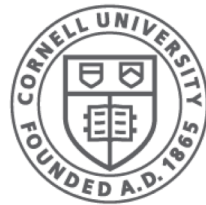


# Gates and Logic: From Transistors to Logic Gates and Logic Circuits

CS 3410: Computer System Organization and  
Programming



**Cornell CIS**  
COMPUTING AND INFORMATION SCIENCE

[K. Bala, A. Bracy, E. Sirer, and H. Weatherspoon]

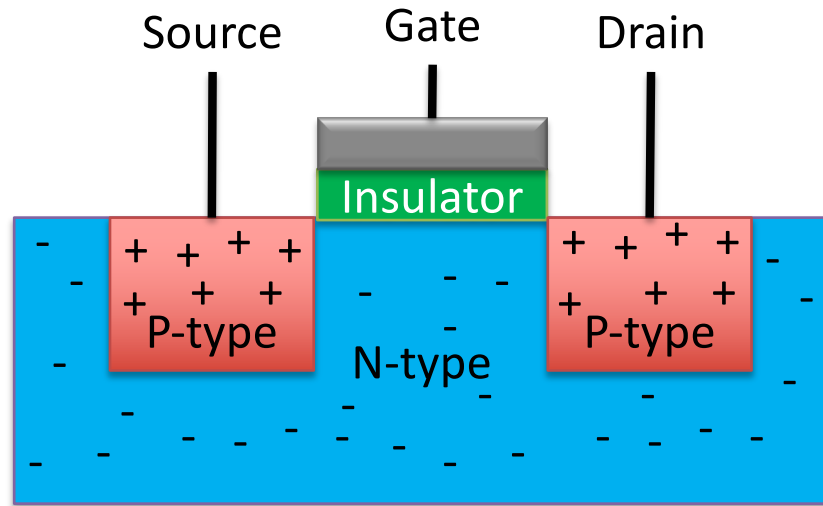
# Goals for Today

- From Switches to Logic Gates to Logic Circuits
- Transistors, Logic Gates, Truth Tables
- Logic Circuits
  - Identity Laws
  - From Truth Tables to Circuits (Sum of Products)

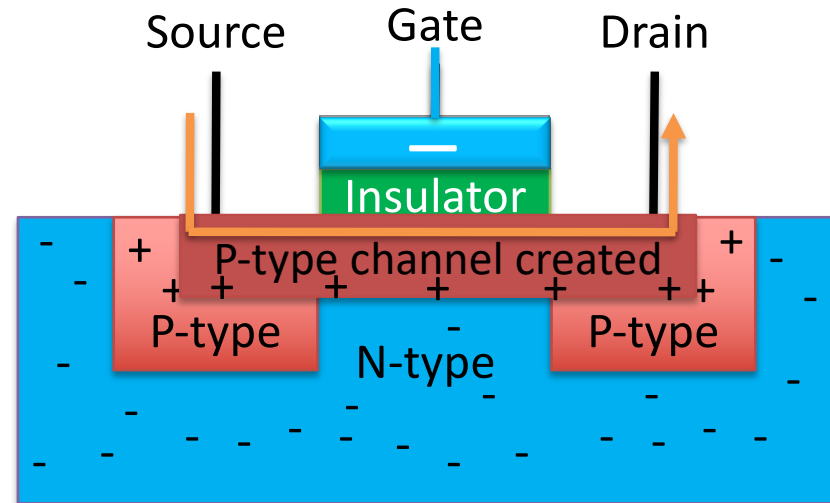
# Silicon Valley & the Semiconductor Industry

- Transistors:
- Youtube video “How does a transistor work”  
<https://www.youtube.com/watch?v=IcrBqCFLHIY>
- Break: show some Transistor, Fab, Wafer photos

# Transistors 101



Off



On

**N-Type Silicon:** negative free-carriers (electrons)

**P-Type Silicon:** positive free-carriers (holes)

**P-Transistor:** negative charge on gate generates electric field that creates a (+ charged) p-channel connecting source & drain

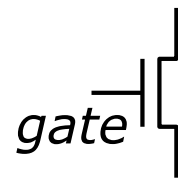
**N-Transistor:** works the opposite way

Metal-Oxide Semiconductor (Gate-Insulator-Silicon)

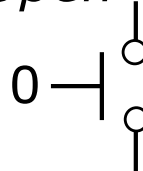
- Complementary MOS = **CMOS** technology uses both p- & n-type transistors

