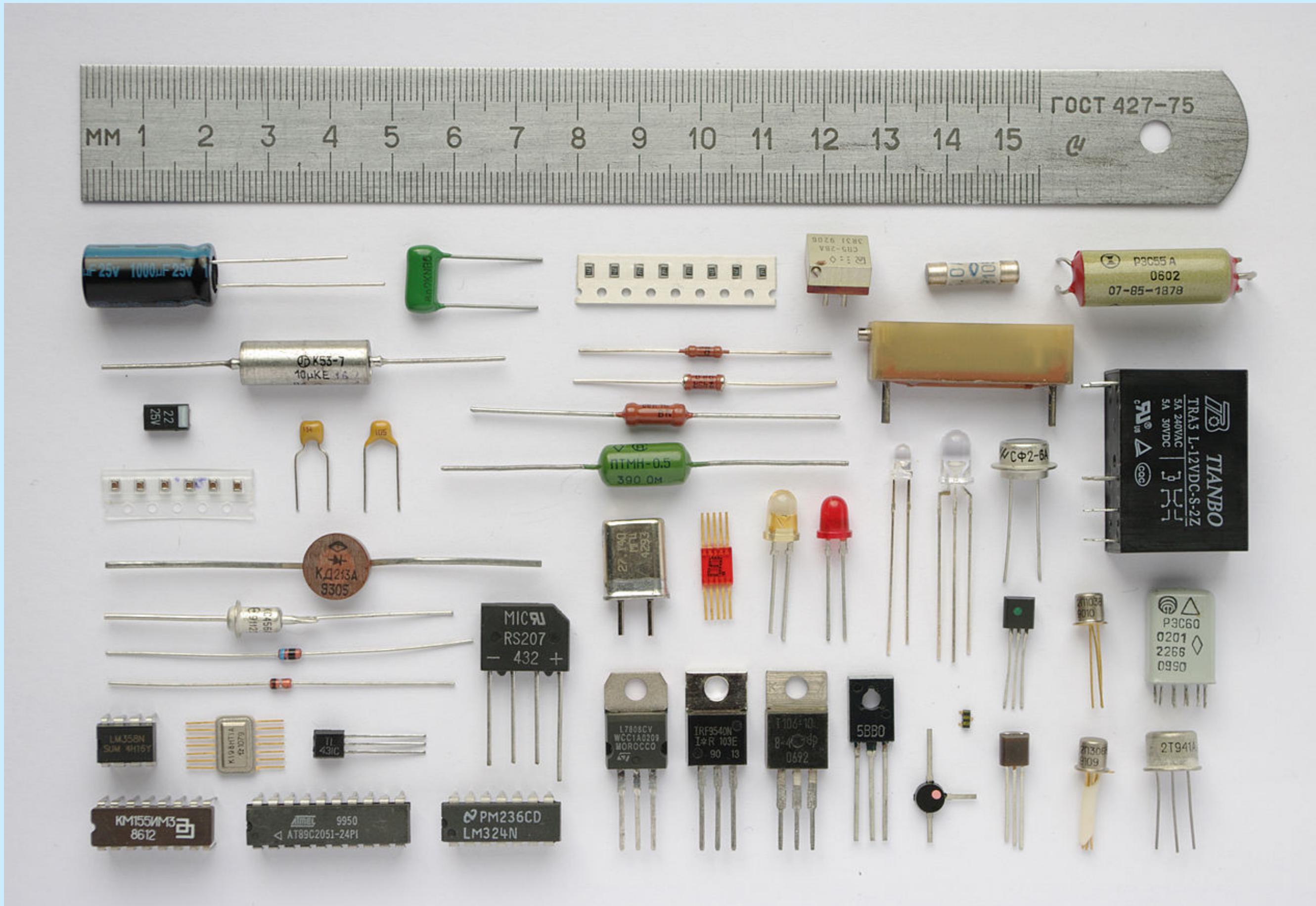
Lecture 1 summary



https://en.wikipedia.org/wiki/Electronic_component#/media/File:Componentes.JPG

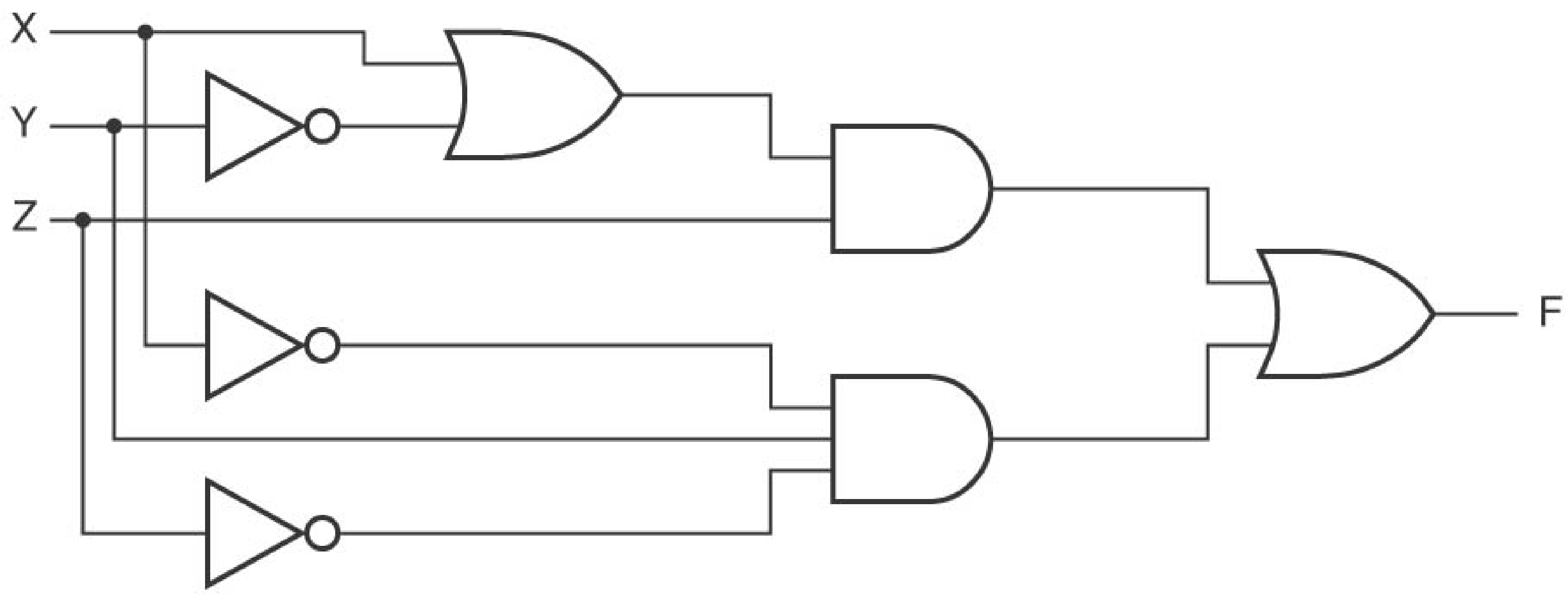
L1: Key concepts

- **Boolean Constants: 0 and 1**
- **Boolean variables, logic signals**
- **Logic levels represented by voltage**
- **Basic logic gates: AND, OR, NOT**
- **Logic symbols, logic expression, truth table**
- **Logic circuits has inputs and outputs, need power supply to operate**

L1: Key concepts (cont)

