درس مدارهاي منطقي ديجيتال

مرجع: مدارهاي منطقي ديجيتال

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:

$$\ldots \quad 10^4 \quad 10^3 \quad 10^2 \quad 10^1 \quad 10^0$$

$$1 \quad 7 \quad 3$$

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

$$= 100 + 70 + 3$$

$$= 173$$

سیستم دودویی اعداد(binary):

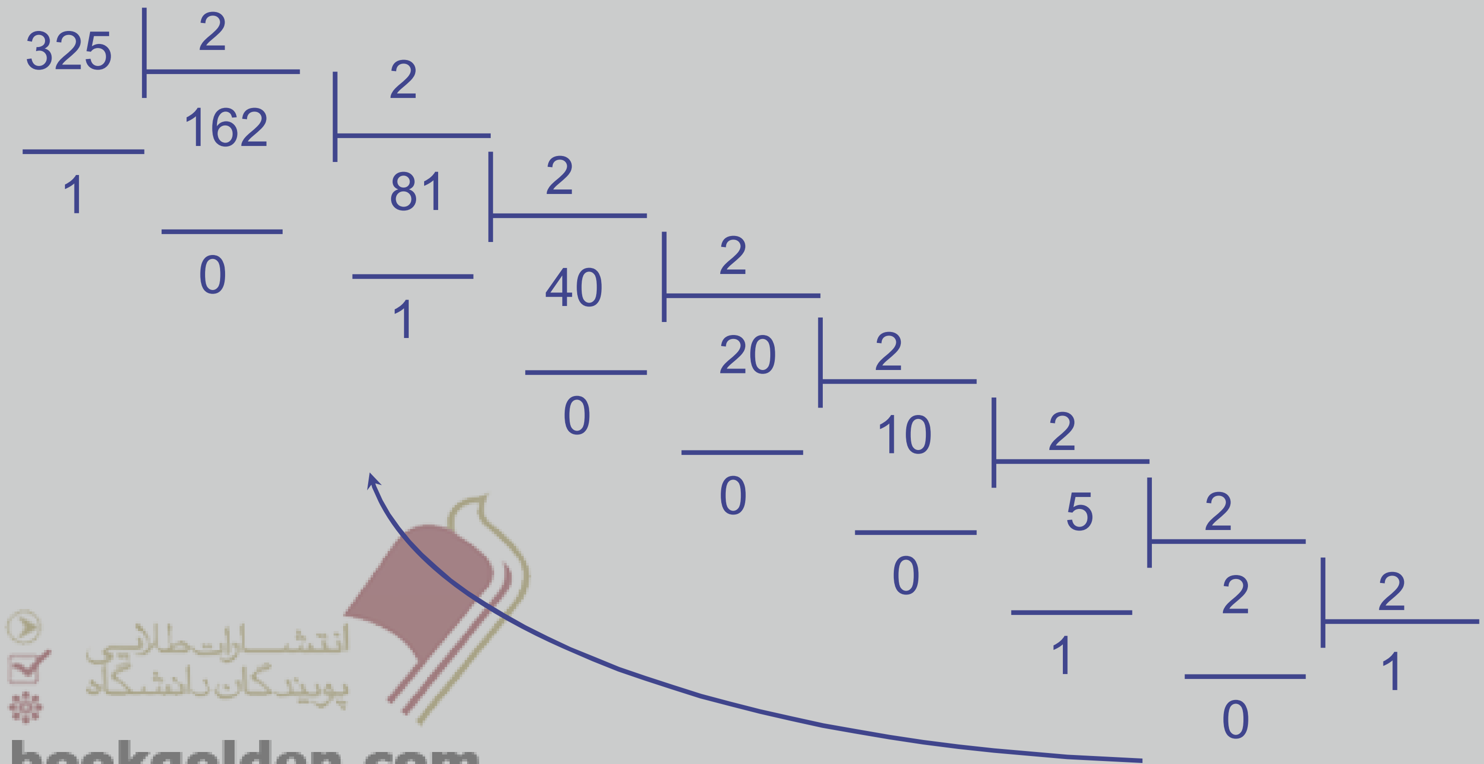
- آسان برای کامپیوتر ها, ناملموس برای ما
- از ارقام دودویی((binary digits (bits))، به جای ارقام ده دهی استفاده می کند.
- n بیت داده شده می تواند نشانگر 2^n عدد باشد.
- با ده انگشت می شود تا 1023 شمرد!
- در این سیستم نیز از موقعیت، وزن را تعیین می کند.

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 :

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

25 = 1 1 0 0 1

16 8 1

تبدیل از مبنای دو به مبنای ده

$$(1\,0\,1\,1\,1\,0)_{21} = 0 \times 1 + 1 \times 2 + 1 \times 4 + 1 \times 8 + 0 \times 16 + 1 \times 32 = (46)_{10}$$

$$\begin{array}{cccccc} 1 & 0 & 1 & 1 & 1 & 0 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \end{array}$$

اعداد اعشاری

$25.43 \rightarrow 11001.01101 \ldots$

$0.43 * 2 = 0.86$
$0.86 * 2 = 1.72$
$0.72 * 2 = 1.44$
$0.44 * 2 = 0.88$
$0.88 * 2 = 1.76$
…

اعداد بدون علامت در قالب n بیتی:

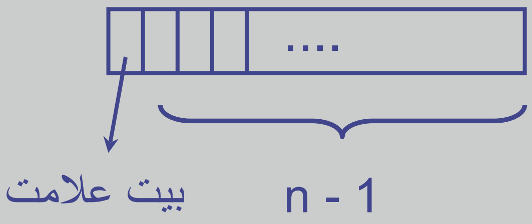
- حداقل 0
- حداکثر $2^n - 1$

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

اعداد علامت دار

1- سیستم علامت مقدار

0 : +
1 : -



2 – سیستم متمم دو

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

$$A - B = A + \underbrace{\overline{B} + 1}_{\text{متمم دو}}$$

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 قالبn 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 متمم دو	سیستم متمم یک	سیستم علامت مقدار
000 = +0	000 = +0	000 = +0
001 = +1	001 = +1	001 = +1
010 = +2	010 = +2	010 = +2
011 = +3	011 = +3	011 = +3
100 = -0	100 = -3	100 = -4
101 = -1	101 = -2	101 = -3
110 = -2	110 = -1	110 = -2
111 = -3	111 = -0	111 = -1

متمم 2 :

1- عدد بدون علامت به صورت باینری نوشته شود. $(49)_{10} = (110001)_2$

2 – قالب ریزی 0110001

3 – اگر عدد مثبت بود، کار تمام است، اما اگر عدد منفی است لازم است متمم دو شود.

جمع و تفریق اعداد علامت دار :

```
 - 49          1 0 0 1 1 1 1
 + 23          0 0 1 0 1 1 1
 ------        --------------
 - 26          1 1 0 0 1 1 0
```

- اگر در جمع خطای سرریز رخ داد، باید خمع 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 اعدا 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 :

$$\begin{array}{r} 25.50 \\ -\ 38.75 \\ \hline \end{array} \longrightarrow \begin{array}{r} 0011001.1000 \\ 1011001.0100 \\ \hline 1110010.1100 \end{array}$$

$$\underbrace{1110010}_{-13}.1100 \quad 0.5 \quad 0.25$$

مبنای 4، 8، 16

$25 \rightarrow (11001)_2$

$0\,1\,1\,0\,0\,1 \rightarrow (121)_4$

$0\,1\,1\,0\,0\,1 \rightarrow (31)_8$

$0\,0\,0\,1\,1\,0\,0\,1 \rightarrow (19)_{16}$

ضرب و تقسیم اعداد باینری :

ضرب به روش معمولی :

```
     1110
  *
     0101
  ---------
    1110
   0000
  1110
 0000
---------
 1000110
```

ضرب به روش جمع های متوالی :

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

کدینگ اطلاعات :

هدف : ورورد به سیستم دیجیتال

معیار ها :

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

Binary Coded Decimal

	BCD
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 کاراکتر ها، 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)

تعداد کلیه سیستم های خود مکمل :

0 1 2 3 4

8 * 7 * 6 * 5 * 4 = 6720

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 :

1- چند کد وزنی و خود مکمل با ارزش های 1، 2، 2، 4 وجود دارد؟

2- چند کد وزنی و خود مکمل با ارزش 2421 وجود دارد؟

3 – ارزش های دیگری غیر از این ارزش بگویید.

4 – ارزش منفی هم در اعداد قرار دهید.

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

6 – 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

اعشاری < 1

25

1	1	0	0	1	.

صحیح >= 1

نما

$43.85 \rightarrow \underbrace{0.4385} * 10^{2}$

0 < مانتیس < 1

علامت نما	نما	مانتیس

$1\,0\,1\,0\,1\,1\,.\,1\,1\,0\,1 = 0\,.\,\underbrace{1\,0\,1\,0\,1\,1\,1\,1\,0\,1} * 2^{+6}$

0.5 < 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 :

توان های 2 ⟵ بیت های کنترلی

داده خام : 1 0 1 1

	0	1	1	0	0	1	1
بیت های کنترلی	1	2	3	4	5	6	7

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 :

P_1 P_2 P_4

0 1 0 0 1 1 1

B_3 B_5 B_6 B_7

داده ارسالی : 0 1 0 0 1 0 1

داده دریافتی : 0 1 0 0 1 1 1

$$\begin{array}{l} P_1 = 0 \\ P_2 = 0 \\ P_4 = 1 \\ \hline \quad 6 \end{array}$$

$6 \longrightarrow B_6$ رخداد خطا

- یک بیت خطا قابل تصحیح
- دو بیت خطا قابل تشخیص

فصل 2

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.
- درسطح 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(2)

مثال:

سطح ولتاژ بالا ← عبارت درست است.

سطح ولتاژ پائین ← عبارت نادرست است.

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

اصول جبر بول (1)

اصول اساسی:

اصل 1:

تعریف:برای هر a و b که متعلق به مجموعه ی k هستند، $a+b$ و $a.b$ نیز به مجموعه ی k تعلق دارند.

($a.b$، And و $a+b$، Or نامیده می شود).

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

اصول جبر بول (2)

اصل 2:

موجودیت عناصر 0 و 1:

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

$$x + 0 = x$$

$$x \cdot 1 = x$$

اصول جبر بول (3)

اصل 3:

$$x + y = y + x$$

خاصیت عناصر + و . :

$$x \, . \, y = y \, . \, x$$

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 (4)

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 (5)

اصل 4:

خاصیت شرکت پذیری اعمال **+** و **.**

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

$$x . (y . z) = (x . y) . z$$

اصول جبر بول (6)

اصل 5:

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 + (2)

=

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 (1)

1.خاصیت خود توانی:

$$a + a = a$$

$$a \, . \, a = a$$

2.عناصر بی اثر در . و + :

$$a \, . \, 1 = a$$

$$a + 0 = a$$

اصول اساسی جبر بول (2)

3.متمّمِ متمّم:

$$a'' = a$$

4.قانون جذب:

$$a + a \, . \, b = a$$

$$a \, .(a + b) = a$$

اصول اساسی جبر بول (3)

5. قانون 5

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

b) $a(a' + b) = a\,b$

مثال:

- $B + AB'C'D = B + AC'D$ [ق5(a)]
- $(X + Y)((X + Y)' + Z) = (X + Y)Z$ [ق5(b)]

6. قانون 6

a) $ab + ab' = a$

b) $(a + b)(a + b') = a$

اصول اساسی جبر بول (3)

مثال:

- $ABC + AB'C = AC$ [ق6(a)]
- $(W' + X' + Y' + Z')(W' + X' + Y' + Z)(W' + X' + Y + Z')(W' + X' + Y + Z)$

$= (W' + X' + Y')(W' + X' + Y + Z')(W' + X' + Y + Z)$ [ق6(b)]

$= (W' + X' + Y')(W' + X' + Y)$ [ق6(b)]

$= (W' + X')$ [ق6(b)]

اصول اساسی جبر بول (3)

7.قانون 7

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

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

مثال:

- $wy' + wx'y + wxyz + wxz'$

$= wy' + wx'y + wxy + wxz'$ [ق7(a)]

$= wy' + wy + wxz'$ [ق7(a)]

$= w + wxz'$ [ق7(a)]

$= w$ [ق7(a)]

قوانین دمرگان(1)

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

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

این قانون می تواند به صورت زیر تعمیم پیدا کند:

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

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

قوانین دمرگان(2)

مثال:

- $(a + bc)'$

$$= (a + (bc))'$$

$$= a'(bc)'$$

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

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

قوانین دمرگان(3)

مثال های بیشتری از قوانین دمرگان:

- $(a(b + z(x + a')))' = a' + (b + z(x + a'))'$ [د(b)]
  - $= a' + b' (z(x + a'))'$ [د(a)]
  - $= a' + b' (z' + (x + a')')$ [د(b)]
  - $= a' + b' (z' + x'(a')')$ [د(a)]
  - $= a' + b' (z' + x'a)$ [متمّمِ متمّم]
  - $= a' + b' (z' + x')$ [ق5(a)]

- $(a(b + c) + a'b)' = (ab + ac + a'b)'$ [اصل5(b)]
  - $= (b + ac)'$ [ق6(a)]
  - $= b'(ac)'$ [د(a)]
  - $= b'(a' + c')$ [د(b)]

اصول اساسی جبر بول (4)

8.قانون8

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

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

مثال:

- $AB + A'CD + BCD = AB + A'CD$ [ق9(a)]
- $(a + b')(a' + c)(b' + c) = (a + b')(a' + c)$ [ق9(b)]
- $ABC + A'D + B'D + CD$

$= ABC + (A' + B')D + CD$ [اصل5(b)]

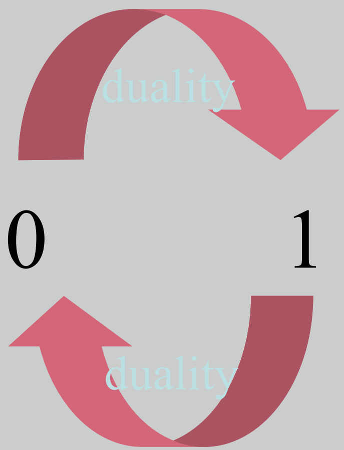
$= ABC + (AB)'D + CD$ [د(b)]

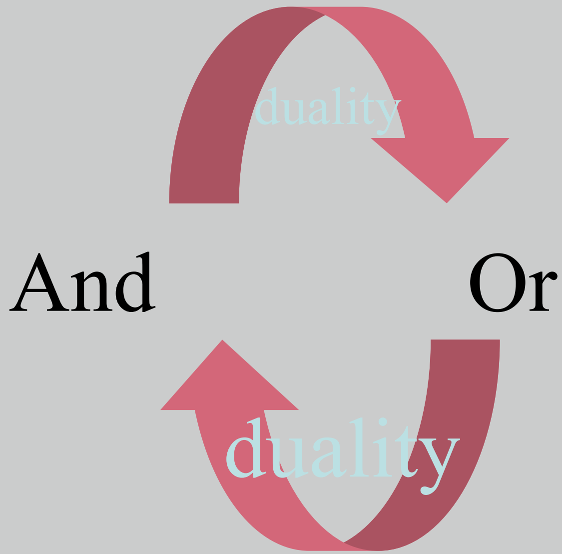
$= ABC + (AB)'D$ [ق9(a)]

$= ABC + (A' + B')D$ [د(b)]

$= ABC + A'D + B'D$ [اصل5(b)]

دوگان (duality)





مثال:

$x+y'z \xleftrightarrow{\text{دوگان}} x.(y'+z)$

مینترم (SOP) و ماکسترم ها (POS)(1)

x	y	z	x+y+z	Minterm		Maxterm	
0	0	0	0	x’.y’.z’	m_0	x+y+z	M_0
0	0	1	1	x’.y’.z	m_1	x+y+z’	M_1
0	1	0	1	x’.y.z’	m_2	x+y’+z	M_2
0	1	1	1	x’.y.z	m_3	x+y’+z’	M_3
1	0	0	1	x.y’.z’	m_4	x’+y+z	M_4
1	0	1	1	x.y’.z	m_5	x’+y+z’	M_5
1	1	0	1	x.y.z’	m_6	x’+y’+z	M_6
1	1	1	1	x.y.z	m_7	x’+y’+z’	M_7

مینترم (SOP) و ماکسترم ها (POS)(2)

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)(2)

مثال: تابع زیر را به صورت مینترمی بنویسید.

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

1. رسم جدول درستی

x	y	F
0	0	0
0	1	0
1	0	0
1	1	1

2. تعیین مینترم ها

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

مینترم (SOP) و ماکسترم ها (POS)(3)

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

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

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

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

قضیه گسترش شانون:

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

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

مثال:

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

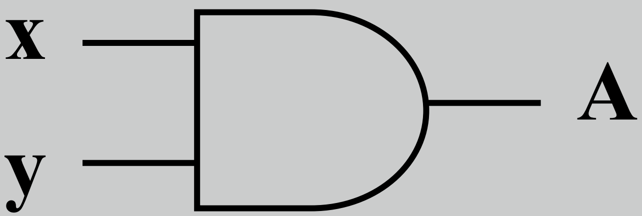
Xor & Xnor

- $x \oplus y = x \cdot y' + x' \cdot y$
- $x \odot y = x' \cdot y' + x \cdot y$

x	y	x . 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) (1)

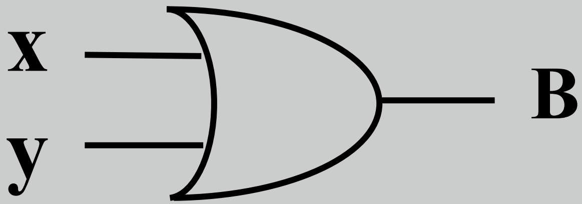
- And:



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

گیت ها(دریچه ها) (1)

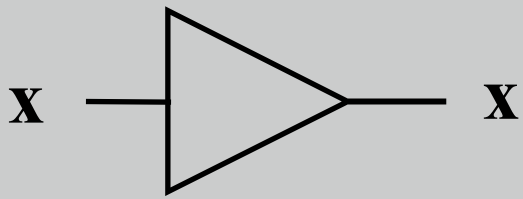
- Or:



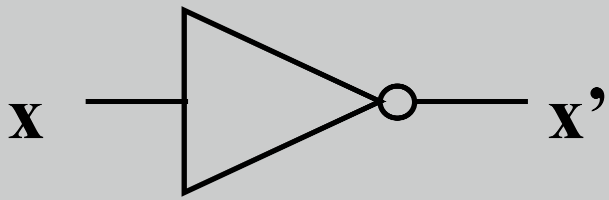
x	y	A = x + y
0	0	0
0	1	1
1	0	1
1	1	1

گیت ها (2)

تقویت کننده:



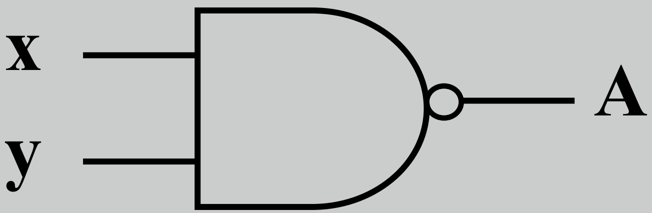
متمّم:



x	x’
0	1
1	0

گیت ها (3)

- Nand:

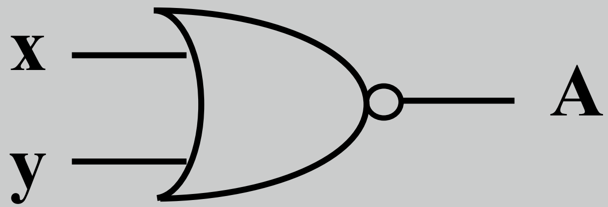


x	y	A
0	0	0
0	1	0
1	0	0
1	1	1

x

گیت ها (3)

- Nor:

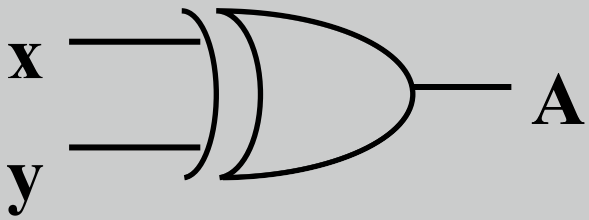


x	y	A
0	0	1
0	1	0
1	0	0
1	1	0

x

گیت ها (4)

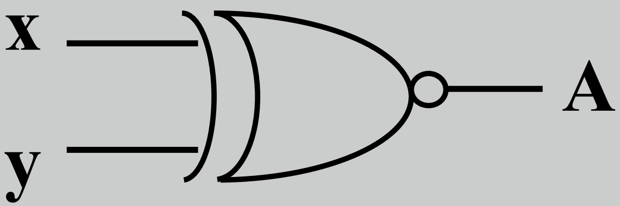
- Xor:



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

گیت ها (4)

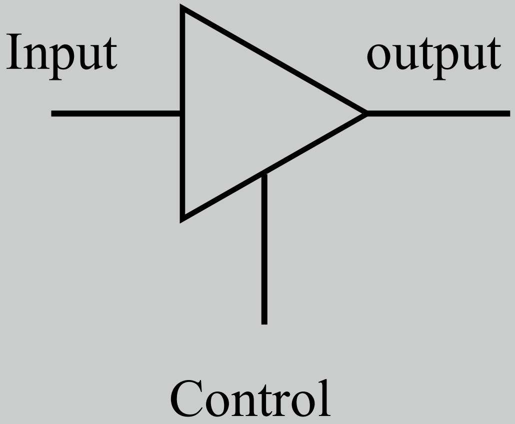
- Xnor:



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

گیت یا بافر 3 وضعیتی (1)

این گیت ها دارای یک دریچه ورودی،
یک خروجی و یک کلید کنترل است
که هر گاه کلید کنترل 1 گردد؛
ورودی بر روی خروجی قرار میگیرد.



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

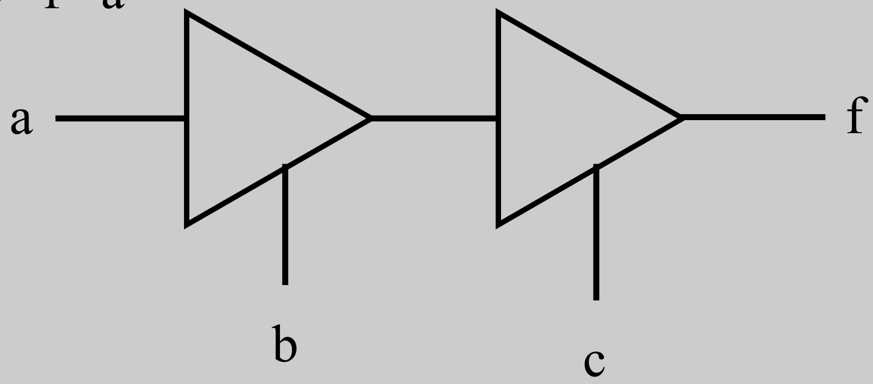
گیت یا بافر 3 وضعیتی (2)

اتصال سری:

b=0 $\xrightarrow{so}$ Off

b=1 $\xrightarrow{if}$ c = 0 $\xrightarrow{so}$ Off

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

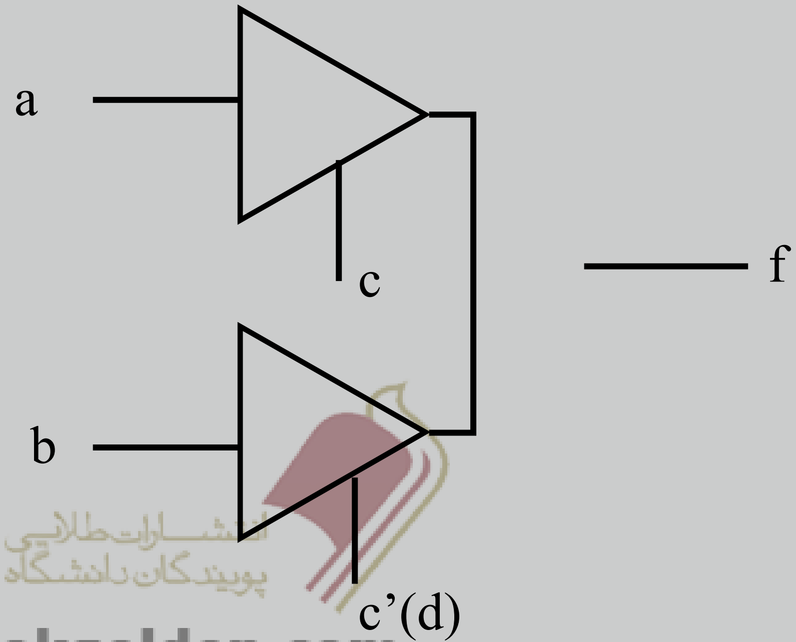


گیت یا بافر 3 وضعیتی (3)

اتصال موازی:

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

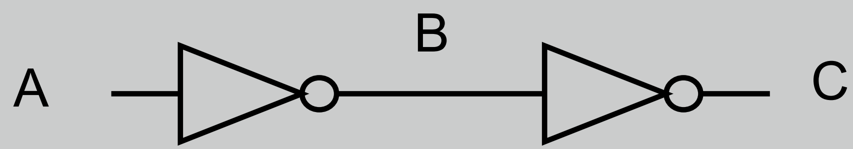
$$c = 1 \xrightarrow{\text{so}} f = a$$

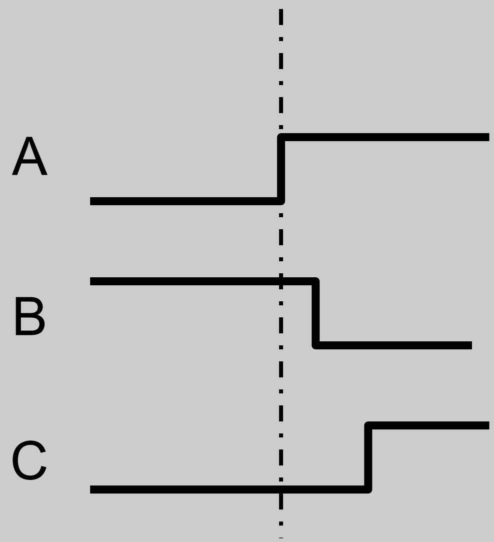


$c.d = 0$

تأخیر در انتشار(1)

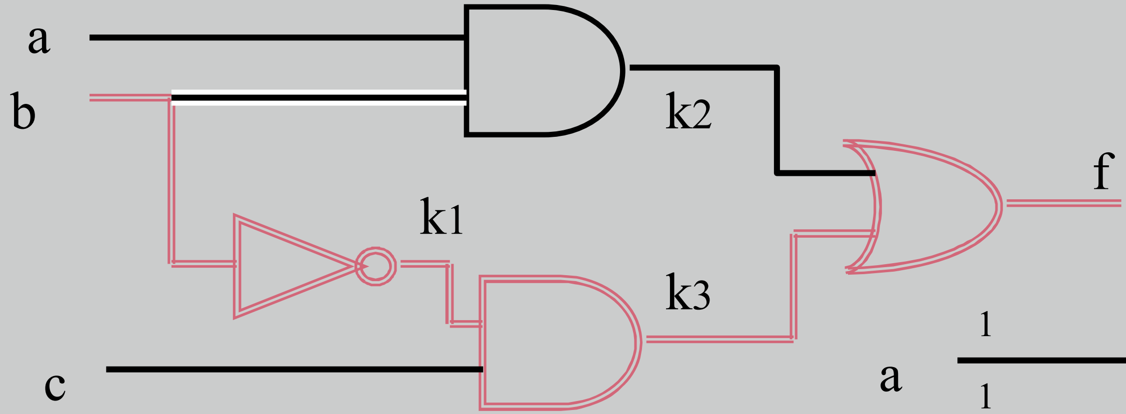
- Real implementations are not quite so perfect
- Computation actually takes some time
- Communication actually takes some time



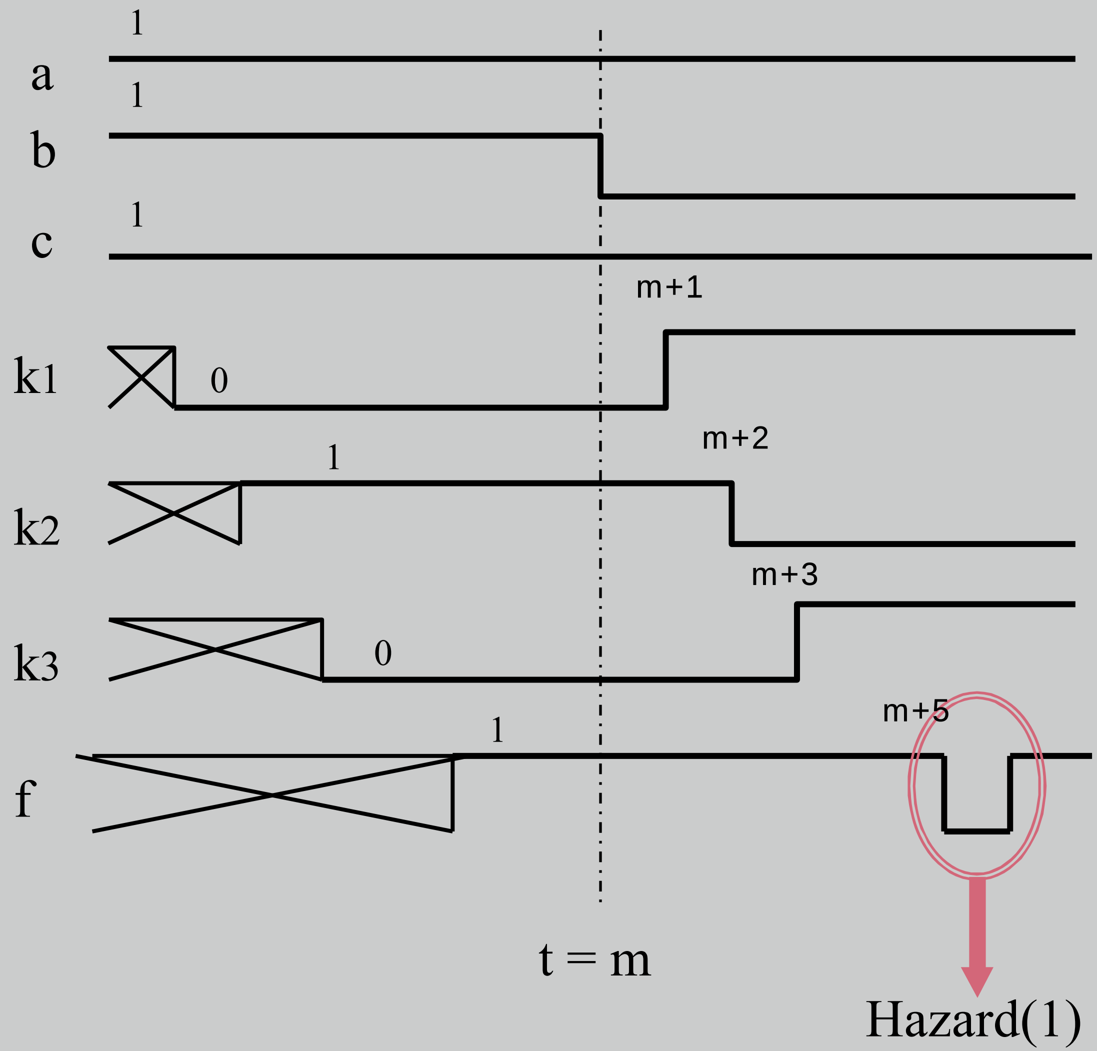


Timing Diagram

تأخیر(2)



مثال:



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

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

کد گِری(1)

در این کد،هر کدام از کد ها تنها در یک بیت با کد قبلی متفاوت است و این روند چرخشی است؛یعنی آخرین کد و اولین کد نیز تنها در 1 بیت متفاوتند.

کد گری(2)

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

BCD 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

نحوه تولیدکدگری(3)

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

فصل 3

خصوصیات توابع سوییچی

جدول کارنا

برای ساده سازی توابع با حداکثر 6 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 3 ورودی

f(x,y,z)

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

جدول کارنا برای 4 ورودی

$$f(\underbrace{x,y}_{},\underbrace{z,t}_{})$$

xy \ 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 5 ورودی (1)

$f(\underbrace{x,y},\underbrace{z,t,e})$

xy \ zte	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 5 ورودی (2)

به جای 1 جدول 32 خانه ای میتوان از 2 جدول 16 خانه ای استفاده کرد.

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

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 5 ورودی (3)

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

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

e=1

xy \ zt	00	01	11	10
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

1. رسم جدول کارنا با توجه به سایزها
2. آوردن مینترم ها داخل جدول کارنا
3. تعیین cube
4. تبدیل cube ها به شکل جبری

اصول ساده سازی کارنا

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

1. cube قابل بزرگتر شدن نباشد.
2. حداقل یک 1 در cube موجود 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 adja-cencies.
- 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

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

xy \ zte	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) (1)

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

از این حالات به عنوان یک مؤلفه ی موثر در ساده سازی به خوبی میتوان استفاده کرد؛ به این صورت که اگر 1 بودن برخی از این حالات باعث بزرگتر شدن ها و ساده سازی بیشتر شود، ما آنها را 1 فرض میکنیم و اگر نه، به نفع ماست که آنها را 0 فرض کنیم.

توابع نا کامل (با don’t-care) (2)

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

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

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

انواع شکل مدارات 2 طبقه(1)

می دانیم هر تابع جبری با هر شکل و اندازه ای با استفاده از یک جدول درستی قابل نمایش است؛ و به فرم 2طبقه ی Or-And یا And-Or است

حال با توجه به اینکه گیت های Nand و Nor نیز مفیداند؛ میخواهیم ببینیم چه فرم های 2 طبقه دیگری وجود دارد.

انواع شکل مدارات 2 طبقه (2)

طبقه 0	طبقه 1	طبقه 2
Not	And	And
	Or	Or
	Nand	Nand
	Nor	Nor

حالات ممکن مدارات 2 طبقه

طبقه 1	طبقه 2			
	And	Or	Nand	Nor
And		★		★
Or	★		★	
Nand	★		★	
Nor		★		★

ساده سازی cube مورب جدول کارنا

مثال:

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

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

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

روش ساده سازی کویین مک کلاسکی

(Quine-McCluskey) (1)

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 6 ورودی کاربردی ندارد زیرا کار کردن با آن ساده نیست.

روش ساده سازی کویین مک کلاسکی

(Quine-McCluskey) (2)

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

روش ساده سازی کویین مک کلاسکی

(3) (Quine-McCluskey)

مثال:

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

ab \ cd	00	01	11	10
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	
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	
13	1 1 0 1	Group 3 (three 1's)
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 combing them into a column of (n-1)-variable implicants, checking off each minterm that is combined. Repeat for each column, combing (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	
2	0010	✓
4	0100	✓
8	1000	✓
6	0110	✓
9	1001	✓
10	1010	✓
12	1100	✓
13	1101	✓
15	1111	✓

Minterms	a b c d	
2,6	0-10	PI_2
2,10	-010	PI_3
4,6	01-0	PI_4
4,12	-100	PI_5
8,9	100-	✓
8,10	10-0	PI_6
8,12	1-00	✓
9,13	1-01	✓
12,13	110-	✓
13,15	11-1	PI_7

Minterms	a b c d	
8,9,12,13	1-0-	PI_1

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 (9)

	2	4	6	8	9	10	12	13	15
PI_1				*	*		*	*	
PI_2	*		*						
PI_3	*					*			
PI_4		*	*						
PI_5		*					*		
PI_6				*		*			
PI_7								*	*

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)

$$f(a,b,c,d) = PI_1 + PI_3 + PI_4 + PI_7$$

$$= 1\text{-}0\text{-} + \text{-}010 + 01\text{-}0 + 11\text{-}1$$

$$= a.c' + b'.c.d' + a'.b.d' + a.b.d$$

ساده سازی Q-M برای سیستم های چند خروجی

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

روش کار را با یک مثال می بینیم.

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

ساده سازی Q-M برای سیستم های چند خروجی (2)

مینترم ها: 0,2,4,5,6,7,8,10,12,13,15

در ابتدا فرض میکنیم همه ی مینترم ها و don’t-care های داده شده مربوط به 1 تابع میباشد و آنها را دسته بندی میکنیم و مرحله 1و2 را به صورت گفته شده در قسمت قبل انجام میدهیم.

