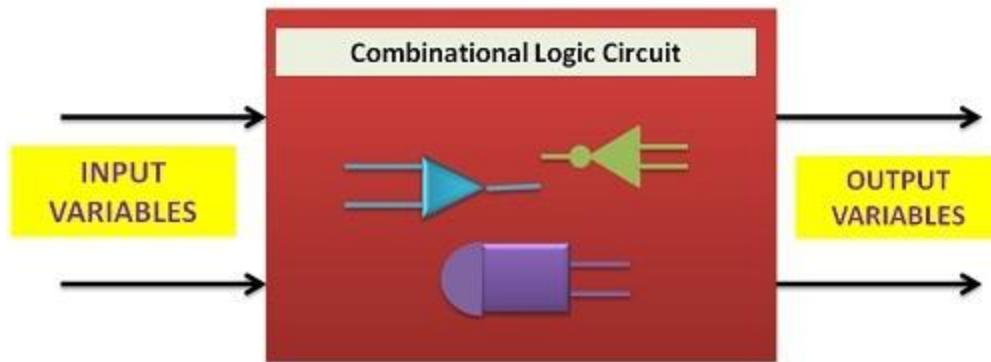


# مدارهای منطقی

## فصل چهارم

### مدارهای ترکیبی



# سرفصل مطالب

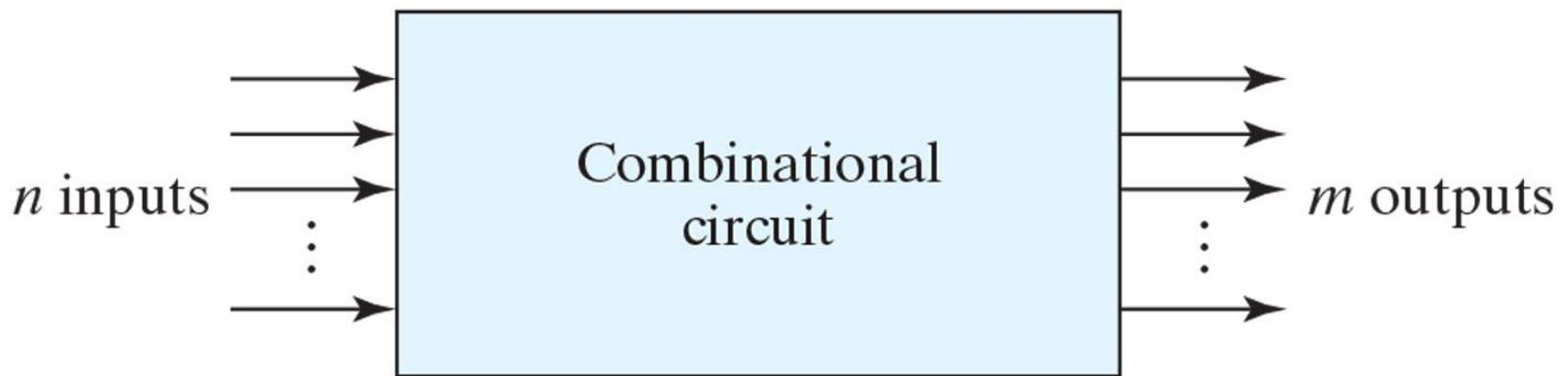
- تعریف مدارهای ترکیبی
- تحلیل مدارهای ترکیبی
- طراحی مدارهای ترکیبی
- قطعات ترکیبی پایه (کدگشا، کدگذار، تسهیم‌کننده، ...)
- جمع‌کننده، تفریق‌کننده، مقایسه‌کننده
- حافظه‌های برنامه‌پذیر
- مناظره در مدارهای ترکیبی



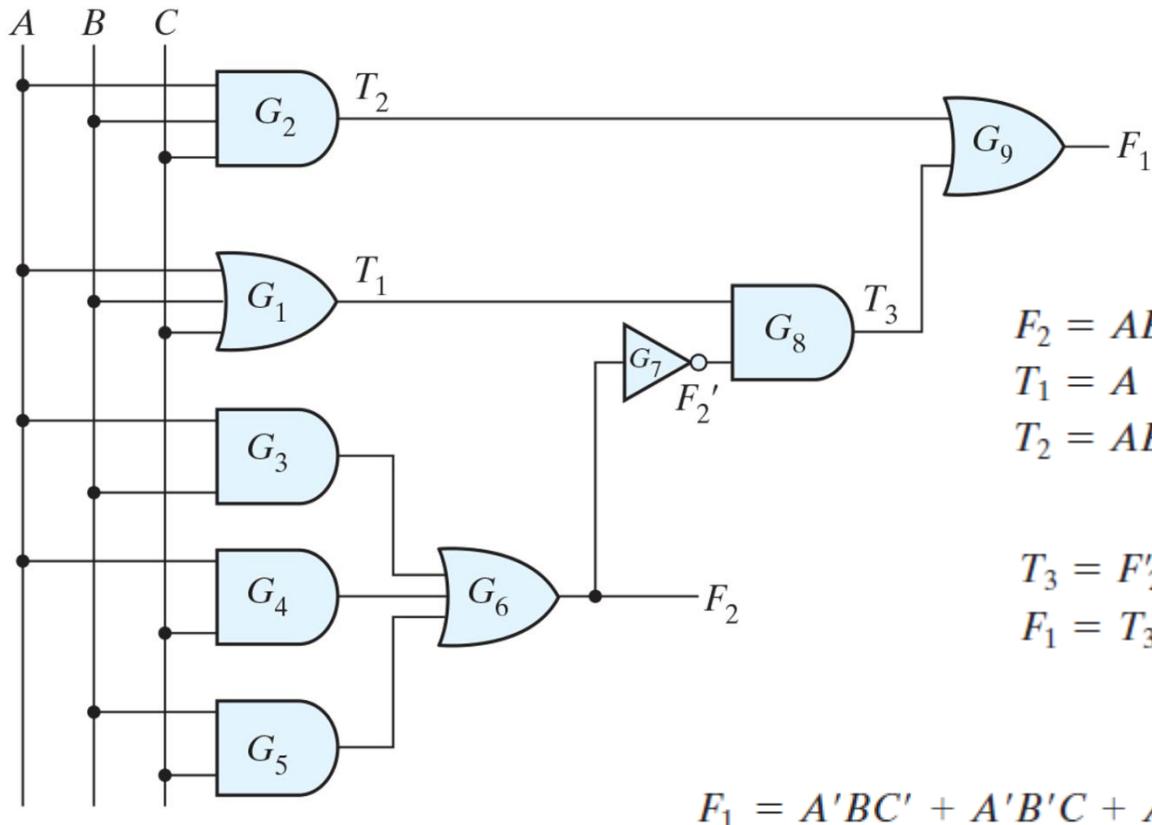
# مدارهای ترکیبی

مجموعه‌ای از گیت‌های منطقی به هم متصل که خروجی آنها در هر لحظه تابعی از ورودی‌های آنها در **همان لحظه** است و ارتباطی به ورودی‌های

قبلی ندارد



# مثال: تحلیل یک مدار نمونه



# جدول درستی مثال قبل

<b>A</b>	<b>B</b>	<b>C</b>	<b><math>F_2</math></b>	<b><math>F'_2</math></b>	<b><math>T_1</math></b>	<b><math>T_2</math></b>	<b><math>T_3</math></b>	<b><math>F_1</math></b>
0	0	0	0	1	0	0	0	0
0	0	1	0	1	1	0	1	1
0	1	0	0	1	1	0	1	1
0	1	1	1	0	1	0	0	0
1	0	0	0	1	1	0	1	1
1	0	1	1	0	1	0	0	0
1	1	0	1	0	1	0	0	0
1	1	1	1	0	1	1	0	1



# مراحل طراحی

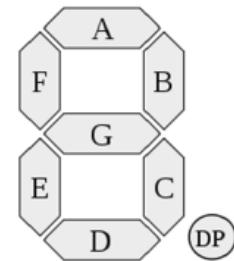
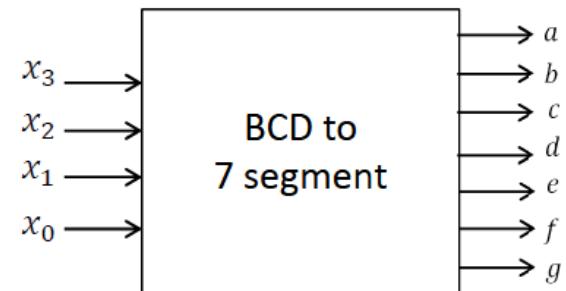
- تعریف مسئله
- تعیین تعداد ورودی‌های موجود و خروجی‌های موردنیاز
- نامگذاری ورودی‌ها و خروجی‌ها
- رسم جدول درستی برای تبیین ارتباط میان ورودی‌ها و خروجی‌ها
- به دست آوردن تابع بولی ساده شده هر خروجی
- رسم دیاگرام منطقی مدار



# مثال ۱

$x_3$	$x_2$	$x_1$	$x_0$	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0	0	1	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							

don't cares



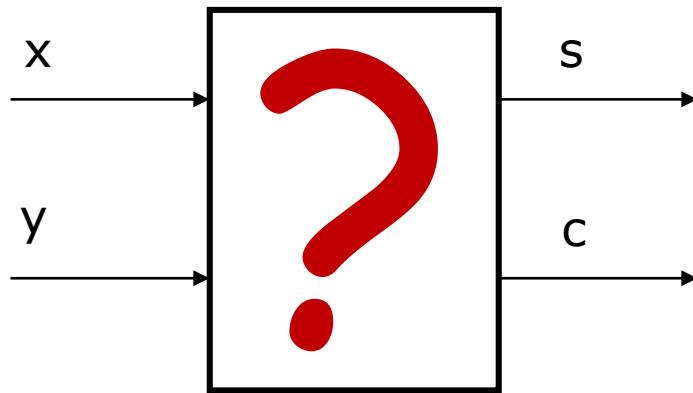
# مثال ۱

$x_3$	$x_2$	$x_1$	$x_0$	$b$
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	x
1	0	1	1	x
1	1	0	0	x
1	1	0	1	x
1	1	1	0	x
1	1	1	1	x

$x_3x_2$	0	1	11	10
0	1	1	X	1
1	1	0	X	1
11	1	1	X	X
10	1	0	X	X

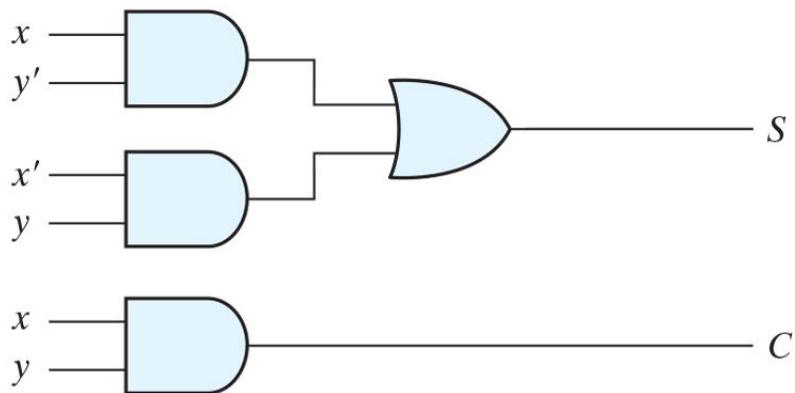
$$b = x_2' + x_1' \cdot x_0' + x_1 \cdot x_0$$

## مثال ۲ : طراحی یک مدار برای جمع دو بیت

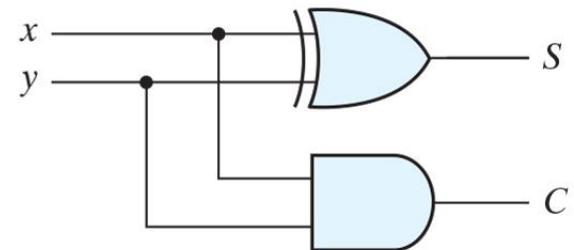


## مثال ۲ : (ادامه)

<b>x</b>	<b>y</b>	<b>c</b>	<b>s</b>
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

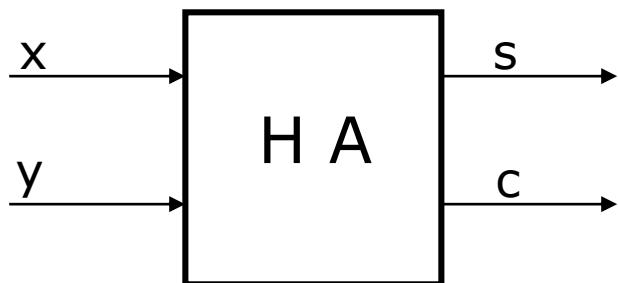


$$S = xy' + x'y \\ C = xy$$

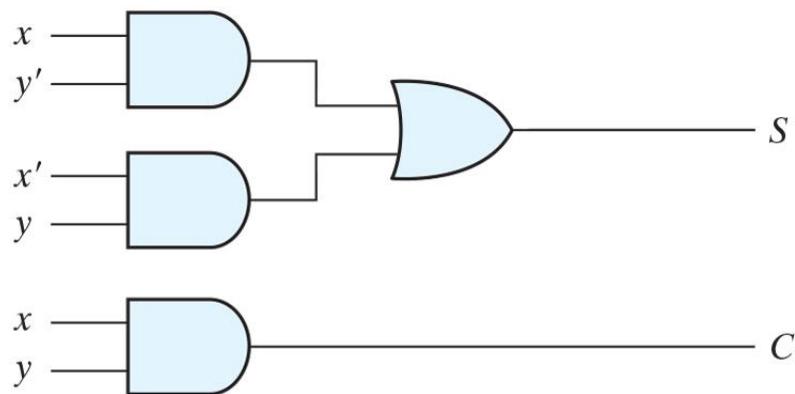


$$S = x \oplus y \\ C = xy$$

# جمع‌کننده نیمه‌افزا (Half Adder)

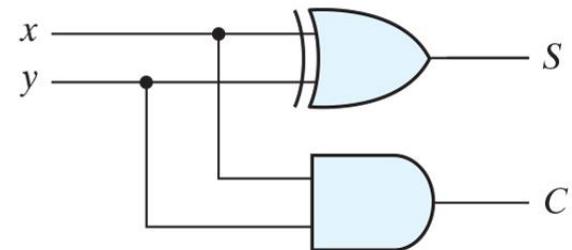


<b>x</b>	<b>y</b>	<b>c</b>	<b>s</b>
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



$$S = xy' + x'y$$

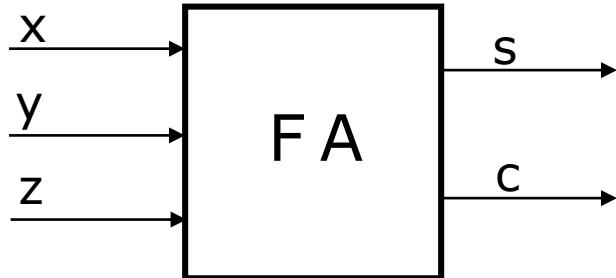
$$C = xy$$



$$S = x \oplus y$$

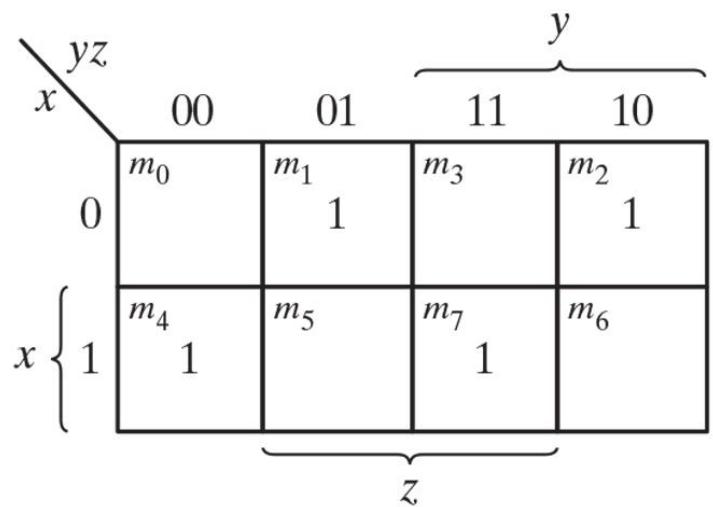
$$C = xy$$

# جمع‌کننده تمام‌افزا (Full Adder)



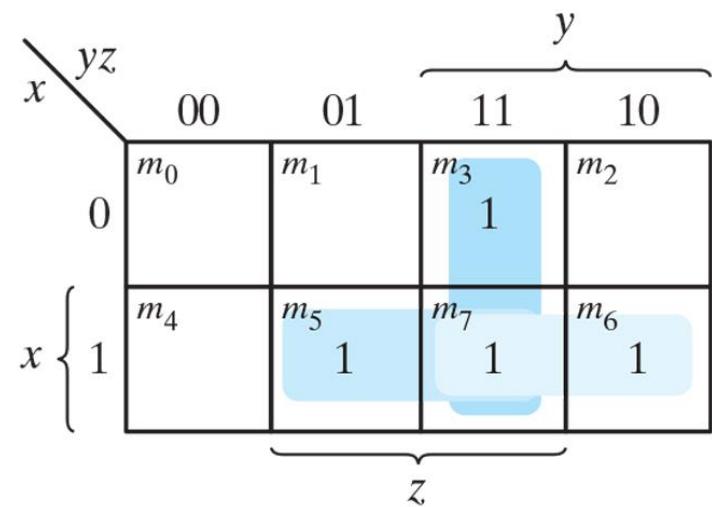
x	y	z	c	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

# جمع‌کننده تمام افزای (Full Adder)



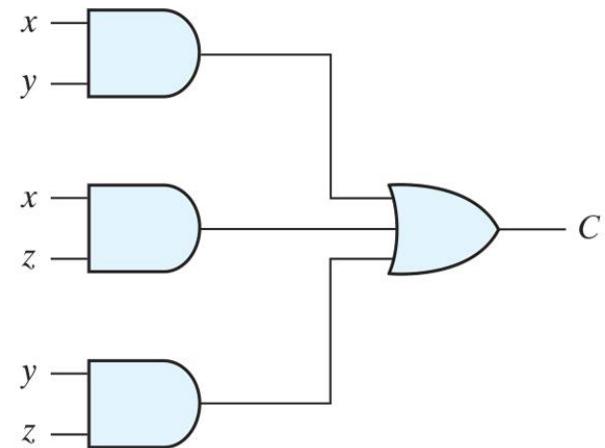
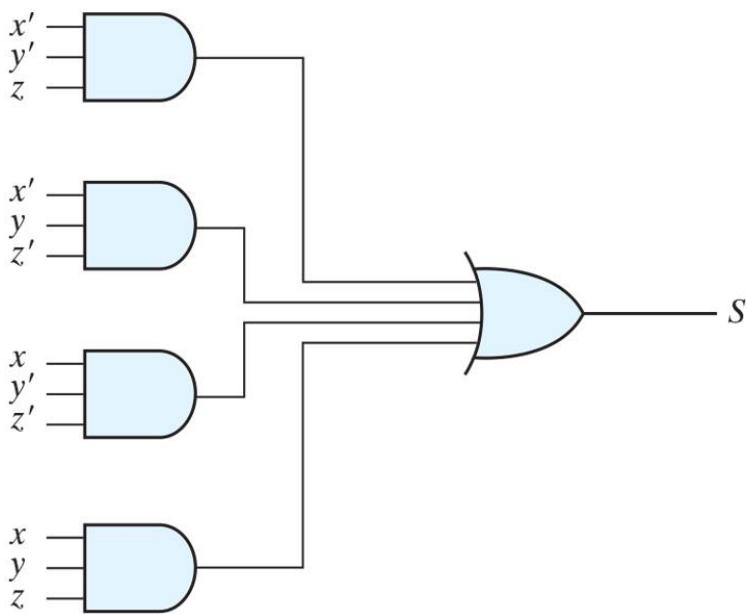
$$(a) S = x'y'z + x'yz' + xy'z' + xyz$$

