

درس مدارهاي منطقي ديگيتال

مرجع: مدارهاي منطقي ديگيتال

نوشته: مانو-- مترجم: دكتر سپيدنام

تهيه كننده: مجتبي پورمحقق

(عضو هيئت علمي مركز فريمان)

فصل اول:

ورود به سیستم دیجیتال

سیستم ده دهی اعداد (Decimal):

❑ آشنایی پیچیدگی را پنهان می کند؟

❑ ده رقم 0..9

❑ موقعیت ، وزن تعیین می کند:

... 10^4 10^3 10^2 10^1 10^0

1 7 3


$$= 1 \times 10^2 + 7 \times 10^1 + 3 \times 10^0$$

$$= 100 + 70 + 3$$

$$= 173$$

سیستم دودویی اعداد (binary):

- آسان برای کامپیوترها، ناملموس برای ما
- از ارقام دودویی (binary digits (bits))، به جای ارقام ده دهی استفاده می کند.
- n بیت داده شده می تواند نشانگر 2^n عدد باشد.
- با ده انگشت می شود تا ۱۰۲۳ شمرد!
- در این سیستم نیز از موقعیت، وزن را تعیین می کند.

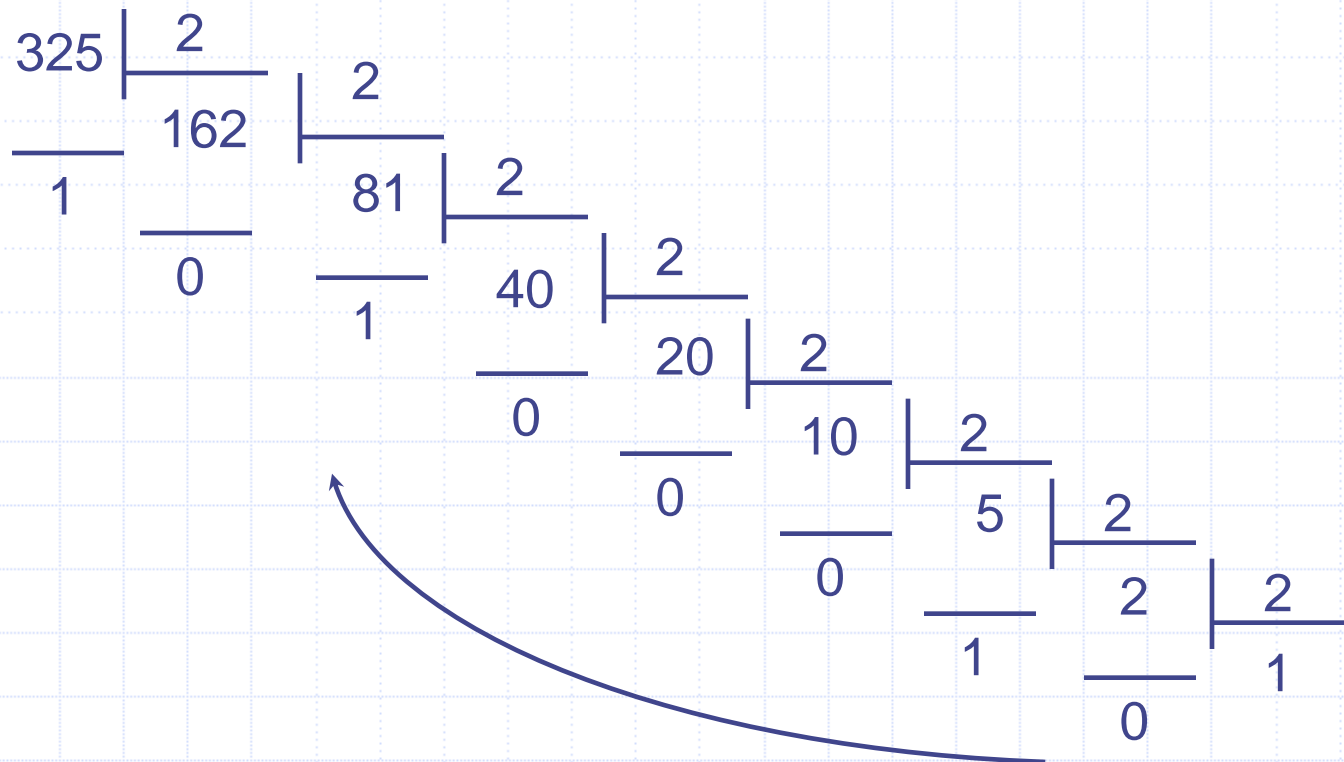


Dec	2^3	2^2	2^1	2^0	Binary
0				0	0
1				1	1
2			1	0	10
3			1	1	11
4		1	0	0	100
5		1	0	1	101
6		1	1	0	110
7		1	1	1	111
8	1	0	0	0	1000

تبدیل از مبنای ده به مبنای دو

روش اول : تقسیمات متوالی

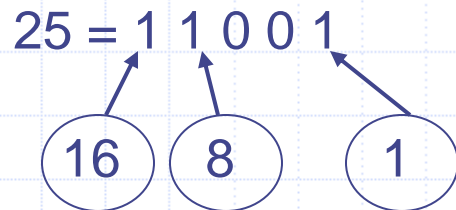
$$(325)_{10} \rightarrow (101000101)_2$$




روش دوم : کاهش متوالی توان های دو

توان های دو :

$1 \rightarrow 2 \rightarrow 4 \rightarrow 8 \rightarrow 16 \rightarrow 32 \rightarrow 64 \rightarrow 128 \rightarrow 256 \rightarrow 512 \rightarrow 1024 \rightarrow \dots$



تبدیل از مبنای دو به مبنای ده


$$\begin{array}{cccccc} (1 & 0 & 1 & 1 & 1 & 0)_{21} & = & 0 \times 1 + 1 \times 2 + 1 \times 4 + 1 \times 8 + 0 \times 16 + 1 \times 32 = & (46)_{10} \\ \swarrow & \swarrow & \downarrow & \downarrow & \searrow & \searrow & & & \\ 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 & & & \end{array}$$

اعداد اعشاری

$$25.43 \rightarrow 11001.01101 \dots$$

$$0.43 * 2 = 0.86$$

$$0.86 * 2 = 1.72$$

$$0.72 * 2 = 1.44$$

$$0.44 * 2 = 0.88$$

$$0.88 * 2 = 1.76$$

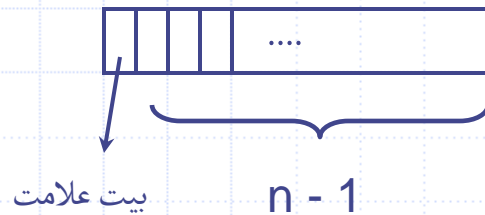
...

حداقل 0
حداکثر $2^n - 1$

} اعداد بدون علامت در قالب n بیتی:

$$2^0 + 2^1 + \dots + 2^a = 2^{(a+1)} - 1$$

اعداد علامت دار



۱- سیستم علامت مقدار
 $0 : +$
 $1 : -$

۲- سیستم متمم دو

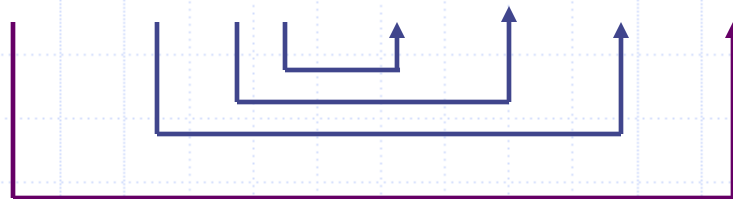
$$258 - 194 = 258 + (999 - 194) + 1 - 1000 =$$

$$A - B = A + \overline{B} + 1$$

متمم دو

در روش متمم دو :

$$1\ 0\ 0\ 1\ 0\ 1\ 1 = +2^0 + 2^1 + 2^3 - 2^6 = -53$$



تمرین : یک عدد منفی پیدا کنید، که روش نمایش آن در سیستم متمم دو و قالب n بیتی عینا مشابه نمایش آن در سیستم علامت مقدار و قالب n بیتی باشد.

تمرین : سیستمی برلی ارائه اعداد اعشاری منفی نشان دهید که به کمک آن بتوان جمع و تفریق را انجام داد و درگیر رقم قرض نشد.

روش های ممکن جهت نمایش اعداد علامت دار:

سیستم علامت مقدار

$$000 = +0$$

$$001 = +1$$

$$010 = +2$$

$$011 = +3$$

$$100 = -4$$

$$101 = -3$$

$$110 = -2$$

$$111 = -1$$

سیستم متمم یک

$$000 = +0$$

$$001 = +1$$

$$010 = +2$$

$$011 = +3$$

$$100 = -3$$

$$101 = -2$$

$$110 = -1$$

$$111 = -0$$

سیستم متمم دو

$$000 = +0$$

$$001 = +1$$

$$010 = +2$$

$$011 = +3$$

$$100 = -0$$

$$101 = -1$$

$$110 = -2$$

$$111 = -3$$

متمم ۲ :

$$(49)_{10} = (110001)_2$$

۱- عدد بدون علامت به صورت باینری نوشته شود.

0110001

۲- قالب ریزی

۳- اگر عدد مثبت بود، کار تمام است، اما اگر عدد منفی است لازم است متمم دو شود.

جمع و تفریق اعداد علامت دار :

$$\begin{array}{r} - 49 \\ + 23 \\ \hline - 26 \end{array}$$
$$\begin{array}{r} 1001111 \\ 0010111 \\ \hline 1100110 \end{array}$$

- اگر در جمع خطای سرریز رخ داد ، باید جمع را در قالب بزرگتری انجام دهیم.

- در سیستم بدون علامت خطای سرریز همان **Carry** است.

خطای سرریز (Over flow)

- در جمع اعداد بدون علامت ، رخداد سرریز همان رقم نقلی است.
- در جمع و تفریق اعداد علامت دار ، سرریز در دو هنگام ممکن است رخ دهد: جمع دو عدد مثبت یا جمع دو عدد منفی.

تشخیص رخداد سرریز:

راه اول : اگر حاصلجمع دو عدد مثبت عددی منفی شود و یا جمع دو عدد منفی ، عددی مثبت ،

راه دوم : در صورتی که دو رقم نقلی آخر مساوی باشند.

جمع اعداد اعشاری :

$$\begin{array}{r}
 25.50 \\
 - 38.75 \\
 \hline
 \end{array}
 \longrightarrow
 \begin{array}{r}
 0011001.1000 \\
 1011001.0100 \\
 \hline
 1110010.1100 \\
 \underbrace{\hspace{1.5cm}}_{-13} \quad \swarrow \quad \searrow \\
 \hspace{1.5cm} 0.5 \quad 0.25
 \end{array}$$

$$25 \rightarrow (11001)_2$$

مبنای ۴، ۸، ۱۶

$$\begin{array}{c}
 011001 \\
 \longleftarrow \quad \underbrace{\hspace{1cm}} \\
 \end{array}
 \rightarrow (121)_4$$

$$\begin{array}{c}
 011001 \\
 \longleftarrow \quad \underbrace{\hspace{1cm}} \\
 \end{array}
 \rightarrow (31)_8$$

$$\begin{array}{c}
 00011001 \\
 \longleftarrow \quad \underbrace{\hspace{1.5cm}} \\
 \end{array}
 \rightarrow (19)_{16}$$

ضرب و تقسیم اعداد باینری :

ضرب به روش معمولی :

$$\begin{array}{r} 1110 \\ * 0101 \\ \hline 1110 \\ 0000 \\ 1110 \\ 0000 \\ \hline 1000110 \end{array}$$

ضرب به روش جمع های متوالی :

$$\begin{array}{r} 1110 \\ 1110 \\ + 1110 \\ 1110 \\ 1110 \\ \hline 1000110 \end{array}$$

کدینگ اطلاعات :

هدف : ورود به سیستم دیجیتال

معیارها :
- افزایش سرعت
- کاهش فضا
- راحتی کار با آن
- امنیت
- اطمینان

Binary Coded Decimal

	B C D
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1

(دارای وزن)

- در مورد کاراکترها ، از کد اسکی آنها استفاده می کنیم.

ex - 3

	0 0 0 0
	0 0 0 1
	0 0 1 0
0	0 0 1 1
1	0 1 0 0
2	0 1 0 1
3	0 1 1 0
4	0 1 1 1
5	1 0 0 0
6	1 0 0 1
7	1 0 1 0
8	1 0 1 1
9	1 1 0 0
	1 1 0 1
	1 1 1 0
	1 1 1 1

	ex - 3
0	0 0 1 1
1	0 1 0 0
2	0 1 0 1
3	0 1 1 0
4	0 1 1 1
5	1 0 0 0
6	1 0 0 1
7	1 0 1 0
8	1 0 1 1
9	1 1 0 0

(خود مکمل)

$$\begin{array}{c}
 0 \\ \downarrow \\ 8
 \end{array}
 *
 \begin{array}{c}
 1 \\ \downarrow \\ 7
 \end{array}
 *
 \begin{array}{c}
 2 \\ \downarrow \\ 6
 \end{array}
 *
 \begin{array}{c}
 3 \\ \downarrow \\ 5
 \end{array}
 *
 \begin{array}{c}
 4 \\ \downarrow \\ 4
 \end{array}
 = 6720$$

تعداد کلیه سیستم های خود مکمل :

یک کد وزنی و خود مکمل :

	2	4	2	1
0	0	0	0	0
1	0	0	0	1
2	1	0	0	0
3	0	0	1	1
4	0	1	0	0
5	1	0	1	1
6	1	1	0	0
7	0	1	1	1
8	1	1	1	0
9	1	1	1	1

تمرین :

۱- چند کد وزنی و خود مکمل با ارزش های ۱، ۲، ۲، ۴ وجود دارد؟

۲- چند کد وزنی و خود مکمل با ارزش ۲۴۲۱ وجود دارد؟

۳ — ارزش های دیگری غیر از این ارزش بگویید.

۴ — ارزش منفی هم در اعداد قرار دهید.

۵- چه ویژگی ای باید این ارزش ها داشته باشند؟

۶ — روشی برای جمع و تفریق دودویی اعدادی که با سیستم BCD و ex-3 کد شدند، بیابید.

نمایش اعداد غیر صحیح (اعشاری):

0.257

.	0	1	0	1	1
---	---	---	---	---	---

$1 < \text{اعشاری}$

25

1	1	0	0	1	.
---	---	---	---	---	---

$1 \geq \text{صحیح}$

$$43.85 \rightarrow \underbrace{0.4385}_{\text{مانتیس}} * 10^{\text{نمای}} \textcircled{2}$$

$0 < \text{مانتیس} < 1$

--	--	--

علامت نمای

نمای

مانتیس

$$101011.1101 = 0.\underbrace{1010111101}_{\text{مانتیس}} * 2^{+6}$$

$0.5 < \text{مانتیس} < 1$

Parity

- توازن یا همپایگی

- در سیستم هایی که حداکثر احتمال بروز یک خطا وجود دارد.

خاصیت Parity طولی و عرضی:

- قابلیت تشخیص دو خطا را دارد ، ولی فقط یک خطا را می تواند تصحیح کند.

کد همینگ :

بیت های کنترلی

توان های ۲ ←

داده خام : 1 0 1 1

0	1	1	0	0	1	1
---	---	---	---	---	---	---

1 2 3 4 5 6 7

بیت های کنترلی

Parity زوج

$$P_1 = P (B_3, B_5, B_7) = 0$$

$$P_2 = P (B_3, B_6, B_7) = 1$$

$$P_4 = P (B_5, B_6, B_7) = 0$$

داده نهایی : 0 1 1 0 0 1 1

خطایابی :

داده ارسالی: 0 1 0 0 1 0 1

داده دریافتی: 0 1 0 0 1 1 1

P_1	P_2		P_4			
0	1	0	0	1	1	1
		B_3		B_5	B_6	B_7

P_1	=	0
P_2	=	0
P_4	=	1
<hr/>		
6	→	B_6 رخداد خطا

- یک بیت خطا قابل تصحیح

- دو بیت خطا قابل تشخیص

روش های جبری برای تحلیل

و

طراحی مدارهای منطقی

دستگاه های دیجیتالی

□ جبر بول:

➤ یک عبارت منطقی می تواند "درست" یا "نادرست" باشد (0 یا 1).

➤ شامل فرمول های جبری مربوط به ترکیب های مقادیر منطقی است.

◆ در سطح سخت افزار:

➤ هر عبارت منطقی با یک سیگنال الکتریکی نشان داده می شود.

➤ ارزش منطقی هر عبارت با ولتاژ الکتریکی سیگنال ، مشخص

می شود.

دستگاه های دیجیتالی (۲)

مثال:

عبارت درست است.	سطح ولتاژ بالا
عبارت نادرست است.	سطح ولتاژ پائین

➤ عملگرهای منطقی با گیت های منطقی پیاده سازی می شوند.

اصول جبر بول (۱)

اصول اساسی:

اصل ۱:

تعریف: برای هر a و b متعلق به مجموعه K هستند، و K نیز به مجموعه K متعلق دارند.

$a.b$ And $a+b$ (میدانی شود).

$$\text{If } a \& b \in K \longrightarrow \begin{cases} a.b \in K \\ a+b \in K \end{cases}$$

اصول جبر بول (۲)

اصل ۲:

موجودیت عناصر 0 و 1:

$$x + 0 = x$$

$$x \cdot 1 = x$$

x	$x + 0$	$x \cdot 1$
0	0	0
1	1	1

اصول جبر بول (۳)

اصل ۳:

$$x + y = y + x$$

$$x \cdot y = y \cdot x$$

خاصیت عناصر + و . :

x	y	x.y	y.x	x+y	y+x
0	0	0	0	0	0
0	1	0	0	1	1
1	0	0	0	1	1
1	1	1	1	1	1

اصول جبر بول^(۴)

x	y	$x+y$	$x.y$	$y.x'$	x'
0	0	0	0	0	1
0	1	1	0	1	1
1	0	1	0	0	0
1	1	1	1	1	0

اصول جبر بول (۵)

اصل ۴:

خاصیت شرکت پذیری اعمال + و .

$$(x + y) + z = x + (y + z)$$

$$x \cdot (y \cdot z) = (x \cdot y) \cdot z$$

اصول جبر بول (۶)

اصل ۵:

خاصیت توزیع پذیری + بر . و . بر +:

$$x \cdot (y + z) = x \cdot y + x \cdot z$$

$$x + (y \cdot z) = (x + y) \cdot (x + z)$$

آزمون درستی توزیع پذیری + بر . و . بر + (۲)

=

x	y	z	y.z	x+y.z	x+y	x+z	(x+y)(x+z)
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

اصول اساسی جبر بول^(۱)

۱. خاصیت خود توانی:

$$a + a = a$$

$$a \cdot a = a$$

۲. عناصر بی اثر در $+$ و \cdot :

$$a \cdot 1 = a$$

$$a + 0 = a$$

اصول اساسی جبر بول^(۲)

۳. متهم متهم:

$$a'' = a$$

۴. قانون جذب:

$$a + a \cdot b = a$$

$$a \cdot (a + b) = a$$

اصول اساسی جبر بول (۳)

۵. قانون ۵

$$\text{a) } a + a'b = a + b$$

$$\text{b) } a(a' + b) = ab$$

مثال:

$$\square B + AB'C'D = B + AC'D$$

$$\square (X + Y)((X + Y)' + Z) = (X + Y)Z$$

[5(a)ق]

[5(b)ق]

۶. قانون ۶

$$\text{a) } ab + ab' = a$$

$$\text{b) } (a + b)(a + b') = a$$

اصول اساسی جبر بول^(۳)

مثال:

- $ABC + AB'C = AC$ [ثقی(a)]
- $(W' + X' + Y' + Z')(W' + X' + Y' + Z)(W' + X' + Y + Z')(W' + X' + Y + Z)$
 $= (W' + X' + Y')(W' + X' + Y + Z')(W' + X' + Y + Z)$ [ثقی(b)]
 $= (W' + X' + Y')(W' + X' + Y)$ [ثقی(b)]
 $= (W' + X')$ [ثقی(b)]

اصول اساسی جبر بول (۳)

۷. قانون ۷

$$a) \quad ab + ab'c = ab + ac$$

$$b) \quad (a + b)(a + b' + c) = (a + b)(a + c)$$

مثال:

$$\begin{aligned} \square \quad & wy' + wx'y + wxyz + wxz' \\ &= wy' + wx'y + wxy + wxz' \\ &= wy' + wy + wxz' \\ &= w + wxz' \\ &= w \end{aligned}$$

[7(a)ق]

[7(a)ق]

[7(a)ق]

[7(a)ق]

قوانین دمرگان^(۱)

$$(x.y)' = x' + y'$$

$$(x+y)' = x'.y'$$

این قانون می تواند به صورت زیر تعمیم پیدا کند:

$$(x.y.....t)' = x' + y' + ... + t'$$

$$(x+y+...+t)' = x'.y'.....t'$$

قوانین دمرگان^(۲)

مثال:

$$\begin{aligned}\square (a + bc)' &= (a + (bc))' \\ &= a'(bc)' \\ &= a'(b' + c') \\ &= a'b' + a'c'\end{aligned}$$

قوانین دمرگان^(۳)

مثال های بیشتری از قوانین دمرگان:

$$\begin{aligned}\square (a(b + z(x + a')))' &= a' + (b + z(x + a'))' && [\neg(b)] \\ &= a' + b' (z(x + a'))' && [\neg(a)] \\ &= a' + b' (z' + (x + a'))' && [\neg(b)] \\ &= a' + b' (z' + x'(a'))' && [\neg(a)] \\ &= a' + b' (z' + x'a) && [\text{متنم متنم}] \\ &= a' + b' (z' + x') && [5(a)]\end{aligned}$$

$$\begin{aligned}\square (a(b + c) + a'b)' &= (ab + ac + a'b)' && [5(b) \text{ اصل}] \\ &= (b + ac)' && [6(a)] \\ &= b'(ac)' && [\neg(a)] \\ &= b'(a' + c') && [\neg(b)]\end{aligned}$$

اصول اساسی جبر بول^(۴)

۸. قانون ۸

$$(a) \quad ab + a'c + bc = ab + a'c$$

$$(b) \quad (a + b)(a' + c)(b + c) = (a + b)(a' + c)$$

مثال:

$$- \quad AB + A'CD + BCD = AB + A'CD$$

[9(a)ق]

$$- \quad (a + b')(a' + c)(b' + c) = (a + b')(a' + c)$$

[9(b)ق]

$$- \quad ABC + A'D + B'D + CD$$

$$= ABC + (A' + B')D + CD$$

[5(b)اصل]

$$= ABC + (AB)'D + CD$$

[d(b)]

$$= ABC + (AB)'D$$

[9(a)ق]

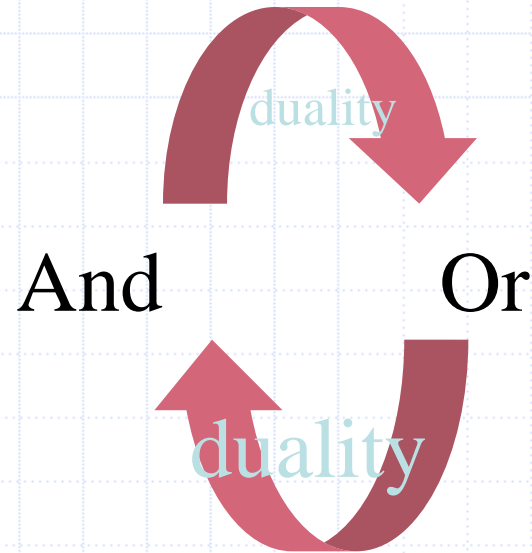
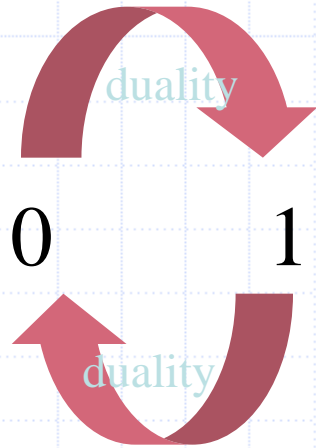
$$= ABC + (A' + B')D$$

[d(b)]

$$= ABC + A'D + B'D$$

[5(b)اصل]

دوگان (duality)



مثال:

$$x + y'z \longleftrightarrow x \cdot (y' + z)$$

دوگان

مینترم (SOP) و ماکسترم ها (POS)^(۱)

x	y	z	$x+y+z$	Minterm		Maxterm	
0	0	0	0	$x'.y'.z'$	m0	$x+y+z$	M ₀
0	0	1	1	$x'.y'.z$	m1	$x+y+z'$	M ₁
0	1	0	1	$x'.y.z'$	m2	$x+y'+z$	M ₂
0	1	1	1	$x'.y.z$	m3	$x+y'+z'$	M ₃
1	0	0	1	$x.y'.z'$	m4	$x'+y+z$	M ₄
1	0	1	1	$x.y'.z$	m5	$x'+y+z'$	M ₅
1	1	0	1	$x.y.z'$	m6	$x'+y'+z$	M ₆
1	1	1	1	$x.y.z$	m7	$x'+y'+z'$	M ₇

مینترم (SOP) و ماکسترم ها (POS)^(۲)

مثال:

$$f(x,y,z) = \sum m(1,2,4,5,6)$$



$$f(x,y,z) = \prod M(0,3,7)$$

مینترم (SOP) و ماکسترم ها (POS) (۲)

مثال: تابع زیر را به صورت مینترمی بنویسید.

$$F(x, y) = x \cdot y$$

x	y	F
۰	۰	۰
۰	۱	۰
۱	۰	۰
۱	۱	۱

۱. رسم جدول درستی

۲. تعیین مینترم ها

$$F(x, y) = \sum F(2)$$

مینترم (SOP) و ماکسترم ها (POS)(۳)

مثال: $f(A, B, Q, Z)$ و $f'(A, B, Q, Z)$ را به صورت مینترمی بنویسید.

$$f(A, B, Q, Z) = A'B'Q'Z + A'B'QZ + A'BQZ + A'BQZ$$

$$\begin{aligned} f(A, B, Q, Z) &= A'B'Q'Z + A'B'QZ + A'BQZ' + A'BQZ \\ &= m_0 + m_1 + m_6 + m_7 \\ &= S m(0, 1, 6, 7) \end{aligned}$$

$$\begin{aligned} f'(A, B, Q, Z) &= m_2 + m_3 + m_4 + m_5 + m_8 + m_9 + m_{10} + m_{11} + m_{12} \\ &\quad + m_{13} + m_{14} + m_{15} \\ &= S m(2, 3, 4, 5, 8, 9, 10, 11, 12, 13, 14, 15) \end{aligned}$$

قضیه گسترش شانون:

$$(a). f(x_1, x_2, \dots, x_n) = x_1 f(1, x_2, \dots, x_n) + (x_1)' f(0, x_2, \dots, x_n)$$

$$(b). f(x_1, x_2, \dots, x_n) = [x_1 + f(0, x_2, \dots, x_n)] [(x_1)' + f(1, x_2, \dots, x_n)]$$

مثال:

- $f(A, B, C) = AB + AC' + A'C$
 - $f(A, B, C) = AB + AC' + A'C = A f(1, B, C) + A' f(0, B, C)$
 $= A(1 \times B + 1 \times C' + 1' \times C) + A'(0 \times B + 0 \times C' + 0' \times C) = A(B + C') + A'C$
 - $f(A, B, C) = A(B + C') + A'C = B[A(1 + C')] + A'C + B'[A(0 + C') + A'C]$
 $= B[A + A'C] + B'[AC' + A'C] = AB + A'BC + AB'C' + A'B'C$
 - $f(A, B, C) = AB + A'BC + AB'C' + A'B'C$
 $= C[AB + A'B \times 1 + AB' \times 1' + A'B' \times 1] + C'[AB + A'B \times 0 + AB' \times 0' + A'B' \times 0]$
 $= ABC + A'BC + A'B'C + ABC' + AB'C'$

Xor & Xnor

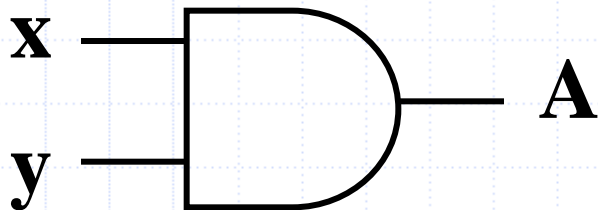
$$\square x \oplus y = x \cdot y' + x' \cdot y$$

$$\square x \odot y = x' \cdot y' + x \cdot y$$

x	y	$x \cdot y$	$x + y$	$x \oplus y$	$x \odot y$
0	0	0	0	0	1
0	1	0	1	1	0
1	0	0	1	1	0
1	1	1	1	0	1

گیت ها (دریچه ها) ^(۱)

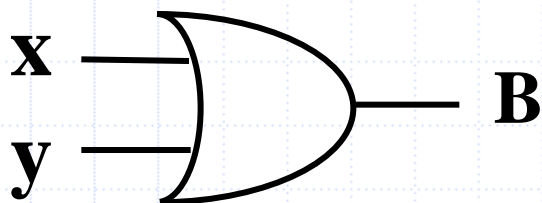
□ And:



x	y	A = x . y
0	0	0
0	1	0
1	0	0
1	1	1

گیت ها (دریچه ها) ^(۱)

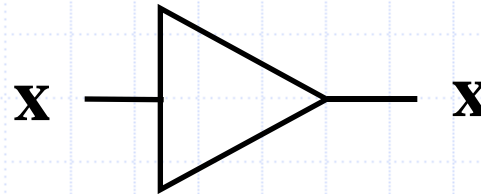
□ Or:



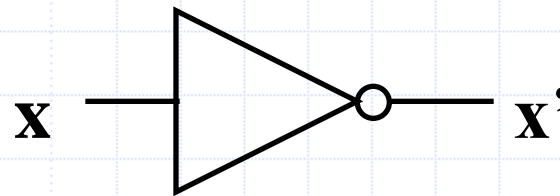
x	y	A = x + y
0	0	0
0	1	1
1	0	1
1	1	1

گیت ها (۲)

تقویت کننده:



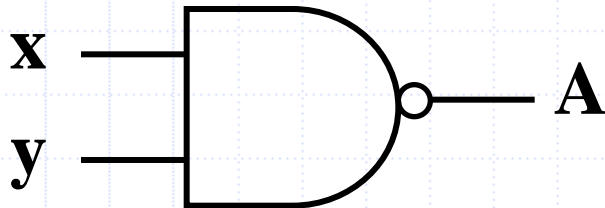
متعم:



x	x'
0	1
1	0

گیت ها (۳)

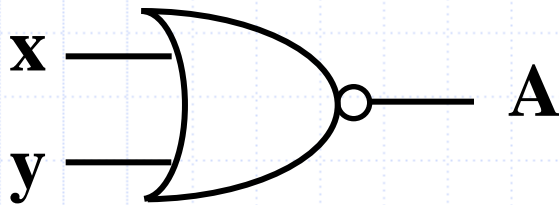
□ Nand:



x	y	A
0	0	0
0	1	0
1	0	0
1	1	1

گیت ها (۳)

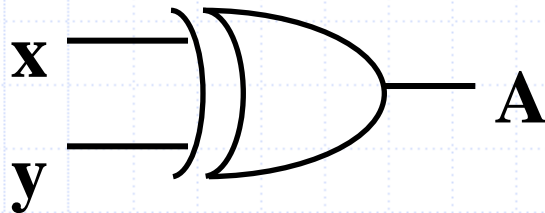
□ Nor:



x	y	A
0	0	1
0	1	0
1	0	0
1	1	0

گیت ها (۴)

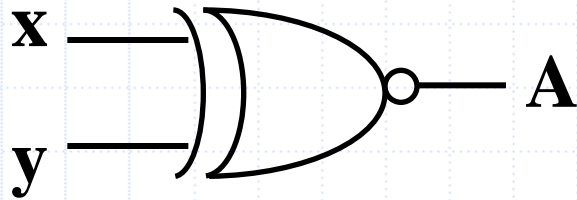
□ Xor:



x	y	A
0	0	0
0	1	1
1	0	1
1	1	0

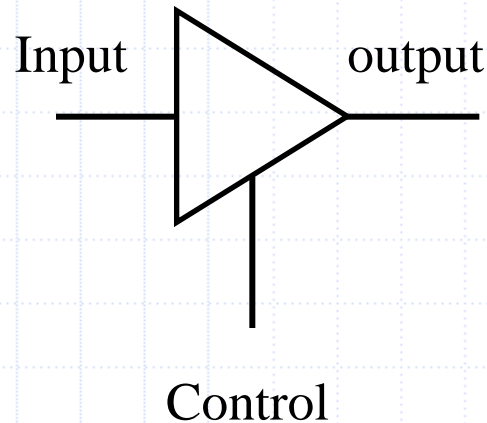
گیت ها (۴)

□ Xnor:



x	y	A
0	0	0
0	1	0
1	0	0
1	1	1

گیت یا بافر ۳ وضعیتی^(۱)

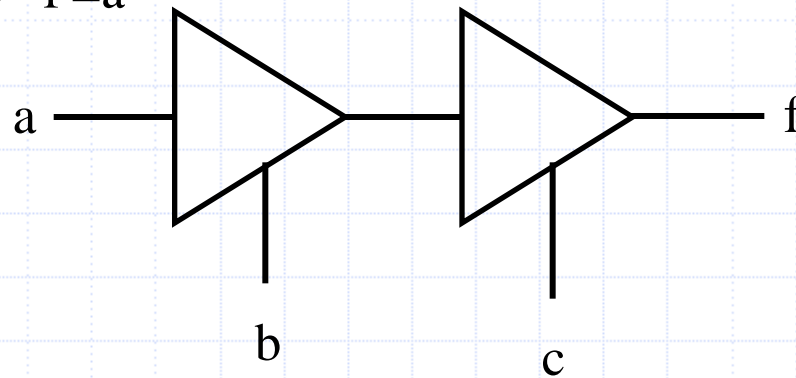
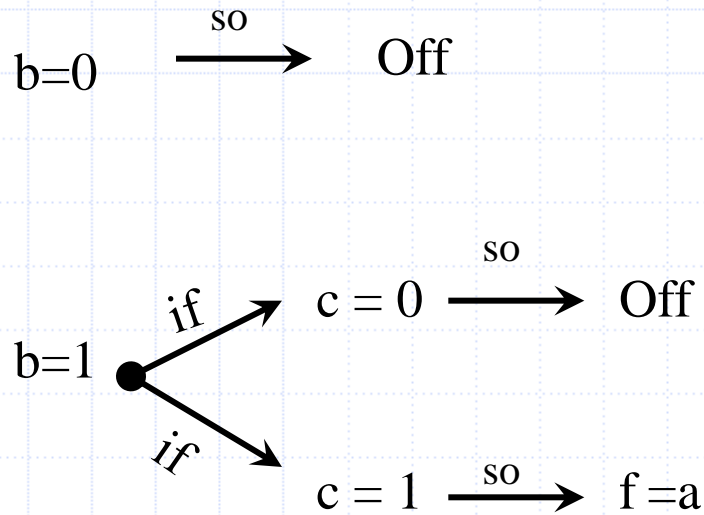


این گیت ها دارای یک درجه ورودی ،
یک خروجی و یک کلید کنترل است
که هر گاه کلید کنترل ۱ گردد ؛
ورودی بر روی خروجی قرار میگیرد.

$$\text{Output} = \begin{cases} \text{Input} & \text{If control} = 1 \\ \text{Hz} & \text{If control} = 0 \end{cases}$$

گیت یا بافر ۳ وضعیتی (۲)

اتصال سری:

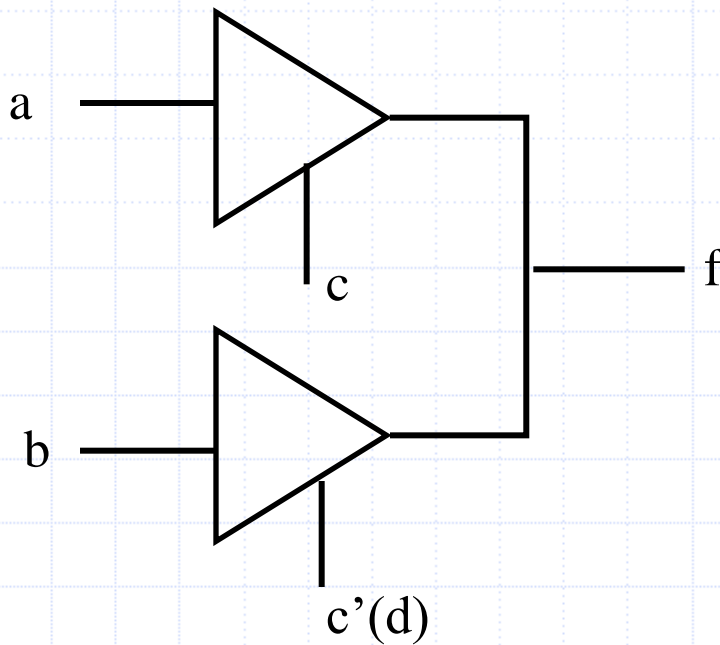


گیت یا بافر ۳ وضعیتی^(۳)

اتصال موازی:

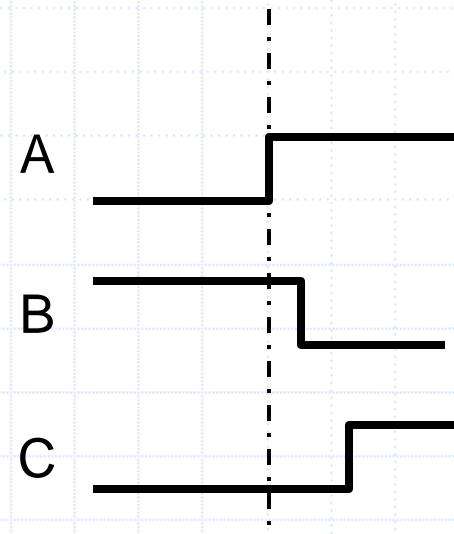
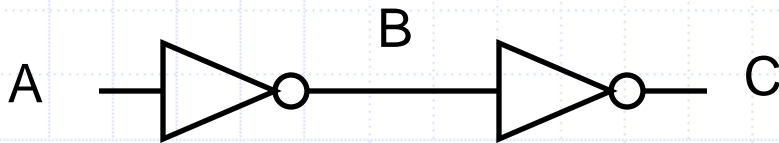
$$c = 0 \xrightarrow{so} f = b$$

$$c = 1 \xrightarrow{so} f = a$$



تأخیر در انتشار^(۱)

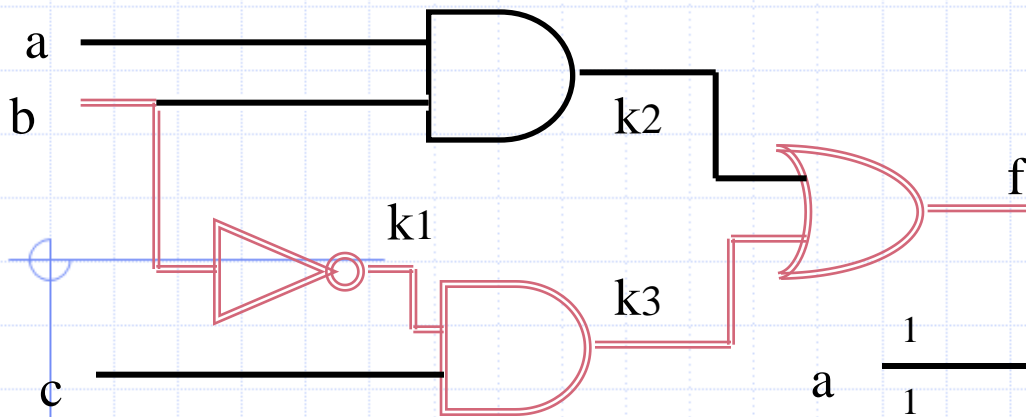
- ❑ Real implementations are not quite so perfect
- ❑ Computation actually takes some time
- ❑ Communication actually takes some time



Timing Diagram

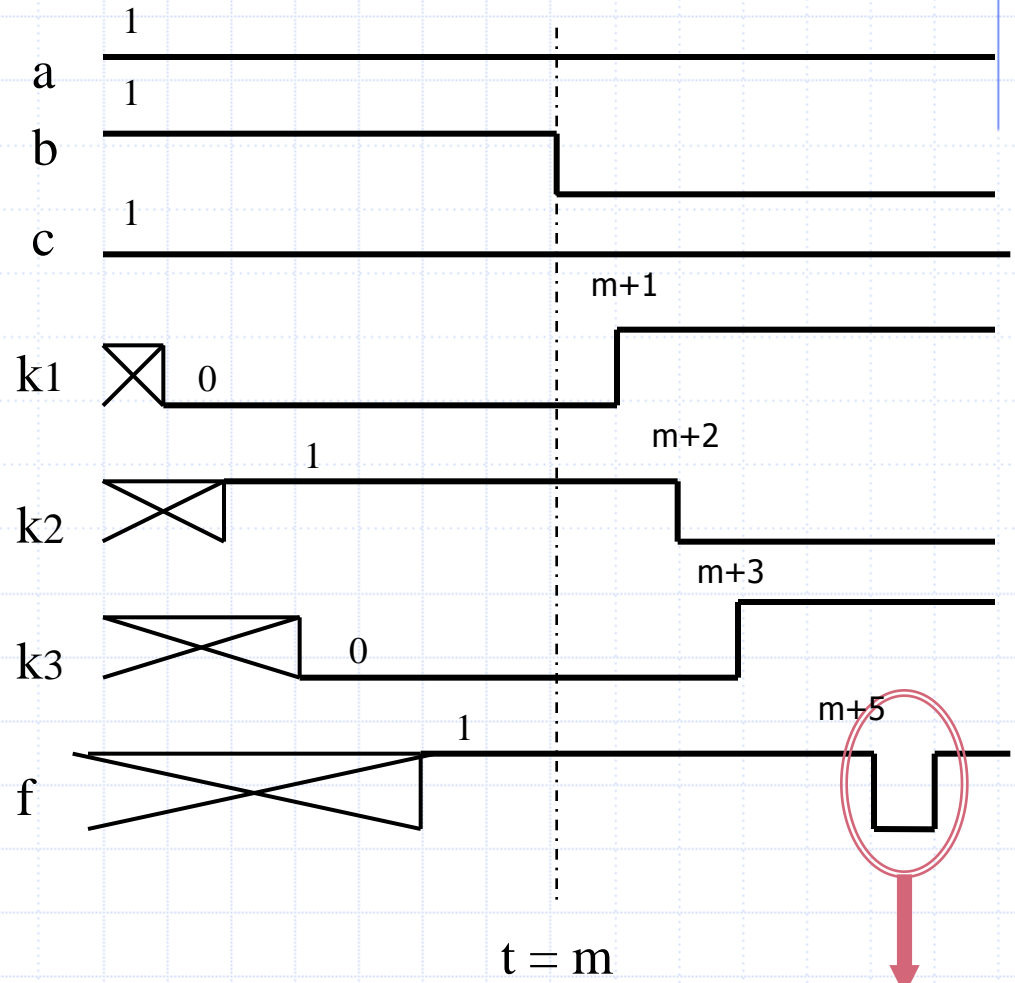
تأخير (٢)

مثال:



$$t = 0 \rightarrow \begin{cases} a=1 \\ b=1 \\ c=1 \end{cases}$$

$$t = m \rightarrow \begin{cases} a=1 \\ b=0 \\ c=1 \end{cases}$$



Hazard(1)

کد گری^(۱)

در این کد، هر کدام از کدها تنها در یک بیت با کد قبلی متفاوت است و این روند چرخشی است؛ یعنی آخرین کد و اولین کد نیز تنها در ۱ بیت متفاوتند.

