ELECTRICAL ENGINEERING
UNIVERSITY of WASHINGTON

Lecture 2: CMOS Logic

Based on material prepared by prof. Visvesh S. Sathe

ELECTRICAL & COMPUTER ENGINEERING

Acknowledgements

All class materials (lectures, assignments, etc.) based on material prepared by Prof. Visvesh S. Sathe, and reproduced with his permission

Visvesh S. Sathe
Associate Professor
Georgia Institute of Technology
https://psylab.ece.uw.edu

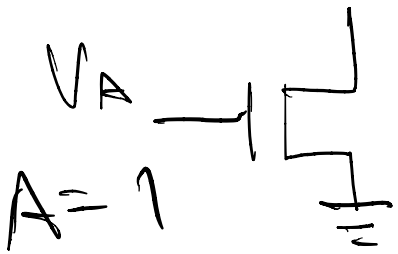
UW (2013-2022)
GaTech (2022-present)

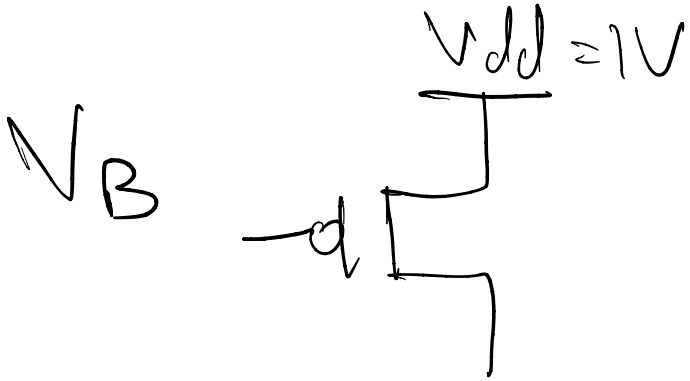
Course Announcements

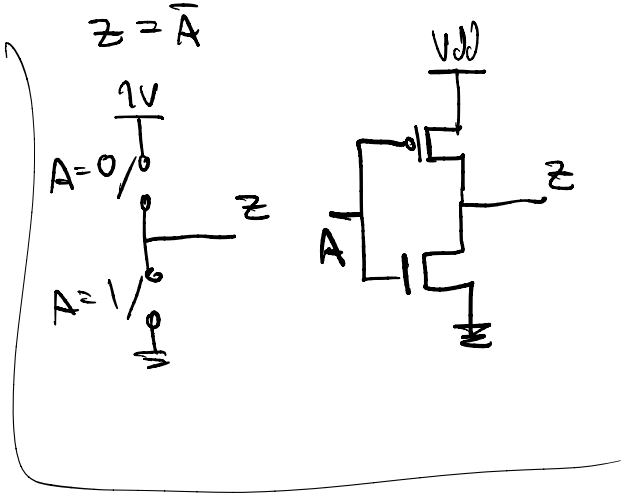
- By now, you should have
  - Make sure you can access the linux labs and have a directory for this class
  - Set up VNC connection

- Reservations for ECE 357:
  - Monday to Thursday: 2:00 pm to 5:00 pm
  - Friday: 11:30 am to 1:30 pm *(Kevin's office hours)*
  - Saturday and Sunday: 9:00 am to 1:00 pm
  - ***You're welcome to work there during any other time as well***, but there is no priority reservation.

Transistors as switches: CMOS inverter







$V_A \geq V_{thn}$

$V_A = Vdd = 1V$ on

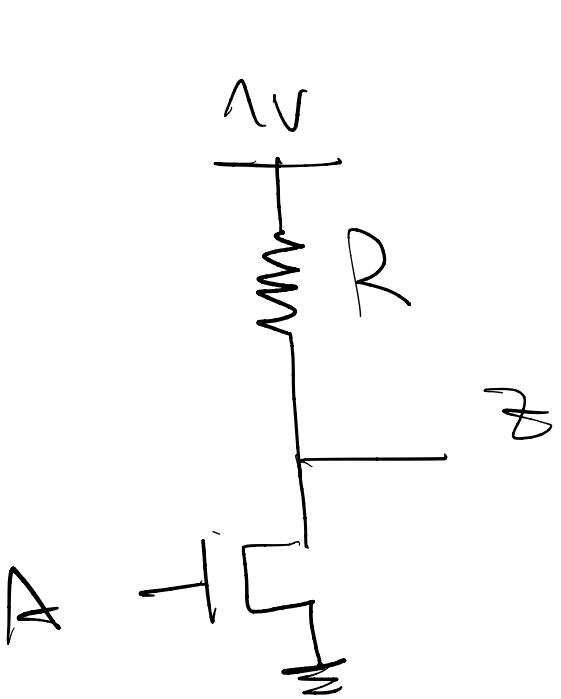
$V_A = 0$ off

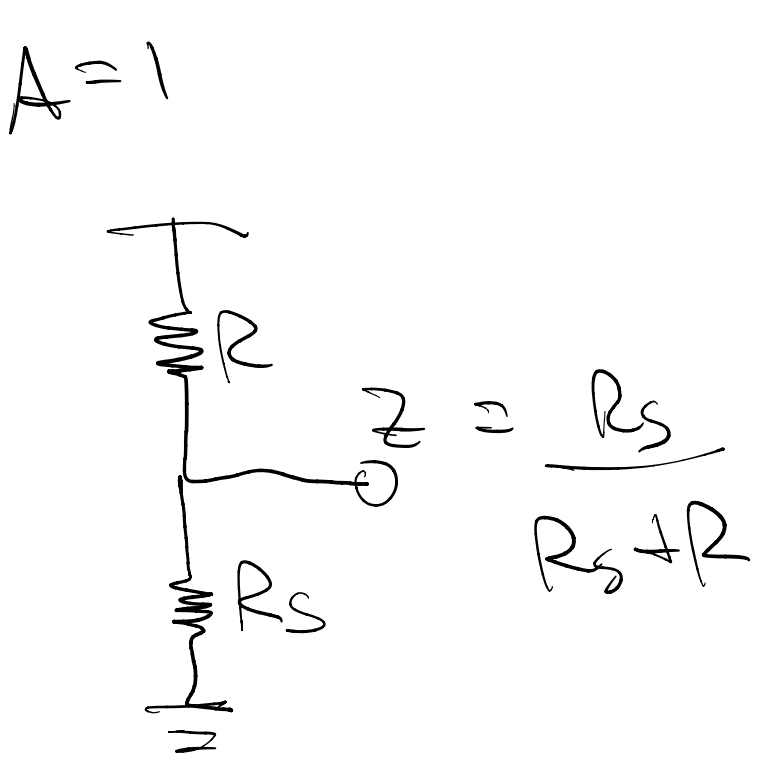
$V_B \leq Vdd - V_{thp}$ on

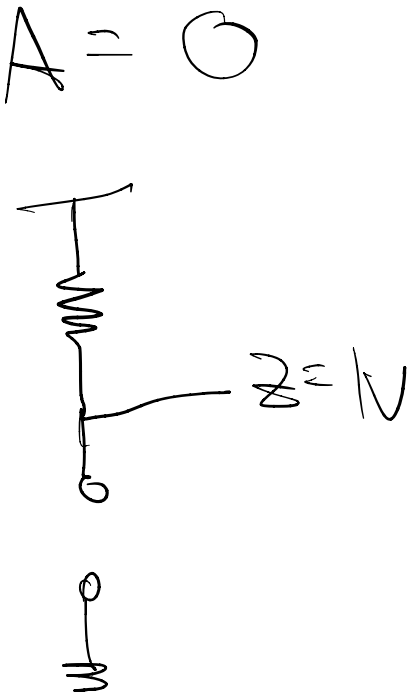
$V_B = 1V$ off

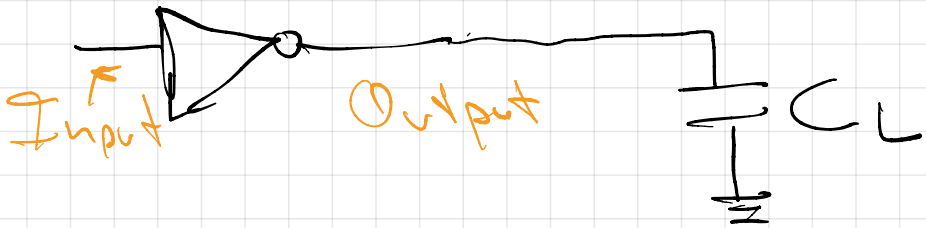
$V_B = 0V$ on

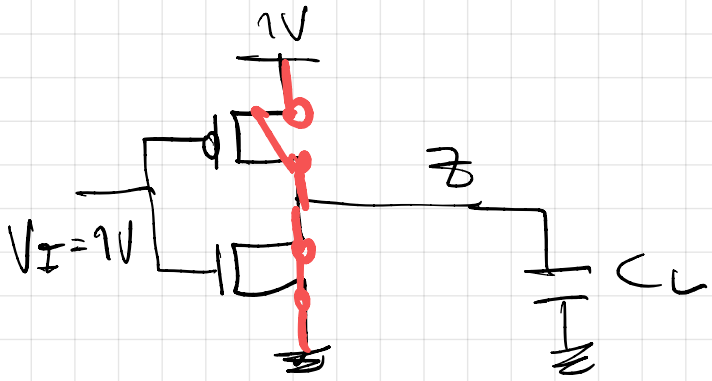
Quick aside: NMOS logic



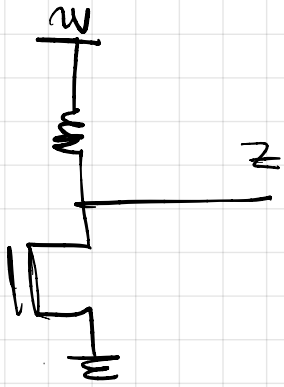






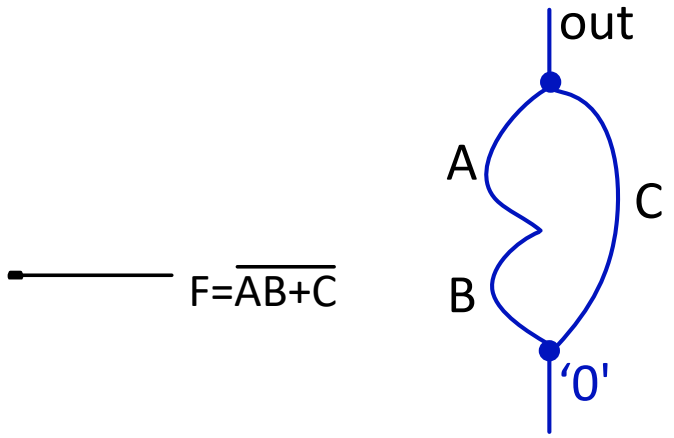


$V_Z(t=0) = 0.5V$



CMOS logic

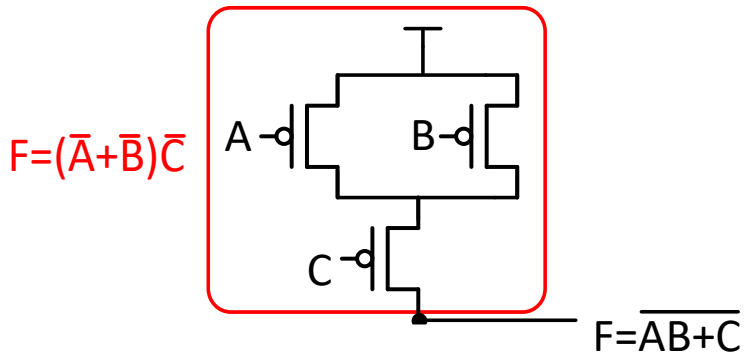
$F=(\overline{A}+\overline{B})\overline{C}$

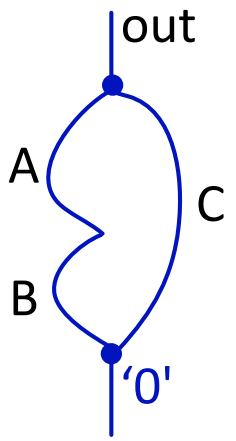


$F=\overline{AB+C}$

- Implement logic by connecting output to 1 (V_{dd}) or 0 (V_{ss})
  - Connect output Z to V_{dd} if function evaluates to '1'
  - Relay network connecting V_{dd} to Z referred to as the pull-up network (PUN)
  - Connect Z to V_{ss} if function evaluates to '0'
  - Relay network connecting V_{ss} to Z referred to as the pull-down network (PDN)
  - ➔Z will not be connected to *both* V_{dd} and V_{ss} . PUN and PDN ***complementary***
- Viewed as graphs, the networks are duals of each other

CMOS logic





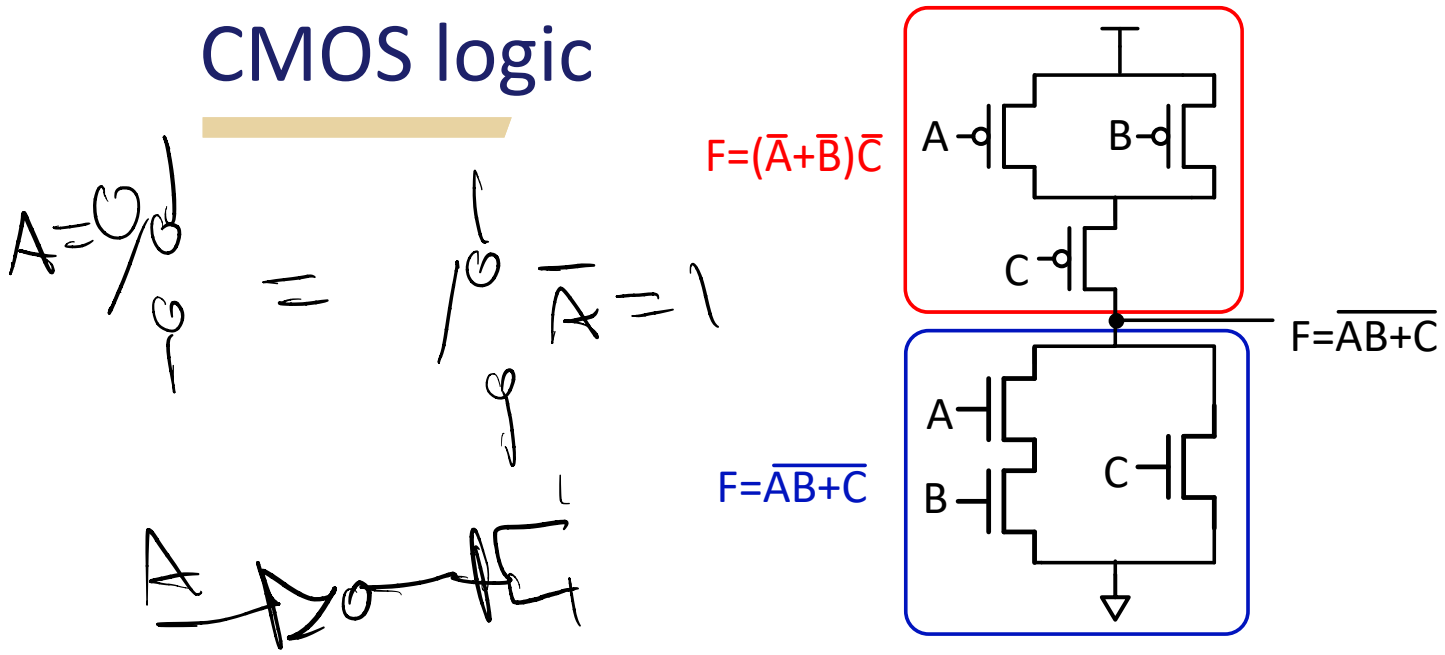
$\overline{AB+C}$

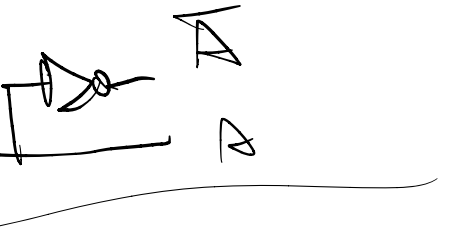
$(\bar{A}+\bar{B})\bar{C}$

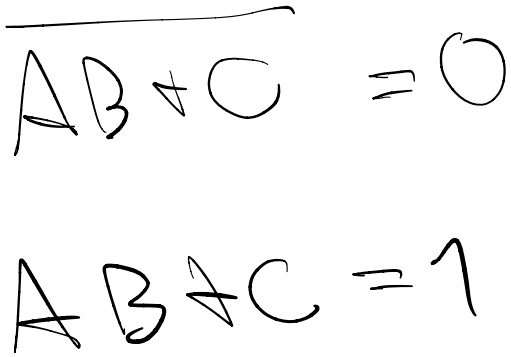
$F=\overline{AB+C}$

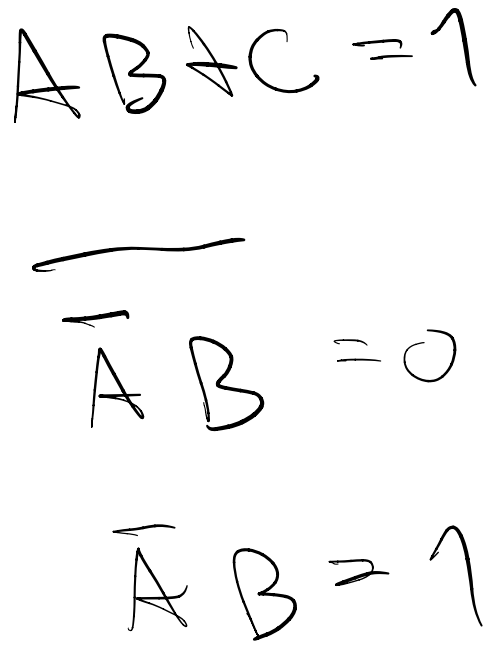
- Implement logic by connecting output to 1 (V_{dd}) or 0 (V_{ss})
  - Connect output Z to V_{dd} if function evaluates to '1'
  - Relay network connecting V_{dd} to Z referred to as the pull-up network (PUN)
  - Connect Z to V_{ss} if function evaluates to '0'
  - Relay network connecting V_{ss} to Z referred to as the pull-down network (PDN)
  - ➔Z will not be connected to *both* V_{dd} and V_{ss} . PUN and PDN ***complementary***
- Viewed as graphs, the networks are duals of each other

CMOS logic



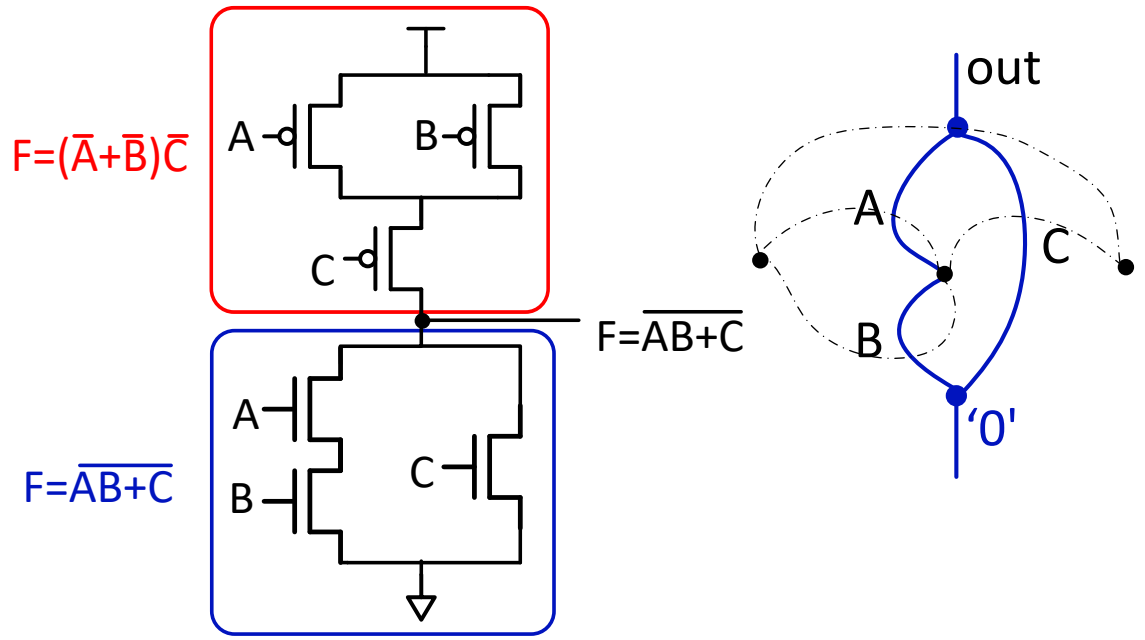






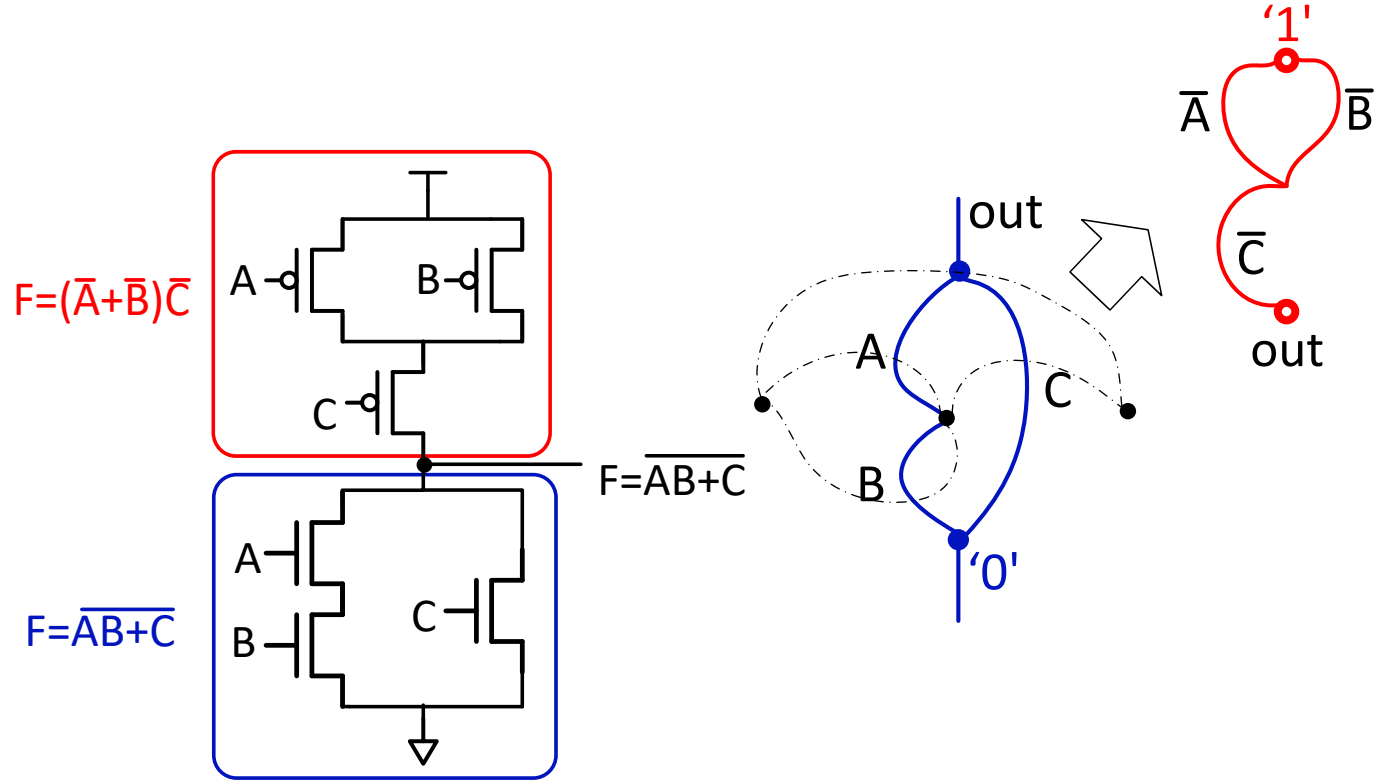
- Implement logic by connecting output to 1 (V_{dd}) or 0 (V_{ss})
  - Connect output Z to V_{dd} if function evaluates to '1'
  - Relay network connecting V_{dd} to Z referred to as the pull-up network (PUN)
  - Connect Z to V_{ss} if function evaluates to '0'
  - Relay network connecting V_{ss} to Z referred to as the pull-down network (PDN)
  - ➔Z will not be connected to *both* V_{dd} and V_{ss} . PUN and PDN ***complementary***
- Viewed as graphs, the networks are duals of each other

