

شکل ۱۸-۷. وسیله منطقی ترتیبی برنامه پذیر

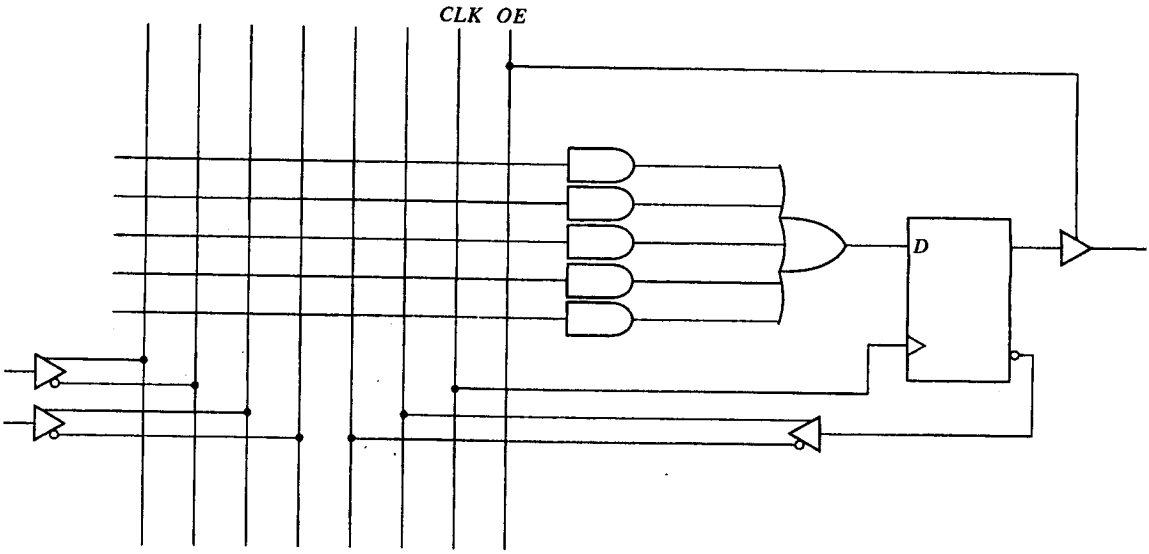
PLD ترتیبی را گاهی PLD ساده (SPLD) هم می‌گویند تا آن را از نوع PLD پیچیده تفکیک کنند. SPLD علاوه بر آرایه AND-OR حاوی فلیپ فلاپ‌ها در تراشه مدار مجتمع هم می‌باشد. نتیجه این ترکیب مدار ترکیبی شکل ۱۸-۷ است. در PLA یا PAL برای ایجاد یک ثبات، فلیپ فلاپ‌ها لحاظ شده‌اند. خروجی‌های مدار را از گیت‌های OR یا از خروجی‌های فلیپ فلاپ می‌توان دریافت کرد. مقداری اتصالات برنامه‌پذیر نیز برای ایجاد جملات ضرب حاصل از خروجی فلیپ فلاپ‌ها به کمک آرایه AND لحاظ شده است. فلیپ فلاپ‌ها می‌توانند از نوع D یا JK باشند.

اولین وسیله برنامه‌پذیری که پیاده‌سازی مدار ترتیبی را پشتیبانی می‌کند توالی‌گر منطقی برنامه‌پذیر موردی (FPLS) است. نمونه‌ای از FPLS حول یک PLA با چند خروجی که فلیپ فلاپ‌ها را راه می‌اندازند تشکیل شده است. به لحاظ این که فلیپ فلاپ‌ها می‌توانند به عنوان نوع D یا JK کار کنند، انعطاف‌پذیرند. FPLS از نظر تجارتی موفقیتی نداشت زیاد تعداد اتصالات برنامه‌پذیر آن فوق‌العاده زیاد بود. رایج‌ترین نوع SPLD، PAL ترکیبی همراه با فلیپ فلاپ‌های D است.

PAL حاوی فلیپ فلاپ را PAL ثبات‌دار گویند تا بدین وسیله مشخص شود که وسیله علاوه بر آرایه AND-OR دارای فلیپ فلاپ نیز هست. هر بخش از یک SPLD را ماکروسل گویند. ماکروسل مداری است که حاوی جمع حاصلضرب، تابع منطقی ترکیبی و یک فلیپ فلاپ اختیاری است. در اینجا فرض خواهیم کرد که جمع حاصلضرب AND-OR به کار رفته باشد ولی در عمل می‌تواند هر یک از پیاده‌سازی‌های دو سطحی، بخش ۷-۳، به کار گرفته شود.

شکل ۱۹-۷ نمودار منطقی یک ماکروسل را نشان می‌دهد. آرایه AND-OR مثل آنچه در PAL ترکیبی شکل ۱۶-۷ دیدیم می‌باشد. خروجی به وسیله یک فلیپ فلاپ D حساس به لبه راه‌اندازی می‌شود. فلیپ فلاپ به یک ورودی ساعت متصل است و حالتش با لبه ساعت تغییر می‌کند. خروجی فلیپ فلاپ به بافر سه حالته متصل است (یا وارونگر) که این بافر نیز با یک سیگنال فعال‌ساز خروجی به نام OE کنترل می‌شود. خروجی فلیپ فلاپ به یکی از ورودی‌های گیت‌های AND برنامه‌پذیر برای تهیه حالت فعلی مدار ترتیبی پسخورد شده است. نمونه‌ای از SPLD دارای 8 تا 10 ماکروسل در یک بسته آی‌سی است. همه فلیپ فلاپ‌ها به یک ورودی CLK مشترک وصلند و همه بافرهای سه حالته با ورودی OE کنترل می‌شوند. با برنامه‌ریزی ورودی‌های انتخاب می‌توان از مولتی پلکسرها برای انتخاب دو یا چهار مسیر مجزا، استفاده کرد.

علاوه بر آرایه AND، یک ماکروسل دارای امکانات برنامه‌ریزی دیگری نیز هست. انتخاب‌هایی از

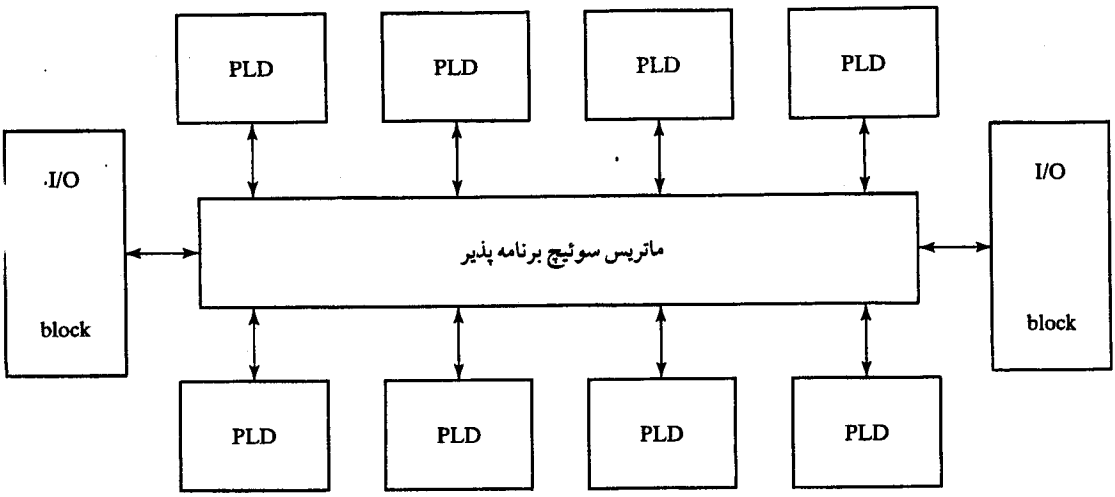


شکل ۱۹-۷. مدار منطقی ماکروسل‌پایه

این نوع امکانات شامل توانایی در استفاده یا نادیده گرفتن فلیپ فلاپ، انتخاب لبه ساعت، انتخاب پاک کردن و پیش تنظیم برای ثبات، و انتخاب خروجی صحیح یا متمم است. یک گیت XOR برای برنامه‌ریزی حالت صحیح/ متمم بکار رفته است.

طراحی یک سیستم دیجیتال با استفاده از PLD اغلب نیاز به اتصال چندین وسیله برای تکمیل ویژگی‌ها دارد. برای این نوع کاربرد، اقتصادی‌تر است از یک وسیله منطقی برنامه‌پذیر پیچیده (CPLD) استفاده شود. CPLD مجموعه‌ای از PLD های منفرد روی یک مدار مجتمع می‌باشد. ساختار اتصالات برنامه‌پذیر ارتباط PLD را به یکدیگر به همان طریق PLD های منفرد امکان پذیر می‌سازد.

شکل ۲۰-۷ آرایش کلی یک CPLD است. این مدار از چند PLD مرتبط به هم که از طریق یک ماتریس سوئیچ برنامه‌ریزی می‌شود تشکیل شده است. بلوک‌های ورودی/ خروجی (I/O) اتصالات به پایه‌های آی‌سی را مهیا می‌سازد. هر پایه I/O با یک بافر سه حالته راه‌اندازی می‌شود و می‌تواند به عنوان خروجی یا ورودی برنامه‌ریزی شود. ماتریس سوئیچ ورودی‌ها را از بلوک I/O دریافت می‌کند و آن را به سمت ماکروسل خاصی هدایت می‌نماید. به همین طریق، خروجی‌های انتخابی از ماکروسل‌ها به خروجی‌های موردنظر ارسال می‌شوند. معمولاً هر PLD حاوی 8 تا 16 ماکروسل است. ماکروسل‌های داخل هر PLD معمولاً به طور کامل وصلند. اگر ماکروسلی جملات ضرب به کار نرفته‌ای داشته باشد می‌توان از آنها در ماکروسل‌های مجاور استفاده کرد. در بعضی از موارد فلیپ فلاپ ماکروسل را می‌توان به عنوان D ، JK یا T برنامه‌ریزی کرد. سازندگان مختلف، راه‌های متفاوتی را برای معماری کلی CPLD ها به کار گرفته‌اند. زمینه‌های اختلاف عبارتند از PLD های درونی خاص (گاهی به آنها



شکل ۲۰-۷. آرایش کلی CPLD

بلوک‌های تابعی می‌گویند)، نوع ماکروسل، بلوک‌های I/O و ساختار اتصالات درونی برنامه‌پذیر. بهترین راه شناسایی وسیله خاص یک سازنده مطالعه مقالات سازنده است.

قطعه اصلی به کار رفته در طراحی VLSI آرایه گیتی است، یک آرایه گیتی از الگویی از گیت در یک ناحیه از سیلیکان ساخته شده و هزاران بار تکرار شده است تا تمام تراشه با گیت پوشش یابد. ابعاد آرایه از یک هزار تا یکصدهزار گیت بسته به تکنولوژی به کار رفته در یک تراشه می‌باشد. طراحی با آرایه‌های گیتی لازم می‌دارد تا مشتری الگوی موردنظر خود را برای سازنده فراهم کند. چند سطح اول فرآیند ساخت مشترک بوده و مستقل از تابع منطقی نهایی است. مراحل بعدی ساخت لازم می‌دارد تا اتصالات درونی گیت‌ها طبق مشخصات داده شده به وسیله طراح به هم وصل شوند.

آرایه گیتی برنامه‌پذیر موردی (FPGA) یک مدار VLSI است که قابل برنامه‌نویسی در محل کاربر است. بعضی از آنها آرایه‌ای متشکل از هزاران بلوک منطقی دارند، که به وسیله بلوک‌های ورودی و خروجی برنامه‌پذیر محصور شده‌اند و از طریق اتصالات قابل برنامه‌ریزی به هم متصل می‌گردند. آرایش‌های داخلی بسیار متنوعی در این گروه از وسایل وجود دارد. رفتار هر وسیله به مدار داخل بلوک و راندمان اتصالات برنامه‌ریزی شده بستگی دارد.

نوعی از بلوک منطقی FPGA از جداول نظاره، مولتی پلکسرها، گیت‌ها و فلیپ فلاپ‌ها ساخته شده است. جداول نظاره نوعی جدول درستی ذخیره شده در SRAM است و توابع مدار ترکیبی را برای بلوک منطقی فراهم می‌کند. این توابع از جدول درستی ذخیره شده در SRAM و شبیه به روش پیاده‌سازی مدارهای ترکیبی با ROM تحقق می‌یابند، بخش ۵-۷. مثلاً یک SRAM 2 × 16 می‌تواند جدول درستی یک مدار ترکیبی که چهار ورودی و دو خروجی دارد را ذخیره نماید. بخش منطق ترکیبی همراه با تعدادی مولتی پلکسرهای برنامه‌پذیر برای بدست آوردن معادلات ورودی به فلیپ فلاپ و

خروجی بلوک منطقی به کار می‌رود.

مزیت استفاده از RAM به جای ROM در ذخیره جدول درستی این است که جدول با نوشتن در حافظه قابل برنامه‌نویسی است. عیب آن هم این است که حافظه فرار بوده و هنگام قطع برق باید محتوای جدول نظاره دوباره بار شود. برنامه را می‌توان از کامپیوتر مرکزی یا از PROM های روی بورد بار کرد. برنامه تا برنامه‌ریزی مجدد FPGA و یا تا قطع برق در SRAM باقی می‌ماند. هر بار که برق وصل شود وسیله باید دوباره برنامه‌ریزی گردد. قابلیت برنامه‌ریزی مجدد FPGA کاربردهای زیادی را با پیاده‌سازی مدارهای منطقی مختلف در آن ممکن می‌سازد.

طراحی با PLD ، CPLD یا FPGA نیاز به ابزارهای قوی طراحی کامپیوتری CAD دارد تا روال طراحی آسان شود. ابزارهای متعددی مانند بسته‌های ورود شماتیک و زبان‌های توصیف سخت‌افزاری (HDL) مانند ABEL ، VHDL . Verilog موجودند. ابزارهای طراحی موجود بلوک‌های منطقی را مستقر، آرایش و به هم وصل می‌کنند تا با یک زبان سطح بالا در HDL تحقق یابد.

مسائل

۷-۱ واحدهای حافظه زیر با تعداد کلمات و تعداد بیت در هر کلمه مشخص می‌شود. در هر حالت چند خط آدرس و چند خط ورودی- خروجی داده وجود دارد؟
(الف) 16 × 4K ، (ب) 8 × 2G ، (پ) 32 × 16M ، (ت) 64 × 256K

۷-۲ تعداد بایت‌های ذخیره شده در حافظه‌های مسئله ۱-۷ را مشخص کنید.

۷-۳ شماره کلمه 723 حافظه در شکل ۳-۷، حاوی معادل دودویی 3451 است. آدرس 10 بیتی و محتوای حافظه 16 بیتی کلمه را لیست نمایید.

۷-۴ شکل موج زمانبندی سیکل حافظه برای اعمال نوشتن و خواندن را نشان دهید. ساعت CPU را 25MHz و سیکل حافظه را 60ns در نظر بگیرید.

۷-۵ یک برنامه تست برای حافظه توصیف شده در مثال ۱-۷ HDL بنویسید. برنامه تست عدد دودویی 7 را در آدرس 0 ، و 14 دودویی را در آدرس 60 ذخیره می‌کند. آنگاه دو آدرس برای اطمینان از ذخیره شدن خوانده می‌شوند.

۷-۶ حافظه RAM 4 × 4 شکل ۶-۷ را در یک نمودار بلوکی محصور کنید و تمام ورودی‌ها و خروجی‌ها را نشان دهید. با فرض خروجی‌های سه حالته، یک حافظه 8 × 8 با چهار RAM 4 × 4 بسازید.

۷-۷ یک حافظه 4 × 16K از یک روش دیکدکردن متقارن (coincident) با تقسیم دیکدر داخلی به دو بخش X و Y استفاده می‌کند.
(الف) سایز هر دیکدر چقدر است و چندگیت AND برای دیکد آدرس لازم است؟
(ب) وقتی آدرس ورودی معادل 6000 دودویی است خطوط انتخاب X و Y فعال شده را معین کنید.

۷-۸ (الف) چند تراشه RAM 8 × 32K برای تهیه یک حافظه با ظرفیت 256K لازم است؟

(ب) چند خط آدرس برای دستیابی به 256K بایت لازم است؟ چند خط از آنها به ورودی‌های آدرس همه تراشه‌ها متصل می‌شوند.

(پ) برای ورودی‌های انتخاب تراشه چند خط باید دیکدگردد.

۷-۹ یک تراشه DRAM از مولتی پلکس کردن دوبعدی استفاده می‌کند. این تراشه دارای 13 خط آدرس مشترک است که در آن آدرس سطر 1 بیت بیشتر است. ظرفیت حافظه چقدر است.

۷-۱۰ با فرض کلمه داده 8 بیت 01011011، یک کلمه مرکب 13 بیت برای کد همینگ تولید کنید تا یک خطا را اصلاح و دو خطا را تشخیص دهد.

۷-۱۱ کد همینگ 15 بیت را برای کلمه داده 11 بیت 11001001010 بدست آورید.

۷-۱۲ یک کد همینگ 12 بیت حاوی 8 بیت داده و 4 بیت توازن از حافظه خوانده شده است. اگر کلمه 12 بیت خوانده شده برابر زیر باشد کلمه داده 8 بیت اولیه نوشته در حافظه چیست:

(الف) 000011101010 (ب) 101110000110 (پ) 101111110100

۷-۱۳ چند بیت چک یا تست توازن باید در کلمه داده لحاظ شود تا برای موارد زیر یک خطا تصحیح و دو خطا تشخیص داده شود (الف) 16 بیت (ب) 32 بیت (پ) 48 بیت.

۷-۱۴ در صورت لزوم کد همینگ را برای چهار بیت D_3، D_5، D_6 و D_7 همراه با سه بیت توازن P_1، P_2 و P_4 فرموله کنید.

(الف) کلمه کد مرکب را برای کلمه داده 0010 بدست آورید.

(ب) سه بیت تست C_4، C_2 و C_1 را با فرض نبود خطا ارزیابی نمایید.

(پ) فرض کنید هنگام نوشتن در حافظه خطایی در بیت 5 رخ داده باشد. نشان دهید که چگونه خطا در این بیت شناسایی و اصلاح شده است.

(ت) بیت توازن P_8 را برای تشخیص دو بیت خطا در کد لحاظ کنید. فرض کنید که خطاها در بیت‌های P_2 و D_5 رخ دهد. نشان دهید که چگونه دو خطا تشخیص داده می‌شوند.

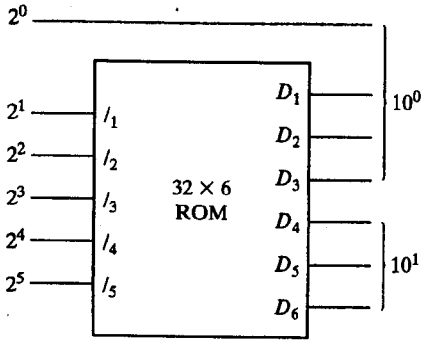
۷-۱۵ با فرض تراشه ROM 8 × 32 با یک ورودی فعال‌ساز، اتصالات بیرونی لازم برای ساخت یک ROM 8 × 128 را نشان دهید. از چهار تراشه و یک دیکدر استفاده کنید.

۷-۱۶ یک تراشه ROM با 8 × 4096 بیت دارای دو ورودی انتخاب تراشه بوده و با یک منبع تغذیه 5V کار می‌کند. برای بسته مدار مجتمع چند پایه لازم است؟ نمودار بلوکی را ترسیم کنید و همه ورودی‌ها و پایانه‌ها را در ROM علامت بزنید.

۷-۱۷ یک ROM 6 × 32 همراه با خط 2^0 مطابق شکل (م۷-۱۷)، یک عدد دودویی 6 بیت را به عدد BCD متناظرش تبدیل می‌کند. مثلاً 100001 (33 دهدهی به 0011 011 در BCD تبدیل می‌شود) جدول درستی ROM را مشخص نمایید.

۷-۱۸ سایز ROM (تعداد کلمات و تعداد بیت در هر کلمه) لازم که بتواند جدول درستی برای مدارهای ترکیبی زیر را در خود جای دهد چیست:

(الف) یک ضرب کننده که دو عدد 4 بیت را در هم ضرب کند.



شکل (م ۷-۱۷)

(ب) یک جمع- تفریق‌گر 4 بیت

(پ) یک مولتی پلکسر 2 به 1 چهارتایی با ورودی‌های فعال‌ساز و انتخاب مشترک

(ت) یک دیکدر BCD به هفت قسمتی با یک ورودی فعال‌ساز

۷-۱۹ جدول درستی یک ROM 4 × 8 را که توابع بولی زیر را پیاده‌سازی کند ایجاد کنید.

$$A(x, y, z) = \Sigma(1, 2, 4, 6)$$
$$B(x, y, z) = \Sigma(0, 1, 6, 7)$$
$$C(x, y, z) = \Sigma(2, 6)$$
$$D(x, y, z) = \Sigma(1, 2, 3, 5, 7)$$

اکنون با فرض ROM به عنوان یک حافظه، محتوای آدرس‌های 1 و 4 را معین نمایید.

۷-۲۰ جدول برنامه‌سازی PLA برای چهار تابع بولی لیست شده در مسئله ۷-۱۹ را ایجاد کنید. تعداد جملات ضرب را حداقل کنید.

۷-۲۱ جدول برنامه‌ریزی PLA را برای یک مدار ترکیبی که مربع 3 بیت را تولید کند مشخص کنید. تعداد جملات ضرب را حداقل نمایید. (شکل ۷-۱۲ برای پیاده‌سازی ROM معادل ملاحظه شود).

۷-۲۲ جدول برنامه‌ریزی PLA برای تبدیل BCD به افزونی 3 با توابع بولی ساده شده در شکل ۳-۴ را لیست کنید.

۷-۲۳ مسئله ۷-۲۲ را با PAL تکرار نمایید.

۷-۲۴ در زیر جدول درستی یک مدار ترکیبی 3 ورودی، 4 خروجی نشان داده شده است. جدول برنامه‌ریزی PAL را برای مدار نشان دهید و نقشه فیوز در نمودار PAL را مشابه شکل ۷-۱۷ علامت بزنید.

ورودی‌ها			خروجی‌ها			
x	y	z	A	B	C	D
0	0	0	0	1	0	0
0	0	1	1	1	1	1
0	1	0	1	0	1	1
0	1	1	0	1	0	1
1	0	0	1	0	1	0
1	0	1	0	0	0	1
1	1	0	1	1	1	0
1	1	1	0	1	1	1

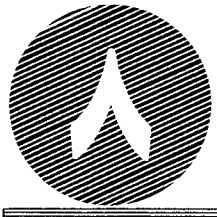
۷-۲۵ با به‌کارگیری ماکروسل ثباتی شکل ۱۹-۷، نقشه فیوز برای مدار ترتیبی حاوی دو ورودی x و y و یک فلیپ فلاپ A که با معادله D_A ورودی زیر تعریف شده را نشان دهید.

$$D_A = x \oplus y \oplus A$$

۷-۲۶ نمودار PAL شکل ۱۶-۷ را با لحاظ سه فلیپ فلاپ ساعت‌دار D بین گیت‌های OR و خروجی‌ها طبق ۱۹-۷ اصلاح کنید. نمودار باید با نمودار بلوکی مدار ترتیبی همخوانی داشته باشد. برای انجام این کار سه گیت بافر- وارونگر و سه خط عمودی برای خروجی‌های فلیپ فلاپ‌هایی که از طریق اتصالات برنامه‌پذیر به آرایه AND وصل می‌شوند لازم است. با به‌کارگیری نمودار PAL ثباتی اصلاح شده، نقشه فیوزی که سه شمارنده آرایه‌ای 3 بیت و یک نقلی خروجی را پیاده‌کند نشان دهید.

مراجع

1. TOCCI, R. J. and N. S. WIDMER. 2001. *Digital Systems Principles and Applications*, 8th ed. Upper Saddle River, NJ: Prentice Hall.
2. KITSON, B. 1984. *Programmable Array Logic Handbook*. Sunnyvale, CA: Advanced Micro Devices.
3. WAKERLY, J. F. 2000. *Digital Design: Principles and Practices*, 3rd ed. Upper Saddle River, NJ: Prentice Hall.
4. NELSON, V. P., H. T. NAGLE, J. D. IRWIN, and B. D. CARROLL. 1995. *Digital Logic Circuit Analysis and Design*. Upper Saddle River, NJ: Prentice Hall.
5. HAMMING, R. W. 1950. Error Detecting and Error Correcting Codes. *Bell Syst. Tech. J.* 29: 147-160.
6. LIN, S., and D. J. COSTELLO, J R. 1983. *Error Control Coding*. Englewood Cliffs, NJ: Prentice-Hall.
7. 1988. *Programmable Logic Data Book*. Dallas: Texas Instruments.
8. TRIMBERGER, S. M. 1994. *Field Programmable Gate Array Technology*. Boston: Kluwer Academic Pub.
9. 1994. *The Programmable Logic Data Book*, 2nd ed. San Jose, CA: Xilinx, Inc.
10. 1986. *Memory Components Handbook*. Santa Clara, CA: Intel.



سطح انتقال ثباتی

۸-۱ نمایش در سطح انتقال ثباتی

یک سیستم دیجیتال مداری است ترتیبی که از فلیپ فلاپ‌ها و گیت‌ها ساخته می‌شود. مدارهای ترتیبی را می‌توان با جداول حالت فصل ۵ نمایش داد. نمایش سیستم‌های دیجیتال با یک جدول درستی اگر غیرممکن نباشد، مشکل هست، زیرا تعداد حالات بسیار زیاد خواهد بود. برای غلبه بر این مشکل، سیستم‌های دیجیتال با روش مدولی طراحی می‌شوند. کل سیستم به چند زیرسیستم مدولی تقسیم می‌شود، که هر یک کار معینی را انجام می‌دهد. مدول‌ها از وسایل دیجیتالی همچون، ثبات‌ها، دیکدرها، مولتی پلکسرها، عناصر محاسباتی و مدارکنترل ساخته می‌شوند. این مدول‌ها با مسیرهای داده یا همان پردازشگر و کنترل مشترک به هم وصل می‌شوند تا یک سیستم دیجیتال به وجود آید.

مدول‌های دیجیتال در بهترین فرم براساس ثبات‌ها و عملیاتی که روی داده‌های ذخیره شده در آنها انجام می‌شود تعریف می‌گردند. مثال‌هایی از عملیات ثباتی عبارتند از شیفت یا جابجایی، شمارش، پاک کردن و بار شدن است. فرض بر این است که ثبات‌ها عنصر مبنای سیستم دیجیتال باشند. جریان اطلاعات و پردازش روی داده‌های ذخیره شده را **عملیات انتقال ثباتی** (RTL) گویند. یک سیستم دیجیتال هنگامی در سطح انتقال ثباتی نمایش داده شده است که با سه عنصر زیر مشخص شده باشد.

۱- مجموعه‌ای از ثبات‌ها در سیستم

۲- عملیات انجام شده روی داده‌های ذخیره شده در ثبات‌ها

۳- کنترلی که رشته عملیات را در سیستم نظارت کند

ثبات مجموعه‌ای از فلیپ فلاپ‌هاست که اطلاعات دودویی را ذخیره می‌کند و قادر به انجام یک یا چند عمل جزیی است. یک ثبات می‌تواند اطلاعات جدیدی را بارکند، یا آن را به راست و چپ جابجا کند. یک شمارنده ثباتی است که یک عدد را یکی یکی افزایش می‌دهد. یک فلیپ فلاپ به تنهایی یک ثبات

1 بیت است، که می‌تواند به 1 نشانده شده یا به 0 پاک شود و یا این که متمم گردد. در واقع، فلیپ فلاپ‌ها و گیت‌های مربوط به آنها در هر مدار ترتیبی را می‌توان بنا به تعریف ثبات خواند.

عملیات اجرائی روی اطلاعات ذخیره شده در ثبات‌ها اعمالی ساده است که به طور موازی روی رشته‌ای از بیت‌ها در یک سیکل ساعت انجام می‌شود. نتیجه حاصل از یک عمل ممکن است جایگزین اطلاعات قبلی در یک ثبات شود. به همین ترتیب نتیجه ممکن است به ثبات دیگری رفته و داده قبلی را در ثبات خود بدون تغییر رها کند. مدارهای دیجیتالی که در فصل ۶ معرفی شدند ثبات‌هایی هستند که عملیاتی ابتدایی را انجام می‌دهند. یک شمارنده با بار شدن موازی قادر است افزایش و بار شدن را اجرا نماید. یک شیفت رجیستر دو طرفه، می‌تواند جابجایی به راست و چپ را انجام دهد. عامل کنترل که رشته عملیات را آغاز می‌نماید، متشکل از سیگنال‌های زمانبندی است و عملیات را به ترتیب و براساس روشی از پیش تعین شده اجرا می‌کند. شرایط خاصی که به نتایج عملیات قبلی وابسته می‌باشد ممکن است رشته عملیات بعدی را تعیین کند. خروجی‌های مدار کنترل متغیرهای دودویی می‌باشند و انواع عملیات را در ثبات آغاز می‌کنند.

انتقال اطلاعات از یک ثبات به ثباتی دیگر به صورت نمادین با استفاده از عملگر جایگزینی مشخص می‌شود. عبارت

$$R2 \leftarrow R1$$

بیانگر یک انتقال از ثبات R_1 به ثبات R_2 است. این عبارت جایگزین شدن محتوای R_2 را با R_1 بیان می‌نماید. بنا به تعریف محتوای ثبات R_1 پس از انتقال تغییری نمی‌کند. فلش بیان‌کننده انتقال و جهت آن است.

عبارتی که انتقال ثباتی را بیانگر است بر این دلالت دارد که مدارهایی از خروجی‌های ثبات مبدأ به ورودی‌های ثبات مقصد وجود دارند و نیز ثبات مقصد دارای قابلیت بار شدن موازی است. معمولاً هدف این نیست که انتقال با هر پالس ساعت رخ دهد، بلکه می‌باید تحت شرایط از پیش تعیین شده‌ای صورت گیرد. یک عبارت شرطی براساس if–then مانند زیر بیان می‌شود.

$$\text{If } (T1 = 1) \text{ then } (R2 \leftarrow R1)$$

که در آن T_1 یک سیگنال کنترل تولید شده در بخش کنترل است. توجه کنید که ساعت به عنوان یک متغیر در نظر گرفته نشده است. فرض بر این است که همه انتقال‌ها در یک گذر لبه ساعت رخ دهند. گرچه یک شرط کنترل ممکن است قبل از لبه ساعت، به وقوع بپیوندد، ولی تا گذر ساعت تکمیل نگردد، انتقال واقعی رخ نخواهد داد.

برای جدایی دو یا چند عمل که به طور همزمان اجرا شوند از ویرگول استفاده می‌کنیم. عبارت زیر را در نظر بگیرید.

$$\text{If } (T3 = 1) \text{ then } (R2 \leftarrow R1, R1 \leftarrow R2)$$

این عبارت تعویض محتوای دو ثبات را در لبه ساعت با شرایط $T_3 = 1$ بیان می‌کند. این گونه اعمال همزمان با ثبات‌هایی امکان‌پذیر است که فلیپ فلاپ‌های حساس به لبه دارند. دیگر انتقال‌های ثباتی

نمونه در زیر معرفی شده‌اند.

$$R1 \leftarrow R1 + R2$$
$$R3 \leftarrow R3 + 1$$
$$R4 \leftarrow \text{shr } R4$$
$$R5 \leftarrow 0$$

جمع با جمع‌کننده موازی دودویی انجام می‌شود، افزایش با یک شمارنده و جابجایی به کمک شیفت رجیستر صورت می‌گیرد. انواع عملیاتی که معمولاً در سیستم‌های دیجیتال با آن مواجه می‌شویم به چهار گروه زیر دسته‌بندی می‌شوند:

۱- عملیات انتقال که داده را از یک ثبات به ثبات دیگر منتقل می‌نماید.
۲- عملیات حسابی که محاسبات را روی داده‌های ثبات‌ها اجرا می‌نمایند.
۳- عملیات منطقی که اعمال بیتی داده‌های غیر عددی را در ثبات‌ها انجام می‌دهند.
۴- عملیات شیفت یا جابجایی که داده را در ثبات جابجا می‌کنند.

عملیات انتقال محتوای اطلاعاتی داده جابجا شده از ثبات مبدأ به ثبات مقصد را تغییر نمی‌دهد. سه نوع دیگر محتوای اطلاعاتی را در حین جابجایی عوض می‌کنند. علائم و نشانه‌های انتقال ثبات به کار رفته در عملیات انتقال داده استاندارد نیستند. در اینجا دو نوع از نشانه‌ها معرفی شده تا سیستم‌های دیجیتال در سطح انتقال ثباتی توصیف گردند. بخش بعدی سمبل‌های RTL به کار رفته در Verilog HDL را معرفی می‌کنند.

۲-۸ سطح انتقال ثباتی (RTL)

سیستم‌های دیجیتال در سطح انتقال ثباتی با استفاده از زبان توصیف سخت‌افزاری توصیف می‌گردند. در Verilog HDL ، توصیف‌های سطح انتقال ثباتی (RTL) ترکیبی از ساختار رفتاری و جریان داده را به کار می‌برند. انتقال‌های ثباتی با عبارات تخصیص اجرایی مشخص می‌شدند. توابع مدارهای ترکیبی با تخصیص مداوم یا عبارات تخصیص اجرایی مشخص شدند. سمبل به کار رفته برای یک انتقال یک علامت مساوی یا فلش است. همزمانی با ساعت با استفاده از عبارت always همراه با کنترل واقعه Posedge یا negedge حاصل می‌گردد. مثال‌های زیر راه‌های ممکن موجود برای یک انتقال در Verilog HDL را نشان می‌دهد.

```
assign  S = A + B;          Continuous assignment          تخصیص مداوم

always @ (A or B)           Procedural assignment (without a clock)
    S = A + B;                                          تخصیص اجرایی

always @ (posedge clock)    Blocking procedural assignment
  begin                                                 تخصیص اجرایی بلوکی
   RA = RA + RB;
   RD = RA;
  end
```

```
always @ (negedge clock)   Non-blocking procedural assignment
  begin                                    تخصیص اجرایی غیربلوکی
   RA <= RA + RB;
   RD <= RA;
  end
```

تخصیص‌های مداوم یا پیوسته برای مدارهای ترکیبی به کار می‌روند. عبارت تخصیص قبلی یک جمع دودویی، با ورودی‌های A و B و خروجی S را نشان می‌دهد. عملوند هدف در یک عبارت تخصیص (در اینجا S) نمی‌تواند یک ثبات باشد. خروجی‌های مدارهای ترکیبی نمی‌توانند با عبارت اجرایی ساعت‌دار به یک ثبات انتقال یابند. تخصیص‌های اجرایی غیرساعت‌دار در مثال دوم راه دیگری را برای مشخص کردن یک مدار ترکیبی نشان می‌دهد.

دو نوع تخصیص اجرایی وجود دارد: بلوکی و غیربلوکی. این دو با سمبل‌های مربوطه‌شان از یکدیگر تفکیک می‌شوند. تخصیص‌های بلوکی از سمبل (=) به عنوان عملگر انتقال و تخصیص‌های غیربلوکی از (= >) به عنوان عملگر استفاده می‌نمایند. عبارات تخصیص بلوکی به طور متوالی طبق لیست در بلوک ترتیبی اجرا می‌شوند. تخصیص‌های غیربلوکی عبارات را از سمت راست ارزیابی می‌کنند ولی تا تمام عبارات ارزیابی نشوند تخصیصی به سمت چپ اختصاص نمی‌دهند. دو مثال فوق را در نظر بگیرید. در تخصیص اجرایی بلوکی، اولین عبارت حاصل جمع را به RA و دومین عبارت مقدار جدید RA را به RD منتقل می‌کند. در پایان، هر دو RA و RD مقدار یکسانی دارند. در تخصیص اجرایی غیربلوکی، دو عمل به طور همزمان انجام می‌گردد بنابراین RD مقدار اولیه RA را دریافت خواهد کرد.

برای اطمینان از عملیات همزمان در طراحی RTL ، لازم است تخصیص‌های اجرایی غیربلوکی برای متغیرهایی به کار روند که عبارت always–clocked را دنبال می‌کنند. دلیل این است که از احتمال تضاد بین مدل طراحی و توصیف HDL جلوگیری شود. تخصیص غیربلوکی ظاهر شده در عبارت always–clocked رفتار یک مدار ترتیبی همزمان را به دقت مدل‌سازی می‌کند.

عملگرهای HDL

عملگرهای HDL –Verilog و سمبل‌های به کار رفته آنها در طراحی RTL در جدول ۱-۸ لیست شده‌اند. عملگرهای حسابی، منطقی و جابجایی برای توصیف عملیات انتقال ثباتی لازمند. عملگرهای منطقی و ارتباطی برای مشخص کردن شرایط کنترل مورد نیازند. عملیات حسابی با اعداد دودویی انجام می‌شوند. اعداد منفی به صورت متمم 2 معرفی می‌گردند. عملگر **modulus** باقیمانده و حاصل از تقسیم دو عدد را تولید می‌نماید. مثلاً باقیمانده 3 ÷ 14 برابر 2 خواهد بود.

دو نوع عملگر منطقی وجود دارد: بیتی و کاهشی. عملگرهای بیتی عمل منطقی بیت به بیت را روی دو عملوند انجام می‌دهند. آنها یک بیت را از یک عملوند بدست آورده و عمل را با بیت متناظرش در عملوند دیگر، اجرا می‌کنند. عملگرهای کاهشی اعمال را روی یک عملوند تک انجام می‌دهند. آنها عمل را بیت به بیت از راست به چپ اجرا می‌کنند و نتیجه 1 بیتی را بدست می‌آورند. مثلاً NOR (~1) با

جدول ۱-۸. عملگرهای Verilog HDL

نوع عملگر	سمبل	عمل اجرا شده
حسابی	+	جمع
	-	تفریق
	*	ضرب
	/	تقسیم
	%	مدولوس
منطقی	~	نفی (متمم)
بیتی	&	AND
or	|	OR
کاهش	^	OR (XOR)
منطقی	!	نفی
	&&	AND
	| |	OR
جابجایی	>>	جابجایی به راست ست
	<<	جابجایی به چپ پ
	{ , }	ادغام
نسبی (ارتباطی)	>	بزرگتر از
	<	کوچکتر از
	==	تساوی
	!=	نامساوی
	>=	بزرگ تر از یا مساوی
	<=	کوچکتر از یا مساوی

عملوند 00101 نتیجه 0 و با عملوند 00000 نتیجه 1 را می‌دهد. از دستور **Negate** به عنوان یک عملگر کاهشی در اینجا استفاده نشده است. جدول درستی عملگرهای بیتی در جدول ۹-۴ بخش ۱۱-۴ ملاحظه شد.

عملیات منطقی و ارتباطی می‌توانند از متغیرها یا عبارات به عنوان عملوند استفاده کنند. این اعمال اصولاً برای تعیین حالات صحیح و غلط به کار می‌روند. این عملیات اگر شرایط صحیح باشد 1 و اگر غلط باشند، مقدار 0 را نتیجه می‌دهند. اگر شرط نامعین باشد x را خواهند داد. وقتی عملوند یک عدد است، اگر عدد برابر 0 باشد، 0 را تولید می‌نمایند و اگر غیرصفر باشد 1 را بدست می‌دهند. مثلاً اگر A = 1010 و B = 0000 باشد، آنگاه A به عنوان 1 (عدد غیرصفر است) و B به عنوان 0 تلقی می‌شوند. نتایج دیگر اعمال با این مقادیر عبارتند از:

```
A && B = 0
A || B = 1
!A = 0
!B = 1
(A > B) = 1
(A == B)= 0
```

عملگرهای جابجایی یک عملوند برداری را به دفعات مشخص شده‌ای به راست یا چپ جابجا می‌کنند. بیت‌های تخلیه شده با صفرها پر می‌گردند. مثلاً اگر R = 11010 باشد، عبارت

```
R = R >> 1;
```

R را یک بار به راست جابجا می‌کند. مقدار جدید R برابر 01101 است. عملگر ادغام مکانیزمی برای عملگرهای چندگانه ضمیمه فراهم می‌نماید. از این عملگرها می‌توان برای جابجایی و از جمله، بیت‌های انتقالی به مکان‌های خالی در شیفت رجیستر استفاده کرد. این نکته در مثال ۱-۶ HDL برای شیفت رجیستر نشان داده شده است.

عبارات حلقه‌زنی

Verilog HDL چهار نوع حلقه دارد که امکان اجرای مکرر عبارات اجرایی را میسر می‌سازد. این عملگرها عبارتند از: repeat ، forever ، while و for . همه عبارات حلقه‌زنی باید در بلوک initial یا always قرار گیرند.

حلقه repeat عبارات مربوطه را به دفعات مشخصی اجرا می‌کند. در زیر مثالی که قبلاً به کار رفته است ملاحظه می‌گردد:

```
initial
   begin
       clock = 1'b0;
       repeat (16)
       #5 clock = ~ clock;
   end
```

این بلوک هشت سیکل ساعت را با سیکل زمانی 10 واحد زمانی تولید می‌نماید.

