فصل ششم

ثبات‌ها و شمارنده‌ها

دو نوع مهم از مدارات ترتیبی عبارتند از ثبات‌ها[1] (رجیسترها) و شمارنده‌ها.
یک ثبات n ـ بیتی گروهی از n فلیپ فلاپ (و احیاناً تعدادی گیت) است که قادر به ذخیره‌ی n بیت از اطلاعات دودویی است (هر فلیپ فلاپ قادر به ذخیره‌ی یک بیت اطلاعات است).
یک شمارنده ذاتاً یک ثبات است که با هر لبه‌ی فعال کلاک واردِ یکی از حالات از قبل مشخص و تعیین‌شده می‌شود.

ثبات‌ها

ساده‌ترین ثبات در شکل زیر نشان داده شده است.
در این نوع ساده، با هر لبه‌ی بالارونده‌ی کلاک، داده‌ی چهاربیتیِ ورودی در خروجیِ ثبات بارگذاری می‌شود. لذا امکان کنترل زمان دلخواه برای انجام بارگذاری وجود ندارد مگر این که کلاک سیستم را فقط در زمان‌های مورد نظرمان فعال کنیم که این کار رایج و توصیه شده نیست زیرا کلاک یک سیستم در حکم پمپی است که ضربان ثابتی را برای اجزاء مختلف آن سیستم فراهم می‌کند تا در ارتباطات و تبادل داده دارای نظم و هماهنگی باشند؛ بنابراین، وجود یک کلاک منظم برای یک سیستم بسیار ضروری است. حال، برای برقراری امکان بارگذاری داده در زمان‌های دلخواه، دو راه در اختیار داریم: کنترل کلاک وارد شده به ثبات، کنترل داده‌های وارد شده به ثبات. در روش اول، کلاک را با سیگنال کنترلی ترکیب (برای مثال، AND) می‌کنیم تا در زمان‌های مورد نظرمان کلاک وارد ثبات شود. در روش دوم، داده‌ها را با سیگنال کنترلی ترکیب می‌کنیم تا تعیین کنیم داده‌های چه موقع به ورودی فلیپ فلاپ‌های ثباتها رسیده و سپس در لحظه‌ی لبه‌ی فعالِ کلاک توسط فلیپ فلاپ‌ها ذخیره شوند. از بین این دو روش، روش کنترل داده بیشتر توصیه می‌شود؛ زیرا در روش اول، ترکیب کلاک سیستم با تعداد گیت منطقی (به منظور کنترل آن) تاخیر نابرابری در مسیرهای مختلف رسیدنِ کلاک سیستم به ورودیِ کلاک فلیپ فلاپ‌ها به وجود می‌آورد. برای همزمانی کاملا سیستم، باید مطمئن بود که همه‌ی پالس‌های کلاک به طور همزمان به هر نقطه از سیستم می‌رسد و بنابراین، همه‌ی فلیپ‌فلاپ‌ها به طور همزمان تریگر می‌شوند.

[1] Register

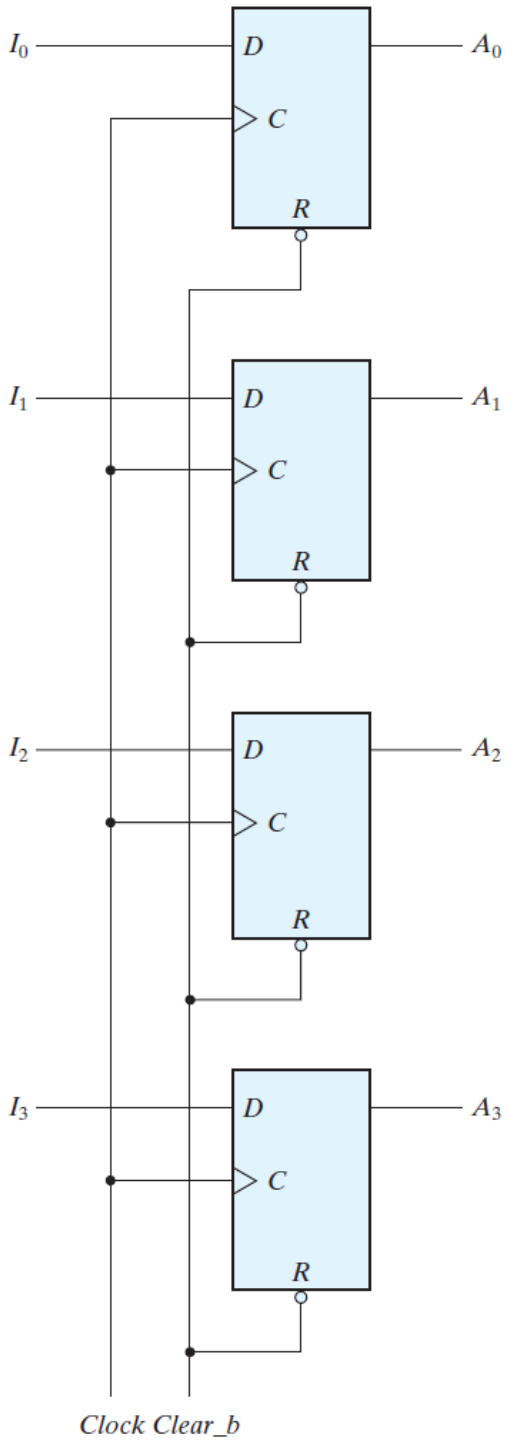


FIGURE 6.1
Four-bit register

ثبات با (قابلیتِ) بارگذاریِ موازی

در این ثبات، برای فراهم آوردن امکان بارگذاری داده‌ی ورودی در زمان دلخواه، از یک ورودی کنترل ویژه استفاده شده است:

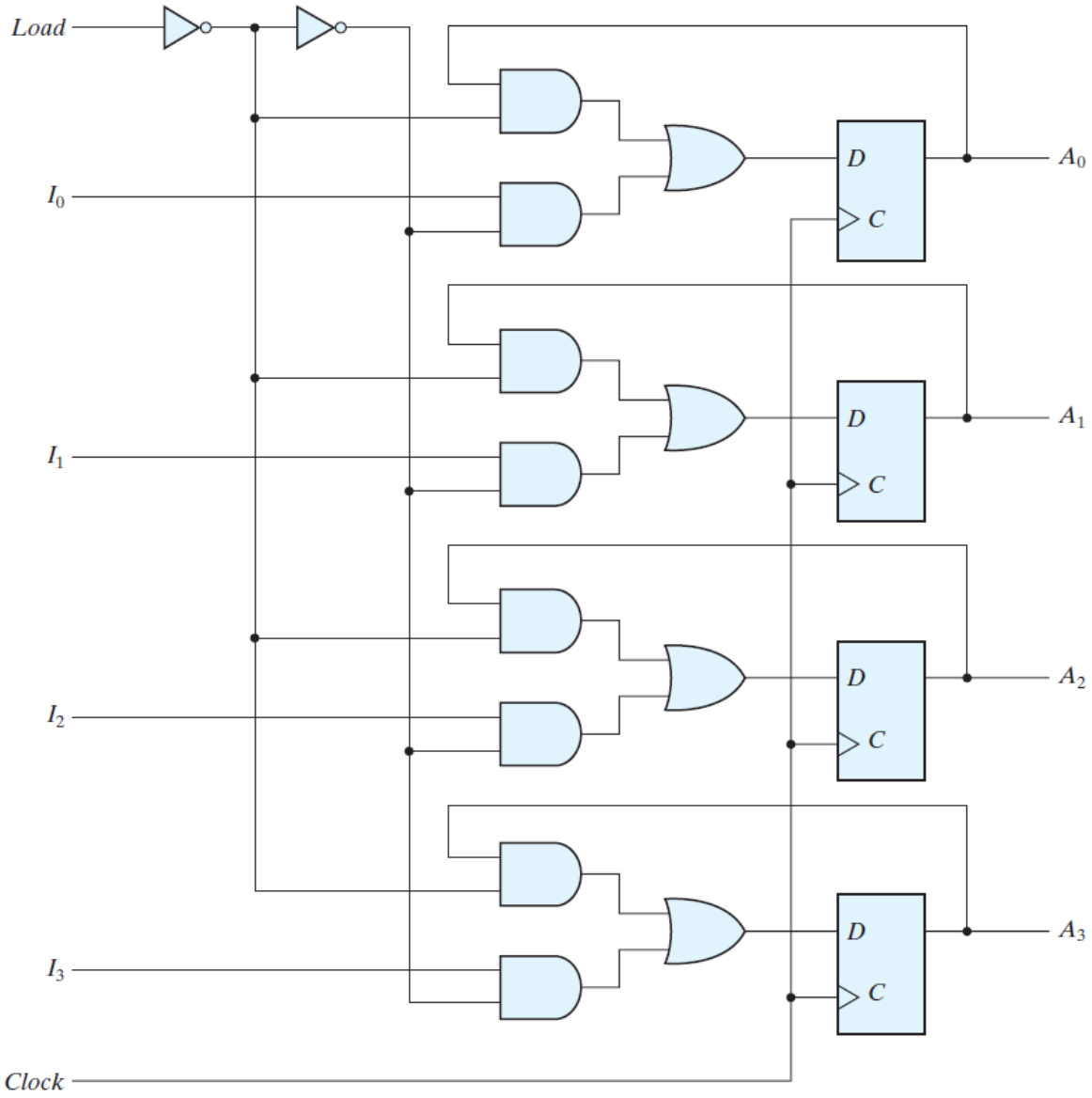


FIGURE 6.2
Four-bit register with parallel load

شیفت‌رجیسترها

یک «شیفت رجیستر» یا «ثبات جابجایی»، ثباتی است که قادر به شیفت دادن (یا جابجا کردن) اطلاعاتش به راست یا چپ است. اگر اطلاعات به سمت چپ جابجا شوند، از سمت راست باید یک ورودی سریال داشته باشیم و بالعکس. به دلیل وجود ویژگی جابجا کردن، انتظار می‌رود خروجی هر فلیپ فلاپ موجود در این نوع ثبات، به ورودی فلیپ فلاپ دیگر راه داشته باشد.

ساده‌ترین شیفت‌رجیستر:

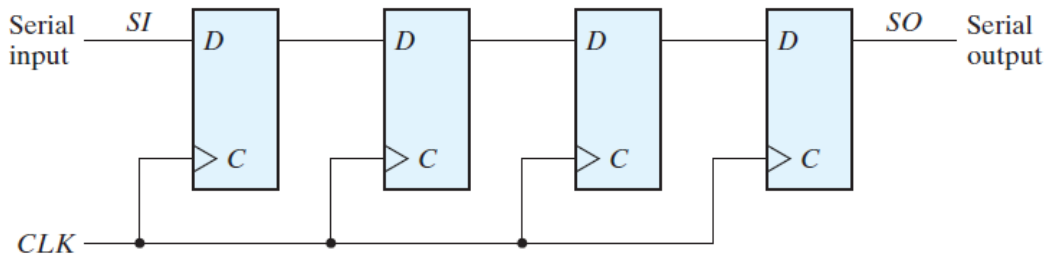
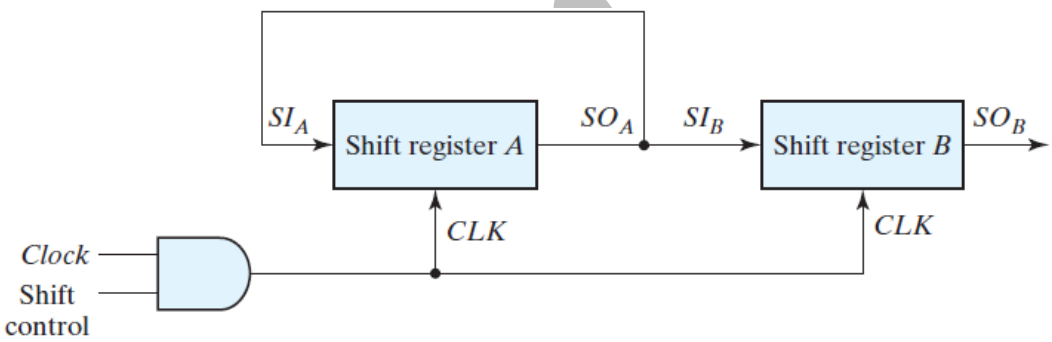


FIGURE 6.3
Four-bit shift register

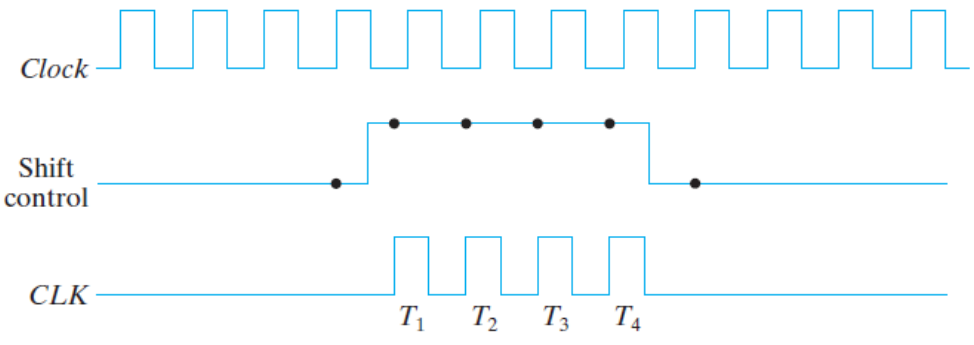
در این شیفت رجیستر هیچ کنترلی در مورد زمان دلخواه انجام شیفت در اختیار نداریم.

انتقال سریال

منظور انتقال محتویات یک شیفت رجیستر به شیفت رجیستر دیگر است.



(a) Block diagram



(b) Timing diagram

FIGURE 6.4
Serial transfer from register *A* to register *B*

پس از چهار پالس کلاک مقدار اولیه‌ی ثبات A به داخل این ثبات برمی‌گردد.

Table 6.1
Serial-Transfer Example

Timing Pulse	Shift Register *A*				Shift Register *B*			
Initial value	1	0	1	1	0	0	1	0
After T_1	1	1	0	1	1	0	0	1
After T_2	1	1	1	0	1	1	0	0
After T_3	0	1	1	1	0	1	1	0
After T_4	1	0	1	1	1	0	1	1

جمع سریال

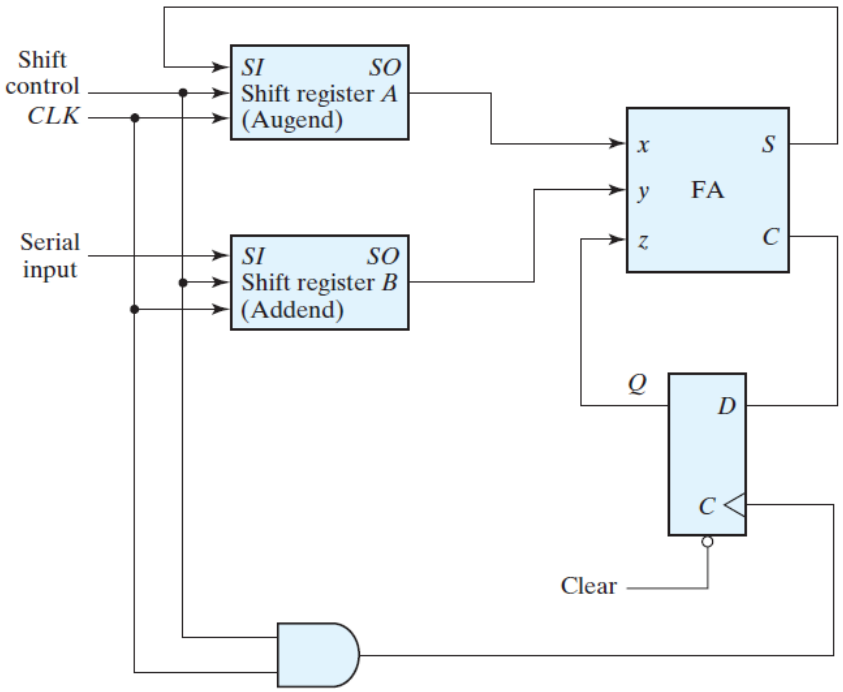


FIGURE 6.5
Serial adder

ثبات A می‌تواند نقش یک انباره یا آکومولاتور را ایفا کرده و مرتباً مقادیری را که در ثبات B قرار می‌گیرند، در خود انباشته کند. پس از شروعِ عملیات جمعِ A با اولین عدد موجود در ثبات B، بیتهای عدد بعدی با هر لبه‌ی فعالِ کلاک وارد ثبات B می‌شوند تا در مرحله‌ی بعدیِ (در صورت تداوم کلاک‌زنی) با محتویات انباشته شده در A جمع شوند. پس از اتمام عملیات جمعِ A با عدد فعلیِ ذخیره شده در ثبات B و قبل از شروع عملیات جدید، از طریق ورودیِ Clear رقم نقلی تولید شده در عملیات جمع باید پاک (یعنی صفر) شود.

