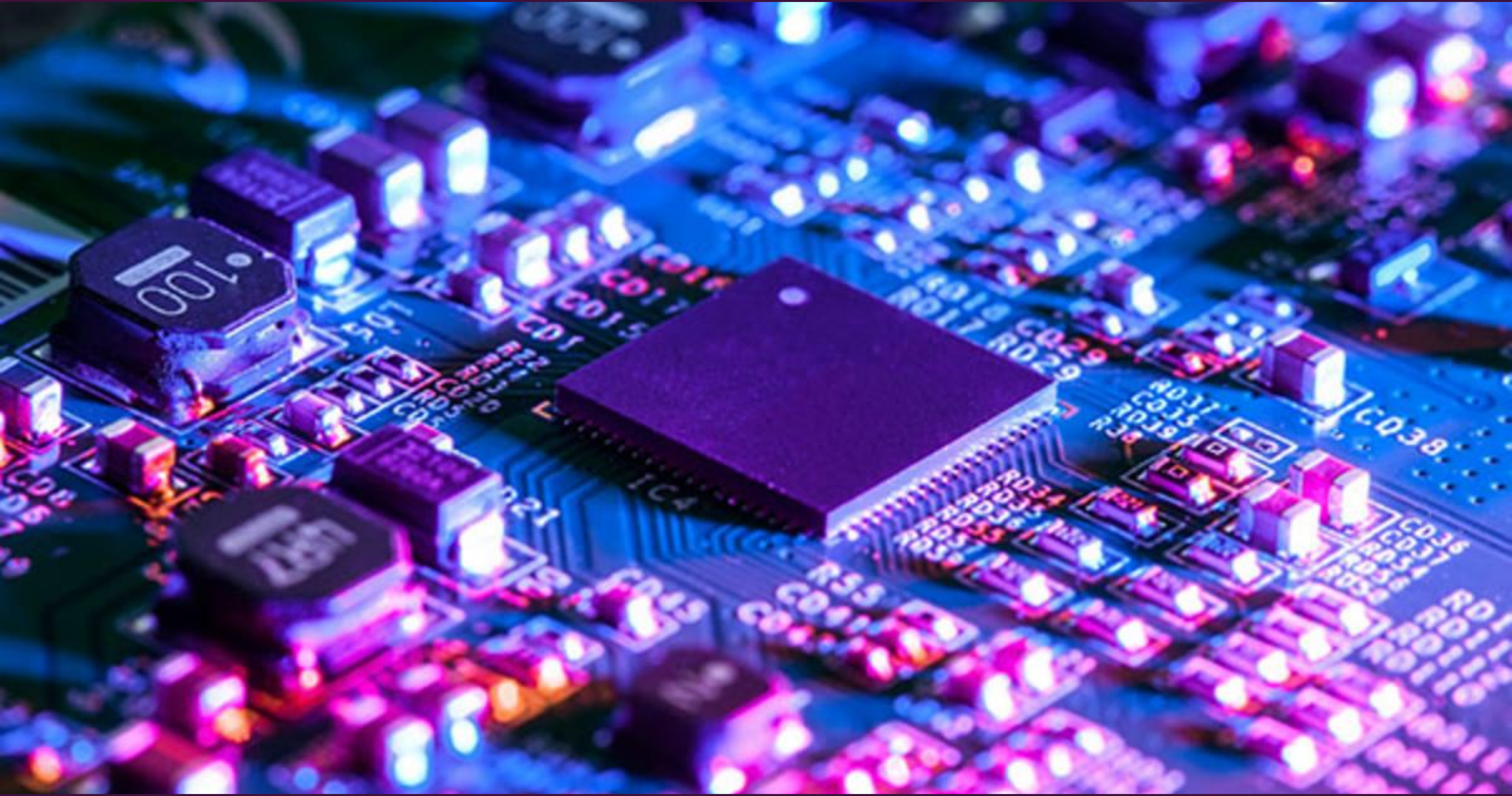
Electronics II – Chapter 4

Boolean Algebra and Logic Simplification



Electronics Research Laboratory

FIGURE 4-3 Application of commutative law of addition.

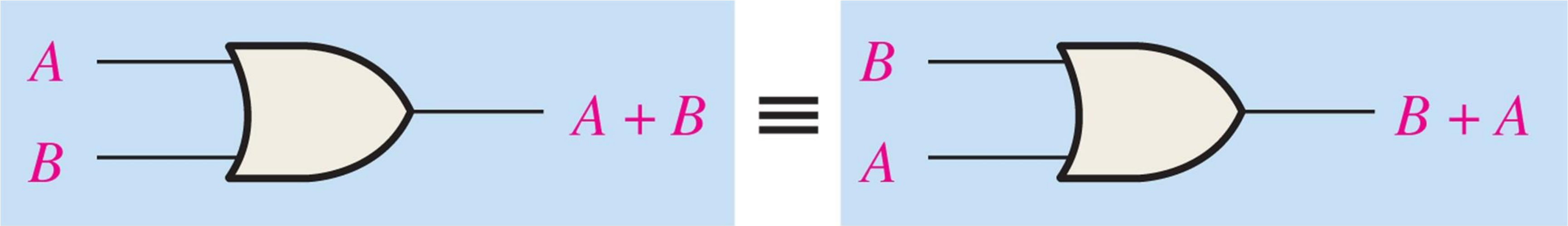


FIGURE 4-4 Application of commutative law of multiplication.

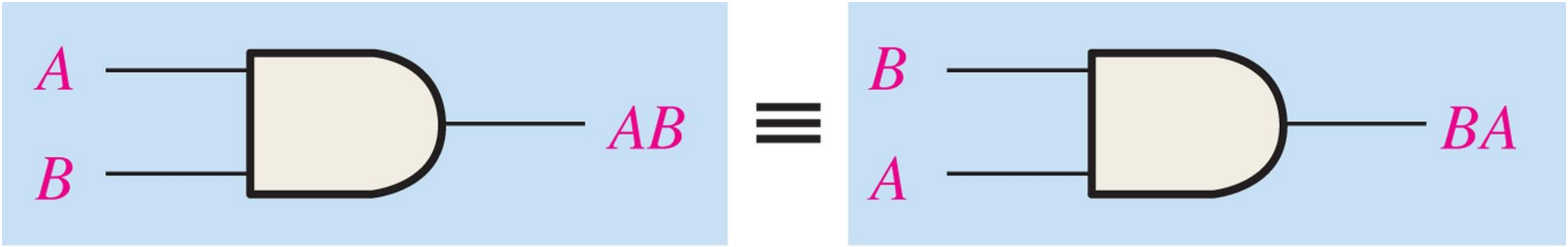


FIGURE 4-5 Application of associative law of addition.

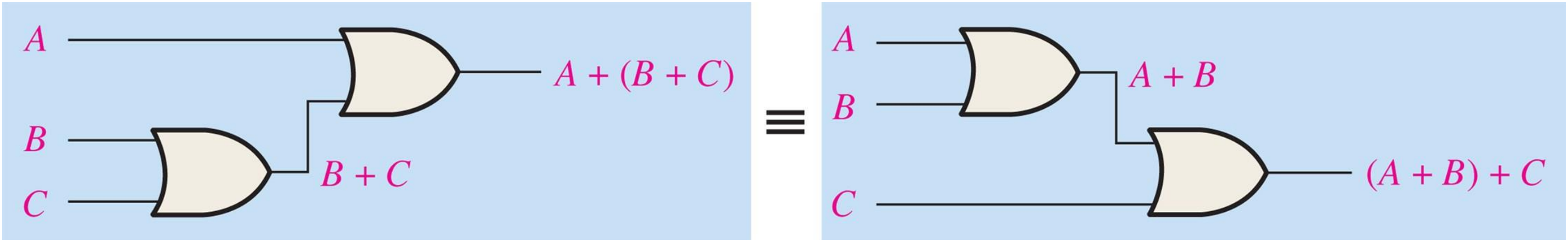


FIGURE 4-6 Application of associative law of multiplication.