CMOS logic



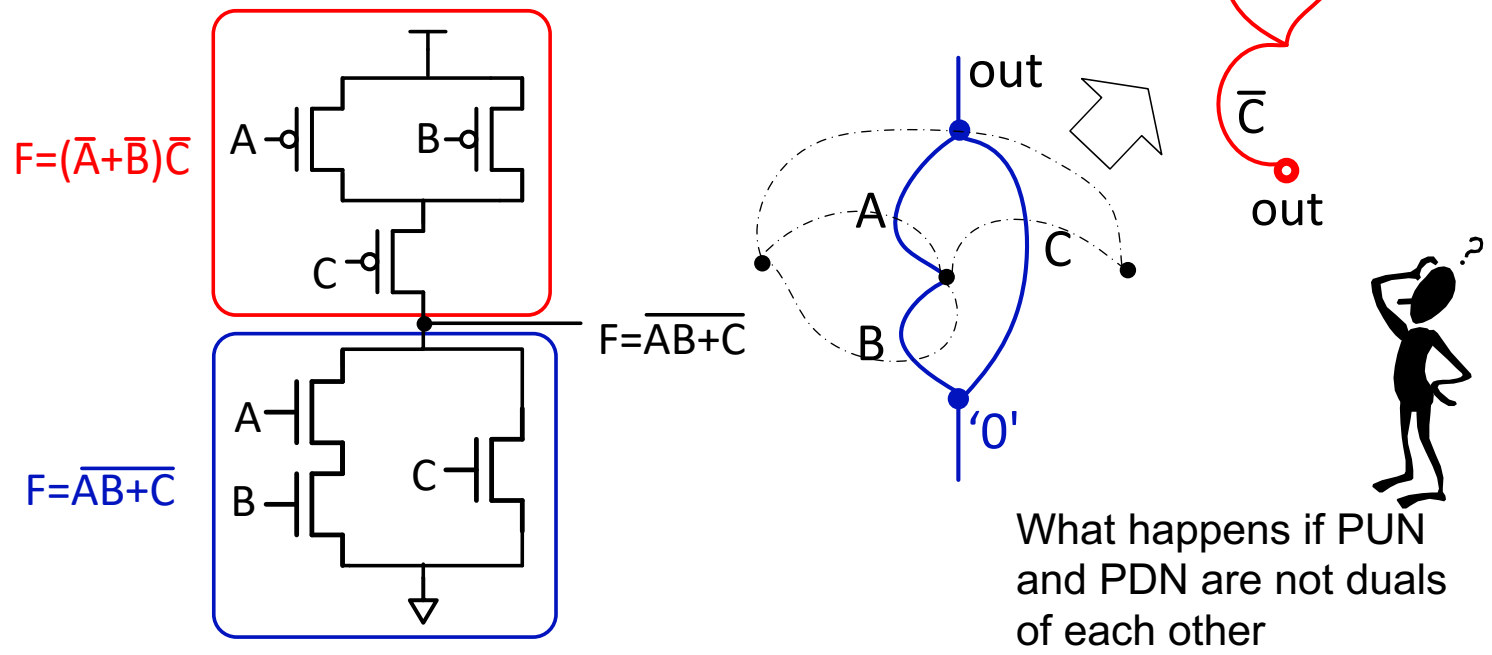
- Implement logic by connecting output to 1 (V_{dd}) or 0 (V_{ss})
  - Connect output Z to V_{dd} if function evaluates to '1'
  - Relay network connecting V_{dd} to Z referred to as the pull-up network (PUN)
  - Connect Z to V_{ss} if function evaluates to '0'
  - Relay network connecting V_{ss} to Z referred to as the pull-down network (PDN)
  - ➔Z will not be connected to *both* V_{dd} and V_{ss} . PUN and PDN ***complementary***
- Viewed as graphs, the networks are duals of each other

CMOS logic



- Implement logic by connecting output to 1 (V_{dd}) or 0 (V_{ss})
  - Connect output Z to V_{dd} if function evaluates to '1'
  - Relay network connecting V_{dd} to Z referred to as the pull-up network (PUN)
  - Connect Z to V_{ss} if function evaluates to '0'
  - Relay network connecting V_{ss} to Z referred to as the pull-down network (PDN)
  - ➔Z will not be connected to *both* V_{dd} and V_{ss} . PUN and PDN ***complementary***
- Viewed as graphs, the networks are duals of each other

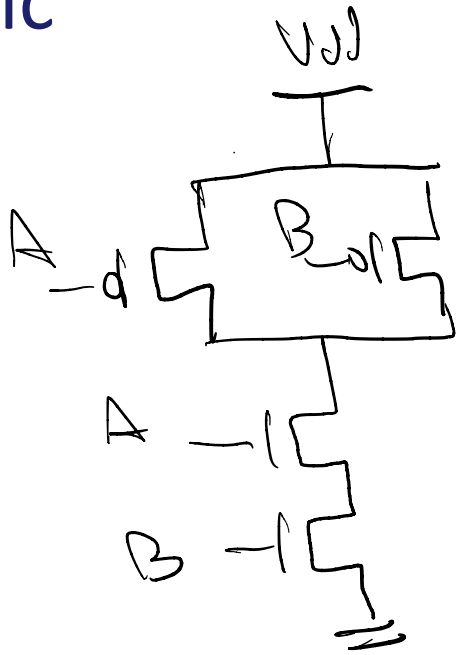
CMOS logic

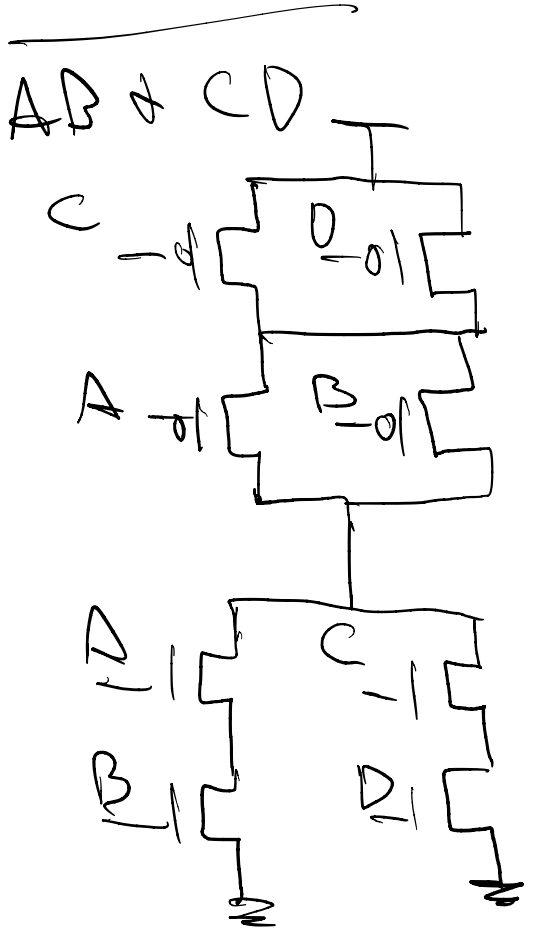


- Implement logic by connecting output to 1 (V_{dd}) or 0 (V_{ss})
  - Connect output Z to V_{dd} if function evaluates to '1'
  - Relay network connecting V_{dd} to Z referred to as the pull-up network (PUN)
  - Connect Z to V_{ss} if function evaluates to '0'
  - Relay network connecting V_{ss} to Z referred to as the pull-down network (PDN)
  - ➔Z will not be connected to *both* V_{dd} and V_{ss} . PUN and PDN ***complementary***
- Viewed as graphs, the networks are duals of each other

CMOS logic

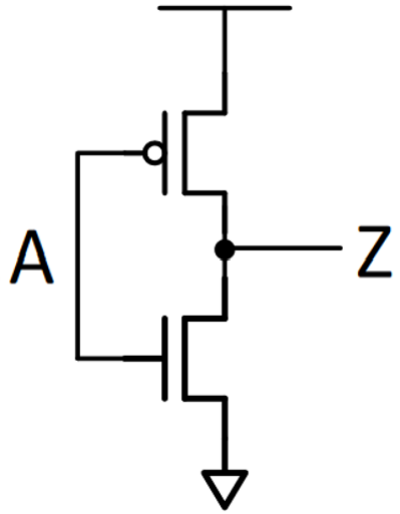
$Z = \overline{AB}$





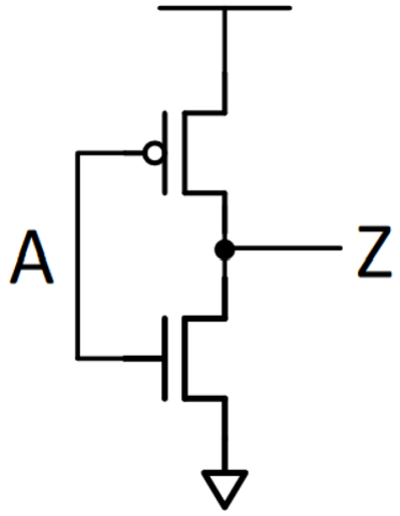
- Using A,B,C,D as inputs, implement using CMOS Logic
  - Z = ~A
  - Z=~(A&B)
  - Z=~(A&B + C&D)

CMOS logic



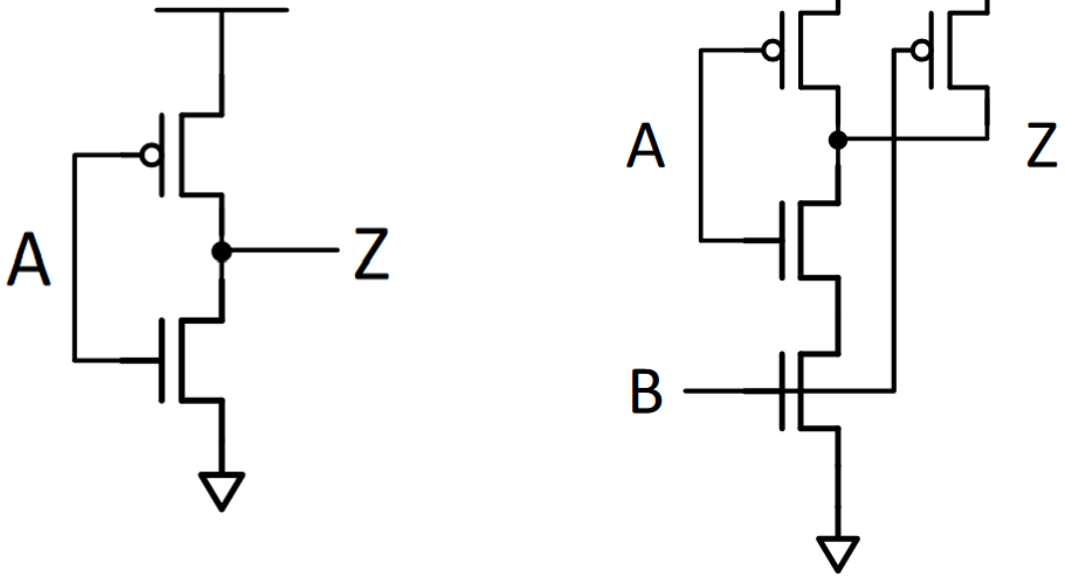
- Using A,B,C,D as inputs, implement using CMOS Logic
  - Z = ~A
  - Z=~(A&B)
  - Z=~(A&B + C&D)

CMOS logic



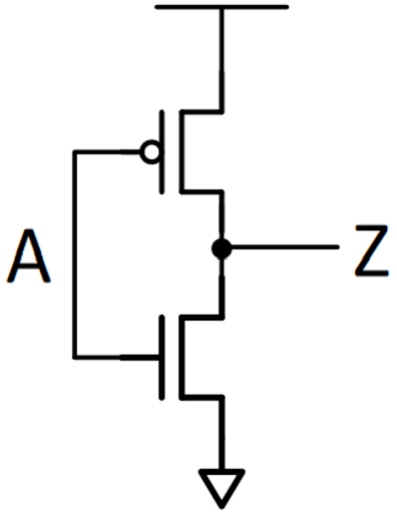
- Using A,B,C,D as inputs, implement using CMOS Logic
  - Z = ~A
  - Z=~(A&B) === ~A|~B
  - Z=~(A&B + C&D)

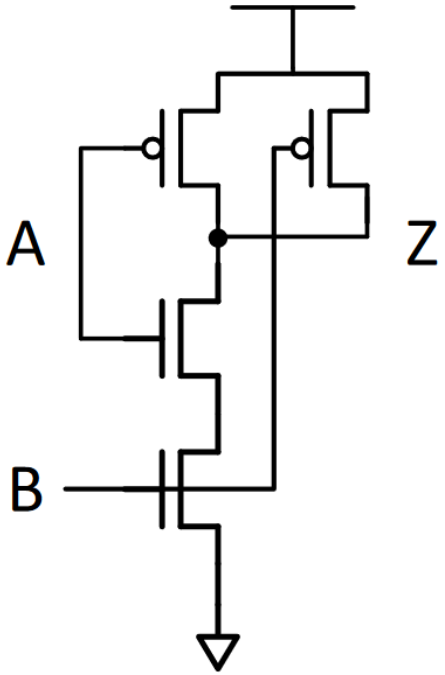
CMOS logic

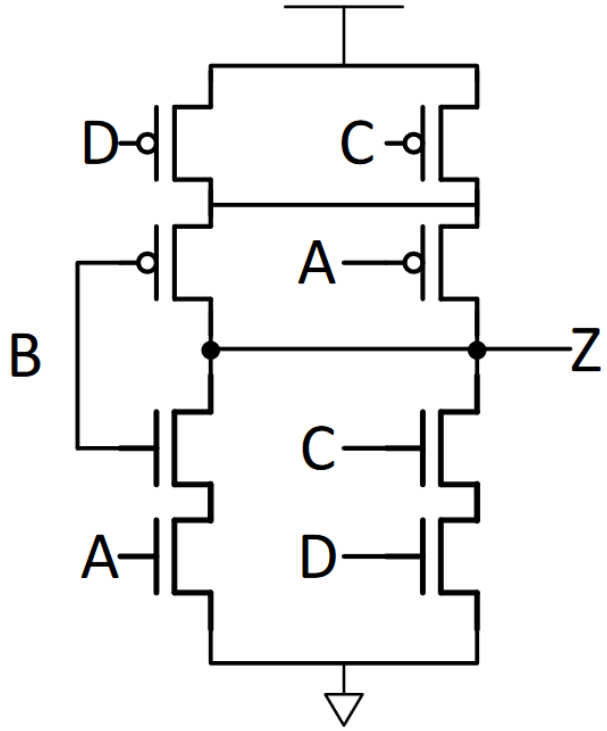


- Using A,B,C,D as inputs, implement using CMOS Logic
  - Z = ~A
  - Z=~(A&B) === ~A|~B
  - Z=~(A&B + C&D)

CMOS logic





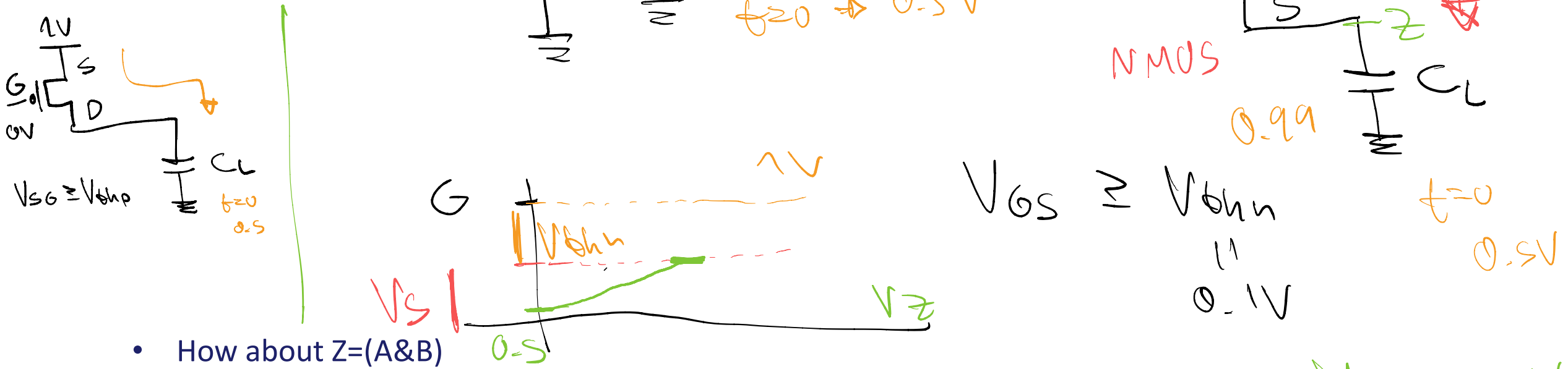


- Using A,B,C,D as inputs, implement using CMOS Logic
  - Z = ~A
  - Z=~(A&B) === ~A|~B
  - Z=~(A&B + C&D)

A short note on the Threshold Voltage

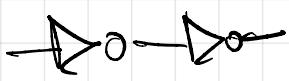
- Story thus far:
  - NMOS conducts If $V_{gate} > (V_{drain}$ or $V_{source})$
  - PMOS conducts if $V_{gate} < (V_{drain}$ or $V_{source})$
- In Reality : V_{th} , the threshold voltage plays a role
  - An “overhead” cost that must be paid to enable device to conduct
  - NMOS conducts If $V_{gate}-V_{th} > (V_{drain}$ or $V_{source})$
  - PMOS conducts if $V_{gate} + V_{th} < (V_{drain}$ or $V_{source})$

CMOS logic



- How about Z=(A&B)
  - Nmos “passes a 1” poorly, PMOS “passes a 0” poorly (More on this shortly)
    - NMOS gate must exceed source/drain by a threshold to conduct
    - PMOS gate must be lower than source/drain by a threshold to conduct
    - Recall discussion on current flow in relay networks
  - How do I get an AND gate then?

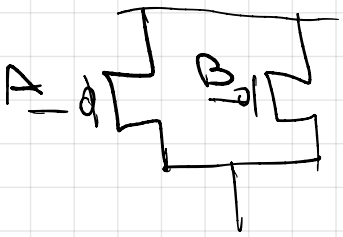
$Z = \overline{\overline{A}} = A$

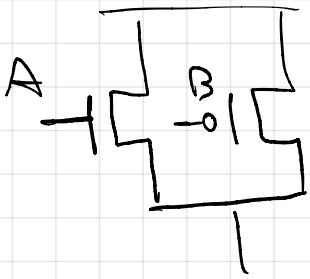


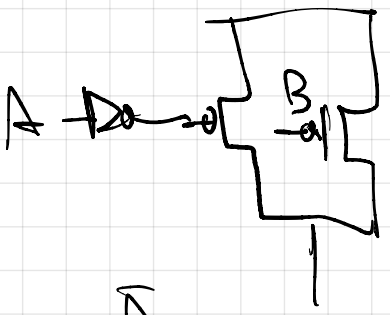
$Z = \overline{AB}$?

