SELÇUK ÜNİVERSİTESİ
MÜHENDİSLİK-MİMARLIK FAKÜLTESİ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ BÖLÜMÜ

LOJİK DEVRELER
DERS NOTLARI

Konya- 2010

KONULAR

1. Analog ve Sayısal (Dijital) Sistemler
2. Sayı Sistemleri, Toplama, Çıkarma, Faz Enerjili Sayılar
3. Sayısal (Dijital) Kodlama
4. Lojik Devre Temelleri
5. Boolean Cebri ve Aksiyomları
6. Lojik Fonksiyonların Sadeleştirilmesi (Karnaugh Diyagramı ile sadeleştirme)
7. Dijital Entegre Lojik Aileler (CRTL, DTL ve TTL)
8. Lojik Devre Katalog Bilgileri
9. Kombinasyonel Devreler (Toplayıcı, Çıkarıcı, Kod Çevirici, Kod Çıkarıcı, Displayler, Buffer, Multiplexer, Demultiplexer, Komparatör)
10. Flip-Floplar (RS-FF, JK-FF, D-FF, T-FF)
11. Sayıcılar (Binary sayıcılar, asenkron sayıcılar, ring sayıcı, Johnson sayıcı, 7493 entegresi)

KAYNAKLAR

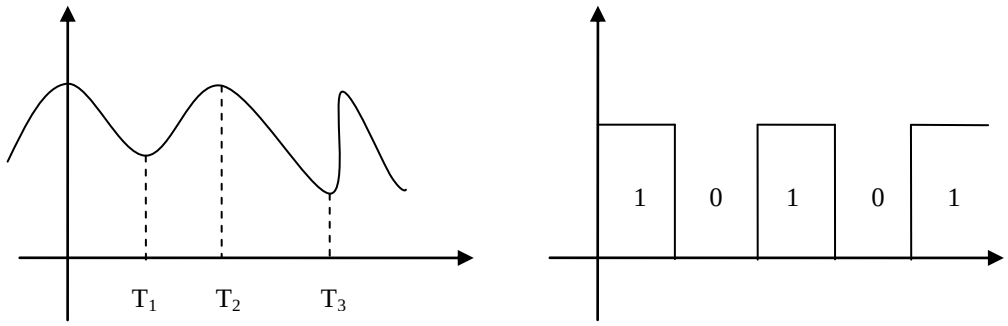
1. Sayısal Tasarım, Morris MANO, MEB Yayınları.
2. Lojik Devreler, Prof. Dr. Emin ÜNALAN, İTÜ Yayınları.
3. Lojik Devreler, Prof. Dr. Emre HARMANCI, İTÜ Yayınları.
4. Lojik Devreler, Prof. Dr. Ahmet DEVİROĞLU, İTÜ yayınları
5. Lojik Devre Tasarımı, Dr. Taner ARSAN, Dr. Rıfat ÇÖLKESEN, Papatya Yayınları.
6. Lojik Devre Tasarımın Temelleri ve Uygulamaları, Prof. Dr. Şirzat Kahramanlı.

1. ANALOG VE SAYISAL (DİJİTAL) SİSTEMLER

1.1. Analog- Sayısal İşaretler

Gerçek dünyada karşılaştığımız birçok fiziksel büyüklüğün (akım, gerilim, sıcaklık, ışık şiddeti vb.) değeri sürekli bir aralık içinde kesintisiz değişmektedir. Sınırlar arasındaki her türlü olası değeri alabilirler. Bu tür işaretlere "analog işaretler" denir.

Sayısal işaretler ise belirli bir aralıkta atlamalı değerler alabilen işaretlerdir. En çok bilinen sayısal işaret ikili (binary) olanıdır. İkili işarette yalnızca iki değer (1/0, darbe/boşluk, H/L, açık/kapalı, var/yok gibi) söz konusudur.



Şekil 1.1 Analog ve sayısal işaret örneği

1.2. Sayısal Sistemlerin Avantajları

Eskiden analog sistemlerin kullanıldığı birçok alanda (fotoğrafçılık, video-ses kayıtları, haberleşme sistemleri vb.) günümüzde daha avantajlı olan sayısal sistemler kullanılmaktadır.

Sayısal sistemlerin avantajları şu şekilde sıralanabilir:

1. Bir sayısal sisteme belli bir giriş kümesi defalarca uygulandığında hep aynı çıkış kümesi elde edilir. Analog sistemler ise çevre koşullarından daha çok etkilenir.
2. Sayısal tasarım (Lojik tasarım) aldığı matematiksel değerler açısından daha kolaydır. Ayrıca sayısal sistemleri test etme ve hatalardan arındırmak da analog sistemlere göre daha kolaydır.
3. Esneklik ve programlanabilirlik bakımından da sayısal sistemler daha avantajlıdır. Bu sayede aynı sistem değişen gereksinimlere göre yeniden programlanabilir.
4. Bilgilerin sayısal ortamda saklanması ve işlenmesi daha kolaydır.
5. Sayısal sistemler daha hızlı çalışmaktadır.

6. Sayısal sistemler küçülmekte ve ucuzlamaktadır.
7. Verinin sistemler arası iletişimi kolay ve esnektir.
8. Gelişmeye ve yenilenmeye açıktır.

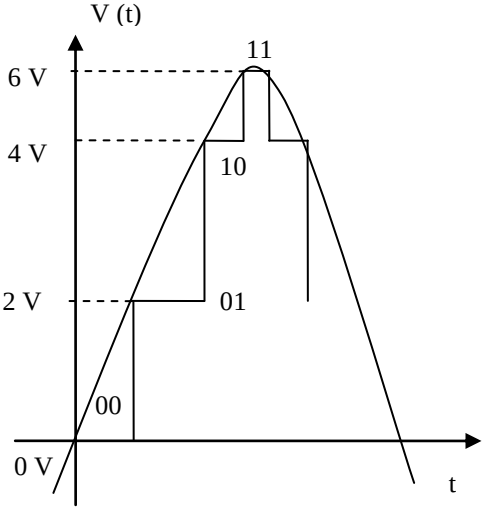
Analog sistemlerin avantajları şu şekilde sıralanabilir:

1. Analog sinyal gösterebileceği değer aralığında her değeri alabilir. Böylece söz konusu değer tam olarak gösterilebilir. Sayısal sinyal ise göstereceği değer bölgesinde sabittir. Değerin katlarını gösterir ve her noktayı ölçemeyiz.
2. Sürekli ve kesintisizdir.
3. Analog sinyalin algılanması kolay olduğu için işlenmesi de basittir. Sayısal sinyallerin algılanması daha zor ve işlenmesi karmaşıktır.

1.3. Analog/Sayısal Dönüşüm

Aslında analog bir işaretin ikili bir işarete dönüştürülmesi kolay bir işlemdir. Bir kırpıcı devresiyle yapılabilir. Çünkü analog işareti belirli bir eşiği geçince 1, altında kalınca 0 üretilerek ikiliye dönüşüm gerçekleştirilebilir. Ancak analog işaretten kodlanmış sayısal işaret elde edilmesi yada sayısal işaretten analog işaret elde edilmesi için ADC (Analog-to-Digital Converter) yada DAC (Digital-to-Analog Converter) adı verilen özel elemanların kullanılması gerekir. ADC/DAC için iki önemli parametre vardır. Bu parametreler kodların kaç bit olacağı ve dönüştürme hızıdır. Kodların kaç bit olduğu doğrudan elemanın kaç bitlik olacağını yani dönüştürme duyarlılığını belirler. Örneğin giriş değeri $0-16\,\mathrm{V}$ arasında ise dönüştürücü $4-\text{bitlik}$' dir ve $\text{Duyarlılık}=\frac{16}{2^4}=1\mathrm{V}$ olacaktır. $8-\text{bitlik}$ ise $\text{Duyarlılık}=\frac{16}{2^8}=0.0625\mathrm{V}$ olur. Kısaca dönüştürücünün bit sayısı artarsa duyarlılığı artar.

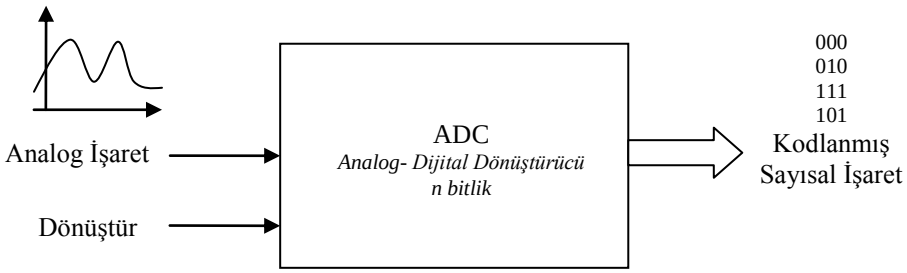
Dönüştürme hızı ise, dönüştürmede kullanılan yönteme göre değişir. En hızlı olan Flash dönüştürücülerdir. Dönüştürme hızı, bir dönüştürme işlemini en kötü durumda ne kadar sürede yapacağını belirler.



Şekil 1.2 Analog işaretin sayısala dönüştürülmesi
0→00(0 Volt)
1→01(2 Volt)
2→10(4 Volt)
3→11(6 Volt)

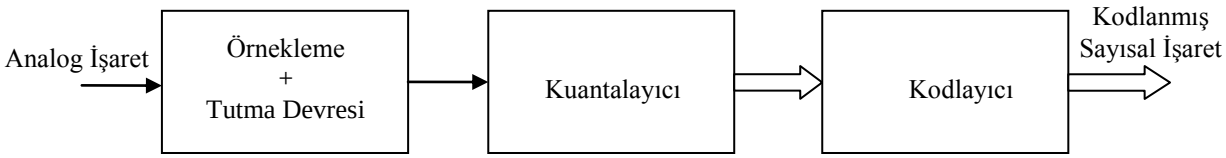
1.3.1.ADC (Analog-to-Digital Converter)

Analog→Sayısal dönüştürme işlemi, analog işaretin taşıdığı bilginin kuantalanmış ve kodlanmış şeklini sayısal bir kod ile ifade etme anlamına gelir.



Şekil 1.3 Analog-Dijital dönüştürme işlemi

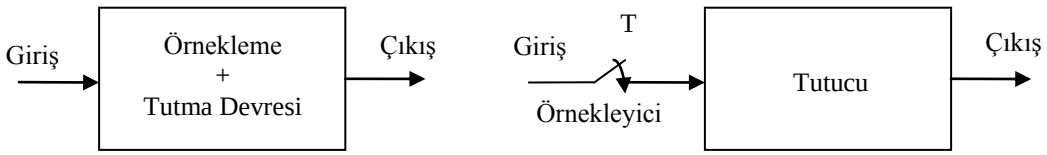
ADC' nin blok diyagramı aşağıdaki şekilde görülmektedir.



Şekil 1.4 ADC' nin blok diyagramı

Örnekleme ve Tutma İşlemleri

Örnekleme elemanının görevi belirli aralıklarla analog işaretten örnekler alır ve bu değerleri belirli bir süre tutar.



Şekil 1.5 Örnekleme ve Tutma Devresi

Burada T örnekleme zamanını belirtir. Örnekleme süresi, örnekleme periyodundan çok küçük olduğu için sıfır kabul edilir.

Örnekleme elemanının çalışma ilkesi Shannon teoremi olarak bilinen örnekleme teoremine dayanmaktadır. Bu teorem, bant sınırlı herhangi bir $x(t)$ işareti belirli aralıklarla örneklendikten sonra, alınan örnek değerlerde herhangi bir bozulma olmaksızın $x(t)$ işaretini tekrar elde edebilmek için örnekleme frekansının seçilmesini bir kurala bağlamaktadır. Buna göre, $x(t)$ işaretine ait en yüksek frekanslı bileşenin frekansı ω_c radyan/saniye ise seçilebilecek en düşük örnekleme frekansı, en yüksek frekans olan ω_c' nin iki katı olmalıdır. Bu durumda ω_s örnekleme frekansı olmak üzere,

$$\omega_s \geq 2\omega_c$$

örnekleme teoreminin en genel ifadesi olarak belirlenir.

Kuantalama

Sürekli bir büyüklüğü belirli sayıda eşit aralıklı basamaklara ayırma işlemi olarak tanımlanabilir. Kuantalama kombinasyonundaki eleman sayısı arttıkça duyarlılık artar. Duyarlılık çıkış kodunda değişiklik oluşturabilecek en küçük giriş değeridir. Kuantalama işleminin duyarlılığı kullanılan ADC' nin yapısına göre değişir.

n bit ile kodlama yapılacağı düşünülürse, 2^n tane kuanta düzeyi ve 2^{n-1} tane kuantalama aralığı elde edilir. Kuantalama aralığının büyüklüğü (duyarlılık) a ile gösterilirse;

$$a = \frac{V_{max} - V_{min}}{2^n} \quad (\text{bazı uygulamalarda a} = \frac{V_{max} - V_{min}}{2^n - 1})$$

olarak bulunur.

Örnek 1.1. -3 Volt ile $+5$ Volt arasında değişen bir analog işaretin 1 Volt duyarlılıkla sayısal olarak ifade etmek için kaç bit kullanmak gerekir?

$$a = 1\ Volt \text{ olduğuna göre } 1 = \frac{5-(-3)}{2^n} \Rightarrow 2^n = 8 \Rightarrow n = 3 \text{ bit}$$

Örnek 1.2. Bir analog işaret 0-15 Volt arasında kesintisiz değerler alabilmektedir. Bu işaretin 100 mV duyarlılıkla kodlanmış sayısal işarete dönüştürülmesi için kaç bitlik bir ADC gerekir?

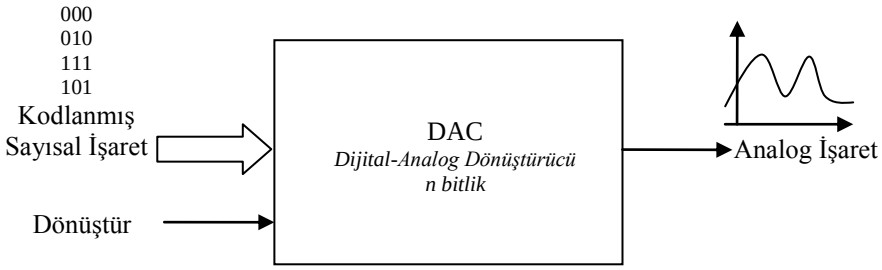
$$a = 100\ mV \text{ olduğuna göre } 0.1 = \frac{15-0}{2^n} \Rightarrow 2^n = 150 \Rightarrow n = 7.23 \text{ bit} \cong 8 \text{ bit}$$

Kodlama

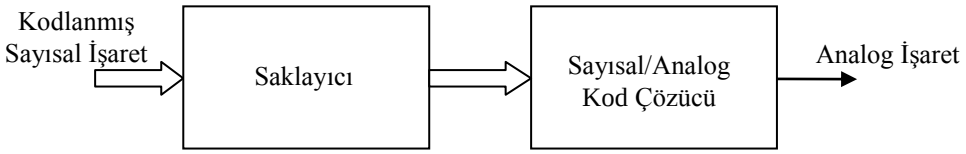
Kodlama kuantalama düzeylerine ikili sayı sisteminde birer kod verme işlemidir.

1.3.2. DAC (Digital-to-Analog Converter)

Sayısal veriden analog işaret üreten elemandır. Girişlerine n bitlik sayısal işaret uygulanır ve belirli bir gecikmeyle çıkışlarında buna karşılık düşen analog işaret elde edilir. Sayısal veriyi analog işarete dönüştürmek için pek çok yöntem mevcuttur. Bunlardan en çok kullanılanları "ağırlık orantılamalı direnç devresi ve işlemsel kuvvetlendirici" dir.



Şekil 1.6 Dijital-Analog dönüştürme işlemi



Şekil 1.7 DAC' nin blok diyagramı

Örnek 1.3. 4-bitlik bir DAC devresinin çıkış gerilim aralığı 0 V ile 6 V arasında değişmektedir. 0000 sayısal girişi 0 V' a, 1111 sayısal girişi 6 V' a karşılık düşecek biçimde DAC' a ait dönüştürme tablosu oluşturunuz.

$$a = \frac{V_{max} - V_{min}}{2^n} = \frac{6-0}{2^4} = 0.375 \text{ Volt yada } a = \frac{V_{max} - V_{min}}{2^n - 1} = \frac{6-0}{2^4 - 1} = 0.4 \text{ Volt}$$

Sayısal Girişler	Analog Çıkış $a = \frac{V_{max} - V_{min}}{2^n}$	Analog Çıkış $a = \frac{V_{max} - V_{min}}{2^n - 1}$
0000	0.000	0.0
0001	0.375	0.4
0010	0.750	0.8
0011	1.125	1.2
0100	1.500	1.6
0101	1.875	2.0
0110	2.250	2.4
0111	2.625	2.8
1000	3.000	3.2
1001	3.375	3.6
1010	3.750	4.0
1011	4.125	4.4
1100	4.500	4.8
1101	4.875	5.2
1110	5.250	5.6
1111	5.625	6.0

2. SAYI SİSTEMLERİ

Sayı sistemleri aşağıdaki gibi kategorize edilebilir:

a) Onluk (Decimal) sayı sistemi → 10 tabanlı

b) Sekizlik (Octal) sayı sistemi → 8 tabanlı

c) İkili (Binary) sayı sistemi → 2 tabanlı

d) Onaltılı (Hexadecimal) sayı sistemi → 16 tabanlı

Herhangi bir tabandaki sayı şu şekilde ifade edilir:

- $(173{,}25)_{10} = 1.10^2 + 7.10^1 + 3.10^0 + 2.10^{-1} + 5.10^{-2}$
- $(1247{,}172)_8 = 1.8^3 + 2.8^2 + 4.8^1 + 7.8^0 + 1.8^{-1} + 7.8^{-2} + 2.8^{-3}$
- $(10111)_2 = 1.2^4 + 0.2^3 + 1.2^2 + 1.2^1 + 1.2^0$

10 tabanı	2 tabanı	8 tabanı	16 tabanı	BCD
0	0000	0	0	0000
1	0001	1	1	0001
2	0010	2	2	0010
3	0011	3	3	0011
4	0100	4	4	0100
5	0101	5	5	0101
6	0110	6	6	0110
7	0111	7	7	0111
8	1000	10	8	1000
9	1001	11	9	1001
10	1010	12	A	
11	1011	13	B	
12	1100	14	C	
13	1101	15	D	
14	1110	16	E	
15	1111	17	F	

2.1. n tabanından 10 tabanına dönüşüm

- $(547{,}6)_8 = 5.8^2 + 4.8^1 + 7.8^0 + 6.8^{-1}$
- $(1001{,}0111)_2 = 1.2^3 + 0.2^2 + 0.2^1 + 1.2^0 + 0.2^{-1} + 1.2^{-2} + 1.2^{-3} + 1.2^{-4}$
- $(3CD8)_{16} = 3.16^3 + 12.16^2 + 13.16^1 + 8.16^0$

2.2. m tabanından n tabanına dönüşüm

$$r = p.\frac{\ln m}{\ln n}$$

$m =$ mevcut taban, $n =$ dönüştürülecek taban,

$p =$ virgülden sonraki veri sayısı, $r =$ virgülden sonraki veri sayısı

- $(253{,}263)_{10} = (\mathbf{375{,}206})_8$

  $r = 3 \times \dfrac{\ln 10}{\ln 8} = 3.32 \quad 253/8 = 31\ (kalan = 5) \Longrightarrow 375$

  $31/8 = 3 \quad (kalan = 7)$

  $0.263 \times 8 = \mathbf{2.104} \quad 0.104 \times 8 = \mathbf{0.832} \quad 0.832 \times 8 = \mathbf{6.656} \Longrightarrow \mathbf{206}$

- $(0.37)_{10} = (\mathbf{0.0101111})_2$

  $r = 2 \times \dfrac{\ln 10}{\ln 2} = 6.64$

  $0.37 \times 2 = \mathbf{0.74} \quad 0.74 \times 2 = \mathbf{1.48} \quad 0.48 \times 2 = \mathbf{0.96} \ldots\ldots\ldots. \Longrightarrow \mathbf{0101111}$

- $(0.25)_{10} = (\mathbf{0.01})_2$

  $r = 2 \times \dfrac{\ln 10}{\ln 2} = 6.64$

  $0.25 \times 2 = \mathbf{0.5} \quad 0.5 \times 2 = \mathbf{1.00} \Longrightarrow \mathbf{01}$

- $(0.7304)_8 = (\mathbf{0.1011} \ldots)_2$

  $r = 4 \times \dfrac{\ln 8}{\ln 2} = 12$

  $0.7304 \times 2 = \mathbf{1.4608} \quad 0.4608 \times 2 = \mathbf{0.9216}$

  $0.9216 \times 2 = \mathbf{1.8432} \quad 0.8432 \times 2 = \mathbf{1.6864} \ldots\ldots\ldots. \Longrightarrow \mathbf{1011} \ldots$

2.3. Sayı Sistemlerinde Toplama ve Çıkarma

2.3.1. İkili sayılarda toplama ve çıkarma

a	b	$a + b$	Elde	$a - b$	Ödünç
0	0	0	0	0	0
0	1	1	0	1	1
1	0	1	0	1	0
1	1	0	1	0	0

$1101 + 1111 = 11100\ (13 + 15 = 28)$

$1011 - 0101 = 0110 \quad (11 - 5 = 6)$

2.3.2. 8' li ve 16' lı sayılarda toplama ve çıkarma

$(15)_8 + (37)_8 = (54)_8 \quad (9)_{16} + (8)_{16} = (11)_{16}$

$(26)_8 - (17)_8 = (07)_8 \quad (26)_{16} - (17)_{16} = (0F)_{16}$

2.3.3. İşaretli Sayıların Gösterimi

Binary sayının en solundaki bit (MSB) "işaret biti" olarak kullanılır. Bu bit "1" ise sayı negatif, "0" ise pozitif olduğu anlaşılır. Normal ikili sayı ile karışmaması için en anlamlı bitin yanına 0 yada 1 eklenerek altı çizilir. Diğer yöntem ile 1' e ve 2' ye tümleme alınarak sayı gösterilir. Bu gösterim özellikle negatif sayılarda uygulanır.

ÖNEMLİ: ***1' e tümleme ⟹ 1 olan bitler 0, 0 olan bitler 1 yapılır.***

2' ye tümleme ⟹1'e tümlemeyle elde edilen sonuca 1 eklenerek bulunur.

- $-5_{10} = \underline{1}0101_2 \quad 5 = \underline{0}0101_2$ *(İşaret biti kullanılıyor)*
- $5_{10} = 0101_2 \xrightarrow{1'\text{e tümleme}} 1010_2 \xrightarrow{2'\text{e tümleme}} 1011_2 \implies -5_{10} = 1011_2$ *(İş.bit kullanılmıyor)*

İkili tabanda gösterimleri aynı olmasa da bu iki sayı 10 tabanında -5' e karşılık gelir.

2.3.4. İşaretli sayılarda tabana göre tümleme aritmetiği ile çıkarma işlemi

N sayısının r' ye göre tümleyenini hesaplamak için şu formül kullanılır: $\boldsymbol{r^n - N}$ (n:işlemdeki en büyük sayının tamsayı sayısı)

10 tabanında

- $(66358)_{10} - (2164)_{10} = ?$

  $(2164)_{10}$ sayısının 10' a tümleyeni alınır: $10^5 - (2164)_{10} = (97836)_{10}$

  $(66358)_{10} + (97836)_{10} = (164194)_{10}$ Elde oluştuğu için sonuç pozitiftir ve atılır, çıkarma işleminin sonucu **+64194** olarak elde edilir.
- $(2164)_{10} - (66358)_{10} = ?$

  $(66358)_{10}$ sayısının 10' a tümleyeni alınır: $10^5 - (66358)_{10} = (33642)_{10}$

  $(2164)_{10} + (33642)_{10} = (35806)_{10}$ Elde oluşmadığı için sonuç negatiftir, bu yüzden sonucun 10' a tümleyeni alınır:

  $(10^5)_{10} - (35806)_{10} = (-64194)_{10}$ olarak bulunur.

İkili tabanda

- $(1110011)_2 - (1101010)_2 = ?$

  $(1101010)_2$ sayısının 2' e tümleyeni alınır: $1^{'}$tümleme $\longrightarrow (0010101)_2$

  $2^{'}$tümleme$\longrightarrow (0010101)_2 + (0000001)_2 = (0010110)_2$

  $(1110011)_2 + (0010110)_2 = (10001001)_2$ taşma olduğu için sonuç pozitif olarak alınır, elde göz önünde bulundurulmaksızın sonucun $+\mathbf{0001001}$ olduğu söylenir.

- $(1101010)_2 - (1110011)_2 = ?$

  $(1110011)_2$ sayısının 2' e tümleyeni alınır: $1^{'}$tümleme $\longrightarrow (0001100)_2$

  $2^{'}$tümleme$\longrightarrow (0001100)_2 + (0000001)_2 = (0001101)_2$

  $(1101010)_2 + (0001101)_2 = (1110111)_2$ elde yok, sonuç negatiftir bu yüzden 2' ye tümleyeni alınır.

  $(1110111)_2$ sayısının 2' e tümleyeni alınır: $1^{'}$tümleme $\longrightarrow (0001000)_2$

  $2^{'}$tümleme$\longrightarrow (0001000)_2 + (0000001)_2 = (0001001)_2$

  Sonuç=$-\mathbf{0001001_2}$

- $X = 1010100, Y = 1000011$ ikiye tümleyenlerini kullanarak aşağıdaki işlemleri yapınız.

  a) $X - Y = ?$ b) $Y - X = ?$

  a) $X - Y = 1010100 - 1000011 = ?$ $1^{'}$tümleme $\longrightarrow (0111100)_2$

  $2^{'}$tümleme$\longrightarrow (0111100)_2 + (0000001)_2 = (0111101)_2$

  $X - Y = 1010100 + 0111101 = 10010001$ elde atılır, sonuç pozitiftir.

  Sonuç= $+\mathbf{0010001}$' dir.

  b) $Y - X = 1000011 - 1010100 = ?$ $1^{'}$tümleme $\longrightarrow (0101011)_2$

  $2^{'}$tümleme$\longrightarrow (0101011)_2 + (0000001)_2 = (0101100)_2$

  $Y - X = 1000011 + 0101100 = 1101111$ elde yok, sonuç negatif, sonucun 2' ye tümleyenini alınır. $0010000 + 0000001 = -\mathbf{0010001} \rightarrow$ Sonuç $= -\mathbf{0010001}$

1' e tümleyenle çıkarma

- $X = 1010100, Y = 1000011$ 1' e tümleyenlerini kullanarak aşağıdaki işlemleri yapınız.

  a) $X - Y = ?$ b) $Y - X = ?$

  a) $Y^{'} = Y^{'}nin\ 1^{'}e\ tümleyeni$

  $X + Y^{'} = 1010100 + 0111100 = 10010000$

  $0010000 + 0000001(Elde\ Aktarım) = \mathbf{0010001}$

b) $X^{'} = X^{'} in\ 1^{'} e\ \ t\ddot{u}mleyen$

$Y + X^{'} = 1000011 + 0101011 = 1101110$

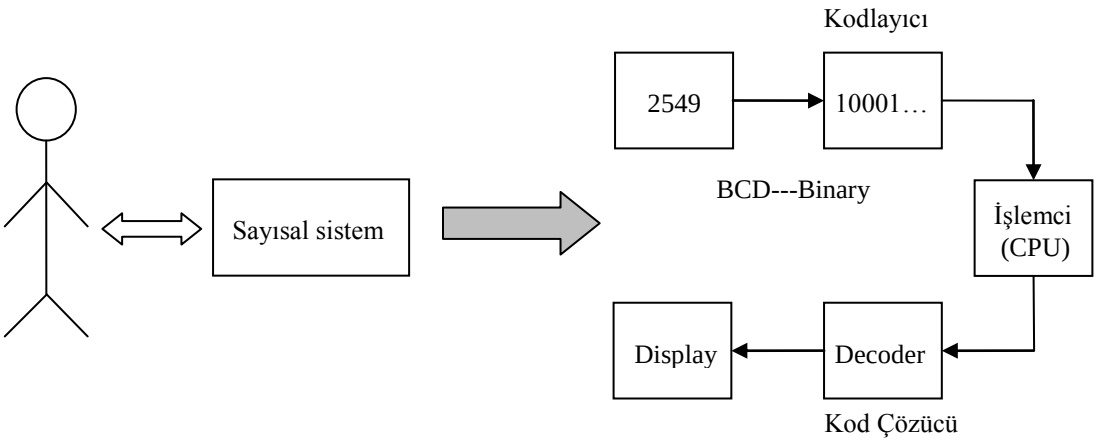
Elde yok, sonucun $1^{'}$e tümleyeni alınır $(1101110)^{'} = \mathbf{-0010001}$

- $(1101)_2 - (0111)_2 = (1101)_2 + (1000)_2 = (10101)_2 + (0001)_2 = +0110_2$
- $(0111)_2 - (1101)_2 = (0111)_2 + (0010)_2 = (1001)_2 \xrightarrow{1^{'} e\ t\ddot{u}mleme} = -0110_2$

3. SAYISAL (DİJİTAL) KODLAMA

Kodlama

Bilginin veya verinin sayısal olarak gösterilmesi için kullanılan yöntemdir. Kodlama sonlu elemanlı bir kümenin her bir elemanına birer tane kod verilmesi olarak ifade edilebilir. Ayrıca ortak özellikleri olan iki kümenin birbirine denk düşürülmesine de "kodlama" denir. Sayısal sistemler temelde ikili (binary) sayı düzenine dayanır. Dolayısıyla sayısal olarak gösterilmesi ya da saklanması gereken bilgi/veri önce kodlanarak sayısal olarak sembolize edilmelidir. Örnek olarak; alfabedeki harflerin sayısal ifadelerle gösterilmesi verilebilir.