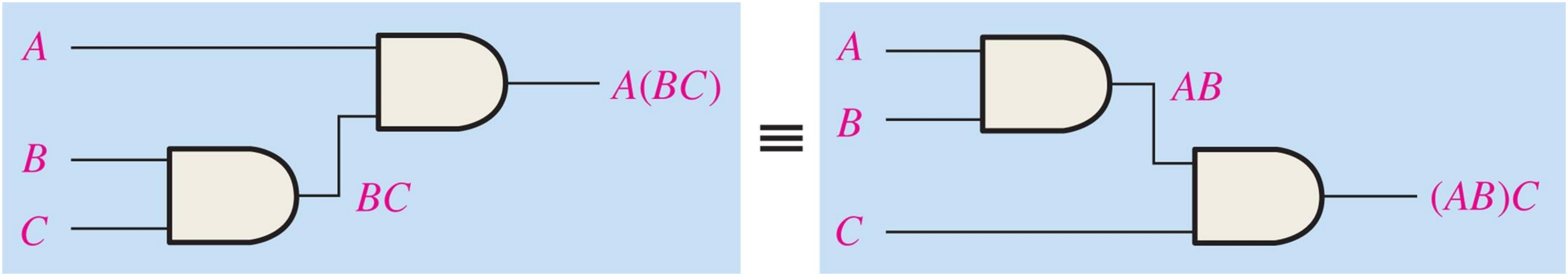
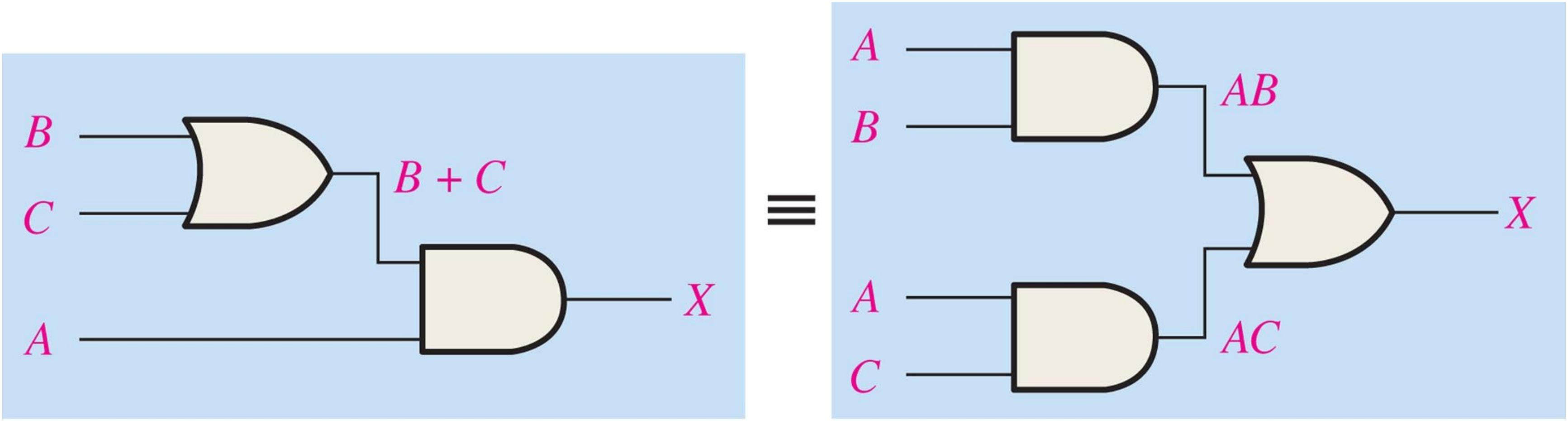


FIGURE 4-7 Application of distributive law.



$X = A(B + C)$

$X = AB + AC$

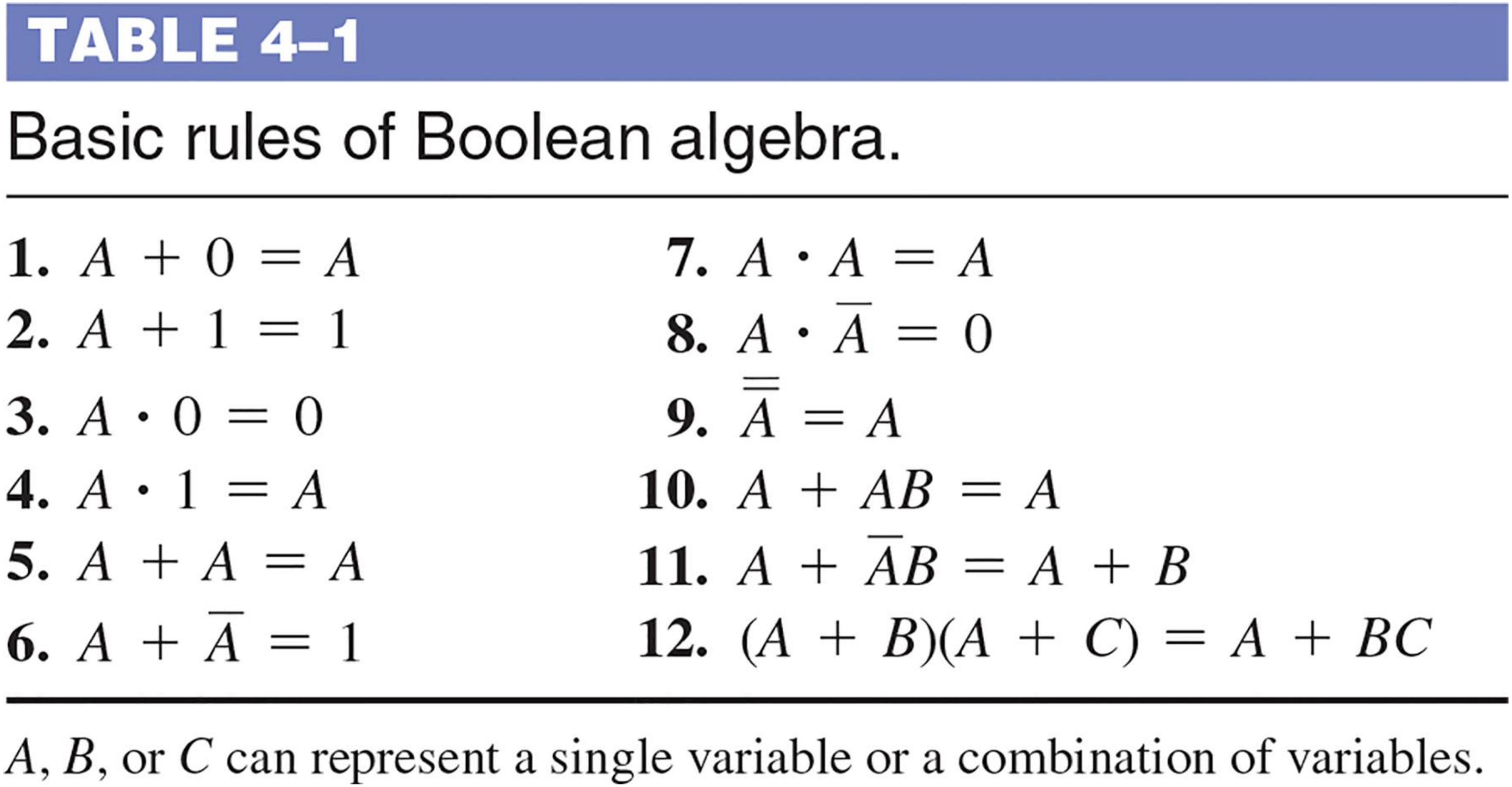


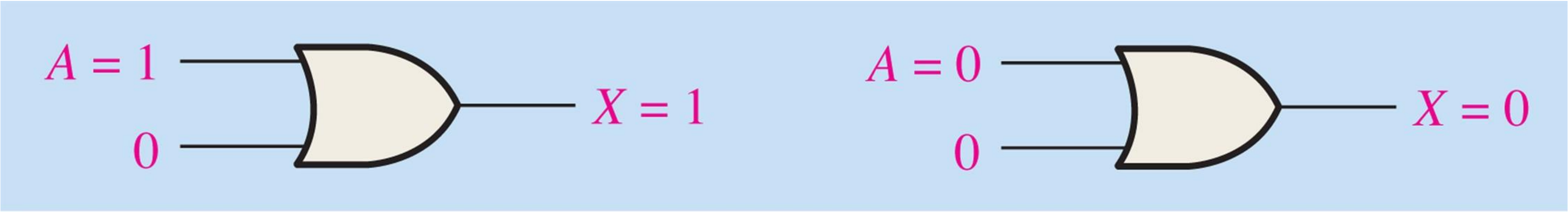
TABLE 4–1

Basic rules of Boolean algebra.

1. $A + 0 = A$
2. $A + 1 = 1$
3. $A \cdot 0 = 0$
4. $A \cdot 1 = A$
5. $A + A = A$
6. $A + \overline{A} = 1$
7. $A \cdot A = A$
8. $A \cdot \overline{A} = 0$
9. $\overline{\overline{A}} = A$
10. $A + AB = A$
11. $A + \overline{A}B = A + B$
12. $(A + B)(A + C) = A + BC$

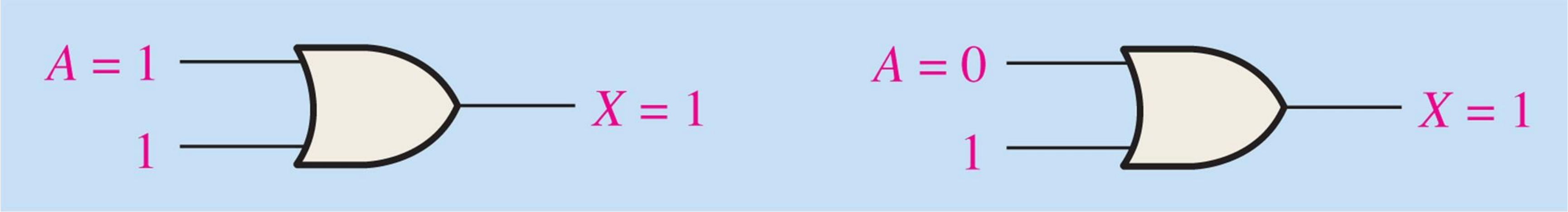
A, B, or C can represent a single variable or a combination of variables.

