EEM212 - SAYISAL DEVRE TASARIMI DERS NOTLARI

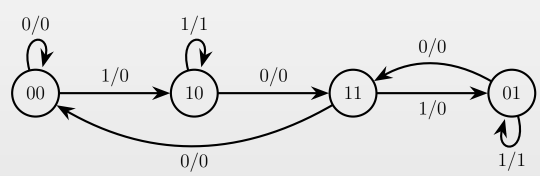
DERS NOTU 9: ÇÖZÜMLÜ ÖRNEKLER I

Dr. İsmail Öztürk *

<ismail.ozturk@amasya.edu.tr>

(**ver. 1.1**)

Örnek 0.1:



Yukarıdaki sonlu durum makinesini 3×8 *kod çözücü (decoder), D FF, VEYA kapıları kullanarak tasarlayınız.*

—

Durumları AB ile; girişi x ile ve çıkışı y ile gösterdiğimizde durum diyagramına bakarak durum tablosunu aşağıdaki gibi oluştururuz:

Şimdiki Durum		**Giriş**	**Sonraki Durum**		**Çıkış**
A	B	x	A	B	y
0	0	0	0	0	0
0	0	1	1	0	0
0	1	0	1	1	0
0	1	1	0	1	1

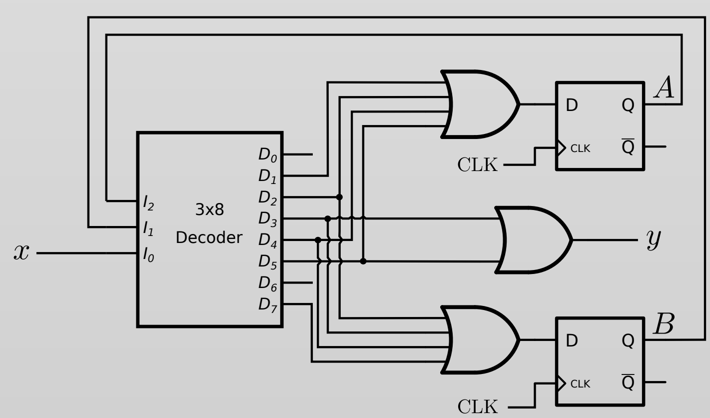
*Amasya Üniversitesi Teknoloji Fakültesi EEM Bölümü
Daha fazla bilgi için: https://iozturk.com

1	0	0	1	1	0
1	0	1	1	0	1
1	1	0	0	0	0
1	1	1	0	1	0

Bu noktadan sonra D FF girişlerine bağlanması gereken kombinasyonel devreler ile çıkışın kombinasyonel devresini tasarlamamız gerekir. Hatırlayacağınız üzere 3×8 bir kod çözücünün çıkışlarının her biri bir miniterime karşılık gelmekteydi. Tasarımda kod çözücü kullanacağımız için kombinasyonel devreleri miniterimlerin toplamı formunda ifade etmemiz gerekir. Buna uygun olarak FF girişlerine bağlanması gereken kombinasyonel devreler ile çıkış devresini miniterimlerin toplamı cinsinden aşağıdaki gibi elde ederiz:

$$
\begin{aligned}
D_A(A,B,x) &= \sum(1,2,4,5) = m_1 + m_2 + m_4 + m_5 \\
D_B(A,B,x) &= \sum(2,3,4,7) = m_2 + m_3 + m_4 + m_7 \\
y(A,B,x) &= \sum(3,5) = m_3 + m_5
\end{aligned}
$$

3×8 kod çözücünün çıkışları sırasıyla $m_0, m_1, \ldots, m_7$ olduğu için FF girişlerine bağlanması gereken kombinasyonel devreleri ilgili kod çözücü çıkışlarını VEYA kapılarına sokarak elde ederiz. Benzer şekilde y çıkışını elde etmek için tek yapmamız gereken kod çözücünün 3. ve 5. çıkışlarını VEYA kapısına bağlamaktır. Bunları yaptığımızda aşağıdaki devreyi elde etmiş oluruz:



Kod çözücü girişlerini bağlarken A'nın MSB ve x'in LSB olmasına dikkat etmeniz gerekir!

