



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**
- **Logic behavior described by truth table, timing diagram/waveform**
- **Logic circuit described by logic expression, circuit diagram**
- **Evaluation of logic expression: order of precedence**

Which concepts are not clear to you after viewing L1?

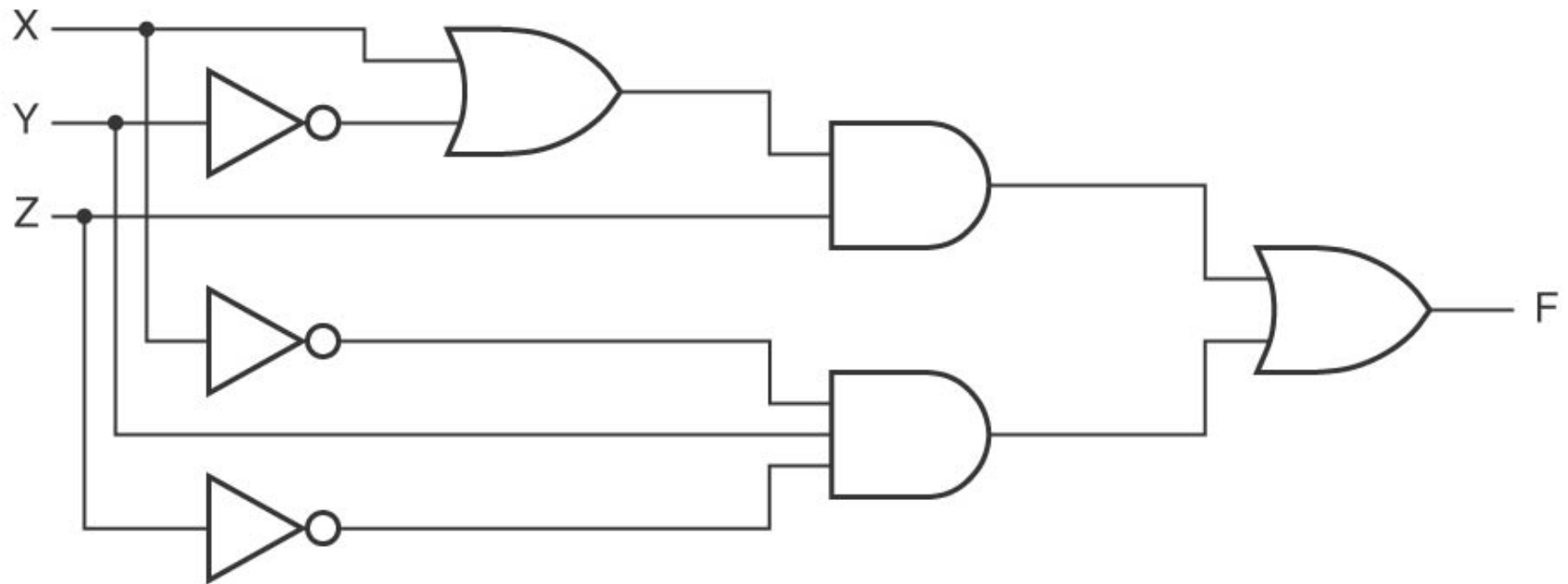
- A. AND, OR, NOT
- B. Truth table
- C. Logic expression
- D. Timing diagram
- E. Circuit diagram
- F. None

Example: logic behavior described by truth table

<i>Row</i>	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

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.

