

فصل اول

سیستم‌های دودوئی

مقدمه:

هدف از یادگیری این درس طراحی و ایجاد مدارهایی می‌باشد که وظیفه‌مندی خاصی را دنبال می‌کند، که طراح انتظار دارد. از این درس می‌توان در تهیه ساعتهای دیجیتال، شمارنده‌ها و ... استفاده کرد.

نمایش اعداد:

هر عدد را می‌توان با استفاده از ارقام آن به شکل زیر نمایش داد، همچنین می‌توان با استفاده از رابطه (۱-۱) و بدست آوردن مقدار \sum وزن آن عدد را بدست آورد.

$$N = (a_{n-1}a_{n-2}a_{n-3} \dots a_1a_0 \dots a_{-1}a_{-2}a_{-3} \dots a_{-m})_r = \sum_{i=-m}^{n-1} a_i r^i \quad (1-1)$$

مثال:

وزن عدد ۸ (365) را به دست آورید

$$(365)_8 = (5 \times 8^0) + (6 \times 8^1) + (3 \times 8^2) = 5 + 48 + 192 = 245$$

تبديل مبنای:

تبديل مبنای ۵۵ به ۵۰:

برای این کار از دو راه می‌توان استفاده کرد:

استفاده از تقسیمات متوالی: به این صورت که عدد مبنای 10 را بر 2 تقسیم می‌کنیم و باقی مانده را بدست می‌آوریم. تا زمانی که خارج قسمت کوچکتر از 2 نشده به تقسیم کردن ادامه می‌دهیم. سپس آخرین خارج قسمت (که به هر شکل یک می‌باشد) را به عنوان سمت چپ‌ترین بیت در نظر می‌گیریم. و بعد از آن باقی مانده‌ها را که با صفر و یا یک هستند به ترتیب از سمت راست انتخاب می‌کنیم و می‌نویسیم.
مثال:

عدد ۱۰ (34) را به مبنای 2 تبدیل کنید:

$$\begin{array}{r} 34 & | 2 \\ 34 & | 17 & | 2 \\ 0 & | 16 & | 8 & | 2 \\ & 1 & | 8 & | 4 & | 2 & \Rightarrow (100010)_2 \\ & 0 & | 4 & | 2 & | 2 \\ & 0 & | 2 & | 1 \\ & 0 \end{array}$$

با استفاده از قریقات متوالی: به اینصورت که با در نظر گرفتن عدد مورد نظر باید یک n پنداش کنیم که 2^n بزرگترین عدد باشد که از عدد مورد نظر ما کوچکتر است. به عنوان مثال اگر عدد دلخواه ما 273 بود باید n را 8 در نظر بگیریم یعنی $2^8 = 256$ که از عدد ما کوچکتر است اگر n را 7 فرض می‌کردیم $128 = 2^7$ از عدد ما کوچکتر بود اما هنوز عدد دیگری مانند 256 وجود داشت که به عدد مورد نظر یعنی 273 نزدیکتر بود. پس از انتخاب n حاصل 2^n را از عدد مورد نظر کم می‌کنیم.

$$273 - 2^8 = 273 - 256 = 17$$

بنابراین با توضیحات فوق به این نتیجه می‌رسیم که در عدد 273 یک 256 تا وجود داشته حال باید همین کار را در مورد باقی مانده انجام دهیم.

$$17 \rightarrow n = 4 \rightarrow 17 - 2^4 = 1 \Rightarrow 1 \rightarrow n = 0$$

حال با جایگذاری خواهیم داشت:

$$\begin{array}{ccccccccc} 256 & 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\ (1) & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1_2 \end{array}$$

به بیان ساده‌تر برای تبدیل باید بینیم که در عددی که می‌خواهیم از مبنای 10 به 2 ببریم تا 2^n , چند 2^{n-1} , چند 2^{n-2} و ... و بالاخره چند 2^0 وجود دارد.

تبدیل از مبنای ۱۰ به ۵۵:

با استفاده از وزن اعداد صورت می‌گیرد؛ یعنی می‌توان با استفاده از رابطه (۱-۱) تبدیل مبنای ۲ به ۱۰ را انجام داد.

مثال:

عدد $(100010)_2$ را به مبنای ۱۰ تبدیل کنید.

$$(100010)_2 = (0 \times 2^0) + (1 \times 2^1) + (0 \times 2^2) + (0 \times 2^3) + (0 \times 2^4) + (1 \times 2^5) = 34$$

نکته: در واقع در این روش باید وزن ارقام را از سمت راست بالای ارقام بنویسیم که از 2^0 شروع می‌شود و تا رسیدن به n رقم به 2^n می‌رسد. حال در انتها برای بدست آوردن وزن این اعداد باید فقط وزن‌هایی که پایه یک دارند باهم جمع شوند.

مثال:

وزن $(100110)_2$ را بدست آورید.

$$\begin{array}{ccccccccc} 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ \text{پایه} & & & & & & \end{array}$$

$$2^6 + 2^2 + 2^1 = 64 + 4 + 2 = 70$$

یا به صورت

$$\begin{array}{ccccccccc} 64 & 32 & 16 & 8 & 4 & 2 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ \text{پایه} & & & & & & \end{array}$$

$$= 64 + 4 + 2 = 70$$

تبدیل از مبنای ۱۰ به ۷۰:

با استفاده از تقسیمات متولی صورت می‌گیرد:

به این صورت که باید عدد مبنای 10 را بر 16 تقسیم کنیم و این کار را تا جایی ادامه دهیم که خارج قسمت کوچکتر از 16 شود. سپس با انتخاب آخرین خارج قسمت و همچنین باقی مانده‌ها از راست به چپ شروع به نوشتن عدد می‌کنیم. ممکن است باشد که در تقسیمات تا جایی که باقی مانده‌ها کوچکتر از 10 هستند مشکلی وجود ندارد اما برای باقی مانده‌های 10 تا 15 به شکل زیر عمل می‌کنیم.

$$\begin{array}{l} 10 \rightarrow A \\ 11 \rightarrow B \\ 12 \rightarrow C \\ 13 \rightarrow D \\ 14 \rightarrow E \\ 15 \rightarrow F \end{array}$$

یعنی باید حروف A و B و C و D و E و F را جایگزین کنیم.

مثال:

عدد 346 را به مبنای 16 ببرید.

346 | 16

$$\begin{array}{r} 336 \quad 21 \quad | 16 \\ \hline 10 \quad 16 \quad 1 \end{array} \quad (1510) \rightarrow (15A)$$

(1510) تبدیل می شود به (15A)

تبدیل از مبنای شانزده به ۱۶:

با استفاده از وزن اعداد صورت می گیرد؛ یعنی با استفاده از رابطه (۱-۱) تبدیل مبنای 16 به 10 انجام می دهیم.

مثال:

عدد 16(F) را به مبنای 10 تبدیل کند.

$$(15F)_{16} = (F \times 16^0) + (5 \times 16^1) + (1 \times 16^2) = 15 + 80 + 256 = 351$$

تبدیل از مبنای ۱۰ به هشت:

با استفاده از تقسیمات متوالی انجام می شود به این صورت که عدد مبنای 10 را بر ۸ تقسیم می کنیم و باقی مانده را بدست می آوریم، سپس خارج قسمت را بر ۸ تقسیم می کنیم و به همین شکل ادامه می دهیم تا زمانی که خارج قسمت کوچکتر از ۸ شود، سپس از راست به چپ شروع به نوشتن ارقام می کنیم.

مثال:

عدد 245 را به مبنای 8 تبدیل کند.

$$\begin{array}{r} 245 | 8 \\ 240 \quad 30 | 8 \\ \hline 5 \quad 24 \quad 3 \end{array} \Rightarrow (365)_8 = (245)_{10}$$

تبدیل مبنای هشت به ۱۰:

این عمل یا کمک وزن اعداد صورت می گیرد به این صورت که با استفاده از رابطه (۱-۱) می توان تبدیل مبنای هشت به ۱۰ را انجام داد.

مثال:

$$(361)_8 = (1 \times 8^0) + (6 \times 8^1) + (3 \times 8^2) = 1 + 48 + 192 = 241$$

تبدیل مبنای دو به شانزده:

از سمت راست جداسازی بصورت 4 بیت 4 یعنی انجام میدهیم سپس با استفاده از وزن اعداد به جواب می‌رسیم.
 فقط باید توجه داشت که در هنگام جداسازی اگر در دسته آخر کمتر از 4 یعنی قرار گرفت به اندازه تعداد بیت کمتر سمت چپ صفر می‌گذاریم و دسته را کامل می‌کنیم. هر یعنی رادر وزن متناظرش ضرب می‌کنیم و اعداد بدست آمده در یک دسته را با هم جمع می‌کنیم.

مثال:

(100010)₂ را به مبنای 16 ببرید.

$$\begin{array}{r}
 8 \ 4 \ 2 \ 1 \quad 8 \ 4 \ 2 \ 1 \\
 (0 \ 0 \ 1 \ 0 \quad 0 \ 0 \ 1 \ 0)_2 \\
 \downarrow \ \downarrow \ \downarrow \ \downarrow \quad \downarrow \ \downarrow \ \downarrow \ \downarrow \\
 \underline{0+0+2+0} \quad \underline{0+0+2+0} \\
 \quad \quad \quad \quad \Rightarrow = (22)_{16}
 \end{array}$$

تبدیل مبنای شانزده به دو:

برای تبدیل مبنای شانزده به دو باید هر رقم را به صورت چهار یتی در مبنای دو نوشت لازم به ذکر است که جایگاه ارقام نباید تغییر کند.

مبنای 16 مبنای 2

0	0000
1	0001
2	0010
3	0011
4	0100
:	:
A \leftarrow 10	1010
B \leftarrow 11	1011
C \leftarrow 12	1100
D \leftarrow 13	1101
E \leftarrow 14	1110
F \leftarrow 15	1111

مبنای 16 = 4 یعنی

مثال: عدد (1 6 D) در مبنای 16 را به مبنای 2 تبدیل کنید.

$$(1 \ 6 \ D)_{16} \rightarrow \begin{cases} 1 \rightarrow 0001 \\ 6 \rightarrow 0110 \\ D \rightarrow 1101 \end{cases} \rightarrow (000101101101)_2$$

تبدیل مبنای دو به هشت:

در این تبدیل جداسازی 3 یعنی انجام می‌دهیم:

به اینصورت که از سمت راست سه یعنی سه یعنی جدا می‌کنیم سپس با استفاده از مقداردهی وزنی، وزن هر دسته را به دست می‌آوریم.
 باید توجه کرد اگر آخرین دسته تعداد بیتهاش کمتر از سه بود به اندازه تعداد بیت کمتر صفر در سمت چپ قرار می‌دهیم تا دسته کامل شود.
 هر یعنی رادر وزن متناظرش ضرب می‌کنیم و اعداد بدست آمده در یک دسته را با هم جمع می‌کنیم.

مثال:

عدد $(100010)_2$ را به مبنای 8 تبدیل کنید.

$$100010 \Rightarrow \begin{cases} 2 = \text{با استفاده از وزنها} \Rightarrow 010: \text{دسته اول} \\ 4 = \text{با استفاده از وزنها} \Rightarrow 100: \text{دسته دوم} \end{cases} \Rightarrow (42)_8$$

تبدیل مبنای هشت به دو:

برای تبدیل مبنای 8 به دو باید هر رقم را به صورت مبنای 2 اما به صورت سه یکتی بنویسیم. ضمناً باید توجه کرد که جایگاه ارقام نباید تغییر کند.

مبنای 8 = 3 بیت

مبنای 8	مبنای 2
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

مثال:

عدد $(712)_8$ را به مبنای 2 تبدیل کنید:

$$(712)_8 = (?)_2 \Rightarrow \begin{cases} 2 \rightarrow 010 \\ 1 \rightarrow 001 \\ 7 \rightarrow 111 \end{cases} \Rightarrow (111001010)_2$$

تبدیل مبنای شانزده به هشت:

برای اینکه تبدیل مبنای 16 به 8 را انجام دهیم بهتر است ابتدا عدد در مبنای 16 را به مبنای 2 بیریم سپس عدد بدست آمده در مبنای 2 را به مبنای 8 تبدیل کنیم.

مثال:

عدد $(5D3)_{16}$ را به مبنای 8 بیریم.

$$(5D3)_{16} = (010111010011)_2 = (2723)_8$$

تبدیل مبنای هشت به شانزده:

برای تبدیل مبنای 8 به 16 می‌توان عمل عکس تبدیل 16 به 8 را انجام داد به این معنی که ابتدا عدد مورد نظر را به مبنای 2 بیریم سپس حاصل را به مبنای 16 بازگردانیم.

مثال:

عدد $(376)_8$ را به مبنای 16 تبدیل کنید.

$$(376)_8 = (01111110)_2 = (\text{FE})_{16}$$

برای تبدیل عدد اعشاری در هر مبنای 10 با استفاده از رابطه $(1-1)$ باید وزن آن را بدست آوریم.

مثال:

عدد $(100010/110)_2$ را به مبنای 10 تبدیل کنید:

$$(100010/110)_2 = \begin{cases} \text{قسمت صحیح} & = (0 \times 2^0) + (1 \times 2^1) + (0 \times 2^2) + (0 \times 2^3) + (0 \times 2^4) + (1 \times 2^5) = 34 \\ \text{قسمت اعشاری} & = (1 \times 2^{-1}) + (1 \times 2^{-2}) + (0 \times 2^{-3}) = 0.75 \end{cases}$$

بنابراین حاصل برابر است با:

$$(34.75)_{10}$$

تبدیل مبنای اعشاری:

برای تبدیل عدد اعشاری در مبنای ده به هر مبنای (n) (به عنوان مثال مبنای 2) باید عمل ضرب متوالی را انجام دهیم، عملیات به این شکل است که قسمت صحیح را به مانند قبل و با استفاده از روشایی که قبلاً اشاره شد، بدست می‌آوریم. لذا برای بدست آوردن قسمت اعشاری به شکل زیر عمل می‌کنیم:

ابتدا قسمت اعشار را جدا می‌کنیم و آن را در مبنای مورد نظر یعنی 2 ضرب می‌کنیم حاصل این ضرب یک قسمت صحیح و یک قسمت اعشاری خواهد داشت. حال قسمت صحیح را جدا می‌کنیم و قسمت اعشاری را در مبنای 2 دوباره ضرب می‌کنیم، این عمل را تا جایی ادامه می‌دهیم تا به قسمت اعشاری صفر برسیم و یا اینکه به یک دوره گردش دست پیدا کنیم.

توجه: اگر در صورت سوال خواسته شده بود که تا n رقم اعشار را حساب کنیم باید n بار عمل ضرب را انجام دهیم.

مثال:

عدد $(0.513)_{10}$ را به مبنای 2 و 8 تبدیل کنید.

$$\begin{aligned} (0.513)_{10} &= 0.513 \times 2 = 1.026 \rightarrow 0.026 \\ &0.026 \times 2 = 0.052 \\ &0.052 \times 2 = 0.104 \\ &0.104 \times 2 = 0.208 \\ &0.208 \times 2 = 0.416 \\ &0.416 \times 2 = 0.832 \end{aligned}$$

$$(0.513)_{10} = (0.10000\dots)_2$$

$$\begin{aligned} (0.513)_{10} &= 0.513 \times 8 = 4.104 \\ &0.104 \times 8 = 0.832 \\ &0.832 \times 8 = 6.656 \\ &0.656 \times 8 = 5.248 \\ &0.248 \times 8 = 1.984 \end{aligned}$$

$$(0.513)_{10} = (0.40651\dots)_8$$

محاسبات:

اگر بخواهیم عملیات جمع را در مبنای ۲ انجام دهیم به این ترتیب عمل می‌کنیم که، ابتدا ارقام از سمت راست زیر هم قرار می‌گیرند سپس از همان سمت راست شروع به جمع رقم به رقم می‌کنیم به این صورت که پس از جمع دو رقم دو حالت ممکن است برای حاصل جمع پیش بیاید. حالت اول زمانی است که حاصل جمع بزرگتر یا مبنای ۲ باشد که در این صورت باید یک مبنای ۲ از حاصل جمع کم شود باقی مانده نوشته می‌شود و یک عدد "۱" به بالای رقم بعدی انتقال می‌باید. اما اگر حاصل جمع کوچکتر از مینا، بود تنها کافیست که همان حاصل نوشته شود.

در زمانی که دو عدد n رقمی را با هم جمع می‌کنیم انتظار داریم که حاصل نیز عدد n رقمی باشد اما زمان‌هایی به وجود می‌آید. که حاصل $n+1$ رقمی می‌شود به این رقم جدید به وجود آمده رقم نقلی یا carry گفته می‌شود.

مثال:

اعداد ${}_8(457)$ و ${}_8(203)$ را جمع کنید.

 457 $+ 203$

ابتدا ۳ و ۷ با یکدیگر جمع می‌شود حاصل عدد ۱۰ خواهد بود که چون حاصل بزرگتر از مبنای ۸ می‌باشد باید یک مبنای ۸ از ۱۰ کم شود.

10

حاصل بدست آمده را در ستون اول می‌نویسیم و یک عدد ۱ در بالای رقم سمت چپ می‌نویسیم.

8
1 $\frac{1}{457}$ $+ 203$ $\frac{2}{}$

حال باید ۵ و صفر و یک جمع شود حاصل ۶ می‌شود که از مبنای ۸ کوچکتر است. بنابراین خودش را به عنوان حاصل جمع می‌نویسیم.

 $\frac{1}{457}$ $+ 203$ $\frac{662}{}$

حال اگر بخواهیم عملیات تفریق در مبنای ۲ را انجام دهیم به این صورت عمل می‌کنیم که ابتدا ارقام از سمت راست زیر هم قرار گیرند سپس از همان سمت راست شروع به عملیات تفریق رقم به رقم می‌کنیم. در این حالت ممکن است رقم عدد بالایی بزرگتر از رقم عدد پائینی باشد می‌توان به راحتی عمل تفریق را انجام داد.

اما اگر رقم بالایی کوچکتر از رقم پائینی باشد در این صورت باید از رقم سمت چپ یکی قرض گرفت البته این عملیات در صورتی که رقم سمت چپ صفر باشد باید از رقم سمت چپ بعدی برای آن صفر قرض گرفت. پس از قرض گرفتن باید به رقم سمت راست یک مبنای اضافه شود و سپس عملیات تفریق انجام می‌شود.

مثال: تفریق زیر را در مبنای ۱۶ انجام دهید.

A05

 $- 3B7$

چون رقم سمت راست عدد اول کوچکتر از رقم سمت راست عدد دوم است بنا بر این باید از رقم دوم قرض گرفته و چون این رقم صفر است باید از رقم سوم قرض گرفته شود. بنا بر این عدد A تبدیل به 9 می شود و رقم دوم یعنی صفر تبدیل به 16 می شود (بک عدد مبنای آن اضافه می شود) حال باید از این عدد یکی کم شود و رقم اول یعنی 5 تبدیل به $(16 + 5 = 21)$ می شود.

$$\begin{array}{r}
 15 \\
 9 \quad 16 \quad 21 \\
 \cancel{\cancel{A}} \quad \cancel{\cancel{B}} \quad 5 \\
 - \quad 3 \quad B \quad 7 \\
 \hline
 6 \quad 4 \quad E
 \end{array}$$

مثال: جمع و تفریق زیر را در مبنای 2 انجام دهید:

$$\begin{array}{r}
 1 \ 1 \ 1 \ 1 \ 1 \\
 1 \ 1 \ 0 \ 1 \ 0 \ 1 \\
 + 0 \ 1 \ 1 \ 1 \ 1 \ 1 \\
 \hline
 1 \ 0 \ 1 \ 0 \ 1 \ 0 \ 0
 \end{array}
 \qquad
 \begin{array}{r}
 2 \ 1 \ 2 \\
 0 \ 1 \ \cancel{2} \ \cancel{0} \ 2 \\
 \cancel{1} \ \cancel{1} \ \cancel{0} \ \cancel{1} \ \cancel{0} \ \cancel{1} \\
 - 0 \ 1 \ 1 \ 1 \ 1 \ 1 \\
 \hline
 0 \ 1 \ 0 \ 1 \ 1 \ 0
 \end{array}$$

برای انجام عملیات ضرب در مبنای 2 نیز باید مانند عملیات جمع و قوانین مربوط به ضرب را در نظر بگیریم.

مثال: ضرب زیر را در مبنای 16 انجام دهید.

A2F

 $\times 50E$

ابتدا E را در عدد A2F ضرب می کنیم این کار را باید رقم انجام دهیم. یعنی ابتدا E را در F ضرب می کنیم در این صورت حاصل برابر 210 می شود حال باید بینیم که چند مبنای در این حاصل وجود دارد. مشاهده می شود که در عدد 210 می توان 13 عدد 16 ایجاد کرد. حال باید 13 عدد 16 را از 210 کم کنیم. در این صورت خواهیم داشت:

$$\begin{array}{r}
 210 \\
 - 13 \times 16 \\
 \hline
 2
 \end{array}$$

باقي مانده 2 به عنوان رقم سمت راست قرار می گیرد و 13 یا D روی رقم سمت چپ قرار می گیرد و سپس E در 2 ضرب می کنیم حاصل 28 می شود این حاصل با 13 یا D که از مرحله قبل آمده جمع می شود. حاصل 41 می شود همان طور که مشاهده می شود دو مبنای 16 در 41 موجود است.

$$\begin{array}{r}
 41 \\
 - 2 \times 16 \\
 \hline
 9
 \end{array}$$

حاصل 9 به عنوان جواب قرار می گیرد و 2 در روی رقم سمت چپ قرار می گیرد. حال با همین روش گفته شده E را در A ضرب می کنیم حاصل 140 می شود و با 2 جمع می کنیم. حاصل 142 می شود که در آن 8 بار مبنای 16 وجود دارد. باید از 142 این 8 مبنای 16 را کم کنیم.

$$\begin{array}{r}
 142 \\
 - 8 \times 16 \\
 \hline
 E \text{ با } 14
 \end{array}$$

مدار منطقی

حاصل 14 یا E می نویسیم و 8 را بالای رقم سمت چپ می بریم اما چون رقم دیگری وجود ندارد 8 پائین می آید.

$$\begin{array}{r} 2 \quad 13 \\ A \quad 2 \quad F \\ \times \quad \quad \quad E \\ \hline 8 \quad E \quad 9 \quad 2 \end{array}$$

حال یک رقم به سمت چپ شیفت می دهیم.

چون یک رقم صفر است می توان دوبار به سمت چپ شیفت دهیم.

$$\begin{array}{r} A \quad 2 \quad F \\ \times \quad 5 \quad 0 \quad E \\ \hline 8 \quad E \quad 9 \quad 2 \end{array}$$

$$\begin{array}{r} 1 \\ 8 \quad E \quad 9 \quad 2 \\ + \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \\ + \quad 3 \quad 2 \quad E \quad B \quad 0 \quad 0 \\ \hline 3 \quad 3 \quad 7 \quad 9 \quad 9 \quad 2 \end{array}$$

$$\begin{array}{r} 4 \\ A \quad 2 \quad F \\ \times \quad \quad \quad 5 \\ \hline 3 \quad 2 \quad E \quad B \end{array}$$

قوانین جمع و تفریق و ضرب در مبنای 2 داریم:

$$\begin{array}{r} +1 \\ +1 \\ \hline \text{carry} \end{array}$$

$$\begin{array}{r} +0 \\ +1 \\ \hline 1 \end{array}$$

$$\begin{array}{r} +1 \\ +0 \\ \hline 1 \end{array}$$

$$\begin{array}{r} +0 \\ +0 \\ \hline 0 \end{array}$$

مثال:

حاصل جمع زیر را بدست آورید:

$$\begin{array}{r} 1001101 \\ +0111101 \\ \hline 1 \ 0001010 \end{array}$$

carry (رقم نقلی)

تفریق:

در تفریق با پاد می داریم:

$$\begin{array}{r} -1 \\ -1 \\ \hline 0 \end{array}$$

$$\begin{array}{r} 0 \\ -1 \\ \hline \text{①} \end{array}$$

$$\begin{array}{r} 0 \\ -0 \\ \hline 0 \end{array}$$

مثال:

حاصل تفریق زیر را بدست آورید:

$$\begin{array}{r} 1100101 \\ -0111010 \\ \hline 0101011 \end{array}$$

ضرب:

در ضرب با اینری داریم:

$$\begin{array}{r} \times 1 \\ \times 0 \\ \hline 1 \\ 0 \\ \hline 1 \\ 0 \\ 0 \\ 0 \end{array}$$

مثال:

حاصل ضرب زیر را بدست آورید:

$$\begin{array}{r} 1011 \\ \times 1101 \\ \hline 1011 \\ + 0000 \\ + 1101 \\ + 1101 \\ \hline 10100111 \end{array}$$

قوانين جمع و ضرب در مبنای ۲

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

قوانين مربوط به جمع

*	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	10	12	14	16
3	0	3	6	11	14	17	22	25
4	0	4	10	14	20	24	30	34
5	0	5	12	17	24	31	36	43
6	0	6	14	22	30	36	44	52
7	0	7	16	25	34	43	52	61

قوانين مربوط به ضرب

مدار منطقی

قوانين جمع و ضرب در مبنای 10

+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	10
2	2	3	4	5	6	7	8	9	10	11
3	3	4	5	6	7	8	9	10	11	12
4	4	5	6	7	8	9	10	11	12	13
5	5	6	7	8	9	10	11	12	13	14
6	6	7	8	9	10	11	12	13	14	15
7	7	8	9	10	11	12	13	14	15	16
8	8	9	10	11	12	13	14	15	16	17
9	9	10	11	12	13	14	15	16	17	18

قوانين مربوط به جمع

×	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	10	12	14	16	18
3	0	3	6	9	12	15	18	21	24	27
4	0	4	8	12	16	20	24	28	32	36
5	0	5	10	15	20	25	30	35	40	45
6	0	6	12	18	24	30	36	4	48	54
7	0	7	14	21	28	35	42	49	56	63
8	0	8	16	24	32	40	48	56	64	72
9	0	9	18	27	36	45	54	63	72	81

قوانين مربوط به ضرب

قوانين جمع و ضرب در مبنای 16

قوانين مربوط به جمع

+	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
2	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11
3	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
4	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13
5	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14
6	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15
7	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16
8	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17
9	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18
A	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19
B	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A
C	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	0	2	4	6	8	A	C	E	10	12	14	16	18	1A	1C	1E
3	0	3	6	9	C	F	12	15	18	1B	1E	21	24	27	2A	2D
4	0	4	8	C	10	14	18	1C	20	24	28	2C	30	34	38	3C
5	0	5	A	F	14	19	1E	23	28	2D	32	37	3C	41	46	4B
6	0	6	C	12	18	1E	24	2A	30	36	3C	42	48	4E	54	5A
7	0	7	E	15	1C	23	2A	31	38	3F	46	4D	54	5B	62	69
8	0	8	10	18	20	28	30	38	40	48	50	58	60	68	70	78
9	0	9	18	1B	24	2D	36	3F	48	51	5A	63	6C	75	7E	87
A	0	A	14	1E	28	32	3C	46	50	5A	64	6E	78	82	8C	96
B	0	B	16	21	2C	37	42	4D	58	63	6E	79	84	90	9C	A8
C	0	C	18	24	30	3C	48	54	60	6C	73	84	90	A9	B6	C3
D	0	D	1A	27	34	41	4E	5B	68	75	82	8F	9C	A8	B6	C4
E	0	E	1C	2A	38	46	54	62	70	7E	8C	9A	A5	B4	C3	D2
F	0	F	1E	2D	3C	4B	5A	69	78	87	96	A5	B4	C3	D2	E1

مکمل گیری:

دو نوع مکمل گیری وجود دارد:

مکمل r

مکمل r-1

اگر بخواهیم از عدد N مکمل گیری کنیم، خواهیم داشت که:

II تعداد ارقام صحیح

I مبنای عدد

III تعداد ارقام اعشاری

مکمل r عدد N

[N]_{r-1} : N

رابطه (۱-۲)

$$[N]_r = r^n - (N)_r$$

رابطه (۱-۳)

$$[N]_{r-1} = r^n - r^{-m} - (N)_r$$

اگر عدد N، رقم اعشار نداشته باشد برای بدست آوردن مکمل r-1 آن باید m را صفر در نظر بگیریم.

$$m=0 \Rightarrow [N]_{r-1} = r^n - 1 - (N)_r$$

اگر پایه مورد بحث ما دسیمال دهدهی (Decimal) بود یعنی 10^m می توان هم از روابط (۱-۲) و (۱-۳) برای بدست آوردن مکمل r و r-1 استفاده کرد و هم از روش زیر:

اگر بخواهیم مکمل r یک عدد در مبنای ۲ را بدست آوریم کافیست بالای تمام ارقام آن عدد صفر بگذاریم و یک عدد یک در سمت چپ آن صفرها بگذاریم و سپس عمل تفریق را انجام دهیم. در ضمن اگر بخواهیم مکمل (1-r) یک عدد در مبنای ۲ را بدست آوریم باید بالای هر رقم آن یک (1-r) قرار دهیم و سپس عمل تفریق را انجام دهیم.

مثال:

مکمل 10 عدد 256 را بدست آورید:

مکمل 10 عدد 256

$$\begin{array}{r} 1000 \\ - 256 \\ \hline 744 \end{array}$$

مثال:

مکمل 9 عدد 657 را بدست آورید:

مکمل 9 عدد 657

$$\begin{array}{r} 999 \\ - 657 \\ \hline 342 \end{array}$$

مثال:

مکمل 8 عدد 561 را بدست آورید.

مکمل 8 عدد 561

$$\begin{array}{r} 1000 \\ - 561 \\ \hline 217 \end{array}$$

مثال:

مکمل 2 و 1 اعداد زیر را بدست آورید:

مکمل 10

مکمل 9

مکمل 10

مکمل 9

$$(234)_{10} = 10^3 - 234 = 766$$

$$10^3 - 1 - 234 = 765$$

$$(234.68)_{10} = 10^3 - 234.68 = 765.32$$

$$10^3 - 10^{-2} - 234.68 = 1000 - 0.01 - 234.68 = 765.31$$

مثال: مکمل 16 و 15 عدد 3EA ف را در مبنای 16 بدست آورید.

* ساده‌تر است که ابتدا مکمل (1-2) را بدست آوریم سپس آن را با یک "1" جمع کنیم.

FFFF

- 3EA F

مکمل 15 → 15

مکمل 16 = 1 + مکمل 15

C150

+ 1

مکمل 16 → 16

بدست آوردن مکمل 2 در مبنای 2

از سمت راست شروع می‌کنیم تا زمانی که به اولین یک برسیم تمام اعداد را بدون تغییر می‌نویسیم بعد از اولین 1 به هر عددی که رسیدیم عکس آن را می‌نویسیم اگر صفر بود یک و اگر یک بود صفر.

مثال:

(1001010010)₂

0110101110

مکمل 2:

بدست آوردن مکمل ۱ در مبنای ۲

برای به دست آوردن مکمل ۱ عدد اولیه را عیناً عکس می‌کنیم. یعنی یک به صفر و صفر به یک تبدیل می‌شود.

011010110

مکمل ۱

$$(011001110)_2 = 100110001$$

مکمل ۱

راه دیگر بدست آوردن مکمل ۲ در مبنای ۲

ابتدا مکمل یک عدد را بدست می‌آوریم سپس به سمت راست‌ترین بیت، عدد یک را اضافه می‌کنیم.

$$+1 \text{ مکمل } 1 = \text{مکمل } 2$$

مثال:

مکمل ۲ عدد $(011010110)_2$ را بدست آورید.

$$\begin{array}{r} \text{مکمل } 1 \\ 100101001 \\ + \quad \quad \quad 1 \\ \hline \text{مکمل } 2 \end{array}$$

توجه شود که اگر عدد اعشاری بود یک به سمت راست‌ترین بیت اضافه می‌شود.

مثال:

مکمل ۲ عدد $(10011/101)_2$ را بدست آورید:

$$(10011/101)_2 \rightarrow \text{مکمل } 1 \rightarrow (01100/010)$$

$$01100/010 \\ + \quad \quad \quad 1 \\ \hline 01100/011$$

تفريق به کمک مکمل گيري:

دو روش برای اين عمليات وجود دارد که عبارتند از:

۱- استفاده از مکمل ۲ :

$$(\text{مکمل } 2 \text{ عدد } N) + M - M = M + (\text{مکمل } 2 \text{ عدد } N) : \text{رابطه (۱-۴)}$$

با استفاده از رابطه (۱-۴) می‌توانیم حاصل تفريقي $M-N$ را بدست آوريم حاصل بدست آمده دارای دو حالت می‌باشد:

الف: زمانی که حاصل دارای Carry باشد که در اين حالت تنها Carry را حذف کرده عدد باقیمانده حاصل مورد نظر ما می‌باشد.

مثال:

حاصل تفريقي زير را بدست آوريد:

$$\begin{array}{r} 4897 \rightarrow M \\ - 1439 \rightarrow N \\ \hline ? \end{array}$$

$$\begin{array}{r} M \\ + [N]_r \\ \hline ? \end{array}$$

بنابراین از N مکمل ۲ می‌گیریم \Rightarrow

$$\begin{array}{r} 4897 \\ + 8561 \\ \hline \text{Carry } 13458 \end{array}$$

مدار منطقی

ب: زمانی که حاصل بدست آمده فاقد Carry باشد، از آن مکمل r می‌گیریم. سپس یک منفی جلویش می‌گذاریم. عدد بدست آمده حاصل مورد نظر می‌باشد.

$$\begin{array}{r} 1439 \\ - 4897 \Rightarrow ? \\ \hline \end{array} \quad \begin{array}{l} M \\ + [N]_{r-1} \\ \hline ? \end{array} \quad \text{بنابراین باید از } N \text{ مکمل } r \text{ بگیریم} \Rightarrow \begin{array}{r} 1439 \\ + 5103 \\ \hline 6542 \end{array}$$

\rightarrow مکمل r می‌گیریم \rightarrow ۳۴۵۸

۲- استفاده از مکمل $r-1$: رابطه (۱-۵)

$$M-N = M + (N - \text{عدد } r-1)$$

با استفاده از رابطه (۱-۵) می‌توانیم حاصل تفریق $M-N$ را بدست آوریم، حاصل بدست آمده دارای دو حالت می‌باشد.

الف: زمانی که حاصل دارای carry باشد، carry را حذف می‌کنیم و حاصل را با عدد یک جمع می‌کنیم، عدد بدست آمده حاصل مورد نظر می‌باشد.

مثال:

حاصل تفریق زیر را بدست آورید:

$$\begin{array}{r} 4897 \\ - 1439 \\\hline ? \end{array} \quad \begin{array}{l} M \\ + [N]_{r-1} \\ \hline ? \end{array} \quad \text{باید از } N \text{ مکمل } r-1 \text{ بگیریم} \Rightarrow \begin{array}{r} M \\ + [N]_{r-1} \\ \hline ? \end{array} \quad \text{داریم} \Rightarrow$$

$$\Rightarrow [1439]_{r-1} = 8560 \Rightarrow \begin{array}{r} 4897 \\ + 8560 \\ \hline \text{Carry} \rightarrow 3457 \end{array}$$

حال carry را حذف می‌کنیم و حاصل را با عدد "۱" جمع می‌کنیم:

$$\begin{array}{r} 3457 \\ + 1 \\ \hline 3458 \end{array} \quad \rightarrow M, N$$

ب: زمانی که حاصل بدست آمده فاقد carry باشد، از آن مکمل $r-1$ می‌گیریم، سپس یک منفی جلوی آن می‌گذاریم. عدد بدست آمده حاصل مورد نظر می‌باشد.

مثال:

حاصل تفریق را بدست آورید:

$$\begin{array}{r} 1439 \\ - 4897 \\\hline ? \end{array} \quad \begin{array}{l} M \\ + [N]_{r-1} \\ \hline ? \end{array} \quad \text{بنابراین باید از } N \text{ مکمل} \Rightarrow \begin{array}{r} M \\ + [N]_{r-1} \\ \hline ? \end{array} \quad \text{داریم} \Rightarrow$$

$r-1$ بگیریم.



پس باید از حاصل دوباره مکمل ۱-۲ بگیریم \rightarrow و یک منفی جلوی آن قرار می‌دهیم:

$$\Rightarrow [4897]_{r=1} = 3458 \\ = -3458 \text{ حاصل تفریق } M, N$$

کدهای عددی

خصوصیات کدگذاری اعداد:

وزن دار بودن: به این معنا که با استفاده از این وزنهای بتوانیم مقدار دهدۀ را بدست آوریم.

خود مکمل بودن: کدی خود مکمل است که مکمل یک آن در دو دویی با مکمل ۹ آن در دهدۀ برابر باشد اگر یک کد وزن دار بخواهد خود مکمل باشد باید جمع وزنهایش با هم ۹ شود.

خود مکمل نیست. $9 \neq 1+2+4+8+10+16+1$ مجموع وزنهای (101110)

اعکاسی بودن: کدهایی دارای خاصیت اعکاسی هستند که کد قبل و بعد تنها در یک بیت با هم تفاوت داشته باشند. مثلًا ۰۰۰ و ۰۰۱ و ... تنها در یک بیت اختلاف دارند.

000
001
101
100

۴ کد نمونه که دارای خاصیت اعکاسی هستند.

چند روش کدگذاری

کد (Binary Coded Decimal) :BCD

یک کد چهاریتی بصورت زیر است که می‌تواند اعداد ۰ تا ۹ را به وجود آورد این کد یک کد وزن دار می‌باشد که وزنهای آن ۱ ۲ ۴ ۸ می‌باشد. در اصل با توجه به مفهوم این کد می‌توان دریافت که در این کد هر رقم یک عدد دهدۀ باید به باینری در آید. برای بدست آوردن مکمل ۹ یک عدد از کد BCD، دو راه وجود دارد که عبارتند از:

راه اول: ابتدا از آن عدد مکمل یک گرفته سپس آن را با ۱۰۱۰ جمع می‌کیم.

0000
0001
⋮
1001

مثال:

مکمل ۹ عدد ۲ در کد BCD را بدست آورید:

$$\begin{array}{r} & 1101 \\ 2 & \rightarrow 0010 & \longrightarrow & + 1010 \\ & & & \hline \text{Carry} \rightarrow 1 & 0111 \end{array}$$

carry را حذف می‌کیم جواب حاصل مکمل ۹ عدد ۲ در BCD است.

راه دوم: عدد مورد نظر را با (0110) جمع می‌کیم یعنی در اصل عدد مورد نظر را با ۶ جمع می‌کیم از عدد حاصل مکمل یک می‌گیریم. جواب حاصل مکمل ۹ عدد مورد نظر است. اگر حاصل را با یک جمع کیم به مکمل ۱۰ میرسیم.

مدار منطقی

مثال:

مکمل ۹ و ۱۰ عدد یک را در کد BCD بدست آورید:

0001

$$\begin{array}{r} \text{مکمل } 9 = 1000 \\ + 0110 \\ \hline 0111 \end{array}$$

$$0111 = 1000 + 1 = 1001 \quad \text{مکمل } 10 = 1000 + 1 = 1001$$

کد Excess

کد افزونی به صورت k -Excess است یعنی k واحد به عدد باینری اضافه می‌کند. مهمترین ویژگی این کدها این است که وزن دار نیستند. متداول‌ترین کد افزونی Excess-3 می‌باشد. برای بدست آوردن یک کد Excess-3 باید عدد باینری را با ۳ جمع کنیم و همینطور برای تبدیل کد Excess-3 به باینری باید ۳ واحد از آن کم کنیم.

از دیگر خصوصیات این کد خود مکمل بودن آن می‌باشد. یعنی مکمل یک آن در دو دویی با مکمل ۹ آن در دهدهی برابر می‌باشد.

Binary	Excess
0000	0011
0001	0100
0010	0101
0011	0110
0100	0111

مثال:

یک Excess-3 برای بردن آن به باینری باید ۳ واحد از آن را کم کنیم این عدد برابر ۱۱ در مبنای ده است و ۳ واحد که از آن کم شود برابر ۸ می‌شود. یعنی عدد ۱۰۰۰ در باینری.

1011

$$\begin{array}{r} + 0011 \\ \hline 1000 \end{array}$$

کد Gray

مهمترین خاصیت این کد انعکاسی بودن آن است مثلاً در دو یتی آن داریم:

$$\left. \begin{array}{l} 00 \rightarrow 0 \\ 01 \rightarrow 1 \\ 11 \rightarrow 2 \\ 10 \rightarrow 3 \end{array} \right\} \Rightarrow \text{این اعداد مربوط به کد Gray است.}$$

همانطور که ملاحظه می‌شود هر کد با کد بعدی تنها در یک یتی متفاوت است. این کد را می‌توان به کمک آینه ایجاد کرد. فرض کنید می‌خواهیم از عدد صفر تا ۷ را در کد Gray ایجاد کنیم. برای ایجاد آن سه یتی نیاز می‌باشد. بنابراین به دو آینه نیاز داریم. (توجه داریم که اگر بخواهیم یک کد n یتی Gray ایجاد کنیم باید $2^n - 1$ آینه استفاده کنیم.)

مداد هفتاد

در شروع فقط می‌دانیم که صفر Gray همان "0" و یک Gray همان "1" می‌باشد. حال با قرار دادن یک آینه زیر این دو رقم انعکاس زیر را خواهیم داشت:

دهدهی	Gray
0	0
1	1
2	0
3	1

پس از تولید انعکاس باید این قانون را در نظر بگیریم که سمت چپ هر عدد بالای آینه یک بیت صفر و پائین آینه یک بیت یک قرار می‌دهیم. بنابرایم خواهیم داشت:

دهدهی	Gray
0	0 0
1	0 1
2	1 1
3	1 0

حال برای ایجاد چهار کد بعدی باید یک بار دیگر از آینه استفاده کنیم بنابراین:

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

حال طبق همان قانون قبلی برای اعداد بالای آینه یک بیت صفر سمت چپ و برای اعداد پائین آینه یک بیت یک سمت چپ قرار می‌دهیم بنابراین در انتهای خواهیم داشت:

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

اما همان‌طور که مشاهده می‌شود اگر بخواهیم کدهای بزرگتری را در Gray ایجاد کنیم کار بسیار دشواری را پیش رو خواهیم داشت بنابراین بهتر است از روش‌های زیر برای این کار استفاده کرد.

مدارهای خطی

تبدیل عدد دودویی به معادل Gray

برای تبدیل عدد دو دویی به معادل Gray می‌توان از رابطه (۱-۵) استفاده کرد. همانطور که ملاحظه می‌شود پر از زش ترین بیت در هر دو کد یکی است ($b_n = g_n$) برای بدست آوردن کدهای بعد هم همانطور که می‌بینیم باید هر بیت را با بیت کناری آن X-OR کنیم تا حاصل یعنی معادل Gray تولید شود.

$$(b_n b_{n-1} \dots b_1)_2 = (g_n g_{n-1} \dots g_1)_G$$

$$\begin{cases} g_n = b_n \\ g_k = b_k \oplus b_{k+1} \quad k=1, \dots, n-1 \end{cases} \quad \text{رابطه (۱-۵)}$$

تبدیل کد Gray به معادل دودویی:

$$\begin{cases} g_n = b_n \\ b_k = \sum_{i=k}^n g_i \pmod{2} \end{cases} \quad \text{رابطه (۱-۶)}$$

برای تبدیل کد Gray به معادل دودویی باید از رابطه (۱-۶) استفاده کرد. اما به بیان دیگر می‌توان به این شکل عمل کرد که سمت چپ ترین رقم که در هر دو کد یکی می‌باشد، برای بدست آوردن بیت بعدی باید دو بیت آخر کد Gray را با هم \oplus X-OR کنیم تا بیت دوم بدست آید و سپس حاصل را بایت سوم کد Gray می‌کنیم تا بیت سوم بدست آید و به همین شکل تا آخرین بیت.

مثال:

عدد $G(1001)$ را به معادل بازنی تبدیل کنید:

	کد Gray	کد دهدهی	
00 → 0	0000	0	$(1 \quad 0 \quad 0 \quad 1)_G$
	0001	1	$\downarrow \quad \downarrow \quad \downarrow$
	0011	2	$\oplus \quad \oplus \quad \oplus$
	0010	3	$\nearrow \quad \nearrow \quad \nearrow \quad \nearrow \quad \downarrow$
01 → 0	0110	4	$b_4 \quad b_3 \quad b_2 \quad b_1$
	0111	5	$(1 \quad 1 \quad 1 \quad 0)_2$
	0101	6	
	0100	7	
11 → 2	1100	8	
	1101	9	
	1111	10	
	1110	11	
10 → 3	1010	12	
	1011	13	
	1001	14	
	1000	15	

مثال:

 عدد $(1010)_2$ را به معادل Gray تبدیل کنید.

$$\begin{array}{cccc}
 b_4 & b_3 & b_2 & b_1 \\
 (1 & 0 & 1 & 0)_2 \\
 \downarrow & \downarrow & \downarrow & \downarrow \\
 \oplus & \oplus & \oplus & \\
 \downarrow & \downarrow & \downarrow & \\
 (1 & 1 & 1 & 1)_G \\
 g_4 & g_3 & g_2 & g_1
 \end{array}$$

تبدیل مبنای ۱۰ به کد BCD:

هر رقم مبنای ۱۰ معادل با یک عدد چهار بیتی در کد BCD می‌باشد. حال اگر بخواهیم یک عدد مبنای ۱۰ را که از چندین رقم تشکیل شده است را به کد BCD تبدیل کنیم باید تک تک ارقام را به حالت باینری چهار بیتی بنویسیم. یعنی با استفاده از ۱ وزن ۸۴۲۱ آن رقم را به باینری تبدیل کنیم.

مثال:

عدد ۳۴۵ در مبنای ۱۰ را به معادل BCD تبدیل کنید.

$$\begin{array}{ccc}
 3 & 4 & 5 \\
 \swarrow & \downarrow & \searrow \\
 0011 & 0100 & 0101
 \end{array} \Rightarrow (001101000101)_{BCD}$$

تبدیل کد BCD به مبنای ۱۰:

برای این کار کافیست از سمت راست دسته‌های چهار بیتی را تشکیل دهیم و بالای هر دسته وزن ۸۴۲۱ را قرار دهیم و وزن هر دسته را بدست آوریم یعنی هر یک دسته را در وزن متناظرش ضرب کنیم سپس اعداد یک دسته را با هم جمع کنیم تا حاصل یعنی معادل دهدهی بدست آید. در ضمن باید توجه داشت که اگر آخرین دسته کمتر از چهار بیت در خود داشت باید سمت چپ آن صفر قرار دهیم تا دسته به صورت کامل درآید.

مثال:

 عدد $(1101111001)_{BCD}$ را به معادل دهدهی تبدیل کنید.

$$\begin{array}{ccc}
 0011 & 0111 & 1001 \\
 8421 & 8421 & 8421 \\
 0+0+2+1 & 0+4+2+1 & 8+0+0+1 \\
 \downarrow & \downarrow & \downarrow \\
 3 & 7 & 9
 \end{array} \Rightarrow (379)_{10}$$

تبدیل مبنای ۱۰ به کد Excess-3:

برای انجام این تبدیل باید هر رقم را به معادل Ex-3 تبدیل کنیم یعنی اینکه ابتدا به کمک وزن اعداد هر رقم مبنای ۱۰ را به معادل باینری تبدیل می‌کنیم سپس ۱۱۰۰ را (یعنی ۳) به معادل باینری اضافه می‌کنیم تا معادل Ex-3 بدست آید. باید توجه داشت که در این نوع تبدیل هم باید از وزن ۸۴۲۱ استفاده شود.

کد منطقی

مثال:

عدد ۳۵۷ در مبنای ۱۰ را به معادل Ex-3 تبدیل کنید.

$$\begin{array}{r}
 3 \quad 5 \quad 7 \\
 \swarrow \quad \downarrow \quad \searrow \\
 0011 \quad 0101 \quad 0111 \\
 + \quad \quad \quad \leftarrow \\
 0011 \quad 0011 \quad 0011 \\
 \hline
 0110 \quad 1000 \quad 1010
 \end{array}
 \quad
 \begin{array}{l}
 \text{مرحله اول تبدیل هر رقم به معادل باینری} \\
 \text{مرحله دوم جمع با } 0011 \\
 \Rightarrow (011010001010)_{\text{Ex-3}}
 \end{array}$$

تبدیل کد Ex-3 به مبنای ۱۰:

برای این کار ابتدا از سمت راست شروع به دسته‌بندی چهار بیتی می‌کیم، با استفاده از وزن ۱ ۴ ۲ ۱ برای هر دسته مقدار وزن آن را پیدا می‌کیم سپس عدد بدست آمده را منهای ۳ می‌کنیم تا حاصل در مبنای ۱۰ بدست آید. باید توجه داشت که اگر آخرین دسته کمتر از چهار بیت بود سمت چپ آن به تعداد بیتها کمتر صفر قرار می‌دهیم تا دسته کامل شود.

مثال:

عدد $(10110011100)_{\text{Ex-3}}$ را به معادل دهدی تبدیل کنید.

$$\begin{array}{ccccccc}
 0101 & 1001 & 1100 \\
 8421 & 8421 & 8421 \\
 0+4+0+1 & 8+0+0+1 & 8+4+0+0 \\
 \downarrow & \downarrow & \downarrow \\
 5 & 9 & 12 \\
 -3 & 3 & 3 \\
 \hline
 2 & 6 & 9 \Rightarrow (269)_{10}
 \end{array}
 \quad
 \begin{array}{l}
 \text{مرحله اول بدست آوردن وزن هر دسته با کمک } 8421 \\
 \text{مرحله دوم تفريق حاصل از 3 و بدست آوردن معادل دهدی}
 \end{array}$$

سایر کدهای وزن دار:

۱ ۲ ۴ ۸ معمولترین وزنهای کد BCD را تشکیل می‌دهد ولی وزنهای دیگری مانند ۱ ۲ ۴ ۲ ۱ ، ۲ ۴ ۲ ۱ ، ۴ ۱ ۲ ۱ ، ... وجود دارد. این وزنهای در واقع قراردادی است و قانون خاصی ندارد. برخی از این کدها از جمله ۱ ۲ ۴ ۸ کدی می‌باشد که قابلیت ساختن اعداد منفی را نیز دارد. علامت '-' روی برخی از وزنهای به معنی منفی بودن آن وزن می‌باشد در اصل در صورت استفاده از آن وزن مقدارش از وزن کل کم خواهد شد.

مثال:

عدد ۱۱۱۱ در کد ۱ ۲ ۴ ۸ چند است؟

$$1111 = (1 \times 1) + (1 \times 2) + (1 \times (-4)) + (1 \times 8) = 1 + 2 + (-4) + 8 = 7$$

مثال:

عدد ۲۲ را با کد ۱ ۲ ۴ ۸ ۴ ۲ ۱ ۶ بنویسید؟

در عدد ۲۲ یک ۱۶ وجود دارد پس زیر وزن ۱۶ عدد یک می‌گذاریم سپس مقدار ۱۶ را از ۲۲ کم می‌کنیم.

۲۲ - ۱۶ = ۶
در عدد ۶، مقدار ۸ وجود ندارد پس زیر وزن ۸ عدد صفر می‌گذاریم. اما در ۶ یک ۴ وجود دارد. زیر مقدار ۴ عدد یک قرار می‌دهیم. بعد ۴ را

از ۶ کم می‌کنیم. ۶ - ۴ = ۲. در عدد ۲ یک ۲ وجود دارد پس زیر وزن ۲ هم یک قرار می‌دهیم و مقدار وزن ۱ نیز صفر خواهد شد.

$$\begin{array}{r}
 22 \\
 -16 \\
 \hline
 6 \\
 16 \ 8 \ 4 \ 2 \ 1 \quad -4 \\
 1 \ 0 \ 1 \ 1 \ 0 \quad \underline{-} \\
 \hline
 2 \\
 . \underline{-} 2 \\
 \hline
 0
 \end{array}$$

یکی از این کدها $\bar{1} \ 2 \ \bar{4} \ 8$ نی خواهیم بینیم که اعداد در این کد چگونه ساخته می‌شود. کوچکترین عددی که در این کد ساخته می‌شود عدد ۵ - می‌باشد که آن ۰۱۰۱ می‌باشد. یعنی:

$$(0 \times 8) + (0 \times (-4)) + (1 \times 2) + (0 \times (-1)) = 10$$

حال فرض کنید بخواهیم عدد ۵ را در این کد بسازیم به این صورت عمل می‌کنیم که:

به کمک اعداد مثبت نمی‌توان ۵ را ساخت زیرا عدد ۸ که بزرگتر از ۵ است و عدد ۲ که کوچکتر از ۵ می‌باشد. بنابراین باید با بازی با وزن‌ها عدد ۵ را بسازیم. ابتدا در زیر وزن ۸ عدد یک را قرار می‌دهیم در این حالت با فرض صفر بودن سایر وزن‌ها به عدد ۸ خواهیم رسید.

(یعنی $\bar{1} \ 2 \ \bar{4} \ 8$) بنابراین برای رسیدن به ۵ باید سه واحد از ۸ کم کنیم. و چون وزن ۳ - وجود ندارد مجبور هستیم از ۴ کمک

بگیریم. (یعنی $\bar{1} \ 2 \ \bar{4} \ 8$) در این صورت به عدد ۴ خواهیم رسید همان‌طور که ملاحظه می‌شود این عدد کوچکتر از عدد مورد نظر

ما یعنی ۵ می‌باشد پس باید به آن یک واحد اضافه کنیم اما می‌بینیم که وزن مثبت "۱" وجود ندارد و فقط برای اضافه کردن می‌توان از ۲ + استفاده کرد. با استفاده از این وزن حاصل برابر ۶ می‌شود ($\bar{1} \ 2 \ \bar{4} \ 8$) حال کافیست که تنها یک واحد از آن کم شود بنابراین که به

صورت $\bar{1} \ 2 \ \bar{4} \ 8$ در خواهد آمد.

مدار منطقی

حال با کمی تلاش می‌توان جدول زیر که مربوط به اعداد این کد می‌باشد را بدست آورد.

دهدهی	8	4	2	1
-5	0	1	0	1
-4	0	1	0	0
-3	0	1	1	1
-2	0	1	1	0
-1	0	0	0	1
0	0	0	0	0
1	0	0	1	1
2	0	0	1	0
3	1	1	0	1
4	1	1	0	0
5	1	1	1	1
6	1	1	1	0
7	1	0	0	1
8	1	0	0	0
9	1	0	1	1
10	1	0	1	0

مجموعه اعداد منفی و صفر تولید شده توسط کد ۸ ۴ ۲ ۱

مجموعه اعداد مثبت

کدهای تشخیص و تصحیح خطای

اولین موضوع مورد بحث فاصله بین دو کد است فاصله دو کد عبارتست از میزان اختلاف بین دو کد یعنی تعداد تفاوت در صفرها و یکها در جایگاههای یکسان مثلاً بین دو کد $I = 0010$ و $J = 1110$ فاصله چند است؟

$$0010 \rightarrow I$$

$$1110 \rightarrow J \Rightarrow d(I, J) = 2$$

زیرا اولین بیت از سمت راست ($j=0$) با $I=0$ و دومین بیت ($j=1$) با $I=1$ باهم برابرند. پس اختلاف وجود ندارد. اما در بیتهای سوم و چهارم مقدار j متفاوت است. پس این دو عدد (۲ کد) در ۲ بیت باهم اختلاف دارند.

مورد دیگر وزن یک کد است که با w نشان داده می‌شود مثلاً درمثال بالا (I) برابر است با تعداد یکهای موجود در آن کد:

$$w(I) = 2$$

مورد دیگر تعداد خطاهای قابل کشف است که با S نشان داده می‌شود.

T : تعداد خطاهای قابل تصحیح می‌باشد.

کد parity (توازن):

این کد، کد تشخیص خطاهاست parity دو نوع دارد: زوج و فرد.

پریتی زوج به این معناست که تعداد یکهای داده باید زوج باشد.

پریتی فرد به این معناست که تعداد یکهای داده باید فرد باشد.

پریتی مورد استفاده در بحث مورد نظر پریتی یک یستی است که به آن افقی پریتی گفته می‌شود.

در ضمن parity ماتریسی نیز وجود دارد که به سطر و ستون parity اضافه می‌کند.

فرض کنید اطلاعات پنج یتی 10110 فرستاده می‌شود، اگر با پریتی فرد فرستاده شود چون تعداد یکها فرد است به آخر آن صفر اضافه می‌شود که درست باقی بماند ولی اگر با پریتی زوج فرستاده شود چون باید تعداد یکهایش زوج باشد به پیت آخر یک، یک اضافه می‌شود تا تعداد یکها زوج شود.

$$10110 \begin{cases} 0 & \rightarrow \text{فرد} \\ 1 & \rightarrow \text{زوج} \end{cases}$$

یکی دیگر از کدهایی که برای عملیات تشخیص و تصحیح خطا به کار برده می‌شود که همینگ می‌باشد نحوه کدگذاری در این روش کمی متفاوت با کد Parity است. حال در این قسمت به بررسی یک داده 4 یتی می‌پردازیم که روی آن سه پیت کد همینگ نیز اضافه می‌شود.

کد همینگ

اگر یک کد ارسالی در همینگ به صورت $C_1C_2b_3C_4b_5b_6b_7$ باشد داریم:

اطلاعات اصلی $\rightarrow b_3b_5b_6b_7$

$C_1C_2C_4$ های اضافه شده در کد همینگ Parity

$$C_1 = b_3 \oplus b_5 \oplus b_7$$

$$C_2 = b_3 \oplus b_6 \oplus b_7$$

$$C_4 = b_5 \oplus b_6 \oplus b_7$$

یک کد همینگ با فاصله d قابلیت کشف S خطا ($S = d - 1$) و تصحیح T خطا را دارد.

مثال:

کد همینگی را در نظر بگیرید که حداقل فاصله در آن 6 باشد تعداد خطای قابل تشخیص و تصحیح را بدست آورید.

$$d = 6$$

$$S = d - 1 = 6 - 1 = 5$$

$$T = \left\lceil \frac{d-1}{2} \right\rceil = \left\lceil \frac{6-1}{2} \right\rceil = 2$$

مدار منطقی

مثال: اگر فرض کنیم بخواهیم داده 1011 را توسط همینگ کد کنیم به چه داده‌ای خواهیم رسید؟
حل:

$$\left. \begin{array}{l} b_3 = 1 \\ b_5 = 0 \\ b_6 = 1 \\ b_7 = 1 \end{array} \right\} \text{همان‌طور که ملاحظه می‌شود} \\ \text{می‌باشد. بنابراین داریم:}$$

$$C_1 = b_3 \oplus b_5 \oplus b_7 = 1 \oplus 0 \oplus 1 = 0$$

$$C_2 = b_3 \oplus b_6 \oplus b_7 = 1 \oplus 1 \oplus 1 = 1$$

$$C_4 = b_5 \oplus b_6 \oplus b_7 = 0 \oplus 1 \oplus 1 = 0$$

بنابراین کد ایجاد شده به صورت زیر در می‌آید:

011011

کدهای Ascii

برای نشان دادن کاراکترهای موجود در یک سیستم کامپیوتر می‌توان از یک نوع کدگذاری استفاده کرد که یک عدد را به آن کاراکتر تخصیص دهد به این نوع کدها، کدهای ASCII گفته می‌شود که توسط استانداردهای مختلف نمایش داده شده است.

به عنوان مثال برای نمایش A از عدد 1000001 استفاده می‌شود که معادل 65 می‌باشد. یا برای کاراکتر a از عدد 1100001 می‌توان استفاده کرد.

برای نمایش برخی کاراکترهای خاص نیز می‌توان از کدهای ASCII استفاده کرد، مانند % (0100101) یا \$ (0100010).
نکته:

اگر به یک کد پریتی زوج اضافه کنیم حداقل اختلاف 2 بیت می‌باشد ($d=2$) که می‌توان یک خطأ را تشخیص داد ولی نمی‌توان هیچ خطایی را تصحیح کرد.
نکته:

مزیت کد وزن‌دار 1 2 4 2 این است که عمل جمع هر دو عدد در این کد با مکمل گیری بیت به بیت انجام می‌شود این کد یک کد خود مکمل است. چون

$$2+4+2+1=9$$

نکته:

در جمع دو عدد باینری n بیتی که حاصل n بیتی داشته باشد برای بدست آوردن مقدار overflow از رابطه (۱-۷) استفاده می‌کنیم.

$$x_7 x_6 \dots x_0$$

$$\underline{y_7 y_6 \dots y_0}$$

$$p_7 p_6 \dots p_0$$

$$\text{over flow} \rightarrow V = x_7 y_7 \bar{p}_7 + \bar{x}_7 \bar{y}_7 p_7$$

رابطه (۱-۷)



مجموعه مسائل فصل اول:

۱- تغییرات مبنای زیر را حل کنید:

(الف)

$$(273)_{10} = (100010001)_2$$

$$\begin{array}{r} 273 \quad |2 \\ - 136 \quad |2 \\ \hline 1 \quad \quad 68 \quad |2 \\ \hline 0 \quad \quad 34 \quad |2 \\ \hline 0 \quad \quad 17 \quad |2 \\ \hline 0 \quad \quad 8 \quad |2 \\ \hline 1 \quad \quad 4 \quad |2 \\ \hline 0 \quad \quad 2 \quad |2 \\ \hline 0 \quad \quad 1 \\ \hline 0 \end{array}$$

$$\begin{array}{r} 273 \\ 2^8 \quad 2^7 \quad 2^6 \quad 2^5 \quad 2^4 \quad 2^3 \quad 2^2 \quad 2^1 \quad 2^0 \\ 256 \quad 128 \quad 64 \quad 32 \quad 16 \quad 8 \quad 4 \quad 2 \quad 1 \\ \hline (1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 1)_2 \end{array}$$

273

- 256

17

- 16

1

(ب)

$$(3124)_{10} = (\quad)_2$$

$$\begin{array}{r} 2048 \quad 1024 \quad 512 \quad 256 \quad 128 \quad 64 \quad 32 \quad 16 \quad 8 \quad 4 \quad 2 \quad 1 \\ (1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0 \quad 0)_2 \end{array}$$

$$\begin{array}{r} 3124 \\ -2048 \\ \hline 1076 \\ -1024 \\ \hline 52 \\ -32 \\ \hline 20 \\ -16 \\ \hline 4 \\ -4 \\ \hline 0 \end{array}$$

$$(365)_{10} = (\quad)_8$$

$$\begin{array}{r} 365 \\ \underline{- 45} \quad |8 \\ \hline 5 \end{array}$$

$$8^2 \quad 8^1 \quad 8^0$$

$$64 \quad 8 \quad 1$$

$$(5 \quad 5 \quad 5)_8$$

(د)

$$64 \quad 32 \quad 16 \quad 8 \quad 4 \quad 2 \quad 1$$

$$(1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0)_2 = (\quad)_{10}$$

(ه)

$$8 \quad 1$$

$$(4 \quad 7)_8 = (\quad)_{10}$$

$$(7 \times 1) + (4 \times 8) = (39)_{10}$$

(و)

$$256 \quad 16 \quad 1$$

$$(1 \quad A \quad F)_{16} = (431)_{10}$$

$$\downarrow \quad \downarrow$$

$$10 \quad 15$$

(ي)

$$(257)_8 = (\quad)_2$$

$$(257)_8 \rightarrow \begin{cases} 7 \rightarrow 111 \\ 5 \rightarrow 101 \rightarrow (10101111)_2 \\ 2 \rightarrow 010 \end{cases}$$

(ن)

$$(100111011101)_2 = (4735)_8$$

۲- مکمل ۱ و ۰- اعداد زیر را بدست آورید.

$$(25)_{10}$$

$$\text{مکمل } 10$$

$$\begin{array}{r} 100 \\ -25 \\ \hline 75 \end{array}$$

$$\text{مکمل } 9$$

$$\begin{array}{r} 99 \\ -25 \\ \hline 74 \end{array}$$

$$(347)_8$$

$$\text{مکمل } 8$$

$$\begin{array}{r} 1000 \\ -347 \\ \hline 431 \end{array}$$

$$\text{مکمل } 7$$

$$\begin{array}{r} 777 \\ -347 \\ \hline 430 \end{array}$$

$$(100100)_2$$

$$\text{مکمل } 2 (011100)$$

$$\text{مکمل } 1 (011011)$$

۳- تبدیل کدهای زیر را انجام دهید.

$$(011)_2 = (110)_{EX-3}$$

$$\begin{array}{r} 011 \rightarrow 3 \\ + \frac{3}{6} \rightarrow 110 \end{array}$$

$$(11000101)_B = (10100111)_G$$

$$(10100111)_G = (11000101)_B$$

۴- جمع و تفریق و ضرب های زیر را انجام دهید.

(الف)

$$\begin{array}{r} 1 \ 1 \\ A \ 2 \ F \ 4 \\ + 5 \ 0 \ D \ E \\ \hline F \ 3 \ D \ 2 \end{array} \quad \text{مبنا 16}$$

$$\begin{array}{r} 15 \ 20 \\ 9 \ 16 \cancel{\alpha} \ 20 \\ \cancel{\alpha} \ \emptyset \ \cancel{\beta} \ \cancel{\alpha} \\ - 5 \ 2 \ D \ E \\ \hline 4 \ D \ 7 \ 6 \end{array} \quad \text{مبنا 16}$$

$$\begin{array}{r} 4 \ 4 \\ 9 \ 8 \\ 2 \ F \ E \\ \times 5 \ 0 \ A \\ \hline 1 \ D \ E \ C \\ + E \ B \ 6 \ 0 \ 0 \\ \hline E \ D \ 3 \ E \ C \end{array} \quad \text{مبنا 16}$$

$$\begin{array}{r} 2 \ 1 \ 1 \\ 1 \ 1 \ 0 \ 1 \\ + 1 \ 1 \ 1 \ 0 \\ + 0 \ 1 \ 1 \ 1 \\ \hline 1 \ 0 \ 0 \ 0 \ 1 \ 0 \end{array}$$

(ب)

توضیح:

اگر 4 عدد "1" در مبنای دو با هم جمع شوند از حاصل باید 2 بار مبنای 2 کم شود و 2 به عنوان Carry مرحله بعد تولید شود، پس خواهیم داشت:

اما چون در مبنای 2 رقم "2" وجود ندارد باید به جای آن معادلش یعنی 10 را قرار دهیم.

$$\begin{array}{r} 2 \\ + 1 \\ + 1 \\ + 1 \\ + 1 \\ \hline 100 \end{array}$$

مجموعه تست‌های فصل اول:

۱ - کدام گزینه غلط است؟

(۱) عدد 308 در مبنای شانزده برابر است با 788 در مبنای ده

(۲) عدد 10111001 در مبنای دو برابر است با B9 در مبنای شانزده

(۳) عدد 377 در مبنای هشت برابر است با 111111 در مبنای دو

(۴) عدد 242 در مبنای ده برابر است با 242 به مرز (کد) BCD

۲ - معادل عدد 19 - در مبنای ده، به صورت مکمل دو (Two's complement) کدام است؟

101110 (۱)

011110 (۲)

101101 (۳)

110101 (۴)

۳ - معادل اکتاال عدد $_{16}(A0F)$ کدام است؟

5117 (۱)

61171 (۲)

5017 (۳)

6017 (۴)

۴ - عدد 10010011 به صورت BCD یافته شده است. معادل این دو کدام است؟

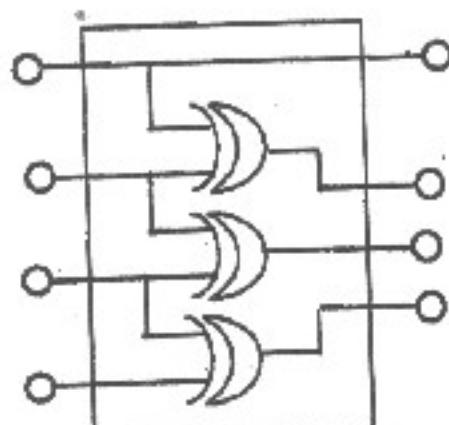
10100010 (۱)

01011101 (۲)

11101100 (۳)

01101100 (۴)

۵ - شکل زیر کدام مبدل است؟



(۱) کد سه افزون به کد باینری

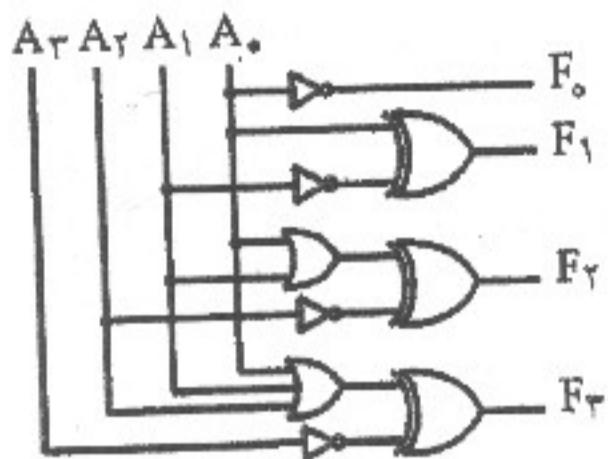
(۲) کد باینری به گری

(۳) کد گری به باینری

(۴) کد باینری به کد سه افزون



۹- با توجه به مدار زیر و به فرض این که اعداد a و f در سیستم مکمل دو باشند رابطه بین این دو عدد چهار یتی چیست؟ ($f = F_3F_2F_1F_0$, $a = A_3A_2A_1A_0$)



(۱) مکمل a است.

(۲) برابر با منفی a است.

(۳) برابر $a + 1$ است.

(۴) برابر $a - 1$ است.

فصل دوم

جبر بول، ساده‌سازی

و اصول اولیه

اصول اولیه و قوانین جبر بول:

	اپراتور جمع +	اپراتور ضرب ×	شماره اصل
شبه جذب	$a \cdot (\bar{a} + b) = a \cdot b$	$a + \bar{a}b = a + b$	1
عضو خنثی	$a + 0 = a$	$a \cdot 1 = a$	2
جابه جایی	$a + b = b + a$	$a \cdot b = b \cdot a$	3
پخشی	$a + (b \cdot c) = (a + b) \cdot (a + c)$	$a \cdot (b + c) = (a \cdot b) + (a \cdot c)$	4
عضو وارون	$a + \bar{a} = 1$	$a \cdot \bar{a} = 0$	5
خود توانی	$a + a = a$	$a \cdot a = a$	6
جذب	$a + ab = a$	$\overline{a + b} = \bar{a} \cdot \bar{b}$	7
دمورگان	$\overline{a + b} = \bar{a} \cdot \bar{b}$	$\overline{ab} = \bar{a} + \bar{b}$	8

مجموع حاصلضربها:

sop : sum of products

ضرب حاصل جمعها:

pos : product of sums

انواع جملات در جبر بول

به عنوان مثال تابع f_1 تشکیل شده از sopها می باشد.

$$f_1(A, B, C) = A'B + B'C + AC' + ABC$$

تابع f_2 تشکیل شده از posها می باشد.

$$f_2(A, B, C) = (A' + B)(B' + C)(A' + B' + C')$$

فرمتهای استاندارد:

برای هر تابع n متغیره، اگر عبارت ضرب شامل تمام n متغیر باشد و هر متغیر بصورت مکمل یا غیر مکمل (a, a') فقط یکبار در عبارت ظاهر شوند عبارت ضرب را minterm گویند. هر کدام از n متغیر استفاده شده در این میترم نشانگر یا یک بودن متغیر می باشد. به این صورت که اگر در یک میترم متغیری خودش ظاهر شود نشان دهنده یک بودن آن متغیر است و اگر مکملش ظاهر شود نشانه صفر بودن آن می باشد.

$$F(A, B, C) = \Sigma(0, 1, 5, 7)$$

شماره صفر minterm

$$0 \rightarrow 000 \rightarrow A'B'C'$$

شماره یک minterm

$$1 \rightarrow 001 \rightarrow A'B'C$$

شماره ۵ minterm

$$5 \rightarrow 101 \rightarrow AB'C$$

شماره ۷ minterm

$$7 \rightarrow 111 \rightarrow ABC$$

$$F(A, B, C) = A'B'C' + A'B'C + AB'C + ABC$$

اگر n متغیر داشته باشیم 2^n minterm داریم.

برای هر تابع n متغیره اگر عبارت جمع شامل تمام n متغیر باشد و هر متغیر بصورت مکمل یا غیر مکمل (a, a') فقط یکبار در عبارت ظاهر شوند عبارت جمع را maxterm گویند برای یک تابع n متغیر 2^n تا maxterm داریم.

$$F(A, B, C) = \prod(2, 3, 4, 6)$$

$$2 \rightarrow 010 \rightarrow A + B' + C$$

$$3 \rightarrow 011 \rightarrow A + B' + C'$$

$$4 \rightarrow 100 \rightarrow A' + B + C$$

$$6 \rightarrow 110 \rightarrow A' + B' + C$$

$$F(A, B, C) = (A + B' + C) \times (A + B' + C') \times (A' + B + C) \times (A' + B' + C)$$

در حالت کلی بدون در نظر گرفتن خروجی، 2^n یعنی 8 تا minterm و 8 تا maxterm داریم:

$$\underbrace{A'B'C' + A'B'C + \dots + ABC}_{8 \text{ تا}}$$

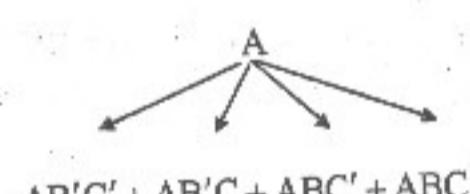
نکته: اگر تعداد یکها و صفرهای خروجی برابر بود minterm را می‌نویسیم اگر تعداد صفرها کمتر بود maxterm را می‌نویسیم اگر تعداد یکها کمتر بود minterm را می‌نویسیم.

تبديل SOP و POS به فرم‌های استاندارد (Maxterm و Minterm):

اگر بخواهیم در یک تابع (f) که از n متغیر تشکیل شده است یک sop یا یک pos که شامل m متغیر می‌باشد ($m < n$) را به فرم استاندارد تبدیل کنیم، باید تمامی ترکیبات مختلف ($n-m$) متغیر را که در آن sop یا pos کم دارد را اضافه کنیم. تفاوت sop و pos با maxterm و minterm در این است که در maxterm و minterm باید در هر جمله تمام متغیرها ظاهر شوند ولی در sop و pos این گونه نیست). یعنی یک sop یا pos تبدیل به 2^{n-m} تا maxterm یا minterm خواهد شد.

فرض می‌کنیم در تابع f که دارای سه متغیر A, B و C می‌باشد، A یک SOP و B, C یک Minterm نیست زیرا B و C را ندارد. حال برای اینکه بخواهیم آن را به فرم استانداردد آوریم داریم:

ترکیبات مختلف از $n-m$		BC
متغیر باقی مانده	0	$B'C'$
	0	$B'C$
	1	BC'
	1	BC



اگر در تبدیل آن به فرم استاندارد حالت‌های شبیه به هم دیده شده باید یکی را حذف کرد.

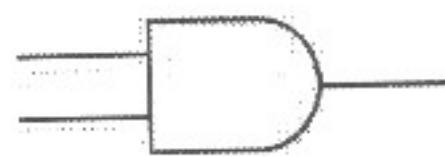
نکره: ترتیب قرار گرفتن متغیرها باید حفظ شود یعنی اگر در تابع $f(A, B, C)$ sop یا pos را تبدیل به حالت استاندارد کنیم و متغیرهای D و B را کم داشت پس از تولید maxterm یا minterm دقت می‌کنیم که ترتیب D و C و B و A حفظ شود.

گیت‌های منطقی:

: خروجی یک AND تنها در زمانی یک می‌باشد که هر دو ورودی مقدار یک داشته باشد.

A	B	AND
0	0	0
0	1	0
1	0	0
1	1	1

$$f = A \cdot B$$



: خروجی یک OR تنها زمانی صفر است که هر دو ورودی صفر باشد.

A	B	OR
0	0	0
0	1	1
1	0	1
1	1	1

$$f = A + B$$



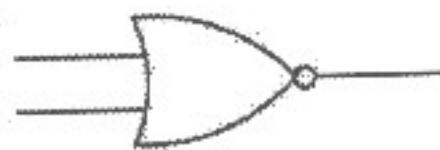
: خروجی یک NAND تنها زمانی صفر است که هر دو ورودی یک باشد.

A	B	NAND
0	0	1
0	1	1
1	0	1
1	1	0



$$f = A' + B'$$

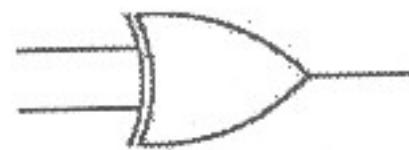
: خروجی یک NOR تنها زمانی یک است که هر دو ورودی صفر باشد.



A	B	NOR
0	0	1
0	1	0
1	0	0
1	1	0

$$f = A' \cdot B'$$

: خروجی XOR تنها زمانی یک است که ورودیها با یکدیگر متفاوت باشند.



A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

$$f = A \oplus B = A' \cdot B + A \cdot B'$$

: خروجی X-NOR زمانی یک است که ورودیها هر دو یکسان باشند.



A	B	X-NOR
0	0	1
0	1	0
1	0	0
1	1	1

$$f = A \odot B = A' \cdot B' + A \cdot B$$



Not ورودی را مکمل می کند. یعنی صفر را به یک و یک را به صفر تبدیل می کند.



A	NOT
0	1
1	0

$$f = A'$$

خروجی همان ورودی است. Buffer



A	Buffer
0	0
1	1

$$f = A$$

جدول درستی:

A	B	AND	OR	NAND	NOR	XOR	X-NOR	not A	Buffer A
0	0	0	0	1	1	0	1	1	0
0	1	0	1	1	0	1	0	1	0
1	0	0	1	1	0	1	0	0	1
1	1	1	1	0	0	0	1	0	1

منطق مثبت و منفی:

در منطق مثبت High به معنی یک است و Low به معنی صفر منطقی است.

در منطق منفی High به عنوان صفر منطقی است و Low به عنوان یک منطقی است.

به عنوان مثال جدول منطق مثبت و منفی گیت AND را با هم مقایسه می کنیم.

منطق مثبت \Rightarrow

A	B	AND
L	L	L
L	H	L
H	L	L
H	H	H

و \Rightarrow منطق منفی

A	B	AND
H	H	H
H	L	H
L	H	H
L	L	L

همانطور که ملاحظه می شود می توان دریافت که جدول AND در منطق منفی همان جدول AND در منطق مثبت است با این تفاوت که جای H و L با یکدیگر عوض شده است.

تجویه ایجاد جدول درستی:

جدول درستی جدولی است که تمامی حالات ممکن برای n متغیر موجود در تابع را تولید می کند. این جدول از دو قسمت ورودی و خروجی تشکیل شده است که قسمت ورودی تعداد ردیف های جدول را نیز مشخص می کند.

برای ایجاد یک جدول درستی برای یک تابع n متغیره باید 2^n ردیف به وجود آورد. هر کدام از این ردیف ها یک شماره مخصوص به خود را نیز دارند. شماره ردیف های جدول درستی از صفر شروع می شود و تا $1 - 2^n$ نیز ادامه پیدا می کند.

مدار منطقی

در اصل این شماره ردیف‌ها شماره‌های مربوط به maxterm و minterm را نیز مشخص می‌کندو در ضمن با استفاده از همین شماره ردیف‌ها می‌توان حالت‌های متفاوت برای n متغیر را ایجاد کرد. به عنوان مثال اگر یکتابع ۵ متغیره مشکل از متغیرهای A, B, C, D, E داشته باشیم و بخواهیم در ستون ورودی‌ها و ردیف ۵ مقدار بنویسیم داریم:

16 8 4 2 1

A B C D E

0 0 1 0 1

ردیف	a	minterm	maxterm
0	0	a'	a
1	1	a	a

جدول درستی یک متغیره

ورودی

ردیف	$\overbrace{a \ b}$	minterm	maxterm
0	0 0	$a'b'$	$a + b$
1	0 1	$a'b$	$a + b'$
2	1 0	ab'	$a' + b$
3	1 1	ab	$a' + b'$

جدول درستی ۲ متغیره

جدول درستی سه متغیره

ردیف	$\overbrace{a \ b \ c}$	minterm	maxterm
0	0 0 0	$a'b'c'$	$a + b + c$
1	0 0 1	$a'b'c$	$a + b + c'$
2	0 1 0	$a'bc'$	$a + b' + c$
3	0 1 1	$a'bc$	$a + b' + c'$
4	1 0 0	$ab'c'$	$a' + b + c$
5	1 0 1	$ab'c$	$a' + b + c'$
6	1 1 0	abc'	$a' + b' + c$
7	1 1 1	abc	$a' + b' + c'$

تکه: در جدول درستی در ستون خروجی آنها بیان کردند که مقدارشان یک است مشخص کننده تابع خروجی براساس minterm هاست و آنها یک مقدارشان صفر است مشخص کننده تابع خروجی براساس maxterm هاست.

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

$$f(A, B, C) = \Sigma(0, 1, 5, 7) = \Pi(2, 3, 4, 6)$$



مثال: جدول درستی طراحی کنید که مربوط به یک میز رأی گیری باشد، این میز رأی گیری ۳ نفره می‌باشد به این صورت که هر سه نفر قدرت رأی مساوی دارند. این میز دارای یک چراغ است و چراغ زمانی روشن می‌شود که حداقل دو نفر از رأی دهنده‌گان جواب مثبت داده باشند. (پذیرفته شدن طرح باعث روشن شدن چراغ و یک شدن خروجی می‌باشد).

رأی دهنده‌گان

ردیف	$\overbrace{a \ b \ c}$	چراغ f
0	0 0 0	0
1	0 0 1	0
2	0 1 0	0
3	0 1 1	1
4	1 0 0	0
5	1 0 1	1
6	1 1 0	1
7	1 1 1	1

$$\begin{cases} f(a, b, c) = \sum(3, 5, 6, 7) \\ \text{minterm} \quad \begin{cases} f(a, b, c) = a'b'c + ab'c + abc' + abc \end{cases} \end{cases}$$

یا

$$\begin{cases} f(a, b, c) = \prod(0, 1, 2, 4) \\ f(a, b, c) = (a + b + c) \times (a + b + c') \times (a + b' + c) \times (a' + b + c) \end{cases}$$

مثال: جدول درستی مربوط به یک میز رأی چهار نفره را که دارای دو چراغ سبز و آبی است را طراحی کنید. چراغ آبی زمانی روشن می‌شود که نفر دوم و سوم جواب مثبت داده باشند و چراغ سبز زمانی روشن می‌شود که حداقل سه نفر جواب مثبت داده باشند. تابع خروجی مربوط به چراغ سبز و آبی را به صورت minterm و maxterm بنویسید.

چراغ سبز چراغ آبی نفر چهارم نفر اول

ردیف	$\overbrace{a \ b \ c \ d}$	B	G
0	0 0 0 0	0	0
1	0 0 0 1	0	0
2	0 0 1 0	0	0
3	0 0 1 1	0	0
4	0 1 0 0	0	0
5	0 1 0 1	0	0
6	0 1 1 0	1	0
7	0 1 1 1	1	1
8	1 0 0 0	0	0
9	1 0 0 1	0	0
10	1 0 1 0	0	0
11	1 0 1 1	0	1
12	1 1 0 0	0	0
13	1 1 0 1	0	1
14	1 1 1 0	1	1
15	1 1 1 1	1	1

$$B(a, b, c, d) = \sum(6, 7, 14, 15) = a'bcd' + a'bcd + abcd' + abcd$$

یا

$$B(a, b, c, d) = \prod(0, 1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 13)$$

$$G(a, b, c, d) = \sum(7, 11, 13, 14, 15) = a'bcd + ab'cd + abc'd + abcd' + abcd$$

یا

$$G(a, b, c, d) = \prod(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12)$$

مثال: جدول درستی طراحی کنید یک عدد سه بیتی ورودی را به توان 2 برساند.