A=1 / = / $\bar{A}$=0

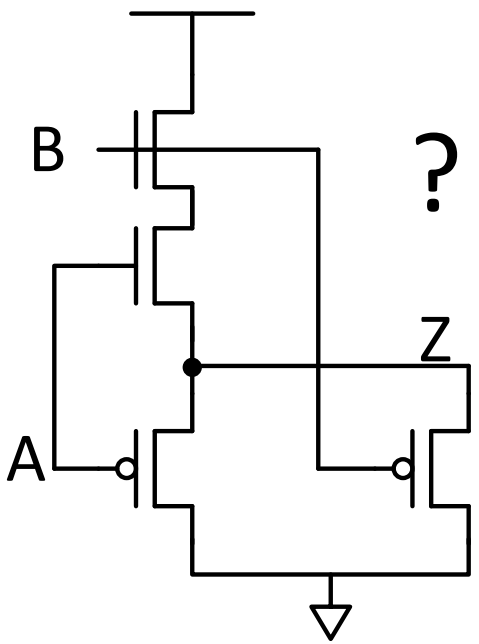
$Z = \overline{AB}$





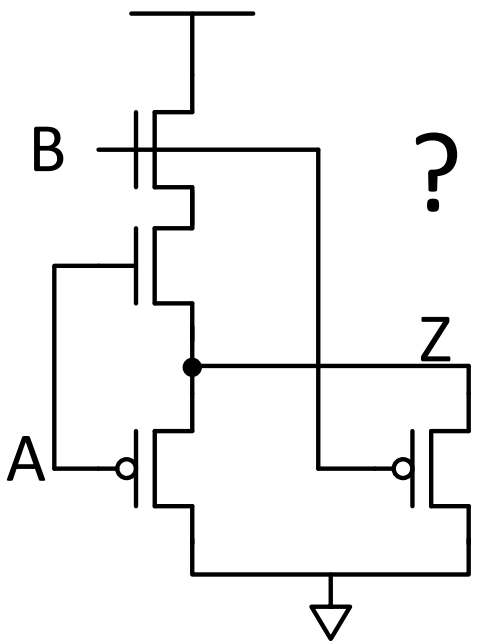


CMOS logic



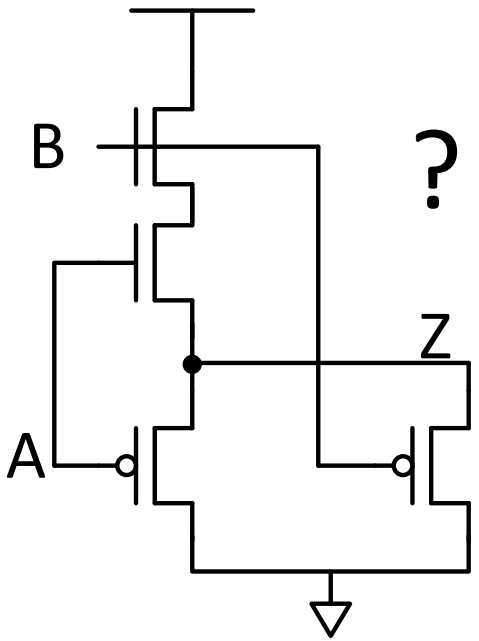
- How about Z=(A&B)
  - Nmos “passes a 1” poorly, PMOS “passes a 0” poorly (More on this shortly)
    - NMOS gate must exceed source/drain by a threshold to conduct
    - PMOS gate must be lower than source/drain by a threshold to conduct
    - Recall discussion on current flow in relay networks
  - How do I get an AND gate then?

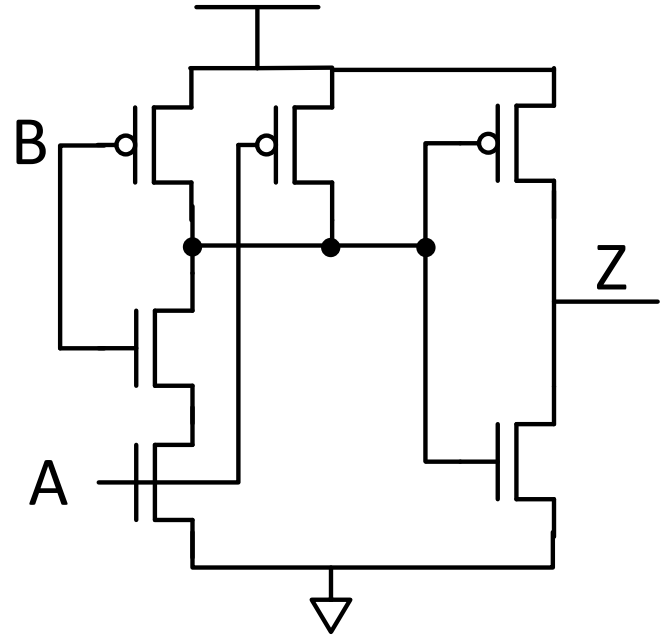
CMOS logic



- How about Z=(A&B)
  - Nmos “passes a 1” poorly, PMOS “passes a 0” poorly (More on this shortly)
    - NMOS gate must exceed source/drain by a threshold to conduct
    - PMOS gate must be lower than source/drain by a threshold to conduct
    - Recall discussion on current flow in relay networks
  - How do I get an AND gate then? Z= ~ (~(A&B)))

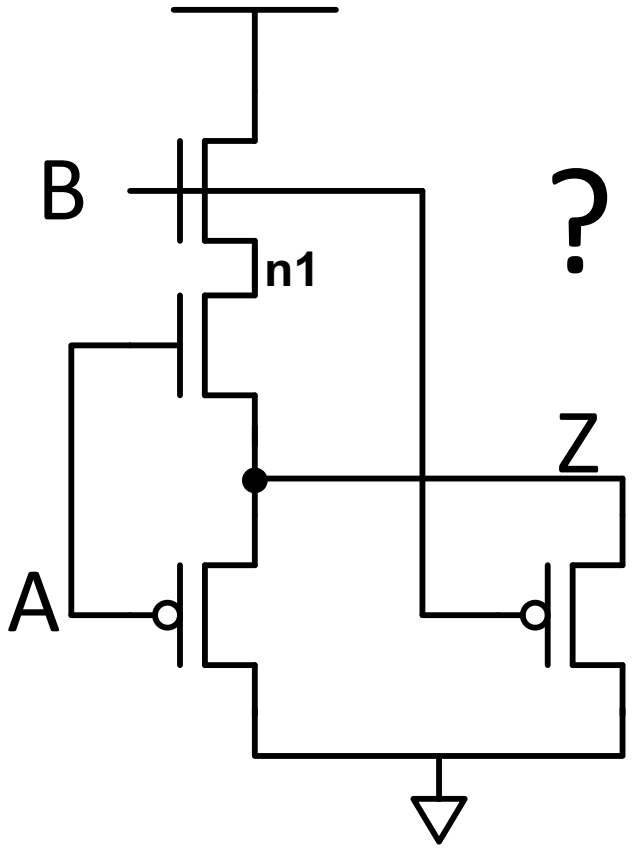
CMOS logic





- How about Z=(A&B)
  - Nmos “passes a 1” poorly, PMOS “passes a 0” poorly (More on this shortly)
    - NMOS gate must exceed source/drain by a threshold to conduct
    - PMOS gate must be lower than source/drain by a threshold to conduct
    - Recall discussion on current flow in relay networks
  - How do I get an AND gate then? Z= ~ (~(A&B)))

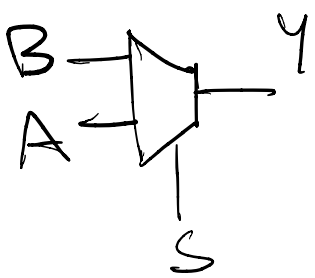
Quick Aside: Voltage swing range for Z

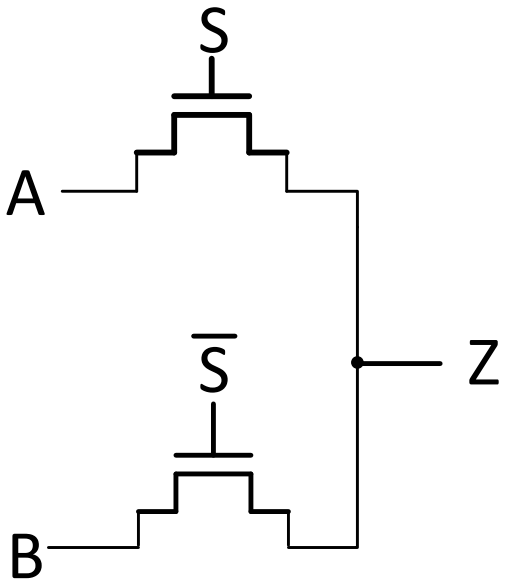


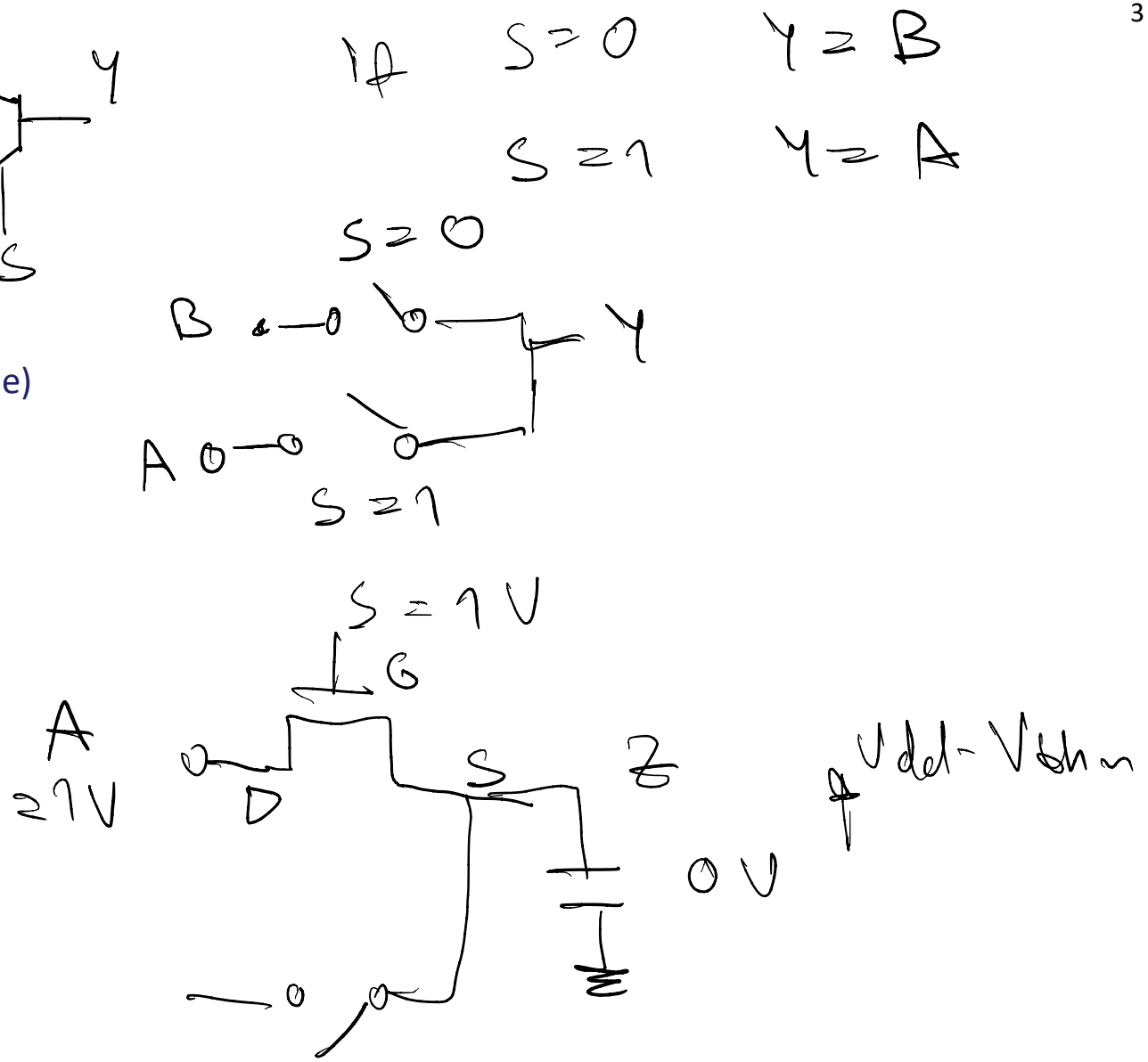
- What is the maximum voltage Z can reach (Ignoring leakage)
- What is the minimum voltage Z can reach (Ignoring leakage)

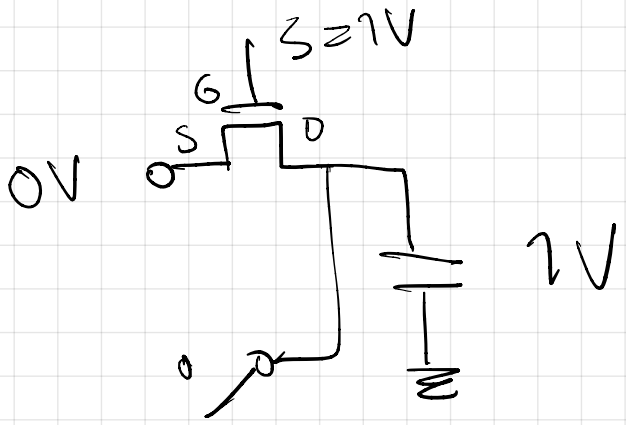
Mux

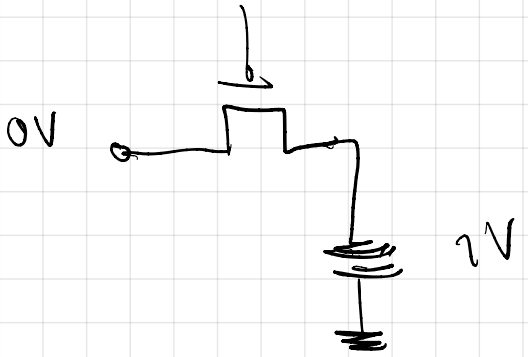
- Z = (S==1)? A:B
  - $Z = SA + \bar{S}B = \overline{\overline{SA + \bar{S}B}} = \overline{(\bar{S} + \bar{A})(S + \bar{B})}$
  - Use complimentary inputs (create ~S if not available)

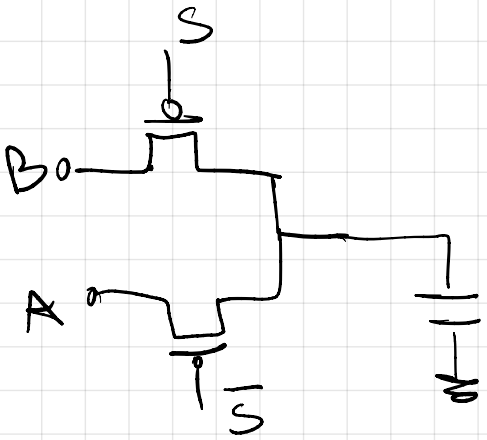






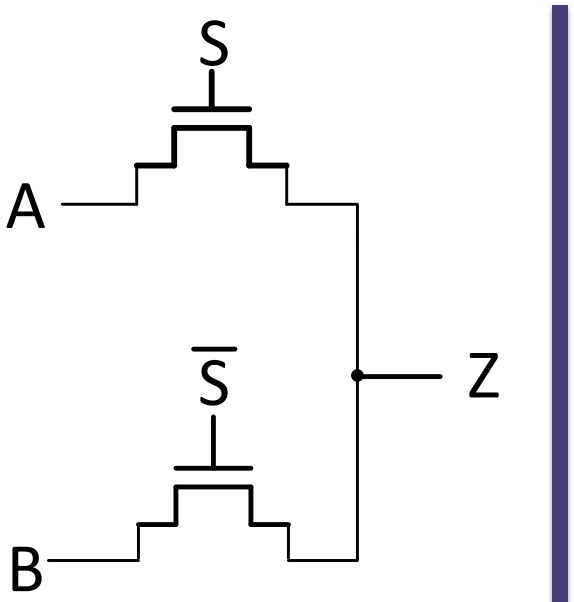

S≥1V
G
S
D
0V
1V


0V
1V


S
B
A
S̄

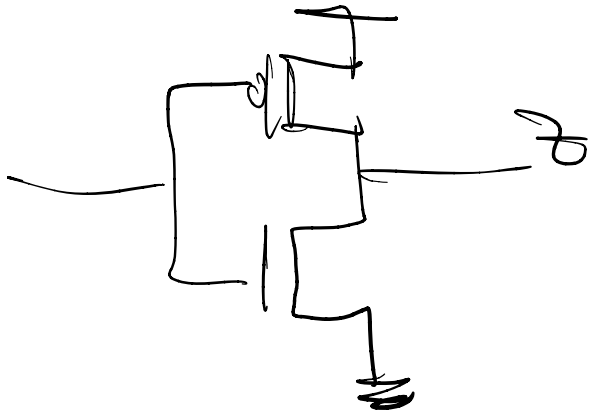
Mux

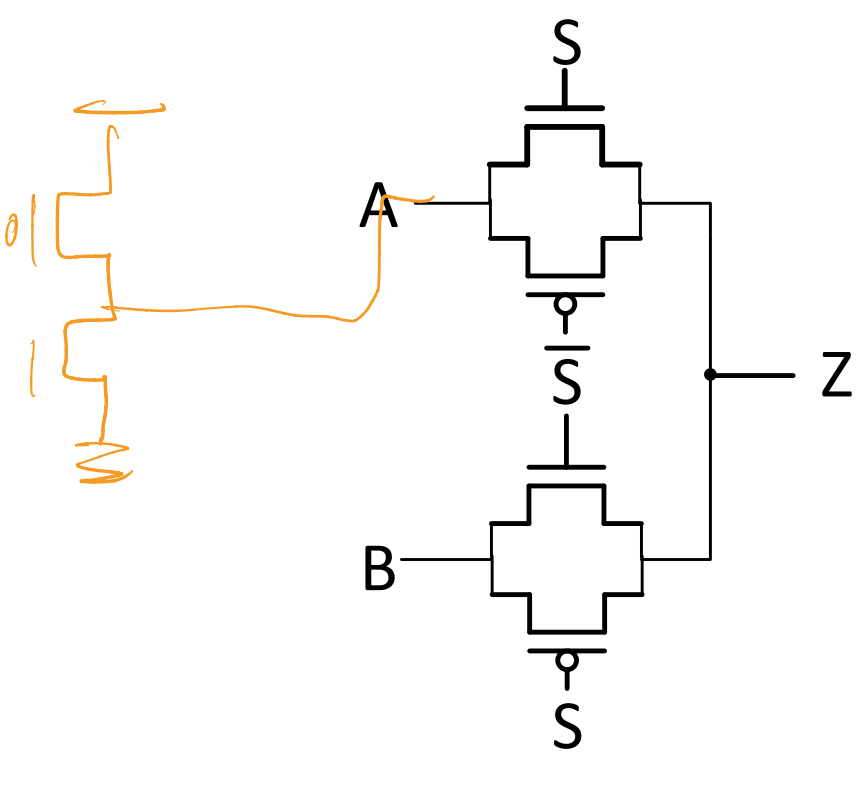
- Z = (S==1)? A:B
  - $Z = SA + \bar{S}B = \overline{\overline{SA + \bar{S}B}} = \overline{(\bar{S} + \bar{A})(S + \bar{B})}$
  - Use complimentary inputs (create ~S if not available)
  - Ensure that both 1 and 0 are passed effectively



Mux

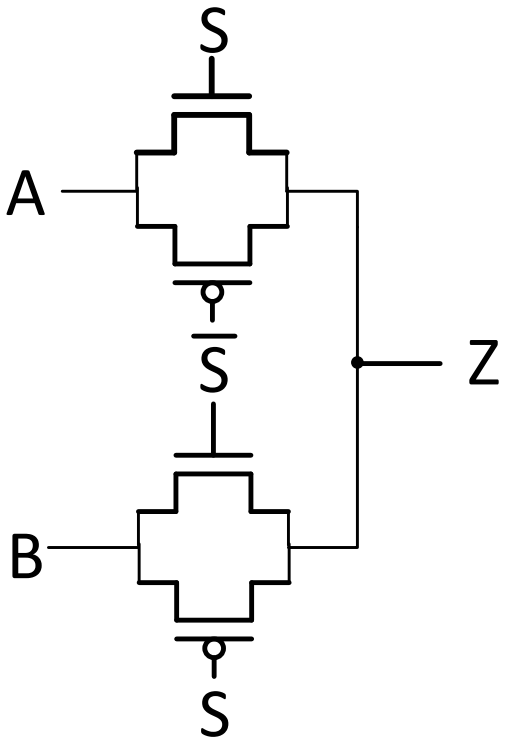
- Z = (S==1)? A:B
  - $Z = SA + \bar{S}B = \overline{\overline{SA + \bar{S}B}} = \overline{(\bar{S} + \bar{A})(S + \bar{B})}$
  - Use complimentary inputs (create ~S if not available)
  - Ensure that both 1 and 0 are passed effectively

