معماری کامپیوتر

نویسنده

موریس مانو

فهرست مطالب

فصل ۱: مدارهای منطقی دیجیتال ۱

۱-۱ کامپیوترهای دیجیتال ۱
۱-۲ گیت‌های منطقی ۴
۱-۳ جبر بول ۷
۱-۴ ساده سازی با نقشه ۱۱
۱-۵ مدارهای ترکیبی ۱۸
۱-۶ فلیپ فلاپها ۲۲
۱-۷ مدارهای ترتیبی ۲۹
مسائل ۳۸

فصل ۲: قطعات دیجیتال ۴۱

۲-۱ مدارهای مجتمع ۴۱
۲-۲ دیکدرها ۴۴
۲-۳ مولتی پلکسرها ۴۸
۲-۴ ثبات‌ها ۵۰
۲-۵ شیفت رجیسترها ۵۴
۲-۶ شمارنده‌های دودویی ۵۷
۲-۷ واحد حافظه ۶۱
مسائل ۶۵

فصل ۳: نمایش داده‌ها ۶۸

۳-۱ انواع داده‌ها ۶۸
۳-۲ متمم‌ها ۷۶
۳-۳ نمایش ممیز - ثابت ۷۹
۳-۴ نمایش ممیز شناور ۸۵
۳-۵ انواع دیگر کدهای دودویی ۸۶
۳-۶ کدهای آشکارسازی خطا ۸۹
مسائل ۹۲

فصل ۴: انتقال ثبات‌ها و ریزعمل‌ها ۹۴

۴-۱ زبان انتقال ثبات ۹۴
۴-۲ انتقال ثبات ۹۶

www.homework.ir

۴-۳ انتقال‌های گذرگاهی و حافظه‌ای ۹۹
۴-۴ ریزعمل‌های حسابی ۱۰۳
۴-۵ ریزعمل‌های منطقی ۱۱۰
۴-۶ ریزعمل‌های شیفت ۱۱۵
۴-۷ واحد حساب، منطق و شیفت ۱۱۸
مسائل ۱۲۰

فصل ۵: سازمان و طراحی یک کامپیوتر پایه ۱۲۴
۵-۱ کدهای دستورالعمل‌ها ۱۲۴
۵-۲ ثبات‌های کامپیوتر ۱۲۸
۵-۳ دستورالعمل‌های کامپیوتر ۱۳۳
۵-۴ زمانبندی و کنترل ۱۳۶
۵-۵ سیکل دستورالعمل ۱۴۰
۵-۶ دستورالعمل‌های ارجاع به حافظه ۱۴۵
۵-۷ ورودی - خروجی و وقفه ۱۵۰
۵-۸ تشریح کامل کامپیوتر ۱۵۷
۵-۹ طراحی کامپیوتر پایه ۱۵۹
۵-۱۰ طراحی مدار منطقی انباره ۱۶۴
مسائل ۱۶۸

فصل ۶: برنامه نویسی کامپیوتر پایه ۱۷۳
۶-۱ مقدمه ۱۷۳
۶-۲ زبان ماشین ۱۷۴
۶-۳ زبان اسمبلی ۱۷۹
۶-۴ اسمبلر ۱۸۳
۶-۵ حلقه در برنامه‌نویسی ۱۹۰
۶-۶ برنامه‌نویسی اعمال حسابی و منطقی ۱۹۳
۶-۷ زیرروال‌ها ۱۹۸
۶-۸ برنامه‌نویسی ورودی - خروجی ۲۰۲
مسائل ۲۰۹

فصل ۷: کنترل ریز برنامه‌نویسی شده ۲۱۳
۷-۱ حافظه کنترل ۲۱۳
۷-۲ دنبال کردن آدرس ۲۱۶

www.homework.ir

۳-۷ مثال ریزبرنامه ۲۲۱
۴-۷ طراحی واحد کنترل ۲۳۱
مسائل ۲۳۶

فصل ۸: واحد مرکزی پردازش ۲۴۰
۱-۸ مقدمه ۲۴۰
۲-۸ سازمان ثبات‌های عمومی ۲۴۱
۳-۸ سازمان پشته ۲۴۶
۴-۸ قالب دستورالعمل‌ها ۲۵۴
۵-۸ روشهای آدرس دهی ۲۵۹
۶-۸ انتقال و دستکاری داده‌ها ۲۶۵
۷-۸ کنترل برنامه ۲۷۲
۸-۸ کامپیوتر کم دستور RISC ۲۸۲
مسائل ۲۹۱

فصل ۹: پردازش خط لوله‌ای و برداری..... ۲۹۸
۱-۹ پردازش موازی ۲۹۸
۲-۹ خط لوله ۳۰۱
۳-۹ خط لوله حسابی ۳۰۶
۴-۹ خط لوله دستورالعمل ۳۰۹
۵-۹ خط لوله RISC ۳۱۴
۶-۹ پردازش برداری ۳۱۹
۷-۹ پردازشگر آرایه ۳۲۶
مسائل ۳۲۸

فصل ۱۰: معماری کامپیوتر ۳۳۱
۱-۱۰ مقدمه ۳۳۱
۲-۱۰ جمع و تفریق ۳۳۲
۳-۱۰ الگوریتم ضرب ۳۳۸
۴-۱۰ الگوریتم‌های تقسیم ۳۴۵
۵-۱۰ عمل های حسابی ممیز شناور ۳۵۲
۶-۱۰ واحد حساب دهدهی ۳۶۲
۷-۱۰ اعمال حسابی دهدهی ۳۶۸
مسائل ۳۷۴

www.homework.ir

فصل ۱۱ : سازمان ورودی - خروجی ۳۷۹

۱۱-۱ وسایل جانبی ۳۷۹
۱۱-۲ واسطه ورودی - خروجی ۳۸۳
۱۱-۳ انتقال غیرهمزمان داده ۳۹۰
۱۱-۴ شیوه‌های انتقال ۴۰۱
۱۱-۵ وقفه اولویت دار ۴۰۶
۱۱-۶ دستیابی مستقیم به حافظه (DMA) ۴۱۶
۱۱-۷ پردازنده ورودی و خروجی IOP ۴۲۰
۱۱-۸ تبادل اطلاعات سری ۴۳۰
مسائل ۴۴۱

فصل ۱۲ : سازمان حافظه ۴۴۵

۱۲-۱ سلسله مراتب حافظه ۴۴۵
۱۲-۲ حافظه اصلی ۴۴۸
۱۲-۳ حافظه کمکی ۴۵۴
۱۲-۴ حافظه تداعیگر ۴۵۷
۱۲-۵ حافظه کش ۴۶۳
۱۲-۶ حافظه مجازی ۴۷۱
۱۲-۷ سخت افزار مدیریت حافظه ۴۷۸
مسائل ۴۸۵

فصل ۱۳ : چند پردازنده‌ها ۴۸۹

۱۳-۱ مشخصات چند پردازنده‌ها ۴۸۹
۱۳-۲ ساختارهای اتصالات متقابل ۴۹۱
۱۳-۳ داوری بین پردازنده‌ها ۵۰۰
۱۳-۴ ارتباط و همگامی بین پردازنده‌ها ۵۰۷
۱۳-۵ همبستگی حافظه کش ۵۱۰
مسائل ۵۱۴

www.homework.ir

www.homework.ir مشاوره و انجام پروژه هاي دانشجويي
دانلود فصلهايي از كتب دانشگاهي

۱ مدارهای منطقی دیجیتال

۱-۱ کامپیوترهای دیجیتال
۱-۲ گیت‌های منطقی
۱-۳ جبر بول
۱-۴ ساده سازی با نقشه
۱-۵ مدارهای ترکیبی
۱-۶ فلیپ فلاپها
۱-۷ مدارهای ترتیبی

۱-۱ کامپیوترهای دیجیتال

کامپیوتر دیجیتال سیستمی دیجیتالی[1] (رقمی) است که انواع کارهای محاسباتی را انجام می‌دهد. کلمه دیجیتال بدان معنی است که اطلاعات در کامپیوتر توسط متغیرهایی که تعداد محدودی از مقادیر گسسته را بخود اختصاص می‌دهند نمایش داده می‌شوند. این مقادیر در داخل کامپیوتر بوسیله اجزائی که می‌توانند تعداد محدودی از حالت‌های گسسته را در خود حفظ کنند پردازش می‌شوند. مثلاً ارقام دهدهی 0، 1، 2، ... 9، ده مقدار گسسته را فراهم می‌آورند. اولین کامپیوترهای الکترونیک دیجیتال که در اواخر دهه ۱۹۴۰ ساخته شدند اساساً برای محاسبات عددی بکار می‌رفتند. در این مورد، عناصر گسسته ارقام هستند و در واقع اصطلاح کامپیوتر دیجیتال از این کاربرد گرفته شده است. در عمل، کامپیوترهای دیجیتال با قابلیت اطمینان بیشتری کار می‌کنند بشرطی که تنها از دو حالت استفاده شود. به دلیل محدودیت‌های فیزیکی اجزاء، و نیز بعلت گرایش انسان به منطق دودویی (یعنی بکارگیری

1- Digital

عبارات صحیح و غلط، بله یا خیر) قطعات دیجیتال که مقید به نگهداری مقادیر گسسته هستند مقید به نگهداری و حفظ فقط دو مقدار که دودویی [1] نامیده می‌شوند نیز می‌باشند.

کامپیوترهای دیجیتال از سیستم اعداد دودویی [2] استفاده می‌کنند که دو رقم بیشتر ندارد: 0 و 1. یک رقم دودویی بیت خوانده می‌شود. اطلاعات در کامپیوترهای دیجیتال بوسیله گروههایی از بیت‌ها نشان داده می‌شوند. با استفاده از تکنیک‌های کدگذاری، گروه‌های بیت‌ها نه تنها برای نمایش اعداد دودویی بلکه برای سایر سمبل‌های گسسته، همچون ارقام دهدهی یا حروف الفبا نیز بکار برده می‌شوند. با استفاده صحیح از مجموعه‌های دودویی و بکارگیری روشهای مختلف کدگذاری می‌توان گروه‌های بیت‌ها را برای ساخت مجموعه‌های کامل دستورالعمل [3]ها جهت انجام محاسبات مختلف بکار برد.

برخلاف اعداد دهدهی متداول که سیستم مبنای 10 را بکار می‌برند، اعداد دودویی سیستم مبنای 2 با دو رقم 0 و 1 را استفاده می‌نمایند. معادل دهدهی یک عدد دودویی را می‌توان با بسط آن به صورت یک سری از توان‌های 2 بدست آورد. مثلاً عدد دودویی 1001011 کمیتی را نشان می‌دهد که قابل تبدیل به یک عدد دهدهی است و از ضرب هر بیت در توان صحیحی از 2 بترتیب زیر بدست می‌آید:

$$1 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 75$$

هفت بیت 1001011 یک عدد دودویی را نشان می‌دهند که معادل دهدهی آن 75 است. با این وجود همین گروه هفت بیتی وقتی بعنوان کد دودویی حروف الفبا در نظر گرفته شود نمایشگر حرف K است. این گروه می‌تواند یک کد کنترلی را هم برای مشخص کردن برخی تصمیمات منطقی در کامپیوترهای دیجیتال نمایش دهد. به بیان دیگر، گروه بیت‌ها در یک کامپیوتر دیجیتال برای نمایش بسیاری از چیزهاست. این مطلب شبیه به مفهوم استفاده از حروف الفبا یکسان در ساخت زبانهای مختلف مانند انگلیسی و فرانسه است.

گاهی اوقات یک سیستم کامپیوتر به دو بخش عملیاتی با ماهیت نرم‌افزاری [4] و سخت افزاری [5] تقسیم می‌گردد. سخت افزار کامپیوتر شامل تمام قطعات الکترونیک و الکترومکانیک تشکیل دهنده ماهیت فیزیکی آن است. نرم‌افزار کامپیوتر از دستورالعمل‌ها و داده‌هایی [6] تشکیل شده که کامپیوتر برای انجام کارهای مختلف داده پردازی از آنها استفاده می‌نماید. رشته‌ای از دستورات، برنامه [7] خوانده می‌شود. داده‌هایی که برنامه با آنها کار می‌کند پایگاه داده‌ها [8] را تشکیل می‌دهد.

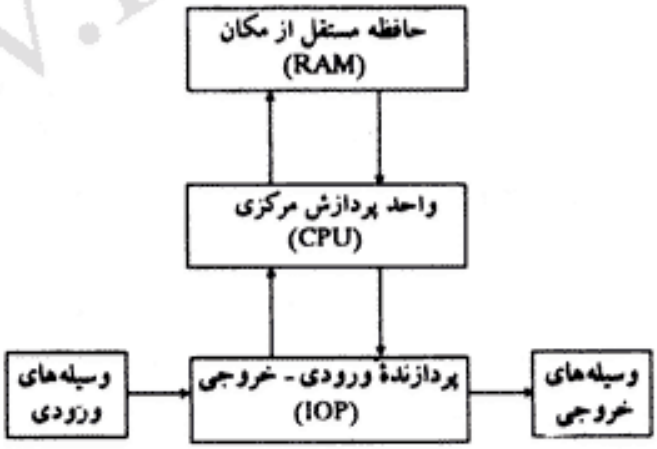
یک سیستم کامپیوتر از ترکیب سخت‌افزار، و نرم‌افزار سیستم برای آن تشکیل شده است. نرم‌افزار سیستم یک کامپیوتر گزیده‌ای از برنامه‌هاست که هدف از کاربرد آن بهره‌گیری موثرتر از کامپیوتر

1– Binary
2– Binary Number System
3– Instruction
4– Software
5– Hardware
6– Data
7– Program
8– Data Base

می‌باشد. برنامه‌هایی که بصورت یک بسته نرم‌افزاری سیستم درآمده‌اند سیستم عامل[1] خوانده می‌شوند. این برنامه‌ها از برنامه‌هایی که کاربر برای حل مسائل خاصی می‌نویسد متمایز است. بعنوان مثال، یک برنامه به زبان سطح بالاکه توسط کاربر برای حل یک داده‌پردازی خاص نوشته شده است یک برنامه کاربردی است ولی یک کامپایلر[2] که برنامه زبان سطح بالا را به برنامه‌ای به زبان ماشین تبدیل می‌کند یک برنامه سیستم است. خریداری که کامپیوتر را می‌خرد علاوه بر سخت‌افزار به انواع نرم‌افزارهای موجود برای بکارگیری موثرتر کامپیوتر هم نیاز دارد. نرم‌افزار سیستم بخش لاینفک یک سیستم کامپیوتری است. نقش آن جبران اختلافات موجود بین نیازهای کاربر و توانایی سخت‌افزار است.

سخت‌افزار کامپیوتر معمولاً به سه بخش عمده مطابق شکل ۱-۱ تقسیم می‌شود. واحد پردازش مرکزی[3] CPU حاوی یک واحد حساب و منطق[4] ALU برای کار روی داده‌ها، تعدادی ثبات برای ذخیره‌کردن داده‌ها، واحدهای کنترل برای برداشت[5] و اجرای[6] دستورالعمل‌هاست. حافظه یک کامپیوتر محل نگهداری دستورالعمل‌ها و داده‌هاست. این حافظه، حافظه با دستیابی تصادفی[7] خوانده می‌شود زیرا CPU قادر است به هر مکانی از حافظه بطور تصادفی دستیابی نموده و در یک فاصله زمانی ثابت اطلاعات دودویی را از آنجا بردارد. پردازنده[8] ورودی و خروجی (IOP) حاوی مدارات الکترونیک برای کنترل تبادل و انتقال اطلاعات بین کامپیوتر و دنیای خارج است. وسایل[9] ورودی- خروجی متصل به کامپیوتر عبارتند از صفحه کلیدها، چاپگرها، پایانه‌ها، راه‌اندازهای دیسک مغناطیسی و سایر وسایل تبادل اطلاعات و ارتباطی دیگر است.

این کتاب دانش ابتدایی لازم را برای درک عملیات سخت‌افزاری یک سیستم کامپیوتر فراهم



شکل ۱-۱ بلاک دیاگرام یک کامپیوتر دیجیتال

1- Operating System
2- Compiler
3- Central Processing Unit
4- Arithmetic Logic Unit
5- Fetch
6- Execute
7- Random Access Memory
8- Processor
9- Device

می‌سازد. گاهی اوقات موضوع، بسته به علاقه محقق، از سه دیدگاه مورد بررسی قرار می‌گیرد. بهنگام‌بحث درباره سخت‌افزار کامپیوتر لازم است تا بین سازمان کامپیوتر[1]، طراحی کامپیوتر[2] و معماری کامپیوتر[3] تمایز قایل شویم.

سازمان کامپیوتر به نحوه عملکرد اجزاء قطعات سخت‌افزاری و چگونگی اتصالات آنها در تشکیل سیستم کامپیوتر اشاره دارد. فرض براین است که قطعات مختلف در مکان‌های صحیح خود قرار دارند و آنچه لازم است انجام شود بررسی ساختار سازمانی است جهت تحقیق عملکرد کامپیوتر.

طراحی کامپیوتر به طراحی سخت‌افزاری کامپیوتر مربوط می‌شود. پس از تهیه مشخصات کامپیوتر، طراح باید سخت‌افزار سیستم را بوجود آورد. طراحی کامپیوتر به تعیین نوع سخت‌افزار مربوط می‌شود و نیز اینکه چگونه آنهابهم متصل گردند. این جنبه از سخت‌افزار کامپیوتر را گاهی پیاده سازی[4] کامپیوتر می‌خوانند.

معماری کامپیوتر به ساختار و رفتار کامپیوتر از دیدگاه کاربر مربوط می‌شود. این بحث شامل مطالبی درباره قالب اطلاعات، مجموعه دستورالعمل‌ها، و تکنیک‌های آدرس‌دهی حافظه است. طراحی معماری یک سیستم کامپیوتری درباره مشخصات انواع ماژول‌های[5] عملیاتی، مانند پردازنده‌ها و حافظه‌ها و ساختاردهی آنها برای تشکیل یک سیستم کامپیوتر بحث می‌کند.

این کتاب درباره هر سه نوع موضوع مربوط به سخت‌افزار کامپیوتر بحث می‌نماید. در فصل‌های ۱ الی ۴، ما انواع قطعات دیجیتال راکه در سازمان و طراحی کامپیوتر بکار رفته‌اند ارائه می‌نمائیم. فصل ۵ تا ۷ مراحلی راکه یک طراح باید در طراحی یک کامپیوتر دیجیتال ساده طی کند پوشش می‌دهد. فصل‌های ۸ و ۹ درباره معماری واحد پردازش مرکزی است. در فصل‌های ۱۱ و ۱۲ سازمان و معماری پردازنده ورودی و خروجی و واحد حافظه ارائه شده است.

۲-۱ گیت‌های منطقی

اطلاعات دودویی در کامپیوترهای دیجیتال با کمیت‌هایی فیزیکی که سیگنال نامیده می‌شود نمایش داده می‌شود. سیگنال‌های الکتریکی همچون ولتاژهای موجود در سرتاسر کامپیوتر در یکی از دو حالت قابل تشخیص از هم قرار دارند. این دو حالت نماینده یک متغیر دودویی هستند که می‌تواند برابر 1 یا 0 تصور شود. مثلاً یک کامپیوتر دیجیتال ممکن است از یک سیگنال 3 ولت برای نمایش 1 دودویی و 0.5 ولت برای نشان دادن 0 دودویی استفاده نماید. پایانه‌های ورودی مدارهای دیجیتال سیگنال‌های 3 و 0.5 را پذیرفته و پایانه‌های خروجی هم باسیگنالهای 3 و 0.5 به ورودی‌ها پاسخ می‌دهند. ورودی‌ها و خروجی‌های 3 و 0.5 بترتیب متناظر با مقادیر دودویی 1 و 0 می‌باشند.

منطق دودویی با متغیرهای دودویی و نیز با اعمالی که مفهوم منطقی دارند سروکار دارد. از این منطق برای توصیف عملیات و یا پردازش اطلاعات دودویی بصورت عبارات جبری و یا جداول

1- Computer Organization 2- Computer Design 3- Computer Architecture
4- Implementation 5- Madule

استفاده می‌شود. دستکاری اطلاعات دودویی توسط مدارهای منطقی بنام گیت [1] انجام می‌شود. گیت‌ها بلاک‌های سخت‌افزاری هستند که سیگنال‌های دودویی 1 یا 0 را در خروجی، بسته به شرایط ورودی‌هایشان تولید می‌کنند. امروزه انواع گیت‌های منطقی در سیستم‌های کامپیوتر دیجیتال بکار برده می‌شوند. هر گیت سمبل گرافیک خاص خود را داشته و عملکردش توسط یک عبارت جبری نمایش داده می‌شود. رابطه ورودی و خروجی برای متغیرهای دودویی برای هر گیت را می‌توان توسط یک جدول بنام جدول درستی [2] نشان داد.

نام، سمبل گرافیکی، توابع جبری و جداول درستی هشت گیت منطقی در شکل ۲-۱ لیست شده‌اند. هر گیت دارای یک یا دو متغیر ورودی است که با A و B و یک خروجی که با x مشخص شده است. گیت AND تابع منطقی AND را تولید می‌کند، یعنی خروجی آن 1 است اگر هر دو ورودی A و B برابر 1 باشد؛ در غیر اینصورت خروجی آن 0 است. این شرایط برای گیت AND بصورت جدول مشخص شده است. جدول همچنین نشان می‌دهد که x برابر 1 است اگر هر دو ورودی A و B برابر 1 باشند. سمبل جبری عملکرد تابع AND شبیه سمبل ضرب در حساب معمولی است. ما می‌توانیم برای نمایش عمل از یک نقطه در بین دومتغیر و یا بدون هر نوع علامتی در بین آنها استفاده کنیم. گیت‌های AND ممکن است بیش از دو ورودی داشته باشند و بنا به تعریف خروجی آنها هنگامی 1 است که تمام ورودی‌ها 1 باشند.

گیت OR تابع OR غیر انحصاری را تولید می‌کند، یعنی خروجی 1 است اگر ورودی A یا B یا هر دو 1 باشند؛ در غیر اینصورت خروجی 0 است. سمبل جبری تابع OR علامت + است که شبیه علامت جمع حسابی است. گیت‌های OR ممکن است بیش از دو ورودی داشته باشند، و بنا به تعریف، خروجی هنگامی 1 است که هر یک از ورودی‌ها 1 باشند.

مدار معکوس کننده [3] وضعیت منطقی هر سیگنال دودویی را معکوس می‌کند. این مدار تابع NOT یا متمم را تولید می‌نماید. سمبل جبری مورد استفاده برای متمم منطقی، علامت پریم یا خط روی سمبل متغیر مورد نظر است. در این کتاب ما از علامت پریم برای متمم منطقی یک متغیر دودویی استفاده می‌کنیم و خط روی حرف را برای مشخص کردن زیر عمل متمم که در فصل ۴ تعریف خواهد شد نگه میداریم. دایره کوچک در خروجی سمبل گرافیکی معکوس کننده، مشخص کننده متمم منطقی است.

یک سمبل مثلثی شکل، مربوط به یک بافر [4] (میانگیر) است. یک بافر هیچگونه تابع منطقی را تولید نمی‌کند چون مقدار دودویی خروجی آن برابر با مقدار دودویی ورودی است. این مدار صرفاً به منظور تقویت توان مورد استفاده قرار می‌گیرد. مثلاً، بافری که ولتاژ 3 ولت را برای عدد دودویی 1 مورد استفاده قرار می‌دهد. هرگاه در ورودی 3 ولت را دریافت کند خروجی 3 ولت را تولید خواهد کرد. با این وجود مقدار توان الکتریکی لازم در ورودی بافر بسیار کمتر از توان تولید شده در خروجی آن است. هدف اصلی از بکارگیری بافر راه‌اندازی سایر گیت‌هاست که توان قابل ملاحظه‌ای را نیاز دارند.

1- Gate 2- Truth table 3- Inverter
4- Buffer

نام	سمبل گرافیکی	تابع جبری	جدول درستی
AND		$x=A\cdot B$ یا $x=AB$	A B | x 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1
OR		$x=A+B$	A B | x 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1
معکوس‌کننده		$x=A'$	A | x 0 | 1 1 | 0
میانگیر		$x=A$	A | x 0 | 0 1 | 1
NAND		$x=(AB)'$	A B | x 0 0 | 1 0 1 | 1 1 0 | 1 1 1 | 0
NOR		$x=(A+B)'$	A B | x 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 0
یای انحصاری (XOR)		$x=A\oplus B$ یا $x=A'B+AB'$	A B | x 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0
NOR انحصاری یا تابع برابری		$x=(A\oplus B)'$ یا $x=A'B'+AB$	A B | x 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 1

شکل ۲-۱ گیت‌های منطقی دیجیتال

تابع NAND متمم تابع AND است و همانطور که از سمبل گرافیکی آن مشخص است از یک سمبل گرافیکی AND که بدنبال آن دایره کوچکی تشکیل شده است. کلمه NAND از اختصار NOT–AND بدست آمده است. گیت NOR متمم OR است و سمبل گرافیکی OR که بدنبال آن دایره کوچکی آمده است را بکار می‌برد. هر دو گیت NAND و NOR ممکن است بیش از دو ورودی داشته باشند، و خروجی آنها همیشه بترتیب متمم توابع AND و OR است.

گیت OR انحصاری[1] دارای سمبل گرافیکی شبیه به گیت OR است جز اینکه یک خط منحنی اضافی در سمت ورودی آن وجود دارد. خروجی این گیت هنگامی 1 است که فقط یکی از ورودی‌ها 1 باشد ولی نباید هر دوی آنها تواماً یک باشند. تابع OR انحصاری (XOR یا EOR) می‌تواند توسط سمبل گرافیکی خاص خودش و یا برحسب گیت‌های AND و OR و متمم طبق شکل ۲-۱ نشان داده شود. NOR انحصاری[2]، همانطور که از سمبل گرافیکی آن مشخص است متمم OR انحصاری است. خروجی این گیت فقط هنگامی 1 است که هر دو ورودی 1 و یا هر دو ورودی 0 باشند. نام مناسبتری برای گیت OR انحصاری تابع فرد[3] است، یعنی خروجی 1 است اگر تعداد فردی از ورودی‌ها 1 باشند. بنابراین در یک تابع OR انحصاری سه ورودی، خروجی هنگامی 1 است که فقط یکی از ورودی‌ها 1 و یا هر سه ورودی 1 باشد. گیت‌های EOR و ENOR معمولاً دو ورودی هستند و به ندرت با سه ورودی یا بیشتر یافت می‌شوند.

۳-۱ جبر بول

جبر بول جبری است که با متغیرهای دودویی و اعمال منطقی سروکار دارد. متغیرها با حروفی همچون A و B و x و y مشخص می‌شوند. سه عمل اصلی منطقی عبارتند از AND، OR و متمم. یک تابع بول را می‌توان با استفاده از متغیرهای دودویی، سمبل‌های عملیات منطقی، پرانتزها و یک علامت تساوی نشان داد. تابع بول برای هر مقدار مفروض از متغیرها می‌تواند 1 یا 0 باشد. بعنوان مثال تابع بول زیر را در نظر بگیرید

$$F = x + y'z$$

تابع F برابر با 1 است اگر x برابر 1 و یا هر دو y' و z برابر 1 باشند؛ در غیر اینصورت F برابر 0 است. ولی اینکه می‌گوییم $y' = 1$ معادل است با اینکه بگوییم $y = 0$، چون y' متمم y است. بنابراین ما ممکن است بگوییم که F هنگامی 1 است که $x = 1$ یا اگر $yz = 01$ باشد. رابطه بین یک تابع و متغیرهای دودویی‌اش را با یک جدول درستی می‌توان نشان داد. برای نمایش یک تابع در یک جدول درستی، لیستی از 2^n ترکیب از n متغیر دودویی نیاز داریم. همانطور که در جدول شکل ۳-۱ (الف) دیده می‌شود

1– Exclusive OR (XOR, EOR)

2– Exclusive NOR (ENOR) (Equivalence)

3– Odd Function

هشت ترکیب جداگانه برای 3 متغیر x و y و z وجود دارد. تابع F برای ترکیب‌هایی 1 است که در آنها $x = 1$، $yz = 01$ باشد؛ این تابع برای سایر ترکیب‌ها صفر است.

یک تابع بول را می‌توان از یک عبارت جبری به یک دیاگرام منطقی متشکل از گیت‌های AND، OR و یا معکوس کننده تبدیل کرد. دیاگرام منطقی F در شکل ۳-۱ (ب) نشان داده شده است برای ورودی y، معکوس کننده‌ای وجود دارد که متمم y′ را تولید می‌نماید. برای جمله y′z یک گیت ANDوجود دارد و از یک گیت OR برای ترکیب دو جمله استفاده شده است. در یک دیاگرام منطقی، متغیرهای تابع ورودی‌های مدار و سمبل تابع، خروجی آن را تشکیل می‌دهد.

هدف از جبر بول تسهیل تحلیل و طراحی مدارهای دیجیتال است. این جبر ابزار مناسبی برای اعمال زیر نیز هست.

۱- بیان رابطه جبری بین متغیرها بصورت یک جدول درستی

۲- بیان رابطه‌ای بین ورودی- خروجی یک دیاگرام منطقی بصورت عبارت جبری

۳- یافتن مدار ساده‌تر برای تابع

نمایش یک جدول درستی بصورت یک عبارت جبری به طرق مختلف امکان پذیر است. برطبق قوانین حاکم در جبر بول، با دستکاری یک عبارت بولی می‌توان فرم ساده‌تری که گیت‌های کمتری نیاز دارد را برای آن بدست آورد. برای دیدن چگونگی انجام این اعمال، ابتدا توانایی‌های دستکاری جبر بول را می‌بینیم.

جدول ۱-۱ ابتدایی‌ترین اتحادهای[1] جبر بول را نشان می‌دهد. تمام اتحادهای جدول بوسیله جدول درستی قابل اثباتند. هشت اتحاد اول رابطه اساسی بین یک متغیر و خودش، یا با ثابتهای 1 و 0 را نشان می‌دهد. پنج اتحاد بعدی (از معمولی 9 تا 13) شبیه به روابط در جبر معمولی هستند. اتحاد 14 در جبر معمولی صادق نیست ولی در دستکاری[2] عبارات بولی مفید است. اتحادهای 15 و 16 تئوری‌های دومورگان نامیده می‌شوند و در زیر بحث شده‌اند. آخرین اتحاد بیان می‌دارد که اگر متغیری دوبار متمم شود مقدار اولیه آن بدست می‌آید.

z	y	z	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

(الف) جدول درستی



(ب) دیاگرام منطقی

شکل ۳-۱ جدول درستی و دیاگرام منطقی برای $F = x + y'z$.

1- Indentity 2- Manipulation

جدول ۱-۱ اتحادهای اساسی جبر بول

(1) $x + 0 = x$	(2) $x \cdot 0 = 0$
(3) $x + 1 = 1$	(4) $x \cdot 1 = x$
(5) $x + x = x$	(6) $x \cdot x = x$
(7) $x + x' = 1$	(8) $x \cdot x' = 0$
(9) $x + y = y + x$	(10) $xy = yx$
(11) $x + (y + z) = (x + y) + z$	(12) $x(yz) = (xy)z$
(13) $x(y + z) = xy + xz$	(14) $x + yz = (x + y)(x + z)$
(15) $(x + y)' = x'y'$	(16) $(xy)' = x' + y'$
(17) $(x')' = x$	

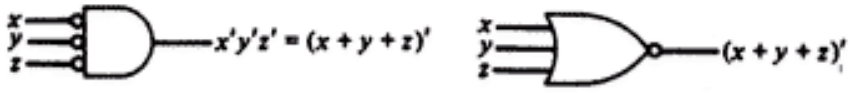
اتحادهایی که در جدول لیست شده‌اند هم در مورد متغیرهای تکی و هم در مورد توابع بولی که برحسب مجموعه‌ای از متغیرهای دودویی بیان شده‌اند صادق‌اند. مثلاً عبارت جبری بولی زیر را در نظر بگیرید:

$$AB' + C'D + AB' + C'D$$

با فرض $x = AB' + C'B$ رابطه فوق برابرست با $x + x$. با استفاده از اتحاد 5 در جدول ۱-۱ می‌بینیم که $x + x = x$ است. بنابراین عبارت فوق بصورت زیر کاهش می‌یابد

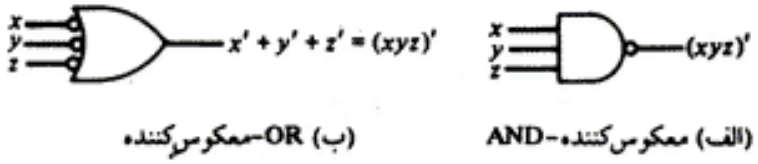
$$AB' + C'D + AB' + C'D = AB' + C'D$$

تئوری دومورگان در کار با گیت‌های NOR و NAND بسیار با اهمیت است. این تئوری بیان می‌کند که یک گیت NOR که عمل $(x + y)'$ را انجام می‌دهد معادل با $x'y'$ است. بطور مشابه یک تابع NAND می‌تواند به یکی از دو صورت $(xy)'$ یا $(x' + y')$ بیان شود. باین دلیل همانطور که در شکل‌های ۴-۱ و ۵-۱ دیده می‌شود گیت‌های NOR و NAND دو سمبل نمایشی متمایز دارند. بعوض نمایش گیت NOR با یک OR و دایره کوچکی بدنبال آن، می‌توان آن را با یک گیت AND با دایره‌های کوچکی در تمام ورودی‌هایش نشان داد. سمبل معکوس‌کننده-AND برای گیت NOR از تئوری دومورگان حاصل می‌شود و در آن دایره کوچک به معنی متمم سازی است. بطور مشابه، گیت NAND دارای دو سمبل متمایز است، شکل ۵-۱.



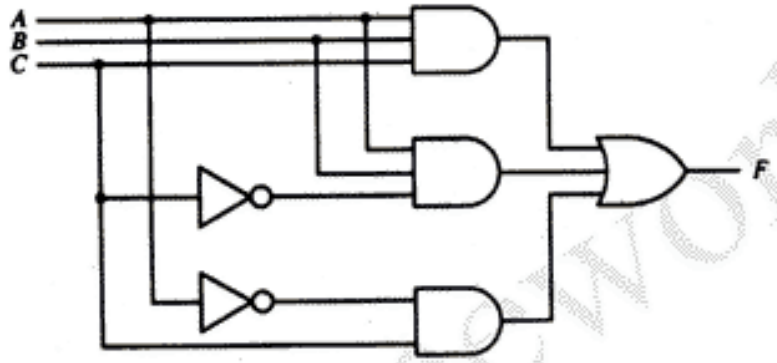
(ب) AND-معکوس‌کننده (الف) معکوس‌کننده-OR

شکل ۴-۱ دو سمبل گرافیکی برای گیت NOR.

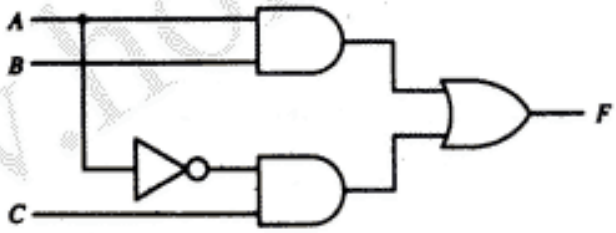


(ب) OR-معکوس‌کننده (الف) معکوس‌کننده-AND

شکل ۵-۱ دو سمبل گرافیکی برای گیت NAND.



$F = ABC + ABC' + A'C$ (الف)



$F = AB + A'C$ (ب)

شکل ۶-۱ دو دیاگرام منطقی برای یک تابع بولی.

برای دیدن چگونگی بکارگیری دستکاری جبر بول در ساده کردن مدارهای دیجیتال، دیاگرام منطقی شکل ۶-۱ (الف) را ملاحظه کنید. خروجی مدار می‌تواند بصورت عبارت جبری زیر بیان شود

$$F = ABC + ABC' + A'C$$

هر جمله متعلق به یک گیت AND است و گیت OR جمع منطقی سه جمله را تشکیل می‌دهد. برای متمم‌های A' و C' دو معکوس کننده[1] لازم است. عبارت فوق می‌تواند بصورت زیر ساده شود.

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

1– Inverter

توجه کنید طبق رابطه 7 که در جدول ۱-۱ $1 = (C + C')$ و به موجب اتحاد 4، $AB \cdot 1 = AB$ است.

دیاگرام منطقی عبارت ساده شده در شکل ۶-۱ (ب) کشیده شده است. این دیاگرام تنها به چهار گیت در عوض شش گیت شکل ۶-۱ (الف) نیاز دارد. دو مدار معادل بوده و جدول درستی یکسانی را برای ورودی‌های A، B و C و خروجی F فراهم می‌آورند.

متمم یک تابع

متمم یک تابع مانند F را، وقتی که بصورت جدول درستی نشان داده شده باشد، می‌توان از تعویض 1 ها به 0 ها بدست آورد. بهنگام نمایش تابع بصورت عبارت جبری، متمم تابع با توجه بـه تـئوری دُمورگان حاصل می‌شود. فرم کلی تئوری دُمورگان بصورت زیر نوشته می‌شود

$$(x_1 + x_2 + x_3 + \cdots + x_n)' = x_1' x_2' x_3' \cdots x_n'$$
$$(x_1 x_2 x_3 \cdots x_n)' = x_1' + x_2' + x_3' + \cdots + x_n'$$

از تئوری کلی دمورگان می‌توانیم رویه ساده‌ای برای بدست آوردن متمم یک تابع نتیجه بگیریم. این کار با تبدیل تمام گیت‌های AND به OR و تمام OR ها به AND و سپس متمم سازی هر متغیر بتنهایی، حاصل می‌شود. بعنوان مثال رابطه زیر و متمم آن را در نظر بگیرید

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

عبارت متمم شده از تعویض اعمال AND و OR و متمم کردن هر متغیر بتنهایی، حاصل گشته است. توجه کنید که C' متمم C است.

