سخت افزار (۲)

رشته کامپیوتر- نرم افزار

دانشگاه آزاد اسلامی

سازمان سما

آموزشکده فنی و حرفه ای سما اندیشه

تألیف : مهندس رضوانی

نام درس : تحلیل و طراحی مدارهای منطقی ترکیبی و پیاده‌سازی مدارهای محاسباتی و مبدل های کد

هدف از این فصل آشنایی با مفاهیم زیر می‌باشد :

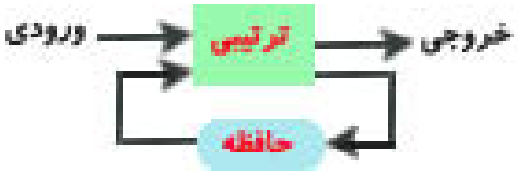
- شناخت انواع مدارهای منطقی و تعریف مدارهای منطقی ترکیبی و ترتیبی
- شناخت تفاوت های میان مدارهای ترکیبی و مدارهای ترتیبی و ویژگی های هر کدام
- نحوه تحلیل و آنالیز مدارهای منطقی ترکیبی و نمودارهای مداری متشکل از گیت ها
- مراحل طراحی مدارهای منطقی ترکیبی
- ساخت مدارهای محاسباتی جمع کننده، تفریق گر، جمع کننده BCD و ضرب‌کننده

انواع مدارهای منطقی : مدارهای منطقی در سیستم های دیجیتال می‌توانند از نوع ترکیبی یا ترتیبی باشند.

مدارهای منطقی ترکیبی : یک مدار ترکیبی متشکل از تعدادی گیت منطقی است که خروجی آنها در هر لحظه از زمان مستقیماً به ورودی های همان لحظه بستگی داشته و به ورودی های قبلی بستگی ندارد. این نوع مدار پردازشی را انجام می‌دهد که با مجموعه‌ای از توابع بولی مشخص می‌گردد.

یک مدار ترکیبی می‌تواند دارای n متغیر ورودی و m متغیر خروجی باشد. برای هر ترکیب ممکن از ورودی ها، فقط یک مقدار برای هر خروجی موجود است. بنابراین، یک مدار ترکیبی با یک جدول درستی که مقادیر خروجی‌ها را برای هر ترکیب از متغیرهای ورودی لیست می‌نماید، نشان داده می‌شود. هر یک از خروجی های یک مدار ترکیبی را می‌توان توسط یک عبارت جبری نیز بیان نمود.

مدارهای منطقی ترتیبی : مدارهای ترتیبی علاوه بر گیت های منطقی، از عناصر حافظه نیز استفاده می‌کنند. خروجی های این مدارها در هر لحظه از زمان، تابعی از ورودی ها و حالت عناصر حافظه در آن لحظه است. لذا عملکرد مدار باید به وسیله حالات داخلی و ترتیب زمانی ورودی ها مشخص گردد.



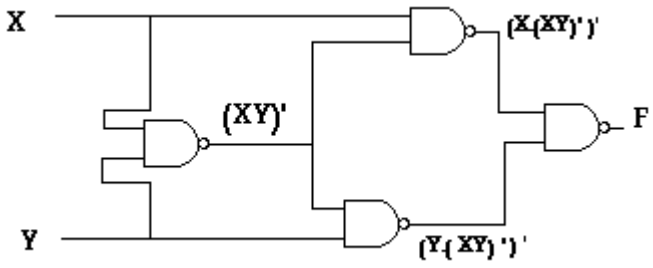
در این فصل به بررسی مدارهای ترکیبی، ویژگی ها و نحوه تحلیل و طراحی آنها می‌پردازیم. مدارهای منطقی ترتیبی، مبحث فصل بعدی این درس خواهد بود.

تحلیل مدارهای ترکیبی : تحلیل یا آنالیز یک مدار ترکیبی با یک نمودار منطقی آغاز شده و با مجموعه‌ای از توابع بول، یک جدول درستی یا توضیحاتی از عملکرد مدار پایان می‌یابد.

مراحل تحلیل یک مدار ترکیبی و به دست آوردن توابع بولی خروجی :

۱. تعیین تابع خروجی گیت هایی که تابعی از ورودی ها هستند و نام گذاری آنها.
۲. تعیین تابع خروجی گیت هایی که تابعی از متغیرهای ورودی و گیت های برچسب خورده قبلی هستند و نام گذاری آنها.
۳. تکرار مرحله ۲ تا دستیابی به خروجی های مدار.
۴. به دست آوردن توابع بولی خروجی نهایی بر حسب متغیرهای ورودی اولیه با جایگزینی توابع به دست آمده در مراحل قبل.
۵. رسم جدول درستی برای خروجی های مدار.
۶. شرح عملکرد مدار با توجه به ارتباط میان مقادیر خروجی ها با ورودی ها در هر حالت.

مثال : جدول درستی مدار زیر را رسم نموده و عملکرد آن را آنالیز نمایید.



T0 = (XY)'
T1 = (Y.T0)' = (Y.(XY)')' = Y' + (XY) = (Y'+X)(Y'+Y) = Y' + X
T2 = (X.T0)' = (X.(XY)')' = X' + (XY) = (X'+X)(X'+Y) = X' + Y
F = (T1.T2)' =((Y'+X).(X'+Y))'= (X'+Y)' + (Y'+X)' = XY' + X'Y ⟶ F = XY' + X'Y ⟶ F = X XOR Y

X	Y	F
0	0	0
0	1	1
1	0	1
1	1	0

طبق جدول ملاحظه می‌شود که عملکرد این مدار معادل گیت XOR دو متغیره است.

طراحی مدارهای ترکیبی : طراحی مدارهای ترکیبی با مشخصات مسئله آغاز و به فرم نمودار مدار منطقی یا مجموعه‌ای از توابع بول برای خروجی ها پایان می‌یابد.

مراحل طراحی یک مدار ترکیبی به روش کلاسیک :

۱. تعیین تعداد ورودی ها و خروجی ها با استفاده از مشخصات مدار و تخصیص سمبل های حرفی به آن ها
۲. تشکیل جدول درستی مربوط به ورودی ها و خروجی های مدار
۳. به دست آوردن توابع بولی ساده شده برای هر خروجی به صورت تابعی از متغیرهای ورودی
۴. ترسیم نمودار منطقی مدار در صورت لزوم

روش های کلاسیک و غیر کلاسیک در طراحی مدارهای منطقی ترکیبی : در طراحی مدارهای ترکیبی به روش کلاسیک، با افزایش متغیرهای ورودی، ترسیم جدول درستی و جداول کارنو برای ساده‌سازی توابع خروجی به مراتب مشکل تر شده و ضریب اشتباه طراح را بالا خواهد برد.

اما توابع دیجیتالی نیز وجود دارند که ذاتاً دارای نظم درونی هستند و معمولاً با روال های الگوریتمی و روش های غیر کلاسیک و ساده‌تر مانند استفاده از توابع طراحی شده قبلی، قابل طراحی‌اند. در روش غیر کلاسیک، هر مسئله‌ای روش خاص خودش را دارد و از روال منظمی پیروی نمی‌کند.

در ادامهٔ درس، طراحی مدارهای مختلف ترکیبی به خصوص مدارهای محاسباتی- منطقی مورد استفاده در ALU و مدارات مبدل کد را مورد بررسی قرار می‌دهیم.