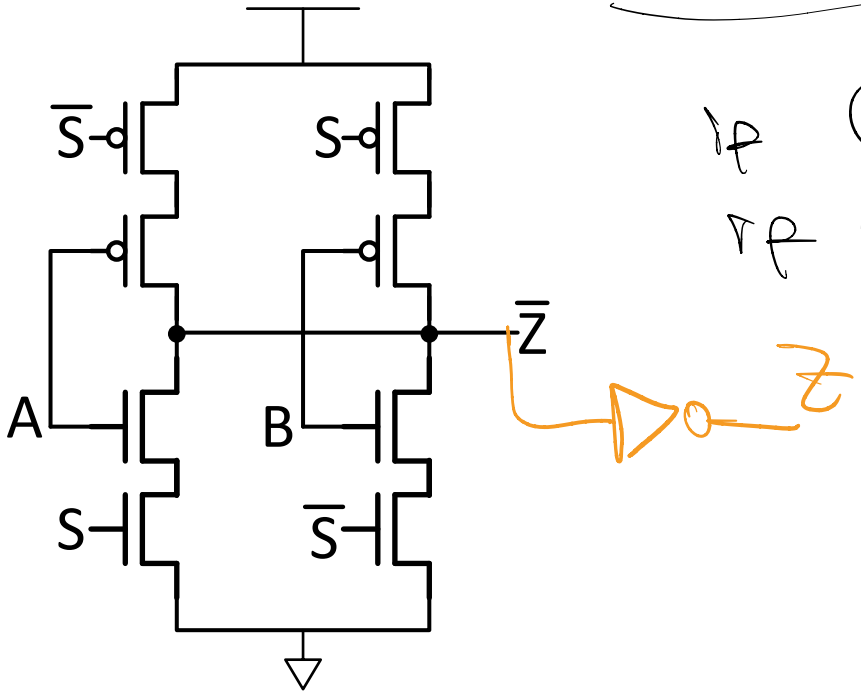




Mux

- Z = (S==1)? A:B
  - $Z = SA + \bar{S}B = \overline{\overline{SA + \bar{S}B}} = \overline{(\bar{S} + \bar{A})(S + \bar{B})}$
  - Use complimentary inputs (create ~S if not available)
  - Ensure that both 1 and 0 are passed effectively





Z=1 (Z̄=0) | Z=0 (Z̄=1)

B=1 S=0 | B=0 S=0

A=1 S=1 | A=0 S=1

ip (S=0) Z=B

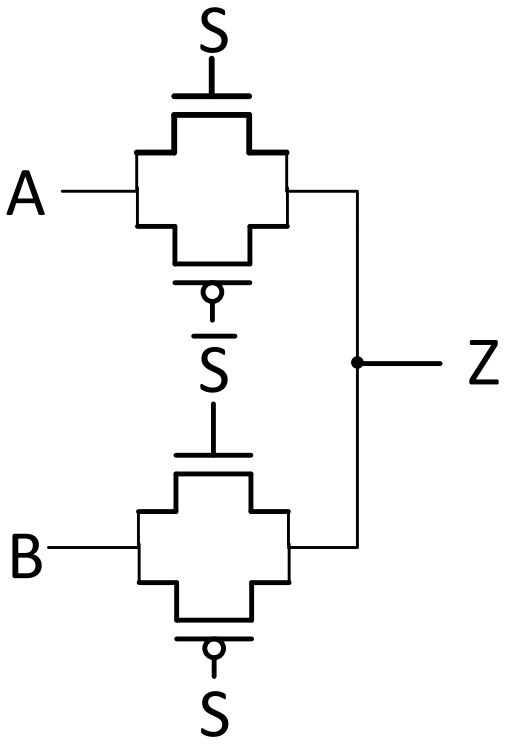
ip (S=1) Z=A

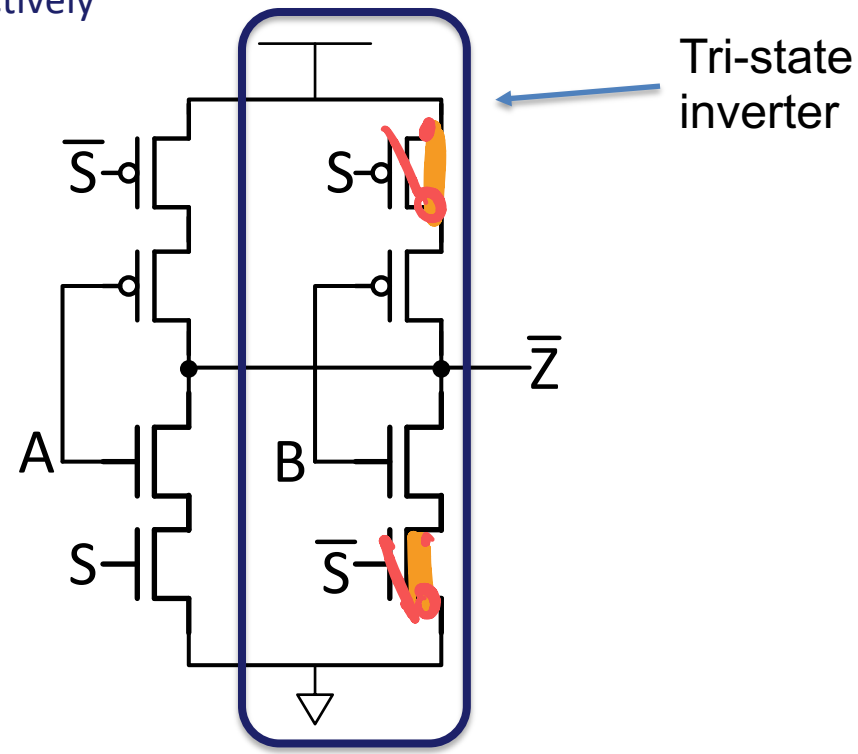
Mux

- Z = (S==1)? A:B
  - $Z = SA + \bar{S}B = \overline{\overline{SA + \bar{S}B}} = \overline{(\bar{S} + \bar{A})(S + \bar{B})}$
  - Use complimentary inputs (create ~S if not available)
  - Ensure that both 1 and 0 are passed effectively

S=0

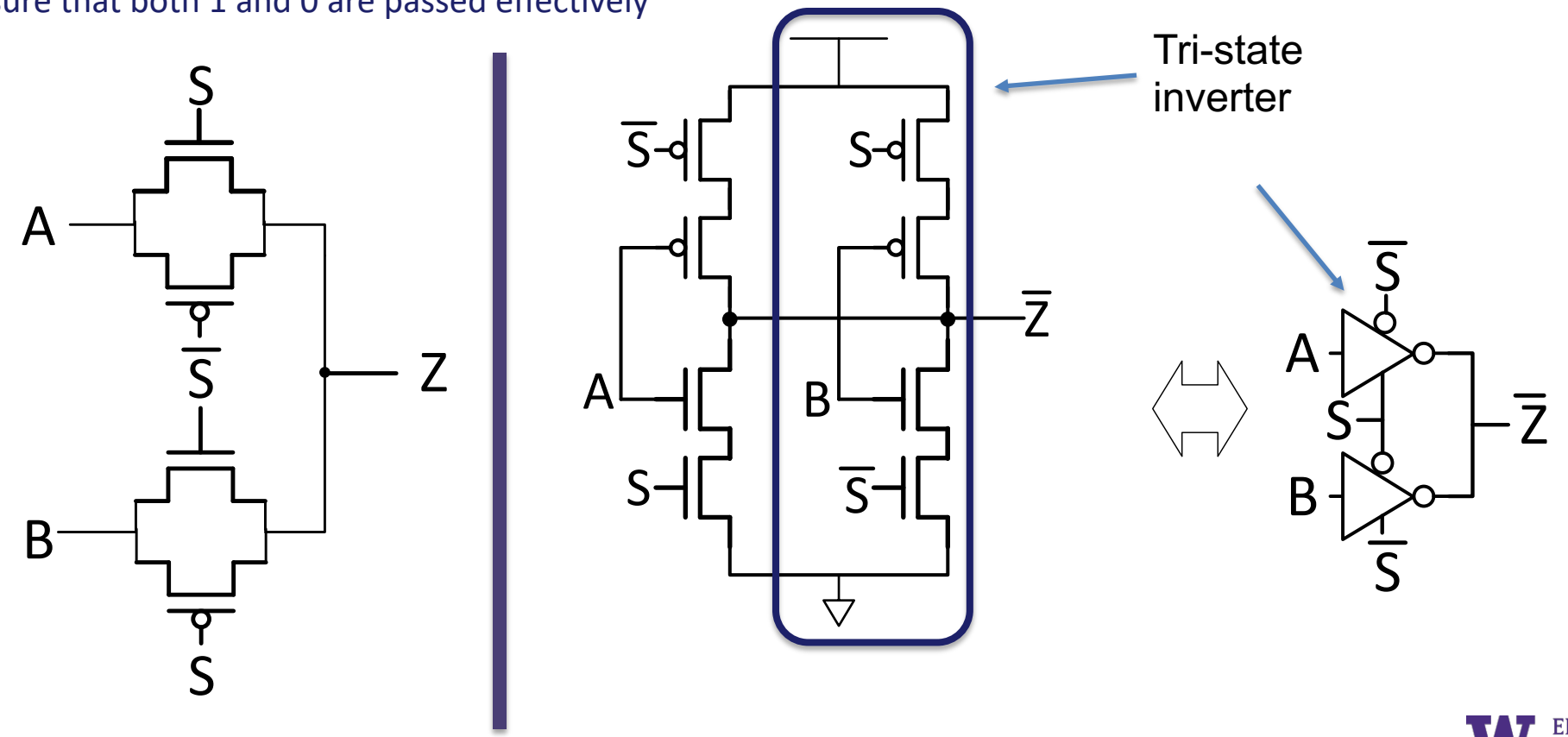
S=1

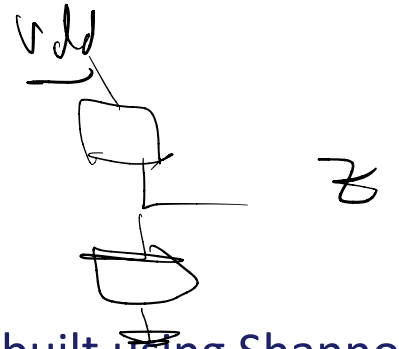




Mux

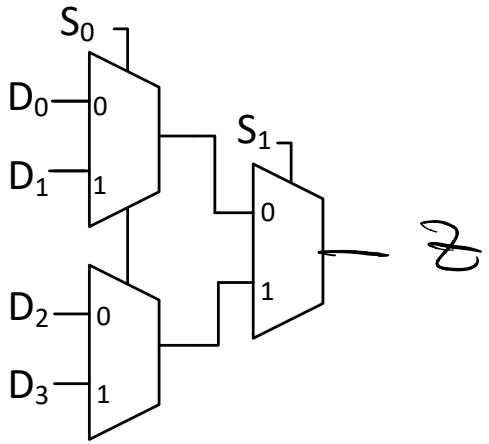
- Z = (S==1)? A:B
  - $Z = SA + \bar{S}B = \overline{\overline{SA + \bar{S}B}} = \overline{(\bar{S} + \bar{A})(S + \bar{B})}$
  - Use complimentary inputs (create ~S if not available)
  - Ensure that both 1 and 0 are passed effectively





N:1 Mux

- Any Boolean function can be built using Shannon's expansion theorem
- Two extreme approaches



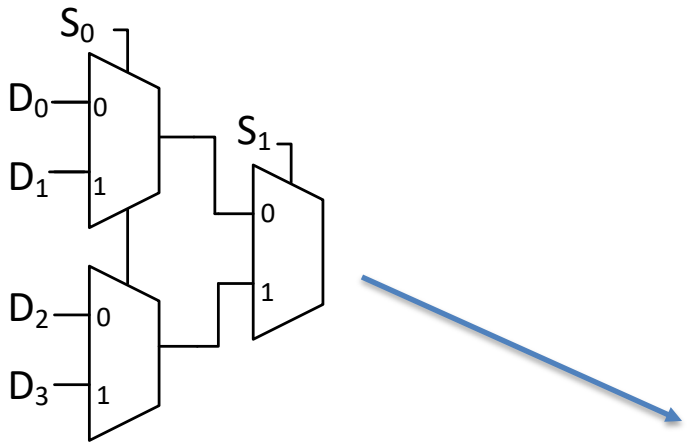
$S_0 S_1$

2 bit

$$Z = S_0 S_1 D_3 + \bar{S}_0 S_1 D_2 + S_0 \bar{S}_1 D_1 + \bar{S}_0 \bar{S}_1 D_0$$

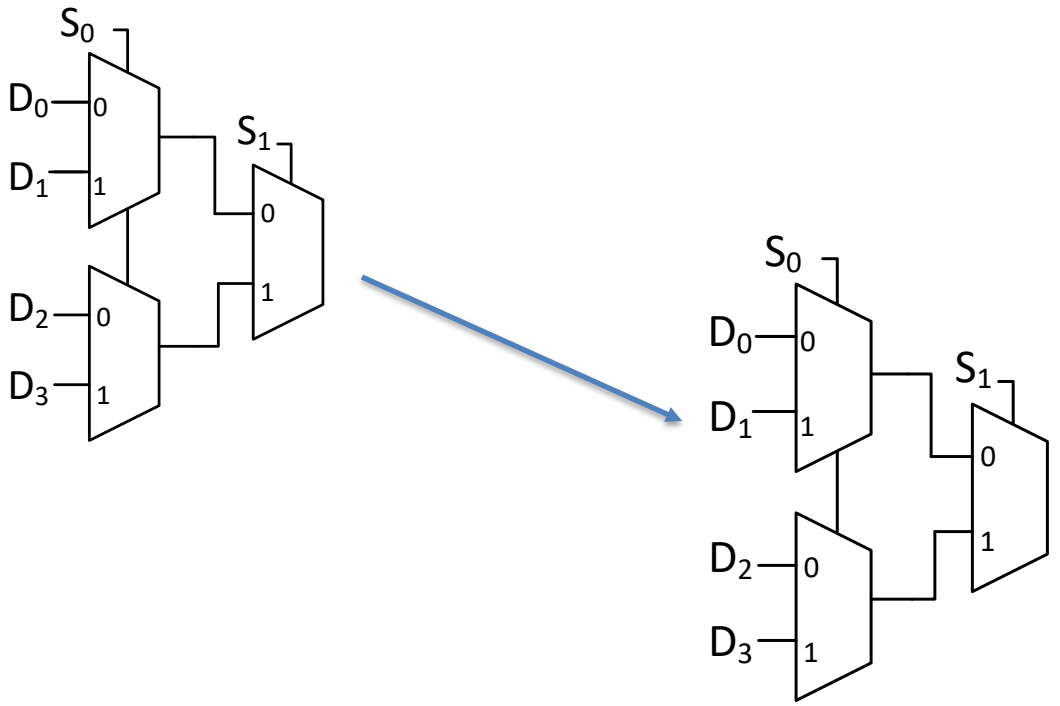
N:1 Mux

- Any Boolean function can be built using Shannon's expansion theorem
- Two extreme approaches



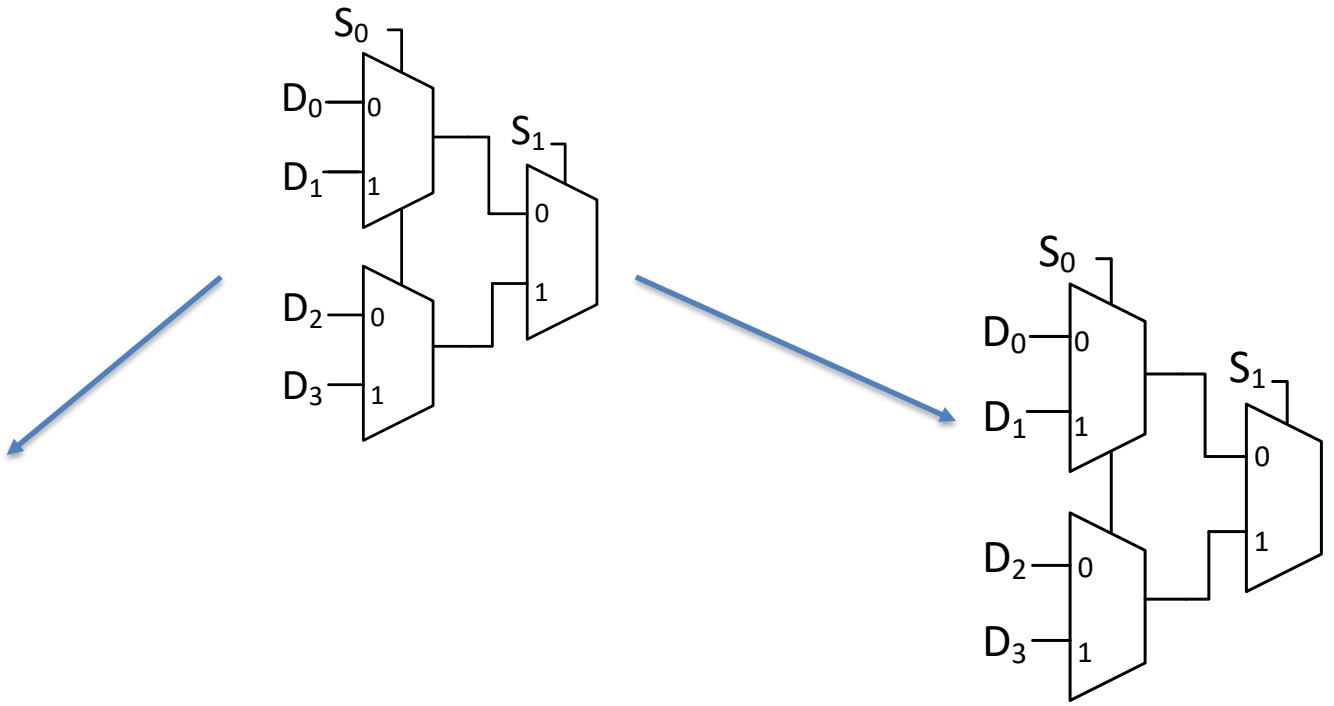
N:1 Mux

- Any Boolean function can be built using Shannon's expansion theorem
- Two extreme approaches

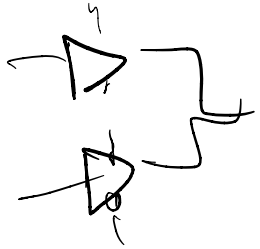


N:1 Mux

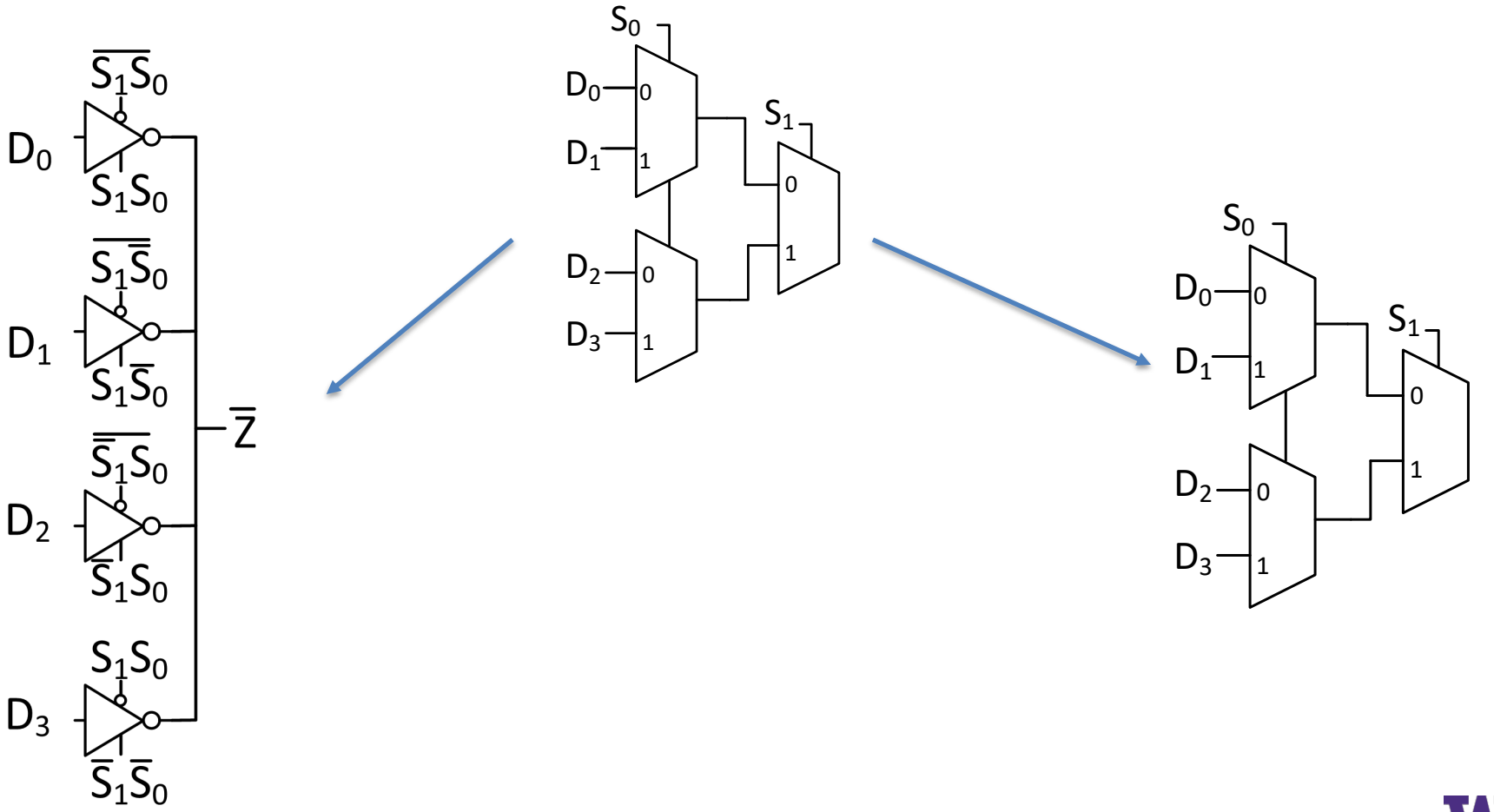
- Any Boolean function can be built using Shannon's expansion theorem
- Two extreme approaches