Şekil 3.1. Sayısal sistem ile insanın diyalog kurması
(2549 kişinin klavyeden girdiği sayı)

Örnek 3.1. Altı elemanlı bir kümenin kodlamasını inceleyelim.

$$F = \{\text{Ali, Veli, Sıra, Masa, Ayşe, Tahta}\}$$

$$000\ 010\ 011\ \ 001\quad 100\quad 101$$

Bu F kümesindeki 6 elemanı kodlamak için n bite ihtiyaç olduğu düşünülürse;

$$2^n = 6 \rightarrow n \cong 3\ bit$$

olarak bulunur.

3 bit ile kodlama yapıldığında kullanılmayan 2 durum kaldığı için bu kodlama türüne "artıklı kodlama" denir. Eğer bit sayısı ile oluşan bütün kodlar kullanılırsa "artıksız kodlama" dan söz edilir.

Örnek 3.2. Alfabedeki harflerin her biri bir kodla ifade edilecek olsa gereken bit sayısı

$$2^n = 29 \rightarrow n \cong 5\ bit$$

olarak bulunur.

Sayısal Kodlama

Sayısal kodlama, sayıların bellekte tutulma şeklini belirler, sayılar ya doğrudan ikili karşılıkları kullanılarak yada sayının her hanesindeki rakama bir kod atanarak temsil edilirler. Sayıların doğrudan ikili tabandaki karşılıkları kullanıldığında ilgili sayı doğrudan ikili tabana dönüştürülür ve bellekte öyle temsil edilir. Örneğin 6 sayısı 0110, 32 sayısı 10000 karşılıklarıyla temsil edilir. Diğer bir saklama şekliyse sayının hanelerine ait rakamlara birer kod karşı düşürülmesiyle yapılır. Örneğin BCD saklama şeklinde 0-9 arasındaki her bir rakama 4 bitlik bir kod atanır ve sayı saklanırken sayının hanelerindeki rakamlara ait kodlar kullanılır. Örneğin 6 sayısı 0110, 32 sayısı 0011 0010 olarak temsil edilir. BCD dışında üç fazlalık, Aiken ve Gray kodlama şekilleri de kullanılır.

3.1. İkili Kodlanmış Ondalık Gösterim (BCD-Binary Coded Decimal)

Ondalık sayının her hanesinin ikili olarak kodlanmasıdır. Bir taban dönüşümü değildir. Örneğin $(24)_{10} \rightarrow (11000)_2$ bu bir ikili taban karşılığıdır. Ancak BCD karşılığı $(0010\ 0100)_2$ olarak yazılır. BCD kodlamada 0-9 arası sayılar kullanılır. 10-15 arası kullanılmamaktadır. Dolayısıyla BCD kodlama artıklı bir kod olarak karşımıza çıkmaktadır.

3.1.1. BCD Toplama

- Hanelerin toplanması sırasında sonuç 9' a eşit veya küçükse sonuç zaten BCD' dir.
- Toplama sonucu 10-16 arasında ise sonuç BCD kodunda değildir. Sonucu BCD olarak ifade etmek için, ikili kodlanmış 6 sayısını ekleyerek oluşan eldeyi bir üst kademeye aktarmak gerekir.

Örnek 3.3. $(25)_{10} + (11)_{10} = (36)_{10} \Longrightarrow (0010\ 0101)_2 + (0001\ 0001)_2 = (0011\ 0110)_2$

Örnek 3.4. $(24)_{10} + (39)_{10} = (63)_{10} \Longrightarrow (0010\ 0100)_2 + (0011\ 1001)_2 = (0101\ 1101)_2$

$$(0101\ 1101)_2 + (0110)_2 = (1\ \ 0011)_2$$

Birler basamağının elde biti onlar basamağına eklenir.

$$(1\ \ 0011)_2 + (0101\ 0000)_2 = (0110\ 0011)_2$$

3.1.2. BCD Çıkarma

- BCD' de çıkarma işlemi gerçekleştirmek için öncelikle çıkarılacak sayının 10' a tümleyenini almak gerekir. Bu aşamadan sonra 10 tabanındaki sayı BCD olarak ifade edilir ve toplama işlemi gerçeklenir.

Örnek 3.5. $(69)_{10} - (32)_{10} = ? \Longrightarrow (10^2)_{10} - (32)_{10} = (68)_{10}$

10'a tümleyeni alındı ve toplandı

$$\Longrightarrow (69)_{10} + (68)_{10} = (1\ 37)_{10}$$

İşlemin ikili tabanda gerçeklenmesi

$(0110\ 1001)_2 + (0110\ 1000)_2 = (1100\ \ 1\ 0001)_2$

$(1100\ \ 1\ 0001)_2 + (0110\ 0110)_2 = (1\ 0010\ 1\ \ 0111)_2$ Birler basamağının elde biti onlar basamağına eklenir. $= (\ 1\ 0011\ 0111)_2$

Buradaki elde biti sayının pozitif olduğunu gösterir. O yüzden dikkate alınmaz.

3.2. Üç Fazlalık Kodu (Excess-3)

İkili sistemin üç fazlası alınarak oluşturulan kodlama işlemidir. Ağırlığı olmayan simetrik bir koddur.

Sayı	3-Fazlalık Kodu
0	0011
1	0100
2	0101
3	0110
4	0111
5	1000
6	1001
7	1010
8	1011
9	1100

3.2.1. Üç fazlalık kodunda aritmetik işlemler

Üç fazlalık kodunda iki sayının toplanması sonucunda elde edilen sayının yine üç fazlalık kodunda olması gerekmektedir. Ancak bazı durumlarda üç fazlalık kodunda bir sayıyı ifade etmek için ilave işlemelere ihtiyaç duyulur.

1. Toplamada mevcut hanelerin dışında üst haneye geçen bir sayı bulunmazsa ikili tabanda 3 çıkarmak gerekir.
2. Toplamada üst haneye geçen bir sayı oluşursa, ikili tabanda 3 eklenir. Bu ekleme işlemi üst haneye geçen sayı için de yapılır.

Örnek 3.6. $(2)_{10} + (3)_{10} = (5)_{10} \Longrightarrow (0101) + (0110) = (1011)$ ***(1. kural)***

$$(1011) - (0011) = (1000)$$

Elde edilen sonuç da üç fazlalık kodundadır ve ondalık tabanda 5' karşılık gelir.

Örnek 3.7. $(5)_{10} + (8)_{10} = (13)_{10} \Longrightarrow (1000) + (1011) = (1\ 0011)$ ***(2. kural)***

$$(1\ 0011) + (0011) = (0001\ 0110)$$

$$(0001\ 0110) + (0011\ 0000) = (0100\ 0110)$$

Elde edilen sonuç da üç fazlalık kodundadır ve ondalık tabanda 13' karşılık gelir.

3.3. Aiken Kodu

Tabloda verildiği gibi, 0-9 arasındaki sayıların ilk beş ve son beş rakamlarının ikili karşılıklarından oluşur, simetrik bir koddur. Bu kodlama türünün özelliği (0-4) arasındaki ilk beş sayının bilinen ikili kodlamaya eşdeğer olduğu, (5-9) arasındaki ikinci beş sayının ise ilk beş sayının 1' e tümleyeni olduğu söylenebilir.

Sayı	Aiken Kodu
0	0000
1	0001
2	0010
3	0011
4	0100
5	1011
6	1100
7	1101
8	1110
9	1111

3.3.1. Aiken kodunda aritmetik işlemler

1. Toplama işleminin sonucu yine Aiken kodunda olduğundan düzenlemeye gerek olmayan durumlar vardır.
2. Toplama işlemi sonucunda doğru netice ile karşılaşılması halinde üst kademeye geçen bir elde yoksa sonuca ikili tabanda 6 eklenir.
3. Toplama işlemi sonucunda üst haneye geçen bir elde oluşursa, sonuçtan ikili tabanda 6 çıkarılır.

Örnek 3.8. $(3)_{10} + (6)_{10} = (9)_{10} \Longrightarrow (0011) + (1100) = (1111)$ ***(1. kural)***

Örnek 3.9. $(2)_{10} + (3)_{10} = (5)_{10} \Longrightarrow (0010) + (0011) = (0101)$ ***(2. kural)***

$$(0101) + (0110) = (1011)$$

Örnek 3.10. $(8)_{10} + (9)_{10} = (17)_{10} \Longrightarrow (1110) + (1111) = (1\ 1101)$ ***(3. kural)***

$(1\ 1101) - (0110) = (1\ 0111)$ ***(Burada çıkarma işlemi tümleyen aritmetiğine göre de yapılabilir)***

Elde edilen sonuç aiken kodunda olmadığı için 2. kurala dönülür ve 6 eklenir.

$$(1\ 0111) + (0110) = (1\ 1101) \Longrightarrow \text{Aiken kodunda } 17 \text{ sayısını ifade eder}$$

3.4. Bitişik Kodlar ve Gray Kodu

Birbirini izleyen sayılara karşılık alınan ikili kod sözcükleri arasındaki uzaklık (Hamming uzaklığı) "1" ise bu tür kodlara "Bitişik Kodlar" adı verilir. Ayrıca kod sözcüklerinin birincisi ile sonuncusu arasındaki uzaklık yine "1" ise bu tür kodlamalara "Çevrimli Bitişik Kodlar" denir. Örneğin, 0' dan 3' e kadar sayılar kodlamada 00, 01, 11, 10 sözcükleri kullanılırsa bitişik kodlama yapılmış olur.

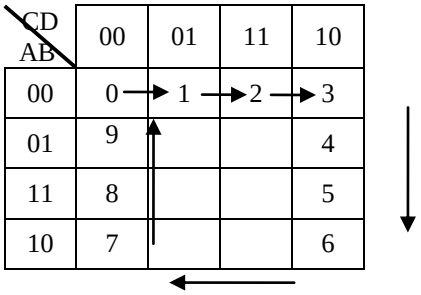
Örnek 3.11. Dört bitlik çevrimli bir BCD kodlaması oluşturulması

Bunun için 2 girişli, dört satırlı ve dört sütunlu bir tablo çizilir. Satır ve sütunlar yukarıdaki örnekteki gibi çevrimli kodla simgelendirilir (Karnaugh Diyagramı)

Sayı	Kod Sözcüğü (ABCD)
0	0000
1	0001
2	0011
3	0010
4	0110
5	1110
6	1010
7	1000
8	1100
9	0100

Örnek 3.12. Karnaugh diyagramı ile çevrimli bir BCD kodunun elde edilmesi

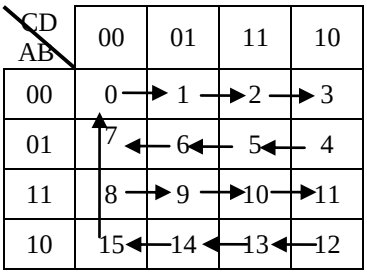
Karnaugh diyagramı üzerindeki birbirine komşu olan hanelerden on tanesini bir çevrim yapacak şekilde seçecek olursak, bu haneleri işaretleyen satır ve sütun simgeleri çevrimli bir kod sözcükleri kümesi oluşturur.



3.4.1. Gray kodu

2^n elemanlı bir küme için 2 tabanında artıksız ve çevrimli bir kodlama yapılırsa yansımalı bir kod yani "Gray Kodu" elde edilir. Sayma işleminde ve sütun tarama işlemlerinde kullanılır. Gray kodu Karnaugh diyagramının geçişlerinde kullanılacaktır.

Oluşturulan kodlar aşağıdaki tabloda gösterilmiştir.



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

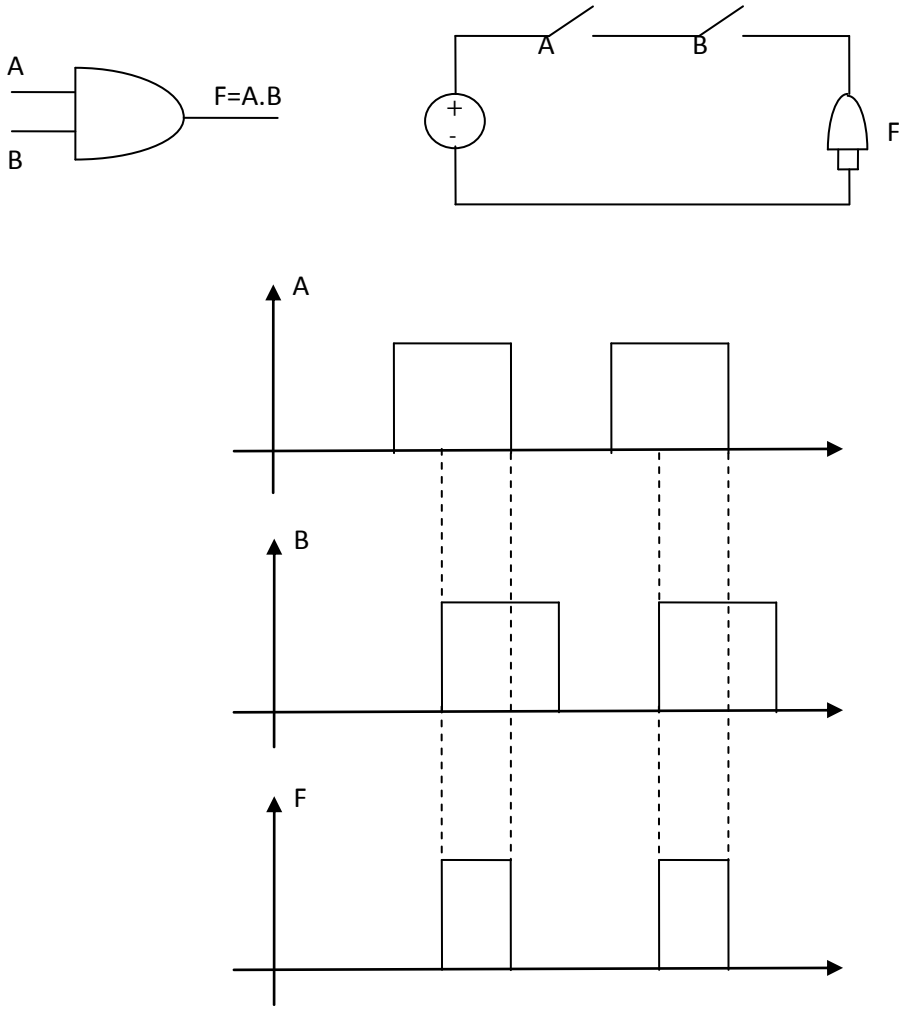
4. LOJİK DEVRE TEMELLERİ

Lojik devreler ikili işaretler veya ikili kodlanmış veriler üzerinde çalışan ve temeli Boole cebrine dayanan düzeneklerdir. Lojik devrelerde biri "lojik 0" diğeri "lojik 1" olarak adlandırılan iki durum vardır. Bilgisayarlar dahil tüm sayısal sistemler bu iki lojik değerin farklı şekilde kombinasyonları yapılarak tasarlanır.

4.1. AND (VE) Kapısı

Birbirine VE işlemi ile bağlı iki önermeden oluşan bir birleşik önermenin doğru olması, her iki önermenin de doğru olmasına bağlıdır.

A	B	$F = A.B$
0	0	0
0	1	0
1	0	0
1	1	1

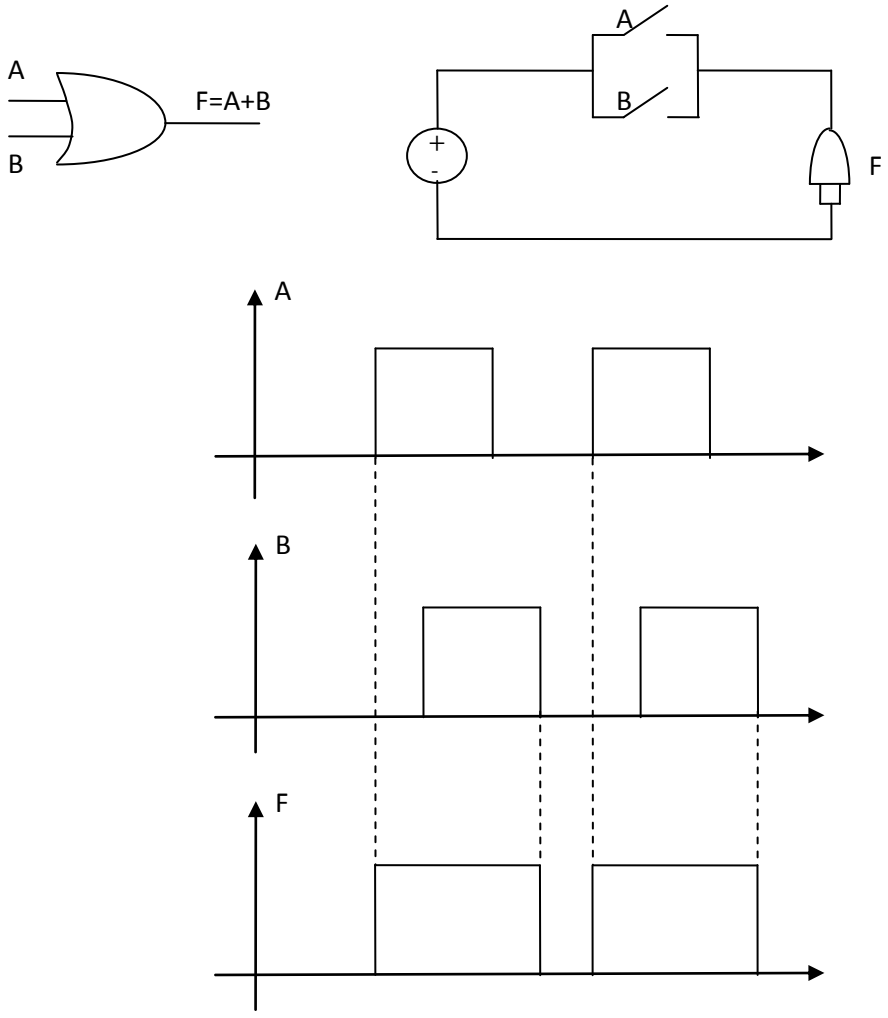


Şekil 4.1. AND kapısı, anahtar devrelerindeki karşılığı ve zamanlama diyagramı

4.2. OR (VEYA) Kapısı

Birbirine VEYA işlemi ile bağlı iki önermeden oluşan bir birleşik önermenin doğru olması, birleşik önermeyi meydana getiren önermelerden en az birinin doğru olmasına bağlıdır.

A	B	$F = A + B$
0	0	0
0	1	1
1	0	1
1	1	1

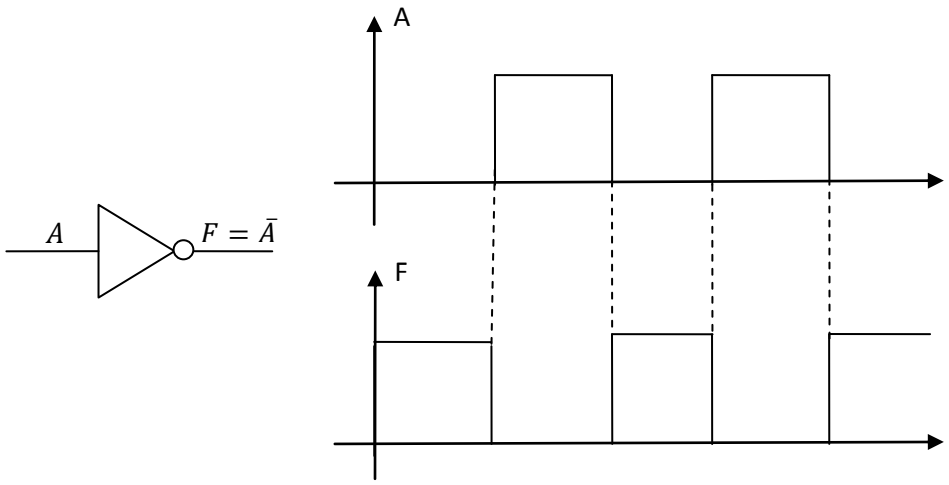


Şekil 4.2. OR kapısı, anahtar devrelerindeki karşılığı ve zamanlama diyagramı

4.3. NOT (DEĞİL) Kapısı

"NOT" işlemi uygulanan önerme, başlangıçta doğru ise yanlış, yanlış ise doğru olacaktır.

A	$F = \bar{A}$
0	1
1	0

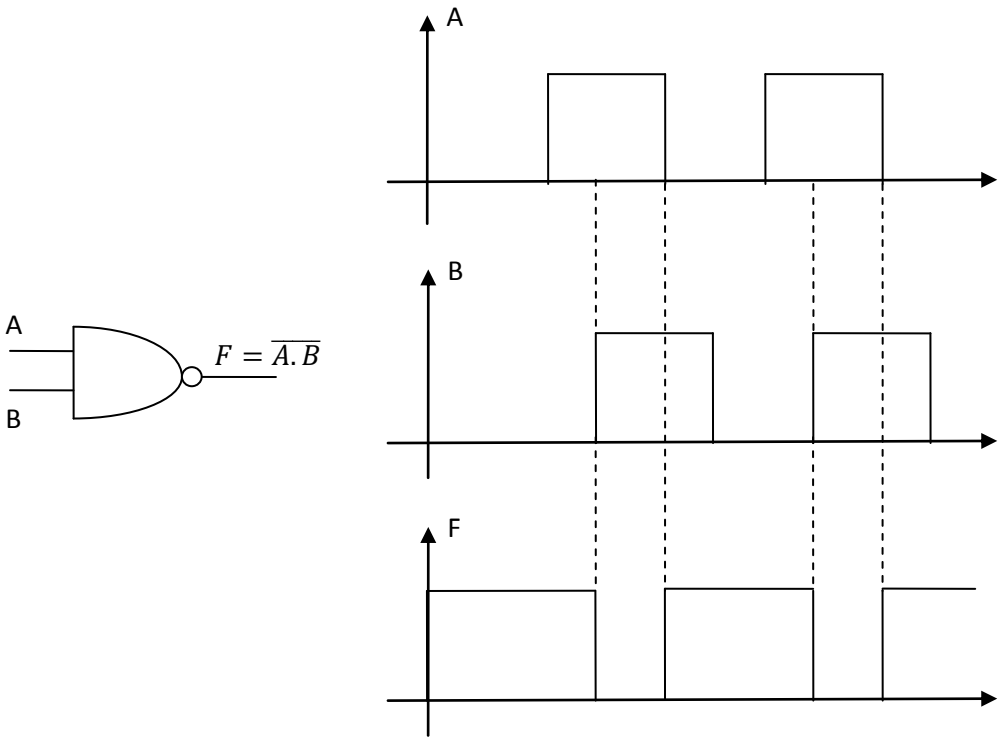


Şekil 4.3. NOT kapısı ve zamanlama diyagramı

4.4. NAND (VE DEĞİL) Kapısı

Birbirine VE DEĞİL işlemi ile bağlı iki önermeden oluşan bir birleşik önermenin yanlış olması, her iki önermenin de doğru olmasına bağlıdır.

A	B	$F = \overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

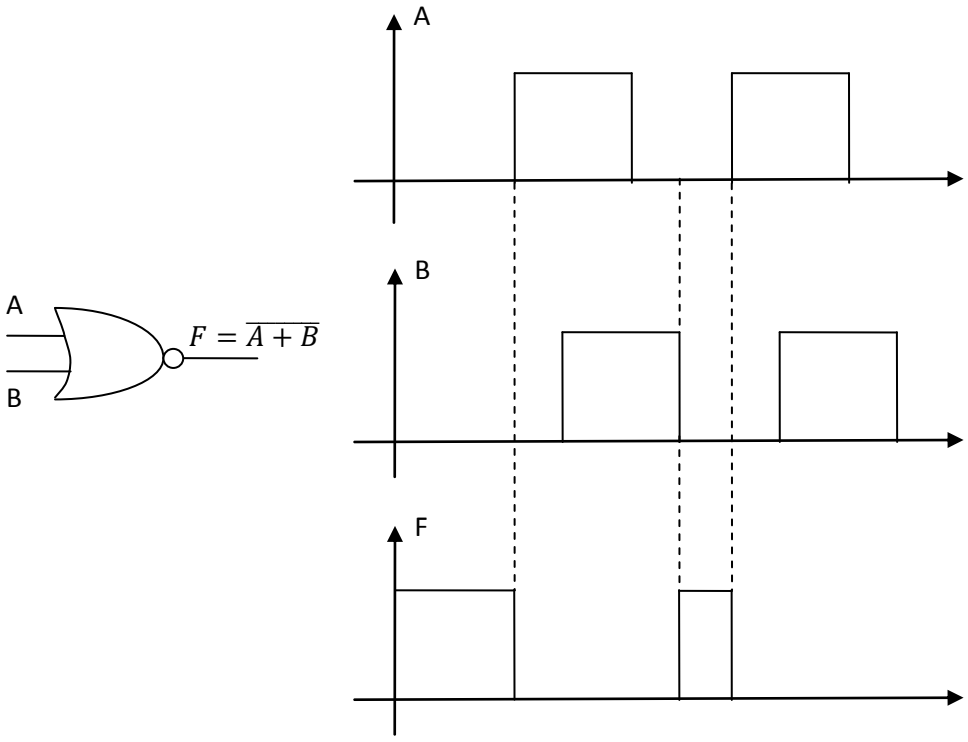


Şekil 4.4. NAND kapısı ve zamanlama diyagramı

4.5. NOR (VEYA DEĞİL) Kapısı

Birbirine VEYA DEĞİL işlemi ile bağlı iki önermeden oluşan bir birleşik önermenin doğru olması, her iki önermenin de yanlış olmasına bağlıdır.

A	B	$F = \overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0



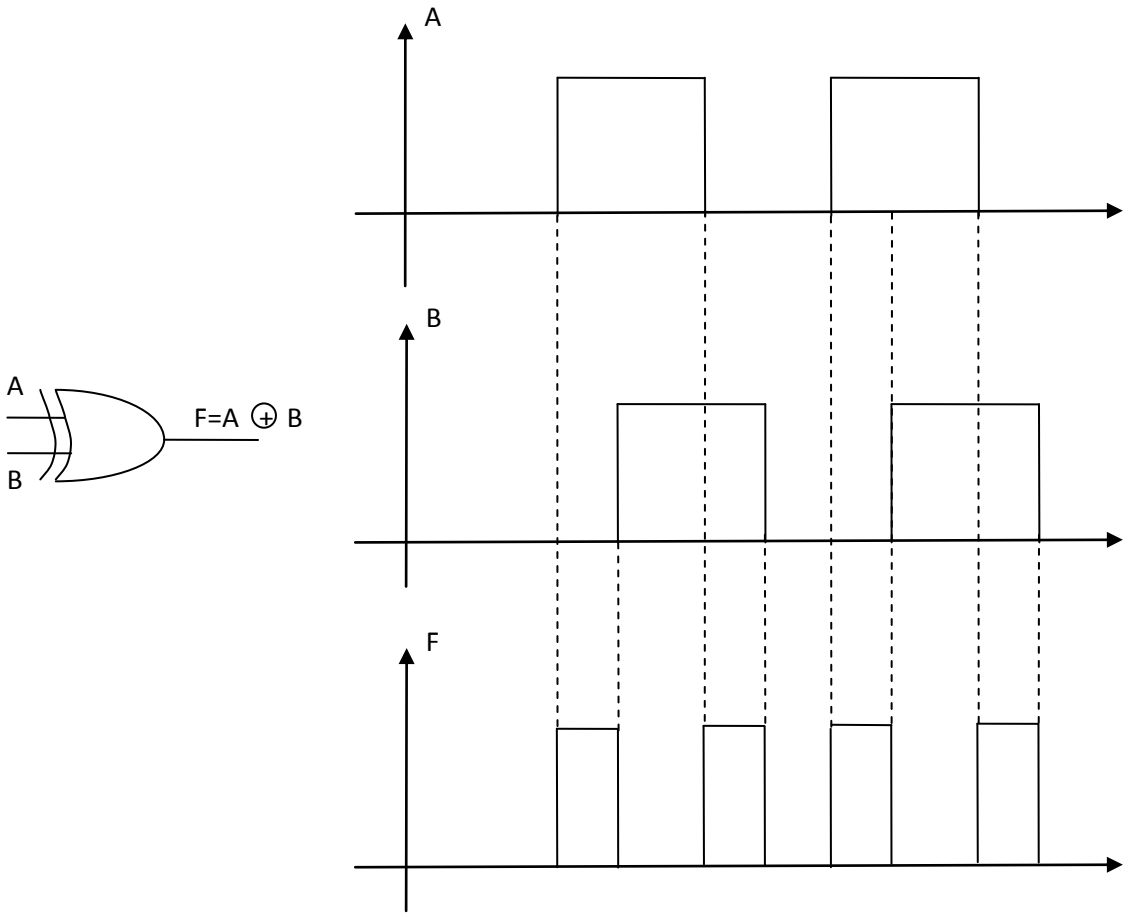
Şekil 4.5. NOR kapısı ve zamanlama diyagramı

4.6. XOR (ÖZEL VEYA) Kapısı

Birbirine ÖZEL VEYA işlemi ile bağlı iki önermeden oluşan bir birleşik önermenin doğru olması, birleşik önermeyi meydana getiren önermelerden birinin doğru diğerinin yanlış olmasına bağlıdır.

