

3. Logic Gates and Boolean Algebra

Digital circuits can be found in smart phones, computers, washing machines, cars, etc.

Logic gates are the basic building blocks of digital circuits.

Boolean algebra is used to describe, design and simplify digital circuits.

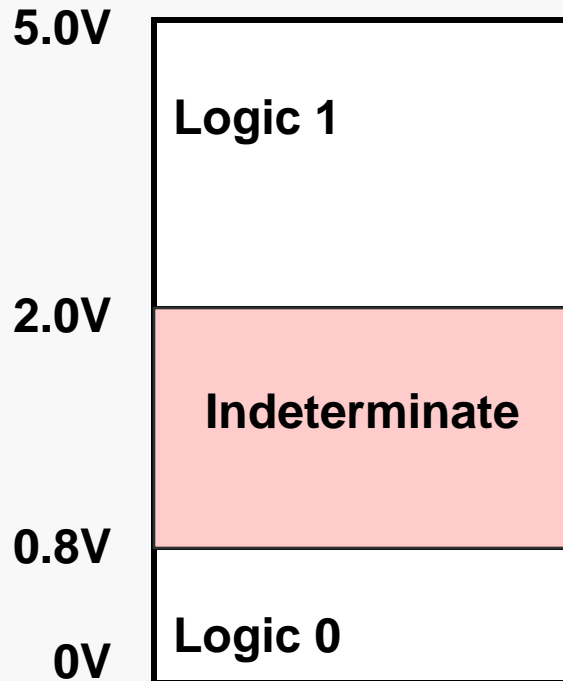
Boolean Constants – only 2 values

- **TRUE, FALSE**
- **Logic HIGH, Logic LOW**
- **HI, LO**
- **1, 0**

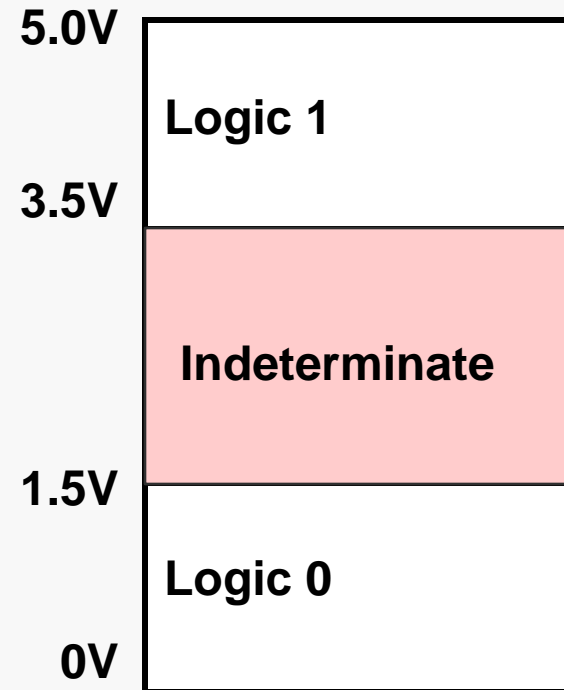
Boolean (logic) variables can only assume one of the two values

Common Logic-level voltage ranges

TTL



CMOS (74AC)

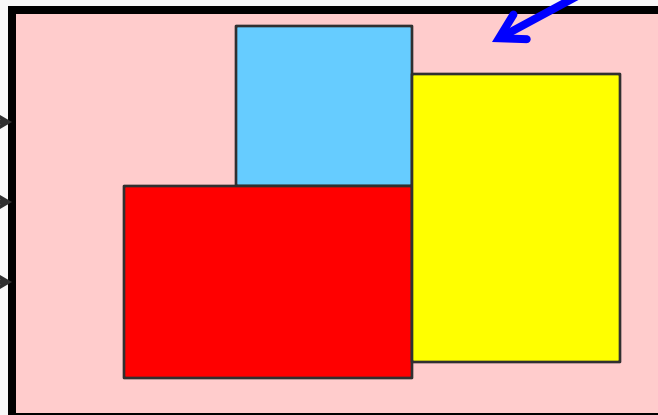


A typical logic circuit

- 1 or more logic inputs
- 1 or more logic outputs
- Outputs are related to inputs by logic functions

inputs:

X
Y
Z



Smaller logic
circuit blocks

output:

F

Truth table

- Logic function can be fully described by a **Truth Table**.
- The Truth table
 - shows how a logic circuit's output responds to various combinations of logic inputs
 - has 2^N number of input combinations for N inputs
 - lists all possible input combinations in the binary counting sequence

A typical N-input truth table

2^N
rows

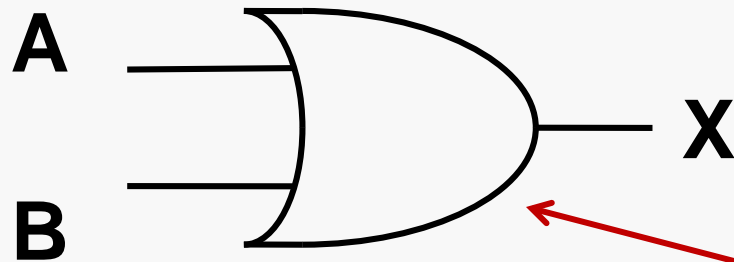
Input 1	Input 2	...	Input N-1	Input N	Output
0	0	...	0	0	1
0	0	...	0	1	0
0	1	0	0
0	1	1	.
.
.
1	0	0	.
1	0	1	.
1	1	...	1	0	.
1	1	...	1	1	.

3 Basic Logic Operations

- **Logical addition - OR**
- **Logical multiplication - AND**
- **Logical complement or inversion - NOT**

In digital circuits, these are realised by electronic devices called logic gates.

Logical OR Operation



Logic
symbol

- Logical OR operation

- $X = A + B$

- $X = A \text{ OR } B$

Logic
expression

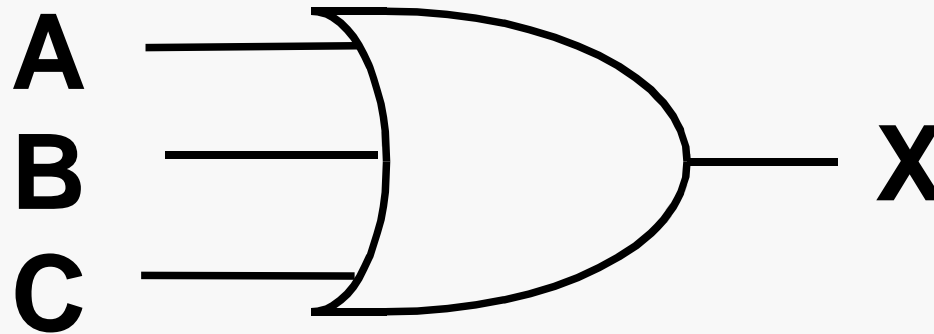
- 2-input OR gate: $X = A + B$

Truth table for a 2-input OR gate

($X = A+B$)

Inputs		Output
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

$X=1$ if at least one input is = 1



3-input OR gate:

$$X = A + B + C$$

OR operation result will be 1 if at least one input is 1

Truth Table for a 3-input OR gate

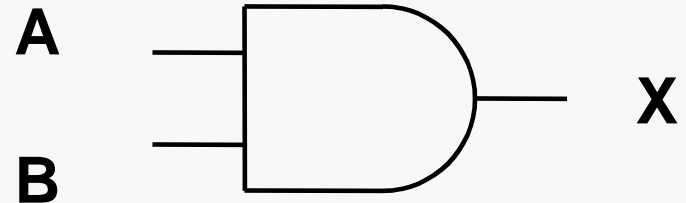
$(X = A+B+C)$

Inputs			Output
A	B	C	X
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Logical AND operation

Logical AND operation

- $X = A \bullet B$
- $X = A \text{ AND } B$
- $X = AB$

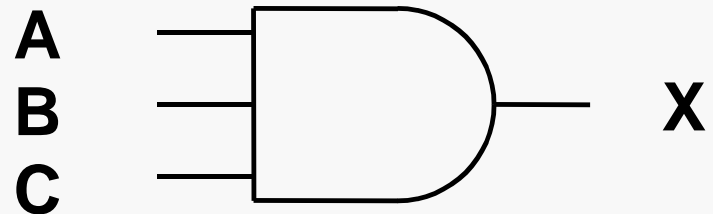


2-input AND gate:

- $X = AB$

3-input AND gate:

- $X = ABC$



AND operation result will be 1 only if all inputs are 1

Truth table for 2-input AND gate

$(X = AB)$

Inputs		Output
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

Output $X=1$ only if all inputs are 1

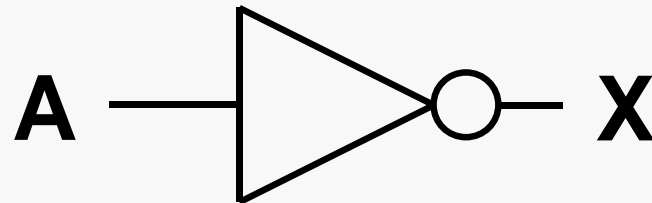
The Truth Table for a 3-input AND gate ($X = ABC$)

Inputs			Output
A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Logical NOT operation

- NOT gate only has 1 input and it is commonly known as an **inverter**
- the output is the complement/inverse of the input

- $X = \overline{A}$



- $X = A'$ ← we will use this notation

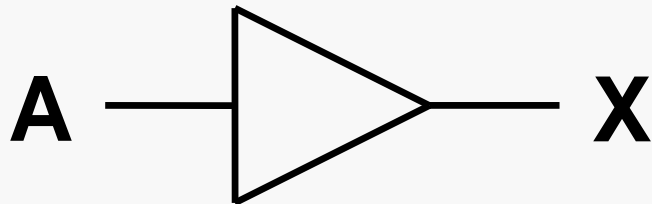
Logical NOT operation

- Its truth table is very simple

input	output
A	$X = A'$
0	1
1	0

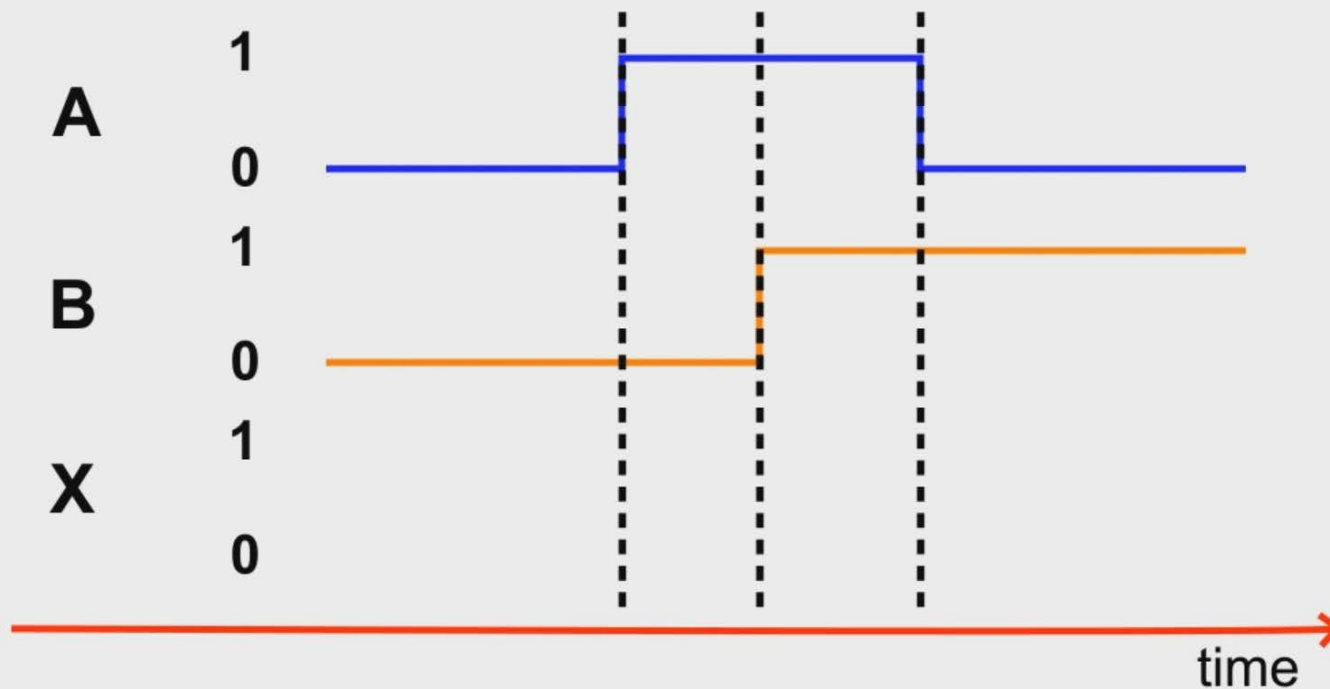
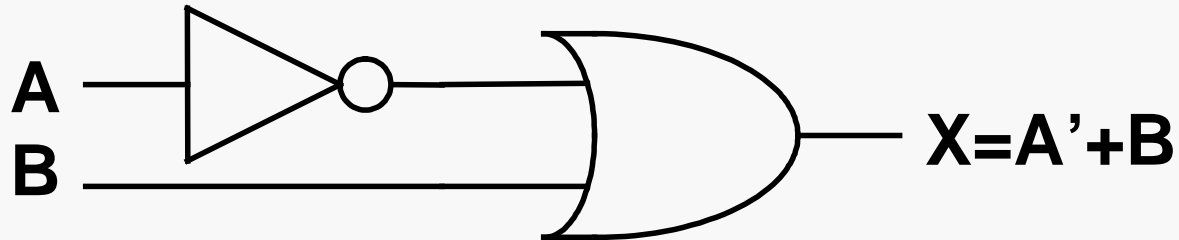
Buffer

- Its truth table is also very simple
- No change in logic



input	output
A	$X = A$
0	0
1	1

- **Example: Sketch the logic waveform of X**



- **Timing diagram with more realistic appearance, showing propagation delays, rise time and fall time**

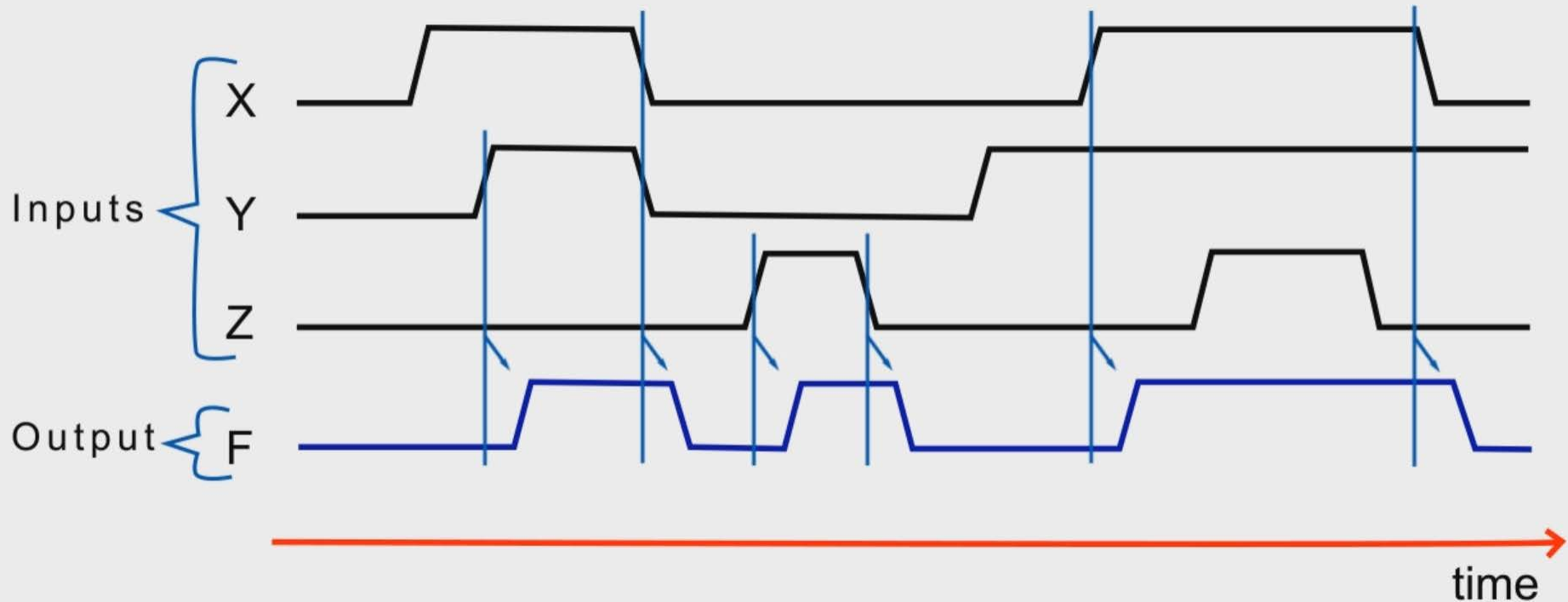


Fig. 3.17 (taken from Wakerly)