کد گری (۲)

x	y	z
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Gray code

x	y	z
0	0	0
0	0	1
0	1	1
0	1	0
1	1	0
1	1	1
1	0	1
1	0	0

BCD code

نحوه تولید کدگری (۳)

۱	۱	۱	۱
۲	۱	۱	۱
۳	۱	۱	۱
۴	۱	۱	۱
۵	۱	۱	۱
۶	۱	۱	۱
۷	۱	۱	۱

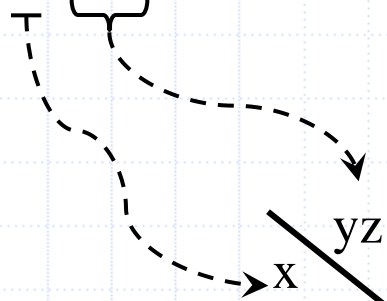
خصوصیات توابع سویچی

جدول کارنا

برای ساده سازی توابع با حداکثر ۶ ورودی ، میتوان از جدول کارنا استفاده کرد.
در این روش جدولی با توجه به تعداد ورودی ها در نظر گرفته میشود ؛ و به هر مینترم یک خانه از این جدول اختصاص میابد.

جدول کارنا برای ۳ ورودی

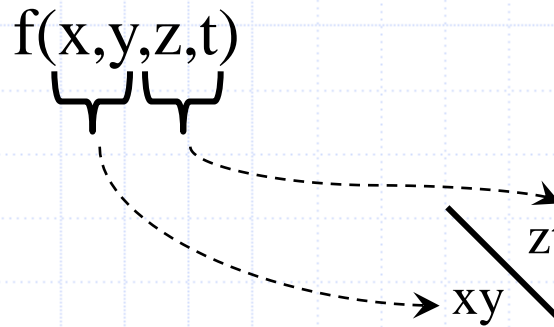
$f(x,y,z)$



	00	01	11	10
0	0	1	3	2
1	4	5	7	6

جدول کارنا برای ۴ ورودی

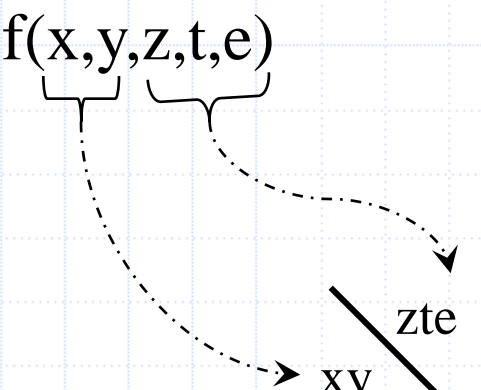
$f(x,y,z,t)$



	zt			
	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

جدول کارنا برای ۵ ورودی^(۱)

$f(x,y,z,t,e)$



	000	001	011	010		110	111	101	100
00	0	1	3	2		6	7	5	4
01	8	9	11	10		14	15	13	12
11	24	25	27	26		30	31	29	28
10	16	17	19	18		22	23	21	20

جدول کارنا برای ۵ ورودی (۲)

$f(x,y,z,t,e)$

به جای ۱ جدول ۳۲ خانه ای میتوان از ۲ جدول ۱۶ خانه ای استفاده کرد.

yz \ te				
	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

$x=0$

xy \ zt				
	00	01	11	10
00	16	17	19	18
01	20	21	23	22
11	28	29	31	30
10	24	25	27	26

$x=1$

جدول کارنا برای ۵ ورودی (۳)

$f(x,y,z,t,e)$

		zt			
		00	01	11	10
xy	00	1	3	7	5
	01	9	11	15	13
	11	25	27	31	29
	10	17	19	23	21

$e=1$

		zt			
		00	01	11	10
xy	00	0	2	6	4
	01	8	10	14	12
	11	24	26	30	28
	10	16	18	22	20

$e=0$

ساده سازی توابع با کمک جدول کارنا

۱. رسم جدول کارنا با توجه به سائزها

۲. آوردن مینترم ها داخل جدول کارنا

۳. تعیین cube

۴. تبدیل cube ها به شکل جبری

اصول ساده سازی کارنا

انتخاب `cube` در صورتی درست است که کلیه شرایط زیر برقرار باشد:

۱. قابل بزرگتر شدن نباشد.
۲. حداقل یک `cube` در وجود باشد که در هیچ `cube` دیگری شرکت نکرده باشد.

Algorithm ⁽¹⁾

- ❑ 1. count the number of adjacencies for each minterm on the k-map.
- ❑ 2. select an uncovered minterm with the fewest number of adjacencies.
- ❑ 3. generate a prime implicant, select the one that covers the most uncovered minterms.
- ❑ 4. Repeat step 2 & 3 until all minterms have been covered

مثالی برای جدول کارنا

$$f(x,y,z,t,e) = \sum m(2,4,5,6,7,8,9,10,11,12,13,15,16,18,22,24,25,27,28,29,31)$$

The Karnaugh map is a 4x8 grid. The columns are labeled with 5-bit binary strings (x,y,z,t,e) and the rows are labeled with 4-bit binary strings (x,y,z,t). The map contains 1s in the following cells (row, column): (00, 010), (00, 110), (00, 111), (00, 101), (00, 100), (01, 000), (01, 001), (01, 011), (01, 010), (01, 111), (01, 101), (01, 100), (11, 000), (11, 001), (11, 011), (11, 111), (11, 101), (11, 100), (10, 000), (10, 010), (10, 110), (10, 111). The map is divided into two 4x4 sections by a vertical dashed line. A dashed blue arrow indicates a wrap-around from the right edge to the left edge. A dashed red arrow indicates a wrap-around from the bottom edge to the top edge. Several groups of 1s are circled in red and blue, representing prime implicants.

zte \ xy	000	001	011	010	110	111	101	100
00				1	1	1	1	1
01	1	1	1	1		1	1	1
11	1	1	1			1	1	1
10	1			1	1			

$$f(x,y,z,t,e) = xyz + x'yz' + xz't'e' + ye + yt' + y'te'$$

توابع نا کامل (با don't-care) (۱)

دلیل که در ورودی don't-care حالات بی اهمیتی هستند در خروجی به این اتفاق نمیافتد.

از این حالات به عنوان یک مؤلفه ی موثر در ساده سازی به خوبی میتوان استفاده کرد ؛ به این صورت که اگر ۱ بودن برخی از این حالات باعث بزرگتر شدن ها و ساده سازی بیشتر شود ، ما آنها را ۱ فرض میکنیم و اگر نه ، به نفع ماست که آنها را ۰ فرض کنیم.

توابع نا کامل (don't-care) (۲)

$$f(x,y,z,t) = \sum m(1,2,7,11,12,15) + d(0,3,6,9,13,14)$$

$$f(x,y,z,t) = x'z + xy + y't$$

xy \ zt	00	01	11	10
00	*	1	*	1
01	*	*	1	*
11	1	*	1	*
10	*	*	1	*

انواع شکل مدارات ۲ طبقه^(۱)

می دانیم هر تابع جبری با هر شکل و اندازه ای با استفاده از یک جدول درستی قابل نمایش است ؛ و به فرم ۲ طبقه ی

و

نیز مفیداند ؛ میخواهیم ببینیم چه فرم

And-Or Or-And

های ۲ طبقه دیگری وجود دارد.

Nand Nor

انواع شکل مدارات ۲ طبقه (۲)

طبقه ۰	طبقه ۱	طبقه ۲
Not	And	And
	Or	Or
	Nand	Nand
	Nor	Nor

حالات ممکن مدارات ۲ طبقه

طبقه ۱	طبقه ۲			
	And	Or	Nand	Nor
And		★		★
Or	★		★	
Nand	★		★	
Nor		★		★

مورب جدول کارنا cube

ساده سازی

مثال:

zt		00	01	11	10
xy	00		1		1
	01	1		1	
	11		1		1
	10	1		1	

$$f(x,y,z,t) = a'.c'(b \oplus d) + a.c(b \oplus d) + a'.c(b \odot d) + a.c'(b \cdot d)$$

$$f(x,y,z,t) = (b \oplus d) \odot (a \odot c)$$

روش ساده سازی کویین مک کلاسکی

(Quine-McCluskey) (۱)

روش دیگری برای ساده سازی توابع می باشد.

مزیت این روش به جدول کارنا ، اینست که اگر ورودی های ما زیاد هم باشند ؛ کار کردن با آن ساده است ، ولی جدول کارنا برای توابعی با بیش از ۶ ورودی کاربردی ندارد زیرا کار کردن با آن ساده نیست.

روش ساده سازی کویین مک کلاسکی

(Quine-McCluskey) (۲)

مراحل و روش این نوع ساده سازی را به همراه یک مثال می بینیم.

روش ساده سازی کوپین مک کلاسی

(۳) (Quine-McCluskey)

مثال:

$$f(a,b,c,d) = \sum m(2,4,6,8,9,10,12,13,15)$$

		cd			
		00	01	11	10
ab	00				1
	01	1			1
	11	1	1	1	
	10	1	1		1

Q-M Tabular Minimization Method ⁽⁴⁾

- Step 1. list in a column all the minterms of the function to be minimized in their binary representation. Partition them into groups according to the number of 1 bits in their binary representation. This partitioning simplifies identification of logically adjacent minterms since, to be logically adjacent, two minterms must differ in exactly one literal.

Q-M Tabular Minimization Method ⁽⁵⁾

Minterms	a b c d	
2	0 0 1 0	
4	0 1 0 0	Group 1 (a single 1)
8	1 0 0 0	
<hr/>		
6	0 1 1 0	
9	1 0 0 1	Group 2 (two 1's)
10	1 0 1 0	
12	1 1 0 0	
<hr/>		
13	1 1 0 1	Group 3 (three 1's)
<hr/>		
15	1 1 1 1	Group 4 (four 1's)

Q-M Tabular Minimization Method ⁽⁶⁾

- Step 2. perform an exhaustive search between neighboring groups for adjacent minterms and combining them into a column of $(n-1)$ -variable implicants, checking off each minterm that is combined. Repeat for each column, combining $(n-1)$ -variable implicants into $(n-2)$ -variable implicants, and so on, until no further implicants can be combined.

Q-M Tabular Minimization Method (7)

Minterms	a b c d	Minterms	a b c d	Minterms	a b c d
2	0010 ✓	2,6	0-10 PI₂	8,9,12,13	1-0- PI₁
4	0100 ✓	2,10	-010 PI₃		
8	1000 ✓	4,6	01-0 PI₄		
6	0110 ✓	4,12	-100 PI₅		
9	1001 ✓	8,9	100- ✓		
10	1010 ✓	8,10	10-0 PI₆		
12	1100 ✓	8,12	1-00 ✓		
13	1101 ✓	9,13	1-01 ✓		
15	1111 ✓	12,13	110- ✓		
		13,15	11-1 PI₇		

Q-M Tabular Minimization Method ⁽⁸⁾

- the final result is a list of prime implicants of the switching function.
- Step 3. construct a prime implicants chart that lists minterms along the horizontal and prime implicants along the vertical, with an * entry placed wherever a certain prime implicant (row) covers a given minterm (column).

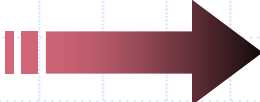
Q-M Tabular Minimization Method ⁽¹⁰⁾

- Step 4. Select a minimum number of prime implicants that cover all the minterms of the switching function.

Q-M Tabular Minimization Method ⁽¹¹⁾

	✓ 2	✓ 4	✓ 6	✓ 10
PI ₂	*		*	
PI ₃	*			*
PI ₄		*	*	
PI ₅		*		
PI ₆				*

Q-M Tabular Minimization Method ⁽¹²⁾


$$\begin{aligned} f(a,b,c,d) &= \text{PI}_1 + \text{PI}_3 + \text{PI}_4 + \text{PI}_7 \\ &= 1-0- + -010 + 01-0 + 11-1 \\ &= a.c' + b'.c.d' + a'.b.d' + a.b.d \end{aligned}$$

ساده سازی برای سیستم های چند خروجی Q-M

حال از این روش برای ساده سازی سیستم های با چند ورودی متفاوت استفاده می کنیم.

روش کار را با یک مثال می بینیم.

$$\left\{ \begin{array}{l} f_{\alpha}(a,b,c,d)=\sum m(0,2,7,10)+d(12,15) \\ f_{\beta}(a,b,c,d)=\sum m(2,4,5)+d(6,7,8,10) \\ f_{\gamma}(a,b,c,d)=\sum m(2,7,8)+d(0,5,13) \end{array} \right.$$

ساده سازی Q-M برای سیستم های چند خروجی (۲)

مینترم ها: 0,2,4,5,6,7,8,10,12,13,15

در ابتدا فرض میکنیم همه ی مینترم ها و ها don't-care

داده شده مربوط به ۱ تابع میباشد و آنها را دسته بندی

میکنیم و مرحله ۱ و ۲ را به صورت گفته شده در قسمت قبل

انجام میدهیم.

ساده ساز Q-M برای سیستم های چند خروجی (۳)

MIN TERM	abcd	Flags		MIN TERM	abcd	Flags		MIN TERM	abcd	Flags	
0	0000	$\alpha\gamma$	✓	0,2	00-0	$\alpha\gamma$	PI ₂	4,5,6,7	01--	β	PI ₁
2	0010	$\alpha\beta\gamma$	PI ₁₀	0,8	-000	γ	PI ₃				
4	0100	β	✓	2,6	0-10	β	PI ₄				
8	1000	$\beta\gamma$	PI ₁₁	2,10	-010	$\alpha\beta$	PI ₅				
5	0101	$\beta\gamma$	✓	4,5	010-	β	✓				
6	0110	β	✓	4,6	01-0	β	✓				
10	1010	$\alpha\beta$	✓	8,10	10-0	β	PI ₆				
12	1100	α	PI ₁₂	5,7	01-1	$\beta\gamma$	PI ₇				
7	0111	$\alpha\beta\gamma$	PI ₁₃	5,13	-101	γ	PI ₈				
13	1101	γ	✓	6,7	011-	β	✓				
15	1111	α	✓	7,15	-111	α	PI ₉				

Q-M

ساده سازی برای سیستم های چند خروجی (۴)

		0	2	7	10	2	4	5	2	7	8
PI ₁	β						*	*			
PI ₂	$\alpha\gamma$	*	*						*		
PI ₃	γ										*
PI ₄	β					*					
PI ₅	$\alpha\beta$		*		*						
PI ₆	β										
PI ₇	$\beta\gamma$							*		*	
PI ₈	γ										
PI ₉	α			*							
PI ₁₀	$\alpha\beta\gamma$		*			*			*		
PI ₁₁	$\beta\gamma$										*
PI ₁₂	α										
PI ₁₃	$\alpha\beta\gamma$			*						*	

ساده سازی Q-M برای سیستم های چند خروجی (۵)

		f_α	f_γ	
		✓ 7	✓ 7	✓ 8
PI ₃	γ			*
PI ₇	$\beta\gamma$		*	
PI ₉	α	*		
PI ₁₁	$\beta\gamma$			*
PI ₁₃	$\alpha\beta\gamma$	*	*	

$$f_\alpha = PI_2 + PI_5 + PI_{13}$$

$$f_\beta = PI_1 + PI_5$$

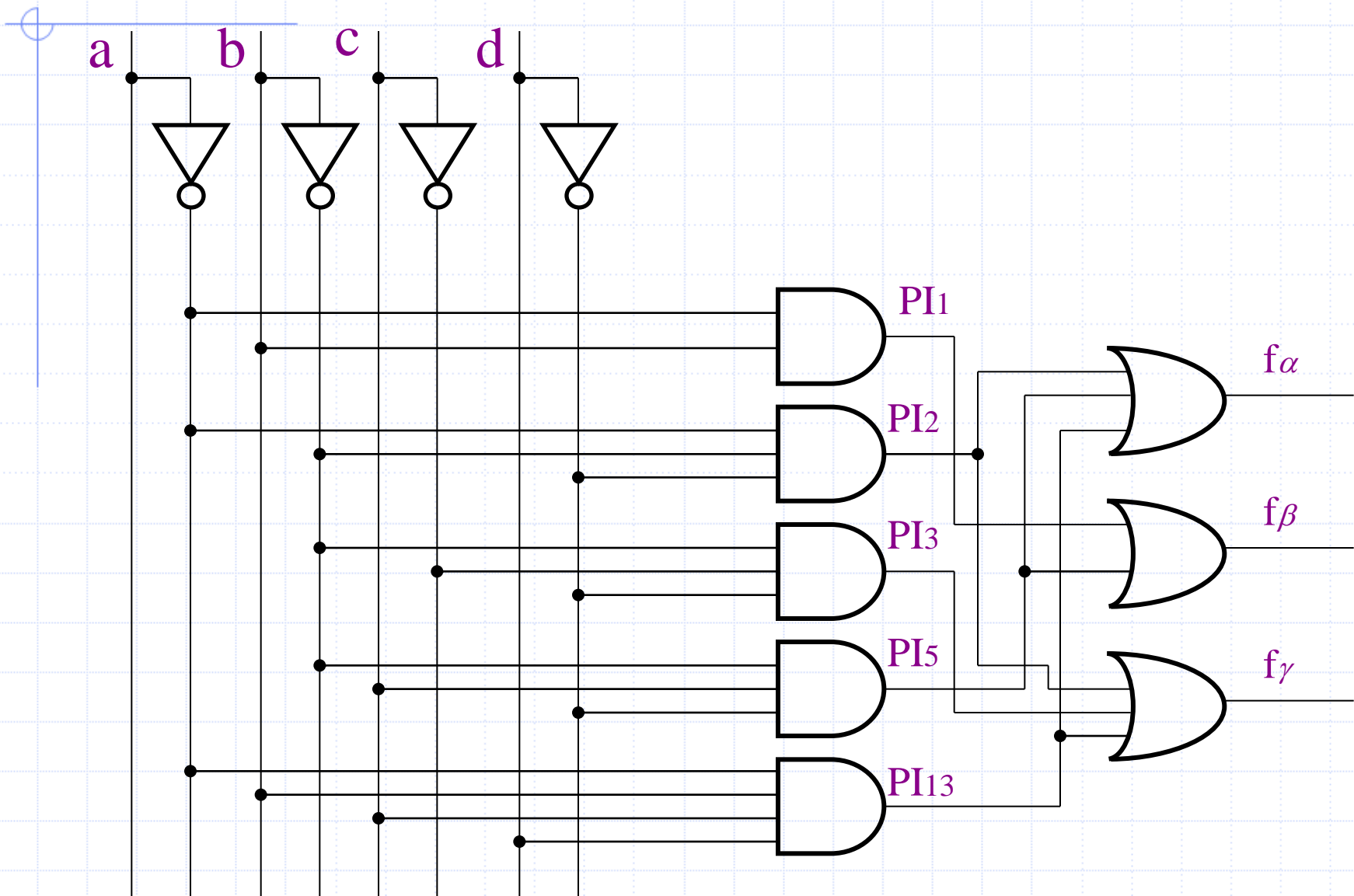
$$f_\gamma = PI_2 + PI_3 + PI_{13}$$

$$f_a = a'b'd' + b'cd' + a'bcd$$

$$f_\beta = a'b + b'cd'$$

$$f_\gamma = a'b'd' + b'c'd' + a'bcd$$

ساده سازی Q-M برای چند خروجی (۶)



فصل ۴

مدارهای منطقی ترکیبی مایجولی

فهرست مطالب

- ☐ طراحی مدار
- ☐ طراحی مایکروپروسسور
- ☐ Half Adder و Full Adder
- ☐ دیکدر
- ☐ اینکدر
- ☐ مالتی پلکسر (تسهیم کننده)
- ☐ دی مالتی پلکسر (پخش کننده داده ورودی)
- ☐ مقایسه گر
- ☐ A seven segment display

طراحی مدار

□ تعیین تعداد بیت های ورودی و خروجی مدار Interface

□ رسم جدول Truth Table

□ بدست آوردن یک تابع برای خروجی

□ ساده سازی توابع بدست آمده (کارتو / Q-M)

مثال :

Truth table

a	b	c	Even Parity
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$$p_e = \sum m(1, 2, 4, 7)$$

b c	00	01	11	10
a				
0	0	1	0	1
1	1	0	1	0

$$P_e = (a \oplus b) \oplus c$$

طرحي ماجولار مدار

اگر تعداد بیت های ورودی و خروجی بیش از ۴ یا ۵ باشد در رسم جدول صحت با مشکل برخورد می کنیم. (پیچیدگی حافظه)

راهکار

- بدون رسم جدول درستی به خروجی مدار برسیم. (رهیافت ذهنی)
- طراحی ماجولار مدار. (طراحی پیمانه ای) (از نظر زمانی بهینه نیست)

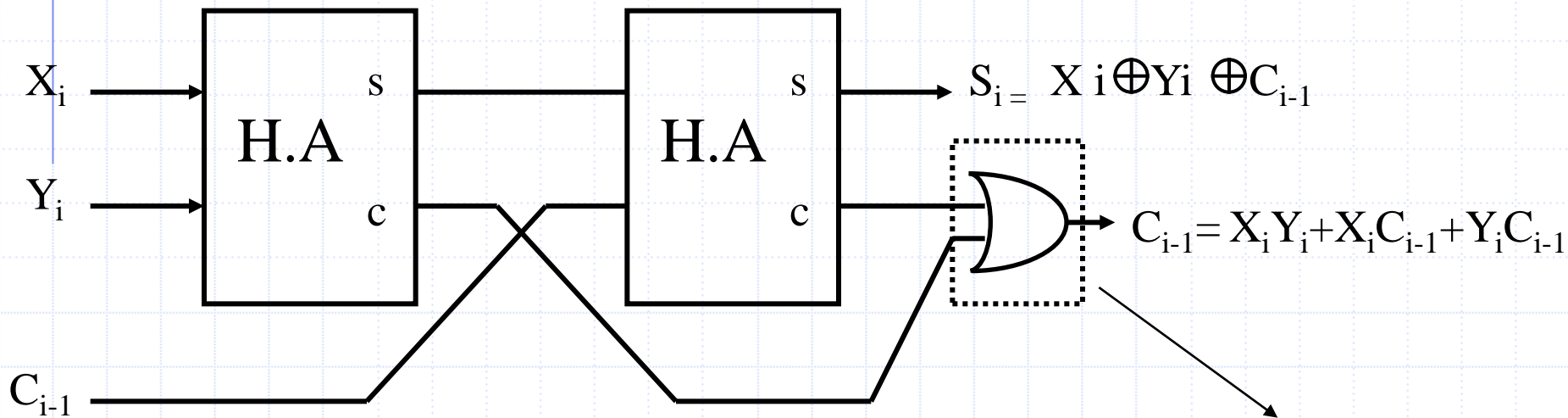
(١) Half Adder, Full Adder

Full Adder □ يك مدار ترکیبی با سه ورودی و دو خروجی است که دو بیت داده و یک رقم نقلی را با هم جمع کرده و حاصل جمع و رقم نقلی را محاسبه می کند.

Half Adder □ يك مدار ترکیبی با دو ورودی و دو خروجی است که دو بیت دودویی را با هم جمع کرده و حاصل جمع و رقم نقلی را محاسبه می کند.

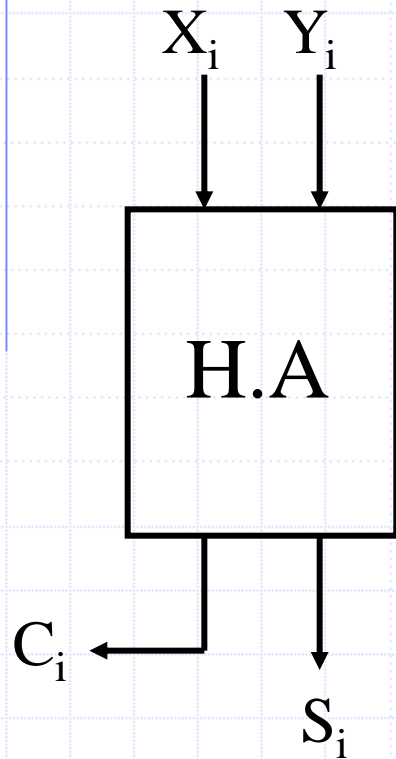
(۲) Half Adder , Full Adder

يك Full adder را ميتوان توسط ۲ عدد Half adder طراحي كرد.



مي تواند توسط يك گيت XOR
جايزين شود.

بلوك دياگرام (H.A)

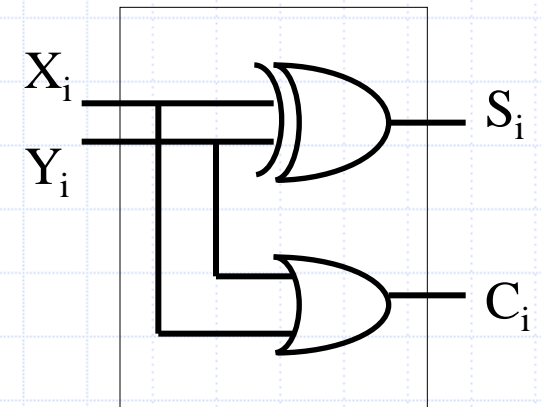


Truth Table

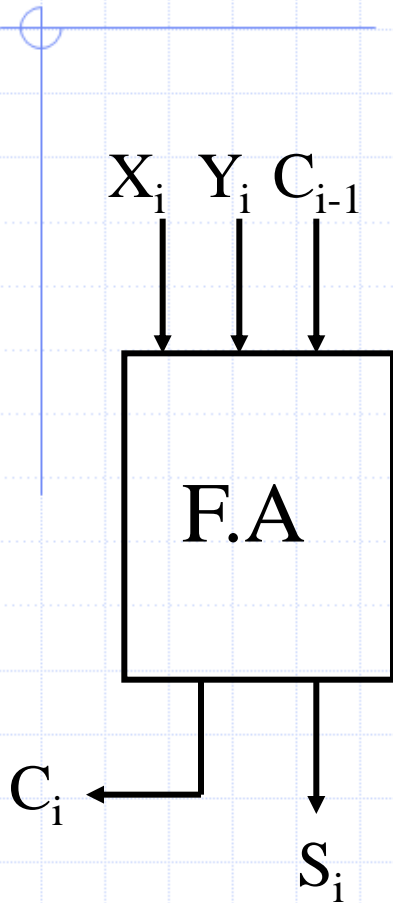
X_i	Y_i	C_i	S_i
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



$$\begin{cases} S_i = X_i \oplus Y_i \\ C_i = X_i Y_i \end{cases}$$



بلوك دياگرام (F.A)



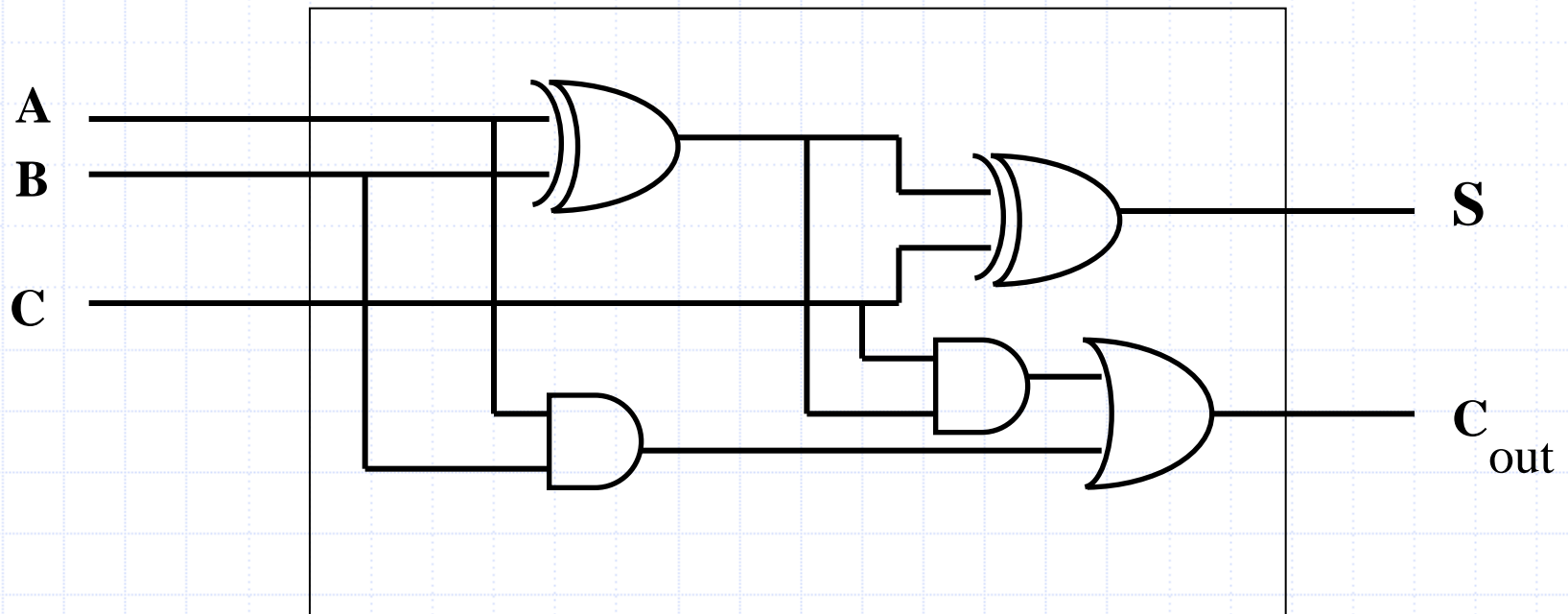
Truth Table

X_i	Y_i	C_{i-1}	C_i	S_i
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

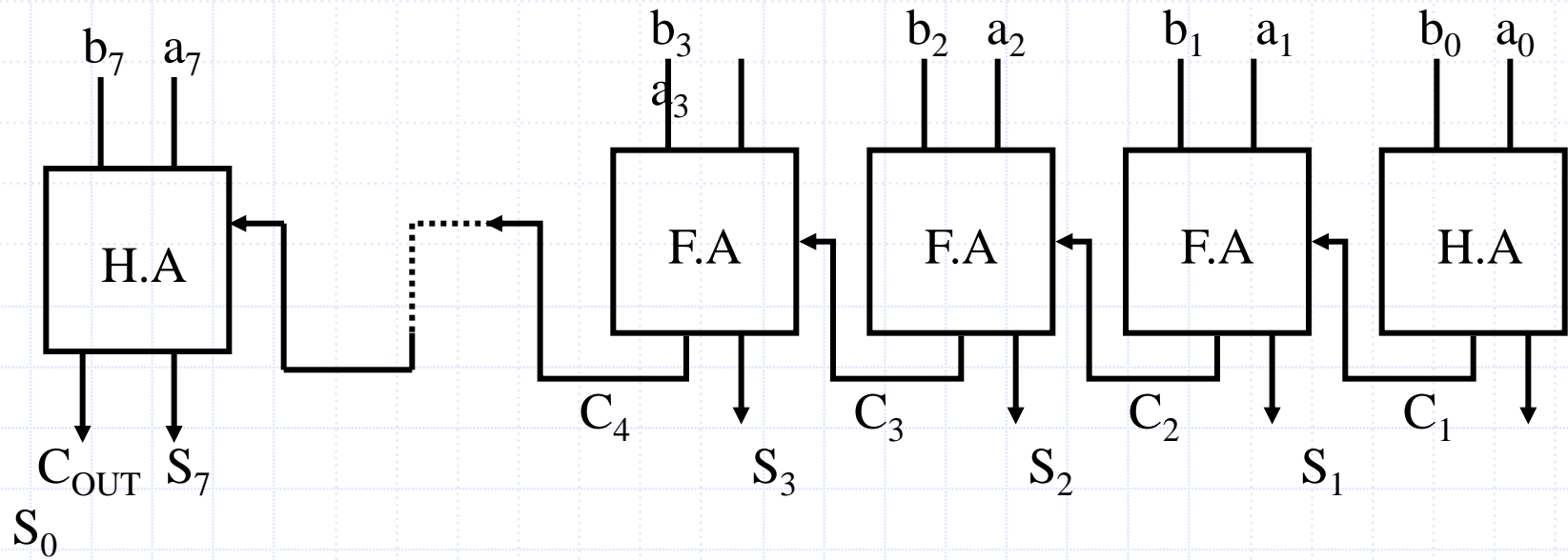
$$\rightarrow \begin{cases} S_i = X_i \oplus Y_i \oplus C_{i-1} \\ C_i = X_i Y_i + X_i C_{i-1} + Y_i C_{i-1} \end{cases}$$

دیاگرام منطقي (F.A)

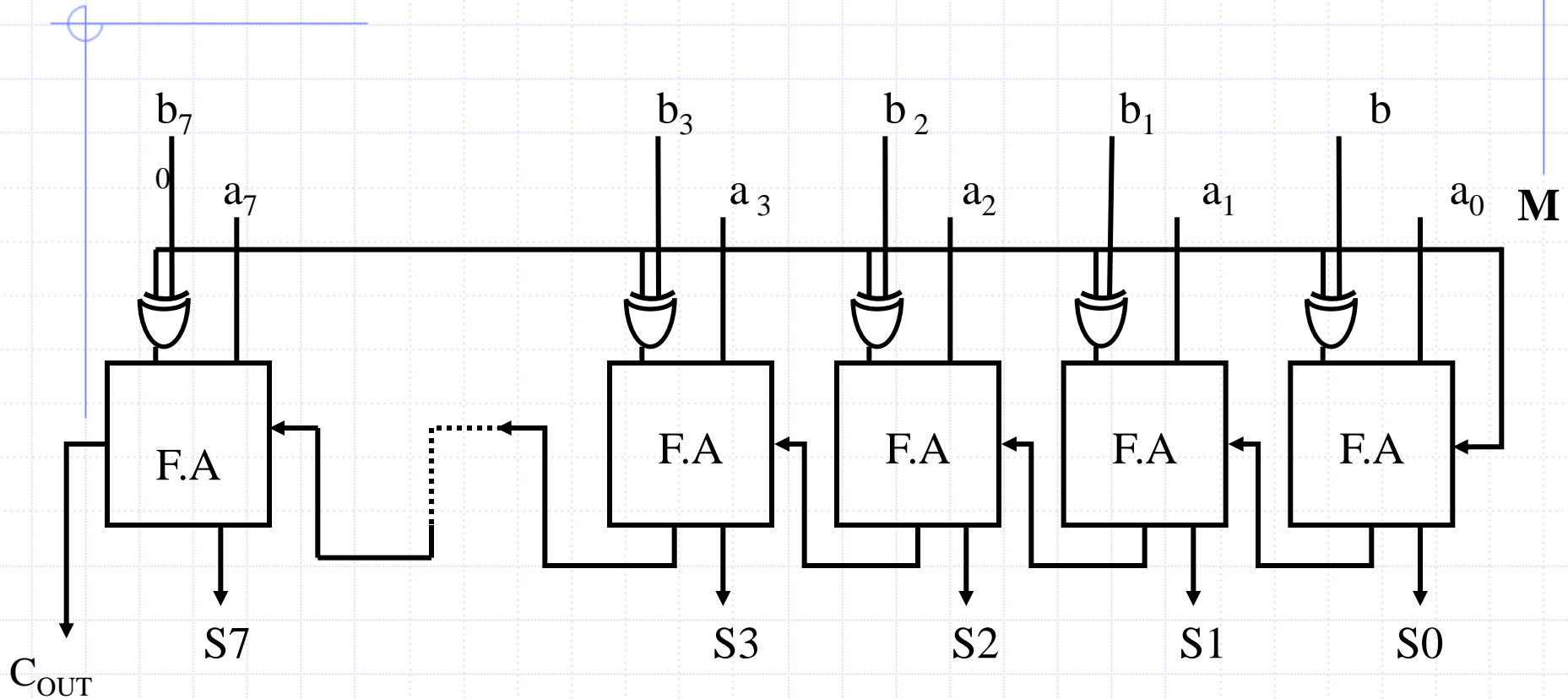
$$C_{out} = C (A \oplus B) + AB$$



Ripple Carry Adder (RCA)



Ripple Carry Adder (RCA)



If $M = 0 \Rightarrow A + B$

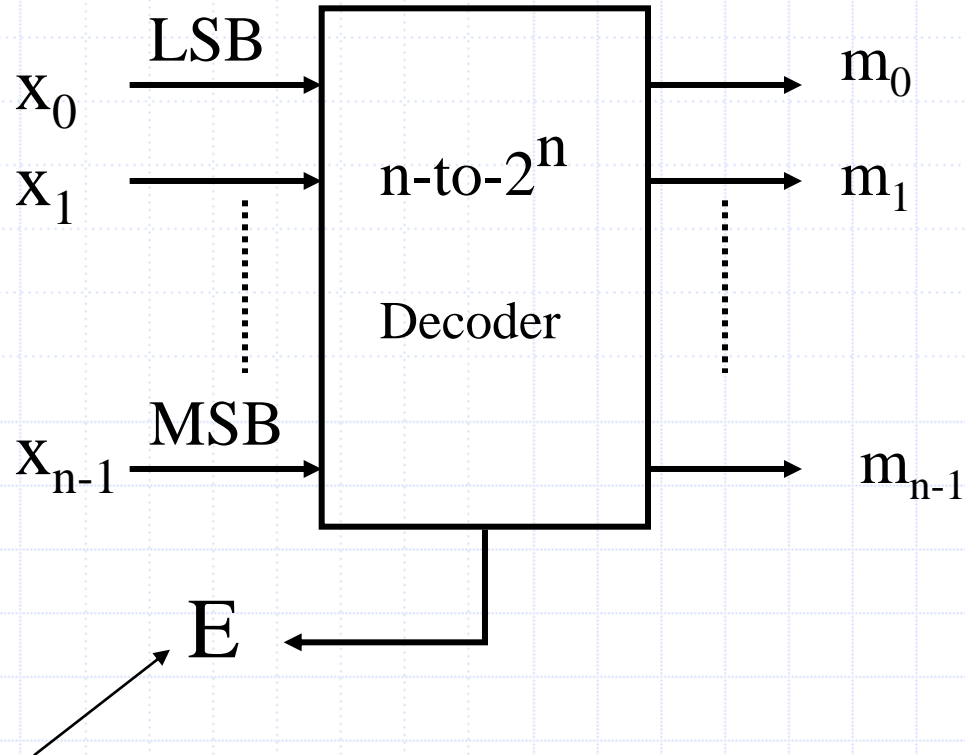
If $M = 1 \Rightarrow A - B \text{ or } (A + \bar{B} + 1)$

ديکدر

□ ديکدر n به 2^n يك شبکه منطقي ترکیبي است با n خط ورودی و 2^n سیگنال خروجی.

□ عنصری است که مینتروم ها را می سازد.

ماحول دیکدر n به 2^n

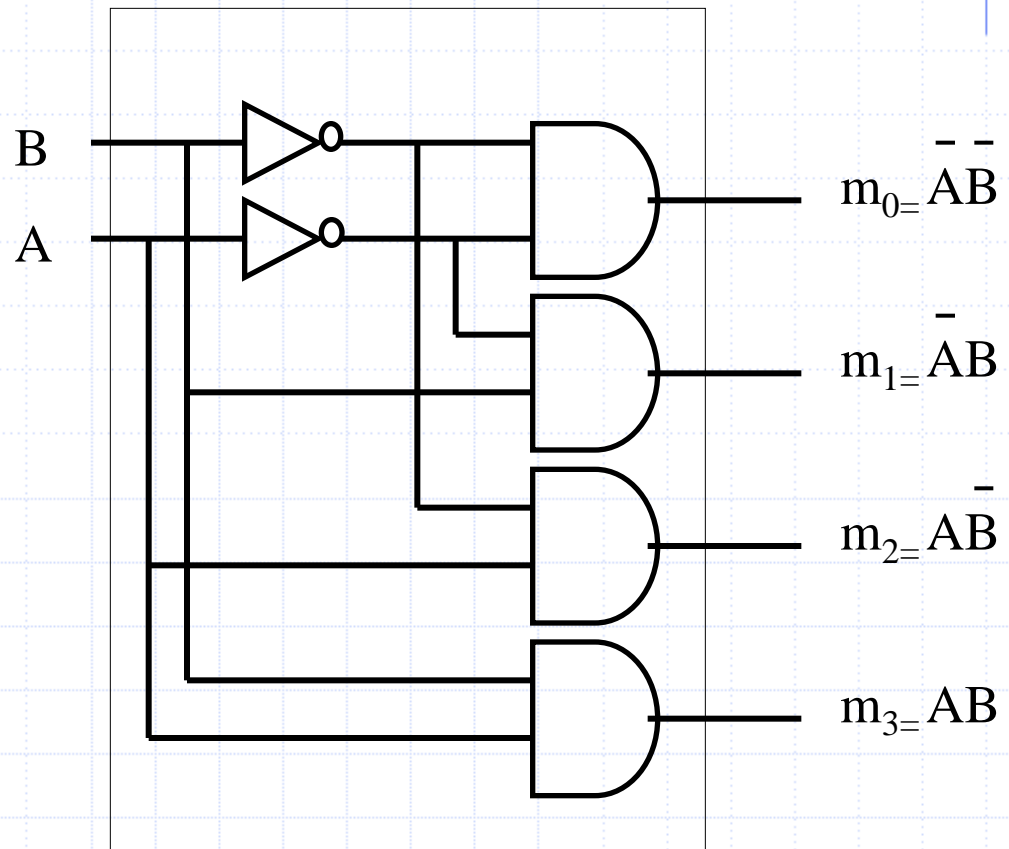


معمولا Active Low هستند.