$$S = x \oplus y \oplus z$$



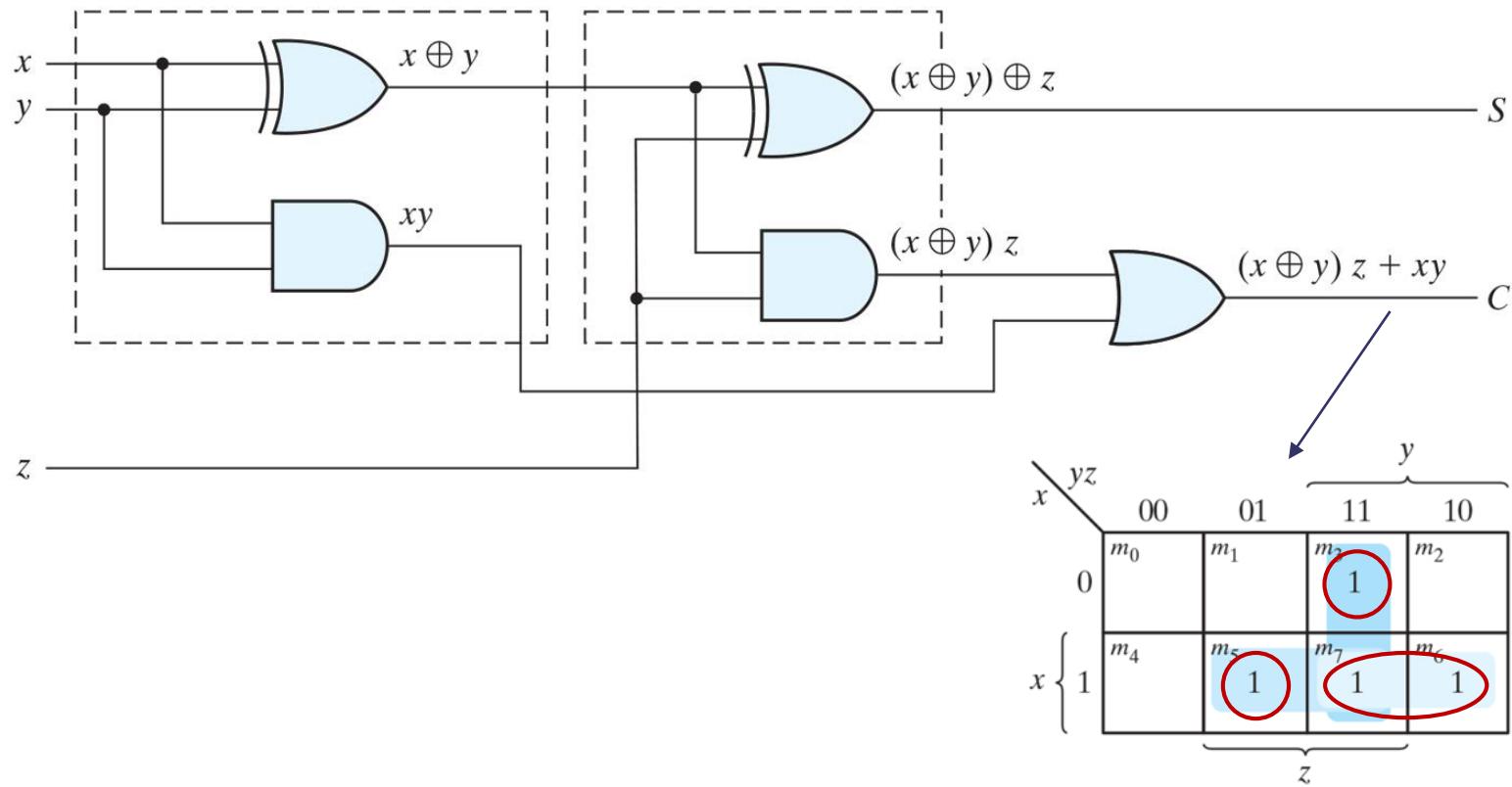
$$(b) C = xy + xz + yz$$

# جمع‌کننده تمام افزای (Full Adder)

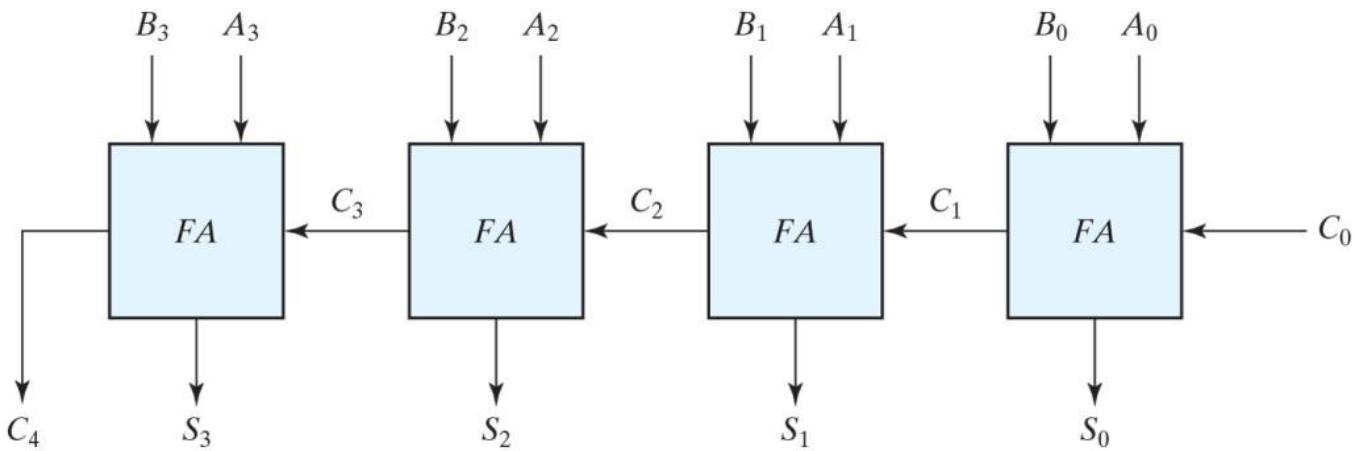


# جمع‌کننده تمام‌افزا (Full Adder)

استفاده از دو جمع‌کننده نیم‌افزا و یک گیت OR



# جمع‌گذاره پهار بیتی



<b>Subscript <math>i</math>:</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>	
Input carry	0	1	1	0	$C_i$
Augend	1	0	1	1	$A_i$
Addend	0	0	1	1	$B_i$
Sum	1	1	1	0	$S_i$
Output carry	0	0	1	1	$C_{i+1}$

# مثال: مبدل Excess-3 به BCD

<b>Input BCD</b>				<b>Output Excess-3 Code</b>			
<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>w</b>	<b>x</b>	<b>y</b>	<b>z</b>
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0



# مثال: مبدل Excess-3 به BCD

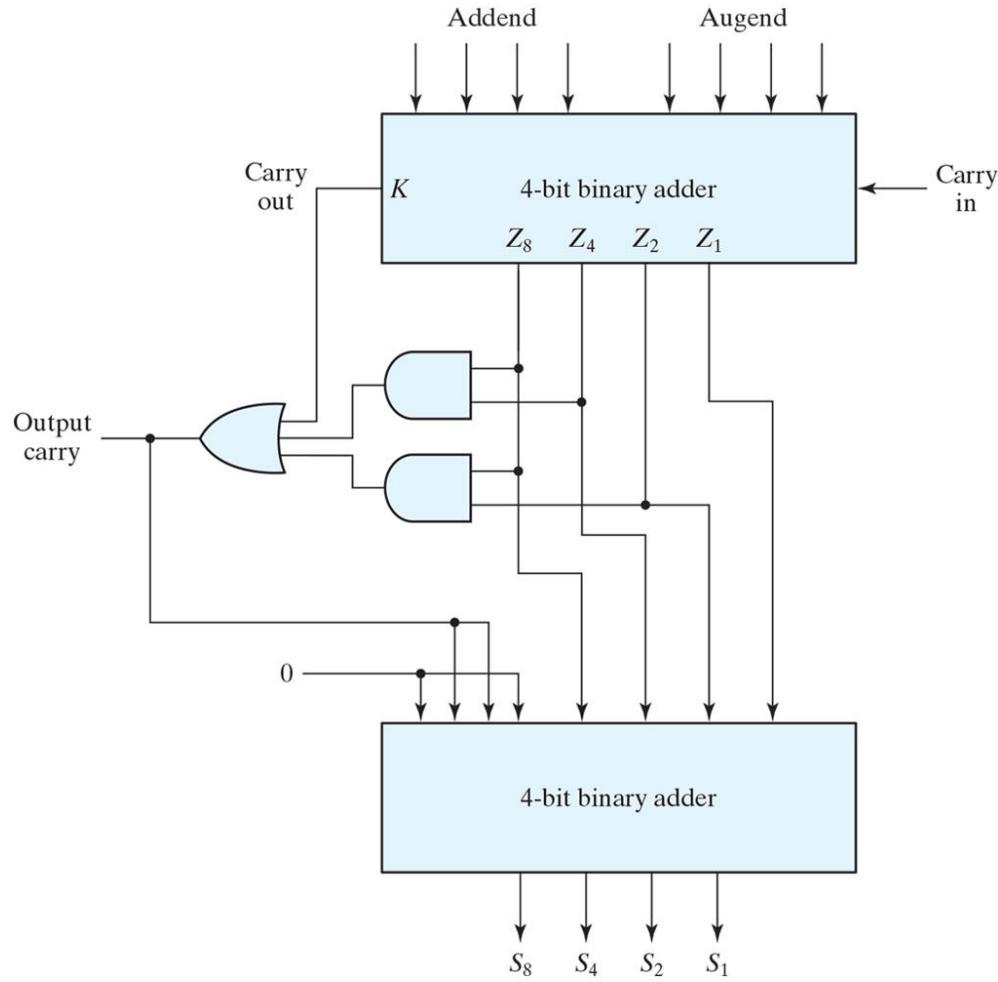


# مجموع دو رقم BCD

Binary Sum					BCD Sum					Decimal
K	Z <sub>8</sub>	Z <sub>4</sub>	Z <sub>2</sub>	Z <sub>1</sub>	C	S <sub>8</sub>	S <sub>4</sub>	S <sub>2</sub>	S <sub>1</sub>	
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9
0	1	0	1	0	1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19 = 9 + 9 + 1



# جمع دو رقم BCD (ادامه)



# (Half Subtractor) نیم تفریق کننده

$x$	$y$	$B$	$D$
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

$$D = x'y + xy'$$

$$B = x'y$$



# تفریق‌کننده کامل (Full Subtractor)

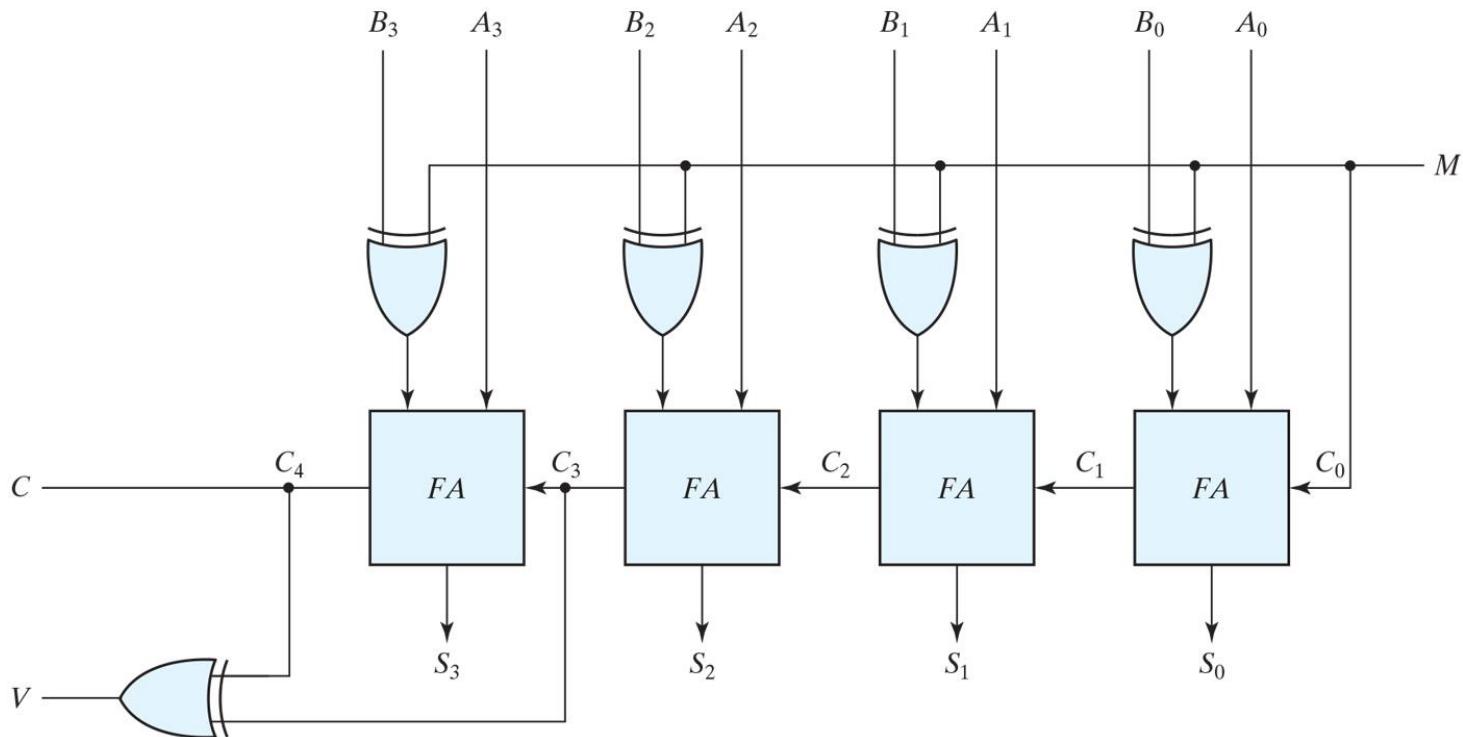
$B_i$	$x$	$y$	$B_o$	$D$
0	0	0	0	0
0	0	1	1	1
0	1	0	0	1
0	1	1	0	0
1	0	0	1	1
1	0	1	1	0
1	1	0	0	0
1	1	1	1	1

$$D = B_i \oplus x \oplus y$$

$$B_o = x'y + yB_i + x'B_i$$



# تفریق به کمک جمع



# سرویز (overflow)

سرویز وقتی رخ می‌دهد که جمع دو عدد  $n$  رقمی ( $n$  بیت)،  $n+1$  رقمی (بیت) را اشغال کند

carries:      0    1

$$\begin{array}{r}
 +70 \\
 +80 \\
 \hline
 +150
 \end{array}
 \qquad
 \begin{array}{r}
 0\ 1000110 \\
 0\ 1010000 \\
 \hline
 \textcircled{1} 0010110
 \end{array}$$

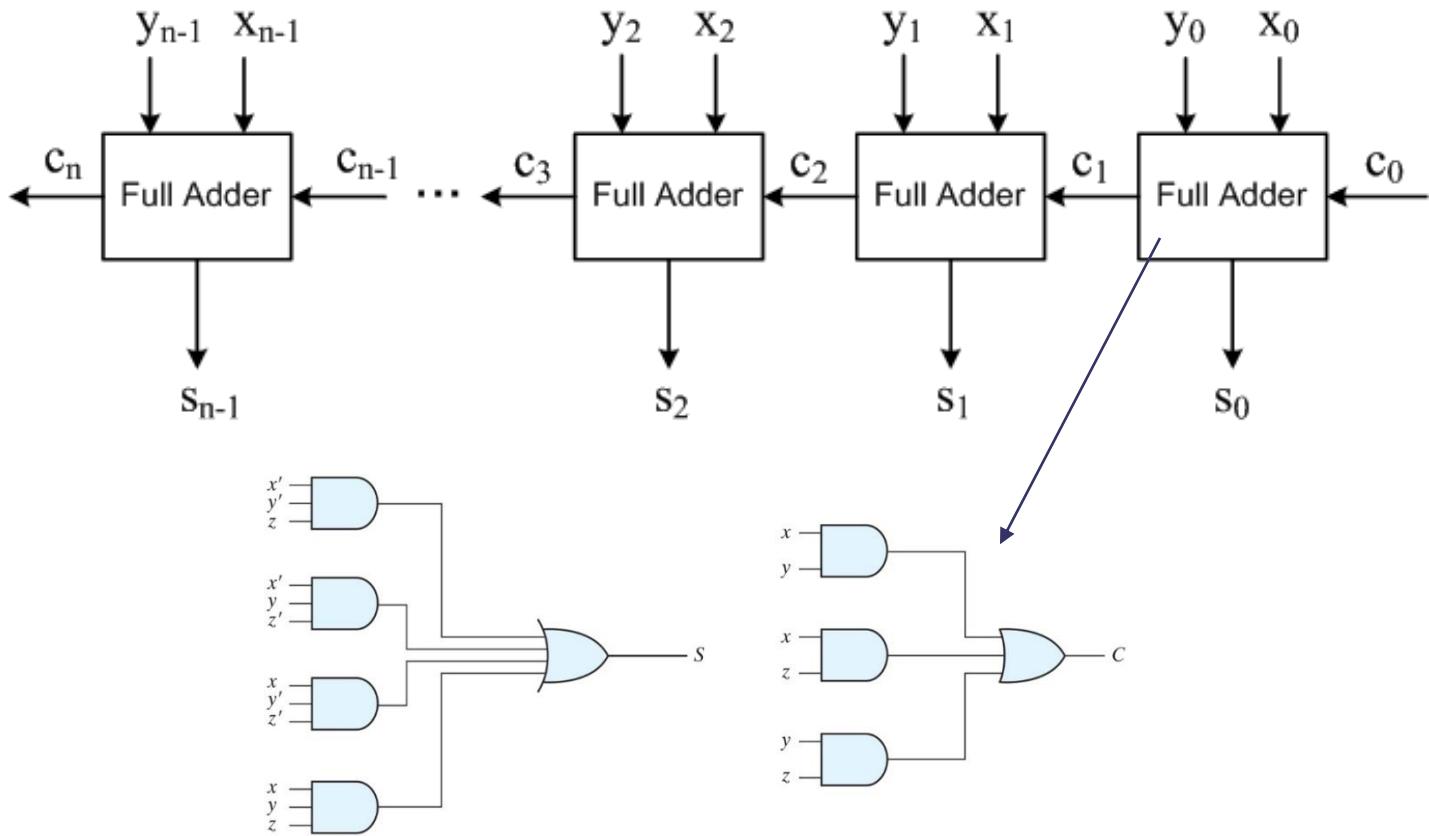
carries:      1    0

$$\begin{array}{r}
 -70 \\
 -80 \\
 \hline
 -150
 \end{array}
 \qquad
 \begin{array}{r}
 1\ 0111010 \\
 1\ 0110000 \\
 \hline
 \textcircled{0} 1101010
 \end{array}$$

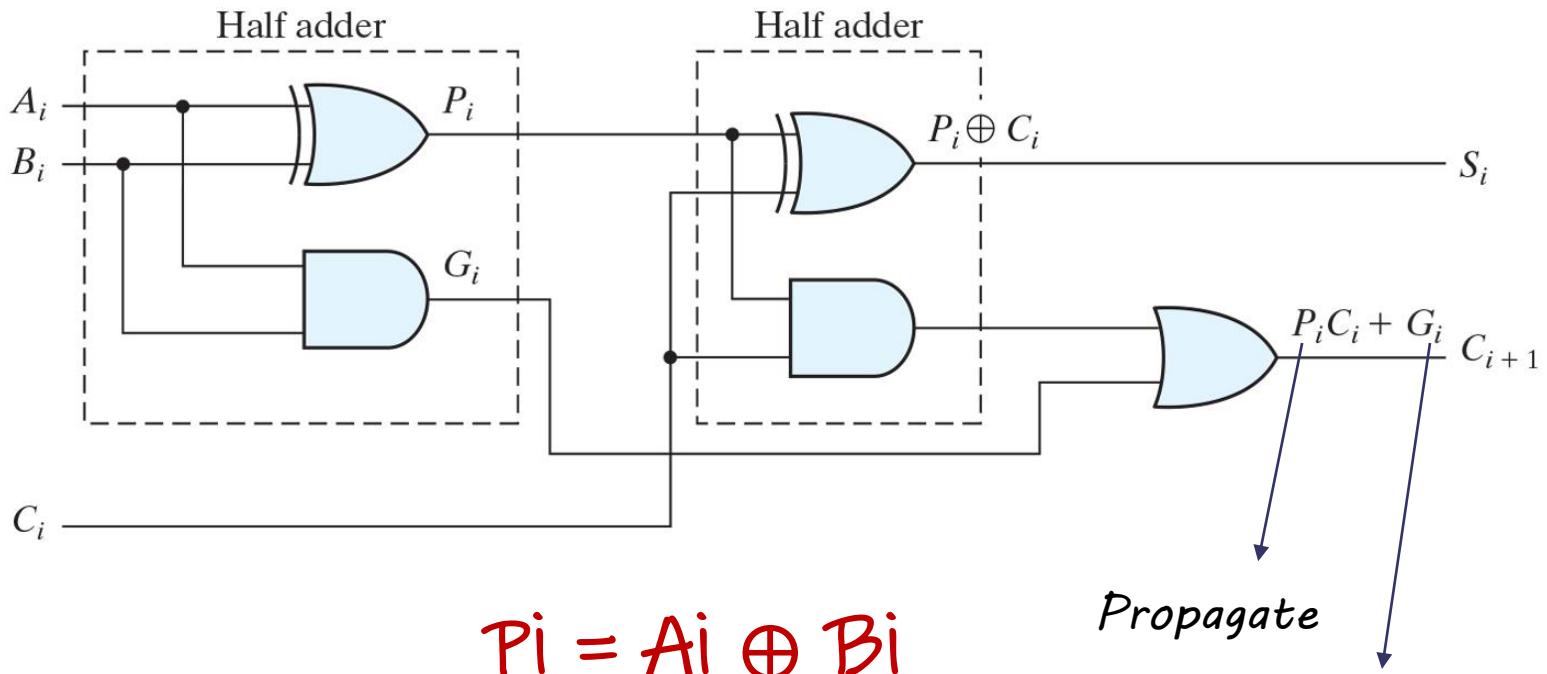
در جمع اعداد دودویی علامت‌داری که با مکمل ۲ نمایش داره می‌شوند، نشانه سرویز این است که نتیجه جمع دو عدد مثبت، منفی شود یا برعکس