# CMOS Notation

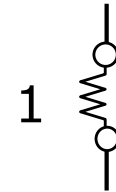
N-type



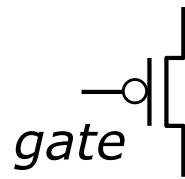
*Off/Open*



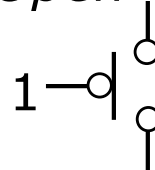
*On/Closed*



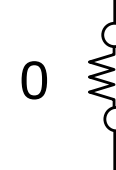
P-type



*Off/Open*



*On/Closed*



Gate input controls whether current can flow between the other two terminals or not.

*Hint:* the "o" bubble of the p-type tells you that this gate wants a 0 to be turned on

## iClicker Question

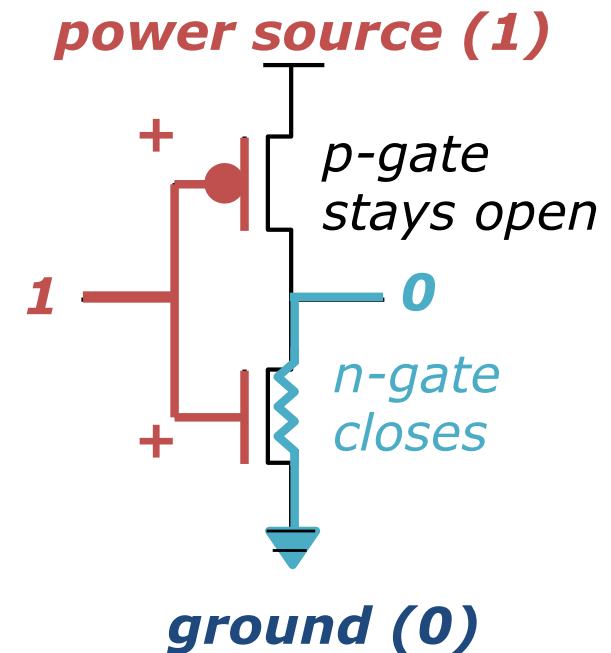
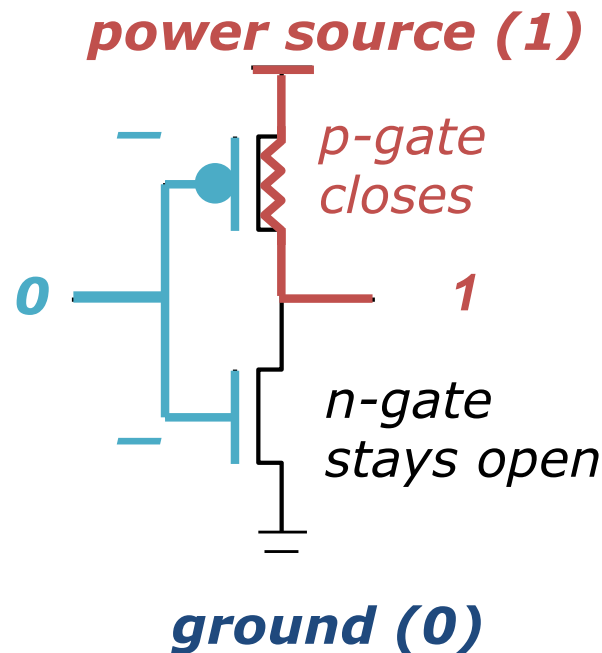
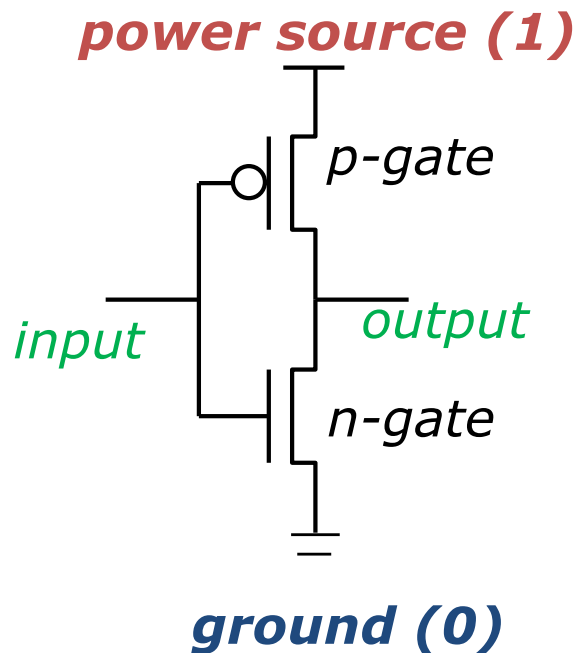
Which of the following statements is **false**?