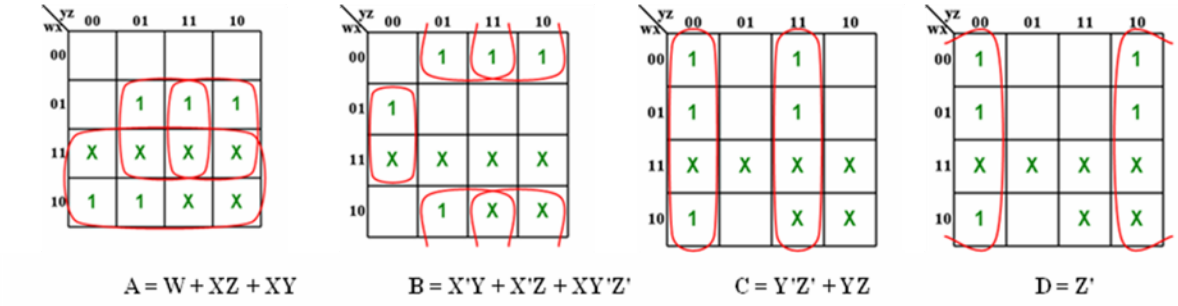
مدار مبدل کد BCD به افزونی ۳ (Excess-3) : سیستم های دیجیتال مختلف از کدهای متفاوتی برای ذخیره-سازی اطلاعات استفاده می‌کنند. برای برقراری ارتباط میان دو سیستم که از کدهای مختلفی برای بیان اطلاعات یکسان استفاده می‌کنند، یک مدار مبدل کد باید بین آن دو قرار داده شود.

بنابراین یک مدار مبدل کد، مداری است که دو سیستم را علیرغم به کارگیری کدهای دودویی متفاوت، با هم سازگار می‌سازد.

مثال : یک مدار مبدل کد BCD به افزونی ۳ طراحی نمایید.

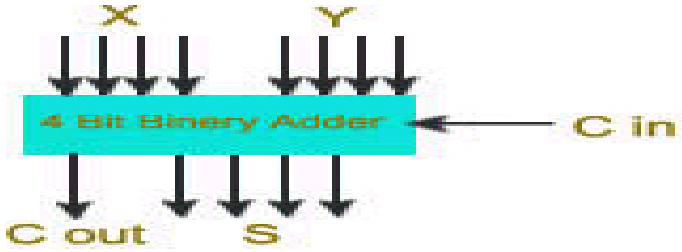
روش کلاسیک : در این روش، طبق مراحل ذکر شده در صفحات قبلی عمل می‌کنیم.

W	X	Y	Z	A	B	C	D
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	X	X	X	X
1	0	1	1	X	X	X	X
1	1	0	0	X	X	X	X
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X



روش غیر کلاسیک : با توجه به عملکرد کد افزونی ۳ می توان نتیجه گرفت که اگر بتوانیم به هر کد ورودی BCD، ۳ واحد اضافه کنیم به مطلوب مسئله خواهیم رسید.

برای این منظور توسط یک جمع کننده چهار بیتی که در مثال های آتی طراحی خواهیم نمود، عدد ورودی را با یک عدد ثابت ۳ معادل باینری 0011 جمع کرده و رقم نقلی ورودی را صفر در نظر می گیریم.



توجه ۱ : همانطور که ملاحظه می شود روش غیر کلاسیک ما را مستقیماً به مدار می رساند ولی برای هر صورت مسئله، روش خاص خود را دارد. در حالی که روش کلاسیک برای تمام مسائل راه حل یکسانی دارد و مراحل آن مشخص می باشد.

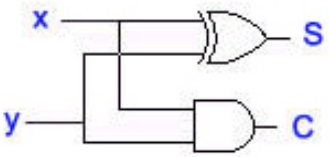
توجه ۲ : در طراحی مدارهای منطقی با اینکه با روش های طراحی کلاسیک شروع می کنیم، ولی بسته به مهارت به دست آمده می‌توان از راه حل‌های تحلیلی بهره گرفت. در ادامهٔ درس از راه حل های غیر کلاسیک بیشتر استفاده می شود.

جمع کننده بیتی یا نیم جمع کننده (Half Adder) : نیم جمع کننده مداریست که ۲ بیت را با یکدیگر جمع می‌کند. حاصل جمع ۲ بیت، عددی ۲ بیتی خواهد بود. برای طراحی مدار نیم جمع کننده، مراحل طراحی را دنبال می کنیم. به این ترتیب که شکل بلوکی، جدول درستی، جدول کارنو و در نهایت روابط منطقی را به دست می آوریم.

در این مدار، x و y ورودی های مدار و S و C خروجی های مدار هستند. S به معنی Sum یا حاصل جمع ۲ بیت و C به معنی Carry یا رقم نقلی خروجی است. جدول درستی مدار به صورت زیر است :

X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$C = X.Y$
$S = X \oplus Y$



تمام جمع کننده (Full Adder) : در جمع دو عدد n بیتی به علت وجود رقم نقلی به جمع کننده‌ای که قابلیت جمع سه بیت با هم را داشته باشد، احتیاج داریم. پس طراحی جمع کننده سه بیت مورد نیاز است. جمع کننده‌ای که قابلیت جمع سه بیت با هم را داشته باشد، تمام جمع کننده (Full Adder) نامیده می‌شود. حاصل جمع ۳ بیت، عددی ۲ بیتی خواهد بود.

برای طراحی مدار تمام جمع کننده، مراحل طراحی را دنبال می‌کنیم. به این ترتیب که شکل بلوکی، جدول درستی، جدول کارنو و در نهایت روابط منطقی را به دست می آوریم.

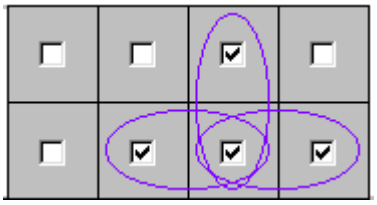
در این مدار، x، y وz ورودی های مدار و S و C خروجی های مدار هستند. S به معنی Sum یا حاصل جمع ۳ بیت و C به معنی Carry یا رقم نقلی خروجی است. جدول درستی مدار به صورت زیر است :

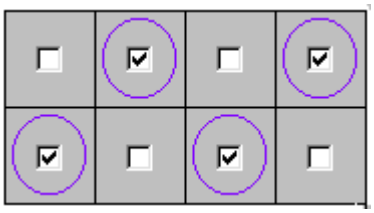
X	Y	Z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$C = \sum m(3,5,6,7) = X'YZ + XY'Z + XYZ' + XYZ = XZ + XY + YZ$

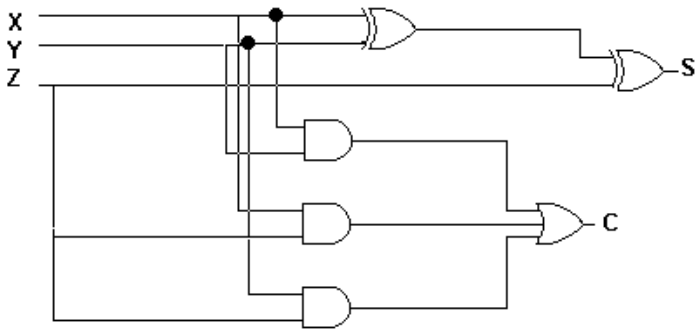
$S = \sum m(1,2,4,7) = X'Y'Z + X'YZ' + XY'Z' + XYZ = X \text{ XOR } Y \text{ XOR } Z$