# (Ripple-Carry Adder) موج‌گونه جمع‌کننده



# Carry Generate / Propagate



$$P_i = A_i \oplus B_i$$

$$G_i = A_i \cdot B_i$$

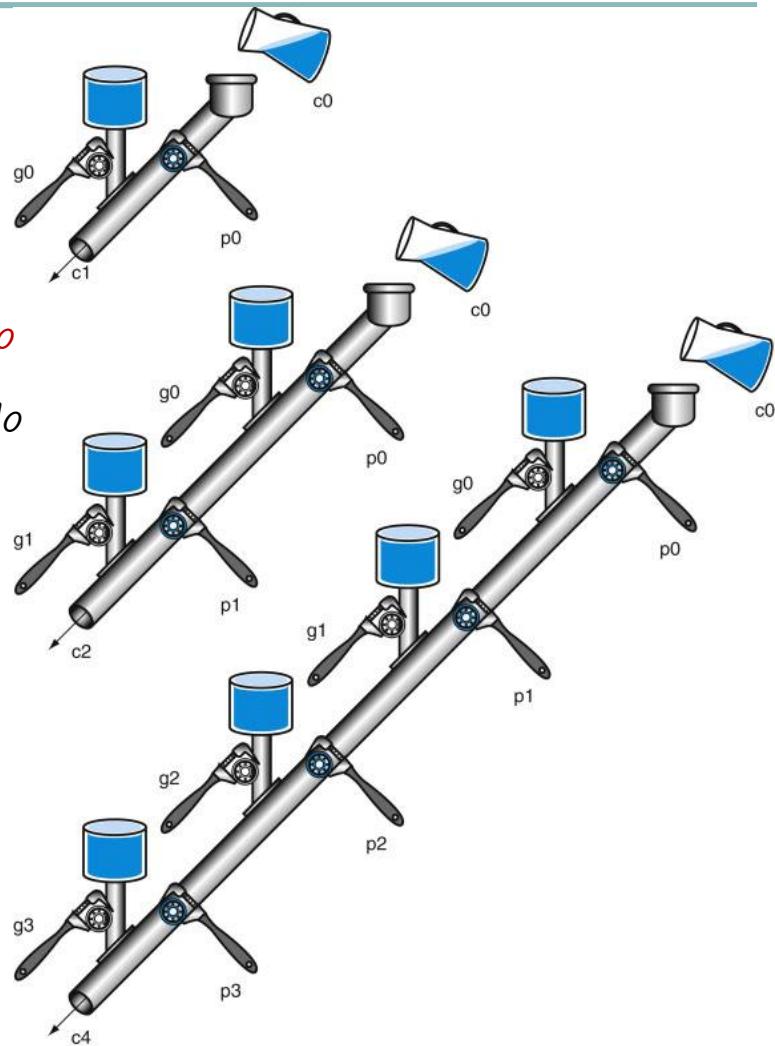
# تذاصر با لوله‌ها و شیرهای آب

$$c_1 = g_0 + p_0 \cdot c_0$$

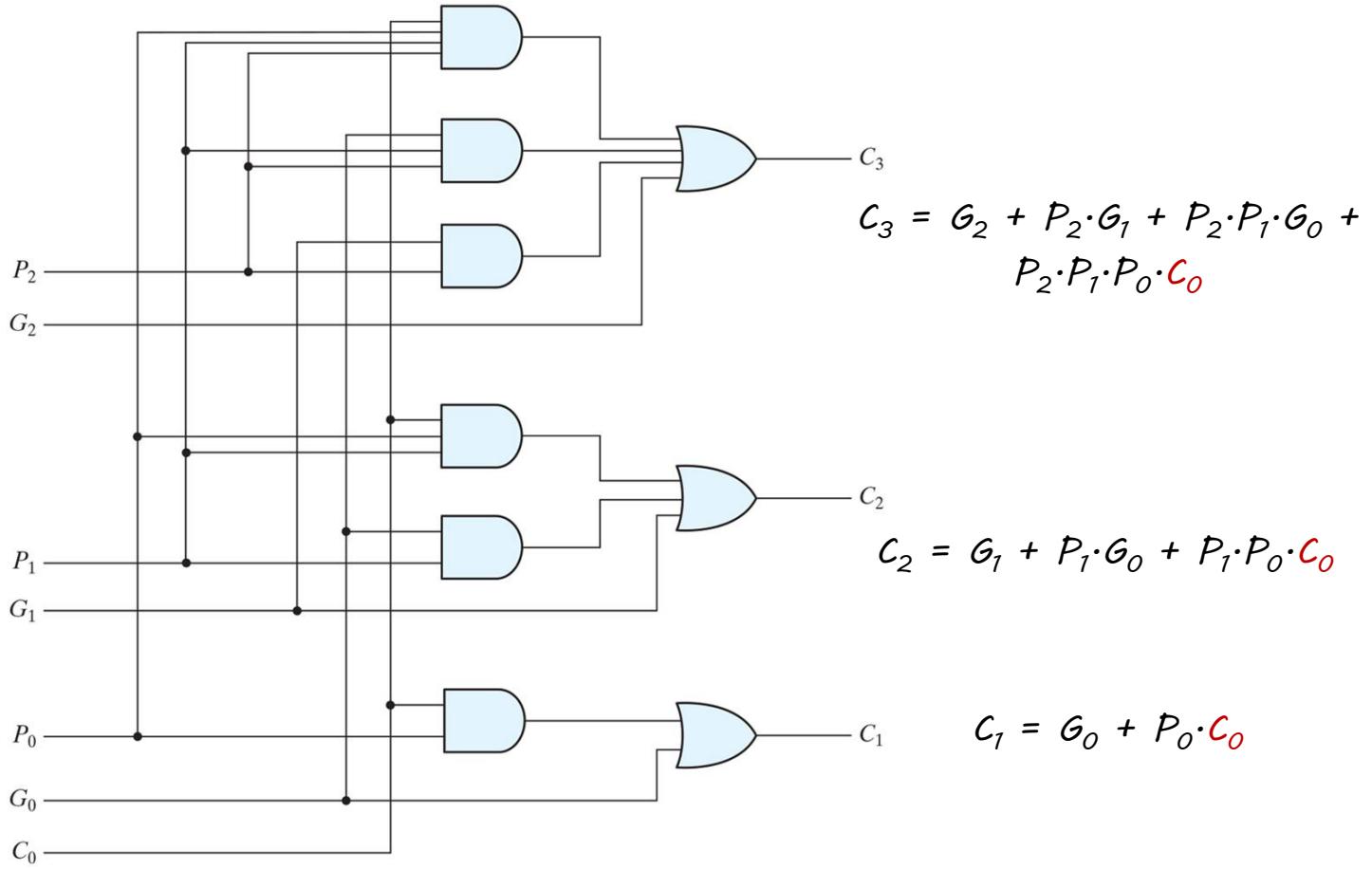
$$c_2 = g_1 + p_1 \cdot g_0 + p_1 \cdot p_0 \cdot c_0$$

$$c_3 = g_2 + p_2 \cdot g_1 + p_2 \cdot p_1 \cdot g_0 + p_2 \cdot p_1 \cdot p_0 \cdot c_0$$

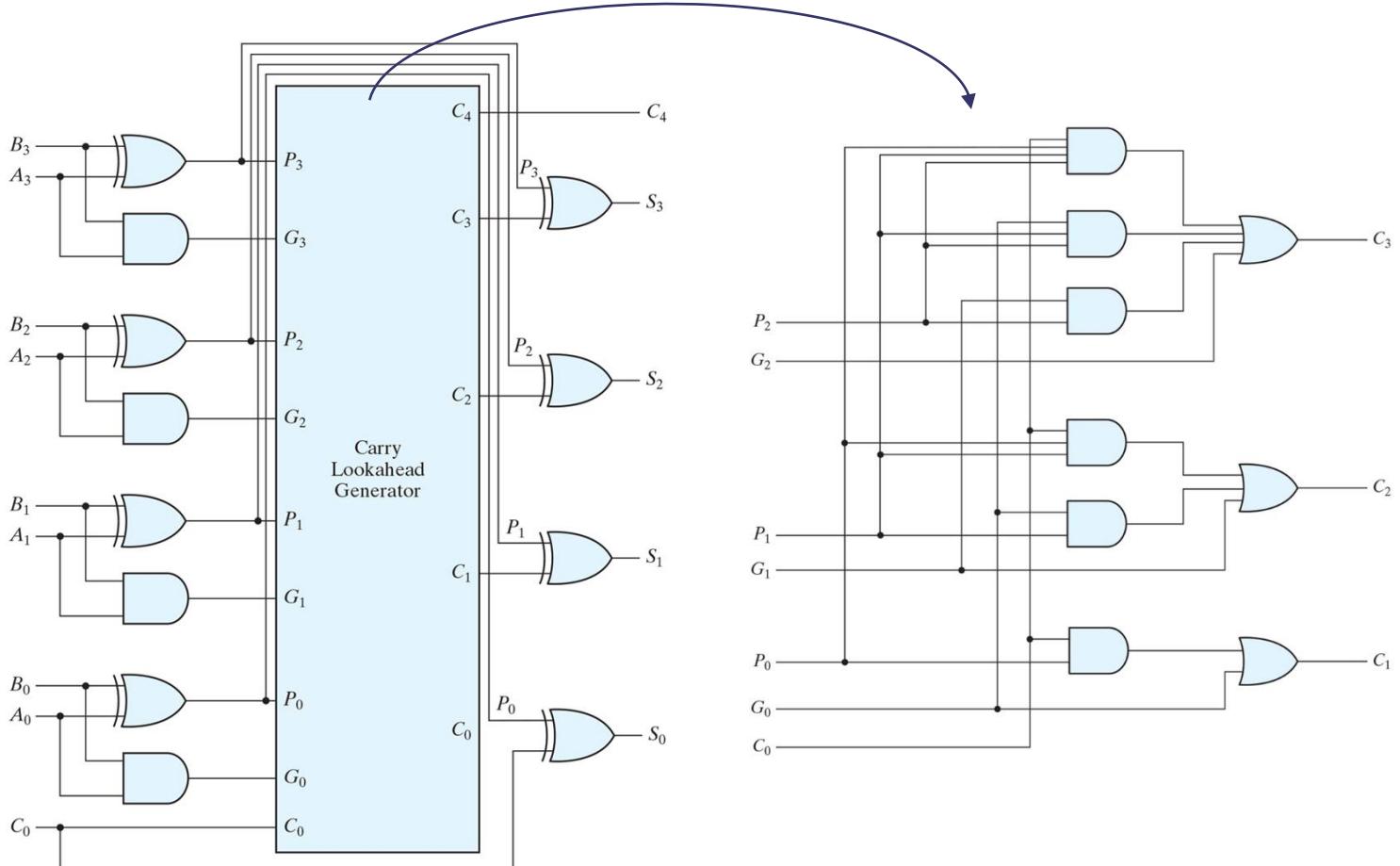
$$c_4 = g_3 + p_3 \cdot g_2 + p_3 \cdot p_2 \cdot g_1 + p_3 \cdot p_2 \cdot p_1 \cdot g_0 \\ + p_3 \cdot p_2 \cdot p_1 \cdot p_0 \cdot c_0$$



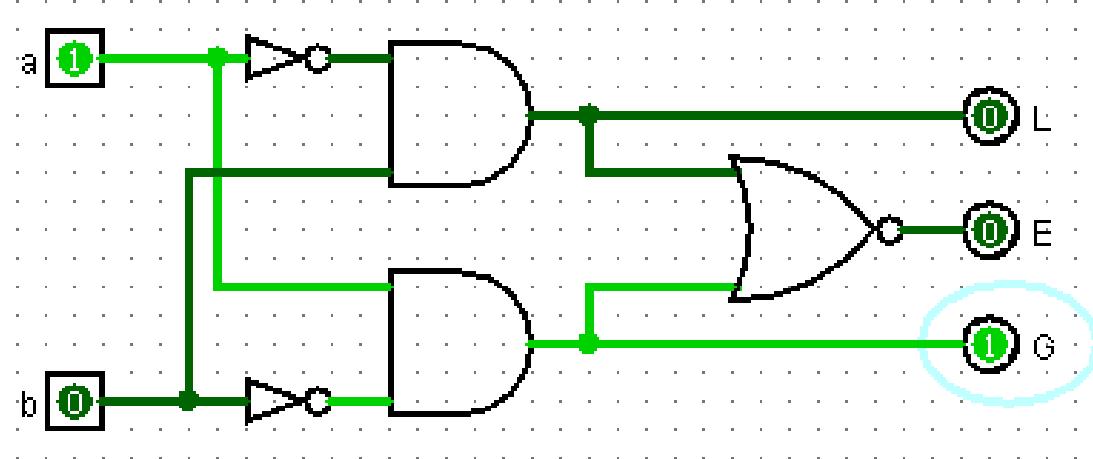
# مدار پیش‌بینی بیت نقلی Carry Look-Ahead



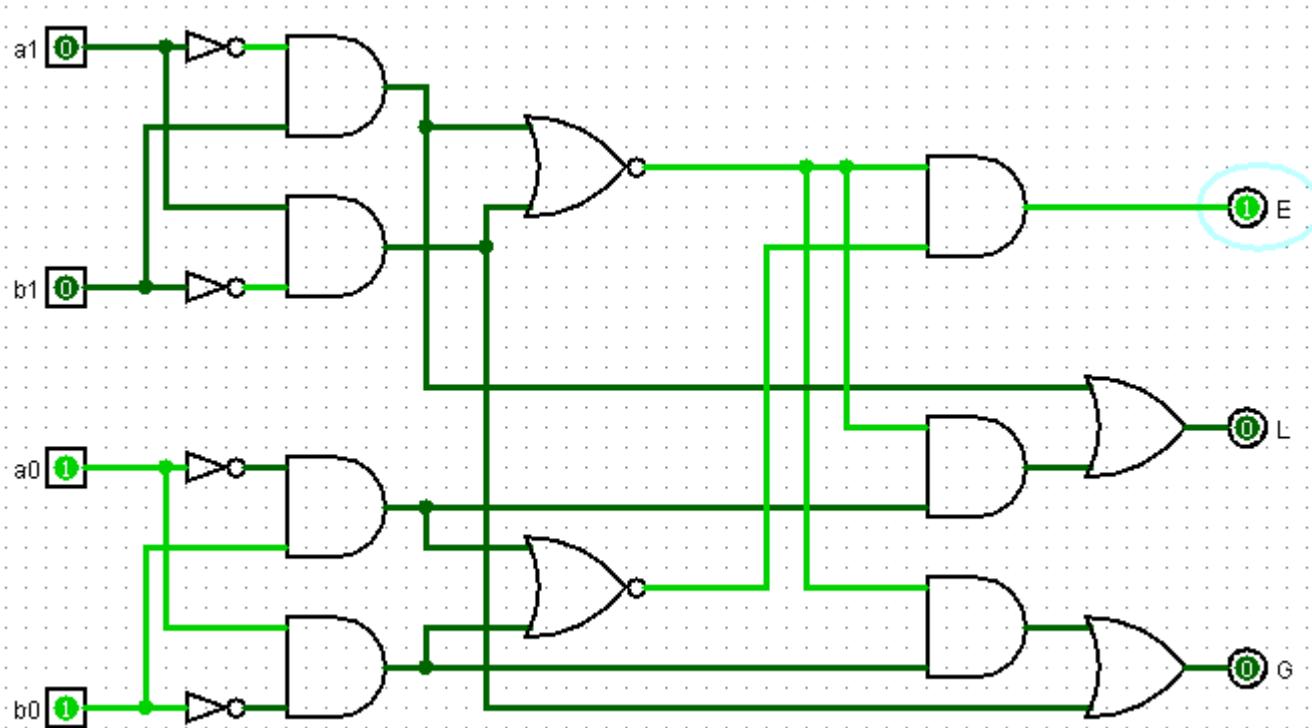
# جمع‌گننده پهار بیتی CLA



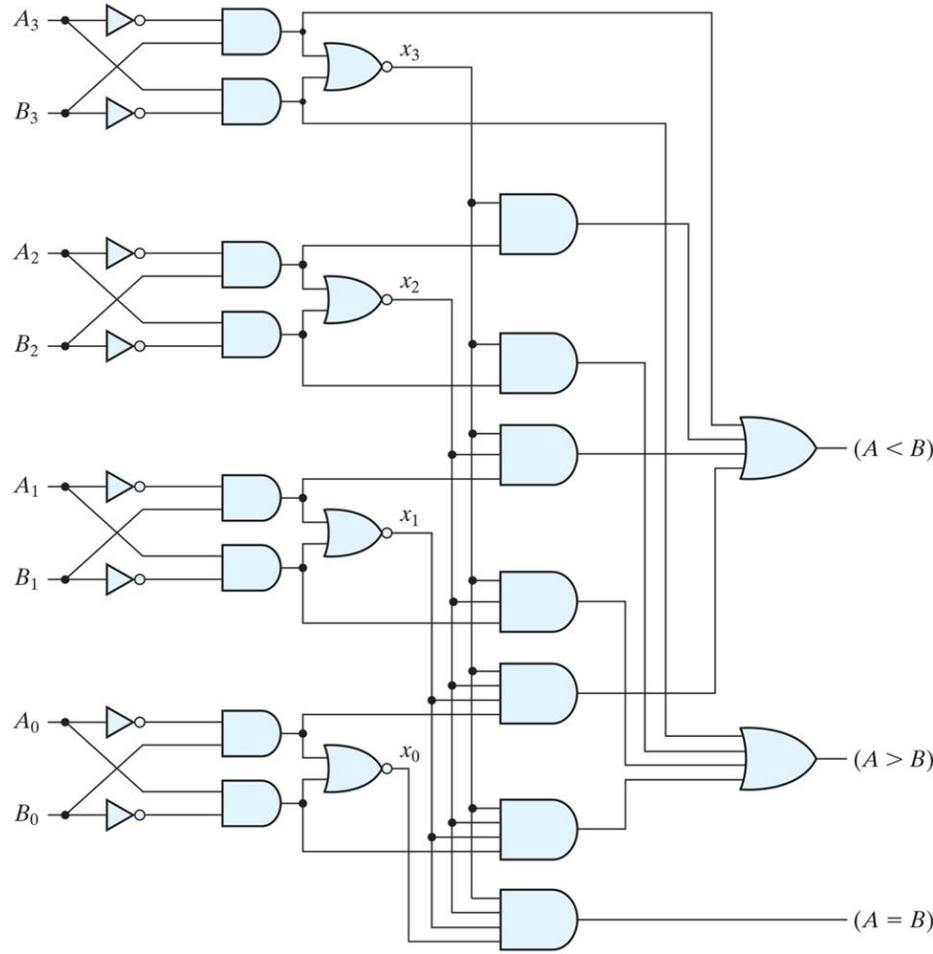
# مقایسه‌گننده یک بیتی



# مقایسه‌گننده دو بیتی

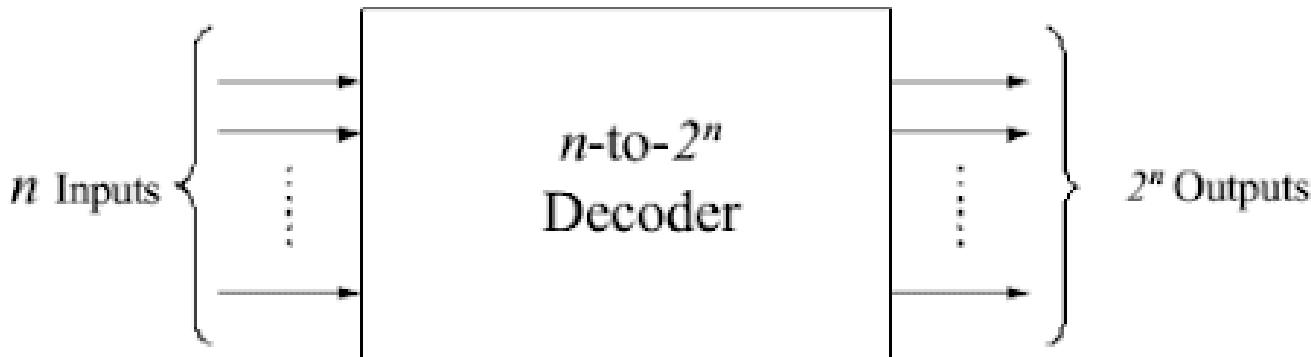


# مقایسه کننده چهار بیتی



# کدگشای (Decoder)

- یک مدلار ترکیبی که یک نگاشت بین  $n$  نمودار ورودی و  $2^n$  نمودار خروجی برقرار می‌کند
- در هر لحظه، فقط خروجی **فعال** است که شماره آن توسط خطوط ورودی مشخص شده