Ardışıl devrelerin kombinasyonel devrelerini mux kullanarak da tasarlayabiliriz. Bunun için kombinasyonel devre tasarımı ile alakalı ders notlarına bakınız.

Örnek 0.2:

$0-7-2-5-0$ *şeklinde sayan bir sayıcıyı JK FF kullanarak en sade haliyle tasarlayınız. Sadeleştirme yaparken maksiterimleri kullanınız ve kullanılmayan durumları* 0 *durumuna yönlendiriniz.*

—

Sayılar 0 ile 7 arasında değiştiğinden üç tane FF kullanılması gerekir. JK FF uyarım tablosunu kullanarak (önceki notlara bakın) devrenin durum tablosunu aşağıdaki gibi doldururuz:

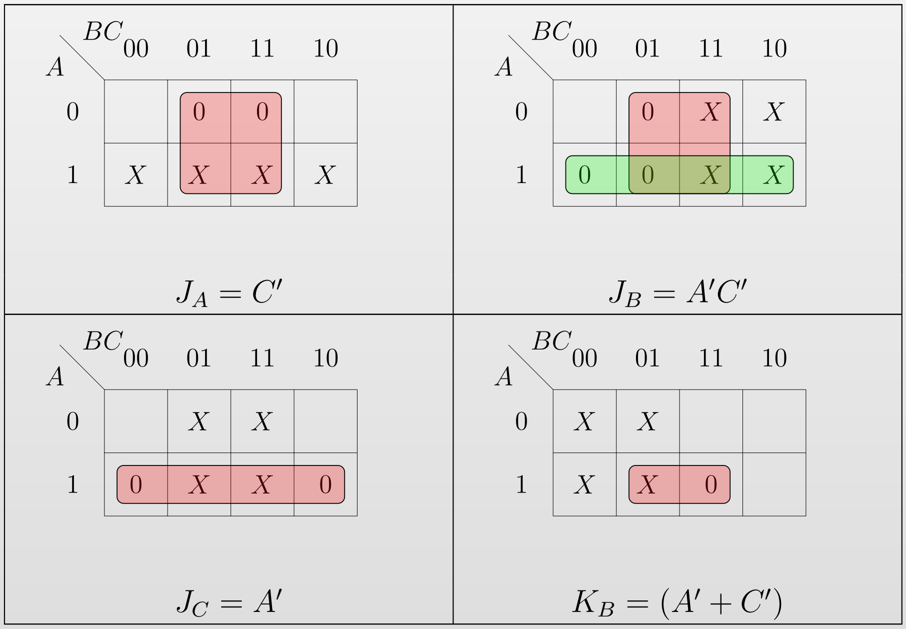
Şimdiki Durum			**Sonraki Durum**			**Flip Flop Girişleri**					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	1	1	1	1	X	1	X	1	X
0	0	1	0	0	0	0	X	0	X	X	1
0	1	0	1	0	1	1	X	X	1	1	X
0	1	1	0	0	0	0	X	X	1	X	1
1	0	0	0	0	0	X	1	0	X	0	X
1	0	1	0	0	0	X	1	0	X	X	1
1	1	0	0	0	0	X	1	X	1	0	X
1	1	1	0	1	0	X	1	X	0	X	1

Buradan FF girişlerine bağlanması gereken Boole denklemlerini maksiterimlerin çarpımı cinsinden aşağıdaki gibi elde ederiz:

$$J_A(A,B,C) = \prod(1,3) \cdot d(4,5,6,7)$$
$$K_A(A,B,C) = d(0,1,2,3) = 1$$
$$J_B(A,B,C) = \prod(1,4,5) \cdot d(2,3,6,7)$$
$$K_B(A,B,C) = \prod(7) \cdot d(0,1,4,5)$$
$$J_C(A,B,C) = \prod(4,6) \cdot d(1,3,5,7)$$
$$K_C(A,B,C) = d(0,2,4,6) = 1$$

