

6.Yazmaçlar (Registers), Sayıcılar (Counters)

6.1.Yazmaçlar

- Paralel Yüklemeli Yazmaçlar
- Ötelemeli Yazmaçlar

6.2.Sayıcılar

- İkili Asenkron Sayıcılar (Binary Ripple Counter)
- İkili Kodlanmış Onlu Asenkron Sayıcı (Bcd Binary Coded Decimal Ripple Counter)
- İkili Senkron Sayıcılar
- Paralel Yüklemeli İkili Senkron Sayıcılar
- Sayıcıların Uygulaması Olarak, Zamanlama Dizileri Üreten Devreler
 - Kelime-Zamanı Darbesinin Üretilmesi
 - Sayısal Sistemlerde Zaman İşaretleri
 - Johnson Sayıcılar Ve 8 Li Zaman İşareti Üreteçleri

1

6.1 Yazmaç (Register)

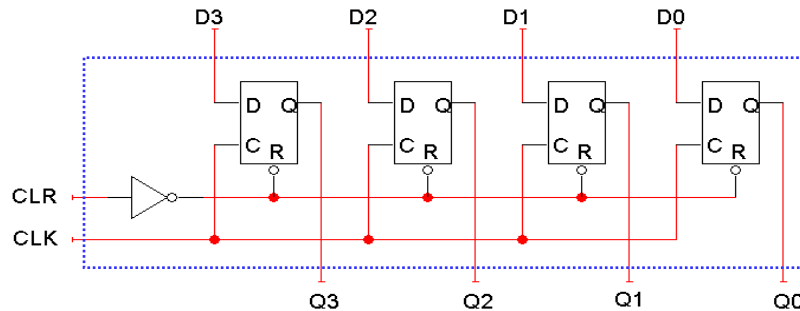
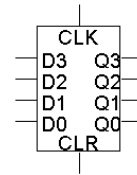
- Genel bir ardışıl devre: yazmaçlar
 - Ardışıl devre analiz ve sentezi için iyi bir örnektir.
 - Ayrıca daha büyük çaplı ardışıl devrelerin tasarımında kullanılabilirler.
- Flip-flop'lar tek bir bit tutarken **yazmaçlar** daha büyük miktarda veri tutabilmektedir.
 - yazmaçlar modern işlemci tasarımının merkezidir.
 - Birden fazla çeşitte yazmaçlar vardır.

yazmaçların avantajları

- Flip-flop'lar sadece tek bir bit tutabilirler.
 - Örneğin, iki bitlik sayıcı tasarımı için iki tane flip-flop kullanmak zorundayız.
 - Pekçok bilgisayar "integer" lar ve "single-precision floating-point number" (32 bitlik kesirli sayı: 32 = 1 (i: işaret) + 8 (ü: üs) + 23 (k: kesir)) lar ile çalışır ve bunlar da 32-bit uzunluğundadır.
 $(-1)^i 2^{u-127} \times (1.k)$
 (double-precision floating-point number: 1+11+52)
- Bir yazmaç (register) bir flip-flop'un birden fazla bit (tutacak) saklayacak halde geliştirilmiş halidir.
- Yazmaçlar genellikle işlemcilerde geçici saklama işi olarak kullanılır (temporary storage).
 - Ana hafızaya göre daha hızlı ve daha elverişlidir.
 - Ayrıca yazmaçlar kompleks hesaplamaları hızlandırır.

Temel bir yazmaç

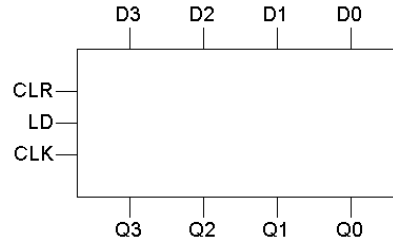
- Temel yazmaçların tasarımı oldukça basittir. Birkaç tane flip-flop'u birlikte bağlayarak birden fazla biti saklayabiliriz!
- Bir 4-bit yazmaç aşağıda verilmiştir.
 - Bu yazmaçta D FF lar kullanılmıştır. Böylece FF giriş denklemleri ile uğraşmadan veriyi tutabiliriz.
 - Tüm flip-flop'ların CLK ve CLR sinyali ortaktır.