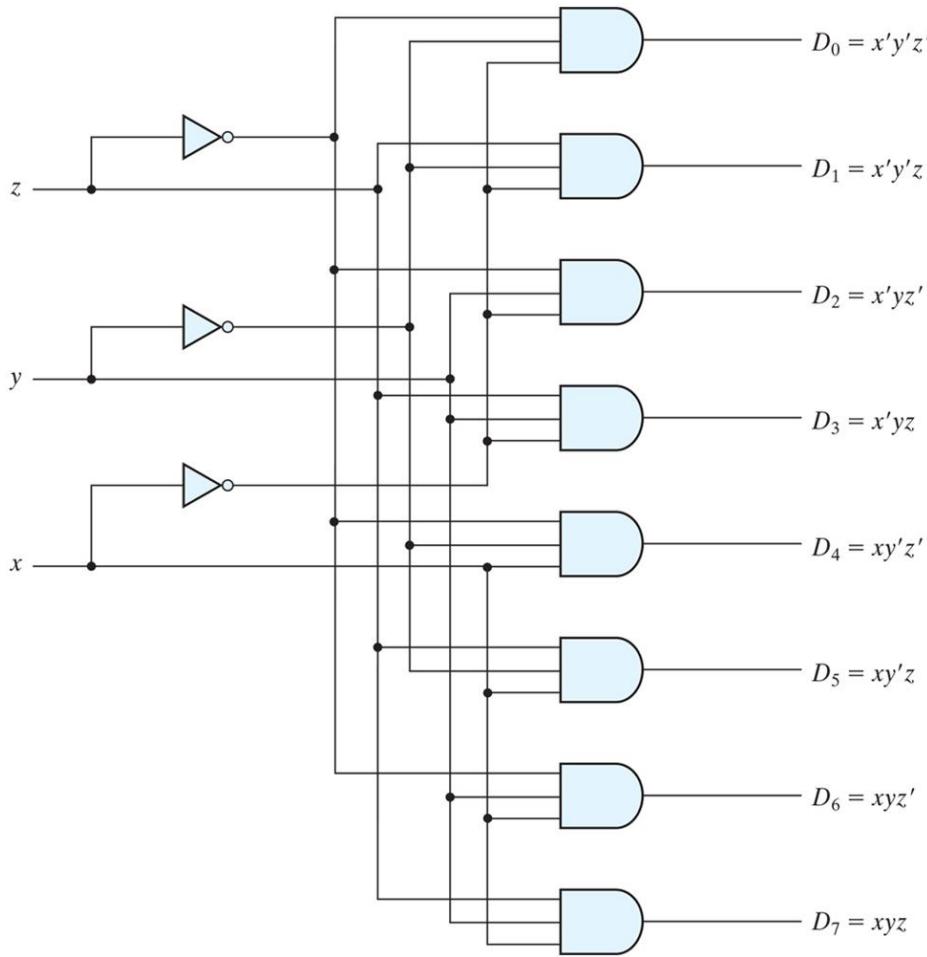


# مثال ۱ : دیکودر ۸×۲

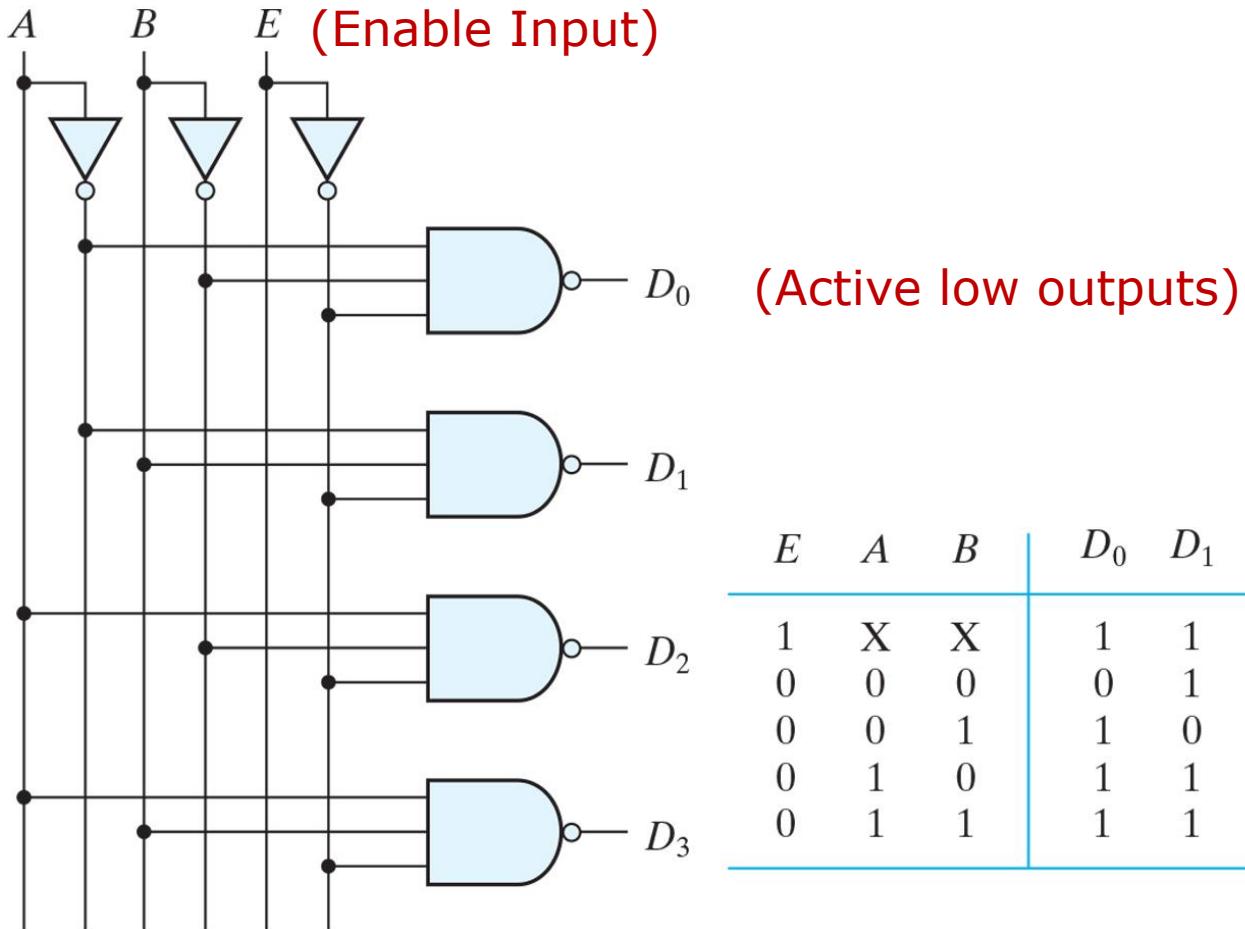
Inputs			Outputs							
$x$	$y$	$z$	$D_0$	$D_1$	$D_2$	$D_3$	$D_4$	$D_5$	$D_6$	$D_7$
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1



# مثال ۱ : دیکودر ۸ × ۳



# مثال ۲: دیکودر ۴x۱

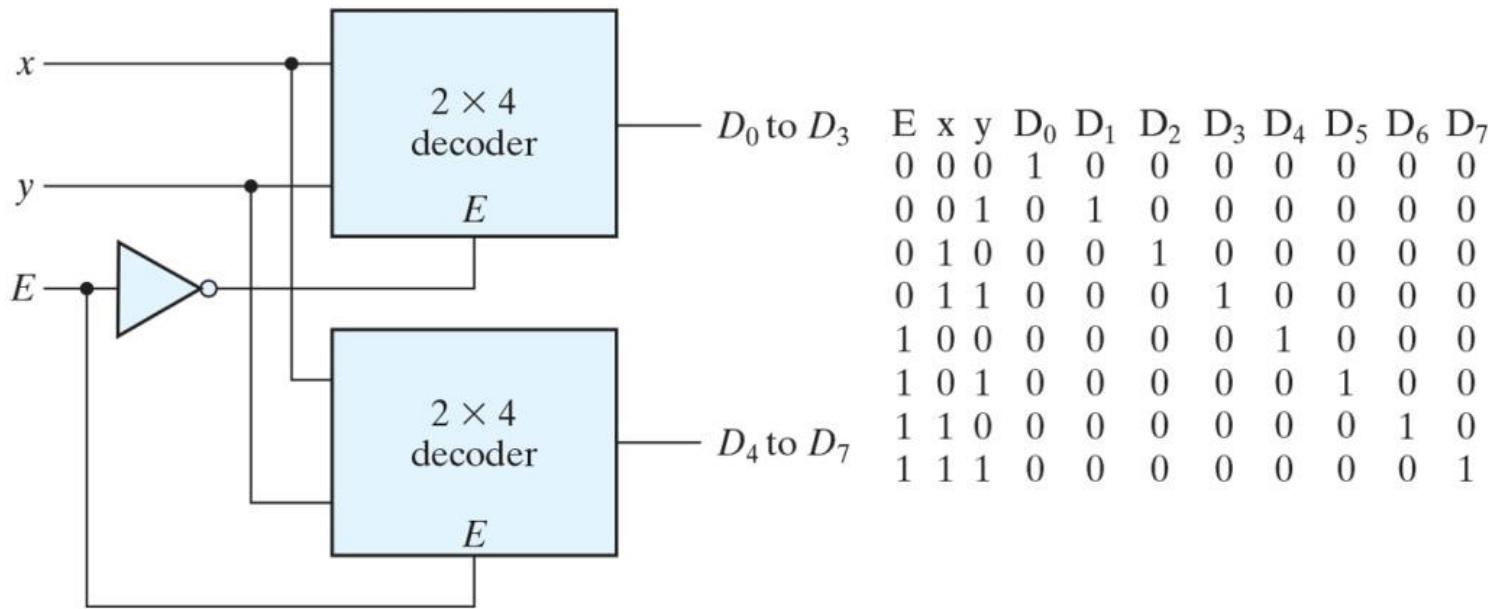


## مثال ۳

با استفاده از دو دیکودر  $16 \times 16$  یک دیکودر  $8 \times 32$  بسازید

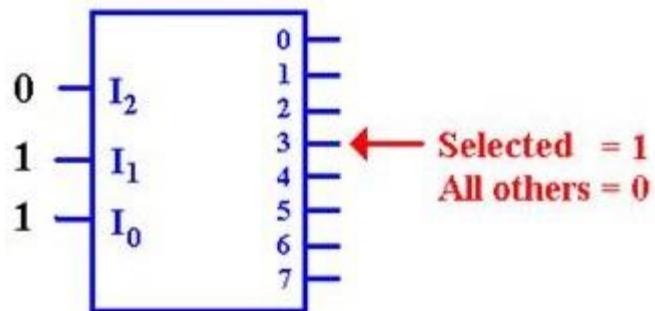


# مثال ۳



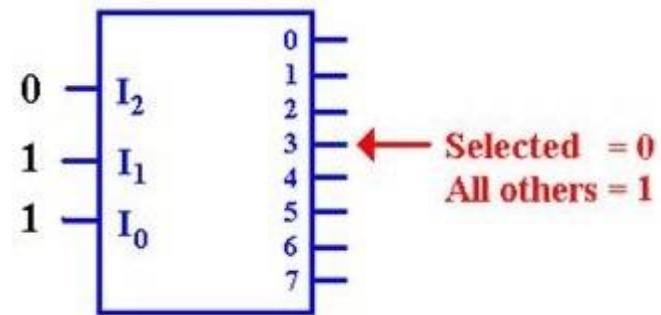
# فعال بالا / فعال پایین

Active High Outputs



Minterm Generator

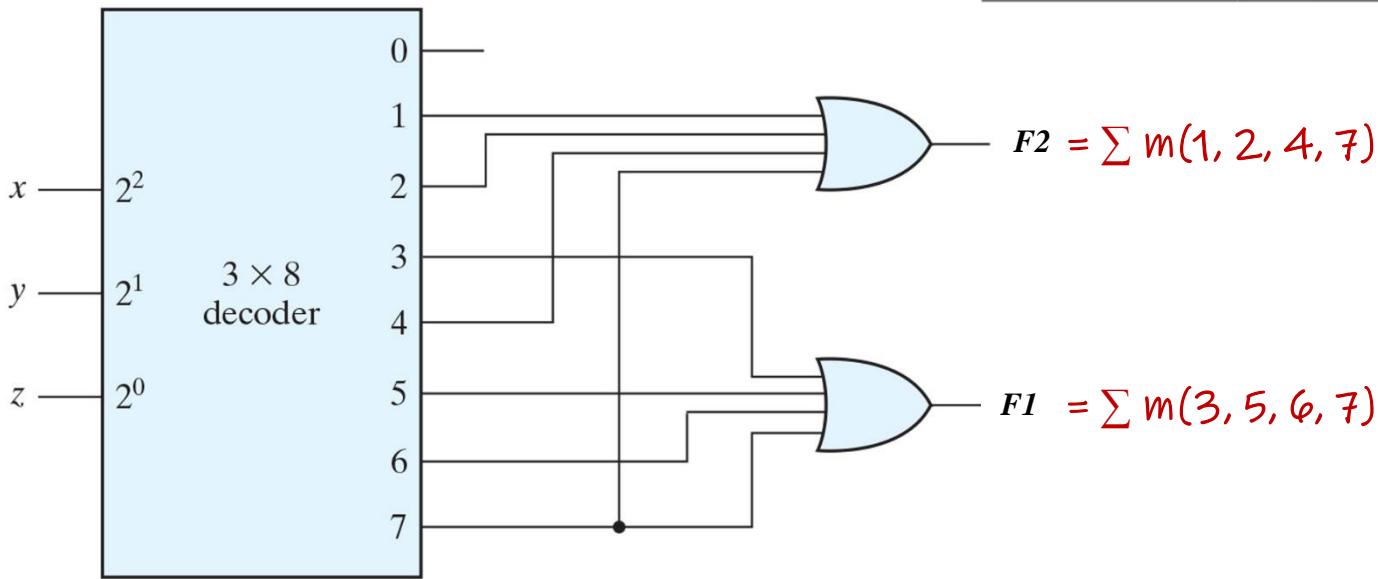
Active Low Outputs



Maxterm Generator

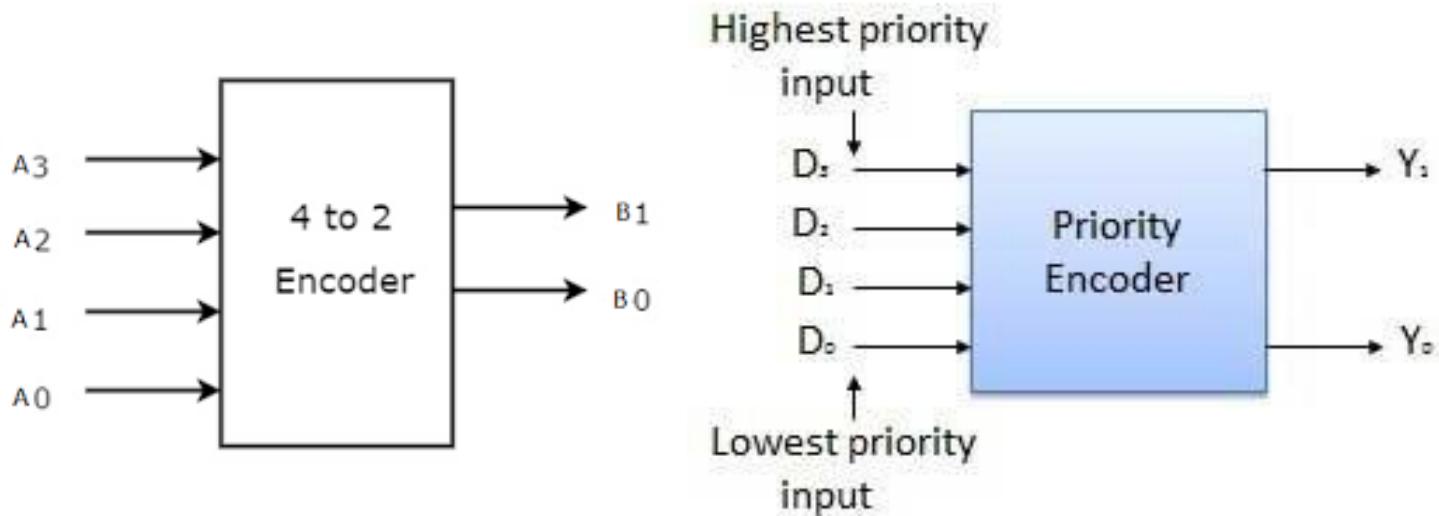
# ساخت توابع با کدگشایی

<b>x</b>	<b>y</b>	<b>z</b>	<b>F1</b>	<b>F2</b>
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

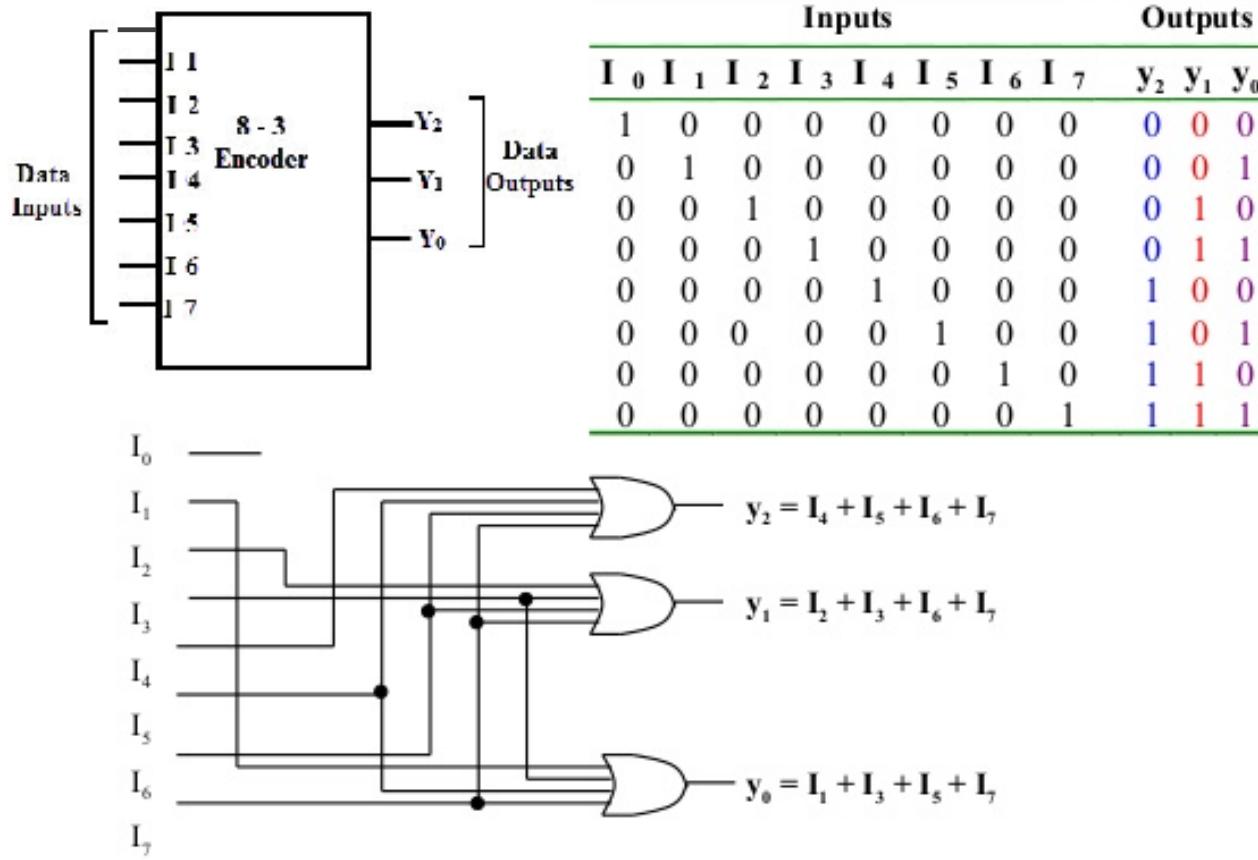


# کدگذار (Encoder)

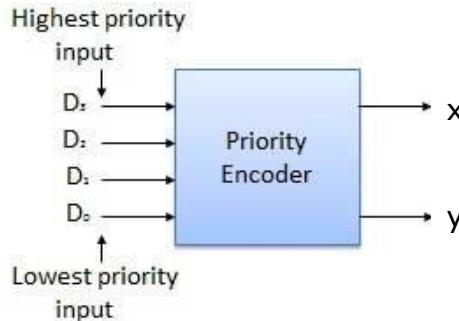
- یک مدلار ترکیبی که یک نگاشت بین  $2^n$  خروجی و  $n$  خروجی برقرار می‌کند
- در هر لحظه، خروجی شماره خروجی **فعال** ورودی را نشان می‌دهد



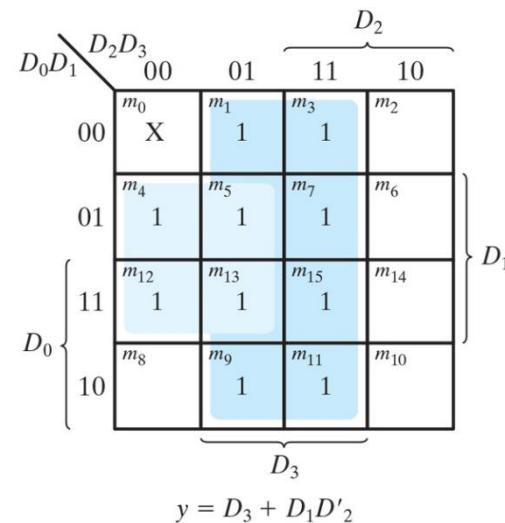
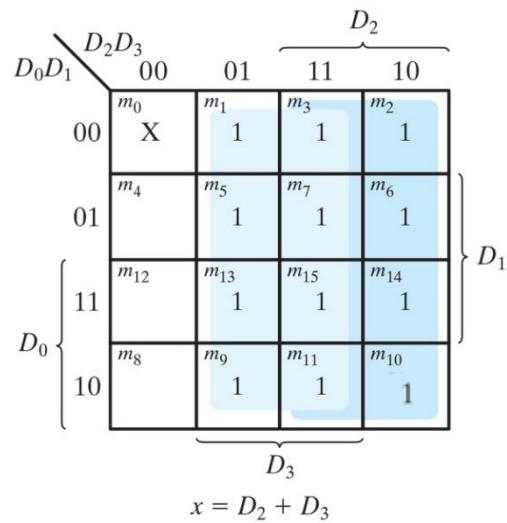
# مثال ۱ : کدگذار $3 \times 8$



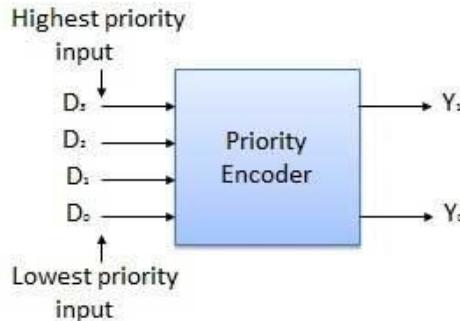
# مثال ۲: کدگذار اولویت دار $4 \times 2$



