Digital Logic Design

Hajar Falahati

Department of Computer Engineering
IRAN University of Science and Technology

hfalahati@iust.ac.ir

Switching Functions

- Switching algebra
  - Boolean algebra with the set of elements $K = \{0, 1\}$
  - If there are n variables, we can define 2^{2^n} switching functions.

AB	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
00	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
01	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
10	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
11	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

- A switching expression as follows:
  - $f_0(A,B) = 0$, $f_6(A,B) = AB' + A'B$, $f_{11}(A,B) = AB + A'B + A'B' = A' + B$, ...

Some Definition: 1

- **Literal**
  - A variable, complemented or uncomplemented
  - A , $\overline{A}$, B , $\overline{B}$
- **Product term**
  - A literal or literals ANDed together
  - $(A \cdot B \cdot \overline{C})$, $(\overline{A} \cdot C)$, $(B \cdot \overline{C})$
- **Sum term**
  - A literal or literals ORed together.
  - $(A + B + \overline{C})$, $(\overline{A} + C)$, $(B + \overline{C})$
- **Minterm**
  - A product that includes all the variables
  - $(A \cdot B \cdot \overline{C})$, $(\overline{A} \cdot \overline{B} \cdot C)$, $(\overline{A} \cdot B \cdot \overline{C})$
- **Maxterm**
  - A sum that includes all the variables
  - $(A + B + \overline{C})$, $(\overline{A} + \overline{B} + C)$, $(\overline{A} + B + \overline{C})$

Algebraic Forms

- SOP
  - Sum of Products
  - ORing product terms
  - f(A, B, C) = ABC + A'C + B’C
- POS
  - Product of Sums
  - ANDing sum terms
  - f (A, B, C) = (A' + B' + C')(A + C')(B + C’)
- Canonical SOP
  - Represented as a sum of minterms only.
  - Example: $f_1(A,B,C) = A'BC' + ABC' + A'BC + ABC$
- **Canonical POS**
  - Represented as a product of **maxterms** only.
  - Example: $f_2(A,B,C) = (A+B+C)(A+B+C')(A'+B+C)(A'+B+C')$
  -

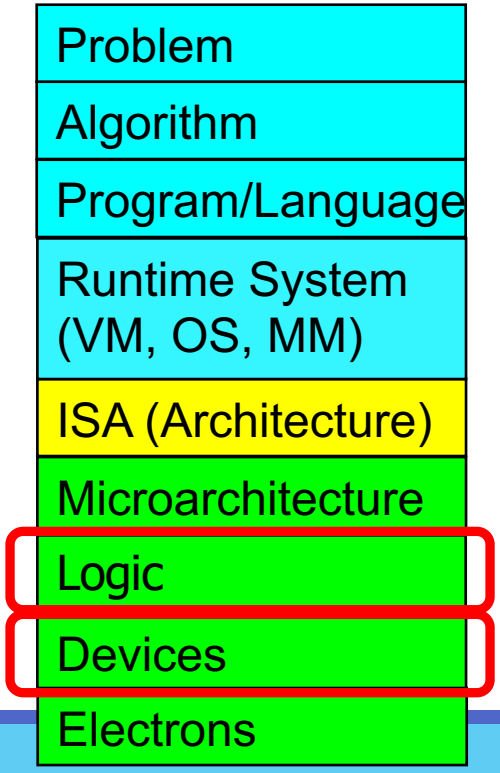
Outline

- Logic Gates

Transistors

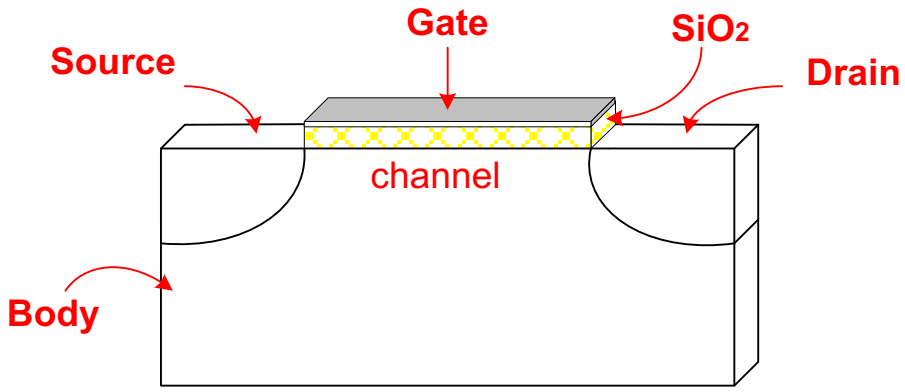
Transistor

- **Computers are built from very large numbers of very simple structures**
  - Intel's Pentium IV microprocessor, first offered for sale in 2000, was made up of more than 42 million MOS transistors
  - Intel's Core i7 Broadwell-E, offered for sale in 2016, is made up of more than 3.2 billion MOS transistors



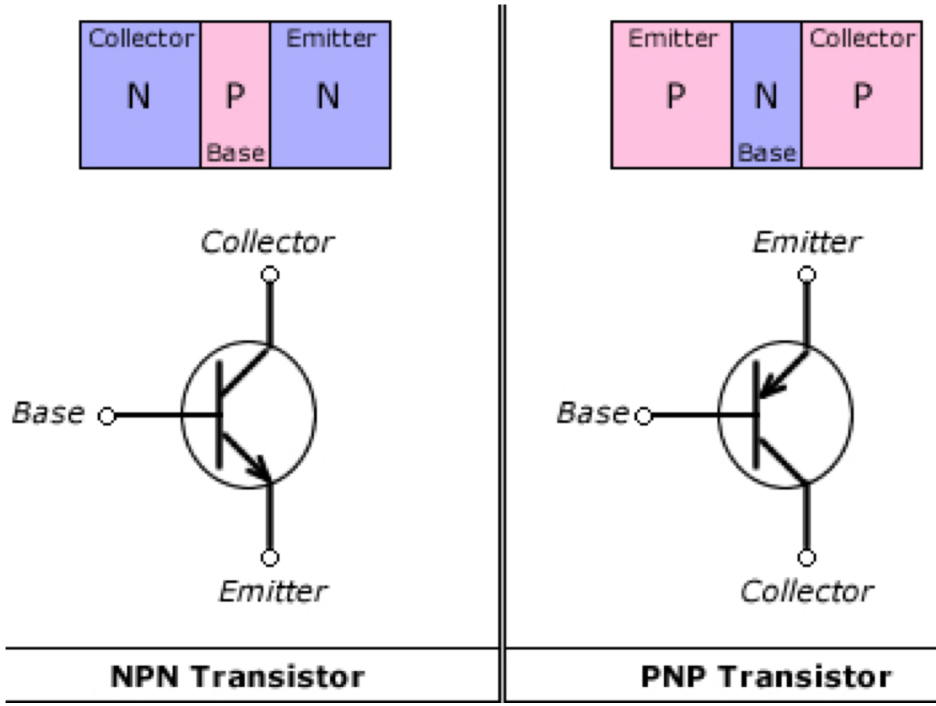
Transistor Structure

- Structure of a typical transistor
- A transistor conducts when the channel is filled with carriers
  - Negative carriers (free electrons)
  - Positive carriers (holes electrons)
- Why is this useful?
  - We can combine many of these to realize simple logic gates



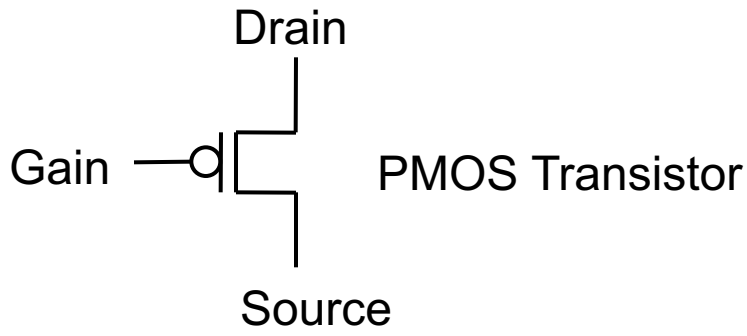
BJT Transistor

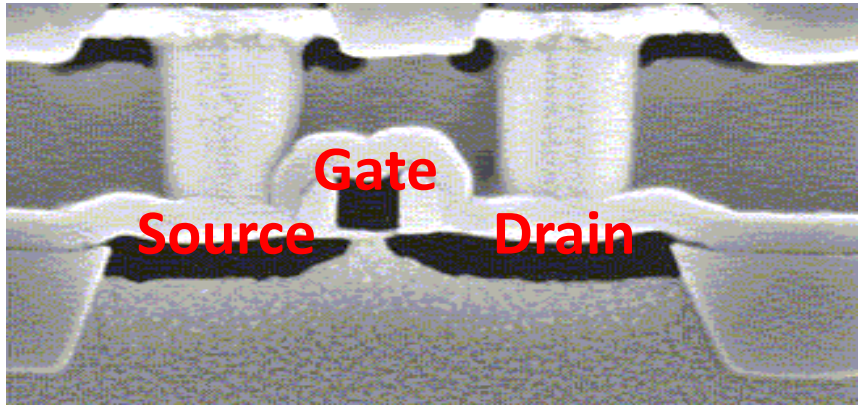
- Bipolar Junction Transistor (BJT)

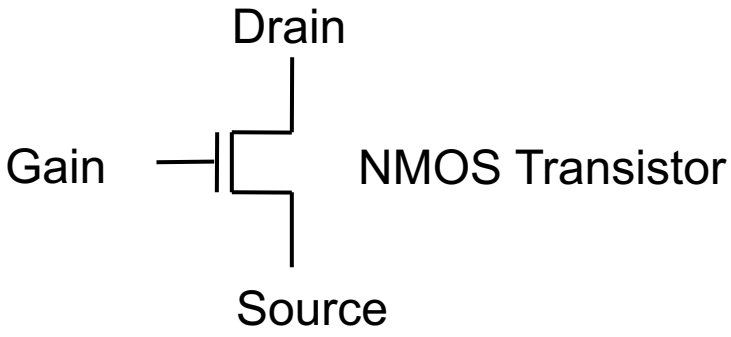


MOS Transistor

- Field Effect Transistors (FET)
  - PMOS: if Gain =0 → electron flow from drain to source
  - NMOS: if Gain =1 → electron flow from drain to source

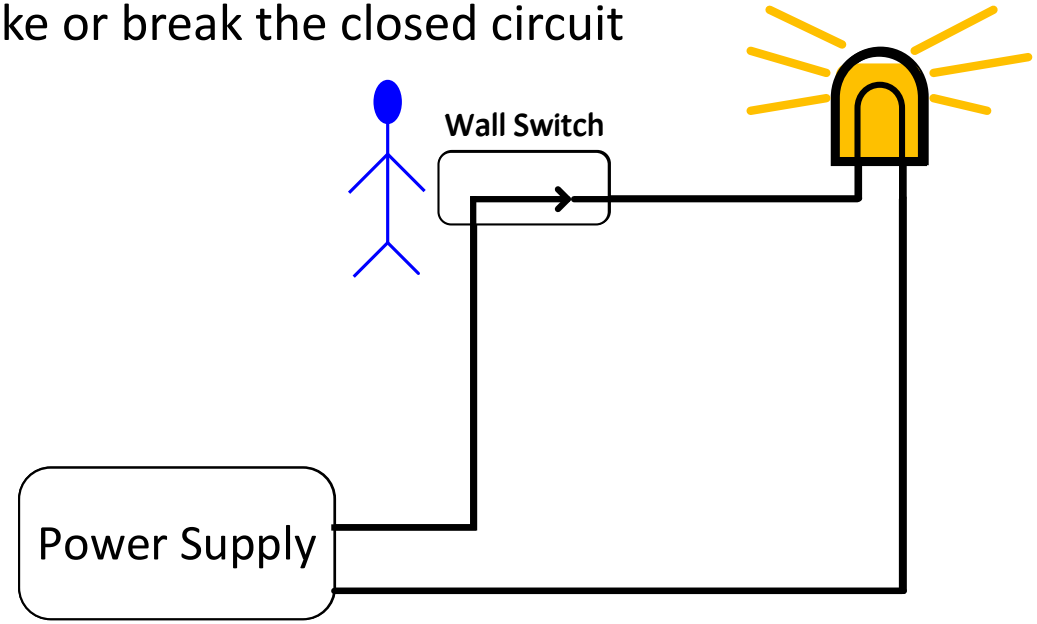






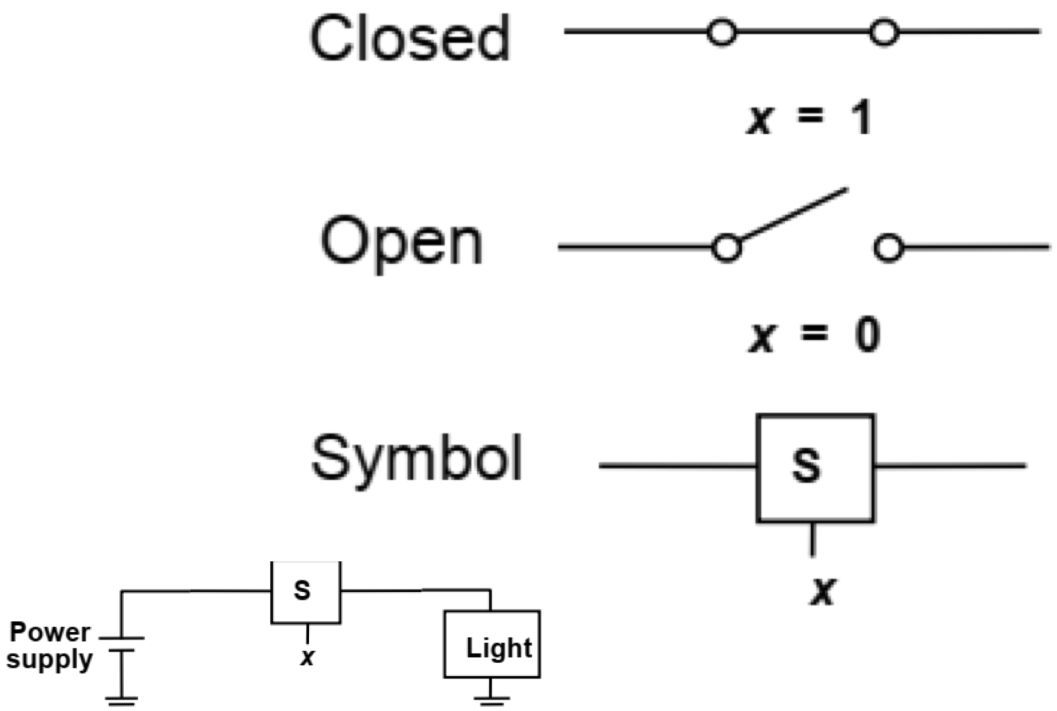
How Does a Transistor Work?