N:1 Mux

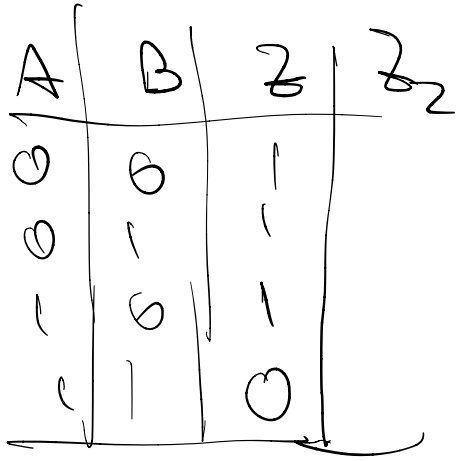


- Any Boolean function can be built using Shannon's expansion theorem
- Two extreme approaches



"Bubble" Propagation (De-Morgan's Theorem)

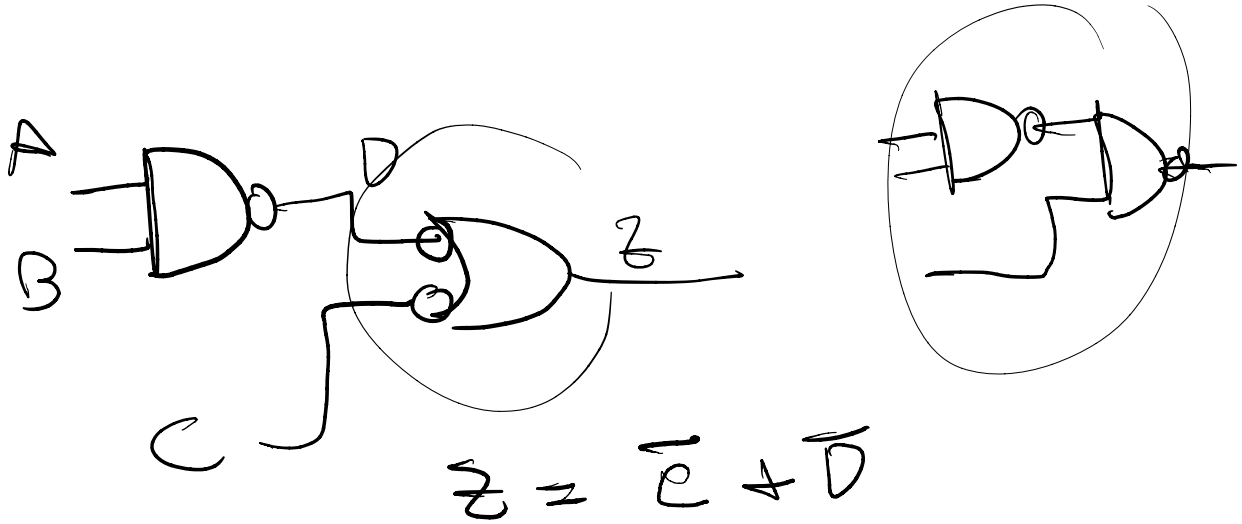
$$\bar{\bar{X}} = X$$



$$\overline{AB} = \bar{A} + \bar{B}$$

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

$\overline{AB}$



"Bubble" Propagation (De-Morgan's Theorem)

$$\bar{\bar{X}} = X$$

"Bubble" Propagation (De-Morgan's Theorem)

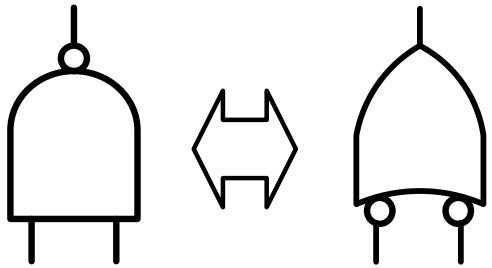
$$\bar{\bar{X}} = X$$

$$\overline{AB} = \bar{A} + \bar{B}$$

"Bubble" Propagation (De-Morgan's Theorem)

$$\bar{\bar{X}} = X$$

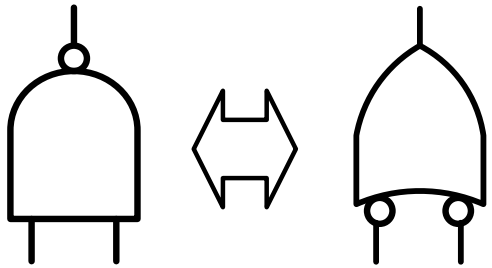
$$\overline{AB} = \bar{A} + \bar{B}$$



"Bubble" Propagation (De-Morgan's Theorem)

$$\bar{\bar{X}} = X$$

$$\overline{AB} = \bar{A} + \bar{B}$$

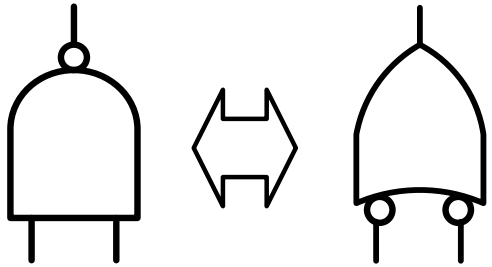


$$\overline{A + B} = \bar{A} \cdot \bar{B}$$

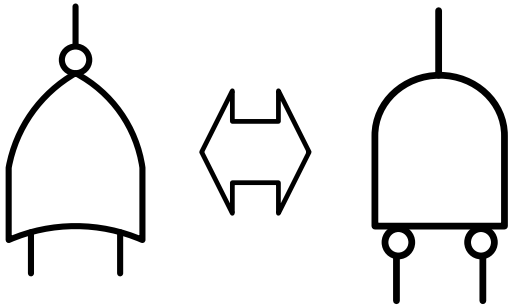
"Bubble" Propagation (De-Morgan's Theorem)

$$\bar{\bar{X}} = X$$

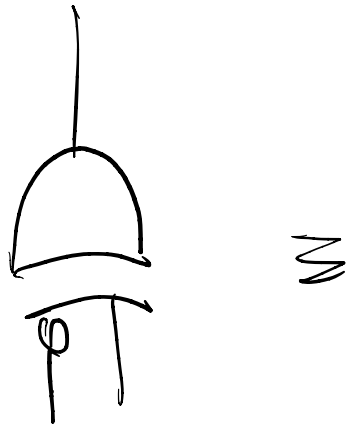
$$\overline{AB} = \bar{A} + \bar{B}$$

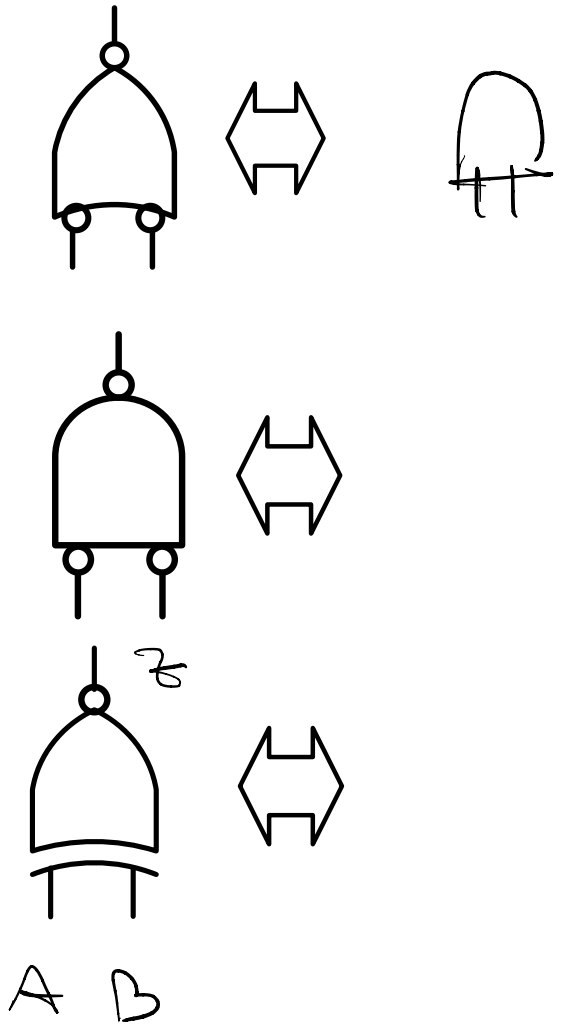


$$\overline{A + B} = \bar{A} \cdot \bar{B}$$



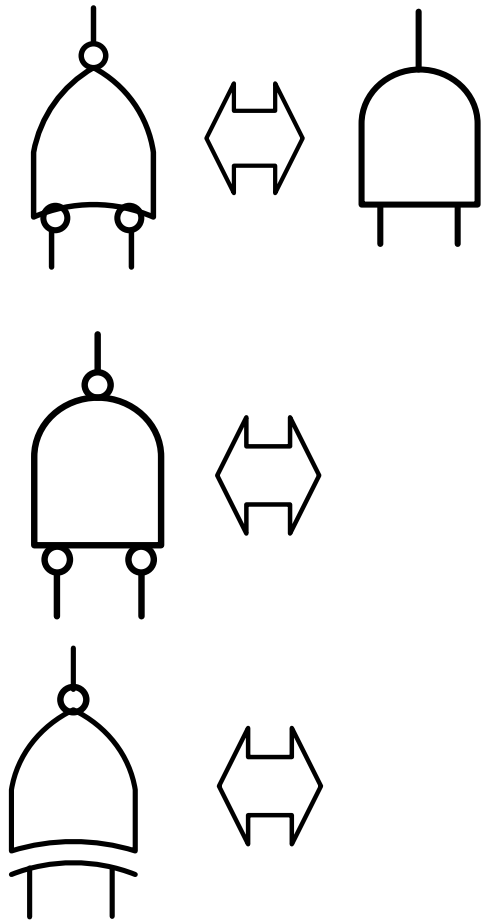
Break-out



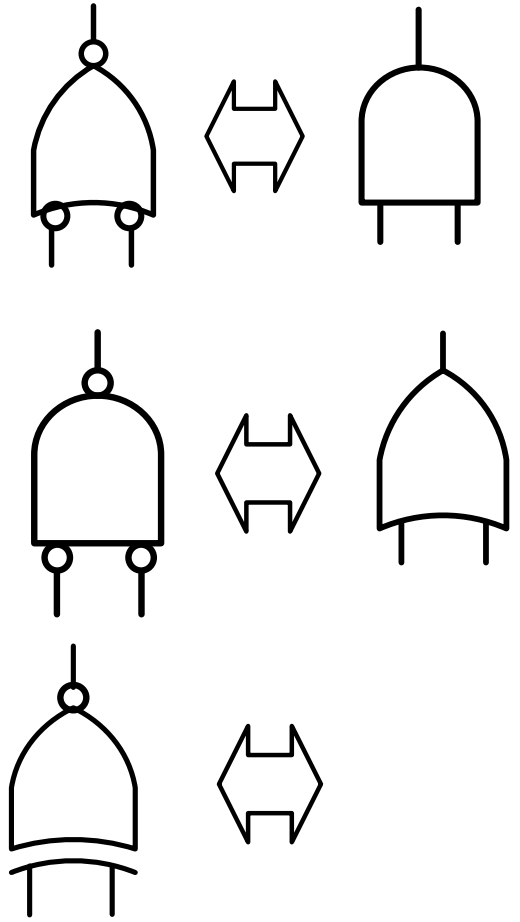


$\overline{A}$	A	B	Z
1	0	0	1
1	0	1	0
0	1	0	0
0	1	1	1

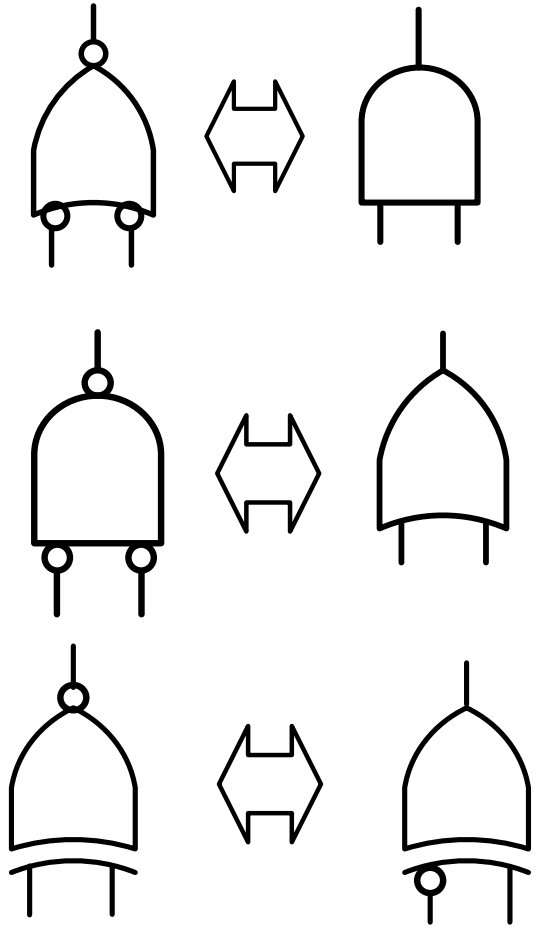
Break-out



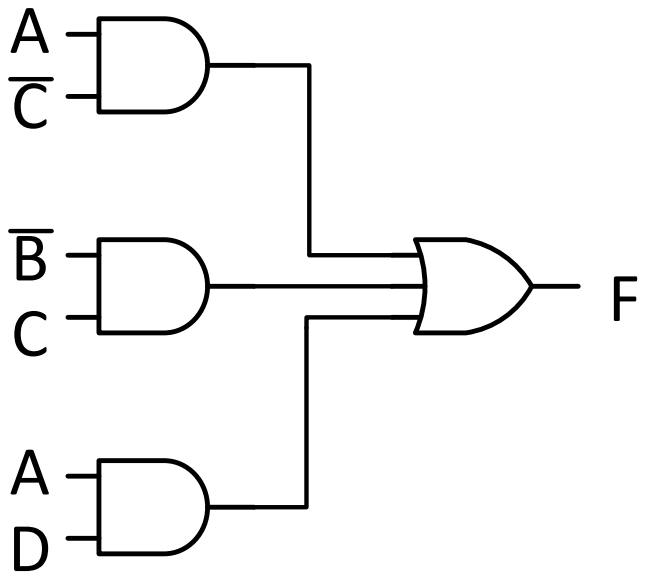
Break-out



Break-out

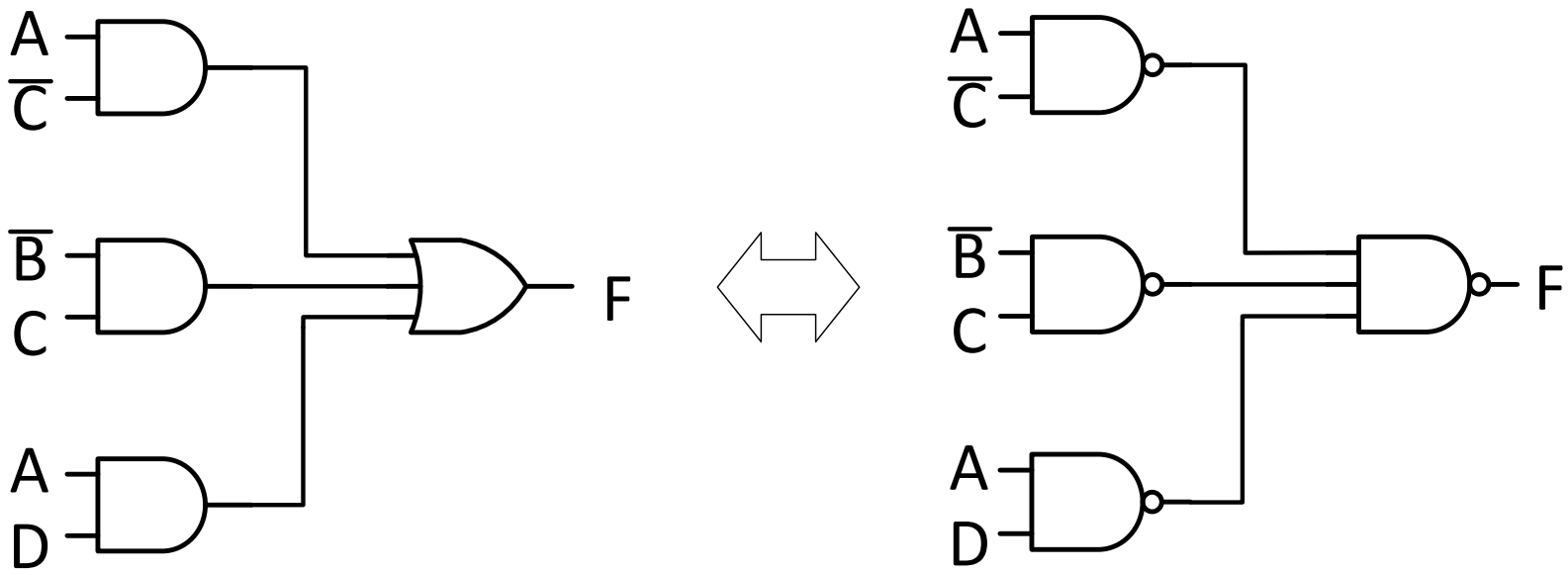


Sum of Products….



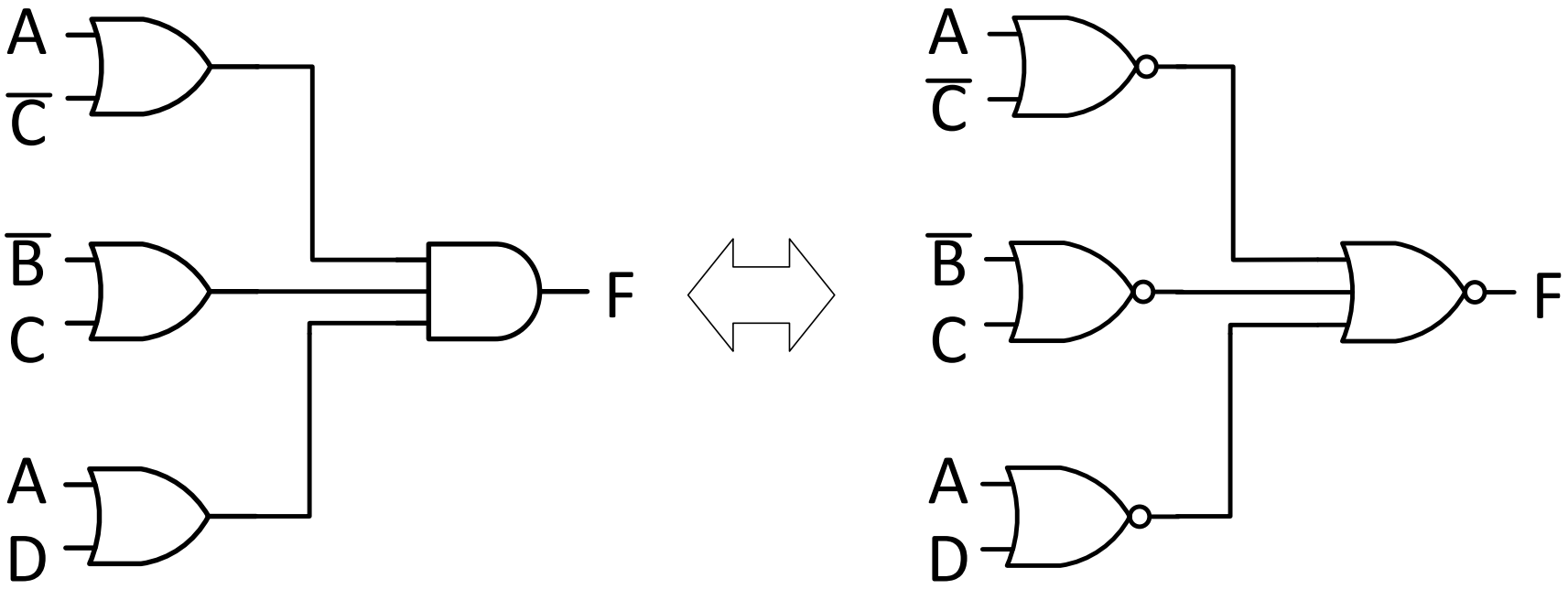
- Boolean functions often represented as sum of products
  - E.g. $F = A\bar{C} + \bar{B}C + AD$

Sum of Products….