ساده سازی Q-M برای سیستم های چند خروجی (3)

MIN TERM	abcd	Flags	
0	0000	$\alpha\gamma$	✓
2	0010	$\alpha\beta\gamma$	PI_{10}
4	0100	β	✓
8	1000	$\beta\gamma$	PI_{11}
5	0101	$\beta\gamma$	✓
6	0110	β	✓
10	1010	$\alpha\beta$	✓
12	1100	α	PI_{12}
7	0111	$\alpha\beta\gamma$	PI_{13}
13	1101	γ	✓
15	1111	α	✓

MIN TERM	abcd	Flags	
0,2	00-0	$\alpha\gamma$	PI_2
0,8	-000	γ	PI_3
2,6	0-10	β	PI_4
2,10	-010	$\alpha\beta$	PI_5
4,5	010-	β	✓
4,6	01-0	β	✓
8,10	10-0	β	PI_6
5,7	01-1	$\beta\gamma$	PI_7
5,13	-101	γ	PI_8
6,7	011-	β	✓
7,15	-111	α	PI_9

MIN TERM	abcd	Flags	
4,5,6,7	01--	β	PI_1

ساده سازی Q-M برای سیستم های چند خروجی

		f_α			(4)	f_β			f_γ		
		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 (5)

		f_α	f_γ	
		7	7	8
PI_3	γ			*
PI_7	$\beta\gamma$		*	
PI_9	α	*		
PI_{11}	$\beta\gamma$			*
PI_{13}	$\alpha\beta\gamma$	*	*	

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

$$f_\beta = PI_1 + PI_5$$

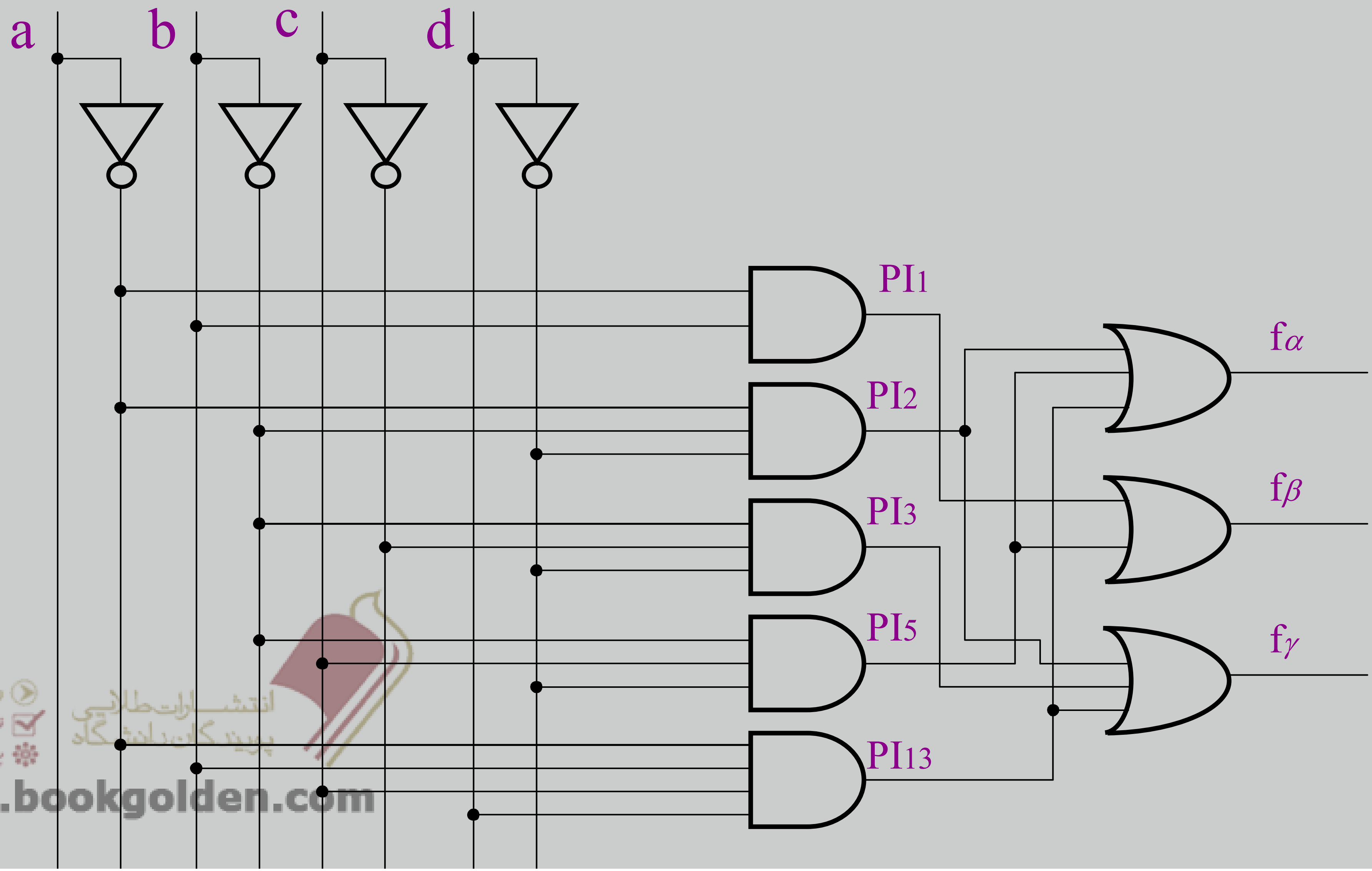
$$f_\gamma = PI_2 + PI_3 + PI_{13}$$

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

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

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

ساده سازی Q-M برای سیستم های چند خروجی (6)



فصل 4

مدارهاي منطقي تركيبي ماجولي

فهرست مطالب

- طراحي مدار
- طراحي ماجولار مدار
- Full Adder و Half Adder
- ديكدر
- اينكدر
- مالتي پلكسر(تسهيم كننده)
- دي مالتي پلكسر(پخش كننده داده ورودي)
- مقايسه گرها
- A seven segment display

طراحي مدار

- تعين تعداد بيت هاي ورودي وخروجي مدار **Interface**
- رسم جدول **Truth Table**
- بدست آوردن يك تابع براي خروجي
- ساده سازي توابع بدست آمده (كارنو/ 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

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

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

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

طرحي ماجولار مدار

اگر تعداد بيت هاي ورودي وخروجي بيش از 4 يا 5 باشد در رسم جدول صحت با مشكل برخورد مي كنيم .(پيچيدگي حافظه)

راهكار

- ❑ بدون رسم جدول درستي به خروجي مدار برسيم.(رهيافت ذهني)
- ❑ طراحي ماجولار مدار.(طراحي پيمانه اي) (از نظر زماني بهينه نيست)

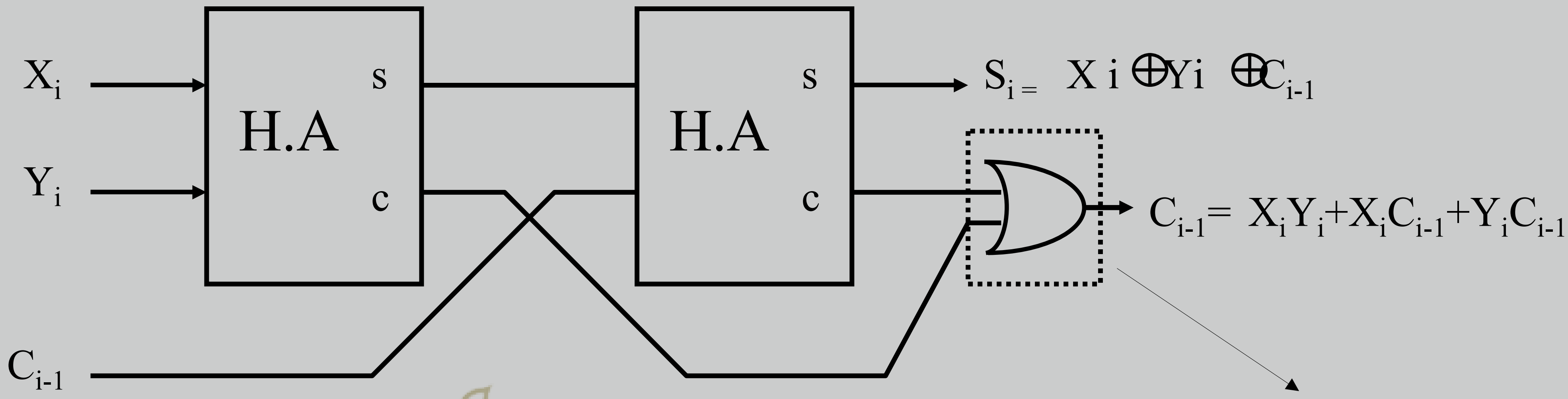
Full Adder و Half Adder (1)

- Full Adder:يك مدار تركيبي با سه ورودي و دو خروجي است كه دو بيت داده ويك رقم نقلي را با هم جمع كرده و حاصل جمع ورقم نقلي را محاسبه مي كند.

- Half Adder:يك مدار تركيبي با دو ورودي و دو خروجي است كه دو بيت دودويي را با هم جمع كرده و حاصل جمع ورقم نقلي را محاسبه مي كند.

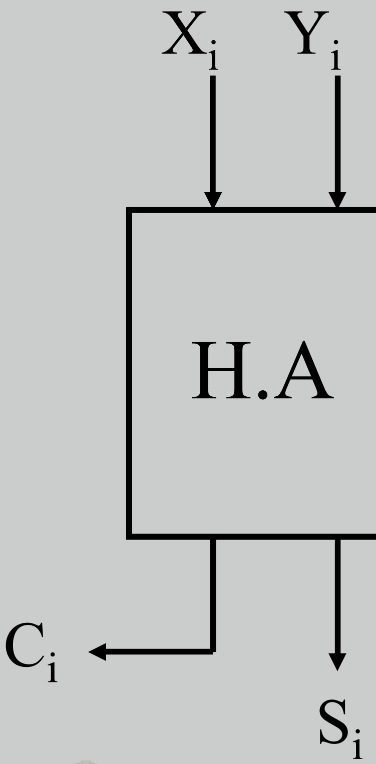
Full Adder و Half Adder (2)

يك Full adder را ميتوان توسط 2 عدد Hull adder طراحي كرد.



مي تواند توسط يك گيت XOR جايگزين شود.

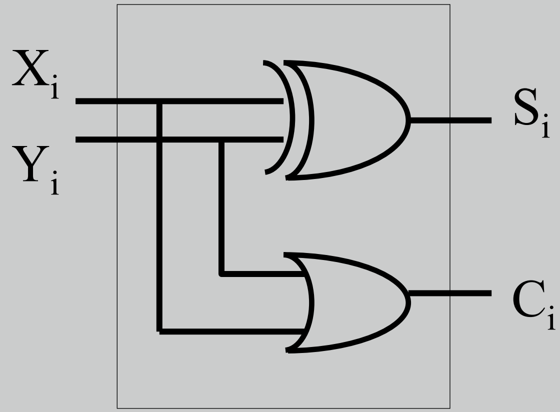
بلوك دياگرام (H.A)



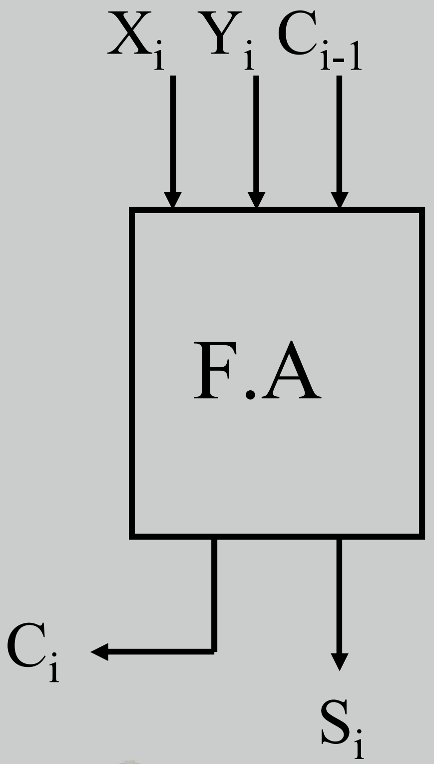
Truth Table

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)



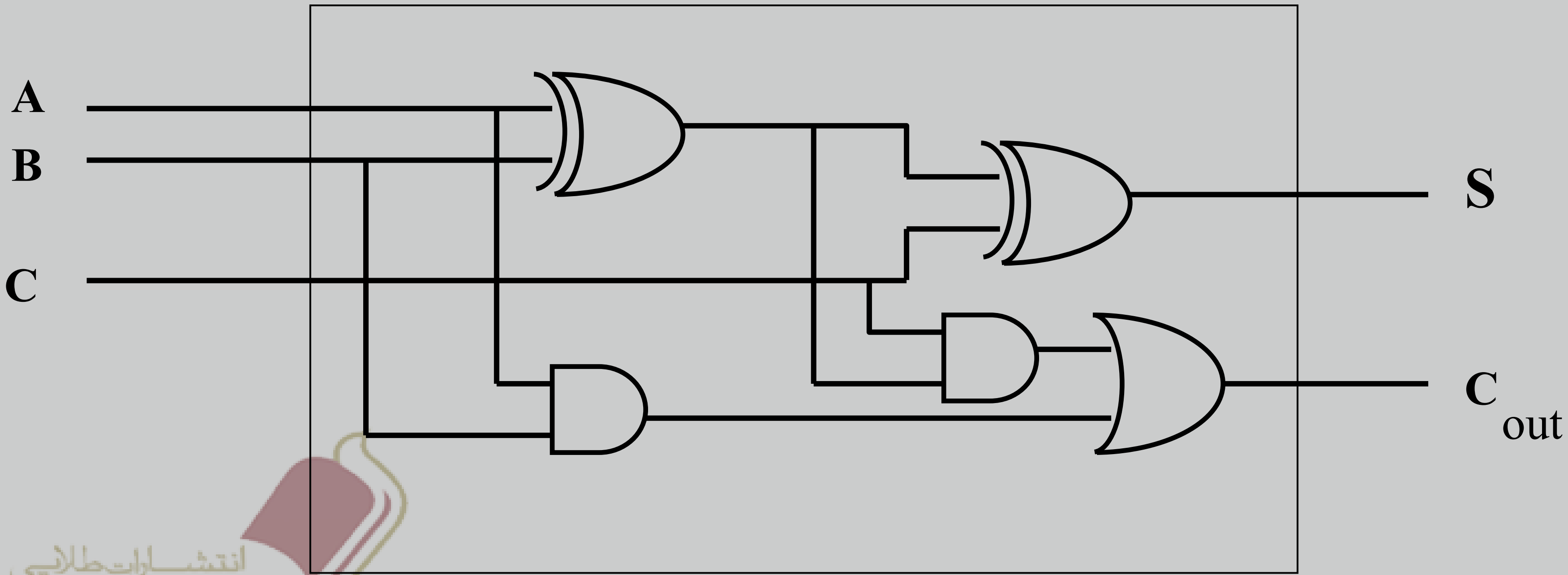
Truth Table

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

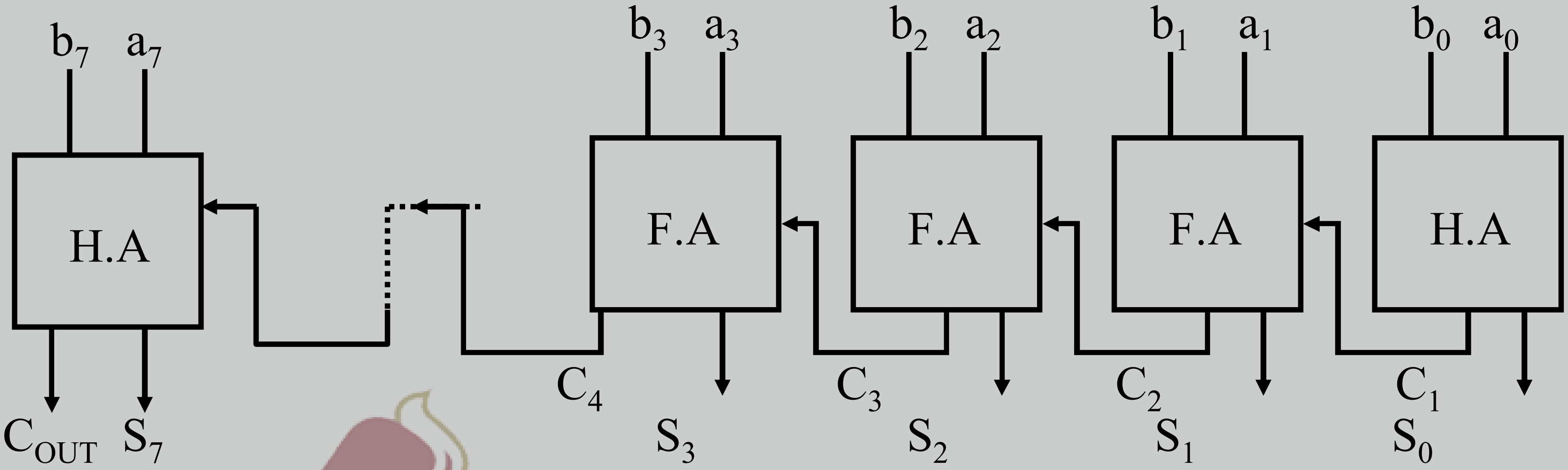
$$\Rightarrow \begin{cases} S_i = X_i \oplus Y_i \oplus C_{i-1} \\ C_i = X_iY_i + X_iC_{i-1} + Y_iC_{i-1} \end{cases}$$

دياگرام منطقي (F.A)

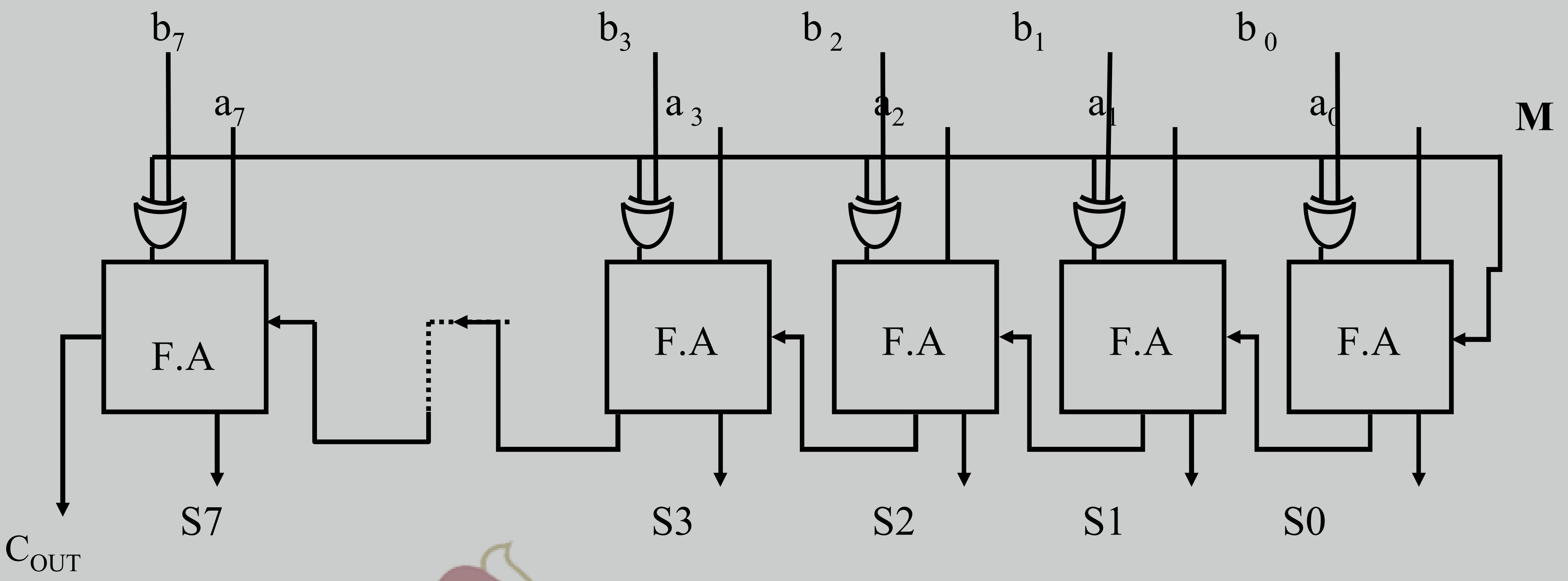
$$C_{out} = C(A \oplus B) + AB$$



Ripple Carry Adder (RCA)



Ripple Carry Adder (RCA)



If M =0 → A+B

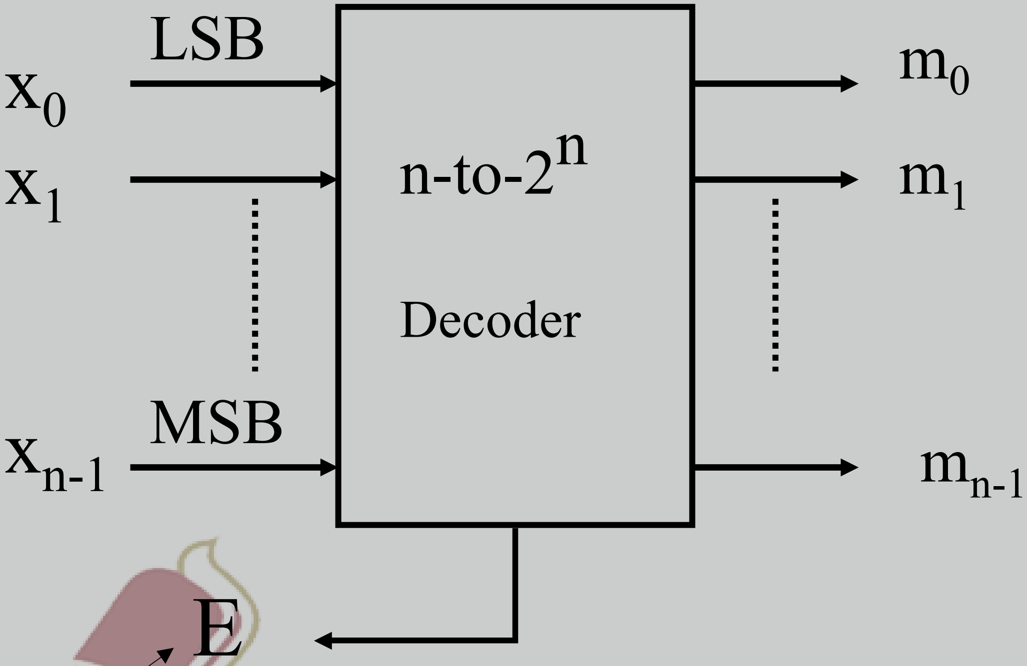
If M =1 → A-B or $(A+\bar{B}+1)$

ديكدر

- ديكدر n به 2^n يك شبكه منطقي تركيبي است با n خط ورودي و 2^n سيگنال خروجي.

- عنصري است كه مينترم ها را مي سازد.

ماجول دیکدر n به 2^n

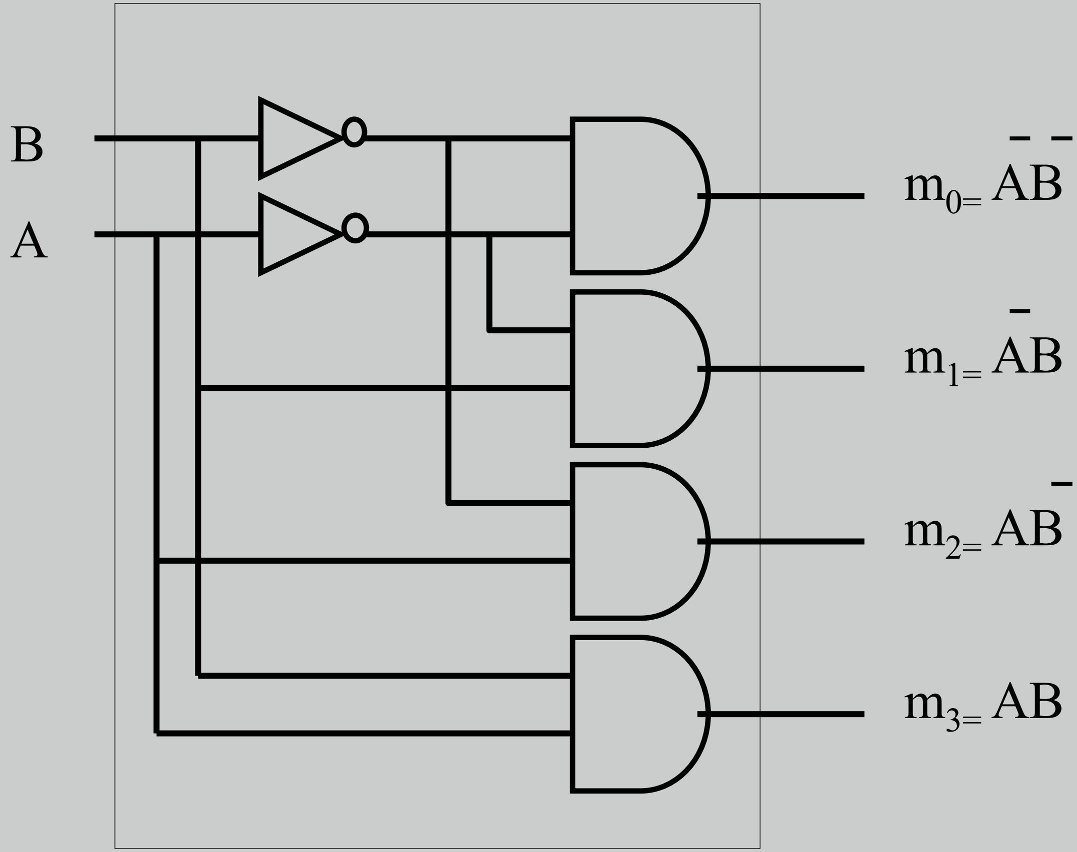


معمولا Active Low هستند.

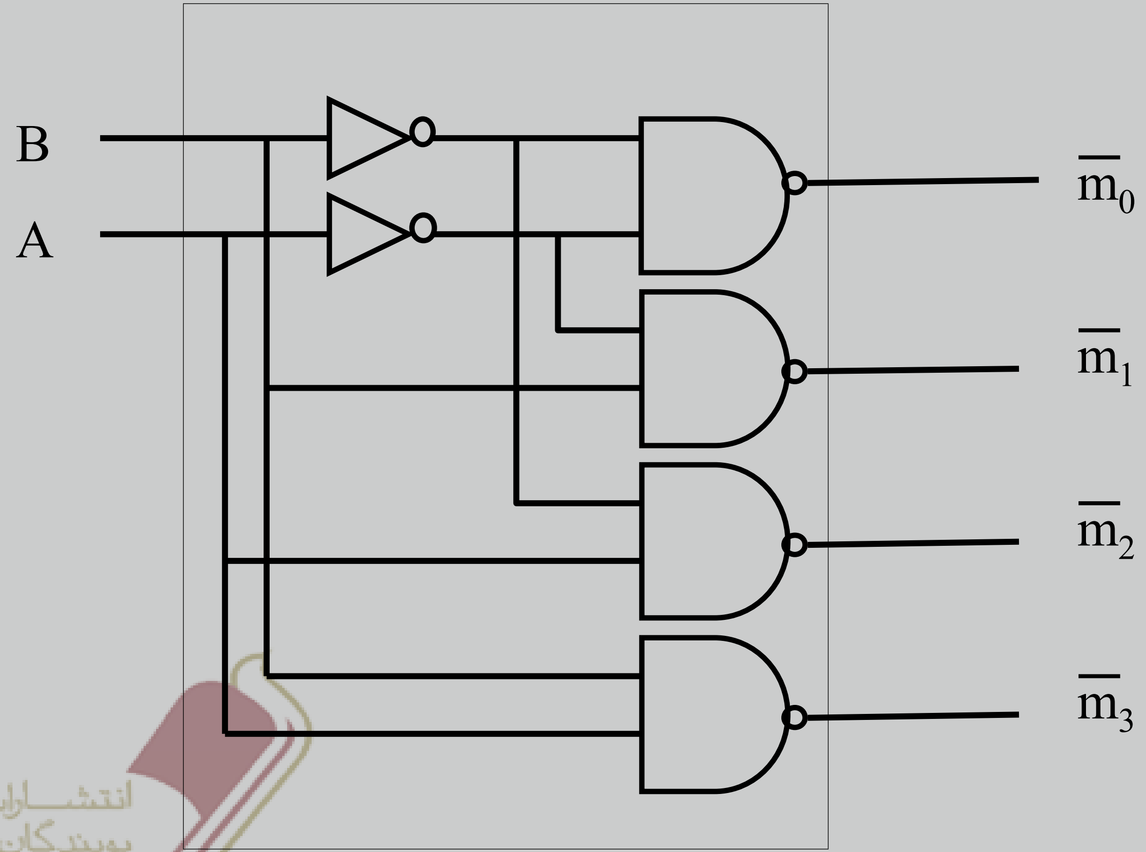
دياگرام منطقي (موازي و خروجي هاي فعال بالا)

Truth Table

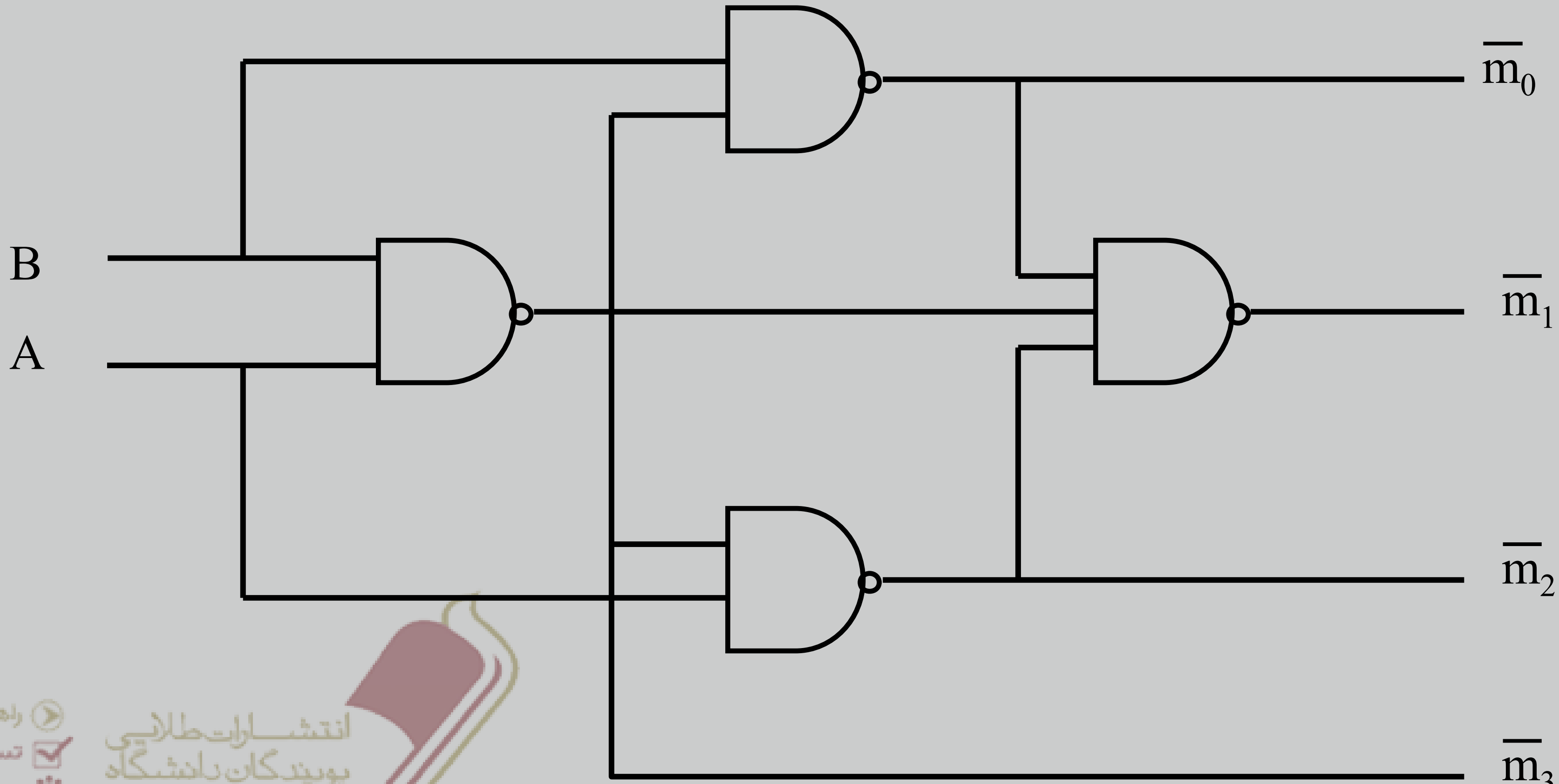
E	A	B	m_0	m_1	m_2	m_3
0	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1
0	×	×	0	0	0	0



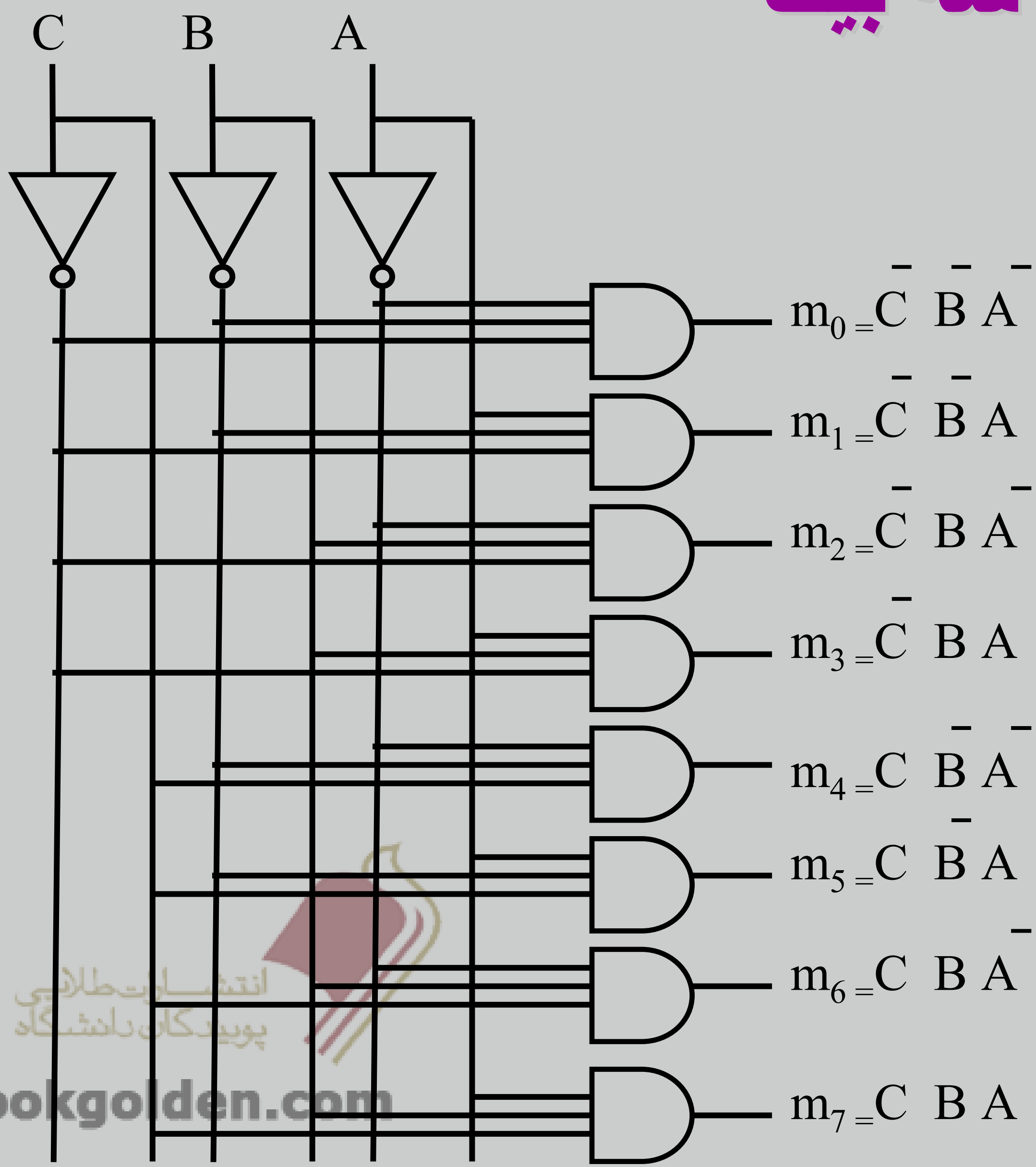
دیاگرام منطقي (موازي و خروجي هاي فعال پايين)



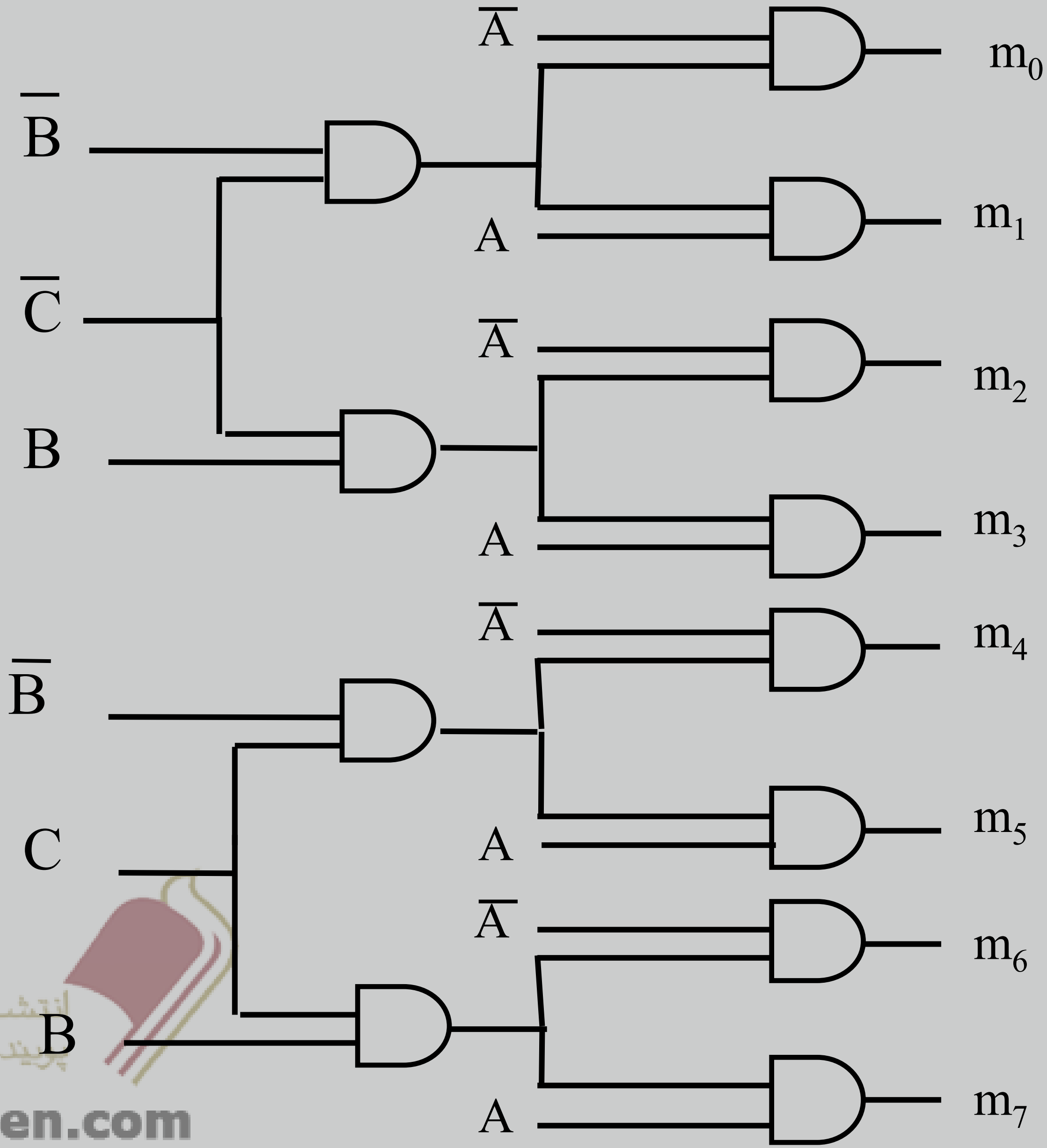
ساختماني ديگر



ديكدر نوع موازي سه بيت



دیکدر نوع درخت سه بیت



پياه سازي توابع منطقي با ديكدر ها

مثال:

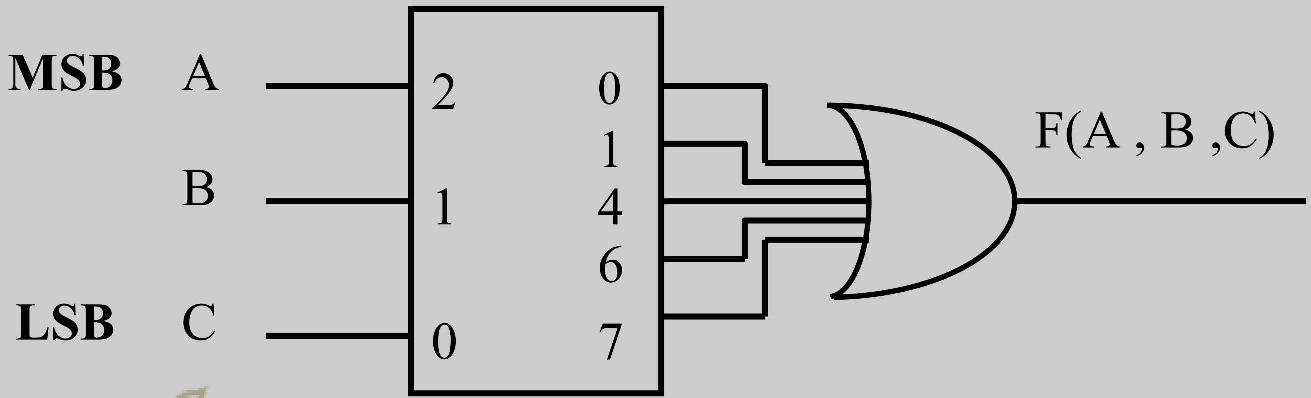
$$F(A, B, C) = \sum m(0, 1, 4, 6, 7) = \prod M(2, 3, 5)$$

تابع را به چندين طريق مي توانيم پياده نماييم:

1. **يك ديكدر (با خروجي فعال بالا) ويك گيت OR بكار بريم.**
2. **يك ديكدر (با خروجي فعال پايين) ويك گيتNAND بكار بريم.**
3. **يك ديكدر (با خروجي فعال بالا) ويك گيت NOR بكار بريم.**
4. **يك ديكدر (با خروجي فعال پايين) ويك گيت AND بكار بريم.**

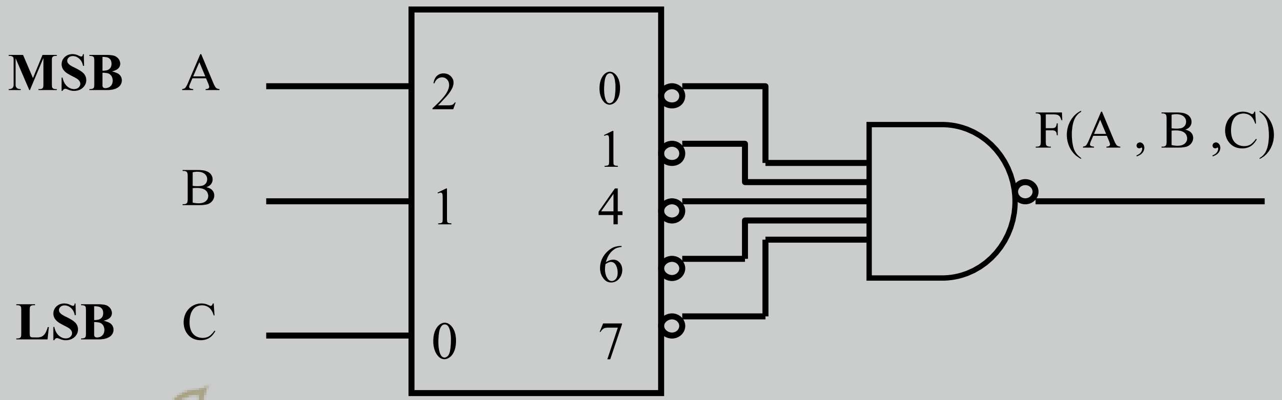
يك ديكدر (با خروجي فعال بالا) ويك گيت OR بكار بريم.

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



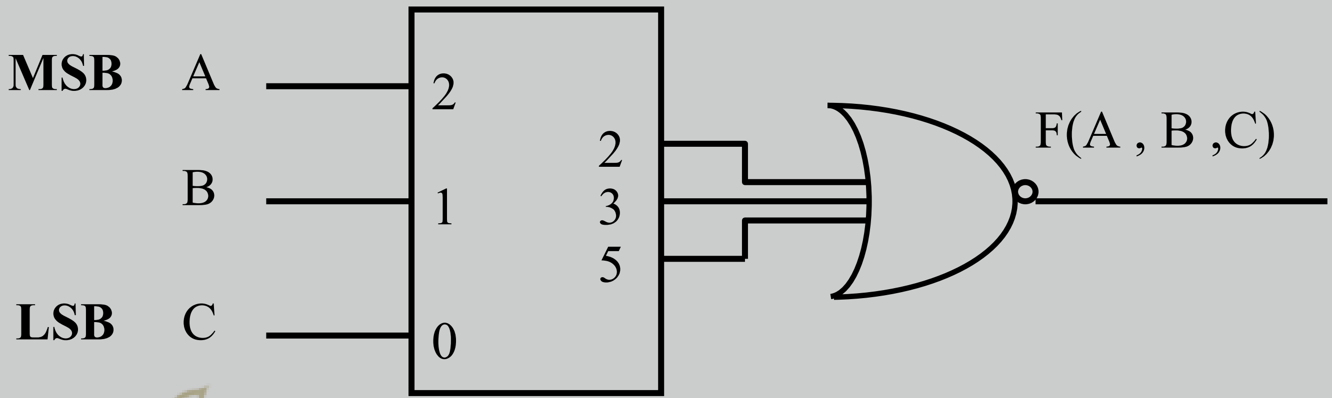
يك ديكدر (با خروجي فعال پايين) ويك گيت NAND بكار بريم.

$$F(A, B, C) = \overline{\bar{m}_0 \cdot \bar{m}_1 \cdot \bar{m}_4 \cdot \bar{m}_6 \cdot \bar{m}_7}$$



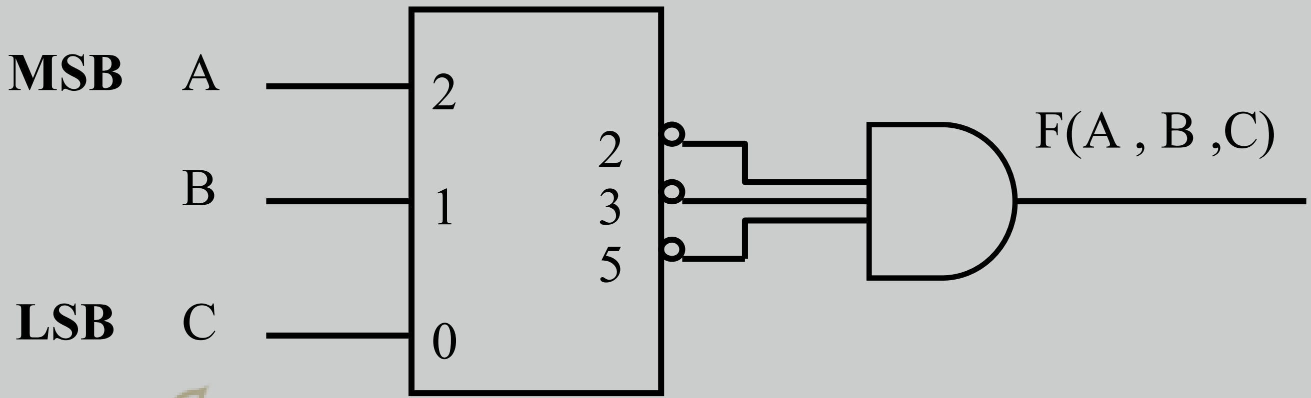
يك ديكدر (با خروجي فعال بالا) ويك گيت NOR بكار بريم.

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

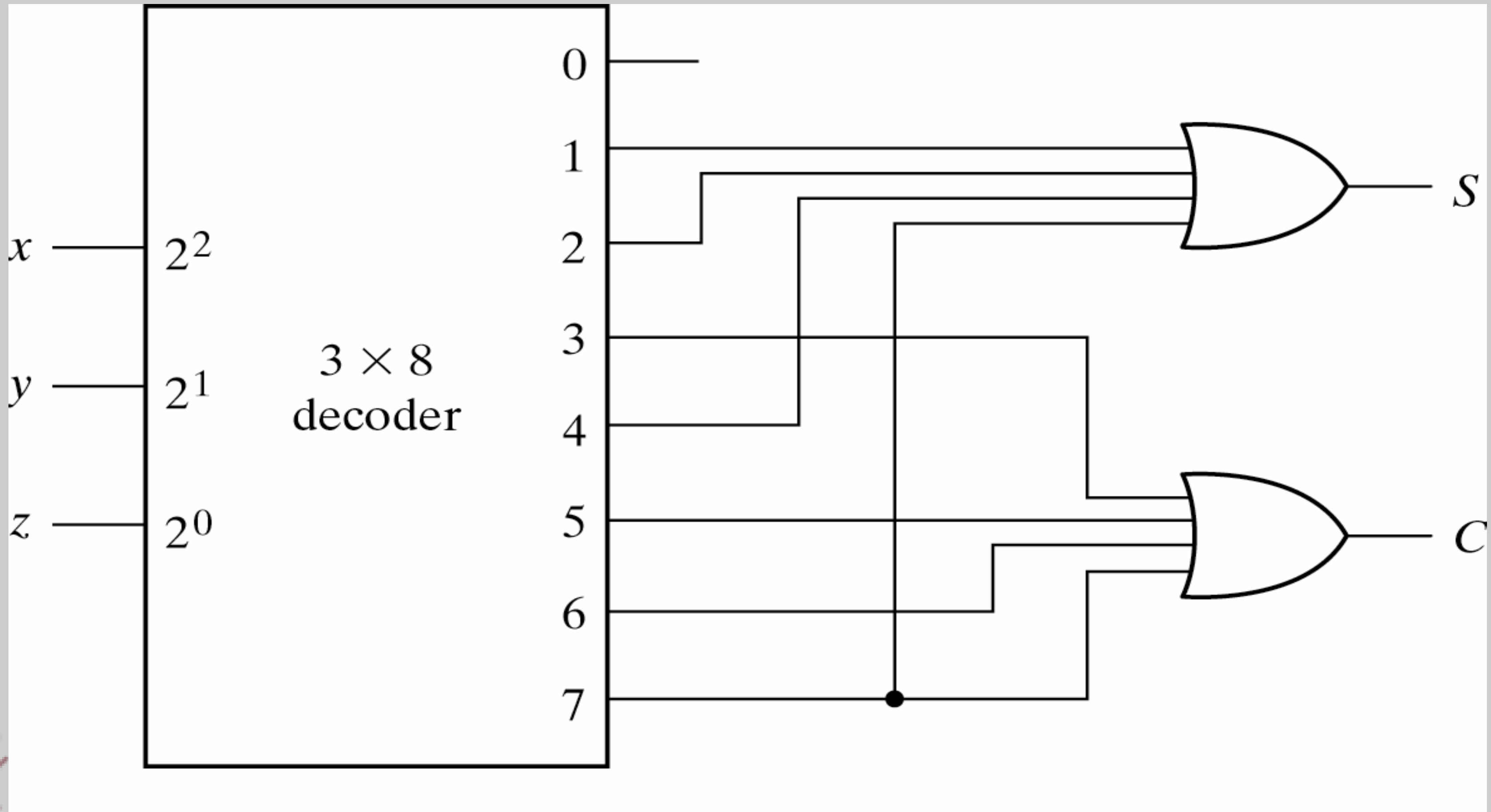


يك ديكدر (با خروجي فعال پايين) ويك گيت AND بكار بريم.

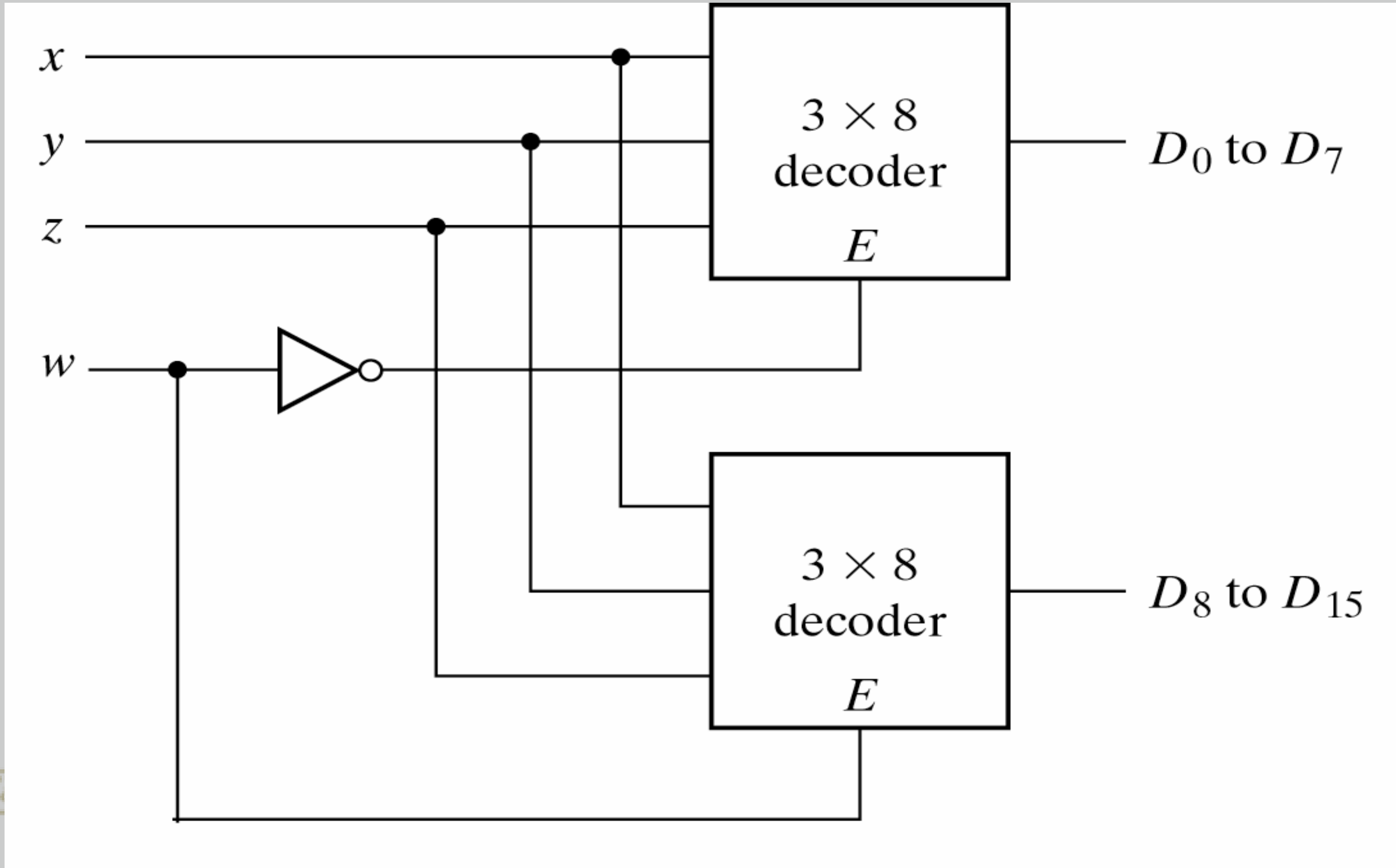
$$F(A , B ,C) = \overline{m_2} \cdot \overline{m_3} \cdot \overline{m_5}$$



ساختنFull Adder به وسیله دیکدر:



ساختن دیکدر بزرگتر:



اينكدر

- اينكدر يك ماجول تركيبي است كه براي هر سيگنال ورودي به دستگاه يك كد خروجي منحصر به فرد را اختصاص مي دهد.
- اگر يك ماجول اينكدر n ورودي داشته باشد خروجي s بايد در رابطه زير صدق كند:

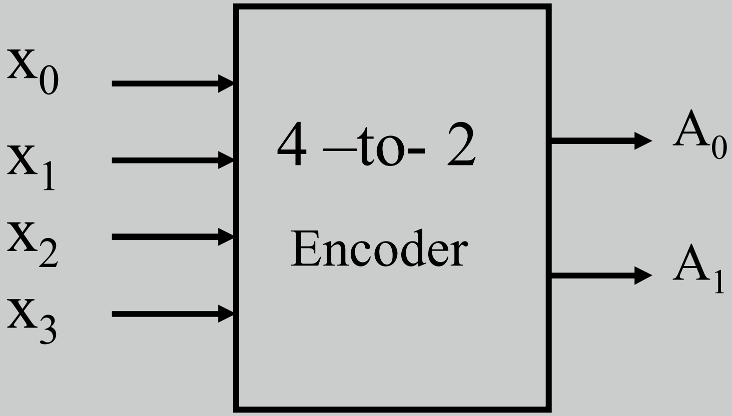
$$2^s \geq n$$

or

$$s \geq \log_2 n$$

مثال:

يك اينكدر براي براي چهار خط ورودي طراحي كنيد بشرطي كه در هر لحظه از زمان فقط يك ورودي فعال باشد.



A_1

d	1	d	1
0	d	d	d
d	d	d	d
0	d	d	d

$A_1 = X_2 + X_3$

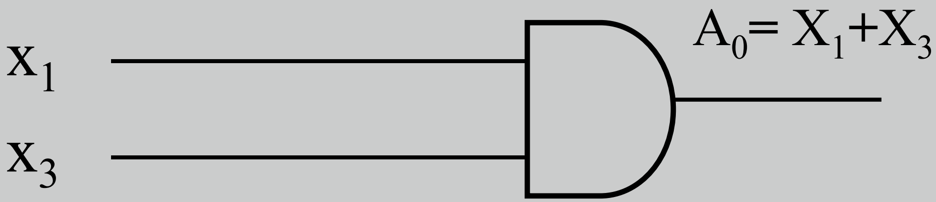
A_0

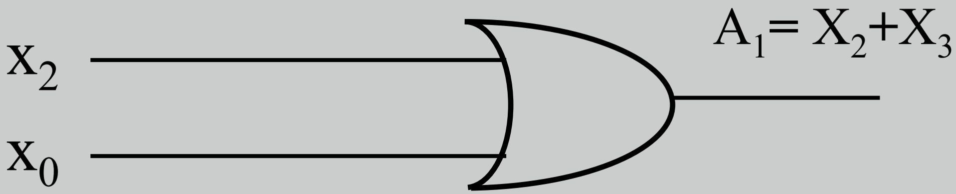
d	0	d	1
0	d	d	d
d	d	d	d
1	d	d	d

$A_0 = X_1 + X_3$

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

دياگرام منطقي





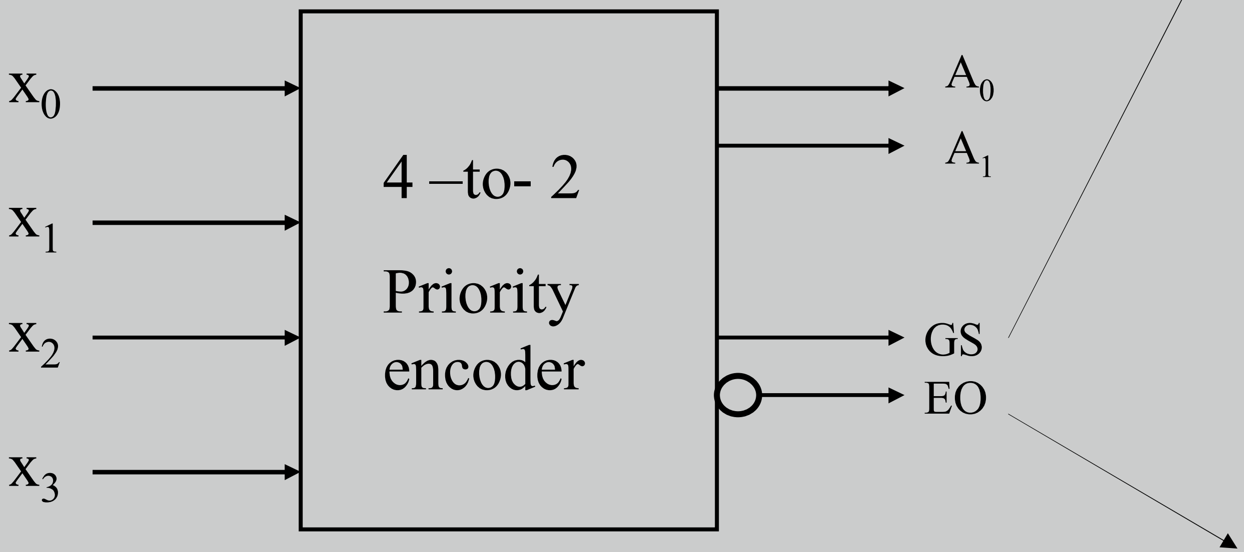
اينكدر اولويت

اينكدر اولويت اجازه مي دهد تا چندين خط ورودي فعال شوند ولي عدد دودويي خارج شده از آن انديسي است كه در خطوط ورودي بالاترين اولويت را دارد.

براي ساده كردن طراحي بالاترين اولويت به بالاترين انديس اختصاص يافته است و بالاترين اولويت بعدي به دومين انديس بالاتر و الي آخر تخصيص داده شده است.

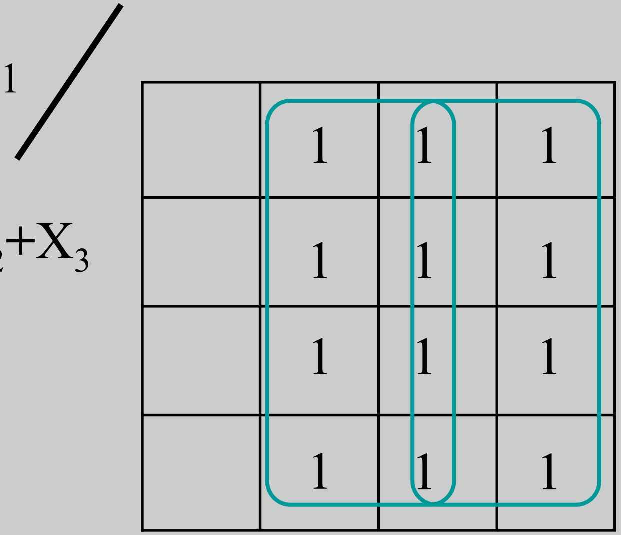
بلوك دياگرام

اگر هيچ يك از خطوط ورودي فعال نباشد
EO=1



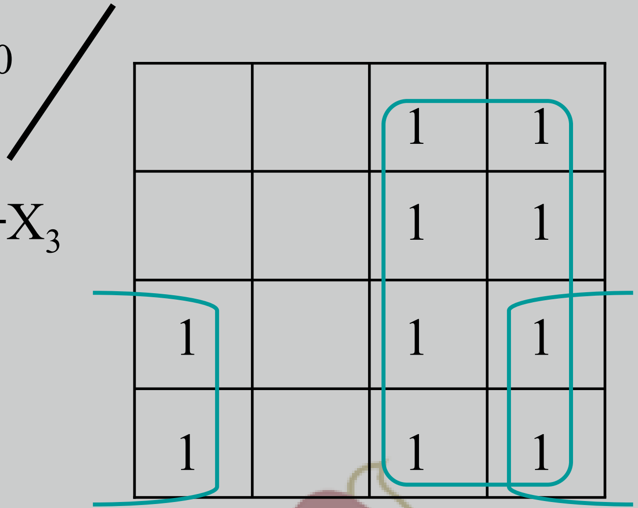
اگر بيش از يكي از خطوط ورودي فعال
باشد GS=1

A_1



$A_1 = X_2 + X_3$

A_0

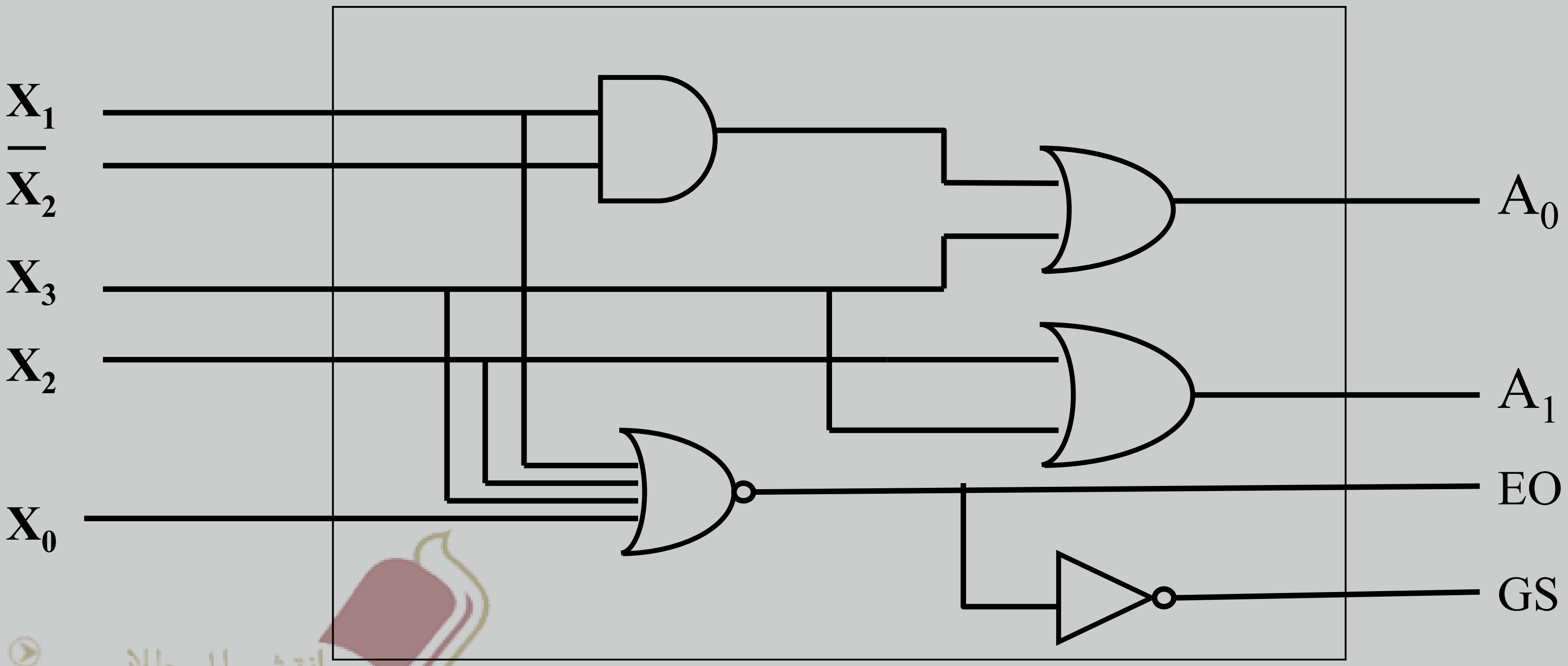


$A_0 = X_1 + X_3$

$EO = \overline{GS} = \overline{X_0 + X_1 + X_2 + X_3}$

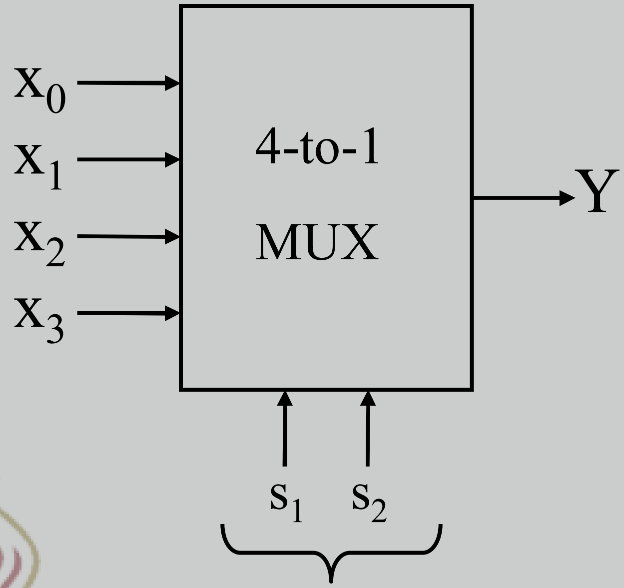
x_3	x_2	x_1	x_0	A_1	A_0	GS	EO
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	1	1	0
0	0	1	1	0	1	1	0
0	1	0	0	1	0	1	0
0	1	0	1	1	0	1	0
0	1	1	0	1	0	1	0
0	1	1	1	1	0	1	0
1	0	0	0	1	1	1	0
1	0	0	1	1	1	1	0
1	0	1	0	1	1	1	0
1	0	1	1	1	1	1	0
1	1	0	0	1	1	1	0
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	0
1	1	1	1	1	1	1	0

دياگرام منطقي



مالتي پلكسر(تسهيم كننده)

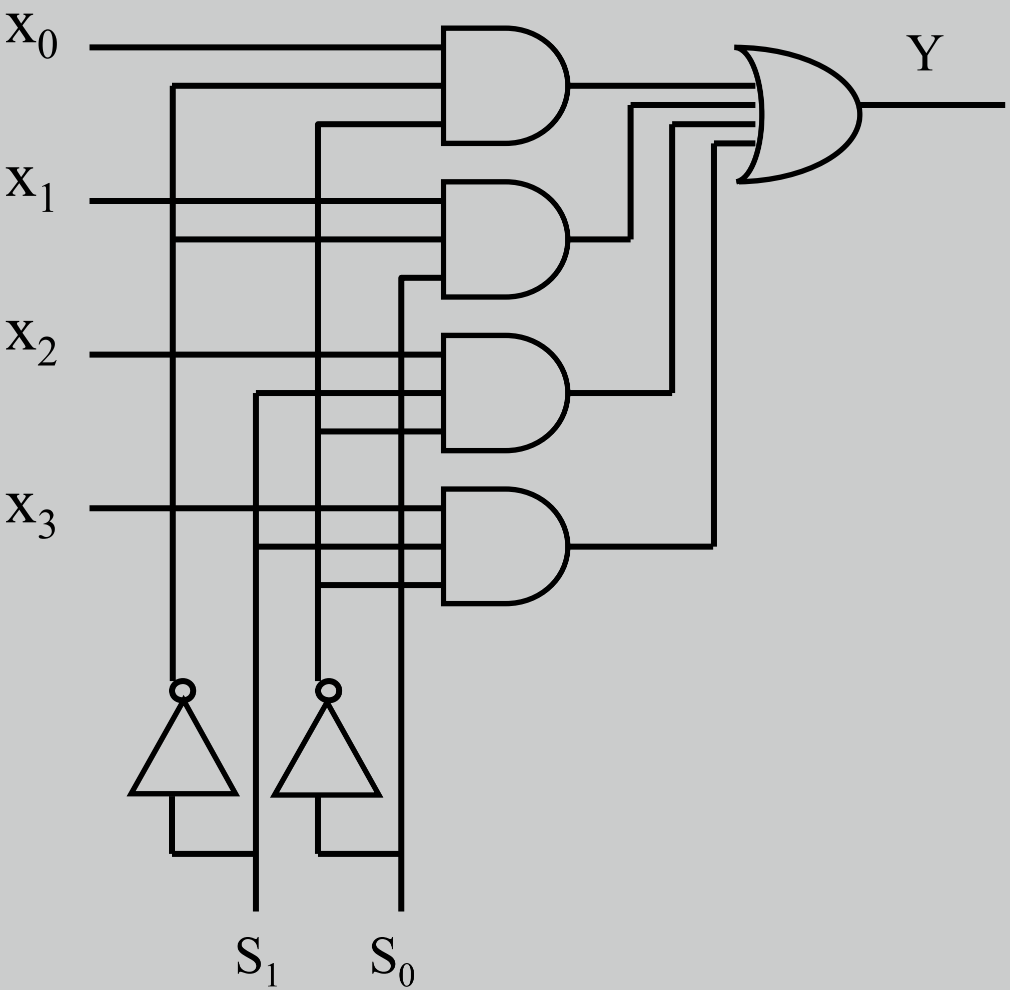
بطور كلي مالتي پلكسر(انتخابگر داده) يك ماجول است كه يكي از چند خط ورودي را انتخاب و آن را روي خط خروجي ظاهر مي سازد.