۱-۴ ساده سازی با نقشه

پیچیدگی دیاگرام منطقی که یک تابع بول را پیاده سازی می‌کند مستقیماً به پیچیدگی عبارت جبری که تابع از روی آن پیاده سازی می‌شود بستگی دارد. هرچند نمایش هر تابع بصورت جدول درستی تنها به یک شکل ممکن است ولی همان تابع می‌تواند بصور مختلف جبری نشان داده شود. عبارت تابع را می‌توان با استفاده از روابط اصلی جبر بول ساده کرد. با این وجود، این رویه گاهی اوقات مشکل است زیرا مراحل متوالی دستکاری تابع بعلت عدم کفایت قوانین قابل پیش بینی نیستند. روش نقشه (جدول) رویه ساده و مستقیمی را برای ساده کردن عبارت بول در اختیار می‌گذارد. ایـن روش را مـی‌توان یک نمایش تصویری از جدول درستی تصور نمود که امکان تفسیری آسان و انتخاب جملات می‌نیمم برای بیان جبری تابع را فراهم می‌آورد. روش نقشه را روش کارنو یا روش نقشه K نیز می‌نامند.

هر ترکیبی از متغیرها در یک جدول درستی مینترم نام دارد. مثلاً، جدول درستی شکل ۳-۱ هشت مینترم دارد. هنگامی که تابع n متغیره بوسیله نقشه کارنو نشان داده شود دارای 2^n مینترم خواهد بود که

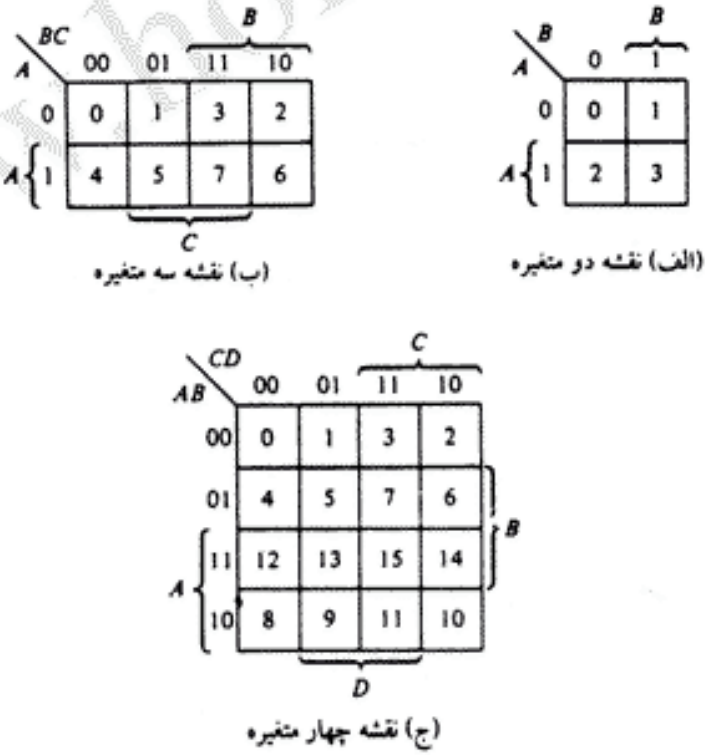
معادل 2^n عدد دودویی حاصل از n بیت است. تابع بولی به ازاء بعضی از مینترم‌ها برابر 1 و برخی دیگر 0 است. با نوشتن معادل دهدهی مینترم‌هایی که مقدار تابع را 1 می‌کنند می‌توان اطلاعات موجود در جدول درستی را بشکل فشرده‌ای درآورد. مثلاً جدول درستی شکل ۳-۱ بصورت زیر بیان می‌شود

$$F = (x, y, z) = \sum (1, 4, 5, 6, 7)$$

حروف داخل پرانتز متغیرهای دودویی را بترتیبی که در جملات مینترم ظاهر می‌شوند نشان می‌دهند. سمبل $\sum$ نشاندهنده مجموع مینترم‌هایی است که پس از آن در پرانتز آمده‌اند. مینترم‌هایی که مقدار 1 را برای تابع تولید می‌کنند با معادل دهدهی‌شان در پرانتز آمده‌اند. مینترم‌هایی که در پرانتز وجود ندارند مقدار 0 را برای تابع ایجاد می‌کنند.

نقشه کارنو دیاگرامی است متشکل از تعدادی مربع که هر مربع نشاندهنده یک مینترم است. در مربعات مربوط به مینترم‌هایی که عدد 1 را برای تابع تولید کنند عدد 1 و در بقیه 0 یا چیزی نمی‌نویسند. با تشخیصی الگوهای مختلف و ترکیب مربع‌هایی که در نقشه با 1 ها مشخص شده‌اند می‌توان عبارات جبری دیگری برای تابع بدست آورد و نهایتاً از میان آنها مناسبترین را برگزید.

نقشه‌های توابع دو، سه و چهار متغیره در شکل ۷-۱ نشان داده شده‌اند. تعداد مربع‌های تشکیل دهنده نقشه n متغیره 2^n است. 2^n مینترم، بخاطر سهولت ارجاع به آنها با معادل‌های دهدهی‌شان



شکل ۷-۱ نقشه برای توابع دو، سه و چهار متغیره

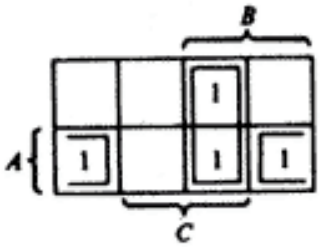
مشخص می‌شوند. شماره‌های مینترم‌ها با ترتیب خاصی به خانه‌ها تخصیص داده شده‌اند بطوری که مینترم‌های مربوط به دو مربع مجاور فقط در یک متغیر با هم اختلاف دارند. نام متغیرها در دو طرف یک خط مورب در گوشه نقشه نوشته می‌شوند. 0 ها و 1 های نوشته شده در کنار سطرها و ستون‌ها مقدار متغیرها را مشخص می‌کنند. در زیر آکلاد مربوط به هر متغیر، نیمی از خانه‌های نقشه مشخص شده که درآنها متغیر مربوطه بدون پریم است. در بقیه خانه‌های نقشه متغیر دارای پریم (متمم) می‌باشد. مینترم متناظر با هر مربع از روی اعداد دودویی مربوط به متغیرها که در طول سمت چپ و بالای نقشه نوشته شده‌اند تعیین می‌شود. مثلاً مینترم 5 در نقشه سه متغیره در سیستم دودویی 101 است، که می‌تواند از کنار هم قرارگرفتن 1 در سطر دوم و 01 از ستون دوم بدست آید. این مینترم مقداری را برای متغیرهای A و B و C مشخص می‌کند که در آن A و C بدون پریم و B پریم‌دار است (یعنی AB′C). از طرف دیگر مینترم 5 در یک نقشه چهار متغیره دارای چهار متغیر می‌باشد. عدد دودویی برای چهار بیت 0101 می‌باشد و مینترم مربوطه هم A′BC′D می‌باشد.

مینترم‌های مربعات مجاور در نقشه بجز در یک متغیر یکسانند. این متغیر در یک مربع نسبت به مربع مجاور متمم است. براساس این تعریف از مجاورت، مربعات انتهائی در سطرها نیز مجاور تصور خواهند شد. موضوع برای مربعات انتهایی در ستون‌ها نیز صادق است. در نتیجه، چهار مربع در گوشه‌های نقشه نیز مجاور خواهند بود. یک تابع بول که بوسیله جدول درستی‌اش نمایش داده شده است به یک نقشه منتقل می‌گردد. بدین ترتیب که در مربع‌هایی‌که تابع در آنها 1 است عدد 1 نوشته می‌شود. مربع‌های مجاوری که در آنها 1 وجود دارد بصورت گروهی باهم ترکیب می‌شوند. تعداد مربع‌های هر گروه باید توان صحیحی از 2 باشد. هر گروه می‌تواند در یک یا چند مربع با یک یا چند گروه دیگر اشتراک داشته باشد. هر گروه از مربعات یک عبارت جبری را نشان می‌دهد، و OR این جملات، عبارت ساده شده جبری را برای تابع فراهم می‌آورد. مثالهای زیر کاربرد نقشه را در ساده سازی تابع بول نشان می‌دهند.

در مثال اول تابع تولی زیر را ساده می‌کنیم:

$$F(A, B, C) = \sum (3, 4, 6, 7)$$

نقشه سه متغیره این تابع در شکل ۸-۱ نشان داده شده است. در این نقشه چهار مربع با 1 مشخص شده‌اند که هر کدام متعلق به یکی از مینترم‌هایی است که برای تابع، 1 را ایجاد می‌نمایند. این مربع‌ها متعلق به مینترم‌های 3، 4، 6 و 7 بوده و از شکل ۷-۱ (ب) تشخیص داده می‌شوند. در ستون سوم دو



شکل ۸-۱ نقشه برای $F(A, B, C) = \Sigma (3, 4, 6, 7)$

مربع مجاور با هم‌ترکیب می‌شوند. این ستون متعلق به B و C بوده و جمله BC را تولید می‌کنند. دو مربع باقیمانده حاوی 1 در دو گوشه سطر دوم، مجاورند و متعلق به سطر A و دو ستون C′ می‌باشند. بنابراین این دو جمله AC′ را تولید می‌کنند. عبارت جبری ساده شده برای تابع عبارتست از OR دو جمله:

$$F = BC + AC'$$

در مثال دوم تابع بولی زیر را ساده می‌کنیم:

$$F(A, B, C) = \sum (0, 2, 4, 5, 6)$$

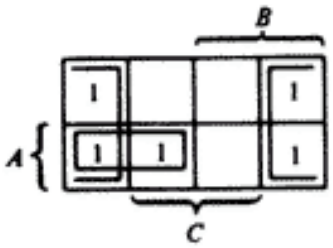
پنج مینترم در مربع‌های مربوطه خود در نقشه سه متغیره در شکل ۹-۱ با 1 علامت زده شده‌اند. چهار مربع در اولین و چهارمین ستون مجاورند و جملهٔ C′ را بدست می‌دهند. تنها مربع باقیمانده که با 1 مشخص شده متعلق به مینترم 5 بوده و قابل ترکیب با مربع مینترم 4 می‌باشد که جمله AB′ از آن نتیجه می‌شود. تابع ساده شده عبارتست از:

$$F = C' + AB'$$

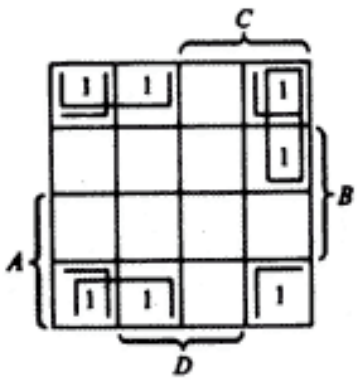
مثال سوم نیاز به یک نقشه چهار متغیره دارد

$$F(A; B, C, D) = \sum (0, 1, 2, 6, 8, 9, 10)$$

بخشی از نقشه که توسط این تابع چهار متغیره پوشش یافته است مربع‌هایی است که در شکل ۱۰-۱ با 1 مشخص شده‌اند. تابع دارای 1 هایی در چهار گوشه نقشه است که اگر بصورت یک گروه با هم ترکیب شوند



شکل ۹-۱ نقشه برای
$F(A, B, C) = \Sigma (0, 2, 4, 5, 6)$



شکل ۱۰-۱ نقشه برای
$F(A, B, C, D) = \Sigma (0, 1, 2, 6, 8, 9, 10)$

جمله B′D′ را تولید می‌کنند. دلیل ترکیب این مربعات مجاور بودن آنهابعلت مجاور بودن لبه‌های بالا و پایین و یا چپ و راست است که جمله B′C′ را بدست می‌دهند. تنها 1 باقیمانده در مربع مربوط به مینترم 6 است که با مینترم 2 ترکیب می‌شود و جمله A′CB′ را می‌سازد. تابع ساده شده عبارتست از

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

ساده سازی با ضرب حاصل جمع‌ها

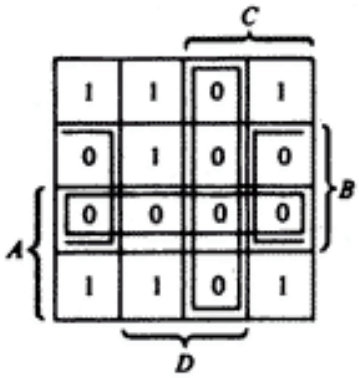
در تمام مثال‌های قبلی توابع بول حاصل از نقشه‌ها به فرم جمع حاصلضرب‌ها بیان شده بودند. جملات ضرب در واقع AND جملات و جمع آنهابمعنی OR این جملات است. گاهی هم مناسبتر است تا یک عبارت جبری را بصورت ضرب حاصل‌جمع‌ها بدست آوریم. مجموع، همان OR جملات و حاصلضرب نیز AND آنها هستند. با کمی تصحیح، ضرب حاصل‌جمع‌ها را از نقشه می‌توان بدست آورد.

روش بدست آوردن عبارت ضرب حاصل جمع‌ها، از خصوصیات اساسی جبر بول حاصل می‌گردد. 1 ها در نقشه نشاندهنده مینترم‌هایی است که برای تابع تولید 1 می‌نمایند. مربع‌هایی که با 1 پر نشده‌اند، برای تابع 0 تولید می‌کنند. اگر ما مربع‌های خالی را با 0 پر کنیم و سپس آنها را با هم بر اساس مربع‌های مجاور ترکیب نمائیم، متمم تابع را خواهیم داشت. با متمم نمودن F′ ، عبارتی برای F بصورت ضرب حاصل‌جمع‌ها بدست می‌آید. بهترین روش برای درک بهتر ارائه یک مثال است. می‌خواهیم تابع بولی زیر را به هر دو روش جمع حاصلضرب‌ها و ضرب حاصل‌جمع‌ها ساده کنیم.

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

1 های موجود در نقشه شکل ۱۱-۱ نشان دهنده تمام مینترم‌هایی است که 1 را برای تابع تولید می‌کنند. مربع‌هایی که با 0 مشخص شده‌اند نشان دهنده مینترم‌هایی هستند که در F وجود ندارند و بنابراین متمم F می‌باشند. ترکیب مربع‌های 1، فرم ساده شده تابع را بصورت مجموع حاصلضرب‌ها فراهم می‌کند:

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



شکل ۱۱-۱ نقشه برای
$F(A, B, C, D) = \Sigma (0, 1, 2, 5, 8, 9, 10)$

اگر مربع‌های 0 ترکیب شوند، طبق شکل، متمم ساده شده تابع بدست می‌آیند.

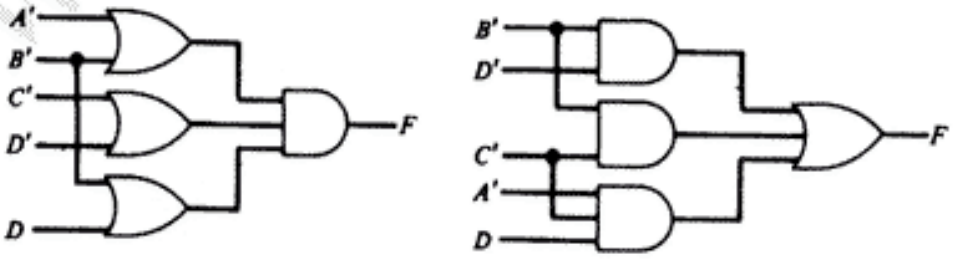
$$F' = AB + CD + BD'$$

با متمم‌گیری از F' فرم ساده شده تابع ضرب حاصل جمع‌ها بدست می‌آید:

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

دیاگرام‌های منطقی دو عبارت ساده شده در شکل ۱۲-۱ نشان داده شده‌اند. عبارت جمع حاصلضرب با تعدادی گیت AND در شکل ۱۲-۱ (الف) پیاده سازی[1] شده است که هر عبارت مربوط به یک گیت AND می‌باشد. خروجی‌های گیت‌های AND به ورودی‌های یک گیت OR متصل است. همان تابع بکمک گیت‌های OR بصورت ضرب حاصل‌جمع‌ها در شکل ۱۲-۱ (ب) پیاده‌سازی شده است. خروجی‌های گیت‌های OR به ورودی‌های گیت AND متصل شده‌اند و هر گیت OR نیز متناظر با یک جمله OR است. در هر یک از دو فرم فرض شده است که متغیرهای ورودی مستقیماً بصورت متمم نیز موجود باشند، بنابراین معکوس کننده‌ای در مدار بکار نرفته است. الگوی بکار رفته در شکل ۱۲-۱ فرم کلی پیاده سازی هر تابع بولی است که به یکی از شکل‌های استاندارد بیان شده باشد. در فرم جمع حاصلضرب‌ها تعدادی گیت AND به یک گیت OR متصل می‌شوند. در نوع ضرب حاصلجمع‌ها تعدادی گیت OR به یک گیت AND وصل می‌گردند.

همانطور که در شکل ۱۳-۱ (الف) نشان داده شده است جمع حاصلضرب‌ها می‌تواند با گیت‌های NAND پیاده سازی شود. توجه کنید که دومین گیت NAND با استفاده از سمبل گرافیکی شکل ۵-۱(ب) رسم شده است. دوایر کوچکی در دو انتهای سه خط از این دیاگرام دیده می‌شوند. دو دایره در یک خط به معنی دوبار متمم شدن می‌باشد، و چون $(x')' = x$ می‌باشد دو دایره را می‌توان حذف



(ب) ضرب حاصل جمع‌ها

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

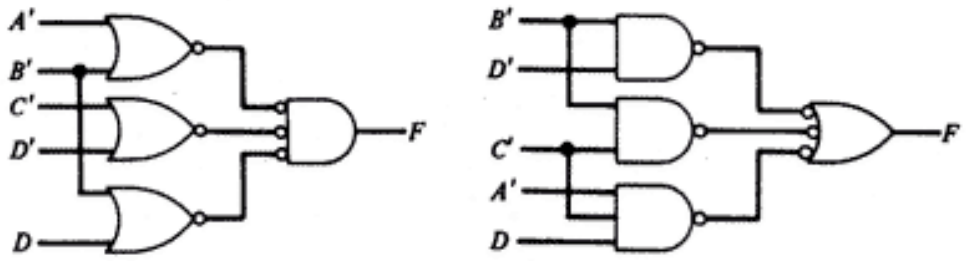
(الف) جمع حاصلضرب‌ها

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

شکل ۱۲-۱ دیاگرام‌های منطقی با گیت‌های AND و OR

1- Implementation

کرد و دیاگرام نتیجه معادل با مدار ۱۲-۱ (الف) خواهد بود. بطور مشابه عبارت ضرب حاصل‌جمع‌ها توسط گیت‌های NOR طبق شکل ۱۳-۱ (ب) پیاده سازی می‌شود. گیت NOR دوم با استفاده از سمبل گرافیکی شکل ۴-۱ (ب) رسم شده است. مجدداً دو دایره در دو انتهای هر خط حذف می‌شوند، و دیاگرام حاصله معادل شکل ۱۲-۱ (ب) خواهد بود.



(ب) با گیتهای NOR (الف) با گیتهای NAND

شکل ۱۳-۱ دیاگرام‌های منطقی با گیت‌های NAND و NOR

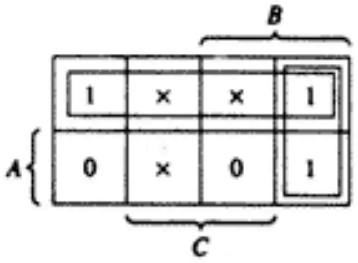
حالات بی‌اهمیت

1 ها و 0 ها در نقشه مینترم‌هایی را نشان می‌دهند که بترتیب تابع را 1 یا 0 می‌کنند. گاهی مهم نیست که تابع به ازاء یک مینترم معین، 0 ایجاد کند و یا 1. چون تابع ممکن است تابع 0 یا 1 باشد، می‌گوییم اهمیتی ندارد که خروجی تابع برای این مینترم چیست. مینترم‌هایی که 0 یا 1 را برای یک تابع تولید می‌نمایند حالات بی‌اهمیت خوانده شده و با علامت X در نقشه مشخص می‌شوند. این حالات بی‌اهمیت را می‌توان برای ساده‌تر کردن عبارت جبری بکار برد.

وقتی که مربع‌های مجاور برای تابع در نقشه انتخاب می‌شوند، X ها ممکن است 0 یا 1 فرض شوند، بسته به اینکه کدام یک عبارت ساده‌تری را فراهم آورند. بعلاوه اگر یک X به ساده شدن تابع کمک نکند لزومی به استفاده از آن نیست. در هر حال، انتخابی که بعمل می‌آید فقط به حد ساده سازی حاصل بستگی دارد. بعنوان مثال، تابع بولی زیر را با جملات بی‌اهمیت آن در نظر بگیرید.

$$F(A, B, C) = \sum (0, 2, 6)$$
$$d(A, B, C) = \sum (1, 3, 5)$$

مینترم‌هایی که در F آمده‌اند مقدار 1 را برای تابع تولید می‌کنند. مینترم‌های بی‌اهمیت که در d آمده‌اند هر یک از دو مقدار 0 یا 1 را می‌توانند تولید نمایند. باقیمانده مینترم‌ها یعنی مینترم‌های 4 و 7 مقدار 0 را برای تابع تولید می‌کنند. نقشه در شکل ۱۴-۱ نشان داده شده است. مینترم‌های F با 1، مینترم‌های d با X، و مربع‌های باقیمانده با 0 مشخص شده‌اند. 1 ها و X ها به هر روش مناسبی با هم ترکیب می‌شوند



شکل ۱۴-۱ مثالی از نقشه با حالات بی‌اهمیت

تا حداکثر تعداد مربع‌های مجاور را در بربگیرند. لزومی ندارد که تمام یا هر یک از X ها را در این ترکیب‌ها در نظر بگیریم ولی تمام 1 ها باید در نظر گرفته شوند. با جایگزینی X ها در مینترم‌های 1، 3 با 1 ها، جمله A′ را از سطر اول بدست می‌آوریم. 1 باقیمانده برای مینترم 6 هم با مینترم 2 ترکیب شده و از آن جمله BC′ بدست می‌آید. عبارت ساده شده بصورت زیر است:

$$F = A' + BC'$$

توجه کنید که مینترم بی‌اهمیت 5 در ترکیب گنجانده نشد زیرا به ساده شدن عبارت کمکی نمی‌نماید. همچنین اگر مینترم‌های بی‌اهمیت 1 و 3 با 1 جایگزین نشده بودند عبارت ساده شده بصورت زیر می‌بود

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

این عبارت به دو گیت AND و یک گیت OR نیاز دارد در حالی که عبارت قبلی فقط به یک AND و یک OR نیاز داشت.

اگر به X های نقشه 0 یا 1 اختصاص یابد تابع بطور کامل مشخص می‌گردد. بنابراین عبارت

$$F = A' + BC'$$

فرم ساده شده تابع

$$F(A, B, C) = \sum (0, 1, 2, 3, 6)$$

است. تابع ساده شده ترکیبی از مینترم‌های اصلی 0، 2 و 6 و مینترم بی‌اهمیت 1 و 3 است. مینترم 5 در تابع گنجانده نشده است. از آنجا که مینترم‌های 1، 3 و 5 بعنوان حالات بی‌اهمیت در تابع در نظر گرفته شدند ما مینترم‌های 1 و 3 را با مقدار 1 و مینترم 5 را 0 انتخاب کردیم. این انتخاب باین علت صورت گرفت تا ساده‌ترین فرم برای عبارت بولی حاصل شود.

۱-۵ مدارهای ترکیبی

یک مدار ترکیبی آرایشی از گیت‌های منطقی متصل بهم با مجموعه‌ای از ورودی‌ها و خروجی‌هاست. در

هر لحظه از زمان، مقادیر دودویی خروجی‌ها تابعی از ترکیب دودویی ورودی‌هاست. بلاک دیاگرام مدار ترکیبی در شکل ۱۵-۱ دیده می‌شود. n متغیر دودویی ورودی از یک منبع خارجی سرچشمه گرفته و m متغیر دودویی خروجی هم به یک مقصد خارجی می‌روند و در بین این دو، اتصالات داخلی گیت‌های منطقی قرار دارد. یک مدار ترکیبی اطلاعات دودودیی داده‌های ورودی را به داده‌های خروجی تبدیل می‌نماید. مدارهای ترکیبی در کامپیوترهای دیجیتال به منظور تولید تصمیمات کنترلی دودویی استفاده می‌شوند. همچنین بخش‌های لازم برای پردازش داده هم از این مدارات ساخته شده‌اند.

مدار ترکیبی بوسیله جدول درستی‌اش که رابطه دودویی بین n متغیر ورودی دودویی و m متغیر خروجی را نشان می‌دهد توصیف می‌شود. جدول درستی مقادیر دودویی خروجی را برای هر 2^n ترکیب ورودی نشان می‌دهد. مدار ترکیبی می‌تواند بوسیله m تابع بولی، که هر یک متعلق به یک متغیر خروجی است، مشخص شود. هر تابع برحسب n متغیر ورودی بیان می‌گردد.

تحلیل یک مدار ترکیبی با دیاگرام مدار منطقی شروع و با مجموعه‌ای از توابع بولی یا جدول درستی ختم می‌گردد. اگر مدار دیجیتال بوسیله توصیف کلامی تابعش بیان شود، تابع بولی یا جدول درستی برای تحقیق عملکرد آن کافی است. اگر هدف کار مدار باشد، باید عملکرد آن را با استفاده از توابع بولی یا جدول درستی بدست آمده تفسیر کرد. موفقیت در این کار مستلزم تجربه و آشنایی با مدارهای دیجیتال است. توانایی در مرتبط ساختن یک جدول درستی یا مجموعه‌ای از توابع بولی با یک کار پردازش اطلاعات هنری است که از تجربه فرد حاصل می‌شود.

طراحی مدارهای ترکیبی از تشریح کلامی مسئله آغاز می‌شود و به یک دیاگرام مدار منطقی ختم می‌گردد. رویه طراحی مراحل زیر را شامل می‌شود

۱- بیان مسئله

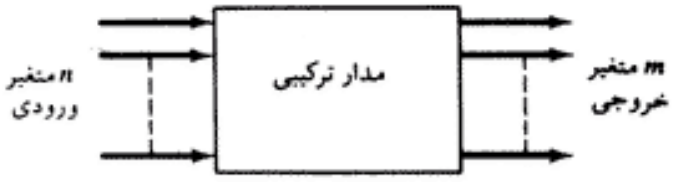
۲- اختصاص سمبل‌های حرفی به متغیرهای ورودی و خروجی

۳- بدست آوردن جدول درستی که رابطه بین ورودی‌ها و خروجی‌ها را تعریف می‌کند

۴- بدست آوردن توابع بولی برای هر یک از خروجیها

۵- رسم دیاگرام منطقی

ما برای نشان دادن روش طراحی مدارهای ترکیبی، دو مثال از مدارهای حسابی را ارائه می‌نماییم.



شکل ۱۵-۱ بلاک دیاگرام یک مدار ترکیبی

این مدارها به عنوان بلاک‌های ساختمانی پایه در ساخت مدارهای حسابی پیچیده‌تر بکار می‌روند.

نیم جمع کننده[1]

اساسی‌ترین مدار محاسباتی دیجیتال، مدار جمع دو رقم دودویی است. مدار ترکیبی که جمع حسابی دو بیت را انجام می‌دهد نیم جمع کننده (HA) نامیده می‌شود. مداری که جمع سه بیت را انجام می‌دهد (جمع دو بیت و بیت رقم نقلی قبلی) تمام جمع کننده[2] خوانده می‌شود. نام تمام جمع کننده از این حقیقت ناشی می‌شود که برای پیاده سازی یک تمام جمع کننده، دو نیم جمع کننده لازم است.

متغیرهای ورودی یک نیم جمع کننده مضاف و مضاف‌الیه خوانده می‌شوند. متغیرهای خروجی نیز مجموع و رقم نقلی نام دارند. در این مدار دو خروجی باید وجود داشته باشد زیرا جمع 1 + 1 عدد دودویی 10 است که دو رقم دارد. ما سمبل‌های x و y را به دو متغیر ورودی و S[3] و C[4] را به دو متغیر خروجی اختصاص می‌دهیم. جدول درستی نیم جمع کننده در شکل ۱۶-۱ (الف) نشان داده شده است. خروجی C برابر 0 است مگر اینکه هر دو ورودی 1 باشند. خروجی S بیت کم ارزشتر[5] (پایین رتبه) مجموع می‌باشد. توابع بولی دو خروجی را می‌توان مستقیماً از جدول درستی بدست آورد.

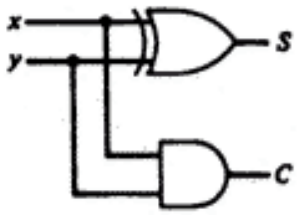
$$S = x'y + xy' = x \oplus y$$

$$C = xy$$

دیاگرام منطقی تابع در شکل ۱۶-۱ (ب) نشان داده شده است. این دیاگرام از یک گیت OR انحصاری و یک گیت AND تشکیل شده است.

تمام جمع کننده

یک تمام جمع کننده (FA) مداری ترکیبی است که مجموع حسابی سه بیت را تشکیل می‌دهد. مدار



(ب) دیاگرام منطقی

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

(الف) جدول درستی

شکل ۱۶-۱ نیم جمع‌کننده

1- Half Adder
2- Full Adder
3- Sum
4- Carry
5- Least Significant

دارای سه ورودی و دو خروجی است. دو متغیر ورودی که با x و y مشخص شده‌اند بیانگر دو بیت با معنایی هستند که باید جمع شوند. ورودی سوم یعنی z رقم نقلی حاصل از عمل جمع در مکان کم ارزشتر قبلی در عدد است. برای مدار دو خروجی لازم است زیرا جمع حسابی سه رقم دودویی بین 0 تا 3 است و اعداد دودویی 2 یا 3 به دو رقم نیاز دارند. خروجی‌ها با S و C مشخص شده‌اند. متغیر S مقدار کم‌ارزشتر مجموع و C مقدار نقلی خروجی که مقدار با ارزشتر مجموع نیز هست را می‌دهند. جدول درستی تمام جمع کننده در جدول ۲-۱ نشان داده شده است. هشت سطر زیر متغیرهای ورودی تمام ترکیب‌های ممکنی راکه متغیرها می‌توانند اختیار کنند مشخص می‌نمایند. مقادیر متغیرهای خروجی از حاصل جمع بیت‌های ورودی معین می‌شوند. وقتی همه بیت‌ها 0 باشند، خروجی 0 است. خروجی S هنگامی 1 می‌شود که فقط یک ورودی یا همه ورودی‌ها 1 باشند. خروجی C نیز برابر 1 است اگر هر دو یا هر سه ورودی 1 باشند.

نقشه‌های شکل ۱۷-۱ برای یافتن عبارات جبری دو متغیر خروجی بکار می‌روند. 1 ها درمربع‌های نقشه‌های S و C مستقیماً از مینترم‌های جدول درستی تعیین می‌شوند. مربعات مربوط به خروجی S که در آن‌ها 1 ها قرار دارند با هم ترکیب نمی‌شوند زیرا با هم مجاور نیستند. ولی چون خروجی هنگامی 1 است که تعداد فردی از ورودی‌ها 1 باشند، پس S یک تابع فرد بوده و رابطه OR انحصاری بین ورودی‌ها برقرار خواهد بود (بحث انتهای بخش ۲-۱ ملاحظه شود). مربع‌هائی که برای C مقدار 1 دارند را می‌توان بطرق مختلف با هم ترکیب کرد. یک عبارت ممکن برای C بفرم زیر است:

$$C = x'y + (xy + xy')\, z$$

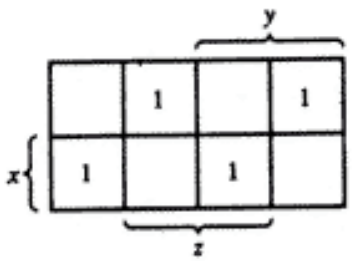
نظر به رابطه $x'y + xy' = x \oplus y$، دو عبارت بولی مربوط به تمام جمع کننده بصورت زیر خواهد بود

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

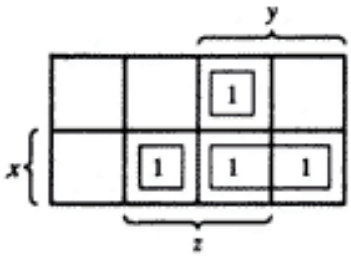
$$C = xy + (x \oplus y)\, z$$

جدول ۲-۱ جدول درستی برای تمام جمع‌کننده

خروجی‌ها			ورودی‌ها	
x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

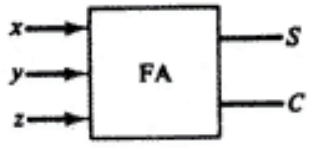


$S = x'y'z' + x'yz' + xy'z' + xyz$
$= x \oplus y \oplus z$

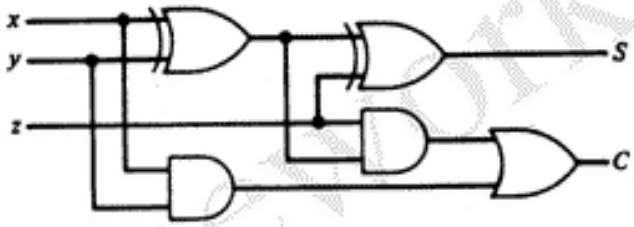


$C = xy + xz + yz$
$= xy + (x'y + xy')z$

شکل ۱۷-۱ نقشه‌ها برای تمام جمع‌کننده



(ب) بلاک دیاگرام



(الف) دیاگرام منطقی

شکل ۱۸-۱ مدار تمام جمع‌کننده

دیاگرام منطقی تمام جمع کننده در شکل ۱۸-۱ رسم شده است. توجه کنید که مدار تمام جمع کننده از دو نیم جمع کننده و یک گیت OR ساخته شده است. هنگام استفاده از یک تمام جمع کننده (FA)، شکل ۱۸-۱ (ب) بکار گرفته خواهد شد.

۱-۶ فلیپ فلاپها[1]

تاکنون مدارهای بررسی شده دیجیتال از نوع ترکیبی بودند، که در آنها خروجی‌ها در هر لحظه کلاً به ورودی‌های همان لحظه بستگی داشتند. هرچند هر سیستم دیجیتال احتمالاً یک مدار ترکیبی دارد، اغلب سیستم‌هایی که در عمل با آنها مواجه می‌شویم دارای عناصر حافظه نیز هستند و لذا سیستم باید در چهارچوب مدارهای ترتیبی مورد بررسی قرار گیرد. متداول‌ترین نوع مدار ترتیبی نوع همگام[2] (همزمان یا سنکرون) آن است. مدارهای ترتیبی همگام در لحظه‌های گسسته و معینی از زمان بر عناصر حافظه اثر می‌گذارند. همزمان سازی با یک وسیله زمانبندی یا یک مولد پالس ساعت[3] حاصل می‌شود که رشته‌ای از پالس‌های ساعت را بطور پریودیک تولید می‌نماید. پالس‌های ساعت در سراسر سیستم توزیع می‌شوند و عناصر حافظه فقط با رسیدن پالس همگام کننده تأثیر می‌پذیرند. مدارهای همگام

1- Flip Flop
2- Synchronous
3- Clock Pulse Generator

ساعت‌دار مدارهایی هستند که بیش از همه در عمل با آنها مواجه می‌شویم. این مدارها ندرتاً با مسئله ناپایداری روبرو هستند و زمانبندی آنها به مقاطع زمانی مستقل قابل تفکیک می‌باشند. هر یک از این مقاطع بطور جداگانه قابل بررسی می‌باشند.

عناصر حافظه بکار رفته در مدارهای ترتیبی ساعت دار فلیپ فلاپ خوانده می‌شوند. فلیپ فلاپ یک سلول[1] دودویی است که می‌تواند یک بیت اطلاعات را در خود ذخیره نماید. این سلول دارای دو خروجی است، یکی برای مقدار عادی بیتی که در آن ذخیره می‌شود و دیگری مقدار متمم آن. فلیپ فلاپ یک وضعیت[2] دودویی را در خود نگهمیدارد تا اینکه یک پالس ساعت موجب تغییر آن حالت شود. تفاوت بین انواع فلیپ فلاپها در تعداد ورودی‌های آنها و نحوه تأثیر این ورودی‌ها بر وضعیت یا حالت دودویی آنهاست. متداول‌ترین انواع فلیپ فلاپ‌ها در بخش‌های زیر ارائه شده‌اند.

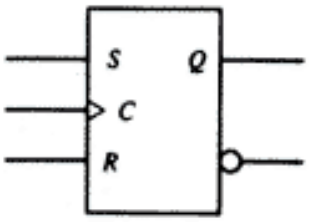
فلیپ فلاپ SR

سمبل گرافیکی فلیپ فلاپ SR در شکل ۱۹-۱ (الف) دیده می‌شود. این فلیپ فلاپ دارای سه ورودی با نام‌های نشاندن[3] (S) حذف کردن[4] (R) و ساعت[5] (C) است. خروجی فلیپ فلاپ Q نامیده می‌شود وگاهی دارای خروجی متمم شده نیز هست که با دایره کوچکی در پایانه خروجی دیگر مشخص شده است. در جلو حرف C یک نوک فلش قرار گرفته که بمعنی دینامیک بودن (پویا بودن) ورودی است. سمبل دینامیکی نشاندهنده این واقعیت است که فلیپ فلاپ به گذار مثبت (لبه بالا رونده) پالس پاسخ می‌دهد.