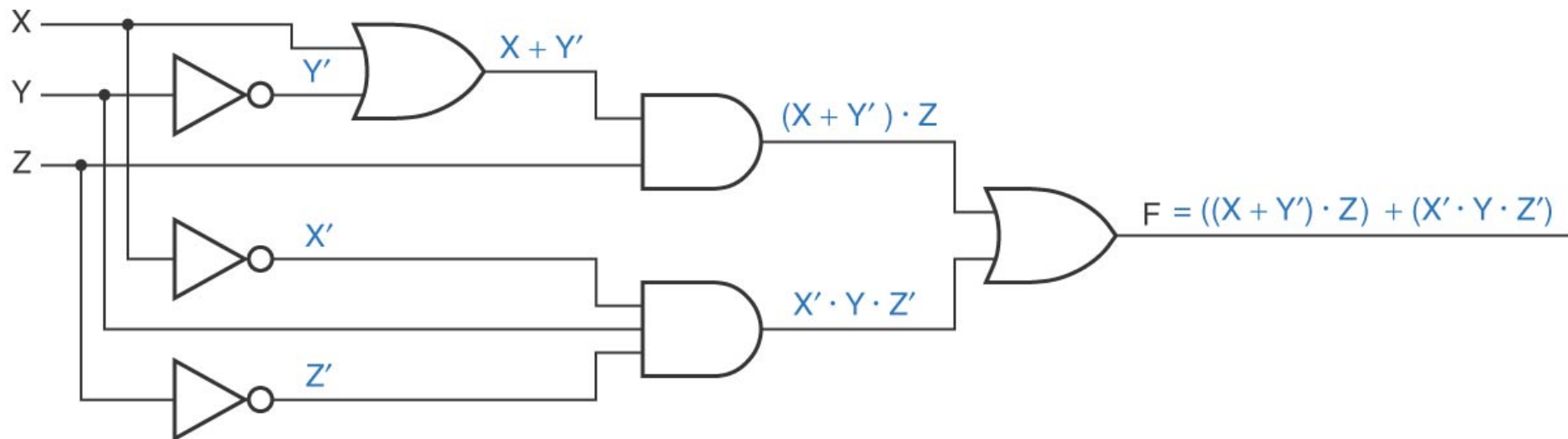
**Example: circuit
described by diagram**



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.

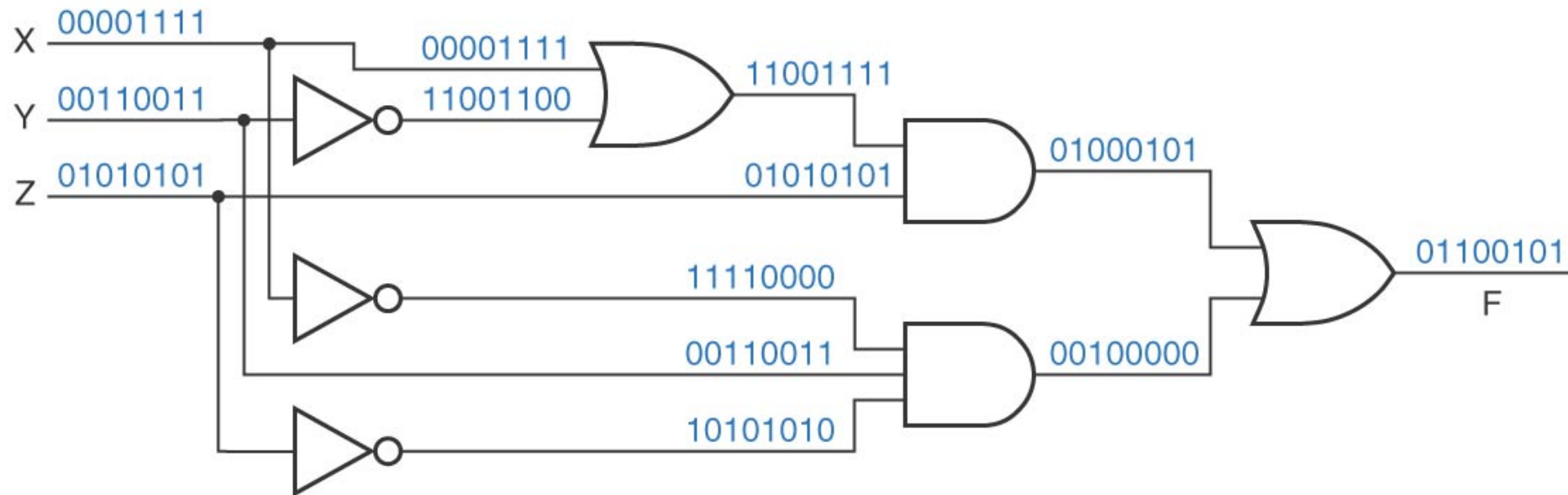
Example: circuit described by logic expression