- In order for the lamp to glow, electrons must flow
- In order for electrons to flow, there must be a closed circuit from the power supply to the lamp and back to the power supply
- The lamp can be turned on and off by simply manipulating the wall switch to make or break the closed circuit

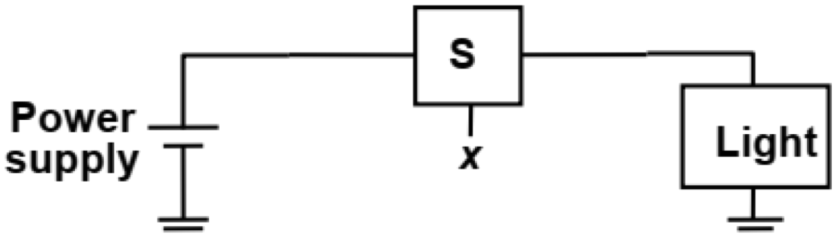


Switches

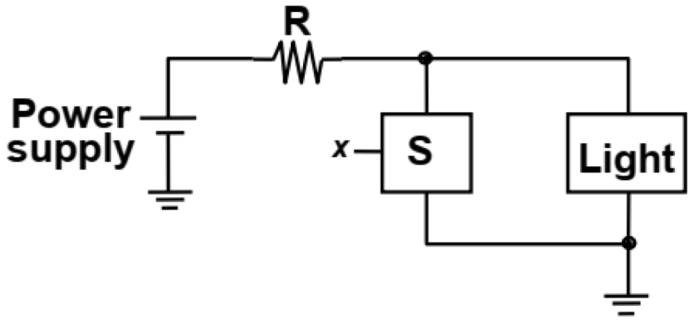
- A switch has two states
  - Closed/ On
  - Open/ OFF



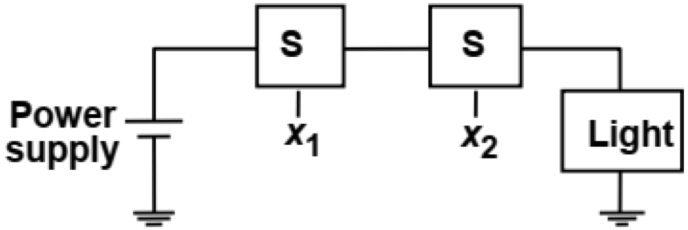
Switches Samples



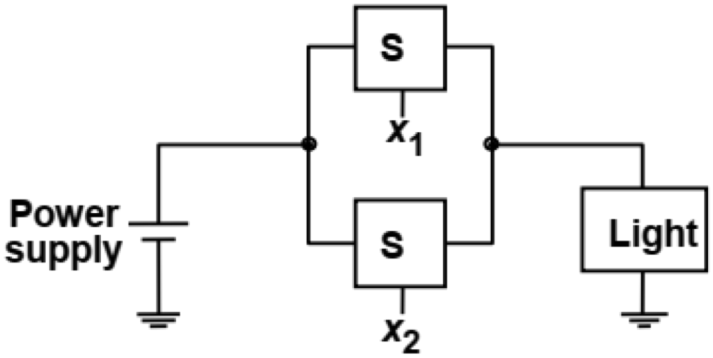
$$L(x) = \begin{cases} 0 & \text{Light Off} \\ 1 & \text{Light On} \end{cases}$$



$$L(x) = \bar{x}$$



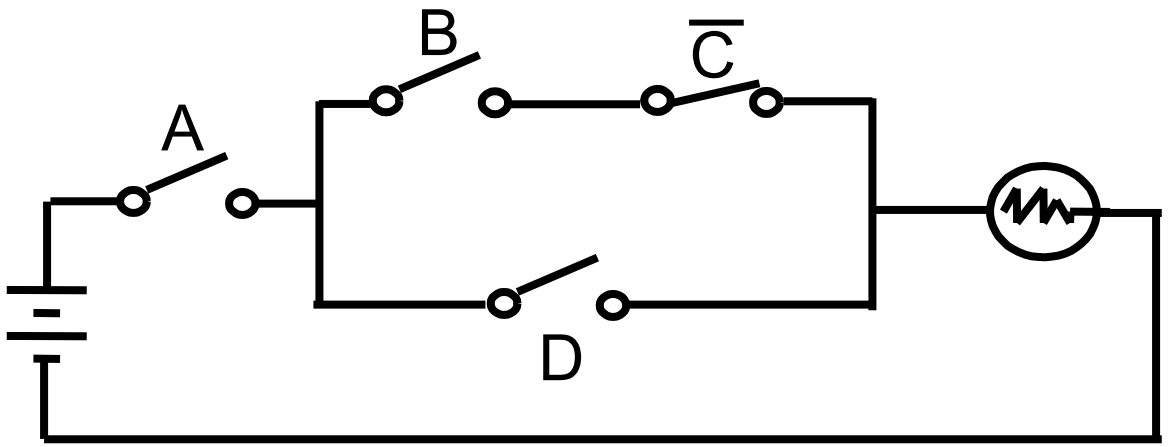
$$L(x_2, x_1) = x_1 \cdot x_2$$

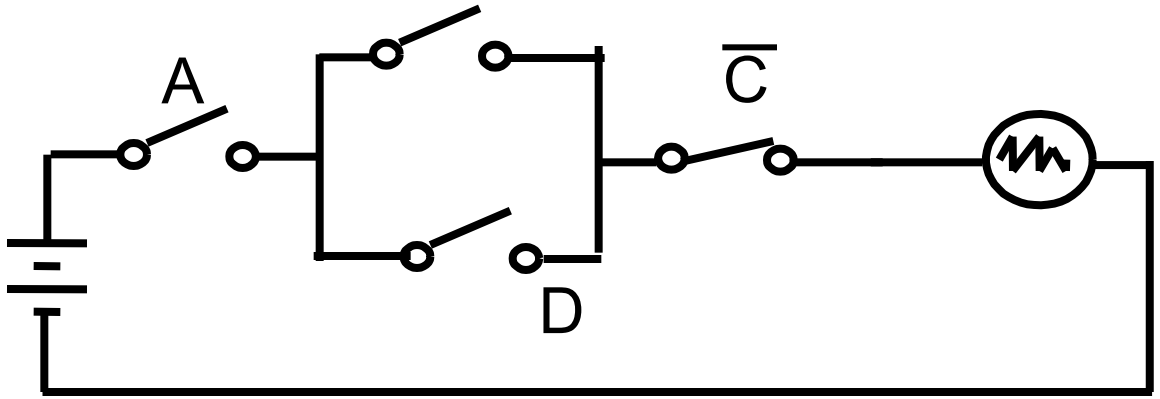


$$L(x_2, x_1) = x_1 + x_2$$

Switches: Sample 1

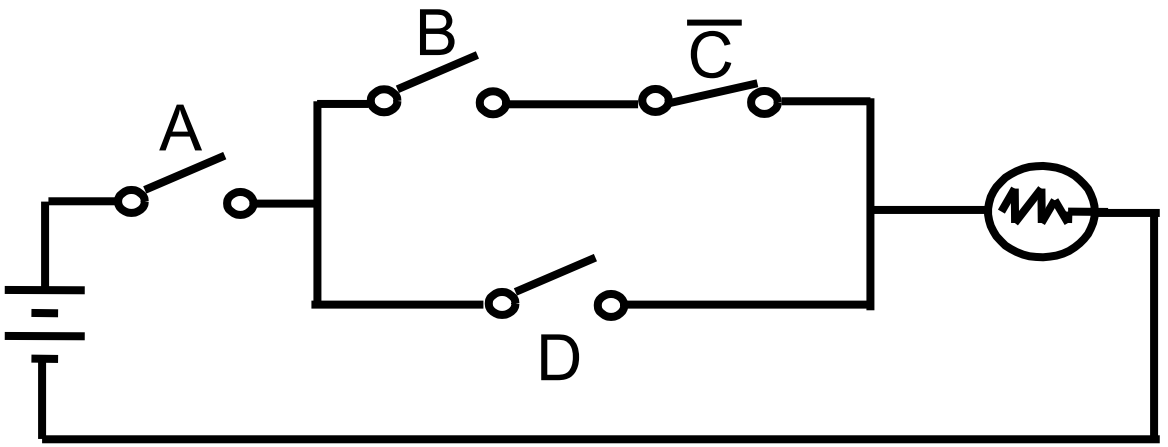
- Find L(A,B,C,D)



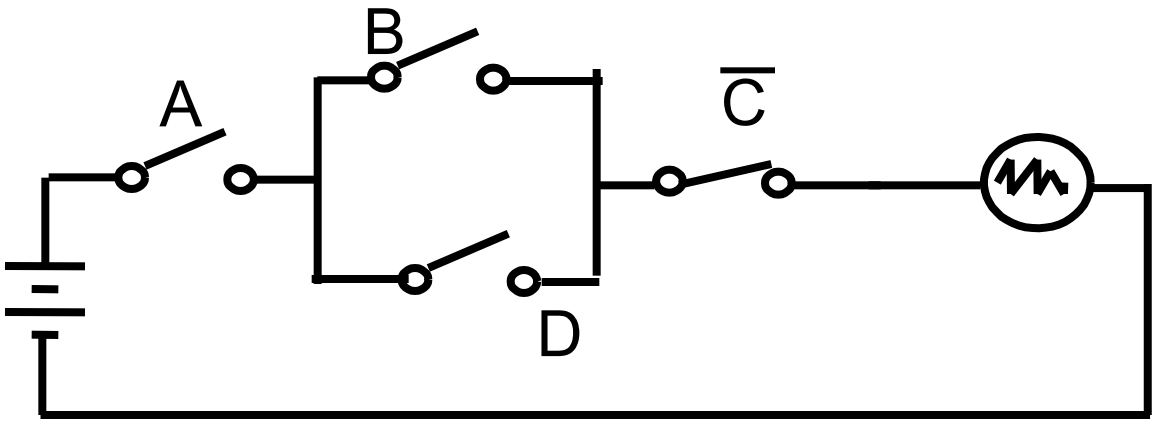


Switches: Sample1 (cont'd)

- Find L(A,B,C,D)
  - A (B $\bar{C}$ + D)
  - A B $\bar{C}$ + A D

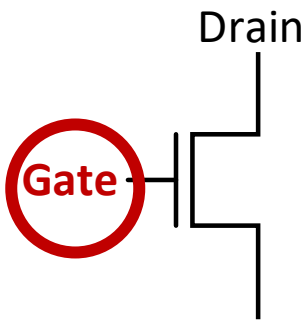


  - A (B + D) $\bar{C}$



Switching and Transistors

- Instead of the wall switch, we could use an n-type or a p-type MOS transistor to make or break the closed circuit



Schematic of an n-type MOS transistor

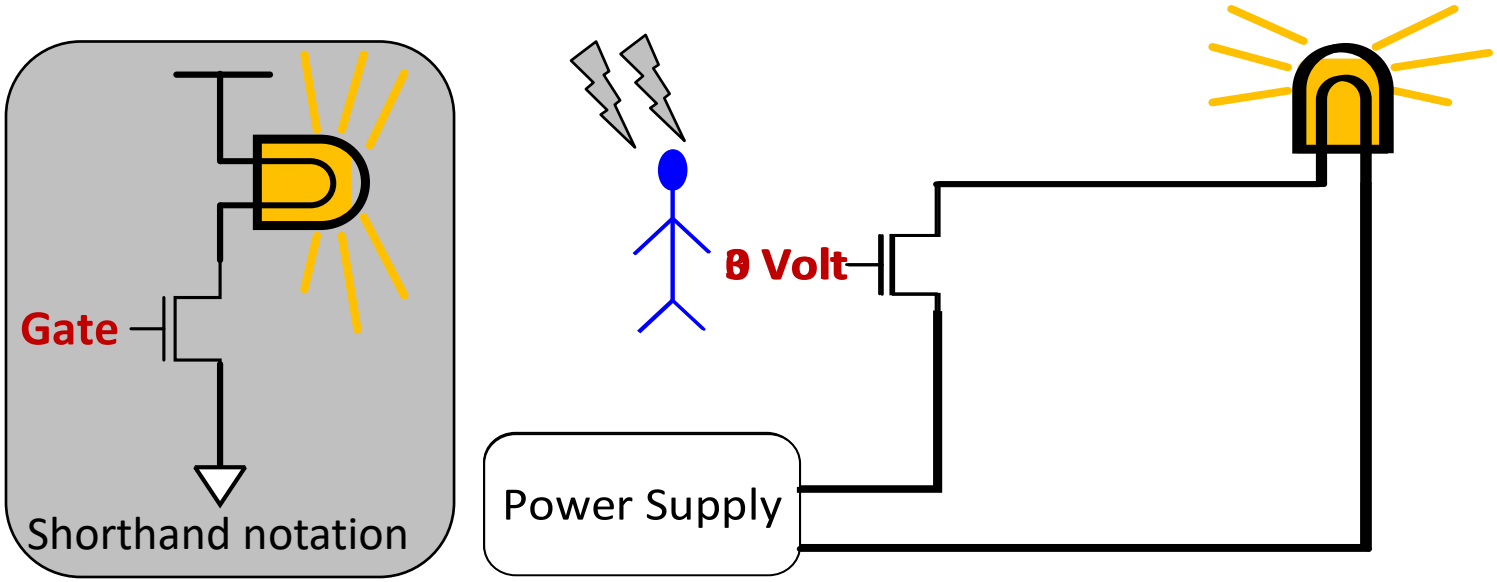
If the gate of an n-type transistor is supplied with a **high** voltage, the connection from source to drain acts like a piece of wire

Depending on the technology, 0.3V to 3V

If the gate of the -type transistor is supplied with 0V, the connection between the source and drain is broken

How Does a Transistor Work?