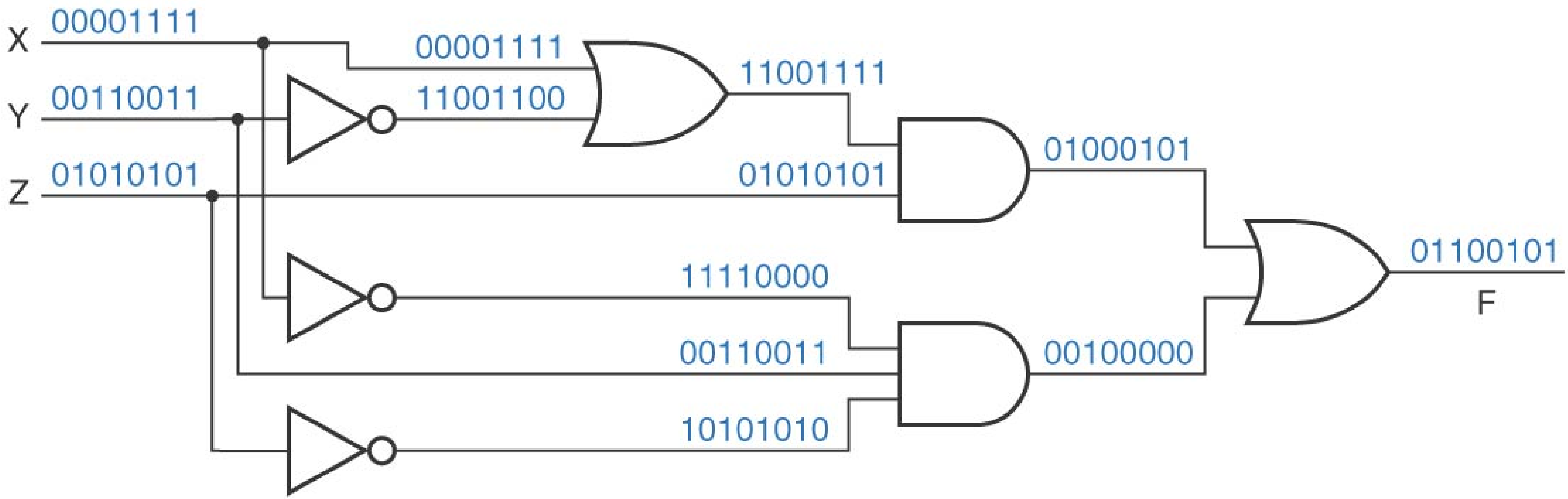
- **An output can connect to one or more inputs**
- **Outputs should not be connected together unless one is very sure**
- **Timing diagram/waveform**
- **A logic circuit can be described by truth table, logic expression, or circuit diagram**
- **Evaluation of logic expression: order of precedence**

Example:
describe by circuit diagram



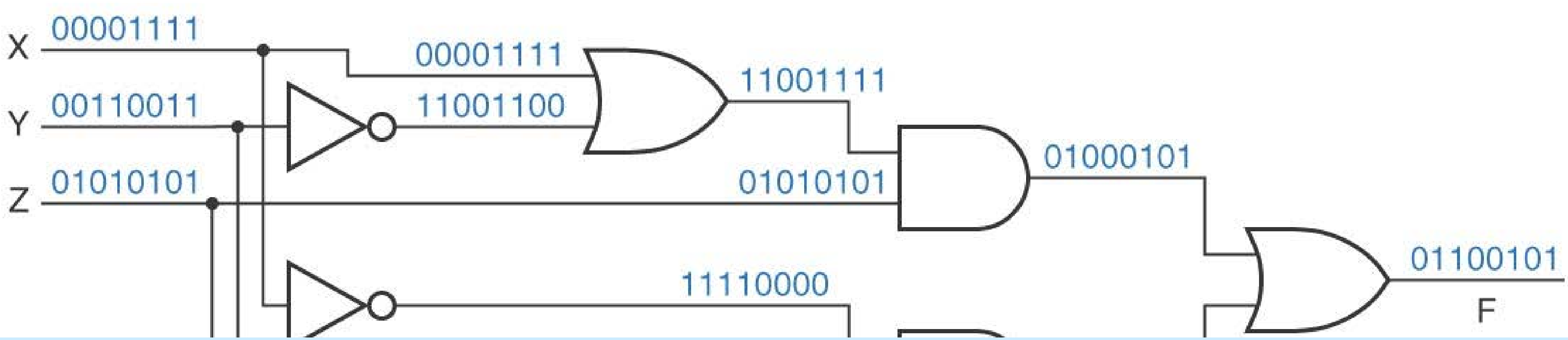
From *Digital Design: Principles and Practices*, Fourth Edition, John F. Wakerly, ISBN 0-13-186389-4.

