Razi University

Logic Circuits Design

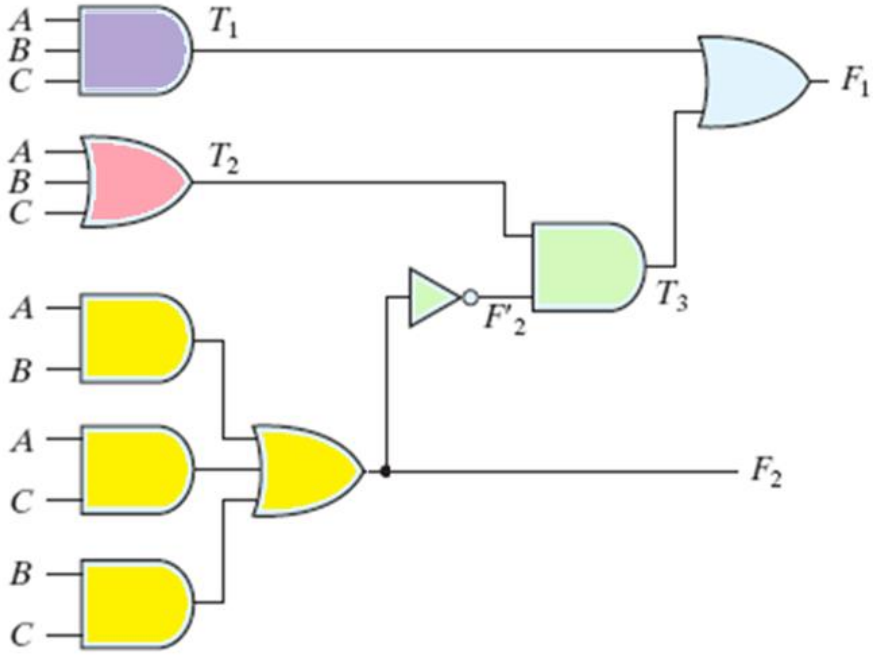
Computer Engineering Department of
Razi University

Dr. Abdolhossein Fathi

طراحی مدارهای ترکیبی

تحلیل مدارات ترکیبی

- در تحلیل مدارات از روی شماتیک مدار تابع مربوط به هر خروجی استخراج می گردد. برای استخراج از جدول صحت استفاده می گردد.



A	B	C	T_1	T_2	F_2	F_2'	T_3	F_1
0	0	0	0	0	0	1	0	0
0	0	1	0	1	0	1	1	1
0	1	0	0	1	0	1	1	1
0	1	1	0	1	1	0	0	0
1	0	0	0	1	0	1	1	1
1	0	1	0	1	1	0	0	0
1	1	0	0	1	1	0	0	0
1	1	1	1	1	1	0	0	1

$T1 = ABC$

$T2 = A + B + C$

$T3 = T2F2' = (A + B + C)(AB + AC + BC)'$

$F2 = AB + AC + BC$ $F1 = T1 + T3 = ABC + (A + B + C)(AB + AC + BC)'$

مراحل طراحی مدارهای ترکیبی

- تعیین ورودی ها وخروجی های مدار (n ورودی و m خروجی)
- تشکیل جدول درستی بر اساس ورودیها و خروجیها
  (2^n ترکیب ورودی و m ستون خروجی)
- بدست آوردن یک تابع برای هر خروجی (با توجه به توصیف مدار)
- ساده سازی توابع بدست آمده
- رسم نمودار شماتیکی مدار

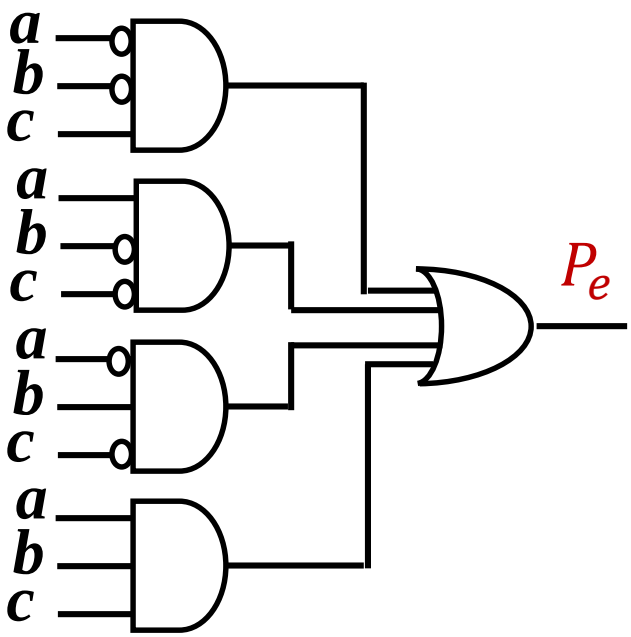
مثال :

- مداری را طراحی کنید که هرگاه تعداد یکهای بین سه ورودی آن فرد باشد (Even Parity) مقدار خروجی یک گردد.

a	b	c	P_e
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

$$P_e(a,b,c) = \sum m(1,2,4,7)$$

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



$$P_e = a'b'c + ab'c' + abc + a'bc'$$

طرحی ماجولار مدار

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

برای طراحی این مدارات به دو شیوه می توان عمل کرد:

- بدون رسم جدول درستی به خروجی مدار برسیم.
  - رهیافت ذهنی و وابسته به خلاقیت طراح
- طراحی ماژولار مدار: مدار را به واحد های کوچکتر (به نام ماژول) شکسته که با ترکیب و اتصال تعدادی از آنها مدار نهایی بدست خواهد آمد.
  - طراحی پیمانه ای
  - از نظر زمانی بهینه نیست

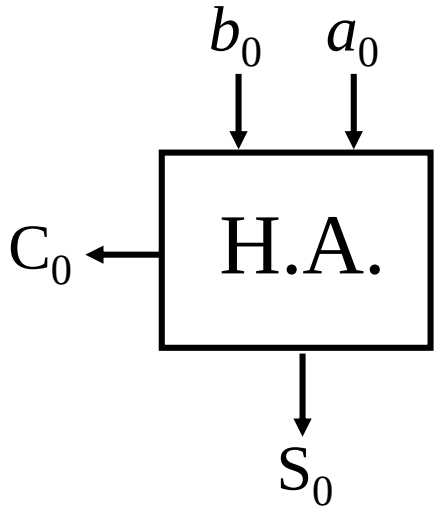
طراحی جمع کننده

- در طراحی مدار جمع کننده n بیتی می توان فرآینده جمع در رقم های مختلف را بصورت ماژول وار طراحی نمود:

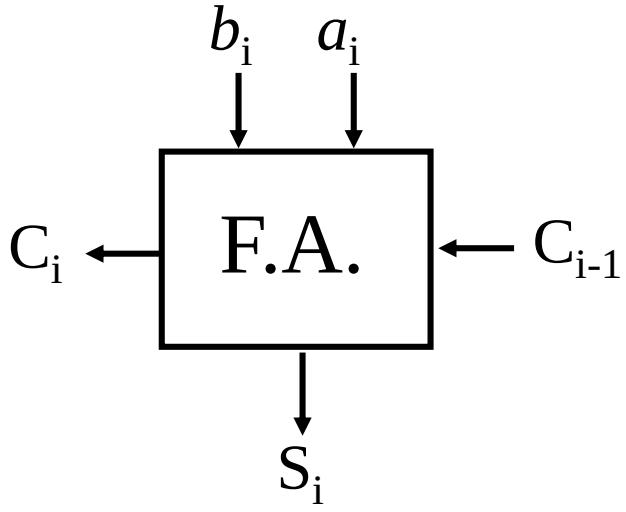
رقم نقلی تولید شده	$(C=c_5c_4c_3c_2c_1c_0)$	0111010
	$(A=a_5a_4a_3a_2a_1a_0)$	0101010
	$(B=b_5b_4b_3b_2b_1b_0)$	+0011010
حاصل جمع	$(S=s_5s_4s_3s_2s_1s_0)$	1000100

- **Half Adder**: در طبقه اول (رقم یکان) فقط باید دو بیت مربوط به a_0 و b_0 را با هم جمع کرد. یک مدار ترکیبی که با جمع دو ورودی، دو خروجی حاصل جمع s_0 و رقم نقلی c_0 را محاسبه می کند.
- **Full Adder**: در طبقه دوم به بعد در هر طبقه علاوه بردو بیت مربوط به a_i و b_i باید رقم نقلی طبقه قبل (c_{i-1}) هم با آنها جمع گردد. یک مدار ترکیبی که با جمع سه ورودی، دو خروجی حاصل جمع s_i و رقم نقلی c_i را محاسبه می کند.

طراحی Half Adder و Full Adder

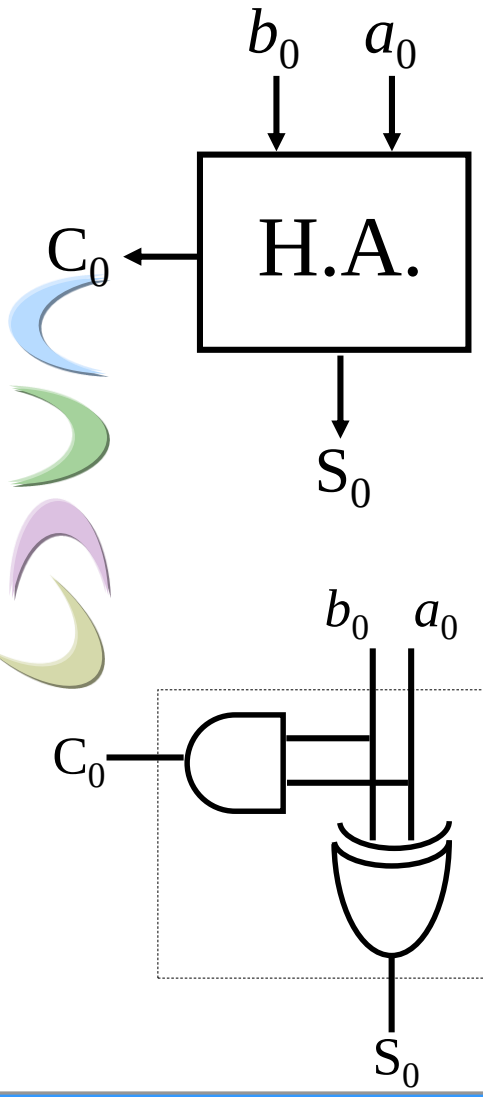


- **Half Adder**:یک مدار ترکیبی با دو ورودی و دو خروجی است که دو بیت کم ارزش داده را با هم جمع کرده و حاصل جمع و رقم نقلی را محاسبه می کند.



- **Full Adder**:یک مدار ترکیبی با سه ورودی و دو خروجی است که دو بیت داده ویک رقم نقلی طبقه قبل را با هم جمع کرده و حاصل جمع و رقم نقلی را محاسبه می کند.

طراحی نیم جمع کننده

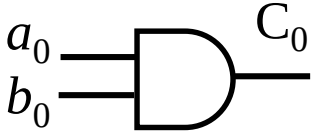


جدول درستی

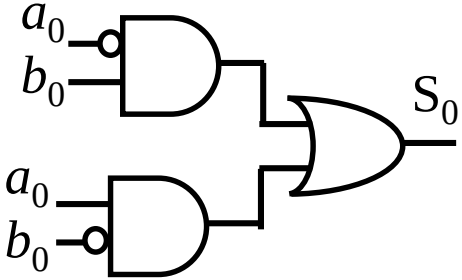
a_0	b_0	C_0	S_0
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

b_0 \ a_0	0	1
0	0	0
1	0	1

$$C_0 = a_0 . b_0$$



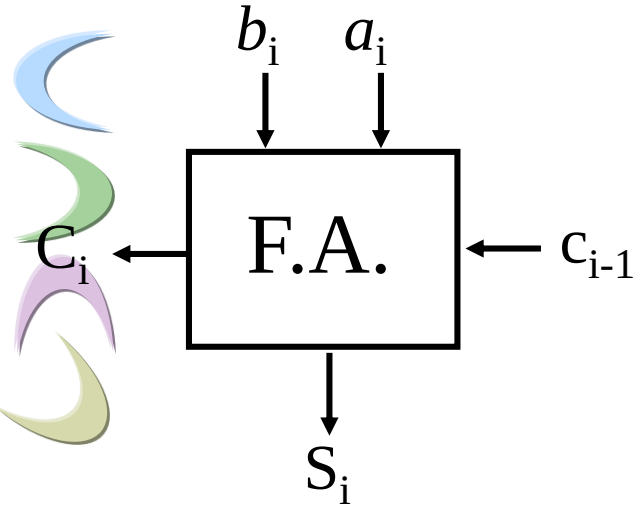
b_0 \ a_0	0	1
0	0	1
1	1	0



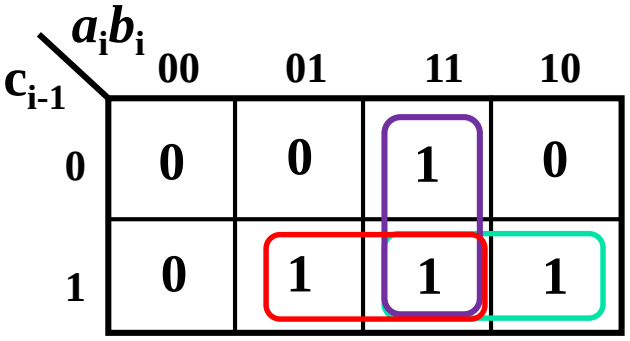
$$S_0 = \overline{a}_0 . b_0 + a_0 . \overline{b}_0 = a_0 \oplus b_0$$

