سومین ویرایش

انتشارات خراسان

طراحی دیجیتال

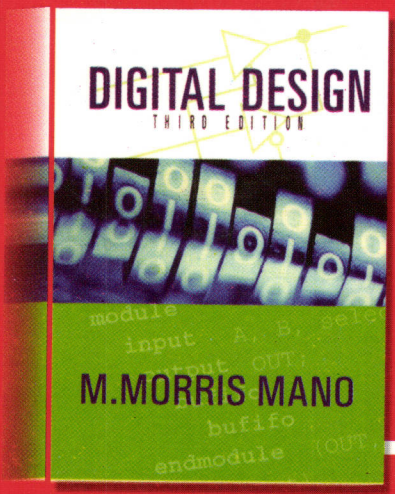
(مدار منطقی)

مؤلف : موریس مانو

مترجم : دکتر قدرت سپیدنام



همراه با CD رایگان



بسم الله الرحمن الرحیم

طراحی دیجیتال (مدار منطقی)

سومین ویرایش

پروفسور موریس مانو

دانشگاه ایالت کالیفرنیا، لُس آنجلس

دکتر قدرت سپیدنام

دانشکده مهندسی دانشگاه فردوسی مشهد

انتشارات نص

مانو، موریس ۱۹۲۷ - Mano. M. Morris
طراحی دیجیتال (مدار منطقی) / مؤلف موریس مانو؛ مترجم قدرت سپیدنام،
مشهد: انتشارات خراسان ، ۱۳۴۷ .
۵۳۶ ص . مصور، جدول، نمودار .
ISBN : 964-6342-Q-1
فهرستنویسی بر اساس اطلاعات فیپا (فهرست نویسی پیش از انتشار).
Digital Design
این کتاب در سال ۱۳۶۹ توسط همین ناشر در دو جلد منتشر شده است.
کتابنامه .
چاپ اول : ۱۳۸۱
۱. کامپیوترهای رقمی. - مدارها. ۲ . مدارهای منطقی، ۳. طراحی منطقی . ۴. مدارهای مجتمع رقمی. الف. سپیدنام، قدرت، مترجم. ب . عنوان .
۶ط ۲م ۴/TKv۸۸۸ ۶۲۱/۳۵۹

انتشارات خراسان

طراحی دیجیتال (مدار منطقی)

نوشته : موریس مانو
ترجمه : دکتر قدرت سپیدنام
چاپ سوم، زمستان ۸۲ ، ۱۰۰۰۰ نسخه، انتشارات خراسان
چاپخانه سروش
حق چاپ برای ناشر محفوظ است

قیمت با CD : ۳۰۰۰ تومان

«فهرست مطالب»

فصل اول : سیستم‌های دودویی ۱

۱-۱ سیستم‌های دیجیتال ۱
۱-۲ اعداد دودویی ۳
۱-۳ تبدیل مبنای اعداد ۵
۱-۴ اعداد مبنای هشت و شانزده ۸
۱-۵ متمم‌ها ۱۰
۱-۶ اعداد دودویی علامت‌دار ۱۴
۱-۷ کدهای دودویی ۱۸
۱-۸ ذخیره‌سازی دودویی و ثبات‌ها ۲۶
۱-۹ منطق دودویی ۲۹
مسائل ۳۳

فصل دوم : جبر بول و گیت‌های منطقی ۳۶

۲-۱ تعاریف اولیه ۳۶
۲-۲ تعریف اصول اساسی جبر بول ۳۸
۲-۳ قضایای اصلی و خواص جبر بول ۴۰
۲-۴ توابع بول ۴۳
۲-۵ فرم‌های استاندارد و متعارف ۴۷
۲-۶ دیگر اعمال منطقی ۵۴
۲-۷ گیت‌های منطقی دیجیتال ۵۶
۲-۸ مدارهای مجتمع ۶۲
مسائل ۶۵

فصل سوم : حداقل سازی در سطح گیت ۶۸

۳-۱ روش نقشه ۶۸
۳-۲ نقشه چهار متغیره ۷۴
۳-۳ نقشه پنج متغیره ۷۸
۳-۴ ساده‌سازی با ضرب حاصل جمع‌ها ۸۰
۳-۵ حالات بی‌اهمیت ۸۳
۳-۶ پیاده‌سازی با NAND و NOR ۸۶
۳-۷ دیگر پیاده‌سازی‌های دو سطحی ۹۳

۳-۸ تابع OR انحصاری ۹۸
۳-۹ زبان توصیف سخت‌افزاری (HDL) ۱۰۳
مسائل ۱۱۰

فصل چهارم : منطق ترکیبی ۱۱۵

۴-۱ مدارهای ترکیبی ۱۱۵
۴-۲ روش تحلیل ۱۱۶
۴-۳ روش طراحی ۱۱۹
۴-۴ جمع‌کننده - تفریق‌گر دودویی ۱۲۲
۴-۵ جمع‌کننده دهدهی ۱۳۳
۴-۶ ضرب دودویی ۱۳۵
۴-۷ مقایسه‌گر مقدار ۱۳۶
۴-۸ دیکدرها ۱۳۹
۴-۹ انکدرها ۱۴۳
۴-۱۰ مولتی پلکسرها ۱۴۶
۴-۱۱ HDL برای مدارهای ترکیبی ۱۵۲
مسائل ۱۶۷

فصل پنجم : مدارهای منطقی ترتیبی همزمان ۱۷۳

۵-۱ مدارهای ترتیبی ۱۷۳
۵-۲ لچ‌ها ۱۷۵
۵-۳ فلیپ فلاپ‌ها ۱۷۹
۵-۴ تحلیل مدارهای ترتیبی ساعت‌دار ۱۸۶
۵-۵ HDL برای مدارهای ترتیبی ۱۹۷
۵-۶ کاهش و تخصیص حالت ۲۰۶
۵-۷ روش طراحی ۲۱۱
مسائل ۲۱۹

فصل ششم : ثبّات و شمارنده‌ها ۲۲۴

۶-۱ ثبّات‌ها ۲۲۴
۶-۲ شیفت رجیسترها ۲۲۶
۶-۳ شمارنده‌های موج‌گونه ۲۳۴
۶-۴ شمارنده‌های همزمان ۲۳۹
۶-۵ دیگر شمارنده‌ها ۲۴۶

۶-۶ HDL برای ثبات‌ها و شمارنده‌ها ۲۵۱
مسائل ۲۵۷

فصل هفتم: حافظه و منطق برنامه پذیر ۲۶۱

۷-۱ مقدمه ۲۶۱
۷-۲ حافظه با دستیابی تصادفی RAM ۲۶۲
۷-۳ دیکد کردن حافظه ۲۶۹
۷-۴ تشخیص و تصحیح خطا ۲۷۴
۷-۵ حافظه فقط خواندنی ROM ۲۷۷
۷-۶ آرایه منطقی برنامه‌پذیر PLA ۲۸۳
۷-۷ منطق آرایه‌ای برنامه‌پذیر PAL ۲۸۷
۷-۸ وسایل برنامه‌پذیر ترتیبی ۲۹۰
مسائل ۲۹۵

فصل هشتم: سطح انتقال ثباتی ۲۹۹

۸-۱ نمایش در سطح انتقال ثباتی ۲۹۹
۸-۲ سطح انتقال ثباتی (RTL) ۳۰۱
۸-۳ ماشین‌های حالت الگوریتمی (ASM) ۳۰۸
۸-۴ مثال طراحی ۳۱۳
۸-۵ توصیف HDL مثال طراحی ۳۲۰
۸-۶ ضرب کننده دودویی ۳۲۶
۸-۷ مدار کنترل ۳۳۱
۸-۸ توصیف HDL ضرب کننده دودویی ۳۳۵
۸-۹ طراحی با مولتی پلکسر ۳۴۰
مسائل ۳۴۶

فصل نهم: مدارهای ترتیبی غیرهمزمان ۳۵۱

۹-۱ مقدمه ۳۵۱
۹-۲ روش تحلیل ۳۵۳
۹-۳ مدارهای لچ‌دار ۳۶۱
۹-۴ روش طراحی ۳۶۹
۹-۵ کاهش جداول حالت و روند ۳۷۶
۹-۶ تخصیص حالت فاقد رقابت ۳۸۴
۹-۷ هزارد (HAZARD) ۳۸۹

۸-۹ مثال طراحی ۳۹۴
مسائل ۴۰۰

فصل دهم: مدارهای مجتمع دیجیتال ۴۰۶

۱-۱۰ مقدمه ۴۰۶
۲-۱۰ مشخصات ویژه ۴۰۸
۳-۱۰ مشخصه‌های ترانزیستور دو قطبی ۴۱۳
۴-۱۰ مدارهای RTL و DTL ۴۱۶
۵-۱۰ منطق ترانزیستور- ترانزیستور TTL ۴۱۹
۶-۱۰ منطق کوپلاژ امیتر ECL ۴۲۹
۷-۱۰ فلز -اکسید- نیمه‌هادی (MOS) ۴۳۱
۸-۱۰ MOS متمم (CMOS) ۴۳۴
۹-۱۰ مدارهای گیت انتقال CMOS ۴۳۷
۱۰-۱۰ مدل‌سازی سطح- سوئیچ با HDL ۴۴۰
مسائل ۴۴۴

فصل یازدهم: تمرینات آزمایشگاهی ۴۴۷

۰-۱۱ مقدمه‌ای بر آزمایش‌ها ۴۴۷
۱-۱۱ اعداد دودویی و دهدهی ۴۵۱
۲-۱۱ گیت‌های منطقی دیجیتال ۴۵۴
۳-۱۱ ساده‌سازی توابع بول ۴۵۶
۴-۱۱ مدارهای ترکیبی ۴۵۸
۵-۱۱ مبدل‌های کد ۴۶۰
۶-۱۱ طراحی با مولتی پلکسر ۴۶۱
۷-۱۱ جمع و تفریق‌گر ۴۶۳
۸-۱۱ فلیپ فلاپ‌ها ۴۶۶
۹-۱۱ مدارهای ترتیبی ۴۶۸
۱۰-۱۱ شمارنده‌ها ۴۷۰
۱۱-۱۱ شیفت رجیستر ۴۷۱
۱۲-۱۱ جمع سریال ۴۷۵
۱۳-۱۱ واحد حافظه ۴۷۶
۱۴-۱۱ هندبال لامپی ۴۷۸
۱۵-۱۱ مولد پالس ساعت ۴۸۲
۱۶-۱۱ جمع‌کننده موازی و انباره ۴۸۴

۱۱-۱۷ ضرب کننده دودویی ۴۸۶
۱۱-۱۸ مدارهای ترتیبی غیرهمزمان ۴۸۹
۱۱-۱۹ آزمایش‌های شبیه‌سازی VERILOG HDL ۴۹۰

فصل دوازدهم : سمبل‌های گرافیکی استاندارد ۴۹۵

۱۲-۱ سمبل‌های مستطیلی شکل ۴۹۵
۱۲-۲ سمبل‌های مصوب ۴۹۸
۱۲-۳ نشانه وابستگی ۵۰۰
۱۲-۴ سمبل‌های عناصر ترکیبی ۵۰۲
۱۲-۵ سمبل فلیپ فلاپ‌ها ۵۰۴
۱۲-۶ سمبل ثبات‌ها ۵۰۶
۱۲-۷ سمبل شمارنده‌ها ۵۰۹
۱۲-۸ سمبل RAM ۵۱۱
مسائل ۵۱۲

پاسخ به مسائل انتخابی ۵۱۴

فصل ۱ ..... ۵۱۴
فصل ۲ ..... ۵۱۵
فصل ۳ ..... ۵۱۵
فصل ۴ ..... ۵۱۶
فصل ۵ ..... ۵۱۸
فصل ۶ ..... ۵۱۹
فصل ۷ ..... ۵۲۱
فصل ۸ ..... ۵۲۱
فصل ۹ ..... ۵۲۳
فصل ۱۰ ..... ۵۲۴

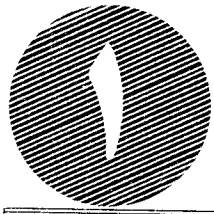
واژه نامه ۵۲۵

سخن ناشر

"این کتاب گفتاری جدید درباره طراحی دیجیتال است و روشهای مناسبی برای طراحی مدارهای دیجیتال را به گونه‌ای روان، ساده و دقیق به خواننده می‌آموزد"

این گفتار کوتاه ، نظر مؤلف درباره کتاب است. درستی سخن مؤلف را استادان و دانشجویان ایرانی کتاب "طراحی دیجیتال" از همان سال نگارش (۱۹۸۴) تاکنون با انتخاب این کتاب و معرفی آن به خوانندگان جدید، تأیید کرده‌اند.

انتشارات خراسان ترجمه کتاب Digital design را پس از چاپ نخستین ویرایش آن با ترجمه سلیس و دقیق آقای دکتر قدرت سپیدنام استاد دانشکده مهندسی دانشگاه مشهد به خوانندگان عرضه کرده است و پس از هر بار تجدید نظر مؤلف، تجدید نظر شده آن را با ترجمه شیوای مترجم به چاپ رسانده است. اینک ترجمه سومین ویرایش آن با چاپ دو رنگ به همراه CD به اساتید و متخصصان و دانشجویان عرضه می‌شود. امید است ضمن در نظر داشتن تلاشی که مصروف آن گشته، از راهنمایی‌های ارزنده خوانندگان صاحب‌نظر بی‌نصیب نمانیم.



سیستم‌های دودویی

۱-۱ سیستم‌های دیجیتال

سیستم‌های دیجیتال در زندگی روزانه بشر نقش برجسته‌ای دارند و به این دلیل ما دوره تکنولوژی فعلی را عصر دیجیتال می‌خوانیم. سیستم‌های دیجیتال در مخابرات، تجارت، کنترل ترافیک، هدایت سفینه‌های فضایی، اعمال جراحی، هواشناسی، اینترنت و بسیاری از دیگر زمینه‌های تجاری، صنعتی و علمی به کار می‌روند. ما از تلفن‌های دیجیتال، تلویزیون‌های دیجیتال، دیسک‌های چند منظوره دیجیتال، دوربین‌های دیجیتال، و البته کامپیوترهای دیجیتال استفاده می‌کنیم. مهمترین خاصیت یک کامپیوتر دیجیتال، همگانی بودن آن است. کامپیوتر می‌تواند رشته‌ای از دستورات به نام برنامه را که روی داده‌های مفروض عمل می‌کنند، دنبال نماید. کاربر می‌تواند برنامه یا داده خود را طبق نیاز انتخاب و اجرا کند. به علت این انعطاف، کامپیوترهای همه منظوره دیجیتال می‌توانند عملیات پردازش اطلاعات را در محدوده وسیعی از کاربردها انجام دهند.

یکی از ویژگی‌های سیستم دیجیتال توانمندی آنها در دستکاری عناصر گسسته اطلاعاتی است. هر مجموعه‌ای که به تعداد متناهی از عناصر محدود باشد اطلاعات گسسته را داراست. مثال‌هایی از عناصر گسسته عبارتند از 10 رقم دهدهی، 26 حرف الفباء، 52 ورق بازی، 64 مربع بازی شطرنج. کامپیوترهای دیجیتال اولیه برای محاسبات عددی به کار می‌رفتند. در این حال، عناصر گسسته به کار رفته، ارقام بودند. نام دیجیتال یا رقمی از این مفهوم حاصل شده است. عناصر گسسته اطلاعاتی در یک سیستم دیجیتال با کمیت‌های فیزیکی به نام سیگنال نشان داده می‌شوند. رایج‌ترین سیگنال‌های الکتریکی عبارتند از ولتاژ و جریان. وسایل الکترونیکی به نام ترانزیستور در مداراتی که این سیگنال‌ها را پیاده‌سازی می‌کنند به طور چشمگیری به کار می‌روند. سیگنال‌ها در بسیاری از سیستم‌های دیجیتال الکترونیک امروزی تنها دو مقدار را دارا هستند و بنابراین گوییم دودویی‌اند. یک رقم دودویی که بیت خوانده می‌شود دو

مقدار دارد: 0 و 1 . عناصر گسسته اطلاعاتی با گروهی از بیت‌ها بـه نـام کـدهای دودویی نـمایش داده می‌شوند. مثلاً ارقام دهدهی 0 تا 9 در سیستم اعداد دیجیتال با کد چـهار بـیتی نشـان داده مـی‌شوند. با به کارگیری تکنیک‌های مختلف، گروههایی از بیت‌ها برای نمایش سمبل‌های گسسته تعریف می‌شوند و سپس در توسعه یک سیستم در قالب دیجیتال مورد استفاده قرار می‌گیرد. در نتیجه، یک سیستم دیجیتال سیستمی است که عناصر گسسته اطلاعاتی به شکل دودویی را در درون دستکاری می‌کند.

کمیت‌های اطلاعاتی یا ذاتاً گسسته‌اند و یا از نمونه‌برداری (کوانتیزه کردن) فرآیندهای پیوسته حاصل می‌شوند. به عنوان مثال یک لیست حقوق ذاتاً یک فرآیند یا رویداد گسسته بوده و حاوی نام کارمند، شماره تأمین اجتماعی، حقوق هفتگی، مالیات بردرآمد و غیره است. پرداختی به یک کارمند با استفاده از مقادیر داده گسسته مانند حروف الفبایی (نام‌ها)، ارقام (حقوق)، و نمادها یا سمبل‌های خاص (مانند $) پردازش می‌گردد. از طرف دیگر یک محقق ممکن است یک پدیده را به صورت پیوسته مشاهده کند، ولی فقط مقادیر خاصی را به صورت جدول ثبت نماید. بنابراین فرد محقق داده پیوسته را نمونه‌برداری می‌نماید ولی هر کمیت در جدول را از عناصر گسسته می‌سازد. در بسیاری از حالات نمونه‌برداری از یک فرآیند به طور خودکار به وسیله دستگاهی بنام مبدل آنالوگ به دیجیتال انجام می‌شود.

بهترین مثال از یک سیستم دیجیتال، کامپیوتر دیجیتال همه منظوره است. بخش‌های اصـلی یک کامپیوتر عبارتند از واحد حافظه، واحد پردازش مرکزی و واحدهای ورودی- خروجی. واحد حافظه برنامه‌ها و داده‌های وارده، خارج شـونده و مـیانی را ذخیره مـی‌کند. واحـد پـردازش مـرکزی اعـمال محاسباتی و دیگر عملیات روی داده‌ها را برحسب آنچه در برنامه مشخص شده، انجام می‌دهد. داده‌ها و برنامه‌هایی که به وسیله کاربر آماده شده‌اند توسط وسایل ورودی مانند صفحه کلید به حافظه انـتقال می‌یابند. یک وسیله خروجی مثل چاپگر نتایج حاصل از محاسبات را دریافت کرده و به کاربر ارائـه می‌دهد. یک کامپیوتر دیجیتال می‌تواند به چندین وسیله ورودی- خروجی وصل شود. یکی از وسایل مفید واحد مخابره است که تبادل داده را از طریق اینترنت با دیگر کاربران برقرار می‌سازد. یک کامپیوتر دیجیتال دستگاهی توانمند است که نه تنها می‌تواند محاسبات ریاضی را انجام دهد، بلکه قادر است اعمال منطقی را هم اجرا نماید. به علاوه می‌تواند جهت تصمیم‌گیری براساس شرایط داخلی یا خارجی برنامه‌ریزی شود.

برای اسـتفاده از مـدارهـای دیـجیتال در تـولیدات تـجاری دلایـل اسـاسی وجـود دارد. هـمچون کامپیوترهای دیجیتال، دستگاه‌های دیجیتال نیز قابل بـرنامه‌ریزی‌انـد. بـا تـعویض بـرنامه در وسـیله برنامه‌پذیر، سخت‌افزار یگانه‌ای قابل استفاده در کاربردهای متفاوت خواهد بود. کاهش قیمت شدید در وسایل دیجیتال به دلیل پیشرفت در تکنولوژی مدارهای مجتمع دیجیتال مرتباً روی می‌دهد. با افزایش تعداد ترانزیستورها در یک قطعه سیلیکان، توابع پیچیده‌تری قابل پیاده‌سازی شده، قیمت هـر واحـد کاهش یافته و قیمت دستگاه‌های دیجیتال روزبروز کاهش می‌یابد. دستگاه‌های ساخته شده با مدارهای مجتمع می‌توانند با سرعتی تا صد میلیون عمل در ثانیه را انجام دهند. می‌توان با اسـتفاده از کـدهای اصلاح خطا عملکرد سیستم‌های دیجیتال را به شدت اطمینان بخش نمود. مثالی از این نـوع، دیسک

چند کاره دیجیتال (DVD) است که در آن اطلاعات ویدیویی، صوتی و دیگر گونه‌ها بدون از دست رفتن یک قلم داده، ضبط می‌گردد. اطلاعات دیجیتال در DVD چنان ضبط می‌شود که هر کد در هر نمونه دیجیتال قبل از نمایش به طور خودکار خطایابی شده و اصلاح می‌گردد.

یک سیستم دیجیتال از به هم پیوستن مدول‌های دیجیتال بدست می‌آید. برای درک عمل هر مدول، دانش و آگاهی مدارهای دیجیتال و عمل منطقی آنها لازم است. هفت فصل اول این کتاب ابزار اصلی طراحی دیجیتال مانند ساختارهای منطقی گیتی، مدارهای ترکیبی و ترتیبی و وسایل منطقی برنامه‌پذیر را ارائه می‌کند. فصل ۸ طراحی دیجیتال را در سطح انتقال بین ثباتی (RTL) معرفی می‌نماید. فصل‌های ۹ و ۱۰ در مورد مدارهای ترتیبی غیرهمزمان (آسنکرون یا غیرهمگام) و دیگر خانواده‌های منطقی دیجیتال مجتمع بحث می‌کنند. فصل‌های ۱۱ و ۱۲ مدارهای مجتمع تجاری را معرفی کرده و نشان می‌دهند که چگونه در یک آزمایشگاه برای انجام آزمایشات به هم وصل می‌شوند.

یک گرایش مهم در طراحی دیجیتال، استفاده از زبان توصیف سخت‌افزاری (HDL) است. HDL نوعی زبان برنامه‌ریزی است که برای توصیف مدارهای دیجیتال به صورت متن به کار می‌رود. این زبان برای شبیه‌سازی یک سیستم دیجیتال و اطمینان از صحت عمل آن قبل از ساخت مورد استفاده قرار می‌گیرد. HDL در کنار ابزارهای طراحی منطقی دیگر برای خودکارسازی طراحی استفاده می‌شود. توصیف HDL مدارهای دیجیتال در سرتاسر این کتاب ارائه شده است.

همانطور که قبلاً گفته شد سیستم‌های دیجیتال کمیت‌های گسسته اطلاعات، که به فرم دودویی نمایش داده شده‌اند را دستکاری می‌نمایند. عملوندهای به کار رفته در محاسبات را می‌توان در سیستم اعداد دودویی بیان کرد. دیگر عناصر گسسته از جمله ارقام دهدهی به صورت کدهای دودویی نشان داده می‌شوند. پردازش داده به وسیله عناصر منطقی دودویی و با استفاده از سیگنال‌های دودویی انجام می‌گیرد. کمیات نیز در عناصر حافظه دودویی ذخیره می‌شوند. هدف این فصل معرفی مفاهیم دودویی متعدد به صورت یک مرجع برای مطالعات بعدی در فصل‌های آینده است.

۲-۱ اعداد دودویی

یک عدد دهدهی مانند 7,392 کمیتی معادل با 7 هزارتایی، بعلاوه 3 صدتایی، بعلاوه 9 ده‌تایی بعلاوه 2 واحد را نشان می‌دهد. هزارها، صدها و غیره توانی از 10 هستند که با توجه به مکان ضرایب معین می‌گردند. دقیق‌تر بگوییم، 7,392 را می‌توان به صورت زیر نوشت.

$$7 \times 10^3 + 3 \times 10^2 + 9 \times 10^1 + 2 \times 10^0$$

با این وجود عرف این است که فقط ضرایب را بنویسیم و توان‌های لازم 10 را از مکان آنها استنتاج کنیم. به طور کلی یک عدد با نقطه اعشاری با یک سری ضرایب به صورت زیر نمایش داده می‌شود.

$$a_5a_4a_3a_2a_1a_0 \cdot a_{-1}a_{-2}a_{-3}$$

که ضرایب a_j هر یک از ده رقم (9, ..., 2, 1, 0) بوده و j مکان عدد را نشان می‌دهد، و از این رو توان 10

که ضریب در آن ضرب می‌گردد مشخص خواهد شد. این مطلب به صورت زیر بیان می‌شود:

$$10^5a_5 + 10^4a_4 + 10^3a_3 + 10^2a_2 + 10^1a_1 + 10^0a_0 + 10^{-1}a_{-1} + 10^{-2}a_{-2} + 10^{-3}a_{-3}$$

سیستم‌اعداد دهدهی را در مبنای 10 گوییم زیرا از 10 رقم استفاده می‌کند و ضرایب در توانی از 10 ضرب می‌گردند. سیستم دودویی یک سیستم اعداد متفاوت است. ضرایب سیستم اعداد دودویی فقط دو مقدار ممکن را دارند: 0 و 1 . هر ضریب a_j در 2^j ضرب می‌گردد. مثلاً معادل دهدهی عدد دودویی 11010.11 برابر 26.75 می‌باشد، که از ضرب ضرایب در توان‌هایی از 2 بدست می‌آید:

$$1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} = 26.75$$

به طور کلی، یک عدد در مبنای r به صورت حاصلضرب توانهای r در ضرایب مربوطه‌اش بیان می‌گردد:

$$a_n \cdot r^n + a_{n-1} \cdot r^{n-1} + \ldots + a_2 \cdot r^2 + a_1 \cdot r + a_0 + a_{-1} \cdot r^{-1} + a_{-2} \cdot r^{-2} + \ldots + a_{-m} \cdot r^{-m}$$

که ضرایب a_j بین 0 تا $r-1$ می‌باشند. برای تفکیک اعداد در مبناهای مختلف، ضرایب را در داخل پرانتزها نوشته و اندیس مبنارا در زیر آن می‌گذاریم (به جز در اعداد دهدهی که محتوا بیانگر دهدهی بودن است). مثالی از عدد مبنای 5 چنین است

$$(4021.2)_5 = 4 \times 5^3 + 0 \times 5^2 + 2 \times 5^1 + 1 \times 5^0 + 2 \times 5^{-1} = (511.4)_{10}$$

مقدار ضرایب در مبنای 5 فقط 0، 1، 2، 3، 4 می‌باشند. سیستم اعداد هشت هشتی یک سیستم مبنای 8 با هشت رقم 0، 1، 2، 3، 4، 5، 6، 7 می‌باشد. مثالی از یک عدد مبنای هشت 127.4 است. برای تعیین مقدار معادل دهدهی، عدد را به صورت یک سری از توانها با مبنای 8 بسط می‌دهیم.

$$(127.4)_8 = 1 \times 8^2 + 2 \times 8^1 + 7 \times 8^0 + 4 \times 8^{-1} = (87.5)_{10}$$

توجه کنید که ارقام 8 و 9 نمی‌توانند در یک عدد هشت هشتی ظاهر شوند.

هنگامی که تعداد ارقام کمتر از 10 باشد مرسوم است که r رقم مورد نیاز برای ضرایب از سیستم دهدهی گرفته شود. هنگامی که مبنای عدد از 10 بزرگتر است از حروف الفبا برای تکمیل 10 رقم دهدهی استفاده می‌گردد. مثلاً در سیستم اعداد شانزده شانزدهی (مبنای 16)، ده رقم اول از سیستم دهدهی گرفته می‌شوند. حروف A، B، C، D، E و F به ترتیب به جای ارقام 10، 11، 12، 13، 14 و 15 به کار می‌روند. مثالی از یک عدد در مبنای 16 به صورت زیر است.

$$(B65F)_{16} = 11 \times 16^3 + 6 \times 16^2 + 5 \times 16^1 + 15 \times 16^0 = (46,687)_{10}$$

همانطورکه قبلاً اشاره شد، ارقام در یک عدد دودویی بیت خوانده می‌شوند. وقتی که یک بیت برابر 0 است در عمل جمع تبدیل مبنا نقشی ندارد. بنابراین تبدیل دودویی به دهدهی با جمع توان‌هایی از 2 که ضرایب آن 1 است صورت می‌گیرد. مثلاً،

$$(110101)_2 = 32 + 16 + 4 + 1 = (53)_{10}$$

در عدد فوق چهار عدد 1 دیده می‌شود. دهدهی مربوطه جمع چهار توان از 2 می‌باشد. 24 عدد اول حاصل از 2 به توان n در جدول ۱-۱ لیست شده‌اند. در کارهای کامپیوتری 2^{10} را K (کیلو)، 2^{20} را

جدول ۱-۱. توانهایی از 2

n	2n	n	2n	n	2n
0	1	8	256	16	65,536
1	2	9	512	17	131,072
2	4	10	1,024	18	262,144
3	8	11	2,048	19	524,288
4	16	12	4,096	20	1,048,576
5	32	13	8,192	21	2,097,152
6	64	14	16,384	22	4,194,304
7	128	15	32,768	23	8,388,608

M (مگـــا)، 2^{30} را G (گـــیگا) و 2^{40} را T (تـــرا) مـــی‌گویند. بـــنابرایـــن $4K = 2^{12} = 4096$ و $16M = 2^{24} = 16777216$ می‌باشد. معمولاً ظرفیت کامپیوتر به بایت داده می‌شود. یک بایت برابر با هشت بیت بوده و می‌تواند یک کاراکتر از صفحه کلید را در خود جای دهد. یک دیسک سخت (هارد) کامپیوتر با ظرفیت 4 گیگا دارای ظرفیت $4G = 2^{32}$ بایت می‌باشد.

اعمال حسابی با اعدادی در مبنای r از همان قواعد اعداد دهدهی استفاده می‌کنند. هنگامی که از مبنایی به جز 10 استفاده می‌شود باید دقت کرد که تنها r رقم مجاز به کار گرفته شود. در زیر مثال‌هایی برای جمع، تفریق و ضرب دودویی آورده شده است.

مضروب :	1011	مفروق منه :	101101	مضاف :	101101
مضروب فیه :	× 101	مفروق :	−100111	مضاف الیه :	+100111
	1011	باقیمانده :	000110	حاصل جمع :	1010100
	0000				
	1011				
حاصل ضرب :	10111				

جمع دو عدد دودویی مشابه قوانین دهدهی محاسبه می‌شود، به جز این که ارقام جمع در هر مکان باارزش فقط می‌تواند 0 یا 1 باشد. هر رقم نقلی حاصل در یک مکان مفروض، به وسیله جفت رقم‌های مرتبه بالاتر (باارزش‌تر) مورد استفاده قرار می‌گیرد. تفریق کمی پیچیده‌تر است. قوانین باز هـم هـمان قوانین دهدهی هستند، به جز این که قرض در یک مکان باارزش، 2 را به رقم مفروق منه مـی‌افـزایـد. (قرض در سیستم دهدهی، 10 واحد به رقم مفروق منه اضافه می‌کند.) عمل ضرب خیلی ساده است. ارقام مضروب فیه همیشه 1 یا 0 هستند. بنابراین حاصلضرب‌های جزیی برابر 0 یا برابر با مضروب می‌باشند.

۳-۱ تبدیل مبنای اعداد

همانطور که قبلاً مشاهده شد، تبدیل یک عدد در مبنای r به مبنای ده با بسط عدد به صورت یک سری از توانها و جمع همه جملات انجام می‌شود. اکنون برای تبدیل معکوس یک عدد دهدهی به یک عدد در مبنای r روالی کلی را ارائه می‌کنیم. اگر عدد حاوی نقطه ممیز باشد، لازم است تا عدد بـه دو

بخش صحیح و کسری تفکیک گردد زیرا هر بخش باید به طور جداگانه تبدیل شود. تبدیل یک عدد صحیح دهدهی به یک عدد در مبنای r با تقسیم عدد و همه خارج قسمت‌های متوالی بر r و جمع‌آوری باقیمانده‌ها انجام می‌گردد. بهتر است روال را با مثالی تشریح کنیم.

مثال ۱-۱

عدد 41 را به دودویی تبدیل کنید. ابتدا 41 را بر 2 تقسیم می‌کنیم تا خارج قسمت 20 و باقیمانده 1/2 بدست آید. خارج قسمت مجدداً بر 2 تقسیم می‌گردد تا خارج قسمت و باقیمانده جدیدی بدست آید. این روال تا رسیدن به خارج قسمت 0 ادامه می‌یابد. ضرایب عدد دودویی موردنظر به طریق زیر از باقیمانده‌ها بدست می‌آید:

	خارج قسمت صحیح		باقیمانده	ضریب عدد دودویی
$41/2 =$	20	+	$\frac{1}{2}$	$a_0 = 1$
$20/2 =$	10	+	0	$a_1 = 0$
$10/2 =$	5	+	0	$a_2 = 0$
$5/2 =$	2	+	$\frac{1}{2}$	$a_3 = 1$
$2/2 =$	1	+	0	$a_4 = 0$
$1/2 =$	0	+	$\frac{1}{2}$	$a_5 = 1$

جواب : $(41)_{10} = (a_5a_4a_3a_2a_1a_0)_2 = (101001)_2$

بنابراین، پاسخ $(101001)_2 = (a_5a_4a_3a_2a_1a_0) = (41)_{10}$ می‌باشد.

روال فوق را می‌توان به طریق ساده‌تر زیر دستکاری کرد:

خارج قسمت صحیح	باقیمانده
41	
20	1
10	0
5	0
2	1
1	0
0	1

جواب = 101001

تبدیل اعداد صحیح دهدهی به هر سیستم مبنای r مشابه مثال فوق است به جز این که تقسیم در عوض 2 بر r انجام می‌گردد.

❑

مثال ۲-۱

عدد 153 را به مبنای هشت ببرید. مبنای موردنظر r برابر 8 است. ابتدا 153 بر 8 تقسیم می‌شود تا خارج قسمت صحیح 19 و باقیمانده 1 حاصل گردد. سپس 19 بر 8 تقسیم می‌شود تا خارج قسمت 2 و باقیمانده 3 را بدست دهد. بالاخره 2 بر 8 تقسیم گردیده تا خارج قسمت 0 و باقیمانده 2 بدست آید. این روند به صورت مناسب زیر انجام می‌گردد:

153	
19	1
2	3
0	2

$= (231)_8$

در تبدیل قسمت کسری مبنای ده به دودویی از روش مشابه با بخش صحیح استفاده می‌شود. با این وجود به جای تقسیم از ضرب و به جای باقیمانده‌ها، بخش‌های صحیح انتخاب می‌گردند. مجدداً بهتر است این روش با مثالی تشریح شود.

❑

مثال ۳-۱

عدد $(0.6875)_{10}$ را به دودویی تبدیل کنید. ابتدا 0.6875 در 2 ضرب می‌شود تا یک عدد صحیح و یک کسر حاصل گردد. کسر دوباره در 2 ضرب می‌شود تا یک عدد صحیح جدید و یک کسر جدید بدست آید. این فرآیند ادامه می‌یابد تا بخش کسری صفر گردد و یا تعداد ارقام دقت مناسبی را ارائه دهند. ضرایب عدد دودویی از اعداد صحیح به صورت زیر بدست می‌آید.

	صحیح		کسری	ضریب
$0.6875 \times 2 =$	1	+	0.3750	$a_{-1} = 1$
$0.3750 \times 2 =$	0	+	0.7500	$a_{-2} = 0$
$0.7500 \times 2 =$	1	+	0.5000	$a_{-3} = 1$
$0.5000 \times 2 =$	1	+	0.0000	$a_{-4} = 1$

بنابراین پاسخ $(0.1011)_2 = (0.a_{-1}a_{-2}a_{-3}a_{-4})_2 = (0.6875)_{10}$ خواهد بود.

برای تبدیل یک عدد کسری از مبنای 10 به یک عدد در مبنای r، روش مشابهی به کار می‌رود. با این تفاوت که به جای ضرب در 2، ضرب در r انجام می‌گردد و ضرایب به جای 0، 1، از محدوده 0 تا $r-1$ خواهند بود.