حل:

در این گونه مسائل باید ابتدا بزرگترین عدد ورودی را در نظر بگیریم و آن را به توان 2 برسانیم. در این مساله 7^2 برابر 49 می‌باشد حال باید 49 را به صورت بازنزی در آوریم تا بفهمیم که خروجی باید چند بیتی باشد.

ردیف				32	16	8	4	2	1
	x	y	z	a	b	c	d	e	f
0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	1
2	0	1	0	0	0	0	1	0	0
3	0	1	1	0	0	1	0	0	1
4	1	0	0	0	1	0	0	0	0
5	1	0	1	0	1	1	0	0	1
6	1	1	0	1	0	0	1	0	0
7	1	1	1	1	1	0	0	0	1

$$a(x, y, z) = \sum(6, 7) \text{ یا } \prod(0, 1, 2, 3, 4, 5)$$

$$b(x, y, z) = \sum(4, 5, 7) \text{ یا } \prod(0, 1, 2, 3, 6)$$

$$c(x, y, z) = \sum(3, 5) \text{ یا } \prod(0, 1, 2, 4, 6, 7)$$

$$d(x, y, z) = \sum(2, 6) \text{ یا } \prod(0, 1, 3, 4, 5, 7)$$

$$e(x, y, z) = \sum(\quad) \text{ یا } \prod(0, 1, 2, 3, 4, 5, 6, 7)$$

$$f(x, y, z) = \sum(1, 3, 5, 7) \text{ یا } \prod(0, 2, 4, 6)$$

به دست آوردن توابع از روی جدول درستی

برای نوشتند توابع از روی جدول درستی ابتدا ستون خروجی را در نظر می‌گیریم. سپس با توجه به اینکه می‌خواهیم تابع را به صورت

maxterm یا minterm بنویسیم، به یک یا صفر بودن خروجی توجه می‌کنیم.

اگر بخواهیم تابع را به صورت minterm بنویسیم باید یک‌های ستون خروجی را در نظر بگیریم. و minterm متاظر با هر یک، خروجی را بنویسیم. برای این کار باید در نظر داشته باشیم که اگر در ستون ورودی‌ها مقدار هر متغیر یک بود باید خودش را بنویسیم و اگر صفر بود باید مکملش را بنویسیم. (به عنوان مثال مقدار A در ردیف ششم یک است و B مقدارش صفر و C دارای ارزش یک می‌باشد بنابراین آن به شکل AB'C در می‌آید). پس از نوشتند تمامی minterm‌ها باید آنها را با هم جمع کنیم تا تابع بدست آید.



حال اگر بخواهیم تابع را به صورت maxterm بنویسیم باید صفرهای ستون خروجی را در نظر بگیریم و maxterm های متاظر با هر یک، از ردیفهای خروجی با ارزش صفر را بنویسیم. برای این کار باید در نظر داشت که اگر در ستون ورودی ها مقدار هر متغیر یک بود باید مکمل آن متغیر را بنویسیم و اگر صفر بود باید خود آن متغیر را بنویسیم. (به عنوان مثال در ردیف پنجم جدول درستی می‌بینیم که مقدار A یک می‌باشد و B صفر را به خود اختصاص داده و C نیز ارزش صفر دارد بنابراین maxterm مربوط به این ردیف برابر $A' + B + C$ می‌باشد). پس از نویشتن تمامی maxterm ها باید آنها را در هم ضرب کنیم تا تابع بدست آید.

$$x \begin{cases} 0 \rightarrow x' \\ 1 \rightarrow x \end{cases} \text{ min term}$$

$$x \begin{cases} 0 \rightarrow x \\ 1 \rightarrow x' \end{cases} \text{ max term}$$

	ورودی			خروجی	
	A	B	C	Z	
0	0	0	0	0	$\rightarrow A + B + C$
1	0	0	1	0	$\rightarrow A + B + C'$
2	0	1	0	0	$\rightarrow A + B' + C$
3	0	1	1	1	$\rightarrow A'BC$
4	1	0	0	0	$\rightarrow A' + B + C$
5	1	0	1	1	$\rightarrow AB'C$
6	1	1	0	1	$\rightarrow ABC'$
7	1	1	1	1	$\rightarrow ABC$

$$\Rightarrow Z(A, b, c) = \Sigma(3, 5, 6, 7)$$

$$Z = A'BC + AB'C + ABC' + ABC$$

$$\Rightarrow Z(A, b, c) = \Pi(0, 1, 2, 4)$$

$$Z = (A + B + C) \times (A + B + C') \times (A + B' + C) \times (A' + B + C)$$

مثال:

جدول درستی طراحی کنید که ۳ ورودی داشته باشد و ۴ خروجی به شکلی که خروجی ها از جمع ورودی با عدد ۳ بدست آمده باشد. سهیس تابع هر کدام از خروجی ها را بدست آورید.

	ورودی ها			خروجی ها			
	A	B	C	x	y	w	z
0	0	0	0	0	0	1	1
1	0	0	1	0	1	0	0
2	0	1	0	0	1	0	1
3	0	1	1	0	1	1	0
4	1	0	0	0	1	1	1
5	1	0	1	1	0	0	0
6	1	1	0	1	0	0	1
7	1	1	1	1	0	1	0

$$x = AB'C + ABC' + ABC$$

$$y = A'B'C + A'BC' + A'BC + AB'C'$$

$$w = A'B'C' + A'BC + AB'C' + ABC$$

$$z = A'B'C' + A'BC' + AB'C' + ABC'$$

تقدیم عملگرها

منظور از تقدیم این است که ارجحیت با کدام عملگر می‌باشد. با توجه به دانش قبلی که در ریاضیات داشتیم می‌دانیم که بالاترین تقدیم با اپراتور پرانتز می‌باشد یعنی در صورت وجود پرانتز باید ابتدا داخل آن محاسبه شود سپس به بقیه عملگرها رسیدگی کنیم.

(۱) پرانتز

not (x') (۲)

and (۳)

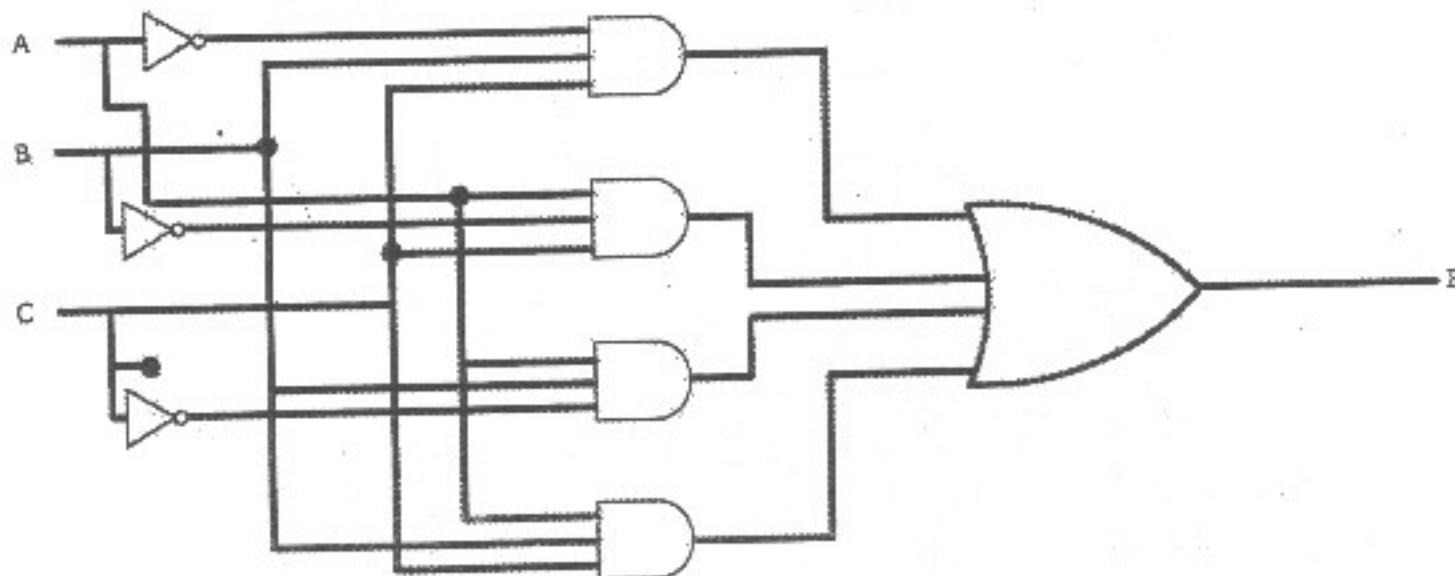
or (۴)

پیاده سازی توابع بول با گیت‌ها

برای پیاده سازی توابع به کمک گیتها باید ابتدا با استفاده از جدول درستی که در اختیار داریم تابع را بدست آوریم. سپس با توجه به شکل تابع بدست آمده آن را به کمک گیتهای مورد نیاز پیاده سازی کنیم. البته در ادامه خواهیم دید که به کمک روش‌های مختلف باید ابتدا تابع را ساده کنیم و سپس آن را پیاده سازی کنیم. به عنوان مثال اگر بخواهیم از روی جدول درستی زیر عمل پیاده سازی را انجام دهیم. باید ابتدا تابع را به فرم maxterm (یا minterm هایش) در آوریم سپس به کمک گیتهای NOT، OR، AND به کمک ورودی‌های a, b, c می‌توان عمل پیاده سازی را به فرم زیر انجام دهیم.

	a	b	c	F
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

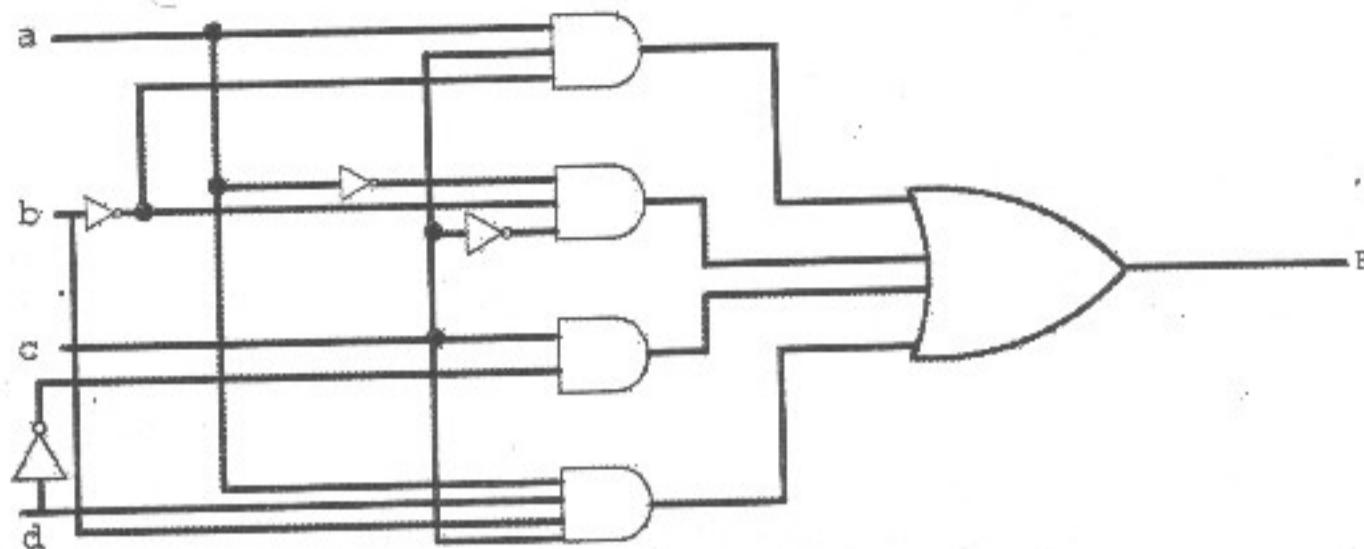
$$F(a, b, c) = \sum(3, 5, 6, 7) = a'b'c + ab'c + abc' + abc$$



مثال:

تابع زیر را به کمک گیتها پیاده سازی کنید.

$$F(a, b, c) = ab'c + a'b'c' + cd' + abcd$$



$$2^n = 2^3 = 8$$

مثال: مداری طراحی و پیاده سازی کنید که دارای سه ورودی باشد و خروجی معادل Ex - 3 ورودی باشد.

(با استفاده از جدول کارنو ساده کنید)

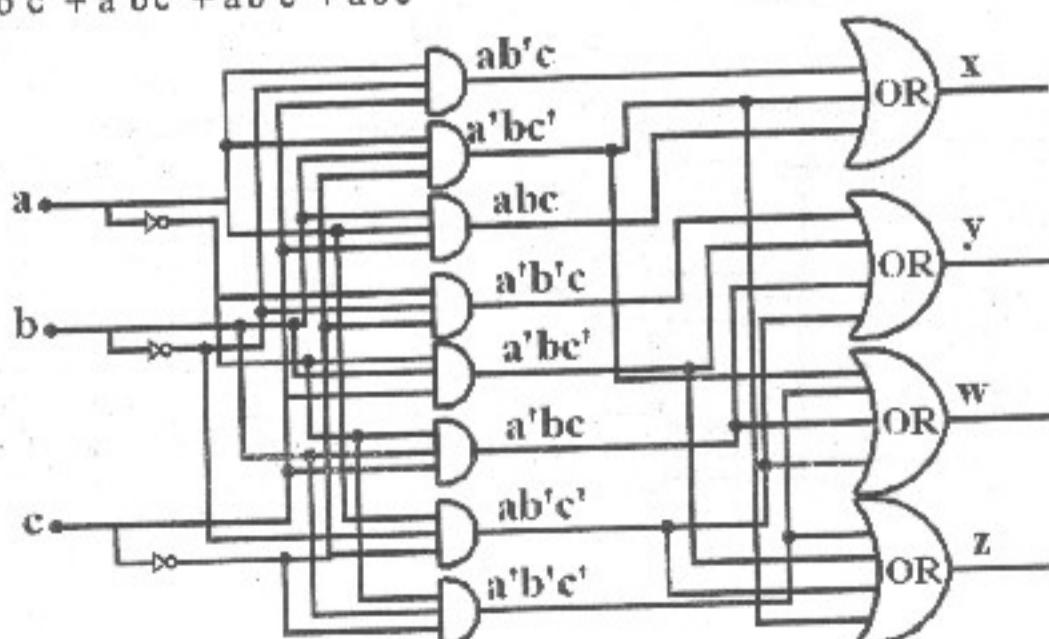
	4	2	1	x	y	w	z
	a	b	c				
0	0	0	0	0	0	1	1
1	0	0	1	0	1	0	0
2	0	1	0	0	1	0	1
3	0	1	1	0	1	1	0
4	1	0	0	0	1	1	1
5	1	0	1	1	0	0	0
6	1	1	0	1	0	0	1
7	1	1	1	0	1	0	0

$$x = \sum(5, 6, 7) = ab'c + abc' + abc$$

$$y = \sum(1, 2, 3, 4) = a'b'c + a'bc' + a'bc + ab'c'$$

$$w = \sum(0, 3, 4, 7) = a'b'c' + a'bc + ab'c' + abc$$

$$z = \sum(0, 2, 4, 6) = a'b'c' + a'bc' + ab'c' + abc'$$

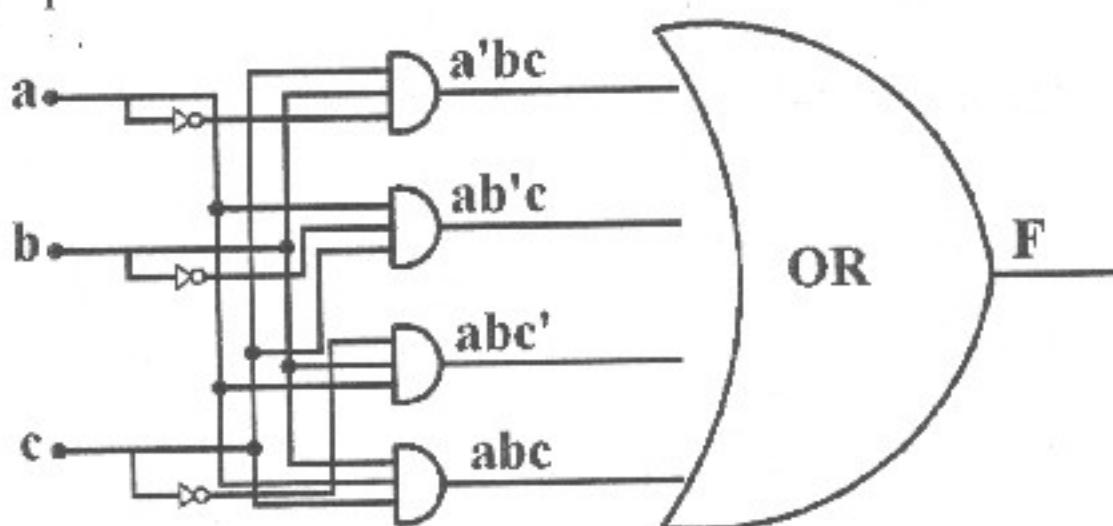


مدار منطقی

مثال : به کمک مدارهای منطقی میز رأی گیری سه نفرهای را طراحی کنید که دارای یک چراغ باشد. این چراغ در صورتی روشن می شود که یک طرح پذیرفته شود در غیر اینصورت خاموش خواهد بود. در صورتی یک طرح مورد قبول وارد می شود که حداقل دو نفر از سه رأی دهنده موجود جواب مثبت بدهند (جواب مثبت یعنی اینکه یک باشد و جواب منفی یعنی صفر باشد) مدار را طراحی و پیاده سازی کنید.

	a	b	c	F
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

$$F = \sum (3, 5, 6, 7) = a'b'c + ab'c + abc' + abc$$



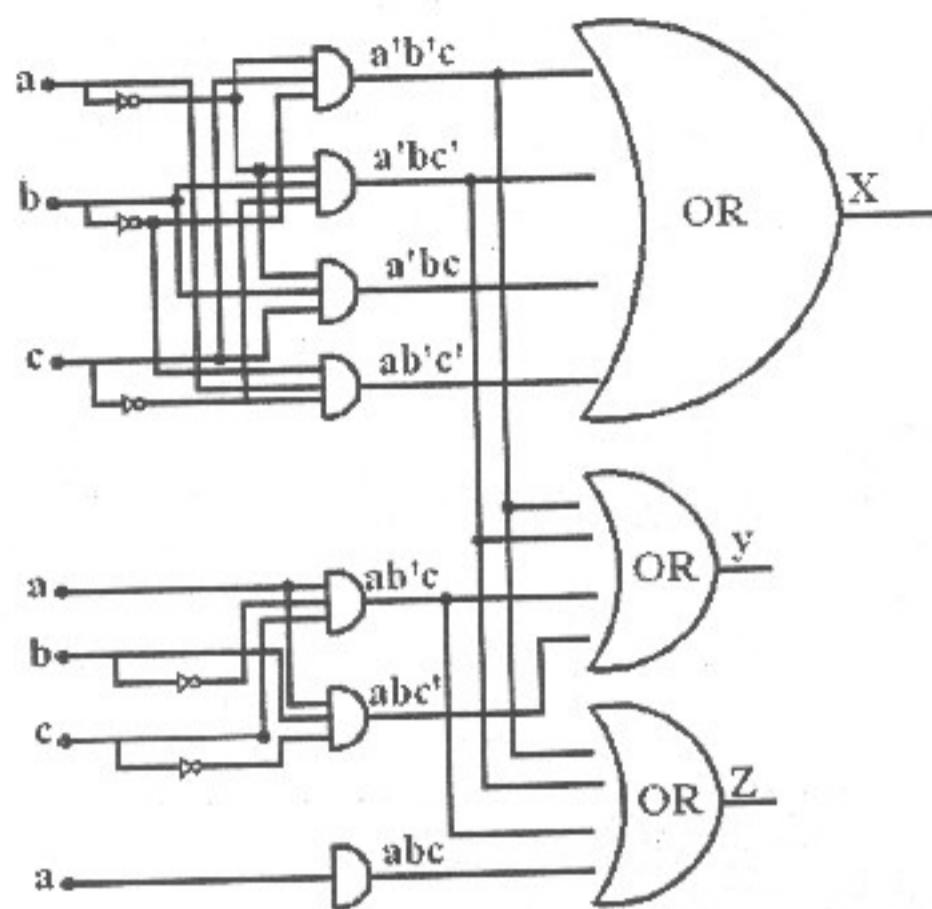
مثال : مداری طراحی و پیاده سازی کنید که ورودی آن یک عدد سه بیتی باشد و خروجی آن مکمل ۲ ورودی. (مکمل ۲ یعنی به اولین یک که رسیدیم خودش را نوشه و بقیه را عکس می کنیم).

	a	b	c	x	y	z
0	0	0	0	0	0	0
1	0	0	1	1	1	1
2	0	1	0	1	1	0
3	0	1	1	1	0	1
4	1	0	0	1	0	0
5	1	0	1	0	1	1
6	1	1	0	0	1	0
7	1	1	1	0	0	1

$$x = \sum (1, 2, 3, 4) = a'b'c + a'b'c' + a'b'c + ab'c'$$

$$y = \sum (1, 2, 5, 6) = a'b'c + a'b'c' + ab'c + abc'$$

$$z = \sum (1, 3, 5, 7) = a'b'c + a'b'c + ab'c + abc$$



مکمل گیری از قوای:

$$x' \rightarrow x$$

$$x \rightarrow x'$$

$$1 \rightarrow 0$$

$$0 \rightarrow 1$$

$$x \rightarrow +$$

$$+ \rightarrow x$$

برای مکمل گیری باید به جای ۱ صفر و به جای صفر ۱ قرار می‌دهیم و به جای ضرب، جمع و به جای جمع هم ضرب قرار می‌دهیم

ساده سازی با استفاده از جدول کارنو:

مرحله اول: باید تمام میترم‌ها را به صورت کامل درآورد.

مرحله دوم: تعیین نمودن تعداد خانه‌های جدول کارنو با استفاده از فرمول 2^n که n در آن تعداد ورودی‌ها می‌باشد.

مرحله سوم: باید قسمت بندی ورودی‌ها را روی جدول انتخاب کنیم مهمترین نکته در این مرحله وجود همپوشانی برای ورودی‌ها است.

مرحله چهارم: باید جدول را با استفاده از میترم‌ها و قرار دادن یک‌ها در خانه‌های جدول پر کنیم.

مرحله پنجم: ساده کردن جدول که خود شامل دو قسم است:

قسمت اول: پیدا کردن همسایه‌ها (همسایه به خانه‌هایی گفته می‌شود که حداقل یک ضلع مشترک داشته باشند)

قسمت دوم: باید ناحیه همسایه‌ها را بخواهیم و تشخیص دهیم محدوده مورد نظر کجا قرار دارد.

* همسایه‌ها باید بزرگترین مضرب 2 باشند. (2^n) .

مدار منطقی

مثال:

با توجه به درستی زیر تابع f را به کمک جدول کارتئو ساده کنید.

a	b	c	F
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

تعداد خانه‌های جدول

$$= 2^3 = 8$$

$$F = a'b'c + ab'c + abc' + abc$$

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

$$F = c + ab$$

$$F = a'b'c + ab'c + abc' + a'b'c + abc$$

مثال:

تابع F را به کمک جدول کارتئو ساده کنید:

$$F = (a, b, c, d) = \Sigma(1, 2, 3, 4, 7, 11, 12, 14, 15)$$

a	b	c	d	F
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
A	1	0	1	0
B	1	0	1	1
C	1	1	0	0
D	1	1	0	1
E	1	1	1	0
F	1	1	1	1

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

$$F = cd + abc + a'b'c + a'b'd + bc'd'$$

مثال: جدول کارنو را برای ساده‌سازی مربوط به میز رأی گیری مهندس نفره را رسم کنید.

$$F(a, b, c) = \sum(3, 5, 6, 7)$$

	bc	00	01	11	10
a	0	0	1	1	2
	1	4	5	7	6

در این ستون a ها صفر است.
در این ستون a ها یک است.

c

مرحله اول تبدیل جملات

ضرب به فرم استاندارد

مثال:

تابع F را وارد جدول کارنو کنید:

$$F(w, x, y, z) = y'z + wxy' + wxz' + w'x'z$$

تعداد خانه‌های جدول $= 2^4 = 16$

حل: باید ابتدا جملات را به فرم استاندارد (minterm) درآوریم:

$$y'z = xy'z + x'y'z \Rightarrow wxy'z + w'xy'z + wx'y'z + w'x'y'z$$

$$wxy' = wxy'z + wxy'z'$$

$$wxz' = wxyz' + wxy'z'$$

$$w'x'z = w'x'yz + w'x'y'z$$

$$F = wxy'z + w'x'y'z + wx'y'z + wxy'z' + wxyz' + w'x'yz$$

حال با توجه به اینکه تابع به صورت استاندارد (minterm) درآمده می‌توان آن را وارد جدول کارنو کرد. با توجه به اینکه تابع f به صورت (w, x, y, z) می‌باشد، با اختصاص خانه‌های جدول و نیز مشخص کردن صفر یا یک بودن هر متغیر به این صورت که اگر متغیر خودش

بود یک و اگر متمم شود صفر قرار می‌دهیم، می‌توان جدول را پر کنیم. به عنوان مثال در $wxyz'$ در واقع به جای $wxyz'$ باید صفر و به جای x و y یک قرار دهیم و minterm به شکل ۱۱۰ در می‌اید که دو بیت سمت چپ ردیف و دو بیت سمت راست ستون را مشخص می‌کند.

سپس از تقاطع آنها می‌توان به خانه مورد نظر هست یافت.

مدار منطقی

ردیف: ۰۱

حال تمامی minterm ها را به شکل زیر پر می کنیم.

WX	00	01	11	10
YZ		1	1	
00		1		
01				
11	1	1		1
10		1		

شماره گذاری خانه های جدول کارنو:

در شماره گذاری خانه های جدول شماره های داخل جدول همان شماره minterm ها می باشد. و شماره یرونی مربوط به وجود و عدم وجود متغیر می باشد. بعضی صفر به معنای آن است که متغیر وجود ندارد و یک به معنای وجود متغیر می باشد.

x	y	0	1
0	0	1	
1	2	3	

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

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

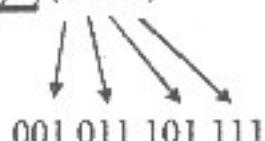
DE	BC	00	01	11	10
		16	17	19	16
		20	21	23	22
		28	29	31	32
		24	25	27	26

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

مثال:

تابع F را وارد جدول کارنو کنید.

$$F(x, y, z) = \sum(1, 3, 5, 7)$$



	00	01	11	10	$F = z$
0	0	1	3	2	
1	4	5	7	6	

مثال:

تابع زیر را در جدول کارنو وارد کنید و همسایه‌گیری کنید.

0000 0001 0010 0101 1000 1001 1010



$$t(x, y, w, z) = \sum(0, 1, 2, 5, 8, 9, 10, 15)$$

$$t = y'w' + y'z' + x'w'z + xyzw$$

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

مثال:

با استفاده از کارنو 5 متغیره تابع زیر را ساده کنید:

$$F(A, B, C, D, E) = \Sigma(0, 2, 6, 7, 8, 10, 14, 15, 16, 18, 20, 21, 22, 23, 24, 26, 28, 29, 30, 31)$$

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

 $A = 0$

	00	01	11	10
00	1			1
00	16	17	19	18
01				
01	20	21	23	22
11				
11	1		1	1
10				
10	28	29	31	30
11				
11	1			1
10				
10	24	25	27	26

 $A = 1$

$$f = C'E' + AC + CD$$

مدار منطقی



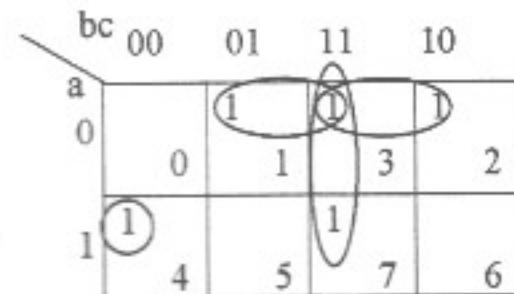
مثال: مداری طراحی و به کمک جدول کارنو ساده کنید که دارای ۳ ورودی برای ورودی‌های صفر تا چهار خروجی، ورودی به اضافه ۳ باشد و از ۵ تا ۷ خروجی، ورودی منهای ۳ باشد.

a	b	c	x	y	z
0	0	0	0	1	1
1	0	0	1	0	0
2	0	1	1	0	1
3	0	1	1	1	0
4	1	0	0	1	1
5	1	0	1	0	0
6	1	1	0	0	1
7	1	1	1	0	0

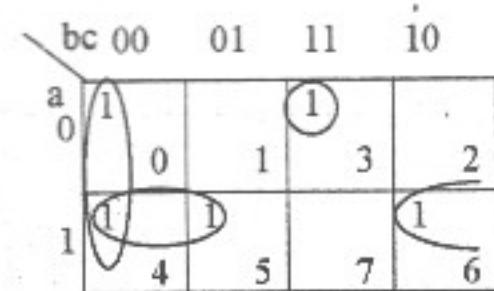
$$x = \Sigma(1, 2, 3, 4, 7) = a'b'c + a'b'c' + a'b'c + ab'c' + abc$$

$$y = \Sigma(0, 3, 4, 5, 6) = a'b'c' + a'b'c + ab'c' + ab'c + abc'$$

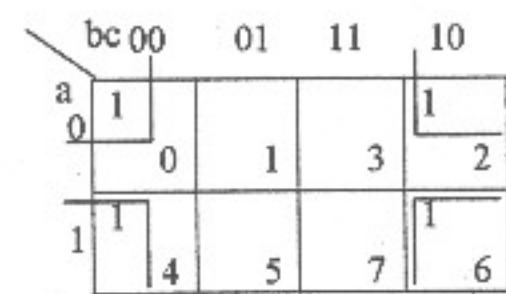
$$z = \Sigma(0, 2, 4, 6) = a'b'c' + a'b'c + ab'c' + abc'$$



$$x = a'c + a'b + bc + ab'c'$$



$$y = ab' + b'c' + ac' + a'b'c$$



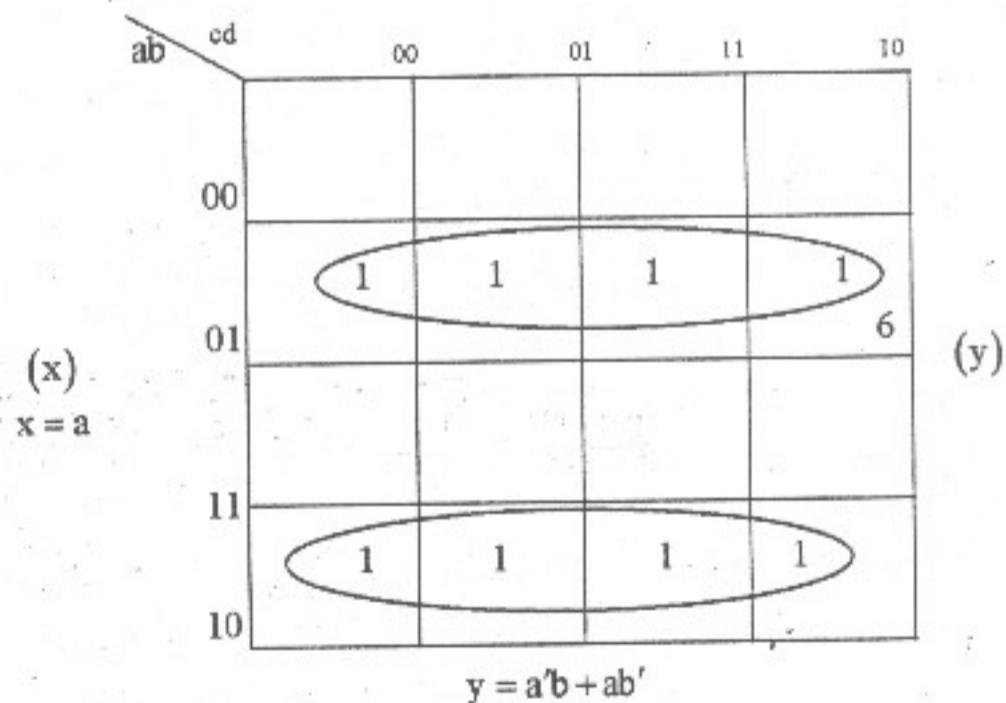
$$z = c'$$

مثال: مداری طراحی کنید و آن را به کمک جدول کارنو ساده کنید که ورودی آن یک عدد 4 بیتی باشد و خروجی اش معادل Gray ورودی باشد.

$$2^4 = 16$$

	a	b	c	d	x	y	w	z
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

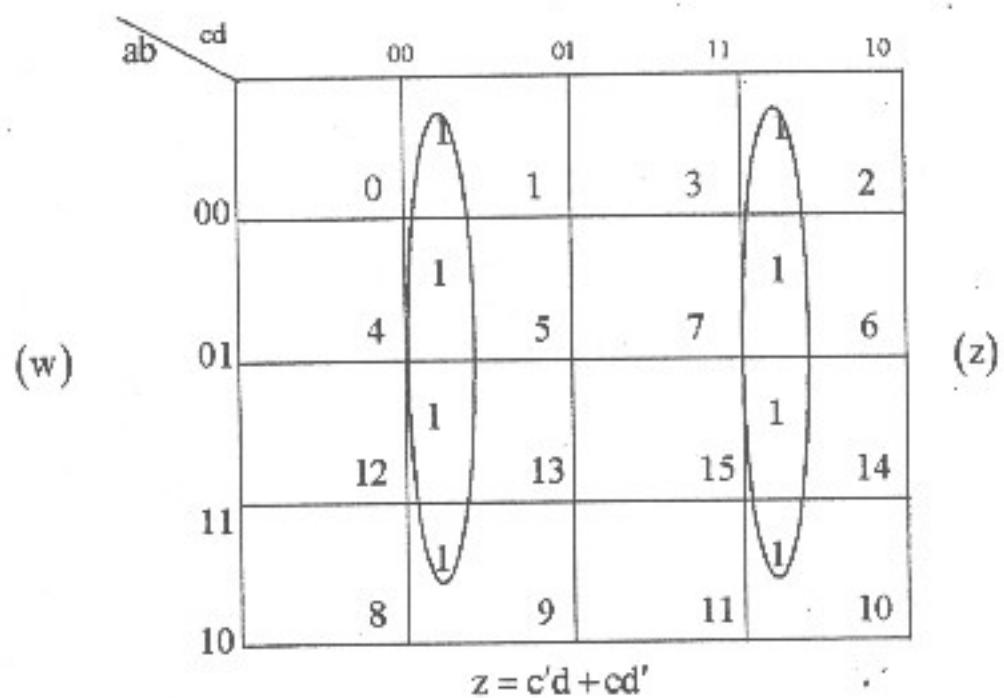
ab	cd	00	01	11	10
1					
00	0	1	3	2	
01	4	5	7	6	
11	1 12 1	1 13 1	1 15 1	1 14 1	
10	8	9	11	10	



مدار منطقی

	cd	00	01	11	10
ab	00	0	1	1	1
	01	1	4	5	7
	11	12	13	15	14
	10	8	9	11	10

$$w = b'c + b'c'$$



شکل دیگر شماره گذاری جدول کارنو:

شماره گذاری جدول کارنو (یا در اصل اختصاص خانه‌های جدول کارنو به متغیرها) انواع متفاوتی دارد، اولین نکته مهم این است که باید اصل همپوشانی در شماره گذاری یا اختصاص خانه‌های جدول رعایت شود. دومین نکته این است که باید شماره گذاری جدول به نحوی باشد که یک فرم استاندار را رعایت کرده باشد. به این مفهوم که ساده شده یک تابع به کمک دو جدول کارنو که از شماره گذاری‌های متفاوت اما استاندار استفاده می‌کنند نباید تفاوت داشته باشند.

شکل دیگر شماره گذاری جدول کارنو

اگر جای متغیرها را در جدول کارنو تغییر دهیم به شکل جدید شماره گذاری می‌رسیم. به این مفهوم اگر در یک کارنوی چهار متغیره که تابع $F(x,y,w,z)$ را نشان می‌دهد به جای اینکه به ماتند شماره گذاری در حالت قبل xy را به ردیفها و wz را به ستونها اختصاص دهیم، بر عکس عمل کنیم یعنی xy را به ستونها و wz را به ردیفها اختصاص دهیم، شماره‌های داخل جدول کارنو به فرم زیر تغییر می‌کنند.

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

مالحظه می‌شود تنها تفاوت این روش با روش قبلی در نحوه ناحیه بندی آن می‌باشد. توجه داشته باشید که تفاوتی ندارد که از کدام نوع شماره گذاری برای جدول کارنو خود استفاده می‌کنید، تنها این مسئله مهم است که باید حتماً از استاندارهای شماره گذاری استفاده کنید تا در پایان ساده شده تابع به درستی به وجود آید.



نحوه شکل گیری شماره‌های داخلی جدول کارنو:

هر خانه از جدول کارنو یک شماره دودویی مختص به خود را دارد است. به این شکل که در کنار هر ردیف و هر ستون یک شماره دودویی دیده می‌شود (مثلًا 00 یا 10) و با توجه به نام متغیری که به آن اختصاص داده شده به مفهوم وجود یا عدم وجود آن متغیر می‌باشد.

به عنوان مثال در جدول کارنوی زیر که مربوط بهتابع $F(A,B,C)$ می‌باشد داریم:

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

کنار ردیف اول عدد صفر نوشته شده که به مفهوم عدم وجود A در این ردیف می‌باشد و ردیف دوم یک به معنای اختصاص این ردیف به A است.

یا عدد 01 در ستون دوم به معنای وجود C و عدم وجود B می‌باشد.

حال اگر بخواهیم یک خانه را شماره‌گذاری کنیم با توجه به ارزش مکانی متغیرها (به این مفهوم که در مثال فوق A بیشترین ارزش مکانی و C کمترین ارزش را دارد) مقدار دودویی تبدیل به معادل دهدگی می‌باشد. به عنوان مثال در جدول فوق اگر بخواهیم خانه‌ای در ردیف اول و در ستون چهارم را شماره‌گذاری کنیم داریم:

$$\begin{cases} A = 0 \\ BC = 10 \end{cases} \rightarrow ABC = 010 \quad .2$$

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

A'	B	C	شماره
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

این روش را برای تمامی خانه‌های جدول کارنو می‌توان پیاده کرد.

بنابراین ملاحظه می‌شود که با تغییر در جای متغیرها، باز هم می‌توان خانه‌های جدول را با توجه به ارزش مکانی آنها شماره‌گذاری کرد.

به عنوان مثال برای $F(x,y,z)$ داریم:

مدار منطقی

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

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

$\overbrace{\hspace{250pt}}$

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

$\overbrace{\hspace{150pt}}$

$f(x, y, w, z)$
 $x \quad y \quad w \quad z$
 1 1 0 0

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

$\overbrace{\hspace{150pt}}$

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

	x00	x01	x11	x10
y00	0	4	12	8
y01	1	5	13	9
y11	3	7	15	11
y10	2	6	14	10

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

حالت‌های بی‌اهمیت یا Don't care:

به حالت‌هایی گفته می‌شود که یک بودن یا صفر بودن آنها تفاوت نمی‌کند. و طراح مطابق با نیاز خود آن را صفر یا یک در نظر می‌گیرید.
در جدول کارنو به جای حالات don't care باید X گذاشته شود.

ویژگی اصلی حالت‌های بی‌اهمیت در این است که می‌توان آنها را هم با صفرها و هم با یک‌ها همسایه گرفت.

* اگر در یک جدول درستی حالت‌ها فقط یک و don't care یا صفر و don't care باشند احتیاج به جدول کارنو نداریم زیرا در اولی حاصل برابر یک و در دومی حاصل برابر صفر است.

مثال:

تابع f را با حالات بی‌اهمیت d در نظر بگیرید، آن را ساده کنید:

$$f(a, b, c) = \sum(1, 3, 7) \quad d = \sum(0, 2, 5)$$

	a00	a01	a11	a10
bc00	X	1	1	X
bc01	0	1	3	2
bc11	X	1	1	
bc10	4	5	7	6

$F = C$

شماره‌های صفر و 2 را در نظر نمی‌گیریم، چون تابع را مشکل‌تر می‌کنند. اما شماره 5 را به عنوان یک در نظر می‌گیریم و می‌بینیم که باعث ساده‌تر شدن تابع می‌شود.

حال فرض کنیم که همه حالات بی‌اهمیت را صفر در نظر بگیریم، خواهیم دید که تابع به فرم $F = a'c + bc$ در می‌آید. که از فرم قبلی ($F = c$) پیچیده‌تر می‌باشد بنابراین در این مثال استفاده از حالت بی‌اهمیت به سود ما بود.

مدار منطقی

اما نمی‌توان گفت که حالات بی‌اهمیت همیشه برای ما مفید خواهد بود؛ بلکه با توجه به مکان این حالات می‌توان از آنها استفاده بهینه کرد.
نکته:

اگر خانه‌های ۱,۲,۴,۷ جدول کارنو یک بود \leftarrow XOR سه ورودی

اگر خانه‌های ۰,۳,۵,۶ جدول کارنو یک بود \leftarrow XNOR سه ورودی

یا اگر در خانه‌های جدول خانه‌ایی که مقدارشان یک است در موقعیت یکی در میان بودند و شروع از خانه صفر بود تابع X-NOR ورودی‌ها پیاده سازی شده و اگر شروع از خانه یک بود تا X-OR پیاده سازی شده است.

wz	X-OR	چهار ورودی	
xy		1	1
1	1	1	
1	1		1
1	1	1	
1	1		1

wz	XNOR	چهار ورودی	
xy		1	1
1	1	1	
1	1		1
1	1	1	
1	1		1

ایجاب کننده (Implicant)

در یک تابع n متغیره به صورت $f(x_1, x_2, \dots, x_n)$ اگر جمله حاصلضرب P حداقل یک نقطه یک تابع را پوشاند و هیچ نقطه صفر را پوشاند ایجاب کننده تابع F نامیده می‌شود. در اصل یک عبارت ضرب می‌باشد که یک با چند Minterm را شامل شود.

ایجاب کننده نخستین (Prime Implicant) یا (PI)

ایجاب کننده P از تابع F را ایجاب کننده نخستین گویند هر گاه برای هر ایجاب کننده دیگر مانند P نقطه‌ای وجود داشته باشد که به وسیله P پوشانده نشود ولی به وسیله P پوشانده شود.

مثال:

با توجه به جدول درستی زیر نوع P ‌ها را مشخص کند.

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	-
1	0	1	1
1	1	0	0
1	1	1	0

ایجاب کننده است $\rightarrow P_1 = A'B'$

ایجاب کننده نیست $\rightarrow P_2 = A'B$

ایجاب کننده است $\rightarrow P_3 = AB'$

حال می‌توان P_4 را به وجود آورد که از ترکیب P_1, P_2, P_3 تشکیل شده است:

$$P_1, P_3 \Rightarrow P_4 = B'$$

همانطور که ملاحظه می‌شود P_4 توانایی پوشش چهار مینترم شماره‌های (۰, ۱, ۴, ۵) را دارد. در اصل P_4 یک ایجاب کننده نخستین می‌باشد زیرا نمی‌توان پوشش دیگری پیدا کرد که چهار مینترم فوق را پوشاند.

نکته: ملاحظه می‌شود که هر ایجاب کننده در واقع یک جمله حاصلضرب است که ساده شده minterm‌ها می‌باشد. بنابراین اگر بتوان یک تابع را به صورت ایجاب کننده‌هایش در آوریم آن را به ساده‌ترین صورت نوشتیم.

حال اگر تابع را به شکل PI‌هایش بنویسیم در اصل ساده‌ترین صورت آن تابع را بدست آورده‌ایم.



بدست آوردن API‌های توابع:

می‌دانیم که هر minterm یک نقطه از تابع را می‌پوشاند و از ترکیب دو میترم که فقط در یک بیت (یک متغیر) با هم تفاوت دارند می‌توان جمله حاصلضربی آیجاد نمود که دو نقطه از تابع را می‌پوشاند. متغیری که متفاوت بود حذف شده و جای آن می‌توان علامت "-"- به معنی بی‌همیت را قرارداد حال جمله حاصلضرب بدست آمده را می‌توان با بقیه جملات حاصلضرب که آنها هم از ساده شدن minterm‌های دیگر تشکیل شده، مقایسه کرد و عمل ساده کردن را انجام داد، این کار را تا جایی ادامه می‌دهیم تا امکان هیچ ترکیب دیگری برای ساده سازی وجود نداشته باشد. با انجام عملیات فوق در اصل بد API‌های یک تابع رسیده‌ایم.

روش QM برای ساده‌سازی توابع (کوئین مک کلاسکی):

با استفاده از الگوریتم QM می‌توان توابع را به راحتی ساده کرد:

مرحله ۱: دسته‌بندی minterm‌ها:

منظور از دسته‌بندی minterm‌ها جداسازی آنها با توجه به تعداد یک‌های آنها می‌باشد به عنوان مثال دو؟؟ که به صورت ABCD و ABC'D هستند در اصل به صورت 1101 و 0111 هستند که در هر دو آنها سه عدد "1" وجود دارد بنابراین هر دو در یک دسته قرار خواهد گرفت. بنابراین در این مرحله جدولی را تشکیل می‌دهیم که در آن دسته‌های مختلف minterm‌ها وجود دارند.

مرحله ۲: مقایسه minterm‌ها:

در این مرحله باید عمل مقایسه بین minterm را انجام دهیم به این شکل که هر minterm از یک دسته را با minterm در دسته بعدی مقایسه می‌کنیم اگر تنها در یک بیت تفاوت داشتند آن بیت متفاوت را حذف می‌کنیم. عمل مقایسه را تا زمانی انجام می‌دهیم که هیچ دو minterm از دو دسته متفاوت وجود نداشته باشند که با هم مقایسه نشده باشند به عنوان مثال در مقایسه دو minterm شماره‌های ۵ و ۷ که در دو دسته متفاوت قرار دارند خواهیم داشت:

$$\begin{array}{l} 101 \rightarrow \text{minterm 5} \\ 111 \rightarrow \text{minterm 7} \end{array} \left. \right\} \rightarrow 1-1$$

همانطور که ملاحظه می‌شود در دوین بیت با هم متفاوت هستند بنابراین آن بیت حذف می‌شود و به جای آن علامت "-". به معنای بی‌همیت یا Dont care قرار می‌گیرد.

پس از انجام عمل مقایسه بین هر دو کنار شماره آن علامت "—" را می‌زنیم. در انتها باید کنار تمام minterm‌ها علامت "—" گذاشته شده باشد.

مرحله ۳: مرحله دوم ساده‌سازی:

با انجام مرحله ۲ به یک ستون جدید در جدول رسیده‌ایم حال باید به آنها نیز به چشم minterm نگاه کنیم و مانند مرحله اول شروع به دسته‌بندی آنها کنیم سهی همان مراحل که برای مقایسه minterm‌ها انجام داده بودیم را اعمال کنیم.

باید توجه داشت که برای انجام دسته‌بندی تمامی حالات بی‌همیت را صفر در نظر می‌گیریم و عمل دسته‌بندی را انجام می‌دهیم.

پس از انجام عمل مقایسه می‌ینیم که معکن است در هر دسته minterm‌ی وجود داشته باشد که جلوی آن علامت "—" وجود ندارد، در اصل این جمله حاصلضرب یکی PI‌های ما خواهد بود که با هیچ جمله حاصلضرب دیگری قابل ساده شدن نیست.

پس از انجام مراحل فوق در انتها به یک جمله حاصلضرب می‌رسیم که آن جمله هم یک PI خواهد بود.

مدار منطقی

مثال:

تابع $f(A, B, C, D) = \sum(0, 1, 2, 5, 8, 10, 14, 15)$ را به کمک روش M ساده کنید.

مرحله اول: دسته‌بندی

minterm شماره	A	B	C	D	
0	0	0	0	0	دسته صفر
1	0	0	0	1	
2	0	0	1	0	دسته یک
8	1	0	0	0	
5	0	1	0	1	دسته دو
10	1	0	1	0	
14	1	1	1	0	دسته سه
15	1	1	1	1	دسته چهار

همان‌طور که ملاحظه می‌شود minterm‌ها به پنج دسته تقسیم شدند که در دسته ایم تعداد یک‌ها ناتای می‌باشد.

مرحله دوم: مقایسه بین دو minterm در دسته‌های متفاوت

minterm شماره	A	B	C	D	minterm شماره مقایسه شده	A	B	C	D
0	0	0	0	0	(0,1) →	0	0	0	-
1	0	0	0	1	(0,2) →	0	0	-	0
2	0	0	1	0	(0,8) →	-	0	0	0
8	1	0	0	0	(1,5) →	0	-	0	1
5	0	1	0	1	(2,10) →	-	0	1	0
10	1	0	1	0	(8,10) →	1	0	-	0
14	1	1	1	0	(10,14) →	1	-	1	0
15	1	1	1	1	(14,15) →	1	1	1	-

حال باید حاصل minterm‌های ساده شده را در دسته‌بندی کنیم:

	A	B	C	D	نام
(0,1) →	0	0	0	-	$P_2 \} P_3 \} P_4 \}$ دسته صفر
(0,2) →	0	0	-	0	
(0,8) →	-	0	0	0	
(1,5) →	0	-	0	1	$P_5 \} P_6 \} P_7 \}$ دسته یک
(2,10) →	-	0	1	0	
(8,10) →	1	0	-	0	دسته دو
(10,14) →	1	-	1	0	
(14,15) →	1	1	1	0	دسته سه



حال باز هم باید عمل مقایسه را انجام دهیم:

$$\begin{array}{l} (P_3, P_7) \rightarrow -0-0 \\ (P_4, P_6) \rightarrow -0-0 \end{array} \Rightarrow$$

متترم‌های شماره (0, 2, 8, 10) با یکدیگر ساده شده‌اند و تبدیل به:

$$\begin{array}{cccc} A & B & C & D \\ - & 0 & - & 0 \end{array}$$

شده است بنابراین $B'D'$ یک PI می‌باشد و تابع به مجموع PI‌های زیر تبدیل شده است:

$$\{A'B'C' + A'C'D + ACD' + ABCD' + B'D'\}$$

انتخاب پوشش مینیمم از مجموعه PI‌های تابع:

پوشش: انتخاب حداقل تعداد از PI‌های تابع با کمترین هزینه به طوری که هر نقطه "1" تابع حداقل به وسیله یکی از PI‌های تابع پوشانده شود را پوشش گویند.

جدول پوشش: جدولی می‌باشد که به ازای هر PI یک سطر و به ازای یک‌های تابع (minterm) ستون دارد.

تحوه پر کردن جدول پوشش: در تقاطع سطر i ام و ستون j ام یک درج می‌کنیم اگر ایجاد کننده نخستین P_i متترم M_j را پوشاند. تکه: نقاط Don't Case در جدول درج نمی‌شوند زیرا نیازی به پوشش آنها نیست.

برای بدست آوردن پوشش مینیمم کافیست جدول پوشش را ساده کرد برای ساده کردن جدول پوشش کافیست از الگوریتم زیر پیروی کیم:

۱- حذف PI‌های ضروری

در صورتی که متترم M_{ij} تهابه وسیله یک P_i پوشانده شود آن را PI ضروری گویند که باید در هر پوشش مینیمم ظاهر شده باشد جدول را با حذف سطر P_i و کلید ستون‌های پوشیده شده به وسیله آن را حذف می‌کنیم.

۲- حذف سطراهای مغلوب:

اگر P_i کلید نقاط متناظر با P_j را پوشاند در این صورت گوئیم P_i غالب بر سطر P_j است. در این صورت سطر مغلوب یعنی P_j را حذف می‌کیم.

۳- حذف ستون‌های غالب:

اگر هر سطحی که m_i را نیز پوشاند و نسبت به m_j نقاط بیشتری داشته باشد گوئیم m_i غالب (Dominant) نیز m_j است. در این صورت ستون غالب یعنی m_i را حذف می‌کیم.

مثال:

جدول پوشش تابع

$$f(x_1, x_2, x_3, x_4) = \sum(0, 1, 2, 5, 8, 10, 14) + d(7)$$

برای بدست آوردن مجموعه PI‌های تابع f از روش QM می‌توان سود برد:

$$PI = \{x'_1 x'_2 x'_3, x'_1 x'_3 x'_4, x'_1 x'_2 x_4, x_1 x_3 x'_4, x_1 x_2 x_3, x_2 x_3 x_4, x'_2 x'_4\}$$

شماره PI		m_0	m_1	m_2	m_5	m_8	m_{10}	m_{14}	m_{15}
P_1	$x'_1 x'_2 x'_3$	1	1						
P_2	$x'_1 x'_3 x'_4$		1		1				
P_3	$x'_1 x'_2 x_4$						1	1	
P_4	$x_1 x_3 x'_4$							1	1
P_5	$x_1 x_2 x_3$								1
P_6	$x_2 x_3 x_4$		1			1	1		
P_7	$x'_2 x'_4$			1					

مرحله اول حذف PI‌های ضروری:

ضروری است بنابراین سطر P_7 و ستون‌های m_0, m_2, m_8, m_{10} از جدول حذف می‌شود.

مرحله دوم حذف سطرهای مغلوب:

همان‌طور که دیده می‌شود P_2 بر P_3 و P_1 و P_5 بر P_6, P_4 غالب است.بنابراین P_6, P_4, P_1 باید حذف شوند.

		m_1	m_5	m_{14}	m_{15}
P_1	$x'_1 x'_2 x'_3$	1			
P_2	$x'_1 x'_3 x'_4$	1	1		
P_3	$x'_1 x'_2 x_4$		1		
P_4	$x_1 x_3 x'_4$			1	
P_5	$x_1 x_2 x_3$			1	1
P_6	$x_1 x_2 x_3$				1
P_7	$x_2 x_3 x_4$				

بنابراین تنها سطرهای P_5, P_2 باقی می‌مانند که به همراه PI7 تابع را به صورت کامل پوشش می‌دهند.

$$f = P_7 + P_2 + P_5 = x'_2 x'_4 + x'_1 x'_3 x'_4 + x_1 x_2 x_3$$

تاخیر انتشار:

مدت زمان تاخیر بین تغییر ورودی و خروجی یک گیت را گویند به این مفهوم که چه مدت زمانی طول می‌کشد که تغییر در ورودی باعث تغییر در خروجی شود و دونوع تاخیر انتشار وجود دارد:

T_{PHL} : مدت زمان تاخیری که خروجی از یک به صفر تغییر می‌کند.

T_{PLH} : مدت زمان تاخیری که خروجی از صفر به یک تغییر می‌کند.

$$T_{PD} = \frac{T_{PHL} + T_{PLH}}{2}$$

پیاده سازی با گیت های NOR و NAND:

برای پیاده سازی NOT بصورت تمام NAND از یک NAND تک ورودی و برای پیاده سازی تمام NOR از یک NOR تک ورودی استفاده می‌کنیم.

در اصل دو راه برای به وجود آوردن NOR یا NAND تک ورودی داریم که به صورت NOT عمل کند، راه اول این است که دو ورودی را به یکدیگر وصل کنیم و راه دوم این است که یک ورودی را حتماً یک کنیم.

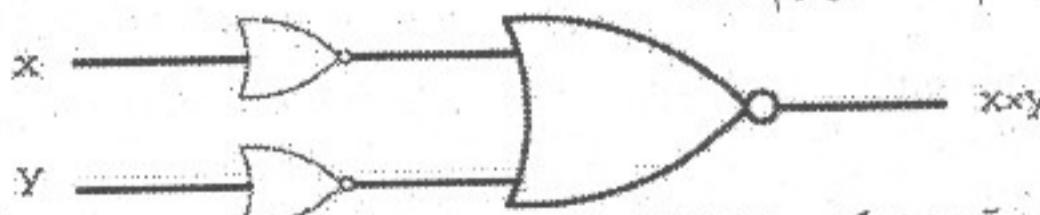
باید توجه داشت که اگر در NAND تک ورودی و یا در NOR تک ورودی پشت سر هم باشند چون عمل NOT کردن دو بار انجام می‌شود به معنی این است که هیچ عملی انجام نشده و در اصل این دو NOT پشت سر هم کار یک سیم را انجام می‌دهند و یکدیگر را خشی می‌کنند بنابراین می‌توان آنها را حذف کرد و یک سیم به جای آنها گذاشت.

برای پیاده سازی AND در طراحی تمام NAND خواهیم داشت:



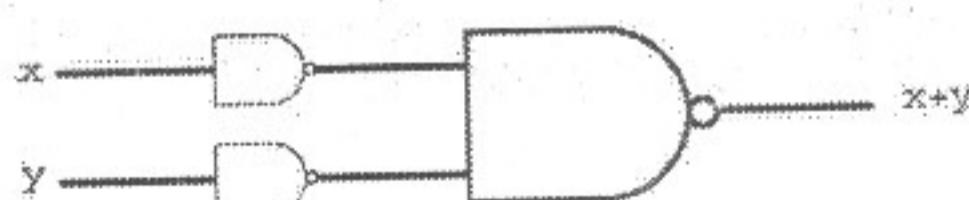
یعنی باید هر کجا که گیت AND دیدیم آن را حذف کنیم و به جای آن شکل تمام NAND آن را قرار دهیم.

برای پیاده سازی AND در طراحی تمام NOR خواهیم داشت:



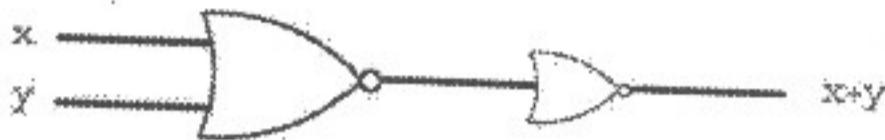
یعنی هر کجا گیت AND داشتیم باید آن را با شکل تمام NOR که طراحی کردیم جایگزین کنیم.

برای پیاده سازی OR در طراحی تمام NAND خواهیم داشت:



مدار منطقی

برای پیاده سازی OR در طراحی تمام NOR خواهیم داشت:



برای پیاده سازی Not در طراحی تمام NOR و NAND خواهیم داشت:



دلیل استفاده از طراحی مدارهای تمام NOR و NAND:

به علت استفاده از IC ها طراحی تمام NOR و NAND ممکن است 2 یا 4 عدد از آن گیت وجود داشته باشد بنابراین مقرر نبود که از یک IC استفاده لازم رایبریم.

به عنوان مثال فرض کنید بخواهیم تابع $F = AB + C'$ را پیاده سازی کنیم در حالت عادی باید از یک گیت OR و یک گیت AND و یک گیت NOT استفاده کنیم که برای پیاده سازی به کمک ICها باید 3 عدد IC متفاوت را به کار بیریم. با توجه به اینکه در هر IC چهار گیت از آن نوع وجود دارد بنابراین در پایان پیاده سازی، سه گیت OR، سه گیت AND و 5 گیت NOT بدون استفاده خواهیم داشت. که نشاندهنده عدم استفاده صحیح (طراحی صحیح) می باشد.

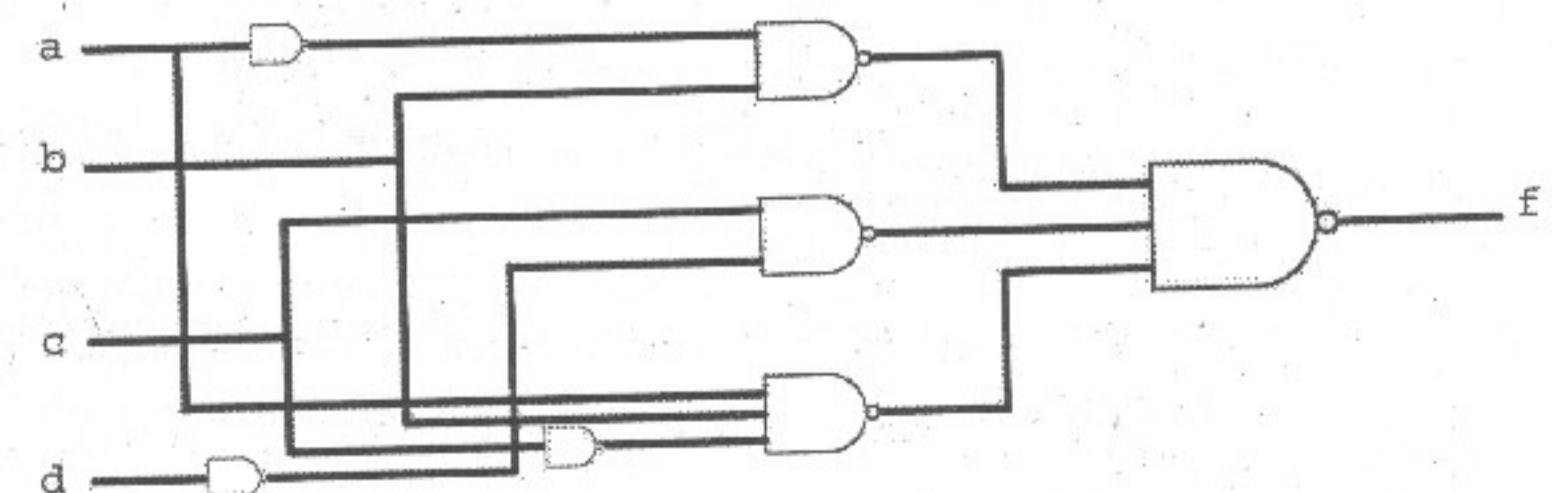
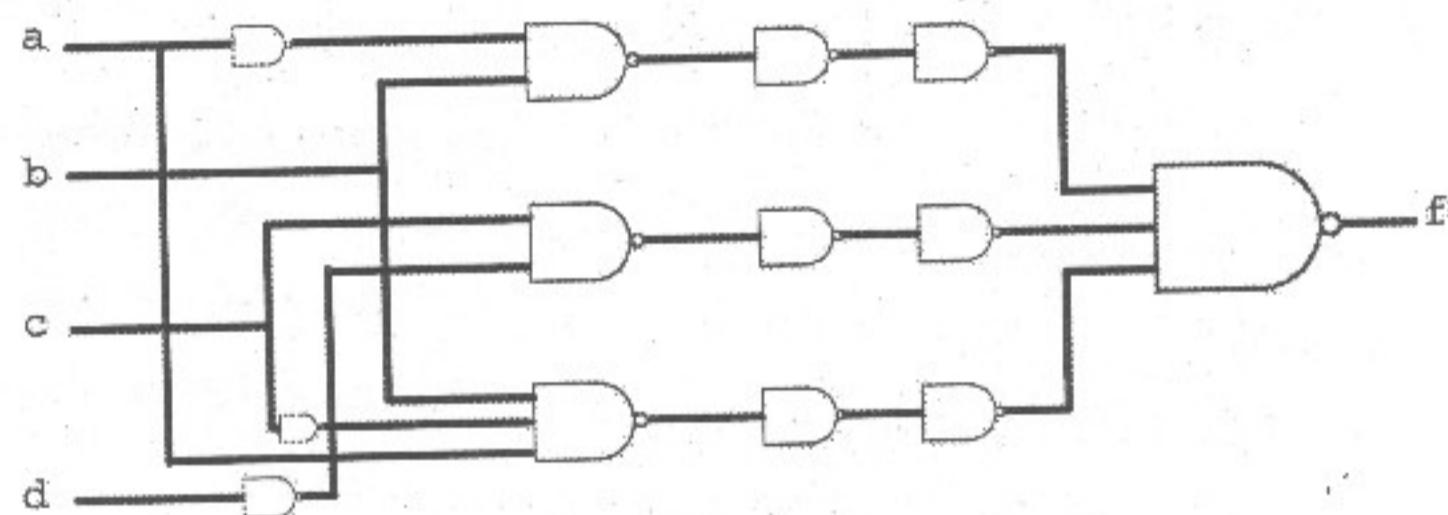
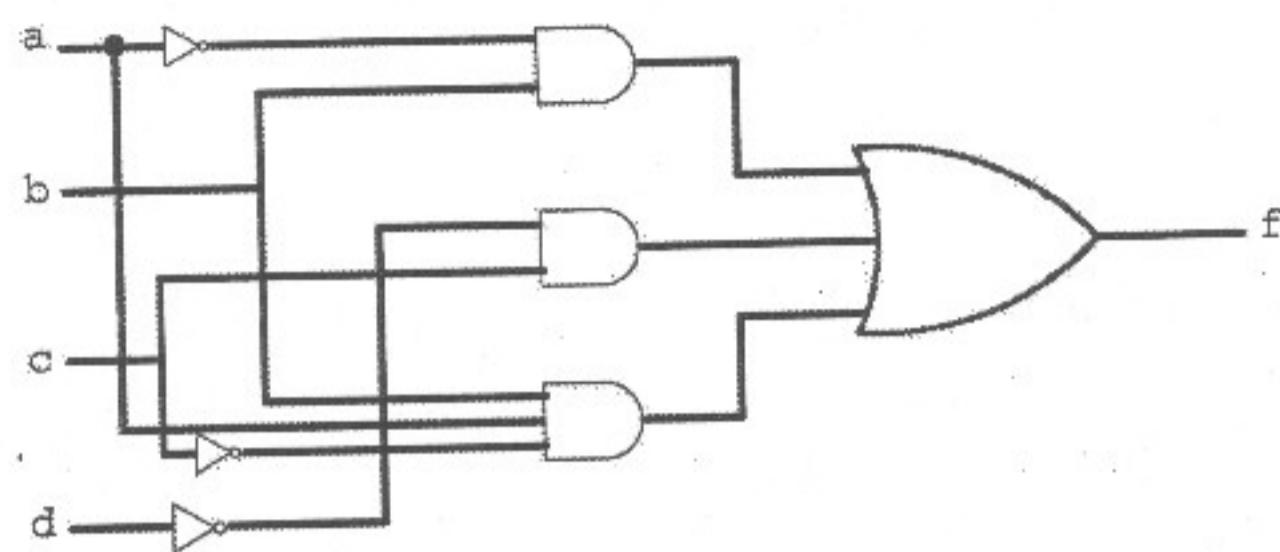
حال اگر فرض کنیم تابع را بخواهیم به کمک NAND پیاده سازی کنیم خواهیم داشت:

ملاحظه می شود که می توان تابع فوق را به کمک دو گیت NAND پیاده سازی کرد. بنابراین لازم است یک IC از نوع NAND خریداری کرد و از نیمی از آن استفاده کرد.

می بینیم که با طراحی جدید هم پیچیدگی مدار پایین آمده و هم سخت افزار کمتری استفاده شده است. دلیل دوم استفاده از گیتهای NAND و NOR این است که این گیتها به تهایی قادر به پیاده سازی تمامی مدارات ترکیبی می باشند به این معنی که این گیتها از نظر عملیاتی کامل هستند.

NAND و NOR از نظر عملیاتی کامل هستند به این معنی که تمام مدارها را می توان به کمک NAND و NOR پیاده سازی کرد. ●

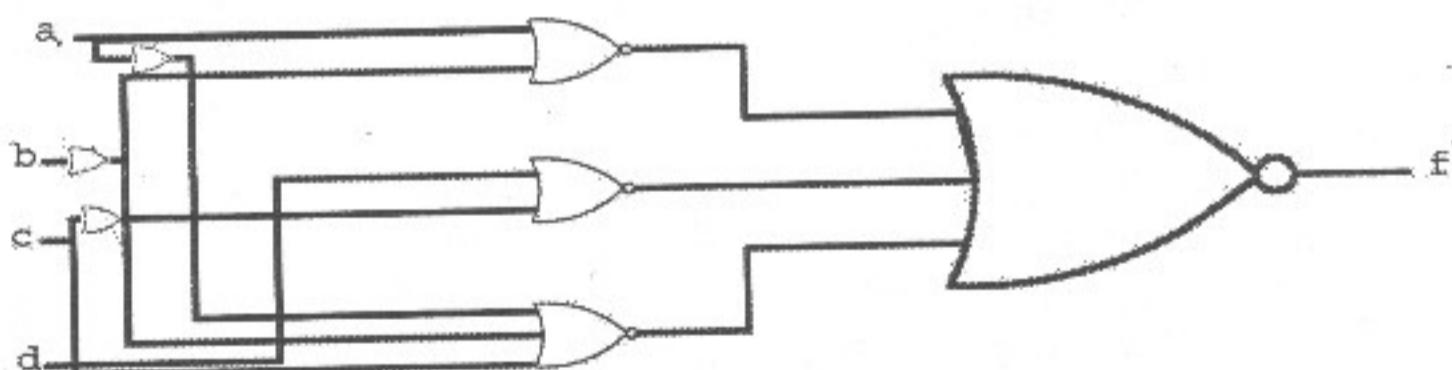
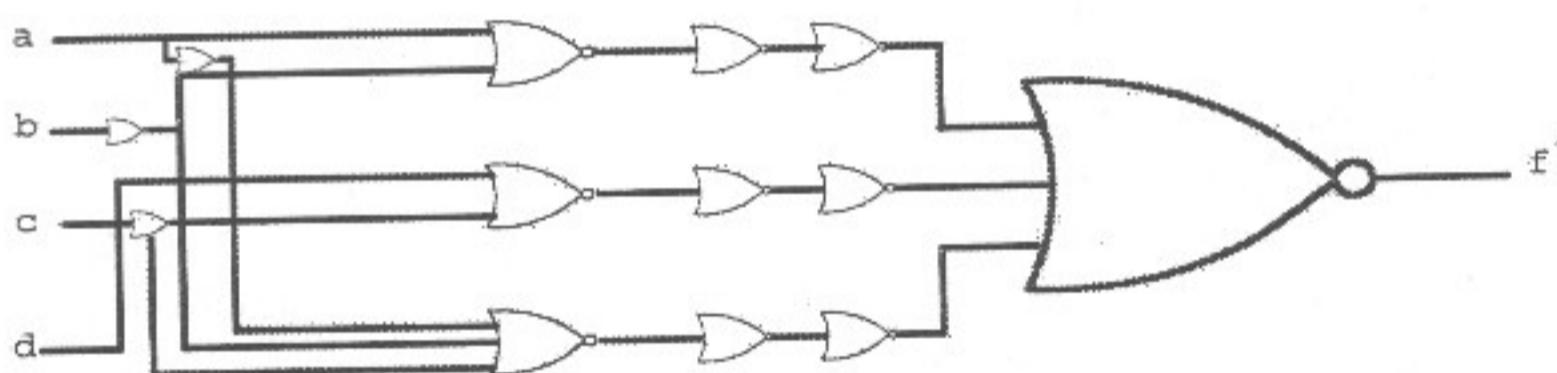
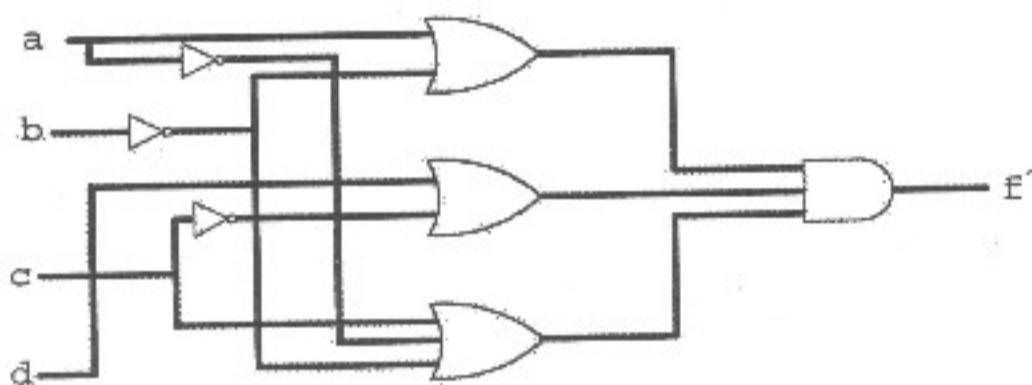
مثال:

 تابع $F = A'B + CD' + ABC'$ را به صورت تمام NAND پیاده سازی کنید.


مثال:

تابع $F = A'B + CD' + ABC'$ در نظر گرفته و F' را به صورت تمام NOR پیاده سازی کنید.

* این را در نظر داشته باشید گاهی پیاده سازی F' را حتی و به صرفه تر است زیرا ممکن است از گیت های کمتری استفاده کنیم و در انتها می توان با قرار دادن یک Not جلوی F' آن را به F تبدیل کرد.



مثال:

توابع بولی زیر را به صورت ضرب حاصل جمع ها ساده کنید.

$$\text{الف) } F(w, x, y, z) = \sum(0, 2, 5, 6, 7, 8, 10)$$

	yz wx	00	01	11	10
00	1				1
01		1	1		1
11					
10	1				1

$$F(w, x, y, z) = x'z' + w'xz + w'yz'$$

$$F(w, x, y, z) = (x+z)(w+x'+z')(w+y'+z)$$

$$\text{ب) } F(A, B, C, D) = \prod(1, 3, 5, 7, 13, 15)$$

	CD AB	00	01	11	10
00			1	1	
01			1	1	
11			1	1	
10					

$$F(A, B, C, D) = BD + A'B'D \Rightarrow D(B + A'B')$$

$$F(A, B, C, D) = (B + A')D$$

$$F(A, B, C, D) = (B + A')(B + B')D$$

$$F(A, B, C, D) = B'A + D'$$

مثال:

عبارت زیر را ساده کنید و آن را با مدارهای دو طبقه گیت NAND پیاده‌سازی کنید.

$$F(ABCD) = AB' + ABD + ABD' + A'C'D' + A'BC'$$

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

$$ABD(C+C') = ABCD + ABDC'$$

$$ABD'(C+C') = ABD'C + ABD'C'$$

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

$$A'BC'(D+D') = A'BC'D + A'BC'D'$$

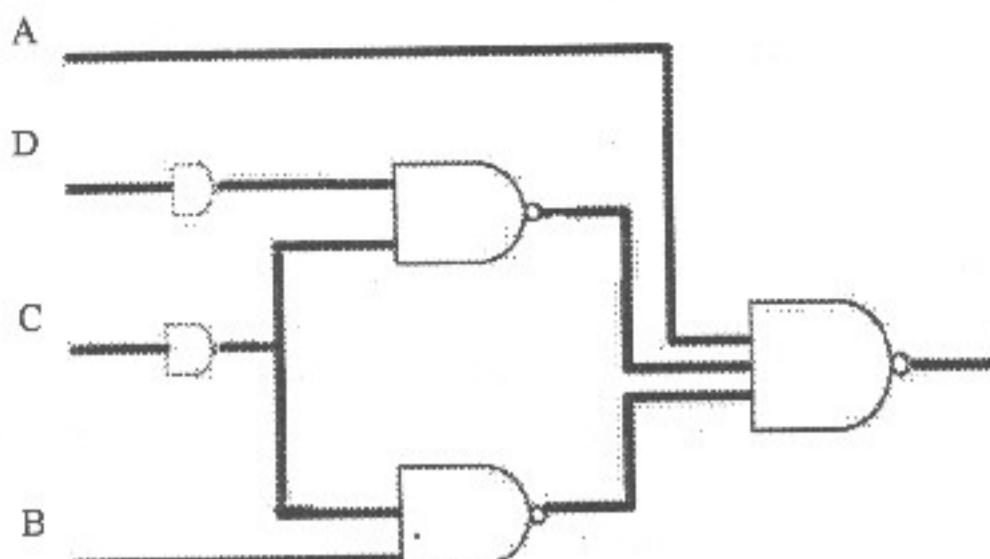
$$F(A, B, C, D) = AB'CD + AB'CD' + AB'C'D + AB'C'D' + ABCD + ABC'D$$

$$\begin{array}{ccccccccc}
 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\
 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\
 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\
 + & & & & & & & & \\
 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\
 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1
 \end{array}$$

	CD AB	00	01	11	10
00		1			
01		1	1		
11		1	1	1	1
10		1	1	1	1

$$F = A + BC' + C'D'$$

مدار منطقی



مثال:

یک نمودار گیت NAND رسم کنید به نحوی که تابع زیر را پیاده‌سازی کرده آن را متمم نماید.

$$F(A, B, C, D) = \sum(0, 1, 2, 3, 4, 8, 9, 12)$$

		CD	00	01	11	10
		AB	00	1	1	1
		00	1			
		01				
		11				
		10	1	1		

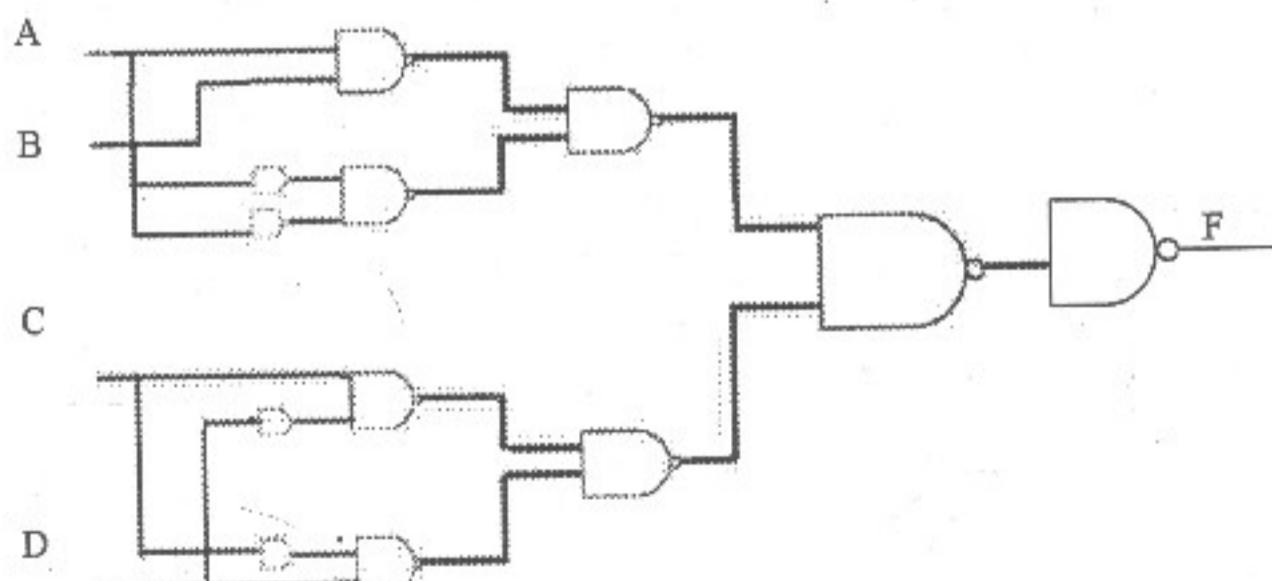
$$F = C'D' + AB' + B'C'$$

$$F' = (C+D)(A+B)(B+C)$$

مثال:

یک نمودار منطقی با استفاده از گیت‌های NAND دو ورودی برای پیاده‌سازی عبارت زیر رسم کنید.

$$(AB + A'B')(CD' + C'D)$$





مثال:

تابع بول F را همراه با حالات بی اهمیت d با کمتر از دو گیت NOR پیاده سازی کنید.

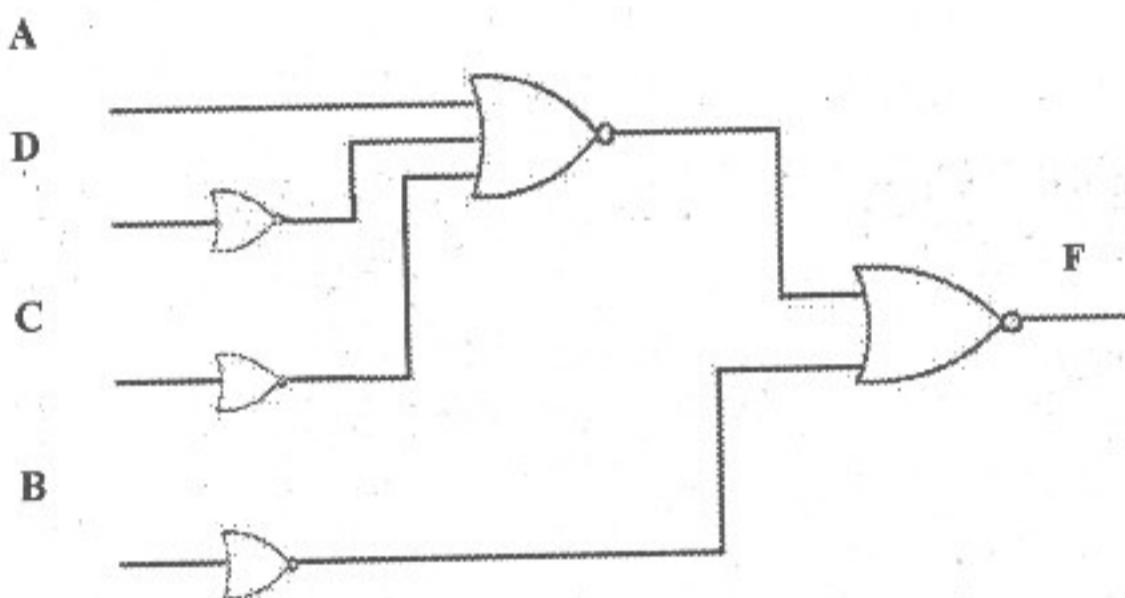
$$F(A, B, C, D) = \sum (0, 1, 2, 9, 11)$$

$$d(A, B, C, D) = \sum (8, 10, 14, 15)$$

		CD	00	01	11	10
		AB	1	1		1
		00				
		01				
		11		X	X	
		10	X	1	1	X

$$F = AB' + B'C' + B'D'$$

$$F = B'(A + C' + D')$$



مثال:

با استفاده از نقشه چهار متغیره توابع زیر را ساده نمایند.

$$F(A, B, C, D) = \sum (0, 2, 4, 5, 6, 7, 8, 10, 13, 15)$$

$$F = BD + A'B + B'D'$$

		CD	00	01	11	10
		AB	1			1
		00	1	1	1	1
		01	1	1	1	1
		11		1	1	
		10	1			1

$$F = AB' + B'C' + B'D'$$

$$F = B'(A + C' + D')$$

مدار منطقی

مجموعه مسائل فصل دوم:

۱- کارنوهای زیر را ساده کنید.

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

$$f = bd$$

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

$$f = bd + b'd'$$

ab	cd	00	01	11	10
00	1			1	2
01	4	1 5	1 7	6	
11	12	13	15	14	
10	1	8	9	11	10

$$f = bd + b'd' + a'b'c + ac'd$$

مدار منطقی



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

$$F = w'$$

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

$$f = w' + x'y + xy'$$

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

$$f = x'yw' + x'yz + xyw + xwz + x'y'wz'$$

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

$$f = bc'd + b'c + b'd' + a'cd'$$

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

$$F = xz' + xy + x'y'z$$

مدار منطقی

-۲- به کمک جدول کارنوی زیر تابع $f(x,y,w,z) = \pi(0,2,4,6,8,10,12,14)$ را ساده کنید.