حلقه forever اجرای مداوم یک عبارت اجرا را ایجاد می‌نماید. مثلاً حلقه زیر یک ساعت دائمی ایجاد می‌نماید:

```
initial
   begin
     clock = 1'b0;
     forever
     # clock = ~ clock;
   end
```

حلقه while اجرای یک عبارت یا بلوکی از عبارات را تکرار می‌نماید به شرطی که عبارتی صحت داشته باشد. در صورت غلط بودن شرط، عبارت هرگز اجرا نمی‌شود. در زیر استفاده از حلقه while نشان داده شده است.

```
integer count;
initial
   begin
     count = 0;
     while (count < 64)
     count = count + 1;
   end
```

مقدار شمارش از 0 تا 63 افزایش می‌یابد. حلقه در شماره 64 خارج می‌شود. هنگام برخورد با عبارات حلقه، بهتر است از داده نوع صحیح برای دستکاری کمیت‌ها استفاده شود. اعداد صحیح با کلمه کلیدی integer اعلان می‌شوند و در مثال قبل نیز مشاهده گردید. گرچه از کلمه کلیدی reg می‌توان برای متغیرها استفاده کرد، ولی بهتر است از متغیرهای صحیح در شمارش استفاده شود. متغیرهایی که از نوع داده reg هستند، به عنوان اعداد بی‌علامت ذخیره می‌شوند. آنهایی که از نوع integer هستند به صورت اعداد علامت‌دار و به فرم متمم 2 ذخیره می‌گردند. عرض پیش فرض یک عدد صحیح حداقل 32 بیت است.

حلقه for حاوی سه بخش است که با دو نقطه ویرگول از هم جدا شده‌اند.

- یک مقدار یا شرایط اولیه
- عبارتی برای تست پایان شرط
- عبارتی برای تغییر متغیر کنترل

در زیر مثالی برای حلقه for ارائه شده است.

```
for(i = 0; i < 8; i + 1)
procedural statements
```

عبارت حلقه، عبارات اجرایی را هشت بار تکرار می‌کند. متغیر کنترل i است و مقدار اولیه $i = 0$ می‌باشد، و حلقه مادامی که i کوچکتر از 8 باشد، تکرار خواهد شد. هر بار حلقه تکرار شود i یک واحد اضافه می‌شود.

توصیف یک دیکدر با 2 به 4 با استفاده از حلقه for در مثال ۱-۸ HDL نشان داده شده است. چون خروجی Y در یک عبارت اجرایی ارزیابی شده است، باید به صورت reg اعلان شود. متغیر کنترل برای حلقه یک عدد صحیح است. وقتی که حلقه گسترش یابد چهار حالت زیر را بدست خواهیم آورد (IN و Y دودویی و اندیس Y دهدهی است):

if $IN = 00$ then $Y(0) = 1$ else $Y(0) = 0$
if $IN = 01$ then $Y(1) = 1$ else $Y(1) = 0$
if $IN = 10$ then $Y(2) = 1$ else $Y(2) = 0$
if $IN = 11$ then $Y(3) = 1$ else $Y(3) = 0$

مثال ۱-۸، HDL

```
//description of 2x4 decoder
//using for loop statement
module decoder (IN, Y);
   input  [1:0] IN;     //Two binary inputs
   output [3:0] Y;     //Four binary outputs
   reg [3:0] Y;
   integer I;           //control variable for loop
   always @ (IN)
     for (I = 0; I <= 3; I = I + 1)
       if (IN == I) Y[I] = 1;
       else Y[I] = 0;
endmodule
```

سنتز منطقی

سنتز منطقی یک فرآیند اتوماتیک در انتقال یک توصیف زبان سطح بالا مانند HDL به یک netlist بهینه شده از گیت‌ها است که عملیات مشخص شده با کد منبع (source code) را اجرا می‌نمایند. تکنولوژی‌های موردنظر متعددی در پیاده‌سازی طرح وجود دارد. استفاده مؤثر از یک توصیف HDL لازم می‌دارد تا طراحان یک سبک خاص فروشنده را پیش گیرند تا به عنوان ابزار طراحی خاص مناسب باشد. نوع آی‌سی‌هایی که طرح را پیاده می‌کنند ممکن است مدار مجتمع از نوع کاربرد خاص (ASIC)، یک وسیله منطقی برنامه‌پذیر (PLD)، یا یک آرایه گیتی برنامه‌پذیر منطقی (FPGA) باشد.

ابزار طراحی منطقی برنامه‌هایی هستند که کد منبع، یک زبان توصیف سخت‌افزاری را تفسیر کرده و آن را به یک ساختار گیتی ترجمه یا بدل می‌کنند. طراحی‌های نوشته شده در HDL برای اهداف طراحی منطقی در سطح انتقال ثباتی می‌باشند. دلیل این است که ساختارهای HDL به کار رفته در توصیف RTL به راحتی قابل تبدیل به توصیف سطح گیتی می‌باشند.

عبارت assign برای توصیف مدارات ترکیبی به کار می‌رود. یک عبارت assign با توابع بولی به فرم مدار گیتی مربوطه تفسیر می‌گردد. یک عبارت با یک علامت بعلاوه (+) به صورت یک جمع دودویی با مدار جمع‌کننده کامل تفسیر می‌شود. عبارتی با علامت منها (−) به تفریق‌گری متشکل از جمع‌کننده کامل و گیت‌های XOR (شکل ۱۳-۴) تبدیل می‌گردد.

یک عبارت با عملگر شرطی مثل

```
assign Y = S ? I1 : I0;
```

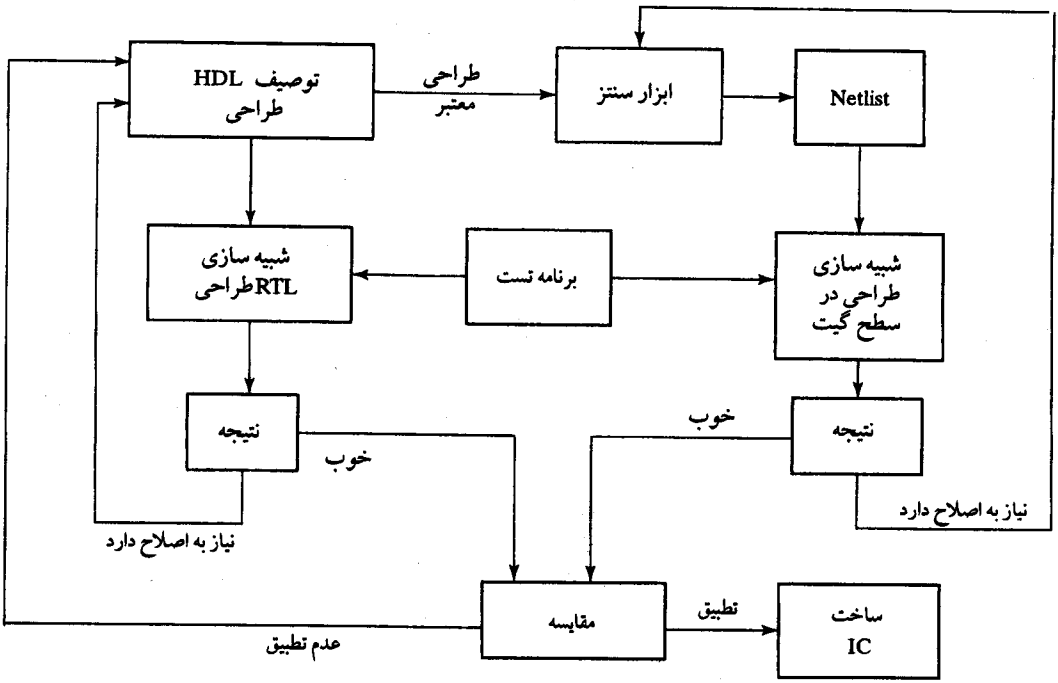
به یک مولتی پلکسر با ورودی کنترل S و ورودی‌های داده I_1 و I_0 برگردانده می‌شود. عبارت با عملگرهای شرطی چندگانه مولتی پلکسرهای بزرگتر را مشخص می‌نماید.

عبارت always ممکن است مداری ترکیبی یا ترتیبی را تداعی کند. برای مدارهای ترتیبی، کنترل واقعه باید لبه posedge یا negedge ساعت باشد، در غیر این صورت عبارت اجرایی یک مدار ترکیبی را مشخص خواهد کرد. مثلاً

```
always @ (I1 or I0 or S)
      if (S) Y = I1;
      else   Y = I0;
```

به یک مولتی پلکسر 2 به 1 ترجمه می‌گردد. عبارت case برای مولتی پلکسرهای بزرگتر به کار می‌رود. ساعت always @ posedge یا negedge مدار ترتیبی ساعت‌دار را مشخص می‌نماید. مدارهای مربوطه متشکل از فلیپ فلاپ‌های D و گیت‌هایی هستند که عملیات انتقال ثبات را پیاده می‌کنند. مثال‌هایی از اینگونه مدارها، ثبات‌ها و شمارنده‌ها می‌باشند. یک عبارت توصیف مدار ترتیبی، به یک مدار کنترل با فلیپ فلاپ D و گیت‌ها ترجمه می‌شود. بنابراین هر عبارت در یک توصیف RTL با سنتزگر تفسیر شده و به مدار گیتی و فلیپ فلاپ مربوطه تخصیص می‌یابد.

فلوچارت ساده شده فرآیند طراحی در شکل ۱-۸ نشان داده شده است. توصیف RTL طرح HDL شبیه‌سازی شده و برای اطمینان از صحت عملکرد چک می‌شود. برنامه تست سیگنال تحریک را برای



شکل ۱-۸. فرآیند شبیه سازی و سنتز HDL

شبیه‌سازی فراهم می‌نماید. اگر نتیجه شبیه‌سازی رضایت‌بخش نباشد، توصیف HDL اصلاح شده و دوباره چک می‌گردد. وقتی که نتایج شبیه‌سازی یک طرح معتبری را نشان دهد، توصیف RTL به سنتزگر (سیستم طراحی) منطقی اعمال می‌شود. ابزار سنتزگر، یک netlist معادل با توصیف سطح گیت طرح را تولید می‌کند. مدار سطح گیت، با همان محرک طرح RTL شبیه‌سازی می‌گردد. اگر تصحیحاتی لازم باشد، عمل شبیه‌سازی تا دستیابی به یک شبیه‌سازی مطلوب ادامه می‌یابد. نتایج دو شبیه‌سازی با هم مقایسه می‌گردند تا تطابق آنها مشاهده گردد. اگر به هم نزدیک نباشند طراح برای تغییر توصیف HDL بازگشته و هر خطایی را اصلاح می‌نماید. آنگاه دوباره توصیف به سنتزگر منطقی برده می‌شود تا یک توصیف سطح گیت جدید تولید گردد. به محض رضایت طراح از نتایج همه شبیه‌سازی‌ها، مدار قابل ساخت با یک مدار مجتمع است.

سنتزگر منطقی چندین مزیت را داراست. زمان کمتری در نوشتن یک توصیف HDL نیاز دارد و مدار سطح گیت آن مداری را بدست می‌دهد که از روی وارده‌های نمودار طرح آن قابل تهیه است. سادگی در تغییر توصیف امکان تغییر طراحی‌ها را عملی می‌سازد. تست اعتبار توصیف با شبیه‌سازی، ساده‌تر از ارزیابی نمونه سخت‌افزاری تولیدی است. بانک اطلاعاتی برای ساخت مدار مجتمع به طور خودکار با ابزار سنتزگر تولید می‌شود.

۸-۳ ماشین‌های حالت الگوریتمی (ASM)

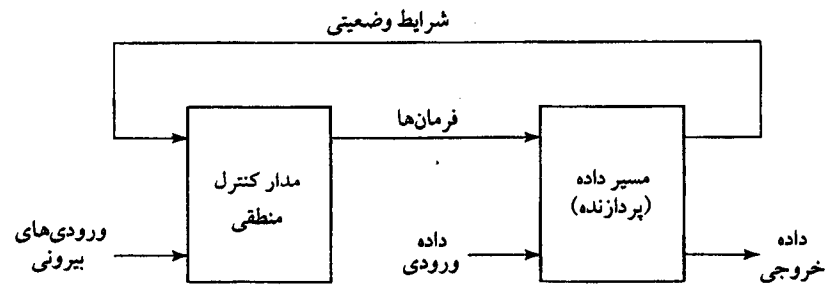
اطلاعات دودویی ذخیره شده در یک سیستم دیجیتال را می‌توان به دو صورت اطلاعات داده و کنترل دسته‌بندی کرد. داده عناصر گسسته‌ای از اطلاعات است که برای انجام اعمال حسابی، منطقی، جابجایی و دیگر عملیات پردازش داده دستکاری می‌شود. این عملیات با عناصر دیجیتالی چون جمع‌کننده‌ها، دیکدرها، مولتی پلکسرها، شمارنده‌ها و شیفت رجیسترها پیاده‌سازی می‌شوند. اطلاعات کنترلی، سیگنال‌های فرمانی که انواع عملیات را در بخش داده مدیریت می‌کنند تولید می‌نمایند تا کار پردازش داده موردنظر انجام شود. طراحی منطقی یک سیستم دیجیتال را می‌توان به دو بخش تقسیم کرد. یک بخش متعلق به مدارهای دیجیتال است که عملیات پردازش داده را انجام می‌دهند. بخش دیگر مربوط به طراحی مدارهای کنترلی است که عملیات مختلف و توالی آنها را مدیریت می‌نماید.

رابطه بین منطق کنترل و پردازش داده در یک سیستم دیجیتال در شکل ۲-۸ مشاهده می‌شود. مسیر پردازش داده، که به آن عموماً مسیر داده می‌گویند، برحسب نیازهای سیستم، داده را در ثبات دستکاری می‌کند. مدار کنترل یک رشته از فرامین را به مسیر داده صادر می‌نماید. مدار کنترل شرایط وضعیت را از مسیر داده دریافت کرده و برای تعیین رشته سیگنال‌های کنترل از آنها استفاده می‌کند.

مدار کنترلی که سیگنال‌هایی را برای توالی عملیات در مسیر داده تولید می‌کند یک مدار ترتیبی است که حالات داخلی‌اش فرامین کنترل را به سیستم دیکته می‌نماید. در هر زمان، حالت کنترل ترتیبی مجموعه‌ای از فرامین از پیش تعیین شده‌ای را آغاز می‌کند. بسته به شرایط وضعیت و دیگر ورودی‌های بیرونی، کنترل ترتیبی به حالت بعدی می‌رود تا اعمال دیگری را در مسیر داده آغاز کند و نیز حالت بعدی خود زیر سیستم کنترل را معین نماید.

وظایف رشته کنترل و مسیر داده سیستم دیجیتال با الگوریتم‌های سخت‌افزاری مشخص می‌گردد. یک الگوریتم متشکل است از تعداد معینی از مراحل اجرایی یا رویه‌ای، که چگونگی تهیه یک حل را برای یک مسئله با یک مقدار تجهیزات پیاده‌سازی می‌کند. ابتکاری‌ترین و مشکل‌ترین بخش از طراحی فرموله کردن الگوریتم‌های سخت‌افزاری برای رسیدن به اهداف است.

فلوچارت روشی مناسب برای مشخص کردن رشته مراحل اجرایی و مسیرهای تصمیم‌گیری در یک الگوریتم است. فلوچارت برای الگوریتم سخت‌افزاری جملات را به یک نمودار اطلاعاتی تبدیل



شکل ۲-۸. واکنش‌بین کنترل و مسیر داده

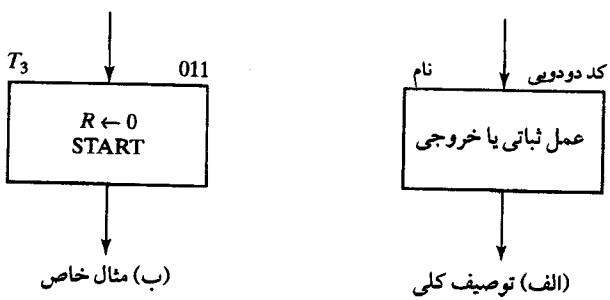
می‌کند که رشته عملیات آنها را به همراه شرایط لازم‌الاجرا، یک به یک می‌شمارد. یک فلوچارت خاص که فقط برای تعریف الگوریتم‌های سخت‌افزاری به وجود آید چارت ماشین حالت الگوریتمی (ASM) نام دارد. ماشین حالت نیز نام دیگری برای یک مدار ترتیبی است که ساختار مبنا برای یک سیستم دیجیتال می‌باشد.

چارت ASM ظاهراً همان چارت معمولی است، ولی تفسیر متفاوتی دارد. فلوچارت معمولی رشته مراحل اجرایی و مسیرهای تصمیم‌گیری را برای یک الگوریتم به ترتیب و بدون ملاحظه ارتباط زمانی آنها، توصیف می‌کند. چارت ASM رشته وقایع به همراه ارتباط زمانی بین حالات یک کنترل‌گر و وقایعی که ضمن رفتن از یک حالت به حالت بعدی رخ می‌دهند را توصیف می‌نماید. این چارت خصوصاً برای مشخص کردن دقیق رشته کنترل و عملیات مسیر داده (واحد پردازش) در یک سیستم دیجیتال انتخاب شده است، ضمن این که محدودیت‌های سخت‌افزار دیجیتال نیز لحاظ شده است.

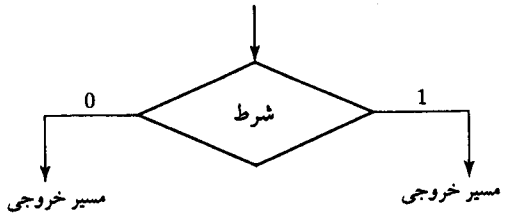
چارت ماشین حالت الگوریتمی

چارت ماشین حالت الگوریتمی (ASM) نوع خاصی از چارت است که برای توصیف عملیات ترتیبی یک سیستم دیجیتال مناسب می‌باشد. چارت از سه جزء مبنا ساخته شده است: جعبه حالت، جعبه تصمیم و جعبه شرط. یک حالت در رشته کنترل با جعبه حالت نشان داده می‌شود، شکل ۳-۸. جعبه حالت مستطیل شکل است و در داخل آن عملیات ثباتی یا نام سیگنال خروجی که کنترل تولید می‌کند نیز در این جعبه قرار دارند. به حالت نام سمبلیکی داده شده که در سمت چپ بالای جعبه نوشته می‌شود، کد تخصیص یافته به حالت در سمت راست بالا قرار می‌گیرد. شکل ۳-۸ (ب) مثال خاصی از یک جعبه حالت را نشان می‌دهد. حالت، نام سمبلیک T_3 را داشته و کد دودویی 011 به آن اختصاص یافته است. در داخل جعبه، عمل ثباتی $R \leftarrow 0$ نوشته شده و به معنی پاک شدن ثبات R به 0، در حالت T_3 است. نام START در داخل جعبه، می‌تواند مثلاً، یک سیگنال خروجی باشد که عمل خاصی را آغاز می‌نماید.

جعبه تصمیم تأثیر یک ورودی را روی زیر سیستم کنترل توصیف می‌کند. جعبه مربوط به آن طبق شکل ۴-۸ به فرم لوزی است که دو یا چند مسیر خروجی را داراست. شرط تست ورودی در داخل جعبه نوشته می‌شود. یکی از خروجی‌ها متعلق به صحت شرط و دیگری متعلق به عدم صحت آن است.



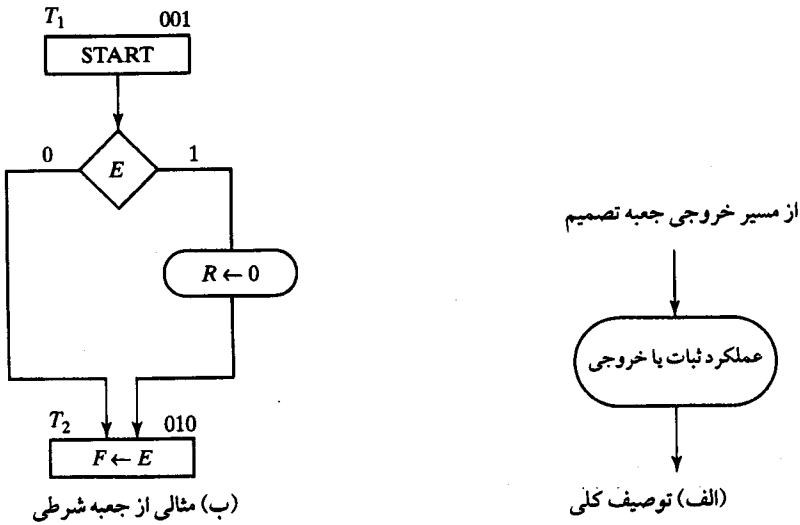
شکل ۳-۸. جعبه حالت



شکل ۴-۸. جعبه تصمیم

اگر به یک ورودی شرط مقدار دودویی اختصاص دهیم، دو مسیر با 1 و 0 مشخص می‌شوند.

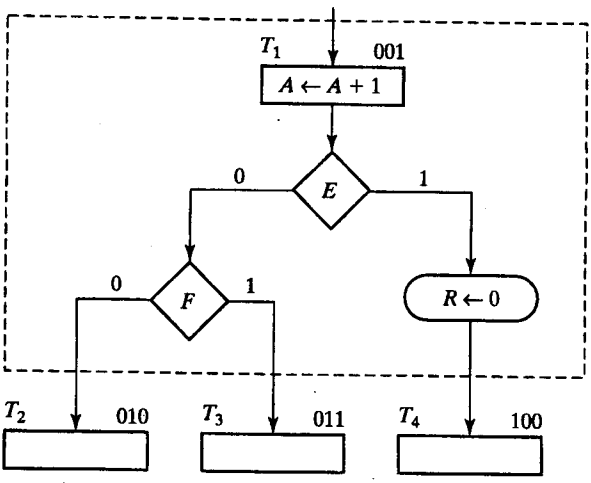
جعبه‌های حالت و تصمیم با توجه به فلوچارت‌های معمولی، آشنا به نظر می‌رسند. سومین عنصر یعنی جعبه شرطی تنها منحصر به چارت ASM است. شکل بیضی مانند جعبه شرطی در شکل ۵-۸ مشاهده می‌شود. گوشه‌های قوسی شکل، آن را از جعبه حالت متمایز می‌سازد. مسیر ورودی به جعبه شرطی باید از یک مسیر خروجی در جعبه تصمیم بیاید. عملیات ثباتی و خروجی‌های لیست شده در جعبه شرطی در طول یک حالت مفروض به شرط برآورده شدن شرط، صورت می‌پذیرد. شکل ۵-۸ (ب) مثالی را با یک جعبه شرطی نشان می‌دهد. کنترل یک سیگنال خروجی START را در حالت T_1 تولید می‌کند. کنترل حالت ورودی E را چک می‌کند. اگر $E = 1$ باشد، آنگاه $R = 0$ خواهد شد، در غیر این صورت R بدون تغییر می‌ماند. در هر دو صورت حالت بعدی T_2 خواهد بود.



شکل ۵-۸. جعبه شرطی

بلوک ASM

یک بلوک ASM ساختاری است متشکل از یک جعبه حالت و تمام جعبه‌های تصمیم و شرطی که به خروجی آن وصل شده‌اند. یک بلوک ASM دارای یک ورودی و تعداد نامشخصی از مسیرهای

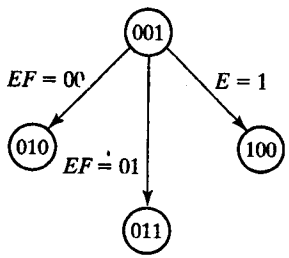


شکل ۶-۸. بلوک ASM

خروجی است که به وسیله ساختار جعبه‌های تصمیم نشان داده می‌شود. مثالی از یک بلوک ASM در شکل ۶-۸ ملاحظه می‌گردد. در ارتباط با حالت T_1 دو جعبه تصمیم و یک جعبه شرطی وجود دارد. خطوط نقطه‌چین حول تمام مجموعه، آن را از بقیه چارت جدا می‌سازد ولی معمولاً این عمل صورت نمی‌گیرد زیرا حالت ASM هر بلوک را منحصراً با توجه به ساختار آن توصیف می‌نماید. یک جعبه حالت بدون جعبه تصمیم و یا شرطی، یک بلوک ساده می‌باشد.

هر بلوک در چارت ASM یک حالت از سیستم را در یک پالس ساعت توصیف می‌نماید. عملیات درون جعبه‌های حالت و شرط در شکل ۶-۸ در یک پالس ساعت مشترک و در حالی که سیستم در T_1 است اجرا می‌گردند. همان پالس ساعت نیز کنترل‌گر سیستم را با توجه به مقادیر دودویی E و F به حالت‌های T_2 ، T_3 و یا T_4 خواهد برد.

چارت ASM خیلی شبیه به نمودار حالت است. هر بلوک حالت معادل با یک حالت در مدار ترتیبی است. جعبه تصمیم معادل با اطلاعات دودویی نوشته شده در امتداد فلشی است که دو حالت را در نمودار حالت بهم وصل می‌کند. در نتیجه گاهی بهتر است چارت را به نمودار حالت تبدیل نموده و سپس روش مدار ترتیبی را برای طراحی مدار کنترل دنبال نماییم. به منظور تشریح، چارت شکل ۶-۸ به صورت نمودار شکل ۷-۸ کشیده شده است. حالات با دوایری همراه با مقادیر دودویی‌شان که در



شکل ۷-۸ . نمودار حالت معادل با چارت ASM در شکل ۶-۸

داخل هر یک از دوایر نوشته شده، نشان داده شده‌اند. خطوط جهت‌دار شرایطی را نشان می‌دهند که حالت بعدی را معین می‌کنند. عملیات شرطی و غیرشرطی که باید اجرا شوند در نمودار حالت آورده نشده‌اند.

بررسی‌های زمانبندی

زمانبندی همه ثبات‌ها و فلیپ فلاپ‌ها به وسیله یک مدار ساعت اصلی کنترل می‌گردد. پالس‌های ساعت نه فقط به بخش مسیر داده بلکه به تمام فلیپ فلاپ‌های مدار کنترل نیز اعمال می‌گردند. ورودی‌ها نیز با پالس‌های ساعت همزمانند زیرا این ورودی‌ها، خروجی‌های مدار دیگری هستند که از همان پالس‌های ساعت استفاده می‌کنند. اگر سیگنال ورودی در یک زمان دلخواه و مستقل از پالس ساعت تغییر یابد آن را ورودی غیرهمزمان یا آسنکرون می‌نامیم. ورودی‌های غیرهمزمان ممکن است موجب بروز انواع مشکلات، طبق آنچه در فصل ۹ خواهیم دید، گردند. برای ساده‌کردن طراحی، فرض خواهیم‌کرد که تمام ورودی‌ها نسبت به پالس ساعت همزمان بوده و با لبه پالس ساعت تغییر حالت می‌دهند.

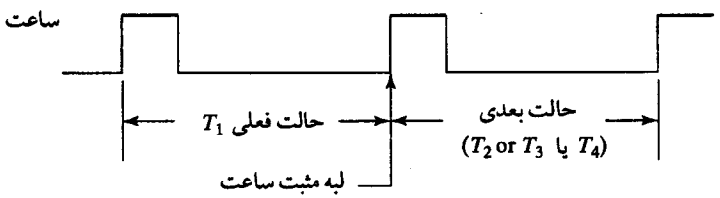
تفاوت عمده بین یک فلوچارت و یک ASM چارت در تفسیر زمانی روابط بین عملیات مختلف است. مثلاً اگر شکل ۶-۸ یک چارت معمولی باشد، عملیات لیست شده در آن یکی پس از دیگری با ترتیب زمانی دنبال می‌شود. با این فرض ابتدا ثبات A افزایش یافته و آنگاه E اضافه می‌گردد. اگر $E = 1$ باشد آنگاه ثبات R پاک شده و کنترل به حالت T_4 می‌رود در غیر این صورت با $E = 0$ در قدم بعدی F ارزیابی شده و به حالت T_2 یا T_3 خواهد رفت. برعکس یک چارت ASM کل بلوک را یک جا و به طور واحد در نظر می‌گیرد. تمام اعمالی که در داخل بلوک باشند به طور همزمان و در لحظه گذر لبه پالس ساعت و در لحظه انتقال سیستم از حالت T_1 به حالت بعدی اتفاق خواهند افتاد. این نکته به طور مصور در شکل ۸-۸ نشان داده شده است. فرض بر این است که لبه مثبت ساعت همه فلیپ فلاپ‌ها را تریگر کند. اولین انتقال مثبت ساعت مدار کنترل را به حالت T_1 می‌برد. ضمن قرارگرفتن سیستم در حالت T_1 مدار کنترل ورودی‌های F و E را چک کرده و سیگنال‌های مناسبی را بر طبق آنها تولید می‌نماید. عملیات زیر در لحظه گذر لبه پالس مثبت بعدی به طور همزمان رخ می‌دهد:

۱- ثبات A افزایش می‌یابد

۲- اگر $E = 1$ باشد ثبات R پاک می‌شود

۳- بسته به مقدار E و F، کنترل به حالت بعدی T_2 یا T_3 یا T_4 منتقل می‌گردد

توجه‌کنید که دو عمل در بخش مسیر داده و یک تغییر حالت در مدار کنترل به طور همزمان اتفاق می‌افتند.



شکل ۸-۸. انتقال بین حالات

۸-۴ مثال طراحی

اکنون اجزاء یک چارت ASM و نمایش انتقال ثبات را با مروری بر یک مثال طراحی نشان خواهیم داد. مثال را با مشخصات اولیه آغاز کرده و کار را با تشکیل یک ASM مناسب که از آن طراحی سخت‌افزار دیجیتال حاصل می‌شود، دنبال می‌کنیم.

می‌خواهیم سیستم دیجیتالی با دو فلیپ فلاپ E و F و یک شمارنده دودویی 4 بیت A طراحی کنیم. فلیپ فلاپ‌های A با A_4، A_3، A_2 و A_1 نام‌گذاری شده‌اند که در آن A_4 باارزش‌ترین بیت شمارش را نگه می‌دارد. یک سیگنال شروع S، عمل سیستم را با پاک کردن شمارنده A و فلیپ فلاپ F آغاز می‌کند. آنگاه شمارش با پالس ساعت بعدی شروع و تا پایان عملیات این افزایش ادامه دارد. بیت‌های A_3 و A_4 توالی عمل را معین می‌کنند.

- اگر $A_3 = 0$ باشد، $E = 0$ شده و شمارش ادامه می‌یابد.
- اگر $A_3 = 1$ باشد $E = 1$ شده؛ سپس اگر $A_4 = 0$ باشد شمارش ادامه می‌یابد، ولی اگر مقدار $A_4 = 1$ باشد در لبه پالس ساعت بعدی $F = 1$ خواهد بود.

سپس اگر S=0باشد، سیستم در حالت شروع باقی می‌ماند، ولی اگر S=1شود سیکل عمل تکرار می‌گردد.

چارت ASM

چارت ASM در شکل ۸-۹ دیده می‌شود. وقتی عملی اجرا نشود، سیستم در حالت اولیه T_0 بوده و منتظر سیگنال شروع S می‌باشد. وقتی ورودی S برابر 1 باشد، شمارنده A و فلیپ فلاپ F به 0 پاک می‌شود و کنترل‌گر به حالت T_1 می‌رود. توجه کنید که جعبه شرطی به دنبال جعبه تصمیم S آمده است. این بدان معنی است که اگر $S = 1$ باشد شمارنده و فلیپ فلاپ در T_0 پاک خواهند شد و در همان زمان کنترل به حالت T_1 خواهد رفت.

بلوک مربوط به حالت T_1 دارای دو جعبه تصمیم‌گیری و دو جعبه شرطی است. شمارنده با هر پالس ساعت افزایش می‌یابد. در همان زمان سه عمل دیگر رخ می‌دهد:

$E = 0$ شده و کنترل در T_1 می‌ماند $(A_3 = 0)$؛ یا

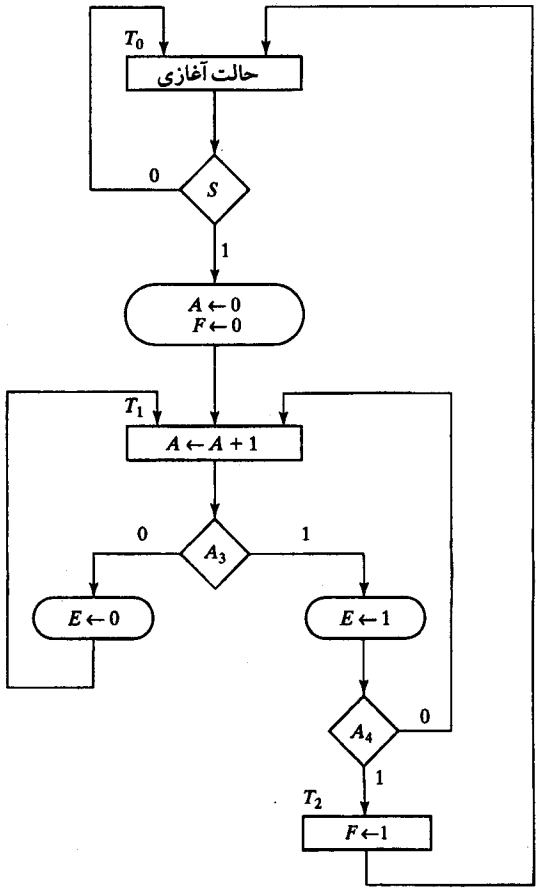
$E = 1$ شده و کنترل در T_1 باقی خواهد ماند $(A_3A_4 = 10)$؛ یا

$E = 1$ شده و کنترل به حالت T_2 می‌رود $(A_3A_4 = 11)$.

وقتی کنترل در حالت T_2 است، فلیپ فلاپ $F = 1$ شده و مدار به حالت اولیه T_0 باز می‌گردد.

چارت ASM از سه حالت و سه بلوک تشکیل شده است. بلوک متعلق به T_0 متشکل از جعبه حالت، یک جعبه تصمیم و یک جعبه شرطی است. بلوک T_2 فقط جعبه حالت دارد. کنترل منطقی یک ورودی بیرونی، S، و دو ورودی حالت A_3 و A_4 دارد.

در اینجا راه ترجمه یا برگرداندن یک توصیف کلمه‌ای را به چارت ASM نشان خواهیم داد. باید توجه داشت که مثال طراحی در چارت ASM کاربرد عملی ندارد و بسته به تفسیر ممکن است ساده و یا فرموله شود. با این وجود، پس از ایجاد چارت ASM، روش طراحی مدار بسیار ساده خواهد بود.



شکل ۹-۸. چارت ASM برای مثال طراحی

ترتیب زمانبندی

هر بلوک در چارت ASM بیانگر عملیاتی است که قرار است در طول یک پالس ساعت صورت گیرد. عملیات مشخص شده در داخل یک جعبه حالت و جعبه‌های شرطی در بخش مسیر داده انجام می‌شود. تغییر از یک حالت به حالت بعدی در مدار کنترل اتفاق می‌افتد. برای درک روابط زمانی موجود، عملیات را از ابتدای پالس شروع (استارت) تا بازگشت به حالت اولیه قدم به قدم پس از هر پالس ساعت لیست خواهیم کرد.

جدول ۲-۸ مقادیر دودویی شمارنده و دو فلیپ فلاپ را پس از هر پالس ساعت نشان می‌دهد. جدول به طور جداگانه وضعیت A_3 و A_4 را به همراه حالت فعلی کنترل‌گر نیز داراست. ما با حالت T_1 و درست پس از این که سیگنال ورودی S شمارنده و فلیپ فلاپ F را پاک می‌کند آغاز می‌کنیم. فرض بر

جدول ۲-۸. رشته‌ای از عملیات برای مثال طراحی

شمارنده				فلیپ فلاپ‌ها			
A_4	A_3	A_2	A_1	E	F	شرایط	حالت
0	0	0	0	1	0	$A_3 = 0, A_4 = 0$	T_1
0	0	0	1	0	0		
0	0	1	0	0	0		
0	0	1	1	0	0		
0	1	0	0	0	0	$A_3 = 1, A_4 = 0$	
0	1	0	1	1	0		
0	1	1	0	1	0		
0	1	1	1	1	0		
1	0	0	0	1	0	$A_3 = 0, A_4 = 1$	
1	0	0	1	0	0		
1	0	1	0	0	0		
1	0	1	1	0	0		
1	1	0	0	0	0	$A_3 = 1, A_4 = 1$	
1	1	0	1	1	0		T_2
1	1	0	1	1	1		T_0

این است که مقدار E برابر 1 باشد زیرا E در T_0 برابر 1 است (در انتهای جدول نشان داده شده است) و مقدار آن در گذر T_0 به T_1 عوض نمی‌شود. سیستم در طول 13 پالس ساعت بعدی در T_1 باقی می‌ماند. هر پالس شمارنده را افزایش داده و E را 0 یا 1 می‌کند. به ارتباط لحظه‌ای که $A_3 = 1$ و لحظه‌ای که $E = 1$ می‌شود توجه نمایید. وقتی $A = 0011$ است، پالس ساعت بعدی شمارنده را به 0100 افزایش می‌دهد، و در همان لحظه $A_3 = 0$ بوده و بنابراین $E = 0$ می‌گردد. پالس بعدی شمارنده را از 0100 به 0101 می‌برد، که در آن $A_3 = 1$ بوده و بنابراین $E = 1$ خواهد شد. به طور مشابه $E = 0$ می‌شود ولی نه هنگامی که شمارش از 0111 به 1000 می‌رود بلکه موقعی که از 1000 به 1001 برود و در این حالت فعلی شمارنده $A_3 = 0$ است.

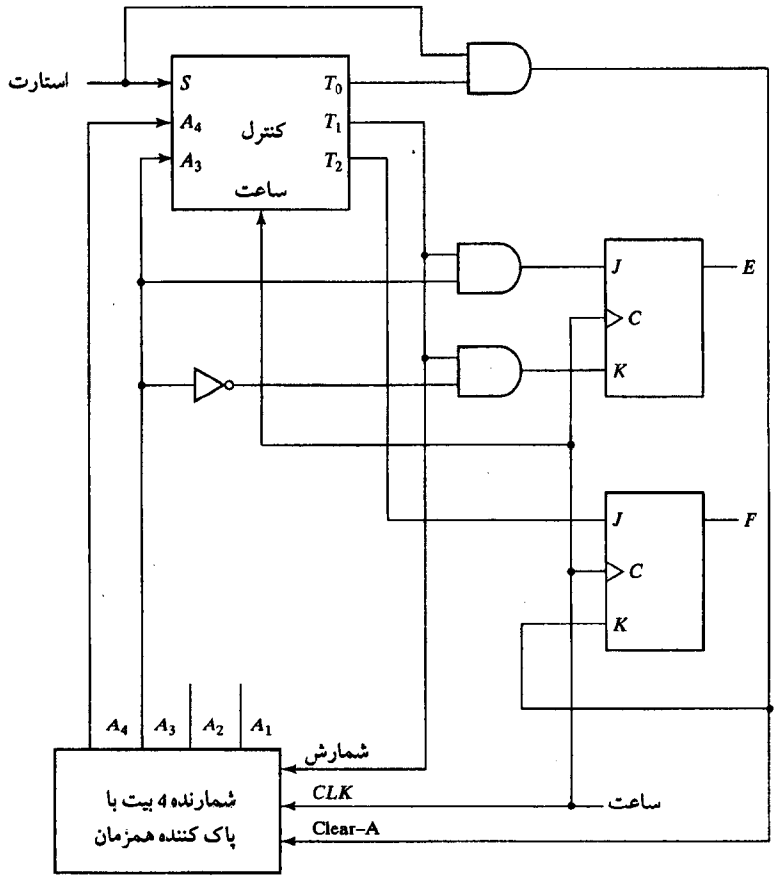
وقتی شمارش به 1100 برسد هر دو مقدار A_3 و A_4 برابر 1 خواهند بود. پالس ساعت بعدی A را یک واحد افزایش داده، E را برابر 1 نموده و کنترل را به حالت T_2 منتقل می‌کند. کنترل برای مدت یک پریود ساعت در T_2 می‌ماند. لبه ساعت متعلق به T_2 فلیپ فلاپ F را 1 می‌کند و کنترل به T_0 باز می‌گردد. سیستم مادامی که $S = 0$ باشد در T_0 باقی خواهد ماند.

با ملاحظه جدول ۲-۸، مشاهده می‌شود که عملیات انجام شده روی E به اندازه یک پالس ساعت به تعویق می‌افتد. این اختلاف بین یک چارت ASM و فلوچارت عادی است. اگر شکل ۹-۸ یک چارت معمولی می‌بود، می‌توانستیم فرض کنیم که ابتدا A افزایش یافته و مقدار افزایش یافته برای چک کردن وضعیت A_3 به کار برده می‌شود. عملیاتی که در سخت‌افزار دیجیتال رخ می‌دهند، همانطور که در بلوک ASM مشخص شده است در یک پالس ساعت بوده و مشابه با آنچه در فلوچارت معمولی بود به

صورت متوالی اتفاق نمی‌افتند. بنابراین مقدار مورد بررسی A_3 در جعبه تصمیم از وضعیت فعلی شمارنده و قبل از افزایش اختیار می‌شود. به این علت است که جعبه تصمیم متعلق به همان بلوکی است که T_1 در آن قرار دارد. مدارات دیجیتال در کنترل، سیگنال‌هایی را برای تمام عملیات مشخص شده در بلوک فعلی و قبل از ورود به بلوک بعدی تولید می‌کنند. لبه ساعت بعدی همه عملیات را در ثبات‌ها، فلیپ فلاپ‌ها، از جمله فلیپ فلاپ کنترل‌گر اجرا می‌کند و حالت بعدی را معین خواهد کرد.