Burada $\prod(1,3) \cdot d(4,5,6,7)$ ifadesi $\prod(1,3) = M_1M_3$ maksiterimleri ile $d(4,5,6,7)$ fark etmez durumlarını belirtmek için kullanılan notasyondur. Dikkat ederseniz K_A ile K_C ifadeleri herhangi bir maksiterim içermemektedir. Dolayısıyla, bu ifadeler için fark etmez durumları 1 alınır ve sırasıyla $K_A = 1$, $K_C = 1$ elde ederiz. Yani bunları Karnaugh haritası ile sadeleştirmemize gerek yoktur.

Diğer Boole ifadelerini ise Karnaugh haritaları ile aşağıdaki gibi sadeleştiririz:



Bulduğumuz Boole denklemlerini aşağıdaki gibi sıralayabiliriz:

$$
\begin{aligned}
J_A &= C' \\
K_A &= 1 \\
J_B &= A'C' \\
K_B &= A' + C' \\
J_C &= A' \\
K_C &= 1
\end{aligned}
$$

Örnek 0.3:

$0 - 7 - 2 - 5 - 0$ *şeklinde sayan bir sayıcıyı T FF kullanarak en sade haliyle tasarlayınız. Kullanılmayan durumların ne olacağı dikkate alınmayacaktır.*

—

Sayılar 0 ile 7 arasında değiştiğinden üç tane FF kullanılması gerekir. Durum tablosunda 0, 7, 2, 5 durumları aşağıdaki gibi doldurulur:

Şimdiki Durum			Sonraki Durum			Flip Flop Girişleri		
A	B	C	A	B	C	T_A	T_B	T_C
0	0	0	1	1	1			
0	0	1						
0	1	0	1	0	1			
0	1	1						
1	0	0						
1	0	1	0	0	0			
1	1	0						
1	1	1	0	1	0			

Daha sonra kullanılmayan durumların tabloya doldurulması gerekir. Bir önceki örnekten farklı olarak kullanılmayacak durumların sonraki durumlarının ne olacağı serbest bırakılmıştır. Bu nedenle kullanılmayan durumların sonraki değerlerine fark etmez (X) değerleri yazılır:

Şimdiki Durum			Sonraki Durum			Flip Flop Girişleri		
A	B	C	A	B	C	T_A	T_B	T_C
0	0	0	1	1	1			
0	0	1	X	X	X			
0	1	0	1	0	1			
0	1	1	X	X	X			
1	0	0	X	X	X			
1	0	1	0	0	0			
1	1	0	X	X	X			
1	1	1	0	1	0			

Önceki durum sonraki durum ilişkileri belirlendikten sonra flip flop girişlerinin ne olması gerektiği T FF uyarım tablosuna bakılarak belirlenir (önceki notlara bakınız). Eğer tabloda sonraki durum fark etmez ise flip flopun girişinin ne olacağı da fark etmez (0 da 1 de olabilir). Bu nedenle bu satırlardaki flip flop girişlerine de X yazılır. Böylelikle durum tablosu aşağıdaki gibi doldurulmuş olunur:

Şimdiki Durum			**Sonraki Durum**			**Flip Flop Girişleri**		
A	B	C	A	B	C	T_A	T_B	T_C
0	0	0	1	1	1	1	1	1
0	0	1	X	X	X	X	X	X
0	1	0	1	0	1	1	1	1
0	1	1	X	X	X	X	X	X
1	0	0	X	X	X	X	X	X
1	0	1	0	0	0	1	0	1
1	1	0	X	X	X	X	X	X
1	1	1	0	1	0	1	0	1