جدول کارنوی خروجی های تابع تمام جمع کننده به شرح زیرند :

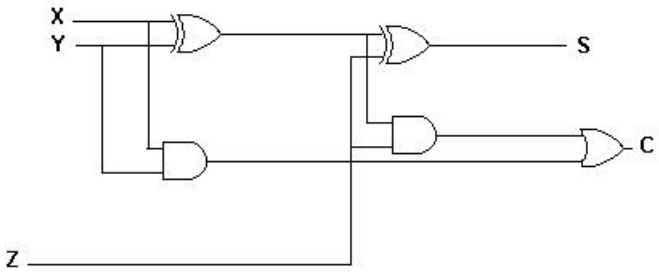




نمودار منطقی خروجی های تابع به شکل زیر است :

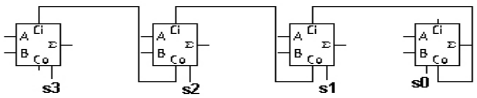


پیاده‌سازی تمام جمع کننده با استفاده از نیم جمع کننده : با استفاده از دو نیم جمع کننده و یک گیت OR می‌توان یک تمام جمع کننده ساخت. این عمل با یک دست کاری ساده در پوشش های جدول درستی به شکل زیر امکان‌پذیر خواهد بود :



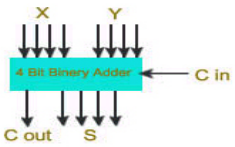
جمع کننده دودویی n بیتی : در طراحی جمع کننده‌های n بیتی به روش کلاسیک، با افزایش تعداد بیت های اعداد، تعداد سطرهای جدول های درستی به شدت افزایش یافته و به دست آوردن توابع ساده شده خروجی به راحتی امکان پذیر نخواهد بود. به عنوان مثال، برای طراحی یک جمع کننده چهار بیتی به یک جدول درستی با $۲^۹=۵۱۲$ سطر نیاز خواهیم داشت. لذا به روش غیر کلاسیک عمل خواهیم کرد.

در این روش، برای جمع اعداد چند بیتی نیاز به جمع کننده های بزرگتر داریم. پس با استفاده از n عدد تمام جمع کننده، یک جمع کننده n بیتی می سازیم. به عنوان مثال، در یک جمع کننده چهار بیتی، چهار عدد تمام جمع کننده در کنار هم قرار می‌گیرند. لذا این مدار یک مدار تکرار پذیر است. در این جمع کننده، رقم نقلی به صورت موج گونه از تمام جمع کننده ها منتشر شده و عبور می کند. به این نوع جمع کننده، جمع کننده با رقم نقلی موج گونه گفته می شود.



X = X3X2X1X0

Y = Y3Y2Y1Y0

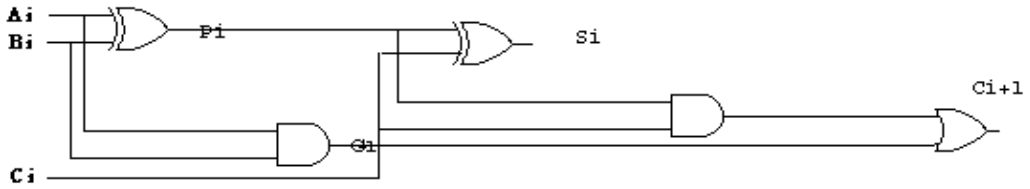


C : Ripple Carry　رقم نقلی موجی

این نوع جمع کننده را جمع کننده با رقم نقلی موج گونه می‌نامیم. اشکال عمدهٔ این روش طراحی جمع کننده، زمان انتشار طولانی مورد نیاز برای تولید رقم نقلی نهایی از بیت پر ارزش اعداد می‌باشد، زیرا برای تولید خروجی های نهایی در هر جمع کننده، ۲ طبقه گیت تاخیر وجود خواهد داشت و خروجی هر طبقه نیز به آماده بودن ورودی طبقه ماقبل خود وابسته می‌باشد. لذا به عنوان مثال برای یک جمع کننده ۴ بیتی، ۸ طبقه گیت تاخیر وجود خواهد داشت. این تاخیر برای یک جمع کننده ۸ بیتی، برابر ۱۶ خواهد بود و مشاهده می‌شود که با افزایش تعداد بیت های اعداد، زمان لازم برای تولید خروجی نهایی به شدت افزایش خواهد یافت.

طراحی جمع کننده n بیتی با پیش بینی رقم نقلی : برای حل مشکل جمع کننده با رقم نقلی موج‌گونه، باید وابستگی نقلی خروجی هر طبقه به طبقات قبلی حذف شود، لذا جمع کننده جدیدی طراحی شده است که در آن، رقم نقلی ورودی هر جمع کننده بیتی، به رقم نقلی طبقه قبل وابسته نبوده و زودتر آماده می‌شود. در این حالت همه رقم‌های نقلی بطور همزمان آماده خواهند شد. به این ترتیب این نوع جمع کننده، تاخیر رقم های نقلی جمع کننده موج گونه را برای آماده شدن جواب نخواهد داشت.

برای این منظور با استفاده از عملیات جبری روابط جدیدی برای محاسبه رقم نقلی هر طبقه به دست آورده‌اند و برای طرح مدار مربوطه از تمام جمع کننده‌ها ایده گرفته شده است.



در مدارهای فوق، وابستگی رقم نقلی دیده می شود. اما با استفاده از دست کاری های جبری همان طور که در روابط زیر مشاهده می‌شود، رقم نقلی بیت های بعدی فقط به رقم نقلی اولی بستگی داشته و به سایر رقم‌های نقلی ارتباط ندارد.

$P = A_i \text{ xor } B_i$

$G = A_iB_i$

$S_i = A_i \text{ xor } B_i \text{ xor } C_i = P_i \text{ xor } C_i$

$C_{i+1} = A_i.B_i + C_i(A_i \text{ xor } B_i) = P_iC_i + G_i$

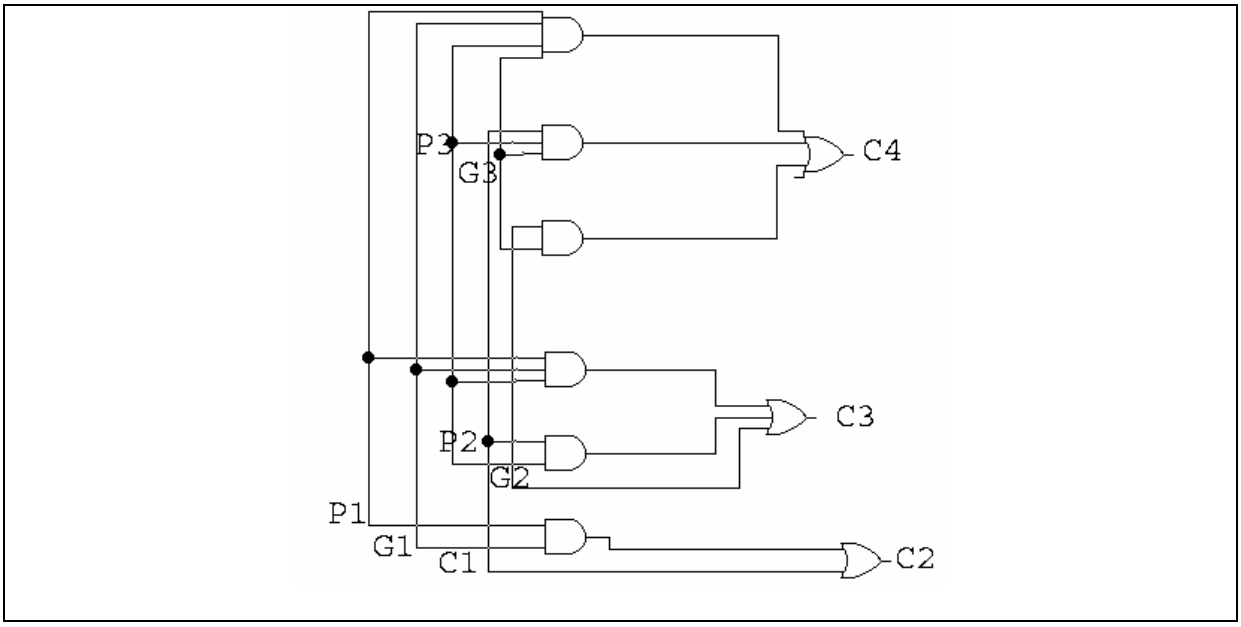
$S_i = P_i \oplus C_i$
$C_{i+1} = P_iC_i + G_i$