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



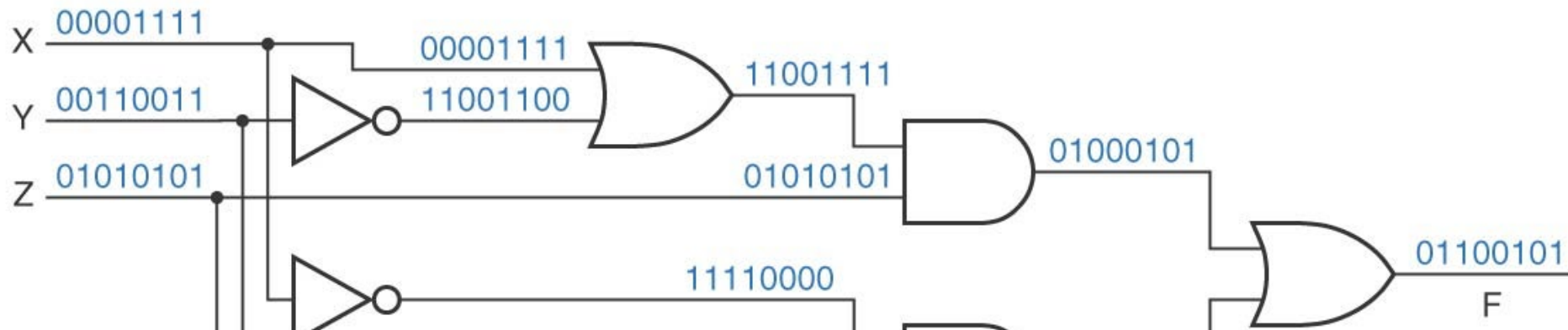
From *Digital Design: Principles and Practices*, Fourth Edition, John F. Wakerly, ISBN 0-13-186389-4.
©2006, Pearson Education, Inc., Upper Saddle River, NJ. All rights reserved.

Example: evaluate circuit output from inputs



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.