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

$$f = z$$

$$f(x,y,w,z) = \pi(0,2,4,6,8,10,12,14) = \Sigma(1,3,5,7,9,11,13,15)$$

-۳- جدول کارنوی زیر را به کمک Maxterm ساده کنید.

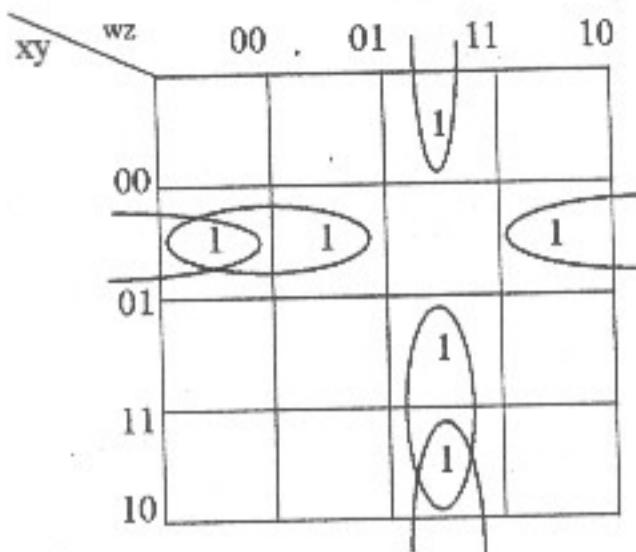
xy\wz	00	01	11	10
00	0	1	3	2
01	0 4	0 5	0 7	0 6
11	X	0	15	14
10	12	13	11	10

$$f(x,y,w,z) = (x+y')(y'+z')(y+w'+z)$$

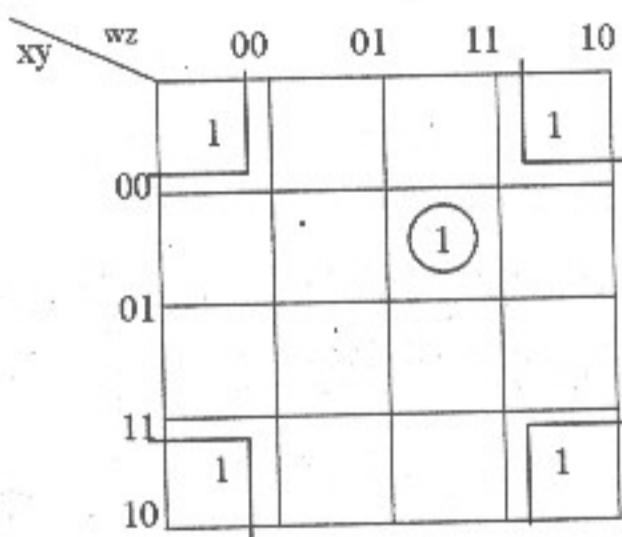
-۴- جداول کارنوی زیر را همسایه‌گیری کنید.

xy\wz	00	01	11	10
00	0	1	3	2
01	1 2	1 5	1 7	1 6
11	12	13	15	14
10	8	9	11	10

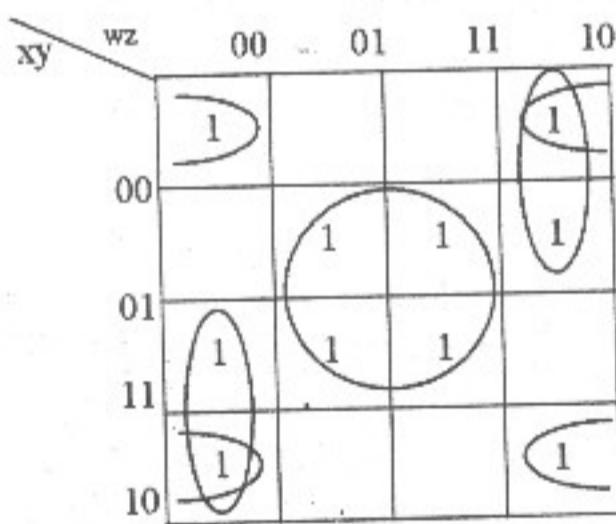
$$F = x'y + xwz$$



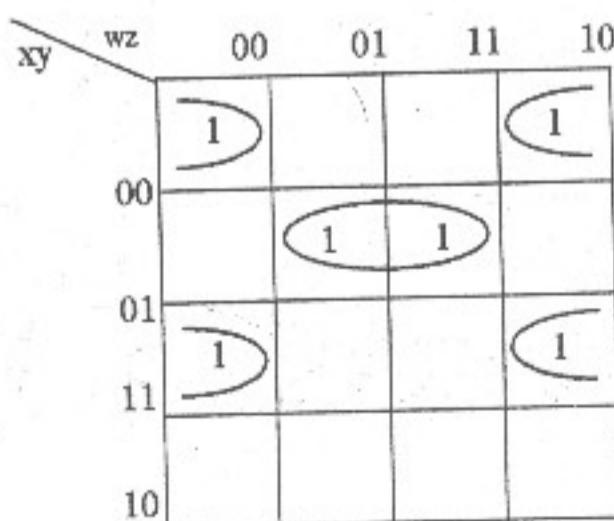
$$F = xwz + x'yw' + x'yz' + y'wz$$



$$F = y'z' + x'ywz$$



$$F = yz + y'z' + x'wz' + xw'z'$$



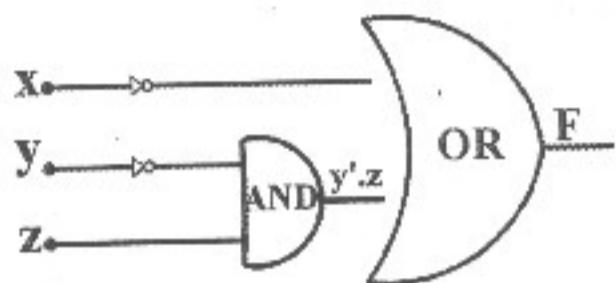
$$F = x'y'z' + x'yz + xyz'$$

مدار منطقی

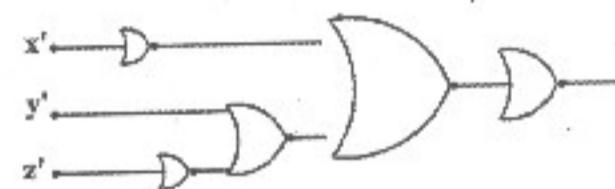
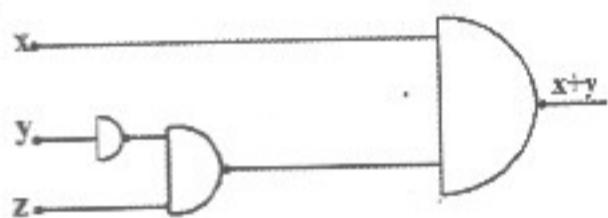
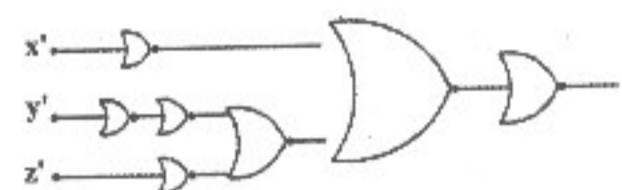
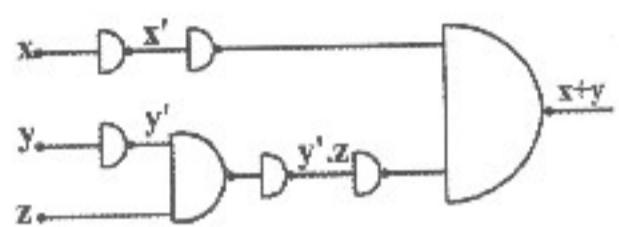
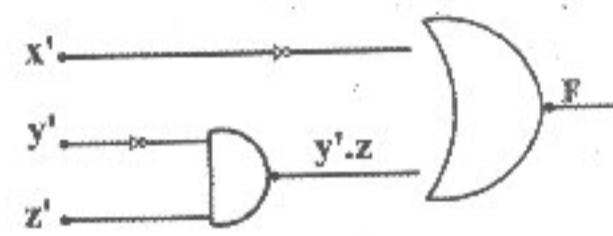
- تابع زیر را به صورت تمام NAND و تمام NOR پیاده سازی کنید.

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

تمام NAND



تمام NOR

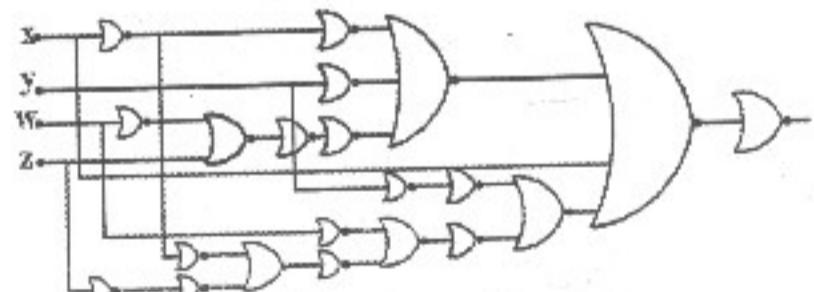
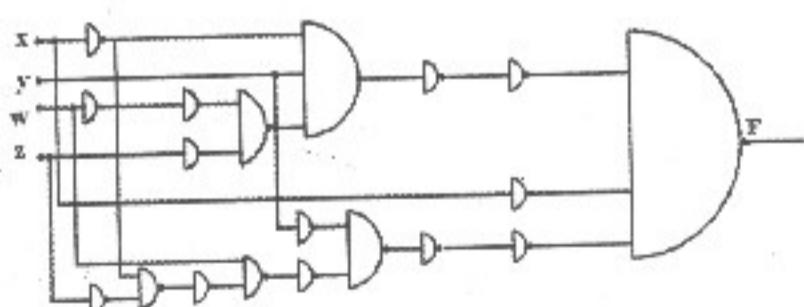
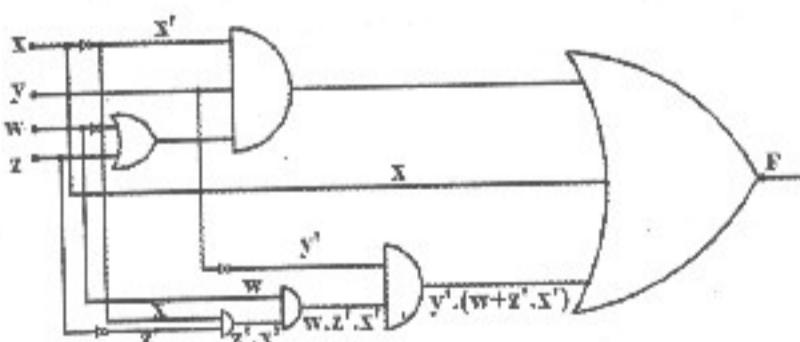


- تابع زیر را بصورت تمام NAND پیاده سازی کنید و همچنین پیاده سازی تمام NOR.

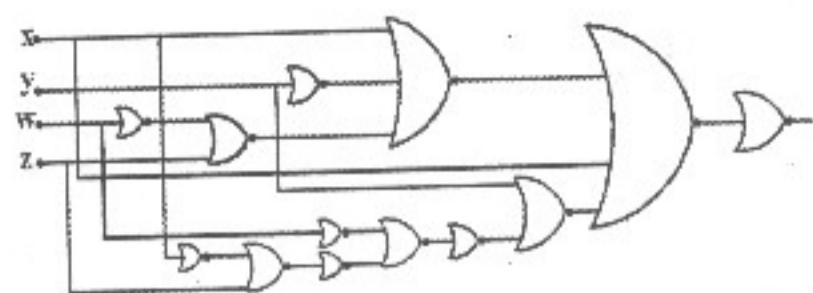
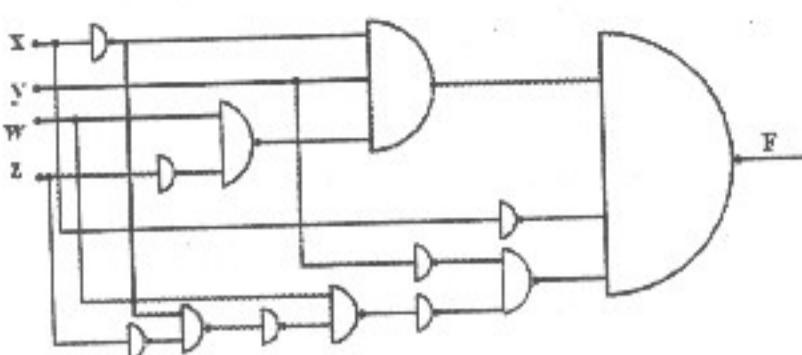
$$F = x'y(\bar{w} + z) + x + y'(\bar{w} \cdot \bar{z}' \cdot x')$$

تمام NAND

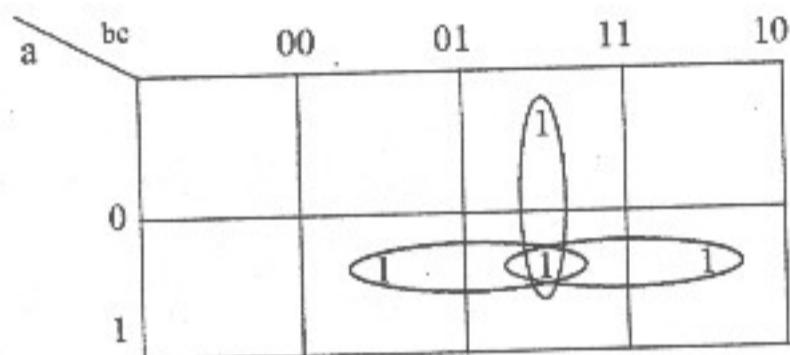
تمام NOR



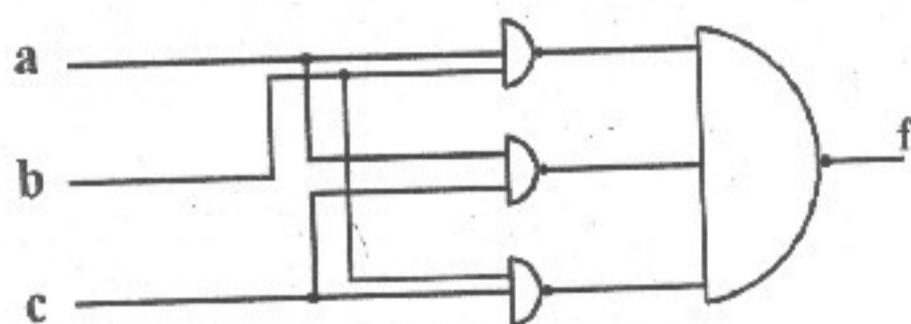
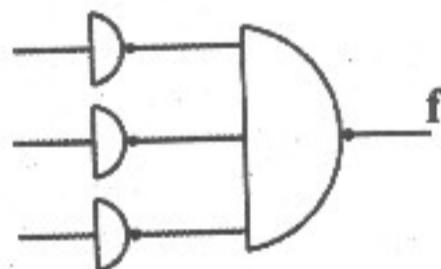
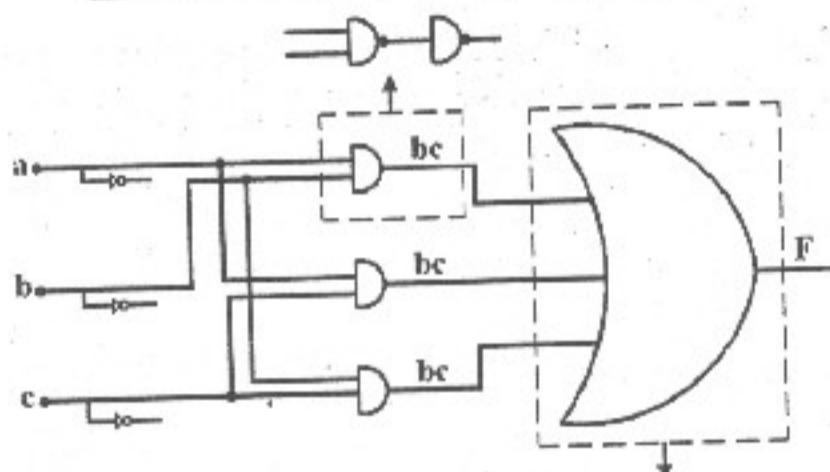
مدار منطقی



-۷- می خواهیم مدار مربوط به میز رای گیری سه نفره را پیاده سازی تمام NAND انجام دهیم.



$$F = a \cdot c + a \cdot b + b \cdot c$$

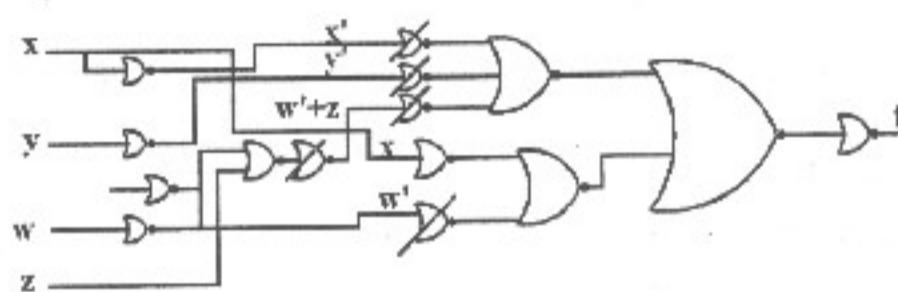
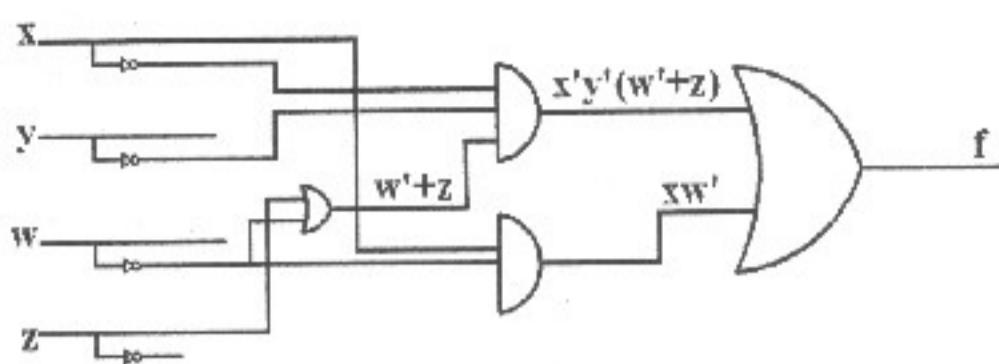


هر جا دو NOR یا NAND یک ورودی پشت سرهم قرار گیرند به جای هر دو سیم قرار می دهیم.

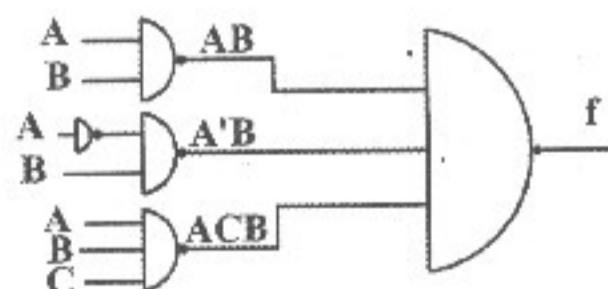
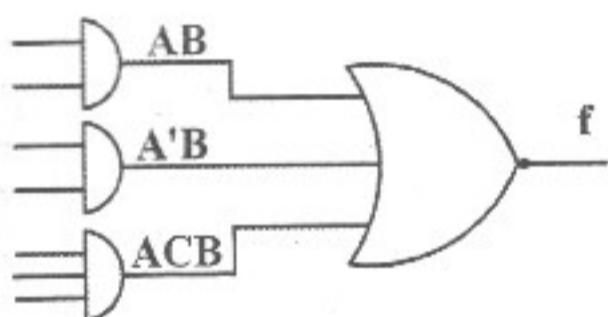
-۸- می خواهیم تابع زیر را بصورت تمام NOR پیاده سازی کنیم.

$$F = x' \cdot y' \cdot (w' + z) + x \cdot w'$$

مدار منطقی



-۹- تابع $F = AB + A'CD + A'B'$ را به صورت تمام NAND پیاده‌سازی کنید.



-۱۰- مداری طراحی کنید که ورودی آن یک رقم BCD و خروجی آن معادل gray ورودی باشد.

A	B	C	D	x	y	w	z
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1

مدار منطقی

$$X = AB'C'$$

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

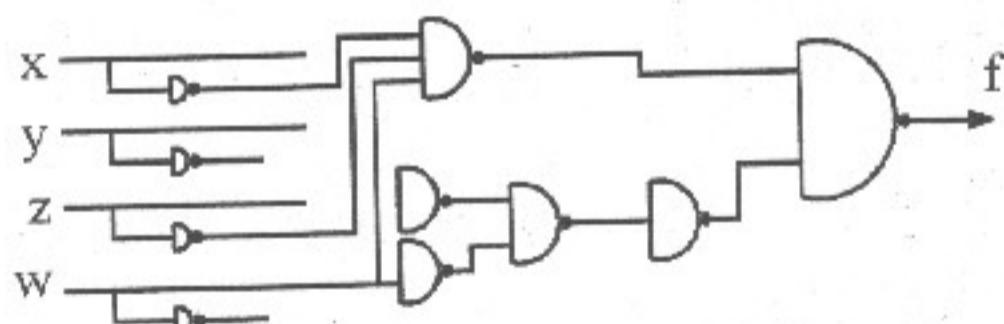
$$y = B + A$$

	AB	CD	00	01	11	10
00			0	1	1	2
01	1	4	1	5	3	6
11	X	X	X 0	X 0		
10	12	13	15	14		
11			X 11	X 10		
10	8	9	11	10		

$$w = B'C + BC' = B \oplus C$$

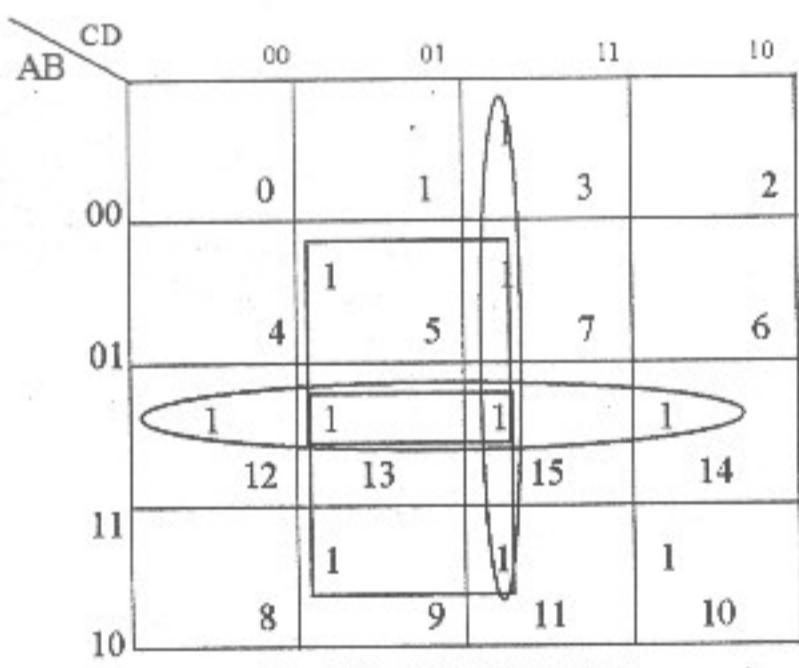
	AB	CD	00	01	11	10
00			1	1	3	2
01			0	1	5	6
11	X 0	X 0	X 13	X 15	X 0	X 14
10	12	13	15	14	X 0	X 10
11			1	9	X 0	X
10	8	9	11	10		

$$Z = C'D + CD' = C \oplus D$$

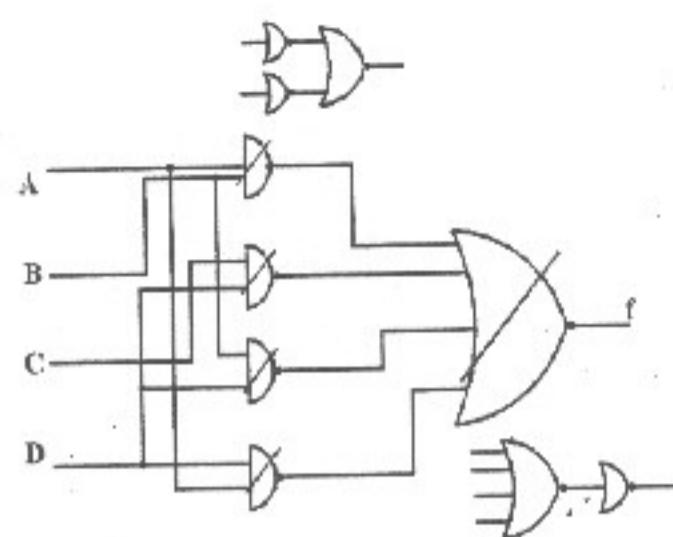
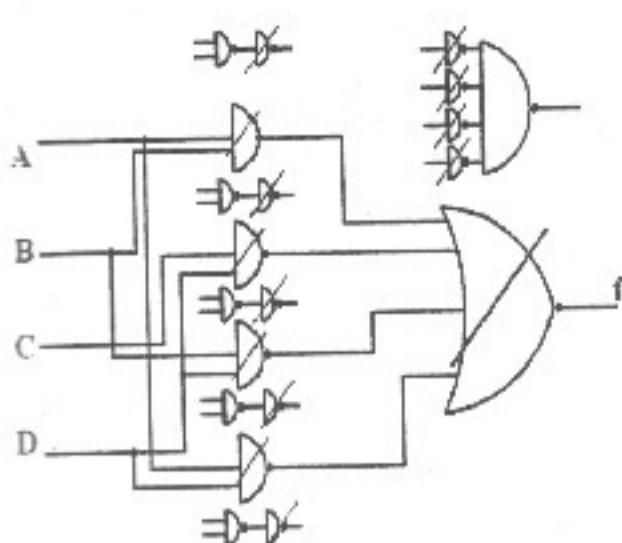


۱۱- مداری طراحی کنید که یک میز رأی گیری چهارنفره را شیوه‌سازی کند. به صورتی که نفر اول دارای سه حق رأی، نفر دوم دارای چهار حق رأی، نفر سوم دو حق رأی، نفر چهارم دارای پنج حق رأی باشد. خروجی مدار در زمانی یک است که مجموع حق رأی‌ها ۷ یا بیشتر باشد.

	^۳ A	^۴ A	^۲ C	^۵ D	F
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	1



$$F = AB + CD + BD + AD$$



۱۲- تابع F را به صورت maxterm و minterm بنویسید:

	a	b	c	d	F
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	0
A	1	0	1	0	0
B	1	0	1	1	1
C	1	1	0	0	1
D	1	1	0	1	0
E	1	1	1	0	1
F	1	1	1	1	1

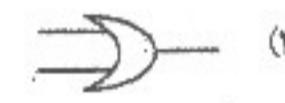
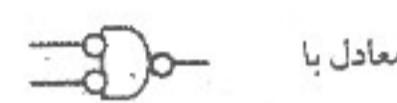
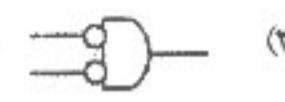
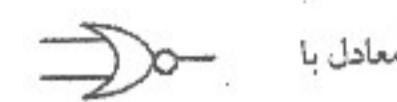
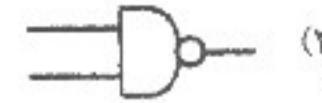
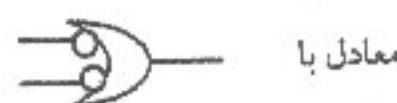
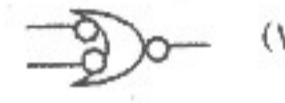
$$F(a, b, c, d) = \sum(1, 4, 7, 11, 12, 14, 15) = a'b'c'd + a'bc'd' + a'bcd + ab'cd + abc'd' + abcd' + abcd$$

$$F(a, b, c, d) = \prod(0, 2, 3, 5, 6, 8, 9, 10, 13) = (a+b+c+d).(a+b+c'+d).(a+b+c'+d').(a+b'+c+d').$$

$$(a+b'+c'+d).(a'+b+c+d).(a'+b+c+d').(a'+b'+c+d')$$

مجموعه تست‌های فصل دوم:

۱- در کدامیک از گزینه‌های زیر، دروازه‌های منطقی معادل نیستند؟



۲- کدامیک از گزینه‌های زیر قوانین حاکم بر مدارهای منطقی را نقض می‌کنند؟

$$A \cdot B + A' \cdot C + B \cdot C = A \cdot B + A' \cdot C \quad (۱)$$

$$A' + A = 1 \quad (۲)$$

$$A + A' \cdot B = A + B \cdot C \quad (۳)$$

$$A \cdot B + A \cdot C = A + B \cdot C \quad (۴)$$

۳- در کدامیک از گزینه‌های زیر ساده‌ترین رابطه برای پیاده کردن جدول کارنو مقابله، در قالب یک مدار منطقی را نشان می‌دهد؟

AB	CD	00	01	11	10
00	1				
01	1	1	1	1	
11		1	1		
10		1	1		

$$A' \cdot B + B \cdot D + B' \cdot C' \cdot D' + A \cdot C' \cdot D \quad (۱)$$

$$A' \cdot B + B \cdot D + A' \cdot C' \cdot D' + A \cdot B' \cdot C' \quad (۲)$$

$$A' \cdot B + B \cdot D + B' \cdot C' \cdot D' + A \cdot B' \cdot C' \quad (۳)$$

$$A' \cdot B + B \cdot D + C' \cdot D' + B' \cdot C' \quad (۴)$$

۴- ساده شده تابع $F = A(B + \bar{C}A) + (BC)(B + \bar{C}A)$ کدام است؟

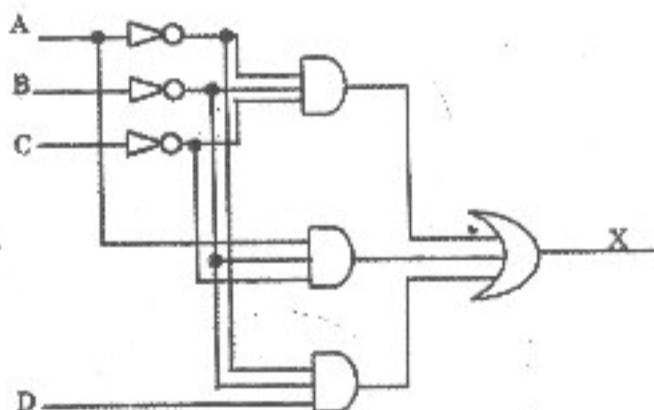
$$AB + A\bar{C} + \bar{B}C \quad (۱)$$

$$\bar{A}\bar{B} + \bar{A}C + \bar{B}C \quad (۲)$$

$$\bar{A}\bar{B} + A\bar{C} + BC \quad (۳)$$

$$AB + A\bar{C} + BC \quad (۴)$$

۵- خروجی مدار شکل مقابل کدام است؟



$$\bar{ABC} + A\bar{BC} + \bar{ABC} \quad (۱)$$

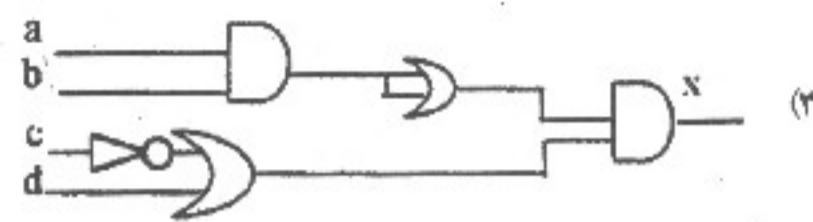
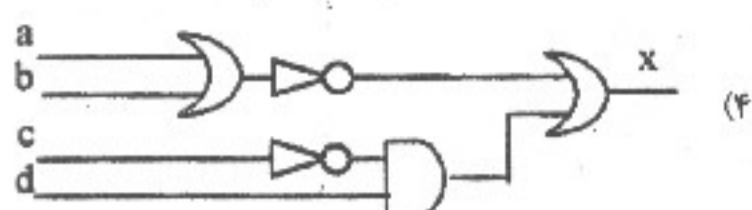
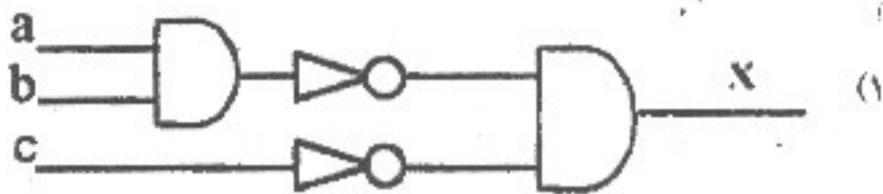
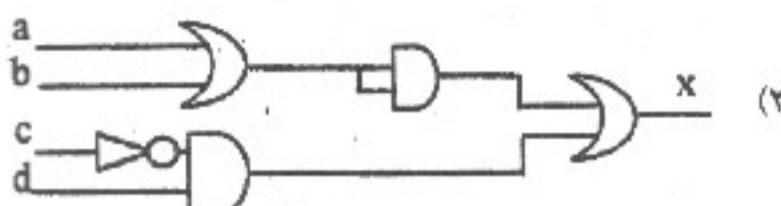
$$\bar{ABC} + \bar{ABC} + \bar{ABD} \quad (۲)$$

$$\bar{ABC} + A\bar{BC} + \bar{ABD} \quad (۳)$$

$$ABC + \bar{ABC} + \bar{ABD} \quad (۴)$$

مدار منطقی

۱۵- مدار معادل رابطه $X = (\overline{a+b} + \overline{c}d)$ کدام می‌باشد؟



۱۶- برای تابع $f(A, B, C, D, E) = \sum m(1, 4, 9, 11, 13, 15, 17, 19, 22, 25, 27, 29, 30, 31) + \sum d(3, 12, 20)$ حالت بی اهمیت است) کدام گزینه ساده شده زیر صحیح است:

$$f(A, B, C, D, E) = \overline{B}\overline{E} + \overline{C}E + \overline{A}\overline{C}\overline{D}\overline{E} + A\overline{C}D\overline{E} \quad (1)$$

$$f(A, B, C, D, E) = BE + \overline{C}E + \overline{A}\overline{C}\overline{D}\overline{E} + A\overline{C}D\overline{E} \quad (2)$$

$$f(A, B, C, D, E) = BE + \overline{C}E + \overline{A}\overline{C}\overline{D}\overline{E} + \overline{A}\overline{C}D\overline{E} \quad (3)$$

$$f(A, B, C, D, E) = \overline{B}\overline{E} + \overline{C}E + A\overline{C}\overline{D}\overline{E} + A\overline{C}DE \quad (4)$$

۱۷- برای یک مدار مبدل که جدول درستی آن در زیر داده شده است برای خروجی X کدام پاسخ صحیح است؟

D	F	E	D	C	B	A	Z	J	X
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	1
0	0	0	0	0	1	1	0	1	0
0	0	0	0	1	1	1	0	1	1
0	0	0	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1

$$x = A \oplus B \oplus C \oplus D \oplus E \oplus F \oplus G \quad (1)$$

$$x = B \oplus D \oplus C \quad (2)$$

$$x = (A \odot B \odot C) \quad (3)$$

$$X = A \oplus B \odot C \oplus D \quad (4)$$

۱۸- کدامیک از حاصلضرب maxterm ها نمایش Standard تابع زیر می‌باشد.

$$f(W, X, Y, Z) = XY + \overline{W}Y + WXY$$

$$\Pi M(0, 1, 8, 9, 10, 11) \quad (1)$$

$$\Pi M(0, 1, 2, 3, 7, 12, 13, 15) \quad (2)$$

$$\Pi M(1, 3, 4, 5, 7, 12, 13, 14, 15) \quad (3)$$

$$\Pi M(0, 1, 2, 6, 8, 9, 11) \quad (4)$$

مدار منطقی

-۱۹- برای تابع بولی ۵ متغیره زیر ساده‌ترین صورت ضرب حاصل جمع‌ها کدام است؟

$$F(A, B, C, D, E) = \prod(0, 8, 9, 10, 12, 16, 17, 25, 29), d(2, 7, 13, 23)$$

$$(A + \bar{B})(A + \bar{C})(A + \bar{D} + E)(\bar{B} + \bar{C} + E)(B + \bar{C} + \bar{E}) \quad (1)$$

$$(A + C + E)(A + \bar{B} + D)(\bar{A} + B + C + D)(\bar{B} + D + \bar{E}) \quad (2)$$

$$(D + B + E)(A + C + \bar{D} + \bar{E})(\bar{A} + C + E)(\bar{A} + C + D) \quad (3)$$

$$(A + \bar{B})(C + \bar{E})(\bar{A} + C + \bar{D} + \bar{E})(\bar{A} + C + D + \bar{E}) \quad (4)$$

-۲۰- ساده‌ترین صورت تابع مقابل کدام است؟

$$F(A, B, C, D, E) = \sum(0, 2, 6, 8, 9, 11, 13, 15, 16, 18, 22) + d(4, 10, 12, 14, 20, 26, 28, 30)$$

$$F = A'E' + AB'D'E' + A'B + ADE \quad (1)$$

$$F = A'E' + A'B + AB'E' \quad (1)$$

$$F = A'B + B'E' \quad (2)$$

$$F = A'E' + AB' \quad (3)$$

-۲۱- در تابع بولی که دارای ترمونهای زیر می‌باشد، ساده‌ترین فرم حاصل جمع حاصل‌ضرب‌ها کدام است؟

$$f(A, B, C, D, E) = \sum m(0, 2, 4, 9, 11, 12, 14, 18, 20, 21, 27, 29), d(6, 10, 16, 22, 25)$$

$$\bar{A}\bar{D}\bar{C} + A\bar{B}D + BCD + B\bar{C}\bar{D} + B\bar{C}E \quad (1)$$

$$AC\bar{D} + A\bar{B}C + \bar{A}\bar{D}\bar{C} + B\bar{C}\bar{E} \quad (2)$$

$$ABC + AB\bar{D} + BCD + BCE + ABE \quad (3)$$

$$\bar{B}\bar{E} + B\bar{C}E + \bar{A}\bar{C}\bar{E} + AC\bar{D}E \quad (4)$$

-۲۲- برای تابع بولی ۵ متغیره زیر ساده‌ترین صورت حاصل جمع حاصل‌ضرب‌ها کدام است؟

$$F(A, B, C, D, E) = \sum(0, 3, 8, 14, 15, 16, 18, 24, 26, 27, 29), D(6, 7, 9, 19, 22)$$

$$\bar{C}\bar{D}\bar{E} + A\bar{B}C + \bar{A}\bar{B}\bar{C}E + ABCDE \quad (1)$$

$$\bar{A}CD + \bar{C}\bar{D}E + \bar{B}\bar{C}D + \bar{A}\bar{B}DE + ABC\bar{D}E \quad (2)$$

$$\bar{A}CD + \bar{C}\bar{D}\bar{E} + A\bar{C}D + \bar{A}\bar{B}\bar{C}E + ABC\bar{D}E \quad (3)$$

$$\bar{A}CD + A\bar{C} + \bar{A}\bar{B}\bar{C}\bar{E} + ABCD\bar{E} + \bar{C}\bar{D}E \quad (4)$$

-۲۳- تابع $F = \bar{A}\bar{B}C + A\bar{B}\bar{C} + ABC$ بر حسب حاصل ضرب مجموعه‌ها کدام است؟

$$(A + B + C)(A + \bar{B} + C)(A + \bar{B} + \bar{C})(\bar{A} + B + \bar{C})(\bar{A} + \bar{B} + C) \quad (1)$$

$$(\bar{A} + \bar{B} + \bar{C})(\bar{A} + B + C)(A + \bar{B} + C)(A + \bar{B} + \bar{C})(\bar{A} + B + \bar{C}) \quad (2)$$

$$(\bar{A} + \bar{B} + \bar{C})(A + \bar{B} + C)(A + \bar{B} + \bar{C})(\bar{A} + B + \bar{C})(\bar{A} + \bar{B} + C) \quad (3)$$

$$(A + B + C)(\bar{A} + B + C)(A + \bar{B} + \bar{C})(\bar{A} + B + \bar{C})(\bar{A} + \bar{B} + C) \quad (4)$$

- ۲۴- صورت مینیمم تابع زیر کدام است؟

$$F(X_1, X_2, X_3) = \bar{X}_1 \bar{X}_2 \bar{X}_3 + X_1 \bar{X}_2 \bar{X}_3$$

$$\bar{X}_1 + X_1 \quad (۱)$$

$$X_1 \bar{X}_2 \bar{X}_3 \quad (۲)$$

$$\bar{X}_2 \bar{X}_3 \quad (۳)$$

$$\bar{X}_1 \bar{X}_2 \bar{X}_3 \quad (۴)$$

- ۲۵- در منطق منفی هر دروازه OR به چه دروازه منطقی در منطق مثبت تبدیل می شود؟

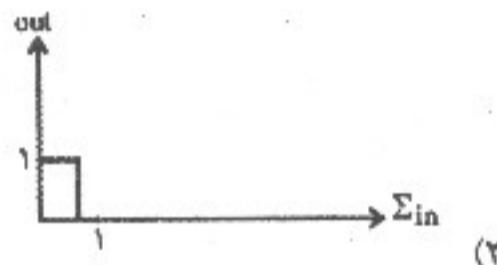
$$\text{XNOR} \quad (۱)$$

$$\text{NAND} \quad (۲)$$

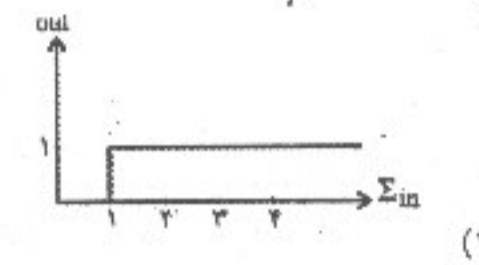
$$\text{NOR} \quad (۳)$$

$$\text{AND} \quad (۴)$$

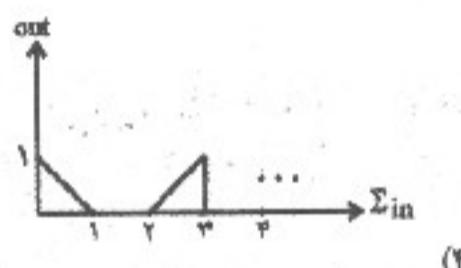
- ۲۶- کدامیک از اشکال زیر یک XOR را مشخص می سازد؟ (Σ_{in} جمع ورودی ها را نشان می دهد و out خروجی مدار است)



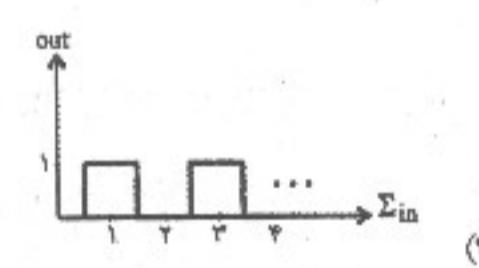
(۲)



(۱)

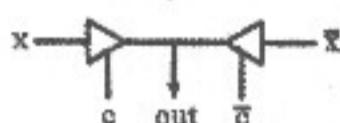


(۴)



(۳)

- ۲۷- کدام عملیات مربوط به مدار زیر است؟



$$\bar{C}\bar{X} \quad (۱)$$

$$CX \quad (۲)$$

$$\overline{C \oplus X} \quad (۳)$$

$$C \oplus X \quad (۴)$$

- ۲۸- ساده شده جدول کارنوی زیر کدام است؟

	AB	00	01	11	10
CD	00	0	0	0	0
	01	0	1	1	1
	11	d	d	d	d
	10	1	1	d	d

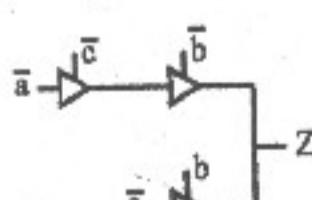
$$A' + AD + BC \quad (۱)$$

$$A' + AD' + BD' \quad (۲)$$

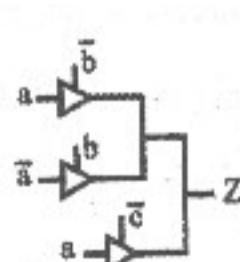
$$C + AD' + BD \quad (۳)$$

$$C + AD + BD \quad (۴)$$

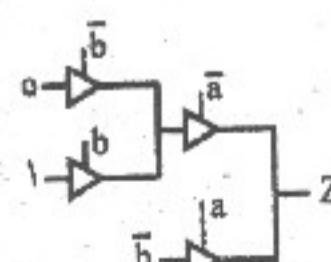
- ۲۹- با استفاده از گیت های سه حالته کدام مدار دارای خروجی $z(a \oplus b) + \bar{a}c$ می باشد؟



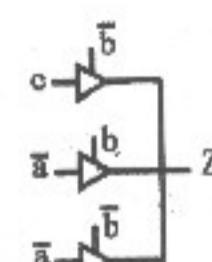
(۴)



(۳)



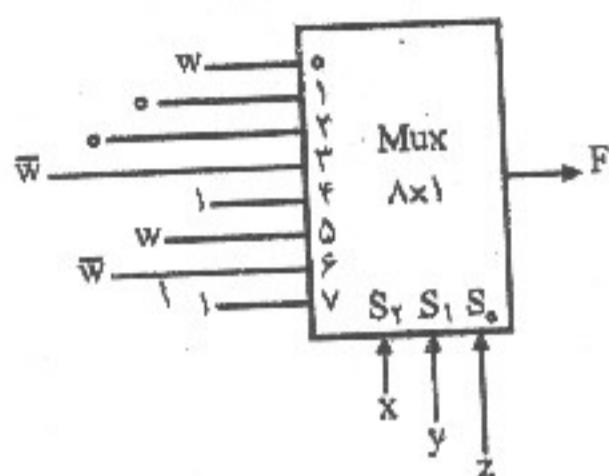
(۲)



(۱)

مدار منطقی

۳۰- مدار زیر، پیاده‌سازی کدامیک از روابط زیر است؟



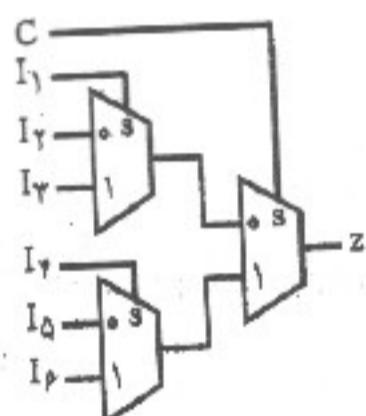
$$F = (w, x, y, z) = \sum(0, 1, 3, 5, 7, 11, 14) \quad (1)$$

$$F = (w, x, y, z) = \sum(1, 3, 6, 7, 8, 11, 12) \quad (2)$$

$$F = (w, x, y, z) = \sum(3, 4, 6, 7, 8, 12, 13, 15) \quad (3)$$

$$F = (w, x, y, z) = \sum(3, 5, 7, 9, 11, 13, 15) \quad (4)$$

۳۱- در شکل زیر با ارتباط دادن ورودی‌های a و b به خطوط $I_1, I_2, I_3, I_4, I_5, I_6$ مدار را به گونه‌ای بسازید که خروجی Z معادل تابع $Z = abc + a\bar{c} + b\bar{c}$ شود؟



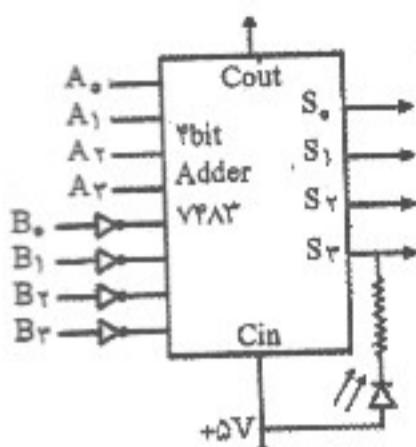
$$I_1 I_2 I_3 I_4 I_5 I_6 = ab0 \quad (1)$$

$$I_1 I_2 I_3 I_4 I_5 I_6 = ab1 \quad (2)$$

$$I_1 I_2 I_3 I_4 I_5 I_6 = abb \quad (3)$$

$$I_1 I_2 I_3 I_4 I_5 I_6 = 0lab10 \quad (4)$$

-۳۲- به فرض اینکه اعداد A و B در سیستم مکمل ۲ (2's Complement) بین ۷ - و ۷ + هستند در مدار زیر روشن شدن LED یانگر چیست؟ A و B هم علامت هستند.



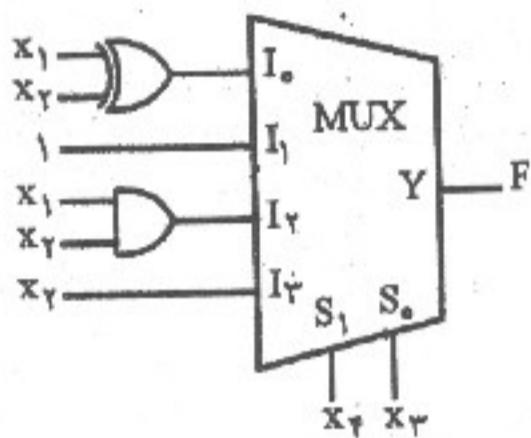
$$A \geq B \quad (4)$$

$$A = B \quad (3)$$

$$A \leq B \quad (2)$$

$$A < B \quad (1)$$

-۳۳- تابع خروجی برای مدار رویرو به چه صورت است؟



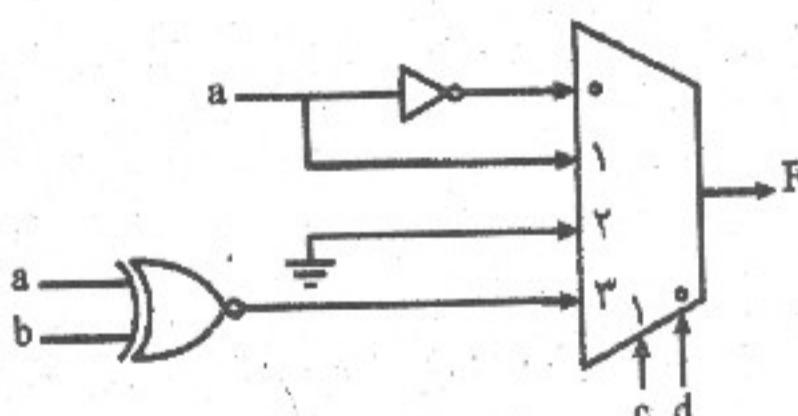
$$F(x_4, x_3, x_2, x_1) = \sum m(1, 4, 5, 9, 10, 11, 12, 13, 15) \quad (2)$$

$$F(x_4, x_3, x_2, x_1) = \sum m(1, 2, 3, 4, 5, 7, 11, 13, 15) \quad (4)$$

$$F(x_4, x_3, x_2, x_1) = \sum m(1, 2, 4, 5, 6, 7, 11, 14, 15) \quad (1)$$

$$F(x_4, x_3, x_2, x_1) = \sum m(0, 3, 4, 5, 6, 7, 10, 11, 15) \quad (3)$$

-۳۴- مدار زیر، پاده‌سازی کدام رابطه است؟



$$f(a, b, c, d) = \sum m(0, 1, 3, 5, 7) \quad (1)$$

$$f(a, b, c, d) = \sum m(1, 3, 5, 7, 11, 15) \quad (2)$$

$$f(a, b, c, d) = \sum m(0, 3, 4, 9, 13, 15) \quad (3)$$

$$f(a, b, c, d) = \sum m(0, 3, 5, 7, 13, 15) \quad (4)$$

فصل سوم

مدارهای ترکیبی

پایه‌ترین مبحث در مورد مدارهای منطقی و یا شاید هم اصلی‌ترین بحث همین مدارهای ترکیبی می‌باشد. همانطور که از اسم آنها به نظر می‌رسد این مدارات از ترکیب گیتهای متفاوت ایجاد شده‌اند که یک عملکرد خاص را انجام می‌دهند.

در واقع طراحی مدارات ترکیبی با استفاده از مطالب گفته شده در فصول یک و دو انجام می‌شود. بنابراین قبل از شروع این فصل حتماً فصول قبل به صورت کامل مطالعه شود.

الگوریتم برای طراحی مدارهای ترکیبی:

۱- بدست آوردن جدول درستی از روی صورت مسئله

۲- بدست آوردن توابع بولی از روی جدول درستی

۳- ساده کردن در صورت نیاز

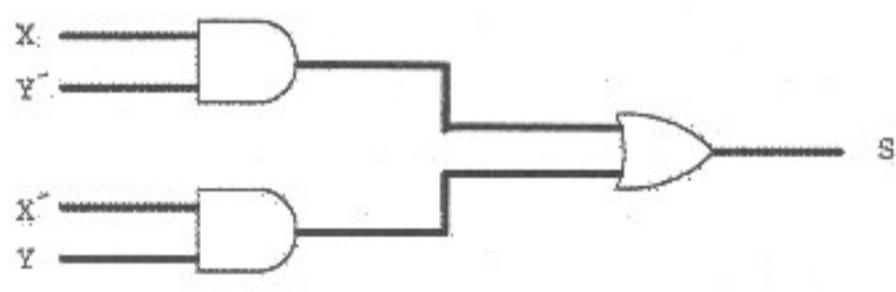
۴- پیاده‌سازی تابع ساده شده به کمک گیت‌ها

نیم جمع کننده :Half Adder

دو بیت را با هم جمع می‌کند حاصل جمع را در S و رقم نقلی را در C نشان می‌دهد یعنی مداری است با دو ورودی و دو خروجی.

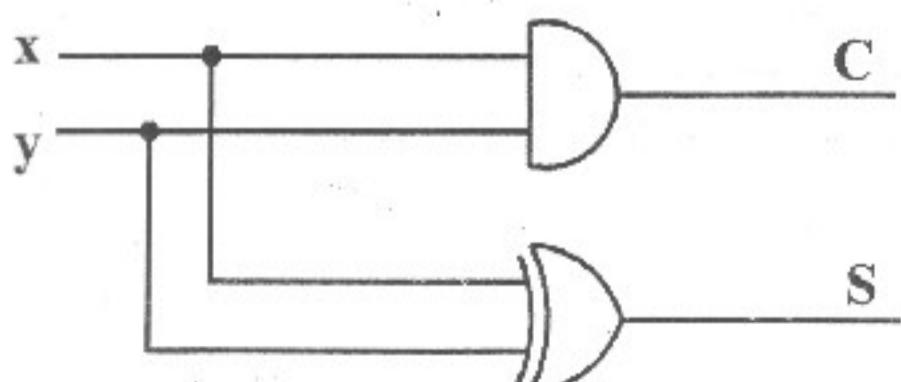
x	y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

AND XOR



$$S = x \oplus y$$

$$C = x \times y$$



تمام جمع کنندۀ Full Adder

دو بیت ورودی و یک بیت carry قبل را با هم جمع می‌کند و یک بیت به عنوان حاصل جمع (S) و یک بیت به عنوان Carry یا رقم نقلی (C) را به خروجی منتقل می‌کند. یعنی مداری است با سه ورودی و دو خروجی.

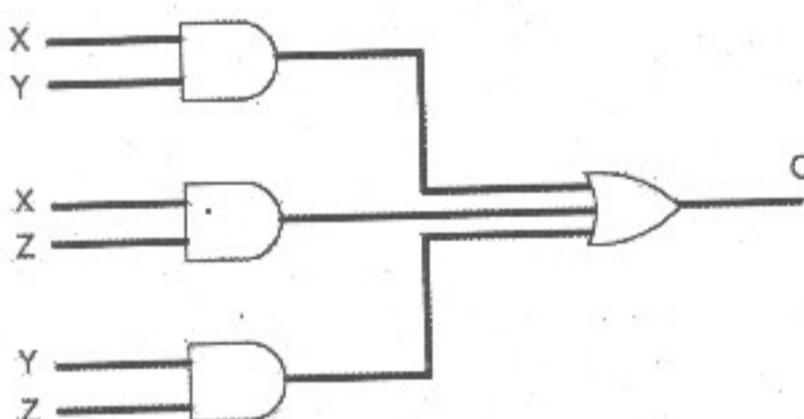
x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$C = xy + xz + yz$$

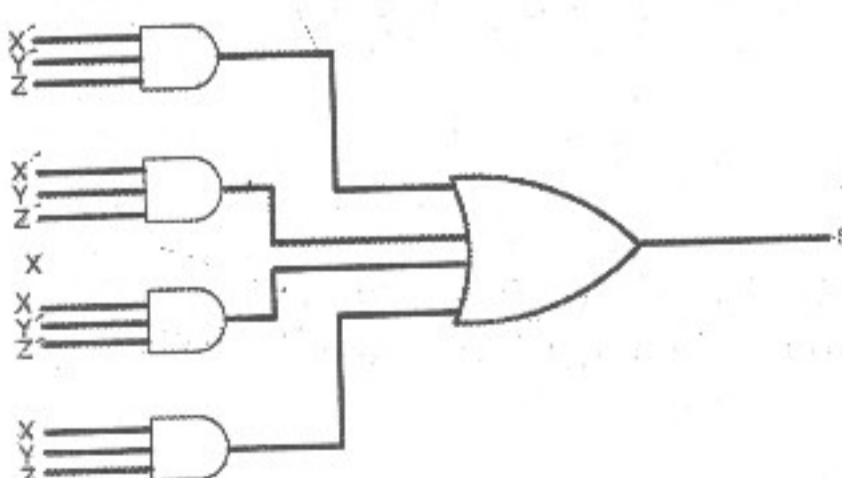
$$S = x'y'z + x'yz' + xy'z' + xyz$$

x	yz	00	01	11	10
0				1	
1			1	1	1

$$C = xy + xz + yz$$



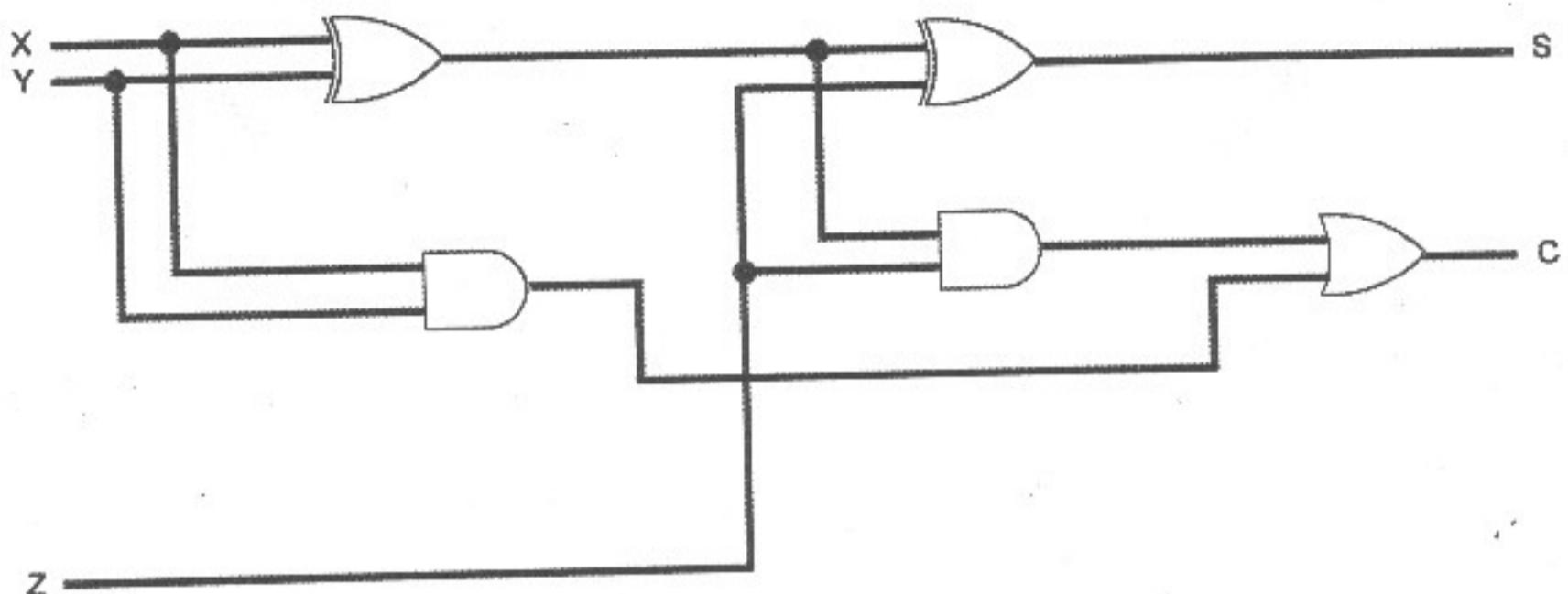
x	yz	00	01	11	10
0			1		
1		1		1	



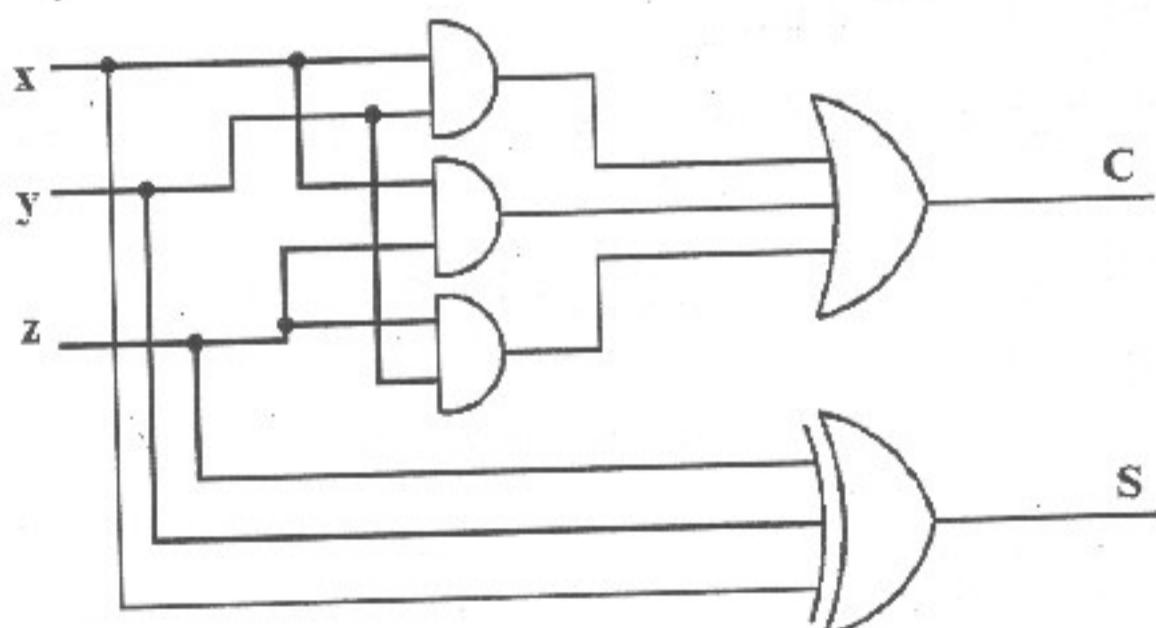
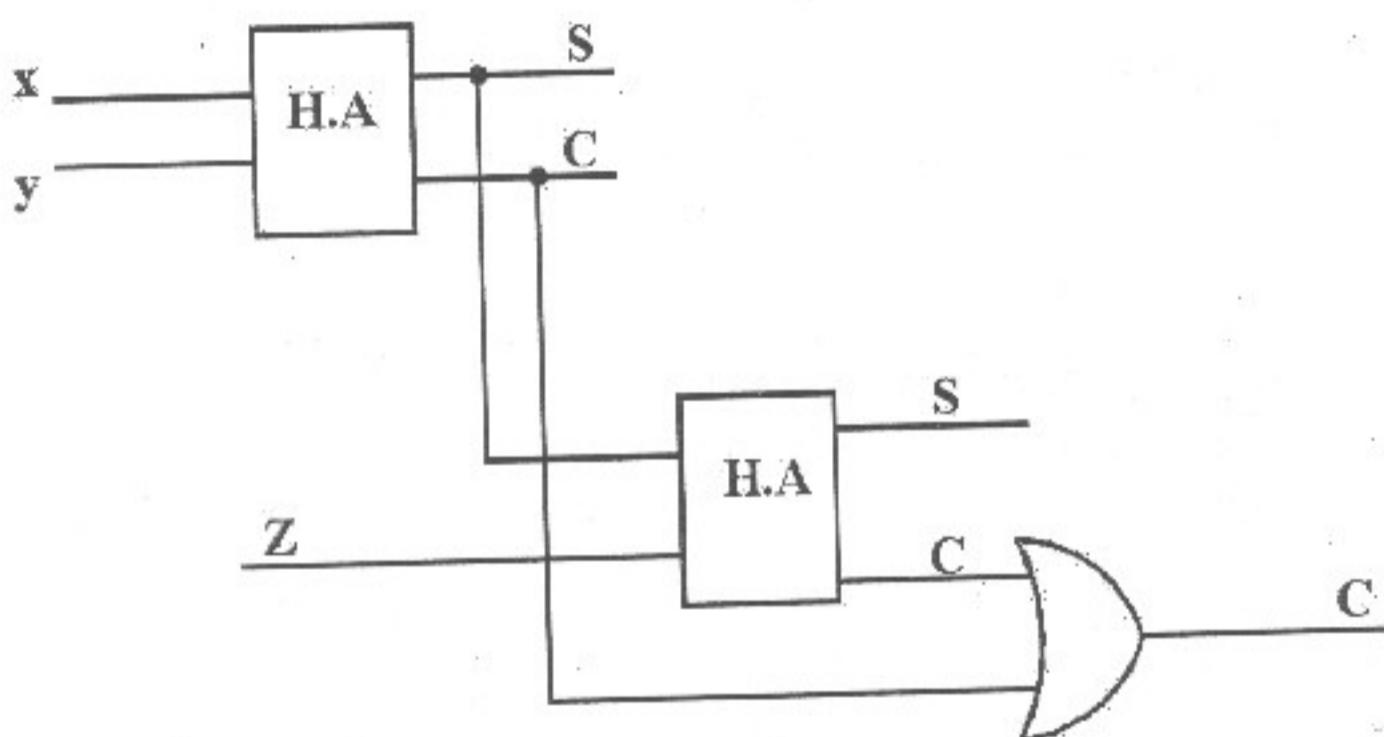
$$S = x'y'z + x'yz' + xy'z' + xyz$$

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

مدار منطقی



پیاده سازی یک تمام جمع کننده کامل با دو نیم جمع کننده و یک گیت OR



نیم تفاضل کننده :Half Subtractor

دو بیت را از هم کم می کند، حاصل تفاضل را در D و رقم قرضی را در B قرار می دهد. یعنی مداری است با دو ورودی و دو خروجی.

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

$$B = x'y$$

$$D = x \oplus y$$

تمام تفاضل کننده

دو بیت و رقم قرضی مرحله قبل را از هم کم می کند، حاصل تفاضل را در D و رقم قرضی را در B قرار می دهد یعنی مداری است ورودی و دو خروجی.

x	y	z	B	D
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

$$B = x'y'z + x'y'z' + x'yz + xyz$$

$$D = x \oplus y \oplus z$$

x	yz	00	01	11	10
0			1	1	1
1				1	

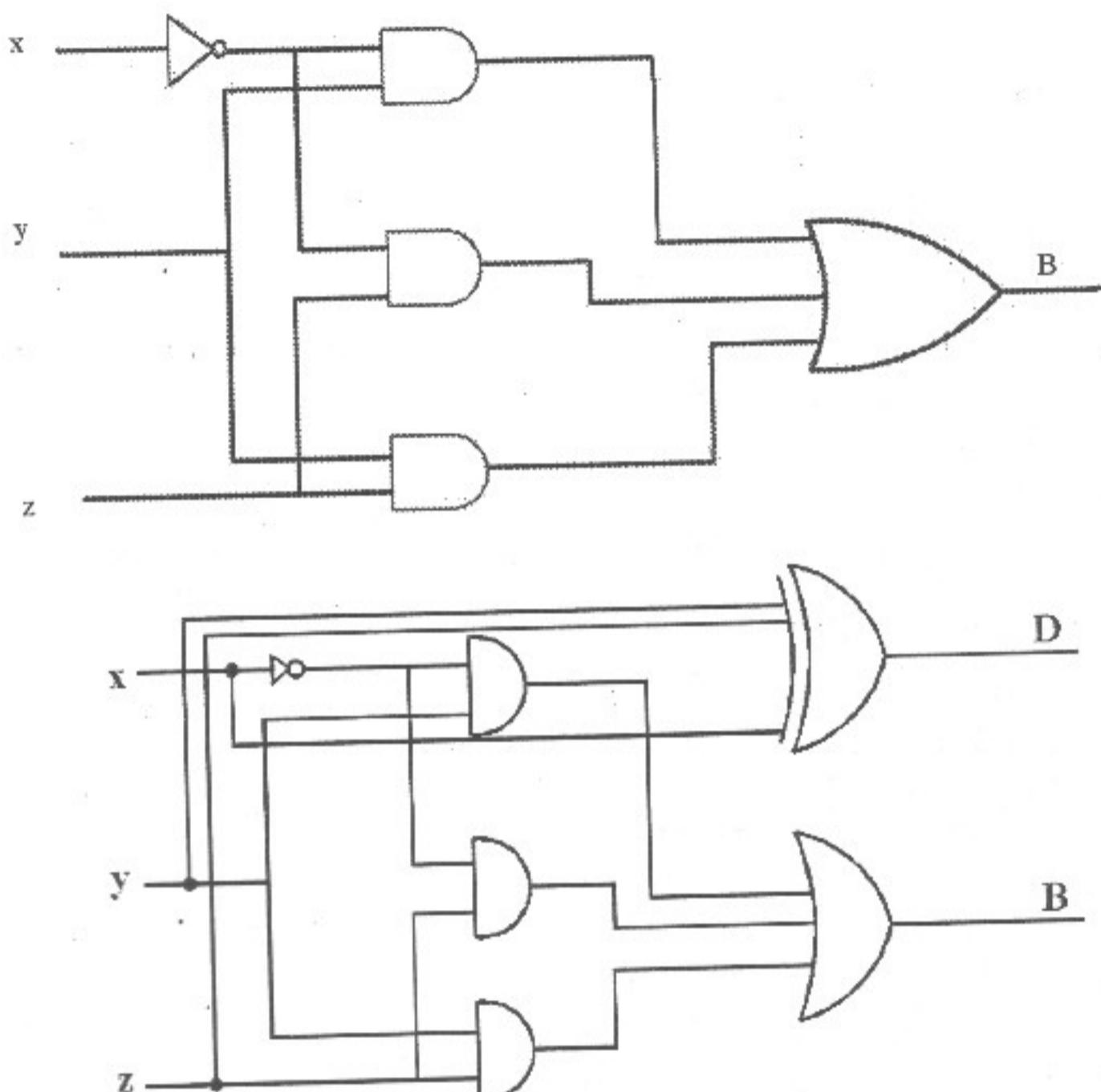
$$B = x'z + x'y + yz$$

x	yz	00	01	11	10
0		1			1
1		0	1	3	2

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

$$D = x \oplus y \oplus z$$

مدار منطقی



طراحی مدار تولید کننده بیت Parity

دلیل استفاده از بیت Parity برای تشخیص خطاست.

فرد اگر تعداد یک‌ها فرد بود به سمت راست صفر اضافه می‌شد و اگر تعداد یک‌ها زوج بود به سمت راست یک اضافه می‌شد.

x	y	z	فرد P
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

تشخیص خطا با بیت پریتی برای ۳ ورودی

این مدار دارای سه ورودی x و y و z می‌باشد، در واقع اطلاعات رسیده می‌باشند و P بیت پریتی است که فرستنده روی آن گذاشته، حال بررسی می‌کنیم که p گذاشته شده صحیح است اگر درست باشد مقدار C یک است و در غیر اینصورت صفر، به عنوان مثال:

X	Y	Z	P	C
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

در دو خط اول و دوم X و Y و Z صفر هستند.

پس باید Parity آن ۱ باشد در نتیجه خط دوم درست است.

$$C = x \oplus y \oplus z \oplus p$$

زوج اگر تعداد یک‌ها فرد بود سمت راست یک اضافه می‌کنیم و اگر تعداد یک‌ها زوج بود سمت راست صفر اضافه می‌کنیم.
فرد و زوج برای دو ورودی Parity

x	y	فرد P	زوج P
0	0	1	0
0	1	0	1
1	0	0	1
1	1	1	0

XNOR XOR

نتهه: مطالب فوق به این معنی است که اگر parity فرد را برای یکتابع بدست آوریم می‌توانیم با قرار دادن یک گیت Not در خروجی آن را به parity زوج تبدیل کنیم و برعکس.

مدار parity زوج و فرد:

	a	b	c	فرد P	زوج P
0	0	0	0	1	0
1	0	0	1	0	1
2	0	1	0	0	1
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	1	0
6	1	1	0	1	0
7	1	1	1	0	1

جدول ملطفی

a	bc	00	01	11	10
0	0	1		3	1
1	1		1	7	6
	4	5		7	6

$$\text{زوج } p = A \oplus B \oplus C$$

a	bc	00	01	11	10
0	0	1		1	3
1	1		1	5	7
	4	5		7	6

$$\text{فرد } p = A \odot B \odot C$$

نمایشگرهای 7-segment

xywz	a	b	c	d	e	f	g
0 0000	1	1	1	1	1	1	0
1 0001	0	1	1	0	0	0	0
2 0010	1	1	0	1	1	0	1
3 0010	1	1	1	1	0	0	1
4 0100	0	1	1	0	0	1	1
5 0101	1	0	1	1	0	1	1
6 0110	1	0	1	1	1	1	1
7 0111	1	1	1	0	0	0	0
8 1000	1	1	1	1	1	1	1
9 1001	1	1	1	1	0	1	1



تبدیل کد BCD به 7-Segment تمام حروف با جدول کارنو به همراه گیت

$$a = x'y'w'z' + x'y'wz' + x'y'wz + x'yw'z + x'ywz' + x'ywz + xy'w'z' + xy'w'z$$

$$b = x'y'w'z' + x'y'w'z + x'y'wz' + x'y'wz + x'yw'z' + x'ywz + xy'w'z' + xy'w'z$$

$$c = x'y'w'z' + x'y'w'z + x'y'wz + x'yw'z' + x'ywz + x'ywz' + x'ywz + xy'w'z' + xy'w'z$$

$$d = x'y'w'z' + x'y'wz' + x'y'wz + x'yw'z + x'ywz' + xy'w'z' + xy'w'z$$

$$e = x'y'w'z' + x'y'wz' + x'ywz' + xy'w'z'$$

$$f = x'y'w'z' + x'yw'z' + x'yw'z + x'ywz' + x'ywz + xy'w'z' + xy'w'z$$

$$g = x'y'wz' + x'y'wz + x'yw'z' + x'yw'z + x'ywz' + xy'w'z' + xy'w'z$$

wz	xy	00	01	11	10
00	1	1		1	1
01					
11	x	x	x	x	x
10	1	1	x	x	x

این جدول تنها برای g رسم شده می‌توان برای بقیه خروجی‌ها نیز رسم کرد.

نکته:

در کد BCD تا عدد 9 را در نظر می‌گیریم پس در جدول کارنو خانه‌های 9 به بعد don't care به حساب می‌آیند.



توضیحات:

منظور از گیت یک ورودی به این معنی نمی‌باشد که آن گیت دو ورودی ندارد بلکه منظور اینست که یک ورودی آن را مقدار ثابت می‌دهیم و خروجی را در مقابل آن می‌سنجیم.

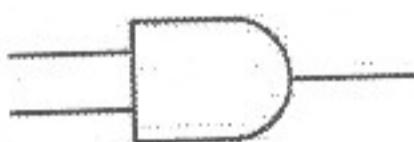
مثالاً اگر یک ورودی گیت AND را ثابت و برابر یک قرار دهیم به آن هر ورودی دیگر بدھیم در خروجی ظاهر خواهد شد.

اطلاعاتی در مورد گیت‌ها:

باید توجه داشت که تمام گیتهای زیر دارای یک ورودی ثابت می‌باشند که با C نمایش داده شده است.

یک ورودی AND:

مقدار ثابت



	x	F
C = 0	0	0
	1	0
C = 1	0	0
	1	1

$\rightarrow F = 0$

$\rightarrow F = x$

نتیجه: در گیت AND اگر یک ورودی صفر باشد خروجی حتماً صفر است. اما اگر یک ورودی یک باشد خروجی برابر ورودی دیگر است.

یک ورودی OR:

مقدار ثابت C



	x	F
C = 0	0	0
	1	1
C = 1	0	1
	1	1

$\rightarrow F = x$

$\rightarrow F = 1$

* اگر یک ورودی یک باشد خروجی حتماً یک است. اگر یک ورودی گیت OR صفر باشد خروجی برابر ورودی دوم است.

یک ورودی NAND:

مقدار ثابت C



	x	F
C = 0	0	1
	1	1
C = 1	0	1
	1	0

$\rightarrow F = 1$

$\rightarrow F = x'$

* در NAND اگر یک ورودی صفر باشد خروجی حتماً یک است و اگر یک ورودی یک باشد خروجی NOT شده ورودی دوم است.

یک ورودی NOR:

مقدار ثابت C



	x	F
C = 0	0	1
	1	0
C = 1	0	0
	1	0

$\rightarrow F = x'$

$\rightarrow F = 0$

مدار منطقی

* اگر یک ورودی یک باشد خروجی حتماً صفر است. اگر یک ورودی صفر باشد خروجی NOT ورودی دوم است.

یک ورودی XOR:



	X	F
C = 0	{ 0	0
	{ 1	1
C = 1	{ 0	1
	{ 1	0

$\rightarrow F = x$

$\rightarrow F = x'$



	X	F
C = 0	{ 0	1
	{ 1	0
C = 1	{ 0	0
	{ 1	1

$\rightarrow F = x'$

$\rightarrow F = x$

جمع کننده دو به دو موازی

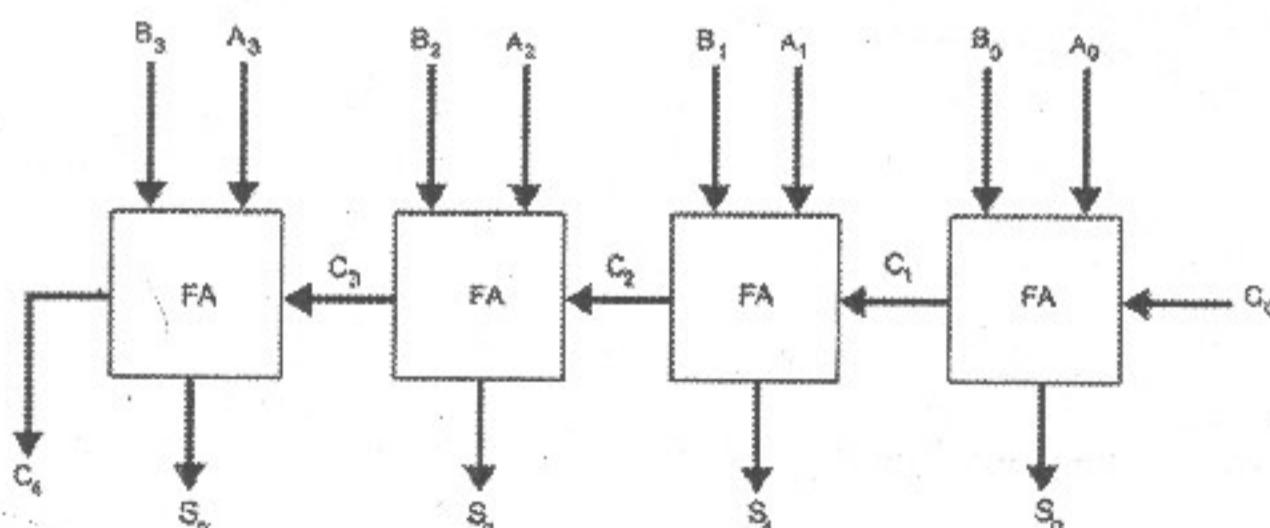
برای جمع دو عدد n بیتی از full Adder استفاده می‌شود که هر یک شامل یک C_{in} ، C_{out} و حاصل جمع (S) می‌باشد. (n بیتی)

برای خروجی هر مرحله، C_{in} برای مرحله بعدی است.

$$A = (A_{n-1} A_{n-2} \dots A_2 A_1 A_0)$$

$$B = (B_{n-1} B_{n-2} \dots B_2 B_1 B_0)$$

$$S = (S_{n-1} S_{n-2} \dots S_2 S_1 S_0)$$



جمع کننده 4 بیت

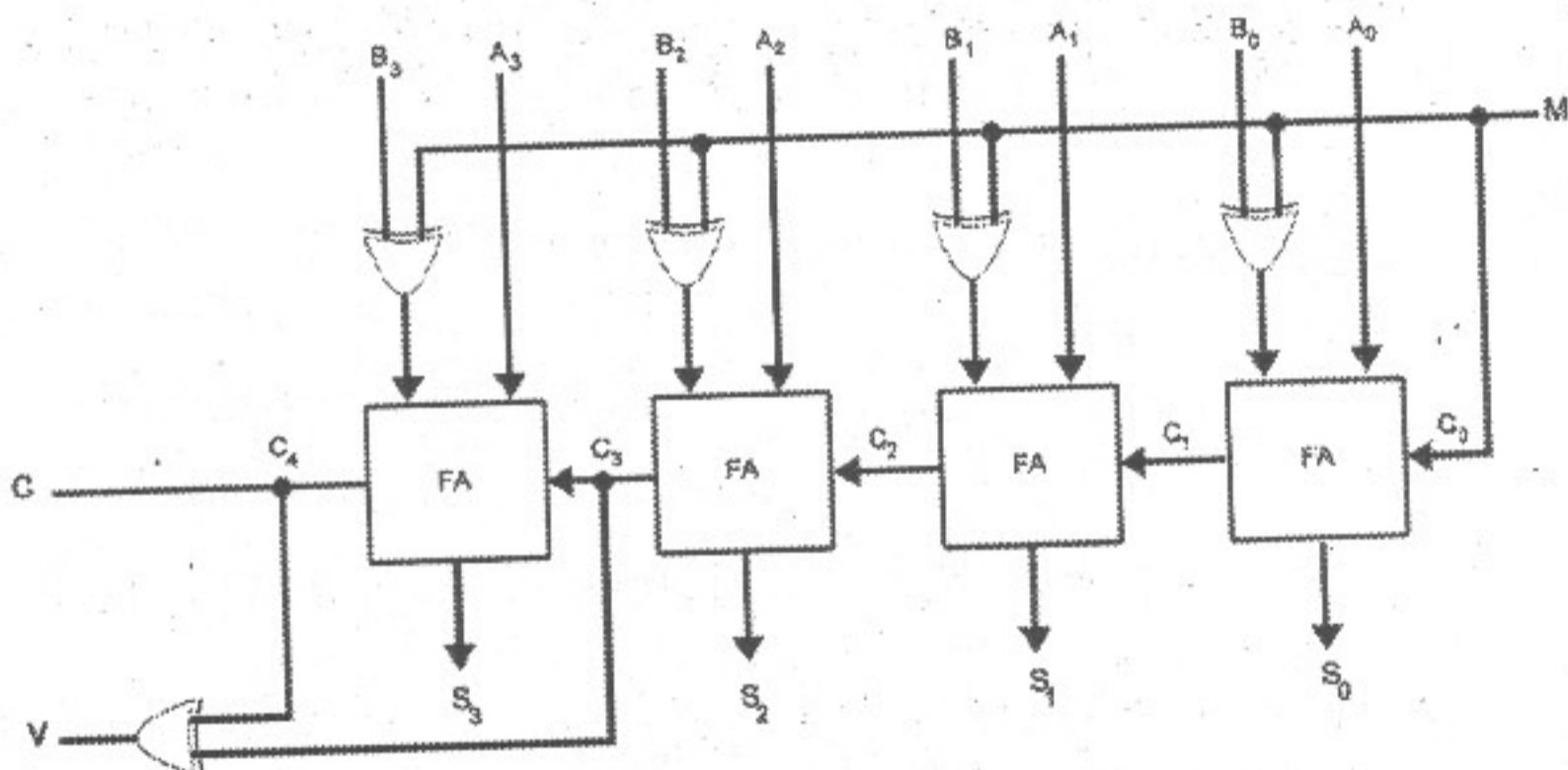
تفریق دودویی

برای تفریق دو عدد A و B (A-B) ابتدا از B مکمل ۲ می‌گیریم سپس A را با مکمل ۲ عدد B جمع می‌کنیم.