A	B	$F = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

$F = A \oplus B = \bar{A}B + A\bar{B}$



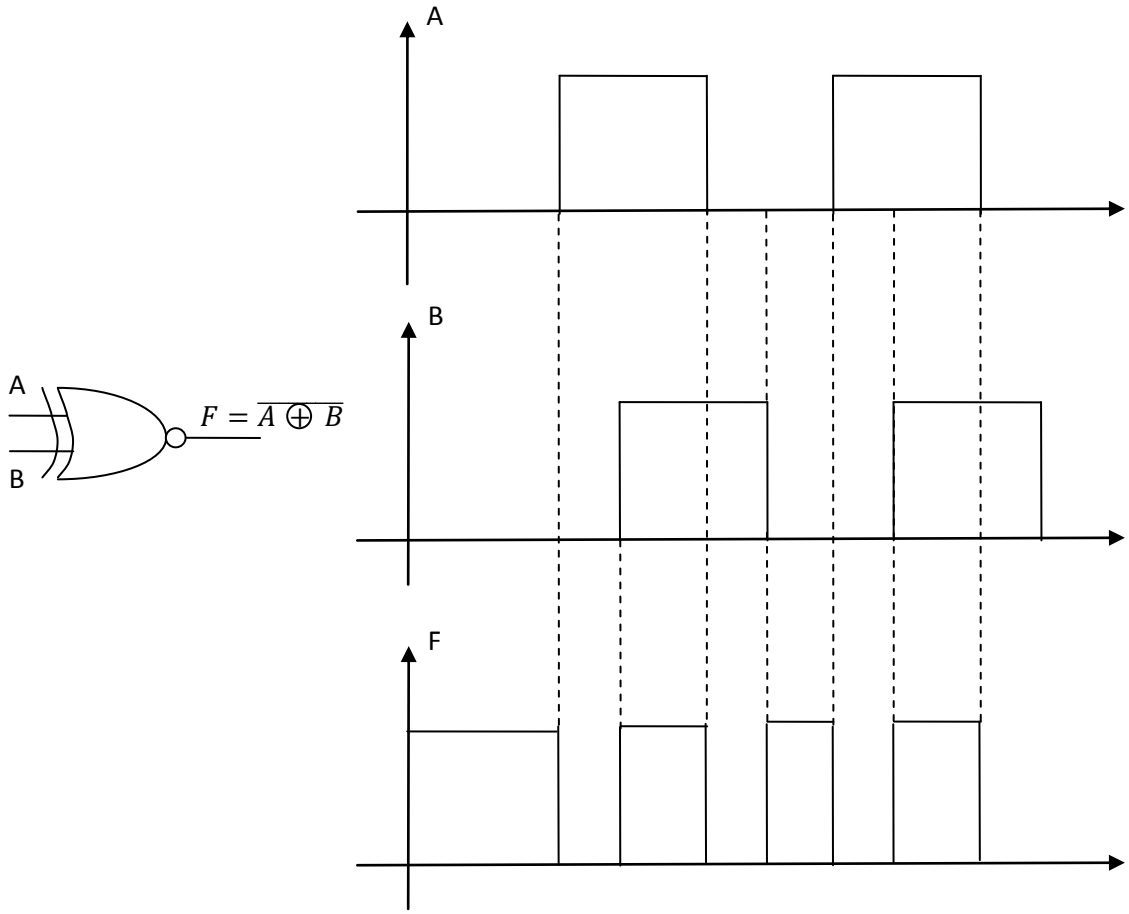
Şekil 4.6. XOR kapısı ve zamanlama diyagramı

4.7. XNOR (ÖZEL VEYA DEĞİL) Kapısı

Birbirine ÖZEL VEYA DEĞİL işlemi ile bağlı iki önermeden oluşan bir birleşik önermenin doğru olması, her iki önermenin de yanlış olmasına veya her iki önermenin de doğru olmasına bağlıdır.

A	B	$F = \overline{A \oplus B}$
0	0	1
0	1	0
1	0	0
1	1	1

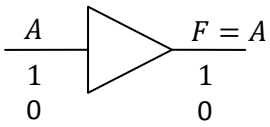
$F = A \odot B = AB + \bar{A}\bar{B}$



Şekil 4.7. XNOR kapısı ve zamanlama diyagramı

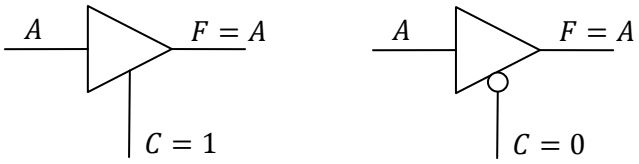
4.8. TAMPON (BUFFER) Elemanı

Akım kuvvetlendirmek amacıyla entegrenin çıkışına bağlanır.



Şekil 4.8. TAMPON Elemanı

4.9. THREE STATE BUFFER



Şekil 4.9. THREE-STATE BUFFER

5. BOOLEAN CEBRİ AKSİYOM VE TEOREMLERİ

Her disiplinde olduğu gibi lojik devre tasarımı da belirli kurallar çerçevesinde yapılır. Lojik devrelerde bu kurallar kümesinin dayanağı Boole Cebri' dir. Bütün cebirsel yapılarda olduğu gibi Boole Cebri' nde de doğru olarak kabul edilen ve doğruluğu ispatlanabilen önermeler olmak üzere iki temel kurallar dizisi vardır. Doğru olarak kabul edilen önermelere "*aksiyom*", doğruluğu ispatlanabilen önermelere ise "*teorem*" adı verilir.

5.1. Boole Cebri Aksiyomları

"0" ve "1" ikilisinden oluşan bir B kümesine "+" ve "·" işlemleri uygulanmış olsun.

1. Her bir değişken "0" veya "1" değerinden sadece birini alabilir. Değişken "1" değerini almıyor ise değeri "0" dır.

$$a \neq 0 \Longrightarrow a = 1$$

$$a \neq 1 \Longrightarrow a = 0$$

2. a) $1 + 1 = 1$ Birbirine VEYA ile bağlı iki önermenin ikisi de doğru ise birleşik önerme de doğrudur.

   b) $0 \cdot 0 = 0$ Birbirine VE ile bağlı iki önermenin ikisi de yanlış ise birleşik önerme de yanlıştır.

3. a) $0 + 0 = 0$ Birbirine VEYA ile bağlı iki önermenin ikisi de yanlış ise birleşik önerme de yanlıştır.

   b) $1 \cdot 1 = 1$ Birbirine VE ile bağlı iki önermenin ikisi de doğru ise birleşik önerme de doğrudur.

4. a) $1 + 0 = 1$ Birbirine VEYA ile bağlı iki önermeden biri doğru ise birleşik önerme de doğrudur.

   b) $0 \cdot 1 = 0$ Birbirine VE ile bağlı iki önermeden birisi yanlış ise birleşik önerme de yanlıştır.

5.2. Boole Cebri Teoremleri

1. a) $a + b = b + a$ Değişme Özelliği

   $b)\ a \cdot b = b \cdot a$

2. a) $a + b + c = (a + b) + c = a + (b + c)$ Birleşme Özelliği

   $b)\ a \cdot b \cdot c = (a \cdot b) \cdot c = a \cdot (b \cdot c)$

3. a) $a + b \cdot c = (a + b) \cdot (a + c)$ Dağılma Özelliği

   $b)\ a \cdot (b + c) = (a \cdot b) + (a \cdot c)$

4. a) $a + a = a$ Değişkende Fazlalık Özelliği
   $b)\ a \cdot a = a$
5. a) $a + a.b = a$ Yutma Özelliği
   $b)\ a \cdot (a + b) = a$
6. a) $\overline{(a)} = \bar{a}$ İşlemde Fazlalık Özelliği
   $b)\ \overline{(\bar{a})} = a$
7. a) $\overline{(a + b + c + \cdots)} = \bar{a} \cdot \bar{b} \cdot \bar{c} \cdot \cdots$ De Morgan Kuralı
   $b)\ \overline{(a \cdot b \cdot c \cdot \cdots)} = \bar{a} + \bar{b} + \bar{c} + \cdots$
8. a) $a + \bar{a} = 1$ Sabit Özelliği
   $b)\ a \cdot \bar{a} = 0$
9. a) $0 + a = a$ Etkisizlik Özelliği
   $b)\ 1 \cdot a = a$
10. a) $1 + a = 1$ Yutan Sabit Özelliği
    $b)\ 0 \cdot a = 0$
11. a) $(a + \bar{b}) \cdot b = a \cdot b$
    $b)\ a \cdot \bar{b} + b = a + b$
12. a) $(a + b) \cdot (\bar{a} + c) \cdot (b + c) = (a + b) \cdot (\bar{a} + c)$
    $b)\ a \cdot b + \bar{a} \cdot c + b \cdot c = a \cdot b + \bar{a} \cdot c$
13. a) $(a + b) \cdot (\bar{a} + c) = a \cdot c + \bar{a} \cdot b$
    $b)\ a \cdot b + \bar{a} \cdot c = (a + c) \cdot (\bar{a} + b)$
14. a) $f(a, b, c, d, \cdots) = [a + f(0, b, c, d, \cdots)] \cdot [\bar{a} + f(1, b, c, d, \cdots)]$ Shannon Teoremi
    $b)\ f(a, b, c, d, \cdots) = [a \cdot f(1, b, c, d, \cdots)] + [\bar{a} \cdot f(0, b, c, d, \cdots)]$

Örnek 5.1. $F = A.(\overline{B + C}) + \bar{A} + (\bar{B} \cdot C) \Longrightarrow \bar{F} = ?$

Verilen fonksiyona De Morgan Teoremini uygulayınız.

$$\bar{F} = \overline{A \cdot (\overline{B + C}) + \bar{A} + (\bar{B} \cdot C)}$$

$$\bar{F} = \bar{A} + \overline{(\overline{B + C})} \cdot \overline{(\bar{A})} \cdot \overline{(\bar{B} \cdot C)}$$

$$\bar{F} = (\bar{A} + B + C) \cdot A \cdot (B + \bar{C})$$

$$\bar{F} = (\bar{A} + B + C) \cdot (AB + A\bar{C})$$

Örnek 5.2. $F_1 = \bar{x}yz + \bar{x}\bar{y}z \quad F_2 = x(\bar{y}\bar{z} + yz)$

Verilen fonksiyonların terslerini bulunuz.

$\overline{F_1} = \overline{\bar{x}yz + \bar{x}\bar{y}z} = \overline{(\bar{x}yz)} \cdot \overline{(\bar{x}\bar{y}z)} = (x + \bar{y} + \bar{z}) \cdot (x + y + \bar{z}) \Longrightarrow$ De Morgan Teoremi

$\overline{F_2} = \overline{x(\bar{y}\bar{z} + yz)} = \bar{x} + \big((y + z) \cdot (\bar{y} + \bar{z})\big) \Longrightarrow$ De Morgan Teoremi

Dualite Prensibi

Kuralların ispatı doğruluk tabloları yapılarak gerçeklenebilir. Ayrıca bu kurallar AND ve OR işlemleri ile de yazılabilir.

Bir Boole ifadesinin duali mantıksal çarpım ve toplamların ve 1 ile 0' ların yer değiştirmesiyle bulunur.

$x(y+0)$ ifadesinin duali $x+(y\cdot 1)$ olarak bulunur.

$\bar{x}\cdot 1+(\bar{y}\cdot z)$ ifadesinin duali $(\bar{x}+0)\cdot(\bar{y}+z)$ olarak bulunur.

5.3. Minimum ve Maksimum Terimler

x ve y şeklinde iki terim için minimum terimler (minterm) ve maksimum terimler (maksterm) aşağıdaki tabloda verilmiştir.

x	y	Minterm	m'ye indis	Maksterm	M' ye indis
0	0	$\bar{x}\cdot\bar{y}$	m_0	$x+y$	M_0
0	1	$\bar{x}\cdot y$	m_1	$x+\bar{y}$	M_1
1	0	$x\cdot\bar{y}$	m_2	$\bar{x}+y$	M_2
1	1	$x\cdot y$	m_3	$\bar{x}+\bar{y}$	M_3

5.3.1. Minimum terimler kanonik biçimi

(Doğruluk tablosu kullanarak çarpımların toplamı çözümü)

Minimum terimlerin toplamından oluşmuş ifadeye *Minimum Terimler Kanonik Biçimi (Çarpımlar Toplamı Kanonik Biçimi)* denir. Minimum terimler m_i şeklinde gösterildiklerine göre, bir Boole fonksiyonuna ilişkin minimum terimler kanonik biçimi $\sum m_i$ şeklinde gösterilir. Kanonik kelimesi lojik fonksiyonu oluşturan terimlerin ya kendilerinin ya da tümleyenlerinin çarpım terimlerin içinde mutlaka bulunması anlamına gelmektedir.

Örnek 5.3. VEYA işlemine ait minimum terimler kanonik biçimini bulunuz.

a	b	$F=a+b$	Minterm	m'ye indis
0	0	0	$\bar{a}\cdot\bar{b}$	m_0
0	1	1	$\bar{a}\cdot b$	m_1
1	0	1	$a\cdot\bar{b}$	m_2
1	1	1	$a\cdot b$	m_3

$$F=a+b=\bar{a}\cdot b+a\cdot\bar{b}+a\cdot b=m_1+m_2+m_3=\sum(1,2,3)$$

Örnek 5.4. Bir elektrik motorunun akım, gerilim ve gövde sıcaklığı kontrol edilecektir. Bu değişkenlerden herhangi ikisi istenen sınır değerleri aştığında ikaz, her üçü beraber istenen sınır değeri aşarsa devre dışı butonları devreye girecektir. İlgili devrenin kontrol ünitesini minimum terimler kanonik biçimi (çarpımların toplamı çözümü) kullanarak gerçekleştirelim.

	A	V	T	i	DD
0	0	0	0	0	0
1	0	0	1	0	0
2	0	1	0	0	0
3	0	1	1	1	0
4	1	0	0	0	0
5	1	0	1	1	0
6	1	1	0	1	0
7	1	1	1	1	1

$m_3 = \bar{A}VT$

$m_5 = A\bar{V}T$

$m_6 = AV\bar{T}$

$m_7 = AVT$

$$i = \bar{A}VT + A\bar{V}T + AV\bar{T} + AVT = \sum(m_3 + m_5 + m_6 + m_7) \qquad DD = AVT$$

$$i(A,V,T) = \sum(3,5,6,7)$$

5.3.2. Maksimum terimler kanonik biçimi

(Doğruluk tablosu kullanarak toplamların çarpımı çözümü)

Maksimum terimlerin çarpımından oluşmuş ifadeye *Maximum Terimler Kanonik Biçimi (Toplamların Çarpımı Kanonik Biçimi)* denir. Maksimum terimler M_i şeklinde gösterildiklerine göre, bir Boole fonksiyonuna ilişkin maksimum terimler kanonik biçimi $\prod M_i$ şeklinde gösterilir.

Örnek 5.5. VE işlemine ait maksimum terimler kanonik biçimini bulunuz.

a	b	$F = a \cdot b$	Maksterm	M'ye indis
0	0	0	$a + b$	M_0
0	1	0	$a + \bar{b}$	M_1
1	0	0	$\bar{a} + b$	M_2
1	1	1	$\bar{a} + \bar{b}$	M_3

$$F = a \cdot b = (a + b) \cdot (a + \bar{b}) \cdot (\bar{a} + b) = M_0 \cdot M_1 \cdot M_2 = \prod(0,1,2)$$

Örnek 5.6. Yukarıdaki örnekte verilen motor problemini maksimum terimler kanonik biçimi (toplamların çarpımı çözümü) kullanarak gerçekleştirelim.

	A	V	T	i	$\bar{\imath}$
0	0	0	0	0	1
1	0	0	1	0	1
2	0	1	0	0	1
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	1	0
6	1	1	0	1	0
7	1	1	1	1	0

$M_0 = A + V + T$

$M_1 = A + V + \bar{T}$

$M_2 = A + \bar{V} + T$

$M_4 = \bar{A} + V + T$

Problemin çözümünde iki yol izlenebilir.

Birinci yol: Burada önce gerçekleştirilecek fonksiyonun tersi alınır.

$$\bar{\imath}(A,V,T) = \sum(0,1,2,4)$$

$$\bar{\imath} = \bar{A}\bar{V}\bar{T} + \bar{A}\bar{V}T + \bar{A}V\bar{T} + A\bar{V}\bar{T}$$

$$\overline{(\bar{\imath})} = \overline{\bar{A}\bar{V}\bar{T} + \bar{A}\bar{V}T + \bar{A}V\bar{T} + A\bar{V}\bar{T}}$$

$$i = (A + V + T) \cdot (A + V + \bar{T}) \cdot (A + \bar{V} + T) \cdot (\bar{A} + V + T)$$

İkinci yol: Tersini almaya gerek yoktur. Doğrudan fonksiyonun sıfır olduğu yerlere bakılır.

$$i = (A + V + T) \cdot (A + V + \bar{T}) \cdot (A + \bar{V} + T) \cdot (\bar{A} + V + T)$$

$$i = (M_0 \cdot M_1 \cdot M_2 \cdot M_4) = \prod(M_0, M_1, M_2, M_4) = \prod(0,1,2,4)$$

6. LOJİK FONKSİYONLARIN SADELEŞTİRİLMESİ

Lojik fonksiyonların indirgenmesinde amaç, lojik ifadenin farklı giriş değerlerine göre çıkış değerinin değişikliğe uğratılmadan daha az sayıda terimle ifade edilmesidir. Böylece daha az maliyetli tasarımlar yapılabilir ve yalın ifadelerle uğraşma imkânı doğar.

Lojik fonksiyonların sadeleştirilmesinde en çok kullanılan iki yöntem şunlardır:

1. Karnaugh Diyagramı Yöntemi
2. Quine-McCluskey Tablo Yöntemi

6.1. Karnaugh Diyagramı ile Sadeleştirme

- 2 değişkenli bir problem için Karnaugh diyagramının gerçekleştirilmesi

x	y	$f(x,y)$
0	0	m_0
0	1	m_1
1	0	m_2
1	1	m_3

x \ y	0	1
0	m_0	m_1
1	m_2	m_3

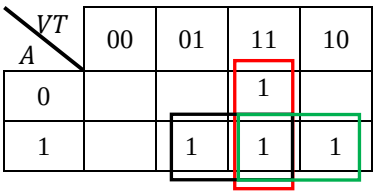
- 3 değişkenli bir problem için Karnaugh diyagramının gerçekleştirilmesi

x	y	z	$f(x,y,z)$
0	0	0	m_0
0	0	1	m_1
0	1	0	m_2
0	1	1	m_3
1	0	0	m_4
1	0	1	m_5
1	1	0	m_6
1	1	1	m_7

		$\bar{y}$	$\bar{y}$	y	y
	x \ yz	00	01	11	10
$\bar{x}$	0	m_0	m_1	m_3	m_2
x	1	m_4	m_5	m_7	m_6
		$\bar{z}$	z	z	$\bar{z}$

<u>Örnek 6.1.</u> Örnek 5.4 ile verilen problemde elde edilen fonksiyonun Karnaugh diyagramı kullanılarak sadeleştirilmesi.

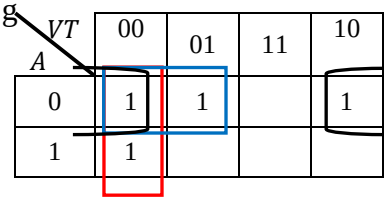
	A	V	T	i	DD	
0	0	0	0	0	0	
1	0	0	1	0	0	
2	0	1	0	0	0	
3	0	1	1	1	0	$m_3 = \bar{A}VT$
4	1	0	0	0	0	
5	1	0	1	1	0	$m_5 = A\bar{V}T$
6	1	1	0	1	0	$m_6 = AV\bar{T}$
7	1	1	1	1	1	$m_7 = AVT$

- 

A \ VT	00	01	11	10
0			1	
1		1	1	1

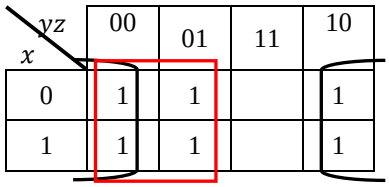
$i = \bar{A}VT + A\bar{V}T + AV\bar{T} + AVT$

$i = VT + AT + AV$

- g 

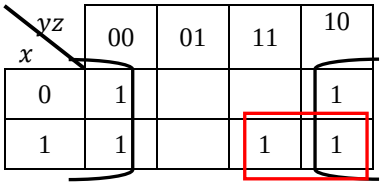
A \ VT	00	01	11	10
0	1	1		1
1	1			

$F = \bar{V}\bar{T} + \bar{A}\bar{V} + \bar{A}\bar{T}$

- 

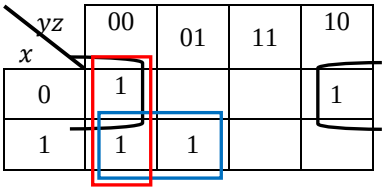
x \ yz	00	01	11	10
0	1	1		1
1	1	1		1

$F = \bar{y} + \bar{z}$

- 

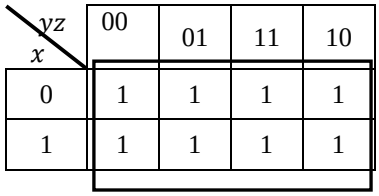
x \ yz	00	01	11	10
0	1			1
1	1		1	1

$F = \bar{z} + xy$

- 

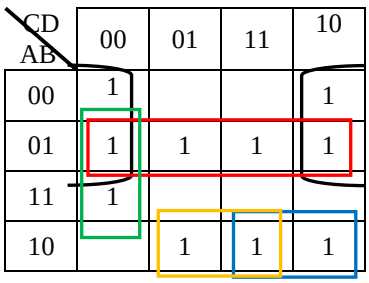
x \ yz	00	01	11	10
0	1			1
1	1	1		

$F = \bar{y}\bar{z} + x\bar{y} + \bar{x}\bar{z}$

- 

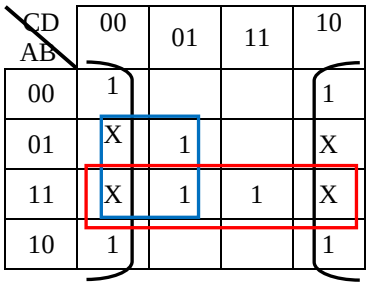
x \ yz	00	01	11	10
0	1	1	1	1
1	1	1	1	1

$F = 1$

- 

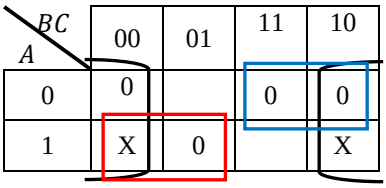
AB \ CD	00	01	11	10
00	1			1
01	1	1	1	1
11	1			
10		1	1	1

$F = \bar{A}B + \bar{A}\bar{D} + AD\bar{B} + A\bar{B}C + B\bar{C}\bar{D}$

- 

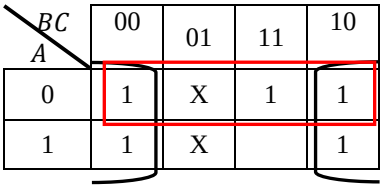
CD \ AB	00	01	11	10
00	1			1
01	X	1		X
11	X	1	1	X
10	1			1

$F = \bar{D} + B\bar{C} + AB$

- 

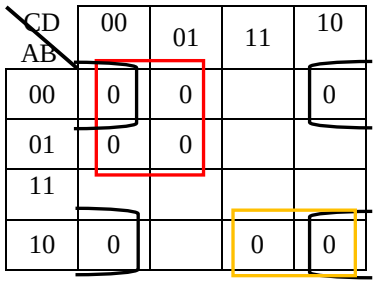
BC \ A	00	01	11	10
0	0		0	0
1	X	0		X

$F = C \cdot (\bar{A} + B) \cdot (A + \bar{B})$

- 

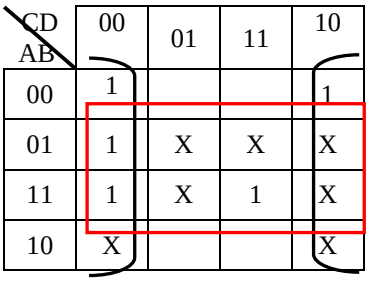
BC \ A	00	01	11	10
0	1	X	1	1
1	1	X		1

$F = \bar{C} + \bar{A}$

- 

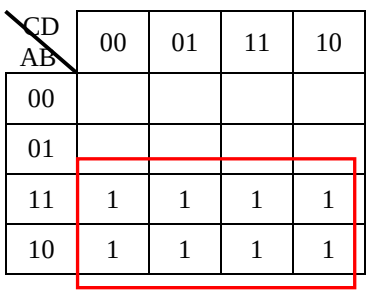
CD \ AB	00	01	11	10
00	0	0		0
01	0	0		
11				
10	0		0	0

$F = (B + D) \cdot (A + C) \cdot (\bar{A} + B + \bar{C})$

- 

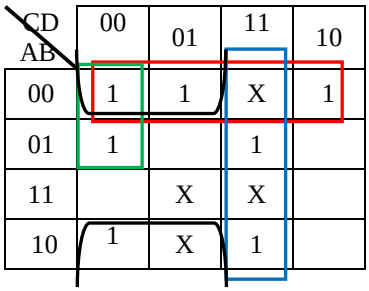
CD \ AB	00	01	11	10
00	1			1
01	1	X	X	X
11	1	X	1	X
10	X			X

$F = \bar{D} + B$

- 

CD \ AB	00	01	11	10
00				
01				
11	1	1	1	1
10	1	1	1	1

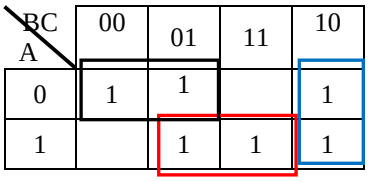
$F = A$

- 

CD \ AB	00	01	11	10
00	1	1	X	1
01	1		1	
11		X	X	
10	1	X	1	

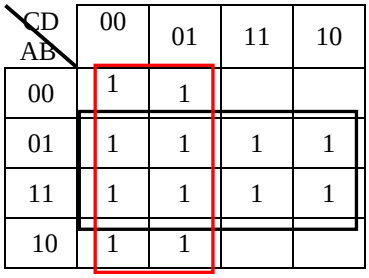
$F(A,B,C,D) = \sum(0,1,2,4,7,8,11) + \Phi(3,9,13,15)$

$F(A,B,C,D) = \bar{A}\bar{B} + CD + \bar{A}\bar{C}\bar{D} + \bar{B}\bar{C}$

- 

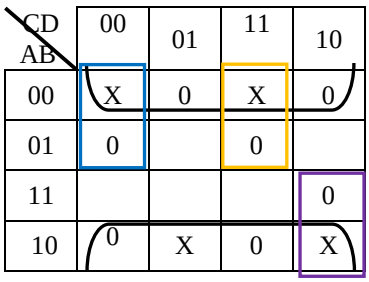
BC \ A	00	01	11	10
0	1	1		1
1		1	1	1

$F(A,B,C,D) = \bar{A}\bar{B} + AC + B\bar{C}$

- 

CD \ AB	00	01	11	10
00	1	1		
01	1	1	1	1
11	1	1	1	1
10	1	1		

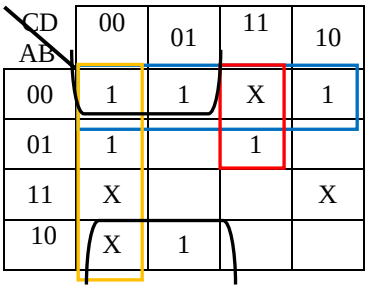
$F(A,B,C,D) = \bar{C} + B$

- 

CD \ AB	00	01	11	10
00	X	0	X	0
01	0		0	
11				0
10	0	X	0	X

$F(A,B,C,D) = \prod(1,2,4,7,8,11,14) + \Phi(0,3,9,10)$

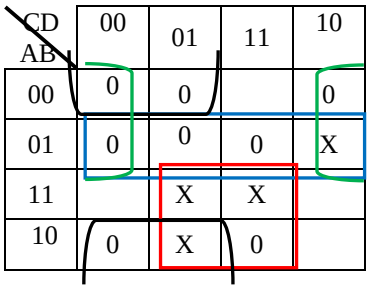
$F = B \cdot (A + C + D) \cdot (A + \bar{C} + \bar{D}) \cdot (\bar{A} + \bar{C} + D)$

- 

CD \ AB	00	01	11	10
00	1	1	X	1
01	1		1	
11	X			X
10	X	1		

$F(A,B,C,D) = \sum(1,2,4,7,9) + \Phi(3,8,12,14)$

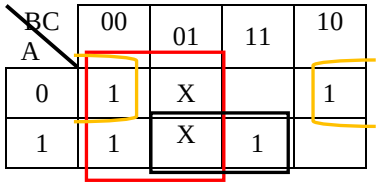
$F = \bar{A}\bar{B} + \bar{C}\bar{D} + \bar{B}\bar{C} + \bar{A}CD$

- 

CD \ AB	00	01	11	10
00	0	0		0
01	0	0	0	X
11		X	X	
10	0	X	0	

$F(A,B,C,D) = \prod(0,1,2,4,5,7,8,11) + \Phi(6,9,13,15)$

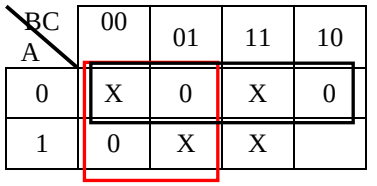
$F = (B + C)(A + \bar{B})(\bar{A} + \bar{D})(A + D)$

- 

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

$F(A,B,C,D) = \sum(0,2,4,7) + \Phi(1,5)$

$F = \bar{B} + \bar{A}\bar{C} + AC$

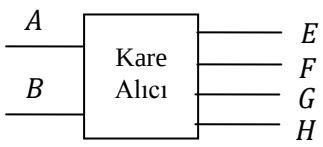
- 

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

$F(A,B,C,D) = \prod(1,2,4,5) + \Phi(0,3,7)$

$F = \bar{B} + \bar{A}$

Örnek 6.2. Girişine giren 2 bitlik sayıların karesini alan lojik devreyi doğruluk tablosu çıkararak hesaplayınız.