عملکرد فلیپ فلاپ SR بشرح زیر است. اگر سیگنالی در ورودی ساعت C وجود نداشته باشد مقادیر در ورودی‌های S و R هرچه باشند، خروجی مدار تغییری نمی‌یابد. فقط وقتی که سیگنال ساعت از 0 به 1 تغییر نماید خروجی تحت تأثیر ورودی‌های S و R قرار خواهد گرفت. بهنگام تغییر C از 0 به

S	R	Q (t + 1)	
0	0	Q (t)	بلاتغییر
0	1	0	پاک شدن، 0
1	0	1	نشانده شدن، 1
1	1	?	نامعین

(ب) جدول مشخصه



(الف) سمبل گرافیکی

شکل ۱۹-۱ فلیپ فلاپ SR

1- Cell
2- State
3- Set
4- Reset
5- Clock

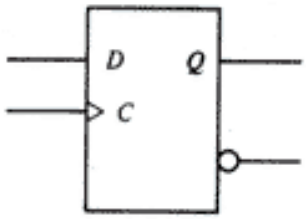
1 اگر S = 1 و R = 0 باشند در خروجی Q مقدار 1 نشانده می‌شود. اگر در تغییر C از 0 به 1 مقادیر S=0 و R= 1 باشند خروجی Q با 0 پاک می‌شود .اگر در گذار ساعت (از 0 به 1)، S و R هر دو 0 باشند خروجی تغییری نمی‌کند. هنگامی که S و R هر دو 1 باشند، خروجی غیر قابل پیش‌بینی است و به تأخیرهای داخلی که در مدار رخ می‌دهد بستگی داشته و ممکن است 0 و یا 1 گردد.

جدول مشخصه شکل ۱۹-۱ (ب) عملکرد یک فلیپ فلاپ SR را جدول وار نشان می‌دهد. در ستون‌های S و R مقادیر دودویی دو ورودی مشخص شده‌اند. Q(t) نیز حالت خروجی Q در یک زمان معین است (که به آن حالت فعلی می‌گویند). Q(t + 1) حالت دودویی خروجی Q پس از لبه مثبت پالس ساعت است (حالت بعدی). اگر S = R = 0 باشد، لبه (گذار) پالس تغییری در حالت بوجود نمی‌آورد [یعنی Q(t) = Q (t+1)]. اگر S = 0 و R = 1 باشد فلیپ فلاپ به حالت 0 (پاک شده) می‌رود. اگر S = 1 و R = 0 باشد فلیپ فلاپ به وضعیت 1 (نشانده شده) می‌رود. به فلیپ فلاپ در S=1 و R = 1 نباید پالس داد زیرا تولید حالت بعدی نامعینی را خواهد کرد. این حالت نامعین کنترل فلیپ فلاپ را مشخص می‌سازد و لذا در عمل بندرت مورد استفاده قرار می‌گیرد.

فلیپ فلاپ D[1]

فلیپ فلاپ D (داده) با تغییر کوچکی در فلیپ فلاپ SR بدست می‌آید. فلیپ فلاپ SR با قرار دادن یک معکوس کننده بین S و R و اختصاص سمبل D به تنها ورودی حاصل، به فلیپ فلاپ D تبدیل می‌شود. ورودی D هنگام انتقال پالس ساعت از 0 به 1 نمونه برداری می‌شود. اگر D = 1 باشد، خروجی فلیپ فلاپ به حالت 1 می‌رود، اما در D = 0 ، خروجی فلیپ فلاپ به وضعیت 0 خواهد رفت.

سمبل گرافیکی و جدول مشخصه فلیپ فلاپ D در شکل ۲۰-۱ نشان داده شده است. از جدول مزبور درمی‌یابیم که حالت بعدی Q(t+1) توسط ورودی D معین می‌شود. این ارتباط بوسیله معادله



(الف) سمبل گرافیکی

D	Q (t + 1)	
0	0	پاک شدن، 0
1	1	نشانده شدن، 1

(ب) جدول مشخصه

شکل ۲۰-۱ فلیپ فلاپ D

1- Data File Flop

مشخصه زیر بیان می‌گردد

$$Q(t+1) = D$$

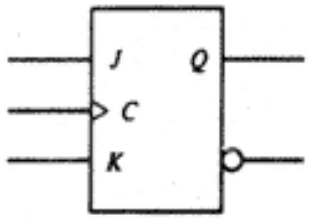
این بدان معنی است که خروجی Q فلیپ فلاپ مقدار خود را، هر بار بهنگام انتقال سیگنال ساعت از 0 به 1 از ورودی D دریافت می‌کند.

دقت کنید که حالتی از فلیپ فلاپ وجود ندارد که خروجی را تغییر ندهد. هرچند فلیپ فلاپ D از مزیت داشتن فقط یک ورودی (بدون احتساب C) بهره می‌برد ولی عیب آن این است که جدول مشخصه آن فاقد حالت "بلاتغییر"[1] $Q(t+1) = Q(t)$ می‌باشد. در این فلیپ فلاپ حالت بلاتغییر بدین صورت انجام می‌شود که یا باید سیگنال ساعت غیر فعال گردد و یا خروجی فلیپ فلاپ به ورودی آن بازگردانده[2] شود، بطوری که پالس‌های ساعت حالت فلیپ فلاپ را تغییر ندهند.

فلیپ فلاپ JK

یک فلیپ فلاپ JK نوع اصلاح شده فلیپ فلاپ SR است که حالت "نامعین"[3] در فیلپ فلاپ SR در آن "معین" شده است. ورودی‌های J و K در نشاندن و یا پاک کردن فلیپ فلاپ بترتیب مانند S و R عمل می‌کنند. وقتی هر دو ورودی J و K برابر 1 باشند، انتقال یا گذار پالس ساعت خروجی‌های فلیپ‌فلاپ را به حالت متممشان سوئیچ می‌کنند.

سمبل گرافیکی و جدول مشخصه فلیپ فلاپ JK معادل با ورودی R در شکل ۲۱-۱ نشان داده شده است. ورودی J معادل ورودی S و ورودی K معادل ورودی R در فلیپ‌فلاپ SR است. در فلیپ فلاپ JK اگر هر دو ورودی J و K برابر 1 باشند، بجای حالت نامعین، حالت متمم $Q(t+1) = Q'(t)$ را خواهیم داشت.



(الف) سمبل گرافیکی

J	K	Q (t + 1)	
0	0	Q (t)	بلاتغییر
0	1	0	پاک شدن، 0
1	0	1	نشانده شدن، 1
1	1	Q′ (t)	متمم

(ب) جدول مشخصه

شکل ۲۱-۱ فلیپ فلاپ JK

1- Unchanged 2- Feedback 3- Indeterminate

فليپ فلاپ T[1]

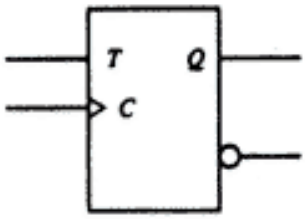
نوع ديگری از فليپ فلاپها كه در كتب درسی ديده می‌شود فليپ فلاپ T می‌باشد. اين فليپ فلاپ، كه در شكل ۲۲-۱ نشان داده شده است از فليپ فلاپ نوع JK و با اتصال ورودی‌های J و K به يكديگر و در نتيجه ايجاد يک ورودی واحد كه با T نشان داده می‌شود بدست می‌آيد. بنابراين فليپ فلاپ T فقط دو حالت دارد. وقتی T = 0 باشد (J = K = 0) يک گذار ساعت حالت فليپ فلاپ را تغييرنمی‌دهد. وقتی T = 1 باشد (J = K = 1) انتقال ساعت حالت فليپ فلاپ را متمم می‌كند. اين حالات را می‌توان با معادله مشخصه زير بيان كرد

$$Q(t+1) = Q(t) \oplus T$$

فليپ فلاپ‌های تحريک شونده با لبه[2]

متداول‌ترين نوع فليپ فلاپ‌های مورد استفاده در همگام كردن تغيير حالت به هنگام تغيير وضعيت يا گذار پالس ساعت فليپ فلاپ تحريک شونده با لبه (يا فليپ فلاپ حساس به لبه) می‌باشد. در اين نوع فليپ فلاپ‌ها، تغييرات در خروجی بازاء سطح خاصی از ورودی پالس ساعت رخ می‌دهد. وقتی كه سطح ورودی پالس ساعت از اين سطح آستانه[3] تجاوز كند ورودی‌ها قفل می‌شوند بطوری كه فليپ فلاپ به تغييرات ديگر در ورودی، تا بازگشت پالس ساعت به 0 و ظهور پالس ساعتی ديگر، پاسخی نمی‌دهد. در بعضی از اين نوع فليپ فلاپ‌ها تأثير ورودی‌ها در خروجی در لبه بالا رونده سيگنال ساعت (گذار يا انتقال در لبه مثبت) و برخی نيز در لبه پايين رونده سينگال (گذار در لبه منفی) ميسّر است.

شكل ۲۳-۱ (الف) سيگنال پالس ساعت را در يک فليپ فلاپ D حساس به لبه مثبت نشان



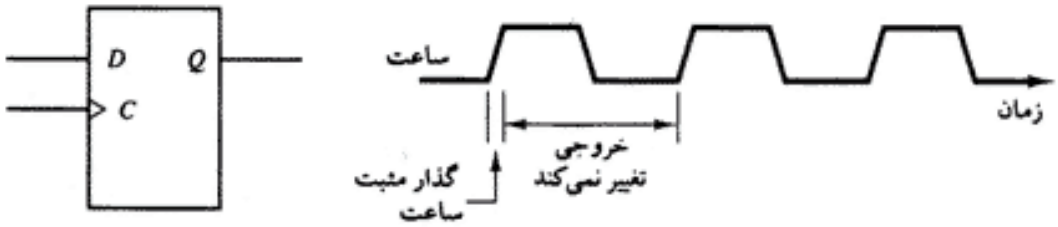
(الف) سمبل گرافيکی

T	Q(t + 1)	
0	Q(t)	بلاتغيير
1	Q'(t)	متمم

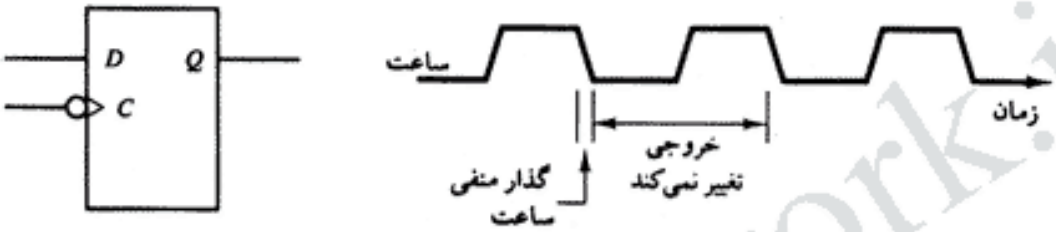
(ب) جدول مشخصه

شكل ۲۲-۱ فليپ فلاپ T

1- Toggle
2- Edge- Triggered Flip Flop
3- Threshold level



(الف) فليپ فلاپ D تحريک شونده با لبه مثبت



(ب) فليپ فلاپ D تحريک شونده با لبه منفی

شكل ۲۳-۱ فليپ فلاپ تحريک شونده با لبه

می‌دهد. مقدار ورودی D در لبه مثبت پالس ساعت به خروجی Q منتقل می‌شود. هنگامی که پالس ساعت در سطح 1، در سطح 0 و یا در حال انتقال از سطح 1 به 0 باشد خروجی تغییری نمی‌کند. گذار یا انتقال مثبت پالس شامل دو فاصله زمانی است که یکی حداقل زمان آماده سازی[1] می‌باشد و طی آن ورودی D باید در یک سطح ثابتی قبل از گذار پالس ساعت باقی بماند، و دیگری زمان معینی با نام زمان نگهداری[2] است که طی آن ورودی D نباید پس از گذار پالس ساعت تغییر کند. زمان مربوط به انتقال یا گذار مثبت کسر بسیار کوچکی از کل دوره تناوب پالس ساعت است.

شکل ۲۳-۱ (ب) سمبل‌های گرافیکی و دیاگرام زمانی مربوط به فلیپ فلاپ D حساس به لبه منفی را نشان می‌دهد. در سمبل گرافیکی، یک دایره کوچک منفی‌کننده[3] در جلو علامت دینامیکی در ورودی C قرار دارد. این علامت بیانگر تحریک فلیپ فلاپ در لبه منفی پالس است. در این حالت پاسخ فلیپ فلاپ در قبال انتقال سطح سیگنال ساعت از سطح 1 به 0 می‌باشد.

نوع دیگری از فلیپ‌فلاپ‌که در بعضی از سیستم‌ها بکار می‌رود فلیپ فلاپ ماستر- اسلیو[4] (بمعنی حاکم و پیرو) است. این نوع مدار از دو فلیپ فلاپ تشکیل شده است. اولین فلیپ فلاپ ماستر نامیده می‌شود و به سطح مثبت پالس ساعت پاسخ می‌دهد، و دومین فلیپ فلاپ اسلیو می‌باشد که به سطح منفی پالس ساعت واکنش نشان می‌دهد. در نتیجه خروجی بهنگام تغییر 1 به 0 سیگنال ساعت تغییر

1- Set UP Time
2- Hold Time
3- Negation
4- Master – Slave

می‌نماید. امروزه گرایش در کنار گذاشتن فلیپ فلاپ ماستر- اسلیو و بکارگیری فلیپ فلاپ حساس به لبه دیده می‌شود.

فلیپ فلاپ‌های موجود بصورت مدار مجتمع گاهی دارای پایانه‌های خاصی برای نشاندن[1] و پاک‌کردن[2] غیر همگام[3] می‌باشند. این ورودی‌ها معمولاً نشاندن اولیه[4] و پاک کردن نامیده می‌شوند و با سطح منفی سیگنال ورودی خود بدون نیاز به پالس ساعت بر فلیپ فلاپ اثر می‌گذارند. این ورودی‌ها برای قرار دادن فلیپ فلاپ‌ها در یک وضعیت اولیه قبل از شروع بکار با پالس ساعت مفیدند.

جداول تحریک

جداول تحریک فلیپ فلاپ‌ها حالت بعدی را هرگاه ورودی‌ها و حالت فعلی معلوم باشند مشخص می‌کنند. معمولاً هنگام طراحی مدارهای ترتیبی انتقال از حالت فعلی به حالت بعدی برای ما شناخته شده است و ما مایلیم تا ورودی‌های فلیپ فلاپ را طوری بیابیم که انتقال مناسب انجام یابد. به همین دلیل به جدولی نیاز داریم که برحسب تغییر مفروضی در حالت، ورودی‌های مورد نیاز را نشان دهد. چنین لیستی از حالت‌ها، جدول تحریک فلیپ فلاپ نامیده می‌شود.

جدول ۳-۱ جداول تحریک چهار نوع فلیپ فلاپ را نشان می‌دهد. هر جدول متشکل از دو ستون Q (t) و Q (t+1) و یک ستون برای هر ورودی است که چگونگی وقوع انتقال مورد نظر را نشان می‌دهد. انتقال از حالت فعلی Q(t) به حالت بعدی Q(t+1) به چهار طریق امکان پذیر است.

جدول ۳-۱ جدول تحریک برای چهار فلیپ فلاپ

فلیپ فلاپ SR

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

فلیپ فلاپ D

Q(t)	Q(t + 1)	D
0	0	0
0	1	1
1	0	0
1	1	1

فلیپ فلاپ JK

Q(t)	Q(t + 1)	J	K
0	0	0	×
0	1	1	×
1	0	×	1
1	1	×	0

فلیپ‌فلاپ T

Q(t)	Q(t + 1)	T
0	0	0
0	1	1
1	0	1
1	1	0

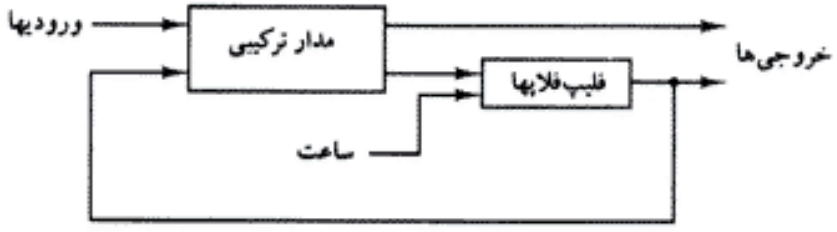
1- Set
2- Clear
3- asynchronous
4- Preset

ورودی‌های مورد نیاز برای هر یک از چهار انتقال، از اطلاعات موجود در جدول مشخصه بدست آمده‌اند. سمبل X در جداول نشاندهنده حالت بی‌اهمیت است و باین معنی است که 0 یا 1 بودن ورودی مهم نیست.

دلیل وجود حالات بی‌اهمیت در جداول تحریک این است که برای دستیابی به یک انتقال مورد نظر دو راه وجود دارد. مثلاً در یک فلیپ فلاپ JK، انتقال از حالت فعلی 0 به حالت بعدی 0 را می‌توان با قرار دادن J و K در حالت 0 (برای حالت بلاتغییر) یا با قرار دادن $J = 0$ و $K = 1$ برای پاک کردن فلیپ‌فلاپ (گرچه از قبل پاک شده است) بدست آورد. در هر دو مورد J باید 0 باشد ولی K در مورد اول 0 و در مورد دوم 1 است. چون در هر صورت انتقال مورد نظر رخ خواهد داد، ما ورودی K را با X علامت می‌زنیم و لذا طراح اجازه خواهد داشت مقدار K را با توجه به اینکه کدام مناسبتر است، 1 یا 0 انتخاب کند.

۱-۷ مدارهای ترتیبی

مدار ترتیبی از اتصال مجموعه‌ای از فلیپ‌فلاپها و گیت‌ها بوجود می‌آید. گیت‌ها به تنهایی مدارهای ترکیبی را تشکیل می‌دهند ولی وقتی با فلیپ فلاپها همراه باشند مدار حاصل از رده مدارهای ترتیبی خواهد بود. بلوک دیاگرام یک مدار ترتیبی ساعت‌دار در شکل ۲۴-۱ نشان داده شده است. این مدار از یک مدار ترکیبی و تعدادی فلیپ فلاپ ساعت‌دار تشکیل شده است. بطور کلی، هر تعداد یا هر نوع فلیپ فلاپ می‌تواند در مدار بکار رود. همانطور که در دیاگرام نشان داده شده است، بلاک مدار ترکیبی سیگنالهای دودویی را از ورودی‌های خارجی و از خروجی‌های فلیپ‌فلاپها دریافت می‌کند. خروجی‌های مدار ترکیبی به خروجی‌های خارجی و به ورودی‌های فلیپ فلاپها می‌روند. گیت‌های تشکیل دهنده مدار ترکیبی مقادیر ذخیره شونده در فلیپ فلاپها را بعد از هر گذار پالس ساعت معین می‌کنند. خروجی فلیپ فلاپها به نوبه خود به ورودی‌های مدار ترکیبی اعمال شده و رفتار مدار را تعیین می‌کنند. این فرآیند نشان می‌دهد که خروجی‌های خارجی مدار ترتیبی تابعی از ورودی‌های خارجی و حالت فعلی[1] فلیپ فلاپها است. بعلاوه، حالت بعدی[2] فلیپ فلاپها نیز تابعی از حالت فعلی آنها و ورودی‌های خارجی



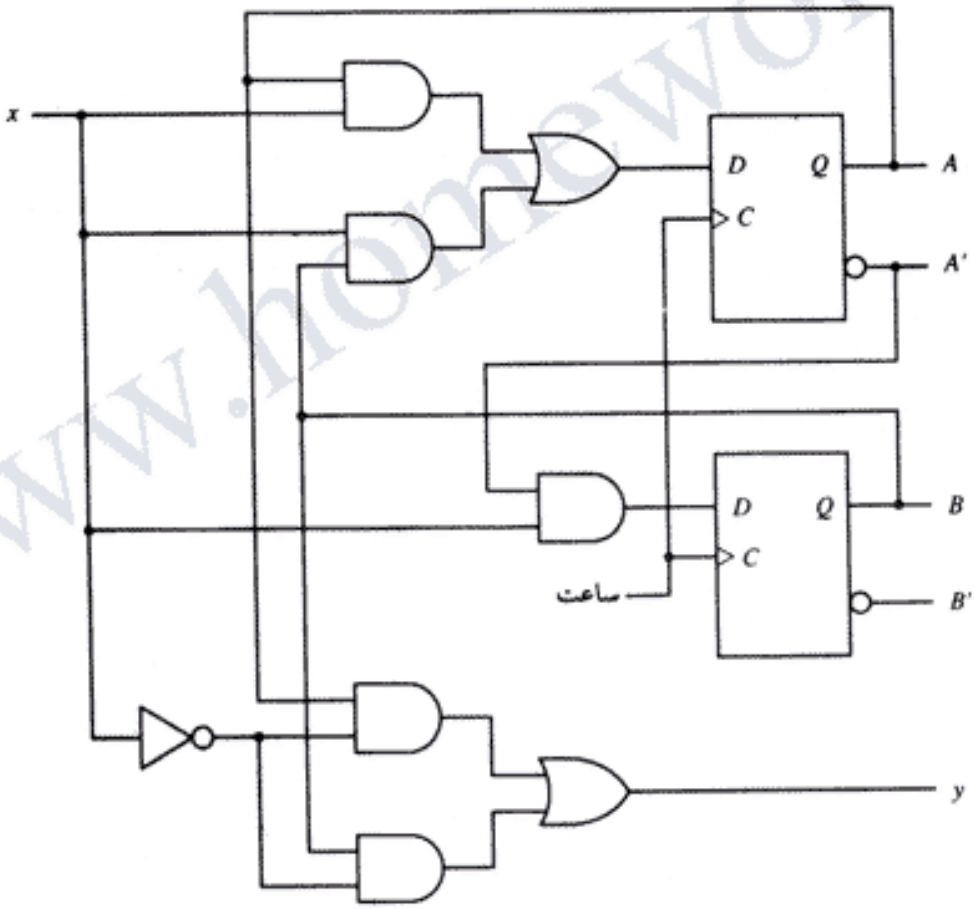
شکل ۲۴-۱ بلاک دیاگرام یک مدار ترتیبی همگام ساعت‌دار

1- Present State 2- Next State

می‌باشد. بنابراین یک مدار ترتیبی با یک رشته ورودی‌های زمانبندی شده، خروجی‌های خارجی و حالات دودویی فلیپ فلاپهای داخلی‌اش مشخص می‌گردد.

معادلات ورودی فلیپ فلاپها[1]

مثالی از یک مدار ترتیبی در شکل ۲۵-۱ دیده می‌شود. این مدار دارای یک متغیر ورودی x، یک متغیر خروجی y و دو فلیپ فلاپ D ساعت‌دار است. گیت‌های AND، OR و معکوس کننده بخش ترکیبی مدار منطقی را تشکیل می‌دهند. اتصالات بین گیت‌ها در مدار ترکیبی می‌توانند بوسیله مجموعه‌ای از عبارات بولی مشخص گردد. بخشی از مدار ترکیبی که ورودی‌های فلیپ فلاپ را تولید می‌کنند بوسیله مجموعه‌ای از عبارات بولی که معادلات ورودی فلیپ فلاپها نام دارند توصیف می‌شود. ما سمبل ورودی هر فلیپ فلاپ را بعنوان متغیر معادله ورودی و اندیس را برای مشخص کردن سمبل



شکل ۲۵-۱ مثالی از یک مدار ترتیبی

1- Flip Flop Input Equation

انتخاب شده برمی‌گزینیم. بنابراین در شکل ۲۵-۱ ما دو معادله ورودی خواهیم داشت که با D_A و D_B مشخص می‌شوند. اولین حرف در هر مورد نماینده ورودی D یک فلیپ فلاپ D و حرف اندیس نام فلیپ فلاپ مربوطه است. معادلات ورودی، توابع بولی برای متغیرهای ورودی فلیپ فلاپ بوده و از بررسی مدار بدست می‌آیند. چون خروجی گیت OR به ورودی D از فلیپ فلاپ A متصل شده است ما اولین معادله ورودی را می‌نویسیم

$$D_A = Ax + Bx$$

که A و B خروجی‌های دو فلیپ فلاپ و x ورودی خارجی است. دومین معادله ورودی از خروجی گیت AND متصل به ورودی D فلیپ فلاپ B حاصل می‌شود.

$$D_B = A'x$$

این مدار ترتیبی یک خروجی خارجی نیز دارد که تابعی از متغیر ورودی و حالات فلیپ فلاپها است. این خروجی می‌تواند بوسیله عبارت جبری زیر مشخص شود

$$y = Ax' + Bx'$$

از این مثال در می‌یابیم که یک معادله ورودی فلیپ فلاپ عبارتی بولی برای یک مدار ترکیبی است. متغیر اندیس‌دار نام یک متغیر دودویی برای خروجی یک مدار ترکیبی است. این خروجی همواره به یکی از ورودی‌های فلیپ فلاپ‌ها متصل است.

جدول حالت[1]

رفتار یک مدار ترتیبی از ورودیها، خروجیها و حالت فلیپ فلاپ‌های آن معین می‌شود. خروجی‌ها و حالت بعدی تابعی از ورودی‌ها و حالت فعلی هستند. یک مدار ترتیبی توسط جدول حالتی مشخص می‌شود که خروجی‌ها و حالات بعدی را به صورت تابعی از ورودی و حالات فعلی مرتبط می‌سازد. در مدارهای ترتیبی ساعت‌دار، انتقال از حالت فعلی به حالت بعدی بوسیله رخداد یک سیگنال ساعت ممکن می‌شود.

جدول حالت شکل ۲۵-۱ در جدول ۴-۱ نشان داده شده است. جدول متشکل از چهار بخش است که با عناوین حالت فعلی، ورودی، حالت بعد و خروجی مشخص شده‌اند. بخش حالت فعلی حالات فلیپ فلاپ‌های A و B را در هر لحظه از زمان مفروض t نشان می‌دهد. بخش ورودی مقداری از x را در هر حالت فعلی معین می‌سازد. بخش حالت بعدی حالات فلیپ فلاپ‌ها را در پریود زمانی بعدی، t+1، بدست می‌دهد. بخش خروجی نیز مقدار y را در حالت فعلی و وضعیت ورودی می‌دهد.

1- State Table

جدول ۴-۱ جدول حالت برای مدار شکل ۲۵-۱

حالت فعلی		ورودی	حالت بعدی		خروجی
A	B	x	A	B	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

در تشکیل یک جدول حالت، ابتدا لیست تمام ترکیبات دودویی حالت فعلی و ورودی را می‌نویسیم. در این حالت ما هشت ترکیب دودویی از 000 تا 111 خواهیم داشت. سپس مقادیر حالت‌های بعدی از دیاگرام منطقی یا از معادلات ورودی بدست می‌آیند. معادله ورودی برای فلیپ فلاپ A برابر است با

$$D_A = Ax + Bx$$

مقدار حالت بعدی هر فیلپ فلاپ برابر با مقدار ورودی D در حالت فعلی است. انتقال از حالت فعلی به حالت بعدی پس از اعمال سیگنال ساعت بوقوع می‌پیوندد. بنابراین هرگاه حالات فعلی و ورودی شرایط $Ax = 1$ یا $Bx = 1$ را ایجاد کنند D_A برابر 1 و در نتیجه حالت بعدی A هم 1 خواهد شد. این مطلب در جدول حالت با سه عدد 1 زیر ستون حالت بعدی A آورده شده است. بطور مشابه، معادله ورودی برای فلیپ فلاپ B با رابطه زیر داده می‌شود.

$$D_B = A'x$$

حالت بعدی B در جدول حالت هنگامی 1 است که A برابر 0 و x برابر 1 باشد. ستون خروجی نیز از معادله خروجی زیر بدست می‌آید.

$$y = Ax' + Bx'$$

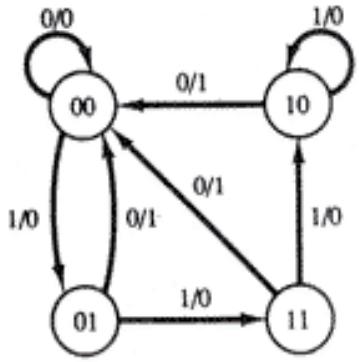
جدول حالت هر مدار ترتیبی با استفاده از روشی که در این مثال دیده شد بدست می‌آید. بطور کلی، یک مدار ترتیبی m فلیپ فلاپ، n متغیر ورودی و p متغیر خروجی دارای m ستون حالت فعلی، n ستون برای ورودی‌ها، m ستون برای حالت بعدی و p ستون برای خروجی‌هاست. ستون‌های حالت فعلی و ورودی با هم ترکیب شده و ما 2^{m+n} ترکیب دودویی، از 0 تا $2^{m+n} - 1$ را در زیر آنها می‌نویسیم. ستون‌های حالت بعدی و خروجی توابعی از حالت فعلی و مقادیر ورودی بوده و مستقیماً

از مدار یا معادلات بولی که مدار را توصیف می‌کنند بدست می‌آیند.

دیاگرام حالت

اطلاعات موجود در جدول حالت را می‌توان بصورت گرافیکی در یک دیاگرام حالت نشان داد. در این نوع دیاگرام، یک حالت بوسیله یک دایره و انتقال بین حالات بوسیله خطوط جهت‌داری که دوایر را بهم وصل می‌کنند نشان داده می‌شود. دیاگرام حالت مدار ترتیبی شکل ۲۵-۱ در شکل ۲۶-۱ نشان داده شده است. دیاگرام حالت همان اطلاعات جدول حالت را در دسترس می‌گذارد. و مستقیماً از جدول ۴-۱ بدست می‌آید. عدد دودویی داخل هر دایره حالت فلیپ فلاپ‌ها را مشخص می‌کند. درکنار خطوط جهت‌دار دو عدد دودویی که با ممیز از هم جدا شده‌اند نوشته شده است. اولین عدد مقدار ورودی را در حالت فعلی و عدد بعد از ممیز، خروجی را در همان حالت نشان می‌دهد. مثلاً کنار خط جهت‌داری که حالت 00 را به 01 وصل می‌کند 0/1 نوشته شده است، و بدان معنی است که وقتی مدار ترتیبی در حالت 00 است بازاء ورودی 1، خروجی 0 خواهد بود. پس از اعمال یک پالس ساعت، مدار به حالت بعدی 01 می‌رود. همین اعمال پالس ساعت ممکن است ورودی را عوض کند. اگر ورودی 0 شود خروجی 1 شده ولی اگر ورودی در 1 باقی بماند خروجی در 0 خواهد ماند. این اطلاعات از دیاگرام حالت و با دنبال کردن دو خط جهت‌داری که به دایره 01 وصل شده‌اند حاصل می‌گردد. خط جهت‌داری که دایره را به خودش وصل می‌کند نشان می‌دهد که تغییر حالتی رخ نداده است.

بین جدول حالت و دیاگرام حالت هیچ تفاوتی جز در نحوه نمایش وجود ندارد. بدست آوردن جدول حالت از دیاگرام منطقی آسان‌تر است و نمودار حالت مستقیماً از جدول حالت بدست می‌آید. دیاگرام حالت یک دید مصور را از انتقال حالات فراهم می‌نماید و بشکلی است که برای تفسیر عملکرد مدار مناسب‌تر می‌باشد. مثلاً دیاگرام حالت شکل ۲۶-۱ بوضوح نشان می‌دهد که با شروع حالت از 00 مادامی که ورودی 1 است خروجی 0 خواهد بود. اولین ورودی 0 پس از رشته‌ای از 1 ها، خروجی 1 تولید خواهد کرد و مدار را به حالت اولیه 00 برمی‌گرداند.



شکل ۲۶-۱ دیاگرام حالت یک مدار ترتیبی

مثال طراحی

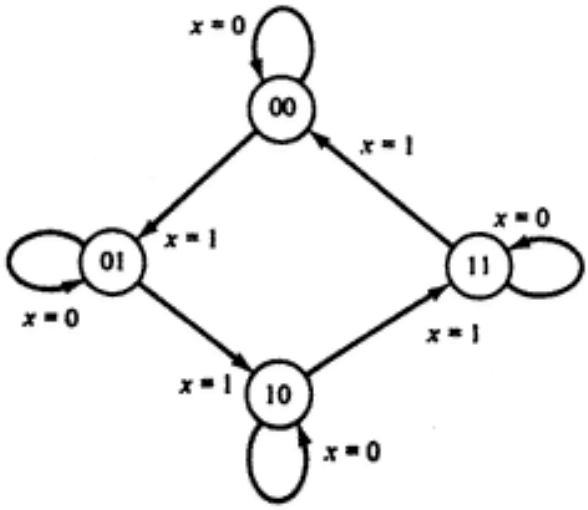
اکنون روش طراحی مدارهای ترتیبی بوسیله ارائه یک مثال خاص نشـان داده مـی‌شود. در روش طراحی ابتدا مشخصات مدار را به یک دیاگرام حالت تبدیل می‌کنیم. سپس دیاگرام حالت به یک جدول حالت تبدیل می‌شود. بالاخره از جدول حالت برای تهیه دیاگرام مدار منطقی استفاده می‌شود.

ما می‌خواهیم یک مدار ترتیبی طراحی کنیم که هر وقت ورودی خارجی x بـرابـر 1 بـاشد رشـته حالات دودویی 00، 01، 10 و 11 را تکرار نماید. بازاء $x = 0$ هم حالت مدار تغییری نکند. این نوع مدار یک شمارنده دودویی دو بیتی است زیرا رشته حالات با شمارش متوالی دو رقم دودویی یکسان می‌باشد. ورودی x یک متغیر کنترل است که مشخص می‌کند شمارش چه زمانی آغاز شود.

شمارنده دودویی به دو فلیپ فلاپ برای نمایش دو رقم نیاز دارد. دیاگرام حالت برای مدار ترتیبی در شکل ۲۷-۱ نشان داده شده است. دیاگرام بدین منظور رسم شده است تا نشان دهیم مادامی که $x = 1$ است مدار شمارش دودویی را ادامه می‌دهد. حالت بعد از 11، حالت 00 است که باعث تکرار شمارش می‌گردد. اگر $x = 0$ باشد، حالت مدار بدون تغییر باقی می‌ماند. این مدار ترتیبی خروجی خارجی ندارد و بنابراین فقط مقدار دودویی در جوار خطوط واصل در دیاگرام نوشته شده است. در این مدار حالت فلیپ فلاپ‌ها بعنوان خروجی‌های شمارنده تصور شده‌اند.

ما قبلاً سمبل x را به متغیر ورودی اختصاص دادیم. حال سمبل‌های A و B را به خروجی‌های دو فلیپ فلاپ تخصیص می‌دهیم. حالت بعدی A و B را، می‌توان بعنوان تابعی از حالت فعلی و ورودی x از دیاگرام حالت به جدول حالت منتقل ساخت. پنج ستون اول جدول ۵-۱ جدول حالت را بوجود می‌آورند. وارده‌های این جدول مستقیماً از دیاگرام حالت بدست می‌آیند.

جدول تحریک یک مدار ترتیبی نوع تعمیم یافته یک جدول حالت است. این تعمیم شامل لیستی



شکل ۲۷-۱ دیاگرام حالت برای شمارنده دودویی

از تحریک‌های ورودی فلیپ فلاپ می‌باشد که موجب انتقال به حالت مورد نظر می‌شوند. شرایط ورودی فلیپ فلاپها تابعی از نوع فلیپ فلاپ بکار رفته می‌باشد. ما اگر از فلیپ فلاپهای JK استفاده کنیم، نیاز به ستون‌های ورودی‌های J و K در هر فلیپ فلاپ داریم. ورودیهای فلیپ فلاپ A با J_A و K_A و ورودی‌های فلیپ فلاپ B با J_B و K_B نشان داده می‌شوند.

جدول تحریک برای فلیپ فلاپ JK که در جدول ۳-۱ نشان داده شد در اینجا برای بدست آوردن جدول تحریک مدار ترتیبی بکار گرفته می‌شود. مثلاً در اولین سطر جدول ۵-۱، انتقالی را از حالت فعلی 0 به حالت بعدی 0 در فلیپ فلاپ A داریم. در جدول ۳-۱ می‌بینیم که انتقال حالتی از Q(t)=0 به Q(t+1)=0 در یک فلیپ فلاپ JK مستلزم ورودی J = 0 و K = X می‌باشد. بنابراین 0 و X بترتیب در سطر اول و در زیر J_A و K_A نوشته می‌شوند. چون اولین سطر انتقال فلیپ فلاپ B را هم از حالت فعلی 0 به حالت بعدی 0 نشان می‌دهد، لذا مجدداً 0 و X در سطر اول و در زیر J_B و K_B نوشته شده‌اند. دومین سطر جدول ۵-۱ انتقال فلیپ فلاپ B را از حالت فعلی 6 به حالت بعدی 1 نشان می‌دهد. با توجه به جدول ۳-۱ می‌بینیم که انتقال از Q(t)=0 به Q(t+1)=1 مستلزم ورودی J=1 و K=x است. بنابراین 1 و 0 در سطر دوم و بترتیب در زیر J_B و K_B نوشته می‌شوند. این روند برای هر سطر جدول و هر فلیپ فلاپ ادامه می‌یابد و شرایط ورودی در جدول ۳-۱ در سطر مناسب و برای هر فلیپ فلاپ مورد نظر نوشته می‌شود.