اما برای این کار از گیت X-OR استفاده می‌کنیم یعنی یک ورودی X-OR را ثابت و برابر با مقدار Logic یک قرار می‌دهیم. و ورودی دیگر را همان عدد B می‌گذاریم. با این عمل در واقع مکمل یک عدد B را بدست آورده‌ایم حال باید آن را با یک جمع کنیم تا حاصل همان مکمل ۲ شود.

در واقع ورودی هر Full Adder یک یست از عدد A و یک خروجی X-OR که از بیت متاضر B بدست آمده می‌باشد. باید توجه داشت دیگر ورودی X-OR مورد نظر خط M می‌باشد که هنگامی که مقدارش صفر است عمل جمع B و A را انجام می‌دهد و اگر M مقدارش یک باشد عمل تفریق صورت می‌گیرد.

برای اینکه عمل تفریق به درستی انجام شود و A با مکمل ۲ عدد B جمع شود باید ورودی C_{in} وصل کرد تا زمانی که $M=1$ است و عمل تفریق انجام می‌شود، A با مکمل یک عدد B و عدد یک جمع شود تا حاصل تفریق درست باشد.



جمع - تفریق گر

افتشار رقم نقلی

با توجه به شکل فوق، می‌بینیم که برای انجام عمل جمع، هر مرحله منتظر Carry مرحله قبل می‌ماند تا آن را دریافت کند بعد عمل جمع را انجام دهد. این تأخیر تا مراحل بعدی ادامه دارد به همین دلیل در انتها تاخیر زیادی خواهیم داشت. لذا برای از بین بردن این تاخیر با توجه به شکل F.A به پیش‌بینی رقم نقلی می‌پردازیم.

$$P_i = A_i \oplus B_i$$

$$G_i = A_i B_i$$

$$S_i = P_i \oplus C_i$$

مدار منطقی

ورودی C_0 =

$$C_{i+1} = G_i + P_i C_i$$

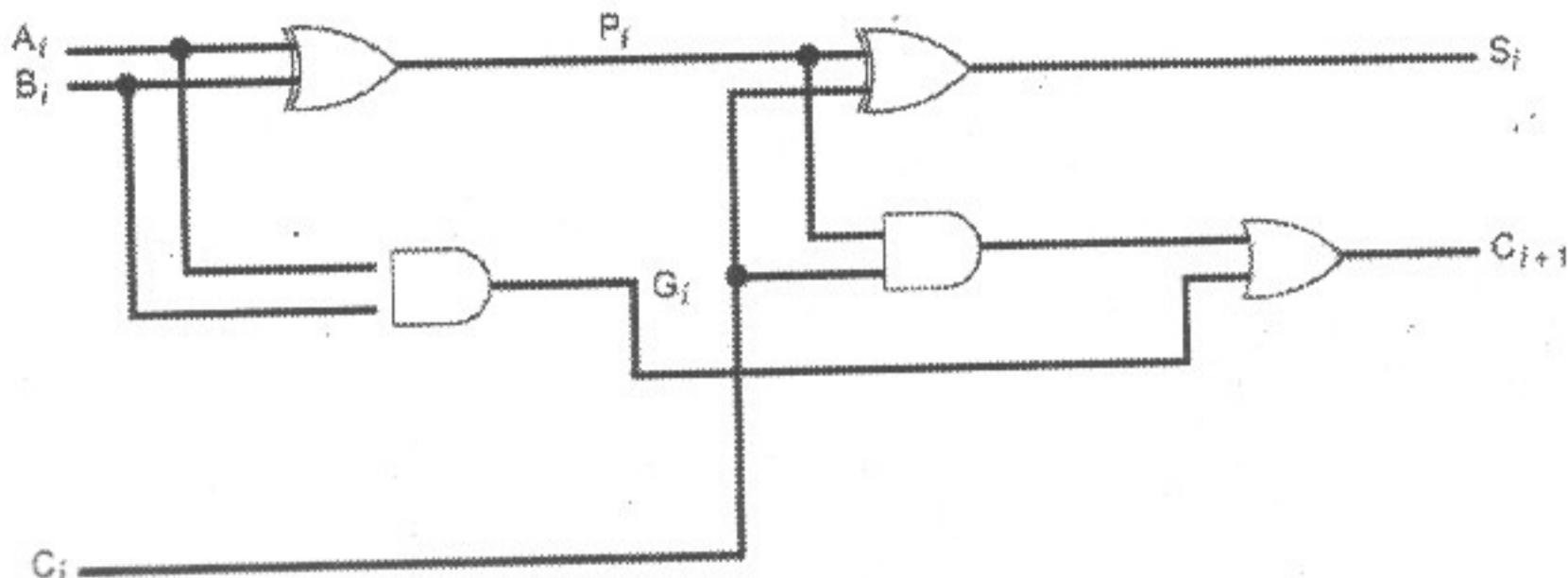
$$C_1 = G_0 + P_0 C_0$$

$$C_2 = G_1 + P_1 C_1$$

$$G_1 + P_1(G_0 + P_0 C_0)$$

$$C_3 = G_2 + P_2 C_2$$

$$G_2 + P_2(G_1 + P_1(G_0 + P_0 C_0))$$

جمع کننده کامل با P و G

حال می‌توانیم C_0 را نیز بر حسب C_{in} یعنی $C_{in} = C_0$ بدست آوریم و لازم نیست منتظر محاسبه آن باشیم.

نهایا مشکل این روش وجود محاسبات بسیار بالای آن می‌باشد، اما لازم به ذکر است که اگر دو عددی که با هم جمع می‌شوند تعداد یتیهاشان زیاد باشد این روش خیلی سریعتر از عدم استفاده از آن می‌باشد.

جمع کننده BCD

همانطور که میدانیم اعداد BCD از صفر شروع و به ۹ ختم می‌شوند و می‌بینیم که:

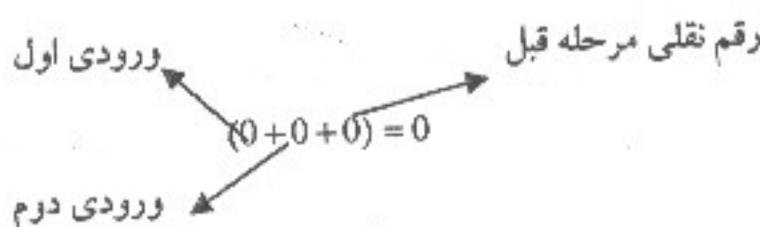
$$13 = \begin{cases} \text{BCD} & = 1001 \\ \text{باينري} & = 1101 \end{cases}$$

بزرگترین حاصل جمع در BCD

$$9 + 9 + 1 = 19$$

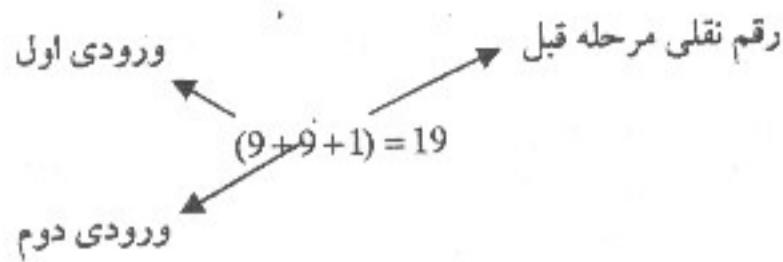
$$1\ 1001$$

بدینهی است که کوچکترین حاصل جمع نیز صفر می‌باشد یعنی هر دو ورودی صفر باشند و همچنین از مرحله قبل نیز Carry وجود نداشته باشد.





با توجه به توضیحات بالا به این نتیجه می‌رسیم که بزرگترین حاصل جمع مکمل در BCD جمع دو عدد ۹ و جمع حاصل با کری مرحله قبل می‌باشد.



مشکل موجود در این نوع جمع استفاده از جمع کننده دو به دو موازی می‌باشد. که اعداد خروجی را به صورت باینری تولید می‌کند. بنابراین باید اعداد صفر تا ۱۹ را به معادل BCD تبدیل کنیم. برای این کار از جدول زیر استفاده می‌کنیم.

K	جمع دودویی					جمع BCD					دهدنه
	Z ₈	Z ₄	Z ₂	Z ₁	C	S ₈	S ₄	S ₂	S ₁	0	
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1	1
0	0	0	1	0	0	0	0	1	0	2	2
0	0	0	1	1	0	0	0	1	1	3	3
0	0	1	0	0	0	0	1	0	0	4	4
0	0	1	0	1	0	0	1	0	1	5	5
0	0	1	1	0	0	0	1	0	1	6	6
0	0	1	1	1	0	0	1	1	1	7	7
0	1	0	0	0	0	1	0	0	0	8	8
0	1	0	0	1	0	1	0	0	1	9	9
1	1	0	1	0	1	0	0	0	0	10	10
0	1	0	1	1	1	0	0	0	1	11	11
0	1	1	0	0	1	0	0	1	0	12	12
0	1	1	0	1	1	0	0	1	1	13	13
0	1	1	0	1	1	0	0	1	0	14	14
0	1	1	1	0	1	0	1	0	1	15	15
0	1	1	1	1	1	0	1	0	0	16	16
1	0	0	0	0	1	0	1	1	1	17	17
1	0	0	0	1	1	0	1	0	0	18	18
1	0	0	1	0	1	1	0	0	1	19	19
1	0	0	1	1	1	1	0	0	1	19	19

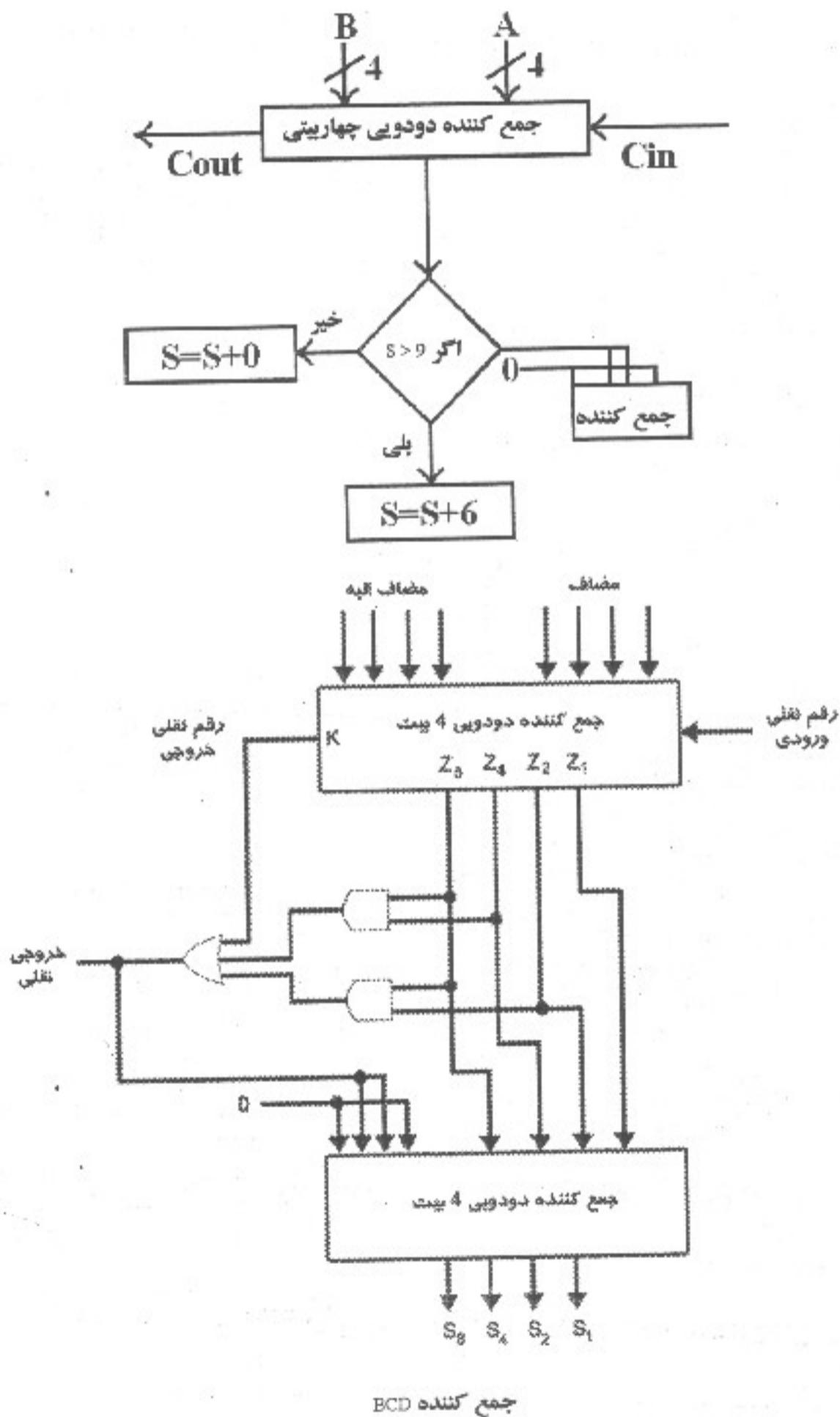
با توجه به جدول فوق مشاهده می‌شود که اعداد ۰ تا ۹ در باینری معادل BCD هستند اما از اعداد ۱۰ تا ۱۹ باید به معادل BCD تبدیل شوند. با توجه به جدول (با استفاده از جدول کارنو) می‌توان خروجی‌های ساده شده را بدست آورد. اما با توجه به شکل جدول می‌توان دید که به اعداد باینری مقدار 6 اضافه شده تا حاصل به صورت BCD درآید. به عنوان مثال عدد 16 در باینری معادل 10000 می‌باشد. برای تبدیل آن به 16 در BCD باید به آن (0110) را اضافه کرد یعنی به صورت (10110) که 4 بیت سمت راست معادل 6 و سمت چپ‌ترین بیت معادل 1 است.

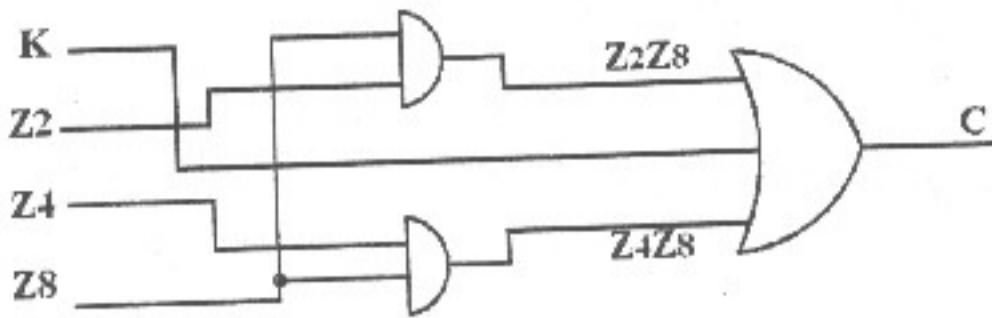
مدار منطقی

برای این کار از دو جمع کننده دو دویی ۴ یتی باید استفاده کرد به این صورت که دو عددی را که می‌خواهیم با هم جمع کنیم را به جمع کننده اول وارد می‌کنیم حاصل را به جمع کننده دوم اعمال می‌کنیم و با توجه به جدول و به دست آوردن معادله خروجی C مربوط به کری خواهیم داشت:

بنابراین با استفاده از خروجی‌های جمع کننده اول خروجی C را تولید می‌کنیم. و باز با توجه به جدول می‌بینیم زمانی باید مقدار عددی 6 را به عدد بازتری اضافه کنیم که مقدار C ، یک است.

حال باید دیگر ورودی جمع کننده دوم را عدد 6 بگذاریم (0110) می‌بینیم که این مقدار زمانی فعال است که مقدار خروجی C یک باشد.





در مقایسه دو عدد باینری باید مقایسه یست به بیت انجام شود. می‌دانیم با گیت X-NOR (یا X-OR فرقی نمی‌کند) می‌توان عمل مقایسه را انجام داد. در واقع تنها می‌توان با یک X-NOR مساوی بودن یست‌ها را چک کرد و برای یک داده n بیتی می‌توان از n عدد X-NOR استفاده کرد؛ تا عمل مقایسه انجام شود. به این شکل که اگر تمام خروجی X-NOR ها، یک شود یعنی دو عدد n بیتی با یکدیگر مساوی هستند، اما در غیر اینصورت باید بتوان تشخیص داد کدامیک بزرگتر هستند.

رابطه ۱-۹

$$x_i = A_i \odot B_i, \quad i=0, \dots, n-1$$

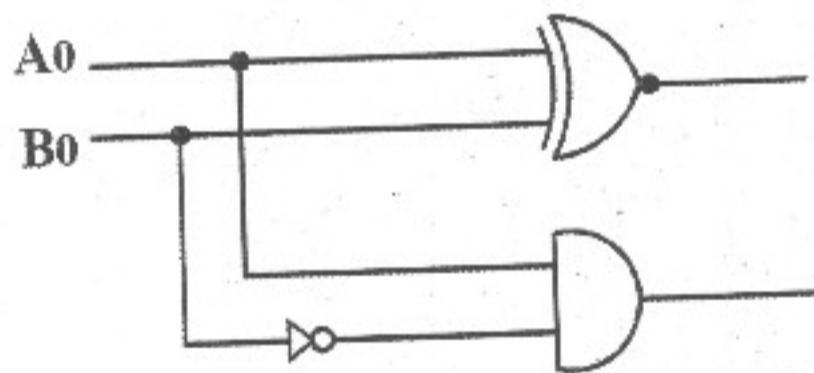
$$\Rightarrow x_1 = A_1 \odot B_1$$

$$x_2 = A_2 \odot B_2$$

⋮

$$x_n = A_n \odot B_n$$

A	B	\odot
0	0	1
0	1	0
1	0	0
1	1	1

اگر $A_0 = B_0 \leftarrow f = 1$ اگر $A_0 \neq B_0 \leftarrow f = 0$

①

اگر $A_0 > B_0 \leftarrow f = 1$ اگر $A_0 < B_0 \leftarrow f = 0$

②

اگر با هم اتفاق یافتد $\leftarrow A_0 < B_0 \leftarrow ①, ②$

در صورتی $A > B$ است که بیت پر ارزش آن از بیت پر ارزش B بزرگتر باشد یعنی بیت n ام، A باید یک و بیت n ام، B باید صفر باشد. اما زمانی که بیت های n ام مساوی بودند باید به بیت $n-1$ ام توجه کرد و به همین ترتیب اگر این بیت ها هم مساوی بودند باید به بیت های کم ارزش تر برسیم.

اگر دو عدد چهار بیتی $A = A_3A_2A_1A_0$ و $B = B_3B_2B_1B_0$ را در نظر بگیریم می‌بینیم که اگر دو عدد با هم متساوی باشند حاصل $x_i = A_i \odot B_i$ و در غیر اینصورت خواهیم داشت:

مدار منطقی

به معنی یک بودن A و صفر بودن B می‌باشد: AB' و به معنی یک بودن A است:

به معنی صفر بودن A است: A'

بنابراین ملاحظه می‌شود که:

$$f(A > B) \Rightarrow A_3B'_3 + x_3 \cdot A_2B'_2 + x_3x_2A_1B'_1 + x_3x_2x_1A_0B'_0 \rightarrow f \begin{cases} 1 \rightarrow A > B \\ 0 \rightarrow A \neq B \end{cases} \quad (1-10)$$

$$f(A < B) \Rightarrow A'_3B_3 + x_3 \cdot A'_2B_2 + x_3x_2A'_1B_1 + x_3x_2x_1A'_0B_0 \rightarrow f \begin{cases} 1 \rightarrow A < B \\ 0 \rightarrow A \neq B \end{cases} \quad (1-11)$$

مثال:

دو عدد $A = 1111$ و $B = 1110$ را با روش فوق با یکدیگر مقایسه کند.

با استفاده از رابطه ۱-۱۱ داریم:

$$\begin{array}{l} A = 1111 \Rightarrow \begin{cases} A_3 = 1 & B_3 = 1 \\ A_2 = 1 & B_2 = 1 \\ A_1 = 1 & B_1 = 1 \\ A_0 = 1 & B_0 = 0 \end{cases} \\ B = 1110 \Rightarrow \begin{cases} A_3B'_3 + x_3 \cdot A_2B'_2 + x_3x_2A_1B'_1 + x_3x_2x_1A_0B'_0 \\ 0 + (1)(0) + (1)(1)(0) + (1)(1)(1)(1) \end{cases} \end{array}$$

در صورتی که حاصل $A > B$ اما در غیر اینصورت نمی‌توان گفت که حتماً $A < B$ است زیرا در صورت تساوی A و B نیز حاصل محاسبات فوق صفر خواهد بود و برای اینکه بتوانیم تشخیص دهیم که $A < B$ است باید حتماً از رابطه ۱-۱۱ استفاده کنیم.

$$A_3B'_3 = 0$$

$$x_3 = A_3 \cdot B_3 = 1$$

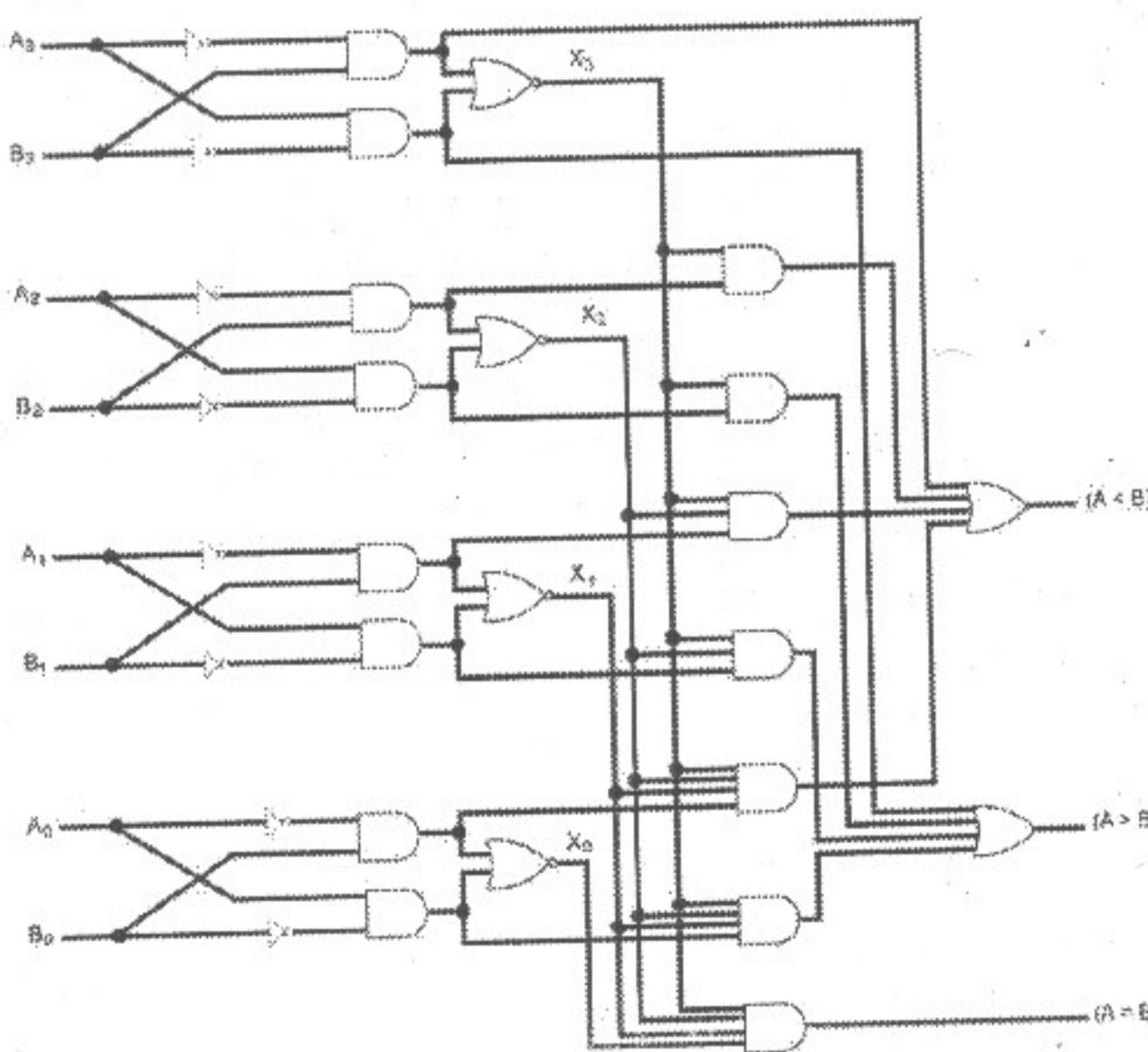
$$A_2B'_2 = 0$$

$$x_2 = A_2 \cdot B_2 = 1$$

$$A_1B'_1 = 0$$

$$x_1 = 1$$

$$A_0B'_0 = 1$$



مقایسه ۴-بیتی مقدار چهار بیتی



دیکدر (Decoder)

n ورودی

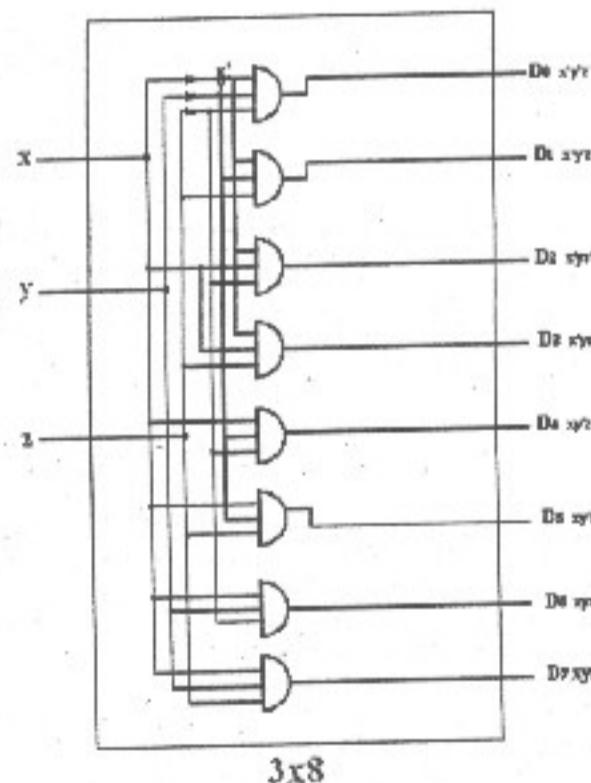
2ⁿ خروجی

یک دیکدر شامل n ورودی و 2ⁿ خروجی می‌باشد. در واقع با اعمال ورودی‌ها می‌توان تمام 2ⁿ میترم موجود را پیاده‌سازی کرد. به این شکل که با اعمال ورودی و با توجه به مقدار باینری آن همان شماره در خط خروجی (شماره میترم) فعال می‌شود. یعنی همان خط مقدارش یک و بقیه صفر می‌شود بنابراین مشاهده می‌شود با یک دیکدر می‌توان مسائل ترکیبی مطرح شده در قبل را به راحتی پیاده‌سازی کرد. باید توجه داشت که در یک Decoder در هر لحظه تنها یک خروجی یک می‌باشد.

ورودی‌ها			خروجی‌ها							
x	y	z	D ₀	D ₁	D ₂	D ₃	D ₄	D ₅	D ₆	D ₇
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

ساخته‌مان داخلی یک دیکدر

	x	y	z	
0	0	0	0	x'y'z'
1	0	0	1	x'y'z
2	0	1	0	x'yz'
3	0	1	1	x'yz
4	1	0	0	xy'z'
5	1	0	1	xy'z
6	1	1	0	xyz'
7	1	1	1	xyz



- طراحی به کمک دیکدر

۱- یک دیکدر OR و گیت Active high یک‌های خروجی

۲- یک دیکدر NOR و گیت Active high صفرهای خروجی

۳- یک دیکدر AND و گیت Active low صفرهای خروجی

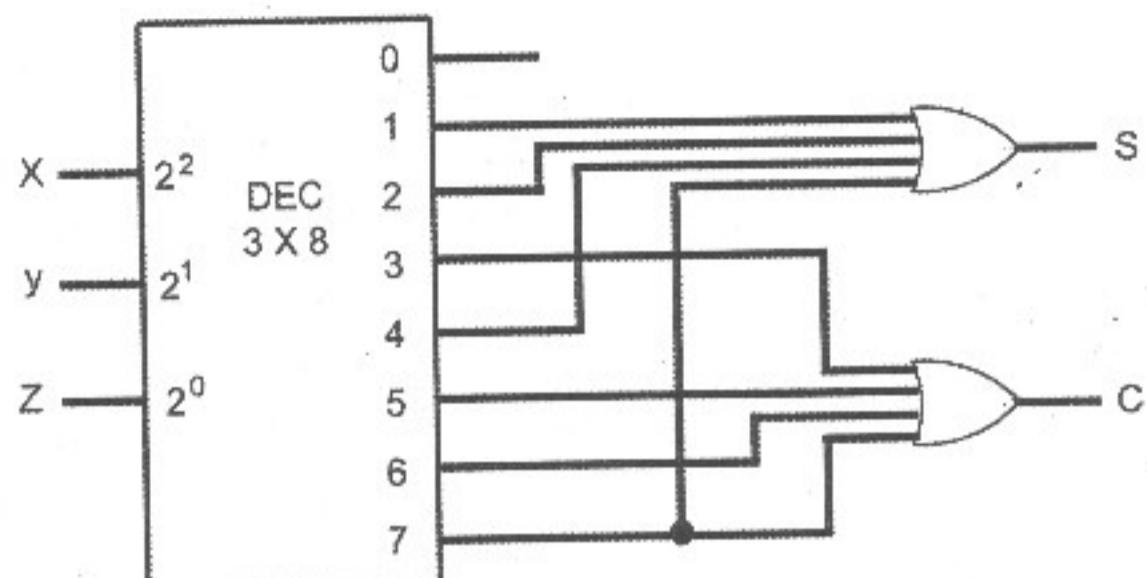
۴- یک دیکدر NAND و گیت Active low یک‌های خروجی

مدار منطقی

مثال:

یک مدار تمام جمع کننده با استفاده از یک دیکدر و گیت OR طراحی کنید. (به تعداد خروجی گیت OR داریم)

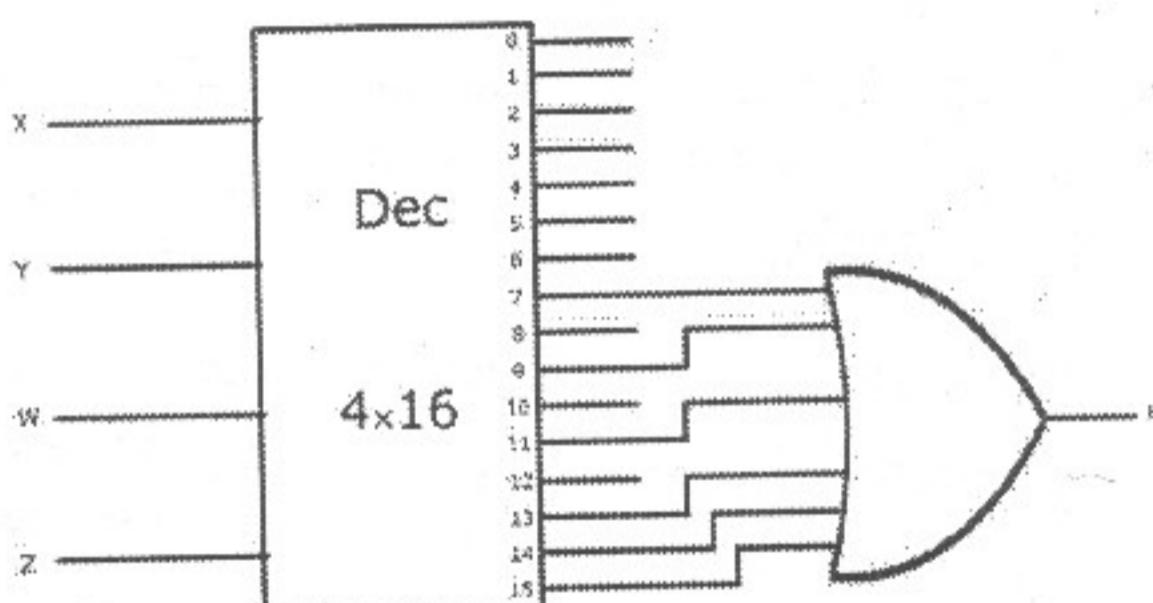
x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	1	0	1
1	1	0	1	0
1	1	0	1	0
1	1	1	1	1



مثال:

یک میز رای گیری که ۴ رای دهنده وجود دارد (x,y,z,w) زمانی چراغ رای گیری روشن می شود که یا سه نفر جواب مثبت داده باشند یا اینکه نفر اول و چهارم رای مثبت داده باشند.

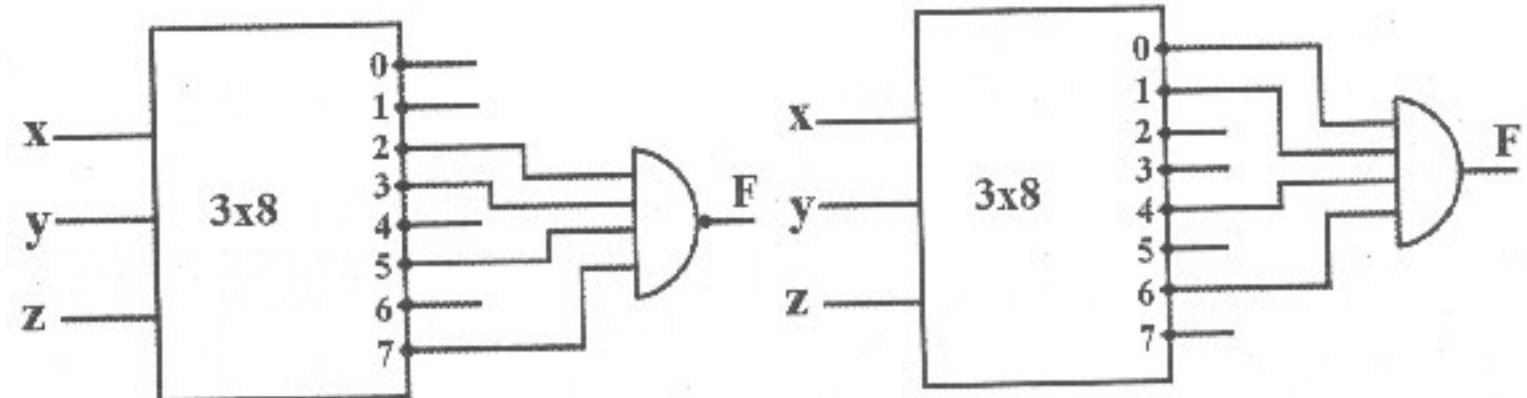
x	y	w	z	f
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1



مثال - فرض کنید تابع f به صورت زیر تعریف شده می خواهیم به کمک یک دیکدر AND و گیت های NAND و Active low F را پیاده سازی کنیم.

Active low \rightarrow AND \rightarrow صفرهای خروجی

	x	y	z	f
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	0	0	0	0
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1



دیکدر همراه با خط Enable

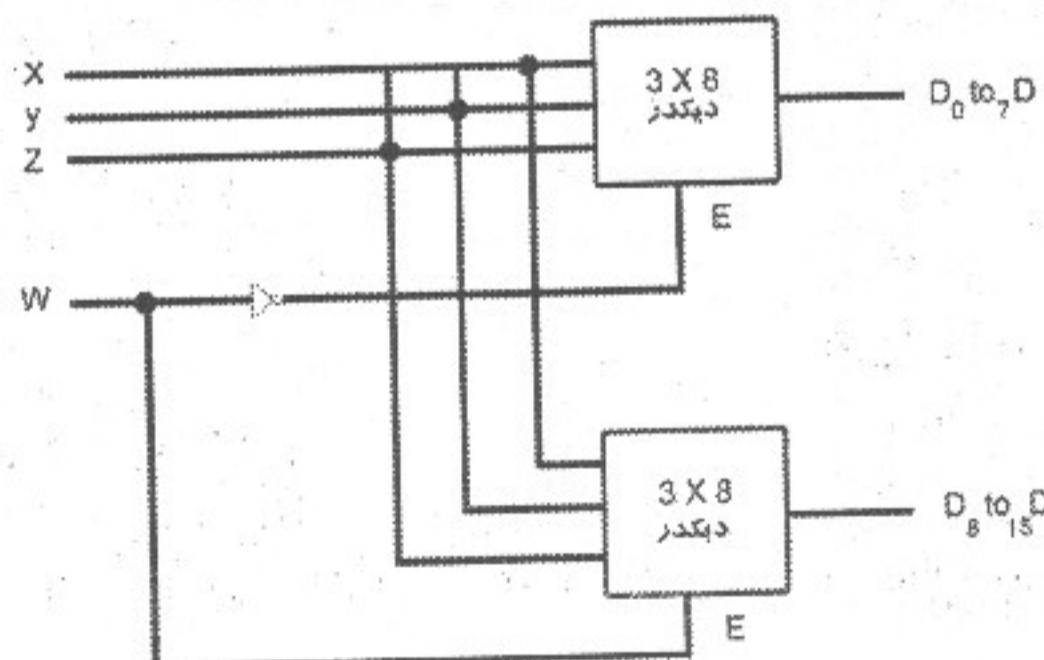
این دیکدر به وسیله یک خط به نام Enable قادر به کار کردن می‌باشد یعنی زمانی که خط $E = 1$ باشد دیکدر فعال و در زمان صفر بودن E دیکدر غیرفعال است. غیرفعال بودن آن به این معنی است که تمام خروجی‌ها مقدارشان صفر است.

ورودی‌ها								خروجی‌ها		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	1	0	1
0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

ترکیب دیکدرها:

می‌توان با استفاده از دیکدرها با ورودی کمتر دیکدرهایی با خروجی یافته ساخت به عنوان مثال با استفاده از دو دیکدر 8×3 می‌توان یک دیکدر 16×4 ساخت.

با توجه به شکل ملاحظه می‌شود که دیکدر به وجود آمده دارای 4 ورودی و 16 خروجی می‌باشد، نحوه کار آن به این شکل است که در زمانی که E مقدارش یک است دیکدر پایین فعال است و دیکدر بالا غیر فعال بنابراین خروجی‌های دیکدر بالائی همگی صفر هستند و بسته به مقداری که به x و y و z داده می‌شود که خط خروجی در دیکدر پایین یک می‌شود بنابراین می‌توان با مقداردهی به x و y و z و E تمامی 16 میترم موجود را تولید کرد.



دیکدر 16×4 ساخته شده با دو دیکدر 8×3

مدار ملطفی

خط E در شکل فوق به صورت Active high می‌باشد در این حالت هنگامی که E مقدارش صفر باشد دیکدر به صورت غیرفعال می‌باشد به این معنا که تمام خروجی‌ها غیرفعال است (اگر دیکدر Active high بود خروجی‌ها همه صفر و اگر Active low بود خروجی‌ها همه یک می‌شود).

اما زمانی که E مقدار یک داشته باشد دیکدر فعال می‌شود و با توجه به عدد ورودی یک شماره خروجی فعال می‌شود..

انکدرها (Encoder)

انکدر یعنی عکس دیکدر می‌باشد یعنی 2^n ورودی و n خروجی است. در هر لحظه تنها یک ورودی فعال است که باعث نمایش یک مقدار باینری در خروجی است.

$$x = D_4 + D_5 + D_6 + D_7$$

$$y = D_2 + D_3 + D_6 + D_7$$

$$z = D_1 + D_3 + D_5 + D_7$$

انکدرهای اولویت

در انکدر اولویت به جای فعال بودن تنها یک یست از ورودی‌ها می‌تواند چند یست فعال باشد. اما باید توجه داشت برای نمایش خروجی به بالرژش‌ترین یست‌ها توجه می‌شود یعنی اگر در ورودی دو خط فعال بود آن خطی در نظر گرفته می‌شود که ارزش یستی آن بیشتر باشد. به این معنی که برای نشان دادن خروجی ابتدا بالرژش‌ترین یست ورودی چک می‌شود و اگر مقدارش یک بوده به دیگر ورودی‌ها اهمیتی نمی‌دهد و اگر مقدار بالرژش‌ترین یست ورودی صفر بود سراغ یستهای بعدی می‌رود.

ورودی‌ها				خروجی‌ها		
D ₀	D ₁	D ₂	D ₃	x	y	v
0	0	0	0	x	x	0
1	0	0	0	0	0	1
x	1	0	0	0	1	1
x	x	1	0	1	0	1
x	x	x	1	1	1	1

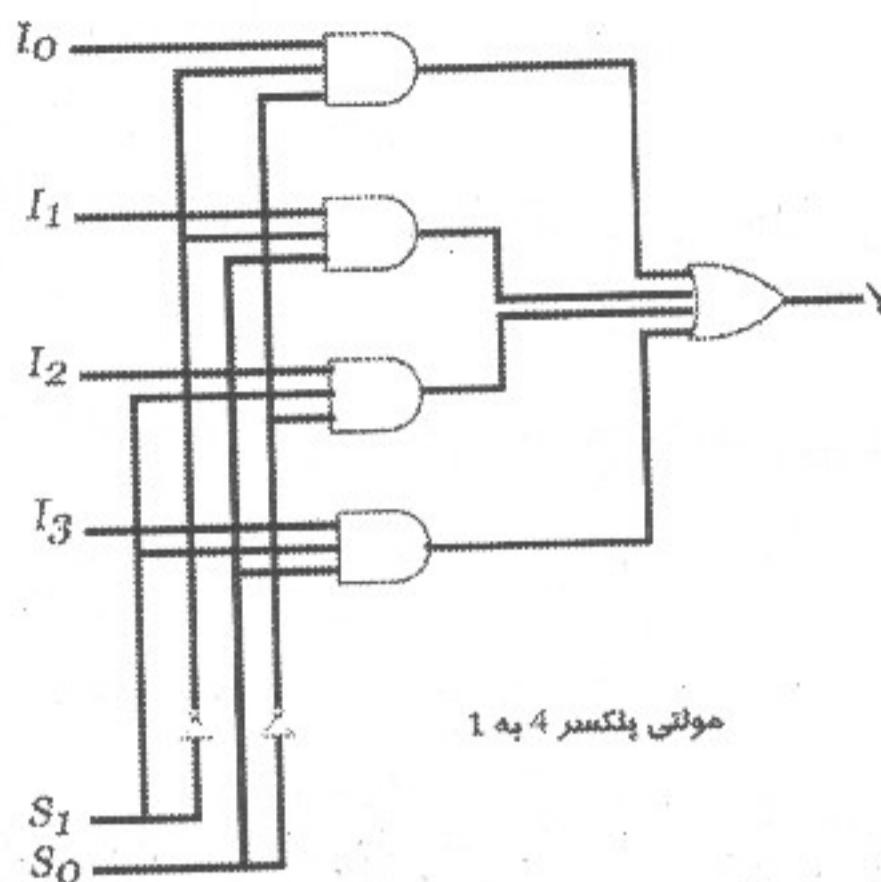
مولتی پلکسر MUX (Multiplexer)

در مولتی پلکسر 2^n خط ورودی داریم و n خط select و یک خروجی. در اصل مقدار وارد شده به خطوط select باعث فعال شدن یکی از خطوط ورودی و انتقال آن به خروجی می‌شود. برای پیاده‌سازی توابع ترکیبی می‌توان از مولتی پلکسر استفاده کرد.

S ₁	S ₀	y
0	0	I ₀
0	1	I ₁
1	0	I ₂
1	1	I ₃

2^n ورودی

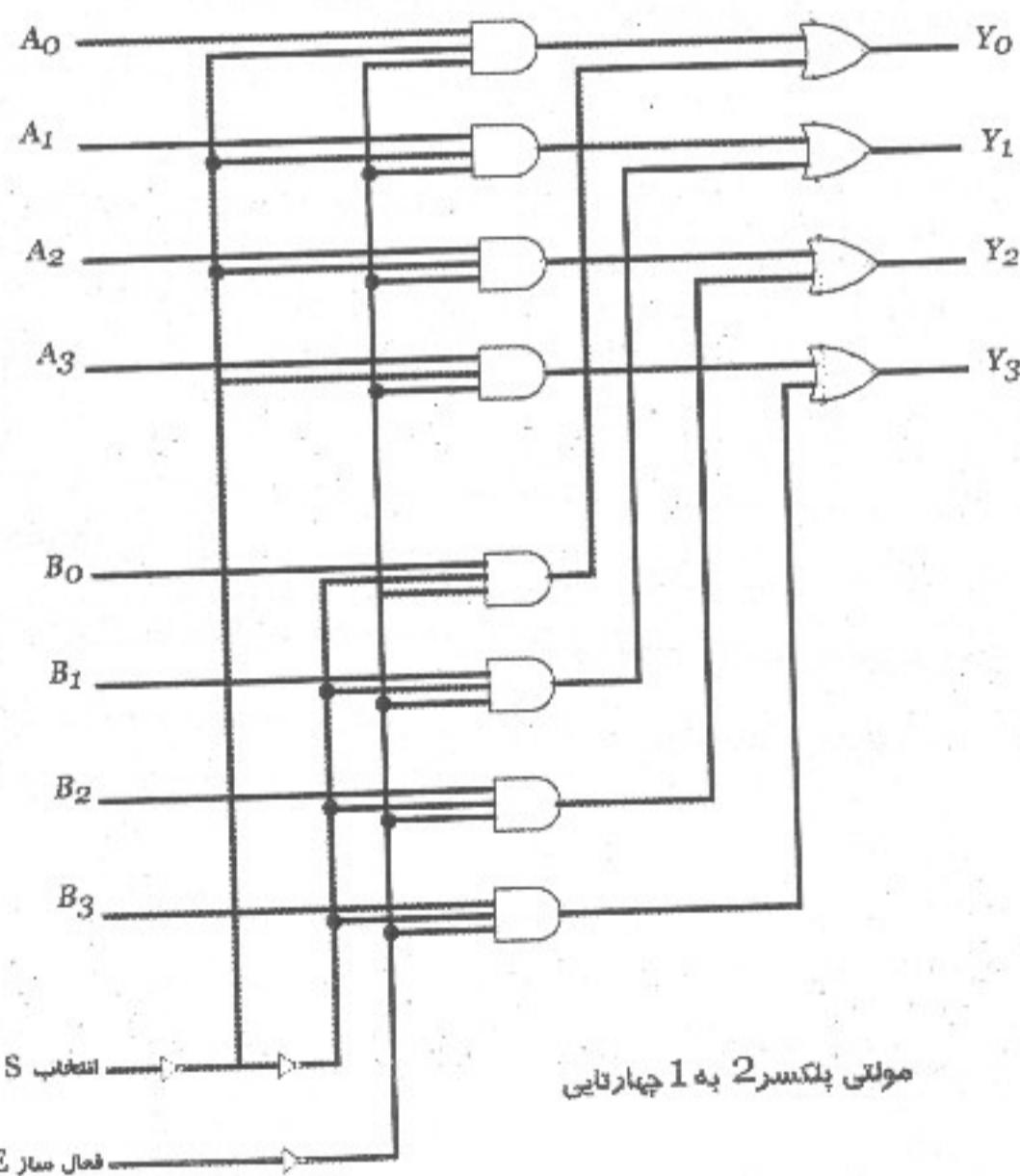
1 خروجی



مولتی پلکسر ۴ به ۱

شماره ۱ زمانی خروجی اش برابر I_0 می‌شود که هر دو S_0 و S_1 برابر صفر باشد.

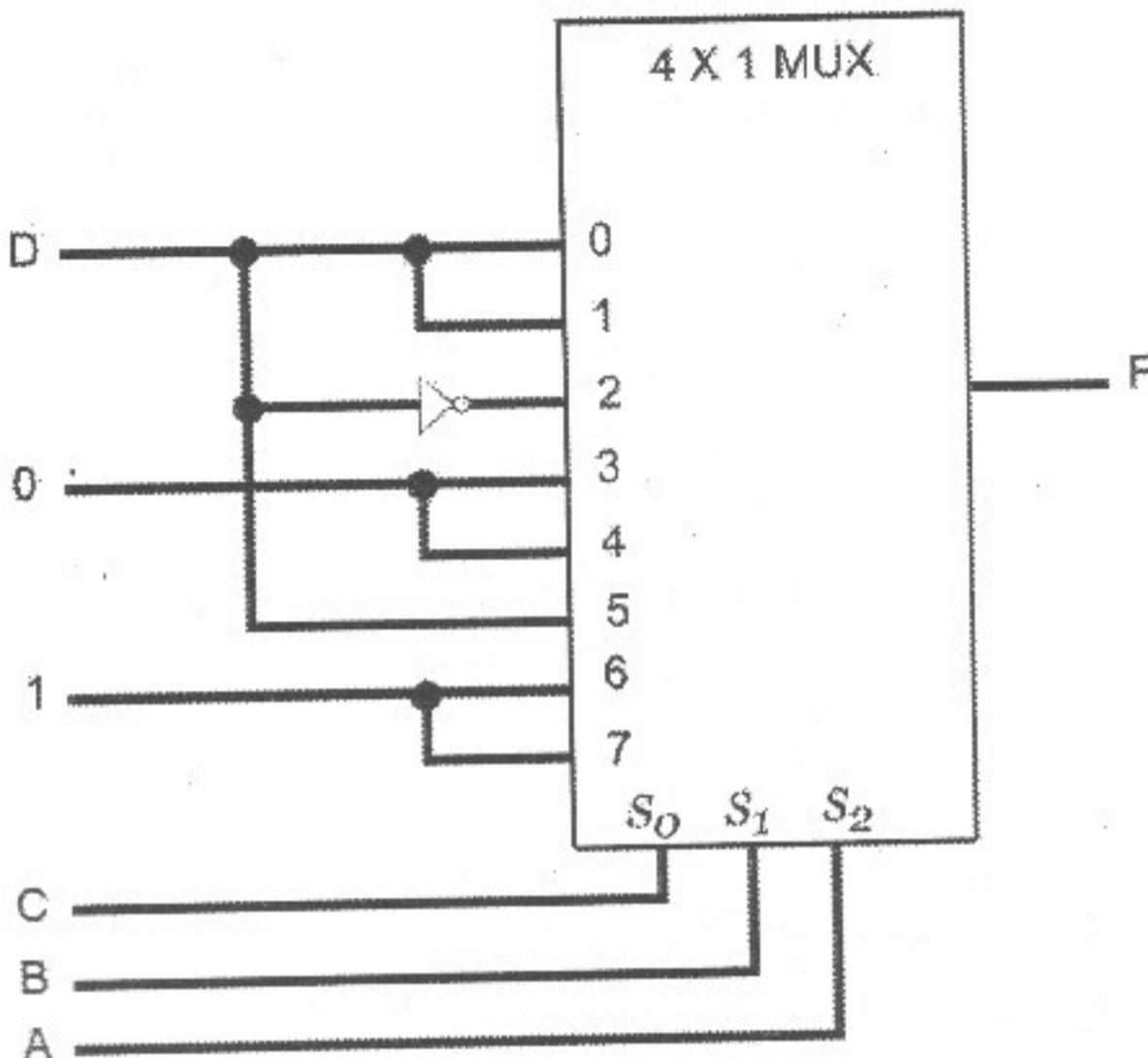
شماره ۳ زمانی مقدار خروجی اش برابر I_2 می‌شود که $S_0 = 0$ و $S_1 = 1$ باشد.



مولتی پلکسر ۲ به ۱ چهارتایی

فعال ساز E

مدار منطقی

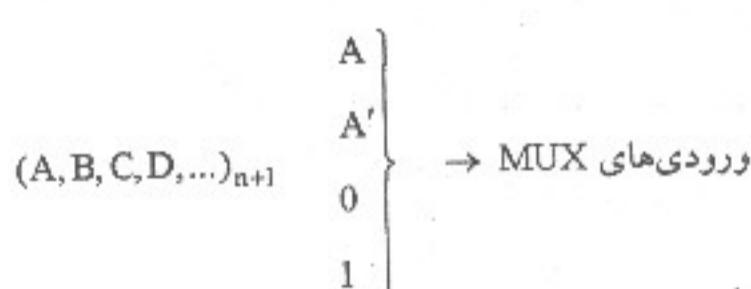


پیاده‌سازی یک تابع F ورودی با یک مولتی‌پلکسور

طراحی به کمک **multiplexer**

چهار راه متفاوت برای طراحی به کمک mux وجود دارد که عبارتند از:

(الف) طراحی به کمک بیت با ارزش (روش $n-1$)



اگر بخواهیم با مولتی‌پلکسور یک تابع n متغیره را تولید کنیم (طراحی کنیم) باید $n-1$ متغیر به خطوط select اختصاص دهیم و ورودی‌های مولتی‌پلکسور را به صورت A و A' و 0 و 1 در آوریم بنابراین برای پیاده‌سازی یک تابع n متغیره باید از یک مولتی‌پلکسور $2^{n-1} \times 1$ استفاده کنیم. قبل از هر چیز باید توجه داشت که تابع باید به صورت میترم‌ها در آورده شود.

باید توجه داشت که هیچ لزومی ندارد که با ارزش‌ترین بیت را به ورودی بدهیم و $n-1$ متغیر باقی مانده را به خطوط select بلکه متظور از توضیحات فوق این است که باید یکی از متغیرها را به ورودی بدهیم و مابقی را به خطوط select.

نکته: برای طراحی مدار به کمک مالتی‌پلکسور، به تعداد خروجی‌ها احتیاج به مولتیپلکسور داریم.

مثال:

یک میز رای گیری داریم که سه نفر رای دهنده دارد (A,B,C) زمانی چراغ رای گیری روشن می شود که حداقل دو نفر جواب مثبت بدهند. به کمک یک MUX آن را طراحی کنید.

$$\text{تعداد متغیرها} \quad n = 3 \rightarrow n - 1 = 2$$

$$\text{select } 2^{n-1} = 2^2 = 4$$

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

	I ₀	I ₁	I ₂	I ₃
A'	0	0	0	1
A	0	1	1	1
	0	A	A	1

اگر مقادیر ستون I_n برای A و A' هر دو یک بود مقدار واقعی I_A است.

اگر مقادیر ستون I_n برای A و A' هر دو صفر بود مقدار واقعی I_n = 0 است.

ب) طراحی به کمک یست کم ارزش (روش n-1)

تذکرہ: در این روش‌ها تنها مشکل شماره‌گذاری جدول تغییر می‌کند.

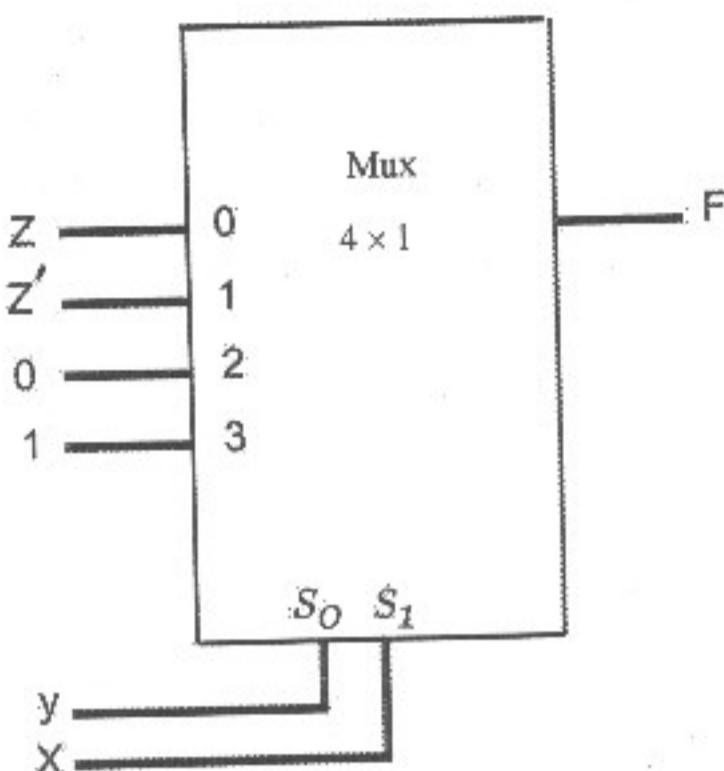
مثال:

تابع f(x,y,z) = $\sum(1,2,4,5)$ را به کمک MUX پیاده‌سازی کنید. می‌خواهیم این بار به جای این که x را به ورودی‌ها بدهیم (با ارزش ترین یست) به جایش z (کم ارزش ترین یست) را به ورودی‌ها اختصاص دهیم.

بنابراین در تشکیل جدول باید در شماره‌گذاری تغییراتی بدهیم.

I ₀	I ₁	I ₂	I ₃
z'	2	6	4
z	3	7	5
z	z'	0	1

مدار منطقی



ج) طراحی با بیت انتخابی: (روش n-1)

در این روش ابتدا یک بیت را برای اختصاص به خطوط ورودی انتخاب می‌کنیم. حال باید حالت‌ها مختلف n بیت باقی مانده را از صفر تا $2^n - 1$ در نظر بگیریم. و برای هر حالت ایجاد شده در جدول درستی دو ردیف وجود دارد که باید در این دو ردیف مقدار بیت انتخاب شده به عنوان ورودی را با خروجی مقایسه کنیم در این مقایسه چهار حالت ممکن است پیش بیاید که مقادیر مربوط به خطوط ورودی را ایجاد می‌کنند (در واقع همان I_0 تا I_{2^n-1}) این حالات عبارتند از:

- ۱- اگر بیت انتخابی و خروجی با هم برابر باشند باید به خط ورودی MUX بیت انتخابی اختصاص داده شود.
- ۲- اگر بیت انتخابی و خروجی و مخالف هم باشند به خط ورودی MUX باید not شده بیت انتخابی را به خط ورودی اختصاص دهیم.
- ۳- اگر در مقایسه بیت خروجی و بیت انتخابی مشاهده شد که در هر دو ردیف خروجی یک شده باید به خط ورودی MUX مقدار منطقی یک اختصاص داده شود.
- ۴- اگر در مقایسه بیت خروجی و بیت انتخابی مشاهده شود که در هر دو ردیف مقدار خروجی صفر شده باید به خط ورودی MUX (I_0 تا I_{2^n-1}) مقدار منطقی صفر اختصاصی داده شود.

مثال: به کمک MUX تابع $f(a, b, c) = \sum(1, 5, 6, 7)$ را طراحی کنید.

حل:

در این مثال متغیر a را برای اختصاص به خطوط ورودی MUX در نظر می‌گیریم. بنابراین باید برای دو متغیر باقی مانده حالت‌های مختلف را در نظر بگیریم. و با توجه به این حالات عملیات مقایسه را انجام می‌دهیم.

ابتدا حالت صفر به صفر را برای ac در نظر بگیرید: (تعیین ورودی I_0) برای این حالت دو ردیف صفر و دو را می‌توان در نظر گرفت. اگر b که به عنوان بیت انتخابی محسوب می‌شود را با خروجی در آن دو ردیف مقایسه کنیم می‌بینیم که در هر دو ردیف بدون توجه به مقدار b خروجی برابر صفر است بنابراین باید به خط ورودی I_0 مقدار صفر اختصاص داده شود.

حال حالت صفر یک را در نظر می‌گیریم یعنی در اصل می‌خواهیم مقدار ورودی I_1 را بذست آوریم: همان‌طور که مشاهده می‌شود باید ردیف‌های یک و سه یا هم مقایسه شوند در این مقایسه می‌بینیم که در ردیف یک مقدار یک با خروجی f به عکس می‌باشد و در ردیف سه نیز همین حالت برقرار است بنابراین باید به ورودی I_1 مقدار کار را اختصاص دهیم. با همین منوال مشخص می‌شود که مقدار I_2 برابر ۰ و I_3 برابر یک می‌باشد، بنابراین:

	a	b	c	f
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	0
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

۵: در این روش تمامی متغیرها به خطوط select اختصاص داده می‌شود. و با توجه به جدول درستی ورودی‌ها مقادیری را به خود اختصاص می‌دهند به این صورت که مقدار خروجی ردیف صفر به ورودی I_0 اختصاص داده می‌شود و مقدار خروجی $I_1 - I_2 - I_3$ به I_0 اختصاص داده می‌شود.

مثال: تابع $f(a, b, c, d) = \pi(0, 3, 4, 7, 10; 11, 15)$ را به کمک mux پیاده‌سازی کنید.

	a	b	c	d	f
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	1
10	1	0	1	0	0
11	1	0	1	1	0
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	0

به کمک مولتی پلکسر و چهار روش آن مداری طراحی کنید که زوج و فرد بودن یک عدد سه بیتی را مشخص کند.
روش الف: (روش ۱-۱) (به کمک بیت با ارزش)

	I_0	I_1	I_2	I_3
A'	①	1	②	3
A	④	5	⑥	7
	1	0	1	0

روش ب: (روش I - n) (به کمک یست کم ارزش)

	I_0	I_1	I_2	I_3
C'	①	②	④	⑥
C	1	3	5	7
	C'	C'	C'	C'

روش ج: (روش I - n) (به کمک یست انتخابی)

در این روش B را به عنوان یست انتخابی در نظر گرفتیم حال باید حالات مختلف A و C را تحلیل کنیم.

	A	B	C	f
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	0

$I_0 = 1 \leftarrow AC = 00$

$I_1 = 0 \leftarrow AC = 01$

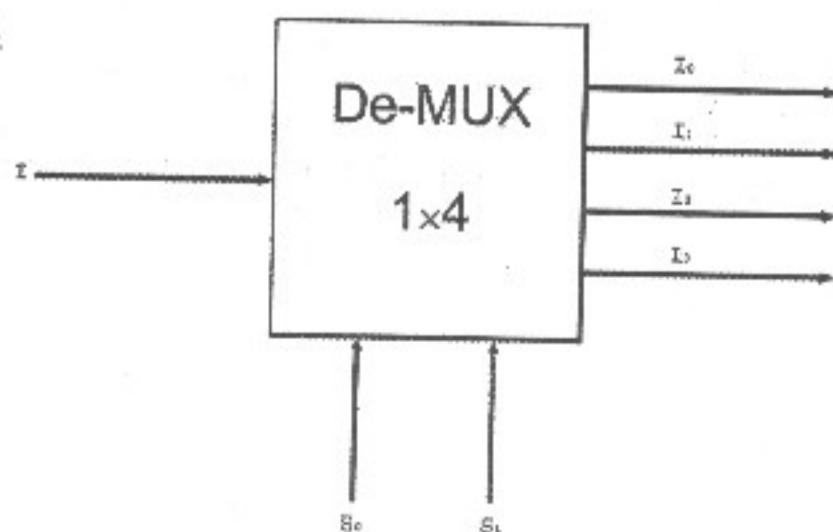
$I_2 = 1 \leftarrow AC = 10$

$I_3 = 0 \leftarrow AC = 11$

روش د:

	A	B	C	f
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	0

دی مالتی پلکسر DMUX یا De Multi Plexer
 عکس MUX می‌باشد به این معنی که تنها یک خط ورودی دارد و 2^n خط خروجی. در یک دی مالتی پلکسر $n \times 1$ از خط select دهیم مشخص می‌کنیم که مقدار ورودی باید استفاده می‌شود. تا یک ورودی را به خروجی منتقل کند. یعنی با ورودی که به خطوط select می‌دهیم داده شود. در کدامیک از خروجی‌ها نمایش داده شود.



حافظه و منطق برنامه ریز

ROM

ROM از عناصر مدارهای ترکیبی است. مداری است شامل n ورودی و m خروجی مثلاً (32 \times 8) ROM دارای 5 ورودی $(32 \times 2^5)^n$ و 8 خروجی است. به ازای هر n ورودی، 2^n ترکیب متناوب وجود دارد که هر کدام آنها یک آدرس گویند. هر ترکیبی از خطوط خروجی یک نامیده می‌شود. یعنی با توجه به آدرسی که در ورودی به آن می‌دهیم در خروجی تعداد صفر و یک وارد می‌شود.

PLA

هنگامی که تعداد حالات بی‌اهمیت در یک مدار با تعداد خروجی زیاد بالا باشد به جای استفاده از ROM می‌توان از PLA استفاده کرد. در ROM تمام ترکیبات ورودی (میترمها) به صورت کامل موجود بود اما در PLA تنها جملات حاصل ضرب وجود دارد یعنی ها به صورت کامل تولید نمی‌شود. منظور از جملات حاصل ضرب SOP ها هستند. به جای دیکدر در PLA از گیت‌های AND استفاده می‌وشد و از هر کدام از این گیت‌ها می‌توان برای تولید جملات حاصل ضرب متغیرهای ورودی استفاده کرد. گیت‌های AND و OR موجود در PLA از ابتدا همراه با اتصالات ماینشن ساخته می‌شوند تا قطع اتصالات مناسب توابع موردنیاز به فرم حاصل ضرب پیاده‌سازی می‌شوند یک PLA شامل n خط ورودی و m خط خروجی و k جمله حاصل ضرب و m جمله حاصل جمع می‌باشد تعداد اتصالات در یک PLA از رابطه زیر بدست می‌آید.

$$(2n \times k) + (m \times k) + m$$

برمبنای اینکه از یک ROM(8 \times 6) استفاده کنیم می‌توانیم از یک ROM(8 \times 4) استفاده کنیم.

ROM انواع

سه نوع ROM داریم : EEPROM , EPROM , PROM : نقشه موردنظر را به شرکت سازنده داده می‌شود. شرکت اتصالات را می‌سوزاند و مدار موردنظر را می‌سازد. در PROM نمی‌شود برای بار دوم برنامه ریزی کرد.

PROM : EPROM که قابلیت یک شدن دارد یعنی می‌توان یکبار اتصال برقرار کرد بار بعدی حذف کرد.

EEPROM : در PROM دیگر اتصالات را با اشعه ماوراء بنفش قطع می‌کنیم تا که با جریان الکتریکی این کار را انجام می‌دهیم.

a	b	c	f ₁	f ₂	f ₃	f ₄	f ₅	f ₆
0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1
0	1	0	0	0	0	1	0	0
0	1	1	0	0	1	0	0	1
1	0	0	0	1	0	0	0	0
1	0	1	0	1	1	0	0	1
1	1	0	1	0	0	1	0	0
1	1	1	1	1	0	0	0	1

پیاده‌سازی منطق ترکیبی با استفاده از منطق ROM

برای پیاده‌سازی یک مدار ترکیبی با n ورودی و m خروجی بر یک ROM از نوع $n \times 2^m$ نیاز داریم.

برنامه‌ریزی ROM

منظور قطع کردن اتصالات در ROM می‌باشد برای این کار باید جدول برنامه ROM را تهیه کرد. منظور از تهیه جدول برنامه ROM این است که باید مشخص کنیم کدامیک از خطوط خروجی دیگر باید وارد OR شوند. در یک ROM در ابتدا تمام خطوط خروجی وارد تمام گیت‌های OR می‌شوند به این منظور که برای هر یک از m گیت OR، 2^n ورودی خواهیم داشت. اما پس از بدست آوردن جدول برنامه ROM خطوطی را که اتصالشان را لازم نداریم قطع می‌کنیم (این کار را برای تمام گیت‌های OR انجام می‌دهیم). اما در صورتی که بخواهیم پیاده‌سازی را به کمک IC ROM انجام دهیم تنها کافی است جدول برنامه بدست آمده را به شرکت سازنده ROM ارائه دهیم تا مدار ترکیبی موردنظر ما را به وجود آورد.

مثال: به کمک یک ROM مداری طراحی کنید که عدد ورودی 3 بیتی را دریافت کند و در خروجی مربع آن را نمایش دهد.

قوه تکمیل جدول به این صورت است که جمله‌های ضرب SOP‌ها را به کمک ورودی‌ها می‌سازیم یعنی در شکل 100 و یا حالت بی‌همیت. به این معنا که اگر جمله ضرب متغیر ورودی را در خود داشت به جای آن 1 و اگر مکملش را داشت به جایش صفر و در غیر این صورت (-) حالت بی‌همیت را می‌گذاریم. در ستون خروجی‌ها اگر تابع f_1 از SOP‌های k, L, \dots تشکیل شده بود ($f_1 = SOP_k + SOP_L + \dots$) در آن ردیف‌ها (k, L, \dots) می‌گذاریم و در ردیف‌های باقی ماتده (-) می‌گذاریم. اگر تابع f_i خودش را استفاده کردیم ذر ردیف انتهای مقادیر T را می‌نویسیم و اگر مکملش را استفاده کردیم در آن ردیف C می‌نویسیم تنها باید توجه داشت که اگر در ستونی C نوشته بود باید در انتهای پیاده‌سازی، آن خروجی را متمم کنیم حال برای قسمت نهایی پیاده‌سازی باید تمام ورودی را هم خودشان و هم مکملشان را ایجاد کنیم.

سپس به تعداد جملات حاصلضرب، گیت AND و به تعداد خروجی، گیت OR قرار دهیم. در انتهای خروجی OR‌ها همان توابع موردنیاز ما می‌باشند.

PAL

قطعه منطقی برنامه‌پذیری است با ارائه OR بدون تغییر و ارائه AND برنامه‌پذیر مشاهده می‌شود.

تفاوت PLA, ROM

در این است که در ROM میترم‌ها تولید می‌شود اما در PLA، SOP‌ها تولید می‌شود. در طراحی با ROM خود تابع را می‌خواهیم اما در PLA تابع را به صورت ساده شده خواهیم داشت ROM و PLA در این است که هر دو تابع پیچیده با تعداد خروجی بالا را پیاده‌سازی می‌کنند.



طراحی به کمک PLA:

چون در طراحی به کمک SOP از PLA ها استفاده می کنیم بنابراین برای طراحی باید ابتدا تابع موردنظر را ساده کنیم. باید توجه داشت که در این نوع طراحی هم خود تابع و هم مکملش باید پیاده شود (سازده سازی توسط کارتونو ...) سپس فرضی را انتخاب می کنیم که کمترین تعداد SOP را در خود داشته باشد. سپس جملات مجزا را در تمام توابعی که می خواهیم به کمک PLA پیاده سازی کنیم، انتخاب می کنیم و جدولی به شکل زیر طراحی می کنیم.

اگر T بود به این معنی که تابعی که پیاده سازی کردیم مکملش است و اگر f بود به این معنی است که خود تابع را پیاده سازی می کنیم.

	متغیرهای ورودی	خروجی
SOP	A B C ...	F ₁ F ₂ F ₃ ...
SOP ₁		
SOP ₂		مقدار آن یا C است یا T

پیاده سازی به کمک PAL با مشخصات فوق بسیار آسان تر از پیاده سازی به کمک PLA می باشد. اما انعطاف پذیری PAL بسیار کمتر از PLA می باشد.

مشخصات PAL

۱- کلیه ورودی ها و مکمل آنها موجود است.

۲- دارای بردار AND قابل برنامه ریزی می باشد (تعدادشان کمتر از تعداد n هاست)

۳- دارای بردار OR با ورودی های ثابت می باشد. باید توجه داشت در طراحی به کمک PAL برخلاف PLA نمی توان یک جمله ضرب را بین دو یا چند OR به اشتراک گذاشت.

بنابراین هر تابع بدون توجه به توابع دیگر ساده می شود.

مثال:

تابع f_1, f_2 را به کمک PLA پیاده سازی کنید.

SOP	A	B	C	f_1	f_2
A B	1	1	-	1	1
A C	1	-	1	1	1
B C	-	1	1	1	-
$A'B'C'$	0	0	0	-	1
				C	T

ساخته مان داخلی ROM

یک ROM از یک دیکدر و تعدادی گیت OR تشکیل شده است. دیکدر هم به وسیله تعدادی AND پیاده سازی شده می خواهیم یک ROM(16×4) طراحی کنیم این ROM، 4 ورودی و 4 خروجی دارد. باید از یک دیکدر 16×4 استفاده کنیم هر کدام از انتقالات یک بیوز سر راه خود دارند.

هر کدام از میترم ها که لازم باشد ارتباط برقرار است.

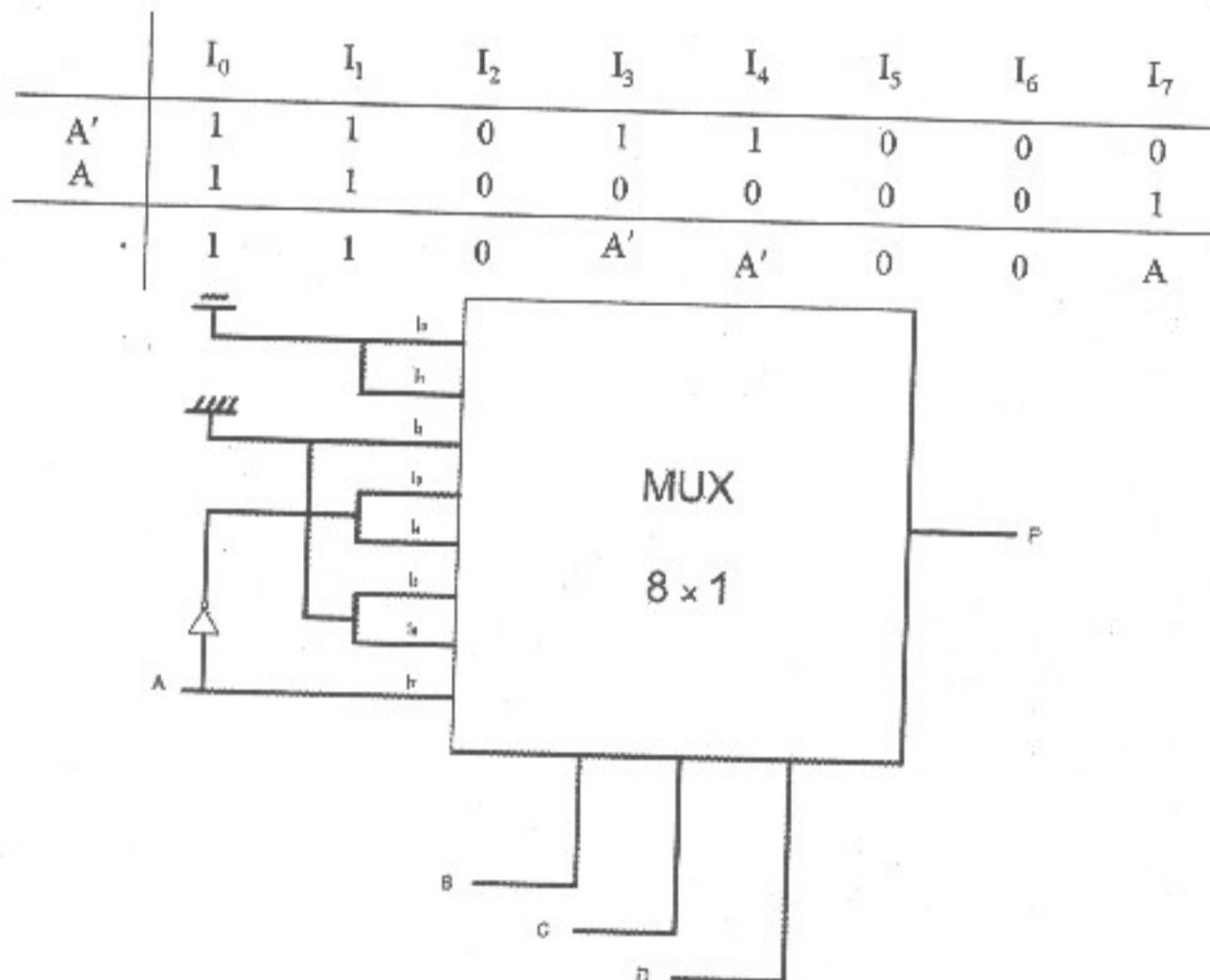
مجموعه مسائل فصل سوم:

۱ - تابع را به کمک یک MUX پیاده سازی کنید.

$$F(A, B, C, D) = \Sigma(0, 1, 3, 4, 8, 9, 15)$$

$$n+1=4 \Rightarrow n=3$$

$$2^3 = 8$$



۲ - به کمک یک دیکدر و یک گیت OR جدول درستی زیر را پیاده سازی کنید.

	x	y	z	F
0	0	0	0	1
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	0

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

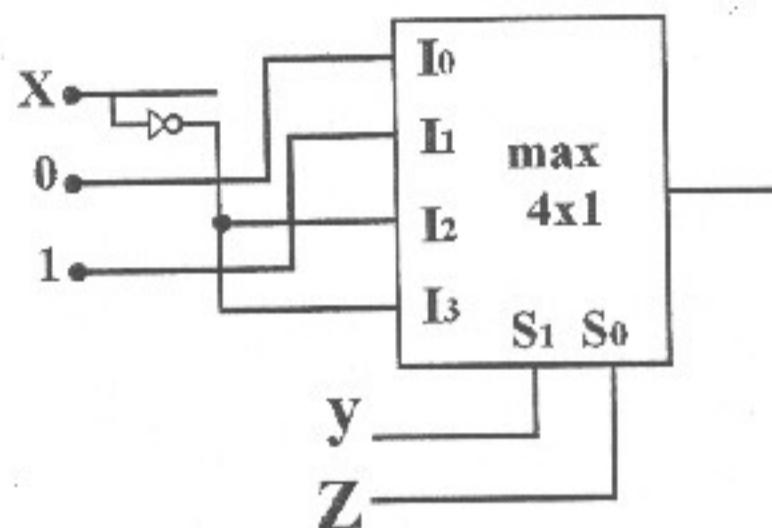
$$F = x'y' + z'$$

مدار منطقی

۳- به کمک یک مالتی پلکسر تابع f را طراحی کنید.