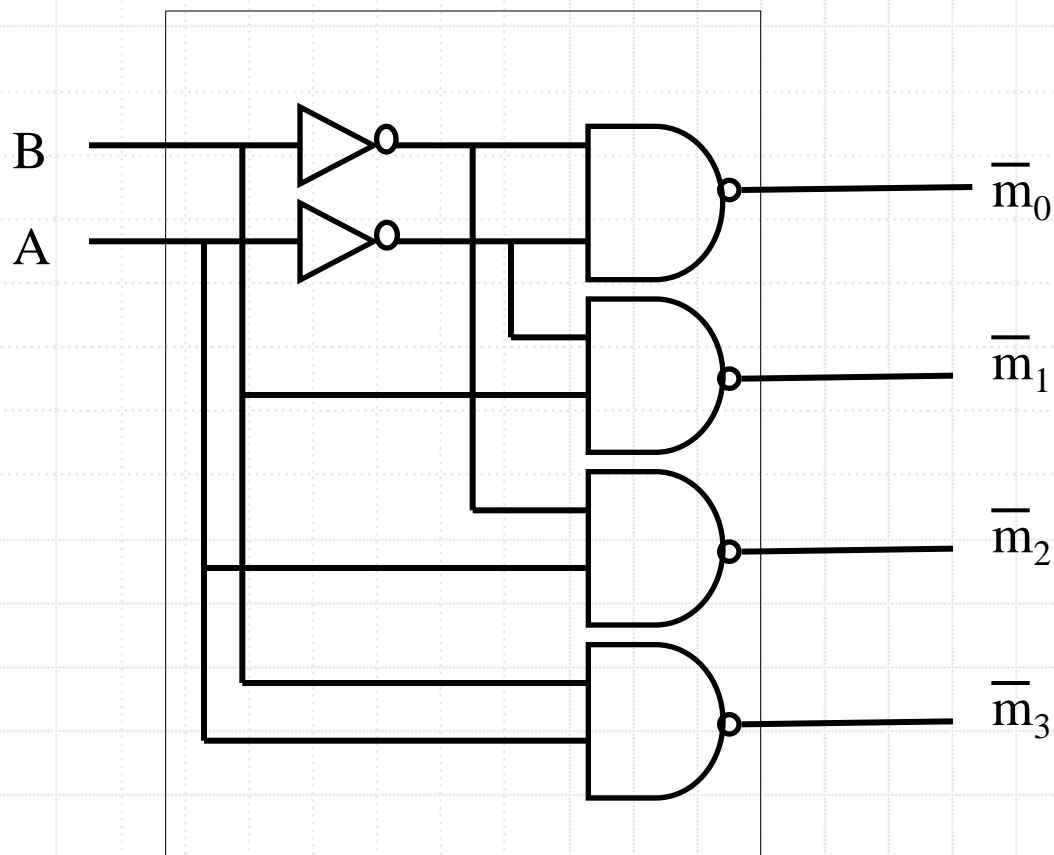
دیاگرام منطقي (موازي و خروجي هاي فعال بالا)

Truth Table

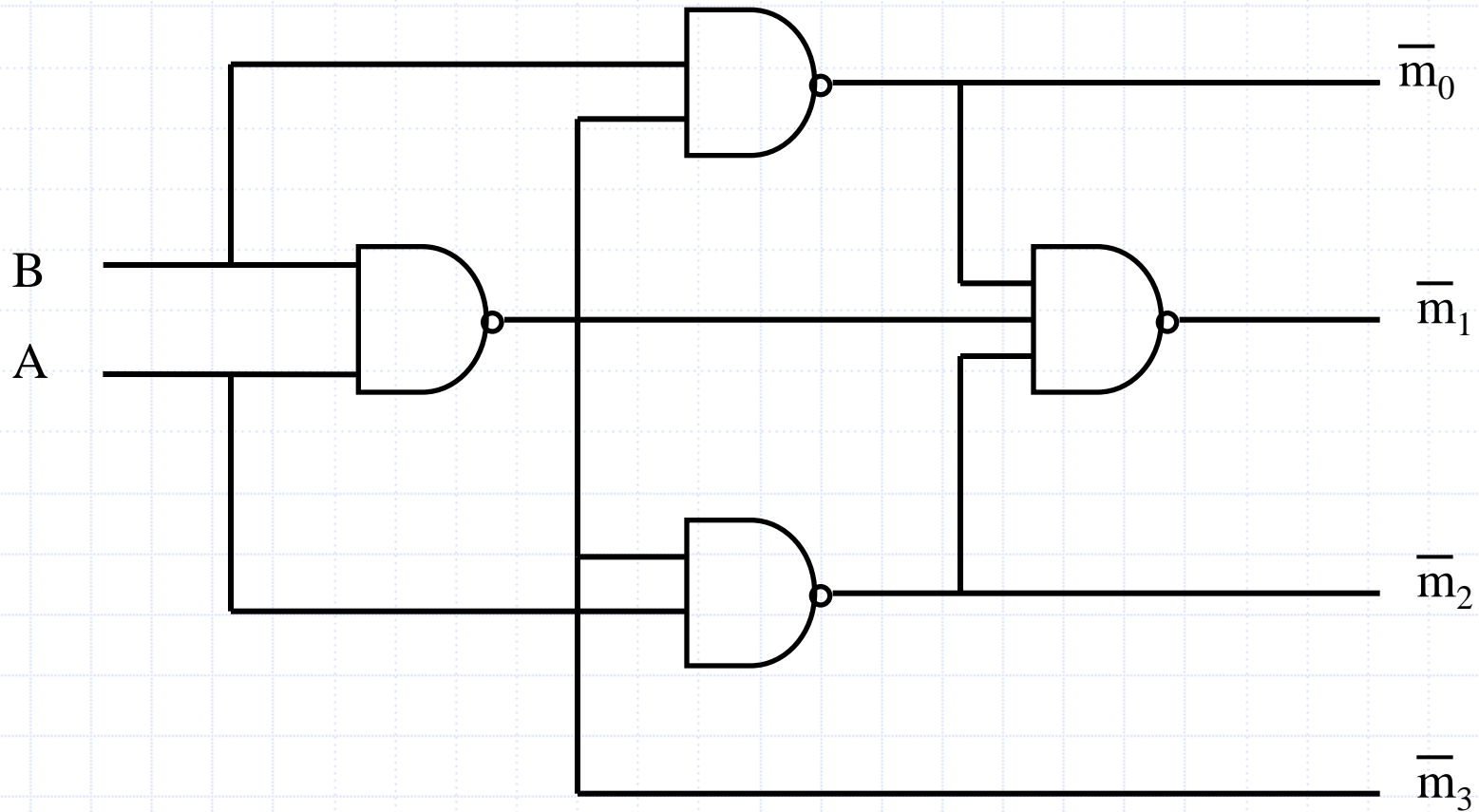
E	A	B	m_0	m_1	m_2	m_3
0	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1
0	×	×	0	0	0	0



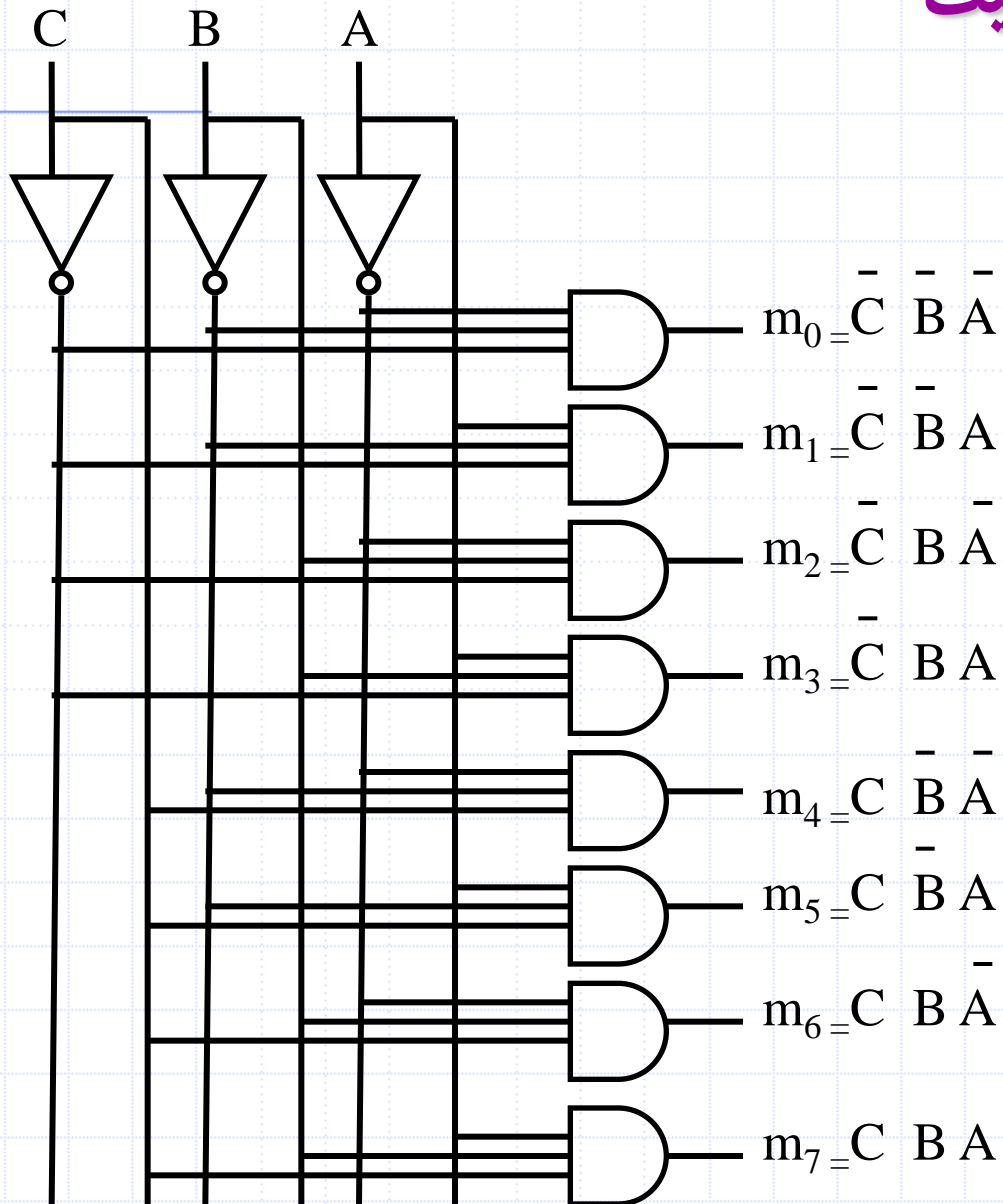
دیاگرام منطقي (موازي و خروجي هاي فعال پايين)



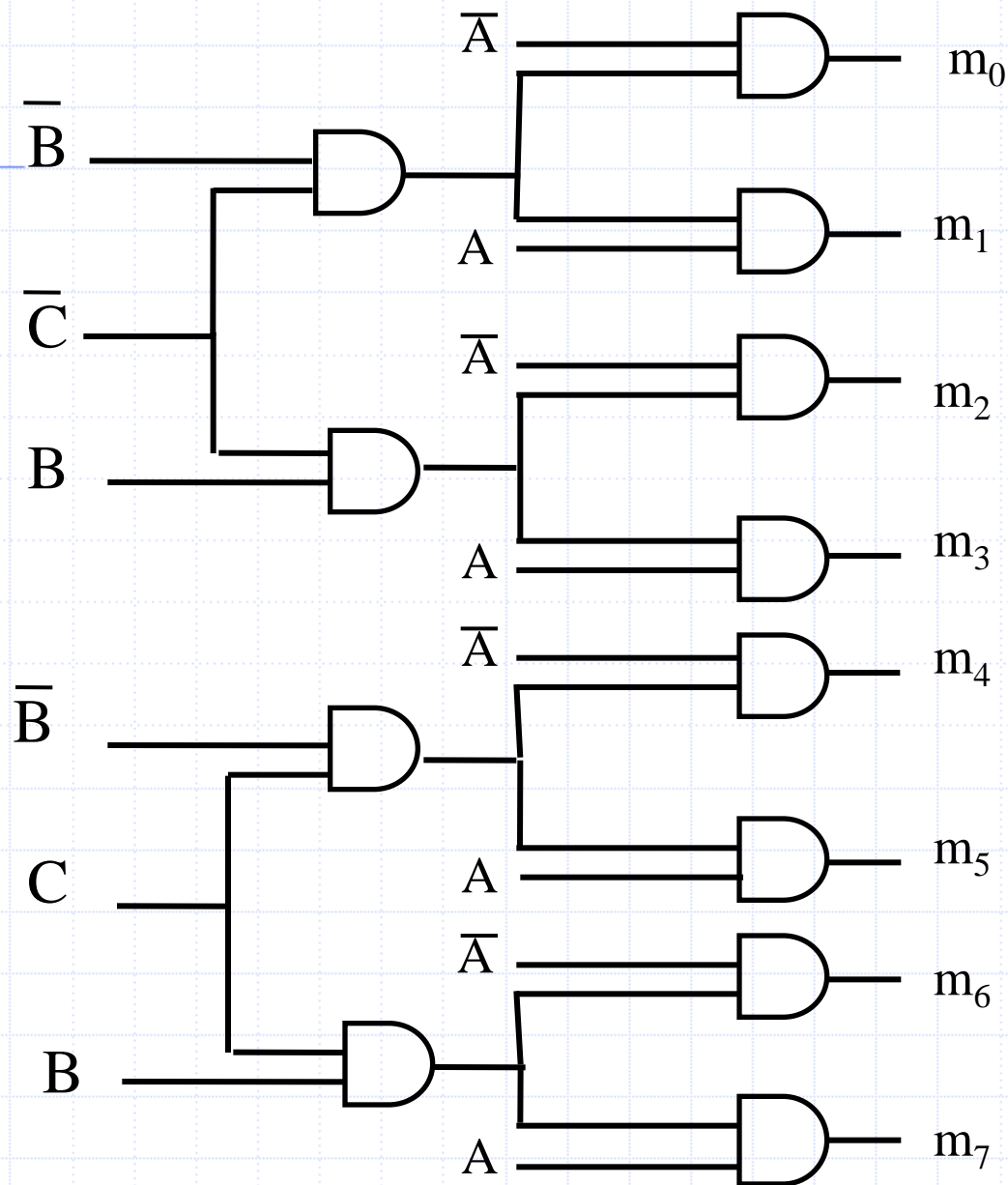
ساختهانی دیگر



دیگر نوع موازی سه بیت



دیگر نوع درخت سه بیت



پياه سازي توابع منطقي با ديکدر ها

مثال:

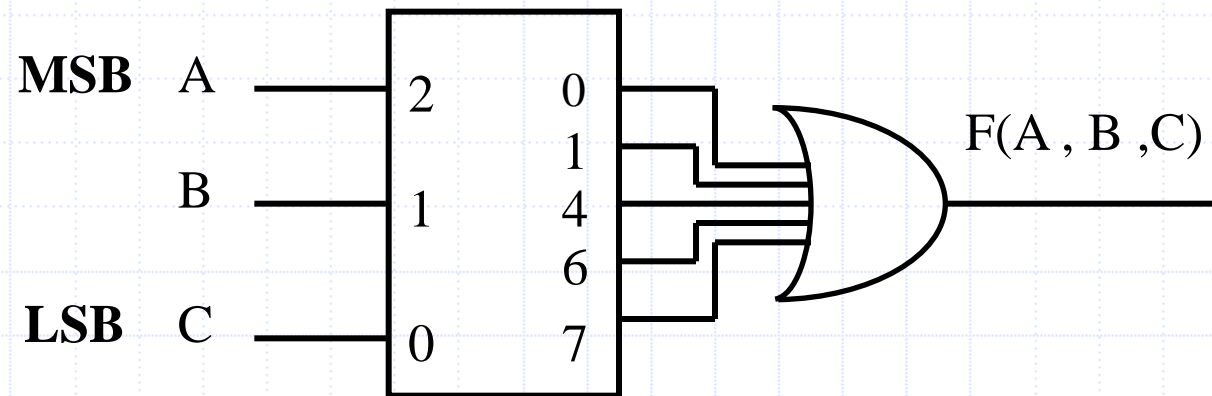
$$F(A, B, C) = \sum m(0, 1, 4, 6, 7) = \prod M(2, 3, 5)$$

تابع را به چندین طریق می توانیم پیاده نماییم:

۱. يك ديکدر (با خروجي فعال بالا) ويك گيت **OR** بکار بریم.
۲. يك ديکدر (با خروجي فعال پايين) ويك گيت **NAND** بکار بریم.
۳. يك ديکدر (با خروجي فعال بالا) ويك گيت **NOR** بکار بریم.
۴. يك ديکدر (با خروجي فعال پايين) ويك گيت **AND** بکار بریم.

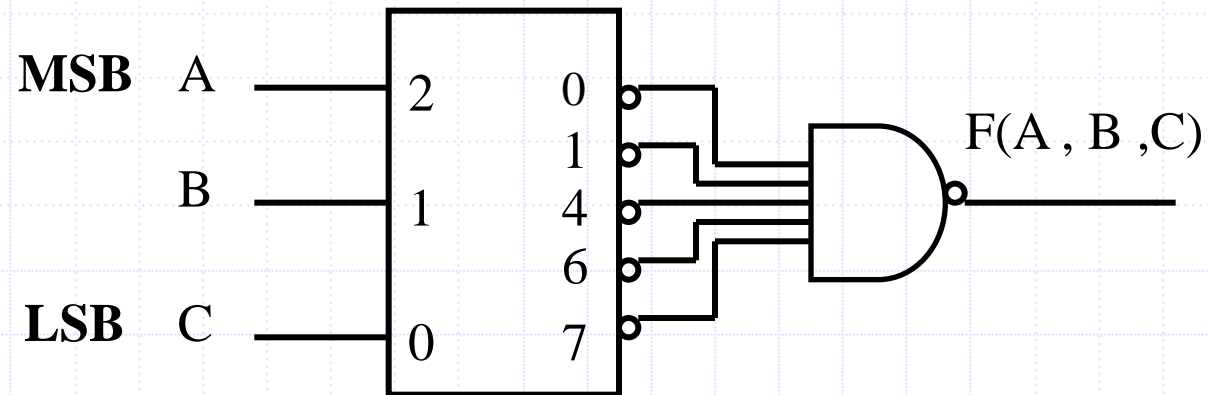
يك ديكدر (با خروجي فعال بالا) ويك گيت OR بكار بريم.

$$F(A, B, C) = m_0 + m_1 + m_4 + m_6 + m_7$$



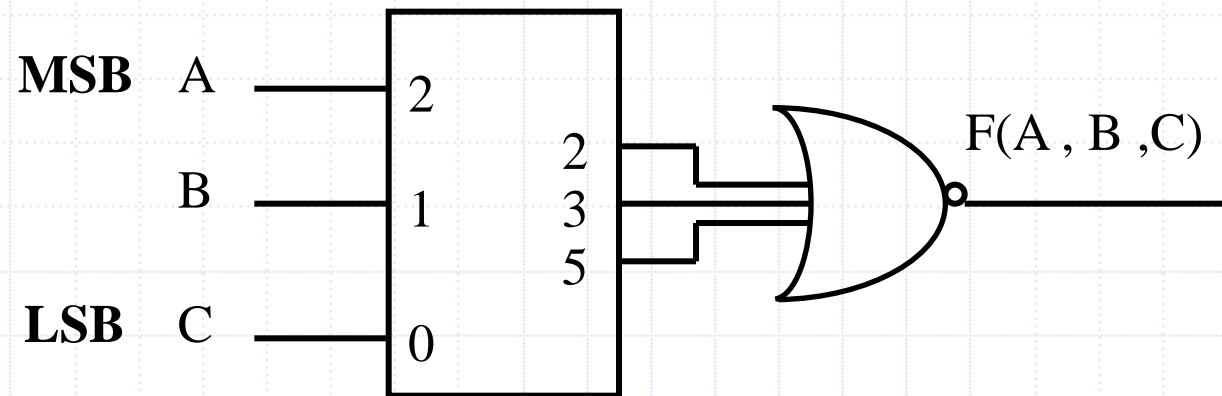
يك ديكدر (با خروجي فعال پايين) ويك گيت NAND بكار بريم.

$$F(A, B, C) = \overline{m_0 \cdot m_1 \cdot m_4 \cdot m_6 \cdot m_7}$$



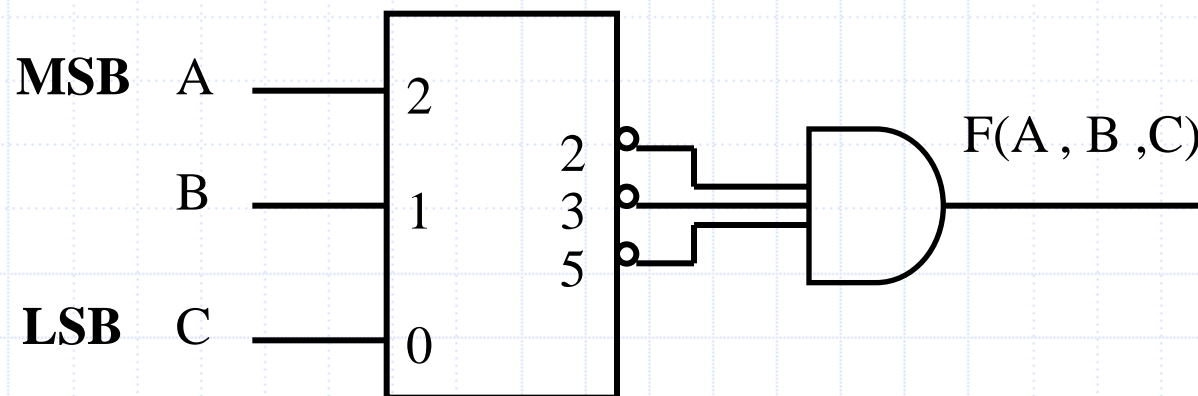
يك ديكر (با خروجي فعال بالا) ويك گيت **NOR** بكار بریم.

$$F(A, B, C) = \overline{m_2 + m_3 + m_5}$$

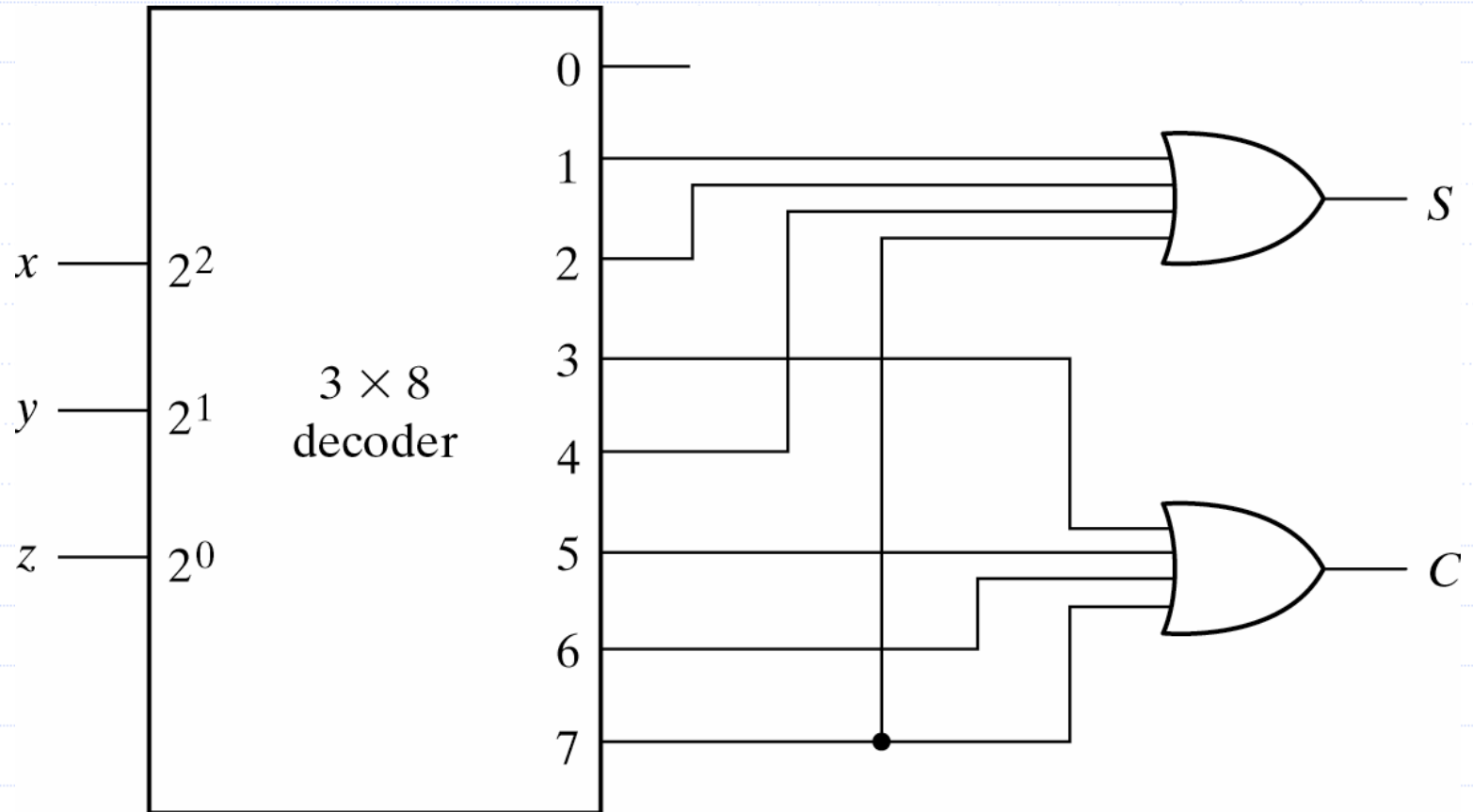


يك ديكدر (با خروجي فعال پايين) ويك گيت **AND** بكار بريم.

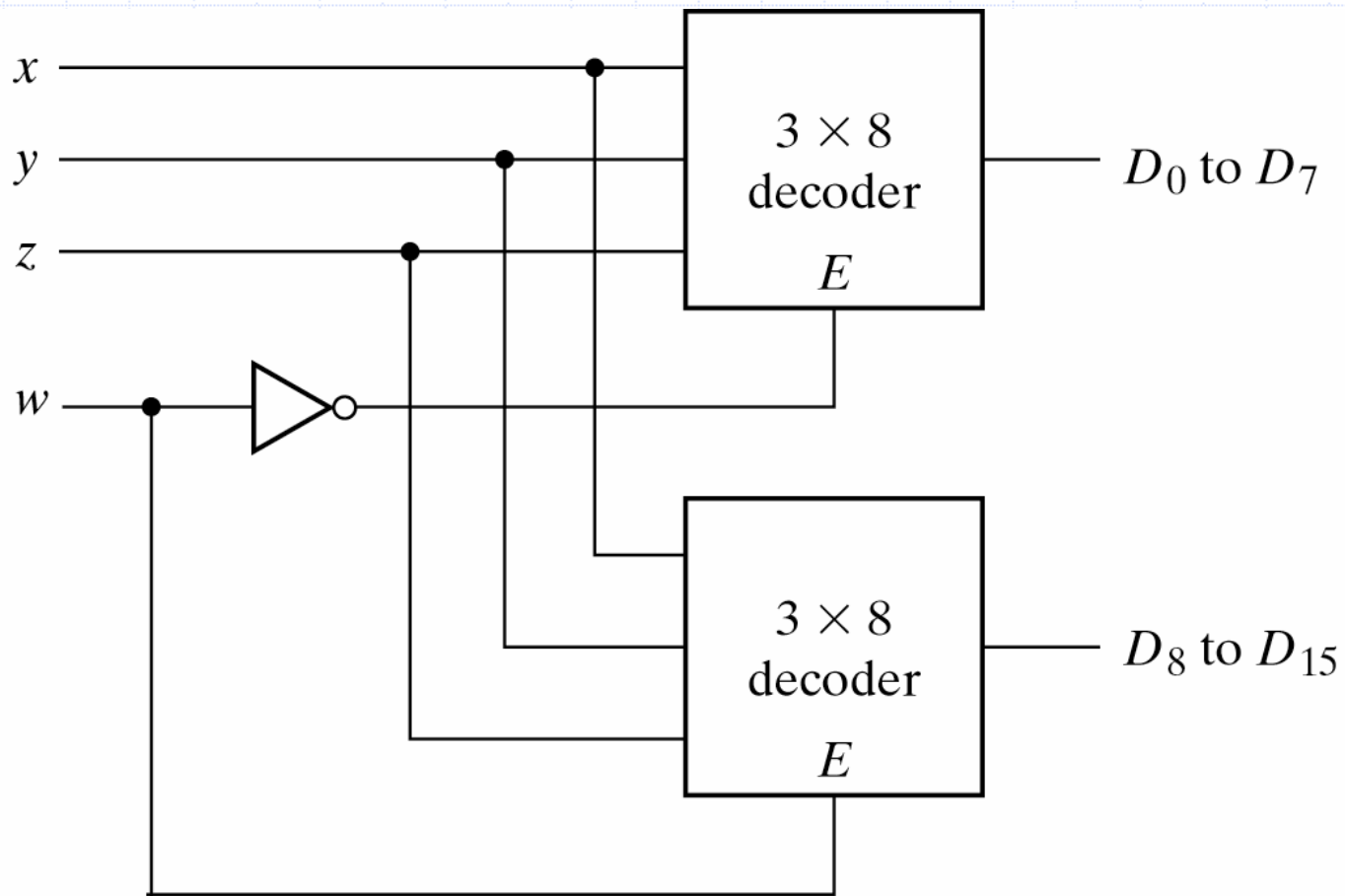
$$F(A, B, C) = \bar{m}_2 \cdot \bar{m}_3 \cdot \bar{m}_5$$



ساختن Full Adder به وسیله دیکدر:



ساختن دیکدر بزرگتر:



اینکدر

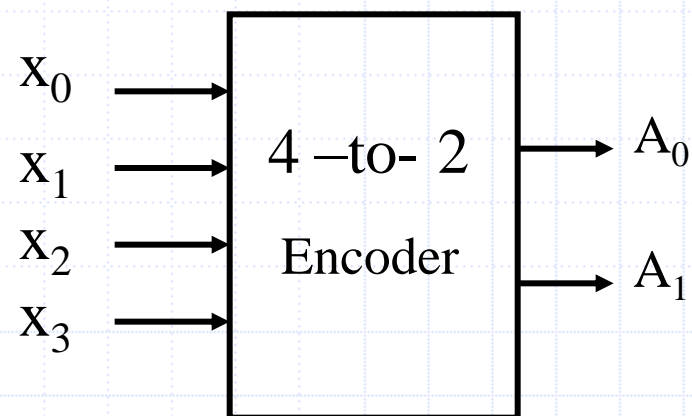
□ اینکدر يك ماجول ترکیبی است که برای هر سیگنال ورودی به دستگاه يك کد خروجی منحصر به فرد را اختصاص می دهد.

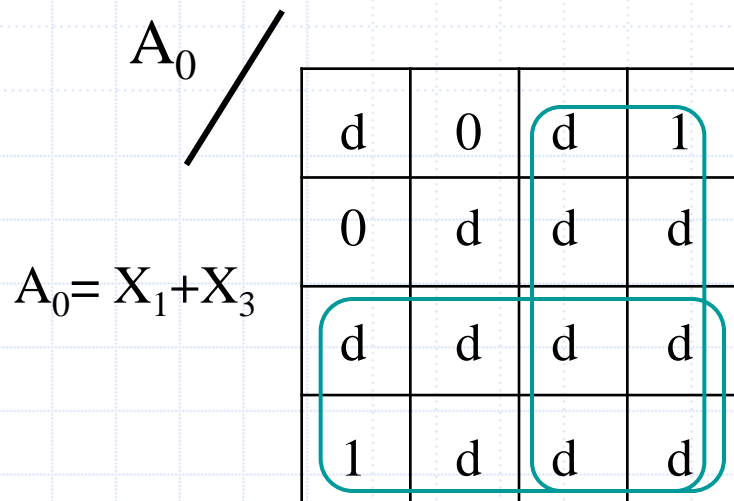
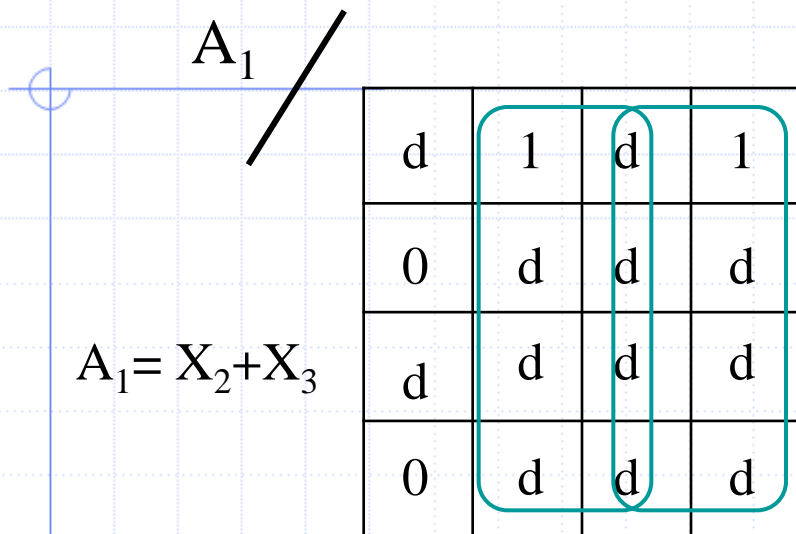
□ اگر يك ماجول اینکدر n ورودی داشته باشد خروجی S باید در رابطه زیر صدق کند:

$$\text{or} \quad 2^s \geq n$$
$$s \geq \log_2 n$$

مثال:

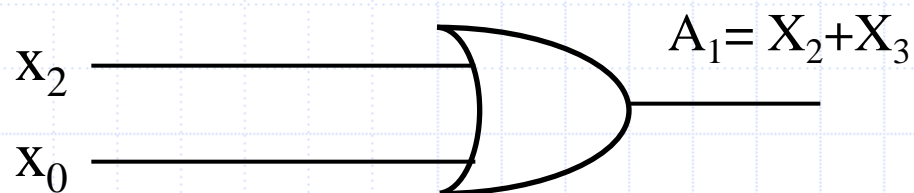
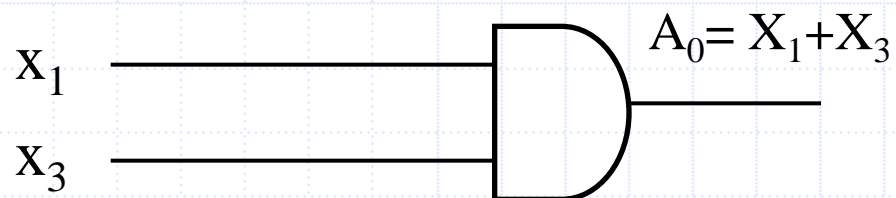
يك اينكدر براي براي چهار خط ورودی طراحی کنید بشرطی که در هر لحظه از زمان فقط يك ورودی فعال باشد.





x_3	x_2	x_1	x_0	A_1	A_0
0	0	0	0	d	d
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1	d	d
0	1	0	0	1	0
0	1	0	1	d	d
0	1	1	0	d	d
0	1	1	1	d	d
1	0	0	0	1	1
1	0	0	1	d	d
1	0	1	0	d	d
1	0	1	1	d	d
1	1	0	0	d	d
1	1	0	1	d	d
1	1	1	0	d	d
1	1	1	1	d	d

دیاگرام منطقي

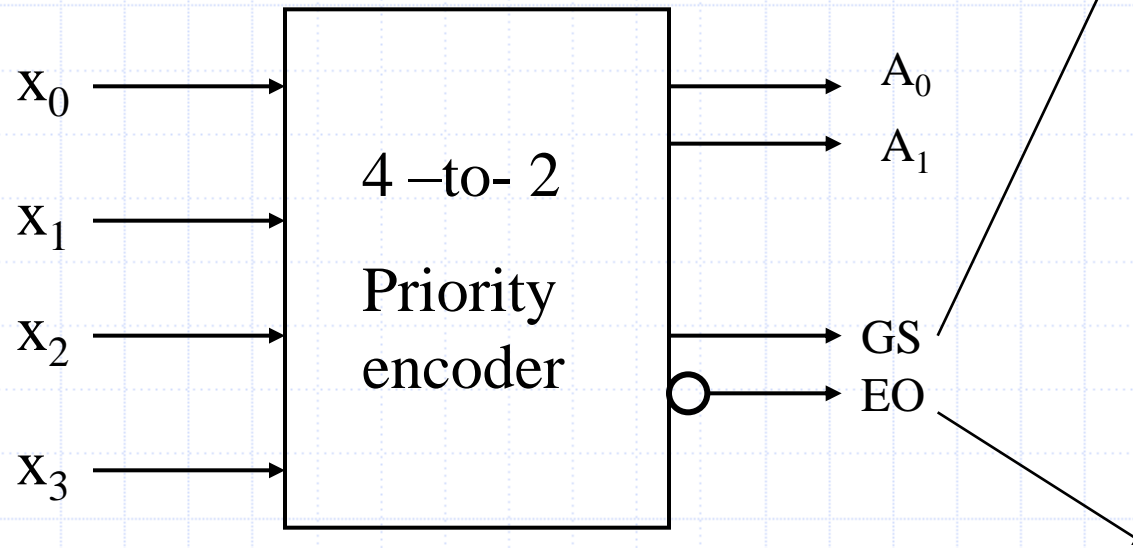


اینکدر اولویت

اینکدر اولویت اجازه می دهد تا چندین خط ورودی فعال شوند ولی عدد دودویی خارج شده از آن اندیسی است که در خطوط ورودی بالاترین اولویت را دارد.

برای ساده کردن طراحی بالاترین اولویت به بالاترین اندیس اختصاص یافته است و بالاترین اولویت بعدی به دومین اندیس بالاتر و الی آخر تخصیص داده شده است.

بلوك دیاگرام



اگر هیچ يك از خطوط ورودی فعال نباشد
 $EO=1$

اگر بیش از یکی از خطوط ورودی فعال باشد
 $GS=1$

A_1

$A_1 = X_2 + X_3$

	1	1	1
	1	1	1
	1	1	1
	1	1	1

A_0

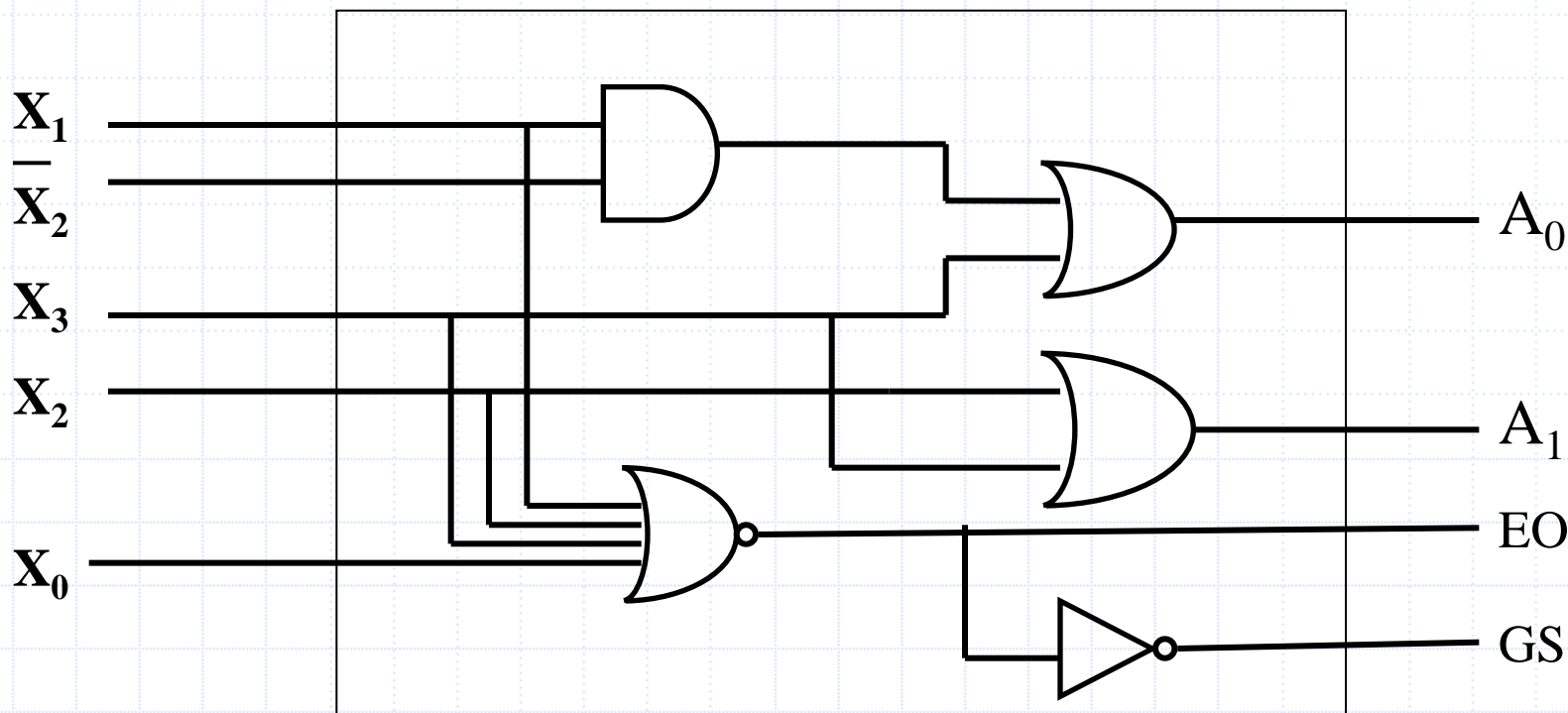
$A_0 = X_1 + X_3$

		1	1
		1	1
1		1	1
1		1	1

$EO = GS = X_0 + X_1 + X_2 + X_3$

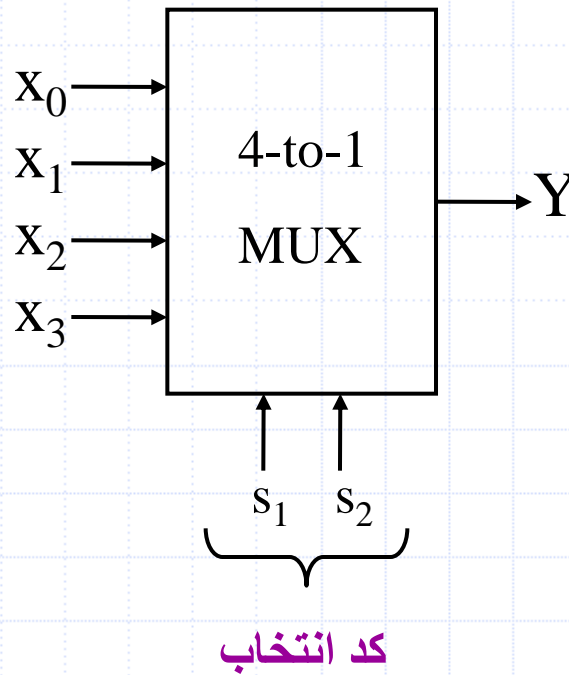
x_3	x_2	x_1	x_0	A_1	A_0	GS	EO
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	1	1	0
0	0	1	1	0	1	1	0
0	1	0	0	1	0	1	0
0	1	0	1	1	0	1	0
0	1	1	0	1	0	1	0
0	1	1	1	1	0	1	0
1	0	0	0	1	1	1	0
1	0	0	1	1	1	1	0
1	0	1	0	1	1	1	0
1	0	1	1	1	1	1	0
1	1	0	0	1	1	1	0
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	0
1	1	1	1	1	1	1	0

دیاگرام منطقي



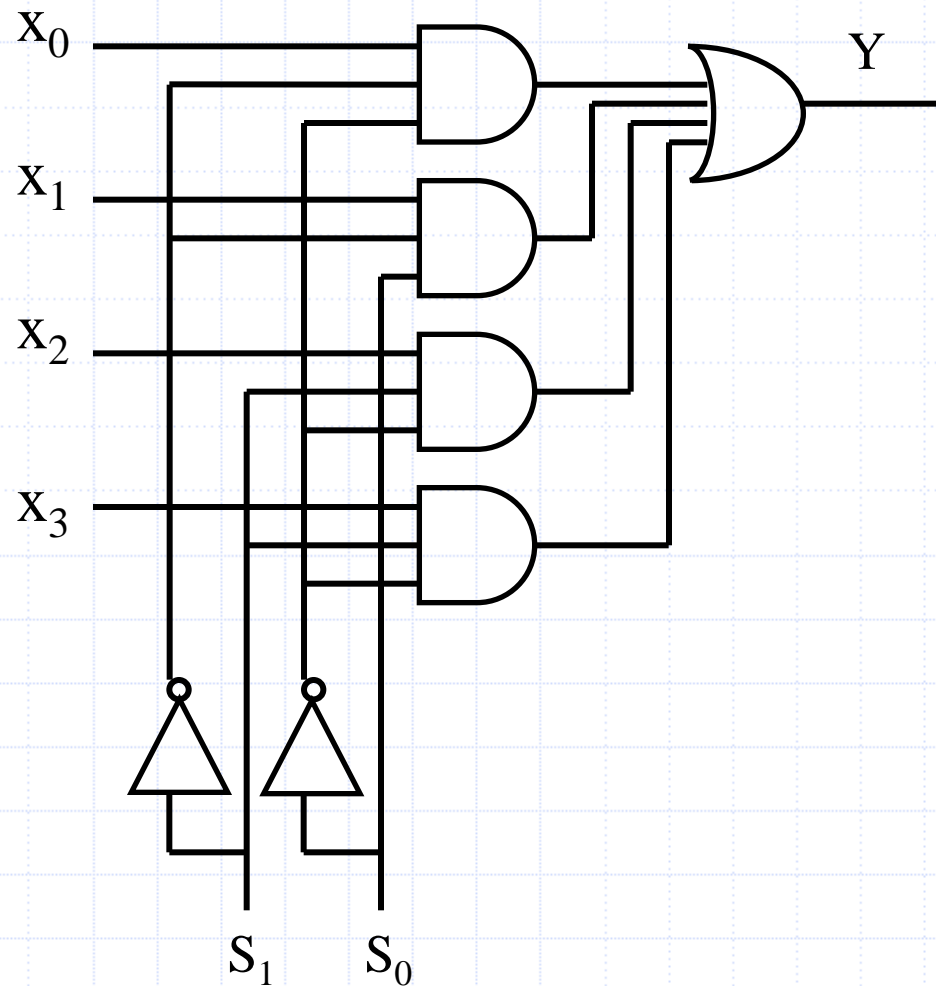
مالتی پلکسر (تسہیم کنندہ)

بطور کلی مالتی پلکسر (انتخابگر داده) یک ماجول است که یکی از چند خط ورودی را انتخاب و آن را روی خط خروجی ظاهر می سازد.