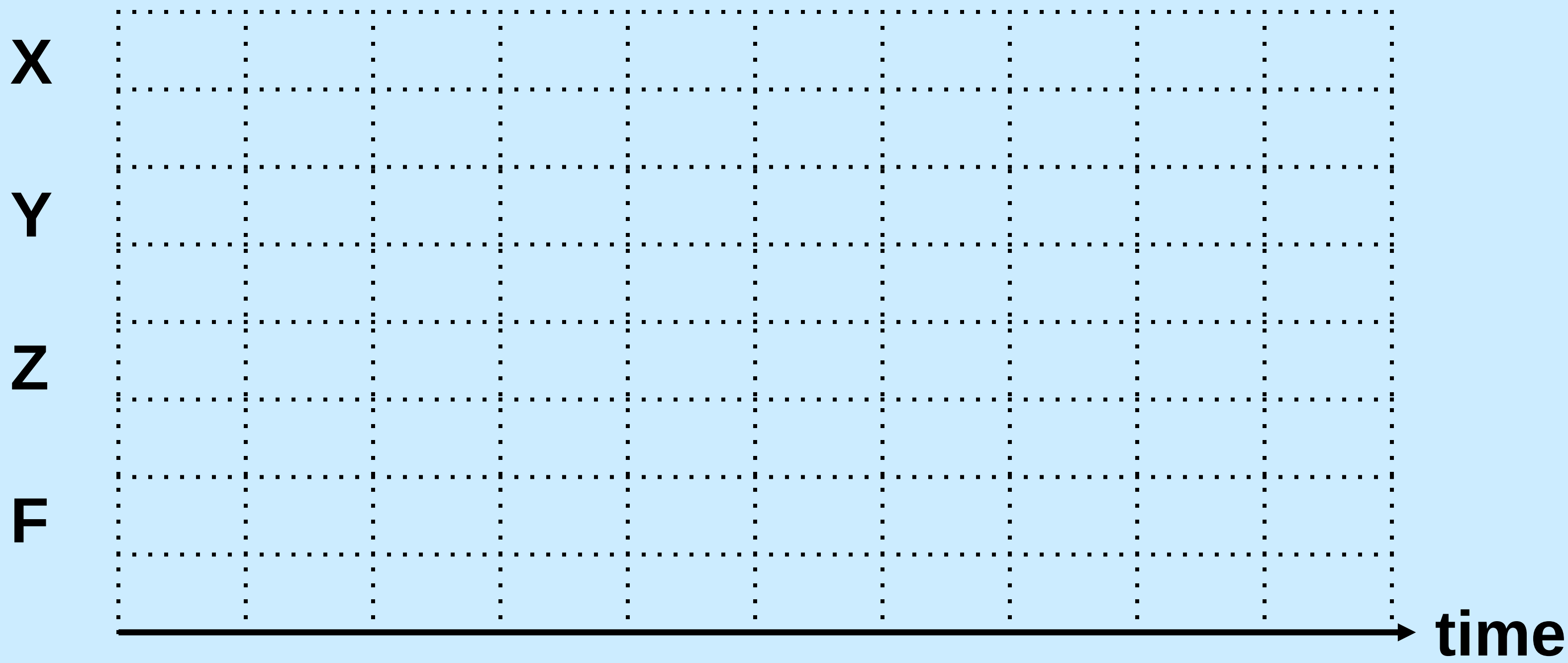
Example:
evaluate output from inputs



From *Digital Design*: *Principles and Practices*, Fourth Edition, John F. Wakerly, ISBN 0-13-186389-4.

Sketch timing waveforms





Example: describe by truth table

Row	X	Y	Z	F
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	0
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1

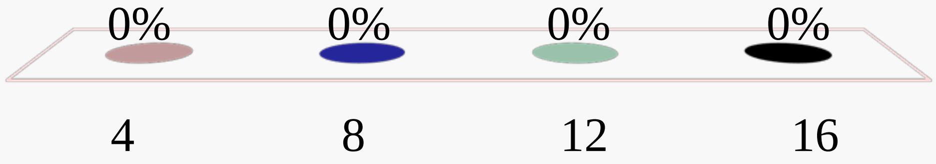
How many different 2-input truth tables can be constructed? Assume each has 1 output.

A. 4

B. 8

C. 12

D. 16



Inputs		Possible outputs					
a	b	x					
0	0	0	0	0	• • •	1	1
0	1	0	0	0		1	1
1	0	0	0	1		1	1
1	1	0	1	0		0	1

2^n

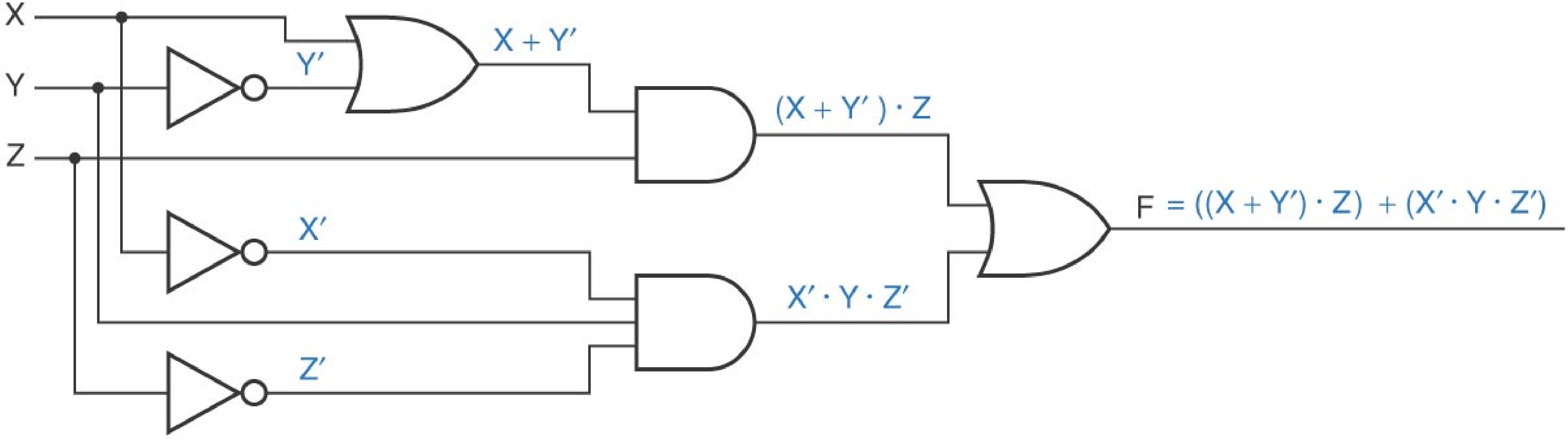
2^(2^n)

Commonly-used 2-input logic gates:

- **AND, NAND**
- **OR, NOR**
- **XOR, XNOR**

Example:
describe by logic expression

F = (X + Y’) Z + X’ Y Z’



From *Digital Design: Principles and Practices*, Fourth Edition, John F. Wakerly, ISBN 0-13-186389-4.