❑

مثال ۴-۱

$(0.513)_{10}$ را به مبنای هشت ببرید.

$$0.513 \times 8 = 4.104$$
$$0.104 \times 8 = 0.832$$
$$0.832 \times 8 = 6.656$$
$$0.656 \times 8 = 5.248$$
$$0.248 \times 8 = 1.984$$
$$0.984 \times 8 = 7.872$$

جواب تا هفت رقم با معنی که از بخش صحیح حاصلضرب‌ها بدست می‌آید برابر است با

$$(0.513)_{10} = (0.406517\ldots)_8$$

تبدیل اعداد دهدهی که دارای هر دو بخش صحیح و کسری هستند با تبدیل جداگانه دو بخش و ترکیب جواب‌ها صورت می‌گیرد. با استفاده از مثال‌های ۱-۱ و ۳-۱ داریم:

$$(41.6875)_{10} = (101001.1011)_2$$

با استفاده از مثال‌های ۲-۱ و ۴-۱ داریم:

$$(153.513)_{10} = (231.406517)_8$$

❑

۴-۱ اعداد مبنای هشت و شانزده

تبدیل از مبنای دو به مبنای هشت و شانزده، و بالعکس نقش عمده‌ای در کامپیوترهای دیجیتال بازی می‌کند. چون $8 = 2^3$ و $16 = 2^4$ است، هر رقم در مبنای هشت متعلق به سه رقم دودویی و هر رقم مبنای شانزده متعلق به چهار رقم دودویی است. در جدول ۲-۱ شانزده عدد اول سیستم اعداد دهدهی، دودویی، هشت هشتی و شانزده شانزدهی لیست شده‌اند.

تبدیل از دودویی به هشت هشتی به سادگی با تفکیک عدد دودویی به گروه‌های سه رقمی در دو طرف نقطه دودویی بدست می‌آید. سپس به هر گروه یک رقم مبنای هشت تعلق می‌گیرد. مثال زیر روال مربوطه را نشان می‌دهد:

$$(10\ 110\ 001\ 101\ 011 \cdot 111\ 100\ 000\ 110)_2 = (26153.7406)_8$$
$$\quad 2\ \ 6\ \ 1\ \ 5\ \ 3\ \ \ 7\ \ 4\ \ 0\ \ 6$$

تبدیل از مبنای دو به مبنای شانزده نیز مشابه با روند فوق است، با این تفاوت که عدد دودویی به گروه‌های چهار رقمی تفکیک می‌شوند:

$$(10\ 1100\ 0110\ 1011 \cdot 1111\ 0010)_2 = (2C6B.F2)_{16}$$
$$\quad 2\ \ C\ \ 6\ \ B\ \ F\ \ 2$$

پس از مطالعه مقادیر لیست شده در جدول ۲-۱، به سادگی می‌توان رقم مبنای شانزده را برای هر گروه از ارقام دودویی به خاطر سپرد.

جدول ۲-۱ . اعداد با مبناهای متفاوت

دهدهی (مبنای 10)	دودویی (مبنای 2)	هشتایی (مبنای 8)	شانزده تایی (مبنای 16)
00	0000	00	0
01	0001	01	1
02	0010	02	2
03	0011	03	3
04	0100	04	4
05	0101	05	5
06	0110	06	6
07	0111	07	7
08	1000	10	8
09	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

تبدیل از مبنای هشت یا شانزده به دودویی با روشی عکس روش بالا انجام می‌گردد. هر رقم مبنای هشت با سه رقم مبنای دو معادل خود جایگزین می‌شود. به طور مشابه، هر رقم مبنای شانزده با چهار رقم دودویی معادلش جایگزین خواهد شد. این مطلب در مثال‌های زیر تشریح شده است:

$$(673.124)_8 = (\underset{6}{110}\ \underset{7}{111}\ \underset{3}{011}\ .\ \underset{1}{001}\ \underset{2}{010}\ \underset{4}{100})_2$$

و

$$(306.D)_{16} = (\underset{3}{0011}\ \underset{0}{0000}\ \underset{6}{0110}\ .\ \underset{D}{1101})_2$$

اساساً کار با اعداد دودویی، به دلیل این‌که تعداد ارقامشان سه یا چهار برابر معادلشان در مبنای ده می‌باشد، مشکل است. مثلاً عدد دودویی 1111 1111 1111 معادل 4095 است. با این وجود کامپیوترهای دیجیتال اعداد دودویی را به کار می‌برند و گاهی نیز لازم است تا کاربر مستقیماً به وسیله اعداد دودویی با ماشین ارتباط برقرار کند. یک راه برای حفظ سیستم دودویی در کامپیوتر، که در ضمن تعداد ارقام را برای انسان کاهش می‌دهد، استفاده از رابطه بین سیستم اعداد دودویی و هشت هشتی یا شانزده شانزدهی است. با این روش، انسان برحسب اعداد مبنای هشت یا شانزده فکر کرده و در مواقعی که ارتباط مستقیم با ماشین لازم است، تبدیل لازمه را با بررسی این اعداد انجام خواهد داد. به این ترتیب عدد دودویی 1111 1111 1111 که دارای 12 رقم است در مبنای هشت به صورت چهار رقم 7777 و یا در مبنای شانزده به شکل FFF در می‌آید. به هنگام تبادل اطلاعات با انسان، نمایش مبنای هشت یا شانزده اعداد دودویی مطلوب‌تر است زیراکه در این مبناها اعداد با 1/3 یا 1/4 تعداد ارقامشان در دودویی قابل نمایش‌اند. بنابراین اغلب کتابچه‌های راهنمای کامپیوتر از اعداد مبنای هشت یا شانزده برای نمایش کمیت‌های دودویی استفاده

می‌کنند. گرچه نمایش مبنای شانزده مناسب‌تر به نظر می‌رسد ولی انتخاب یکی از این دو کاملاً اختیاری است.

۱-۵ متمم‌ها

متمم‌ها در کامپیوترهای دیجیتال برای ساده کردن عمل تفریق و یا عملیات منطقی به کار می‌روند. در هر مبنایی چون r، دو نوع متمم وجود دارد: یکی متمم مبنا و دیگری متمم مبنای کاهش یافته. فرم اول به متمم r و دومی به متمم $r-1$ موسوم است. وقتی که مقدار مبنا یا پایه را جایگزین کنیم، برای اعداد دودویی، متمم‌های 2 و 1 و برای اعداد دهدهی، متمم‌های 10 و 9 را خواهیم داشت.

متمم در مبنای کاهش یافته

با فرض داشتن عددی n رقمی مانند N در مبنای r، متمم $(r-1)$ عدد به صورت $(r^n-1) - N$ تعریف می‌شود. برای اعداد دهدهی، $r = 10$ و $r-1 = 9$ است، و به این ترتیب متمم 9 عدد N برابر $(10^n-1) - N$ خواهد بود. در اینجا 10^n نمایشگر عددی است که متشکل از یک 1 و به دنبال آن n عدد 0 می‌باشد. 10^n-1 عددی است که با n عدد 9 نشان داده می‌شود. مثلاً اگر $n = 4$ باشد، داریم $10^4 = 10000$ و $10^4-1 = 9999$. به این ترتیب نتیجه می‌شودکه متمم 9 یک عدد دهدهی با تفریق هر رقم از 9 حاصل خواهد شد. به چند مثال عددی زیر توجه کنید.

متمم 9 عدد 546700 برابر است با 999999 - 546700 = 453299
متمم 9 عدد 012398 برابر است با 999999 - 012398 = 987601

برای اعداد دودویی، $r = 2$ و $r-1 = 1$ است، بدین ترتیب متمم 1 عدد N، $(2^n-1) - N$ خواهد بود. مجدداً، 2^n برابر با یک عدد دودویی است که از یک 1، و n عدد 0 تشکیل شده است. 2^n-1 یک عدد دودویی متشکل از n عدد 1 می‌باشد. مثلاً اگر $n = 4$ باشد، داریم $(10000)_2 = 2^4$ و $(1111)_2 = 2^4-1$. بنابراین متمم 1 یک عدد دودویی از تفریق هر رقم از 1 بدست می‌آید. با این وجود، هنگام تفریق ارقام دودویی از 1، یکی از دو حالت $1 - 0 = 1$ و یا $0 = 1 - 1$ را خواهیم داشت، که سبب می‌شود هر بیت از 0 به 1 و از 1 به 0 تبدیل شود. بنابراین متمم یک عدد دودویی با تغییر 1ها به 0 و 0ها به 1 حاصل می‌گردد. در زیر مثال‌هایی آورده شده است:

متمم 1 عدد 1011000 برابر است با 0100111
متمم 1 عدد 0101101 برابر است با 1010010

متمم $(r-1)$ اعداد مبنای هشت و شانزده بترتیب از تفریق ارقام آنها از 7 یا F (15 دهدهی) حاصل می‌شود.

متمم مبنا

متمم r یک عدد n رقمی مانند N در مبنای r به صورت r^n-N به ازاء $N \neq 0$ و برابر با 0 در ازاء $N=0$ تعریف می‌شود. از مقایسه این متمم با متمم $(r-1)$ نتیجه می‌شود که متمم r از جمع 1 با متمم $(r-1)$

حاصل می‌شود زیرا $r^n-N = [(r^n-1) - N] + 1$ می‌باشد. به این ترتیب متمم 10 یک عدد دهدهی مانند 2389 برابر است با $7610 + 1 = 7611$ که از جمع 1 به مقدار متمم 9 حاصل می‌گردد. متمم 2 عدد دودویی 101100 برابر است با $010011 + 1 = 010100$ و از جمع 1 با مقدار متمم 1 بدست می‌آید.

چون 10^n عددی است که با یک 1 و n عدد 0 به دنبال آن نمایش داده می‌شود، 10^n-N که متمم 10 عدد N است نیز با تغییر ندادن 0های کم‌ارزش‌تر و تفریق اولین رقم غیر صفر کم‌ارزش‌تر از 10 و تفریق همه رقم‌های باارزش‌تر از 9 حاصل می‌گردد.

متمم 10 عدد 012398 برابر 987602 می‌باشد

متمم 10 عدد 246700 برابر 753300 است

متمم 10 اولین عدد با تفریق 8 از 10 در کم‌ارزش‌ترین مکان و تفریق دیگر ارقام از 9 حاصل شده است. متمم 10 دومین عدد بدین ترتیب حاصل گشته است که دو 0 کم‌ارزش‌تر رها می‌شوند، 7 از 10 و دیگر ارقام از 9 تفریق می‌گردند.

به طور مشابه متمم 2 می‌تواند با رها کردن همه 0های کم‌ارزش‌تر و نیز تغییر نکردن اولین 1 و جایگزینی همه 0ها با 1ها و 1ها با 0ها در دیگر ارقام باارزش‌تر حاصل شود.

متمم 2 عدد 1101100 برابر 0010100 است

متمم 2 عدد 0110111 برابر 1001001 است

متمم 2 اولین عدد با رها کردن دو 0 کم‌ارزش‌تر و اولین 1 و سپس جایگزینی همه 1ها با 0 و 0ها با 1 در چهار رقم با ارزش‌تر باقیمانده بدست می‌آید. متمم 2 دومین عدد با رها کردن اولین 1 و متمم کردن دیگر ارقام حاصل می‌گردد.

در تعاریف قبلی، فرض شد که اعداد دارای نقطه ممیز نیستند. اگر عدد اولیه N حاوی ممیز باشد باید آن را موقتاً حذف نمود تا متمم r و $(r-1)$ بدست آید. آنگاه آن را به مکان مربوطه‌اش باز می‌گردانیم. توجه داشته باشید که متمم یک متمم، عدد را به حالت اولیه‌اش باز می‌گرداند. متمم r عدد N برابر r^n-N است. متمم یک متمم برابر است با $r^n - (r^n-N) = N$، که همان عدد اولیه است.

تفریق به کمک متمم‌ها

روش مستقیم تفریق که در مدارس ابتدایی بیان شد از مفهوم قرض کردن استفاده می‌نماید. در این روش وقتی که یک رقم در مفروق‌منه کوچکتر از مفروق باشد یک 1 از رقم باارزش‌تر قرض گرفته می‌شود. این روش هنگامی که تفریق با قلم و کاغذ انجام شود به خوبی کار می‌کند. با این وجود، هنگام پیاده‌سازی تفریق با سخت‌افزار دیجیتال، روش کمتر از روش‌های متمم کارآیی دارد.

تفریق دو عدد n رقمی بی‌علامت $M - N$ در مبنای r به صورت زیر انجام می‌شود.

۱- مفروق منه M را به متمم r مفروض N، اضافه کنید. یعنی

$$M + (r^n-N) = M - N + r^n$$

۲- اگر $M \geq N$ باشد، عمل جمع یک رقم نقلی انتهایی تولید می‌کند که باید چشم‌پوشی شود؛ آنچه

باقی می‌ماند نتیجه M − N است.

۳- اگر M < N باشد، عمل جمع هیچگونه رقم نقلی انتهایی تولید نمی‌کند و جواب برابر با $r^n - (M - N)$ است که همان متمم r عمل (M − N) می‌باشد. برای یافتن جواب معمول، متمم r حاصل جمع را بدست می‌آوریم و سپس یک علامت منفی در جلو آن می‌گذاریم.

مثال‌های زیر روال را تشریح می‌کنند.

مثال ۵-۱

با استفاده از متمم 10 تفریق 3250 − 72532 را انجام دهید.

M	=	72532
متمم 10 عدد N	= +	96750
حاصل جمع	=	169282
چشم‌پوشی از رقم نقلی انتهایی 10^5	= −	100000
جواب	=	69282

توجه کنید که M دارای 5 رقم ولی N فقط 4 رقمی است. چون هر دو عدد باید دارای تعداد ارقام برابری باشند، پس N به صورت 03250 نوشته می‌شود. متمم 10 این عدد N ، یک 9 در باارزش‌ترین مکان تولید می‌نماید. تولید رقم نقلی در باارزش‌ترین مکان دلالت بر M ≥ N دارد و نتیجه نیز مثبت است.

❑

مثال ۶-۱

با استفاده از متمم 10 ، تفریق 72532 − 3250 را بدست آورید.

M	=	03250
متمم 10 عدد N	= +	27468
حاصل جمع	=	30718

که رقم نقلی در آن وجود ندارد.

بنابراین جواب (متمم 10 عدد 30718)− = 69282− است.

❑

توجه کنید که چون 72532 > 3250 است نتیجه منفی است. نظر به این که ما با اعداد بی‌علامت سروکار داریم، نمی‌توان برای این حالت نتیجه بدون علامتی بدست آورد. وقتی تفریق را با متمم‌ها انجام می‌دهیم، جواب منفی از نبود رقم نقلی انتهایی و نتیجه متمم تشخیص داده می‌شود. هنگام کار با کاغذ و قلم، می‌توانیم جواب را به یک عدد منفی علامت‌دار تغییر دهیم تا به فرم معمول‌تر در آید. تفریق با متمم‌ها برای اعداد دودویی به روش مشابهی در مثال‌های زیر آمده است.

مثال ۷-۱

با فرض دو عدد دودویی X = 1010100 و Y = 1000011، تفریق‌های زیر را انجام دهید.

(الف) X − Y و (ب) Y − X با استفاده از متمم 2

(الف)

X	=	1010100
متمم 2 عدد Y	=	+ 0111101
حاصل جمع	=	10010001
چشم‌پوشی از نقلی انتهایی 2^7	=	−10000000
X − Y	=	0010001 جواب :

(ب)

Y	=	1000011
متمم 2 عدد X	=	+ 0101100
حاصل جمع	=	1101111

رقم نقلی انتهایی وجود ندارد.

بنابراین، جواب Y − X = (متمم 2 عدد 1101111) − = 0010001− است. ❑

تفریق اعداد بی‌علامت را می‌توان با متمم (r−1) نیز انجام داد. به خاطر دارید که متمم (r−1) ، یک واحد کمتر از متمم r است. به این علت، نتیجه جمع مفروق منه با متمم مفروق حاصل جمعی تولید می‌کند که یکی کمتر از تفاضل صحیح به هنگام رخداد نقلی انتهایی است. حذف رقم نقلی انتهایی و افزودن 1 به حاصل جمع را رقم نقلی چرخشی می‌خوانند.

مثال ۸-۱

مثال ۷-۱ را با استفاده از متمم 1 انجام دهید.

(الف) X − Y = 1010100 − 1000011

X	=	1010100
Y	=	+ 0111100
حاصل جمع	=	10010000
رقم نقلی انتهایی	=	+ 1
X − Y	=	0010001 جواب :

(ب) Y − X = 1000011 − 1010100

Y	=	1000011
X	=	+ 0101011
حاصل جمع	=	1101110

که رقم نقلی انتهایی وجود ندارد.

بنابراین جواب Y − X = (متمم 1 عدد 1101110)− = 0010001− است. ❑

توجه کنید که نتیجه منفی با اخذ متمم 1 از حاصل جمع بدست آمده است زیرا این نوع متمم به کار رفت. روال رقم نقلی انتهایی چرخشی در تفریق اعداد دهدهی بی‌علامت با متمم 9 نیز قابل استفاده است.

۶-۱ اعداد دودویی علامت‌دار

اعداد صحیح مثبت و از آن جمله صفر را می‌توان با اعداد بی‌علامت نشان داد. با این وجود برای نمایش اعداد صحیح منفی به علامتی نیاز داریم. در حساب معمولی، یک عدد منفی را با یک علامت منها و عدد مثبت را با علامت بعلاوه نشان می‌دهند. به دلیل محدودیت در سخت‌افزار، کامپیوترها باید هر چیزی را با ارقام دودویی نشان دهند. مرسوم است که علامت را با یک بیت واقع در سمت چپ‌ترین مکان عدد نمایش دهند. معمولاً اعداد مثبت را با گذاشتن 0 و اعداد منفی را با گذاشتن 1 در محل بیت مزبور معرفی می‌نمایند.

لازم است بدانیم که هر دو گروه اعداد دودویی علامت‌دار و بی‌علامت هنگام ارائه به کامپیوتر از رشته بیت‌ها تشکیل شده‌اند. معمولاً کاربر علامت‌دار بودن یا نبودن عدد را معین می‌نماید. اگر عدد دودویی علامت‌دار باشد سمت چپ‌ترین بیت، علامت، و بقیه بیت‌ها عدد هستند. اگر عدد دودویی بدون علامت فرض شود، سمت چپ‌ترین بیت، باارزش‌ترین بیت عدد خواهد بود. مثلاً رشته بیت‌های 01001 می‌تواند به عنوان 9 (دودویی بی‌علامت) و یا 9+ (دودویی علامت‌دار) در نظر گرفته شود زیرا سمت چپ‌ترین بیت 0 است. رشته بیت‌های 11001 به عنوان 25 بی‌علامت و یا 9- علامت‌دار خواهد بود زیرا در سمت چپ‌ترین مکان عدد، رقم 1 وجود دارد که بیانگر منفی بودن عدد، و بقیه چهار بیت عدد 9 را نشان می‌دهد. معمولاً اگر نوع عدد از قبل مشخص باشد هیچگونه اشتباهی در تشخیص وجود نخواهد داشت.

نمایش اعداد علامت‌دار در آخرین مثال فوق، نمایش **مقدار- علامت‌دار** نامیده می‌شود. در این نام‌گذاری، عدد شامل مقدار و یک سمبل (+ یا -) یا یک بیت (0 یا 1) برای مشخص نمودن علامت است. این روش در محاسبات معمولی مورد استفاده می‌باشد. وقتی که عملیات حسابی در یک کامپیوتر پیاده‌سازی می‌شوند، بهتر است روش دیگری به نام سیستم متمم- علامت‌دار برای ارائه اعداد منفی به کار گرفته شود. در این سیستم، یک عدد منفی با متمم‌اش مشخص می‌شود. در حالی که سیستم مقدار- علامت، عدد را با تغییر علامتش منفی می‌کند، **سیستم متمم- علامت‌دار**، منفی عدد را با متمم‌سازی‌اش تهیه می‌نماید. چون اعداد مثبت همواره با 0 در سمت چپ‌شان شروع می‌شوند متمم همواره با 1 آغاز می‌گردد، که بیانگر عدد منفی خواهد بود. سیستم متمم- علامت‌دار می‌تواند از متمم 1 یا متمم 2 استفاده کند ولی متمم 2 رایج‌تر است.

به عنوان مثال فرض کنید که عدد 9 با هشت بیت در دودویی نشان داده شده باشد. 9+ با یک بیت 0 در سمت چپ‌ترین مکان و به دنبال آن معادل دودویی 9 می‌آید که نتیجه 00001001 خواهد بود. توجه داشته باشید که تمام هشت بیت باید مقدار داشته باشند، بنابراین پس از بیت علامت بقیه مکان‌ها تا اولین 1 از سمت چپ با 0 پر می‌شوند. هر چند که برای نمایش 9+ فقط یک راه وجود دارد، برای

نمایش 9- سه روش موجود است.

نمایش مقدار- علامت‌دار 10001001
نمایش متمم 1 _ علامت‌دار 11110110
نمایش متمم 2 _ علامت‌دار 11110111

در سیستم مقدار- علامت‌دار، با تغییر بیت علامت در سمت چپ‌ترین مکان، از 0 به 1 ، عدد 9- به 9+ تبدیل می‌شود. در متمم 1 _ علامت‌دار، 9- را با متمم کردن همه بیت‌های 9+ ، از جمله بیت علامت بدست می‌آوریم. در متمم 2 _ علامت‌دار، 9- با متمم 2کردن تمام بیت‌های عدد مثبت از جمله بیت علامت حاصل می‌شود.

جدول ۳-۱ همه اعداد دودویی 4 بیت را به هر سه فرم نمایش، نشان می‌دهد. عدد دهدهی معادل نیز به منظور وجود مرجع آورده شده است. توجه کنید که اعداد مثبت در هر سه نمایش یکسانند و دارای یک 0 در سمت چپ‌ترین مکان می‌باشند. سیستم متمم 2_ علامت‌دار تنها یک نمایش برای 0 دارد که همیشه مثبت است. دو سیستم دیگر دارای 0 مثبت و 0 منفی‌اند، چیزی که در محاسبات معمولی با آن مواجه نمی‌شویم. مجدداً توجه کنید که همه اعداد منفی دارای 1 در سمت چپ‌ترین بیت‌اند. به این ترتیب ما آنها را از اعداد مثبت تفکیک می‌نماییم. با چهار بیت قادریم 16 عدد دودویی را نشان دهیم. در سیستم مقدار- علامت‌دار و متمم 1 ، هشت عدد مثبت و هشت عدد منفی و از جمله دو عدد صفر وجود دارد. در نمایش متمم 2 ، هشت عدد مثبت از جمله صفر و هشت عدد منفی موجود است.

سیستم مقدار- علامت‌دار در حساب معمولی مورد استفاده است و هنگامی که در کامپیوتر به کار

جدول ۳-۱ . اعداد دودویی علامت‌دار

دهدهی	متمم 2 - علامت دار	متمم 1 - علامت دار	مقدار - علامت دار
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000
−0	—	1111	1000
−1	1111	1110	1001
−2	1110	1101	1010
−3	1101	1100	1011
−4	1100	1011	1100
−5	1011	1010	1101
−6	1010	1001	1110
−7	1001	1000	1111
−8	1000	—	—

رود، مشکلاتی به همراه دارد زیرا باید علامت و مقدار به طور جداگانه دستکاری شوند. بنابراین، معمولاً متمم- علامت به کار گرفته می‌شود. متمم 1 نیز مشکلاتی را به بار می‌آورد و به ندرت در محاسبات به کار می‌رود. متمم 1 برای اعمال منطقی مفید است چون تبدیل 0 به 1 و یا 1 به 0 معادل با عمل متمم منطقی است. این مطلب در فصل بعدی نشان داده خواهد شد. بحث زیر در مورد محاسبات دودویی علامت‌دار، اختصاصاً به نمایش اعداد منفی متمم 2_ علامت‌دار تعلق دارد. روال‌های مشابهی قابل اعمال به سیستم متمم 1_ علامت‌دار بوده و در آن رقم نقلی چرخشی همچون اعداد بدون علامت به حساب خواهد آمد.

جمع حسابی

جمع دو عدد در سیستم مقدار- علامت از قوانین معمول در حساب تبعیت می‌نماید. اگر علامت‌ها یکسان باشند دو مقدار را با هم جمع کرده و به حاصل جمع علامت مشترک را تخصیص می‌دهیم. اگر علامت‌ها یکی نباشند، مقدار کوچکتر را از بزرگتر کم می‌کنیم و به نتیجه حاصل علامت عدد بزرگتر را اختصاص می‌دهیم. مثلاً $-12 = -(37 - 25) = -(37) + (+25)$ است، که با تفریق مقدار کوچکتر 25 از مقدار بزرگتر 37 و استفاده از علامت 37 برای علامت نتیجه انجام شده است. این فرآیند به مقایسه علامت‌ها و اندازه‌ها و سپس اجرای جمع و تفریق نیاز دارد. روال مشابهی در نمایش مقدار- علامت برای اعداد دودویی قابل اعمال است. برعکس قوانین جمع اعداد در سیستم مقدار- علامت نیازی به مقایسه و تفریق ندارد، بلکه فقط باید آنها را جمع کرد. روال بسیار ساده بوده و برای اعداد دودویی به صورت زیر بیان می‌گردد:

جمع دو عدد دودویی علامت‌دار با اعداد منفی که به فرم متمم 2 نمایش داده شده‌اند از جمع دو عدد از جمله بیت‌های علامت حاصل می‌شود. رقم نقلی حاصل از بیت علامت چشم‌پوشی می‌گردد. مثال‌های عددی برای جمع در زیر آمده است.

+ 6	00000110	− 6	11111010
+13	00001101	+13	00001101
+19	00010011	+ 7	00000111
+ 6	00000110	−6	11111010
−13	11110011	−13	11110011
− 7	11111001	−19	11101101

توجه کنید که اعداد منفی باید از ابتدا به صورت متمم 2 باشند و حاصل جمع اگر منفی باشد به صورت متمم 2 خواهد بود.

در هر یک از چهار حالت فوق، عمل انجام شده جمعی است که در آن بیت علامت هم لحاظ شده است. در این روش هر رقم نقلی خروجی نادیده گرفته می‌شود و نتایج منفی به فرم متمم 2 هستند.

برای یافتن یک جواب صحیح، باید مطمئن باشیم که برای جای دادن نتیجه، تعداد کافی بیت وجود

دارد. اگر با دو عدد n بیت شروع کنیم و حاصل جمع n+1 بیت را اشغال کند گوییم سرریز رخ داده است. هنگامی که جمع با کاغذ و قلم انجام می‌شود سرریز مسئله‌ای نیست زیرا ما از نظر عرض صفحه محدودیت نداریم. در این گونه موارد فقط یک 0 به بالاترین مکان عدد مثبت و یا یک 1 به بالاترین مکان عدد منفی می‌افزاییم تا آنها را به n+1 بیت گسترش دهیم و سپس جمع را اجرا نماییم. ولی سرریز در کامپیوتر مشکل‌ساز است زیرا تعداد بیت‌هایی که عدد را نگه می‌دارند محدود می‌باشد، و نتیجه‌ای که از مقدار نهایی به میزان 1 واحد تجاوز کند را نمی‌توان در آن جای داد.

فرم متمم اعداد منفی برای کسانی که به سیستم مقدار- علامت عادت کرده‌اند، ناآشناست. برای تعیین یک عدد منفی وقتی که به فرم متمم 2 علامت‌دار باشد، لازم است که آن را به یک عدد مثبت تبدیل کنیم تا به شکل آشناتری در آید. مثلاً عدد دودویی علامت‌دار 11111001 یک عدد منفی است زیرا سمت چپ‌ترین بیت برابر 1 است. متمم 2 آن 00000111 می‌باشد که معادل دودویی عدد 7+ است. به این ترتیب تشخیص می‌دهیم که عدد منفی اولیه برابر 7− بوده است.

تفریق حسابی

تفریق دو عدد دودویی علامت‌دار، وقتی که اعداد منفی به صورت متمم 2 باشند بسیار ساده است و می‌تواند به صورت زیر بیان شود:

متمم 2 مفروق (از جمله بیت علامت) را بدست آورید و آن را به مفروق منه (از جمله بیت علامت) اضافه کنید. رقم نقلی خروجی از مکان بیت علامت چشم‌پوشی شود.

این پدیده به این علت رخ می‌دهد که اگر علامت مفروق عوض شود، تفریق به جمع تبدیل خواهد شد. این نکته با روابط زیر نشان داده شده است:

$$(\pm A) - (+B) = (\pm A) + (-B);$$
$$(\pm A) - (-B) = (\pm A) + (+B).$$

اما تغییر یک عدد مثبت به یک عدد منفی به سادگی با بدست آوردن متمم 2 آن امکان‌پذیر است. عکس مطلب فوق نیز صحیح می‌باشد زیرا متمم یک عدد منفی متمم، یک عدد مثبت معادل تولید می‌نماید. تفریق 7+ = (13−) − (6−) را در نظر بگیرید. در دودویی با هشت بیت، این تفریق به صورت (11111010 − 11110011) است. با یافتن متمم 2 مفروق (13−)، یعنی (13+)، تفریق به فرم جمع در می‌آید. در دودویی این عمل به صورت 100000111 = 00001101 + 11111010 می‌باشد. با حذف رقم نقلی انتهایی پاسخ صحیح 00000111 (7+) خواهد بود.

لازم به تذکر است که جمع و تفریق اعداد دودویی در سیستم متمم- علامت مشابه با قوانین جمع و تفریق معمولی است. بنابراین کامپیوترها دارای سخت‌افزار مشترک برای هر دو نوع عمل حسابی می‌باشند. کاربر یا برنامه‌نویس باید نتایج چنین جمع یا تفریقی را به طور متفاوت تفسیر کند و این تفسیر به فرض اولیه وی یعنی علامت‌دار بودن یا بی‌علامت بودن اعداد بستگی دارد.

۱-۷ کدهای دودویی

سیستم‌های دیجیتال از سیگنال‌هایی که دو مقدار مجزا و عناصری از مدار که دو حالت با ثبات دارند استفاده می‌کنند. بین سیگنال‌های دودویی، عناصر مدار دودویی و ارقام دودویی رابطه مستقیمی وجود دارد. مثلاً یک عدد دودویی n رقمی را می‌توان با n عنصر مدار دودویی که هر یک دارای یک سیگنال خروجی معادل 0 یا 1 اند، نشان داد. سیستم‌های دیجیتال نه تنها اعداد دودویی بلکه بسیاری از اجزاء گسسته اطلاعات را هم دستکاری و نمایش می‌دهند و روی آنها عمل می‌کنند. هر عنصر گسسته اطلاعاتی را در میان یک گروه از مقادیر می‌توان با استفاده از کد دودویی نشان داد. کدها باید به صورت دودویی باشند زیرا کامپیوترها فقط قادرند 1ها و 0ها را نگه دارند. با ید توجه داشت که کدها فقط نماد یا سمبل نمایش اطلاعات را عوض می‌کنند و نه مفهوم آنها را. اگر بیت‌های یک کامپیوتر را به طور تصادفی مورد بررسی قرار دهیم، ملاحظه خواهیم کرد که اغلب به جای اعداد دودویی، اطلاعات کد شده در آنها وجود دارد.

یک کد دودویی n بیت، گروهی متشکل از n بیت است که 2^n ترکیب ممکن از 1ها و 0ها را داراست، و هر ترکیب یک عنصر از مجموعه کد شده را نمایش می‌دهد. یک مجموعه چهار عنصری با دو بیت کد می‌شود، که به هر عنصر یکی از ترکیبات بیتی زیر تخصیص می‌یابد: 00، 01، 10، 11. یک مجموعه هشت عضوی به کد 3 بیت نیاز دارد و برای یک مجموعه 16 عنصری یک کد 4 بیت لازم است. ترکیب بیتی یک کد n بیتی با شمارش دودویی از 0 تا 2^n-1 حاصل می‌گردد. به هر عنصر باید یک ترکیب بیتی دودویی منحصر به فرد اختصاص یابد و هیچ دو عنصر دارای مقدار یکسانی نمی‌باشند؛ در غیر این صورت تخصیص کد گنگ و بی‌معنی خواهد بود.

گرچه حداقل تعداد بیت‌های لازم برای 2^n کد مجزا، برابر n است، حداکثر تعداد بیت‌ها برای تعریف یک کد دودویی وجود ندارد. مثلاً 10 رقم دهدهی با 10 بیت قابل کد شدن است، و هر رقم دهدهی به یکی از ترکیبات نُه 0 و یک 1 تخصیص می‌یابد. در این کد دهدهی، رقم 6 به ترکیب بیتی 0001000000 اختصاص می‌یابد.

کد BCD

گرچه سیستم اعداد دودویی طبیعی‌ترین سیستم برای یک کامپیوتر است، ولی بسیاری از مردم به سیستم دهدهی عادت دارند. یکی از راه‌های حل این مشکل تبدیل اعداد دهدهی به دودویی، اجرای همه محاسبات به دودویی و سپس تبدیل نتایج دودویی به دهدهی است. این روش لازم می‌دارد تا اعداد دهدهی را در کامپیوتر ذخیره کنیم تا بتوانند به دودویی تبدیل شوند. چون کامپیوتر فقط می‌تواند مقادیر دودویی را قبول کند، باید ارقام دهدهی را با کدی مرکب از 1ها و 0ها نشان دهیم. هنگامی که این ارقام به فرم کد شده در کامپیوتر ذخیره شوند، می‌توان مستقیماً عملیات حسابی را روی این اعداد دهدهی اجرا نمود.

اگر تعداد عناصر در مجموعه به صورت توانی از 2 نباشد، کد دودویی دارای ترکیبات بیتی تخصیص نیافته خواهد بود. 10 رقم دهدهی چنین مجموعه‌ای را می‌سازند. یک کد دودویی که بتواند 10 عنصر را

جدول ۴-۱ . دهدهی کد شده به دودویی (BCD)

سمبل دهدهی	رقم BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

از هم تفکیک کند باید حداقل 4 بیت داشته باشد، ولی 6 ترکیب از 16 ترکیب ممکن تخصیص نیافته باقی می‌ماند. با مقداردهی 10گانه به 4 بیت می‌توان کدهای دودویی متفاوتی بدست آورد. کدی که برای ارقام دهدهی معمولاً به‌کار می‌رود در جدول ۴-۱ دیده می‌شود. این کد را دهدهی کد شده به دودویی می‌خوانند و به طور خلاصه آن را با BCD نمایش می‌دهند. چند کد دهدهی دیگر بعداً در این بخش نمایش داده خواهند شد.

جدول ۴-۱ هر کد 4 بیتی را به یک رقم دهدهی نسبت می‌دهد. یک عدد k رقمی در BCD به 4k بیت نیاز دارد. عدد دهدهی 396 در BCD با 12 بیت به صورت 0110 1001 0011 نمایش داده می‌شود، که در آن هر گروه 4 بیتی یک رقم دهدهی را نشان می‌دهد. هرگاه عدد دهدهی در BCD بین 0 تا 9 باشد با عدد دودویی‌اش معادل است. یک عدد BCD بزرگتر از 10 با عدد دودویی معادلش، هر چند که از 0ها و 1ها تشکیل شده، متفاوت است. به علاوه ترکیبات دودویی 1010 تا 1111 به‌کار نرفته‌اند و در BCD مفهومی ندارند. عدد دهدهی 185 و مقدار مربوطه آن را در BCD و دودویی ملاحظه کنید:

$$(185)_{10} = (0001\ 1000\ 0101)_{BCD} = (10111001)_2$$

مقدار BCD دارای 12 بیت است، ولی عدد دودویی معادل آنها تنها 8 بیت لازم دارد. واضح است یک عدد BCD نسبت به مقدار دودویی به بیت‌های بیشتری احتیاج دارد. با این وجود، استفاده از اعداد دهدهی دارای مزیت است زیرا داده‌های ورودی و خروجی به وسیله انسان‌هایی که سیستم‌های دهدهی را به‌کار می‌برند تولید می‌شود.

توجه داشته باشید که اعداد BCD اعداد دهدهی هستند و نه اعداد دودودویی، هر چند که آنها در ساختارشان از بیت‌ها استفاده می‌کنند. تنها تفاوت بین یک عدد دهدهی و BCD این است که اعداد دهدهی سمبل‌های 0، 1، 2، ...، 9 و اعداد BCD سمبل‌های 0000، 0001، 0010، ...، 1001 را به‌کار می‌برند. مقدار دهدهی دقیقاً یکی است. عدد 10 دهدهی در BCD با هشت بیت 0000 0001 و عدد 15 با 0101 0001 نمایش داده می‌شوند. مقادیر دودویی معادل آنها به ترتیب 1010 و 1111 است که تنها چهار بیت دارند.