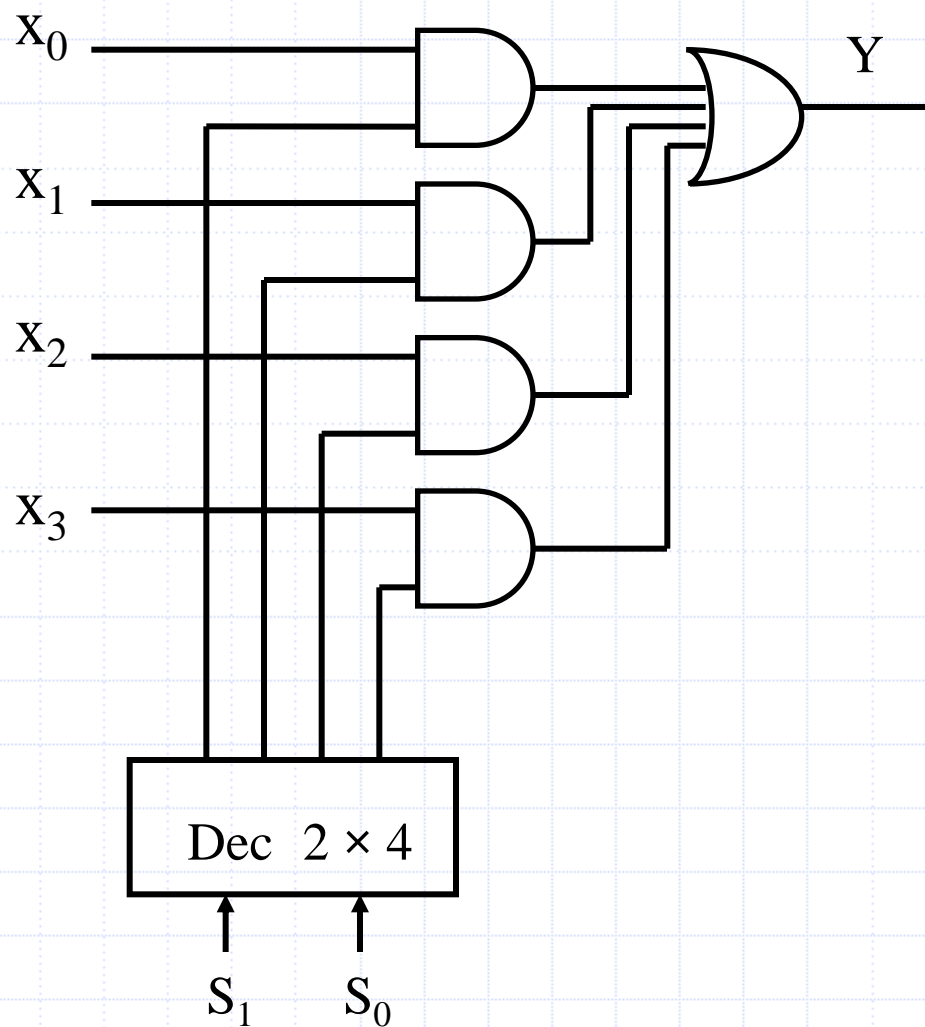


مدار معادل دو طبقه

S_1	S_0	Y
0	0	x_0
0	1	x_1
1	0	x_2
1	1	x_3



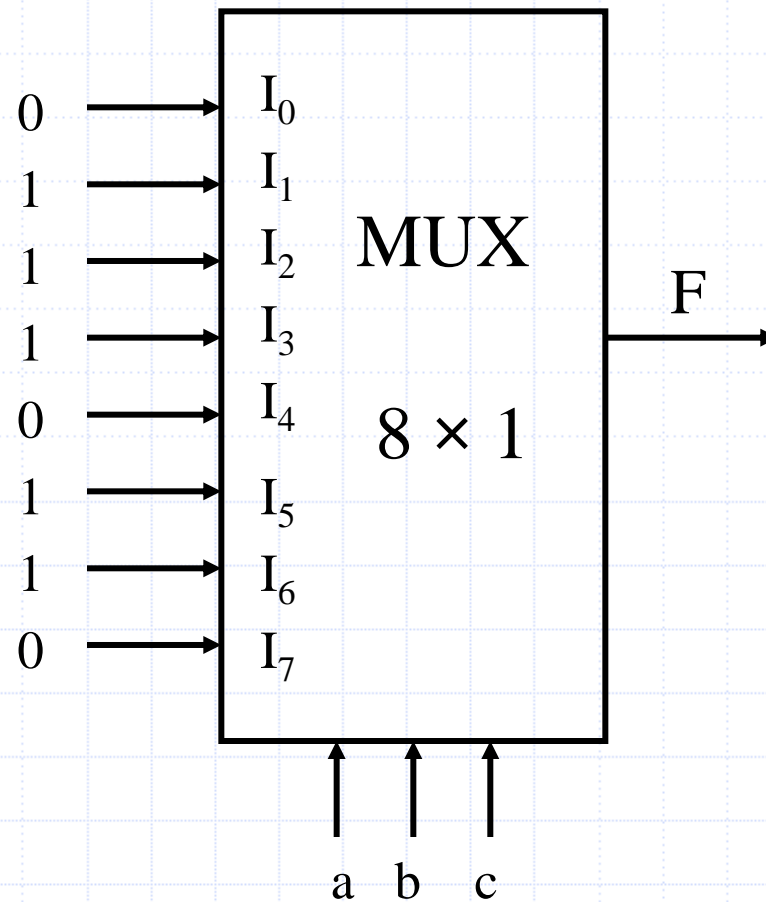
دیاگرام منطقي



مثال ١ :

$$F(A, B, C) = \sum m(1, 2, 3, 5, 6)$$

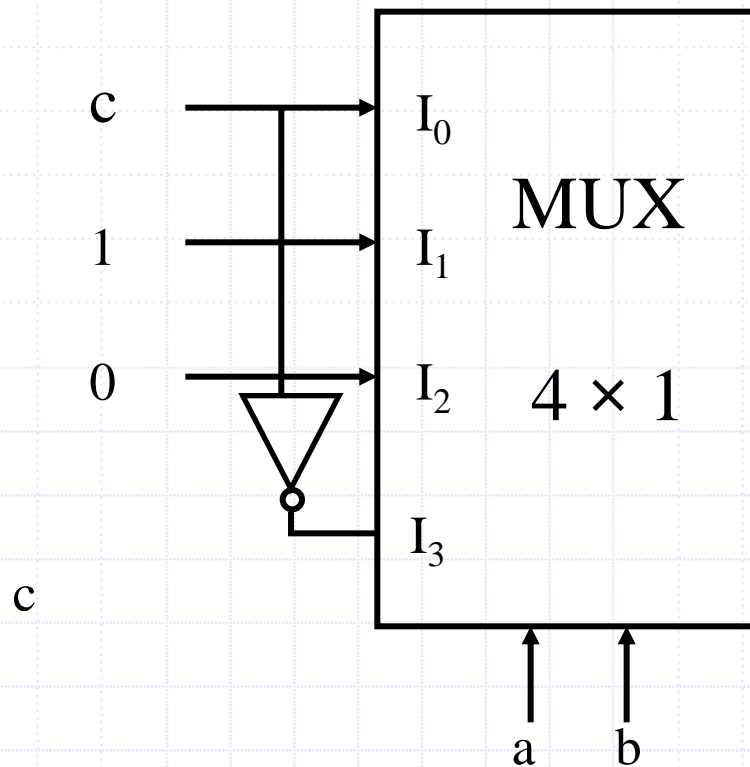
a	b	c	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0



مثال ٢ :

$$F(A, B, C) = \sum M(1, 2, 3, 6)$$

	a	b	c	F
I_0	0	0	0	0
	0	0	1	1
I_1	0	1	0	1
	0	1	1	1
I_2	1	0	0	0
	1	0	1	0
I_3	1	1	0	1
	1	1	1	0



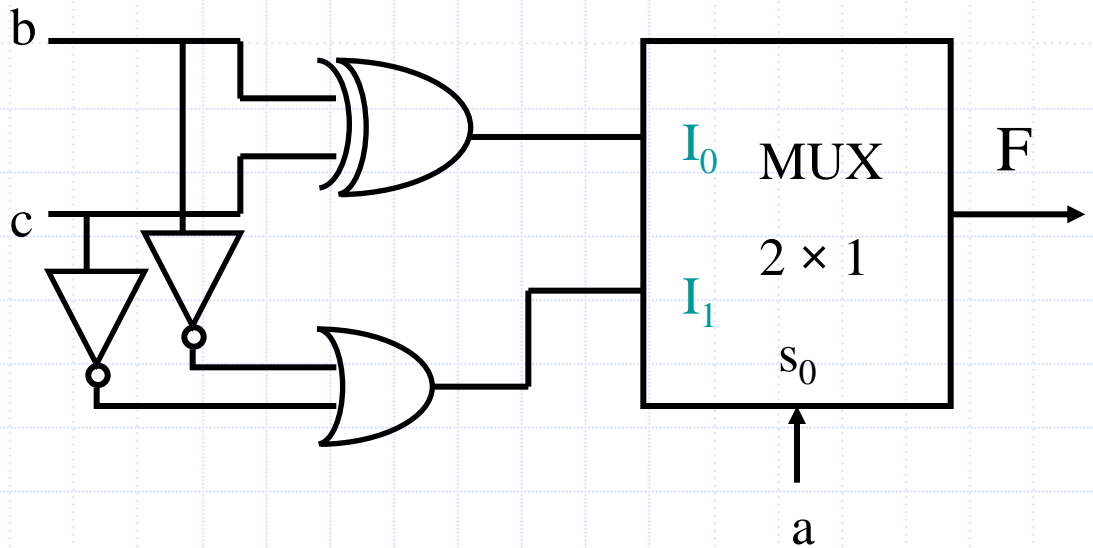
مثال ٣ :

$$F(A, B, C) = \sum m(1, 2, 4, 5, 6)$$

	a	b	c	F
I_0	0	0	0	0
	0	0	1	1
	0	1	0	1
	0	1	1	0
I_1	1	0	0	1
	1	0	1	1
	1	1	0	1
	1	1	1	0

$$\begin{cases} I_0 = b \oplus c \\ I_1 = \bar{b} + c = \bar{b}c \end{cases}$$

b c	00	01	11	10
a 0		1		1
1	1	1		1

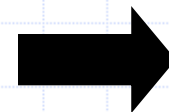


مثال ٢ :

$$F(A, B, C, D) = \sum m(1, 3, 5, 6, 7, 10, 11, 15)$$

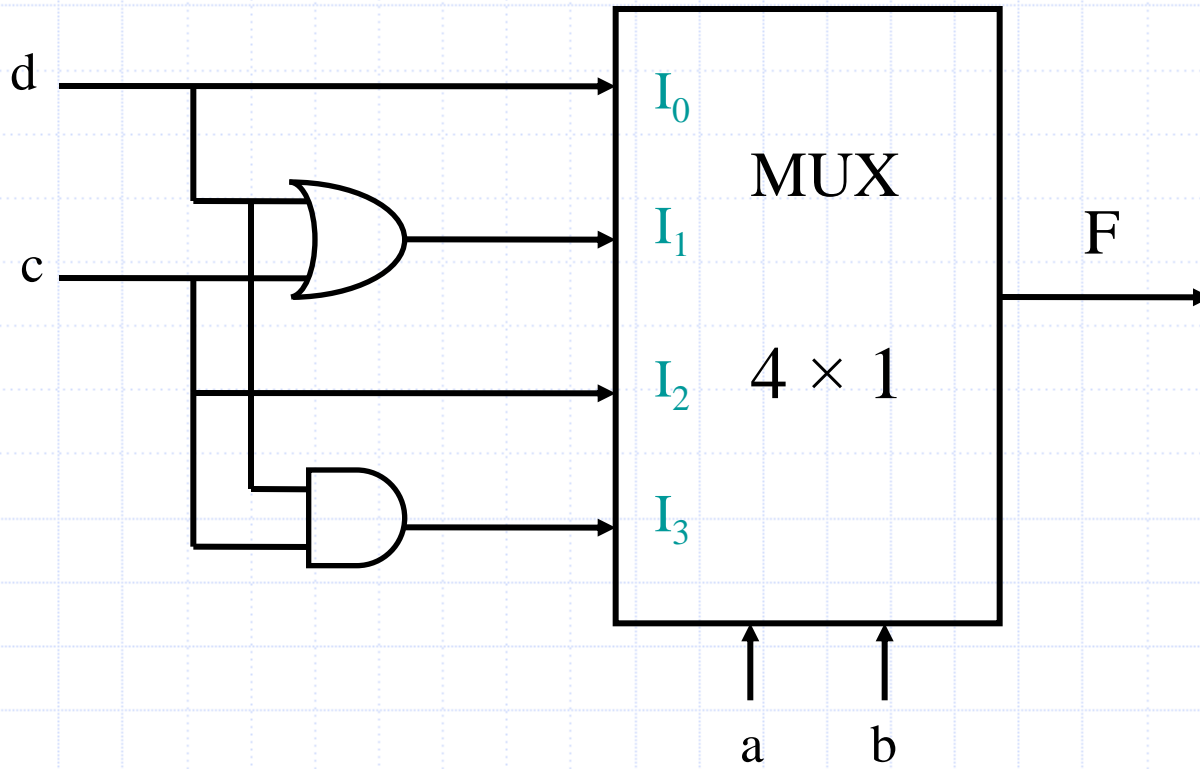
	a	b	c	d	F
I_0	0	0	0	0	0
	0	0	0	1	1
	0	0	1	0	0
	0	0	1	1	1
I_1	0	1	0	0	1
	0	1	0	1	1
	0	1	1	0	1
	0	1	1	1	1
I_2	1	0	0	0	0
	1	0	0	1	0
	1	0	1	0	1
	1	0	1	1	1
I_3	1	1	0	0	0
	1	1	0	1	0
	1	1	1	0	0
	1	1	1	1	1

	c d	00	01	11	10
a b	00		1	1	
	01		1	1	1
	11			1	
	10			1	1



$$\left\{ \begin{array}{l} I_0 = d \\ I_1 = d + c \\ I_2 = C \\ I_3 = cd \end{array} \right.$$

مثال ٤ :



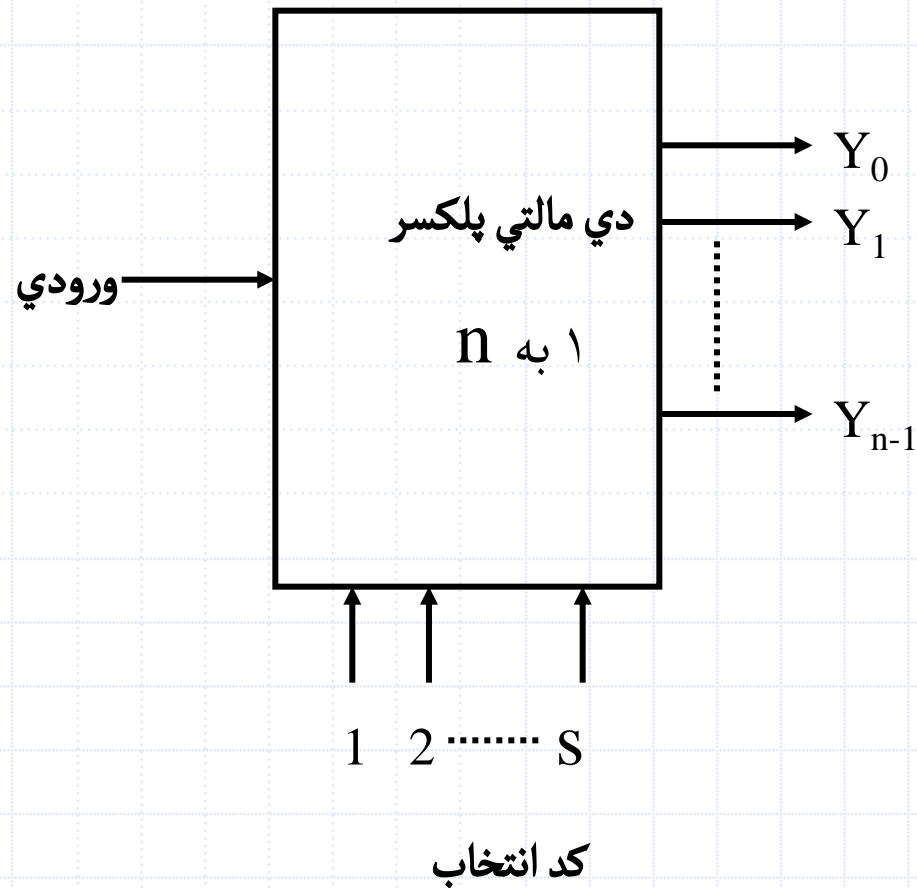
دي مالتی پلکسر (پخش کننده داده ورودی)

یک مدار منطقی ترکیبی که خط را به یک خط ورودی را به یکی از n خط خروجی وصل می کند
خط خروجی خاص با یک کد انتخاب S بیتی معین می شود که:

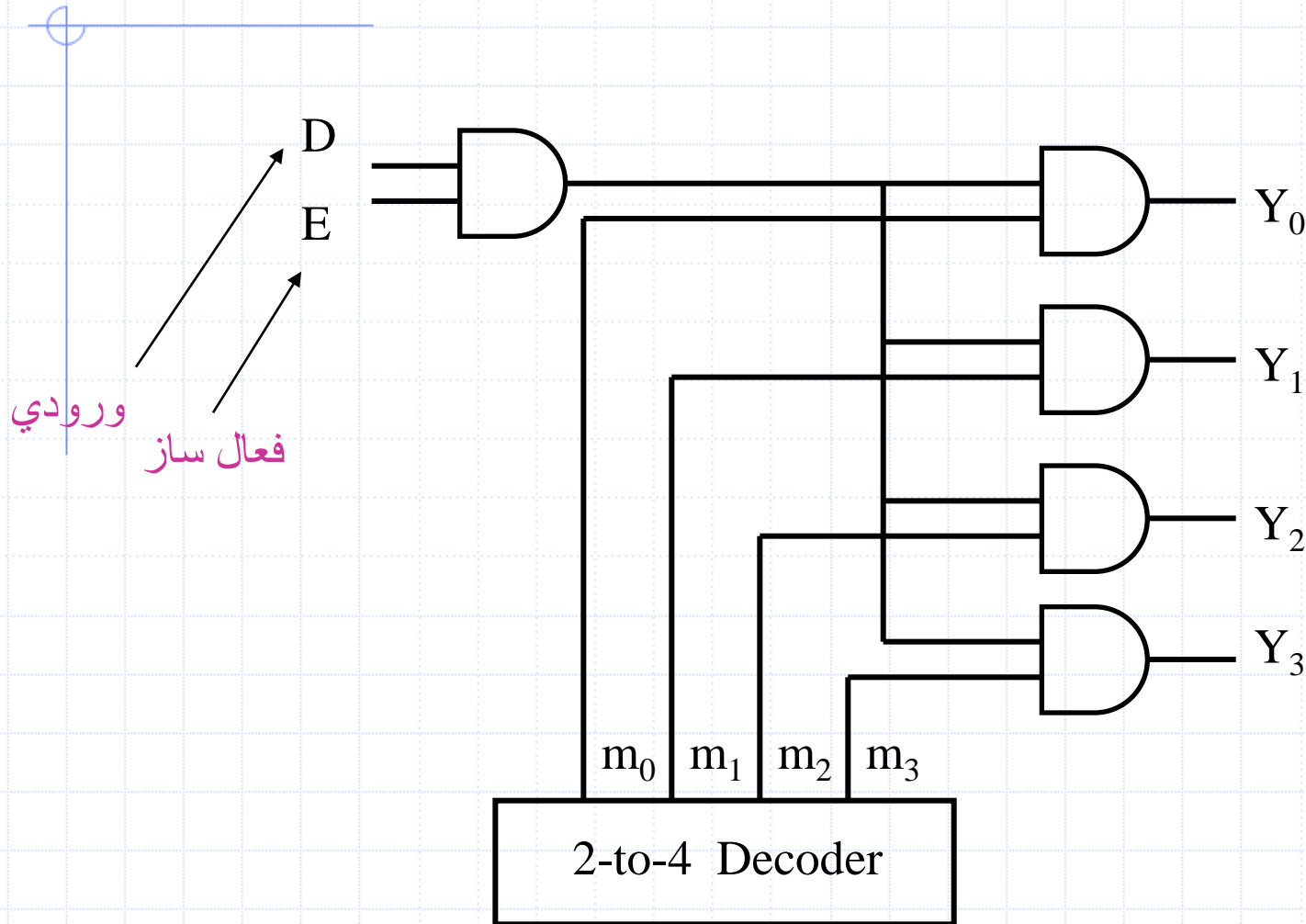
$$2^S \geq n$$

در این حالت کد انتخاب برای تولید مینترم های S بکار می رود.

دیاگرام عملیاتی



دي مالتی پلکسر ۱ به ۴ با فعال ساز



مقایسه گر ها

□ مقایسه گر قطعه ای محاسباتی است که اندازه نسبی دو عدد دودویی را تعیین می کند.

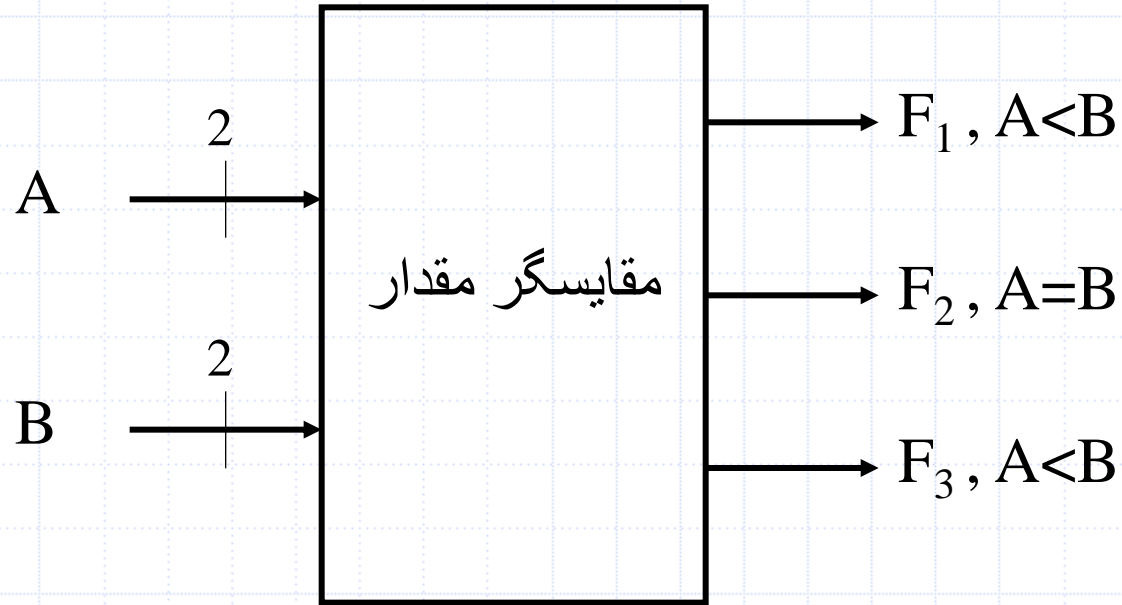
□ در یک مقایسه گر سه تصمیم کاملاً دیکد شده در مورد دو کلمه انجام و در خروجی ها قرار

می گیرند. یعنی $A > B$, $A < B$, $A = B$ اگر

$$A = (A_{n-1} A_{n-2} \dots A_0)$$

$$B = (B_{n-1} B_{n-2} \dots B_0)$$

دیاگرام عملیاتی



$$\begin{cases} F_1 = 1, \text{ If } A < B \\ F_2 = 1, \text{ If } A = B \\ F_3 = 1, \text{ If } A > B \end{cases}$$

مثال :

مقایسه گری طراحی کنید که دو کلمه

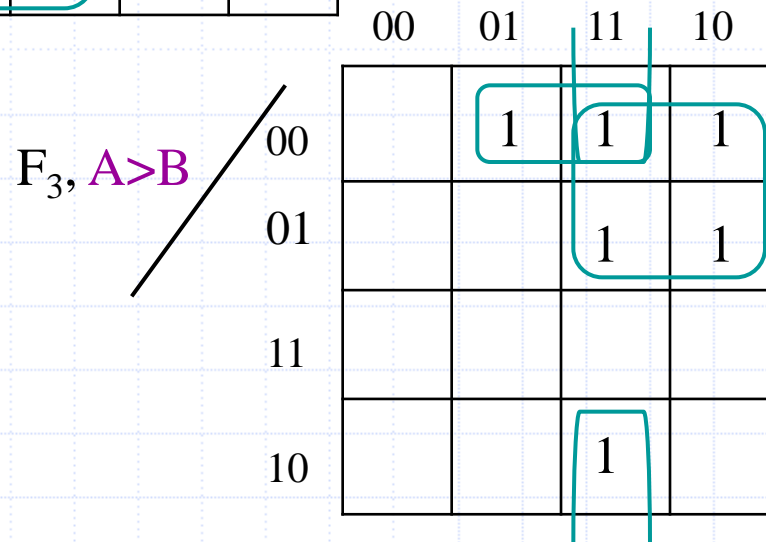
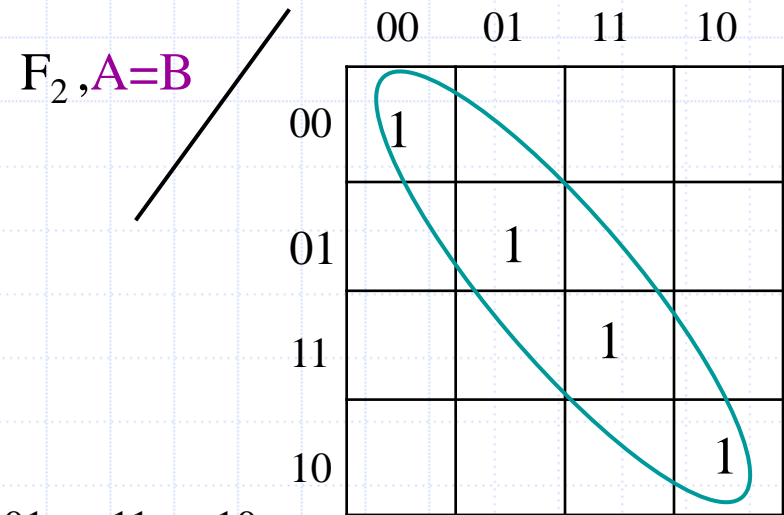
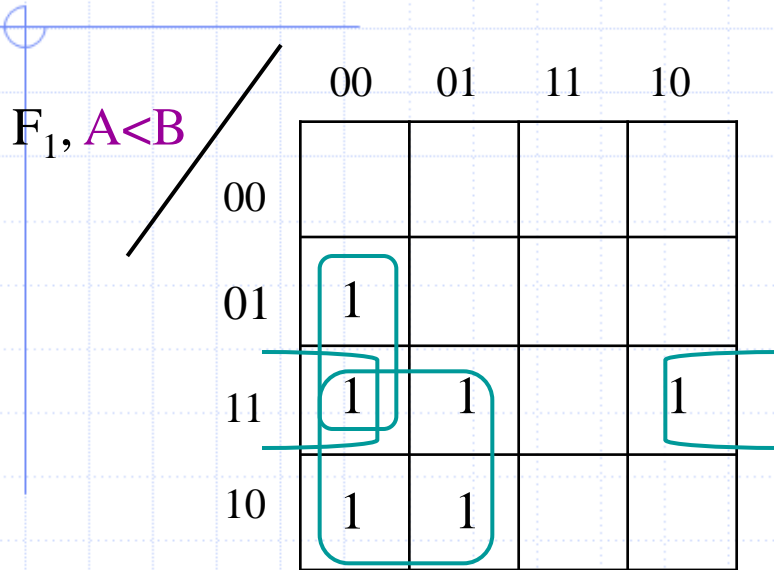
$$A = (A_1 A_0)_2$$

$$B = (B_1 B_0)_2$$

در کد دودویی مقایسه کند.

A_1	A_2	B_2	B_2	F_1	F_2	F_3
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	1

نقشه هاي كارنو



توابع خروجي

$$F_1 = \overline{A_1} B_1 + \overline{A_1} \overline{A_0} B_0 + \overline{A_0} B_1 B_0$$

For $(A_1 A_0)_2 < (B_1 B_0)_2$

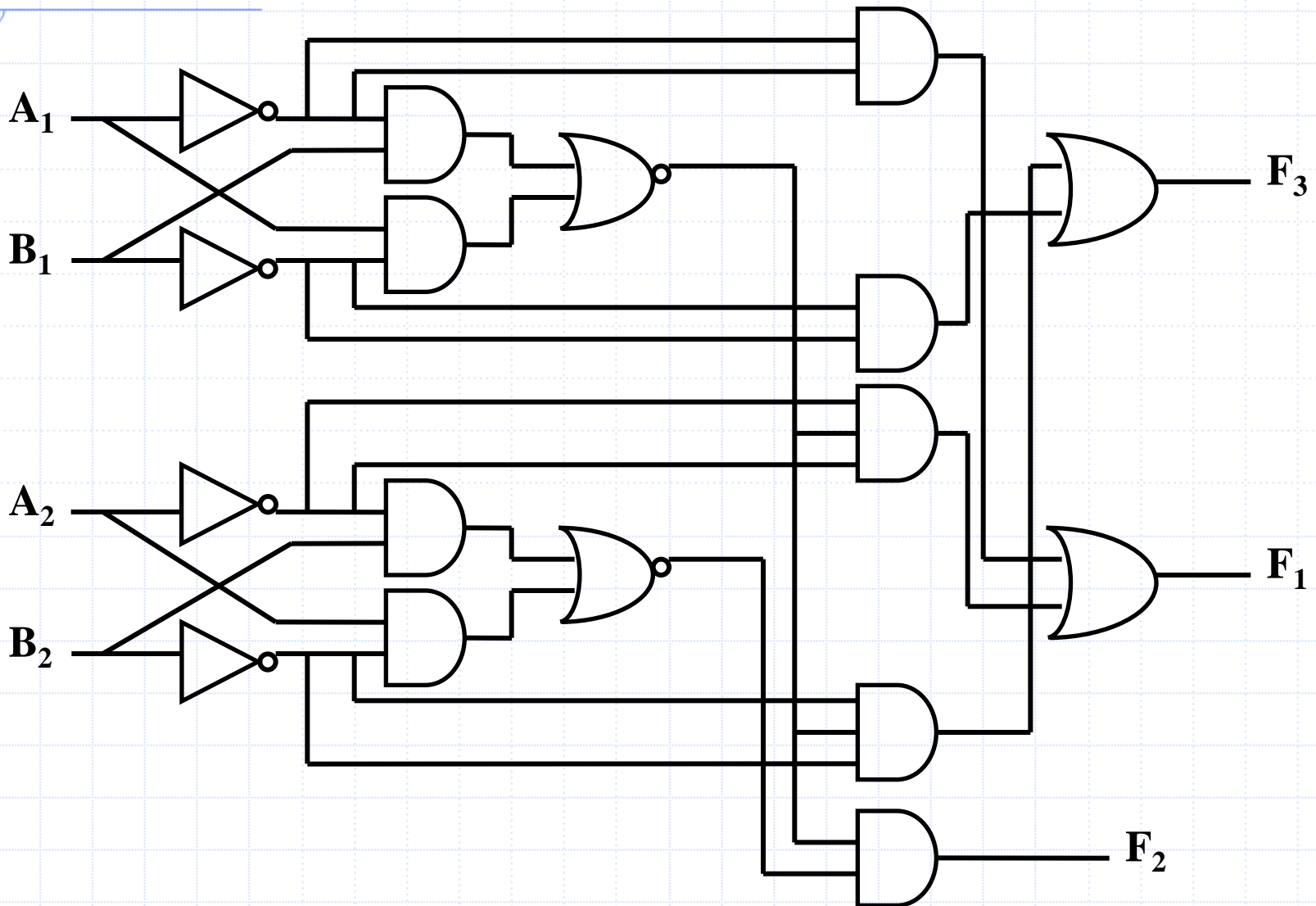
$$F_2 = \overline{A_1} \overline{A_0} \overline{B_1} B_0 + \overline{A_1} \overline{A_0} \overline{B_1} \overline{B_0} + \overline{A_1} \overline{A_0} B_1 \overline{B_0} + \overline{A_1} \overline{A_0} B_1 B_0$$

For $(A_1 A_0)_2 = (B_1 B_0)_2$

$$F_3 = \overline{A_1} B_1 + \overline{A_1} \overline{B_1} \overline{B_0} + \overline{A_1} A_0 \overline{B_0}$$

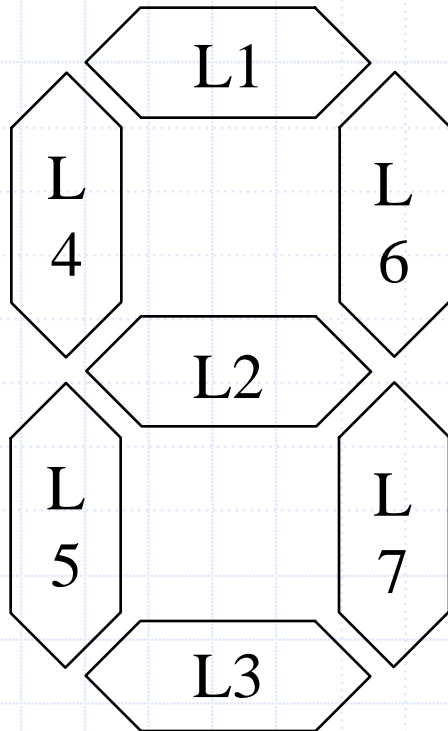
For $(A_1 A_0)_2 > (B_1 B_0)_2$

تحقيق منطقي يك مقايسه گر دو بيت

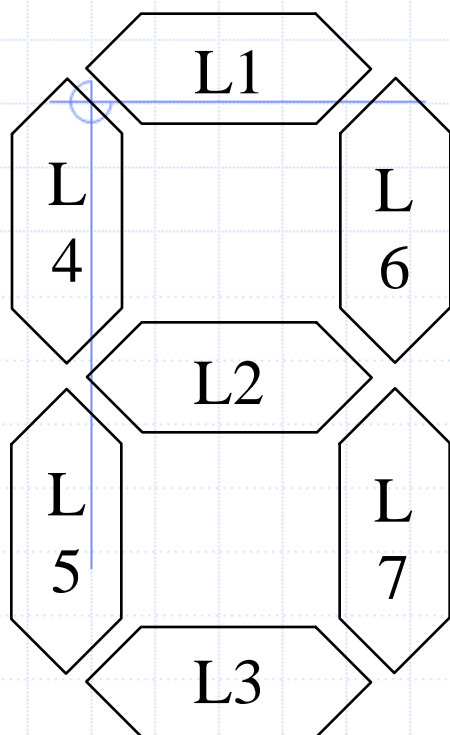


Seven Segment Display

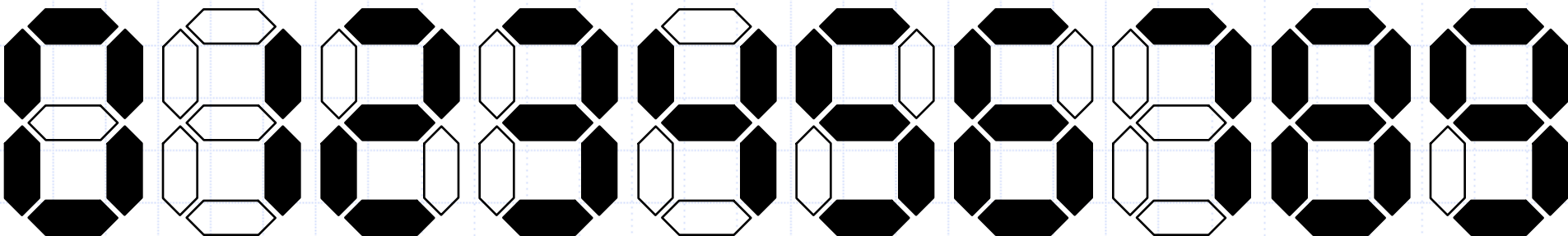
مثال :



B3	B2	B1	B0	Val
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

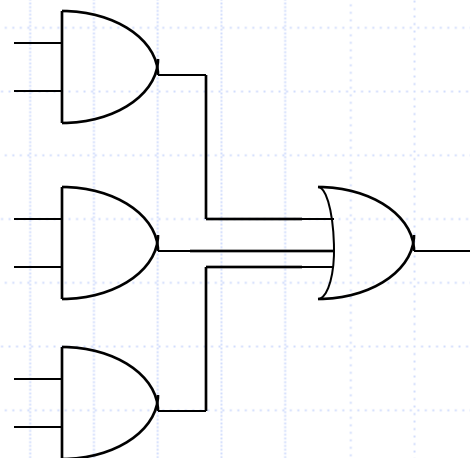


B3	B2	B1	B0	Val	L1	L2	L3	L4	L5	L6	L7
0	0	0	0	0	1	0	1	1	1	1	1
0	0	0	1	1	0	0	0	0	0	1	1
0	0	1	0	2	1	1	1	0	1	1	0
0	0	1	1	3	1	1	1	0	0	1	1
0	1	0	0	4	0	1	0	1	0	1	1
0	1	0	1	5	1	1	1	1	0	0	1
0	1	1	0	6	1	1	1	1	1	0	1
0	1	1	1	7	1	0	0	0	0	1	1
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1



□ المينت L4:

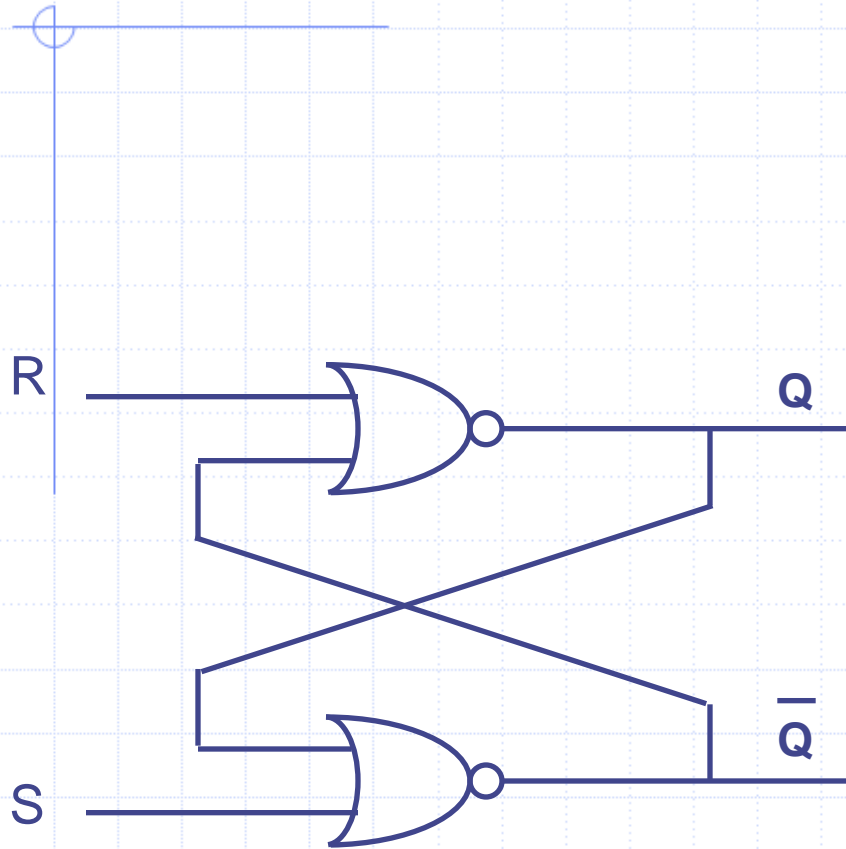
B3	B2	B1	B0	L4
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1



فصل ششم:

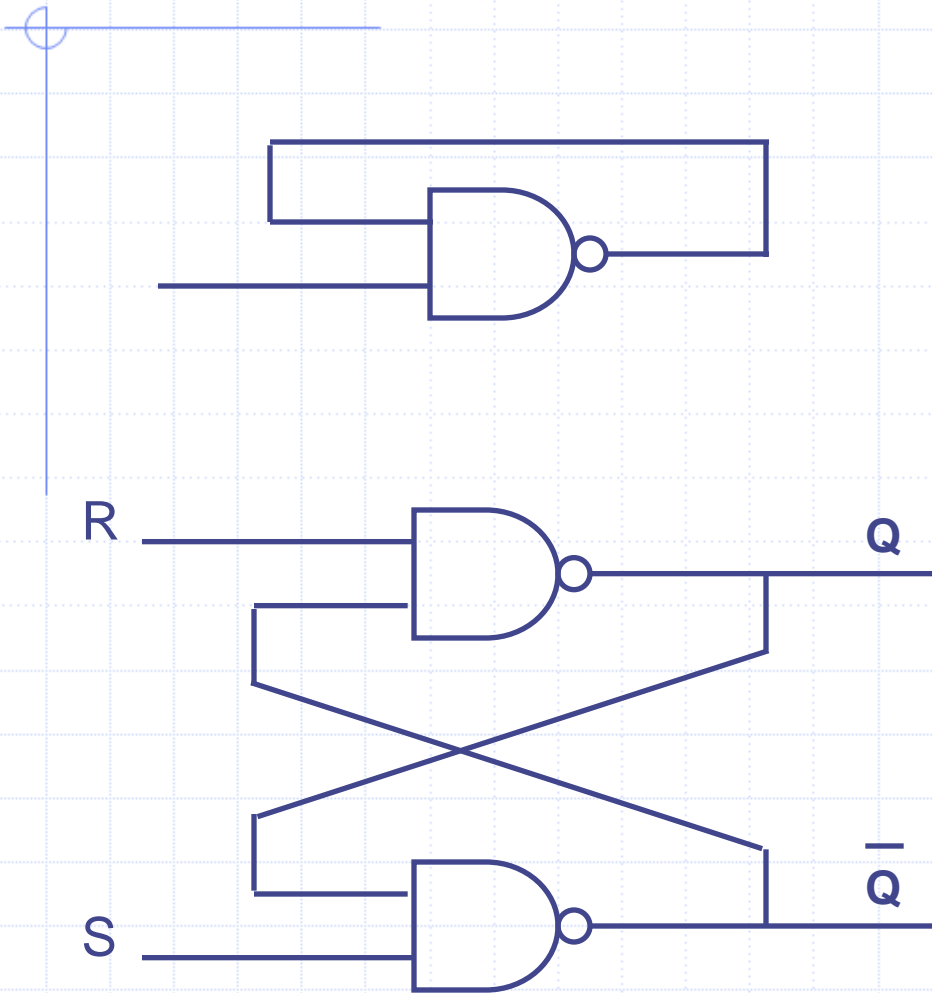
مدارات ترتیبی

مفهوم Latch :



R	S	$Q(t+1)$	$\bar{Q}(t+1)$
0	1	1	0
1	0	0	1
0	0	$Q(t)$	$\bar{Q}(t)$
1	1	نامعین	نامعین

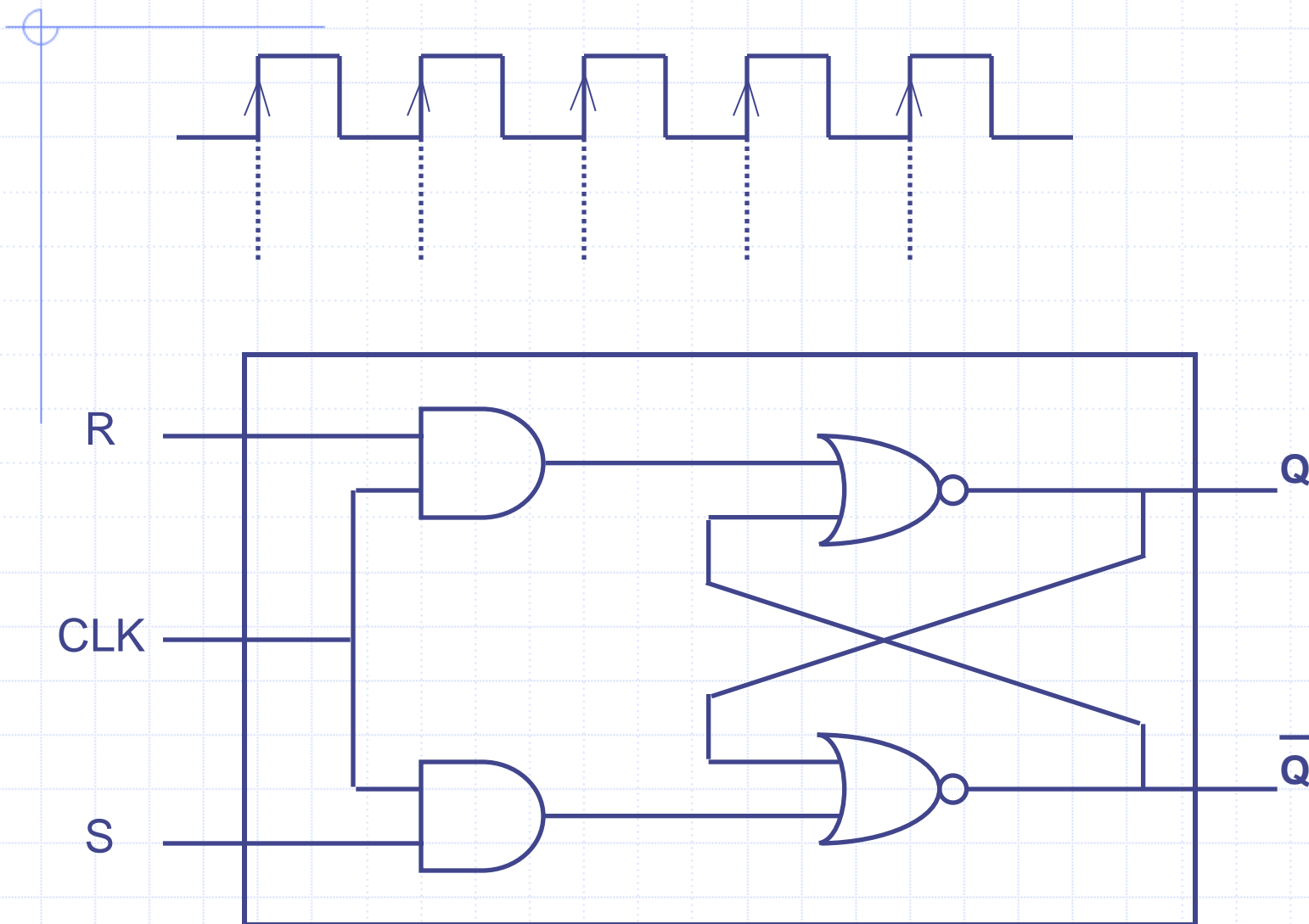
نمونه ی دیگر :



A	B	O1	O2
0	0	1	1
0	1	1	0
1	0	0	1
1	1	نامعین	نامعین

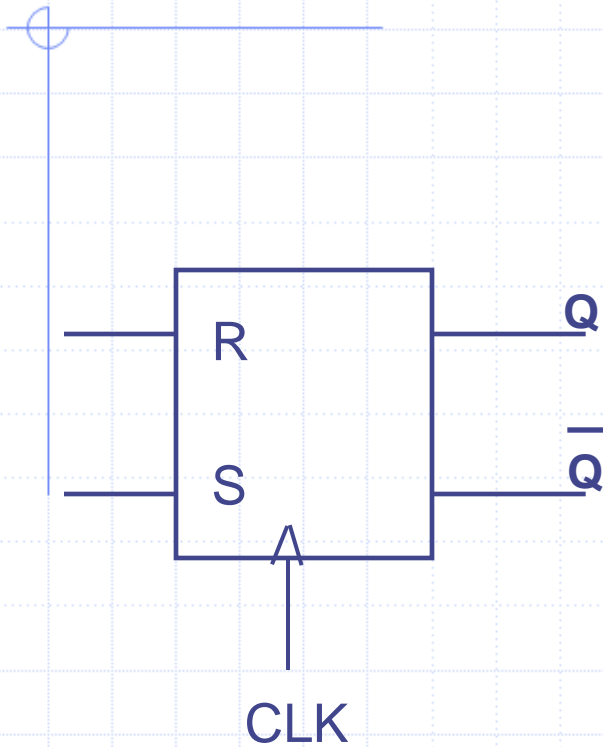
: CLK

پالس های ساعت که باعث همگام سازی مدار می شود .



انواع فلیپ فلاپ ها: RS, JK, T, D

فلیپ فلاپ RS

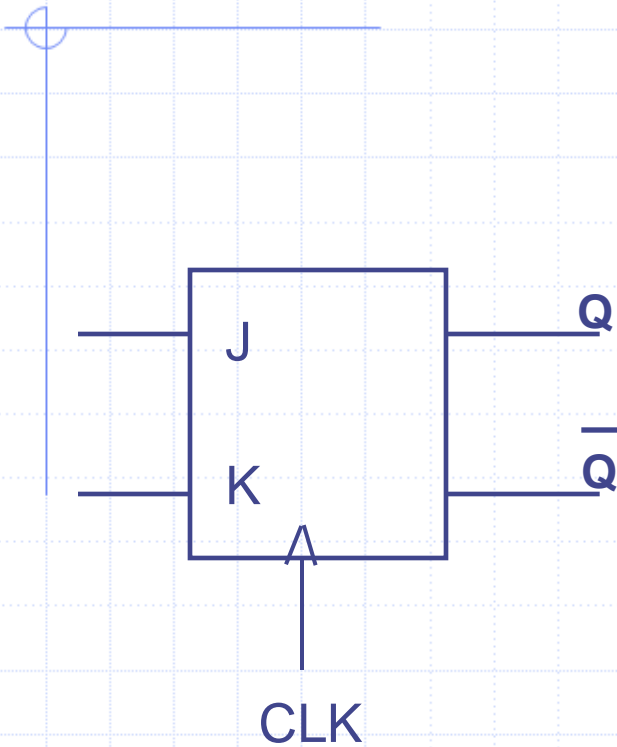


S	R	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	نامعین

(جدول مشخصه)

انواع فلیپ فلاپ ها: (ادامه)

فلیپ فلاپ JK



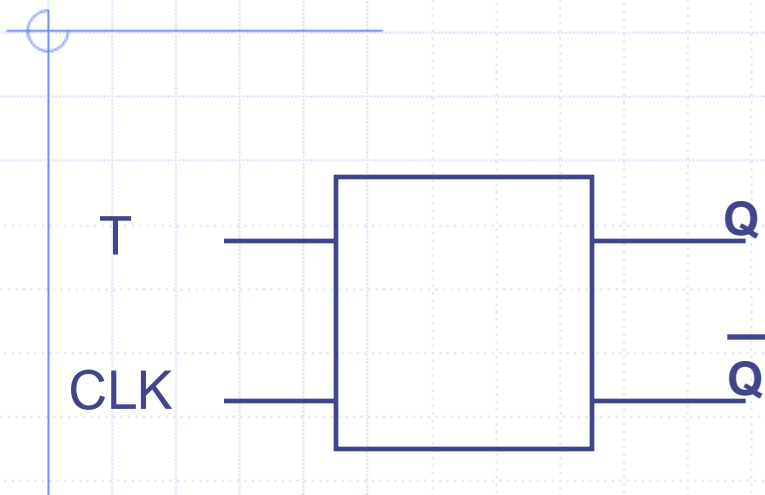
T — { D {

J	K	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	$\overline{Q(t)}$

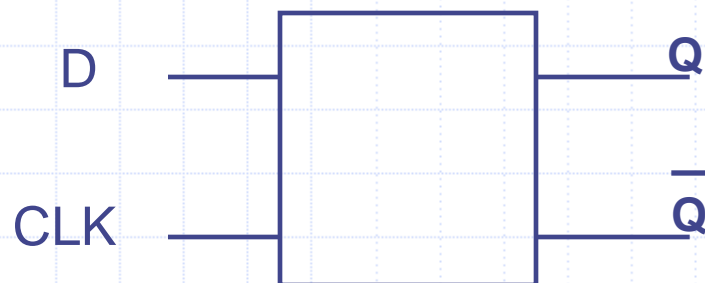
(جدول مشخصه)

انواع فلیپ فلاپ ها: (ادامه)

فلیپ فلاپ D, T



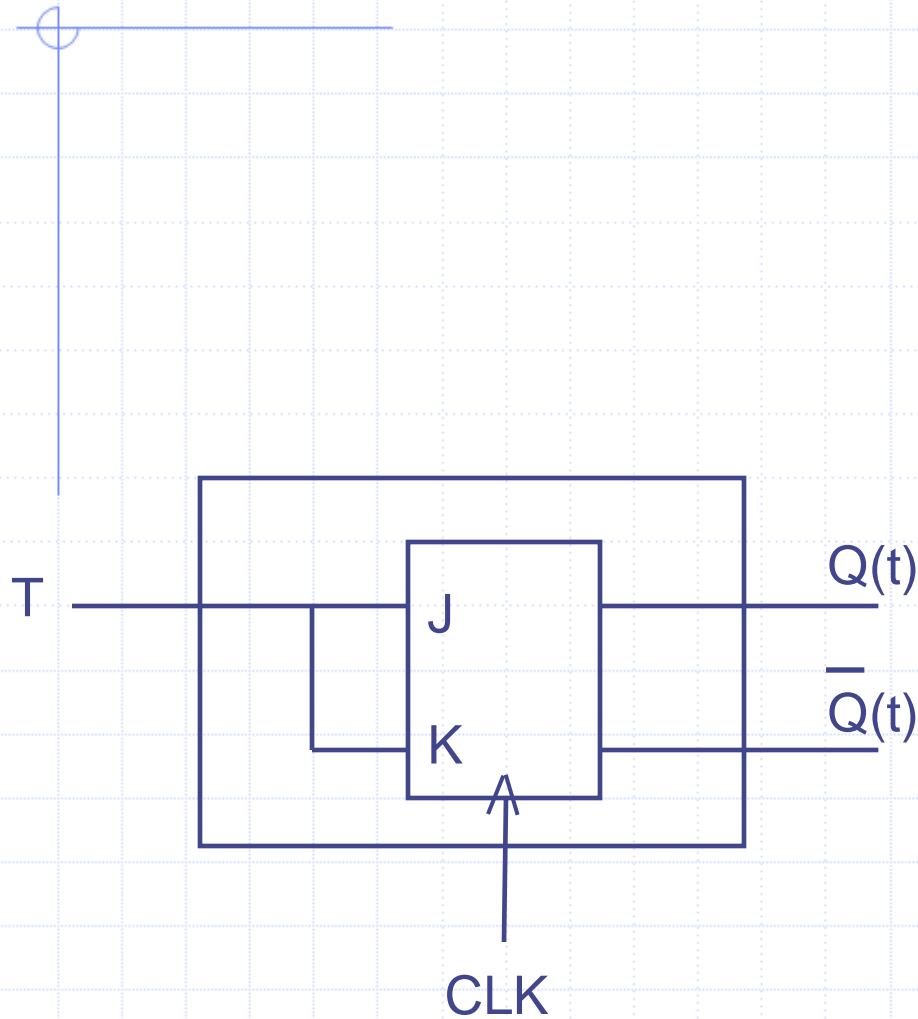
T	Q(t+1)
0	Q(t)
1	$\overline{Q(t)}$



D	Q(t+1)
0	0
1	1

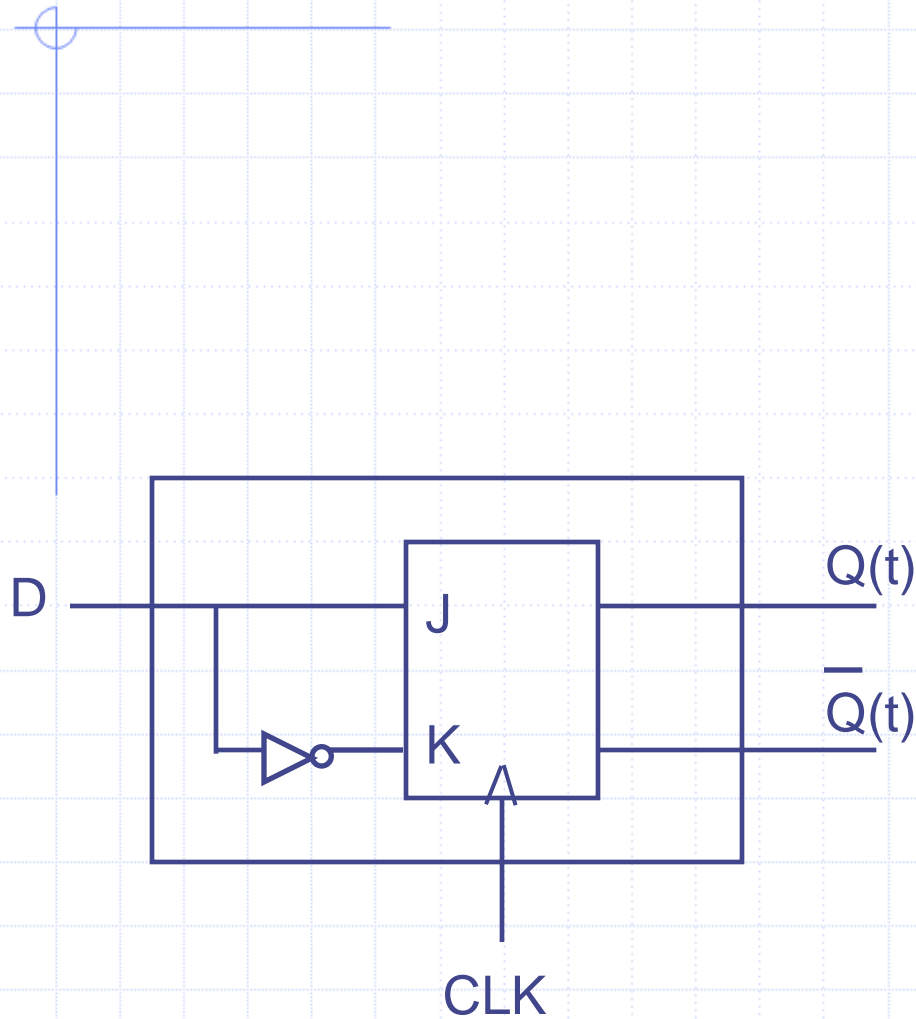
(جدول مشخصه)

مثال ۱: به کمک فلیپ فلاپ JK یک فلیپ فلاپ T بسازید.



T	J	K	Q(t+1)
0	0	0	Q(t)
1	1	1	$\overline{Q}(t)$

مثال ۲ : به کمک فلیپ فلاپ JK یک فلیپ فلاپ D بسازید.



D	J	K	Q(t+1)
0	0	1	Q(t)
1	1	0	$\overline{Q(t)}$

not

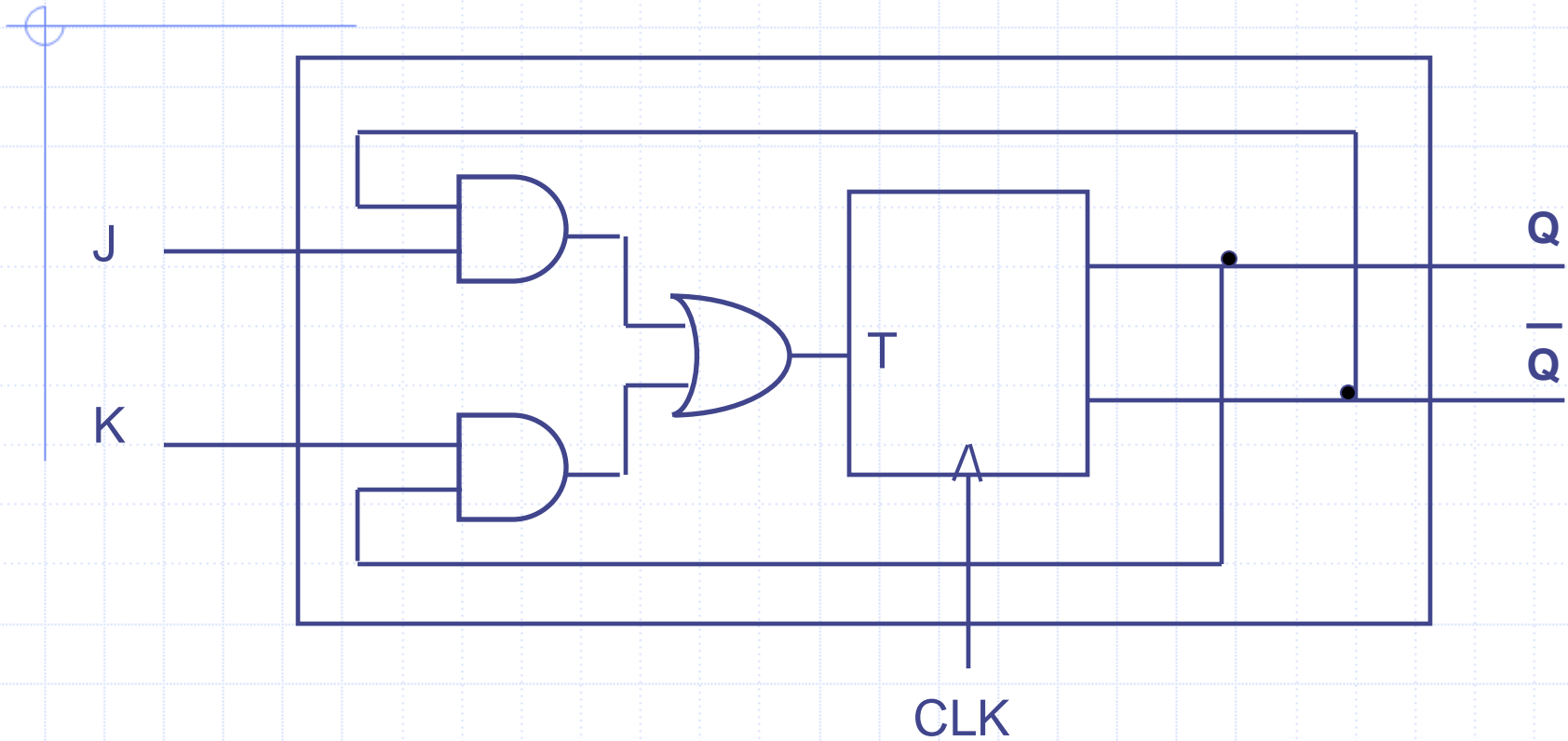
مثال ۳: به کمک فلیپ فلاپ T یک فلیپ فلاپ JK بسازید.

Q(t)	J	K	T	Q(t+1)
0	0	0	0	0
0	0	1	0	0
0	1	0	1	1
0	1	1	1	1
1	0	0	0	1
1	0	1	1	0
1	1	0	0	1
1	1	1	1	0

	JK			
	00	01	11	10
Q(t)				
0			1	1
1		1	1	

$$T = J \bar{Q}(t) + K Q(t)$$

مثال ۳: (ادامه)



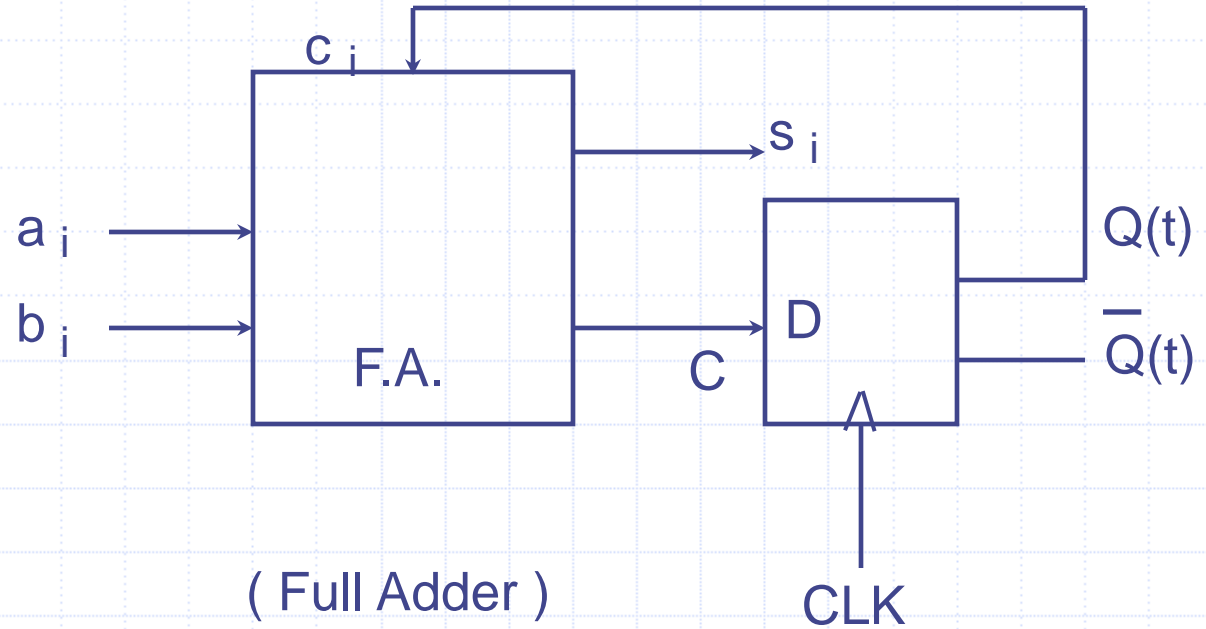
مثال ۴: به کمک فلیپ فلاپ D یک فلیپ فلاپ JK بسازید.

مثال ۵: مداری طراحی کنید که دو عدد n بیتی را با هم جمع کند، به طوریکه در هر کلاک پالس دو بیت داده شود.

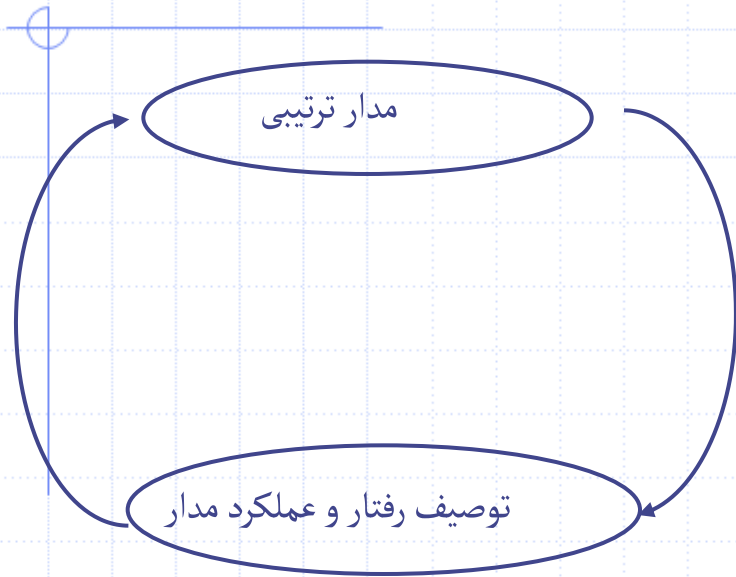
A: $a_3 \ a_2 \ a_1 \ a_0$

B: $b_3 \ b_2 \ b_1 \ b_0$

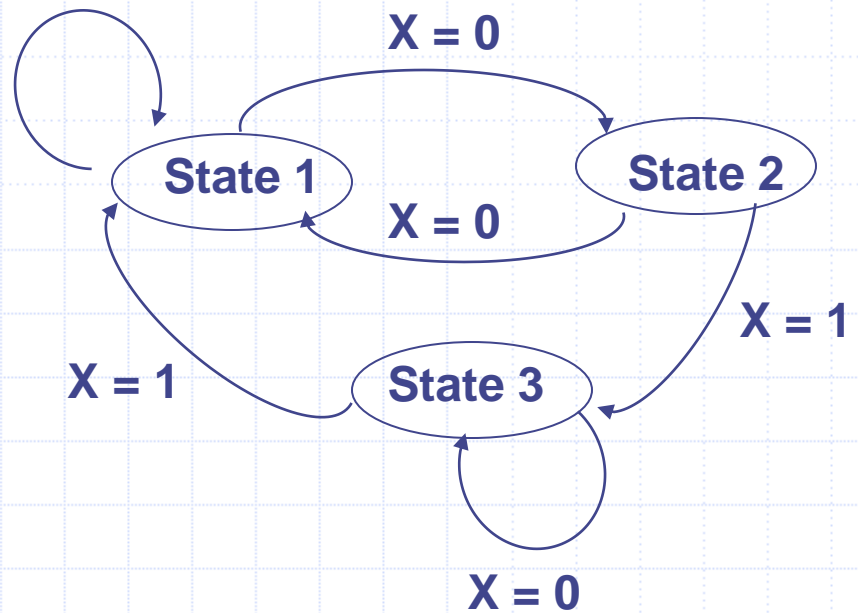
C: $s_3 \ s_2 \ s_1 \ s_0$



روند تجزیه و تحلیل مدارات ترتیبی :



$X = 1$

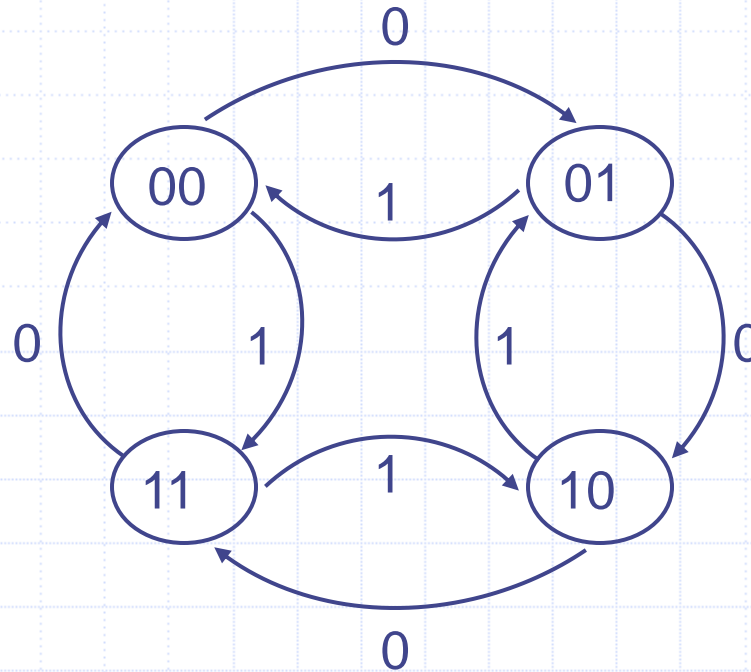


مثال ۶: یک شمارنده بالاشمار (با ورودی ۱) و پایین شمار (با ورودی ۰).

- مشخص کردن حالت ها و ترسیم آن

(تعداد فلیپ فلاپ ها)

۲ = تعداد حالات



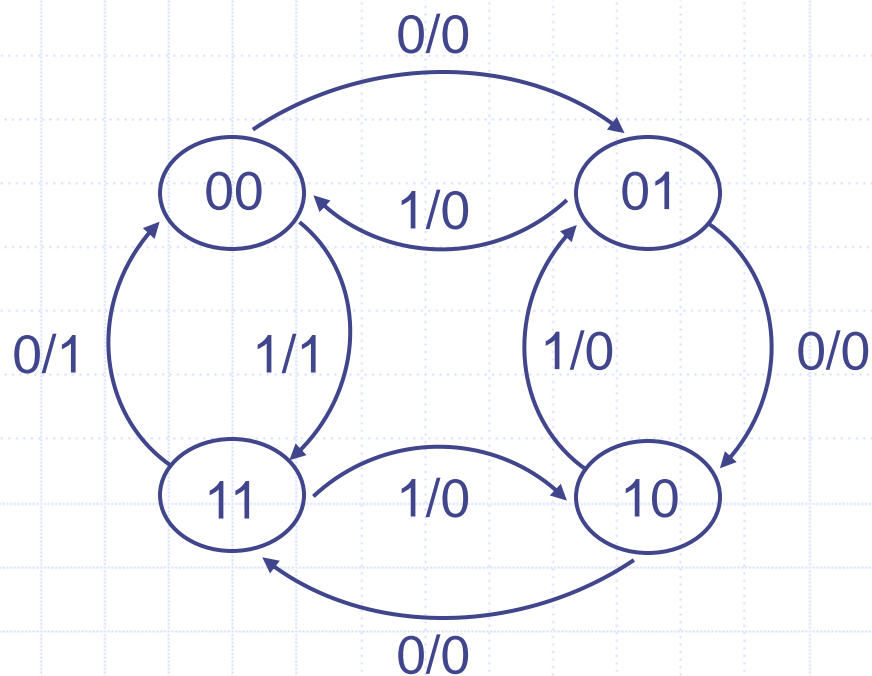
طراحی مدار های ترتیبی :

تجزیه و تحلیل

طراحی



مثال ۷: طراحی یک شمارنده دو بیتی بالا شمار (با ورودی ۰) و پایین شمار (با ورودی ۱)، خروجی Carry
(به کمک فلیپ فلاپ های JK)



(State Diagram)

جدول تحریک :

Q(t)	Q(t+1)	D	T	R	S	J	K
0	0	0	0	X	0	0	X
0	1	1	1	0	1	1	X
1	0	0	1	1	0	X	1
1	1	1	0	0	X	X	0

مثال ۷: (ادامه)

تعداد فلیپ فلاپ ها : $\log_2 4 = 2$

رسم جدول حالت :

$Q_2(t)$	$Q_1(t)$	x	$Q_2(t+1)$	$Q_1(t+1)$	J_2	K_2	J_1	K_1	Z
0	0	0	0	1	0	X	1	X	0
0	0	1	1	1	1	X	1	X	1
0	1	0	1	0	1	X	X	1	0
0	1	1	0	0	0	X	X	1	0
1	0	0	1	1	X	0	1	X	0
1	0	1	0	1	X	1	1	X	0
1	1	0	0	0	X	1	X	1	1
1	1	1	1	0	X	0	X	1	0

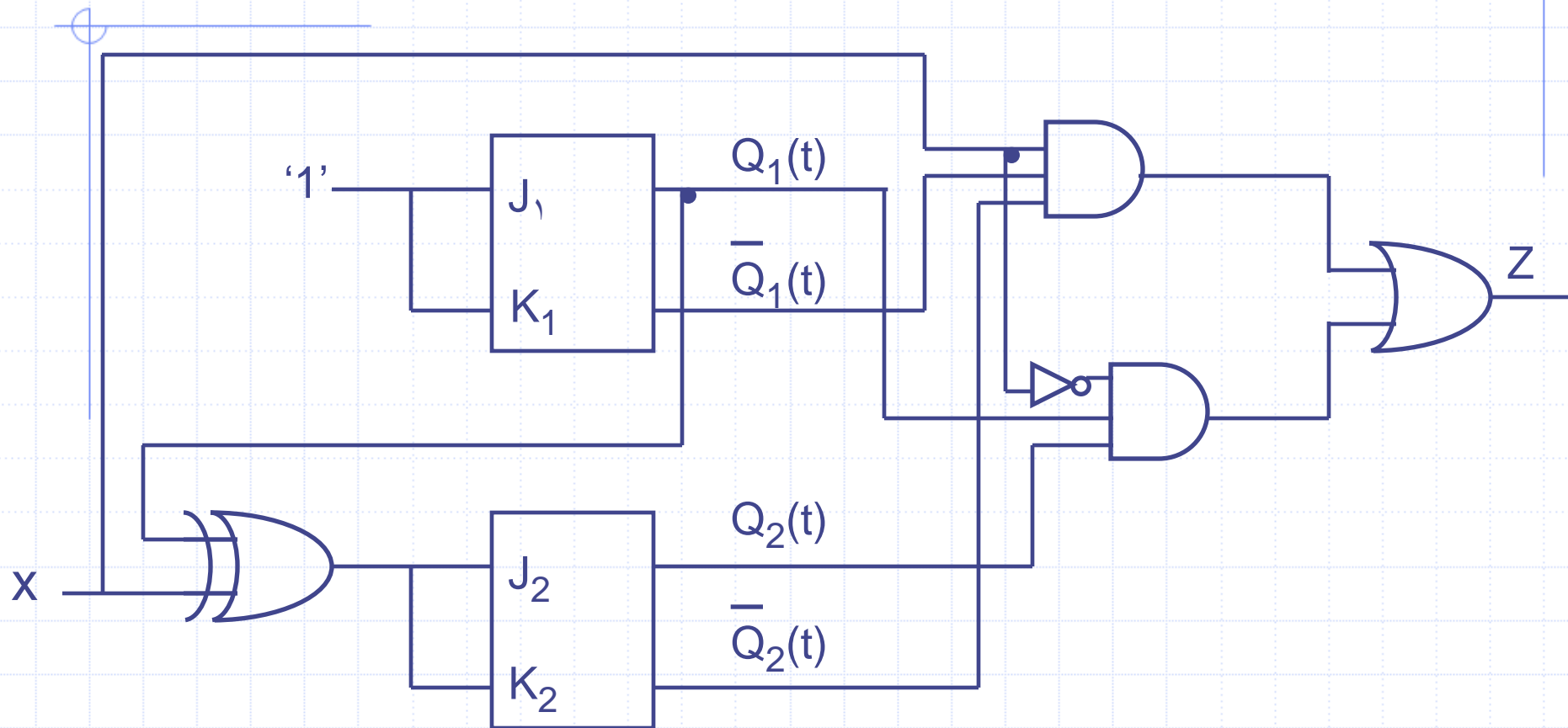
جدول کارنا :

		$Q_1(t) \ x$			
		00	01	11	10
$Q_2(t)$	0	X	X	X	X
	1		1		1

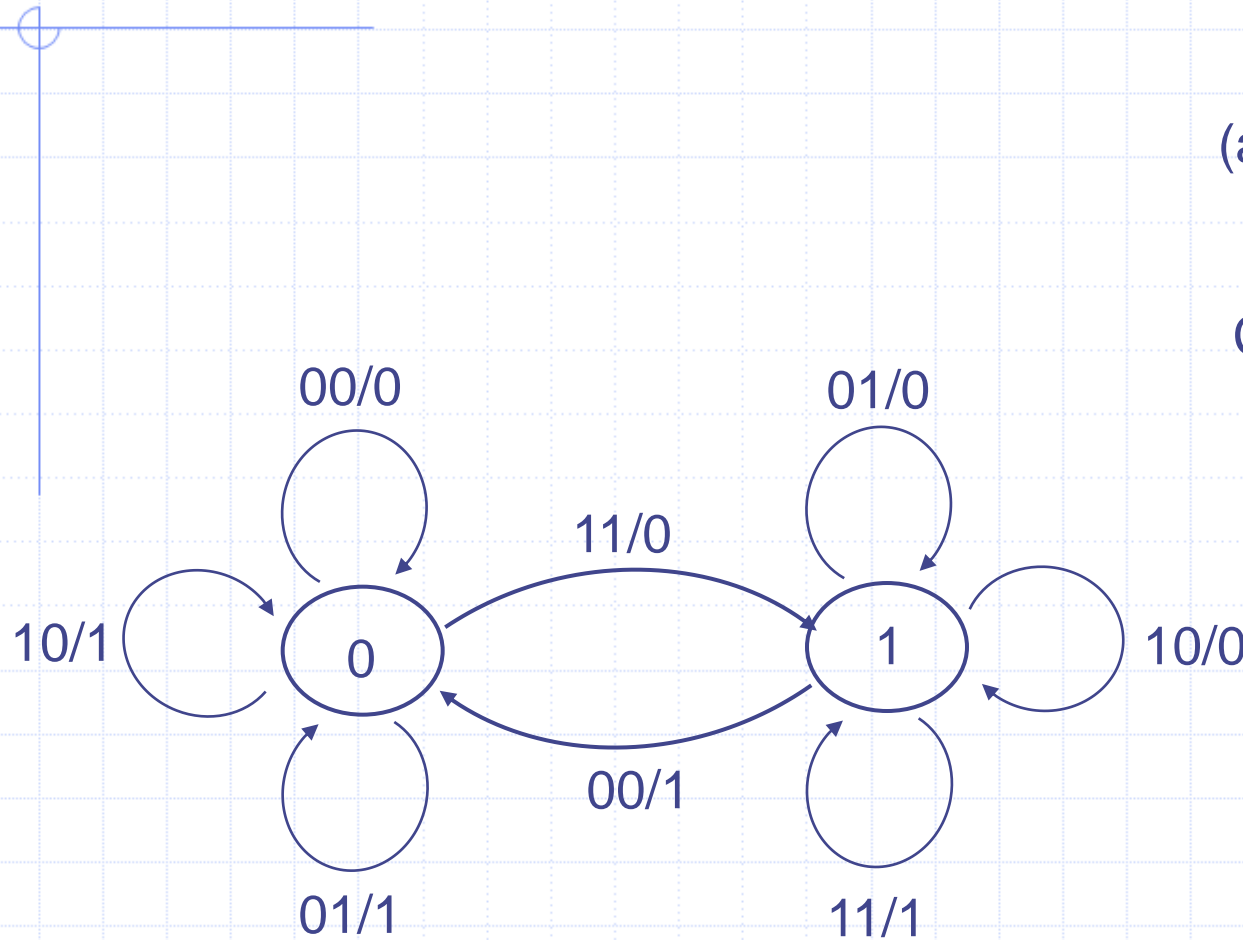
$$K_2 = Q_1(t) \oplus x$$

		$Q_1(t) \ x$			
		00	01	11	10
$Q_2(t)$	0		1		1
	1	X	X	X	X

$$J_2 = Q_1(t) \oplus x$$



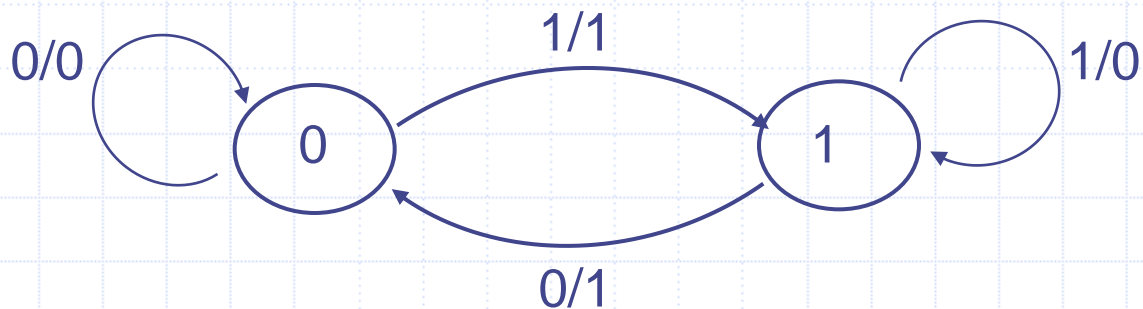
مثال ۸: مداری ترتیبی طراحی کنید که در هر کلاک پالس یک بیت هم مرتبه (هم ارزش) ، از دو عدد مثل **a** و **b** را دریافت کند و مجموع آنها را در خروجی نمایش دهد.