How about describe by timing diagram?

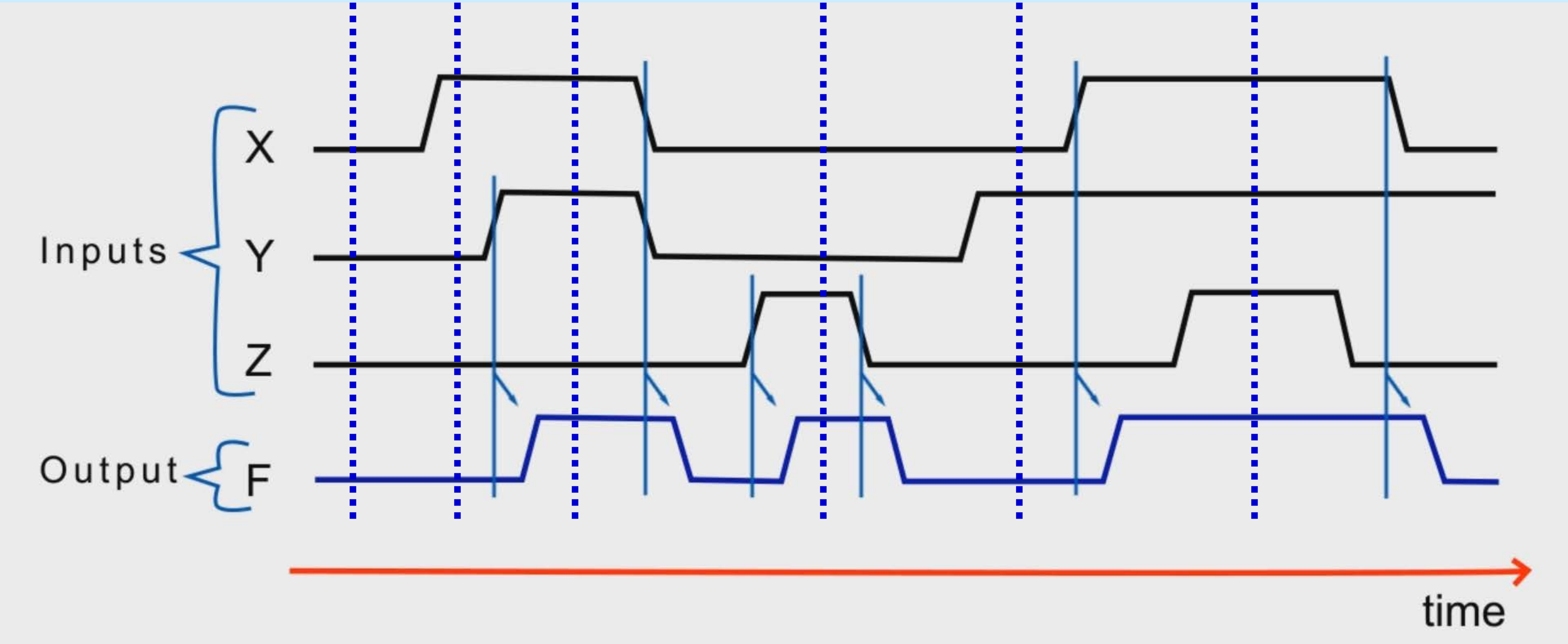


Fig. 3.17 (taken from Wakerly)

Truth table

inputs			output
X	Y	Z	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	?
1	0	0	0
1	0	1	?
1	1	0	1
1	1	1	1

Lecture 2: Key concepts

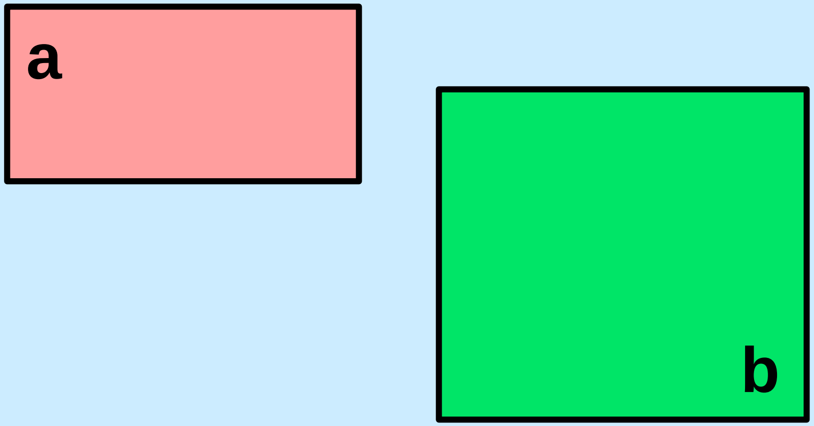
- **Boolean theorems: single and multivariable, DeMorgan’s theorems**
- **Theorems are required for algebraic manipulation and simplification**
- **NOR and NAND gates: universal gates can replace basic logic gates AND, OR, NOT**

Boolean theorems

- **Essential to know and apply the theorems**
- **For those interested in more proofs on Boolean theorems (optional):**
  - http://www.electrical4u.com/boolean-algebra-theorems-and-laws-of-boolean-algebra/
  - http://mines.humanoriented.com/410/books/boolean_algebra.pdf

Absorption laws

Absorption laws:

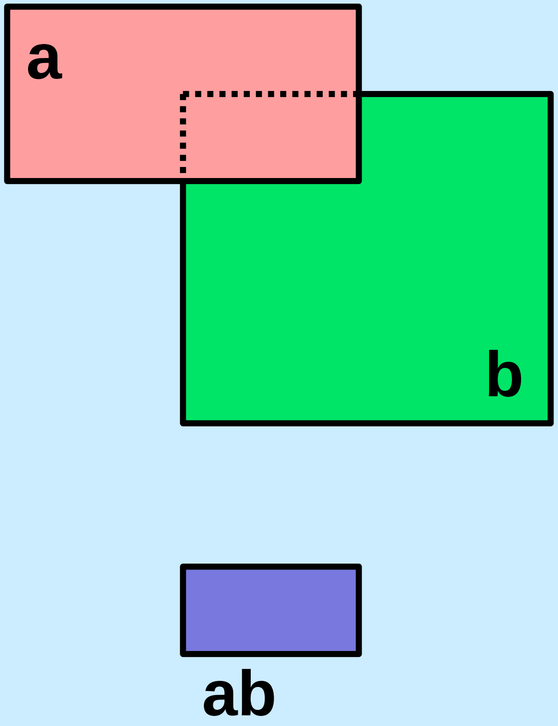


a + ab = a

Absorption laws (cont)

Absorption laws:

$a + ab = a$



ab is absorbed in a

Simple example for illustration

Let

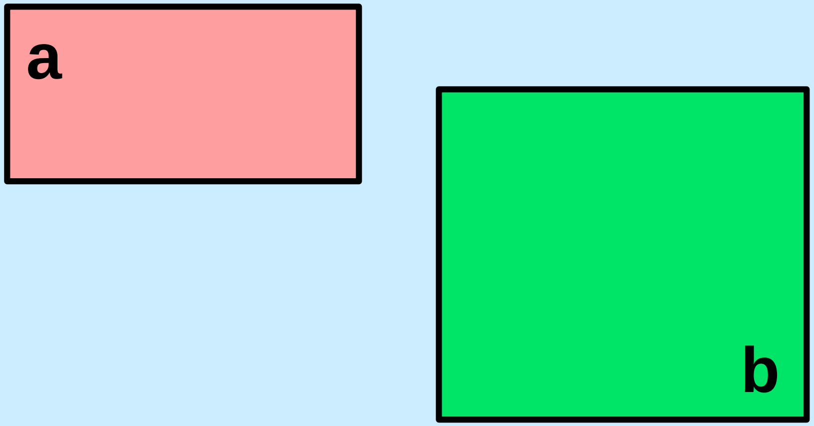
- A=1 means a person <u>likes</u> apples (A=0 means a person <u>dislikes</u> apples)
- B=1 means a person <u>likes</u> oranges (B=0 means a person <u>dislikes</u> oranges)

$$A + AB = A\,(1+B) = A(1) = A$$

A person who likes apples (A=1) may also like oranges (B=1)

Absorption laws (cont)

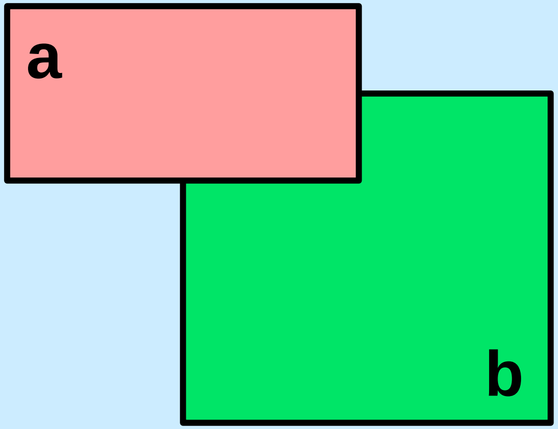
Absorption laws:



a + a’b = a + b

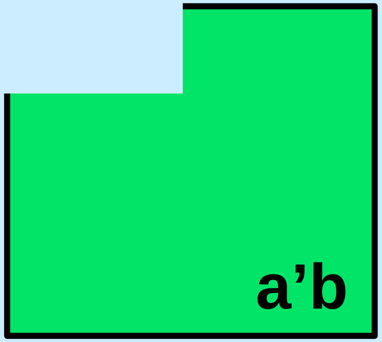
Absorption laws (cont)

Absorption laws:



a + a’b = a + b

a’b is absorbed in b



Simple example for illustration

Let

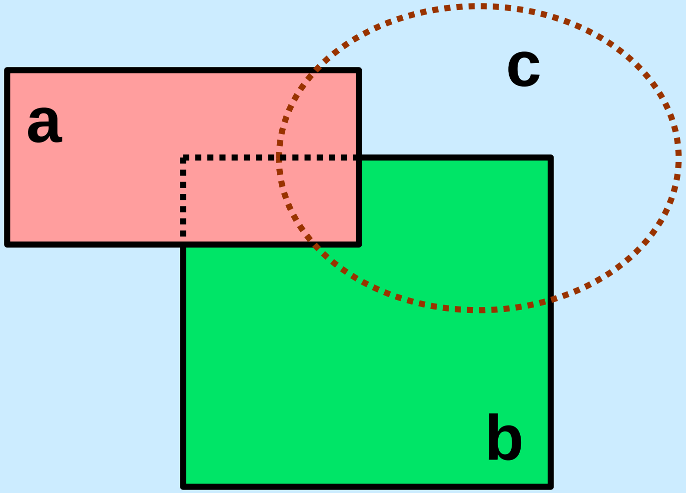
- A=1 means a person likes apples (A=0 means a person dislikes apples)
- B=1 means a person likes oranges (B=0 means a person dislikes oranges)

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

A person who likes apples or oranges may like either one or both

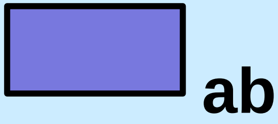
Consensus law

Consensus law:



bc = (ab + a'b) c

= abc + a'bc



ab + a'c + bc = ab + a'c

**A part of bc is absorbed in ab,
the other part is absorbed in a'c**

Simple example for illustration

Let

- A=1 means a person <u>likes</u> apples (A=0 means a person <u>dislikes</u> apples)
- B=1 means a person <u>likes</u> oranges (B=0 means a person <u>dislikes</u> oranges)
- C=1 means a person <u>likes</u> pears (C=0 means a person <u>dislikes</u> pears)

Example (continue)

AB + A'C means a person who likes both apples and oranges, <u>or</u> likes pears but dislikes apples.