طراحی مدار تمام جمع کننده

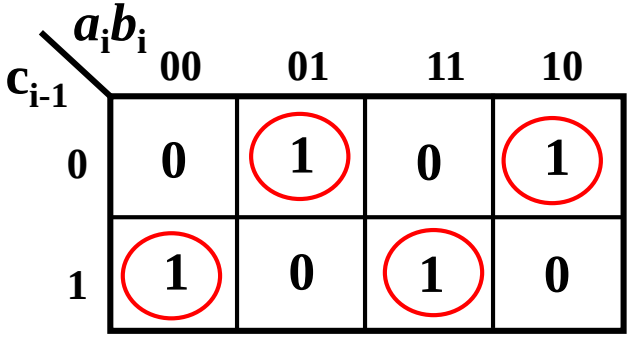
جدول درستی



a_i	b_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



$$C_i = a_ib_i + a_ic_{i-1} + b_ic_{i-1}$$

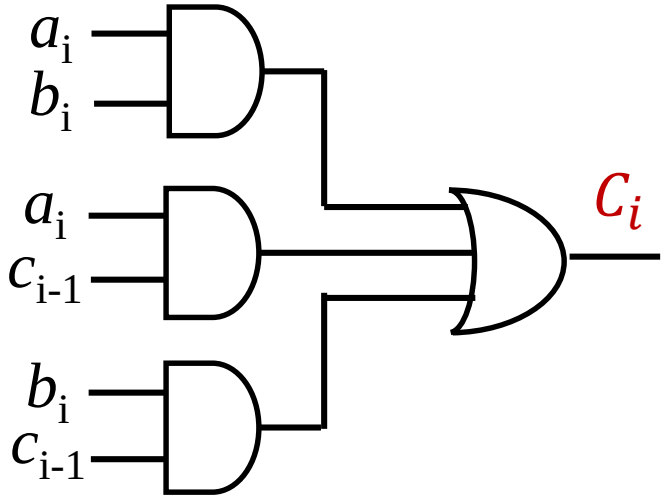


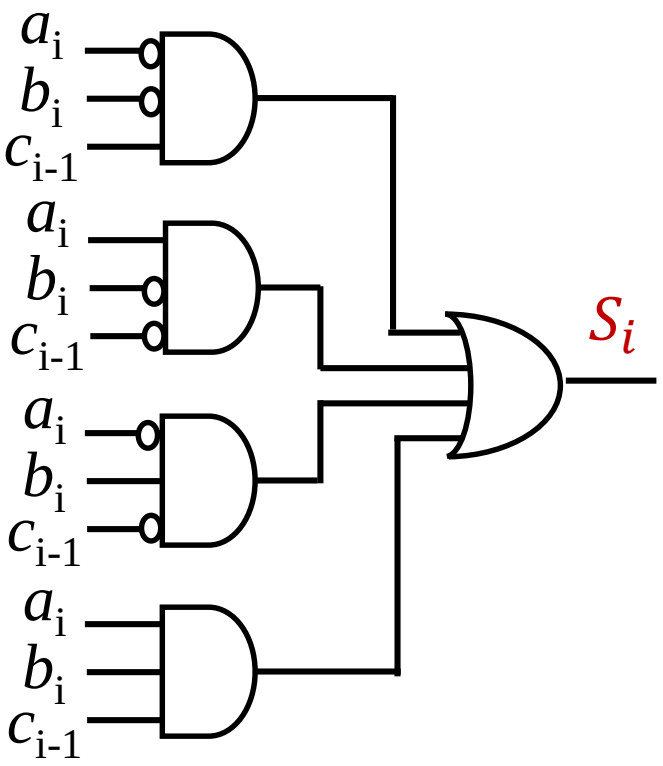
$$S_i = \bar{a}_i\bar{b}_ic_{i-1} + a_i\bar{b}_i\bar{c}_{i-1} + \bar{a}_ib_i\bar{c}_{i-1} + a_ib_ic_{i-1} = a_i \oplus b_i \oplus c_{i-1}$$

طراحی مدار تمام جمع کننده

$$C_i = a_i b_i + a_i c_{i-1} + b_i c_{i-1}$$

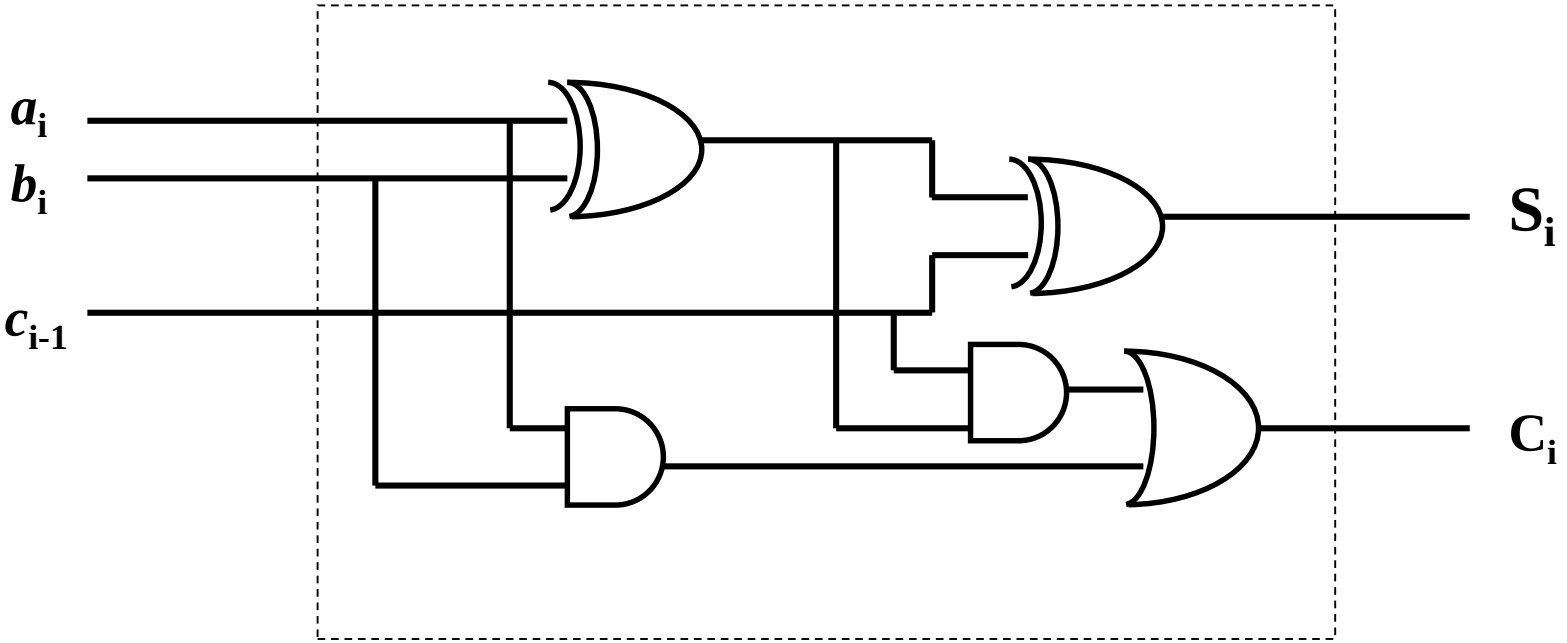
$$S_i = \bar{a}_i \bar{b}_i c_{i-1} + a_i \bar{b}_i \bar{c}_{i-1} + \bar{a}_i b_i \bar{c}_{i-1} + a_i b_i c_{i-1}$$





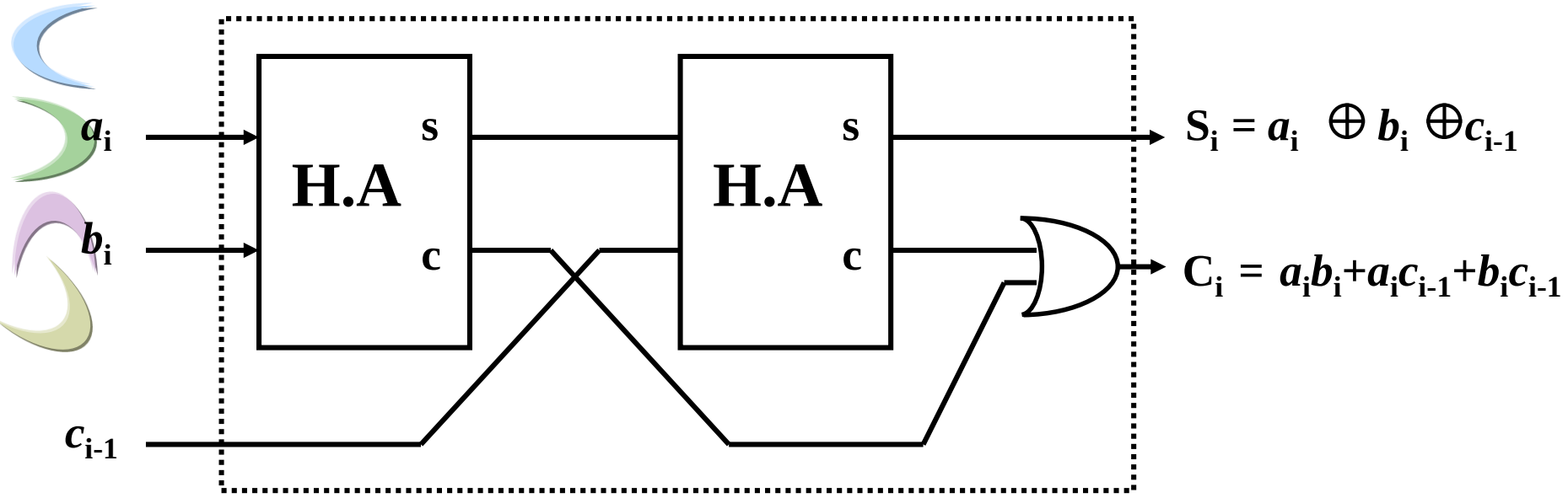
طراحی مدار تمام جمع کننده

$$\begin{cases} S_i = \bar{a}_i\bar{b}_i c_{i-1} + a_i\bar{b}_i\bar{c}_{i-1} + \bar{a}_i b_i\bar{c}_{i-1} + a_i b_i c_{i-1} = a_i \oplus b_i \oplus c_{i-1} \\ C_i = a_i b_i + a_i c_{i-1} + b_i c_{i-1} \end{cases} \Rightarrow C_i = c_{i-1}(a_i \oplus b_i) + a_i b_i$$

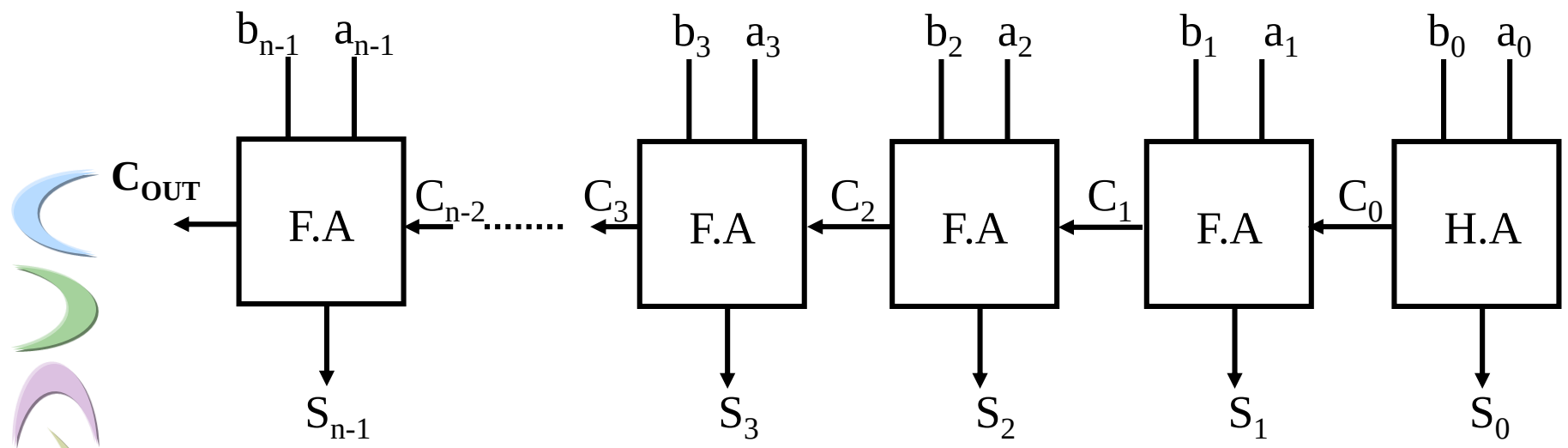


طراحی مدار تمام جمع کننده

یک **Full adder** را میتوان توسط ۲ عدد **Hull adder** طراحی کرد.



طراحی جمع کننده n بیتی
Ripple Carry Adder (RCA)



مشکل این جمع کننده سرعت پایین آن (بخاطر زمان انتقال زنجیره رقم های نقلی) می باشد.

برای افزایش سرعت محاسبات می توان با استفاده از مدارات اضافی رقم های نقلی را در یک مرحله باری همه طبقات محاسبه کرد تا بتوان تمام جمع ها را با هم بصورت موازی انجام داد (**Carry Look-Ahead**).

تمرین: مدار یک جمع کننده پنج بیتی را بصورت **Carry Look-Ahead** طراحی کنید.

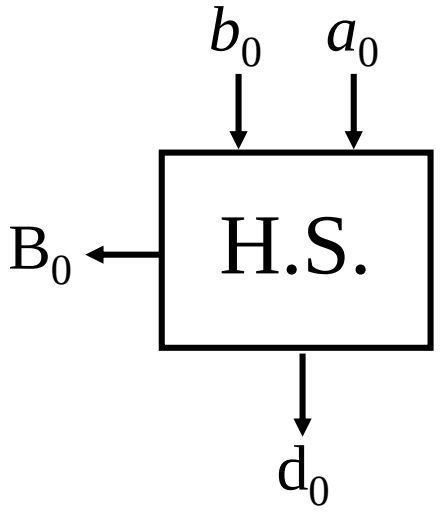
طراحی تفریق کننده

- شبیه طراحی مدار جمع کننده n بیتی می توان فرآینده تفریق دو عدد n بیتی در رقم های مختلف را بصورت ماژول وار طراحی نمود:

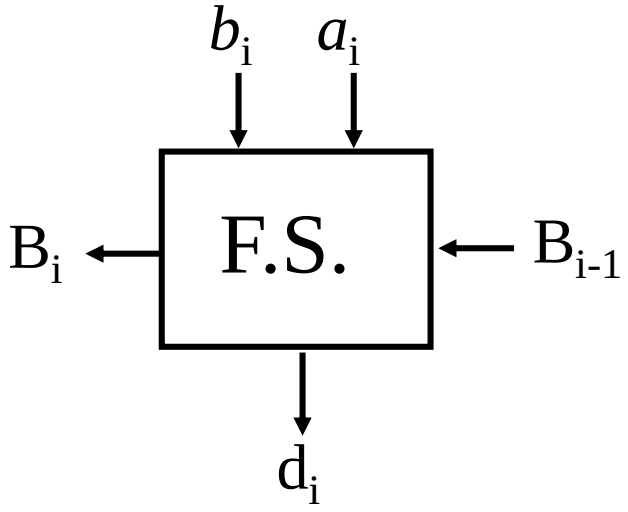
رقم قرضی تولید شده	0010000 (B=$B_5B_4B_3B_2B_1B_0$)
	0101010 (A=$a_5a_4a_3a_2a_1a_0$)
	- 0011010 (B=$b_5b_4b_3b_2b_1b_0$)
حاصل تفریق	0010000 (D=$d_5d_4d_3d_2d_1d_0$)

- **Half Subtractor**: در طبقه اول (رقم یکان) فقط باید دو بیت مربوط به a_0 و b_0 را از هم کم کرد. و طبقه قبلی برای قرض دادن وجود ندارد. یک مدار ترکیبی که با تفریق دو ورودی، دو خروجی حاصل تفریق d_0 و رقم قرضی مورد نیاز B_0 برای محاسبه حاصل تفریق را تولید می کند.
- **Full Subtractor**: در طبقه دوم به بعد در هر طبقه علاوه بردو تفریق دو بیت مربوط به a_i و b_i باید رقم قرض داده شده به طبقه قبل (B_{i-1}) هم از حاصل آنها کم گردد. یک مدار ترکیبی که با تفریق دو ورودی B_{i-1} و b_i از a_i، دو خروجی حاصل تفریق d_i و رقم قرضی مورد نیاز B_i را محاسبه می کند.