اکنون اطلاعات موجود در یک جدول تحریک مانند جدول ۵-۱ را ملاحظه می‌کنیم. می‌دانیم که یک مدار ترتیبی از تعدادی فلیپ فلاپ و یک مدار ترکیبی تشکیل شده است. با توجه به شکل ۲۴-۱ می‌بینیم که خروجی‌های مدار ترتیبی باید به چهار ورودی فلیپ فلاپها یعنی J_A، K_A، J_B و K_B متصل شوند. ورودی‌های مدار ترکیبی عبارت از ورودی خارجی x و مقادیر حالات فعلی فلیپ فلاپ‌های A و B می‌باشند. بعلاوه توابع بولی که یک مدار ترکیبی را مشخص می‌کنند از یک جدول درستی که رابطه ورودی - خروجی مدار را نشان می‌دهد حاصل می‌گردند. وارده‌هایی که ورودی‌های مدار ترکیبی را

جدول ۵-۱ جدول تحریک برای شمارنده دودویی

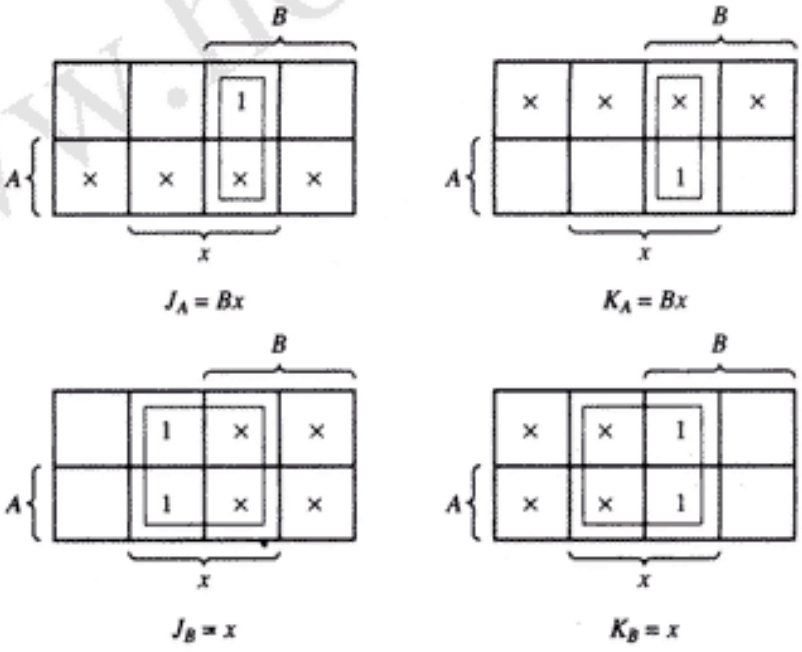
حالت فعلی		ورودی	حالت بعدی		ورودی‌های فلیپ‌فلاپ			
A	B	x	A	B	J_A	K_A	J_B	K_B
0	0	0	0	0	0	×	0	×
0	0	1	0	1	0	×	1	×
0	1	0	0	1	0	×	×	0
0	1	1	1	0	1	×	×	1
1	0	0	1	0	×	0	0	×
1	0	1	1	1	×	0	1	×
1	1	0	1	1	×	0	×	0
1	1	1	0	0	×	1	×	1

جدول تحریک، یک دیاگرام حالت رابه جدول درستی مورد نیاز در طراحی مدار ترکیبی که بخشی از مدار ترتیبی است مبدل می‌سازد.

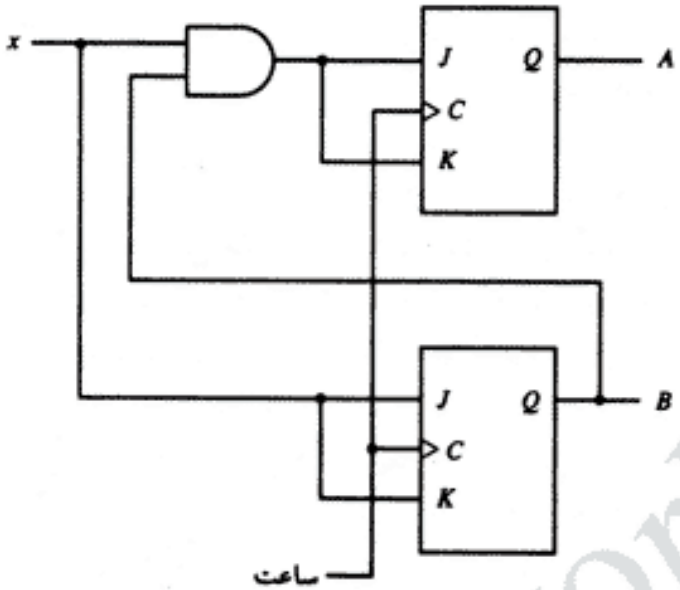
اکنون توابع بولی ساده شده در مدار ترکیبی را می‌توان بدست آورد. ورودیها متغیرهای A ، B و x می‌باشند. خروجی‌ها نیز متغیرهای J_A ، K_A ، J_B و K_B هستند. اطلاعات حاصل از جدول تحریک به نقشه‌های شکل ۲۸-۱ انتقال یافته و چهار معادله ورودی ساده شده فلیپ فلاپ‌ها از آن بدست آمده است

$$J_A = Bx \qquad K_A = Bx$$
$$J_B = x \qquad K_B = x$$

دیاگرام منطقی که در شکل ۲۹-۱ رسم شده است از دو فلیپ فلاپ JK و یک گیت AND تشکیل یافته است. توجه کنید که ورودی‌های J و K حالت بعدی شمارنده را پس از وقوع پالس ساعت معین می‌کنند. اگر هر دو ورودی J و K برابر 0 باشند پالس ساعت تأثیری نخواهد داشت. یعنی حالت فلیپ فلاپ‌ها تغییری نمی‌کند. بنابراین وقتی $x = 0$ باشد، تمام ورودی‌های فلیپ فلاپ‌ها برابر 0 و حالت فلیپ فلاپها بلاتغییر باقی می‌مانند حتی اگر پالسهای ساعت مرتباً به مدار اعمال شوند.



شکل ۲۸-۱ نقشه‌های مدار ترکیبی شمارنده



شکل ۲۹-۱ دیاگرام منطقی یک شمارنده دو بیتی

روش طراحی

طراحی مدارهای ترتیبی رئوس مطالبی راکه در مثال طراحی دیده شد دنبال می‌کند. ابتدا رفتار مدار بصورت دیاگرام حالت فرموله می‌شود. تعداد فلیپ فلاپ‌های لازم برای مدار با توجه به تعداد بیت‌های داخل دایره‌ها در دیاگرام حالت معین می‌گردد. تعداد ورودی‌ها برای مدار در امتداد خطوط واصل بین دایره‌ها مشخص می‌شود. ما سپس حروفی را برای علامت گذاری تمام فلیپ فلاپها، متغیرهای ورودی و خروجی بکار گرفته و برای بدست آوردن جدول حالت به مرحله بعد می‌رویم.

بازاء m فلیپ فلاپ و n ورودی، جدول حالت شامل m ستون برای حالت فعلی، n ستون برای ورودی‌ها و m ستون برای حالت بعدی خواهد بود. تعداد سطرهای جدول تا 2^{m+n} می‌باشد که هر سطر مربوط به یک ترکیب دودویی از حالت فعلی و ورودی‌هاست. در هر سطر ما حالت بعدی را آنچنانکه در دیاگرام حالت مشخص شده لیست می‌کنیم. سپس نوع فلیپ فلاپ بکار رفته در مدار انتخاب می‌شود. جدول حالت به جدول تحریک با اضافه کردن ستون‌های هر ورودی در هر فلیپ فلاپ تعمیم می‌یابد. جدول تحریک برای فلیپ فلاپ بکار رفته می‌تواند از جدول ۳-۱ بدست آید. از اطلاعات موجود در این جدول و با بررسی انتقال حالت فعلی به حالت بعدی در جدول حالت، اطلاعات مربوط به شرایط ورودی فلیپ فلاپها را در جدول تحریک بدست می‌آوریم.

جدول درستی بخش ترکیبی مدار ترتیبی در جدول تحریک موجود است. ستون‌های حالت فعلی و ورودی‌ها، ورودی‌های جدول درستی را تشکیل می‌دهند. وضعیت ورودی فلیپ فلاپها خروجی‌های جدول درستی را تشکیل می‌دهند. با استفاده از مفهوم ساده سازی بکمک نقشه، ما مجموعه‌ای از

معادلات ورودی فلیپ فلاپها را برای مدار ترکیبی بدست می‌آوریم. هر معادله ورودی فلیپ فلاپ یک نمودار منطقی را مشخص می‌کند که خروجی آن باید به یکی از ورودی‌های فلیپ فلاپها وصل شود. مدار ترکیبی حاصل، همراه با فلیپ فلاپها، مدار ترتیبی را تشکیل می‌دهد.

اغلب، خروجی فلیپ فلاپها بعنوان بخشی از خروجی‌های مدار ترتیبی تصور می‌شوند. با این وجود مدار ترکیبی ممکن است خروجی خارجی هم داشته باشد. در چنین حالتی توابع بولی خروجی‌های خارجی از جدول حالت با کمک تکنیک‌های طراحی بدست می‌آیند.

مجموعه‌ای از معادلات ورودی فلیپ فلاپها یک مدار ترتیبی را بفرم جبری مشخص می‌کنند. روش یافتن دیاگرام منطقی از مجموعه معادلات ورودی فلیپ فلاپها یک روند مستقیم و سر راست است. ابتدا تمام ورودی‌ها و خروجی‌ها را نام‌گذاری می‌کنیم. سپس مدار ترکیبی را با توجه به معادلات بولی حاصل از معادلات ورودی فلیپ فلاپها رسم می‌نمائیم. و بالاخره خروجی‌های فلیپ فلاپها را به ورودی‌های مدارت ترکیبی و خروجی‌های مدار ترکیبی را به ورودی‌های فلیپ فلاپها وصل می‌نمائیم.

مسائل

۱-۱ با استفاده از جدول درستی صحت تئوری دومورگان را برای سه متغیر تحقیق کنید:

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

۱-۲ جدول درستی یک تابع OR انحصاری (تابع فرد) سه متغیر را تشکیل دهید:

$$x = A \oplus B \oplus C$$

۱-۳ عبارات زیر را با استفاده جبر بول ساده کنید:

(الف) $A + AB$ (ب) $AB + AB'$

(ج) $A'BC + AC$ (د) $A'B + ABC' + ABC$

۱-۴ عبارات زیر را با استفاده از جدول بول ساده کنید:

(الف) $AB + A(CD + CD')$ (ب) $(BC' + A'D)(AB' + CD')$

۱-۵ با استفاده از تئوری دمورگان نشان دهید که:

(الف) $(A + B)'(A' + B')' = 0$ (ب) $A + A'B + A'B' = 1$

۱-۶ با فرض عبارت بولی $F = x'y + xyz'$:

(الف) یک عبارت جبری برای متمم، F' بدست آورید.

(ب) نشان دهید که $FF' = 0$ (ج) نشان دهید که $F + F' = 1$

۱-۷ با فرض تابع بول

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

(الف) جدول درستی تابع را بدست آورید.
(ب) دیاگرام منطقی را با استفاده از عبارت بولی اصلی رسم کنید.
(ج) عبارت جبری را با بکارگیری جبر بول ساده کنید.
(د) جدول درستی را با استفاده از عبارت ساده شده تابع تشکیل داده و نشان دهید که همان جدول بخش (الف) است.
(هـ) دیاگرام منطقی را با استفاده از عبارت ساده شده بدست آورده و تعداد کل گیت‌ها را با دیاگرام بخش (ب) مقایسه کنید.

۱-۸ توابع بولی زیر را با استفاده از نقشه سه متغیره ساده کنید.

(الف) $F(x, y, z) = \sum (0, 1, 5, 7)$ (ب) $F(x, y, z) = \sum (1, 2, 3, 6, 7)$
(ج) $F(x, y, z) = \sum (3, 5, 6, 7)$ (د) $F(x, y, z) = \sum (0, 2, 3, 4, 6)$

۱-۹ توابع بولی زیر را با استفاده از نقشه چهار متغیره ساده کنید:

(الف) $F(A, B, C, D) = \sum (4, 6, 7, 15)$
(ب) $F(A, B, C, D) = \sum (3, 7, 11, 13, 14, 15)$
(ج) $F(A, B, C, D) = \sum (0, 1, 2, 4, 5, 7, 11, 15)$
(د) $F(A, B, C, D) = \sum (0, 2, 4, 5, 6, 7, 8, 10, 13, 15)$

۱-۱۰ عبارات زیر را ۱) بصورت جمع حاصلضربها و ۲) بصورت ضرب حاصل‌جمع‌ها ساده کنید:

(الف) $x'z' + y'z' + yz' + xy$
(ب) $AC' + B'D + A'CD + ABCD$

۱-۱۱ توابع بولی زیر را بشکل جمع حاصلضرب‌ها با استفاده از نقشه چهار متغیره ساده کنید. دیاگرام منطقی را، الف) با گیت‌های AND–OR؛ ب) با گیت‌های NAND رسم کنید.

$$F(A, B, C, D) = \sum (0, 2, 8, 9, 10, 11, 14, 15)$$

۱-۱۲ تابع بولی زیر را بشکل ضرب حاصل جمع‌ها با استفاده از نقشه چهار متغیره ساده کنید. دیاگرام منطقی را، الف) با گیت‌های OR–AND؛ ب) با گیت‌های NOR رسم کنید.

$$F(w, x, y, z) = \sum (2, 3, 4, 5, 6, 7, 11, 14, 15)$$

۱-۱۳ تابع بولی F را همراه با حالات بی‌اهمیت d در فرم ۱) جمع‌حاصلضرب‌ها و ۲) ضرب حاصلجمع‌ها ساده کنید.

$$F(w, x, y, z) = \sum (0, 1, 2, 3, 7, 8, 10)$$
$$F(w, x, y, z) = \sum (5, 6, 11, 15)$$

۱-۱۴ با استفاده از جدول ۲-۱ عبارت بولی S (جمع) خارج شده از یک تمام جمع کننده بشکل جمع حاصلضرب‌ها را بدست آورید. سپس با دستکاری‌های جبری نشان دهید که S می‌تواند بصورت OR انحصاری سه متغیره بیان شود.

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

۱-۱۵ یک مدار ترتیبی بنام تابع اکثریت بدین صورت تعریف می‌شود که هرگاه مقدار متغیرهای ورودی 1 بیشتر از ورودی‌های 0 باشند خروجی 1 است. تابع اکثریت سه ورودی را طراحی کنید.

۱-۱۶ یک مدار ترکیبی با سه متغیر x، y و z و سه خروجی A، B و C طراحی کنید. اگر ورودی‌های دودویی 0، 1، 2 یا 3 باشد خروجی یکی بیشتر از ورودی است. وقتی ورودی‌ها 4، 5، 6 و 7 باشند خروجی دودویی یکی کمتر از ورودی باشند.

۱-۱۷ نشان دهید که یک فلیپ فلاپ JK را با قرار دادن یک معکوس کننده بین ورودی‌های J و K می‌توان به یک فلیپ فلاپ D تبدیل کرد.

۱-۱۸ با استفاده از اطلاعات موجود در جدول مشخصه فلیپ فلاپ JK در جدول شکل ۲-۱ (ب) جدول تحریک را برای فلیپ فلاپ JK بدست آورده و جواب خود را با جدول ۳-۱ مقایسه کنید.

۱-۱۹ یک مدار ترتیبی دارای دو فلیپ فلاپ A و B، دو ورودی x و y و یک خروجی z است. معادلات ورودی فلیپ فلاپ و نیز خروجی مدار بقرار زیرند:

$$D_A = x'y + xA$$
$$D_B = x'B + xA$$
$$z = B$$

(الف) دیاگرام منطقی را برای مدار رسم کنید
(ب) جدول حالت را برپا کنید.

۱-۲۰ یک پایین شمار دو بیتی را طراحی کنید، این یک مدار ترتیبی دارای دو فلیپ فلاپ و یک ورودی x می‌باشد. وقتی $x = 0$ است حالت فلیپ فلاپ تغییر نمی‌کند. وقتی $x = 1$ باشد، ترتیب حالات 11، 10، 01، 00 و 11 و تکرار آن خواهد بود.

۱-۲۱ یک مدار ترتیبی با دو فلیپ فلاپ JK و دو ورودی E و x طراحی کنید. اگر $E = 0$ باشد، مدار بدون توجه به مقدار x در همان حالت باقی می‌ماند. وقتی $E = 1$ و $x = 1$ است مدار به یک سری حالات از 00 به 01، به 10، به 11 و 00 و تکراری از آن وارد می‌شود. وقتی $E = 1$ و $x=0$ باشد مدار به حالات انتقالی 00 به 11، به 10، به 01 و 00 و تکراری از آن وارد می‌شود.

۲ قطعات دیجیتال

۲-۱ مدارهای مجتمع
۲-۲ دیکدرها
۲-۳ مولتی پلکسرها
۲-۴ ثبات‌ها
۲-۵ شیفت رجیسترها
۲-۶ شمارنده‌های دودویی
۲-۷ واحد حافظه

۲-۱ مدارهای مجتمع[1]

مدارهای دیجیتال با مدارهای مجتمع ساخته می‌شوند. یک مدار مجتمع (یا آی سی)، یک کریستال کوچک نیمه هادی بنام تراشه[2] است که قطعات الکترونیکی را برای گیت‌های دیجیتال در خود دارد. اتصالات داخل تراشه، مدار مورد نیاز را بوجود می‌آورد. تراشه در داخل یک محفظه پلاستیک و یا سرامیک جاسازی می‌شود و اتصالات آن با سیم‌های طلایی نازک به پایه‌های خارجی جوش داده می‌شود تا مدار مجتمع بوجود آید. تعداد پایه‌ها ممکن است از 14 پایه در بسته‌های کوچک تا 100 پایه یا بیشتر در بسته‌های بزرگتر تغییر کند. هر مدار مجتمع یا آی سی دارای یک مشخصه عددی است که روی سطح بسته‌بندی آن برای شناسایی چاپ می‌شود. هر سازنده یک کتابچه راهنما یا کاتالوگ با شرح دقیق و تمام اطلاعات لازم درباره آی سی های ساخت خود را چاپ می‌کند.

1- Integrated Circuit (IC) 2– Chip

با پیشرفت تکنولوژی مدارهای مجتمع، تعداد گیت‌هایی که می‌توانست در یک تراشه جای گیرد به میزان قابل ملاحظه‌ای افزایش یافت. تراشه‌هایی که دارای چند گیت داخلی بودند و آن دسته که چند صد گیت را دارا بودند در بسته‌هایی با ظرفیت یا مقیاس کوچک، متوسط و یا بزرگ جای داده شدند.

مدارهای مجتمع با مقیاس کوچک[1] (SSI) دارای چند گیت مستقل در یک بسته واحد هستند. ورودی‌ها و خروجی‌های گیت‌ها مستقیماً به پایه‌های بسته متصل‌اند. تعداد گیت‌ها معمولاً کمتر از ۱۰ و محدود به تعداد پایه‌ها در IC می‌باشد.

قطعات مجتمع با مقیاس متوسط[2] (MSI) دارای تقریباً ۱۰ الی ۲۰۰ گیت در هر بسته می‌باشند. این وسیله‌ها معمولاً توابع دیجیتال ساده همچون دیکدرها، جمع‌کننده‌ها و ثبات‌ها را اجرا می‌نمایند.

مدارها یا وسایل مجتمع با مقیاس بزرگ[3] (LSI) بین ۲۰۰ تا چند هزار گیت در هر بسته دارند. این بسته‌ها سیستم‌های دیجیتالی همچون پردازنده‌ها، تراشه‌های حافظه و ماژول‌های قابل برنامه‌ریزی را شامل می‌شوند. قطعات مجتمع با مقیاس بسیار بزرگ[4] (VLSI) حاوی هزاران گیت در یک بسته‌اند. مثالهایی از این گروه عبارتنداز، آرایه‌های بزرگ حافظه و تراشه‌های پیچیده ریز کامپیوترها. VLSIها بدلیل کوچکی و ارزانی انقلابی در تکنولوژی ساخت سیستم‌های کامپیوتری بوجود آورده و به طراحان امکان ساخت و ایجاد ساختارهایی را دادند که قبلاً اقتصادی نبودند.

مدارهای مجتمع دیجیتال نه تنها براساس عملکرد منطقی‌شان طبقه‌بندی می‌شوند بلکه از نظر تکنولوژی خاص مدارهایی که به آن تعلق دارند نیز دسته‌بندی می‌گردند. تکنولوژی بکار رفته در مدار را خانواده منطقی دیجیتال[5] می‌خوانند. هر خانواده منطقی، مدار الکترونیکی پایه خاصی را داراست که مدارها و توابع دیجیتال پیچیده‌تر براساس آن تهیه می‌شوند. مدارپایه در هر تکنولوژی یک گیت NAND، NOR و یا معکوس کننده است. در نام گذاری تکنولوژی، از قطعات الکترونیک بکار رفته در ساخت مدار پایه معمولاً استفاده می‌شود. بسیاری از خانواده‌های مختلف منطقی بصورت مدارهای مجتمع در سطح تجاری عرضه شده‌اند. متداول‌ترین خانواده‌ها در زیر معرفی شده‌اند.

TTL – منطق ترانزیستور – ترانزیستور[6]
ECL – منطق کوپل امیتر[7]
MOS – منطق فلز – اکسید – نیمه هادی[8]
CMOS – منطق فلز – اکسید – نیمه هادی مکمل[9]

1- Small Scale Integration
2- Medium Scale Integration
3- Large Scale Integration
4- Very Large Scale Integration
5- Digital Logic Family
6- Transistor – Transistor Logic
7- Emitter Coupled Logic
8- Metal – Oxide Semiconductor
9- Complementary Metal Oxide Semiconductor

TTL یک خانواده متداول است که سالها مورد استفاده بوده و بعنوان استاندارد تلقی می‌شود. ECL در سیستم‌هایی که به سرعت عمل بالا نیاز دارند ترجیح داده می‌شود. MOS برای مدارهایی که نیاز به تراکم بالا دارند مناسب است، و CMOS در سیستم‌های کم مصرف بکار می‌رود.

خانواده منطقی ترانزیستور - ترانزیستور گونه تکامل یافته تکنولوژی قدیمی‌تری است که در آن از دیود و ترانزیستور برای ساخت گیت پایه NAND استفاده می‌شده است. این تکنولوژی منطق دیود - ترانزیستور[1] (DTL) خوانده می‌شده است. بعدها برای بهبود عملکرد مدار بجای دیود از ترانزیستور استفاده شد و نام خانواده جدید، منطق ترانزیستور - ترانزیستور گذاشته شد. بهمین دلیل نام "ترانزیستور" در این عبارت دو بار تکرار شده است. علاوه بر نوع استاندارد TTL، انواع دیگری از این خانواده عبارتند از: TTL سرعت بالا،[2] TTL توان پائین[3] (یا کم مصرف)، TTL شوتکی[4]، TTL شوتکی توان پایین[5] و TTL شوتکی پیشرفته[6]. منبع تغذیه مدارهای TTL، ۵ ولت است و دوسطح منطقی تقریباً 0 و ۳/۵ ولت می‌باشند.

خانواده کوپل امیتر سریع‌ترین مدارهای دیجیتال را بفرم مجتمع در اختیار می‌گذارد. ECL در مدارهایی مانند سوپر کامپیوترها و پردازنده‌های سیگنال که در آنها سرعت بالا ضرورت دارد، بکار می‌رود. ترانزیستورها در گیت‌های ECL در حالت غیر اشباع کار می‌کنند و رسیدن به تأخیرهای انتشاری در حد ۱ تا ۲ نانو ثانیه در آنها میسّر است.

منطق فلز - اکسید - نیمه‌هادی (MOS) یک ترانزیستور تک قطبی است که به جریان یک نوع حامل الکتریکی وابسته است. این حامل‌ها ممکن است الکترون (در نوع کانال n) یا حفره (در نوع کانال P) باشند. این، برخلاف ترانزیستور دوقطبی بکار رفته در گیت‌های TTL و ECL است، که در حین عملکرد هر دو نوع حامل در آن وجود دارند. یک MOS کانال p را PMOS و یک MOS کانال n را NMOS می‌نامند. معمولاً در مدارهایی که فقط یک نوع ترانزیستور MOS وجود دارد از NMOS استفاده می‌شود. در تکنولوژی CMOS هردو نوع ترانزیستور PMOS و NMOS که بشکل مکمل در تمام مدارها بسته شده‌اند بکار رفته است. بزرگترین مزیت CMOS نسبت به دو قطبی تراکم بالای مدارها در بسته‌بندی، ساده بودن تکنیک ساخت و عملکرد مقرون به صرف آن بدلیل مصرف توان کم است.

بعلت مزایای بی‌شمار، مدارهای مجتمع انحصاراً در تهیه انواع قطعات لازم در طراحی سیستم‌های کامپیوتر به کار می‌روند. برای درک سازمان و طراحی کامپیوترها، آشنایی با انواع قطعات و اجزاء بکار رفته در مدارهای مجتمع اهمیت دارد. به این دلیل، اجزاء اصلی به همراه خواص منطقی آنها تشریح شده است. این اجزاء مجموعه‌ای از واحدهای عملیاتی دیجیتال را فراهم می‌کنند که در طراحی کامپیوترهای دیجیتال بعنوان بلاک‌های ساختمانی اصلی (پایه) به کار می‌روند.

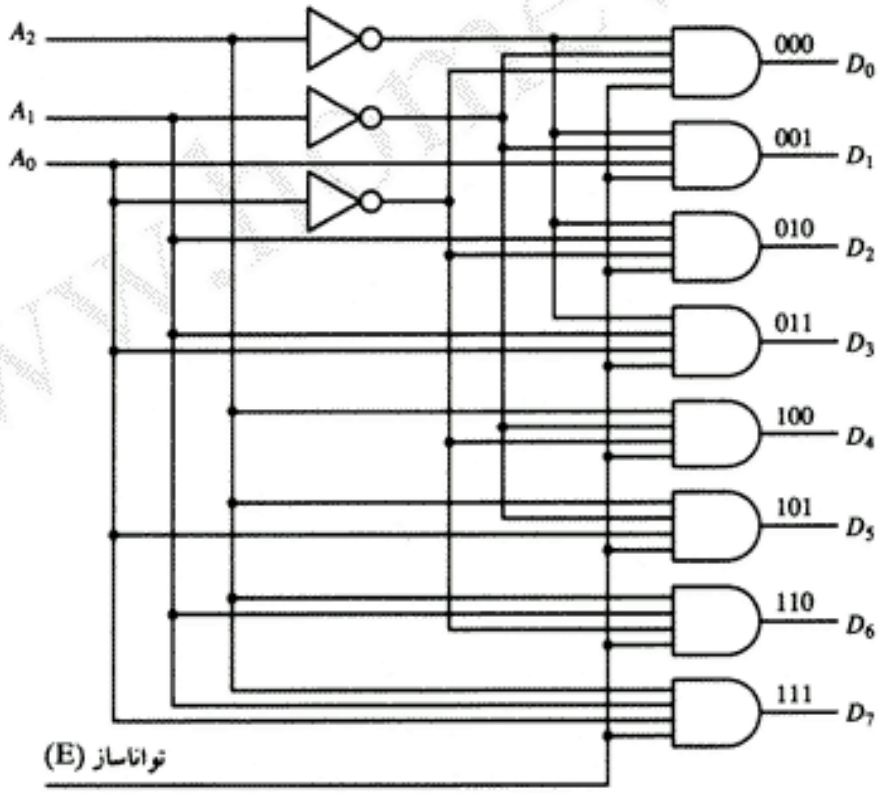
1- Diode –Transistor Logic 2- High Speed TTL 3- Low Power TTL
4- Schottky TTL 5- Low – Power Schottky TTL
6- Advanced Schottky TTL

۲-۲ دیکدرها[1]

کمیت‌های اطلاعاتی گسسته در سیستم‌های کامپیوتر دیجیتال با کدهای دودویی نشان داده می‌شوند. یک کد دودویی ۸ بیتی قادر است 2^n عنصر مستقل اطلاعاتی کد شده را نمایش دهد. یک دیکدر مداری است ترکیبی که اطلاعات دودویی را از طریق n خط ورودی دریافت نموده و آنها را به حداکثر 2^n خط خروجی مستقل منحصر بفرد تبدیل می‌کند. اگر اطلاعات دیکد شده n بیتی ترکیبات بلااستفاده یا بی‌اهمیت داشته باشد در اینصورت دیکدر دارای خروجی‌های کمتر از 2^n خواهد بود.

دیکدرهائی که در این بخش مورد بحث قرار گرفته‌اند دیکدرهای n به m خطی نام دارند که در آن $m \leq 2^n$ است. هدف از بکارگیری آنها تولید 2^n ترکیب دودویی از n متغیر ورودی است. دیکدر دارای n ورودی و m خروجی بوده و دیکدر n×m نیز نامیده می‌شود.

دیاگرام منطقی یک دیکدر 3 به 8 در شکل ۱-۲ نشان داده شده است. سه ورودی داده A_0، A_1 و A_2 به هشت خروجی داده تبدیل شده‌اند که هر خروجی یکی از ترکیبات سه متغیر ورودی دودویی را نشان



شکل ۱-۲ دیکدر 3 به 8 خطی

1- Decoder

می‌دهد. سه معکوس کننده، متمم ورودی‌ها را فراهم می‌کنند و هر یک از هشت گیت AND یکی از ترکیبات دودویی را تولید می‌نماید. کاربرد خاصی از این دیکدر تبدیل دودویی به هشتایی[1] است. متغیرهای ورودی یک عدد دودویی را نشان می‌دهند و خروجی‌ها، هشت رقم یک عدد سیستم هشتایی را نمایش می‌دهند. با این وجود یک دیکدر 3 به 8 خطی می‌تواند برای دیکد کردن هر کد سه بیتی بکار رفته و هشت خروجی راکه هر بار یکی از ترکیبات کد دودویی است، فراهم آورد.

دیکدرهای تجاری دارای یک یا چند ورودی تواناسازی[2] (فعال سازی) برای کنترل کار مدار هستند. دیکدر شکل ۱-۲ دارای یک ورودی تواناساز، E می‌باشد. دیکدر هنگامی فعال شده است که E برابر 1 باشد و اگر E مساوی 0 بود دیکدر ناتوان (غیرفعال) شده است. عملکرد دیکدر با استفاده از یک جدول درستی که در جدول ۱-۲ دیده می‌شود آشکارتر خواهد شد. هر گاه ورودی تواناساز E برابر 0 باشد، تمام خروجی‌ها، بدون توجه به مقادیر سه ورودی داده، 0 خواهند شد. سه علامت × در جدول حالات‌بی اهمیت ورودی‌ها را نشان می‌دهند. اگر ورودی تواناساز، 1 شود، دیکدر در وضع عادی خودکار خواهد کرد. برای هر ترکیب ممکن ورودی، هفت خروج 0 و فقط یک خروجی 1 وجود دارد که بیانگر عدد هشتایی معادل عدد دودویی در خطوط ورودی داده است.

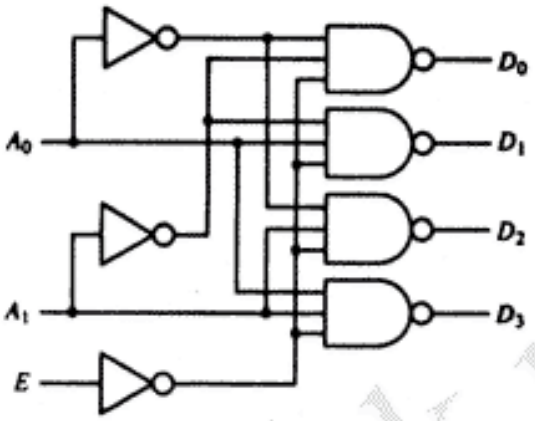
دیکدر باگیت NAND

بعضی از دیکدرها بجای گیت AND با گیت NAND ساخته می‌شوند. چون گیت NAND عمل گیت AND را با خروجی معکوس شده انجام می‌دهد، تولید خروجی‌های معکوس شده با دیکدر مقرون به صرفه‌تر است. یک دیکدر 2 به 4 همراه با ورودی توانا ساز که از گیت‌های NAND ساخته شده است در شکل ۲-۲ نشان داده شده است. مدار دارای خروجی‌های متمم بوده و با ورودی تواناساز متمم شده کار می‌کند. دیکدر هنگامی تواناست که E برابر 0 شود. همانطور که از جدول درستی

جدول ۱-۲ جدول درستی برای دیکدر 3 به 8 خطی

تواناساز	ورودی‌ها			خروجی‌ها							
E	A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	×	×	×	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0

1- Octal
2- Enable



(الف) دیاگرام منطقی

E	A_1	A_0	D_0	D_1	D_2	D_3
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0
1	×	×	1	1	1	1

(ب) جدول درستی

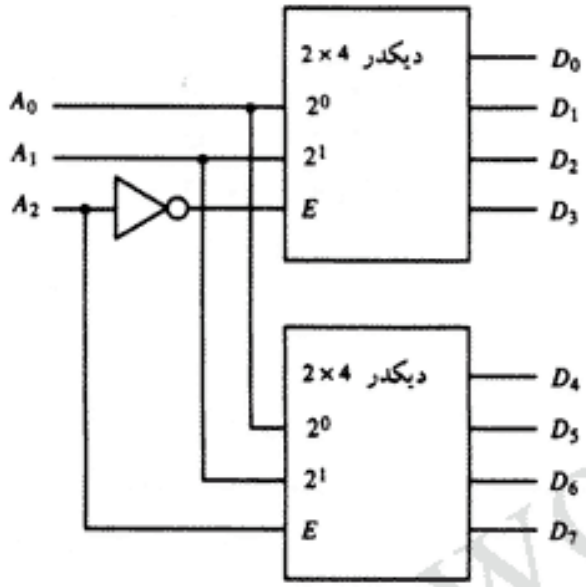
شکل ۲-۲ دیکدر 2 به 4 خطی با گیت‌های NAND

پیداست، تنهادر هر لحظه از زمان فقط یک خروجی برابر 0، و بقیه سه خروجی دیگر برابر 1 هستند. خروجی که نماینده عدد دودویی واقع در ورودی‌های A_0 و A_1 است دارای مقدار 0 است. هر گاه E برابر 1 باشد، مدار صرف نظر از مقدار دودویی ورودی‌های دیگر غیرفعال خواهد بود. وقتی که مدار غیرفعال است،هیچ یک از خروجی‌ها انتخاب نمی‌شود و تمام خروجی‌ها برابر 1 خواهند بود. بطور کلی، یک دیکدر ممکن است با خروجی‌های متمم شده یا نشده کار کند. ورودی تواناساز هم ممکن است با یک سطح سیگنال 0 یا 1 فعال شود. برخی دیکدرها دارای دو یا چند ورودی تواناساز هستند و برای فعال شدن مدار شرایط منطقی معینی باید در آنها برقرار باشد.

گسترش دیکدر

مواردی پیش می‌آید که دیکدری با اندازه مشخصی مورد نیاز است ولی تنها اندازه کوچکتر آن موجود است در چنین مواقعی می‌توان دو و یا چند دیکدر را با هم ترکیب نمود و با استفاده از ورودی‌های فعال‌ساز (تواناساز) دیکدر بزرگتری بوجود آورد. بنابراین اگر یک دیکدر 6 به 64 مورد نیاز باشد می‌توان آن را با چند دیکدر 4 به 16 تشکیل داد.

شکل ۳-۲ نشان می‌دهد که چگونه از اتصال دیکدرهای دارای ورودی‌های تواناساز می‌توان دیکدرهای بزرگتری ساخت. در این شکل دو دیکدر 2 به 4 با هم ترکیب شده‌اند تا یک دیکدر 3 به 8 تشکیل شود. دو بیت کم ارزشتر ورودی‌ها به هر دو دیکدر متصل شده‌اند. بیت پرارزشتر به ورودی تواناساز یکی از دیکدرها و از طریق یک معکوس کننده به ورودی تواناساز دیکدر دیگر وصل است. فرض براین است که دیکدر وقتی که E=1 است فعال شود. اگر E=0 باشد دیکدر غیرفعال و تمام خروجی‌های آن در سطح 0 است. هرگاه A_2=0 باشد، دیکدر بالایی فعال و دیکدر پائینی غیرفعال است. خروجی‌های



شکل ۳-۲ یک دیکدر 8×3 (یا 3 به 8) ساخته شده با دو دیکدر 2×4

دیکدر تحتانی همگی غیرفعال و در وضعیت 0 می‌باشند. خروجی‌های دیکدر فوقانی خروجی‌های D_0 الی D_3 را، بسته به مقادیر A_1 و A_0، تولید می‌کنند (در این حالت A_2=0 است). اگر A_2=1 باشد، دیکدر پائینی فعال و دیکدر بالایی غیرفعال است. دیکدر پائینی در این وضعیت معادل دودویی D_4 تا D_7 را تولید می‌کند زیرا این عددهای دودویی رقم 1 را در مکان A_2 دارا هستند.

مثال فوق فوائد ورودی فعال‌ساز (تواناساز) را در دیکدرها یا هرمدارمنطقی ترکیبی نشان داد. ورودی‌های تواناساز ابزار مناسبی برای اتصال دو یا چند مدار به منظورگسترش ورودی‌ها و خروجی‌ها می‌باشند.

انکدرها[1]

انکدر (کدگذار) یک مدار دیجیتال است که عکس عمل دیکدر را انجام می‌دهد. یک انکدر دارای 2^n (یا کمتر) خط ورودی و n خط خروجی است. خطوط خروجی کد دودویی متناظر مقدار ورودی را تولید می‌کنند. مثالی از یک انکدر، انکدر هشت هشتی به دودویی است، که جدول درستی آن در جدول ۲-۲ داده شده است. این مدار دارای هشت ورودی، که هر یک به یک رقم هشت هشتی (هشتایی) متعلق بوده و سه خروجی که عدد دودویی متناظر را تولید می‌کنند، است. فرض براین است که فقط یک ورودی دارای مقدار 1 در هر لحظه از زمان است؛ در غیر این صورت مدار مفهومی ندارد.