جمع BCD

جمع دو رقم دهدهی در BCD، همراه با رقم نقلی احتمالی از جفت رقم کم‌ارزش‌تر قبلی را در نظر بگیرید. چون هر رقم از 9 تجاوز نمی‌کند، جمع نمی‌تواند بزرگتر از 19 = 1 + 9 + 9 باشد که در آن 1، رقم نقلی قبلی است. فرض کنید می‌خواهیم ارقام BCD را به شکل اعداد دودویی با هم جمع کنیم. جمع دودویی، نتیجه‌ای بین 0 تا 19 را تولید خواهد کرد. این مقادیر به دودویی برابرند با 0000 تا 10011، ولی به فرم BCD برابر با 0000 تا 1001 1 می‌باشند. اولین رقم، رقم نقلی و چهار بیت بعدی رقم جمع BCD است. وقتی حاصل جمع دودویی برابر یا کم‌تر از 1001 است (بدون نقلی)، رقم BCD مربوطه صحیح است. با این وجود، وقتی جمع دودویی بزرگتر یا مساوی 1010 باشد، نتیجه یک رقم BCD نامعتبر است. جمع $6 = (0110)_2$ با حاصل جمع دودویی، آن را به رقم صحیح بدل کرده و در صورت لزوم رقم نقلی نیز تولید خواهد کرد. دلیل این است که اختلاف بین یک رقم نقلی در باارزش‌ترین مکان بیتی حاصل از جمع دودویی و نقلی دهدهی برابر است با $16 - 10 = 6$. جمع BCD زیر را در نظر بگیرید:

4	0100	4	0100	8	1000
+5	+0101	+8	+1000	+9	1001
9	1001	12	1100	17	10001
			+0110		+0110
			10010		10111

در هر حالت دو رقم BCD و نیز فرم دودویی آنها با هم جمع می‌شوند. اگر جمع دودویی بزرگتر یا برابر 1010 باشد، به آن 0110 را می‌افزاییم تا رقم BCD حاصل جمع و نقلی صحیح حاصل شود. در مثال اول حاصل جمع برابر 9 می‌باشد که یک رقم حاصل جمع BCD صحیح است. در مثال دوم، جمع دودویی یک رقم BCD نامعتبر تولید می‌کند. افزایش 0110 به آن رقم حاصل جمع BCD صحیح 0010 (2) را همراه با یک رقم نقلی به وجود می‌آورد. در مثال سوم، جمع دودویی یک رقم نقلی خواهد داشت. این وضعیت هنگامی رخ می‌دهد که حاصل جمع مساوی یا بزرگتر از 16 باشد. گرچه چهار بیت دیگر کمتر از 1001 است، حاصل جمع نیاز به اصلاح دارد زیرا دارای رقم نقلی است. با جمع 0110 رقم حاصل جمع BCD 0111 (7) و یک نقلی BCD حاصل می‌گردد.

جمع دو عدد BCD بی‌علامت n رقمی روال مشابهی دارد. جمع 760 = 576 + 184 را به BCD در نظر بگیرید:

BCD carry	1	1		
	0001	1000	0100	184
	+0101	0111	0110	+576
Binary sum	0111	10000	1010	
Add 6		0110	0110	
BCD sum	0111	0110	0000	760

اولین جفت رقم BCD کم‌ارزش‌تر، یک رقم BCD برابر با 0000 و یک رقم نقلی برای جفت رقم بعدی را

تولید می‌کند. جفت رقم دوم بعلاوه نقلی قبلی حاصل جمع 0110 و یک رقم نقلی برای جفت رقم بعدی را به وجود می‌آورد. جفت رقم سوم بعلاوه یک رقم نقلی حاصل جمع دودویی 0111 را تولید کرده و نیاز به اصلاح ندارد.

حساب دهدهی

نمایش اعداد دهدهی علامت‌دار در BCD مشابه اعداد علامت‌دار در دودویی است. ما می‌توانیم از هر یک از هر دو سیستم مقدار- علامت‌دار یا متمم- علامت‌دار استفاده کنیم. علامت یک عدد دهدهی معمولاً با چهار بیت نمایش داده می‌شود تا با کد 4 بیت ارقام دهدهی همسان باشد. معمولاً علامت مثبت با چهار 0 و علامت منها با 9 BCD یعنی 1001 نشان داده می‌شود.

سیستم مقدار- علامت‌دار به ندرت در کامپیوترها به کار می‌رود. این سیستم می‌تواند متمم 9 یا متمم 10 باشد، ولی اغلب متمم 10 به کار گرفته می‌شود. برای بدست آوردن متمم 10 یک عدد BCD، ابتدا متمم 9 را بدست اورده و به کم‌ارزش‌ترین رقم 1 واحد می‌افزاییم. متمم 9 نیز با کسر هر رقم از 9 حاصل می‌گردد.

روالی که در بخش قبل برای سیستم متمم 2 علامت‌دار بنا نهاده شد به سیستم متمم 10 علامت‌دار در اعداد دهدهی نیز قابل اعمال است. جمع با افزودن همه ارقام، از جمله رقم علامت و چشم‌پوشی از رقم نقلی انتهایی انجام می‌شود. در اینجا فرض می‌شود که همه اعداد منفی به فرم متمم 10 باشند. جمع +135 = (-240) + (+375) را در نظر بگیرید که در سیستم متمم علامت‌دار انجام شده است.

$$\begin{array}{r} 0\ 375 \\ +9\ 760 \\ \hline 0\ 135 \end{array}$$

عدد 9 واقع در سمت چپ‌ترین مکان عدد دوم نمایشگر یک علامت منفی و 9760 متمم 10 عدد 0240 است. برای بدست آوردن 135+ ، دو عدد با هم جمع و رقم نقلی نادیده گرفته می‌شود. البته اعداد دهدهی داخل کامپیوتر، از جمله ارقام علامت باید به BCD باشند. همانطور که قبلاً اشاره شد جمع با ارقام BCD انجام می‌شود.

تفریق اعداد دهدهی اعم از علامت‌دار یا بی‌علامت در سیستم متمم 10 مشابه با حالت دودویی است. متمم 10 مفروق را بدست آورید و آن را به مفروق منه اضافه کنید. بسیاری از کامپیوترها برای انجام محاسبات حسابی اعداد دهدهی در BCD، سخت‌افزار خاصی دارند. کاربر کامپیوتر می‌تواند برای انجام عمل حسابی با اعداد دهدهی، بدون نیاز به تبدیل آنها، به دودویی برنامه‌نویسی کند.

دیگر کدهای دهدهی

کدهای دودویی برای ارقام دهدهی به حداقل چهار بیت در قبال هر رقم نیاز دارند. با ایجاد 10 ترکیب مختلف در چهار بیت کدهای مختلف را می‌توان ایجاد کرد. کدهای BCD و سه نوع کد دیگر در جدول۵-۱ نشان داده شده‌اند. هر کد تنها 10 ترکیب بیتی از 16 ترکیب ممکن را در چهار بیت به کار

جدول ۵-۱. چهار کد دودویی متفاوت برای ارقام دهدهی

رقم دهدهی	BCD 8421	2421	افزونی 3	8 4-2-1
0	0000	0000	0011	0 0 0 0
1	0001	0001	0100	0 1 1 1
2	0010	0010	0101	0 1 1 0
3	0011	0011	0110	0 1 0 1
4	0100	0100	0111	0 1 0 0
5	0101	1011	1000	1 0 1 1
6	0110	1100	1001	1 0 1 0
7	0111	1101	1010	1 0 0 1
8	1000	1110	1011	1 0 0 0
9	1001	1111	1100	1 1 1 1
ترکیبات بیتی بکارنرفته	1010	0101	0000	0 0 0 1
	1011	0110	0001	0 0 1 0
	1100	0111	0010	0 0 1 1
	1101	1000	1101	1 1 0 0
	1110	1001	1110	1 1 0 1
	1111	1010	1111	1 1 1 0

می‌برد. شش ترکیبی که در هر حال به کار نروند دارای مفهوم نیستند و باید از آنها اجتناب کرد.

کدهای BCD و 2421 از جمله کدهای وزین هستند. در یک کد وزین به هر مکان از بیت وزنی تخصیص داده شده است به نحوی که هر رقم با جمع اوزان تمام 1ها در ترکیب کد بدست می‌آید. کد BCD دارای وزن‌های 8، 4، 2، 1 است که مربوط به توازنی از دو برای هر بیت است. مثلاً تخصیص بیتی 0110 با توجه به وزن 1ها برای 6 تفسیر می‌شود زیرا $8 \times 0 + 4 \times 1 + 2 \times 1 + 1 \times 0 = 6$. ترکیب بیتی 1101 وقتی با کد 2421 وزین شود معادل دهدهی $2 \times 1 + 4 \times 1 + 2 \times 0 + 1 \times 1 = 7$ را خواهد داد. توجه کنید که در کد 2421 بعضی از ارقام به دو طریق کدگذاری می‌شوند. عدد 4 دهدهی به ترکیب‌های بیتی 0100 یا 1010 متعلق است زیرا هر دو ترکیب عدد 4 را نشان می‌دهند.

کدهای 2421 و افزونی 3- مثال‌هایی از کدهای خود متمم هستند. این کدها خواصی دارند که با توجه به آن متمم 9 عدد دهدهی مستقیماً از تغییر 0ها به 1 و 1ها به 0 در کد حاصل می‌شود. مثلاً عدد دهدهی 395 در افزونی 3- به صورت 1000 1100 0110 می‌باشد. متمم 9 آن یعنی 604 به صورت 0111 0011 1001 می‌باشد که در واقع با متمم هر بیت از کد بدست می‌آید (مثل متمم 1 عدد دودویی).

کد افزونی 3- به دلیل خود متممی‌اش در کامپیوترهای قدیمی به کار می‌رفت. این کد بی‌وزن است و هر ترکیب کدی در آن از جمع مقدار دودویی متناظرش با 3 حاصل می‌شود. توجه داشته باشید که کد BCD خود متمم نیست.

کد 8، 4، 2- و 1- مثالی از تخصیص هر دو نوع وزن مثبت و منفی به یک کد دهدهی است. در این حال، ترکیب بیتی 0110 برای 2 دهدهی تخصیص یافته و از رابطه $8\times0+4\times1+(-2)\times1+(-1)\times0=2$ محاسبه می‌شود.

کدگری

خروجی داده بسیاری از سیستم‌های فیزیکی کمیت‌های پیوسته را فراهم می‌کنند. این داده‌ها باید قبل از اعمال به یک سیستم دیجیتال، به فرم دیجیتال تبدیل شوند. اطلاعات پیوسته یا آنالوگ با وسیله‌ای به نام مبدل آنالوگ به دیجیتال به فرم دیجیتال در می‌آیند. گاهی بهتر است برای نمایش داده دیجیتال از کد گری در جدول ۶-۱ استفاده شود. مزیت کد گری نسبت به کد دودویی این است که وقتی از یک کد به کد دیگری می‌رویم تنها یک بیت تغییر می‌یابد. مثلاً در رفتن از 7 به 8، کد گری از 0100 به 1100 تغییر می‌کند. دیده می‌شود که تنها بیت اول سمت چپ از 0 به 1 عوض شده است و سه بیت دیگر ثابت باقی می‌مانند. هنگامی که این کد را با کد دودویی مقایسه نماییم، تغییر از 7 به 8، به صورت 0111 به 1000 خواهد بود و ملاحظه می‌گردد که هر چهار بیت تغییر مقدار داده‌اند.

کد گری در کاربردهایی مورد استفاده است که رشته معمول اعداد دودویی ممکن است در طول انتقال یا تبدیل از یک عدد به دیگری خطایی تولید کنند. اگر اعداد دودویی استفاده شوند، تغییر از 0111 به 1000 ممکن است در صورت تغییر طولانی‌تر سمت راست‌ترین بیت، عدد میانی 1001 را تولید کند که خطاست. کد گری این نوع خطا را حذف می‌کند زیرا تنها یک بیت تغییر در حین انتقال بین دو عدد وجود دارد.

نوعی کاربرد کد گری به هنگام نمایش داده آنالوگ ناشی از تغییرات پیوسته مکان یک شفت موتور می‌باشد. شفت به قطعات تقسیم می‌گردد و به هر قطعه یک عدد تخصیص می‌یابد. اگر قطعات مجاور منطبق با رشته کد گری باشد، ابهام به هنگام تشخیص خطای ناشی از جدایی دو قطعه حذف می‌گردد.

جدول ۶-۱. کد گری 4 بیتی

معادل دهدهی	کد گری
0	0000
1	0001
2	0011
3	0010
4	0110
5	0111
6	0101
7	0100
8	1100
9	1101
10	1111
11	1110
12	1010
13	1011
14	1001
15	1000

کدکارا کتراسکی

در بسیاری از کاربردهای کامپیوترهای دیجیتال نه تنها نیاز به دستکاری روی داده‌های عددی بلکه روی حروف نیز وجود دارد. برای مثال یک کمپانی بیمه با میلیونها سند، از یک کامپیوتر دیجیتال برای پردازش فایل‌هایش استفاده می‌کند. برای نمایش نام و سایر مشخصات طرف‌های قرارداد، داشتن یک کد دودویی برای حروف الفبا ضروری است. به علاوه همان کد دودویی می‌باید اعداد دهدهی و بعضی کاراکترهای خاص دیگر مانند $ را نیز نمایش دهد. یک مجموعه کاراکتر الفبا عددی مجموعه‌ای از عناصر، متشکل از 10 رقم عدد، 26 حروف الفبا و تعداد معینی از علائم خاص است. چنین مجموعه‌ای بین 36 تا 64 عنصر برای حروف بزرگ و یا بین 64 تا 128 عنصر با حروف بزرگ و کوچک برای هر کلید دارد. در حالت اول به شش بیت و در حالت دوم به هفت بیت نیاز است.

کد دودویی استاندارد برای کاراکترهای الفبا عددی، اَسکی (ASCII) است. این کد از هفت بیت برای کد کردن 128 کاراکتر، طبق جدول (۷-۱) استفاده می‌کند. هفت بیت کد با b_1 تا b_7 مشخص شده‌اند که b_7 باارزش‌ترین بیت را تشکیل می‌دهد. مثلاً حروف A در اسکی به صورت 1000001 (ستون 100 و سطر 0001) می‌باشد. کد اسکی حاوی 94 کاراکتر گرافیکی و 34 کاراکتر غیرچاپی برای عملیات کنترلی مختلف است. کاراکترهای گرافیکی نیز از 26 حرف بزرگ (A تا Z)، 26 حرف کوچک (a تا z)، 10 عدد (0 تا 9) و 32 کاراکتر قابل چاپ مانند %، * و $ تشکیل شده است.

34 کاراکتر کنترل در جدول اسکی با اسامی خلاصه شده‌ای مشخص شده‌اند. این کاراکترها در پایین شکل همراه با نوع عملشان ذکر شده‌اند. این کاراکترها برای رده‌دهی داده‌ها و مرتب نمودن مطالب چاپ شدنی در قالب موردنظر به کار می‌روند. سه نوع کاراکترهای کنترلی وجود دارند: افکتورهای فورمت، جداسازی اطلاعات و کاراکترهای کنترل تبادل اطلاعات. افکتور فورمت قالب آنچه راکه باید چاپ شود کنترل می‌نماید. این گروه شامل پَسبَر (BS)، جدول‌بندی افقی (HT) و بازگشت نورد (CR) است. جداسازی‌های اطلاعات، داده‌ها را به صورت پاراگراف‌ها و صفحات دسته‌بندی می‌کند. از جمله آنها می‌توان از جداساز رکورد (RS) و جداساز فایل (FS) نام برد. کاراکترهای کنترل تبادل اطلاعات در حین انتقال متن بین پایانه‌های دور از هم مفیدند. مثال‌هایی از این نوع عبارتند از کاراکتر شروع متن (STX) و ختم متن (ETX) که برای قاب‌بندی یک پیام متنی به هنگام انتقال از خط تلفن به کار می‌رود.

اسکی یک کد 7 بیتی است ولی اغلب کامپیوترها واحدهای هشت بیتی اطلاعات که بایت نام دارند را دستکاری می‌کنند. بنابراین کاراکترهای اسکی اغلب هر کدام در یک بایت ذخیره می‌شوند. بیت اضافی گاهی برای اهداف دیگری به کار می‌رود و اغلب به کاربرد بستگی دارد. مثلاً بعضی از چاپگرها کاراکترهای اسکی را به صورت 8 بیتی می‌شناسند که در آن باارزش‌ترین بیت برابر 0 است. 128 کاراکتر 8 بیتی دیگر را با قرار دادن 1 در باارزش‌ترین مکان برای فونت‌های نوع ایتالیک یا الفبای یونانی می‌توان به کار برد.

کدتشخیص خطا

برای تشخیص خطاها در مخابره یا پردازش داده، گاهی بیت هشتمی به نام بیت توازن به کاراکتر

جدول ۷-۱ . کد استاندارد امریکایی برای تبادل اطلاعات (ASCII)

$b_4b_3b_2b_1$	$b_7b_6b_5$ = 000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

کاراکترهای کنترل

NUL	Null	DLE	Data-link escape
SOH	Start of heading	DC1	Device control 1
STX	Start of text	DC2	Device control 2
ETX	End of text	DC3	Device control 3
EOT	End of transmission	DC4	Device control 4
ENQ	Enquiry	NAK	Negative acknowledge
ACK	Acknowledge	SYN	Synchronous idle
BEL	Bell	ETB	End-of-transmission block
BS	Backspace	CAN	Cancel
HT	Horizontal tab	EM	End of medium
LF	Line feed	SUB	Substitute
VT	Vertical tab	ESC	Escape
FF	Form feed	FS	File separator
CR	Carriage return	GS	Group separator
SO	Shift out	RS	Record separator
SI	Shift in	US	Unit separator
SP	Space	DEL	Delete

اسکی اضافه می‌شود. بیت توازن، بیتی اضافی است که حاوی پیامی بوده و طی آن تعداد 1های کل، زوج یا فرد خواهد شد. دو کاراکتر زیر به همراه توازن زوج یا فرد دیده می‌شوند.

	با توازن زوج	با توازن فرد
ASCII A = 1000001	01000001	11000001
ASCII T = 1010100	11010100	01010100

در هر حالت یک بیت به سمت چپ‌ترین مکان کد می‌افزاییم تا تعداد 1ها در کاراکتر برای توازن زوج، زوج و یا این که تعداد 1ها در کاراکتر برای توازن فرد، فرد گردد. به طور کلی یکی از دو توازن اختیار می‌شود ولی توازن زوج معمول‌تر می‌باشد.

بیت توازن در تشخیص خطا در حین انتقال اطلاعات از یک مکان به مکان دیگر مفید است. این کار با تولید یک بیت توازن زوج برای هر کاراکتر در سمت فرستنده انجام می‌گردد. کاراکترهای 8 بیتی که به همراه بیت‌های توازن می‌باشند به مقصد ارسال می‌گردند. سپس توازن هر کاراکتر در سمت گیرنده چک می‌شود. اگر توازن کاراکتر دریافتی زوج نباشد، حداقل یک بیت در حین انتقال تغییر کرده است. این روش یک، سه یا هر تعداد فردی از خطا را در هر کاراکتر انتقال یافته تشخیص می‌دهد. در این حالت تعداد زوجی از خطاها قابل تشخیص نخواهد بود. کدهای خطای دیگر برای محافظت از خطاهای زوج لازم است.

این‌که پس از شناسایی خطا چه کاری باید انجام داد به کاربرد مربوطه بستگی دارد. یک امکان این است که با فرض اتفاقی بودن خطا و عدم تکرار، تقاضای ارسال مجدد گردد. در این حالت اگر گیرنده یک خطای توازن را شناسایی کند، یک کاراکتر کنترل اسکی، ASCII NAK ، (تصدیق نفی) را متشکل از هشت بیت با توازن زوج 0101 1001 بازپس می‌فرستد. اگر خطایی شناسایی نشد، گیرنده کاراکتر کنترل ACK (تصدیق) را با کد 00000110 باز می‌فرستد. سمت فرستنده دوباره با ارسال پیام NAK پاسخ می‌دهد تا این که توازن صحیح دریافت شود. اگر پس از چند نوبت تکرار، انتقال همچنان دارای خطا بود، پیام خطایی به اپراتور برای چک کردن عامل خطا در خط انتقال فرستاده می‌شود.

۸-۱ ذخیره‌سازی دودویی و ثبات‌ها

اطلاعات دودویی در یک کامپیوتر دیجیتال، نوعی موجودیت فیزیکی در یک محیط ذخیره‌سازی اطلاعات برای ذخیره تک‌تک بیت‌ها باید داشته باشد. یک سلول دودویی وسیله‌ای است که از خود دو حالت باثبات را به نمایش می‌گذارد و قابل استقرار در یکی از دو حالت است. ورودی سلول سیگنال‌های تحریک کننده‌ای را برای استقرار در یکی از دو حالت دریافت می‌کند. خروجی سلول کمیتی فیزیکی است که دو حالت را از هم تفکیک می‌نماید. وقتی خروجی در سلول در یکی از دو حالت باشد اطلاعات ذخیره شده 1 و در حالت با ثبات دیگر 0 است.

ثبات‌ها

ثبات گروهی از سلول‌های دودویی است. یک ثبات با n سلول، هر کمیت گسسته اطلاعاتی را که حاوی n بیت باشد، می‌تواند ذخیره کند. حالت یک ثبات عددی n تایی از 1ها و 0ها است که هر بیت حالت یک سلول را در ثبات بیان می‌کند. محتوای یک ثبات تابعی از تفسیر اطلاعات ذخیره شده در آن است. مثلاً یک ثبات 16 بیتی را با محتوای زیر در نظر بگیرید:

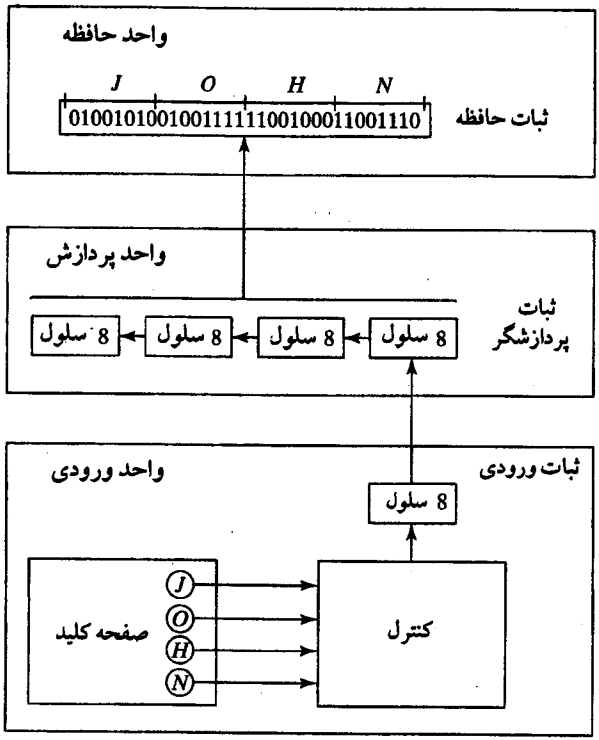
1100001111001001

یک ثبات با 16 سلول می‌تواند در یکی از 2^{16} حالت ممکن باشد. اگر فرض کنیم که محتوای یک ثبات

عدد صحیح دودویی را نشان می‌دهد، آنگاه ثبات می‌تواند هر عدد دودویی از 0 تا $2^{16}-1$ را ذخیره کند. برای مثال خاص فوق، محتوای ثبات عدد دودویی معادل با 50121 دهدهی است. اگر فرض کنیم که ثبات کاراکترهای کد هشت بیتی الفبا عددی را ذخیره کرده است، محتوای ثبات می‌تواند هر دو کاراکتر با معنی باشد. برای کد اسکی با توازن زوج واقع در هشتمین بیت باارزش، ثبات حاوی دو کاراکتر C (هشت بیت سمت چپ) و I (هشت بیت سمت راست) می‌باشد. از طرف دیگر اگر محتوای ثبات بصورت چهار رقم دهدهی تفسیر شود، محتوای ثبات یک عدد دهدهی چهار رقمی خواهد بود. درکد افزونی -3 مثال بالا عدد دهدهی 9096 است. درکد BCD این محتوا بی‌معنی است زیرا ترکیب بیتی 1100 به هیچ رقم دهدهی تخصیص نیافته است. با توجه به مثال، واضح است که یک ثبات قادر است اجزاء گسسته‌ای از اطلاعات را در خود ذخیره نماید و نیز آرایش بیتی یکسانی ممکن است برای انواع دیگر داده، تفسیر متفاوتی داشته باشد.

انتقال بین ثباتی

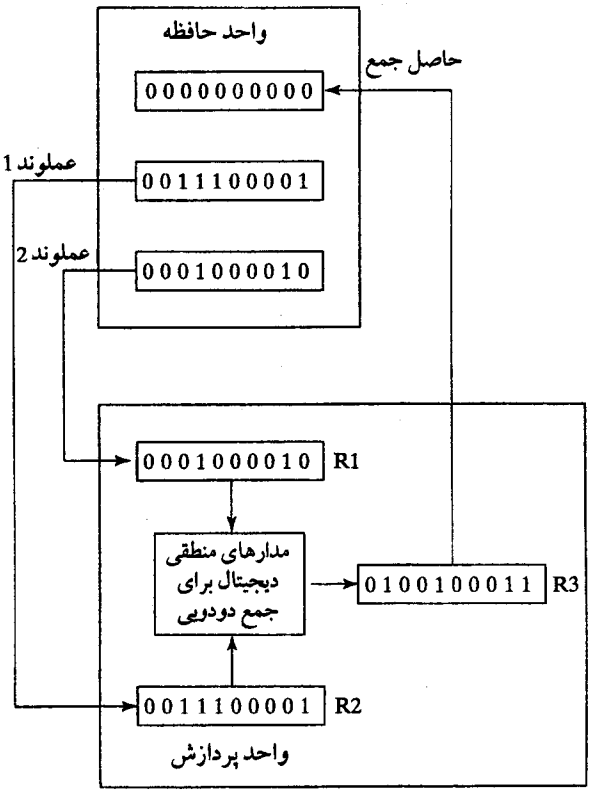
یک سیستم دیجیتال با ثبات‌هایش و قطعاتی که پردازش داده را اجرا می‌کنند مشخص می‌شود. عمل انتقال بین ثباتی یک عمل بنیادی در سیستم‌های دیجیتال است. این عمل متشکل از انتقال اطلاعات دودویی از یک مجموعه ثبات به مجموعه دیگر است. انتقال ممکن است مستقیماً از یک ثبات به دیگری باشد و یا از طریق مدارهای پردازش داده برای انجام یک عمل صورت گیرد. شکل ۱-۱ انتقال



شکل ۱-۱ . انتقال اطلاعات به وسیله ثباتها

اطلاعات را در میان ثبات‌ها نشان می‌دهد و نیز انتقال اطلاعات دودویی از یک صفحه کلید به یک حافظه به تصویر کشیده شده است. فرض می‌شود واحد ورودی دارای یک صفحه کلید، مدار کنترل و یک ثبات ورودی است. هر بار کلیدی فشرده شود، کنترل یک کد کاراکتر الفبا عددی هشت بیتی معادل را وارد ثبات ورودی می‌نماید. فرض می‌کنیم کد وارده از نوع اسکی و دارای توازن فرد باشد. اطلاعات از ثبات ورودی وارد هشت سلول کم‌ارزش‌تر یک ثبات پردازنده می‌گردد. پس از هر انتقال، ثبات ورودی پاک می‌شود تا کنترل بتواند پس از زدن کلید، کد هشت بیت جدید را وارد کند. قبل از ورود یا انتقال هر هشت بیت کاراکتر به ثبات پردازنده، کاراکتر قبلی به هشت سلول بعدی در سمت چپ خود منتقل می‌شود. وقتی کار انتقال چهار کاراکتر کامل شد، ثبات پردازنده پر شده و محتوای آن به یک ثبات حافظه منتقل می‌گردد. محتوای ذخیره شده در ثبات حافظه که در شکل ۱-۱ ملاحظه می‌گردد از انتقال کاراکترهای "J"، "O"، "H" و "N" پس از زدن چهار کلید مناسب حاصل شده است.

برای پردازش کمیت‌های گسسته‌ای از اطلاعات به فرم دودویی، کامپیوتر باید مجهز به وسایلی باشد تا داده مورد پردازش را حفظ و نیز بیت‌های اطلاعات را دستکاری کند. وسیله‌ای که برای نگهداری داده به کار می‌رود ثبات است. دستکاری متغیرهای دودویی به کمک مدارهای منطقی دیجیتال انجام می‌شود. شکل ۲-۱ فرآیند جمع دو عدد دودویی 10 بیتی را نشان می‌دهد. واحد حافظه که معمولاً



شکل ۲-۱ . مثالی برای پردازش اطلاعات دودویی

متشکل از میلیون‌ها ثبات است، در نمودار تنها با سه ثباتش نشان داده شده است. بخشی از واحد پردازشگر که در شکل آمده شامل سه ثبات R_1، R_2 و R_3 به همراه مدارهای منطقی دیجیتال است که بیت‌های ثبات R_1 و ثبات R_2 را دستکاری کرده و نتیجه را که یک حاصل جمع حسابی است به R_3 منتقل می‌سازد. ثبات‌های تشکیل دهنده حافظه اطلاعات را ذخیره می‌کنند و قادر نیستند دو عملوند را پردازش نمایند. با این وجود، اطلاعات ذخیره شده در حافظه قابل انتقال به ثبات‌های پردازشگر است. نتایج حاصل در ثبات‌های پردازنده می‌تواند مجدداً به ثبات‌های حافظه برای ذخیره کاربر بعدی باز فرستاده شود. نمودار، محتوای ارسال شده دو عملوند در ثبات‌های حافظه را به R_1 و R_2 نشان می‌دهد. مدارهای منطقی حاصل جمع را تولید می‌کنند، که بعد به ثبات R_3 منتقل می‌گردد. اکنون محتوای R_3 قابل بازگشت به یکی از ثبات‌های حافظه است.

دو مثال آخر، چگونگی توانمندی جریان اطلاعات را به صورت خیلی ساده تشریح کردند. ثبات‌های سیستم، اجزاء مبنا و اصلی ذخیره و نگهداری اطلاعات دودویی‌اند. مدارهای منطقی دیجیتال اطلاعات دودویی ذخیره شده در ثبات‌ها را پردازش می‌نمایند. مدارهای منطقی دیجیتال و ثبات‌ها در فصل‌های ۲ الی ۶ پوشش یافته‌اند. واحد حافظه در فصل ۷ توصیف شده است. انتقال در سطح ثبات که مورد استفاده در توصیف و طراحی سیستم‌هاست نیز در فصل ۸ بیان شده است.

۹-۱ منطق دودویی

منطق دودویی با متغیرهایی که دو ارزش گسسته و عملیاتی که مفهوم منطقی دارند، سروکار دارد. دو ارزشی که متغیرها اختیار می‌کنند ممکن است با اسامی مختلفی نامگذاری شوند (مثل صحیح و غلط، بله و خیر و غیره)، اما برای ما بهتر است آن را برحسب بیت تصور کنیم و مقادیر 1 و 0 را به آن تخصیص دهیم. منطق دودویی معرفی شده در این بخش معادل با جبری به نام جبر بول است. جبر بول در فصل ۲ به طور مشروح‌تر بیان شده است. در این بخش جبر بول به روشی غیرمستدل بوده و ارتباط آن با مدارهای منطقی دیجیتال و سیگنال‌های دودویی بیان شده است.

تعریف منطق دودویی

منطق دودویی شامل متغیرهای دودویی و عملیات منطقی است. متغیرها با حروف الفبایی مانند A، B، C، x، y، z و غیره نامگذاری می‌شوند، که هر متغیر فقط و فقط دو مقدار مجزای 0 و 1 دارد. سه نوع عملیات منطقی اصلی وجود دارند: AND، OR و NOT.

۱- AND: این عمل به وسیله یک "." یا بدون ذکر هر عملگری نمایش داده می‌شود. مثلاً $x.y = z$ یا $xy = z$ را چنین می‌خوانیم "x AND y برابر است با z". عمل منطقی AND چنین تفسیر می‌شود که، $z = 1$ است اگر و فقط اگر $x = 1$ و $y = 1$ باشد؛ در غیر این صورت $z = 0$ است. (به یاد داشته باشید که x و y و z متغیرهای دودویی هستند و نمی‌توانند به جز 1 و 0 چیز دیگری باشند.)

۲- OR: عملی است که با علامت بعلاوه نشان داده می‌شود. مثلاً $x + y = z$ را چنین می‌خوانیم

"x OR y برابر است با z" و به این معنی است که $z = 1$ است به شرطی که $x = 1$ یا $y = 1$ و یا هر دو $x = 1$ و $y = 1$ باشند. اگر هر دو $x = 0$ و $y = 0$ باشد آنگاه $z = 0$ خواهد بود.

۳- NOT : این عمل با یک علامت پریم نشان داده می‌شود (و گاهی با یک خط بار). مثلاً $x' = z$ (یا $\bar{x} = z$) و چنین خوانده می‌شود، "not x برابر است با z" و به این معنی است که z چیزی است که x نیست. به بیان دیگر اگر $x = 1$ باشد آنگاه $z = 0$؛ اما اگر $x = 0$ باشد، آنگاه $z = 1$ است. عمل NOT را متمم هم می‌گویند چون 1 را به 0 و 0 را به 1 تبدیل می‌کند.

منطق دودویی شبیه حساب دودویی است، و اعمال AND و OR به ترتیب به اعمال ضرب و جمع شباهت دارند. در حقیقت سمبل‌های به کار رفته برای AND و OR همان‌هایی هستند که برای ضرب و جمع مورد استفاده قرار می‌گیرند. معهذا منطق دودویی را نباید با حساب دودویی اشتباه کرد. مسئله‌ای که باید مورد توجه قرار گیرد این است که یک متغیر حسابی، عددی را مشخص می‌کند که ممکن است دارای چندین رقم باشد. یک متغیر منطقی همیشه 0 و یا 1 است. مثلاً در حساب دودویی داریم $1 + 1 = 10$ (می‌خوانیم: "یک بعلاوه یک برابر است با 2")، در صورتی که در منطق دودویی، داریم $1 + 1 = 1$ (می‌خوانیم: "یک OR یک، برابر است با 1").

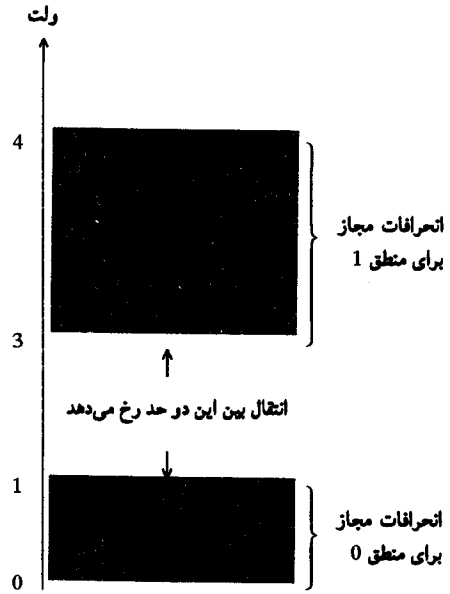
برای هر ترکیبی از مقادیر x و y، مقدار معینی برای z وجود دارد که این مقدار پس از اِعمال یا تعریف عمل منطقی مشخص می‌گردد. این تعاریف را می‌توان به صورت خلاصه یا استفاده از جدول درستی فهرست کرد. یک جدول درستی، جدولی است متشکل از تمام ترکیبات ممکن متغیرها و بیانگر ارتباط بین مقادیر آنها و نتایج حاصل از عمل مربوطه روی آنها می‌باشد. به عنوان مثال جداول درستی برای عملگرهای AND و OR با متغیرهای x و y، با لیست کردن همه مقادیر ممکن آنها وقتی به صورت زوج ترکیب شده‌اند، حاصل می‌شود. نتیجه عمل برای هر ترکیب به طور جداگانه آمده است. جداول درستی AND و OR و NOT در جدول ۸-۱ نشان داده شده‌اند. این جداول تعریف عملیات مذکور را به طور شفاف بیان می‌دارند.

گیت‌های منطقی

گیت‌های منطقی، مدارهایی الکترونیک هستند که روی یک یا چند سیگنال ورودی عمل می‌کنند تا یک سیگنال خروجی تولید نمایند. سیگنال‌های الکترونیکی مانند ولتاژها یا جریان‌هایی که در سرتاسر یک سیستم دیجیتال وجود دارند دو مقدار جدا از هم را اختیار می‌کنند. مدارهایی که با ولتاژ کار می‌کنند

جدول ۸-۱ . جدول درستی عملیات منطقی

AND			OR			NOT	
x	y	$x \cdot y$	x	y	$x + y$	x	x'
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

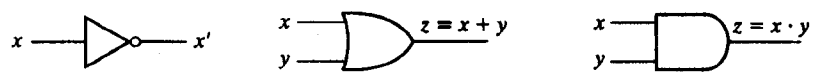


شکل ۳-۱ . مثالی از سیگنال‌های دودویی

به دو سطح ولتاژ که نمایشگر یک متغیر دودویی و برابر با منطق 1 و منطق 0اند واکنش نشان می‌دهند. مثلاً یک سیستم دیجیتال خاص ممکن است منطق 0را به عنوان سیگنالی برابر با 0ولت و منطق 1را به صورت سیگنالی برابر با 4 ولت تعریف کند. در عمل، هر سطح ولتاژ، محدوده مورد قبولی مانند شکل۳-۱ را داراست.