- (A) P- and N-type transistors are both used in CMOS designs.
- (B) As transistors get smaller, the frequency of your processor will keep getting faster.
- (C) As transistors get smaller, you can fit more and more of them on a single chip.
- (D) Pure silicon is a semi conductor.
- (E) Experts believe that Moore's Law will soon end.

# 2-Transistor Combination: NOT

- Logic gates are constructed by combining transistors in complementary arrangements
- Combine p&n transistors to make a NOT gate:

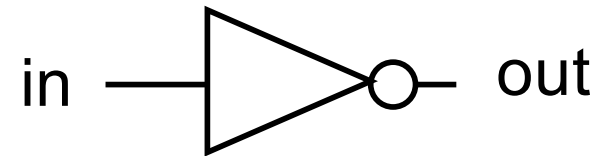
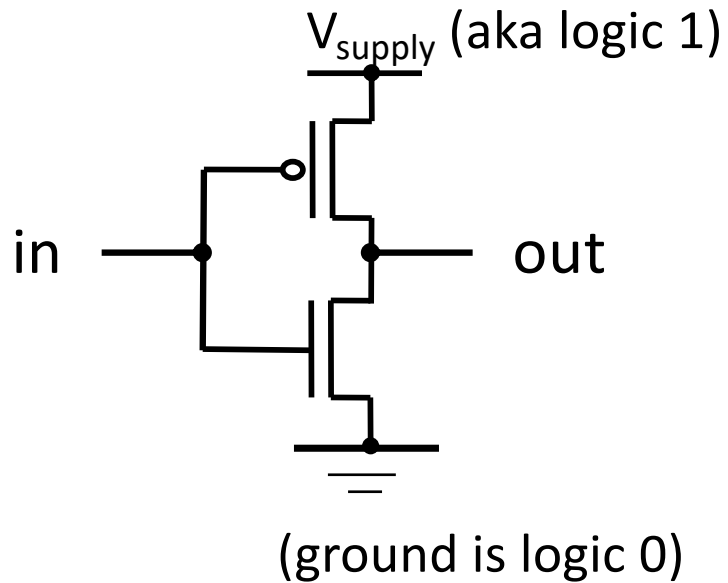
*CMOS Inverter :*



# Inverter

Function: NOT

Symbol:



Truth Table:

In	Out
0	1
1	0



# Logic Gates

- Digital circuit that either allows signal to pass through it or not
- Used to build logic functions
- Seven basic logic gates:

**AND**

**OR**

**NOT,**

**NAND** (not AND),

**NOR** (not OR),

**XOR**

**XNOR** (not XOR)



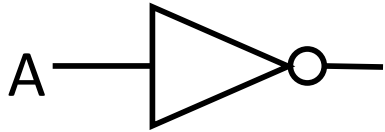
**George Boole, (1815–1864)**

**Did you know?**

**George Boole** Inventor of the idea of logic gates. He was born in Lincoln, England and he was the son of a shoemaker.

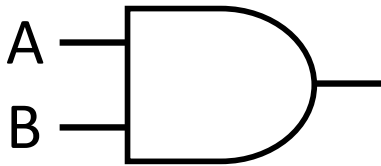
# Logic Gates: Names, Symbols, Truth Tables

NOT:



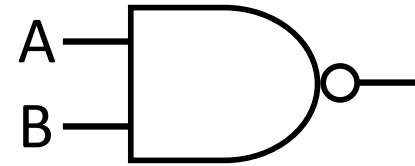
A	Out
0	1
1	0

AND:



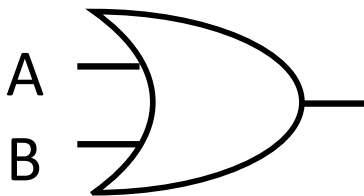
A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1

NAND:



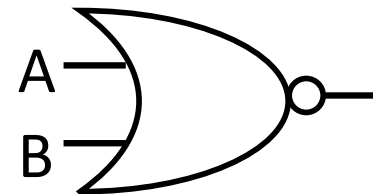
A	B	Out
0	0	1
0	1	1
1	0	1
1	1	0

OR:



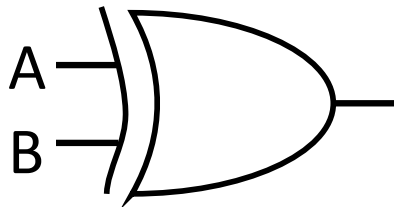
A	B	Out
0	0	0
0	1	1
1	0	1
1	1	1

NOR:



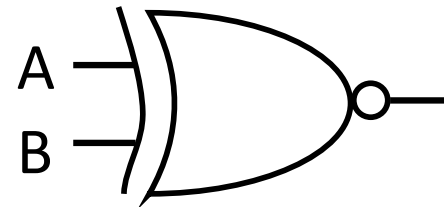
A	B	Out
0	0	1
0	1	0
1	0	0
1	1	0

XOR:



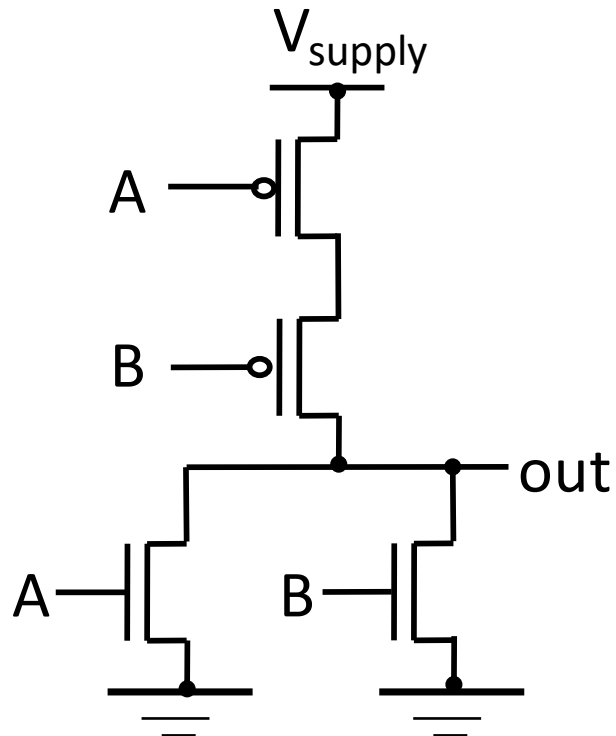
A	B	Out
0	0	0
0	1	1
1	0	1
1	1	0

XNOR:



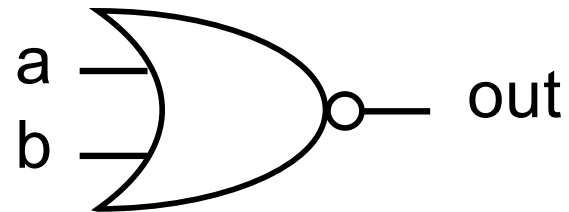
A	B	Out
0	0	1
0	1	0
1	0	0
1	1	1

# NOR Gate



Function: NOR

Symbol:

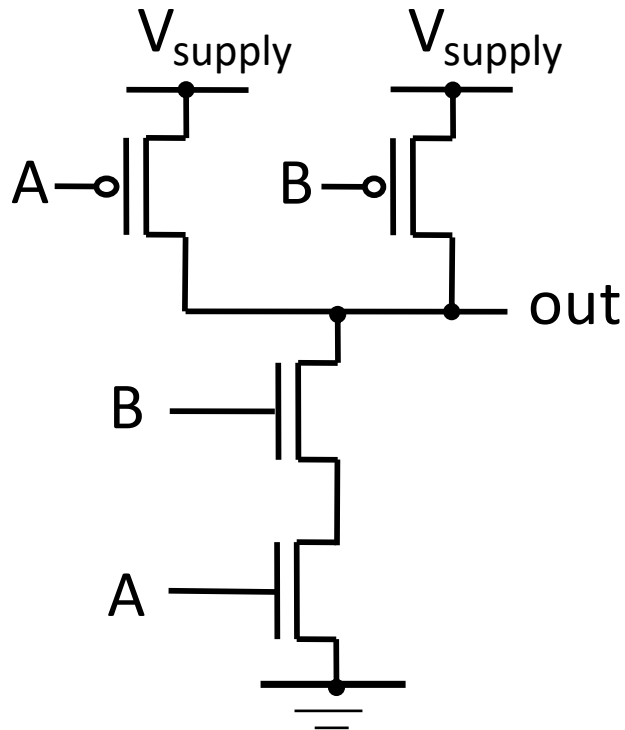


Truth Table:

A	B	out
0	0	1
0	1	0
1	0	0
1	1	0

# iClicker Question

Which Gate is this?