طراحی مسیر داده

چارت ASM همه اطلاعات لازم را برای طراحی سیستم دیجیتال فراهم می‌سازد. نیازهای طراحی مسیر داده در داخل جعبه‌های حالت شرطی مشخص می‌گردد. مدار کنترل از جعبه‌های تصمیم و گذرهای حالت لازم بدست می‌آید. نموداری که سخت‌افزار مثال را نشان می‌دهد در شکل ۱۰-۸ ملاحظه می‌شود. زیر سیستم کنترل با ورودی‌ها و خروجی‌هایش نشان داده شده است. طراحی مشروح کنترل بعداً بررسی شده است. مسیر داده یا همان پردازشگر از شمارنده‌ای چهار بیتی، دو فلیپ فلاپ، و



شکل ۱۰-۸ . پردازشگر داده برای مثال طراحی

تعدادی گیت تشکیل شده است. شمارنده مشابه با آنچه در شکل ۱۲-۶ دیدیم می‌باشد، به جز این که برای پاک کردن همزمان نیاز به گیت‌های اضافی است. شمارنده هر بار که کنترل در T_1 است با هر پالس ساعت افزایش می‌یابد. واضح است که وقتی کنترل در حالت T_0 و کلید S در 1 باشد شمارنده پاک می‌شود. این عملکرد شرطی به یک گیت AND نیاز دارد تا هر دو شرط را تضمین کند. دو عملکرد شرطی دیگر از دو گیت AND دیگر برای نشاندن یا پاک کردن فلیپ فلاپ E استفاده می‌کند. فلیپ فلاپ F بدون شرط در T_2 به 1 نشانده می‌شود. توجه کنید که همه فلیپ فلاپ‌ها و ثبات‌ها از جمله فلیپ فلاپ‌های کنترل از ساعت مشترکی استفاده می‌کنند.

نمایش انتقالی ثباتی

یک سیستم دیجیتال، با مشخص کردن ثبات‌ها در سیستم، عملیات اجرایی و رشته کنترل، در سطح انتقال ثباتی (RTL) نمایش داده می‌شود. عملیات ثباتی و اطلاعات کنترلی را می‌توان با چارت ASM مشخص کرد. گاهی بهتر است تا مدار کنترل و عملیات ثباتی را از مسیر داده جدا کنیم. اطلاعات کنترل و عملیات انتقال ثباتی را می‌توان طبق شکل ۱۱-۸ به طور جداگانه نشان داد. نمودار حالت رشته حالت و عملیات ثباتی با علائم معرفی شده در بخش ۱-۸ نمایش داده می‌شوند. این نمایش راه دیگری نسبت به روش چارت ASM در شکل ۹-۸ می‌باشد. اطلاعات برای نمودار حالت مستقیماً از چارت ASM بدست می‌آید. نام حالت در هر جعبه حالت ذکر شده است. شرایطی که تغییر حالتی را سبب می‌شوند در داخل جعبه‌های تصمیم مشخص شده‌اند. خطوط جهت‌دار بین حالات و شرایط مربوط به آنها مسیر مشابهی را با چارت ASM دنبال می‌کنند. عملیات انتقال ثباتی برای هر یک از سه حالت به دنبال نام حالت و یک جفت نقطه (:) آمده است. زیرا این عملیات از جعبه‌های حالت مستطیلی و جعبه‌های حالت بیضی شکل در چارت ASM حاصل شده است.

جدول حالت

می‌توان نمودار حالت را به جدول حالت تبدیل کرد که از آن مدار ترتیبی کنترل‌گر قابل طراحی است. ابتدا باید مقادیری دودویی را به هر حالت در چارت ASM تخصیص دهیم. برای n فلیپ فلاپ در مدار

T_0: if $(S = 1)$ then $A \leftarrow 0, F \leftarrow 0$

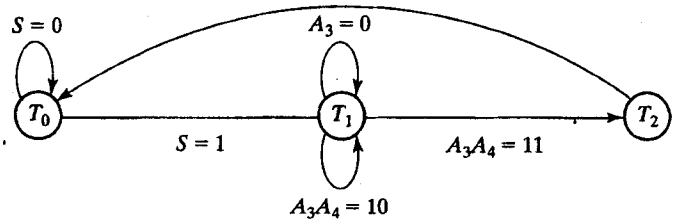
T_1: $A \leftarrow A + 1$

if $(A_3 = 1)$ then $E \leftarrow 1$

if $(A_3 = 0)$ then $E \leftarrow 0$

T_2: $F \leftarrow 1$

(ب) عملیات انتقال ثبات



(الف) نمودار حالت کنترل

شکل ۱۱-۸. توصیف سطح انتقال ثباتی

کنترل ترتیبی چارت ASM می‌تواند تا 2^n حالت را دارا باشد. چارتی با سه یا چهار حالت به یک مدار ترتیبی به دو فلیپ فلاپ نیاز دارد. با چهار تا هشت حالت، سه فلیپ فلاپ احتیاج است. هر ترکیبی از مقادیر فلیپ فلاپ‌ها یک عدد دودویی را برای یکی از حالات نشان می‌دهد.

یک جدول حالت برای یک کنترل‌گر، لیستی از حالات فعلی و مقادیر متناظرشان در حالت بعدی و خروجی‌هاست. در بسیاری از موارد، شرایط ورودی بی‌اهمیتی وجود دارند که باید لحاظ شوند، بنابراین پیشنهاد می‌شود تا جدول حالت تشکیل و این حالات در نظر گرفته شوند. ما مقادیر دودویی زیر را به سه حالت تخصیص می‌دهیم: $T_0 = 00$ ، $T_1 = 01$ ، $T_2 = 11$. حالت دودویی 10 به کار نرفته است و به عنوان حالت بی‌اهمیت تلقی می‌گردد. جدول حالت مربوط به نمودار حالت در جدول۳-۸ نشان داده شده است. دو فلیپ فلاپ مورد نیازند و با G_1 و G_0 نشانگذاری شده‌اند. سه ورودی و سه خروجی نیز وجود دارند. ورودی‌ها از شرایط درون جعبه‌های تصمیم اخذ می‌شوند. خروجی‌ها معادل با حالت فعلی کنترل‌اند. توجه کنید که در جدول سطری برای هر گذر ممکن بین حالات در نظر گرفته شده است. حالت اولیه 00 بسته به ورودی S به حالت 01 می‌رود یا در 00 می‌ماند. دو ورودی دیگر با علامت X مشخص شده‌اند و در تعیین حالت بعدی نقشی ندارند. ضمن قرار گرفتن سیستم در حالت دودویی 00، کنترل خروجی T_0 را تولید می‌کند تا عملیات ثباتی موردنظر آغاز گردد. گذر از حالت دودویی 01 به ورودی‌های A_3 و A_4 وابسته است. سیستم هنگامی به حالت دودویی 11 خواهد رفت که $A_3A_4 = 11$ باشد؛ در غیر این صورت در حالت دودویی 01 باقی خواهد ماند. بالاخره، حالت دودویی 11 مستقل از متغیرهای ورودی، در پالس ساعت بعدی به 00 خواهد رفت.

مدار کنترل

روال طراحی یک مدار ترتیبی در فصل ۵ با شروع از جدول حالت ملاحظه شد. اگر این روش به جدول ۳-۸ اعمال شود، لازم است از یک جدول پنج متغیره برای ساده کردن معادلات ورودی استفاده کنیم. دلیل این است که در زیر ستون حالت فعلی و ورودی، پنج متغیر وجود دارد. در عوض استفاده از نقشه برای ساده کردن معادلات ورودی، می‌توانیم آنها را با بررسی جدول حالت بدست آوریم. برای

جدول ۳-۸. جدول حالت برای کنترل شکل ۱۰-۸

سمبل حالت فعلی	حالت فعلی		ورودی‌ها			حالت بعدی		خروجی‌ها		
	G_1	G_0	S	A_3	A_4	G_1	G_0	T_0	T_1	T_2
T_0	0	0	0	X	X	0	0	1	0	0
T_0	0	0	1	X	X	0	1	1	0	0
T_1	0	1	X	0	X	0	1	0	1	0
T_1	0	1	X	1	0	0	1	0	1	0
T_1	0	1	X	1	1	1	1	0	1	0
T_2	1	1	X	X	X	0	0	0	0	1

طراحی مدار ترتیبی با فلیپ فلاپ‌های D، لازم است به ستون‌های حالت بعدی در جدول حالت رفته و همه شرایطی که توسط آنها هر فلیپ فلاپ به 1 می‌رود را مشخص نماییم. از جدول ۳-۸، توجه دارید که ستون حالت بعدی G_1 تنها یک 1 در پنجمین ستون دارد. ورودی D فلیپ فلاپ G_1 باید در حین حالت فعلی T_1 وقتی هر دو ورودی A_3 و A_4 برابر 1 هستند، برابر 1 گردد. این نکته با معادله ورودی فلیپ فلاپ D چنین نشان داده می‌شود.

$$D_{G1} = T_1 A_3 A_4$$

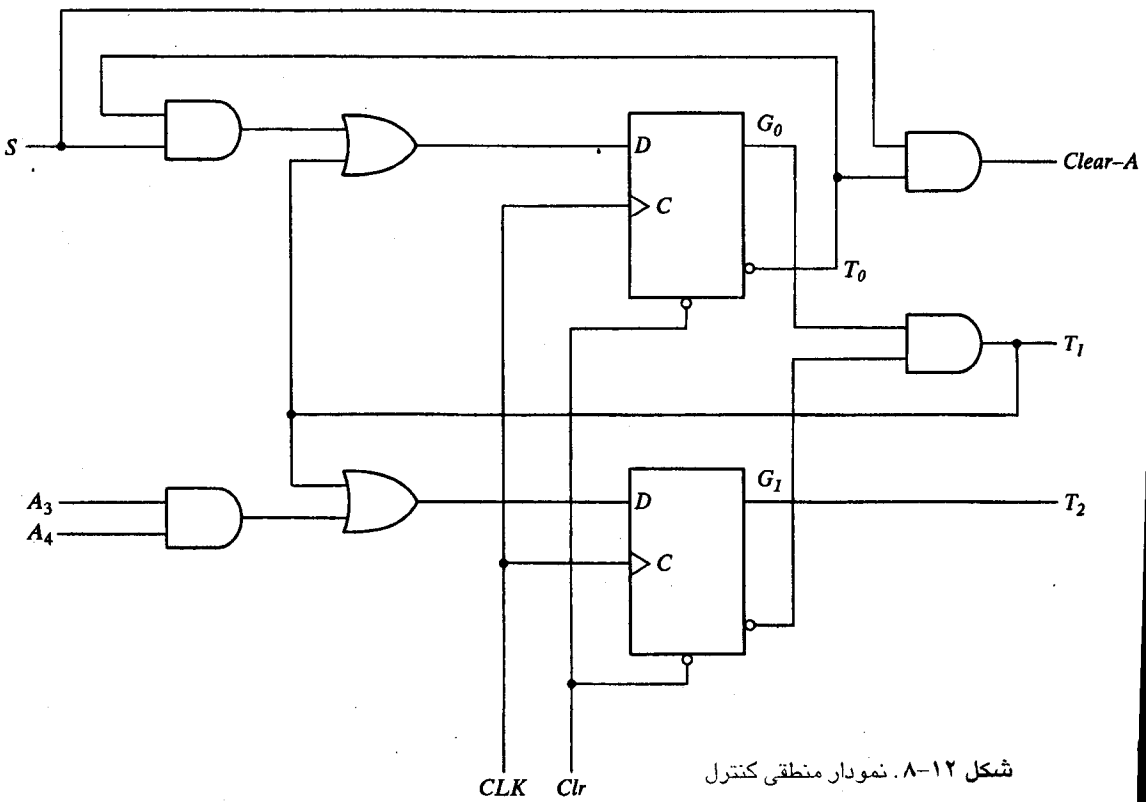
به طور مشابه ستون حالت بعدی G_0 دارای چهار 1 بوده و شرط 1 شدن آن چنین است.

$$D_{G0} = T_0 S + T_1$$

برای بدست آوردن سه تابع خروجی، از حالت بی‌استفاده 10 می‌توان استفاده کرد و معادلات خروجی زیر را بدست آورد.

$$T_0 = G_0'$$
$$T_1 = G_1' G_0$$
$$T_2 = G_1$$

نمودار منطقی کنترل در شکل ۱۲-۸ ملاحظه می‌گردد. این مدار ساختار درونی بلوک کنترل در شکل ۱۰-۸ را نشان می‌دهد که همراه با گیت AND که سیگنال clear A را تولید می‌کند، نشان می‌دهد.



شکل ۱۲-۸. نمودار منطقی کنترل

۸-۵ توصیف HDL مثال طراحی

در فصل‌های قبل، مثال توصیف HDL را برای مدارهای ترکیبی، ترتیبی و قطعات استانداردی چون مولتی پلکسرها و شمارنده‌ها و ثباتها نشان دادیم. اکنون در موقعیتی هستیم تا این اجزاء را در توصیف یک طرح خاص لحاظ کنیم. همانطور که قبلاً اشاره شد یک طرح می‌تواند در سطح ساختاری یا رفتاری باشد. توصیف‌های رفتاری را می‌توان بصورت سطح انتقال ثباتی یا سطح الگوریتمی خلاصه دسته‌بندی کرد. در نتیجه اکنون سه سطح طراحی را مطالعه می‌کنیم: توصیف ساختاری، توصیف RTL و توصیف رفتاری مبتنی بر الگوریتم.

توصیف ساختاری، پایین‌ترین و مشروح‌ترین سطح است. سیستم دیجیتال بر حسب اجزاء فیزیکی و اتصالات درونی‌شان مشخص می‌شوند. انواع اجزاء موجود عبارتند از گیت‌ها، فلیپ‌فلاپ‌ها و مدارهای استانداردی مثل مولتی‌پلکسرها و شمارنده‌ها. طراحی بطور سلسله مراتبی به واحدهای عملیاتی تفکیک می‌شود و هر واحد با مدول HDL توصیف می‌گردد. یک مدول سطح بالا کل سیستم را با ذکر همه مدول‌های سطح پایین‌تر ترکیب می‌کند.

توصیف RTL، سیستم دیجیتال را بر حسب ثبات‌ها، عملیات انجام شده، و کنترلی که ترتیب عملیات را کنترل می‌کند، مشخص می‌نماید. این نوع توصیف متشکل از عبارات اجرایی است که رابطه بین انواع عملیات طرح را بدون ارجاع به ساختار خاص معین می‌کند. توصیف RTL وجود آرایشی سخت افزاری را در میان ثبات‌ها تعیین می‌کند. این کار به طراح اجازه می‌دهد که طرحی را خلق کند که با عناصر دیجیتال استاندارد قابل ساخت است.

توصیف رفتاری مبتنی بر الگوریتم، خلاصه‌ترین سطح است. این روش کار طرح را به فرم یک الگوریتم اجرایی مثل زبان برنامه نویسی توصیف می‌نماید. در این روش هیچ اشاره‌ای به چگونگی پیاده شدن طرح با سخت افزار وجود ندارد. روش فوق الذکر، مناسبترین راه شبیه‌سازی سیستم‌های پیچیده برای تحقیق در مورد ایده‌های طرح است. توصیف‌های این سطح برای درک کاربرهای غیر فنی که زبان برنامه‌نویسی را بدانند در اختیار قرار می‌گیرد. بعضی نتایج ساخت در این سطح ممکن است قابل ترکیب و طراحی نباشد.

اکنون توصیف‌های RTL و ساختاری را با مثال طراحی بخش قبل پی‌می‌گیریم. یک توصیف مبتنی بر الگوریتم در بخش ۸-۸ تشریح شده است.

توصیف RTL

توصیف HDL یک طرح RTL را می‌توان به سه بخش تقسیم کرد. بخش اول ورودی‌ها، خروجی‌ها و ثبات‌ها را در طراحی تعریف می‌کند. دومین بخش رشته کنترل را مشخص می‌نماید. سومین بخش عملیات انتقال ثباتی و خروجی‌ها را فراهم می‌سازد. توصیف RTL مثال طراحی در مثال ۸-۲ HDL نشان داده شده است. به دنبال آن چارت ASM شکل ۸-۹ و ویژگی‌های انتقال ثباتی شکل ۸-۱۱ آمده است. ورودی‌ها عبارتند از S، CLK (ساعت، و Clr (پاک)). ورودی پاک کردن برای مقداردهی اولیه

```
//RTL description of design example (Fig.8-11)
module Example_RTL (S,CLK,Clr,E,F,A);
//Specify inputs and outputs
//See block diagram Fig. 8-10
   input S,CLK,Clr;
   output E,F;
   output [4:1] A;
//Specify system registers
   reg [4:1] A;                  //A register
   reg E, F;                     //E and F flip-flops
   reg [1:0] pstate, nstate;     //control register
//Encode the states
   parameter T0 = 2'b00, T1 = 2'b01, T2 = 2'b11;
//State transition for control logic
//See state diagram Fig. 8-11(a)
   always @(posedge CLK or negedge Clr)
      if (~Clr) pstate = T0;     //Initial state
      else pstate <= nstate;     //Clocked operations
   always @ (S or A or pstate)
       case (pstate)
         T0: if(S) nstate = T1; else nstate = T0;
         T1: if(A[3] & A[4]) nstate = T2; else nstate = T1;
         T2: nstate = T0;
       endcase
//Register transfer operations
//See list of operations Fig.8-11(b)
   always @(posedge CLK)
       case (pstate)
         T0: if(S)
               begin
                  A <= 4'b0000;
                  F <= 1'b0;
               end
         T1:
               begin
                  A <= A + 1'b1;
                  if (A[3]) E <= 1'b1;
                  else  E <= 1'b0;
               end
         T2:  F <= 1'b1;
      endcase
endmodule
```

حالت کنترل با T_0 مورد نیاز است. خروجی‌ها عبارتند از فلیپ فلاپ‌های E و F و ثبات A. ثبات کنترل با یک بردار 2 بیت که شرایط فعلی و بعدی را نگه می‌دارد مشخص شده است. به سه شرط کنترل نام سمبلیک داده شده و به مقادیر دودویی انکد شده است.

بخش بعدی توصیف RTL رشته کنترل را با دو عبارت always مشخص می‌نماید. اولین بلوک always دو عمل را مهیا می‌کند: ورودی Clr حالت فعلی را به T_0 مقداردهی اولیه می‌نماید، و لبه مثبت CLK (posedge) گذر حالت را با ساعت همگام یا همزمان می‌کند. دومین بلوک always شامل یک شرط case است که گذر از حالت فعلی به حالت بعدی را مشخص می‌کند. مثلاً اگر حالت فعلی T_0 و $S = 1$ باشد حالت بعدی T_1 خواهد بود. اگر $S = 0$ باشد، T_0 تغییر نخواهد کرد. تغییر از T_0 به T_1 (با $S = 1$) فقط در لبه مثبت CLK اجرا خواهد شد که با اولین عبارت always دیکته شده است.

سومین بخش توصیف RTL عمل انتقال ثباتی را در هر سه حالت کنترل معین می‌سازد. این بخش عملیات انتقال ثباتی لیست شده در شکل ۱۱-۸ را دنبال می‌نماید. توجه کنید که برای عملیات انتقال ثباتی از عبارات غیربلوکی (= >) استفاده شده است. این کاربری خصوصاً در حین حالت کنترل T_1 حیاتی است. در این حالت، A یک واحد افزایش می‌یابد و مقدار A[3] برای تعیین حالت E چک می‌شود. اگر از تخصیص بلوکی استفاده شود، باید دو عبارت یکی برای چک کردن E و سپس افزایش عبارت A به کار رود. با این وجود با استفاده از تخصیص‌های غیربلوکی، باید همزمانی لازم بدون توجه به رتبه عبارات لیست شده ایجاد شود.

تست توصیف طرح

رشته عملیات مثال طراحی در بخش قبل بررسی شد. جدول ۲-۸ مقادیر E و F را ضمن افزایش A نشان می‌دهد. در اینجا توصیه می‌شود برنامه تستی پیاده شود تا اعتبار توصیف HDL مشخص شود. برنامه تست (test bench) در مثال ۳-۸ HDL چنین مدولی را تهیه می‌نماید. (روال نوشتن یک برنامه تست در بخش ۱۱-۴ ملاحظه شد). مدول تست سیگنال‌هایی برای S، CLK، Clr تولید کرده و نتایج حاصل از ثبات‌های A، E و F را چک می‌کند. در آغاز سیگنال Clr در 0 قرار می‌گیرد تا کنترل آغاز گردد و نیز S و CLK برابر 0 می‌شوند. در $t = 5$، Clr با تنظیم آن در 1 غیرفعال می‌شود، با 1 کردن ورودی S، این ورودی فعال می‌شود، و ساعت برای 16 سیکل تکرار می‌گردد. عبارت $monitor مقادیر A، E و F را هر 10ns یک بار نشان می‌دهد. خروجی شبیه‌سازی که در مثال لیست شده، به صورت لگاریتم است. در آغاز، $t = 0$، مقدار ثبات‌ها نامعلومند بنابراین با سمبل x علامت‌زنی شده‌اند. اولین گذر لبه مثبت ساعت در $t = 0$، A و F را پاک می‌کند، ولی E را تغییر نمی‌دهد، پس E در این زمان نامعلوم است. بقیه جدول مثل جدول ۲-۸ است. توجه کنید که چون S در $t = 160$ هنوز برابر 1 است آخرین وارده در جدول نشان می‌دهد که A و F برابر 0 شده ولی E تغییری نکرده و در 1 باقی می‌ماند. این حالت در دومین گذر از T_0 به T_1 رخ می‌دهد.

```
//Test bench for design example
module test_design_example;
  reg S, CLK, Clr;
  wire [4:1] A;
  wire E, F;
//Instantiate design example
   Example_RTL dsexp (S,CLK,Clr,E,F,A);
    initial
      begin
           Clr = 0;
           S = 0;
           CLK = 0;
        #5 Clr = 1; S = 1;
           repeat (32)
           begin
            #5 CLK = ~ CLK;
           end
      end
    initial
     $monitor("A = %b E = %b F = %b time = %0d", A,E,F,$time);
endmodule

Simulation log:

A = xxxx E = x F = x time = 0
A = 0000 E = x F = 0 time = 10
A = 0001 E = 0 F = 0 time = 20
A = 0010 E = 0 F = 0 time = 30
A = 0011 E = 0 F = 0 time = 40
A = 0100 E = 0 F = 0 time = 50
A = 0101 E = 1 F = 0 time = 60
A = 0110 E = 1 F = 0 time = 70
A = 0111 E = 1 F = 0 time = 80
A = 1000 E = 1 F = 0 time = 90
A = 1001 E = 0 F = 0 time = 100
A = 1010 E = 0 F = 0 time = 110
A = 1011 E = 0 F = 0 time = 120
A = 1100 E = 0 F = 0 time = 130
A = 1101 E = 1 F = 0 time = 140
A = 1101 E = 1 F = 1 time = 150
A = 0000 E = 1 F = 0 time = 160
```

توصیف ساختاری

توصیف RTL یک طرح متشکل از عبارات اجرایی است که رفتار عملیاتی یک مدار دیجیتال را معین می‌کند. این نوع نمایش می‌تواند به وسیله ابزار سنتز (طراحی) HDL مورد استفاده قرار گیرد تا

مدار معادل سطح گیت طرح مشخص گردد. می‌توان طرح را برحسب ساختارش به جای عملکرد توصیف کرد. توصیف ساختاری طرح متشکل از ذکر اجزایی است که ساختار اتصالات میان آنها را تعریف می‌کند. از این لحاظ یک توصیف ساختاری با نمودار طرح یا نمودار بلوکی مدار برابر است.

نمودار بلوکی شکل ۱۰-۸ اطلاعات مورد نیاز توصیف ساختاری را تعریف می‌کند. برای سادگی مدار به سه بخش زیر تفکیک می‌شود.

۱- بلوک کنترل
۲- فلیپ فلاپ‌های E و F و گیت‌های مربوطه
۳- شمارنده با پاک کننده همزمان

توصیف سلسله مراتبی می‌تواند با ذکر تودرتو یا لانه‌ای سه بخش ایجاد شود.

مثال ۴-۸ HDL توصیف ساختاری مثال طراحی را نشان می‌دهد. این توصیف از شش مدول تشکیل شده است که با چهار بخش زیر قابل توصیف است.

۱- اولین مدول سه عنصر را ذکر می‌کند.
۲- دو مدول بعدی کنترل و فلیپ فلاپ D را توصیف می‌نمایند.
۳- دو مدول بعدی E و F و فلیپ فلاپ JK آنها را توصیف می‌کند.
۴- آخرین مدول شمارنده را توصیف می‌نماید.

اولین مدول ورودی‌ها و خروجی‌ها به مدار را اعلان می‌نماید. لیست پورت با لیست توصیف RTL یکی است. در اینجا خروجی‌ها به صورت reg اعلان نمی‌شوند، زیرا در مدول‌های سطح پایین‌تر با reg معرفی شده‌اند. مدول سطح بالا همه طرح را با ذکر سه جزء سطح پایین در برمی‌گیرد. بعضی از پورت‌ها در مدول‌های ذکر شده ورودی‌ها یا خروجی‌های مدارند. دیگر پورت‌ها ورودی‌های تولیدی از دیگر مدول‌ها یا خروجی‌های مورد لزوم دیگر مدول‌ها هستند. مثلاً مدول کنترل ct1 دارای ورودی‌های A[3] و A[4] است که از خروجی A[4 : 1] مدول counter ctr می‌آید. خروجی‌های T_1 و T_2 در مدول ct1 به عنوان ورودی در مدول EF استفاده شده است.

مدول کنترل مدار شکل ۱۲-۸ را توصیف می‌نماید. خروجی‌های دو فلیپ فلاپ G_1 و G_0 و ورودی‌های DG1 و DG0 نمی‌توانند به صورت reg اعلان شوند زیرا در عبارات تخصیص مداوم به کار رفته‌اند. پنج عبارت assign بخش ترکیبی مدار را مشخص می‌نمایند. دو معادله ورودی فلیپ فلاپ و سه معادله خروجی وجود دارد. خروجی‌های فلیپ فلاپ‌های G1 و G0 و معادلات ورودی DG1 و DG0 جایگزین خروجی Q و ورودی D دو فلیپ فلاپ‌های فوق‌الذکر شده‌اند. آنگاه فلیپ فلاپ D در مدول بعدی توصیف شده است. مدول EF الگوی یکسانی با دو فلیپ فلاپ JK را دنبال می‌کند. معادلات ورودی فلیپ فلاپ ابتدا بدست می‌آیند و فلیپ فلاپ JK با این مقادیر به عنوان ورودی‌ها ذکر می‌شود. آخرین مدول شمارنده را با پاک کننده همگام توصیف می‌نماید.

توصیف ساختاری با برنامه تست مثال ۳-۸ تست شد. تنها تغییر لازم جایگزینی ذکر مثال از

```
//Structural description of design example
//See block diagram Fig. 8-10
module Example_Structure (S,CLK,Clr,E,F,A);
   input S,CLK,Clr;
   output E,F;
   output [4:1] A;
//Instantiate control circuit
   control ctl (S,A[3],A[4],CLK,Clr,T2,T1,Clear);
//Instantiate E and F flip-flips
   E_F EF (T1,T2,Clear,CLK,A[3],E,F);
//Instantiate counter
   counter ctr (T1,Clear,CLK,A);
endmodule

//Control circuit (Fig. 8-12)
module control (Start,A3,A4,CLK,Clr,T2,T1,Clear);
   input Start,A3,A4,CLK,Clr;
   output T2,T1,Clear;
   wire G1,G0,DG1,DG0;
//Combinational circuit
   assign  DG1 = A3 & A4 & T1,
           DG0 = (Start & ~G0) | T1,
           T2  = G1,
           T1  = G0 & ~G1,
           Clear = Start & ~G0;
//Instantiate D flip-flop
   DFF G1F (G1,DG1,CLK,Clr),
       G0F (G0,DG0,CLK,Clr);
endmodule

//D flip-flop
module DFF (Q,D,CLK,Clr);
   input D,CLK,Clr;
   output Q;
   reg Q;
   always @ (posedge CLK or negedge Clr)
     if (~Clr) Q = 1'b0;
     else Q = D;
endmodule

//E and F flip-flops
module E_F (T1,T2,Clear,CLK,A3,E,F);
   input T1,T2,Clear,CLK,A3;
   output E,F;
   wire E,F,JE,KE,JF,KF;
```

```
//Combinational circuit
   assign JE = T1 & A3,
          KE = T1 & ~A3,
          JF = T2,
          KF = Clear;
//Instantiate JK flip-flop
   JKFF EF (E,JE,KE,CLK),
        FF (F,JF,KF,CLK);
endmodule

//JK flip-flop
module JKFF (Q,J,K,CLK);
   input J,K,CLK;
   output Q;
   reg Q;
   always @ (posedge CLK)
     case ({J,K})
       2'b00: Q = Q;
       2'b01: Q = 1'b0;
       2'b10: Q = 1'b1;
       2'b11: Q = ~Q;
     endcase
endmodule

//counter with synchronous clear
module counter (Count,Clear,CLK,A);
   input Count,Clear,CLK;
   output [4:1] A;
   reg [4:1] A;
   always @ (posedge CLK)
     if (Clear) A<= 4'b0000;
     else if (Count) A <= A + 1'b1;
     else A <= A;
endmodule
```

Example_RTL به Example_Structure است. نتیجه شبیه‌سازی برای توصیف ساختاری مثل شبیه‌سازی خروجی حاصل از توصیف RTL می‌باشد.

۶-۸ ضرب‌کننده دودویی

این بخش دومین طراحی را معرفی می‌کند. مدار یک الگوریتم سخت‌افزاری را برای ضرب کننده دودویی نمایش داده و آرایش ثبات را برای پیاده‌سازی آن پیشنهاد نموده و سپس برای نمایش طراحی مسیر داده و کنترل به کمک چارت ASM به پیش می‌رود.

سیستمی که معرفی خواهیم کرد دو عدد دودویی بی‌علامت را در هم ضرب می‌کند. در بخش ۶-۴،

یک الگوریتم سخت‌افزاری برای اجرای ضرب ایجاد شد که نتیجه‌اش یک مدار ترکیبی ضرب با چند جمع‌کننده و گیت‌های AND بود. برعکس در این بخش، الگوریتم سخت‌افزاری یک ضرب کننده ترتیبی که فقط از یک جمع کننده و یک شیفت رجیستر ساخته شده را بدست می‌دهد.

ضرب دو عدد دودویی با کاغذ و قلم با جمع و شیفت متوالی انجام می‌گردد. این فرآیند با یک مثال عددی بهتر تشریح می‌شود. اکنون ضرب دو عدد دودویی 10111 و 10011 را در نظر بگیرید:

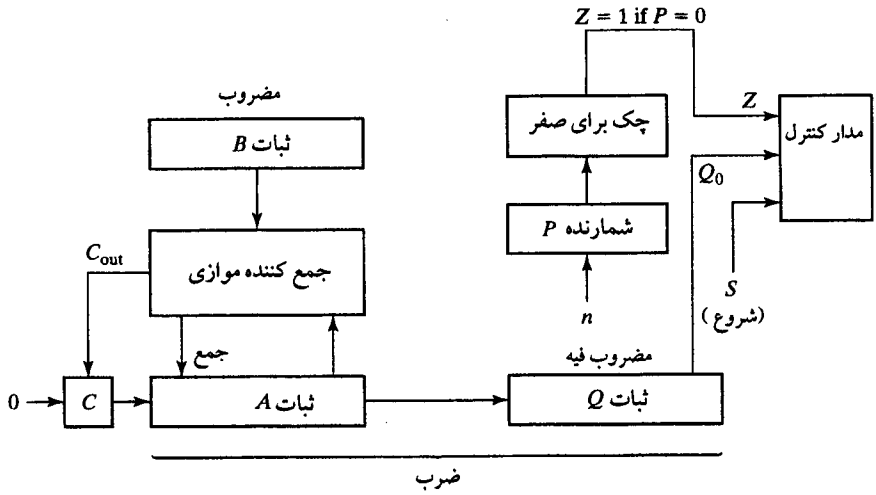
مضروب	10111	23
مضروب فیه	10011	19
	10111	
	10111	
	00000	
	00000	
	10111	
حاصلضرب	110110101	437

فرآیند شامل نگاه به بیت‌های متوالی مضروب‌فیه از سمت کم‌ارزش‌ترین بیت است. اگر بیت مضروب فیه 1 است، مضروب در پایین کپی می‌شود؛ در غیر این صورت 0 کپی می‌گردد. اعدادی که در خطوط متوالی کپی می‌شوند نسبت به قبل یک مکان به چپ جابجا می‌شوند. بالاخره اعداد با هم جمع شده و حاصل جمع آنها، حاصلضرب را تشکیل خواهد داد. حاصلضرب حاصل از ضرب دو عدد n بیتی می‌تواند تا 2n بیت فضا اشغال نماید.

وقتی که روال ضرب با سخت‌افزار دیجیتال پیاده شود، بهتر است تا فرآیند کمی تغییر کند. ابتدا در عوض تهیه مدار دیجیتال برای ذخیره و جمع همزمان به تعداد 1ها در مضروب فیه، بهتر است مداری فراهم گردد که فقط جمع دو عدد دودویی محاسبه شده و به طور متوالی حاصلضرب‌های جزیی حاصل در یک ثبات انبار شود. دوم، در عوض جابجایی مضروب به چپ حاصلضرب جزیی به راست جابجا می‌شود. این موجب می‌شود تا مضروب و حاصلضرب جزیی در کنار هم قرار گیرند. سوم، وقتی بیت مورد نظری در مضروب فیه 0 است، نیاز به جمع 0ها به حاصلضرب جزیی نیست زیرا مقدار حاصل را تغییر نمی‌دهد.

آرایش ثبات

نمودار بلوکی برای مضروب‌فیه دودویی در شکل ۱۳-۸ نشان داده شده است. مضروب در ثبات B ، مضروب فیه در ثبات Q و حاصلضرب جزیی در ثبات A ایجاد و در A و Q ذخیره می‌گردند. یک جمع کننده موازی محتوای ثبات B را به ثبات A اضافه می‌کند. فلیپ فلاپ C ، رقم نقلی را پس از جمع ذخیره می‌نماید. شمارنده P در آغاز با یک عدد که برابر تعداد 1ها در مضروب فیه است پر می‌شود. این شمارنده پس از هر بار تشکیل حاصلضرب جزیی یک واحد کم می‌شود. وقتی محتوای شمارنده به صفر برسد، حاصل ضرب کل در جفت ثبات A و B تشکیل شده است و بنابراین فرآیند متوقف می‌گردد.

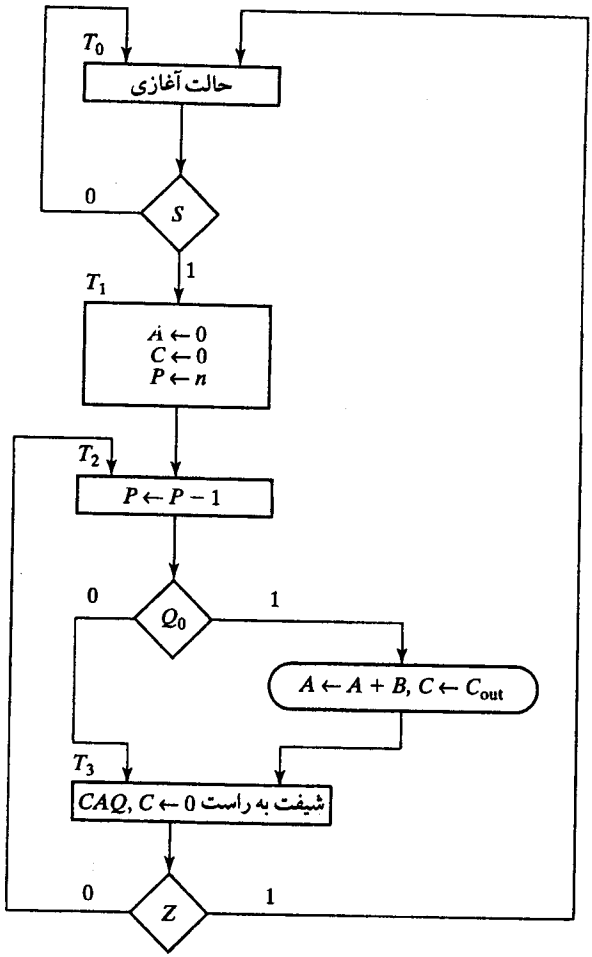


شکل ۱۳-۸. نمودار بلوکی ضرب کننده دودویی

مدار کنترل تا سیگنال شروع S برابر 1 شود در حالت اولیه باقی می‌ماند. آنگاه سیستم عمل ضرب را انجام می‌دهد. حاصل جمع A و B، n بیت باارزش‌تر حاصلضرب جزیی را تولید می‌کند که به A منتقل می‌گردد. نقلی خروجی از جمع، چه 0 و چه 1 به C منتقل می‌گردد. هر دو مقدار ضرب جزیی در A و مضروب فیه در Q به راست جابجا می‌شوند. بیت کم‌ارزش‌تر A به باارزش‌ترین بیت در Q منتقل می‌گردد؛ نقلی از C به باارزش‌ترین مکان در A می‌رود؛ و 0 به C وارد می‌شود. پس از عمل جابجایی به راست، یک بیت از حاصلضرب جزیی وارد Q می‌گردد ضمن این که بیت‌های مضروب فیه در Q یک مکان به راست می‌روند. به این ترتیب کم‌ارزش‌ترین بیت ثبات Q که به آن Q_0 می‌گوییم بیتی از مضروب فیه را نگه می‌دارد که در مرحله بعدی برای ادامه کار باید مورد وارسی قرار گیرد. مدار کنترل، جمع کردن یا نکردن را برحسب این بیت ورودی معین خواهد کرد. این مدار سیگنال Z را هم از مداری که شمارنده P را برای وجود صفر چک می‌کند دریافت می‌نماید. Q_0 و Z ورودی‌های وضعیت واحد کنترل هستند. ورودی شروع S یک ورودی بیرونی به کنترل است. خروجی‌های مدار کنترل عملیات لازم را در ثبات‌ها فعال می‌کنند.

چارت ASM

چارت ASM برای ضرب کننده دودویی در شکل ۱۴-۸ ملاحظه می‌شود. در آغاز مضروب در B و مضروب فیه در Q است. مادامی که مدار در حالت شروع و $S = 0$ است، هیچ اتفاقی رخ نمی‌دهد و سیستم در حالت اولیه T_0 باقی خواهد ماند. فرآیند ضرب با $S = 1$ شروع شده و کنترل به حالت T_1 می‌رود. ثبات A و فلیپ فلاپ C به 0 پاک می‌شوند و شمارنده دفعات، با عدد n مقداردهی می‌شود که n تعداد بیت‌ها در مضروب فیه است. آنگاه سیستم به T_2 می‌رود. بیت مضروب فیه در Q_0 چک می‌شود



شکل ۱۴-۸. چارت ASM برای ضرب دودویی

و اگر برابر 1 بود، مضروب در B به حاصلضرب جزیی در A اضافه می‌شود. رقم نقلی حاصل از جمع به C منتقل می‌گردد. حاصلضرب جزیی در A و C اگر $Q_0 = 0$ باشد، دست نخورده باقی می‌مانند. شمارنده P بدون توجه به مقدار Q_0 یک واحد کم می‌شود. در هر حال حالت بعدی T_3 است. ثبات‌های A، C و Q در یک ثبات مرکب CAQ تصور خواهند شد و محتوای آن یک بار به راست جابجا می‌شود تا حاصلضرب جزیی جدیدی حاصل گردد. این عمل شیفت در فلوچارت به فرم عبارت زیر نمایش شده است

$$\text{Shift right } CAQ, C \leftarrow 0$$

بااستفاده از سمبل‌های ثباتی جداگانه، عمل جابجایی با عملیات ثباتی زیر توصیف می‌گردد

$$A \leftarrow \text{shr } A, A_{n-1} \leftarrow C$$
$$Q \leftarrow \text{shr } Q, Q_{n-1} \leftarrow A_0$$
$$C \leftarrow 0$$

هر دو ثبات A و Q به راست جابجا شده‌اند. سمت چپ‌ترین بیت A که با A_{n-1} نشان داده می‌شود رقم نقلی C را دریافت می‌کند. سمت چپ‌ترین بیت Q، یا Q_{n-1} سمت راست‌ترین بیت A یعنی A_0 را دریافت می‌نماید و C به 0 بازنشانی می‌گردد. به بیان دیگر این یک جابجایی طویل در ثبات مرکب CAQ است با یک 0 که از ورودی سریال متصل به C وارد آن می‌شود.