طراحی Half Subtractor و Full Subtractor

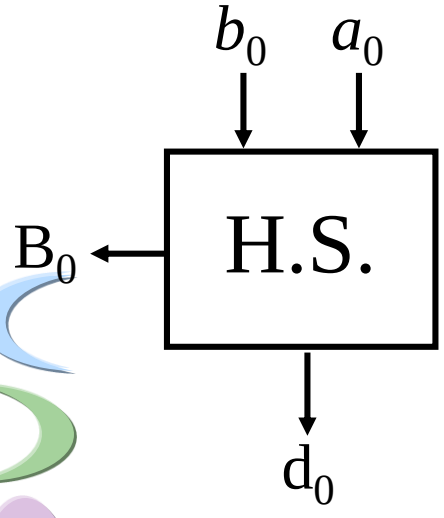


- **Half Subtractor**: یک مدار ترکیبی که دو ورودی a_0 و b_0 را از هم کم کرده و دو خروجی حاصل تفریق d_0 و رقم قرضی مورد نیاز B_0 برای محاسبه حاصل تفریق را تولید می کند.



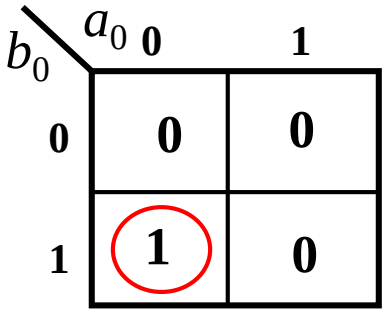
- **Full Subtractor**:یک مدار ترکیبی که دو ورودی B_{i-1} و b_i را از ورودی a_i کم کرده و دو خروجی حاصل تفریق d_i و رقم قرضی مورد نیاز B_i را محاسبه می کند.

طراحی نیم تفریق کننده

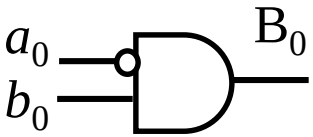


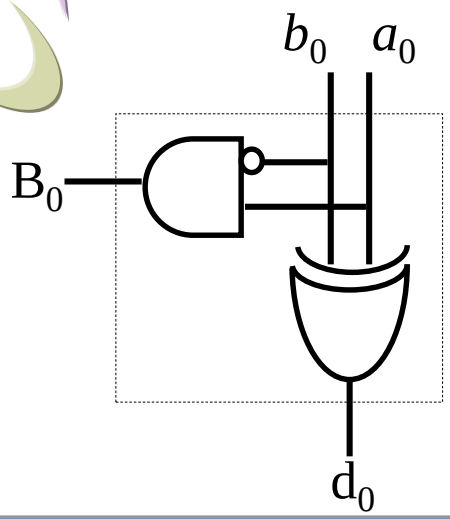
جدول درستی

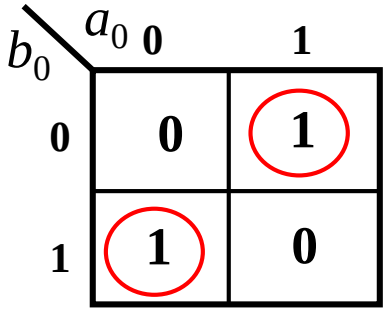
a_0	b_0	B_0	$d_0(a_0\text{-}b_0)$
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

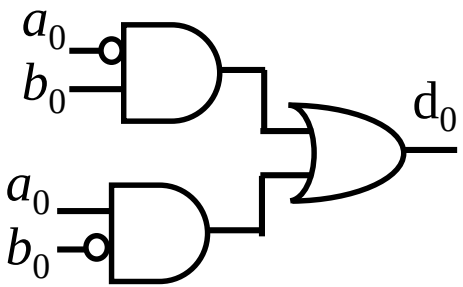


$$B_0 = \overline{a}_0 . b_0$$



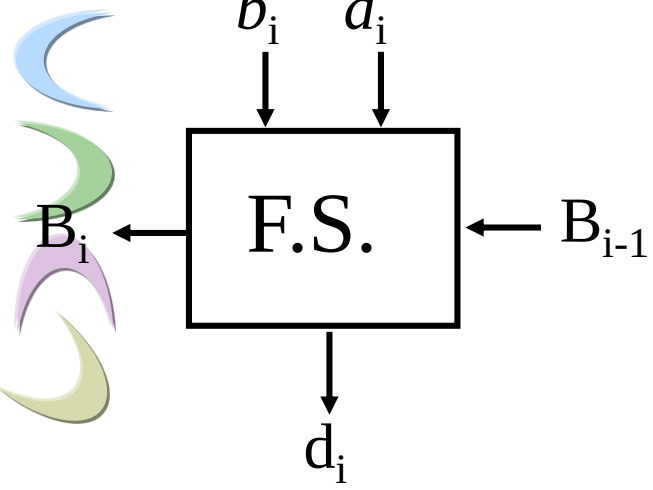






$$d_0 = \overline{a}_0 . b_0 + a_0 . \overline{b}_0 = a_0 \oplus b_0$$

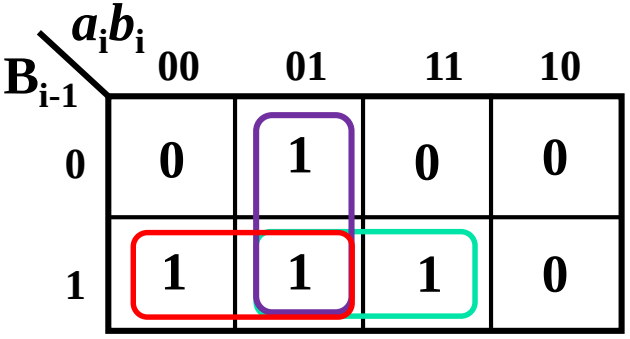
طراحی مدار تمام جمع کننده



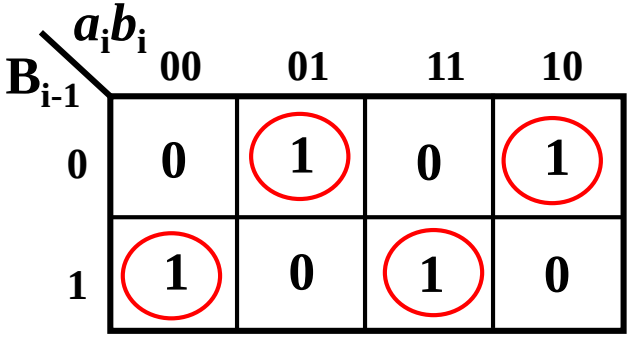
جدول درستی

$d_i = a_i - b_i - B_{i-1}$

a_i	b_i	B_{i-1}	B_i	d_i
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1



$$B_i = \bar{a}_i b_i + \bar{a}_i B_{i-1} + b_i B_{i-1}$$

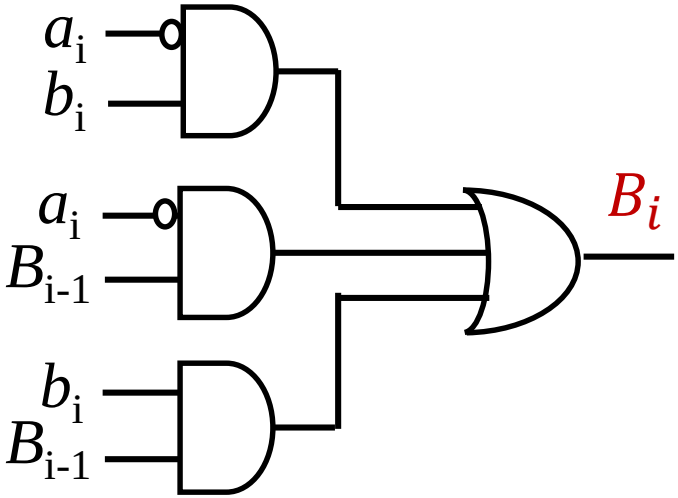


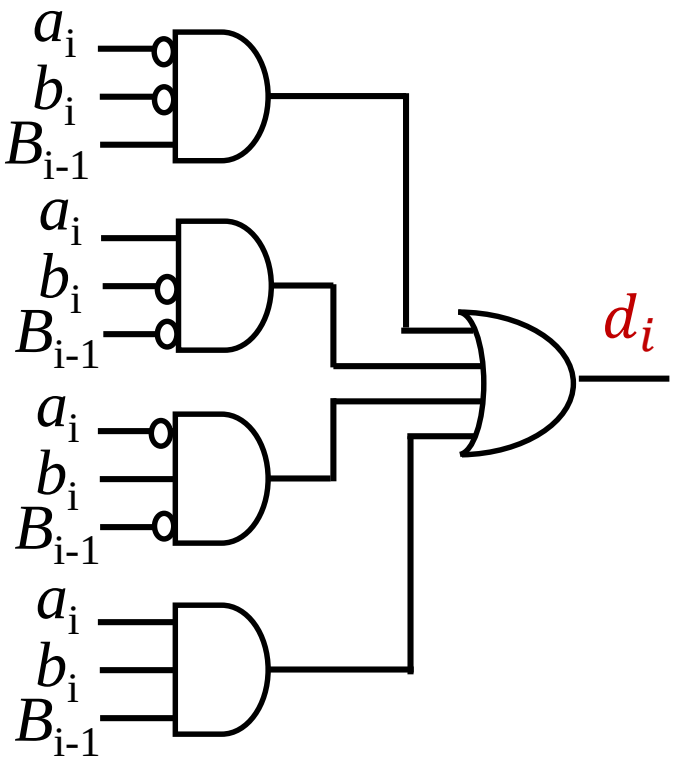
$$d_i = \bar{a}_i \bar{b}_i B_{i-1} + a_i \bar{b}_i \bar{B}_{i-1} + \bar{a}_i b_i \bar{B}_{i-1} + a_i b_i B_{i-1} = a_i \oplus b_i \oplus B_{i-1}$$

طراحی مدار تمام تفریق کننده

$$\mathbf{d_i} = \bar{a}_i\bar{b}_iB_{i-1} + a_i\bar{b}_i\bar{B}_{i-1} + \bar{a}_ib_i\bar{B}_{i-1} + a_ib_iB_{i-1}$$

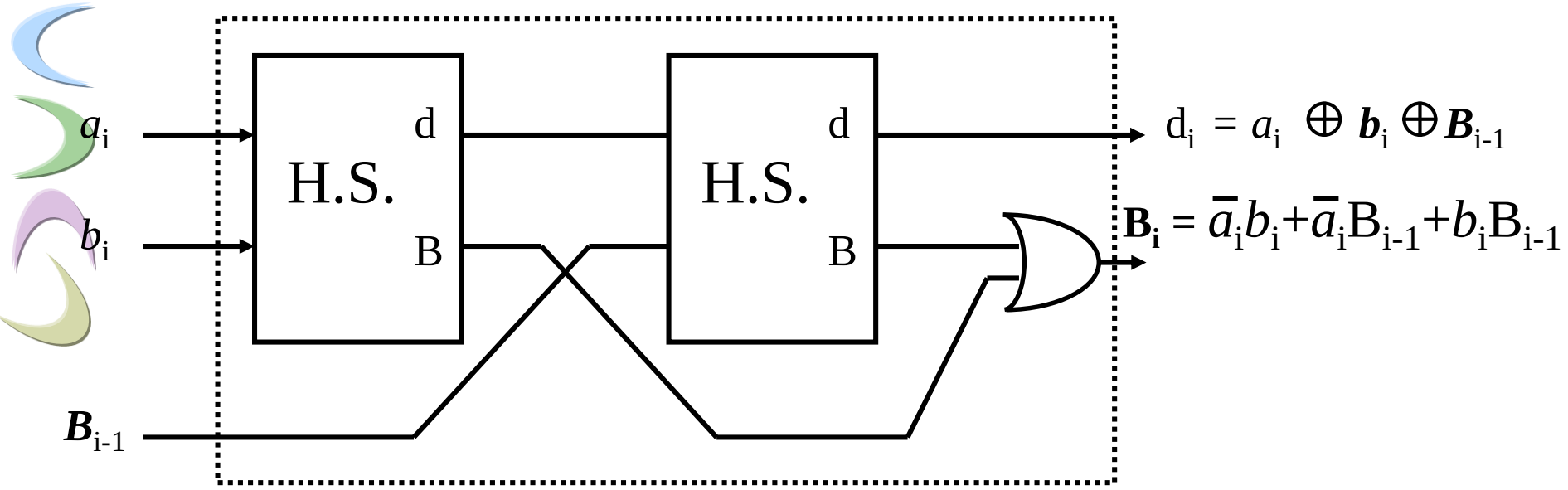
$$\mathbf{B_i} = \bar{a}_ib_i + \bar{a}_iB_{i-1} + b_iB_{i-1}$$



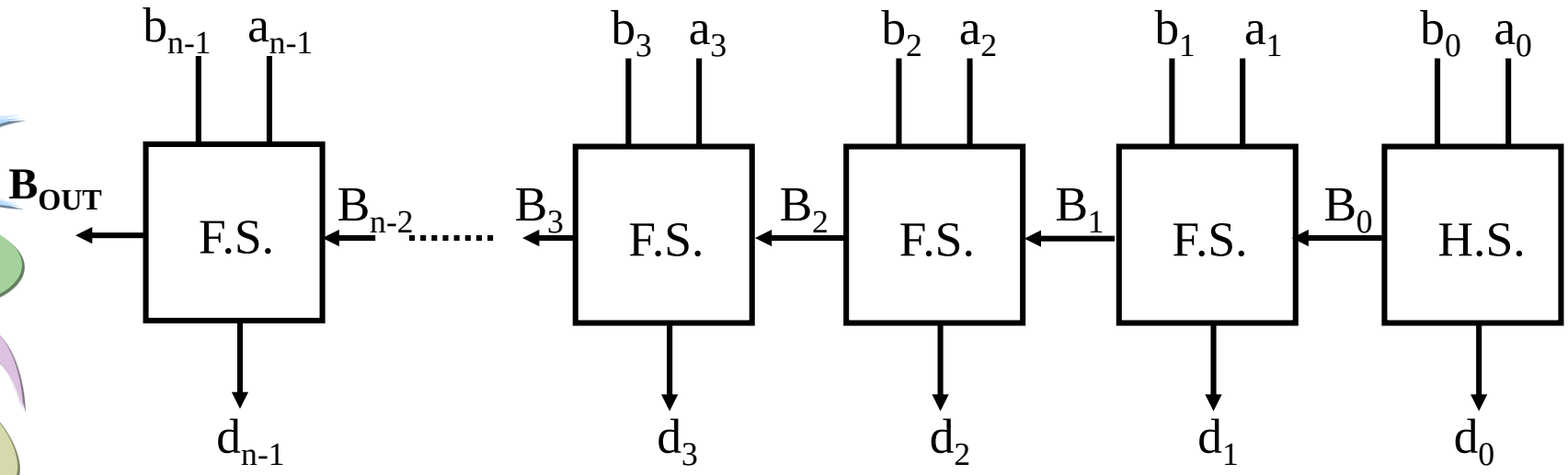


طراحی مدار تمام تفریق کننده

یک **Full Subtractor** را می توان توسط ۲ عدد **Hull Subtractor** طراحی کرد.