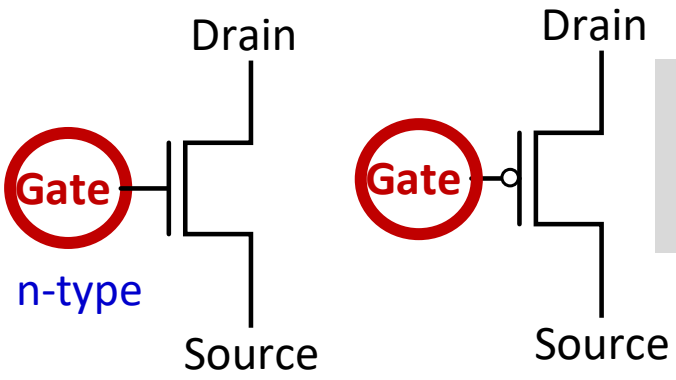
- The n-type transistor in a circuit with a battery and a bulb



How Does a Transistor Work?

- The p-type transistor works in exactly the opposite fashion from the n-type transistor

The circuit is closed when the gate is supplied with 3V

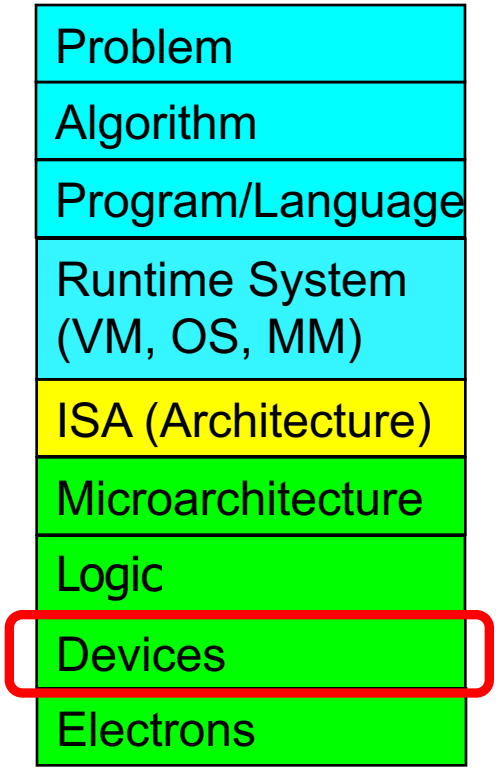


The circuit is closed when the gate is supplied with 0V

Logic gates

One Level Higher in Abstraction

- How do we build logic out of MOS transistors?
- We construct basic logic structures out of individual MOS transistors
- These logical units are named logic gates
  - They implement simple Boolean functions

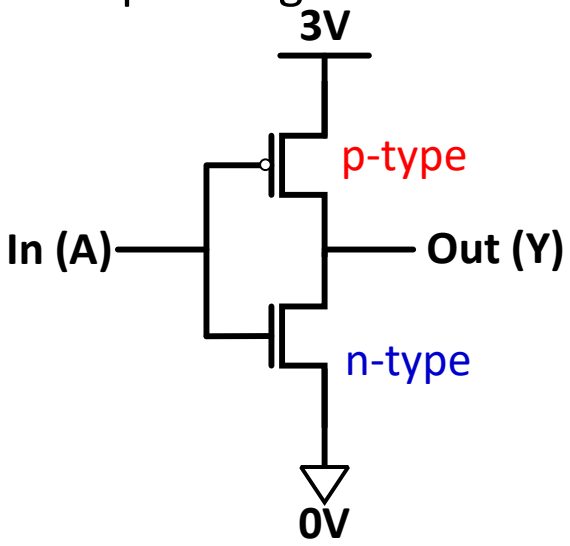


Making Logic Blocks Using CMOS Technology

- Modern computers use both n-type and p-type transistors, i.e. Complementary MOS (CMOS) technology

nMOS + pMOS = CMOS

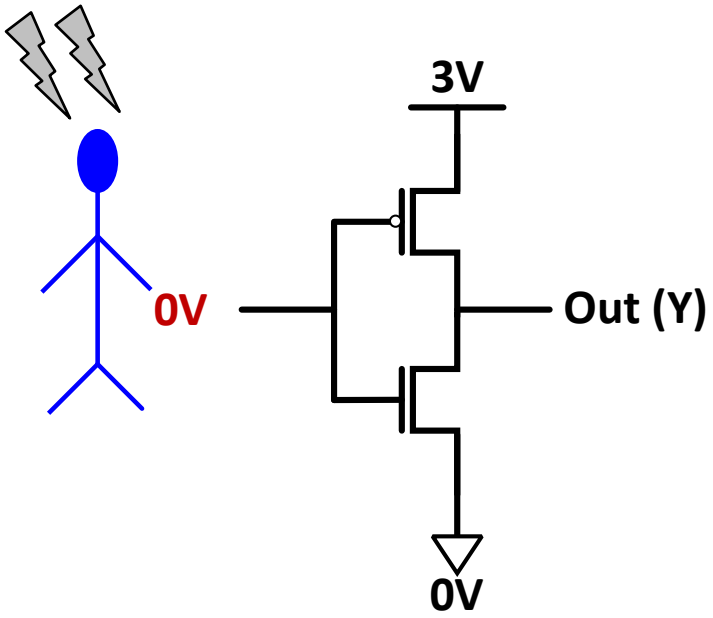
- The simplest logic structure that exists in a modern computer

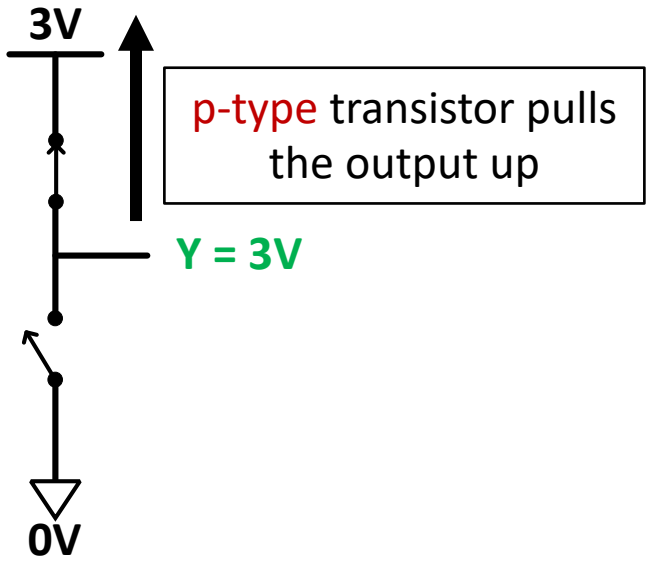


What does this circuit do?

Functionality of Our CMOS Circuit

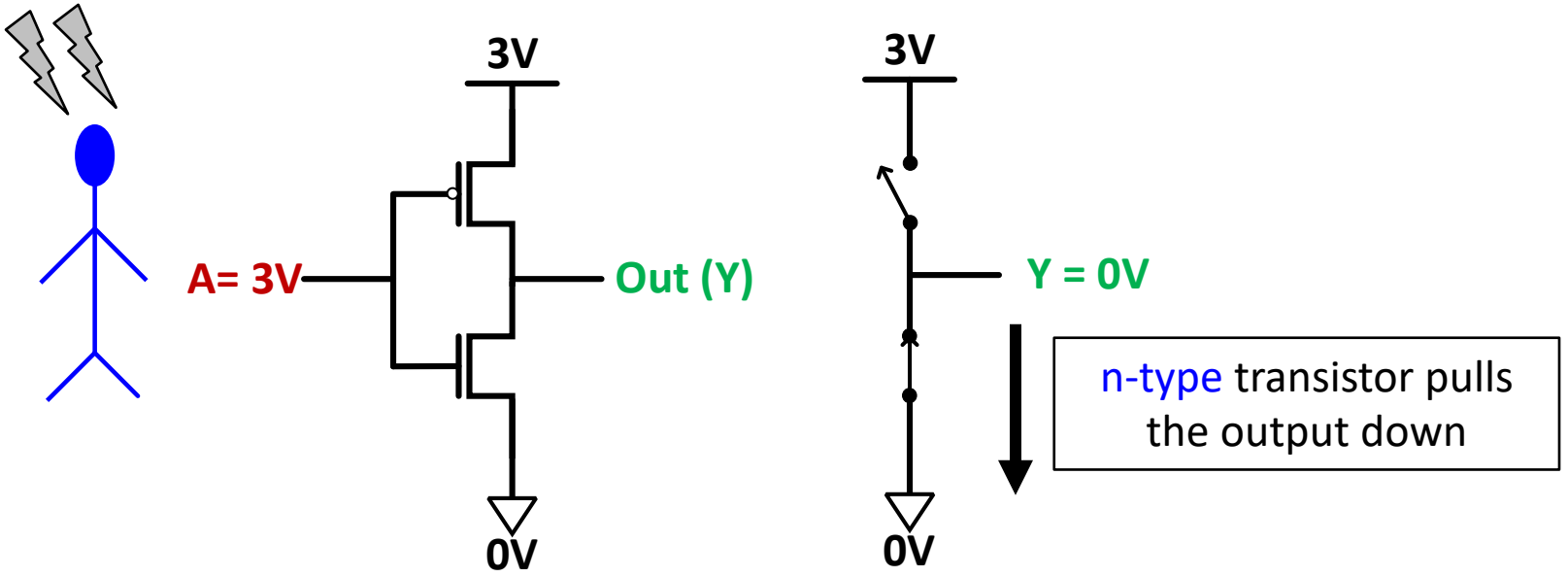
What happens when the input is connected to 0V?





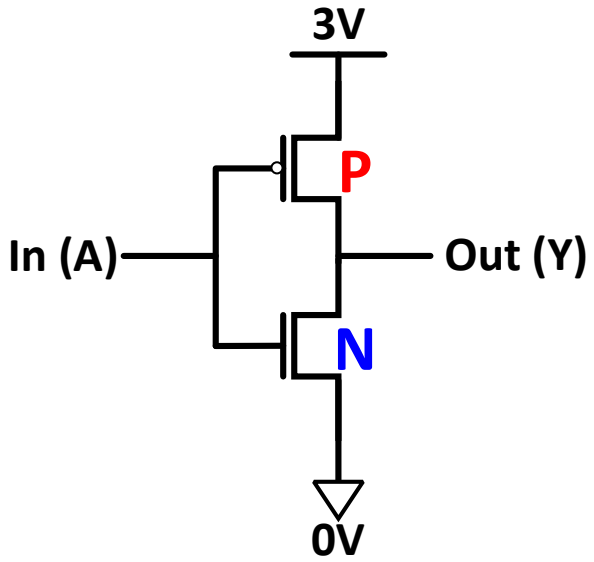
Functionality of Our CMOS Circuit

What happens when the input is connected to 3V?



CMOS NOT Gate

- This is actually the CMOS NOT Gate
- Why do we call it NOT?
  - If A = 0V then Y = 3V
  - If A = 3V then Y = 0V
- **Digital circuit:** one possible interpretation
  - Interpret 0V as logical (binary) 0 value
  - Interpret 3V as logical (binary) 1 value

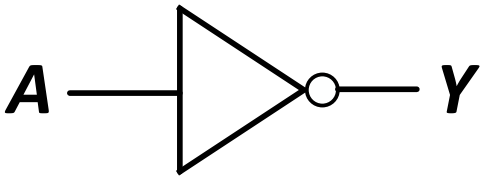


A	P	N	Y
0	ON	OFF	1
1	OFF	ON	0

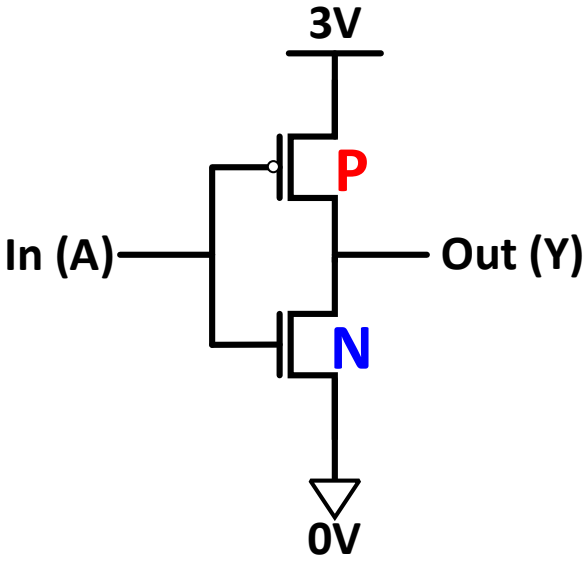
$$Y = \bar{A}$$

CMOS NOT Gate

- We call it a NOT gate



$$Y = \bar{A}$$



- Truth Table
  - What would be the logical output of the circuit for each possible input

A	Y
0	1
1	0

Logic Gates: NOT

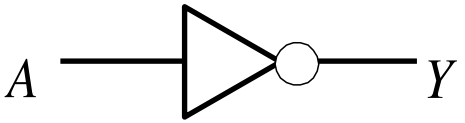
- (a) NOT logic function.
- (b) Electronic NOT gate.
- (c) Standard symbol.
- (d) IEEE block symbol.

a	$fNOT\ (a) = \bar{a}$
0	1
1	0

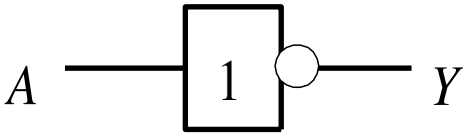
(a)

A	Y
L	H
H	L

(b)



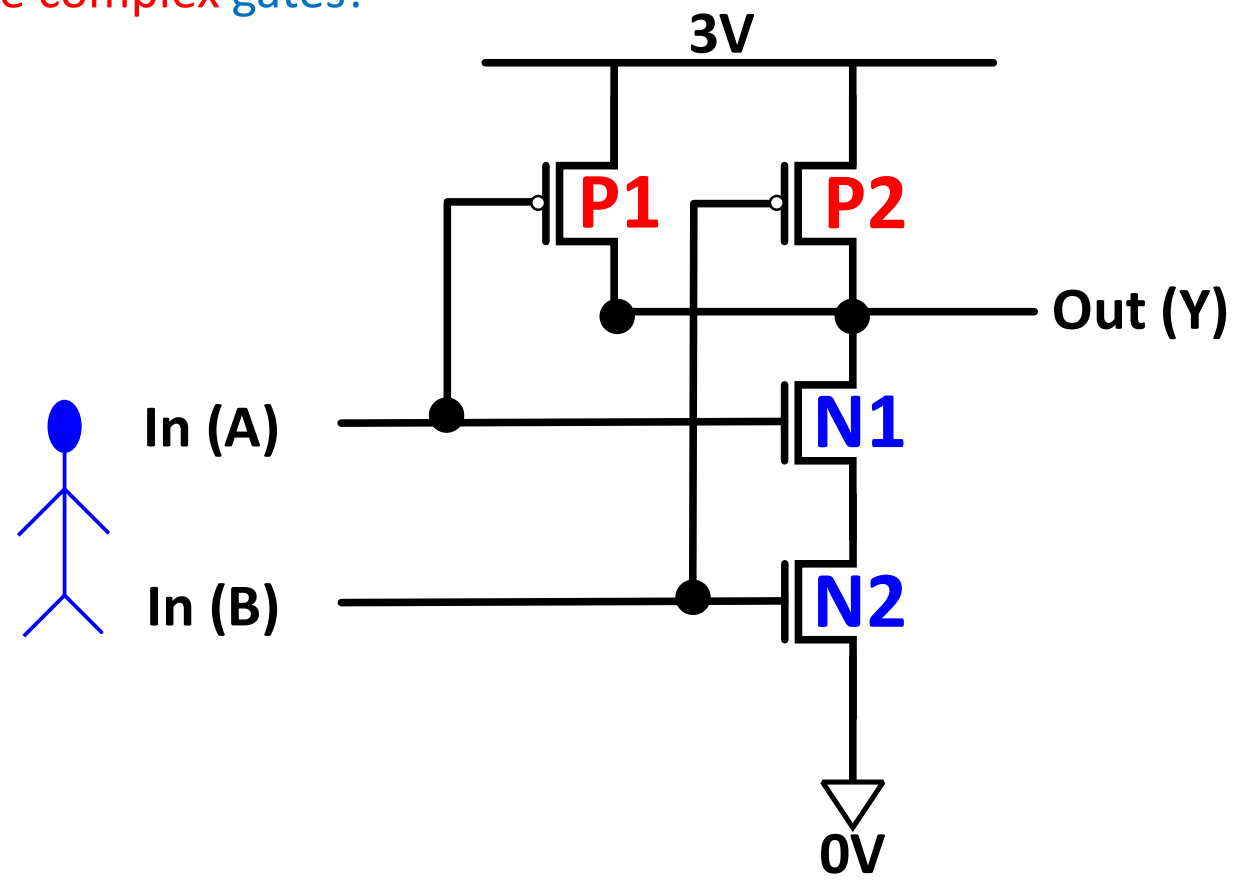
(c)



(d)

Another CMOS Gate: What Is This?