انکدر را می‌توان با گیت‌های OR، که ورودی‌هایشان مستقیماً از جدول درستی تعیین می‌شوند،

1- Encoder

پیاده‌سازی کرد. خروجی A_0 بشرطی 1 است که رقم ورودی هشت هشتی 1 یا 3 یا 5 یا 7 باشد. شرایط مشابهی برای دو خروجی دیگر برقرار است. این شرایط را می‌توان بوسیله توابع بولی زیر بیان کرد.

$$A_0 = D_1 + D_3 + D_5 + D_7$$
$$A_1 = D_2 + D_3 + D_6 + D_7$$
$$A_2 = D_4 + D_5 + D_6 + D_7$$

انکدر فوق با سه گیت OR قابل پیاده سازی است.

۳-۲ مولتی پلکسرها[1]

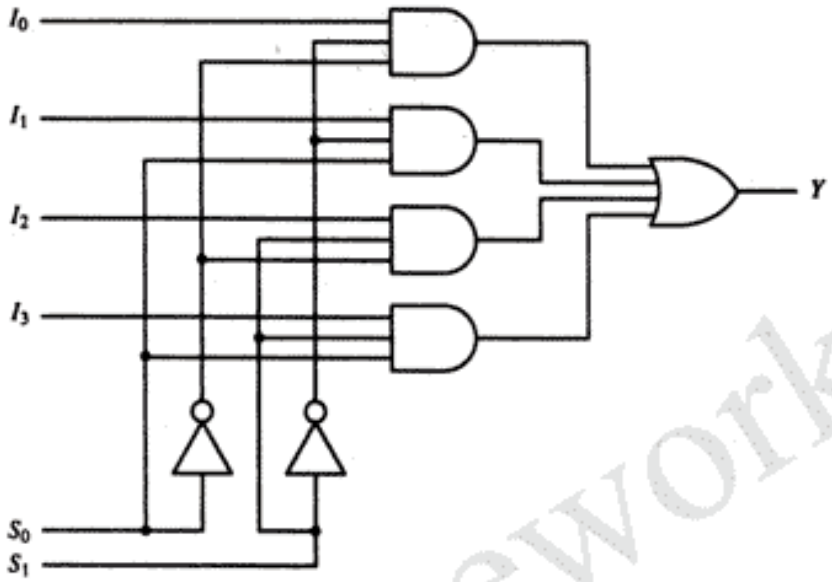
مولتی پلکسر یک مدار ترکیبی است که اطلاعات دودویی را از یکی از 2^n ورودی دریافت و آن را به یک مسیرخروجی هدایت می‌نماید. مجموعه‌ای از خطوط ورودی انتخاب[2] خط داده خاصی را برای خروجی انتخاب می‌کنند. یک مولتی پلکسر 2^n به 1 دارای 2^n خط ورودی داده و n خط ورودی انتخاب می‌باشد. ترکیب این ورودی‌ها تعیین کننده خط داده ورودی انتخاب شده برای خروجی است.

یک مولتی پلکسر 4 به 1 در شکل ۲-۴ نشان داده شده است. هر یک از چهار ورودی داده I_0 تا I_3 به ورودی یک گیت AND اعمال شده است. دو خط انتخاب S_1 و S_0 برای انتخاب یک گیت AND بخصوص دیکد شده‌اند. خروجی گیت‌های AND به ورودی‌های یک گیت OR اعمال شده تا تنها خروجی مدار را ایجاد نماید. برای درک طرز کار مدار، حالت $S_1S_0=10$ را در نظر بگیرید. دو ورودی گیت AND مربوط به ورودی I_2 در وضعیت 1 هستند. سومین ورودی این گیت به I_2 متصل است. سه

جدول ۲-۲ جدول درستی برای انکدر هشت هشتی به دودویی

ورودی‌ها								خروجی‌ها		
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	A_2	A_1	A_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

1- Multiplexer 2- selection



شکل ۴-۲ مولتی‌پلکسر 4 به 1

گیت AND دیگر، هر یک حداقل یک ورودی 0 دارند که در نتیجه آن خروجی شان 0 می‌شود. بنابراین خروجی گیت OR برابر مقدار I_2 است و باین ترتیب مسیری از ورودی‌های منتخب به خروجی ایجاد شده است.

مولتی پلکسر 4 به 1 شکل ۴-۲ دارای شش ورودی و یک خروجی است. جدول درستی که عملکرد مدار را نشان می‌دهد نیاز به 64 سطر خواهد داشت زیرا شش متغیر ورودی 2^6 ترکیب دودویی دارد. این جدول بیش از حد طولانی است و در اینجا نشان داده نخواهد شد.

روش مناسب‌تری برای بیان عملکرد مولتی پلکسر استفاده از جدول تابع است. جدول تابع برای مولتی پلکسر در جدول ۳-۲ نشان داده شده است. این جدول رابطه بین چهار ورودی داده و تنها خروجی مدار را بصورت تابعی از ورودی‌های انتخاب S_1 و S_0 نشان می‌دهد. هرگاه ورودی‌های انتخاب برابر 00 باشند، خروجی Y برابر I_0 است. اگر ورودی‌های انتخاب 01 شوند، ورودی I_1 به خروجی Y متصل است، و برای دو ورودی دیگر نیزاستدلال مشابهی وجود دارد. مولتی پلکسر را انتخاب کننده داده[1] هم می‌نامند، چون این مدار یکی از چند ورودی داده را برمی‌گزیند و اطلاعات دودویی را به خروجی هدایت می‌نماید.

گیت‌های AND و معکوس کننده‌ها در مولتی پلکسر مشابه یک مدار دیکدر می‌باشند و خطوط ورودی انتخاب را دیکد می‌کنند. بطور کلی، یک مولتی پلکسر 2^n به 1 خطی، از یک دیکدر n به 2^n

1- Data Selector (or dala switch)

جدول ۳-۲ جدول تابع برای مولتی‌پلکسر 4 به 1 خطی

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

با اضافه کردن 2^n خط ورودی به آن، یعنی یک خط از هر ورودی داده، ساخته می‌شود. اندازه[1] یک مولتی پلکسر بوسیله تعداد 2^n ورودی داده و تنها خروجی‌اش مشخص می‌شود. این اندازه بطور ضمنی بیانگر این واقعیت است که تعداد خطوط انتخاب نیز n است. مولتی پلکسر را غالباً با نام اختصاری MUX می‌خوانند.

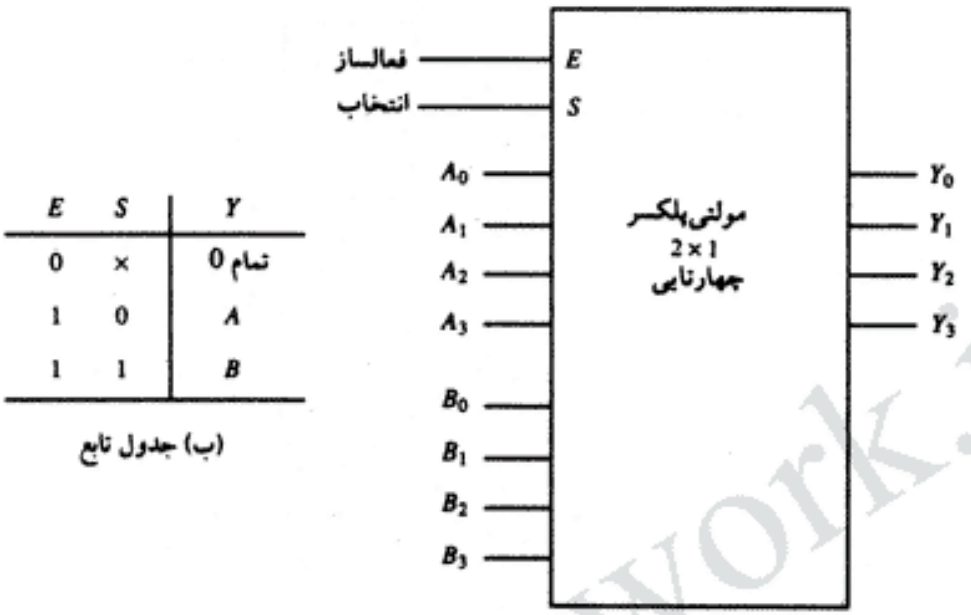
همانند دیکدرها، مولتی پلکسرها هم ممکن است دارای ورودی تواناساز باشند تا عملکرد آن را کنترل نمایند. هر وقت ورودی تواناساز در حالت غیرفعال باشد، خروجی‌ها از کار می‌افتند، و اگر در حالت فعال قرارگیرد، مدار به صورت یک مولتی پلکسر بطور عادی عمل می‌کند. ورودی تواناساز برای اتصال دو یا چند مولتی پلکسر و تشکیل یک مولتی پلکسر با تعداد ورودی‌های بیشتر مفید است.

در برخی از موارد دو یا چند مولتی پلکسر در یک بسته مدار مجتمع جای داده می‌شود. ورودی‌های انتخاب و تواناساز در ساختار چند واحدی، برای همه مولتی پلکسرها مشترک‌اند. بلاک دیاگرام یک مولتی پلکسر چهارتایی 2 به 1، بعنوان مثال در شکل ۵-۲ نشان داده شده است. مدار دارای چهار مولتی پلکسر است که هر یک قادر به انتخاب یکی از دو خط ورودی می‌باشد. می‌توان خروجی Y_0 را چنان تعیین کرد که از ورودی A_0 یا B_0 گرفته شود. بطور مشابه Y_1 می‌تواند مقدار A_1 یا B_1 را داشته باشد و الی آخر . خط ورودی انتخاب S یکی از خطوط را در هر یک از چهار مولتی پلکسر انتخاب می‌کند. ورودی تواناساز E برای عملکرد عادی فعال می‌شود. اگرچه مدار دارای چهار مولتی پلکسر است، ما می‌توانیم این تصور را داشته باشیم که مدار یکی از دو خط داده چهار بیتی را انتخاب می‌نماید. همان طور که در جدول تابع نشان داده شد، هرگاه E=1 باشد، مدار فعال می‌شود. آنگاه اگر S=0 باشد چهار ورودی A مسیری به خروجی خواهند داشت. از طرف دیگر، اگر S=1 باشد، چهار ورودی B به خروجی اعمال می‌شوند. هرگاه E=0 باشد، همه خروجی‌ها مستقل از مقدار S، تماماً 0 خواهند بود.

۴-۲ ثبات‌ها[2]

ثبات مجموعه‌ای از فلیپ فلاپ‌هاست و هر فلیپ فلاپ قادر است یک بیت از اطلاعات را در خود

1- Size 2- Register



E	S	Y
0	×	تمام 0
1	0	A
1	1	B

(ب) جدول تابع

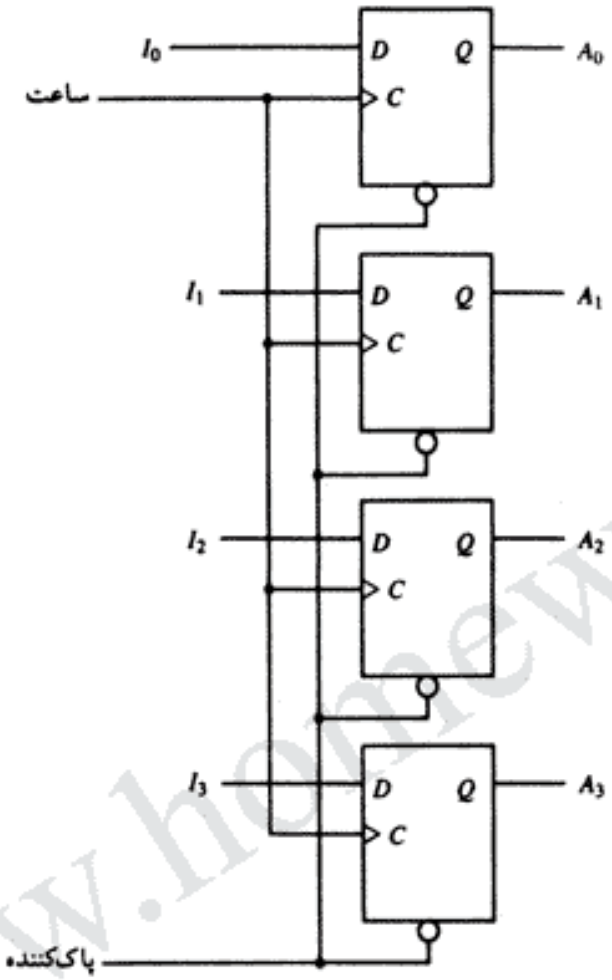
(الف) بلاک دیاگرام

شکل ۵-۲ مولتی‌پلکسر 2 به 1 چهارتایی

ذخیره کند. یک ثبات n بیتی n فلیپ فلاپ دارد و می‌تواند هر اطلاعات دودویی n بیتی را در خود ذخیره کند. یک ثبات، علاوه بر فلیپ فلاپ‌ها، ممکن است گیت‌های ترکیبی، که کارهای پردازش داده معینی را انجام می‌دهند، را هم در خود داشته باشد. یک ثبات در تعریف جامع‌تر خود متشکل از گروهی از فلیپ فلاپها و گیتهاست که بر تغییر حالت (خروجی) فلیپ فلاپها تأثیر دارند. فلیپ فلاپها اطلاعات دودویی را ذخیره می‌کنند و گیت‌ها زمان و چگونگی انتقال اطلاعات به ثبات را کنترل می‌نمایند.

انواع مختلفی از ثبات‌های تجاری وجود دارند. ساده‌ترین ثبات حاوی فلیپ فلاپها و فاقد هر نوع گیت خارجی است. شکل ۶-۲ چنین ثباتی را که از چهار فلیپ فلاپ D ساخته شده نشان می‌دهد. ورودی پالس ساعت مشترک در لبه بالارونده خود همه فلیپ فلاپ‌هارا فعال[1] می‌سازد و داده دودویی موجود در چهار ورودی بداخل ثبات چهاربیتی منتقل می‌شود. چهار خروجی را می‌توان در هر زمانی نمونه برداری کرد و اطلاعات ذخیره شده در ثبات را بدست آورد. ورودی پاک کردن[2] به پایه خاصی در هر فلیپ‌فلاپ متصل است. هرگاه این ورودی 0 شود تمام فلیپ‌فلاپ‌ها بطور غیرهمزمان (آسنکرون) و قبل از عملکرد پالس ساعت پاک[3] می‌شوند. ورودی پاک کردن بهنگام عملکرد عادی فلیپ‌فلاپ‌ها باید در منطق 1 نگهداری شوند. توجه کنید که پالس ساعت، ورودی D را توانا می‌سازد ولی ورودی پاک کردن مستقل از پالس ساعت است.

1- Trigger 2- Clear 3- Reset



شکل ۶-۲ ثبات 4 بیت

انتقال اطلاعات جدید بداخل ثبات "بارکردن"[1] ثبات نامیده می‌شود. اگر تمام بیت‌های ثبات بطور همزمان با یک انتقال پالس ساعت بار شوند، گوئیم بارشدن بصورت موازی صورت گرفته است. یک گذار یا انتقال پالس ساعت اعمال شده به ورودی‌های C در ثبات شکل ۶-۲ تمام چهار ورودی I_0 تا I_3 را به صورت موازی بار خواهد کرد. در این آرایش، اگر بخواهیم محتوای ثبات بدون تغییر باقی بماند باید پالس ساعت به مدار اعمال نگردد.

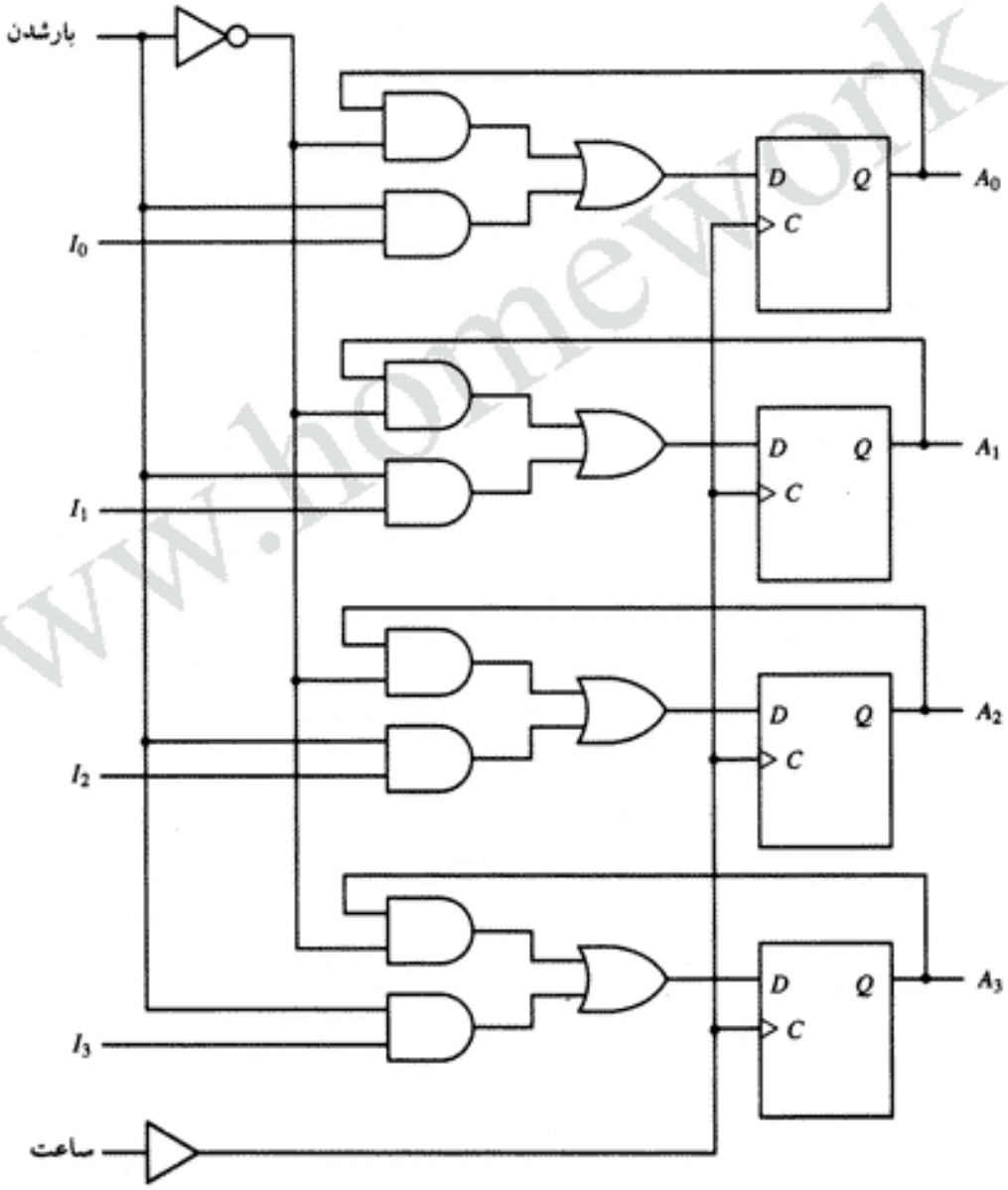
ثبات با بار شدن موازی

بسیاری از سیستم‌های دیجیتال دارای یک مدار ساعت اصلی می‌باشند که رشته‌ای از پالس ساعت را

1- Loading

فراهم می‌نماید. این پالس‌های ساعت به تمام فلیپ فلاپها و ثبات‌ها در سیستم اعمال می‌شوند. ساعت اصلی مانند پمپی که ضربان ثابتی را به تمام بخش‌ها ارسال کند عمل می‌نماید. سیگنال کنترل جداگانه را باید بکار برد تا در مورد تأثیر پالس ساعت خاص بر روی ثبات خاص تصمیم‌گیری نماید.

یک ثبات چهار بیتی با ورودی کنترل بارکردن که از طریق گیت‌ها به ورودی‌های D می‌رسند در شکل ۷-۲ نشان داده شده‌اند. ورودی‌های C پالس‌های ساعت را در همه زمانها دریافت می‌کنند. گیت بافر در ورودی ساعت توان دریافتی لازم را از مولد ساعت کاهش می‌دهد. هرگاه پالس ساعت فقط به ورودی یک گیت در عوض چند گیت وصل شود توان کمتری لازم است ولی چنانچه این گیت بافر مورد



شکل ۷-۲ یک ثبات 4 بیت با بارشدن موازی

استفاده قرارنگرفته و پالس ساعت به هر چهار ورودی مستقیماً وصل شود توان بیشتری از مولد پالس اخذ خواهد شد.

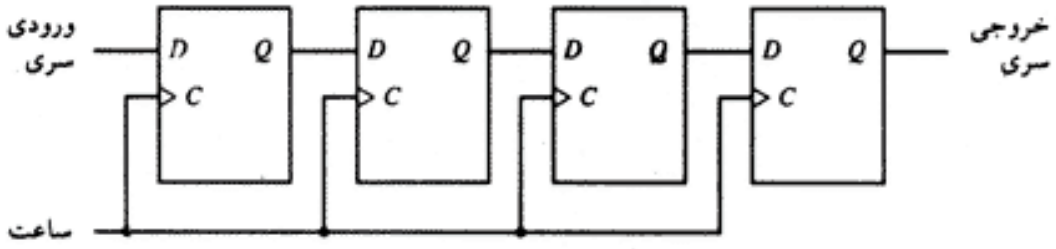
ورودی بارکردن عملی را که با هر پالس ساعت در ثبات انجام می‌شود تعیین می‌کند. هرگاه ورودی بارکردن برابر 1 باشد، داده‌های چهار ورودی با لبه مثبت پالس ساعت (گذار مثبت) به داخل ثبات انتقال می‌یابند. اگر ورودی باردهی 0 باشد، ورودی‌های داده ممانعت شده و ورودی‌های فلیپ‌فلاپ D به خروجی‌های آن وصل می‌شوند. اتصال برگشت[1] (پسخور) از خروجی به ورودی لازم است زیرا فلیپ‌فلاپ D حالت بلاتغییر ندارد. با هر پالس ساعت ورودی D حالت بعدی خروجی را معین می‌کند. برای حفظ خروجی و عدم تغییر آن، باید ورودی D را با خروجی Q برابر کنیم.

توجه کنید که پالس‌های ساعت همیشه به ورودی‌های C اعمال می‌شوند. ورودی بارکردن تعیین می‌کند که پالس بعدی موجب پذیرش اطلاعات شود یا اطلاعات موجود در ثبات بدون تغییر باقی بماند. انتقال اطلاعات از ورودی به ثبات برای هر چهار بیت در یک گذر پالس ساعت بطور همزمان صورت می‌گیرد.

۵-۲ شیفت رجیسترها[2]

ثباتی که قادر است اطلاعات دودویی را در یک یا دو جهت انتقال دهد شیفت رجیستر (یا ثبات انتقال) نامیده می‌شود. آرایش منطقی یک شیفت رجیستر از زنجیره‌ای از فلیپ فلاپهای متوالی تشکیل شده، که در آن خروجی یک فلیپ فلاپ به ورودی فلیپ فلاپ بعدی متصل شده است. تمام فلیپ‌فلاپ‌ها پالس ساعت مشترکی را که موجب جابجایی یک طبقه به طبقه بعدی است دریافت می‌کنند.

ساده‌ترین شیفت رجیستر ممکن، فقط از فلیپ فلاپ‌ها استفاده می‌کند، شکل ۸-۲. خروجی هر فلیپ فلاپ به ورودی D فلیپ فلاپ سمت راستی وصل شده است. پالس ساعت برای همه آنها مشترک است. ورودی سری تعیین کننده آنچه به سمت چپ‌ترین فلیپ فلاپ وارد می‌شود می‌باشد. خروجی سری نیز از سمت راست‌ترین فلیپ فلاپ دریافت می‌شود.



شکل ۸-۲ شیفت رجیستر 4 بیت

1- Feedback
2- Shift Register

گاهی لازم است شیفت را چنان کنترل کنیم که با وقوع پالس‌های معینی و نه همه پالس‌ها بوقوع بپیوندد. این عمل با ممانعت از اعمال پالس ساعت به ورودی ثبات میسر است. اگر شیفت رجیستر شکل ۸-۲ بکار رود، عمل شیفت را می‌توان با اتصال ساعت به ورودی یک گیت AND کنترل کرد. در این صورت ورودی دوم این گیت می‌تواند جابجایی را با راه دادن و یا راه ندادن آن کنترل نماید. همچنین می‌توان با اضافه کردن مدارهایی، عمل جابجایی را از طریق ورودی‌های D فلیپ فلاپ‌ها کنترل نمود و نه از طریق ورودی ساعت.

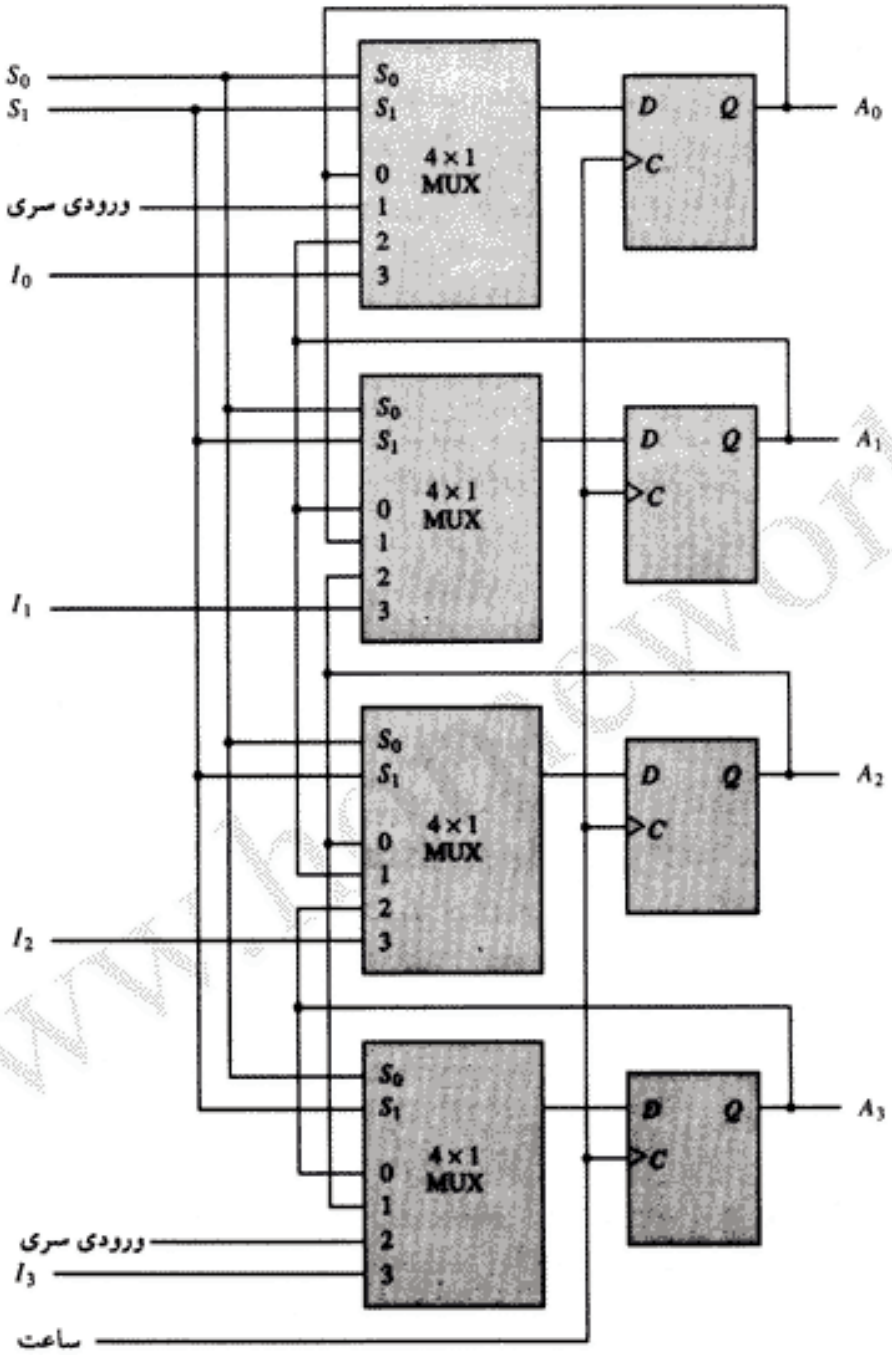
شیفت رجیستر دو جهته با بارشدن موازی

ثباتی که بتواند عمل شیفت را فقط در یک جهت انجام دهد ثبات شیفت رجیستر یک طرفه[1] خوانده می‌شود. ثباتی که قادر باشد شیفت یا جابجایی را درهر دو جهت انجام دهد شیفت رجیستر دو جهته نامیده می‌شود. برخی شیفت رجیسترها پایانه‌های ورودی و خروجی لازم را برای انتقال موازی نیز در اختیار دارند. کامل‌ترین شیفت رجیستر تمام توانایی‌های ذکر شده زیر را داراست. ثبات‌های دیگر ممکن است برخی از این قابلیت‌ها را داشته باشند. البته هر شیفت رجیستر حداقل یک عمل شیفت را انجام می‌دهد.

۱- یک ورودی پالس‌های ساعت برای همزمانی تمام عملیات
۲- عمل شیفت به راست و یک خط ورودی سری برای شیفت به راست
۳- عمل شیفت به چپ و یک خط ورودی سری مربوط به شیفت به چپ
۴- عمل بارشدن موازی و n خط ورودی برای این انتقال موازی
۵- n خط خروجی موازی
۶- یک حالت کنترلی که با وجود اعمال مداوم پالس‌های ساعت، اطلاعات موجود در ثبات را بدون تغییر باقی گذارد.

یک شیفت رجیستر چهار بیتی با بارشدن موازی در شکل ۹-۲ نشان داده شده است. هر طبقه از یک فلیپ‌فلاپ D و یک مولتی پلکسر 4×1 تشکیل شده است. دو ورودی انتخاب S_1 و S_0 یکی از ورودی‌های داده مولتی پلکسر را برای فلیپ فلاپ D انتخاب می‌کند. خطوط انتخاب شده عملکرد ثبات را طبق جدول تابع ۴-۲ کنترل می‌کند. هرگاه ورودی‌های کنترل $S_1\ S_0 = 00$ باشند ورودی داده 0 هر مولتی پلکسر انتخاب می‌شود. در این وضعیت مسیری از خروجی هر فلیپ فلاپ به ورودی همان فلیپ فلاپ برقرار می‌شود. پالس ساعت بعدی مقدار دودویی قبلی هر فلیپ فلاپ را بداخل آن انتقال می‌دهد. و در نتیجه تغییر حالتی رخ نمی‌دهد. وقتی که $S_1\ S_0 = 01$ باشد، پایانه‌ای در هر مولتی پلکسر که با 1 علامت زده شده است به ورودی D فلیپ فلاپ مربوطه‌اش راه می‌یابد. این خود سبب یک شیفت

1- Unidirectional



شکل ۹-۲ شیفت رجیستر دوطرفه با بار شدن موازی

به راست می‌گردد و ضمن آن داده ورودی سری بداخل فلیپ فلاپ A_0 منتقل می‌شود و بهمین ترتیب محتوای هر فلیپ فلاپ A_{i-1} به فلیپ فلاپ A_i انتقال می‌یابد که در آن $i= 1,2,3$ می‌باشد. اگر $S_1S_0=10$ باشد، اطلاعات دودویی از هر ورودی I_0 تا I_3 وارد فلیپ فلاپ مربوط به خود می‌شوند و در نتیجه یک

جدول ۲-۴ جدول تابع برای ثبات شکل ۲-۹

کنترل شیوه		
S_1	S_0	عملکرد ثبات
0	0	بلاتغییر
0	1	شیفت به راست (پایین)
1	0	شیفت به چپ (بالا)
1	1	بار شدن موازی

بار شدن موازی صورت گرفته است. دقت کنید که براساس شکل ترسیم شده، شیفت به راست محتوای ثبات‌ها را روبه پائین و شیفت به چپ جابجایی را بسمت بالا باعث می‌گردد.

شیفت رجیسترها اغلب برای ارتباط سیستم‌های دیجیتال، با فواصل دور از یکدیگر، بکار می‌روند. مثلاً فرض کنید که بخواهیم کمیتی n بیتی را بین دو نقطه انتقال دهیم. اگر فاصله بین مبداء و مقصد خیلی زیاد باشد، استفاده از n خط برای انتقال موازی آنها پر هزینه خواهد بود. ممکن است استفاده از یک خط واحد مقرون به صرفه‌تر باشد. فرستنده، داده n بیتی موازی رادر شیفت رجیستر بار کرده و سپس داده را از خروجی سریال انتقال می‌دهد. گیرنده، داده سری را از طریق خط ورودی سری بداخل یک شیفت رجیستر منتقل می‌سازد. وقتی که محل n بیت در شیفت رجیسترها جای گرفت می‌توان آن را از طریق خروجی‌های موازی در ثبات برداشت کرد. بنابراین فرستنده یک عمل تبدیل موازی به سری داده‌ها را انجام می‌دهد و بالعکس گیرنده داده‌های سری را به موازی باز می‌گرداند.

۲-۶ شمارنده‌های دودویی

ثباتی که با اعمال پالس‌های ورودی رشته‌ای از حالت‌های از پیش تعیین شده را طی می‌کند، شمارنده خوانده می‌شود. پالس‌های ورودی ممکن است پالس‌های ساعت و یا از یک منبع خارجی حاصل شده باشند. همچنین ممکن است فواصل زمانی پالس یکنواخت و یا تصادفی (غیریکنواخت) باشند. شمارنده‌ها را تقریباً در تمام تجهیزاتی که در آن‌ها مدارهای منطقی دیجیتال وجود دارد می‌توان یافت. از شمارنده‌ها می‌توان برای شمارش تعداد دفعات وقوع یک واقعه استفاده کرد و نیز برای تولید سیگنال‌های زمان بندی در کنترل رشته‌ای از اعمال در کامپیوترها بکار می‌برد.

از میان رشته‌های مختلفی که یک شمارنده می‌تواند طی کند، رشته دودویی ساده‌ترین و سرراست‌ترین است. شمارنده‌ای که رشته اعداد دودویی را دنبال می‌کند شمارنده دودویی نام دارد. یک شمارنده دودویی n بیتی، ثباتی است که از n فلیپ فلاپ و گیت‌های مربوطه‌اش تشکیل شده و رشته از حالات را براساس شمارش دودویی n بیت، از 0 تا 2^n-1 دنبال می‌کند. طراحی شمارنده‌های دودویی با

توجه به روش ارائه شده برای مدارهای ترتیبی که در بخش ۷-۱ کلیات آن بیان شد انجام می‌گیرد. روش ساده‌تری در طراحی، مستقیماً از بررسی حالاتی که ثبات باید اختیار نماید تا شمارش انجام شود امکان‌پذیر است.

با مرور رشته‌ای از اعداد دودویی مانند 0000، 0001، 0010، 0011 و غیره، در می‌یابیم که بیت پائین رتبه‌تر باهر شمارش متمم می‌شود و هر یک از بیت‌های دیگر هنگام افزایش اگروفقط اگر همه بیت‌های کم ارزش‌تر از آن 1 باشند متمم می‌شود. مثلاً شمارش دودویی از 0111 (7) به 1000 (8) با ؛ الف) متمم کردن بیت کم ارزشتر، ب) متمم کردن بیت دوم، چون اولین بیت در 0111 برابر 1 است، ج) متمم کردن بیت مرتبه سوم، چون دو بیت اول در 0111 برابر 1 می‌باشد، د) متمم کردن بیت چهارم، چون سه بیت اول در 0111 همگی برابر 1 می‌باشند، بدست می‌آید.

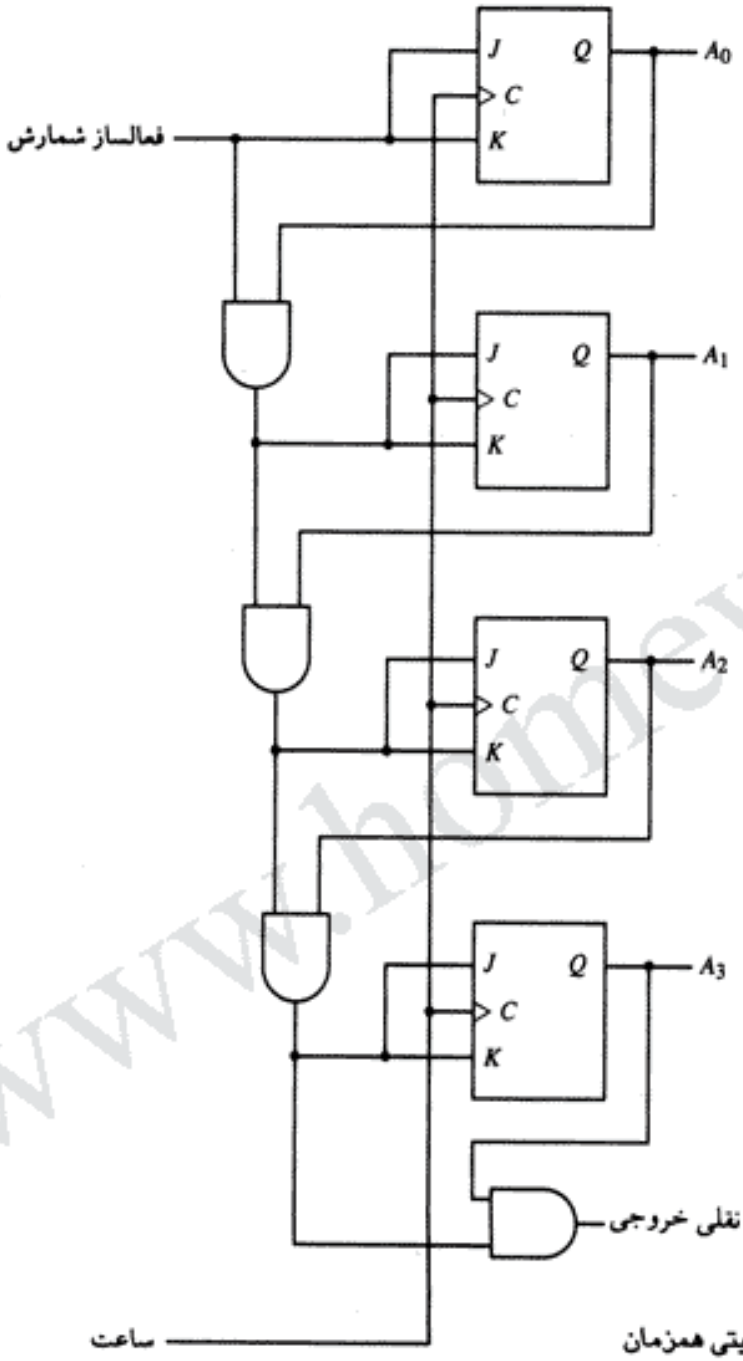
مدار شمارنده معمولاً از فلیپ فلاپهایی که قابلیت متمم شدن دارند استفاده می‌نماید. فلیپ فلاپ‌های T و JK این خاصیت را دارا هستند. بخاطر بیاورید که فلیپ فلاپ JK در صورتی که هر دو ورودی J و K برابر 1 باشند و ساعت گذار مثبت انجام دهد متمم می‌شد. در J=K=0 خروجی فلیپ‌فلاپ تغییر نمی‌کرد. بعلاوه، شمارنده را می‌توان توسط یک ورودی تواناساز کنترل کرد تا بدون جداکردن سیگنال ساعت از فلیپ فلاپها شمارنده را روشن و یا خاموش کند.