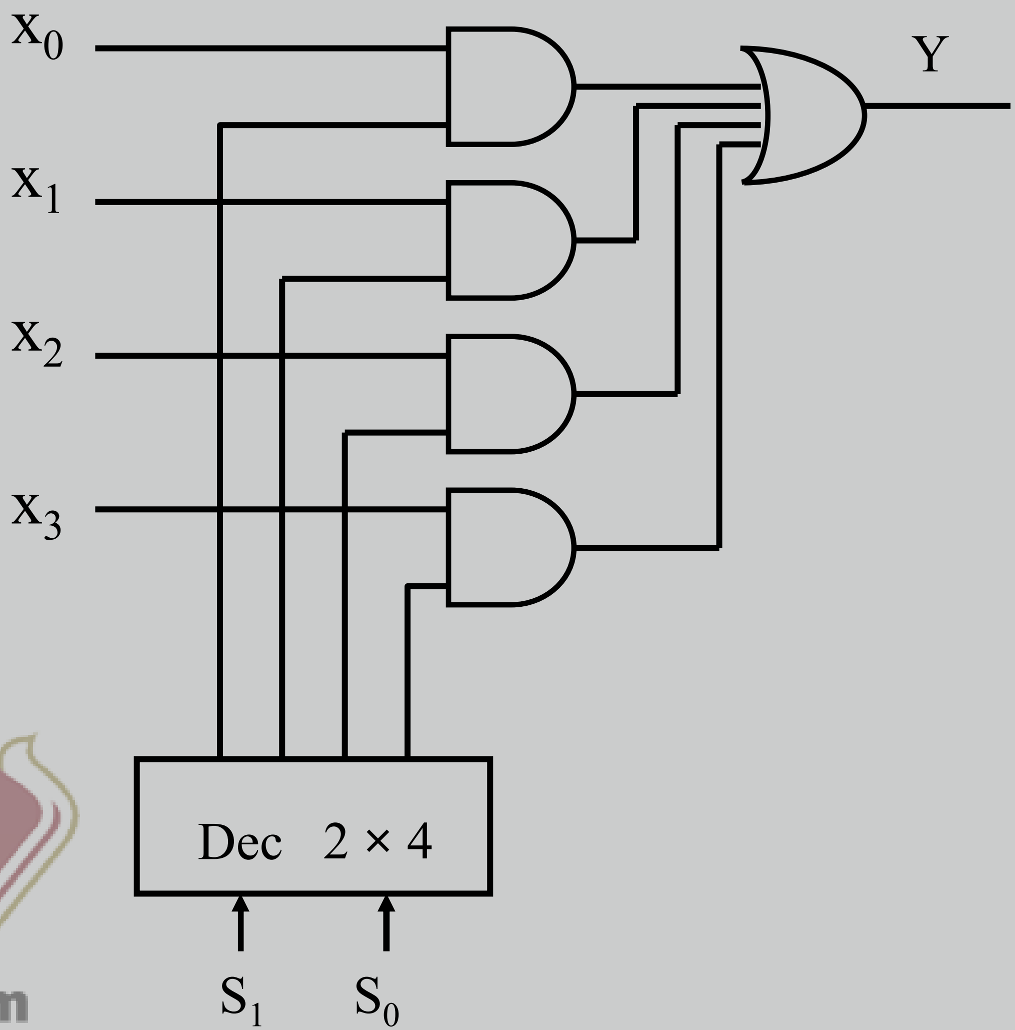
كد انتخاب

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



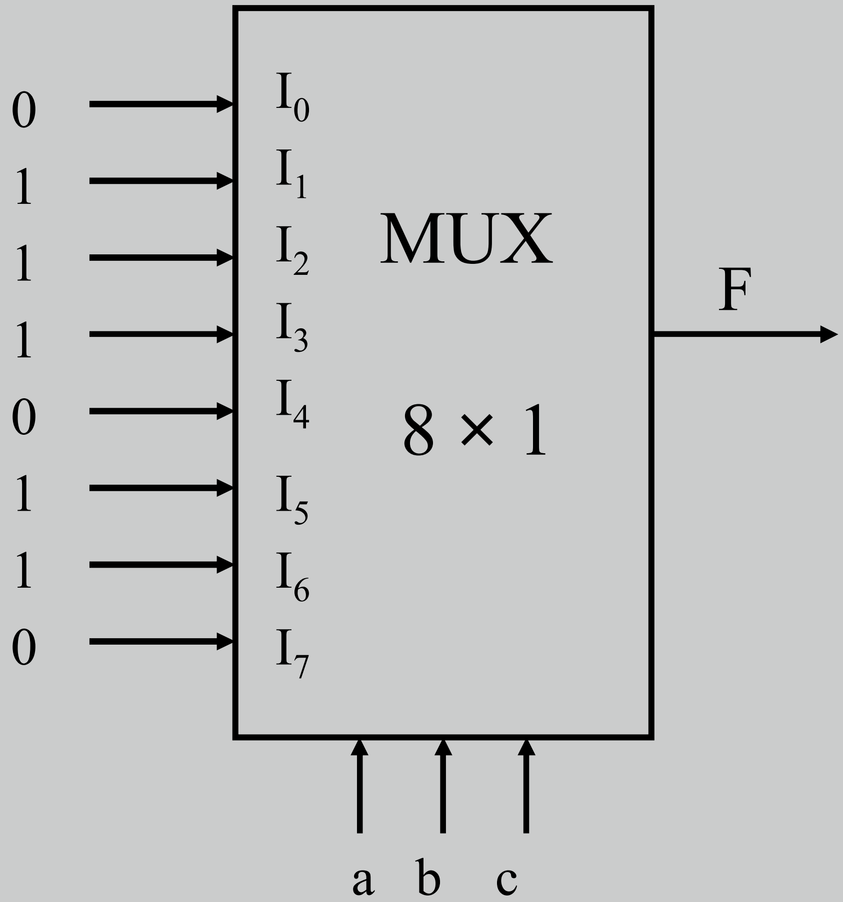
دياگرام منطقي



مثال1 :

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

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

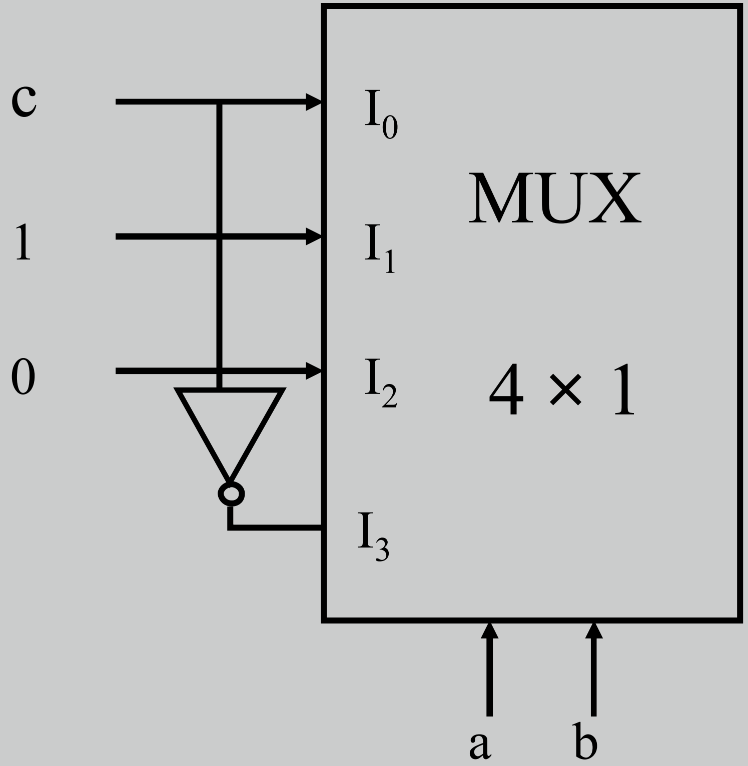


مثال2 :

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

	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

c



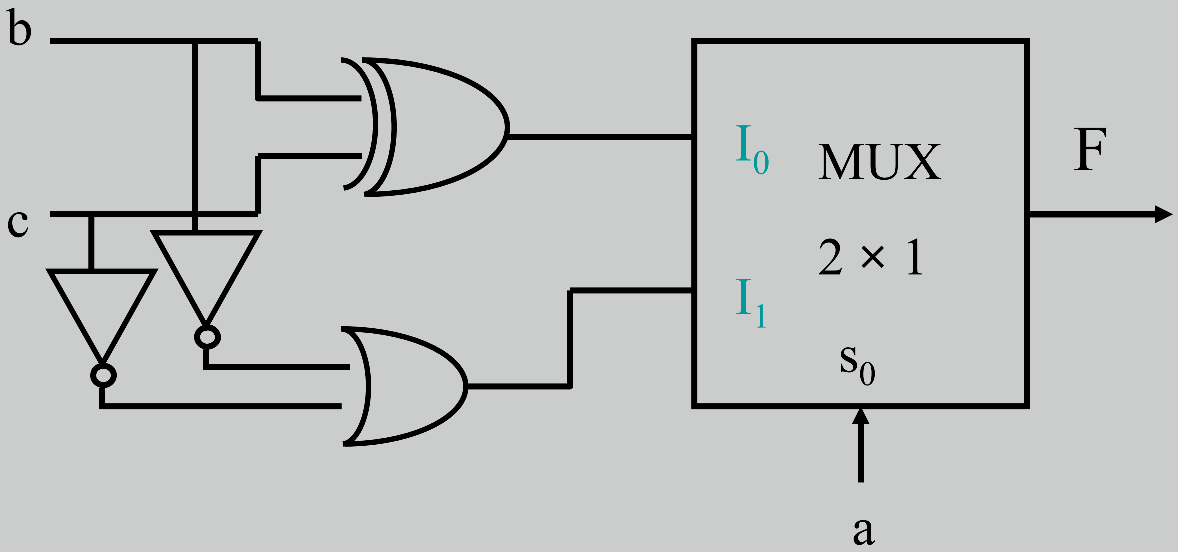
مثال3 :

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

	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

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

a \ bc	00	01	11	10	
0		1		1	I_0
1	1	1		1	I_1



مثال4 :

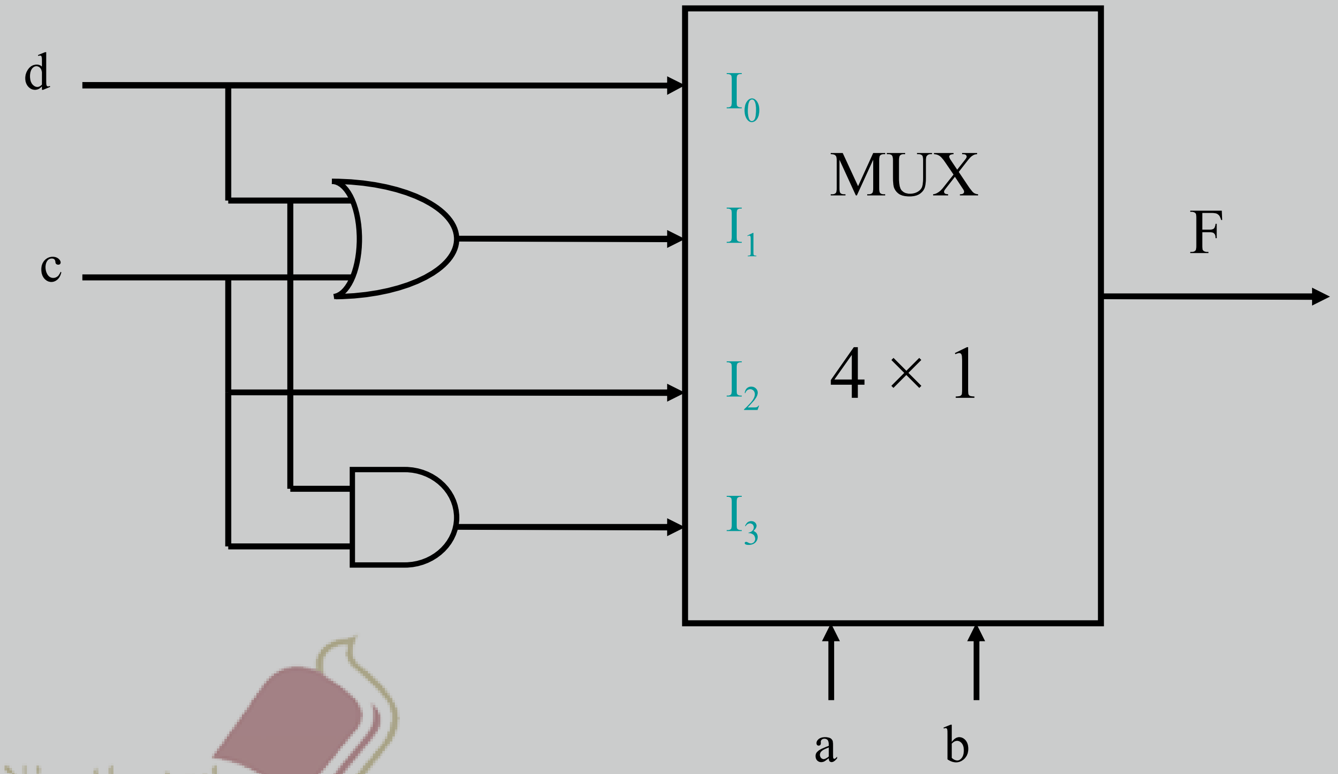
$$F(A, B, C, D) = \sum m(1, 3, 5, 6, 7, 10, 11, 15)$$

	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

ab \ cd	00	01	11	10	
00		1	1		I_0
01		1	1	1	I_1
11			1		I_2
10			1	1	I_3

$$\begin{cases} I_0 = d \\ I_1 = d + c \\ I_2 = C \\ I_3 = cd \end{cases}$$

مثال4 :



دي مالتي پلكسر(پخش كننده داده ورودي)

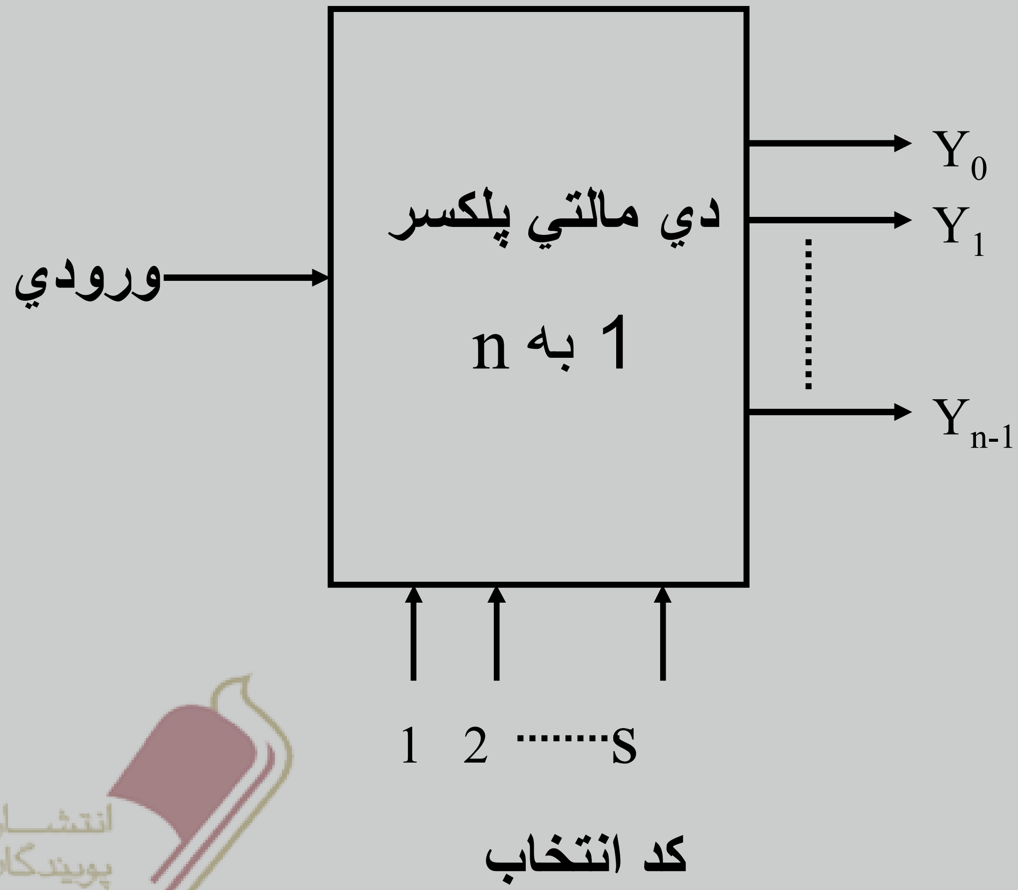
يك مدار منطقي تركيبي كه خط را به يك خط ورودي را به يكي از n خط خروجي وصل مي كند

خط خروجي خاص با يك كد انتخاب s بيتي معين مي شود كه:

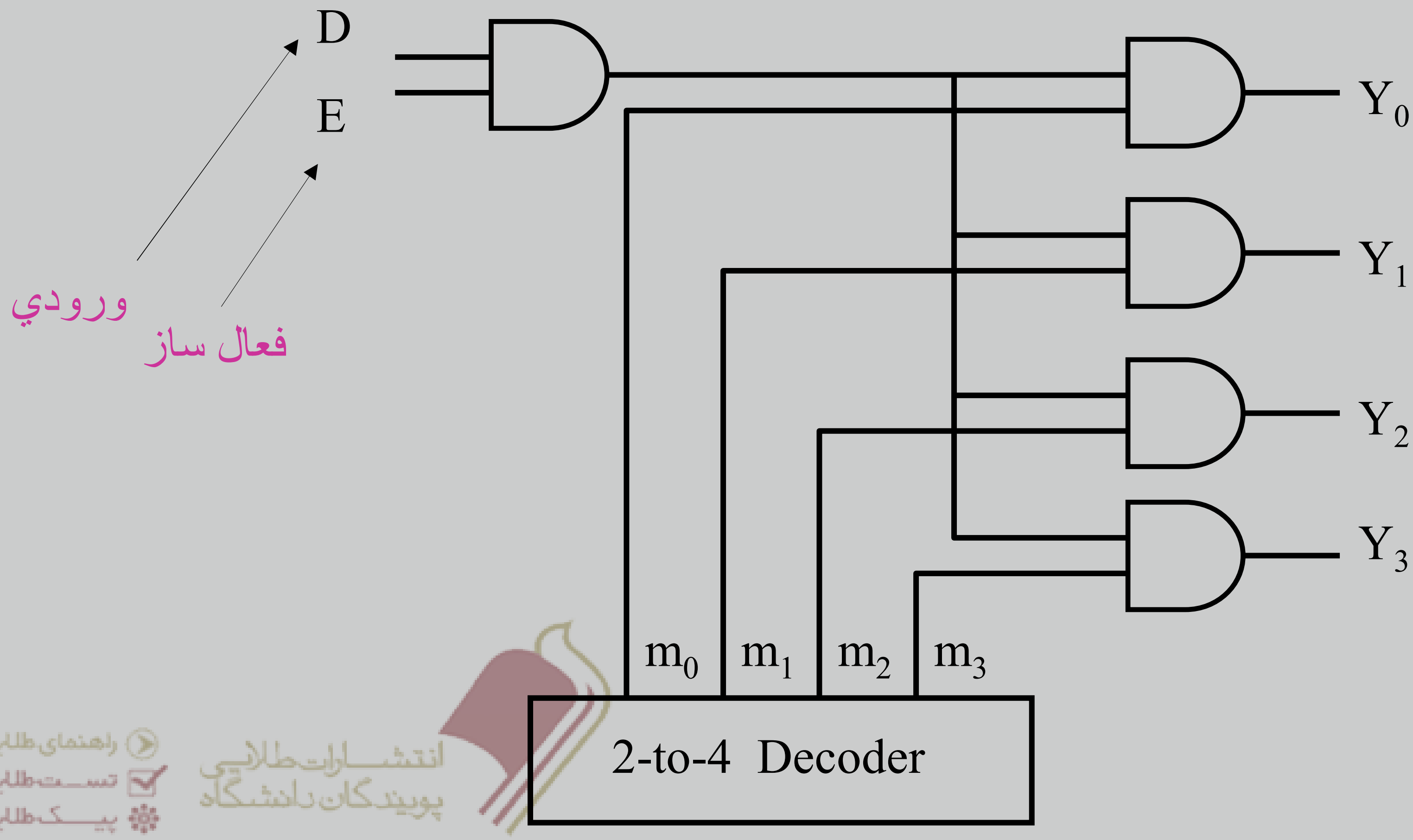
$$2^{S} \geq n$$

در اين حالت كد انتخاب براي توليد مينترم هاي s بكار مي رود.

دياگرام عملياتي



دي مالتي پلکسر 1 به 4 با فعال ساز



مقايسه گرها

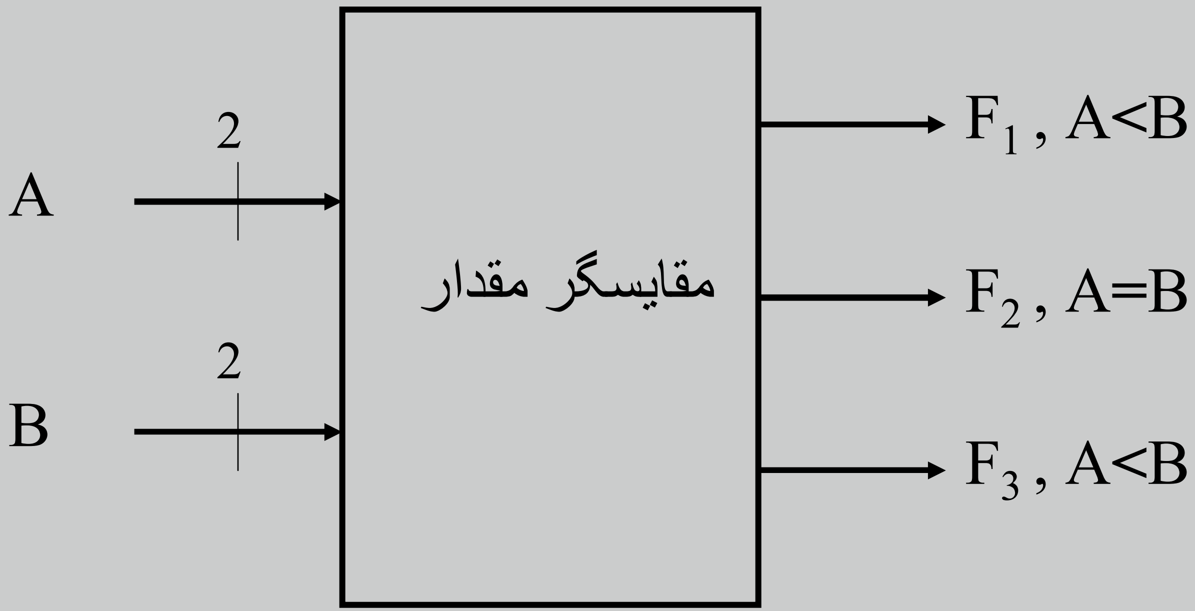
- مقايسه گر قطعه اي محاسباتي است كه اندازه نسسبي دو عدد دودويي را معيين مي كند.

- در يك مقايسه گر سه تصميم كاملا ديكد شده در مورد دو كلمه انجام و در خروجي ها قرار مي گيرند. يعني A=B , A>B , A>B اگر

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

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

دیاگرام عملیاتي



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

مثال :

مقايسه گري طراحي كنيد كه دو كلمه

$B=(B_1B_0)_2$ و $A=(A_1A_0)_2$ را

در كد دودويي مقايسه كند.

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

F_1, A<B

	00	01	11	10
00				
01	1			
11	1	1		1
10	1	1		

F_2 ,A=B

	00	01	11	10
00	1			
01		1		
11			1	
10				1

F_3, A>B

	00	01	11	10
00		1	1	1
01			1	1
11				
10			1	

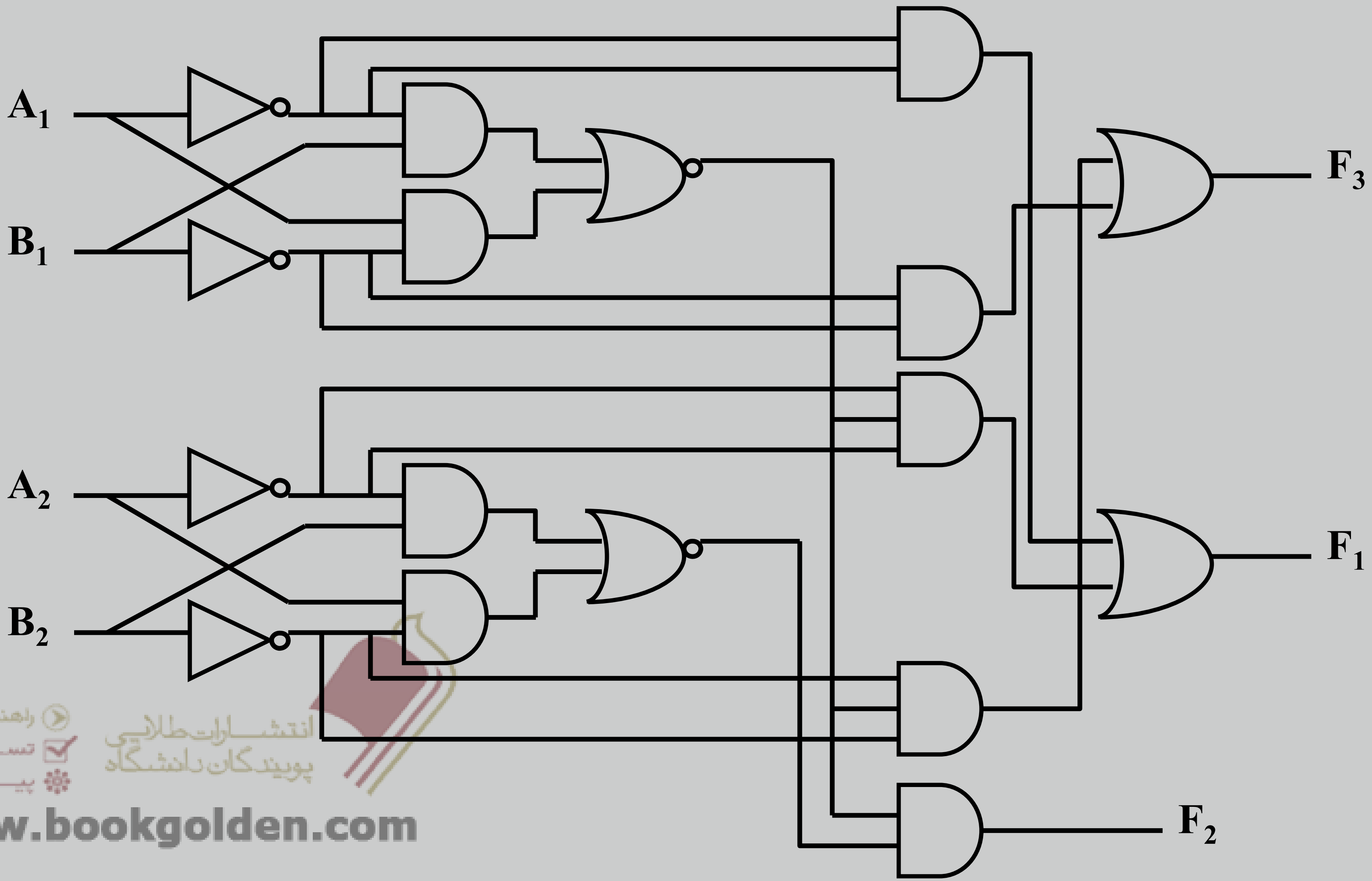
توابع خروجي

$F_1 = \overline{A_1}\,B_1 + \overline{A_1}\,\overline{A_0}\,B_0 + \overline{A_0}\,B_1B_0$ For $(A_1A_0)_2 < (B_1B_0)_2$

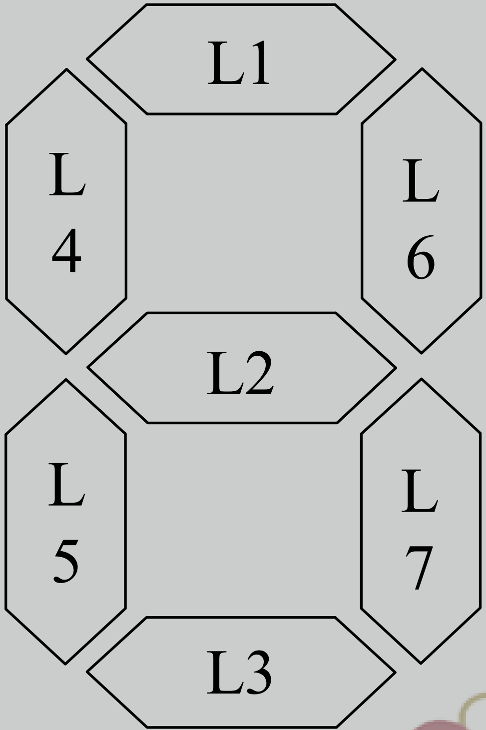
$F_2 = \overline{A_1}\,\overline{A_0}\,\overline{B_1}\,\overline{B_0} + \overline{A_1}\,A_0\,\overline{B_1}\,B_0 + A_1\overline{A_0}\,B_1\,\overline{B_0} + A_1A_0B_1B_0$ For $(A_1A_0)_2 = (B_1B_0)_2$

$F_3 = A_1\overline{B_1} + A_1\overline{B_1}\,\overline{B_0} + A_1A_0\overline{B_0}$ For $(A_1A_0)_2 > (B_1B_0)_2$

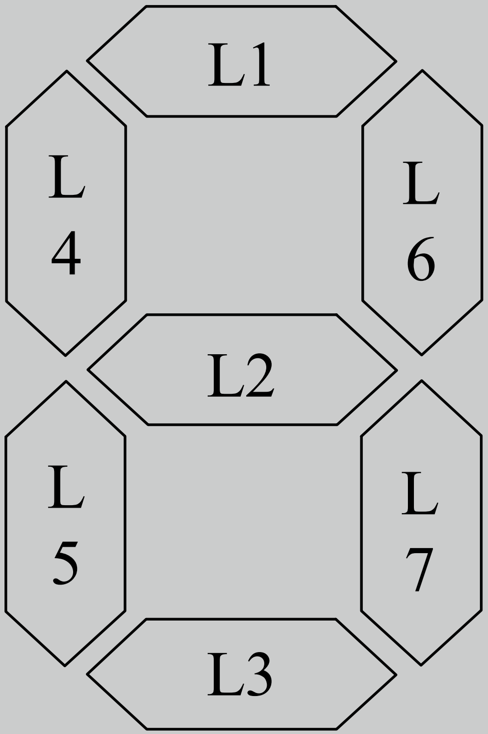
تحقيق منطقي يك مقايسه گر دو بيت



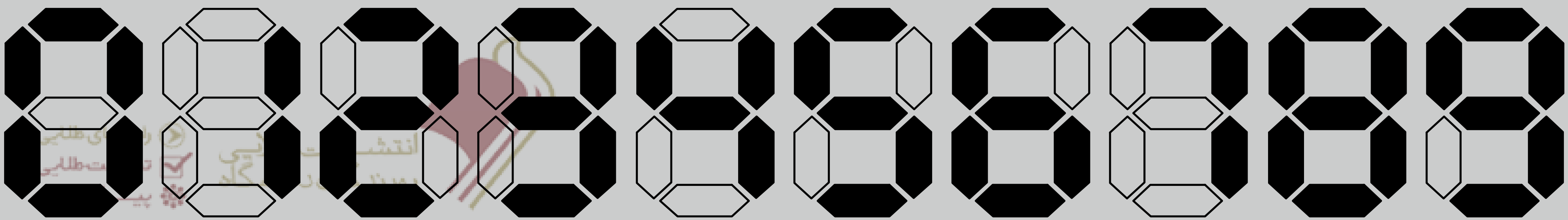
مثال : Seven Segment Display



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

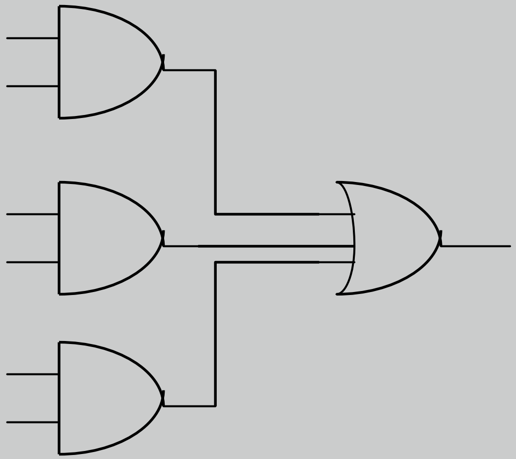


B3	B2	B1	B0	Val	L1	L2	L3	L4	L5	L6	L7
0	0	0	0	0	1	0	1	1	1	1	1
0	0	0	1	1	0	0	0	0	0	1	1
0	0	1	0	2	1	1	1	0	1	1	0
0	0	1	1	3	1	1	1	0	0	1	1
0	1	0	0	4	0	1	0	1	0	1	1
0	1	0	1	5	1	1	1	1	0	0	1
0	1	1	0	6	1	1	1	1	1	0	1
0	1	1	1	7	1	0	0	0	0	1	1
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1



❑ المنت L4:

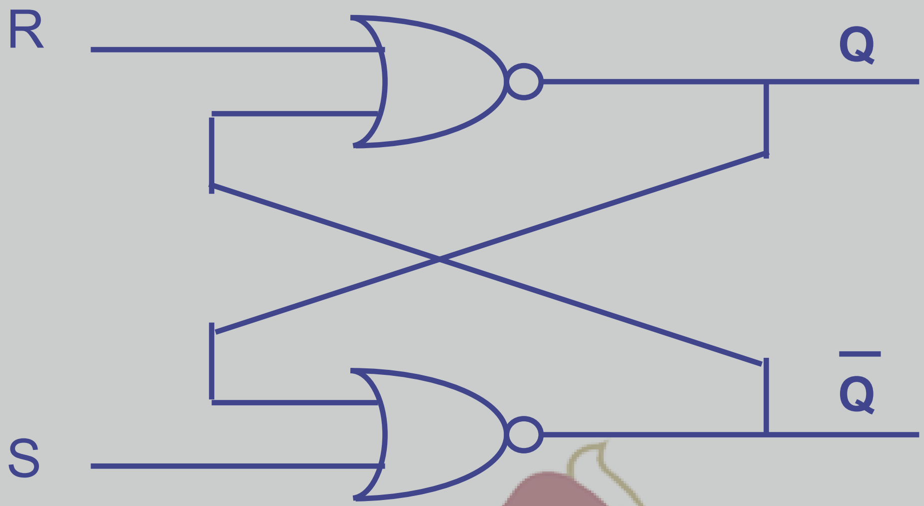
B3	B2	B1	B0	L4
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1



فصل ششم:

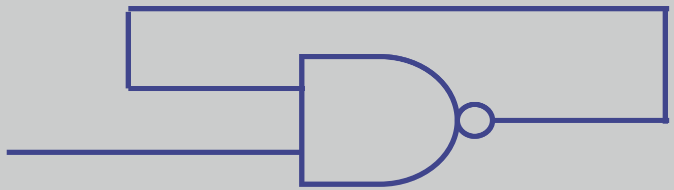
مدارات ترتیبی

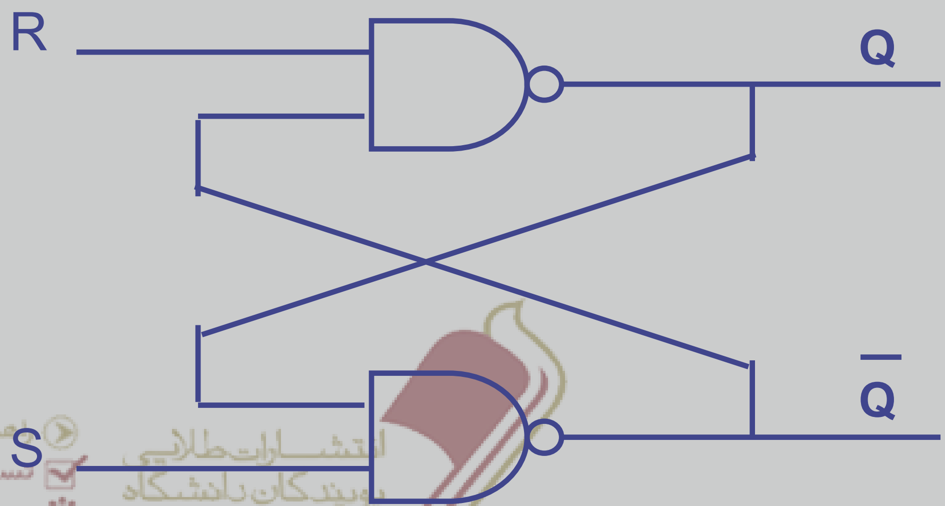
مفهوم Latch :



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

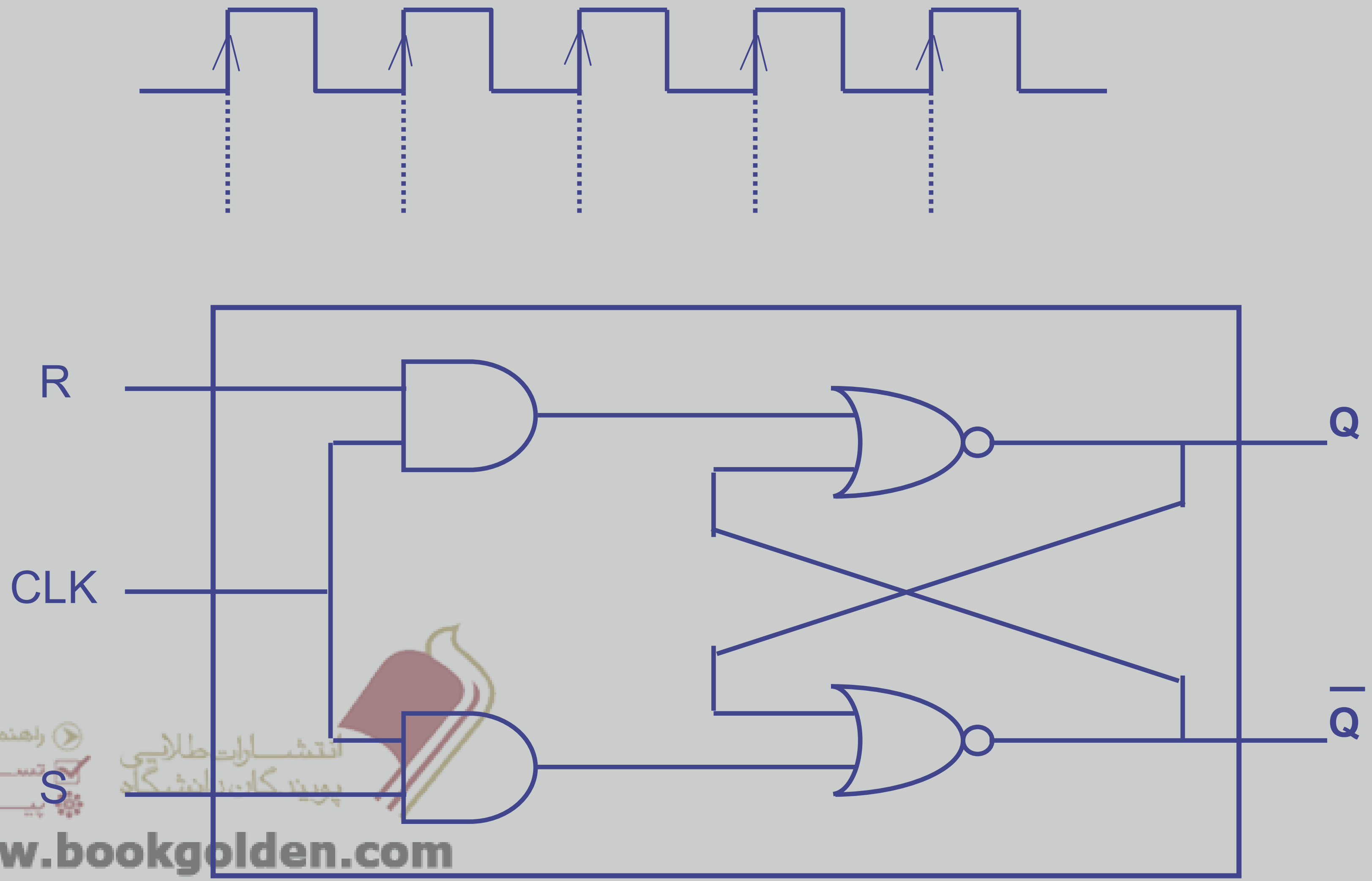
نمونه ی دیگر :





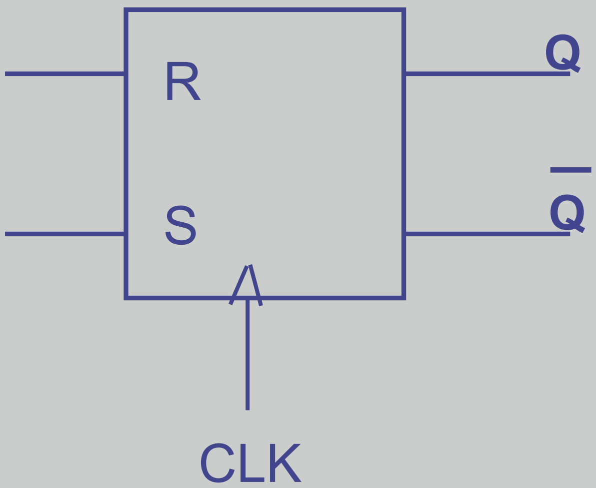
A	B	O1	O2
0	0	1	1
0	1	1	0
1	0	0	1
1	1	نامعین	نامعین

CLK: پالس های ساعت که باعث همگام سازی مدار می شود .