طراحی جمع‌کننده‌ی سریال به کمک روش معرفی شده (مبتنی بر جدول حالت و نوشتن معادله‌ی ورودی)

فرض می‌کنیم دو شیفت رجیستر برای ذخیره‌ی اعدادی که قرار است با هم به صورت سریال جمع شوند، موجود است. خروجی‌های سریال از ثبات‌ها را x و y می‌نامیم. مدار ترتیبی دارای ورودی‌های x و y است. خروجی مدار نیز، S، حاصل جمع دو بیت تعریف می‌شود. حالت فعلی مدار، Q، رقم نقلیِ ورودی و حالت بعدی مدار نیز رقم نقلی تولید شده در اثر عمل جمع تعریف می‌شود. بنابراین، جدول حالت به صورت زیر در می‌آید؛ اگر بخواهیم در طراحی مدار از JKFF استفاده کنیم، باید معادلات تحریک JKFF را نیز در جدول وارد کنیم تا بتوانیم به معادلات ورودیِ فلیپ فلاپ برسیم:

Table 6.2
State Table for Serial Adder

Present State	Inputs		Next State	Output	Flip-Flop Inputs	
Q	x	y	Q	S	J_Q	K_Q
0	0	0	0	0	0	X
0	0	1	0	1	0	X
0	1	0	0	1	0	X
0	1	1	1	0	1	X
1	0	0	0	1	X	1
1	0	1	1	0	X	0
1	1	0	1	0	X	0
1	1	1	1	1	X	0

البته اگر بخواهیم در طراحی مدار از DFF استفاده کنیم، مدار به صورت مدار نشان داده شده‌ی قبلی در خواهد آمد.
با کمک جدول کارنو و استفاده از جدول حالت اخیر، معادلات ساده‌شده‌ی ورودیِ فلیپ فلاپ JKFF به صورت زیر خواهد شد:

$$J_Q = xy$$
$$K_Q = x'y' = (x + y)'$$
$$S = x \oplus y \oplus Q$$

دیاگرام منطقی مدار به صورت زیر می‌شود:

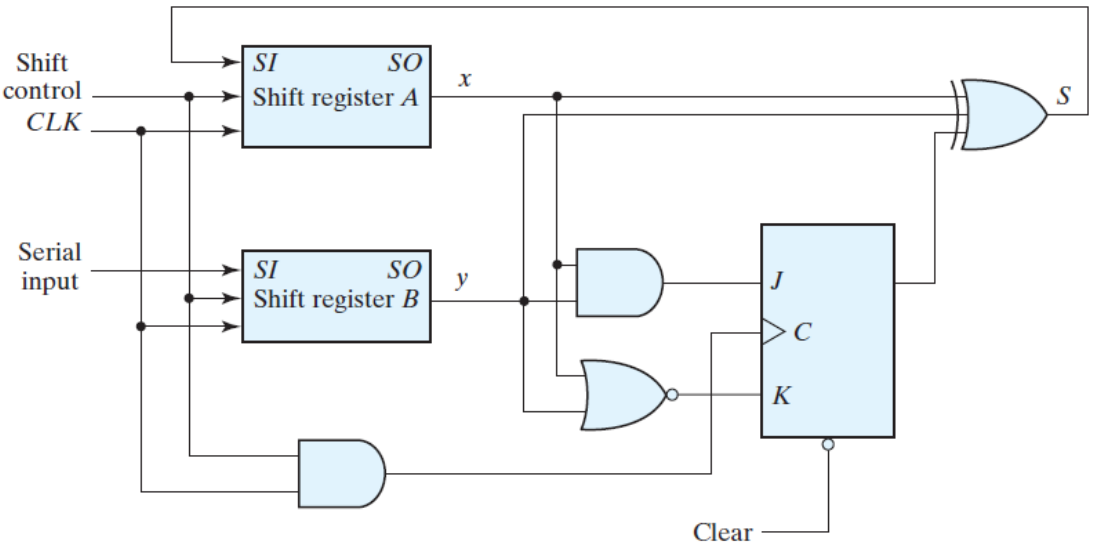


FIGURE 6.6
Second form of serial adder

ملاحظه می کنید که خروجیِ S نه تنها تابعی از دو ورودیِ x و y بلکه تابعی از رقم نقلی (یا همان حالت مدار یا خروجیِ فلیپ فلاپ) نیز می‌باشد.

شیفت‌رجیسترهای یونیورسال

شیفت رجیستر یک‌جهته[1]: شیفت‌رجیستری که تنها قادر به جابجایی داده در یک جهت باشد.
شیفت رجیستر دوجهته[2]: شیفت‌رجیستری که قادر به جابجایی داده در هر دو جهت باشد.
شیفت رجیستر جامع یا یونیورسال[3]: شیفت‌رجیستری که علاوه بر جابجایی دوطرفه، قادر به بارگذاری موازی داده‌ها باشد.

شیفت‌رجیسترهای یونیورسال معمولاً دارای امکانات و قابلیت‌های زیر هستند:
۱ ـ کنترل Clear برای صفر کردن مقدار/خروجیِ ثبات،
۲ ـ ورودی کلاک برای سنکرون‌سازی عملیات‌ها،
۳ـ کنترل جابجایی به راست (برای فعال کردن عملیات جابجایی به راست) به همراه خطوط ورودی و خروجی سریال مربوط به جابجایی به راست،
۴ ـ کنترل جابجایی به چپ (برای فعال کردن عملیات جابجایی به چپ) به همراه خطوط ورودی و خروجی سریال مربوط به جابجایی به چپ،
۵ـ کنترل بارگذاری موازی (برای فعال کردن عملیات انتقال موازی) به همراه n خط ورودی مربوط به انتقال موازی،

[1] Unidirectional shift register
[2] Bidirectional shift register
[3] Universal shift register

۶ـ n خط خروجی موازی،

۷ـ یک حالت کنترلی که علیرغم وجود پالسهای کلاک، موجب حفظ و نگهداشتن اطلاعات ذخیره شده در ثبات می‌شود.