A	B	E	F	G	H
0	0	0	0	0	0
0	1	0	0	0	1
1	0	0	1	0	0
1	1	1	0	0	1

$E = A \cdot B$ $G = Lojik\ 0$ $F = A \cdot \bar{B}$ $H = \bar{A}B + AB = B(A + \bar{A}) = B$

Örnek 6.3. $A = A_1A_0$ ve $B = B_1B_0$ sayıları karşılaştırılacaktır. $A > B, A = B$ ve $A < B$ çıkışlarını veren devreyi doğruluk tablosunu çıkartarak Karnaugh diyagramı ile gerçekleştiriniz.

A_1	A_0	B_1	B_0	$A > B$	$A = B$	$A < B$
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0

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

B_1B_0 / A_1A_0	00	01	11	10
00				
01	1			
11	1	1		1
10	1	1		

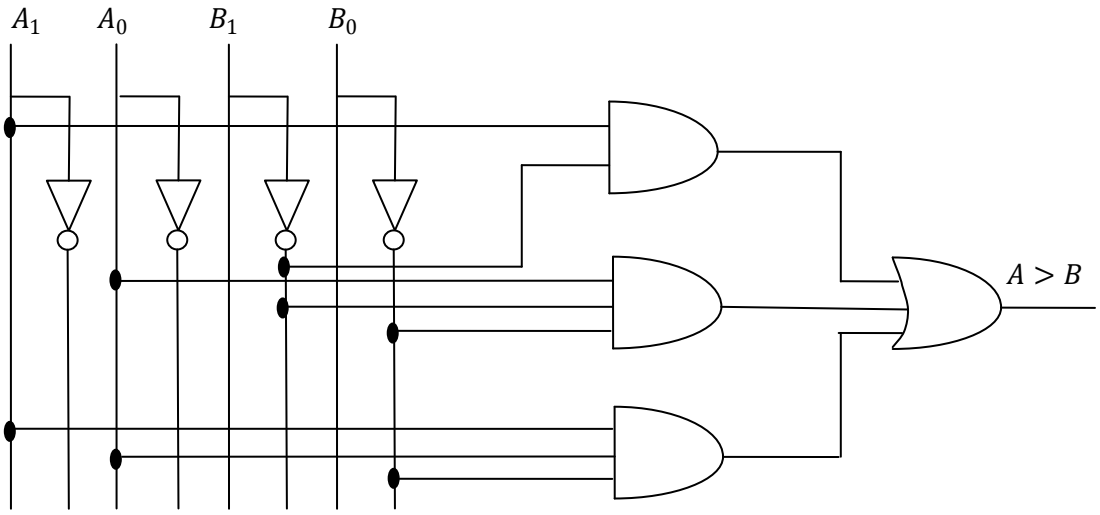
$$A > B \Longrightarrow F_1 = A_1\overline{B_1} + A_0\overline{B_1}\,\overline{B_0} + A_1A_0\overline{B_0}$$

B_1B_0 / A_1A_0	00	01	11	10
00	1			
01		1		
11			1	
10				1

$$A = B \Longrightarrow F_2 = \overline{A_1}\,\overline{A_0}\,\overline{B_1}\,\overline{B_0} + \overline{A_1}\,A_0\,\overline{B_1}\,B_0 + A_1\,A_0\,B_1\,B_0 + A_1\,\overline{A_0}\,B_1\,\overline{B_0}$$

B_1B_0 / A_1A_0	00	01	11	10
00		1	1	1
01			1	1
11				
10			1	

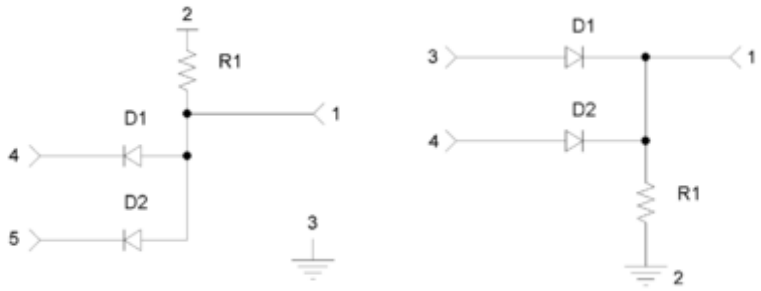
$$A < B \Longrightarrow F_3 = \overline{A_1}\,B_1 + \overline{A_0}\,B_1\,B_0 + \overline{A_1}\,\overline{A_0}\,B_0$$



7. DİJİTAL ENTEGRE LOJİK AİLELER

Üretilen entegre lojik aileler aşağıda sıralanmıştır:

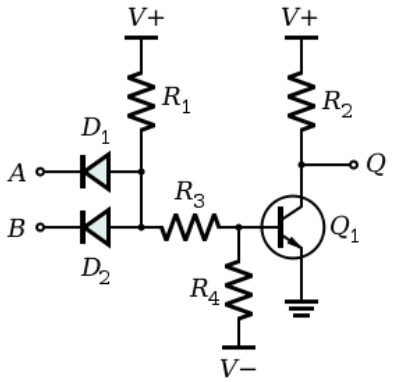
1. DL (Diyot Lojik)



Şekil 7.1. İki girişli AND ve OR kapıları

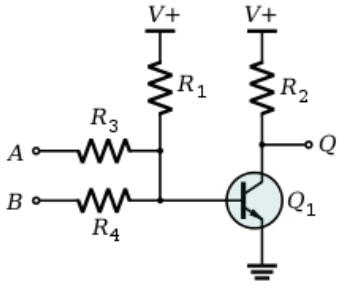
2. TTL (Tranzistör-Tranzistör Lojik): Bipolar teknolojisi ile üretilirler. 74 ve 54 serisi ile ifade edilirler. Bazı özel durumlarda 50 ve 80 serileriyle üretilmektedir.

3. DTL (Diyot-Tranzistör Lojik)



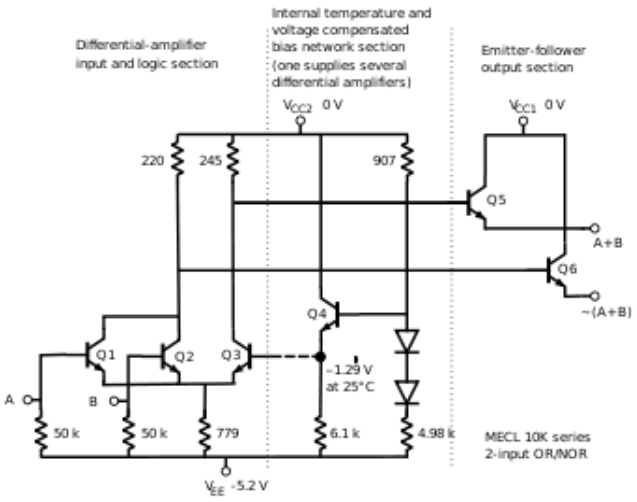
Şekil 7.1. İki girişli DTL NAND kapısı

4. RTL (Direnç-Tranzistör Lojik) TTL ve DTL' den ucuz ve daha az transistor kullanarak yapıldığı için tercih edilebilir.



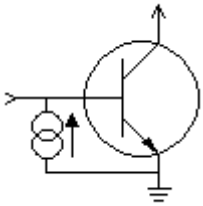
Şekil 7.1. İki girişli RTL NOR kapısı

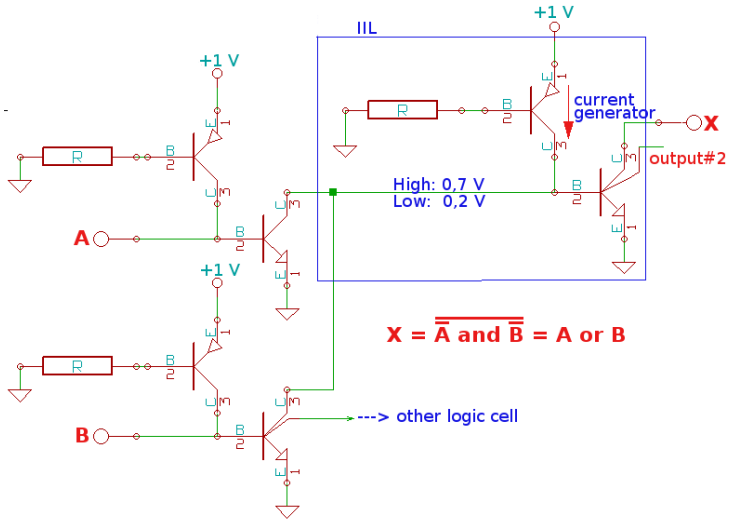
5. ECL (Emitör Kuplajlı Lojik- Emitter Coupled Logic): Hızlı işlem gereken uygulamalarda kullanılır.



Şekil 7.1. Motorola ECL 10,000 basic gate circuit diagram

6. I^2L (Akım Enjekte Edilmiş Lojik) : Eleman yoğunluğunun çok olduğu devrelerde kullanılır.





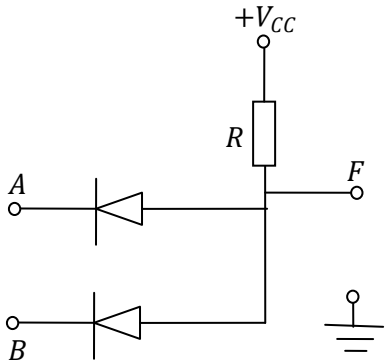
Şekil 7.2. I^2L devresi

7. MOS (Metal Oksit Yarı İletken- Metal Oxide Semiconductor) : Eleman yoğunluğunun çok olduğu devrelerde kullanılır.

8. CMOS (Bütünleştirilmiş MOS- Complementary Metal Oxide Semiconductor): Az enerji harcar. 40 serisi ile ifade edilirler. MOS teknolojisi ile üretilirler.

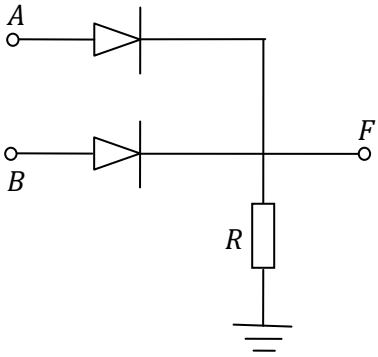
7.1. DL (Diyot Lojik)

AND Kapısı



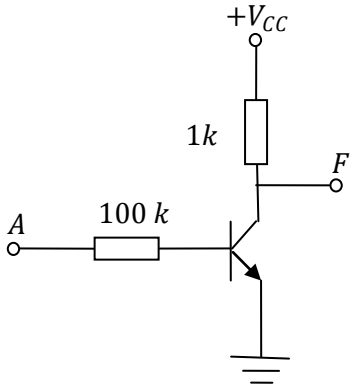
A	B	F
0	0	0
0	1	0
1	0	0
1	1	1

OR Kapısı



A	B	F
0	0	0
0	1	1
1	0	1
1	1	1

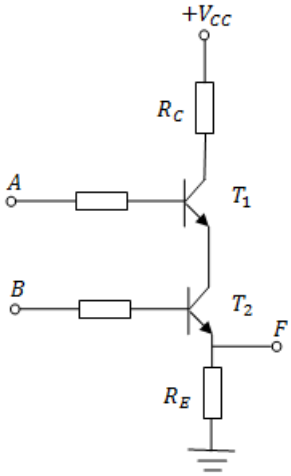
NOT Kapısı



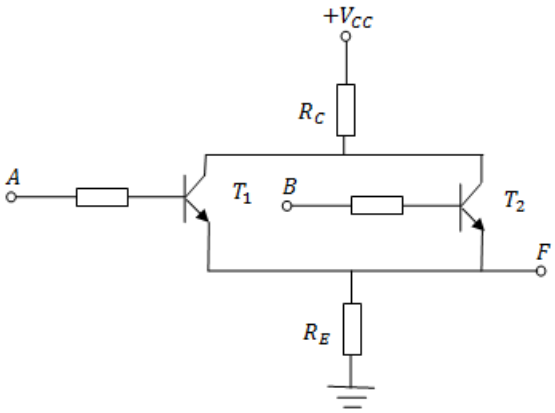
A	F
0	1
1	0

7.2. TL (Tranzistör Lojik)----TTL

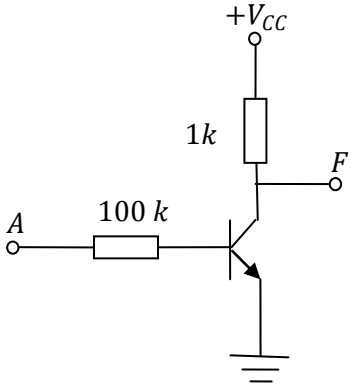
AND Kapısı



OR Kapısı



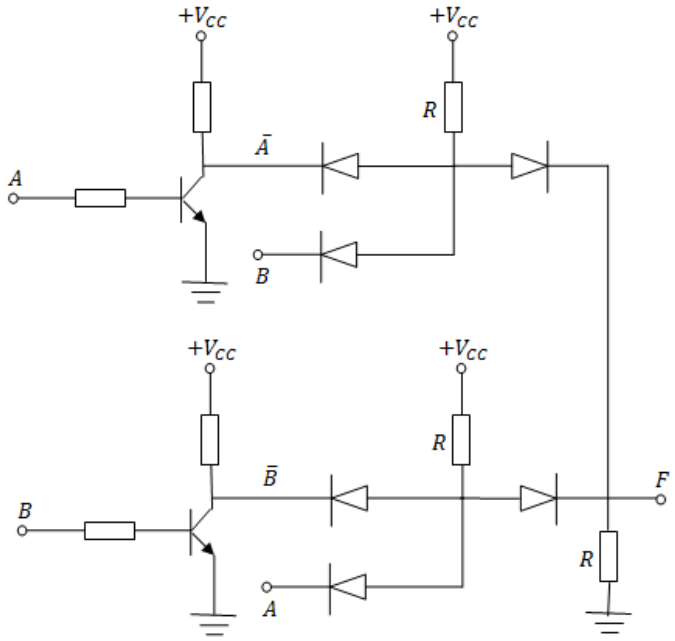
NOT Kapısı



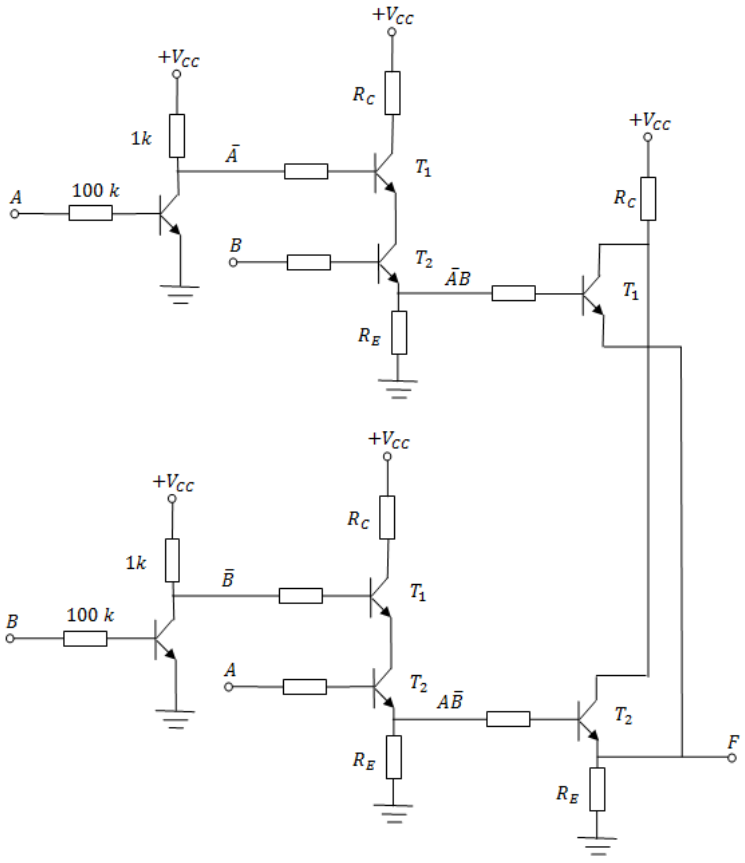
Örnek 7.1. $F = A \oplus B = \bar{A}B + A\bar{B}$ fonksiyonunu

a) DL ve b) TL ile gerçekleştiriniz.

a)



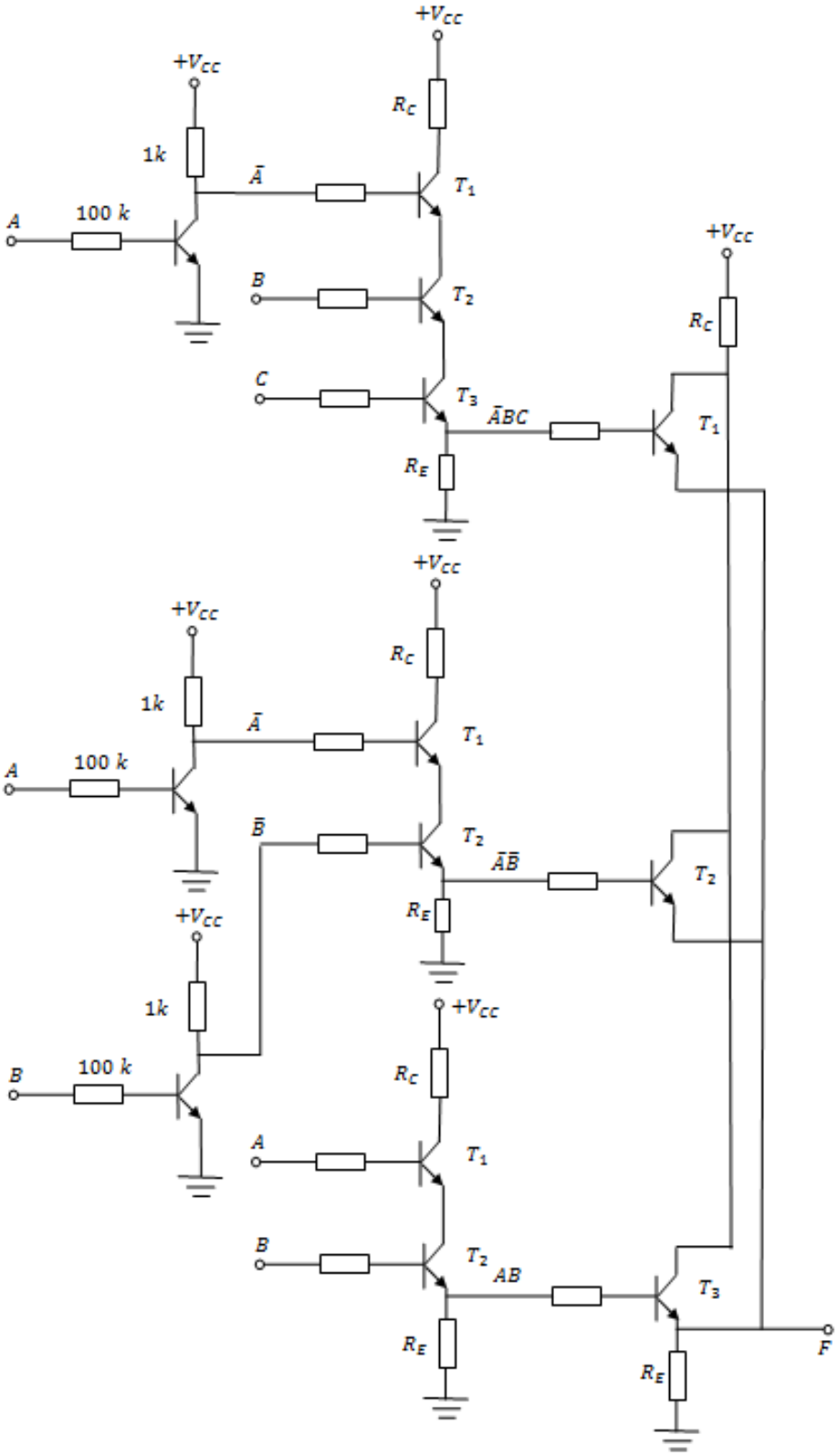
b)



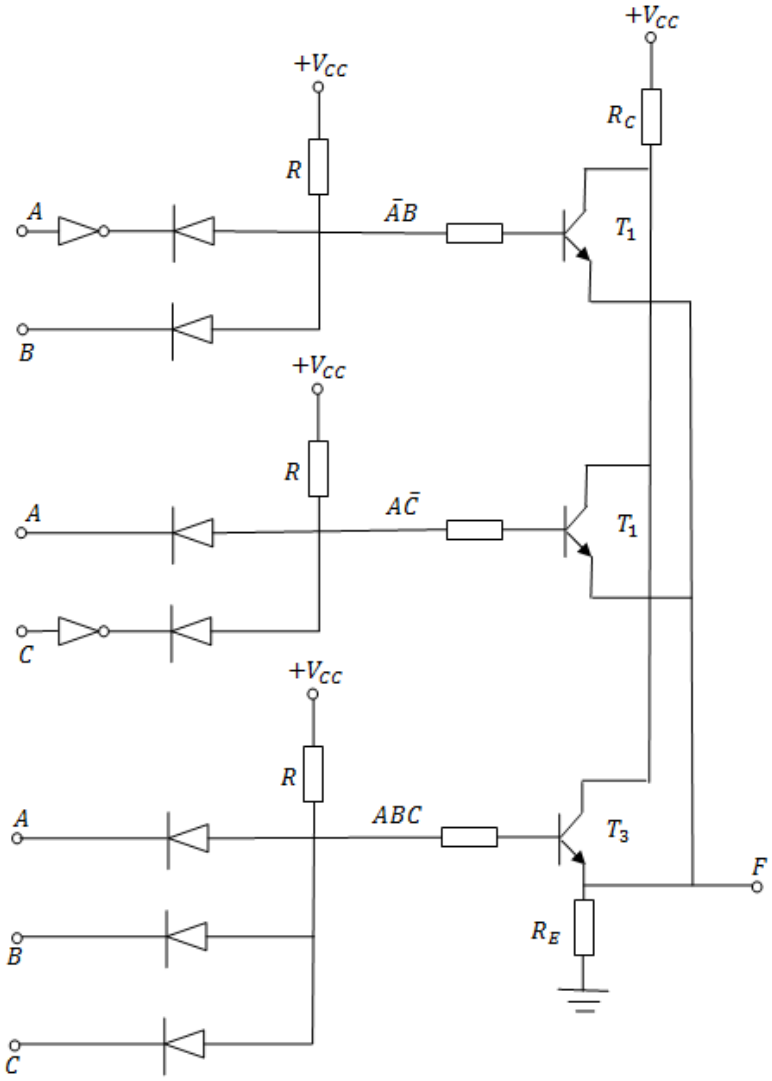
Örnek 7.2. $F = \bar{A}BC + \bar{A}\bar{B} + AB$ fonksiyonunu

a) DTL b) TTL ve c) RTL ile gerçekleştiriniz.

b)



Örnek 7.3. $F = \bar{A}B + A\bar{C} + ABC$ fonksiyonunu DTL ile gerçekleştiriniz.



8. LOJİK DEVRE KATOLOG BİLGİLERİ

Tümdevreler, direnç, kapasitör, diyot, transistör ve diğer elektronik elemanlarla bunların bağlantılarından oluşan küçük silikon ve yarı iletken parçacıklar üzerine gerçekleştirilmiş devreler topluluğudur. İkiye ayrılır:

1. Analog Tümdevreler: Analog işaretleri kullanan devrelerdir. Bu yapıdaki elemanlara işlemsel kuvvetlendiriciler, gerilim regülatörleri, frekans çoğullayıcılar, modülatörler ve osilatörler örnek olarak verilebilir.

2. Sayısal Tümdevreler: Sayısal işaretleri kullanan devrelerdir. Bu yapıdaki elemanlara lojik kapılar, saklayıcılar, kodlayıcılar, kod çözücüler, sayıcılar, bellek elemanları karşılaştırıcılar, veri çoğullayıcılar ve veri toplayıcılar örnek olarak verilebilir. Sayısal tümdevreler içerdikleri kapı yoğunluklarına göre dört sınıfa ayrılır:

- Küçük Ölçekli Tümdevreler (SSI-Small Scale Integration): Kapı sayısı 10' dan az ise SSI' dan söz edilir.
- Orta Ölçekli Tümdevreler (MSI-Medium Scale Integration): Kapı sayısı 10-100 arasında ise MSI söz konusudur. Kod çeviriciler, kodlayıcılar, çoğullayıcılar, veri toplayıcılar gibi…
- Büyük Ölçekli Tümdevreler (LSI-Large Scale Integration): Kapı sayısı 100-birkaç bin arasında ise LSI' dan söz edilir. Hesaplayıcılar, sayısal saatler ve bellek birimleri gibi…
- Çok Büyük Ölçekli Tümdevreler (VLSI-Very Large Scale Integration): Kapı sayısı birkaç binden büyükse VLSI söz konusudur. Mikrobilgisayarlar, yeni tip mikroişlemciler ve mikrokontrolörler gibi…

Çizelge 8.1. Üretim Teknolojisine Göre Tümdevrelerin Lojik Kodları

Üretim Teknolojisi	**Kodu**
Standart TTL	74
Yüksek Hızlı TTL	74H
Düşük Güç Tüketen TTL	74L
Shottky TTL	74S
Düşük Güç Tüketen Shottky TTL	74LS
İleri Shottky TTL	74AS
Düşük Güç Tüketen İleri Shottky TTL	74ALS
CMOS	40
TTL ile Bağlantı Uyumlu CMOS	74C

Yüksek Hızlı ve TTL Bağlantı Uyumlu CMOS	74HC
Yüksek Hızlı ve TTL Elektriksel Uyumlu CMOS	74HCT

8.1. Kombinasyonel Devreler

7400----Dörtlü iki girişli NAND

7402----Dörtlü iki girişli NOR

7404----Altılı tümleme

7408----Dörtlü iki girişli AND

8.2. Tümleşik Kombinasyonel Devreler

Belirli bir işi yerine getiren kapılarla tasarlanmış devrelerdir. Bu devreler içinde yarı ve tam toplayıcılar, seçiciler, kodlayıcılar, PAL (Programlanabilir Dizi Elemanı), PLA (Programlanabilir Lojik Dizi Elemanı) ve çoğullayıcılar gibi elemanlar sayılabilir.

8.3. Ardışıl Devreler

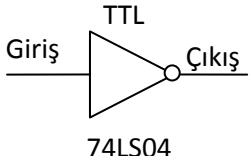
Ardışıl devreler, kendi içerisine bellek elemanı barındıran lojik devrelerde kullanılmaktadır. En temelde "flip-flop" adı verilen elemanlar kullanarak tasarlanırlar. Bilinen en temel ardışıl devreler sayıcılar, saklayıcı, mikroişlemci, ALU' dur.