Boolean Algebra

- **Helps to analyse logic circuits.**
- **Express operations mathematically.**
- **Similar to normal algebra but much simpler.**
- **It does not have fraction or negative number.**

Order of Precedence in Boolean Algebra:

- Complement over a single variable (inversion)
- Expression within parentheses
- AND
- OR

Examples:

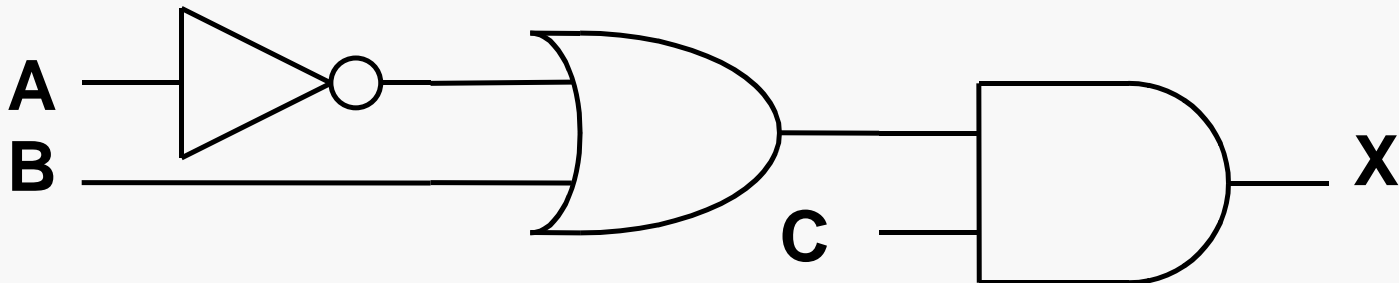
1. $Y = A + B C'$

2. $Y = (A + B) C'$

3. $Y = A + (B C)'$

Describing logic circuits algebraically

- **AND, OR and NOT operations**
 - are basic building blocks of digital system
 - can completely describe any logic circuit
- **Example: express output X in terms of inputs A, B and C**



Evaluating logic circuit outputs

From a Boolean expression, the logic level of an output can be determined for any values of the circuit inputs.

Example

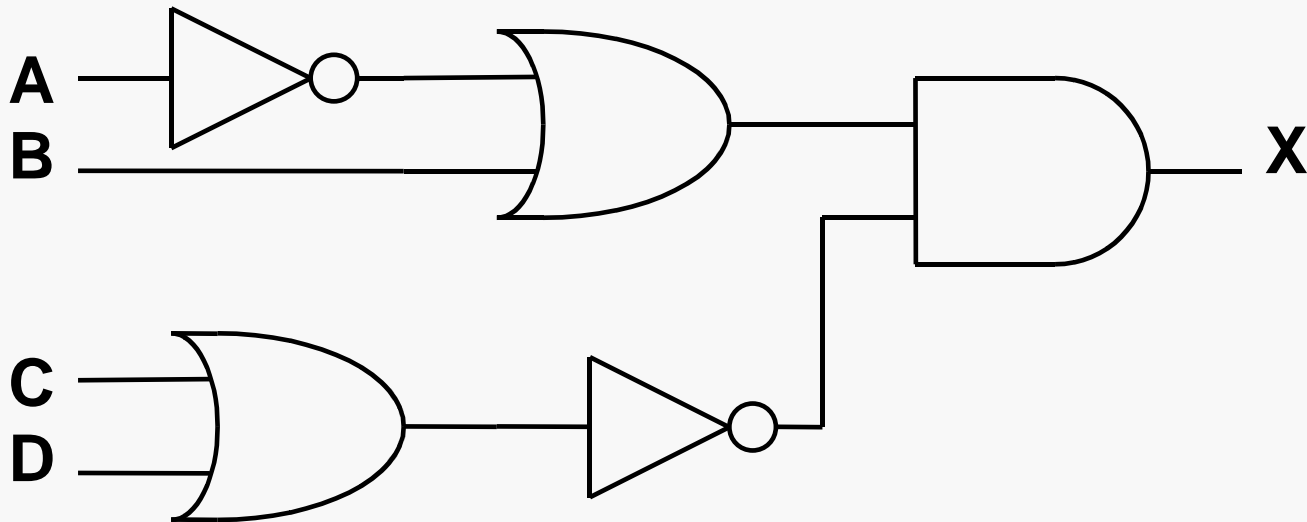
$$X = A'(B+C)(A+D)'$$

**If inputs A,B,C,D = 0,1,1,0
X = ?**

**If inputs A,B,C,D = 1,1,1,1
X = ?**

**If inputs A,B,C,D = 0,0,0,0
X = ?**

Determining Instantaneous Output Level from a Logic Circuit Diagram



e.g. if $A=1$, $B=1$, $C=0$, $D=0$, then $X=?$

Implementing Circuits from Boolean expressions

Example $Y = AC + BC' + A'BC$

Boolean Theorems

- Many of the theorems are similar to those in normal algebra.
- The theorems can be used to simplify logic expressions and therefore can help to simplify logic circuits.
- Simpler circuits cost less to build and are less prone to failure.

Boolean Theorems

Axioms:

$$X = 0 \text{ if } X \neq 1$$

$$X = 1 \text{ if } X \neq 0$$

$$0 \cdot 0 = 0$$

$$1 \cdot 1 = 1$$

$$0 \cdot 1 = 1 \cdot 0 = 0$$

$$1 + 1 = 1$$

$$0 + 0 = 0$$

$$1 + 0 = 0 + 1 = 1$$

Single variable theorems:

$$X \bullet 0 = 0$$

$$X \bullet 1 = X$$

$$X \bullet X = X$$

$$X \bullet X' = 0$$

$$X + 1 = 1$$

$$X + 0 = X$$

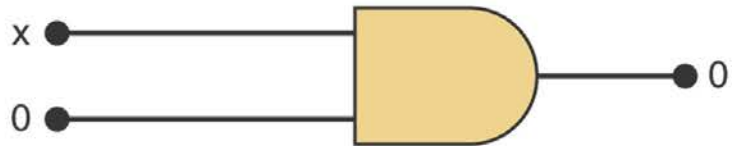
$$X + X = X$$

$$X + X' = 1$$

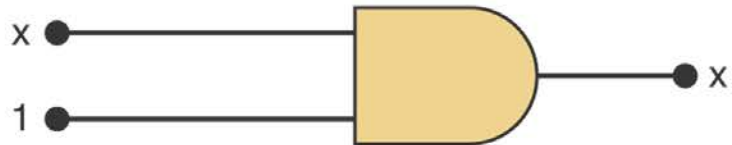
$$(X')' = X$$

Duality: any theorem or identity in switching algebra remains true if 0 and 1 are swapped and \cdot and $+$ are swapped throughout

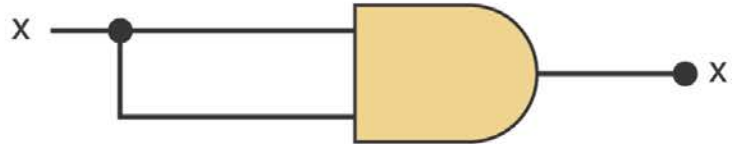
Single variable theorems:



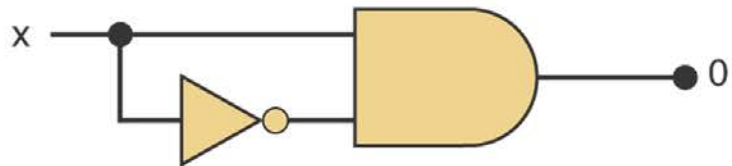
(1) $x \cdot 0 = 0$



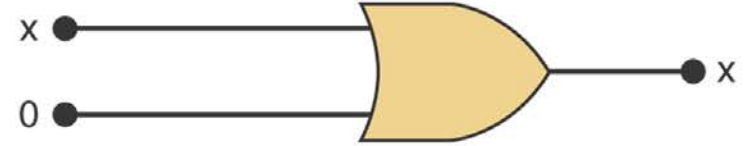
(2) $x \cdot 1 = x$



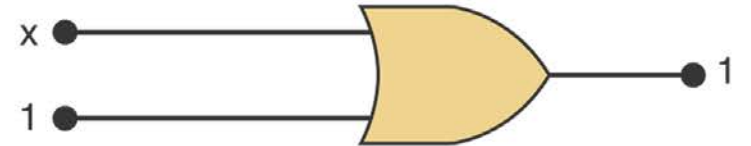
(3) $x \cdot x = x$



(4) $x \cdot \bar{x} = 0$



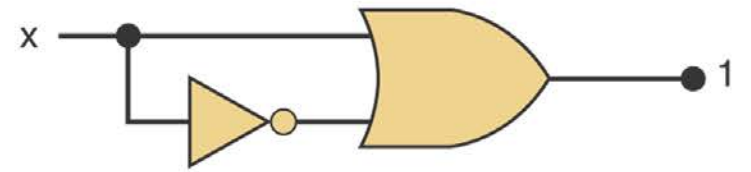
(5) $x + 0 = x$



(6) $x + 1 = 1$



(7) $x + x = x$



(8) $x + \bar{x} = 1$

Fig. 3-25 (Tocci, 10th ed. Pg. 77)