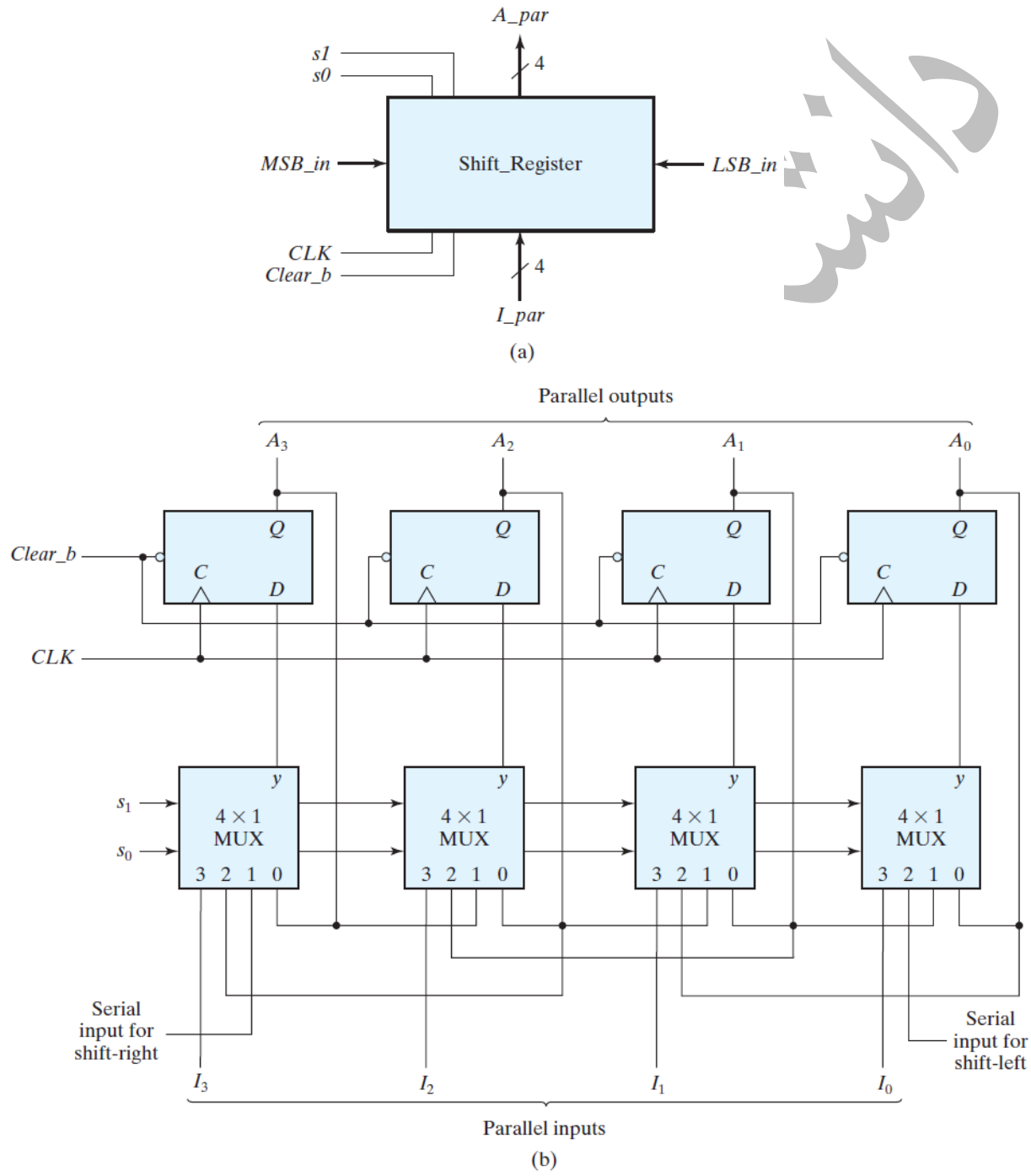


FIGURE 6.7
Four-bit universal shift register

یکی از مهمترین کاربردهای شیفت‌رجیسترها، انتقال داده بین دو سیستم دیجیتال دور از هم است. فرستنده داده‌ی موازی n بیتی را به شیفت‌رجیستر داده تا پس از تبدیل موازی به سریال، تنها از طریق یک خط داده، به گیرنده ارسال شود. گیرنده به کمک یک شیفت‌رجیستر داده‌ی سری را دریافت و آن را به موازی تبدیل می‌کند.

شمارنده‌های موج‌گونه (یا ریپل)

شمارنده‌ها دو دسته‌اند: موج‌گونه و همزمان. در موج‌گونه (ریپل)، هر فلیپ‌فلاپ ورودیِ کلاک خود را از خروجی یا خروجی‌های دیگر فلیپ فلاپ‌ها می‌گیرد (به بیان دیگر، ورودیِ کلاک تمام فلیپ‌فلاپ‌ها لزوماً از سیگنال کلاکِ ورودی تامین نمی‌شود). اما در نوع همزمان (سنکرون)، کلاکِ تمام فلیپ‌فلاپ‌ها فقط توسط سیگنال کلاک ورودی به سیستم تامین می‌شود.

شمارنده‌ها برحسب نوع شمارش خود نیز به دسته‌هایی تقسیم می‌شود. برای مثال شمارنده‌ی دودویی، BCD، و گری.

همچنین بر حسب این که شمارنده تنها رو به بالا قادر به شمارش باشد یا رو به پایین یا رو به هر دو جهت، شمارنده را به ترتیب بالا ـ شمار[1]، پایین ـ شمار[2]، و بالا ـ پایین ـ شمار[3] می‌نامند.

شمارنده‌ی موج‌گونه‌ی دودیی

دو نمونه از چنین شمارنده‌ای در شکل بعدی نشان داده شده است. اساس عملکرد این شمارنده‌ها با توجه به نحوه شمارش دودویی که قسمتی از آن در جدول زیر نشان داده شده است، قابل درک است: هر بیتِ A_i از خروجیِ شمارنده زمانی باید متمم شود که بیتِ A_{i-1} یک گذر پایین رونده (یعنی از 1 به 0) را طی کرده باشد. اولین بیت (یعنی A_0) نیز با هر لبه‌ی فعال کلاکِ ورودی (با نام Count) باید مرتباً متمم شود. (انتشار سیگنال از یک طبقه به طبقه‌ی دیگر مانند موج است)

Table 6.4
Binary Count Sequence

A_3	A_2	A_1	A_0
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0

[1] Up counter
[2] Down counter
[3] Up-Down counter

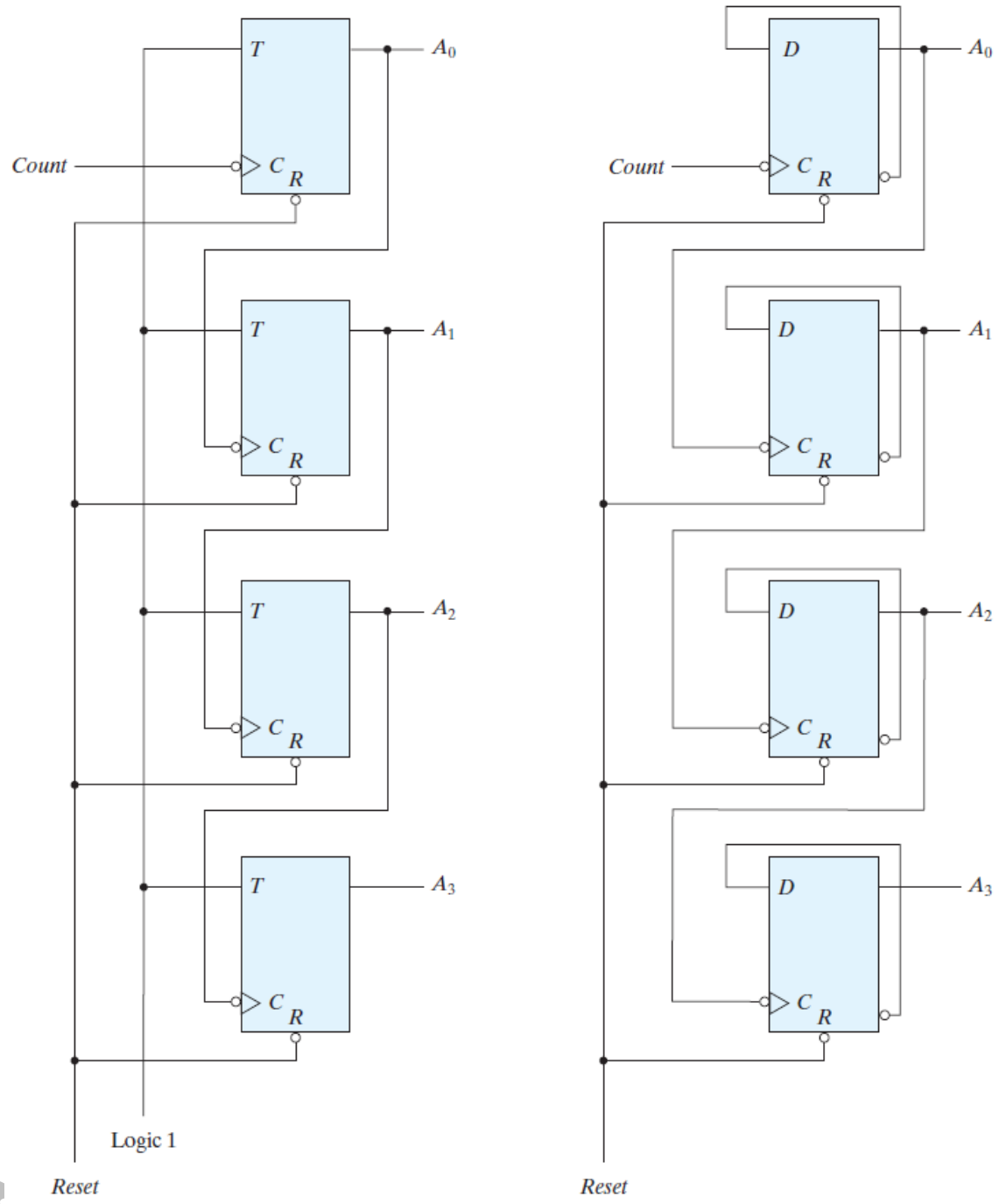


FIGURE 6.8
Four-bit binary ripple counter

دو شمارنده‌ی فوق، بالاشمار هستند. برای تبدیل این شمارنده‌ها به نوع پایین‌شمار، می‌توانیم دوباره به جدول شمارش پایین‌شمار دقت کرده و ملاحظه کنیم که: بیت A_0 مجدداً با هر لبه‌ی فعالِ کلاک باید متمم شود. بیت‌های بعدی باید زمانی متمم شوند که بیت قبلی‌شان یک گذر از 0 به 1 طی کرده باشد؛ بنابراین، در شمارنده‌های اخیر کافی است یا فلیپ‌فلاپ‌های

حساس به لبه‌ی مثبت استفاده شوند یا این که ورودیِ C هر فلیپ فلاپ را به خروجی متمم‌ساز فلیپ فلاپ قبلی متصل کنیم.