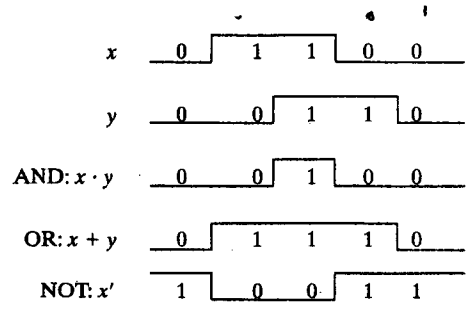
پایانه‌های ورودی مدارهای دیجیتال سیگنال‌های دودویی را در محدوده مُجازی می‌پذیرند و در پایانه‌های خروجی در محدوده مجازی پاسخ می‌دهند. ناحیه میانی بین دو ناحیه مجاز تنها هنگام گذر از یک حالت به حالت دیگر قطع می‌شود. هر اطلاعات محاسباتی یا کنترلی موردنظر را می‌توان با عبور سیگنال‌هایی دودویی از میان ترکیباتی از گیت‌ها مورد استفاده قرار داد، که هر سیگنال بیانگر یک متغیر دودویی بوده و یک بیت از اطلاعات را حمل می‌کند.

سمبل‌های گرافیکی مورد استفاده برای سه نوع گیت در شکل ۴-۱ دیده می‌شوند. گیت‌ها، بلوک‌هایی سخت‌افزاری‌اند که با ورودی منطقی مناسبی، در خروجی خود 0 یا 1 تولید می‌نمایند. سیگنال ورودی xو yدرگیت‌های ANDو OR در یکی از چهار حالت ممکن قرار دارند: 00، 01، 10 و



(الف) گیت ANDدو ورودی (ب) گیت OR دو ورودی (پ) گیت NOTبرای معکوس کردن (وارونگر)

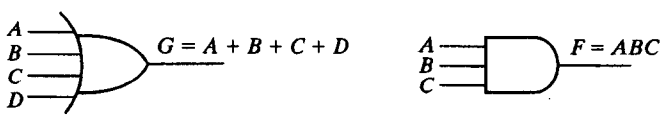
شکل ۴-۱. سمبل‌های مداری منطقی دیجیتال



شکل ۵-۱ . سیگنال‌های ورودی - خروجی برای گیت‌های (الف)، (ب) ، (پ) از شکل ۴-۱

11. این سیگنال‌ها همراه با خروجی خود در شکل ۵-۱ دیده می‌شوند. نمودارهای زمانی پاسخ هر گیت را برای چهار گیت فوق نشان می‌دهند. محور افقی نمودار زمانی، زمان و محور عمودی سیگنال‌ها را ضمن تغییر بین دو سطح ولتاژ ممکن نمایش می‌دهد. سطح پایین منطق 0 و سطح بالا منطق 1 را نشان می‌دهد. هنگامی در خروجی گیت AND منطق 1 وجود دارد که هر دو سیگنال ورودی در منطق 1 باشند. گیت OR هنگامی خروجی 1 دارد که یکی از سیگنال‌های ورودی در منطق 1 باشد. گیت NOT را معمولاً وارونگر یا معکوس‌گر هم می‌گویند. دلیل انتخاب این نام با توجه به پاسخ سیگنال در نمودار زمانی مشخص است، و در آن نشان داده شده است که سیگنال خروجی مفهوم منطق ورودی را معکوس کرده است.

گیت‌های AND و OR ممکن است بیش از دو ورودی داشته باشند. یک گیت AND با سه ورودی و یک OR با چهار ورودی در شکل ۶-۱ ملاحظه می‌شود. گیت AND سه ورودی به شرطی خروجی 1 دارد که هر سه ورودی آن 1 باشد. اگر هر یک از ورودی‌ها 0 باشند، خروجی AND برابر 0 خواهد بود. گیت OR چهار ورودی هنگامی خروجی 1 تولید می‌کند که یکی از ورودی‌ها در 1 منطقی باشد. خروجی هنگامی 0 می‌شود که همه ورودی‌ها در منطق 0 باشند.



(الف) گیت AND سه ورودی (ب) گیت OR چهار ورودی

شکل ۶-۱. مدار سوئیچینگ نمایش دهنده منطق دودویی

مسائل

۱ - ۱ اعداد مبنای 8 و 16 بین 16 تا 32 را لیست کنید. با استفاده از کاراکترهای A و B برای دو رقم آخر، اعداد 10تا 26 را در مبنای 12 لیست نمایید.

۱ - ۲ تعداد دقیق بایت‌ها در یک سیستم حاوی (الف) 32K بایتی، (ب) 64M بایتی و (پ) 6.4G بایتی چقدر است؟

۱ - ۳ بزرگترین عددی که می‌توان با 12 بیت نشان داد چند است؟ معادل دهدهی و شانزده شانزدهی آن چند است؟

۱ - ۴ اعداد زیر با مبناهای داده شده را به مبنای دهدهی تبدیل کنید: $(4310)_5$ و $(198)_{12}$.

۱ - ۵ برای صحیح بودن عملیات زیر، مبنای اعداد را در هر حالت معین کنید.
(الف) 14/2 = 5 ؛ (ب) 54/4 = 13 ؛ (پ) 24 + 17 = 40

۱ - ۶ حل معادله درجه دوم $x^2 - 11x + 22 = 0$ برابر $x = 3$ و $x = 6$ است. مبنای اعداد چیست؟

۱ - ۷ اعداد زیر را به دهدهی بیان کنید: $(10110.0101)_2$ ، $(16.5)_{16}$ و $(26.24)_8$.

۱ - ۸ اعداد دودویی زیر را به مبنای شانزده و مبنای 10 تبدیل نمایید:
(الف) 1.11010 ، (ب) 1110.10
بگویید چرا جواب دهدهی در (ب) 8 برابر (الف) است.

۱ - ۹ عدد مبنای شانزده 68BE را به دودویی و سپس از دودویی به مبنای هشت تبدیل کنید.

۱ - ۱۰ عدد دهدهی 345 را به دو روش به دودویی تبدیل نمایید:
(الف) مستقیماً به دودویی تبدیل نمایید. (ب) ابتدا به مبنای شانزده و سپس از مبنای شانزده به دودویی تبدیل سازید. کدام روش سریع‌تر است؟

۱ - ۱۱ تبدیل‌های زیر را عمل کنید:
(الف) عدد دهدهی 34.4375 را به دودویی تبدیل کنید.
(ب) معادل دودویی $\frac{1}{3}$ را تا 8 رقم بعد از نقطه ممیز بدست آورید. سپس از دودویی به دهدهی تبدیل کنید. جواب چقدر به $\frac{1}{3}$ نزدیک است؟
(پ) نتیجه دودویی در (ب) را به مبنای 16 ببرید. سپس نتیجه را به دهدهی تبدیل کنید. آیا جواب یکی است.

۱ - ۱۲ بدون تبدیل دهدهی جمع و ضرب‌های زیر را انجام دهید.
(الف) اعداد دودویی 1011 و 101 .
(ب) اعداد مبنای شانزده 2E و 34 .

۱ - ۱۳ تقسیم زیر را در دودویی انجام دهید: 101 : 1011111

۱ - ۱۴ متمم‌های 9 و 10 اعداد دهدهی زیر را بدست آورید.
(الف) 98127634 (ب) 72049900 (پ) 10000000 (ت) 00000000

۱ - ۱۵ (الف) متمم 16 عدد AF3B را بدست آورید.
(ب) AF3B را به دودویی تبدیل کنید.

(پ) متمم 2 جواب (ب) را پیدا کنید.

(ت) جواب در (پ) را به مبنای 16 تبدیل کنید و با پاسخ در (الف) مقایسه نمایید.

۱-۱۶ متمم 1 و 2 اعداد دودویی زیر را بدست آورید.

(الف) 11101010 (ب) 01111110 (پ) 00000001 (ت) 10000000 (ث) 00000000

۱-۱۷ روی اعداد بی‌علامت زیر با متمم 10 مفروق، تفریق انجام دهید. جایی که جواب منفی می‌شود، متمم 10 آن را بدست آورید و جلو آن علامت منها بگذارید. صحت پاسخ خود را تحقیق کنید.

(الف) 3049-7188 (ب) 2100-150 (پ) 7992-2997 (ت) 375-1321

۱-۱۸ روی اعداد دودویی بی‌علامت زیر با استفاده از متمم 2 مفروق، تفریق را اجرا کنید.

(الف) 11001−11011 (ب) 10101−110100

(پ) 110000−1011 (ت) 101011−101010

۱-۱۹ اعداد دهدهی زیر به فرم مقدار- علامت نشان داده شده است: 9826+ و 801+ . آنها را به متمم 10 تبدیل کنید و اعمال زیر را انجام دهید: (توجه کنید که حاصل جمع 10627+ بوده و شش رقم نیاز دارد).

(الف) (801+) + (9826+) (ب) (801−) + (9826+)

(پ) (801+) + (9826−) (ت) (801−) + (9826−)

۱-۲۰ عدد دهدهی 61+ و 27+ را با متمم 2_ علامت‌دار به دودویی تبدیل کنید و ارقام لازم را برای ذخیره اعداد در نظر بگیرید. سپس عبارات (61−) + (27+)، (61+) + (27−) و (61−) + (27−) را اجرا کنید. جواب‌ها را به دهدهی بازگردانید و تحقیق کنید که آنها صحیح‌اند.

۱-۲۱ دهدهی 9126 را به هر دو نوع کد BCD و اسکی تبدیل نمایید. برای اسکی، بیت توازن فرد در سمت چپ قرار داده می‌شود.

۱-۲۲ اعداد بی‌علامت دهدهی 965 و 672 را به BCD نشان دهید و سپس مراحل لازم برای جمع زدن آنها را طی کنید.

۱-۲۳ با استفاده از وزن‌های 1، 1، 3، 6 برای ارقام دهدهی یک کد دودویی وزین بسازید.

۱-۲۴ عدد دهدهی 6027 را (الف) به کد BCD، (ب) افزونی −3 و (پ) کد 2421 نشان دهید.

۱-۲۵ متمم 9 عدد 6027 را پیدا کنید و آن را به کد 2421 بیان نمایید. نشان دهید که نتیجه متمم 1، پاسخ (پ) در مسئله ۱-۲۴ است. این نشان می‌دهد که کد 2421 خود متمم است.

۱-۲۶ برای 52 ورق بازی نوعی کد دودویی تخصیص دهید.

۱-۲۷ عبارت "G.Boole" را با استفاده از یک کد هشت بیتی به اسکی بنویسید. بین حروف قسمت‌های مختلف نام و نام‌خانوادگی فاصله بگذارید. (جورج بول یک ریاضی‌دان قرن ۱۹ بود. جبر بول، که در فصل بعد معرفی شده است به اسم او نام‌گذاری شده است)

۱-۲۸ کدهای اسکی زیر را دیکد کنید: 1001010، 1100001، 1101110، 1100101، 0100000، 1000100، 1101111، 1100101.

۱-۲۹ در زیر رشته‌ای از کاراکترهای اسکی آمده‌اند که الگوی بیتی آن به منظور فشردگی به مبنای 16 تبدیل شده

است: 4A ، EF ، 68 ، 6E ، 20 ، C4 ، EF ، E5 . در میان 8 بیت در هر جفت رقم، سمت چپ‌ترین بیت مربوط به بیت توازن است. دیگر بیت‌ها کد اسکی را تشکیل می‌دهند.
(الف) هر یک را به فرم بیتی تبدیل کنید و سپس آن را به اسکی ببرید.
(ب) توازن به کار رفته را مشخص نمایید: فرد یا زوج.

۱-۳۰ در اسکی چند کاراکتر چاپ وجود دارد؟ چند تا از آنها کاراکترهای خاص هستند (نَه عدد و نَه حرف)؟

۱-۳۱ برای تبدیل حروف بزرگ به کوچک در اسکی کدام بیت باید عوض شود، و بالعکس؟

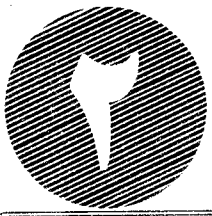
۱-۳۲ حالت یک ثبات 12 بیتی 100010010111 است. این محتوا برای حالات زیر چه چیز را نشان می‌دهد.
(الف) سه رقم دهدهی در BCD ؟ (ب) سه رقم دهدهی در کد افزونی 3 ؟
(پ) سه رقم دهدهی در کد 1-2-84 ؟ (ت) یک عدد دودویی؟

۱-۳۳ با بیت توازن زوج در سمت چپ‌ترین مکان، کد اسکی 10 رقم دهدهی را لیست کنید.

۱-۳۴ یک گیت AND سه ورودی با خروجی F و یک گیت OR سه ورودی با خروجی G را فرض کنید. ورودی‌ها A ، B و C اند. سیگنال‌های خروجی F و G ، به عنوان تابعی از سه ورودی ABC اند. هر 8 ترکیب ممکن را برای ABC به کار ببرید. (مشابه با نمودار زمان‌بندی شکل ۵-۱ عمل کنید).

مراجع

1. CAVANAGH, J. J. 1984. *Digital Computer Arithmetic.* New York: McGraw-Hill.
2. SCHMID, H. 1974. *Decimal Computation.* New York: John Wiley.
3. MANO, M. M. 1988. *Computer Engineering: Hardware Design.* Englewood Cliffs, NJ: Prentice-Hall.
4. NELSON, V. P., H. T. NAGLE, J. D. IRWIN, and B. D. CARROLL 1997. *Digital Logic Circuit Analysis and Design.* Upper Saddle River, NJ: Prentice Hall.



جبر بول و گیت‌های منطقی

۲-۱ تعاریف اولیه

جبر بول را می‌توان مانند هر سیستم منتجه ریاضی، به وسیله مجموعه‌ای از عناصر، یک مجموعه از الگوها و تعدادی اصول اثبات نشده یا بدیهیات تعریف نمود. یک مجموعه از عناصر کلکسیونی از اشیاء است که دارای خواص مشترکی باشند. اگر S یک مجموعه و x و y عناصر مشخصی از آن باشند، آنگاه $x \in S$ به این معنی است که x عضوی از مجموعه S و $y \notin S$ یعنی y عضوی از S نیست. یک مجموعه با تعداد قابل شمارشی از عناصر با یک جفت آکولاد مشخص می‌شود: $A = \{1, 2, 3, 4\}$، یعنی عناصر مجموعه A عبارتند از 1 ,2 ,3 و 4. یک عملگر دودویی روی یک مجموعه از عناصر، S، قانونی است که به هر جفت از عناصر S، یک عنصر منحصر به فرد از S را تخصیص دهد. به عنوان مثال رابطه $a * b = c$ را در نظر بگیرید. (*) را یک عملگر دودویی می‌خوانیم به شرطی که بتواند عنصر c را به جفت عنصر a و b منتسب نماید ضمن اینکه رابطه $a, b, c \in s$ معتبر باشد. با این وجود اگر $a, b \in S$ و $c \notin S$ باشد، (*) یک عملگر دودویی نیست.

اصول یک سیستم ریاضی، فرضیات اولیه را تشکیل می‌دهند که با استفاده از آنها می‌توان قوانین و تئوری‌ها و خواص سیستم را نتیجه گرفت. مهمترین اصول بکار رفته در فرموله کردن ساختارهای جبری عبارتند از:

۱- بسته بودن : یک مجموعه S نسبت به عملگر دودویی بسته است به شرطی که برای هر جفت عنصر از S، این عملگر عنصر منحصر به فردی از آن را به جفت عنصر منتسب نماید. به عنوان مثال، مجموعه اعداد طبیعی $N = \{1, 2, 3, 4, ...\}$ را نسبت به عملگر جمع (+) بسته گوییم زیرا برای هر دو عنصر $a \in s$ و b عنصر دیگری مانند $c \in N$ می‌توان یافت بطوری که $a+b= c$ باشد. مجموعه اعداد طبیعی نسبت به عملگر تفریق بسته نیست چون داریم $2 - 3 = -1$ در

حالی که $2, 3 \in N$ و $(-1) \notin N$ است.

۲- اصل شرکت‌پذیری : یک عملگر دودویی (*) روی مجموعه S شرکت‌پذیر است اگر داشته باشیم

$$(x * y) * z = x * (y * z) \qquad \text{به ازای همه مقادیر } x, y, z \in S$$

۳- اصل جابجایی : یک عملگر (*) روی یک مجموعه دارای خاصیت جابجایی است هرگاه

$$x * y = y * x \qquad \text{به ازای هر } x, y \in S$$

۴- عنصر شناسه : مجموعه S نسبت به عملگر (*) روی مجموعه S دارای عنصر شناسه است، اگر عنصر $e \in S$ با خاصیت زیر موجود باشد.

$$e * x = x * e = x \qquad \text{به ازای هر } x \in S$$

مثال: عنصر 0 یک عنصر شناسه نسبت به عملگر (+) روی مجموعه اعداد صحیح $I = \{..., -3, -2, -1, 0, 1, 2, 3, ...\}$ است، چون

$$x + 0 = 0 + x = x \qquad \text{به ازای هر } x \in I$$

مجموعه اعداد طبیعی N دارای عنصر شناسه نیست زیرا 0 جزو مجموعه نمی‌باشد.

۵- معکوس : مجموعه‌ای چون S با عنصر شناسه e نسبت به عملگر (*) دارای معکوس است به شرطی که برای هر $x \in S$، یک $y \in S$ وجود داشته باشد به نحوی که

$$x * y = e$$

مثال: در مجموعه اعداد صحیح I، با e=0، معکوس عنصر a برابر $(-a)$ است چون $a+(-a)=0$

۶- اصل توزیع‌پذیری : اگر (*) و (.) دو عملگر روی مجموعه S باشند، (*) را روی (.) توزیع‌پذیر گوییم هرگاه

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

مثالی جبری در این مورد میدان یا حوزه است. میدان مجموعه‌ای از عناصر است، همواره با دو عملگر دودویی، که هر یک دارای خواص 1 تا 5 بوده و هر دو عملگر برای تشکیل خاصیت 6 با یکدیگر ترکیب می‌شوند. مجموعه اعداد حقیقی، همراه با عملگرهای دودویی (+) و (.)، میدان اعداد حقیقی را تشکیل می‌دهند. میدان اعداد حقیقی مبنای جبر معمولی و حساب است. عملگرها و اصول دارای مفاهیم زیر هستند:

عملگر دودویی (+) جمع را تعریف می‌کند.

شناسه جمع، 0 است.

معکوس جمع، تفریق می‌باشد.

عملگر دودویی (.) ضرب را تعریف می‌نماید.

شناسه ضرب 1 می‌باشد.

معکوس ضرب $a = 1/a$ تقسیم را تعریف می‌کند، یعنی $a \cdot (1/a) = 1$.

تنها اصل توزیع‌پذیری قابل اعمال مربوط به عملگر (.) روی (+) است:

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

۲-۲ تعریف اصول اساسی جبر بول

در سال ۱۸۵۴ جورج بول یک برخورد سیستماتیک با منطق را معرفی نمود و برای اهداف خود یک سیستم جبری راکه امروزه آن را جبر بول می‌نامیم پایه‌ریزی کرد. در سال ۱۹۳۸ نیز سی.ای. شانون یک جبر بول دو مقداری به نام جبر سوئیچینگ را معرفی کرد که در آن خواص مدارهای سوئیچینگ با این جبر قابل ارائه است. برای تعریف مستدل جبر بول، ما اصول فرموله شده به وسیله ای. .ی. هانتینگتون در سال ۱۹۰۴ را بکار می‌بریم.

جبر بول یک ساختار جبری است که با عناصر مجموعه، یعنی B، همراه با دو عملگر دودویی (+) و (.) تعریف می‌شود به شرطی که اصول زیر (هانتینگتون) در آن معتبر باشد.

۱- (a) مجموعه نسبت به عملگر (+) بسته باشد.

(b) مجموعه نسبت به عملگر (.) بسته باشد.

۲- (a) یک عنصر شناسه 0 برای (+) وجود داشته باشد.

(b) یک عنصر شناسه 1 برای (.) وجود داشته باشد.

۳- (a) مجموعه نسبت به (+) دارای خاصیت جابجایی باشد: $x + y = y + x$

(b) مجموعه نسبت به (.) دارای خاصیت جابجایی باشد: $x \,.\, y = y \,.\, x$

۴- (a) (.) نسبت به (+) توزیع‌پذیر است: $x \,.\, (y + z) = (x \,.\, y) + (x \,.\, z)$

(b) (+) نسبت به (.) توزیع‌پذیر است: $x + (y \,.\, z) = (x + y) \,.\, (x + y)$

۵- برای هر عنصر $x \in B$، عنصری مثل $x' \in B$ وجود دارد (به آن متمم x می‌گوییم) به نحوی که:

(a) $x + x' = 1$ و (b) $x \,.\, x' = 0$

۶- حداقل دو عنصر $x, y \in B$ وجود دارد به نحوی که $x \neq y$ باشد.

با مقایسه جبر بول با حساب و جبر معمولی (حوزه یا میدان اعداد حقیقی) تفاوت‌های زیر قابل ملاحظه‌اند:

۱- اصول هانتینگتون فاقد اصل شرکت‌پذیری است. با این وجود، این اصول برای جبر بول معتبر و برای هر دو عملگر از دیگر اصول قابل استنتاج است.

۲- اصل توزیع‌پذیری (+) روی (.)، یعنی $x + (y \,.\, z) = (x + y) \,.\, (x + z)$، برای جبر بول معتبر است، ولی در جبر معمولی قابل قبول نیست.

۳- جبر بول دارای معکوس‌های جمع و ضرب نیست؛ بنابراین عملگرهای تفریق و تقسیم وجود ندارند.

۴- اصل ۵ عملگری به نام متمم را معرفی می‌نماید که در جبر معمولی وجود ندارد.

۵- جبر معمولی در مورد اعداد حقیقی بحث می‌کند، که یک مجموعه با بی‌نهایت عنصر را شامل می‌شود. جبر بول در مورد مجموعه‌ای از عناصر، B، بحث می‌نماید که هنوز آن را معرفی نکرده‌ایم، ولی بعداً در جبر بول دو مقداری یا دو ارزشی معرفی خواهد شد (کاربرد بعدی ما از این جبر مورد توجه است)، و در آن B بصورت مجموعه‌ای از دو عنصر 0 و 1 تعریف می‌شود.

جبر بول از بعضی لحاظ با جبر معمولی هماهنگی دارد. انتخاب سمبل‌های (+) و (.) به این علت مورد توجه قرار گرفته‌اند تا دستکاری جبر بول به وسیله شخصی که با جبر معمولی آشنایی دارد، ساده گردد. گرچه وی می‌تواند بخشی از دانش خود از جبر معمولی را در جبر بول بکار ببرد، ولی مبتدیان باید مراقب باشند تا قوانین جبر معمولی را در جاهایی که قابل استفاده نیستند، بکار نبرند.

لازم است تا اختلاف بین یک مجموعه از عناصر متعلق به یک ساختار جبری و متغیرهای یک سیستم جبری را بدانیم. مثلاً، اجزاء حوزه اعداد حقیقی، اعداد هستند، در صورتی که متغیرهایی مانند a ، b و c که در جبر معمولی بکار می‌روند، سمبل‌هایی هستند که به جای اعداد حقیقی بکار می‌روند. بطور مشابه در جبر بول مجموعه B دارای متغیرهایی مانند x، y و z می‌باشد که صرفاً سمبل هستند و عناصر را نشان می‌دهند. در اینجا لازم است بدانیم که به منظور داشتن یک جبر بول باید:

۱- عناصر مجموعه B مشخص باشند.

۲- قوانین عمل دو عملگر دودویی معین باشند.

۳- مجموعه عناصر، B ، همراه با دو عملگر، برای شش اصل هانتینگتون معتبر باشد.

بسته به انتخاب عناصر B و قوانین عملیات، می‌توان چندین جبر بول را فرموله کرد. در ادامه کار ما فقط با جبر دو ارزشی که تنها دو عنصر دارد، سروکار خواهیم داشت. جبر بول دو ارزشی در تئوری مجموعه‌ها و منطق کاربرد دارد. هدف ما در این کتاب کاربرد جبر بول در مدارهای منطقی گیتی است.

جبر بول دو ارزشی

جبر بول دو ارزشی روی مجموعه دو عنصری، $B = \{0, 1\}$ ، به همراه قوانین برای دو عملگر دودویی (+) و (.) که در جدول زیر نشان داده شده تعریف می‌شود (قانون عملگر متمم برای تصدیق اصل ۵ است):

x	y	x · y
0	0	0
0	1	0
1	0	0
1	1	1

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

x	x′
0	1
1	0

این قوانین دقیقاً مثل اعمال AND، OR و NOT در جدول ۸-۱ می‌باشند. اکنون نشان می‌دهیم که اصول هانتینگتون برای مجموعه $B=\{0, 1\}$ و دو عملگرهای دودویی که قبلاً تعریف شده‌اند، معتبر است.

۱- با توجه به جداول، بسته بودن کاملاً روشن است زیرا نتیجه هر عملگر 1 یا 0 بوده و $1,0 \in B$ می‌باشند.

۲- با توجه به جداول می‌بینیم که $0 + 1 = 1 + 0 = 1$ (a) $0 + 0 = 0$

(b) $1.1 = 1$ $1.0 = 0.1 = 0$

این روابط بیانگر وجود عناصر شناسه 0 برای (+) و 1 برای (.) ، طبق تعریف اصل ۲ می‌باشند.

۳- اصول جابجایی با توجه به تقارن جداول عملگرها آشکار است.

۴- (a) صحت اصل توزیع‌پذیری $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$ را با ایجاد جدول برای تمام مقادیر ممکن x, y و z، می‌توان تحقیق کرد. برای هر ترکیب، $x \cdot (y + z)$ را بدست آورده و نشان می‌دهیم که برابر $(x \cdot z) + (x \cdot y)$ است.

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

(b) صحت اصل توزیع‌پذیری (+) روی (.) را نیز می‌توان مثل بند قبل تحقیق نمود.

۵- با استفاده از جدول متمم، به سادگی می‌توان دید که:

(a) $x + x' = 1$، زیرا $1 + 1 = 0 + 0' = 0$ و $1 + 1' = 1 + 0 = 1$ است

(b) $x \cdot x' = 0$، زیرا $0.1 = 0.0' = 0$ و $1.1' = 1.0 = 0$ است، که اصل ۵ را تصدیق می‌کند.

۶- اصل ۶ نیز صادق است زیرا جبر بول دو ارزشی دارای دو مقدار مجزای 0 و 1 با $0 \neq 1$ است.

تا اینجا ما یک جبر دو ارزشی با عملگرهای AND و OR و یک عملگر متمم معادل با NOT ایجاد کردیم. بنابراین جبر بول به روش مستدل ریاضی بنا گردید و نشان داده شد که معادل با منطق دودویی غیرمستدل بخش ۹-۱ است. بیان غیرمستدل برای درک کاربرد جبر بول در مدارهای گیتی مفید است. روش مستدل برای بیان و ایجاد تئوری‌ها و خواص سیستم جبری مورد توجه است. جبر بول دو ارزشی تعریف شده در این بخش را "جبر سوئیچینگ" نیز می‌نامند. برای تأکید بر تشابه بین جبر بول دو ارزشی و دیگر سیستم‌های دودویی، این جبر در بخش ۹-۱ "منطق دودویی" نامیده شد. از این پس، کلمه "دو ارزشی" را در بحث‌های بعدی از جبر بول حذف می‌کنیم.

۳-۲ قضایای اصلی و خواص جبر بول

دوگانگی

اصول هانتینگتون بصورت جفت جفت لیست شده‌اند و بصورت بخش‌های (a) و (b) مشخص شدند. هر یک از این دو را با تعویض عملگرها و عنصر شناسه می‌توان از دیگری بدست آورد. این خاصیت در جبر بول به اصل دوگانگی معروف است. خصوصیات فوق بیان می‌دارد که هر عبارت جبری منتج از اصول جبر بول با تعویض عملگرها و شناسه‌ها باز هم معتبر باقی می‌ماند. در جبر بول دو ارزشی، عناصر شناسه و عناصر مجموعه B یکسانند: یعنی 1 و 0اند. اصل دوگانگی کاربردهای متعددی دارد. اگر دوگان

یک عبارت جبری موردنظر باشد تنها کافی است عملگرهای AND و OR تعویض و 0ها به 1 و 1ها به 0 تبدیل گردند.

تئوری‌های اساسی

جدول (۱-۲) شش تئوری و چهار اصل از جبر بول را لیست کرده است. به خاطر سادگی از علامت (.) مادامی که اشتباه ایجاد نشود، صرف‌نظر شده است. تئوری‌ها و اصول لیست شده اساسی‌ترین روابط در جبر بول‌اند. تئوری‌ها نیز مانند اصول بصورت جفت جفت ارائه شده‌اند و هر رابطه دوگان زوج خود است. اصول، بدیهیات ساختار جبری بوده و اثباتی لازم ندارند. تئوری‌ها باید با توجه به اصول ثابت شوند، اثبات تئوری‌ها با یک متغیر در زیر نشان داده شده‌اند. در سمت مقابل روابط ، شماره اصل بکار رفته نوشته شده است.

تئوری ۱ (a): $x + x = x$

$$
\begin{aligned}
x + x &= (x + x) \cdot 1 && \text{به وسیله اصل: ۲ (b)}\\
&= (x + x)(x + x') && \text{(a) ۵}\\
&= x + xx' && \text{(b) ۴}\\
&= x + 0 && \text{(b) ۵}\\
&= x && \text{(a) ۲}
\end{aligned}
$$

تئوری ۱ (b): $x \cdot x = x$

$$
\begin{aligned}
x \cdot x &= xx + 0 && \text{به وسیله اصل: ۲ (a)}\\
&= xx + xx' && \text{(b) ۵}\\
&= x(x + x') && \text{(a) ۴}\\
&= x \cdot 1 && \text{(a) ۵}\\
&= x && \text{(b) ۲}
\end{aligned}
$$

جدول ۱-۲. اصول و تئوری‌های جبر بول

اصل ۲	(a)	$x + 0 = x$	(b)	$x \cdot 1 = x$
اصل ۱	(a)	$x + x' = 1$	(b)	$x \cdot x' = 0$
تئوری ۱	(a)	$x + x = x$	(b)	$x \cdot x = x$
تئوری ۲	(a)	$x + 1 = 1$	(b)	$x \cdot 0 = 0$
تئوری ۳ رجعت		$(x')' = x$		
اصل ۳ جابجایی	(a)	$x + y = y + x$	(b)	$xy = yx$
تئوری ۴ شرکت پذیری	(a)	$x + (y + z) = (x + y) + z$	(b)	$x(yz) = (xy)z$
اصل ۴ توزیع پذیری یا پخش	(a)	$x(y + z) = xy + xz$	(b)	$x + yz = (x + y)(x + z)$
تئوری ۵ دمورگان	(a)	$(x + y)' = x'y'$	(b)	$(xy)' = x' + y'$
تئوری ۶ جذب	(a)	$x + xy = x$	(b)	$x(x + y) = x$

توجه کنید که تئوری ۱ (b) دوگان ۱ (a) است و هر مرحله از اثبات در بخش (b) دوگان بخش (a) می‌باشد. به این ترتیب هر تئوری دوگان از اثبات زوجش حاصل می‌گردد.

تئوری ۲ (a): $x + 1 = 1$

$x + 1 = 1 \cdot (x + 1)$	به وسیله اصل : ۲ (b)
$= (x + x')(x + 1)$	۵ (a)
$= x + x' \cdot 1$	۴ (b)
$= x + x'$	۲ (b)
$= 1$	۵ (a)

تئوری ۲ (b): براساس دوگانگی $x \cdot 0 = 0$

تئوری ۳: $x = (x')'$. با توجه به اصل ۵، داریم $x + x' = 1$ و $x \cdot x' = 0$، که متمم x را تعریف می‌کند. متمم x' برابر x است و در نتیجه همان $(x')'$ می‌باشد. بنابراین، چون متمم منحصر به فرد است داریم $(x')' = x$.

با استفاده از اصول و تئوری‌های اثبات شده قبلی می‌توان تئوری‌های دو یا سه متغیره را بصورت جبری ثابت کرد. مثلاً: تئوری جذب را در نظر بگیرید.

تئوری ۶ (a): $x + xy = x$

$x + xy = x \cdot 1 + xy$	به وسیله اصل : ۲ (b)
$= x(1 + y)$	۴ (a)
$= x(y + 1)$	۳ (a)
$= x \cdot 1$	۲ (a)
$= x$	۲ (b)

تئوری ۶ (b): براساس دوگانگی $x(x + y) = x$

صحت تئوری‌های جبر بول را می‌توان با کمک جداول درستی هم تحقیق کرد. در جداول درستی، هر دو طرف رابطه چک می‌شوند تا نتایج مشابهی در ازاء همه ترکیبات متغیرها حاصل شود. تئوری جذب زیر صحت اولین جدول درستی را نشان می‌دهد.

x	y	xy	$x + xy$
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

اثبات جبری اصل شرکت‌پذیری و تئوری دمورگان طولانی بوده و در اینجا آورده نمی‌شوند. با این وجود اعتبار آنها با جداول درستی نشان داده می‌شود. مثلاً جدول درستی برای اولین تئوری دمورگان

$(x + y)' = x'y'$ در زیر نشان داده شده است.

x	y	$x + y$	$(x + y)'$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

x'	y'	$x'y'$
1	1	1
1	0	0
0	1	0
0	0	0

تقدم عملگرها

در ارزیابی عبارات جبر بول تقدم اول با پرانتز، دوم با NOT، سوم با AND و چهارم با OR است. به بیان دیگر، عبارت داخل پرانتز باید قبل از سایر عملگرها ارزیابی شود. عملگر مقدم بعدی متمم است. پس از آن AND و بالاخره OR قرار دارد. به عنوان مثال، جدول درستی را برای تئوری دمورگان تشکیل می‌دهیم سمت چپ عبارت $(x + y)'$ است. بنابراین داخل پرانتز ابتدا ارزیابی می‌شود و سپس نتیجه متمم می‌گردد. سمت راست عبارت $x'y'$ است. بنابراین متمم x و y هر دو ابتدا ارزیابی شده و حاصل AND می‌گردد. توجه کنید که در محاسبات معمولی هم روال مشابهی (به جز برای متمم) برای ضرب و جمع به ترتیب به جای AND و OR برقرار است.

۲-۴ توابع بول

جبر بول جبری است که با متغیرهای دودویی و عملیات منطقی سروکار دارد. یک تابع به وسیله یک عبارت جبری متشکل از متغیرهای دودویی، ثابت‌های 0 و 1 و سمبل‌های عملیاتی منطقی تشکیل شده است. برای مقدار مفروضی از متغیرهای دودویی، تابع می‌تواند 1 یا 0 باشد. به عنوان مثال تابع بولی زیر را در نظر بگیرید:

$$F_1 = x + y'z$$

اگر x برابر 1 باشد و یا هر دو y' و z برابر 1 باشند تابع F_1 برابر 1 است. در غیر این صورت F_1 برابر 0 خواهد بود. عمل متمم، وقتی $y' = 1$ باشد، $y = 0$ را دیکته می‌کند. بنابراین، می‌توانیم بگوییم که اگر $x = 1$ یا اگر $y = 0$ و $z = 1$ باشد آنگاه $F_1 = 1$ خواهد بود. یک تابع بول رابطه‌ای منطقی را بین متغیرها بیان می‌کند. این تابع با تعیین مقدار دودویی عبارت برحسب همه مقادیر ممکن متغیرها ارزیابی می‌شود.