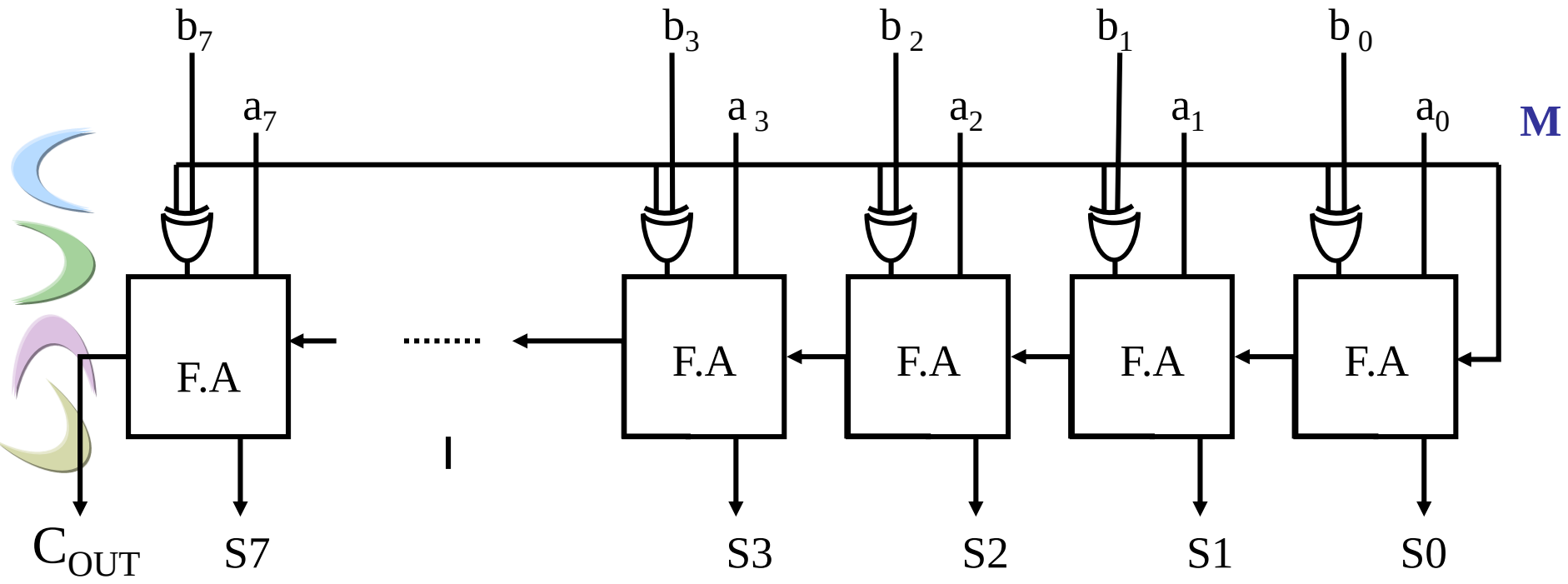


طراحی تفریق کننده n بیتی



مشکل این جمع کننده سرعت پایین آن (بخاطر زمان انتقال زنجیره رقم های قرضی) می باشد.

طراحی جمع کننده/تفریق کننده n بیتی

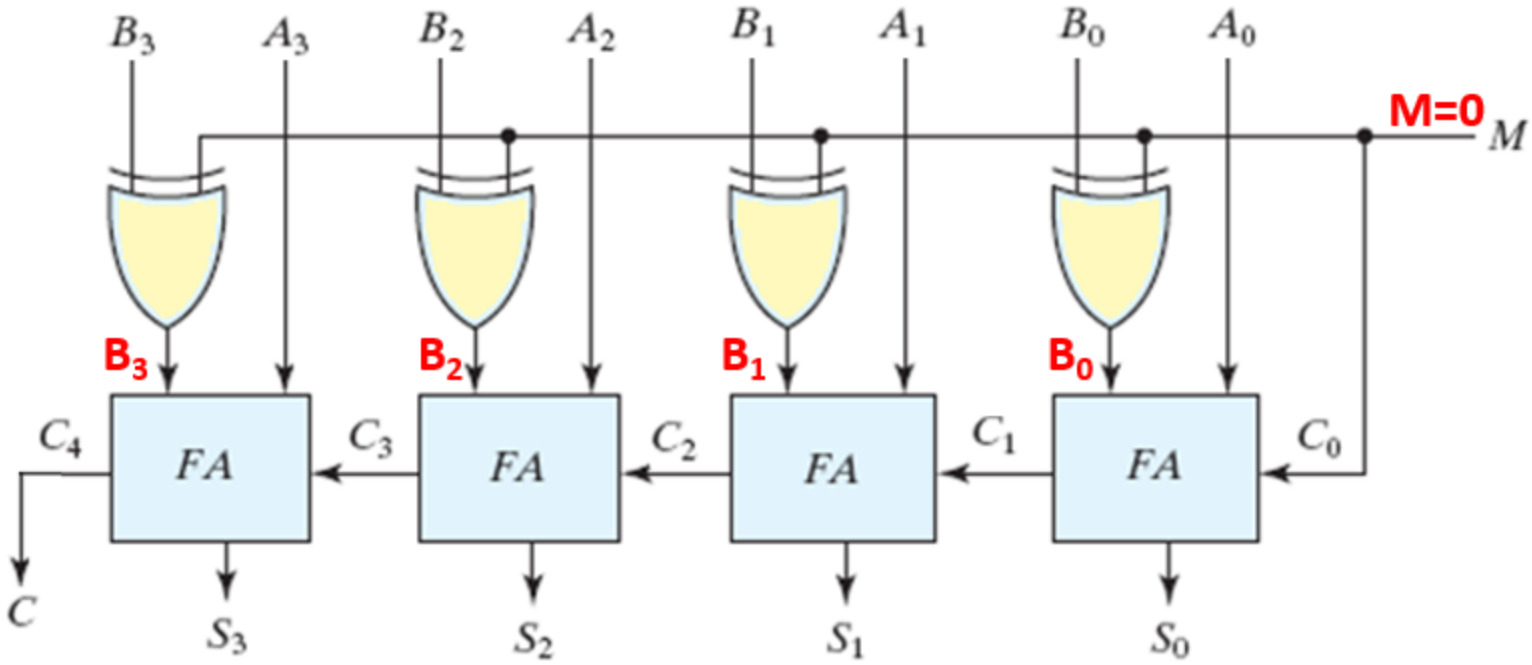


If M =0 $\rightarrow S = A + B = A + (B \oplus 0) + 0 = A + (B \oplus M) + M$

If M =1 $\rightarrow S = A - B = A + (\overline{B} + 1) = A + (B \oplus M) + M$

طراحی جمع کننده/تفریق کننده n بیتی

M: Mode (Control Signal)

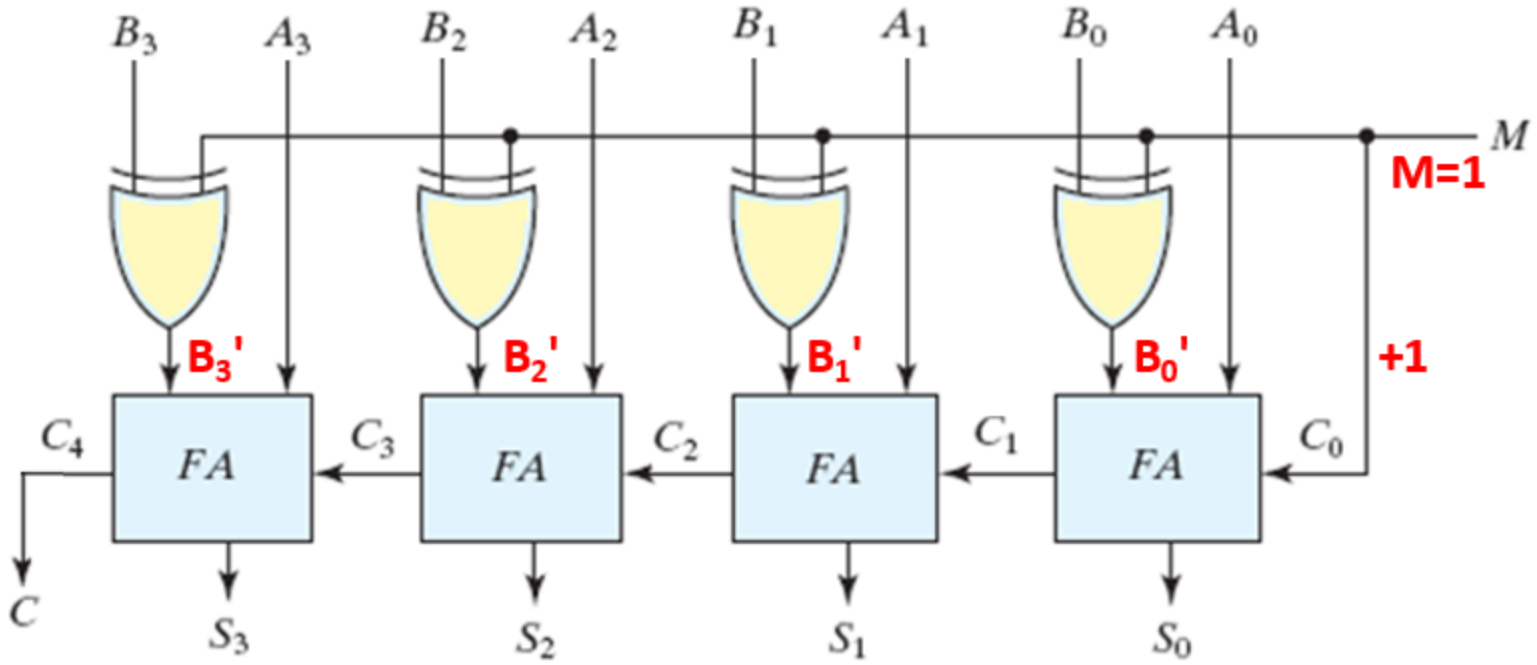


M=0, Adder

S = sum of A & B = A + B + 0 = A + B

طراحی جمع کننده/تفریق کننده n بیتی

M: Mode (Control Signal)



M=1, Subtractor S = subtract of A & B = A + B' + 1 = A + (-B) = A - B

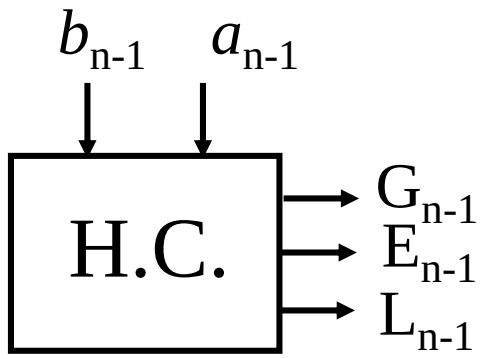
طراحی مقایسه کننده

- شبیه طراحی مدار جمع کنندهو تفریق کننده n بیتی می توان فرآینده مقایسه دو عدد n بیتی را هم بصورت ماژول وار طراحی نمود:

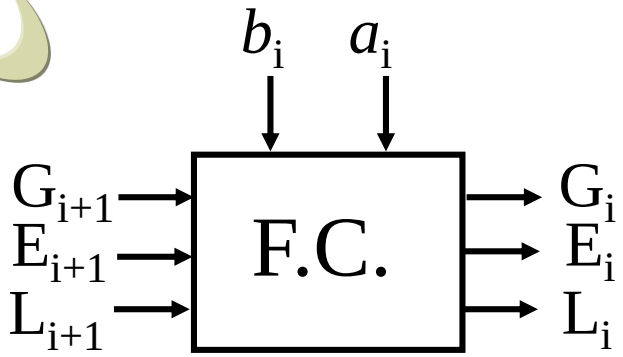
$$\begin{array}{ll} & \\ 0101010 & (A=a_5a_4a_3a_2a_1a_0) \\ 0011010 & (B=b_5b_4b_3b_2b_1b_0) \\ \hline \end{array}$$

- **Half Comprator**: در طبقه اول (با ارزشترین رقم) فقط باید دو بیت با ارزش a_5 و b_5 را با هم مقایسه کرد. لذا یک مدار ترکیبی برای مقایسه دو ورودی، و سه خروجی بزرگتر بودن (A>B) G_{n-1} و برابر بودن (A=B) E_{n-1} و کوچکتر بودن (A<B) L_{n-1} را تولید می کند.
- **Full Comprator**: در طبقات بعد، در هر طبقه ابتدا به نتیجه مقایسه طبقات با ارزش قبلی نگاه کرده و سپس اگر نتایج قبلی بیانگر مساوی بودن آنها باشد ارقام این طبقه یعنی دو بیت a_i و b_i با هم مقایسه خواهند شد. لذا یک مدار ترکیبی برای مقایسه دو عدد شامل پنج ورودی، و سه خروجی (بزرگتر بودن (A>B) G_{i-1} و برابر بودن (A=B) E_{i-1} و کوچکتر بودن (A<B) L_{i-1}) خواهد بود.

طراحی Half Comparator و Full Comparator

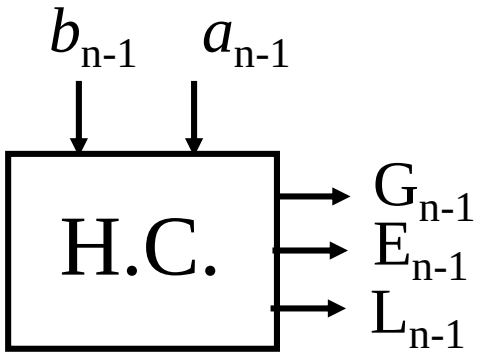


- **Half Comprator**:طبقه اول یک مدار ترکیبی برای مقایسه دو ورودی (با ارزشترین رقم) a_{n-1} و b_{n-1} و تعیین سه خروجی بزرگتر بودن (A>B) G_{n-1} و برابر بودن (A=B) E_{n-1} و کوچکتر بودن (A<B) L_{n-1} می باشد.



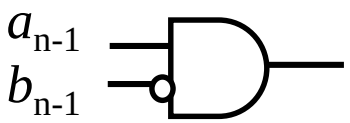
- **Full Comprator**: در طبقات بعد، ابتدا به نتیجه مقایسه طبقات با ارزش قبلی نگاه کرده و سپس اگر نتایج قبلی بیانگر مساوی بودن آنها باشد ارقام این طبقه یعنی دو بیت a_i و b_i با هم مقایسه خواهند شد. لذا مدار مقایسه شامل پنج ورودی، و سه خروجی خواهد بود.