8.4. TTL ve CMOS Tümdevre Özellikleri

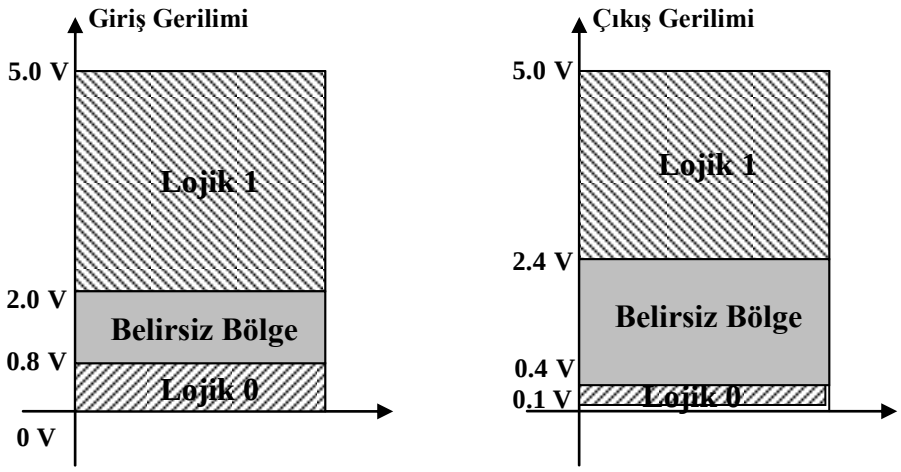
Bütün lojik kapı elemanları, filip-floplar, kodlayıocılar, kod çözücüler, seçiciler, dağıtıcılar gibi lojik elemanlar TTL ve CMOS teknolojisi ile üretilmektedir.

TTL tümdevre özellikleri

Bilindiği üzere lojik devreler 0 ve 1 ile ifade edilen ikili işaretler ile çalışırlar. Ancak her lojik ailenin lojik 0 ve lojik 1 seviyeleri birbirinden farklıdır.

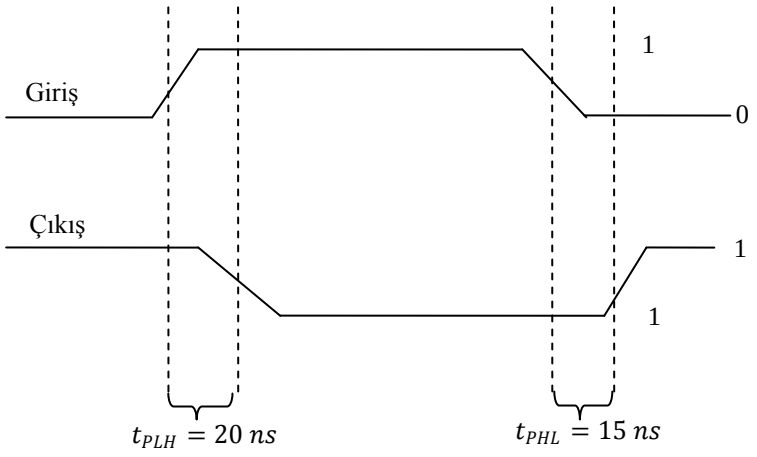


TTL tümdevrelerin besleme gerilimi $+5\ \mathrm{V}$ dur. Bu aileye ait bütün elemanların lojik giriş seviyeleri lojik çıkış seviyeleri ile aynıdır. Örneğin TTL teknolojisi ile üretilmiş 7404 entegresinde giriş gerilimi$0\ \mathrm{V} - 0.8\ \mathrm{V}$ arasındaysa Lojik 0, $2.0\ \mathrm{V} - 5.0\ \mathrm{V}$ arasında ise Lojik 1seviyesindedir.



Şekil 8.1. TTL tümdevrelerin giriş ve çıkış gerilimlerine ilişkin özellikler

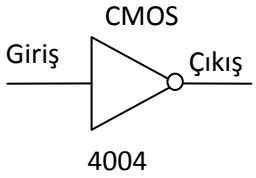
Bir elemanın girişinde oluşan bir işaret değişiminin çıkışta görülmesi için geçen süreye "***propagasyon gecikmesi***" denir. Bir NOT lojik kapısının girişine uygulanan $0-1-0$ geçişli bir işarete karşılık çıkışında oluşan $1-0-1$ şeklindeki işaret değişimi arasındaki gecikmeler değerlendirildiğinde şu sonuçla karşılaşılır. Lojik kapının 1-0 geçişine (15 ns' lik propagasyon gecikmesi) oranla 0-1 geçişinde (20 ns' lik propagasyon gecikmesi) daha büyük bir gecikme söz konusudur. TTL elemanların güç harcama miktarı 10 mW civarındadır. Fakat tümdevre içindeki kapıların kullanımları güç tüketimini etkiler.



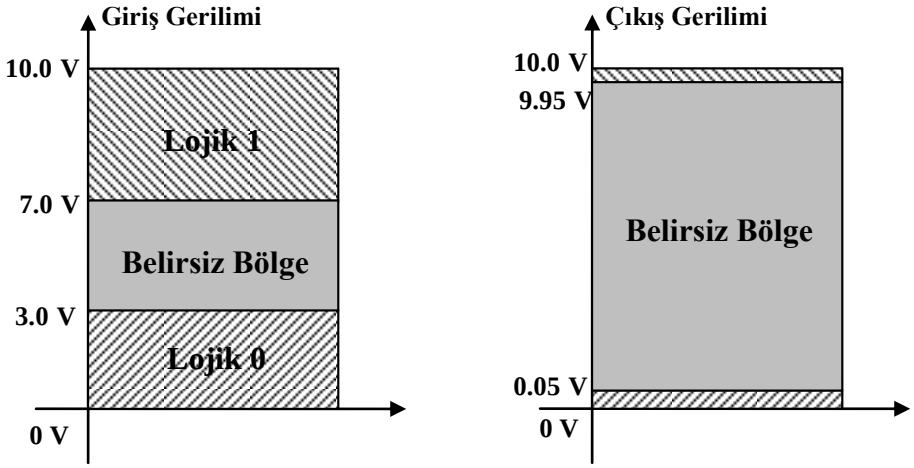
Şekil 8.2. TTL tümdevrelerin propagasyon gecikmesi

CMOS tümdevre özellikleri

CMOS tümdevrelerin besleme gerilimi $+10$ V dur.



CMOS teknolojisi ile üretilmiş entegreslerde giriş gerilimi $0\,\text{V} - 3.0\,\text{V}$ arasındaysa Lojik 0, $7.0\,\text{V} - 10.0\,\text{V}$ arasında ise Lojik 1seviyesindedir. Ayrıca gürültü filtreleme konusunda CMOS entegrelerde TTL' lerden daha iyidir.



Şekil 8.3. CMOS tümdevrelerin giriş ve çıkış gerilimlerine ilişkin özellikler

Standart CMOS tümdevrelerde propagasyon gecikmesi $25 - 100$ ns arasında değişir. Fakat, yeni nesil yüksek hızlı CMOS devrelerde (HC serisi) bu gecikme 8 ns mertebesine düşmektedir. CMOS elemanların güç harcama miktarı $0.01 - 1$ mW mertebesindedir.

9. KOMBİNASYONEL DEVRELER

Kombinasyonel devreler, çıkışı yalnızca o andaki giriş değerlerine bağlı olan ve bilinen kapıların bir araya getirilmesiyle oluşan lojik devrelerdir. Kombinasyonel devrelerde devrenin geçmiş bilgileri veya durumları tutulmaz; çıkışlar doğrudan o andaki girişlerin fonksiyonudur. Kombinasyonel devrelere "belleksiz" veya "statik devreler" de denir. Dolayısıyla yalın lojik ifadelerin gerçekleştirilmesinde kombinasyonel devreler kullanılır. Yarı toplayıcı, tam toplayıcı, yarı çıkarıcı, tam çıkarıcı, seçici (MUX), dağıtıcı (DEMUX), kod çözücü, kodlayıcı, komparatör ve ALU birer kombinasyonel tümleşik devredir.

Şekil 9.1. Kombinasyonel devrelerin gösterilişi

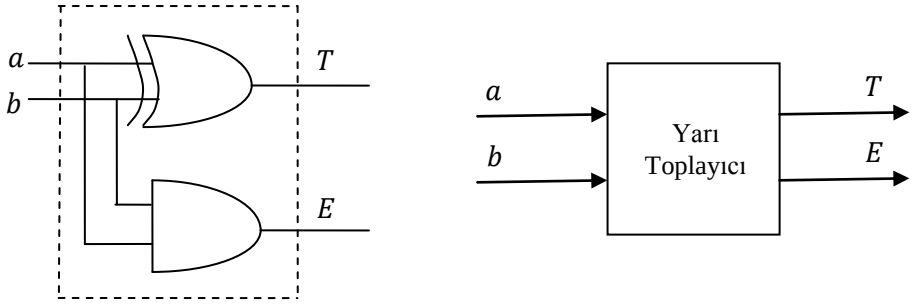
9.1. Toplayıcılar

9.1.1. Yarı toplayıcı (Half Adder):

Yarı toplayıcı elde girişi olmaksızın 1 bitlik iki sayının toplamını bulan bir kombinasyonel devredir.

a	***b***	**Toplam** ***T***	**Elde** ***E***
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$T = \bar{a}b + a\bar{b} \qquad E = ab$$



Şekil 9.2. Yarı toplayıcı devresi

9.1.2. Tam toplayıcı (Full Adder):

Girişinde elde bitinin olduğu ve bir bitlik iki sayı ile birlikte toplandığı bir kombinasyonel devredir.

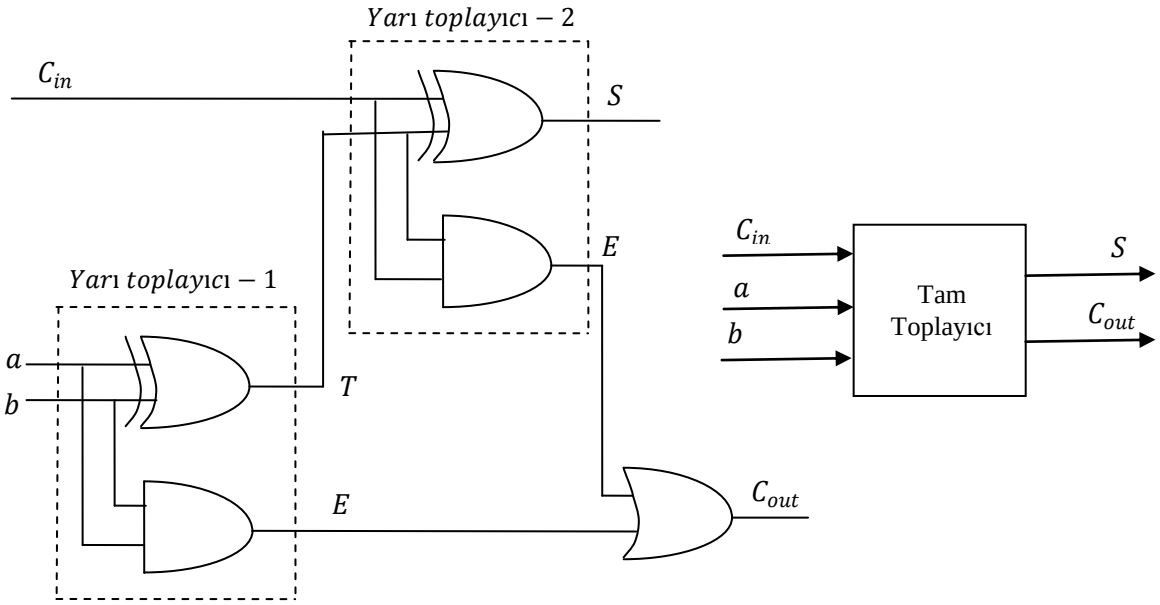
C_{in}	a	b	Toplam S	Elde C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

C_{in} \ ab	00	01	11	10
0	0	1	0	1
1	1	0	1	0

C_{in} \ ab	00	01	11	10
0	0	0	1	0
1	0	1	1	1

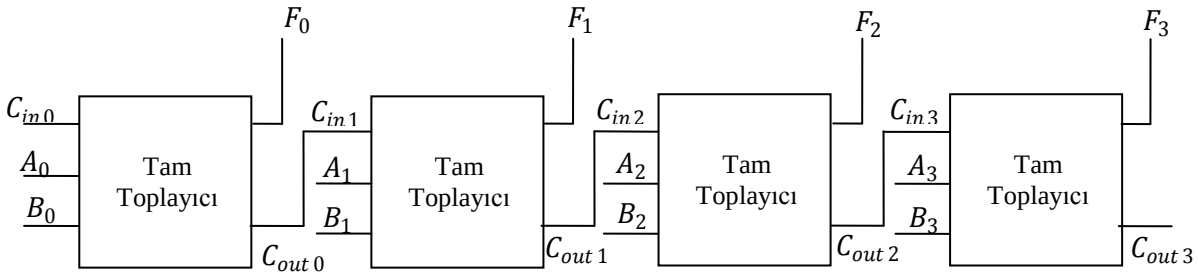
$$T = C_{in}\bar{a}\bar{b} + \overline{C_{in}}\bar{a}b + C_{in}ab + \overline{C_{in}}a\bar{b} = \overline{C_{in}}\left(\bar{a}b + a\bar{b}\right) + C_{in}\left(ab + \bar{a}\bar{b}\right) \Longrightarrow T = a \oplus b \oplus C_{in}$$

$$C_{out} = C_{in}b + C_{in}a + ab$$



Şekil 9.3. Tam toplayıcı devresi

Örnek 9.1. 4 bitlik iki sayının toplamını bulan paralel toplayıcı devresini full adder (FA) kullanarak gerçekleyiniz.

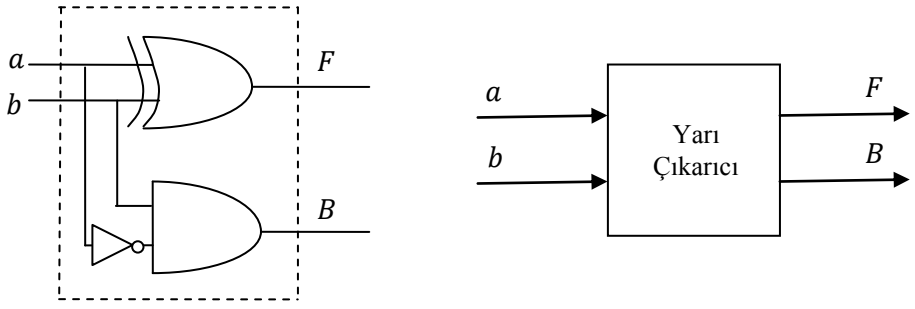


9.2. Çıkarıcılar

9.2.1. Yarı çıkarıcı (Half Subtactor)

Mikroişlemcilerde iki sayıyı birbirinden çıkarma işlemi ikinci sayının 2' ye tümleyenini alıp ilk sayı ile toplamak suretiyle gerçekleştirilir. Toplam sonucunda elde oluşursa sonuç pozitiftir, elde oluşmaz ise sonuç negatiftir ve yine tümleyeni alınarak önüne bir eksi işareti konulur. Lojik devrelerde bu işlem basit bir kombinasyonel devre ile gerçekleştirilir. Borç biti olmadan çıkarma işlemi yapan kombinasyonel devreye *yarı çıkarıcı* denir.

a	***b***	**Fark** ***F***	**Borç** ***B***
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0



Şekil 9.4. Yarı çıkarıcı devresi

9.2.2. Tam çıkarıcı (Full Subtactor)

Çıkarma işlemi sırasında bir önceki haneden bir borç biti geldiği düşünülürse bu durumda bir borç girişini yarı çıkarıcıya eklemek gerekir.

a	b	B_{in}	Fark F	Borç B_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

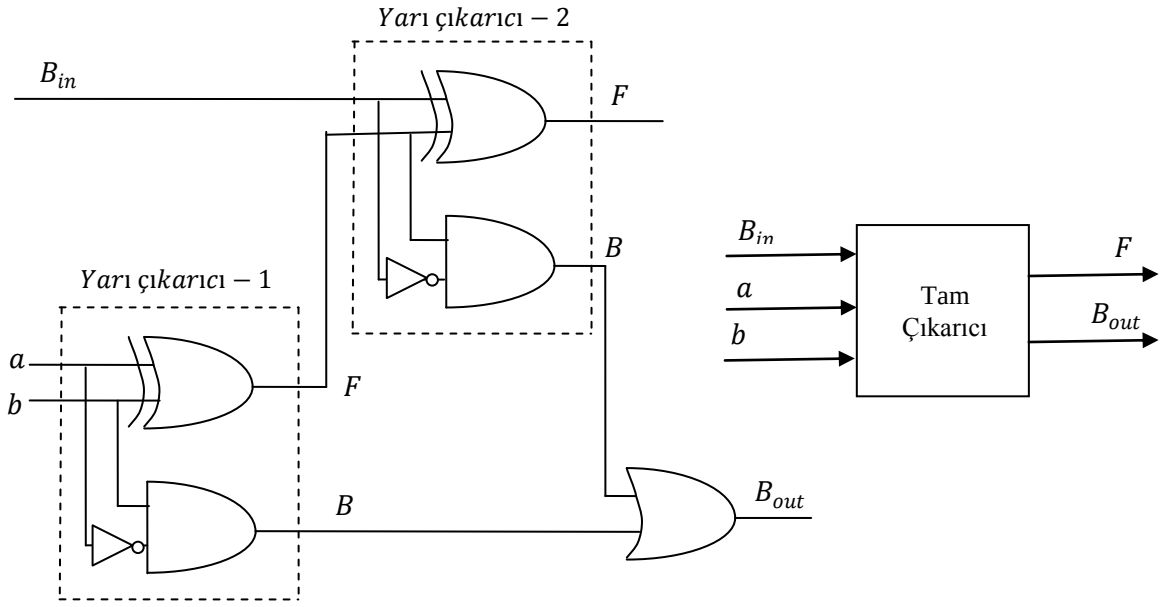
a \ bB_{in}	00	01	11	10
0	0	1	0	1
1	1	0	1	0

a \ bB_{in}	00	01	11	10
0	0	1	1	1
1	0	0	1	0

$$F = B_{in}\bar{a}\bar{b} + \overline{B_{in}}\bar{a}b + B_{in}ab + \overline{B_{in}}a\bar{b} = \overline{B_{in}}\left(\bar{a}b + a\bar{b}\right) + B_{in}\left(ab + \bar{a}\bar{b}\right)$$

$$\Longrightarrow T = a \oplus b \oplus B_{in}$$

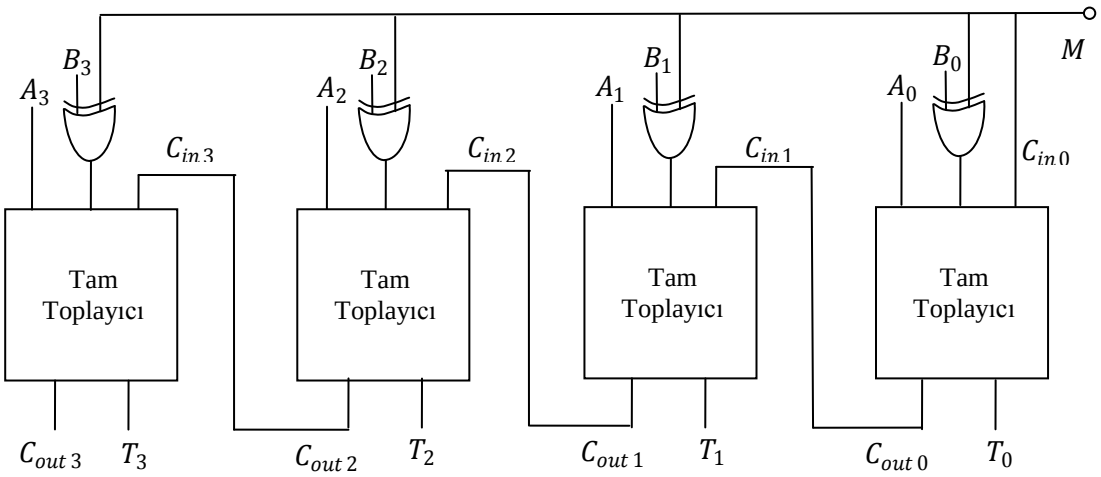
$$C_{out} = \bar{a}B_{in} + \bar{a}b + bB_{in}$$

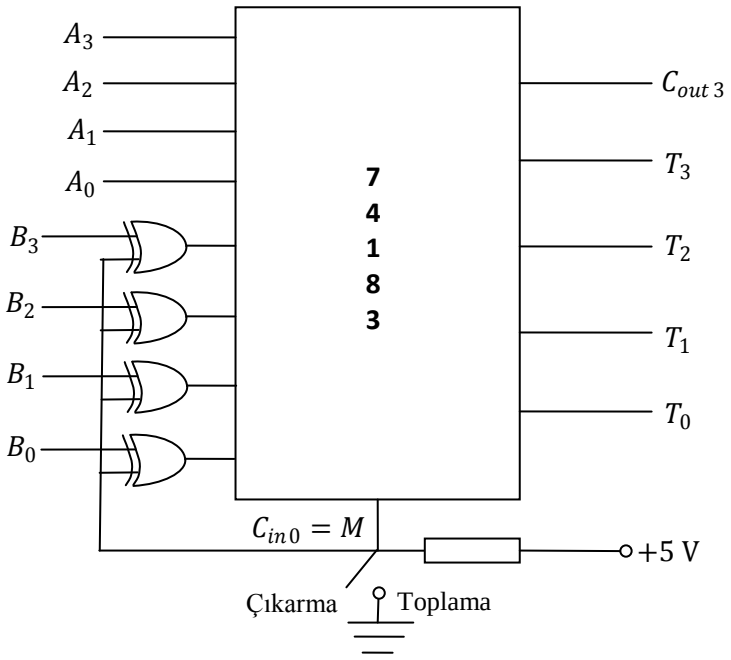


Şekil 9.5. Tam çıkarıcı devresi

Örnek 9.2. Tam toplayıcı (full adder kullanarak) kullanarak 4 bitlik toplama/çıkarma işlemlerini yapan devreyi gerçekleyiniz.

Örnekte tasarlanması istenilen devre iki işlemi gerçekleştirecektir. Bu sebepten söz konusu devrede bir seçme (kontrol) girişine ihtiyaç vardır. Tasarımda çıkarma işleminin gerçekleştirilmesinde tümleyen aritmetiği kullanılmaktadır. Eğer toplama işlemi yapılacaksa işlem doğrudan gerçeklenir, ancak çıkarma işlemi yapılacaksa çıkarılacak sayı ikiye tümlenir ve ilk sayı ile toplanır. Her iki durumda da bir toplama işleminin gerçeklenmesi gerektiğinden toplama ve çıkarma işlemini yapacak devre tasarımında tam toplayıcılar kullanılır. $M = 0$ *iken toplama*$(A + B)$, $M = 1$ *iken çıkarma işlemi*$(A + \bar{B} + C_{in0})$ *gerçeklenir (XOR kapısında girişin biri 0 iken diğer girişin aynısı alınır, girişin biri 1 iken diğer girişin tümleyeni alınır).*





9.3. Kod Çeviriciler

Dijital sistemlerde birçok kod sistemi kullanılmaktadır. Bir sistemin çıkışı çoğu zaman diğer bir sisteme giriş olarak uygulanmaktadır. Eğer bu iki sistem aynı bilgiler için farklı kodları kullanıyorsa bu iki sistem arasına kod çevirici sistemler yerleştirilmelidir. Şekil 9.6' da gösterildiği gibi Sistem A' nın çıkışları Sistem B' nin girişlerini oluşturmaktadır. İki sistemin art arda bağlanması için bir kod çeviriciye ihtiyaç vardır.

Şekil 9.6. Kod çevirici örneği

Örnek 9.3. BCD' yi 3-Fazlalık koduna dönüştüren devreyi (kod çözücü) tasarlayınız.

A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

$$W_{A,B,C,D} = \sum(5,6,7,8,9) + \Phi(10,11,12,13,14,15)$$

$$X_{A,B,C,D} = \sum(1,2,3,4,9) + \Phi(10,11,12,13,14,15)$$

$$Y_{A,B,C,D} = \sum(0,3,4,7,8) + \Phi(10,11,12,13,14,15)$$

$$Z_{A,B,C,D} = \sum(0,2,4,6,8) + \Phi(10,11,12,13,14,15)$$

CD / AB	00	01	11	10
00				
01		1	1	1
11	X	X	X	X
10	1	1	X	X

$$W = A + BC + BD$$

$$X = \bar{B}C + \bar{B}D + B\bar{C}\bar{D}$$

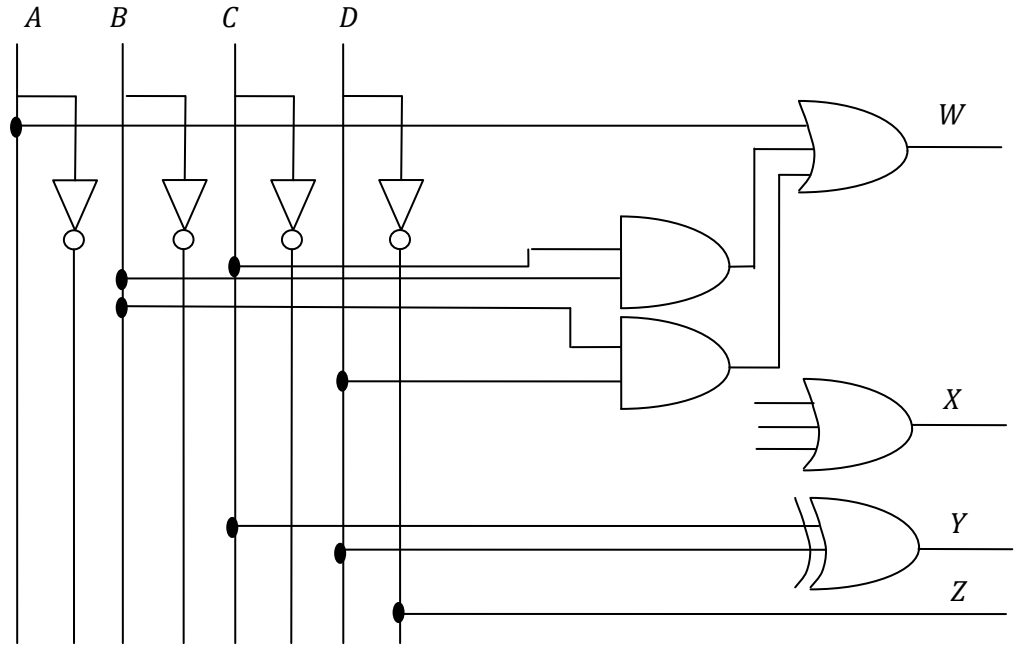
CD / AB	00	01	11	10
00		1	1	1
01	1			
11	X	X	X	X
10		1	X	X

CD / AB	00	01	11	10
00	1		1	
01	1		1	
11	X	X	X	X
10	1		X	X

$$Y = CD + \bar{C}\bar{D} = C \odot D$$

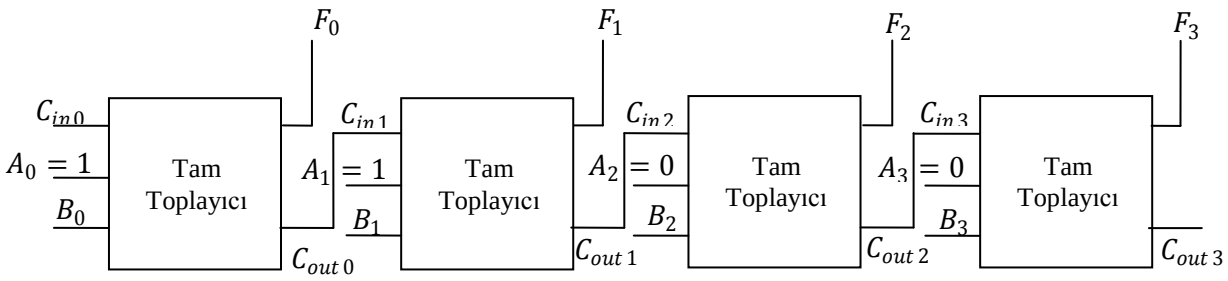
CD / AB	00	01	11	10
00	1			1
01	1			1
11	X	X	X	X
10	1		X	X

$$Z = \bar{D}$$



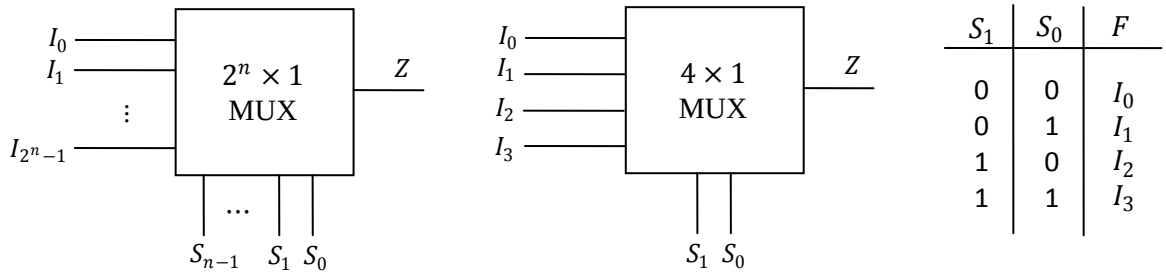
Örnek 9.4. 4 bitlik tam toplayıcı kullanarak BCD sayı sistemini 3-fazlalık koduna çeviren kod çeviriciyi tasarlayınız.

$A_3A_2A_1A_0 = 0011$ olarak kabul edilirse $B_3B_2B_1B_0$ BCD sayısının 3-fazlalık kodundaki değeri $F_3F_2F_1F_0$ şeklinde olacaktır.