یک جدول بولی بصورت یک جدول درستی هم می‌تواند نشان داده شود. جدول درستی لیستی از 1ها و 0ها است که به متغیرهای دودویی تخصیص می‌یابد، و ستونی که مقدار نتایج را برای هر ترکیب نشان می‌دهد. تعداد سطرها در جدول درستی 2^n است، که n تعداد متغیرها در تابع است. ترکیبات دودویی برای جدول درستی از شمارش اعداد دودویی و از 0 تا $2^n - 1$ بدست می‌آید. جدول (۲-۲)، جدول درستی تابع F_1 را نشان می‌دهد. در این جدول هشت ترکیب دودویی ممکن برای تخصیص بیتی

جدول ۲-۲. جدول درستی برای F_1 و F_2

x	y	z	F_1	F_2
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	1
1	0	0	1	1
1	0	1	1	1
1	1	0	1	0
1	1	1	1	0

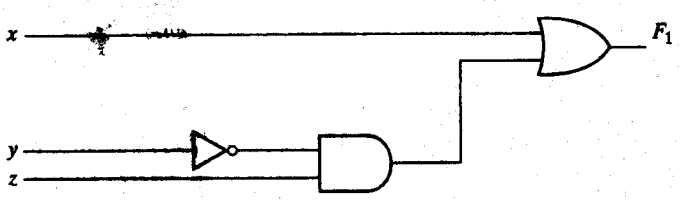
به سه متغیر x و y و z وجود دارد. ستونی که برچسب F_1 دارد در ازاء هر ترکیب 0 یا 1 است. جدول نشان می‌دهد که وقتی $x = 1$ یا $yz = 01$ باشد تابع F_1 برابر 1 است. در غیر این صورت 0 خواهد بود.

یک تابع بول را می‌توان از یک عبارت جبری به یک نمودار مداری متشکل از گیت‌های منطقی تبدیل کرد. نمودار مدار منطقی F_1 در شکل ۱-۲ نشان داده شده است. برای تولید متمم ورودی y وارونگری (معکوس‌گر) وجود دارد. برای جمله $y'z$ یک گیت AND و برای ترکیب آن دو یک گیت OR بکار رفته است. در نمودارهای مدار منطقی، متغیرهای تابع به عنوان ورودی مدار و متغیر دودویی F_1 به عنوان خروجی مدار در نظر گرفته می‌شوند.

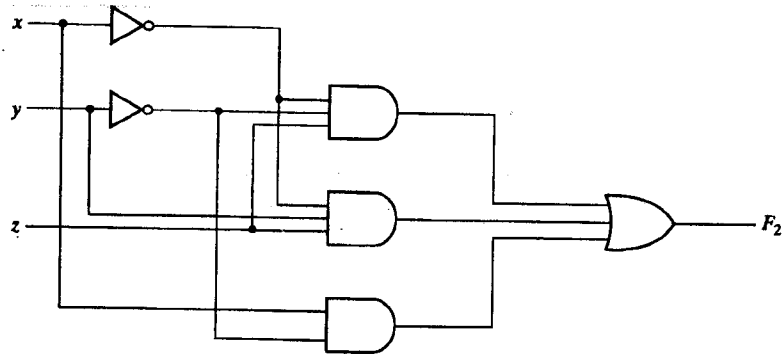
برای نمایش F_1 در یک جدول درستی تنها یک راه وجود دارد. با این وجود، وقتی تابع به فرم یک عبارت جبری است، می‌تواند به فرم‌های متفاوتی نشان داده شود. عبارت خاصی که برای مشخص کردن تابع مورد استفاده قرار می‌گیرد اتصالات میان گیت‌ها در نمودار مدار منطقی را دیکته می‌نماید. گاهی اوقات ممکن است با دستکاری یک عبارت بولی توسط قوانین جبر بول، عبارت ساده‌تری برای یک تابع بدست آوریم و بنابراین تعداد گیت‌ها در مدار و تعداد ورودی‌ها به هر گیت را کاهش دهیم. مثلاً تابع بولی زیر را در نظر بگیرید:

$$F_2 = x'y'z + x'yz + xy'$$

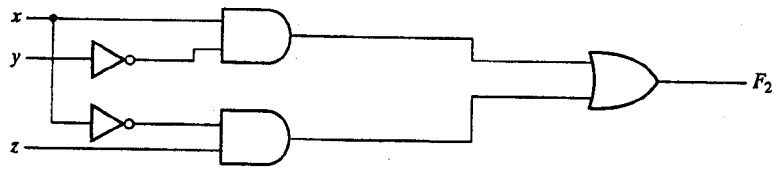
این تابع در شکل ۲-۲ (الف) پیاده‌سازی شده است. متغیرهای x و y به کمک وارونگر متمم شده‌اند تا x' و y' بدست آیند. سه جمله در عبارت با سه گیت AND پیاده‌سازی شده‌اند. گیت OR نیز، OR منطقی سه جمله را فراهم می‌سازد. جدول درستی F_2 در جدول ۲-۲ آمده است. وقتی که $xyz = 001$ یا 011 و یا $xy = 10$ باشد (بدون توجه به z)، تابع برابر 1 است، در غیر این صورت 0 است. این شرایط چهار 1 و چهار 0 برای F_2 تولید می‌کنند.



شکل ۱-۲. پیاده‌سازی با گیت $F_1 = x + y'z$



$F_2 = x'y'z + x'yz + xy'$ (الف)



$F_2 = xy' + x'z$ (ب)

شکل ۲-۲. پیاده‌سازی تابع بول F_2 با گیت

اکنون ساده‌سازی ممکن برای تابع را با اعمال بعضی از ویژگی‌های جبر بول ملاحظه کنید:

$$F_2 = x'y'z + x'yz + xy' = x'z(y' + y) + xy' = x'z + xy'$$

تابع تنها به دو جمله کاهش یافته و قابل پیاده‌سازی با گیت مطابق شکل ۲-۲ (ب) است. بدیهی است که مدار شکل (ب) ساده‌تر از (الف) می‌باشد، ولی هر دو یک تابع را پیاده‌سازی می‌کنند. تساوی دو عبارت را می‌توان به کمک جدول درستی هم تحقیق کرد. عبارت ساده شده، وقتی xz = 01 یا xy = 10 باشد، برابر 1 است. این تابع هم همان چهار 1 را در جدول تولید می‌کند. چون هر دو عبارت جدول درستی یکسانی را تولید می‌کنند به آنها معادل گوییم. بنابراین، دو مدار به ازاء همه ترکیبات ممکن متغیرهای ورودی، خروجی‌های یکسانی دارند. هر دو عبارت تابع یکسانی را تولید می‌کنند ولی یکی از آنها گیت‌ها و ورودی‌های کمتری نسبت به دیگری دارد و بنابراین چون سیم‌بندی و قطعات کمتری نیاز است بر دیگری ترجیح داده می‌شود.

دستکاری جبری

وقتی که یک عبارت بولی با گیت‌های منطقی پیاده‌سازی شود، هر جمله به یک گیت نیاز دارد و هر متغیر در جمله یک ورودی به یک گیت است. ما لیترال را یک متغیر تک در یک جمله می‌نامیم که ممکن است متمم شود یا نشود. مثلاً تابع شکل ۲-۲ (الف) دارای سه جمله و هشت لیترال است، دو تابع شکل ۲-۲ (ب) دو جمله و چهار لیترال دارد. اغلب در تابع بول با کاهش تعداد جملات، تعداد لیترال‌ها، یا

هر دو مدار ساده‌تری حاصل می‌شود. هدف از دستکاری جبر بول غالباً کاهش یک عبارت به منظور دستیابی به یک مدار ساده‌تر است. توابعی که تا پنج متغیر دارند قابل ساده‌سازی با روش نقشه که در فصل بعد آمده، هستند. برای توابع بول پیچیده‌تر، طراحان دیجیتال از برنامه‌های کامپیوتر کوچک‌سازی استفاده می‌کنند. تنها روش دستی موجود، روال سعی و کاهش می‌باشد که از روابط ساده و تکنیک‌های دستکاری آشنا استفاده می‌کند. مثال‌های زیر دستکاری جبری جبر بول را نشان می‌دهد.

مثال ۱-۲

توابع بولی زیر را با حداقل لیترال‌ها ساده کنید.

۱- $x(x' + y) = xx' + xy = 0 + xy = xy.$

۲- $x + x'y = (x + x')(x + y) = 1(x + y) = x + y.$

۳- $(x + y)(x + y') = x + xy + xy' + yy' = x(1 + y + y') = x.$

۴-
$$\begin{aligned} xy + x'z + yz &= xy + x'z + yz(x + x') \\ &= xy + x'z + xyz + x'yz \\ &= xy(1 + z) + x'z(1 + y) \\ &= xy + x'z. \end{aligned}$$

۵- از دوگان تابع ۴ $(x + y)(x' + z)(y + z) = (x + y)(x' + z)$

❑

توابع ۱ و ۲ دوگان یکدیگرند، و در مراحل حلی خود نیز از خاصیت دوگانگی استفاده کرده‌اند. راه ساده‌تری برای ساده‌سازی تابع ۳ به کمک اصل ۴ (b) در جدول ۱-۲ آمده است: یعنی $(x + y)(x + y') = x + yy' = x$. چهارمین تابع این واقعیت را بیان می‌دارد که یک افزایش در تعداد لیترال‌ها گاهی منجر به عبارت نهایی ساده‌تر می‌گردد. تابع ۵ مستقیماً ساده نشده است، ولی می‌تواند با استفاده از دوگان مراحل مربوط به تابع ۴ بدست آید. توابع ۴ و ۵ را به عنوان تئوری وفاق می‌شناسند.

متمم یک تابع

متمم یک تابع F برابر F' است و از تعویض 0ها با 1 و 1ها با 0 در مقدار F بدست می‌آید. متمم یک تابع را می‌توان بصورت جبری از تئوری دمورگان نیز بدست آورد. جفت تئوری مذکور در جدول ۱-۲ برای دو متغیر داده شد. تئوری‌های دمورگان به سه یا چند متغیر هم قابل گسترش‌اند. با استفاده از اصول و تئوری‌های لیست شده در جدول ۱-۲، فرم سه متغیره اولین تئوری دمورگان به طریق زیر ثابت می‌شود.

$$\begin{aligned} (A + B + C)' &= (A + x)' && \text{با فرض } B + C = x \\ &= A'x' && \text{با تئوری ۵ (a) دمورگان} \\ &= A'(B + C)' && B + C = x \text{ را جایگزین کنید} \\ &= A'(B'C') && \text{با تئوری ۵ (a) دمورگان} \\ &= A'B'C' && \text{با تئوری ۴ (b) شرکت‌پذیری} \end{aligned}$$

تئوری‌های دمورگان برای هر تعدادی از متغیرها، مشابه حالت دو متغیره بوده و با روش جایگزینی متوالی، مشابه روشی که در فوق مشاهده شد، می‌توان آن را بدست آورد. فرم عمومی تئوری دمورگان به صورت زیر است:

$$(A + B + C + D + \ldots + F)' = A'B'C'D' \ldots F'$$
$$(ABCD \ldots F)' = A' + B' + C' + D' + \ldots + F'$$

این تئوری بیان می‌دارد که متمم یک تابع با تعویض عملگرهای AND و OR و متمم کردن هر لیترال حاصل می‌شود.

مثال ۲-۲

متمم توابع $F_1 = x'yz' + x'y'z$ و $F_2 = x(y'z' + yz)$ را بدست آورید. متمم‌ها را با اعمال هر تعداد تئوری دمورگان بصورت زیر بدست آورید:

$$F_1' = (x'yz' + x'y'z)' = (x'yz')'(x'y'z)' = (x + y' + z)(x + y + z')$$
$$F_2' = [x(y'z' + yz)]' = x' + (y'z' + yz)' = x' + (y'z')'(yz)'$$
$$= x' + (y + z)(y' + z')$$

□

روال ساده‌تری برای بدست آوردن متمم یک تابع این است که دوگان تابع و متمم هر لیترال بدست آید. این روش با توجه به فرم عمومی تئوری دمورگان نتیجه می‌شود. به خاطر داشته باشید که دوگان یک تابع با تبدیل عملگر AND به OR و تبدیل 1ها و 0ها به یکدیگر بدست می‌آید.

مثال ۲-۳

متمم توابع F_1 و F_2 مثال ۲-۲ را با استفاده از دوگان‌ها و متمم‌های هر لیترال بدست آورید.

۱) $F_1 = x'yz' + x'y'z'$

دوگان تابع F_1 برابر است با: $(x' + y' + z')\ (x' + y' + z)$

متمم هر لیترال برابر است با: $(x + y' + z)\ (x + y + z') = F_1'$

۲) $F_2 = x(y'z' + yz)$

دوگان تابع F_2 برابر است با: $x + (y' + z')\ (y + z)$

متمم هر لیترال برابر است با: $x' + (y + z)\ (y' + z') = F_2'$

□

۲-۵ فرم‌های استاندارد و متعارف

مینترم‌ها و ماکسترم‌ها

یک متغیر دودویی ممکن است به فرم معمولی (x) یا متمم (x') ظاهر شود. اکنون تصور کنید که دو

جدول ۳-۲ . مینترم‌ها و ماکسترم‌ها برای سه متغیر دودویی

			مینترم‌ها		ماکسترم‌ها	
x	y	z	جمله	علامت	جمله	علامت
0	0	0	$x'y'z'$	m_0	$x + y + z$	M_0
0	0	1	$x'y'z$	m_1	$x + y + z'$	M_1
0	1	0	$x'yz'$	m_2	$x + y' + z$	M_2
0	1	1	$x'yz$	m_3	$x + y' + z'$	M_3
1	0	0	$xy'z'$	m_4	$x' + y + z$	M_4
1	0	1	$xy'z$	m_5	$x' + y + z'$	M_5
1	1	0	xyz'	m_6	$x' + y' + z$	M_6
1	1	1	xyz	m_7	$x' + y' + z'$	M_7

متغیر دودویی x و y با عملگر AND با هم ترکیب شوند. چون هر متغیر ممکن است به هر یک از دو شکل فوق ظاهر گردد، چهار ترکیب برای آنها متصور است: $x'y'$، $x'y$، xy' و xy. هر یک از این چهار جمله AND را یک مینترم یا یک جمله ضرب استاندارد گویند. بطور مشابه n متغیر را می‌توان ترکیب کرده و 2^n مینترم به وجود آورد. 2^n مینترم مختلف را می‌توان با روشی مشابه با آنچه در جدول ۳-۲ آمده، نشان داد. اعداد دودویی از صفر تا $2^n - 1$ زیر ستون n متغیر لیست شده‌اند. هر مینترم از AND تمام n متغیر بدست می‌آید که در آن هر متغیر پریم‌دار متعلق به بیت 0 و بدون پریم با 1 نشان داده می‌شود. سمبل هر مینترم نیز در جدول با m_j نشان داده شده است. که در آن J معادل دهدهی عدد دودویی مربوط به مینترم است.

به طریق مشابهی، n متغیر یک جمله OR تشکیل می‌دهند که هر متغیر ممکن است پریم‌دار یا بدون پریم باشد. 2^n ترکیب ممکن را ماکسترم یا جمع استاندارد گویند. هشت ماکسترم برای سه متغیر، همراه با سمبل آنها در جدول ۳-۲ لیست شده‌اند. هر 2^n ماکسترم برای n متغیر بطریق مشابهی حاصل می‌شود. هر ماکسترم از یک جمله OR با n متغیر بدست می‌آید که در آن متغیر پریم‌دار با 1 و بدون پریم با 0 نشان داده می‌شود. توجه کنید که هر ماکسترم، متمم مینترم مربوطه‌اش می‌باشد و بالعکس.

یک تابع بول می‌تواند بصورت جبری با استفاده از جدول درستی و با تشکیل مینترم‌های هر ترکیب از متغیرهایی که برای تابع، 1 را تولید می‌کنند، و اجرای عملگر روی OR همه این جملات ایجاد شود. مثلاً f_1 در جدول ۴-۲ با ترکیبات 001، 100 و 111 بصورت $x'y'z$, $xy'z'$ و xyz بیان می‌شود. چون هر

جدول ۴-۲ . توابع سه متغیره تابع f_1 تابع f_2

x	y	z	تابع f_1	تابع f_2
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

یک از این مینترم‌ها $f_1 = 1$ را ایجاد می‌نمایند پس:

$$f_1 = x'y'z + xy'z' + xyz = m_1 + m_4 + m_7$$

به سادگی می‌توان نشان داد که:

$$f_2 = x'yz + xy'z + xyz' + xyz = m_3 + m_5 + m_6 + m_7$$

این مثال‌ها خصوصیت مهمی از جبر بول را به نمایش می‌گذارند: یعنی هر تابع بولی را می‌توان بصورت جمع مینترم‌ها نشان داد "جمع به معنی OR جملات است".

اکنون متمم تابع بول را ملاحظه نمایید، می‌توان آن را با تشکیل مینترم‌هایی در جدول درستی که 0 تابع را تولید می‌کنند، ایجاد کرد و سپس آنها را OR نمود. متمم f_1 چنین است:

$$f_1' = x'y'z' + x'yz' + x'yz + xy'z + xyz'$$

اگر متمم f'_1 را بدست آوریم تابع f_1 بدست خواهد آمد.

$$\begin{aligned} f_1 &= (x + y + z)(x + y' + z)(x' + y + z')(x' + y' + z) \\ &= M_0 \cdot M_2 \cdot M_3 \cdot M_5 \cdot M_6 \end{aligned}$$

بطور مشابه، می‌توان عبارت f_2 را از جدول بدست آورد:

$$\begin{aligned} f_2 &= (x + y + z)(x + y + z')(x + y' + z)(x' + y + z) \\ &= M_0 M_1 M_2 M_4 \end{aligned}$$

این مثال‌ها نیز دومین خاصیت جبر بول را به نمایش می‌گذارند: هر تابع بول را می‌توان بصورت ضرب ماکسترم‌ها (ضرب به معنی AND است) در آورد. روال تهیه ضرب ماکسترم‌ها مستقیماً از جدول درستی به فرم زیر میسر است. برای هر ترکیبی از متغیرها ماکسترم‌هایی که در تابع 0 تولید می‌کنند را تشکیل دهید، و سپس AND همه ماکسترم‌ها را بدست آورید. توابع بول که بصورت جمع مینترم‌ها یا ضرب ماکسترم بیان شوند را فرم متعارف نامند.

مجموع مینترم‌ها

قبلاً بیان شد که برای هر n متغیر دودویی 2^n مینترم مجزا وجود دارد و هر تابع بولی می‌تواند بصورت مجموعی از مینترم‌ها در آید. مینترم‌هایی که جمع آنها توابع بول را تعریف می‌کنند، آنهایی هستند که 1های تابع را در جدول درستی تشکیل می‌دهند. چون تابع در قبال هر مینترم می‌تواند 0 و یا 1 باشد، و چون 2^n مینترم وجود دارد، می‌توان تعداد توابع ممکن که با n متغیر ایجاد می‌شود را 2^{2^n} دانست. گاهی بهتر است تابع بول را برحسب جمع مینترم‌ها بیان کنیم. اگر در این فرم نبود، می‌توان ابتدا آن را بصورت جمع جملات AND در آورد. آنگاه هر ترم برای یافتن همه متغیرها در آن وارسی می‌شود. اگر یک یا چند متغیر وجود نداشته باشند، می‌توان جمله را در عبارتی مثل $x + x'$، AND نمود، که x یکی از متغیرهای مفقود شده است. مثال زیر مطلب را روشن می‌کند.

مثال ۴-۲

تابع بولی $F = A + B'C$ را بصورت جمع مینترم‌ها در آورید. تابع سه متغیر A، B و C دارد. در اولین جمله A، دو متغیر مفقود است؛ بنابراین:

$$A = A(B + B') = AB + AB'$$

این تابع هنوز هم یک متغیر کسر دارد

$$\begin{aligned} A &= AB(C + C') + AB'(C + C') \\ &= ABC + ABC' + AB'C + AB'C' \end{aligned}$$

جمله دوم $B'C$ یک متغیر کم دارد

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

با ترکیب همه جملات داریم:

$$\begin{aligned} F &= A + B'C \\ &= ABC + ABC' + AB'C + AB'C' + A'B'C \end{aligned}$$

دیده می‌شود که $AB'C$ دوبار تکرار شده است و برحسب تئوری $(x + x = x)$ می‌توان یکی از آنها را حذف کرد. با مرتب نمودن مینترم‌ها به ترتیب صعودی داریم:

$$\begin{aligned} F &= A'B'C + AB'C + AB'C + ABC' + ABC \\ &= m_1 + m_4 + m_5 + m_6 + m_7 \end{aligned}$$

❑

گاهی بهتر است تابع بول را وقتی بصورت جمع مینترم‌هاست به فرم خلاصه زیر نشان دهیم:

$$F(A, B, C) = \Sigma(1, 4, 5, 6, 7)$$

سمبل جمع Σ به معنی OR جملات است؛ اعدادی که به دنبال آن می‌آیند نیز مینترم‌های تابع هستند. حروف داخل پرانتز در جلو F، لیستی از متغیرهای تشکیل دهنده جملات مینترم را نشان می‌دهند. روش دیگری برای تشکیل مینترم‌های تابع بول تهیه مستقیم جدول درستی تابع از عبارت جبری و سپس خواندن مینترم‌ها از جدول درستی است. تابع بول مثال ۴-۲ را در نظر بگیرید:

$$F = A + B'C$$

جدول درستی در جدول ۵-۲ مستقیماً از عبارت جبری با لیست هشت ترکیب زیر متغیرهای A، B و C و اعمال 1 زیر ستون F برای ترکیباتی که در آن $A = 1$ و $BC = 01$ است فراهم شده است. سپس از جدول درستی می‌توان مشاهده کرد که مینترم‌های تابع، جملات 1، 4، 5، 6 و 7 می‌باشند.

ضرب ماکسترم‌ها

هر یک از 2^{2^n} تابع متشکل از n متغیر را می‌توان بصورت ضرب ماکسترم‌ها نیز بیان داشت. برای بیان توابع بول به عنوان ضرب ماکسترم‌ها، ابتدا باید جملات OR را تشکیل دهیم. این کار را می‌توان با

جدول ۵-۲. جدول درستی برای $F = A + B'C$

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

استفاده از قانون توزیع‌پذیری $x + yz = (x + y)(x + z)$ انجام داد. سپس هر متغیر مفقود در هر جمله OR با xx'، OR می‌شود. روش با مثال زیر روشن‌تر خواهد شد.

مثال ۵-۲

تابع بول $F = xy + x'z$ را بصورت ضرب جملات ماکسترم نشان دهید. ابتدا تابع را با استفاده از اصل توزیع‌پذیری به فرم جملات OR در آورید:

$$\begin{aligned} F &= xy + x'z = (xy + x')(xy + z) \\ &= (x + x')(y + x')(x + z)(y + z) \\ &= (x' + y)(x + z)(y + z) \end{aligned}$$

تابع سه متغیر دارد: x، y و z. هر جمله OR فاقد یک متغیر است؛ بنابراین

$$\begin{aligned} x' + y &= x' + y + zz' = (x' + y + z)(x' + y + z') \\ x + z &= x + z + yy' = (x + y + z)(x + y' + z) \\ y + z &= y + z + xx' = (x + y + z)(x' + y + z) \end{aligned}$$

با ترکیب همه جملات و حذف تکراری‌ها، خواهیم داشت:

$$\begin{aligned} F &= (x + y + z)(x + y' + z)(x' + y + z)(x' + y + z') \\ &= M_0M_2M_4M_5 \end{aligned}$$

نمایش ساده‌تر به شکل زیر است:

$$F(x, y, z) = \Pi(0, 2, 4, 5)$$

سمبل ضرب، Π، بیانگر AND ماکسترم‌هاست. اعداد داخل پرانتز شماره ماکسترم‌های تابع‌اند. ❑

تبدیل فرم‌های متعارف به یکدیگر

متمم یک تابع که بصورت مجموع مینترم‌ها نشان داده شده برابر است با مجموع مینترم‌هایی که در تابع اصلی وجود ندارند. دلیل این است که تابع اصلی با آن دسته از مینترم‌ها بیان شده است که تابع را 1

می‌کنند، در صورتی که متمم آن در ازاء مینترم‌هایی 1 می‌شود که تابع را 0 نموده‌اند. به عنوان مثال تابع زیر را در نظر بگیرید:

$$F(A, B, C) = \Sigma(1, 4, 5, 6, 7)$$

متمم این تابع به شکل زیر است:

$$F'(A, B, C) = \Sigma(0, 2, 3) = m_0 + m_2 + m_3$$

اکنون اگر متمم F' را با روش تئوری دمورگان بدست آوریم، F را به فرم متفاوتی خواهیم داشت:

$$F = (m_0 + m_2 + m_3)' = m_0' \cdot m_2' \cdot m_3' = M_0 M_2 M_3 = \Pi(0, 2, 3)$$

آخرین تبدیل در رابطه فوق از تعریف مینترم‌ها و ماکسترم‌ها در جدول (۳-۲) حاصل می‌شود. با توجه به جدول درستی رابطه زیر معتبر است:

$$m_j' = M_j$$

یعنی ماکسترم jام، متمم مینترم jام است و بالعکس.

آخرین مثال تبدیل یک تابع مینترمی به معادل ماکسترمی را نشان می‌دهد. بحث مشابهی نشان می‌دهد که تبدیل ضرب ماکسترم‌ها به جمع مینترم‌ها نیز به طریق فوق است. اکنون یک روال کلی را بیان می‌کنیم. برای تبدیل یک فرم متعارف به فرم متعارف دیگر، سمبل‌های Σ و Π را با هم عوض کنید و شماره‌های مفقود شده را از فرم اصلی تابع، لیست نمایید. برای یافتن جملات مفقود، باید بدانیم که تعداد کل جملات 2^n است، که در آن n تعداد متغیرهای دودویی در تابع می‌باشد.

یک تابع بولی می‌تواند از یک عبارت جبری به کمک جدول درستی و روال تبدیل متعارف به ضربی از ماکسترم‌ها تبدیل شود. به عنوان مثال عبارت بولی زیر را ملاحظه نمایید.

$$F = xy + x'z$$

ابتدا جدول درستی تابع را طبق جدول ۶-۲ بدست می‌آوریم. 1 های زیر ستون F از ترکیب $xy = 11$ یا $xz = 01$ بدست می‌آیند. مینترم‌های تابع در جدول درستی شماره‌های 1 ، 3 ، 6 و 7 می‌باشند. تابع برحسب مجموع مینترم‌ها چنین است:

$$F(x, y, z) = \Sigma(1, 3, 6, 7)$$

جدول ۶-۲ . جدول درستی برای $F = xy + x'z$

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

چون جمعاً در یک تابع از سه متغیر هشت مینترم یا ماکسترم وجود دارد، جملات مفقود عبارتند از 0، 2، 4 و 5. تابعی که برحسب ضرب ماکسترم‌ها بیان شود برابر زیر است:

$$F(x, y, z) = \Pi(0, 2, 4, 5)$$

و این همان جوابی است که در مثال ۵-۲ بدست آمد.

فرم‌های استاندارد

دو نمایش متعارف جبر بول، فرم‌های اصلی تابع‌اند که هر کس می‌تواند با توجه به جدول درستی به آنها دسترسی پیدا کند. این فرم‌ها معمولاً دارای حداقل متغیرها نیستند زیرا هر مینترم یا ماکسترم باید بنا به تعریف دارای تمام متغیرها اعم از متمم یا غیرمتمم باشد.

راه دیگری برای بیان تابع بول، فرم استاندارد است. در این فرم، جمله‌هایی که تابع را تشکیل می‌دهند ممکن است یک، دو یا هر تعدادی از متغیرها را دارا باشند. دو نوع فرم استاندارد وجود دارد. یکی جمع حاصلضرب‌ها و دیگری ضرب حاصل جمع‌ها.

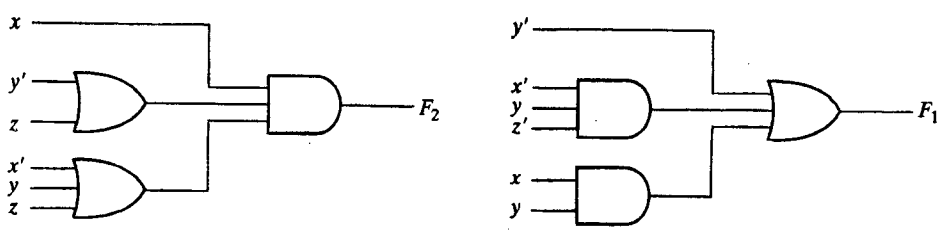
جمع حاصلضرب‌ها، یک عبارت بولی است شامل جملات AND که آنها را جملات ضرب می‌گوییم و هر یک دارای یک یا چند لیترال است. علامت جمع به معنی OR این جملات است. مثالی از یک تابع بصورت جمع حاصلضرب‌ها را در زیر ملاحظه نمایید.

$$F_1 = y' + xy + x'yz'$$

این عبارت سه جمله، با یک، دو و سه لیترال دارد. جمع آنها اثر OR را داراست.

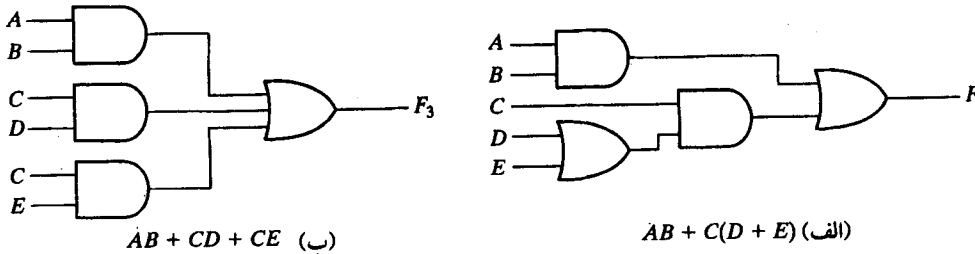
نمودار منطقی جمع حاصلضرب‌ها متشکل از گروهی گیت AND است که بدنبال آن یک گیت OR می‌آید. الگوی این آرایش در شکل ۳-۲ (الف) آمده است. هر جمله ضرب نیاز به یک گیت AND دارد. این نکته در یک ورودی تک لیترالی مستثنی است. جمع منطقی با یک گیت OR صورت می‌گیرد که ورودی‌هایش خروجی گیت‌های AND و نیز تک ورودی مذکور است. ضمناً فرض بر این است که متمم متغیرهای ورودی مستقیماً موجودند بنابراین وارونگر در نمودار لحاظ نشده است. این آرایش را پیاده‌سازی دو سطحی یا دو طبقه می‌گویند.

ضرب حاصل جمع‌ها، یک عبارت بولی حاوی جملات OR است که به آن جملات جمع می‌گویند.



(ب) ضرب حاصل جمع‌ها (الف) جمع حاصلضرب‌ها

شکل ۳-۲. پیاده‌سازی دو سطحی



شکل ۴-۲ . پیاده‌سازی سه و دو سطحی

هر جمله می‌تواند به هر تعداد لیترال داشته باشد. ضرب به معنی AND این جملات است. مثالی از یک تابع بصورت ضرب حاصل جمع‌ها چنین است:

$$F_2 = x(y' + z)(x' + y + z')$$

این تابع دارای سه جمله جمع، با یک، دو و سه لیترال است. ضرب نیز یک عملگر AND می‌باشد. استفاده از لغات ضرب و جمع از شباهت عمل AND با ضرب حسابی و شباهت عمل OR با جمع حسابی مشتق شده است. ساختار گیتی ضرب حاصل جمع متشکل از گروهی گیت OR برای جملات جمع (به جز برای تک لیترال) و به دنبال آن یک گیت AND می‌باشد. این نکته در شکل ۳-۲ (ب) دیده می‌شود، این نوع استاندارد عبارت به یک ساختار دو سطحی (یا دو طبقه) از گیت‌ها منجر می‌گردد.

یک تابع بول ممکن است بصورت غیراستاندارد نیز بیان شود. مثلاً تابع

$$F_3 = AB + C(D + E)$$

نه جمع حاصلضرب و نه ضرب حاصل جمع است. پیاده‌سازی این عبارت در شکل ۴-۲ (الف) دیده می‌شود. این مدار به دو گیت AND و دو گیت OR نیاز دارد. در این مدار سه سطح گیت وجود دارد. می‌توان در آن با استفاده از اصل توزیع‌پذیری پرانتز را حذف و آن را به فرم استاندارد در آورد.

$$F_3 = AB + C(D + E) = AB + CD + CE$$

عبارت جمع حاصلضرب در شکل ۴-۲ (ب) پیاده شده است. بطور کلی، یک پیاده‌سازی دو سطحی ترجیح داده می‌شود زیرا به هنگام انتشار ورودی‌ها به سمت خروجی‌ها حداقل مقدار تأخیر را در گیت‌ها تولید می‌کند.

۶-۲ دیگر اعمال منطقی

وقتی که عملگرهای AND و OR بین دو متغیر x و y قرار می‌گیرند، به ترتیب دو تابع بولی $x.y$ و $x+y$ را تشکیل می‌دهند. قبلاً بیان شد که برای n متغیر 2^{2^n} تابع دودویی وجود دارد. برای دو متغیر، $n=2$ و تعداد توابع بولی ممکن 16 است. بنابراین توابع AND و OR تنها دو تابع از 16 تابع ممکن با دو متغیر دودویی هستند. پیشنهاد می‌گردد 14 تابع باقیمانده مشخص شده و راجع به خصوصیت آنها تحقیق گردد.

جدول درستی 16 تابع که با دو متغیر دودویی x و y تشکیل گردیده در جدول ۷-۲ لیست شده

جدول ۷-۲ . جدول درستی برای 16 تابع از دو متغیر دودویی

x	y	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

است. هر یک از 16 ستون F_0 و F_{15} ، جدول درستی یک تابع ممکن برای دو متغیر x و y را نشان می‌دهد. توجه داشته باشید که توابع از 16 ترکیب ممکن تخصیص یافته به F معین می‌گردند. 16 تابع را می‌توان با توابع بول نشان داد. این توابع در ستون اول جدول ۸-۲ دیده می‌شوند. عبارات بولی از نظر تعداد لیترال حداقل شده‌اند.

گرچه هر تابع را می‌توان برحسب عملگرهای دودویی AND ، OR و NOT بیان کرد، اما دلیلی وجود ندارد که کسی عملگر خاصی را برای بیان سایر توابع تعیین ننماید. چنین عملگرهایی در ستون دوم جدول ۸-۲ لیست شده‌اند. با این وجود همه سمبل‌های جدید که در جدول نشان داده شده‌اند، بجز OR انحصاری (⊕) کاربرد چندانی به وسیله طراحان ندارند.

به دنبال هر یک از توابع در جدول ۸-۲، نام و توضیحی که تابع را به نحوی تشریح می‌کند، آورده شده است. 16 تابع لیست شده فوق به سه گروه زیر تقسیم می‌شوند.

۱- دو تابع که ثابت‌های 0 و 1 را تولید می‌کنند (Identity , Null)

۲- چهار تابع یکانی از نوع متمم و انتقال (Transfer , Complement)

جدول ۸-۲ . عبارات بولی 16 تابع حاصل از دو متغیر

توابع بول	سمبل عملگر	نام	توضیحات
$F_0 = 0$		Null	Binary constant 0
$F_1 = xy$	$x \cdot y$	AND	x and y
$F_2 = xy'$	x/y	Inhibition	x, but not y
$F_3 = x$		Transfer	x
$F_4 = x'y$	y/x	Inhibition	y, but not x
$F_5 = y$		Transfer	y
$F_6 = xy' + x'y$	$x \oplus y$	Exclusive-OR	x or y, but not both
$F_7 = x + y$	$x + y$	OR	x or y
$F_8 = (x + y)'$	$x \downarrow y$	NOR	Not-OR
$F_9 = xy + x'y'$	$(x \oplus y)'$	Equivalence	x equals y
$F_{10} = y'$	y'	Complement	Not y
$F_{11} = x + y'$	$x \subset y$	Implication	If y, then x
$F_{12} = x'$	x'	Complement	Not x
$F_{13} = x' + y$	$x \supset y$	Implication	If x, then y
$F_{14} = (xy)'$	$x \uparrow y$	NAND	Not-AND
$F_{15} = 1$		Identity	Binary constant 1

۳- ده تابع باقیمانده شامل هشت عمل مختلف AND ، OR ، NAND ، NOR ، XOR (یا OR انحصاری)، XNOR (یا هم ارزی) Inhibition (نهی) و Implication (استلزام) می‌باشند.

ثابت‌ها برای توابع دودویی فقط می‌توانند 0 یا 1 باشند. تابع متمم، متمم هر متغیر دودویی را تولید می‌نماید. تابعی که برابر یک متغیر ورودی است را Transfer یا Buffer (انتقال) می‌نامند، زیرا متغیر x یا y از طریق یک گیت بدون تغییر مقدار عبور کرده است. از هشت عملگر دودویی، دوتای آنها (نهی و استلزام) به وسیله طراحان مدارات منطقی بکار می‌روند، ولی به ندرت در منطق کامپیوتر از آنها استفاده می‌شود. عملگرهای AND و OR قبلاً در جبر بول ذکر شدند. چهار تابع دیگر بطور گسترده در طراحی دستگاه‌های دیجیتال مورد استفاده‌اند.