Multivariable theorems:

- **Commutative laws:**

$$A + B = B + A$$

$$A \bullet B = B \bullet A$$

- **Associative laws:**

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

$$A(BC) = (AB)C = ABC$$

- **Distributive laws:**

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

$$(A + B)(C + D) = AC + BC + AD + BD$$

Absorption laws

$$A + AB = A$$

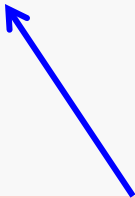
proof:

$$\begin{aligned} A + AB &= A(1 + B) \\ &= A \bullet 1 = A \end{aligned}$$

$$A + A'B = A + B$$

proof:

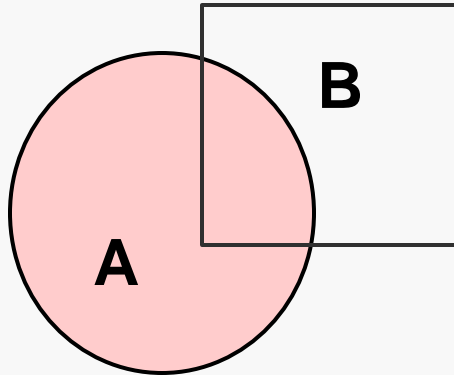
$$\begin{aligned} A + A'B &= A + AB + A'B \\ &= A + (A + A')B \\ &= A + (1) B \\ &= A + B \end{aligned}$$



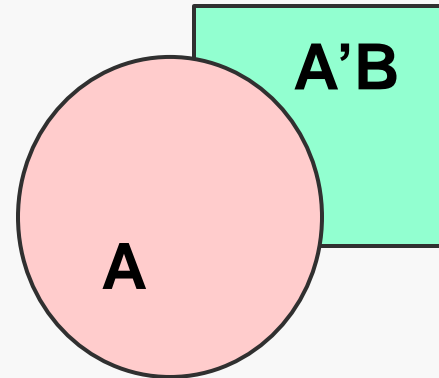
A technique commonly
used in algebraic
simplification

Absorption laws (Venn diagrams)

$$A + AB = A$$



$$A + A'B = A + B$$



Consensus

$$AB + A'C + BC = AB + A'C$$

proof:

$$BC = ABC + A'BC$$

$$\text{Thus } AB + A'C + BC = AB + A'C + ABC + A'BC$$

$$= AB + ABC + A'C + A'BC$$

$$= AB(1+C) + A'C(1+B)$$

$$= AB + A'C$$

DeMorgan's Theorems

- $(A + B)' = A' \bullet B'$

proof:

A	B	$(A+B)'$	$A' \bullet B'$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

DeMorgan's Theorems

- $(AB)' = A' + B'$

proof:

A	B	$(AB)'$	$A' + B'$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

DeMorgan's Theorems generalise to many variables

$$(A+B+C+D+\dots)' = A' \bullet B' \bullet C' \bullet D' \bullet \dots$$

$$(ABCD\dots)' = A'+B'+C'+D'+ \dots$$

- Add or remove inverter to each variable
- Interchange AND with OR

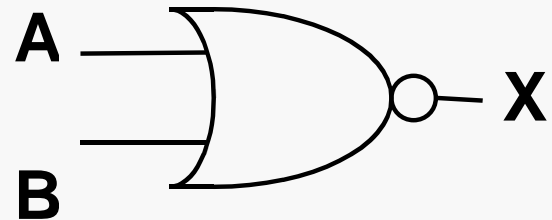
Example

Simplify $[A (B + C')' D]'$

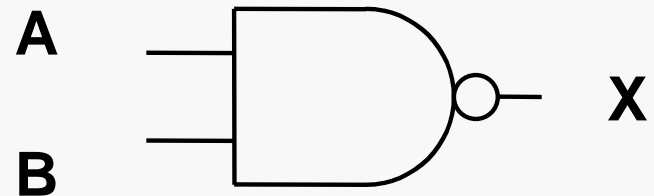
NOR gate & NAND gate

- Combines basic operations of OR & AND with NOT

- NOR: $X = (A+B)'$



- NAND: $X = (AB)'$



Truth table for 2-input NOR gate

$$X = A \text{ NOR } B$$

Inputs		Output
A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

Output $X=1$ only when all inputs are 0

Truth table for 2-input NAND gate

$X = A \text{ NAND } B$

Inputs		Output
A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

Output $X=0$ only when all inputs are 1

Truth table for 3-input NOR gate

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

Inputs			Output
A	B	C	X
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Truth table for 3-input NAND gate

$$X = (ABC)'$$

Inputs			Output
A	B	C	X
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Summary

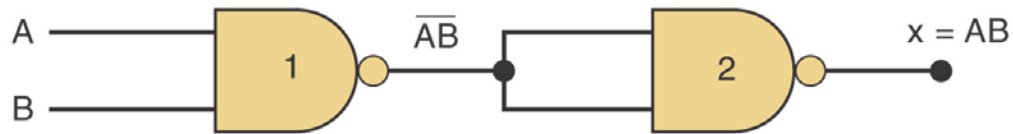
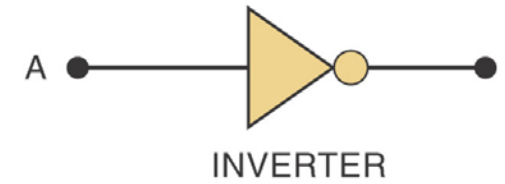
- **OR:** output is 1 when any of the inputs is 1
- **AND:** output is 1 when all the inputs are 1
- **NOT:** output is 0 when input is 1 and vice versa
- **NOR:** output is 0 when any of the inputs is 1
- **NAND:** output is 0 when all the inputs are 1

Universality of NAND gates and NOR gates

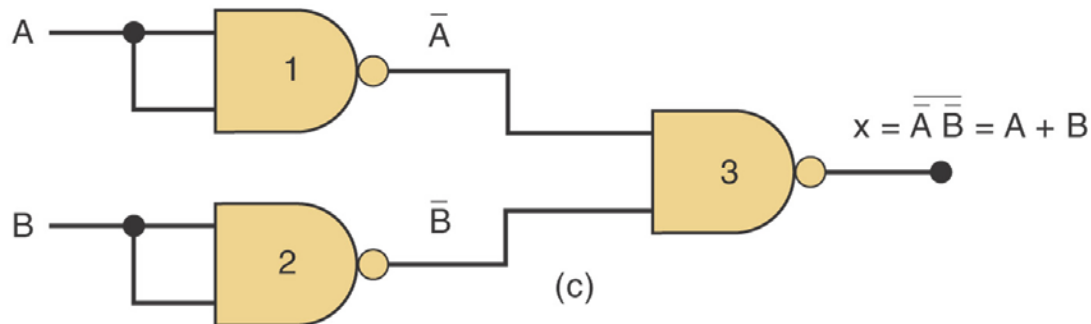
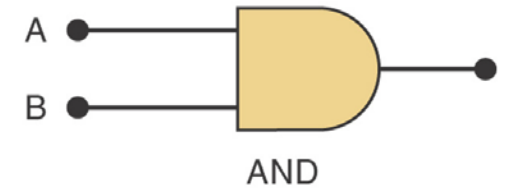
- **NAND gates can be used to form AND gate, OR gate and NOT gate**
- **Therefore, NAND gates can be used to implement any Boolean function**
- **Similarly for NOR gates**
- **Equivalence can be proved by DeMorgan's theorems**



(a)



(b)



(c)