همانطور که در شمارنده دودویی 4 بیتی شکل ۱۰-۲ دیده می‌شود اینگونه شمارنده‌های دودویی همزمان (سنکرون) الگوی منظمی دارند. ورودی‌های C در تمام فلیپ فلاپ‌ها ساعت مشترکی را دریافت می‌کنند. اگر فعال ساز شمارش[1] 0 باشد تمام ورودی‌های J و K در 0 باقی می‌مانند و خروجی‌های شمارنده تغییری نمی‌کند. طبقه اول یعنی A_0 وقتی که شمارنده فعال شود و ساعت گذار مثبت انجام دهد متمم می‌گردد. هر یک از سه فلیپ فلاپ دیگر هنگامی متمم می‌شوند که تمام فلیپ فلاپ‌های پائین رتبه‌شان 1 و شمارش نیز فعال شود. زنجیره گیت‌های AND منطق لازم را برای ورودی‌های J و K تهیه می‌کنند. با استفاده از رقم نقلی خروجی می‌توان شمارنده رابه طبقات بیشتری گسترش داد که در هر طبقه یک فلیپ فلاپ و یک گیت AND مورد نیاز است.

شمارنده دودویی با بارشدن موازی

شمارنده‌هایی که در سیستم‌های دیجیتال مورد استفاده قرار می‌گیرند اغلب نیاز به امکان بارشدن موازی برای دریافت یک مقدار اولیه قبل از شمارش دارند. شکل ۱۱-۲ دیاگرام منطقی یک شمارنده دودویی که دارای امکان بارشدن موازی و نیز پاک شدن همزمان به 0 می‌باشد را نشان می‌دهد. اگر ورودی پاک کننده 1 باشد، همه ورودی‌های K را 1 می‌کند و بنابراین با گذار بعدی ساعت همه فلیپ‌فلاپ‌ها را 0 می‌نماید. اگر ورودی کنترل بارکردن برابر 1 باشد، عمل شمارش را غیرفعال نموده و موجب می‌شود تا چهار ورودی داده بداخل چهار فلیپ فلاپ انتقال یابند (بشرطی که ورودی پاک کردن

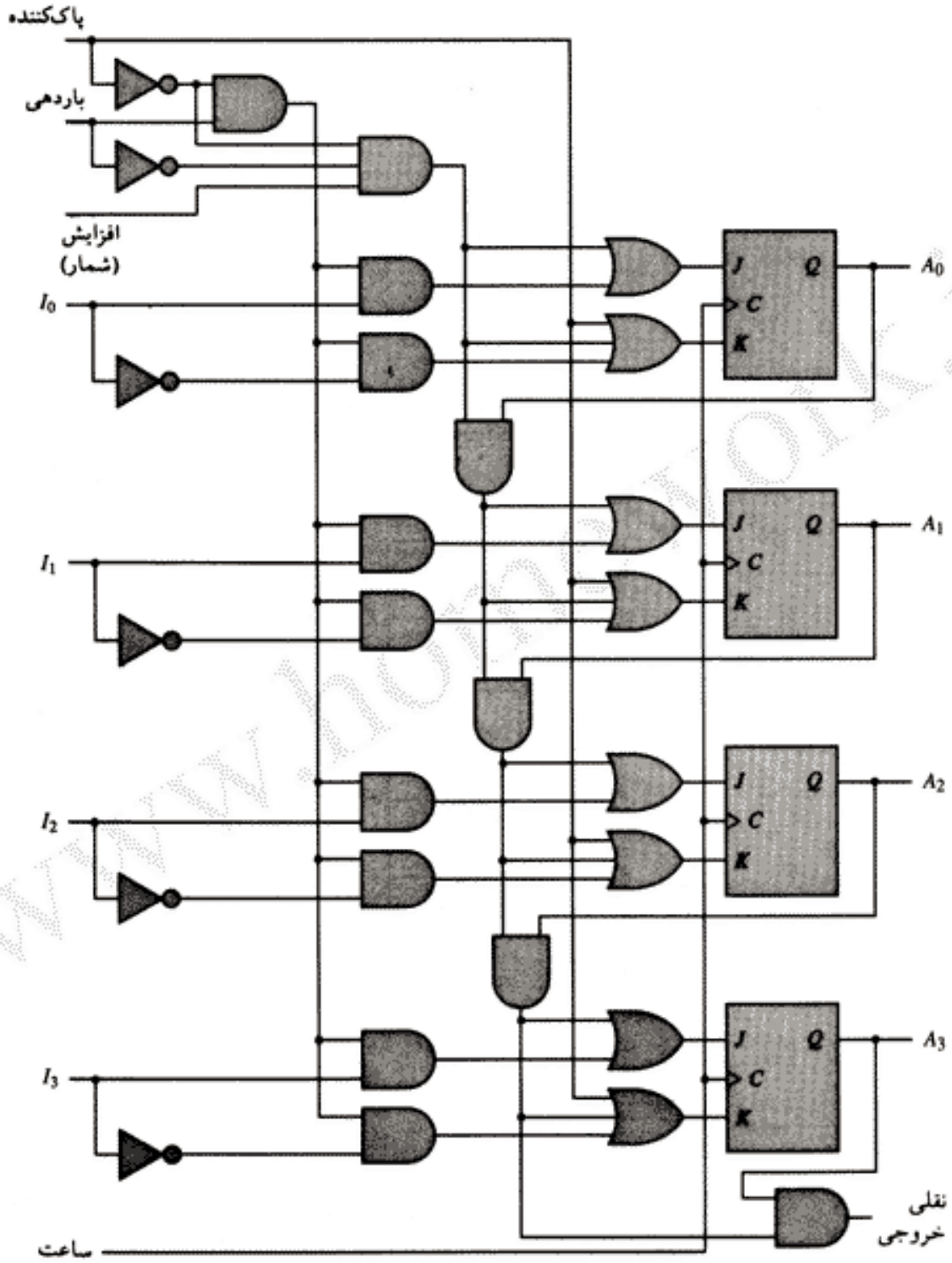
1- Count Enable



شکل ۱۰-۲ شمارنده همزمان 4 بیتی همزمان

0باشد). اگر هر دو ورودی پاک کردن و بارکردن 0و ورودی افزایش [1] 1 باشد مدار بصورت یک شمارنده دودویی عمل خواهد کرد.

1- Increment



شکل ۱۱-۲ شمارنده دودویی 4 بیتی با بارشدن موازی و پاک شدن همزمان

جدول ۵-۲ جدول تابع برای ثبات شکل ۱۱-۲

عمل	افزایش	بارشدن	پاک شدن	ساعت
بلاتغییر	0	0	0	↑
1 واحد افزایش شمارش	1	0	0	↑
بارکردن ورودی‌های I_0 تا I_3	×	1	0	↑
پاک کردن خروجی‌ها به 0	×	×	1	↑

عملکرد مدار در جدول ۵-۲ خلاصه شده است. اگر همه ورودی‌های پاک کردن، بارکردن و افزایش 0 باشند هیچ تغییر در خروجی نخواهد بود حتی اگر پالس‌ها به ورودی C اعمال گردند. اگر ورودی‌های پاک وبارکردن در 0 نگهداشته شود، ورودی افزایش عملکرد راکنترل نموده و خروجی‌ها بازاء هر گذار پالس ساعت به عدد دودویی بعدی در شمارش تغییر خواهند کرد. هر وقت ورودی بارکردن 1 باشد، بشرطی که ورودی پاک کردن ناتوان (غیرفعال) ولی ورودی افزایش 0 یا 1 باشد، داده‌های ورودی بداخل فلیپ فلاپ‌ها بار می‌شوند. با اعمال ورودی پاک کردن، ثبات بدون توجه به مقادیر ورودی‌های بارکردن و افزایش ، 0 خواهد شد.

شمارنده‌هایی که دارای قابلیت بارکردن موازی هستند در طراحی کامپیوترهای دیجیتال خیلی مفیدند. در فصل‌های بعدی، ما از آنها بعنوان ثبات‌هایی که امکان افزایش و بار شدن دارند استفاده خواهیم کرد. عمل افزایش محتوای یک ثبات را یک واحد افزایش می‌دهد. با تواناکردن ورودی شمارش در طول یک پریود ساعت، محتوای ثبات می‌تواند یک واحد افزایش یابد.

۲-۷ واحد حافظه

واحد حافظه مجموعه‌ای از سلول‌های ذخیره‌سازی به همراه مدارات لازم برای انتقال اطلاعات به داخل و خارج آنهاست. حافظه اطلاعات دودویی را بصورت دسته‌هایی از بیت بنام کلمه[1] ذخیره می‌کند. یک کلمه در حافظه مجموعه‌ای از بیت‌هاست که بصورت یک واحد، به داخل و خارج حافظه انتقال می‌یابد. کلمه حافظه[2] گروهی از 1 ها و 0 هاست و ممکن است بعنوان یک عدد، یک کد دستورالعمل، یک یا چند کاراکتر الفبا عددی یا هر اطلاعات دودویی کد شده دیگر باشد. یک گروه هشت بیتی، بایت[3] خوانده می‌شود. در اغلب حافظه‌های کامپیوتر، تعداد بیت‌های هر کلمه مضربی از 8 است. بنابراین یک کلمه 16 بیتی حاوی دو بایت است، و یک کلمه 32 بیتی نیز از چهار بایت ساخته می‌شود. ظرفیت حافظه کامپیوترهای تجاری معمولاً بصورت تعداد بایت‌هایی که می‌توان در آن ذخیره کرد، بیان می‌شود.

ساختمان درونی یک واحد حافظه بوسیله تعداد کلمات آن و تعداد بیت‌های هر کلمه مشخص

1- word 2- Memory word 3- Byte

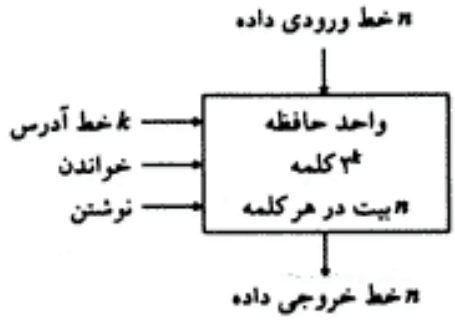
می‌گردد. خطوط ورودی خاصی که کلمه بخصوصی را انتخاب می‌کنند خطوط آدرس[1] نام دارند. به هر کلمه در حافظه یک شماره شناسایی تعلق می‌گیرد که به آن آدرس اتلاق می‌شود، و از 0 آغاز و با 1، 2، 3 تا 2^k-1 ادامه می‌یابد که در آن k تعداد خطوط آدرس است. انتخاب یک کلمه خاص در داخل حافظه با اعمال آدرس دودویی K بیتی به خطوط آدرس تحقق می‌یابد. دیکدری در داخل حافظه این آدرس را دریافت کرده و مسیرهای لازم را برای انتخاب بیت‌های کلمه خاص باز می‌کند. حافظه‌های کامپیوتر ممکن است از 1024 کلمه، که 10 خط آدرس نیاز دارند، تا 2^{32} کلمه، که 32 خط آدرس لازم دارند، متغیر باشد. رسم براین است که تعداد کلمات (یا بایت‌ها) را در یک حافظه با یکی از حروف K (کیلو)[2]، M (مگا)[3] یا G (گیگا)[4] بیان کنند. K برابر با 2^{10}، M برابر با 2^{20} و G برابر با 2^{30} می‌باشد. بنابراین $64K=2^{16}$، $2M=2^{21}$ و $4G=2^{32}$.

دو نوع عمده حافظه که در سیستم‌های کامپیوتری بکار می‌روند عبارتند از: حافظه با دستیابی تصادفی[5] (RAM) و حافظه فقط خواندنی[6] (ROM).

حافظه با دستیابی تصادفی

در حافظه با دستیابی تصادفی (RAM)، انتقال اطلاعات از هر مکان دلخواه بصورت تصادفی و دلخواه امکان‌پذیر است. یعنی عمل نشاندن یک کلمه در هر حافظه یکسان بوده و زمان لازم برای آن مستقل از مکان فیزیکی سلول در حافظه است. مفهوم "دستیابی تصادفی" نیز از همین مطلب ناشی می‌شود.

ارتباط بین حافظه و محیط اطراف آن از طریق خطوط داده ورودی و خروجی، انتخاب آدرس، و خطوط کنترل که جهت انتقال را مشخص می‌کنند صورت می‌گیرد. بلاک دیاگرام یک واحد RAM در شکل ۱۲-۲ نشان داده شده است. n خط ورودی داده اطلاعات راکه باید در حافظه ذخیره شوند فراهم می‌آورند، و n خط خروجی داده نیز اطلاعاتی راکه از حافظه بیرون می‌آیند در اختیار می‌گذارند. k خط آدرس یک عدد دودویی k بیتی را برای انتخاب کلمه موردنظر از میان 2^k کلمه، در داخل حافظه فراهم



شکل ۱۲-۲ بلاک دیاگرام یک حافظه با دستیابی تصادفی (RAM)

1- Address
2- kilo
3- Mega
4- Giga
5- Random Access Memory, RAM
6- Read Only Memory, ROM

می‌آورند. دو خط کنترل ورودی جهت انتقال موردنظر را مشخص می‌کنند.

در یک حافظه RAM می‌توان دو عمل نوشتن[1] و خواندن[2] را انجام داد. سیگنال نوشتن، انتقال به داخل و خواندن، انتقال به خارج را میسر می‌سازند. در پذیرش یکی از این دو سیگنال کنترل، مدارات داخلی حافظه اعمال لازم را انجام می‌دهند. مراحلی که برای انتقال یک کلمه جدید و ذخیره آن در حافظه باید طی شود بقرار زیر است.

۱- اعمال آدرس کلمه مورد نظر به خطوط آدرس

۲- اعمال بیت‌های داده‌ای که قرار است در حافظه ذخیره شوند، بر روی خطوط ورودی داده.

۳- فعال کردن ورودی نوشتن.

سپس واحد حافظه بیت‌های موجود بر روی خطوط ورودی داده را دریافت کرده و آنها را در کلمه تعیین شده بوسیله خطوط آدرس جای می‌دهد.

مراحلی که برای انتقال یک کلمه به خارج از حافظه باید طی شود بقرار زیر است.

۱- اعمال آدرس دودویی کلمه موردنظر بر روی خطوط آدرس

۲- فعال کردن ورودی خواندن

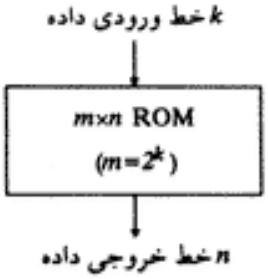
سپس واحد حافظه، بیت‌های کلمه انتخاب شده بوسیله آدرس را برداشته و بر روی خطوط خروجی داده قرار می‌دهد. محتوای کلمه انتخابی پس از خواندن تغییری نمی‌کند.

حافظه فقط خواندنی

حافظه فقط خواندنی (ROM) همانطور که از نامش مشخص است، حافظه‌ای است که فقط عمل خواندن را انجام می‌دهد؛ یعنی دارای امکان نوشتن نیست. این بدان معنی است که اطلاعات دودویی ذخیره شده در ROM حین تولید سخت افزار در آن بصورت دائمی در آمده است و نمی‌توان با نوشتن کلمات مختلف در آن، محتوای آن را عوض کرد. در حالیکه RAM یک وسیله همه منظوره است و محتوای آن در طول پردازش قابل تغییر می‌باشد قطعه ROM محدود به خواندن کلماتی است که در داخل آن بطور دائم ذخیره شده است. اطلاعات دودویی که باید در ROM ذخیره شود بوسیله طراح مشخص شده و سپس هنگام ساخت قطعه با شکل خاصی از اتصالات داخلی در آن پیاده می‌شود. نوعی از ROM ها دارای فیوزهای الکترونیکی خاص هستند که می‌توان آنها را برای آرایش خاصی "برنامه دهی" کرد. به محض ایجاد الگوی موردنظر، حتی با قطع برق نیز اطلاعات در آن باقی خواهد ماند.

یک ROM، با اندازه n×m آرایه‌ای از سلول‌های دودویی است که بصورت m کلمه n بیتی سازمان‌دهی شده است. همانطور که در شکل ۱۳-۲ دیده می‌شود، ROM دارای k خط ورودی آدرس

1- Write 2- Read



شکل ۱۳-۲ بلاک دیاگرام حافظه فقط خواندنی (ROM)

برای انتخاب $m=2^k$ کلمه از حافظه، و n خط خروجی است که هر یک از این خطوط به یک بیت از هر کلمه مرتبط است. یک مدار مجتمع ROM ممکن است دارای یک یا چند ورودی فعال ساز (تواناساز) بوده و برای تشکیل یک ROM با ظرفیت بیشتر بهم متصل شوند.

ROM به خط کنترل خواندن نیازی ندارد زیرا در هز زمان معین، خطوط خروجی خودبخود n بیت کلمه انتخاب شده توسط مقدار آدرس را در اختیار می‌گذارند. چون خروجی‌ها فقط تابعی از ورودی‌های همان لحظه (خطوط آدرس) هستند، ROM را بعنوان مدارهای ترکیبی طبقه‌بندی می‌نمایند. در واقع ساختار داخلی یک ROM از یک دیکدر و مجموعه‌ای از گیت‌های OR تشکیل شده است. در این وسیله نیازی به وجود عنصر ذخیره کننده، همچون RAM، نیست زیرا مقادیر بیت‌ها در ROM بصورت دائمی تثبیت شده‌اند.

ROM ها کاربرد گسترده‌ای در طراحی سیستم‌های دیجیتال دارند. یک ROM اساساً یک رابطه ورودی - خروجی مشخص شده بوسیله یک جدول درستی را تولید می‌کند. بنابراین، هر مدار ترکیبی را با k ورودی و n خروجی پیاده‌سازی می‌کند. وقتی که ROM در یک سیستم کامپیوتری بعنوان واحد حافظه بکار گرفته شد به منظور ذخیره کردن برنامه‌های ثابتی که تغییری نمی‌کنند و یا جداول حاوی ثابت‌ها مورد استفاده قرارگرفته است. ROM در ساخت واحد کنترل کامپیوتر دیجیتال نیز بکار گرفته می‌شود. در این حال، این واحد برای ذخیره اطلاعات کد شده‌ای که نماینده رشته متغیرهای کنترل داخلی لازم برای فعال کردن اعمال مختلف در کامپیوتر است استفاده می‌شود. یک واحد کنترل که از ROM برای ذخیره سازی اطلاعات دودویی کنترل استفاده می‌کند یک واحد کنترل ریزبرنامه نویسی[1] نام دارد. این موضوع در فصل ۷ مفصل‌تر بحث شده است.

انواع ROM

مسیرهای لازم در یک ROM به سه طریق مختلف قابل برنامه ریزی است. اولین روش، یعنی

1- Microprogrammed Control Unit

برنامه ریزی با ماسک[1]، در حین آخرین فرآیند ساخت توسط کمپانی سازنده نیمه رسانا، انجام می‌شود. در روش ساخت ROM، مشتری جدول درستی آنچه راکه ROM باید برآن اساس کار کند پر می‌نماید. جدول درستی ممکن است بروش‌های مختلفی تهیه و یا با قالب خاصی در خروجی کامپیوتر ارائه شود. سازنده، ماسک مناسبی را برای مسیرها تهیه می‌کند تا 1ها و 0ها بر طبق جدول درستی مشتری تهیه شوند. این روش پرهزینه است زیرا فروشنده، مبلغ خاصی را بخاطر ساخت سفارشی ROM تقاضا می‌کند. به این دلیل، برنامه ریزی با ماسک فقط هنگامی اقتصادی است که تعداد زیادی ROM با آرایش یکسان سفارش گردد.

برای تعداد کم، روش دوم که حافظه قابل برنامه ریزی فقط خواندنی،[2] PROM نام دارد اقتصادی‌تر است. بهنگام سفارش تمام فیوزهای PROM دست نخورده هستند و لذا تمام بیت‌های ذخیره شده در کلمه 1 خواهند بود. فیوزهای PROM را می‌توان با اعمال پالس‌های جریان از طریق پایانه‌های خروجی برای هریک از آدرس‌ها سوزاند. یک فیوز سوخته حالت دودویی 0 را بوجود می‌آورد و فیوز دست نخورده حالت دودویی 1 را می‌دهد. این مطلب به کاربر این امکان را می‌دهد تا PROM مورد نظر خود را در آزمایشگاه خود برای بدست آوردن رابطه بین آدرس ورودی و کلمه ذخیره شده برنامه‌نویسی کند. دستگاه‌های خاصی بنام برنامه‌نویس PROM برای انجام این عمل بصورت تجاری موجود است. در هر صورت، روش‌های موجود در برنامه نویسی ROMها سخت افزاری می‌باشند هر چند که از کلمه "برنامه نویسی" هم در بیان آن استفاده شده باشد.

اعمال برنامه‌نویسی سخت افزاری ROMها و PROM ها غیرقابل برگشتند، و به محض اینکه برنامه نویسی شدند، الگوی ثابتی در آنها ایجاد شده و دیگر تغییر نخواهند کرد. چنانچه قرار باشد پس از ایجاد الگوی بیت‌ها، آنها را تغییر دهیم، قطعه باید کلاً دور انداخته شود. نوع سومی از ROMها نیز وجود دارد که PROM پاک شونده یا EPROM خوانده می‌شود. EPROM را، حتی اگر فیوزهایش قبلاً سوزانده شده باشد، می‌توان بصورت اولیه بازسازی کرد. وقتی EPROM در زیر اشعه ماوراء بنفش برای مدت معینی قرار گیرد، تابش طول موج کوتاه گیت‌های داخلی راکه نقش فیوز را بعهده دارند تخلیه می‌نماید. پس از پاک کردن، EPROM به حالت اولیه‌اش باز می‌گردد و می‌تواند با مجموعه جدیدی از کلمات برنامه نویسی گردد. انواع بخصوصی از PROM را می‌توان با سیگنالهای الکتریکی، بجای نور ماوراء بنفش، پاک کرد. این PROMها را PROM پاک شدنی الکتریکی یا EEPROM می‌نامند.

مسائل

۲-۱ قطعات SSI TTL اغلب بصورت مدارات مجتمع 14 پایه عرضه می‌شوند. دو پایه برای منبع تغذیه در نظر گرفته شده و بقیه برای پایانه‌های ورودی و خروجی بکار می‌روند. چند مدار از انواع زیر را در یک چنین بسته‌ای می‌توان گنجانید؟ الف) معکوس کننده، ب) گیت OR انحصاری

1- Mask Programming 2- Programmable Read Only Memory

دو ورودی، ج) گیت OR سه ورودی، د) گیت AND چهار ورودی، ه) گیت NOR پنج ورودی، و) گیت NAND هشت ورودی، ز) فلیپ فلاپ JK ساعت دار با پاک کننده غیرهمزمان.

۲-۲ تراشه‌های MSI وجود توابع دیجیتال ساده‌ای همچون دیکدر، مولتی پلکسر، ثبات‌ها و شمارنده‌ها را امکان‌پذیر می‌سازند. تراشه‌های زیر مدارهای مجتمع از نوع TTL هستند که چنین توابعی را تولید می‌نمایند. مشخصات آنها را در کتاب راهنما یافته و آنها را با قطعات متناظر ارائه شده در این فصل مقایسه کنید.

الف) IC نوع 74155، دیکدر دوتایی 2 به 4

ب) IC نوع 74157، مولتی پلکسر چهار تایی 2 به 1 خطی

ج) IC نوع 74194، شیفت رجیستر چهار بیت دوطرفه با بارشدن موازی

د) IC نوع 74163، شمارنده دودویی چهاربیت بابارشدن موازی و پاک شدن همزمان

۲-۳ یک دیکدر 5 به 32 را با چهار دیکدر 3 به 8 که دارای ورودی تواناساز است و یک دیکدر 2 به 4 بسازید . از بلاک دیاگرام مشابه شکل ۳-۲ استفاده کنید.

۲-۴ دیاگرام منطقی یک دیکدر 2 به 4 را فقط با گیت NOR ترسیم کنید. ورودی فعال ساز نیز در نظر گرفته شود.

۲-۵ دیکدر شکل 2-2 را طوری تصحیح کنید که مدار اگر E=1 باشد فعال و اگر E=0 باشد غیرفعال باشد. جدول درستی را برای نوع تصحیح شده رسم کنید.

۲-۶ دیاگرام منطقی یک انکدر با هشت ورودی و سه خروجی ، که جدول درستی آن در جدول ۲-۲ آورده شد را ترسیم کنید. وقتی تمام ورودی‌ها 0 باشند خروجی چیست؟ اگر فقط ورودی D برابر 0 باشد خروجی چیست. روشی را پیشنهاد کنید که این دو حالت را از هم تمیز دهد.

۲-۷ یک مولتی پلکسر 16 به 1 را با دو مولتی پلکسر 8 به 1 و یک مولتی پلکسر 2 به 1 بسازید. برای هر سه مولتی پلکسر از بلاک دیاگرام استفاده کنید.

۲-۸ بلاک دیاگرام یک مولتی پلکسر را ترسیم کرده و عملکرد آن را بوسیله جدول تابع توضیح دهید.

۲-۹ یک گیت AND دو ورودی رادر ثبات شکل ۶-۲ بگنجانید و خروجی این گیت را به ورودی‌های ساعت همه فلیپ فلاپ‌ها وصل کنید. یکی از ورودی‌های گیت AND پالس‌های ساعت را از مولد پالس ساعت دریافت می‌کند. ورودی دیگر گیت AND کنترل بارشدن موازی را فراهم می‌آورد. عملکرد ثبات جدید را تشریح کنید.

۲-۱۰ هدف از گیت بافر در ورودی ساعت ثبات شکل ۷-۲ چیست؟

۲-۱۱ به ثباتی که دارای امکان بارشدن موازی در شکل ۷-۲ است،امکان ورودی پاک کننده را اضافه کنید.

۲-۱۲ مقدار اولیه محتوای یک ثبات 1101 است. ثبات شش بار با استفاده از ورودی سری 101101 به راست شیفت داده می‌شود. محتوای ثبات پس از شیفت چیست.

۲-۱۳ فرق بین انتقال سری و موازی چیست؟ با استفاده از یک شیفت رجیستر که قابلیت بارشدن موازی دارد توضیح دهید چگونه می‌توان ورودی سری را به خروجی موازی و بالعکس تبدیل کرد.

۲-۱۴ شمارنده حلقوی یک شیفت رجیستر مطابق شکل ۸-۲ می‌باشد که خروجی سری‌اش به ورودی سری آن وصل شده است. با شروع از حالت اولیه 1000، رشته حالات چهار فلیپ فلاپ را پس از هر شیفت لیست کنید.

۲-۱۵ یک شیفت رجیستر چهاربیتی دوطرفه با قابلیت بار شدن موازی طبق شکل ۹-۲ در یک بسته مدار مجتمع بسته‌بندی شده است.

الف) بلاک دیاگرام مدارمجتمع را با تمام ورودی‌ها و خروجی‌هایش ترسیم کنید.

ب) با استفاده از دو مدار مجتمع بلاک دیاگرام یک شیفت رجیستر 8 بیت با قابلیت بار شدن موازی را رسم کنید.

۲-۱۶ در یک شمارنده دودویی ده بیت چند فلیپ فلاپ در شمارش بعدی متمم می‌شوند.

الف) 1001100111 ب) 0011111111؟

۲-۱۷ اتصالات لازم بین چهار شمارنده دودویی چهاربیتی با بارشدن موازی (شکل ۱۱-۲) برای ایجاد یک شمارنده دودویی 16 بیت با بارشدن موازی را نشان دهید. برای هر شمارنده چهاربیتی از یک بلاک دیاگرام استفاده کنید.

۲-۱۸ نشان دهید که چگونه یک شمارنده دودویی با بارشدن موازی شکل ۱۱-۲ را می‌توان به یک شمارنده تقسیم بر N تبدیل کرد (یعنی شمارنده‌ای که از 0000 تا N شمرده و به 0000 باز گردد). بعنوان یک حالت خاص مدار یک تقسیم بر 10 را با استفاده از شکل ۱۱-۲ و یک گیت AND خارجی رسم کنید.

۲-۱۹ واحدهای حافظه زیر با تعداد کلمات، ضرب در تعداد بیت‌ها مشخص شده‌اند. در هر مورد چند خط آدرس و چند خط داده ورودی - خروجی لازم است؟

الف) 2K×16 ب) 64K×8

ج) 16M×32 د) 4G×64

۲-۲۰ تعداد بایت‌هایی را که می‌توان در حافظه‌های مسئله ۱۹-۲ ذخیره کرد مشخص کنید.

۲-۲۱ چند تراشه حافظه 128×8 برای ایجاد یک حافظه 4096×16 لازم است؟

۲-۲۲ با فرض داشتن یک ROM به ظرفیت 32×8 بیت و یک ورودی تواناساز، اتصالات خارجی لازم برای ساخت یک تراشه ROM با ظرفیت 128×8 و یک دیکدر را نشان دهید.

۲-۲۳ یک تراشه ROM با ظرفیت 4096×8 دارای دو ورودی تواناساز بوده و با منبع تغذیه 5 ولت کارمی‌کند. چند پایه برای بسته مدار مجتمع لازم است؟ بلاک دیاگرام آن را رسم و پایانه‌های ورودی و خروجی را نام گذاری کنید.

۳ نمایش داده‌ها

۳-۱ انواع داده‌ها
۳-۲ متمم‌ها
۳-۳ نمایش ممیز - ثابت
۳-۴ نمایش ممیز شناور
۳-۵ انواع دیگر کدهای دودویی
۳-۶ کدهای آشکارسازی خطا

۳-۱ انواع داده‌ها

اطلاعات دودویی در کامپیوترهای دیجیتال در حافظه‌ها یا ثبات‌های پردازنده ذخیره می‌شوند. ثبات‌ها حاوی داده‌ها یا اطلاعات کنترل هستند. اطلاعات کنترل متشکل از یک بیت یا گروهی از بیت‌هاست که برای مشخص کردن سیگنال‌های فرمان لازم در دستکاری داده‌های سایر ثبات‌ها استفاده می‌شود. داده‌ها، اعداد و اطلاعات دودویی کد شده دیگری می‌باشند که برای حصول به نتایج محاسباتی مورد نظر روی آنها کار می‌شود.در این فصل متداول‌تر نوع داده‌ها را که در کامپیوترهای دیجیتال یافت می‌شوند معرفی می‌کنیم و چگونگی نمایش انواع مختلف داده‌ها، بشکل کد شده دودویی در ثبات‌های کامپیوتر را نشان خواهیم داد.

انواع داده‌های موجود در ثبات‌های کامپیوترهای دیجیتال را می‌توان به ۳ فرم زیر طبقه‌بندی کرد: ۱) اعدادی که در عملیات حسابی بکار می‌روند، ۲) حروف الفبا مورد استفاده در پردازش داده، و ۳) سایر سمبل‌های گسسته که برای اهداف خاصی بکار می‌روند. همه انواع داده‌ها، بجز اعداد دودویی، به فرم دودویی کد شده در ثبات‌های کامپیوتر نمایش داده می‌شوند. این بدان علت است که ثبات‌ها از

فليپ فلاپ ساخته شده‌اند و فليپ فلاپ‌ها وسيله دوحالتي هستند كه قادرند فقط 1ها و 0ها را در خود ذخيره كنند. سيستم اعداد دودويي طبيعي‌ترين سيستم بكار رفته در يک كامپيوتر ديجيتال است. ولي گاهي مناسبتر است سيستم اعداد ديگري را بكار برد. بخصوص سيستم اعداد دهدهي به اين دليل كه بوسيله انسان جهت عمليات حسابي بكار مي‌روند.

سيستم‌هاي عددنويسي

يک سيستم عددنويسي با پايه[1] يا مبناي[2] r سيستمي است كه سمبل‌هاي متمايز از همي را براي نمايش r رقم بكار مي‌برد. اعداد بارشته‌اي از سمبل‌هاي ارقام نمايش داده مي‌شوند. براي تعيين كميتي كه يک عدد نمايشگر آنست، لازم است تا هر رقم را در توان صحيحي از r ضرب كرده و سپس مجموع همه اين حاصلضرب‌ها محاسبه شود. مثلاً، سيستم اعداد دهدهي كه همه روزه مورد استفاده است سيستم مبناي 10 را بكار مي‌برد. 10 سمبل عبارتند 0، 1، 2، 3، 4، 5، 6، 7، 8، 9. رشته رقم 724.5 بيانگر كميت زير تلقي مي‌شود.

$$7 \times 10^2 + 2 \times 10^1 + 4 \times 10^0 + 5 \times 10^{-1}$$

يعني 7 صدتايي بعلاوه 2 ده تايي بعلاوه 4 يكي بعلاوه 5 دهم. بطور مشابه هر عدد دهدهي را مي‌توان بهمين ترتيب براي يافتن كميت آن تفسير كرد

سيستم عددنويسي دودويي مبناي 2 را بكار مي‌برد. دو سمبل ارقام مورد استفاده، 0 و 1 هستند. رشته ارقام 101101 بيانگر كميت زير است.

$$1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 45$$

براي تفكيک اعداد با مبناهاي مختلف، اعداد در پرانتزهايي قرار داده شده ومبناي عدد بصورت انديس در زير پرانتز قرار مي‌گيرد. مثلاً براي نشان دادن تساوي بين چهل و پنج دودويي و دهدهي مي‌نويسيم $(101101)_2 = (45)_{10}$.

علاوه برسيستم‌هاي اعداد دودويي و دهدهي ، سيستم‌هاي هشت هشتي (مبناي 8) و شانزده شانزدهي (مبناي 16) نيز در كامپيوترهاي ديجيتال اهميت دارند. هشت سمبل اعداد هشت هشتي عبارتند از: 0، 1، 2، 3، 4، 5، 6، 7. شانزده سمبل اعداد شانزده شانزدهي عبارتند از: 0، 1، 2، 3، 4، 5، 6، 7، 8، 9، A، B، C، D، E، F. شش سمبل آخر، متاسفانه، همان حروف الفبا هستند و گاهي ممكن است موجب اشتباه شوند. با اين وجود، اين قراردادي است كه پذيرفته شده است. وقتي كه حروف A، B، C، D، E، F در مبناي شانزده بكار روند به ترتيب متناظر با اعداد 10، 11، 12، 13، 14، 15 مي‌باشند.

يک عدد در مبناي r را با تشكيل جمع حاصلضرب‌هاي ارقام در وزن‌هايشان مي‌توان به سيستم

1- Base 2- Radix

دهدهی تبدیل کرد. مثلاً عدد هشت هشتی 736.4 به صورت زیر به دهدهی تبدیل می‌شود:

$$(736.4)_8 = 7\times8^2 + 3\times8^1 + 6\times8^0 + 4\times8^{-1}$$
$$= 7\times64 + 3\times8 + 6\times1 + 4:8 = (478.5)_{10}$$

عدد دهدهی معادل عدد F3 در مبنای 16 از محاسبه زیر بدست می‌آید

$$(F3)_{16} = F\times16 + 3 = 15\times16 + 3 = (243)_{10}$$

تبدیل از دهدهی به معادل مبنای r آن، با جداسازی بخش صحیح عدد از کسری آن و تبدیل هر بخش بطور جداگانه صورت می‌گیرد. تبدیل یک عدد صحیح دهدهی به مبنای r بوسیله تقسیمات متوالی بر r و کنار هم گذاشتن باقیمانده‌ها انجام می‌شود. شکل ۱-۳ این عملیات را نشان می‌دهد.

تبدیل عدد دهدهی 41.6875 به دودویی ابتدا با جداکردن عدد به دو قسمت صحیح 41 و کسری 0.6875 آغاز می‌شود. برای تبدیل بخش صحیح، 41 را بر r=2 تقسیم می‌کنیم تا خارج قسمت صحیح 20 و باقیمانده 1 بدست آید. خارج قسمت را دوباره بر 2 تقسیم می‌کنیم تا خارج قسمت و باقیمانده جدیدی بدست آید. این روند تا 0 شدن خارج قسمت ادامه می‌یابد. ضرائب عدد دودویی از باقیمانده‌ها حاصل می‌گردد، بدین ترتیب که اولین باقیمانده بیت مرتبه پائین عدد دودویی تبدیل شده را تشکیل می‌دهد.