مقدار موجود در شمارنده P پس از تشکیل هر حاصلضرب جزیی چک می‌شود. اگر محتوای P صفر نباشد بیت وضعیت Z برابر 0 بوده و فرآیند برای تشکیل یک حاصلضرب جزیی تکرار می‌گردد. فرآیند با رسیدن شمارنده P به 0 پایان یافته و در این حال ورودی کنترل برابر 1 خواهد بود. توجه کنید که حاصلضرب جزیی تشکیل شده در A به اندازه یک بیت در هر بار به Q منتقل می‌شود و نهایتاً جایگزین مضروب فیه خواهد شد. حاصلضرب نهایی در A و Q خواهد بود که A بیت‌های باارزش‌تر و Q بیت‌های کم‌ارزش‌تر حاصل ضرب است.

مثال عددی قبلی در جدول ۴-۸ برای شفافیت روند ضرب تکرار شده است. روال مراحل ذکر شده در چارت ASM را دنبال می‌کند.

نوع ثبات‌های لازم برای زیر سیستم پردازش داده از عملیات ثباتی لیست شده در چارت ASM بدست می‌آید. ثبات A یک شیفت رجیستر با امکان بار شدن موازی برای پذیرش جمع از جمع‌کننده است و باید قابلیت پاک شدن همزمان را داشته باشد تا ثبات به 0 بازنشانی شود. ثبات Q هم یک شیفت رجیستر است. شمارنده P یک پایین شمار دودویی با امکان بار شدن موازی یک ثابت دودویی است. فلیپ‌فلاپ

جدول ۴-۸. مثال عددی ضرب دودویی

	C	A	Q	P
مضروب B = 10111				
مضروب فیه در Q	0	00000	10011	101
$Q_0 = 1$: با B جمع کن		10111		
ضرب جزئی اول	0	10111		100
جابجایی به راست CAQ	0	01011	11001	
$Q_0 = 1$: با B جمع کن		10111		
ضرب جزئی دوم	1	00010		011
جابجایی به راست CAQ	0	10001	01100	
$Q_0 = 0$ جابجایی به راست CAQ	0	01000	10110	010
$Q_0 = 0$ جابجایی به راست CAQ	0	00100	01011	001
$Q_0 = 1$: با B جمع کن		10111		
پنجمین ضرب جزئی	0	11011		
جابجایی به راست CAQ	0	01101	10101	000
حاصلضرب نهایی در AQ = 0110110101				

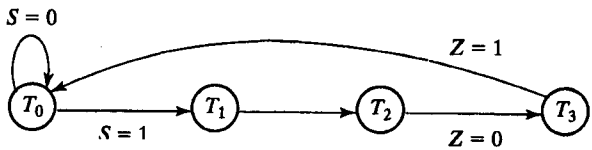
C باید طوری طراحی شود تا نقلی ورودی و پاک شدن همزمان را بپذیرد. ثبات‌های B و Q دارای قابلیت بار شدن موازی هستند تا مضروب و مضروب فیه را قبل از شروع فرآیند ضرب، دریافت نمایند.

۸-۷ مدارکنترل

طراحی یک سیستم دیجیتال را می‌توان به دو بخش تفکیک کرد: طراحی انتقال ثباتی در مسیر داده و طراحی مدارکنترل. طراحی مدارکنترل یک طراحی مدار ترتیبی است. در این صورت بهتر است نمودار کنترل ترتیبی را بدست آوریم. یک چارت ASM مشابه یک نمودار حالت است. بلوک‌های مستطیلی که جعبه‌های حالت را تداعی می‌کنند حالات مدار ترتیبی هستند. بلوک‌های لوزی شکلی که جعبه تصمیم‌گیری را تداعی می‌نمایند، شرایط را برای گذر به حالت بعدی در نمودار حالت مشخص می‌کنند. به عنوان مثال، نمودار حالت کنترل برای ضرب کننده دودویی که در بخش قبل بدست آمد. در شکل ۸-۱۵ نشان داده شده است. اطلاعات نمودار مستقیماً از چارت ASM شکل ۸-۱۴ حاصل شده است. چهار حالت T_0 تا T_3 از جعبه‌های حالت مستطیلی بدست آمده‌اند. ورودی‌های S و Z از جعبه‌های تصمیم لوزی اخذ شده‌اند. عملیات انتقال ثباتی برای هر چهار حالت در زیر نمودار حالت آمده است. آنها از حالت متناظر و جعبه‌های شرطی در چارت ASM فراهم شده‌اند.

به هنگام پیاده‌سازی یک مدارکنترل دو دیدگاه مورد بحث وجود دارد. یکی ایجاد رشته حالات موردنظر و تهیه سیگنال‌های کنترل عملیات ثباتی است. رشته حالات در نمودار حالت مشخص می‌گردد. سیگنال‌های کنترل عملیات در ثبات‌ها در عبارات انتقال ثبات مشخص می‌شوند. در ضرب کننده، این سیگنال‌ها T_1 (برای پاک کردن A و C و بارکردن یک عدد در P)، T_2 (برای کاهش P)، T_3 (برای شیفت رجیستر CAQ)، و Q_0 جهت افزودن B به A و یا نیفزودن آن می‌باشد. نمودار بلوکی کنترل در شکل ۸-۱۶ دیده می‌شود. ورودی‌های مدار کنترل ترتیبی S و Z هستند و طبق نمودار حالت خروجی‌ها نیز T_0 ، T_1 ، T_2 و T_3 می‌باشند. گیت AND مولد $L = T_2Q_0$ برای بارکردن حاصل جمع در ثبات A به شرط $Q_0 = 1$ در حالت T_2 است.

یکی از مراحل با اهمیت در طراحی، تخصیص کد دودویی به حالات است. ساده‌ترین تخصیص،



T_0: Initial state

T_1: $A \leftarrow 0, C \leftarrow 0, P \leftarrow n$

T_2: $P \leftarrow P - 1$

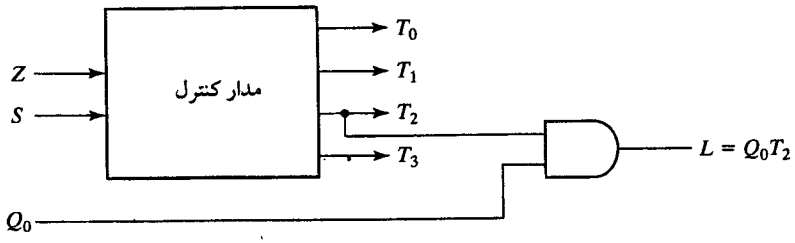
if $(Q_0) = 1$ then $(A \leftarrow A + B, C \leftarrow C_{out})$

T_3: shift right CAQ, $C \leftarrow 0$

(ب) عملیات انتقال ثباتی

(الف) نمودار حالت

شکل ۸-۱۵. مشخصات کنترل برای ضرب دودویی



شکل ۱۶-۸. نمودار بلوکی کنترل

اختصاص مستقیم اعداد دودویی، طبق جدول ۵-۸ می‌باشد. تخصیص مشابه دیگر، کدگری است که در آن هنگام رفتن از یک عدد به بعدی، فقط یک بیت تغییر می‌نماید. تخصیص حالتی که اغلب در طراحی کنترل به کار می‌رود تخصیص 1 _ بارز است. این تخصیص به تعداد حالات مدار از بیت استفاده می‌کند. در هر لحظه از زمان، تنها یک بیت برابر 1 است (1 بارز) در حالی که دیگر بیت‌ها در 0 نگهداشته می‌شوند. این نوع تخصیص برای هر حالت از یک فلیپ فلاپ استفاده می‌کند.

چون کنترل یک مدار ترتیبی است، می‌توان طبق مطالب فصل ۵، آن را با روش مدار ترتیبی طراحی کرد. با این وجود در بسیاری از موارد این روش عملاً پیچیده است زیرا این گونه مدارهای کنترل دارای تعداد زیادی حالات و ورودی هستند. در نتیجه، لازم است از روش‌های خاصی برای طراحی کنترل استفاده شود. اکنون دو نمونه از این گونه مدارها را ارائه می‌کنیم. یکی از آنها از یک ثبات و دیکدر و دیگری از فلیپ فلاپ برای هر حالت استفاده می‌کند.

ثبات و دیکدر

همانطور که از نام این روش پیداست، روش ثبات و دیکدر از یک ثبات برای کنترل حالات و یک دیکدر برای تهیه یک خروجی متعلق به هر حالت استفاده می‌کند. ثباتی با n فلیپ فلاپ، می‌تواند 2^n حالت داشته و دیکدر n به 2^n نیز دارای 2^n خروجی است. یک ثبات اساساً مداری با n فلیپ فلاپ همراه با گیت‌های متعلقه برای تأثیر بر گذر حالت است.

نمودار حالت کنترل برای ضرب دودویی دارای چهار حالت و دو ورودی است. برای پیاده‌سازی آن با یک ثبات و دیکدر، لازم است دو فلیپ فلاپ برای ثبات و یک دیکدر 2 به 4 انتخاب کنیم. گرچه مثال

جدول ۵-۸. تخصیص حالت برای کنترل

حالت	دودویی	کدگری	یک بارز
T_0	00	00	0001
T_1	01	01	0010
T_2	10	11	0100
T_3	11	10	1000

جدول ۶-۸. جدول حالت برای مدار کنترل

حالت فعلی		ورودی		حالت بعدی		خروجی‌ها			
G_1	G_0	S	Z	G_1	G_0	T_0	T_1	T_2	T_3
0	0	0	X	0	0	1	0	0	0
0	0	1	X	0	1	1	0	0	0
0	1	X	X	1	0	0	1	0	0
1	0	X	X	1	1	0	0	1	0
1	1	X	0	1	0	0	0	0	1
1	1	X	1	0	0	0	0	0	1

تقریباً ساده است، ولی روال را می‌توان به مدارهای پیچیده‌تر هم تعمیم داد.

جدول حالت کنترل ترتیبی در جدول ۶-۸ نشان داده شده است. این جدول مستقیماً از نمودار شکل ۱۵-۸ (الف) حاصل شده است. در اینجا دو فلیپ فلاپ را G_1 و G_0 می‌نامیم و حالات 00، 01، 10 و 11 را به ترتیب T_0، T_1، T_2، T_3 می‌خوانیم. توجه کنید که ستون ورودی‌ها دارای وارده‌های بی‌اهمیت‌اند. این حالات بی‌اهمیت متعلق به مواردی است که ورودی برای تعیین حالت بعدی به‌کار نرود. خروجی‌های مدار کنترل با نام حالات معین شده‌اند. حالت 1 برای هر متغیر خروجی خاص در هر زمان از مقدار حالت فعلی‌اش حاصل می‌گردد. بنابراین وقتی حالت فعلی $G_1G_0 = 00$ است، خروجی T_0 باید 1 و دیگر خروجی‌ها باید 0 باشند. چون خروجی‌ها فقط تابعی از حالت فعلی هستند، می‌توان آنها را با مدار دیکدری که دارای دو ورودی G_1 و G_0 و چهار خروجی T_0، T_1، T_2 و T_3 می‌باشند، تولید کرد.

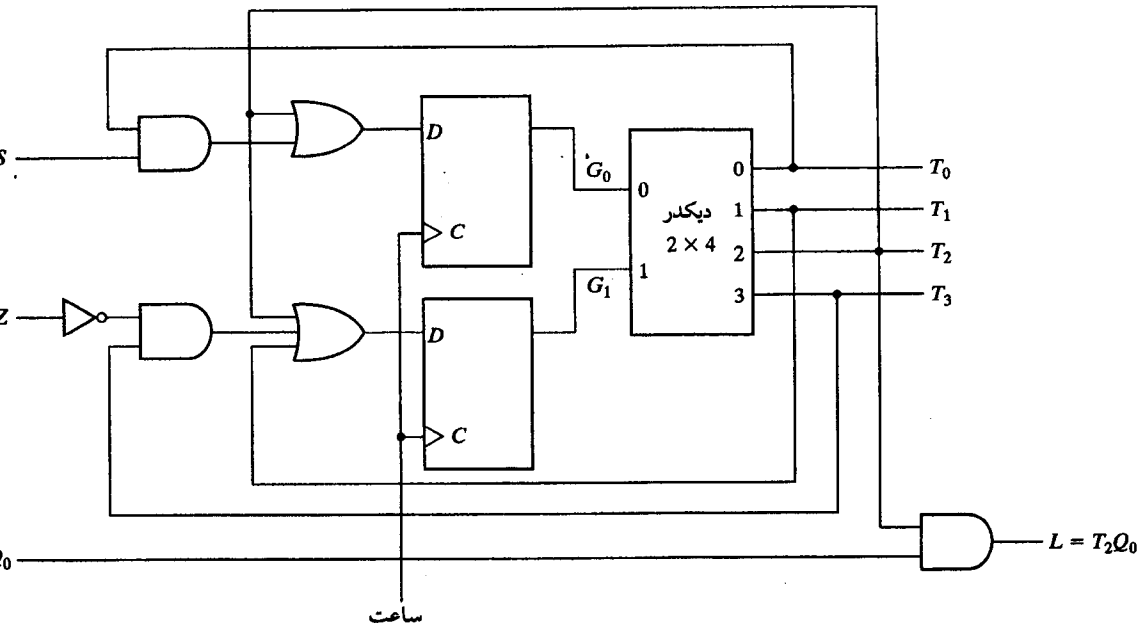
با استفاده از روش کلاسیک فصل ۵ می‌توان مدار ترتیبی را از جدول حالت طراحی کرد. این مثال دارای تعداد کمی حالت و ورودی است و بنابراین می‌توان از نقشه کارنو برای ساده کردن تابع بول استفاده نمود. در بسیاری از کاربردهای مدار کنترل، تعداد ورودی‌ها و حالات بسیار زیادتر است. در تهیه معادلات ورودی ساده شده برای فلیپ فلاپ‌ها، به کارگیری روش کلاسیک کار زیادی را می‌طلبد. چنانچه فرض کنیم خروجی‌های دیکدر برای استفاده در طراحی در دسترسند طراحی بسیار ساده‌تر خواهد شد. در عوض به کارگیری نقشه برای ساده کردن معادلات فلیپ فلاپ‌ها، می‌توانیم آنها را مستقیماً با وارسی جدول حالت بدست آوریم. مثلاً، با توجه به شرایط حالت بعدی در جدول حالت، در می‌یابیم که حالت بعدی G_1 به شرطی 1 می‌باشد که حالت فعلی T_1 یا T_2 و یا T_3 همراه با $Z=0$ باشد. این شرایط را می‌توان با معادله زیر بیان کرد.

$$D_{G1} = T_1 + T_2 + T_3Z'$$

که در آن D_{G1} ورودی D فلیپ فلاپ G_1 است. به طور مشابه ورودی D مربوط به G_0 برابر است با

$$D_{G0} = T_0S + T_2$$

البته با روش بررسی جدول حالت نمی‌توان مطمئن شد که توابع بول به بهترین وجه ساده شده‌اند. به این دلیل، توصیه می‌شود برای اطمینان از صحت معادلات، تحلیل‌های لازم صورت گیرد، تا بدین وسیله گذر حالات لازم تولید شود.



شکل ۸-۱۷ . نمودار منطقی کنترل برای ضرب دودویی با استفاده از یک ثبات و دیکدر

نمودار منطقی مدار کنترل در شکل ۸-۱۷ ترسیم شده است. این مدار متشکل از یک ثبات با دو فلیپ فلاپ G_1 و G_0 و یک دیکدر 4 × 2 است. خروجی‌های دیکدر برای تولید ورودی‌ها به مدار حالت بعدی و خروجی‌های کنترل به کار رفته‌اند. خروجی‌های کنترل‌گر باید برای فعال کردن عملیات ثباتی لازم به مسیر داده (بخش پردازش) متصل گردند.

یک فلیپ فلاپ برای هر حالت

روش ممکن دیگر در طراحی مدار کنترل استفاده از تخصیص 1 بارز است که یک مدار ترتیبی با یک فلیپ فلاپ را نتیجه می‌دهد. هر بار تنها یکی از فلیپ فلاپ‌ها در 1 و بقیه در وضعیت 0 هستند. انتشار یک 1 از یک فلیپ فلاپ به دیگری براساس کنترل مدار تصمیم‌گیری صورت می‌گیرد. در این آرایش هر فلیپ فلاپ یک حالت را نمایش می‌دهد و فقط هنگامی که بیت کنترل به آن منتقل شود وجود دارد.

این روش بیشترین تعداد فلیپ فلاپ را بکار می‌برد. مثلاً یک مدار ترتیبی با 12 حالت به حداقل چهار فلیپ فلاپ نیاز دارد. با روش یک فلیپ فلاپ برای هر حالت، مدار نیاز به 12 فلیپ فلاپ خواهد داشت. در نگاه اول چنین به نظر می‌رسد که به علت وجود تعداد زیادی فلیپ فلاپ، هزینه این سیستم افزایش خواهد یافت. ولی مدار مزایایی را داراست که نمی‌توان آنها را نادیده گرفت. یکی از این مزایا این است که مدار را با وارسی چارت ASM یا نمودار حالت می‌توان طراحی کرد. اگر فلیپ فلاپ‌های نوع D به کار گرفته شود نیازی به جداول حالت و تحریک نیست. دیگر مزایای قابل ذکر، صرفه‌جویی در

تلاش‌هایی است که در طراحی صورت می‌گیرد، افزایش سادگی اعمال، و احتمالاً کاهش در تعداد کل گیت‌ها به دلیل عدم نیاز به دیکدر است.

روال طراحی را می‌توان با مدار کنترل نمودار حالت شکل ۱۵-۸ (الف) نشان داد. چون در نمودار حالت چهار حالت وجود دارد، چهار فلیپ فلاپ D اختیار کرده و خروجی آنها را T_0 ، T_1 ، T_2 و T_3 می‌نامیم. معادلات ورودی برای 1کردن هر فلیپ فلاپ از حالت فعلی و شرایط ورودی در روی خطوط جهت‌داری که به حالت بعدی می‌رود مشخص می‌گردد. مثلاً فلیپ فلاپ T_0 هنگامی با لبه پالس ساعت بعدی 1 می‌شود که مدار در حالت فعلی T_3 و ورودی Z برابر 1 یا اگر مدار در حالت فعلی T_0 است $S = 0$ باشد. این شرایط با معادله ورودی زیر نشان داده می‌شود.

$$D_{T0} = T_0S' + T_3Z$$

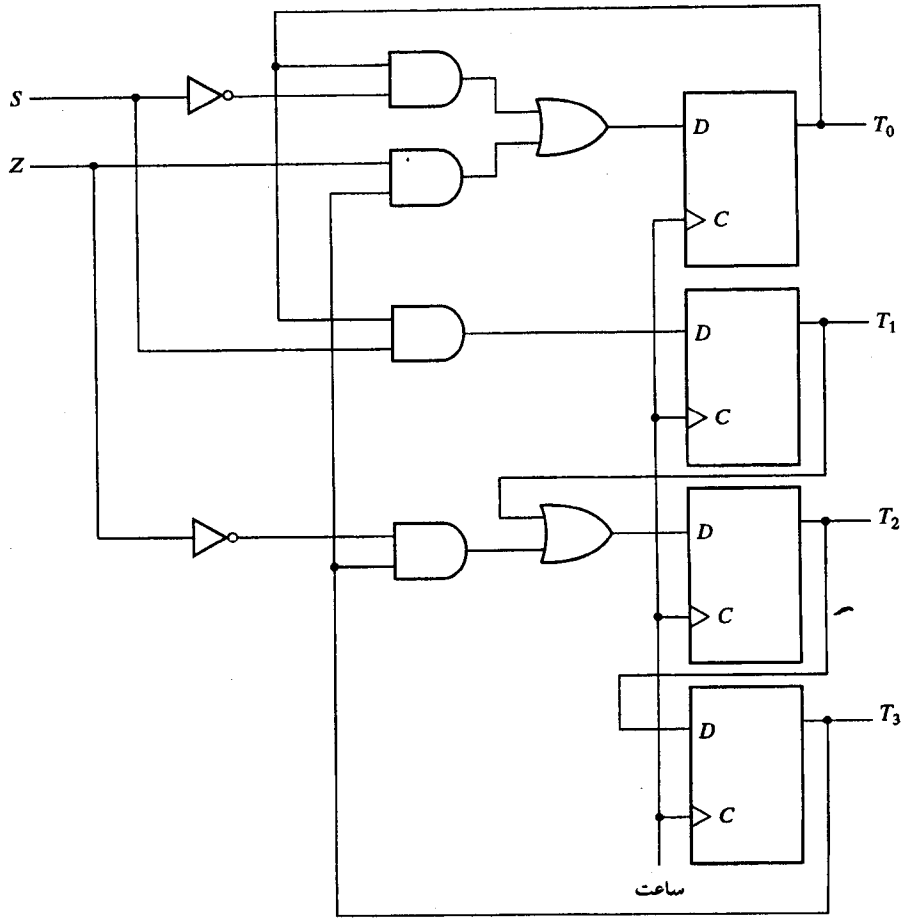
که در آن D_{T0} ورودی D فلیپ فلاپ T_0 است. در واقع، شرط 1کردن یک فلیپ فلاپ مستقیماً از نمودار حالت و از خطوط جهت‌دار متصل به فلیپ فلاپ مربوطه که با حالت بعدی فلیپ فلاپ AND شده بدست می‌آید. اگر تعداد خطوطی که به یک حالت می‌روند بیش از یکی باشد، همه حالات باید OR شوند. با استفاده از این روال برای سه فلیپ فلاپ دیگر معادلات ورودی چنین خواهند بود.

$$D_{T1} = T_0S$$
$$D_{T2} = T_1 + T_3Z'$$
$$D_{T3} = T_2$$

نمودار منطقی کنترل‌گر در شکل ۱۸-۸ دیده می‌شود. این شکل دارای چهار فلیپ فلاپ D از T_0 تا T_3 و گیت‌های مربوطه است که با معادلات ورودی نشان داده شده‌اند. در ابتدا T_0 باید در 1 نشانده شود. همه فلیپ فلاپ‌های دیگر باید 0 گردند، به این ترتیب فلیپ فلاپی که حالت آغازین را نشان می‌دهد فعال می‌گردد. این کار را می‌توان با به کارگیری یک پیش تنظیم غیرهمزمان روی فلیپ فلاپ T_0 و یک پاک کردن غیرهمزمان برای دیگر فلیپ فلاپ‌ها تحقق بخشید. به محض این که مدار شروع به کار کند، کنترل‌گر یک فلیپ فلاپ بر هر حالت شروع به انتشار حالت موجود خواهد کرد. در هر لبه پالس ساعت تنها یک فلیپ فلاپ در 1 و بقیه در 0 خواهند بود، زیرا ورودی آنها 0 است.

۸-۸ توصیف HDL ضرب‌کننده دودویی

دومین مثال برای توصیف HDL یک طرح RTL، در مثال ۵-۸ نشان داده شده است. مثال مربوط به طراحی ضرب کننده دودویی بخش ۶-۸ می‌باشد. می‌توان توصیف را به پنج بخش که هر بخش با توضیحاتی همراه است تقسیم کرد. بخش اول همه ورودی‌ها و خروجی‌ها را طبق شکل ۱۳-۸ لیست می‌کند. ورودی‌های داده به B و Q و خروجی‌های حاصل از A و Q بردارهای 5 بیتی هستند. دلیل انتخاب 5 بیت برای این مثال این است که بتوانیم نتایج را با مثال عددی جدول ۴-۸ مقایسه کنیم. بخش دوم همه ثبات‌ها از جمله ثبات کنترل، همراه با انکد کردن چهار حالت را اعلان می‌کند. بخش سوم مداری ترکیبی را با عبارت **assign** مشخص می‌نماید. در ضرب کننده دو مدار ترکیبی وجود دارد. یکی



شکل ۱۸-۸. نمودار منطقی کنترل با استفاده از یک فلیپ فلاپ در هر مرحله

از آنها خروجی شمارنده را برای یافتن صفر خروجی وارسی می‌کند؛ و دیگری یک جمع‌کننده موازی است. مبدأ جمع کننده به عنوان بخشی از عملگر انتقال ثباتی که مضروب را با حاصلضرب جزیی جمع می‌کند در نظر گرفته خواهد شد. شناسایی صفر با یک گیت NOR پنج ورودی انجام می‌گردد.

بخش چهارم گذر حالت را برای کنترل توصیف کرده و نمودار حالت شکل ۱۵-۸ (الف) را دنبال می‌نماید. آخرین بخش توصیف عملیات انتقال ثباتی برای بخش مسیر داده است. به دنبال آن اعمال لیست شده در چارت ASM شکل ۱۵-۸ (ب) آمده است. در چارت ASM ورودی‌های داده لیست نشده‌اند، ولی در اینجا برای چک کردن عمل مدار باید لحاظ شوند. در حین حالت آغازین، مضروب به ثبات B انتقال می‌یابد. مضروب فیه در زمان T_1 منتقل می‌شود. در T_2، مضروب به حاصلضرب جزیی اضافه می‌شود به شرطی که Q[0] = 1 باشد. در T_3، حاصلضرب جزیی به راست جابجا می‌شود.

```
//RTL description of binary multiplier
//Block diagram Fig. 8-13 and ASM chart Fig. 8-14
//n = 5 to compare with Table 8-4
module mltp(S,CLK,Clr,Binput,Qinput,C,A,Q,P);
   input S,CLK,Clr;
   input [4:0] Binput,Qinput;          //Data inputs
   output C;
   output [4:0] A,Q;
   output [2:0] P;
//System registers
   reg C;
   reg [4:0] A,Q,B;
   reg [2:0] P;
   reg [1:0] pstate, nstate;           //control register
   parameter T0=2'b00, T1=2'b01, T2=2'b10, T3=2'b11;
//Combinational circuit
   wire Z;
   assign Z = ~|P;                     //Check for zero
//State transition for control
//See state diagram Fig. 8-15(a)
   always @(negedge CLK or negedge Clr)
     if (~Clr) pstate = T0;
     else pstate <= nstate;
   always @(S or Z or pstate)
     case (pstate)
       T0: if (S) nstate = T1; else nstate = T0;
       T1: nstate = T2;
       T2: nstate = T3;
       T3: if (Z) nstate = T0;
           else   nstate = T2;
     endcase
//Register transfer operations
//See register operation Fig. 8-15(b)
   always @(negedge CLK)
     case (pstate)
       T0: B <= Binput;                //Input multiplicand
       T1: begin
             A <= 5'b00000;
             C <= 1'b0;
             P <= 3'b101;             //Initialize counter to n=5
             Q <= Qinput;              //Input multiplier
           end
       T2: begin
             P <= P - 3'b001;          //Decrement counter
             if (Q[0])
             {C,A} <= A + B;           //Add multiplicand
           end
       T3: begin
             C <= 1'b0;                //Clear C
             A <= {C,A[4:1]};          //Shift right A
             Q <= {A[0],Q[4:1]};       //Shift right Q
           end
     endcase
endmodule
```

مثال ۶-۸ HDL یک برنامه تست را برای تست ضرب کننده نشان می‌دهد. ورودی‌ها عبارتند از: مضروب و مضروب فیه در B و Q . خروجی حاصلضرب در A و Q قرار دارد. ما مقدار نقلی C و شمارنده P را هم چک می‌کنیم. ورودی‌های دودویی 5 بیتی همان ورودی‌های به کار رفته در جدول ۴-۸ هستند. دومین عبارت آغازین 13 پالس ساعت را 10 واحد زمانی برای هر کدام فراهم می‌نماید. بررسی گذر حالات نشان می‌دهدکه سیکل‌های T_2 و T_3 ، در حلقه 5 بار تکرار می‌شوند (برای هر حاصلضرب جزیی یک بار). این خود 10 پالس ساعت را لازم دارد. به هنگام $Z = 1$ ، سه سیکل ساعت دیگر هم برای T_0 ، T_1 و بازگشت به T_0 لازم است.

نتایج محاسبات با استفاده از تکلیف سیستم $strobe نمایش داده شده است. این تکلیف مشابه تکلیف‌های $display و $monitor تشریح شده در بخش ۱۱-۴ می‌باشد. تکلیف $strobe یک مکانیزم همزمانی را برای اطمینان از نمایش داده پس از اجرای همه عبارات در یک مرحله از زمان فراهم می‌کند. این مکانیزم در مدارهای ترتیبی همزمانی که در آنها هر مرحله زمانی با لبه ساعت آغاز می‌شود اهمیت دارد. استفاده از $strobe پس از عبارت always @ negedge CLK به این معنی است که صفحه نمایش مقادیر سیگنال را پس از لبه منفی ساعت نشان خواهد داد.

مدول test_mltp در مثال ۶-۸ HDL مدول mltp در مثال ۵-۸ را ذکر می‌نماید. هنگام شبیه‌سازی با Verilog HDL هر دو مدول باید لحاظ شوند. نتیجه این شبیه‌سازی اعدادی مشابه با جدول ۴-۸ را نمایش خواهد داد.

مثال ۶-۸ HDL

```
//Testing binary multiplier
module test_mltp;
//Inputs for multiplier
   reg S,CLK,Clr;
   reg [4:0] Binput,Qinput;
//Data for display
   wire C;
   wire [4:0] A,Q;
   wire [2:0] P;
//Instantiate multiplier
   mltp mp(S,CLK,Clr,Binput,Qinput,C,A,Q,P);
   initial
     begin
          S=0; CLK=0; Clr=0;
      #5  S=1; Clr=1;
          Binput = 5'b10111;
          Qinput = 5'b10011;
      #15 S = 0;
     end
```

```
   initial
     begin
         repeat (26)
      #5 CLK = ~CLK;
     end
//Display computations and compare with Table 8-4
   always @(negedge CLK)
     $strobe ("C=%b  A=%b  Q=%b  P=%b  time=%0d",C,A,Q,P,$time);
endmodule
```

```
Simulation log:

C=x  A=xxxxx  Q=xxxxx  P=xxx  time=10
C=0  A=00000  Q=10011  P=101  time=20
C=0  A=10111  Q=10011  P=100  time=30
C=0  A=01011  Q=11001  P=100  time=40
C=1  A=00010  Q=11001  P=011  time=50
C=0  A=10001  Q=01100  P=011  time=60
C=0  A=10001  Q=01100  P=010  time=70
C=0  A=01000  Q=10110  P=010  time=80
C=0  A=01000  Q=10110  P=001  time=90
C=0  A=00100  Q=01011  P=001  time=100
C=0  A=11011  Q=01011  P=000  time=110
C=0  A=01101  Q=10101  P=000  time=120
C=0  A=01101  Q=10101  P=000  time=130
```

توصیف رفتاری ضرب کننده

می‌توان یک ضرب کننده دودویی را در سطح رفتاری مانند مثال ۸-۷ HDL هم توصیف کرد. دو ورودی به عنوان بردارهای 8 بیت و خروجی به عنوان بردار 16 بیت اعلان شده‌اند. در اینجا سخت‌افزار خاصی مانند آنچه در توصیف‌های RTL یا ساختاری ملاحظه شد، مورد نیاز نیست. اجرا فقط با یک عبارت ضرب صورت می‌گیرد. در این حال، اگر نرم‌افزار سنتز (طراحی) طراحی شده باشد، توصیف ممکن است برای یک مدار ترکیبی ضرب ایجاد گردد. به طور کلی، توصیف‌های مبتنی بر الگوریتم همیشه قابل سنتز یا ترکیب نیستند.

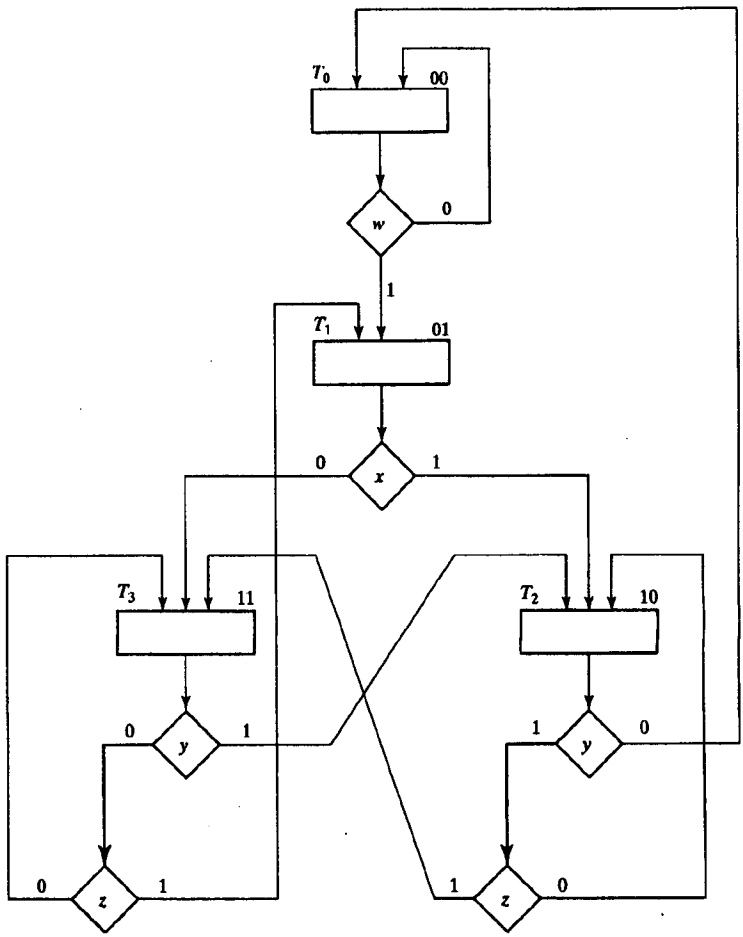
مثال ۸-۷ HDL

```
//Behavioral description of multiplier (n = 8)
module Mult (A,B,Q);
   input [7:0] B,Q;
   output [15:0] A;
   reg [15:0] A;
   always @ (B or Q)
     A = B * Q;
endmodule
```

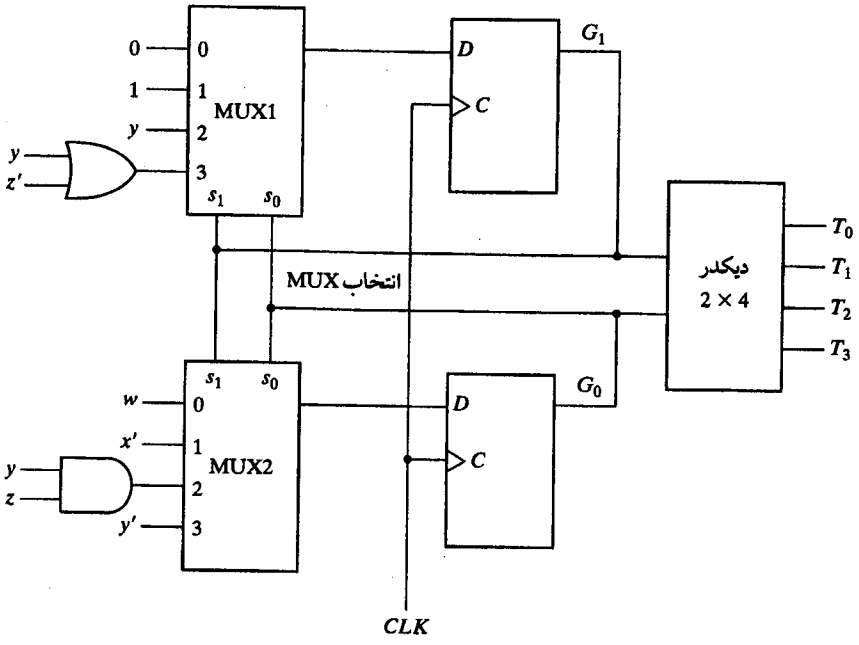
۸-۹ طراحی با مولتی پلکسر

کنترل ثبات و دیکدر از سه عنصر تشکیل شده بود: فلیپ فلاپ‌ها که مقدار حالت دودویی را نگه می‌دارند، دیکدر که خروجی‌های کنترل را فراهم می‌کند، و گیت‌ها که حالت بعدی و سیگنال‌های خروجی را تولید می‌نمایند. در بخش ۱۰-۴ نشان داده شده که یک مدار ترکیبی را می‌توان به جای گیت با مولتی پلکسر پیاده‌سازی کرد. از جایگزینی گیت‌ها با مولتی پلکسرها الگوی منظمی از سه طبقه عناصر حاصل می‌گردد. اولین سطح متشکل از مولتی پلکسرهایی است که حالت بعدی ثبات را معین می‌کنند. دومین سطح حاوی ثباتی است که حالت دودویی فعلی را نگه می‌دارد. سومین سطح دیکدری دارد که خروجی جداگانه‌ای را برای هر حالت کنترل فراهم می‌کند. این سه عنصر سلول‌های استاندارد از پیش تعریف شده در بسیاری از مدارهای مجتمع می‌باشد.

به عنوان مثال، چارت ASM شکل ۱۹-۸ را ملاحظه کنید. این چارت از چهار حالت و چهار ورودی کنترل تشکیل شده است. جعبه‌های حالت خالی در اینجا گذاشته شده‌اند زیرا ما فقط علاقمند به



شکل ۱۹-۸ . مثالی از چارت ASM با چهار ورودی کنترل



شکل ۲۰-۸. پیاده سازی کنترل با مولتی پلکسر

رشته کنترلی هستیم که مستقل از عملیات ثباتی است. تخصیص دودویی برای هر حالت در گوشه سمت راست بالای جعبه‌های حالت مشخص شده است. جعبه‌های تصمیم، گذرهای حالت را به صورت تابعی از چهار ورودی کنترل w، x، y و z مشخص می‌نمایند. پیاده‌سازی کنترل سه طبقه‌ای در شکل ۲۰-۸ نشان داده شده است. این مدار شامل مولتی پلکسرهای MUX 1، MUX 2 با یک ثبات متشکل از دو فلیپ فلاپ G_0 ، G_1 و یک دیکدر با چهار خروجی است. خروجی‌های ثبات به ورودی‌های دیکدر و نیز ورودی‌های انتخاب مولتی پلکسرها وصل شده است. به این ترتیب حالت فعلی ثبات برای انتخاب یکی از ورودی‌های هر مولتی پلکسر به کار برده می‌شود. آنگاه خروجی‌های مولتی پلکسرها به ورودی‌های D فلیپ فلاپ‌های G_1 و G_0 وصل می‌شود. هدف از کاربرد مولتی پلکسرها تهیه ورودی برای فلیپ فلاپ مربوطه‌اش می‌باشد به نحوی که برابر با مقدار دودویی حالت بعدی باشد. ورودی مولتی پلکسرها از جعبه‌های تصمیم‌گیری و گذر حالت موجود در چارت ASM حاصل می‌گردد. مثلاً حالت 00 بسته به مقدار ورودی w در 00 می‌ماند و یا به 01 می‌رود. چون حالت بعدی G_1 در هر حال 0 است یک 0 در ورودی شماره 0 مولتی پلکسر 1 قرار می‌دهیم. اگر $w = 0$ باشد حالت بعدی G_0 برابر 0 و اگر $Q = 1$ باشد $G_0 = 1$ خواهد بود. چون حالت بعدی G_0 برابر w است، ورودی کنترل w را به ورودی 0 از MUX 2 وصل می‌نماییم. معنی این کار این است که وقتی ورودی‌های انتخاب مولتی‌پلکسرها برابر حالت فعلی 00 است، خروجی مولتی پلکسرها مقداری را به دودویی تولید می‌کنند که در پالس ساعت بعدی به ثبات منتقل می‌گردد.