- Boolean functions often represented as sum of products
  - E.g. $F = A\bar{C} + \bar{B}C + AD$

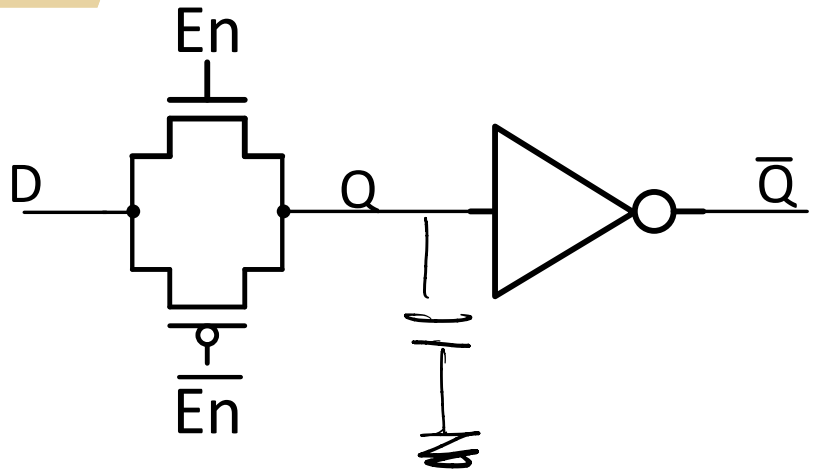
….or Product of Sums

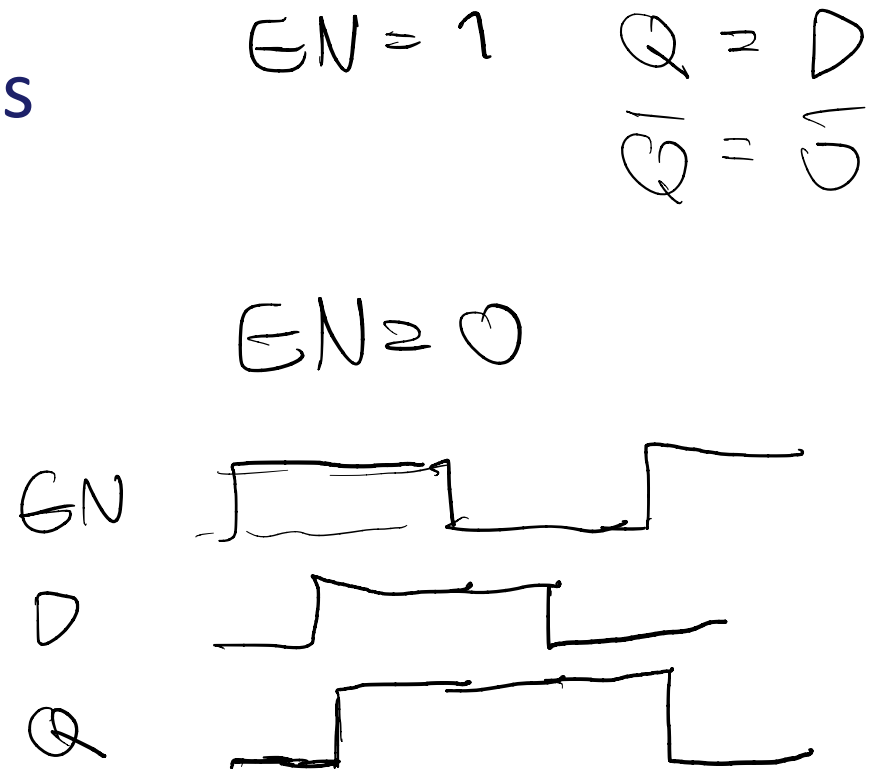


- For boolean functions often represented as product of sums
  - E.g. $F = (A + \bar{C})(\bar{B} + C)(A + D)$

Retaining state: Latches and Flip-flops

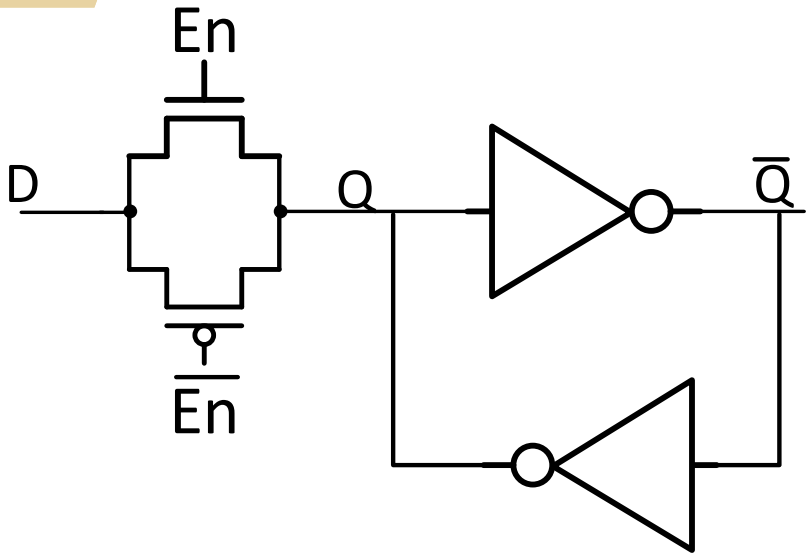
Retaining state: Latches and Flip-flops

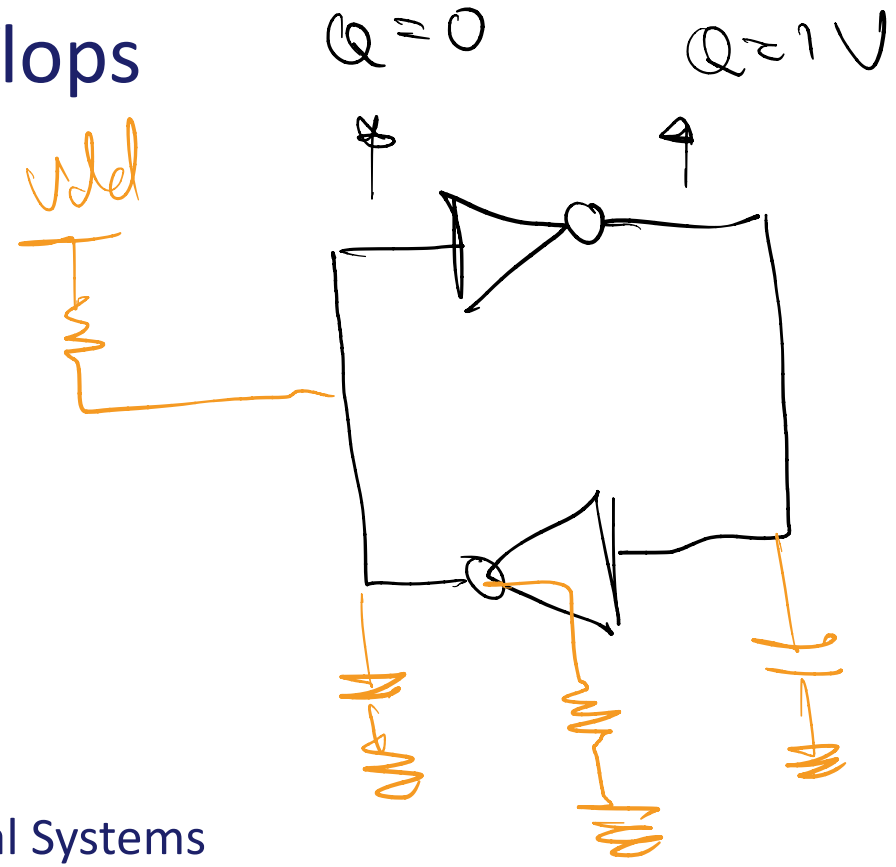




- Logic-level (or state) storage is important for Digital Systems
- Capacitance on Q allows charge (state) storage

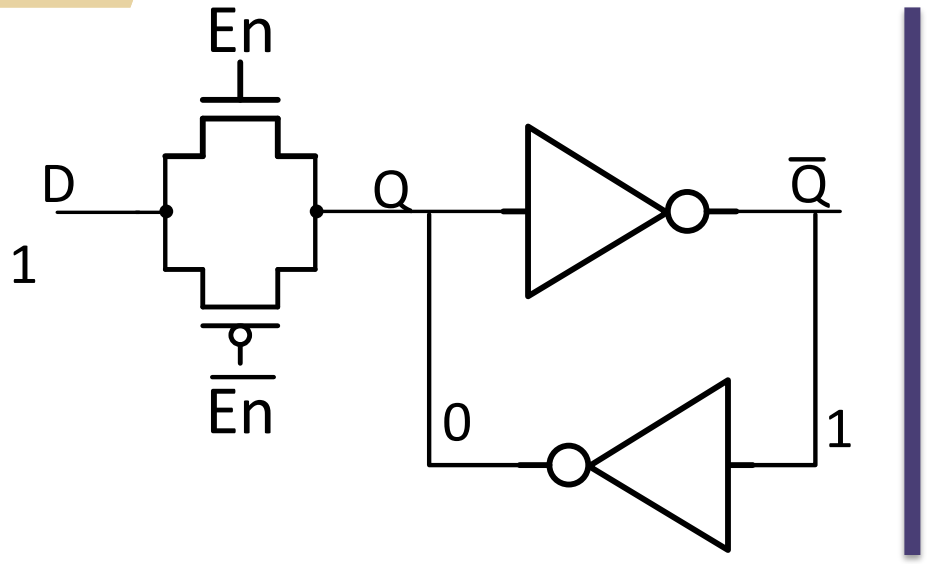
Retaining state: Latches and Flip-flops





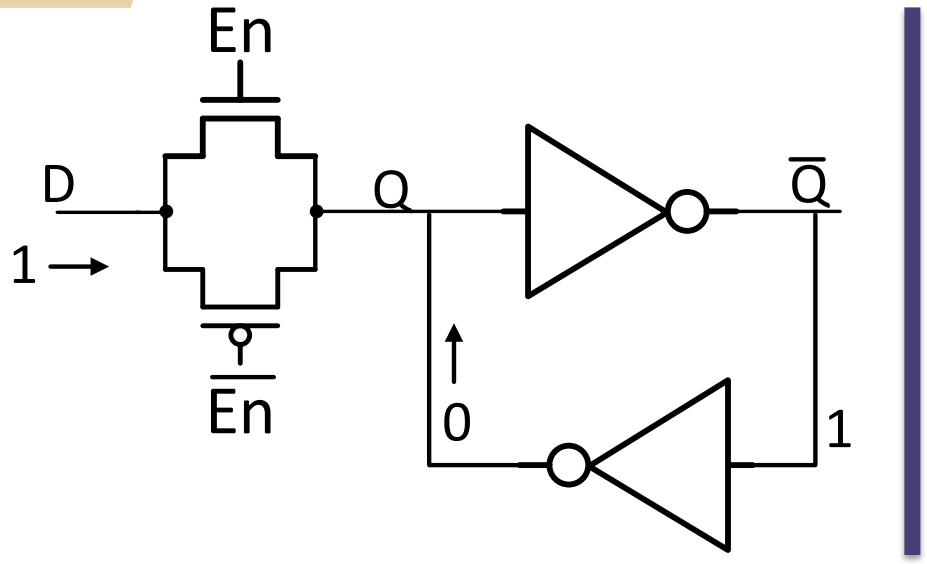
- Logic-level (or state) storage is important for Digital Systems
- Capacitance on Q allows charge (state) storage

Retaining state: Latches and Flip-flops



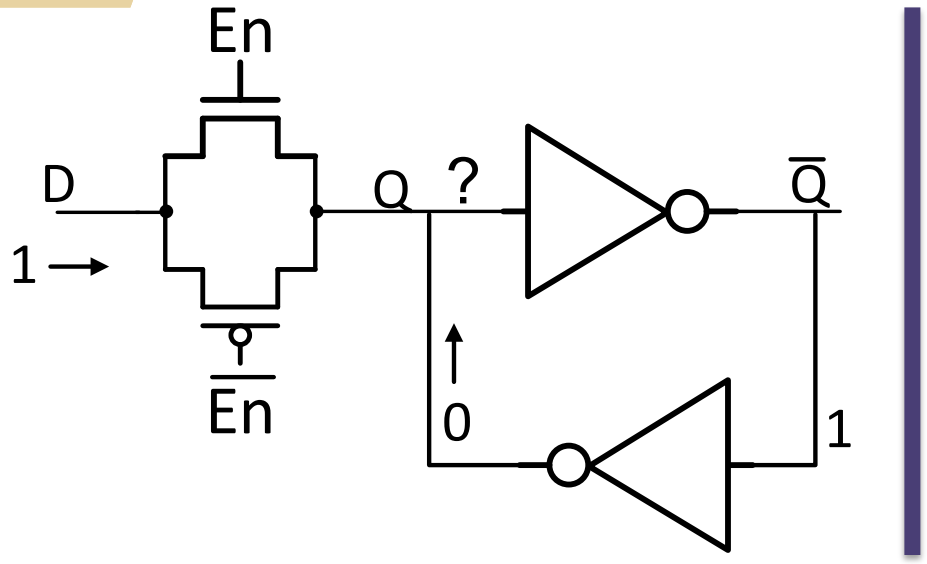
- Logic-level (or state) storage is important for Digital Systems
- Capacitance on Q allows charge (state) storage

Retaining state: Latches and Flip-flops



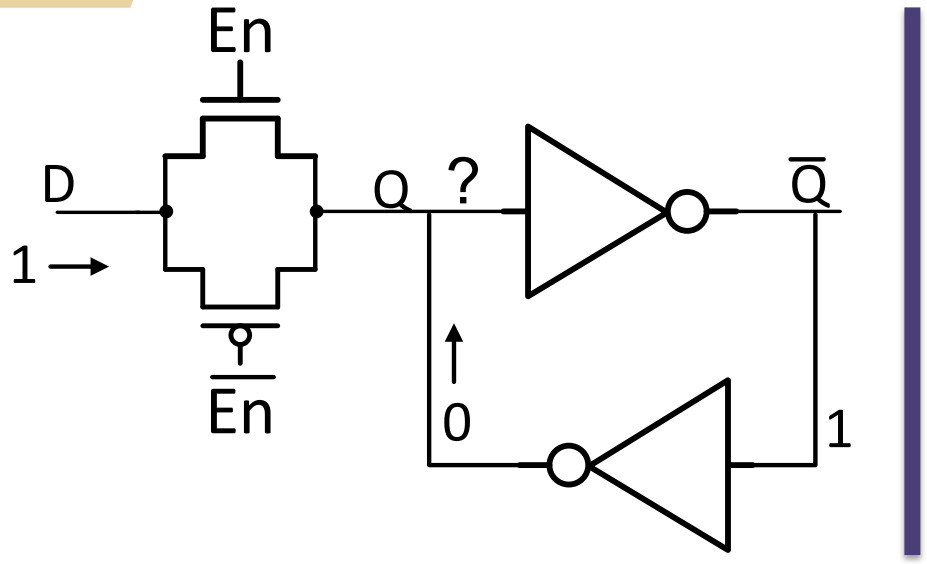
- Logic-level (or state) storage is important for Digital Systems
- Capacitance on Q allows charge (state) storage

Retaining state: Latches and Flip-flops



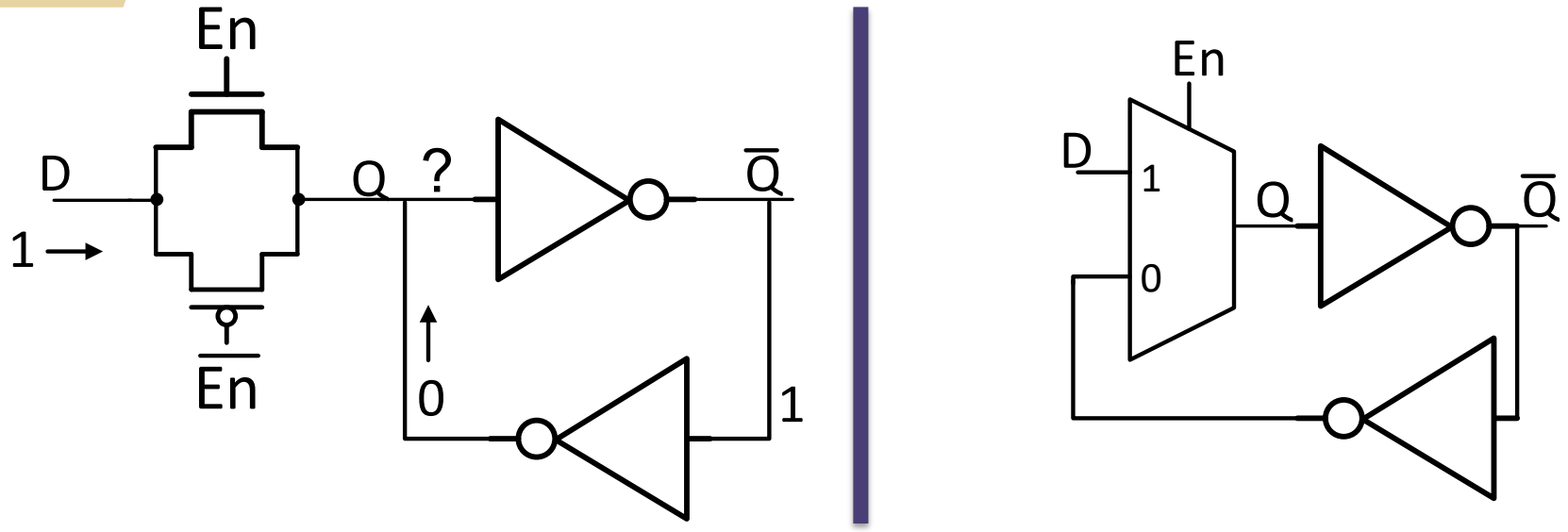
- Logic-level (or state) storage is important for Digital Systems
- Capacitance on Q allows charge (state) storage

Retaining state: Latches and Flip-flops



- Logic-level (or state) storage is important for Digital Systems
- Capacitance on Q allows charge (state) storage
- Avoid fight
  - Select D to drive Q if En = 1
  - Select Qx to drive Q if En = 0

Retaining state: Latches and Flip-flops



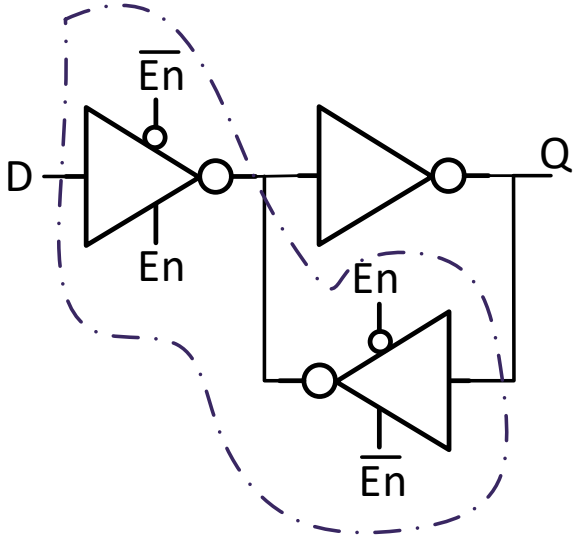
- Logic-level (or state) storage is important for Digital Systems
- Capacitance on Q allows charge (state) storage
- Avoid fight
  - Select D to drive Q if En = 1
  - Select Qx to drive Q if En = 0

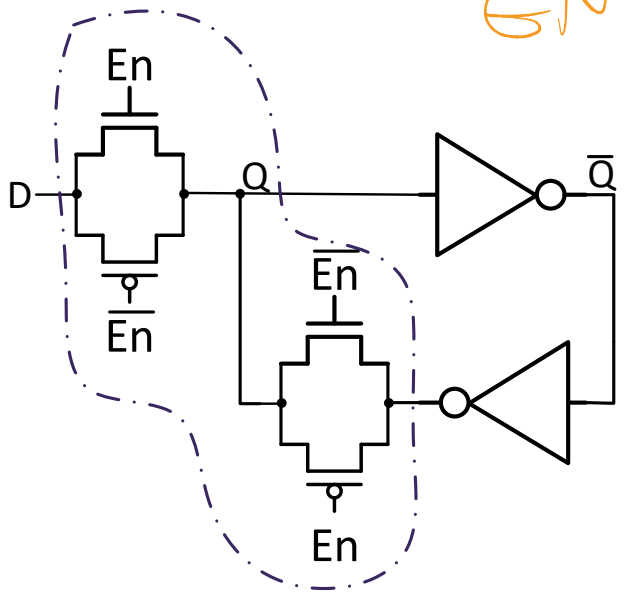
Latch: Basic Structure and Operation

EN = 1 Q = D

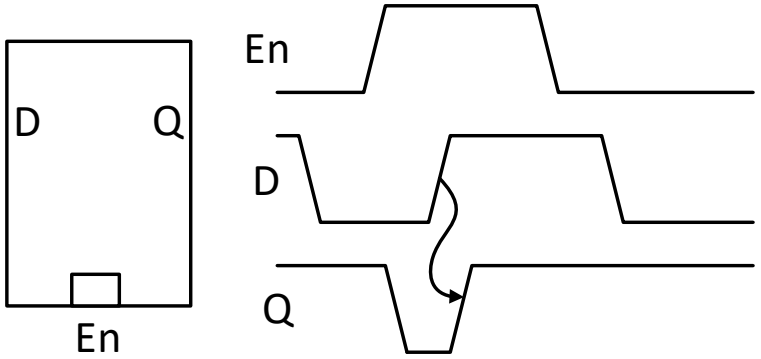
EN = 0 Q = Q

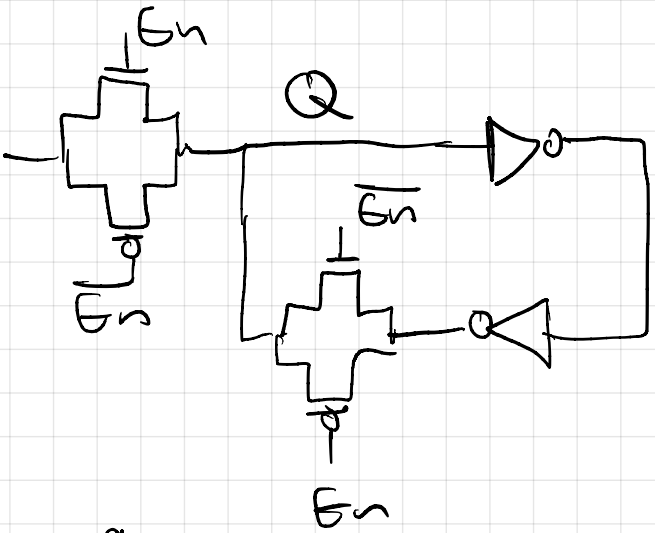
Q retains its state (value)