Function:

Symbol:

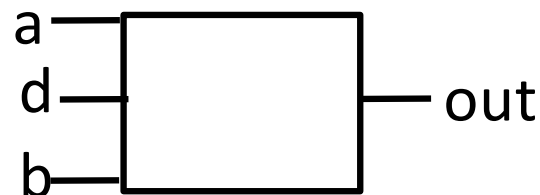
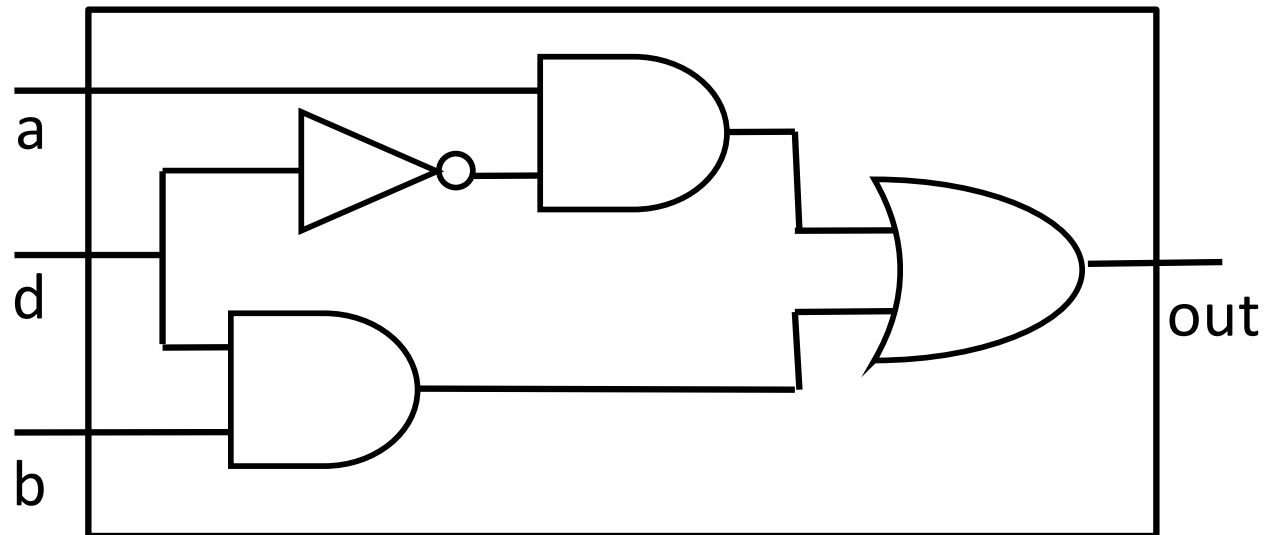
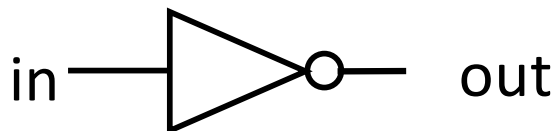
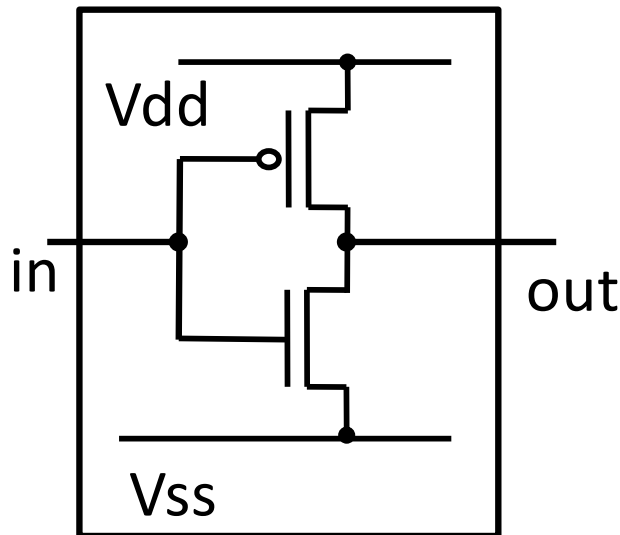
Truth Table:

A	B	out
0	0	
0	1	
1	0	
1	1	

- (A) NOT 
- (B) OR 
- (C) XOR 
- (D) AND 
- (E) NAND 

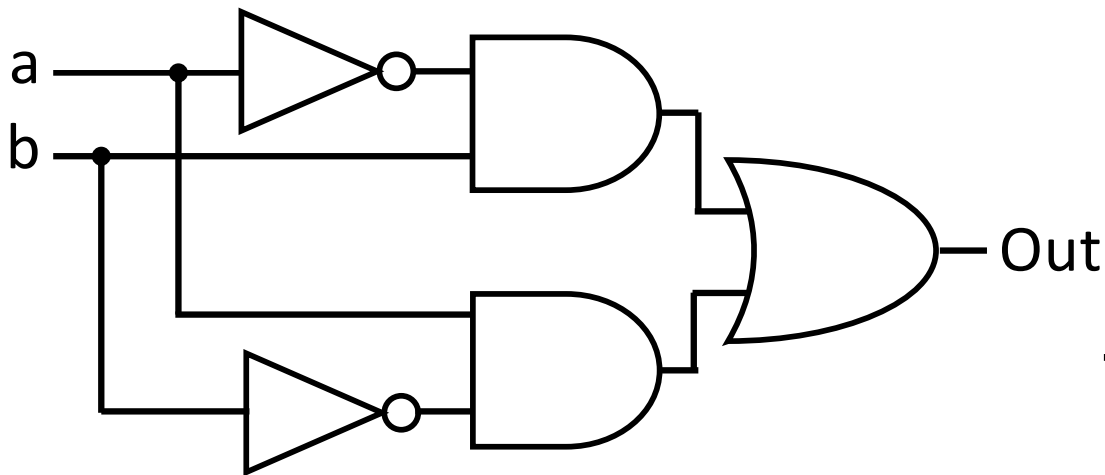
# Abstraction

- Hide complexity through simple abstractions
  - **Simplicity**
    - Box diagram represents inputs and outputs
  - **Complexity**
    - Hides underlying NMOS- and PMOS-transistors and atomic interactions



# iClicker Question

Which Gate is this?








Function:

Symbol:

Truth Table:

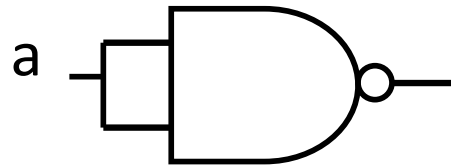
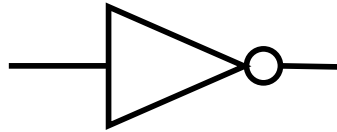
A	B	out
0	0	
0	1	
1	0	
1	1	

- (A) NOT 
- (B) OR 
- (C) XOR 
- (D) AND 
- (E) NAND 

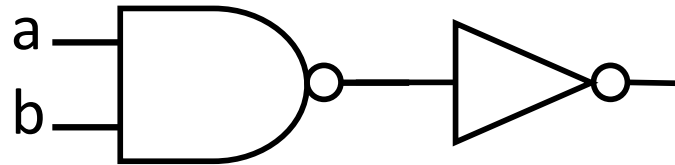
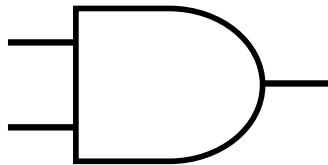
# Universal Gates

- NAND and NOR:
  - Can implement *any* function with NAND or just NOR gates
  - useful for manufacturing

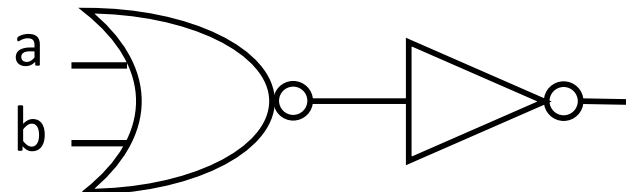
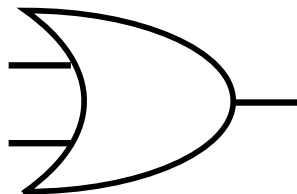
- NOT:



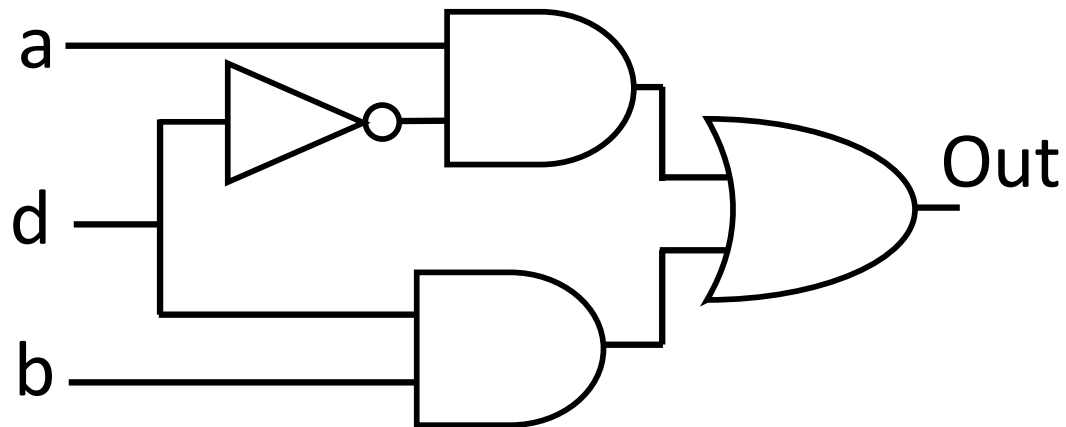
- AND:



- OR:



# What does this logic circuit do?



Function:

Symbol:

Truth Table:

a	b	d	Out
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	



Multiplexing Like a Boss



# Goals for Today

- From Switches to Logic Gates to Logic Circuits
- Transistors, Logic Gates, Truth Tables
- Logic Circuits
  - From Truth Tables to Circuits (Sum of Products)
  - **Identity Laws**
- Logic Circuit Minimization
  - Algebraic Manipulations
  - Karnaugh Maps

# Logic Implementation

How to implement a desired logic function?

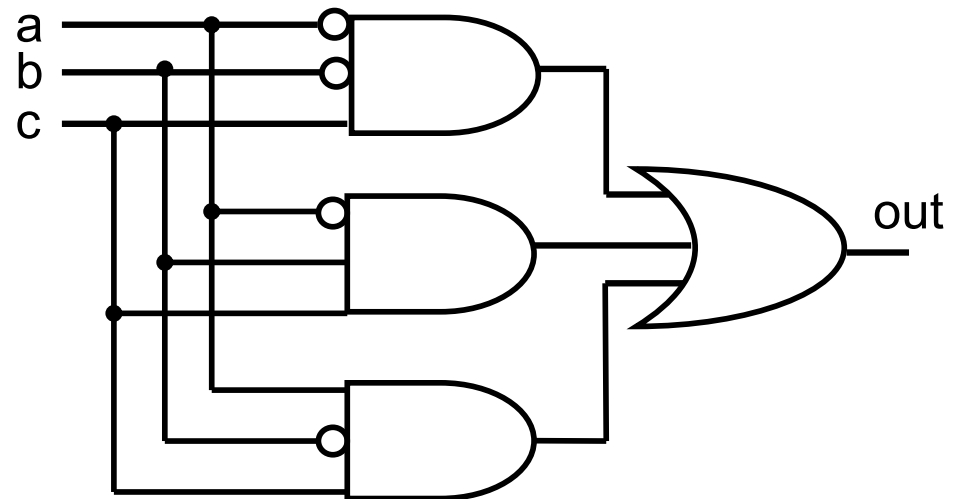
a	b	c	out	minterm
0	0	0	0	$\bar{a} \bar{b} \bar{c}$
0	0	1	1	$\bar{a} \bar{b} c$
0	1	0	0	$\bar{a} b \bar{c}$
0	1	1	1	$\bar{a} b c$
1	0	0	0	$a \bar{b} \bar{c}$
1	0	1	1	$a \bar{b} c$
1	1	0	0	$a b \bar{c}$
1	1	1	0	$a b c$

1) Write **minterms**

2) Write **sum of products**:

OR of all minterms where out=1

$$\text{out} = \bar{a}\bar{b}c + \bar{a}bc + a\bar{b}c$$



**Any** combinational circuit can be implemented in two levels of logic (ignoring inverters)

# Logic Equations

NOT:  $= \bar{a} \quad = !a \quad = \neg a$

AND:  $= a \cdot b \quad = a \& b \quad = a \wedge b$

OR:  $= a + b \quad = a | b \quad = a \vee b$

XOR:  $= a \oplus b = a\bar{b} + \bar{a}b$

NAND:

$$\overline{(a \cdot b)} = !(a \& b) = \neg (a \wedge b)$$

NOR:

$$\overline{(a + b)} = !(a | b) = \neg (a \vee b)$$

XNOR:

$$\overline{(a \oplus b)} = \overline{a\bar{b} + \bar{a}b}$$

## Logic Equations

- Constants: true = 1, false = 0
- Variables: a, b, out, ...
- Operators (above): AND, OR, NOT, etc.

# Identities

Identities useful for manipulating logic equations

- For optimization & ease of implementation

$$a + 0 = a$$

$$a + 1 = 1$$

$$a + \bar{a} = 1$$

$$a \cdot 0 = 0$$

$$a \cdot 1 = a$$

$$a \cdot \bar{a} = 0$$

# Identities

Identities useful for manipulating logic equations

- For optimization & ease of implementation

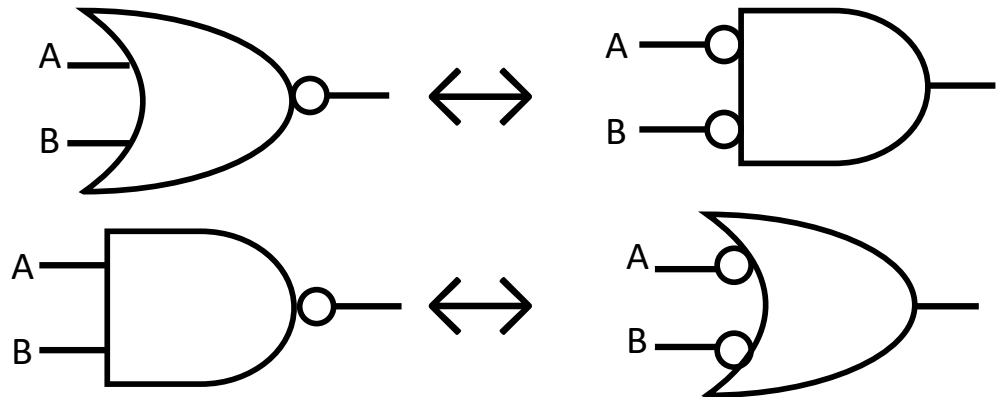
$$\overline{(a + b)} = \bar{a} \bullet \bar{b}$$

$$\overline{(ab)} = \bar{a} + \bar{b}$$

$$a + ab = a$$

$$a(b+c) = ab + ac$$

$$\overline{a(b + c)} = \bar{a} + \bar{b} \bullet \bar{c}$$



# Goals for Today

- From Switches to Logic Gates to Logic Circuits
- Transistors, Logic Gates, Truth Tables
- Logic Circuits
  - Identity Laws
  - From Truth Tables to Circuits (Sum of Products)
- **Logic Circuit Minimization**
  - *No details this semester, but quick discussion of what this is and why you'd do it.*

# Checking Equality w/Truth Tables

circuits  $\leftrightarrow$  truth tables  $\leftrightarrow$  equations

Example:  $(a+b)(a+c) = a + bc$

a	b	c	(a+b)	LHS	(a+c)	RHS	bc
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

# Checking Equality w/Truth Tables

circuits  $\leftrightarrow$  truth tables  $\leftrightarrow$  equations

Example:  $(a+b)(a+c) = a + bc$

a	b	c	$(a+b)$	LHS	$(a+c)$	RHS	$bc$
0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0
0	1	0	1	0	0	0	0
0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0
1	0	1	1	1	1	1	0
1	1	0	1	1	1	1	0
1	1	1	1	1	1	1	1



# Summary

- Most modern devices made of billions of transistors
  - You will build a processor in this course!
  - Modern transistors made from semiconductor materials
  - Transistors used to make logic gates and logic circuits
- We can now implement any logic circuit
  - Use P- & N-transistors to implement NAND/NOR gates
  - Use NAND or NOR gates to implement the logic circuit
  - *Efficiently*: use K-maps to find required minimal terms