$S1 = P1 \oplus C1$
$C2 = P1C1 + G1$

$S2 = P2 \oplus C2$
$C3 = P2C2 + G2$
$C3 = P2P1C1 + P2G1 + G2$

$S3 = P3 \oplus C3$
$C4 = P3C3 + G3$
$C4 = P3P2P1C1 + P3P2G1 + P3G2 + G3$

در شکل زیر مدار تولید رقم‌های نقلی برای یک جمع کننده ۴ بیتی رسم شده است. همان گونه که مشاهده می‌شود، تاخیر انتشار لازم برای تولید رقم نقلی خروجی در تمام طبقات جمع کننده با هم یکسان بوده و برابر ۴ طبقه گیت است.



تفریق کننده بیتی یا نیم تفریق کننده (Half Subtractor) : برای تفریق دو عدد چند بیتی لازم است ابتدا تفریق دو بیت را انجام دهیم، لذا در این مرحله می‌خواهیم مدار تفریق دو بیت یا نیم تفریق کننده را طراحی کنیم.

نیم تفریق کننده مداریست که ۲ بیت را از هم کم می‌کند. حاصل تفریق، عددی ۲ بیتی خواهد بود. این مدار عبارت زیر را محاسبه می کند که ممکن است یک بیت از بیت سمت چپ قرض گرفته شود : **X-Y** برای طراحی مدار نیم تفریق کننده، مراحل طراحی را دنبال می کنیم. به این ترتیب که شکل بلوکی، جدول درستی، جدول کارنو و در نهایت روابط منطقی را به دست می آوریم.

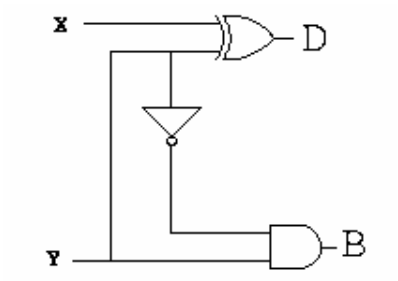
در این مدار، x و y ورودی های مدار و D و B خروجی های مدار هستند. D به معنی Difference یا حاصل تفریق ۲ بیت و B به معنی Borrow یا رقم قرضی خروجی است. جدول درستی مدار به صورت زیر است :

X	Y	B	D
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

معادلات خروجی‌ها و نمودار مداری نیم تفریق کننده به صورت زیرمی باشند :

D = X xor Y

B = X Y'



تمام تفریق کننده (Full subtractor) : در تفریق دو عدد n بیتی به علت وجود رقم قرضی به تفریق کننده‌ای نیاز است که قابلیت تفریق ۲ بیت همراه با رقم قرضی احتمالی را از طبقه قبل داشته باشد. تفریق کننده‌ای که قابلیت تفریق ۳ بیت از هم را داشته باشد ، تمام تفریق کننده (Full Subtractor) نامیده می‌شود. در تمام تفریق کننده عبارت زیر محاسبه می شود :

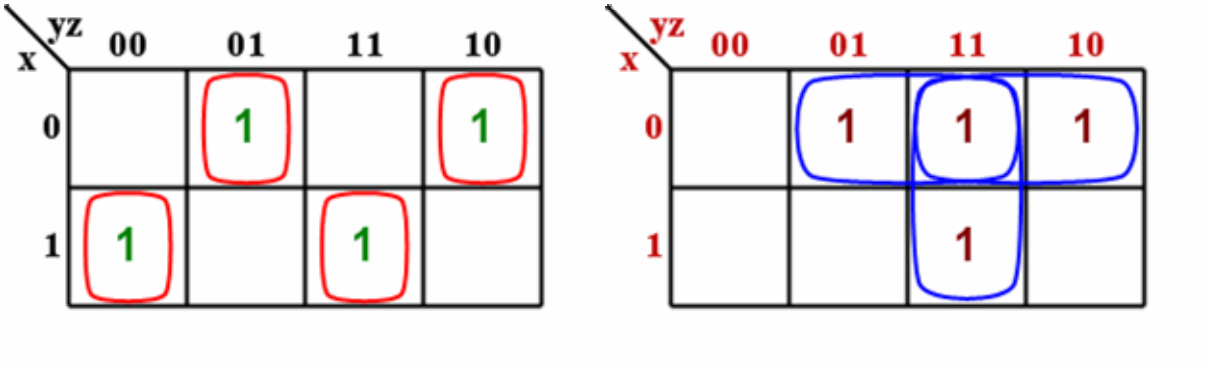
X-(Y+Z) یا X-Y-Z

جدول درستی، جداول کارنو و در نهایت روابط منطقی خروجی های تمام تفریق کننده به شرح زیر به دست می‌آیند :

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

$D = X \text{ xor } Y \text{ xor } Z$

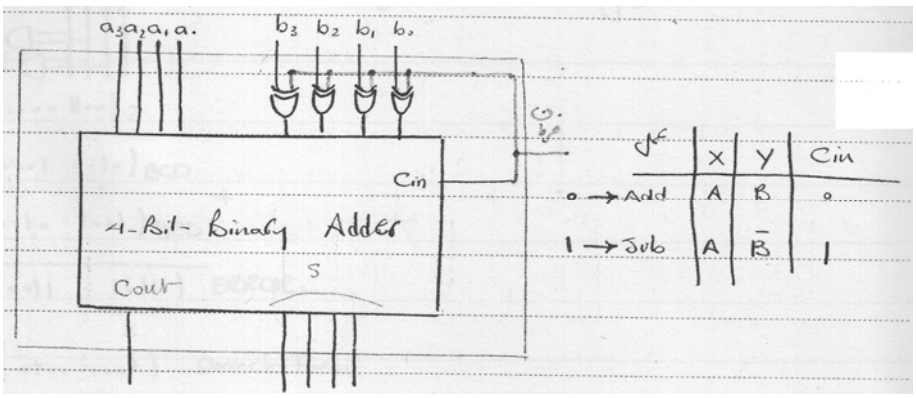
$B = YX'+ZX'+YZ$



جمع و تفریق کننده n بیتی : برای تفریق اعداد n بیتی به دو روش زیر می‌توان عمل نمود :

۱. با استفاده از تمام تفریق کننده ها تفریق کننده n بیتی طراحی می‌کنیم. در این روش، مراحل کار مشابه مراحل طراحی جمع کننده های n بیتی با استفاده از تمام جمع کننده ها می‌باشد.