Sketch timing waveforms

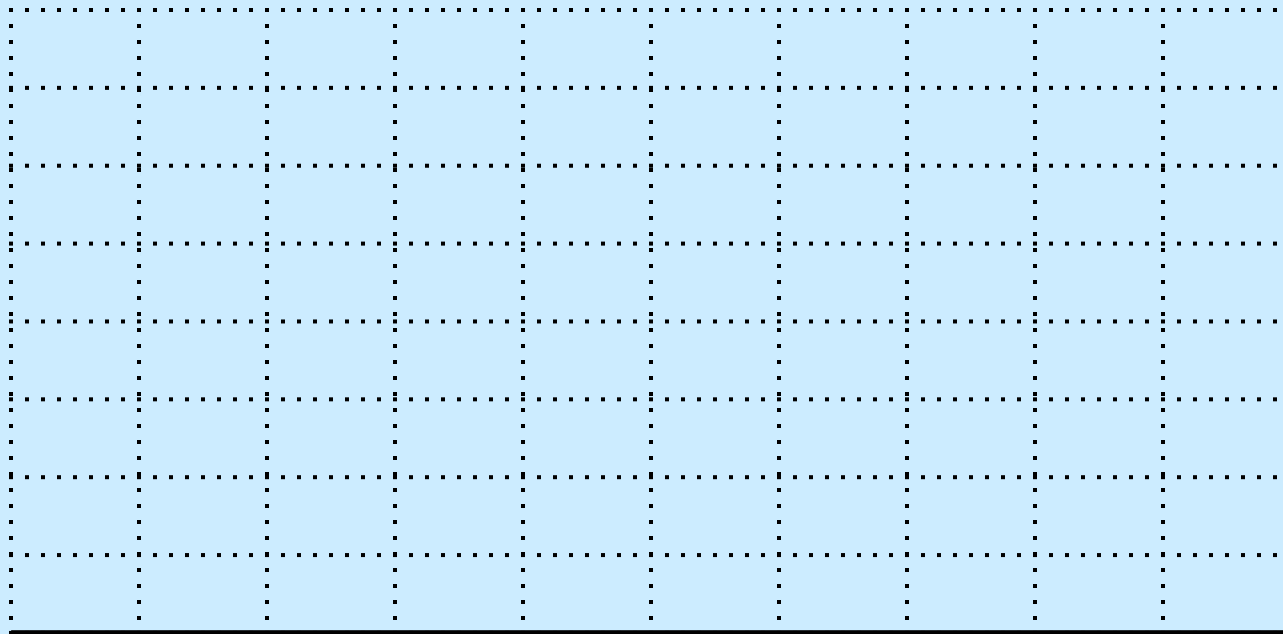


X

Y

Z

F



time
F12 - 9

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

A. 4

B. 8

C. 12

D. 16

Can the truth table be obtained from a timing diagram?