9.4. Multiplexer (MUX-Seçici)

Sayısal veri iletiminde pek çok olası giriş yolundan birini seçerek bu yolu çıkışa veren özel bir kombinasyonel devredir. Bazı uygulamalarda *veri seçici* ya da *veri toplayıcı* adı da verilir.



S_1	S_0	F
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

Şekil 9.7. Multiplexer

Örnek 9.5. Doğruluk tablosu aşağıda verilen kombinasyonel devreyi x_3 ve x_4 seçme girişleri olacak şekilde 4×1 MUX kullanarak gerçekleştiriniz.

x_1	x_2	x_3	x_4	$f(x_1, x_2, x_3, x_4)$
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

$$f(x_1, x_2, x_3, x_4) = \overline{x_1}.\overline{x_2}.x_3.\overline{x_4} + \overline{x_1}.\overline{x_2}.x_3.x_4 + \overline{x_1}.x_2.\overline{x_3}.x_4 + \overline{x_1}.x_2.x_3.x_4$$
$$+x_1.\overline{x_2}.\overline{x_3}.\overline{x_4} + x_1.\overline{x_2}.\overline{x_3}.x_4 + x_1.\overline{x_2}.x_3.\overline{x_4} + x_1.\overline{x_2}.x_3.x_4$$

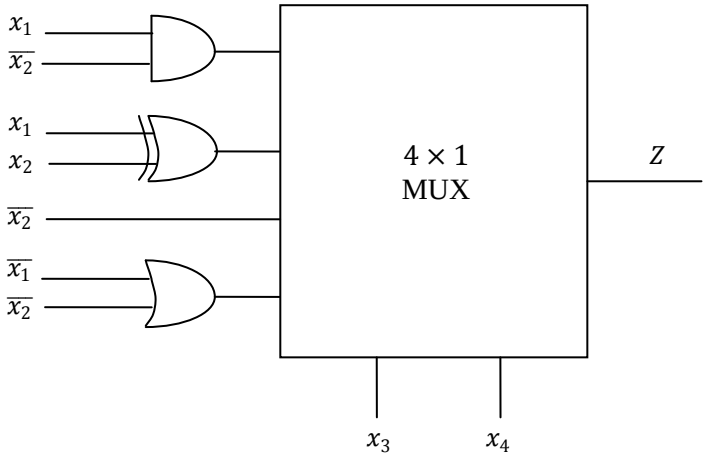
Fonksiyonu seçme girişleri cinsinden paranteze almak gerekir. Fonksiyon incelenecek olursa seçme girişlerinin aldığı değerler $x_3.\overline{x_4}$, $x_3.x_4$, $\overline{x_3}.x_4$, $\overline{x_3}.\overline{x_4}$ şeklindedir. Ortak paranteze alınırsa fonksiyon şu şekilde olur:

$$f(x_1, x_2, x_3, x_4) = x_3.x_4(\overline{x_1}.x_2 + \overline{x_1}.\overline{x_2} + x_1.\overline{x_2}) + x_3.\overline{x_4}\,.(\overline{x_1}.\overline{x_2} + x_1.\overline{x_2})$$
$$+\overline{x_3}.x_4(\overline{x_1}.x_2 + x_1.\overline{x_2}) + \overline{x_3}.\overline{x_4}(x_1.\overline{x_2})$$

Ortak paranteze alınan ifadelerin sadeleştirilmesi ile

$$f(x_1,x_2,x_3,x_4) = x_3.x_4(\overline{x_1}.(x_2+\overline{x_2}) + x_1.\overline{x_2}) + x_3.\overline{x_4}\,.(\overline{x_2}(\overline{x_1}+x_1))$$
$$+\overline{x_3}.x_4(\overline{x_1}.x_2 + x_1.\overline{x_2}) + \overline{x_3}.\overline{x_4}(x_1.\overline{x_2})$$
$$f(x_1,x_2,x_3,x_4) = x_3.x_4(\overline{x_1}+\overline{x_2}) + x_3.\overline{x_4}\,.(\overline{x_2})$$
$$+\overline{x_3}.x_4(\overline{x_1}.x_2 + x_1.\overline{x_2}) + \overline{x_3}.\overline{x_4}(x_1.\overline{x_2})$$

x_3	x_4	I
0	0	$x_1.\overline{x_2}$
0	1	$x_1 \oplus x_2$
1	0	$\overline{x_2}$
1	1	$\overline{x_1}+\overline{x_2}$



NOT: $\overline{x_1}.(x_2+\overline{x_2}) + x_1.\overline{x_2} = \overline{x_1}.1 + x_1.\overline{x_2} = \overline{x_1}.1 + x_1.\overline{x_2} = \overline{x_1} + x_1.\overline{x_2} \Longrightarrow$

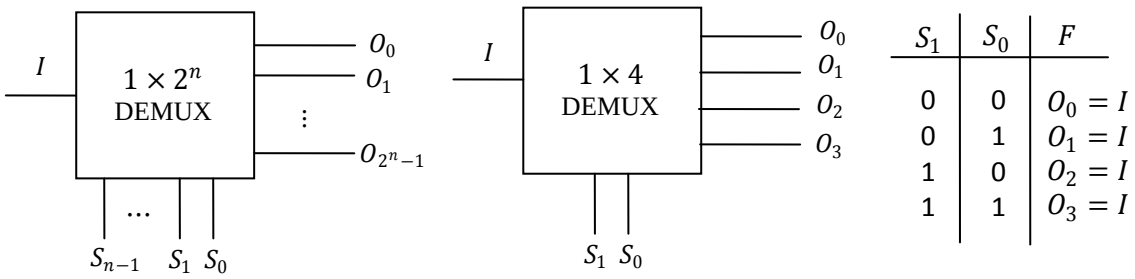
Boole cebri 9.kural Etkisizlik Özelliği$(1.a = a)$

$\overline{x_1} + x_1.\overline{x_2} = \overline{x_1} + \overline{x_2} \Longrightarrow (b = \overline{x_1}, \quad a = \overline{x_2})$

Boole cebri 11.kural Etkisizlik Özelliği$\left(a \cdot \bar{b} + b\right) = a + b$

9.5. Demultiplexer (DEMUX-Çoğullayıcı)

Sayısal veri iletiminde bir noktadan gelen giriş işaretini bir çok olası çıkış yolundan birini seçip, bu yola veren özel bir kombinasyonel devredir. MUX ile DEMUX' un birlikte kullanıldığı en bilinen sistem ilkel telefon santralleridir.



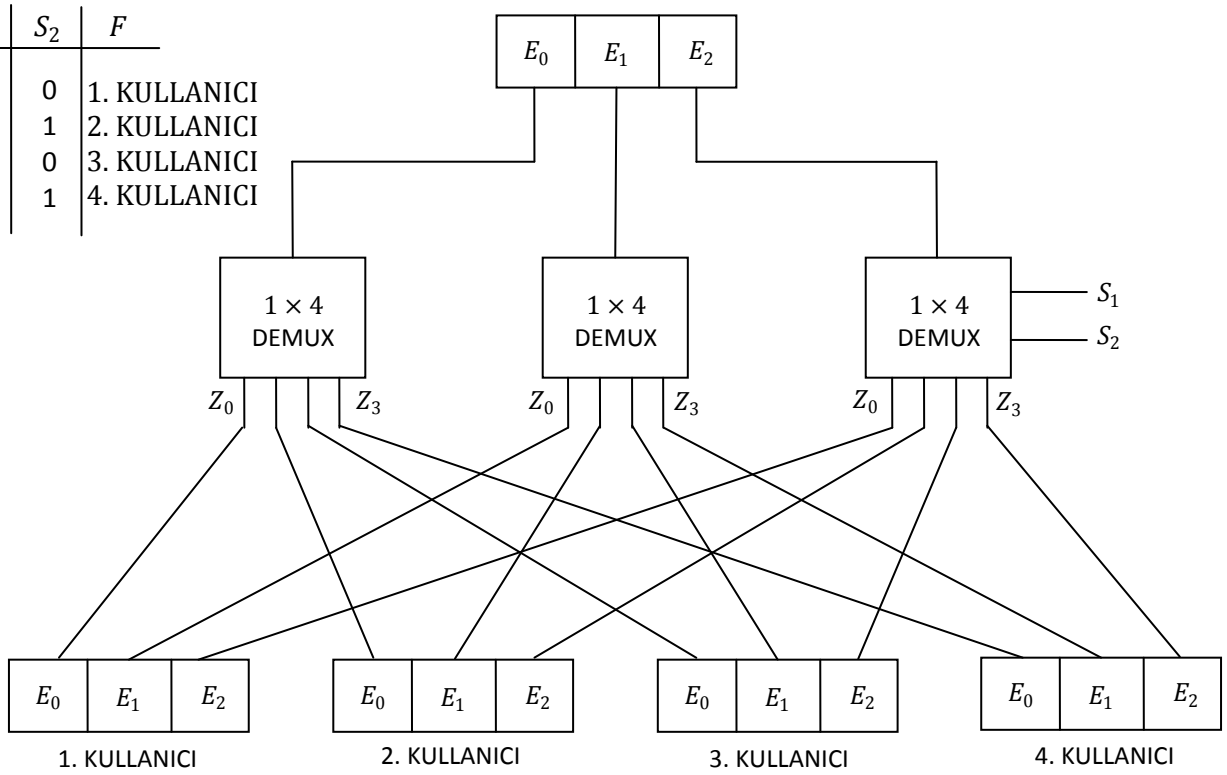
S_1	S_0	F
0	0	$O_0 = I$
0	1	$O_1 = I$
1	0	$O_2 = I$
1	1	$O_3 = I$

Şekil 9.8. Demultiplexer

Örnek 9.6. 3 bitlik bir ifade dört kullanıcıya aktarılacaktır. Bu işlem her kullanıcının seçilmesi ve bilginin sadece bu kullanıcılara ulaştırılması şeklinde olacaktır. Bu devreyi DEMUX kullanarak gerçekleştiriniz.

Aktarılacak ifade 3 bitlik ve kullanıcı sayısı 4 olduğuna göre her bir bit için bir tane DEMUX kullanılacak ve kullanıcı sayısı kadar çıkışı olması istenecektir. Dolayısıyla 3 adet 1×4' lük DEMUX kullanılacaktır.

S_1	S_2	F
0	0	1. KULLANICI
0	1	2. KULLANICI
1	0	3. KULLANICI
1	1	4. KULLANICI

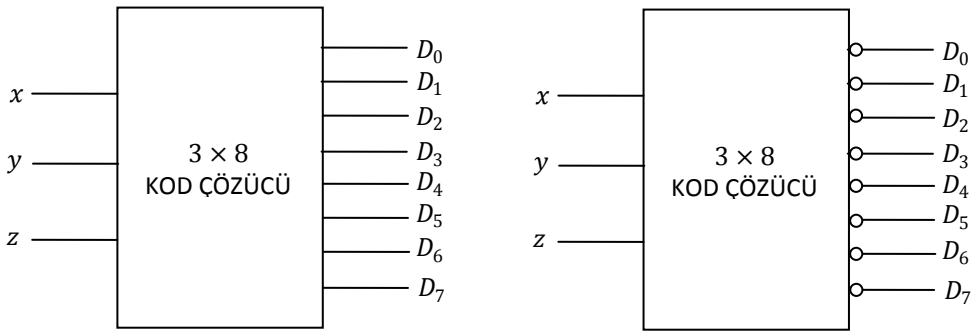


ÖDEV: 3 kullanıcıdan gelen 3 bitlik veri tek merkezden alınabilecektir. İlgili devreyi MUX kullanarak gerçekleştiriniz.

9.6. Kod Çözücü (Decoder)

Kod çözücü n bitlik bir sözcüğün kodunu çözüp olası en çok 2^n çıkış yolundan sadece birini aktif hale getiren bir kombinasyonel devredir. 3 girişli, $2^3 = 8$ çıkışlı bir kod çözücüye ait doğruluk tablosu ve devre aşağıdaki gibidir.

GİRİŞLER			ÇIKIŞLAR							
x	y	z	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1



(a) *(b)*

Şekil 9.9. 3×8 *Kod çözücü a) Normal çıkışlı b) Tümleyen çıkışlı*

NOT: Kod çözücüler pratikte kullanılırken bazı noktalara dikkat edilmesi gerekir. Kod çözücülerde veri girişlerinin yanı sıra bir de entegre devrenin çalışıp çalışmamasını sağlayan enable (etkinleştirme) girişleri de bulunmaktadır. 2×4 kod çözücülerde 1 tane tümleyen girişli, 3×8 kod çözücülerde 2' si tümleyen girişli 1' i normal girişli olmak üzere 3 tane enable girişi yer almaktadır. Örneğin 74138 entegresinde $\overline{E_1}, \overline{E_2}$ ve E_3 olamk üzere 3 tane enable girişi vardır, bu girişler sırasıyla 001 değerini aldığı zaman entegre çıkış üretir.

Örnek 9.7. Bir başkan, bir başkan yardımcısı ve iki üyeden oluşan bir komisyonun alacağı kararları gösteren bir oylama sistemi yapılacaktır. Başkanın oyu 3, başkan yardımcısının oyu 2, üyelerin oyu ise 1 ağırlığındadır. EVET oylarının ağırlık katsayıları fazla ise "EVET", HAYIR oylarının ağırlık katsayıları yüksek ise "HAYIR", eşitlik halinde başkan yardımcısı ve üyelerin verdiği çoğunluk oylarına göre karar alınacaktır. Bütün üyeler oy kullanmak zorundadır. EVET oyu Lojik1' dir. Bu oylama sistemine ait lojik fonksiyonun doğruluk tablosunu çıkararak, fonksiyonu Karnaugh diyagramı ile sadeleştirerek tasarlanan devreyi çiziniz.

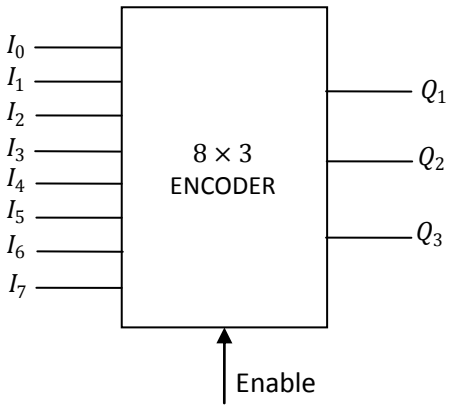
B	***BY***	Ü**1**	Ü**2**	***F***
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Ü1Ü2 / BBY	00	01	11	10
00				
01			1	
11	1	1	1	1
10		1	1	1

$$F = BBY + B\text{Ü}1 + B\text{Ü}2 + BY\text{Ü}1\text{Ü}2$$

9.7. Kodlayıcı (Encoder)

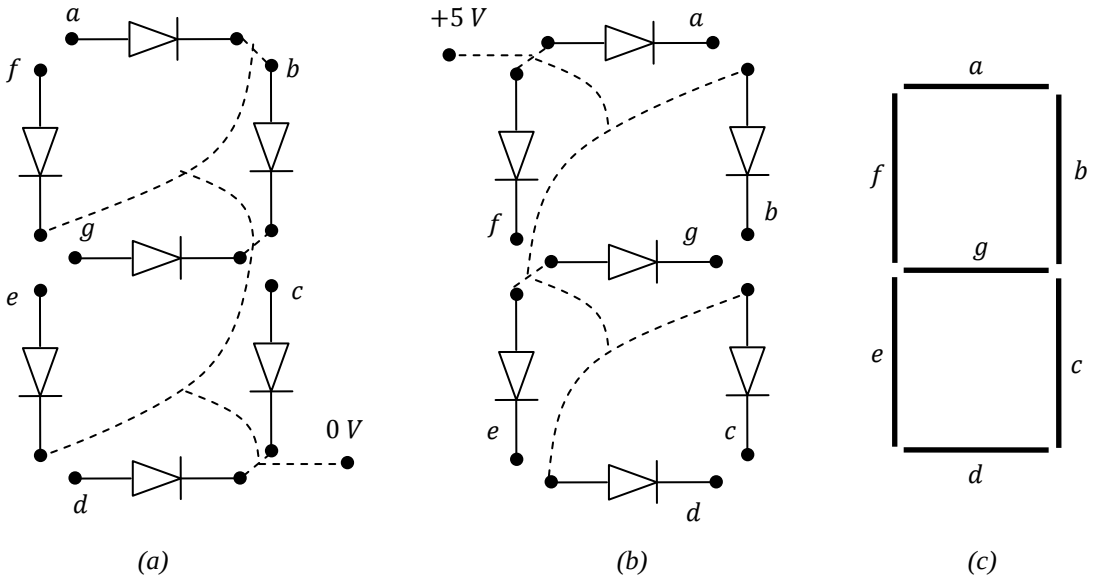
Decoderin tersi işlem yapan bir kombinasyonel devredir. Kodlayıcı bir lojik işareti başka bir lojik devre tarafından işlenebilecek şekle getiren devredir.



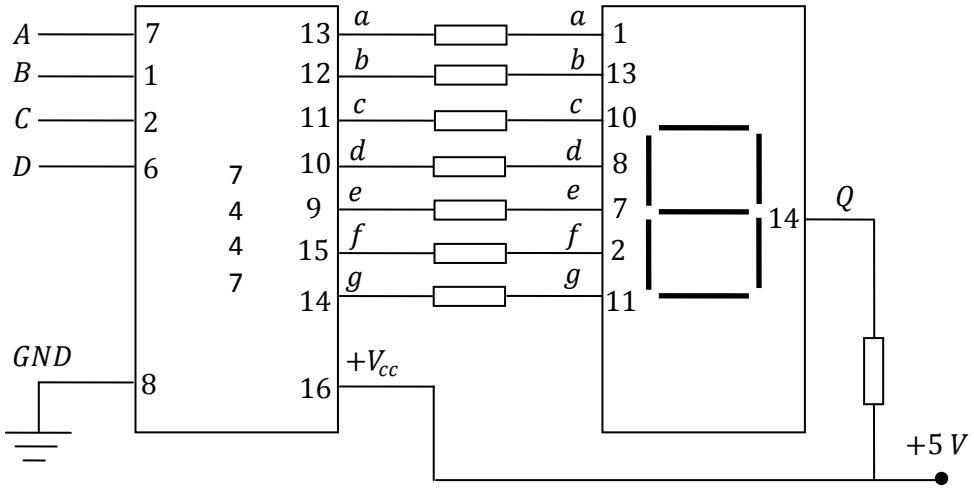
Şekil 9.10. 8 × 3 *Encoder*

9.8. Display (7 parçalı gösterge-7 segment display)

BCD kodunda verilmiş olan bir ifadenin, sayı biçiminde gösterilebilmesi amacıyla kullanılan ışıklı bir kombinasyonel devredir. Yedi parçalı göstergenin her parçası a' dan g' ye kadar kodlanmıştır.

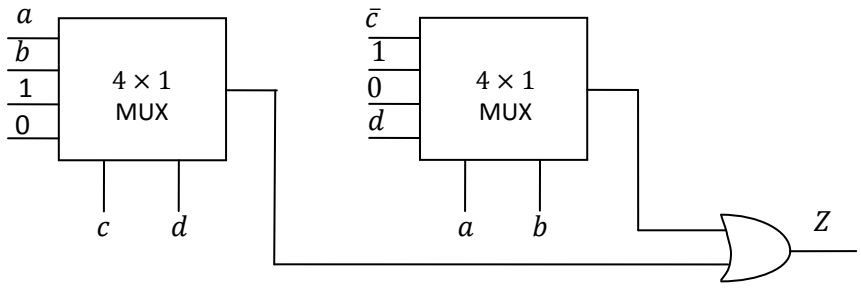


Şekil 9.11.(a) Ortak katotlu display, (b) Ortak anotlu display, (c) 7-Parçalı Gösterge



Şekil 9.12. 7-segment displayin sürülmesi (7447BCD/display kod çözücü kullanımı)

Örnek 9.8. *(2006-vize)* İki tane MUX ve bir VEYA kapısı içeren devreye ilişkin şema aşağıda verilmiştir. Devreye ait Z lojik fonksiyonunu doğruluk tablosu ile oluşturarak Karnough diyagramı yardımıyla elde ediniz.



a	**b**	**c**	**d**	**F_1**	**F_2**	**Z**
0	0	0	0	0	1	1
0	0	0	1	0	1	1
0	0	1	0	1	0	1
0	0	1	1	0	0	0
0	1	0	0	0	1	1
0	1	0	1	1	1	1
0	1	1	0	1	1	1
0	1	1	1	0	1	1
1	0	0	0	1	0	1
1	0	0	1	0	0	0
1	0	1	0	1	0	1
1	0	1	1	0	0	0
1	1	0	0	1	0	1
1	1	0	1	1	1	1
1	1	1	0	1	0	1
1	1	1	1	0	1	1

c	d	F_1
0	0	a
0	1	b
1	0	1
1	1	0

a	b	F_2
0	0	$\bar{c}$
0	1	1
1	0	-

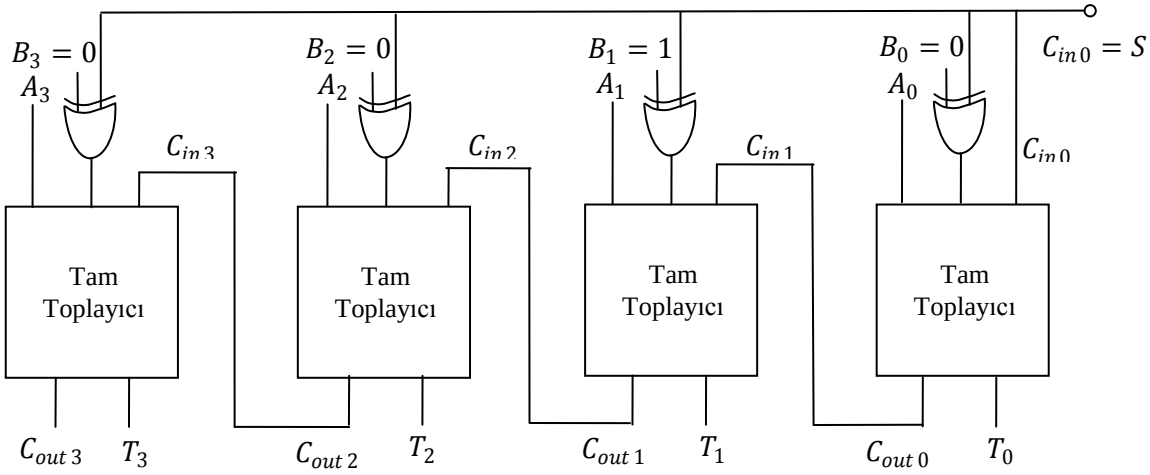
ab \ cd	00	01	11	10
00	1	1	0	1
01	1	1	1	1
11	1	1	1	1
10	1	0	0	1

$$Z = b + \bar{a}\bar{c} + \bar{d}$$

Örnek 9.9. *(2006-vize)* 4 bitlik $A_3A_2A_1A_0$ sayısı, S seçme ucuna bağlı olarak 2 artırılıp/çıkarılacaktır. $S = 0$ olduğunda sayının değeri 2 artarken, $S = 1$ olduğunda sayının değeri 2 eksilecektir. Örneğin A sayısı 9 ise $S = 0$ olduğunda $F = 11$, $S = 1$ olduğunda $F = 7$ olarak çıkışta görülecektir. Gerekli devreyi birer bitlik tam toplayıcılar kullanarak blok olarak tasarlayınız.

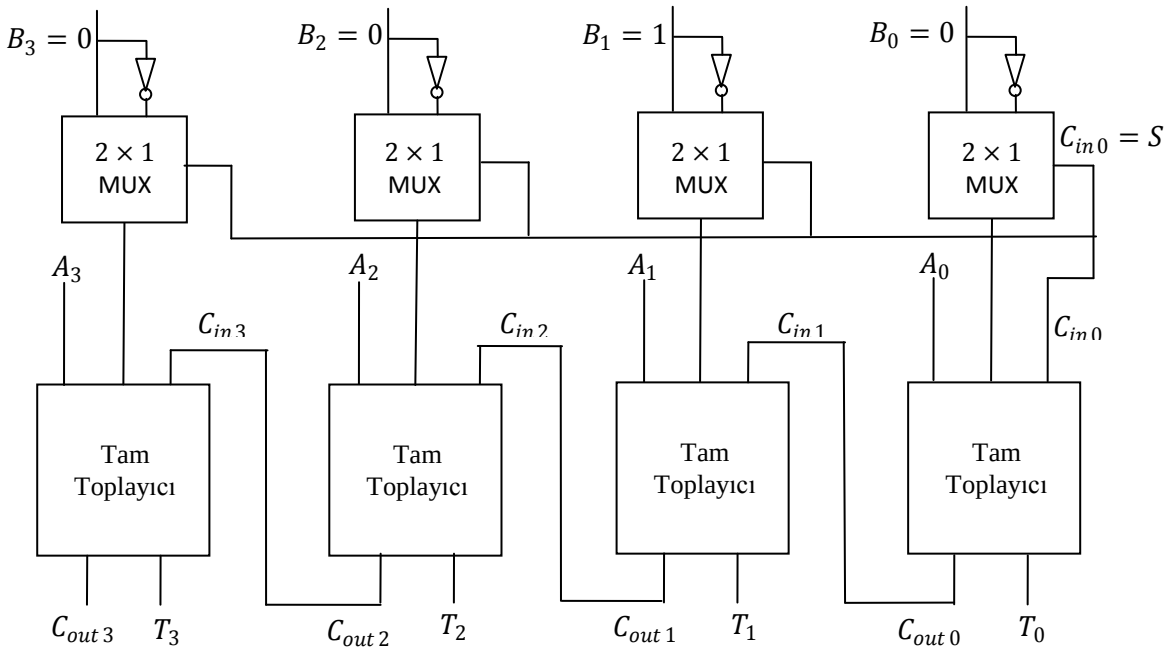
I.YOL *1. giriş 0 olduğunda XOR kapısının üreteceği değer diğer girişin aynısıdır. XOR kapısının 1. Girişi S seçme ucudur. S=0 olduğunda XOR çıkışları yani tam toplayıcıların girişleri ikinci verinin aynısını verecektir ($B_3B_2B_1B_0$). Bu durumda tam toplayıcı girişleri A ve B verileri olacaktır. A=0010 olduğuna göre tam toplayıcı çıkışından B sayısının 2 fazlası alınmış olur.*

1. giriş 1 olduğunda XOR kapısının üreteceği değer diğer girişin tümleyenidir. XOR kapısının 1. Girişi S seçme ucudur. S=1 olduğunda XOR çıkışları yani tam toplayıcıların girişleri ikinci verinin tümleyenini verecektir ($\overline{B_3B_2B_1B_0}$). Bu durumda tam toplayıcı girişleri A ve $\overline{B}$ verileri olacaktır. $A + \overline{B} + C_{in0}$ yani tümleyenle çıkarma ($A - B$) yapılmış olacak ve tam toplayıcı çıkışında A sayısının 2 eksiği bulunacaktır.



II.YOL *Çözüm 2x1' lik MUX' lar kullanılarak da gerçekleştirilebilir. Tasarımdan görüleceği gibi seçme ucu MUX' ların seçme uçlarına bağlanmıştır. S=0 olduğundan MUX' un 1. Girişindeki veri çıkışına aktarılacaktır. Böylece tam toplayıcıları girişlerinde 0010 ve A verileri olacaktır ve toplama işlemi gerçekleştirilmiş olur.*

Aynı şekilde S=1 olduğunda MUX' un 2 girişi çıkışa aktarılır. Bu durumda çıkışlarda $\overline{B}$görülecektir. Ve 2' ye tümleyenle çıkarma esasına göre çıkarma $A - B$ gerçekleştirilmiş olur.