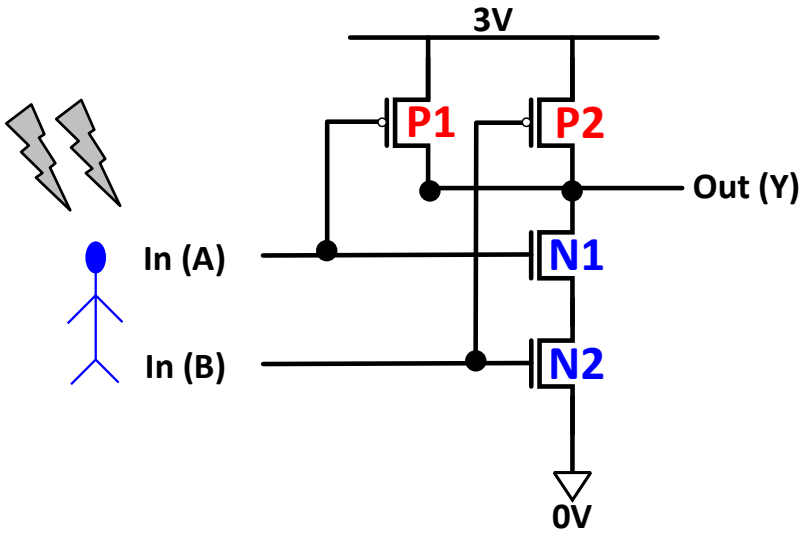
- Let's build more complex gates!



CMOS NAND Gate

- Let's build more complex gates!

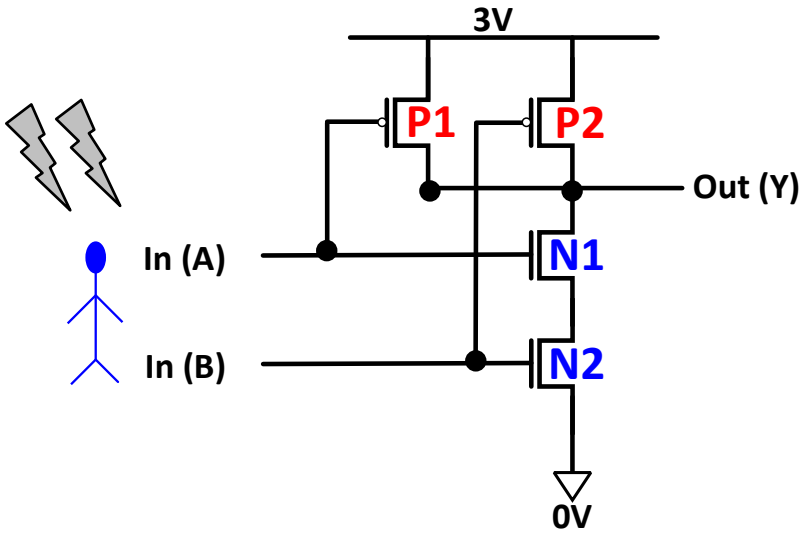
$$Y = \overline{A \cdot B} = \overline{AB}$$



A	B	P1	P2	N1	N2	Y
0	0					
0	1					
1	0					
1	1					

CMOS NAND Gate

- Let's build more complex gates!



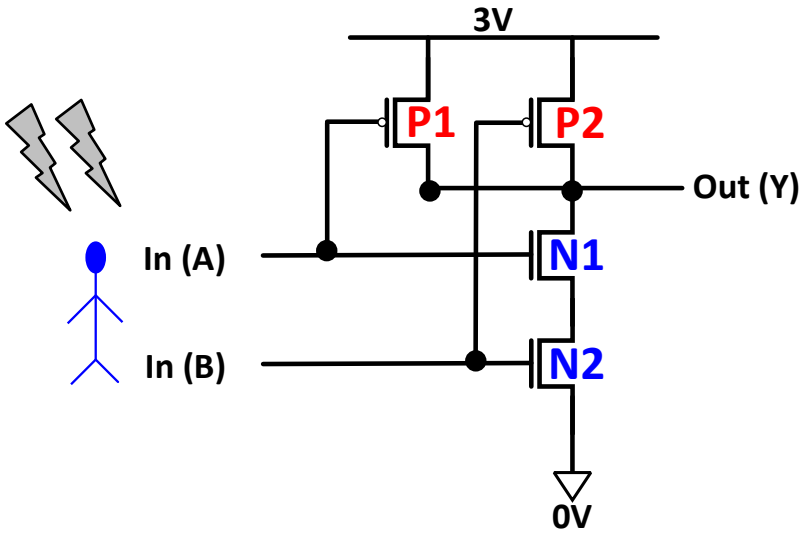
$$Y = \overline{A \cdot B} = \overline{AB}$$

A	B	P1	P2	N1	N2	Y
0	0	ON	ON	OFF	OFF	1
0	1					
1	0					
1	1					

- P1 and P2 are in parallel; only one must be ON to pull the output up to 3V

CMOS NAND Gate

- Let's build more complex gates!



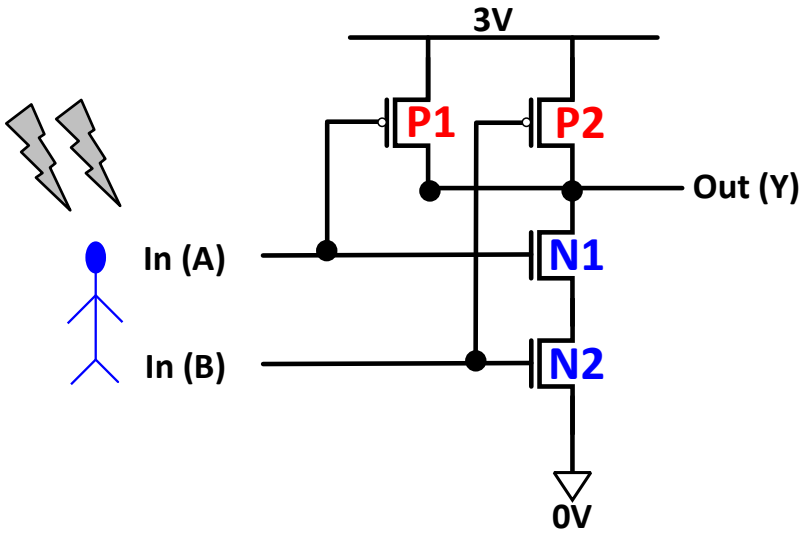
$$Y = \overline{A \cdot B} = \overline{AB}$$

A	B	P1	P2	N1	N2	Y
0	0	ON	ON	OFF	OFF	1
0	1	ON	OFF	OFF	ON	1
1	0					
1	1					

- P1 and P2 are in parallel; only one must be ON to pull the output up to 3V

CMOS NAND Gate

- Let's build more complex gates!



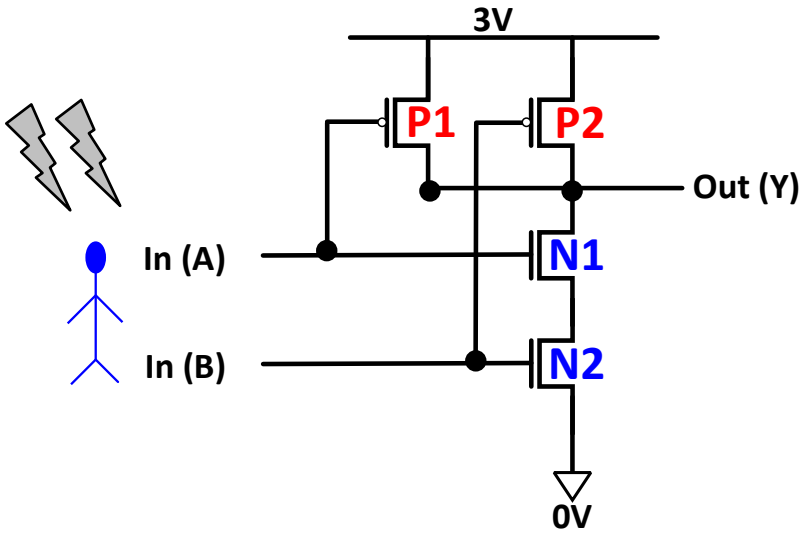
$$Y = \overline{A \cdot B} = \overline{AB}$$

A	B	P1	P2	N1	N2	Y
0	0	ON	ON	OFF	OFF	1
0	1	ON	OFF	OFF	ON	1
1	0	OFF	ON	ON	OFF	1
1	1					

- P1 and P2 are in parallel; only one must be ON to pull the output up to 3V
- N1 and N2 are connected in series; both must be ON to pull the output to 0V

CMOS NAND Gate

- Let's build more complex gates!



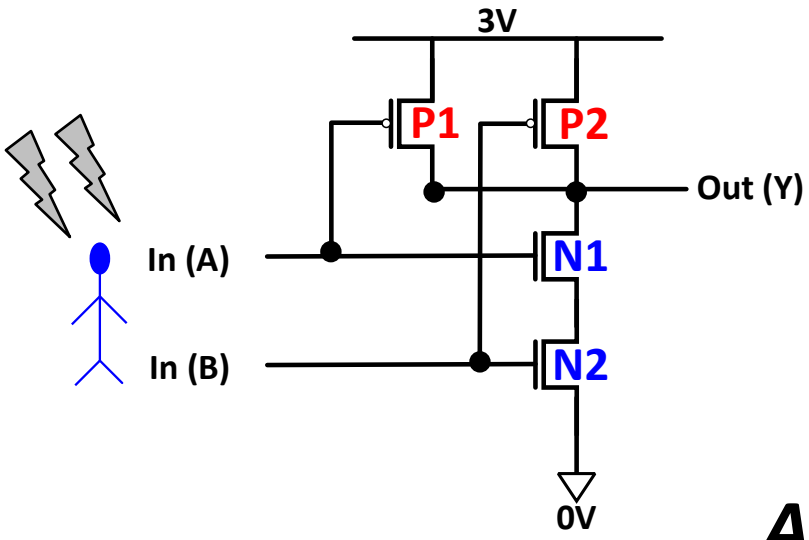
$$Y = \overline{A \cdot B} = \overline{AB}$$

A	B	P1	P2	N1	N2	Y
0	0	ON	ON	OFF	OFF	1
0	1	ON	OFF	OFF	ON	1
1	0	OFF	ON	ON	OFF	1
1	1	OFF	OFF	ON	ON	0

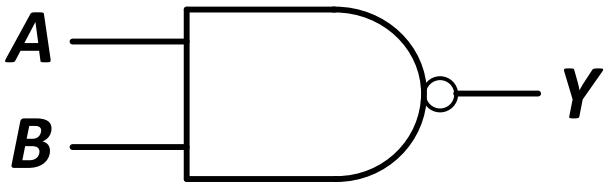
- P1 and P2 are in parallel; only one must be ON to pull the output up to 3V
- N1 and N2 are connected in series; both must be ON to pull the output to 0V

CMOS NAND Gate

- Let's build more complex gates!