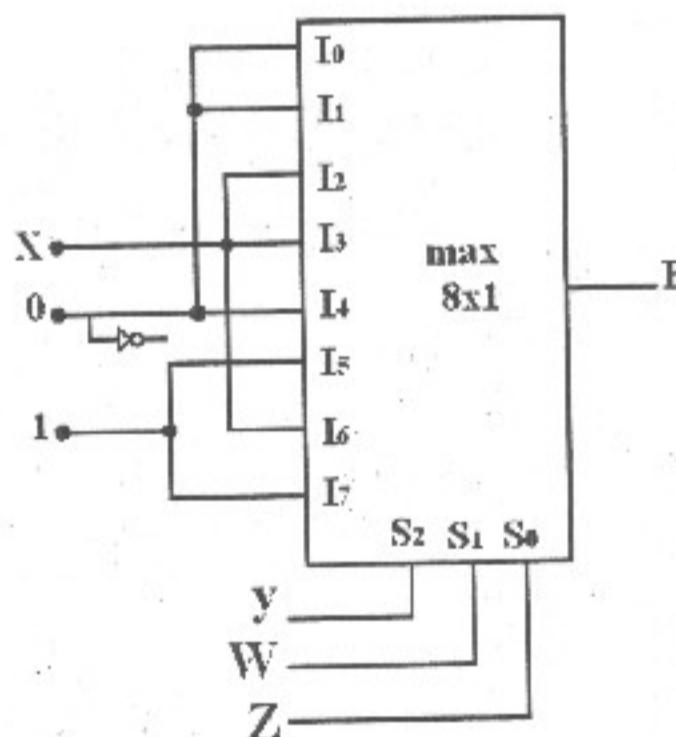
x	y	z	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

ورودی‌ها	I_0	I_1	I_2	I_3
X'	0	1	2	3
X	4	5	6	7
	0	1	X'	X'



۴- مداری با چهار ورودی طراحی کنید که یک خروجی داشته باشد. خروجی در زمانی یک می‌شود که $w = x \cdot y$, یک باشد و z زمانی که یک باشد این مدار را به کمک مولتی پلکسر طراحی کنید.

x	y	w	z	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1



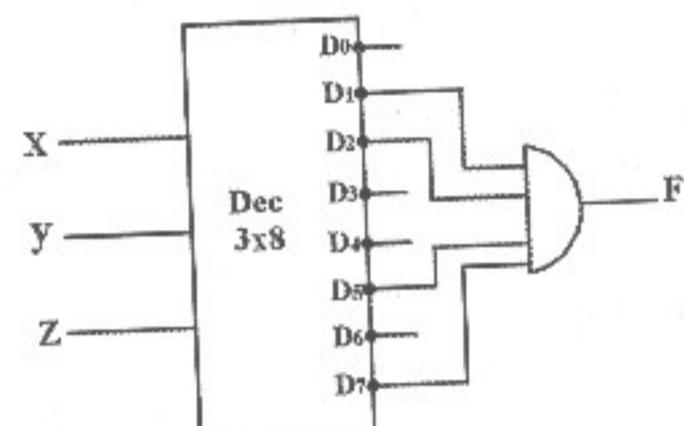
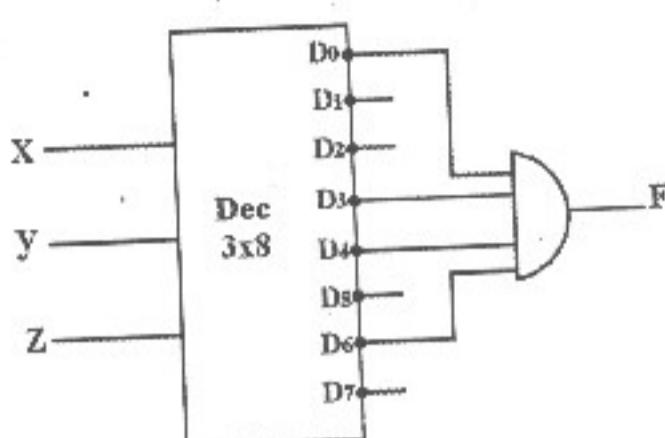
ورودی‌ها	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
X'	0	1	2	3	4	5	6	7
X	8	9	10	11	12	13	14	15
	0	0	X	X	0	1	X	1

مدار منطقی

۵- راجع به طراحی زیر تحقیق کنید: فرض کنید تابع F بصورت زیر تعریف شده باشد:

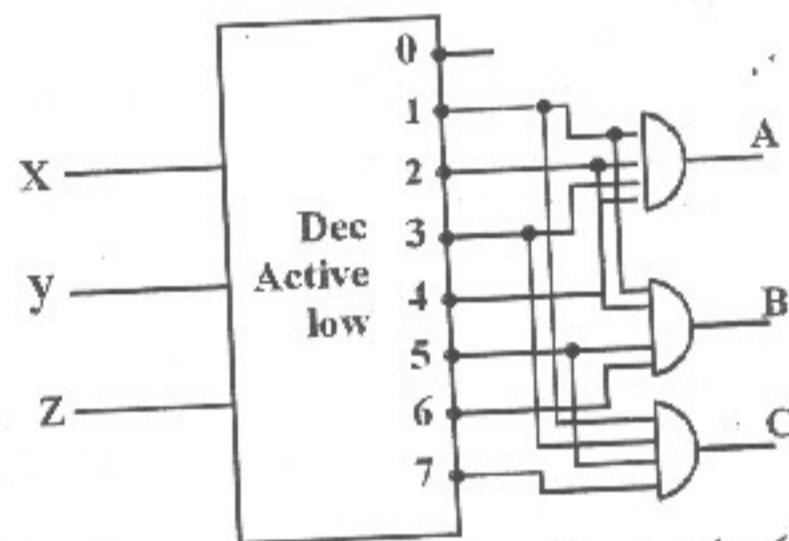
می خواهیم به کمک یک دیکدر Active low و گیت های AND و NAND تابع F را پیاده سازی کیم.

x	y	z	F
0	0	0	0
1	0	0	1
2	0	1	1
3	0	1	0
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1



۶- به کمک یک دیکدر Active low NAND و گیت ۳x2 مداری طراحی کنید که مکمل ۲ ورودی ها را در خروجی نمایش دهد؟ (مدار دارای ۳ ورودی باشد)

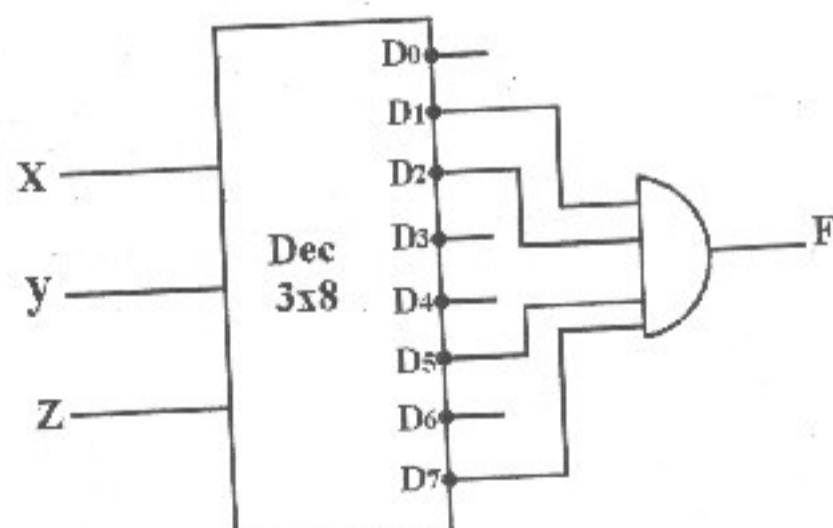
x	y	z	A	B	C
0	0	0	0	0	0
1	0	0	1	1	1
2	0	1	1	1	0
3	0	1	1	0	1
4	1	0	1	0	0
5	1	0	0	1	1
6	1	1	0	0	1
7	1	1	0	0	0



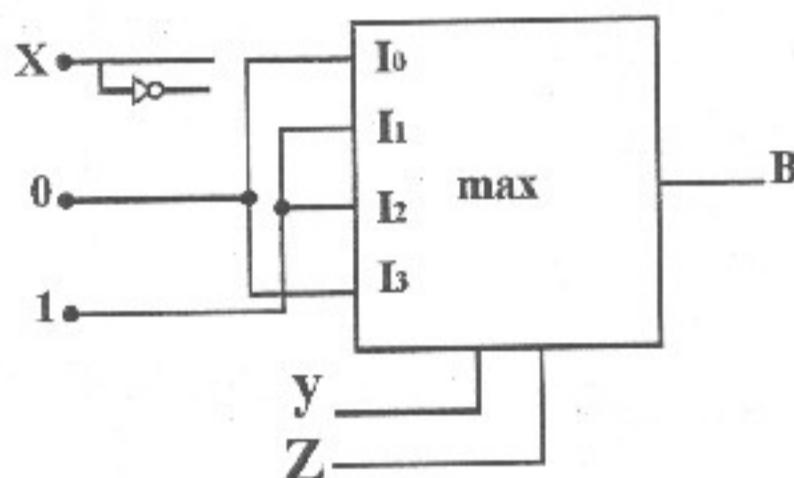
۷- به کمک یک مولتی پلکسر مداری طراحی کنید که خروجی، ورودی به اضافه یک باشد. (توضیحات: اگر ورودی 7 بود خروجی صفر می شود. مدار دارای ۳ ورودی باشد.)

x	y	z	A	B	C
0	0	0	0	0	1
1	0	0	0	1	0
2	0	1	0	0	1
3	0	1	1	0	0
4	1	0	1	0	1
5	1	0	1	1	0
6	1	1	0	1	1
7	1	1	0	0	0

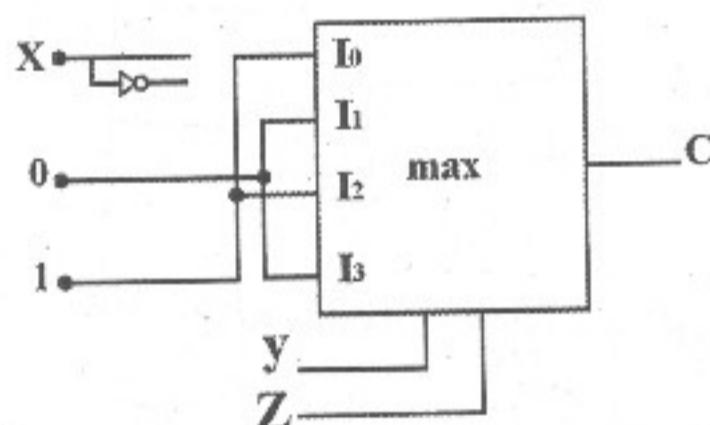
ورودی ها	I_0	I_1	I_2	I_3
X'	0	1	2	3
X	4	5	6	7
	X	X	X	X'



ورودی‌ها	I_0	I_1	I_2	I_3
X'	0	1	2	3
X	4	5	6	7
	0	1	1	0

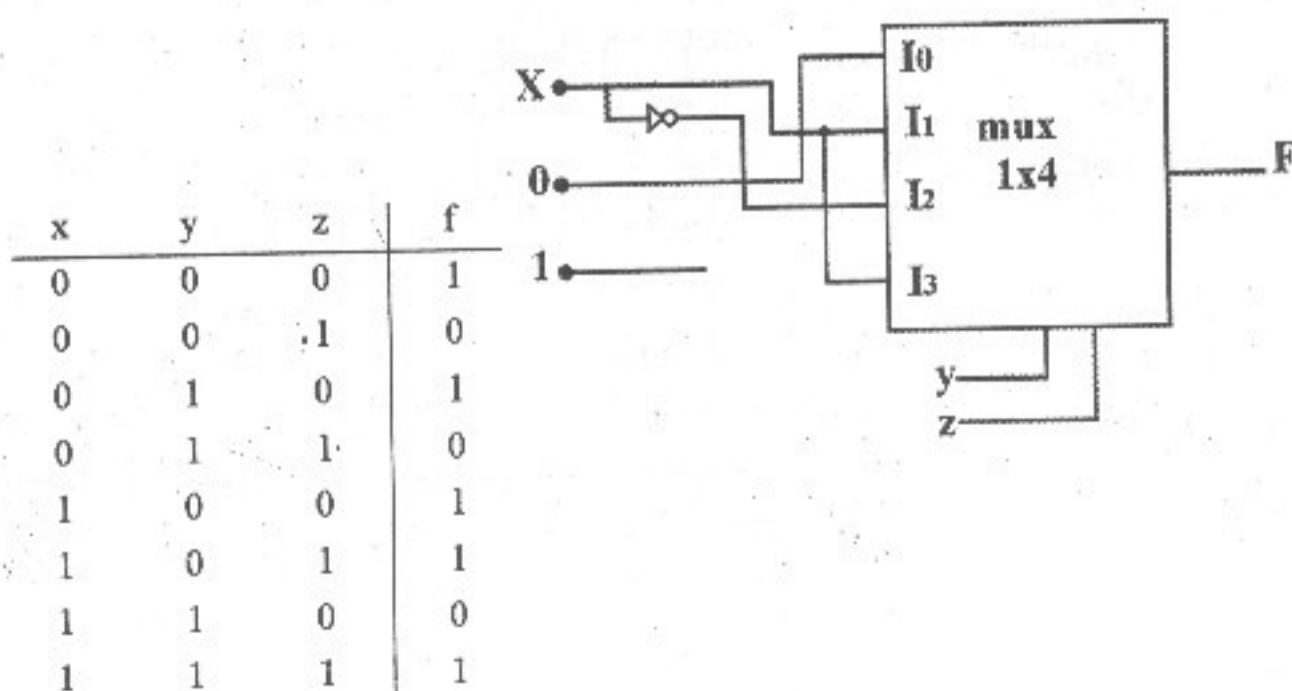


ورودی‌ها	I_0	I_1	I_2	I_3
X'	0	1	2	3
X	4	5	6	7
	1	0	1	0



۲۸- به کمک یک mux تابع f را طراحی کنید.

آنایی که f آنها مساوی یک است دورشان خط می‌کشیم.



	I_0	I_1	I_2	I_3
x'	(0)	1	(2)	3
x	(4)	(5)	6	(7)

مدار منطقی

- ۹ - به کمک یک Mux تابع F را طراحی کنید.

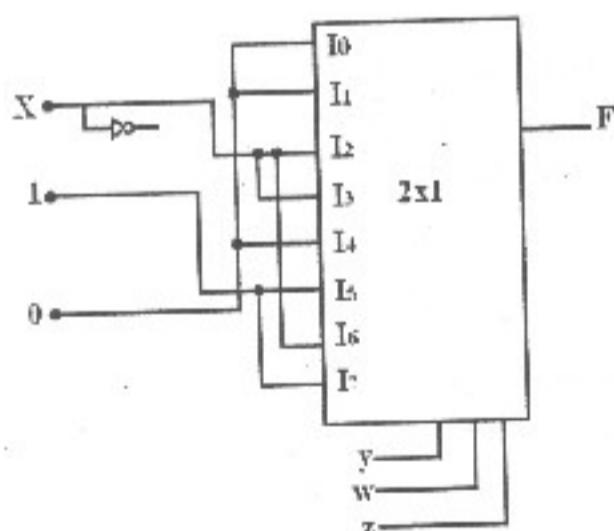
مداری است با چهار ورودی و یک خروجی، خروجی در زمانی یک می شود که یا w, x , هر دو یک باشند و یا y, z , هر دو یک باشند ترتیب ورودی های زیر به شکل زیر می باشد.

 x y w z

$$2^4 = 16$$

x	y	w	z	f
0	0	0	0	0
0	0	1	0	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

	I ₀	I ₁	I ₂	I ₃	I ₄	I ₅	I ₆	I ₇
x'	0	1	2	3	4	5	6	7
x	8	9	10	11	12	13	14	15
	0	0	X	X	0	1	X	1



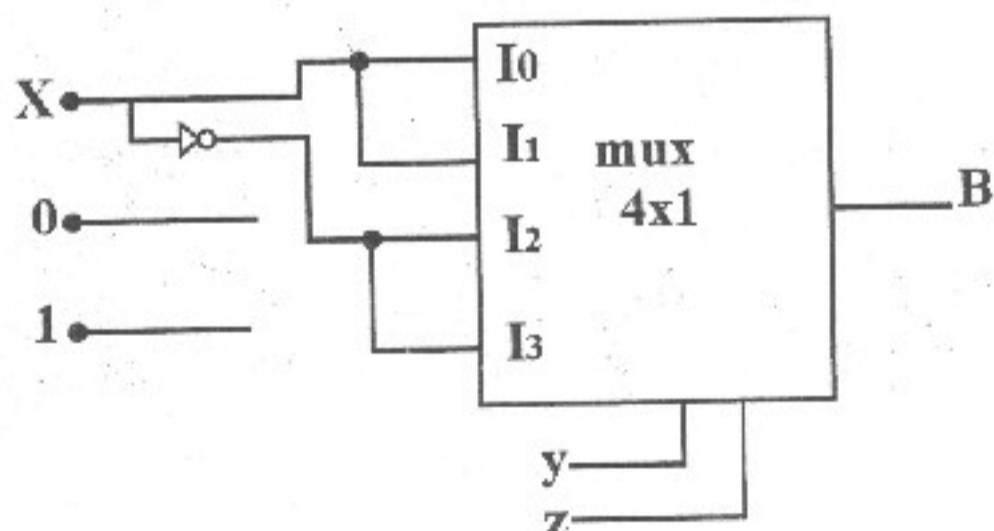
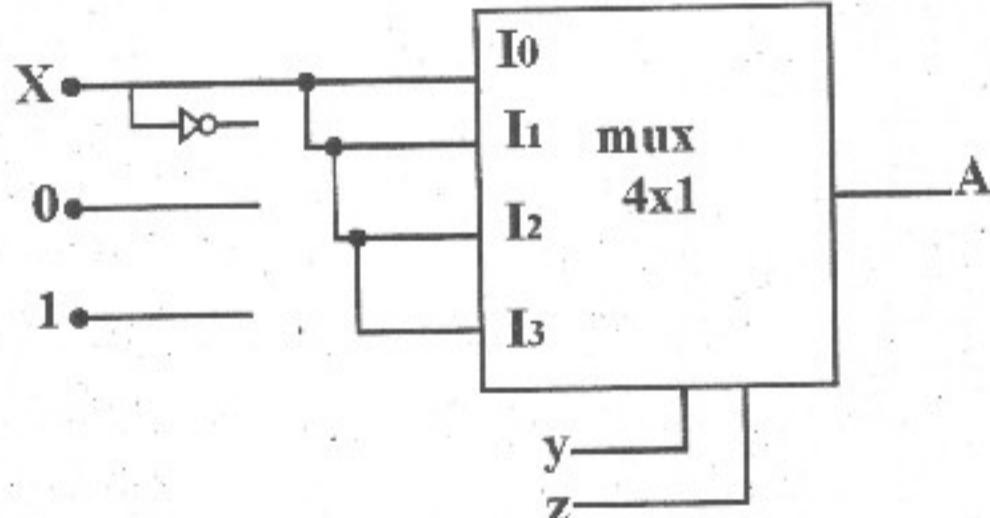
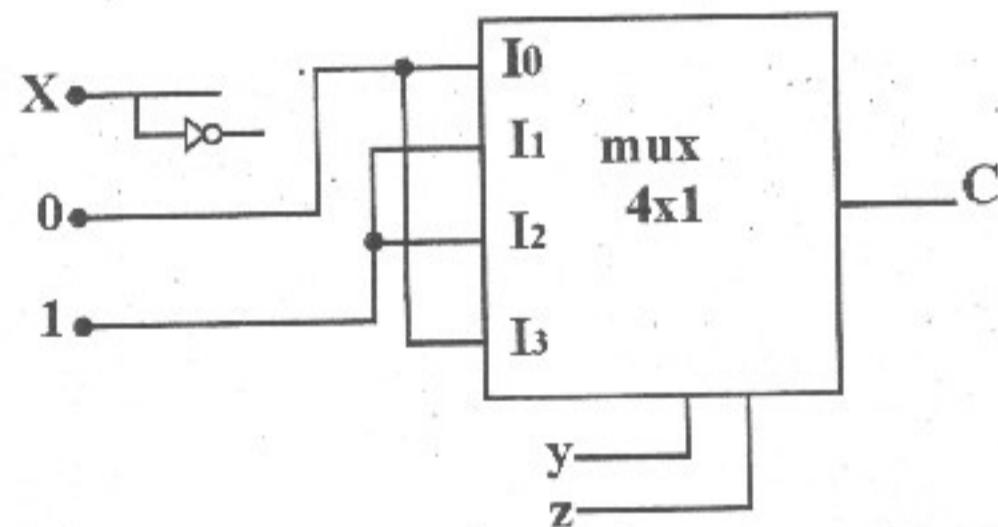
۱۰ - مدار تبدیل کد باینری به کد gray را به سه بیت ورودی رابه کمک Mux طراحی کنید؟

x	y	z	A	B	C
0	0	0	0	0	0
1	0	0	0	0	1
2	0	1	0	1	1
3	0	1	1	0	0
4	1	0	1	1	0
5	I	0	1	1	1
6	1	1	0	1	0
7	1	1	1	0	0

	I_0	I_1	I_2	I_3
x'	0	1	2	3
x	④	⑤	⑥	⑦

	I_0	I_1	I_2	I_3
x'	0	1	②	3
x	4	③	6	7

	I_0	I_1	I_2	I_3
x'	0	①	②	3
x	4	⑤	⑥	7



مدار ملطفی

۱۱- مداری طراحی کنید که ورودی آن یک عدد سه بیتی باشد و خروجی آن مربع ورودی باشد. مطلوب است:

الف - پیاده‌سازی (عادی)

ب - پیاده‌سازی تمام NAND

ج - طراحی به کمک دیکدر Active high و گیت OR

د - طراحی به کمک دیکدر Active low NAND و گیت

ه - طراحی به کمک مولتی پلکسر (MUX)

x	y	z	A	B	C	D	E	F
0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1
0	1	0	0	0	0	1	0	0
0	1	1	0	0	1	0	0	0
1	0	0	0	1	0	0	0	0
1	0	1	0	1	1	0	0	1
1	1	0	0	1	0	1	0	0
1	1	1	1	1	0	0	0	1

$$7^2 = 49 \rightarrow 110001$$

به تعداد خروجی برای طراحی جدول کارنو رسم می‌کنیم.



تعداد خانه‌های جدول کارنو

- برای خروجی A جدول کارنو احتیاج نداریم زیرا تنها یک‌ها برای عدد 6 و 7 است که تنها در یک بیت اختلاف دارند و ورودی متناظر آن بیت حذف می‌شود.

- خروجی D هم مانند A است.

- خروجی E هم چون همه صفر است پس مساوی صفر است.

- خروجی F هم دقیقاً مانند ستون ورودی Z است پس مساوی Z می‌شود.

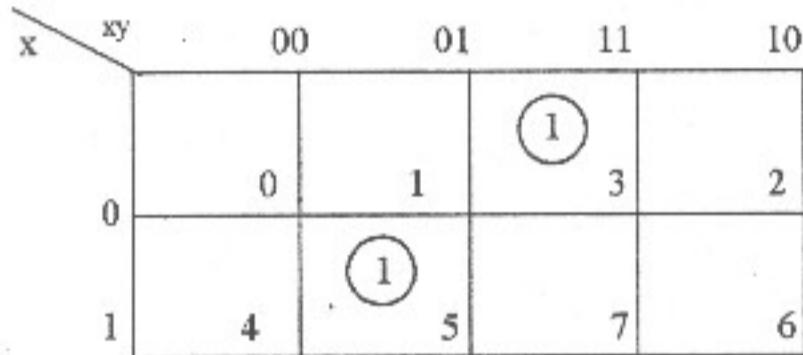
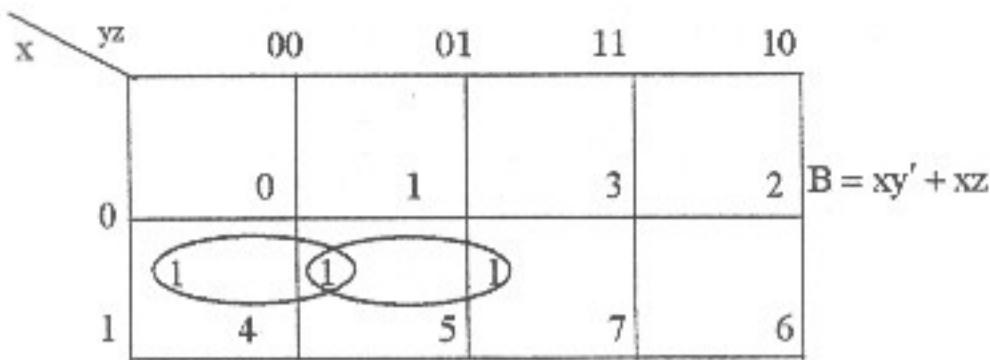
- برای خروجی B و C هم جدول کارنو رسم می‌کنیم.

$$A = xy$$

$$D = yz'$$

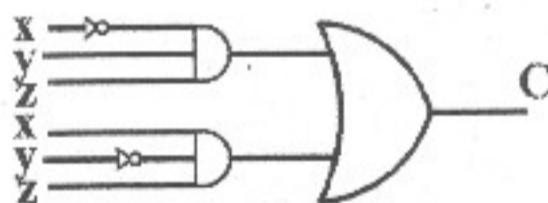
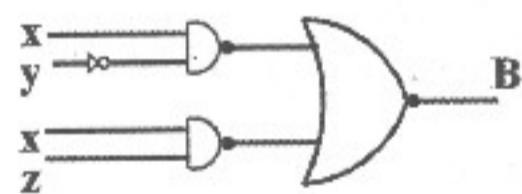
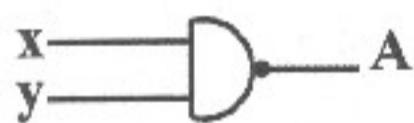
$$E = 0$$

$$F = z$$

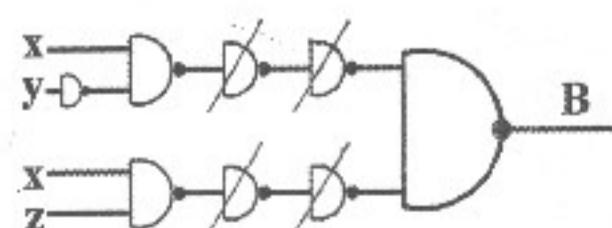
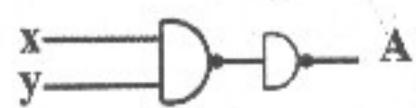


$$C = xy'z + x'y'z$$

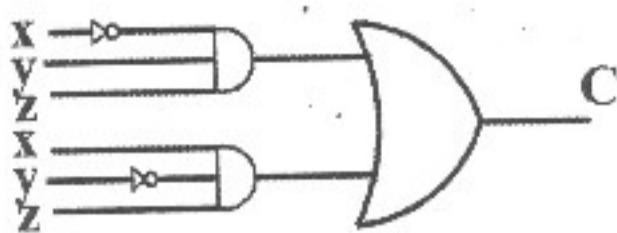
الف - پیاده‌سازی (عادی)


 0 ————— E
 z ————— F

ب - پیاده‌سازی تمام NAND



مدار منطقی

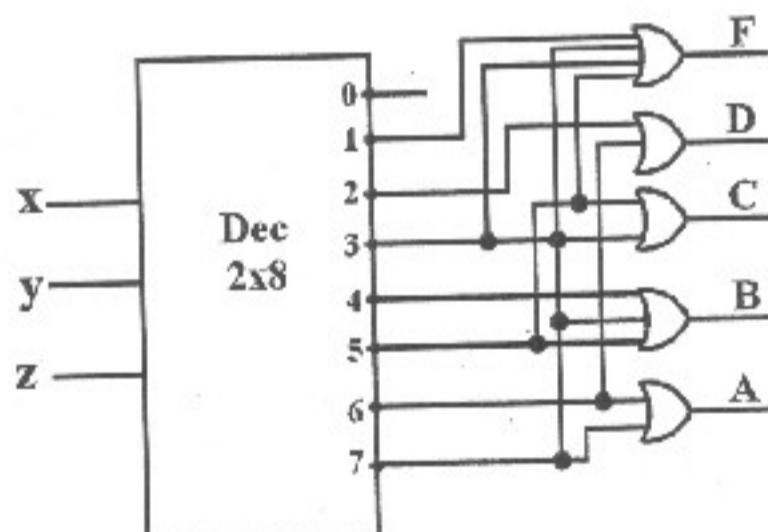


0 ————— E
z ————— F



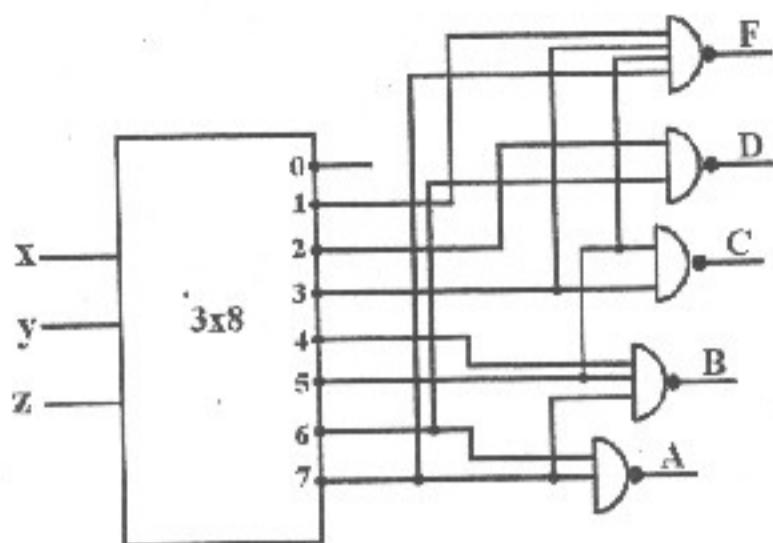
ج - طراحی به کمک دیکدر Active high و گیت OR
نکته:

- Dec {
- یک‌های خروجی \rightarrow OR \rightarrow Active high
 - صفرهای خروجی \rightarrow NOR \rightarrow Active high
 - صفرهای خروجی \rightarrow AND \rightarrow Active low
 - یک‌های خروجی \rightarrow NAND \rightarrow Active low



نکته: اگر بخواهیم به کمک دیکدر مداری طراحی کنیم که n خروجی داشته باشد باید از n گیت استفاده کنیم اما تنها یک دیکدر برای طراحی کافی است.

د - طراحی به کمک دیکدر Active low با گیت NAND



۵- طراحی به کمک مولتیپلکسر (MUX)

نکته: به تعداد خروجی مدار باید Mux داشته باشیم.

	I_0	I_1	I_2	I_3
x'	0	1	2	3
x	4	5	⑥	⑦
	0	0	x	x

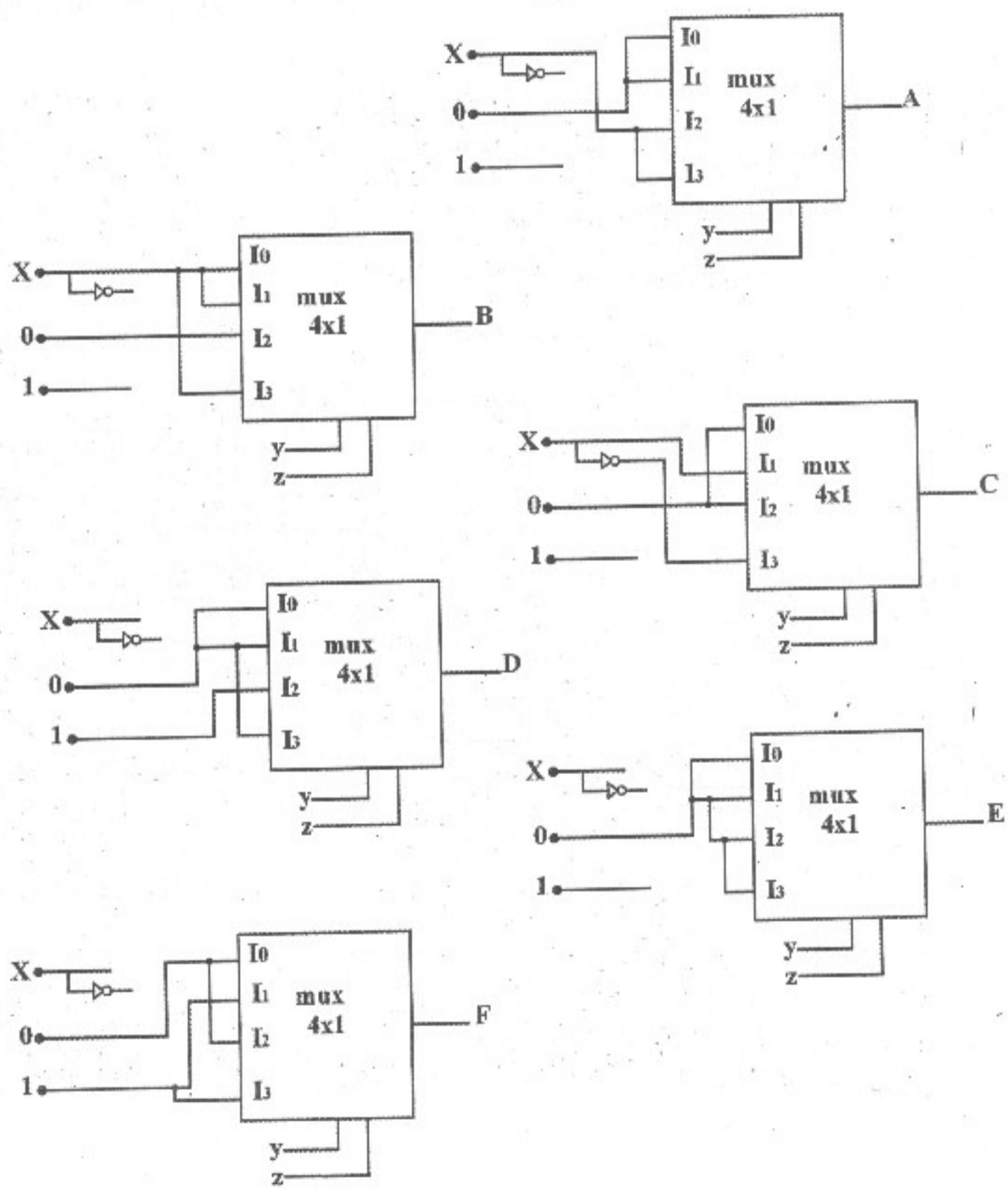
	I_0	I_1	I_2	I_3
x'	0	1	2	3
x	④	⑤	6	⑦
	x	x	0	x

	I_0	I_1	I_2	I_3
x'	0	1	2	③
x	4	③	6	7
	0	x	0	x'

	I_0	I_1	I_2	I_3
x'	0	1	②	3
x	4	5	⑥	7
	0	0	1	0

	I_0	I_1	I_2	I_3
x'	0	1	2	3
x	4	5	6	7
	0	0	0	0

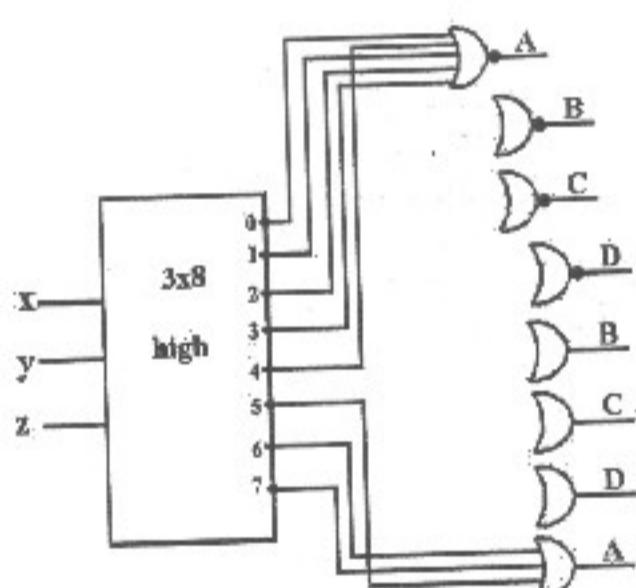
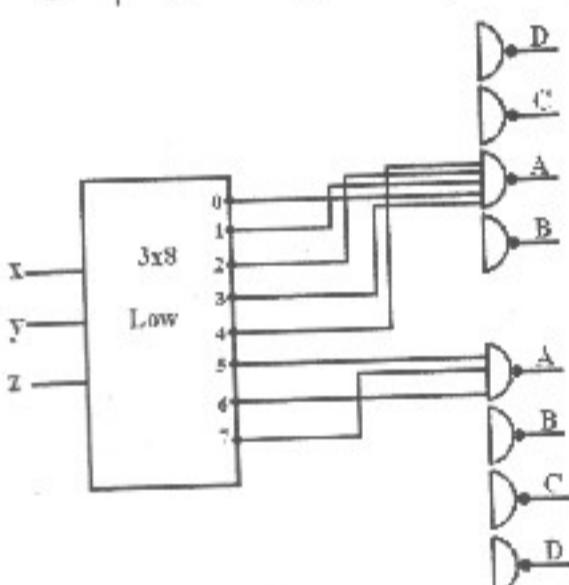
	I_0	I_1	I_2	I_3
x'	0	①	2	③
x	4	⑤	6	⑦
	0	1	0	1



مدار همنطقی

۱۲ - به کمک دیگرهای Active low و Active high و مالتی پلکسر مداری طراحی کنید که ورودی را (به صورت باپنری) به معادل ۳-EX آن تبدیل کند.

x	y	z	A	B	C	D
0	0	0	0	0	1	1
0	0	1	0	1	0	0
0	1	0	0	1	0	1
0	1	1	0	1	1	0
1	0	0	0	1	1	1
1	0	1	1	0	0	0
1	1	0	1	0	0	1
1	1	1	1	0	1	0



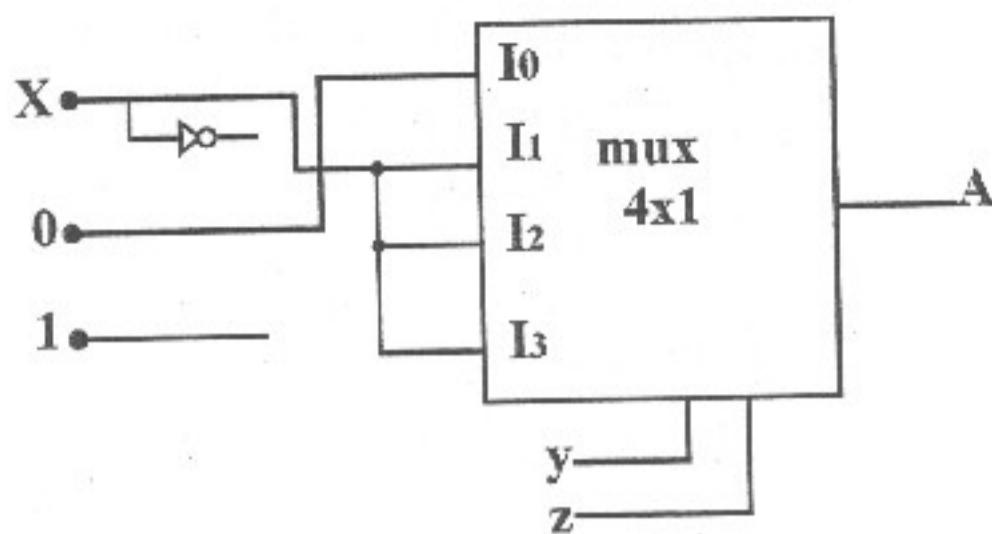
	I ₀	I ₁	I ₂	I ₃
x'	0	1	2	3
x	4	⑤	⑥	⑦
	0	x	x	x

	I ₀	I ₁	I ₂	I ₃
x'	0	①	②	③
x	④	5	6	7
	x	x'	x'	x'

	I ₀	I ₁	I ₂	I ₃
x'	①	1	2	③
x	④	5	6	⑦
	1	0	0	1

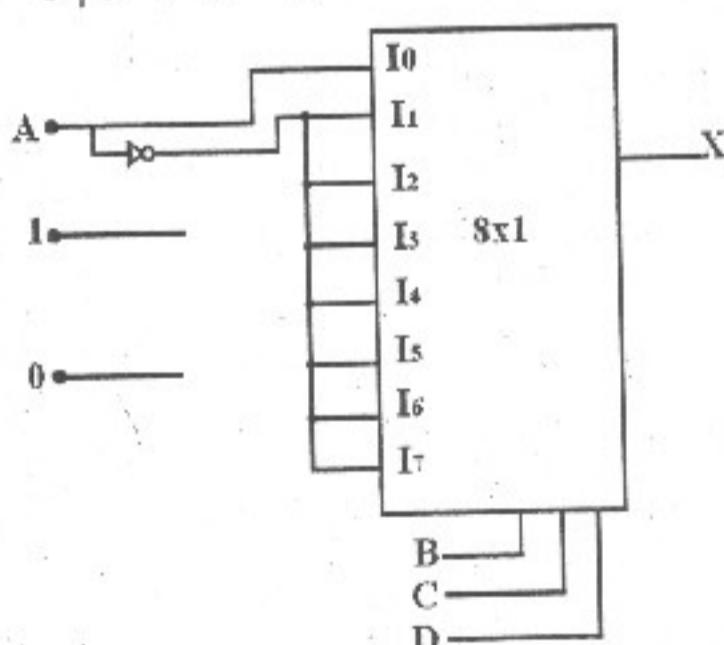
	I ₀	I ₁	I ₂	I ₃
x'	①	1	②	3
x	④	5	⑥	7
	1	0	1	0

برای Mux D , C , B رسم می کنیم.



۱۳ - مداری با چهار ورودی به کمک Mux طراحی کنید که خروجی مکمل ۲ ورودی باشد.

	A	B	C	D	x	y	w	z
0	0	0	0	0	0	0	0	0
1	0	0	0	1	1	1	1	1
2	0	0	1	0	1	1	1	0
3	0	0	1	1	1	1	0	1
4	0	1	0	0	1	1	0	0
5	0	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	0
7	0	1	1	1	1	0	0	1
8	1	0	0	0	1	0	0	0
9	1	0	0	1	0	1	1	1
A	1	0	1	0	0	1	1	0
B	1	0	1	1	0	1	0	1
C	1	1	0	0	0	1	0	0
D	1	1	0	1	0	0	1	0
E	1	1	1	0	0	0	1	0
F	1	1	1	1	0	0	0	1



برای y, z, w هم شکل رسم شود.

مدار منطقی

	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
A'	0	①	②	③	④	⑤	⑥	⑦
A	⑧	9	10	11	12	13	14	15
	A	A'						

	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
A'	0	①	②	3	4	⑤	⑥	7
A	8	⑨	⑩	⑪	12	⑬	⑭	15
	0	1	1	0	0	1	1	0

	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
A'	0	①	②	③	④	5	6	7
A	8	⑨	⑩	⑪	⑫	13	14	15
	0	1	1	1	1	0	0	0

	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
A'	0	①	2	③	4	⑤	6	⑦
A	8	⑨	10	⑪	12	⑬	14	⑮
	0	1	0	1	0	1	0	1



مجموعه تست‌ها:

۱ - مدار داخلی یک مالتی‌پلکسر 4×1 از چه گیت‌هایی تشکیل شده است؟

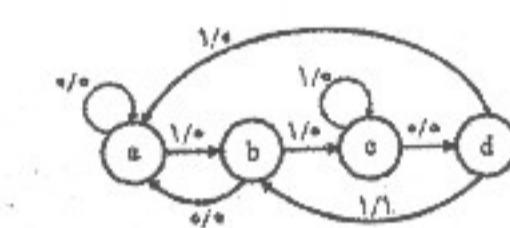
(۱) ۴ گیت AND و یک گیت OR و ۲ گیت NOT

(۲) ۴ گیت AND و یک گیت OR و ۴ گیت NOT

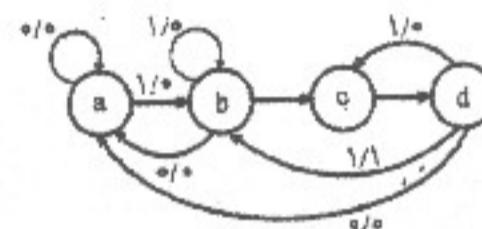
(۳) ۴ گیت AND و دو گیت OR و ۲ گیت NOT

(۴) ۴ گیت AND و دو گیت OR و ۴ گیت NOT

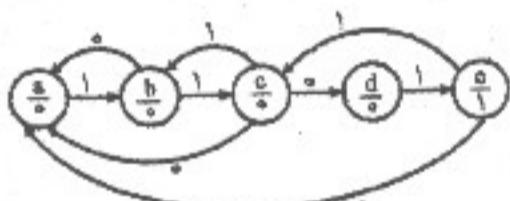
۲ - یک State Diagram به صورت Moore رسم کنید که ۱۱۰۱ را روى ورودی پیدامی کند. وقتی این Sequence یافت شد خروجی یک می‌شود.



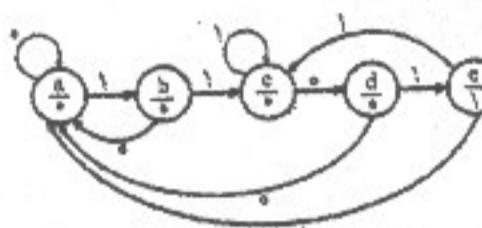
(۳)



(۱)



(۴)



(۲)

۳ - نمودار حالت مقابله را با استفاده از دو فلیپ‌فلاب D پیاده‌سازی کنیم. تابع فلیپ‌فلاب مربوط به یست بنا ارزش بیشتر را مولتی‌پلکسر MUX و تابع ورودی فلیپ‌فلاب دیگر را MUX₀ می‌سازد. کدام گزینه ورودی‌های مولتی‌پلکسرها را مشخص می‌کند؟

ورودی‌های select₀ و select₁ مربوط به مولتی‌پلکسر را به ترتیب Q₀ و Q₁ وصل می‌کنیم.

$$\text{MUX}_1 : I_0 = x \quad I_1 = \bar{x} \quad I_2 = \bar{x} \quad I_3 = \bar{\bar{x}} \quad (1)$$

$$\text{MUX}_0 : I_0 = \bar{x} \quad I_1 = x \quad I_2 = 1 \quad I_3 = \bar{x} \quad (2)$$

$$\text{MUX}_1 : I_0 = 0 \quad I_1 = 0 \quad I_2 = x \quad I_3 = \bar{x} \quad (3)$$

$$\text{MUX}_0 : I_0 = x \quad I_1 = \bar{x} \quad I_2 = 1 \quad I_3 = 1 \quad (4)$$

$$\text{MUX}_1 : L_0 = 0 \quad L_1 = x \quad L_2 = x \quad L_3 = \bar{x} \quad (5)$$

$$\text{MUX}_0 : L_0 = x \quad L_1 = \bar{x} \quad L_2 = 1 \quad L_3 = \bar{x} \quad (6)$$

$$\text{MUX}_1 : I_0 = x \quad I_1 = 0 \quad I_2 = x \quad I_3 = \bar{x} \quad (7)$$

$$\text{MUX}_0 : I_0 = 1 \quad I_1 = x \quad I_2 = 1 \quad I_3 = \bar{x} \quad (8)$$

مدار منطقی

۴- جدول Prime Implicant زیر برای تابع f بدست آمده است. ساده‌ترین فرم نایاب عبارت است از:

	2	4	6	8	9	10	12	13	15
PI ₁				X	X		X	X	
PI ₂	X		X						
PI ₃	X					X			
PI ₄		X	X						
PI ₅		X					X		
PI ₆				X		X			
PI ₇							X	X	

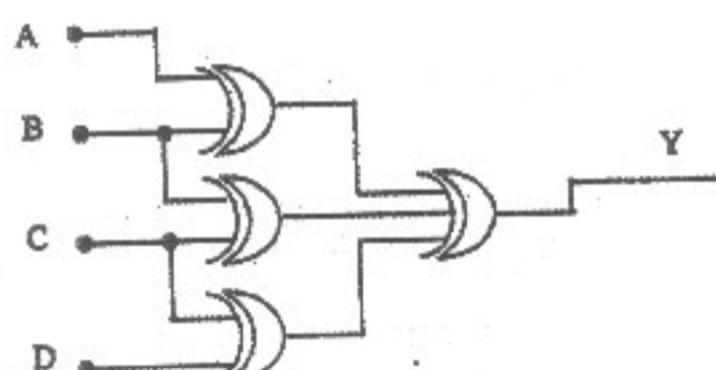
$$f = PI_4 + PI_1 + PI_3 + PI_7 \quad (1)$$

$$f = PI_2 + PI_3 + PI_5 + PI_7 \quad (2)$$

$$f = PI_1 + PI_3 + PI_4 + PI_6 \quad (3)$$

$$f = PI_3 + PI_7 + PI_1 + PI_5 \quad (4)$$

۵- کدام گزینه رابطه حاکم بر مدار شکل زیر را درست تعریف می‌کند؟



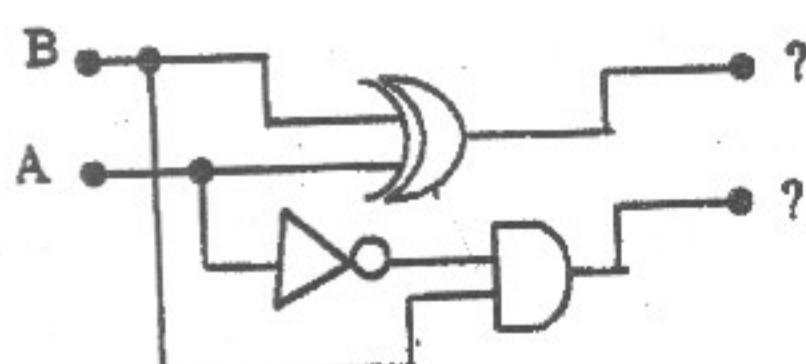
$$Y = (A \oplus B) + (C \oplus D) \quad (1)$$

$$Y = A \oplus B \oplus C \oplus D \quad (2)$$

$$Y = (A \oplus B) + (B \oplus C) + (C \oplus D) \quad (3)$$

$$Y = (A + B) \oplus (B + C) \oplus (C + D) \quad (4)$$

۶- مداری که در شکل زیر ملاحظه می‌کنید چه عملی را انجام می‌دهد؟



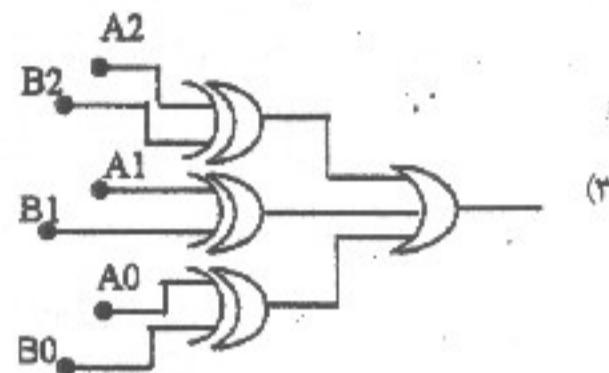
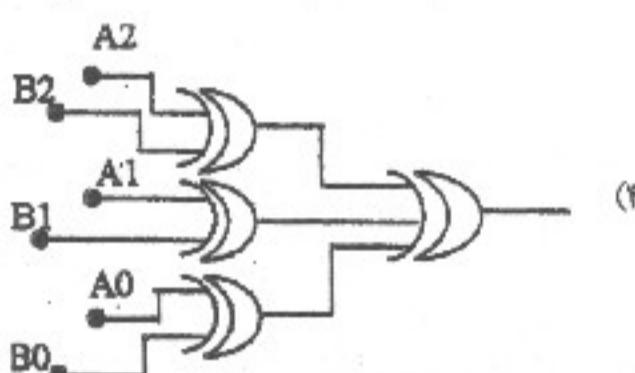
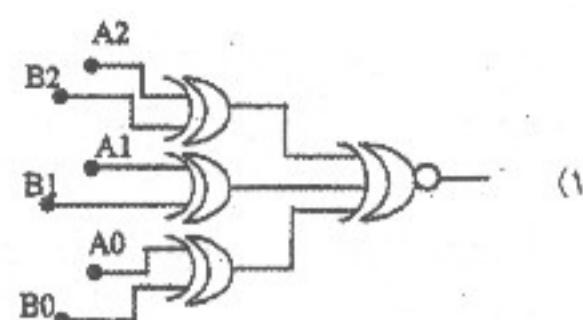
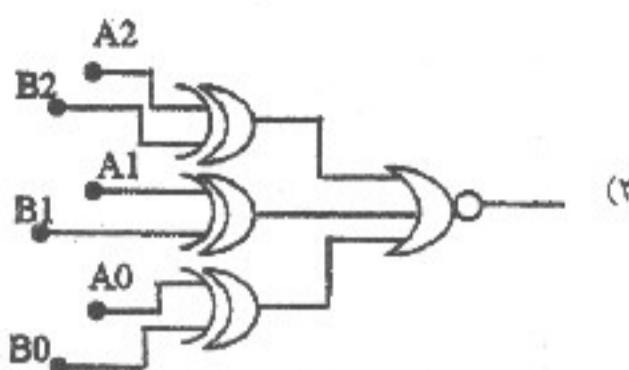
(۱) یک جمع کننده تک بیتی است با احتساب دو برقی (FA)

(۲) یک تفریق کننده تک بیتی است با احتساب دو برقی (FS)

(۳) یک تفریق کننده تک بیتی است بدون احتساب دو برقی (HS)

(۴) یک جمع کننده تک بیتی است بدون احتساب دو برقی (HA)

۷- کدامیک از مدارهای زیر یک مقایسه‌کننده ۳ بیتی را نشان می‌دهد که عدم تساوی A و B باعث ۰ شدن خروجی آن می‌گردد؟



۸- می‌خواهیم یک رمزگشای ۳ به ۸ یا (3 TO 8 Decoder) را بسازیم. مدار این رمزگشا دارای سه ورودی، ۸ خروجی و یک پایه کنترل است، و فقط یکی از خروجی‌ها تحت شرایط زیر در سطح ۰ فعال می‌شود (Active Low):

الف: چنانچه یک پایه فعال کننده مدار به نام E' در سطح ۰ فعال شود.

ب: سه ورودی مدار در سطوحی متناسب با یک عدد بین ۷ و ۰ فعال شده باشند.

تعداد دروازه‌های لازم برای این مدار کدامند؟

(۲) ۳ دروازه NOT و ۸ دروازه AND چهار ورودی

(۱) ۴ دروازه NOT و ۸ دروازه NAND چهار ورودی

(۴) ۳ دروازه NOT و ۸ دروازه AND سه ورودی

(۳) ۴ دروازه NOT و ۸ دروازه NAND سه ورودی

۹- در کدام گزینه تعداد دروازه‌های منطقی مورد نیاز برای یک مالتی‌پلکسر ۴ به ۱ (یک از چهار) درست معرفی شده است؟

(۱) دو دروازه NOT، چهار دروازه OR سه ورودی، یک دروازه AND چهار ورودی

(۲) دو دروازه NOT، چهار دروازه AND سه ورودی، یک دروازه OR چهار ورودی

(۳) یک دروازه NOT، چهار دروازه OR سه ورودی، یک دروازه AND چهار ورودی

(۴) یک دروازه NOT، چهار دروازه AND سه ورودی، یک دروازه OR چهار ورودی

۱۰- برای فرستادن ۵۵ خط اطلاعات به صورت Multiplexer نیاز به چند خط کنترل (Select input) داریم.

۴ (۴)

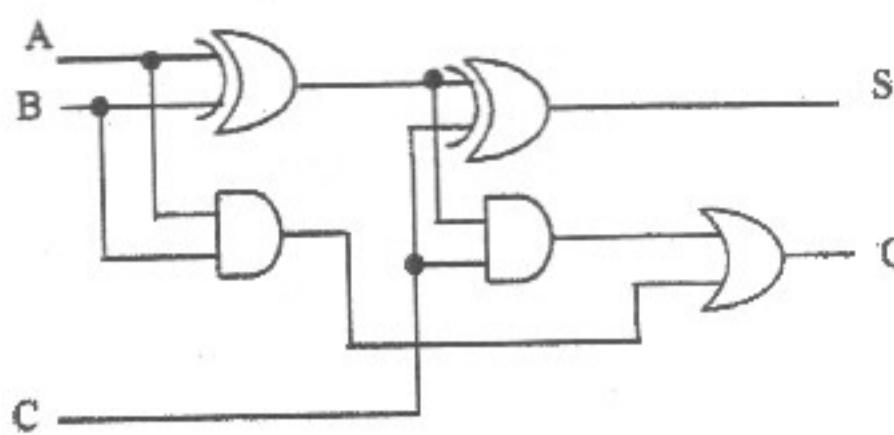
8 (۳)

5 (۲)

6 (۱)

مدار منطقی

۱۱ - شکل زیر چیست؟



- (۱) جمع کننده ناقص
- (۲) تفریق کننده کامل
- (۳) تفریق کننده ناقص
- (۴) جمع کننده کامل

۱۲ - از دو دیکودر 8×3 با پایه Enable کدام دیکودر را می‌توان ساخت؟

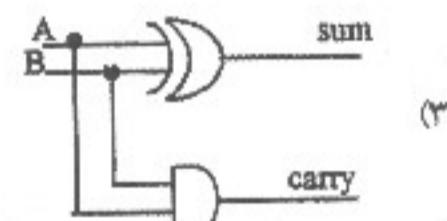
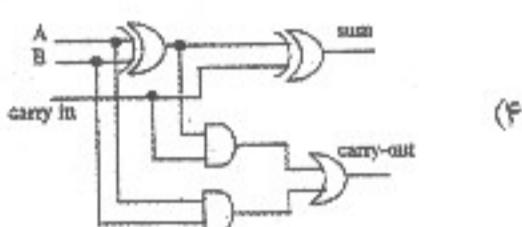
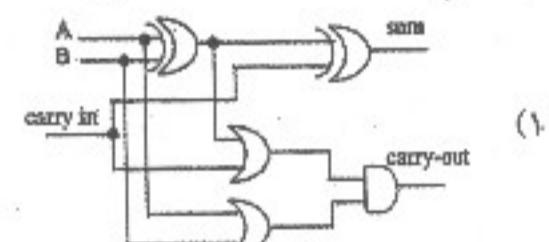
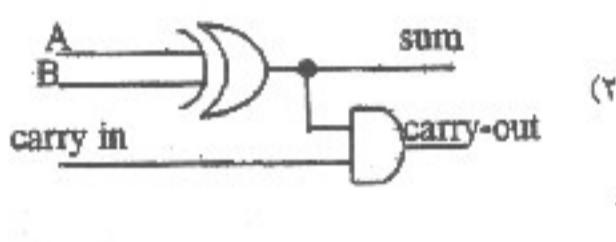
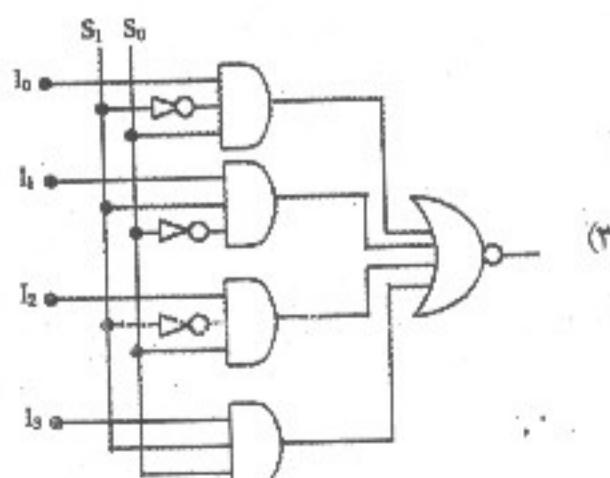
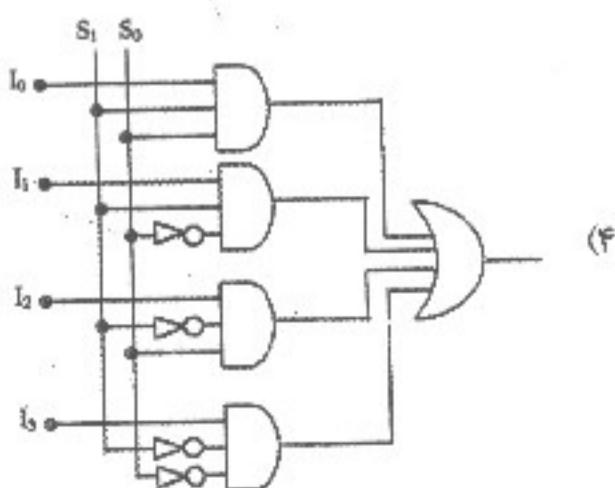
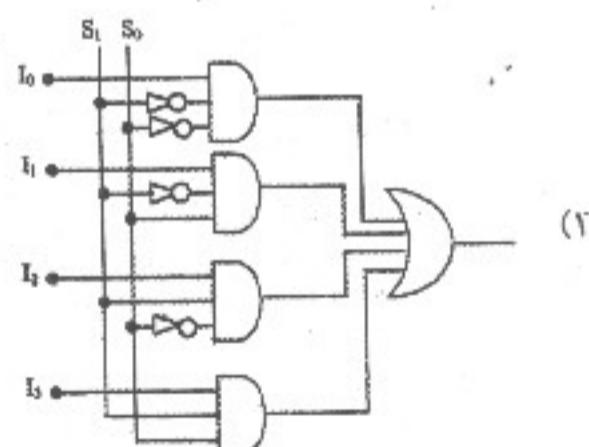
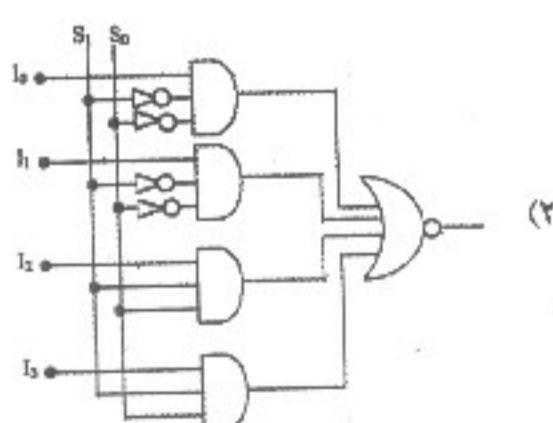
۶ × 16 (۴)

۴ × 8 (۳)

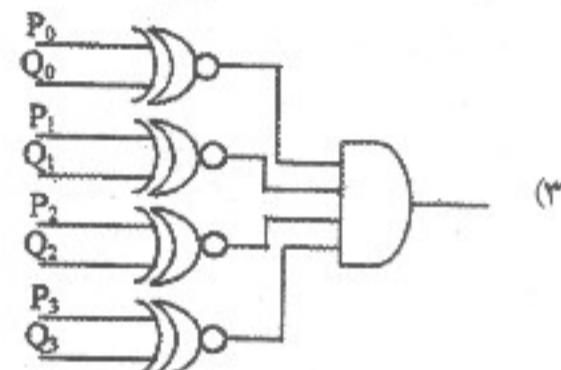
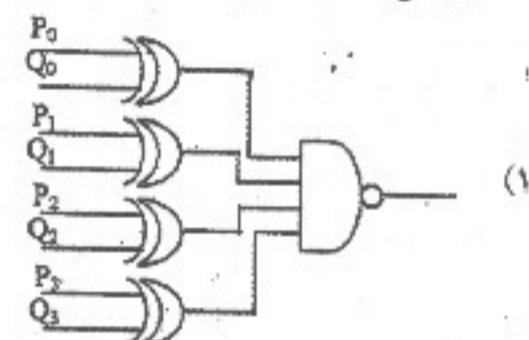
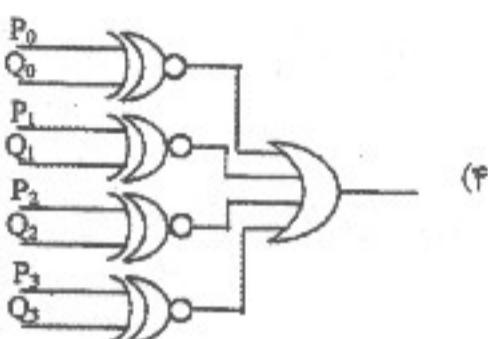
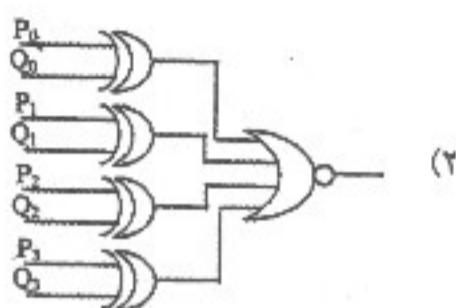
۴ × 16 (۲)

6 × 8 (۱)

۱۳ - کدام گزینه کاملترین واحد جمع کننده دوینی (Full Adder) قبل توسعه به تعداد بیت بالاتر را نشان می‌دهد؟

۱۴ - کدام مدار منطقی می‌تواند بعنوان یک گزینشگر چهار بیتی (4-bit Multiplexer) بکار رود. لازم است متناسب با عدد مبنای دو در پایه‌های گزینشگر S_1, S_0 ورودی متناظر با آن به خروجی مربوط شود.

۱۵ - برای مقایسه دو عدد چهاربیتی P و Q و تعیین تساوی و یا عدم تساوی آنها کدام مدار مناسب است. لازم است در صورت تساوی P و Q خروجی مدار ۱ شود.



۱۶ - با استفاده از مدارهای مجتمع یا مقیاس متوسط چگونه می‌توان یک $\frac{1}{23}$ decoder ایجاد نمود؟

(۲) یک $\frac{1}{2}$ dec و شانزده $\frac{1}{8}$ decoder

(۴) یک $\frac{1}{4}$ dec و هشت $\frac{1}{4}$ dec

(۱) یک $\frac{1}{4}$ dec و هشت $\frac{1}{8}$ decoder

(۳) یک $\frac{1}{8}$ dec و چهار $\frac{1}{4}$ dec

۱۷ - با استفاده از مدارهای MUX 4 به ۱ و گیت‌های منطقی لازم یک مدار مقایسه گر 2 بیتی می‌خواهیم طراحی کیم. (مقایسه دو عدد ۲ بیتی) با کدام عناصر زیر می‌توان عمل را انجام داد، در صورتی که اگر $A \geq B$ باشد خروجی مدار ۱ و اگر $A < B$ باشد خروجی صفر می‌باشد؟

(فرض کنید همه ورودی‌ها و خروجی MUX از نوع active high هستند)

(۲) دو مدار MUX 4 به ۱ و یک NOR

(۱) یک مدار MUX و یک NOT

(۴) دو مدار MUX 4 به ۱ و یک NAND

(۳) یک NOT، یک NAND و یک MUX 4 به ۱

۱۸ - در یک سیستم کنترلی که دارای چهار ورودی و چهار خروجی و دو ورودی کنترل می‌باشد و طبق جدول زیر عمل می‌نماید توابع خروجی E و F کدام است؟

I_1	I_2	E	F	G	-
0	0	1	1	1	1
0	1	A	B	C	D
1	0	\bar{A}	\bar{B}	\bar{C}	\bar{D}
1	1	0	0	0	0

$$E = \bar{I}_1 \bar{I}_2 A + I_1 \bar{I}_2 \bar{A} + I_1 I_2 \quad (۱)$$

$$F = \bar{I}_1 \bar{I}_2 B + I_1 \bar{I}_2 \bar{B} + I_1 I_2 \quad (۲)$$

$$E = \bar{I}_1 \bar{I}_2 + \bar{I}_1 I_2 A + I_1 \bar{I}_2 \bar{A} \quad (۱)$$

$$F = \bar{I}_1 \bar{I}_2 + \bar{I}_1 I_2 B + I_1 \bar{I}_2 \bar{B} \quad (۲)$$

$$E = I_1 I_2 + \bar{I}_1 I_2 + A I_1 \bar{I}_2 + \bar{A} \bar{I}_1 \bar{I}_2 \quad (۳)$$

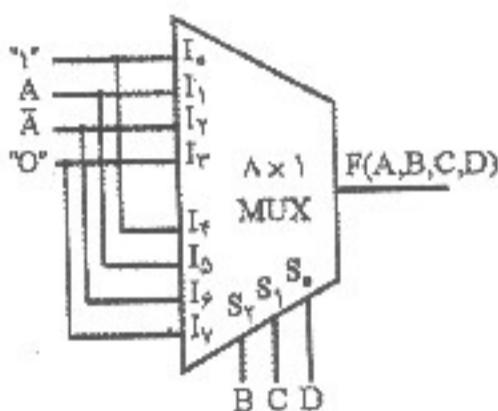
$$F = I_1 I_2 + \bar{I}_1 I_2 B I_1 \bar{I}_2 + \bar{B} \bar{I}_1 \bar{I}_2 \quad (۴)$$

$$E = A + \bar{I}_1 \bar{I}_2 + I_1 I_2 \quad (۳)$$

$$F = B + \bar{I}_1 \bar{I}_2 + I_1 I_2 \quad (۴)$$

مدار منطقی

۱۹ - مداری دارای ۴ ورودی A,B,C,D و خروجی F با استفاده از یک مالتی پلکسر 1×8 طبق شکل زیر طراحی گردیده است، ترمومات این مدار که به ازاء آن خروجی high می‌باشد را بنویسید؟



$$f(a, b, c, d) = \sum m(1, 3, 5, 7, 9, 11, 13) \quad (2)$$

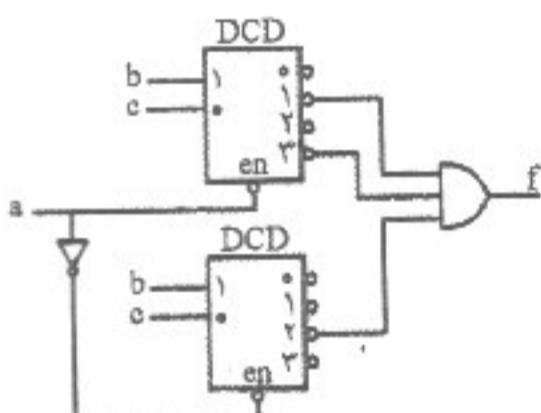
$$f(a, b, c, d) = \sum m(0, 2, 6, 8, 9, 12) \quad (4)$$

0	1	2	3	4	5	6	7
\bar{a}	0	1	2	3	4	5	6
a	8	4	10	11	12	13	14

$$f(a, b, c, d) = \sum m(0, 4, 8, 12, 14) \quad (1)$$

$$f(a, b, c, d) = \sum m(0, 2, 4, 6, 8, 9, 12, 13) \quad (3)$$

۲۰ - شکل مقابل از دو دیکوادر دو ورودی تشکیل شده که دارای active-low enable می‌باشند. خروجی‌های دیکوادرها نیز low-active (فعال-پایین) هستند.تابع خروجی صحیح کدام است؟



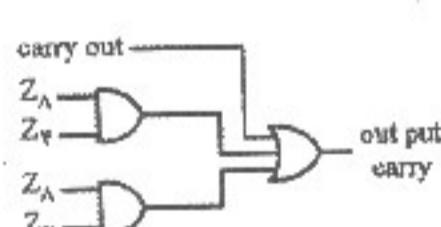
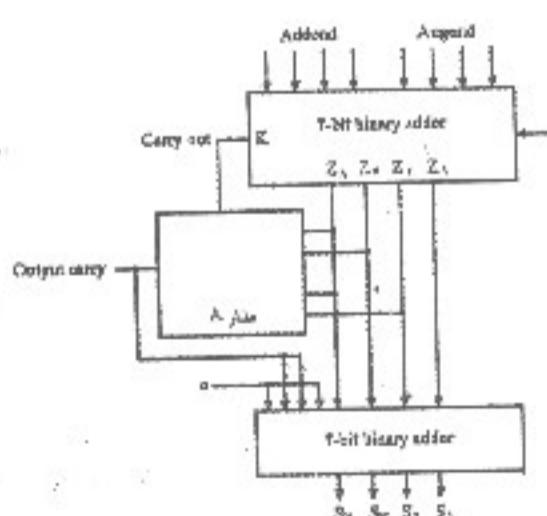
$$f(a, b, c) = \sum m(2, 5, 7) \quad (2)$$

$$f(a, b, c) = \sum m(0, 2, 4, 5, 7) \quad (4)$$

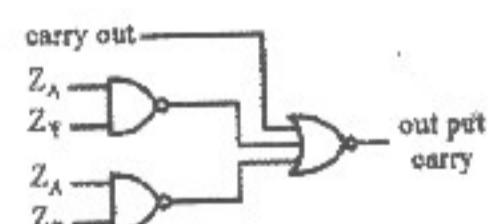
$$f(a, b, c) = \sum m(1, 3, 6) \quad (1)$$

$$f(a, b, c) = \sum m(0, 1, 3, 4, 6) \quad (3)$$

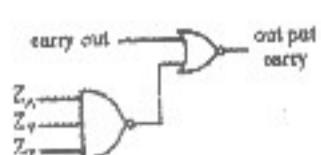
۲۱ - توسط یک مدار (A) و دو تمام جمع کننده می‌توان یک جمع کننده BCD طراحی نمود (مطابق شکل مقابل). مدار A را طراحی نمایید. (کدام گزینه را می‌توان به جای مدار A قرار داد.)



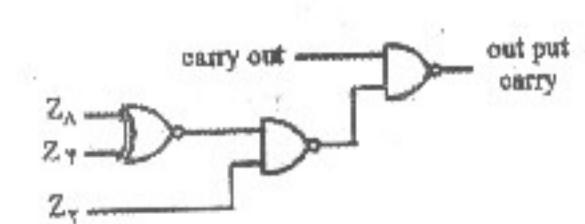
(2)



(1)

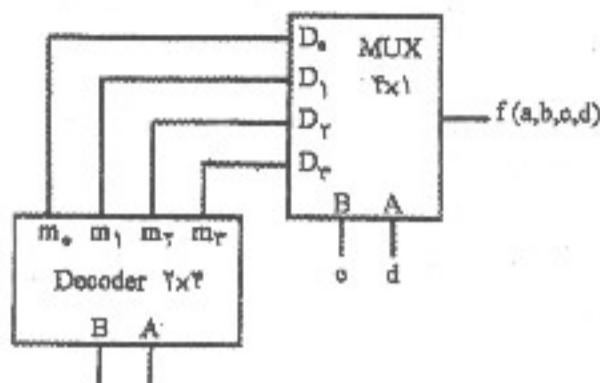


(4)



(3)

۲۲- مدار مقابل را در نظر بگیرید. B ورودی پرازش تر دیکدر (active high) و مالتی پلکسر است کدام گزینه ییانگرتابع (d) است؟



$$F=0 \quad (1)$$

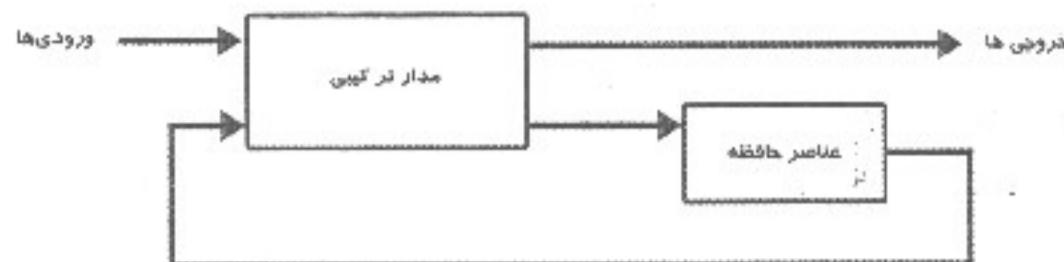
$$F=\sum m(0,5,10,15) \quad (2)$$

$$\bar{F} = \bar{a}\bar{b} + ab + \bar{c}\bar{d} + cd \quad (3)$$

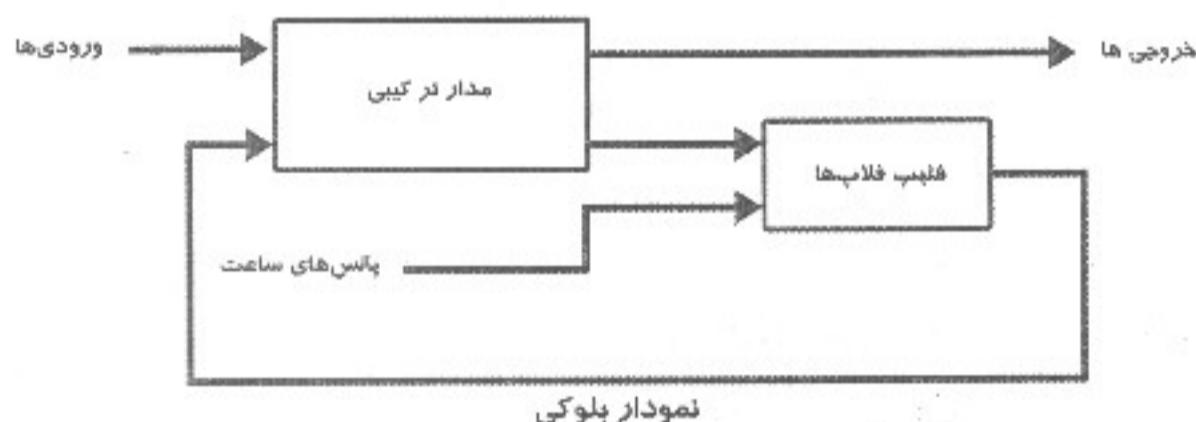
$$F = 1 \quad (4)$$

مدارهای ترتیبی

مدارهای ترتیبی در اصل همان مدارهای ترکیبی هستند که به آنها حافظه اضافه شده این حافظه‌ها می‌توانند از قلیپ فلاب‌ها تشکیل شده باشند.



نمودار بلوکی یک مدار ترتیبی



نمودار بلوکی



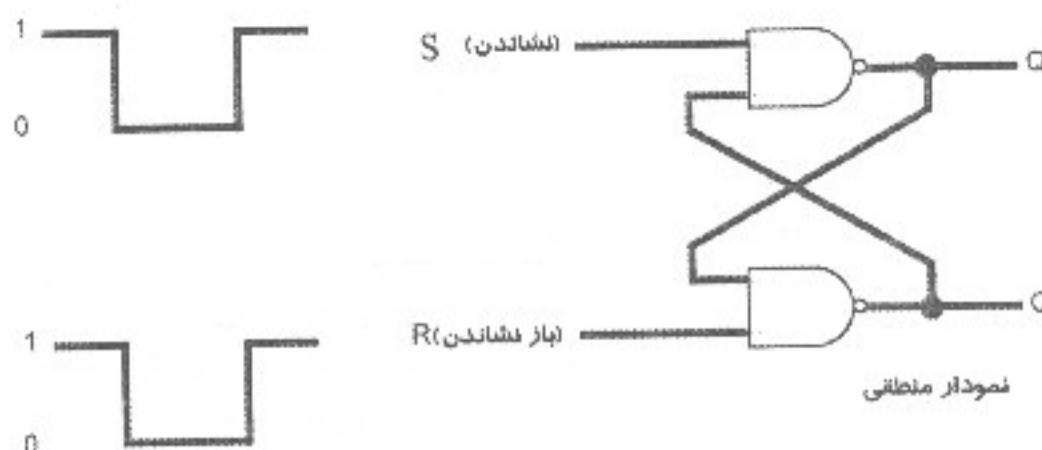
مدار ترتیبی ساعت‌دار همزمان

Flip Flop‌ها

یکی از عناصر حافظه هستند که برای نگهداری اطلاعات از آنها استفاده می‌شوند. فلیپ فلاب‌ها در اصل مهمترین عناصر موجود در مدارهای ترتیبی می‌باشند.

S	R	Q	Q'
1	0	0	1
1	1	0	1
0	1	1	0
1	1	1	0
0	0	1	1

هدایت منطقی



لچ SR با گیت های NAND

در ساختار فلپ فلپ SR که در آن گیت های NAND شکل بالا به کار رفته باید توجه داشت که ورودی های آن به صورت Active low می کنند یعنی جدول درستی آن به شکل زیر تغییر می کند.

R	Q	Q'
0	0	1
1	0	1
1	1	0
1	1	0
0	1	1

همان طور که ملاحظه می شود $(0, 0)$ برای ورودی های (S, R) هیچ تغییری در خروجی مرحله قبل ایجاد نمی کند و همچنین $(1, 1)$ برای ورودی های (S, R) باعث به وجود آمدن مشکل در آن می شود زیرا خروجی های (Q', Q) که باید همیشه با هم متفاوت باشند هر دو برابر یک می شوند.

S	R	Q	Q'	$Q' = \text{Not}(Q)$
0	1	0	1	
0	0	0	1	
1	0	1	0	
0	0	1	0	
1	1	-	-	

SR-Flip Flop

خروجی در لحظه t خروجی در لحظه $t+1$

نشاندن با set کردن S
بازنشاندن با Reset کردن R

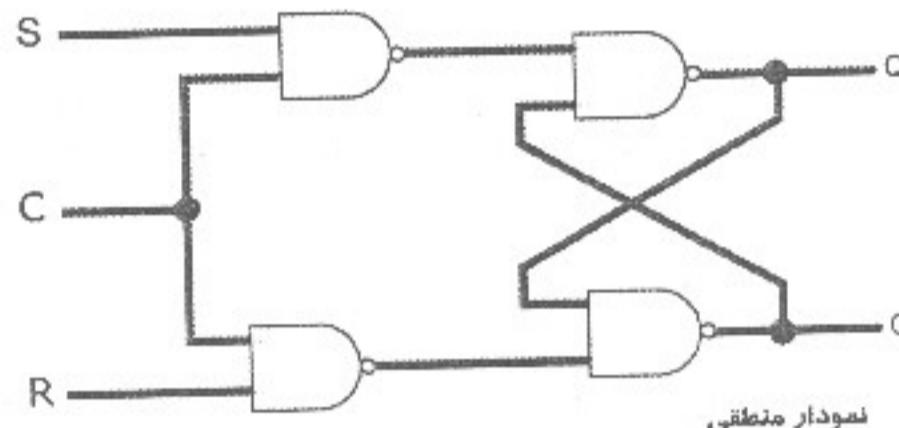
S	R	Q_t	$Q_{(t+1)}$	
0	0	0	0	بدون تغییر $\rightarrow Q_t$
0	0	1	1	
0	1	0	0	$\rightarrow 0$ Reset
0	1	1	0	
1	0	0	1	$\rightarrow 1$ Set
1	0	1	1	
1	1	0	-	نامعین -
1	1	1	-	

مدار منطقی

زمانی که SR هر دو یک هستند حالت نامعین یا حالت ناخواسته است. که در این نوع قلیپ فلاپ به وجود می‌آید.

SR	Q	00	01	11	10
0	Q			x	1
1	Q	1		x	1

$$Q_{t+1} = S + QR' \Rightarrow \begin{cases} S, R = 0 \\ S, R \neq 1 \end{cases}$$



در شکل بالا وقتی CLOCK صفر است قلیپ فلاپ عمل نمی‌کند و هر چه از قبل در خروجی مانده نشان می‌دهد ولی اگر یک باشد به شکل یک R . S قلیپ فلاپ عمل می‌کند و خروجی‌ها مطابق جدول صفحه قبل خواهد بود.

Jk - Flipflop

برای ازین بردن حالت ناخواسته در SR - FF - FF قلیپ فلاپ جدیدی به وجود آورده که نام Jk-FF به کمک خطوط فیدبک که از خروجی Q', Q گرفته شده و به همراه ورودی‌ها به NAND های اولیه وارد شده، مشکل مقدار (1, 1) را حل کردند.

خروجی در لحظه t Q_t خروجی در لحظه $t+1$ Q_{t+1}

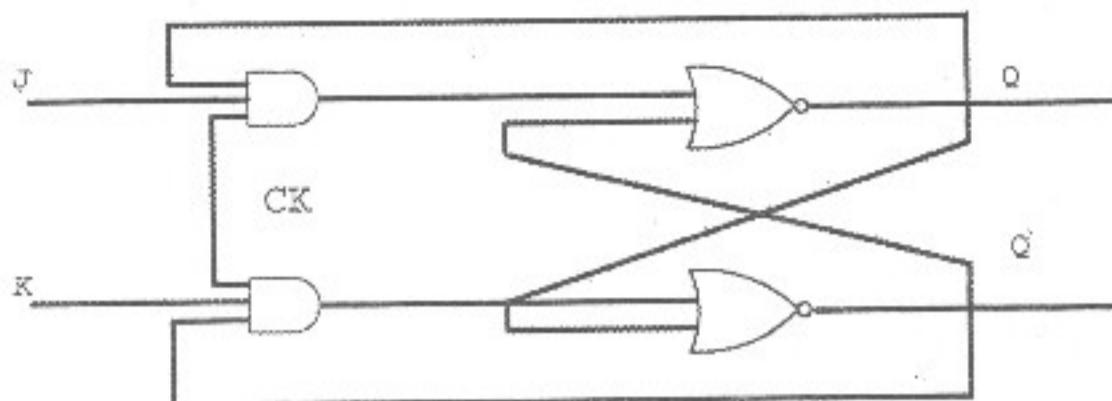
نشاندن یا Set کردن را

J	K	Q_t	Q_{t+1}	خروجی
0	0	0	0	بدون تغییر
0	0	1	1	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	1	
1	1	1	0	متمن

JK	00	01	11	10
Q_t	0		1	1
0	1			
1				

$$\Rightarrow Q_{t+1} = jQ'_t + k'Q_t$$

مدار منطقی



D-Flip Flop

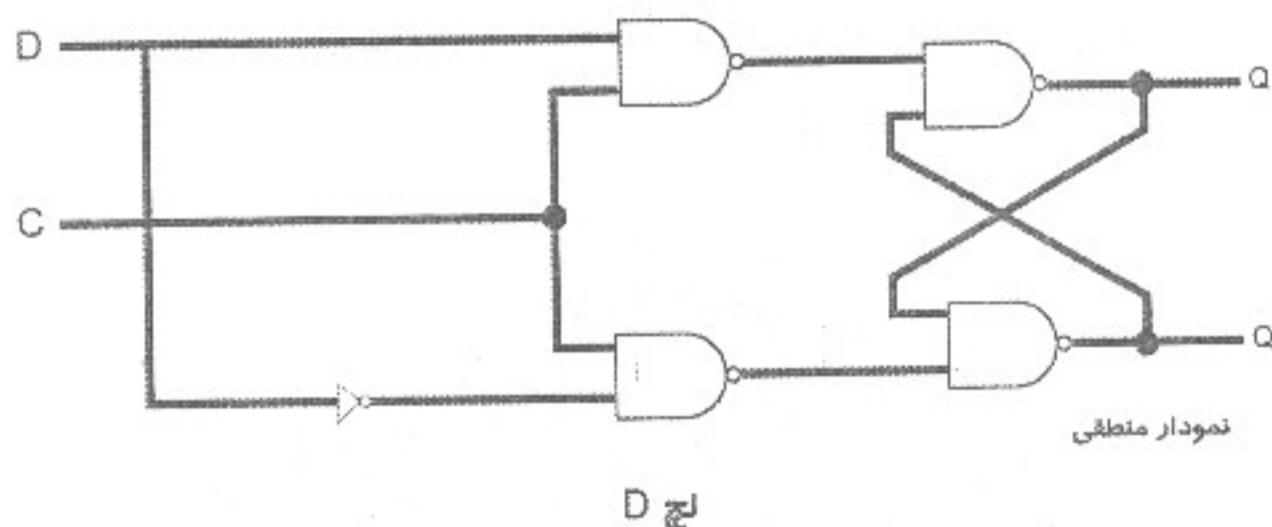
حالتی خاص از SR-FF است.