Örnek 9.10. Aşağıdaki şekilde bir dört yol ağzı ile ortadaki trafik lambası görülmektedir. Yolun altına yerleştirilen algılayıcılar her sokaktan dört yol ağzına yanaşan arabaları hissetmekte ve trafik lambasına iletmektedir. Trafik lambasında kırmızı ve yeşil ışıklar bulunmakta doğu-batı ışığı kırmızı ise kuzey-güney ışığı yeşil ya da tersi olmaktadır. Trafik lambası aşağıdaki kurallara göre çalışmaktadır:

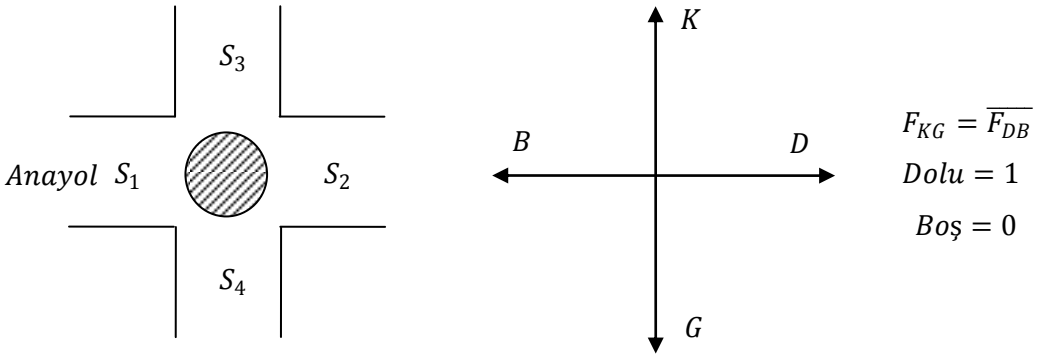
1. Hem S_1 hem S_2 yolu doluysa S_3 ve S_4 ne durumda olursa olsun doğu-batı ışığı yeşil yanacaktır. $(S_1S_2 = 11 \implies F_{DB} = 1)$

2. S_1 veya S_2 den herhangi biri doluysa ve S_3 ve S_4' den yalnızca biri dolu ise doğu-batı ışığı yeşil yanacaktır. $(S_1S_2 = 10\ yada\ S_1S_2 = 01\ iken\ S_3S_4 = 10\ yada\ S_3S_4 = 01 \implies F_{DB} = 1)$

3. Eğer dört yolda başsa doğu-batı ışığı yeşil yanacaktır.$(S_1S_2S_3S_4 = 0000 \implies F_{DB} = 1)$

4. Yukarıdaki kurallar dışında kuzey-güney ışığı yeşil yanacaktır. $(F_{KG} = \overline{F_{DB}})$

Bu durumda doğu-batı ve kuzey-güney ışıkları için birer durum tablosu çıkararak ilgili fonksiyonları gerçekleştiriniz (Devre şeması çizilecek)



S_1	S_2	S_3	S_4	F_{DB}	F_{KG}
0	0	0	0	1	0
0	0	0	1	0	1
0	0	1	0	0	1
0	0	1	1	0	1
0	1	0	0	0	1
0	1	0	1	1	0
0	1	1	0	1	0
0	1	1	1	0	1
1	0	0	0	0	1
1	0	0	1	1	0
1	0	1	0	1	0
1	0	1	1	0	1
1	1	0	0	1	0
1	1	0	1	1	0
1	1	1	0	1	0
1	1	1	1	1	0

S_1S_2 \ S_3S_4	00	01	11	10
00	1	0	0	0
01	0	1	0	1
11	1	1	1	1
10	0	1	0	1

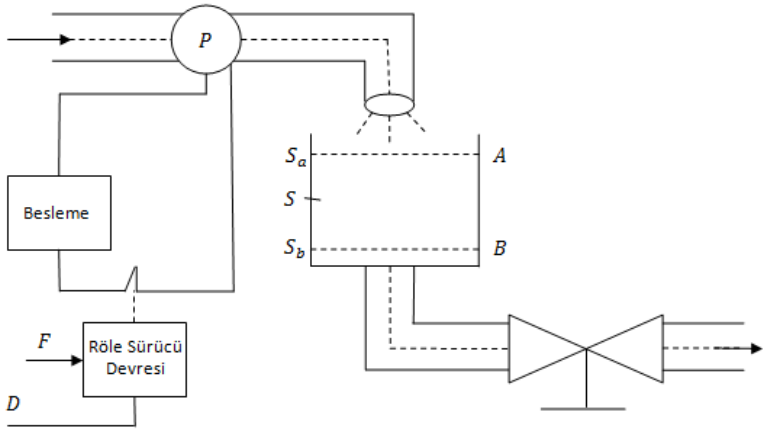
$$F_{DB} = \overline{S_1}\overline{S_2}\overline{S_3}\overline{S_4} + S_2\overline{S_3}S_4 + S_2S_3\overline{S_4} + S_1\overline{S_3}S_4 + S_1S_3\overline{S_4} + S_1S_2$$

Örnek 9.11. Aşağıdaki şekilde gösterildiği gibi bir su deposuna P pompası ile yukarıdan su basılmakta ve bu deponun aşağısına su dağıtılmaktadır. Depodaki suyun seviyesi S, maksimum seviye S_a' dan küçük, minimum seviye S_b ' den büyük olmalıdır. Buna göre S_a ve S_b seviyelerini kontrol etmek için uygun olarak A ve B sıvı seviye sensörleri (algılayıcıları) kullanılmaktadır. Su pompasının aşağıdaki kurallar dâhilinde çalışması istenmektedir.

1. Suyun seviyesi maksimuma ulaşmış ve pompa çalışıyor ise kontrol ünitesi tarafından pompa durdurulacak, eğer pompa duruyor ise mevcut durumu muhafaza ettirilecektir. $(AB = 11\ ve\ D = 1 \Longrightarrow F = 0, AB = 11\ ve\ D = 0 \Longrightarrow\ F = 0)$

2. Suyun seviyesi minimuma düşmüş ve pompa duruyor ise kontrol ünitesi tarafından pompa çalıştırılacak eğer pompa çalışıyor ise çalışmasına devam ettirilecektir.$(AB = 00\ ve\ D = 1 \Longrightarrow F = 1, AB = 00\ ve\ D = 0 \Longrightarrow\ F = 1)$

3. Suyun seviyesi maksimum ve minimum seviyeleri arasında ise kontrol ünitesi tarafından motorların mevcut durumları muhafaza ettirilecektir. $(AB = 01\ ve\ D = 0 \Longrightarrow F = 0, AB = 01\ ve\ D = 1 \Longrightarrow\ F = 1)$



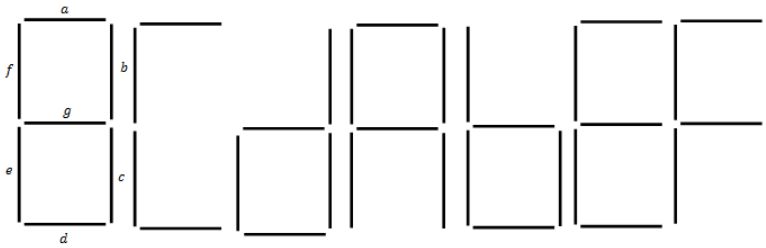
$A: Maksimum\ seviye\ sensörü, B: Minimum\ seviye\ sensörü, D: Pompa,$
$F: Kontrol\ ünitesi\ için\ üretilecek\ fonksiyon$

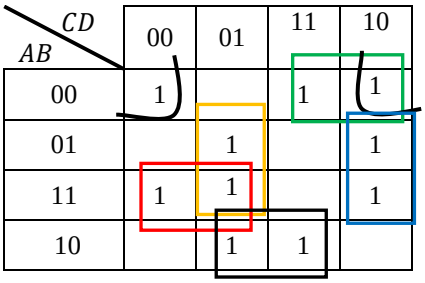
A	***B***	***D***	***F***
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	Mümkün değil
1	0	1	Mümkün değil
1	1	0	0
1	1	1	0

A \ BD	00	01	11	10
0	1	1	1	0
1			0	0

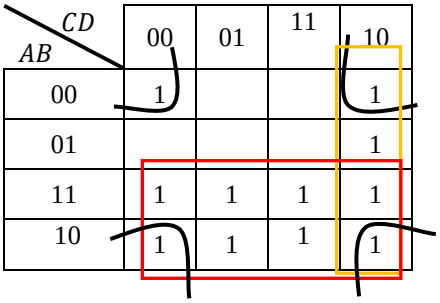
$$F = \bar{A}\bar{B} + \bar{A}D$$

Örnek 9.12. 7 parçalı ortak katotlu display için 0' dan 9' a kadar sayıların yanı sıra aşağıdaki karakterler de gösterilecektir. Gerekli devrenin sadece d ve e çıkışlarını gerçekleyiniz.





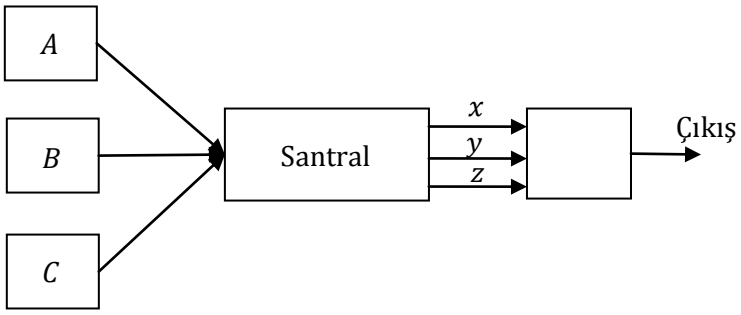
A	B	C	D	d	e
0	0	0	0	1	1
0	0	0	1	0	0
0	0	1	0	1	1
0	0	1	1	1	0
0	1	0	0	0	0
0	1	0	1	1	0
0	1	1	0	1	1
0	1	1	1	0	0
1	0	0	0	1	1
1	0	0	1	0	0
1	0	1	0	0	1
1	0	1	1	1	1
1	1	0	0	1	1
1	1	0	1	1	1
1	1	1	0	1	1
1	1	1	1	0	1



$$d = \bar{A}\bar{B}\bar{D} + \bar{A}\bar{B}C + B\bar{C}D + AB\bar{C} + BC\bar{D} + A\bar{B}D$$

$$e = \bar{B}\bar{D} + A + C\bar{D}$$

Örnek 9.13. Şekildeki telefon sisteminde konuşmada öncelik sırası A, B ve C' dir. Santral bu önceliği seçerek çıkış verecektir. Bu sistemi gerçekleştiriniz (Konuşma isteğinde santral 1 sinyali verecektir).



A	B	C	x	y	z
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	1	0	0
1	0	1	1	0	0
1	1	0	1	0	0
1	1	1	1	0	0

$$f(x) = A,$$

$$f(y) = \bar{A}B\bar{C} + \bar{A}BC = \bar{A}B(\bar{C} + C)$$

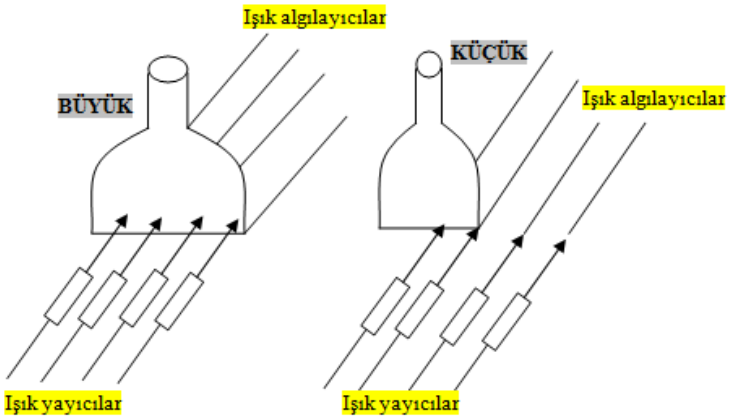
$$f(y) = \bar{A}B,$$

$$f(z) = \bar{A}\bar{B}C$$

Örnek 9.14. Hareketli bir bantta birbiri ardına dizilmiş kavanozların çapını kontrol etmek için yan yana eşit aralıklarla dört ışık demeti yerleştirilmiştir. Kavanozlar bant üzerinde ilerlerken ışık demetlerinin yolunu keserek ışık demetinin üzerine düştüğü fotoselin sinyal vermesini sağlar. Fotosel çıkışı üzerinde ışık yokken 1 olmaktadır. İstenilen çaptaki kavanoz 3 ışık demetini kesmektedir. 4 ışık demetini kesen kavanozun çapı büyük, 2 ya da daha az ışık demetini kesen kavanozun çapı küçüktür. Fotosel çıkışları a, b, c, d ve iyi kavanozlar için fonksiyon değeri 1' dir.

a) İyi/ Iskarta ayrımı yapan lojik fonksiyonu gerçekleştiriniz.

b) Küçük/İyi/Büyük fonksiyonları veren lojik ifadeleri yazınız.



a	b	c	d	$F_{iyi/ıskarta}$	$F_{küçük}$	F_{iyi}	$F_{büyük}$
0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	0
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	0
0	1	0	1	0	0	0	0
0	1	1	0	0	1	0	0
0	1	1	1	1	0	1	0
1	0	0	0	0	1	0	0
1	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	0	0
1	1	0	0	0	1	0	0
1	1	0	1	0	0	0	0
1	1	1	0	1	0	1	0
1	1	1	1	0	0	0	1

$$F_{iyi\,/ıskarta} = \bar{a}bcd + abc\bar{d}bc = bc\left(\bar{a}d + a\bar{d}\right) = bc(a \oplus d)$$

10. FLİP-FLOPLAR

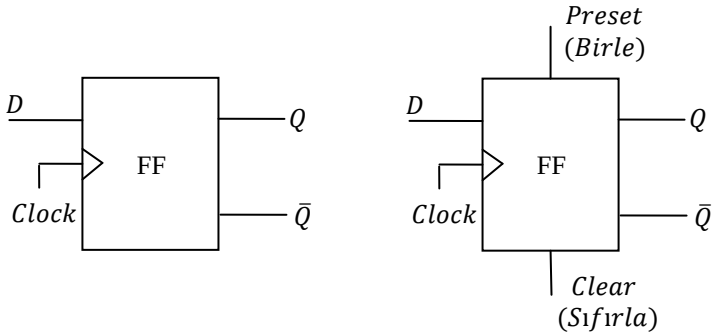
Flip-floplar 1 bitlik saklama birimleridir. En yalın saklama elemanı olarak tek başına kullanıldığı gibi birden çok flip-flop biraraya getirilip register (saklayıcı), sayıcı (counter), bellek veya özel amaçlı birimler oluşturulabilir. Mikroişlemcilerin içerisinde veye kartların üzerinde bayrak olarak adlandırılan değişiklikler de flip-flop' lar üzerinde tutulurlar.

Bir flip-flop' un temel olarak iki tane çıkış, bir tane saat girişi ve bir veya iki tane de veri giriş ucu vardır. Çıkış uçları birbirlerinin tümleyenidir; birisinin değeri Lojik1 ise, diğerininki Lojik 0' dır. Farklı türde flip-flop' lar tasarlanmıştır; her birinin özellikleri farklıdır ve tasarım esnekliği sağlar. Uygun flip-floplar seçilirse, tasarımda ortaya çıkan kombinasyonel devrenin karmaşıklığı azaltılabilir; çünkü devreye ait geçiş fonksiyonları sadeleşir.

Flip- flop' larda iki tür giriş vardır:

1. Senkron girişler: Flip-flop' larda kontrol girişleri olarak kullanılır. JK, SR, D ve T girişleri her bir flip-flop da senkron girişleri olarak adlandırılır. Bu girişlerin çıkışa etkileri clock darbeleriyle senkronize edilir.

2. Asenkron girişler: Bu girişler senkron girişlerin haricinde flip-flop çıkışını belli bir lojik değere atar (clock sinyalinin değişmesine 0' dan 1' e ya da 1' den 0' a geçmesine gerek kalmaksızın asenkron girişin değeri çıkış değiştirir).



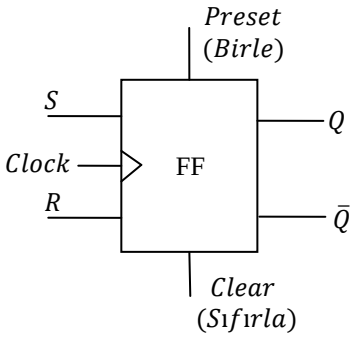
Şekil 10.1. Flip-flop' un genel gösterilişi

Yukarıdaki şekilde D veri girişidir, Q asıl çıkıştır ve $\bar{Q}$ çıkışın tümleyenidir. Her clock darbesinde D girişindeki veri Q çıkışına aktarılır. Senkronlama anında , girişlerindeki değerler ve filp-flop geçiş fonksiyonu kullanılarak çıkış üretilir. Eğer senkronlama anı, clock darbesinin yükselen kenarı ise yükselen kenarda tetiklenen, düşen kenarda ise düşen kenarda tetiklenen, lojik düzeye göre oluyorsa düzey tetiklemeli flip-flop olarak adlandırılır.

10.1. SR Flip-Flop

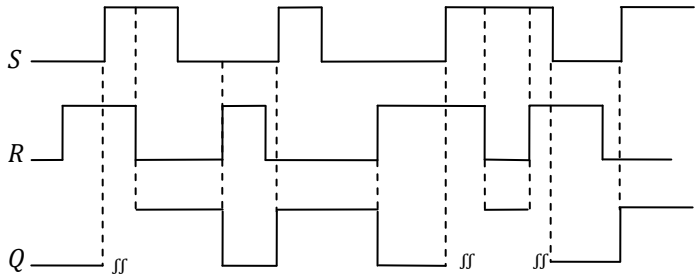
SR flip-flop' u S (Set) ve R (Reset) olarak adlandırılan iki girişe sahiptir. $t+1$ anındaki çıkış değeri SR girişlerinin t anındaki giriş değerlerine bağlıdır. Giriş/çıkış arasındaki bağıntı ve uyarma tablosu aşağıdaki gibidir (her iki girişin 1 olması durumunda geçiş olmaz). Giriş çıkış arasındaki bağıntı aşağıdaki gibidir:

$$Q(t+1) = Q(t)\overline{R(t)} + S(t) \qquad (R(t)\cdot S(t) = 0 \text{ durumları için}) \qquad (10.1)$$

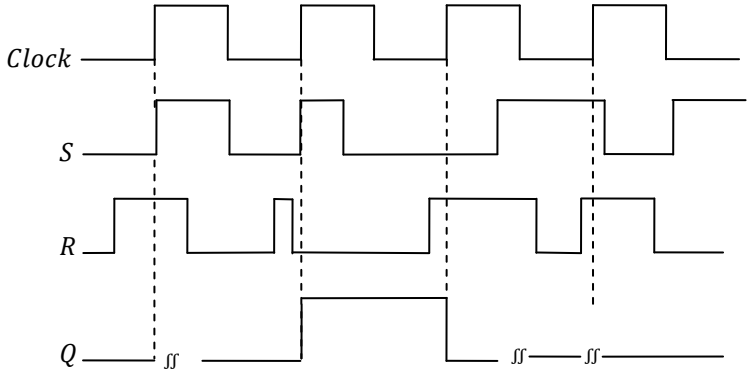


S	**R**	**Clock**	**Q**	$\bar{\boldsymbol{Q}}$
0	0	¶	Q	$\bar{Q}$
0	1	¶	0	1
1	0	¶	1	0
1	1	¶	Tanımsız	

Şekil 10.2 SR flip-flop' un genel gösterilişi ve uyarma tablosu



Şekil 10.3. SR flip-flop için darbe diyagramı

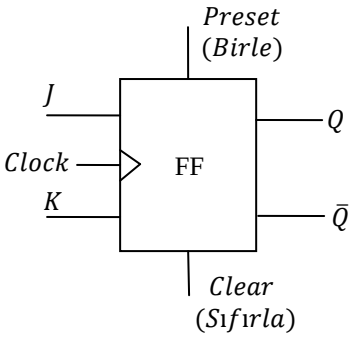


Şekil 10.4. Clock girişli SR flip-flop için darbe diyagramı
(Yükselen kenar tetiklemeli)

10.2. JK Flip-Flop

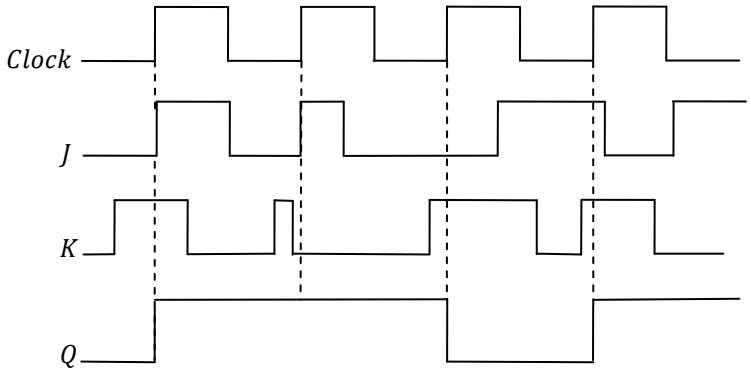
JK flip-flop' u SR flip-flop' a benzer. İki girişi vardır. SR' den farklı olarak iki girişin "1" olması durumunda da durum geçişi olur. Girişler ve çıkış arasındaki bağıntı aşağıdaki gibidir:

$$Q(t+1) = Q(t)\overline{K(t)} + \bar{Q}(t)J(t) \tag{10.2}$$



J	K	$Clock$	Q	$\bar{Q}$
0	0	¶	Q	$\bar{Q}$
0	1	¶	0	1
1	0	¶	1	0
1	1	¶	$\bar{Q}$	Q

Şekil 10.5. JK flip-flop' un genel gösterilişi ve uyarma tablosu

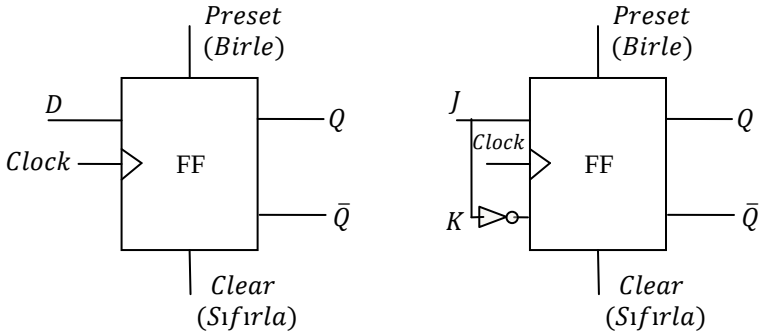


Şekil 10.6. JK flip-flop için darbe diyagramı

10.3. D Flip-Flop

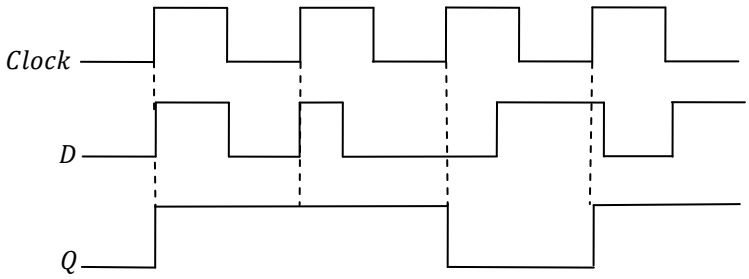
D flip-flop' un D olarak adlandırılan bir girişi vardır. D ucundaki giriş değeri bir sonraki çıkış bilgisidir. Giriş ve çıkış arasındaki bağıntı aşağıdaki gibidir:

$$Q(t+1) = D(t) \tag{10.3}$$



D	Clock	Q	$\bar{Q}$
0	¶	0	1
1	¶	1	0

Şekil 10.7. D flip-flop' un genel gösterilişi ve uyarma tablosu

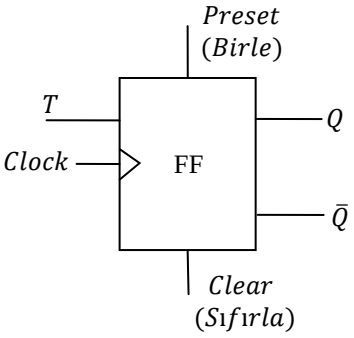


Şekil 10.8. D flip-flop için darbe diyagramı

10.4. T Flip-Flop

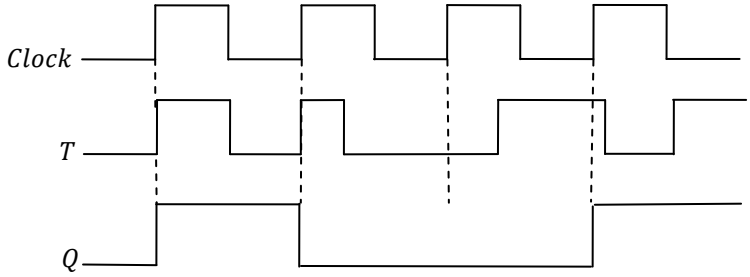
T flip-flop' un T (Toggle) olarak adlandırılan bir girişi vardır. Diğerlerinden biraz farklı bir flip-flop' dur. $T = 0$ kaldığı sürece flip-flop durum değiştirmez. $T = 1$ olursa durum değiştirir. JK girişleri birleştirilir ve tek bir giriş yapılırsa T flip-flop elde edilmiş olur. Giriş ve çıkış arasındaki bağıntı aşağıdaki gibidir:

$$Q(t+1) = Q(t) \oplus T(t) \tag{10.4}$$



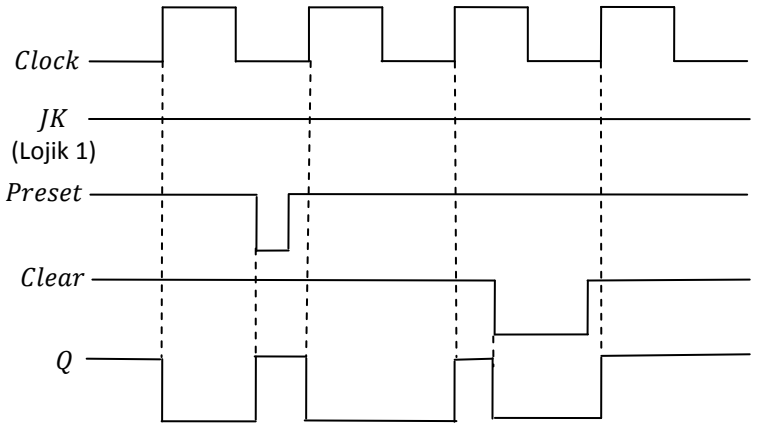
T	Clock	Q	$\bar{Q}$
0	¶	Q	$\bar{Q}$
1	¶	$\bar{Q}$	Q

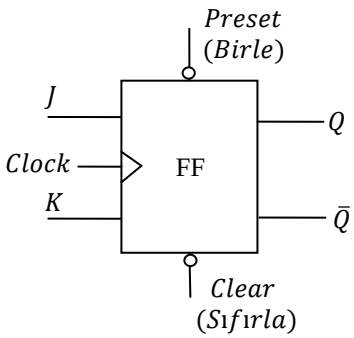
Şekil 10.9. T flip-flop' un genel gösterilişi ve uyarma tablosu



Şekil 10.10. T flip-flop için darbe diyagramı

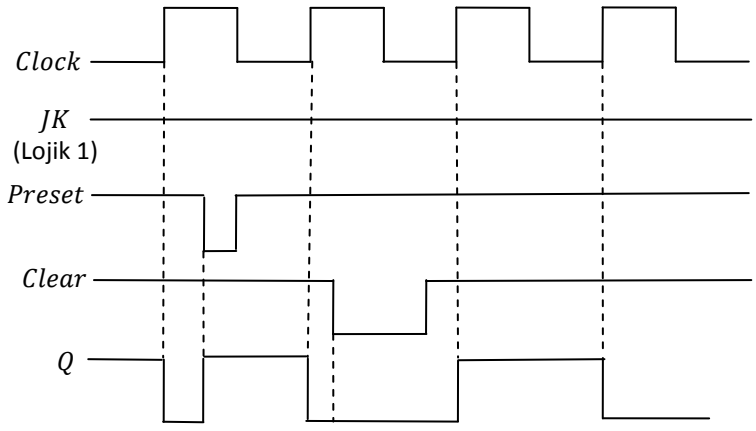
Örnek 10.1. JK, Clock, preset ve clear girişleri verilen flip-flop' un çıkış darbe diyagramını çiziniz (Yükselen kenar tetiklemeli).





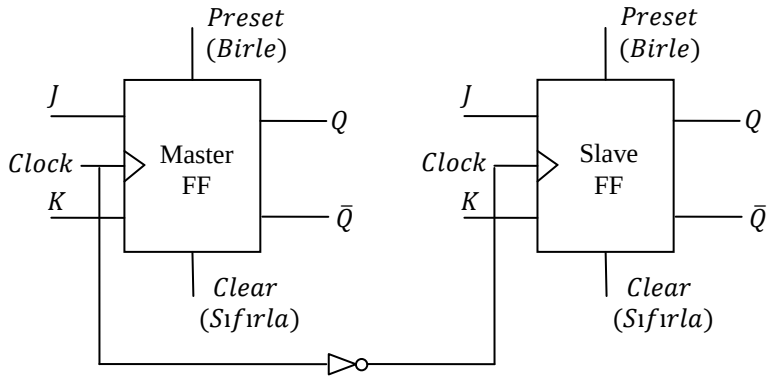
Preset	Clear	Q	$\bar{Q}$
0	0	Tanımsız	
0	1	1	0
1	0	0	1
1	1	Çıkışa etkisi yok	

Örnek 10.2. JK, Clock, preset ve clear girişleri verilen flip-flop' un çıkış darbe diyagramını çiziniz.



10.5. Master-Slave (Ana-Uydu) Flip-Flop

Master-slave flip-flop iki tane flip-flop' un bağlanmasıyla elde edilen ve tetikleme periyodu içerisinde girişindeki yeni değeri alırken çıkışındaki bir önceki değeri bir süre tutan flip-flop düzeneğidir. Şöyle ki; bir flip-flop' un çıkışları, girişlerindeki değerlere bağlı olarak, tetikleme anında yeni durumunu alır. Master-slave flip-flop' larda ise, iki tane flip-flop' tan oluştuğu için girişlere bağlı yeni çıkış değerleri tetikleme anında çıkışlara hemen yansıtılmaz; bir süre gecikmeyle yansıtılır. Bu gecikme kare dalga şeklindeki bir clock işareti için $^1/_2$ periyottur. Aşağıdaki şekilde bir master-slave flip-flop mimarisi görülmektedir. Clock işareti ana olarak adlandırılan birinci flip-flop' a doğrudan uygulanırken, uydu olarak adlandırılan ikinci flip-flop' a tümlenerek uygulanmaktadır.