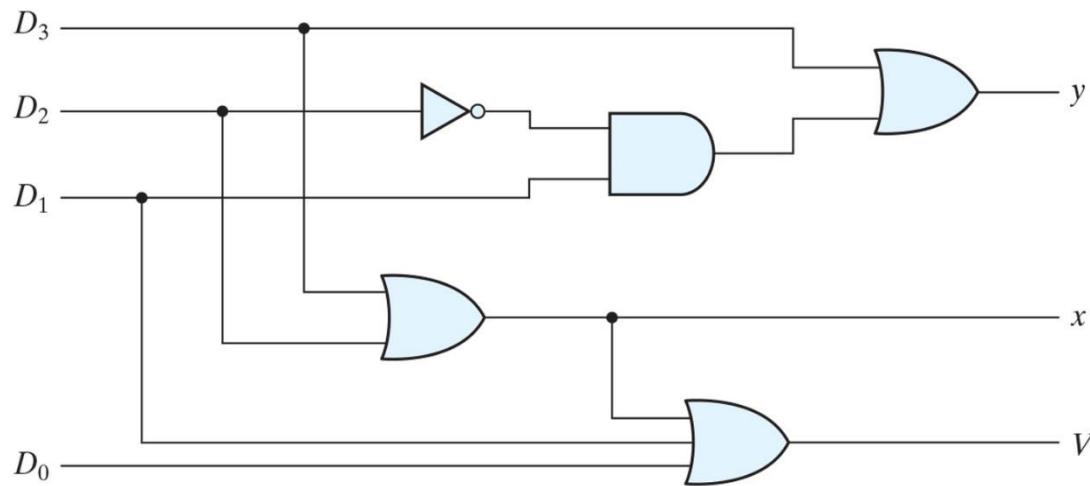
Inputs				Outputs		
D <sub>0</sub>	D <sub>1</sub>	D <sub>2</sub>	D <sub>3</sub>	x	y	V (Valid bit indicator)
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1



# مثال ۲: کدگذار اولویت دار $4 \times 2$

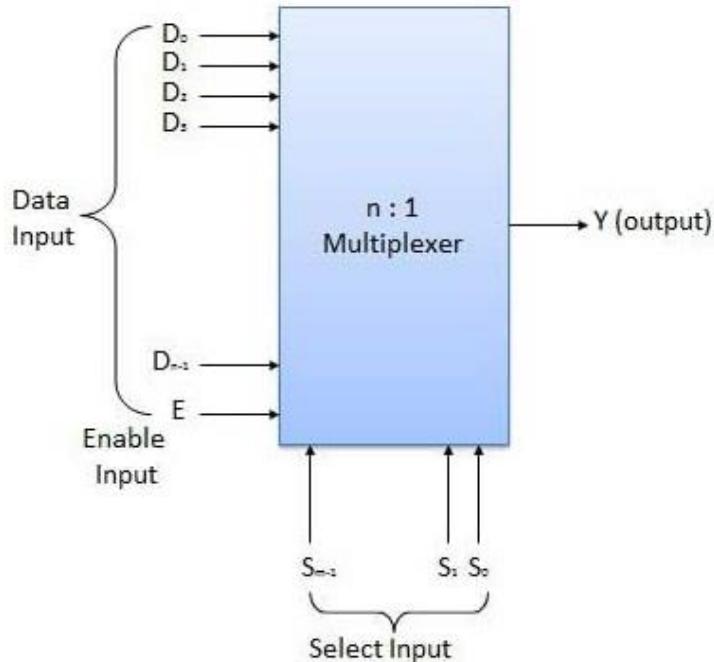


Inputs				Outputs		
$D_0$	$D_1$	$D_2$	$D_3$	$x$	$y$	$V$ (Valid bit indicator)
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1

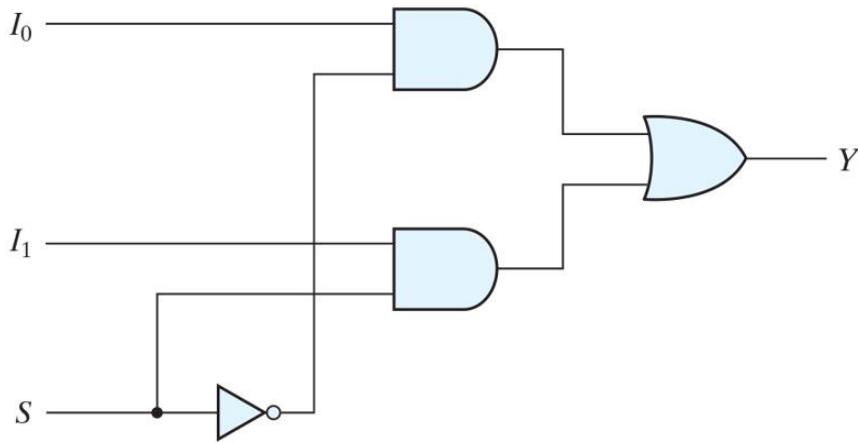


# تسهیم‌کننده (Multiplexer)

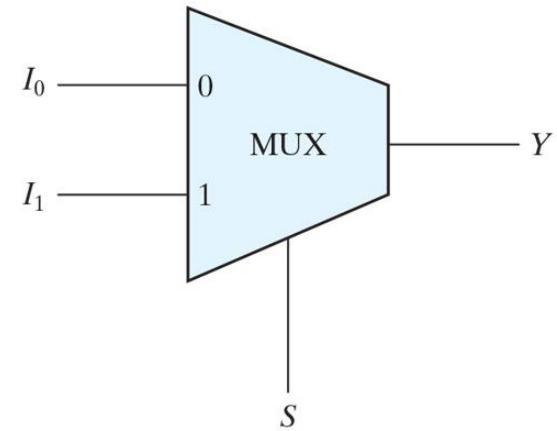
یک مدار ترکیبی که اطلاعات دودویی را از میان یکی از چند خط ورودی انتخاب کرده و آن را به یک خط خروجی هدایت می‌کند



# مثال ۱: تسویه‌یدکننده $1 \times 2$

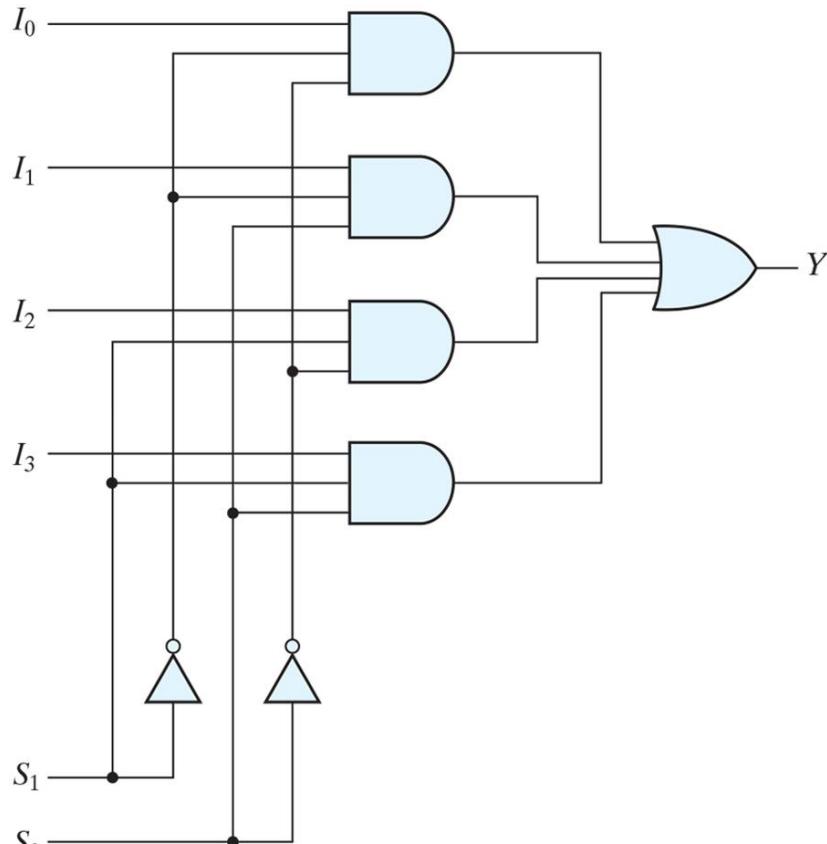


(a) Logic diagram



(b) Block diagram

# مثال ۲: تسویه‌یم کننده $1 \times 4$



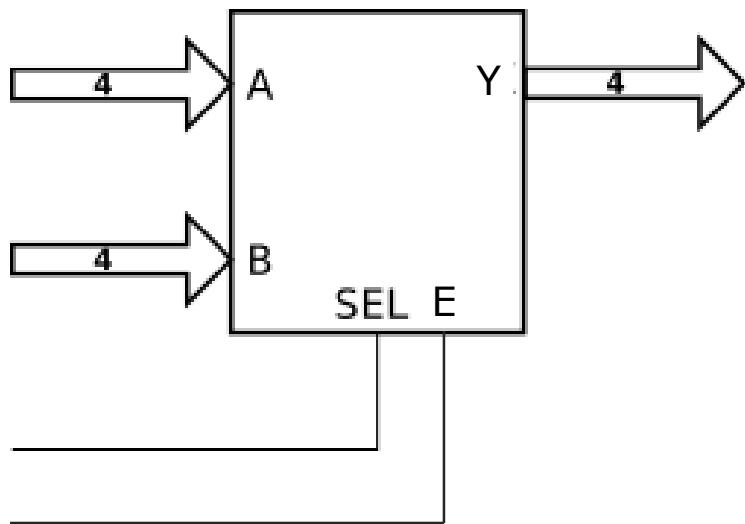
(a) Logic diagram

$S_1$	$S_0$	$Y$
0	0	$I_0$
0	1	$I_1$
1	0	$I_2$
1	1	$I_3$

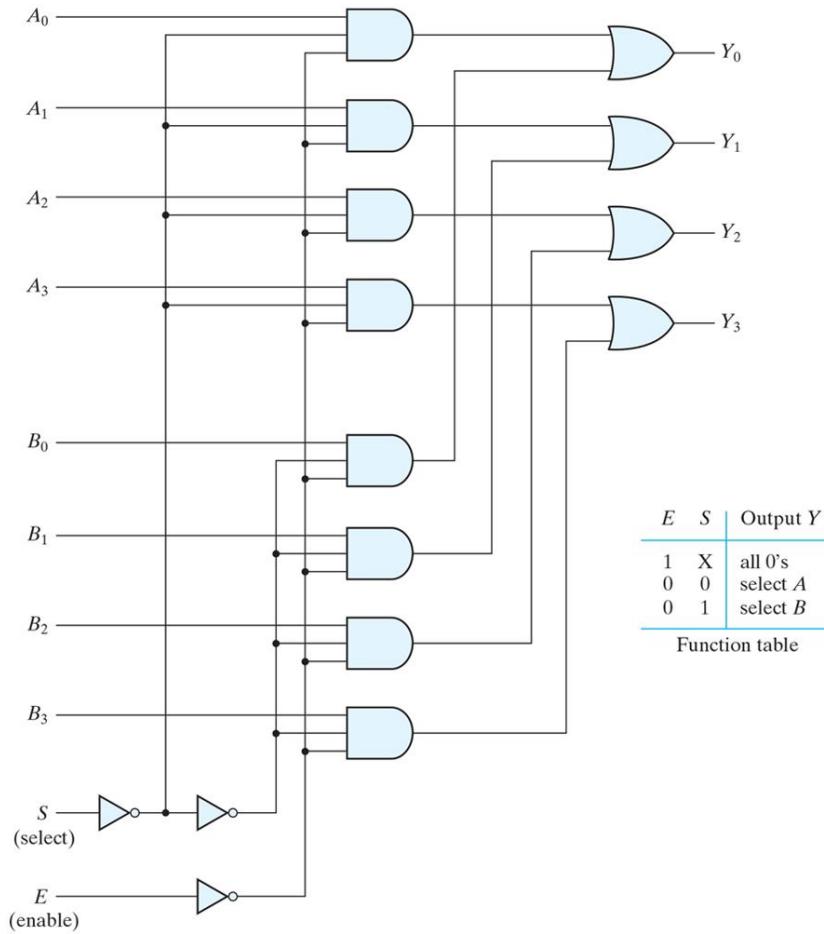
(b) Function table

# مثال ۳

یک مولتیپلکسر  $4 \times 4$  چهاربیتی بسازید



# مثال ۳

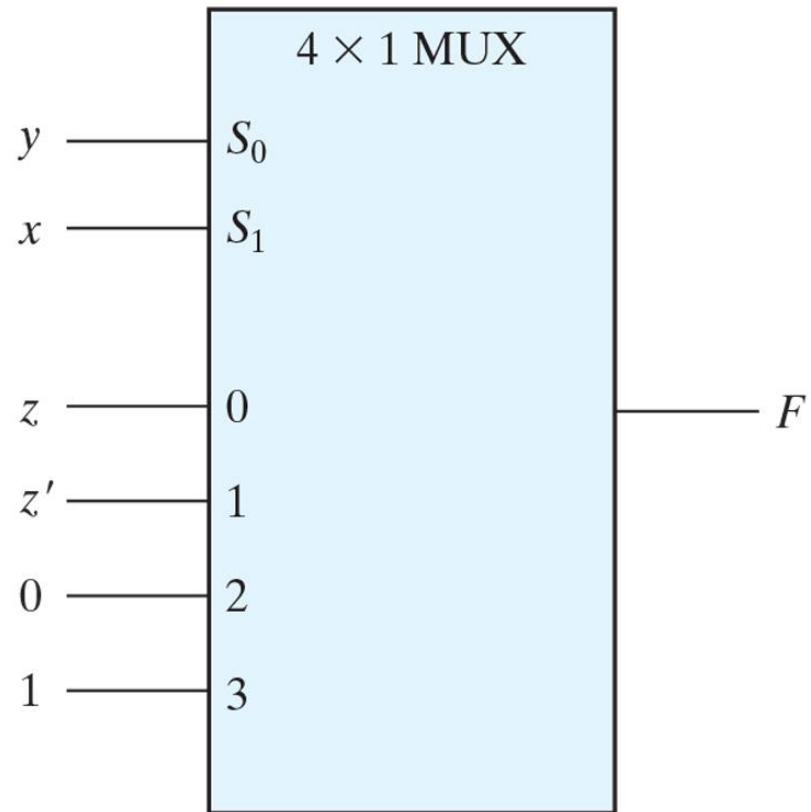


# ساخت توابع با تسهیم‌کننده

$$F(x, y, z) = \Sigma(1, 2, 6, 7)$$

$x$	$y$	$z$	$F$
0	0	0	0
0	0	1	1
<hr/>			$F = z$
0	1	0	1
0	1	1	0
<hr/>			$F = z'$
1	0	0	0
1	0	1	0
<hr/>			$F = 0$
1	1	0	1
1	1	1	1
<hr/>			$F = 1$

(a) Truth table



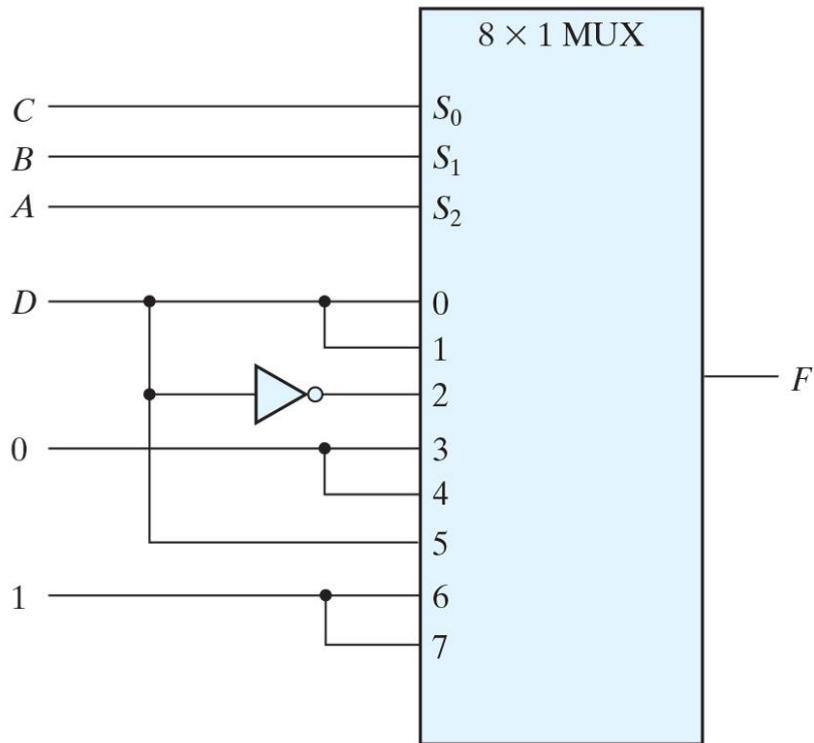
(b) Multiplexer implementation



# ساخت توابع با مولتیپلکسر

$$F(A, B, C, D) = \Sigma(1, 3, 4, 11, 12, 13, 14, 15)$$

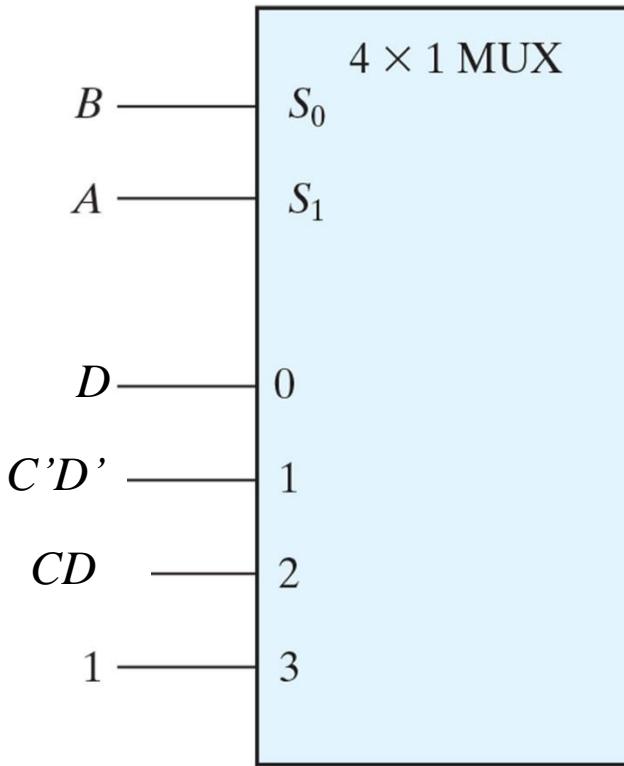
A	B	C	D	F
0	0	0	0	0 $F = D$
0	0	0	1	1
0	0	1	0	0 $F = D$
0	0	1	1	1
0	1	0	0	1 $F = D'$
0	1	0	1	0
0	1	1	0	0 $F = 0$
0	1	1	1	0
1	0	0	0	0 $F = 0$
1	0	0	1	0
1	0	1	0	0 $F = D$
1	0	1	1	1
1	1	0	0	1 $F = 1$
1	1	0	1	1
1	1	1	0	1 $F = 1$
1	1	1	1	1



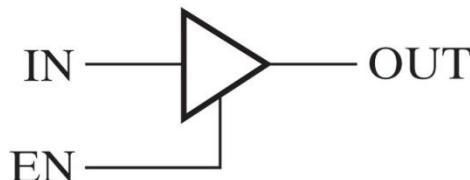
# ساخت توابع با مولتیپلکسر

A	B	C	D	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
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

$$F(A, B, C, D) = \Sigma(1, 3, 4, 11, 12, 13, 14, 15)$$



# بافر سه‌حالت (Tri-state)



(a) Logic symbol

<b>EN</b>	<b>IN</b>	<b>OUT</b>
0	X	Hi-Z
1	0	0
1	1	1

(b) Truth table

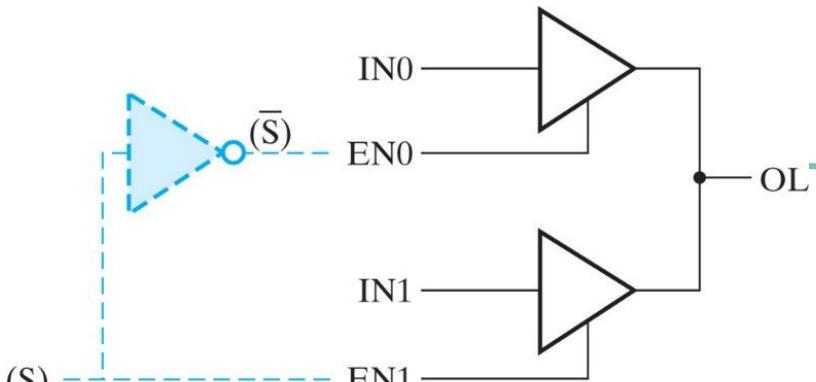
○ خروجی آمپلیتود بالا (Z):

- مثل یک مدار باز عمل می‌کند

- هیچ معنای منطقی ندارد

- اثری بر روی مدار متصل به آن ندارد

# خروجی بافر سه حالت



(a) Logic Diagram

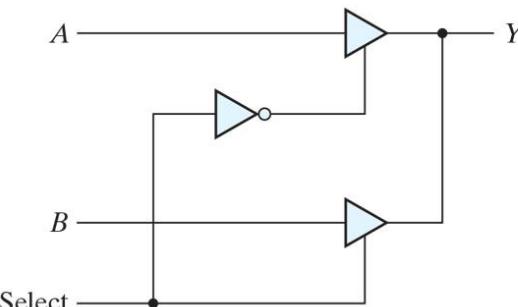
EN1	EN0	IN1	IN0	OL
0	0	X	X	Hi-Z
(S)	0	(S̄)	1	X 0
0	1	X	1	1
1	0	0	X	0
1	0	1	X	1
1	1	0	0	0
1	1	1	1	1
1	1	0	1	
1	1	1	0	