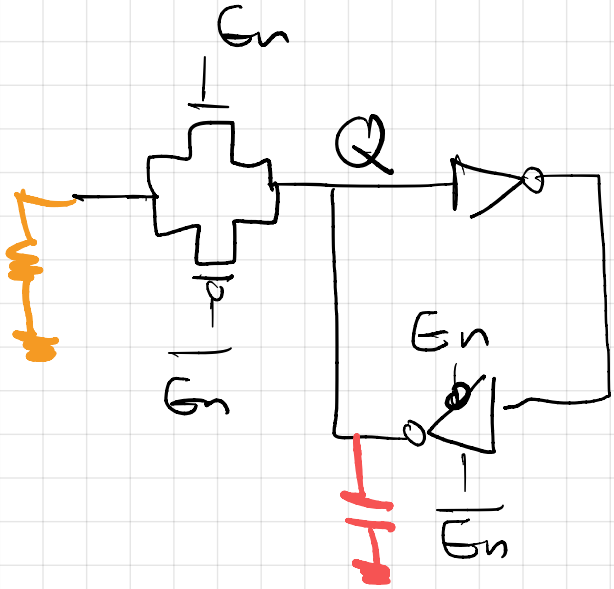


- Latch written when En=1
- Holds state if En=0
- “Level-sensitive” timing element
- Latch “transparent” when En=1, opaque when En=0

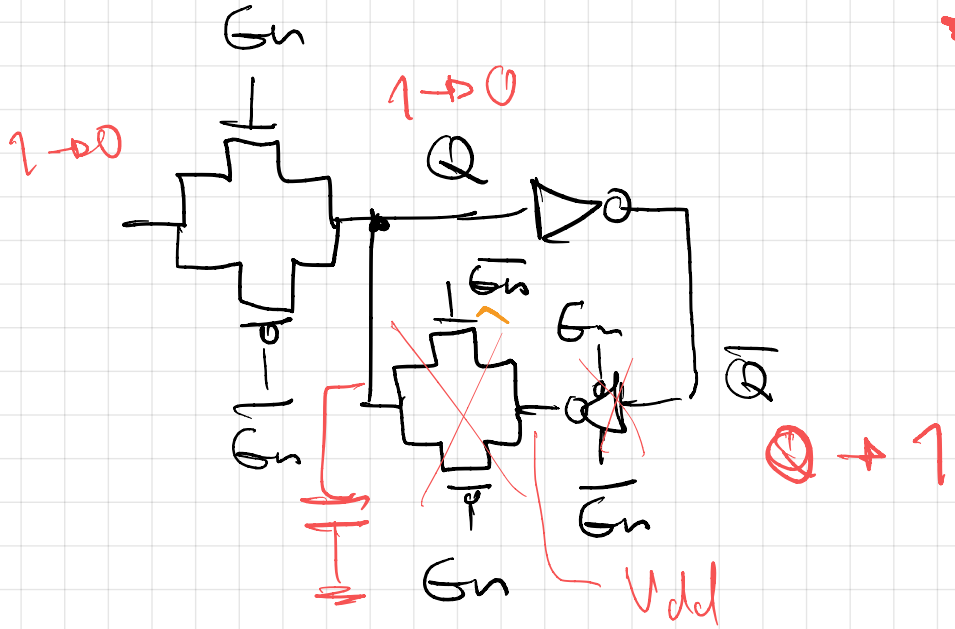


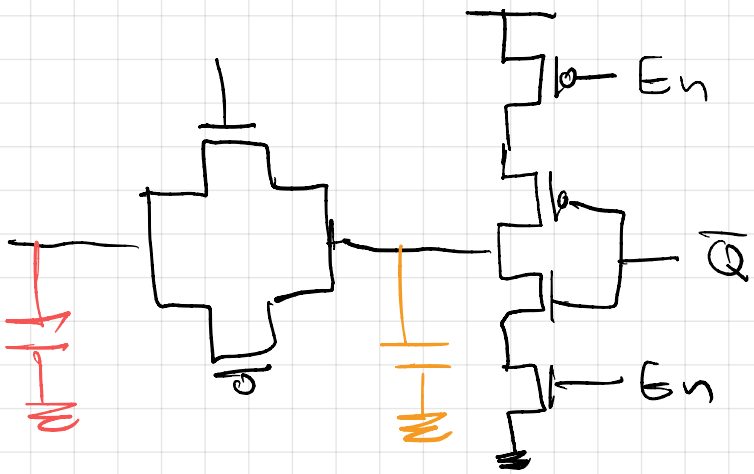

En
Q
En
En
En

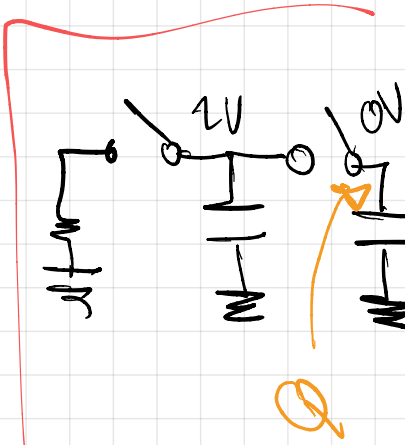
1)


En
Q
En
En
En

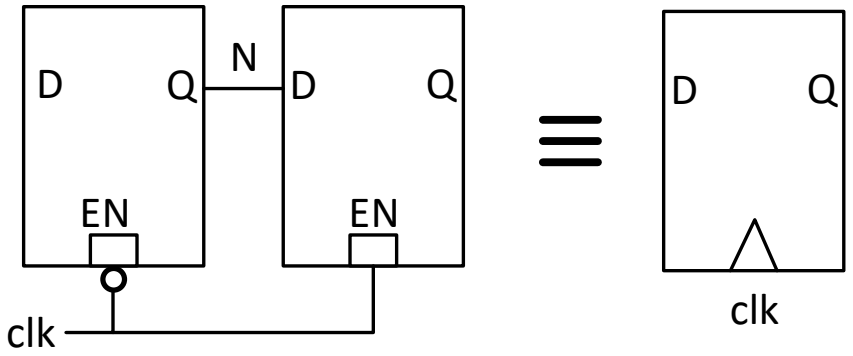
2)

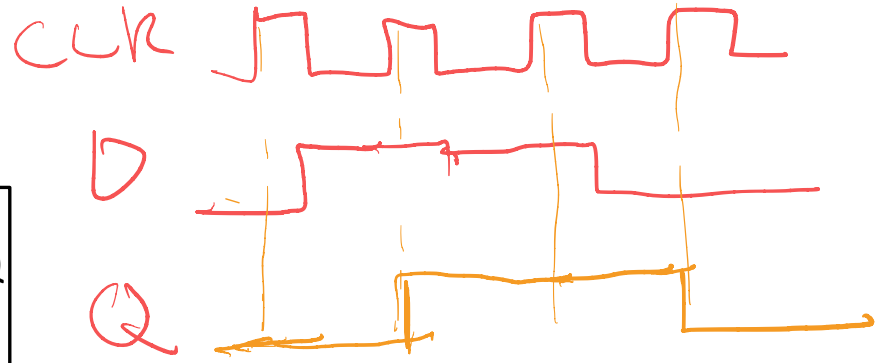

3)
En
1→0
1→0
Q
En
En
En
En
Q
Q → 1
Vdd

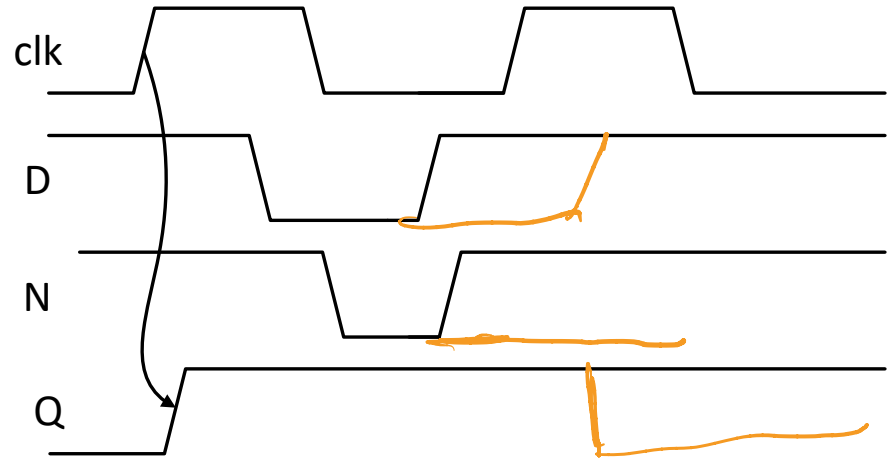

En
Q
En


2V
0V
Q

Flip Flop

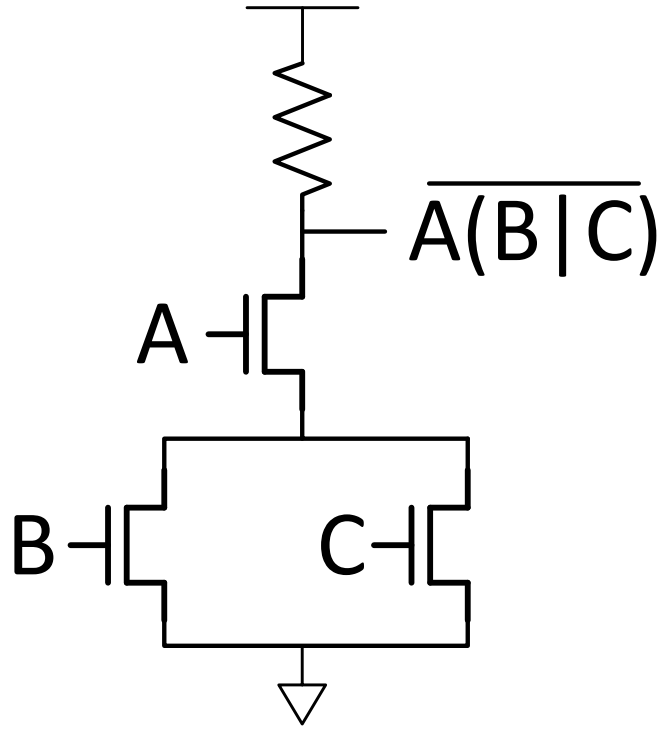


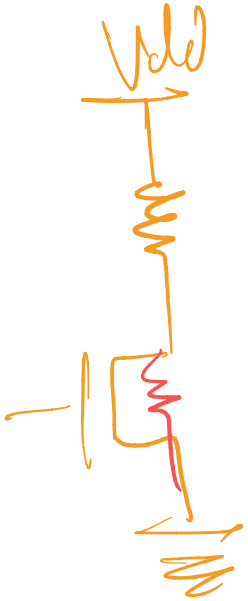




- Edge-triggered timing element (“clocked instead of enabled”)
- Sample on the rising or falling edge of clock

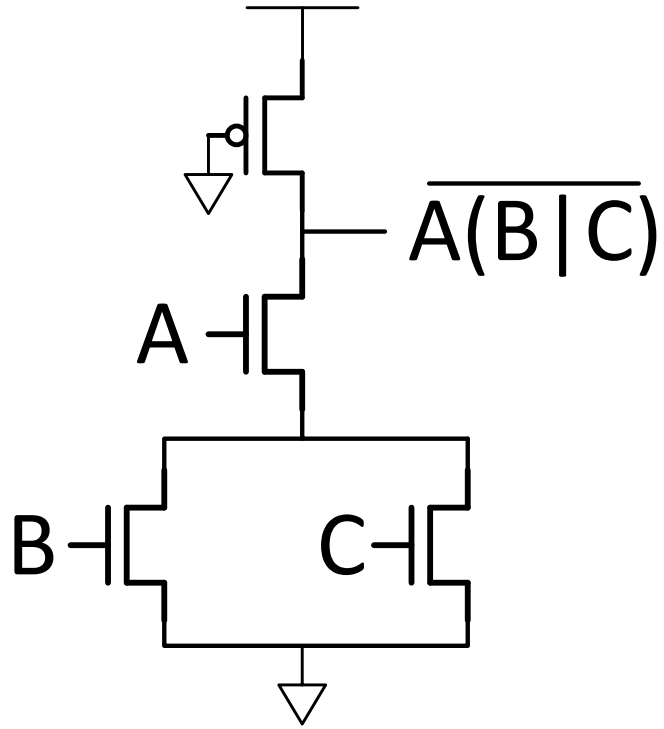
Pseudo NMOS: An alternative?





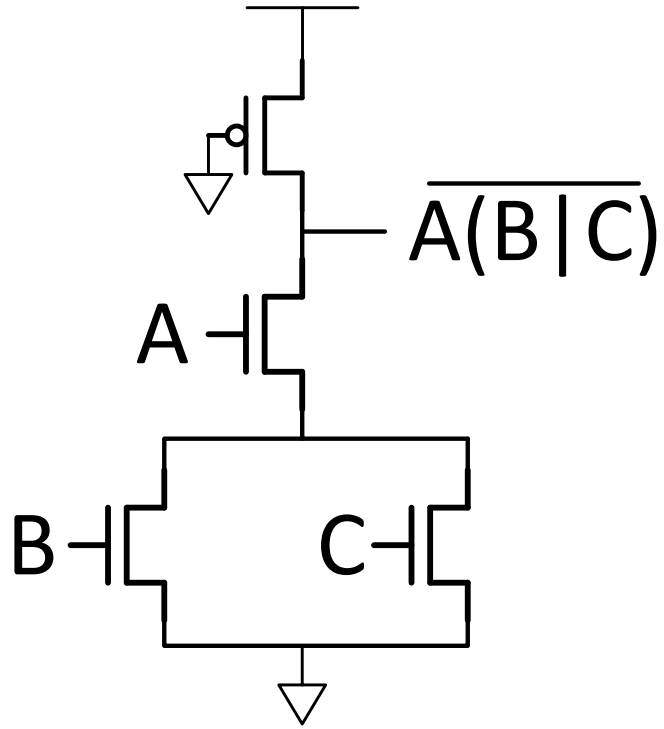
- Pseudo NMOS logic
  - Use resistive load with NMOS PDN

Pseudo NMOS: An alternative?



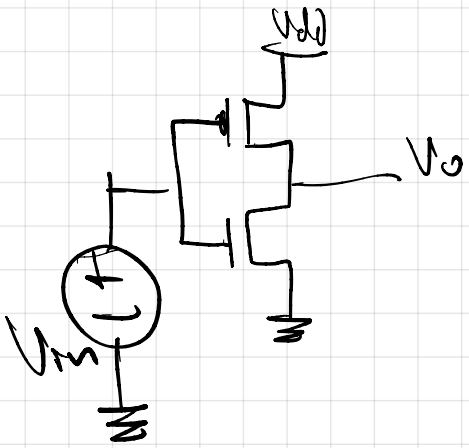
- Pseudo NMOS logic
  - Use resistive load with NMOS PDN

Pseudo NMOS: An alternative?

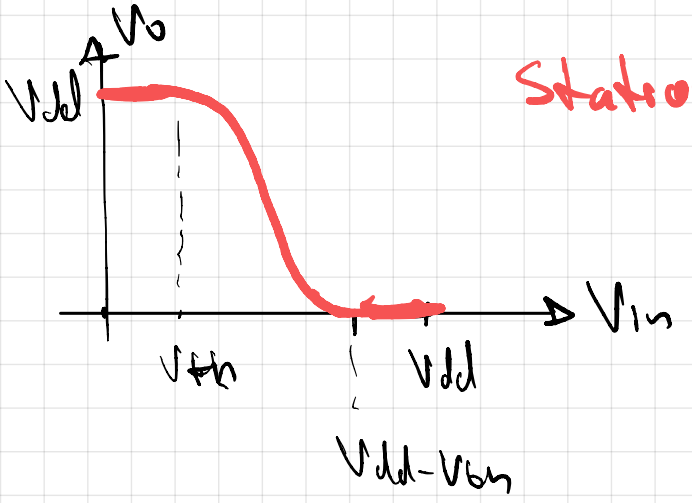


- Pseudo NMOS logic
  - Use resistive load with NMOS PDN

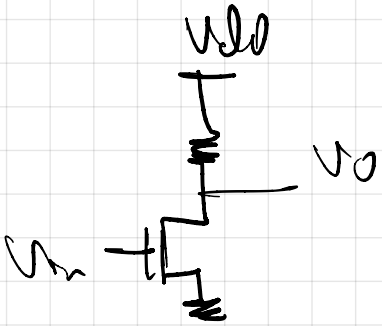
What does Vout vs. Vin look like for this gate

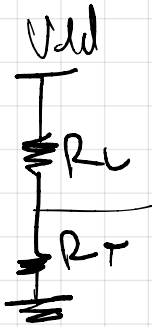


Logic	Voltage
0	0 V
1	Vdd



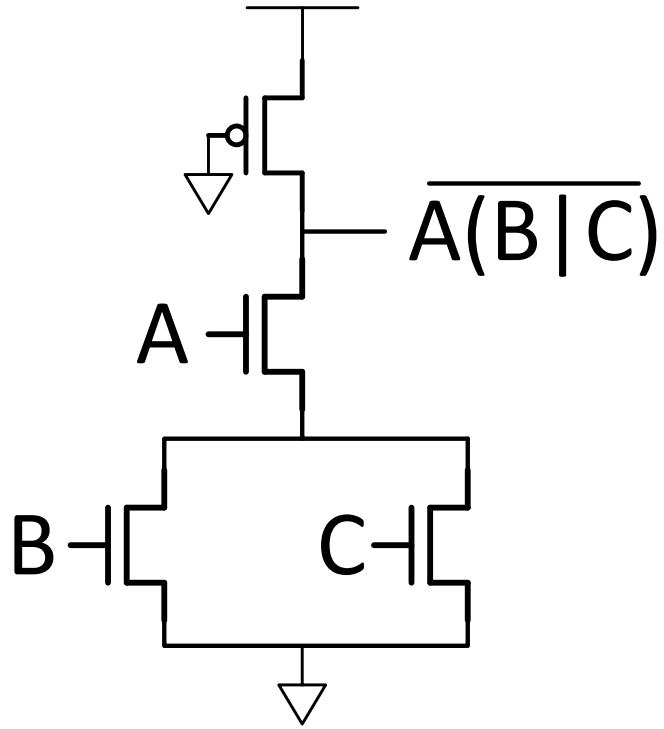
Stationary current = 0 CMOS





$$V_o = V_{dd} \cdot \frac{R_T}{R_T + R_L}$$

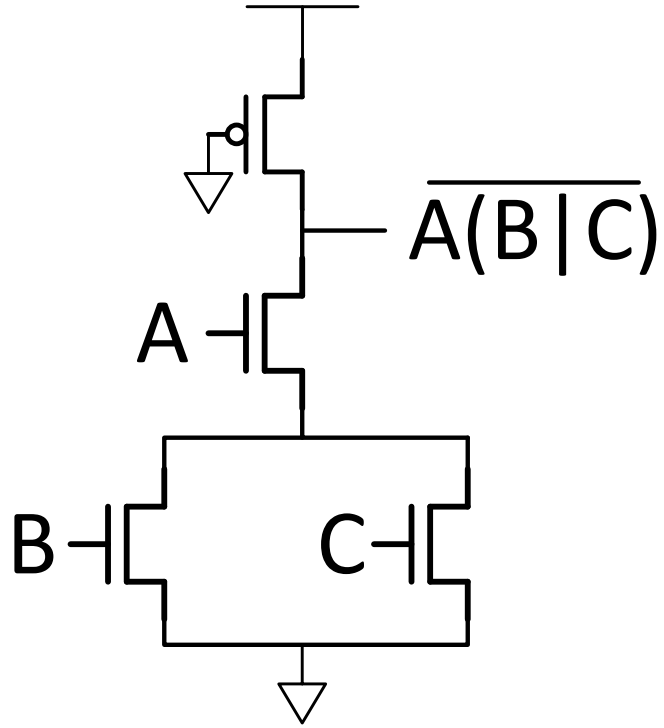
Pseudo NMOS: An alternative?