Buradan FF girişlerine bağlanması gereken Boole denklemlerini miniterimlerin toplamı cinsinden aşağıdaki gibi elde ederiz:

$$T_A(A,B,C) = \sum(0,2,5,7) + d(1,3,4,6)$$
$$T_B(A,B,C) = \sum(0,2) + d(1,3,4,6)$$
$$T_C(A,B,C) = \sum(0,2,5,7) + d(1,3,4,6)$$

T_A ve T_C için sadeleştirme yapmamıza gerek yok. Tüm fark etmez değerlerini 1 alırsak $T_A = T_C = 1$ olur. T_B içinse aşağıdaki gibi bir sadeleştirme yaparız:

$A \backslash BC$	00	01	11	10
0	1	X	X	1
1	X			X

Yukarıdaki sadeleştirmeden $T_B = A'$ elde edilir (bir diğer alternatif seçim de yapılabilirdi). Buna göre, bulduğumuz Boole denklemlerini aşağıdaki gibi sıralayabiliriz:

$$T_A = 1$$
$$T_B = A'$$
$$T_C = 1$$

Gördüğünüz üzere fark etmez durumlarının fazlalığı sayıcı devresini son derece basitleştirdi. Bu durumda, niye önceki örnekte kullanılmayan durumları fark etmez olarak almadık da bir tane durumu başlangıç olarak seçip kullanılmayan tüm durumları o başlangıç durumuna döndürdük diye sorabilirsiniz. Bu sorunun cevabı için bu örnekte bulmuş olduğumuz sayıcıyı simüle ediniz. Simülasyonu yaptığınızda bazılarınız istenildiği gibi $0-7-2-5-0$ şeklinde sayan bir sayıcı gözlemleyecektir. Bazılarınız ise $1-6-3-4-1$ şeklinde bir sayıcı gözlemleyecektir.

Bunun nedeni sayıcı başlatıldığında rastgele bir durumdan başlayacak olmasıdır (power-on-reset ile istenilen durumdan başlama ileride gösterilecek). Eğer sayıcı bizim kullandığımız durumların birinden başlarsa istenildiği gibi çalışacaktır. Fakat, sayıcı devre kullanılmayan durumlardan biri ile başlarsa sonraki durumları fark etmez aldığımız için devrenin sonraki değerlerinin ne olacağı bizim kontrolümüzde olmayacaktır. Sadeleştirme esnasında seçtiğimiz X değerlerine göre kullanılmayan durumlar da kendi içerisinde başka bir döngü oluşturabilir. Yukarıdaki örnekte $1-6-3-4-1$ şeklinde başka bir sayma işlemi gözlemlememizin nedeni budur.

Bazen normal çalışan dijital devreler glitch'ler nedeniyle (zaman analizi kısmında göreceğiz) kullanılmayan durumlardan birinin içine girebilir. Bu durumda da eğer kullanılmayan durumları fark etmez olarak tasarladıysak devrenin bundan sonra nasıl çalışacağını bilemeyiz. Bu örnekteki gibi tamamen başka bir çalışma moduna geçebilir!

İşte bu tür problemlerin önüne geçmek için ardışıl devre tasarlarken daha fazla eleman kullanmak pahasına kullanılmayan durumlar bir başlangıç durumu seçilerek o duruma yönlendirilir. Bu tür bir tasarım daha stabil çalışan bir devre üretecektir: Devre herhangi bir nedenle kullanılmayacak durumların birinin içerisine düştüğü zaman bir sonraki saat darbesinde bizim seçmiş olduğumuz duruma yönlendirilerek normal çalışmasına o durumdan devam edecektir.