بخش کسری عدد با ضرب آن در r=2 حاصل می‌گردد تا یک عدد صحیح همراه با کسری تولید کند. قسمت کسری جدید (بدون بخش صحیح آن) دوباره در 2 ضرب می‌شود تا عدد صحیح همراه با کسر دیگری بدست آید. این روند تا صفر شدن بخش کسری و یا تاجایی که ارقام حاصل پاسخگوی دقت موردنظر بشوند ادامه دارد. ضرائب کسری دودویی از ارقام صحیح و با توجه به قرار گرفتن اولین عدد صحیح بعنوان رقم بعد از ممیز تشکیل می‌گردد . نهایتاً دو قسمت برای تبدیل کل با یکدیگر ترکیب می‌گردند.

بخش صحیح = 41

41	
20	1
10	0
5	0
2	1
1	0
0	1

$(41)_{10} = (101001)_2$

بخش کسری = 0.6875

0.6875
2
1.3750
x 2
0.7500
x 2
1.5000
x 2
1.0000

$(0.6875)_{10} = (0.1011)_2$

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

شکل ۱-۳ تبدیل 41.6875 دهدهی به دودویی

اعداد هشت هشتی و شانزده شانزدهی

تبدیل متقابل دودویی، هشت هشتی و شانزده شانزدهی نقش مهمی در کامپیوترهای دیجیتال دارند. چون $8=2^3$ و $16=2^4$ می‌باشد، هر رقم هشت هشتی متعلق به سه رقم دودویی است و هر رقم مبنای شانزده مربوط به چهار رقم دودویی می‌باشد. تبدیل از دودویی به هشت هشتی بسادگی با تقسیم بندی عدد دودویی به گروه‌های سه بیتی انجام می‌شود. سپس رقم هشت هشتی مربوط به هر گروه از بیت‌ها به آن‌ها اختصاص می‌یابد و رشته ارقام حاصل، معادل هشت هشتی را برای عدد دودویی بدست می‌دهد. مثلاً یک ثبات 16 بیتی را در نظر بگیرید. بطور فیزیکی می‌توان تصور کرد که یک ثبات از 16 سلول ذخیره سازی دودویی تشکیل شده است، که هر سلول قادر است 1 یا 0 را در خود نگهدارد. فرض کنید آرایش بیت‌های ذخیره شده در ثبات طبق شکل ۲-۳ باشد. چون یک عدد دودویی رشته‌ای از ترکیبات 1 و 0 است، ثبات 16 بیتی برای ذخیره کردن هر عدد دودویی از 0 تا 2^{16}-1 مناسب است. برای مثال خاص نشان داده شده، عدد دودویی ذخیره شده در ثبات معادل با 44899 می‌باشد. با شروع از بیت مرتبه پایین‌تر ، ما ثبات را به گروه‌های سه بیتی تقسیم می‌کنیم (بیت شانزدهم به تنهایی باقی می‌ماند). به هر گروه سه بیتی، یک رقم معادل هشت هشتی تخصیص می‌یابد و در بالای آن نوشته می‌شود. رشته ارقام هشت هشتی حاصل نمایش دهنده معادل هشت هشتی عدد دودویی است.

تبدیل دودویی به شانزده شانزدهی مشابه فوق است بجز اینکه بیت‌ها به گروه‌های چهارتایی تقسیم می‌شوند. رقم شانزده شانزدهی مربوطه، برای هر گروه چهاربیتی در زیر ثبات درشکل ۲-۳ نوشته شده‌اند. رشته ارقام شانزده شانزدهی حاصل، معادل عدد دودویی است. رقم هشت هشتی مربوط به هر گروه سه بیتی بسادگی از روی اولین هشت رقم در لیست جدول ۱-۳ بخاطر سپرده می‌شود. کد چهاربیتی معادل هر رقم شانزده شانزدهی را می‌توان در 16 سطر اول جدول ۲-۳ پیدا کرد.

جدول ۱-۳ چند عدد هشت هشتی و نمایش آنها را در ثبات‌ها بشکل کد شده دودویی نشان می‌دهد. کد دودویی از روشی که تشریح شد حاصل می‌شود. به هر رقم هشت هشتی یک کد سه بیتی مطابق آنچه در هشت رقم جدول مشخص شده تخصیص می‌یابد. بطور مشابه، جدول ۲-۳ چند عدد شانزده شانزدهی و نمایش آنها را در ثبات بفرم کد شده دودویی نشان می‌دهد. در اینجا کد دودویی بدین صورت حاصل می‌شود که به هر رقم شانزده شانزدهی یکی از شانزده وارده لیست شده در جدول اختصاص می‌یابد.

با مقایسه اعداد هشت هشتی و شانزده شانزدهی در فرم کد دودویی با معادل عدد دودویی آنها، می‌بینیم که ترکیب بیت‌ها در هر سه فرم یکسان است. مثلاً عدد دهدهی 99 وقتی بصورت دودویی درآید بشکل

هشت‌هشتی	1	2	7	5	4	3
دودویی	1 0 1	0	1 1 1	0 1 1	0 0	1 1
شانزده‌شانزدهی	A		F	6		3

شکل ۲-۳ تبدیل دودویی، هشت هشتی، شانزده شانزدهی

جدول ۱-۳ اعداد هشت هشتی کد شده با دودویی

عدد هشت هشتی	هشت هشتی کد شده با دودویی	معادل دهدهی	
0	000	0	↑
1	001	1	
2	010	2	کد یک رقم
3	011	3	هشت هشتی
4	100	4	
5	101	5	
6	110	6	
7	111	7	↓
10	001 000	8	
11	001 001	9	
12	001 010	10	
24	010 100	20	
62	110 010	50	
143	001 100 011	99	
370	011 111 000	248	

جدول ۲-۳ اعداد شانزده شانزدهی کد شده با دودویی

عدد شانزده شانزدهی	شانزده شانزدهی کد شده با دودویی	معادل دهدهی	
0	0000	0	↑
1	0001	1	
2	0010	2	
3	0011	3	
4	0100	4	
5	0101	5	
6	0110	6	کد یک رقم
7	0111	7	شانزده شانزدهی
8	1000	8	
9	1001	9	
A	1010	10	
B	1011	11	
C	1100	12	
D	1101	13	
E	1110	14	
F	1111	15	↓
14	0001 0100	20	
32	0011 0010	50	
63	0110 0011	99	
F8	1111 1000	248	

1100011 خواهد بود. معادل هشت هشتی عدد دهدهی 99 برابر است با 011 100 001 و معادل شانزده شانزدهی آن 0011 0110 می‌باشد. اگر ما صفرهای سمت چپ را در این سه فرم نادیده بگیریم، می‌بینیم که ترکیب بیت‌های آنها یکی است. در واقع باید هم همینطور باشد زیرا تبدیل مستقیمی بین اعداد دودویی و هشت‌هشتی یا شانزده شانزدهی صورت گرفته است. نکته‌ای که در این بحث وجود دارد این است که یک رشته از 1ها و 0ها که در یک ثبات ذخیره شده‌اند می‌توانند یک عدد دودویی را نمایش دهند، ولی همین رشته می‌تواند بعنوان یک عدد هشت هشتی یا کد دودویی هم تفسیر شود (اگر ما بیت‌ها را به گروه‌های سه بیتی تقسیم کنیم) و یا بعنوان یک عدد شانزده شانزدهی یا کد دودویی تعبیر گردد.

ثبات‌ها در کامپیوترهای دیجیتال حاوی بیت‌های زیادی هستند. مشخص کردن محتوای ثبات‌ها با مقادیر دودویی آنها، رشته‌ای طولانی از ارقام دودویی را نیاز خواهد داشت. بهتر این است که محتوای ثبات‌ها با معادل هشت هشتی یا شانزده شانزدهی مشخص شوند. بدین ترتیب تعداد ارقام به یک سوم در هشت هشتی، و به یک چهارم در شانزده شانزدهی تقلیل می‌یابد. مثلاً عدد دودویی 1111 1111 1111 دارای 12 رقم است. این عدد در نمایش هشت هشتی بصورت 7777 (چهاررقم) و یا در نمایش شانزده شانزدهی بشکل FFF (سه رقم) می‌باشد. در کتب راهنمای کامپیوترها همیشه از نمایش هشت هشتی یا شانزده شانزدهی برای مشخص کردن محتوای ثبات‌ها استفاده می‌شود.

نمایش دهدهی

سیستم اعداد دودویی طبیعی‌ترین سیستم برای یک کامپیوتر است، ولی انسان‌ها به دستگاه دهدهی عادت دارند. یک راه حل برای این مشکل تبدیل تمام اعداد دهدهی ورودی به اعداد دودویی، انجام تمام اعمال حسابی به شکل دودویی و سپس تبدیل مقادیر دودویی به دهدهی برای درک فرد کاربر است. با این وجود، می‌توان اعمال محاسباتی را در کامپیوتر مستقیماً با اعداد دهدهی انجام داد بشرطی که آنها درثبات‌ها بشکل کد شده ذخیره باشند. اعداد دهدهی معمولاً بصورت کاراکترهای الفبا عددی کد شده دودویی وارد کامپیوتر می‌شوند. این کدها، که بعداً معرفی خواهند شد، برای هر رقم دهدهی بین شش تا هشت بیت است. وقتی که اعداد دهدهی برای محاسبات حسابی داخلی بکار روند، تبدیل به یک دودویی چهار بیت برای هر رقم می‌گردند.

یک کد دودویی گروهی از n بیت است که تا 2^n ترکیب متمایز از 0ها و 1ها را بخود می‌گیرد. هر ترکیب نمایشگر یک عنصر از مجموعه‌ای است که قرار است بصورت کد درآید. مثلاً، مجموعه‌ای از چهار عنصر بوسیله یک کد دوبیتی بصورت کد درمی‌آیند که هر عنصر متعلق بیکی از ترکیبهای بیتی؛ 00، 01، 10 و 11 است. مجموعه‌ای از هشت عنصر نیازمند یک کد سه بیتی است و یک مجموعه شانزده عنصری به کد چهاربیتی احتیاج دارد و بهمین ترتیب. اگر تعداد عناصر در مجموعه توانی صحیح از 2 نباشد، برخی از ترکیب بیت‌ها در کد دودویی به هیچ عنصری اختصاص نخواهد یافت. ارقام دهدهی

چنین مجموعه‌ای را تشکیل می‌دهند. یک کد دودویی که بین ده عنصر تمایز بوجود آورد باید حداقل چهاربیت داشته باشد، ولی شش ترکیب بدون استفاده خواهند ماند. کدهای متفاوت متعددی را با چهاربیت بصورت ده ترکیب مختلف و متمایز می‌توان بدست آورد. معمول‌ترین تخصیص بیت‌های بکار رفته برای ارقام دهدهی همان تخصیص دودویی سرراستی است که در ده وارده اول جدول ۳-۳ لیست شده است. این کد خاص، دهدهی کد شده با دودویی [1] نامیده می‌شود و غالباً با BCD نشان داده می‌شود. سایر کدهای دهدهی گاهی اوقات بکار می‌روند و چند تا از آنها در بخش ۵-۳ آورده شده‌اند.

دانستن اختلاف بین تبدیل اعداد دهدهی به دودویی و کد دودویی یک عدد دهدهی بسیار مهم است. مثلاً، وقتی که عدد دهدهی 99 به دودویی تبدیل شود حاصل 1100011 خواهد شد، ولی وقتی بصورت BCD درآید حاصل 1001 1001 خواهد بود. تنها اختلاف بین عدد دهدهی که بصورت سمبل‌های رقمی 0، 1، 2، ....9، نمایش داده شده است و سمبل‌های BCD 0001، 0010، در نمادهای مورد استفاده برای نمایش ارقام است و در واقع خود عدد دقیقاً یکسان است. چند عدد دهدهی و نمایش BCD آنها در جدول ۳-۳ آورده شده است.

نمایش الفا عددی

بسیاری از کاربردهای کامپیوترهای دیجیتال مستلزم کار با داده‌هایی است که علاوه بر اعداد، حروف

جدول ۳-۳ اعداد دهدهی کد شده با دودویی (BCD)

عدد دهدهی	عدد دهدهی کد شده با دودویی (BCD)	
0	0000	↑
1	0001	
2	0010	
3	0011	کد یک
4	0100	
5	0101	رقم دهدهی
6	0110	
7	0111	
8	1000	
9	1001	↓
10	0001 0000	
20	0010 0000	
50	0101 0000	
99	1001 1001	
248	0010 0100 1000	

1- Binary Coded Decimal

الفبا و کاراکترهای خاصی را نیز شامل می‌شود. یک مجموعه از کاراکترهای الفبا عددی متشکل ازمجموعه 10 عنصر ارقام دهدهی، 26 حرف الفبا و تعدادی کاراکترهای خاص مانند $، + و = است. چنین مجموعه‌ای بین 32الی 64 عنصر (اگر فقط حروف بزرگ در نظر گرفته شود) و یا 64 تا 128 عنصر (اگر هر دو حروف بزرگ و کوچک بکار روند) خواهد داشت. در حالت اول،کد دودویی، شش بیت نیاز دارد و در حالت دوم هفت بیت مورد نیاز است. کد استاندارد الفبا عددی دودویی ASCII[1] است که از هفت بیت برای 128 کاراکتر استفاده می‌کند. کدهای دودویی برای حروف بزرگ، ارقام دهدهی، و چند کاراکتر خاص دیگر در جدول ۴-۳ لیست شده‌اند. توجه کنید که ارقام دهدهی در ASCII با حذف سه بیت مرتبه بالاتر 011، قابل تبدیل به BCD هستند. لیست کامل کاراکترهای ASCII در جدول ۱-۱۱ آورده شده است.

جدول ۴-۳ کد استاندارد آمریکایی برای تبادل اطلاعات (ASCII)

کاراکتر	کد دودویی	کاراکتر	کد دودویی
A	100 0001	0	011 0000
B	100 0010	1	011 0001
C	100 0011	2	011 0010
D	100 0100	3	011 0011
E	100 0101	4	011 0100
F	100 0110	5	011 0101
G	100 0111	6	011 0110
H	100 1000	7	011 0111
I	100 1001	8	011 1000
J	100 1010	9	011 1001
K	100 1011		
L	100 1100		
M	100 1101	space	010 0000
N	100 1110		010 1110
O	100 1111	(	010 1000
P	101 0000	+	010 1011
Q	101 0001	$	010 0100
R	101 0010	*	010 1010
S	101 0011	)	010 1001
T	101 0100	-	010 1101
U	101 0101	/	010 1111
V	101 0110	,	010 1100
W	101 0111	=	011 1101
X	101 1000		
Y	101 1001		
Z	101 1010		

1- American Standard Code for Information Interchange

کدهای دودویی نقش مهمی را در عملیات کامپیوترهای دیجیتال ایفا می‌کنند. کدها باید به فرم دودویی باشند زیرا ثبات‌ها قادرند فقط اطلاعات دودویی را نگهداری نمایند. باید توجه داشت که کدهای دودویی صرفاً سمبل‌ها را تغییر می‌دهند و نه مفهوم عناصر گسسته‌ای راکه نمایش می‌دهند. در عملیات کامپیوترهای دیجیتال باید مفهوم بیت‌های ذخیره شده در ثبات‌ها در نظر گرفته شود تا اعمال روی عملوندهای یکسان انجام شوند. در بررسی تصادفی بیت‌های یک ثبات کامپیوتری ملاحظه می‌شود که محتوای آن اغلب اطلاعات کد شده است و نه یک عدد دودویی.

کدهای دودویی برای هر مجموعه از عناصر گسسته مانند نت‌های موسیقی و مهره‌های شطرنج و مکان‌های آنها در روی صفحه شطرنج قابل تعریف‌اند. همچنین این کدها برای معرفی دستوراتی که اطلاعات کنترلی را در کامپیوترها مشخص می‌کنند بکار می‌روند. این فصل درباره نمایش داده‌ها بحث می‌کند. کد دستورات در فصل پنج بحث خواهد شد.

۲-۳ متمم‌ها

در کامپیوترهای دیجیتال از متمم‌ها برای ساده‌کردن عمل تفریق و عملیات منطقی استفاده می‌شود. در هر سیستم مبنای r دو نوع متمم وجود دارد: متمم r و متمم r-1. هرگاه به جای r، مقدار آن در نام متمم قرار گیرد، برای اعداد دودویی متمم 2 و متمم 1 و برای اعداد دهدهی متمم 10 و متمم 9 را خواهیم داشت.

متمم r-1

اگر عدد N در پایه r بصورت n رقم موجود باشد، متمم r-1 عدد N بصورت $(r^n-1)-N$ تعریف می‌شود. برای اعداد دهدهی r=10 و r-1=9 است، بنابراین متمم 9 عدد N برابر است با $(10^n-1)-N$. عدد 10^n از یک 1 و n عدد 0 تشکیل شده است. مقدار 10^n-1 برابر خواهد بود با n عدد 9. مثلاً، با n=4 داریم $10^4=10000$ و $10^4-1=9999$. نتیجه می‌شود که متمم 9 یک عدد دهدهی با تفریق هر یک از ارقام آن از 9 بدست می‌آید. مثلاً متمم 9 عدد 546700 برابرست با 999999−546700=453299 و متمم 9 عدد 12389 برابرست با 99999−12389=87610.

برای اعداد دودویی، r=2 و r−1=1 است، بنابراین متمم 1 عدد N برابر $(2^n-1)-N$ خواهد بود. مجدداً، 2^n بوسیله یک عدد دودویی نمایش داده می‌شود و متشکل است از یک 1 و n عدد 0که بدنبال آن آمده است. 2^n-1 یک عدد دودویی است که بوسیله n عدد 1 نشان داده می‌شود. مثلاً برای n=4 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 تفریق هر رقم از 1 سبب می‌شود تا هر بیت از 0 به 1 و از 1 به 0 تبدیل شود. بنابراین متمم 1 یک عدد دودویی از تبدیل 1ها به 0ها و 0ها به 1ها بدست می‌آید. مثلاً متمم 1 عدد 1011001 برابر است با 0100110، و متمم 1 عدد 0001111 عبارتست از 1110000.

متمم (r−1) اعداد هشت هشتی یا شانزده شانزدهی بترتیب از تفریق هر رقم از 7 یا F (15 دهدهی) بدست می‌آید.

متمم r

متمم r یک عدد n رقمی N در پایه r بازاء N≠0 برابرست با r^n-N و بازاء N=0 برابر 0 است. می‌بینیم که متمم r درمقایسه با متمم r-1، از جمع 1 با متمم r−1 حاصل می‌شود زیرا $r^n-N=[(r^n-1)-N]+1$ می‌باشد. بنابراین متمم 10 عدد دهدهی 2389 برابر است با 7610+1=6711 که از جمع 1 با مقدار متمم 9 بدست می‌آید. متمم 2 عدد دودویی 101100 برابرست با 010011+1=010100که از جمع 1 با مقدار متمم 1 بدست آمده است.

چون 10^n عددی است که با یک 1 و بدنبال آن با n عدد 0 نشان داده می‌شود، لذا 10^n-N که متمم 10 عدد N است، نیز می‌تواند بهمین صورت تعریف شود. بدین ترتیب که تمام 0های کم ارزشتر را دست نخورده باقی گذاشته و اولین رقم غیر 0 را از 10 کم می‌کنیم، و سپس تمام رقم‌های مرتبه بالاتر را از 9 کسر می‌نمائیم. متمم 10 عدد 246700 برابر است با 753300 است و بدین ترتیب حاصل می‌شود که دو رقم 00 را باقی گذاشته، 7 را از 10 کم می‌کنیم، و سه رقم باقیمانده را از 9 کسر می‌کنیم. بطور مشابه، برای بدست آوردن متمم 2 می‌توان همه 0های سمت راست و اولین رقم 1 را دست نخورده باقی گذاشت، و سپس 0 ها بجای 1 و 1 ها بجای 0 در بقیه مکان‌های با ارزشتر گذاشته می‌شوند. متمم 2 عدد 1101100 برابر است با 0010100 و با باقی گذاشتن دو 0 مرتبه پائین‌تر و اولین 1، و سپس با جایگزینی 1ها با 0ها و 0 ها با 1 ها در چهاربیت مرتبه بالاتر بدست می‌آید.

در تعریف فوق فرض شد که اعداد دارای ممیز نیستند. اگر عدد اولیه N دارای نقطه ممیز باشد، باید بطور موقت حذف گردد تا بتوان متمم‌های r و r-1 را تشکیل داد. سپس نقطه ممیز در همان موقعیت نسبی اولیه‌اش وارد می‌شود. ذکر این نکته هم مفید است که متمم یک متمم همان مقدار اولیه را می‌دهد. متمم r عدد N برابر r^n-N است. متمم این متمم $r^n-(r^n-N)=N$ خواهد بود که در واقع همان عدد اولیه می‌باشد.

تفریق اعداد بی علامت

در روش مستقیم تفریق که در دبستان تدریس شده از مفهوم "قرض کردن" استفاده شد. در این روش هرگاه رقمی از مفروق منه کوچکتر از رقم متناظر مفروق باشد یک واحد از عدد مرتبه بالاتر قرض می‌کنیم. بنظر می‌رسد که این ساده‌ترین کار بهنگام انجام تفریق با قلم و کاغذ است. وقتی که تفریق را با سخت افزار دیجیتال انجام دهیم، این روش در مقایسه با روش بکارگیری متمم از کارآیی کمتری برخوردار است.

تفریق دو عدد n رقمی بی علامت $M-N\ (N\neq 0)$ در مبنای r بطریق زیر انجام می‌شود

۱- مفروق منه M را با متمم r مفروق N جمع می‌کنیم. نتیجه این عمل عبارتست از:

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

۲- اگر $M\geq N$ باشد، مجموع، یک رقم نقلی r^n تولید می‌کند که چشم پوشی می‌شود، و آنچه باقی می‌ماند $M-N$ است.

۳- اگر $M<N$ باشد، مجموع، رقم نقلی تولید نکرده و برابرست با $r^n-(N-M)$، که متمم r عدد $(M-N)$ است. برای بدست آوردن جواب بصورت قابل درک، متمم r مجموع را بدست آورده و در جلو آن یک علامت منفی می‌گذاریم

برای مثال، تفریق 59282=13250-72532 را در نظر بگیرید. متمم 10 عدد 13250 برابرست با 86750. بنابراین:

M =	72532
N عدد 10 متمم =	86750
مجموع =	159282
10^5 با حذف رقم نقلی =	− 100000
جواب =	59282

حال مثالی را با $M<N$ در نظر بگیرید. تفریق 72532−13250 عدد منفی 59282را تولید می‌کند. با استفاده از متمم‌ها داریم

M =	13252
N عدد 10 متمم =	+27468
جمع =	40718

در این مثال رقم نقلی وجود ندارد

جواب نهایی 59282 است که برابر با متمم 10 عدد 40718 می‌باشد.

چون ما با اعداد بی علامت سروکار داریم، در واقع راهی برای بدست آوردن نتیجه بی علامت در مثال دوم وجود ندارد. بهنگام کار با قلم و کاغذ، تشخیص می‌دهیم که جواب باید به یک عدد علامت‌دار منفی تبدیل شود. در تفریق با روش متمم، جواب منفی از غیاب رقم نقلی نهایی و نتیجه متمم شده قابل تشخیص است.

تفریق با استفاده از روش متمم برای اعداد دودویی نیز مشابه روش فوق است. با بکارگیری دو عدد

X=1010100 و Y=1000011، تفریق X−Y و Y−X را بکمک متمم‌های 2 انجام می‌دهیم.

X =	1010100
متمم 2 عدد Y =	+ 0111101
مجموع =	10010001
حذف رقم نقلی 2^7 =	− 10000000
جواب: X−Y =	0010001
Y =	1000011
متمم 2 عدد X =	+ 0101100
مجموع =	1101111

که در آن رقم نقلی وجودندارد

جواب عدد منفی 0010001 است که متمم 2 عدد 1101111 می‌باشد

۳-۳ نمایش ممیز - ثابت

اعداد صحیح مثبت، از جمله صفر، بصورت اعداد بی‌علامت نمایش داده می‌شوند. معهذا برای نمایش اعداد صحیح منفی، ما علامتی را برای نمایش مقادیر منفی نیازداریم. در حساب معمولی اعداد منفی با علامت منفی و اعداد مثبت با علامت مثبت مشخص می‌شوند. بعلت محدودیت‌های سخت‌افزاری، کامپیوترها باید هر چیزی، حتی علامت اعداد را، با 0ها و 1ها نمایش دهند. در نتیجه، مرسوم است که علامت را در منتهاالیه سمت چپ عدد قرار دهند. بنا به قرارداد بیت علامت را برای اعداد مثبت برابر 0 و برای اعداد منفی برابر 1 در نظر می‌گیرند.

یک عدد، علاوه بر علامت، ممکن است نقطه ممیز دودویی (یا اعشاری) هم داشته باشد. مکان نقطه دودویی برای نمایش کسرها، اعداد صحیح، و اعداد مخلوط صحیح و کسری بکار می‌رود. نمایش نقطه ممیز دودویی در یک ثبات پیچیده است زیرا مکان آن باید در محلی در ثبات مشخص گردد. دو راه برای مشخص کردن محل ممیز دودویی در یک ثبات وجود دارد: یکی اختصاص یک محل ثابت به آن و دیگری استفاده از نمایش ممیز شناور می‌باشد. در روش ممیز ثابت فرض می‌شود که ممیز همیشه دریک مکان ثابت قرار گرفته است. دومکانی که بیشتر مورد استفاده‌اند عبارتند از: (۱) ممیز دودویی درمنتهاالیه سمت چپ ثبات قرار گیرد و درنتیجه عدد ذخیره شده به یک کسر تبدیل شود؛ (۲) ممیز دودویی در منتهاالیه سمت راست ثبات واقع شود و در نتیجه عدد ذخیره شده یک عدد صحیح گردد. در هر حال، ممیز دودویی واقعاً وجود خارجی ندارد، بلکه با این فرض با عدد بصورت یک کسر یا یک عدد صحیح برخورد می‌شود. در نمایش ممیز شناور با استفاده از یک ثبات دیگر و ذخیره کردن عددی در آن محل

ممیز در ثبات اول مشخص می‌شود. درباره ممیز شناور در بخش بعدی بیشتر صحبت خواهیم کرد.

نمایش صحیح

وقتی که یک عدد دودویی صحیح مثبت باشد، علامت آن با 0 و اندازه آن با یک عدد دودویی مثبت نشان داده می‌شود. هرگاه عدد منفی باشد، علامت توسط 1 ولی بقیه عدد ممکن است به یکی از سه روش زیر نمایش داده شود.

۱- نمایش مقدار (اندازه) علامت دار[1]
۲- نمایش متمم 1 علامت دار[2]
۳- نمایش متمم 2 علامت دار[3]

نمایش مقدارعلامت دار برای یک عدد منفی از مقدار و یک علامت منفی تشکیل شده است. در دو نمایش دیگر، عدد منفی به‌یکی از دو صورت متمم 1 و متمم 2 مقدار مثبت عدد نمایش داده می‌شود. مثلاً، عدد علامت دار 14 را در یک ثبات در نظر بگیرید. 14+ بوسیله بیت علامت 0که به‌دنبال آن مقدار معادل دودویی آن آمده است نشان داده می‌شود، 00001110 :14 . توجه کنید که هر یک از هشت بیت ثبات باید مقداری داشته باشند و لذا 0هایی باید در مکان‌های پرارزشتری که بدنبال بیت علامت آمده‌اند قرار گیرند. هرچند تنها یک راه برای نمایش 14+ وجود دارد، ولی برای نمایش 14− سه روش موجود است.

در نمایش مقدار علامت دار10001110
در نمایش متمم 1 علامت دار1110001
در نمایش متمم 2 علامت دار11110010

نمایش مقدار علامت‌دار برای 14- فقط از متمم کردن بیت علامت بدست می‌آید. نمایش متمم 1 علامت دار 14− با متمم کردن تمام بیت‌های 14+ از جمله بیت علامت، فراهم می‌شود. نمایش متمم 2 علامت دار با بدست آوردن متمم 2 عدد مثبت، از جمله بیت علامت حاصل می‌شود.

سیستم مقدار علامت‌دار در محاسبات معمولی بکار می‌رود، اما اگر در محاسبات کامپیوتری بکار رود کارها کُند و پیچیده خواهد شد. بنابراین معمولاً متمم علامت دار مورد استفاده قرار می‌گیرد. از متمم 1 به این علت که در آن دو نحوه نمایش 0 وجود دارد (0+ , 0−)، به ندرت، بجز در عملیات حسابی در برخی کامپیوترهای قدیمی استفاده می‌شود. متمم 1 در عملیات منطقی بکار گرفته می‌شود چون تغییر 1 به 0 یا 0 به 1 معادل با عمل متمم سازی منطقی است. در بحث زیر که درباره حساب دودویی علامت دار خواهیم کرد برای عددهای منفی منحصراً از نمایش متمم 2 علامت دار استفاده خواهد شد.

1- signed – magnitude 2- signed –1's complement 3- Signed –2's Complement

جمع حسابی

جمع دوعدد در سیستم مقدار (اندازه) علامت‌دار از قواعد حساب معمولی پیروی می‌کند. اگر علامت‌ها یکسان باشند، دو عدد را جمع می‌کنیم و علامت مشترک آنها را به مجموع می‌دهیم. اگر علامت‌ها مختلف باشند، مقدار کوچکتر را از بزرگتر کم می‌کنیم و علامت عددی که مقدار آن بزرگتر باشد را به حاصل می‌دهیم. مثلاً، 12−=(37-25)−=(-37)+(+25) که با تفریق عددی که اندازه کوچکتر دارد، یعنی 25، از عدد بزرگتر 37 و بکاربردن علامت 37 برای نتیجه حاصل انجام شده است. این روند مستلزم مقایسه علامت‌ها و مقادیر و سپس انجام جمع یا تفریق است. (روش جمع اعداد دودویی در نمایش مقدار علامت‌دار در بخش ۲-۱۰ بحث شده است) در مقابل، قاعده جمع کردن اعداد در سیستم متمم 2 نیازی به مقایسه و تفریق ندارد. رویه بسیاری ساده است و به طریق زیر می‌توان آن را بیان کرد. دو عدد را همراه با بیت‌های علامت آنها جمع کنید، و در صورت بوجود آمدن رقم نقلی از محل بیت علامت (منتهاالیه سمت چپ)، از آن صرفنظر کنید. مثال‌های عددی در زیر نشان داده شده‌اند. توجه کنید که اعداد منفی باید ابتداً بصورت متمم 2 در آیند و اگر حاصل جمع بدست آمده منفی باشد بشکل متمم 2 خواهد بود.

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

در هریک از چهارحالت، عمل انجام شده ، حتی برروی بیت علامت، جمع است. هر رقم نقلی خارج شده از بیت علامت نادیده گرفته می‌شود و نتایج منفی خودبخود به فرم متمم 2 خواهند بود.

نمایش اعداد منفی به فرم متمم برای کسانی که با سیستم مقدار علامت دار کار کرده‌اند ناآشناست . برای تعیین مقدار یک عدد منفی وقتی که بشکل متمم 2 علامت دار باشد، لازم است به عدد مثبتی تبدیل شود تا به فرم آشناتری درآید. مثلاً عدد دودویی علامت‌دار 11111001 منفی است زیرا سمت چپ‌ترین بیت 1 است. متمم 2 آن 00000111 می‌باشد که معادل عدد دودویی 7+ است. بنابراین عدد منفی اصلی 7- بوده است.

تفریق حسابی

تفریق دو عدد دودویی وقتی که اعداد منفی به شکل متمم 2 هستند بسیار ساده است و می‌تواند به

شکل زیر بیان شود: متمم 2 مفروق را بگیرید (با در نظر گرفتن بیت علامت) و آنرا با مفروق منه جمع کنید (به همراه بیت علامت). رقم نقلی خروجی از مکان بیت علامت را چشم پوشی نمائید.
این رویه از این واقعیت نتیجه شده است که یک عمل تفریق می‌تواند به یک جمع بدل شود بشرطی که علامت مفروق عوض شود. این مطلب با رابطه زیر نشان داده شده است.

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

اما تغییر یک عدد مثبت به منفی بسادگی با بدست آوردن متمم 2 امکان پذیر است. عکس این موضوع نیز صادق است زیرا متمم یک عدد منفی یک عدد معادل مثبت را بدست می‌دهد. تفریق $7+=(13)-(-6)$ را در نظر بگیرید. این تفریق بصورت هشت بیت دودویی بفرم 11111010−11110011 نوشته می‌شود. با گرفتن متمم 2 عدد (13-) که (13+) را می‌دهد تفریق به جمع تبدیل می‌شود و در نتیجه داریم 100000111=00001101+1111010. با حذف رقم نقلی انتهایی، پاسخ صحیح 00000111 یعنی (7+) بدست می‌آید.

اشاره به این نکته لازم به نظر می‌رسد که بگوئیم جمع و تفریق اعداد متمم 2 علامت دار با همان قواعد جمع و تفریق اعداد بی‌علامت صورت می‌گیرد. بنابراین، درکامپیوتر یک مدار سخت افزاری مشترک برای انجام هر دو عمل حسابی مورد نیاز است. کاربر یا برنامه نویس باید نتایج چنین جمع یا تفریق را، بسته به اینکه اعداد علامت دار فرض شوند، یا بی‌علامت بگونه‌ای متفاوت تفسیر کند.

سرریز[1]

وقتی دوعدد n رقمی جمع شده و حاصل جمع n+1 رقم را اشغال کند، گوئیم سرریز رخ داده است. وقتی جمع را با کاغذ و قلم انجام دهیم، سرریز مسئله‌ای را ایجاد نمی‌کند زیرا محدودیتی در عرض صفحه برای نوشتن حاصل جمع وجود ندارد. سرریز در کامپیوترهای دیجیتال ایجاد اشکال می‌کند زیرا عرض ثبات‌ها محدودیت دارد. نتیجه‌ای که n+1 بیتی باشد را نمی‌توان در ثباتی که دارای عرض استاندارد n بیت است جای داد. با این دلیل اغلب کامپیوترها رخداد یک سرریز را ردیابی می‌کنند، و وقتی که اتفاق بیفتد، فلیپ فلاپ بخصوصی 1 می‌شود که کاربر می‌تواند در هر زمان آن را چک کند.

درک یک سرریز پس ازجمع دو عدد دودویی بستگی به علامت دار بودن یا نبودن آنها دارد. اگر دو عدد بدون علامت باهم جمع شوند، سرریز هنگامی آشکار می‌شود که رقم نقلی از با ارزش‌ترین مکان خارج شود. در حالت علامت دار بودن عدد، سمت چپ ترین بیت همیشه علامت را نمایش می‌دهد و اعداد منفی به فرم متمم 2 خواهند بود. وقتی دو عدد علامت دار جمع شوند، بیت علامت بعنوان

1- Overflow

بخشی از عدد تلقی می‌شود و رقم نقلی مشخص کننده یک سرریز نیست.

بنابراین از جمع دو عدد که یکی مثبت و دیگری منفی باشد سرریز رخ نمی‌دهد، زیرا جمع یک عدد مثبت با یک عدد منفی نتیجه‌ای را تولید می‌کند که از بزرگترین آنها کوچکتر است . سرریز هنگامی روی می‌دهد که دوعددی که با هم جمع می‌شوند هر دو مثبت و یا هر دو منفی باشند. برای اینکه ببینیم چگونه این اتفاق ممکن است رخ بدهد مثال زیر را در نظر بگیرید. دو عدد علامت دار دودویی 70+و 80+ در دو ثبات 8 بیت ذخیره شده‌اند دامنه اعدادی که هر ثبات می‌تواند در خود جای دهد از عدد دودویی 127+تا128− است. چون مجموع 150+ است، از ظرفیت ثبات هشت بیتی تجاوز می‌کند. این وضعیت هنگامی رخ می‌دهد که اعداد هر دو مثبت یا منفی باشند. دو جمع دودویی در زیر همراه با دو رقم نقلی آخر نشان داده شده‌اند.

ارقام نقلی: 01		ارقام نقلی : 10	
+70	0 1000110	−70	1 0111010
+80	0 1010000	−80	1 0110000
+150	1 0010110	−150	0 1101010

توجه کنید که نتیجه هشت بیتی که می‌باید مثبت باشد، دارای بیت علامت منفی است و نتیجه هشت بیتی که باید منفی می‌بود دارای بیت علامت مثبت است. با این وجود اگر، رقم نقلی خارج شده از محل بیت علامت بعنوان بیت علامت در نظر گرفته شده است، جواب 9 بیتی حاصل صحیح خواهد بود. چون جواب نمی‌تواند در هشت بیت جای گیرد گوئیم سرریز رخ داده است.

حالت سرریز را با مشاهده رقم نقلی وارده به بیت علامت و نقلی خارج شدن از بیت علامت می‌توان درک کرد. اگر این دو نقلی با هم مساوی نباشد، یک وضعیت سرریز تولید شده است. این مطلب در مثال‌های فوق که هر دو رقم نقلی نشان داده شده‌اند دیده می‌شود. چنانچه دو رقم نقلی به یک گیت OR انحصاری اعمال شوند، سرریز هنگامی رخ داده است که خروجی این گیت 1 شود.

نمایش ممیز - ثابت دهدهی

نمایش اعداد دهدهی در ثبات‌ها تابعی از کد دودویی مورد استفاده برای نمایش رقم دهدهی است . یک کد دهدهی چهاربیتی نیاز به چهار فلیپ‌فلاپ برای هر رقم دهدهی دارد. نمایش 4385 در BCD به 16 فلیپ فلاپ نیاز دارد. این عدد در یک ثبات با 16 فلیپ فلاپ نمایش داده می‌شود.