FIGURE 4-8



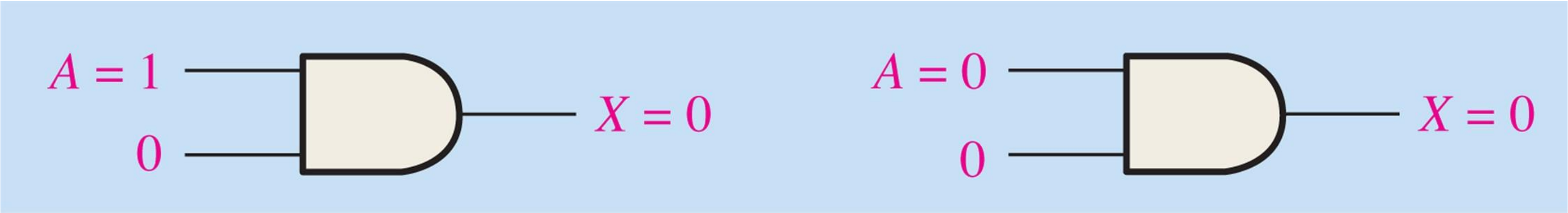
$$X = A + 0 = A$$

FIGURE 4-9



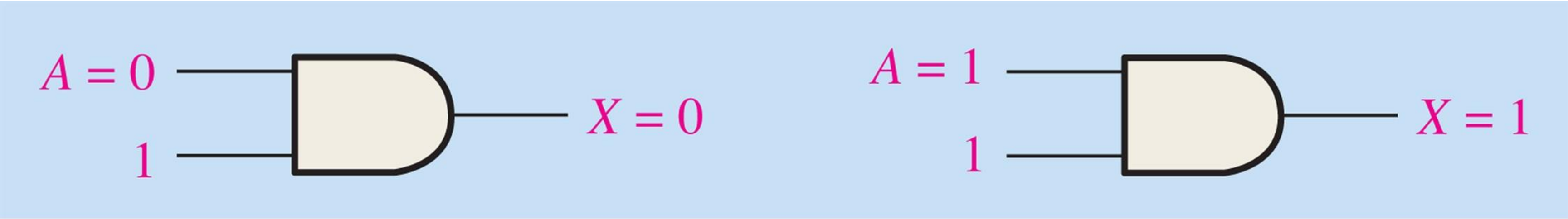
$X = A + 1 = 1$

FIGURE 4-10



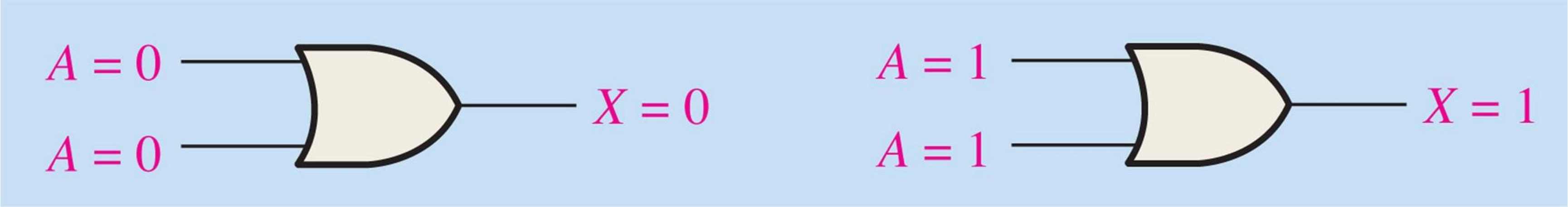
$$X = A \bullet 0 = 0$$

FIGURE 4-11



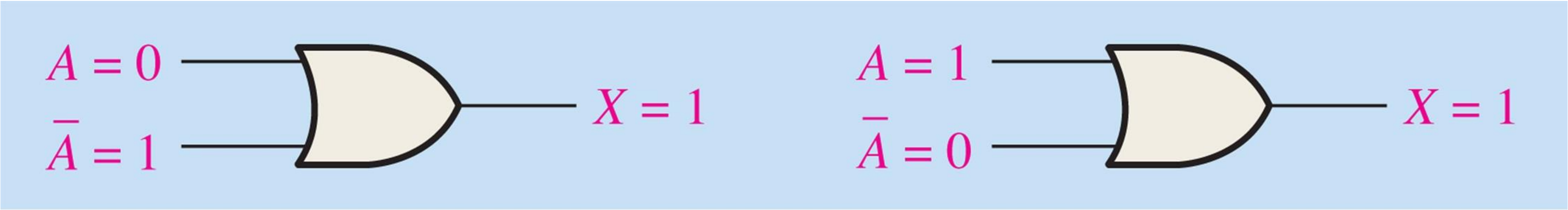
$$X = A \cdot 1 = A$$

FIGURE 4-12



$$X = A + A = A$$

FIGURE 4-13



$$X = A + \overline{A} = 1$$

FIGURE 4-14

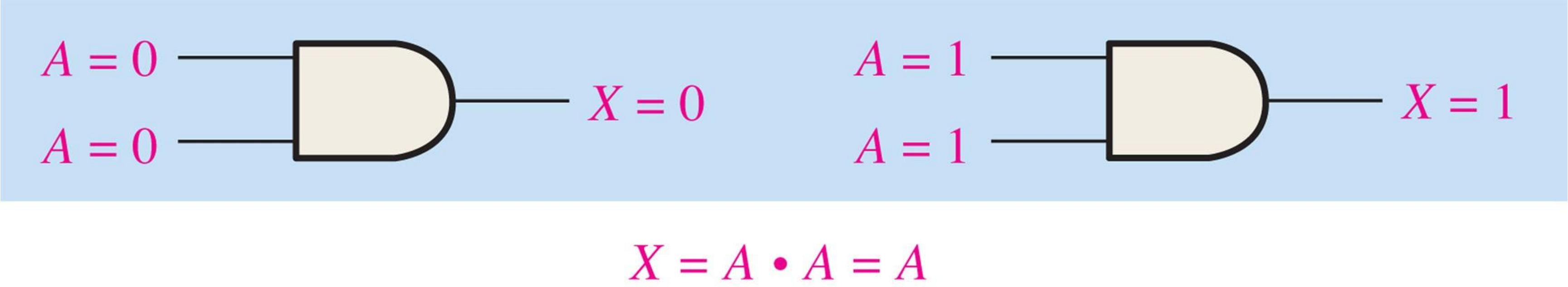
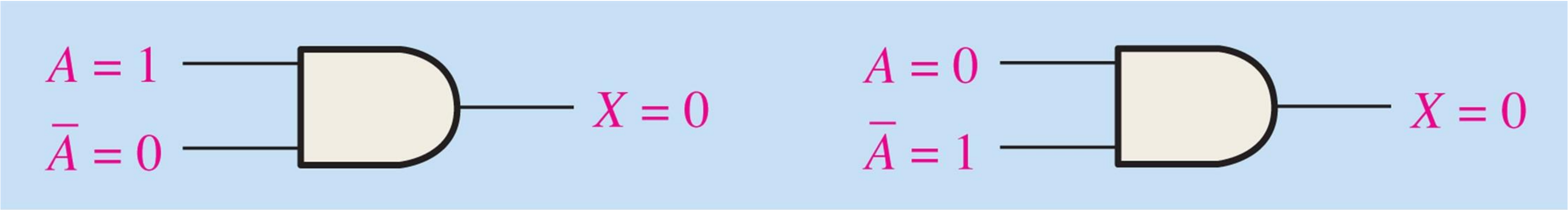
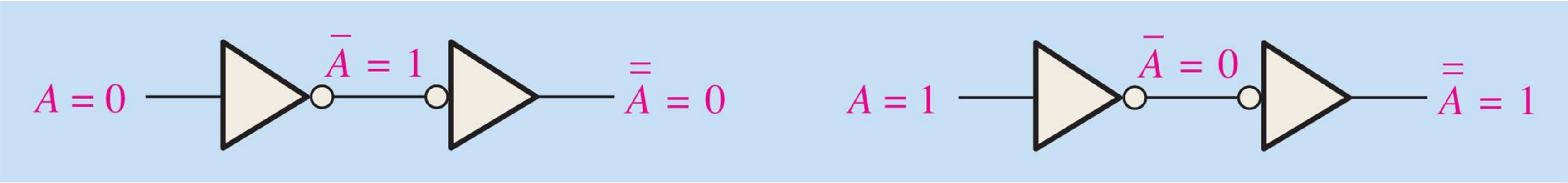


FIGURE 4-15



$$X = A \cdot \overline{A} = 0$$

FIGURE 4-16



$$\overline{\overline{A}} = A$$

TABLE 4–2

Rule 10: $A + AB = A$. Open file T04-02 to verify.

A	B	AB	$A + AB$
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

equal

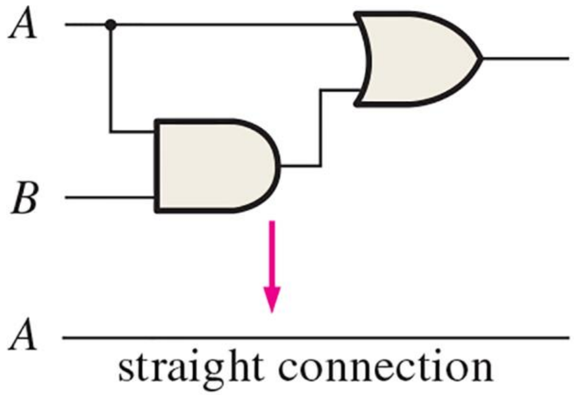


TABLE 4–3

Rule 11: $A + \overline{A}B = A + B$. Open file T04-03 to verify.