جدول ۸-۷ شرایط ورودی مولتی پلکسر

حالت فعلی		حالت بعدی		شرایط ورودی	ورودی ها	
G_1	G_0	G_1	G_0		MUX1	MUX2
0	0	0	0	w'		
0	0	0	1	w	0	w
0	1	1	0	x		
0	1	1	1	x'	1	x'
1	0	0	0	y'		
1	0	1	0	yz'	$yz' + yz = y$	yz
1	0	1	1	yz		
1	1	0	1	$y'z$		
1	1	1	0	y		
1	1	1	1	$y'z'$	$y + y'z' = y + z'$	$y'z + y'z' = y'$

برای ایجاد امکان ارزیابی ورودی‌های مولتی پلکسر جدولی را آماده می‌کنیم تا شرایط ورودی را برای هر گذر ممکن در چارت ASM نشان دهد. جدول ۷-۸ این اطلاعات را برای چارت ASM شکل ۱۹-۸ ارائه می‌کند. از حالت فعلی 00 یا 01 دو گذر و از حالت فعلی 10 یا 11 سه گذر یا انتقال وجود دارد. این حالات در جدول با خطی افقی از یکدیگر جدا شده‌اند. شرایط ورودی در جدول از جعبه‌های تصمیم در چارت ASM بدست آمده‌اند. مثلاً در شکل ۱۹-۸ می‌بینیم که اگر $x = 1$ باشد حالت فعلی 01 به 10 خواهد رفت و اگر $x = 0$ باشد حالت بعدی 11 است. در جدول این شرایط را به ترتیب با x و x' علامت می‌زنیم. دو ستون زیر "ورودی‌های مولتی پلکسر" در جدول مقادیر ورودی را که باید به MUX 1 و MUX 2 اعمال شوند مشخص می‌کنند. ورودی مولتی پلکسر برای هر حالت فعلی از شرایط ورودی وقتی که حالت بعدی فلیپ فلاپ برابر 1 است بدست می‌آید. بنابراین بعد از حالت فعلی 01، حالت بعدی G_1 همواره برابر با 1 و حالت بعدی G_2 برابر با متمم x است. بنابراین هنگام حالت فعلی 01 ثبات، ورودی MUX 1 به 1 و MUX 2 به x' وصل می‌شوند. به عنوان مثالی دیگر، پس از حالت فعلی 10، حالت بعدی G_1 باید برابر 1 باشد به شرطی که شرایط ورودی yz' و یا yz برابر قرار باشد. اگر این دو تابع بولی با یکدیگر OR شوند، همانطور که در جدول دیده می‌شود تنها متغیر y باقی خواهد ماند. اگر شرایط $yz = 11$ در ورودی برقرار باشد، حالت بعدی G_0 برابر 1 خواهد بود. اگر حالت بعدی G_1 پس از حالت مفروضی در 0 باقی بماند، ما یک 0 در ورودی مولتی پلکسر طبق آنچه در حالت فعلی 00 برای مولتی پلکسر MUX 1 نشان داده شده قرار می‌دهیم. همانطور که در حالت فعلی 00 برای MUX 2 دیده شد، اگر حالت بعدی G_1 همواره 1 باشد، در ورودی مولتی پلکسر، 1 قرار می‌دهیم. این کار برای حالت فعلی 01 برای MUX 1 نشان داده شده است. سایر عناصر جدول برای MUX 1 و MUX 2 به طور مشابه حاصل می‌شوند. آنگاه ورودی‌های مولتی پلکسرها از جدول، در پیاده‌سازی کنترل شکل ۲۰-۸ به کار می‌رود. توجه کنید که اگر حالت بعدی یک فلیپ فلاپ تابعی از دو یا چند متغیر باشد، ممکن است مولتی پلکسر به یک یا چند گیت در ورودی‌اش نیاز داشته باشد. در غیر این صورت ورودی مولتی پلکسر برابر با متغیر کنترل، یا متمم آن، یا 0 و یا 1 است.

مثال طراحی - شمارش تعداد 1 ها در یک ثبات

ما پیاده‌سازی یک کنترل مولتی پلکسری را با یک مثال نشان خواهیم داد. مثال، فرموله شدن چارت ASM و پیاده‌سازی زیرسیستم مسیر داده (پردازشگر) را نشان خواهد داد.

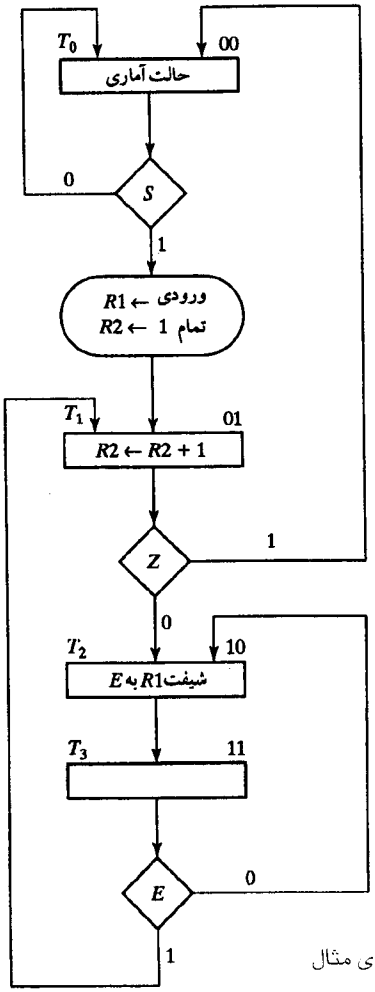
سیستم دیجیتالی که قرار است طراحی شود متشکل از دو ثبات R_1 و R_2 و یک فلیپ فلاپ E می‌باشد. سیستم تعداد 1های عدد بار شده در ثبات R_1 را می‌شمارد و ثبات R_2 را با تعداد شمارش شده پر می‌کند. به عنوان مثال، اگر عدد دودویی بار شده در ثبات R_1 برابر 10111001 باشد، مدار 5 عدد 1 را در R_1 شمرده و ثبات R_2 را با عدد دودویی 101 پر خواهد کرد. این عمل هر بار با جابجایی هر بیت از R_1 به فلیپ فلاپ E صورت می‌گیرد. مقدار E بعداً به وسیله کنترل چک شده و هر دفعه که برابر با 1 باشد ثبات R_2 یک واحد افزایش می‌یابد.

کنترل از یک ورودی بیرونی S برای شروع عملیات و دو ورودی وضعیت E و Z از پردازشگر استفاده می‌کند. E خروجی فلیپ فلاپ و Z خروجی مداری است که محتوای ثبات R_1 را برای یافتن حالت تمام 0 چک می‌کند. وقتی $R_1 = 0$ باشد خروجی $Z = 1$ تولید خواهد شد.

چارت ASM برای مدار در شکل ۲۱-۸ نشان داده شده است. عدد دودویی در R_1 بار می‌شود و R_2 با مقدار تمام 1 پر می‌شود. توجه کنید که عدد تمام 1 در یک ثبات وقتی افزایش یابد حالت تمام 0 را تولید می‌نماید. در حالت T_1 ، ثبات R_2 افزایش می‌یابد و محتوای R_1 مورد وارسی قرار می‌گیرد. اگر محتوا 0 باشد، آنگاه $Z = 1$ شده و نشان می‌دهد که در ثبات R_1 ، 1 ، وجود ندارد، بنابراین عملیات پایان یافته و محتوای R_2 برابر 0 خواهد بود. اگر محتوای R_1 غیرصفر باشد، آنگاه $Z = 0$ شده و به این معنی است که تعدادی 1 در ثبات ذخیره شده است. آنگاه عدد موجود در R_1 جابجا شده و سمت چپ‌ترین بیت آن به E منتقل می‌گردد. این عمل به دفعاتی که برای انتقال 1 به E لازم باشد تکرار می‌گردد. برای هر 1 یافت شده در E ثبات R_2 یک واحد افزایش یافته و ثبات R_1 برای یافتن 1های بیشتر وارسی می‌شود. حلقه بزرگتر آن قدر تکرار می‌گردد تا تمام 1های موجود در R_1 شمرده شوند. دقت کنید که جعبه حالت T_3 دارای عملیات ثباتی نیست، ولی بلوک متعلق به آن حاوی جعبه تصمیم برای E است. و نیز توجه کنید که ورودی سریال به شیفت رجیستر R_1 باید معادل 0 باشد زیرا نمی‌خواهیم 1های بیرونی وارد R_1 شوند.

نمودار بلوکی مدار در شکل ۲۲-۸ نشان داده شده است. کنترل سه ورودی و چهار خروجی دارد. فقط سه خروجی توسط پردازشگر یا مسیر داده استفاده شده است. ثبات R_1 یک شیفت رجیستر است. ثبات R_2 یک شمارنده با امکان بار شدن موازی می‌باشد. برای پرهیز از پیچیدگی نمودار ساعت نشان داده نشده، ولی باید به دو ثبات، فلیپ فلاپ E و فلیپ فلاپ‌های کنترل اعمال گردد.

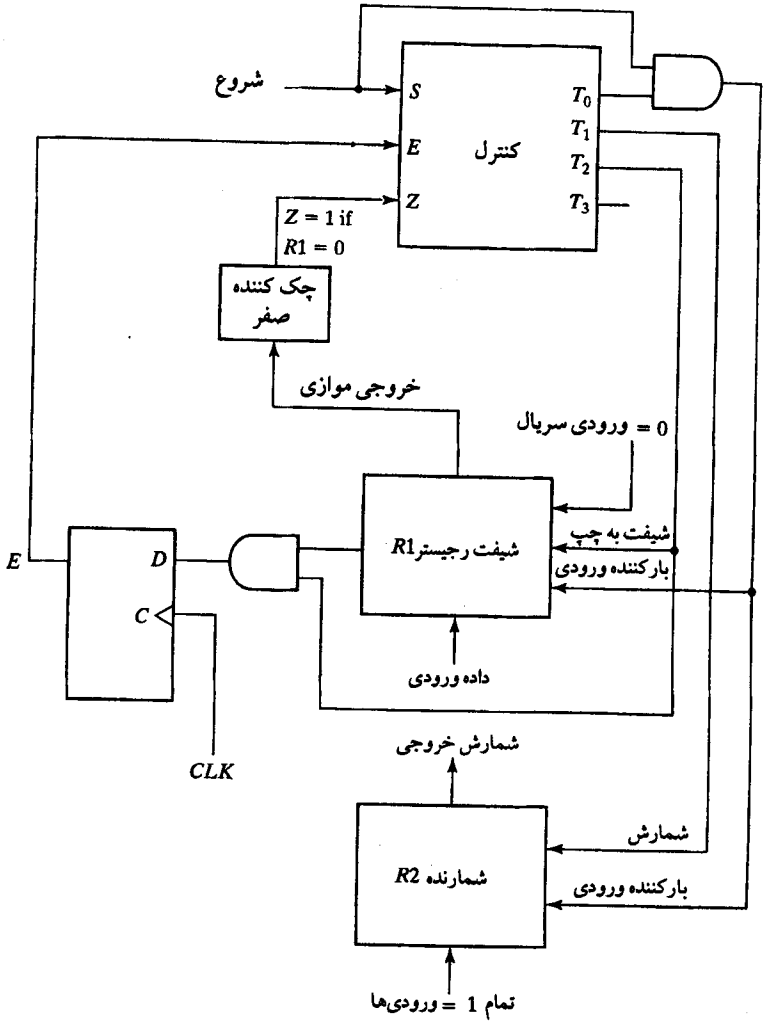
شرایط ورودی مولتی پلکسر برای کنترل از جدول ۸-۸ معین می‌گردد. این شرایط برای هر گذر حالت دودویی ممکن از جدول ASM فراهم می‌شود. به چهار حالت چهار مقدار 00 تا 11 تخصیص یافته است. گذر از حالت فعلی 00 به S وابسته است. گذر از 01 به Z و گذر از حالت فعلی 11 به E بستگی دارد. حالت فعلی 10 بدون شرط در پالس ساعت بعدی به 11 می‌رود. مقادیر زیر ستون‌های



شکل ۸-۲۱. چارت ASM برای مثال

جدول ۸-۸. شرایط ورودی مولتی پلکسر برای مثال طراحی

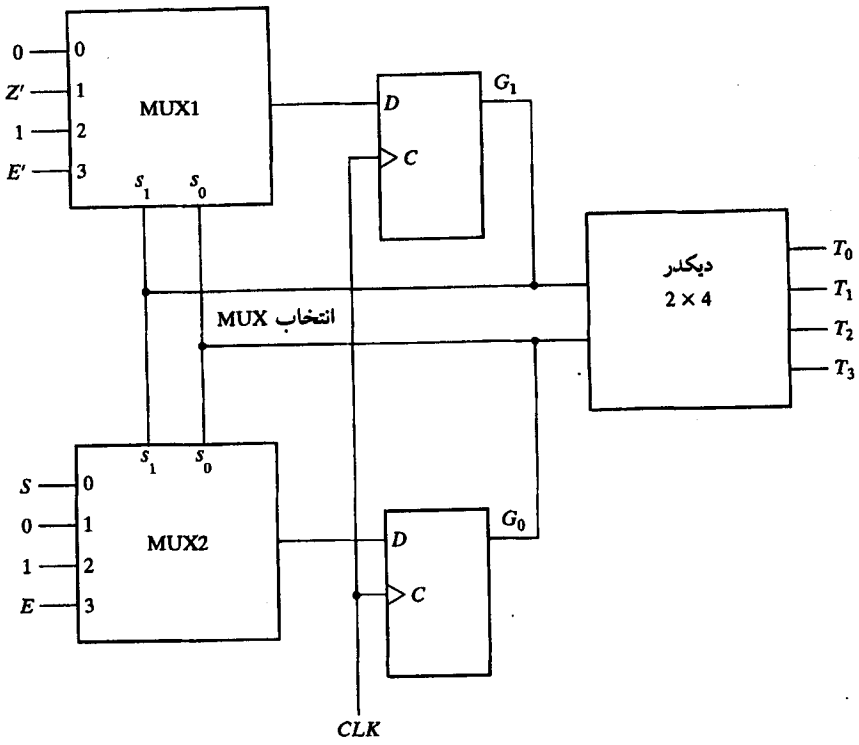
حالت فعلی		حالت بعدی		شرایط ورودی	ورودی‌های مولتی پلکسر	
G_1	G_0	G_1	G_0		MUX1	MUX2
0	0	0	0	S'		
0	0	0	1	S	0	S
0	1	1	0	Z		
0	1	1	1	Z'	Z'	0
1	0	1	1	None	1	1
1	1	1	0	E'		
1	1	0	1	E	E'	E



شکل ۲۲-۸. نمودار بلوکی شمارش 1ها

MUX 1 و MUX 2 در جدول به ترتیب از شرایط بولی ورودی برای حالت بعدی G_1 و G_0 فراهم می‌گردد.

پیاده سازی کنترل مثال طراحی در شکل ۲۳-۸ ملاحظه می‌شود. این مدار یک پیاده سازی سه سطحی است که در آن مولتی‌پلکسرها در سطح اول قرار گرفته‌اند. ورودی‌ها به مولتی پلکسرها از جدول۸-۸ بدست آمده‌اند.



شکل ۲۳-۸ . پیاده سازی کنترل در مثال طراحی

مسائل

۸-۱ به زبان خود عملیات انتقال ثباتی زیر را توصیف نمایید.

(الف) $R2 \leftarrow R2 + 1, R1 \leftarrow R2$

(ب) $R3 \leftarrow R3 - 1$

(پ) $\text{If } (T_1 = 1) \text{ then } (R0 \leftarrow R1) \text{ else if } (T_2 = 1) \text{ then } (R0 \leftarrow R2)$

۸-۲ بخش آغازین یک چارت ASM را رسم کنید. دو سیگنال کنترل x و y وجود دارد. اگر xy = 10 باشد، ثبات R یک واحد اضافه می‌شود و کنترل به حالت دوم می‌رود. اگر xy = 01، شد، ثبات R صفر شده و کنترل از حالت آغازین به حالت سوم می‌رود. در غیر این صورت، کنترل در حالت آغازین می‌ماند.

۸-۳ چارت‌های ASM را برای گذرهای حالت زیر بنویسید:

(الف) اگر x = 0 باشد، کنترل از T_1 به T_2 می‌رود؛ اگر x = 1 باشد یک حالت شرطی تولید کرده و از T_1 به T_2 می‌رود.

(ب) اگر $z = 1$ باشد، کنترل از T_1 به T_2 و سپس به T_3 می‌رود؛ اگر $x = 0$ باشد، کنترل از T_1 به T_3 می‌رود.

(پ) با شروع از T_1؛ آنگاه اگر $xy = 00$ باشد به T_2 برود؛ اگر $xy = 01$ باشد به T_3 برود؛ اگر $xy = 10$ باشد به T_3 برود؛ اگر $xy = 10$ باشد به T_1 برود؛ در غیر این صورت به T_3 برود.

۸-۴ هشت مسیر خروجی از بلوک ASM که از جعبه‌های تصمیم‌گیری بیرون می‌آیند را رسم کنید تا هشت حالت دودویی ممکن برای سه متغیر x و y و z را چک نماید.

۸-۵ تفاوت یک چارت ASM را از چارت معمولی بیان کنید. این تفاوت را با به کارگیری شکل ۵-۸ نشان دهید.

۸-۶ برای یک سیستم دیجیتال یک چارت ASM بنا کنید تا تعداد افراد داخل یک اطاق را بشمارد. افراد از یک درب وارد می‌شوند و هر وقت نور فتوسلی به وسیله فردی قطع شود خروجی آن تغییری از 1 به 0 در سیگنال x ایجاد می‌نماید. افراد از درب دومی اطاق را ترک می‌کنند و خروجی فتوسل مشابهی، y، از 1 به 0 تغییر می‌یابد. مدار شامل یک بالا- پایین شمار است و نمایشگر آن تعداد افراد داخل اطاق را نشان می‌دهد.

۸-۷ یک چارت ASM برای یک مدار با دو ثبات 8 بیت RA و RB که دو عدد دودویی بی‌علامت را دریافت می‌کنند و تفریق زیر را انجام می‌دهند رسم نمایید.

$$RA \leftarrow RA - RB$$

از روش بخش ۵-۱ برای تفریق استفاده کنید و اگر جواب منفی بود یک فلیپ فلاپ قرض را به 1 بنشانید.

۸-۸ یک مدرا دیجیتال با سه ثبات 16 بیتی AR، BR و CR طراحی کنید تا عملیات زیر را انجام دهد.

(الف) دو عدد علامت‌دار 16 بیتی (به متمم 2) را به AR و BR منتقل نماید.

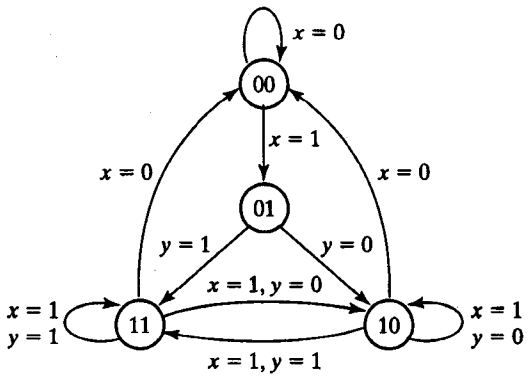
(ب) اگر عدد درون AR منفی است، عدد موجود در AR را بر 2 تقسیم کرده و نتیجه را به CR ببرید.

(پ) اگر عدد در AR مثبت ولی غیرصفر است، عدد درون BR را در 2 ضرب و نتیجه را به CR منتقل کنید.

(ت) اگر عدد درون AR صفر است ثبات CR را صفر نمایید.

۸-۹ کنترل نمودار شکل ۱۱-۸(الف) را با استفاده از یک فلیپ فلاپ بر هر حالت (تخصیص 1 بارز) طراحی کنید.

۸-۱۰ نمودار حالت واحد کنترلی در شکل (م ۱۰-۸) نشان داده شده است. این نمودار چهار حالت و دو ورودی x و y دارد. چارت ASM معادل آن را رسم نمایید و درون جعبه‌های حالت را خالی کنید.



شکل (م ۱۰-۸). نمودار حالت کنترل برای مسائل ۱۰-۸ و ۱۱-۸

۸-۱۱ کنترلی طراحی کنیدکه نمودار حالتش شکل (م ۸-۱۰) باشد. از فلیپ فلاپ D استفاده نمایید.

۸-۱۲ فرض کنید R_1 در شکل ۸-۲۲ یک شیفت رجیستر 4 بیت مطابق شکل ۷-۶ باشد. نشان دهید که چگونه ورودی‌های جابجایی و بار شدن به ورودی‌های S_1 و S_0 شیفت رجیستر وصل می‌شوند.

۸-۱۳ یک شمارنده 4 بیت با ورودی همزمان پاک شدن که در شکل ۸-۱۰ نشان داده شده است طراحی نمایید.

۸-۱۴ یک مدار دیجیتال که دو عدد دودویی را با روش جمع مکرّر در هم ضرب کند طراحی نمایید. مثلاً برای ضرب 5×4، سیستم ضرب را با چهار بار جمع مضروب انجام می‌دهد: یعنی $5 + 5 + 5 + 5 = 20$. اکنون تصور کنید که مضروب در BR، مضروب فیه در AR و حاصلضرب در ثبات PR باشد. یک مدار جمع کننده محتوای BR را با PR جمع می‌کند. یک مدار تشخیص صفر، پس از هر بار کاهش AR، Z را چک می‌کند.

۸-۱۵ ثابت کنید که ضرب دو عدد n بیتی حاصلضربی با طول کمتر یا برابر 2n بیت تولید می‌کند.

۸-۱۶ در شکل ۸-۱۳، ثبات Q مضروب فیه و B مضروب را نگه می‌دارد. هر عدد را 16 بیتی تصور کنید.

(الف) در حاصلضرب، چند بیت انتظار دارید و در کجا در دسترس است؟

(ب) شمارنده P چند بیت است، و عدد اولیه بار شده در آن چند است.

(پ) مداری طراحی کنید که صفر موجود در P را شناسایی نماید.

۸-۱۷ محتوای ثبات‌های A، C و P را مشابه جدول ۸-۴ در حین عمل ضرب برای دو عدد 1111 (مضروب) و 10101 (مضروب فیه)، لیست کنید.

۸-۱۸ زمان عمل ضرب دودویی بخش ۸-۶ را بدست آورید. فرض کنید که ثبات Q دارای n بیت بوده و سیکل ساعت هم t نانو ثانیه است.

۸-۱۹ مدار کنترل یک ضرب کننده دودویی با نمودار حالت شکل ۸-۱۵ را طراحی کنید. از مولتی پلکسر، دیکدر و یک ثبات استفاده نمایید.

۸-۲۰ چارت ASM شکل (م ۸-۲۰) را ملاحظه کنید. عملیات ثباتی مشخص نشده است زیرا ما فقط به طراحی مدار کنترل علاقمندیم.

(الف) نمودار حالت معادل را رسم نمایید.

(ب) کنترل را با یک فلیپ فلاپ در هر حالت طراحی کنید.

(پ) جدول حالت را برای کنترل لیست نمایید.

(ت) کنترل را با سه فلیپ فلاپ D، یک دیکدر و گیت طراحی کنید.

(ث) جدولی را بدست آورید که شرایط ورودی مولتی پلکسر را برای کنترل نشان دهد.

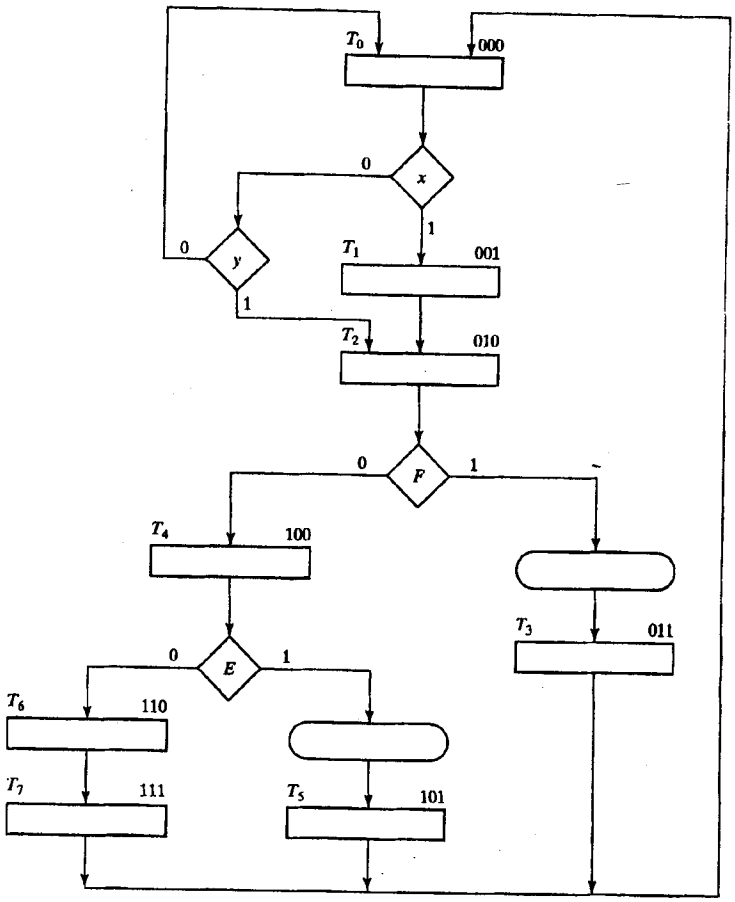
(ج) کنترل را با سه مولتی پلکسر، یک ثبات با سه فلیپ فلاپ و یک دیکدر 8×3 طراحی کنید.

۸-۲۱ در هر بلوک HDL مقدار E چقدر است. فرض کنید RA = 1 است.

(الف)
```
RA = RA - 1;
if (RA == 0) E = 1;
else E = 0;
```

(ب)
```
RA <= RA - 1;
if (RA == 0) E <= 1;
else E <= 0;
```

۸-۲۲ با به کارگیری عملگرهای Verilog HDL در جدول ۸-۱، عملیات زیر را ارزیابی کنید. فرض شود



شکل (م ۲۰-۸)

$A = 4'b0110$ ، $B = 4'b0010$ و $C = 4'b0000$ باشد.

```
A * B; A + B; A - B; ~C; A & B; A | B; A ^ B; & A; ~|C; A || B;
A && C; |A; A < B; A > B; A != B;
```

۲۳-۸ بلوک always زیر را ملاحظه کنید.

```
always @ (posedge CLK)
if (T1) R1 <= R1 + R2;
else if (T2) R1 <= R1 + 1;
else R1 <= R1;
```

با استفاده از یک شمارنده 4 بیت با بار شدن موازی برای R1 (طبق شکل ۱۴-۶) و یک جمع کننده 4 بیت، نمودار بلوکی نشان دهنده اتصالات اجزاء و سیگنال‌های کنترل را برای یک طراحی، رسم نمایید.

۲۴-۸ یک عبارت case چند سطحی اغلب به وسیله یک طراح منطقی به صورت مولتی پلکسرهای سخت‌افزاری برگردانده می‌شود. بلوک case زیر را چگونه به سخت‌افزار برمی‌گردانید؟ هر ثبات را 8 بیت تصور کنید.

```
case (state)
T0:  R4 = R0;
T1:  R4 = R1;
T2:  R4 = R2;
T3:  R4 = R3;
endcase
```

۸-۲۵ مداری طراحی کنید که تعداد 1 ها را در یک ثبات همچون بخش ۸-۹ بشمارد. چارت ASM برای این مدار در شکل ۸-۲۱ و نمودار بلوکی در شکل ۸-۲۲ دیده می‌شود.

(الف) توصیف HDL سطح انتقال ثباتی مدار را بنویسید.

(ب) مدار کنترل را با استفاده از یک فلیپ فلاپ در هر حالت (تخصیص 1 بارز) طراحی کنید. معادلات ورودی را برای هر چهار فلیپ فلاپ بنویسید.

(پ) توصیف ساختاری HDL مدار را با استفاده از کنترل طراحی شده در بخش (ب) و نمودار بلوکی شکل ۸-۲۲ بنویسید.

(ت) یک برنامه تست برای تست مدار بنویسید. برای اطمینان از صحت عملکرد مدار در هر دو برنامه ساختاری و RTL، آن را شبیه‌سازی کنید.

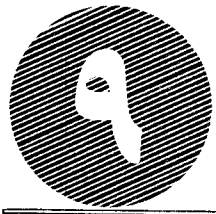
۸-۲۶ توصیف ساختاری HDL ضرب کننده طراحی شده در بخش ۸-۶ را بنویسید. نمودار بلوکی شکل ۸-۱۳ و مدار کنترل شکل ۸-۱۷ را به کار ببرید. طرح را شبیه‌سازی کنید و عملکرد آن را با استفاده از برنامه تست مثال ۸-۶ چک کنید.

۸-۲۷ جمع دو عدد دودویی علامت‌دار در نمایش مقدار - علامت از قوانین حساب معمولی تبعیت می‌کند. اگر هر دو عدد علامت یکسانی داشته باشند، دو مقدار با هم جمع می‌شوند و علامت هم یکی از علائم است. اگر دو عدد علامت مخالف داشته باشند مقدار کوچکتر از بزرگتر کسر می‌شود و نتیجه با علامت عدد بزرگتر خواهد بود. سمت چپ‌ترین بیت علامت و 7 بیت دیگر مقدار را نگاه می‌دارند.

۸-۲۸ توصیف HDL مدار مسئله ۸-۱۴ را بنویسید.

مراجع

1. PALNITKAR, S. 1996. *Verilog HDL: A Guide to Digital Design and Synthesis.* SunSoft Press (A Prentice Hall Title).
2. BHASKER, J. 1997. *A Verilog HDL Primer.* Allentown, PA: Star Galaxy Press.
3. BHASKER, J. 1998. *Verilog HDL Synthesis.* Allentown, PA: Star Galaxy Press.
4. THOMAS, D. E., and P. R. MOORBY. 1998. *The Verilog Hardware Description Language* 4th ed. Boston: Kluwer Academic Publishers.
5. CILETTI, M. D. 1999. *Modeling, Synthesis, and Rapid Prototyping with Verilog HDL.* Upper Saddle River, NJ: Prentice Hall.
6. ARNOLD, M. G. 1999. *Verilog Digital Computer Design.* Upper Saddle River, NJ: Prentice Hall.
7. SMITH, D. J. 1996. *HDL Chip Design.* Madison, AL: Doone Publications.



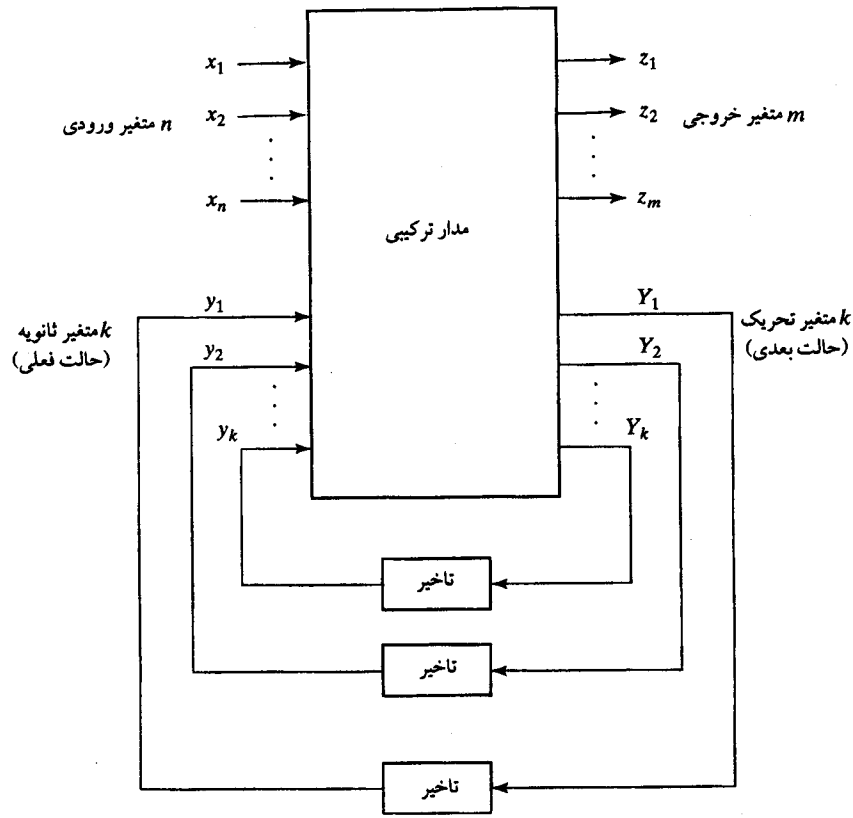
مدارهای ترتیبی غیرهمزمان

۹-۱ مقدمه

یک مدار ترتیبی با دنباله‌ای از ورودی‌ها، خروجی‌های وابسته به زمان و حالات داخلی‌اش مشخص می‌شود. در مدارهای ترتیبی همزمان (سنکرون) تغییر حالت داخلی به وسیلهٔ اعمال پالس‌های ساعت زمانبندی شده صورت می‌گیرد. مدارهای ترتیبی غیرهمزمان (آسنکرون) از پالس‌های ساعت استفاده نمی‌کنند. در این گونه مدارها تغییر حالت به هنگام تغییر در ورودی‌ها رخ می‌دهد. عناصر حافظه در مدارهای ترتیبی همزمان، فلیپ فلاپ‌های ساعت‌دار می‌باشند. عناصر حافظه در مدارهای ترتیبی غیرهمزمان، فلیپ فلاپ‌های بدون ساعت و یا عناصر تأخیر زمانی هستند. توانایی حفظ اطلاعات در یک قطعه تولیدکننده تأخیر زمانی از این حقیقت ناشی می‌شود که این قطعه برای انتشار یک سیگنال از داخل گیت‌های دیجیتال، زمان محدودی را نیاز دارد. یک مدار ترتیبی غیرهمزمان در بسیاری از موارد شبیه به یک مدار ترکیبی پسخوردی (فیدبک) عمل می‌کند.

به دلیل مسائل و مشکلات زمانی مربوط به مسیر پسخورد، طراحی مدارهای ترتیبی غیرهمزمان، بسیار مشکل‌تر از طراحی مدارهای همزمان است. در یک سیستم همزمان که به درستی طراحی شده باشد مشکلات زمانی با راه‌اندازی همزمان (تریگر) کلیه فلیپ‌فلاپ‌ها با لبه پالس از بین می‌رود. تغییر از یک حالت و رفتن به حالت بعدی، طی زمان کوتاه تغییر وضعیت پالس، که همان گذر پالس می‌باشد، صورت می‌گیرد. نظر به این که مدارهای غیرهمزمان از پالس ساعت استفاده نمی‌کنند، تغییر حالت سیستم بلافاصله پس از هر تغییر در ورودی رخ می‌دهد. در این گونه سیستم‌ها باید مراقب بود که با توجه به وجود پسخورد، هر حالت جدید، مدار را در یک وضعیت پایدار نگهدارد.

مدارهای ترتیبی غیرهمزمان در کاربردهای متنوعی مورد استفاده قرار می‌گیرند. از این مدارها هنگامی استفاده می‌شود که سرعت عملیات مدّنظر است، و به ویژه در حالت‌هایی که سیستم دیجیتال



شکل ۱-۹ . نمودار بلوکی یک مدار ترتیبی همزمان

باید به سرعت و بد ون انتظار برای پالس ساعت عکس‌العمل نشان دهد، این مدارها مورد توجه‌انـد. مدارهای غیرهمزمان در سیستم‌های کوچکی که فقط نیاز به تعداد کمی قطعه دارند و هزینه تهیه یک مدار جداگانه برای تولید پالس ساعت مقرون به صرفه نـیست بسـیار اقـتصادی‌انـد. ایـن مـدارهـا در کاربردهایی که سیگنال‌های ورودی به سیستم ممکن است در هر لحظه و مستقل از یک ساعت داخلی تغییر کنند، مفید می‌باشند. اتصال دو واحد که هر یک دارای پالس ساعت مستقل خودش است باید به وسیله مدارهای غیرهمزمان صورت گیرد. طراحان دستگاه‌های دیجیتال اغلب یک سیستم مختلط تولید می‌کنند به نحوی که بعضی از قسمتهای دستگاه همزمان دارای ویژگی‌های یک مدار غیرهمزمان است. آگاهی از عملکرد یک مدار ترتیبی غیرهمزمان می‌تواند در اطمینان از صحت عملکرد آن مفید باشد.

شکل ۱-۹ نمودار بلوکی یک مدار ترتیبی غیرهمزمان را نشان می‌دهد. این مدار شامل یک بخش ترکیبی و عناصر تأخیری برای تشکیل حلقه‌های تأخیر است. مدار دارای n مـتغیر ورودی، m مـتغیر خروجی، و k حالت داخلی است. عناصر تأخیر زمانی را در مدارهـای تـرتیبی مـی‌توان بـه صـورت حافظه‌های کوتاه‌مدت تجسم کرد. در یک مدار گیتی، تأخیر انتشار موجود در مسیر ورودی به خروجی

به اندازه کافی تأخیر در حلقه پسخورد تولید می‌کند. بنابراین هیچ عنصر تأخیر زمانی در مسیر پسخورد لحاظ نمی‌شود. معمولاً متغیرهای حالت فعلی و حالت بعدی در مدارهای ترتیبی غیرهمزمان به ترتیب، متغیرهای ثانوی و متغیرهای تحریک نامیده می‌شوند. متغیرهای تحریک را نباید با متغیرهای جدول تحریک به کار رفته در طراحی مدارهای ترتیبی ساعت دار اشتباه کرد.

وقتی که مقدار یک متغیر ورودی تغییر می‌کند، متغیرهای ثانوی y به طور آنی تغییر نمی‌کنند. انتشار سیگنال از ورودی و از طریق مدار ترکیبی به متغیرهای تحریک Y برای تولید حالت بعدی جدید، به مقدار معینی زمان نیاز دارد. این مقادیر از طریق عناصر تأخیر انتشار می‌یابند تا تبدیل به حالت فعلی جدیدی برای متغیرهای ثانوی شوند. به اختلاف بین yها و Yها توجه کنید. در حالت پایدار، آنها یکی هستند، ولی در حین گذر یکسان نمی‌شوند. برای مقدار مفروضی از متغیرهای ورودی، اگر مدار مذکور به یک حالت با ثبات یا پایدار برسد به نحوی که $y_i = Y_i$ با $i = 1, 2, ..., k$ باشد، سیستم موردنظر پایدار است، در غیر این صورت به طور مداوم در حال گذر پیوسته‌ای بوده و ناپایدار خواهد بود. در مدارهای غیرهمزمان درک این مطلب که گذر از یک حالت پایدار به حالتی دیگر فقط در پاسخ به یک تغییر در متغیرهای ورودی است حائز اهمیت می‌باشد. برعکس در مدارهای همزمان انتقال حالت در پاسخ به اعمال یک پالس ساعت صورت می‌گیرد.

به منظور اطمینان از یک عملکرد صحیح، باید اجازه داد تا مدار ترتیبی غیرهمزمان قبل از هر تغییر جدید در ورودی‌هایش، به یک حالت با ثبات و پایداری برسد. به دلیل وجود تأخیر در سیم‌ها و گیت‌های مدار، داشتن دو تغییر همزمان و یا بیشتر غیرممکن است و حتی به این که کدام اول رخ داده است اطمینان چندانی نیست. بنابراین تصور تغییر همزمان دو یا چند متغیر ورودی منع می‌شود. این قید به این معنی است که در هر لحظه از زمان فقط یک متغیر ورودی مجاز به تغییر است و زمان بین دو تغییر هم باید از زمان رسیدن مدار به یک حالت پایدار طولانی‌تر باشد. این نوع عملیات به عنوان عملیات اساسی خوانده می‌شوند. در عملیات اساسی فرض بر این است که سیگنال‌های ورودی یک به یک و هنگام پایداری مدار تغییر می‌کنند.

۲-۹ روش تحلیل

تحلیل مدارهای ترتیبی غیرهمزمان به معنی تهیه جدول یا نموداری است که رشته حالات داخلی و خروجی‌ها را به صورت تابعی از تغییرات متغیرهای ورودی توصیف نماید. اگر یک مدار ترتیبی غیرهمزمان دارای یک یا چند حلقه پسخورد باشد و یا از فلیپ فلاپ‌هایی که به پالس ساعت حساس نیستند تشکیل شده باشد نمودار منطقی آن را نشان خواهد داد. در این بخش رفتار مدارهای ترتیبی غیرهمزمانی که دارای مسیرهای پسخورد هستند و از فلیپ‌فلاپ‌ها استفاده نمی‌کنند را مورد بررسی قرار می‌دهیم. به فلیپ فلاپ‌های بدون ساعت لچ هم می‌گویند و استفاده از آنها در مدارهای ترتیبی غیرهمزمان در بخش بعد تشریح خواهد شد.

روش تحلیل همراه با سه مثال خاص ارائه می‌شود. مثال اول جدول گذر را معرفی می‌کند. دومین

مثال جدول روند را معرفی می‌نماید. در مثال سوم ثُبات مدارهای ترتیبی غیرهمزمان مورد تفحص قرار می‌گیرد.

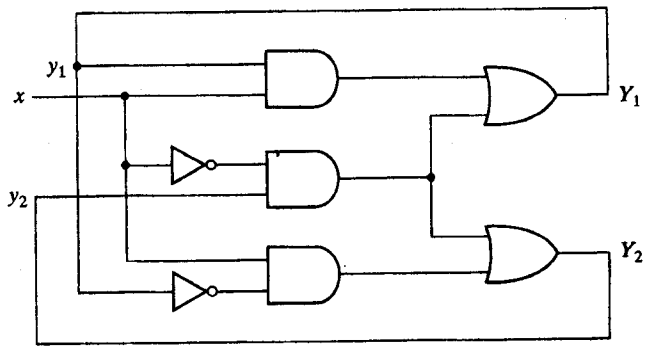
جدول کذر

مثالی از یک مدار ترتیبی غیرهمزمان که فقط با گیت‌ها ساخته شده باشد در شکل ۲-۹ دیده می‌شود. این نمودار به وضوح دو حلقه پسخورد را از خروجی گیت‌های OR به ورودی گیت‌های AND نشان می‌دهد. مدار شامل یک متغیر ورودی x و دو حالت داخلی است. حالات داخلی شامل دو متغیر تحریک Y_1 و Y_2 و دو متغیر ثانویه y_1 و y_2 می‌باشند. تأخیر مربوط به هر حلقه پسخورد از تأخیر انتشار بین هر ورودی y و خروجی Y مربوط به آن بدست می‌آید. هر گیت منطقی در مسیر، تأخیر انتشاری حدود 2 تا 10ns را تولید می‌کند. سیم‌هایی که سیگنال‌های الکترونیکی را هدایت می‌کنند تقریباً یک نانو ثانیه تأخیر را در هر فوت ایجاد می‌نمایند. بنابراین وقتی که مدار ترکیبی و سیم‌های مسیر پسخورد تأخیر کافی را به وجود می‌آورند، نیازی به عناصر اضافی تولید کننده تأخیر نیست.