انواع فلیپ فلاپ ها: RS, JK, T, D

فلیپ فلاپ RS

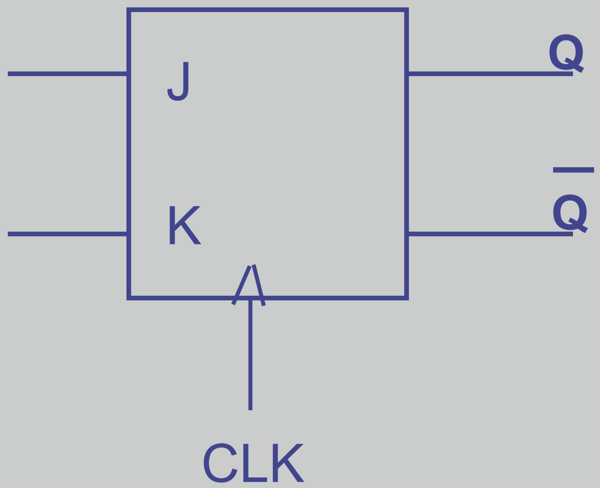


S	R	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	نامعین

(جدول مشخصه)

انواع فلیپ فلاپ ها: (ادامه)

فلیپ فلاپ JK



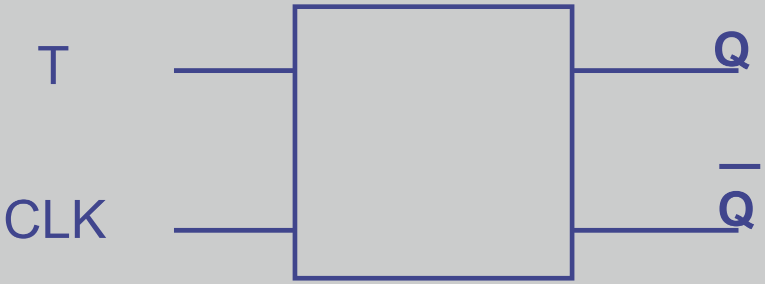
T D

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

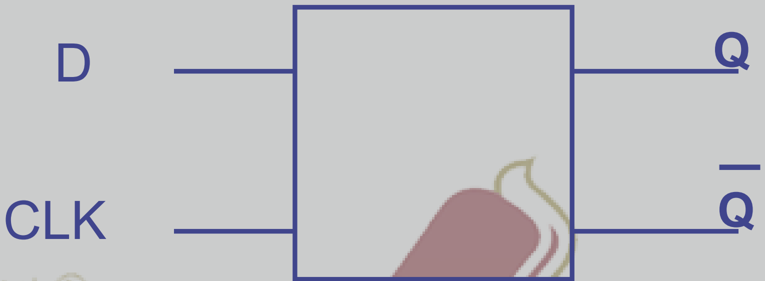
(جدول مشخصه)

انواع فلیپ فلاپ ها: (ادامه)

فلیپ فلاپ T, D



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

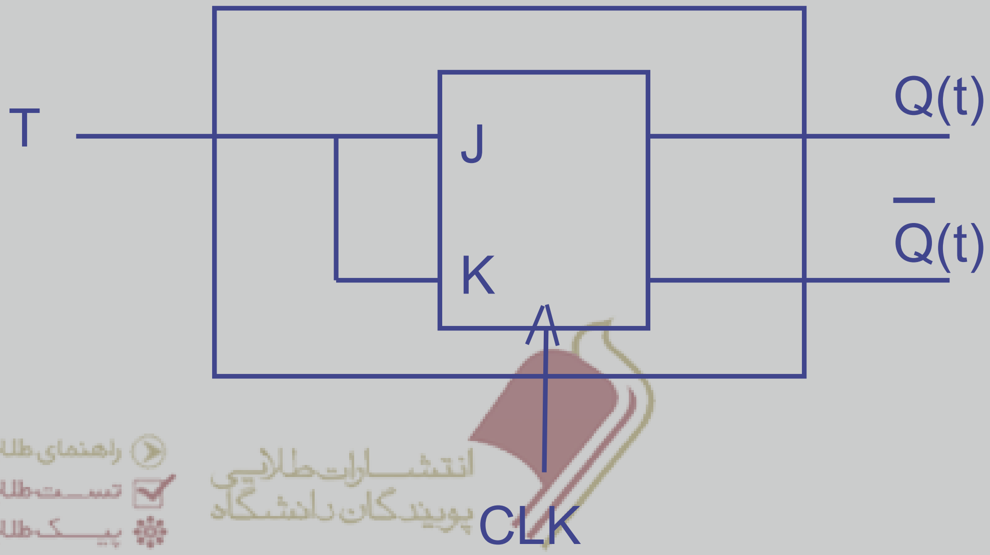


D	Q(t+1)
0	0
1	1

(جدول مشخصه)

مثال1: به کمک فلیپ فلاپ JK یک فلیپ فلاپ Tبسازید.

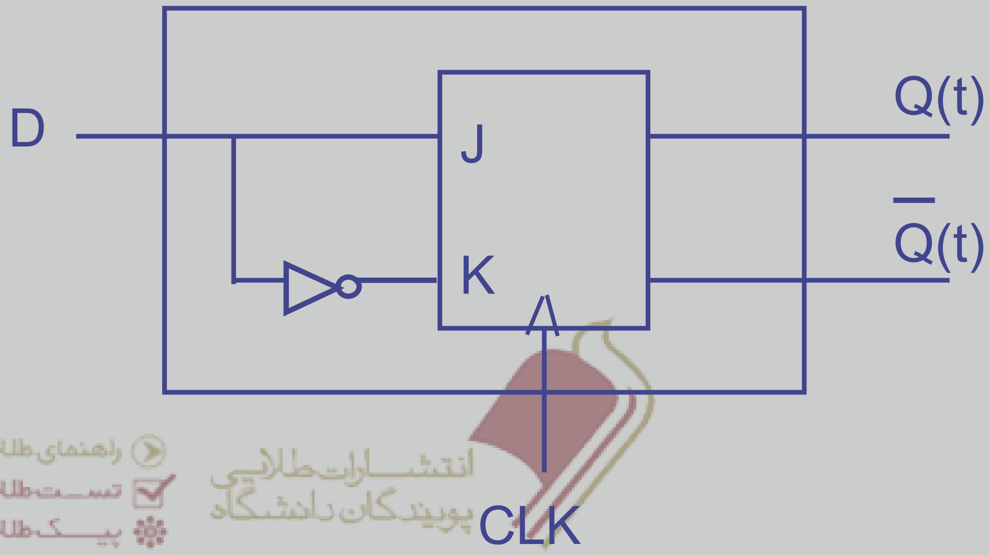
T	J	K	Q(t+1)
0	0	0	$Q(t)$
1	1	1	$\overline{Q}(t)$



مثال2 : 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



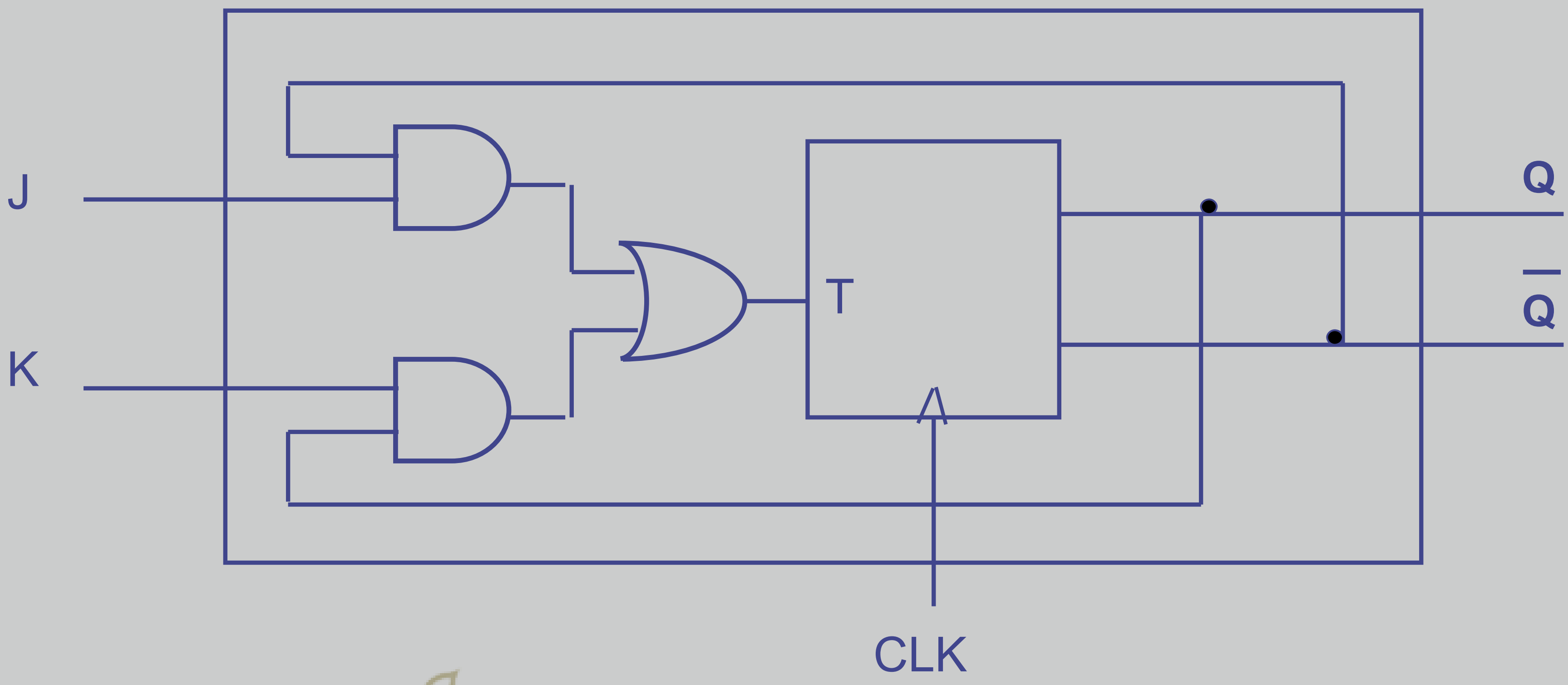
مثال3 : به کمک فلیپ فلاپ T یک فلیپ فلاپ JK بسازید.

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

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

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

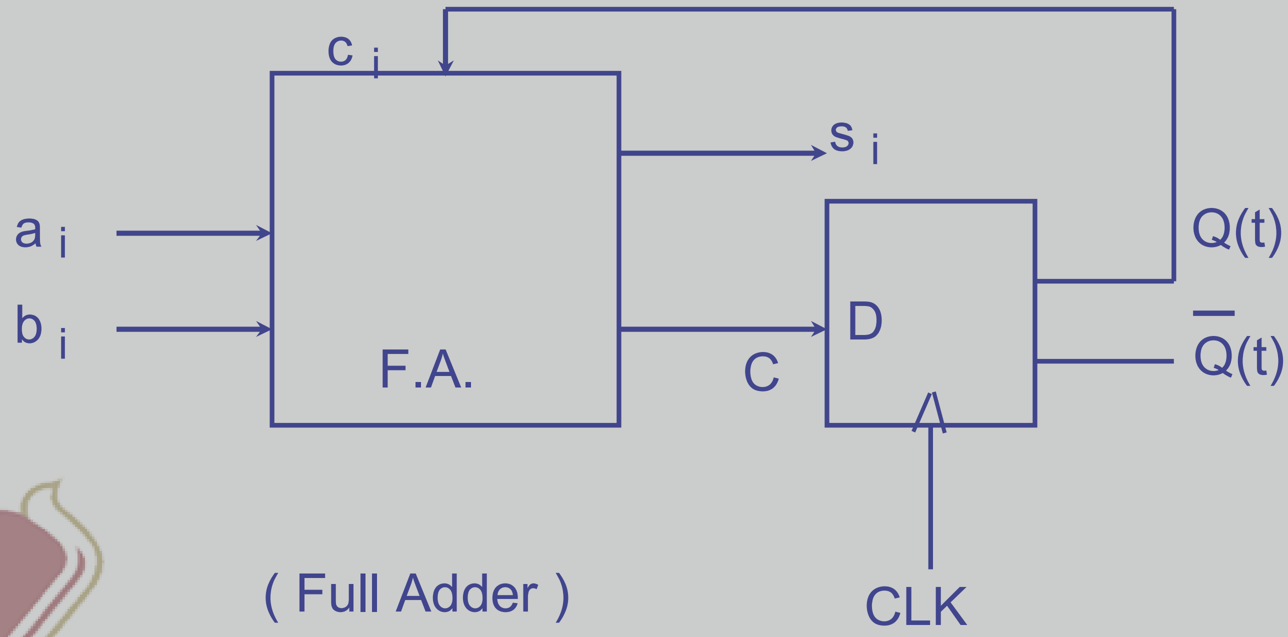
مثال 3: (ادامه)



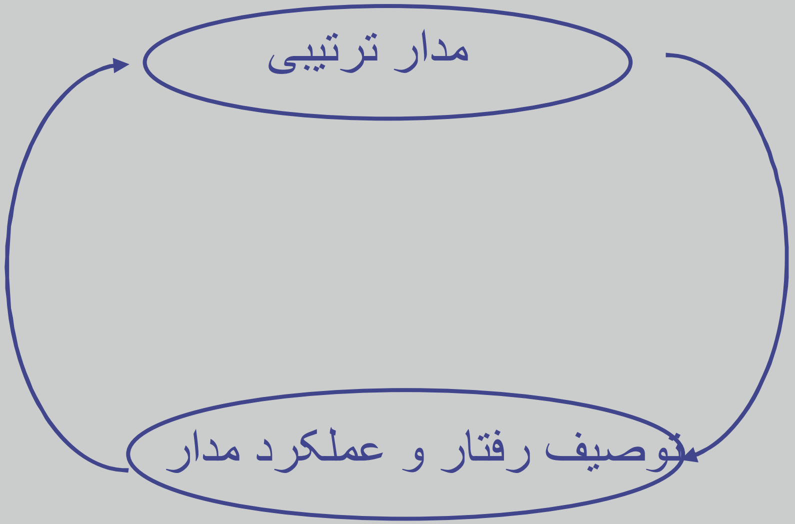
مثال4 :به کمک فلیپ فلاپ D یک فلیپ فلاپ JK بسازید.

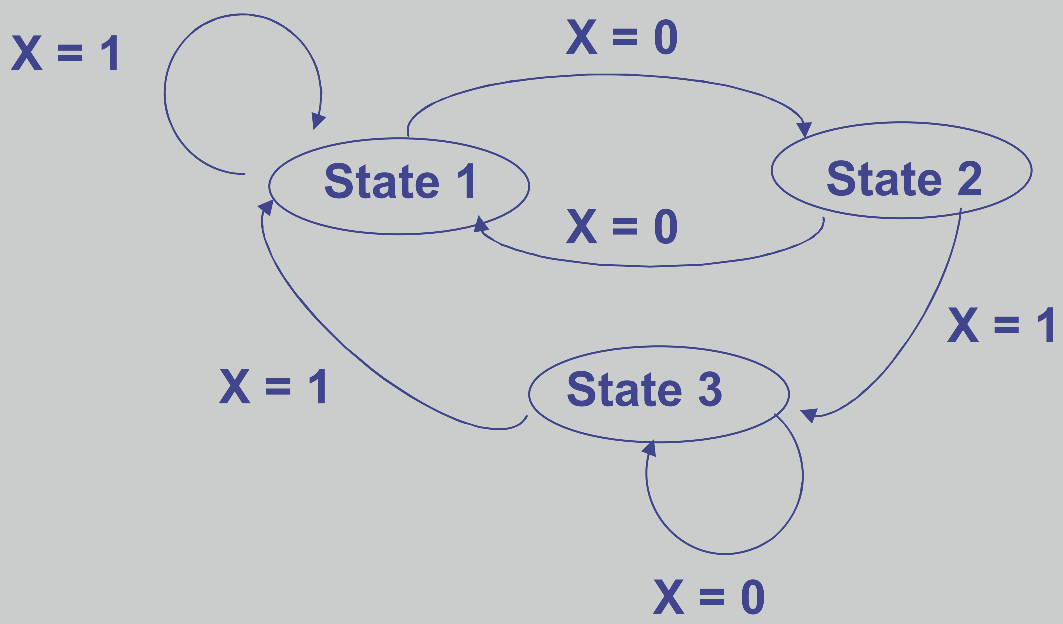
مثال 5: مداری طراحی کنید که دو عدد nبیتی را با هم جمع کند، به طوریکه در هرکلاک پالس دو بیت داده شود.

$$
\begin{array}{lllll}
A: & a_3 & a_2 & a_1 & a_0 \\
B: & b_3 & b_2 & b_1 & b_0 \\
\hline
C: & s_3 & s_2 & s_1 & s_0
\end{array}
$$



روند تجزیه و تحلیل مدارات ترتیبی :



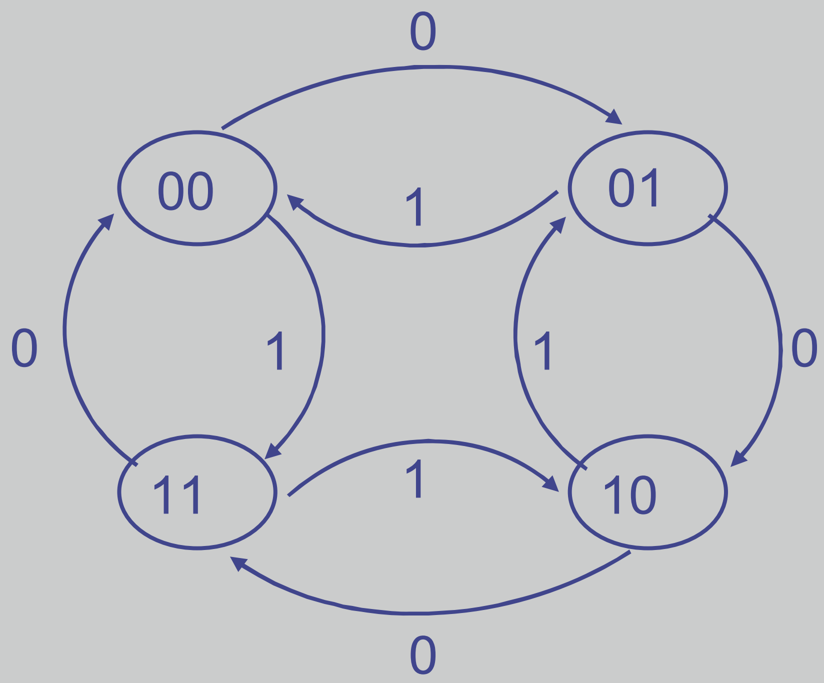


مثال6 : یک شمارنده بالاشمار (با ورودی1) و پایین شمار (با ورودی 0)

- مشخص کردن حالت ها و ترسیم آن

(تعداد فلیپ فلاپ ها)

تعداد حالات = 2



طراحی مدار های ترتیبی :

تجزیه و تحلیل

طراحی

مدار

↓

جدول حالت

↓

S.D.

↓

توصیف عملکرد

توصیف عملکرد

↓

S.D.

↓

S.D.کمینه (کاهش یافته تعداد حالات)

↓

تخصیص مقدار به حالات

↓

جدول حالت (جدول تحریک)

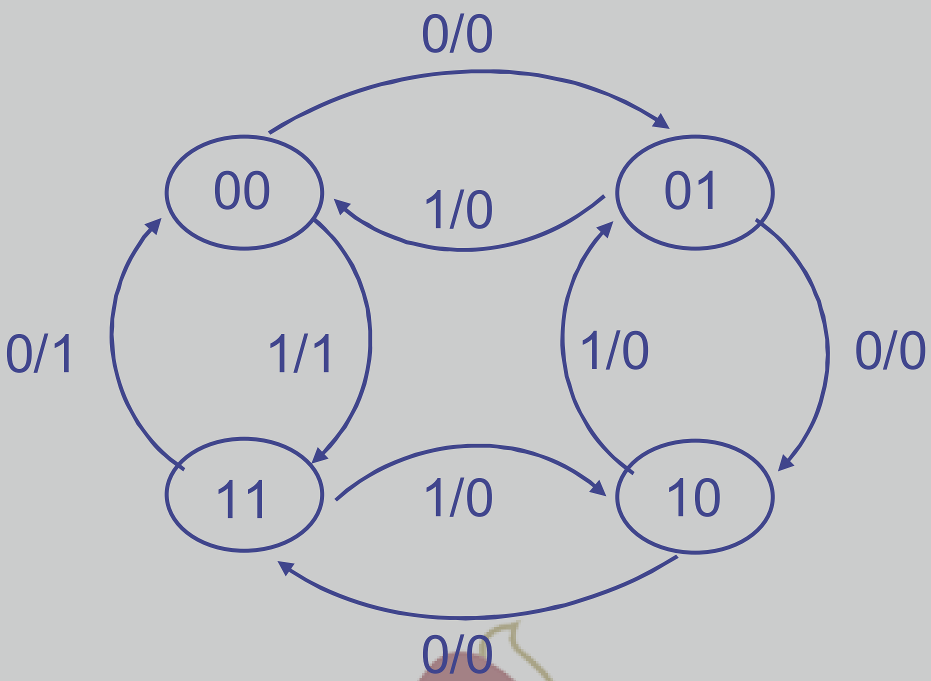
↓

ساده سازی کارنا

↓

مدار

مثال7 : طراحی یک شمارنده دو بیتی بالاشمار (با ورودی 0) و پایین شمار (با ورودی 1)، Carry خروجی
(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 7 : (ادامه)

تعداد فلیپ فلاپ ها : $\lceil \log 4 \rceil = 2$

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

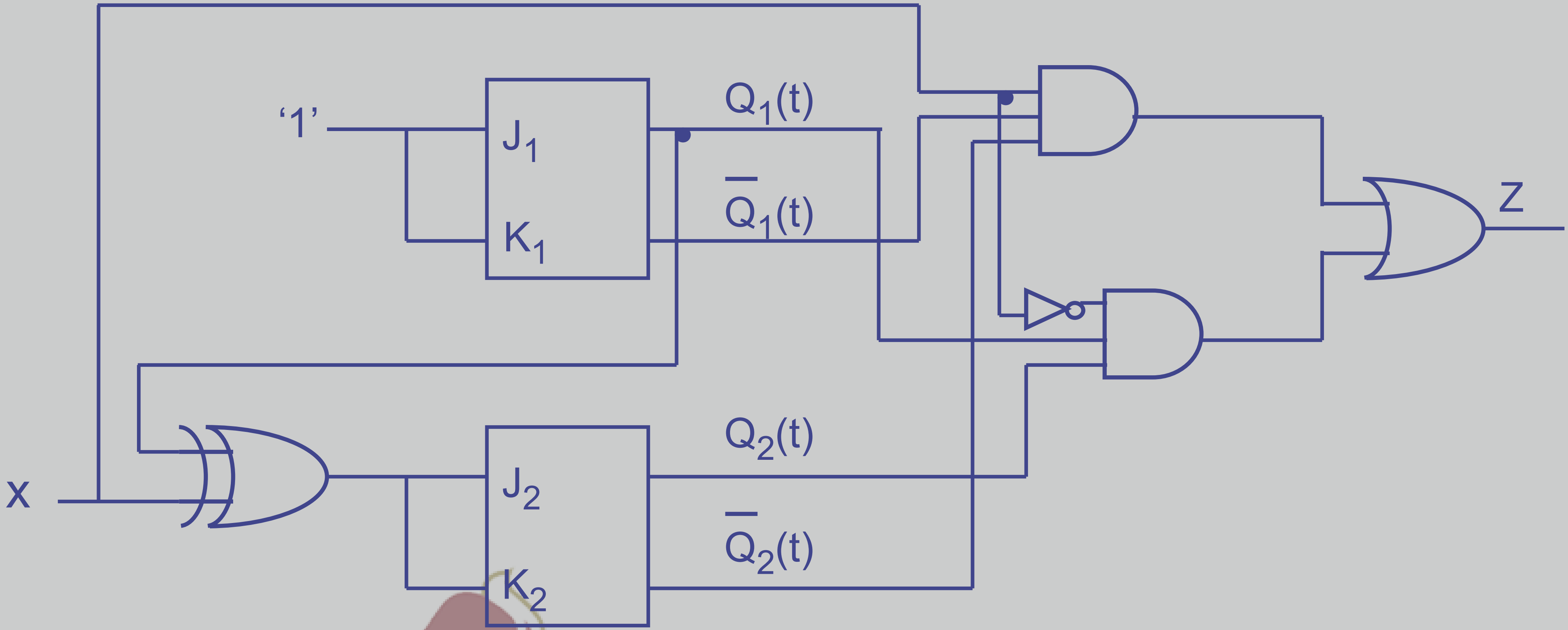
جدول کارنا :

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

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

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

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

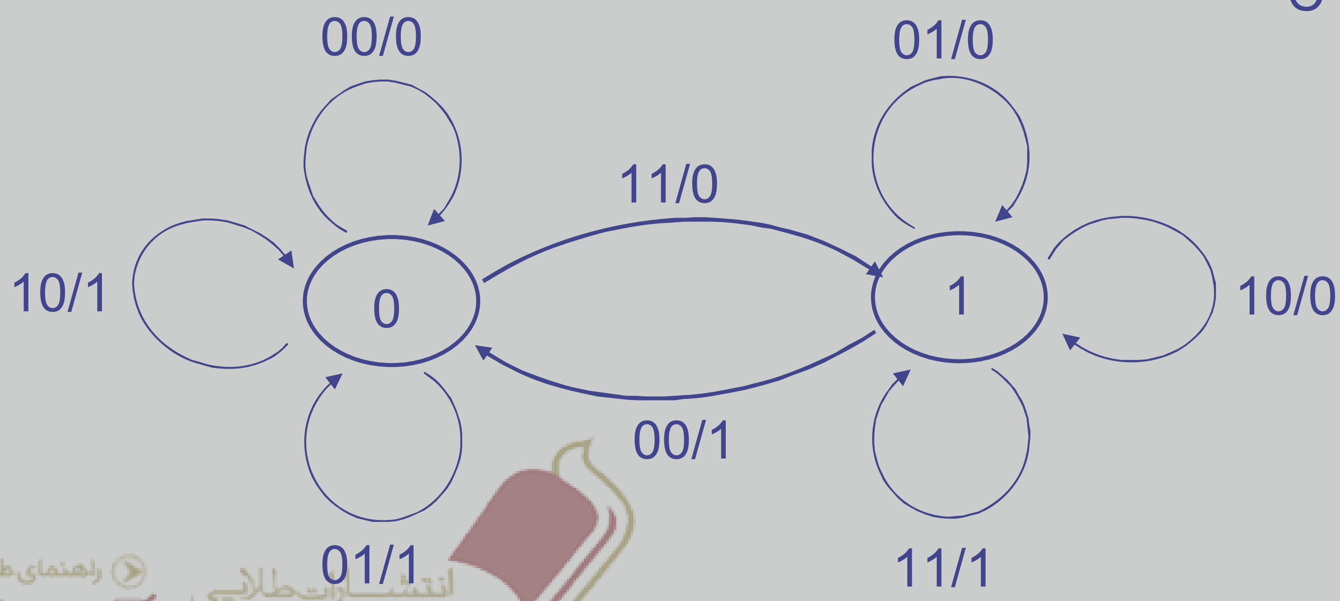

'1'
J1
K1
Q1(t)
Q̄1(t)
z
x
J2
K2
Q2(t)
Q̄2(t)

مثال 8: 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 : 2 بیت (a_i, b_i)

خروجی: حاصل جمع s_i

حالت : رقم نقلی C_{i+1}

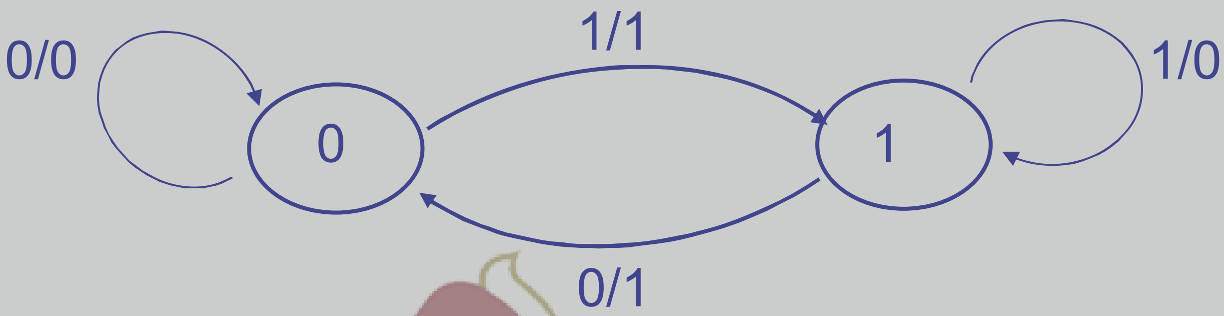


مثال 9 : مداری ترتیبی طراحی کنید که در هر کلاک پالس یک بیت از ورودی دریافت نموده و Parity زوج را روی بیت دریافت شده در این کلاک و بیت دریافت شده در کلاک قبل، در خروجی نمایش دهد.

ورودی : 1 بیت

خروجی:1 بیت p_e

حالت : بیت ما قبل



تمرین :

تمرین 1 : 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.

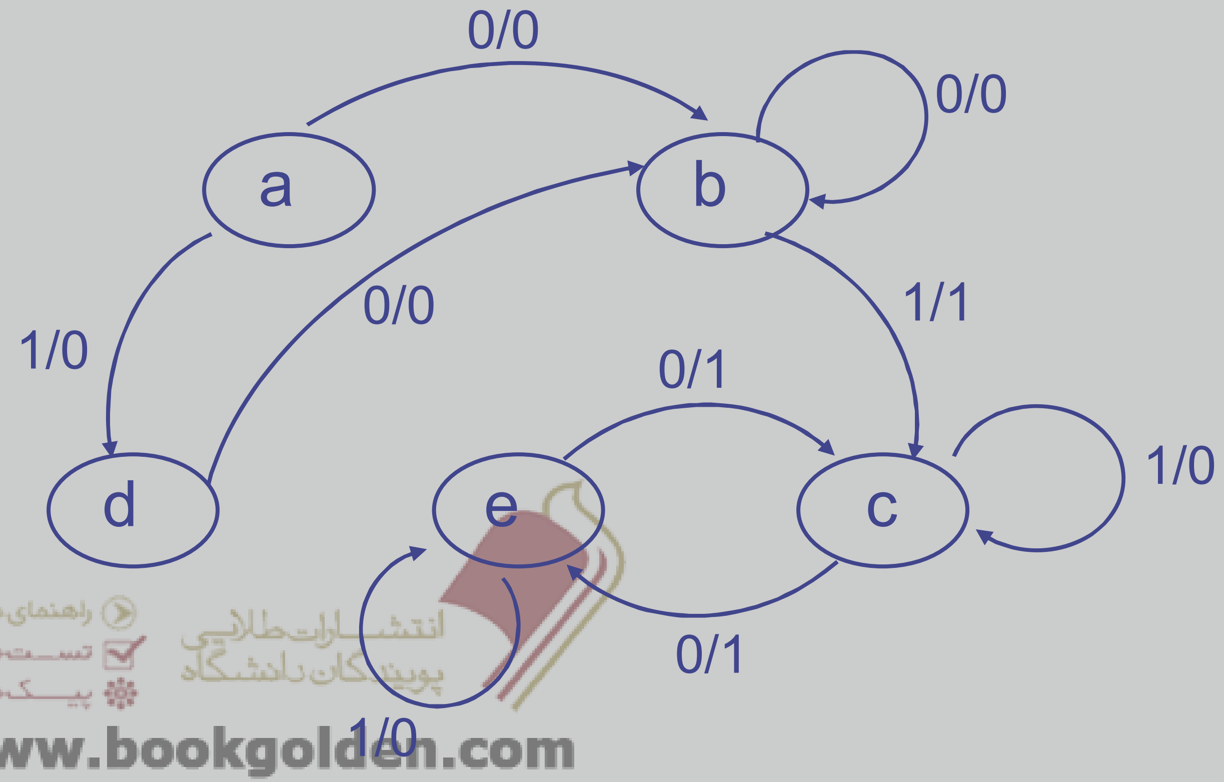
تمرین 2 : مداری ترتیبی طراحی نمایید که در هر کلاک پالس و Parity زوج را روی کلیه بیت های ماقبل بیت جاری محاسبه کند.