(b) Truth table

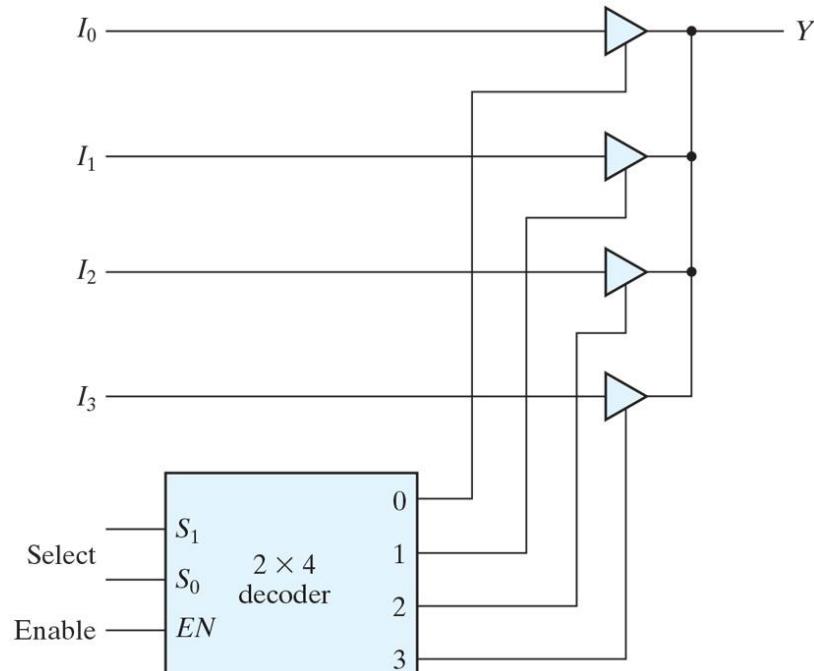
Output is don't care  
since it has no value

Output is don't care  
since it is prohibited

# ساخت تسوییم‌کننده با بافر سه‌حاله



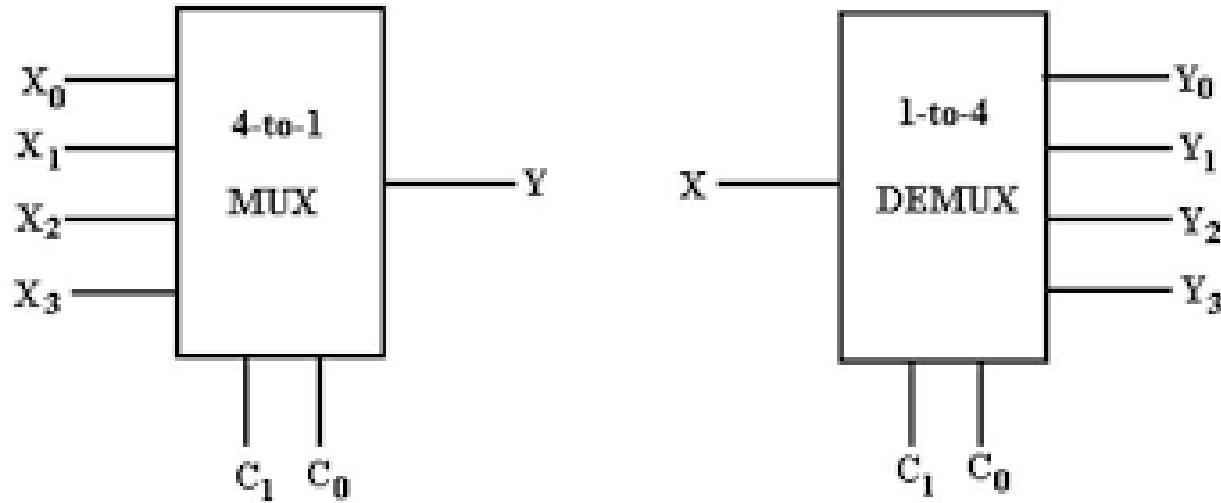
(a) 2-to-1-line mux



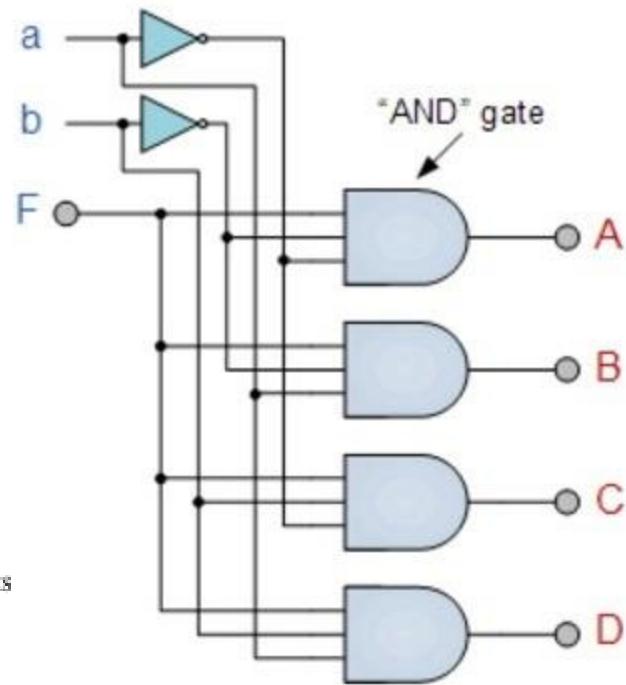
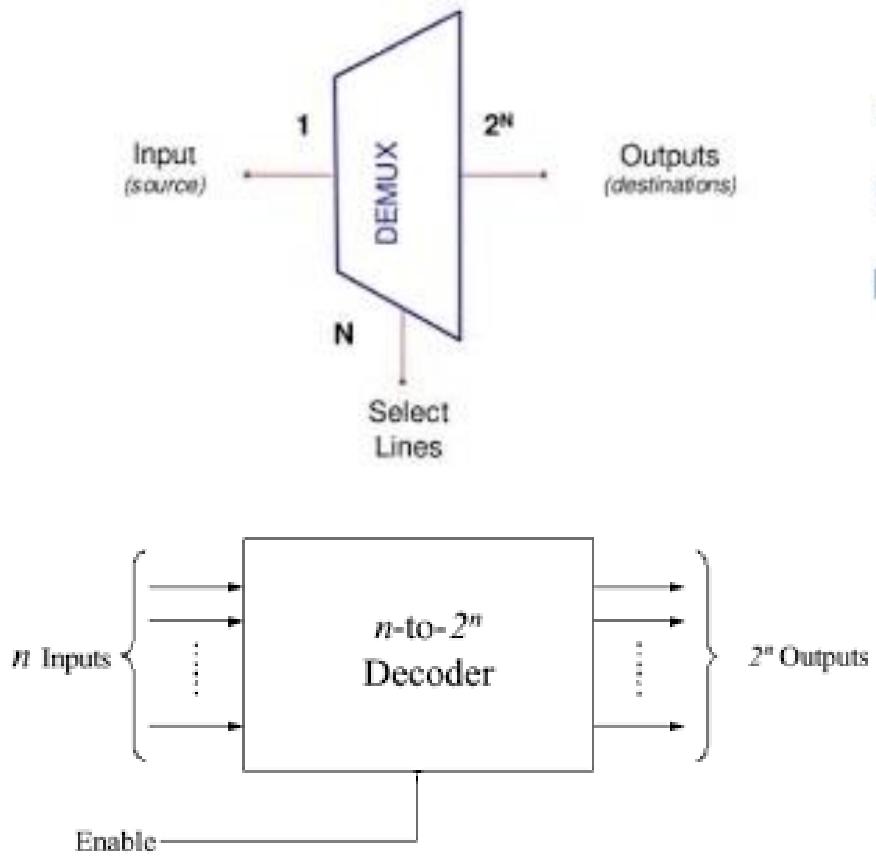
(b) 4-to-1-line mux

# دی‌مالتی‌پلکسor (Demultiplexer)

یک مدار ترکیبی که برعکس تسویه‌یم‌کننده، یک خط ورودی را به یکی از چند خط خروجی هدایت می‌کند



# دی مالتی پلکس



# انواع حافظه

- Content Addressable Memory (CAM)
- Sequential Access Memory (SAM)
- Random Access Memory (RAM)

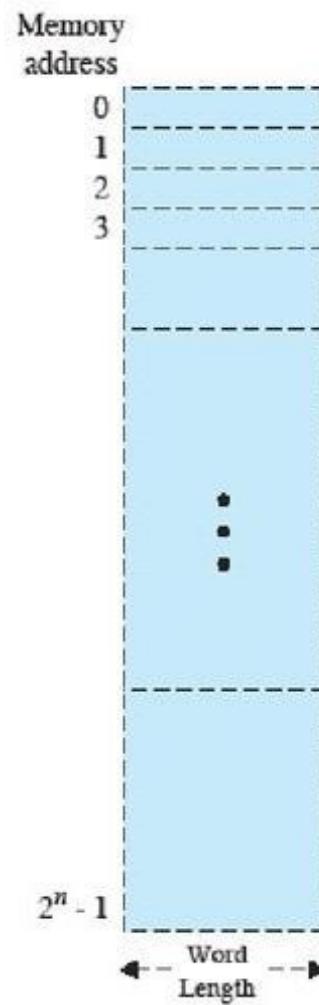
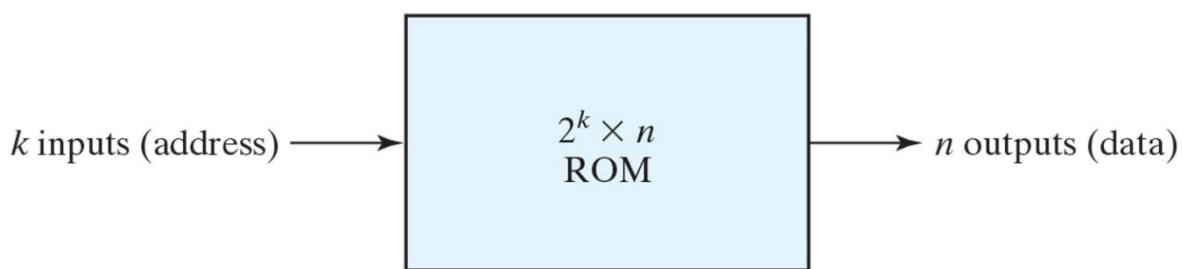


# انواع حافظه با دسترسی تمادفی (RAM)

Memory Type	Category	Erasure	Write Mechanism	Volatility
Random-access memory (RAM)	Read-Write Memory (RWM)	Electrically, byte-level	Electrically	Volatile
Read-only memory (ROM)	Read-Only Memory (ROM)	Not possible	Masks	
Programmable ROM (PROM)				
Erasable PROM (EPROM)		UV light, chip-level		Nonvolatile
Electrically Erasable PROM (EEPROM)	Read-Mostly Memory (RMM)	Electrically, byte-level	Electrically	
Flash memory		Electrically, block-level		



# بلوک دیاگرام یک حافظه فقط خواندنی

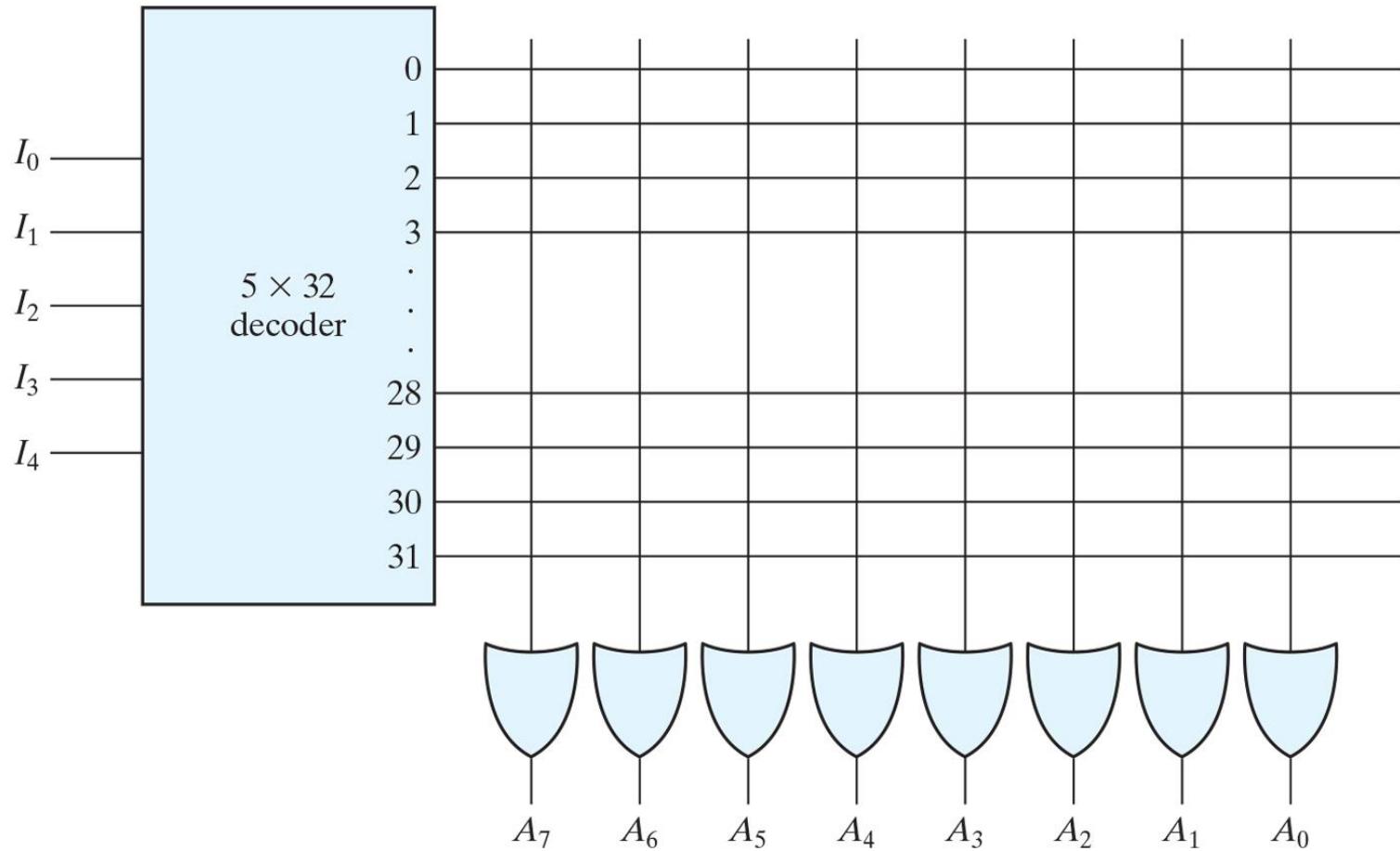


# مدادهای یک حافظه فرضی $16 \times 16$

Memory address		Memory content
Binary	Decimal	
0000000000	0	1011010101011101
0000000001	1	1010101110001001
0000000010	2	0000110101000110
⋮	⋮	⋮
1111111101	1021	1001110100010100
1111111110	1022	0000110100011110
1111111111	1023	1101111000100101

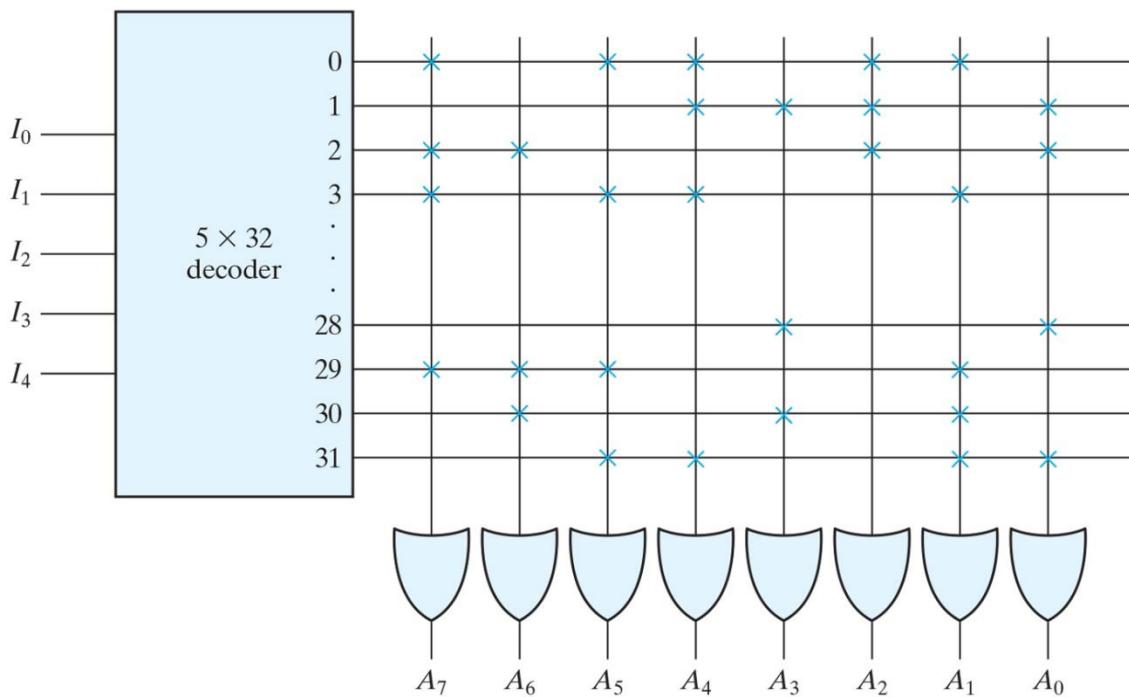


# ROM مدارهای داخلی



# مثال ۱

Inputs					Outputs							
$I_4$	$I_3$	$I_2$	$I_1$	$I_0$	$A_7$	$A_6$	$A_5$	$A_4$	$A_3$	$A_2$	$A_1$	$A_0$
0	0	0	0	0	1	0	1	1	0	1	1	0
0	0	0	0	1	0	0	0	1	1	1	0	1
0	0	0	1	0	1	1	0	0	0	1	0	1
0	0	0	1	1	1	0	1	1	0	0	1	0
⋮					⋮							
1	1	1	0	0	0	0	0	0	1	0	0	1
1	1	1	0	1	1	1	1	0	0	0	1	0
1	1	1	1	0	0	1	0	0	1	0	1	0
1	1	1	1	1	0	0	1	1	0	0	1	1

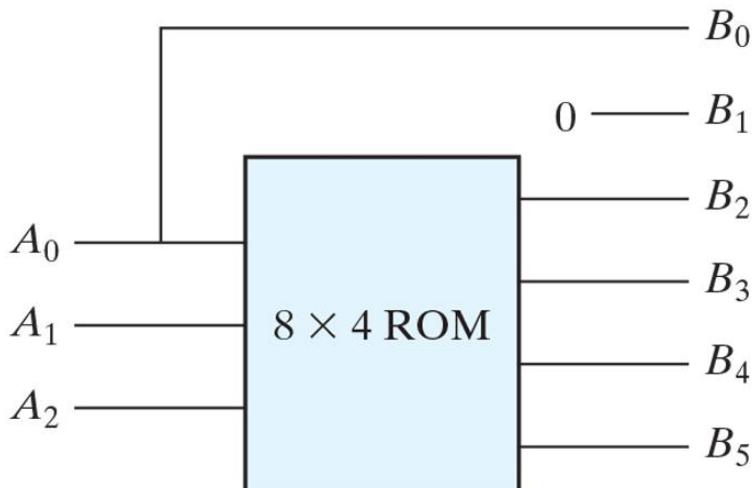


## مثال ۲: تولید مجدد (۹۵) دی

Inputs			Outputs						Decimal
$A_2$	$A_1$	$A_0$	$B_5$	$B_4$	$B_3$	$B_2$	$B_1$	$B_0$	
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1	1
0	1	0	0	0	0	1	0	0	4
0	1	1	0	0	1	0	0	1	9
1	0	0	0	1	0	0	0	0	16
1	0	1	0	1	1	0	0	1	25
1	1	0	1	0	0	1	0	0	36
1	1	1	1	1	0	0	0	1	49



# مثال ۲ (ادامه)



$A_2$	$A_1$	$A_0$	$B_5$	$B_4$	$B_3$	$B_2$
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	1
0	1	1	0	0	1	0
1	0	0	0	1	0	0
1	0	1	0	1	1	0
1	1	0	1	0	0	1
1	1	1	1	1	0	0

(b) ROM truth table

# Combinational PLDs



(a) Programmable read-only memory (PROM)