طراحی نیم مقایسه کننده

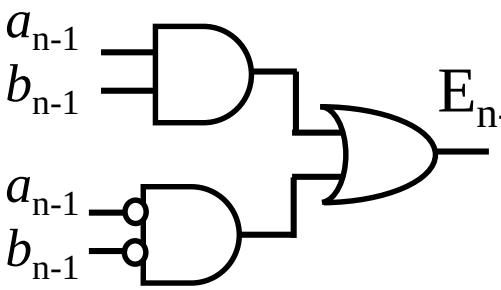


جدول درستی

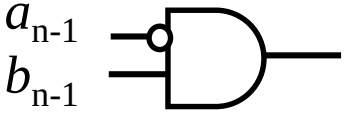
a_{n-1}	b_{n-1}	G_{n-1}	E_{n-1}	L_{n-1}
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0



$$G_{n-1} = a_{n-1}.\overline{b}_{n-1}$$

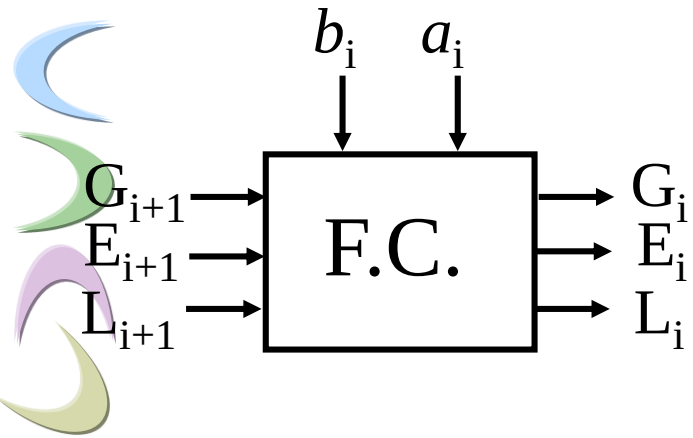


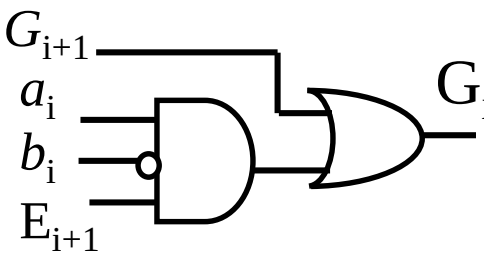
$$E_{n-1} = a_{n-1}.b_{n-1} + \overline{a}_{n-1}.\overline{b}_{n-1}$$



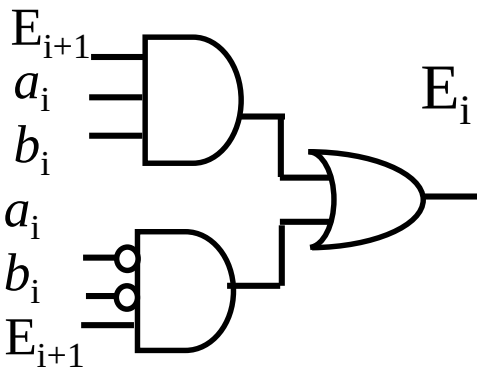
$$L_{n-1} = \overline{a}_{n-1}.b_{n-1}$$

طراحی تمام مقایسه کننده

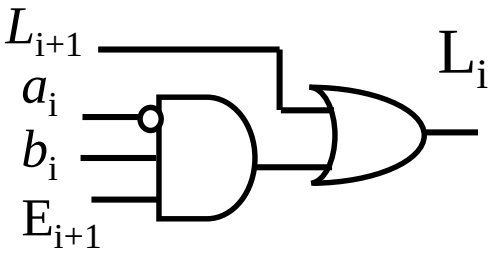




$$G_i = G_{i+1} + E_{i+1} a_i . \overline{b_i}$$

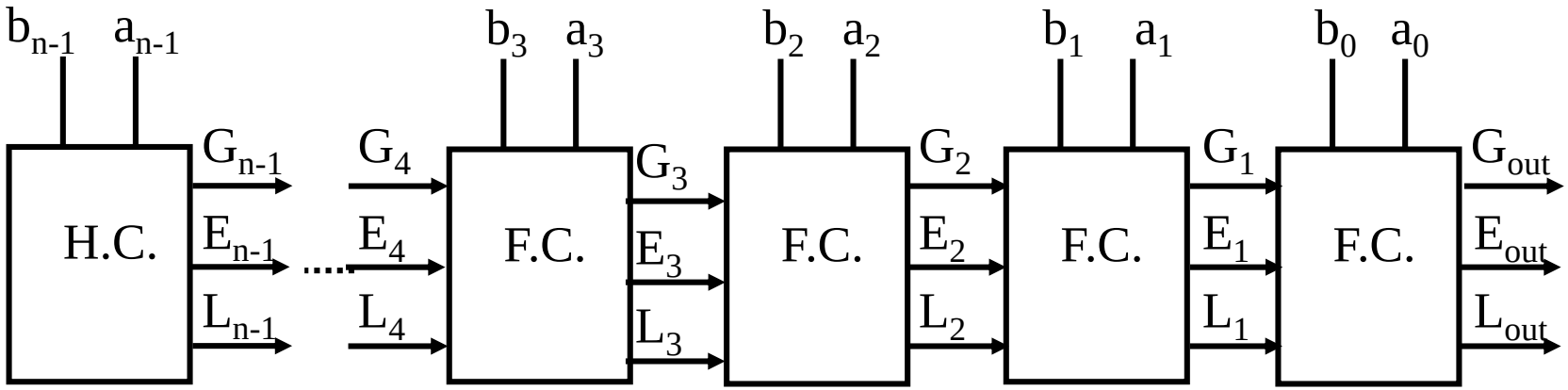


$$E_i = E_{i+1} a_i . b_i + E_{i+1} \overline{a_i} . \overline{b_i}$$



$$L_i = L_{i+1} + E_{i+1} \overline{a_i} . b_i$$

طراحی مقایسه کننده n بیتی

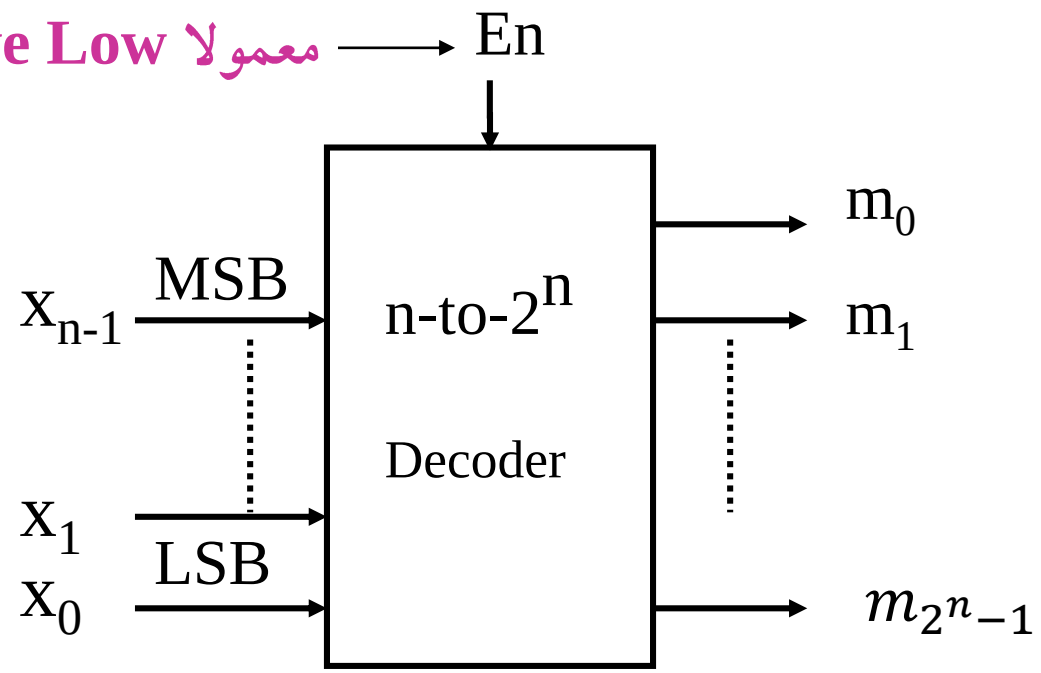


می توان بجای مقایسه کردن از عملیات تفریق استفاده کرد و اگر نتیجه تفریق برابر صفر باشد دو عدد مساوی و اگر مثبت باشد بزرگتر و در غیر اینصورت کوچکتر خواهد بود.

طراحی دیکدر n به 2^n

- دیکدر n به 2^n یک مدار منطقی ترکیبی است که ترکیب n خط ورودی را به یکی از 2^n سیگنال خروجی نگاشت می کند (خط معادل عدد ورودی را فعال می کند).
- دیکدر در حقیقت عنصری است که مینترم ها را در حروجی می سازد.

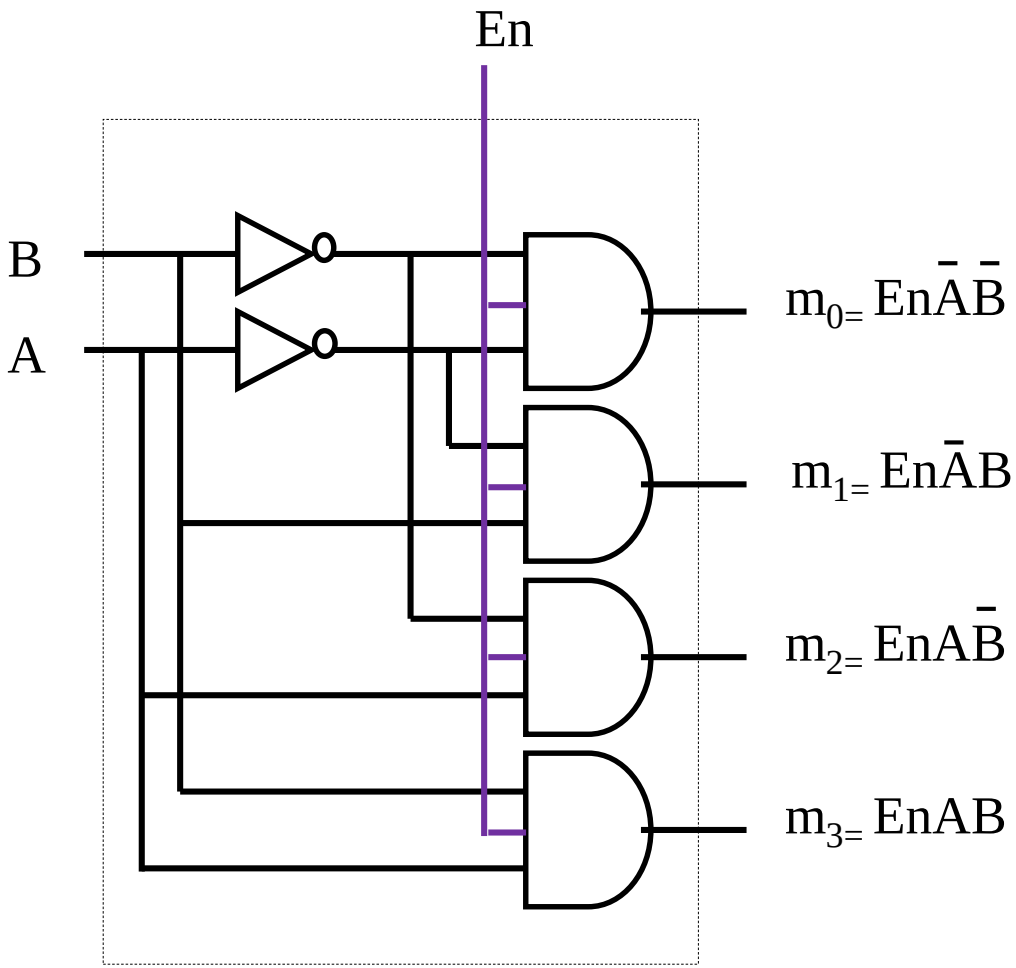
معمولا Active Low هستند.



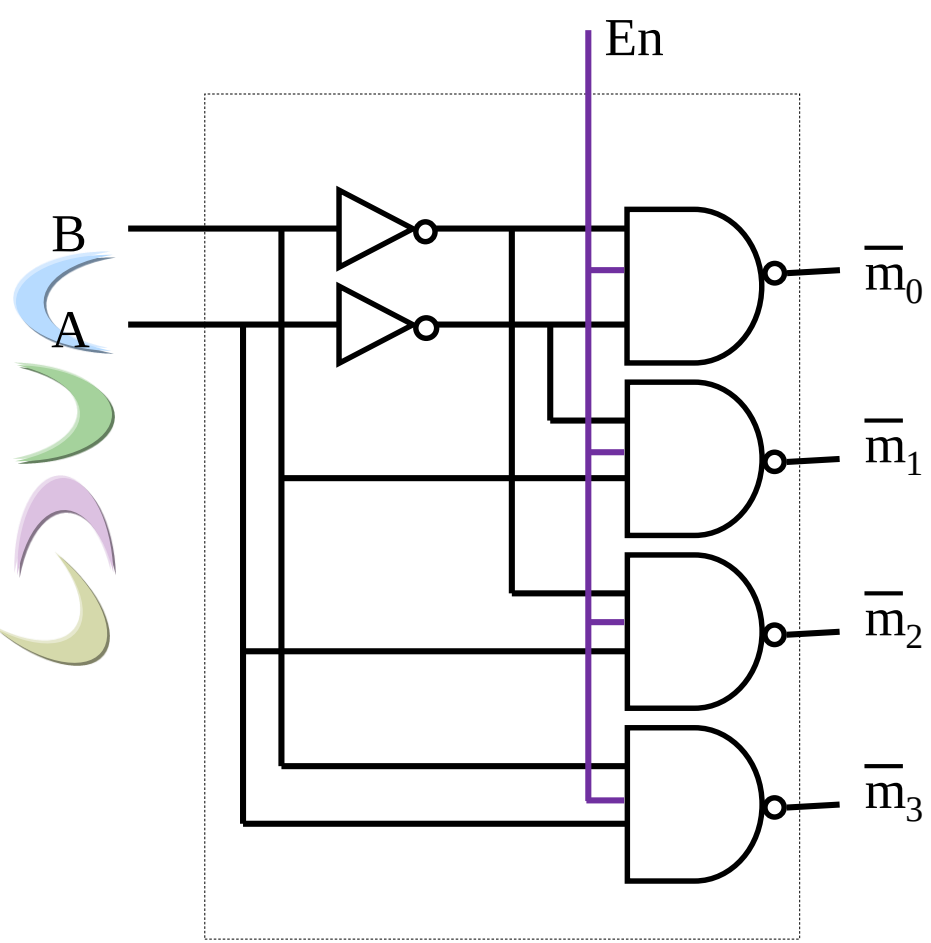
دیاگرام منطقی (موازی و خروجی های فعال بالا)