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



A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Logic Gates: NAND

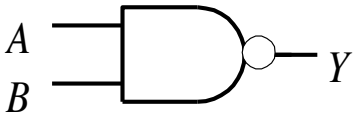
- (a) NAND logic function
- (b) Electronic NAND gate
- (c) Standard symbol
- (d) IEEE block symbol

a	b	$fNAND\ (a, b) = \overline{ab}$
0	0	1
0	1	1
1	0	1
1	1	0

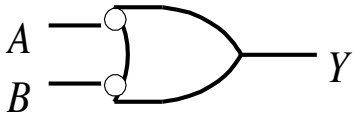
(a)

A	B	Y
L	L	H
L	H	H
H	L	H
H	H	L

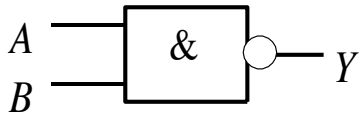
(b)



(c)



(d)

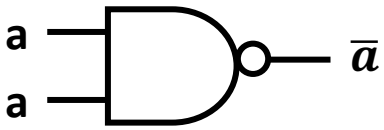


(e)

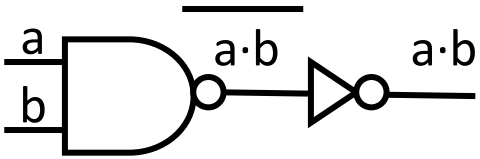
$$\overline{AB} = \overline{A} + \overline{B}$$

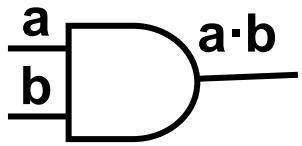
Logic Gates: NAND: Properties

$$f_{NAND}(a,a) = \overline{a \cdot a} = \overline{a} = f_{NOT}(a)$$

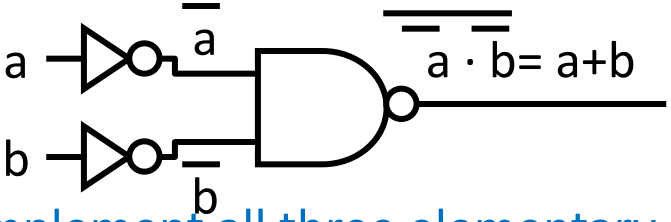


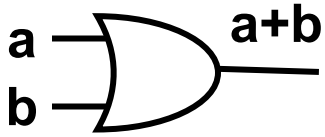
$$f_{NAND}(\overline{a},\overline{b}) = \overline{\overline{a} \cdot \overline{b}} = a + b = f_{OR}(a,b)$$





$$\bar{f}_{NAND}(a,b) = \overline{\overline{a \cdot b}} = a \cdot b = f_{AND}(a,b)$$

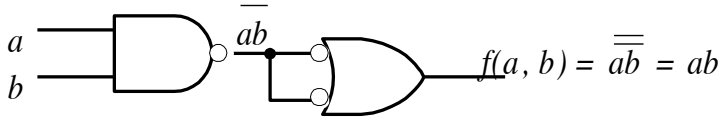




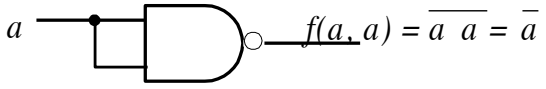
- NAND gate may be used to implement all three elementary operators.

NAND: Universal Gate

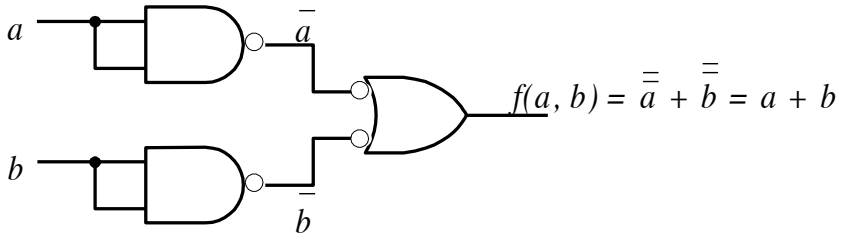
- Universal gate
  - A gate type that can implement any Boolean function.
  - NAND is a universal gate



AND gate



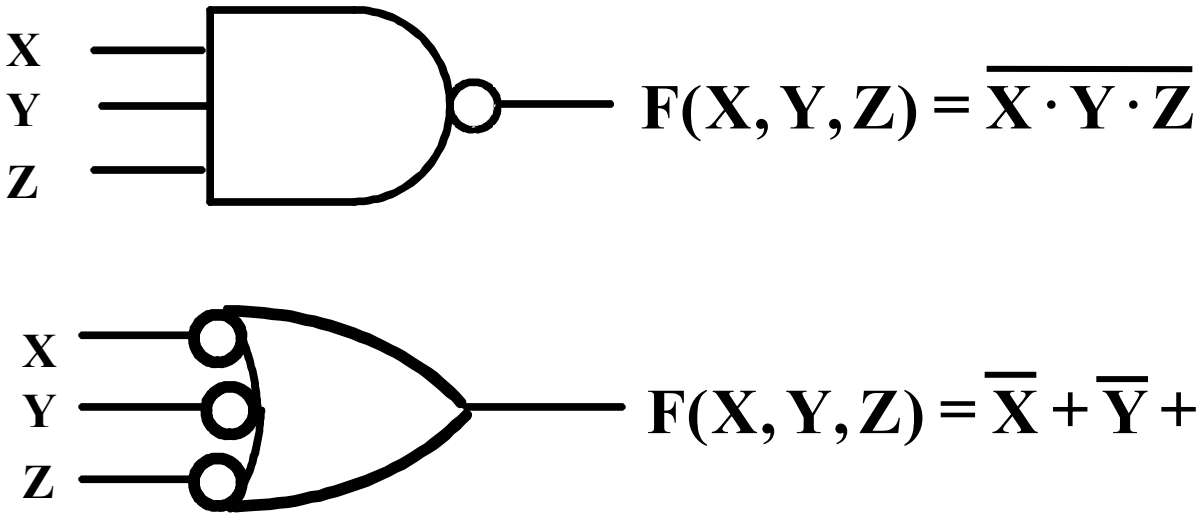
NOT gate



OR gate

Logic Gates: NAND (cont'd)

- NAND operation is NOT associative
- (X NAND Y) NAND Z ≠ X NAND (Y NAND Z)



CMOS AND Gate

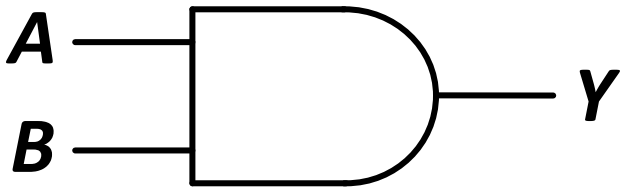
- How can we make a AND gate?
- Let's check NAND gate

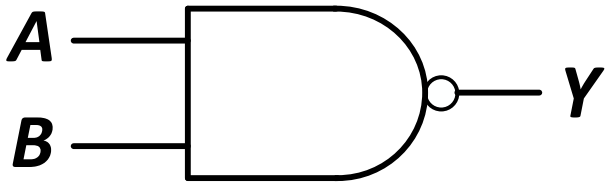
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

$$Y = A \cdot B = AB$$

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

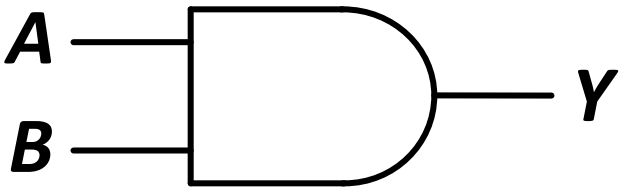




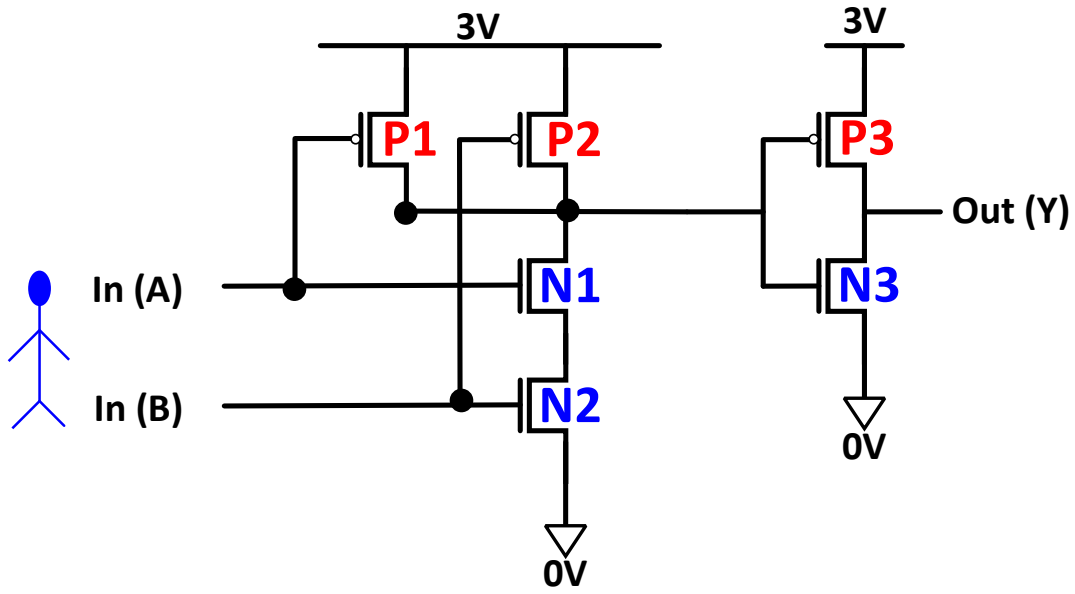
CMOS AND Gate

- How can we make a AND gate?

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



$$Y = A \cdot B = AB$$



We make an AND gate using one NAND gate and one NOT gate

Logic Gates: AND

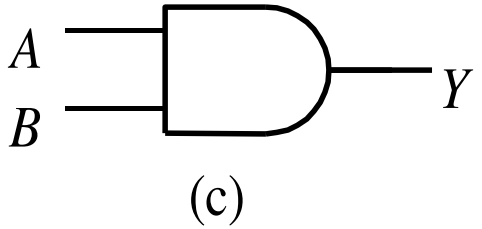
- (a) AND logic function.
- (b) Electronic AND gate.
- (c) Standard symbol.
- (d) IEEE block symbol.

a	b	$f_{AND}(a, b) = ab$
0	0	0
0	1	0
1	0	0
1	1	1

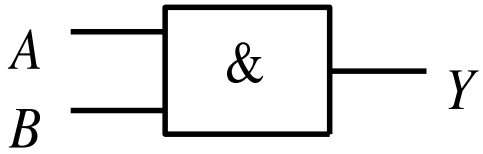
(a)

A	B	Y
L	L	L
L	H	L
H	L	L
H	H	H

(b)



(c)



(d)

Logic Gates: OR

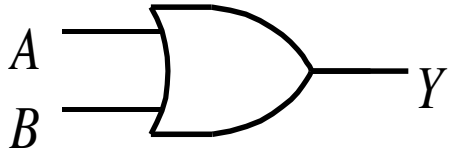
- (a) OR logic function.
- (b) Electronic OR gate.
- (c) Standard symbol.
- (d) IEEE block symbol.

a	b	$f_{OR}(a, b) = a + b$
0	0	0
0	1	1
1	0	1
1	1	1

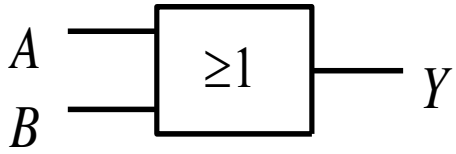
(a)

A	B	Y
L	L	L
L	H	H
H	L	H
H	H	H

(b)



(c)



(d)

Logic Gates: NOR

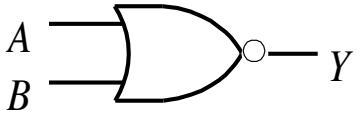
- (a) NAND logic function
- (b) Electronic NAND gate
- (c) Standard symbol
- (d) IEEE block symbol

a	b	$f_{NOR}(a, b) = \overline{a + b}$
0	0	1
0	1	0
1	0	0
1	1	0

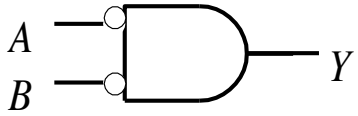
(a)

A	B	Y
L	L	H
L	H	L
H	L	L
H	H	L

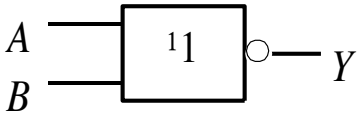
(b)



(c)



(d)



(e)

$$\overline{\boldsymbol{A} + \boldsymbol{B}} = \overline{\boldsymbol{A}}.\overline{\boldsymbol{B}}$$

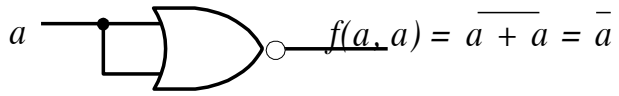
Logic Gates: NOR: Properties

- **NOR gate may be used to implement all three elementary operators**
- **NOR is universal gate**

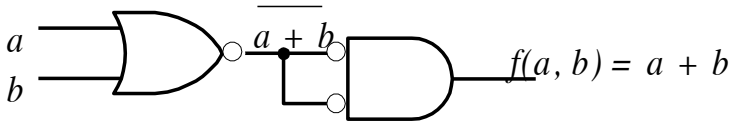
$$f_{NOR}(a,a) = \overline{a+a} = \bar{a} = f_{NOT}(a)$$

$$\bar{f}_{NOR}(a,b) = \overline{\overline{a+b}} = a+b = f_{OR}(a,b)$$

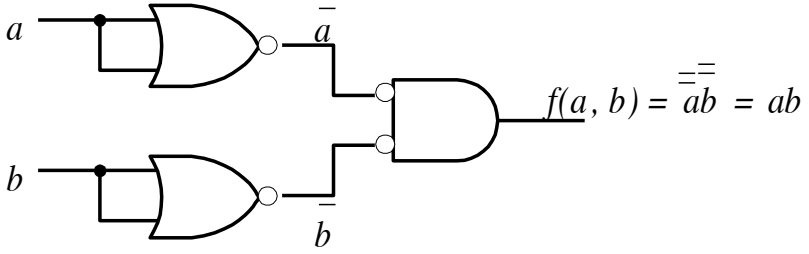
$$f_{NOR}(\bar{a},\bar{b}) = \overline{\bar{a}+\bar{b}} = a \cdot b = f_{AND}(a,b)$$



NOT gate



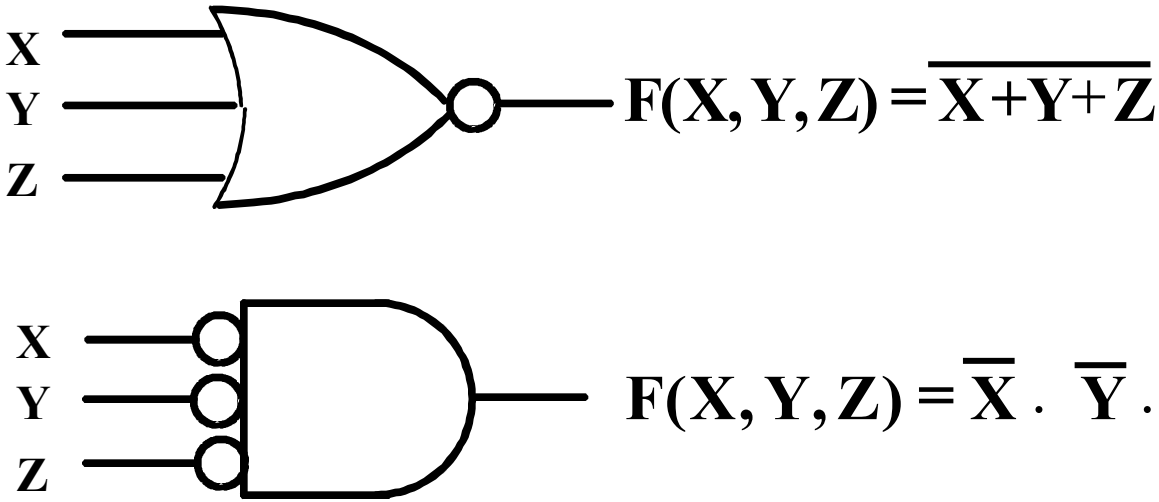
OR gate



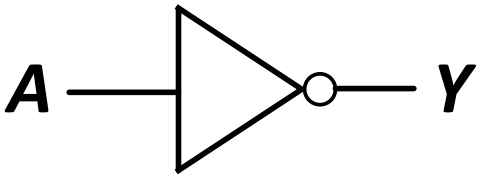
AND gate

Logic Gates: NOR (cont'd)