Örnek 0.4:

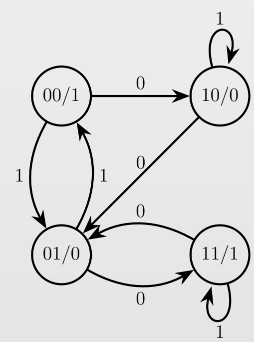
Şimdiki Durum		**Giriş**	**Sonraki Durum**		**Çıkış**
A	B	x	A	B	y
0	0	0	1	0	1
0	0	1	0	1	1
0	1	0	1	1	0
0	1	1	0	0	0
1	0	0	0	1	0
1	0	1	1	0	0
1	1	0	0	1	1
1	1	1	1	1	1

Yukarıda verilmiş olan durum tablosu için: ***a)*** *Mealy mi Moore tipi mi olduğunu belirtiniz.* ***b)*** *Durum diyagramını çiziniz.* ***c)*** *T FF kullanarak en sade haliyle gerçekleştiriniz.*

—

a) Tabloyu incelediğimizde y çıkışının her bir AB durumunda x girişinden bağımsız olarak (yani $x = 0$ olsa da $x = 1$ olsa da) aynı değeri aldığını görürüz. Buna göre, y çıkışı x girişinden bağımsızdır. Bu nedenle tablo bir Moore makinesine aittir.

b) Durum diyagramını Moore makinesi olarak durum tablosuna bakarak aşağıdaki gibi oluştururuz:



c) Öncelikle sonraki durumları elde edebilmek için T FF girişlerinin alması gereken durumları belirleriz. T FF'un uyarım tablosundan durum değişiyorsa FF girişinin 1; durum önceki değerini koruyorsa FF girişinin 0 olması gerektiğini biliyoruz. Buna göre, FF girişleri aşağıdaki tablodaki değerleri almalıdır:

Şimdiki Durum		**Giriş**	**Sonraki Durum**		**FF Girişleri**		**Çıkış**
A	B	x	A	B	T_A	T_B	y
0	0	0	1	0	1	0	1
0	0	1	0	1	0	1	1
0	1	0	1	1	1	0	0
0	1	1	0	0	0	1	0
1	0	0	0	1	1	1	0
1	0	1	1	0	0	0	0
1	1	0	0	1	1	0	1
1	1	1	1	1	0	0	1

Tablodan FF girişlerinin Boole denklemlerini miniterimler cinsinden aşağıdaki gibi elde ederiz:

$$T_A(A, B, x) = \sum(0, 2, 4, 6)$$
$$T_B(A, B, x) = \sum(1, 3, 4)$$
$$y(A, B, x) = \sum(0, 1, 6, 7)$$

Miniterimler ise Karnaugh haritaları ile aşağıdaki gibi sadeleştirilecektir:

$A \backslash Bx$	00	01	11	10
0	1			1
1	1			1

Yukarıdaki Karnaugh haritası $T_A = x'$ sonucunu verir.

$A \backslash Bx$	00	01	11	10
0		1	1	
1	1			

Yukarıdaki Karnaugh haritası $T_B = AB'x' + A'x$ sonucunu verir.

$A \backslash Bx$	00	01	11	10
0	1	1		
1			1	1

Son olarak yukarıdaki harita $y = A'B' + AB$ sonucunu verir. Buna göre, elde ettiğimiz FF giriş bağlantılarını aşağıdaki gibi özetleyebiliriz:

$$
\begin{aligned}
T_A &= x' \\
T_B &= AB'x' + A'x \\
y &= A'B' + AB
\end{aligned}
$$

Elde ettiğimiz sonuçtan da görebileceğiniz üzere çıkış girişten bağımsızdır.

Tanım 0.1: Gray Kodu

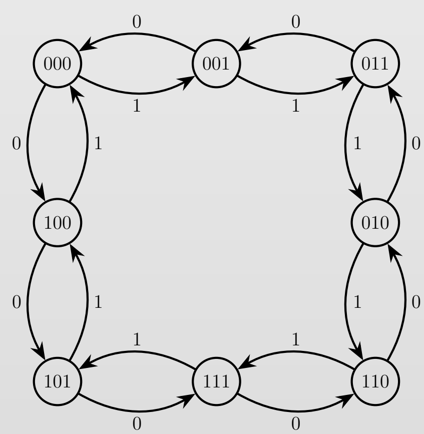
Gray kodu iki ardışık elemanı arasında sadece bir bitin farklılık gösterdiği ikili bit dizileridir. Sayıları normal sırasıyla ikili tabanda ifade ettiğimizde Gray kodunun aksine ardışık elemanlar arasında bazen iki veya daha fazla bit farklılık gösterir. Gray kodunun kullanım amacı her bir durumda sadece tek bir bitin değişmesiyle olası hataların önüne geçmektir. Mesela, elektrik motoru şaftının pozisyonunun belirlenmesinde Gray kodu kullanılır. Böylelikle, pozisyon değişirken iki veya daha fazla bitin birden değişmemesi pozisyon okuması yaparken yanlışlıkla önce değişen bitin okunarak hatalı pozisyon bilgisinin iletilmesini engeller.