تمرین 3 : 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 :

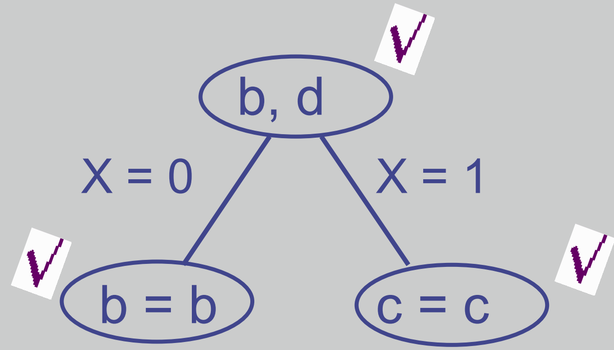
مثال 10 :

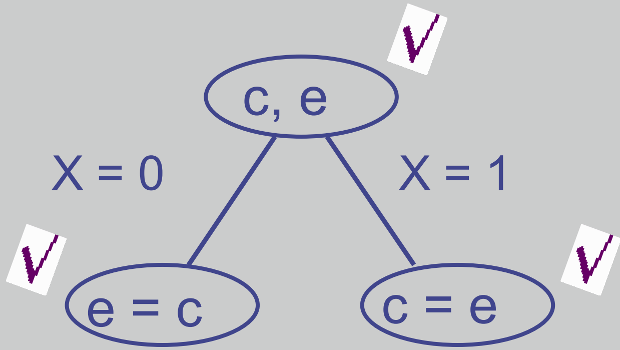
اولین گام : دسته بندی State ها بر اساس خروجی ها



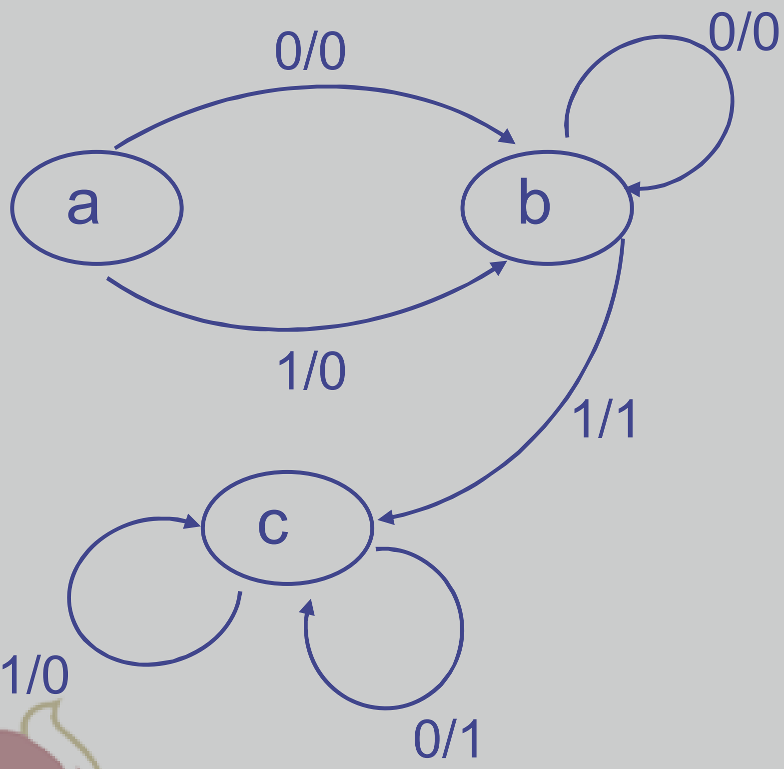
دومین گام :

		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~~



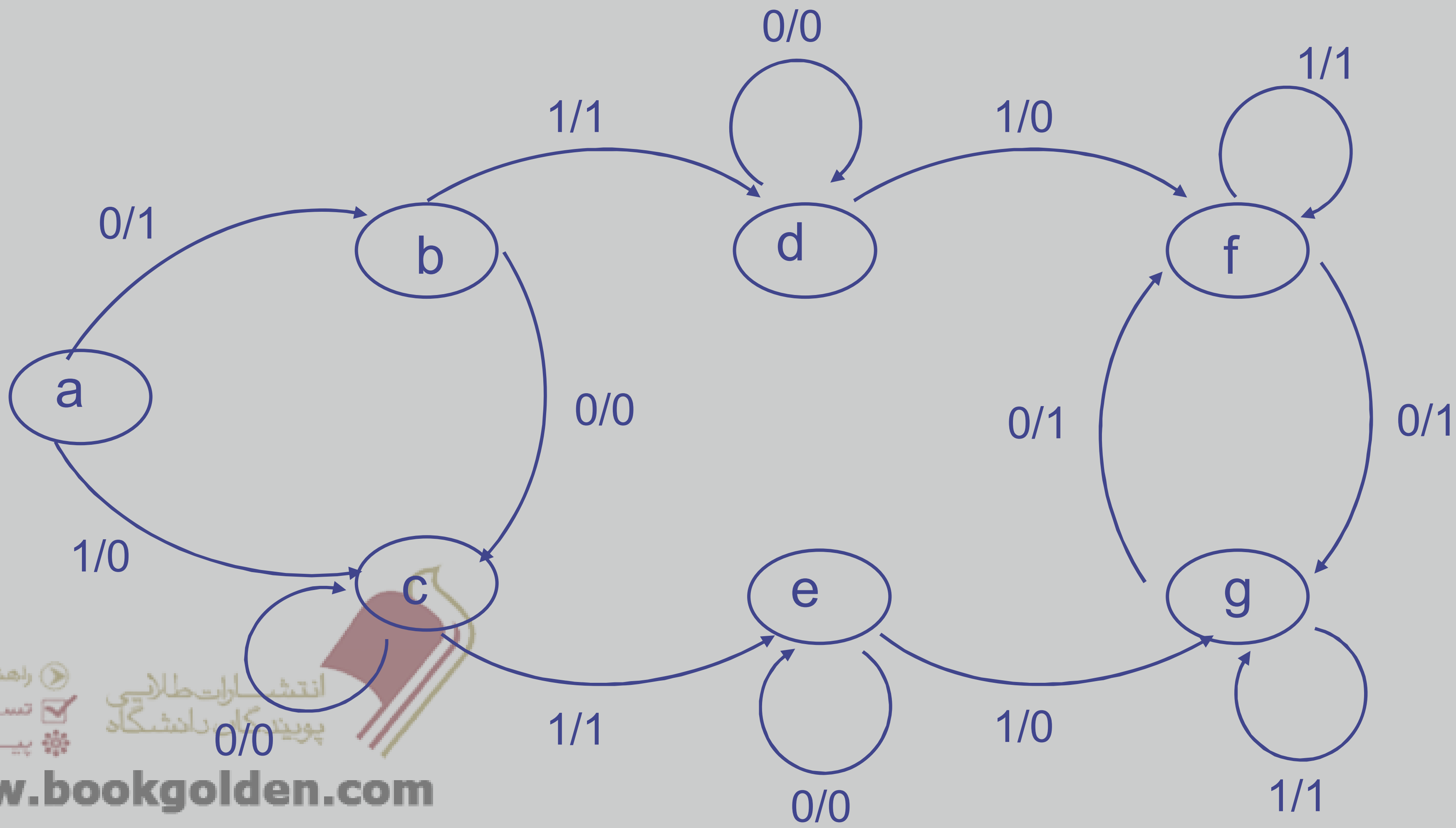


State Diagram معادل :



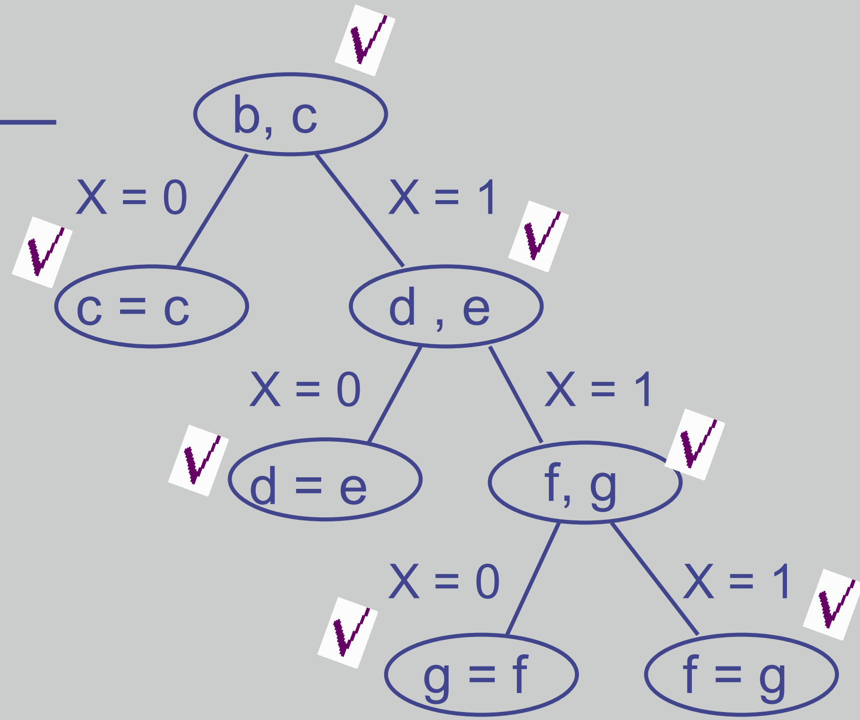
کمینه کردن یک State Diagram : (ادامه)

مثال 11 :

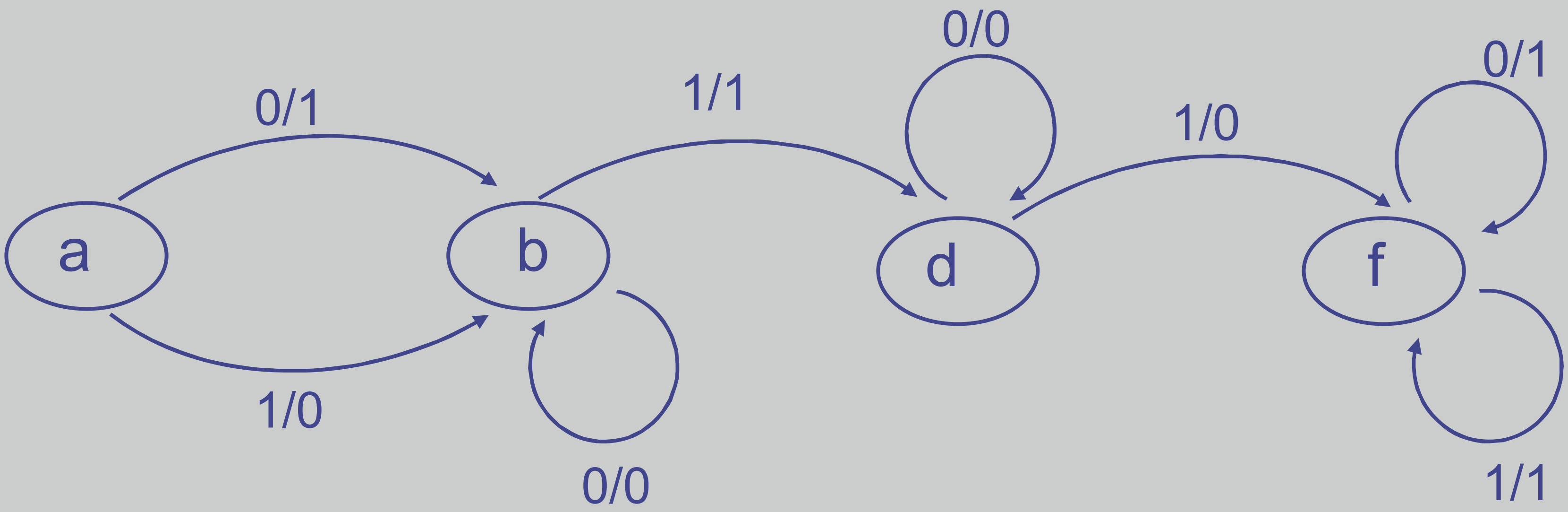


مثال 11 : (ادامه)

		حالت بعدی		خروجی	
		X = 0	X = 1	X = 0	X = 1
10	a	b	~~c~~ b	1	0
01	b	~~c~~ b	d	0	1
~~01~~	~~c~~	~~c~~	~~e~~	~~0~~	~~1~~
00	d	d	f	0	0
~~00~~	~~e~~	~~e~~	~~g~~	~~0~~	~~0~~
11	f	~~g~~ f	f	1	1
~~11~~	~~g~~	~~f~~	~~g~~	~~1~~	~~1~~



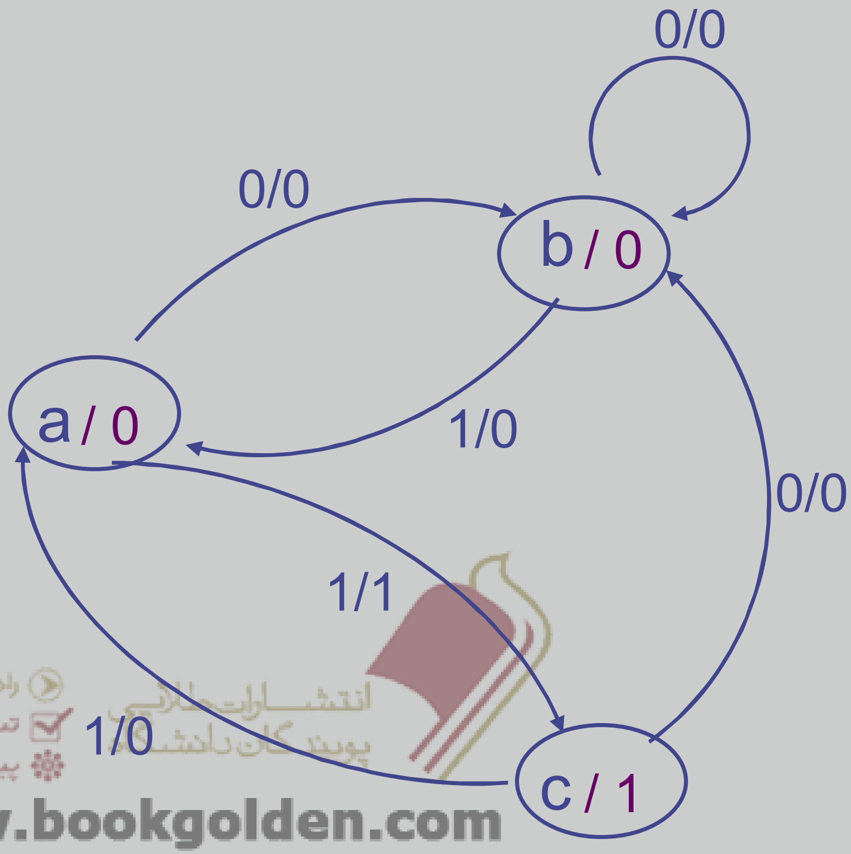
State Diagram معادل :



مدارها :

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

مور (mor) : خروجی تابعی از حالت است.

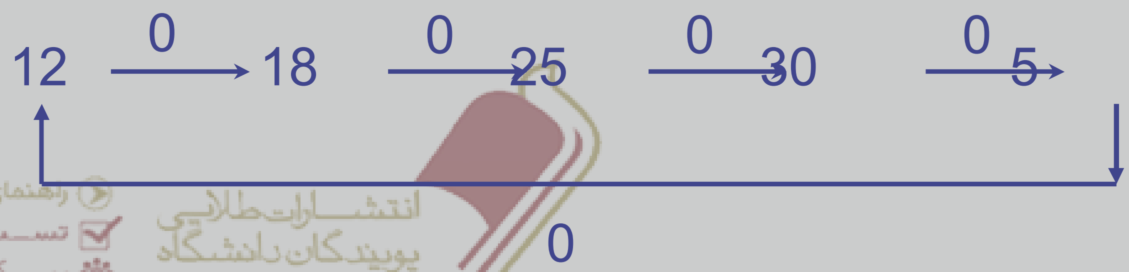


تمرین :

تمرین 1 : مداری ترتیبی طراحی کنید که به عنوان یک تشخیص دهنده ی الگو، الگوی بیتی 1101 را تشخیص دهد و به ازای آن خروجی را Setنماید. توجه داشته باشید که در هر کلاک پالس یک بیت از ورودی دریافت می شود.

تمرین 2 : مداری ترتیبی طراحی نمایید که با رشته بیت ورودی برخورد عددی داشته باشد و در صورتی که عدد دریافت شده، مضرب 5 بود، خروجی را Setکند، در غیر این صورت خروجی صفر باشد.

تمرین 3 : 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 1 در هر مرحله مانند، دو بار ورودی صفر عمل می کند.

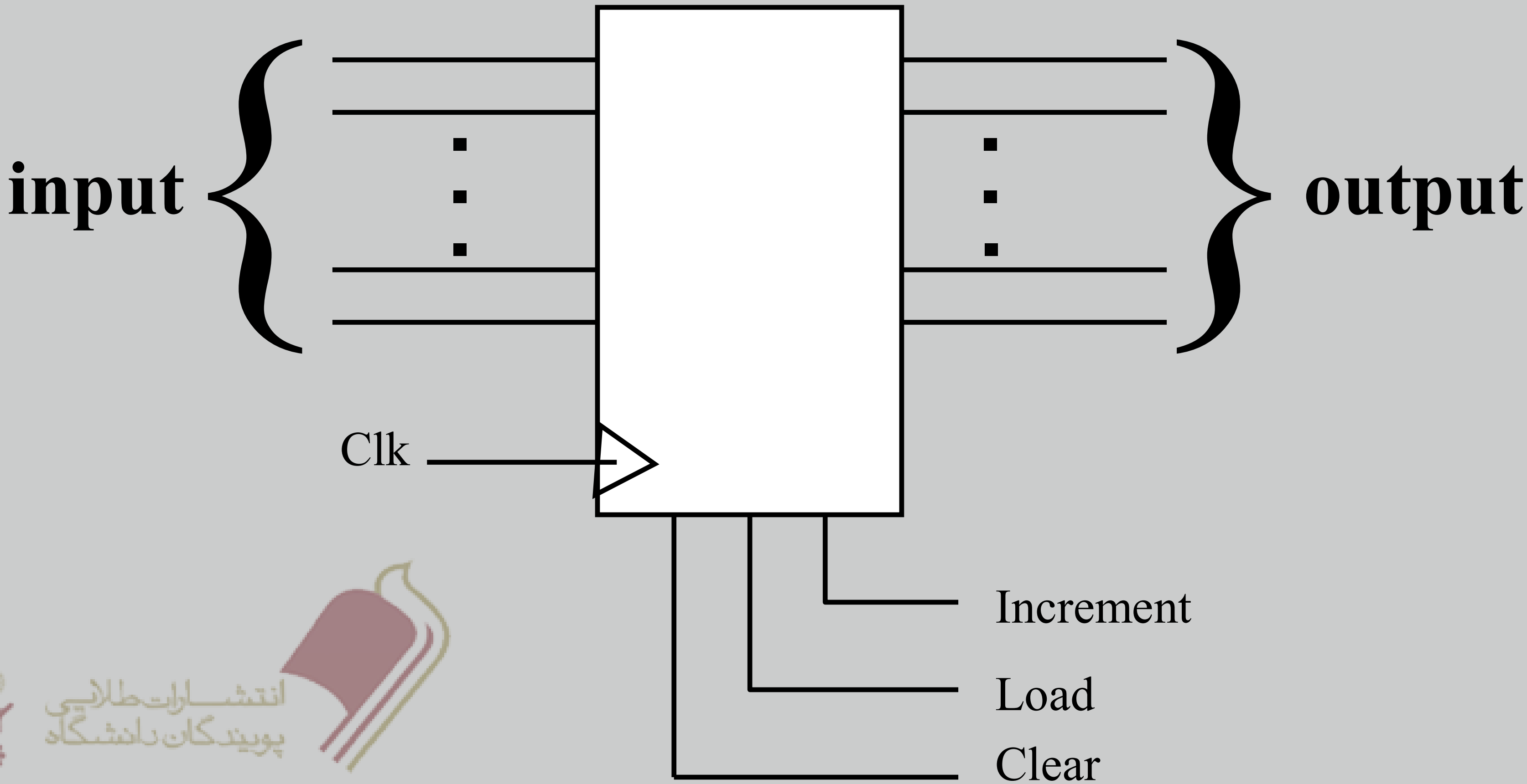


فصل 7

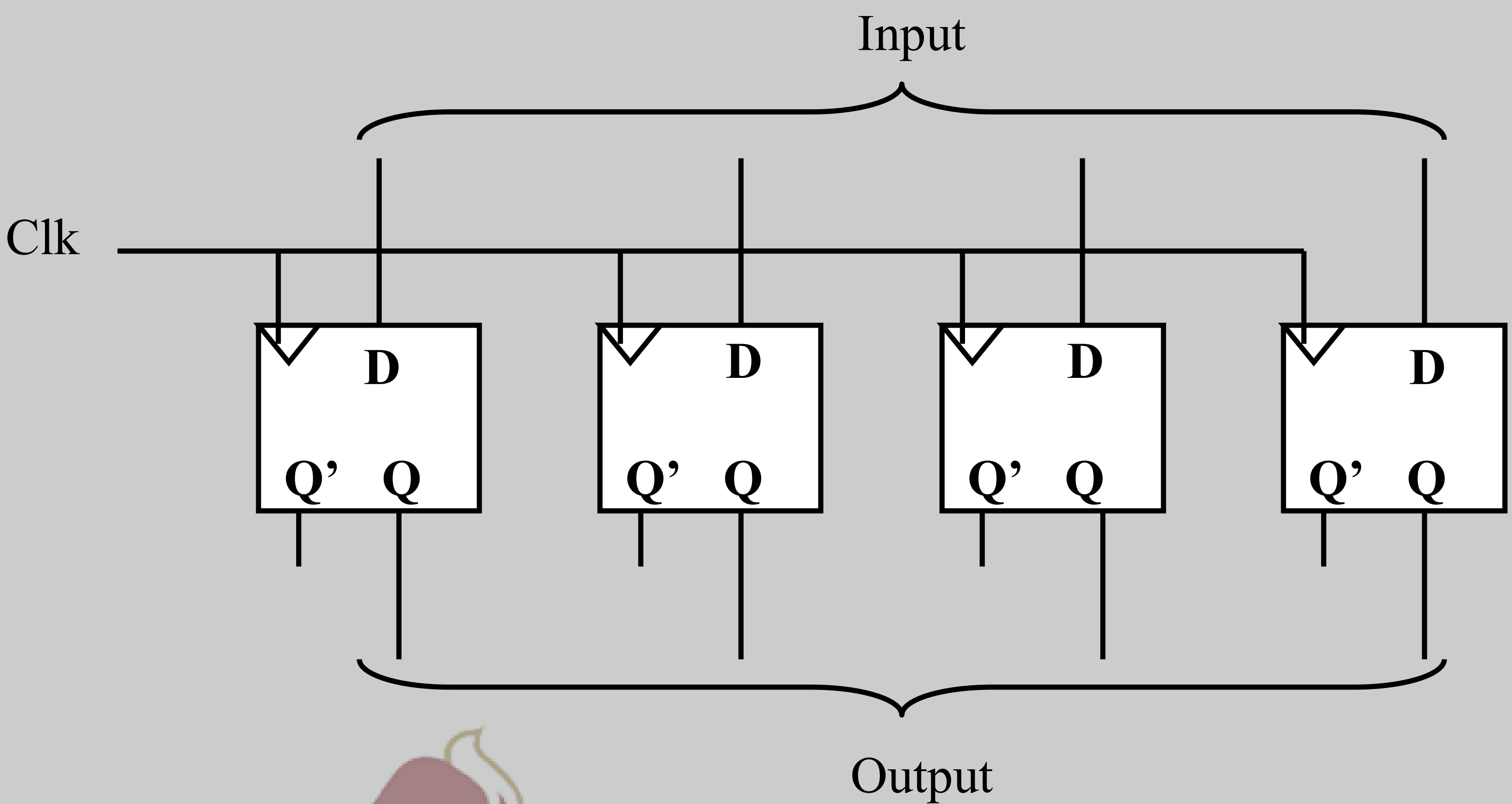
ثبات ها و شیفت رجیستر

فهرست مطالب

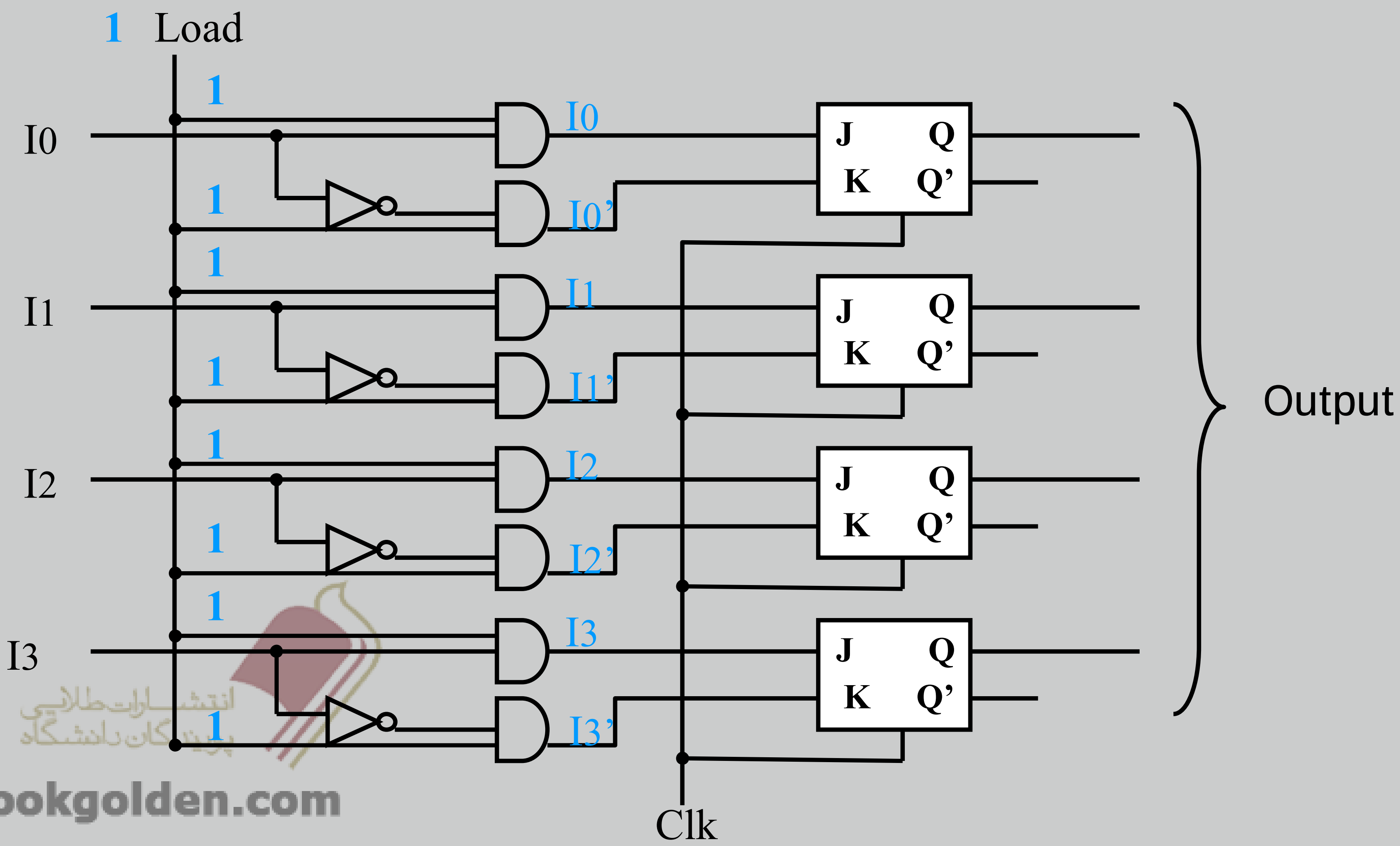
طرح بلوک دیاگرامی ثبات



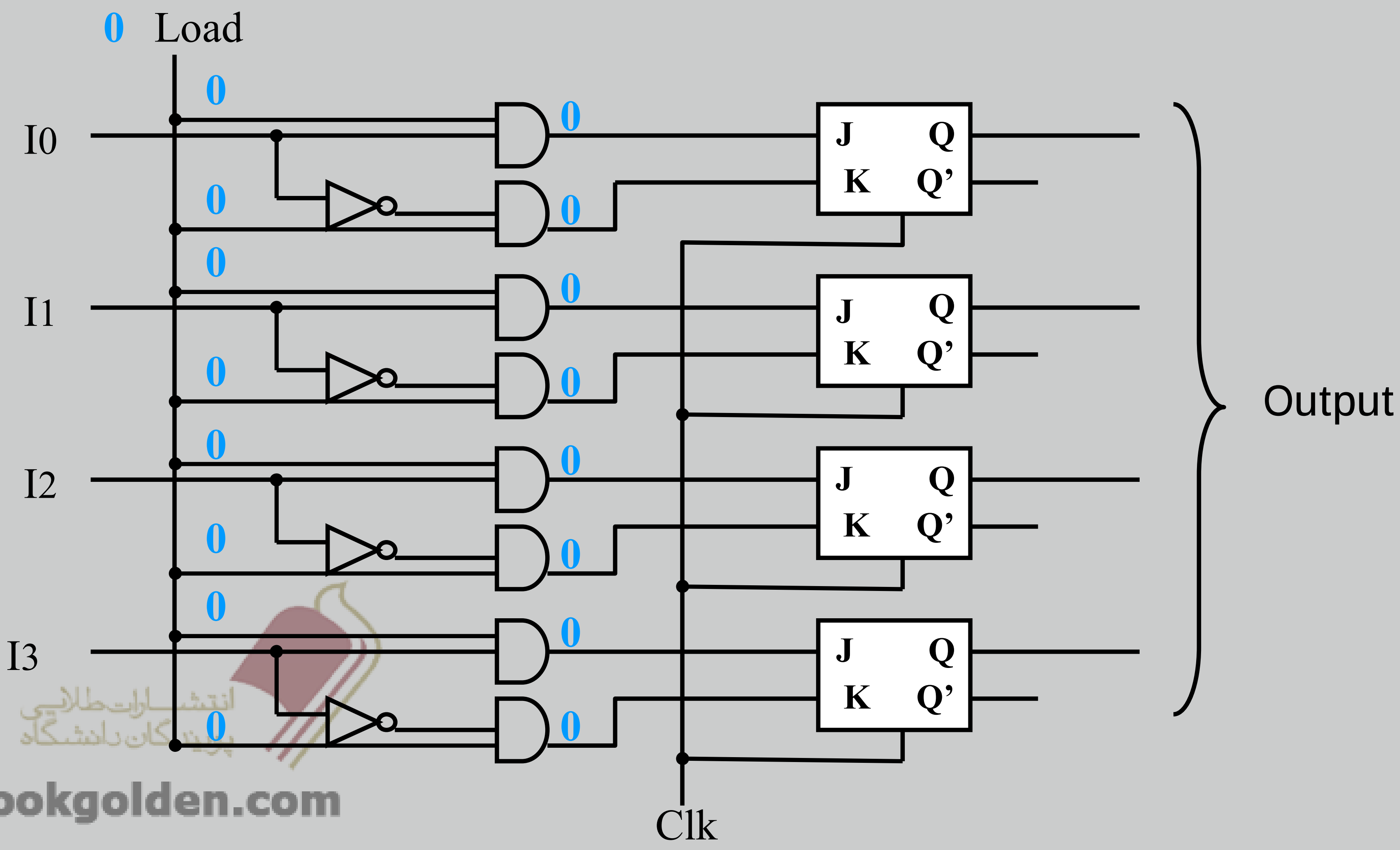
طرح ساده یک ثبات با فیلیپ فلاپ D



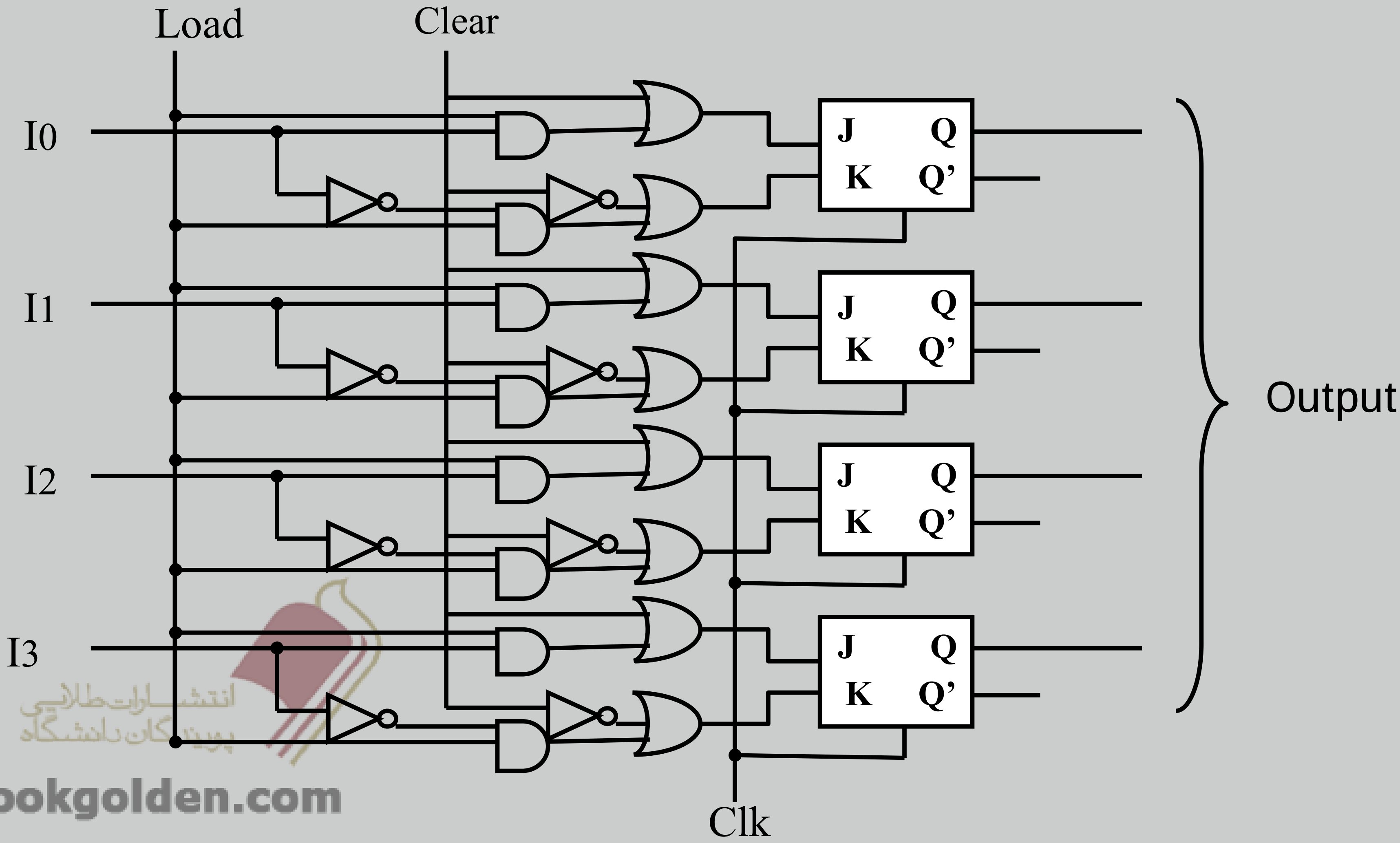
طرح یک ثبات با فیلیپ فلاپJK و پایه Load



طرح یک ثبات با فیلیپ فلاپJK و پایه Load



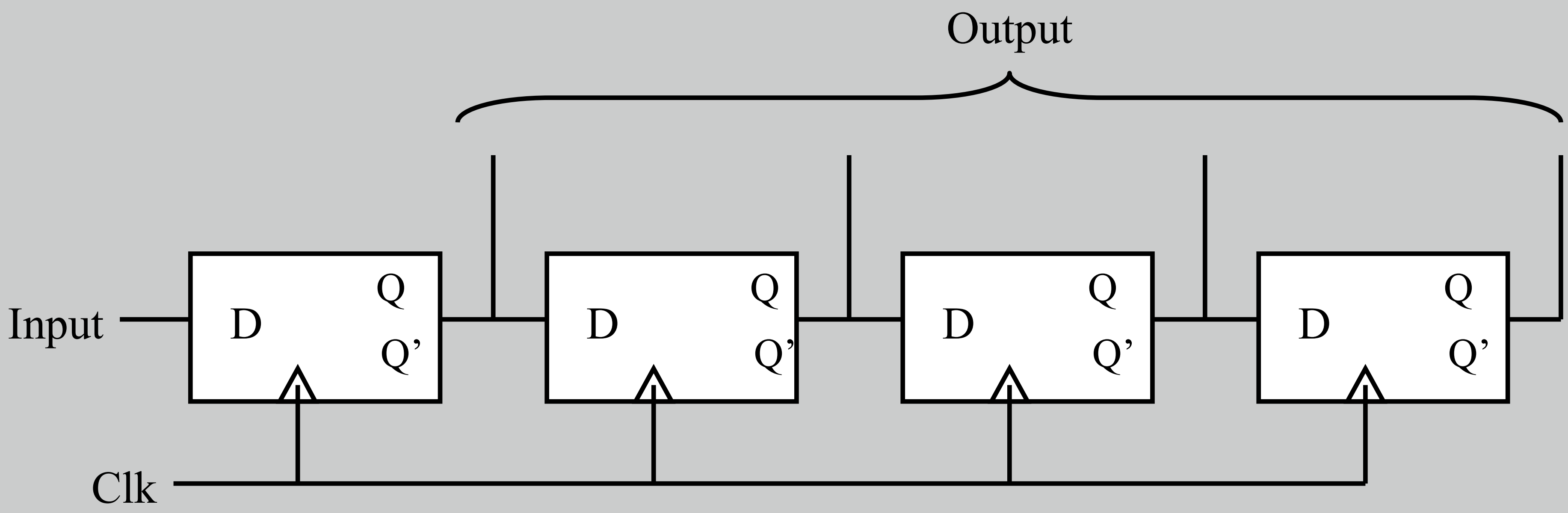
طرح یک ثبات با پایه Load و Clear



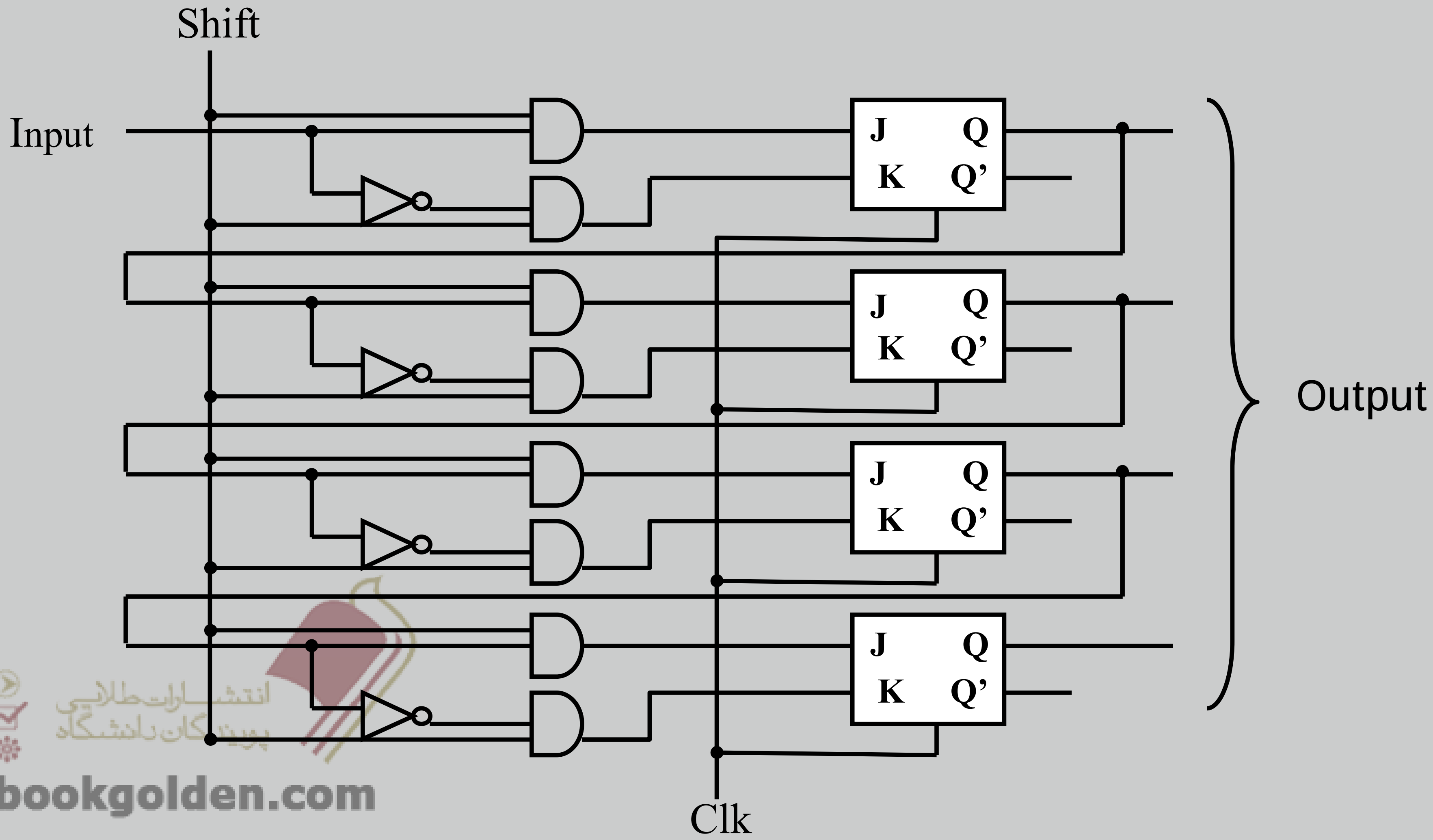
تمرین:

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

شیفت رجیستربا فیلیپ فلاپ D



شیفت رجیستربا فیلیپ فلاپ JK



شمارنده

- سنکرون(هنگام):در این نوع تمام واحدهای ترتیبی مداربا یک Clkکار می کنند.
- آسنکرون(ناهمگام):در این نوع هر واحد Clk مجزایی دارد.