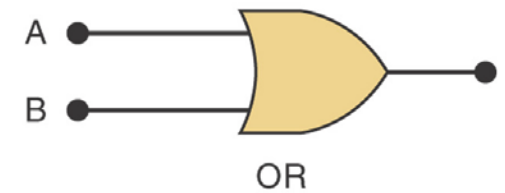
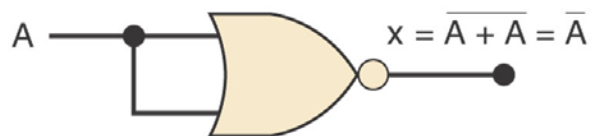
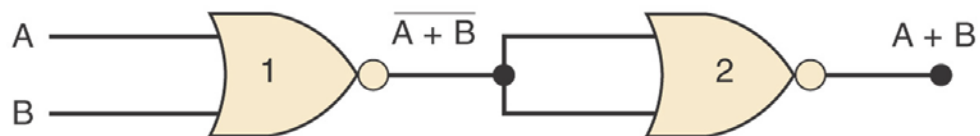
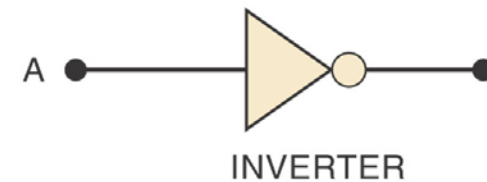


Fig. 3-29 Basic gates from NAND

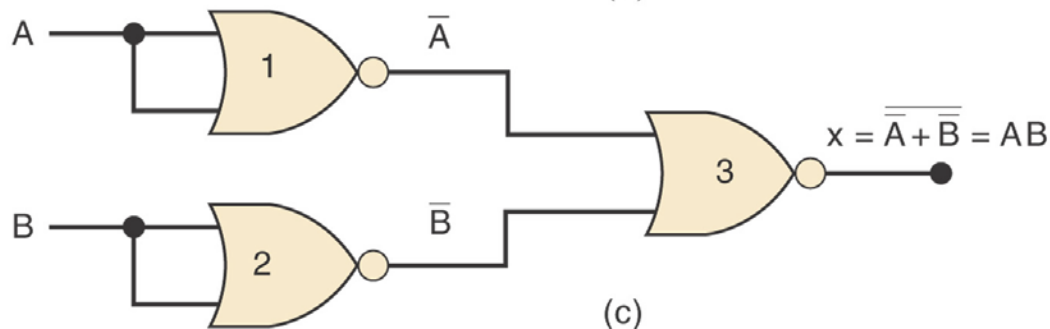
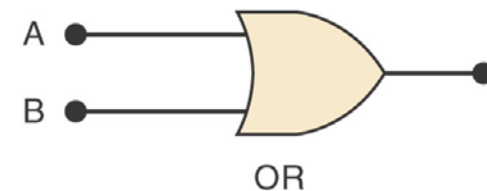
(Tocci, 10th ed. Pg. 84)



(a)



(b)



(c)

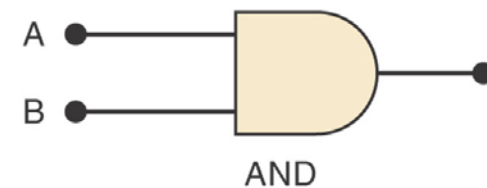


Fig. 3-30 Basic gates from NOR

(Tocci, 10th ed. Pg. 84)

Alternate Logic gate representations

- The alternate symbol is obtained from the standard symbol by applying DeMorgan's theorems
 - adding bubbles to each input and output of the standard symbol
 - removing bubble from the output of the standard symbol
 - change the operation symbol from AND to OR and vice versa
 - NOT symbol shape is not changed

Notice that there is no bubble at the inputs of the standard symbols (Fig. 3-33, left).

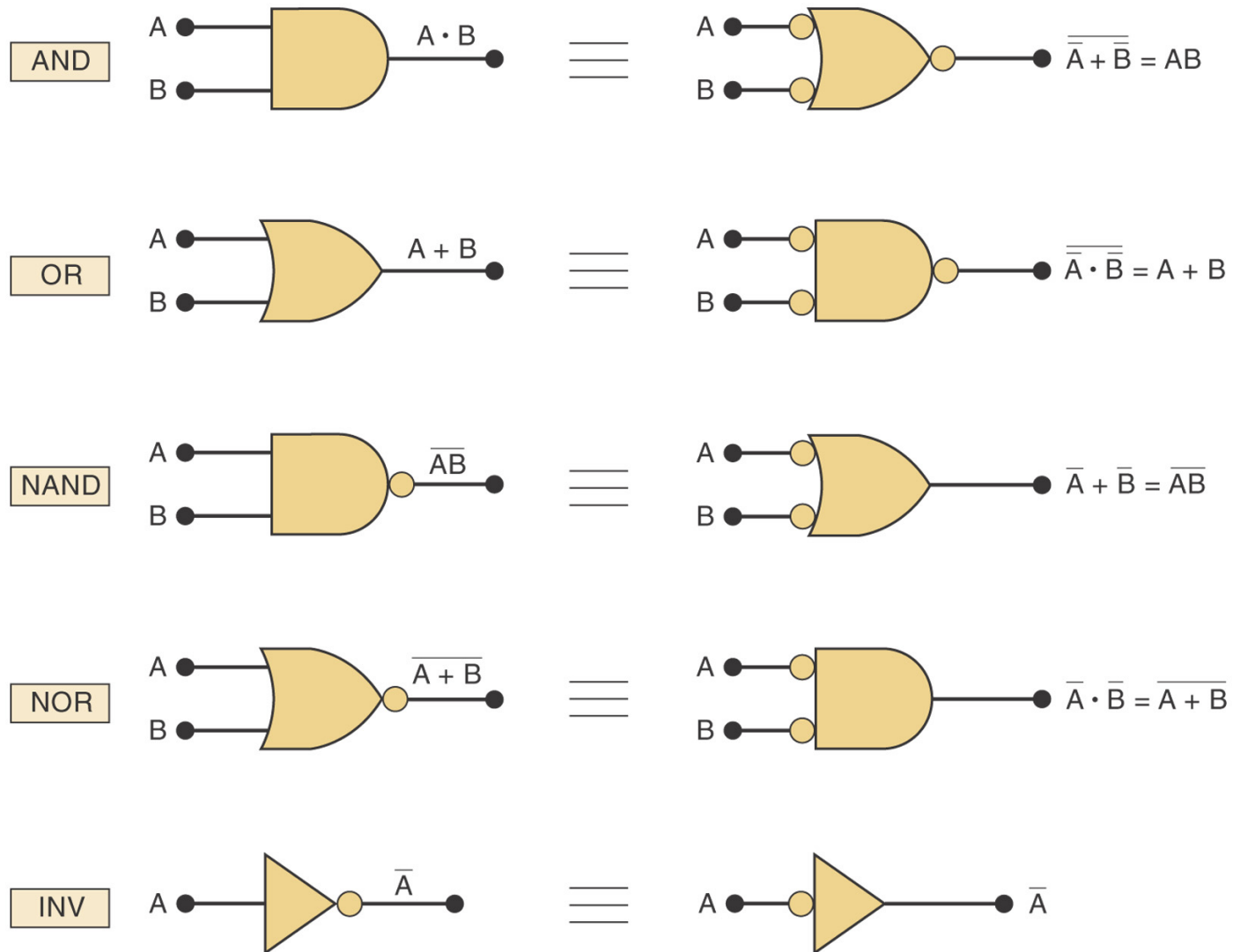
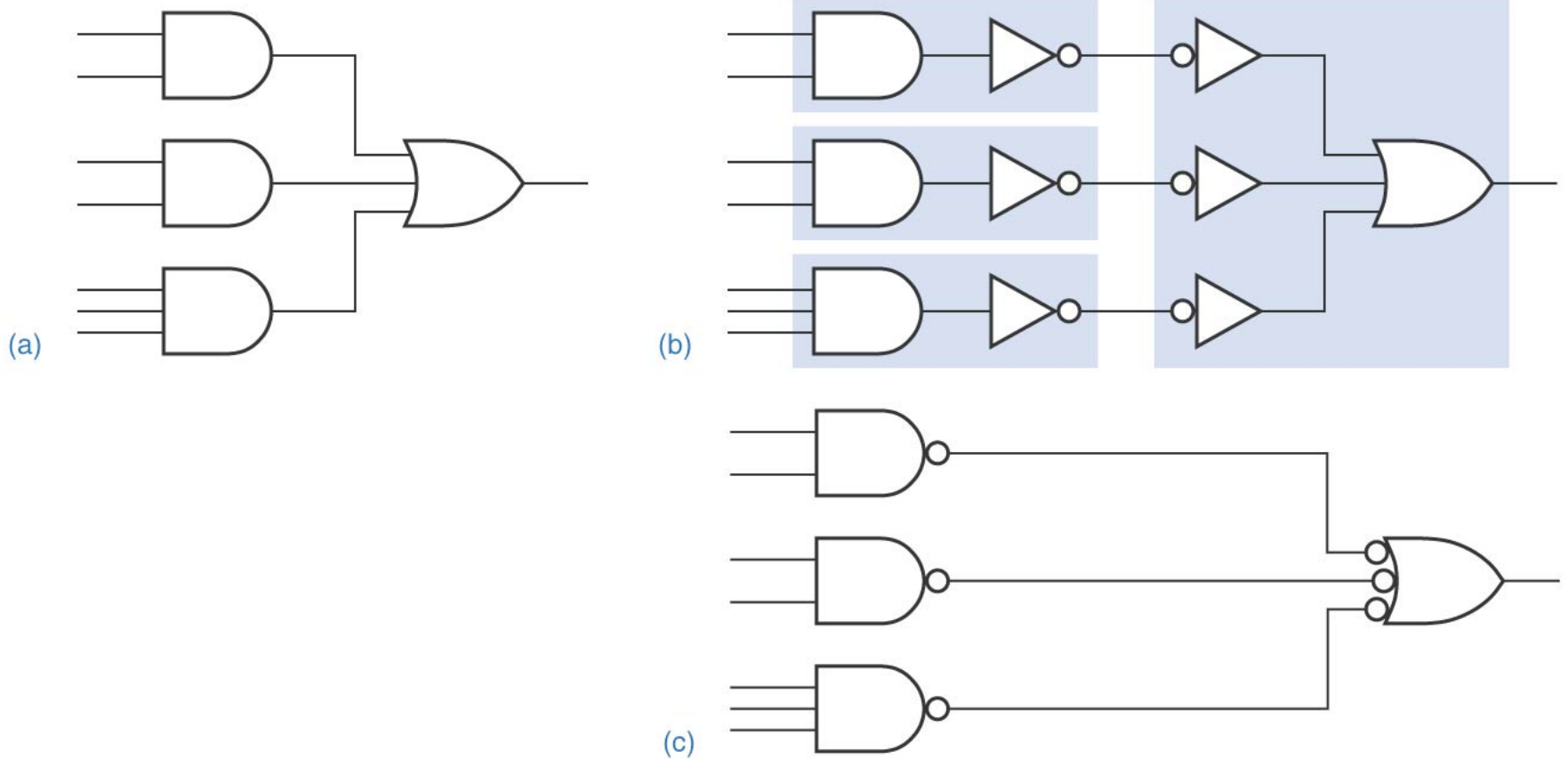