به آن Data F.F نیز گفته می‌شود. که کارش نگهداری اطلاعات در طول یک CLK می‌باشد. در واقع همان داده‌ای که به ورودی آن اعمال می‌شود را به خروجی منتقال می‌دهد.

$$\begin{cases} D = 0 & Q_{t+1} = 0 \\ D = 1 & Q_{t+1} = 1 \end{cases} \Rightarrow Q_{t+1} = D$$

D	Q	Q_{t+1}
0	0	0
0	1	0
1	0	1
1	1	1

همانطور که ملاحظه می‌شود خروجی مرحله قبل هیچ تاثیری در عملکرد D-F.F ندارد و تنها زمانی که CLK فعال شود هر مقداری که روی ورودی قرار دارد به خروجی منتقل می‌شود.



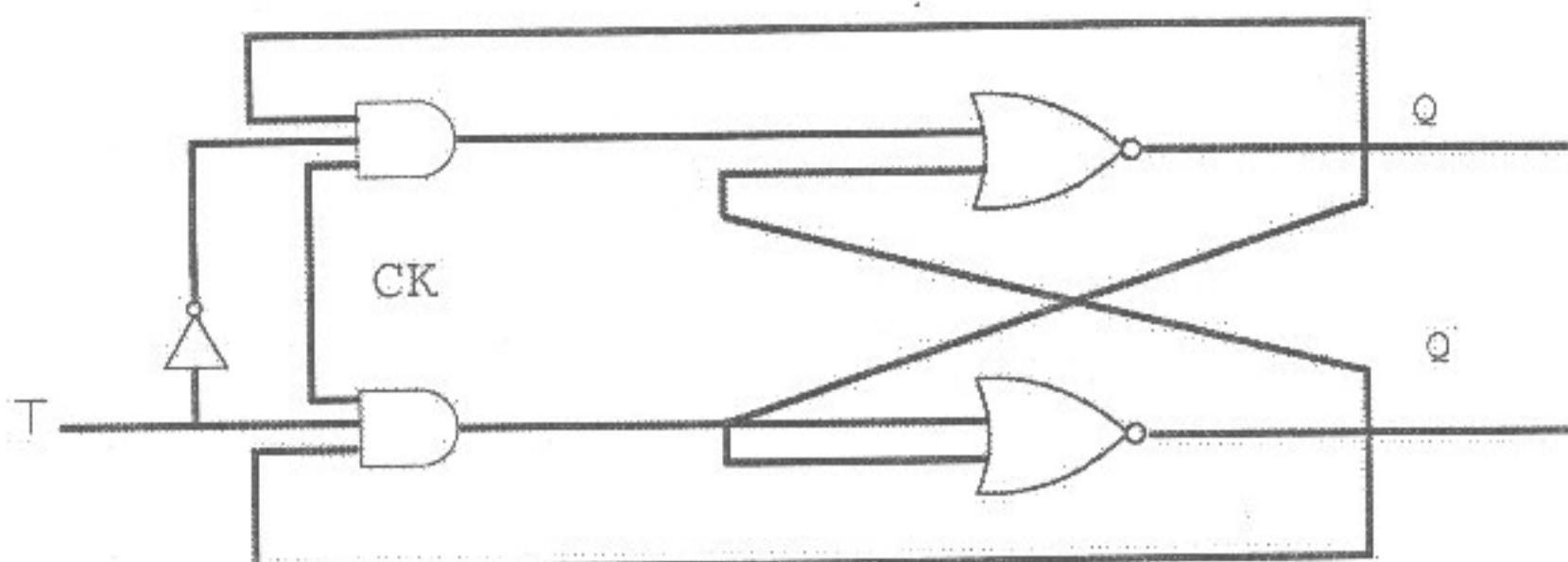
T-Flip Flop

T	$Q_{(t)}$	$Q_{(t-1)}$
0	0	0
1	1	0

وقتی $T=0$ باشد خروجی مرحله بعدی همان خروجی مرحله قبل است.

$$\Rightarrow Q_{t+1} = Q_t \quad (1)$$

مدار منطقی

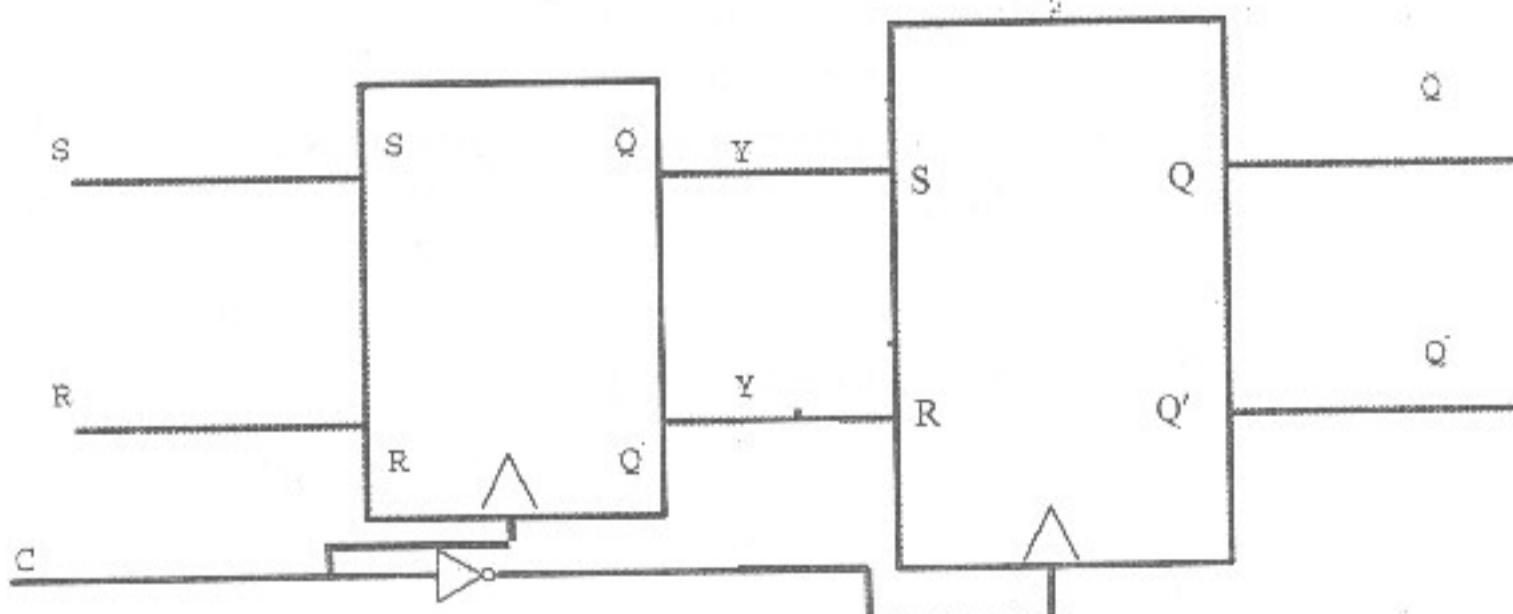


وقتی $T=1$ باشد خروجی مرحله بعد متمم خروجی مرحله قبل است

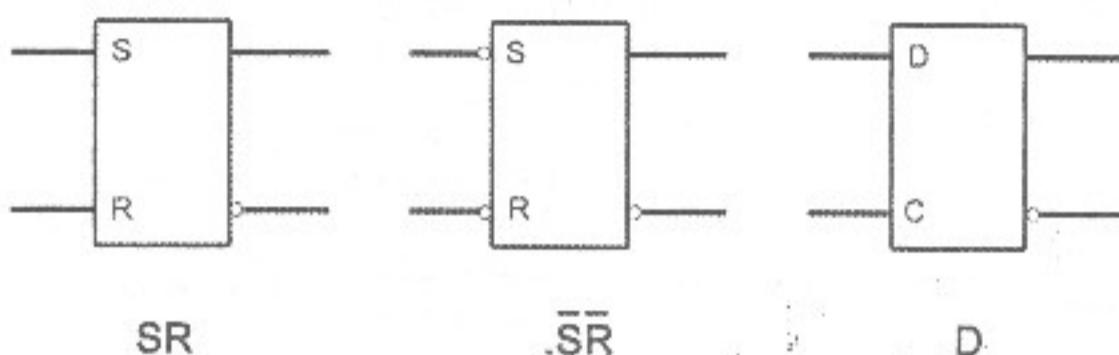
$$T=1 \Rightarrow Q_{t+1} = Q'_t \quad (\text{II})$$

تابع متتابع Master Slave

از دو فلیپ فلاپ پشت سر هم که هر دو از یک نوع هستند (D,T,Jk,RS) تشکیل شده است. خروجی فلیپ فلاپ اول وارد فلیپ فلاپ دوم می شود و در ورودی آن می ماند تا هنگامی که کلاک باعث فعال شدن فلیپ فلاپ دوم شود و خروجی آن فعال شود. و در هر زمان تنها یک فلیپ فلاپ کار می کند به این معنی که کلاک وارد شده در فلیپ فلاپ دوم، متمم شده کلاک وارد شده به فلیپ فلاپ اول است. یعنی در زمانی که مقدار CLK یک باشد تنها فلیپ فلاپ اول کار می کند و در زمانی که صفر است اولین فلیپ فلاپ غیرفعال می شود و فلیپ فلاپ کار نمی کند.



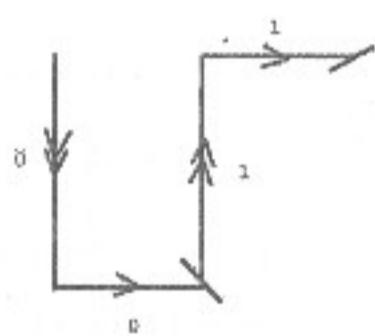
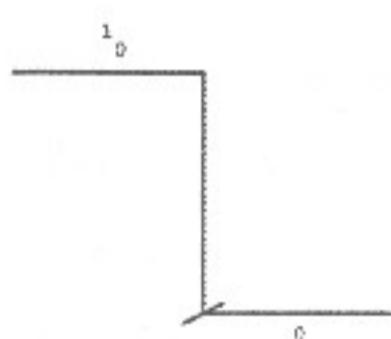
سمبل‌های اگرافیکی



سمبل‌های گرافیکی لج

انواع کلاک پالس:

کلاک پالس یا حساس به لبه است یا حساس به سطح



حساس به لبه:

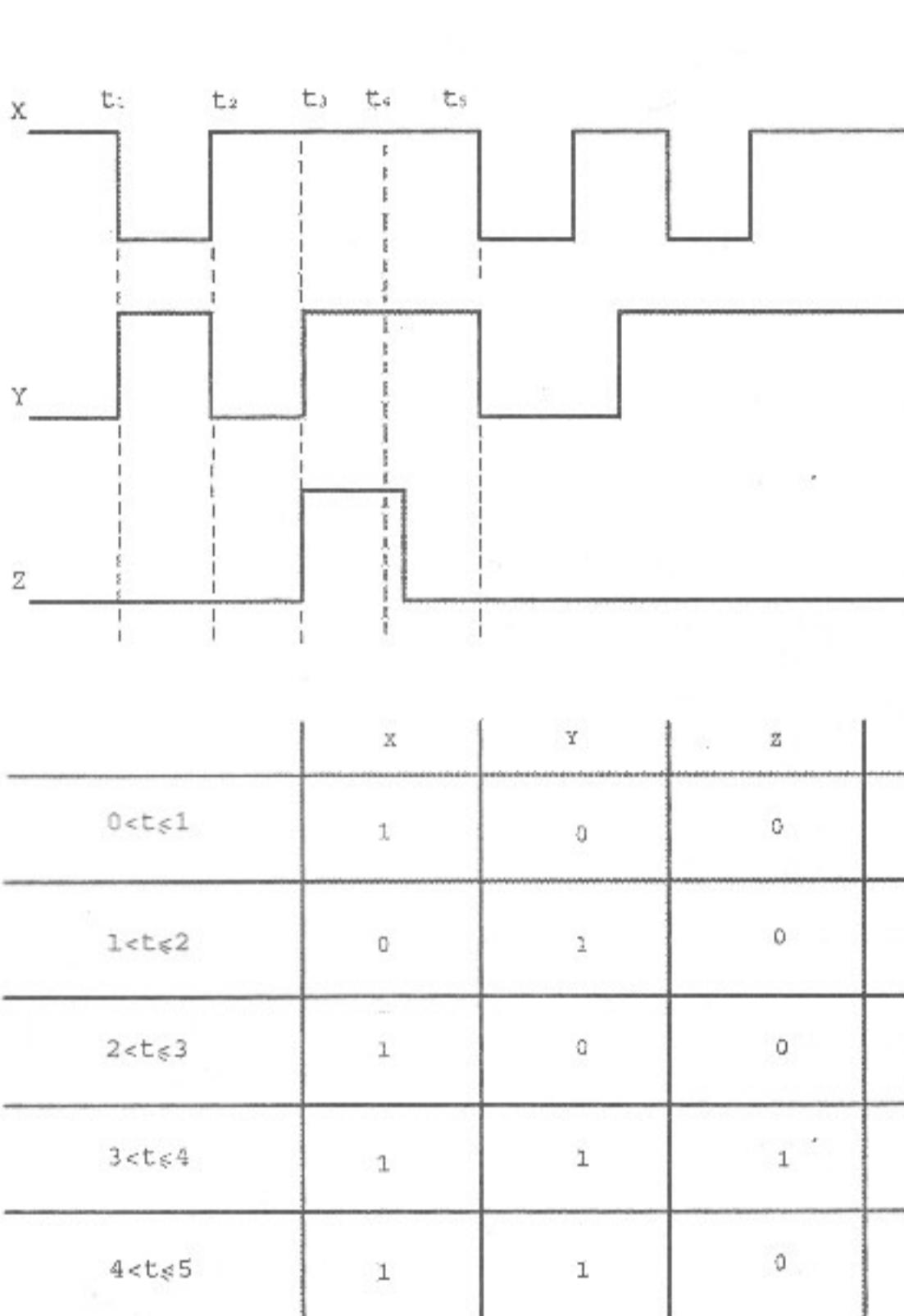
در کلاک پالس حساس به لبه زمانی که حرکت در لبه به سمت عدد صفر تولید می‌شود تا زمانی که تغییر سمت صورت گیرد به محض تغییر جهت حرکت به سمت بالا عدد یک تولید می‌شود

حساس به سطح:

در این نوع کلاک با شروع سطح مقادیر صفر و یک تولید می‌شود به این مفهوم که وقتی که سطح بالا شروع می‌شود مقدار کلاک ۱ و وقتی که سطح پایین شروع می‌شود مقدار کلاک صفر می‌شود این مقادیر تا شروع سطح بعدی به همین شکل می‌مانند.

ورودی‌های مدار قریبی:

در لحظه خاصی مثل $t=1$ نمی‌توان تشخیص داد مقدار x, y, z چقدر است برای مشخص کردن مقدار x, y, z باید از بازه‌های زمانی استفاده کرد.



$$\begin{cases} x = 1 \\ y = 0 \\ z = 0 \end{cases} \quad \text{در بازه } 0 < t \leq 1$$

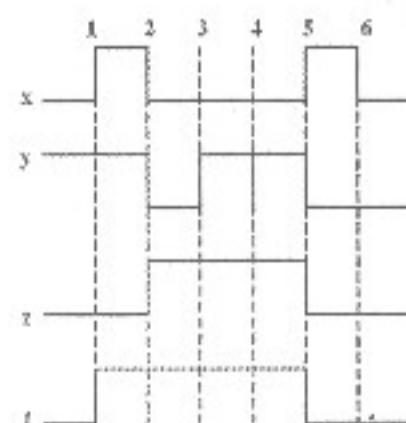
$$\begin{cases} x = 0 \\ y = 1 \\ z = 0 \end{cases} \quad \text{در بازه } 1 < t \leq 2$$

$$\begin{cases} x = 1 \\ y = 0 \\ z = 0 \end{cases} \quad \text{در بازه } 2 < t \leq 3$$

$$\begin{cases} x = 1 \\ y = 1 \\ z = 1 \end{cases} \quad \text{در بازه } 3 < t \leq 4$$

$$\begin{cases} x = 1 \\ y = 1 \\ z = 0 \end{cases} \quad \text{در بازه } 4 < t \leq 5$$

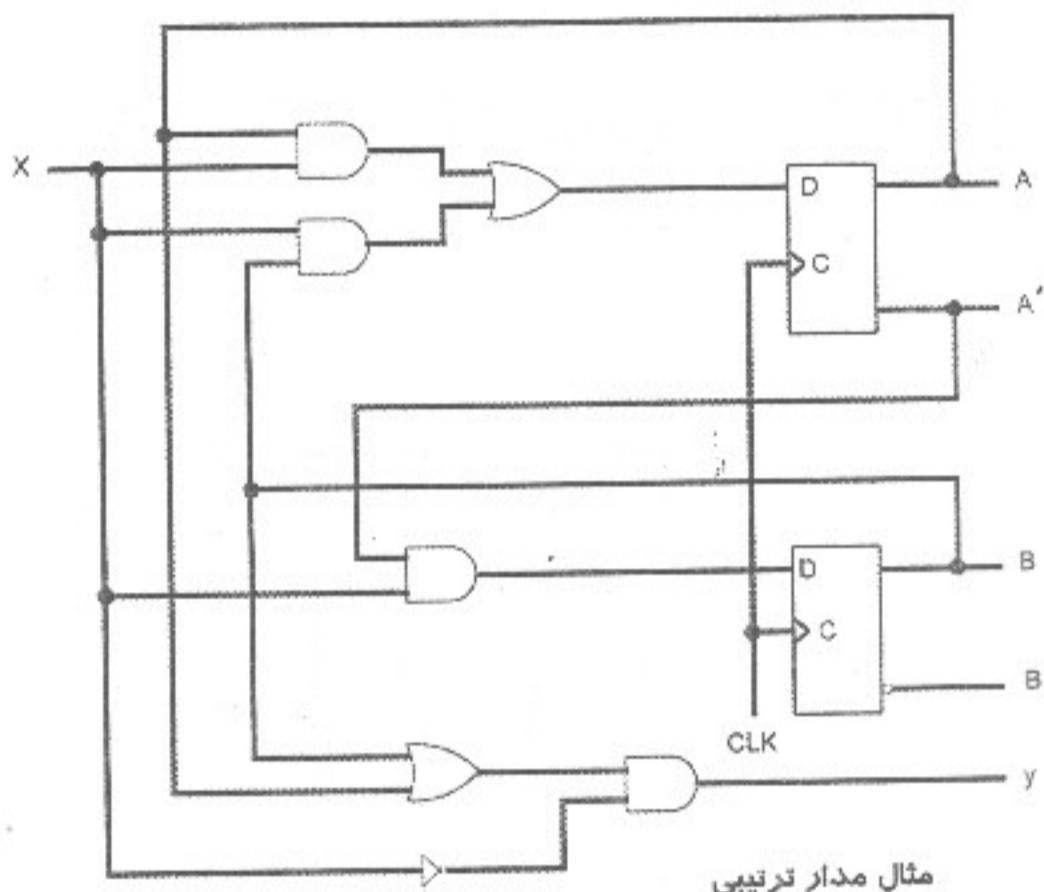
مثال: شکل تابع $F = A + BC'$ را رسم کنید. (با توجه به اینکه کلاک ورودی A و B و C)



تحلیل مدارهای ترتیبی ساعت دار

مرحله ۱- به دست آوردن معادلات خروجی بر حسب ورودی و خروجی های مرحله قبل مثال:

با توجه به شکل مدلار زیر تحلیل و طراحی را انجام دهید.



هر ورودی که بدھیم همان را در خروجی می‌گیریم زیرا هر دو فلیپ فلاپ از نوع D-FF هستند.

$$y = x' \cdot (A + B)$$

$$A_{(1,1)} = (A.x) + (B.x)$$

$$\mathbf{B}_{(t+1)} = \mathbf{A}' \cdot \mathbf{x}$$

مرحله ۲- بدست آوردن جدول

حالت فعلی			حالة بعدى		
A _(t)	B _(t)	X	A _(t+1)	B _(t+1)	Y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

y را بر اساس معادلات خروجی که در مرحله قبل بدست آورده‌یم محاسبه می‌کنیم.

مرحله ۳—دیاگرام حالات

- ۱- تعداد فلیپ فلابها را مشخص می‌کنیم تا تعداد حالات مشخص شود. (تعداد حالات در واقع برابر است با 2^2 به توان تعداد فلیپ فلابها)
- حالات

A	B
0	0
0	1
1	0
1	1

تعداد فلیپ فلابها :

$$2^n = \text{تعداد حالات}$$

- ۲- تعداد ورودی‌ها را مشخص می‌کنیم و حالات ممکن را برایش می‌نویسیم. برای یک ورودی (x) دو حالت داریم

$\left. \begin{array}{l} x = 0 \\ x = 1 \end{array} \right\}$ و برای هر کدام مقدار AB را به دست می‌آوریم.
 اگر دو ورودی داشتیم (x,y) چهار حالت دارد

$$\left. \begin{array}{l} x = 0 \\ x = 1 \end{array} \right\}$$

$$\left. \begin{array}{l} y = 0 \\ y = 1 \end{array} \right\}$$

به این مفهوم که اگر m خط ورودی داشتیم 2^m حالت مختلف برای ورودی‌ها به وجود می‌آید.

در مثال قبل یک ورودی داریم

حالت فعلی		حالت بعدی		خروجی	
		$x = 0$	$x = 1$	$y = 0$	$y = 1$
A	B	A	B	y	Y
0	0	0	0	1	0
0	1	0	0	1	0
1	0	0	0	1	0
1	1	0	0	1	0

با توجه به جدول حالات که در مرحله 2 بدست آوردهیم:

وقتی A,B حالت فعلی $(0,0)$ باشد و $x = 0 \Leftarrow AB \Leftarrow x = 0$ حالت بعدی $(0,0)$ هستند و $y=0$

وقتی A,B حالت فعلی $(0,0)$ باشد و $x = 1 \Leftarrow AB \Leftarrow x = 1$ حالت بعدی $(0,1)$ هستند و $y=0$

وقتی A,B حالت فعلی $(0,1)$ باشد و $x = 0 \Leftarrow AB \Leftarrow x = 0$ حالت بعدی $(0,0)$ هستند و $y=1$

وقتی A,B حالت فعلی $(0,1)$ باشد و $x = 1 \Leftarrow AB \Leftarrow x = 1$ حالت بعدی $(1,1)$ هستند و $y=0$

وقتی A,B حالت فعلی $(1,0)$ باشد و $x = 0 \Leftarrow AB \Leftarrow x = 0$ حالت بعدی $(0,0)$ هستند و $y=1$

وقتی A,B حالت فعلی $(1,0)$ باشد و $x = 1 \Leftarrow AB \Leftarrow x = 1$ حالت بعدی $(1,0)$ هستند و $y=0$

وقتی A,B حالت فعلی $(1,1)$ باشد و $x = 0 \Leftarrow AB \Leftarrow x = 0$ حالت بعدی $(0,0)$ هستند و $y=1$

وقتی A,B حالت فعلی $(1,1)$ باشد و $x = 1 \Leftarrow AB \Leftarrow x = 1$ حالت بعدی $(1,0)$ هستند و $y=0$

حال اگر بر فرض بخواهیم خروجی (y) و حالات بعدی (AB) را به کمک گپت‌ها پیاده‌سازی کنیم با توجه به جدول و با توجه به مقادیر یک به دست آمده در ستون خروجی‌ها و حالات بعدی به مانند طراحی مدارهای ترکیبی عمل می‌کنیم.

با توجه به این که مقدار خروجی z در زمانی که $x=1$ است صفر می‌باشد می‌توان دریافت که x به مانند یک کنترل برای z است. به این معنا که می‌توان لارا از خروجی g گرفت که یک ورودی آن متشتم شده x می‌باشد و با توجه به ستون z در زمان $x=0$ به مانند خروجی یک OR می‌باشد که A, B ورودی‌های آن هستند.

۳-۵- دیاگرام حالات

در بدست آوردن دیاگرام حالت می‌دانیم که باید به تعداد حالات موجود دایره‌ای کشید. و هر کدام را با یک حالت نام‌گذاری کنیم. سپس با استفاده از جدول حالات شروع به ترسیم خطوط می‌کنیم. به این صورت که با توجه به حالت فعلی و حالت بعدی و همچنین مقادیر ورودی و خروجی می‌توان طراحی را انجام داد به عنوان نمونه در مثال قبل زمانی که در حالت فعلی (1,0) می‌باشیم با گرفتن مقدار ورودی $x=0$ به حالت بعدی (0,0) می‌رویم و مقدار خروجی ما $z=1$ می‌شود.

مدل‌های میلی و مور

مدل‌های میلی مدل‌هایی (مدارهایی) هستند که خروجی تابعی از ورودی و حالت فعلی است.

مدل‌های مور مدل‌هایی (مدارهایی) هستند که خروجی فقط تابعی از حالت فعلی است.

مدل‌های میلی و مور دو مدار ترتیبی هستند و برای تشخیص میلی یا مور بودن یک مدار باید حتماً آن را تحلیل کرد.

در مدارهای مور با تغییر ورودی خروجی تغییر نمی‌کند بلکه تنها با تغییر کلاک عوض می‌شود. هریک مدل‌های میلی انتعطاف‌پذیری آنها و معمولاً تعداد حالات کمتر آنها می‌باشد.

شمارنده‌ها جزو مدارهای مور هستند که قابل تبدیل به مدل میلی نمی‌باشند.

دیاگرام حالت مدل میلی

دیاگرام حالت مدل مور

تبدیل مدارهای مور به میلی و میلی به مور:

مثال قبل نمونه‌ای از مدل میلی است مدار علاوه بر AB حالت فعلی به x (ورودی) نیز بستگی دارد.

تحلیل به کمک فلیپ فلاپ‌ها

وقتی در مسئله‌ای گفته می‌شود به صورت ترتیبی حل شود و یا از طریق فلیپ فلاپ‌ها طراحی شود به سود ما است که از فلیپ فلاپ‌ها استفاده کنیم. در مدارهای ترتیبی بیشترین عصبیری که استفاده می‌شود فلیپ فلاپ‌ها هستند.

آسان‌ترین فلیپ فلاپ‌ها برای طراحی KJK و T فلیپ فلاپ است.

مدار منطقی

RS فلیپ فلاپ مانند jk است تها تفاوت در حالت (1,1) ذر SR است که سیستم حالت نامعین خواهد داشت از SR در زمانی استفاده می‌کنند که بخواهیم ورودی‌ها را به صورت دستی وارد کیم. در این صورت اگر ورودی اشتباه وارد شود فلاپ فلاپ به حالت (1,1) می‌رود و سیستم متوقف می‌شود.

الویت استفاده از فلاپ فلاپ‌ها در طراحی:

1-Jk

2-SR

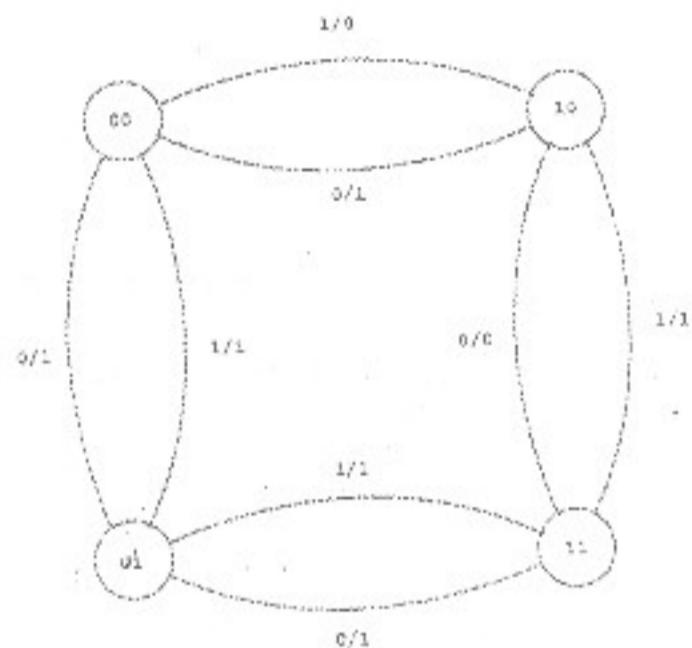
3-T

4-D

برای طراحی با فلاپ فلاپ باید ورودی‌های آن‌ها را (SR, Jk) به کمک توابع ترکیبی تنظیم (طراحی) می‌کیم برای این کار می‌توان از جدول حالات استفاده کرد به این صورت که باید حالات فعلی ورودی را بنویسیم و از روی دیاگرام حالات و یا شکل مسئله حالتهای بعدی و خروجی را بدست آوریم. سپس با استفاده از جدول بدست آمده و با مقایسه حالت فعلی و حالت بعدی و همچنین استفاده از جدول تحریک فلاپ فلاپ‌ها ورودی‌های لازم را برای اعمال به فلاپ فلاپ‌ها به دست می‌آوریم.

به عنوان مثال برای بدست آوردن جدول حالات به شکل فوق از روی دیاگرام حالات (۴-۱۳) خواهیم داشت.

حالات فعلی			وارودی			حالات بعدی			خروجی		
A	B	X	0	1		A	B	Y	0	1	
0	0	0	0	1		0	1	1	0	1	
0	0	1	1	0		1	0	0	1	0	
0	1	0	1	1		1	1	1	1	1	
0	1	1	0	0		0	0	1	1	0	
1	0	0	0	0		0	0	1	1	0	
1	0	1	1	1		1	1	1	1	1	
1	1	0	1	0		1	0	0	0	1	
1	1	1	0	1		0	1	1	1	1	



جدول تحریک فلاپ‌ها

خروجی فلاپ‌ها		وارودی فلاپ‌ها					
$Q_{(t)}$	$Q_{(t+1)}$	S	R	J	K	D	T
0	0	0	x	0	x	0	0
0	1	1	0	1	x	1	1
1	0	0	1	x	1	0	1
1	1	x	0	x	0	1	0

در این جدول با توجه به $Q_{(t)}, Q_{(t+1)}$ که خروجی فلاپ‌ها در حالت فعلی و بعدی هستند ورودی‌ها را می‌نویسیم.

به عنوان مثال:

برای SR فلاپ فلاپ داریم:

اگر به ورودی‌های SR, (0,0) بدهیم هیچ تغییری در خروجی مرحله قبل به وجود نمی‌آید.



اگر به ورودی های SR, (1,1) بدهیم حالت نامعین است.

اگر به ورودی SR, (1,0) بدهیم یعنی Set, R=0, S=1 می کند.

اگر به ورودی SR, (0,1) بدهیم یعنی Reset, R=1, S=0 می کند.

در حالت اول وقتی $Q_{(t-1)} = 0$ بوده $Q_t = 0$ شده است این حالت در SR نشان دهنده این است که با Reset شده یا بدون تغییر بوده است.

$$(0,0) \left\{ \begin{array}{l} \rightarrow \quad \begin{matrix} 0 & 0 \\ 0 & 1 \end{matrix} \text{ بدون تغییر} \\ \rightarrow \quad \begin{matrix} 0 & 1 \\ 0 & 0 \end{matrix} \text{ Reset} \end{array} \right\} \Rightarrow SR' + S'R' = S' \Rightarrow S' = 0, R = X$$

برای بدون تغییر بودن باید ورودی های R, S هر دو صفر باشند و برای Reset شدن باید R=1, S=0 باشد حال میترم ورودی ها را می نویسیم یعنی $S'R' + S'R$ این دو در یک یست متفاوت هستند ییت متفاوت حذف می شود و S' میماند یعنی S باید صفر باشد و R حالت بی اهمیت است چون فرقی نمی کند صفر باشد یا یک.

در حالتی که $Q_{t+1} = 0$ است یعنی Set شده پس ورودی SR باید به صورت (1,0) خواهد بود برای حالت $Q_{t+1} = 1$ یعنی Reset شده پس ورودی ها S, R به صورت (0,1) خواهد بود.

وقتی $Q_{t+1} = 1$ است دو حالت داریم یا بدون تغییر مانده یا set شده است.

$$(1,1) \left\{ \begin{array}{l} \rightarrow \quad \begin{matrix} 0 & 0 \\ 1 & 0 \end{matrix} \text{ بدون تغییر} \\ \rightarrow \quad \begin{matrix} 0 & 0 \\ 1 & 0 \end{matrix} \text{ Set} \end{array} \right\} \Rightarrow SR' + S'R' = R'$$

وقتی ورودی S, R صفر باشد خروجی مرحله بعدی همان خروجی مرحله قبلی است و هنگامی Set می شود که ورودی R, (1,0) باشد برای دو حالت میترم می نویسیم به صورت $S'R' + SR'$ این دو یک یست متفاوت دارند که حذف می کنیم در نهایت به R' می رسیم و ورودی S نیز حالت بی اهمیت است.

برای JK, فلیپ فلاپ داریم:

وقتی $j=k=1$ است خروجی همان مرحله قبل است.

وقتی $j=k=0$ است خروجی همان مرحله قبل است.

وقتی $j=1, k=0$ خروجی Set می شود یعنی اگر خروجی مرحله قبل 0 یا 1 باشد، خروجی مرحله بعد 1 است.

وقتی $j=0, k=1$ خروجی Reset می شود یعنی اگر خروجی مرحله قبل 0 یا 1 باشد، خروجی مرحله بعد 0 است.

با توجه به توضیحات بخش SR برای JK-F.F نیز داریم:

$$(0,0) \left\{ \begin{array}{l} \rightarrow \quad \begin{matrix} j & k \\ 0 & 0 \\ 0 & 1 \end{matrix} \\ \text{بدون تغییر} \end{array} \right\} \Rightarrow j'k' + j'k = j'$$

$$\frac{0 \quad x}{0 \quad x}$$

→ Reset

مدار منطقی

$$(0,1) \left\{ \begin{array}{l} \xrightarrow{\quad} \text{Set} \quad \begin{matrix} j & k \\ 1 & 0 \\ 1 & 1 \\ \hline 1 & x \end{matrix} \Rightarrow jk' + jk = j \\ \xrightarrow{\quad} \text{متهم} \end{array} \right.$$

$$(1,0) \left\{ \begin{array}{l} \xrightarrow{\quad} \text{Reset} \quad \begin{matrix} j & k \\ 0 & 1 \\ 1 & 1 \\ \hline x & 1 \end{matrix} \Rightarrow j'k + jk = k \\ \xrightarrow{\quad} \text{متهم} \end{array} \right.$$

$$(1,1) \left\{ \begin{array}{l} \xrightarrow{\text{بدون تغییر}} \begin{matrix} j & k \\ 0 & 0 \\ 1 & 0 \\ \hline x & 0 \end{matrix} \Rightarrow j'k' + jk' = k' \\ \xrightarrow{\quad} \text{Set} \quad \begin{matrix} j & k \\ 0 & 1 \\ 1 & 0 \\ \hline x & 0 \end{matrix} \end{array} \right.$$

برای D فلیپ فلاپ داریم:

هر چه به ورودی بدهیم همان را در خروجی می‌یابیم.

آخرین خروجی ما است پس هر مقداری که دارد همان هم در ورودی باید باشد.

برای T فلیپ فلاپ داریم:

اگر $T=0$ بود خروجی هیچ تغییری با خروجی مرحله قبل ندارد.اگر $T=1$ بود خروجی متهم خروجی مرحله قبل می‌شود.

مثال: به کمک فلیپ فلاپ J.k مداری طراحی کنید که یک عدد دو دویی، سه بیتی را به معادل gray آن تبدیل کند.

ورودی	حالات فعلی			خروجی			حالات بعدی					
	A _t	B	C	A _{t-1}	B	C	j _A	k _A	j _B	k _B	j _C	k _C
0	0	0	0	0	0	0	0	X	0	X	0	X
0	0	1	0	0	0	1	0	X	0	X	X	0
0	1	0	0	0	1	1	0	X	X	0	1	X
0	1	1	0	0	1	0	0	X	X	0	X	1
1	0	0	0	1	1	0	X	0	1	X	0	X
1	0	1	0	1	1	1	X	0	1	X	X	0
1	1	0	0	1	0	1	X	0	X	1	1	X
1	1	1	1	1	0	0	X	0	X	1	X	1
							0	0				

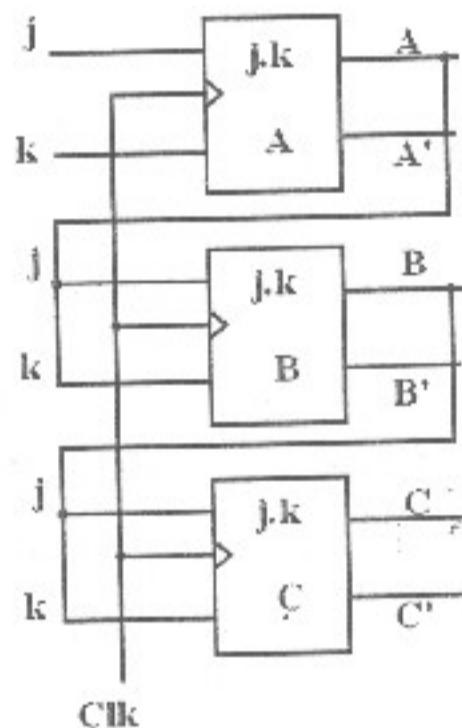
جدول کارنو

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

	AB	00	01	11	10	
A	0	X	X	1	3	2
	1	X	X	1	7	6

	BC	00	01	11	10	
A	0	X	1	X	1	2
	1	4	X	5	X	6

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



کاهش حالات

در طراحی‌ها از قلپ فلاب‌ها استفاده می‌کنیم که نسبت به تجهیزات مدارهای ترکیبی گران‌تر هستند. برای این‌که مدار از لحاظ هزینه بهینه باشد باید حالات معادل را پیدا کنیم تا تعداد قلپ فلاب‌ها کاهش یابد.

حالات معادل

به حالاتی گفته می‌شود که ورودی خروجی و حالت بعدی در هر دو State یکی باشد.

(State) به هر کدام از دایره‌ها در دیاگرام حالات و یا حالاتی متفاوت موجود در مدار گفته می‌شود)

در حالات معادل اگر ورودی و خروجی و مقصد یکی باشد (خروچی / ورودی باهم برابر) دو حالت باهم معادل هستند چون هر دو با ورودی و خروجی مشابه به State رفته‌اند.

مراحل کاهش حالات:

۱- تشکیل جدول حالات (به کمک توضیحاتی که در قبل آمده)

۲- حذف حالات معادل:

بعد از پیدا کردن حالات معادل یکی از آنها را حذف می‌کنیم و در هر کجای جدول حالات که از حالت حذف شده استفاده شده بود حالت معادل آن را قرار می‌دهیم.

۳- تشکیل جدول حالات جدید برای طراحی ورودی‌های جدید

کاهش حالات با استفاده از جدول:

نحوه پر کردن جدول به این صورت است که در خانه تقاطع ۲ حالت یا x یا \checkmark یا یک شرط می‌گذاریم

x: به این معنی که دو حالت با هم معادل نیست

b	x		
e	✓		
d	a,b		x
c	a,c	✓	

a b c d

✓: به این معنی که دو حالت با هم معادل است

حالات معادل حالاتی هستند که ورودی و خروجی و حالت بعدی آنها یکسان نامند

x: زمانی است که خروجی دو حالت با یکدیگر مساوی نبود. اگر هم خروجی یکی بود و هم حالت بعدی \checkmark می‌گذاریم

اگر خروجی یکی باشد ولی حالت‌های بعدی متفاوت باشد حالت فعلی یک شرط است وقتی خروجی‌ها یکی بود و حالت بعدی متفاوت

در صورتی معادلند که a,b (شرط) با هم معادل باشد در جدول به محل تقاطع ab توجه می‌کنیم چون در محل تقاطع x است پس a,b

معادل نیستند در نتیجه a,b هم معادل نیستند و در جدول به جای شرط x می‌گذاریم

هم در صورتی معادلند که a,c معادل باشد در جدول تقاطع خانه‌های a,c علامت \checkmark وجود دارد پس a,c معادلند در نتیجه a,c هممعادلند و به جای شرط \checkmark قرار می‌دهیم.

نام‌گذاری حالات

می‌توان به هر یک از حالات، نامی اختصاص داد که از حروف تشکیل شده باشد. به عنوان مثال می‌توان (a, b, c, \dots, z) را به عنوان یک حالت در نظر گرفت.

حالت فعلی	x	ورودی x	حالت بعدی	خروجی
a	0		b	1
b	0		e	0
c	0		d	1
d	0		b	1
e	0		a	1

حالات‌های d,a دارای ورودی حالت بعدی خروجی یکسان هستند پس $d \equiv a$ می‌باشد.

بنابراین در جدول هر جا a بود می‌توانیم d قرار می‌دهیم. اگر در جدول به جای a,d را قرار دهیم حالت‌های c,e نیز با هم معادل هستند یعنی ورودی حالت بعدی و خروجی آن‌ها پکی خواهد بود پس $c \equiv e$ خواهد شد به این ترتیب می‌توانیم ۵ فلپ فلاب را به ۳ فلیپ فلاب کاهش دادیم.

کاهش حالات به کمک جدول:

می‌خواهیم جدولی طراحی کنیم که در آن تمام حالات یک مسئله با هم مقایسه شوند اگر تعداد حالات را n فرض کنیم می‌خواهیم یک جدول $(n-1) \times (n-1)$ بدست آوریم که در آن تقاطع برای هر دو حالت تنها یک بار وجود دارد (برای دو حالت a,d تنها یک خانه تقاطع وجود دارد که هم برای a,d و هم برای a,d می‌باشد).

تخصیص حالات

به هر حالت یک کد اختصاص می‌دهیم در اینجا چون ۵ حالت داریم به هر کدام یک کد سه بیتی اختصاص می‌دهیم. لزومی ندارد این کدها به ترتیب باشند برای هر حالت متفاوت باید از m بیت استفاده کرد.

$$m = \lceil \log_2 5 \rceil$$

حالت فعلی	ورودی x	حالت بعدی	خروجی
a	0	b	1
b	0	e	0
c	0	d	1
d	0	b	1
e	0	a	1

$$a = 000$$

$$b = 001$$

$$c = 100$$

$$d = 111$$

$$e = 100$$

حالت فعلی	ورودی			حالت بعدی			خروجی
A	B	C	x	A	B	C	y
0	0	0	0	0	0	1	1
0	0	1	0	1	0	0	0
1	0	0	0	1	1	1	1
1	1	1	0	0	0	1	1
1	0	0	0	0	0	0	1

کاهش حالات با استفاده از جدول:

حالت فعلی	ورودی x	حالت بعدی	خروج
a	0	b	1
b	0	e	0
c	0	d	1
d	0	b	1
e	0	a	1

* ۵ تا حالت داریم پس جدول ما $4 \times 4 \times (n-1) \times (n-1)$ است

	b	X			
	c	b, d	X		
(۱)	d	✓	X	d, b	
(۲)	e	a, b	X	d, a	b, a
		a	b	c	d

* (۱) خروجی های a و b با هم برابرند پس به سراغ حالت بعدی آنها می رویم که هر دو b است پس معادلند.

* (۲) در صورتی a و b با هم برابرند یعنی معادلند که a و b با هم معادل باشند.

b	X			
c	X	X		
d	✓	X	X	
e	X	X	✓	X
	a	b	c	d

$$\left\{ \begin{array}{l} c \equiv e \\ a \equiv d \end{array} \right.$$

حالت فعلی	حالت بعدی	خروجی
a	b	1
b	c	0
c	a	1

خلاصه برای پیاده سازی مدارهای ترتیبی:

۱- با تشریح مسئله به یک جدول حالت می رسم.

۲- حالت را کاهش می دهیم.

۳- به هر حالت یک عدد بایری تخصیص می دهیم و جدول حالت را دوباره می سازیم.(تخصیص حالت)

۴- جدول تحریک فلیپ فلاپ موردنیاز را می نویسیم.

۵- معادلات را از روی جدول تحریک بحسب می آوریم

۶- دیاگرام منطقی رارسم می کنیم.

مثال: به کمک FF, JK مدار مربوط به دیاگرام حالات زیر را طراحی کنید.

حالت فعلی		حالت بعدی	
		x	y
A	0	B	1
A	1	E	0
B	0	E	1
B	1	D	0
C	0	B	1
C	1	E	0
D	0	B	0
D	1	C	0
E	0	E	0
E	1	D	0



$$A \equiv B \xrightarrow{\text{اگر}} \begin{cases} B \not\equiv E \\ D \equiv E \end{cases} \rightarrow \text{معادل بست}$$

$$\begin{array}{lll} A \equiv C & \checkmark & x \ A \not\equiv D \ x \\ x \ A \not\equiv E & \times & x \ C \not\equiv D \ x \\ x \ C \not\equiv E & \times & x \ B \not\equiv D \ x \end{array}$$

$$B \equiv C \xrightarrow{\text{اگر}} \begin{cases} B \not\equiv E \\ D \equiv E \end{cases} \times$$

$$D \equiv E \xrightarrow{\text{اگر}} \begin{cases} B \not\equiv E \\ C \equiv D \end{cases} \times$$

فقط A و C با هم معادلند پس به جای C می‌توان A را گذاشت و ستون مربوط به C را حذف کرد.

$$\begin{array}{l} A \rightarrow 00 \\ B \rightarrow 01 \\ D \rightarrow 10 \\ E \rightarrow 11 \end{array}$$

حالت فعلی حالت بعدی

	m	n	x	m	n	y	j_m	k_m	j_n	k_n
A	0	0	0	0	1	1	0	X	1	X
	0	0	1	1	1	0	1	X	1	X
B	0	1	0	1	1	1	1	X	X	0
	0	1	1	1	0	0	1	X	X	1
D	1	0	0	0	1	0	X	1	1	X
	1	0	1	0	0	0	X	1	0	X
E	1	1	0	1	1	0	X	0	X	0
	1	1	1	1	0	0	X	0	X	1

m \ nx	00	01	11	10
0	1 0	1 1	1 3	1 2
1	X 4	X 5	X 7	X 6

$$j_m = n + x$$

m \ nx	00	01	11	10
0	X 2 0	X 1	X 3	X 2
1	1 X	1 5	7	6

$$k_m = n'$$

مدار منطقی

m'nx	00	01	11	10
0	1 0	1 1	X 2	X 3
1	1 4	5	X 7	X 8

$j_n = m' + x'$

m'nx	00	01	11	10
0	X 0	X 1	1 3	2
1	1 X	X 5	1 7	6

$k_n = n' + x$

طراحی شمارنده

دو نوع شمارنده داریم:

(۱) با ترتیب

(۲) بدون ترتیب

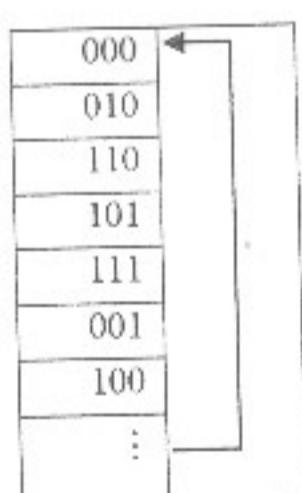
شماره با ترتیب: یعنی این که اگر صفر را شمرد بعد یک را بشمارد، بعد ۲ و همین طور ادامه پیدا کند تا به یک X برسد که بعد از X دور بزند و صفر را بشمارد یا اگر از ۵ شروع کرد بعد ۶ و بعد ۷ را بشمارد تا عدد X، سپس دور بزند و باز از ۵ شروع به شمارش بکند یعنی لزومی ندارد از صفر شروع به شمارش کند ولی از هر کجا که شروع به شمارش کرد به ترتیب جلو برود.

شماره بدون ترتیب: می‌تواند از هر عددی شروع شود ولی اعداد را به ترتیب نمی‌شمارد و قتنی تا x را بشمارد دور می‌زند و از عدد اول شروع به شمارش می‌کند.

● شمارنده‌ها یا دو دویی هستند یا غیر دو دویی.

شمارنده دودویی با ترتیب: یعنی اگر سه بیتی است از 0 0 0 شروع کند به شمارش تا به 1 1 1 برسد.

شمارنده دودویی بدون ترتیب: از 0 0 0 شروع کند بعد 0 1 0 1 1 1 1 0 1 0 1 1 1 را بشمارد دوباره بازگردد به اول.

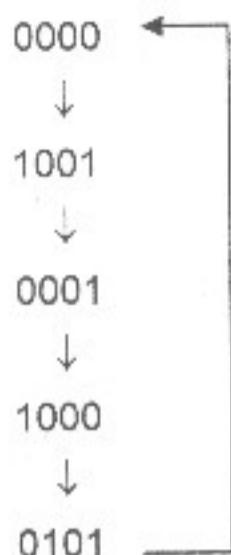


برای مجموعه $\{0, 1, 2, 3, \dots, 8, 9\}$ شمارنده بدون ترتیب این مجموعه به صورت زیر است:

2, ←	7
↓	↑
5	1
↓	↑
4	6
↓	↑
8 → 0 → 9	



شمارنده غیر دودویی: شمارنده کد های BCD یک شمارنده غیر دودویی است که از (0000) شروع می شود و به 1001 می رود و دوباره به اول باز می گردد این شمارنده به 1111 نمی رسد پس دودویی نیست حال اگر این شمارنده بدون ترتیب باشد می تواند مثلاً به صورت زیر باشد.



مشاهده می شود که در یک شمارنده دودویی n بیت تعداد اعداد قابل شمارش 2^n می باشد یعنی به عنوان مثال از صفر تا $1 - 2^n$ اما در یک شمارنده غیر دودویی با n بیت هیچ دلیلی برای شمارش 2^n عدد وجود ندارد. به عنوان مثال شمارنده اعداد BCD از 4 بیت استفاده می کند اما 2^4 یعنی 16 عدد را نمی شمارد بلکه تنها 10 عدد می شمارد.

در یک شمارنده دودویی از n فلیپ فلاب استفاده می شود تا 2^n عدد را بشمارد اما در یک شمارنده غیر دودویی n بیت از n فلیپ فلاب استفاده می شود اما ممکن است 2^n عدد را نشمارد یعنی از فلیپ فلاب ها استفاده بهینه نمی شود.

مثال:

یک شمارنده دودویی (با ترتیب) سه بیتی طراحی کنید که از صفر تا 7 را بشمارد (با استفاده از T فلیپ فلاب) در یک شمارنده وقتی از فلیپ فلاب استفاده می کنیم باید حالت فعلی و حالت بعدی را به وجود بیاوریم حالت فعلی در شمارنده، همان ترکیبات متغیرها می باشد یعنی برای n متغیر 2^n حالت فعلی به وجود می آید. حالت بعدی عددی است که به وجود می آید

حالت فعلی			حالت بعدی			ورودی فلیپ فلاب		
A	B	C	A'	B'	C'	T _A	T _B	T _C
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	0	1
1	1	1	0	0	0	1	1	1

شمارنده ترتیبی است پس از 000 شروع می کند به شمارش می کند تا به 7 برسد بعد برگردد به صفر بر می گردد، چون با استفاده از T فلیپ فلاب می خواهیم طراحی کنیم، باید ورودی های T, FF را به دست آوریم. اگر خروجی مرحله بعدی متمم خروجی مرحله قبل بود $T=1$ است و اگر خروجی مرحله بعد با مرحله قبل یکی بود $T=0$ است.

در مقایسه حالت فعلی و حالت بعدی داریم:

وقتی حالت فعلی (000) است و حالت بعدی (001) بوده، A' نیز صفر مانده پس T_A نیز صفر است. وقتی حالت فعلی (001) بوده و حالت بعدی (010) است، B' نیز صفر مانده پس T_B نیز صفر است. وقتی حالت فعلی (000) است و حالت بعدی (001) بوده، C' یک شده پس T_C یک است. وقتی حالت فعلی (001) بوده و حالت بعدی (010) است، A' نیز صفر مانده پس T_A صفر است. وقتی حالت فعلی (001) بوده و حالت بعدی (010) است، B' یک شده پس T_B یک است. وقتی حالت فعلی (001) بوده و حالت بعدی (010) است، C' صفر شده پس T_C یک است.

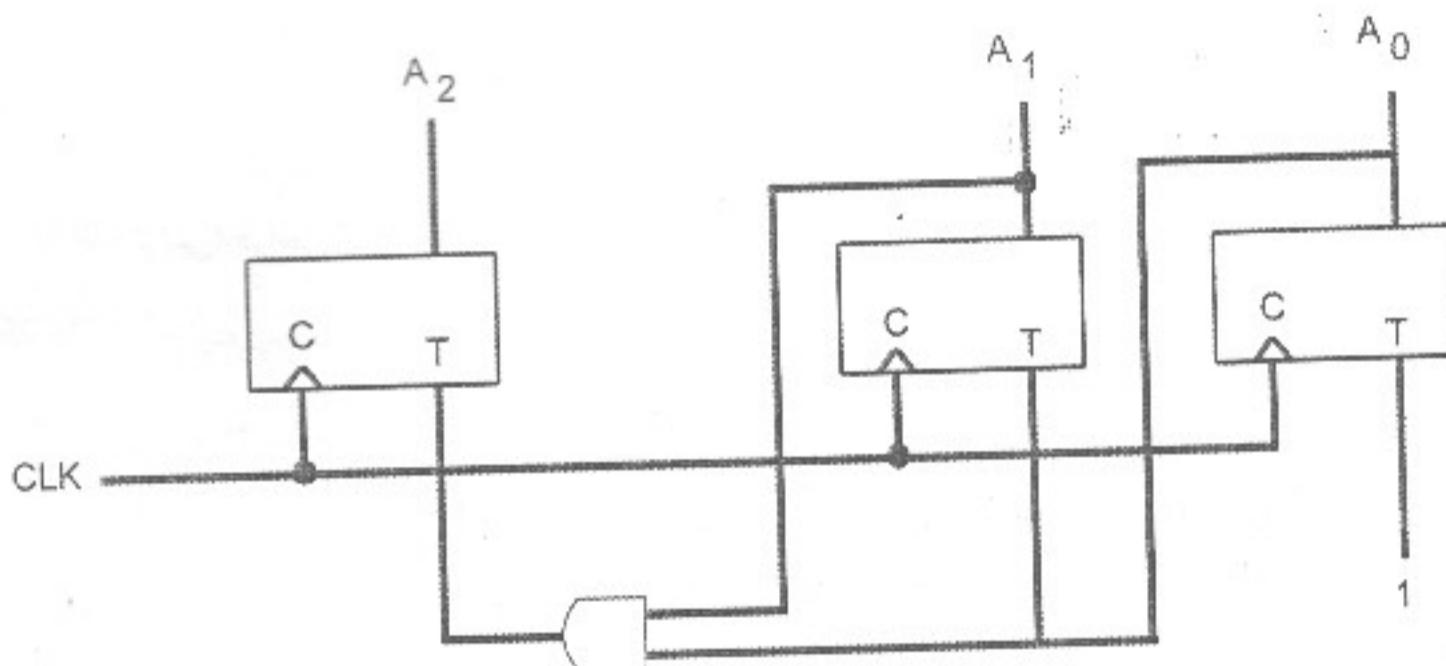
بدین ترتیب بقیه جدول را نیز کامل می کنیم بعد میترم ها را می نویسیم.

$$T_a = bc$$

$$T_b = c$$

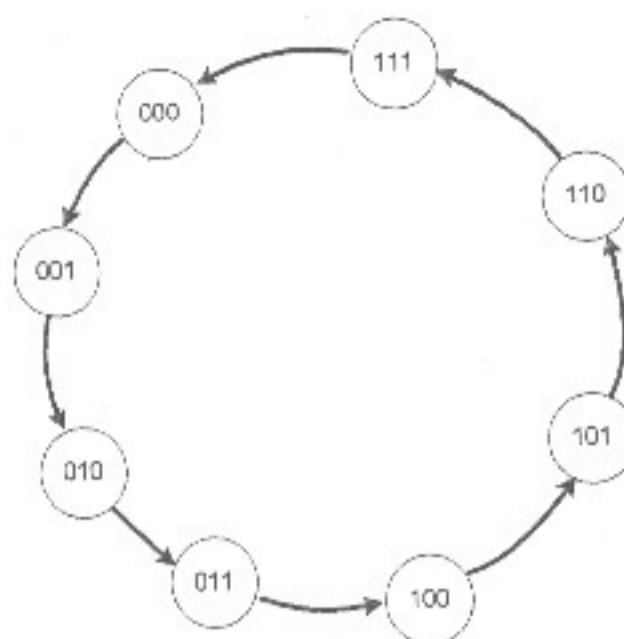
چون هر چه اعمال کردہ ایم، در T_c تأثیری نداشته و همان مقدار یک مانده است بنابراین:

$$T_c = 1$$

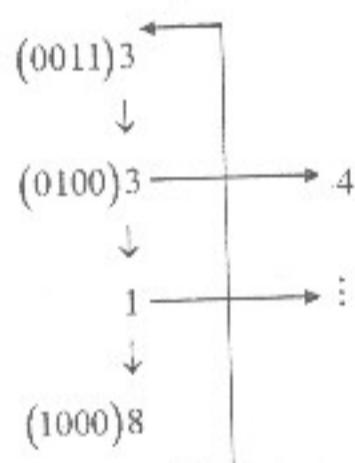


نمودار منطقی شمارنده دو دویی 3 بیت

این دیاگرام با دیاگرام حالت های قبلی متفاوت است قبلاً برای هر State یک ورودی داشتیم اما اینجا ورودی خط clk است. اگر خط clk نباشد ورودی تغییر نمی کند. یعنی ورودی به صورت دستی نداریم.

مدار منطقی


در شمارنده غیر دودویی می توانیم یک مقدار مشخص برا به وجود بیاوریم که از آنجا شروع به شمارش بکنیم و یک مقدار مشخص (دلخواه) هم به وجود بیاوریم که به آن ختم شود مثلاً در BCD از 3 (0011)، شروع به شمارش کنیم تا 8 (1000) بشمارد بعد به اول بازگردد.



- یک شمارنده BCD (تریسی) طراحی کنید (با استفاده از k-فیلیپ فلاب)

اگر مدار به یک حالت ناخواسته و نامعین رسید حالت بعدی را صفر در نظر بگیرید.

حالات فعلی				حالات بعدی				وروودی‌های فلیپ فلاب							
A	B	C	D	A	B	C	D	J _A	k _A	J _B	k _B	J _C	K _C	J _D	k _D
0	0	0	0	0	0	0	1	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	0	X	0	X	1	X	1	X
0	0	1	0	0	0	1	1	0	X	0	X	0	1	X	1
0	0	1	1	0	1	0	0	0	X	1	X	1	X	1	X
0	1	0	0	0	1	0	1	0	X	X	0	0	X	1	X
0	1	0	0	0	1	1	0	0	X	X	0	1	X	X	1
0	1	0	1	0	1	0	0	0	X	X	0	X	0	1	X
0	1	1	0	0	1	1	1	0	X	X	1	X	1	X	1
0	1	1	1	1	0	0	0	1	X	0	X	0	X	1	X
1	0	0	0	1	0	0	1	1	X	1	0	X	0	X	1
1	0	0	1	0	0	0	0	X	1	0	X	X	1	0	X
1	0	1	0	0	0	0	0	X	1	0	X	X	1	X	1
1	0	1	1	0	0	0	0	X	1	X	1	0	X	0	X
1	1	0	0	0	0	0	0	X	1	X	1	0	1	X	1
1	1	0	1	0	0	0	0	X	1	X	1	X	1	0	1
1	1	1	0	0	0	0	0	X	1	X	1	X	1	X	1
1	1	1	1	0	0	0	0	X	1	X	1	X	1	X	1

مدار منطقی

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

$k_B = CD + AB$

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

$J_B = A'CD$

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

$$J_A = A + BCD$$

$$K_A = 1$$

مدار همنطقی

AB	CD	00	01	11	10
00	1	X	X	1	2
01	1	0	1	3	
11				1	6
10	1	X	X	15	14
11	1	X	X	11	10
10	8	9	11		

$$J_D = A' + B'C'$$

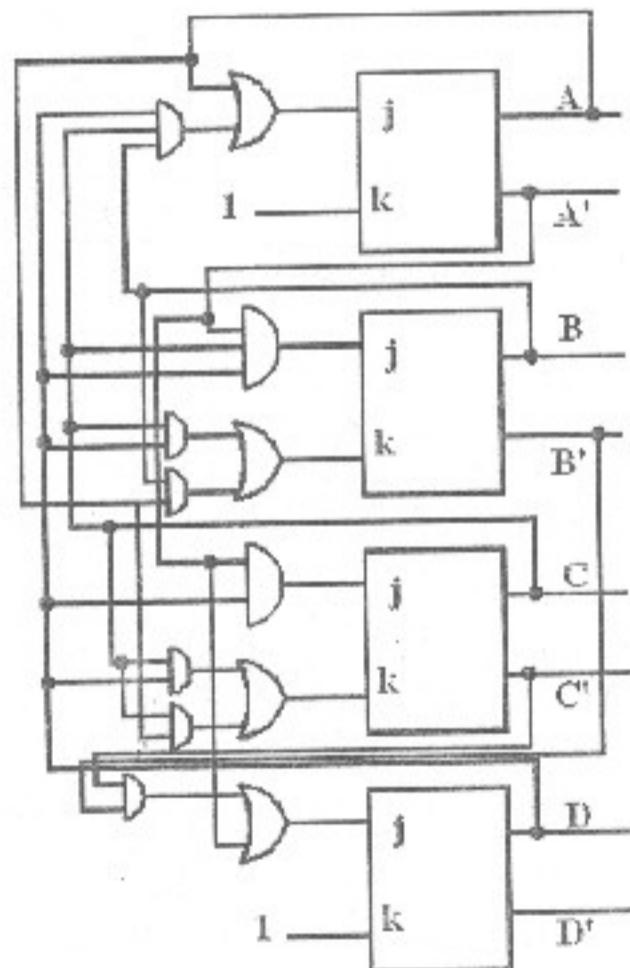
AB	CD	00	01	11	10
00	X	X	1	3	2
01	X	0	5	1	6
11	X	12	13	15	14
10	X	8	9	11	10

$$K_C = CD + AC$$

AB	CD	00	01	11	10
00		1	X	3	2
01		1	X	7	6
11				15	14
10		8	9	11	10

$$J_C = A'D$$

$$K_D = 1$$



مدار منطقی

مثال:

یک شمارنده دودویی (سه بیت) با ترتیب با استفاده از RS فلپ فلاب طراحی کنید؟

A	B	C	A	B	C	S _A	R _A	S _B	R _B	S _C	R _C
0	0	0	0	0	1	0	x	0	x	1	0
0	0	1	0	1	0	0	x	1	0	0	1
0	1	0	0	1	1	0	x	x	1	1	0
0	1	1	1	0	0	1	0	0	1	0	1
1	0	0	1	0	1	x	1	0	x	1	0
1	0	1	1	1	0	x	1	1	0	0	1
1	1	0	1	1	1	x	1	x	1	1	0
1	1	1	0	0	0	0	1	0	1	0	1

$$S_A = A'BC$$

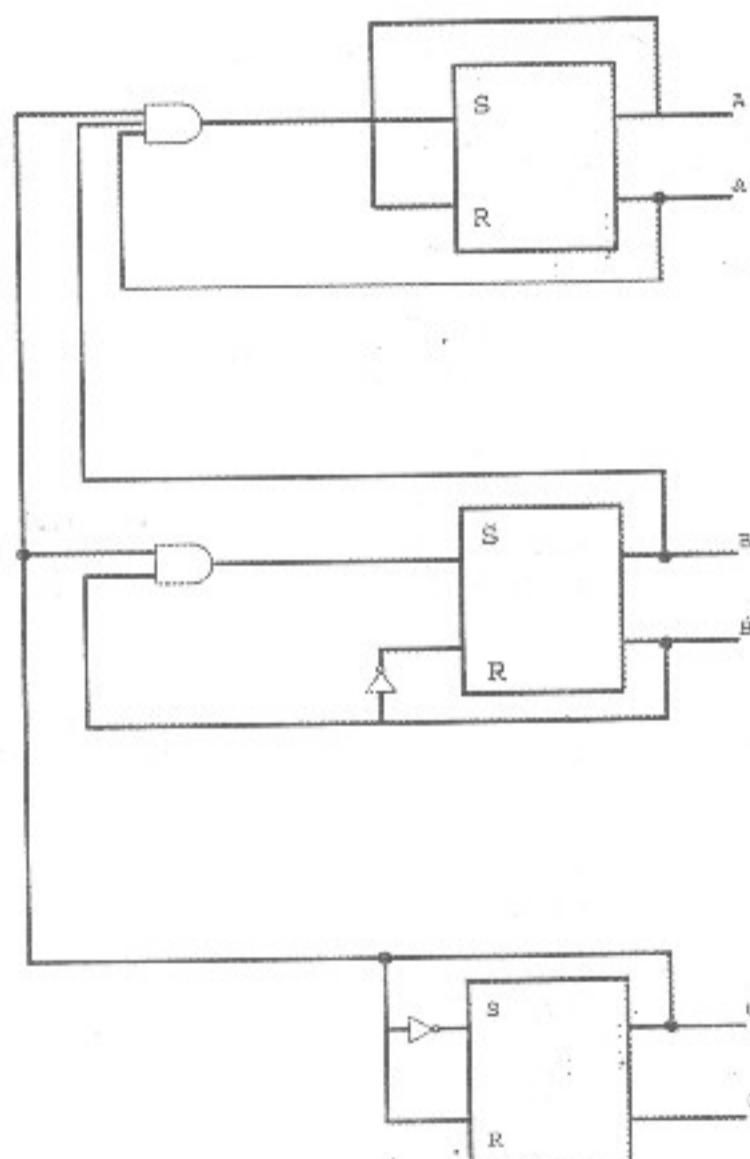
$$R_A = A$$

$$S_B = B'C$$

$$R_B = B$$

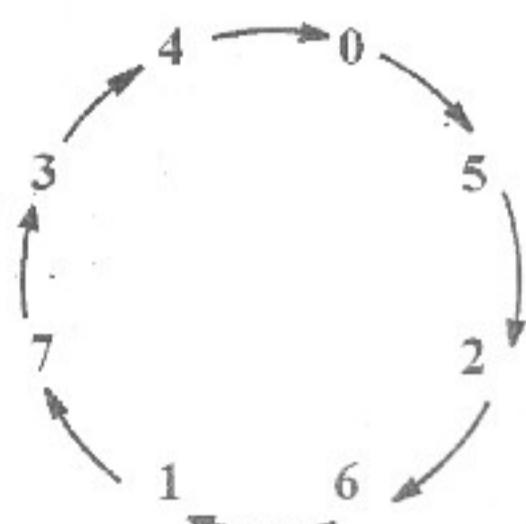
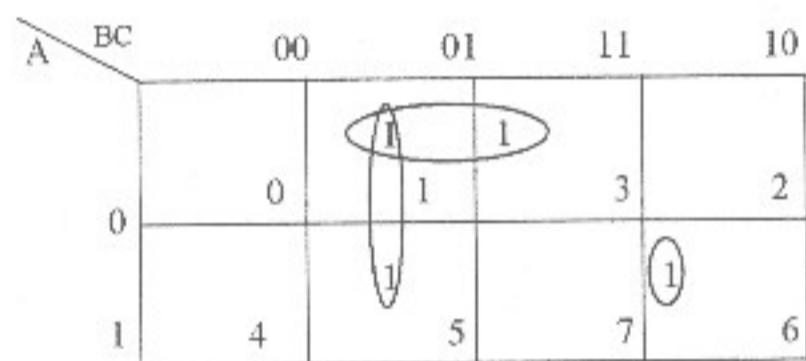
$$S_C = C'$$

$$R_C = C$$



شماره نده غیر ترتیبی سه دستی

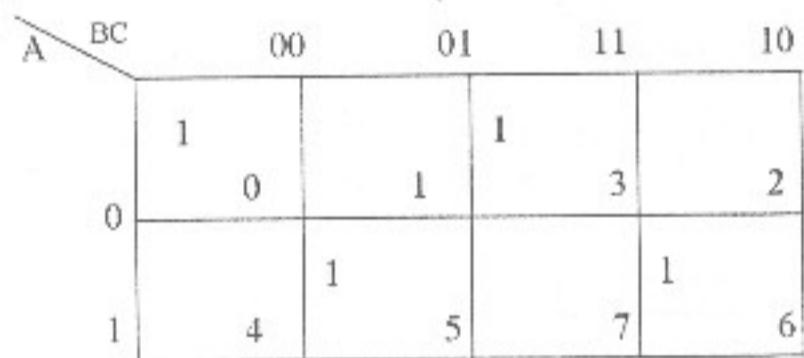
A	B	C	A	B	C	T _A	T _B	T _C
0	0	0	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0
0	1	0	1	1	0	1	0	0
0	1	1	1	0	0	1	1	1
1	0	0	0	0	0	1	0	0
1	0	1	0	1	0	1	1	1
1	1	0	0	0	1	1	1	1
1	1	1	0	1	1	1	0	0



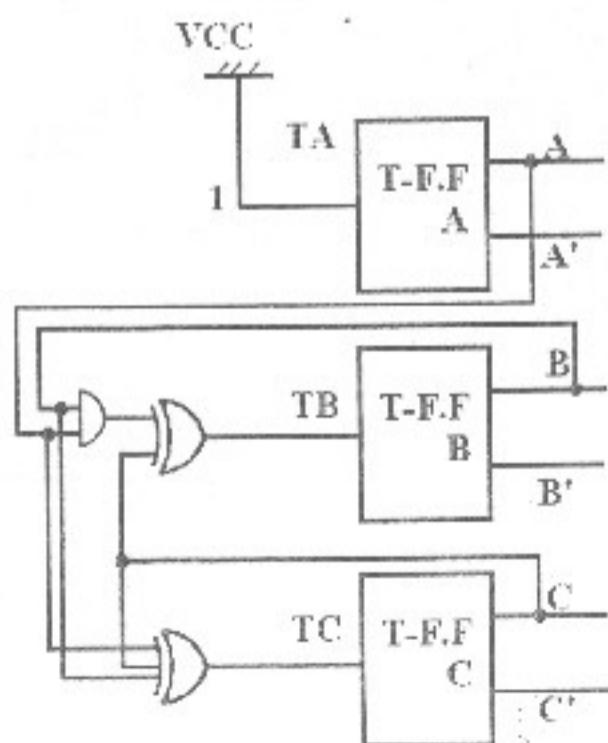
$$T_A = 1$$

$$T_B = \underline{A'C} + \underline{B'C} + ABC'$$

$$C(A' + B') + C'(AB) = C(\overline{AB}) + C'(AB) = C \oplus (AB)$$



$$T_C = A \odot B \odot C$$



مدار منطقی

مثال: به کمک j.K و T فلایپ مداری طراحی کنید که عملیات شمارش صفر تا 7 را به صورت صعودی و 7 تا صفر را به صورت ترولی انجام دهد.

A	B	C	x	A	B	C	j _A	k _A	j _B	k _B	j _C	k _C	S _A	R _A	S _B	R _B	S _C	R _C	
0	0	0	0	0	0	1	0	X	0	X	1	X	0	X	0	X	1	X	
0	0	0	1	1	1	1	1	X	1	X	1	X	1	0	1	X	1	X	
0	0	1	0	0	1	0	0	X	1	X	X	1	0	X	1	X	X	1	
0	0	1	1	0	0	0	0	X	0	X	X	1	0	X	0	X	X	1	
0	1	0	0	0	1	1	0	X	X	0	1	X	0	X	X	0	1	1	X
0	1	0	1	0	0	1	0	X	X	1	1	X	0	X	X	1	1	X	
0	1	1	0	1	0	0	1	X	X	0	X	1	1	0	X	1	X	1	X
0	1	1	1	0	1	0	0	X	X	0	X	1	0	X	X	0	X	X	1
1	0	0	0	1	0	1	X	0	0	X	1	X	X	0	0	X	1	1	X
1	0	0	1	0	1	1	X	1	1	X	1	X	0	1	1	X	1	1	X
1	0	1	0	1	1	0	X	0	1	X	X	1	X	0	1	X	X	1	X
1	0	1	1	1	0	0	X	0	0	X	X	1	X	0	0	X	X	0	X
1	1	0	0	1	1	1	X	0	X	0	1	X	X	0	X	0	1	1	X
1	1	0	1	1	0	1	X	0	X	1	1	X	X	0	X	1	1	X	
1	1	1	0	0	0	0	X	1	X	1	X	1	0	1	X	1	X		
1	1	1	1	1	1	0	X	0	X	0	X	1	X	0	X	0	X	X	1

T_A	T_B	T_C	D_A	D_B	D_C
0	0	1	0	0	1
1	1	1	1	1	1
0	1	1	0	1	0
0	0	1	0	0	0
0	0	1	0	1	1
0	1	1	0	0	1
1	1	1	1	0	0
0	0	1	0	1	0
0	0	1	1	0	1
1	1	1	0	1	1
0	1	1	1	1	0
0	0	1	1	0	0
0	0	1	1	1	1
0	1	1	1	0	1
1	1	1	0	0	0
0	0	1	1	1	0

Cx 00 01 11 10

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

$$j_A = BCx' + B'C'x$$

Cx 00 01 11 10

AB	00	01	11	10
00	X'	X	X	X
01	X	X	X	X
11				
10				

$$k_A = BCx' + B'C'x$$

Cx

AB	00	01	11	10
X	X	X	X	X
X	X	X	X	X
	1			1

$$j_B = c'x + cx' = C \oplus x$$

Cx

AB	00	01	11	10
	(1)			(1)
X	X	X		
X		X	X	X

$$S_A = A'B'C'x + A'BCx$$

Cx

AB	00	01	11	10
X	X	1	1	1
	1			1
X	X	X	X	X

$$k_B = c'x + cx' = C \oplus x$$

Cx

AB	00	01	11	10
X		X	X	X
X	X	X	X	
				(1)

$$R_A = AB'C'x + ABCx'$$

مدار منطقی

	Cx	AB	
X	1		1
X		X	
		1	1

$$S_B = B'C'x + B'Cx'$$

	Cx	AB	
X		X	
X			1
X		X	

$$R_B = BCx' + BC'x$$

	Cx	AB	
X	1		
X			1
		1	

$$T_A = B'C'x + BCx'$$

	Cx	AB	
X	1		1
X			1
		1	

$$T_B = C'x + CX' = C \oplus x$$

	Cx	AB	
X	1		
X			1
		1	

	Cx	AB	
X	1		1
X			1
		1	

$$D_A = ACx' + ABx' + AB'C + A'BCx' + A'B'C'x$$

$$D_B = BCx' + BCx + B'Cx' + B'C'x = B(C \odot x) + B(C \oplus x) = B \oplus (C \oplus x)$$

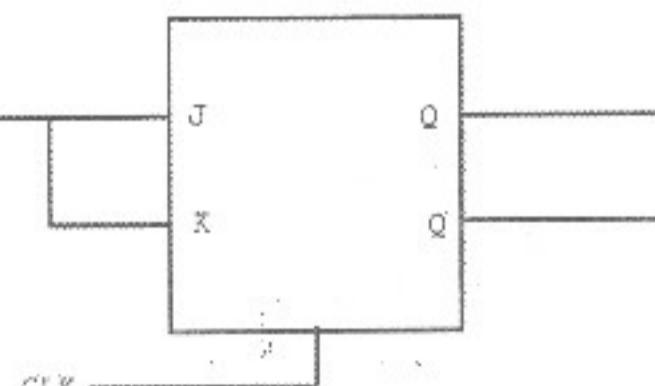
$$B(C \odot x) = \underbrace{B(\overline{C \oplus x})}_{\alpha} + \underbrace{B(C \oplus x)}_{\alpha} = B\alpha' + B\alpha = B \oplus C \oplus x$$

بدست آوردن فلیپ فلاپ‌ها از روش یکدیگر

می‌خواهیم T فلیپ فلاپ‌ها را با استفاده از jk فلیپ فلاپ بدست آوردم توانیم ورودی‌های j,k فلیپ فلاپ را هم وصل کنیم تا T فلیپ فلاپ ساخته می‌شود.

به این معنی که اگر یک jk فلیپ فلاپ را به صورت تک ورودی در آوریم، توانسته‌ایم یک T فلیپ فلاپ بسازیم.

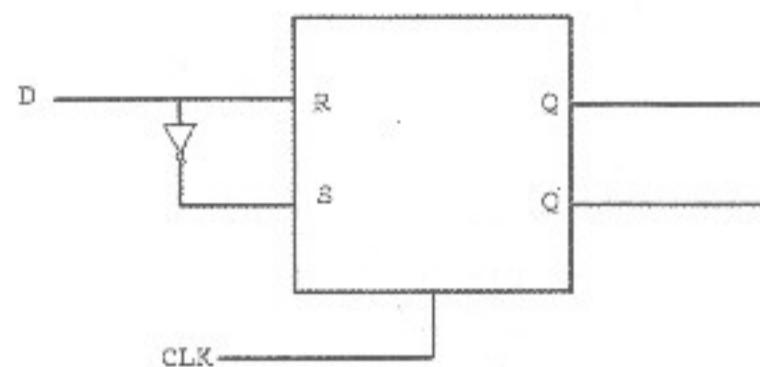
j	k	Q_{t+1}	T	Q_{t+1}
0	0	Q_t	0	Q_t
0	1	Reset	\Rightarrow	
1	0	Set		
1	1	Q'_t	1	Q'_t





اگر بخواهیم D فلیپ فلاب را با استفاده از SR فلیپ فلاب بسازیم باید در ورودی‌های S,R را به هم وصل می‌کنیم به این صورت که خود S و متمم R را به هم وصل می‌کنیم. یعنی اینکه کافیست از ورودی S یک خط بگیریم و آن را مکمل کنیم و وارد R کنیم.

S	R	Q_{t+1}
0	0	Q_t
0	1	Reset
1	0	Set
1	1	نامعین

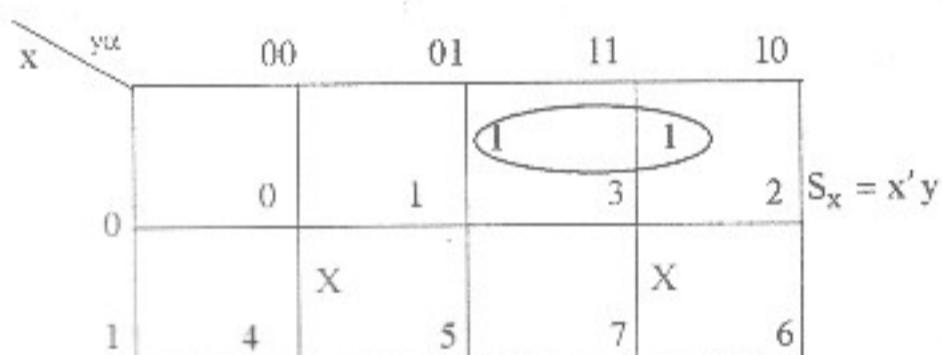


ولی با SR فلیپ فلاب به راحتی نمی‌توانیم T فلیپ فلاب بسازیم چون در SR حالت (1,1) نامعین است. و اگر R و S را به هم وصل کنیم و تشکیل یک ورودی بدھیم آن وقت زمانی که به ورودی مقدار یک برا اختصاص دهیم حالت نامعین به وجود آورده‌ایم. اما با (JK) فلیپ فلاب می‌توان یک D فلیپ فلاب بسازیم کافیست که یک ورودی را یک بار خودش و یک بار متمم‌ش را بسازیم و خودش را به خط J و متمم‌ش را به خط K وصل کنیم.

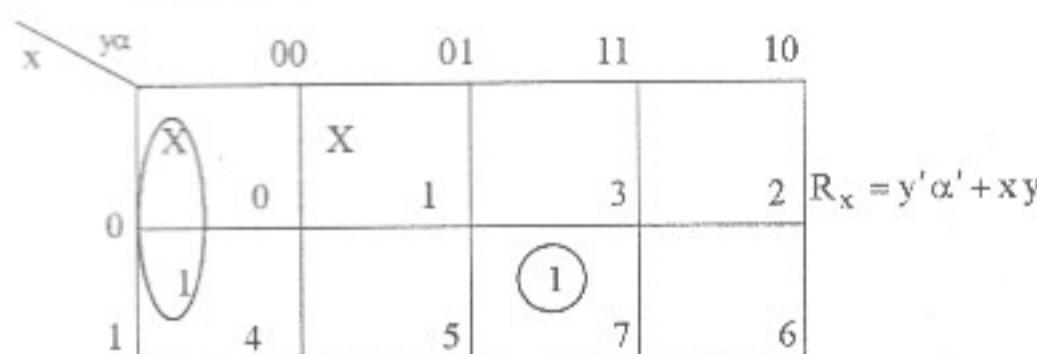
مجموعه مسائل فصل ۴:

۱- با توجه به دیاگرام حالات زیر به کمک فلپ فلام RS مدار را طراحی کنید.