A	B	$\overline{A}B$	$A + \overline{A}B$	$A + B$
0	0	0	0	0
0	1	1	1	1
1	0	0	1	1
1	1	0	1	1

$A + \overline{A}B$ and $A + B$: equal

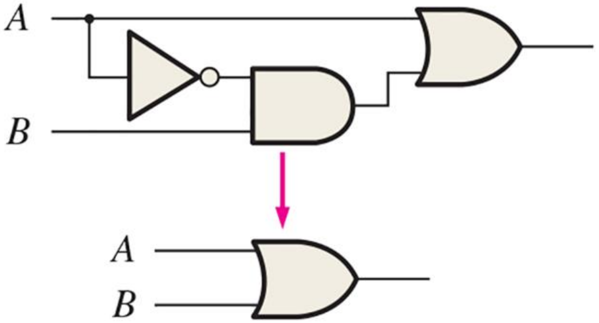


TABLE 4–4

Rule 12: $(A + B)(A + C) = A + BC$. Open file T04-04 to verify.

A	B	C	$A + B$	$A + C$	$(A + B)(A + C)$	BC	$A + BC$
0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	1	1	1	1
1	0	0	1	1	1	0	1
1	0	1	1	1	1	0	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

equal

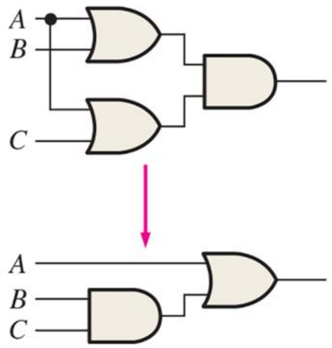
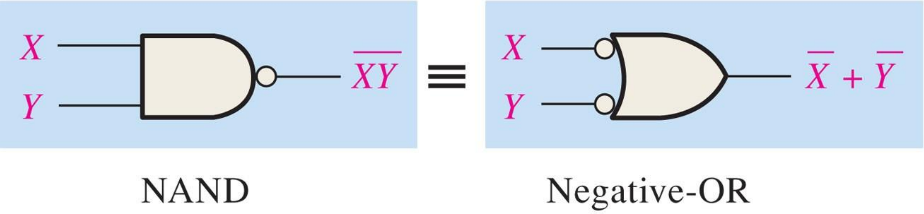
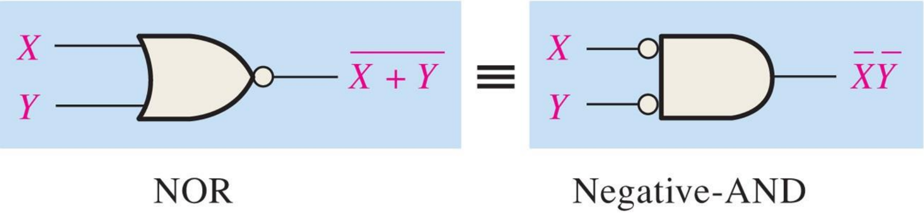


FIGURE 4-17 Gate equivalencies and the corresponding truth tables that illustrate DeMorgan's theorems. Notice the equality of the two output columns in each table. This shows that the equivalent gates perform the same logic function.

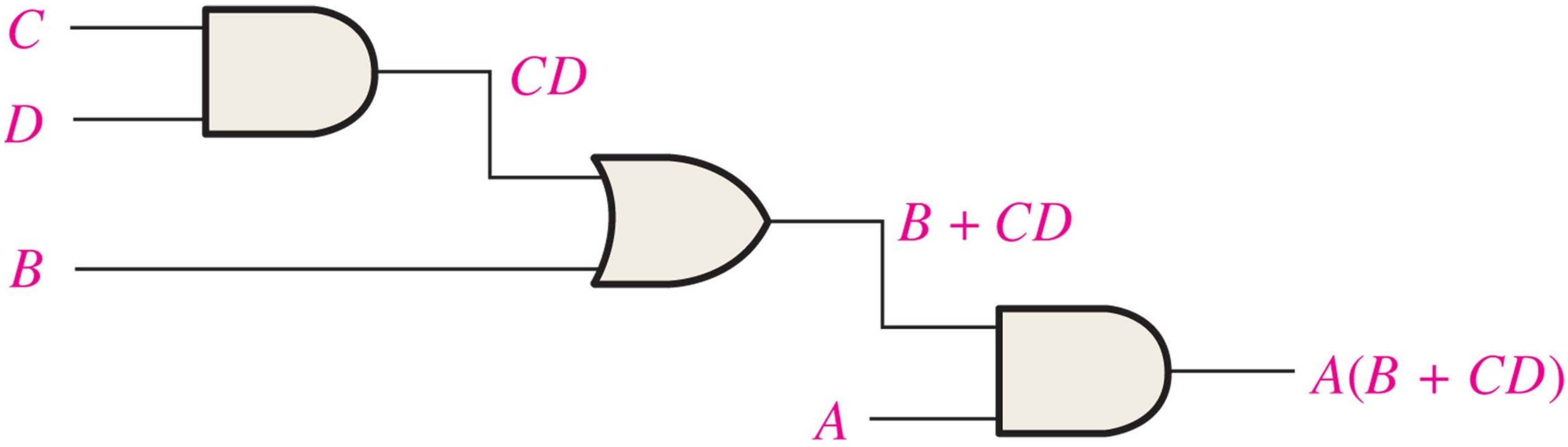


Inputs		Output	
X	Y	$\overline{XY}$	$\overline{X} + \overline{Y}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0



Inputs		Output	
X	Y	$\overline{X + Y}$	$\overline{X}\,\overline{Y}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

FIGURE 4-18 A combinational logic circuit showing the development of the Boolean expression for the output.



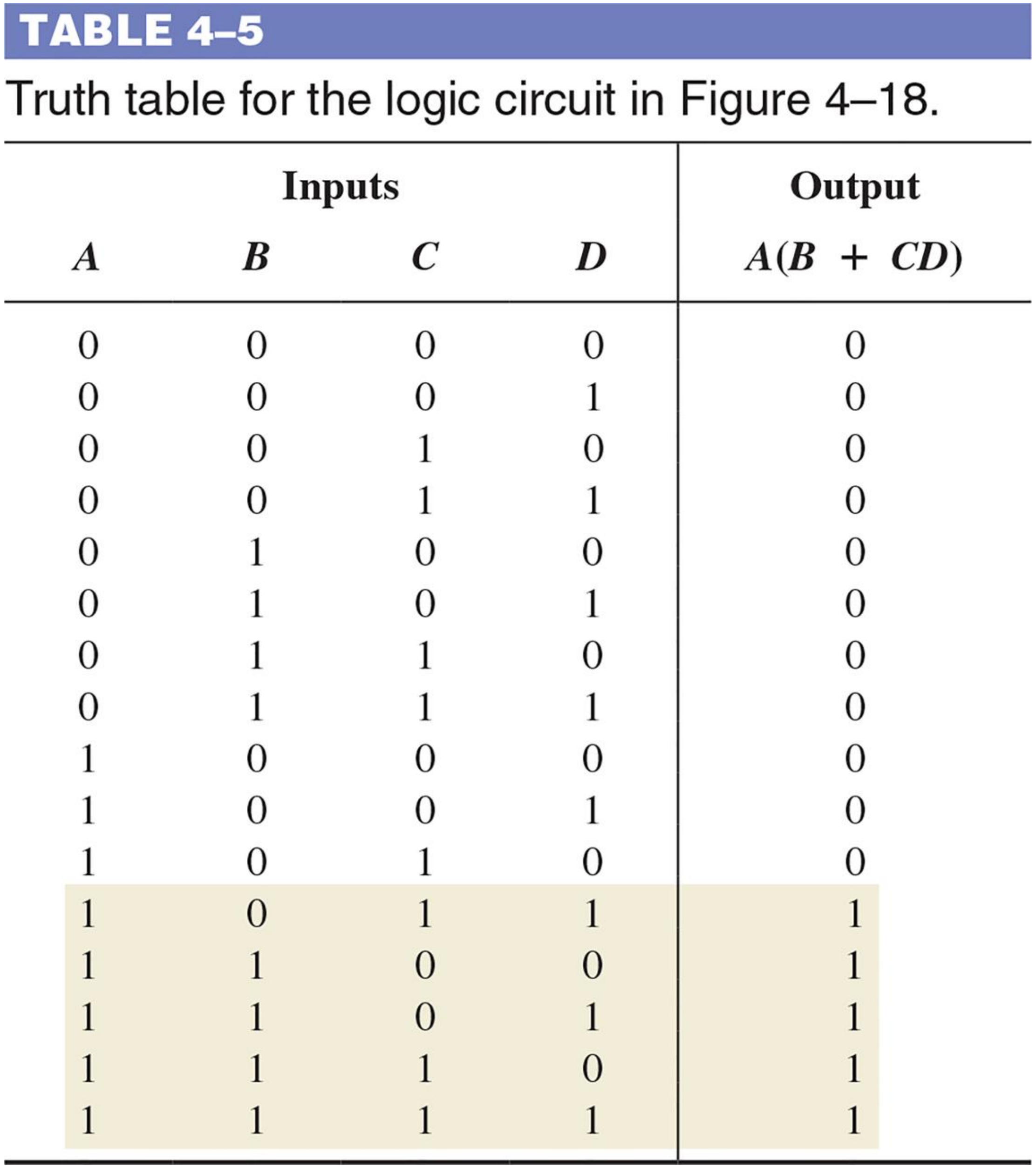


TABLE 4–5

Truth table for the logic circuit in Figure 4–18.