۲. در روش دوم با استفاده از جمع کننده ها و انجام عمل تفریق به روش متمم ۲، می‌توان مدارات جمع کننده - تفریق کننده n بیتی ساخت. در این مدار با یک ورودی کنترل می‌توان تعیین نمود که دو عدد n بیتی با هم جمع شوند و یا از هم تفریق گردند.



همان گونه که در دیاگرام بلوکی فوق مشاهده می‌شود، بیت‌های عدد دوم (B) ابتدا وارد گیت های xor شده و با یک ورودی کنترل یا فرمان xor می‌شوند که این ورودی کنترل به عنوان نقلی ورودی، وارد مدار تمام جمع کننده نیز می‌شود. حال با توجه به مقادیر مختلف بیت کنترل (۰ یا ۱) این مدار می‌تواند عمل جمع و یا تفریق به روش متمم ۲ را انجام دهد.

$$C=0 \Rightarrow S=A+B+0=A+B(Sum)$$

$$C=1 \Rightarrow S=A+B'+1=A-B(Subtract)$$

جمع کننده BCD : جمع کننده BCD، مداریست که دو رقم BCD چهار بیتی را با یکدیگر جمع می‌کند. با اتصال چند جمع کننده BCD به یکدیگر می‌توان جمع اعداد BCD چند رقمی را انجام داد.

برای طراحی جمع کننده BCD دو روش کلاسیک و غیر کلاسیک وجود دارد:

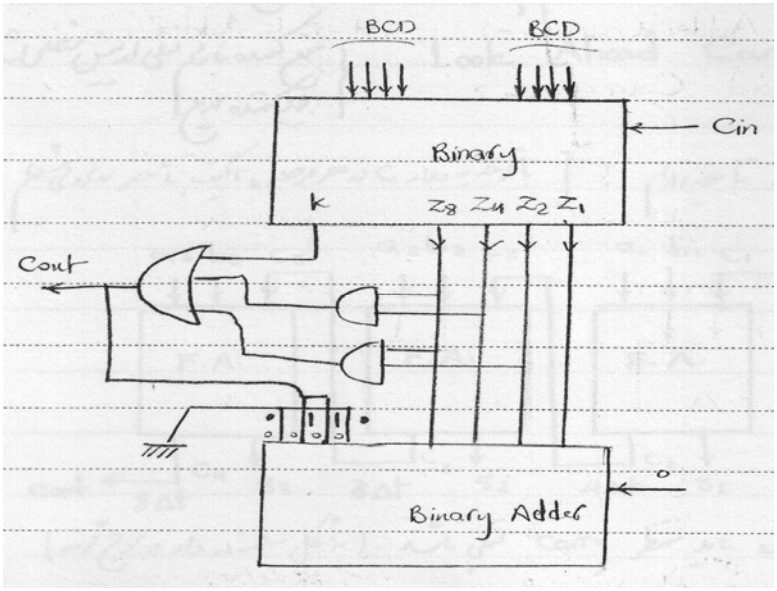
۱. یا باید به روش کلاسیک عمل کنیم و مراحل جدول صحت ، جدول کارنو و به دست آوردن روابط را طی کرده تا به مدار برسیم.در این صورت یک کارنو با ۲۵۶ ردیف لازم داریم که پیاده‌سازی آن و به دست آوردن عبارت های ساده شده خروجی ها مشکل خواهد بود.
۲. در روش غیر کلاسیک می‌توانیم از جمع کننده دودویی استفاده کرده و جمع اعداد BCD را به صورت دودویی انجام دهیم. در این صورت خروجی جمع کننده دودویی چهار بیتی طبق جدول زیر خواهد بود.

DECIMAL	K	Z4	Z3	Z2	Z1	C	S8	S4	S2	S1
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	1
2	0	0	0	1	0	0	0	0	1	0
3	0	0	0	1	1	0	0	0	1	1
4	0	0	1	0	0	0	0	1	0	0
5	0	0	1	0	1	0	0	1	0	1
6	0	0	1	1	0	0	0	1	1	0
7	0	0	1	1	1	0	0	1	1	1
8	0	1	0	0	0	0	1	0	0	0
9	0	1	0	0	1	0	1	0	0	1
10	0	1	0	1	0	1	0	0	0	0
11	0	1	0	1	1	1	0	0	0	1
12	0	1	1	0	0	1	0	0	1	0
13	0	1	1	0	1	1	0	0	1	1
14	0	1	1	1	0	1	0	1	0	0
15	0	1	1	1	1	1	0	1	0	1
16	1	0	0	0	0	1	0	1	1	0
17	1	0	0	0	1	1	0	1	1	1
18	1	0	0	1	0	1	1	0	0	0
19	1	0	0	1	1	1	1	0	0	1

این خروجی در برخی موارد مستلزم اصلاح است. به عبارت دیگر، هرگاه حاصل جمع دو عدد BCD بزرگتر از ۹ شود، نیاز به اصلاح داشته و حاصل باید با عدد ۶ دودویی $(0110)_2$ جمع شود. دلیل جمع با ۶ این است که اختلاف بین یک رقم نقلی در با ارزش‌ترین مکان بیتی حاصل از جمع دودویی و نقلی دهدهی برابر است با : 16-10 = 6. رابطه مدار اصلاح از جدول کارنو یا به روش سعی و خطا به دست می‌آید. در جدول فوق، C همان سیگنال اصلاح خروجی است که هرگاه برابر 1 شود، خروجی جمع کننده دودویی اول با یک ۶ دودویی $(0110)_2$ جمع خواهد شد. طبق جدول، عبارت جبری C برابر خواهد شد با :

$$C = K + Z4.Z2 + Z4.Z3$$

شکل زیر دیاگرام بلوکی یک جمع کننده BCD چهار بیتی را نشان می‌دهد. در این مدار اگر حاصل مدار اصلاح 0 باشد نتیجه با 0000 و اگر 1 باشد نتیجه با 0110 جمع می شود.