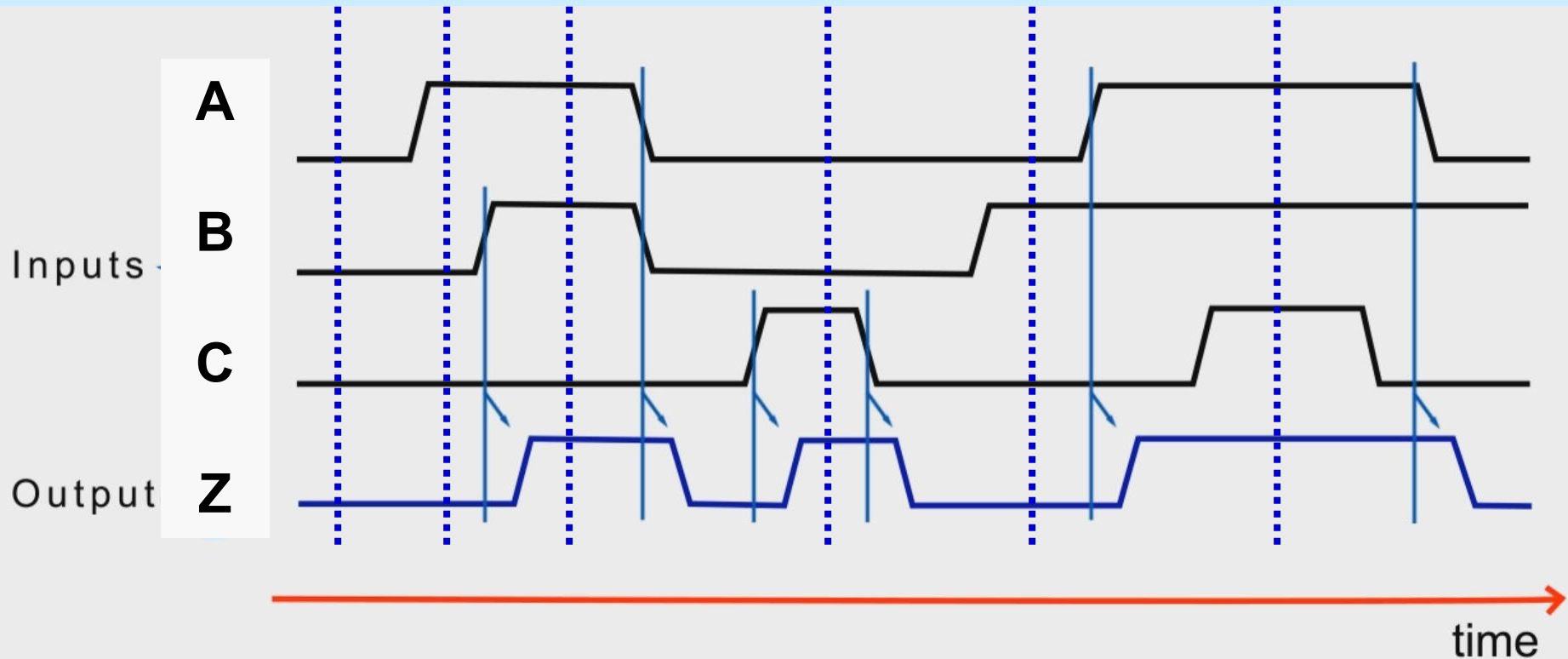


Fig. 3.17 (taken from Wakerly)

L2: 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

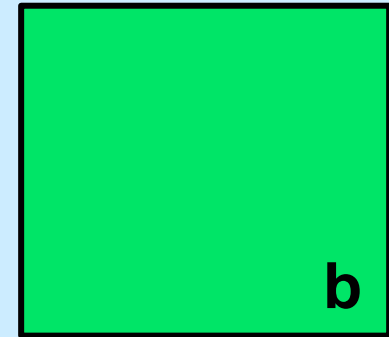
- **Essential to know and apply the theorems**
- **Not essential to name the theorems and prove them**
- **For those interested in proofs on Boolean theorems (optional):**
 - <http://www.electrical4u.com/boolean-algebra-theorems-and-laws-of-boolean-algebra/>
 - http://mines.humanoriented.com/410/books/boolean_algebra.pdf

Which concepts are not clear to you after viewing L2?

- A. Single variable theorems
- B. Absorption and consensus laws
- C. DeMorgan's theorems
- D. NOR, NAND
- E. None

Absorption laws

Absorption law (i):

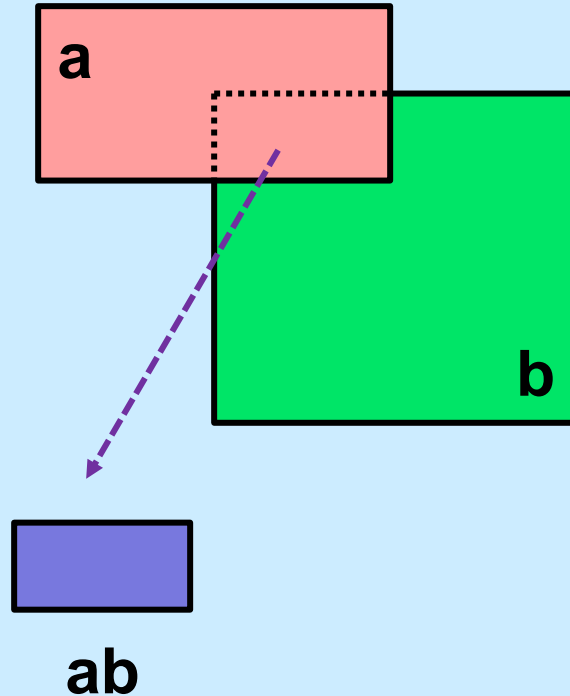


$$a + ab = a$$

Absorption laws (cont)

Absorption law (i):

$$a + ab = a$$



ab is absorbed in a

Simple illustration (i)

“A must come; B may come with A”

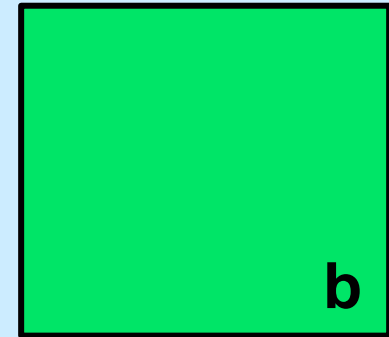
$$A + AB = A(1+B) = A(1) = A$$

Which is effectively the same as

“A must come”

Absorption laws (cont)

Absorption law (ii):