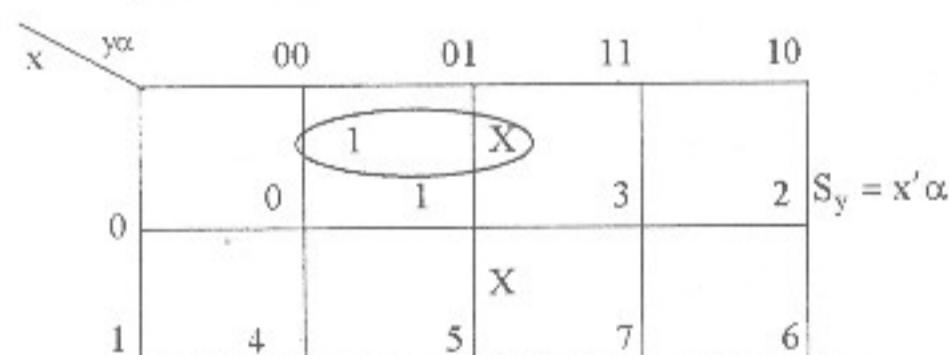
x	y	α	x	y	β	S_x	R_x	S_y	R_y
0	0	0	0	0	0	0	X	0	X
0	0	1	0	1	0	0	X	1	0
0	1	0	1	0	1	1	0	0	1
0	1	1	1	1	1	1	0	X	0
1	0	0	0	0	1	0	1	0	X
1	0	1	1	0	0	X	0	0	X
1	1	0	1	0	0	X	0	0	1
1	1	1	0	1	1	0	1	X	0



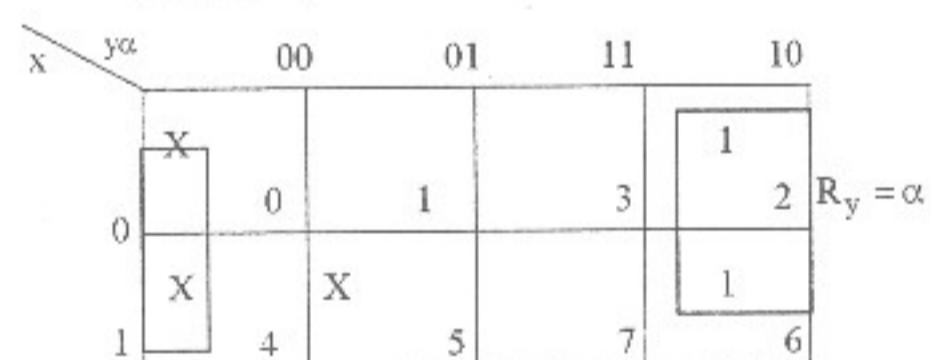
$$S_x = x'y$$



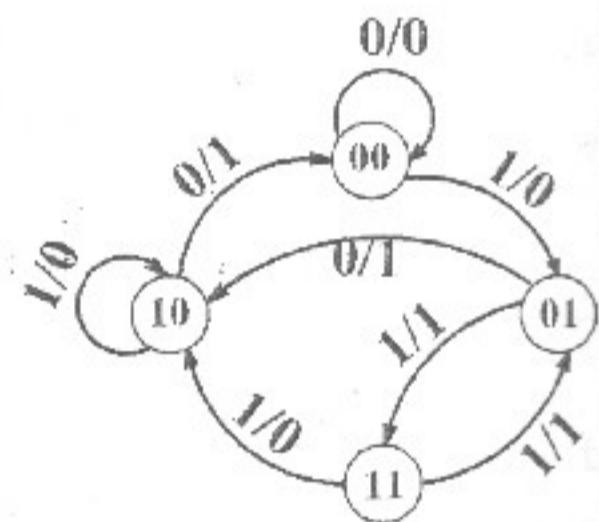
$$R_x = y'\alpha' + xy$$



$$S_y = x'\alpha$$



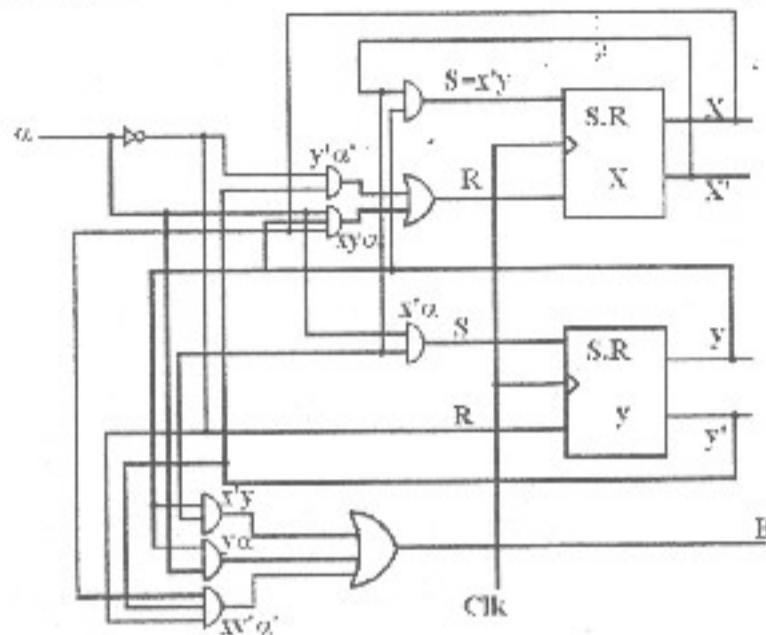
$$R_y = \alpha$$



$x \quad ya$

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

$$\beta = x'y + y\alpha + xy'\alpha'$$



۲ - به کمک J.K-F.F شمارندهای که دیاگرام زیر را پوشش می‌دهد را طراحی کنید. (حالات ناخواسته را صفر قرار دهید).

$0 \rightarrow 2 \rightarrow 4 \rightarrow 6$



A	B	C	A	B	C	J _A	K _A	J _B	K _B	J _C	K _C
0	0	0	0	0	0	0	X	1	X	0	X
0	0	1	0	0	0	0	X	0	X	X	1
0	1	0	1	0	0	1	X	X	1	0	X
0	1	1	0	0	0	0	X	X	1	X	1
1	0	0	1	0	0	X	0	1	X	0	X
1	0	1	0	0	0	X	1	0	X	X	1
1	1	0	0	0	0	X	1	X	1	0	X
1	1	1	0	0	0	X	1	X	1	X	1
										1	0
											1

$A \quad BC$

	00	01	11	10
0	X	X	X	X
1	0	1	1	1

$$K_A = C + B$$

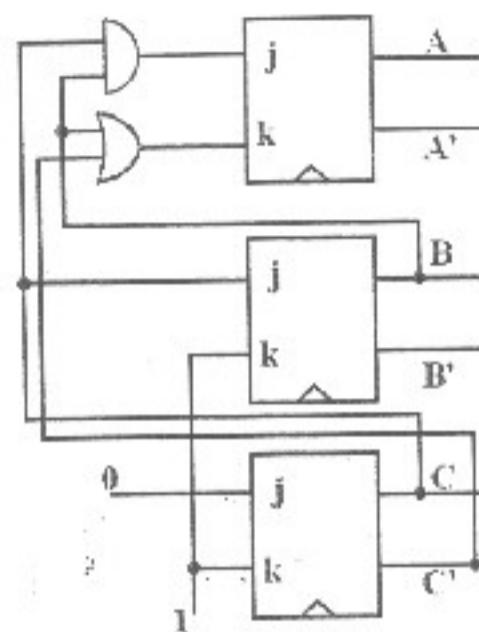
مدار منطقی

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

$$J_A = BC'$$

A	BC	00	01	11	10
0		1	0	1	X
1	1	4	5	X	7
					6

$$J_B = C'$$



۳- شمارنده‌ای طراحی کنید به کمک T-F-F که دیاگرام زیر را بشمارد.

$$\begin{array}{cccccc} 0 \rightarrow & 1 \rightarrow & 7 \rightarrow & 2 \rightarrow & 6 \\ \uparrow & & 4 \leftarrow & \leftarrow 5 & 3 & \downarrow \end{array}$$

$$\lceil \log_2^2 \rceil = 3$$

A	B	C	A	B	C	T _A	T _B	T _C
0	0	0	0	0	1	0	0	1
0	0	1	1	1	1	1	1	0
0	1	0	1	1	0	1	0	0
0	1	1	1	0	1	1	1	0
1	0	1	1	0	0	0	0	1
1	1	0	0	1	1	1	0	1
1	1	1	0	1	0	1	0	1

A	BC	00	01	11	10
0		0	1	1	2
1	1	4	5	1	6

$$T_A = A'C + AC' + B$$

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

$$T_B = CA'$$

مدار منطقی

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

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

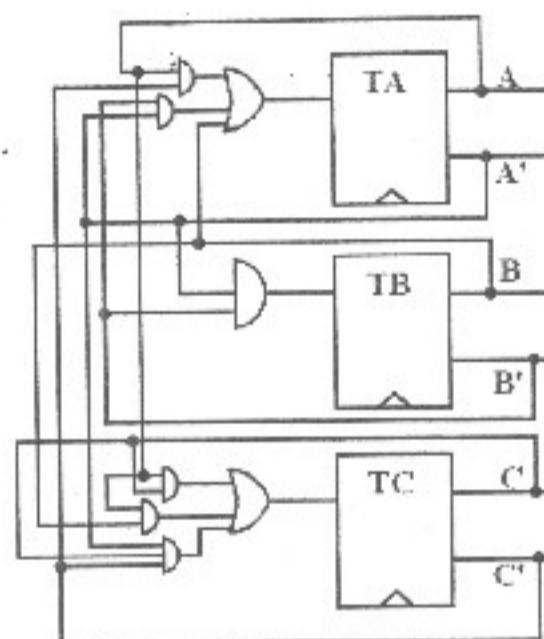
A B C

1 0 1

1 0 0 1

1 0 1 2

1 0 1 6



۴- به کمک R.S - F.F یک شمارنده چهاربیتی طراحی کنید که برای ورودی زوج، حالت بعدی به اضافه ۳ حالت فعلی باشد و برای ورودی فرد از حالت فعلی ۲ تا کم کنیم تا حالت بعدی بدست آید. (طراحی انجام شود)

A	B	C	D	A	B	C	D	S _A	R _A	S _B	R _B	S _C	R _C	S _D	R _D
0	0	0	0	0	0	1	1	0	X	0	X	1	0	1	0
0	0	0	1	1	1	1	1	1	0	1	0	1	0	X	0
0	0	1	0	0	1	0	1	0	X	1	0	0	1	1	0
0	0	1	1	0	0	0	1	0	X	0	X	0	1	X	0
0	1	0	0	0	1	1	1	0	X	X	0	1	0	1	0
0	1	0	1	0	0	1	1	0	X	0	1	1	0	X	0
0	1	1	0	1	0	0	1	1	0	0	1	0	1	1	0
0	1	1	1	0	1	0	1	0	X	X	0	0	1	X	0
1	0	0	0	1	0	1	1	X	0	0	X	1	0	1	0
1	0	0	1	0	1	1	1	0	1	1	0	1	0	X	0
1	0	1	0	1	1	0	1	X	0	1	0	0	1	1	0
1	0	1	1	1	0	0	1	X	0	0	X	0	1	X	0
1	1	0	0	1	1	1	1	X	0	X	0	1	0	1	0
1	1	0	1	1	0	1	1	X	0	0	1	1	0	X	0
1	1	1	0	0	0	0	1	0	1	0	1	0	1	1	0
1	1	1	1	1	1	1	0	X	0	X	0	0	1	X	0

1 0

مدار منطقی

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

$S_A = A'B'C'D + A'BCD'$

	CD	00	01	11	10	AB
00	X	0	1	X	2	
01	X	4	5	X	6	
11		12	13	15	14	
10		8	9	11	10	

$S_A = AB'C'D + ABCD'$

$$S_B = B'C'D + B'CD'$$

$$R_B = BC'D + BCD'$$

$$S_C = C'$$

$$R_C = C$$

$$S_D = 1$$

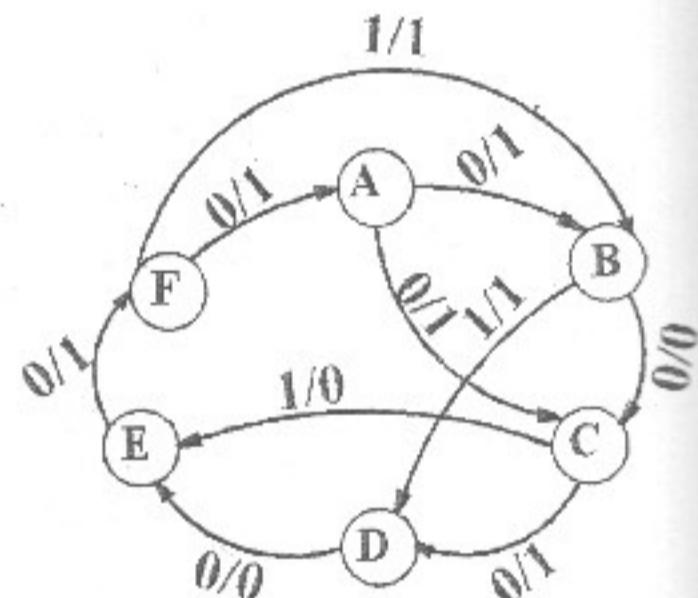
$$R_D = 0$$

۵- مدار مربوط به دیاگرام حالات زیر را به کمک T-F.F طراحی کنید.

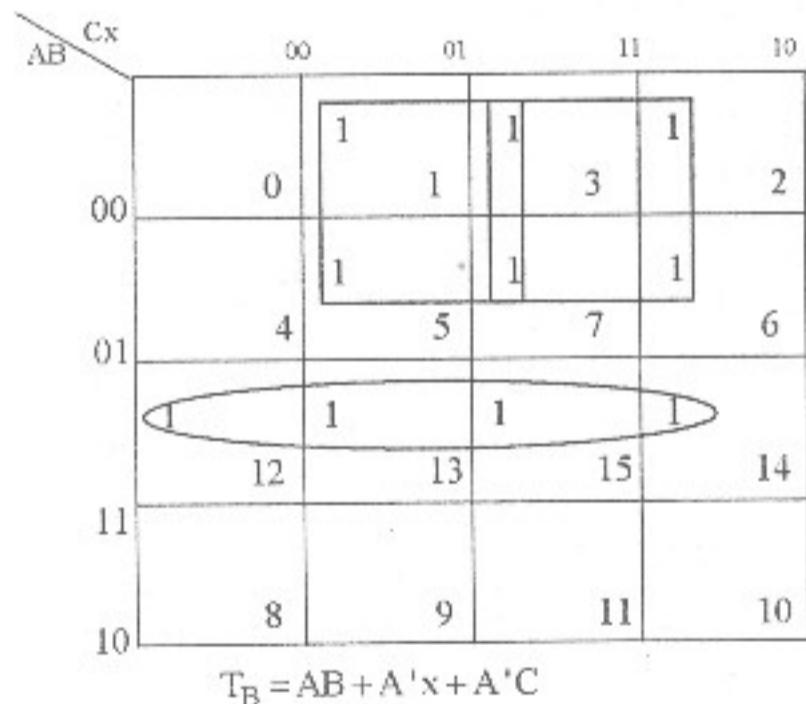
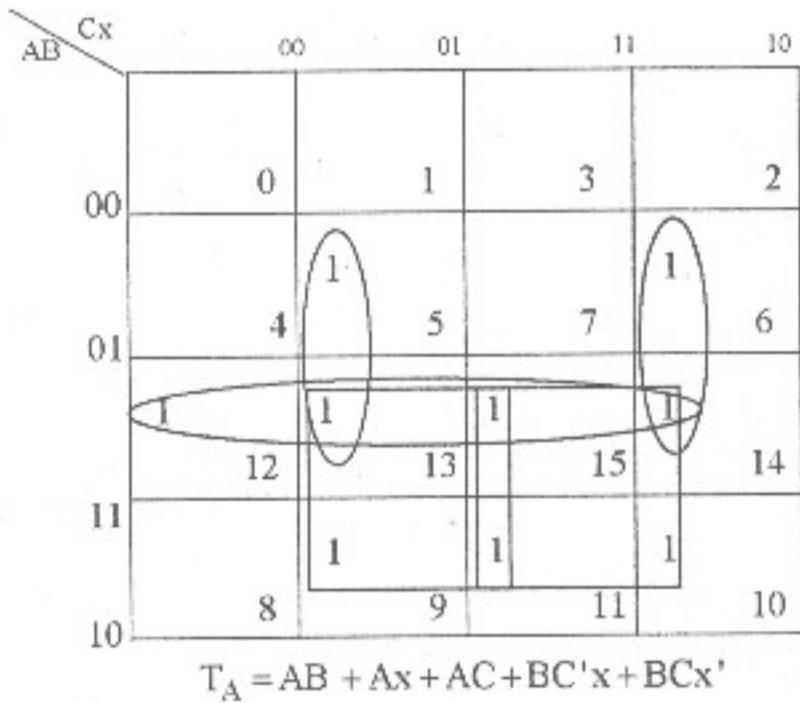
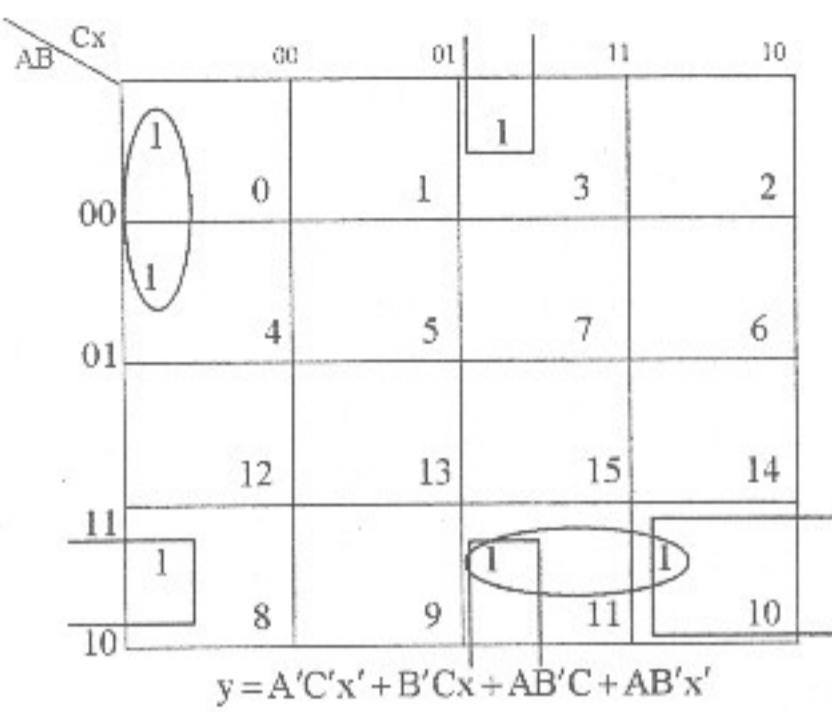
(حالاتی را که نداریم صفر می‌گیریم.)

A	B	C	x	A	B	C	y	T _A	T _B	T _C
0	0	0	0	0	0	1	1	0	0	1
0	0	0	1	0	1	0	0	0	1	0
0	0	1	0	0	1	0	0	0	1	1
0	0	1	1	0	1	1	1	0	1	0
0	1	0	0	0	1	1	1	0	0	1
A 000				0	1	0	1	1	1	0
B 001				0	1	1	0	1	1	1
C 010				0	1	1	1	0	1	1
D 011				1	0	0	0	0	0	1
E 100				1	0	0	1	0	0	0
F 101				1	0	1	0	0	0	1
				1	0	1	1	1	0	0
				1	1	0	0	1	1	0
				1	1	0	1	0	1	0
				1	1	1	0	0	1	1
				1	1	1	1	1	1	1

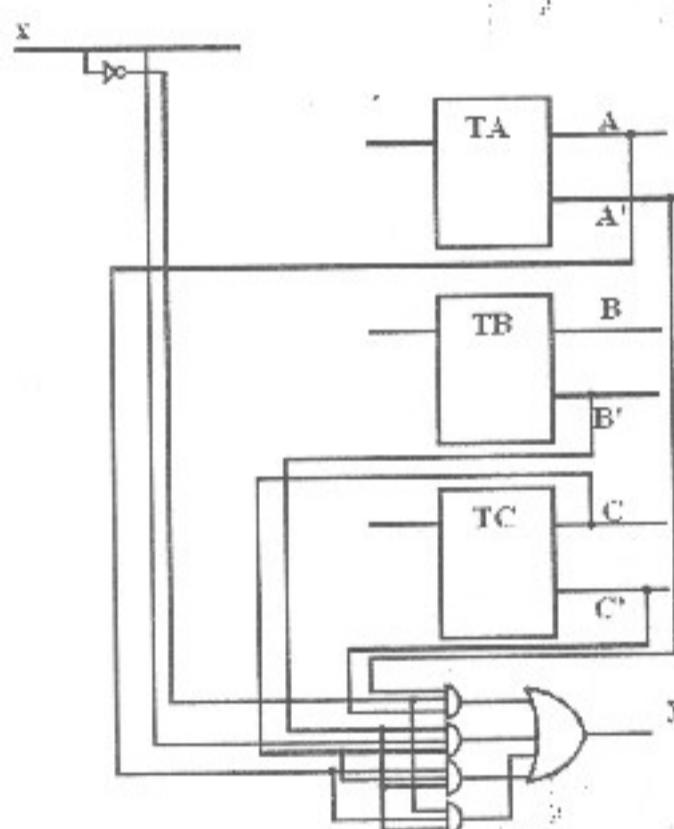
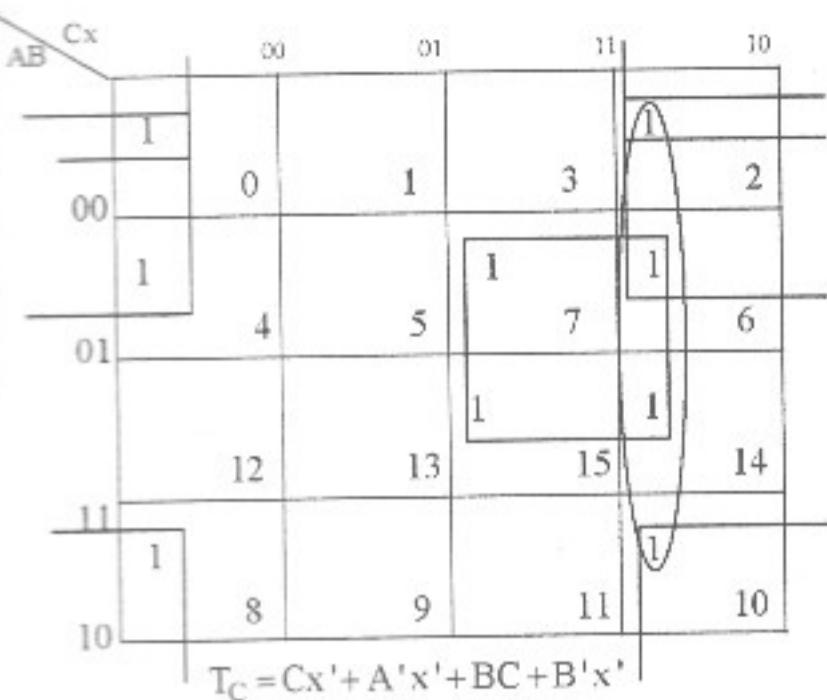
$$\lceil \log_2 6 \rceil = 3$$



مدار منطقی



مدار منطقی



مانند شکل با توجه به روابط بدست آمده از جدول کارنوها رسم شود.

۹- به کمک RS-F.F شمارندهای طراحی کنید که دنباله زیر را بشمارد.

$$\begin{matrix} 0 \rightarrow 6 \rightarrow 2 \rightarrow 4 \\ \uparrow \\ 3 \leftarrow 5 \leftarrow 1 \leftarrow 7 \end{matrix}$$

$$\left[20g \frac{8}{2} \right] = 3 \text{ تعداد بیت‌ها}$$

A	B	C	A	B	C	S _A	R _A	S _B	R _B	S _C	R _C
0	0	0	1	1	0	1	0	1	0	0	X
0	0	1	1	0	1	1	0	0	X	0	0
0	1	0	1	0	0	1	0	0	1	0	X
0	1	1	0	0	0	0	X	0	1	0	1
1	0	0	1	1	1	X	0	1	0	1	0
1	0	1	0	1	1	0	1	1	0	X	0
1	1	0	0	1	0	0	1	X	0	0	X
1	1	1	0	0	1	0	1	0	1	X	0

مدار منطقی

A	BC	00	01	11	10
0		1			1
1	X				

$$S_A = A'B' + A'C'$$

A	BC	00	01	11	10
0				X	1
1	1		1		1

$$R_A = AC + B$$

A	BC	00	01	11	10
0		1			
1	1		1		X

$$S_B = AB' + B'C'$$

A	BC	00	01	11	10
0			X		
1				1	1

$$R_B = BC + A'B$$

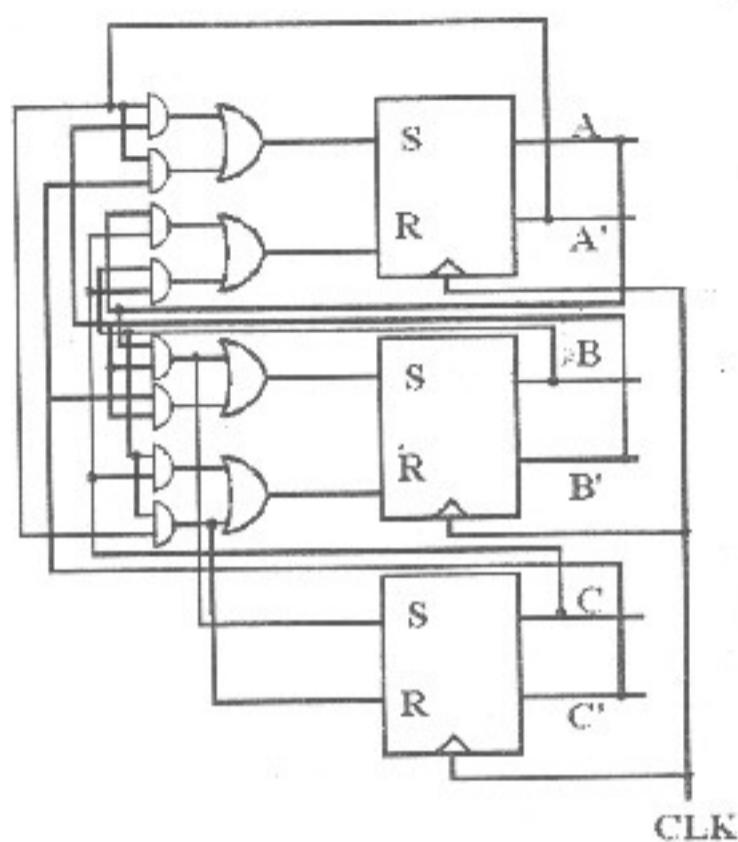
A	BC	00	01	11	10
0			X		
1	1		X	X	

$$S_C = AB'$$

A	BC	00	01	11	10
0				1	X
1	X				

$$R_C = A'B$$

مدار منطقی



۷- به کمک JK.F.F طراحی کنید که ورودی آن یک عدد سه بیتی باشد و خروجی اش معادل Gray ورودی.

حالت فعلی			حالت بعدی			ورودی های F.F					
A _i	B	C	A _{i+1}	B	C	j _A	k _A	j _B	k _B	j _C	k _C
0	0	0	0	0	0	0	X	0	X	0	X
0	0	1	0	0	1	0	X	0	X	X	0
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	0	1	0	0	X	X	0	X	1
1	0	0	1	1	0	X	0	1	X	0	X
1	0	1	1	1	1	X	0	1	X	X	0
1	1	0	1	0	1	X	0	X	1	1	X
1	1	1	1	0	0	X	0	X	1	X	1

BC00		01	11	10			
A	0	0	1	X	3	X	2
1	1	4	5	X	7	X	6

$$j_B = A$$

$$k_B = A$$

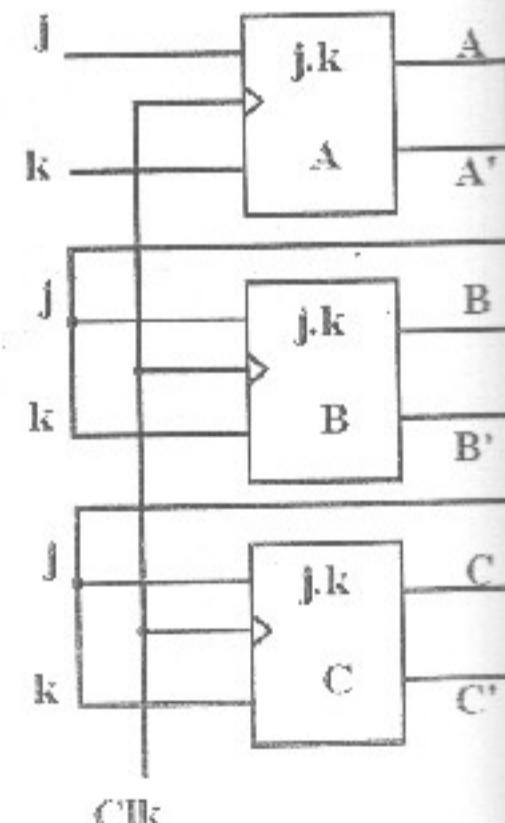
BC		01	H	10	
A	0	X	X	1	2
1	X	0	3	1	2

$$j_C = B$$

$$k_C = B$$

BC00		01	11	10	
A	0	X	1	3	2
1	X	0	1	3	2

BC00		01	H	10
A	0	X	1	3
1	X	0	1	3



مدار منطقی

- به کمک $j_B = C + D$ مداری طراحی کنید که ورودی آن یک عدد 4 بیتی باشد و خروجی اش مکمل 2 ورودی باشد.

A_t	B	C	D	A	B	C	D	j_A	k_A	j_B	k_B	j_C	k_C	j_D	k_D
0	0	0	0	0	0	0	0	0	X	0	X	0	X	0	X
0	0	0	1	1	1	1	1	1	X	1	X	1	X	X	0
0	0	1	0	1	1	1	0	1	X	1	X	X	0	0	X
0	0	1	1	1	1	0	1	1	X	1	X	X	1	X	0
0	1	0	0	1	1	0	0	1	X	X	0	0	X	0	X
0	1	0	1	1	0	1	1	1	X	X	1	1	X	X	0
0	1	1	0	1	0	1	0	1	X	X	1	X	0	0	X
0	1	1	1	1	0	0	1	1	X	X	1	X	1	X	0
1	0	0	0	1	0	0	0	X	0	0	X	0	X	0	X
1	0	0	1	0	1	1	1	X	1	1	X	1	X	X	0
1	0	1	0	0	1	1	0	X	1	1	X	X	0	0	X
1	0	1	1	0	1	0	1	X	1	1	X	X	1	X	0
1	1	0	0	0	1	0	0	X	1	X	0	0	X	0	X
1	1	0	1	0	0	1	1	X	1	X	1	1	X	X	0
1	1	1	0	0	0	1	0	X	1	X	1	X	0	0	X
1	1	1	1	0	0	0	1	X	1	X	1	X	1	X	0

	CD				00	01	11	10
AB	00	01	11	10				
00	0	1	1	2				
01	X	X	X	X	4	5	7	6
11	X	X	X	X	12	13	15	14
10	8	9	11	10				

$$j_B = C + D$$

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

$$k_B = C + D$$

	CD				00	01	11	10
AB	00	01	11	10				
00	0	1	X	X	2			
01	1	5	X	X	4	5	7	6
11	12	13	X	X	15	14		
10	8	9	11	10				

$$j_C = D$$

$$j_D = 0$$

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

$$k_C = D$$

$$k_D = 0$$

مدار منطقی

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

$$K_A = AB + C + D$$

		CD			
		00	01	11	10
AB	J_A	1	1	1	1
	0	1	1	3	2
AB	J_A	1	1	1	1
	4	5	7	6	
AB	X	X	X	X	X
	12	13	15	14	
AB	X	X	X	X	X
	8	9	11	10	

$$J_A = A'B + C + D$$

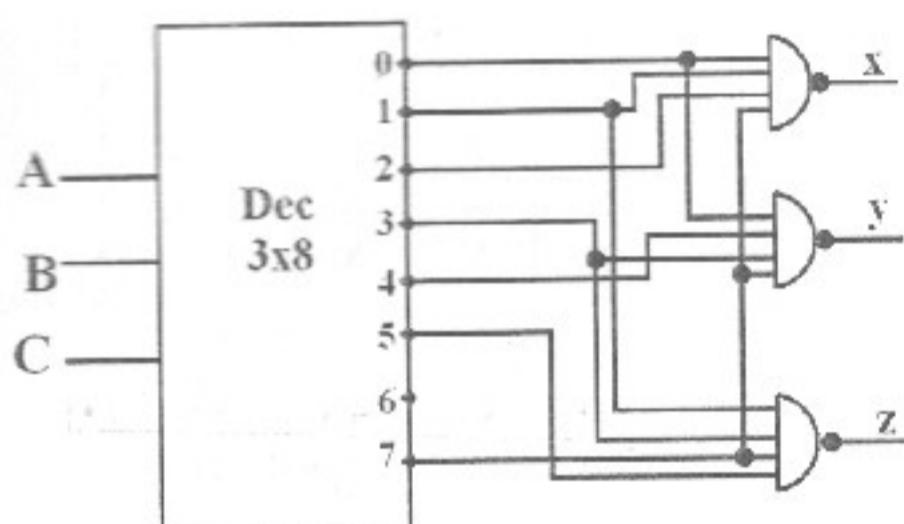
۹ - با توجه به روابط زیر به کمک R.S.F.F مدار را طراحی کنید. (X ورودی و y خروجی)

$$y = x'(A+B)$$

$$A_{t+1} = Ax + Bx$$

$$B_{t+1} = A'x$$

۱۰ - دیاگرام حالات مربوط به مثال فرق را رسم کنید.



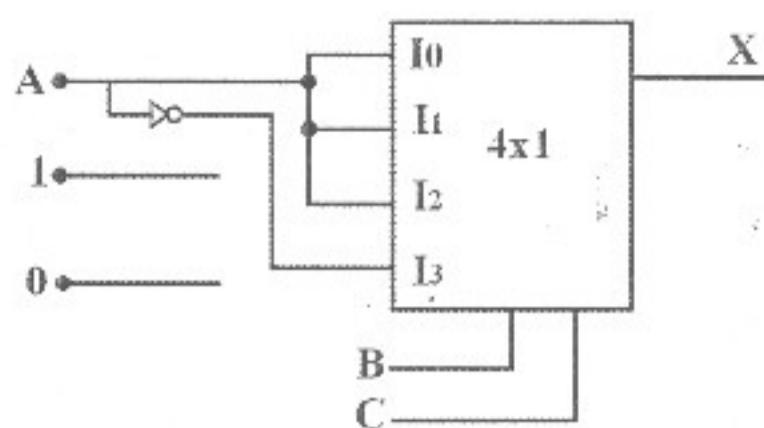
$$\text{تعداد حالت} = 2^8$$

$$n =$$

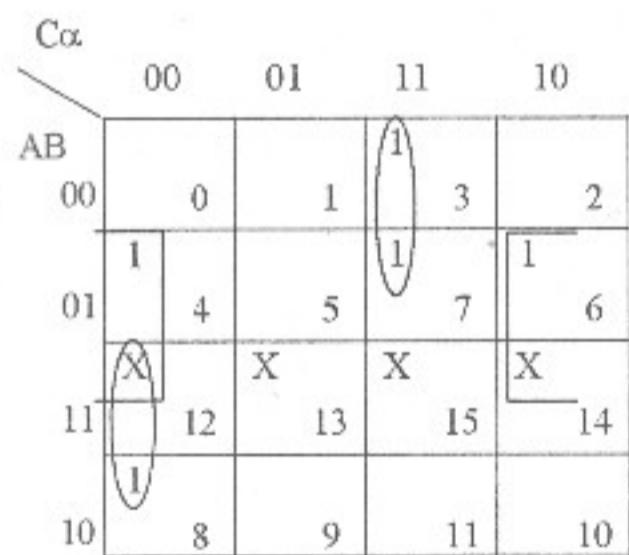
$$\left[\log_2 \right] = \text{تعداد فلیپ فلاب ها}$$

مدار منطقی

۱۱- با توجه به دیاگرام حالات زیر مدار را به کمک K.F.F طراحی کنید.



A	B	C	α	A	B	C	B	j_A	k_A	j_B	k_B	j_C	k_C
0	0	0	0	0	0	1	0	0	X	0	X	1	X
0	0	0	1	1	0	1	0	1	X	0	X	1	X
0	0	1	0	0	1	0	0	0	X	1	X	X	1
0	0	1	1	1	0	0	1	1	X	0	X	X	1
0	1	0	0	0	1	1	1	0	X	X	0	1	X
0	1	0	1	0	0	0	0	0	X	X	1	0	X
0	1	1	0	1	0	0	1	1	X	X	1	X	1
0	1	1	1	0	0	0	1	0	X	X	1	X	0
1	0	0	0	1	0	1	1	X	0	0	X	1	X
1	0	0	1	0	1	1	0	X	1	1	X	1	X
1	0	1	0	0	0	0	0	X	1	0	X	X	1
1	0	1	1	1	0	0	0	X	0	0	X	X	1
1	1	0	0	X	X	X	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X	X	X	X



$$B = B\alpha' + A'C\alpha + AC'\alpha'$$

مدار منطقی

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

$$j_A = B' \alpha + BC\alpha'$$

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

$$k_A = C' \alpha + C\alpha'$$

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

$$j_B = A' C\alpha' + AC'\alpha$$

$$j_C = B' C' + C'\alpha'$$

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

$$k_B = C + \alpha$$

$$k_C = B' C + C\alpha'$$

مدار منطقی

- ۱۲- یک شمارنده ۴ بیتی به کمک JK , F , F طراحی کنید که تنها اعداد زوج را به شمارد (راهنمایی: اگر حالت فرد بودند صفر را بشمارند) (شمارنده‌ای که مرتب است).

A	B	C	D	A	B	C	D	j_A	k_A	j_B	k_B	j_C	k_C	j_D	k_D
0	0	0	0	0	0	1	0	0	X	0	X	1	X	0	X
0	0	0	1	0	0	0	0	0	X	0	X	0	X	X	1
0	0	1	0	0	1	0	0	0	X	1	X	X	1	0	X
0	0	1	1	0	0	0	0	0	X	0	X	X	1	X	1
0	1	0	0	0	1	1	0	0	X	X	0	1	X	0	X
0	1	0	1	0	0	0	0	0	X	X	1	0	X	X	1
0	1	1	0	1	0	0	0	1	X	X	1	X	1	0	X
0	1	1	1	0	0	0	0	0	X	X	1	X	1	X	1
1	0	0	0	1	0	1	0	X	0	0	X	1	X	0	X
1	0	0	1	0	0	0	0	X	1	0	X	0	X	X	1
1	0	1	0	1	1	0	0	X	0	1	X	X	1	0	X
1	0	1	1	0	0	0	0	X	1	0	X	X	1	X	1
1	1	0	0	1	1	1	0	X	0	X	0	1	X	0	X
1	1	0	1	0	0	0	0	X	1	X	1	0	X	X	1
1	1	1	0	0	0	0	0	X	1	X	1	X	1	0	X
1	1	1	1	0	0	0	0	X	1	X	1	X	1	X	1

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

$$k_A = D + BC$$

مدار منطقی

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

$$J_A = BCD'$$

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

$$k_A = D + BC$$

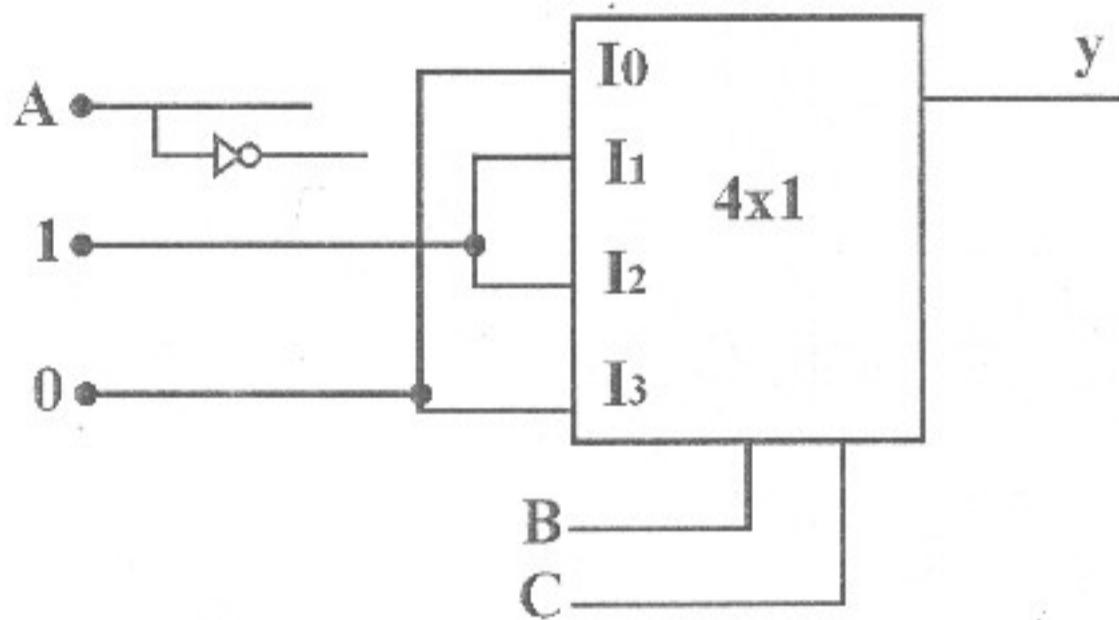
AB	CD	00	01	11	10	
		X	X	X	X	
		0	1	3	2	
		00				1
		4	5	7		6
			1	1		1
		12	13	15		14
		X	X	X	X	
		8	9	11		10

$$k_B = D + C$$

$\bar{A}B$	CD	00	01	11	10
	1	0	1	X	X
00	1			3	2
	4	5		X	X
01	1			7	6
	12	13		X	X
11	1			15	14
	8	9		X	X
10				11	10

$$J_C = D'$$

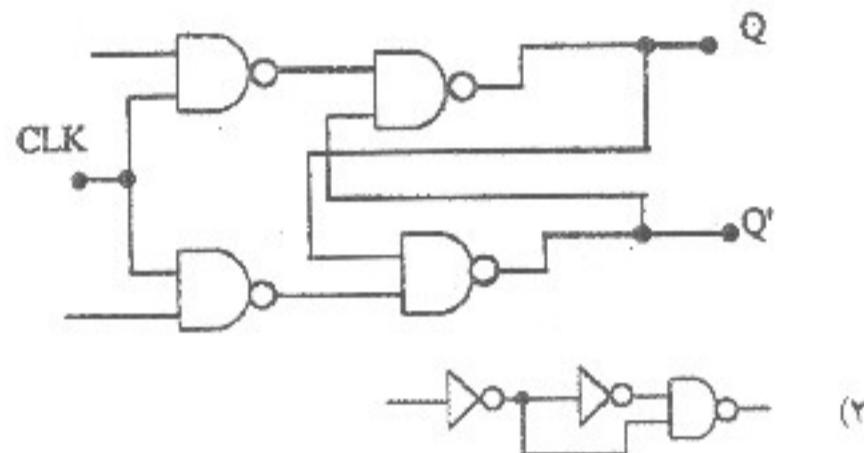
$$K_C = 1 \quad J_D = 0 \quad K_D = 1$$



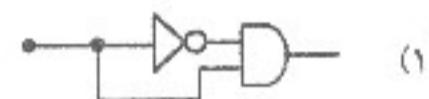
مدار منطقی

مجموعه تست‌های فصل چهارم:

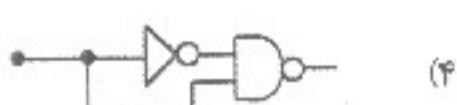
- ۱- در شکل مقابل یک فلیپ فلاب RS ساعت دار ملاحظه می‌کنید. برای آنکه این فلیپ فلاب روی لبه پایین رونده پالس ساعت عمل کند، کدام یک از مدارهای زیر باید در مسیر سیگنال ساعت، در ورودی فلیپ فلاب قرار گیرد؟



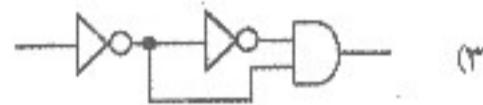
(۲)



(۱)

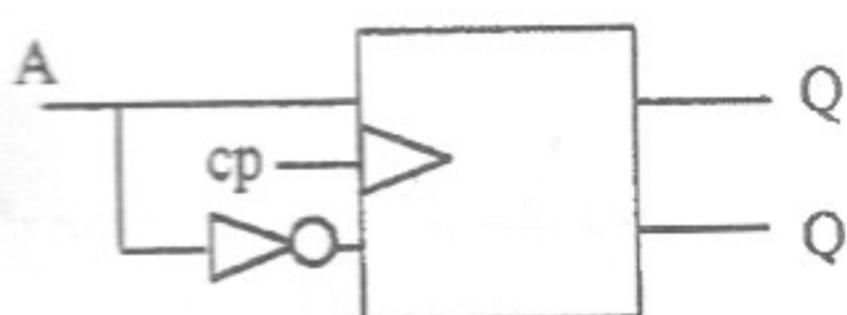


(۲)



(۳)

- ۲- فلیپ فلاب شکل مقابل با کدام لبه پالس فعال شده و از کدام نوع است؟



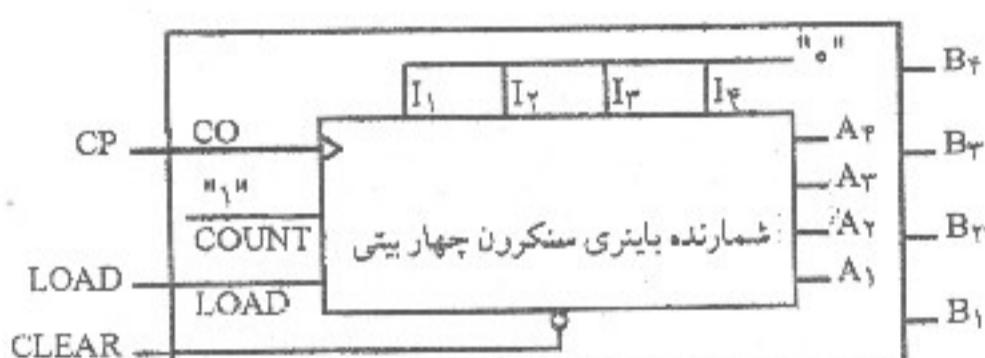
(۱) پایین رونده - D

(۲) پایین رونده - T

(۳) بالا رونده - D

(۴) بالا رونده - T

- ۳- برای آنکه خروجی B_1, B_2, B_3, B_4 مدار زیر اعداد زوج بین صفر و پانزده را مکرراً بشمارد لازم است که:



۱- $B_1 = A_1, B_2 = A_2, B_3 = A_3, B_4 = A_4$, CLEAR = \bar{A}_4 , LOAD = "0". قرار داده شود.

۲- $B_1 = 0, B_2 = A_1, B_3 = A_2, B_4 = A_3$, CLEAR = A_4 , LOAD = "0". قرار داده شود.

۳- $B_1 = 0, B_2 = A_1, B_3 = A_2, B_4 = A_3$, clear = \bar{A}_4 , load = "0". قرار داده شود.

۴- $B_1 = 0, B_2 = A_1, B_3 = A_2, B_4 = A_3$, clear = \bar{A}_4 , load = "1". قرار داده شود.

۴- معادلات ورودی فلپ‌فلاپ T از یک شمارنده نزولی با پسی سنکرون 4 بیتی را بنویسید. (کوچکترین بیت LSB شمارنده Q_A و بیت‌های دیگر به ترتیب Q_B و Q_C و Q_D می‌باشد).

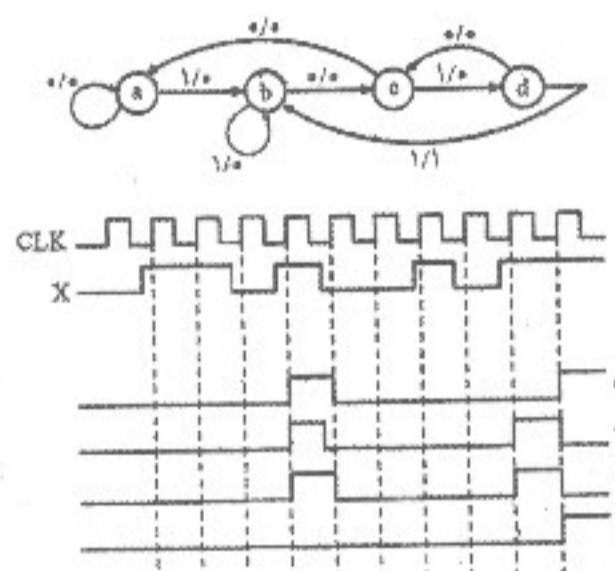
$$T_A = 1 \quad T_B = \bar{Q}_A \quad T_C = \bar{Q}_A \bar{Q}_B \quad T_D = \bar{Q}_A \bar{Q}_B \bar{Q}_C \quad (1)$$

$$T_A = 1 \quad T_B = Q_A \quad T_C = Q_A Q_B \quad T_D = Q_A Q_B Q_C \quad (2)$$

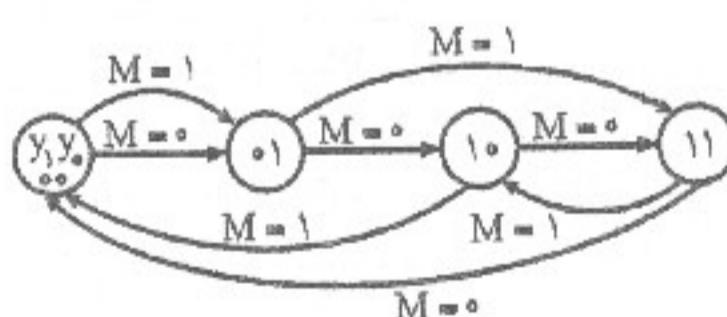
$$T_A = 1 \quad T_B = \bar{Q}_A \quad T_C = Q_A \cdot Q_B \quad T_D = \bar{Q}_A \cdot Q_B \cdot \bar{Q}_C \quad (3)$$

$$T_A = T_B = 1, T_C = \bar{Q}_A, \quad T_D = \bar{Q}_A \cdot \bar{Q}_C \quad (4)$$

۵- در دیاگرام (State Diagram) زیر کدام خروجی درست است؟ وضعیت اولیه سیستم حالت a می‌باشد.



۶- می‌خواهیم یک شمارنده دو بیتی همگام طرح کنیم که با توجه به مقدار ورودی کنترلی M در کد باینری و یا گری (Gary) بشمارد. نمودار حالت در شکل زیر داده شده است. اگر از دو فلپ‌فلاپ K.J استفاده کنیم، مقادیر ورودی فلپ‌فلاپ‌ها را بدست آورید. () و K_1 ورودی‌های فلپ‌فلاپ با ارزش بیشتر هستند)



$$J_1 = y_0, K_1 = M \oplus y_0, J_0 = M' + y'_1, K_0 = M' + y_1 \quad (1)$$

$$J_1 = M' + y'_1, K_1 = M' + y_1, J_0 = y_0, K_0 = M \oplus y_1 \quad (2)$$

$$J_1 = y'_0, K_1 = M y'_0 + M' y_0, J_0 = M + y_1, K_0 = M \oplus y_0 \quad (3)$$

$$J_1 = y_0, K_1 = 1, J_0 = M' y'_0 + M y'_1, K_0 = M' \oplus y_1 \quad (4)$$

مدار منطقی

۷- اگر با استفاده از یک فلیپ‌فلاب T بخواهیم مداری طرح کیم که رفتار یک فلیپ‌فلاب K - J را داشته باشد ورودی فلیپ‌فلاب توسط کدام یک از توابع زیر قابل بیان است؟

$$T = J\bar{O} + \bar{K}Q \quad (2)$$

$$T = J = K \quad (1)$$

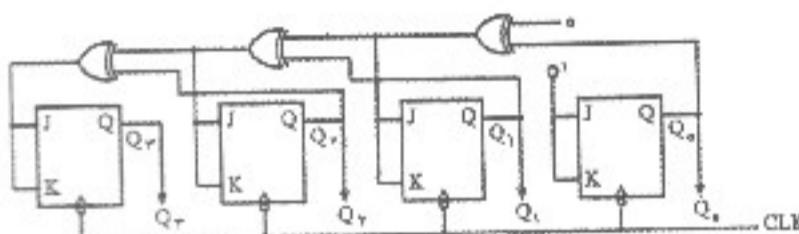
(۳) با فلیپ‌فلاب T نمی‌توان یک فلیپ‌فلاب K - J طرح کرد

$$T = J\bar{Q} + KQ \quad (3)$$

۸- جدول حالت مقابله را با دو فلیپ‌فلاب K - J طراحی می‌کنیم. ورودی فلیپ‌فلاب‌ها را تعیین کنید. (حالت بعدی به ازای دو مقدار مسکن ورودی X داده شده است و $y_2 y_1$ خروجی فلیپ‌فلاب‌ها هستند.)

$y_2 y_1$ حالت بعدی حالت فعلی $(y_2^+ y_1^+)$

$y_2 y_1$	$x = 0$	$x = 1$
00	00	01
01	10	01
10	10	11
11	11	00



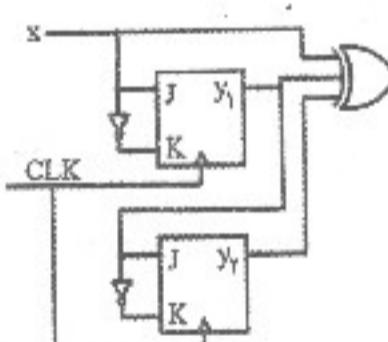
$$J_2 = 1, K_2 = y_1, J_1 = y_1, K_1 = 1 \quad (3)$$

$$J_2 = \bar{x}K_2 = y_1x_1, J_1 = 1, K_1 = y_2 \oplus x \quad (1)$$

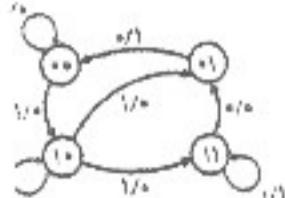
$$J_2 = y_1\bar{x}, K_2 = y_2y_1, J_1 = x, K_1 = y_2\bar{x} \quad (4)$$

$$J_2 = y_1\bar{x}, K_2 = y_1x, J_1 = x, K_1 = \overline{y_2 \oplus x} \quad (3)$$

۹- کدام گزینه دیاگرام حالات مدار زیر را بیان می‌کند؟



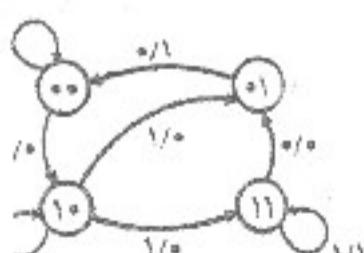
(۳)



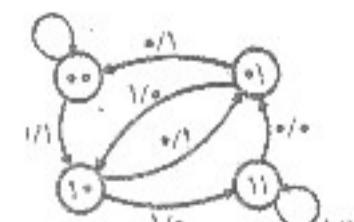
(۱)



(۲)

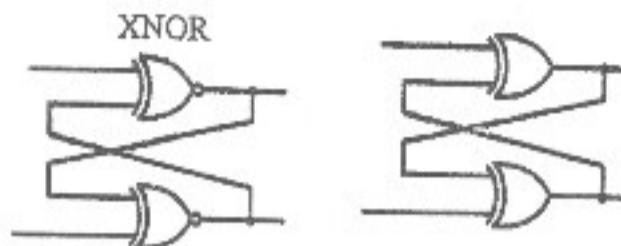


(۴)



مدار منطقی

۱۰ - از دو شکل زیر بگوید کدام یک رامی توان به جای SR - Latch استفاده نمود.



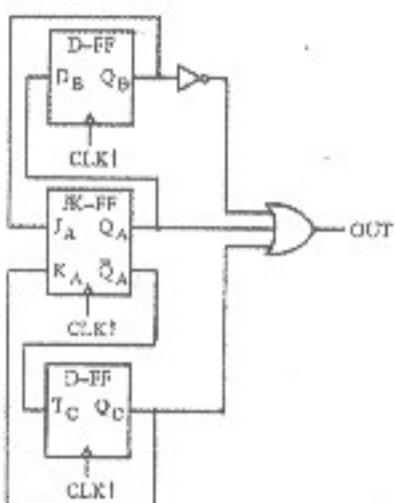
(۱) مدار شکل الف

(۲) مدار شکل ب

(۳) هر دو مدار شکل الف و ب

(۴) هیچ کدام از مدارها الف و ب

۱۱ - مدار مقابل از سه فلیپ فلاب D, JK و T تشکیل شده است اگر داشته باشیم در آغاز شروع کار $Q_A = 1$, $Q_B = 0$ و $Q_c = 1$ پس از دو پالس ساعت وضعیت خروجی فلیپ فلاب ها و خروجی نهایی را معین سازید.



$$Y=1, Q_c=1, Q_B=1, Q_A=0 \quad (۲)$$

$$Y=1, Q_c=1, Q_B=1, Q_A=1 \quad (۱)$$

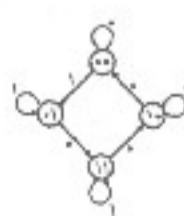
$$Y=1, Q_c=1, Q_B=0, Q_A=1 \quad (۴)$$

$$Y=1, Q_c=0, Q_B=0, Q_A=1 \quad (۳)$$

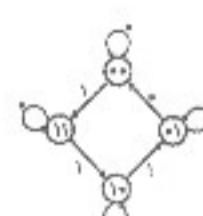
۱۲ - یک مدار ترتیبی سنکرون شامل دو فلیپ فلاب k-j به نام های A و B است و معادلات ورودی فلیپ فلاب ها در زیر نوشته شده است

$$\begin{cases} j_A = B\bar{x} \\ k_A = Bx \\ j_B = x \\ k_B = A \odot x \end{cases}$$

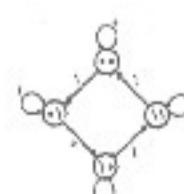
دیگر اگر حالت این مدار ترتیبی چیست؟



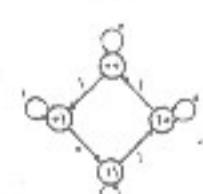
(۳)



(۱)



(۴)



(۲)

مدار منطقی

۱۳ - با یک فلیپ فلاپ D می خواهیم رفتار یک فلیپ فلاپ T را پیاده سازی کیم. در این صورت:

$$D = \bar{T}Q \quad (4)$$

$$D = T\bar{Q} \quad (3)$$

$$D = T \odot Q \quad (2)$$

$$D = T \oplus Q \quad (1)$$

۱۴ - پس از کاهش حالت جدول مقابل چند حالت خواهد داشت.

۵ (۲)

۳ (۱)

6 (۴)

7 (۳)

حالت فعلی	$x=0$	$x=1$
S_1	$S_1/0$	$S_6/0$
S_2	$S_1/1$	$S_6/1$
S_3	$S_4/0$	$S_5/1$
S_4	$S_1/0$	$S_7/0$
S_5	$S_2/0$	$S_3/0$
S_6	$S_4/0$	$S_5/0$
S_7	$S_2/0$	$S_3/0$

۱۵ - کدام جمله در مورد فلیپ فلاپ های Master - slave صحیح است؟

(۱) حساس به لبه

(۲) دو فلیپ فلاپ آن به طور همزمان عمل می کنند.

(۴) هر سه گزینه صحیح است.

(۲) در زمان ۱ بودن clock عمل می کند.

۱۶ - مزیت فلیپ فلاپ JK over SR کدام است؟

(۱) تأخیر آن کمتر است.

(۲) از تعداد گیت های کمتری استفاده می کند.

(۳) برخلاف فلیپ فلاپ SR حالت نامعلوم ندارد.

(۴) مزیت خاصی ندارد.

۱۷ - اگر خروجی فلیپ فلاپ JK صفر باشد، پا اعمال یک پالس ساعت نیز به صفر برسد. ورودی های آن کدام است؟ (d) حالت بسی اهمیت است)

$$k = d, j = 0 \quad (4)$$

$$k = d, j = 1 \quad (3)$$

$$k = 1, j = d \quad (2)$$

$$k = 1, j = 0 \quad (1)$$

۱۸ - کدام یک از روابط زیر صحیح نیست؟

$$Q(t+1) = jQ' + k'Q \quad (1)$$

$$Q(t+1) = TQ' + T'Q \quad (2)$$

$$Q(t+1) = D \quad (3)$$

$$Q(t+1) = S + RQ' \quad (4)$$

مدار همنطقی

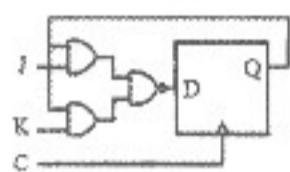
۱۹ - حلت حساس کردن فلیپ فلاب ها به لبه کدام است؟

- (۱) بالا بردن سرعت انتقال و کاهش تأخیر گیت ها
- (۲) کاهش تعداد ترانزیستورهای مصرفی برای ساخت مدار
- (۳) جلوگیری از تأثیر تغییرات ورودی ها بر روی خروجی در هنگام ۱ بودن پالس سرعت
- (۴) امکان استفاده از آنها در مدارات آسنکرون

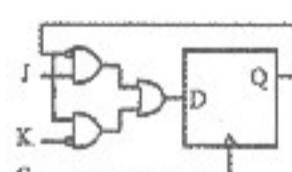
۲۰ - اگر ورودی یک فلیپ فلاب T برابر ۱ باشد، خروجی آن کدام است؟

- (۱) خروجی قبلی
- (۲) ورودی قبلی
- (۳) مکمل خروجی قبلی
- (۴) مکمل ورودی قبلی

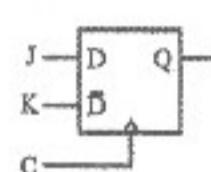
۲۱ - با استفاده از یک D - JK - FF بسازید.



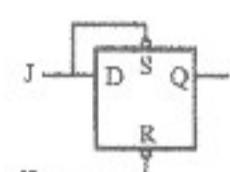
(۴)



(۳)



(۲)



(۱)

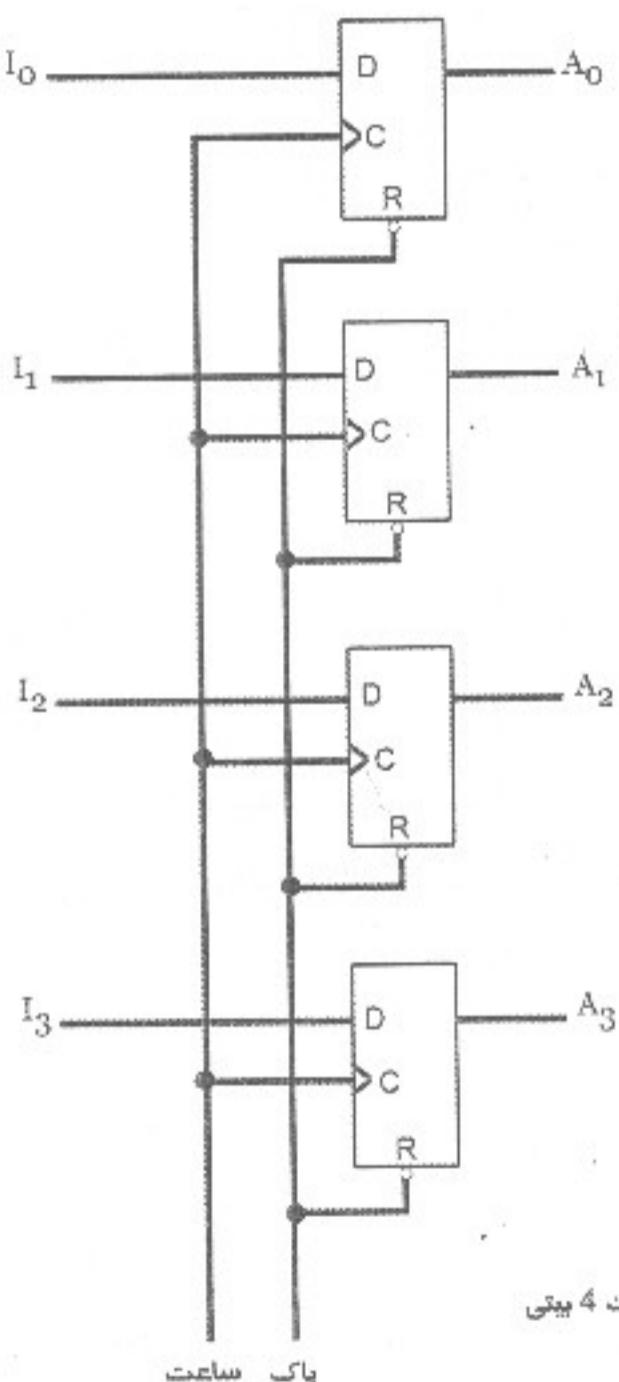
فصل پنجم

ثبات‌ها و شمارنده‌ها

ثبات‌ها یا رجیسترها:

ثبات‌ها مجموعه‌ای از خانه‌های حافظه هستند که هر خانه از یک فلیپ فلاب تشکیل شده و توانایی این را دارد که یک بیت اطلاعات دودویی را درون خود نگه دارد (درون یک فلیپ فلاب) یعنی یک ثبات n یعنی از n فلیپ فلاب تشکیل شده است که در مجموع توانایی ذخیره‌سازی n بیت را دارا می‌باشد ساده‌ترین ثبات فقط از تعدادی فلیپ فلاب تشکیل شده و در ساختمان آن هیچ گونه گیت خارجی وجود ندارد کلاک پالس توانایی فعال‌سازی فلیپ فلاب‌ها و در مجموع ثبات ساده را به خوده دارد.

معمول‌ترین فلیپ فلابی که در ساختن رجیستر مورد استفاده قرار می‌گیرد D فلیپ فلاب است یعنی Data فلیپ فلاب که به خودی خود کار یک بافر را انجام می‌دهد.



I₀ تا I₃ ورودی‌ها هستند clk و clear یا Reset هم خطوط مشترک هستند. از خط clear برای Reset کردن فلیپ فلاب‌ها یا پاک کردن مقادیر قبلی آن‌ها استفاده می‌شود. با صفر کردن خط Reset فلیپ فلاب‌ها پاک می‌شوند.

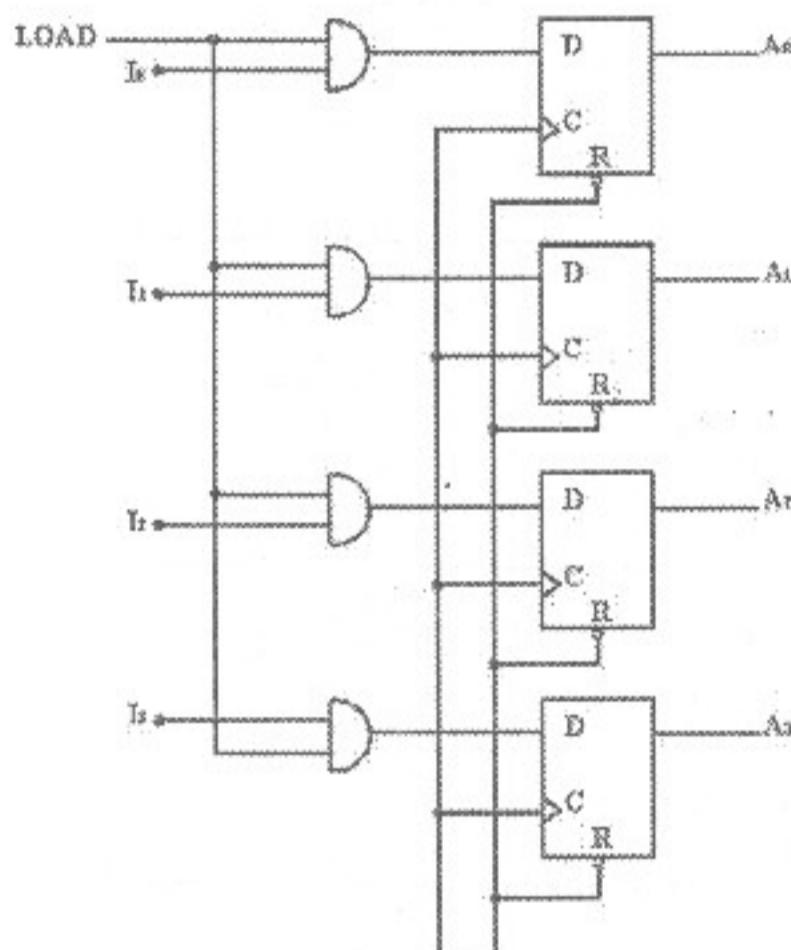


با یک شدن clk مقادیر ورودی I_0 تا I_3 وارد فلیپ فلابها می‌شوند و در خروجی قابل مشاهده می‌باشند در زمانی که clk صفر است مقادیر خروجی مرحله قبل داخل فلیپ فلابها باقی می‌مانند و می‌توان از آن‌ها استفاده کرد اگر تعداد فلیپ فلابها مورد استفاده n تا بود رجیستر به دست آمده n بیتی می‌باشد.

ثبات با امکان رشد موازی loading

تعاریف:

- بار شدن: انتقال اطلاعات جدید به داخل یک ثبات را بار شدن می‌گویند.
- بار شدن موازی: زمانی که همه بیت‌های یک ثبات همزمان به کمک یک clk مشترک یک بار شوند بار شدن موازی می‌گویند. شکل زیر یک نمونه بار شدن موازی است. که با اعمال یک به clk مقادیر I_0 تا I_3 به طور همزمان داخل فلیپ فلاب Load می‌شوند. اما با یک اشکال به این صورت که به علت تاخیری که ممکن است در هر کدام از فلیپ فلابها به وجود آید خروجی‌ها به صورت موازی باز نشوند لذا سیستم جدید را پیشنهاد می‌کنیم.



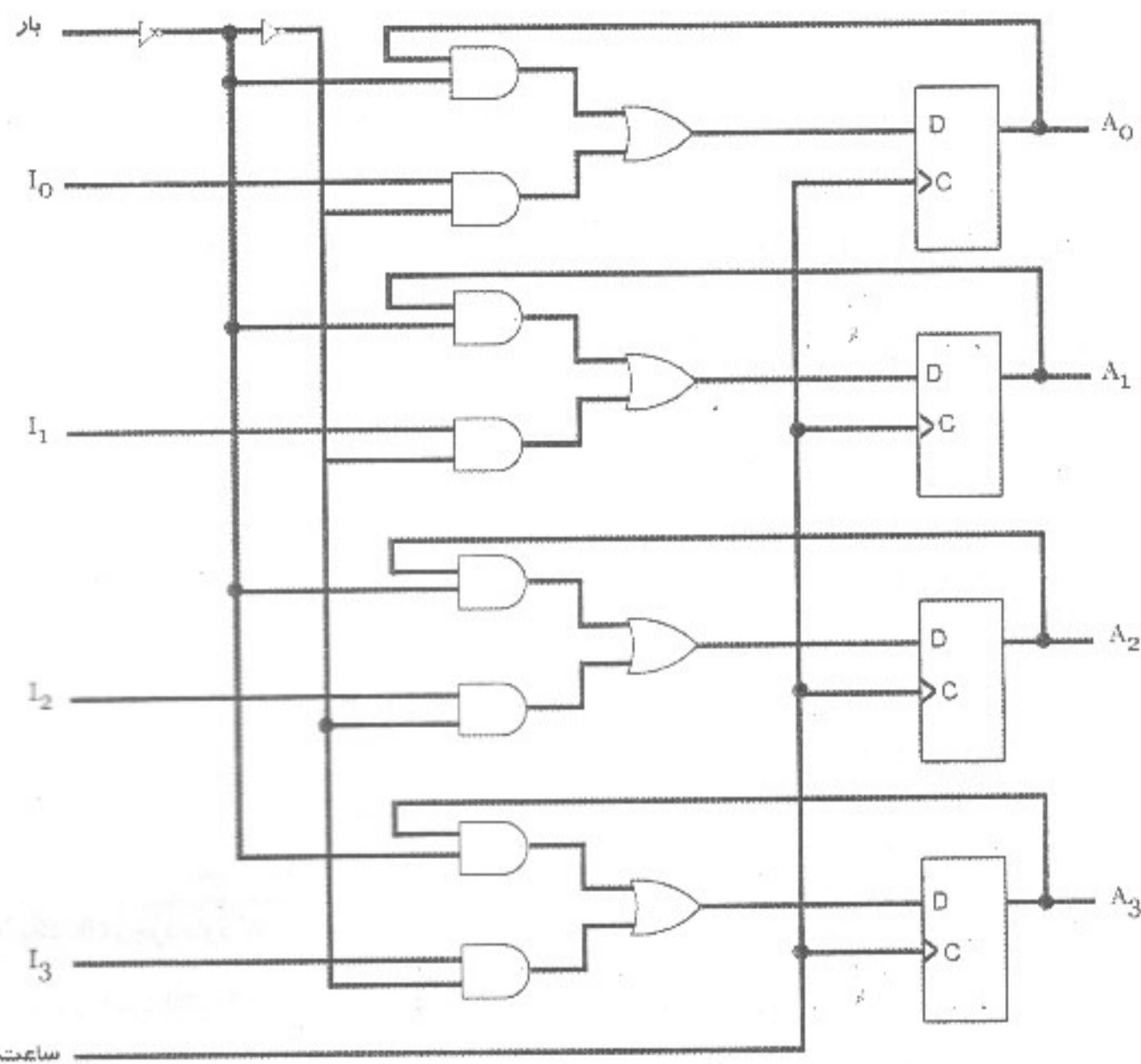
ورودی کنترل بار:

در واقع می‌خواهیم سیستم بار شدن موازی را پیاده سازی کنیم (با یک clk مشترک) اما برای اعمال ورودی به شکل همزمان به فلیپ فلابها از یک گیت AND (یا هر گیت کنترلی دیگر NOR, NAND, OR) برای کنترل بار استفاده می‌کنیم.

در خروجی همه فلیپ فلابها رجیستر خروجی یک AND وارد شده که ورودی clk AND یکی است و دیگری I_0 تا I_3 است.

هدار منطقی

شکل زیر بار شدن موازی با استفاده از گیت‌های کنترلی ورودی D فلیپ فلاب‌ها خروجی OR است ورودی OR خروجی دو AND است. ورودی AND‌ها یک بار از خط load و هم یک بار از متمم خط load به همراه خروجی مرحله قبل فلیپ فلاب با ورودی‌ها I_0 تا I_3 استفاده می‌کند.



ثبتات ۴ بیتی با بار شدن موازی

$$\left. \begin{array}{l} \text{load} = 0 \rightarrow A_3 A_2 A_1 A_0 \\ \text{load} = 1 \rightarrow I_3 I_2 I_1 I_0 \end{array} \right\} \text{خروجی قبل}$$

شیفت رجیستر

ثبتات یا رجیستری که بتواند اطلاعات داخل خود را به سمت راست یا چپ شیفت (انتقال) دهد شیفت رجیستر نامیده می‌شود.

ثبتات یا رجیستری که قابلیت انتقال اطلاعات درونش را به سمت چپ یا راست را داشته باشد شیفت رجیستر گفته می‌شود. Right shift Register اطلاعات درونش را به شیفت رجیستر معمولی این امکان را دارد که با هر بار عرض شدن clock یک شیفت انجام دهد. در این نوع شیفت رجیستر به ازای هر یک بیت اطلاعات که شیفت انجام می‌شود از یک سمت Register سمت راست ترین بیت خارج شده و از سمت چپ آن یک صفحه وارد می‌شود اما در شیفت رجیستر دورانی بیت خارج شده از سمت مخالف دوباره به داخل ثبات بر می‌گردد به این ترتیب در یک

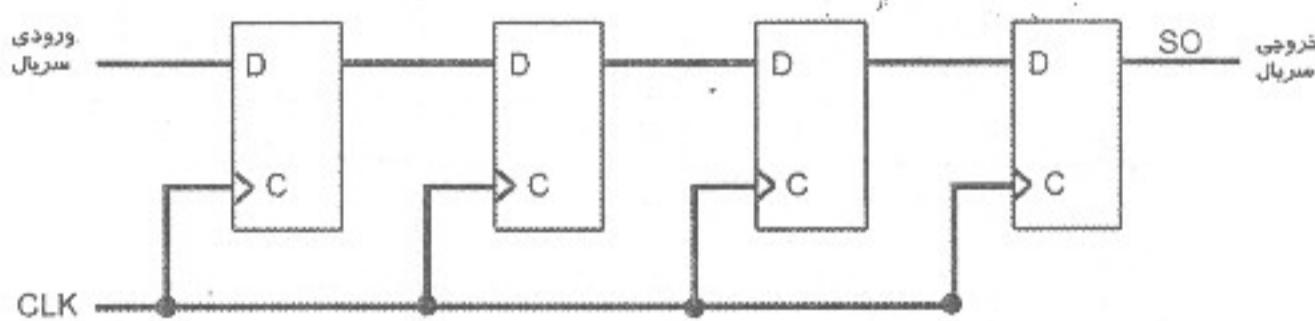
شیفت رجیستر n بیتی معمولی با n بار clock اطلاعات اصلی ثبات پاک شده و به جای آن صفر می‌نشیند اما در یک شیفت رجیستر دورانی n بیتی با n تا clock اطلاعات اولیه ثبات دوباره Load می‌شود در یک شیفت رجیستر معمولی یک بیت شیفت به چه (یعنی وارد شدن یک بیت صفر از سمت راست) معادل با ضرب کردن عدد داخل شیفت رجیستر در عدد 2 می‌باشد و در یک Right shift یک عمل شیفت دادن معادل تقسیم عدد شیفت رجیستر بر 2 می‌شود.

یازده

sh - R → 0	<table border="1"><tr><td>0</td><td>1</td><td>0</td><td>1</td></tr></table>	0	1	0	1	1	سمت راست	5
0	1	0	1					
sh - R → 1	<table border="1"><tr><td>1</td><td>1</td><td>0</td><td>1</td></tr></table>	1	1	0	1	1	سمت راست	13
1	1	0	1					
sh - L → 0	<table border="1"><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr></table>	0	1	1	0	1	سمت چپ	6
0	1	1	0					
sh - L → 1	<table border="1"><tr><td>0</td><td>1</td><td>1</td><td>1</td></tr></table>	0	1	1	1	1	سمت چپ	7
0	1	1	1					

ساختار یک شیفت و چیستر

زنگرهای از فلیپ فلامپ‌ها که خروجی یکی ورودی دیگری (بعدی) است



لیفٹ رجیسٹر 4 یعنی

در شکل بالا یک clk وجود دارد که برای همه فلیپ فلاب‌ها مشترک است در فلیپ فلاب اول یک ورودی SI وارد شده یا ورودی سریال وارد شده است. ورودی قنیپ فلاب اول به Q_1 تبدیل شده که همان ورودی فلیپ فلاب دوم است زیرا همین ترتیب خروجی هر کدام ورودی بعدی می‌شود و خروجی آخر SO یا خروجی سریال می‌شود.

اگر یک SI به شکل 1101 داشته باشیم و وارد شیفت رجیستر شود و فلیپ فلاپ‌ها را هم D_1, D_2, D_3, D_4 نام‌گذاری کنیم و در مرحله اول فلیپ فلاپ‌ها مقادیر دلخواه قبلی را دارد اگر Clear داشتیم یک بار Reset می‌کنیم که همه مقادیر صفر شود.

وقتی clk اول وارد شود به درون ثبات انتقال پیدامی کند یعنی ۱ وارد D_1 می‌شود و بقیه همان مقادیر قبلی را دارند وقتی clk دوم وارد شود صفر وارد D_1 می‌شود و ۱ به D_2 می‌رود و بقیه همان مقادیر قبل را دارند وقتی clk سوم وارد شود ۱ به D_1 صفر به D_2 و ۱ به D_3 می‌رود و D_4 مقدار قبلی خود را دارد وقتی clk چهارم را می‌زنیم کل مقادیر SI که ۱۱۰۱ است به ترتیب از چپ به راست در فلایپ فلاپ‌های D_1 تا D_4 وارد می‌شوند.

بدین ترتیب بعد از 4 اطلاعات وارد رجیستر شد حال عمل شیفت انجام می‌شود یک clk که زده شود 1 از D_4 خارج می‌شود خانه‌های D_1, D_2, D_3 اطلاعات خود را به خانه‌های بعدی می‌دهند و D_4 صفر می‌شود عدد 1 که از D_4 خارج می‌شود جایی ذخیره نمی‌شود اگر اطلاعات n بیتی را به وسیله یک شیفت رجیستر معمولی n بار شیفت دهیم مقادیر رجیسترها تماماً صفر می‌شود.

شیفت رجیستر دورانی

اگر اطلاعات 4 بیتی 0011 را داشته باشیم با استفاده از یک شیفت رجیستر دورانی بخواهیم آن را به راست شیفت دهیم به صورت زیر عمل می کنیم در یک شیفت رجیستر دورانی ۱۱ بیتی اگر ۱۱ بار شیفت دهیم به حالت اولیه می رسیم.

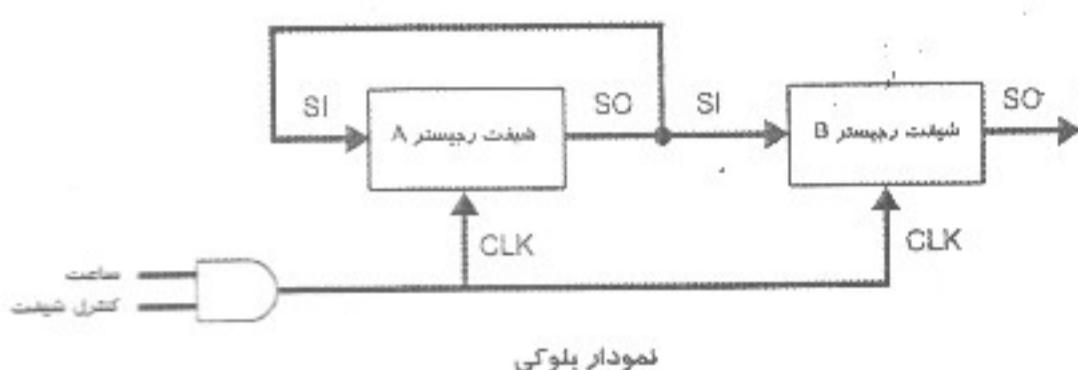
اگر بخواهیم شیفت رجیستر دورانی تولید شود از خروجی سریال به ورودی سریال وصل می کنیم.

0	0	1	1
1	0	0	1

انتقال سریال

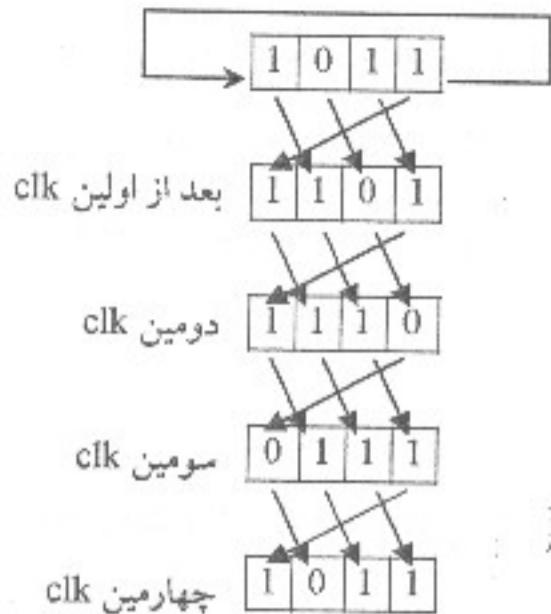
در صورتی که اگر اطلاعات به صورت سریال نباشد و موازی باشد به جای یک ورودی و یک خروجی ۴ ورودی و ۴ خروجی داریم برای اطلاعات 4 بیتی یک شیفت رجیستر اگر انتقال سریال انجام شود باید ۴ بار کلک زده شود ولی اگر انتقال موازی انجام شود با یک کلک می توان اطلاعات را منتقل کرد.

عملیات در سیستم های دیجیتال معمولاً به صورت موازی انجام می شود اما هم اکنون نیز در بعضی موارد و در بعضی سیستم ها از انتقال سریال استفاده می شود. سرعت عملیات موازی به مرتب بیشتر از سرعت عملیات سریال است اما مشکل عملیات موازی در پیچیده بودن سخت افزار و همچنین حجم بالای سخت افزار مورد نیاز است. در صورتی که در انتقال سریال از سخت افزار ساده تر و کم تر استفاده می شود.



الانتقال سریال از نیمات A به نیمات B

شیفت رجیستر راست (دورانی)



در شکل بالا دو شیفت رجیستر B,A داریم که می خواهیم به کمک انتقال سریال محتویات A را وارد B می کنیم که هر کدام از شیفت رجیسترها یک SI (ورودی سریال) و یک SO (سریال خروجی) دارند. یک کلاک مشترک هم دارد A مقدار 1011 را وارد B که مقدارش برای ما مهم نیست ($B = \dots\dots$) می کند.

نحوه کار: با اولین کلاک چون A برابر 1011 است راست ترین بیت اطلاعاتش را از طریق SO خود به SI انتقال می دهد و وارد سمت چپ ترین بیت B می شود بعد از یک کلاک A برابر 0101 خواهد بود و B هم مقدار $\dots\dots 1$ را خواهد داشت.

$$A = 1011 \rightarrow A = 0 = 0101$$

$$B = \dots\dots \rightarrow B = 1 \dots\dots$$

بعد از 4 کلاک اطلاعات A را به B منتقل می کنیم مقدار A=0000 و B=1011 خواهد بود و این حالت انتقال سریال انجام شده ولی محتویات A را از دست داده ایم برای این که مقدار A را از دست ندهیم می توانیم از یک شیفت رجیستر دورانی استفاده کنیم، یعنی از SO شیفت رجیستر A به A,SI وصل کنیم به این ترتیب بعد از 4 کلاک علاوه بر این که محتویات A را به B منتقل کرد هم A نیز مقدار قبلی خود را دارد.