BC means a person who likes both oranges and pears.

- **AB + A'C + BC = AB + A'C**

a person who likes both oranges and pears may like or dislike apples

- **Important: it does NOT imply BC = 0**

DeMorgans theorems

V = (a + b + … + g)’ = a’ b’… g’

Let V=1 means need a visa

- Don’t need a visa (V=0) if one is from country a or b or c … or g

- Need a visa (V=1) if one is not from country a, and not from b, and not from c … and not from g

DeMorgans theorems (cont)

$$J = (a\ b\ c\ \ldots\ g)' = a' + b' + c' + \ldots + g'$$

Let J=1 means need an immunization jab

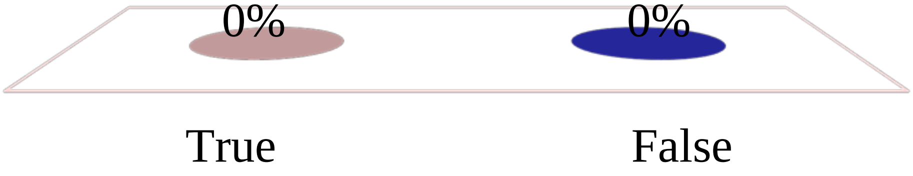
- Don't need a jab (J=0) if one is already immunized for viruses a and b and c … and g
- Need a jab (J=1) if one is not immunized from virus a, or not from b, or not from c … or not from g

$A(B+C)' = AB' + AC'$
True or false?

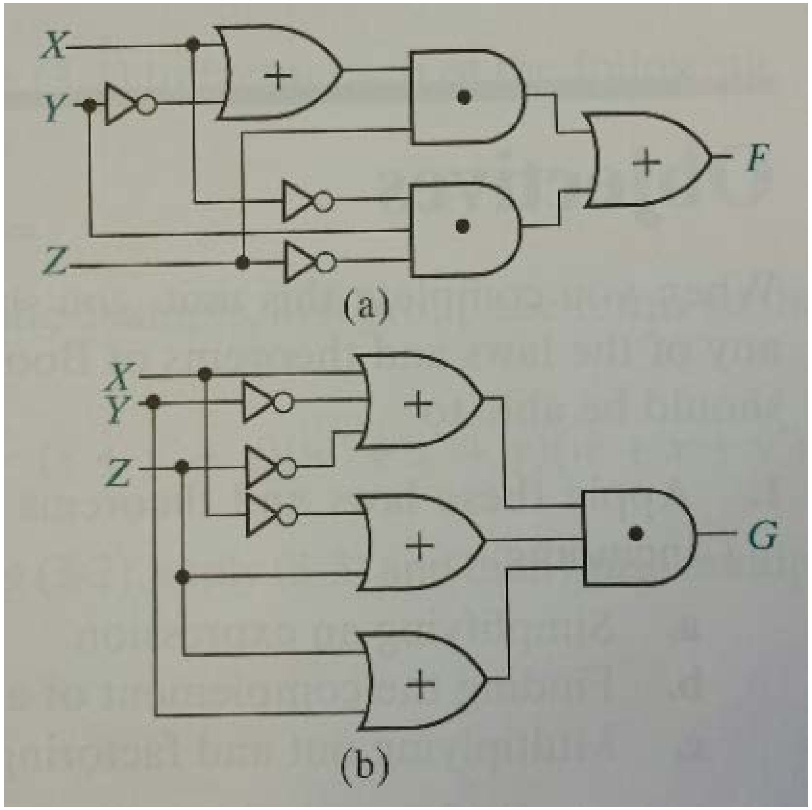
A. True

B. False

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



Are these two circuits algebraically equivalent?

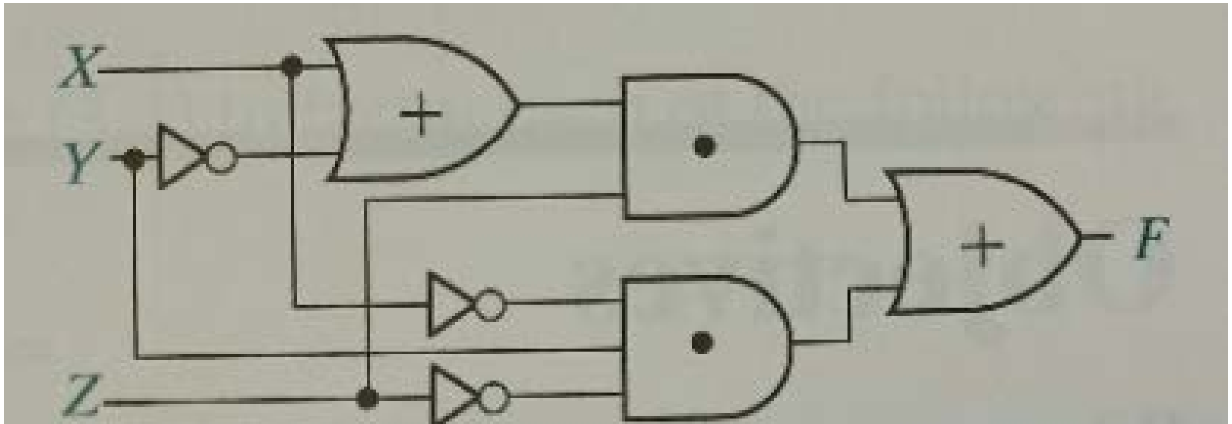


A. Yes

B. No

Source: Q2.30, p 59, Fundamentals of Logic Design, Roth & Kinney, Cengage Learning.

The answer is Yes.