Inputs				Output
A	*B*	*C*	*D*	*A*(*B* + *CD*)
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

FIGURE 4-20 Gate circuits for Example 4–9.

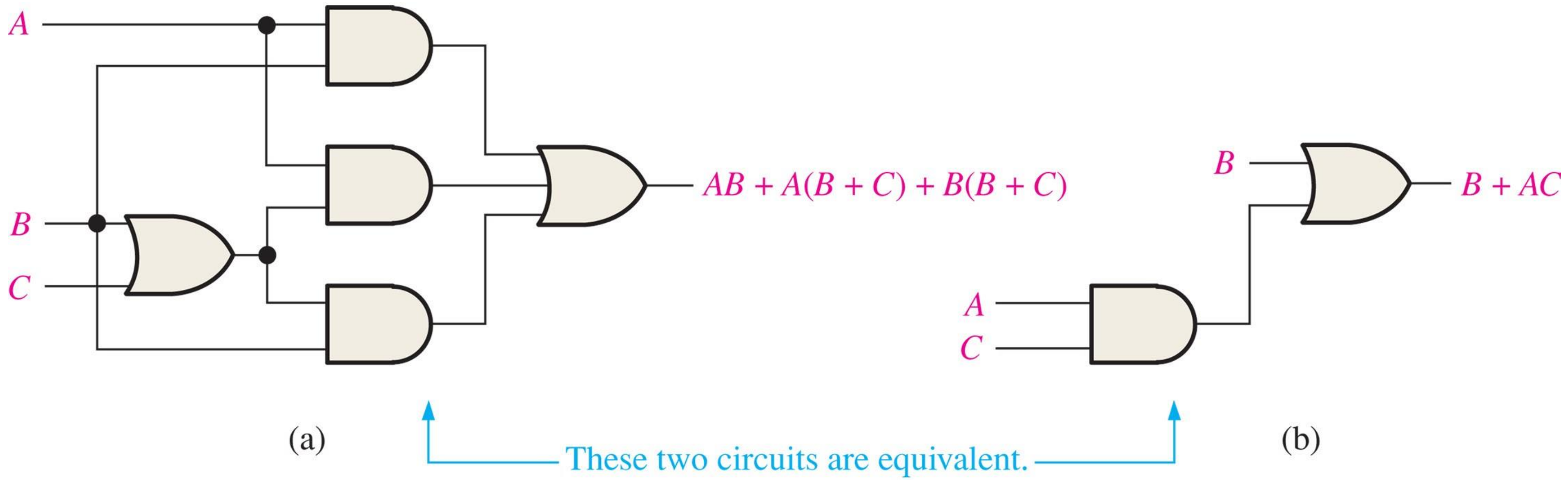


FIGURE 4-22 Implementation of the SOP (Sum-Of-Products) expression *AB* + *BCD* + *AC.*

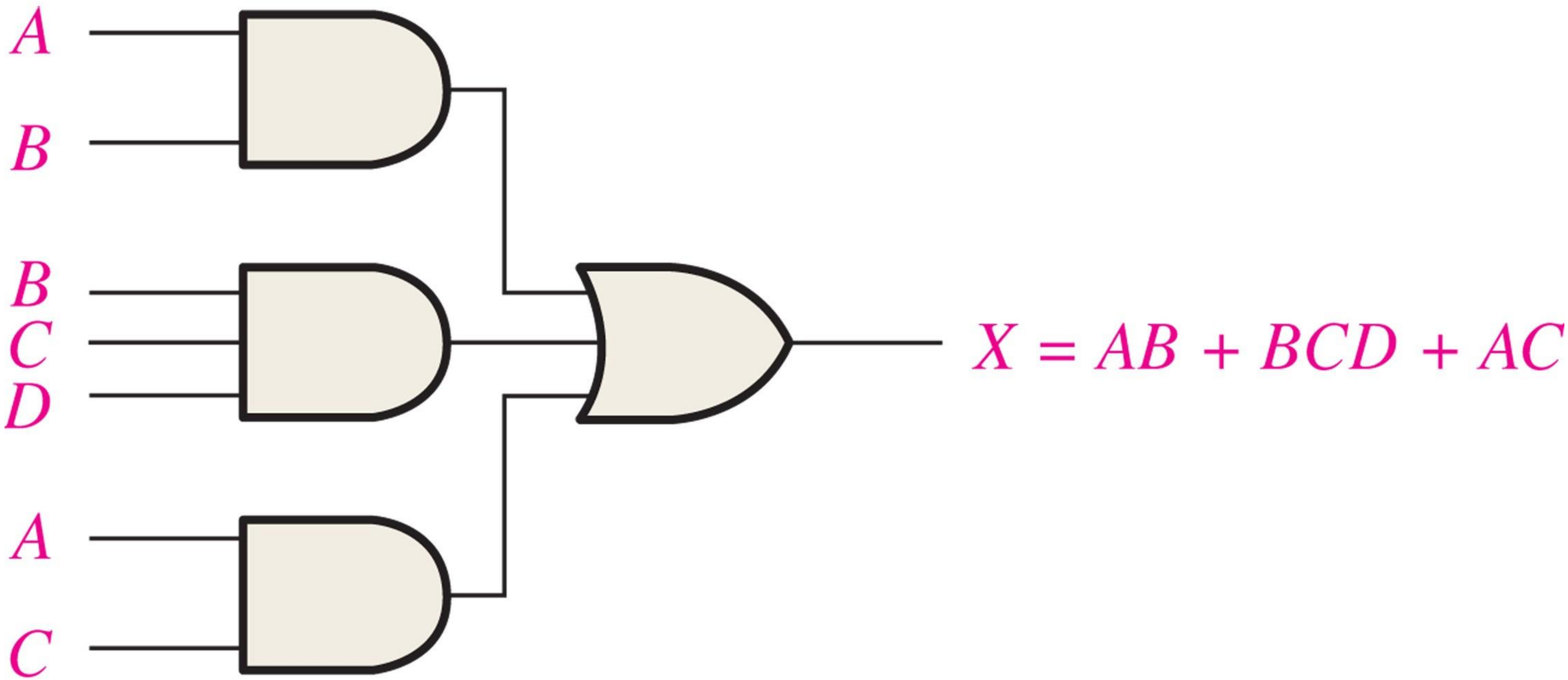
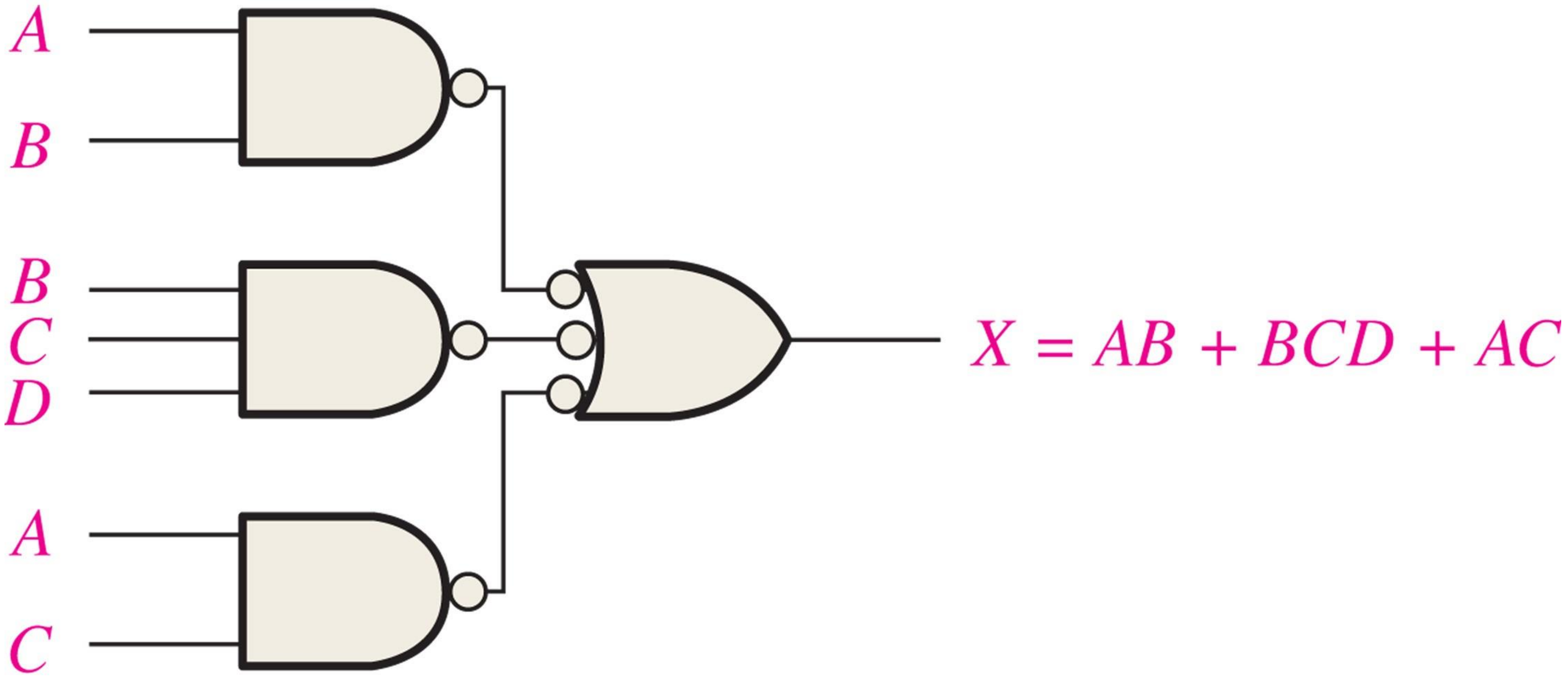


FIGURE 4-23 This NAND/NAND implementation is equivalent to the AND/OR in Figure 4–22.