شمارنده

- منظم
  - بالا شمار
  - پائین شمار
- نامنظم

شمارنده 3 بیتی

بیت 0

Q_2	Q_1	Q_0
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 3 بیتی (ادامه)

بیت 1

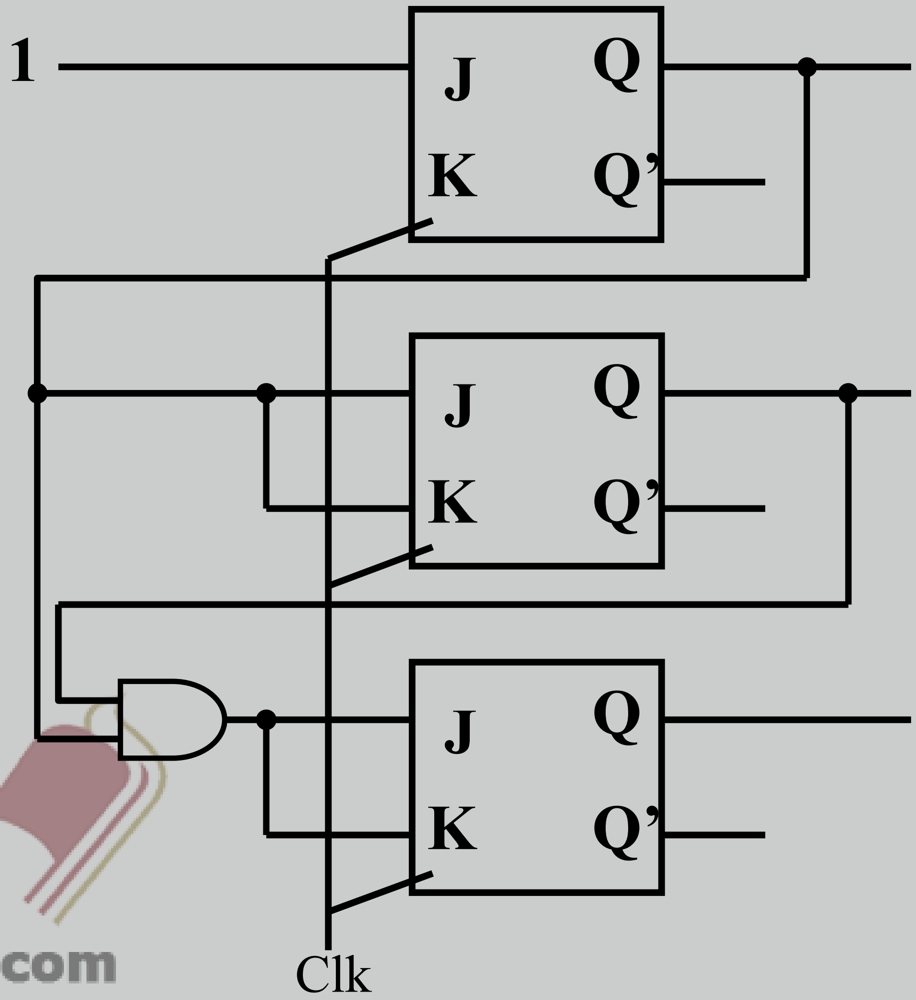
Q2	Q1	Q0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

شمارنده 3 بیتی (ادامه)

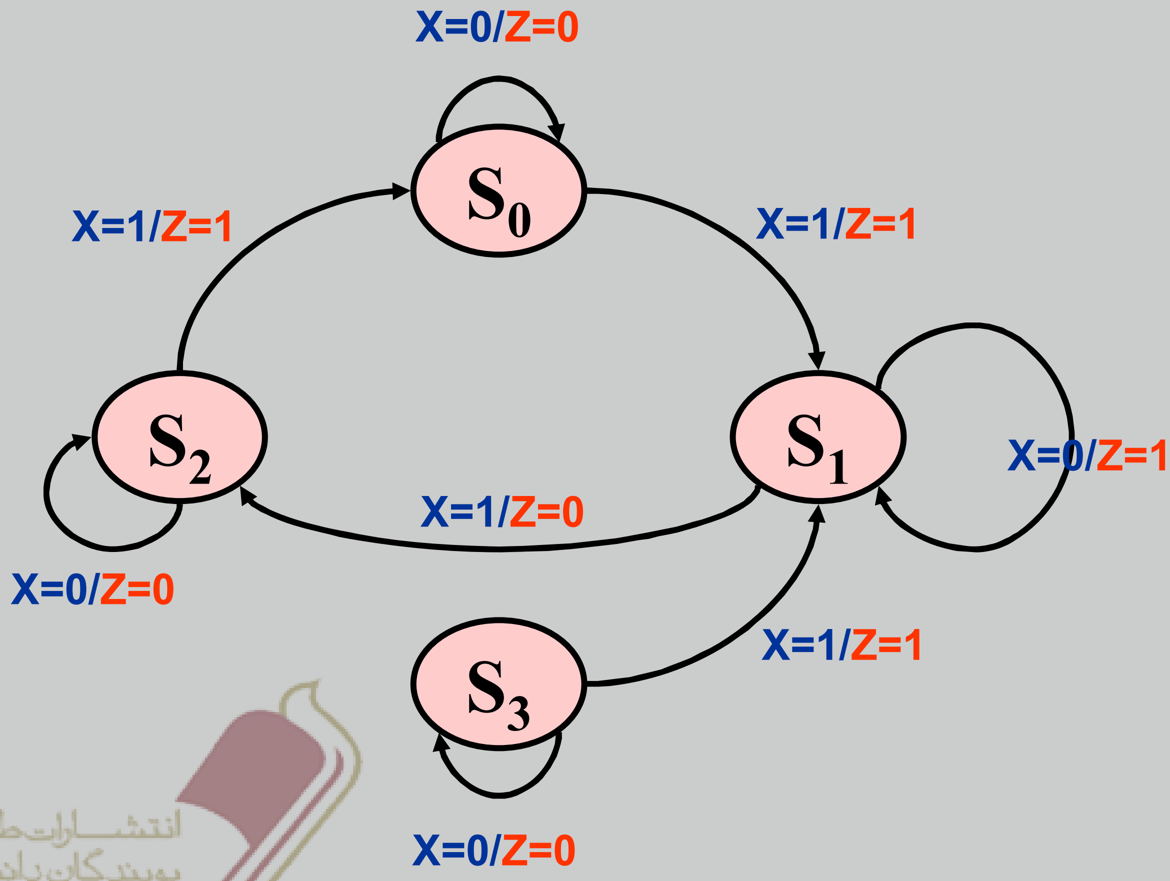
بیت 2

Q_2	Q_1	Q_0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

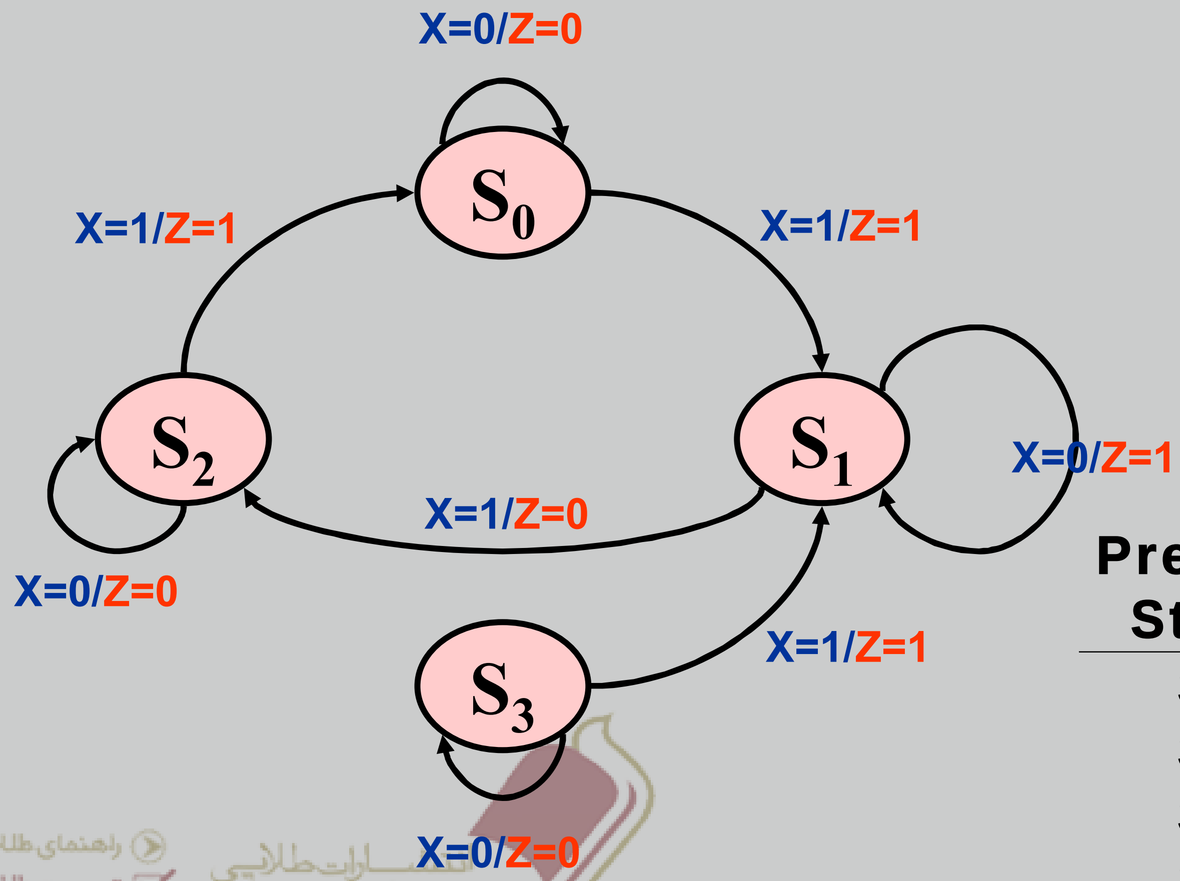
مدار یک شمارنده 3 بیتی سنکرون



مثالی از یک ماشین میلی:

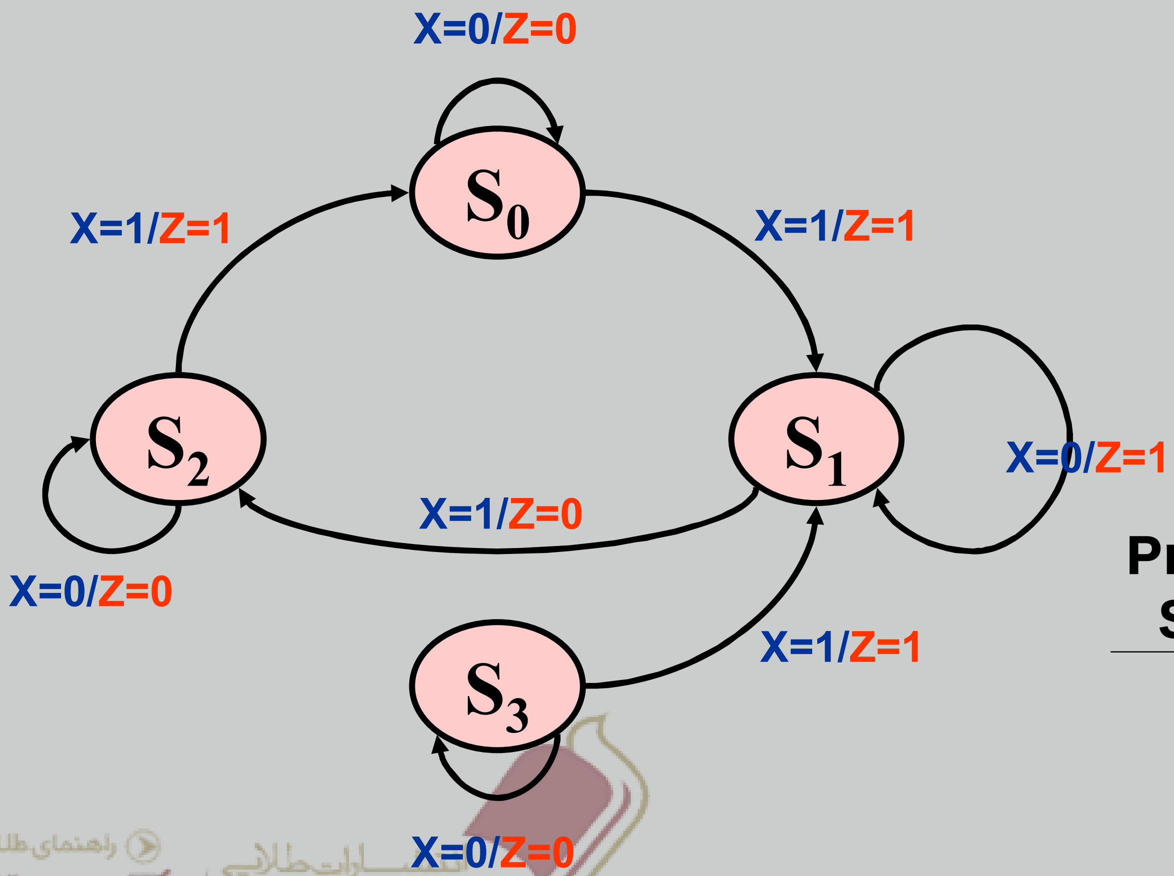


مثالی از یک ماشین میلی:



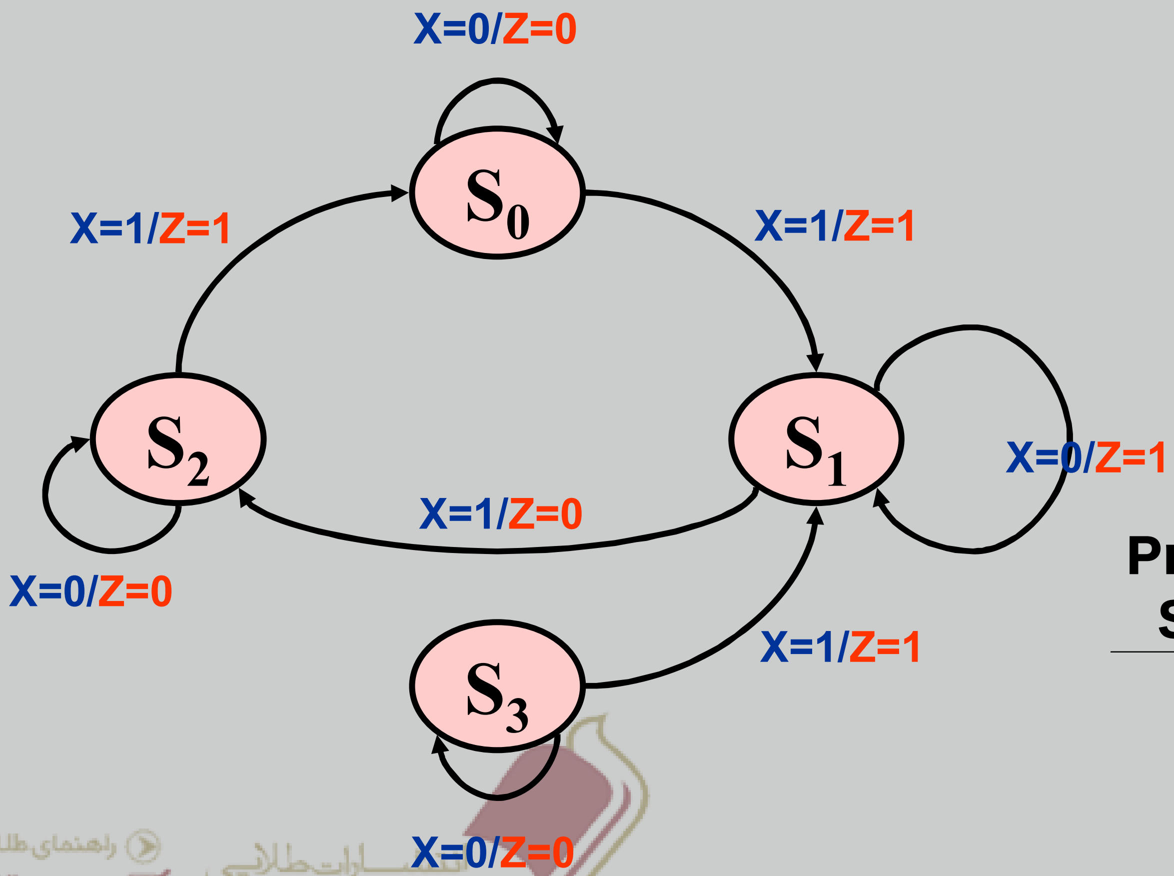
Present State	Next State X=0	Next State X=1	Output X=0	Output X=1
S_0	S_0	S_1	0	1
S_1				
S_2				
S_3				

مثالی از یک ماشین میلی:



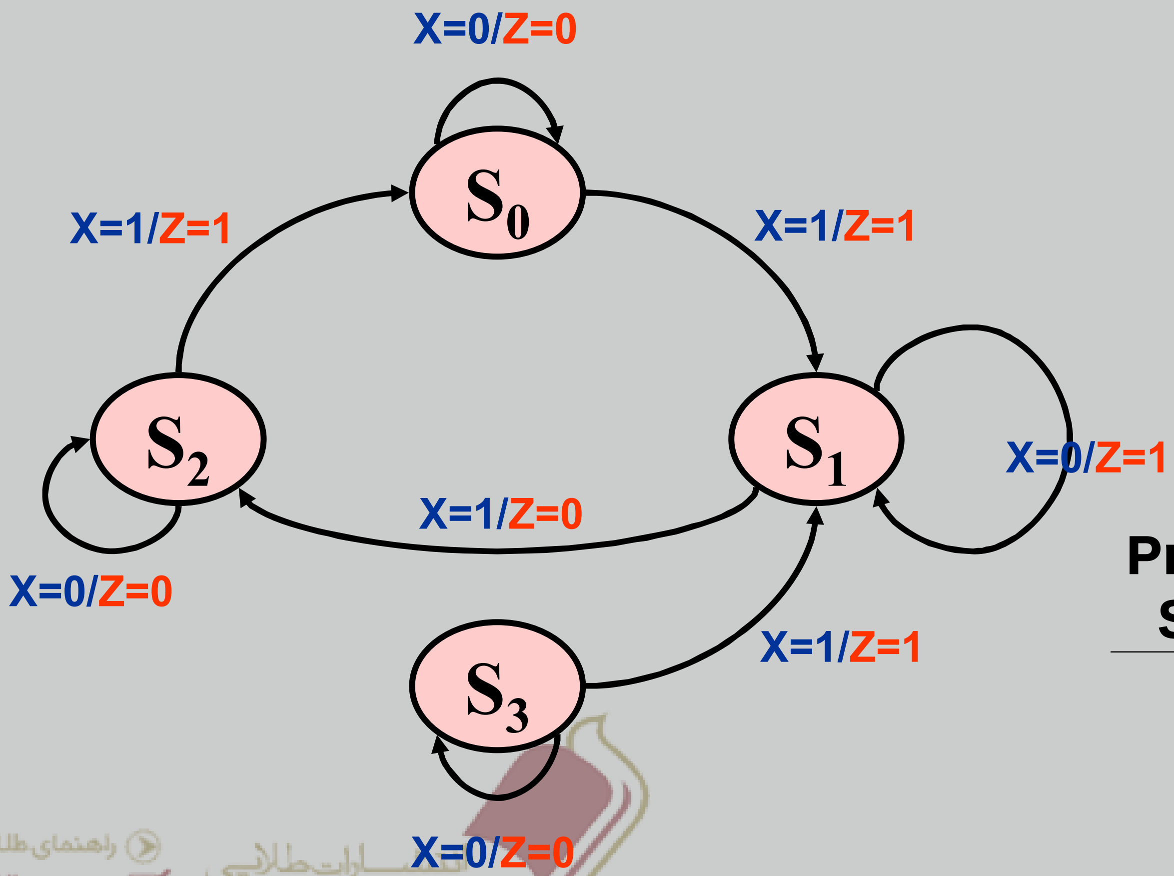
Present State	Next State X=0	Next State X=1	Output X=0	Output X=1
S_0	S_0	S_1	0	1
S_1	S_1	S_2	1	1
S_2				
S_3				

مثالی از یک ماشین میلی:



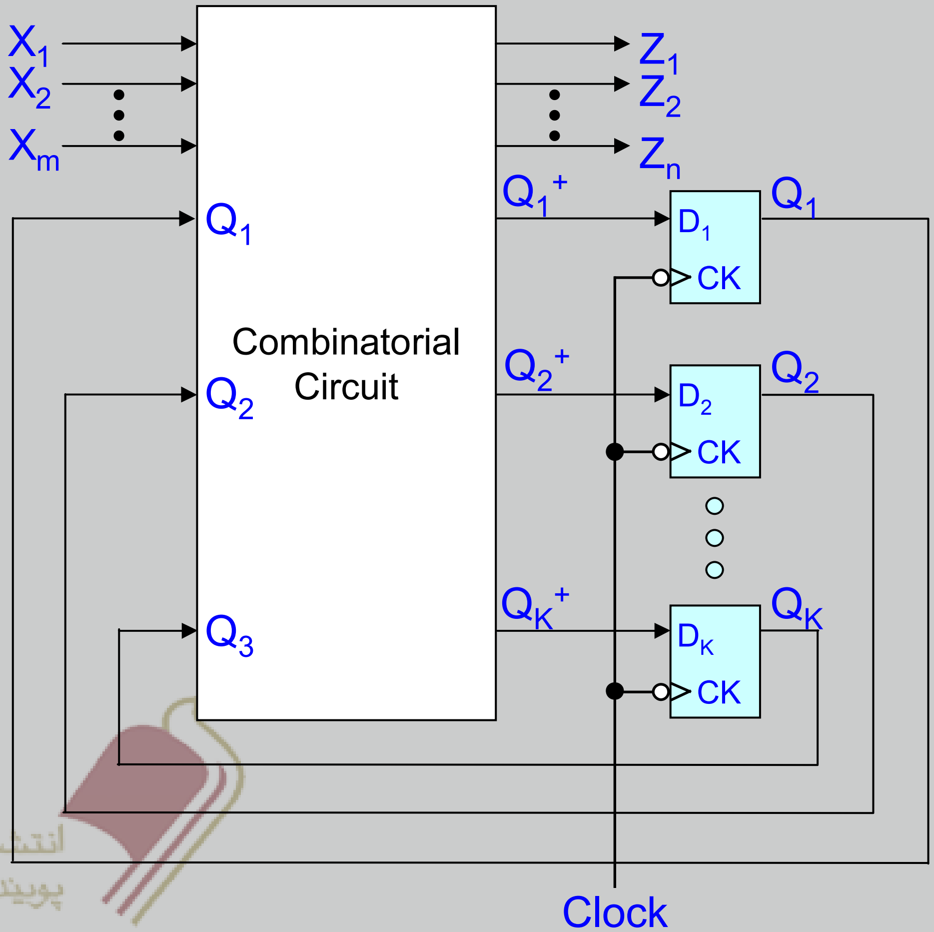
Present State	Next State X=0	Next State X=1	Output X=0	Output X=1
S_0	S_0	S_1	0	1
S_1	S_1	S_2	1	1
S_2	S_2	S_0	0	1
S_3				

مثالی از یک ماشین میلی:



Present State	Next State X=0	Next State X=1	Output X=0	Output 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:



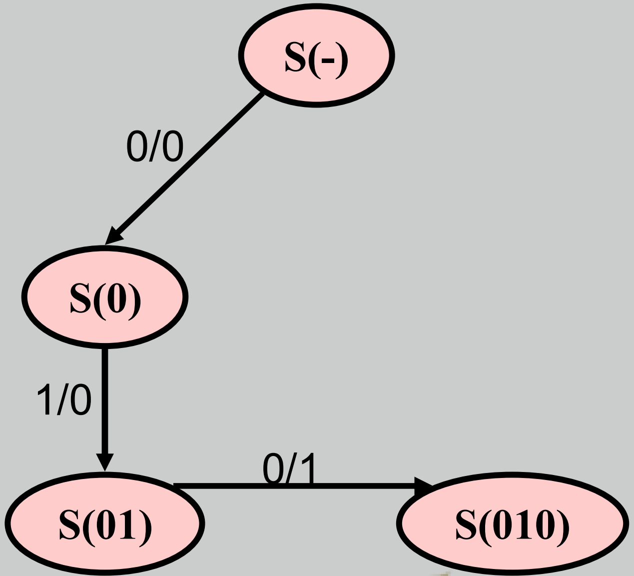
A More Complex Sequence Detector

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

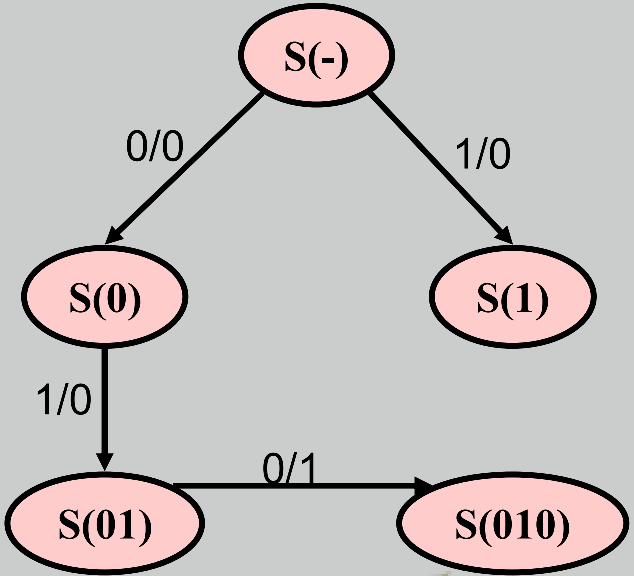
X = 0 0 1 0 1 0 0 1 0 0 0 1 0 0 1 1 0
Z = 0 0 0 1 0 1 0 1 1 0 0 0 1 0 1 0 0