(a)

F = (X+Y')Z + X'YZ'

$G = (X+Y'+Z')(X'+Z)(Y+Z)$

$= (X+Y'+Z')(X'Y + X'Z + YZ + ZZ)$

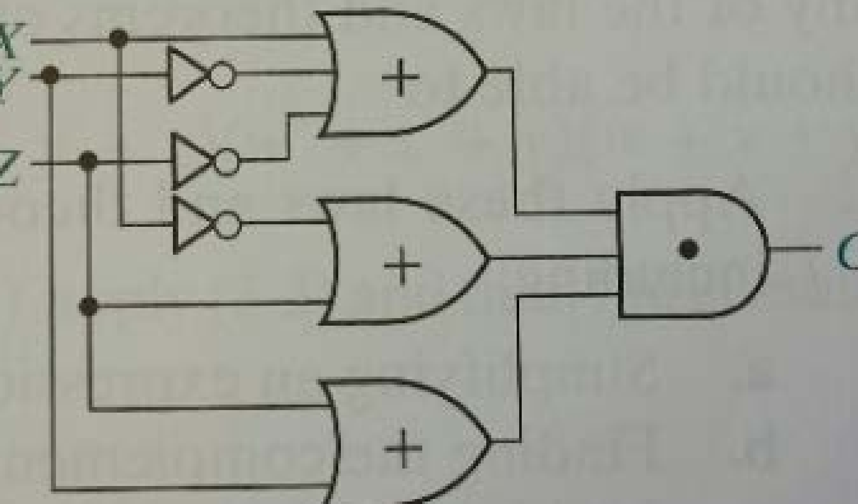
$= (X+Y'+Z')[X'Y + Z(X'+Y+1)]$

$= (X+Y'+Z')[X'Y + Z]$

$= XZ + Y'Z + X'YZ'$

$= (X+Y')Z + X'YZ'$

$= F$

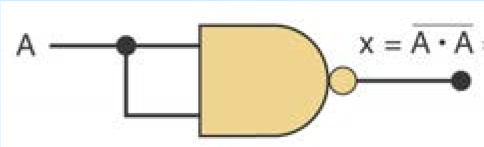
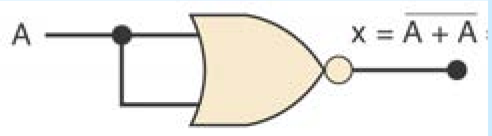
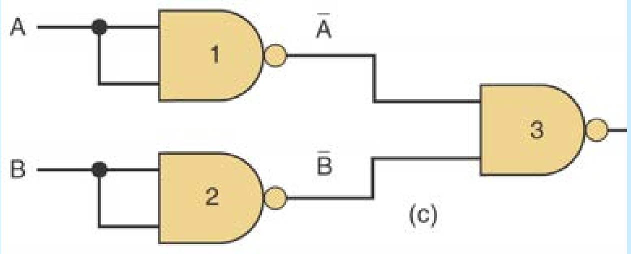
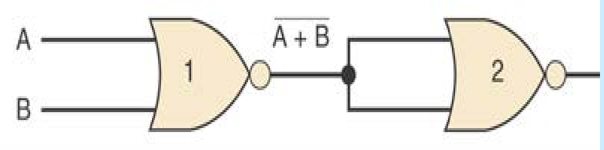
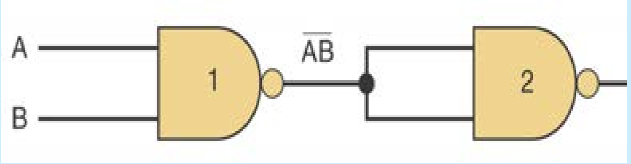
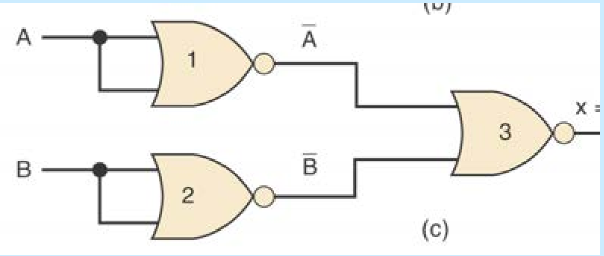


(b)

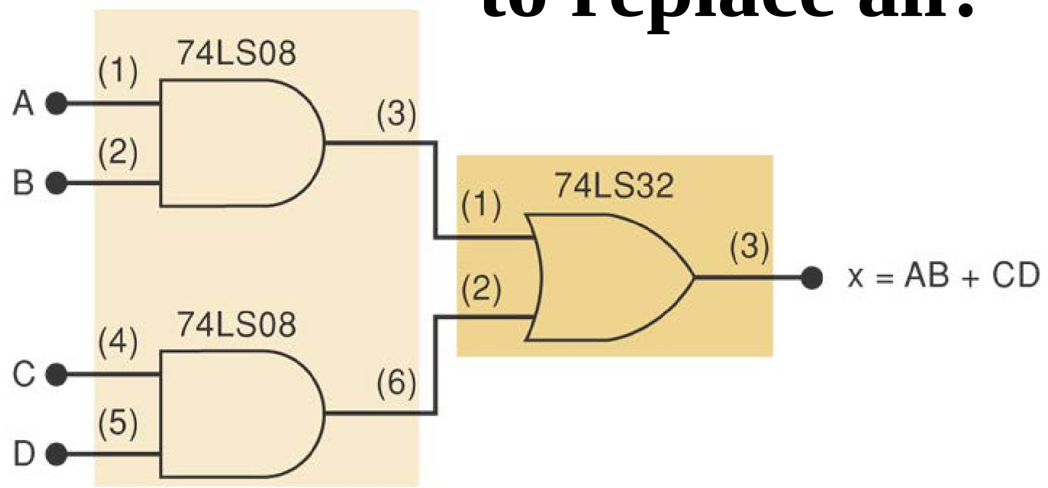
Replacing AND, OR, NOT with purely NAND, or purely NOR

Basic gate	NAND only	NOR only
NOT	$(XX)' = X'$ 1	$(X+X)' = X'$ 1
OR	$X+Y = [\ (X')\ (Y')\ ]'$ 3	$X+Y = [\ (X+Y)'\ ]'$ 2
AND	$XY = [\ (XY)'\ ]'$ 2	$XY = [\ (X') + (Y')\ ]'$ 3

Logic symbol approach

Basic gate	NAND only	NOR only
NOT		
OR		
AND		

How many NAND gates are needed in total to replace all?