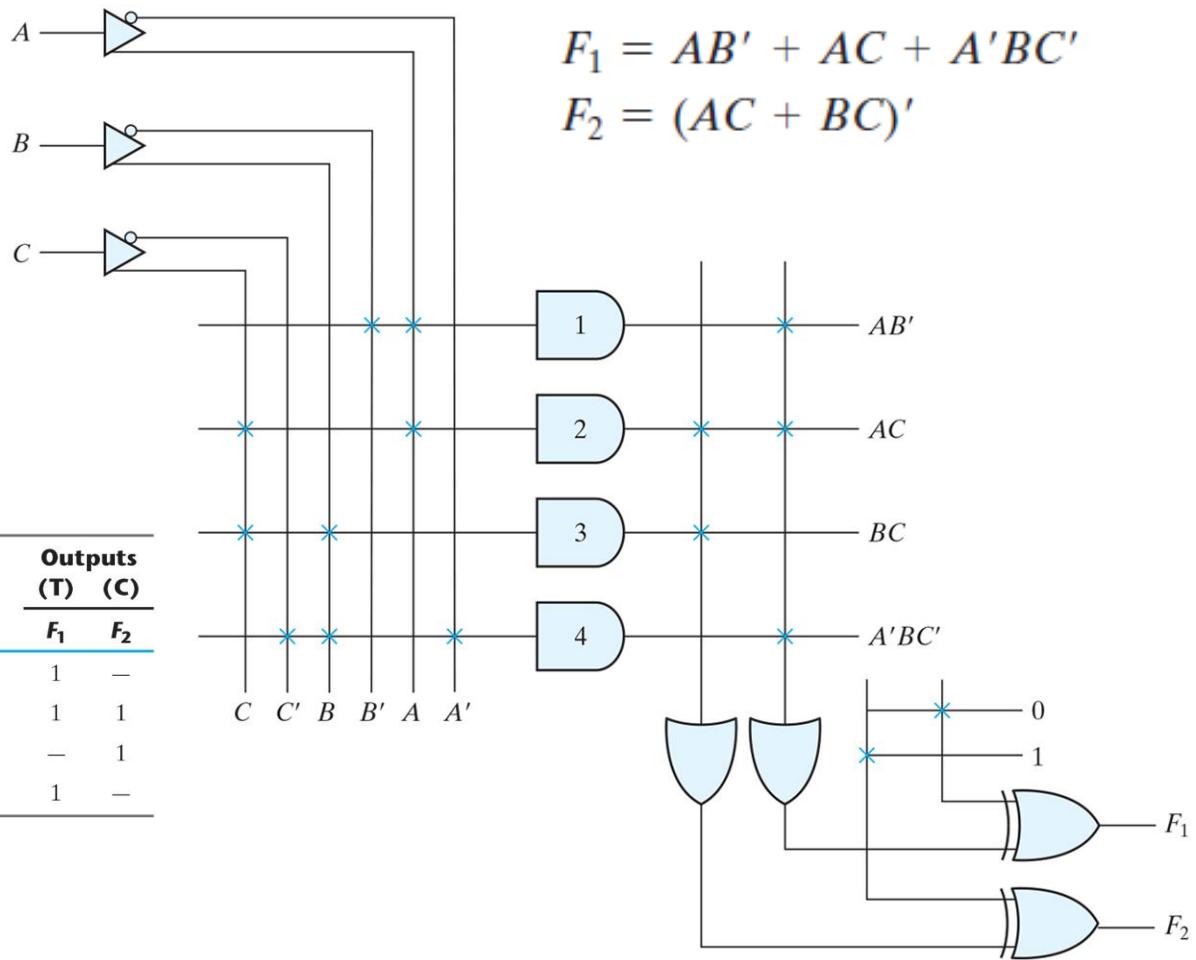
(b) Programmable array logic (PAL)



(c) Programmable logic array (PLA)



# مثال ا: طراحی با PLA



# مثال ۲: طراحی با PLA

$$F_1(A, B, C) = \Sigma(0, 1, 2, 4)$$

$$F_2(A, B, C) = \Sigma(0, 5, 6, 7)$$



		BC		00		01		11		B		10	
		A	0	$m_0$	1	$m_1$	1	$m_3$	0	$m_2$	1		
		A	1	$m_4$	1	$m_5$	0	$m_7$	0	$m_6$	0		

		BC		00		01		11		B		10	
		A	0	$m_0$	1	$m_1$	0	$m_3$	0	$m_2$	0		
		A	1	$m_4$	0	$m_5$	1	$m_7$	1	$m_6$	1		

# مثال ۲: طراحی با PLA

$$F_1(A, B, C) = \Sigma(0, 1, 2, 4)$$

$$F_2(A, B, C) = \Sigma(0, 5, 6, 7)$$

$$F_1 = (AB + AC + BC)'$$

$$F_2 = AB + AC + A'B'C'$$

PLA programming table						
Product term	Inputs			Outputs		
	A	B	C	(C)	(T)	
				$F_1$	$F_2$	
$AB$	1	1	—	1	1	
$AC$	2	1	—	1	1	
$BC$	3	—	1	1	—	
$A'B'C'$	4	0	0	0	—	1

Truth table for  $F_1$ :

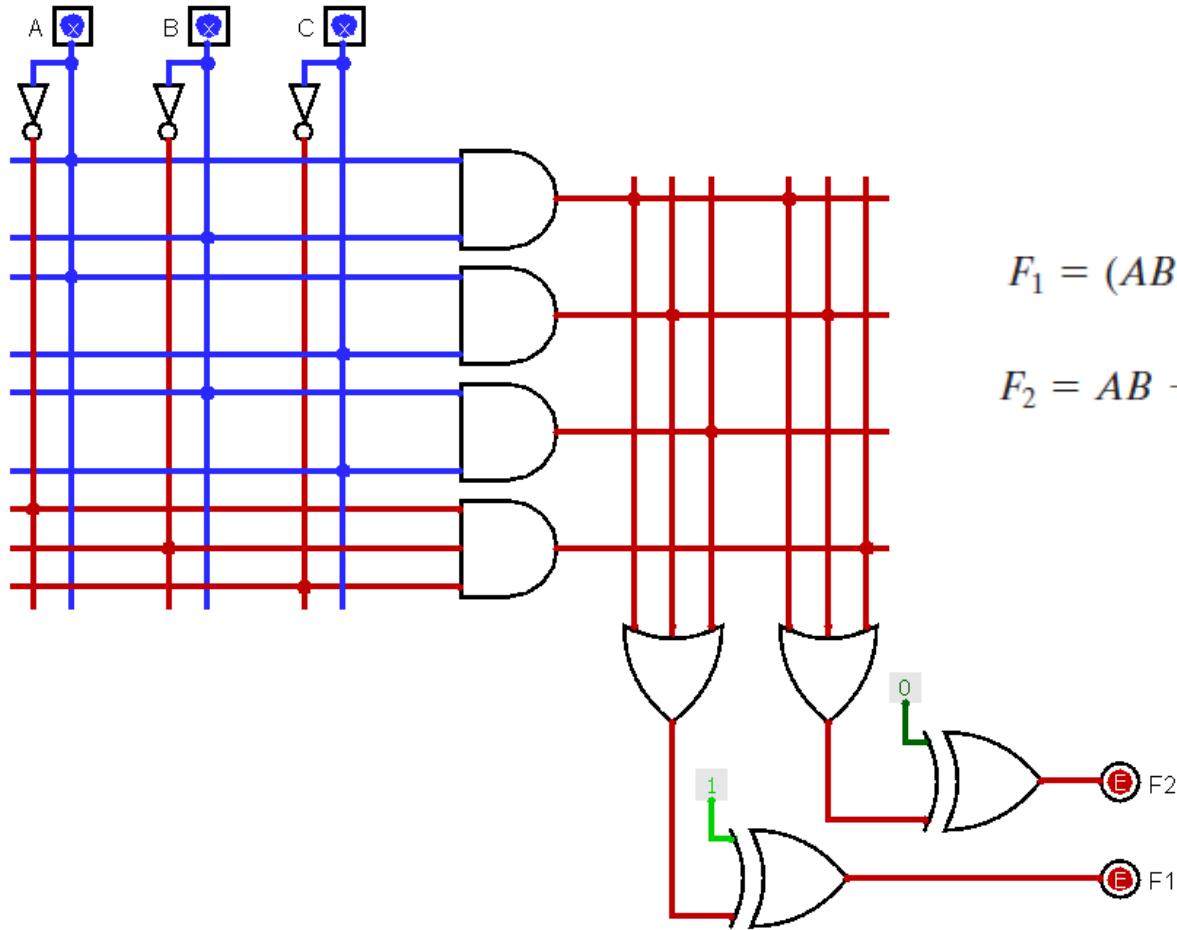
A	BC		B		C
	00	01	11	10	
0	$m_0$	$m_1$	$m_3$	$m_2$	1
	1	1	0	1	
1	$m_4$	$m_5$	$m_7$	$m_6$	0
	1	0	0	0	

Truth table for  $F_2$ :

A	BC		B		C
	00	01	11	10	
0	$m_0$	$m_1$	$m_3$	$m_2$	1
	1	0	0	0	
1	$m_4$	$m_5$	$m_7$	$m_6$	0
	0	1	1	1	



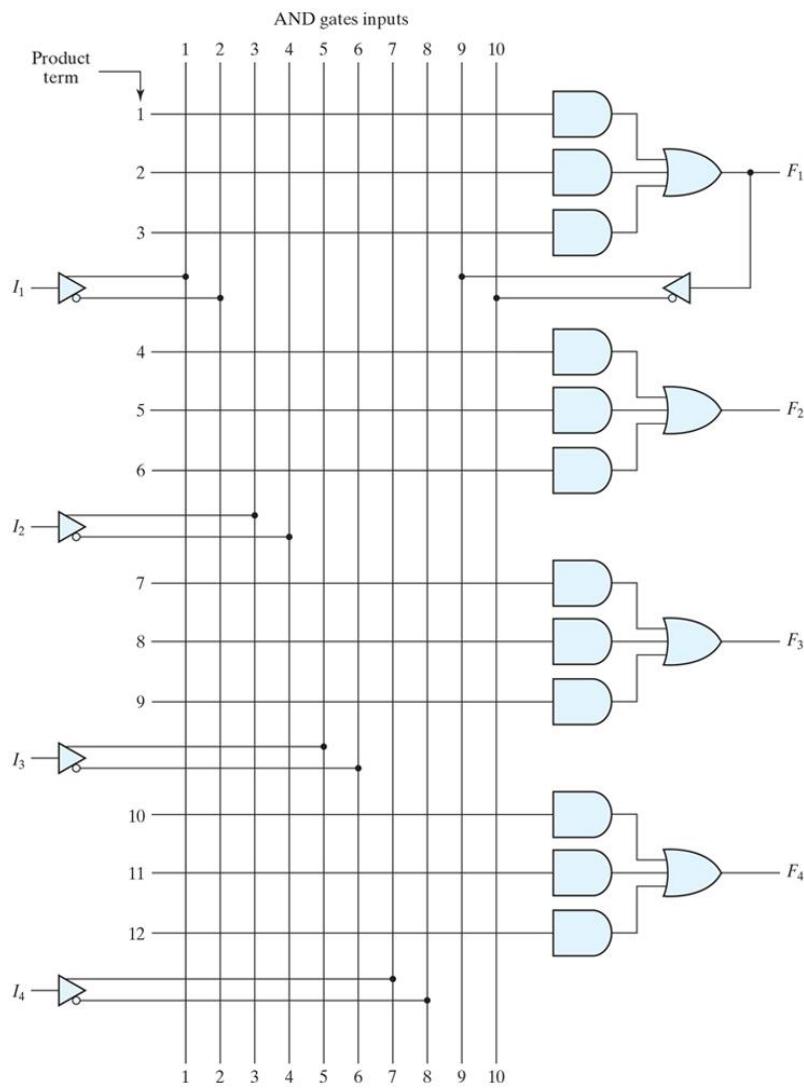
# مثال ۲: طراحی با PLA



$$F_1 = (AB + AC + BC)'$$

$$F_2 = AB + AC + A'B'C'$$

# مثال ۳: طراحی با PAL



$$w(A, B, C, D) = \Sigma(2, 12, 13)$$

$$x(A, B, C, D) = \Sigma(7, 8, 9, 10, 11, 12, 13, 14, 15)$$

$$y(A, B, C, D) = \Sigma(0, 2, 3, 4, 5, 6, 7, 8, 10, 11, 15)$$

$$z(A, B, C, D) = \Sigma(1, 2, 8, 12, 13)$$

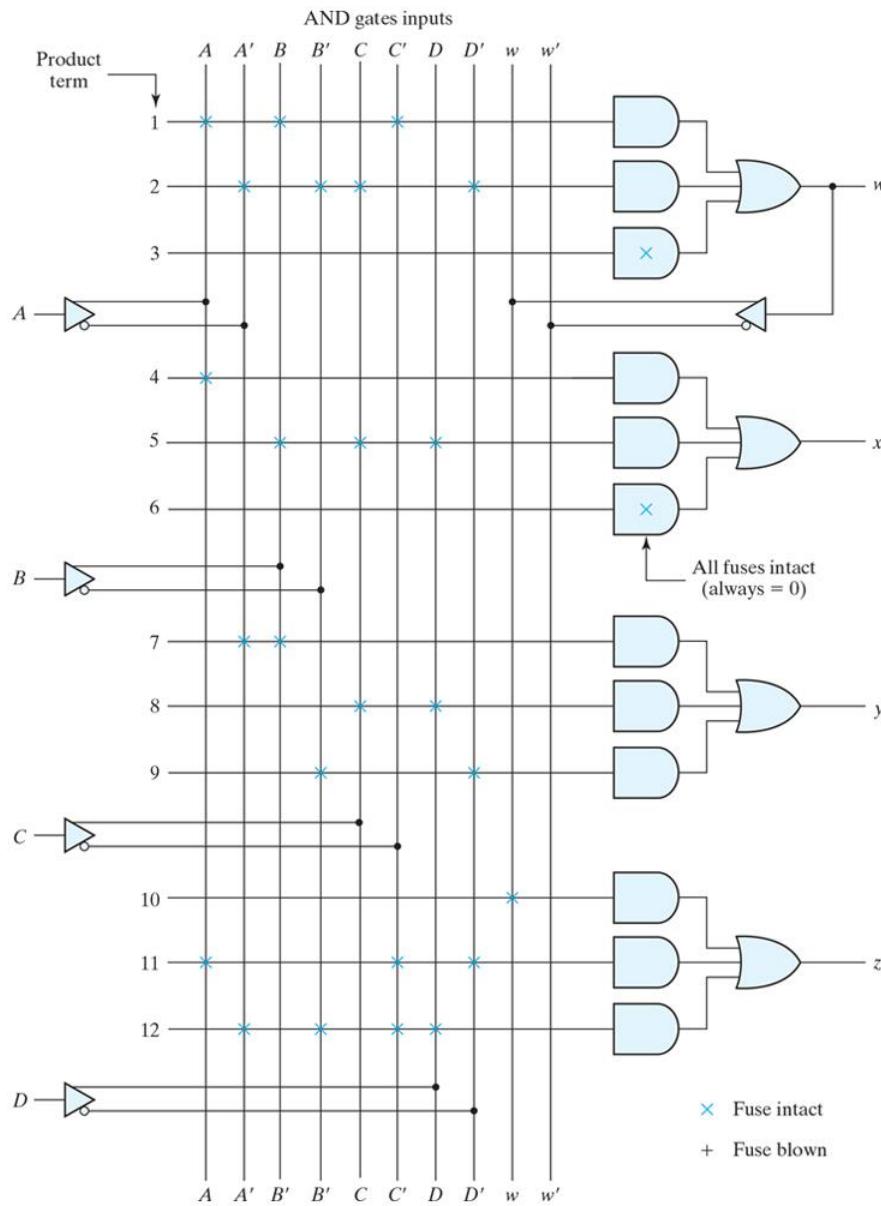


# مثال ۳: طراحی با PAL (ادامه)

<b>Product Term</b>	<b>AND Inputs</b>					<b>Outputs</b>
	<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>w</b>	
1	1	1	0	—	—	$w = ABC' + A'B'CD'$
2	0	0	1	0	—	
3	—	—	—	—	—	
4	1	—	—	—	—	$x = A + BCD$
5	—	1	1	1	—	
6	—	—	—	—	—	
7	0	1	—	—	—	$y = A'B + CD + B'D'$
8	—	—	1	1	—	
9	—	0	—	0	—	
10	—	—	—	—	1	$z = w + AC'D' + A'B'C'D$
11	1	—	0	0	—	
12	0	0	0	1	—	



# مثال م: (ادامه)



$$w = ABC' + A'B'CD'$$

$$x = A + BCD$$

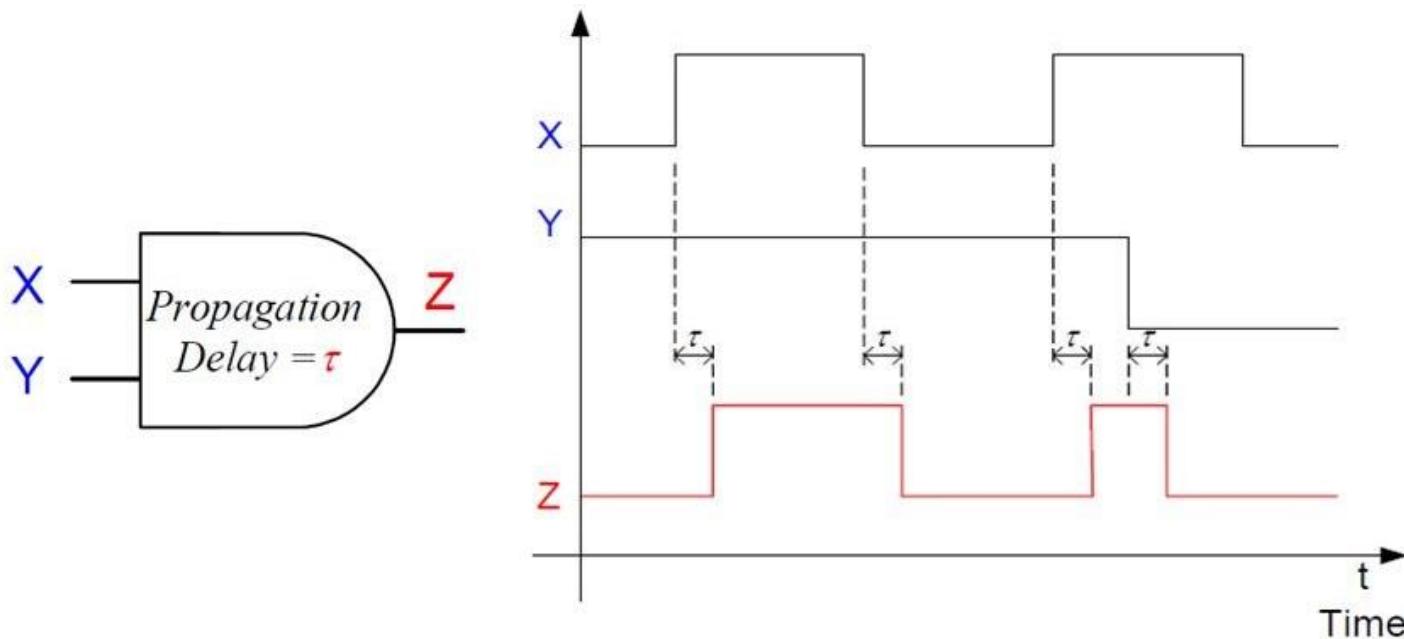
$$y = A'B + CD + B'D'$$

$$\begin{aligned} z &= ABC' + A'B'CD' + AC'D' + A'B'C'D \\ &= w + AC'D' + A'B'C'D \end{aligned}$$



# تاخیر در انتشار (یادآوری)

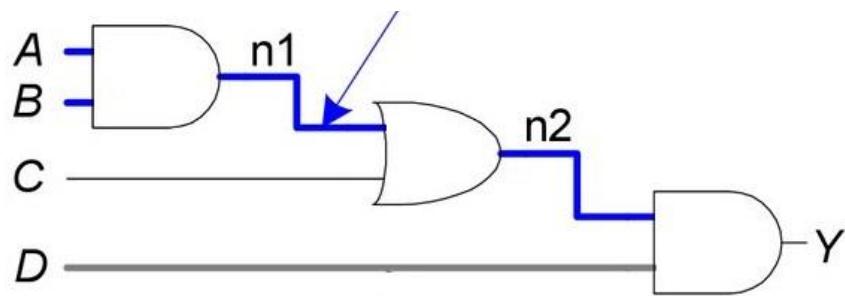
زمان بین تغییر در ورودی تا تغییر در خروجی ناشی از آن



# مسیر بحرانی

- مسیری که بیشترین تأخیر را دارد
- مسیر بحرانی را گاهی بر حسب جمع تأخیرهای هر گیت و اغلب بر حسب تعداد گیتهای مسیر بیان می‌کنیم

مسیر بحرانی (critical path)

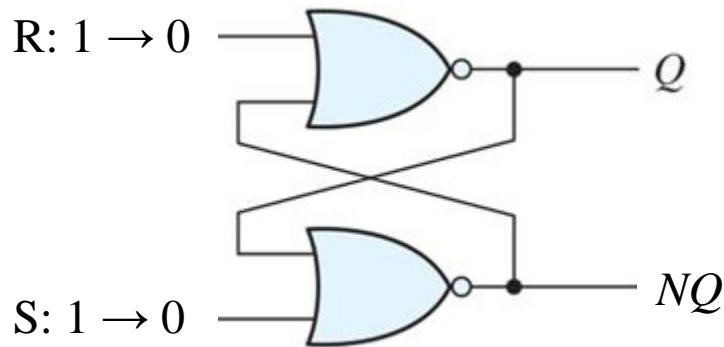


# مسابقه (Race)

- مسابقه در یک مدار منطقی یعنی اقدام به تغییر همزمان دو یا چند اتصال از آن مدار
- اگر چه در مسابقه تغییرات ظاهرا همزمان اعمال می‌شوند، لاما به دلیل وجود تغییر لین تغییرات واقعا همزمان تغییر نمی‌کنند و آنکه زودتر از همه تغییر می‌کند، **برنده** مسابقه نامیده می‌شود
- تغییرها برای طراحان مدارهای منطقی مشخص نیست، بنابراین برنده مسابقه در زمان طراحی **نامعلوم** است
- اگر خروجی نهایی مدار تدت تاثیر برنده مسابقه قرار داشته باشد، مسابقه، **بهرانی** محسوب می‌شود