شمارنده‌ی BCD موج‌گونه

دیاگرام حالت یک شمارنده‌ی BCD موج‌گونه[1] به صورت زیر است:

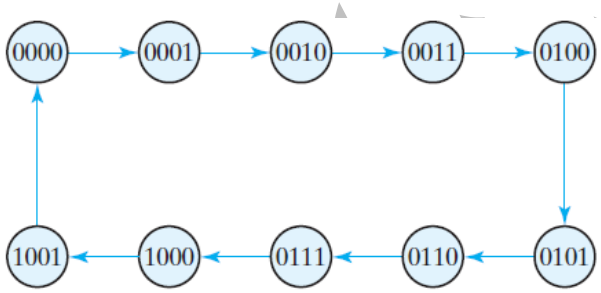


FIGURE 6.9
State diagram of a decimal BCD counter

Q_8	Q_4	Q_2	Q_1	Q_8	Q_4	Q_2	Q_1
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	0	0	0	0

اگر چهار بیت خروجیِ شمارنده را از کم‌ارزش‌ترین به پرارزش‌ترین به ترتیب Q_1، Q_2، Q_4، و Q_8 بنامیم، با توجه به دیاگرام و جدول فوق می‌توان به طور شمّی و اکتشافی قوانین تغییر زیر را استنباط کرد:

ـ حالتِ Q_1 با هر لبه‌ی فعال کلاک متمم می‌شود.

ـ هر بار Q_1 از 1 به 0 برود و Q_8=0 باشد، Q_2 متمم می‌شود. هر گاه Q_8=1 باشد، Q_2 در 0 می‌ماند.

ـ هر بار Q_2 از 1 به 0 برود، Q_4 متمم می‌شود. مادامی که Q_2 یا Q_4 در 0 باشند، Q_8 نیز در 0 خواهد ماند (؟؟).

ـ وقتی هر دو Q_2 و Q_4 برابر 1 شوند، با تغییر 1 به 0 خروجیِ Q_1، خروجیِ Q_8 متمم می‌شود. با گذر بعدیِ Q_1، Q_8 پاک می‌شود.

[1] Ripple BCD counter

با توجه به قوانین فوق، می‌توان مداری به صورت زیر پیشنهاد داد.

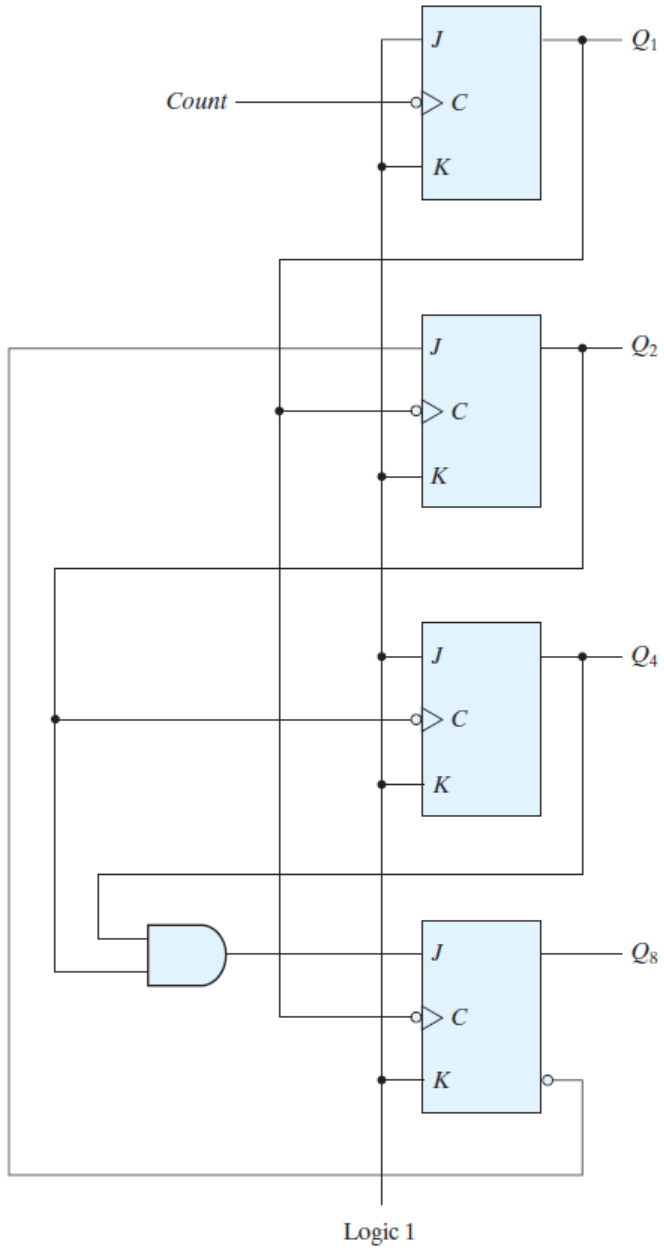


FIGURE 6.10
BCD ripple counter

شمارنده‌ی فوق، یک شمارنده‌ی تک رقمِ BCD است. اگر بخواهیم یک شمارنده‌ی سه رقمیِ BCD داشته باشیم، می‌توانیم از سه شمارنده‌ی تک رقمیِ BCD به صورت زیر در ترکیب با هم استفاده کنیم.

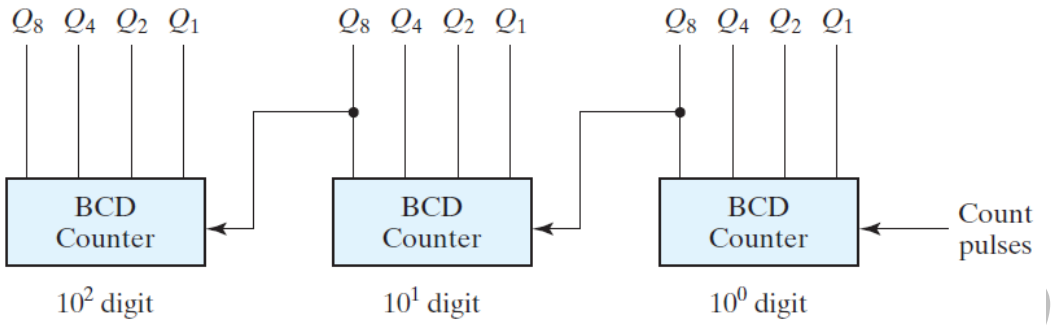


FIGURE 6.11
Block diagram of a three-decade decimal BCD counter

شمارنده‌های همزمان

شمارنده‌ی دودویی

قاعده‌ی شمّی/اکتشافی شمارش دودویی بالاشمار: کم‌ارزش‌ترین بیت، A_0، همواره با هر لبه‌ی فعال کلاک متمم می‌شود. بقیه‌ی بیت‌ها زمانی متمم می‌شوند که تمام بیت‌های قبلی مقدار 1 داشته باشند.

دیاگرام منطقیِ شمارنده‌ی بالاشمار:

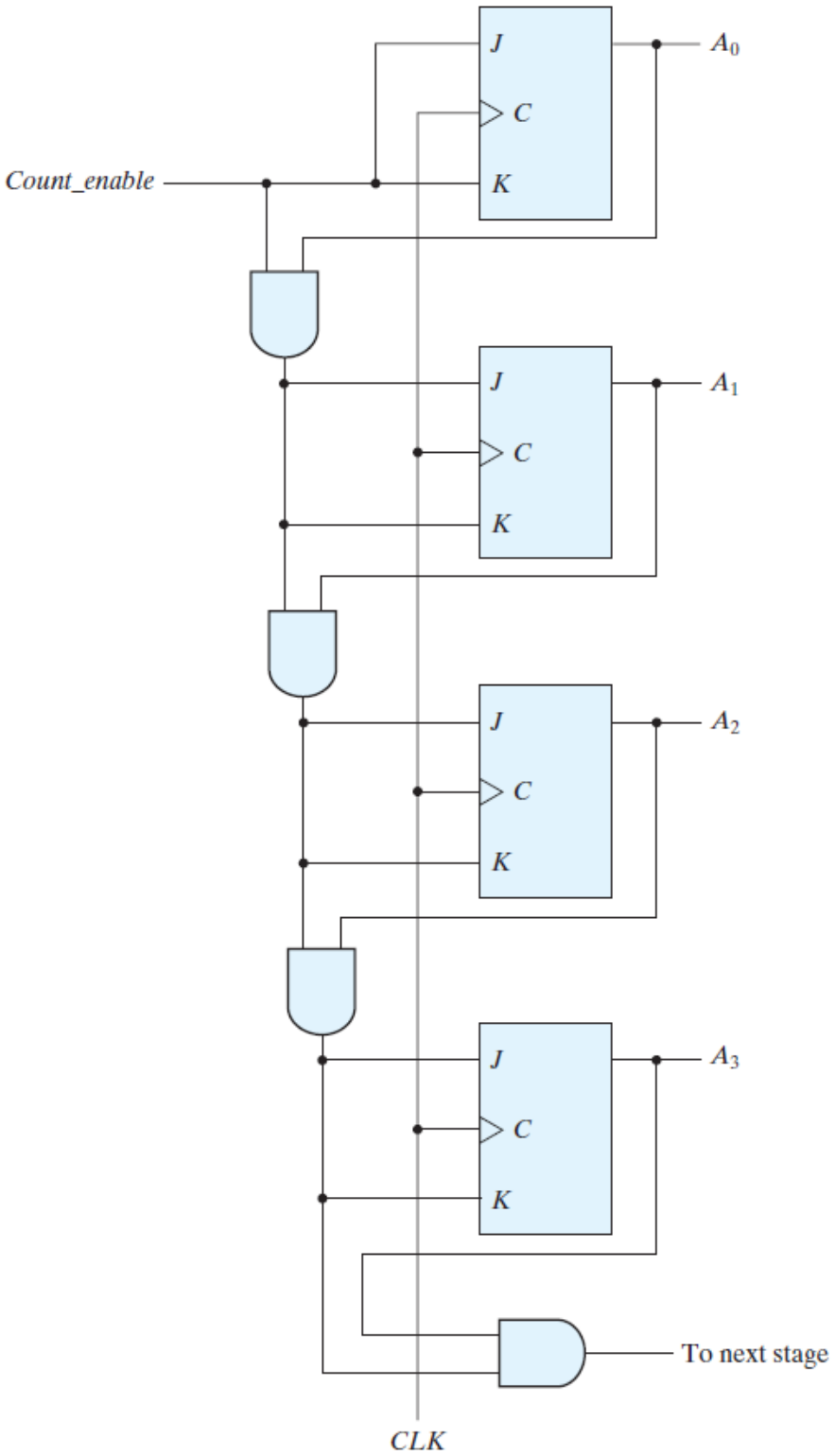


FIGURE 6.12
Four-bit synchronous binary counter

طرح فوق را به راحتی می‌توان تعمیم داد تا شمارنده‌های بزرگتری ایجاد کرد.

برای تبدیل مدار به شمارنده‌ی پایین‌شمار می‌توان از قاعده‌ی مشابهی استفاده کرد با این تفاوت که هر بیت زمانی باید متمم شود که تمام بیت‌های قبلی 0 باشند (همچنان، کم‌ارزش‌ترین بیت با هر لبه‌ی فعال کلاک متمم می‌شود). بنابراین، مدار شمارنده‌ی پایین‌شمار مشابه با مدار شکل ۶ـ۱۲ است با این تفاوت که ورودی هر گیت AND به جای خروجیِ معمولیِ فلیپ فلاپ، از خروجیِ متمم‌ساز آن فلیپ فلاپ گرفته شود.

با ترکیب دو مدار شمارنده‌ی بالاشمار و پایین شمار می‌توان یک شمارنده‌ی بالاـپایین شمار ساخت:

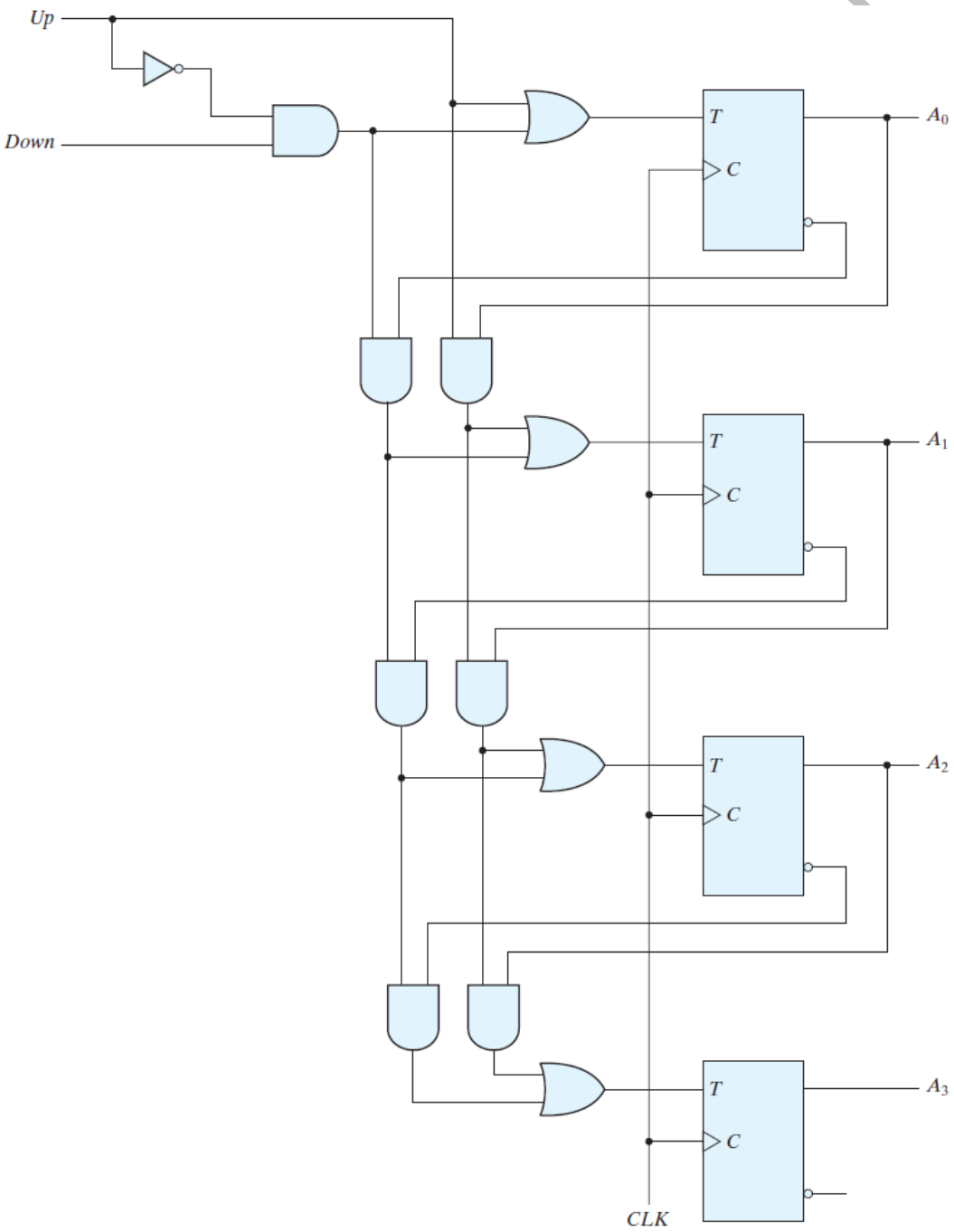


FIGURE 6.13
Four-bit up–down binary counter

شمارنده‌ی BCD

در این جا از روال طراحی گفته شده در فصل قبل، یک شمارنده‌ی BCD با استفاده از TFF طراحی می‌شود.

جدول حالت و معادلات ورودی (بقیه‌ی حالات نشان داده نشده در این جدول، بی‌اهمیت در نظر گرفته می‌شوند):

Table 6.5
State Table for BCD Counter

Present State				Next State				Output	Flip-Flop Inputs			
Q_8	Q_4	Q_2	Q_1	Q_8	Q_4	Q_2	Q_1	y	TQ_8	TQ_4	TQ_2	TQ_1
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	1	0	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	0	1
0	0	1	1	0	1	0	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	0	1
0	1	0	1	0	1	1	0	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	0	1
0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	0	1
1	0	0	1	0	0	0	0	1	1	0	0	1

یک خروجی y برای این منظور در نظر گرفته شده است که گذر از 9 به 0 را نشان دهد. این خروجی در سری کردن چند شمارنده‌ی BCD تک رقمی (مشابه با شکل ۶ـ۱۱ اما با این تفاوت که باید خروجی y را به ورودی شمارش طبقه‌ی بعد متصل کنیم) استفاده می‌شود.