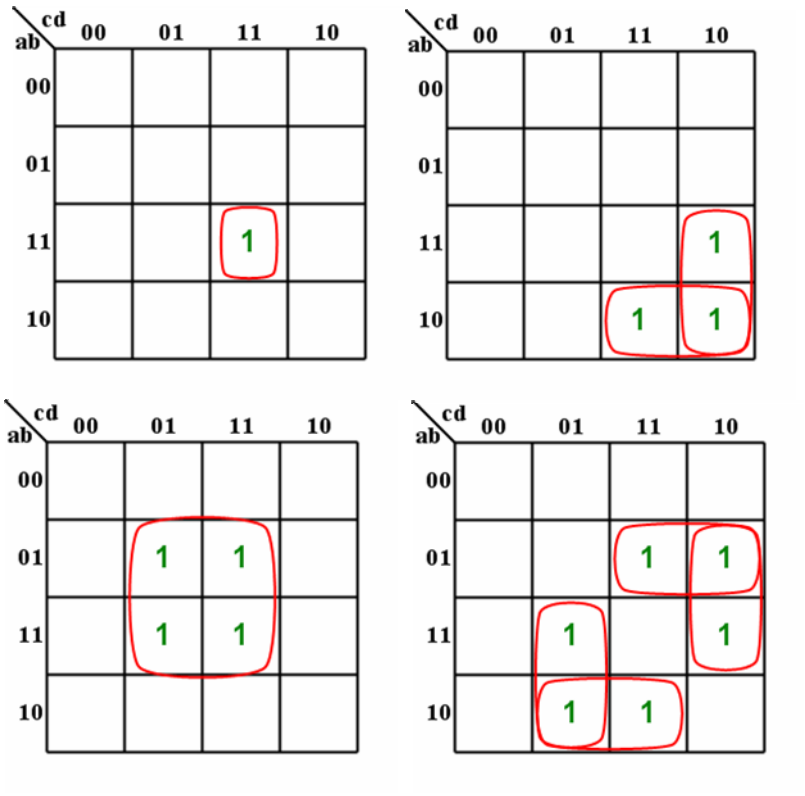


ضرب کننده ها : یکی دیگر از مدارات محاسباتی پر کاربرد در سیستمهای دیجیتال، ضرب کننده ها هستند؛ طراحی ضرب کننده ها را نیز می‌توان به دو روش کلاسیک و غیر کلاسیک انجام داد. در ادامه به یک ضرب کننده ۲ بیتی را به هر دو روش طراحی خواهیم نمود.

۱. روش کلاسیک : در این روش ابتدا دیاگرام بلوکی مدار را رسم کرده و تعداد ورودی ها و خروجی های مدار را مشخص می‌کنیم.

سپس برای پر کردن جدول صحت مدار، تمام ترکیبات ورودی را لیست کرده و به ازای هر ترکیب، حاصل ضرب دو بیت را در دو بیت دیگر بدست آورده و به صورت دودویی در ستون خروجی می نویسیم. در مرحله آخر نیز معادلات خروجی های مدار را به دست می‌آوریم.

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



F1=ABCD

F2=ACD'+ABC'

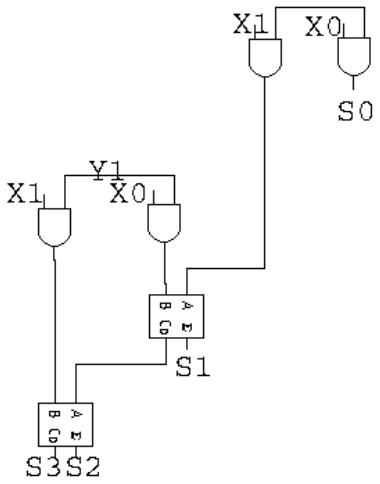
F3=AC'D+AB'D+BCD'+A'BC

F4=BD

۲. روش غیر کلاسیک : در روش غیر کلاسیک، هر مساله راه حل خاص خود را دارد و هیچ راه حلی برای مسائل دیگر عمومیت ندارد. برای طراحی مدار ضرب کننده سعی می‌کنیم روش دستی ضرب را با ابزارهای منطقی پیاده سازی کنیم، بدون اینکه به مراحلی چون جدول صحت نیاز داشته باشیم. عمل ضرب دو بیت در ریاضی معادل عبارت x AND y در منطقی است. ضرب متعارف دستی به صورت زیر انجام می‌شود :

			X1 X0
			Y1 Y0
		X1.Y0	X0.Y0
	X1.Y1	X1.Y1	0
S3	S2	S1	S0

با الگو گرفتن از روش ضرب دستی و استفاده از مدارات جمع کننده و گیت های AND مدار ضرب کننده مورد نظر را طراحی می‌کنیم که دیاگرام بلوکی آن در شکل زیر مشاهده می‌شود:



با به کارگیری روش غیر کلاسیک در طراحی ضرب کننده ها، می‌توان ضرب کننده‌های n بیت در m بیت را بدون نیاز به جداول صحت با تعداد سطرهای زیاد به سادگی طراحی نمود.

نام درس : مقایسه گرها، مدارهای مولد بیت توازن و مدارهای ترکیبی ماجولار

هدف از این فصل آشنایی با مفاهیم زیر می‌باشد :

- آشنایی با مفهوم مقایسه گر مقدار و ساخت مقایسه گرهای یک بیتی و چند بیتی
- مدارهای تولید و تست خطای توازن
- آشنایی با مدارات ترکیبی ماجولار، خواص و کاربرد آنها

مقایسه گر مقدار: یک مقایسه گر مقدار مداری ترکیبی است که دو عدد A و B را مقایسه می‌کند و اندازه نسبی آنها را تعیین می‌کند. نتیجه این مقایسه می‌تواند با سه متغیر خروجی (A=B)، (A>B) و (A<B) مشخص گردد. در برخی مدارات مقایسه کننده مقدار، تنها تحقیق تساوی دو عدد مطلوب است و نه روابط بزرگتر و کوچکتر بودن.

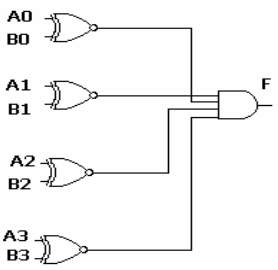
در طراحی مقایسه گرهای مقدار، معمولاً گیت های XOR و XNOR کاربرد زیادی دارند. در ادامه در نظر است یک مقایسه کننده چهاربیتی با استفاده از گیت های XOR و همچنین XNOR طراحی شود که دو عدد را با هم مقایسه کرده و در صورت مساوی بودن خروجی یک و در غیر این صورت خروجی صفر شود.

طراحی مقایسه کننده با XNOR :

A = A3A2A1A0　　B = B3B2B1B0

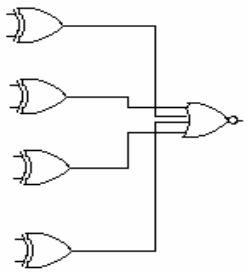
A	B	XNOR
0	0	1
0	1	0
1	0	0
1	1	1

در واقع XNORیک مقایسه کننده بیتی است یعنی دو بیت را با هم مقایسه میکند



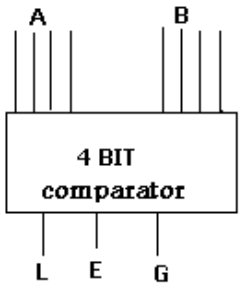
طراحی مقایسه کننده با XOR :

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



مقایسه کننده ۴ بیتی کامل : می‌خواهیم یک مقایسه کننده ۴ بیتی طراحی کنیم که دارای خروجی‌های کوچکتر، مساوی و بزرگتر باشد. این مدار دارای ۳ خروجی E، G و L است که به ترتیب بیانگر حالت های (A=B)، (A>B) و (A<B) می‌باشند.

دیاگرام بلوکی و معادلات خروجی‌های مدار در زیر مشاهده می‌شوند. برای به دست آوردن معادلات خروجی نهایی، توابع میانی X_i تعریف شده‌اند که هر X_i از رابطه $X_i = A_i \text{ xor } B_i$ به دست آمده و تساوی دو بیت متناظر از اعداد A و B را تحقیق می‌کند.



IF (L = 0) then (A > = B) , IF (L = 1) then (A < B)
IF (E = 0) then (A # B) , IF (E =1) then (A = B)
IF (G = 0) then (A < B) , IF (G = 1) then (A > B)

L = A'3B3 + X'3A'2B2 + X'3X'2A'1B1 + X'3X'2X'1A'0B0
G = A3B'3 + X'3A2B'2 + X'3X'2A1B'1 + X'3X'2X'1A0B'0
E = X'3X'2X'1X'0

X3 = A3 XOR B3
X2 = A2 XOR B2
X1 = A1 XOR B1
X0 = A0 XOR B0

مدار تولید و تست خطای توازن : در طراحی این مثال قصد داریم برای یک کد n بیتی، مدار تولید بیت توازن زوج و همچنین مدار تست خطای توازن را طراحی نماییم. بیت توازن زوج، بیتی است که به بیت های کد ارسالی اضافه شده و تعداد یک ها را در کد حاصل زوج خواهد نمود.

