series**



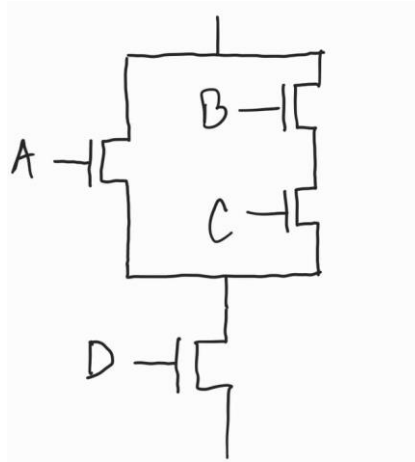
An Example: Building the Pull-down Network

- $Y = [(A+BC)D]'$
 - Step 3: PD network for **(A+BC)** signal
 - A and BC ORed (**A+BC**): A & BC PD networks will be in **parallel**



An Example: Building the Pull-down Network

- $Y = [(A+BC)D]'$
 - Step 4: PD network for $(A+BC)D$ signal
 - $(A+BC)$ and D ANDed in $(A+BC)D$: $(A+BC)$ & D PD networks will be in **series**

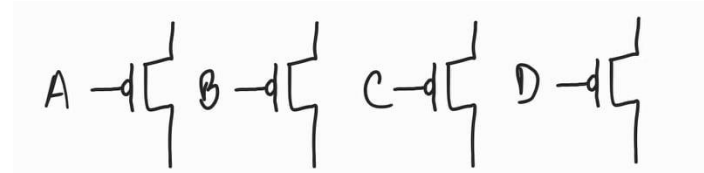
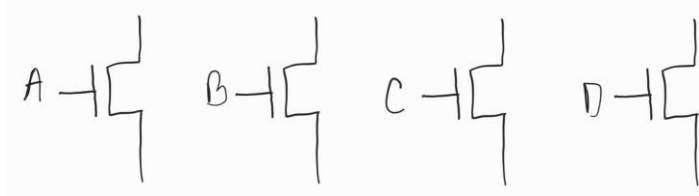


Building the Pull-up Network

- Just follow the CONDUCTION COMPLEMENT rule!!
- Use pMOS transistors instead of nMOS
- Parallel networks in pull-down network will be redrawn in series for pull-up network
- Series networks in pull-down network will be redrawn in parallel for pull-up network

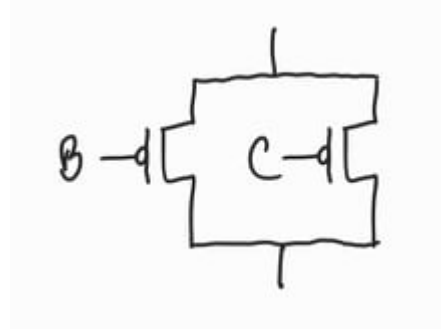
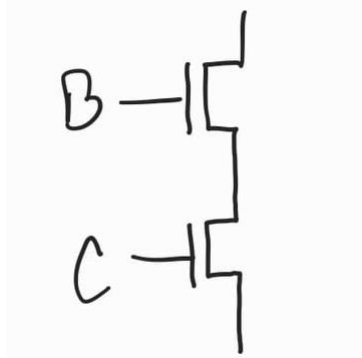
An Example: Building the Pull-up Network

- $Y = [(A+BC)D]'$
 - Step 1: Individual PU networks for A, B, C, D inputs (simple case)
 - 4 single pMOS transistors



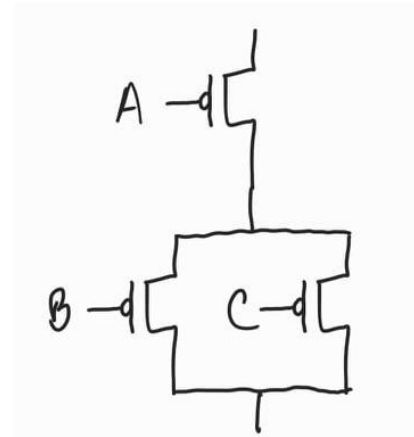
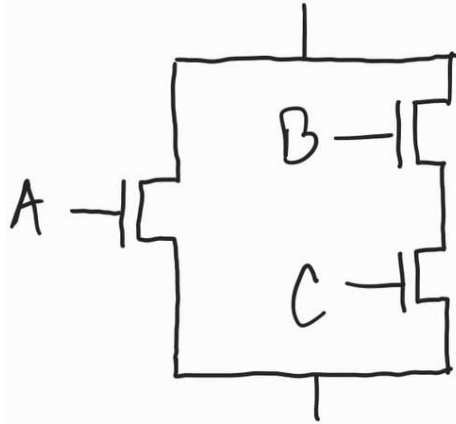
An Example: Building the Pull-up Network