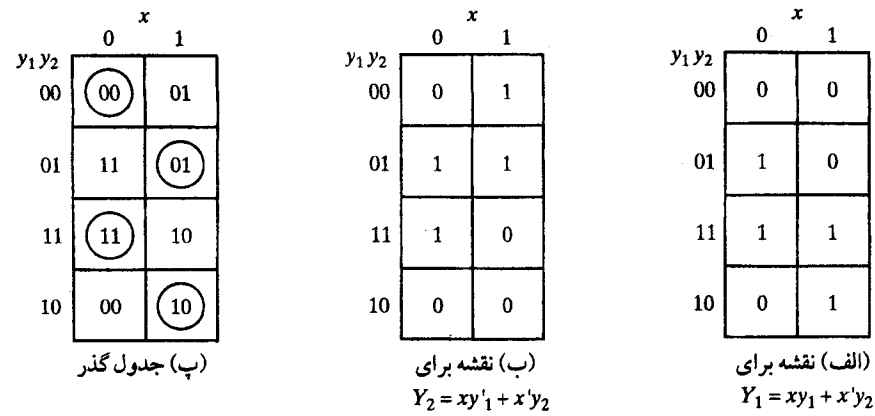
تحلیل یک مدار با ملاحظه متغیرهای تحریک به عنوان خروجی‌ها و متغیرهای ثانویه به عنوان ورودی‌ها شروع می‌شود. آنگاه عبارات بولی را برای متغیرهای تحریک به صورت تابعی از ورودی‌ها و متغیرهای ثانویه بدست می‌آوریم. این عبارات به راحتی از نمودار منطقی بدست می‌آیند.

$$Y_1 = xy_1 + x'y_2$$
$$Y_2 = xy_1' + x'y_2$$

قدم بعد ترسیم توابع Y_1 و Y_2، طبق شکل ۳-۹ (الف) و (ب) در یک نقشه است. مقادیر دودویی انکد شده متغیرهای y برای نام‌گذاری سطرها و متغیر ورودی x برای ستون‌ها به کار رفته‌اند. این آرایش با کمی اختلاف از نقشه سه متغیره‌ای که در فصل‌های قبل به کار رفت، نتیجه می‌شود. با این وجود، علیرغم اعتبار نقشه، این نوع آرایش هنگام بررسی مدارهای ترتیبی غیرهمزمان مناسب‌تر است. توجه کنید که متغیرهای متعلق به مربع‌های مربوطه در امتداد اضلاع نقشه، آن طور که در فصل‌های قبل دیدیم، ذکر نشده‌اند.



شکل ۲-۹ . مثالی از مدار ترکیبی غیرهمزمان



شکل ۳-۹. نقشه‌ها و جدول گذر مدار شکل ۲-۹

جدول گذر شکل ۳-۹ (پ) با ترکیب مقادیر دودویی مربع‌های مربوطه، در نقشه‌های فوق حاصل می‌شود. این جدول در داخل هر مربع مقدار $Y = Y_1Y_2$ را نشان می‌دهد. اولین بیت Y از مقدار Y_1 و دومین بیت از Y_2 در همان مربع بدست می‌آید. برای داشتن یک حالت پایدار، مقدار Y باید برابر با $y = y_1y_2$ باشد. وارده‌هایی از جدول که در آنها $Y = y$ است با دایره محصور شده‌اند تا حالات پایدار مشخص شوند. وارده‌هایی که با دایره محصور نشده‌اند، حالت‌های ناپایدارند.

اکنون به تأثیر یک تغییر در متغیر ورودی توجه کنید. مربع مربوط به $x = 0$ و $y = 00$ در جدول گذر نشان می‌دهد که $Y = 00$ است. چون Y حالت بعدی y را نشان می‌دهد، این حالت یک حالت پایدار است. اگر هنگام $y = 00$، x از 0 به 1 تغییر کند، مدار مقدار Y را به 01 تغییر خواهد داد. این وضعیت یک حالت موقت ناپایدار را نشان می‌دهد زیرا Y با مقدار فعلی y برابر نیست. آنچه که بعد از این اتفاق می‌افتد این است که به محض انتشار سیگنال برای تولید $Y = 01$، مسیر پسخورد موجب می‌شود تا $y = 01$ شود. این مطلب با یک گذر یا انتقال از اولین سطر $(y = 00)$ به سطر دوم که در آن $y = 01$ است، نشان داده می‌شود. اکنون $Y = y$ است و مدار با $x = 1$ به یک حالت پایدار رسیده است. به طور کلی اگر یک ورودی مدار را به حالت ناپایداری بکشاند، ضمن ثابت ماندن x، مقدار y آنقدر تغییر می‌کند تا به یک حالت باثبات (محصور در دایره) برسد. با استفاده از این تحلیل برای بقیه مربعات جدول گذر، در می‌یابیم که مدار شبیه حالات 00، 01، 11، 10 را وقتی x متناوباً 0 و 1 شود تولید خواهد کرد.

به اختلاف بین یک مدار ترتیبی همزمان و غیرهمزمان توجه نمایید. در یک سیستم همزمان حالت فعلی کلاً با مقادیر فیلپ فلاپ‌ها مشخص می‌شود و اگر ساعت غیرفعال باشد در ازاء تغییر ورودی، این حالت تغییری نمی‌کند. در یک مدار غیرهمزمان، حالت داخلی بلافاصله پس از تغییر در ورودی، عوض می‌شود. به این علت گاهی بهتر است حالت داخلی را با مقدار ورودی ترکیب کرده و آن را حالت کلی مدار بنامیم. مداری که جدول گذر آن در شکل ۳-۹ (پ) نشان داده شده است چهار حالت کلی پایدار $y_1y_2x = 000$، 011، 110 و 101 و چهار حالت ناپایدار 001، 010، 111 و 100 را داراست.

جدول ۱-۹. جدول حالات مدار شکل ۲-۹

حالت فعلی		حالت بعدی			
		$x = 0$		$x = 1$	
0	0	0	0	0	1
0	1	1	1	0	1
1	0	0	0	1	0
1	1	1	1	1	0

جدول گذر مدارهای ترتیبی غیرهمزمان مشابه با جدول حالت در مدارهای ترتیبی همزمان است. اگر متغیرهای ثانویه را به عنوان حالت فعلی و متغیرهای تحریک را حالت بعدی تصور کنیم، جدول حالت ۱-۹ را بدست خواهیم آورد. این جدول اطلاعات مشابهی را همچون جدول گذر فراهم می‌کند. در حالت غیرهمزمان یک محدودیتی وجود دارد که در حالت همزمان دیده نمی‌شود. در جدول گذر غیرهمزمان، معمولاً حداقل یک وارده حالت بعدی مثل مقدار حالت فعلی در هر سطر است. در غیر این صورت همه حالات کلی در آن سطر بی‌ثبات خواهند بود.

روش تهیه جدول گذر از نمودار یک مدار ترتیبی غیرهمزمان به طریق زیر است:

۱- همه حلقه‌های پسخورد را در مدار معین کنید.

۲- خروجی هر حلقه را با متغیر Y_i و ورودی مربوطه‌اش را با y_i مشخص نمایید به طوری که $i = 1, 2, ..., k$ باشد که در آن k تعداد حلقه‌های پسخورد در مدار است.

۳- توابع بولی همه Yها را به عنوان تابعی از ورودی‌های بیرونی و yها بدست آورید.

۴- با استفاده از متغیرهای y برای سطرها و ورودی‌های بیرونی برای ستون‌ها، هر تابع Y را در یک نقشه ترسیم کنید.

۵- همه نقشه‌ها را در یک جدول با هم ترکیب نمایید به نحوی که مقدار هر $Y = Y_1Y_2, ..., Y_k$ را در هر مربع نشان دهد.

۶- مقداری از Y در هر مربع که با مقدار $y = y_1y_2, ..., y_k$ در همان ردیف برابر است رادر دایره محصور کنید.

پس از تکمیل جدول گذر، رفتار مدار با مشاهده گذر حالت به عنوان تابعی از تغییرات در متغیرهای ورودی قابل تحلیل است.

جدول روند

در حین طراحی مدارهای ترتیبی غیرهمزمان، بهتر است بدون ذکر مقادیر دودویی، حالت‌ها را با سمبل‌های حرفی نامگذاری کنیم. چنین جدولی را جدول روند نامند. جدول روند مشابه با جدول گذر است با این تفاوت که حالات داخلی به جای اعداد دودویی با حروف نشان داده شده‌اند. جدول روند شامل مقادیر خروجی مدار برای هر حالت پایدار نیز هست.

	x = 0	x = 1
a	(a)	b
b	c	(b)
c	(c)	d
d	a	(d)

(الف) چهار حالت با یک ورودی

	x_1x_2 = 00	01	11	10
a	(a), 0	(a), 0	(a), 0	b, 0
b	a, 0	a, 0	(b), 1	(b), 0

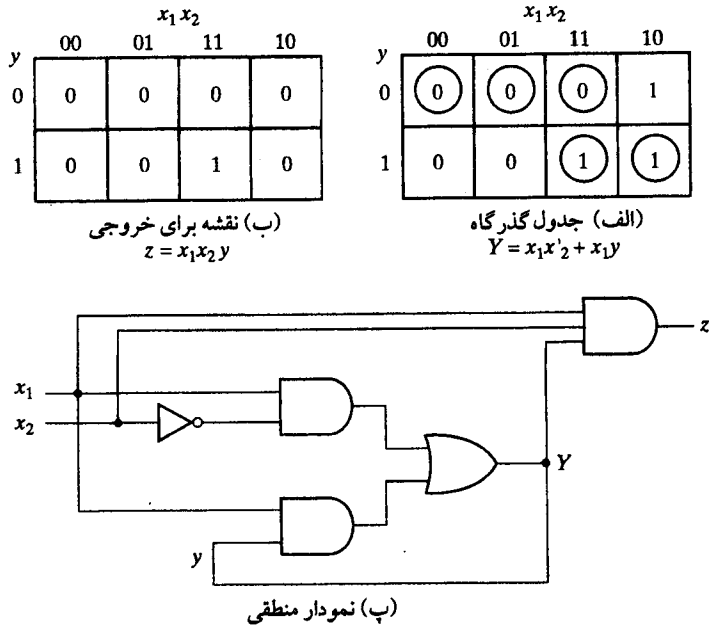
(ب) دو حالت با دو ورودی و یک خروجی

شکل ۴-۹ . مثالهایی از جدولهای روند

چند مثال از جدول روند در شکل ۴-۹ ملاحظه می‌شود. در شکل ۴-۹ (الف) چهار حالت a، b، c و d را ملاحظه می‌کنید. اگر مقادیر $a = 00$، $b = 01$، $c = 11$ و $d = 10$ را به حالات آن تخصیص دهیم شکل ۳-۹ (پ) نتیجه خواهد شد. جدول شکل ۴-۹ (الف) را جدول روند اصلی می‌نامند زیرا در هر سطر فقط دارای یک حالت پایدار می‌باشد. شکل ۴-۹ (پ) جدول روندی را نشان می‌دهد که در همان سطر بیش از یک حالت پایدار را داراست. این جدول دو حالت a و b، دو ورودی x و y و یک خروجی z دارد. مقدار دودویی متغیر خروجی در داخل مربع و مجاور سمبل حالت مدار مشخص شده و با یک ویرگول تفکیک شده است. از روی جدول روند، رفتار زیر قابل استنباط است. اگر $x_1 = 0$ باشد، مدار در حالت a است. اگر x_1 به 1 برود و x_2 در 0 باشد، مدار به حالت b می‌رود. با $x_1x_2 = 11$، مدار ممکن است در حالت a یا حالت b باشد. اگر در حالت a قرار داشته باشد، خروجی 0 و اگر در حالت b باشد، خروجی 1 است. اگر ورودی‌ها از 10 به 11 تغییر کنند حالت b حفظ می‌شود. اگر ورودی‌ها از 01 به 11 بروند مدار در حالت a باقی خواهد ماند. به خاطر دارید که در حالت اساسی، دو متغیر ورودی به طور همزمان نمی‌توانند تغییر کنند و بنابراین، اجازه تغییر 00 به 11 را نخواهیم داد.

برای بدست آوردن مدار توصیف شده با یک جدول روند، لازم است به هر حالت مقدار دودویی منحصر به فردی نسبت دهیم. این تخصیص، جدول روند را به جدول گذر تبدیل خواهد کرد و از آن می‌توان نمودار منطقی را بدست آورد. این کار برای جدول روند شکل ۴-۹ (ب) در شکل ۵-۹ نشان داده شده است، 0 دودویی را به حالت a، 1 دودویی را به حالت b نسبت می‌دهیم. نتیجه این کار جدول گذر شکل ۵-۹ (الف) خواهد بود. نقشه خروجی شکل ۵-۹ (ب) مستقیماً از مقادیر خروجی در جدول روند فراهم شده است. تابع تحریک Y و تابع خروجی z به وسیله دو نقشه ساده شده‌اند. نمودار منطقی هم در شکل ۵-۹ (پ) ملاحظه می‌گردد.

این مثال روال تهیه نمودار منطقی را از یک جدول روند نشان می‌دهد. البته نباید فراموش کرد که روال کار همیشه به این سادگی نیست. در تخصیص مقادیر حالت و خروجی‌های حالات ناپایدار مشکلات متعددی وجود دارد. این مشکلات به تفصیل در بخش‌هایی که به دنبال می‌آیند بحث شده‌اند.

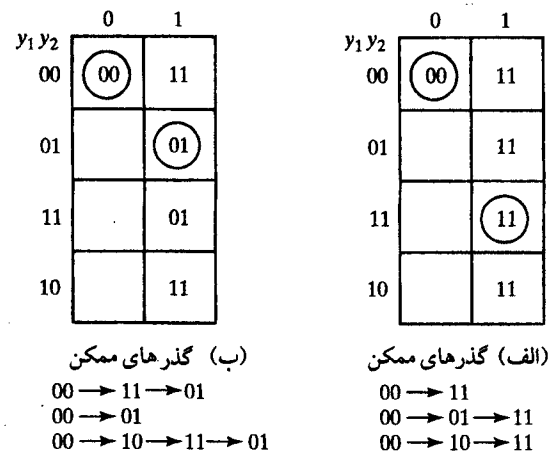


شکل ۵-۹. تعیین مدار مشخص شده بوسیله جدول روند شکل (۴-۹ ب)

وضعیت‌های رقابتی

در یک مدار ترتیبی غیرهمزمان اگر دو یا چند متغیر حالت دودویی در پاسخ به یک متغیر در یک ورودی تغییر کنند، یک وضعیت رقابتی وجود خواهد داشت. هنگامی که مدار با تأخیرهای نامساوی مواجه شود، وضعیت رقابتی موجب می‌گردد تا متغیرهای حالت به طور ناخواسته‌ای تغییر کنند. مثلاً اگر قرار باشد متغیرهای حالت از 00 به 11 تغییر نمایند، تفاوت تأخیرها ممکن است سبب شود تا متغیر اول زودتر از دومی تغییر کند و در نتیجه متغیرهای حالت از 00 به 10 و سپس به 11 بروند. اگر متغیر حالت دوم سریع‌تر از اولی تغییر نماید متغیرهای حالت از 00 به 01 و سپس به 11 تغییر خواهند نمود. بنابراین می‌توان نحوه تغییر متغیرهای حالت را از قبل پیش‌بینی کرد. اگر حالت پایدار نهایی که مدار به آن می‌رسد، مستقل از ترتیب متغیرهای حالت باشد، رقابت مذکور را رقابت غیربحرانی نامند. ولی اگر حالات پایدار نهایی دو یا چند حالت متفاوت باشند و این تفاوت به ترتیب تغییر متغیرهای حالت وابسته باشد، در این صورت رقابت بحرانی خواهد بود. اگر عملیات صحیحی از مدار موردنظر باشد باید از رقابت‌های بحرانی پرهیز شود.

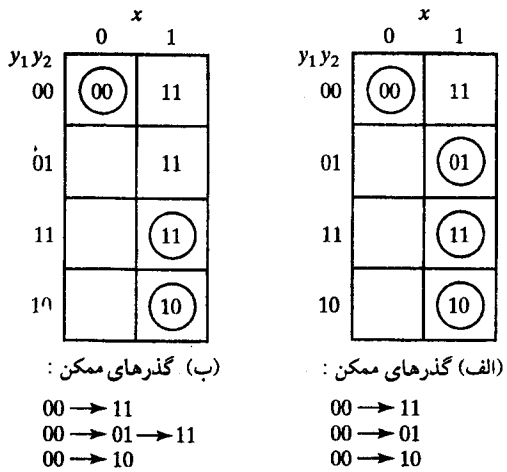
دو مثال شکل ۶-۹ رقابت‌های غیربحرانی را تشریح می‌کنند. ما با حالت پایدار کل $y_1y_2\ x = 000$ آغاز می‌کنیم و سپس ورودی را از 0 به 1 تغییر می‌دهیم. متغیرهای حالت باید از 00 به 11 تغییر کنند که گویای وجود یک وضعیت رقابتی است. گذرهای لیست شده در زیر هر جدول سه امکان تغییر



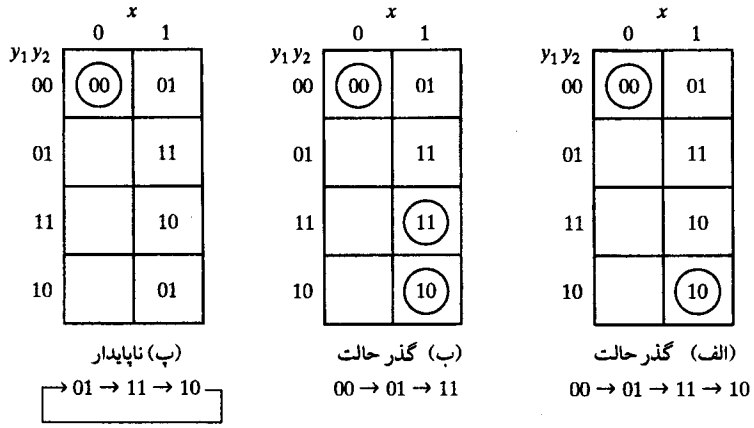
شکل ۶-۹ . مثالهایی از رقابتهای غیربحرانی

متغیرهای حالت را نشان می‌دهند. این گذرها یا انتقال‌ها می‌توانند به طور همزمان از 00 به 11 بروند و یا به ترتیبی از 00 به 10 و سپس به 11 بروند. در هر صورت، حالت پایدار نهایی یکسان و از نوع وضعیت رقابتی غیربحرانی است. در (الف) حالت کلی نهایی y_1y_2 x = 111 و در (ب) برابر با 011 می‌باشد.

جدول گذر شکل ۷-۹ رقابت‌های بحرانی را نشان می‌دهند. ما در اینجا هم با حالت کلی y_1y_2 x = 000 آغاز می‌کنیم و سپس ورودی را از 0 به 1 تغییر خواهیم داد. متغیرهای حالت باید از 00 به 11 تغییر نمایند. اگر آنها به طور همزمان عوض شوند، حالت پایدار کل نهایی 111 خواهد بود. در جدول گذر (الف)، اگر به دلیل تأخیر انتشار نابرابر Y_2 قبل از Y_1 عوض شود، آنگاه مدار به حالت پایدار کل 011 رفته و در آن باقی می‌ماند. از طرف دیگر، اگر ابتدا Y_1 تغییر کند حالت داخلی 10 شده و مدار در حالت پایدار کل 101 باقی خواهد ماند. از این رو، رقابت بحرانی است زیرا مدار، بسته به ترتیب تغییر



شکل ۷-۹ . مثالهایی از رقابتهای بحرانی



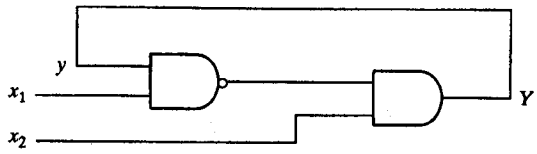
شکل ۸-۹ . مثالهای از سیکل

متغیرهای حالت به حالات پایدار متفاوتی می‌رود. جدول گذر شکل ۷-۹ (ب) یک نمونه رقابت بحرانی دیگری را نشان می‌دهد، که در آن دو گذر ممکن یک حالت پایدار کلی را نتیجه می‌دهند، ولی گذر سوم به حالت کلی دیگری می‌رود. با انتساب صحیح دودویی به متغیرهای حالت می‌توان از وضعیت رقابتی پرهیز کرد. اعداد دودویی باید طوری به متغیرهای حالت تخصیص یابند که در هنگام یک گذر در جدول حالت هر بار تنها یک متغیر حالت عوض شود. موضوع تخصیص حالت بدون رقابت در بخش ۶-۹ مورد بحث قرار گرفته است.

با هدایت مدار به حالات بی‌ثبات میانی و با یک تغییر حالت منحصر می‌توان از رقابت اجتناب کرد. وقتی که مدار از میان یک رشته حالات بی‌ثبات عبور می‌کند، گوییم مدار دارای چرخه یا سیکل است. شکل ۸-۹ وقوع چرخه‌ها را نشان می‌دهد. در اینجا هم با $y_1y_2 = 00$ آغاز می‌کنیم و سپس ورودی را از 0 به 1 تغییر می‌دهیم. جدول گذر بخش (الف) دنباله منحصر به فردی را نشان می‌دهد که به حالت کلی 101 ختم می‌گردد. جدول (پ) نشان می‌دهد که هر چند متغیرهای حالت از 00 به 11 تغییر می‌کنند، ولی گذر منحصر به فردی را از 00 به 01 و سپس به 11 فراهم می‌نمایند. وقتی که از چرخه برای یک مدار استفاده می‌شود باید دقت کرد تا به یک حالت پایدار منتهی گردد. اگر یک چرخه به یک حالت پایدار ختم نشود، مدار از یک حالت ناپایدار به حالت ناپایدار دیگری رفته و کل سیستم ناپایدار خواهد بود. این مطلب در شکل ۸-۹ (پ) و نیز مثال زیر نشان داده شده است.

ملاحظات مربوط به پایداری

به دلیل وجود پسخورد موجود در مدارهای ترتیبی غیرهمزمان، باید مراقب بود تا مدار ناپایدار نشود. یک وضعیت ناپایدار موجب می‌گردد تا مدار بین حالات ناپایدار نوسان کند. روش تحلیل جدول گذر می‌تواند در تشخیص وقوع بی‌ثباتی مفید باشد.



(الف) نمودار منطقی

y \ x_1x_2	00	01	11	10
0	(0)	1	1	(0)
1	0	(1)	0	0

(ب) جدول گذر

شکل ۹-۹. مثالی از مدار ناپایدار

به عنوان مثال مدار شکل ۹-۹ (الف) را ملاحظه کنید. تابع تحریک برابر است با

$$Y = (x_1y)'x_2 = (x_1' + y')x_2 = x_1'x_2 + x_2y'$$

جدول گذر برای مدار در شکل ۹-۹ (ب) نشان داده شده است. مقادیری از Y که با y برابر است در دایره محصور شده و حالات پایدار را نشان می‌دهند. وارده‌های محصور نشده وضعیت‌های ناپایدارند. توجه کنید که ستون 11 دارای حالت پایداری نیست. این بدان معنی است که با تثبیت x_1x_2 در 11، مقادیر Y و y هرگز یکسان نمی‌شوند. اگر $y = 0$ باشد آنگاه $Y = 1$ می‌شود، که این خود نیز موجب گذر به سطر دوم جدول با $y = 1$ و $Y = 0$ خواهد شد. این وضعیت مجدداً بازگشت به سطر اول را سبب می‌گردد. در نتیجه تا وقتی که ورودی 11 باشد متغیر حالت به میزان نامتناهی بین 0 و 1 نوسان خواهد کرد.

وضعیت ناپایدار را می‌توان مستقیماً از نمودار منطقی شناسایی کرد. اجازه بدهید تا $x_1 = 1$، $x_2 = 1$ و $y = 1$ باشد. خروجی گیت NAND برابر 0، خروجی گیت AND برابر 0 و بنابراین $Y = 0$ خواهد شد و در نتیجه $Y \neq y$ است. اگر $y = 0$ باشد، خروجی گیت NAND برابر 1 و خروجی گیت AND برابر 1 می‌گردد و بنابراین $Y \neq y$ خواهد شد. اگر فرض کنیم که هر گیت دارای تأخیر در انتشار 5ns باشد (همراه با تأخیر سیم‌ها) در می‌یابیم که Y در 10ns برابر 0 و در 10ns بعد در 1 قرار خواهد داشت. این وضعیت باعث تولید یک موج مربعی با پریود 20ns خواهد شد. فرکانس نوسان عکس پریود بوده و برابر 50MHz می‌گردد. ناپایداری در مدارهای ترتیبی غیرهمزمان نامطلوب است و باید از آن پرهیز کرد، مگر این که مدار به منظور تولید موج مربعی طراحی شود.

۹-۳ مدارهای لچ‌دار

به لحاظ تسلسل، مدارهای ترتیبی غیرهمزمان قبل از مدارهای همزمان شناخته شده و مورد استفاده بوده‌اند. اولین مدار دیجیتال کاربردی با رله‌ها که با عملیات غیرهمزمان سازگارترند، ساخته شد. به این

دلیل روش سنتی آرایش مدار غیرهمزمان با قطعاتی انجام می‌شد تا به هم متصل و یک یا چند مدار پسخورد را به وجود آورند. هنگامی که مدارهای دیجیتال با اجزاء الکترونیک ساخته می‌شوند، استفاده از لچ SR (بخش ۲-۵) به عنوان عنصر حافظه آسان‌تر است. به کارگیری لچ‌های SR در مدارهای ترتیبی الگوی منظمی در نمودارهای منطقی ایجاد می‌نماید که در آن بخش حافظه کلاً قابل رویت است. در این بخش عملکرد لچ SR با تکنیک بخش قبل تحلیل خواهد شد. آنگاه روالی را برای پیاده‌سازی مدارهای ترتیبی غیرهمزمان با لچ SR نشان خواهیم داد.

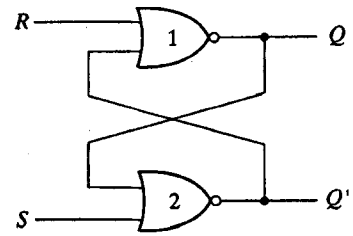
لچ SR

لچ SR مداری دیجیتال با دو ورودی S و R و دو گیت NOR متقاطع یا دو گیت NAND متقاطع است. مدار ساخته شده با گیت NOR در شکل ۱۰-۹ دیده می‌شود. این مدار و جدول درستی‌اش از شکل ۳-۵ بدست آمده است. به منظور تحلیل مدار با استفاده از روش جدول گذر، آن را دوباره در شکل ۱۰-۹ (پ) ترسیم کرده‌ایم تا مسیر پسخورد از خروجی گیت 1 به ورودی گیت 2 ملاحظه شود. خروجی Q معادل با متغیر تحریک Y و متغیر ثانویه y است. تابع بول خروجی برابر است با

$$Y = [(S + y)' + R]' = (S + y)R' = SR' + R'y$$

با تهیه جدول Y طبق شکل ۱۰-۹ (ت) جدول گذر را برای مدار بدست خواهیم آورد.

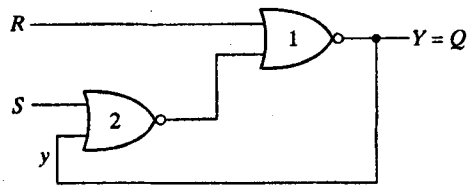
اکنون می‌توانیم رفتار لچ SR را با استفاده از جدول گذر بررسی کنیم. با SR = 10، خروجی



(الف) مدار با کوپل متقاطع

S	R	Q	Q'	
1	0	1	0	
0	0	1	0	(پس از SR = 10)
0	1	0	1	
0	0	0	1	(پس از SR = 01)
1	1	0	0	

(ب) جدول درستی



(پ) مدار با نمایش فیدبک

y \ SR	00	01	11	10
0	(0)	(0)	(0)	1
1	(1)	0	0	(1)

$Y = SR' + R'y$
$Y = S + R'y$ when $SR = 0$

(ت) جدول گذر

شکل ۱۰-۹. لچ SR با گیتهای NOR

$Q = Y = 1$ خواهد شد و در این حال گوییم لچ به 1 نشانده شده است. اگر S را به 0 بازگردانیم مدار در همان حال باقی می‌ماند. با $SR = 01$، خروجی $Q = Y = 0$ شده و در این حال گوییم لچ به 0 بازنشانی شده است. با تغییر R به 0 مدار در همان حالت بازنشانی باقی خواهد ماند. این حالات در جدول درستی هم لیست شده‌اند. وقتی که S و R هر دو در 1 باشند مدار مشکل خواهد داشت. با توجه به جدول درستی ملاحظه می‌شود که هر دو خروجی Q و Q' برابر 0اند. این حالت قانون متمم بودن این دو خروجی را زیر سؤال می‌برد. به علاوه با توجه به جدول گذر می‌بینیم که رفتن از $SR = 11$ به $SR = 00$ نتیجه غیرقابل پیش‌بینی را تولید می‌کند. اگر S ابتدا به 0 برود، خروجی در 0 باقی می‌ماند ولی اگر R زودتر به 0 برود خروجی به 1 خواهد رفت. در یک عملکرد عادی باید مطمئن باشیم که مقدار 1 به طور همزمان به S و R اعمال نمی‌شود. این شرط را می‌توانیم با تابع بولی $SR = 0$ نشان دهیم که بیان می‌دارد AND (ضرب منطقی) S و R همیشه باید 0 باشد.

اگر به تابع تحریک بازگردیم، در می‌یابیم که اگر عبارت بولی SR' و SR را OR کنیم نتیجه تک متغیر S خواهد بود.

$$SR' + SR = S(R' + R) = S$$

از این رابطه نتیجه می‌شود که اگر $SR = 0$ باشد $S = SR'$ خواهد بود. بنابراین تابع تحریک قبلی حاصل

$$Y = SR' + R'y$$

را می‌توان به صورت زیر بیان کرد

$$Y = S + R'y \qquad \text{وقتی } SR = 0$$

برای تحلیل یک مدار با لچ SR، ابتدا باید شرط بولی $SR = 0$ را چک کنیم. آنگاه تابع تحریک کاهش یافته را برای تحلیل مدار به کار خواهیم برد. با این وجود، اگر S و R را به طور همزمان برابر 1 بیابیم، آنگاه لازم است از همان تابع تحریک اولیه و ساده نشده استفاده کنیم.

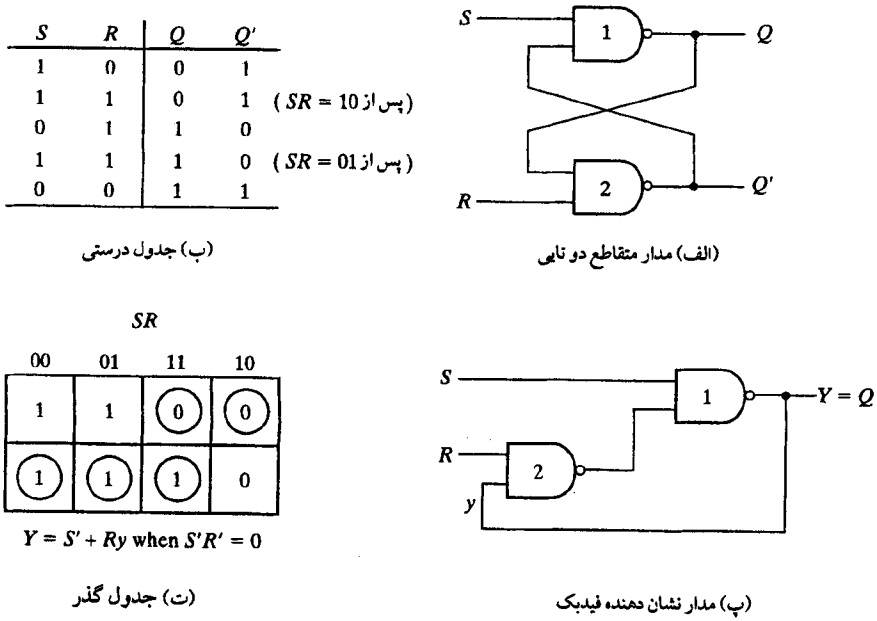
تحلیل لچ SR با گیت‌های NAND در شکل ۱۱-۹ انجام شده است. ورودی‌های لچ SR به طور معمول در 1 هستند، مگر این که حالت لچ را بخواهیم عوض کنیم. اِعمال 0 به R موجب می‌شود تا Q به 0 برود و به این ترتیب لچ در حالت بازنشانی قرار خواهد گرفت. پس از بازگشت R به 1، تغییر S به 0 سبب نشاندن لچ خواهد شد. حالتی که در اینجا باید از آن پرهیز کرد 0 شدن همزمان S و R است. ممانعت از این حالت با $S'R' = 0$ میسر است. تابع تحریک برای مدار عبارت است از

$$Y = [S(Ry)']' = S' + Ry$$

از مقایسه این رابطه با تابع تحریک لچ NOR ملاحظه می‌کنیم که S با S' و R' با R جایگزین شده است. از این رو متغیر ورودی برای لچ NAND، متمم ورودی‌های لچ NOR می‌باشند. به این دلیل لچ NAND را گاهی لچ $S'R'$ هم می‌خوانند (یا لچ $\bar{S}-\bar{R}$).

مثال تحلیل

مدارهای ترتیبی غیرهمزمان را با لچ‌های SR، با یا بدون مسیرهای پسخورد بیرونی هم می‌توان



S	R	Q	Q′	
1	0	0	1	
1	1	0	1	(پس از SR = 10)
0	1	1	0	
1	1	1	0	(پس از SR = 01)
0	0	1	1	

(ب) جدول درستی

(الف) مدار متقاطع دو تایی

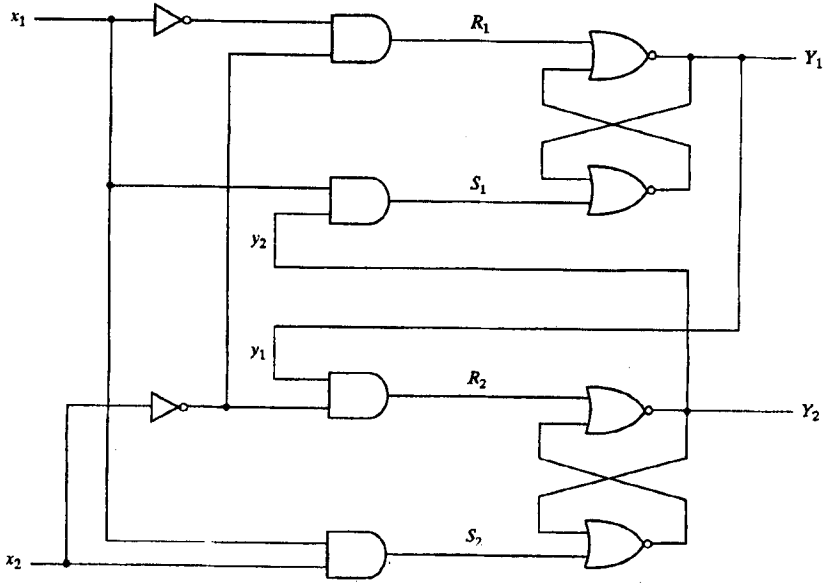
(ت) جدول گذر

(پ) مدار نشان دهنده فیدبک

شکل ۱۱-۹ . لچ SR با گیتهای NAND

ساخت. البته، همیشه مسیر حلقه پسخوردی در خود لچ وجود دارد. تحلیل مداری که با لچ ساخته شده باشد با مثالی نشان داده خواهد شد. با توجه به این مثال، می‌توان روال را تعمیم داده قدم‌های مؤثر بعدی را برای طراحی دیگر مدارهای مشابه برداشت.

مدار شکل ۱۲-۹ دارای دو لچ SR با خروجی‌های Y_1 و Y_2 است. دو ورودی x_1 و x_2 و دو حلقه



شکل ۱۲-۹ . مثالی از یک مدار با لچهای SR

پسخورد بیرونی وجود دارند که متغیرهای ثانویه y_1، y_2 از آنها اخذ شده‌اند. توجه دارید که این مدار شبیه به یک مدار ترتیبی معمولی است که در آن لچ‌ها همچون فیلپ فلاپ‌های بدون ساعت عمل می‌کنند. برای تحلیل مدار، ابتدا توابع بول را برای ورودی‌های S و R هر لچ بدست می‌آوریم.

$$S_1 = x_1y_2 \qquad S_2 = x_1x_2$$
$$R_1 = x_1'x_2' \qquad R_2 = x_2'y_1$$

آنگاه شرط $SR = 0$ را برای اطمینان از عملکرد مدار بررسی می‌کنیم:

$$S_1R_1 = x_1y_2x_1'x_2' = 0$$
$$S_2R_2 = x_1x_2x_2'y_1 = 0$$

نتیجه 0 است زیرا $x_1x'_1 = x_2x'_2 = 0$ می‌باشد.

قدم بعدی بدست آوردن جدول گذر مدار است. به یاد دارید که جدول گذر، مقدار Y را به صورت تابعی از y و x بدست می‌دهد. توابع تحریک از رابطه $Y = S + R'y$ حاصل می‌گردند.

$$Y_1 = S_1 + R_1'y_1 = x_1y_2 + (x_1 + x_2)y_1 = x_1y_2 + x_1y_1 + x_2y_1$$
$$Y_2 = S_2 + R_2'y_2 = x_1x_2 + (x_2 + y_1')y_2 = x_1x_2 + x_2y_2 + y_1'y_2$$

اکنون یک نقشه مرکب برای $Y = Y_1Y_2$ بدست می‌آوریم. همانطور که در شکل ۱۳-۹ ملاحظه می‌شود متغیرهای y به سطرهای نقشه و متغیر x به ستون‌ها تخصیص یافته‌اند. توابع بول Y_1 و Y_2 که در فوق بیان شدند برای ترسیم نقشه مرکب استفاده می‌شوند. در هر سطر وارده‌هایی از Y که دارای مقداری برابر y هستند در دایره محصور شده و حالات پایداری را نمایش می‌دهند. با بررسی جدول گذر، نتیجه می‌گیریم که مدار با ثبات است. اگر مدار در آغاز در $y_1y_2\ x_1x_2 = 1101$ باشد و x_2 از 1 به 0 تغییر کند، یک وضعیت رقابت بحرانی به وجود خواهد آمد. اگر Y_1 قبل از Y_2 به 0 برود، مدار به جای 0000 به 0100 خواهد رفت. با این وجود، با تأخیرهای تقریباً مساوی در گیت‌ها و لچ‌ها، به نظر نمی‌رسد که چنین وضعیت نامطلوبی رخ دهد.

روال تحلیل یک مدار ترتیبی غیرهمزمان با لچ‌های SR را می‌توان به صورت زیر خلاصه کرد.

y_1y_2 \ x_1x_2	00	01	11	10
00	(00)	(00)	01	(00)
01	(01)	(01)	11	11
11	00	(11)	(11)	10
10	00	(10)	11	(10)

شکل ۱۳-۹. جدول گذر مدار (۱۲-۹)

۱- خروجی هر لچ را با Y_i و مسیر فیدبک خارجی‌اش را (اگر وجود داشته باشد) با y_i مشخص کنید که در آن $i = 1, 2, ..., k$ می‌باشد.
۲- توابع بولی را برای ورودی‌های S_i و R_i هر لچ بدست آورید.
۳- برای هر لچ NOR، $SR = 0$ و برای هر لچ NAND، $S'R' = 0$ را چک کنید. اگر این شرایط صادق نبودند، ممکن است مدار به درستی کار نکند.
۴- رابطه $Y = S + R'y$ را برای هر لچ NOR، یا $Y = S' + Ry$ را برای هر لچ NAND ارزیابی کنید.
۵- نقشه‌ای برای y‌ها در سطر و ورودی‌های x در ستون بسازید.
۶- مقدار $Y = Y_1Y_2, ..., Y_k$ را در نقشه پیاده کنید.
۷- همه حالات با ثبات را که در آن $Y = y$ است در دایره محصور نمایید. نقشه حاصل همان جدول گذر است.

جدول تحریک لچ

جدول گذر لچ SR برای تحلیل و تعریف عملکرد لچ بسیار مفید است. این جدول وقتی متغیر ثانویه y و ورودی‌های S و R معلومند متغیر Y را مشخص می‌کند. در حین روند پیاده‌سازی، جدول گذر مدار معلوم می‌باشد و ما علاقمند به یافتن مقادیر S و R هستیم. به این دلیل به جدولی نیاز داریم تا ورودی‌های S و R را برای هر گذر ممکن از y به Y لیست نماید. چنین سیستمی را **جدول تحریک** گوییم.

جدول تحریک لچ SR در شکل ۱۴-۹ (ب) ملاحظه می‌شود. دو ستون اول چهار گذر ممکن از y به Y را نشان می‌دهد. دو ستون بعدی مقادیر ورودی لازمی را مشخص می‌کنند که گذر را موجب می‌شوند. مثلاً برای گذر از $y = 0$ به $Y = 1$، لازم است ورودی $S = 1$ و ورودی $R = 0$ باشد. این مطلب در سطر دوم جدول گذر نشان داده شده است.