Örnek 0.5:

D FF kullanarak aşağı / yukarı sayma özelliğine sahip 3-bitlik bir Gray kodu sayıcı tasarlayınız. Aşağı / yukarı sayma için tek bir giriş kullanılacak olup giriş 0 *ise sayıcı aşağı; giriş* 1 *ise sayıcı yukarı sayma işlemi yapacaktır. 3-bitlik Gray kodları sırasıyla şu şekildedir:* 000 → 001 → 011 → 010 → 110 → 111 → 101 → 100 → 000.

—

Gerçekleştirmek istediğimiz devrenin çalışmasını daha iyi anlayabilmek için aşağıdaki gibi bir durum diyagramı oluşturabiliriz:



Tasarımda D FF kullanacağımız için durum tablosunda sonraki durumlar aynı zamanda D FF girişlerine bağlanması gereken değerler olacaktır. Buna göre, ABC durumlar x giriş olmak üzere durum tablosu aşağıdaki gibi elde edilir:

Şimdiki Durum			**Sonraki Durum** $x = 0$			**Sonraki Durum** $x = 1$		
A	B	C	A	B	C	A	B	C
0	0	0	1	0	0	0	0	1
0	0	1	0	0	0	0	1	1
0	1	0	0	1	1	1	1	0
0	1	1	0	0	1	0	1	0
1	0	0	1	0	1	0	0	0
1	0	1	1	1	1	1	0	0
1	1	0	0	1	0	1	1	1
1	1	1	1	1	0	1	0	1

Tasarım için miniterimleri kullanacağız. Miniterimleri belirlemek için tüm olası A, B, C, x durumlarını alt alta sıralayıp yeni bir tablo oluşturmamız şart değil. A'yı MSB ve x'i LSB olarak kullanırsak yukarıdaki tabloda $x = 0$ sütunları sırasıyla m_0, m_2, m_4, m_6, $\ldots$, m_{14} şeklinde çift indisli miniterimleri; $x = 1$ sütunları ise sırasıyla $m_1, m_3, m_5, m_7, \ldots , m_{15}$ şeklinde tek indisli miniterimleri temsil edecektir. Bunu görebilmek için her bir satırdaki A, B, C, x değerlerine karşılık gelen ondalık indisi belirlemeniz yeterli. Buna göre, D FF girişlerine bağlanması gereken kombinasyonel devre denklemleri miniterimlerin toplamı cinsinden aşağıdaki gibi elde edilir:

$$
\begin{aligned}
D_A(A, B, C, x) &= \sum(0, 8, 10, 14, 5, 11, 13, 15) \\
D_B(A, B, C, x) &= \sum(4, 10, 12, 14, 3, 5, 7, 13) \\
D_C(A, B, C, x) &= \sum(4, 6, 8, 10, 1, 3, 13, 15)
\end{aligned}
$$

Bu Boole denklemlerini sadeleştirerek D FF girişlerini aşağıdaki gibi elde ederiz:

$AB \backslash Cx$	00	01	11	10
00	1			
01		1		
11		1	1	1
10	1		1	1

Yukarıdaki sadeleştirme $D_A = AC + BC'x + B'C'x'$ sonucunu verecektir.