اما این شیفت رجیستر یک مشکل دارد و آن این است که دائم درحال عملیات شیفت دادن هستیم زیرا کلاک پالس دائم فعال است. برای حل این مشکل از یک خط کنترل برای شیفت رجیستر استفاده می کنیم که شیفت رجیستر را کنترل کند. از یک AND استفاده می کنیم که یک ورودی آن خط clk است و ورودی دیگر خط کنترل شیفت است.

اگر کنترل شیفت 1 باشد، خط clk فعال است. اگر کنترل شیفت 0 باشد خط clk غیرفعال است و عمل شیفت انجام نمی شود.

چند اصطلاح:

(زمان کلمه): مدت زمانی که لازم است تا محتوی یک شیفت رجیستر به صورت کامل شیفت داده شود.

(bit time): مدت زمان بین دو کلاک پالس

انواع شیفت رجیسترها:

۱- SISO: ورودی و خروجی هر دو به صورت سریال می باشند.

(Serial Input , serial output)

۲- SIPO: ورودی به صورت سریال و خروجی به صورت موازی است.

(Serial Input , parallel output)

۳- PISO: ورودی به صورت موازی و خروجی به صورت سریال می باشد.

۴- PIPO: ورودی و خروجی هر دو به صورت موازی می باشند. (بهترین نوع شیفت رجیسترها)

شیفت رجیسترها دووجهه با امکان بار شدن موازی (شیفت رجیسترها یونیورسال)

از یک شیفت رجیستر می توانیم برای تبدیل داده سریال به موازی یا داده موازی به سریال استفاده کنیم فقط در این نوع شیفت رجیسترها باید به تمامی خروجی های فلیپ فلاب دسترسی داشته باشیم.

کامل ترین شیفت رجیستر دارای خصوصیات زیر است:

۱- کنترل پاک برای پاک کردن ثبات به (0)

۲- ورودی ساعت برای همزمانی اعمال

۳- کنترل همه جایی به راست (شیفت به راست) برای فعال کردن عمل جابه جایی به راست و خطوط ورودی و خروجی سریال مربوط به جابه جایی راست

۴- کنترل جابه جایی به چپ (شیفت به چپ) برای فعال کردن عمل جابه جایی به چپ و خطوط ورودی و خروجی سریال مربوط به جابه جایی به چپ

۵- کنترل باز کردن موازی برای فعال کردن انتقال موازی و ۶- خط ورودی مربوط به انتقال موازی

۶- خط خروجی

۷- حالت کنترلی که علی رضم وجود پالس ساعت اطلاعات را در ثبات بدون تغییر نگه می دارد.

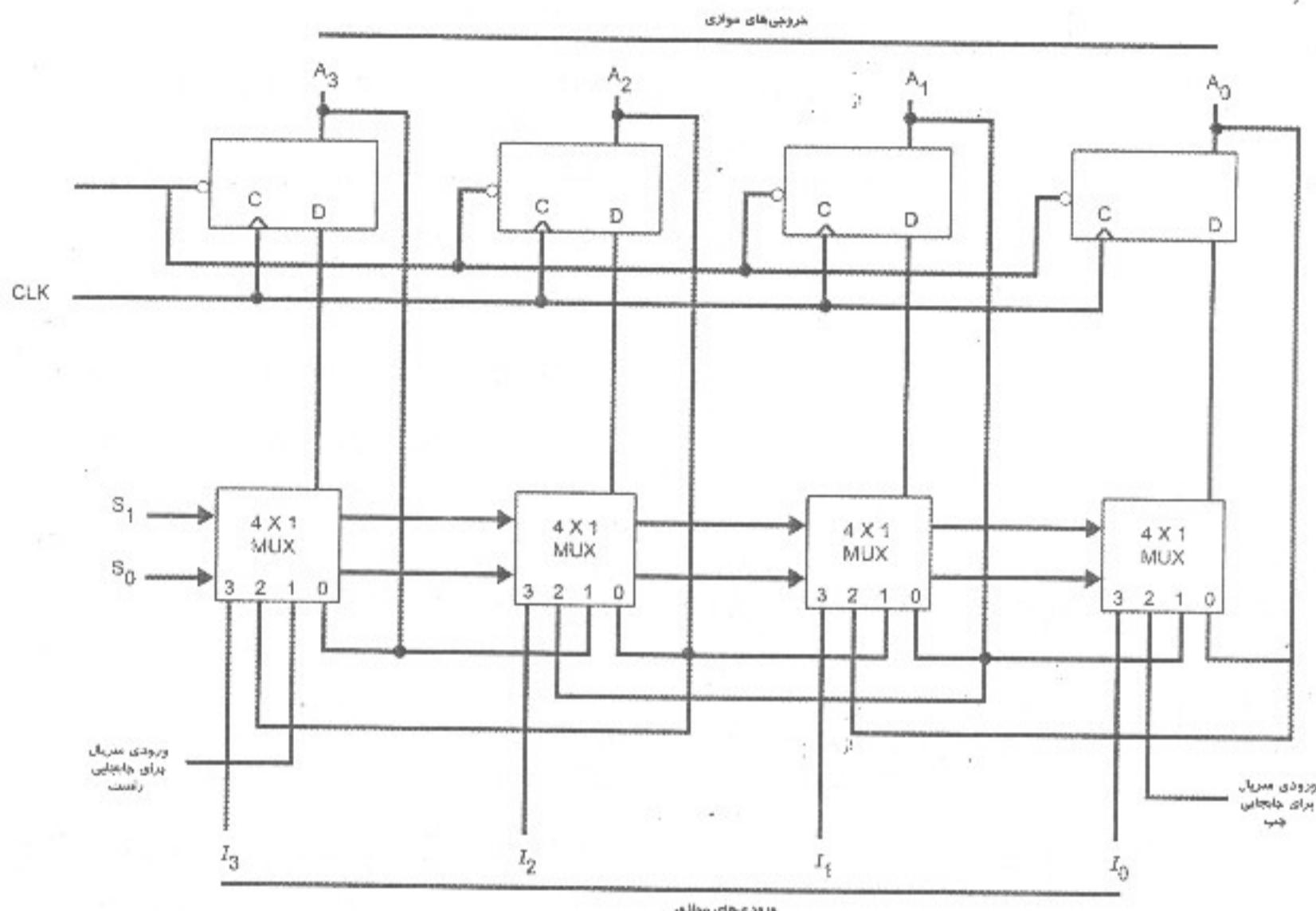
در شکل (۵-۵) ۴ مولتی پلکسر داریم (4×1) و ۲ خط select ۴, ۴ ورودی و خروجی خط های S_0, S_1, S_2, S_3 , select ۴ مولتی پلکسر را به هم وصل می کند، یعنی اگر یک select انجام دهیم و (0,0) به ورودی بدیم خط صفر برای همه فعال می شود.

همچنین در شکل بالا ۴ فلیپ فلاب داریم که از نوع D فلیپ فلاب هستند که یک پایه clear دارند و یک پایه clk دارند که در همه مشترک هستند.

خروجی مولتی پلکسر ورودی فلیپ فلاب است.

اگر S_0, S_1 هر دو صفر باشد (خط select (0,0) است) خط صفر فعال می شود (برای تمام مولتی پلکسرها فعال می شود) در این حالت خروجی صفر وارد D فلیپ فلاب می شود. یعنی یک بافر ساختیم که اطلاعات را از مولتی پلکسر به خروجی می برد و بعد باز به مولتی پلکسر باز می گرداند. (۷-۶ خروجی فلیپ فلاب که وارد خط صفر مولتی پلکسر شده)

اگر $(0,1)$ را در خط select فرار دهیم، خط 1 فعال می‌شود شیفت به راست داده می‌شود حال اگر مقدار $(1,0)$ به خط select بدهیم شیفت به چپ داده می‌شود و خط 2 فعال می‌شود اگر A_1 به پایه 2 مولتی پلکسر 3 وصل می‌شود شیفت رجیستر دورانی خواهد شد اگر $(1,1)$ شیفت به خط 3 فعال می‌شود در این حالت ورودی‌های موازی وارد می‌شود خروجی‌های موازی خارج می‌شود در این حالت را به خطوط select بدهیم خط 3 فعال می‌شود در این حالت ورودی‌های موازی وارد می‌شود خروجی‌های موازی خارج می‌شود در این حالت شیفت رجیستر با پار شدن موازی می‌گویند.



جیسا، یعنی کوئی دشمنت، جست یا توهجه به خلط و خلاط

S_1	S_0	عملکرد
0	0	بلا تغییر
0	1	شیفت راست
1	0	شیفت چپ
1	1	بار کردن موازی

مثال: محتویات شیفت رجیستر 16 بیتی A برابر $A\oplus B^2$ می‌باشد پس از 3 شیفت به راست با ورودی یک و دو شیفت به چپ حاصل آن را در

شیفت و جست B قرار می‌دهیم محتویات B چیست؟

12 بیتی	1010, 0000, 1011, 0010
3 راست	1111, 0100, 0001, 0110
2 چپ	1101, 0000, 0101, 1000

جمع سریال

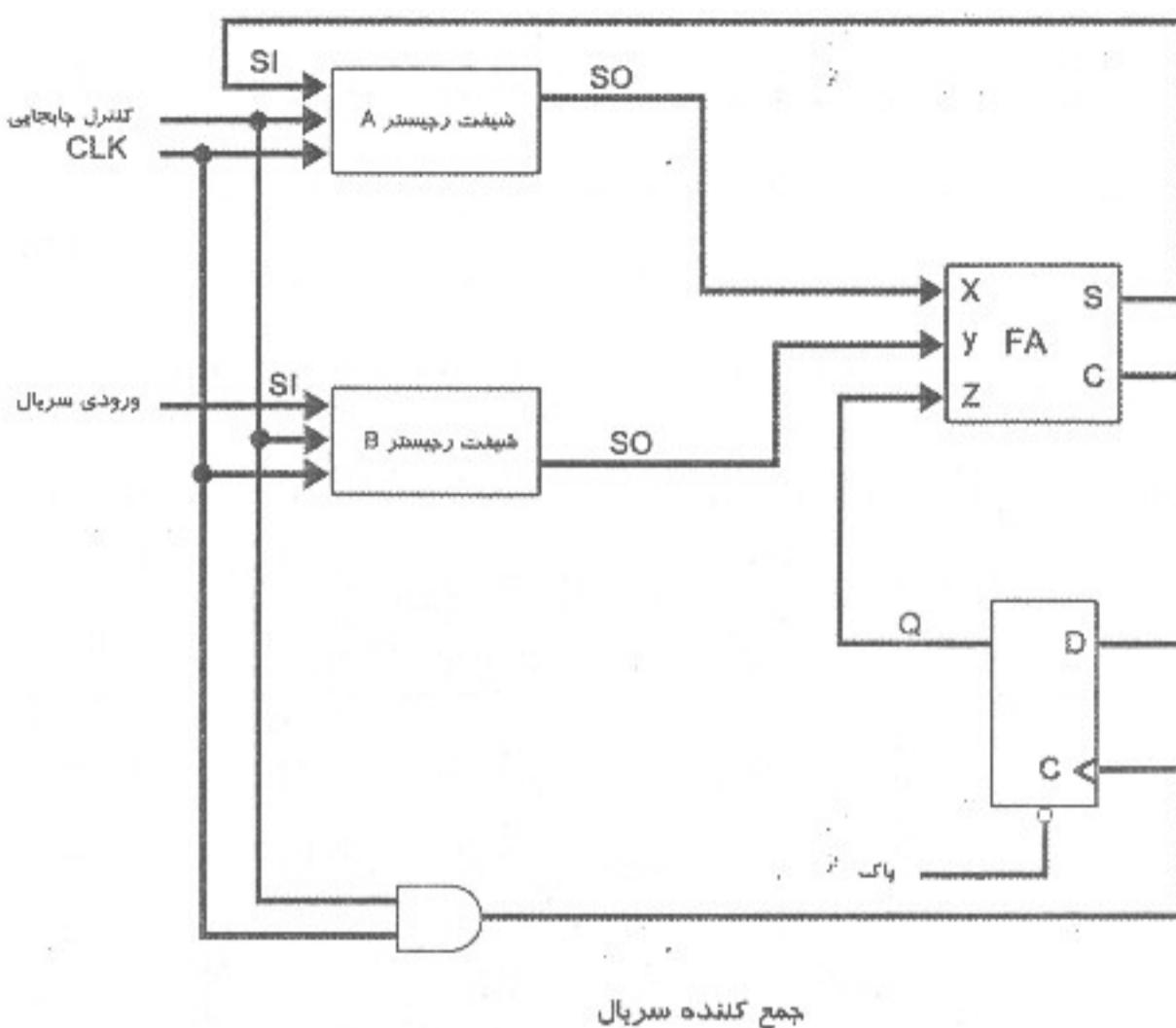
یک شیفت رجیستر به نام A داریم که عدد $A_0 A_1 A_2 \dots A_{n-1} A_n$ در آن وجود دارد یک شیفت رجیستر به نام B داریم که عدد $B_0 B_1 B_2 \dots B_{n-1} B_n$ در آن وجود دارد می خواهیم این دو عدد را با هم جمع کنیم برای این کار یک مدار طراحی می کنیم که یک انتقال سریال انجام می دهد که SO_B, SO_A به f وصل می شوند و یک خروجی به نام S (حاصل جمع) و یک خروجی به نام C (carry) دارد. علاوه بر اینها یک C_{in} وارد می شود. از طریق SI مقدار B.A در شیفت رجیسترها Load می شود حال می خواهیم عمل جمع را انجام دهیم باید یک clk مشترک برای شیفت رجیسترها قرار دهیم که وقتی کلک یک باشد دو شیفت رجیستر یک بیت، شیفت به سمت راست می دهند و یک بیت اطلاعات آنها از طریق SO وارد f می شوند یعنی B_0, A_0 وارد f می شوند و عمل جمع انجام می شود یک مقدار carry دارد و حاصل جمع S است که با carry جمع می شود و S حقیقی را با یک Cout تشکیل می دهد.

وقتی n بار کلک بزنیم و کل جمع های دو به دو انجام شود یک شیفت به راست انجام می دهد. در انتها یک C_{out}, S داریم. تابع f می تواند یک Full Adder باشد که x, y ورودی آن است به اضافه یک ورودی به نام z که همان carry ورودی است و یک خروجی به اسم S و یک خروجی به نام C_{out} دارد.

برای این که carry را وارد کنیم از یک بافر استفاده می کنیم. برای بافر از یک D فلیپ فلاپ استفاده می کنیم این شکل که C_{out} (کری خروجی) وارد ورودی D فلیپ فلاپ می شود و از خروجی فلیپ فلاپ وارد تابع f می شود که همان Full Adder است که عمل جمع را انجام می دهد.

در شکل (۵-۶) دو شیفت رجیستر B.A داریم که به جای تابع f از یک Full Adder استفاده کردیم و مقادیر x, y از طریق رجیسترهای وارد Full Adder B.A می شوند.

یک S خروجی دارد که وارد رجیستر A شده که با این کار اطلاعات A بعد از عمل جمع ازین می رود. خروجی C وارد D فلیپ فلاپ می شود.



بعد از کلاک خوردن از طریق خروجی فلیپ فلاپ وارد Z_{out} است می‌شود C_{out} فلیپ فلاپ از خط کنترل شیفت و clk وجود آمده است.

مثال: یک جمع کننده سریال طراحی کنید که برای محاسبه رقم نقلی از R.S.F.F استفاده کند؟

C_{in}	x	y	C_{ext}	S	S_e	R_e
0	0	0	0	0	0	X
0	0	1	0	1	0	X
0	1	0	0	1	0	X
0	1	1	1	0	1	0
1	0	0	0	1	0	1
1	0	1	1	0	X	0
1	1	0	1	0	X	0
1	1	1	1	0	X	0

$$S_e = xy \quad R_e = x'y' \quad S = C_{in} \oplus x \oplus y$$

تفاوت جمع کننده سریال با جمع کننده موازی در چیست؟

- جمع کننده موازی باید از ثبات‌هایی استفاده کند که امکان Load موازی داشته باشد که این به نوعی سخت افزار پیچیده‌تر است.
- در جمع کننده موازی باید به تعداد بیت‌ها شیفت رजیستر داشته باشیم ولی در جمع کننده سری تمام‌با دو شیفت رجیستر می‌توانیم عمل جمع را انجام دهیم.

مدار مدنی

مثال: مداری طراحی کنید که به کمک کارفلیپ فلاپ جمع کننده سریال را به وجود آورد.

حالت فعلی	ورودی‌ها	حالت بعدی	خروجی	ورودی فلیپ فلاپ		
C_{in}	x	y	C_{out}	S	J_C	K_C
0	0	0	0	0	0	X
0	0	1	0	1	0	0
0	1	0	0	1	0	X
0	1	1	1	0	1	X
1	0	0	0	1	x	1
1	0	1	1	0	x	0
1	1	0	1	0	x	0
1	1	1	1	0	x	0

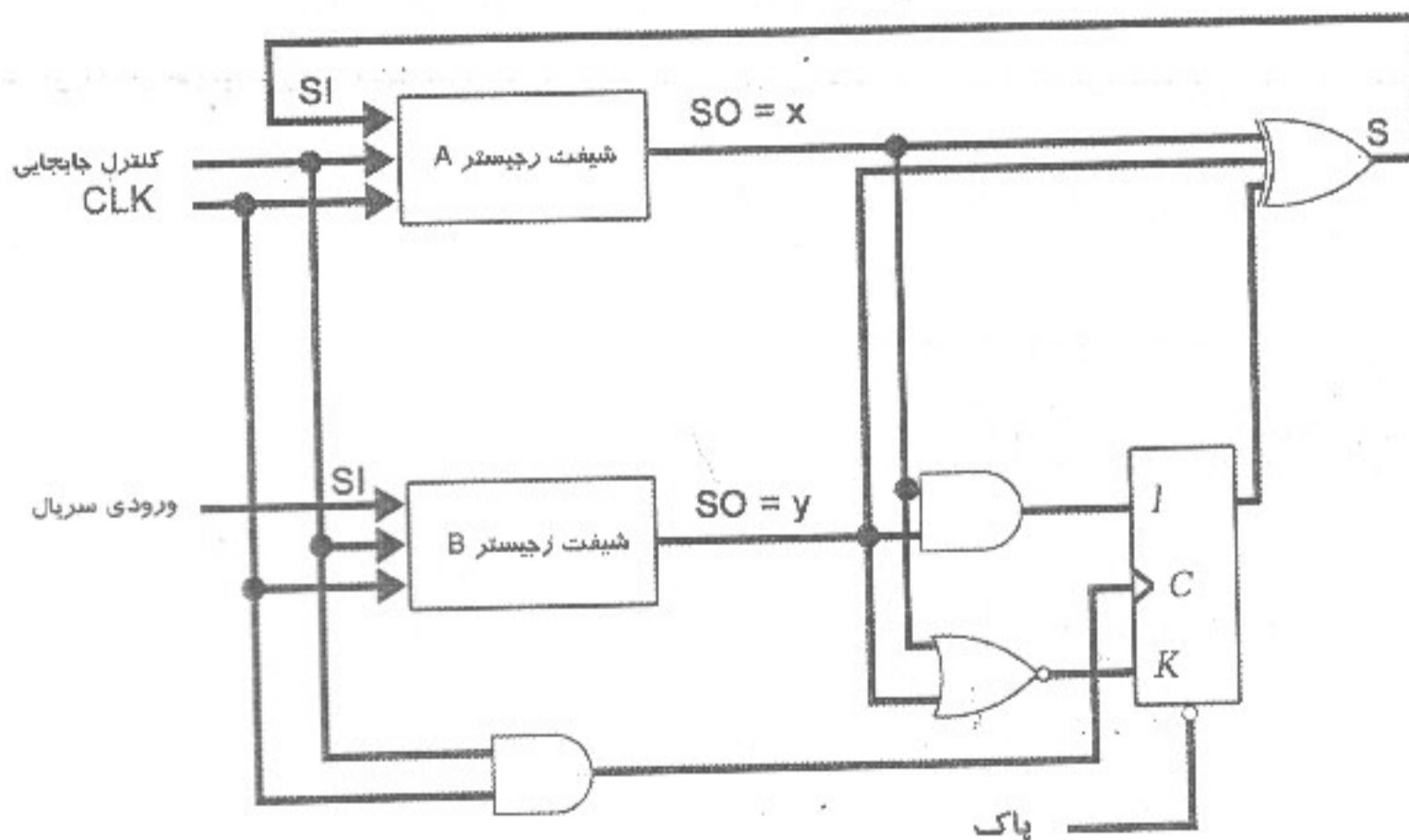
x,y را با هم جمع می کنیم حاصل را با C_m جمع می کنیم.

$$j_c = xy$$

حاصل جمع S و کری در Cout فرار می دهیم.

$$k_e = x'v'$$

$$S = C_{\frac{1}{2}h} \oplus x \oplus y$$



فرم دوم یک جمع کنندۀ سریال

در شکل بالا که فرم دوم یک جمع‌کننده سریال است ورودی Z فلیپ فلام خروجی AND است که ورودی‌های آن x و y هستند که x خروجی سریال (SO) شیفت رجیستر A است و y خروجی سریال (SO) شیفت رجیستر B است. ورودی K فلیپ فلام خروجی یک NOR است که ورودی‌های آن x و y هستند که x خروجی سریال (SO) شیفت رجیستر A است و y خروجی سریال (SO) شیفت رجیستر B می‌باشد.

شمارنده موج گونه دودویی: (آسنکرون یا غیرهم زمان)

معمول اولین فلیپ فلاپ یک کلاک پالس دارد در طراحی فلیپ فلاپها ورودی کلاک از خروجی فلیپ فلاپ قبل به دست می‌آید. به این معنی که خروجی Q فلیپ فلاپ وارد ورودی کلاک فلیپ فلاپ بعدی می‌شود.

در شمارنده آسنکرون حالاتی به نام حالات گذار است که بین دو حالت اصلی قرار می‌گیرند و در واقع مسیر بین یک حالت اصلی با حالت اصلی دیگر هستند.

مثلاً وقتی می‌خواهیم از 1010 تا 1011 بشماریم حالات ناخواسته‌ای در این بین وجود دارند که راه رسیدن به عدد بعدی را به وجود می‌آورند به این حالات که حالت اصلی بسته‌حالات گذار گفته می‌شود.

در شمارنده آسنکرون low Active High یا Active Low بودن کلاک پالس بسیار مهم است اگر ورودی کلاک Active Low باشد شمارش به شکل صعودی است و اگر کلاک به صورت Active High باشد شمارش آن به شکل نزولی است.
برای بوجود آوردن شمارنده آسنکرون صعودی یا نزولی یکی از راه‌ها تغییر دادن شکل کلاک است.

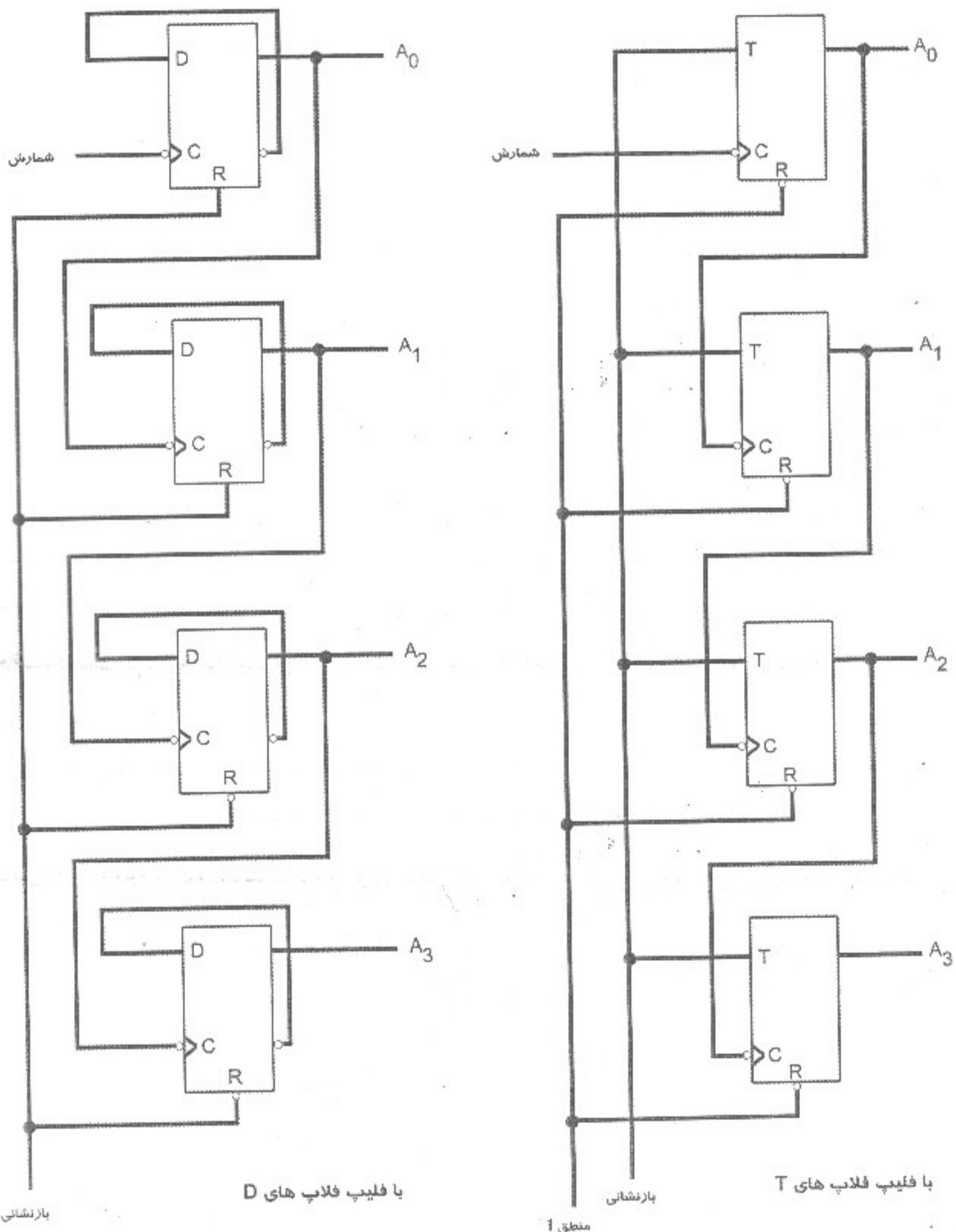
در T فلیپ فلاپ هنگامی که ورودی صفر باشد خروجی همان خروجی مرحله قبل است.

ورودی همه فلیپ فلاپ‌ها یک است هر زمان فلیپ فلاپ فعال شود باعث می‌شود خروجی مرحله قبل متمم شود ولی خروجی مرحله قبل که فعال شود کلاک پالس همه Active Low هستند و با هر گذرا باعث فعال شدن می‌شود.

نهایا فلیپ فلاپ اول است که clk حاصل است وقتی $clk=0$ باشد خروجی A_0 متمم می‌شود یعنی ۱ اعمال می‌شود به دومین فلیپ فلاپ که فعال نمی‌شود پس $A_1 = A_2 = A_3 = 0$ می‌ماند. در کلاک بعدی $T=1$ است خروجی مرحله قبل متمم می‌شود $A_0 = 0$ بعدی ۱ می‌شود پس دومی فعال نمی‌شود و بقیه هم صفر است.

با کلاک بعدی $A_0 = 1$ می‌شود وارد بعدی می‌شود و بدون تغییر باقی می‌ماند بعدی هم $A_1 = 1$ است و $A_2 = A_3 = 0$ خواهد بود.
با کلاک بعدی $A_0 = 0$ وارد فلیپ فلاپ بعدی می‌شود و آن را فعال می‌کند $A_1 = 0$ وارد می‌شود $A_0 = 1$ وارد بعدی می‌شود و بقیه مانند حالت فعلی می‌مانند.

مدار منطقی



شمارنده موج گونه دودویی 4 بیتی

A ₃	A ₂	A ₁	A ₀
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
:	:	:	:

شمارنده BCD موج گونه (ستکرون)

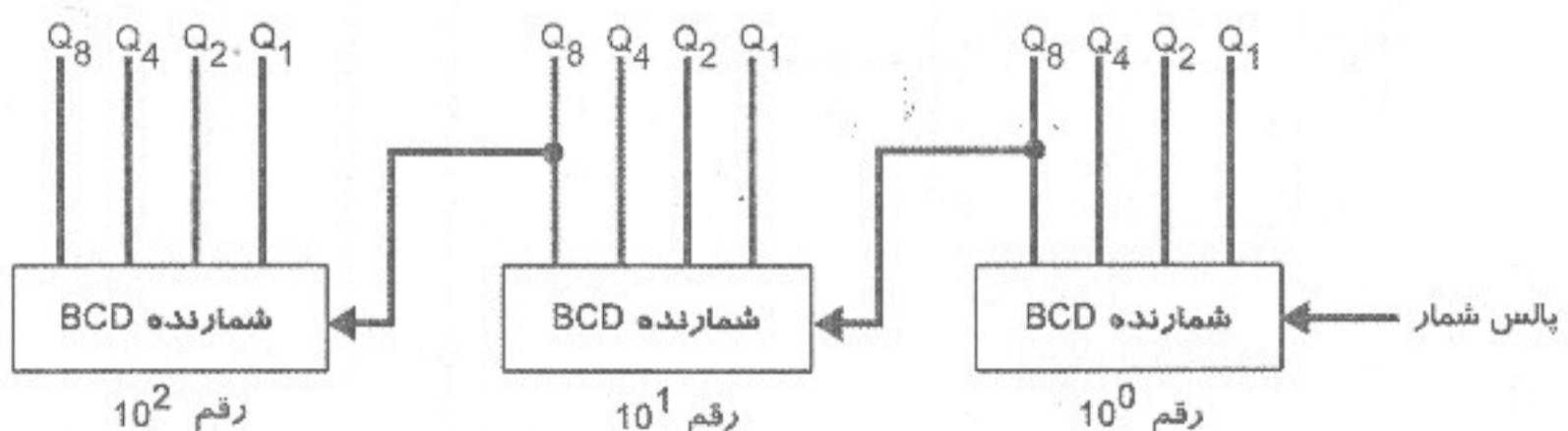
دیگر این حالت یک شمارنده BCD که از (0000) شروع شده تا به (1001) رسیده را در شکل زیر ملاحظه می‌کنید، در اینجا فقط حالات اصلی نوشته شده یعنی حالات گذرا مشخص نشده‌اند.

در شکل صفحه بعد یک شمارنده موج گونه BCD را با jk فلیپ فلاب طراحی کردیم. یعنی در شمارنده آستکرون به clk را به فلیپ فلاب اول می‌دهیم یعنی کم ارزش‌ترین است.

کلاک اول تنها کلاک مربوط به اولین فلیپ فلاب را فعال می‌کند. کلاک دوم از خروجی اولی کلاک سوم از خروجی دوم و کلاک چهارم از خروجی سوم در فلیپ فلاب اول و سوم $j=k=1$ است در دومی. تمام خروجی فلیپ فلاب چهارم (Q'_8) است و $k=1$ در فلیپ فلاب چهارم $j, k=1$, Q_1 خروجی سومی (Q_1') و خروجی دومی (Q_2') است از 0000 شروع می‌کند یک کلاک در اولی باعث می‌شود j فعال شوند چون $j, k=1$ است تمام می‌کنند با کلاک بعدی که از خروجی Q_1 آمده فلیپ فلاب‌های بعدی هم خبر فعال می‌مانند پس داریم $.0001$.

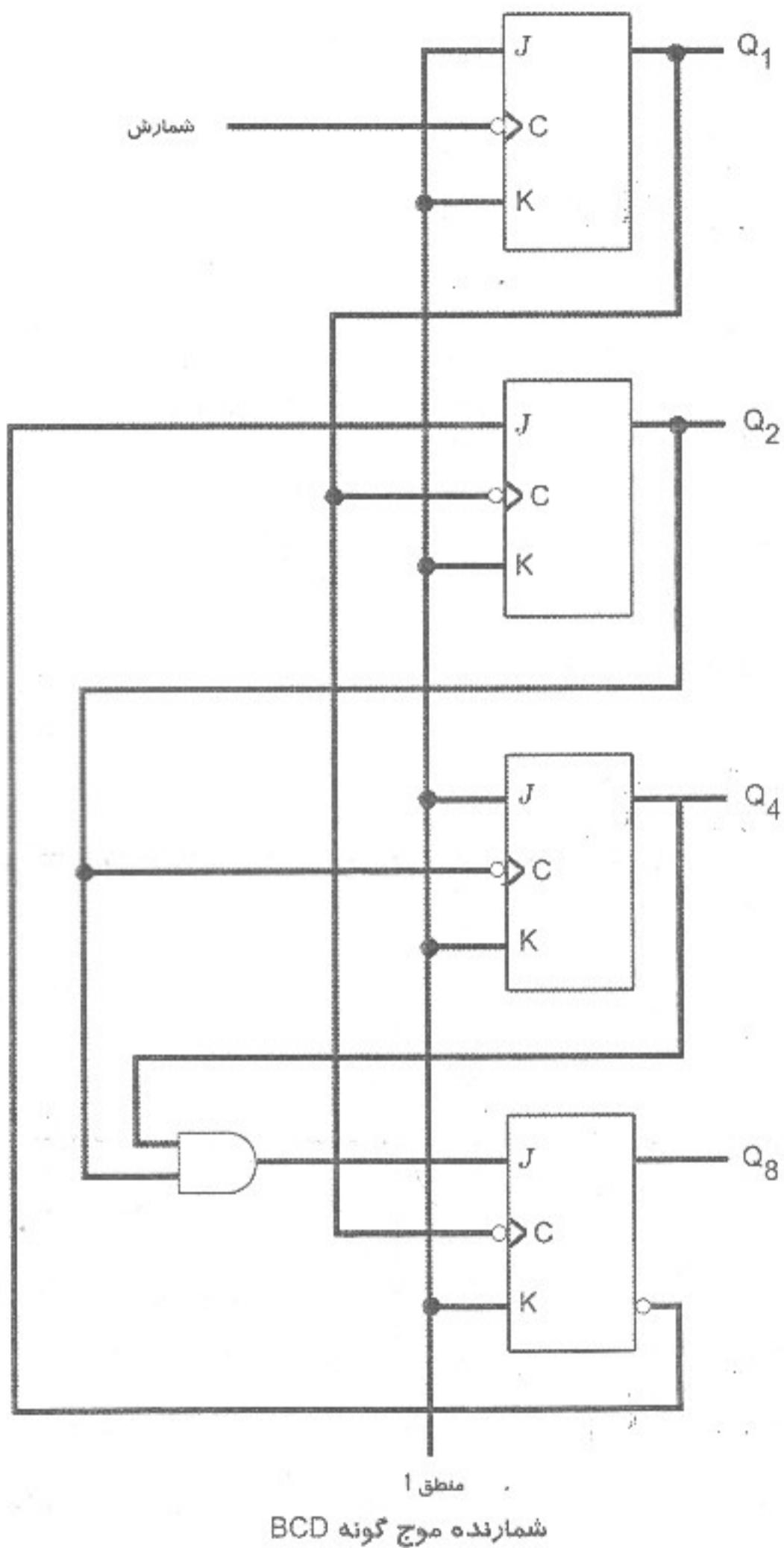
با کلاک بعدی $Q_1 = 0$ می‌شود مقدار صفر وارد فلیپ فلاب بعدی می‌شود

فلیپ فلاب دوم فعال می‌شود $j=1, k=0$ وارد y می‌شود و $j=k=1$ پس متمم می‌کند یعنی $1 = Q_2$ مقدار 1 وارد فلیپ فلاب بعدی می‌شود پس سومی و چهارمی غیرفعال خواهد بود پس داریم 0010 و به همین ترتیب ادامه می‌دهد تا به 1001 برسد.



نمودار بلوکی یک شمارنده دهدهن BCD با سه دهه

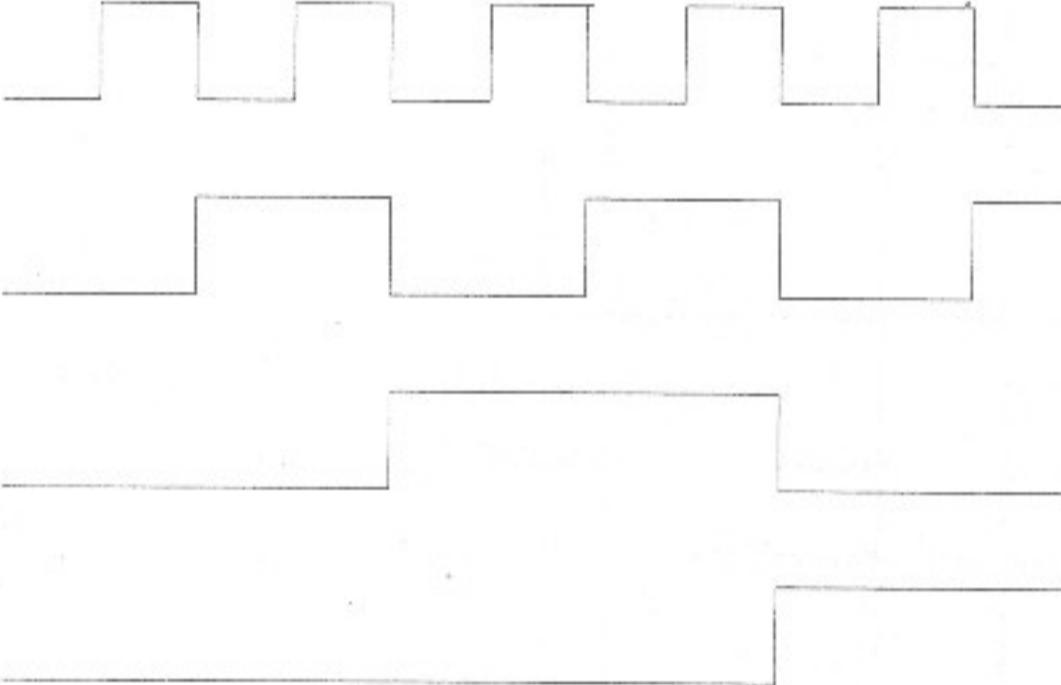
مدار منطقی





Q_4	Q_3	Q_2	Q_1
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
:	:		
1	0	0	0
1	0	0	1
0	0	0	0

طراحی خروجی فلیپ فلاپ یا شمارنده به کمک شکل کلاک پالس:



شمارنده همزمان سنکرون

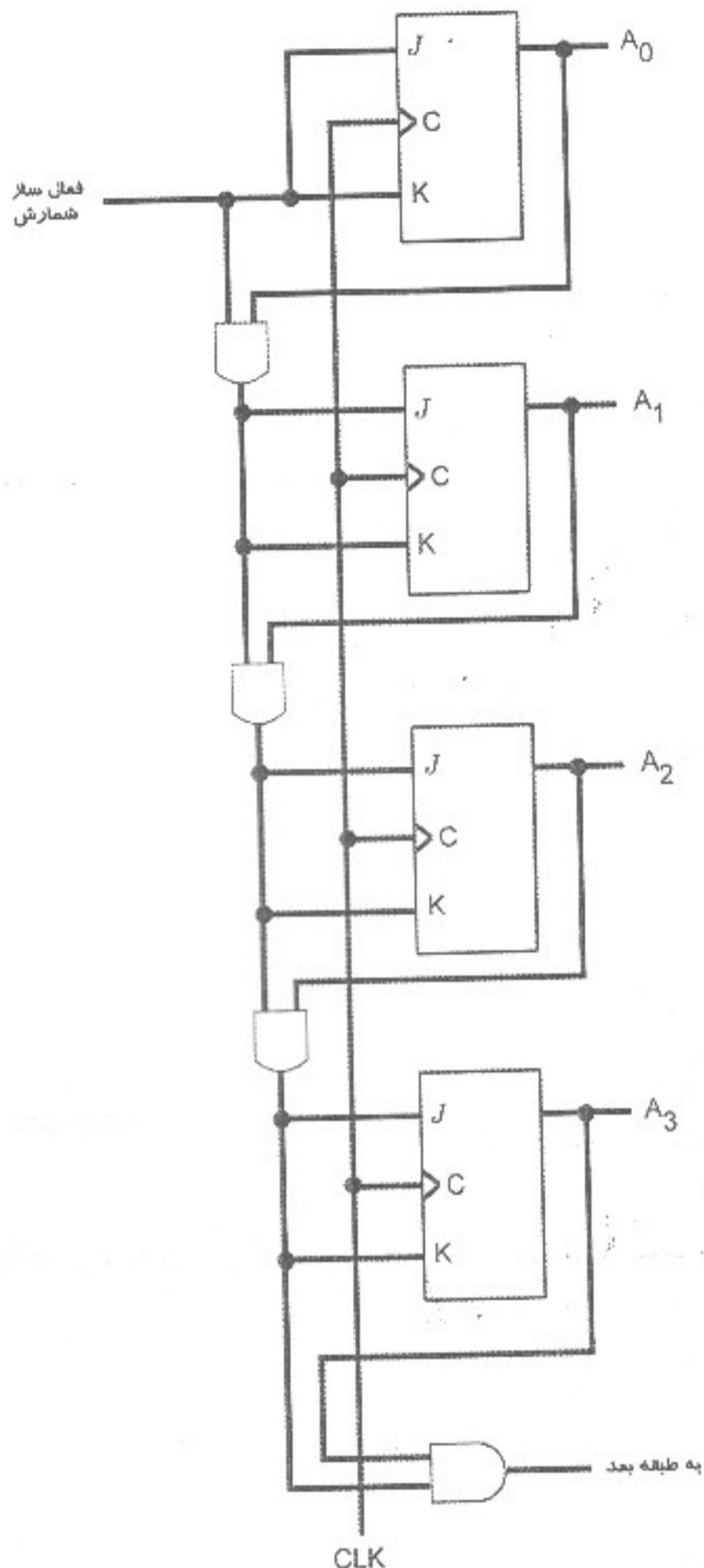
تفاوت شمارنده همزمان با غیرهمزمان در این است که در شمارنده‌های همزمان یک کلاک مشترک وجود دارد که از کلاک مرکزی تأمین می‌شود در صورتی که در شمارنده‌های غیر همزمان کلاک مرکزی تنها وارد اولین فلیپ فلاپ می‌شود و پایه کلاک بقیه فلیپ فلاپ‌ها از خروجی فلیپ فلاپ‌های قبلی تأمین می‌شود.

شمارنده دودویی سنکرون

در یک شمارنده دودویی کلاک همه مشترک است اولین نکته در شمارنده همزمان این است که Active high یا Active Low بودن کلاک پالس تفاوتی در نحوه شمارش نمی‌کند. ورودی j,k فلیپ فلاپ اول به خط فعال ساز شمارش وصل است یعنی وقتی فعال ساز شمارش ۱ باشد در فلیپ فلاپ اول $j=k=1$ باشد و خروجی A_0 را متمم می‌کند.

اگر خروجی $A_0 = 0$ یک باشد باعث می‌شود که ورودی فلیپ فلاپ بعدی $j=k=1$ باشد و باعث متمم شدن خروجی A_1 می‌شود و در صورت صفر بودن هیچ تغییری در مقدار A_1 به وجود نمی‌آورد و به همین ترتیب تا آخر ادامه پیدا می‌کند.

مدار منطقی



شمارنده دودویی همزمان 4 بیتی

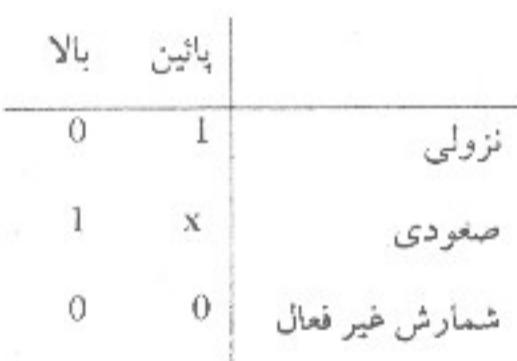


	A_3	A_2	A_1	A_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
⋮	⋮	⋮	⋮	⋮

بالا، پایین شمار دودویی:

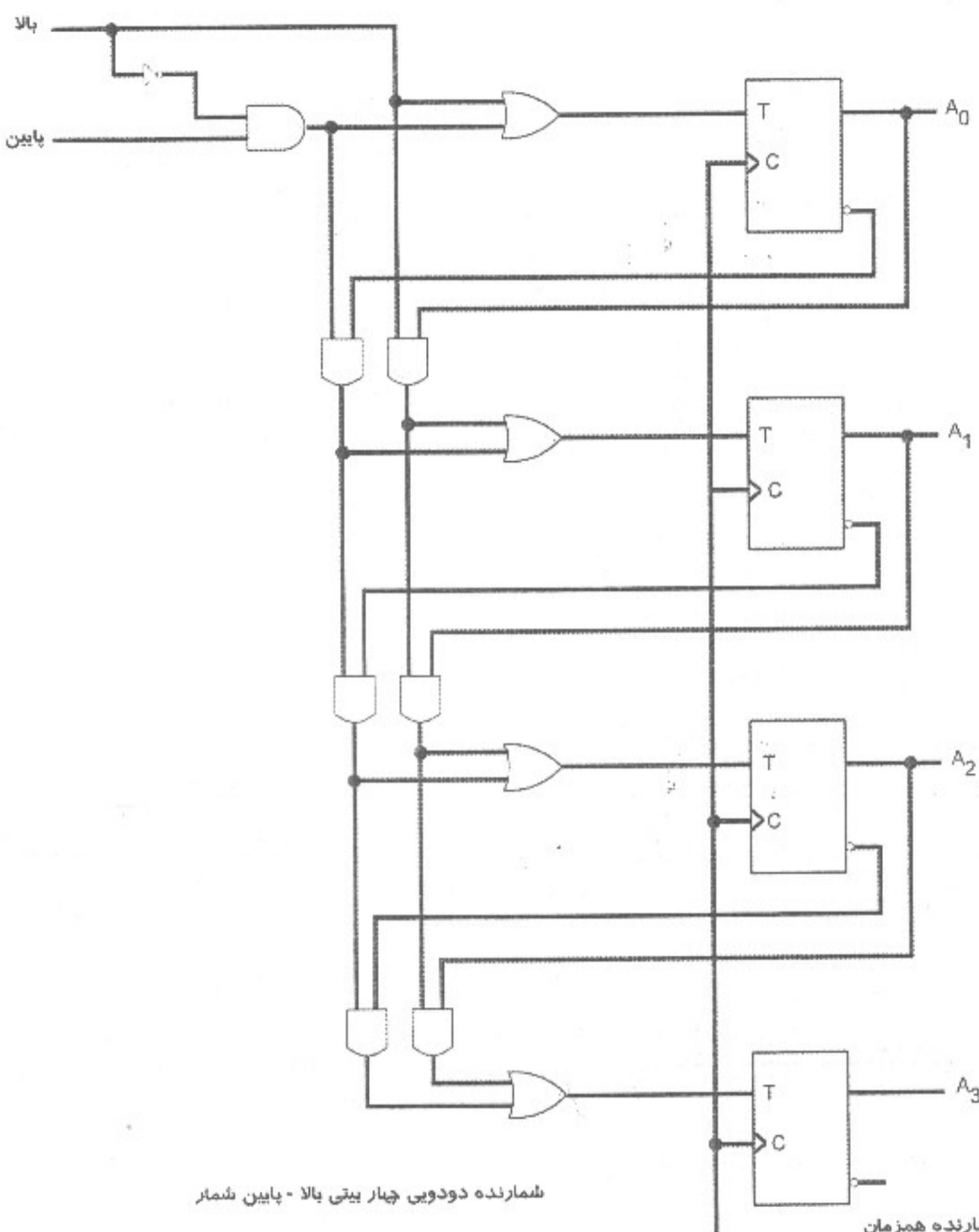
منظور از بالا و پایین شمار دودویی یک مداری است که توانایی دارد صعودی یا نزولی شمارش کند و خط کنترل دارد که این خطوط تعیین می‌کنند که شمارنده صعودی شمارد یا نزولی یا شمارش را قطع کند. شکل (۵-۱۳) یک شمارنده دودویی چهار بیتی بالا - پایین شمار است که اگر خط بالا 1 باشد شمارنده صعودی می‌شمارد خط پائین می‌تواند صفر یا یک باشد و اگر خط بالا صفر باشد و خط پائین مقدارش یک باشد شمارنده به صورت نزولی می‌شمارد.

خط بالا صفر و خط پائین صفر باشد شمارش غیرفعال خواهد بود در طراحی شکل موردنیاز T فلیپ فلاپ استفاده شده در صورتی که بخواهیم از Kaz فلیپ فلاپ استفاده کنیم. تنها باید ورودی زو کارابه هم وصل کنیم.



باید از OR که خروجی اش وارد فلیپ فلاپ می‌شود به صورت زیر استفاده کنیم

مدار منطقی



A_3	A_2	A_1	A_0
0	0	0	0
15	1	1	1
14	1	0	1
5	0	1	0
4	0	1	0

شمارنده BCD

در فصل قبل طراحی شده

حالت فعلی				حالت بعدی				خروجی	ورودی فلیپ فلاپ			
Q_8	Q_4	Q_2	Q_1	Q_8	Q_4	Q_2	Q_1	y	TQ_8	TQ_4	TQ_2	TQ_1
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	1	0	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	0	1
0	0	1	1	0	1	0	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	0	1
0	1	0	1	0	1	1	0	0	0	0	1	1
0	1	1	0	0	1	1	0	0	0	0	1	1
0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	0	1
1	0	0	1	0	0	1	0	1	0	0	0	1
$TQ_1 = 1$				$TQ_4 = Q_2 Q_1$								
$TQ_2 = Q_8 Q_1$				$TQ_8 = Q_8 Q_1 + Q_4 Q_2 Q_1$								
$Y = Q_8 Q_1$												

شمارنده دودویی با امکان بار شدن موازی:

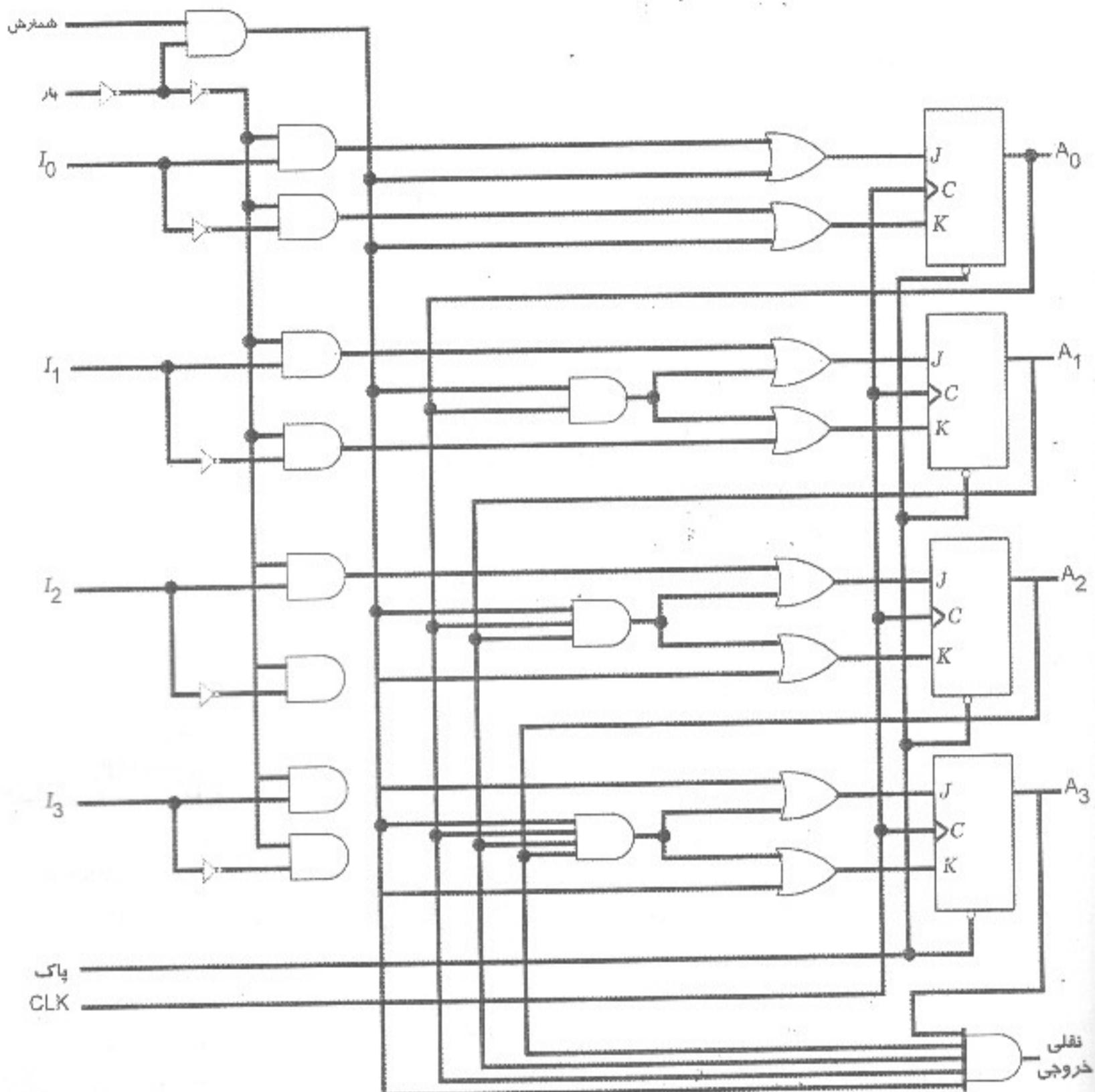
در واقع مثل رجیستر که بار شدن موازی را انجام می‌داد عمل می‌کند، با این تفاوت که یک امکان دیگر به آن اضافه شده که کاملتر بودن است. یعنی علاوه بر این که شمارش را انجام می‌دهد این امکان را دارد که I_{in} بیت را در خود Load کند.

با Load موازی این امکان را داریم که یک عدد را برای شروع شمارش به آن بدهیم در شکل (۱۴-۵) از ورودی‌های I_0 تا I_3 به صورت Load می‌شوند. خط I_0 برابر یک است محل مقدار I از And عبور می‌کند وارد OR می‌شود و در AND هم I' به همراه خط Load I و I' استفاده می‌کنیم به این صورت که در AND اول خود I به همراه خط Load ورودی می‌شود و در AND هم I' به همراه خط Load I' وارد می‌شوند. خط I_1 برابر یک است محل مقدار I از And عبور می‌کند وارد OR می‌شود ورودی دیگری OR خط شمارش است در شمارنده یا Load موازی ورودی داریم یکی خط Load و دیگری کنترل شمارش است.

اگر خط Load برابر یک باشد عمل Load موازی انجام می‌شود اگر خط شمارش یک باشد آن‌گاه $j=k=1$ می‌شود خروجی And صفر می‌شود.

تابع	شمارش	بار	CLK	پاک کننده
به 0 پاک می‌شود	x	x	x	0
ورودی بار شونده	↑	1	x	1
شمارش حالت دودویی بعدی	↑	0	1	1
بلا تغییر	↑	0	0	1

مدار منطقی

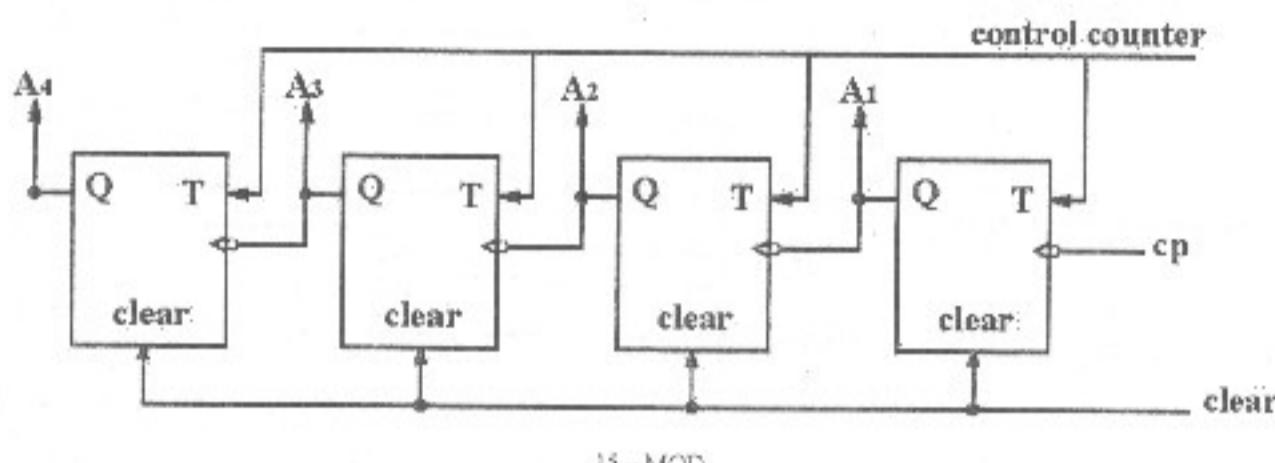


شمارنده دودویی ۴ بیتی با امکان شدن موازی

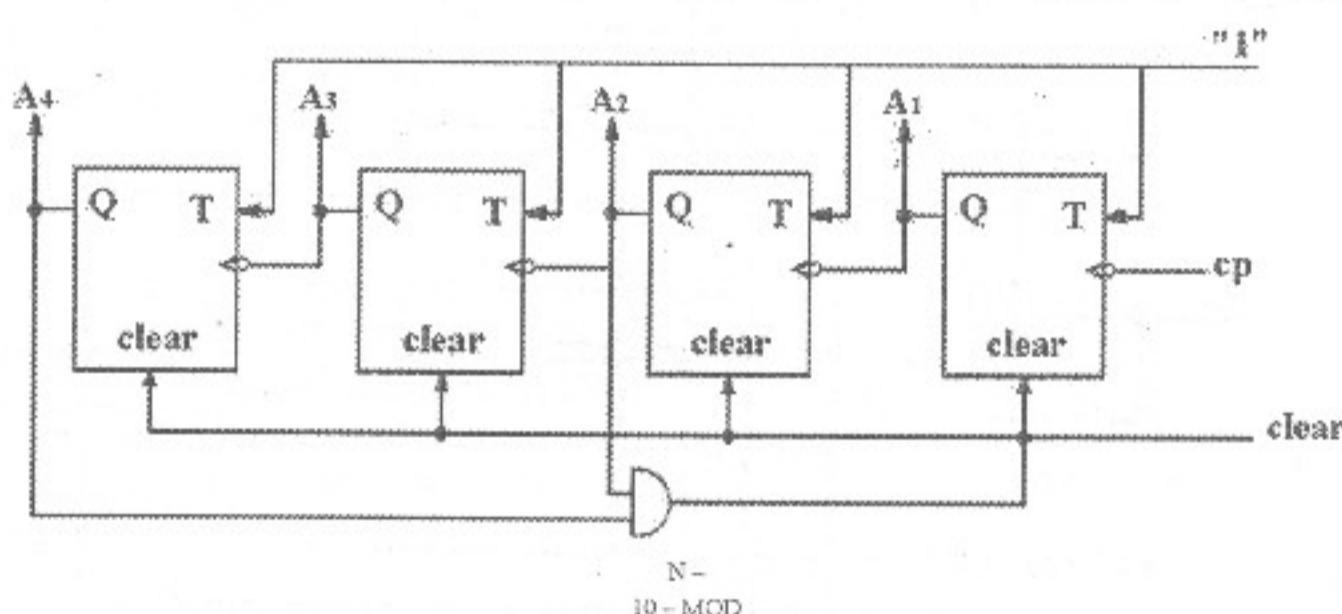
شمارنده سنکرون مازول - (MOD - N)

اگر یک شمارنده بتواند از صفر تا $1 - N$ را بشمارد آن شمارنده را مازول N می‌نامیم. به عنوان مثال اگر شمارنده مازول 8 داشته باشیم این شمارنده از صفر تا 7 را می‌تواند بشمارد.

شمارنده زیر یک شمارنده 15 - MOD می‌باشد.



و در ضمن مدار زیر نیز یک شمارنده سنکرون 10 - MOD می‌باشد یعنی از صفر تا 9 را می‌شمارد و دوباره بارگذاری از نو می‌شود.



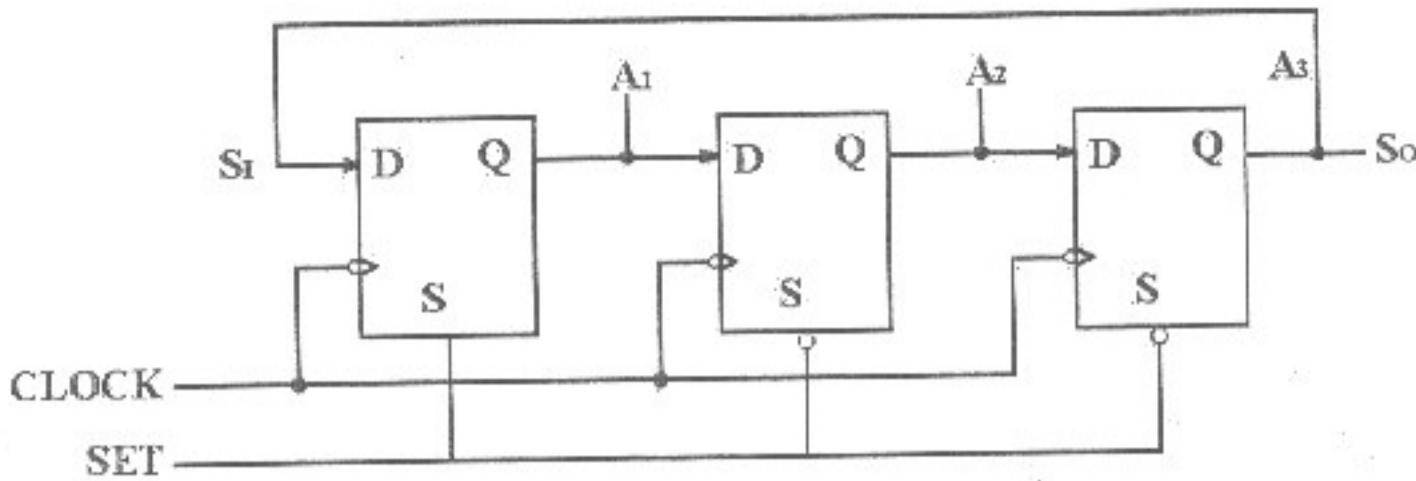
شمارنده حلقوی (Ring Counter)

اگر یک شمارنده n بیتی داشته باشیم که هر کدام از F.F های موجود حملکردن مانند فلپ فلاپ باشد و خروجی آخرین فلپ فلاپ به ورودی اولین فلپ فلاپ وصل شود می‌توان یک شمارنده حلقوی ایجاد کرد. در یک شمارنده حلقوی می‌توان تمامی توانهای صحیح z را ایجاد کرد. که این شمارش تا 2^{n-1} ادامه پیدا می‌کند.

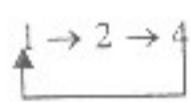
شمارنده n بیتی $1, 2, 4, 8, 16, \dots, 2^{n-1}$

(با n فلپ فلاپ)

مدار منطقی



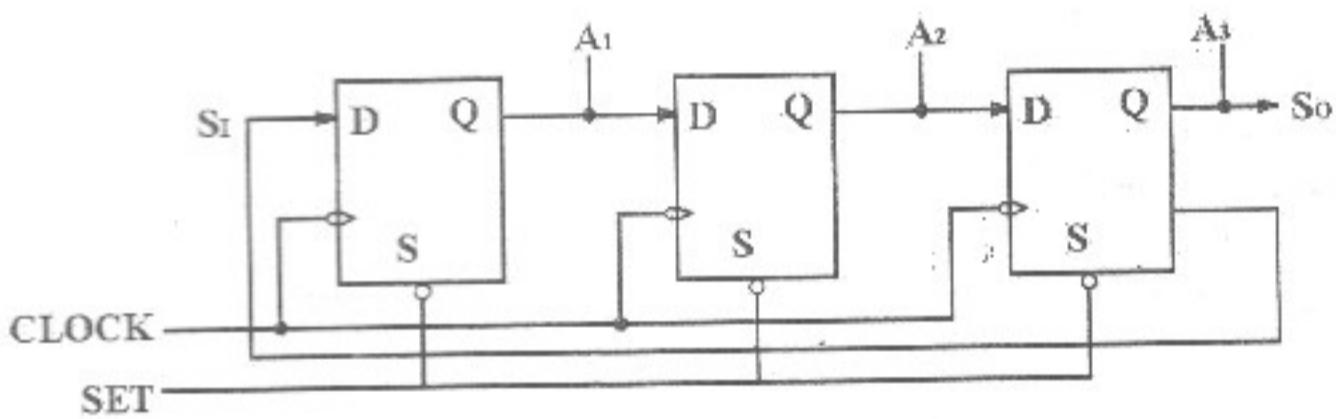
شمارنده زیر یک شمارنده حلقوی نه بیتی می‌باشد که دنباله زیر را تولید می‌کند:



تذکرہ: برای ایجاد مقدار اولیه باید به اوین فلیپ فلاپ خود S را وصل کیم و سایر فلیپ فلاپ‌ها به Reset یا نقیض Set وصل شود.

شمارنده جانسون:

مانند شمارنده حلقوی عمل می‌کند با این تفاوت که ورودی S_1 از 'Q' آخرین فلیپ فلاپ وارد می‌شود. و در ضمن برخلاف شمارنده حلقوی اوین فلیپ فلاپ نیز از طریق نقیش S مقداردهی اولیه می‌شود.



برای تحلیل این شمارنده مقدار SET را صفر در نظر گرفته و در این حالت دنباله $1 \rightarrow 2 \rightarrow 4 \rightarrow 6 \rightarrow 7$ تولید می‌شود.

جانسون

0	0	0	0	1	1	1
1	0	0	1	1	1	0
3	0	1	1	1	0	0
7	1	1	1	0	0	0
6	1	1	0	0	0	1
4	1	0	0	0	1	1
0	0	0	0	1	1	1

فرکانس خروجی شمارندها:

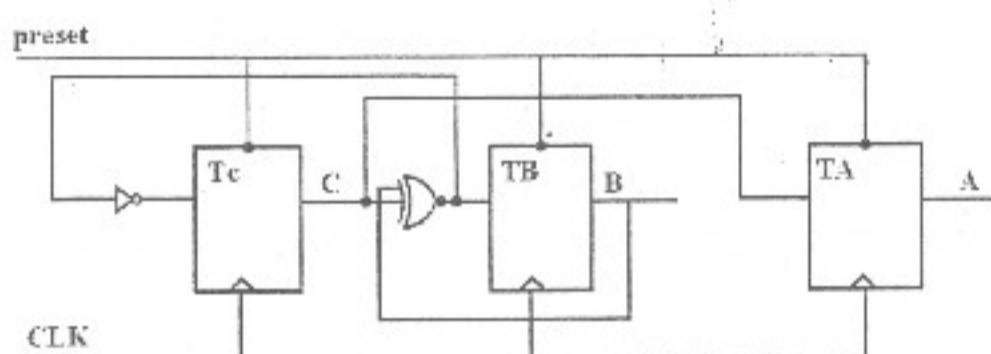
اگر فرکانس ورودی یک شمارنده MHZ باشد و این شمارنده توانایی شمارش n عدد را داشته باشد می‌توان فرکانس خروجی را از رابطه زیر بدست آورد:

$$\frac{\text{فرکانس ورودی}}{\text{تعداد حالات قابل شمارش توسط شمارنده}} = \frac{\text{MHZ}}{n}$$

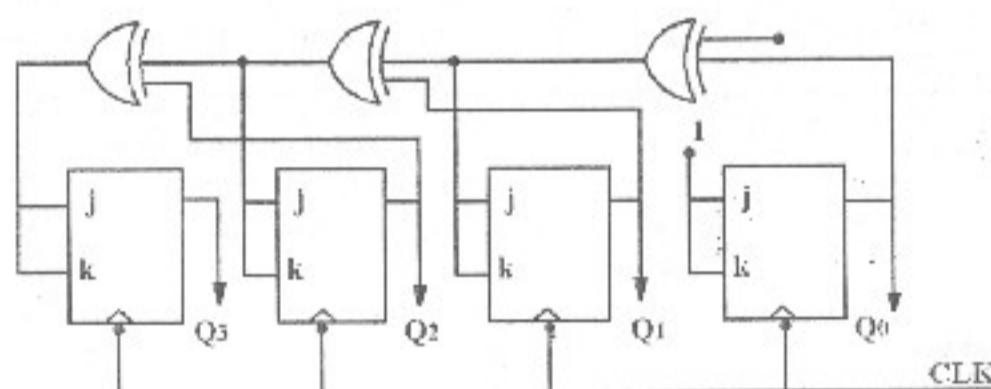
مجموعه مسائل فصل ۵

۱- در شمارنده زیر قبل از اعمال پالس‌های ساعت CIK شمارنده با آمدن پالس‌های ساعت سیکل شمارش عبارت است از :

	C	B	A	
preset	1	1	1	→ 7
	1	0	0	4
	0	0	1	1
	0	1	1	3
	1	1	1	7



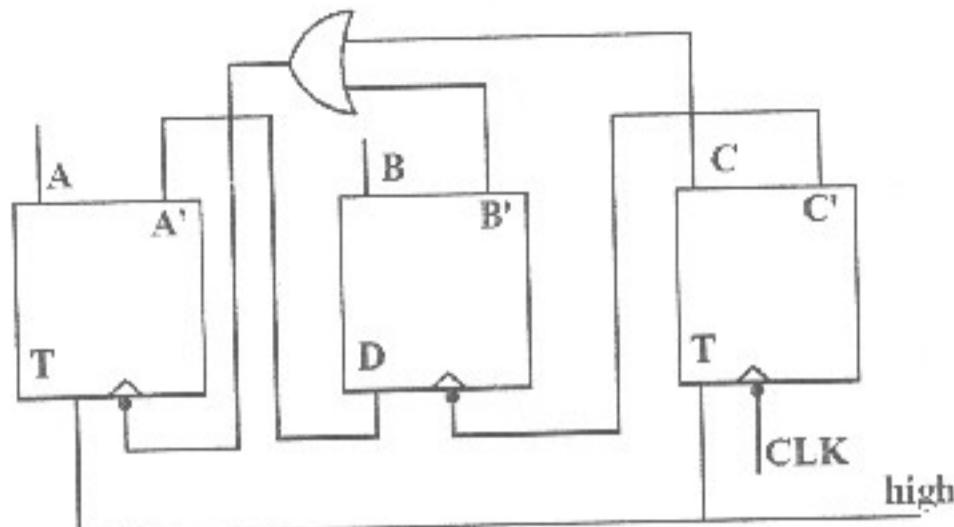
۲- در حالت زیر بیان کنید پس از ۵ پالس ساعت چه رشته‌ای تولید می‌شود؟



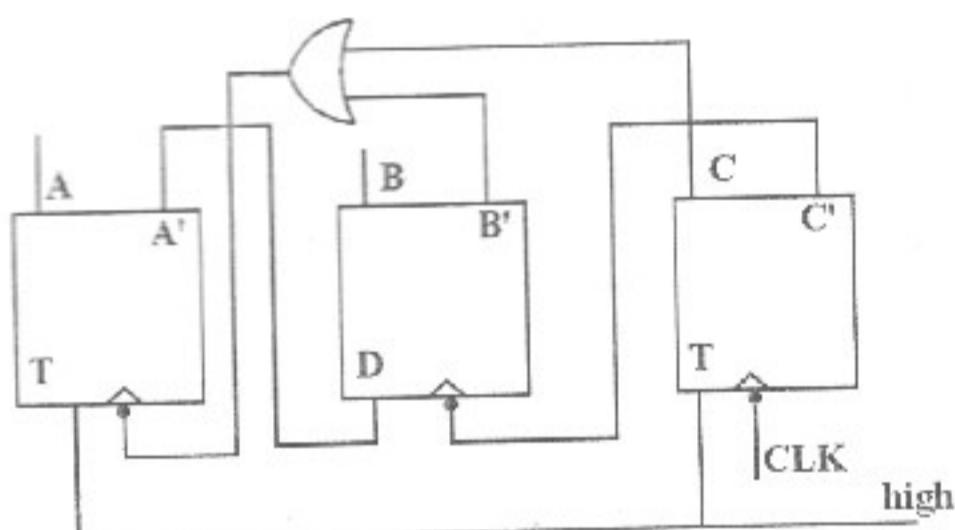
	Q_3	Q_2	Q_1	Q_0		3	1	1	0	1	→	13
	0	0	1	0		4	1	0	1	0	→	10
1	1	1	1	1	→	15	5	0	1	1	→	7
2	0	1	0	0	→	4						

مدار منطقی

۳- اگر $ABC = 000$ باشد تحلیل کنید پس از ۶ کلاک پالس ABC برابر چند خواهد بود؟



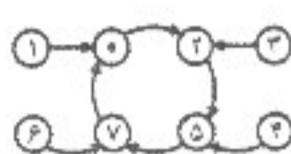
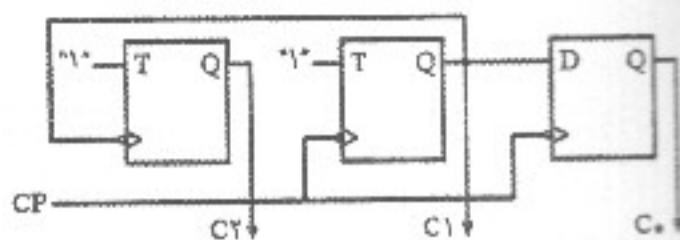
A	B	C	D
0	0	0	
0	0	1	①
0	0	0	②
0	0	1	③
0	0	0	④
0	0	1	⑤
0	0	0	⑥



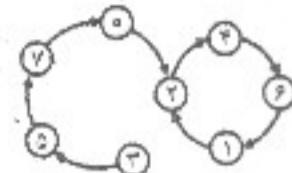
A	B	C
1	0	1
1	0	0
1	0	1
⋮		
1	0	1
		6

مدار منطقی

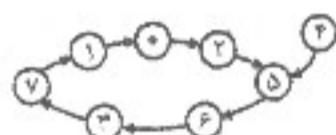
۵- دنباله شمارش مدار مقابله کدام است $\Psi(C_2, C_1, C_0)$



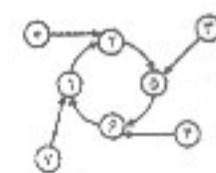
(۱)



(۲)

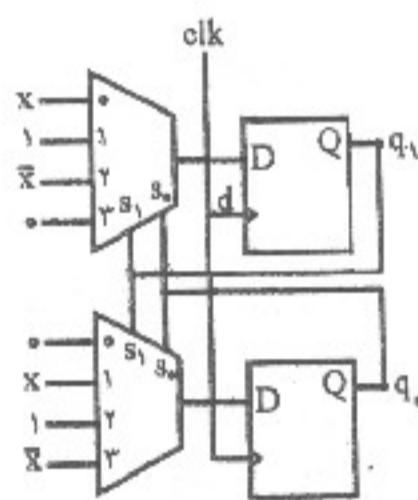


(۳)

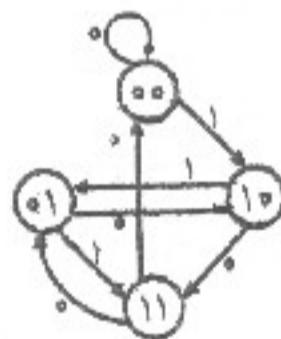


(۴)

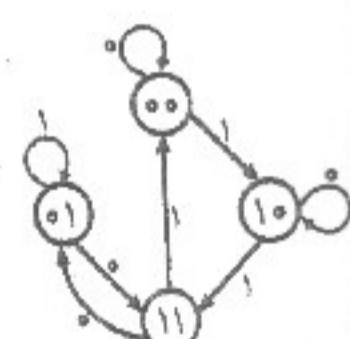
۶- نمودار حالت برای مدار زیر کدام است؟



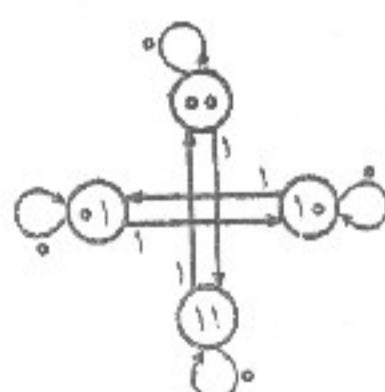
(۱)



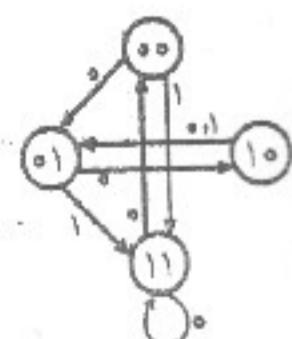
(۱)



(۲)

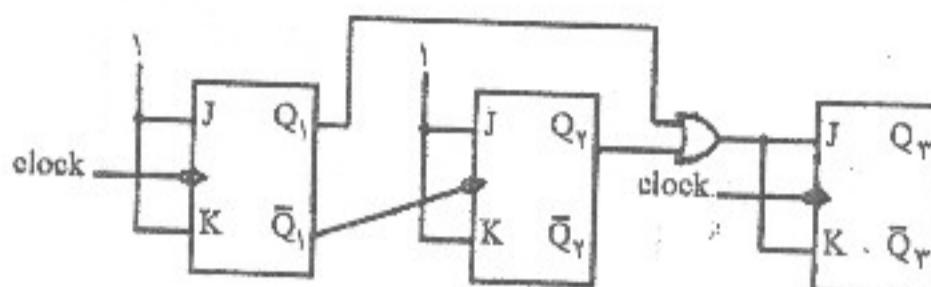


(۳)



(۴)

۷- شمارنده زیر چه رشته‌ای را به صورت تکراری می‌شمارد؟



$① \rightarrow ② \rightarrow ④ \rightarrow ⑥ \rightarrow ⑦ \rightarrow ① \rightarrow ② \rightarrow ⑤ \rightarrow \dots$ (۱)

$① \rightarrow ③ \rightarrow ⑤ \rightarrow ⑦ \rightarrow ② \rightarrow ① \rightarrow ④ \rightarrow ⑥ \rightarrow \dots$ (۲)

$① \rightarrow ③ \rightarrow ⑥ \rightarrow ⑧ \rightarrow ③ \rightarrow ⑥ \rightarrow ⑧ \rightarrow ① \rightarrow ⑥ \rightarrow \dots$ (۳)

$① \rightarrow ③ \rightarrow ⑥ \rightarrow ⑧ \rightarrow ① \rightarrow ④ \rightarrow ⑦ \rightarrow ② \rightarrow ⑤ \rightarrow \dots$ (۴)

۸- کدام یک از جملات زیر این ماشین را بهتر توصیف می‌کند؟

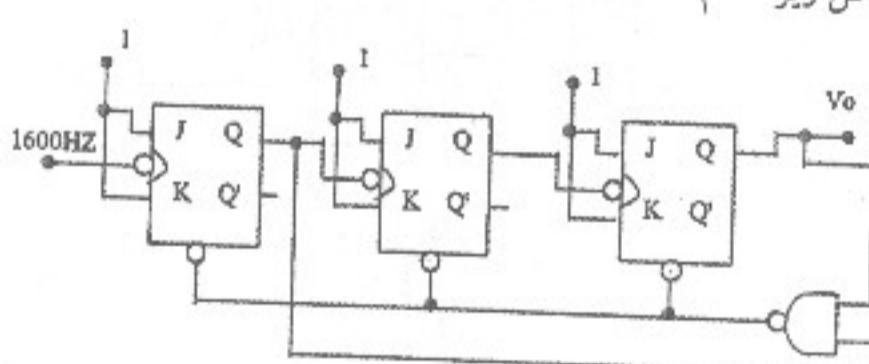
(۱) دارای مسابقه غیر بحرانی است و نیازی به رفع آن نمی‌باشد.

(۲) دارای مسابقه بحرانی (critical race) است و رفع آن به صورت زیر است:

(۳) دارای مسابقه بحرانی است و رفع آن به صورت زیر است:

(۴) هر دو جواب ۲ و ۳ صحیح هستند.

۹- فرکانس موج خروجی در مدار شکل زیر کدام است؟



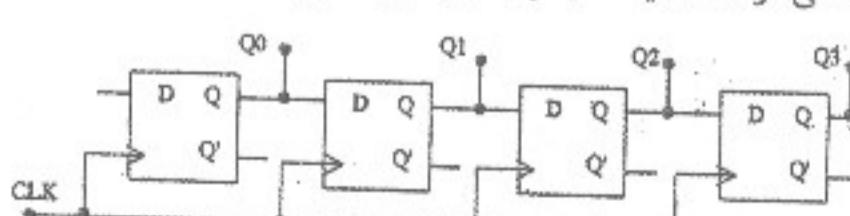
400 HZ (۱)

160 HZ (۲)

200 HZ (۳)

320 HZ (۴)

۱۰- در شکل زیر یک ثبات ۴ یتی مشاهده می‌شود. کدامیک از موارد زیر غیرممکن است؟



(۱) خارج کردن داده‌ها به صورت سری از ثبات

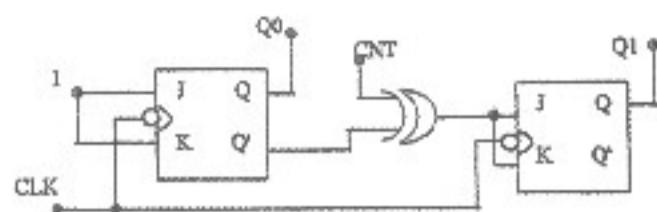
(۲) خارج کردن داده‌ها به صورت موازی از ثبات

(۳) وارد کردن داده‌ها به صورت سری از ثبات

(۴) وارد کردن داده‌ها به صورت موازی به ثبات

مدار منطقی

۱۱۱- مدار شکل زیر:

۱) به شرط $CNT = 1$ یک شمارنده پایین شمار (نزولی) ۴ بیتی است.۲) به شرط $CNT = 0$ یک شمارنده بالا شمار (صعودی) ۲ بیتی است.۳) به شرط $CNT = 0$ یک شمارنده پایین شمار ۲ بیتی است.۴) به شرط $CNT = 1$ یک شمارنده بالا شمار ۴ بیتی است.۱۱۱- یک Shift Register دارای m ورودی و n خروجی است نام این شیفت رجیستر چیست؟

PIPO (۲)

SIPO (۱)

۱) بستگی دارد m بزرگتر است یا نه
۲) چنین چیزی امکان پذیر نیست.۱۱۱- عدد $\underline{\underline{m}}$ (IA) پس از دوبار شیفت منطقی به راست و سپس سه بار شیفت منطقی به چپ به چه عددی تبدیل می شود؟

AB (۴)

IB (۳)

08 (۲)

30 (۱)