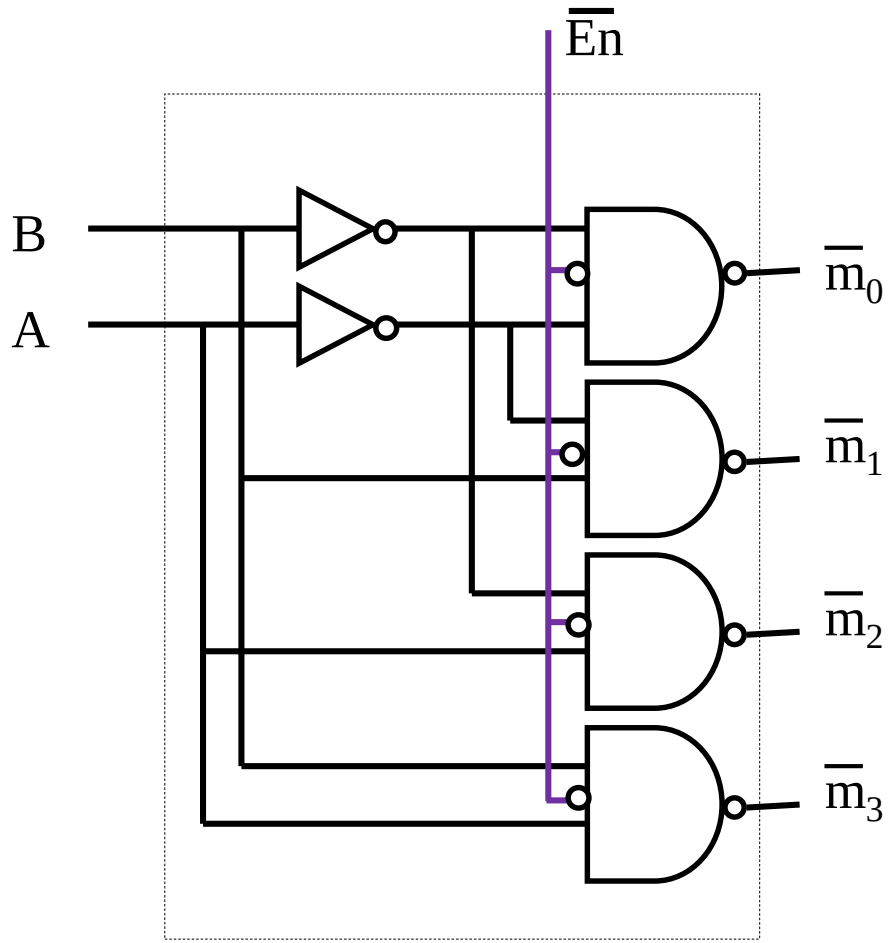
جدول صحت

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

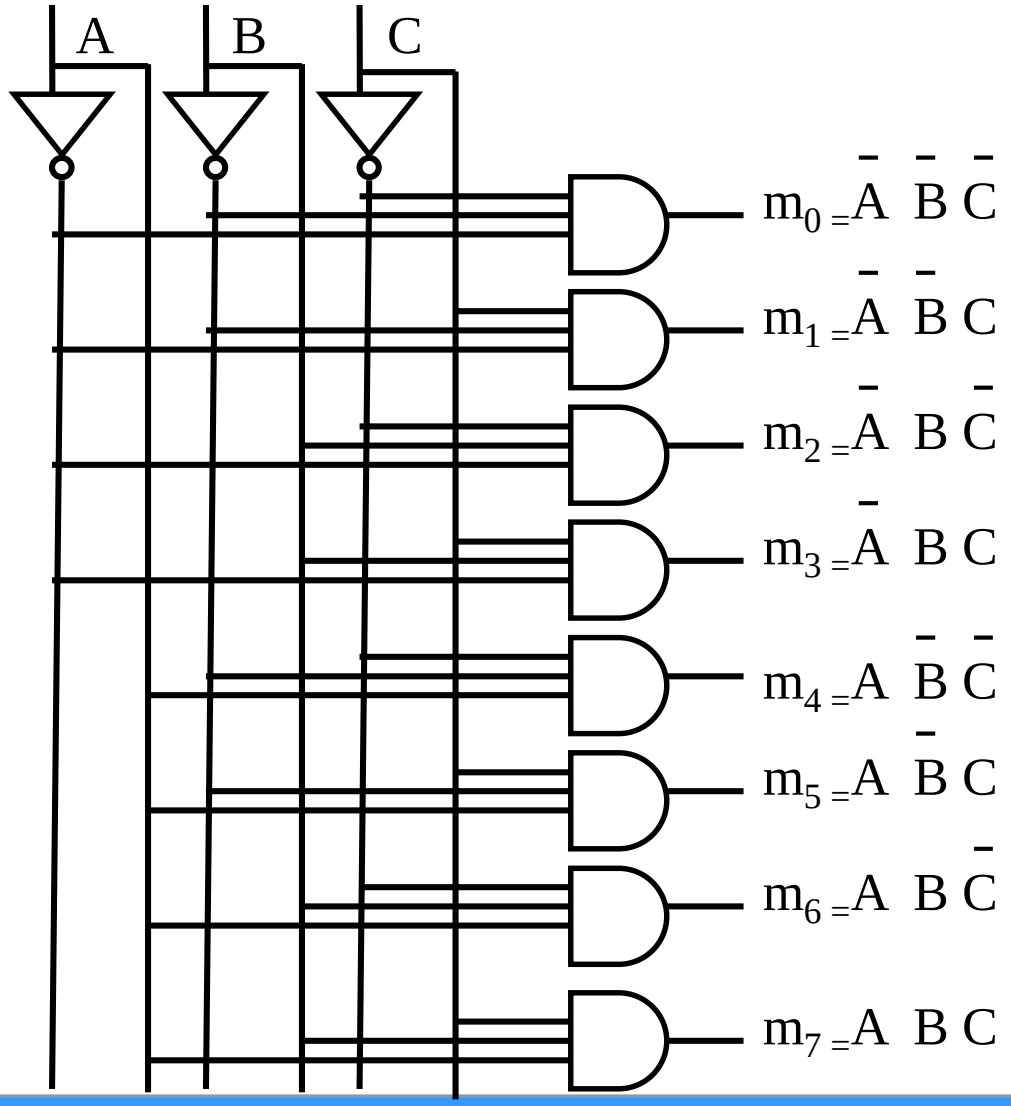


دیاگرام منطقی (موازی و خروجی های فعال پایین)

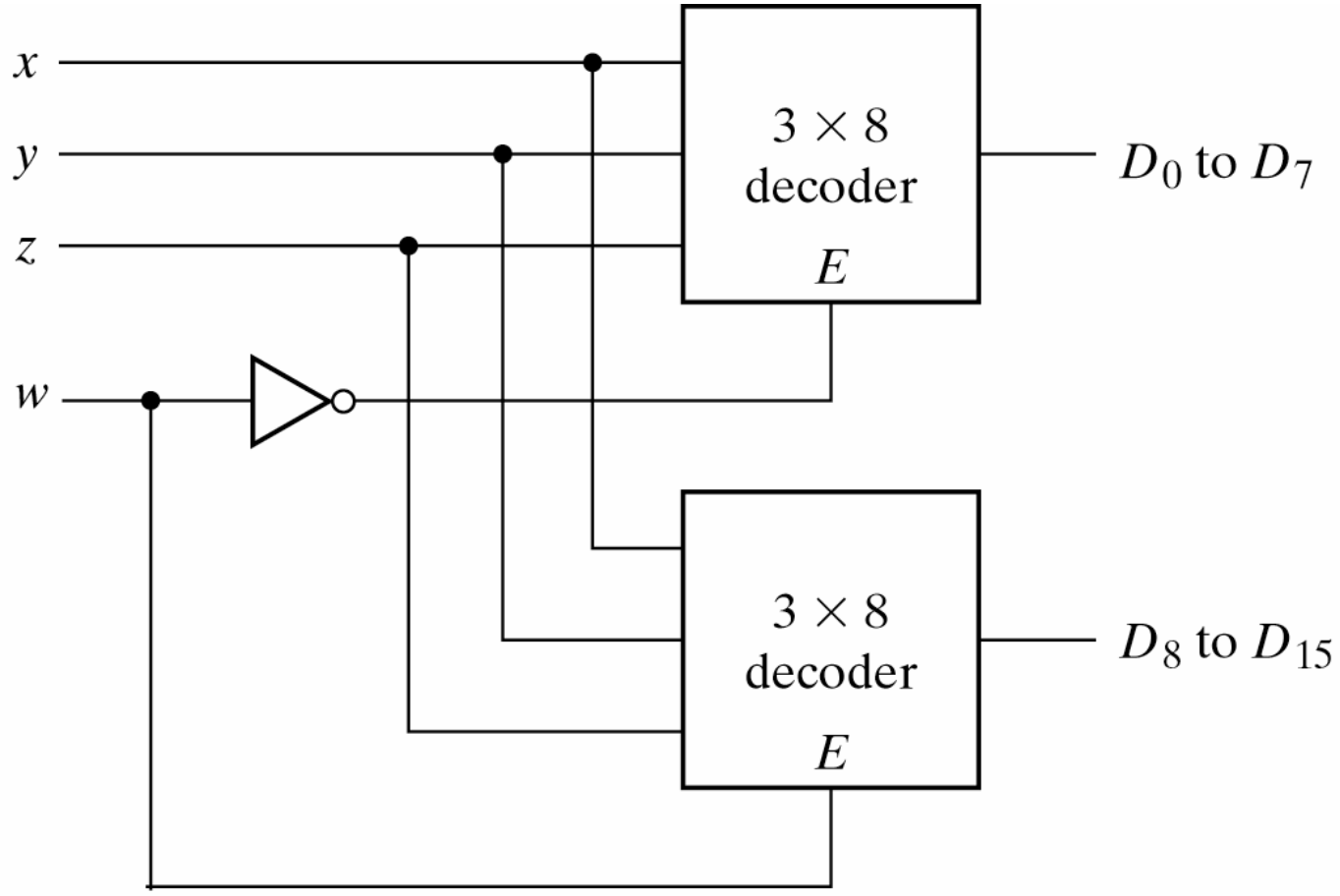




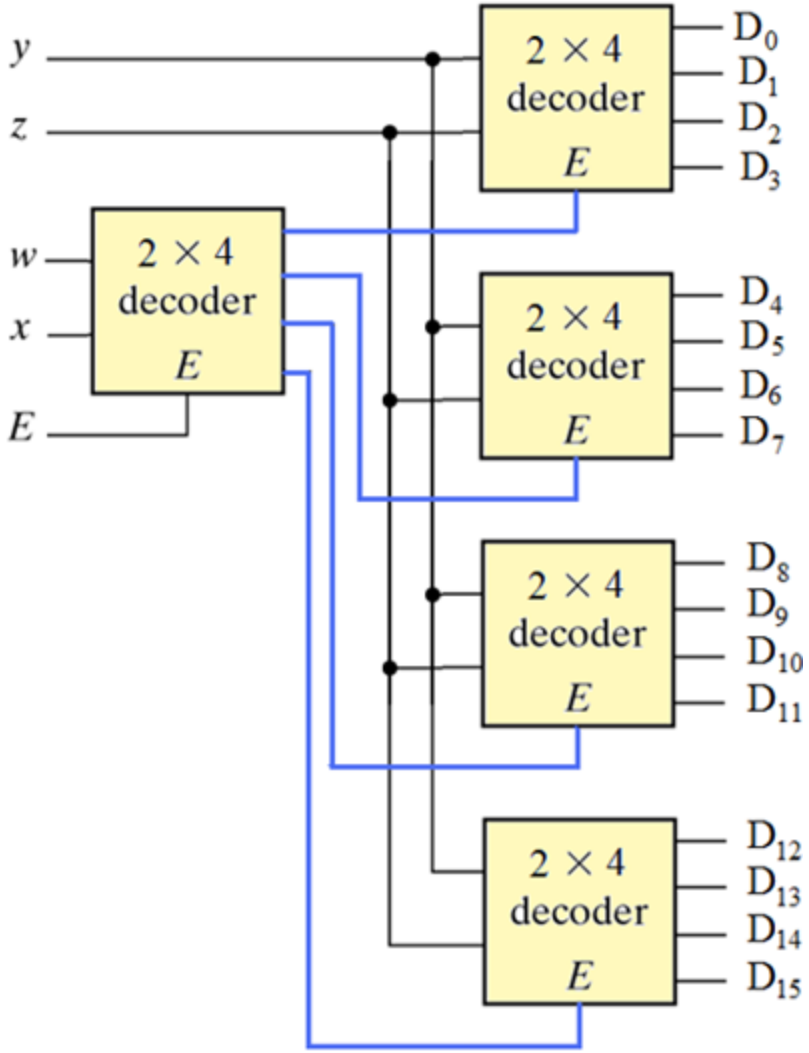
دیکدر نوع موازی سه بیت



ساختن دیکدر بزرگتر:

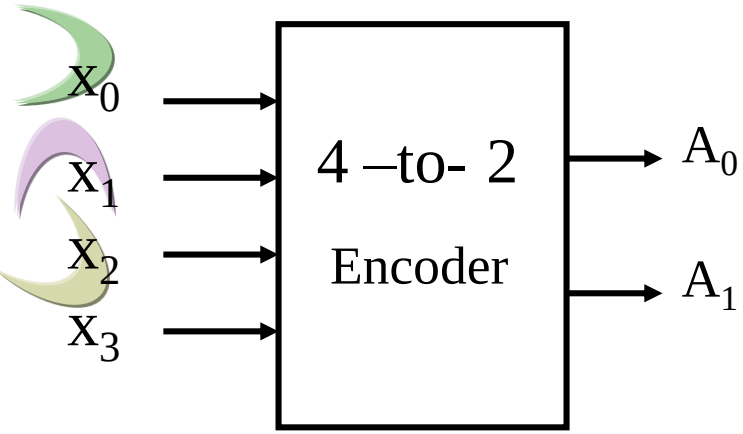


ساختن دیکدر بزرگتر:



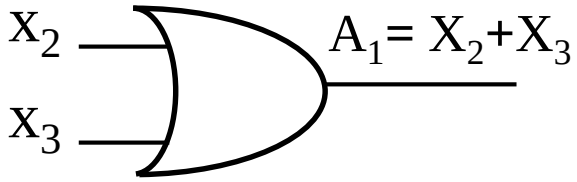
طراحی انکدر

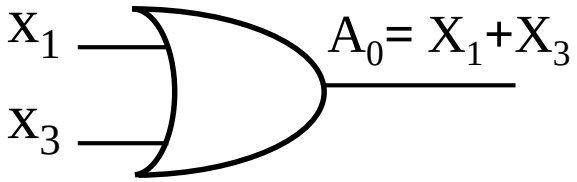
- اینکدر یک مدار ترکیبی است که برای فعال بودن هر کدام از ورودی ها یک کد خروجی منحصر به فرد را ایجاد می کند.
- اگر یک ماجول اینکدر n ورودی داشته باشد خروجی s باید در رابطه زیر صدق کند: $2^s \leq n$ or $s \leq \text{Log}_2 n$