$AB \backslash Cx$	00	01	11	10
00			1	
01	1	1	1	
11	1	1		1
10				1

Yukarıdaki sadeleştirme ise $D_B = BC' + A'Cx + ACx'$ sonucunu verecektir.

$AB \backslash Cx$	00	01	11	10
00		1	1	
01	1			1
11		1	1	
10	1			1

Son olarak yukarıdaki sadeleştirme $D_C = A'B'x + A'Bx' + ABx + AB'x'$ sonucunu verecektir. Elde ettiğimiz FF bağlantılarını aşağıdaki gibi sıralayabiliriz:

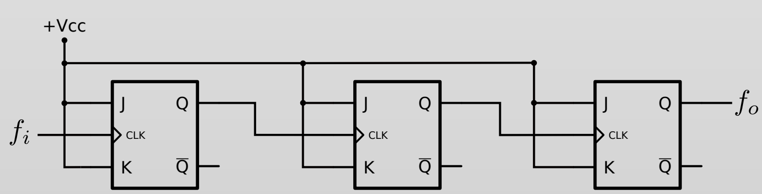
$$D_A = AC + BC'x + B'C'x'$$
$$D_B = BC' + A'Cx + ACx'$$
$$D_C = A'B'x + A'Bx' + ABx + AB'x'$$

Örnek 0.6:

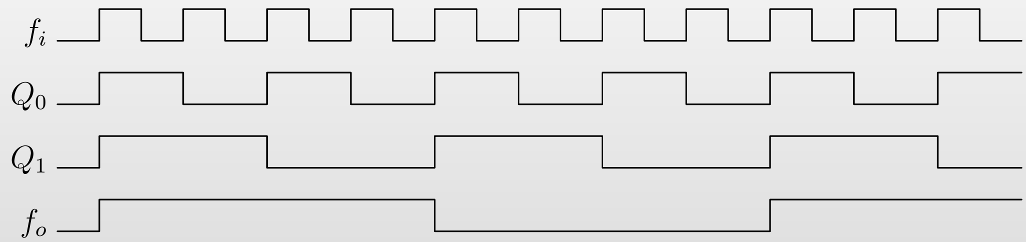
8 kHz frekansa sahip bir kare dalga saat (clock) sinyalini 1 kHz'lik frekansa sahip bir saat sinyaline dönüştürecek devreyi aşağı sayan asenkron sayıcı devresi kullanarak tasarlayınız.

—

Sayıcılar ders notunda asenkron sayıcıları ve nasıl aşağı sayma yapma işlemi elde edebileceğimizi görmüştük. Bu örnekte giriş frekansını 1/8 oranına düşürmek istiyoruz. Bunu 3-bit sayıcı ile yapabiliriz. Tek yapmamız gereken asenkron sayıcının saat girişine orijinal 8 kHz'lik f_i saat sinyalini verdikten sonra MSB çıkışını yeni 1 kHz'lik f_o saat sinyali olarak kullanmak. Bu düzenlemede 3-bitlik sayıcının sekize kadar sayması MSB çıkışında giriş frekansının 1/8'inin elde edilmesini sağlamaktadır. Aşağı sayıcı elde etmek için asenkron saat girişlerini bir önceki flip flopun Q çıkına bağlarsak aşağıdaki şekildeki devreyi elde ederiz:



Frekansın düşmesini daha net görebilmek için devrenin zaman diyagramını aşağıdaki gibi çizebiliriz (Q_0 en soldaki FF'un yani sayıcının LSB çıkışı; Q_1 ise ortadaki FF'un çıkışıdır):



Zaman diyagramından f_o çıkışının periyodunun f_i girişinin periyodunun 8 katı olduğunu görebilirsiniz. Yani frekansı 8 kat daha düşüktür. Zaman diyagramından aşağı sayma işlemini de görebilirsiniz.

Bu tür devrelere **frekans bölücü** adı verilmektedir. Aynı frekans bölücüyü düşen kenar tetiklemeli JK FF ile tasarlayınız.