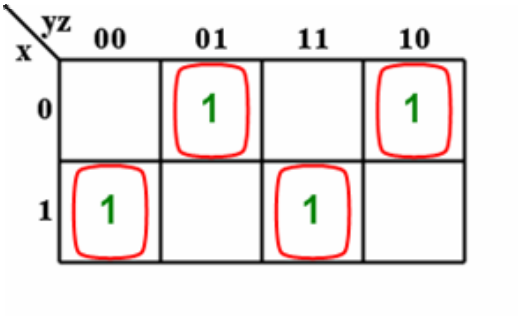
در مقصد نیز گیرنده بیت های کد ارسالی را به همراه بیت توازن دریافت می کند.اگر تعداد کل یک های دریافتی زوج باشد، کد ارسالی درست است، در غیر این صورت خطائی در کد رخ داده است. این روش فقط وجود خطا را تشخیص داده و توان اصلاح آنرا ندارد.

در ادامه مدارات مذکور را برای یک کد ۳ بیتی طراحی می‌نمائیم. در شکل های زیر، P بیت توازن زوج تولید شده در فرستنده و E بیت تشخیص خطای توازن تولید شده در گیرنده می‌باشند.

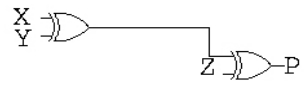
X	Y	Z	P
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

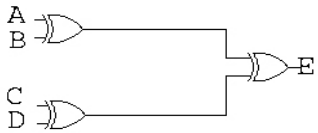
P = X xor Y xor Z

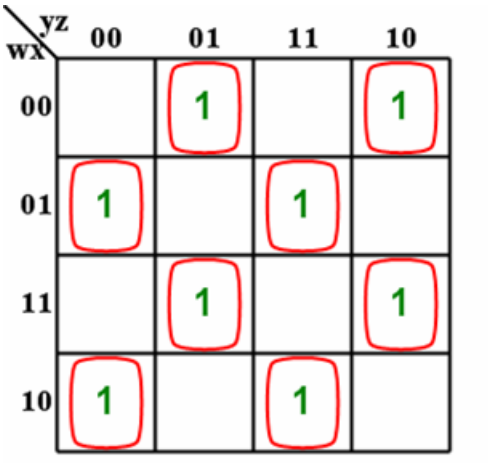
E = A xor B xor C xor D



A	B	C	D	E
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







توجه : توازن زوج NOT توازن فرد است، پس کافیست برای به دست آوردن توازن فرد، خروجی مدار توازن زوج را NOT کنیم.

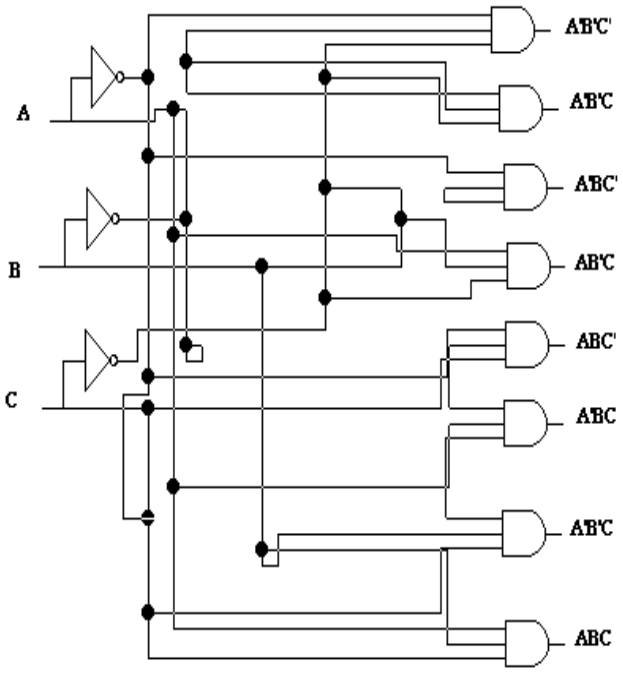
مدارهای ترکیبی ماجولار : گیت ها و مدارهای منطقی برای استفاده در طراحی‌ها به صورت قطعات مدارات مجتمع (IC) در اختیار طراحان قرار می‌گیرند. دسته‌ای از این مدارات که به مدارات ترکیبی ماجولار معروفند، کاربرد زیادی در طراحی سیستم های دیجیتال از جمله واحد کنترل CPU ها دارند. از جمله این مدارات می-توان به دیکدرها، اِنکدرها، مالتی پلکسرها، دی‌مالتی پلکسرها و ابزارهای قابل برنامه ریزی اشاره نمود. در ادامه به معرفی دیکدرها و مالتی پلکسرها، ویژگی ها و کاربردهای آن ها خواهیم پرداخت.

مدار رمز گشا یا دیکدر : دیکدر مداری ترکیبی است که اطلاعات دودویی را از n خط ورودی به حداکثر 2^n خط خروجی منحصر به فرد تبدیل می‌کند. کاربرد دیکدرها، تولید 2^n مینترم از n متغیر ورودی است. در دیکدرها در هر زمان حداکثر یک خروجی فعال خواهد بود.
به عنوان مثال، یک دیکدر با ۳ ورودی دارای ۸ خروجی است و جدول درستی آن در زیر مشاهده می‌شود.

A	B	C	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0		
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

طراحی دیکدر ۸×۳ : در دیکدر در هر لحظه، به ازای هر ترکیب ورودی فقط یک خروجی فعال است. خروجی های دیکدر در واقع همان مینترم‌ها هستند.

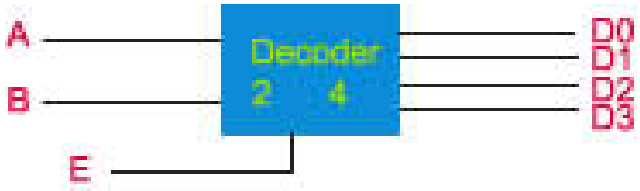
روابط خروجی‌ها و نمودار مداری یک دیکدر ۸×۳ در شکل زیر مشاهده می شود.

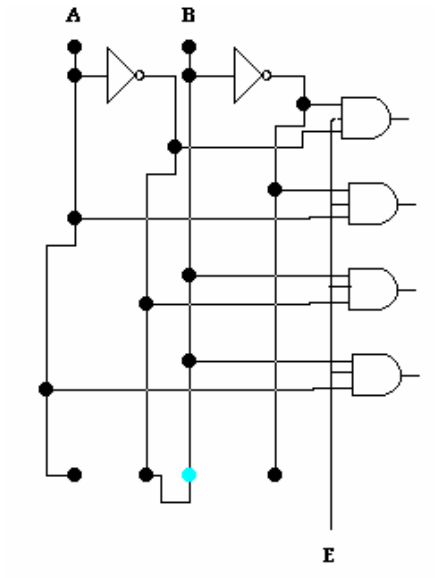


ورودی‌های فعال ساز : اکثر دیکدرها و مدارات ماجولار دارای یک یا چند ورودی فعال ساز هستند تا بتوان آن ها را داخل مدارات بزرگتر استفاده نموده و با سیگنال های کنترلی لازم، فعال یا غیر فعال نمود. این ورودی ها از دو نوع Active High و Active Low می‌باشند که نوع اول با 1 و نوع دوم با 0 فعال می‌شوند.