ورودی فعال	A_1	A_0
x_0	0	0
x_1	0	1
x_2	1	0
x_3	1	1

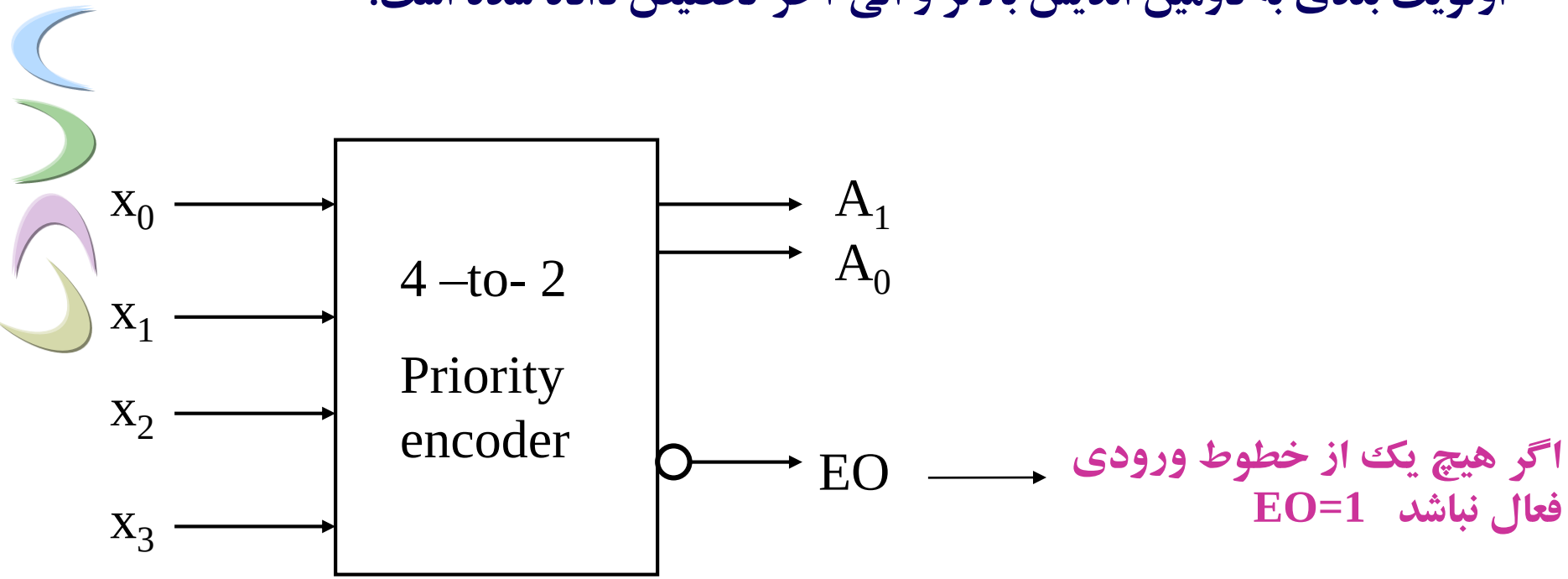
- به عنوان مثال شماتیک کلی مدار یک انکدر برای برای چهار خط ورودی که در هر لحظه از زمان فقط یکی از ورودی فعال باشد بصورت مقابل خواهد بود.





انکدر با اولویت

- اینکدر با اولویت اجازه می دهد تا چندین خط ورودی فعال شوند ولی عدد دودویی خروجی 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.



انکدر با اولویت

X_1X_0 \ X_3X_2	00	01	11	10
00	0	1	1	1
01	0	1	1	1
11	0	1	1	1
10	0	1	1	1

$$A_1 = X_2 + X_3$$

X_1X_0 \ X_3X_2	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	1	0	1	1
10	1	0	1	1

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

$$A_0 = \overline{X}_2 X_1 + X_3$$

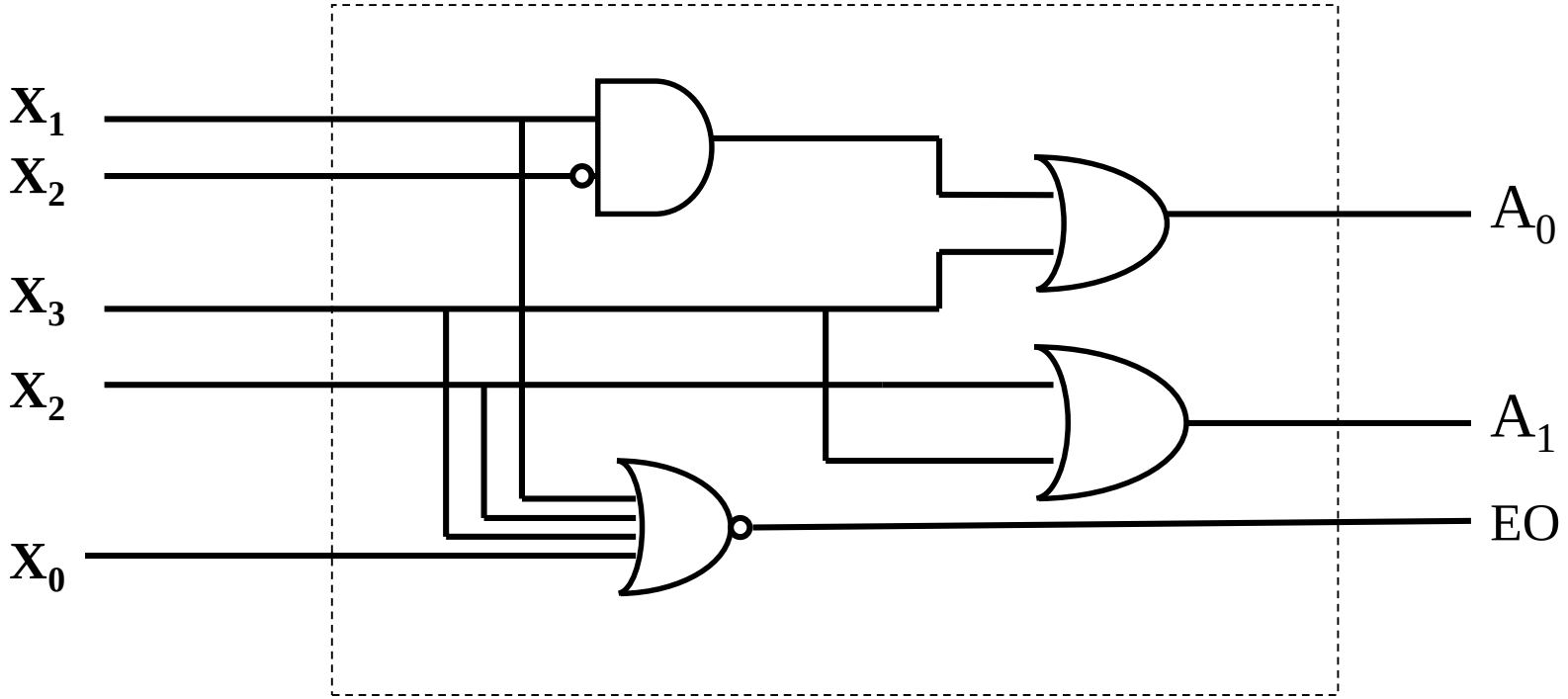
x_3	x_2	x_1	x_0	A_1	A_0	EO
0	0	0	0	0	0	1
0	0	0	1	0	0	0
0	0	1	0	0	1	0
0	0	1	1	0	1	0
0	1	0	0	1	0	0
0	1	0	1	1	0	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0
1	0	0	0	1	1	0
1	0	0	1	1	1	0
1	0	1	0	1	1	0
1	0	1	1	1	1	0
1	1	0	0	1	1	0
1	1	0	1	1	1	0
1	1	1	0	1	1	0
1	1	1	1	1	1	0

انکدر با اولویت

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

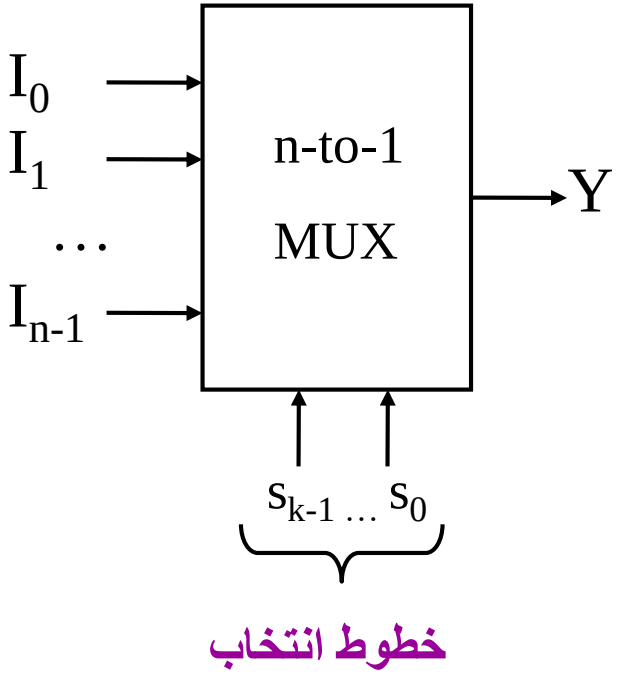
$$A_1= X_2+X_3$$

$$A_0= \overline{X}_2X_1+X_3$$



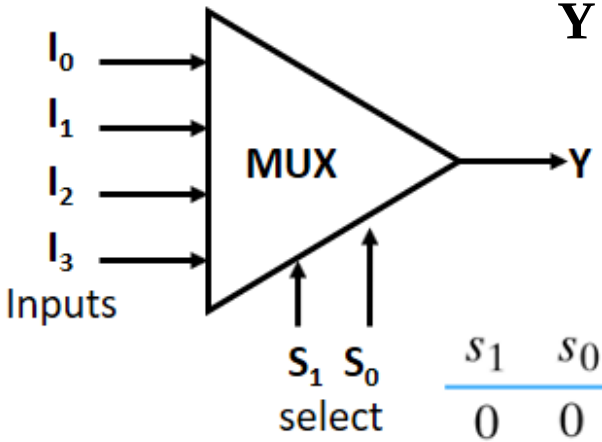
طراحی مالتی پلکسر(تسهیم کننده)

- بطور کلی مالتی پلکسر(انتخابگر داده) یک مدار ترکیبی است که یکی از چند خط ورودی را انتخاب و آن را به خط خروجی منتقل می سازد.



طراحی مالتی پلکسر(تسهیم کننده)

$$Y = \bar{s}_1\bar{s}_0 I_0 + \bar{s}_1 s_0 I_1 + \bar{s}_1 s_0 I_2 + s_1 s_0 I_3$$

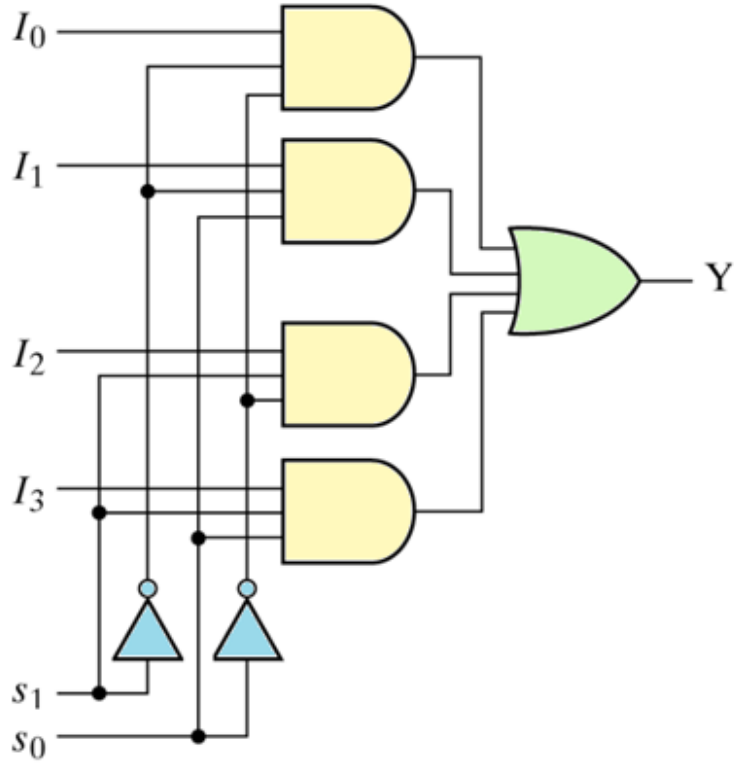


Function table

s_1	s_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

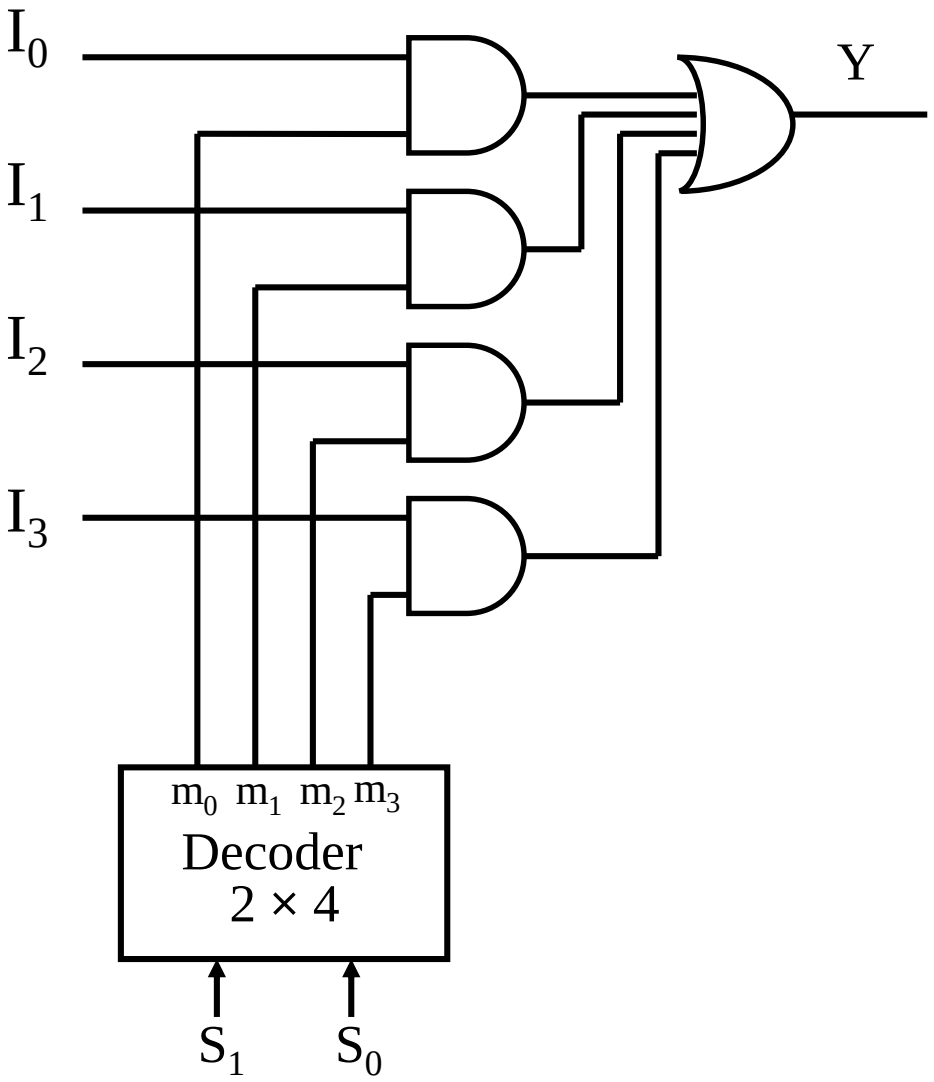
Truth table

s_1	s_0	I_0	I_1	I_2	I_3	Y
0	0	0	X	X	X	0
0	0	1	X	X	X	1
0	1	X	0	X	X	0
0	1	X	1	X	X	1
1	0	X	X	0	X	0
1	0	X	X	1	X	1
1	1	X	X	X	0	0
1	1	X	X	X	1	1