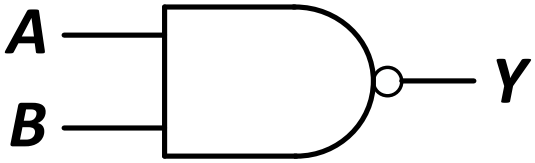
- NOR operation is NOT associative
- (X NOR Y) NOR Z ≠ X NOr (Y NOR Z)



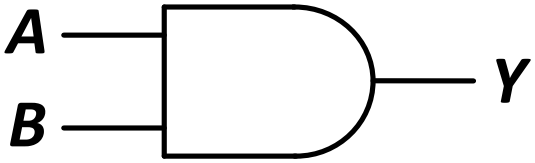
Logic Gates



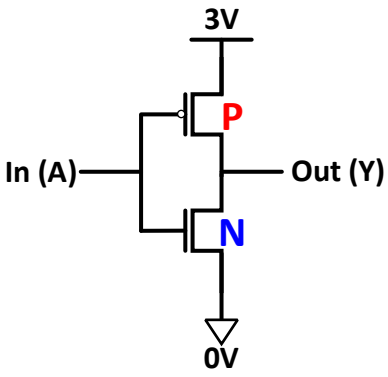
A	Y
0	1
1	0

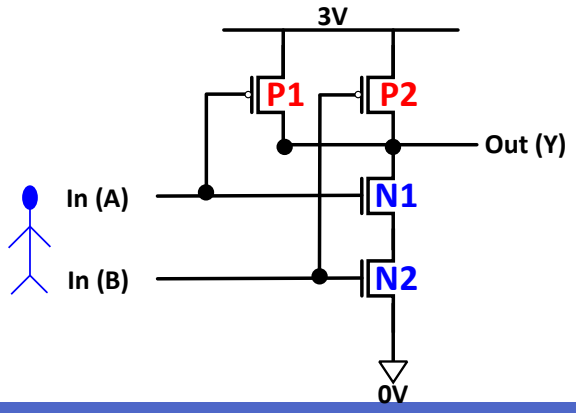


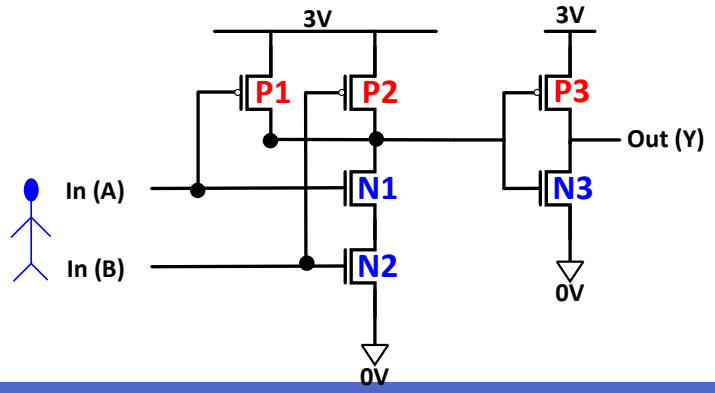
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0



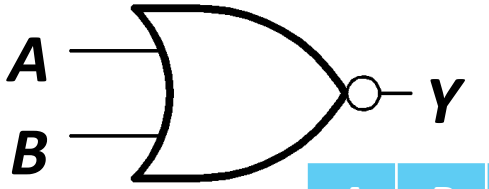
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



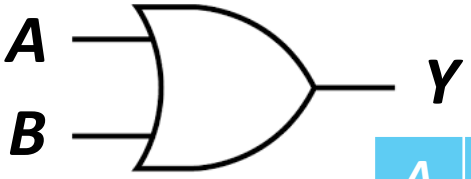




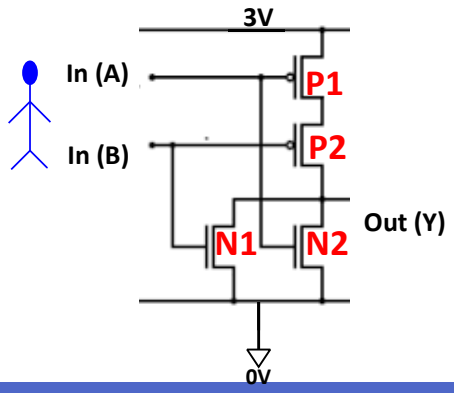
Logic Gates (cont'd)

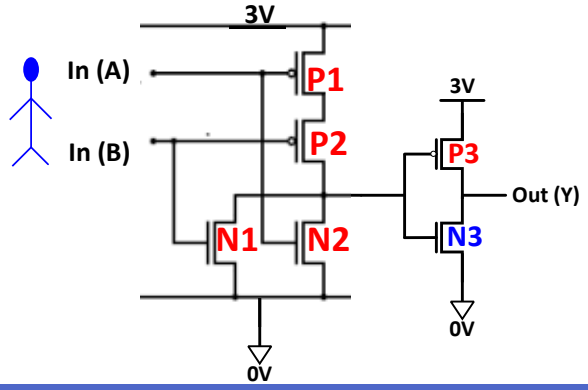


A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0



A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1





Logic Gates: Sample 3

- ≥ 1 in IEEE symbol

ab	$\text{sum}(a, b)$	$\text{sum}(a, b) \geq 1$	$f_{OR}(a, b) = a + b$
00	0	False	0
01	1	True	1
10	1	True	1
11	2	True	1

Logic Gates: Sample 1

- Implement a device which takes two inputs and indicates whether their sum is greater than 1 or not?

ab	$\text{sum}(a, b)$	$\text{sum}(a, b) \geq 1$
00	0	False
01	1	True
10	1	True
11	2	True

Logic Gates: Sample 1

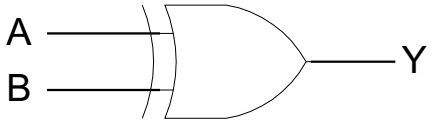
- Implement a device which takes two inputs and indicates whether their sum is greater than 1 or not?

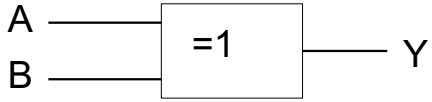
ab	$\text{sum}(a, b)$	$\text{sum}(a, b) \geq 1$	$f_{OR}(a, b) = a + b$
00	0	False	0
01	1	True	1
10	1	True	1
11	2	True	1

Logic Gates: XOR

- Exclusive-OR (XOR)
- (a) XOR logic function
- (b) Electronic XOR gate
- (c) Standard symbol
- (d) IEEE block symbol

$a\ b$	$f_{XOR}(a, b) = a \oplus b$	$A\ B$	Y
0 0	0	L L	L
0 1	1	L H	H
1 0	1	H L	H
1 1	0	H H	L





$$A \oplus B = \bar{A}.B + \bar{B}.A$$

Logic Gates: XOR (cont'd)

- POS of XOR

  $a \oplus b$

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

- Some useful relationships
  - $a \oplus a = 0$
  - $a \oplus \bar{a} = 1$
  - $a \oplus 0 = a$
  - $a \oplus 1 = \bar{a}$
  - $\bar{a} \oplus \bar{b} = a \oplus b$
  - $a \oplus b = b \oplus a$
  - $a \oplus (b \oplus c) = (a \oplus b) \oplus c$

Logic Gates: Sample 2

- Implement a device which takes two inputs and indicates whether their sum is greater than 1 or not?
- Inputs and outputs are 1 bit.

ab	$sum(a, b)$	$sum(a, b) = 1?$
00	0	False
01	1	True
10	1	True
11	2	False

Logic Gates: Sample 2

- Mathematical sum of inputs is *one*
  - Output of XOR gate is asserted
  - Output of XOR is the *modulo*-2 sum of its inputs.

ab	$sum(a, b)$	$sum(a, b) = 1?$	$f(a, b) = a \oplus b$
00	0	False	0
01	1	True	1
10	1	True	1
11	2	False	0

Logic Gates: XNOR

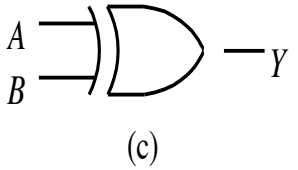
- Exclusive-NOR (XNOR)
- (a) XOR logic function
- (b) Electronic XOR gate
- (c) Standard symbol
- (d) IEEE block symbol

a	b	$f_{XNOR}(a,b) = a \odot b$
0	0	1
0	1	0
1	0	0
1	1	1

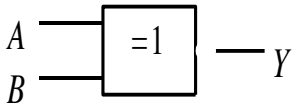
(a)

A	B	Y
L	L	H
L	H	L
H	L	L
H	H	H

(b)



(c)



(d)

$$\overline{A \oplus B} = A \odot B = \overline{\overline{A}.B + \overline{B}.A}$$

Logic Gates: XNOR: POS

- POS of XOR

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

- Some useful relationships
  - $a \oplus a = 0$
  - $a \oplus \bar{a} = 1$
  - $a \oplus 0 = a$
  - $a \oplus 1 = \bar{a}$
  - $\bar{a} \oplus \bar{b} = a \oplus b$
  - $a \oplus b = b \oplus a$
  - $a \oplus (b \oplus c) = (a \oplus b) \oplus c$

Logic Gates: XNOR: SOP

- SOP and POS of XNOR

$$
\begin{aligned}
a \odot b &= \overline{a \oplus b} \\
&= \overline{\bar{a}b + a\bar{b}} \\
&= \overline{\bar{a}b} \cdot \overline{a\bar{b}} \\
&= (a + \bar{b})(\bar{a} + b) \\
&= a\bar{a} + ab + \bar{a}\bar{b} + \bar{b}b \\
&= ab + \bar{a}\bar{b}
\end{aligned}
$$

Logic Gates: XNOR: Is it Correct?

$$a \oplus \bar{b} = a \odot b$$

Logic Gates: XNOR: Is it Correct?

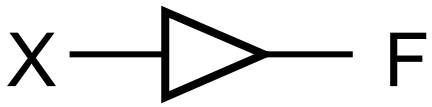
$$a \oplus \bar{b} = a \odot b$$

$$a \oplus \bar{b} = \bar{a}.\bar{b} + a.b$$

$$a \odot b = a.b + \bar{a}.\bar{b}$$

Logic Gates: Buffer

- F(x)= x
- Boolean function is a connection.
- Usage
  - Amplify an input signal
  - Permits more gates to be attached to output



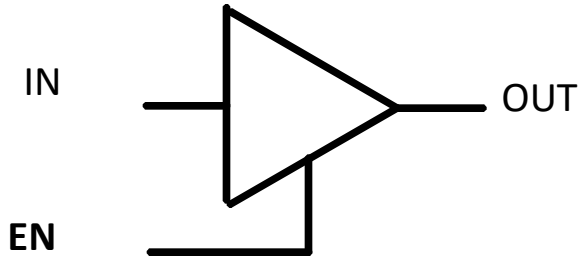
X	F
0	0
1	1

Logic Gates: Hi-Impedance

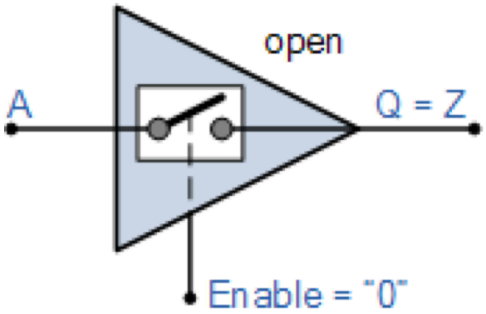
- Logic gates have 0 or 1 as output; Two-state logic
  - Their output cannot connect together
- Three-state logic:
  - Adds a third value; Hi-Impedance output, Hi-Z
  - Three output values: 1, 0, and Hi-Z
  - Output appears to be **disconnected** from the input
  - Behaves as an open circuit between gate input & output
  - Hi-impedance gates can connect their outputs together

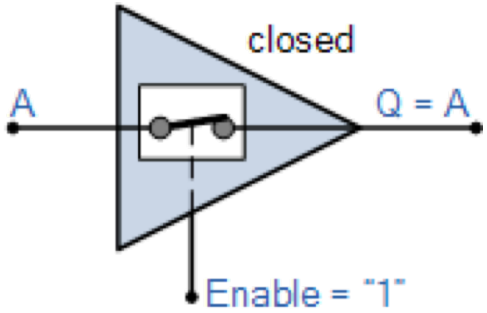
Logic Gates: 3-State Buffer

- Two inputs
  - Data input (IN)
  - Enable control input (EN)
- One output
  - output (OUT)



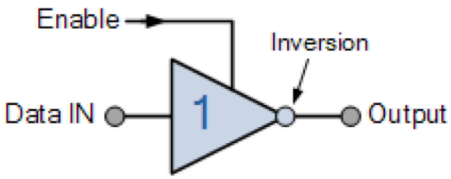
EN	IN	OUT
0	X	Hi-Z
1	0	0
1	1	1

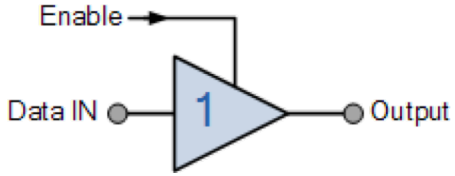




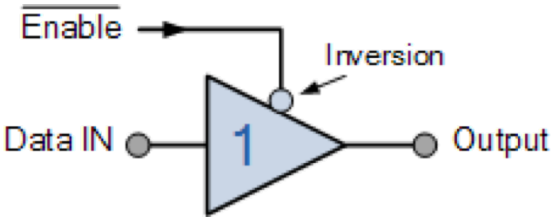
Logic Gates: 3-State Buffer (cont'd)