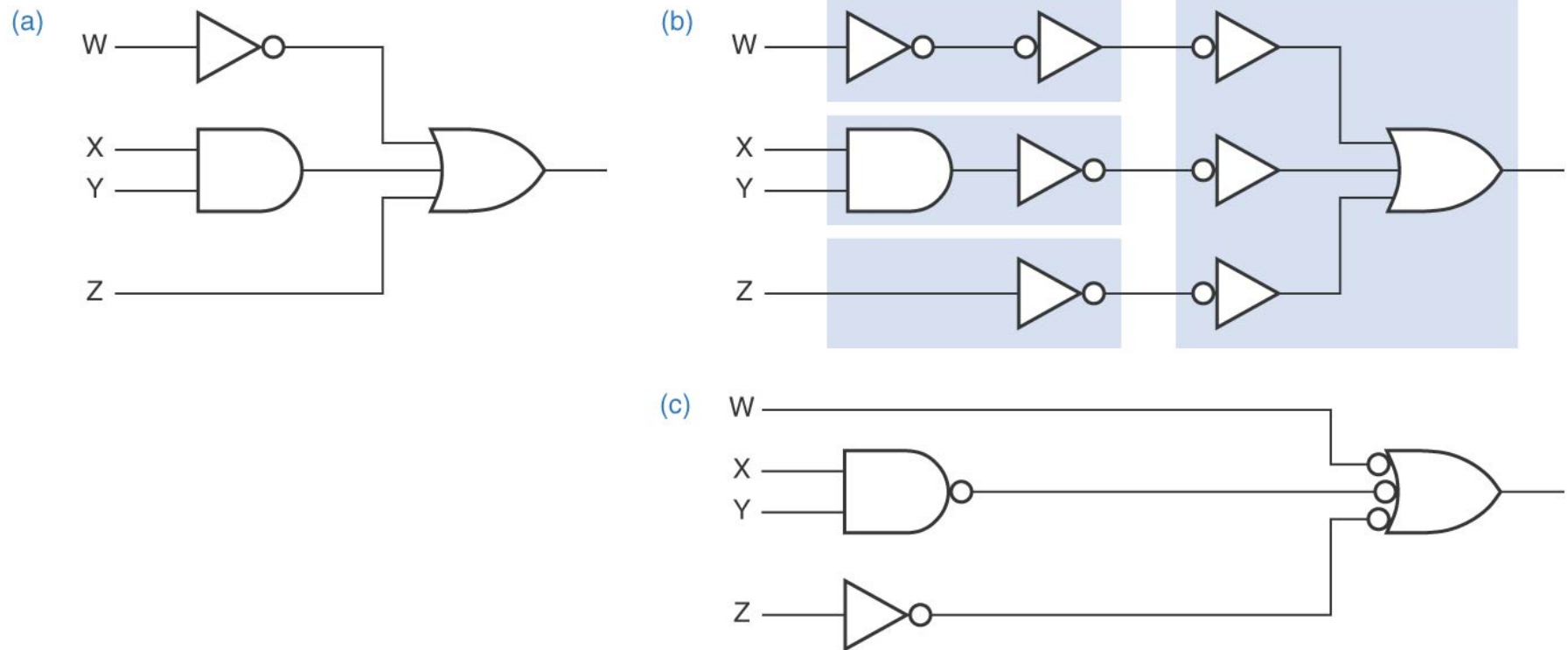
Fig. 3-33 (Tocci, 10th ed. Pg. 87)

Examples



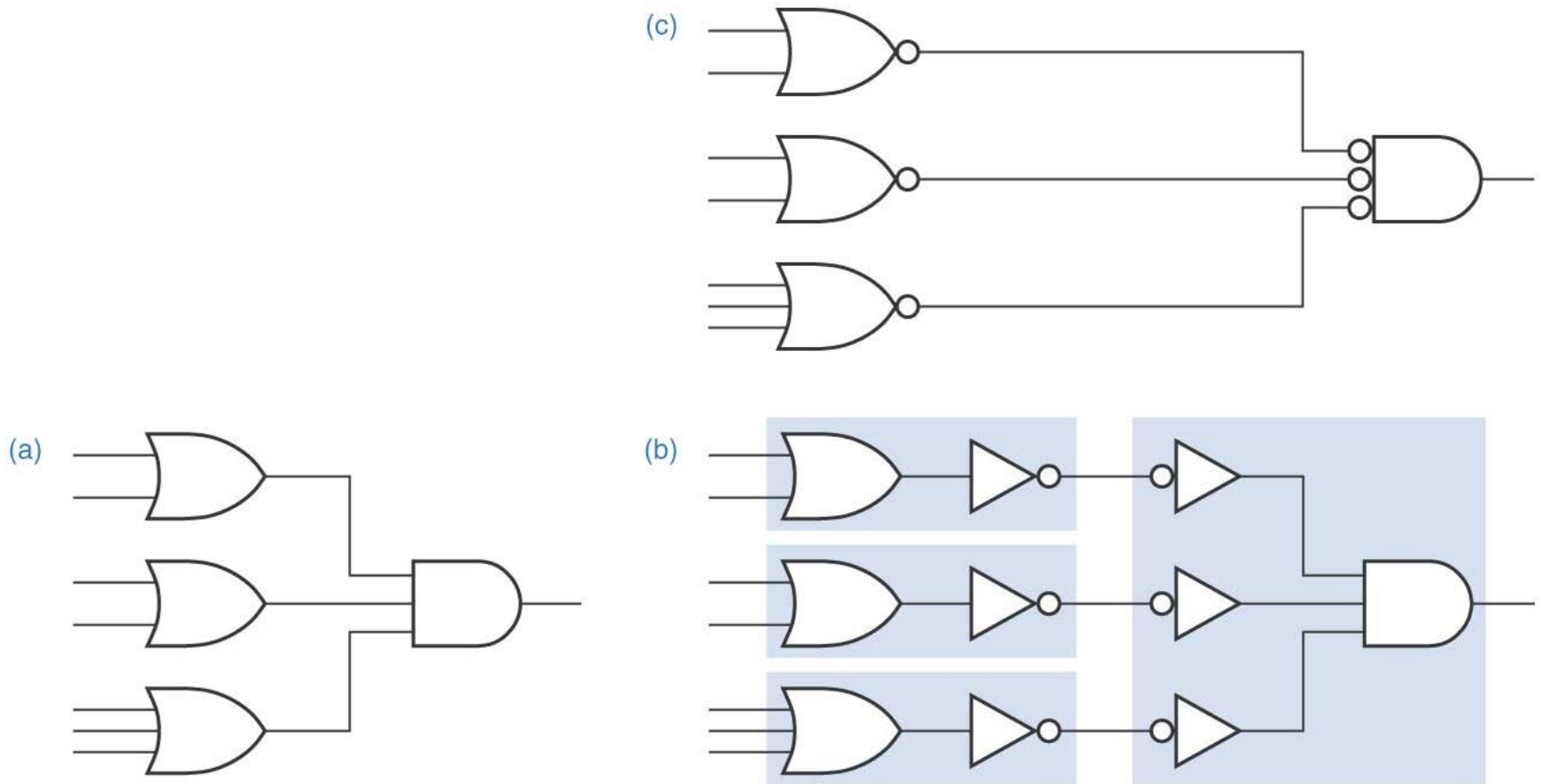
From *Digital Design: Principles and Practices*, Fourth Edition, John F. Wakerly, ISBN 0-13-186389-4.
©2006, Pearson Education, Inc., Upper Saddle River, NJ. All rights reserved.