Paralel yükleme işlemi ekleme

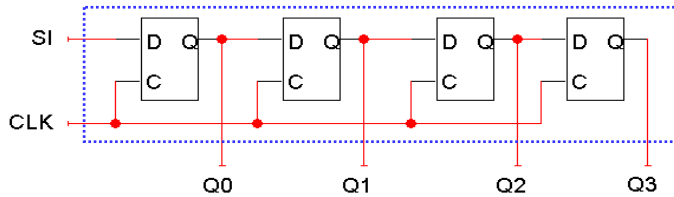
- D_3-D_0 girişleri her bir saat periyotunda Q_3-Q_0 çıkışlarına kopyalanır.
- Şimdiki değerleri birden fazla periyot için tutmak istersek ne yapmalıyız?
- yazmaca bir LD yükleme girişi eklenebilir.
 - Eğer $LD = 0$ ise, yazmaç şimdiki değerlerini tutar.
 - Eğer $LD = 1$ ise, yazmaç D_3-D_0 girişlerinden yeni bir değer alır.

LD	$Q(t+1)$
0	$Q(t)$
1	D_3-D_0



Ötemeli yazmaç (Shift Register)

- Bir ötemeli yazmaç çıkışını her bir saat periyotunda "öteler".



$$\begin{aligned} Q_0(t+1) &= SI \\ Q_1(t+1) &= Q_0(t) \\ Q_2(t+1) &= Q_1(t) \\ Q_3(t+1) &= Q_2(t) \end{aligned}$$

- SI girişi yazmaca ötelenecek yeni biti sağlayan giriştir.
- Örneğin, bir saat pozitif yükselen kenarlarında:

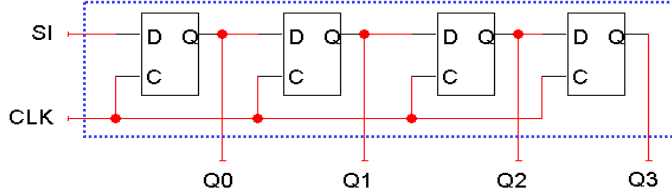
$$\begin{aligned} SI &= 1 \\ Q_0-Q_3 &= 0110 \end{aligned}$$

ise gelecek durum aşağıdaki gibi olacaktır:

$$Q_0-Q_3 = 1011$$

- Şimdiki Q_3 (bu örnek için 0) gelecek periyotta kaybolacaktır.

Öteleme yönü



$$\begin{aligned} Q_0(t+1) &= SI \\ Q_1(t+1) &= Q_0(t) \\ Q_2(t+1) &= Q_1(t) \\ Q_3(t+1) &= Q_2(t) \end{aligned}$$

- Çizilen devre ve verilen örnek yazmaç "sağa" öteliyormuş gibi gösteriyor.

Şimdiki Q_0-Q_3	SI	Gel. Q_0-Q_3
ABCD	X	XABC

- Aslında bizim bitleri nasıl yorumladığımıza bağlıdır. Eğer biz Q_3 'ü en önemli bit olarak düşünersek, böylece yazmaç ters yöne doğru öteleme yapmış olur!

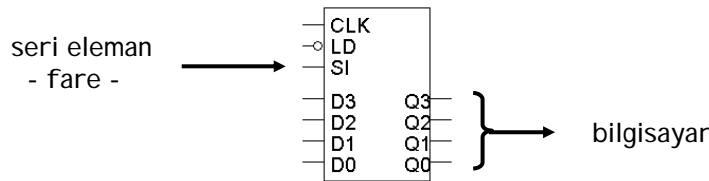
Şimdiki Q_3-Q_0	SI	Gel. Q_3-Q_0
DCBA	X	CBAX

Seri veri alımı/gönderimi (Serial data transfer)

- Ötelemeli yazmaçlar için bir uygulama "seri veri" ve "paralel veri" arasındaki dönüşümdür.
- Genel olarak bilgisayarlar birden fazla bitlik niceliklerle çalışır.
 - ASCII text karakterleri 8 bit uzunluğundadır.
 - "Integer" lar, "single-precision floating-point number" lar, ve "screen pixel" leri 32 bit uzunluğa kadar çıkar.
- Ancak bazen **seri** olarak veri göndermek veya almak gerekir yani tek bir zamanda tek bir bit. Bazı örnekler:
 - Giriş elemanları (örneğin, klavye ve fare)
 - Çıkış elemanları (örneğin, yazıcılar)
 - Herhangi bir seri port, USB veya Firewire elemanları veriyi seri olarak transfer eder.