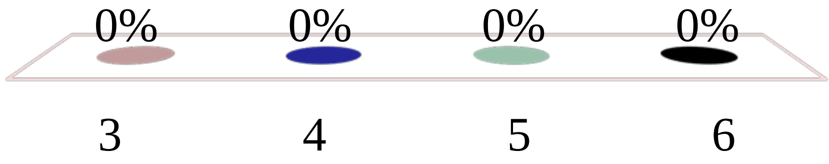


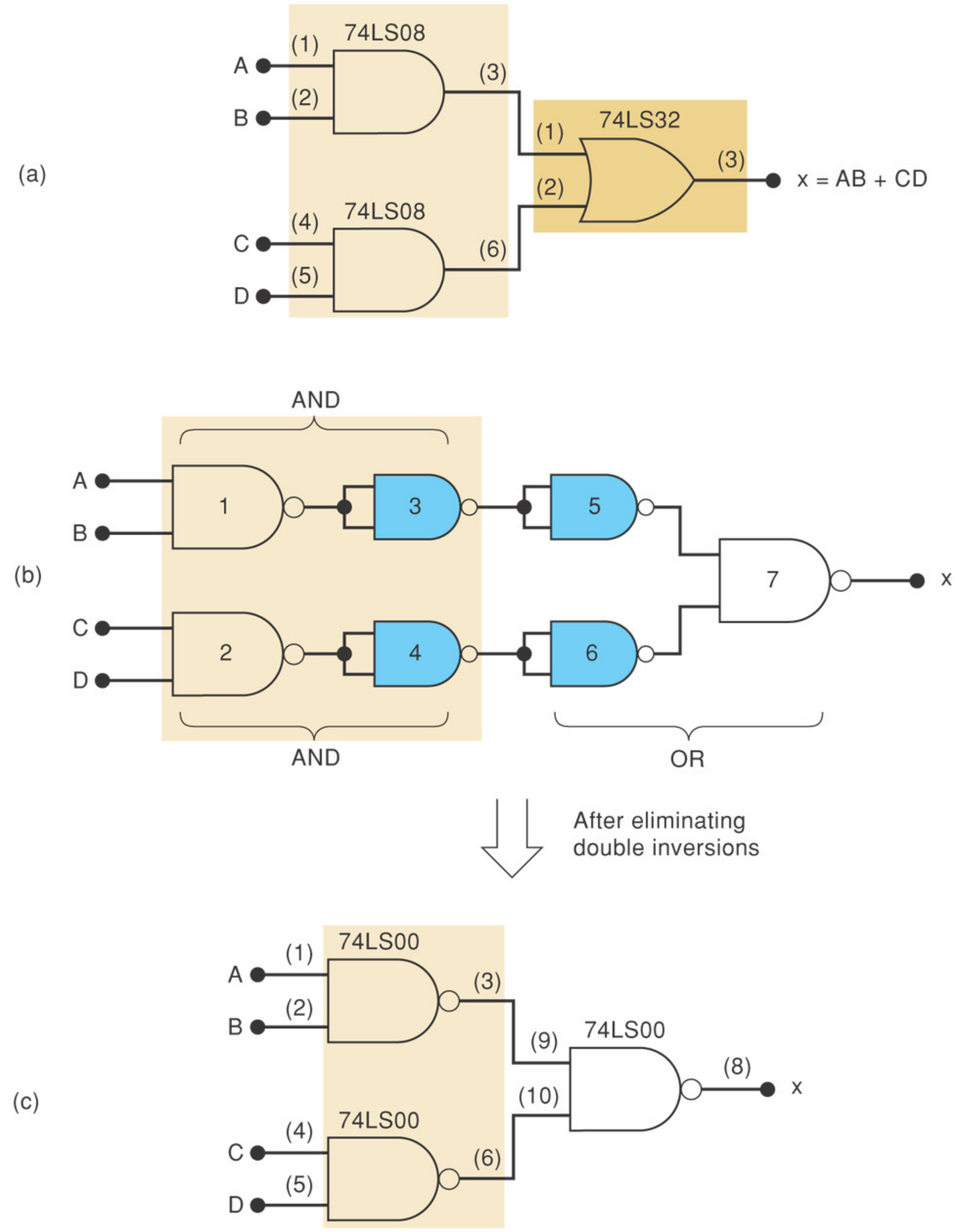
A. 3

B. 4

C. 5

D. 6





Example:

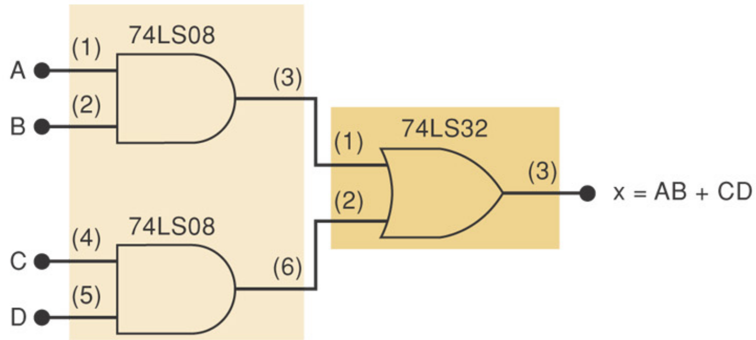
NAND gates replace AND, OR

By diagram

Figure 3-32

Tocci, Widmer, Moss. 10th ed.

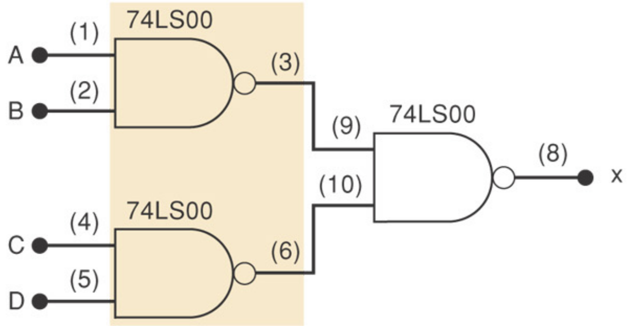
(a)



$$AB + CD = [\ (AB + CD)'\ ]'$$

$$= [\ (AB)'\ (CD)'\ ]'$$

(c)



Example:

NAND gates replace AND, OR

By Boolean expression

Figure 3-32

Tocci, Widmer, Moss. 10th ed.

Universal gates <u>always</u> reduce the number of gates used. True or false?

A. True
B. False

End of L1, L2 summary