ورودی : ۲ بیت (a_i, b_i)
 خروجی: حاصل جمع S_i
 حالت : رقم نقلی C_{i+1}

مثال ۹: مداری ترتیبی طراحی کنید که در هر کلاک پالس یک بیت از ورودی دریافت نموده و Parity زوج را روی بیت دریافت شده در این کلاک و بیت دریافت شده در کلاک قبل، در خروجی نمایش دهد.

ورودی: ۱ بیت
خروجی: ۱ بیت p_e
حالت: بیت ما قبل



تمرین :

تمرین ۱ : مدار ترتیبی طراحی نمایید که در هر کلاک پالس یک بیت از ورودی گرفته ، **Parity** زوج را بر روی سه بیت جاری (این بیت و دو بیت ماقبل) محاسبه نموده و در خروجی قرار دهد.

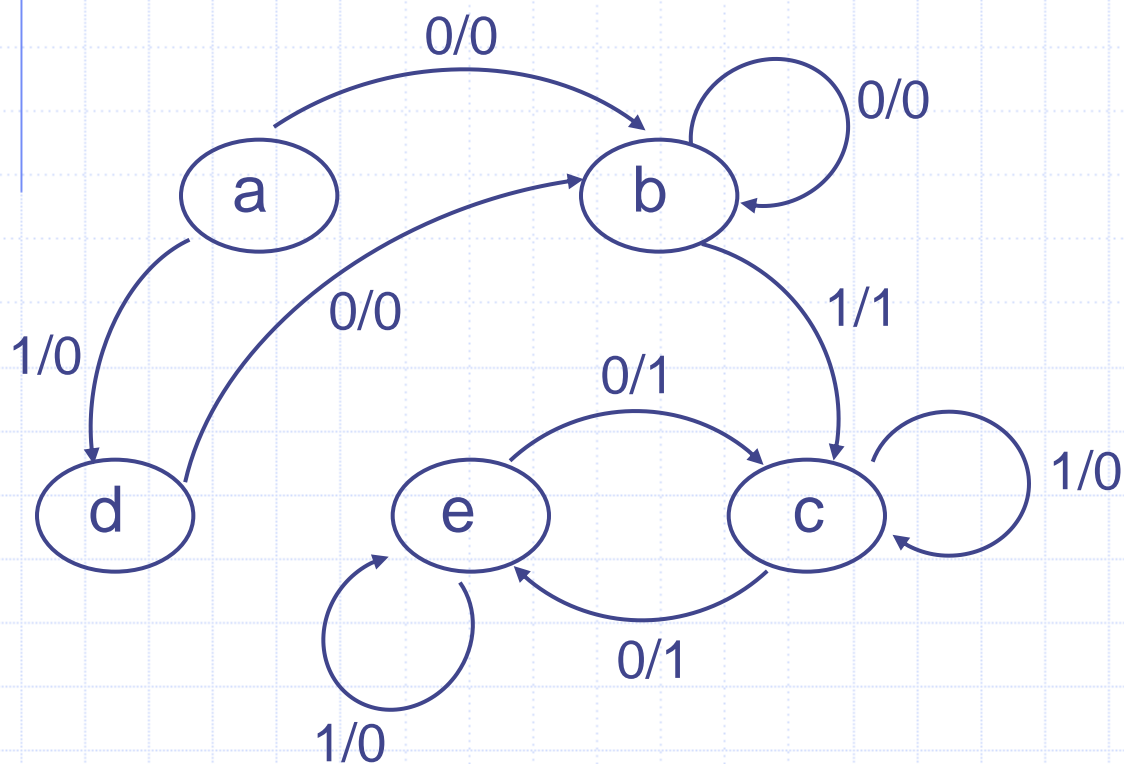
تمرین ۲ : مداری ترتیبی طراحی نمایید که در هر کلاک پالس و بیت جاری محاسبه کند. **Parity** زوج را روی کلیه بیت های ماقبل

تمرین ۳ : مداری ترتیبی طراحی نمایید که در هر کلاک پالس دو بیت از ورودی گرفته ، **Parity** زوج را روی کل بیت های دریافت شده تا این کلاک و خود این کلاک در خروجی نمایش دهد.

کمینه کردن یک State Diagram :

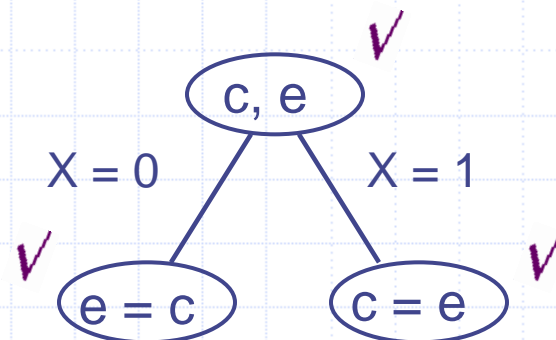
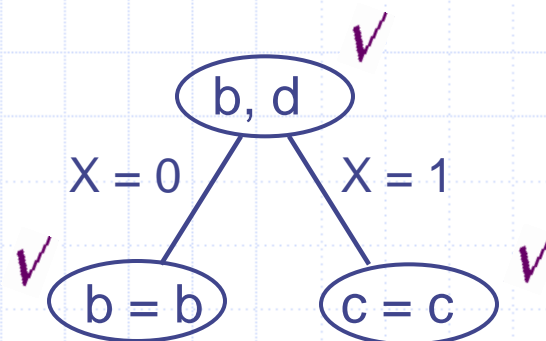
مثال ۱۰ :

اولین گام : دسته بندی State ها بر اساس خروجی ها

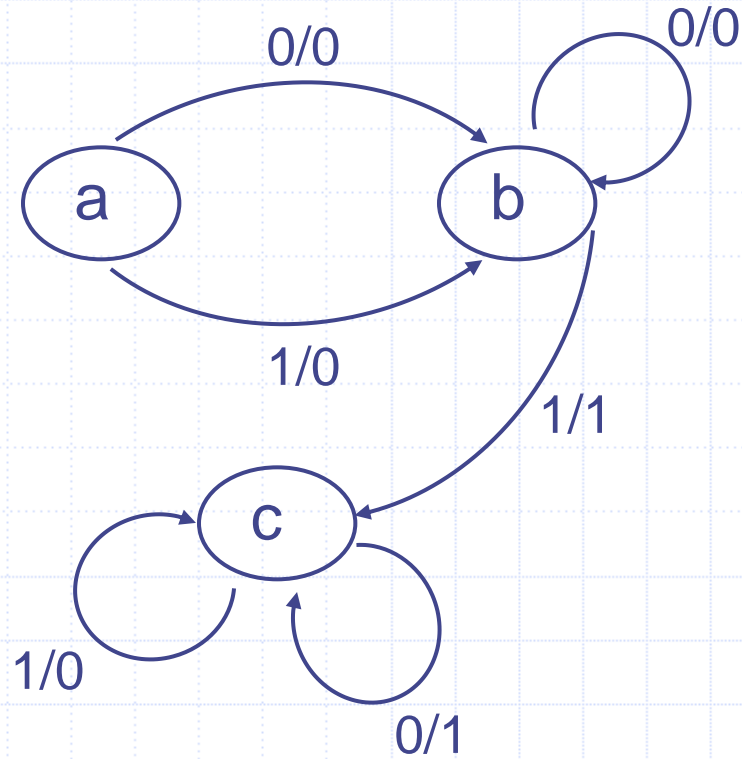


دومین گام :

		حالت بعدی		خروجی	
		X = 0	X = 1	X = 0	X = 1
00	a	b	d ^b	0	0
01	b	b	c	0	1
10	c	e ^c	c	1	0
01	d	b	c	0	1
10	e	c	e	1	0

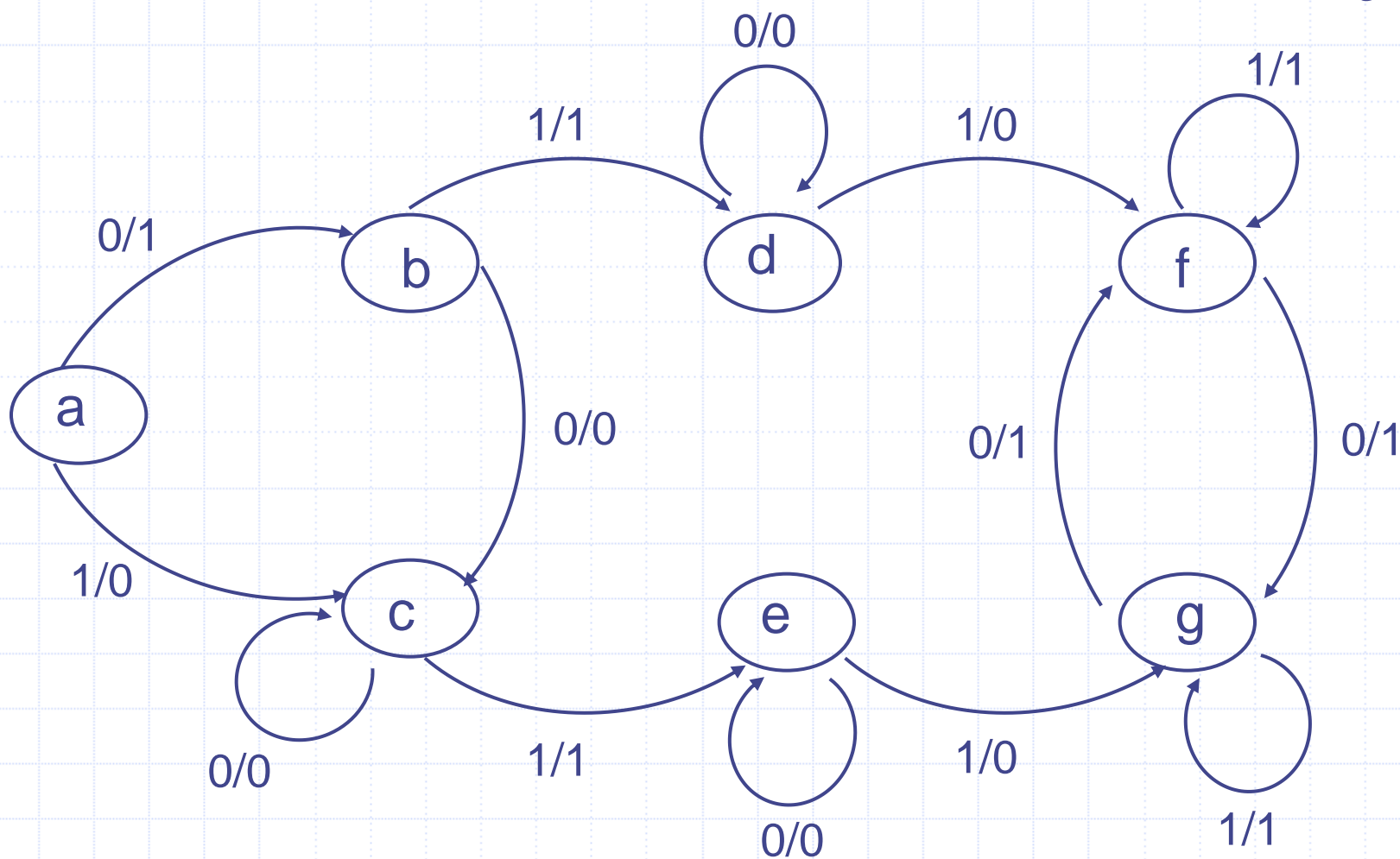


State Diagram معادل :



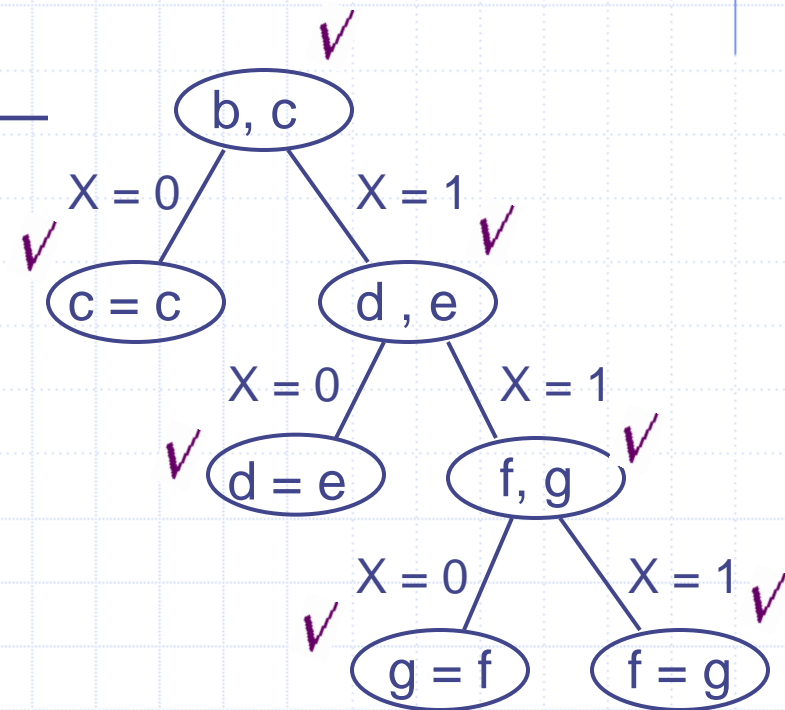
کمینه کردن یک State Diagram : (ادامه)

مثال ۱۱ :

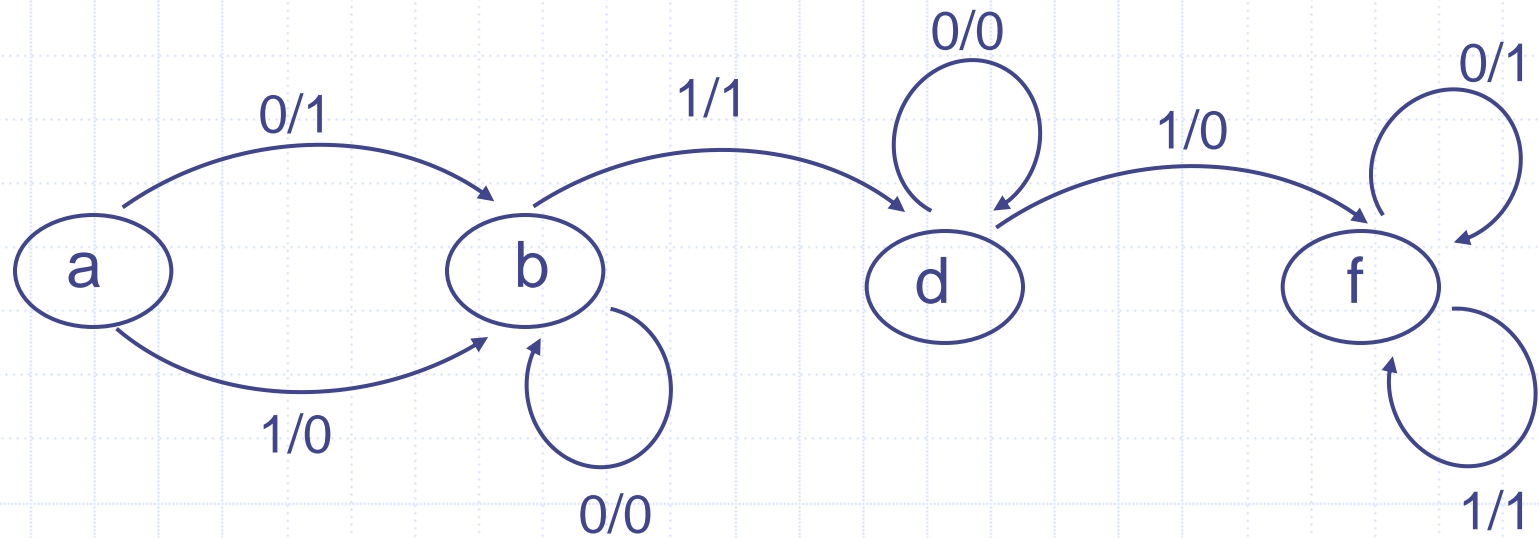


مثال ۱۱: (ادامه)

		حالت بعدی		خروجی	
		X = 0	X = 1	X = 0	X = 1
۱. ۱	a	b	c ^b	1	0
۱. ۱	b	c ^b	d	0	1
۱. ۱	c	c	e	0	1
۱. ۱	d	d	f	0	0
۱. ۱	e	e	g	0	0
۱. ۱	f	g ^f	f	1	1
۱. ۱	g	f	g	1	1



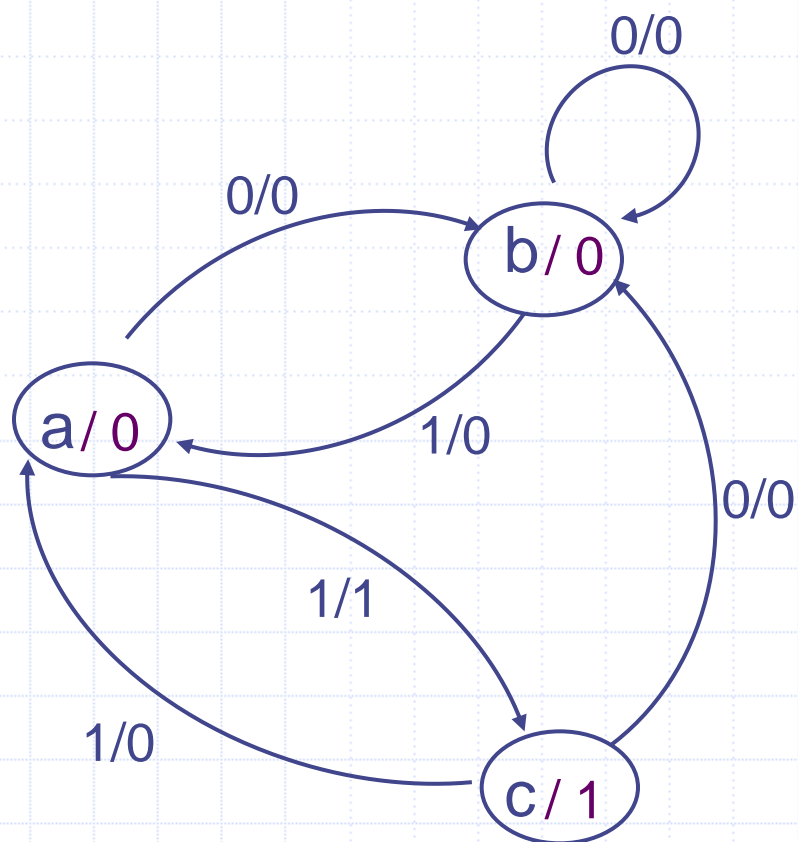
State Diagram معادل :



مدارها:

میلی (mili): خروجی تابعی از ورودی و حالت است.

مور (mor): خروجی تابعی از حالت است.



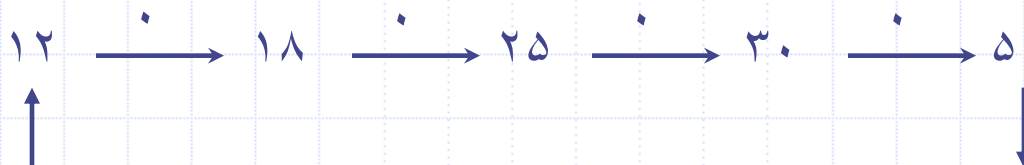
تمرین :

تمرین ۱ : مداری ترتیبی طراحی کنید که به عنوان یک تشخیص دهنده ی الگو ، الگوی بیتی ۱۱۰۱ را تشخیص دهد و به ازای آن خروجی را Set نماید. توجه داشته باشید که در هر کلاک پالس یک بیت از ورودی دریافت می شود.

تمرین ۲ : مداری ترتیبی طراحی نمایید که با رشته بیت ورودی برخورد عددی داشته باشد و در صورتی که عدد دریافت شده ، مضرب ۵ بود ، خروجی را Set کند ، در غیر این صورت خروجی صفر باشد.

تمرین ۳ : شمارنده ای طراحی کنید که به صورت زیر عمل شمارش را انجام دهد. در طراحی این مدار لازم است کلیه اصول ساده سازی برای کاهش حجم مدار ترکیبی را در نظر بگیرید.

ورودی ۱ در هر مرحله مانند ، دو بار ورودی صفر عمل می کند.



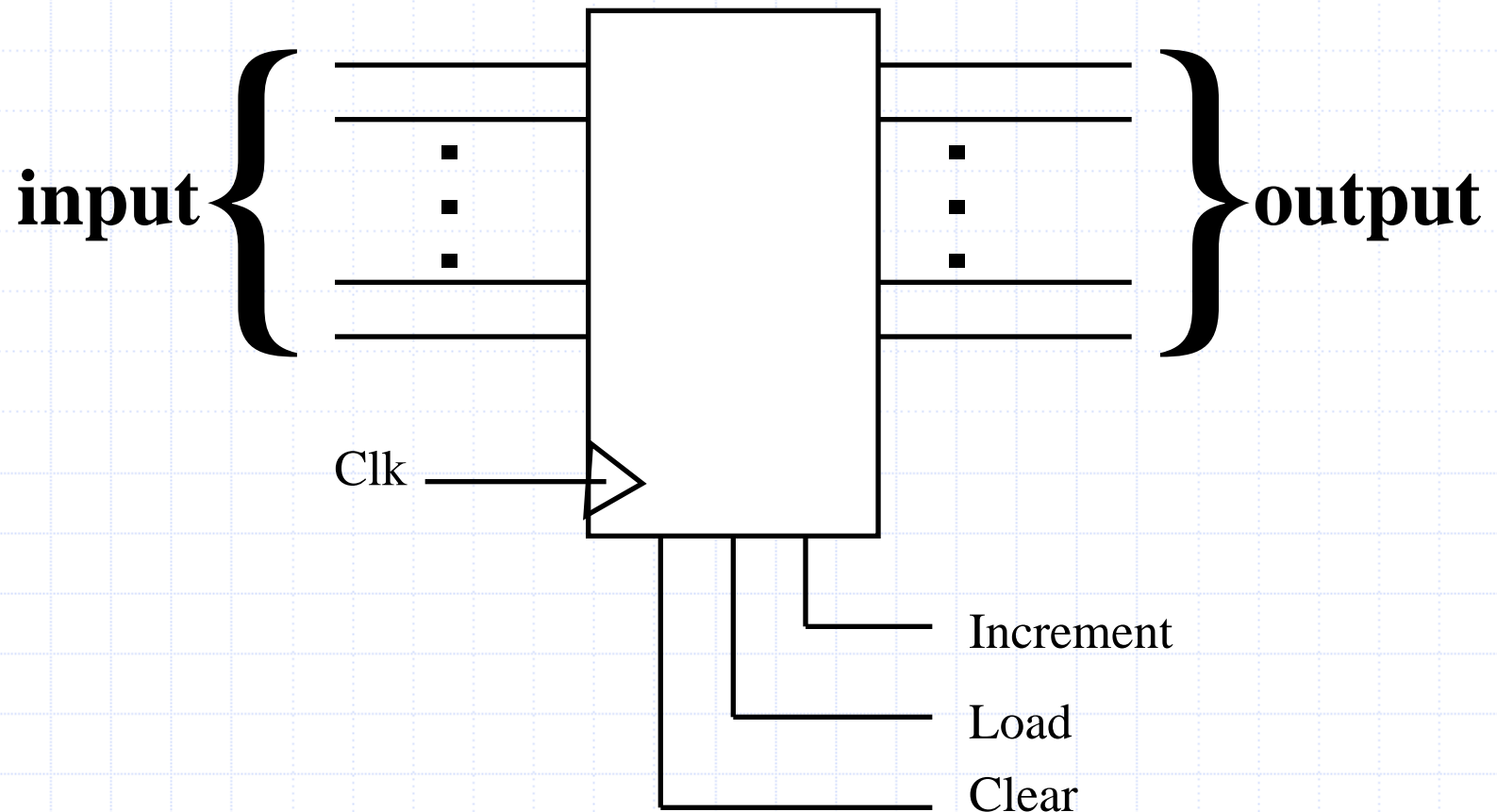
فصل ۷

ثبات ها و شیفٲ رژیستر

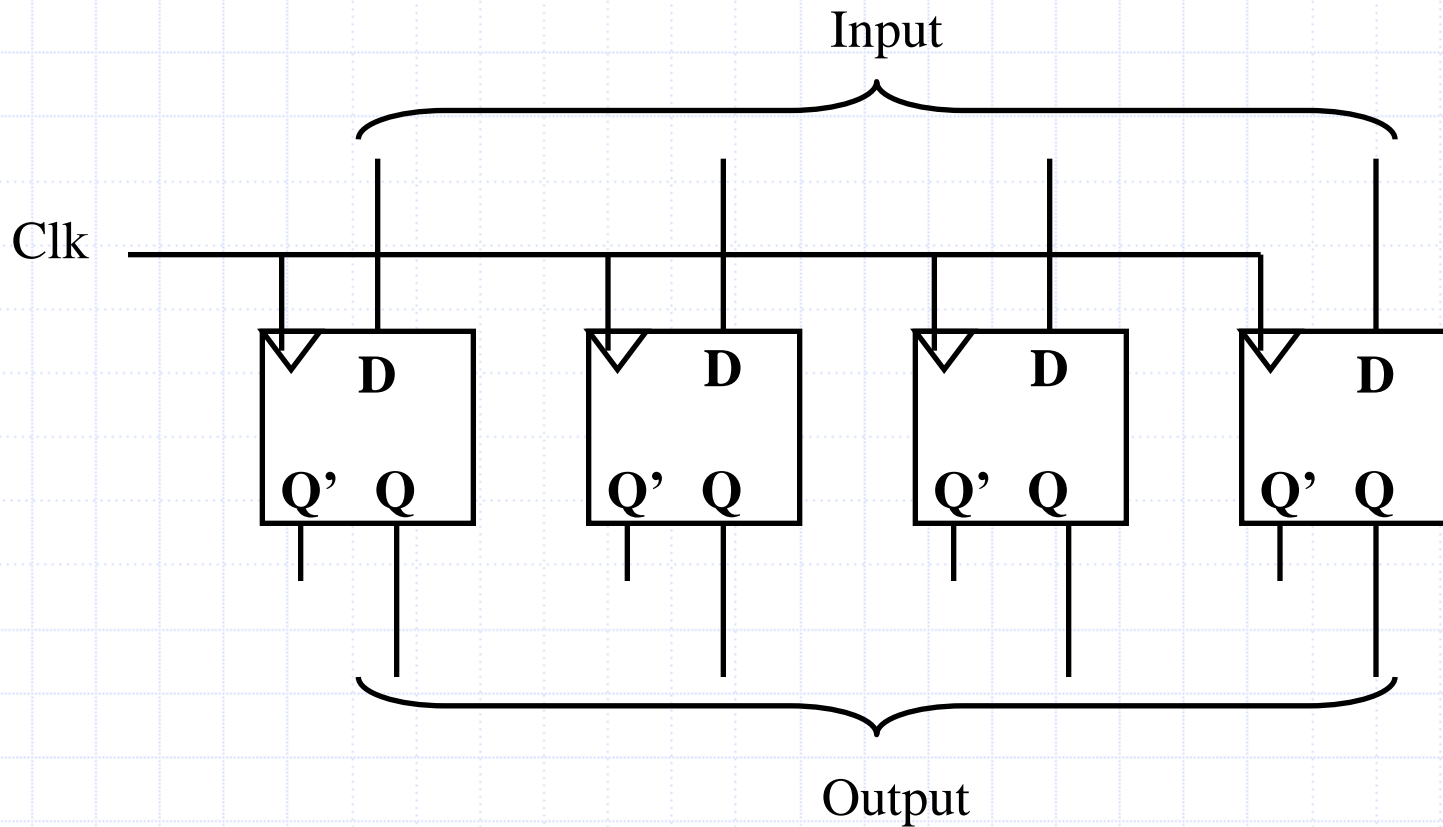
فهرست مطالب

- طرح بلوک دیاگرامی ثبات
- طرح ساده یک ثبات با فلیپ فلاپ D
- طرح یک ثبات با فلیپ فلاپ Jk به پایه Load
- طرح یک ثبات با پایه Load و Clear
- شیفتر رجیستر با فلیپ فلاپ D
- شیفتر رجیستر با فلیپ فلاپ JK
- شمارنده

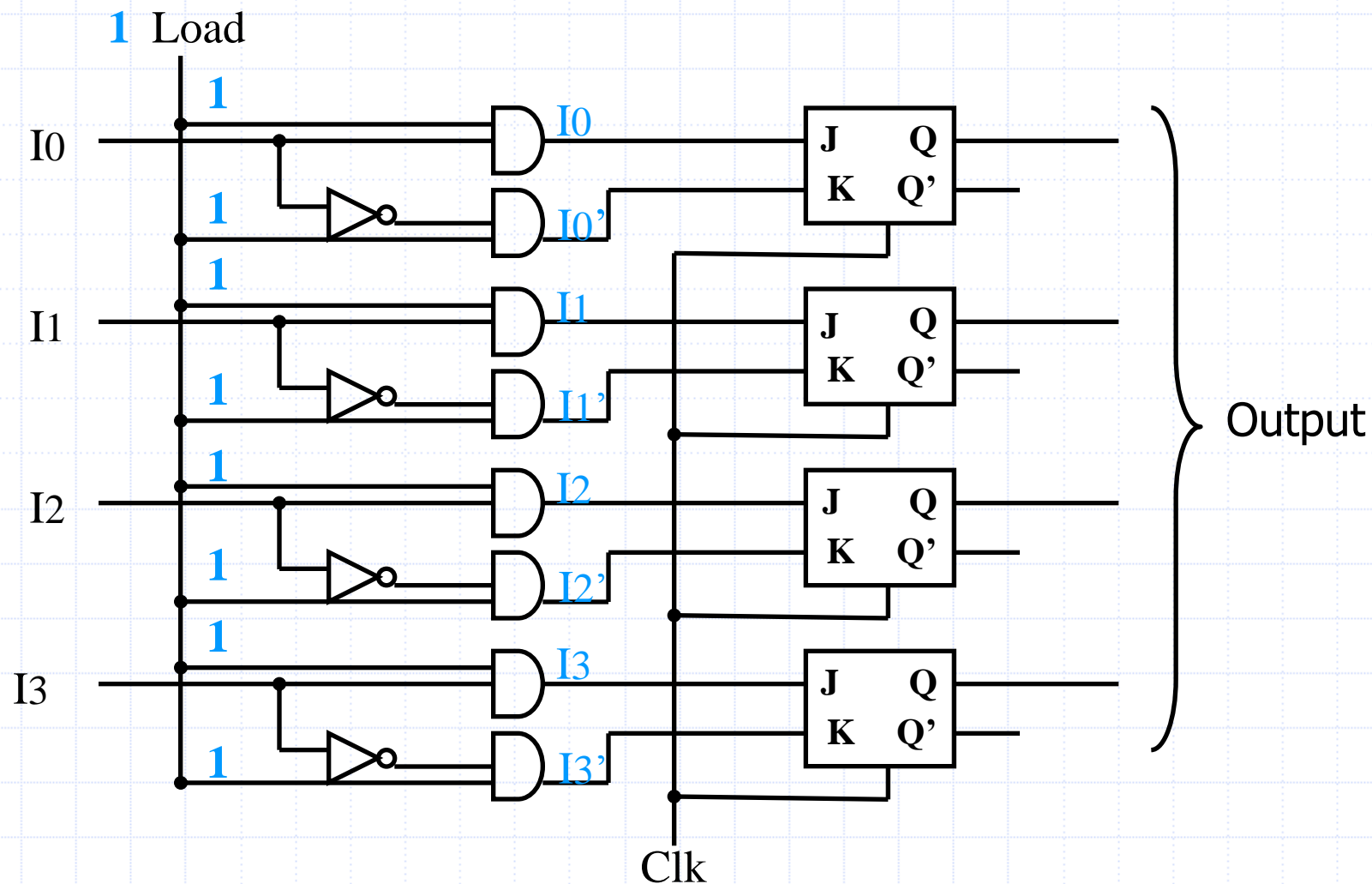
طرح بلوک دیاگرامی ثبات



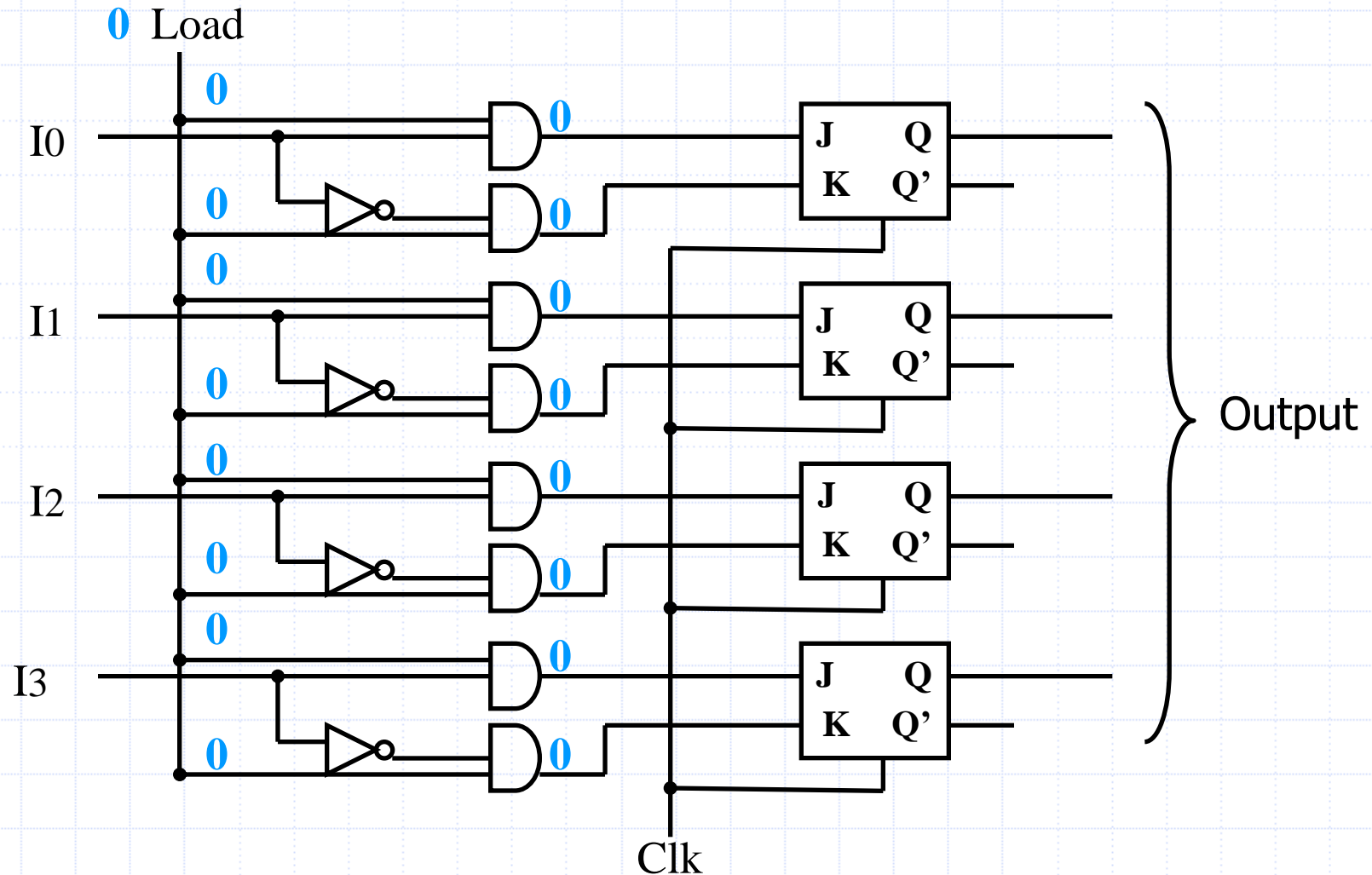
طرح ساده یک ثبات با فلیپ فلاپ D



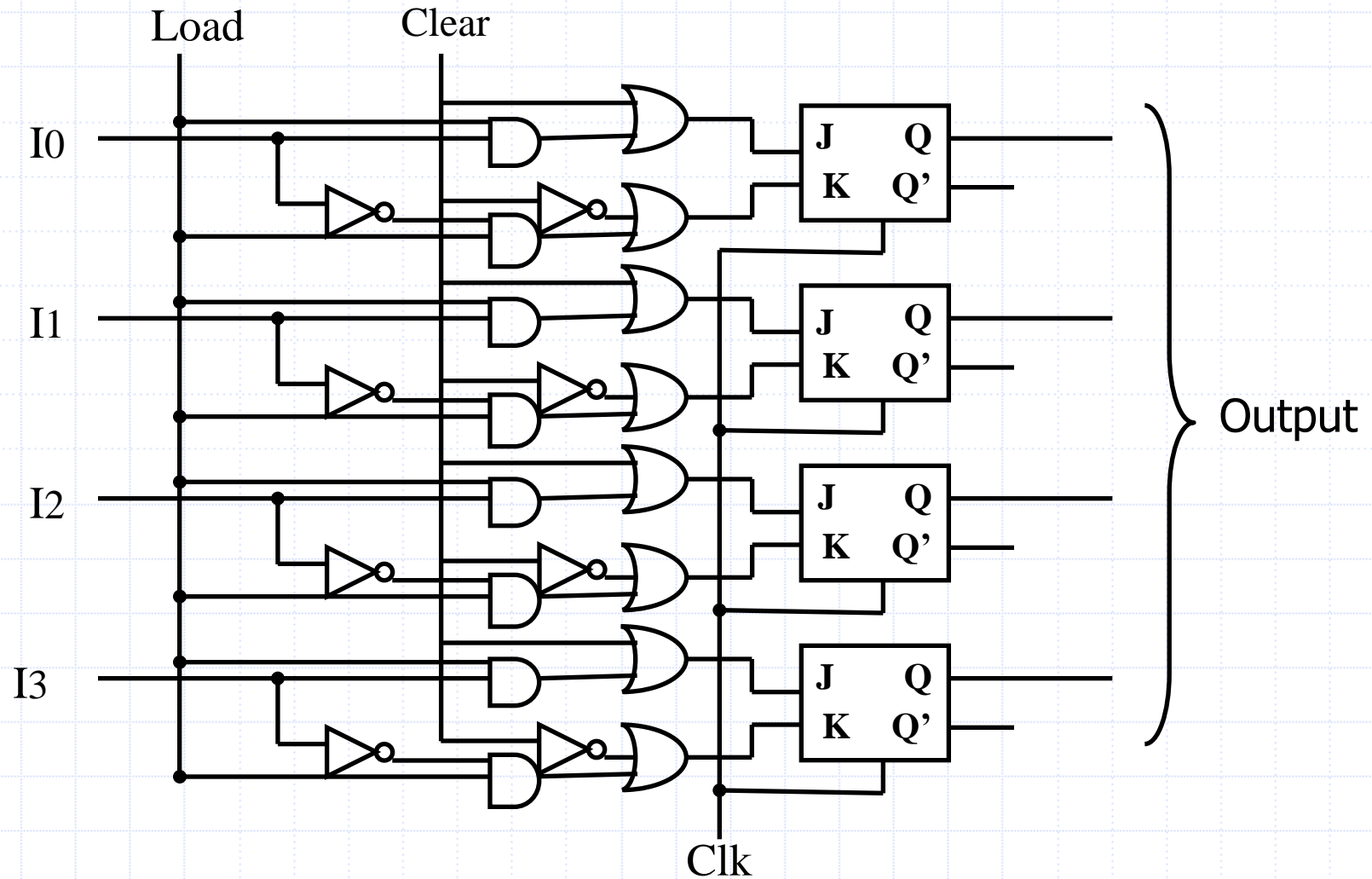
طرح یک ثبات با فلیپ فلاپ JK و پایه Load



طرح یک ثبات با فلیپ فلاپ JK و پایه Load



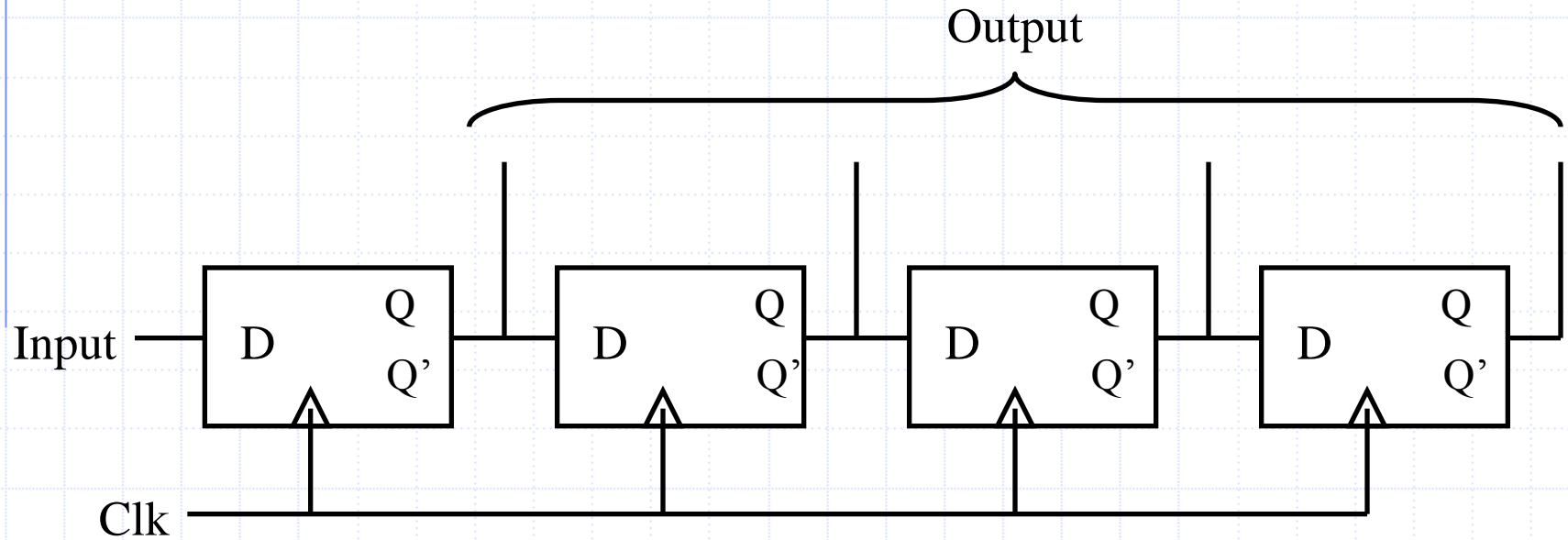
طرح یک ثبات با پایه Load و Clear



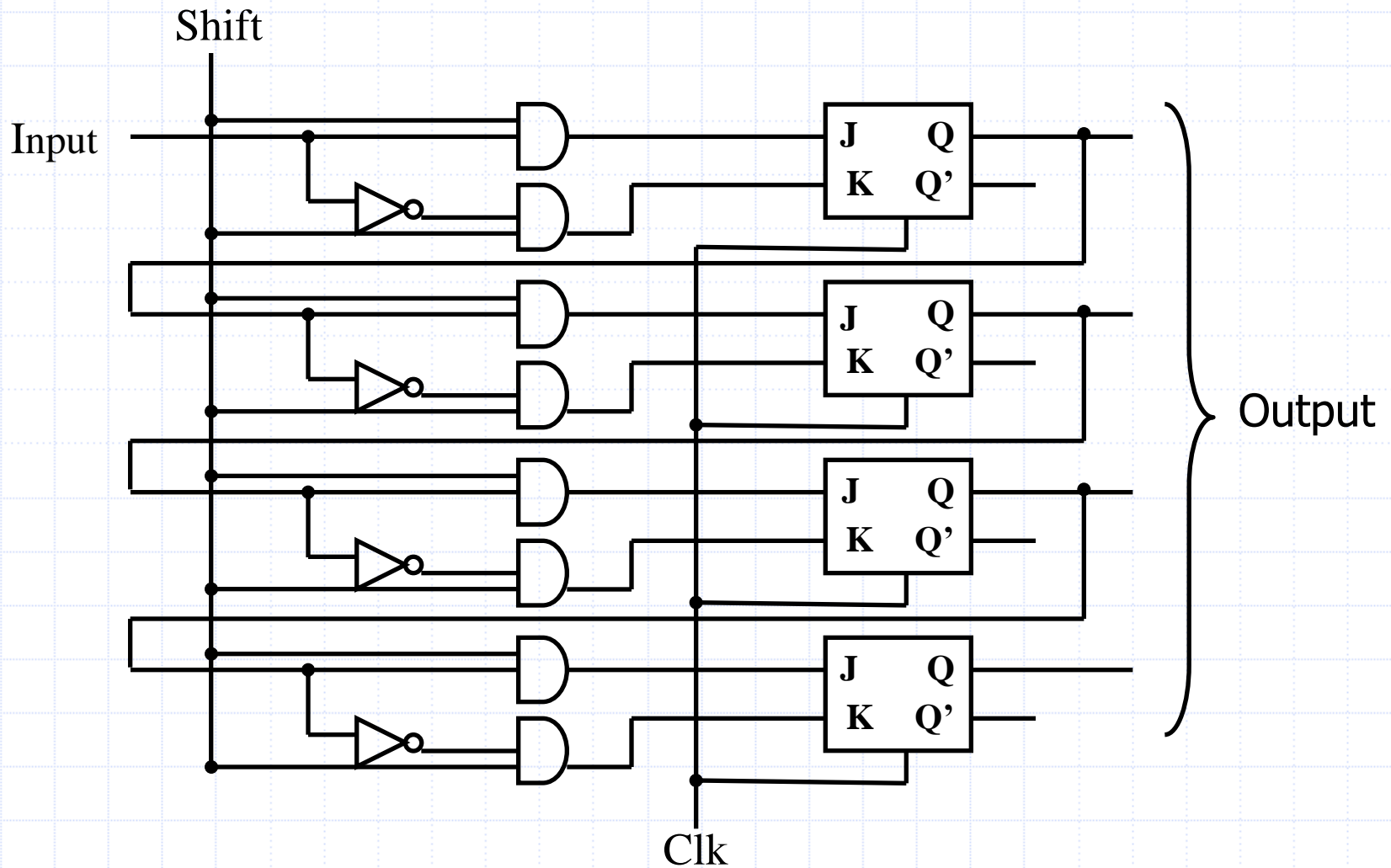
تمرین:

□ ثباتی طراحی کنید پایه سومی به نام Increment داشته باشد.