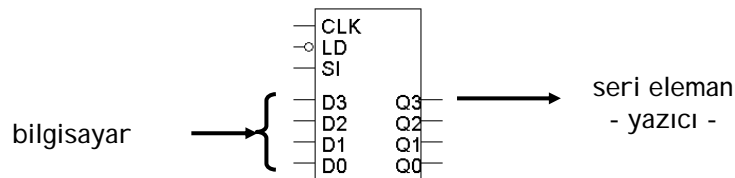
Seri data alımı (Receiving serial data)

- Ötemeli yazmaç kullanarak seri data *alımı* için:
 - Seri eleman yazmacın SI girişine bağlanır.
 - Ötemeli yazmacın Q3-Q0 çıkışları bilgisayara bağlanır.
- Seri eleman her bir saat periyotunda bir bit iletir.
 - Bu bitler ötemeli yazmacın SI girişine gider.
 - Dört saat periyodu sonunda, dört bitlik kelimenin tamamı ötelemeli yazmaçta tutulmuş olur.
- Ardından, bilgisayar tüm bu dört biti bir seferde Q3-Q0 çıkışlarından okur.



Seri data gönderimi (Sending serial data)

- Ötemeli yazmacı kullanarak seri data *gönderimi* için ters işlem yapmalıyız:
 - Bilgisayar (CPU) yazmacın D girişlerine bağlanır.
 - Öteleme çıkışı (bu durumda Q3) seri elemana bağlanır.
- Öncelikle bilgisayar bir saat periyotunda dört bitlik bilgiyi yazmaca gönderir.
- Ardından seri eleman öteleme çıkışını okumaya başlar.
 - Her bir saat periyotunda Q3'de bir bit görünür.
 - Dört periyot sonunda, dört bitlik kelimenin tamamı seri elemana gönderilmiş olur.



Yazmaçlar - Özet

- Yazmaçlar birden fazla biti tutabilen durum elemanlarıdır.
- Birden fazla uygulama alanı vardır:
 - Yazmaçta data tutabilmek için paralel yükleme
 - Yazmacın içeriğini sağa veya sola öteleme
 - Sayıcılar da bir tip yazmaçtır!
- Ötelemeli yazmaçların bir uygulaması seri ve paralel veri arasında dönüşümü sağlamaktır.
- Yazmaçlar bilgisayarda (CPU) data tutar.
 - Aritmetik işlemleri yapan birime (ALU) bilgi aktarımında kullanılır.
 - Cevapları tutmada kullanılır.
- Pek çok program yazmaçların sağladığından daha fazla saklama yerine ihtiyaç duymaktadır.
 - Bunun çözümü için bir sonraki dersimizde RAM'leri tanıyacağız.

(Not: "Saklayıcılar varken neden RAM'lerle uğraşıyoruz?" sorusunun cevabı: Saklayıcılar pahalıdır!)

6.2.Sayıcılar (Counters)

- Sayıcılar n bitlik bir bilgiyi tutmanın yanısıra her saat çevriminde tuttıkları değeri artıran veya azaltan ardışıl devrelerdir.
- Genel olarak iki gruba ayrılır:
 - Senkron sayıcılar
 - Asenkron sayıcılar (Ripple sayıcılar)
- Yaptığı işe göre oldukça fazla kullanılan sayıcılar standart hale gelmiştir ve hazır devre olarak piyasada bulunmaktadır. Biz bu derste birkaç çeşit sayıcı tasarlayacağız. Ayrıca, senkron ve asenkron sayıcılar arasındaki farkları inceleyeceğiz.

Senkron Sayıcılar

- Senkron sayıcılarda tüm flip-flop'lara uygulanan ortak bir saat (clock) devresi yer almaktadır. Böylece tüm flip-flop'lar senkronize bir şekilde saatin her bir pozitif kenarında tetiklenmekte ve çıkış üretmektedirler.

- Örnek:** 0'dan 7'ye kadar sayan bir binary sayıcı tasarlayalım. 000-111 arasında sayacağından 3 digit gerekmektedir. Dolayısıyla da 3 tane flip-flop kullanılacaktır.