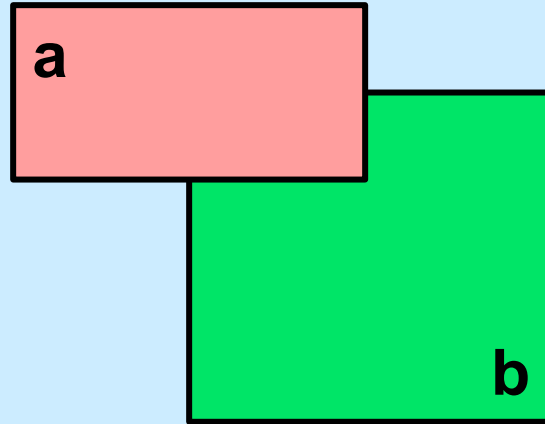


$$a + a'b = a + b$$

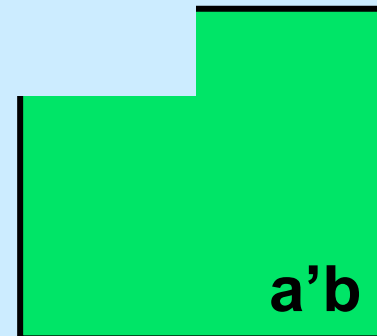
Absorption laws (cont)

Absorption law (ii):

$$a + a'b = a + b$$



a'b is absorbed in b



Simple illustration (ii)

“A must come; if A cannot come then B must come”

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

Which is same as saying

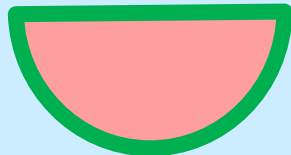
“A or B must come”

Consensus law

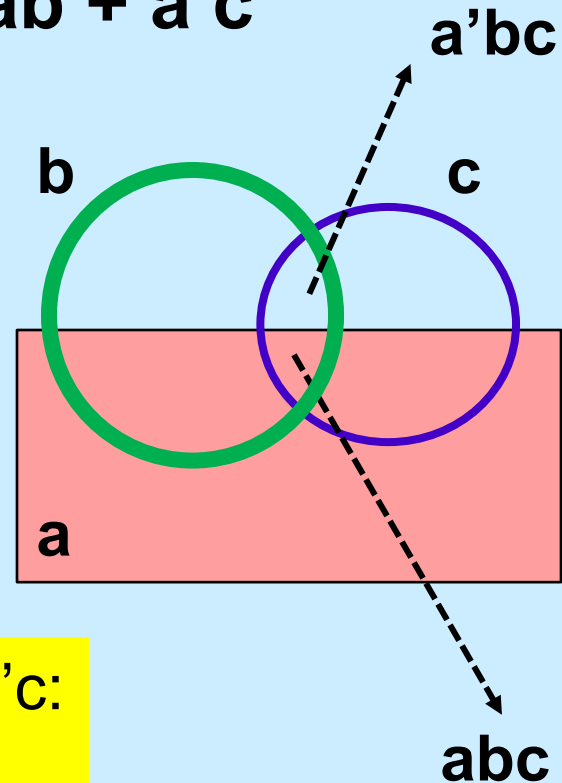
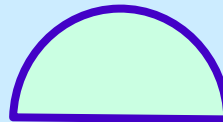
$$ab + a'c + bc = ab + a'c$$

$$\begin{aligned} bc &= (a + a') bc \\ &= abc + a'bc \end{aligned}$$













absorbed in ab :
 $ab + abc =$
 $ab(1+c) = ab$



absorbed in $a'c$:
 $a'c + a'bc =$
 $a'c(1+b) = a'c$



Simple illustration (consensus)

A	B	C	AB	A'C	BC
apple	banana	cherry	Like apple and banana	Dislike apple and like cherry	Like banana and cherry
dislike	dislike	dislike			
dislike	dislike			True	
dislike		dislike			
dislike				True	True
	dislike	dislike			
	dislike				
		dislike	True		
			True		True

Consensus (continue)

Those who like banana and cherry (BC) fall into two mutually exclusive groups:

- *Those who like apple and like banana (AB)*
- *Those who dislike apple and like cherry (A'C)*

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

Consensus (another example)

A=1 (male); A=0 (female)

B=1 (classmate); B=0 (not classmate)

C=1 (friend); C=0 (not friend)

Those who are both your classmate and friend (BC) will fall into one of these two groups:

- *your male classmates (AB)*
- *your female friends (A'C)*

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

DeMorgans theorem (i)

“Have lunch if A has money or B has money”

Have_lunch = A_has\$ + B_has\$

$$\begin{aligned}(\text{Have_lunch})' &= (\text{A_has\$} + \text{B_has\$})' \\ &= (\text{A_has\$})' \bullet (\text{B_has\$})'\end{aligned}$$

Which is same as saying

“No lunch if A has no money and B has no money”

No_lunch = (A_no\$) AND (B_no\$)

DeMorgans theorem (ii)

“Marriage if A agrees and B agrees”

$$\text{marriage} = (A_agrees) \bullet (B_agrees)$$

$$\begin{aligned} (\text{marriage})' &= [(A_agrees) \bullet (B_agrees)]' \\ &= (A_agrees)' + (B_agrees)' \end{aligned}$$

Which is same as saying

“No marriage if A disagrees or B disagrees”

$$\text{No_marriage} = (A_disagrees) \text{ OR } (B_disagrees)$$

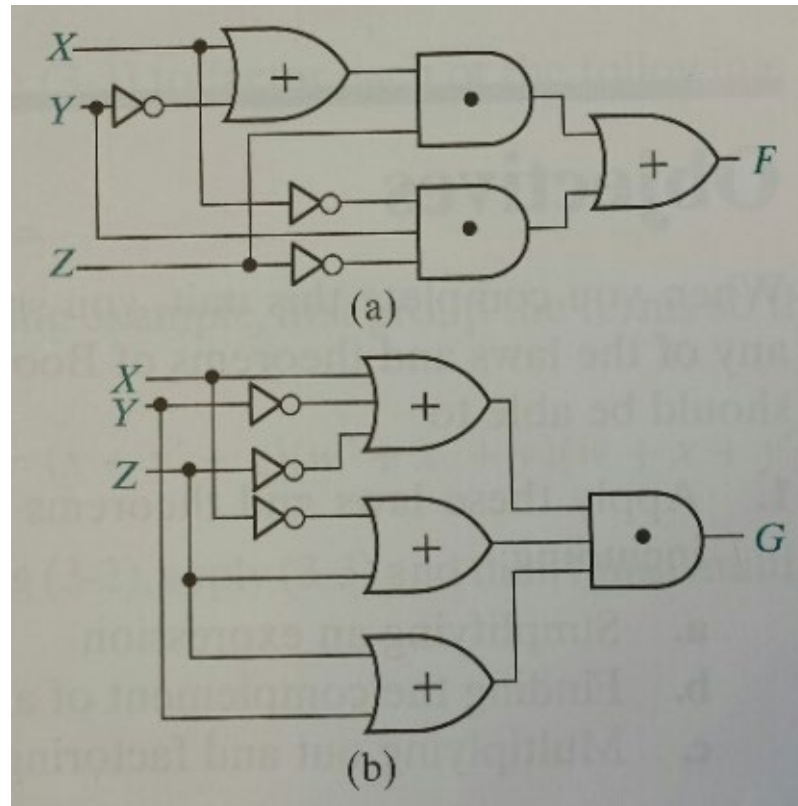
$$\mathbf{A(B+C)' = AB' + AC'}$$

True or false?

A. True

B. False

Are these two circuits algebraically equivalent?