- $Y = [(A+BC)D]'$
 - Step 2: PU network for **BC** signal
 - B & C PD networks was in series
 - B & C PU networks will be in **parallel**



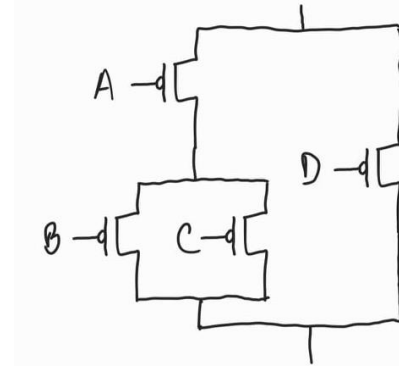
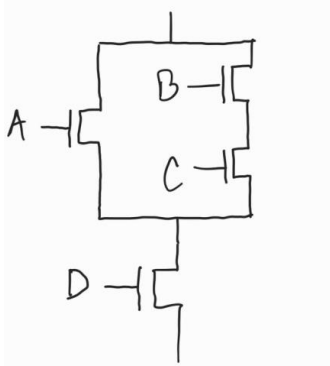
An Example: Building the Pull-up Network

- $Y = [(A+BC)D]'$
 - Step 3: PU network for **(A+BC)** signal
 - A & BC PD networks was in parallel
 - A & BC PU networks will be in **series**



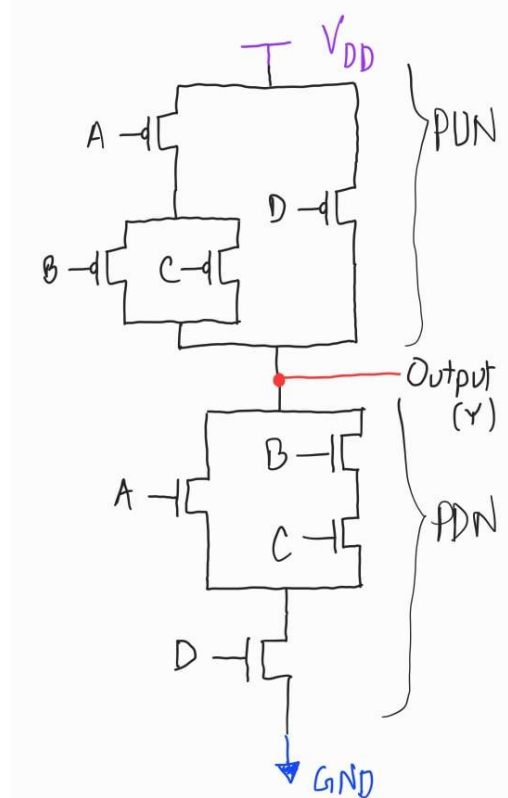
An Example: Building the Pull-up Network

- $Y = [(A+BC)D]'$
 - Step 4: PU network for $(A+BC)D$ signal
 - $(A+BC)$ & D PD networks was in series
 - $(A+BC)$ & D PU networks will be in parallel



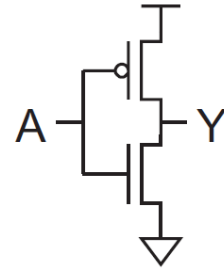
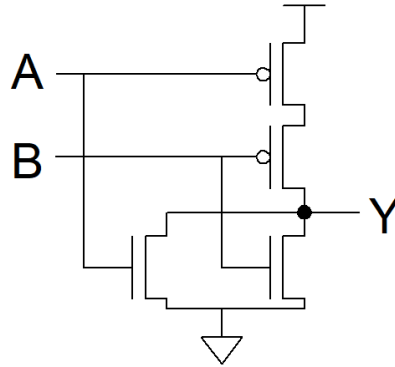
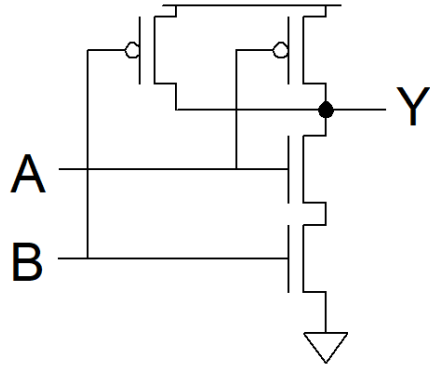
An Example: Putting it all together