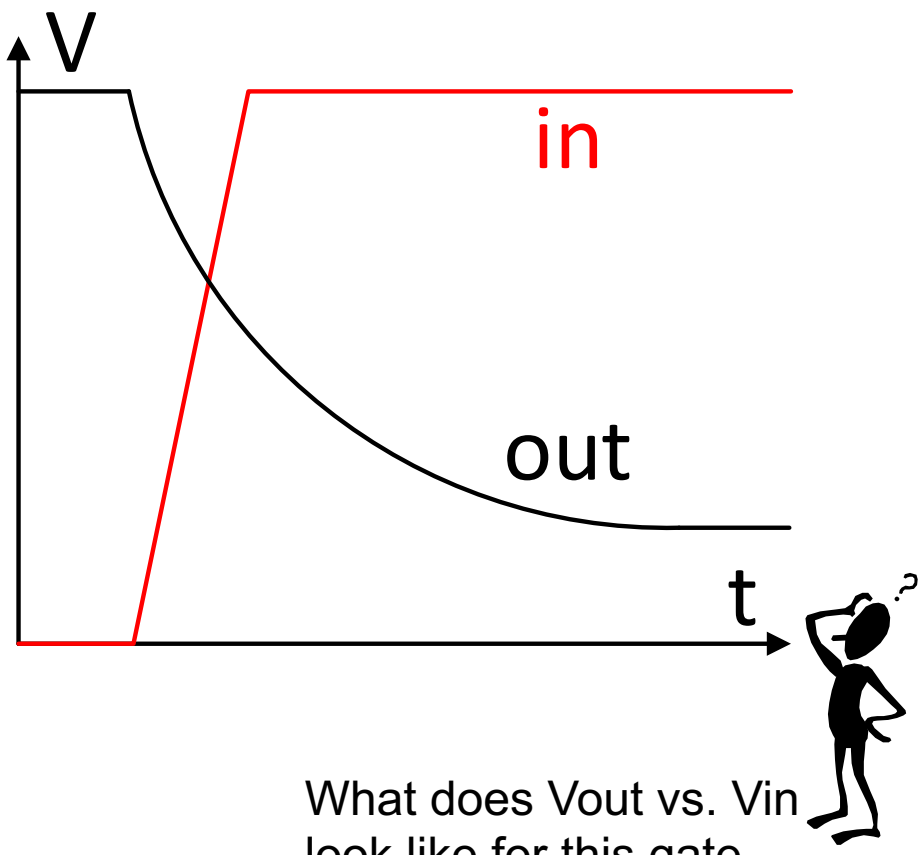


- Pseudo NMOS logic
  - Use resistive load with NMOS PDN
  - Challenges: Voltage swing, Power dissipation

What does Vout vs. Vin look like for this gate

Pseudo NMOS: An alternative?

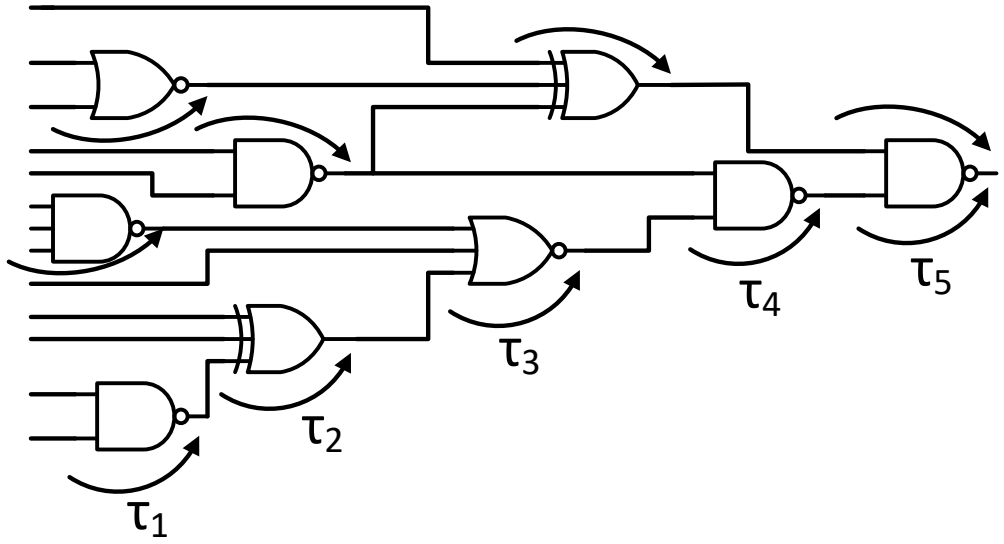




- Pseudo NMOS logic
  - Use resistive load with NMOS PDN
  - Challenges: Voltage swing, Power dissipation

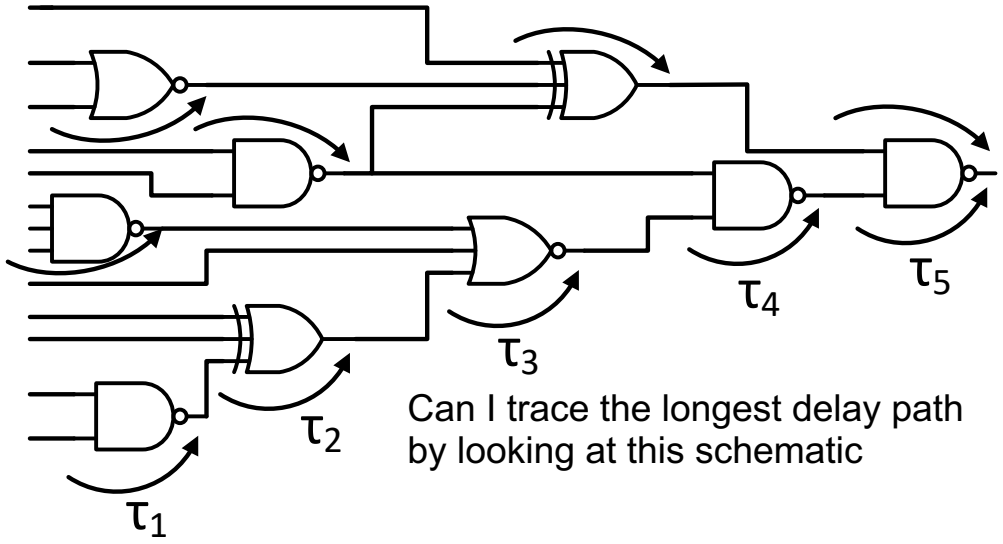
What does Vout vs. Vin look like for this gate

Performance, gate delay, and drive-strength



- Real world digital systems: M(B)illions of CMOS logic gates
  - Performance heavily depends on circuit speed
  - Logical function ➔ Gate implementation ➔ Physical implementation. What governs speed?

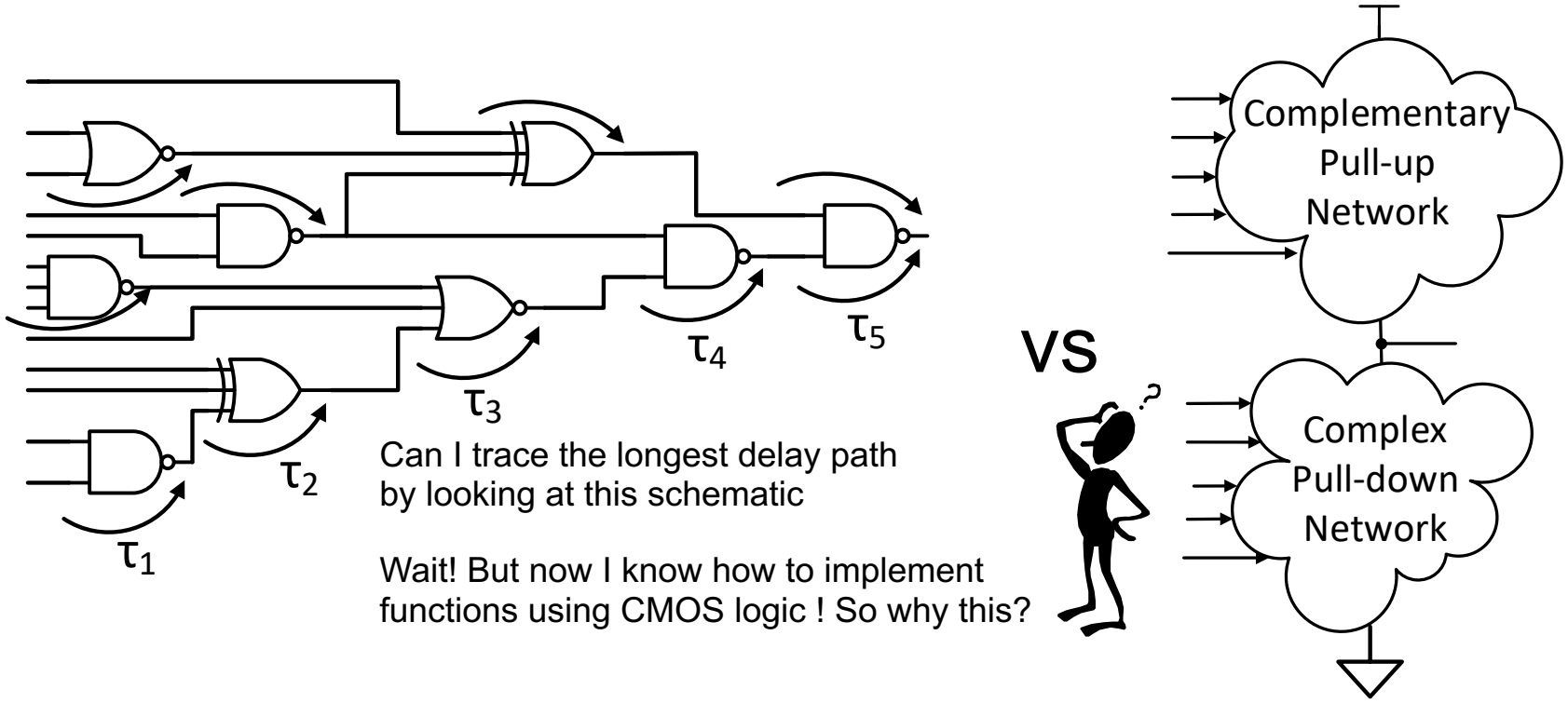
Performance, gate delay, and drive-strength



Can I trace the longest delay path by looking at this schematic

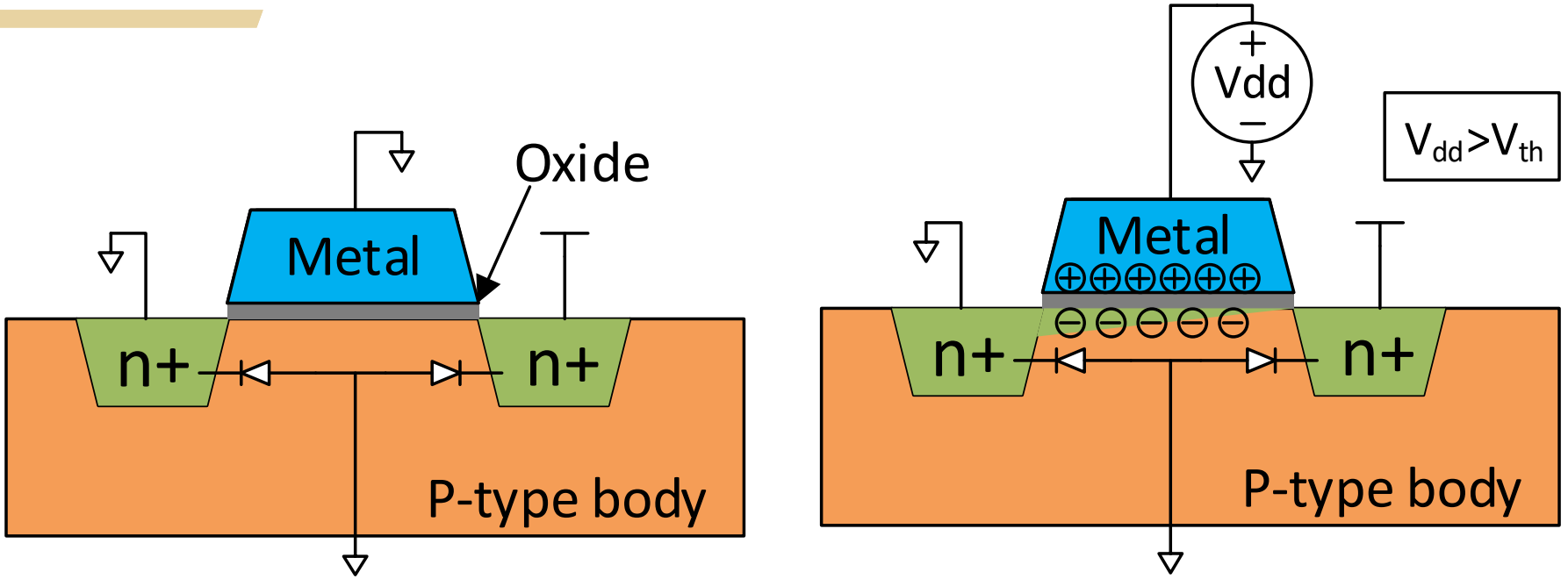
- Real world digital systems: M(B)illions of CMOS logic gates
  - Performance heavily depends on circuit speed
  - Logical function ➔ Gate implementation ➔ Physical implementation. What governs speed?

Performance, gate delay, and drive-strength



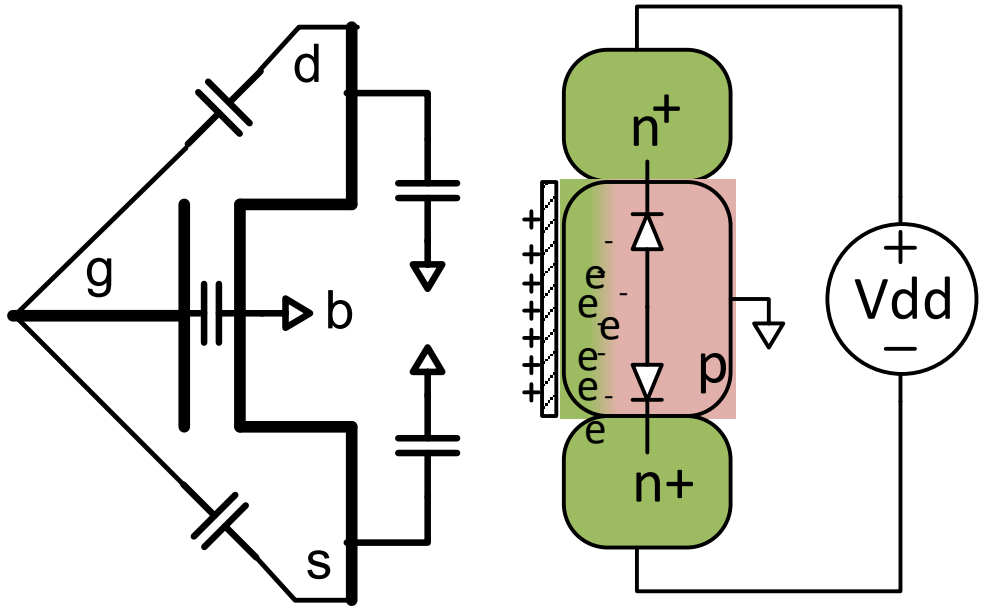
- Real world digital systems: M(B)illions of CMOS logic gates
  - Performance heavily depends on circuit speed
  - Logical function ➔ Gate implementation ➔ Physical implementation. What governs speed?

The MOS Structure



- Metal Oxide Semiconductor
  - "Metal" implemented as poly-crystalline silicon (polysilicon) till recently
  - Conductor-insulator-semiconductor "sandwich"
  - Changing the voltage across Metal-Semiconductor varies the properties of the semiconductor

Quick Detour: Parasitic Capacitance

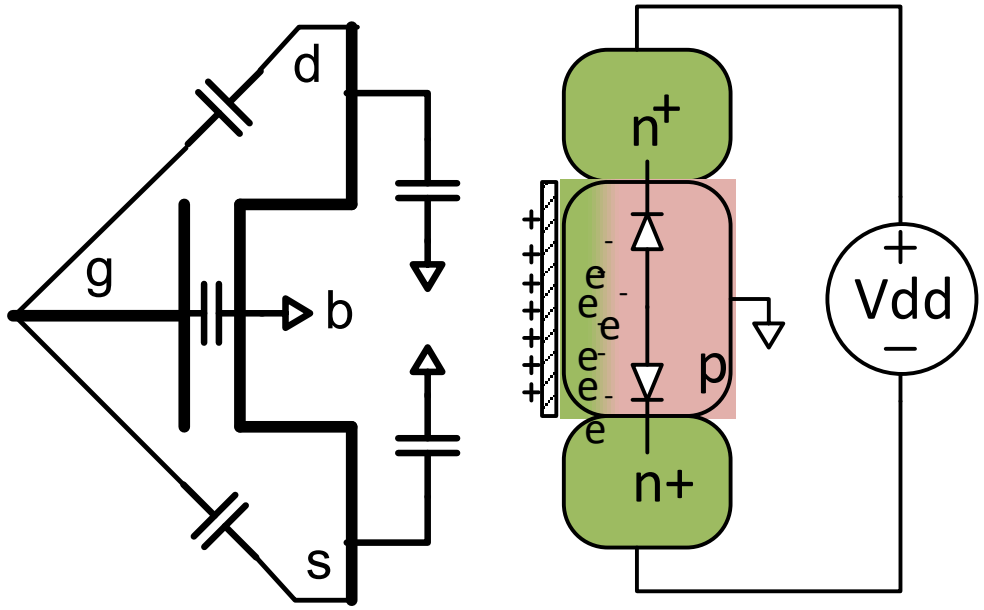


Impact

- Circuit Speed (Digital)
- Power (Digital)
- Bandwidth (Analog)
- Stability (Digital, Analog)

- What is capacitance exactly?
  - So what's so bad about it?
    - V-driven, not Q-driven
    - Low C → Lower Q, faster, lower energy
    - Smaller is faster (Dennard's law)
  - Major Parasitic contributors
    - MOSFET gates, MOSFET source/drains, wire

Quick Detour: Parasitic Capacitance



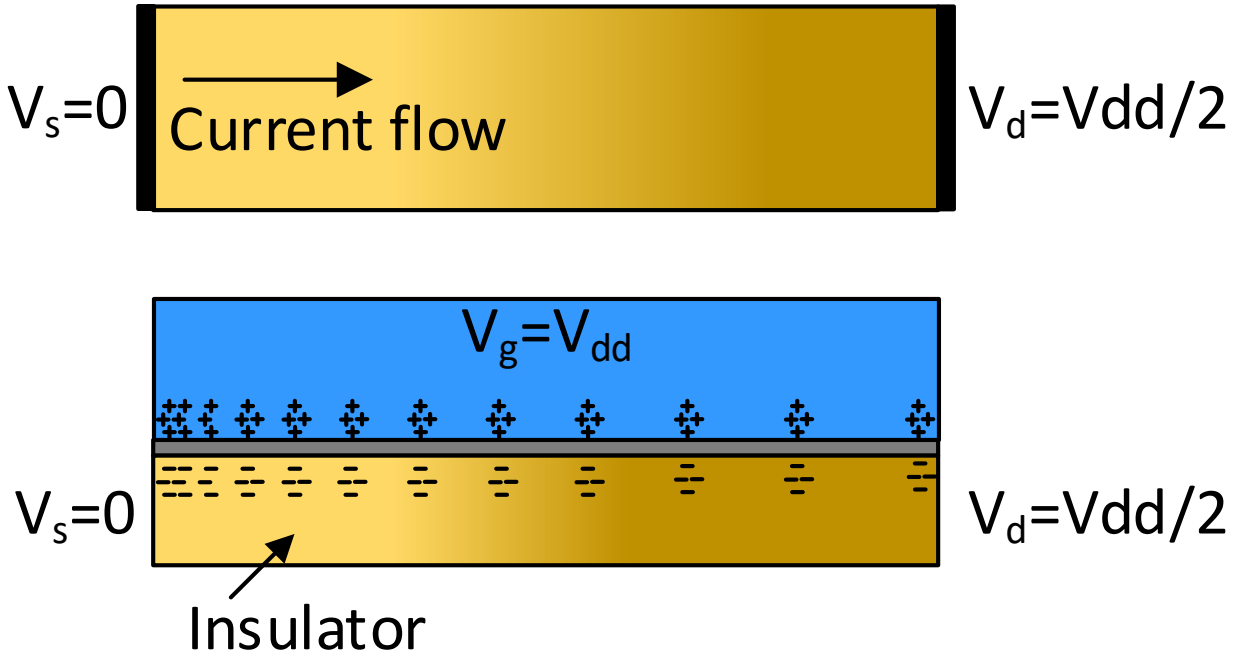
Impact

- Circuit Speed (Digital)
- Power (Digital)
- Bandwidth (Analog)
- Stability (Digital, Analog)

- What is capacitance exactly?
  - So what's so bad about it?
    - V-driven, not Q-driven
    - Low C → Lower Q, faster, lower energy
    - Smaller is faster (Dennard's law)
  - Major Parasitic contributors
    - MOSFET gates, MOSFET source/drains, wire

Calculate the per-unit length capacitance of an infinitely long cylindrical wire, diameter d in free space

MOS Transistor – A 10K foot view



- Capacitor charge buildup allows for a “channel” to form
- Charge density not uniform ➔ ? Impact on R. ? Impact on V(x)
- How can I reduce the resistance of this device further?

Reading/Study assignment

- Reading – 2.1, 2.2, 2.3.1
- Optional Reading – 2.3.*, 2.4.*

- Homework (**IMPORTANT**)
  1. What is the resistance of a cylindrical conductor (in the height direction) given its conductivity, σ, radius *r*
  2. Review the MOS current equations in saturation and linear modes