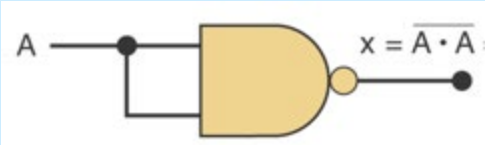
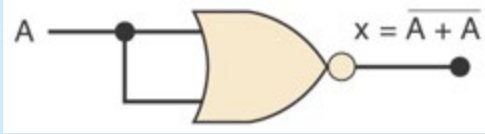
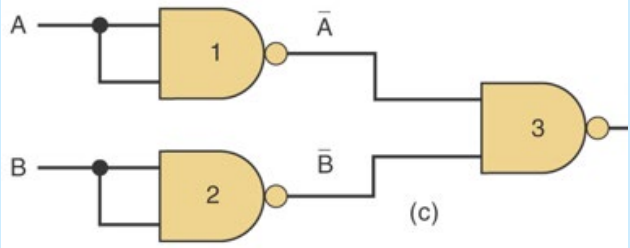
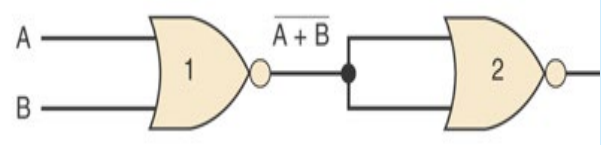
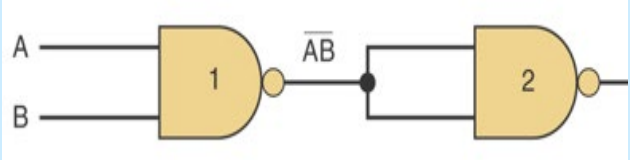
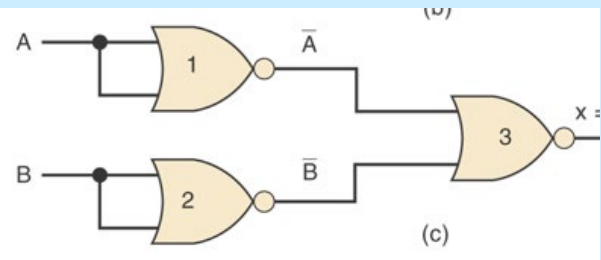
A. Yes

B. No

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		

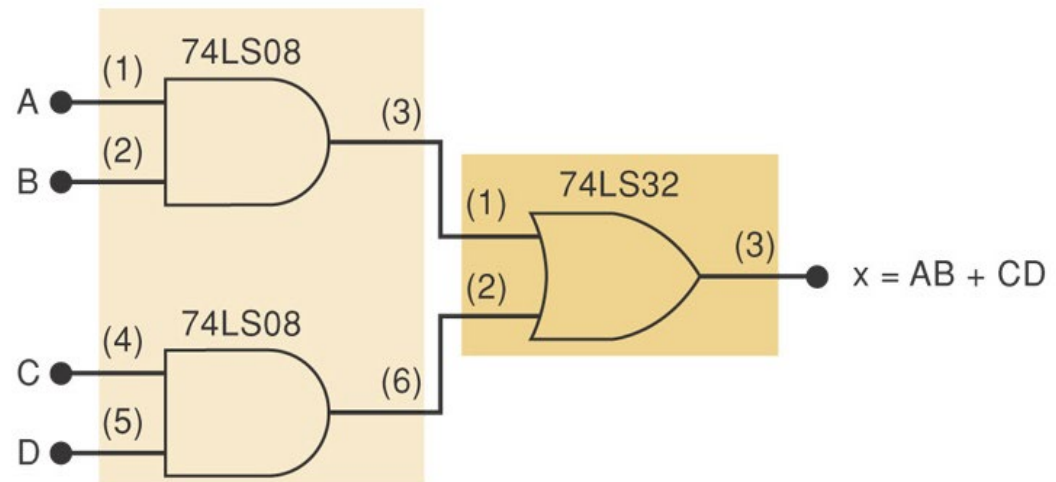
How many NAND gates are needed in total to replace all the gates?

A. 3

B. 4

C. 5

D. 6



**Universal gates always reduce
the number of gates used.
True or false?**

A. True

B. False

End of L1, L2 summary