Mealy Sequence Detector

Target Sequences:
010
1001



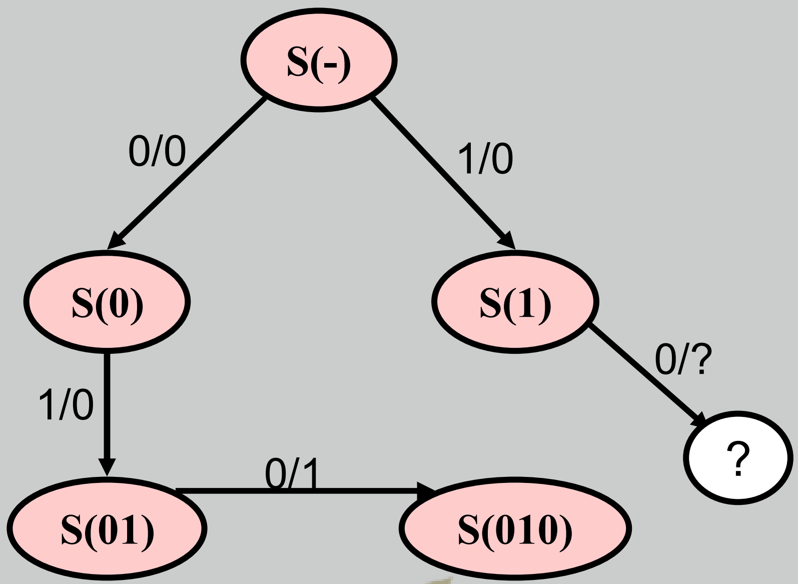
Mealy Sequence Detector



Target Sequences:
010
1001

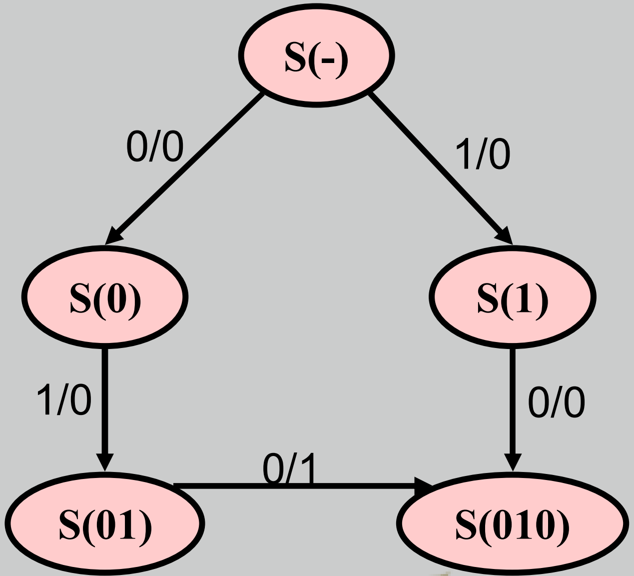
Mealy Sequence Detector

Target Sequences:
010
1001



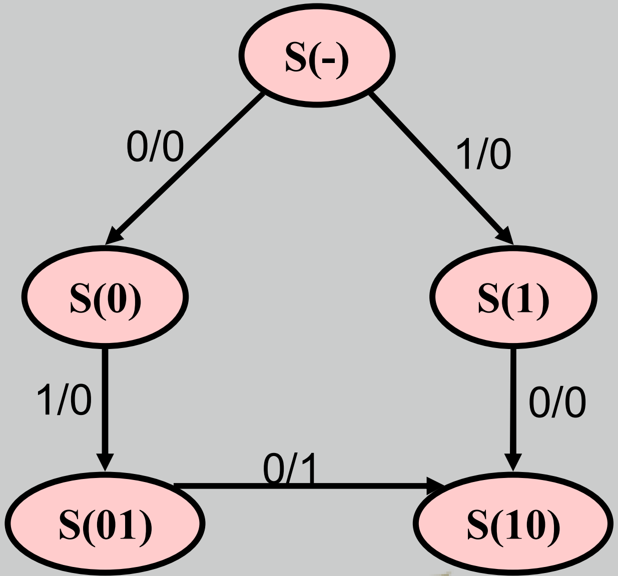
Mealy Sequence Detector

Target Sequences:
010
1001



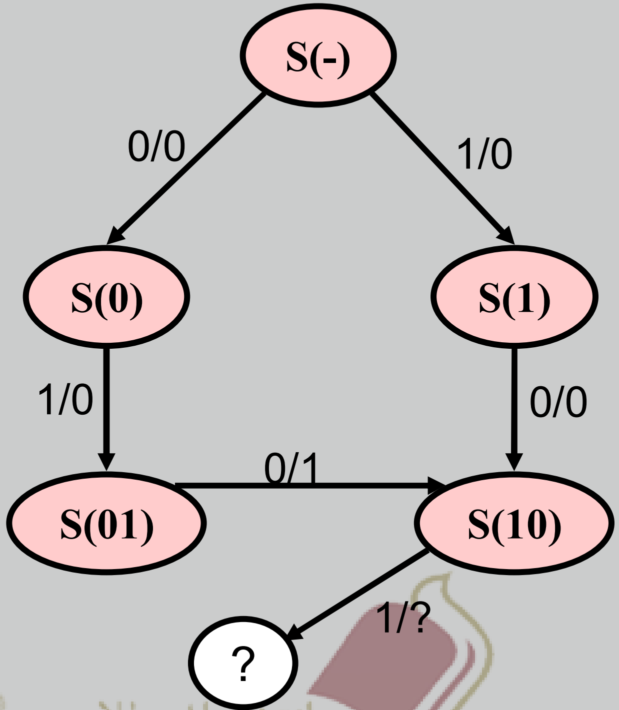
Mealy Sequence Detector

Target Sequences:
010
1001



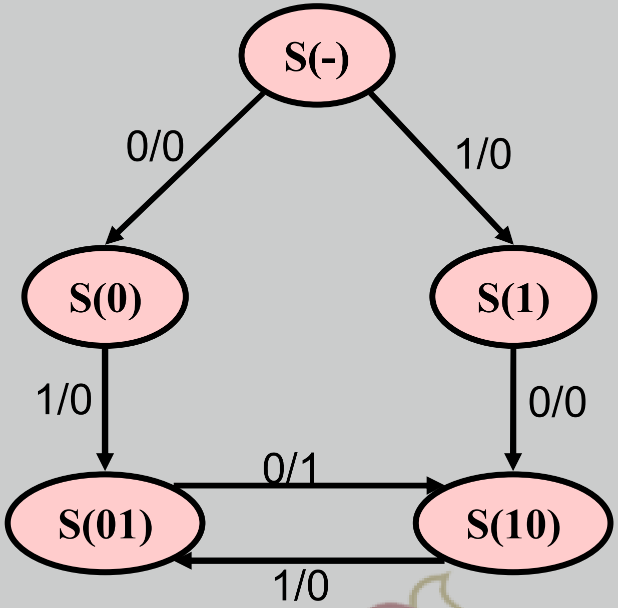
Mealy Sequence Detector

Target Sequences:
010
1001

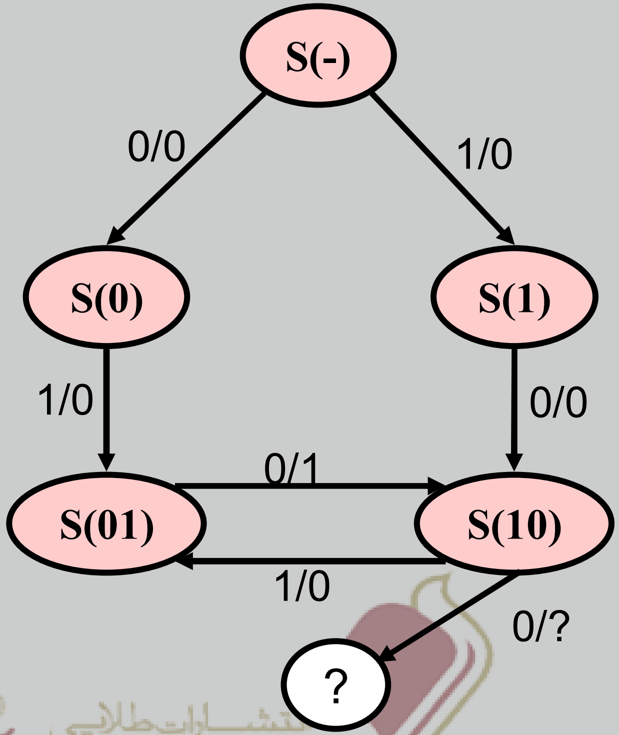


Mealy Sequence Detector

Target Sequences:
010
1001



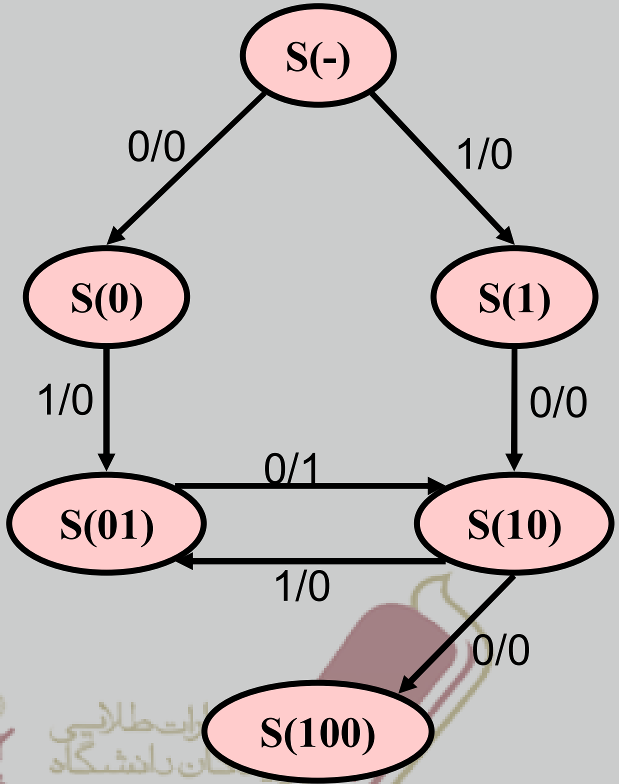
Mealy Sequence Detector



Target Sequences:
010
1001

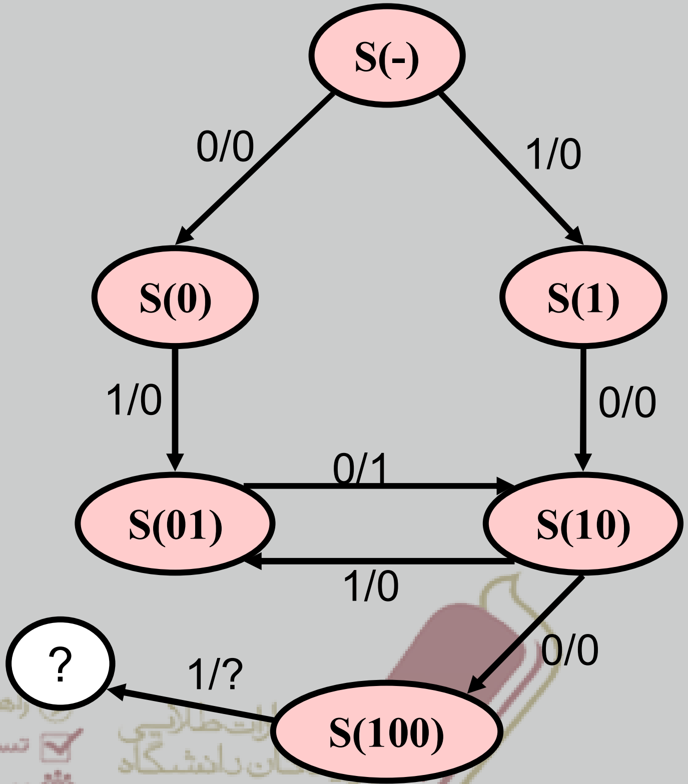
Mealy Sequence Detector

Target Sequences:
010
1001



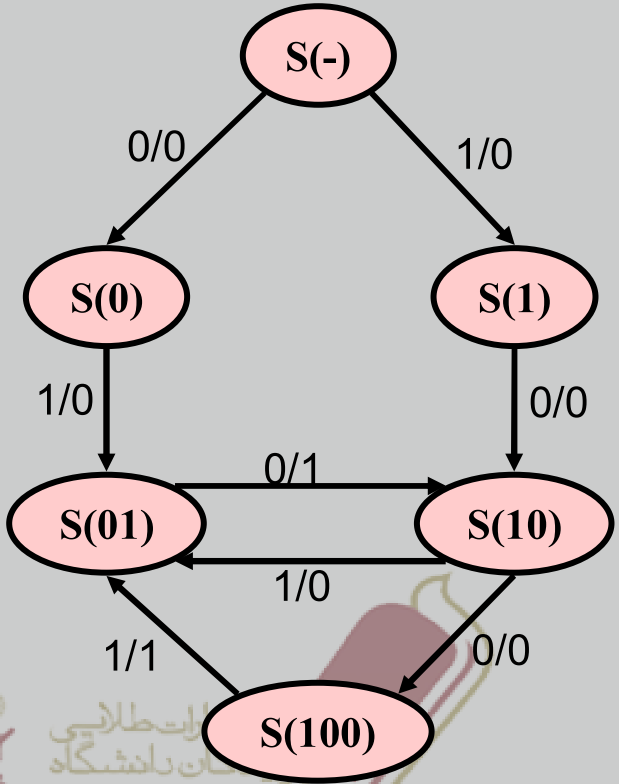
Mealy Sequence Detector

Target Sequences:
010
1001



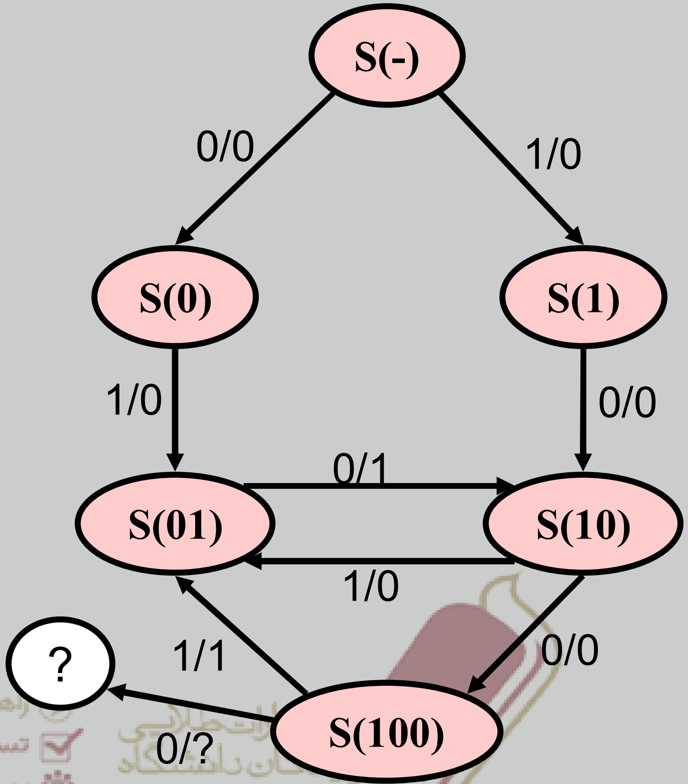
Mealy Sequence Detector

Target Sequences:
010
1001



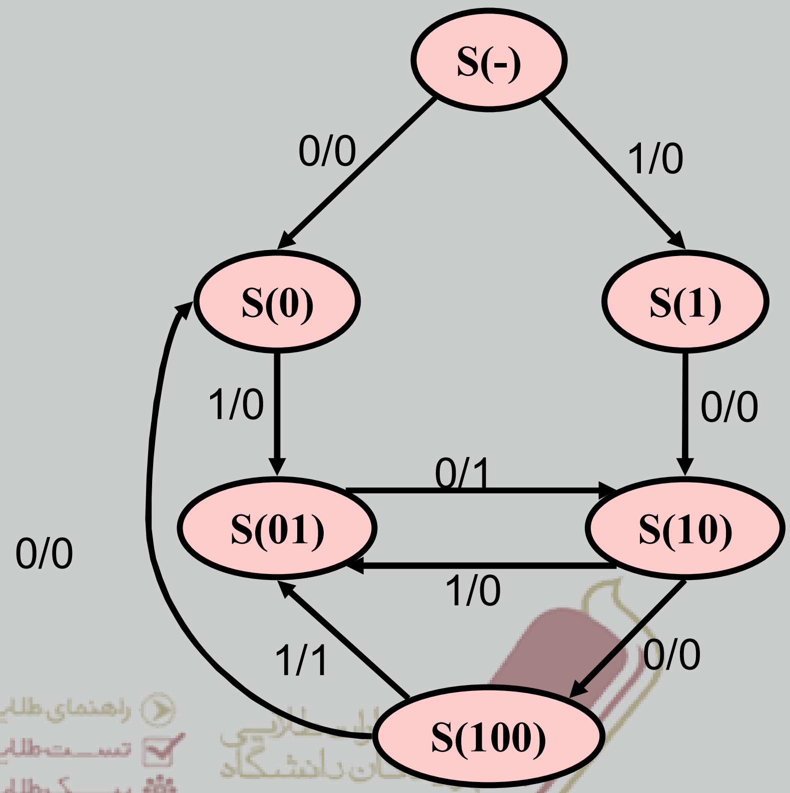
Mealy Sequence Detector

Target Sequences:
010
1001

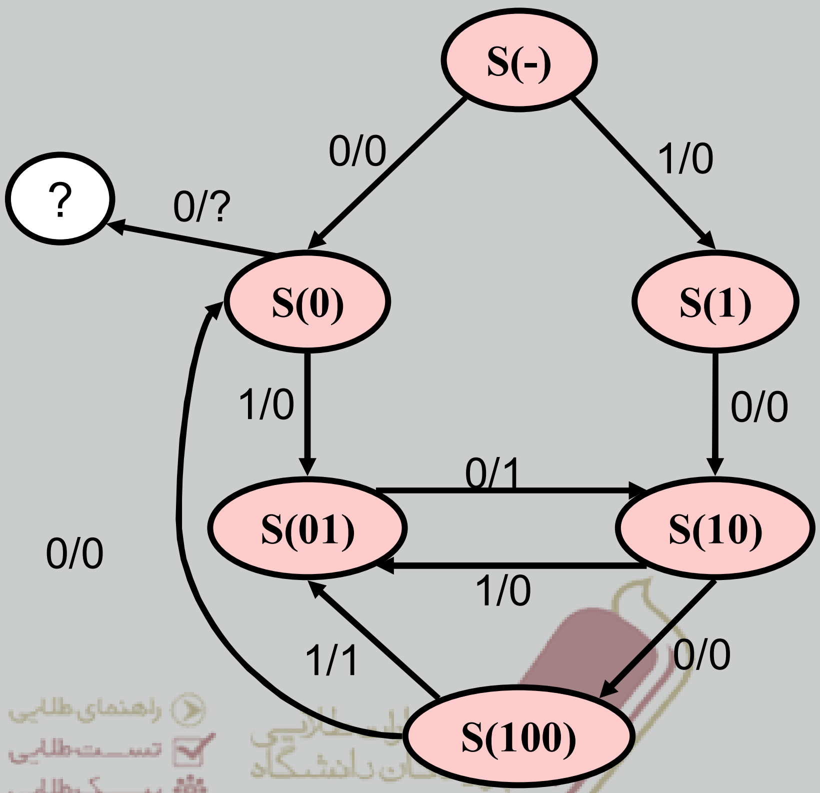


Mealy Sequence Detector

Target Sequences:
010
1001

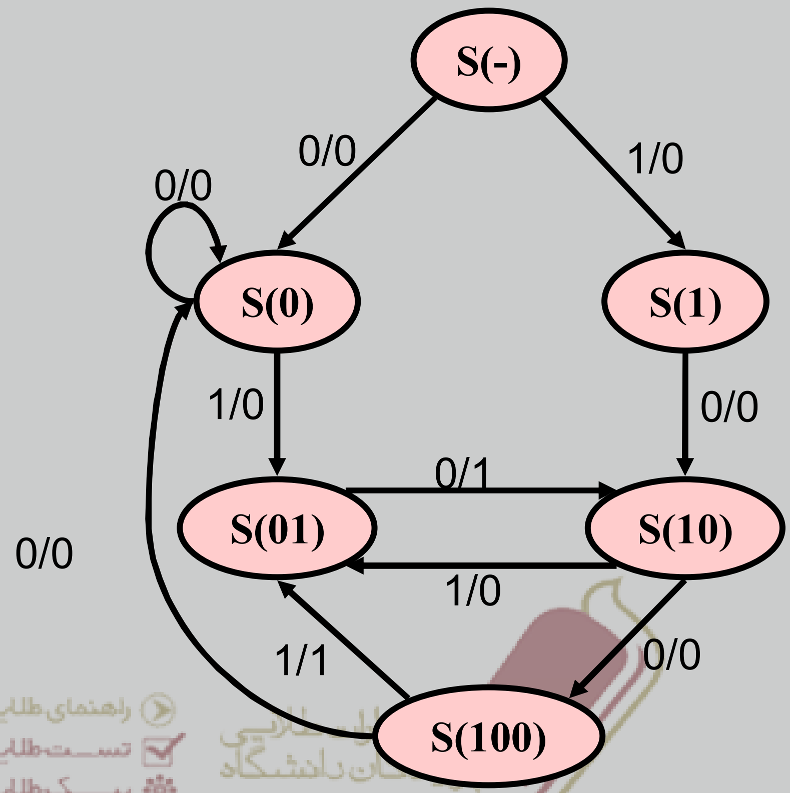


Mealy Sequence Detector



Target Sequences:
010
1001

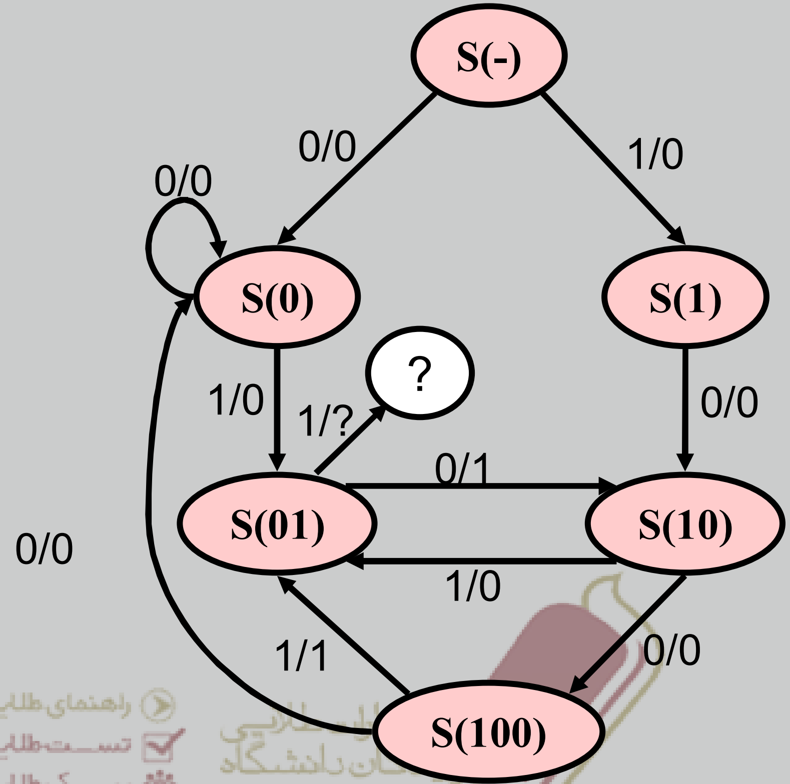
Mealy Sequence Detector



Target Sequences:
010
1001

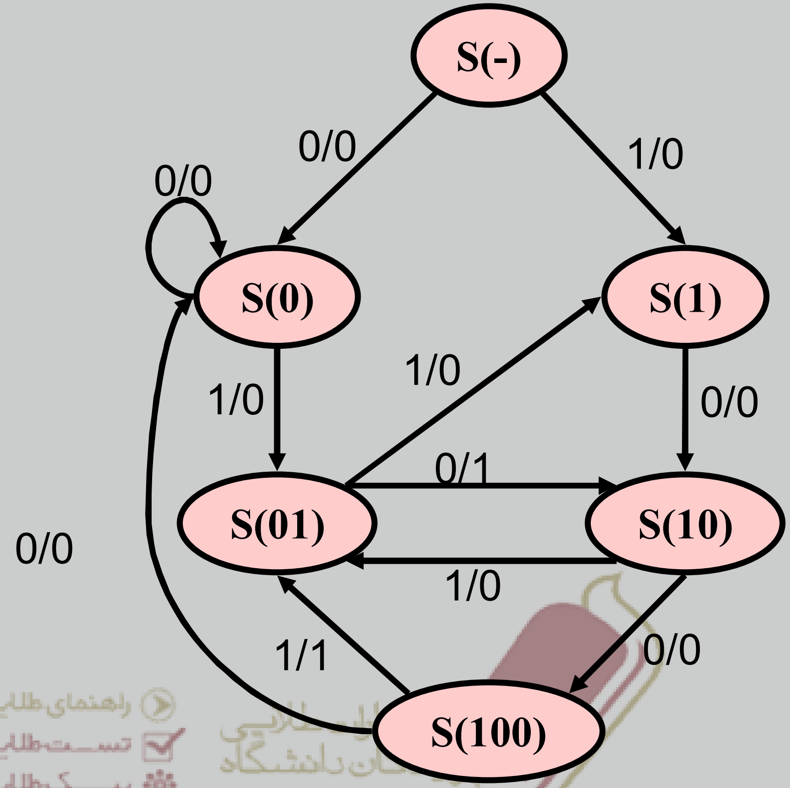
Mealy Sequence Detector

Target Sequences:
010
1001



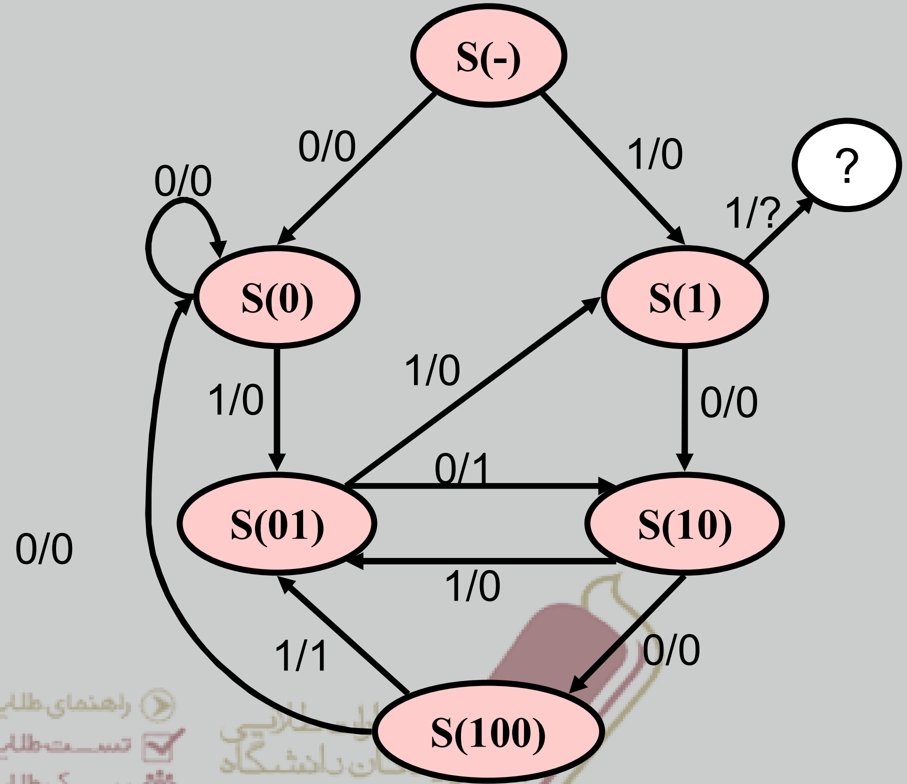
Mealy Sequence Detector

Target Sequences:
010
1001

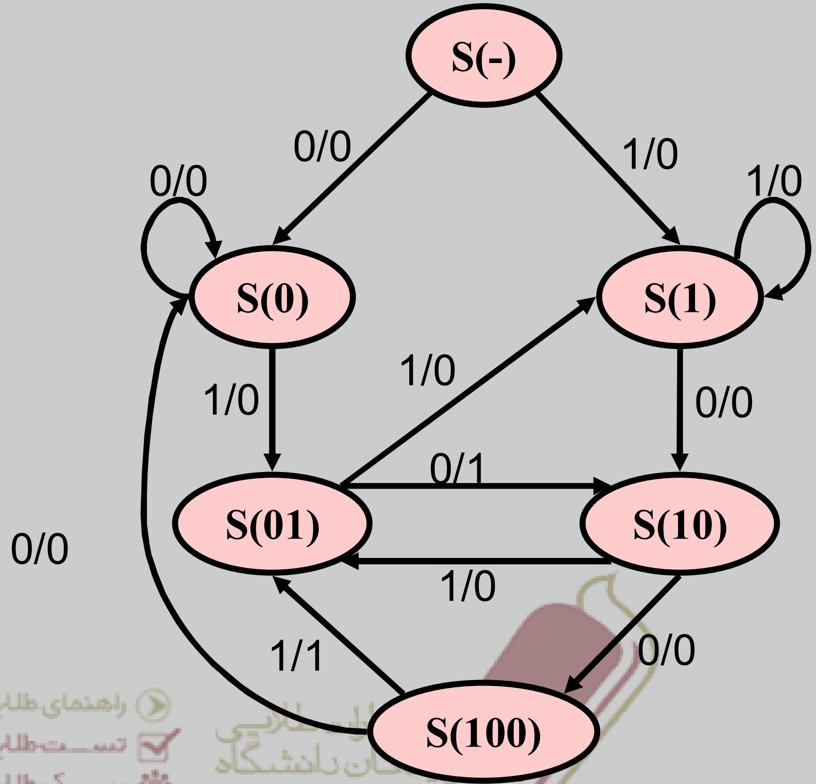


Mealy Sequence Detector

Target Sequences:
010
1001



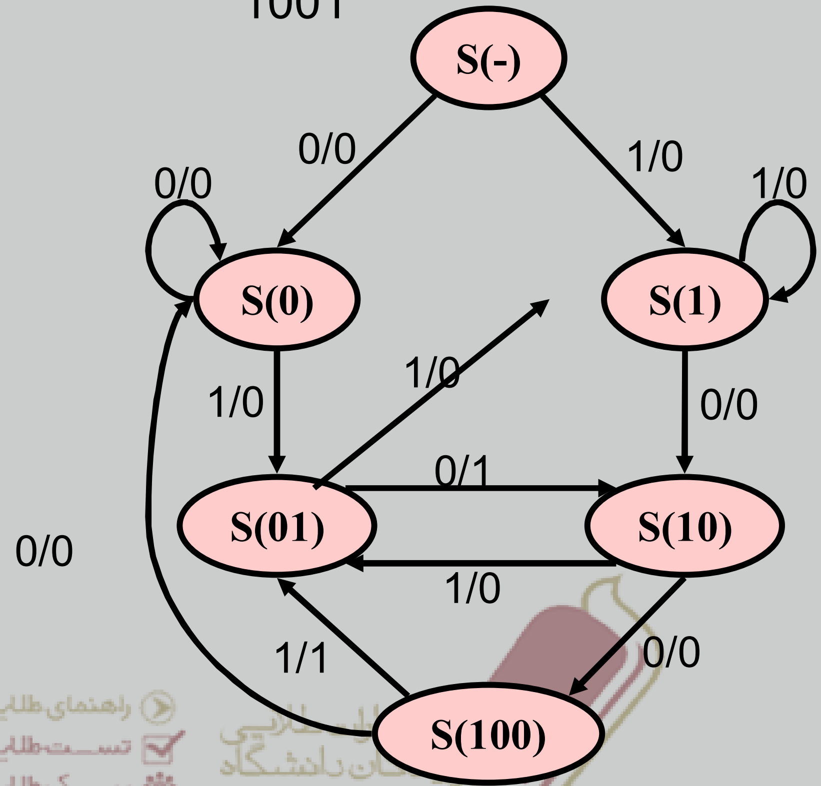
Mealy Sequence Detector



Target Sequences:
010
1001

Mealy Sequence Detector

Target Sequences:
010
1001



Present State	Next State X=0	Next State X=1	Output X=0	Output 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 X=0	Next State X=1	Output X=0	Output 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 X=0	Next State X=1	Output X=0	Output 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

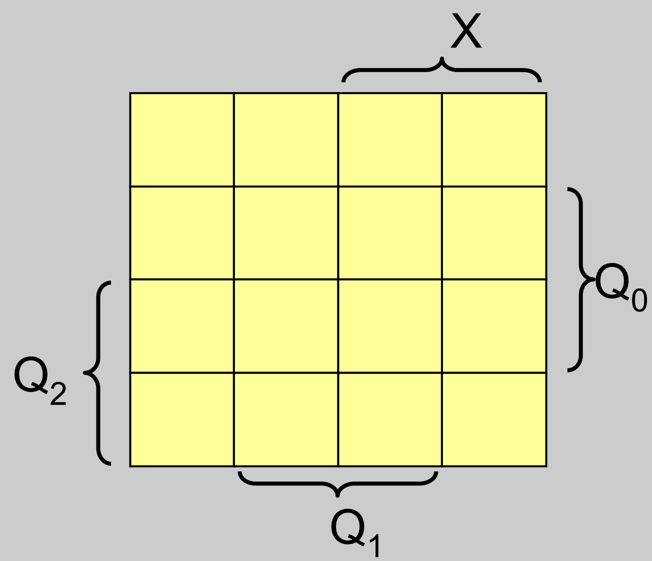
Mealy Sequence Detector

Present State	Next State X=0	Next State X=1	Output X=0	Output 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	Next State		Output	
	X=0	X=1	X=0	X=1
$Q_2Q_1Q_0$	$Q_2^+Q_1^+Q_0^+$	$Q_2^+Q_1^+Q_0^+$		
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

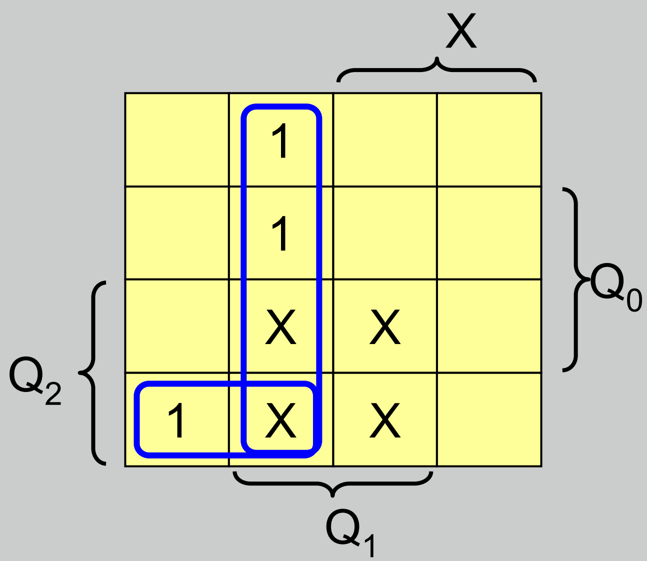
Present State $Q_2Q_1Q_0$	Next State X=0 $Q_2^+Q_1^+Q_0^+$	Next State X=1 $Q_2^+Q_1^+Q_0^+$	Output X=0	Output X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1

		Q_1	Q_1, X	X
		1		
Q_0		1		
Q_2, Q_0		X	X	
Q_2	1	X	X	

$D_2 =$

Mealy Sequence Detector

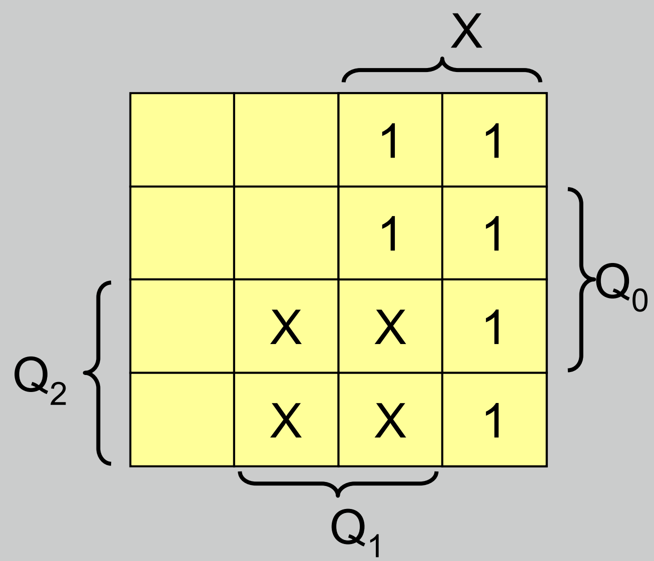
Present State $Q_2Q_1Q_0$	Next State X=0 $Q_2^+Q_1^+Q_0^+$	Next State X=1 $Q_2^+Q_1^+Q_0^+$	Output X=0	Output X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1



$D_2 = Q_1X' + Q_2Q_0'X'$

Mealy Sequence Detector

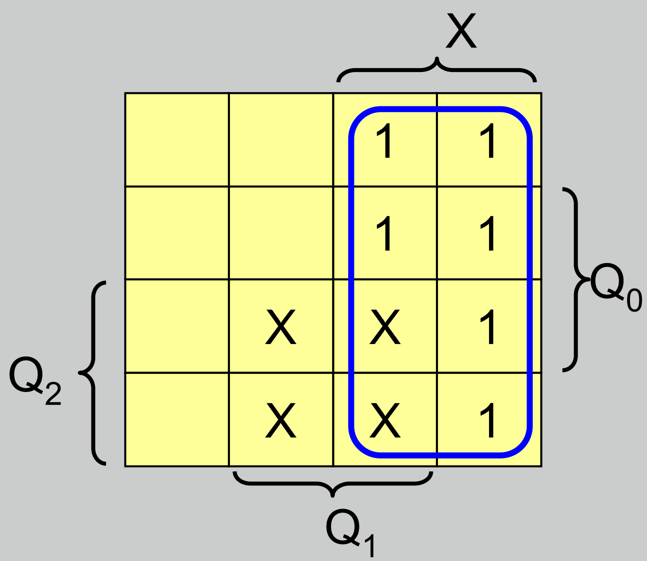
Present State	Next State		Output	
	X=0	X=1	X=0	X=1
$Q_2Q_1Q_0$	$Q_2^+Q_1^+Q_0^+$	$Q_2^+Q_1^+Q_0^+$		
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1



$D_1 =$

Mealy Sequence Detector

Present State	Next State		Output	
	X=0	X=1	X=0	X=1
$Q_2Q_1Q_0$	$Q_2^+Q_1^+Q_0^+$	$Q_2^+Q_1^+Q_0^+$		
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



$D_1 = X$

Mealy Sequence Detector

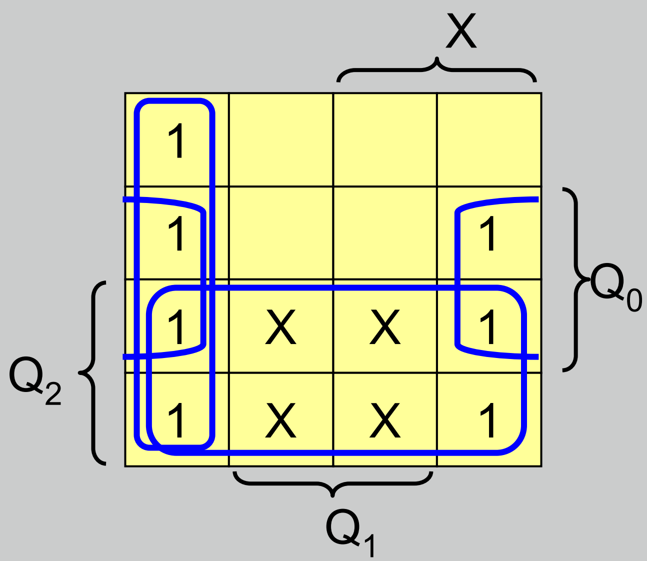
Present State	Next State		Output	
	X=0	X=1	X=0	X=1
$Q_2Q_1Q_0$	$Q_2^+Q_1^+Q_0^+$	$Q_2^+Q_1^+Q_0^+$		
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1

		Q_1	Q_1, X	X
	1			
Q_0	1			1
Q_2, Q_0	1	X	X	1
Q_2	1	X	X	1

$D_0 =$

Mealy Sequence Detector

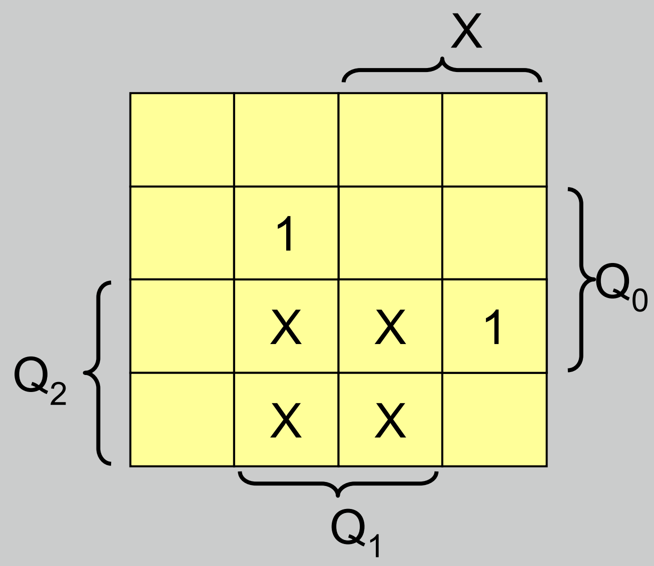
Present State $Q_2Q_1Q_0$	Next State		Output	
	X=0 $Q_2^+Q_1^+Q_0^+$	X=1 $Q_2^+Q_1^+Q_0^+$	X=0	X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1



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

Mealy Sequence Detector

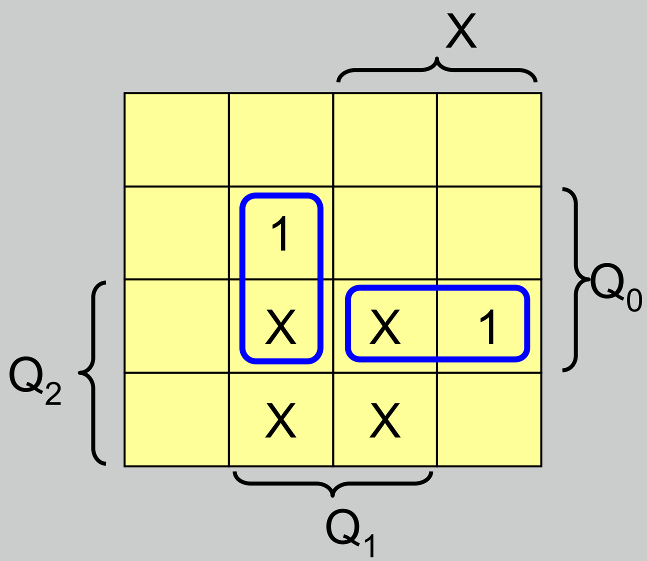
Present State $Q_2Q_1Q_0$	Next State X=0 $Q_2^+Q_1^+Q_0^+$	Next State X=1 $Q_2^+Q_1^+Q_0^+$	Output X=0	Output X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1



Z =

Mealy Sequence Detector

Present State $Q_2Q_1Q_0$	Next State X=0 $Q_2^+Q_1^+Q_0^+$	Next State X=1 $Q_2^+Q_1^+Q_0^+$	Output X=0	Output 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 X=0 $Q_2^+Q_1^+Q_0^+$	Next State X=1 $Q_2^+Q_1^+Q_0^+$	Output X=0	Output 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

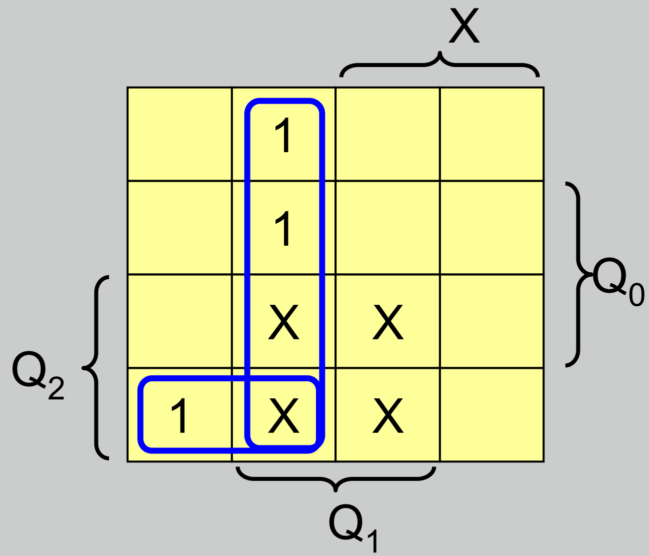
Present State	Next State		Output	
	X=0	X=1	X=0	X=1
$Q_2Q_1Q_0$	$Q_2^+Q_1^+Q_0^+$	$Q_2^+Q_1^+Q_0^+$		
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1
110	1??	0??	?	?
111	1??	0??	?	?

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

$D_1 = X$

$D_2 = Q_1X' + Q_2Q_0'X'$

$X = Q_1Q_0X' + Q_2Q_0X$



Mealy Sequence Detector Design Verification

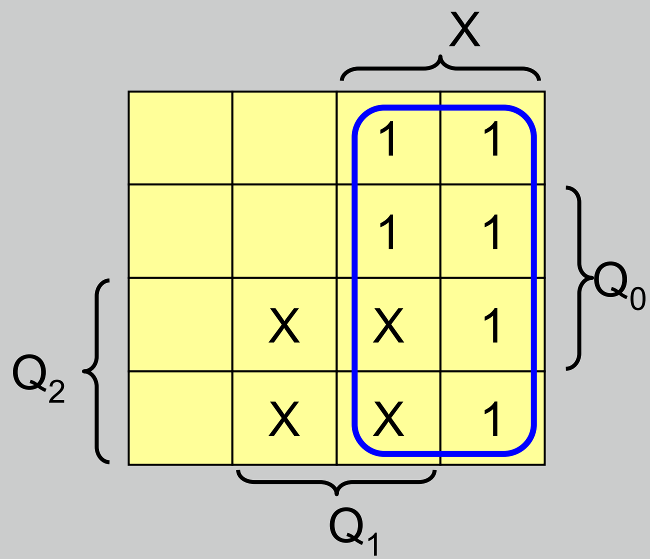
Present State	Next State		Output	
	X=0	X=1	X=0	X=1
$Q_2Q_1Q_0$	$Q_2^+Q_1^+Q_0^+$	$Q_2^+Q_1^+Q_0^+$		
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1
110	10?	01?	?	?
111	10?	01?	?	?

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

$D_1 = X$

$D_2 = Q_1X' + Q_2Q_0'X'$

$X = Q_1Q_0X' + Q_2Q_0X$



Mealy Sequence Detector Design Verification

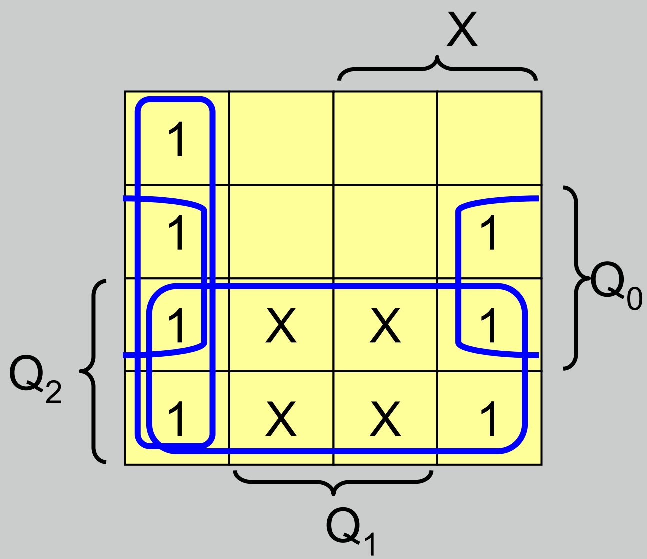
Present State $Q_2Q_1Q_0$	Next State X=0 $Q_2^+Q_1^+Q_0^+$	Next State X=1 $Q_2^+Q_1^+Q_0^+$	Output X=0	Output X=1
000	001	010	0	0
001	001	011	0	0
010	100	010	0	0
011	100	010	1	0
100	101	011	0	0
101	001	011	0	1
110	101	011	?	?
111	101	011	?	?

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

$D_1 = X$

$D_2 = Q_1X' + Q_2Q_0'X'$

$X = Q_1Q_0X' + Q_2Q_0X$



Mealy Sequence Detector Design Verification

Present State $Q_2Q_1Q_0$	Next State X=0 $Q_2^+Q_1^+Q_0^+$	Next State X=1 $Q_2^+Q_1^+Q_0^+$	Output X=0	Output 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

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

$D_1 = X$

$D_2 = Q_1X' + Q_2Q_0'X'$

$X = Q_1Q_0X' + Q_2Q_0X$