$X = AB + BCD + AC$

FIGURE 4-24 Implementation of the POS (Sum-Of-Products) expression $(A + B)(B + C + D)(A + C)$.

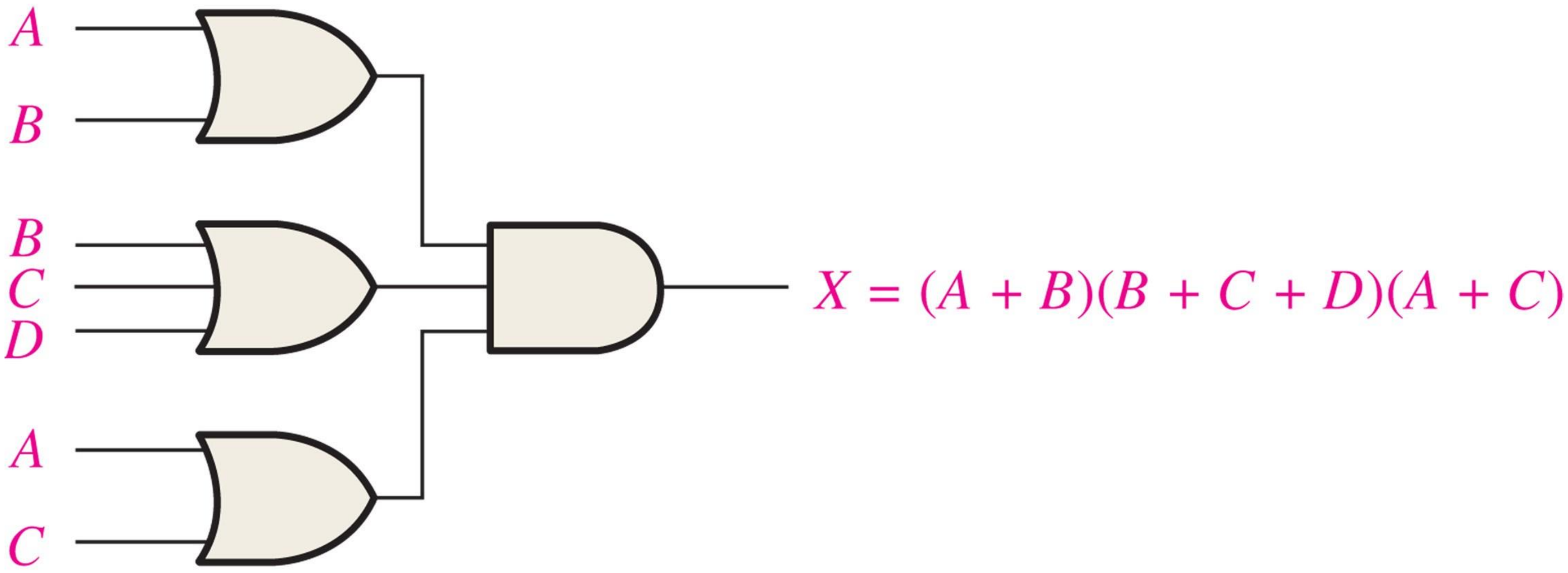
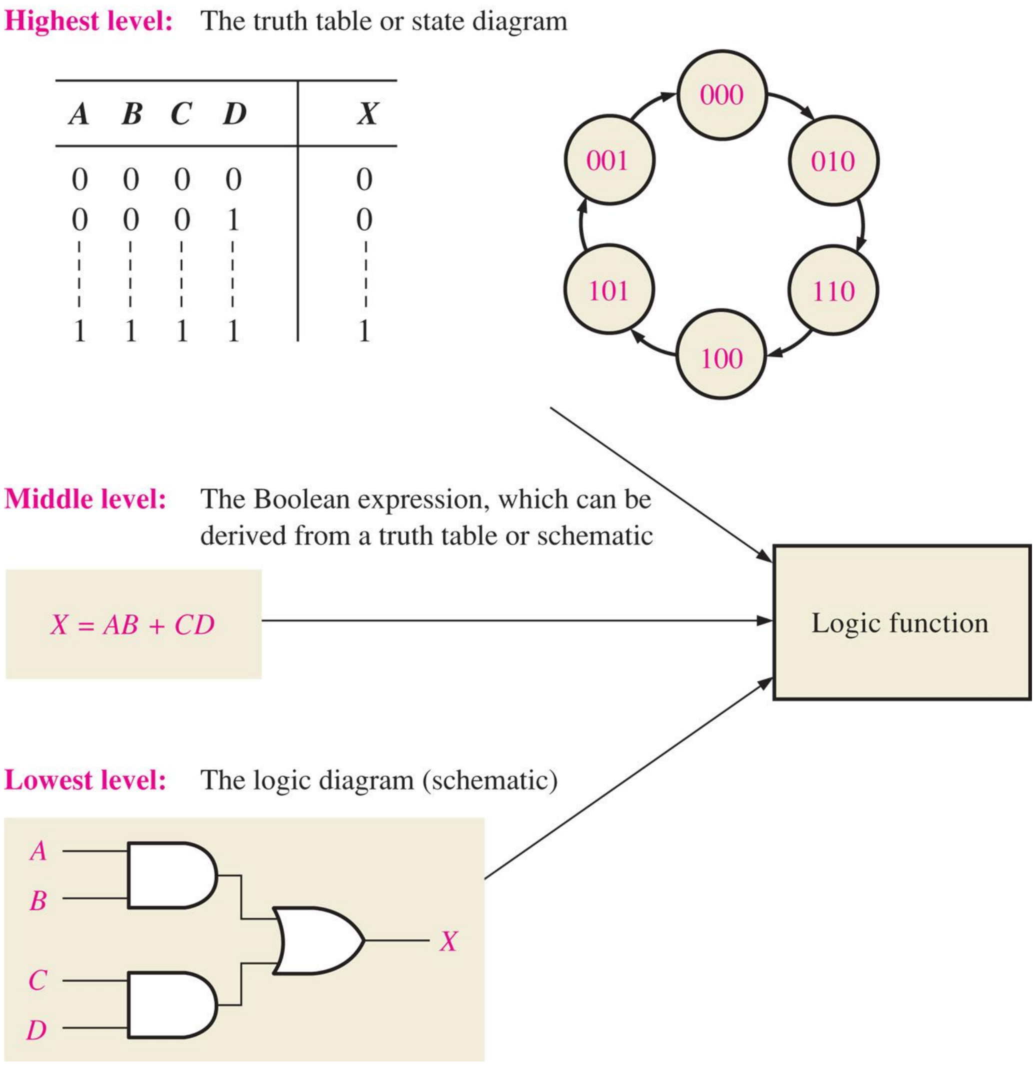


FIGURE 4-49 Illustration of the three levels of abstraction for describing a logic function.



Highest level: The truth table or state diagram

A	*B*	*C*	*D*	*X*
0	0	0	0	0
0	0	0	1	0
⋮	⋮	⋮	⋮	⋮
1	1	1	1	1

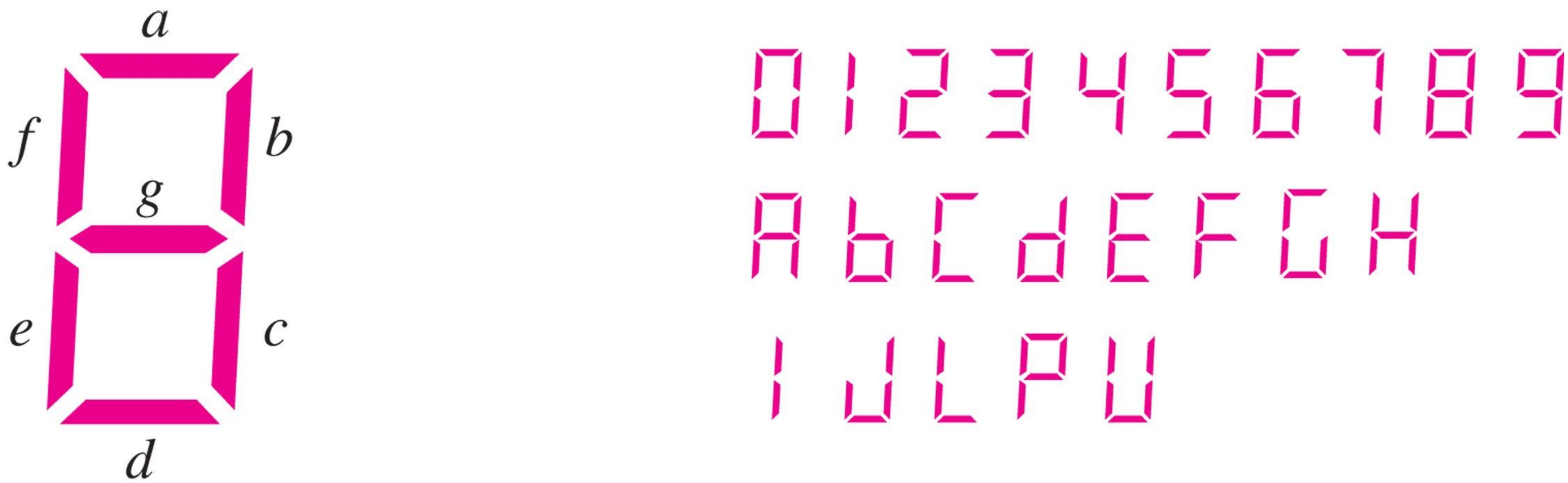
Middle level: The Boolean expression, which can be derived from a truth table or schematic

$X = AB + CD$

Logic function

Lowest level: The logic diagram (schematic)

FIGURE 4-50 Seven-segment display.