شرایط ورودی لازم برای هر چهار گذر در جدول تحریک را مستقیماً از جدول گذر لچ شکل ۱۴-۹ (ت) و پس از حذف حالت ناپایدار $SR = 11$ می‌توان بدست آورد. مثلاً، برای تغییر از $y = 0$ به $Y = 0$، جدول گذر نشان می‌دهد که SR می‌تواند 00 یا 01 باشد. این بدان معنی است که باید $S = 1$ ولی R می‌تواند 0 یا 1 باشد. بنابراین اولین سطر در جدول تحریک نشان می‌دهد که $S = 0$ و $R = X$ است که X حالت بی‌اهمیت بوده و می‌تواند 0 یا 1 باشد.

مثال پیاده‌سازی

پیاده‌سازی یک مدار ترتیبی با لچ‌های SR روشی برای تهیه نمودار منطقی از یک جدول گذر می‌باشد. در این روش لازم است تا تابع بول را برای ورودی‌های S و R هر لچ معین نماییم. سپس نمودار منطقی با رسم لچ‌های SR و گیت‌های منطقی که توابع S و R را پیاده‌سازی می‌کنند، بدست می‌آید. برای تشریح این روش، پیاده‌سازی مثال شکل (۵-۹) را تکرار خواهیم کرد. مدار خروجی تغییر نمی‌کند و بنابراین دوباره تکرار نمی‌شود.

y	Y	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	1

(ب) جدول تحریک لچ

x_1x_2

y	00	01	11	10
0	(0)	(0)	(0)	1
1	0	0	(1)	(1)

(الف) جدول گذر

$Y = x_1x'_2 + x_1y$

x_1x_2

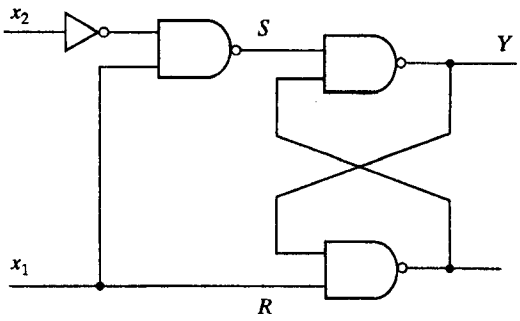
y	00	01	11	10
0	X	X	X	0
1	1	1	0	0

(ت) نقشه برای $R = x'_1$

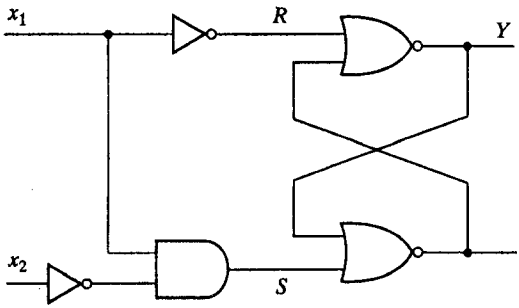
x_1x_2

y	00	01	11	10
0	0	0	0	1
1	0	0	X	X

(پ) نقشه برای $S = x_1x'_2$



(ج) مدار لچ با NAND



(ث) مدار لچ با NOR

شکل ۱۴-۹ . بدست آوردن مداری با لچ، از یک جدول انتقال

جدول گذر شکل ۵-۹ (الف) مجدداً در شکل ۱۴-۹ (الف) آمده است. با توجه به اطلاعات مفروض در جدول گذر و شرایط موجود در جدول تحریک لچ در شکل ۱۴-۹ (پ)، می‌توانیم نقشه‌های ورودی‌های S و R لچ را طبق شکل ۱۴-۹ (پ) و (ت) بدست آوریم. مثلاً، مربع سطر دوم و ستون سوم ($yx_1x_2 = 111$) در شکل ۱۴-۹ (الف) به گذر $y = 1$ به $Y = 1$ نیاز دارد. جدول تحریک برای این تغییر $S = X$ و $R = 0$ را مشخص می‌کند. بنابراین مربع مربوطه در نقشه S با X و در نقشه R با 0 علامت زده شده است. دیگر مربع‌ها همگی به همین روش با مقادیر مربوطه پر شده‌اند. آنگاه نقشه‌ها برای بدست آوردن توابع بول ساده شده به کار می‌روند.

$$S = x_1x'_2 \qquad \text{و} \qquad R = x'_1$$

نمودار منطقی از یک لچ SR و گیت‌های مربوطه برای پیاده‌سازی توابع بولی S و R تشکیل شده است. وقتی که لچ NOR به کار رود مدار مطابق شکل ۱۴-۹ (ث) است. با لچ NAND باید مقادیر متمم شده را برای S و R به کار ببریم.

$$S = (x_1 x_2')' \qquad \text{و} \qquad R = x_1$$

این مدار در شکل ۱۴-۹ (ج) نشان داده شده است.

اکنون می‌توان روش کلی پیاده‌سازی مدار با لچ‌های SR با استفاده از یک جدول گذر را به صورت زیر خلاصه کرد.

۱- با فرض داشتن جدول گذر که تابع تحریک $Y = Y_1 Y_2, \ldots, Y_k$ را مشخص کند، جفت نقشه‌هایی برای S_i و R_i با $i = 1, 2, \ldots, k$ بدست آورید. این کار با استفاده از شرایط مشخص شده در جدول تحریک شکل ۱۴-۹ (پ) انجام می‌شود.

۲- توابع بول ساده شده را برای هر S_i و R_i بدست آورید. دقت کنید که S_i و R_i در مربع‌های متناظر برابر 1 نشوند.

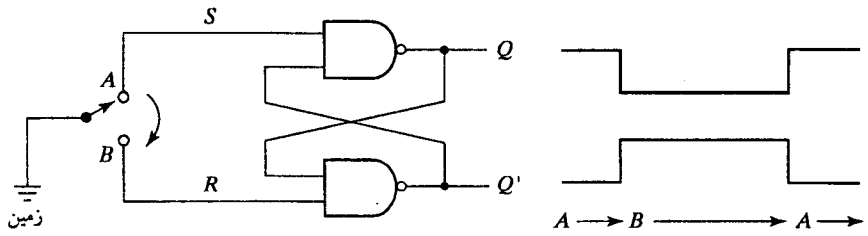
۳- نمودار منطقی را با استفاده از k لچ همراه با گیت‌های لازم برای تولید توابع بول S و R رسم کنید. برای لچ‌های NOR، از توابع بول S و R بدست آمده در مرحله ۲ استفاده کنید.

مثال مفید دیگری از پیاده‌سازی لچ را می‌توان در بخش ۷-۹ در ارتباط با شکل ۳۸-۹ بدست آورد.

مدار نوسان‌گیر

اطلاعات دودویی ورودی در یک سیستم دیجیتال را می‌توان بصورت دستی با سوئیچ‌های مکانیکی، تولید کرد. یک وضعیت سوئیچ، ولتاژی معادل با منطق 1، و دیگری ولتاژ دوم معادل منطق 0 را فراهم می‌سازد. از سوئیچ‌های مکانیکی برای شروع، ختم یا بازنشانی سیستم دیجیتال هم استفاده می‌شود. هنگام تست مدارهای دیجیتال در آزمایشگاه، سیگنال‌های ورودی معمولاً از سوئیچ‌ها می‌آیند. خصیصه معمول یک سوئیچ مکانیکی این است که وقتی اهرم آن از یک سو به سوی دیگر پرتاب می‌شود، اتصال سوئیچ قبل از استقرار نهایی، چندین بار نوسان می‌کند. در نوعی از یک سوئیچ ممکن است این نوسانات چندین میلی ثانیه طول بکشند. نوسانات مذکور ممکن است به دفعات موجب تغییر سیگنال منطقی بین 1 و 0 گردند.

مدار نوسان‌گیر، وظیفه حذف رشته پالس‌هایی را به عهده دارد که از نوسان اتصال مکانیکی حاصل می‌شود و بدین وسیله یک گذر یکنواختی برای سیگنال از 0 به 1 و از 1 به 0 فراهم می‌سازد. یک چنین مداری، طبق شکل ۱۵-۹، از یک سوئیچ دو حالته یک قطبی متصل به لچ SR تشکیل شده است. مرکز اتصال به زمین وصل است تا سیگنالی معادل 0 منطقی تولید کند. وقتی یکی از دو اتصال A یا B از طریق سوئیچ به زمین وصل نشده باشد یک سیگنال 1 منطقی تولید می‌کند. گاهی هر یک از این اتصالات از طریق یک مقاومت به یک ولتاژ ثابت متصل می‌شوند تا بدین وسیله یک سیگنال 1 منطقی ثابت و مناسبی تولید گردد. وقتی که سوئیچ، مطابق شکل از حالت A به حالت B پرتاب و سپس باز



شکل ۱۵-۹. مدار نوسان گیر

می‌گردد، لچ یک تک پالس منفی برای Q و مثبت برای Q' تولید می‌نماید. سوئیچ معمولاً از نوع فشاری است که حالت ثابت آن در وضعیت A قرار دارد. هنگام فشردن کلید، اتصال به B رفته و وقتی رها شود به وضعیت A باز می‌گردد.

عملکرد مدار نوسان‌گیر مطابق زیر است. هنگام اتصال سوئیچ به نقطه A، ما شرایط $S = 0$، $R = 1$ و $Q = 1$ و $Q' = 0$ را داریم، شکل ۱۱-۹ (ب). هنگام اتصال آن به نقطه B، اتصال به زمین سبب رفتن R به 0 و S به 1 می‌گردد زیرا اتصال A باز است. این وضعیت سبب رفتن خروجی Q به 0 و Q' به 1 می‌شود. سوئیچ پس از اولین اتصال به B چندین بار نوسان می‌کند، ولی باید به منظور داشتن یک عملکرد صحیح تصور کنیم که این بازگشت آنقدرها نیست تا به نقطه A برسد. خروجی لچ به علت باقی ماندن Q' در 1 (و Q در 0) بی‌تغییر خواهد ماند زیرا در ازاء 0 یا 1 بودن R (اتصال با زمین و عدم اتصال با زمین) Q' در 1 باقی می‌ماند. وقتی که سوئیچ به وضعیت A بازگردد، S برابر 0 و Q به 1 برمی‌گردد. مجدداً خروجی یک گذر صافی را علاوه بر وجود نوسان در وضعیت A، از خود به نمایش می‌گذارد.

۹-۴ روش طراحی

طراحی یک مدار غیر همزمان با صورت مسئله آغاز و با نمودار منطقی پایان می‌یابد. برای بهینه‌سازی پیچیدگی مدار و تولید یک مدار با ثبات و بدون رقابت تعدادی مراحل طراحی وجود دارد. بطور خلاصه مراحل طراحی بقرار زیرند:

از مشخصات طراحی، یک جدول روند اصلی بدست می‌آوریم. این جدول به حداقل حالات کاهش می‌یابد. آنگاه به حالات مقادیری دودویی تخصیص یافته و از آن جدول گذر بدست می‌آید. بالاخره از جدول گذر، نمودار منطقی مدار ترکیبی همراه با پسخوردی که در آن لچ‌های SR قرار دارند حاصل می‌گردد. فرآیند طراحی با ارائه یک مثال نمایش داده می‌شود. پس از حل مثال، درک مراحل طراحی ساده‌تر خواهد بود. بعضی از مراحل روش‌های مستدل‌تری را لازم دارند که در بخش‌های بعد مفصلاً تشریح شده است.

مثال طراحی

می‌خواهیم یک مدار لچ گیت را با دو ورودی G و D و یک خروجی Q بسازیم. اطلاعات دودویی حاضر در ورودی D وقتی که G برابر 1 باشد به خروجی Q منتقل می‌گردد. مادامی که $G=1$ است

خروجی Q ورودی D را دنبال خواهد کرد. وقتی که D به 0 برود اطلاعات موجود در ورودی D پس از گذر ساعت در خروجی Q باقی می‌ماند. لچ گیت‌دار عنصر حافظه‌ای است که مقدار D را وقتی $G = 1$ است پذیرفته و این مقدار را پس از رفتن G به 0 حفظ می‌نماید. به محض $G = 0$، هر تغییر در D مقدار خروجی Q را عوض نمی‌کند.

جدول روند اصلی

همانطور که قبلاً تعریف شد، یک جدول روند اصلی جدولی است با تنها یک حالت کلی پایدار در هر سطر. به یاد می‌آورید که یک حالت کل از ترکیب حالت داخلی با ورودی ایجاد می‌شود. اگر ابتدا جدولی را با همه حالت‌های کلی سیستم تشکیل دهیم، بدست آوردن جدول روند اصلی ساده می‌گردد. این کار برای لچ گیت‌دار در شکل ۲-۹ نشان داده شده است. هر سطر از جدول بیانگر یک حالت کلی است، که متشکل از یک حرف برای بیان حالت داخلی و ترکیبی از ورودی برای D و G است. خروجی Q هم برای هر حالت کلی نشان داده شده است. ما با دو حالت کلی که $G = 1$ دارند آغاز می‌کنیم. با توجه به ویژگی‌های طراحی، می‌دانیم که اگر $DG = 01$ باشد $Q = 0$ و اگر $DG = 11$ باشد، $Q = 1$ است زیرا با $G = 1$، D باید با Q برابر باشد. ما این حالات را به a و b تخصیص می‌دهیم. وقتی که G به 0 برود، خروجی به آخرین مقدار D بستگی دارد. بنابراین اگر، گذر DG از 01 به 00 و به 10 رخ دهد، آنگاه Q باید در 0 باقی بماند زیرا در زمان گذر G از 1 به 0، D برابر 0 است. اگر گذر DG از 11 به 10 به 00 باشد، آنگاه Q در 1 باقی خواهد ماند. این اطلاعات شش حالت کلی را مطابق جدول نتیجه می‌دهد. توجه کنید که گذر همزمان دو متغیر ورودی، مانند گذر از 01 به 10 یا از 11 به 00 دو عملکرد حالت اساسی مجاز نیست.

جدول روند اصلی برای لچ گیتی در شکل ۱۶-۹ دیده می‌شود. این جدول برای هر حالت دارای یک سطر و برای ترکیب ورودی یک ستون دارد. ابتدا یک مربع در هر سطر متعلق به حالت پایدار در آن سطر

	DG = 00	01	11	10
a	*c*, –	(*a*), 0	*b*, –	–, –
b	–, –	*a*, –	(*b*), 1	*e*, –
c	(*c*), 0	*a*, –	–, –	*d*, –
d	*c*, –	–, –	*b*, –	(*d*), 0
e	*f*, –	–, –	*b*, –	(*e*), 1
f	(*f*), 1	*a*, –	–, –	*e*, –

شکل ۱۶-۹. جدول روند اصلی

جدول ۲-۹ . حالتهای کلی لچ گیت‌دار

حالت	ورودی		خروجی	توضیحات
	D	G	Q	
a	0	1	0	$G = 1$ زیرا $D = Q$
b	1	1	1	$G = 1$ زیرا $D = Q$
c	0	0	0	پس از حالت a یا d
d	1	0	0	پس از حالت c
e	1	0	1	پس از حالت b یا f
f	0	0	1	پس از حالت e

را پر می‌کنیم. این وارده‌ها از جدول ۲-۹ معین می‌شوند. مثلاً وقتی ورودی 01 باشد حالت a با ثبات و خروجی 0 است. این اطلاعات در جدول روند، سطر اول ستون دوم وارد می‌شود. به طور مشابه، پنج حالت دیگر، همراه با خروجی آنها در ستون‌های مربوطه وارد می‌گردند.

سپس، چون تغییر همزمان دو ورودی مجاز نیست، می‌توانیم در سطری که نسبت به متغیرهای ورودی حالت پایدار در دو یا بیشتر متغیر اختلاف دارد خط تیره بگذاریم. مثلاً، اولین سطر در جدول روند حالت پایداری با ورودی 01 را نشان می‌دهد. چون در هر لحظه از زمان تنها یک متغیر مجاز به تغییر است، ورودی می‌تواند به 00 یا 11 تغییر نماید. بنابراین در سطر a ستون 10 دو خط تیره وارد می‌کنیم. این کار نهایتاً به معنی حالت بی‌اهمیت برای حالت بعدی در این مربع است. به دنبال این عمل، دومین مربع در هر سطر جدول روند پر خواهد شد.

آنگاه باید در هر سطر دو مربع دیگر را تعیین تکلیف کنیم. اظهار نظر موجود در جدول ۲-۹ می‌تواند در بدست آوردن اطلاعات لازم به ما کمک کند. مثلاً حالت c مربوط به ورودی 00 است و پس از یک تغییر ورودی از حالت a یا d فرا می‌رسد. به این دلیل یک حالت c ناپایدار در ستون 00 و سطرهای a و d از جدول روند نشان داده شده است. خروجی با خط تیره نشان داده شده است تا حالت بی‌اهمیت را بیانگر باشد. تفسیر این است که اگر مدار در حالت پایدار a باشد و ورودی از 01 به 00 برود، مدار ابتدا به یک حالت ناپایدار بعدی c می‌رود که در این صورت حالت فعلی a به c تبدیل می‌شود و در نتیجه گذری به سطر سوم ستون اول جدول روند را خواهیم داشت. مقادیر حالات ناپایدار دیگر که به طریقی مشابه بدست می‌آیند با خط تیره مشخص شده و به معنی حالات بی‌اهمیت تلقی می‌گردند. تخصیص مقادیر واقعی به خروجی‌ها بعد و پس از تکمیل یک مثال مورد بحث قرار خواهد گرفت.

کاهش جدول روند اصلی

یک جدول روند اصلی تنها یک حالت پایدار در هر سطر دارد. اگر دو یا چند حالت در یک سطر قرار گیرند جدول به تعداد کمتری از سطرها کاهش می‌یابد. جمع کردن حالات پایدار در یک سطر به زیر یک ستون را ادغام می‌گویند. ادغام کردن تعدادی از حالات پایدار واقع در یک سطر به این معنی است که متغیر حالت دودویی که نهایتاً به سطر ادغام شده تخصیص می‌یابد، با تغییر متغیرهای ورودی تغییر

DG	00	01	11	10
a	c, –	(a), 0	b, –	–, –
c	(c), 0	a, –	–, –	d, –
d	c, –	–, –	b, –	(d), 0

DG	00	01	11	10
b	–, –	a, –	(b), 1	e, –
e	f, –	–, –	b, –	(e), 1
f	(f), 1	a, –	–, –	e, –

(الف) حالت های منتخب برای ادغام

DG	00	01	11	10
a, c, d	(c), 0	(a), 0	b, –	(d), 0
b, e, f	(f), 1	a, –	(b), 1	(e), 1

DG	00	01	11	10
a	(a), 0	(a), 0	b, –	(a), 0
b	(b), 1	a, –	(b), 1	(b), 1

(ب) جدول کاهش یافته (دو انتخاب دیگر)

شکل ۱۷-۹ . کاهش جدول روند اصلی

نخواهد کرد. دلیل این است که در یک جدول روند اصلی، متغیر حالت در ازاء هر تغییر ورودی، تغییر می‌کند، ولی در یک جدول روند کاهش یافته، اگر حالت با ثبات بعدی با قبلی در یک سطر قرار داشته باشند، متغیر حالت تغییر نخواهد کرد.

روال کاهش یک جدول روند در بخش بعدی داده شده است. برای تکمیل مثال طراحی بدون پیگیری یک روال مشخص، فرآیند ادغام را با به کارگیری قواعد ساده شده ادغام اعمال خواهیم کرد. در جدول روند می‌توان دو یا چند سطر را در یک سطر ادغام کرد به شرطی که حالات و خروجی در هر یک از ستون‌ها در تضاد نباشند، اگر یک سمبل حالت و وارده‌های بی‌اهمیت در یک ستون ظاهر شوند، حالت در سطر ادغام شده لیست می‌گردد. به علاوه اگر حالت در یکی از سطرها در دایره محصور شده باشد، در سطر ادغام شده هم محصور می‌شود. مقدار خروجی هم در هر حالت پایدار در سطر ادغام شده لحاظ می‌گردد.

اکنون این قواعد را به جدول روند شکل ۱۶-۹ اعمال می‌کنیم. برای این که ببینیم چگونه این کار صورت می‌گیرد، جدول روند به دو بخش و هر بخش به سه سطر، طبق شکل ۱۷-۹ (الف) تقسیم می‌شود. هر بخش سه حالت با ثبات را نشان می‌دهد که قابل ادغامند زیرا هیچ وارده متضادی در هر یک از چهار ستون ملاحظه نمی‌شود. ستون اول حالت c را در همه سطرها و 0 و خط تیره را برای خروجی نشان می‌دهد. چون هر خط تیره بیانگر حالت بی‌اهمیت است، می‌تواند مربوط به هر حالت یا خروجی باشد. دو خط تیره در ستون اول را می‌توان به عنوان خروجی 0 تلقی کرد تا هر سه سطر مشابه با سطری شود که در آن حالت با ثبات c و خروجی 0 است. دومین ستون نشان می‌دهد که خطوط تیره را می‌توان

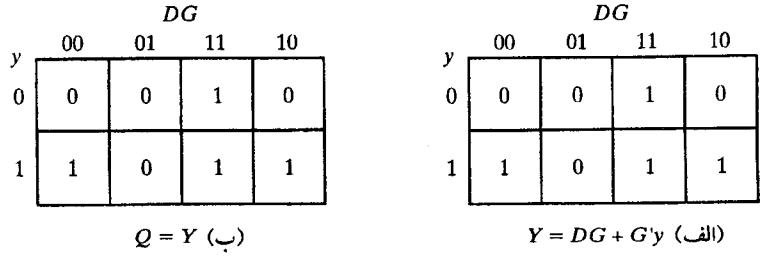
به حالت پایدار a با خروجی 0 تخصیص داد. توجه کنید که اگر حالت در یکی از سطرها محصور باشد، در سطر ادغام شده نیز محصور خواهد بود. به طور مشابه ستون سوم را می‌توان به یک حالت بی‌ثبات b و خروجی بی‌اهمیت ادغام نمود و چهارمین ستون به حالت پایدار d و خروجی 0 ادغام می‌شود. بنابراین سه سطر a، c و d را می‌توان به صورت یک سطر با سه حالت پایدار و یک حالت ناپایدار ادغام کرد. سطر اول شکل ۱۷-۹ (ب) این ادغام را نشان می‌دهد. سطر دوم جدول ادغام شده، از ادغام سطرهای b ، e و f در جدول روند بدست می‌آید. برای ترسیم جدول کاهش یافته دو راه وجود دارد. سمبل‌های حرفی برای حالات را می‌توان حفظ کرد تا رابطه بین جداول روند اصلی و کاهش یافته مشخص شود. روش دیگر این است که برای جداول روند اصلی و کاهش یافته برای همه حالات پایدار در سطرهای ادغام شده سمبل‌های حرفی مشترکی تعریف کنیم. بنابراین حالات c و d با حالت a و حالات e و f با حالت b جایگزین می‌شوند. هر دو روش در شکل ۱۷-۹ (ب) مشاهده می‌شود.

جدول گذر و نمودار منطقی

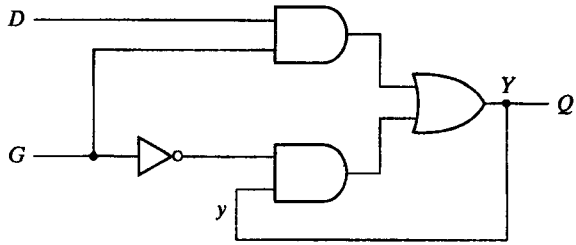
برای بدست آوردن مداری که با جدول روند کاهش یافته تعریف شده باشد، لازم است به هر حالت مقدار دودویی جداگانه‌ای اختصاص یابد. این تخصیص جدول روند را به جدول گذر مبدل می‌سازد. در حالت کلی، برای مصون بودن مدار از رقابت‌های بحرانی، تخصیص حالت دودویی باید صورت گیرد. مسئله تخصیص حالت در مدارهای ترتیبی غیرهمزمان در راه حل‌های آن در بخش ۶-۹ مورد بحث قرار گرفته است. خوشبختانه در جدول روند دو سطری حالت رقابت وجود ندارد، و بنابراین، می‌توانیم طراحی لچ گیت‌دار را قبل از بخش ۶-۹ مطالعه کنیم. با تخصیص 0 به حالت a و 1 به حالت b در جدول روند کاهش یافته شکل ۱۷-۹ (ب)، جدول گذر شکل ۱۸-۹ (الف) بدست می‌آید. در واقع جدول گذر نقشه‌ای برای متغیر تحریک Y است. آنگاه تابع بول ساده شده برای Y از نقشه به صورت زیر بدست می‌آید.

$$Y = DG + G'y$$

در جدول روند ساده شده نهایی دو خروجی بی‌اهمیت وجود دارد. اگر مقادیر خروجی را مطابق شکل ۱۸-۹ (ب) اختیار کنیم، ممکن است خروجی Q را برابر تابع تحریک Y نمود. اگر دیگر مقادیر ممکن را به خروجی بی‌اهمیت نسبت دهیم، خروجی Q را برابر y خواهیم کرد. در هر یک از دو حال،



شکل ۱۸-۹. جدول گذر و نقشهٔ خروجی لچ گیت دار



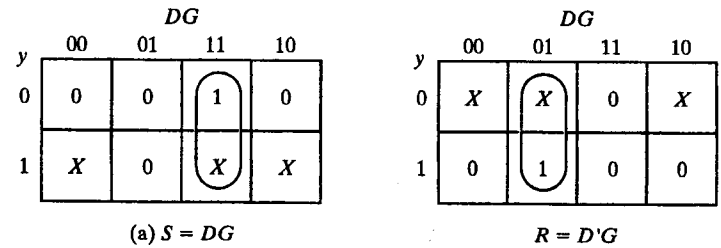
شکل ۱۹-۹. نمودار منطقی لچ گیت‌دار

نمودار منطقی لچ گیت‌دار مطابق شکل ۱۹-۹ خواهد بود.

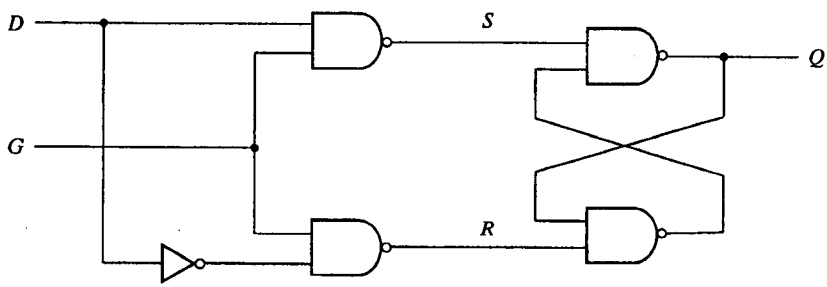
نمودار را با لچ SR هم می‌توان پیاده‌سازی کرد. با توجه به روال بخش ۳-۹ ابتدا توابع بول را برای S و R، طبق شکل ۲۰-۹ (الف) بدست می‌آوریم. نمودار منطقی با گیت‌های NAND در شکل ۲۰-۹ (ب) دیده می‌شود. توجه کنید که لچ گیت‌دار یک لچ D حساس به سطح است که در شکل ۶-۵ بخش ۲-۵ معرفی شد.

تخصیص خروجی‌ها به حالت ناپایدار

حالات پایدار در یک جدول روند دارای مقادیر خروجی خاص مربوط به خود هستند. حالات ناپایدار

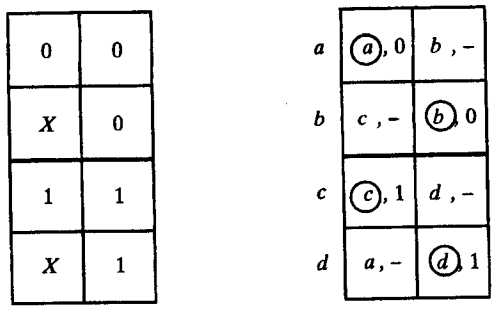


(الف) نقشه‌های S و R



(ب) نمودار منطقی

شکل ۲۰-۹. مدار با لچ SR



(الف) جدول روند **(ب) تخصیص خروجی**

شکل ۲۱-۹. انتساب مقادیر خروجی به حالتهای ناپایدار

دارای وارده‌های نامشخص می‌باشند که با خطوط تیره نشان داده شده‌اند. مقادیر خروجی برای حالات ناپایدار باید طوری انتخاب شوند که حین گذر بین دو حالت، خروجی‌های لحظه‌ای غلط را تولید نکنند. این بدان معنی است که در نتیجه یک گذر بین دو حالت قرار نیست که یک متغیر خروجی تغییر نماید. در این صورت یک حالت بی‌ثبات بین دو حالت پایدار باید خروجی یکسانی با خروجی حالات پایدار داشته باشد. به عنوان مثال جدول حالت شکل ۲۱-۹ (الف) را در نظر بگیرید. گذر از حالت پایدار a به حالت پایدار b از طریق حالت ناپایدار b صورت می‌گیرد. اگر خروجی مربوط به حالت بی‌ثبات b برابر 1 باشد، آنگاه یک پالس کوتاه لحظه‌ای ضمن جابجایی مدار از خروجی 0 در حالت a به خروجی 1 برای حالت بی‌ثبات b رفته و ضمن رفتن به حالت پایدار b دوباره به 0 باز می‌گردد. بنابراین خروجی مربوط به حالت ناپایدار b باید 0 در نظر گرفته شود تا خروجی لحظه‌ای غلط تولید نشود. اگر یک متغیر خروجی به دلیل تغییر حالت، عوض شود، آنگاه به این متغیر یک حالت بی‌اهمیت اختصاص می‌یابد. مثلاً گذر از حالت پایدار b به حالت پایدار c در شکل ۲۱-۹ (الف)، خروجی را از 0 به 1 تغییر مقدار می‌دهد. اگر برای حالت بی‌ثبات c خروجی 0 در نظر گرفته شود، آنگاه متغیر خروجی تا انتهای گذر تغییر نخواهد کرد. اگر یک 1 وارد کنیم، تغییر در آغاز گذر رخ می‌دهد. چون فرقی نمی‌کند که تغییر خروجی چه زمانی اتفاق بیفتد بنابراین برای خروجی حالت ناپایدار c یک X می‌گذاریم. شکل ۲۱-۹ (ب) تخصیص خروجی را برای جدول روند نشان می‌دهد. این جدول چهار حالت ممکن را در تغییر خروجی نمایش می‌دهد. روال تخصیص مقادیر به خروجی‌های مربوط به حالات بی‌ثبات در زیر خلاصه شده است.

۱- به متغیر خروجی که مربوط به حالت ناپایدار بین دو حالت پایدار با متغیر خروجی 0اند مقدار 0 را اختصاص دهید.

۲- به متغیر خروجی که مربوط به حالت ناپایدار بین دو حالت پایدار با متغیر خروجی 1اند، مقدار 1 را منتسب کنید.

۳- به متغیر خروجی که مربوط به حالت ناپایدار بین دو حالت پایدار با متغیرهای متفاوتند (0 و 1 یا 1 و 0) مقدار بی‌اهمیت را نسبت دهید.

خلاصه روش طراحی

طراحی مدارهای ترتیبی غیرهمزمان با استفاده از روش تشریح شده در مثال قبل قابل اجراست. بعضی از مراحل طراحی نیاز به تشریح بیشتری دارند که در بخش‌های زیر توضیح داده شده است. مراحل عبارتند از:

۱- از مشخصات طراحی مفروض، یک جدول روند اصلی بدست آورید. این مشکل‌ترین بخش طراحی است زیرا باید در آن تجربه و تبحر را برای رسیدن به یک تفسیر صحیح از مشخصات مسئله به کار برد.

۲- با ادغام سطرها در جدول روند اصلی، جدول روند کاهش یافته را بدست آورید. یک روال مستدل برای ادغام سطرها در بخش ۵-۹ ارائه شده است.

۳- به هر سطر جدول روند کاهش یافته متغیرهای حالت دودویی را نسبت دهید تا جدول گذر حاصل شود، روش تخصیص حالتی که رقابت‌های بحرانی احتمالی را حذف می‌کند در بخش ۶-۹ داده شده است.

۴- به خطوط تیره مربوط به حالات ناپایدار مقادیر خروجی را نسبت دهید تا نقشه خروجی تهیه شود. این کار قبلاً توضیح داده شد.

۵- توابع بولی تحریک و متغیرهای خروجی را ساده کنید و نمودار منطقی آنها را طبق بخش ۲-۹ رسم نمایید. نمودار منطقی را با لچ‌های SR طبق بخش ۲-۹ و نیز انتهای بخش ۷-۹ می‌توان ترسیم کرد.

۵-۹ کاهش جداول حالت و روند

روال کاهش تعداد حالات داخلی در یک مدار ترتیبی غیرهمزمان، روش به کار رفته در مدارهای همزمان را به خاطر می‌آورد. الگوریتم کاهش حالت یک جدول حالت کامل در بخش ۶-۵ داده شد. ما این الگوریتم را مرور کرده و آن را به روش کاهش حالتی که از یک جدول ایجاب استفاده می‌کند اعمال می‌نماییم. سپس الگوریتم و جدول ایجاب اصلاح می‌شود تا کاهش حالت مربوط به جداول حالت غیرکامل را پوشش دهد. این الگوریتم اصلاح شده برای تشریح روال کاهش جدول روند مدارهای ترتیبی غیرهمزمان به کار خواهد رفت.

جدول ایجاب

روال کاهش حالت جداول حالت کامل بر این اساس استوار است که اگر دو حالت در جدول حالت معادل باشند می‌توان آنها را ترکیب کرده و با یک حالت نشان داد. دو حالت را معادل گوییم اگر به ازاء هر ورودی ممکن، دقیقاً خروجی یکسانی تولید کنند و به حالت بعدی یکسان و یا حالات بعدی معادلی هم بروند. جدول ۶-۶ مثالی از حالات معادلی را نشان می‌دهد که دارای خروجی و حالات بعدی

جدول ۳-۹. جدول حالات برای نشان دادن حالتهای معادل

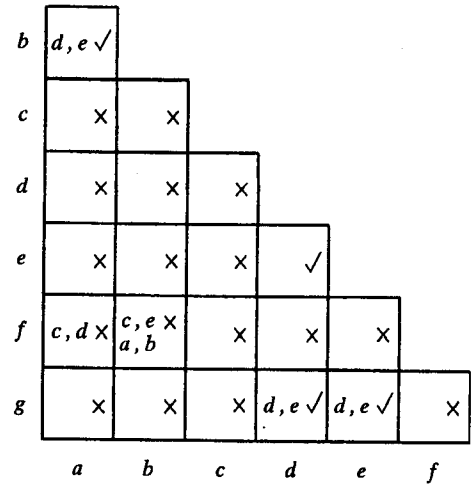
حالت فعلی	حالت بعدی		خروجی	
	x = 0	x = 1	x = 0	x = 1
a	*c*	*b*	0	1
b	*d*	*a*	0	1
c	*a*	*d*	1	0
d	*b*	*d*	1	0

یکسانی در قبال هر ترکیبی از ورودی‌هاست. مواردی وجود دارد که جفت حالت، حالات بعدی یکسانی ندارند، ولی کم و بیش به حالت بعدی معادلی می‌روند. مثلاً جدول حالت ۳-۹ را ملاحظه نمایید. حالات فعلی a و b به ازاء ورودی‌های یکسان، خروجی یکسانی دارند. حالت بعدی آنها برای $x = 0$ برابر c و d و بازاء $x=1$ برابر a و b است. اگر بتوانیم نشان دهیم که جفت حالت (d و c) معادلند، جفت حالت (b و a) هم معادل خواهند بود، زیرا آنها حالت بعدی یکسان یا معادلی خواهند داشت. وقتی چنین رابطه‌ای برقرار باشد، گوییم زوج (b و a) زوج (d و c) را موجب است. با توجه به دو سطر آخر جدول ۳-۹، می‌بینیم که جفت حالت (d و c) موجب (b و a) است. ویژگی حالات معادل این است که اگر (d و c) موجب (b و a) و (b و a) نیز (d و c) را موجب شوند، آنگاه جفت حالات معادلند؛ یعنی a و b معادلند و نیز c و d معادلند. در نتیجه چهار سطر جدول ۳-۹ را می‌توان با ترکیب a و b به یک حالت و c و d را به دومین حالت کاهش داد.

بررسی جفت حالات برای یافتن معادل‌های احتمالی در یک جدول که در آن تعداد قابل توجهی حالت وجود دارد، به طور سیستماتیک توسط جدول ایجاب قابل انجام است. جدول ایجاب چارتی متشکل از خانه‌های مربعی شکل است که هر یک متعلق به یک جفت حالت بوده و فضایی را برای لیست کردن هر حالت ممکن فراهم می‌سازد. با به کارگیری صحیح جدول، می‌توان همه جفت حالات معادل را پیدا کرد. جدول حالت ۴-۹، برای تشریح این روال به کار گرفته خواهد شد. جدول ایجاب در شکل ۲۲-۹ ملاحظه می‌شود. در ضلع سمت چپ عمودی همه حالاتی که در جدول حالت تعریف

جدول ۴-۹. جدول حالتی که قرار است کاهش یابد

حالت فعلی	حالت بعدی		خروجی	
	x = 0	x = 1	x = 0	x = 1
a	*d*	6	0	0
b	*e*	*a*	0	0
c	*g*	*f*	0	1
d	*a*	*d*	1	0
e	*a*	*d*	1	0
f	*c*	*b*	0	0
g	*a*	*e*	1	0



شکل ۲۲-۹. جدول ایجاب

شده‌اند، به جز اولی، لیست شده‌اند. در ضلع پایین جدول هم متغیرها به جز آخری لیست شده‌اند. نتیجه این کار نمایش همه ترکیبات ممکن از دو حالت با مربعی است که در محل تقاطع یک سطر و یک ستون واقع است و در آنها معادل‌ها تست خواهند شد.

در مربعات مربوط به دو حالتی که معادل نیستند علامت ضربدر (×) زده شده است، در حالی که معادل‌ها علامت تأیید (✓) خورده‌اند. بعضی از مربعات وارده‌هایی از حالات موجب دارند که باید بیشتر مورد بررسی قرار گیرند تا معین شود آیا معادلند یا خیر. روال قدم به قدم در پر کردن مربعات به شرح زیر است. ابتدا ضربدری در مربع جفت حالتی قرار می‌دهیم که خروجی‌های آنها در قبال هر ورودی یکی نیست. در این راستا، حالت c خروجی متفاوتی با دیگر حالات دارد و به این علت علامت ضربدر در دو مربع سطر c و چهار مربع ستون c، زده می‌شود. به همین ترتیب نُه مربع دیگر در این جدول وجود دارند که ضربدر زده شده‌اند.

سپس در مربع‌های باقیمانده جفت حالاتی را وارد می‌کنیم که بوسیله جفت حالات بیانگر مربع موجب شده‌اند. این کار را از مربع فوقانی در سمت چپ به پایین انجام می‌دهیم. و سپس کار را با ستون بعدی ادامه داده و به سمت راست پیش می‌رویم. با توجه به جدول حالات، می‌بینیم که معادل بودن (b و a) دلالت بر معادل بودن (e و d) دارد. بنابراین (e و d) در مربعی که به وسیله ستون a و سطر b تعریف می‌شود ثبت می‌گردد. به همین ترتیب کار را تا تکمیل شدن همه جدول ادامه می‌دهیم. دقت کنید که حالات (e و d) با هم معادلند زیرا به حالت بعدی یکسانی رفته و خروجی برابری دارند. در نتیجه یک علامت تأیید (✓) در مربعی می‌گذاریم که به وسیله ستون d و سطر e تعریف می‌شود. این کار بدان معنی است که دو حالت معادلند و از هر جفت حالت موجب مستقل می‌باشند.

قدم بعدی ساختن مسیرهای متوالی در جدول است که برای تعیین مربع‌هایی که باید (×) بخورند به

کار می‌روند. هر مربعی از جدول که حداقل حاوی یک جفت موجب نا معادل است ضربدر می‌خورد. مثلاً مربعی که به وسیله a و f تعریف می‌شود، بعد از c و d ضربدر خورده است زیرا جفت (d و c) مربعی را تعریف می‌کنند که حاوی یک ضربدر است. این روال تا جایی ادامه می‌یابد که دیگر مربعی برای ضربدر زدن وجود نداشته باشد. بالاخره در همه مربعاتی که ضربدر نخورده‌اند، علامت تأیید زده می‌شود. این مربعات جفت حالات معادل را تعریف می‌کنند. در این مثال حالات معادل عبارتند از

$$(a, b) \quad (d, e) \quad (d, g) \quad (e, g)$$

اکنون جفت حالات را به گروه حالات معادل بزرگتری تبدیل می‌نماییم. سه جفت آخر را می‌توان به یک گروه حالت معادل سه‌تایی (d, e, g) تبدیل کرد زیرا هر یک از حالات در گروه با دو حالت دیگر معادل است. آخرین بخش حالات متشکل از حالات معادلی است که از جدول ایجاب همراه با حالات باقیمانده در جدول حالت بدست می‌آیند و با دیگر حالات معادل نیستند، یعنی

$$(a, b) \quad (c) \quad (d, e, g) \quad (f)$$

این بدان معنی است که جدول ۴-۹ قابل کاهش از هفت حالت به چهار حالت است که هر حالت متعلق به یکی از پرانتزهاست. جدول کاهش یافته از جایگزینی b با a و حالات e و g با d حاصل می‌گردد. این جدول در شکل ۵-۹ مشاهده می‌شود.