Örnek 0.7:

Bir ardışıl devrenin çıkışları A ve B olan iki adet JK FF'u ve bir adet x girişi vardır. Flip flopların girişlerinin Boole denklemleri aşağıdaki gibidir:

$$\begin{aligned} J_A &= x & K_A &= B \\ J_B &= x & K_B &= A' \end{aligned}$$

*Buna göre, **a)** JK FF karakteristik denklemini kullanarak A_{n+1} ve B_{n+1} ifadelerini bulunuz. **b)** Devrenin durum diyagramını çiziniz.*

—

Daha önce JK FF karakteristik denkleminin $Q_{n+1} = JQ' + K'Q$ olduğunu ve bu denklemin nasıl elde edildiğini görmüştük. A ve B flip flopları için bu karakteristik denklemi kullanıp yukarıdaki giriş ifadelerini yerine yazarsak aşağıdaki sonraki durum ifadelerini elde ederiz:

$$\begin{aligned} A_{n+1} &= J_A A' + K_A' A = A'x + AB' \\ B_{n+1} &= J_B B' + K_B' B = B'x + AB \end{aligned}$$

Sonraki durum eşitliklerini kullanarak aşağıdaki durum tabosunu kolaylıkla elde edebiliriz:

Şimdiki Durum		**Giriş**	**Sonraki Durum**	
A	B	x	A	B
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0

0	1	1	1	0
1	0	0	1	0
1	0	1	1	1
1	1	0	0	1
1	1	1	0	1

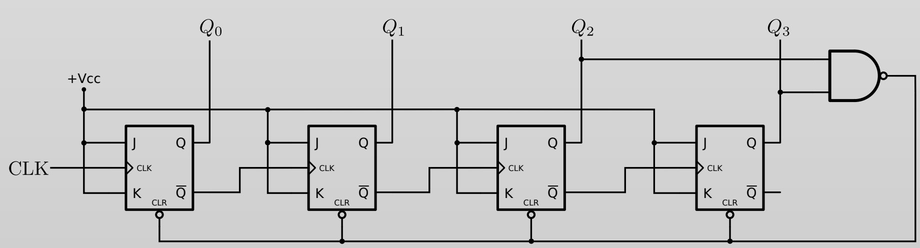
Tablodan durum diyagramını daha önce yaptığımız gibi elde ederiz. Bunu kendiniz yapın.

Örnek 0.8:

Aktif düşük clear girişli ve yükselen kenar tetiklemeli JK FF kullanarak mod 12 *asenkron sayıcı devresi tasarlayınız.*

—

12'ye kadar olan sayıları 4-bitle ifade edebileceğimiz için asenkron sayıcı devresi 4 FF içermelidir. Buna göre sayma işlemi $Q_3Q_2Q_1Q_0 = (0000)_2 = 0$'dan başlayıp $(1011)_2 = 11$ değerinde son bulduktan sonra tekrar başa dönmeli. Bunu ise devre $(1100)_2 = 12$ değerine geldiğinde aktif düşük clear girişlerini 0 yaparak bütün flip flopları asenkron bir şekilde resetlersek yaparız. Dikkat ederseniz 12'ye kadar olan sayılar içerisinde sadece $(1100)_2 = 12$ değerinde $Q_3 = 1$ ve $Q_2 = 1$ olmaktadır. Dolayısıyla, fliğ flopları resetlemek için $Q_3 = Q_2 = 1$ olduğu durumu takip etmemiz gerekir. Bunu bu iki çıkışı VEDEĞİL kapısına sokup VEDEĞİL kapısının çıkışını clear girişlerine bağlayarak elde edebiliriz. Böylelikle VEDEĞİL kapısı sadece $Q_3 = Q_2 = 1$ olduğunda 0 olur ve flip floplar resetlenir. Buna göre devre aşağıdaki gibi olacaktır:



Versiyon Notları

ver. 1.1:

- Sayıcılarda kullanımayan durumlarla alakalı yeni soru eklendi. Metinde değişiklikler yapıldı.