تابع NOR متمم OR بوده و نام آن از not − OR اخذ شده است. بطور مشابه NAND ، متمم AND است و از not − AND مشتق می‌شود. OR انحصاری یا XOR مشابه با OR است ولی حالتی که در آن هر دو متغیر x و y متفقاً برابر 1 باشند، را شامل نمی‌شود. تابع XNOR یا هم‌ارزی تابعی است که هنگام مساوی بودن دو متغیر برابر 1 می‌شود، یعنی وقتی هر دو 0 یا هر دو 1 باشند. توابع XOR و XNOR متمم یکدیگرند و این خاصیت بسادگی با ملاحظه جدول ۷-۲ قابل تشخیص است. جدول درستی برای OR عبارت است از F_6 و برای XNOR نیز F_9 است. این دو تابع متمم یکدیگرند. به این دلیل تابع هم‌ارزی را NOR انحصاری هم می‌گویند و با XNOR نشان می‌دهند.

جبر بول طبق تعریف بخش ۲-۲ دو عملگر دودویی با نام‌های AND و OR و یک عملگر یکانی با نام NOT (متمم) دارد. ما با توجه به تعاریف، برخی از خواص آنها را استنتاج نمودیم و در این بخش تعدادی از عملگرهای دودویی دیگر را برحسب آنها معرفی کردیم. این روال منحصر به فرد نیست. به عنوان مثال می‌توانستیم ابتدا NOR(↓) را تعریف کرده و سپس AND و OR و NOT را برحسب آنها تعریف کنیم. به هر حال دلایل مکفی برای روش منتخب وجود دارد و در واقع مفاهیم AND ، OR و NOT بیشتر بین مردم مصطلح بوده و بر تفکرات حاکم هستند. علاوه بر آن اصول هانتینگتون منعکس کننده طبیعت دوگانی این جبر است و این خود بر خاصیت تقارن (+) و (.) نسبت به یکدیگر دلالت دارد.

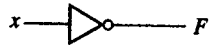
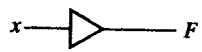
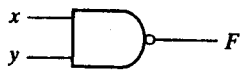
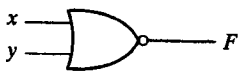
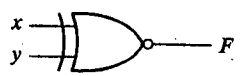
۷-۲ گیت‌های منطقی دیجیتال

چون توابع بول برحسب عملگرهای AND ، OR و NOT بیان شده‌اند، پیاده کردن آنها با استفاده از اینگونه گیت‌ها ساده‌تر خواهد بود. امکان ساخت گیت‌ها برای دیگر اعمال منطقی در عمل مورد توجه است. فاکتورهایی که باید به هنگام ساخت آنها در نظر گرفته شوند عبارتند از:

(۱) امکان سنجی و اقتصادی بودن روش ساخت به هنگام استفاده از قطعات فیزیکی. (۲) امکان گسترش ورودی گیت‌ها به بیش از دو. (۳) در نظر گرفتن خواص اصلی عملگرهای دودویی مثل جابجایی و شرکت‌پذیری. (۴) توانایی گیت در پیاده‌سازی توابع به تنهایی یا همراه با سایر گیت‌ها از شانزده تابع معرفی شده. در جدول (۸-۲) دو تابع برابر با مقدار ثابت و چهارتای دیگر دوبار تکرار شده‌اند. بنابراین تنها ده تابع برای تهیه گیت‌های منطقی کاندید هستند. دو تابع نهی و استلزام دارای

خاصیت جابجایی یا شرکت‌پذیری نیستند و لذا به عنوان گیت‌های منطقی استاندارد مورد استفاده نمی‌باشند. هشت تابع دیگر یعنی: Buffer ، NOT ، AND ، OR ، NAND ، NOR ، XOR و XNOR به عنوان گیت‌های استاندارد در طراحی سیستم‌های دیجیتال بکار می‌روند.

سمبل‌های گرافیکی و جدول درستی هشت گیت فوق در شکل ۵-۲ نشان داده شده‌اند. هر گیت

نام	سمبل گرافیکی	تابع جبری	جدول درستی
AND		$F = xy$	$x\ y \mid F$ $0\ 0 \mid 0$ $0\ 1 \mid 0$ $1\ 0 \mid 0$ $1\ 1 \mid 1$
OR		$F = x + y$	$x\ y \mid F$ $0\ 0 \mid 0$ $0\ 1 \mid 1$ $1\ 0 \mid 1$ $1\ 1 \mid 1$
Inverter		$F = x'$	$x \mid F$ $0 \mid 1$ $1 \mid 0$
Buffer		$F = x$	$x \mid F$ $0 \mid 0$ $1 \mid 1$
NAND		$F = (xy)'$	$x\ y \mid F$ $0\ 0 \mid 1$ $0\ 1 \mid 1$ $1\ 0 \mid 1$ $1\ 1 \mid 0$
NOR		$F = (x + y)'$	$x\ y \mid F$ $0\ 0 \mid 1$ $0\ 1 \mid 0$ $1\ 0 \mid 0$ $1\ 1 \mid 0$
Exclusive-OR (XOR)		$F = xy' + x'y$ $= x \oplus y$	$x\ y \mid F$ $0\ 0 \mid 0$ $0\ 1 \mid 1$ $1\ 0 \mid 1$ $1\ 1 \mid 0$
Exclusive-NOR or equivalence		$F = xy + x'y'$ $= (x \oplus y)'$	$x\ y \mid F$ $0\ 0 \mid 1$ $0\ 1 \mid 0$ $1\ 0 \mid 0$ $1\ 1 \mid 1$

شکل ۵-۲ . گیت‌های منطقی دیجیتال نام، سمبل گرافیکی، تابع جبری، جدول درستی

دارای دو متغیر ورودی دودویی x و y و یک متغیر خروجی دودویی F می‌باشد. مدارهای AND، OR و NOT در شکل ۶-۱ معرفی شده بودند. مدار NOT یا وارونگر وضعیت منطقی یک متغیر دودویی را معکوس می‌نماید و در واقع متمم متغیر را تولید می‌کند. دایره کوچک در خروجی سمبل گرافیکی یک وارونگر (که به آن حباب می‌گویند) بیانگر متمم شدن است. سمبل مثبت بـه تـنهایی عـلامت بـافر می‌باشد. یک بافر عمل انتقال را انجام می‌دهد، ولی یک عمل منطقی تولید نمی‌کند زیرا مقدار دودویی خروجی برابر مقدا رورودی دودویی است. این مدار صرفاً در تقویت توان سیگنال‌ها استفاده شـده و معادل با دو مدار متوالی وارونگر (معکوس‌گر) است.

تابع NAND متمم AND است و همانطور که از سمبل گرافیکی آن مشخص است از یک سمبل AND و یک حباب تشکیل شده است. تابع NOR هم متمم OR است و با یک سمبل OR و به دنبال آن یک حباب نمایش داده می‌شود. گیت‌های NAND و NOR بطور گسترده‌ای به عـنوان گـیت‌های استاندارد مورد استفاده قرار گرفته و بیشتر از OR و AND مورد توجه‌اند. ایـن بـدان عـلت است کـه گیت‌های NAND و NOR به سادگی به وسیله مدارات ترانزیستوری قابل تولید بوده و می‌توان بـه راحتی توابع بول را با آنها پیاده‌سازی کرد.

گیت XOR دارای سمبل مشابهی با OR است، بجز اینکه یک خط منحنی در سمت ورودی‌اش کشیده شده است. گیت XNOR متمم XOR است و لذا حباب کوچکی در خروجی آن وجود دارد.

گسترش ورودی گیت‌ها

گیت‌هایی که در شکل ۵-۲ نشان داده شدند، بجز برای وارونگر و انتقال، قابل گسترش به بیش از دو ورودی می‌باشند. اگر عمل دودویی یک گیت جابجا و شرکت‌پذیر باشد. می‌توان ورودی‌هـای آن را گسترش داد. اعمال AND و OR که در جبر بول تعریف شده‌اند این خاصیت را از خود بـه نـمایش گذاشته‌اند. برای تابع OR داریم:

$$x + y = y + x \qquad \text{(جابجایی)}$$

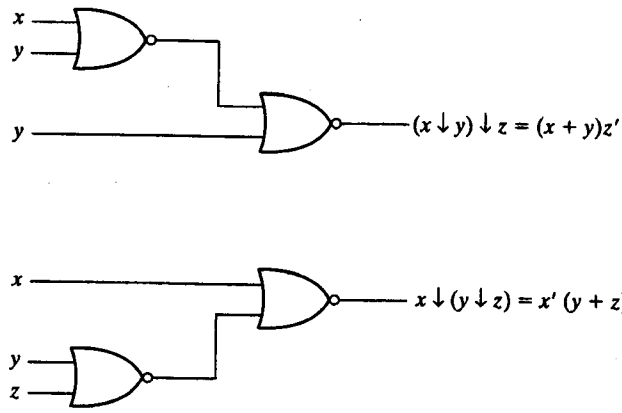
و

$$(x + y) + z = x + (y + z) = x + y + z \qquad \text{(شرکت‌پذیر)}$$

این روابط بیانگر تعویض‌پذیری ورودی‌های گیت و قابل گسترش بودن متغیرهای ورودی به بیش از دو در تابع OR است.

توابع NAND و NOR جابجاپذیرند و ورودی آنها می‌تواند به بیش از دو افزایش یابد، مشروط بر این که در تعریف تابع مختصر تغییری صورت گیرد. مشکل این است که NAND و NOR شرکت‌پذیر نیستند [یعنی $(x \downarrow y) \downarrow z \neq x(y \downarrow z)$]. این نکته در شکل ۶-۲ و معادلات زیر بنمایش گذاشته شده‌اند.

$$(x \downarrow y) \downarrow z = [(x + y)' + z]' = (x + y)z' = xz' + yz'$$
$$x \downarrow (y \downarrow z) = [x + (y + z)']' = x'(y + z) = x'y + x'z$$



شکل ۶-۲ . نمایش شرکت‌ناپذیری عملگر NOR ($x \downarrow y) \downarrow z \neq x \downarrow (y \downarrow z$)

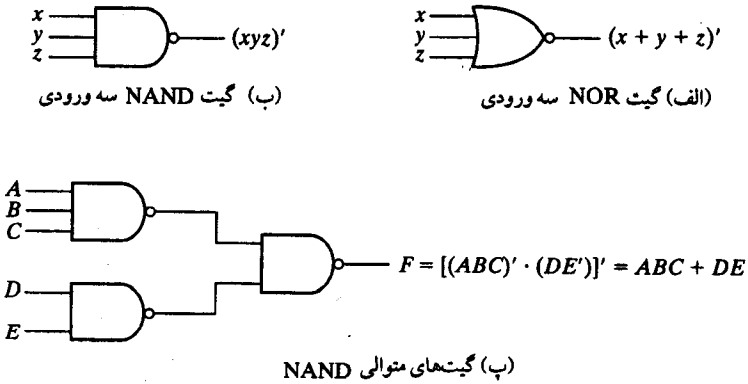
برای غلبه بر این مشکل، گیت NOR (یا NAND) چند ورودی را به عنوان متمم OR (یا AND) آن تعریف می‌کنیم. بنابراین:

$$x \downarrow y \downarrow z = (x + y + z)'$$
$$x \uparrow y \uparrow z = (xyz)'$$

سمبل‌های گرافیکی گیت‌های سه ورودی در شکل ۷-۲ نشان داده شده‌اند. در نوشتن متوالی اعمال NOR و NAND باید پرانتزها به فرم صحیحی انتخاب شوند تا بیانگر ترتیب صحیح گیت‌ها باشند. برای درک این مطلب مدار شکل ۷-۲ (پ) را ملاحظه نمایید. برای این مدار تابع بول باید به فرم زیر نوشته شود:

$$F = [(ABC)'(DE)']' = ABC + DE$$

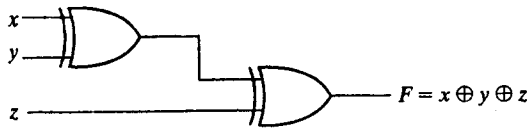
دومین عبارت از تئوری دمورگان نتیجه می‌شود. این رابطه همچنین بیان می‌دارد که جمع حاصلضرب‌ها قابل پیاده شدن با گیت‌های NAND می‌باشد. بحث بیشتر در مورد گیت‌های NAND و NOR در بخش ۶-۳ آورده شده است.



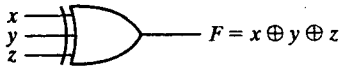
شکل ۷-۲ . گیت‌های NAND و NOR متوالی با چند ورودی

x	y	z	F
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

(پ) جدول درستی



(الف) با گیت‌های دو ورودی



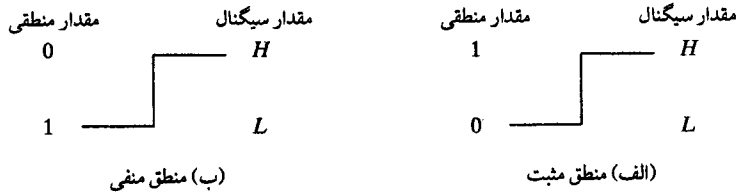
(ب) یک گیت سه ورودی

شکل ۸-۲. گیت XOR سه ورودی

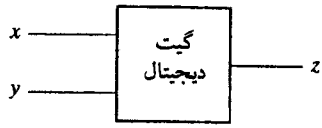
XOR و XNOR هر دو خواص جابجایی و شرکت‌پذیری را دارند و ورودی‌هایشان قابل توسعه به بیش از دو می‌باشد. با این وجود گیت‌های XOR چند ورودی از نقطه نظر سخت‌افزاری متداول نیستند. در واقع حتی فرم دو ورودی آن نیز معمولاً از سایر گیت‌ها ساخته می‌شود. علاوه بر این، تعریف این توابع باید به هنگام گسترش ورودی‌ها تصحیح گردد. تابع XOR یک تابع فرد است یعنی هرگاه ورودی‌ها تعداد فردی 1 داشته باشند، این تابع (خروجی) برابر 1 خواهد بود. ساختمان یک گیت XOR با سه ورودی در شکل ۸-۲ دیده می‌شود. این مدار معمولاً با گیت‌های دو ورودی تهیه می‌شود، شکل (الف). بصورت گرافیکی، آن را می‌توان با یک گیت سه ورودی نشان داد، شکل (ب). جدول درستی (پ) آشکارا مشخص می‌نماید که خروجی F برابر 1 است به شرطی که فقط یکی از ورودی‌ها و یا هر سه ورودی برابر 1، باشند؛ یعنی وقتی تعداد کل 1ها در متغیرهای ورودی فرد است، تابع 1 است. بحث بیشتری در مورد XOR به بخش ۸-۳ موکول شده است.

منطق مثبت و منفی

سیگنال دودویی در ورودی‌ها یا خروجی هر گیت، بجز در حالت گذرا، یکی از دو مقدار را دارد. یک مقدار سیگنال، منطق 1 و دیگری منطق 0 را نمایش می‌دهد. چون دو مقدار سیگنال متعلق به دو ارزش منطقی است، لذا دو انتساب متفاوت برای دو ارزش منطقی می‌توان اختیار کرد، شکل ۹-۲. سطح سیگنال بالاتر با H و سطح سیگنال پایین‌تر با L مشخص شده است. اگر سطح بالا، H، برای منطق 1 بکار رود یک سیستم منطق مثبت تعریف شده است. انتخاب L برای منطق 1 سیستم منطقی منفی را معرفی می‌نماید. کلمات مثبت یا منفی گاهی گمراه کننده هستند زیرا هر دو سطح سیگنال ممکن است



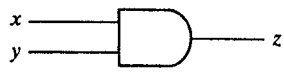
شکل ۹-۲. تخصیص سیگنال و قطبیت منطق



(ب) نمودار بلوکی گیت

x	y	F
L	L	L
L	H	L
H	L	L
H	H	H

(الف) جدول درستی با
L و H



(ت) کمیت AND با منطق مثبت

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

(پ) جدول درستی برای منطق مثبت



(ج) گیت OR با منطق منفی

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

(ث) جدول درستی برای منطقی منفی

شکل ۱۰-۲. نمایش منطق مثبت و منفی

مثبت یا هر دو منفی باشند. در واقع، این قطب‌های سیگنال نیستند که بیانگر نوع منطق می‌باشند، بلکه انتخاب مقادیر منطقی برحسب سطوح نسبی سیگنال‌ها نسبت به هم، نوع منطق را مشخص می‌کنند.

گیت‌های دیجیتال سخت‌افزاری برحسب مقادیر سیگنال H و L تعریف می‌شوند. از این پس انتخاب منطق مثبت و منفی به عهده کاربر است. به عنوان مثال گیت الکترونیک شکل ۱۰-۲ (ب) را در نظر بگیرید. جدول درستی برای این گیت در شکل ۱۰-۲ (الف) لیست شده است. این جدول رفتار فیزیکی گیت را وقتی H برابر 3V و L برابر 0 ولت است نشان می‌دهد. جدول درستی شکل ۱۰-۲ (پ) منطق مثبت را فرض می‌کند که در آن H = 1 و L = 0 است. این جدول درستی همانند جدول عمل AND است. نمودار گرافیکی برای گیت AND با منطق مثبت درشکل ۱۰-۲ (ت) دیده می‌شود.

اکنون تخصیص منطق منفی را برای همان گیت فیزیکی با L = 1 و H = 0 در نظر بگیرید. نتیجه جدول درستی شکل ۱۰-۲ (ث) خواهد بود. گرچه داده‌ها معکوس شده‌اند ولی جدول عمل OR را نشان می‌دهد. سمبل گرافیکی گیت OR با منطق منفی در شکل ۱۰-۲ (ج) دیده می‌شود. مثلث‌های کوچک در ورودی‌ها و خروجی نشانگر قطبیت هستند. وجود این علائم قطبیت همراه با مشخصات پایانه بیانگر فرض منطق منفی برای سیگنال است. بنابراین گیت فیزیکی فوق می‌تواند یک گیت AND با منطق مثبت و نیز یک گیت OR با منطق منفی باشد.

تبدیل منطق مثبت به منفی و بالعکس، عملی است که طی آن در ورودی و خروجی یک گیت 0ها به 1 و 1ها به 0 تبدیل می‌شوند. چون این عمل دوگان تابع را تولید می‌کند، تعویض پایانه‌ها از یک قطبیت به قطبیت دیگر نتیجه‌اش همان یافتن دوگان تابع است. نتیجه این تبدیل این است که همه عملگرهای AND به OR و بالعکس تبدیل شوند. به علاوه نباید از ذکر مثلث در سمبل‌های گرافیکی که بیانگر قطبیت است در منطق منفی، فراموش کرد. در این کتاب از گیت‌ها با منطق منفی استفاده نمی‌کنیم و فرض خواهیم که همه گیت‌ها با منطق مثبت کار کنند.

۸-۲ مدارهای مجتمع

یک مدار مجتمع (IC) یک کریستال نیمه هادی از جنس سیلیکان است که به آن تراشه می‌گویند و حاوی اجزاء الکترونیکی در ساخت گیت‌های دیجیتال می‌باشد. انواع گیت‌ها در داخل تراشه به هم وصل می‌شوند تا مدار مورد نیاز ایجاد گردد. تراشه روی یک محفظه سرامیک یا پلاستیک نصب شده و اتصالات به پایه‌های بیرون برای ایجاد مدار مجتمع، متصل می‌گردند. تعداد پایه‌ها ممکن است از 14 در یک بسته کوچک IC تا چند هزار در یک بسته بزرگ تغییر کند. در روی هر IC یک شماره برای شناسایی چاپ می‌شود. سازندگان معمولاً کتابچه‌ها، کاتالوگ‌ها و سایت‌های اینترنتی حاوی توصیف‌ها و اطلاعات IC هایی که ساخته‌اند را در اختیار می‌گذارند.

سطوح مجتمع‌سازی

IC های دیجیتال اغلب براساس پیچیدگی مدار درونی‌شان که به تعداد گیت‌های منطقی مرتبط است دسته‌بندی می‌شوند. تفکیک تراشه‌هایی که تنها چند یا چند صد و یا چندین هزار گیت دارند با ارجاع به بسته و دسته‌بندی آنها به وسایل مجتمع با فشردگی کم، متوسط، زیاد و خیلی زیاد صورت می‌گیرد.

مدارهای مجتمع با فشردگی کم *(SSI)* دارای چند گیت مستقل در بسته‌اند. ورودی‌ها و خروجی‌های گیت‌ها مستقیماً به پایه‌های بسته متصل می‌شوند. تعداد گیت‌ها معمولاً کمتر از 10 بوده و به وسیله پایه‌های موجود در IC محدود می‌گردند.

مدارهای مجتمع با فشردگی متوسط *(MSI)* دارای فشردگی بین 10 تا 1000 گیت در یک بسته دارند. این دسته از مدارات معمولاً اعمال دیجیتال خاصی را اجرا می‌کنند. توابع دیجیتال MSI با عناوین دیکدرها، جمع کننده‌ها و مولتی پلکسرها در فصل ۴ و ثبات‌ها و شمارنده‌ها در فصل ۶ مطرح شده‌اند.

مدارهای مجتمع با فشردگی زیاد *(LSI)* حاوی هزاران گیت در یک بسته می‌باشند. این دسته از مدارات، پردازنده‌ها، حافظه‌ها و مدارات منطقی برنامه‌پذیر را شامل می‌شوند. بعضی از اجزاء LSI در فصل ۸ معرفی شده‌اند.

مدارهای مجتمع با فشردگی خیلی زیاد *(VLSI)* صدها هزار گیت در یک بسته دارند. از جمله مثال‌ها می‌توان از آرایه‌های حافظه، تراشه میکرو کامپیوترهای پیچیده نام برد. به دلیل کوچکی و ارزانی، وسایل VLSI تکنولوژی طراحی سیستم کامپیوتر را متحول ساخته و به طراح قابلیت ساخت وسایلی را می‌دهد که قبلاً اقتصادی نبودند.

خانواده‌های منطقی دیجیتال

مدارهای مجتمع دیجیتال نه تنها براساس پیچیدگی و عمل منطقی‌شان بلکه براساس تکنولوژی مدار خاصی که به آن تعلق دارند نیز دسته‌بندی می‌گردند. تکنولوژی مدار به نام خانواده مدار منطقی خوانده می‌شود. هر خانواده منطقی دارای مدار الکترونیک مبنای خاص خود بوده و سایر توابع و مدارات پیچیده دیجیتال با استفاده از آنها ساخته می‌شوند. مدار مبنا در هر خانواده، گیت NAND، NOR یا NOT است. قطعات الکترونیک بکار رفته در ساخت مدار مبنا معمولاً برای نام‌گذاری تکنولوژی مورد استفاده قرار می‌گیرد. به لحاظ تجاری انواع متفاوتی از خانواده‌های منطقی مدارات مجتمع معرفی شده‌اند. انواع رایج آنها در زیر لیست شده‌اند.

transistor-transistor logic	TTL
emitter-coupled logic	ECL
metal-oxide semiconductor	MOS
complementary metal-oxide semiconductor	CMOS

TTL مدت مدیدی است که مورد استفاده بوده و به عنوان یک گیت استاندارد شناخته شده است. ECL در سیستم‌هایی که به سرعت بالا نیاز دارد ارجحیت دارد. MOS در مدارهایی که نیاز به چگالی قطعه بالایی دارند مورد استفاده است و CMOS در مواقعی که توان مصرفی باید کم باشد مورد توجه می‌باشد. نظر به این که توان مصرفی کم در طراحی VLSI از اصول است، CMOS تبدیل به یک خانواده منطقی غالب شده است در حالی که از کاربرد خانواده‌های TTL و ECL به تدریج کاسته می‌شود. تحلیل مدار گیت دیجیتال مبنای الکترونیکی در هر خانواده منطقی در فصل ۱۰ ارائه شده است.

مشخصه‌های خانواده‌های منطقی معمولاً با تحلیل مدار گیت مبنا در هر خانواده مقایسه می‌شوند. مهمترین پارامترهای مورد ارزیابی و مقایسه در بخش ۲-۱۰ بحث شده‌اند. در اینجا نیز به خاطر ارجاع مختصراً تعریف می‌گردند.

گنجایش خروجی نشان دهنده تعداد بارهای استانداردی است که خروجی یک نمونه گیت بدون تخریب عملکردش، بتواند راه بیندازد. یک بار استاندارد عبارت است از مقدار جریانی که برای ورودی گیت مشابه دیگر از همان خانواده لازم می‌باشد.

گنجایش ورودی تعداد ورودی‌های موجود در یک گیت است.

توان مصرفی توان تلف شده‌ای است که باید به وسیله منبع تغذیه برای گیت فراهم شود.

تأخیر انتشار عبارت است از متوسط زمان تأخیر در انتقال سیگنال از ورودی به خروجی.

حد پارازیت حداکثر ولتاژ بیرونی است که به ورودی سیگنال اضافه می‌شود ولی موجب تغییر ناخواسته در خروجی مدار نگردد.

طراحی با کمک کامپیوتر (CAD)

طراحی سیستم‌های دیجیتال با مدارهای دیجیتال حاوی میلیون‌ها ترانزیستور کار دشواری است.

سیستم‌هایی با این پیچیدگی را نمی‌توان بدون کمک ابزارهای طراحی کامپیوتری (CAD) ، ایجاد و از صحت عمل آنها مطمئن شد. ابزارهای CAD شامل برنامه‌های نرم‌افزاری است که نمایش‌های کامپیوتری را پشتیبانی کرده و در ایجاد سخت‌افزار دیجیتال به خودکارسازی کمک می‌کنند. خودکارسازی طراحی الکترونیک همه فازهای طراحی مدارهای مجتمع را شامل می‌شود. روالی از یک نمونه طراحی برای ساخت مدارهای VLSI شامل رشته‌ای از مراحل است که از وارده‌های طراحی آغاز و با تولید یک بانک اطلاعاتی حاوی ماسک‌های ساخت IC پایان می‌پذیرد. در ایجاد فیزیکی یک مدار دیجیتال در یک سیلیکان روش‌های مختلفی وجود دارد. طرح می‌تواند، یک مدار مجتمع با کاربرد خاص (ASIC) ، یک آرایه گیتی برنامه‌پذیر موردی (FPGA) ، یک وسیله منطقی برنامه‌پذیر (PLD) یا یک IC سفارشی باشد. با هر یک از این وسایل یک مجموعه ابزار CAD همراه است و نرم‌افزار لازم برای خلق سخت‌افزار را فراهم می‌سازد.

بعضی از سیستم‌های CAD حاوی یک برنامه ادیتور یا ویرایشگر است که نمودار تصویری را روی صفحه نمایش کامپیوتر خلق و اصلاح می‌کند. این فرآیند را ضبط تصویری یا وارده تصویری نامند. با کمک منوها، فرامین صفحه کلید و ماوس یک ادیتور تصویر می‌تواند نمودارهای مدارهای دیجیتال را روی صفحه کامپیوتر ترسیم کند. قطعات از لیستی در کتابخانه درونی، قابل استقرار در روی صفحه کامپیوتر بوده و سپس با خطوطی که سیم‌ها را نمایش می‌دهند به هم وصل می‌شوند. نرم‌افزار وارده تصویر یک بانک اطلاعاتی حاوی اطلاعات تولید شده با شماتیک را خلق و مدیریت می‌کند. گیت‌های ساده و بلوک‌های عملیاتی دارای مدل‌هایی هستند که رفتار و زمانبندی مدار مورد تست را ممکن می‌سازند. این کار با اعمال ورودی‌ها به مدار و استفاده از شبیه‌ساز منطقی برای تعیین خروجی‌ها صورت می‌گیرد.

یک ابزار ساخت مهم در طراحی سیستم‌های دیجیتال استفاده از زبان طراحی سخت‌افزار (HDL) است. HDL یک زبان برنامه‌نویسی است ولی خصوصاً برای توصیف سخت‌افزار دیجیتال آماده شده است. این زبان نمودارهای منطقی و دیگر اطلاعات دیجیتال را به فرم متن نشان می‌دهد. از آن برای شبیه‌سازی و تست و تصدیق عملیاتی قبل از ساخت استفاده می‌شود. کاربرد مهمی از آن نرم‌افزار طراحی منطقی آن است، که طراحی سیستم‌های دیجیتال را خودکار می‌سازد. HDL در سال‌های اخیر اهمیت به سزایی یافته است و بهترین روش موجود در طراحی سیستم‌های دیجیتال پیشرفته است. HDL در بخش ۹-۳ معرفی شده است و به دلیل اهمیت آن، ما توصیف HDL مدارهای دیجیتال، قطعات و روال طراحی را در سرتاسر این کتاب لحاظ کرده‌ایم.

۲-۱ به کمک جداول درستی اعتبار موارد زیر را تحقیق کنید:

(الف) تئوری دمورگان برای سه متغیر: $(xyz)' = x' + y' + z'$ و $(x + y + z)' = x'y'z'$

(ب) اصل توزیع‌پذیری: $x + yz = (x + y)(x + z)$

۲-۲ عبارات بولی زیر را با تعداد حداقل لیترال ساده کنید:

(الف) $xy + xy'$ (ب) $(x + y)(x + y')$

(پ) $xyz + x'y + xyz'$ (ت) $(A + B)'(A' + B')'$

۲-۳ عبارات بولی زیر را با تعداد حداقل لیترال ساده نمایید:

(الف) $ABC + A'B + ABC'$ (ب) $x'yz + xz$

(پ) $(x + y)'(x' + y')$ (ت) $xy + x(wz + wz')$

(ث) $(BC' + A'D)(AB' + CD')$

۲-۴ عبارات بولی زیر را به تعداد لیترال‌های ذکر شده ساده کنید.

(الف) $A'C' + ABC + AC'$ سه لیترال

(ب) $(x'y' + z)' + z + xy + wz$ سه لیترال

(پ) $A'B(D' + C'D) + B(A + A'CD)$ یک لیترال

(ت) $(A' + C)(A' + C')(A + B + C'D)$ چهار لیترال

۲-۵ متمم $F = x + yz$ را پیدا کنید و سپس نشان دهید که $FF' = 0$ و $F + F' = 1$ است.

۲-۶ متمم عبارات زیر را پیدا کنید:

(الف) $xy' + x'y$ (ب) $(AB' + C)D' + E$

(پ) $(x + y' + z)(x' + z')(x + y)$

۲-۷ با فرض توابع F_1 و F_2:

(الف) نشان دهید که تابع بولی $E = F_1 + F_2$ حاوی مجموع مینترم‌های F_1 و F_2 است.

(ب) نشان دهید که تابع بولی $G = F_1F_2$ فقط حاوی مینترم‌های مشترک F_1 و F_2 است.

۲-۸ جدول درستی تابع زیر را لیست کنید:

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

۲-۹ اعمال منطقی اجرا شده بر روی رشته‌ای از بیت‌ها با ملاحظه هر جفت بیت انجام می‌شود (به آن عمل بیتی می‌گویند) با فرض داشتن دو رشته 8 بیتی $A = 10101101$ و $B = 10001110$، نتایج بیتی را پس از انجام اعمال منطقی زیر ارزیابی کنید: (الف) AND، (ب) OR، (پ) XOR، (ت) NOT A، (ث) NOT B.

۲-۱۰ نمودارهای منطقی را برای عبارات بولی زیر رسم نمایید:

(الف) $Y = A'B' + B(A + C)$ (ب) $Y = BC + AC'$

(پ) $Y = A + CD$ (ت) $Y = (A + B)(C' + D)$

۲-۱۱ با فرض تابع بولی زیر:

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

(الف) آن را با گیت‌های AND ، OR و وارونگر (NOT) پیاده‌سازی کنید.
(ب) آن را با گیت‌های OR و NOT پیاده کنید.
(پ) آن را با گیت AND و NOT پیاده نمایید.

۲-۱۲ تابع بول T_1 و T_2 را به حداقل لیترالهای ساده نمایید.

A	B	C	T_1	T_2
0	0	0	1	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

۲-۱۳ جمع منطقی همه مینترم‌های تابع بول n متغیره 1 است.
(الف) جمله بالا را برای $n = 3$ ثابت کنید.
(ب) روالی عمومی، برای اثبات پیشنهاد نمایید.

۲-۱۴ جدول درستی توابع زیر را بدست آورده و هر تابع را بصورت جمع مینترم‌ها و ضرب ماکسترم‌ها بیان کنید:
(الف) $(xy + z)(y + xz)$ (ب) $(A' + B)(B' + C)$
(پ) $y'z + wxy' + wxz' + w'x'z$

۲-۱۵ با فرض تابع بولی زیر

$$F = xy'z + x'y'z + w'xy + wx'y + wxy$$

(الف) جدول درستی آن را پیدا کنید.
(ب) نمودار منطقی را با استفاده از عبارت بولی اولیه بدست آورید.
(پ) تابع را با استفاده از جبر بول به حداقل لیترال‌ها ساده کنید.
(ت) جدول درستی تابع را از عبارت ساده شده بدست آورید و نشان دهید که مشابه بخش (الف) است.
(ث) نمودار منطقی را از عبارت ساده شده رسم کنید و تعداد گیت‌های کل آن را با نمودار (ب) مقایسه نمایید.

۲-۱۶ تابع زیر را بصورت جمع مینترم‌ها و ضرب ماکسترم‌ها بیان کنید:

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

۲-۱۷ متمم توابع زیر را به جمع مینترم‌ها بنویسید:
(الف) $F(A, B, C, D) = \Sigma(0, 2, 6, 11, 13, 14)$ (ب) $F(x, y, z) = \Pi(0, 3, 6, 7)$

۲-۱۸ توابع زیر را به فرم متعارف دیگری بنویسید:
(الف) $F(x, y, z) = \Sigma(1, 3, 7)$ (ب) $F(A, B, C, D) = \Pi(0, 1, 2, 3, 4, 6, 12)$

۲-۱۹ عبارات بولی زیر را به جمع حاصلضرب‌ها و ضرب حاصل جمع‌ها تبدیل کنید.
(الف) $(AB + C)(B + C'D)$ (ب) $x' + x(x + y')(y + z')$

۲-۲۰ بدون ساده‌کردن توابع زیر نمودار منطقی آنها را رسم نمایید:

(الف) $BC' + AB + ACD$ (ب) $(A + B)(C + D)(A' + B + D)$

(پ) $(AB + A'B')(CD' + C'D)$

۲-۲۱ نشان دهید که دوگان XOR با متمم آن برابر است.

۲-۲۲ با جایگزینی عبارت بولی معادل اعمال دودویی در جدول ۸-۲ نشان دهید که:

(الف) عمل نهی نه جابجاپذیر و نه شرکت‌پذیر است.

(ب) عمل XOR جابجاپذیر و اشتراک‌پذیر است.

۲-۲۳ نشان دهید که گیت NAND با منطق مثبت، یک گیت NOR با منطق منفی است و بالعکس.

مراجع

1. BOOLE, G. 1954. *An Investigation of the Laws of Thought.* New York: Dover.
2. SHANNON, C. E. A symbolic analysis of relay and switching circuits. *Trans. AIEE* 57 (1938): 713-723.
3. HUNTINGTON, E. V. Sets of independent postulates for the algebra of logic. *Trans. Am. Math. Soc.,* 5 (1904): 288-309.
4. MANO, M. M. and C. R. Kime. 2000. *Logic and Computer Design Fundamentals,* 2nd ed. Upper Saddle River, NJ: Prentice Hall.
5. DIETMEYER, D. L. 1988. *Logic Design of Digital Systems,* 3rd ed. Boston: Allyn Bacon.



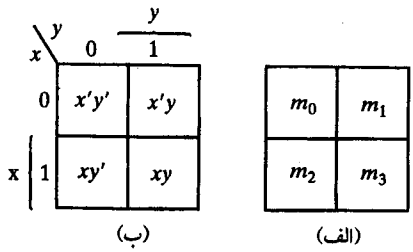
حداقل سازی در سطح گیت

۳-۱ روش نقشه

پیچیدگی گیت‌های منطقی دیجیتال که یک تابع بول را پیاده‌سازی می‌کنند، مستقیماً به پیچیدگی عبارات جبری که توسط آن تابع پیاده‌سازی می‌شوند بستگی دارد. گرچه جدول درستی یک تابع نمایش منحصر به فردی دارد، اما وقتی به صورت جبری بیان شود، می‌تواند فرم‌های متفاوتی داشته باشد. عبارات بول را می‌توان همان‌طور که در بخش ۴-۲ بحث شد، به صورت جبری ساده کرد. با این وجود، این روش حداقل‌سازی به دلیل کمبود قوانین خاص در پیشگویی مرحله بعدی فرآیند دستکاری، مشکل است. روش نقشه، روالی ساده را برای ساده‌سازی توابع بول پیش پا می‌گذارد. این روش را می‌توان فرم مصور جدول درستی تصور کرد. روش نقشه را نقشه کارنو یا نقشه K هم می‌نامند.