(a) Segment arrangement

(b) Formation of the ten digits and certain letters

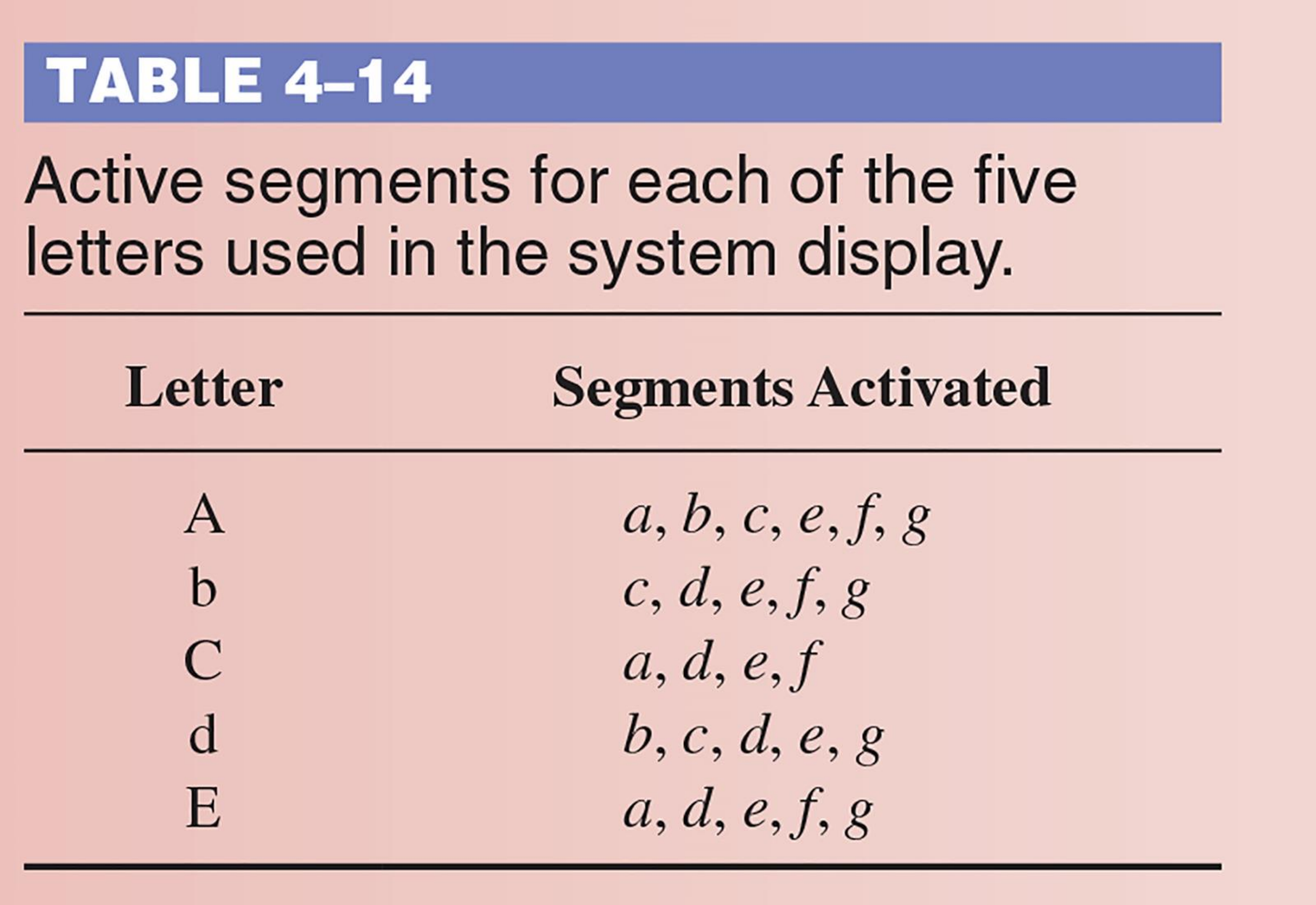
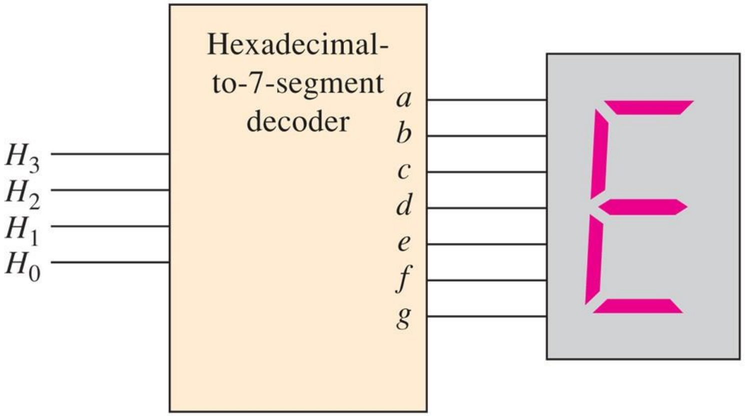


TABLE 4–14

Active segments for each of the five letters used in the system display.

Letter	Segments Activated
A	*a, b, c, e, f, g*
b	*c, d, e, f, g*
C	*a, d, e, f*
d	*b, c, d, e, g*
E	*a, d, e, f, g*

FIGURE 4-51 Hexadecimal-to-7-segment decoder for letters *A* through *E*, used in the system.

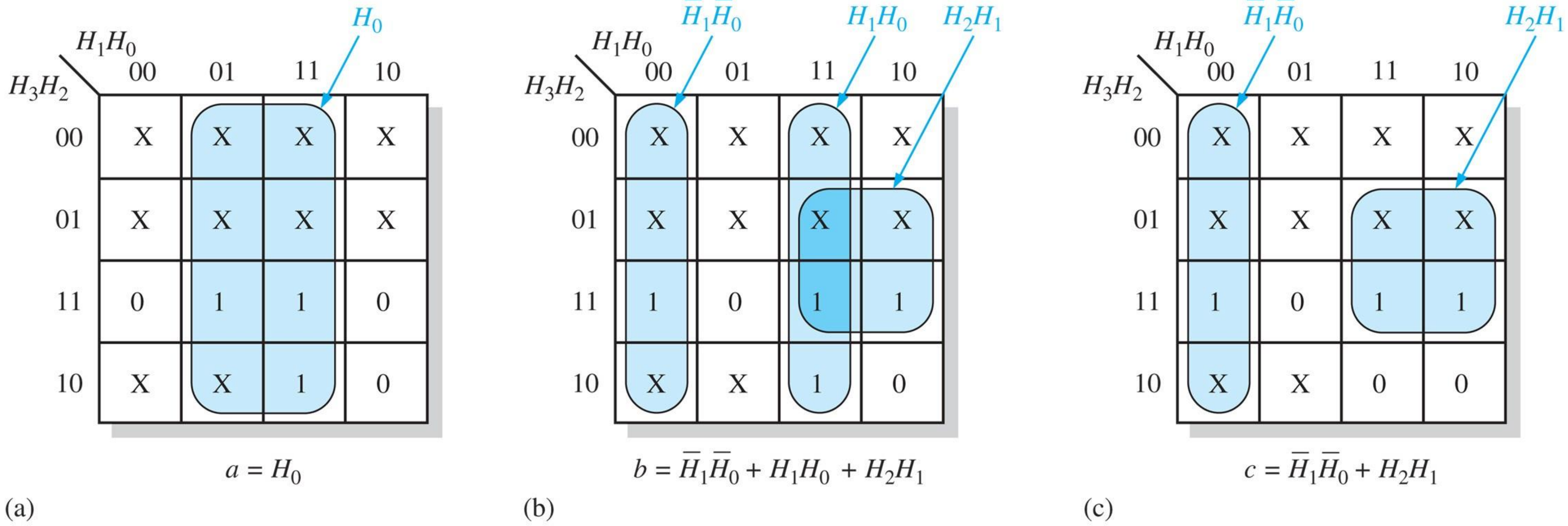


(a)

Letter	Hexadecimal Inputs				Segment Ouputs						
	H_3	H_2	H_1	H_0	*a*	*b*	*c*	*d*	*e*	*f*	*g*
A	1	0	1	0	0	0	0	1	0	0	0
b	1	0	1	1	1	1	0	0	0	0	0
C	1	1	0	0	0	1	1	0	0	0	1
d	1	1	0	1	1	0	0	0	0	1	0
E	1	1	1	0	0	1	1	0	0	0	0
F	1	1	1	1	0	1	1	1	0	0	0

(b)

FIGURE 4-52 Minimization of the expressions for segments *a, b,* and *c.*



H_0
$\overline{H}_1\overline{H}_0$
H_1H_0
H_2H_1
$\overline{H}_1\overline{H}_0$
H_2H_1

H_3H_2 \ H_1H_0	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	0	1	1	0
10	X	X	1	0

$a = H_0$

(a)

H_3H_2 \ H_1H_0	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	1	0	1	1
10	X	X	1	0

$b = \overline{H}_1\overline{H}_0 + H_1H_0 + H_2H_1$

(b)

H_3H_2 \ H_1H_0	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	1	0	1	1
10	X	X	0	0

$c = \overline{H}_1\overline{H}_0 + H_2H_1$

(c)

FIGURE 4-53 Segment-*b* and segment-*c* logic circuits.