Durum Tablosu →

Şimdiki Durum			Gelecek Durum		
Q ₂	Q ₁	Q ₀	Q ₂	Q ₁	Q ₀
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

J₀

		Q ₁ Q ₀	
		0	1
Q ₂	0	1	0
	1	0	1

J₁

		Q ₁ Q ₀	
		0	1
Q ₂	0	0	1
	1	0	0

J₂

		Q ₁ Q ₀	
		0	1
Q ₂	0	0	0
	1	0	0

K₀

		Q ₁ Q ₀	
		0	1
Q ₂	0	0	1
	1	0	1

K₁

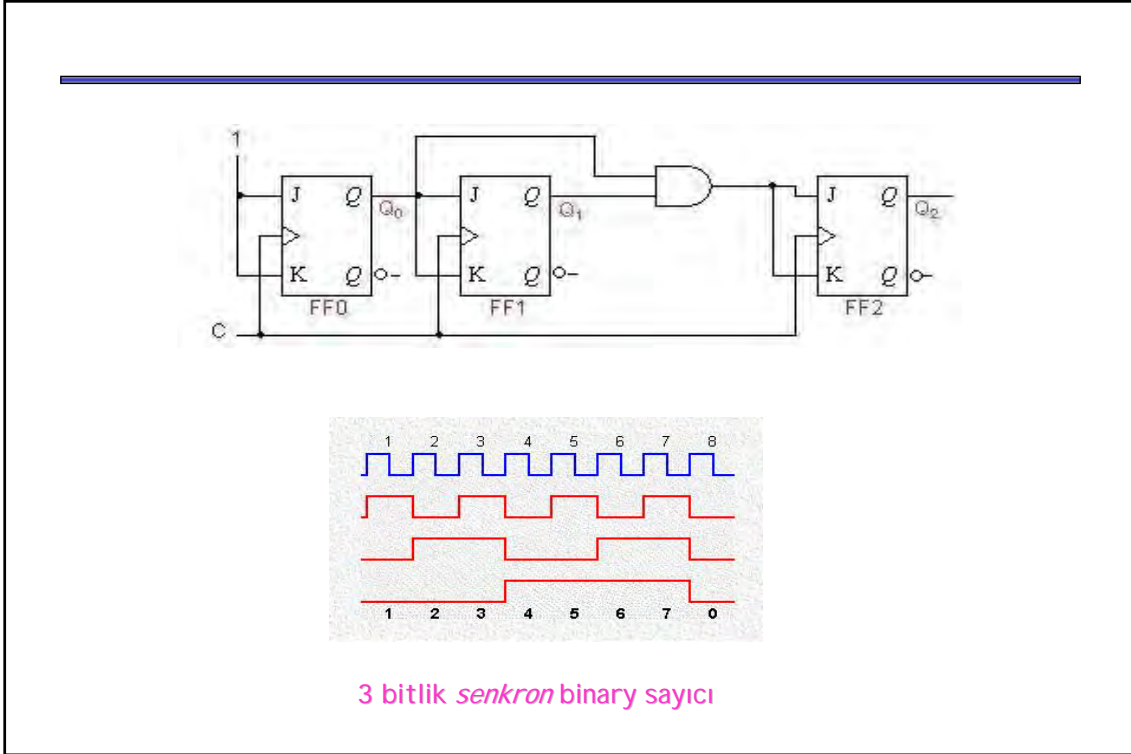
		Q ₁ Q ₀	
		0	1
Q ₂	0	0	0
	1	0	0

K₂

		Q ₁ Q ₀	
		0	1
Q ₂	0	0	0
	1	0	0

	J0	J1	J2																																						
Q ₂	<table border="1"> <tr><td colspan="4">Q₁Q₀</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td></tr> </table>	Q ₁ Q ₀				1	0	0	1	1	0	0	1	<table border="1"> <tr><td colspan="4">Q₁Q₀</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td></tr> </table>	Q ₁ Q ₀				0	1	0	0	0	1	0	0	<table border="1"> <tr><td colspan="4">Q₁Q₀</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td></tr> </table>	Q ₁ Q ₀				0	0	1	0	0	0	1	0		
Q ₁ Q ₀																																									
1	0	0	1																																						
1	0	0	1																																						
Q ₁ Q ₀																																									
0	1	0	0																																						
0	1	0	0																																						
Q ₁ Q ₀																																									
0	0	1	0																																						
0	0	1	0																																						
K0	<table border="1"> <tr><td colspan="4">Q₁Q₀</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td></tr> </table>	Q ₁ Q ₀				0	1	1	0	0	1	1	0	K1	<table border="1"> <tr><td colspan="4">Q₁Q₀</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td></tr> </table>	Q ₁ Q ₀				0	0	1	0	0	0	1	0	K2	<table border="1"> <tr><td colspan="4">Q₁Q₀</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td></tr> </table>	Q ₁ Q ₀				0	0	0	0	0	0	1	0