نقشه نموداری است متشکل از مربعات که هر مربع یک مینترم از تابع را نشان می‌دهد. چون هر تابع بول را می‌توان به مجموعی از مینترم‌ها نشان داد، بنابراین نتیجه می‌شود که یک تابع بولی در نقشه را می‌توان با مربعاتی که مینترم‌های متعلق به آنها در تابع وجود دارد به صورت گرافیکی شناسایی کرد. در واقع نقشه، نمایشی عینی از همه راه‌هایی است که یک تابع ممکن است در فرم استاندارد داشته باشد. با تشخیص همه الگوهای مختلف، کاربر می‌تواند عبارات جبری مختلفی برای یک تابع بدست آورده و از میان آنها ساده‌ترین را انتخاب کند.

عبارات ساده شده حاصل از نقشه همیشه به یکی از دو فرم استاندارد جمع حاصلضرب‌ها و ضرب حاصل جمع‌ها می‌باشد. فرض بر این است که ساده‌ترین عبارت جبری، دارای حداقل جملات با کمترین لیترال در هر جمله باشد. این فرض نموداری با حداقل گیت را فراهم نموده و تعداد ورودی‌ها به گیت نیز حداقل خواهد بود. بعد خواهیم دید که ساده‌ترین عبارت منحصر به فرد نیست. گاهی ممکن است دو یا چند عبارت بیابیم که معیار حداقل‌سازی را برآورد. در این حالت هر یک از دو حل رضایت‌بخش خواهد بود.



شکل ۱-۳. نقشه دو متغیره

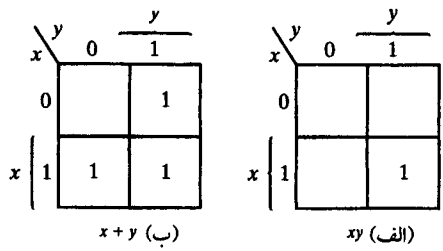
نقشه دو متغیره

نقشه دو متغیره در شکل ۱-۳ (الف) نشان داده شده است. در این نقشه چهار مینترم برای دو متغیر وجود دارد؛ از این رو نقشه متشکل از چهار مربع است، که هر یک متعلق به یک مینترم می‌باشد. نقشه (ب) برای نمایش ارتباط بین مربعات و دو متغیر x و y دوباره ترسیم شده است. 0 و 1 موجود در هر سطر و ستون مقدار متغیر را نشان می‌دهند. متغیر x در سطر 0 پریم‌دار و در سطر 1 بدون پریم است. به طور مشابه y در ستون 0 پریم‌دار و در ستون 1 بدون پریم می‌باشد.

اگر مربع‌هایی را که مینترم آنها متعلق به تابع مفروضی است با علامتی مشخص کنیم، روش مفید دیگری برای نمایش هر یک از 16 تابع ممکن از دو متغیر بدست می‌آید. به عنوان مثال تابع xy در شکل ۲-۳ (الف) دیده می‌شود. چون xy برابر m_3 است، یک 1 در داخل مربع متعلق به m_3 قرار می‌دهیم. به طور مشابه تابع x+y در نقشه شکل ۲-۳ (ب) نشان داده شده است که در آن سه مربع با 1 علامت زده شده‌اند. این مربعات از مینترم‌های تابع به دست آمده‌اند:

$$m_1 + m_2 + m_3 = x'y + xy' + xy = x + y$$

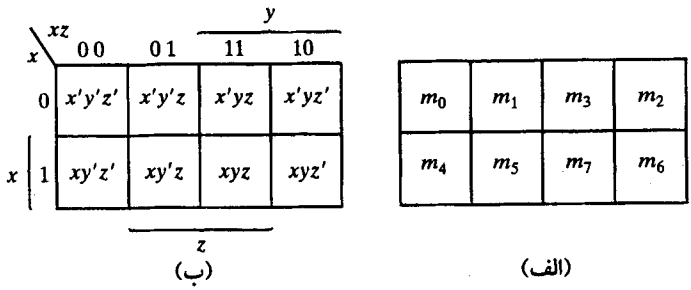
سه مربع از تلاقی x در سطر دوم و متغیر y در ستون دوم، که ناحیه متعلق به x یا y را پوشش می‌دهند، نیز بدست می‌آید.



شکل ۲-۳. نمایش توابع در نقشه

نقشه سه متغیره

یک نقشه سه متغیره در شکل ۳-۳ مشاهده می‌شود. برای سه متغیر هشت مینترم وجود دارد. بنابراین نقشه از هشت مربع تشکیل یافته است. توجه کنید که مینترم‌ها براساس ترتیب دودویی مرتب



شکل ۳-۳. نقشه سه متغیره

نشده‌اند، بلکه ترتیبشان براساس کدگری جدول (۴-۱) فهرست شده است. ویژگی این ترتیب این است که هنگام عبور از یک ستون به ستون مجاور تنها یک بیت از نظر مقدار تغییر می‌کند. برای نشان دادن رابطه بین مربع‌ها و سه متغیر نقشه، بخش (ب) با اعدادی در هر سطر و هر ستون علامت‌گذاری شده است. مثلاً مربع متعلق به m_5 مربوط به سطر 1 و ستون 01 است. وقتی دو عدد در کنار هم قرار گیرند عدد دودویی 101 حاصل می‌شود که معادل دهدهی آن عدد 5 می‌باشد. به طریقی دیگر هم می‌توان به مربع $m_5 = xy'z$ نگاه کرد به این ترتیب که بگوییم m_5 در سطر مربوط به x و ستون متعلق به $y'z$ است (ستون 01). توجه کنید که هر متغیر در چهار مربع مقدار 0 و در چهار مربع دیگر مقدار 1 را دارد. به منظور تفکیک، هر متغیر را در خانه‌های 1 بدون پریم و در خانه‌های 0 با پریم نشان می‌دهیم. برای سادگی، متغیر را با سمبل حرفی‌اش در زیر مربعاتی که بدون پریم هستند می‌نویسیم.

جهت درک برتری‌های نقشه کارنو در ساده‌سازی توابع بول، باید خاصیت مربع‌های همجوار را مشخص کنیم. تنها اختلاف بین هر دو مربع مجاور در نقشه این است که در یکی متغیری با پریم و در دیگری بدون پریم ظاهر می‌شود. مثلاً، m_5 و m_7 در دو مربع مجاور قرار دارند. متغیر y در m_5 پریم‌دار و در m_7 بدون پریم است، ضمن این که دو متغیر دیگر در هر دو مربع یکسانند. با توجه به اصول جبر بول، نتیجه می‌گیریم که جمع دو مینترم در مربع‌های مجاور را می‌توان به یک جمله AND متشکل از دو لیترال ساده کرد. برای روشن شدن مطلب، مجموع دو مربع همجوار مانند m_5 و m_7 را ملاحظه کنید.

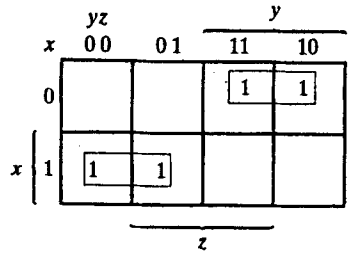
$$m_5 + m_7 = xy'z + xyz = xz(y' + y) = xz$$

در اینجا دو مربع در متغیر y با هم اختلاف دارند که هنگام تشکیل جمع دو مینترم حذف می‌شود. بنابراین هر دو مینترمی که در دو مربع مجاور با هم OR شوند موجب حذف متغیری می‌گردند که در آن دو مینترم متفاوت‌اند. مثال‌های زیر روال حداقل‌سازی یک تابع بول را با یک نقشه توضیح می‌دهد.

مثال ۱-۳

تابع بولی زیر را ساده کنید.

$$F(x, y, z) = \Sigma(2, 3, 4, 5)$$



شکل ۴-۳. نقشه مثال ۱-۳ $F(x, y, z) = \Sigma(2, 3, 4, 5) = x'y + xy'$

ابتدا یک 1 در هر مربعی که مینترم تابع را نشان دهد قرار می‌دهیم. این کار در شکل ۴-۳ به این ترتیب انجام شده است که مربعات مینترم‌های 010، 011، 100 و 101 با 1 علامت زده شده‌اند. قدم بعدی یافتن مربع‌های مجاور است. این کار در نقشه با زیرمربع‌هایی که هر یک دو عدد 1 را در بر می‌گیرند صورت گرفته است. زیرمربع یا مستطیل بالای سمت راست ناحیه پوشش یافته با $x'y$ را شامل می‌شود. این دو مربع در سطر 0 قرار دارند که با x' و نیز در دو ستون آخر با y نشان داده می‌شوند. به طور مشابه مستطیل پایین سمت چپ جمله ضرب xy' را نشان می‌دهد (سطر دوم نشان دهنده x و دو ستون چپ نیز y' است). جمع منطقی این دو جمله ضرب، عبارت ساده شده را نتیجه می‌دهد.

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

❑

مواردی وجود دارد که در آنها دو مربع همجوارند ولی به هم نچسبیده‌اند. در شکل ۳-۳، m_0 مجاور m_2 و m_4 مجاور m_6 است زیرا مینترم‌ها تنها در یک متغیر با هم اختلاف دارند. این مطلب به راحتی با کمک جبر قابل اثبات است.

$$m_0 + m_2 = x'y'z' + x'yz' = x'z'(y' + y) = x'z'$$
$$m_4 + m_6 = xy'z' + xyz' = xz' + (y' + y) = xz'$$

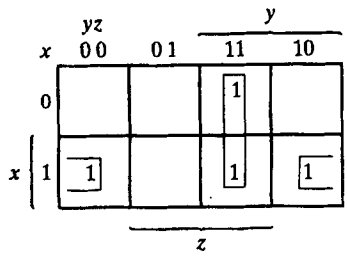
در نتیجه ما باید تعریف مربع‌های همجوار را اصلاح کنیم تا این حالت و دیگر حالات مشابه را نیز شامل شود. این تصحیح بدین صورت انجام می‌گیرد که نقشه کشیده شده در یک سطح از دو لبه سمت چپ و راست مجاور تصور شوند.

مثال ۲-۳

تابع بول زیر را ساده کنید.

$$F(x, y, z) = \Sigma(3, 4, 6, 7)$$

نقشه این تابع در شکل ۵-۳ ترسیم شده است. در این شکل 4 مربع با 1 علامت خورده‌اند که هر کدام متعلق به یک مینترم است. دو مربع همجوار در ستون سوم با هم ترکیب شده‌اند تا جمله دو لیترال yz را به وجود آورند.



شکل ۵-۳. نقشه مثال ۲-۳ $F(x, y, z) = \Sigma(3, 4, 6, 7) = yz + xz'$

دو مربع باقیمانده هم براساس تعریف جدید مجاورند و در نمودار با نیم مربع‌ها محصور شده‌اند. این دو مربع، وقتی ترکیب شوند جمله دو لیترالی xz' را بدست می‌دهند. بنابراین تابع ساده شده به فرم زیر است.

$$F = yz + xz'$$

❑

اکنون به ترکیب چهار مربع همجوار در نقشه سه متغیره توجه نمایید. چنین ترکیبی نشان دهنده جمع منطقی چهار مینترم مجاور است و نتیجه این ترکیب، تولید عبارتی با تنها یک متغیر است. به عنوان مثال جمع منطقی چهار مینترم مجاور 0، 2، 4 و 6 عبارت را به جمله یک لیترالی z' کاهش می‌دهد.

$$\begin{aligned} m_0 + m_2 + m_4 + m_6 &= x'y'z' + x'yz' + xy'z' + xyz' \\ &= x'z'(y' + y) + xz'(y' + y) \\ &= x'z' + xz' = z'(x' + x) = z' \end{aligned}$$

تعداد مربعات مجاوری که ممکن است ترکیب شوند همواره برابر توانی از 2، مانند 1، 2، 4 و 8 می‌باشد. هر چقدر تعداد بیشتری از مربعات همجوار ترکیب شوند جمله حاصلضرب نتیجه، تعداد کمتری لیترال خواهد داشت.

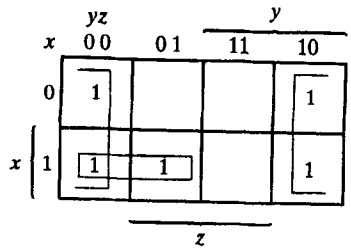
یک مربع یک مینترم را نمایش می‌دهد و دارای سه لیترال است.
دو مربع مجاور یک جمله دو لیترال را نشان می‌دهند.
چهار مربع همجوار یک جمله با یک لیترال را نشان می‌دهند.
هشت مربع همجوار که تمام نقشه را می‌پوشانند همواره تابع 1 را تولید می‌کنند.

مثال ۳-۳

تابع بول زیر را ساده کنید.

$$F(x, y, z) = \Sigma(0, 2, 4, 5, 6)$$

نقشه تابع F در شکل ۶-۳ نشان داده شده است. ابتدا چهار مربع مجاور در اولین و آخرین ستون را با هم ترکیب می‌کنیم تا جمله تک لیترال z' بدست آید. تنها مینترم باقیمانده که متعلق به مینترم 5 است با مربع



شکل ۶-۳. نقشه مثال ۳-۲ $F(x, y, z) = \Sigma(0, 2, 4, 5, 6) = z' + xy'$

مجاورش که قبلاً به کار رفته، ترکیب می‌گردد. این کار نه تنها مجاز است بلکه مفید نیز می‌باشد، زیرا دو مربع مجاور جمله دو لیترالی xy' را تولید می‌کنند در حالی که یک مربع تنها، جمله سه لیترال $xy'z$ را نمایش می‌دهد. تابع ساده به صورت زیر است.

$$F = z' + xy'$$

❑

اگر تابعی به صورت مجموع مینترم‌ها بیان نشود می‌توان از نقشه برای بدست آوردن مینترم‌های تابع استفاده کرد و سپس تابع را به صورت جملاتی با حداقل لیترال‌ها ساده نمود. البته باید عبارت جبری حتماً به صورت جمع حاصلضرب‌ها باشد. هر جمله ضرب را می‌توان با نقشه‌ای متشکل از یک، دو یا چند مربع در نقشه نشان داد. آنگاه مینترم‌های تابع مستقیماً از جدول استخراج می‌شوند.

مثال ۴-۳

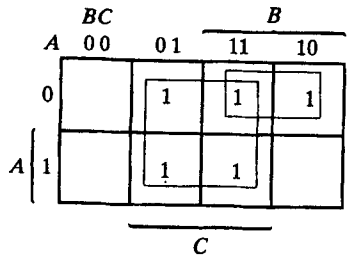
تابع بول زیر را ساده کنید.

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

(الف) آن را به مجموع مینترم‌ها نشان دهید.

(ب) و سپس عبارت مجموع حاصلضرب حداقل را پیدا کنید.

سه جمله ضرب در عبارت دو لیترال دارند و در نقشه سه متغیره، هر یک با دو مربع نشان داده شده‌اند. دو مربع مربوط به جمله اول، $A'C$، در شکل ۷-۳ از تلاقی A' (اولین سطر) و C (دو ستون میانی) بدست



شکل ۷-۳. نقشه مثال ۴-۳ $F = A'C + A'B + AB'C + BC = C + A'B$

می‌آید تا مربعات 001 و 011 را بدهند. توجه کنید که وقتی 1ها را در مربعات می‌گذارید، ممکن است یک 1 را که از جمله قبلی در آن قرار داده شده بیابید. این نکته برای دومین جمله، A′B، رخ می‌دهد که دو عدد 1 در مربع‌های 01 و 010 قرار دارند. مربع 011 با جمله اول، A′C، مشترک است، بنابراین تنها یک 1 در آن قرار داده می‌شود. به همین ترتیب می‌بینیم که جمله AB′C متعلق به مربع 101، یعنی مینترم 5 است، و جمله BC متعلق به دو مربع 011 و 111 می‌باشد. تابع جمعاً پنج مینترم دارد و در نقشه شکل ۷-۳ هم با پنج عدد 1 نشان داده شده است. مینترم‌هایی که مستقیماً از نقشه خوانده می‌شوند عبارتند از 1، 2، 3، 5 و 7. تابع را می‌توان به صورت جمع مینترم‌ها نشان داد.

$$F(A, B, C) = \Sigma(1, 2, 3, 5, 7)$$

عبارت جمع حاصلضرب مفروض اولیه چندین جمله دارد. همانطور که در نقشه مشاهده می‌شود می‌توان آن را ساده کرده و عبارتی دو جمله‌ای بدست آورد.

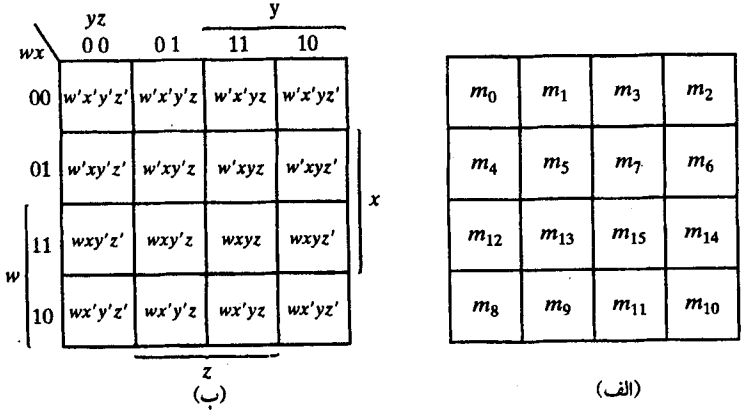
$$F = C + A'B$$

❑

۳-۲ نقشه چهار متغیره

نقشه توابع بول چهار متغیره در شکل ۸-۳ نشان داده شده است. در (الف) 16 جمله مینترم فهرست شده و به هر یک مربعی تخصیص داده شده است. در (ب) نقشه دوباره رسم شده تا بیانگر ارتباط بین چهار متغیر باشد. سطرها و ستون‌ها براساس کدگری شماره‌گذاری شده‌اند، و بین هر دو سطر یا ستون مجاور تنها یک رقم تغییر می‌کند. مینترم متعلق به هر مربع از ترکیب شماره سطر و شماره ستون آن بدست می‌آید. مثلاً وقتی اعداد سطر سوم (11) و ستون دوم (01) ترکیب شوند عدد دودویی 1101 حاصل می‌گردد، که معادل 13 دهدهی است. بنابراین، مربع در سطر سوم و ستون دوم مینترم m_{13} را نمایش می‌دهد.

ساده کردن توابع بول چهار متغیره مشابه با روش به کار رفته برای توابع سه متغیره است. مربعات مجاور مربعاتی هستند که در کنار یکدیگرند. به علاوه نقشه در سطحی واقع است و لبه‌های بالا و پایین و چپ و راست آن نیز مجاور است تا به این ترتیب مربعات همجوار را بسازند. مثلاً m_0 و m_2 و نیز m_{11}.



شکل ۸-۳. نقشه چهار متغیره

و m_3 هر کدام مربعات مجاور را می‌سازند. ترکیب مربعات همجوار به راحتی با بررسی نقشه چهار متغیره قابل تشخیص است:

یک مربع یک مینترم را نمایش می‌دهد، و جمله آن چهار لیترالی است.
دو مربع همجوار یک جمله سه لیترالی را می‌سازند.
چهار مربع همجوار یک جمله دو لیترالی را نشان می‌دهند.
هشت مربع همجوار یک جمله یک لیترالی را نمایش می‌دهند.
شانزده مربع همجوار تابعی برابر 1 را تولید می‌کنند.

هیچ ترکیب دیگری از مربع‌ها نمی‌تواند تابع را ساده کند. دو مثال زیر روال بکار رفته برای توابع بول چهار متغیره را نشان می‌دهد.

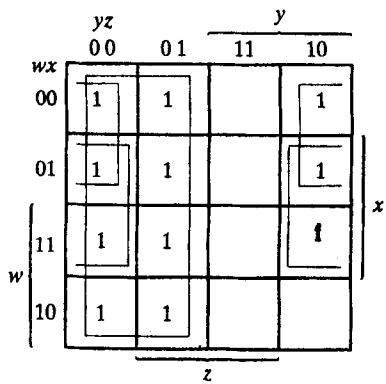
مثال ۵-۳

تابع بول زیر را ساده کنید.

$$F(w, x, y, z) = \Sigma(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14)$$

چون تابع چهار متغیر دارد، باید از نقشه چهار متغیره استفاده کرد. مینترم‌های لیست شده در مجموع فوق با 1ها در نقشه شکل ۹-۳علامت زده شده‌اند. هشت 1 مجاور می‌توانند با هم ترکیب شده و جمله تک لیترالی y' را نتیجه دهند. سه 1 باقیمانده در سمت راست نمی‌توانند با هم ترکیب و جمله ساده‌ای بدهند. آنها باید به صورت دو یا چهار مربع مجاور با هم ترکیب شوند. هر چقدر تعداد مربعات ترکیب شده بیشتر باشد، تعداد لیترال‌ها در جمله کمتر خواهد بود.

در این مثال دو 1 فوقانی سمت راست با دو 1 فوقانی در سمت چپ ترکیب شده و جمله $w'z'$ را می‌دهند. توجه داشته باشید که می‌توان یک مربع را بیش از یک بار به کار برد. حال فقط یک مربع در سطر سوم و ستون چهارم (مربع 1110) باقیمانده است. در عوض انتخاب این مربع به تنهایی، آن را با مربع‌هایی که قبلاً به



شکل ۹-۳. نقشه مثال ۵-۳

$$F(w, x, y, z) = \Sigma(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14) = y' + w'z' + xz'$$

کار رفته‌اند برای ایجاد مربع‌های مجاور ترکیب می‌کنیم. این مربعات شامل دو سطر میانی و دو ستون انتهایی بوده و جمله xz' را تولید می‌کنند. تابع ساده شده به صورت زیر است:

$$F = y' + w'z' + xz'$$

❑

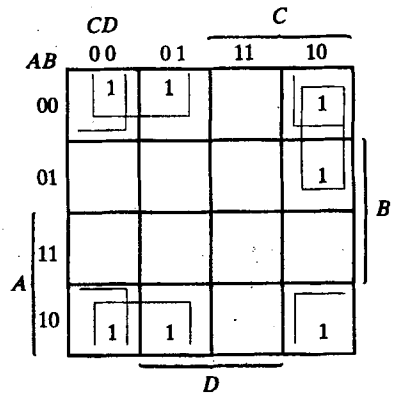
مثال ۶-۳

تابع زیر را ساده کنید.

$$F = A'B'C' + B'CD' + A'BCD' + AB'C'$$

ناحیه مفروش شده با این تابع شامل مربعاتی است که در شکل ۱۰-۳ با 1 علامت زده شده است. این تابع دارای چهار متغیر بوده و همانطور که دیده می‌شود سه جمله سه لیترالی و یک جمله چهار لیترالی دارد. هر جمله سه لیترالی در نقشه با دو مربع نمایش داده شده است. مثلاً $A'B'C'$ در مربعات 0000 و 0001 نشان داده شده است. تابع را می‌توان با انتخاب چهار 1 در گوشه‌ها و ترکیب آنها برای بدست آوردن جمله $B'D'$ ساده کرد. این عمل مجاز است زیرا وقتی نقشه را سطحی تصور کنیم که لبه‌های چپ و راست و لبه‌های پایین و بالای آن با هم مجاورند، این چهار مربع همجوار خواهند بود. دو 1 سمت چپ در سطر بالا با دو 1 در سطر پایین ترکیب می‌شوند تا جمله $B'C'$ حاصل شود. تنها 1 باقیمانده را به صورت دو مربع ترکیب می‌کنیم تا $A'CD'$ حاصل گردد. تابع ساده شده به صورت زیر خواهد بود.

$$F = B'D' + B'C' + A'CD'$$

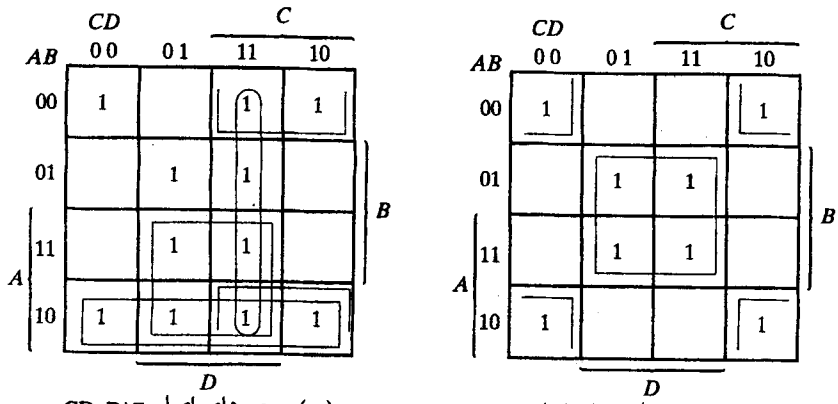


شکل ۱۰-۳. نقشه مثال ۶-۲

❑ $F = A'B'C' + B'CD' + A'BCD' + AB'C' = B'D' + B'C' + A'CD'$

موجب‌های اصلی

هنگام انتخاب مربع‌های مجاور در یک نقشه باید مطمئن شویم که همه مینترم‌های تابع هنگام



شکل ۱۱-۳. ساده‌سازی با موجب‌های اصلی

ترکیب مربع‌ها پوشش داده شده‌اند. همچنین باید تعداد جملات در عبارت حداقل شود و هر جمله‌ای که مینترم آن قبلاً به وسیله دیگر جملات به کار رفته نیز کنار گذاشته شود. گاهی نیز ممکن است دو یا سه عبارت بر معیار ساده‌سازی صحّه بگذارند. روش ترکیب مربع‌ها در نقشه را می‌توان سینماتیک‌تر کرد به شرطی که مفهوم جملات موجب‌های اصلی و موجب‌های اصلی اساسی خوب فهمیده شوند. یک موجب اصلی جمله‌ای حاصلضربی است که از ترکیب حداکثر مربعات مجاور به هم حاصل می‌گردد. اگر مینترمی در یک مربع تنها با یک موجب اصلی پوشش یابد، به آن موجب اصلی اساسی گوییم.

موجب‌های اصلی یک تابع را می‌توان با ترکیب حداکثر تعداد مربعات ممکن بدست آورد. این بدان معنی است که یک 1 تنها اگر در مجاورت هر 1 دیگر در نقشه نباشد، یک موجب اصلی است. دو 1 مجاور به شرطی یک موجب اصلی را ایجاد می‌کنند که در داخل یک گروه چهارتایی مربع‌ها واقع نباشند. چهار 1 مجاور یک موجب اصلی را تشکیل می‌دهند بشرطی که در یک گروه از هشت مربع همجوار نباشند و بهمین ترتیب. موجب اصلی اساسی با نظاره بر مربعات 1 و وارسی تعداد موجب‌های اصلی که آن را پوشش می‌دهد تعیین می‌گردد. یک موجب اصلی، اساسی است اگر تنها موجب اصلی باشد که مینترم را پوشش می‌دهد.

تابع چهار متغیره زیر را در نظر بگیرید:

$$F(A, B, C, D) = \Sigma(0, 2, 3, 5, 7, 8, 9, 10, 11, 13, 15)$$

مینترم‌های تابع با 1 در نقشه‌های شکل ۱۱-۳ علامت زده شده‌اند. بخش (الف) از شکل، دو موجب اصلی اساسی را نشان می‌دهد. یک موجب، اساسی است زیرا تنها یک راه برای پوشش m_0 در چهار مربع مجاور وجود دارد. این چهار مربع جمله $B'D'$ را تعریف می‌کنند. به طور مشابه، برای ترکیب m_5 با چهار مربع مجاور تنها یک راه وجود دارد و جمله BD از آن حاصل می‌گردد. این دو موجب اصلی اساسی هشت مینترم را پوشش می‌دهند. سه مینترم باقیمانده m_3، m_9 و m_{11} باید بعد ملاحظه شوند.

شکل ۱۱-۳ (ب) همه راه‌های ممکن که سه مینترم با موجب‌های اصلی پوشش می‌یابند را نشان می‌دهد. مینترم m_3 می‌تواند با موجب اصلی CD یا $B'C$ پوشش یابد. مینترم m_9 با هر یک از AD یا AB' پوشش می‌یابد. مینترم m_{11} نیز با هر یک از چهار موجب اصلی می‌تواند پوشش پیدا کند. عبارت ساده شده از جمع منطقی دو موجب اصلی اساسی، و هر دو موجب اصلی دیگر که مینترم‌های m_3 و m_9 و m_{11} را پوشش دهند بدست می‌آید. چهار امکان برای بیان تابع با چهار جمله ضرب که هر یک دو لیترال دارند وجود دارد:

$$\begin{aligned} F &= BD + B'D' + CD + AD \\ &= BD + B'D' + CD + AB' \\ &= BD + B'D' + B'C + AD \\ &= BD + B'D' + B'C + AB' \end{aligned}$$

مثال فوق نشان داد که شناسایی موجب‌های اصلی در نقشه در تعیین صور متفاوت تابع ساده شده کمک مؤثری می‌نمایند.

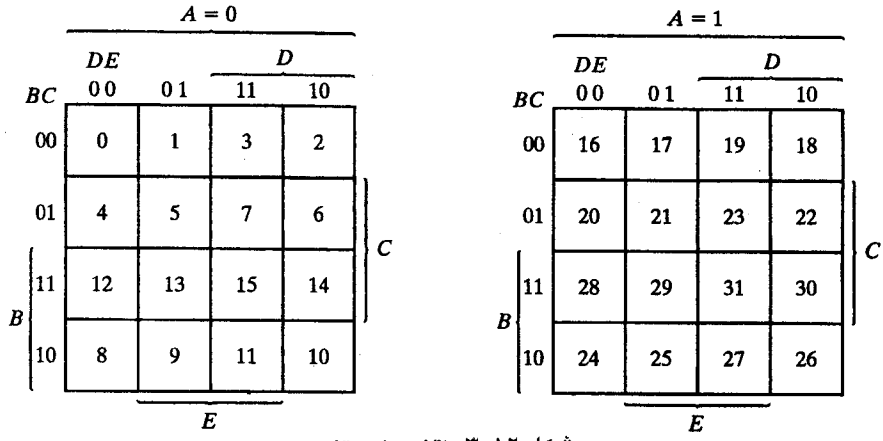
روال یافتن عبارت ساده شده از نقشه لازم می‌دارد که ابتدا تمام موجب‌های اصلی اساسی را معین کنیم. تابع ساده شده از جمع منطقی همه موجب‌های اصلی اساسی، به علاوه دیگر موجب‌های اصلی حاصل می‌گردد. این موجب‌های اصلی ممکن است برای پوشش مینترم‌های باقیمانده‌ای که در موجب اصلی اساسی وجود ندارند لازم باشد. گاهی بیش از یک راه برای ترکیب مربعات وجود دارد و هر ترکیب هم ممکن است عبارت ساده شده یکسانی را تولید کند.

۳-۳ نقشه پنج متغیره

استفاده از نقشه‌هایی که بیش از چهار متغیر دارند چندان ساده نیست. یک نقشه پنج متغیره به 32 مربع و نقشه شش متغیره به 64 مربع نیاز دارد. وقتی تعداد متغیرها زیاد شود، تعداد مربعات هم به طور بی‌رویه‌ای افزایش می‌یابند و یافتن مربعات همجوار بیش از پیش به شکل هندسی وابسته می‌گردد.

یک نقشه پنج متغیره در شکل ۱۲-۳ نشان داده شده است. این نقشه، از دو نقشه چهار متغیره با متغیرهای A ، B ، C ، D و E تشکیل یافته و متغیر A آن دو را از هم تفکیک کرده است. نقشه چهار متغیره سمت چپ 16 مربعی را نشان می‌دهد که در آن $A = 0$ است، و دیگر نقشه چهار متغیره، مربعات مربوط به $A = 1$ را نمایش می‌دهد. مینترم‌های 0 تا 15 متعلق به A=0 و مینترم‌های 16 تا 31 متعلق به A=1 است. هر نقشه چهار متغیره وقتی جداگانه بررسی شود همجواری تعریف شده قبلی خود را حفظ می‌کند. به علاوه هر مربع از نقشه A=0 با مربع متناظرش در مربع A=1 همجوار است. مثلاً مینترم 4 با مینترم 20 و مینترم 15 با 31 مجاور است. بهترین راه تجسم این قانون برای مربع‌های همجوار این است که این دو نیم نقشه را بر روی یکدیگر تصور کنیم. هر دو مربعی که روی هم قرار گیرند مجاور شناخته می‌شوند.

با پیگیری روشی که برای نقشه پنج متغیره به کار رفت، می‌توان نقشه شش متغیره را با 4 نقشه چهار متغیره بدست آورد تا 64 مربع مورد نیاز حاصل گردد. نقشه‌هایی با شش یا تعداد بیشتری متغیر، نیاز به



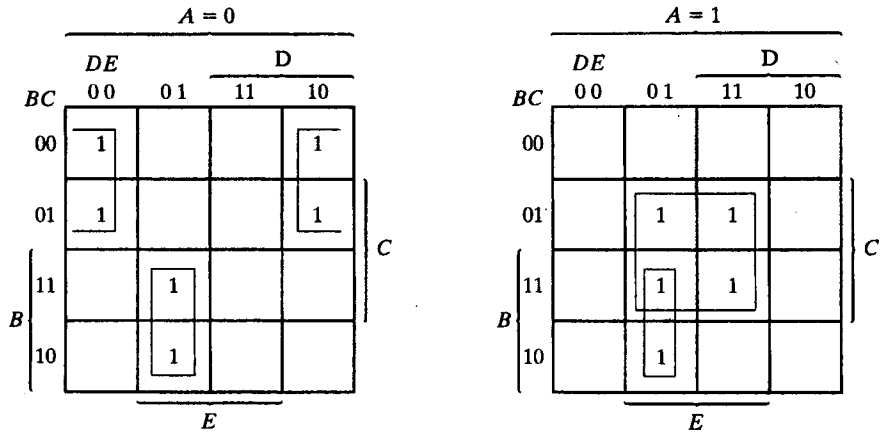
شکل ۱۲-۳. نقشه پنج متغیر

تعداد بی‌شماری مربع داشته و استفاده از آنها غیرعملی است. روش دیگر، استفاده از برنامه‌های کامپیوتری در ساده‌سازی توابع بول با متغیرهای بی‌شمار می‌باشد.

با بررسی و در نظر گرفتن تعریف جدید همجواری مربعات، می‌توان نشان داد که 2^k مربع همجوار به ازاء $k = (0, 1, 2, ..., n)$ در یک نقشه n متغیره ناحیه را مشخص می‌کند که نمایش‌دهنده جمله‌ای با $n - k$ لیترال است. برای این که عبارت فوق مفهوم داشته باشد باید همیشه n بزرگتر از k باشد. وقتی $n = k$ است، تمام سطح نقشه ترکیب شده و تابع یگانی (1) را تولید می‌کند. جدول ۱-۳ رابطه بین تعداد مربعات مجاور و تعداد لیترال در هر جمله را نشان می‌دهد. مثلاً هشت مربع مجاور ناحیه‌ای را در نقشه پنج متغیره ترکیب می‌کنند تا یک جمله دو متغیره حاصل شود.

جدول ۱-۳. رابطه بین تعداد مربعات مجاور و تعداد لیترال‌ها در یک جمله

	تعداد مربعات مجاور	تعداد لیترال‌ها در یک جمله در یک نقشه n متغیره			
K	2^k	$n = 2$	$n = 3$	$n = 4$	$n = 5$
0	1	2	3	4	5
1	2	1	2	3	4
2	4	0	1	2	3
3	8		0	1	2
4	16			0	1
5	32				0



شکل ۱۳-۳. نقشه مثال ۷-۲ $F = A'B'E' + BD'E + ACE$

مثال ۷-۳

تابع بول زیر را ساده کنید.

$$F(A, B, C, D, E) = (0, 2, 4, 6, 9, 13, 21, 23, 25, 29, 31)$$

نقشه پنج متغیره برای این تابع در شکل ۱۳-۳ دیده می‌شود. در بخشی از نقشه که متعلق به مینترم‌های 0 تا 15 است، A = 0 بوده و در آن شش مینترم مقدار 1 را دارند. پنج مینترم دیگر به بخش A = 1 متعلق است. چهار مربع مجاور در نقشه A = 0 با هم ترکیب شده‌اند تا جمله A′B′E′ را بدهند. توجه کنید که باید A′ را نیز در جمله منظور کنیم زیرا تمام مربع‌ها متعلق به نقشه A = 0 می‌باشند. دو مربع در ستون 01 و دو سطر آخر در هر دو بخش نقشه مشترکند. بنابراین آنها چهار مربع مجاور را تشکیل داده و جمله سه متغیره BD′E را می‌سازند. در اینجا متغیر A آورده نشده است زیرا مربع‌های مجاور به هر دو A = 0 و A = 1 متعلق‌اند. جمله ACE از چهار مربع همجوار در نقشه A = 1 بدست می‌آید. تابع ساده شده جمع منطقی سه جمله می‌باشد.

$$F = A'B'E' + BD'E + ACE$$

❑

۳-۴ ساده‌سازی با ضرب حاصل جمع‌ها

در تمام مثال‌های قبلی توابع بول حاصل از نقشه به فرم جمع حاصلضرب‌ها بیان شدند. با کمی اصلاح می‌توان فرم ضرب حاصل جمع‌ها را بدست آورد.