- $Y = [(A+BC)D]'$
- Connect the pull-up network (PUN) and the pull-down network (PDN) in series between V_{DD} and GND
- The Output of the inverting function (Y) is taken from the common node of PUN and PDN
- This is the CMOS logic gate for the given inverting logic function Y



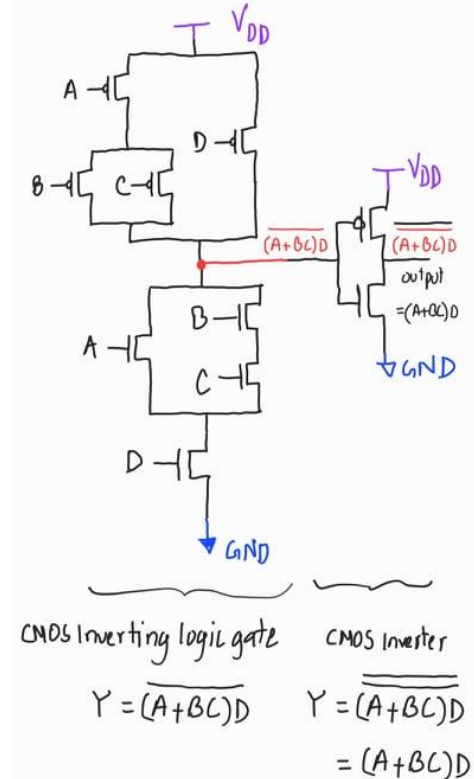
Example: CMOS NAND-2, NOR-2 and Inverter Gates

- NAND-2 [$Y=(AB)'$] NOR-2 [$Y=(A+B)'$] Inverter [$Y=A'$] logic functions can also be implemented in CMOS like this (try yourself first):



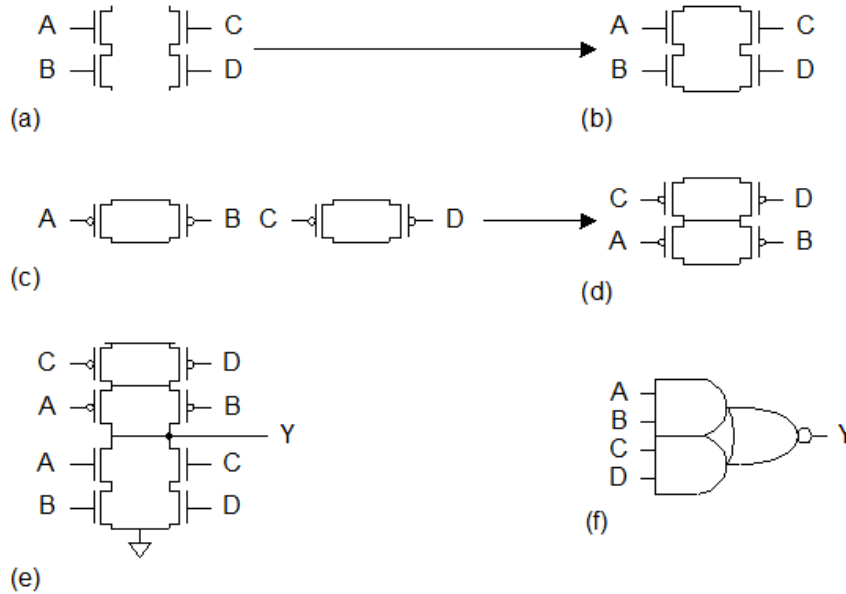
Implementing non-inverting logics in CMOS

- The techniques described so far can only be applied to build any **inverting** logic function
- There are multiple ways to implement **non-inverting** logic functions using CMOS
- The easiest way is to use an additional inverter at the output of the *inverting* function
- Example: $Y = (A+BC)D$ can be build like this
 - Build a CMOS logic gate for $Y = [(A+BC)D]'$
 - Connect the output of the previous gate to the input of an CMOS inverter
 - The output of the inverter will be $Y = (A+BC)D$