Q ₁ Q ₀																																									
0	1	1	0																																						
0	1	1	0																																						
Q ₁ Q ₀																																									
0	0	1	0																																						
0	0	1	0																																						
Q ₁ Q ₀																																									
0	0	0	0																																						
0	0	1	0																																						
	J0 = 1 K0 = 1	J1 = Q0 K1 = Q0	J2 = Q1Q0 K2 = Q1Q0																																						

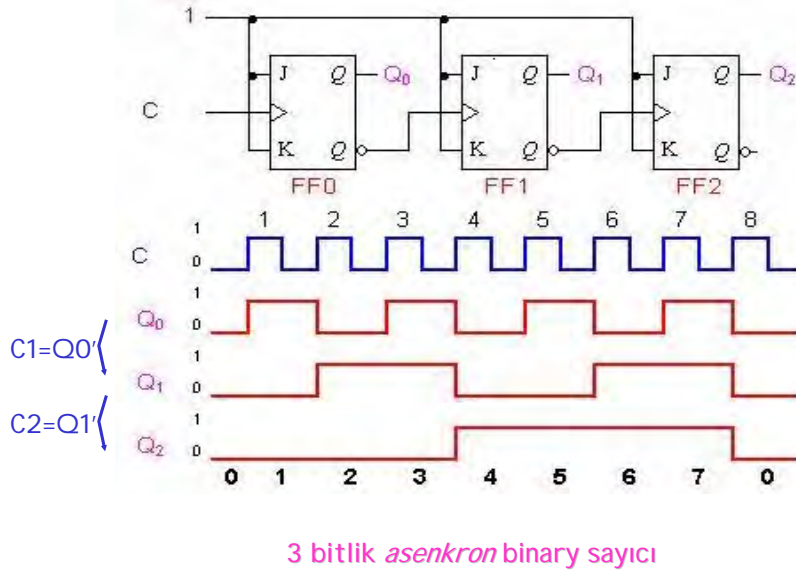
<u>FF0:</u>	J0 = 1 K0 = 1	→	Q(t+1)=Q'(t)
<u>FF1:</u>	J1 = Q0 K1 = Q0	→	Q0=0 ise, J=K=0, Q(t+1)=Q(t) Q0=1 ise, J=K=1, Q(t+1)=Q'(t)
<u>FF2:</u>	J2 = Q1Q0 K2 = Q1Q0	→	Q0=0 veya Q1=0 veya Q0=Q1=0 ise, J=K=0, Q(t+1)=Q(t) Q0=Q1=1 ise, J=K=1, Q(t+1)=Q'(t)



Asenkron Sayıcılar (Ripple Sayıcılar)

- Senkron sayıcıların sentezi için bildiğimiz ardışıl devre sentezi yöntemi uygulanmasına rağmen asenkron sayıcılar için adım adım izlenebilecek geliştirilmiş bir yöntem yoktur. Asenkron sayıcıların sentezi sezgisel olarak yapılır.
- Senkron sayıcılarda tüm flip-flop'lara uygulanan ortak saat girişi asenkron sayıcılarda sadece sayının en önemsiz bitine karşılık gelen flip-flop'un saat girişine uygulanır. Diğer flip-flop'ların saat girişine ise bir önceki flip-flop'un çıkışının tümleyeni uygulanır.
- Asenkron sayıcıların önemli bir dezavantajı: hızları ardarda dizilen flip-flop'ların her birinden gelen yayılma zamanı ile sınırlıdır. Senkron sayıcılarda ise hızı saatin periyotunu ayarlayarak biz belirleriz.