دیکدر ۲×۴ با ورودی کنترل Active High : در این مدار زمانی که ورودی فعال ساز برابر 1 است، دیکدر فعال شده و مینترم متناظر با ترکیب ورودی را در خروجی فعال خواهد نمود. دیاگرام بلوکی، جدول صحت دیکدر به همراه نمودار مداری آن در شکل مشاهده می‌شوند.

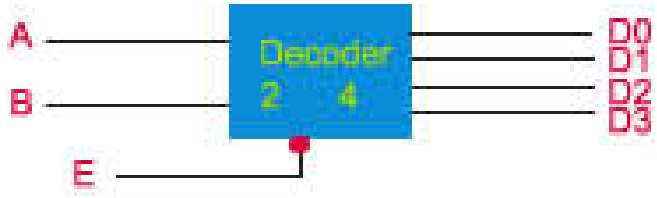
E	A	B	D3	D2	D1	D0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

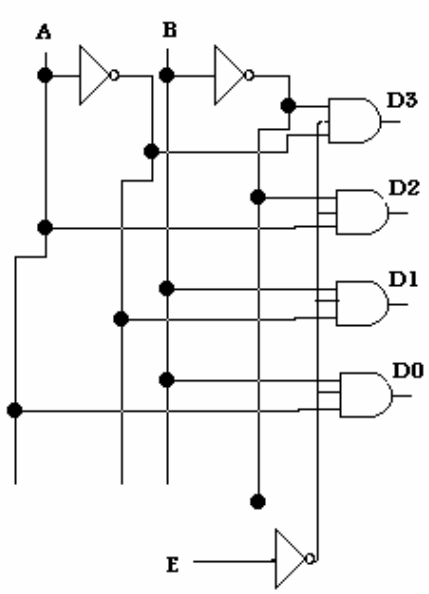




دیکدر ۲×۴ با ورودی کنترل Active Low : در این مدار زمانی که ورودی فعال ساز برابر 0 است، دیکدر فعال شده و مینترم متناظر با ترکیب ورودی را در خروجی فعال خواهد نمود. دیاگرام بلوکی، جدول صحت دیکدر به همراه نمودار مداری آن در شکل مشاهده می‌شوند.

E	A	B	D3	D2	D1	D0
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	1
1	X	X	1	1	1	1





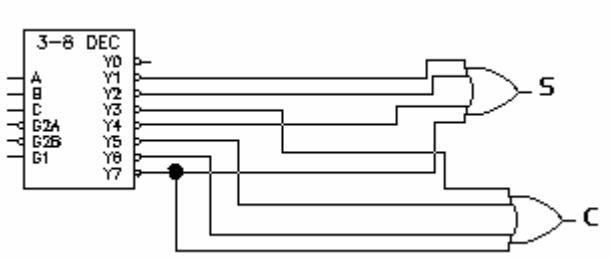
پیاده سازی توابع منطقی با استفاده از دیکدرها : هر تابع جبر بول را می‌توان با استفاده از دیکـدر و گیـت های مناسب پیاده سازی نمود. اما ابتدا باید تابع به فرم استاندارد جمع حاصل ضرب‌ها تبدیل شود.

به عنوان مثال یک تمام جمع کننده را می‌توان با یک دیکدر ۳×۸ و دو عدد گیت OR پیاده سازی کرد.

$S(x,y,z) = \sum m(1,2,4,7)$

$C(x,y,z) = \sum m\ (3,5,6,7)$

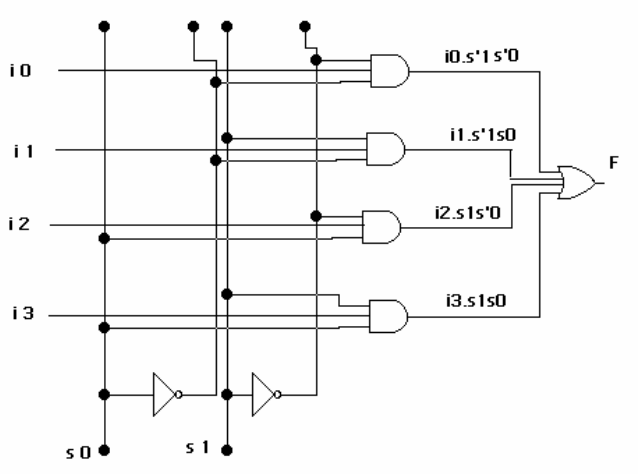
نمودار مداری تمام جمع کننده با استفاده از دیکدر در شکل زیر مشاهده می‌شود.



مدار تسهیم کننده یا مالتی پلکسر: مالتی پلکسر یک مدار انتخاب گر است که از چند ورودی فقط یکی از آنها را انتخاب کرده و به خروجی انتقال می دهد. یک مالتی پلکسر دارای n خط انتخاب، 2^n ورودی و یک خط خروجی می‌باشد که با توجه به مقدار ورودی های انتخاب در هر لحظه، یکی از ورودی ها انتخاب شده و در خروجی مالتی پلکسر ظاهر می‌شود.

دیاگرام بلوکی، جدول عملکرد و نمودار مدار یک مالتی پلکسر ۱×۴ را ملاحظه می‌فرمایید :

S0	S1	F
0	0	I0
0	1	I1
1	0	I2
1	1	I3



پیاده سازی توابع منطقی با استفاده از مالتی پلکسرها : هر تابع جبر بول را می‌توان با استفاده از مالتی پلکسر مناسب و گیت های لازم پیاده سازی نمود. برای این منظور بهتر است که تابع ابتدا در جدول درستی قرار داده شود. به عبارت دیگر می‌توان گفت که هر تابع n متغیره را به راحتی می‌توان با یک مالتی پلکسر با n-1 ورودی انتخاب پیاده سازی نمود. مالتی پلکسرهای مناسب برای پیاده سازی توابع ۳ تا ۵ متغیره را در زیر ملاحظه می‌فرمایید.

برای تابع سه متغیره = MUX 4 :1

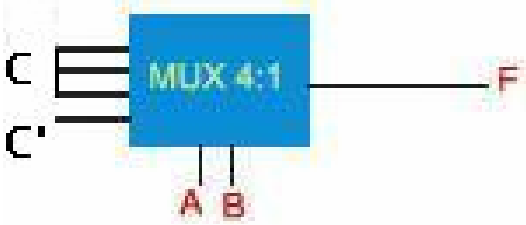
برای تابع چهار متغیره = MUX 8 :1

برای تابع پنج متغیره = MUX 16 :1

مثال : تابع زیر را با مالتی پلکسر مناسب پیاده سازی کنید.

$$F(A,B,C) = \sum(1,3,5,6) = A'B'C + A'BC + AB'C + ABC'$$

با فرض اینکه A و B را به ورودی های انتخاب مالتی پلکسر متصل کنیم، خواهیم داشت :



	I0	I1	I2	I3
C	0	2	4	6*
C'	1*	3*	5*	7*
	C	C	C	C'

حال با فرض اینکه B و C را به ورودی های انتخاب مالتی پلکسر متصل کنیم، خواهیم داشت :