ادغام جدول روند

مواردی وجود دارد که در آن جدول حالت برای یک مدار ترتیبی به طور غیرکامل یا ناقص بیان شده است. این مورد هنگامی پیش می‌آید که بعضی از ترکیبات ورودی‌ها یا رشته‌های ورودی به دلیل محدودیت‌ها هرگز رخ نمی‌دهند. در یک چنین حالتی، حالات بعدی و خروجی‌هایی که به شرط وقوع ورودی‌ها می‌باید رخ می‌دادند، هرگز اتفاق نمی‌افتند و بنابراین به عنوان حالات بی‌اهمیت تلقی می‌شوند. هر چند که مدارهای ترتیبی همزمان گاهی با جداول حالت ناقص نمایش داده می‌شوند، ولی، در اینجا علاقمند به بررسی مدارهای ترتیبی غیرهمزمانی هستیم که جدول روند اصلی آنها همیشه به صورت ناقص تعریف شده است.

حالات ناقص را می‌توان برای کاهش حالات جدول روند با یکدیگر ترکیب کرد. چنین حالاتی را

جدول ۵-۹. جدول حالات کاهش یافته

حالت فعلی	حالت بعدی		خروجی	
	x = 0	x = 1	x = 0	x = 1
a	d	a	0	0
c	d	f	0	1
d	a	d	1	0
f	c	a	0	0

نمی‌توانیم معادل بخوانیم زیرا تعریف قبلی معادل لازم می‌دارد تا همه خروجی‌ها و حالات بعدی برای همه ورودی‌ها مشخص باشند. در عوض دو حالت ناقصی که بتوانند با یکدیگر ترکیب شوند سازگار خوانده می‌شوند. دو حالت هنگامی سازگارند که برای هر ورودی ممکن خروجی یکسان و حالت بعدی آنها به محض اینکه مشخص شدند سازگار باشند. همه حالات بی‌اهمیتی که با خط تیره معین شده‌اند حالات نامشخص را بیان می‌کنند و هنگام جستجوی حالات سازگار تأثیری ندارند.

فرآیندی که باید برای یافتن گروه‌های سازگار در راستای ادغام یک جدول روند اعمال گردد قابل تفکیک به سه مرحله زیر است:

۱- با استفاده از جدول ایجاب همه جفت‌های سازگار را معین کنید.

۲- با استفاده از نمودار ادغام، سازگاری‌های ماکزیمال را بدست آورید.

۳- مجموعه‌ای از سازگارهای مینیمال را پیدا کنید که کلیه حالات را دربرگیرد و نیز مجموعه‌ای بسته باشد.

آنگاه این مجموعه مینیمال برای ادغام سطرهای جدول روند مورد استفاده قرار می‌گیرد. اکنون با استفاده از روند اولیه مثال بخش قبل، سه مرحله فوق را تشریح خواهیم کرد.

زوج‌های سازگار

روال یافتن زوج‌های سازگار در شکل ۲۳-۹ تشریح شده است. جدول روند اصلی در (الف) همانند شکل ۶-۹ می‌باشد. وارده‌ها در هر مربع، حالت بعدی و خروجی را بیان می‌کند. خطوط تیره (–) حالات یا خروجی‌های نامشخص را نمایش می‌دهند. جدول ایجاب، همچون موردی که در یافتن

	00	01	11	10
a	c, –	(a), 0	b, –	–, –
b	–, –	a, –	(b), 1	e, –
c	(c), 0	a, –	–, –	d, –
d	c, –	–, –	b, –	(d), 0
e	f, –	–, –	b, –	(e), 1
f	(f), 1	a, –	–, –	e, –

(الف) جدول روند اصلی

	a	b	c	d	e
b	✓				
c	✓	d, e ×			
d	✓	d, e ×	✓		
e	c, f ×	✓	d, e × c, f ×	×	
f	c, f ×	✓	×	d, e × c, f ×	✓

(ب) جدول ایجاب

شکل ۲۳-۹. جدولهای روند و ایجاب

حالات معادل در وضعیت کامل به کار رفت، برای بدست آوردن حالات سازگار نیز مورد استفاده است. تنها تفاوت این است که وقتی سطرها را مقایسه می‌کنیم، آزادیم تا خطوط تیره را برای تطبیق هر حالت موردنظر تنظیم کنیم.

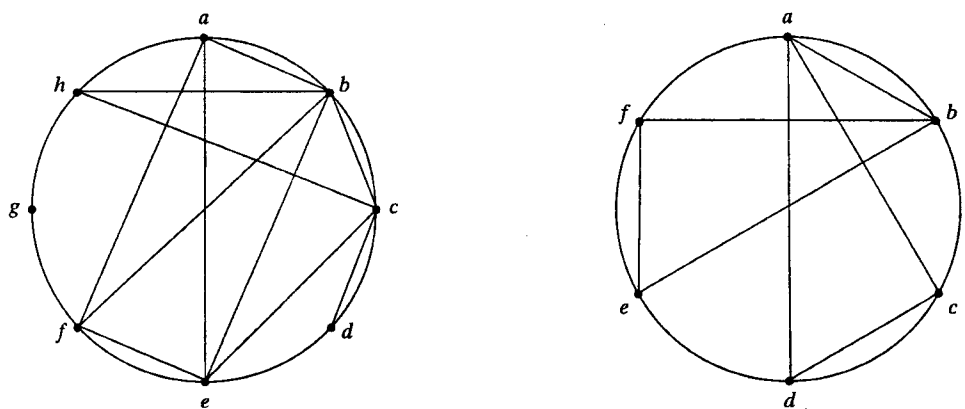
دو حالت را سازگار گوییم اگر در هر ستون از سطر مربوطه در جدول روند، حالات یکسان یا سازگار وجود داشته و تضاد و تناقضی در مقادیر خروجی نباشد. مثلاً سطرهای a و b در جدول روند سازگارند ولی سطرهای a و f فقط اگر c و f سازگار باشند، سازگار خواهند بود. معهذا سطرهای c و f سازگار نیستند زیرا در ستون اول دارای خروجی‌های متفاوتی می‌باشند. این اطلاعات در جدول ایجاب ثبت می‌شود. علامت تأیید (✓) مربعی را مشخص می‌کند که جفت حالات آن سازگارند. حالاتی که سازگار نباشند با (×) علامت خورده‌اند. مربع‌های باقیمانده با زوج‌های موجب پر می‌شوند و بررسی بیشتری را نیاز دارند.

جدول ایجاب اولیه به محض پر شدن، مجدداً مرور می‌گردد تا حالات موجبی که ناسازگارند ضربدر بخورند. مربع‌های باقیمانده که دارای علامت (✓) می‌باشند جفت‌های سازگار را مشخص می‌نمایند. در مثال شکل (۲۳-۹) زوج‌های سازگار عبارتند از:

$$(a,b)\quad (a,c)\quad (a,d)\quad (b,e)\quad (b,f)\quad (c,d)\quad (e,f)$$

سازگارهای ماکزیمال

پس از یافتن زوج‌های سازگار، پیدا کردن بزرگترین مجموعه‌ها از مجموعه‌های سازگار، قدم بعدی است. سازگار ماکزیمال گروهی از سازگارها هستند که حاوی همه ترکیبات ممکن از حالات سازگار باشند. سازگار ماکزیمال را از نمودار ادغام طبق شکل ۲۴-۹، می‌توان بدست آورد. نمودار ادغام گرافی است که در آن هر حالت با یک نقطه بر روی یک دایره مشخص می‌شود. بین دو نقطه که یک زوج سازگار



(الف) سازگارهای ماکزیمال
$(a,b,)\ (a,c,d)\ (b,e,f)$

(ب) سازگارهای ماکزیمال
$(a,b,e,f)\ (b,c,h)\ (c,d)\ (g)$

شکل ۲۴-۹ . نمودارهای ادغام

را تشکیل دهند یک خط رسم می‌شود. کلیه سازگارهای ممکن را می‌توان از روی نمودار ادغام با مشاهده شکل‌های هندسی که در آنها حالات به یکدیگر متصل هستند، بدست آورد. یک نقطه جدا مانده که به دیگر نقاط وصل نباشد حالتی است که با هیچ حالت دیگر سازگار نیست. یک خط، نشان دهنده یک زوج سازگار است. یک مثلث سه حالت سازگار را مشخص می‌کند. در نمودار ادغام، n حالت سازگار با یک کثیر الاضلاع n ضلعی که کلیه قطرهای آن وصل باشند، مشخص می‌شوند.

نمودار ادغام شکل ۲۴-۹ (الف) از روی لیست جفت‌های سازگار مشتق از جدول ایجاب شکل ۲۳-۹ بدست می‌آید. در این شکل هفت خط مستقیم وجود دارد که نقاط را به هم وصل می‌کنند و هر یک متعلق به یک زوج سازگار می‌باشند. الگوی هندسی حاصل از این خطوط شامل دو مثلث (a, c, d) و (b, e, f) و یک خط (a, b) می‌باشد. سازگارهای ماکزیمال عبارتند از:

$$(a, b) \quad (a, c, d) \quad (b, e, f)$$

شکل ۲۴-۹ (پ) نمودار ادغام یک جدول روند هشت حالته را نشان می‌دهد. در این شکل الگوی هندسی شامل مستطیلی است که قطرهایش رسم شده است تا چهار حالت سازگار (a, b, e, f)، یک مثلث (b, c, h)، یک خط (c, d) و یک حالت منفرد g که با هیچ حالت دیگری سازگار نیست، به وجود آید. سازگارهای ماکزیمال عبارتند از:

$$(a, b, e, f) \quad (b, c, h) \quad (c, d) \quad (g)$$

مجموعه سازگار ماکزیمال می‌تواند برای ادغام جدول روند استفاده شود به این ترتیب که در جدول کاهش یافته به هر عضو مجموعه سطری منتسب می‌گردد. در بسیاری از موارد سازگارهای ماکزیمال لزوماً مجموعه سازگارهای مینیمال را تشکیل نمی‌دهند. غالباً تهیه مجموعه کوچکتری از سازگارها که شرط ادغام سطرها را دارا باشند، امکان‌پذیر است.

شرط پوششی بسته

شرطی که برای ادغام سطر وجود دارد این است که مجموعه سازگارهای منتخب باید همه حالات را پوشش داده و بسته باشند. مجموعه‌ای تمام حالات را پوشش می‌دهد که همه حالات جدول اولیه را دارا باشد. شرط بسته بودن هنگامی برقرار است که حالات موجب وجود نداشته باشد و یا این حالات در داخل خود مجموعه باشند. یک مجموعه بسته از سازگارها که شامل همه حالات باشد، مجموعه پوششی بسته نامیده می‌شود. شرط پوششی بسته، با دو مثال زیر توضیح داده شده‌اند.

به سازگارهای ماکزیمال شکل ۲۴-۹ (الف) توجه کنید. اگر (a, b) را حذف کنیم، دو مجموعه سازگار را خواهیم داشت.

$$(a, c, d) \quad (b, e, f)$$

همه حالات ششگانه جدول روند شکل ۲۳-۹ در این مجموعه وجود دارند. این خود حالت پوششی را می‌رساند. با توجه به جدول ایجاب شکل ۲۳-۹ (ب) حالات موجبی برای (a, c) ؛ (a, d) ؛ (c, d)؛ (b, e) ؛ (b, f) و (e, f) وجود ندارند، بنابراین شرط بسته بودن برقرار است. به این ترتیب جدول روند

اصلی قابل ادغام به دو سطر و هر یک برای یکی از سازگارها خواهد بود. جزئیات ساختن جدول کاهش یافته برای این مثال خاص در بخش قبل انجام و در شکل ۱۷-۹ (ب) نشان داده شد.

مثال دوم از یک جدول روند اصلی است (در اینجا نشان داده نشده است) که جدول ایجاب آن در شکل ۵-۹ (الف) دیده می‌شود. زوج سازگارهای حاصل از جدول ایجاب عبارتند از:

$$(a, b)\ (a, d)\ (b, c)\ (c, d)\ (c, e)\ (d, e)$$

با توجه به نمودار ادغام شکل ۲۵-۹ (ب)، سازگارهای ماکزیمال را معین می‌کنیم:

$$(a, b) \quad (a, d) \quad (b, c) \quad (c, d, e)$$

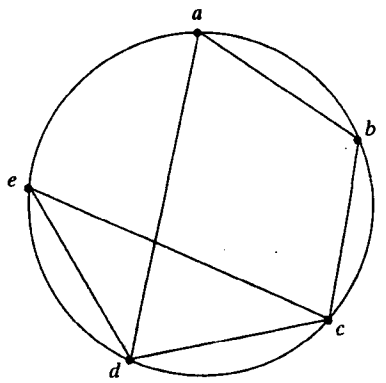
اگر دو سازگار

$$(a, b) \quad (c, d, e)$$

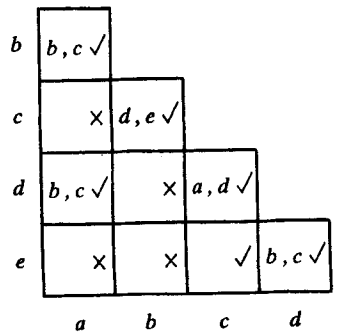
را انتخاب کنیم در مجموعه مزبور هر پنج حالت جدول اصلی را می‌پوشاند. شرط بسته بودن را به کمک جدول بسته بودن شکل ۲۵-۹ (پ) می‌توان چک کرد. جفت‌های موجب که برای هر سازگار لیست شده مستقیماً از جدول ایجاب حاصل شده‌اند. حالات موجب برای (a, b)، حالات (b, c) می‌باشند. ولی (b, c) در مجموعه انتخابی (a, b) (c, d, e) وجود ندارد، در نتیجه مجموعه سازگارها بسته نیست. مجموعه‌ای از سازگارها که شرط پوششی بسته را اغناء می‌کنند عبارتند از:

$$(a, d) \quad (b, c) \quad (c, d, e)$$

این یک مجموعه پوششی است زیرا دارای هر پنج حالت می‌باشد. توجه کنید هر حالت می‌تواند بیش از



(ب) نمودار ادغام



(الف) جدول ایجاب

سازگارها	(a, b)	(a, d)	(b, c)	(c, d, e)
حالات موجب	(b, c)	(b, c)	(d, e)	$(a, d,)$ $(b, c,)$

(پ) جدول بسته بودن

شکل ۲۵-۹. انتخاب مجموعه‌ای از سازگارها

یک بار تولید شود. مجموعه شرط بسته بودن را هم دارد زیرا حالات موجب (a, d) و (b, c) (d, e) می‌باشند که در مجموعه لحاظ شده‌اند. جدول روند اصلی (که در اینجا نشان داده نشده است) را می‌توان با ادغام سطرهای a و d؛ b و c؛ c و d با e از 5 سطر به 3 سطر کاهش داد. توجه کنید انتخاب دیگری که سازگارهای پوششی بسته را شامل می‌شود عبارت است از (a, b) (d, e) (b, c). به طور کلی، ممکن است هنگام کاهش جدول روند اصلی بیش از یک راه ادغام وجود داشته باشد.

۶-۹ تخصیص حالت فاقد رقابت

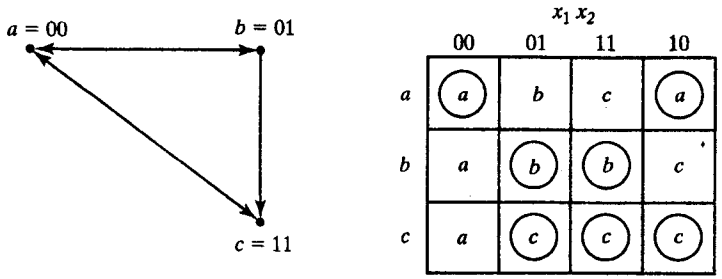
به محض این که یک جدول روند کاهش یافته برای یک مدار ترتیبی غیرهمزمان بدست آمد، قدم بعدی در طراحی، تخصیص مقادیر دودویی به هر حالت پایدار است. این تخصیص موجب تبدیل جدول روند به جدول گذر معادلش می‌شود. هدف اصلی در انتخاب یک تخصیص دودویی صحیح ممانعت از رقابت‌های بحرانی است. موضوع رقابت‌های بحرانی در بخش ۲-۹ به همراه شکل ۷-۹ توضیح داده شد.

از رقابت‌های بحرانی می‌توان اجتناب کرد به این ترتیب که تخصیص حالت دودویی طوری صورت گیرد که در هر لحظه از زمان که گذر حالتی در جدول روند رخ می‌دهد تنها یک متغیر عوض شود. برای حصول به این تفکر، لازم است تا بین حالاتی که گذر اتفاق می‌افتد تخصیص‌های همجوار را نسبت دهیم. دو مقدار دودویی را همجوار گوییم اگر فقط در یک متغیر با هم اختلاف داشته باشند. مثلاً 010 و 011 مجاورند زیرا فقط در بیت سوم با هم اختلاف دارند.

برای اطمینان از داشتن یک گذر فاقد رقابت‌های بحرانی، لازم است هر گذر ممکن بین دو حالت پایدار تست شده و اطمینان حاصل کنیم که هر بار یک متغیر حالت دودویی تغییر می‌کند. البته این کار خسته کننده است، به خصوص اگر سطرها و ستون‌های جدول متعدد باشند. برای ساده کردن موضوع، ما روال تخصیص حالت دودویی را با چند مثالی که فقط سه یا چهار سطر دارند توضیح خواهیم داد. این مثال‌ها روالی کلی را که باید دنبال شوند نشان می‌دهند تا یک تخصیص حالت فاقد رقابت تضمین شود. آنگاه می‌توان روال را به جداول روندی با هر تعداد سطر و ستون تعمیم داد.

مثال جدول روند سه سطری

انتساب یک متغیر دودویی به یک جدول روند دو سطری مشکل رقابت بحرانی را دربرندارد. یک جدول روند سه سطری به تخصیص دو متغیر نیاز دارد. اگر تخصیص مقادیر دودویی به حالات پایدار به درستی انجام نشود ممکن است سبب بروز رقابت‌های بحرانی گردد. مثلاً به جدول روند کاهش یافته شکل ۲۶-۹ (الف) توجه کنید. به منظور سادگی خروجی‌ها از جدول حذف شده‌اند. بررسی سطر a نشان می‌دهد که از حالت a به b در ستون 01 و از حالت a به c در ستون 11 گذری وجود دارد. طبق شکل ۲۶-۹ (ب) این اطلاعات به نمودار گذر انتقال می‌یابد. خطوط مستقیم بین a و b، و نیز بین a و c دو انتقال فوق را نشان می‌دهند. به طور مشابه گذر از دو سطر دیگر نیز با خطوط جهت‌دار در نمودار گذر نمایش داده شده است. نمودار گذر، نمایشی مصور از کلیه انتقال‌های موجود بین سطرهاست.



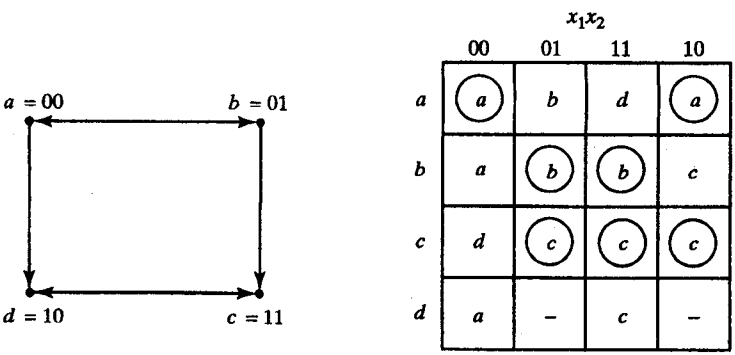
(ب) نمودار انتقال

(الف) جدول روند

شکل ۲۶-۹ . مثال جدول روند سه سطری

برای اجتناب از رقابت‌های بحرانی باید تخصیص حالتی دودویی را بیابیم که در هر گذر حالت آن تنها یک متغیر عوض شود. تلاشی در راستای یافتن چنین تخصیصی در نمودار گذر نشان داده شده است. به حالت a، مقدار دودویی 00 و به حالت c مقدار دودویی 11 منتسب شده است. این تخصیص به هنگام گذر از a به c یک رقابت بحرانی ایجاد خواهد کرد زیرا در متغیرهای حالت دودویی دو تغییر وجود دارد. توجه کنید که گذر از c به a هم موجب بروز رقابت می‌گردد ولی غیر بحرانی است.

با افزودن یک سطر اضافی به جدول روند می‌توانیم یک تخصیص فاقد رقابت را بدست آوریم. استفاده از یک سطر اضافی چهارم تعداد متغیرهای حالت را افزایش نمی‌دهد، بلکه سیکل‌هایی را بین دو حالت پایدار به وجود خواهد آورد. به جدول روند اصلاح شده در شکل ۲۷-۹ توجه کنید. سه سطر اول همان وضعیت جدول سه سطری اصلی را دارند. به سطر چهارم، که با d نام‌گذاری شده و مجاور به a و c است 10 تخصیص یافته است. اکنون گذر از a به c باید از طریق d انجام شود، به این صورت که متغیرهای دودویی از $a = 00$ به $d = 10$ و سپس به $c = 11$ تغییر می‌نمایند. و بنابراین حالت رقابت پرهیز می‌شود. این کار با تغییر سطر a، ستون 11 به d و سطر d، ستون 11 به c انجام می‌شود به همین ترتیب گذر از c به a علیرغم وجود رقابت غیربحرانی در ستون 00 از طریق حالت ناپایدار d انجام می‌گردد.



(ب) نمودار گذر

(الف) جدول روند

شکل ۲۷-۹ . جدول روند با یک سطر اضافی

	x_1x_2 00	01	11	10
$a = 00$	(00)	01	10	(00)
$b = 01$	00	(01)	(01)	11
$c = 11$	10	(11)	(11)	(11)
$d = 10$	00	–	11	–

شکل ۲۸-۹. جدول گذر

جدول گذر مربوط به جدول روند همراه با تخصیص حالت دودویی در شکل ۲۸-۹ نشان داده شده است. دو خط تیره در سطر d بیانگر حالات نامشخصی هستند که می‌توانند به عنوان حالات بی‌اهمیت تلقی شوند. البته باید مراقب بود که تخصیص 10 به این مربعات صورت نگیرد تا بدین وسیله از ایجاد یک حالت پایدار ناخواسته در سطر چهارم جلوگیری به عمل آید.

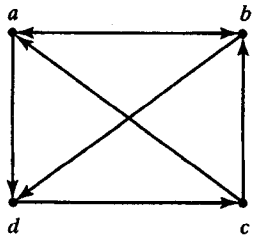
این مثال، استفاده از یک سطر اضافی در جدول روند را به منظور بدست آوردن یک انتساب فاقد رقابت نشان داد. این سطر اضافی به هیچ حالت پایدار خاصی منسوب نشد بلکه در عوض برای تبدیل یک رقابت بحرانی به سیکل مورد استفاده قرار گرفت که از میان حالات همجوار عبور می‌نمود. گاهی ممکن است وجود یک سطر اضافی برای جلوگیری از رقابت‌های بحرانی کافی نباشد و مجبور شویم دو یا چند سطر را به جدول روند اضافه کنیم. این کار در مثال بعدی نشان داده شده است.

مثال جدول روند چهار سطری

یک جدول روند با چهار سطر به حداقل دو متغیر نیاز دارد. هر چند دستیابی به یک تخصیص فاقد رقابت گاهی فقط با دو متغیر حالت دودویی امکان‌پذیر است، ولی در بسیاری از موارد نیاز به سطرهای اضافی برای اجتناب از رقابت‌های بحرانی، استفاده از سه متغیر حالت دودویی را دیکته می‌کند. به عنوان مثال جدول روند و نمودار گذر مربوطه‌اش را در شکل ۲۹-۹ ملاحظه نمایید. اگر گذری در جهت

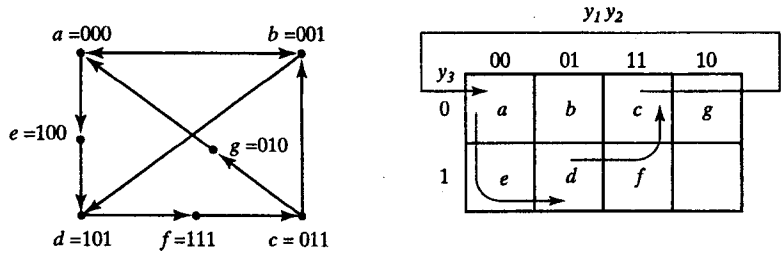
	00	01	11	10
a	b	(a)	d	(a)
b	(b)	d	(b)	a
c	(c)	a	b	(c)
d	c	(d)	(d)	c

(الف) جدول روند



(ب) نمودار گذر

شکل ۲۹-۹. مثال جدول روند چهار سطری



(ب) نمودار گذر (الف) تخصیص یا انتساب دودویی

شکل ۳۰-۹ . انتخاب سطرهای اضافی برای جدول روند

قطب (از b به d یا از c به a) وجود نداشته باشد یافتن یک تخصیص همجوار برای چهار گذر باقیمانده امکان‌پذیر است. با یک یا دو گذر قطری، برای تخصیص دو متغیر دودویی که نیاز همجواری را برآورده سازد، راهی وجود ندارد. بنابراین حداقل سه متغیر حالت دودویی لازم است.

شکل ۳۰-۹ یک نقشه تخصیص حالت را نشان می‌دهد که برای هر جدول روند چهار سطری مناسب است. حالات a، b، c و d حالات اصلی‌اند و حالات e، f و g حالات اضافی‌اند. حالاتی که در مربعات همجوار واقعند، تخصیص‌های مجاور خواهند داشت. عدد دودویی 001 به حالت b منتسب شده و همجوار با سه حالت اصلی دیگر است. گذر از a به d باید از طریق حالت اضافی e صورت گیرد تا چرخه یا سیکلی ایجاد شود به نحوی که هر بار تنها یک متغیر دودویی عوض شود. به طور مشابه گذر از c به a از طریق g انجام می‌گردد و گذر از d به c هم از f هدایت می‌شود. با توجه به تخصیص مربوط به نقشه، می‌توان طبق شکل ۳۱-۹ جدول چهار سطری را به جدول هفت سطری فاقد رقابت گسترش داد.

	00	01	11	10
000 = a	b	(a)	e	(a)
001 = b	(b)	d	(b)	a
011 = c	(c)	g	b	(c)
010 = g	–	a	–	–
110 –	–	–	–	–
111 = f	c	–	–	c
101 = d	f	(d)	(d)	f
100 = e	–	–	d	–

شکل ۳۱-۹ . انتساب حالات برای اصلاح جدول روند

توجه کنید که با وجود بودن هفت سطر، تنها چهار حالت پایدار وجود دارد. حالاتی که در سه سطر بدون دایره مشخص شده‌اند صرفاً برای ایجاد یک گذر بدون رقابت بین حالت‌های پایدار آمده‌اند.

مثال فوق روش انتخاب سطرهای اضافی در یک جدول روند برای دستیابی به یک تخصیص بی‌رقابت را نشان داد. یک نقشه انتساب حالات شبیه به نقشه شکل ۳۰-۹ (الف) می‌تواند در اغلب موارد مفید باشد. گاهی می‌توان از وارده‌های نامعین در جدول روند سود برد. در عوض اضافه کردن سطرها به جدول، ممکن است بتوان با هدایت بعضی از گذرهای حالات از طریق وارده‌های بی‌اهمیت، رقابت‌های بحرانی را حذف نمود. تخصیص واقعی تا رسیدن به یک تخصیص رضایت‌بخش فاقد رقابت با روش سعی و خطا اجرا می‌شود.

روش چند سطری

روشی که با اضافه کردن سطرهای اضافی برای دستیابی به یک گذر فاقد رقابت در جدول روند در دو مثال قبل تشریح شد را گاهی روش سطر مشترک می‌گویند. برای این دستیابی روش دومی نیز وجود دارد که البته به اندازه روش سطر مشترک کارایی ندارد ولی کاربرد آن آسانتر است. این روش، روش چند سطری نامیده می‌شود. در تخصیص چند سطری، هر حالت از جدول روند اولیه با دو یا چند ترکیب از متغیرهای حالت جایگزین می‌گردد. نقشه تخصیص حالت شکل ۳۲-۹ (الف) چنین انتسابی را نشان می‌دهد که قابل استفاده در جدول روند چهار سطری است. برای هر حالت پایدار دو متغیر حالت دودویی وجود دارد که هر یک متمم منطقی دیگری است. مثلاً حالت اولیه a با دو حالت معادل $a_1 = 000$ و $a_2 = 111$ جایگزین شده است. مقادیر خروجی که در اینجا نشان داده نشده‌اند، باید در a_1

	00	01	11	10
$000 = a_1$	b_1	(a_1)	d_1	(a_1)
$111 = a_2$	b_2	(a_2)	d_2	(a_2)
$001 = b_1$	(b_1)	d_2	(b_1)	a_1
$110 = b_2$	(b_2)	d_1	(b_2)	a_2
$011 = c_1$	(c_1)	a_2	b_1	(c_1)
$100 = c_2$	(c_2)	a_1	b_2	(c_2)
$010 = d_1$	c_1	(d_1)	(d_1)	c_1
$101 = d_2$	c_2	(d_2)	(d_2)	c_2

(ب) جدول روند

y_2y_3

y_1	00	01	11	10
0	a_1	b_1	c_1	d_1
1	c_2	d_2	a_2	b_2

(الف) انتساب دودویی

شکل ۳۲-۹. انتساب چند سطری

و a_2 یکسان باشند. توجه کنید که a_1 مجاور به b_1، c_2 و d_1، و a_2 مجاور به c_1، b_2 و d_2 و به طور مشابه هر حالت مجاور به سه حالت با حروف متفاوت است. رفتار مدار در قبال حالت داخلی a_1 و یا a_2 یکسان است و به همین ترتیب دیگر حالات نیز چنین خواصی دارند.

شکل ۳۲-۹ (ب) یک تخصیص چند سطری را برای جدول روند اولیه شکل ۲۹-۹ (الف) نشان می‌دهد. جدول گسترش یافته، از جایگزینی هر سطر از جدول روند اصلی با دو سطر به وجود می‌آیند. مثلاً سطر b با سطرهای b_1 و b_2 جایگزین شده و حالت پایدار b در هر دو سطر b_1 و b_2 در ستون‌های 00 و 11 آورده شده است. پس از این که همه حالت‌های باثبات وارد شدند، حالات ناپایدار با مراجعه به تخصیص‌های مشخص شده در نقشه قسمت (الف) پر می‌شوند. هنگام انتخاب حالت بعدی برای یک حالت فعلی، حالتی که مجاور حالت فعلی است انتخاب می‌شود. در جدول اولیه، حالات بعدی b برای ورودی‌های 10 و 01 به ترتیب a و d هستند. در جدول گسترش یافته حالات بعدی b_1 عبارتند از a_1 و d_2 زیرا این حالات مجاور b_1 می‌باشند. به طور مشابه حالات بعدی b_2 هم a_2 و d_1 می‌باشند زیرا این حالات با b_2 مجاورند.

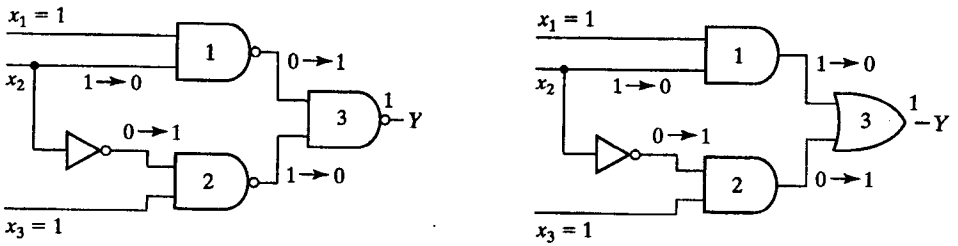
در تخصیص چند سطری، تغییر از یک حالت پایدار به حالت پایدار دیگر، همواره موجب تغییر فقط یک متغیر حالت دودویی می‌شود. هر حالت با ثبات دارای دو انتساب دودویی با خروجی‌های دقیقاً یکسان است. در هر لحظه از زمان تنها یکی از تخصیص‌ها به کار گرفته می‌شود. مثلاً اگر با حالت a_1 و ورودی 01 شروع کنیم و سپس ورودی را به 11، 01 و 00 تغییر دهیم و پس از آن به 01 بازگردیم، رشته حالات داخلی عبارت از a_1، d_1، c_1 و a_2 خواهد بود. اگر چه مدار با حالت a_1 شروع و در حالت a_2 خاتمه می‌یابد، ولی از دید رابطه ورودی- خروجی دو حالت a_1 و a_2 معادل a در جدول روند اصلی‌اند.

۹-۷ هزارد (HAZARD)

هنگام طراحی مدارهای ترتیبی غیرهمزمان، برای دستیابی به عملکردهای صحیح، باید محدودیت‌ها و احتیاط‌های لازم را در نظر گرفت. مدار باید در حالت اساسی کار کند، یعنی در هر لحظه فقط یک ورودی عوض شود و فاقد رقابت‌های بحرانی باشد. به علاوه، یک پدیده دیگر به نام هزارد hazard یا خروجی‌های تصادفی هم وجود دارد که ممکن است موجب اشکال در کار مدار گردد. هزاردها گذرهای سوئیچینگ ناخواسته‌ای هستند که ممکن است در خروجی مدارها ظاهر شوند زیرا مسیرهای مختلف در مدار دارای تأخیرهای انتشار متفاوت می‌باشند. هزاردها در مدارهای ترکیبی رخ می‌دهند و سبب تولید مقدار خروجی غلط موقت می‌گردند. هنگام وقوع این وضعیت احتمال گذر به یک حالت پایدار غلط وجود دارد. بنابراین لازم است مدار به لحاظ وقوع هزاردها مورد بررسی قرار گیرد تا مشخص شود آیا آنها موجب یک عملکرد نادرست می‌شوند یا خیر. برای حذف تأثیر آنها باید اقداماتی را به عمل آورد.

هزاردها در مدارهای ترکیبی

هزارد یا "خروجی تصادفی" وضعیتی است که در آن یک تغییر آنی در خروجی ایجاد کند در زمانی



(الف) مدار AND-OR

(ب) مدار NAND

شکل ۳۳-۹. مدارهایی با هزارد

که نباید ایجاد کند. مدار شکل ۳۳-۹ (الف) رخداد یک هزارد را نشان می‌دهد. فرض کنید هر سه ورودی در ابتدا برابر 1 هستند. این وضعیت موجب می‌شود تا خروجی گیت شماره 1 برابر 1، خروجی گیت شماره 2 برابر 0 و بنابراین خروجی مدار برابر 1 گردد. اکنون فرض کنید x_2 از 1 به 0 برود. خروجی گیت شماره 1 به 0 و خروجی گیت شماره 2 به 1 تغییر خواهد کرد ولی خروجی کل در همان 1 باقی خواهد ماند. با این وجود اگر تأخیر انتشار گیت وارونگر لحاظ شود، خروجی ممکن است برای یک لحظه به 0 برود. تأخیر انتشار در وارونگر ممکن است سبب شود تا قبل از این که خروجی گیت 2 به 1 تغییر کند، خروجی گیت 1 برابر 0 شود. در این لحظه هر دو ورودی گیت 3 برابر 0 هستند و این باعث می‌شود تا برای یک لحظه کوتاه، یعنی در مدت زمان تأخیر انتشار سیگنال ورودی x_2 در وارونگر، خروجی مدار 0 شود.

مدار شکل ۳۳-۹ (ب)، پیاده‌سازی NAND همان تابع بول است. این مدار نیز به دلیلی مشابه دارای هزارد است. چون گیت‌های 1 و 2 از نوع NAND هستند، خروجی آنها متمم گیت‌های AND نظیرشان است. وقتی x_2 از 1 به 0 می‌رود، هر دو ورودی گیت 3 ممکن است برابر 1 گردند و بنابراین تولید یک تغییر لحظه‌ای 0 را در وضعیت 1 بنمایند.

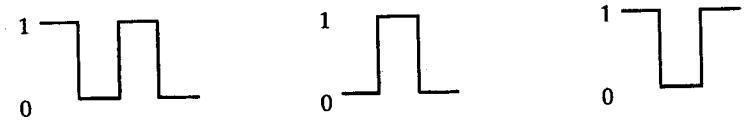
دو مدار شکل ۳۳-۹ تابع بول را به صورت جمع حاصلضرب‌ها پیاده‌سازی کرده‌اند:

$$Y = x_1x_2 + x_2'x_3$$

این نوع پیاده‌سازی ممکن است موجب رفتن خروجی به 0 به هنگام 1 بودن گردد. اگر مدار به صورت ضرب حاصل جمع‌ها پیاده‌سازی شود (بخش ۵-۳)، یعنی

$$Y = (x_1 + x_2')(x_2 + x_3)$$

آنگاه خروجی ممکن است به طور لحظه‌ای به هنگام 0 بودن به 1 برود. حالت اول را هزارد −1 ایستا و دومی را هزارد 0 ایستا می‌نامند. سومین نوع هزارد که هزارد پویا نام دارد موجب می‌شود تا خروجی به جای تغییر از 0 به 1 و یا 1 به 0، سه بار یا بیشتر تغییر حالت دهد. شکل ۳۴-۹ هر سه نوع هزارد را نشان می‌دهد. وقتی مداری به فرم جمع حاصلضرب‌ها یا گیت‌های AND−OR یا با NAND پیاده‌سازی می‌شود، حذف هزارد 1 ایستا عدم وقوع هزارد 0 ایستا و هزارد پویا را تضمین می‌کند.

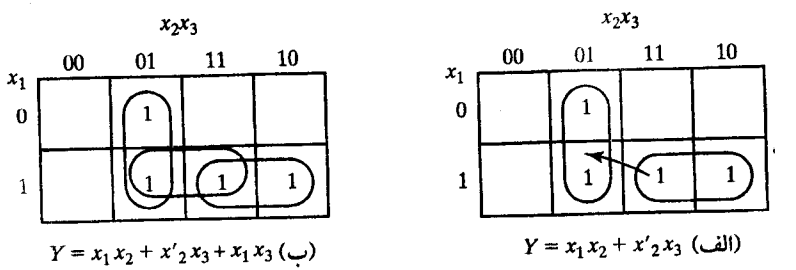


(الف) هزارد 1 ایستا (ب)هزارد 0ایستا (پ) هزارد پویا

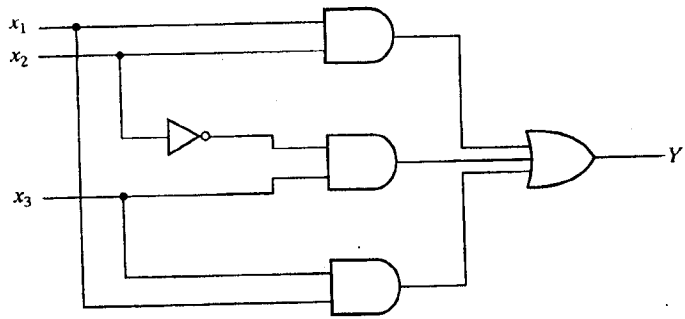
شکل ۳۴-۹. انواع هزارد

رخداد خروجی‌های هزارد را می‌توان با بررسی نقشه مدار آن آشکار کرد. برای تشریح موضوع، نقشه شکل ۳۵-۹ (الف) را ملاحظه نمایید که ترسیمی از تابع پیاده‌سازی شده در شکل ۳۳-۹ است. تغییر x_2 از 1 به 0 مدار را از مینترم 111 به مینترم 101 می‌برد. در اینجا هزارد وجود دارد زیرا متغیر ورودی، جمله حاصلضرب متفاوتی که دو مینترم را می‌پوشاند تولید خواهد کرد. مینترم 111 با جمله ضرب پیاده شده درگیت 1، و مینترم 101 به وسیله جمله ضرب پیاده شده درگیت 2 از شکل ۳۳-۹ پوشش یافته است. هر وقت مدار بخواهد از یک جمله ضرب به جمله ضرب دیگری برود، وقتی که هیچ یک از خروجی‌ها 1 نیستند، موجب تولید خروجی لحظه‌ای ناخواسته 0 می‌گردد.

چاره کار برای حذف هزارد پوشش دادن دو مینترم با جمله ضرب دیگری است که هر دو را در برگیرد. این کار در شکل ۳۵-۹ (ب) نشان داده شده است و در آن دو مینترمی که هزارد را تولید می‌کنند در یک جمله ضرب با یکدیگر ترکیب شده‌اند. مدار فاقد هزارد حاصل از این آرایش در شکل ۳۶-۹ دیده می‌شود. گیت اضافی در مدار جمله حاصلضرب x_1x_3 تولید می‌کند. به طور کلی هزاردها را در مدارهای ترکیبی می‌توان حذف کرد، به این ترتیب که دو مینترمی را که موجب هزارد می‌شوند با جمله ضرب مشترک در هر دو پوشش دهیم. لازمه حذف هزارد افزودن گیت‌های اضافی به مدار است.



شکل ۳۵-۹. نقشه‌های نشان دهندهٔ هزارد و حذف آن



شکل ۳۶-۹. مدار فاقد هزارد