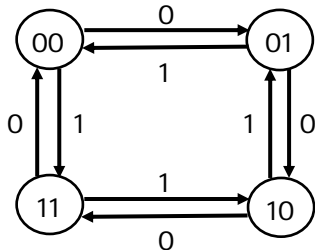
• Örnek:



Senkron Yukarı/Aşağı Sayıcı

- 2-bitlik senkron binary yukarı/aşağı sayıcı
 - Sayıcı çıkışları 00, 01, 10 ve 11 olacak.
 - 1 tane giriş var: X.
 - > X= 0 ise, sayıcı yukarı doğru sayacak
 - > X= 1 ise, sayıcı aşağı doğru sayacak
- İki tane flip-flop'a ihtiyaç var.

Durum diyagramı ve durum tablosu:

Not: g/? çıkış üretilmiyor...

Şimdiki durum		Giriş	Gelecek durum	
Q ₁	Q ₀	X	Q ₁	Q ₀
0	0	0	0	1
0	0	1	1	1
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	0
1	1	1	1	0

- Eğer D FF kullanırsak:
(D girişleri doğrudan gelecek duruma eşit olur.)

Şimdiki Durum		Giriş	Gelecek Durum	
Q ₁	Q ₀	X	Q ₁	Q ₀
0	0	0	0	1
0	0	1	1	1
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	0
1	1	1	1	0

		Q ₀ X	
Q ₁		0	1
		0	1
		1	0
		1	0

$$D_1 = Q_1 \oplus Q_0 \oplus X$$

		Q ₀ X	
Q ₁		1	1
		1	1
		0	0
		0	0

$$D_0 = Q_0'$$

- Eğer JK FF kullanırsak:

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

Şimdiki Durum		Giriş	Gel. Durum		Flip flop girişleri			
Q ₁	Q ₀	X	Q ₁	Q ₀	J ₁	K ₁	J ₀	K ₀
0	0	0	0	1	0	∅	1	∅
0	0	1	1	1	1	∅	1	∅
0	1	0	1	0	1	∅	∅	1
0	1	1	0	0	0	∅	∅	1
1	0	0	1	1	∅	0	1	∅
1	0	1	0	1	∅	1	1	∅
1	1	0	0	0	∅	1	∅	1
1	1	1	1	0	∅	0	∅	1

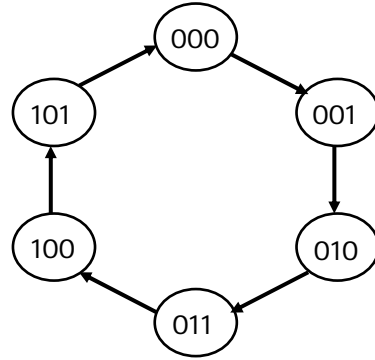
$$J_1 = K_1 = Q_0' X + Q_0 X'$$

$$J_0 = K_0 = 1$$

0-5 sayan sayıcı için 6 ve 7 ne olacak??

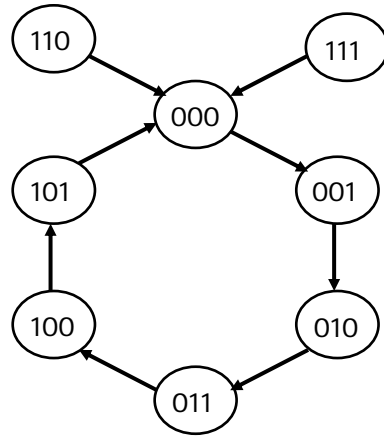
- Eğer 0'dan 5'e sayan bir binary sayıcı tasarlırsak iki tane durum (110 ve 111) ne olacak??
- don't cares* **OR** *do care...* :)

Şimdiki Durum			Gelecek Durum		
Q ₂	Q ₁	Q ₀	Q ₂	Q ₁	Q ₀
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	0	0	0
1	1	0	Ø	Ø	Ø
1	1	1	Ø	Ø	Ø



- Daha *güvenilir* bir devre elde etmek için bu kullanılmayan durumları don't care yerine 0 yazabiliriz.
- Böylece bir şekilde devre bu kullanılmayan durumlara girse bile anlamlı bir sonuç olacaktır.
- Bu bir **kendi-başlayan sayıcı**dır.