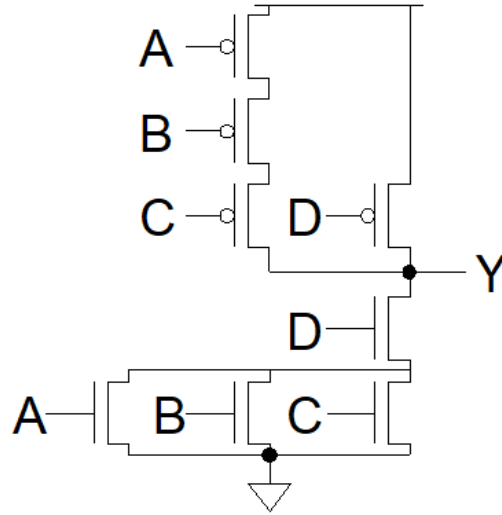
Example: CMOS AOI22 Gate

- $Y = (AB+CD)'$ also called the AND-OR-INVERT-2-2 or AOI22 GATE



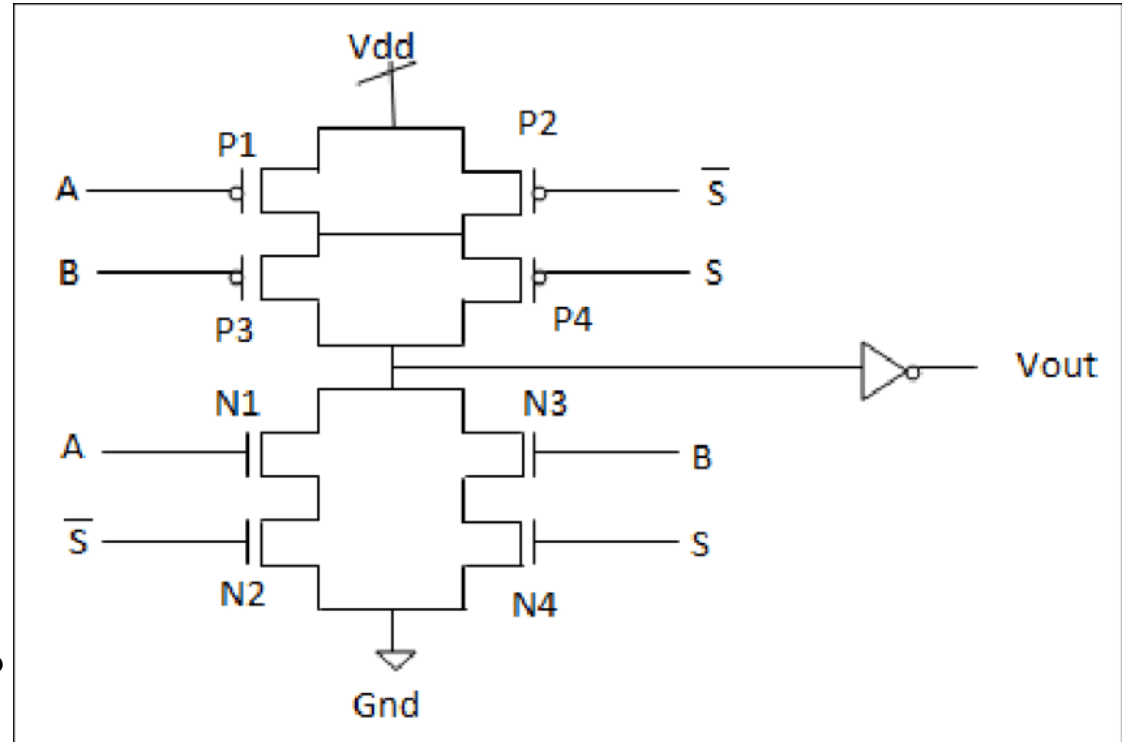
Final Example: CMOS OAI31 Gate

- $Y = [(A+B+C)D]'$ also called the OR-AND-INVERT-3-1 or OAI31 GATE



Final Example: CMOS Implementation of 2x1 MUX using

$$Y = S'A + SB$$



How many transistors required?

Thank you!