روال تهیه یک تابع حداقل برحسب ضرب حاصل جمع‌ها از خواص اصلی توابع بول حاصل می‌گردد. 1های واقع در مربع‌های نقشه نشانگر مینترم‌های تابع است. مینترم‌هایی که در تابع ذکر نشوند

متمم تابع را بیانگرند. با توجه به این مطلب مشاهده می‌کنیم که متمم یک تابع به وسیله مربع‌هایی که با 1 علامت‌زنی نشده‌اند بیان می‌گردد. اگر در مربع‌های خالی 0 قرار داده و آنها را با روش مربع‌های همجوار ترکیب کنیم عبارت ساده شده متمم تابع یعنی F' را بدست خواهیم آورد. متمم F' به ما تابع F را باز می‌گرداند. به دلیل عمومیت تئوری دمورگان تابع حاصل به طور خودکار به صورت ضرب حاصل جمع‌هاست. بهترین روش برای تشریح این مطلب ارائه یک مثال است.

مثال ۸-۳

تابع بولی زیر را (الف) به صورت جمع حاصلضرب‌ها، (ب) ضرب حاصل جمع‌ها ساده کنید.

$$F(A, B, C, D) = \Sigma(0, 1, 2, 5, 8, 9, 10)$$

1های موجود در نقشه شکل ۱۴-۳ همه مینترم‌های تابع را نمایش می‌دهند. مربع‌هایی که با 0 علامت زده شده‌اند مینترم‌های غایب در F را نشان می‌دهند، بنابراین متمم F را بیانگر هستند. ترکیب مربعات حاوی 1ها تابع ساده شده را به صورت جمع حاصلضرب‌ها بدست می‌دهد:

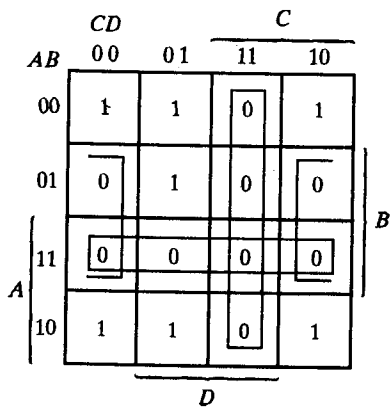
(الف) $$F = B'D' + B'C' + A'C'D$$

اگر مربعات حاوی 0ها را ترکیب کنیم، تابع متمم ساده شده بدست خواهد آمد:

$$F' = AB + CD + BD'$$

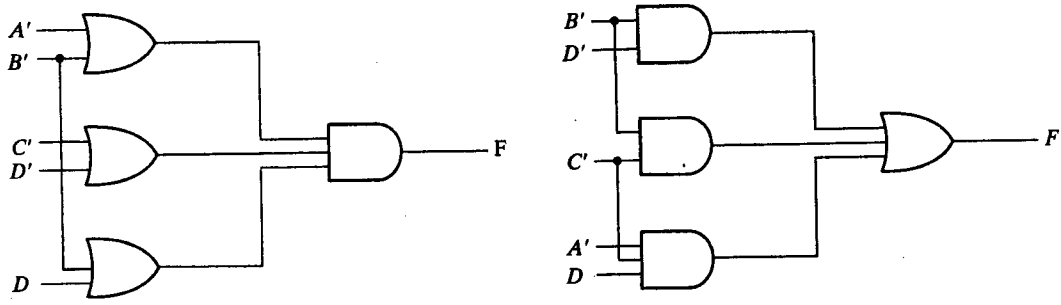
با اعمال تئوری دمورگان (استفاده از دوگان و متمم کردن هر متغیر طبق آنچه در بخش ۴-۲ دیدیم) تابع ساده شده را به صورت ضرب حاصل جمع‌ها بدست می‌آوریم:

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



شکل ۱۴-۳. نقشه برای مثال ۸-۳

$$F(A, B, C, D) = \Sigma(0, 1, 2, 5, 8, 9, 10) = B'D' + B'C' + A'C'D = (A' + B')(C' + D')(B' + D)$$



$F = (A' + B')(C' + D')(B' + D)$ (ب)

$F = B'D' + B'C' + A'C'D$ (الف)

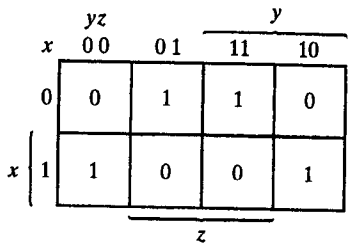
شکل ۱۵-۳. پیاده‌سازی با گیت تابع مثال ۸-۳

پیاده‌سازی عبارت ساده شده حاصل از مثال ۸-۳ در شکل ۱۵-۳ دیده می‌شود. عبارت جمع‌حاصلضرب‌ها در بخش (الف) با گروهی از گیت‌های AND پیاده‌سازی شده است. خروجی گیت‌های AND نیز به ورودی‌های یک گیت OR متصل گردیده است. همان تابع به صورت ضرب حاصل جمع‌ها در شکل (ب) با تعدادی گیت OR ، که هر یک متعلق به یک جمله OR است، پیاده‌سازی شده و خروجی آنها به یک AND منتهی گشته است. در هر دو حال فرض بر این است که متمم متغیرها نیز مستقیماً در دسترسند و بنابراین نیازی به وارونگر نمی‌باشد. الگوهای ایجاد شده در شکل ۱۵-۳ یک سری روش‌های کلی می‌باشند که به وسیله آنها هر تابع بول استاندارد قابل پیاده‌سازی است. در جمع حاصلضرب‌ها، گیت‌های AND به یک OR ختم می‌شوند و در ضرب حاصل جمع‌ها گیت‌های OR به یک AND متصل می‌گردند. هر یک از دو پیکربندی فوق دارای دو سطح از گیت‌ها می‌باشند. بنابراین پیاده‌سازی یک تابع استاندارد را پیاده‌سازی دو سطحی می‌گویند.

مثال ۸-۳ روالی را برای محاسبه فرم ساده شده یک تابع برحسب ضرب حاصل جمع‌ها، وقتی تابع ابتدا به صورت جمع مینترم‌ها است، نشان می‌دهد. این روال هنگامی که تابع در آغاز برحسب ماکسترم‌ها بیان شود نیز معتبر است. برای مثال به جدول درستی (۲-۳) که تابع F را تعریف می‌کند

جدول ۲-۳. جدول درستی تابع F

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



شکل ۱۶-۳. نقشه برای تابع جدول (۲-۳)

توجه نمایید. این تابع به صورت جمع مینترم‌ها چنین بیان می‌شود:

$$F(x, y, z) = \Sigma(1, 3, 4, 6)$$

در ضرب ماکسترم‌ها تابع به صورت زیر است:

$$F(x, y, z) = \Pi(0, 2, 5, 7)$$

به بیان دیگر، 1های تابع، مینترم‌ها را نشان می‌دهند و 0های آن بیانگر جملات ماکسترم هستند. نقشه این تابع در شکل ۱۶-۳ دیده می‌شود. برای ساده کردن این تابع، در مربع مربوط به هر جمله مینترم که تابع به ازاء آن، مقدار 1 دارد، عدد 1گذاشته و بقیه مربع‌ها را با 0 پر می‌کنیم. از طرف دیگر اگر تابع به فرم ضرب ماکسترم‌ها داده شده باشد در ابتدا در مربعاتی که جملات آن در تابع است 0 قرار می‌دهیم و بقیه مربع‌ها با 1 پر می‌شوند. سپس تابع می‌تواند به یکی از فرم‌های استاندارد ساده شود. برای جمع حاصلضرب‌ها، 1ها را با هم ترکیب می‌کنیم و خواهیم داشت:

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

برای ضرب حاصل جمع‌ها، 0ها را با هم ترکیب می‌کنیم تا متمم تابع ساده شده به صورت زیر حاصل شود:

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

که نشان می‌دهد تابع XOR متمم تابع هم‌ارزی (XNOR) است (بخش ۶-۲). با متمم‌گیری مجدد از F' تابع ساده شده را به ضرب حاصل جمع‌ها بدست خواهیم آورد.

$$F = (x' + z')(x + z)$$

برای وارد کردن یک تابع در یک نقشه که برحسب ضرب حاصل جمع‌ها بیان شده است، می‌باید متمم تابع را بدست آورد و در آن مربع‌های مربوطه را با 0 پر کرد. مثلاً تابع

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

را می‌توان با متمم‌گیری از آن وارد جدول کرد.

$$F' = ABC + B'D'$$

آنگاه مربع‌هایی که مینترم‌های F' را تشکیل می‌دهند با 0 پر می‌کنیم. بقیه مربع‌ها را با 1 پر می‌نماییم.

۳-۵ حالات بی‌اهمیت

جمع منطقی مینترم‌های مربوط به یک تابع شرایطی را که تحت آن تابع برابر 1 است، مشخص

می‌نماید. تابع در ازاء بقیه مینترم‌ها 0 است. در این حالت فرض بر این است که همه ترکیبات مقادیر برای متغیرهای تابع معتبرند. در عمل کاربردهایی وجود دارند که در آنها در ازاء ترکیبات معینی از متغیرها، تابع مشخص نیست. مثلاً یک کد دودویی چهار بیتی برای ارقام دهدهی دارای شش ترکیب است که به کار نرفته‌اند و در نتیجه نامشخص تصور می‌گردند. توابعی که در ازاء ترکیبی از ورودی‌ها خروجی‌های نامشخص دارند، تابع غیرکامل نامیده می‌شود. در بسیاری از کاربردها، توجهی به مقدار منتسب به تابع در ازاء مینترم‌های نامعین نخواهیم داشت. به این دلیل مرسوم است که همه مینترم‌های نامشخص در تابع را حالات بی‌اهمیت بخوانیم. از حالات بی‌اهمیت می‌توان برای ساده‌سازی بیشتر عبارت بول در یک نقشه استفاده کرد.

باید توجه داشت که یک مینترم بی‌اهمیت ترکیبی از متغیرهاست که مقدار منطقی آن نامشخص است. به این دلیل نمی‌توان یک حالت بی‌اهمیت را در نقشه با 1 نشان داد زیرا این عمل به این معنی است که تابع برای ترکیب خاص از ورودیها همواره برابر 1 می‌باشد. به طور مشابه گذاشتن 0 در مربع‌های نقشه به معنی 0 بودن همیشگی تابع در آن حالت است. برای تفکیک حالت بی‌اهمیت از 1ها و 0ها از X استفاده می‌کنیم. بنابراین هر X در داخل یک مربع از نقشه به این معنی است که تخصیص 1 یا 0 به تابع به ازاء یک مینترم خاص فاقد اهمیت است.

وقتی مربع‌های مجاور انتخاب می‌گردند تا تابع در جدول ساده شود، مینترم‌های بی‌اهمیت با این ایده که ساده‌ترین فرم برای تابع بدست آید، برابر 1 یا 0 فرض می‌شوند. در ساده‌سازی تابع می‌توانیم با توجه به ساده‌ترین فرم ممکن برای تابع، به حالات بی‌اهمیت 0 یا 1 دهیم.

مثال ۹-۳

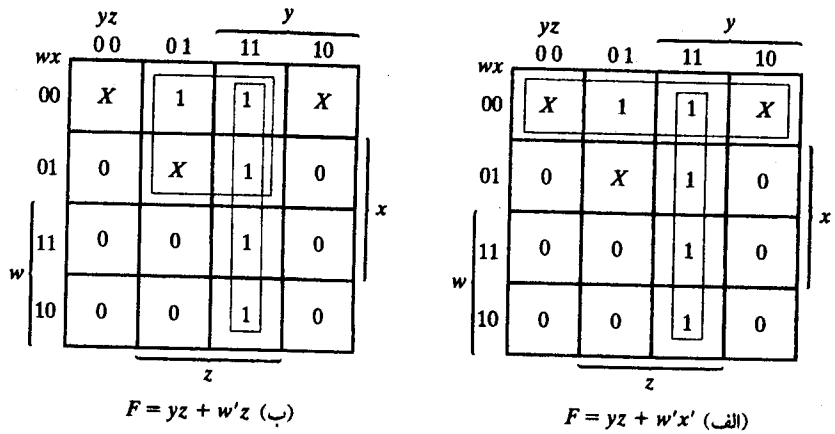
تابع بول زیر را ساده کنید.

$$F(w, x, y, z) = \Sigma(1, 3, 7, 11, 15)$$

که حالات بی‌اهمیت زیر را داراست.

$$d(w, x, y, z) = \Sigma(0, 2, 5)$$

مینترم‌های F ترکیباتی از متغیرها هستند که تابع را برابر 1 می‌کنند. مینترم‌های d مینترم‌های بی‌اهمیتی هستند که ممکن است به آنها 0 یا 1 تخصیص داده شود. ساده‌سازی نقشه در شکل ۱۷-۳ نشان داده شده است. مینترم‌های F با 1 علامت زده شده‌اند، مینترم‌های d با X علامت‌گذاری شده‌اند و بقیه مربع‌ها با 0 پر شده‌اند. برای بدست آوردن عبارت جمع حاصلضرب‌های ساده شده باید هر پنج 1 موجود در نقشه به حساب آیند، ولی بسته به روش ساده‌سازی ممکن است Xها را در نظر بگیریم و یا نگیریم. جمله yz چهار مینترم در سومین ستون را پوشش می‌دهد. مینترم باقیمانده m_1 می‌تواند با مینترم m_3 ترکیب شده و جمله سه لیترالی $w'x'z$ را بدهند. با این وجود با احتساب یک یا دو X همجوار، می‌توانیم چهار مربع مجاور را ترکیب نماییم تا جمله دو متغیره حاصل گردد. در بخش (الف) از نمودار، مینترم‌های بی‌اهمیت 0 و 2 با 1 جایگزین شده‌اند



شکل ۱۷-۳. مثال با حالات بی‌اهمیت

و تابع ساده شده به صورت زیر است.

$$F = yz + w'x'$$

در بخش (ب) از نمودار، مینترم بی‌اهمیت 5 با 1 جایگزین شده و آنگاه تابع ساده شده به فرم زیر است:

$$F = yz + w'z$$

هر یک از دو عبارت شرایط بیان شده برای این مثال را دارا هستند. ❑

مثال قبل نشان داد که مینترم‌های بی‌اهمیت در نقشه در ابتدا با Xها علامت خورده‌اند و فرض می‌شود که بتوانند 0 و یا 1 بشوند. انتخاب 0 و یا 1 به روش ساده کردن تابع غیرکامل وابسته است. پس از انتخاب، تابع ساده شده حاصل متشکل از مجموع مینترم‌ها است و در آنها مینترم‌هایی که در آغاز نامعلوم بوده ولی بعد به عنوان 1 انتخاب شده‌اند نیز وجود خواهند داشت. دو عبارت ساده شده حاصل در مثال ۹-۳ را در نظر بگیرید:

$$F(w, x, y, z) = yz + w'x' = \Sigma(0, 1, 2, 3, 7, 11, 15)$$
$$F(w, x, y, z) = yz + w'z = \Sigma(1, 3, 5, 7, 11, 15)$$

هر دو عبارت شامل مینترم‌های 1، 3، 7، 11 و 15 می‌باشند که تابع F را برابر 1 می‌کنند. مینترم‌های بی‌اهمیت در آن دو به طور متفاوتی به کار گرفته شده‌اند و در اولین عبارت مینترم‌های 0 و 2 برابر 1 گرفته شده و مینترم 5 با انتخاب 0 حذف شده است. در دومین عبارت مینترم 5 برابر 1 و مینترم‌های 0 و 2 با مقدار 0 جایگزین شده‌اند. دو عبارت توابعی را نشان می‌دهند که فرم جبری متفاوتی دارند. هر دو مینترم‌های مشخص شده را می‌پوشانند ولی هر یک مینترم‌های بی‌اهمیت متفاوتی را پوشش می‌دهند. مادامی که تابع مشخص شده غیرکامل است، هر دو عبارت قابل قبول‌اند زیرا تنها اختلاف در مقدار F مینترم‌های بی‌اهمیت می‌باشند.

می‌توان عبارت ضرب حاصل جمع‌ها را هم برای تابع شکل ۱۷-۳ بدست آورد. در این حال، تنها راه برای ترکیب 0ها جایگزینی مینترم‌های بی‌اهمیت شماره 0 و 2 با مقدار 0 می‌باشد و به این ترتیب تابع متمم ساده شده بدست می‌آید:

$$F' = z' + wy'$$

با متمم‌گیری از طرفین، عبارت ساده شده به صورت ضرب حاصل جمع‌ها خواهد بود:

$$F(w, x, y, z) = z(w' + y) = \Sigma(1, 3, 5, 7, 11, 15)$$

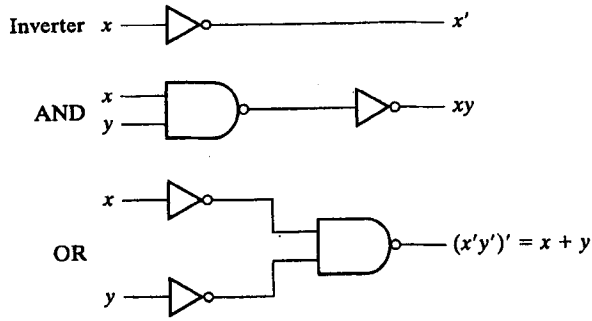
در این حال، ما مینترم‌های شماره 0 و 2 را با مقدار 0 و مینترم 5 را با 1 جایگزین کرده‌ایم.

۳-۶ پیاده‌سازی با NAND و NOR

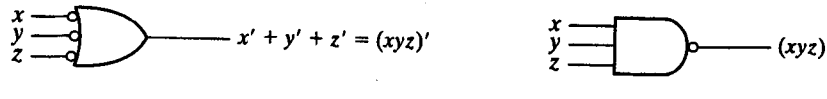
مدارهای دیجیتال اغلب به جای AND و OR با گیت‌های NAND و NOR ساخته می‌شوند. ساختن گیت‌های NAND و NOR با اجزاء الکترونیکی ساده‌تر بوده و به عنوان گیت‌های پایه در تمام خانواده‌های IC های دیجیتال به کار می‌روند. به دلیل مزیت گیت‌های NAND و NOR در طراحی مدارهای دیجیتال، قواعد و روال‌هایی برای تبدیل توابع بول بیان شده برحسب AND ، OR و NOT به نمودارهای منطقی معادل برحسب NAND و NOR بوجود آمده است.

مدارهای NAND

گیت NAND را یک گیت یونیورسال می‌گویند زیرا هر سیستم دیجیتالی را می‌توان با آن پیاده‌سازی کرد. برای این که نشان دهیم هر تابع بولی قابل پیاده‌سازی با گیت‌های NAND می‌باشد، کافی است فقط نشان دهیم که اعمال منطقی AND ، OR و NOT را می‌توان با NAND پیاده‌سازی کرد. این کار درشکل ۱۸-۳ نشان داده شده است. عمل متمم از یک گیت NAND یک ورودی که دقیقاً مثل NOT عمل می‌کند حاصل می‌گردد. عمل AND نیاز به دو گیت NAND دارد. اولی عمل NAND و دومی عمل NOT را انجام می‌دهد. عمل OR از طریق یک گیت NAND و دو NOT در هر ورودی حاصل می‌شود.



شکل ۱۸-۳. عملیات منطقی با گیت‌های NAND



Invert-OR (ب)　　AND-invert (الف)

شکل ۱۹-۳. دو سمبل گرافیکی برای گیت NAND

راهی مناسب برای پیاده‌سازی یک تابع بول با گیت‌های NAND ، بدست آوردن تابع بول ساده شده برحسب عملگرهای بولی و سپس تبدیل تابع به منطق NAND است. تبدیل یک عبارت جبری از AND ، OR و NOT به NAND به سادگی با دستکاری نمودار AND–OR به نمودار NAND انجام می‌شود.

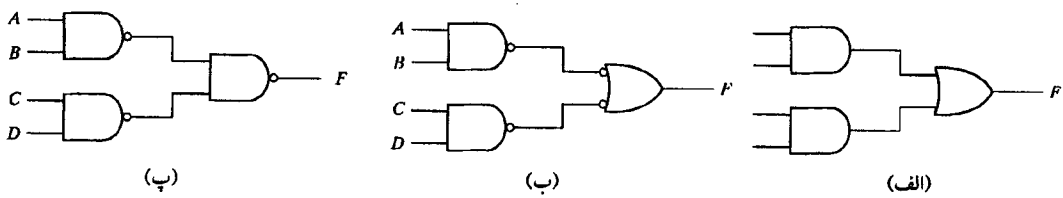
برای ساده سازی تبدیل به منطق NAND ، بهتر است سمبل گرافیکی دیگری برای گیت تعریف کنیم. دو سمبل گرافیکی معادل برای گیت NAND در شکل ۱۹-۳ دیده می‌شود. سمبل AND–invert قبلاً معرفی شد و متشکل بود از یک سمبل AND و به دنبال آن یک دایره کوچک که به آن حباب گفته شد و نقش متمم‌سازی را داشت. به همین ترتیب می‌توان گیت NAND را با سمبل گرافیکی OR با حبابی در هر ورودی نشان داد. سمبل invert–OR برای گیت NAND از تئوری دمورگان و با توجه به این قرارداد که دوایر کوچک به منزله متمم کردن هستند بدست می‌آید. دو سمبل گرافیکی فوق در طراحی و تحلیل مدارهای NAND مفیدند. وقتی هر دو سمبل در یک نمودار به کار روند گوییم مدار با علائم مخلوط نشان داده شده است.

پیاده‌سازی دو سطحی

برای پیاده‌سازی توابع بول با گیت‌های NAND ، تابع باید به فرم جمع حاصلضرب‌ها باشد. برای درک ارتباط بین عبارت جمع حاصلضرب‌ها و معادل NAND آن، به نمودارهای منطقی شکل ۲۰-۳ توجه کنید. هر سه نمودار معادل بوده و تابع زیر را پیاده می‌نمایند.

$$F = AB + CD$$

در (الف) تابع با گیت‌های AND و OR پیاده‌سازی شده است. در (ب)، گیت‌های AND با گیت‌های NAND و گیت OR نیز با یک گیت NAND که با سمبل invert–OR مشخص شده پیاده‌سازی شده است. توجه داشته باشید که یک حباب به معنی متمم و دو حباب در یک مسیر دوبار متمم‌سازی را



(پ)　　(ب)　　(الف)

شکل ۲۰-۳. سه راه پیاده‌سازی تابع F = AB + CD

نشان می‌دهند، پس می‌توانند حذف شوند. حذف حباب‌ها در گیت‌های (ب) مدار شکل (الف) را نتیجه می‌دهد. بنابراین دو نمودار یک تابع را پیاده‌سازی می‌کنند پس معادل‌اند.

در شکل ۲۰-۳ (پ)، خروجی گیت NAND با سمبل گرافیکی AND–invert ترسیم شده است. هنگام رسم نمودارهای منطقی NAND ، هر یک از دو مدار (ب) یا (پ) پذیرفته است. مدار (ب) از علائم مخلوط استفاده کرده است و رابطه مستقیم‌تری را با عبارت بول پیاده شده نشان می‌دهد. صحت پیاده‌سازی NAND در شکل ۲۰-۳ (پ) می‌تواند به صورت جبری تحقیق شود. تابعی که این شکل را پیاده کرده است به سادگی با تئوری دمورگان قابل تبدیل به جمع حاصلضرب‌هاست:

$$F = \big((AB)'(CD)'\big)' = AB + CD$$

مثال ۱۰-۳

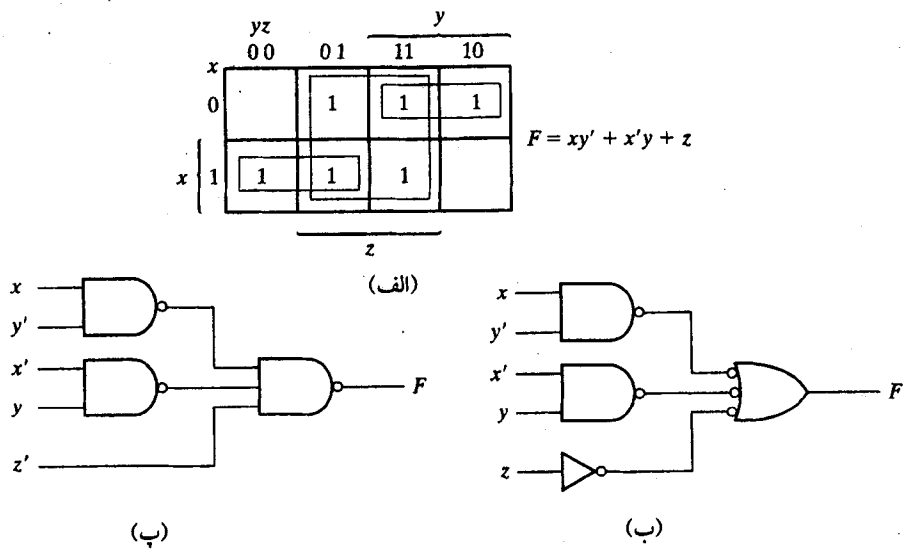
تابع بول زیر را با گیت‌های NAND پیاده کنید.

$$F(x, y, z) = (1, 2, 3, 4, 5, 7)$$

اولین قدم در تبدیل، ساده‌سازی تابع در جمع حاصلضرب‌هاست. این کار به کمک نقشه شکل ۲۱-۳ (الف) انجام شده است و تابع حاصل به صورت زیر است.

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

پیاده‌سازی NAND دو سطحی در شکل ۲۱-۳ (ب) به صورت علائم مخلوط دیده می‌شود. توجه کنید که ورودی z باید یک گیت NAND یک ورودی باشد تا حباب موجود در گیت سطح دوم را جبران کند. روش دیگری برای ترسیم نمودار منطقی در شکل ۲۱-۳ (پ) نشان داده شده است. در اینجا تمام



شکل ۲۱-۳. حل مثال ۱۰-۳

گیت‌های NAND با سمبل یکسان ترسیم شده‌اند. وارونگر با ورودی z حذف شده است ولی متغیر ورودی متمم شده و با z' نشان داده شده است. ❑

روالی که در مثال قبل توصیف شد بیان می‌دارد که یک تابع بول می‌تواند با دو سطح (یا دو طبقه) گیت NAND پیاده‌سازی شود. روال تهیه نمودار منطقی NAND از تابع بول به قرار زیر است:

۱- تابع را ساده کرده آن را به فرم جمع حاصل ضرب‌ها بنویسید.
۲- برای هر جمله ضرب موجود در تابع که حداقل دو لیترال دارد یک گیت NAND بکشید. ورودی به هر یک گیت NAND لیترال‌های جمله‌اند. این مجموعه، گیت‌های سطح اول را تشکیل می‌دهد.
۳- در سطح دوم، یک گیت NAND با ورودی‌هایی که از خروجی‌های سطح اول می‌آیند بکشید. از سمبل گرافیکی AND–invert یا invert–OR استفاده نمایید.
۴- یک جمله با یک لیترال نیاز به یک وارونگر در اولین سطح دارد. با این وجود، اگر تک لیترال متمم شده است می‌توان آن را مستقیماً به گیت NAND سطح دوم وصل کرد.

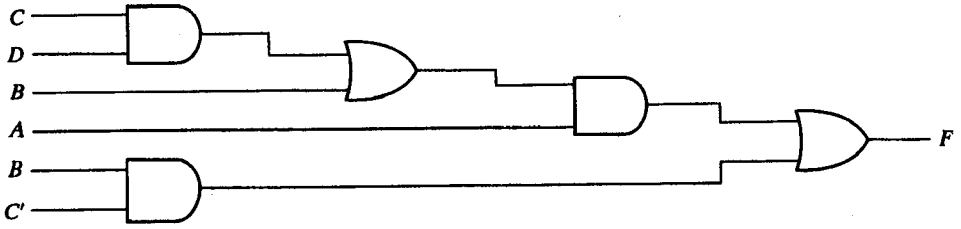
مدارهای NAND چند سطحی

فرم استاندارد بیان توابع بول، پیاده‌سازی دو سطحی (طبقه) را نتیجه می‌دهد. مواردی وجود دارد که طراحی سیستم‌های دیجیتال یک ساختار گیتی با سه یا چند طبقه را نتیجه می‌دهد. رایج‌ترین روش طراحی مدارهای چند طبقه بیان تابع بول برحسب عملیات AND ، OR و NOT می‌باشد. سپس می‌توان تابع را با گیت‌های AND و OR پیاده‌سازی کرد. آنگاه در صورت لزوم تمام مدار را می‌توان به NAND تبدیل نمود. به عنوان مثال تابع بول زیر را ملاحظه کنید:

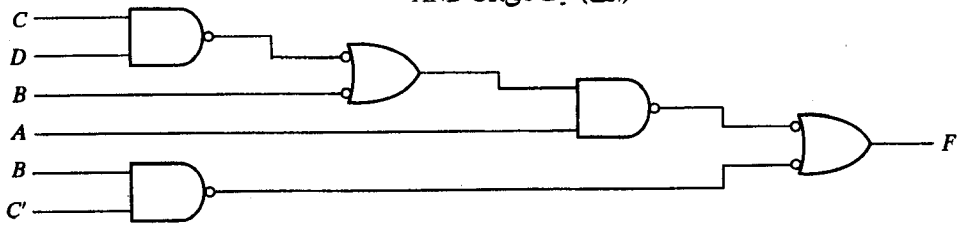
$$F = A(CD + B) + BC'$$

گرچه می‌توان پرانتزها را حذف کرده و عبارت را به صورت جمع حاصلضرب استاندارد در آورد، ولی آن را به عنوان یک مدار چند طبقه مورد بررسی قرار می‌دهیم. پیاده‌سازی AND–OR برای آن در شکل ۲۲-۳ (الف) نشان داده شده است. در مدار، چهار طبقه گیت دیده می‌شود. اولین طبقه دو گیت AND دارد. دومین طبقه یک گیت OR و به دنبال آن یک AND در طبقه سوم آمده و در طبقه چهارم هم یک OR ملاحظه می‌شود. با استفاده از علائم مخلوط، می‌توان یک نمودار منطقی با الگویی از سطوح متناوب AND و OR را به سادگی به مدار NAND تبدیل کرد. این تبدیل در شکل ۲۲-۳ (ب) دیده می‌شود. روال این است که هر گیت AND را به سمبل AND–invert و هر گیت OR را به invert–OR تبدیل کنیم. مدار NAND حاصل، عملکرد یکسانی با نمودار AND–OR دارد به شرطی که در هر مسیر دو حباب وجود داشته باشد. حباب مربوط به ورودی B موجب می‌شود تا یک متمم اضافی صورت گیرد که باید آن را با متغیر ورودی مذکور به لیترال B' جبران کرد.

روال کلی نمودار چند طبقه AND–OR به نمودار تمام NAND با استفاده از علائم مخلوط به شرح زیر است:



(الف) گیت‌های AND-OR



(ب) گیت‌های NAND

شکل ۲۲-۳. پیاده‌سازی $F = A(CD + B) + BC'$

۱- همه گیت‌های AND را با استفاده از سمبل‌های گرافیکی invert–AND به گیت NAND تبدیل کنید.
۲- همه گیت‌های OR را با سمبل‌های گرافیکی invert–OR به NAND تبدیل نمایید.
۳- همه حباب‌ها را در نمودار چک کنید. برای هر حبابی که در یک مسیر جبران نشده است یک وارونگر (گیت NAND یک ورودی) وارد کنید و یا لیترال ورودی را متمم نمایید.

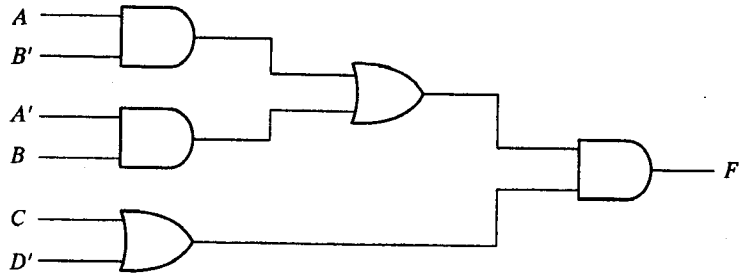
به عنوان مثالی دیگر تابع بول چند سطحی زیر را ملاحظه کنید.

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

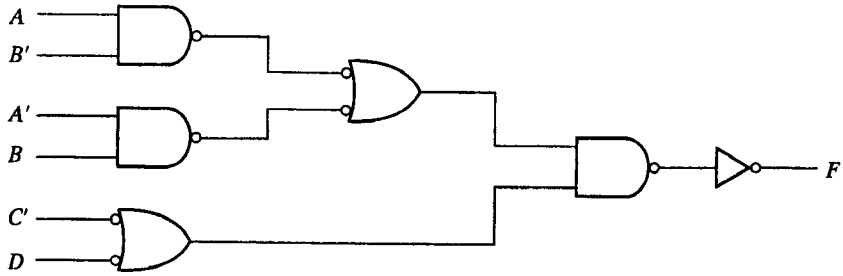
پیاده‌سازی AND–OR تابع در شکل ۲۳-۳ (الف) ملاحظه می‌شود که در آن سه طبقه گیت به کار رفته است. در بخش (ب) نمودار فرم تبدیل شده به NAND آن با علائم مخلوط دیده می‌شود. دو حباب اضافی مربوط به ورودی‌های C و D' موجب متمم شدن آنها به C' و D می‌گردد. حباب موجود در گیت NAND خروجی، مقدار خروجی را متمم می‌کند بنابراین برای بدست آوردن مقدار اصلی تابع مجبوریم یک گیت وارونگر در خروجی به کار ببریم.

پیاده‌سازی NOR

عمل NOR دوگان NAND است. بنابراین تمام روال‌ها و قوانین منطق NOR دوگان روال‌های متناظر و قوانین حاصل در منطق NAND هستند. NOR ، گیت یونیورسال دیگری است که برای پیاده‌سازی هر تابع بول به کار می‌رود. پیاده‌سازی اعمال AND ، OR و NOT با گیت‌های NOR در شکل ۲۴-۳ ملاحظه می‌گردد. عمل متمم، با گیت NOR یک ورودی حاصل شده و عیناً مثل وارونگر

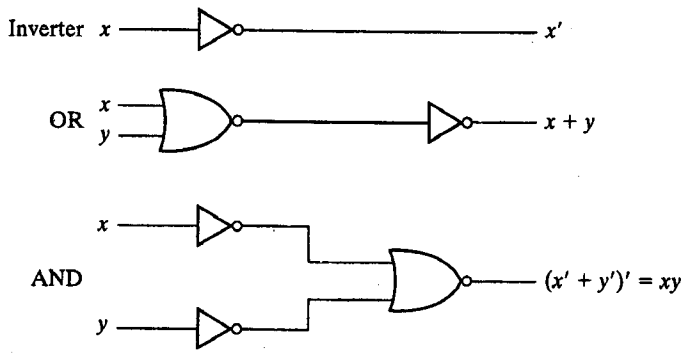


(الف) گیت‌های AND-OR



(ب) گیت‌های NAND

شکل ۲۳-۳. پیاده‌سازی $F = (AB' + A'B)(C + D')$



شکل ۲۴-۳. عملیات منطقی با گیت‌های NOR

عمل می‌کند. عمل OR نیاز به دو گیت NOR و عمل AND از یک گیت NOR که در هر ورودی‌اش یک وارونگر دارد حاصل می‌شود.

دو سمبل گرافیکی برای علائم مخلوط در شکل ۲۵-۳ دیده می‌شود. سمبل OR–invert عمل NOR را با یک OR و به دنبال آن یک متمم تعریف می‌کند. هر دو سمبل عمل NOR یکسانی را به نمایش می‌گذارند و از نظر منطقی با توجه به تئوری دمورگان یکی هستند.