Fig. 3-46



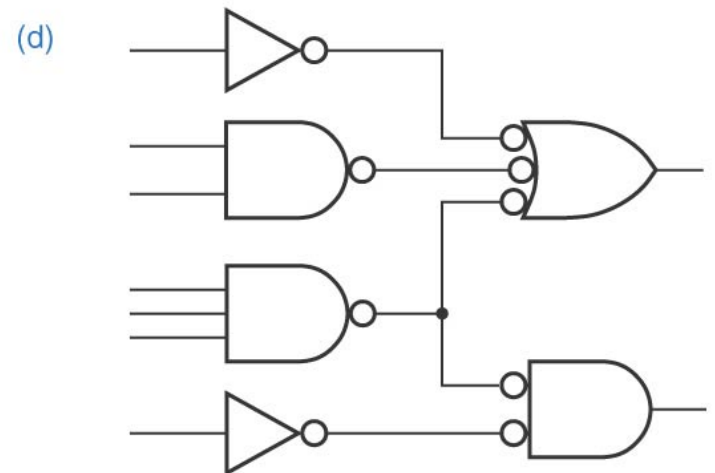
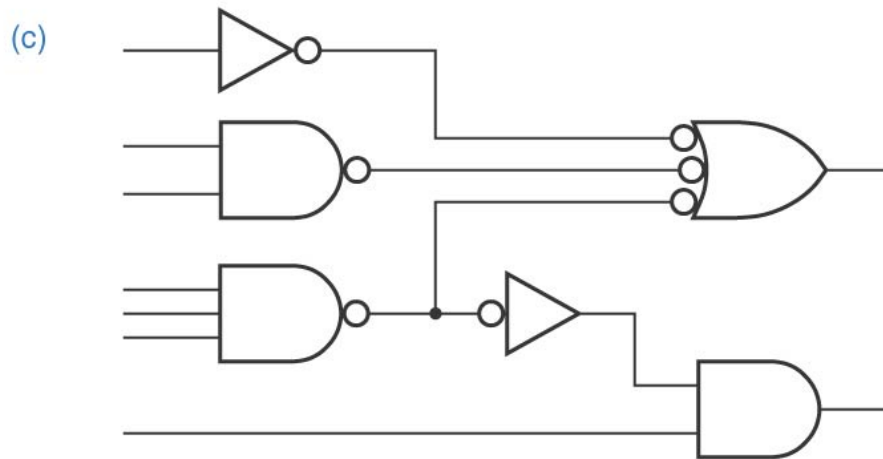
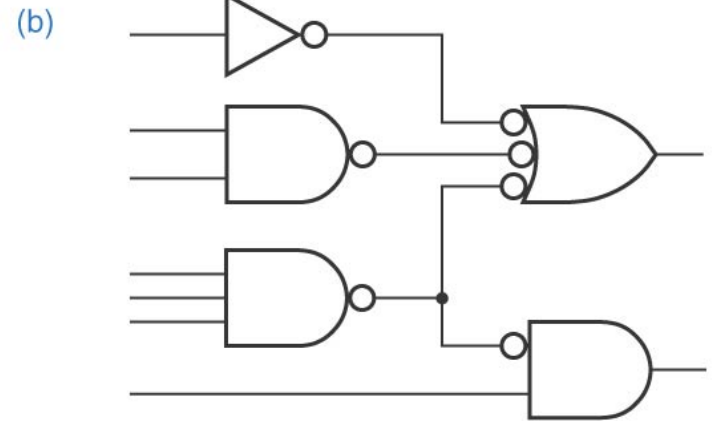
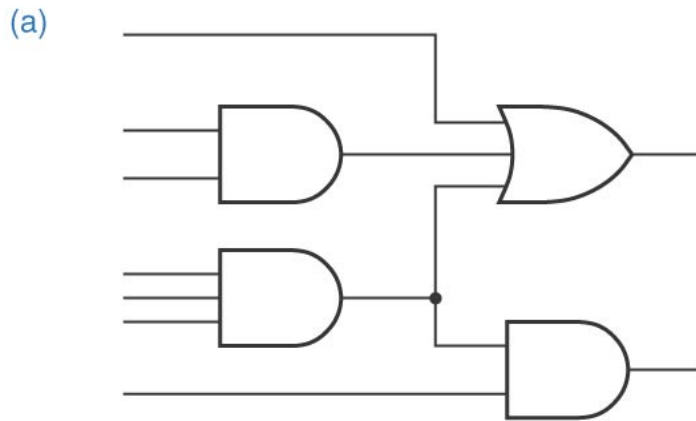
From *Digital Design: Principles and Practices*, Fourth Edition, John F. Wakerly, ISBN 0-13-186389-4.
 ©2006, Pearson Education, Inc., Upper Saddle River, NJ. All rights reserved.

Fig. 3-47



From *Digital Design: Principles and Practices*, Fourth Edition, John F. Wakerly, ISBN 0-13-186389-4.
 ©2006, Pearson Education, Inc., Upper Saddle River, NJ. All rights reserved.

Fig. 3-48



From *Digital Design: Principles and Practices*, Fourth Edition, John F. Wakerly, ISBN 0-13-186389-4.
 ©2006, Pearson Education, Inc., Upper Saddle River, NJ. All rights reserved.

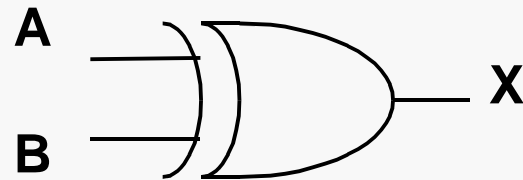
Fig. 3-49

Exclusive-OR gate

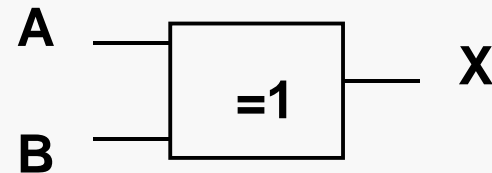
Ex-OR (XOR)

$$X = AB' + A'B$$

$$X = A \oplus B$$



A	B	X
0	0	0
0	1	1
1	0	1
1	1	0



**IEEE
symbol**

**Different
from OR**

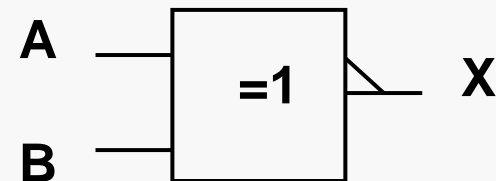
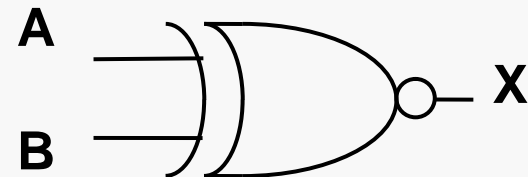
Exclusive-NOR gate

Ex-NOR (XNOR)

$$X = AB + A'B'$$

$$X = (A \oplus B)'$$

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1



**IEEE
symbol**

Application of XOR

- **Bit-wise comparator**
 - **output is 1 if the two multi-bit inputs are different**