شيفت رجیستر با فلیپ فلاپ D



شيفت رجیستر با فلیپ فلاپ JK



شمارنده

□ سنکرون (هنگام): در این نوع تمام واحدهای ترتیبی مدار با یک **Clk** کار می کنند.

□ آسنکرون (ناهمگام): در این نوع هر واحد **Clk** مجزایی دارد.

شمارنده

□ منظم

□ بالا شمار

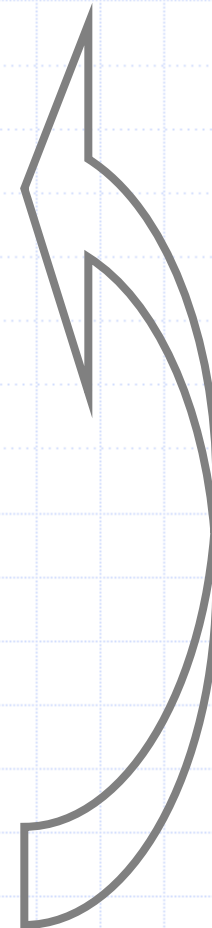
□ پائین شمار

□ نامنظم

شمارنده ۳ بیتی

بیت

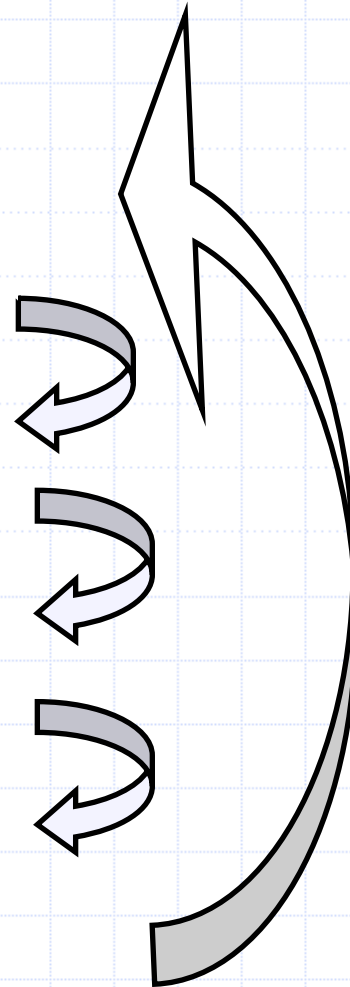
Q2	Q1	Q0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1



شمارنده ۳ بیتی (ادامه)

بیت ۱

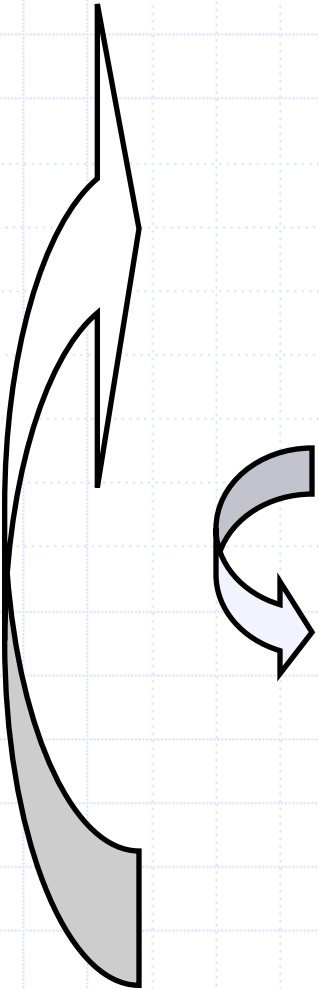
Q2	Q1	Q0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1



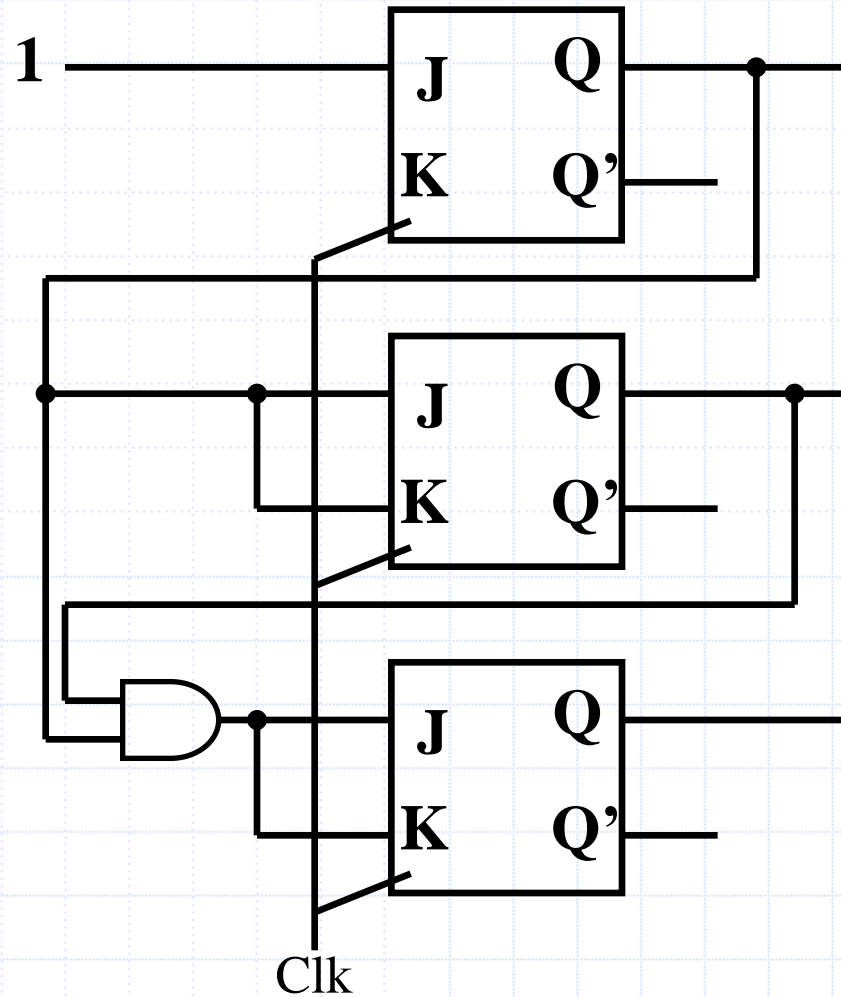
شمارنده ۳ بیتی (ادامه)

بیت ۲

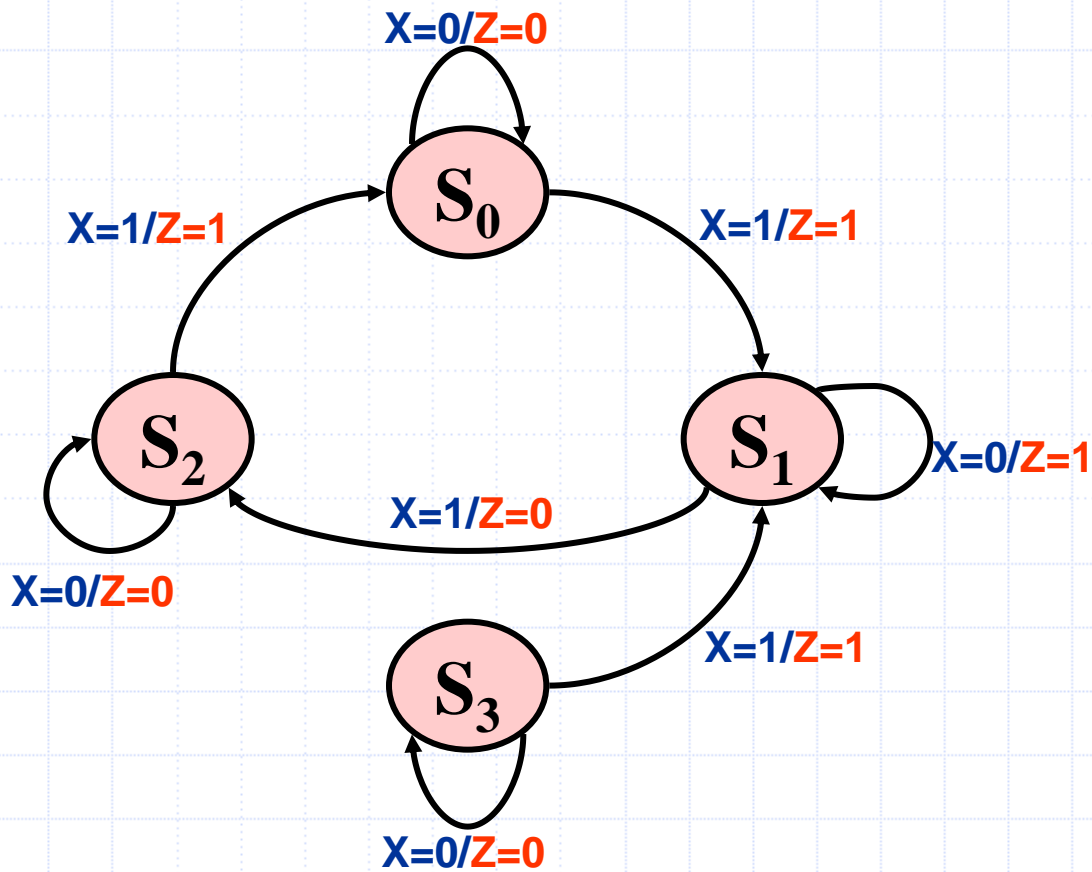
Q2	Q1	Q0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1



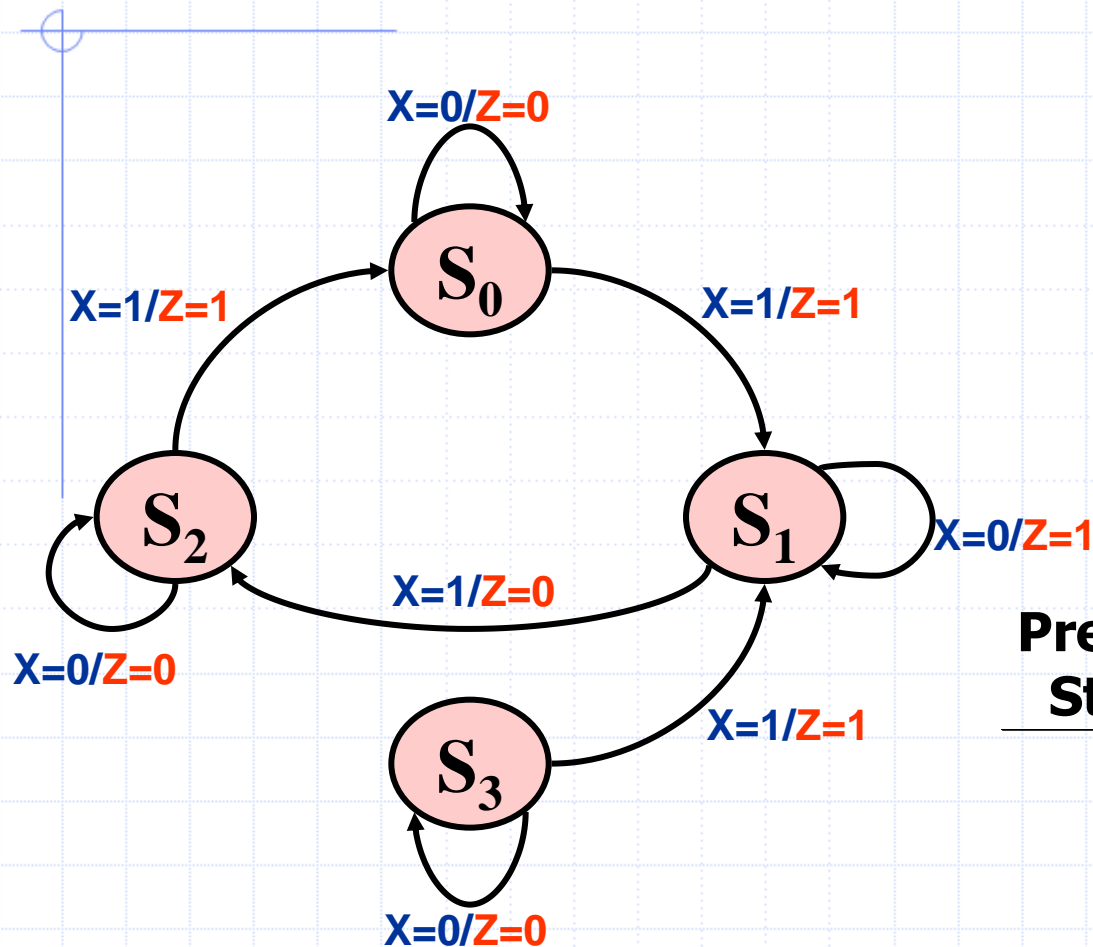
مدار یک شمارنده ۳ بیتی سنکرون



مثالی از یک ماشین میلی:

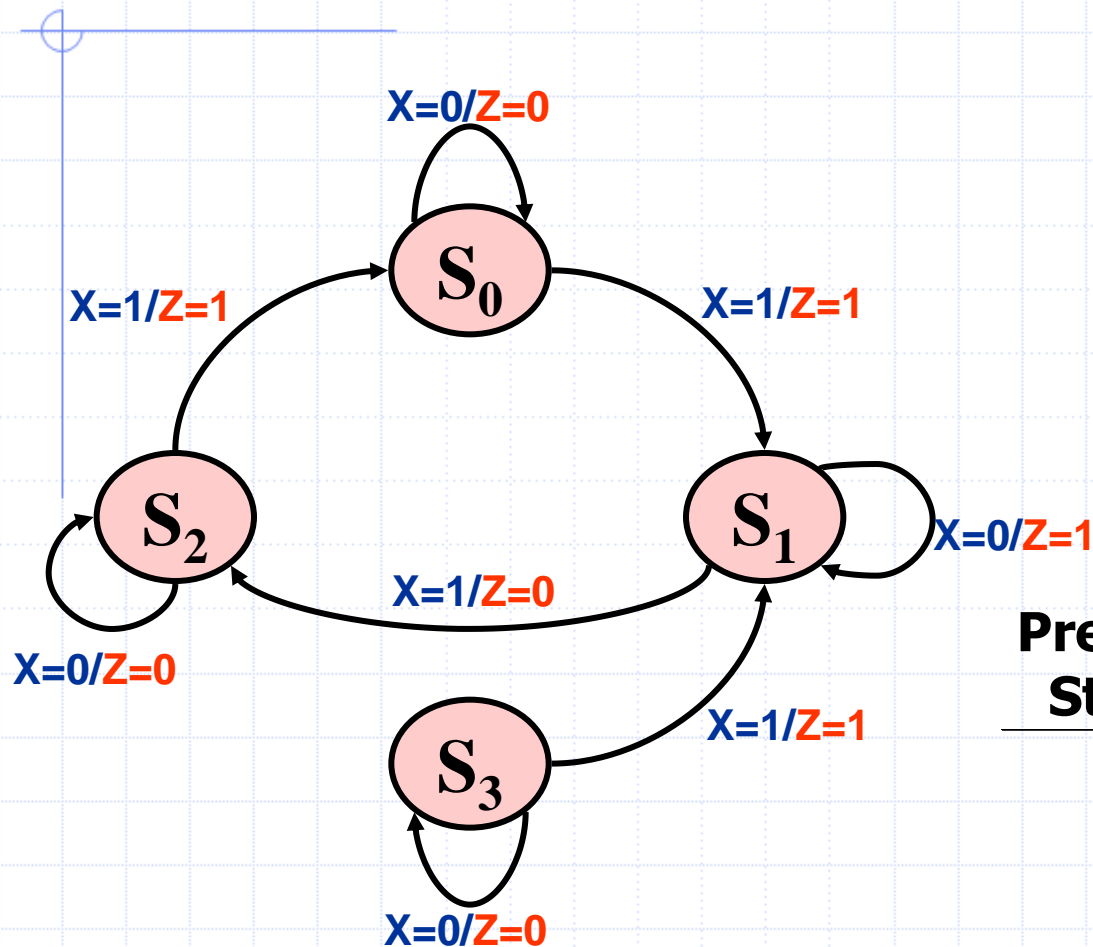


مثالی از یک ماشین میلی:



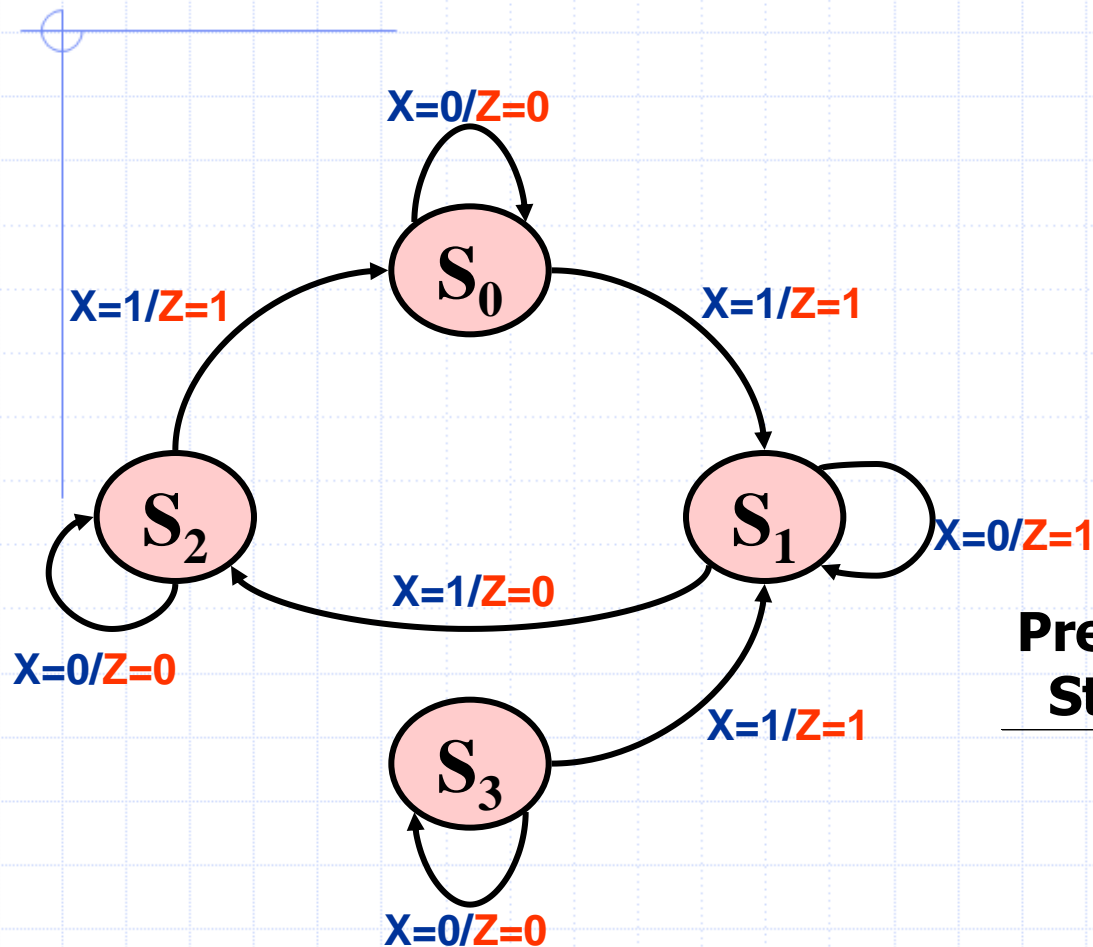
Present State	Next State		Output	
	X=0	X=1	X=0	X=1
S_0	S_0	S_1	0	1
S_1				
S_2				
S_3				

مثالی از یک ماشین میلی:



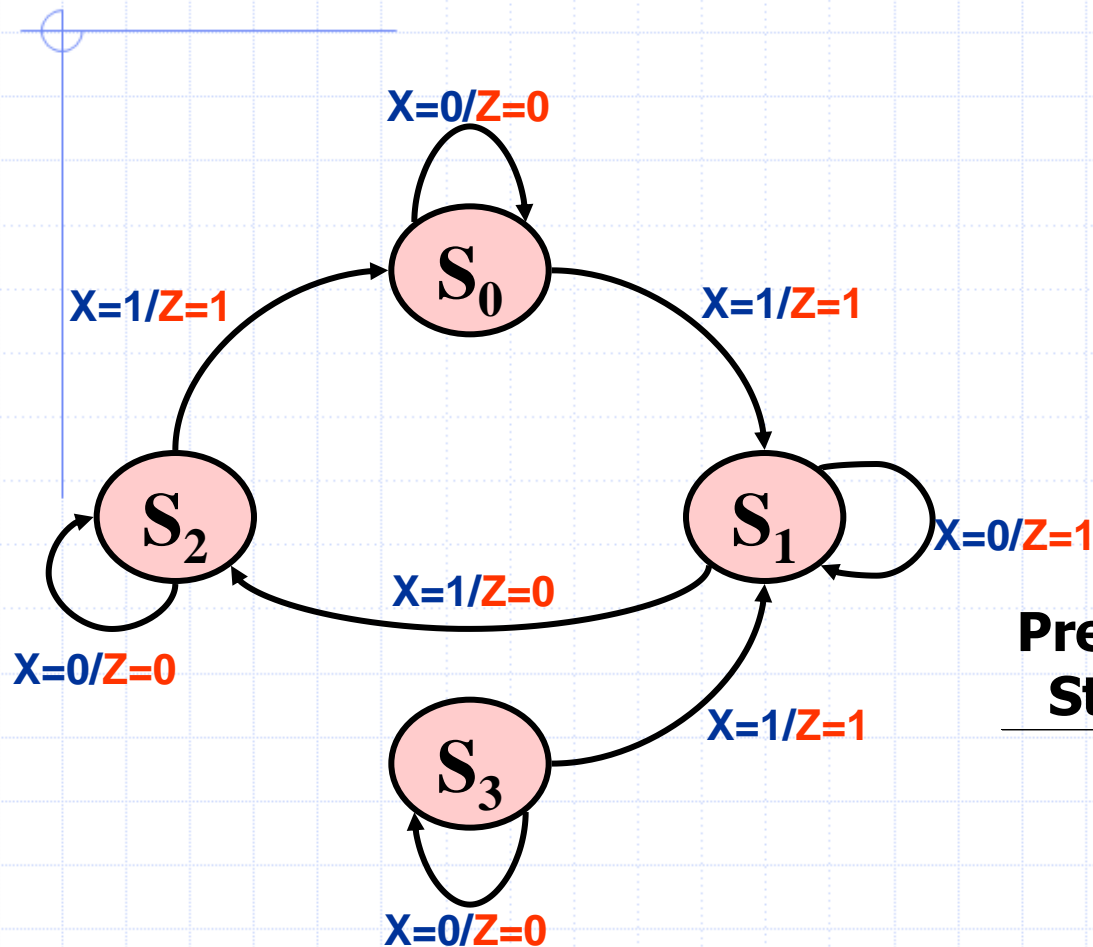
Present State	Next State		Output	
	X=0	X=1	X=0	X=1
S_0	S_0	S_1	0	1
S_1	S_1	S_2	1	1
S_2				
S_3				

مثالی از یک ماشین میلی:



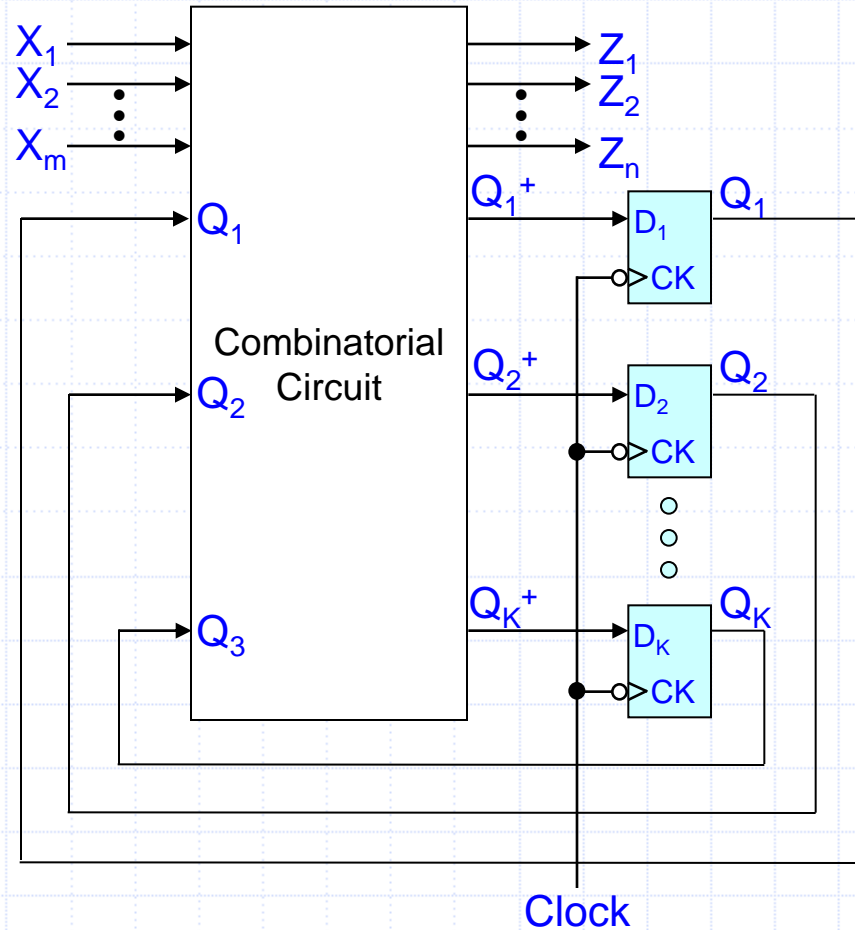
Present State	Next State		Output	
	X=0	X=1	X=0	X=1
S_0	S_0	S_1	0	1
S_1	S_1	S_2	1	1
S_2	S_2	S_0	0	1
S_3				

مثالی از یک ماشین میلی:



Present State	Next State		Output	
	X=0	X=1	X=0	X=1
S_0	S_0	S_1	0	1
S_1	S_1	S_2	1	1
S_2	S_2	S_0	0	1
S_3	S_3	S_1	0	1

مدل عمومی ماشین میلی:



A More Complex Sequence Detector

Design a sequence detector whose output Z is one if the input sequence is 010 or 1001

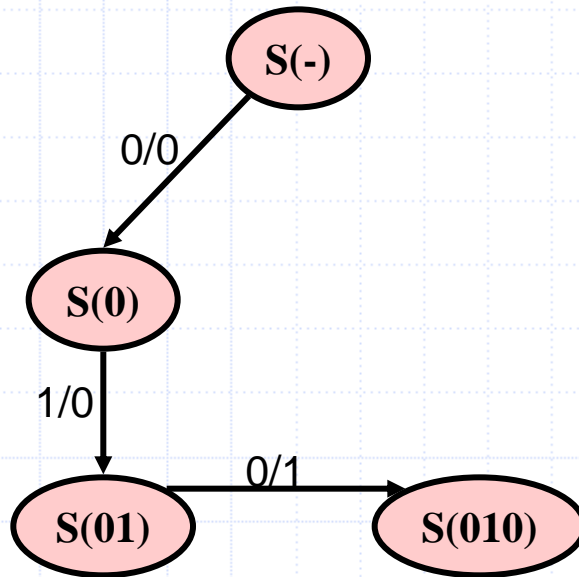
X =	0	0	1	0	1	0	0	1	0	0	0	1	0	0	1	1	0
Z =	0	0	0	1	0	1	0	1	1	0	0	0	1	0	1	0	0

Mealy Sequence Detector

Target Sequences:

010

1001

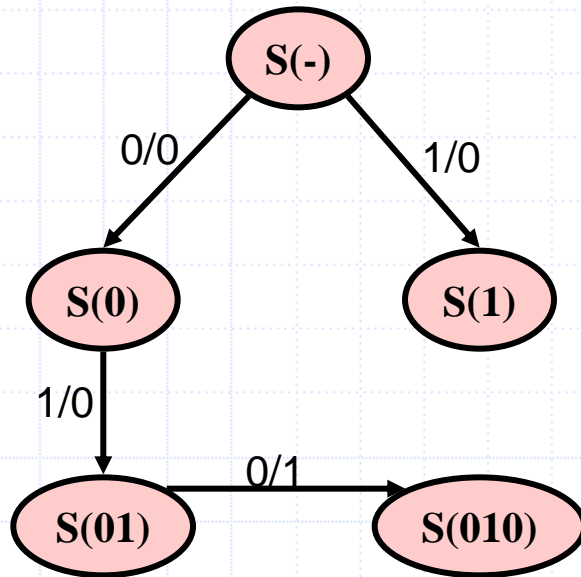


Mealy Sequence Detector

Target Sequences:

010

1001

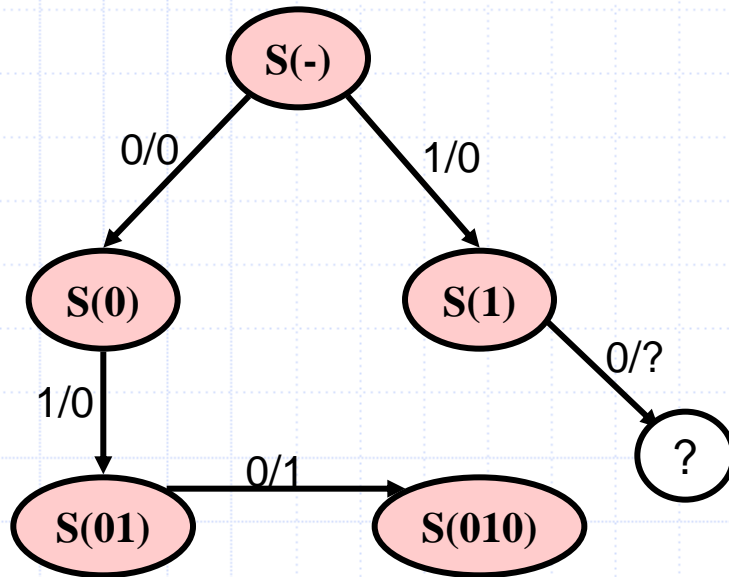


Mealy Sequence Detector

Target Sequences:

010

1001

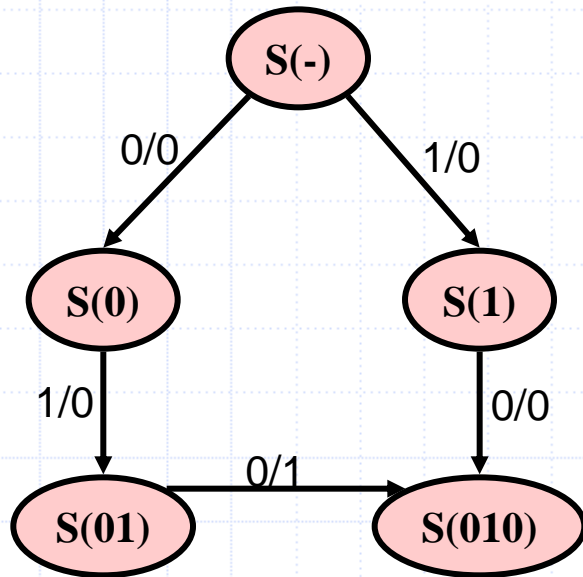


Mealy Sequence Detector

Target Sequences:

010

1001

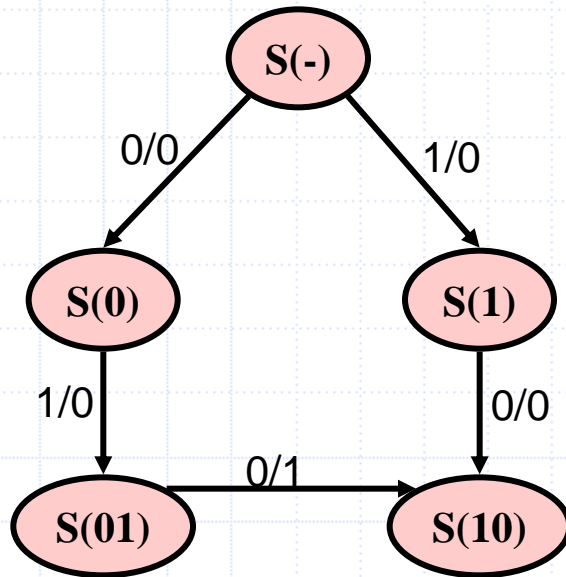


Mealy Sequence Detector

Target Sequences:

010

1001

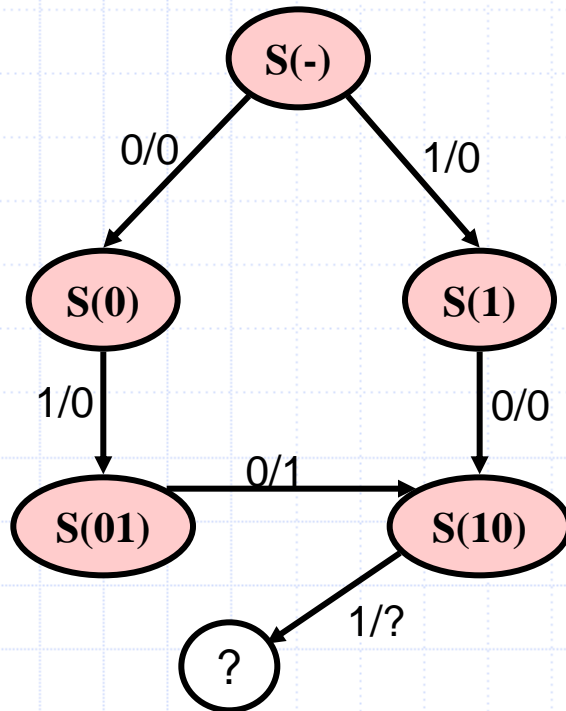


Mealy Sequence Detector

Target Sequences:

010

1001

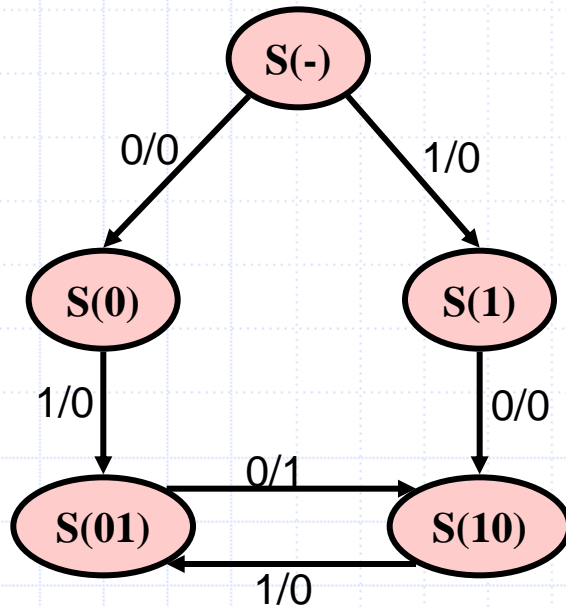


Mealy Sequence Detector

Target Sequences:

010

1001

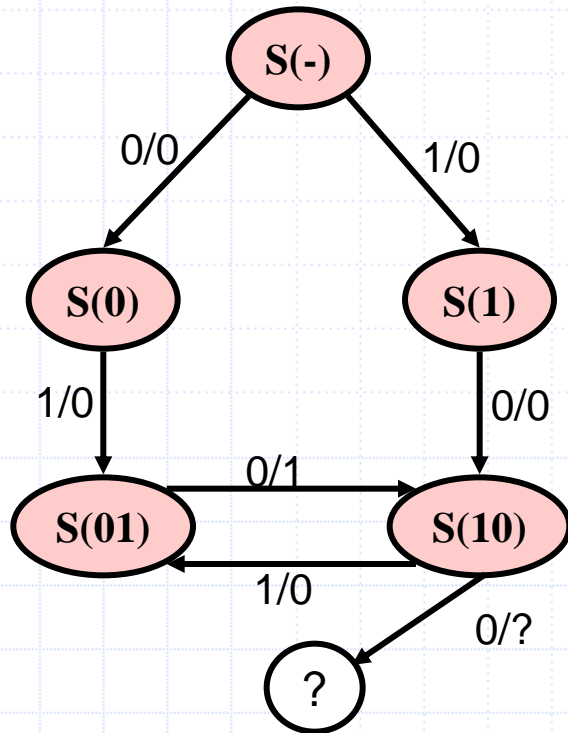


Mealy Sequence Detector

Target Sequences:

010

1001

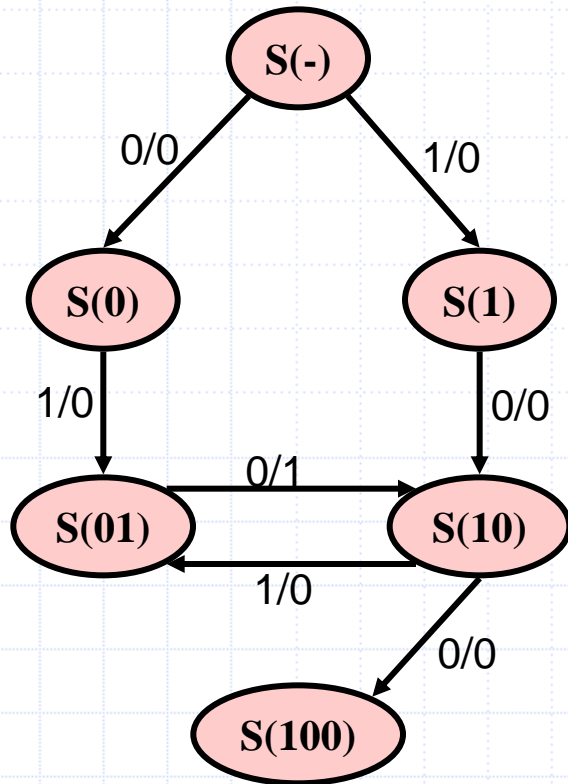


Mealy Sequence Detector

Target Sequences:

010

1001

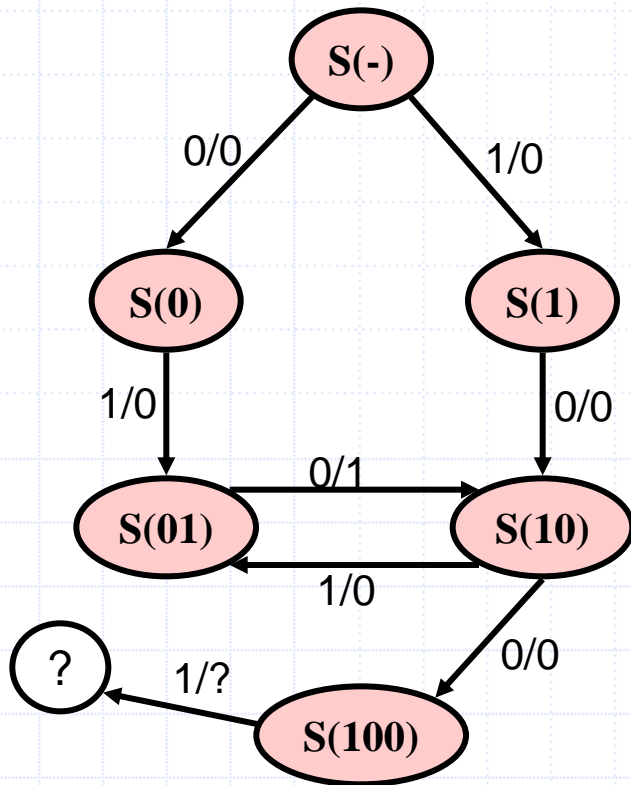


Mealy Sequence Detector

Target Sequences:

010

1001

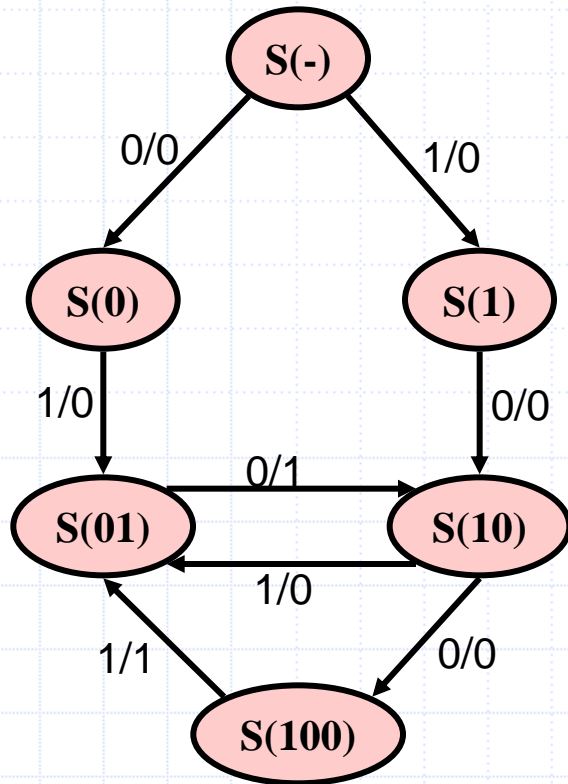


Mealy Sequence Detector

Target Sequences:

010

1001

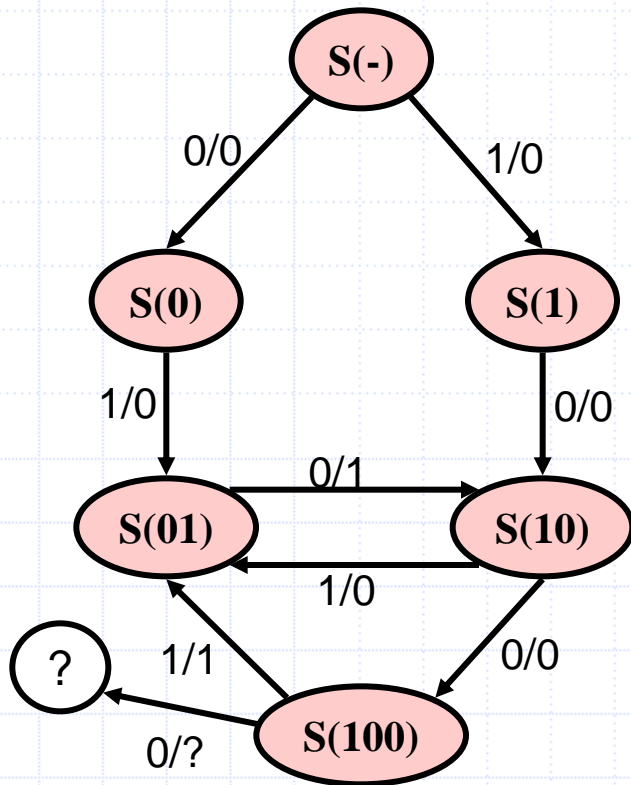


Mealy Sequence Detector

Target Sequences:

010

1001

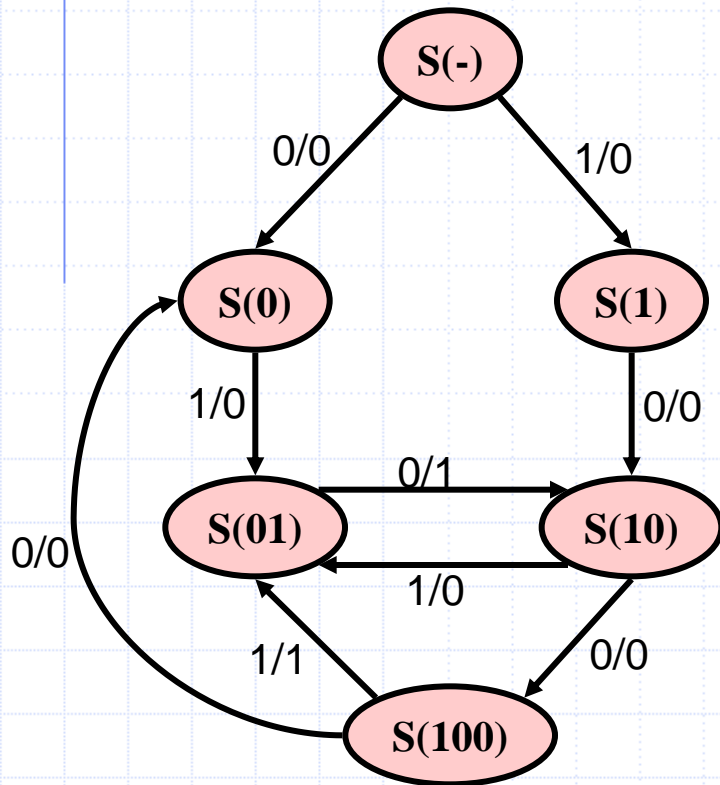


Mealy Sequence Detector

Target Sequences:

010

1001

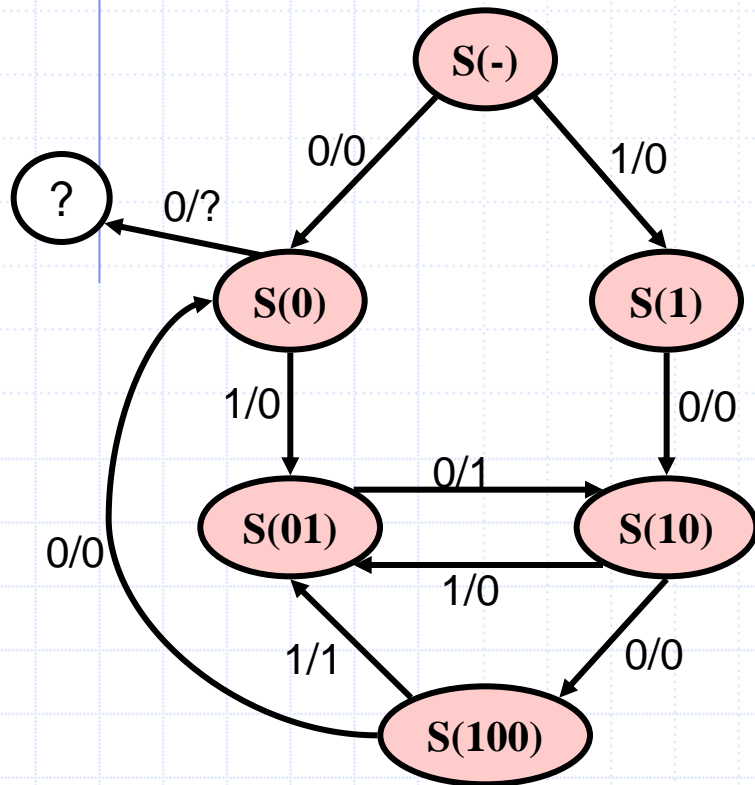


Mealy Sequence Detector

Target Sequences:

010

1001

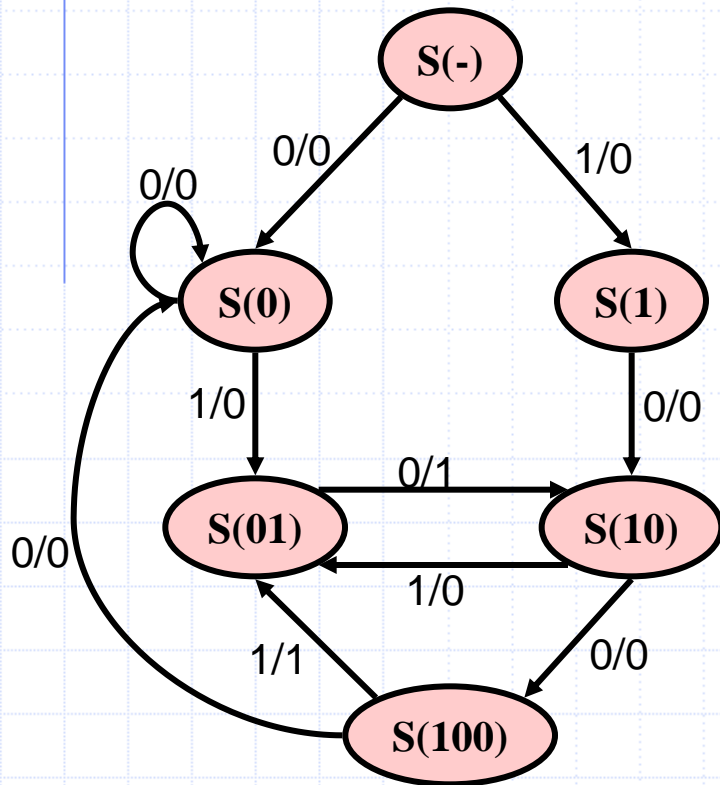


Mealy Sequence Detector

Target Sequences:

010

1001

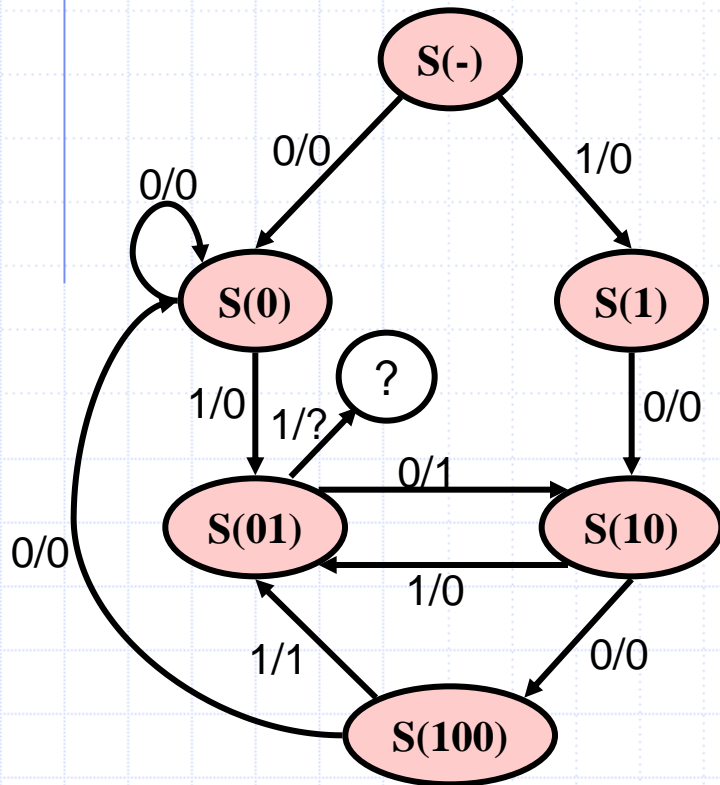


Mealy Sequence Detector

Target Sequences:

010

1001

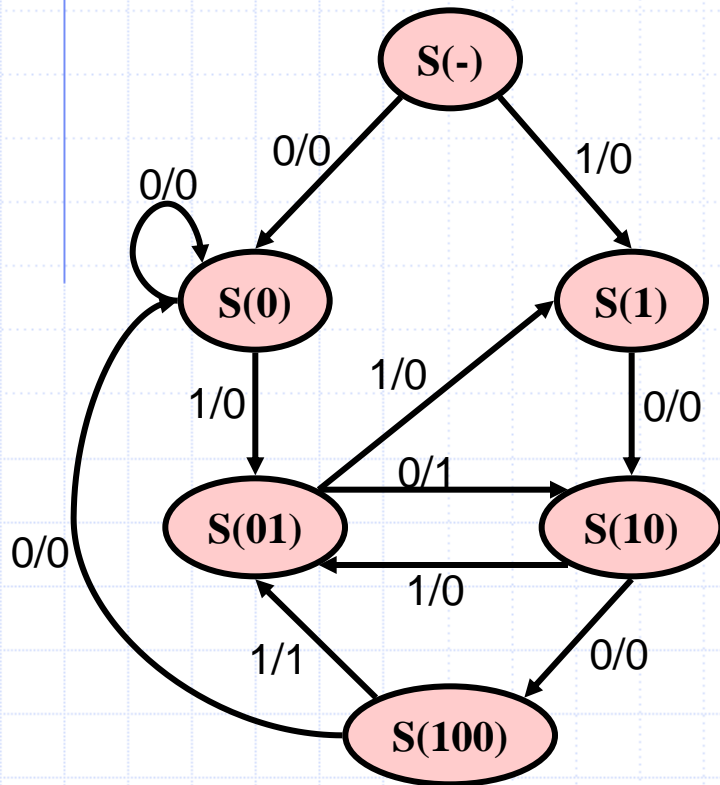


Mealy Sequence Detector

Target Sequences:

010

1001

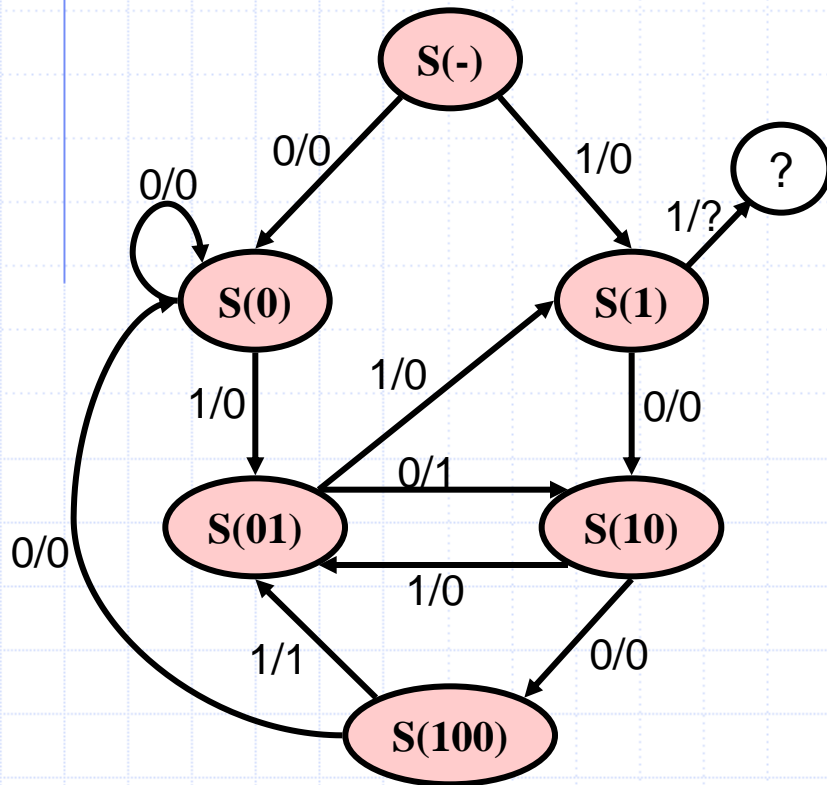


Mealy Sequence Detector

Target Sequences:

010

1001

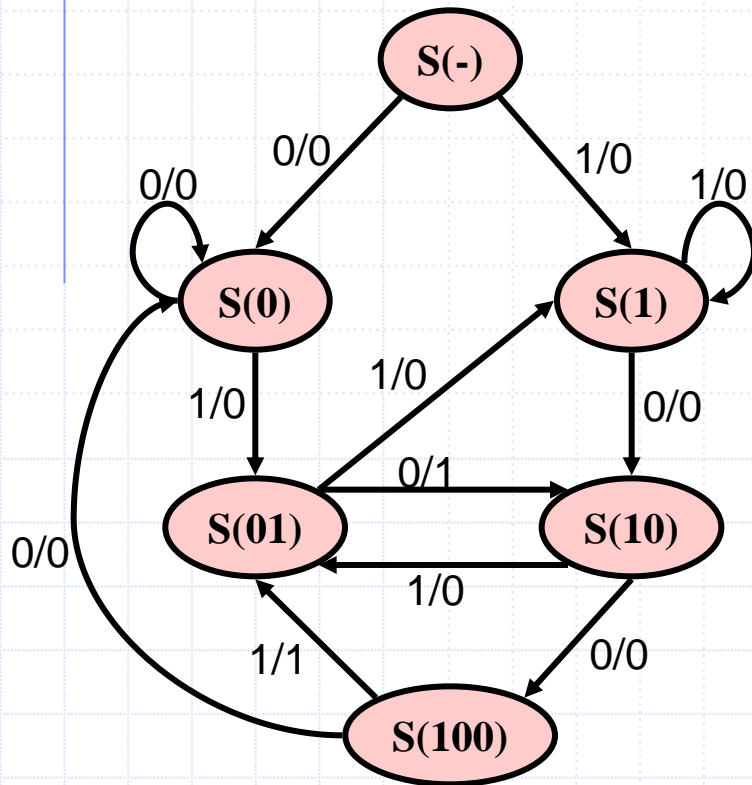


Mealy Sequence Detector

Target Sequences:

010

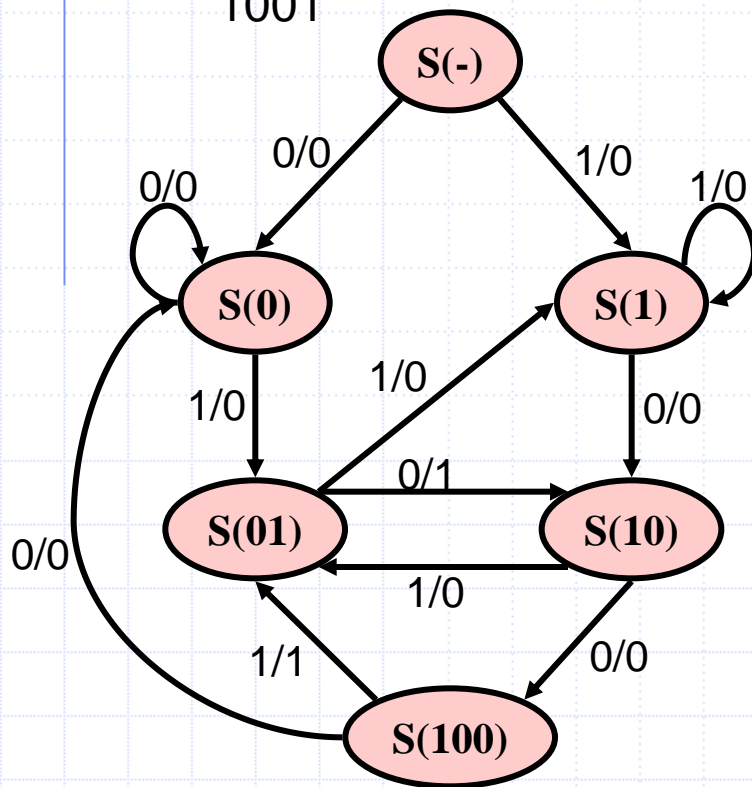
1001



Mealy Sequence Detector

Target Sequences:

010
1001



Present State	Next State		Output	
	X=0	X=1	X=0	X=1
S(-)	S(0)	S(1)	0	0
S(0)	S(0)	S(01)	0	0
S(1)	S(10)	S(1)	0	0
S(01)	S(10)	S(1)	1	0
S(10)	S(100)	S(01)	0	0
S(100)	S(0)	S(01)	0	1

Mealy Sequence Detector

Present State	Next State		Output	
	X=0	X=1	X=0	X=1
S(-)	S(0)	S(1)	0	0
S(0)	S(0)	S(01)	0	0
S(1)	S(10)	S(1)	0	0
S(01)	S(10)	S(1)	1	0
S(10)	S(100)	S(01)	0	0
S(100)	S(0)	S(01)	0	1

State	Code
	$Q_2Q_1Q_0$
S(-)	000
S(0)	001
S(1)	010
S(01)	011
S(10)	100
S(100)	101

Mealy Sequence Detector

Present State	Next State		Output	
	X=0	X=1	X=0	X=1
000	S(0)	S(1)	0	0
S(0)	S(0)	S(01)	0	0
S(1)	S(10)	S(1)	0	0
S(01)	S(10)	S(1)	1	0
S(10)	S(100)	S(01)	0	0
S(100)	S(0)	S(01)	0	1

State	Code $Q_2Q_1Q_0$
S(-)	000
S(0)	001
S(1)	010
S(01)	011
S(10)	100
S(100)	101

Mealy Sequence Detector

Present State	Next State		Output	
	X=0	X=1	X=0	X=1
000	001	S(1)	0	0
001	001	S(01)	0	0
S(1)	S(10)	S(1)	0	0
S(01)	S(10)	S(1)	1	0
S(10)	S(100)	S(01)	0	0
S(100)	001	S(01)	0	1

State	Code $Q_2Q_1Q_0$
S(-)	000
S(0)	001
S(1)	010
S(01)	011
S(10)	100
S(100)	101

Mealy Sequence Detector

Present State	Next State		Output	
	X=0	X=1	X=0	X=1
000	001	010	0	0
001	001	S(01)	0	0
010	S(10)	010	0	0
S(01)	S(10)	010	1	0
S(10)	S(100)	S(01)	0	0
S(100)	001	S(01)	0	1

State	Code
	$Q_2Q_1Q_0$
S(-)	000
S(0)	001
S(1)	010
S(01)	011
S(10)	100
S(100)	101

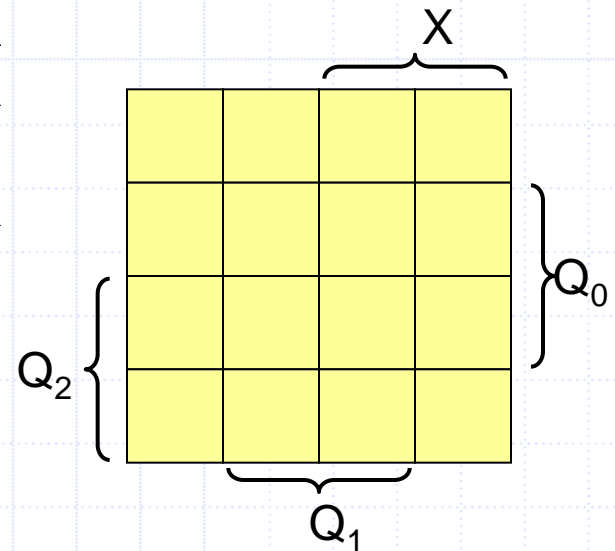
Mealy Sequence Detector

Present State	Next State		Output	
	X=0	X=1	X=0	X=1
000	001	010	0	0
001	001	011	0	0
010	S(10)	010	0	0
011	S(10)	010	1	0
S(10)	S(100)	011	0	0
S(100)	001	011	0	1

State	Code $Q_2Q_1Q_0$
S(-)	000
S(0)	001
S(1)	010
S(01)	011
S(10)	100
S(100)	101

Mealy Sequence Detector

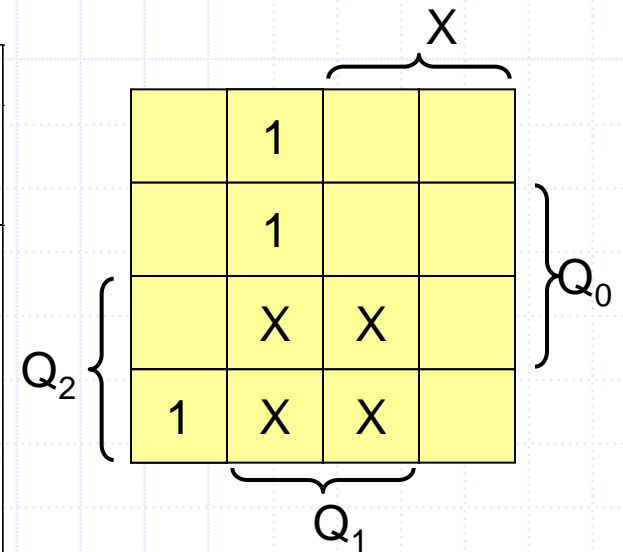
Present State $Q_2Q_1Q_0$	Next State		Output	
	$X=0$	$X=1$	$X=0$	$X=1$
	$Q_2^+Q_1^+Q_0^+$	$Q_2^+Q_1^+Q_0^+$		
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1



Which Karnaugh map cells are don't cares?

Mealy Sequence Detector

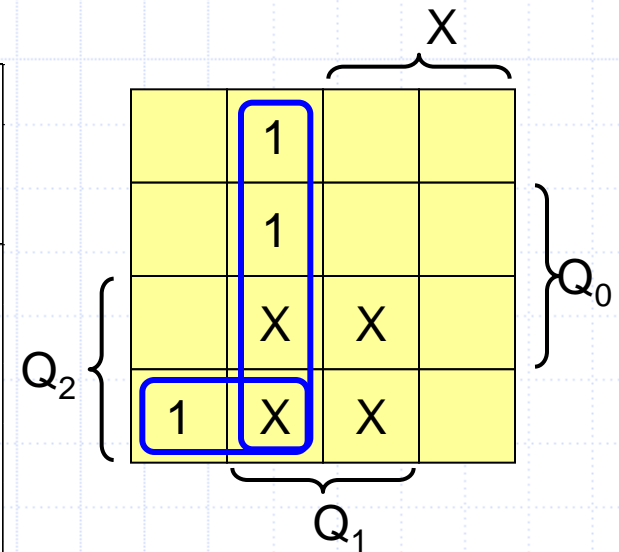
Present State $Q_2Q_1Q_0$	Next State		Output	
	X=0 $Q_2^+Q_1^+Q_0^+$	X=1 $Q_2^+Q_1^+Q_0^+$	X=0	X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1



$$D_2 =$$

Mealy Sequence Detector

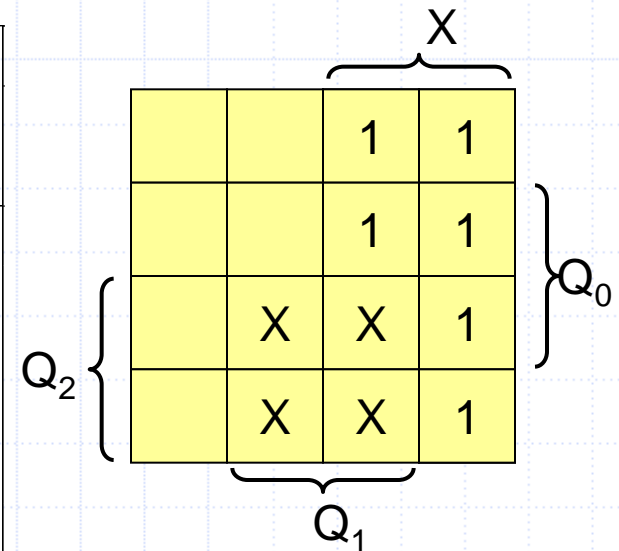
Present State $Q_2Q_1Q_0$	Next State		Output	
	X=0 $Q_2^+Q_1^+Q_0^+$	X=1 $Q_2^+Q_1^+Q_0^+$	X=0	X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1



$$D_2 = Q_1X' + Q_2Q_0'X'$$

Mealy Sequence Detector

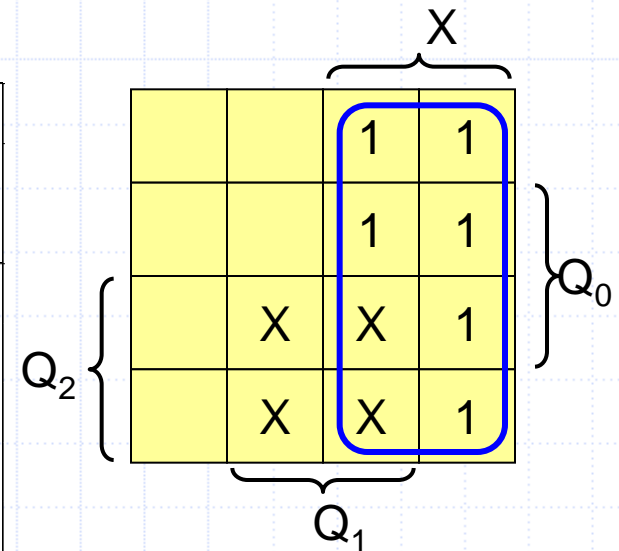
Present State $Q_2Q_1Q_0$	Next State		Output	
	X=0	X=1	X=0	X=1
	$Q_2^+Q_1^+Q_0^+$	$Q_2^+Q_1^+Q_0^+$		
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1



$$D_1 =$$

Mealy Sequence Detector

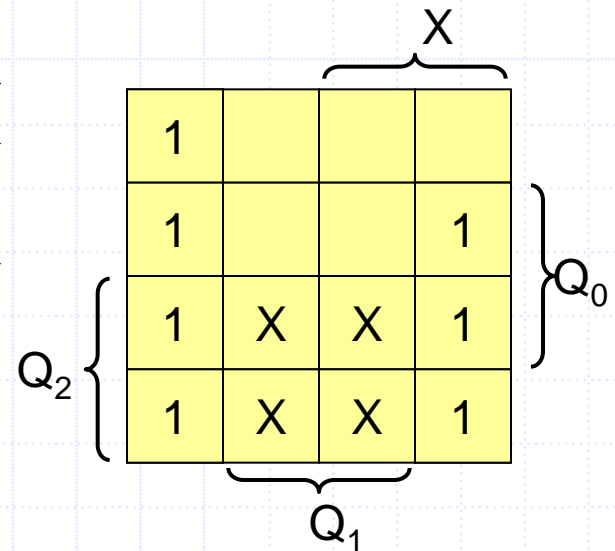
Present State $Q_2Q_1Q_0$	Next State		Output	
	X=0 $Q_2^+Q_1^+Q_0^+$	X=1 $Q_2^+Q_1^+Q_0^+$	X=0	X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1



$$D_1 = X$$

Mealy Sequence Detector

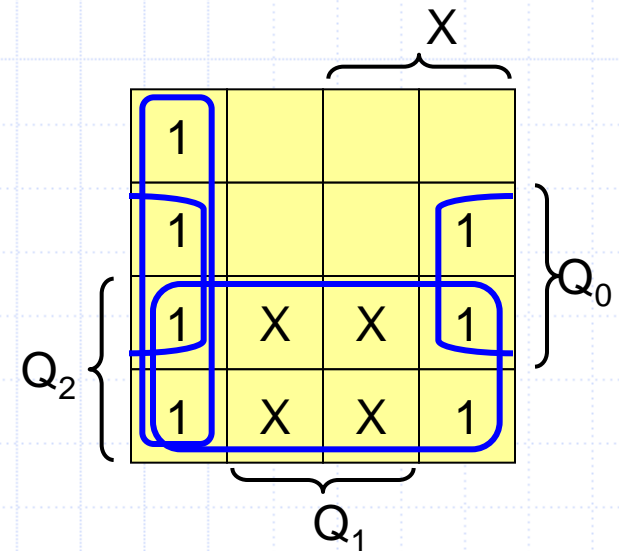
Present State $Q_2Q_1Q_0$	Next State		Output	
	X=0 $Q_2^+Q_1^+Q_0^+$	X=1 $Q_2^+Q_1^+Q_0^+$	X=0	X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1



$D_0 =$

Mealy Sequence Detector

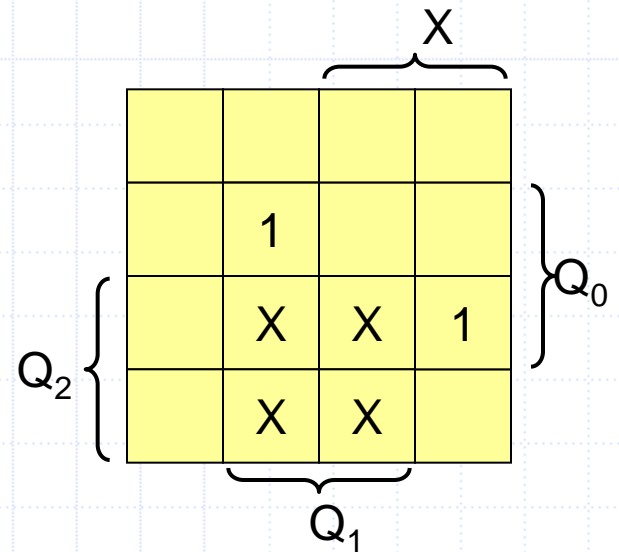
Present State $Q_2Q_1Q_0$	Next State		Output	
	$X=0$ $Q_2^+Q_1^+Q_0^+$	$X=1$ $Q_2^+Q_1^+Q_0^+$	$X=0$	$X=1$
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1



$$D_0 = Q_2 + Q_1'X' + Q_1'Q_0$$

Mealy Sequence Detector

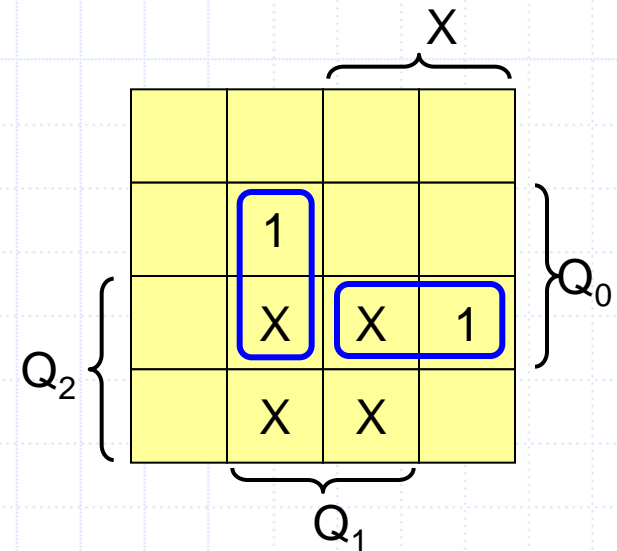
Present State $Q_2Q_1Q_0$	Next State		Output	
	X=0 $Q_2^+Q_1^+Q_0^+$	X=1 $Q_2^+Q_1^+Q_0^+$	X=0	X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1



Z =

Mealy Sequence Detector

Present State $Q_2Q_1Q_0$	Next State		Output	
	X=0 $Q_2^+Q_1^+Q_0^+$	X=1 $Q_2^+Q_1^+Q_0^+$	X=0	X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1



$$Z = Q_1Q_0X' + Q_2Q_0X$$

Mealy Sequence Detector Design Verification

Present State $Q_2Q_1Q_0$	Next State		Output	
	X=0 $Q_2^+Q_1^+Q_0^+$	X=1 $Q_2^+Q_1^+Q_0^+$	X=0	X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1
110	???	???	?	?
111	???	???	?	?

$$D_0 = Q_2 + Q_1'X' + Q_1'Q_0$$

$$D_1 = X$$

$$D_2 = Q_1X' + Q_2Q_0'X'$$

$$Z = Q_1Q_0X' + Q_2Q_0X$$

Mealy Sequence Detector Design Verification

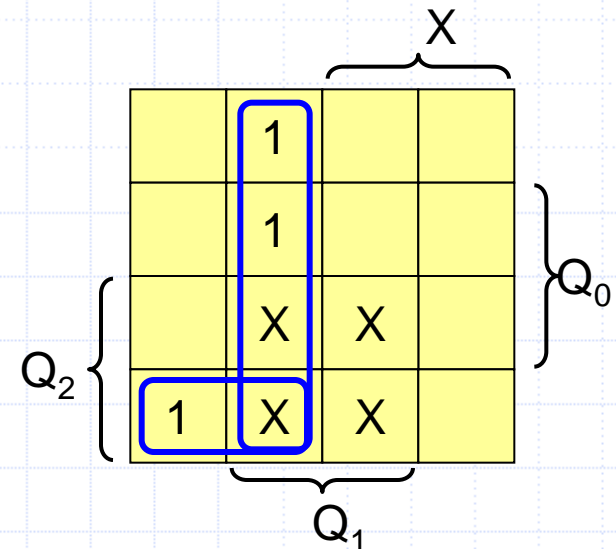
Present State $Q_2Q_1Q_0$	Next State		Output	
	X=0 $Q_2^+Q_1^+Q_0^+$	X=1 $Q_2^+Q_1^+Q_0^+$	X=0	X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1
110	1??	0??	?	?
111	1??	0??	?	?

$$D_0 = Q_2 + Q_1'X' + Q_1'Q_0$$

$$D_1 = X$$

$$D_2 = Q_1X' + Q_2Q_0'X'$$

$$X = Q_1Q_0X' + Q_2Q_0X$$



Mealy Sequence Detector Design Verification

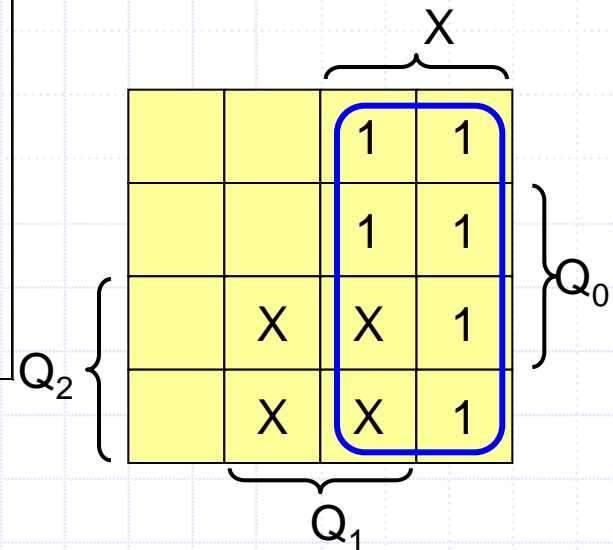
Present State $Q_2Q_1Q_0$	Next State		Output	
	X=0 $Q_2^+Q_1^+Q_0^+$	X=1 $Q_2^+Q_1^+Q_0^+$	X=0	X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1
110	10?	01?	?	?
111	10?	01?	?	?

$$D_0 = Q_2 + Q_1'X' + Q_1'Q_0$$

$$D_1 = X$$

$$D_2 = Q_1X' + Q_2Q_0'X'$$

$$X = Q_1Q_0X' + Q_2Q_0X$$



Mealy Sequence Detector Design Verification

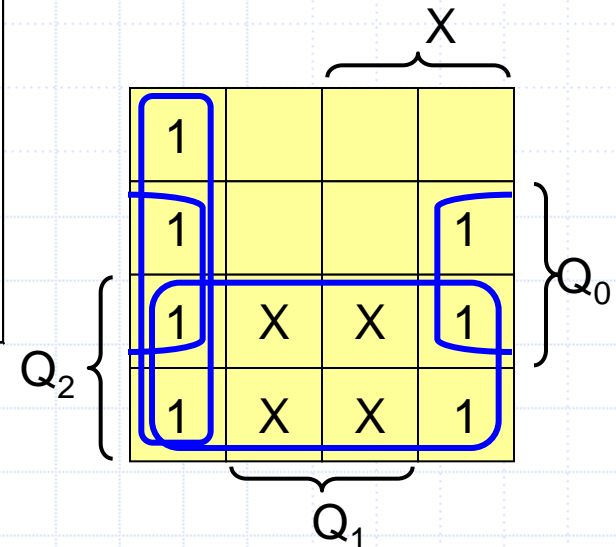
Present State $Q_2Q_1Q_0$	Next State		Output	
	X=0 $Q_2^+Q_1^+Q_0^+$	X=1 $Q_2^+Q_1^+Q_0^+$	X=0	X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1
110	101	011	?	?
111	101	011	?	?

$$D_0 = Q_2 + Q_1'X' + Q_1'Q_0$$

$$D_1 = X$$

$$D_2 = Q_1X' + Q_2Q_0'X'$$

$$X = Q_1Q_0X' + Q_2Q_0X$$



Mealy Sequence Detector Design Verification

Present State $Q_2Q_1Q_0$	Next State		Output	
	X=0 $Q_2^+Q_1^+Q_0^+$	X=1 $Q_2^+Q_1^+Q_0^+$	X=0	X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1
110	101	011	0	0
111	101	011	1	1

$$D_0 = Q_2 + Q_1'X' + Q_1'Q_0$$

$$D_1 = X$$

$$D_2 = Q_1X' + Q_2Q_0'X'$$

$$X = Q_1Q_0X' + Q_2Q_0X$$