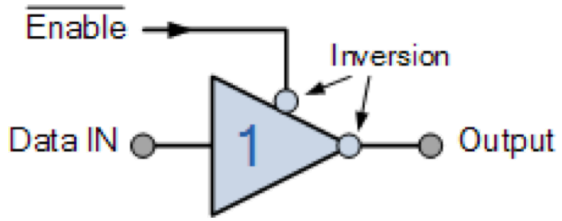
- Active high





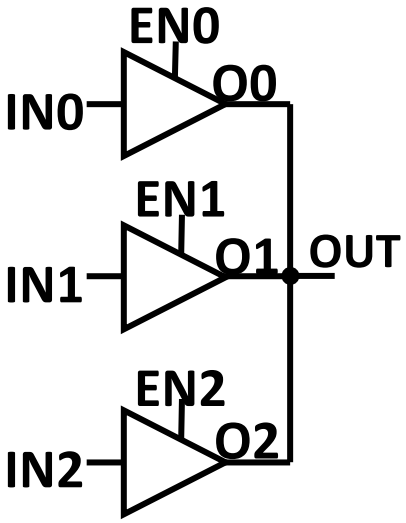
- Active low





Logic Gates: 3-State Buffer (cont'd)

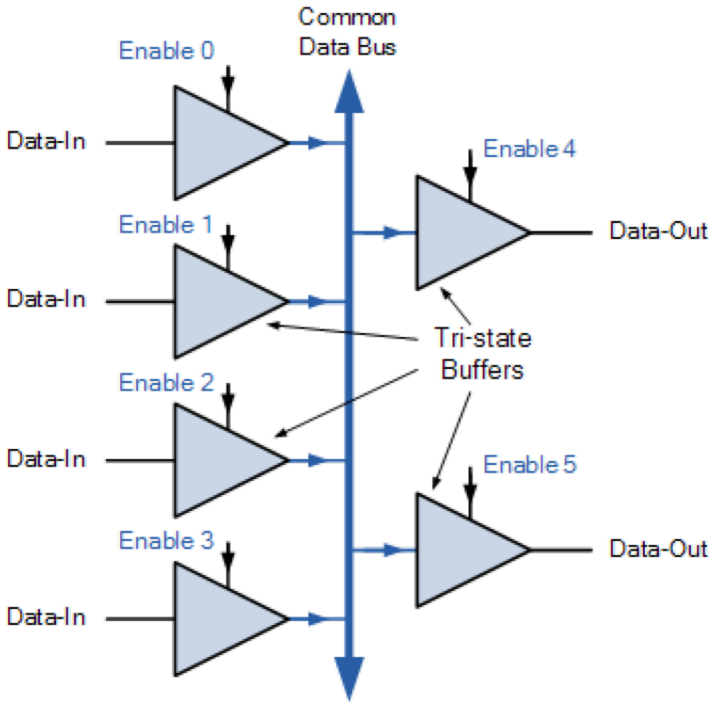
- Output of 3-state buffers can be **wired together**
- At most one 3-state buffer can be enabled.
  - Resolved output is equal to the output of the enabled 3-state buffer
- If multiple 3-state buffers are enabled at the same time
  - Conflicting outputs will burn the circuit



Resolution Table			
O0	O1	O2	OUT
0 or 1	Hi-Z	Hi-Z	O0
Hi-Z	0 or 1	Hi-Z	O1
Hi-Z	Hi-Z	0 or 1	O2
Hi-Z	Hi-Z	Hi-Z	Hi-Z
0 or 1	0 or 1	0 or 1	Burn

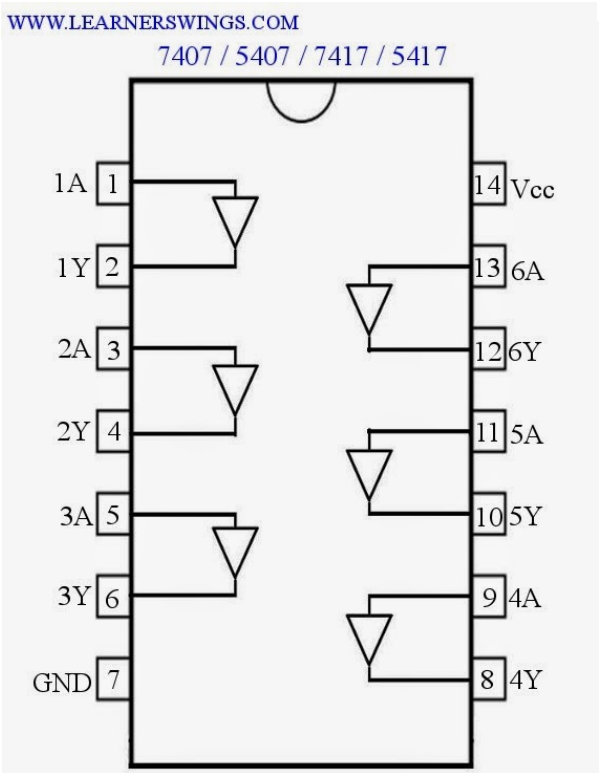
3-State Buffer & Data Bus

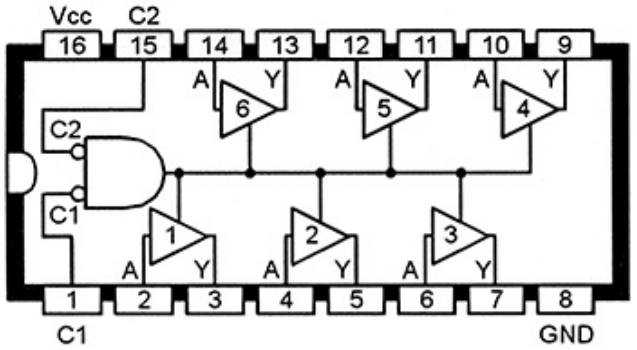
- Isolate devices and circuits from the data bus and one another



Logic Gates: 3-State Buffer (cont'd)

- TTL ICs

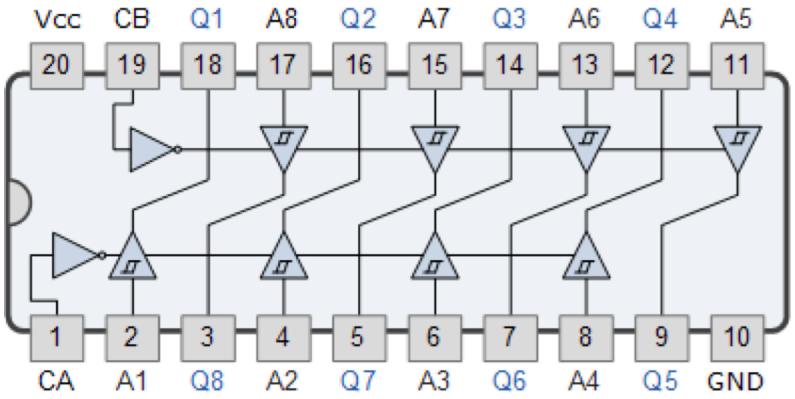




C1	C2	A	Y
1	X	X	HI-Z
X	1	X	HI-Z
0	0	1	1
0	0	0	0

Notes: X = either 0 or 1 logic levels.
Fan-out = 30 per gate.

74LS365



74LS244 Octal Tri-state Buffer

Logic Gates: Types

- Signals and logic values
  - A signal that is set to logic 1 is said to be asserted, active, or true.
  - An active-high signal is asserted when it is high (positive logic).
  - An active-low signal is asserted when it is low (negative logic).
- Positive logic
- Negative logic

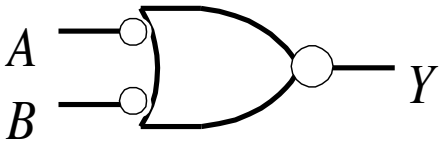
Electric Signal	Logic Value	
	Positive Logic	Negative Logic
High Voltage (H)	1	0
Low Voltage (L)	0	1

Negative Logic: AND

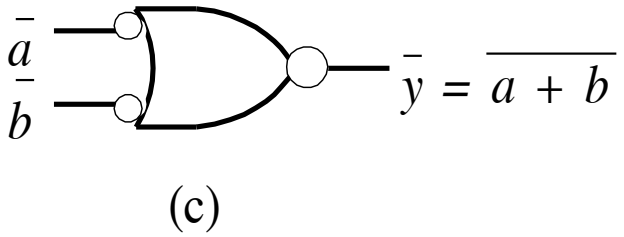
- (a) AND gate truth table (L=1, H=0)
- (b) Alternate AND gate symbol (in negative logic)
- (c) Preferred usage
- (d) Improper usage

A	B	Y
1	1	1
1	0	1
0	1	1
0	0	0

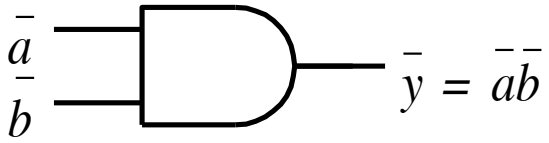
(a)



(b)



(c)



(d)

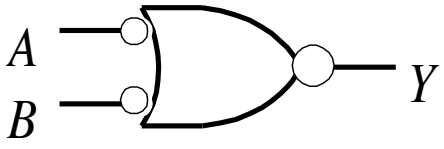
Logic Gates: Negative Logic: AND (cont'd)

y = a . b = $\overline{\overline{a \cdot b}} = \overline{\overline{a} + \overline{b}} = \bar{f}_{OR}(\overline{a}, \overline{b})$

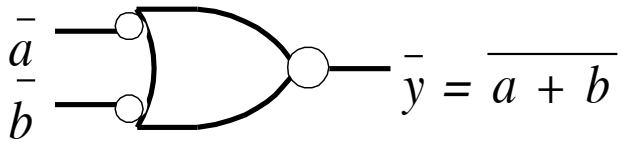
$\bar{y} = \overline{(\overline{a}) + (\overline{b})} = \overline{a + b} = \bar{f}_{OR}(a, b)$

A	B	Y
1	1	1
1	0	1
0	1	1
0	0	0

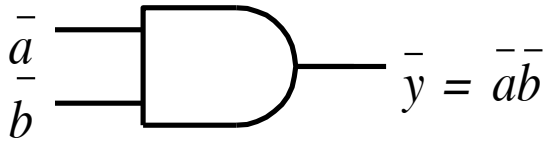
(a)



(b)



(c)



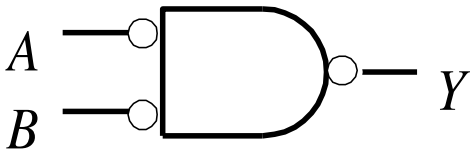
(d)

Logic Gates: Negative Logic: OR

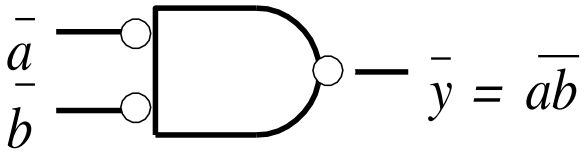
- (a) OR gate truth table (L=1, H=0)
- (b) Alternate OR gate symbol (in negative logic)
- (c) Preferred usage
- (d) Improper usage

A	B	Y
1	1	1
1	0	0
0	1	0
0	0	0

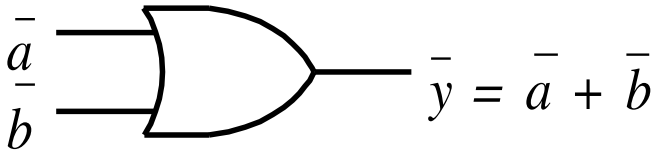
(a)



(b)



(c)



(d)

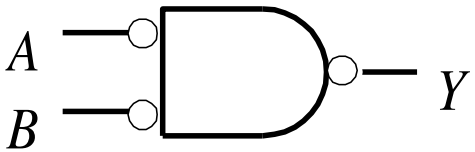
Logic Gates: Negative Logic: OR (cont'd)

$$y = a + b = \overline{\overline{a + b}} = \overline{\bar{a} \cdot \bar{b}} = \bar{f}_{AND}(\bar{a}, \bar{b})$$

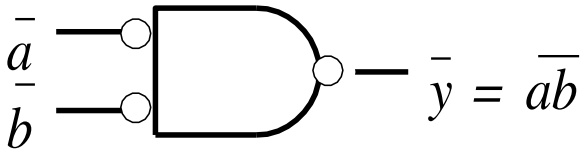
$$\bar{y} = \overline{(\bar{a}) \cdot (\bar{b})} = \overline{a \cdot b} = \bar{f}_{AND}(a, b)$$

A	B	Y
1	1	1
1	0	0
0	1	0
0	0	0

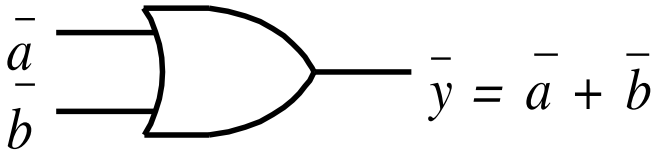
(a)



(b)



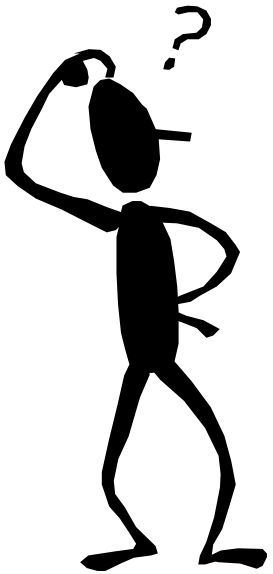
(c)



(d)