معادلات ورودی فلیپ فلاپ‌ها و معادله‌ی خروجی پس از ساده‌سازی به کمک جدول کارنو:

$$T_{Q1} = 1$$
$$T_{Q2} = Q'_8 Q_1$$
$$T_{Q4} = Q_2 Q_1$$
$$T_{Q8} = Q_8 Q_1 + Q_4 Q_2 Q_1$$
$$y = Q_8 Q_1$$

شمارنده‌ی دودویی با قابلیت بارگذاری موازی

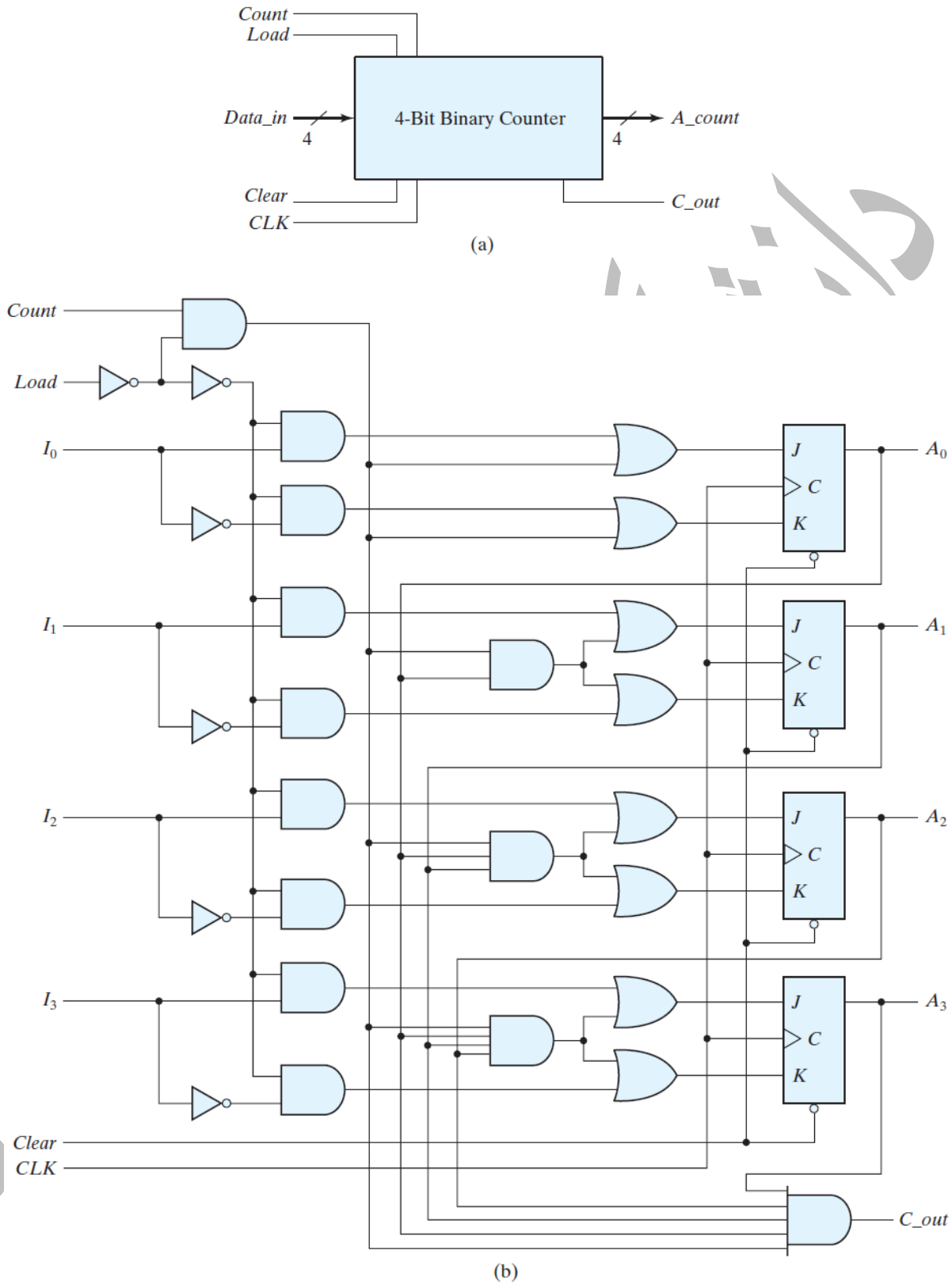


FIGURE 6.14
Four-bit binary counter with parallel load

با استفاده از شمارنده‌ی اخیر می‌توان یک شمارنده‌ی BCD ساخت. دو راه برای انجام این کار عبارتند از:

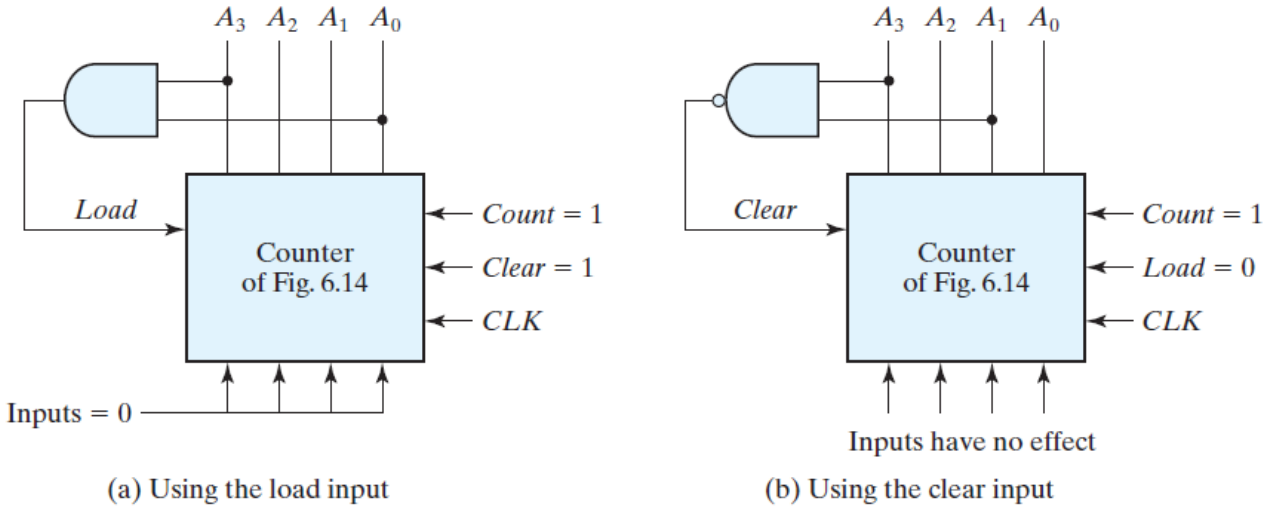


FIGURE 6.15
Two ways to achieve a BCD counter using a counter with parallel load

برخی شمارنده‌های دیگر

شمارنده با حالات بی‌استفاده

ممکن است شمارنده‌ای n ـ بیتی در حالات خود تنها قسمتی از 2^n حالت ممکن را طی کند. در این مواقع سوالی که پیش می‌آید این است که اگر به دلایلی مانند وقوع یک سیگنال خطا، شمارنده وارد یکی از حالات تعریف نشده‌ی خود بشود آیا قادر است پس از یک یا چند پالس بعدیِ کلاک وارد مجموعه‌ی حالات معتبر و تعریف شده‌ی خود شود (یعنی آیا شمارنده خود ـ تصحیح[1] است)؟

در این مواقع یا ما به عنوان طراح، این حالات تعریف نشده و گذر از آنها را از همان ابتدا در جدول حالت پیش‌بینی می‌کنیم تا از خود ـ تصحیح بودن مدار مطمئن شویم. یا این که اگر این مساله را در حین طراحی در نظر نگیریم، باید مدار را آزمایش و بررسی کنیم که آیا خود ـ تصحیح است یا خیر.

مثال: اگر جدول حالت یک شمارنده مبتنی بر JKFF به صورت زیر باشد:

[1] Self-correcting counter

Table 6.7
State Table for Counter

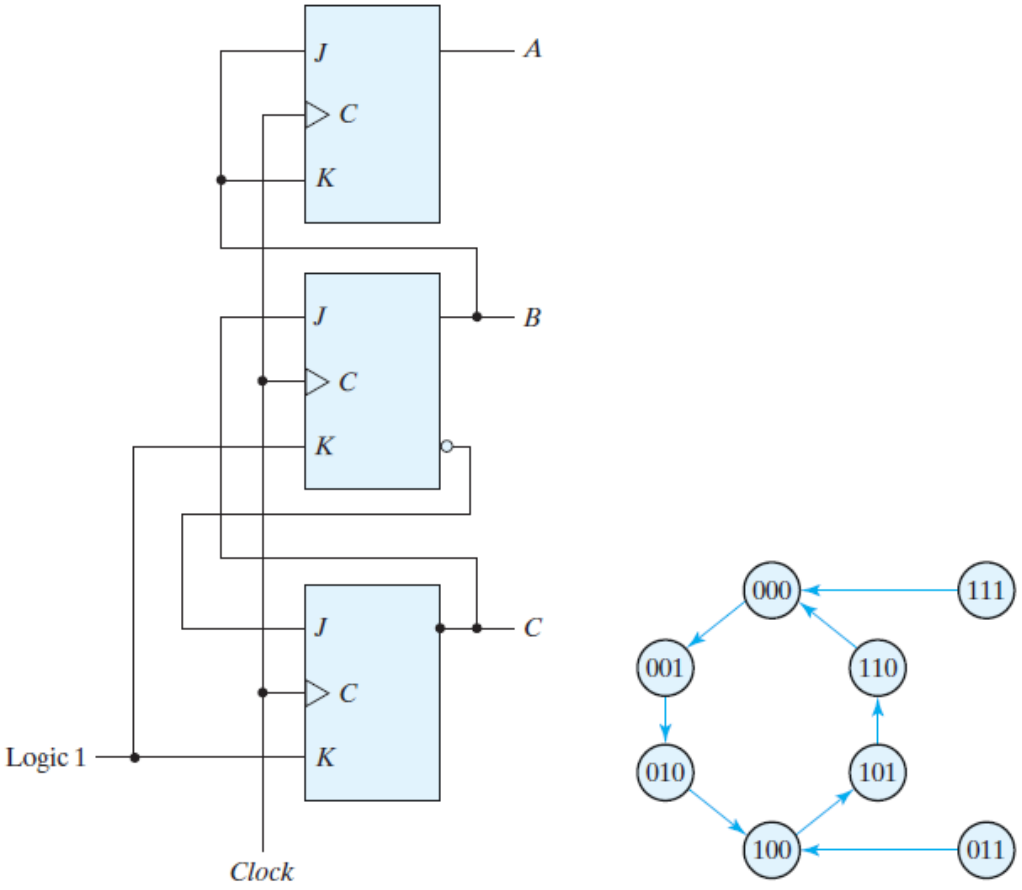
Present State			Next State			Flip-Flop Inputs					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	1	0	0	1	X	X	1	0	X
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	1	1	0	X	0	1	X	X	1
1	1	0	0	0	0	X	1	X	1	0	X