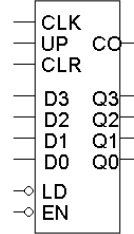
Şimdiki Durum			Gelecek Durum		
Q ₂	Q ₁	Q ₀	Q ₂	Q ₁	Q ₀
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	0	0	0



Daha kapsamlı sayıcılar

- Anlatılanlara göre daha kapsamlı sayıcılar da tasarlanabilir veya standart olarak hazırda bulunabilir:

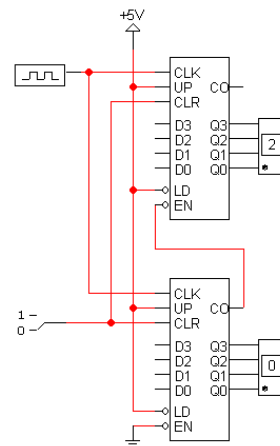
Bu bir 4 bitlik standart sayıcı →



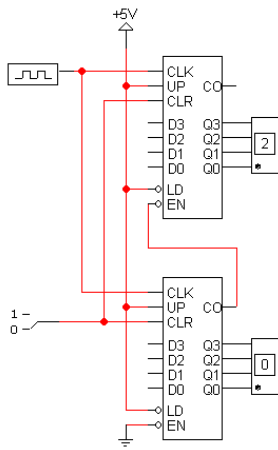
- UP girişinin 1 veya 0 olma durumuna göre yukarı veya aşağı sayar.
- CLR = 1 yapılarak sayıcı hemen sıfırlanabilir (0000).
- D₃-D₀ 'a istenilen 4 bit değer set edilerek ve LD=0 yapılarak, bir sonraki çıkış belirlenebilir.
- Aktive-Düşük (Active-Low) EN girişi yardımıyla sayıcı enable veya disable yapılabilir.
Sayıcı disable (EN=1) olduğunda çıkışındaki sayı aynen korunur (artmaz, azalmaz, yüklenmez veya silinmez).
- CO "Counter Out" çıkışı normalde 1'dir. Sayıcı maksimum değere (1111) ulaştığında ise 0 olur.

8-bitlik sayıcı

- Bu iki tane 4-bitlik sayıcıdan oluşturulmuş bir 8-bitlik sayıcıdır.
 - Altındaki sayıcı en önemsiz dört bite aittir. Yukarıdaki ise en önemli dört bite aittir.
 - Aşağıdaki sayıcı 1111'e ulaştığında (yani, CO = 0 olduğunda), yukarıdaki 1 saat periyodu için aktif (enable) yapar.
- Not:
 - "Clock" ve "clear" girişleri ortaktır.
 - Hexadecimal göstergeler kullanılır.

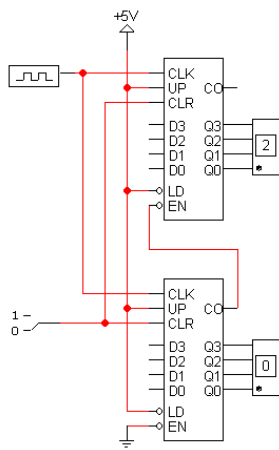


8-bitlik sayıcı



0000 0000	0001 0001	0001 0001
0000 0001	0001 0010	0001 0010
0000 0010	0001 0011	0001 0011
0000 0011	0001 0100	0001 0100
0000 0100	0001 0101	0001 0101
0000 0101	0001 0110	0001 0110
0000 0110	0001 0111	0001 0111
0000 0111	0001 1000	0001 1000
0000 1000	0001 1001	0001 1001
0000 1001	0001 1010	0001 1010
0000 1010	0001 1011	0001 1011
0000 1011	0001 1100	0001 1100
0000 1100	0001 1101	0001 1101
0000 1101	0001 1110	0001 1110
0000 1110	0001 1111	0001 1111
0000 1111	0010 0000	0011 0000
0001 0000		

8-bitlik sayıcı



0000 0000	→ CO=1 ve EN=1, FF1 disable
0000 0001	
⋮	
0000 1110	
0000 1111	→ CO=0 ve EN=0, FF1 enable
0001 0000	→ CO=1 ve EN=1, FF1 disable
0001 0001	
⋮	
0001 1110	
0001 1111	→ CO=1 ve EN=1, FF1 disable
0010 0000	→ CO=0 ve EN=0, FF1 enable
0001 0001	
⋮	
0001 1110	
0001 1111	
0011 0000	