0100 0011 1000 0101

با نمایش اعداد به شکل دهدهی مقدار قابل توجهی از فضای ذخیره‌سازی تلف می‌شود زیرا تعداد بیت‌های لازم برای ذخیره سازی یک عدد دهدهی درکد دودویی بزرگتر از تعداد بیت‌های لازم در نمایش معادل دودویی آن است. همچنین، مدارهای لازم برای انجام محاسبات دهدهی پیچیده‌تر هستند. با این

وجود، استفاده از نمایش دهدهی دارای مزایایی است زیرا داده‌های ورودی و خروجی کامپیوتر بوسیله انسانها که سیستم دهدهی را مورد استفاده قرار می‌دهند تولید می‌شود. در برخی از کاربردها، مانند داده‌پردازی تجاری، حجم محاسبات ریاضی لازم در مقایسه با مقدار لازم برای ورودی و خروجی داده‌های دهدهی کم است. به همین دلیل برخی از کامپیوتر و ماشین حساب‌ها اعمال حسابی را مستقیماً با داده‌های دهدهی (بصورت یک کد دودویی) انجام می‌دهند، و بنابراین در آنها نیازی به تبدیل به دودویی و تبدیل مجدد به دهدهی وجود ندارد. بعضی از سیستم‌های کامپیوتری دارای بخش سخت افزاری برای هر دو نوع محاسبه حسابی برروی داده‌های دودویی و دهدهی می‌باشند.

نمایش اعداد دهدهی علامت دار در BCD مشابه با نمایش اعداد علامت دار دودویی است. ما می‌توانیم سیستم آشنای مقدار علامت دار یا سیستم متمم علامت دار را بکار بریم. علامت یک عدد دهدهی معمولاً با چهاربیت نشان داده می‌شود تا با کد 4 بیتی ارقام دهدهی مطابقت داشته باشد. معمول است که علامت مثبت را با چهار 0 و علامت منفی را با معادل BCD عدد 9 که 1001 است نشان دهند. بکارگیری سیستم اعداد مقدار علامت دار درکامپیوترها مشکل است. سیستم متمم علامت دار هم در فرم متمم 9 و یا متمم 10 می‌باشد. ولی سیستم متمم 10 علامت دار بیشتر استفاده می‌شود. برای بدست آوردن متمم 10 یک عدد BCD، ابتدا متمم 9 را بدست آورده و سپس به کم ارزش‌ترین بیت یک واحد اضافه می‌کنیم. متمم 9 نیز از تفریق هر رقم از 9 حاصل می‌گردد.

روشهای بدست آمده برای سیستم متمم 2 علامت دار در مورد سیستم متمم 10 علامت دار برای اعداد دهدهی نیزصادق است. عمل جمع با اضافه کردن تمام ارقام، از جمله رقم علامت، و چشم پوشی از رقم نقلی نهایی انجام می‌شود. البته فرض براین است که همه اعداد منفی به شکل متمم 10 هستند. به جمع $+135 = (-240) + (+375)$ که در سیستم متمم 10 علامت دار انجام شده توجه کنید.

$$\begin{array}{rr} 0\ 375 & (0000\ 0011\ 0111\ 0101)_{BCD} \\ +9\ 760 & \underline{(1001\ 0111\ 0110\ 0000)_{BCD}} \\ 0\ 135 & (0000\ 0001\ 0011\ 0101)_{BCD} \end{array}$$

عدد 9 در سمت چپ‌ترین مکان عدد دوم نشان دهنده منفی بودن آنست. عدد 9760 متمم 10 عدد 0240 می‌باشد. دوعدد با یکدیگر جمع شده و رقم نقلی نیز چشم پوشی شده است تا $+135$ حاصل گردد. البته اعداد دهدهی، ازجمله ارقام علامت، در داخل کامپیوتر باید بشکل BCD باشند. عمل جمع با جمع‌کننده‌های BCD صورت می‌گیرد (شکل ۱۸-۱۰ را ملاحظه کنید).

تفریق اعداد دهدهی بدون علامت یا سیستم متمم 10 علامت دار همانند حالت دودویی است. متمم 10 مفروق را با مفروق منه جمع می‌کنیم. بسیاری از کامپیوترها سخت افزار خاصی را برای انجام مستقیم محاسبات حسابی با اعداد دهدهی دارند. کاربر کامپیوتر می‌تواند با کمک دستورالعمل‌هایی دربرنامه مشخص کند که اعمال حسابی مستقیماً با عددهای دهدهی و بدون تبدیل آنها به دودویی انجام شود.

۳-۴ نمایش ممیز شناور[1]

نمایش ممیز شناور یک عدد دارای دو بخش است. بخش اول یک عدد ممیز ثابت علامت دار است که مانتیس[2] خوانده می‌شود. بخش دوم محل ممیز دهدهی (یا دودویی) را معین می‌کند و نما[3] نام دارد. مانتیس ممیز ثابت، ممکن است یک عدد کسری و یا صحیح باشد. مثلاً، عدد دهدهی 6132.784 به شکل ممیز شناور و بصورت یک کسر و یک نما نشان داده می‌شود.

کسر	نما
+0.6132789	+04

مقدار نما مشخص می‌کند که مکان واقعی ممیز چهار واحد به سمت راست ممیز نشان داده شده در کسر است. این نمایش معادل با نماد علمی $0.6132789\times10^{+4}$ است.

ممیز شناور همواره برای تفسیر اعدادی به شکل کلی زیر بکار می‌رود.

$$m \times r^e$$

در ثبات‌ها فقط مانتیس m و نمای e بطور فیزیکی نمایش داده می‌شوند (از جمله علامت آنها). پایه[4] r و محل ممیز در مانتیس همیشه بطور ضمنی مساوی بامقدار معینی فرض می‌شوند. مدارهایی که برروی اعداد ممیز شناور در ثباتها عمل می‌کنند براساس این فرضیات برای ایجاد محاسبات صحیح، طراحی می‌شوند.

یک عدد ممیز شناور دودویی نیز به نحوی مشابه نشان داده می‌شود بجز اینکه از پایه 2 برای نما استفاده می‌شود. مثلاً عدد دودویی 1001.11+ با بخش کسری 8 بیتی و نمای 6 بیتی بشکل زیر نمایش داده می‌شود.

کسر	نما
01001110	000100

بخش کسری دارای یک 0 در منتهاالیه سمت چپ است که مثبت بودن آن را مشخص می‌کند. نقطه ممیز دودویی بدنبال بیت علامت آمده است ولی در ثبات نشان داده نمی‌شود. نما دارای معادل دودویی عدد 4+ است. عدد ممیز شناور معادل است با

$$m \times 2^e = + (.1001110)_2 \times 2^{+4}$$

یک عدد ممیز شناور، در صورتی که پرارزشترین رقم مانتیس غیرصفر باشد، نرمالیزه[5] (یا بهنجار)

1- Floating point
2- Mantissa
3- Exponent
4- Radix
5- Normalized

خوانده می‌شود. مثلاً، عدد دهدهی 350 نرمالیزه است ولی 00035 نرمالیزه نیست. صرف نظر از اینکه محل ممیز در کجای مانتیس فرض شود، عدد هنگامی نرمالیزه است که رقم منتهاالیه سمت چپ آن غیرصفر باشد. مثلاً عدد دودویی 00011010 به دلیل داشتن سه 0 در ابتدای آن نرمالیزه نیست. این عدد با سه بار شیفت به چپ و چشم پوشی از 0های ابتدای آن نرمالیزه می‌شود و عدد حاصل 11010000 خواهد بود. سه بار عمل شیفت عدد را در $2^3=8$ ضرب می‌کند. برای حفظ مقدار عدد ممیز شناور، باید سه واحد از نما کسر شود. فرم نرمالیزه بیشترین دقت را برای اعداد ممیز شناور فراهم می‌کند. صفر را نمی‌توان نرمالیزه کرد زیرا دارای رقم غیرصفر نیست. این عدد در نمایش ممیز شناور با مانتیس و نمای تماماً 0 نشان داده می‌شود.

عملیات حسابی با اعداد ممیز شناور پیچیده‌تر از اعمال حسابی با اعداد ممیز ثابت است و اجرای آنها طولانی تر است و بعلاوه به سخت افزار پیچیده‌تری نیاز دارد. با این وجود، نمایش ممیز شناور اغلب برای محاسبات علمی بکار می‌رود زیرا مسائل و مشکلاتی از نظر ضرب و تقسیم در نمایش ممیز ثابت وجود دارد. بسیاری از کامپیوترها و تمام ماشین‌حساب‌های الکترونیکی قابلیت درونی انجام عملیات محاسباتی ممیز شناور را دارا هستند. کامپیوترهایی که فاقد سخت افزار لازم برای محاسبات ممیز شناور می‌باشند مجموعه‌ای از دستورالعمل‌ها را برای کاربر در برنامه‌نویسی مسائل علمی، با اعداد ممیز شناور در اختیار می‌گذارند. عملیات حسابی با اعداد ممیز شناور در بخش ۵-۱۰ بحث شده‌اند.

۵-۳ انواع دیگر کدهای دودویی

در بخش‌های قبلی متداول‌ترین نوع داده‌های کد شده دودویی در کامپیوترهای دیجیتال را معرفی کردیم. گاهی اوقات از کدهای دیگری هم برای اعداد دهدهی و کاراکترهای الفبا عددی استفاده می‌شود. گاهی هم کامپیوترهای دیجیتال از کدهای دودویی دیگری برای کاربردهای خاص استفاده می‌کنند. در این بخش چند کد دودویی، که در کامپیوترهای دیجیتال با آنها مواجه می‌شویم ارائه می‌شود.

کدگری[1]

سیستم‌های دیجیتال فقط قادرند داده‌ها را بشکل گسسته پردازش کنند. بسیاری از سیستم‌های فیزیکی داده‌های خروجی پیوسته‌ای را در اختیار می‌گذارند. این داده‌ها باید، قبل از بکاررفتن بوسیله کامپیوترهای دیجیتال، به فرم دیجیتال تبدیل شوند. اطلاعات پیوسته یا آنالوگ[2] بکمک مبدل‌های آنالوگ به دیجیتال به کمیت‌های دیجیتال بدل می‌شوند. از کد انعکاسی[3] یا کدگری که در جدول ۵-۳ نشان داده شده است گاهی اوقات برای داده‌های دیجیتال تبدیل شده استفاده می‌شود. مزیت کدگری بر

1- Gray Code
2- Analog to Digital Convertor
3- Reflected Code

اعداد دودویی مستقیم این است که کدگری با رفتن از هر عدد به عدد بعدی فقط در یک بیت تغییر می‌کند. به عبارت دیگر، تغییر از هر عدد به عدد بعدی فقط مستلزم تغییر یک بیت از 0 به 1 و یا از 1 به 0 است. یکی از کاربردهای کدگری هنگامی است که داده‌های آنالوگ نشان دهنده تغییر مکان پیوسته یک محور دوار[1] باشد. محور به قسمت‌هایی تقسیم بندی می‌شود و به هر قسمت عددی تخصیص می‌یابد. اگر قطعات مجاور، متناظر با عددهای مجاور در کدگری باشند، ابهام در موقعیت شفت هنگامی که در وضعیت بین دو نقطه قرار گیرد کمتر خواهد شد.

گاهی اوقات از شمارنده‌های کدگری برای تهیه رشته‌های زمانبندی[2] (تنظیم زمان) در کنترل عملیات یک سیستم دیجیتال استفاده می‌شود. شمارنده کدگری شمارنده‌ای است که فلیپ فلاپ‌هایش رشته حالات جدول ۵-۳ را اختیار می‌کنند. این شمارنده‌ها در تغییر از یک حالت به حالت دیگر ابهامی ندارند زیرا در حالت گذار فقط یک بیت آن تغییر می‌کند

کدهای دهدهی دیگر

کدهای دودویی برای ارقام دهدهی حداقل به چهاربیت نیاز دارند. کدهای متعدد مختلفی را با کمک چهاربیت می‌توان بصورت 10 ترکیب مختلف یا بیشتر درآورد. چند نمونه از این کدها در جدول ۶-۳ نشان داده شده است.

BCD را قبلاً معرفی کردیم. در این کد از تخصیص مستقیم معادل دودویی به رقم استفاده شده است. هنگام استفاده از BCD، شش ترکیب بدون استفاده بیت ها که در جدول نشان داده شده‌اند مفهومی ندارند، درست مانند حرف H که هنگام استفاده از سمبل‌های ارقام دهدهی مفهوم ندارد. مثلاً بیان اینکه 1110 1001 یک عدد دهدهی BCD است مانند این است که بگوئیم 9H یک عدد دهدهی است که با سمبل‌های مرسوم نوشته شده باشد. در هر دوی اینها یک سمبل غیرمعتبر وجود دارد و بنابراین یک

جدول ۵-۳ کد گری 4 بیتی

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

1- Shaft

2- Timing Sequence

جدول ۶-۳ چهار کد دودویی مختلف برای ارقام دهدهی

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

عدد بی معنی را نشان می‌دهند.

یکی از معایب استفاده از BCD مشکلات ناشی از محاسبه متمم 9 عدد است. از طرف دیگر، متمم 9 در کدهای 2421 و افزونی 3[1] که در جدول ۶-۳ نشان داده شده‌اند، به آسانی قابل محاسبه است. این دو کد دارای خاصیت خود متممی[2] هستند. بدان معنی که متمم 9 عدد دهدهی هنگامی که بیکی از این دو نوع تبدیل شود براحتی با تعویض 1 ها به 0 و 0 ها به 1 بدست می‌آید. این خاصیت هنگام انجام عملیات حسابی در نمایش متمم علامت‌دار مفیدند.

2421 مثالی از کدهای وزن دار[3] است. در یک کد وزن دار بیتها در وزنه‌های مشخص شده ضرب می‌شوند، و مجموع بیت‌های وزن‌دار شده رقم دهدهی را می‌دهد. مثلاً ترکیب بیتی 1101 وقتی که در رقم‌های مربوطه در 2421 ضرب شوند، معادل دهدهی این ترکیب یعنی $2\times1+4\times1+2\times0+1\times1=7$ بدست می‌آید. به کد BCD می‌توان وزنه‌های 8421 را اختصاص داد و بهمین دلیل کد 8421 خوانده می‌شود

کد افزونی 3 یک کد دهدهی است که در کامپیوترهای قدیمی بکار می‌رفته است. این یک کد غیر وزندار است. کد دودویی اختصاص یافته این کد از عدد دودویی BCD معادل آن پس از جمع با عدد 3(0011) بدست می‌آید.

1- Excess-3 2- Self Complementing 3- Weighted Code

با توجه به جدول ۵-۳ می‌بینیم که کدگری برای 10 وارده اول جدول، کد دهدهی مناسبی نیست. این بدان علت است که انتقال از 9 به 0 مستلزم تغییر سه بیت می‌باشد (از 1101 به 0000).
برای غلبه براین مشکل، ما 10 عددرا، که از سومین وارده 0010 شروع و به 1010 ختم می‌شود، انتخاب می‌کنیم. حال گذر از 1010 به 0010 تنها تغییر یک بیت را موجب می‌شود. چون کد به میزان سه عدد به بالا شیفت داده شده است کدگری افزونی 3 خوانده می‌شود. این کد به همراه سایر کدهای دهدهی در جدول ۶-۳ آورده شده است.

کدهای الفبا عددی دیگر

کد ASCII (جدول ۴-۳) کد استانداری است که معمولاً برای ارسال اطلاعات دودویی بکار می‌رود. هر مشخصه بوسیله یک کد هفت بیتی نمایش داده می‌شود و معمولاً بیت هشتمی هم برای توازن به آن اضافه می‌گردد. (بخش ۶-۳ ملاحظه شود). کد دارای 128 کاراکتر می‌باشد. نود و پنج کاراکتر سمبل‌های گرافیکی[1] را نمایش می‌دهند که شامل حروف بزرگ و کوچک، ارقام 0 تا 9، علائم نقطه گذاری و سمبل‌های خاص هستند. بیست و سه کاراکتر نماینده شکل‌دهنده‌ها[2] می‌باشند که شامل کاراکترهای عملیاتی برای کنترل دستگاه‌های چاپ و یا نمایشگرها از قبیل برگشت نورد[3]، تعویض سطر[4]، جدول‌بندی افقی[5]، و پَسبَر[6] می‌باشند. 10 کاراکتر دیگر برای هدایت جریان انتقال داده‌ها به کارمی‌روند و وضعیت آن راگزارش می‌دهد.
کد الفباء عددی دیگر که در دستگاه‌های IBM بکار می‌رود EBCDIC[7] می‌باشد. در این کد از هشت بیت برای هر کاراکتر استفاده می‌شود (بیت نهم متعلق به توازن است). کارکترهای EBCDIC همان کاراکترهای ASCII می‌باشند ولی بیت‌های اختصاص یافته به آنها متفاوت است.
هرگاه کاراکترهای الفباء عددی برای پردازش داخلی در کامپیوتر بکار روند (ونه به منظور ارسال داده‌ها) بهتر است از یک کد 6 بیتی برای نمایش 64 کاراکتر استفاده شود. با یک کد 6 بیتی می‌توان 26 حرف بزرگ، ارقام 0 تا 9، و تا 28 کاراکتر خاص را مشخص کرد. این مجموعه از کاراکترها معمولاً برای اهداف پردازش داده کافی است. بکار گرفتن بیت‌های کمتر برای کد کاراکترها این مزیت را دارد که فضای حافظه لازم برای ذخیره سازی کمیت‌های داده‌های الفبایی کاهش می‌یابد.

۶-۳ کدهای آشکارسازی خطا

اطلاعات دودویی انتقالی از طریق برخی از وسایل ارتباطی در معرض پارازیت‌های[8] خارجی قرار دارند که ممکن است بیت‌ها را از 0 به 1 و بالعکس تبدیل کند. کد کشف خطا یک کد دودویی است که

1- Graphic
2- Format Effectors
3- Carriage Retarn
4- Line Feed
5- Horizontal Tabulation
6- Back Space
7- Extended BCD Interchange Code
8- Noise

خطاهای رقمی را در طول ارسال کشف می‌کند. خطاهای درک شده را نمی‌توان اصلاح کرد و فقط وجود آنها معلوم می‌شود. معمول این است که تواتر خطاها مشاهده شود. اگر خطاها غیرمتوالی و تصادفی رخ دهند، اطلاعات خاصی که خطا دارد مجدداً ارسال می‌گردد. اگر خطا بیش حد معقول رخ دهد سیستم از نظر وجود اشکال چک می‌شود.

متداول‌ترین کد کشف خطا، بیت توازن[1] است. بیت توازن بیتی است اضافی که به یک پیام دودویی اضافه می‌شود تا تعداد 1ها را در آن زوج یا فرد نماید. یک پیام سه بیتی و دونوع بیت توازن ممکن در جدول ۷-۳ نشان داده شده است. بیت فرد P چنان اختیار می‌شود تا مجموع تمام 1ها را فرد نماید (در تمام چهاربیت). بیت زوج P چنان اختیار می‌شود تا مجموع تمام 1ها را زوج نماید. در هر صورت مجموع 1ها در پیام ارسالی و بیت P در نظر گرفته می‌شود. در هر کاربرد خاص یکی از انواع توازن بکار می‌رود. روش توازن زوج این اشکال را دارد که یکی از ترکیبات بیت‌ها تماماً 0 خواهد داشت، در حالی که در توازن فرد همیشه یکی از بیت‌ها (از چهاربیتی که پیام و P را تشکیل می‌دهند) 1 خواهد بود. دقت کنید که P فرد متمم P زوج است.

هنگام انتقال اطلاعات از یک مکان به مکانی دیگر، با بیت توازن باین صورت برخورد می‌شود که در سمت فرستنده، پیام به یک مولد توازن[2] اعمال می‌شود تا بیت توازن لازم توسط آن تولید شود. پیام، همراه با بیت توازن به مقصد ارسال می‌گردد. در سمت گیرنده، تمام بیت‌های رسیده (دراین حالت چهاربیت) به یک چک کننده توازن[3] اعمال می‌شوند تا توازن انتخابی را چک کند (فرد یا زوج). اگر توازن بیت‌ها با توازن انتخابی مغایرت داشته باشد، خطا اعلام می‌شود. روش توازن می‌تواند وجود یک، سه یا هر تعداد فردی از خطا را کشف کند. اگر تعداد خطاها زوج باشند کشف نمی‌شوند.

شبکه‌های مولد و چک‌کننده توازن مدارهایی منطقی هستند که با توابع OR انحصاری ساخته می‌شوند. دلیل این امر این است که، همانطورکه در بخش ۲-۱ ذکر شد، توابع OR انحصاری برای سه یا

جدول ۷-۳ تولید بیت توازن

پیام *xyz*	فرد (*P*)	زوج (*P*)
000	1	0
001	0	1
010	0	1
011	1	0
100	0	1
101	1	0
110	1	0
111	0	1

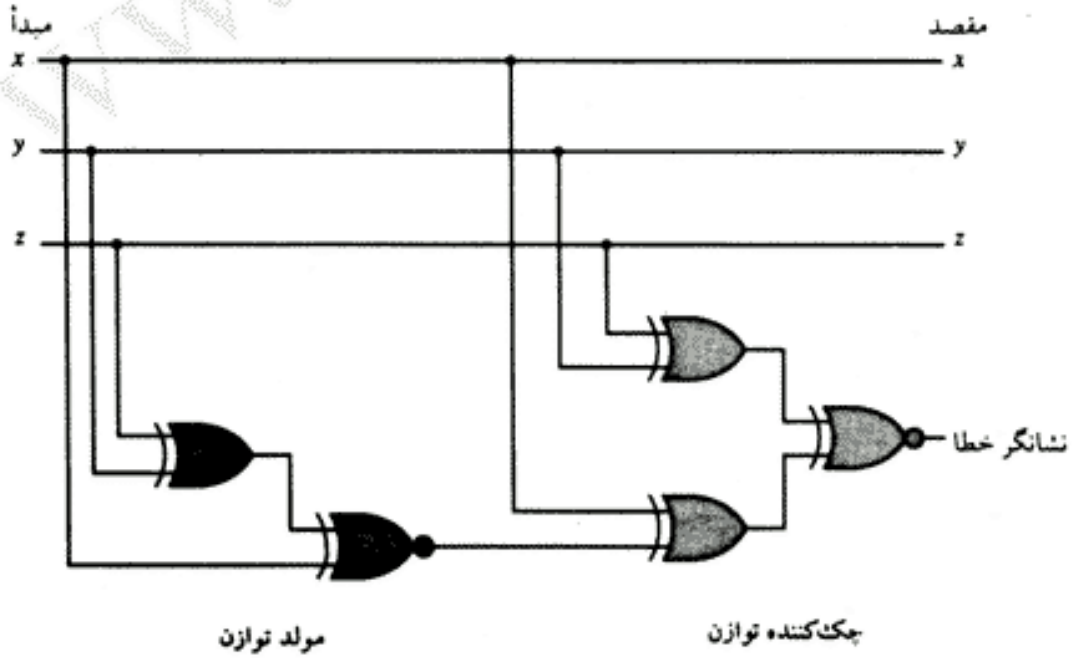
1- Parity Bit 2- Parity Generator 3- Parity Checker

چند متغیر بنا به تعریف یک تابع فرد است. تابع فرد تابعی منطقی است که مقدار آن 1 دودویی است، اگر و فقط اگر تعداد فردی از متغیرها 1 باشند. طبق این تعریف، تابع زوج P عبارتست از OR انحصاری x ، y و z ، زیرا هر گاه یکی از این سه متغیر 1 یا هر سه آنها 1 باشند، تابع برابر 1 خواهد بود (جدول ۳-۷). تابع فرد متمم تابع زوج است.

بعنوان یک مثال، یک پیام سه بیتی راکه قرار است با بیت توازن فرد ارسال شود در نظر بگیرید. در سمت فرستنده، بیت توازن فرد بوسیله یک مولد توازن تولید می‌شود. این مدار، همانطور که در شکل ۳-۳ دیده می‌شود، از یک OR انحصاری و یک گیت NOR انحصاری تشکیل شده است. چون P زوج، OR انحصاری z,y,x و P فرد متمم P زوج است، برای متمم لازم باید یک گیت NOR انحصاری بکار گرفته شود. پیام و بیت توازن فرد به مقصدشان ارسال و به یک چک کننده توازن اعمال می‌شوند. در حین ارسال پیام اگر توازن چهاربیت رسیده زوج باشد خطایی رخ داده است، زیرا اطلاعات ارسالی ابتداً فرد بوده است. اگر خطایی رخ داده باشد خروجی چک کننده توازن 1 خواهد شد، یعنی تعداد 1های چهار ورودی زوج است. چون تابع OR انحصاری چهارورودی یک تابع فرد است، ما می‌باید مجدداً با استفاده از یک گیت NOR انحصاری، خروجی را متمم کنیم.

ذکر این نکته مفید است که بدانیم مولد توازن می‌تواند همان مدار چک کننده توازن را داشته باشد بشرطی که ورودی چهارم دائماً در 0 منطقی نگهداشته شود. مزیت این خاصیت این است که برای تولید و چک کردن توازن می‌توان از یک مدار استفاده کرد.

با توجه به مثال فوق واضح است که مولدها وچک‌کننده‌های توازن با توابع OR انحصاری پیاده سازی می‌شوند. مدارات توازن فرد برای متمم کردن خروجی تابع به یک NOR انحصاری نیاز دارند.



شکل ۳-۳ کشف خطا بوسیله بیت توازن فرد

مسائل

۳-۱ اعداد دودویی زیر را به دهدهی تبدیل کنید: 101110؛ 1110101؛ 110110100

۳-۲ اعداد زیر، که در آنها پایه‌هایشان مشخص شده است را به دهدهی تبدیل کنید: $(12121)_3$؛ $(4310)_5$، $(50)_7$؛ $(198)_{12}$.

۳-۳ اعداد دهدهی زیر را به دودویی تبدیل کنید: 1231؛ 673 و 1998 .

۳-۴ اعداد دهدهی زیر را به مبناهای مشخص شده تبدیل کنید

الف) 7562 به هشت هشتی ب) 1938 به شانزده شانزدهی ج) 175 به دودویی

۳-۵ عدد شانزده شانزدهی F3A7C2 را به دودویی و هشت هشتی تبدیل نمائید.

۳-۶ اگر جواب معادله $x^2-10x+31=0$ برابر $x=5$ و $x=8$ باشد. پایه عددها چیست؟

۳-۷ مقدار همه بیت‌های یک ثبات 12 بیت که عددی معادل 215 دهدهی را نگهمیدارد:

(الف) به دودویی؛ (ب) هشت هشتی کد شده بادودویی، (ج) شانزده شانزدهی کد شده با دودویی

(د) دهدهی کد شده با دودویی (BCD) ، نشان دهید.

۳-۸ آرایش یک ثبات 24 بیتی را که محتوایش معادل یک عدد دهدهی 295 است؛ (الف) به دودویی؛ (ب) به BCD؛ (ج) به ASCII با استفاده از هشت هشتی با توازن زوج، نشان دهید.

۳-۹ نام خود را با استفاده از یک کد هشت بیتی به ASCII نوشته و بیت سمت چپ منتهاالیه را در نظربگیرید. بین نام و نام خانوادگی خود از کارکتر فضای خالی استفاده کنید و در صورت استفاده از فقط حرف اول نام، پس از آن نقطه بگذارید.

۳-۱۰ کد ASCII زیر را دیکد کنید

1001010 1001111 1001000 1001110 0100000 1000100 1001111 1000101

۳-۱۱ متمم 9 اعداد دهدهی هشت رقمی زیر را بدست آورید.

12349876 ; 00980100 ' 90009951 ; , 00000000

۳-۱۲ متمم 10 اعداد دهدهی شش رقمی زیر را بدست آورید.

123900; 090657 ; 100000; , 000000

۳-۱۳ متمم‌های 1 و 2 اعداد دودویی هشت رقمی زیر را بدست آورید

10101110 ; 10000001 ; 10000000; 00000001; 00000000

۳-۱۴ تفریق را با اعداد دهدهی بدون علامت زیر و با استفاده از متمم 10 مفروق انجام دهید.

الف) 5250-1321 ب) 1753-8640

ج) 20-100 د) 1200-250

۳-۱۵ تفریق را با اعداد دودویی بدون علامت زیر و با استفاده از متمم 2 مفروق انجام دهید.

الف) 11010-10000 ب) 11010-1101

ج) 100-110000 د) 1010100-1010100

۳-۱۶ اعمال حسابی (13−)+(42+) و (13−)−(42−) را در دودویی و با استفاده از متمم 2 علامت‌دار برای اعداد منفی انجام دهید.

۳-۱۷ اعمال حسابی (80+)+(70+) و (80−)+(70−) را با نمایش اعداد دودویی به شکل متمم 2 علامت‌دار انجام دهید. از هشت بیت برای نمایش هر عدد به همراه علامت آن استفاده کنید. نشان دهید که در هر مورد سرریز رخ می‌دهد، دو نقلی آخر نامساویند، و علامت معکوس می‌شود.

۳-۱۸ اعمال حسابی زیر را برای اعداد دهدهی مشخص شده با استفاده از نمایش متمم 10 علامت دار برای اعداد منفی انجام دهید

الف) (+785) +(−638) ب) (+785)−(-638)

۳-۱۹ یک عدد دودویی ممیز شناور 36 بیتی دارای 8 بیت بعلاوه علامت برای نما و 26 بیت بعلاوه علامت برای مانیتس می‌باشد. مانتیس یک کسرنرمالیزه شده است. اعداد مانتیس و نما به فرم مقدار علامت‌دار هستند. بزرگترین و کوچکترین عددی که با این شکل می‌توان نشان داد، بجز صفر، کدامند؟

۳-۲۰ عدد $(46.5+)_{10}$ را بصورت یک عدد دودویی ممیز شناور 24 بیتی نشان دهید. مانتیس که یک کسر نرمالیزه است 24 بیت و نما 8 بیت دارند.

۳-۲۱ کدگری گاهی اوقات کدانعکاسی خوانده می‌شود زیرا مقادیر بیت‌ها در دوطرف هریک از توان‌های 2^n انعکاس یافته یکدیگرند. مثلاً همانطور که در جدول 3-5 ملاحظه می‌شود، مقدار سه بیت پائین رتبه در دو طرف خطی که بین 7 و 8 کشیده شود انعکاس یکدیگرند. با استفاده از این خاصیت کدگری مطلوبست:

الف) اعداد کدگری برای 16 تا 31 بعنوان ادامه جدول ۳-۵

ب) کدگری افزونی 3 برای اعداد دهدهی 10 تا 19 به عنوان ادامه جدول ۳-۶

۳-۲۲ عدد دهدهی 8620 را به شکل (الف) BCD؛ (ب) افزونی 3؛ (ج) کد 2421 ؛ (د) عددودویی نمایش دهید.

۳-۲۳ ده رقم BCD را با توازن زوج در منتهاالیه سمت چپ آن (کلاً پنج بیت در هر رقم) نشان دهید. عمل را با توازن فرد تکرار کنید.

۳-۲۴ عدد دهدهی 3984 را بصورت کد 2421 جدول ۳-۶ نشان دهید. تمام بیت‌های عدد کد شده را متمم کنید و نشان دهید که متمم 9 همان عدد 3984 با کد 2421 است.

۳-۲۵ نشان دهید که تابع OR انحصاری $x=A\oplus B\oplus C\oplus D$ یک تابع فرد است. یکی از راه‌ها این است که جدول درستی $y=A\oplus B$ و $z=C\oplus D$ را بدست آورده و سپس جدول درستی $x=y\oplus z$ را بدست آورید. نشان دهید که $x=1$ است بشرطی که تعداد 1ها در A، B و C و D فرد باشد.

۳-۲۶ مدارهای یک مولد توازن سه بیتی و چک کننده توازن چهاربیتی را با استفاده از بیت توازن زوج بدست آورید (مدارات شکل ۳-۳ توازن فرد را بکار برده‌اند).

٤ انتقال ثبات‌ها و ریزعمل‌ها

۴-۱ زبان انتقال ثبات
۴-۲ انتقال ثبات
۴-۳ انتقال‌های گذرگاهی و حافظه‌ای
۴-۴ ریزعمل‌های حسابی
۴-۵ ریزعمل‌های منطقی
۴-۶ ریزعمل‌های شیفت
۴-۷ واحد حساب، منطق و شیفت

۴-۱ زبان انتقال ثبات

یک سیستم دیجیتال مجموعه‌ای از ماژول‌های[1] سخت افزاری متصل بهم است که کاری خاص را در زمینه پردازش اطلاعات انجام می‌دهند. سیستم‌های دیجیتال از نظر اندازه و پیچیدگی از چند مدار مجتمع تا مجموعه‌ای از کامپیوترهای مرتبط متغیرند. در طراحی سیستم‌های دیجیتال از روش ماژولی استفاده می‌شود. ماژول‌ها از اجزایی[2] چون ثبات‌ها، دیکدرها عناصر حسابی و کنترل منطقی ساخته می‌شوند. ماژول‌های مختلف با مسیرهای مشترک داده و کنترل به هم متصل می‌شوند تا یک سیستم کامپیوتر دیجیتال بوجود آید.

ماژول‌های دیجیتال در بهترین فرم براساس ثبات‌ها و عملیاتی که روی داده‌های ذخیره شده در آنها انجام می‌شود تعریف می‌گردند. عملیاتی که روی داده‌های ذخیره شده در ثبات‌ها صورت می‌گیرد ریز عمل[3] نام دارد. به بیان دیگر ریز عمل یک جزء عملیاتی است که بر روی اطلاعات ذخیره شده در یک یا چند ثبات انجام می‌شود. نتیجه عمل ممکن است جایگزین اطلاعات دودویی قبلی در ثبات شود و یا به ثبات دیگری انتقال یابد. مثال‌هایی از ریزعمل عبارتند از، شیفت[4]، شمارش[5]، پاک کردن[6] و بارکردن[7]

1- Module
2- Component
۳- Microoperation
4- Shift
5- Count
6- Clear
7- Load

است. برخی از مؤلفه‌های دیجیتال که در فصل 2 معرفی شدند ثبات‌هایی هستند که ریزعمل‌ها را پیاده سازی می‌کنند. مثلاً، یک شمارنده با قابلیت بارشدن موازی می‌تواند ریزعمل‌های افزایش و بارشدن را اجراکند. یک ثبات شیفت (جابجایی) دوطرفه قادر است ریزعمل‌های شیفت به راست و چپ را انجام دهد.

سازمان داخلی یک کامپیوتر دیجیتال به بهترین نحو توسط موارد زیر مشخص می‌شود:

۱- مجموعه ثبات‌های آن و وظایف آنها

۲- رشته ریزعمل‌های انجام شده روی اطلاعات دودویی ذخیره شده در ثبات‌ها

۳- واحد کنترلی که موجب آغاز رشته ریز عمل‌ها می‌شود.

رشته ریزعمل‌ها در کامپیوتر را می‌توان با تشریح لفظی هر عمل مشخص کرد، ولی این روش معمولاً تشریحی طولانی خواهد بود. بهتر آنست تا روشی سمبلیک را برای توصیف رشته انتقال‌ها بین ثبات‌ها و ریزعمل‌های حسابی و منطقی مختلف مربوط به این انتقال‌ها، مشخص کنیم. استفاده از سمبل بجای توضیحات، روشی سازمان یافته و فشرده‌ای را برای نشان دادن رشته ریزعملها در ثبات ها و توابع کنترلی که موجب اجرای آنها می‌شوند فراهم می‌کند.

نحوه بیان سمبلیک مورد استفاده برای بیان انتقال‌های ریزعملی در بین ثبات‌ها، زبان انتقال ثباتها خوانده می‌شود. اصطلاح "انتقال ثباتها" بیانگر وجود مدارات منطقی سخت افزاری است که می‌تواند یک ریزعمل بیان شده را اجرا نماید و نتیجه عمل را به همان ثبات یا ثبات دیگر انتقال می‌دهد. کلمه "زبان" از برنامه نویسان اقتباس شده است، که این کلمه را برای زبان‌های برنامه نویسی بکار می‌برند. زبان برنامه نویسی رویه‌ای برای نوشتن سمبل‌هایی است که فرآیند محاسباتی خاصی را مشخص می‌کند. بطور مشابه، یک زبان طبیعی مانند انگلیسی، سیستمی است برای نوشتن سمبل‌ها و ترکیب آنها بصورت کلمات و جملات برای ارتباط بین انسانها. زبان انتقال ثبات هم سیستمی است برای بیان رشته ریزعمل‌ها بین ثبات‌های یک ماژول دیجیتال بصورت سمبلیک. چنین زبانی ابزار مناسبی برای توصیف فشرده و دقیق سازمان داخلی کامپیوترهای دیجیتال است. همچنین می‌توان از آن برای تسهیل روند طراحی سیستم‌های دیجیتال استفاده کرد.

اعتقاد براین است که زبان انتقال ثبات بکار رفته تا حد امکان ساده‌باشد، بنابراین بخاطر سپردن آن طولی نخواهد کشید. ما همچنان به تعریف سمبل برای انواع ریزعمل‌ها ادامه خواهیم داد، و همزمان با آن سخت افزاری که ریزعمل بیان شده را پیاده سازی کند معرفی می‌کنیم. علائم سمبلیک معرفی شده در این فصل، در فصل‌های بعدی برای مشخص کردن انتقالات ثبات‌ها ، ریزعمل‌ها، و توابع کنترلی که سازمان سخت افزاری داخلی کامپیوترهای دیجیتال را بیان می‌کنند، بکارمی‌روند. به محض آشنایی با این زبان، سایر سمبل‌های بکار رفته بسادگی قابل آموختن است زیرا بیشتر تفاوتهای بین زبان‌های انتقال ثبات در جزئیات آنهاست، نه در کلیات.