Logic Gates: Sample 4

- Building smoke alarm system



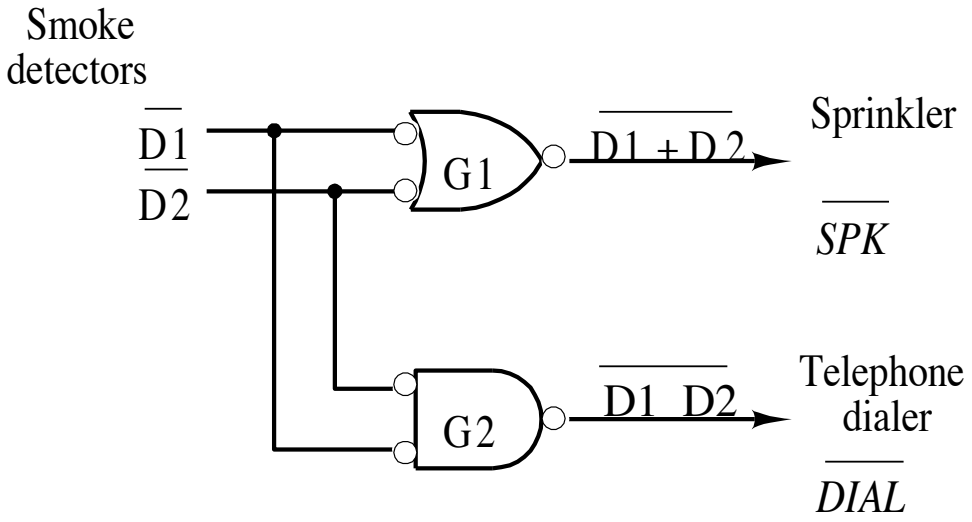
Building Smoke Alarm System

- Components:
  - Two Active-low smoke detectors $\overline{D1}, \overline{D2}$
  - A sprinkler
    - Active-low input to the sprinkler $\overline{SPK}$
  - An automatic telephone dialer
    - Active-low input to the telephone dialer $\overline{DIAL}$
- Behavior:
  - Sprinkler is activated if either smoke detector detects smoke.
  - When both smoke detector detect smoke, fire department is called.

$$\overline{SPK} = \overline{D1 + D2}$$
$$\overline{DIAL} = \overline{D1 \cdot D2}$$

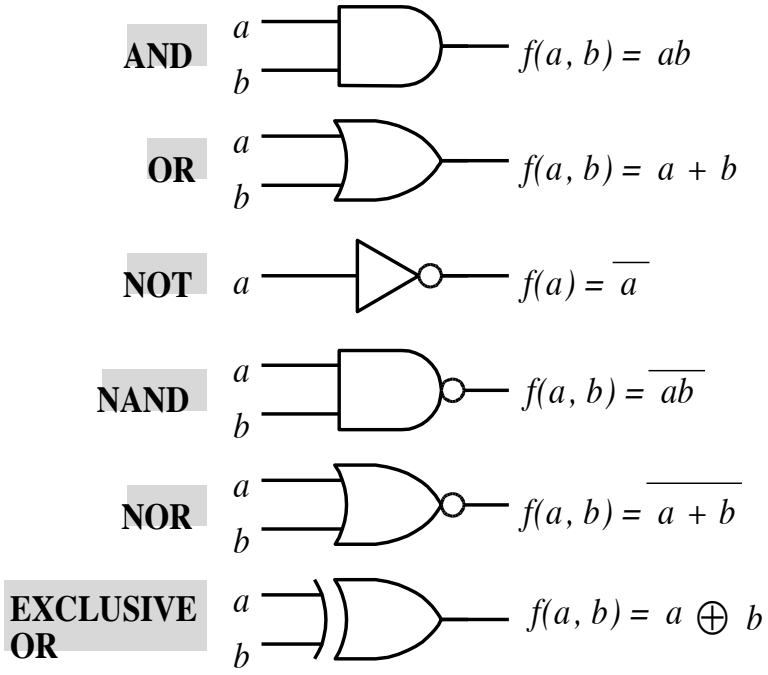
Building Smoke Alarm System

$$\overline{SPK} = \overline{D1 + D2}$$
$$\overline{DIAL} = \overline{D1 \cdot D2}$$

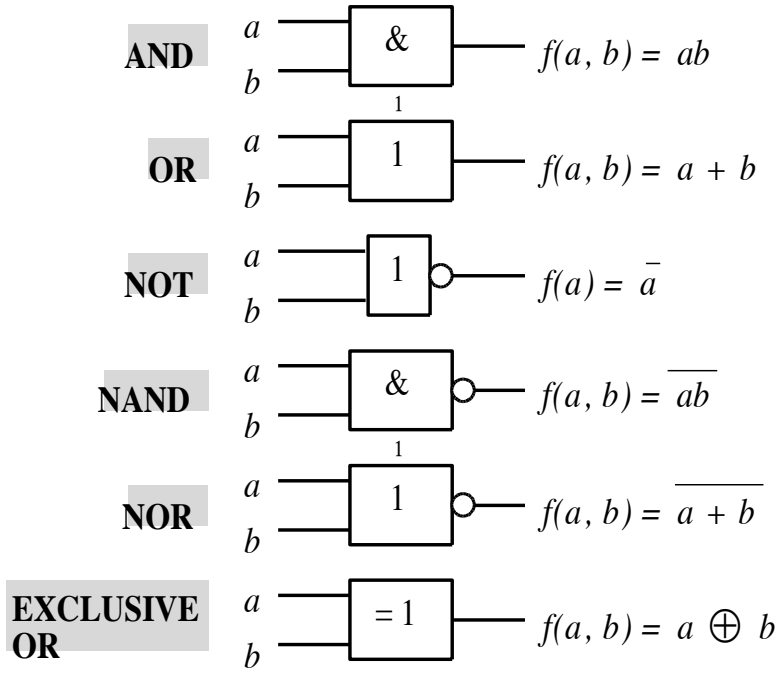


Electronic Logic Gates

- Logic gates

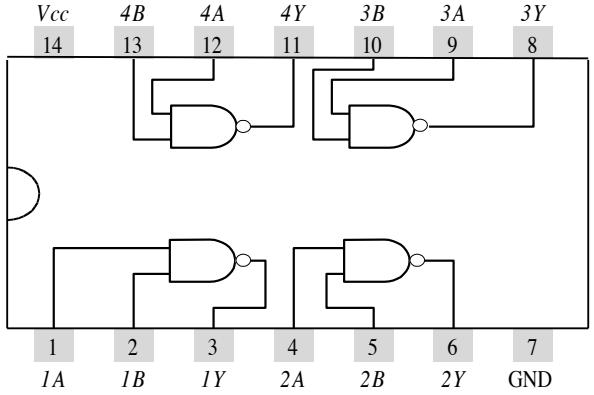


Symbol set 1

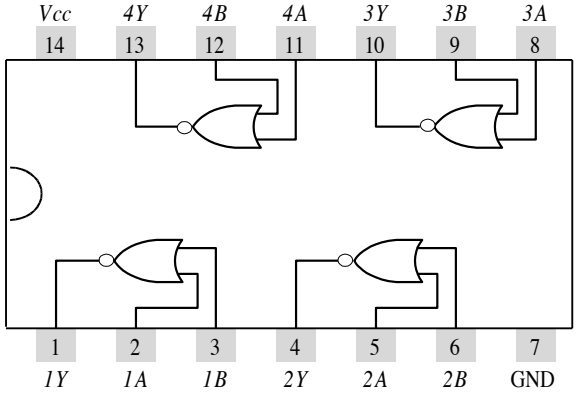


Symbol set 2
(ANSI/IEEE Standard 91-1984)

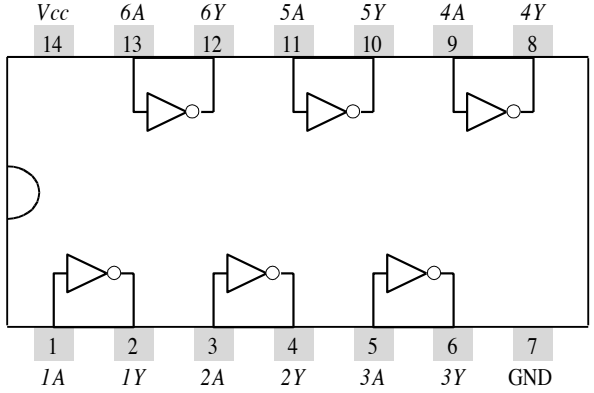
Logic Gates: Sample 1



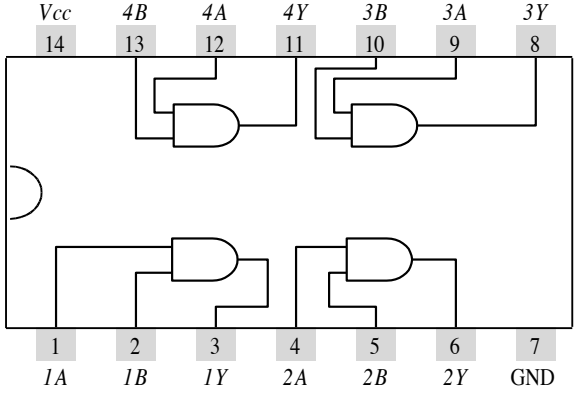
7400: $Y = \overline{AB}$
Quadruple two-input NAND gates



7402: $Y = \overline{A + B}$
Quadruple two-input NOR gates

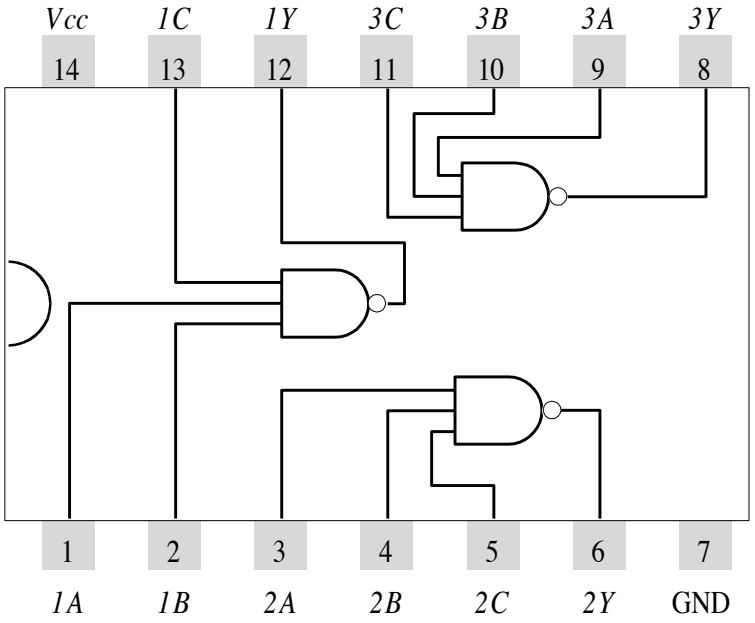


7404: $Y = \overline{A}$
Hex inverters

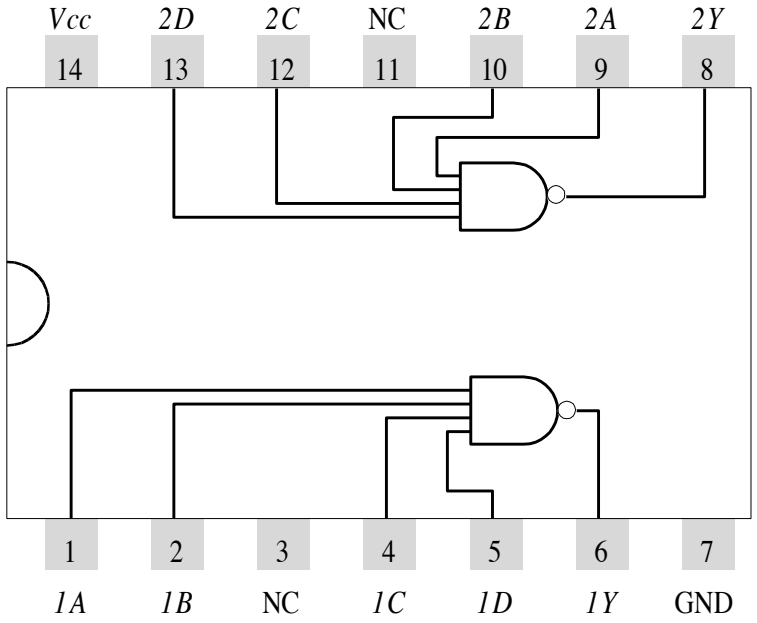


7408: $Y = AB$
Quadruple two-input AND gates

Logic Gates: Sample 2

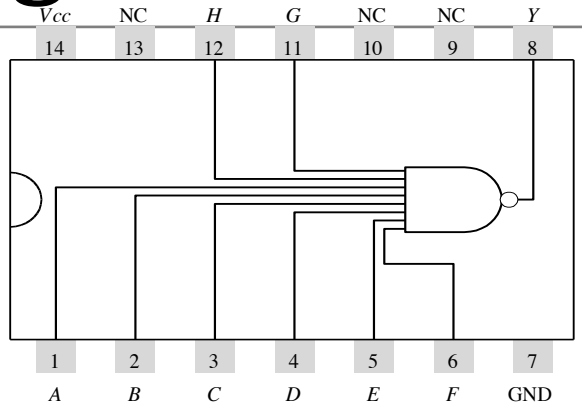


7410: $Y = \overline{ABC}$
Triple three-input NAND gates

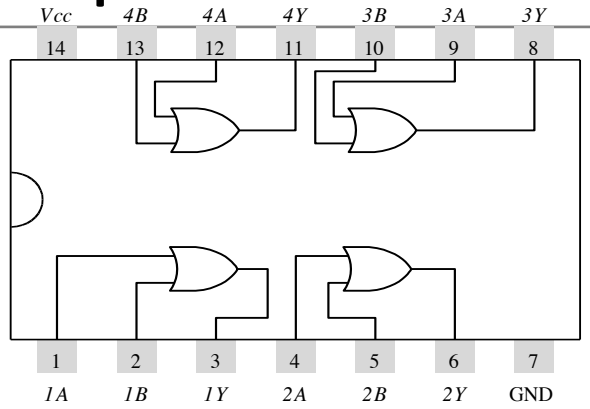


7420: $Y = \overline{ABCD}$
Dual four-input NAND gates

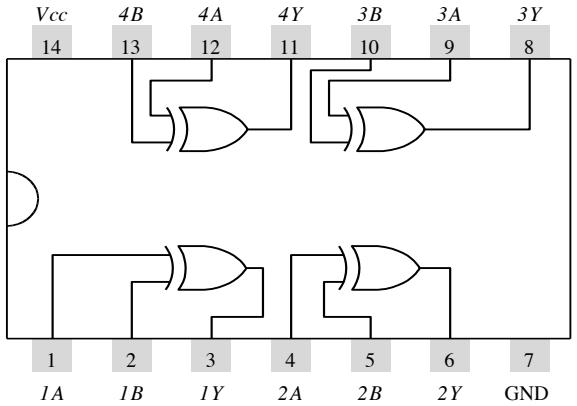
Logic Gates: Sample 3



7430: Y = ABCDEFGH
8-input NAND gate



7432: Y = A + B
Quadruple two-input OR gates



7486: Y = A Å B
Quadruple two-input exclusive-OR gates

Thank You