XOR with multiple inputs

- Essentially an odd-function generator
- Output is 1 if there is an odd number of 1's among all the inputs
- E.g. for 3-input XOR, the output is 1 if there are 1 or 3 bits of 1 among the inputs
- $A \oplus B \oplus C = (A \oplus B) \oplus C$
- $= A \oplus (B \oplus C)$

Logic devices

- Different ways to create a physically functioning logic circuit
- Examples: use standard logic integrated circuits (ICs), application-specific ICs (ASICs), programmable logic devices
- Small-scale integrated logic devices: AND, OR, NOT, NAND, NOR, XOR, XNOR
- We will use some of these in lab experiment 1

**Plastic or ceramic
protective casing**

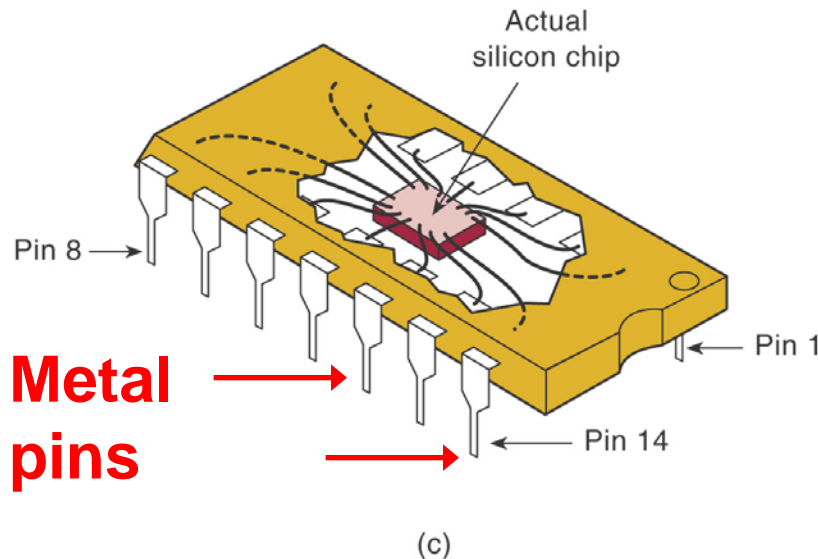
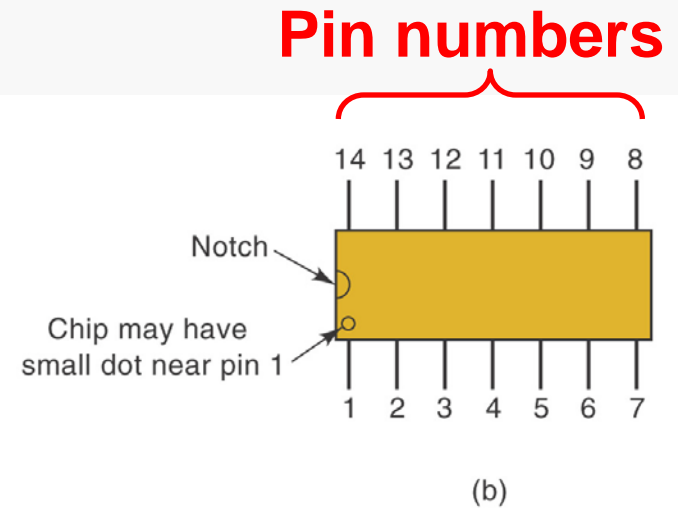
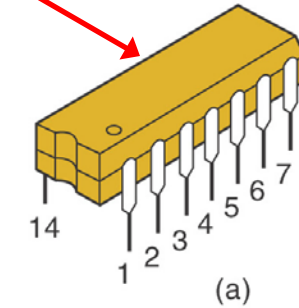
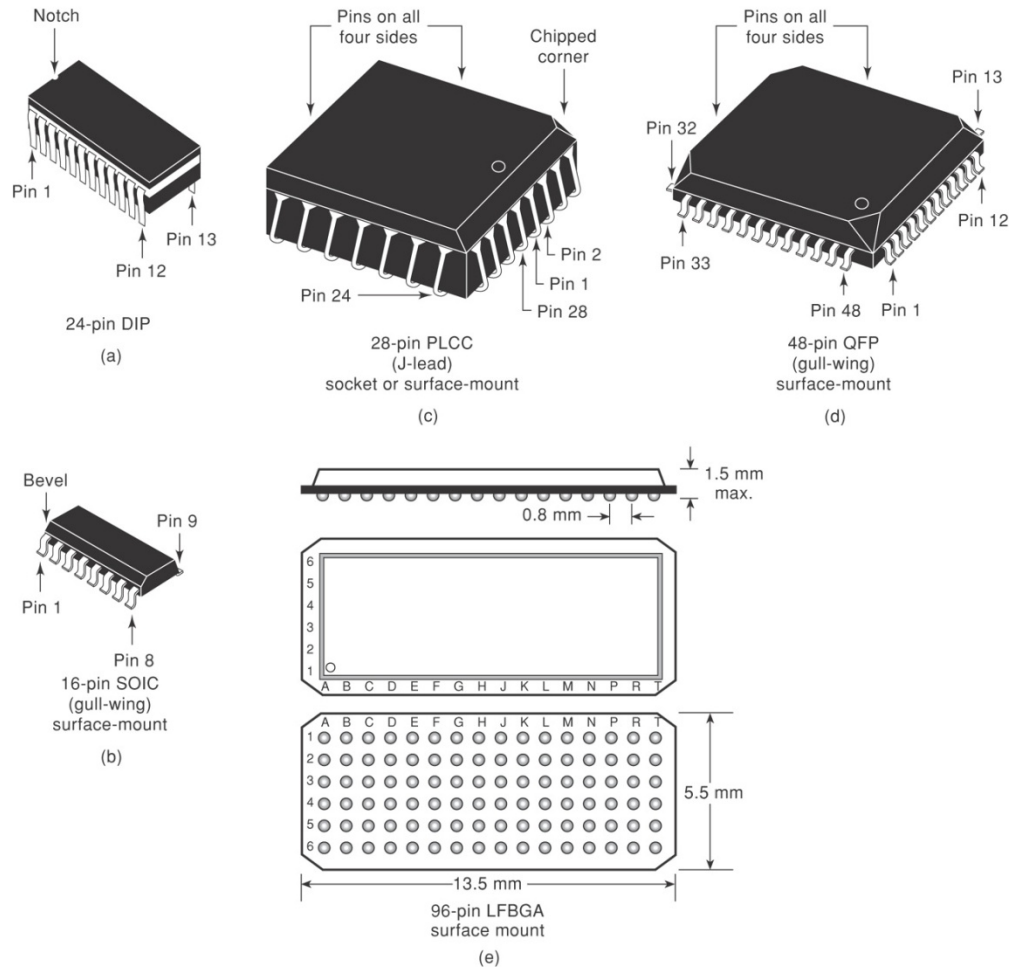


Figure 4.29: (Tocci 10th Ed) Dual-in-line Package

Common IC packaging

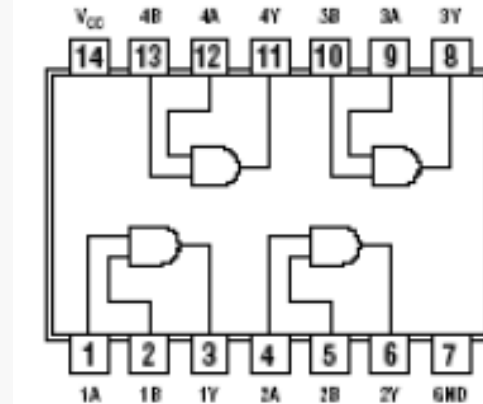
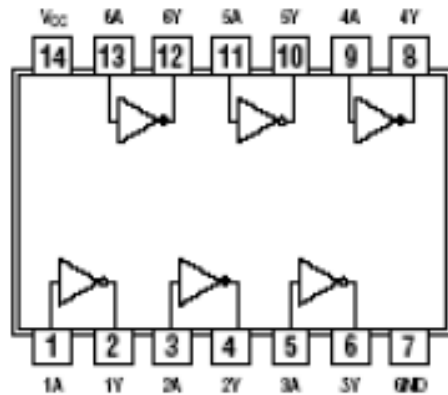


Source: Tocci, 10th ed., pg. 496

Logic circuit connections

- Implement $Y = AB'$

7404
Hex-NOT



7408
Quad-AND

Circuit connection diagram