# مثال: مسابقه بمرانی



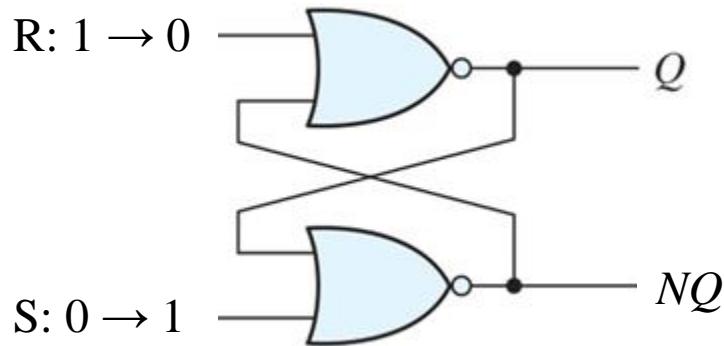
S wins: RS=11 → RS=10 → RS=00

$$\begin{array}{ccc} Q=0 & Q=0 & \textcolor{red}{Q=0} \\ \textcolor{red}{NQ=0} & \textcolor{red}{NQ=1} & \textcolor{red}{NQ=1} \end{array}$$

R wins: RS=11 → RS=01 → RS=00

$$\begin{array}{ccc} \textcolor{red}{Q=0} & \textcolor{red}{Q=1} & \textcolor{red}{Q=1} \\ NQ=0 & \textcolor{red}{NQ=0} & \textcolor{red}{NQ=0} \end{array}$$

# مثال: مسابقه غیربهمارانی



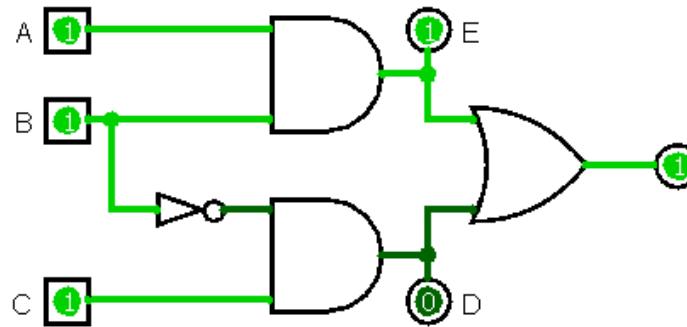
S wins: RS=10 → RS=11 → RS=01

Q=0	Q=0	Q=1
NQ=1	NQ=0	NQ=0

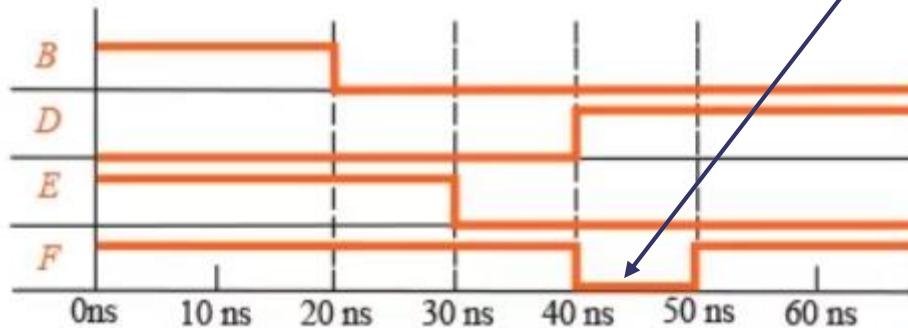
R wins: RS=10 → RS=00 → RS=01

Q=0	Q=0	Q=1
NQ=1	NQ=1	NQ=0

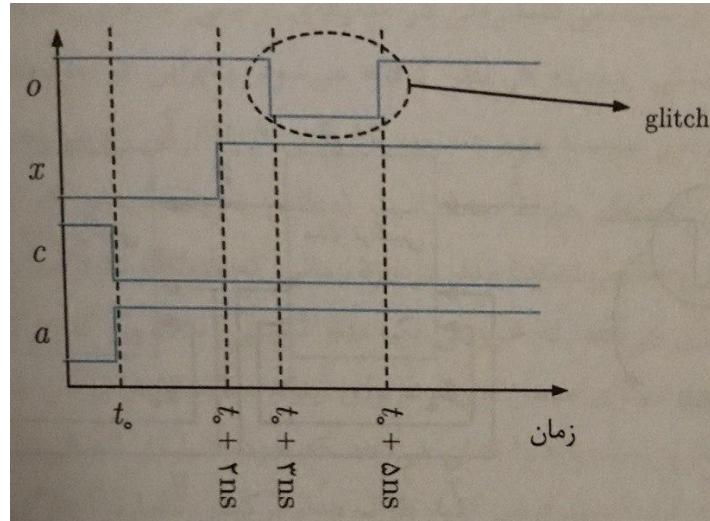
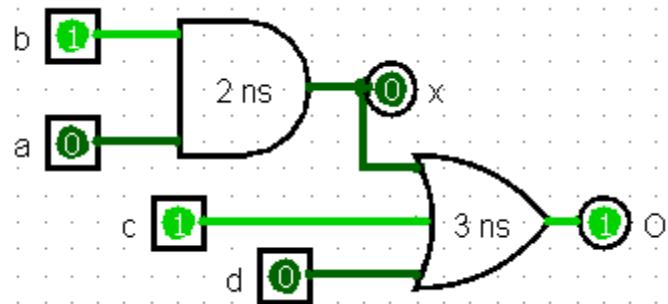
# مفارده (Hazard) – مثال ۱



glitch



# مفتله (Hazard) – مثال ۲



# چند نکته درباره glitch

- به مقادیری از خروجی که مطابق منطق مدار ترکیبی نباید ایجاد شود ولی به دلیل تغییر گیت‌ها ظاهر می‌شود، glitch گفته می‌شود
- **glitch**ها ناشی از بروز مسابقه در مدار ترکیبی هستند
- رخدادن **glitch** بستگی به سیگنال برنده مسابقه دارد
- پنهانی **glitch** بستگی به اختلاف میان تغییر سیگنال‌های رقیب دارد
- پس از تغییر ورودی یک مدار ترکیبی، glitch تا وقتی می‌تواند رخداد و باقی بماند که تغییر مسیر بصرانی سپری نشده باشد. پس از این زمان glitch از بین رفته و خروجی ثابت می‌ماند



# مغایطه (Hazard)

- یک مدار ترکیبی دارای مغایطه است اگر احتمال **glitch** بروز در **خوبی** آن وجود داشته باشد
- مغایطه پنهان (potential hazard)
- فقط با تغییر یک بیت ورودی احتمال **glitch** در خوبی وجود دارد
- مغایطه کارکردی (functional hazard)
  - با تغییر بیش از یک بیت ورودی احتمال **glitch** در خوبی وجود دارد



# انواع مفاطره پنهان

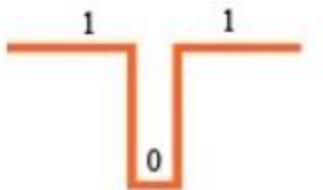
## ○ مفاطره لیستای یک (static 1-hazard)

- فروجی باید یک باشد اما یک glitch با مقدار صفر رخ می‌دهد

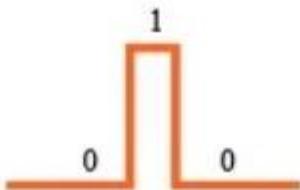
## ○ مفاطره لیستای صفر (static 0-hazard)

- فروجی باید صفر باشد اما یک glitch با مقدار یک رخ می‌دهد

## ○ مفاطره پویا (dynamic hazard)



(a) Static 1- hazard

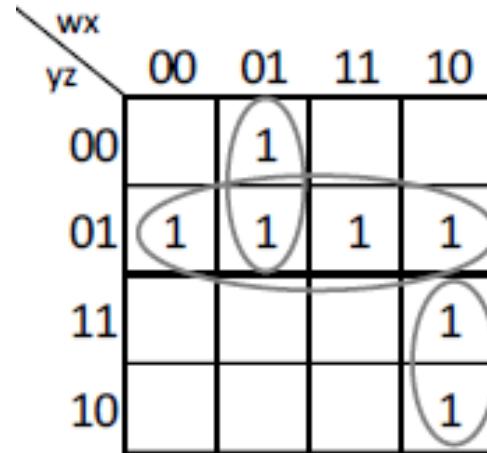
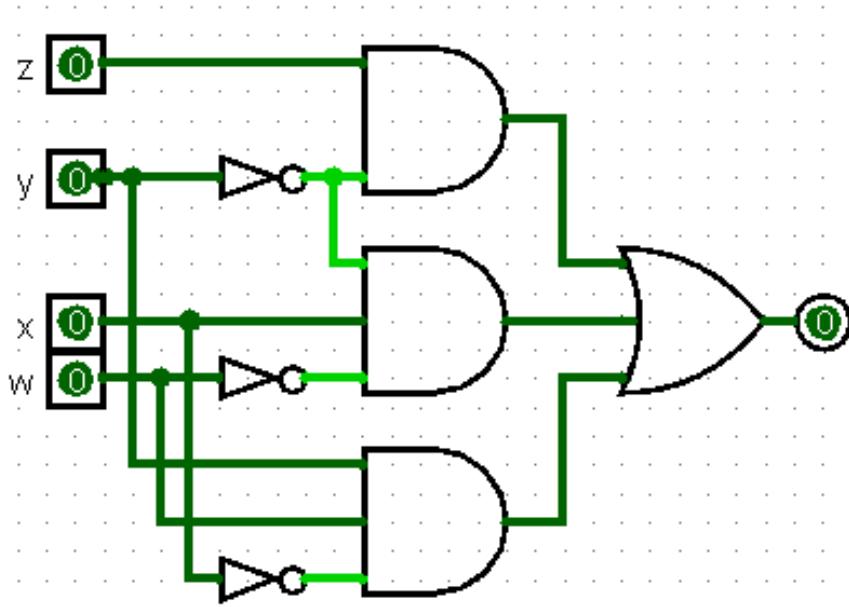


(b) Static 0- hazard



(c) Dynamic hazards

# مدارهای دو طبقه SOP (یادآوری)



# ظواهر در مدارهای دو طبقه ۱-SOP

- در ورودی گیت‌های AND هرگز مسابقه رخ نمی‌دهد
  - با این شرط که هیچ دو ورودی هم‌مان تغییر نکنند
- اما ممکن است در ورودی گیت‌های OR مسابقه رخ دهد
  - باید سایر ورودی‌هایی که در مسابقه شرکت ندارند، صفر باشند
  - سیگنال‌هایی که در مسابقه شرکت دارند، در خلاف هم تغییر وضعیت بدهند
  - اگر سیگنال پایین‌روندۀ برنده شود، glitch رخ خواهد داد
  - بنابراین یک ظواهر از نوع لیستای یک پیش خواهد آمد



## ۲- SOP طبقه دو مدارهای در

- وقتی در یک خانه جدول کارنو هستیم، یعنی ورودی اعمال شده به مدار را مشخص می‌کنیم
- وقتی از یک خانه جدول کارنو به خانه دیگر می‌رویم، یعنی مقدار ورودی‌های مدار را تغییر داده‌ایم
- هر cube در جدول کارنو معادل یک گیت AND است
  - وقتی درون یک cube هستیم، یعنی خروجی گیت AND، یک است
  - وقتی بیرون یک cube باشیم، یعنی خروجی گیت AND متناظر آن صفر است

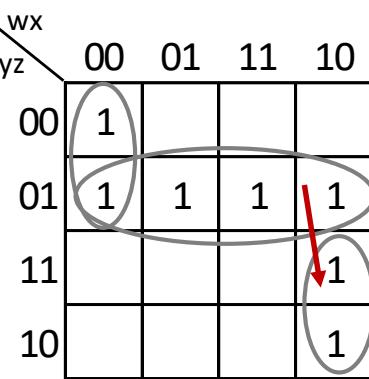


## مظاهره در مدارهای دو طبقه SOP - M

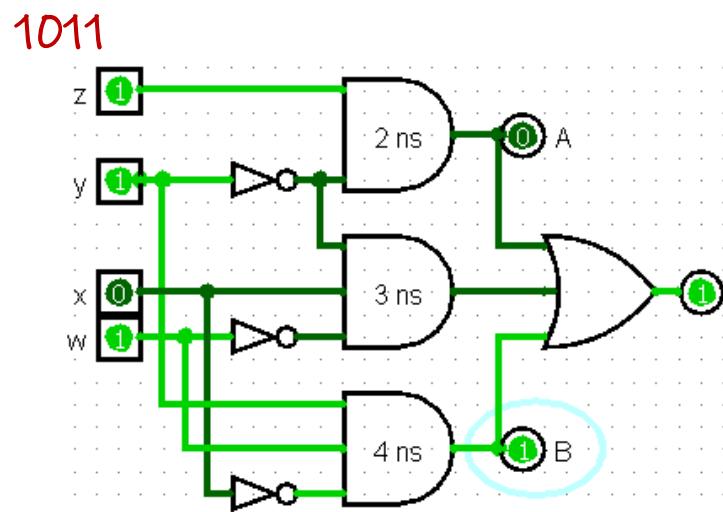
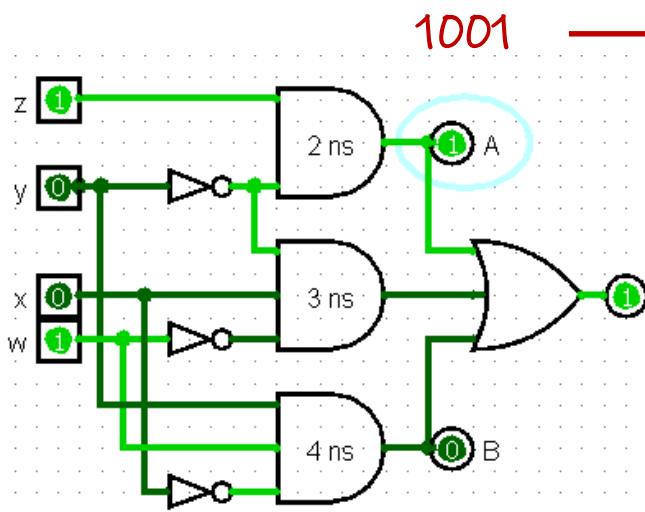
- در جدول کارنو از هر خانه فقط می‌توانیم به خانه‌های مجاور برویم
- وقتی از یک خانه جدول کارنو به خانه مجاور می‌رویم، اگر داخل یک cube باشیم:
  - یا داخل همان cube باقی می‌مانیم، بدون اینکه وارد cube دیگری شویم
  - یا در حالی که داخل همان cube هستم وارد cube دیگری می‌شویم
  - یا از یک cube خارج شده و وارد یک cube دیگر می‌شویم
- تنها در حالت سوم است که ممکن است مظاهره رخ دهد
- مظاهره زمانی رخ می‌دهد که دو cube **مجاور** باشند لاما **همپوشانی** نداشته باشند



# مثال



$$F = w'x'y' + y'z + wx'y$$

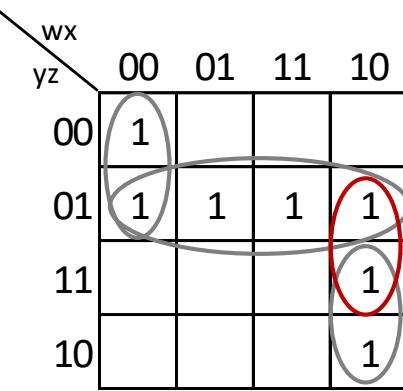


# رفع مفاطرۀ در مدارهای دو طبقه SOP

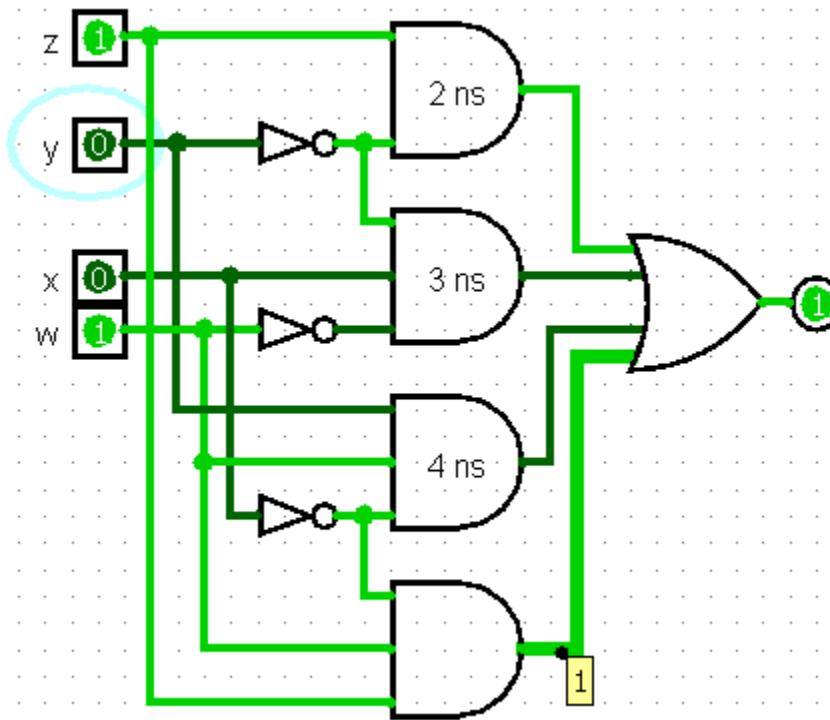
- اضافه کردن یک cube جدید با هدف پوشش دارن cube های مجاور
- در مدارهایی که don't care دارند، ممکن است راه ساده‌تری هم برای رفع مفاطرۀ باشد
  - گاهی don't care ها ناشی از ورودی ممنوعه است و در تیجه برخی گذارها از یک cube به cube مجاور اصلاً پیش نمی‌آید
  - گاهی don't care ها ناشی از حالت‌هایی از ورودی هستند که ممکن است پیش بیاید، در لین صورت شاید بتوان با حذف cube ها از دلفل don't care را کاملاً از هم دور کرد



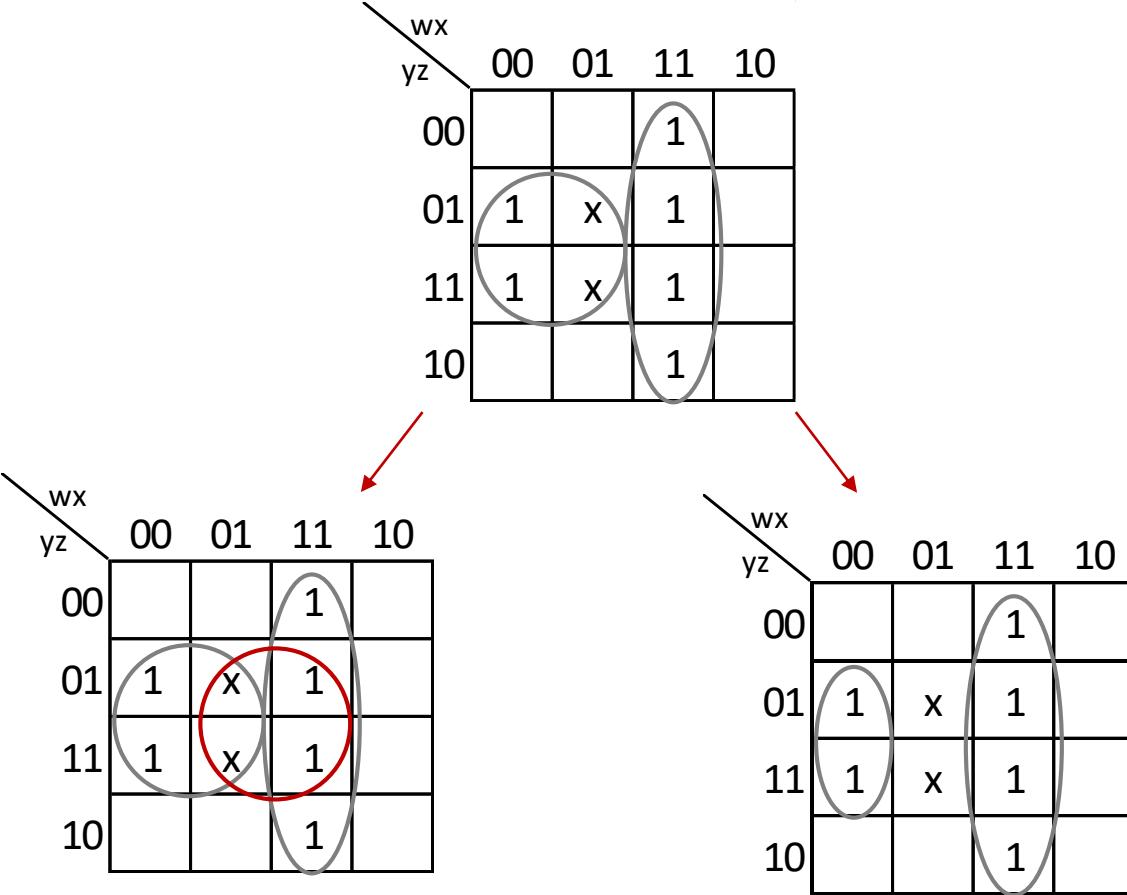
# مثال ۱



$$F = w'x'y' + y'z + wx'y + wx'z$$



# مثال ۲



# جمع‌بندی

## ○ مدارهای ترکیبی:

مجموعه‌ای از گیت‌های منطقی به هم متصل که فرآیند آنها در هر لحظه تابعی از ورودی‌های آنها در همان لحظه است و ارتباطی به ورودی‌های قبلی ندارد

## ○ برنی مدارهای ترکیبی پایه‌ای که برای ساخت مدارهای پیچیده‌تر به کار می‌روند:

کدگشا / کدگذار

تسهیم‌کننده

انواع جمع‌کننده‌ها

حافظه‌های برنامه‌پذیر

## ○ مناظره در مدارهای ترکیبی