ملاحظه می‌کنید که دو حالت 111 و 011 جزو حالات معتبر و تعریف شده‌ی شمارنده نیستند.

معادلات ساده شده‌ی ورودیِ فلیپ فلاپ‌ها عبارتند از:

$$J_A = B \qquad K_A = B$$
$$J_B = C \qquad K_B = 1$$
$$J_C = B' \qquad K_C = 1$$

و دیاگرام منطقی و دیاگرام حالت شمارنده به صورت زیر می‌شود:



(a) Logic circuit diagram (b) State transition diagram

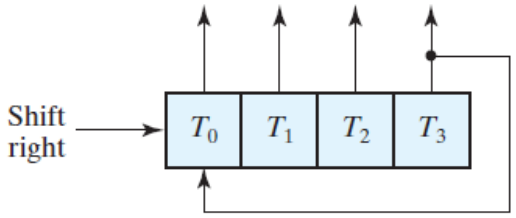
FIGURE 6.16
Counter with unused states

می‌توان بررسی کرد که اگر مدار در یکی از دو حالت به کار نرفته (یا تعریف نشده) قرار داشته باشد، با اولین پالس کلاک، وارد یکی از حالات تعریف شده‌ی خود می‌شود. پس، شمارنده از نوع خود ـ تصحیح است.

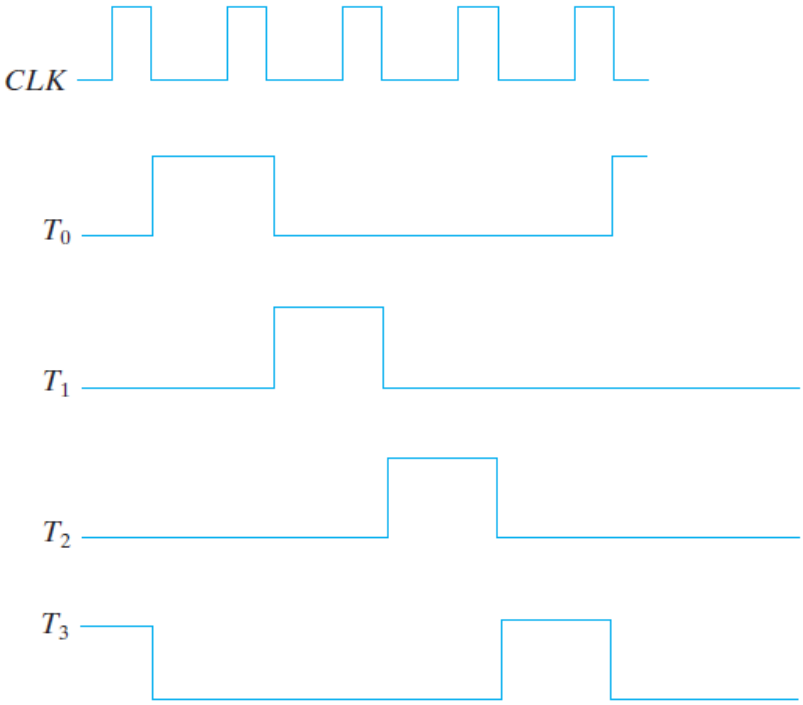
شمارنده‌ی حلقوی (Ring Counter)

شمارنده‌ی حلقوی، شمارنده‌ای است که در خورجی خود تنها یک 1 و بقیه 0 هستند.

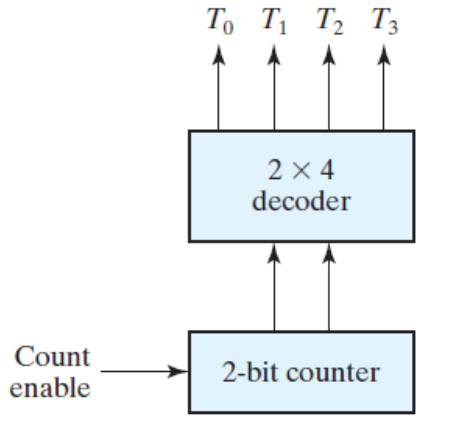
دو راه برای ساخت یک شمارنده‌ی حلقوی، یکی استفاده از شیفت رجیستر (با مقدار اولیه‌ی خاص) و دیگری استفاده از یک شمارنده‌ی دودیی و یک دیکدر است.



(a) Ring-counter (initial value = 1000)



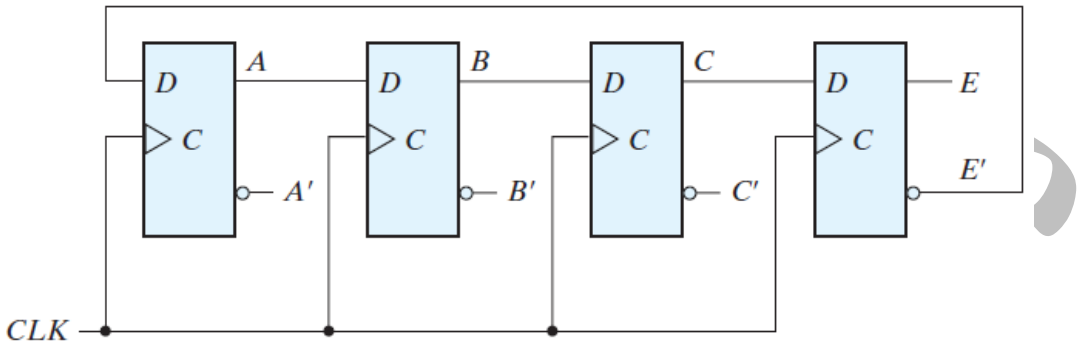
(b) Sequence of four timing signals



(c) Counter and decoder

FIGURE 6.17
Generation of timing signals

شمارنده‌ی حلقوی دنباله‌چرخان[1]



(a) Four-stage switch-tail ring counter

شمارنده‌ی جانسون

شمارنده‌ی جانسون یک شمارنده‌ی حلقوی دنباله‌چرخان است که به کمک تعدادی گیت AND خروجی‌ای مشابه با خروجیِ شمارنده‌ی حلقوی تولید می‌کند. تفاوت شمارنده‌ی جانسون با شمارنده‌ی حلقوی در این است که شمارنده‌ی حلقوی با داشتن k فلیپ فلاپ تنها قادر به تولید k حالت مختلف است اما شمارنده‌ی جانسون به کمک k فلیپ فلاپ و 2k گیت AND قادر به تولید 2k حالت مختلف است.

Sequence number	Flip-flop outputs				AND gate required for output
	A	B	C	E	
1	0	0	0	0	$A'E'$
2	1	0	0	0	AB'
3	1	1	0	0	BC'
4	1	1	1	0	CE'
5	1	1	1	1	AE
6	0	1	1	1	$A'B$
7	0	0	1	1	$B'C$
8	0	0	0	1	$C'E$

(b) Count sequence and required decoding

FIGURE 6.18
Construction of a Johnson counter

[1] Switch-tail ring counter

البته مشکل شمارنده‌ی جانسون این است که اگر مدار وارد یکی از حالات بلااستفاده (یا همان تعریف نشده‌ی) خود بشود، دیگر هیچ گاه به مجموعه‌ی حالات معتبر و تعریف شده‌ی خود برنخواهد گشت. یک راه برای حل این مشکل این است که ورودیِ فلیپ فلاپِ C را به صورت زیر تصحیح کنیم:

$$D_C = (A + C)B$$