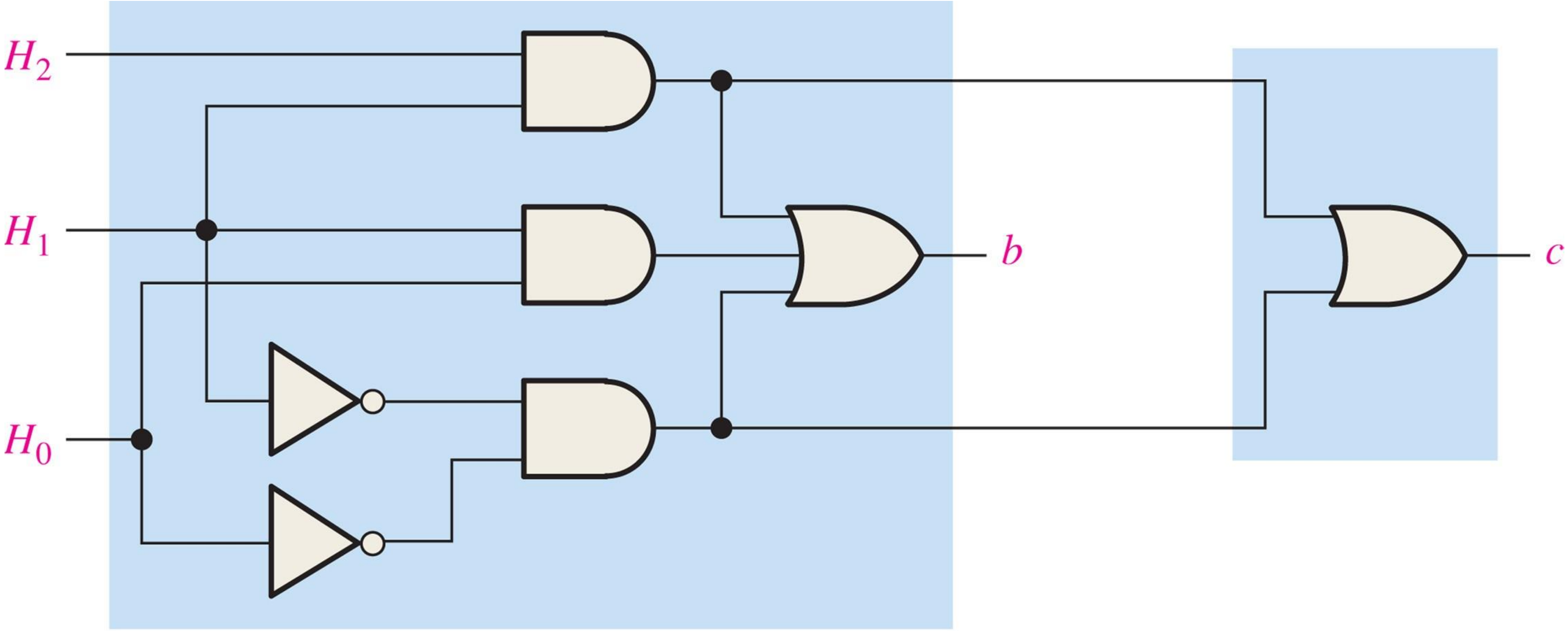
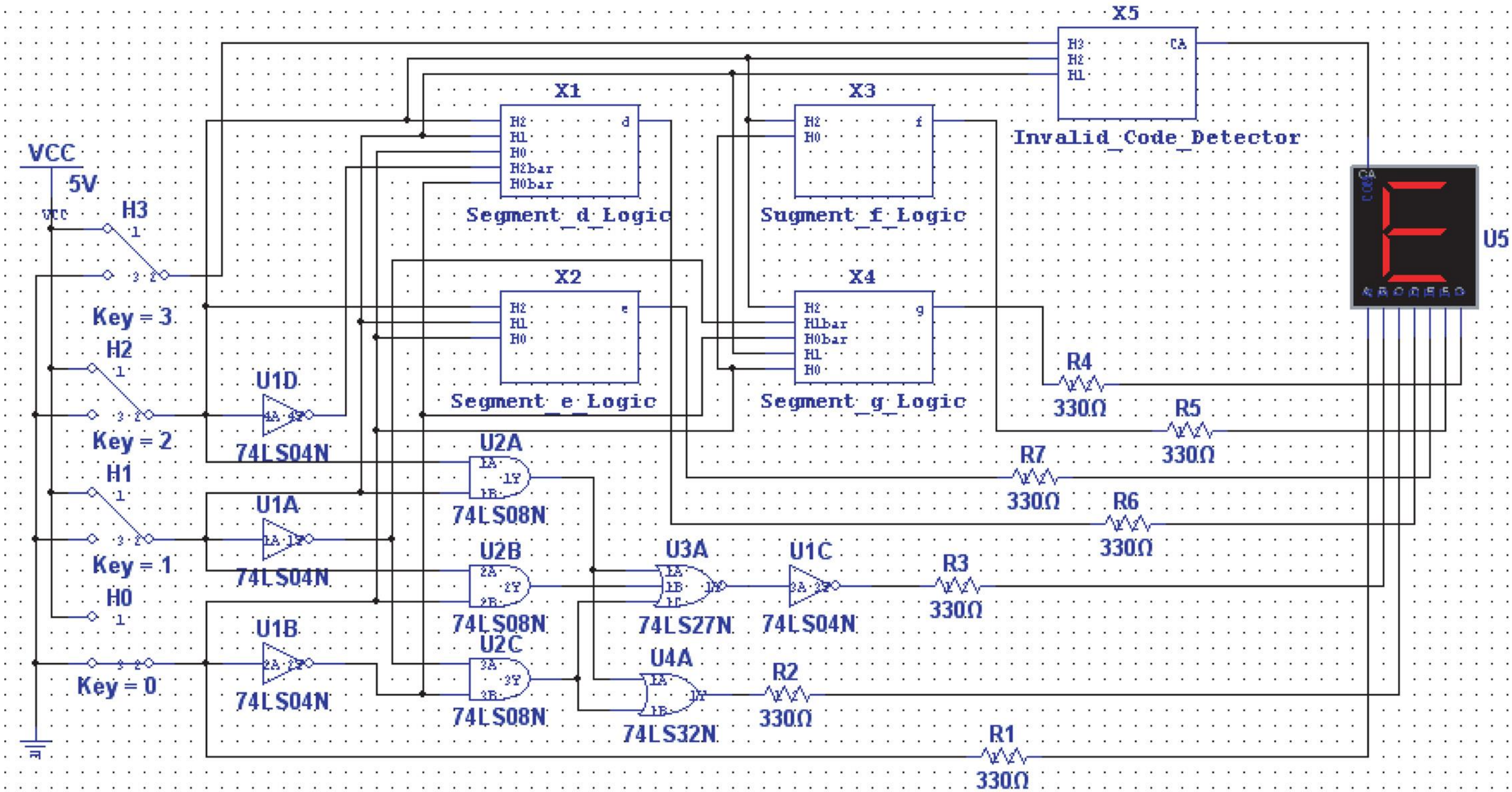


FIGURE 4-54 Multisim circuit screen for decoder and display.


X5
H3
H2
H1
CA
Invalid_Code_Detector
X1
H2
H1
H0
H2bar
H0bar
d
Segment_d_Logic
X3
H2
H0
f
Sugment_f_Logic
X2
H2
H1
H0
e
Segment_e_Logic
X4
H2
H1bar
H0bar
H1
H0
g
Segment_g_Logic
VCC
5V
H3
Key = 3
H2
Key = 2
H1
Key = 1
H0
Key = 0
U1D
74LS04N
U1A
74LS04N
U1B
74LS04N
U2A
74LS08N
U2B
74LS08N
U2C
74LS08N
U3A
74LS27N
U1C
74LS04N
U4A
74LS32N
R4
330Ω
R5
330Ω
R7
330Ω
R6
330Ω
R3
330Ω
R2
330Ω
R1
330Ω
U5