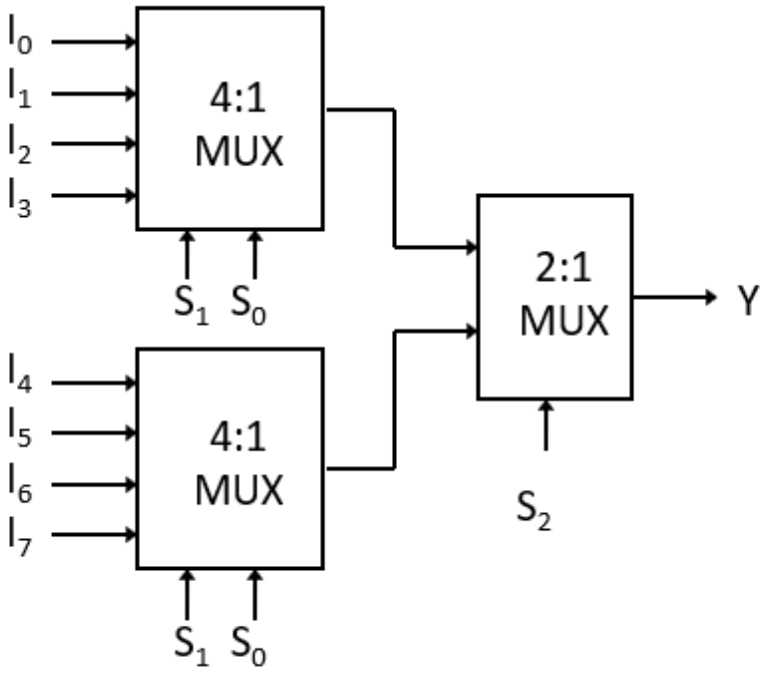
طراحی مالتی پلکسر(تسهیم کننده)

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



ساختن مالتی پلکسر بزرگتر:

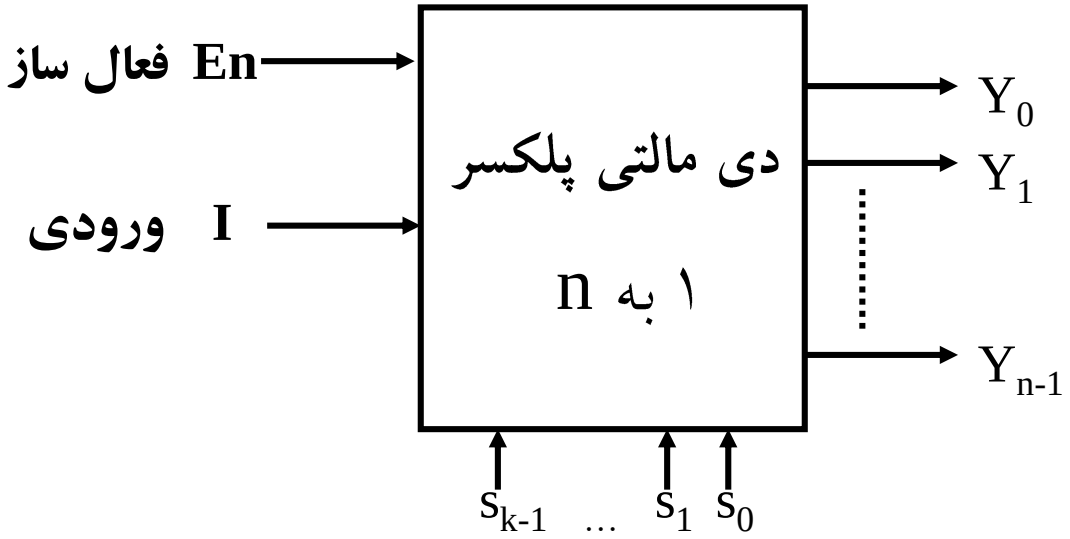
با استفاده از مالتی‌پلکسرهای ۴ به ۱ و ۲ به ۱، یک مالتی‌پلکسر ۸ به ۱ بسازید.



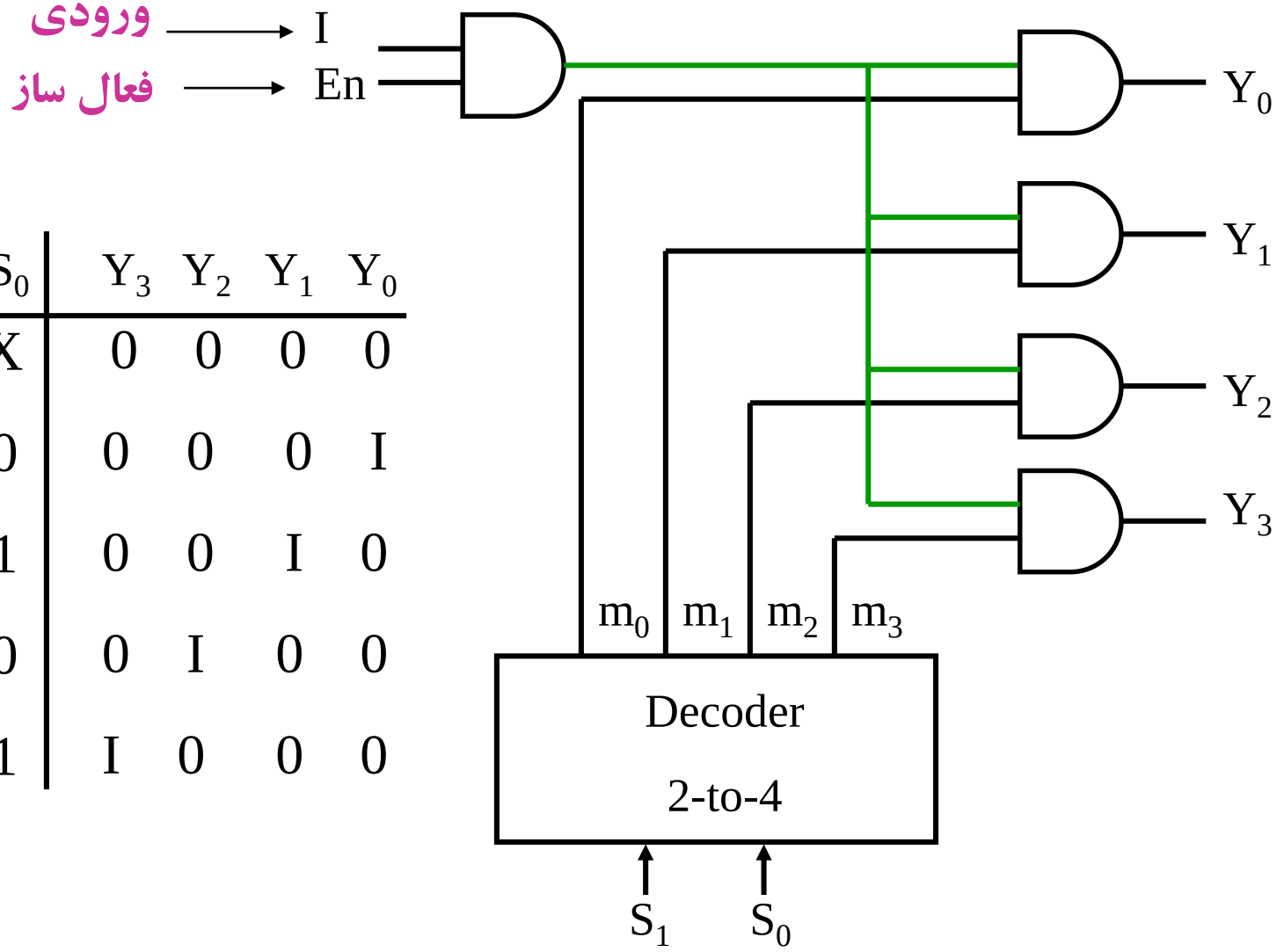
S_2	S_1	S_0	Y
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3
1	0	0	I_4
1	0	1	I_5
1	1	0	I_6
1	1	1	I_7

طراحی دی مالتی پلکسر(پخش کننده داده)

- دی مالتی پلکسر یک مدار منطقی ترکیبی است که یک خط ورودی را به یکی از n خط خروجی وصل می کند.
- انتخاب خط خروجی از طریق خطوط انتخاب s انجام می شود که: $2^k \leq n$
  - در این حالت کد انتخاب برای تولید مینترم های s بکار می رود.



دی مالتی پلکسر ۱ به ۴ با فعال ساز



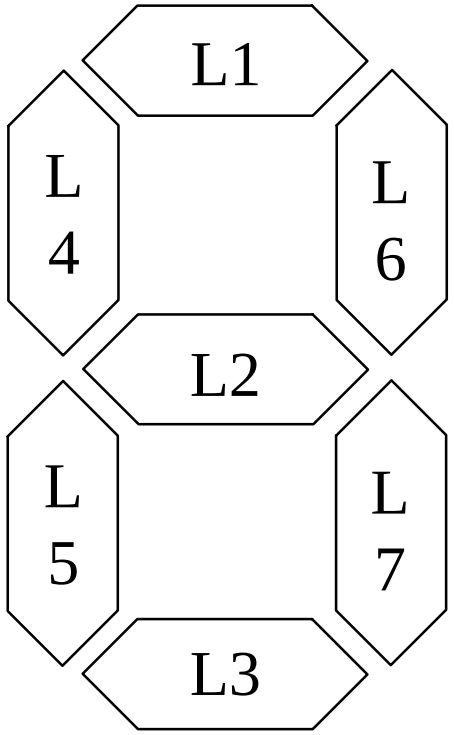
En	S_1	S_0	Y_3	Y_2	Y_1	Y_0
0	X	X	0	0	0	0
1	0	0	0	0	0	I
1	0	1	0	0	I	0
1	1	0	0	I	0	0
1	1	1	I	0	0	0

کاربرد MUX و DMUX

به اشتراک گذاشتن یک کانال ارتباطی بین چندین دستگاه:

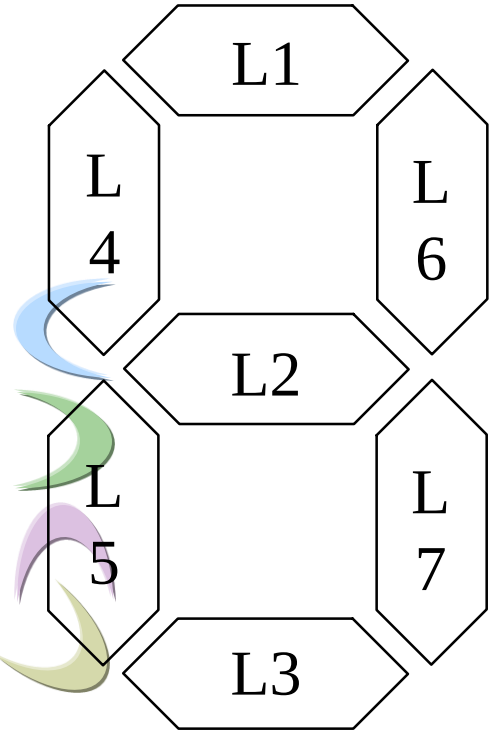
در هر زمان تنها یکی از مبداها و یکی از مقصدها به کانال ارتباطی دسترسی دارند.

نمایشگر ۷ قطعه ای(Seven Segments Display)

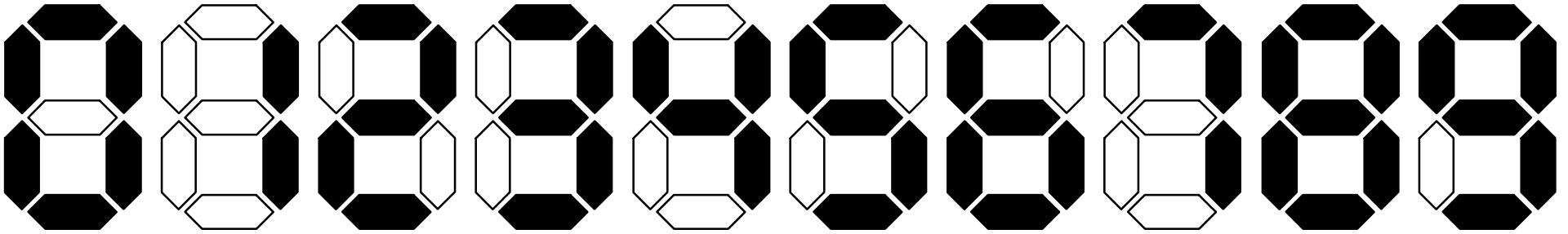


B3	B2	B1	B0	Val
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

نمایشگر ۷ قطعه ای(Seven Segment Display)



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



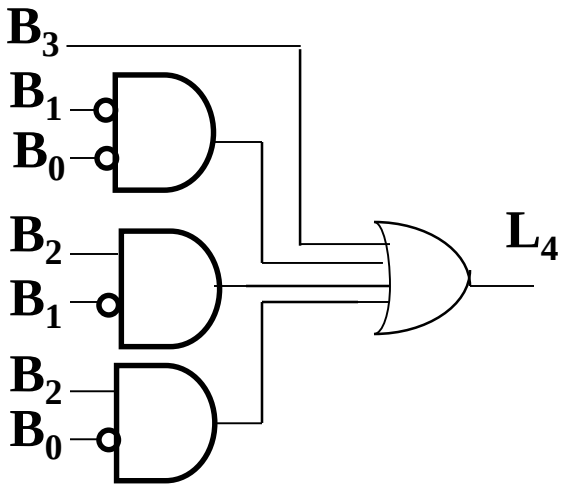
نمایشگر ۷ قطعه ای(Seven Segment Display)

- 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

$$L_4 = \overline{B}_1\overline{B}_0 + B_2\overline{B}_1 + B_2\overline{B}_0 + B_3$$

B_1B_0 \ B_3B_2	00	01	11	10
00	1	1	X	1
01	0	1	X	1
11	0	0	X	X
10	0	1	X	X



- سایر حالات بی اهمیت در نظر گرفته شده است. البته می توان برای آنها نیز الگو تعریف نمود.

تمرین: مدار سایر پایه های کنترلی نمایشگر را به همین روش بدست آورید.