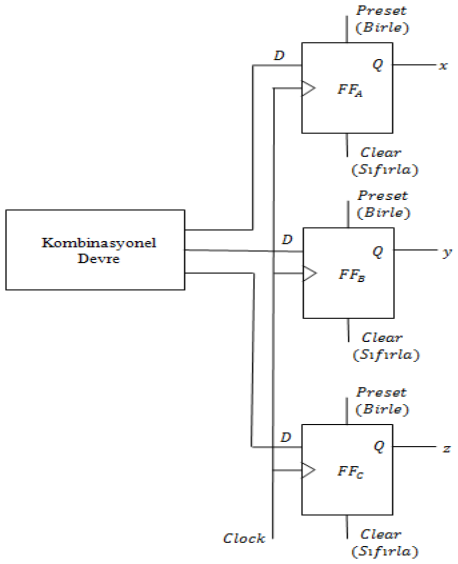
Şekil 10.11. Master-Slave flip-flop mimarisi

Yükselen tetiklemeli flip-floplar kullanıldığı varsayılırsa ana flip-flop clock darbesinin yükselen kenarında girişlerindeki değere karşılık gelen durumu içerisinde tutar, bu durum da uydu flip-flop durum değiştirmez. Çünkü clock işareti uyduya tümlenerek verilmiştir. Dolayısıyla uydu flip-flop saatin düşen kenarına kadar çıkışlarını değiştirmez.

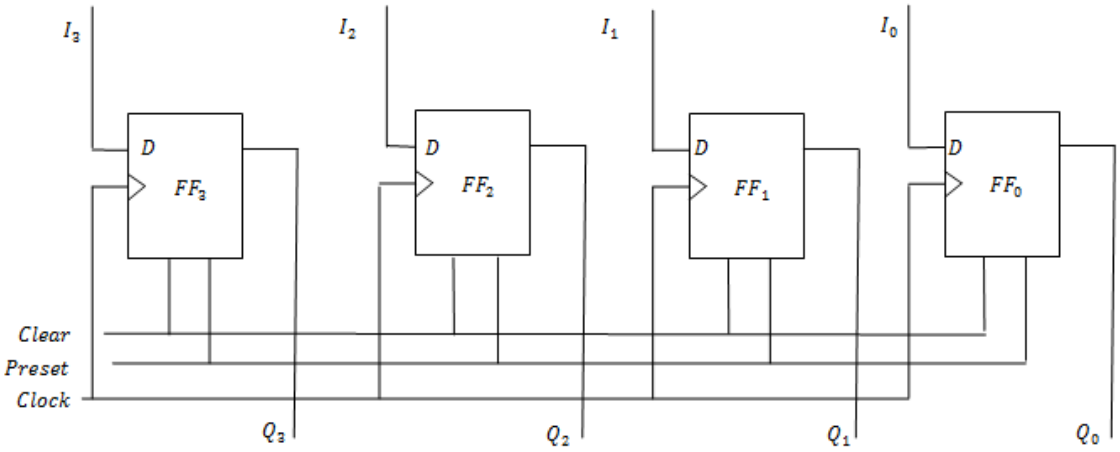
10.6. Flip-Flop Uyarma Tablosu

$Q(t)$	$Q(t+1)$	S	R	J	K	D	T
0	0	0	X	0	X	0	0
0	1	1	0	1	X	1	1
1	0	0	1	X	1	0	1
1	1	X	0	X	0	1	0

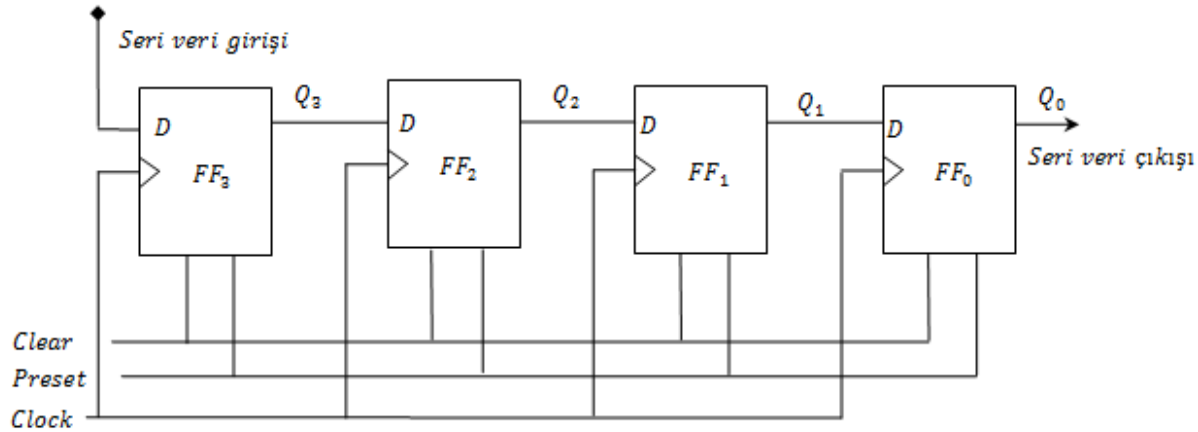
Örnek 10.3. Flip-floplarla paralel veri transferini gerçekleştiriniz.



Örnek 10.4. Flip-floplarla seri veri transferini gerçekleştiriniz.



Örnek 10.5. D flip-floplarla 4 bitlik ötelemeli (shift) bir register (saklayıcı) gerçekleyiniz.



Örnek 10.6. İki adet JK flip-flop ve gerektiği kadar kapı kullanarak a ve b şıklarında istenen devreleri tasarlayınız.

a) 1' e tümleyici devresi

b) 2' ye tümleyici devresi

A	B	A^1	B^1	J_{A^1}	K_{A^1}	J_{B^1}	K_{B^1}
0	0	1	1	1	X	1	X
0	1	1	0	1	X	X	1
1	0	0	1	X	1	1	X
1	1	0	0	X	1	X	1

A \ B	0	1
0	1	1
1	X	X

A \ B	0	1
0	X	X
1	1	1

A \ B	0	1
0	1	X
1	1	X

A \ B	0	1
0	X	1
1	X	1

$$J_{A^1} = K_{A^1} = J_{B^1} = K_{B^1} = 1$$

Örnek 10.7. Bir oylamaya katılan 4 kişi, kararlarını önlerinde bulunan butonlara basarak belirtiyorlar. Teklifi kabul eden kişi butona basmakta, aksi halde basmamaktadır. Teklifin kabul edilmesi için en az 2 kişinin kabul etmesi ve bir önceki teklifin kabul edilmemiş olması gerekmektedir. Başlangıçta bir önceki teklifin kabul edilmediği varsayılarak bu işlemi gerçekleştirecek devreyi tasarlayınız ve şemasını çiziniz (JK ve D tipi flip-floplarla ayrı ayrı gerçekleştirilecektir.).

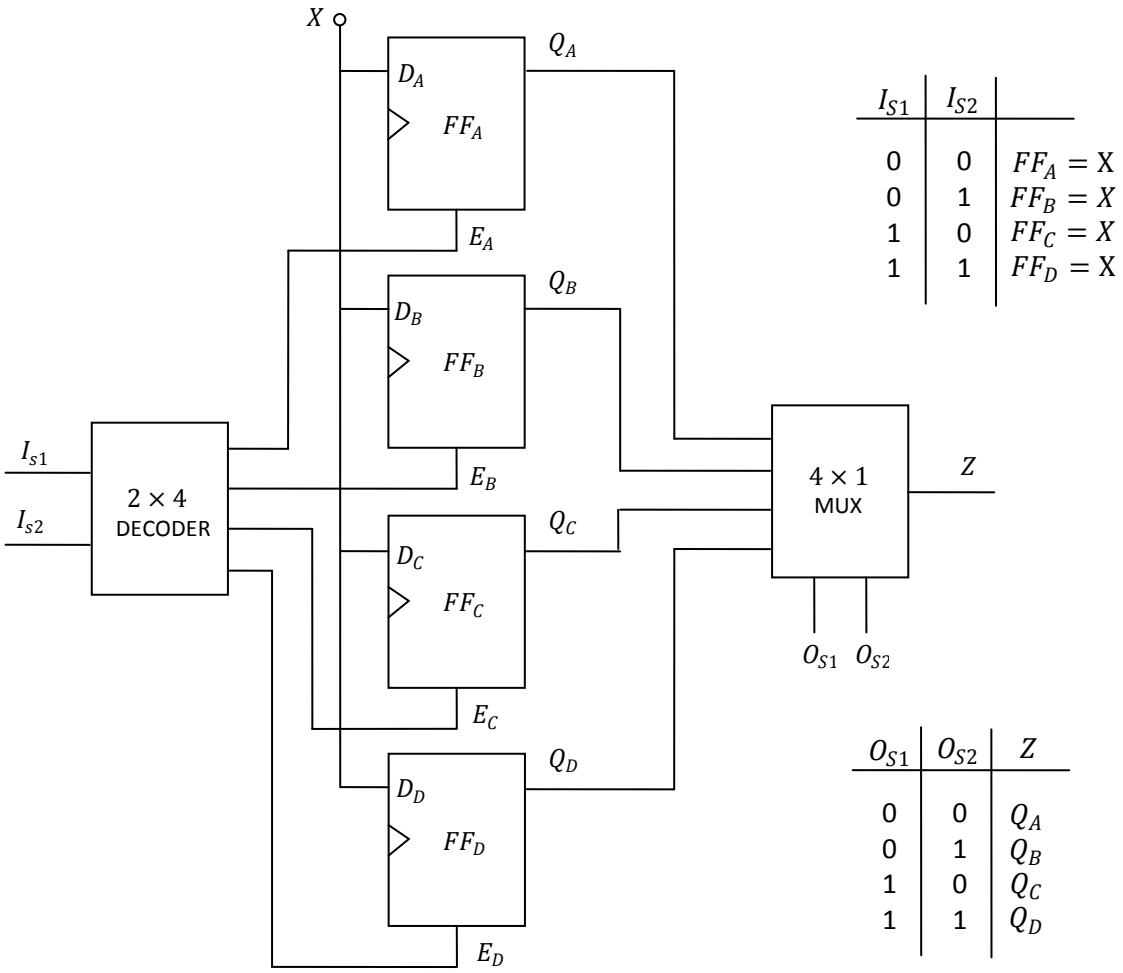
A	B	C	D	T_{ilk}	T_{son}	J_T	K_T	D_T
0	0	0	0	0	0	0	X	0
0	0	0	1	0	0	0	X	0
0	0	1	0	0	0	0	X	0
0	0	1	1	0	1	1	X	1
0	1	0	0	0	0	0	X	0
0	1	0	1	0	1	1	X	1
0	1	1	0	0	1	1	X	1
0	1	1	1	0	1	1	X	1
1	0	0	0	0	0	0	X	0
1	0	0	1	0	1	1	X	1
1	0	1	0	0	1	1	X	1
1	0	1	1	0	1	1	X	1
1	1	0	0	0	1	1	X	1
1	1	0	1	0	1	1	X	1
1	1	1	0	0	1	1	X	1
1	1	1	1	0	1	1	X	1

AB \ CD	00	01	11	10
00	0	0	1	0
01	0	1	1	1
11	1	1	1	1
10	0	1	1	1

$$J_T = AB + CD + BD + AD + BC + AC$$

$$D_T = J_T$$

Örnek 10.8. Bir adet veri girişi (X), bir adet veri çıkışı (Z) olan 4 adet D tipi izin girişli flip-flop içeren bir devre tasarlanacaktır. Devrenin kullanıcısı X' den gelen veriyi bu flip-floplardan istediğine yazdırabilecektir. Z çıkışında da istediği flip-flop' un içeriğini görebilecektir. Devrenin gerekli sayıda giriş seçme (I_s) ve çıkış seçme (O_s) uçları olacaktır. I_s girişleri verinin hangi flip-flop' a yazılacağını, O_s bilgileri ise hangi flip-flop' un içeriğinin çıkışa aktarılacağını belirleyecektir. Bu devreyi gerekli boyutlarda bir adet decoder (kod çözücü) ve bir adet multiplexer (veri seçici) kullanarak tasarlayınız.



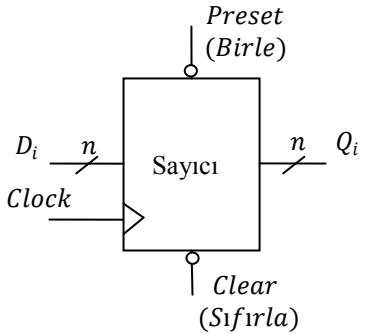
I_{S1}	I_{S2}	
0	0	$FF_A = X$
0	1	$FF_B = X$
1	0	$FF_C = X$
1	1	$FF_D = X$

O_{S1}	O_{S2}	Z
0	0	Q_A
0	1	Q_B
1	0	Q_C
1	1	Q_D

11. SAYICILAR

Sayıcılar, registerlar da olduğu gibi n bitlik bilgi tutmanın yanı sıra her saat çevriminde tuttukları değeri belirli bir sırada değiştiren elemanlardır; artırır veya azaltır. Bir sayıcı genel olarak, aşağıdaki 4 özelliğe göre sınıflandırılabilir:

- Bit sayısına göre: 4 bitlik, 8 bitlik, 32 bitlik gibi
- Sayma sırası ve şekline göre: yukarı (ileri), aşağı (geri), tek sayılar, özel sayılar gibi
- Durumların sayısına göre: modülo sayıcı, 5 durumlu sayıcı gibi
- Geçiş fonksiyonlarının zamanlamasına göre: senkron, asenkron gibi



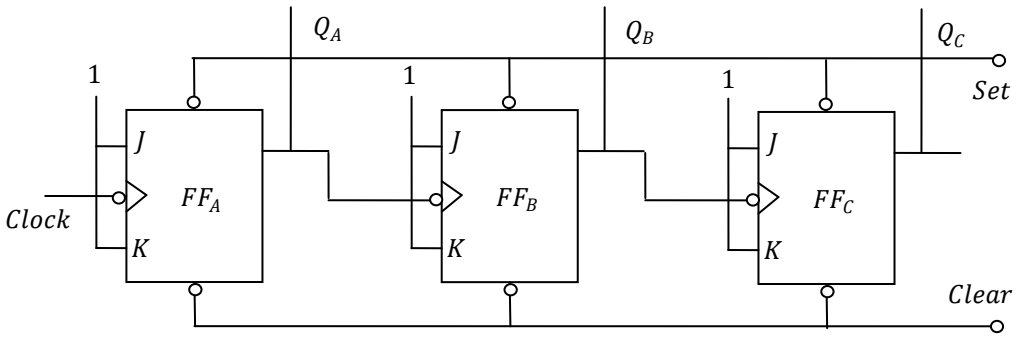
$$Q(t+1) = \begin{cases} D(t) & \text{Preset etkin} \\ 0 & \text{Clear etkin} \\ Q(t)+1 & \text{Clock darbesi} \\ Q(t) & \text{Diğer durumlar} \end{cases}$$

Şekil 11.1. Bir sayıcı için blok gösterim

NOT: Clock darbesi sayıcıyı oluşturan flip-flopların hepsine aynı anda veriliyorsa böyle bir sayıcıya "***senkron sayıcı***" adı verilir. Sayıcıyı oluşturan flip-floplar ardışık olarak birbirini tetikliyorsa buna da "***asenkron sayıcı***" denir. Uygulamalarda çoğunlukla asenkron sayıcı kullanılır. Piyasada buna uygun entegreler mevcuttur (7490, 7493, …)

11.1. Binary (İkili) İleri Sayıcı

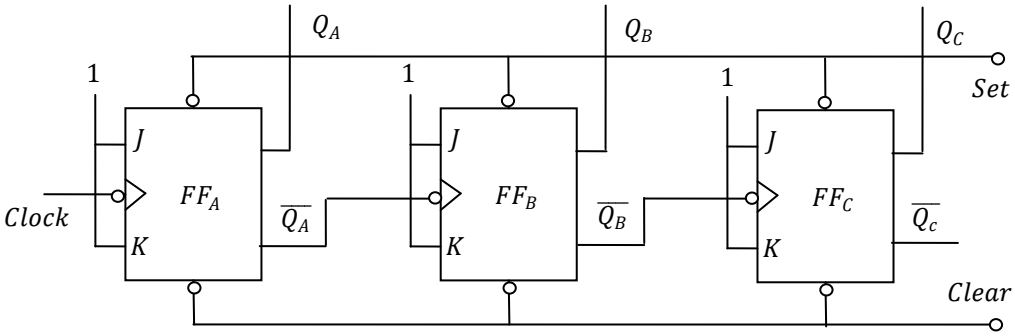
n bitlik çıkış ucu ve 2^n tane değişik durum vardır. Aksi belirtilmediği sürece 0' dan $2^n - 1$' e kadar birer artımla saydığı varsayılır.



ilk	*Clock*	son
CBA		***CBA***
000	¶	001
001	¶	010
010	¶	011
011	¶	100
100	¶	101
101	¶	110
110	¶	111
111	¶	000

Şekil 11.2. Binary ileri sayıcı

11.2. Binary (İkili) Geri Sayıcı

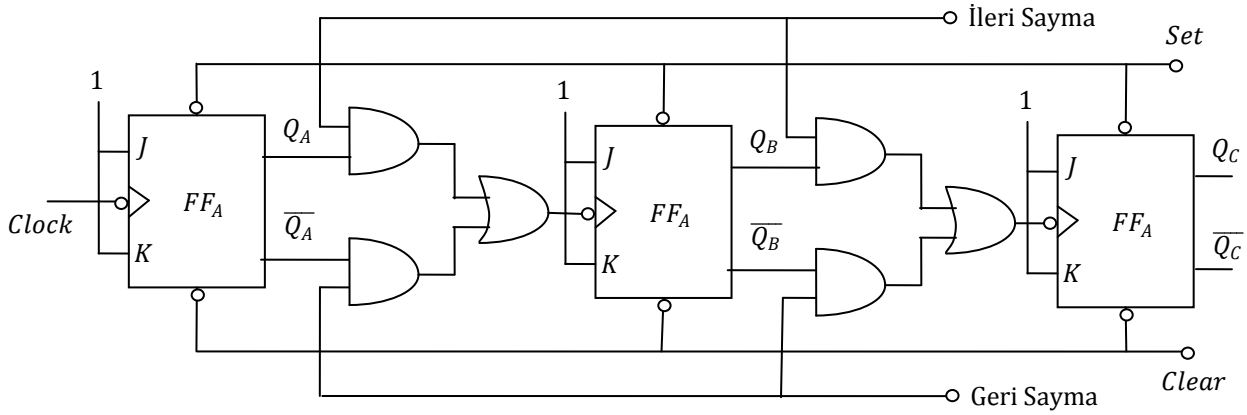


ilk	*Clock*	son
CBA		***CBA***
000	¶	111
111	¶	110
110	¶	101
101	¶	100
100	¶	011
011	¶	010
010	¶	001
001	¶	000

Şekil 11.3. Binary geri sayıcı

NOT: Mod $= 2^{\text{FF sayısı}}$ Mesela, Mod 8 sayıcı için $8 = 2^{n} \Longrightarrow n = 3$ adet FF kullanılması gerekir. Sayıcı $2^{n} - 1 = 8 - 1 = 7$' e kadar sayar.

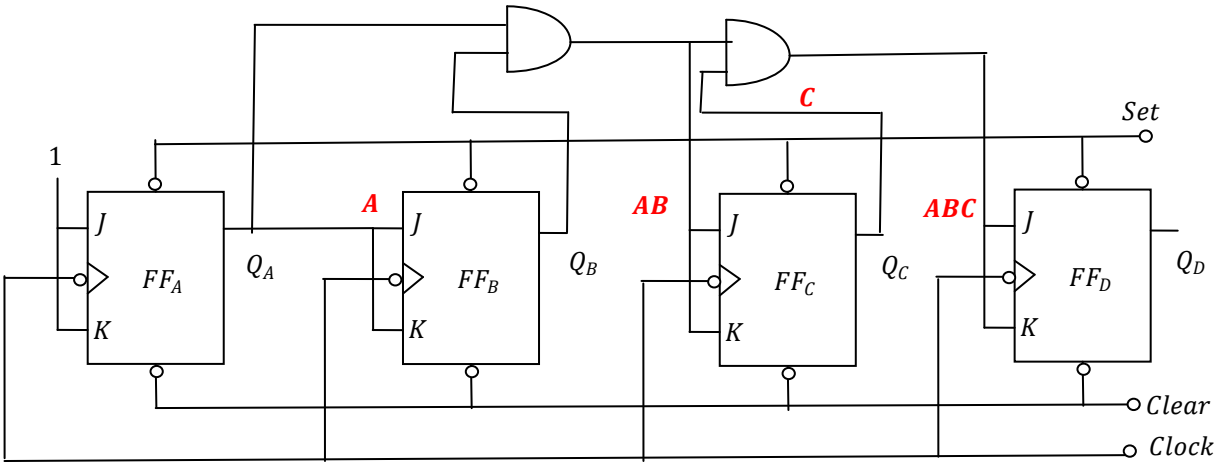
11.3. Asenkron İleri-Geri Sayıcı



Şekil 11.4. Asenkron ileri/ geri sayıcı

İleri sayma	Geri sayma	İşlem
0	0	Sayma yok
0	1	Geri sayma
1	0	İleri sayma
1	1	Sayma yok

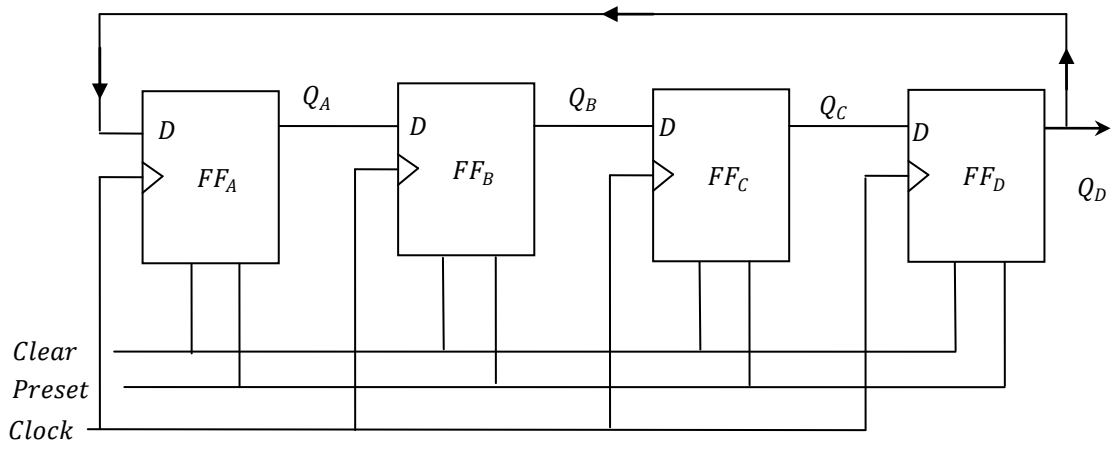
11.4. Senkron Sayıcı



Şekil 11.5. Senkron ileri/ geri sayıcı

11.5. Ring (Halka) Sayıcı

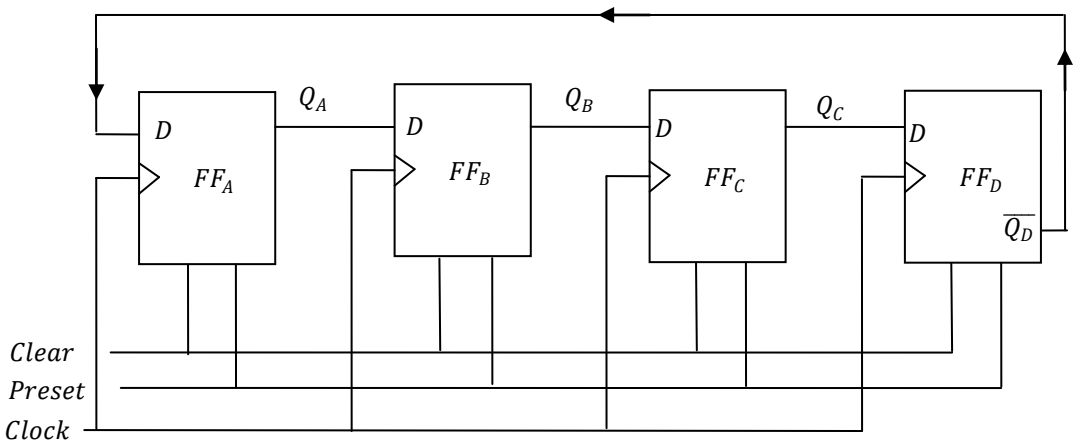
Ring sayıcı shift register gibi içerisindeki değeri lojik olarak sağa veya sola öteleyerek çalışır. Bu sayıcıda flip-floplardan bir tanesinin içinde Lojik 1 depolu olmalıdır.



Şekil 11.6. Ring (Halka) sayıcı

A	B	C	D	İşlem
1	0	0	0	Başlangıç
0	1	0	0	¶
0	0	1	0	¶
0	0	0	1	¶

11.6. Johnson Sayıcı (Dalgalı Halka Sayıcı)

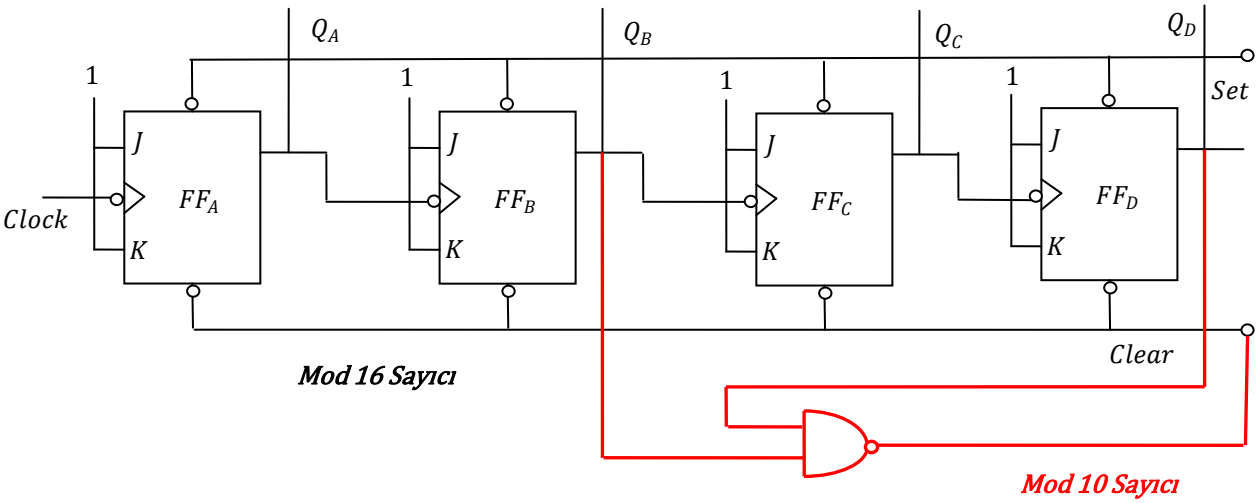


Şekil 11.7.Johnson sayıcı

A	B	C	D	İşlem
0	0	0	0	Başlangıç
1	0	0	0	¶
1	1	0	0	¶
1	1	1	0	¶
1	1	1	1	¶
0	1	1	1	¶
0	0	1	1	¶
0	0	0	1	¶
0	0	0	0	¶

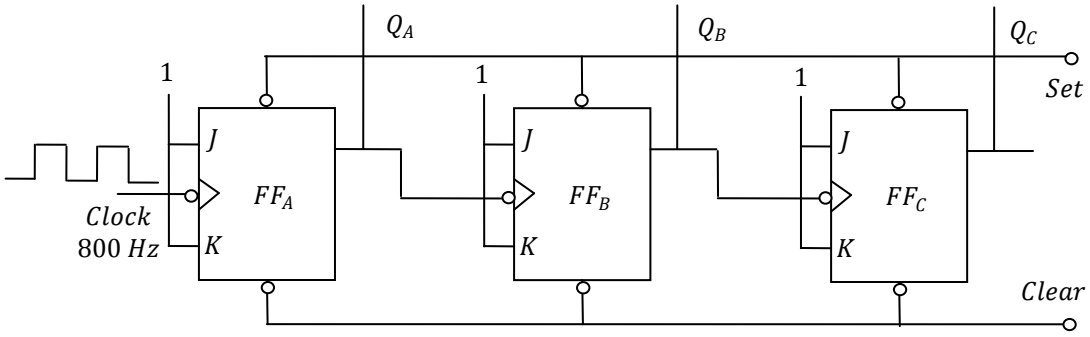
11.7. Sayıcıları İstenilen Bir Sayıda Durdurma

Sayıcıların istenilen bir sayıda durdurulmasında FF' ları clear (reset) uçları kullanılır. 4 bitlik sayıcı asenkron sayıcı içeren entegrelerde genellikle 2 reset girişi vardır.

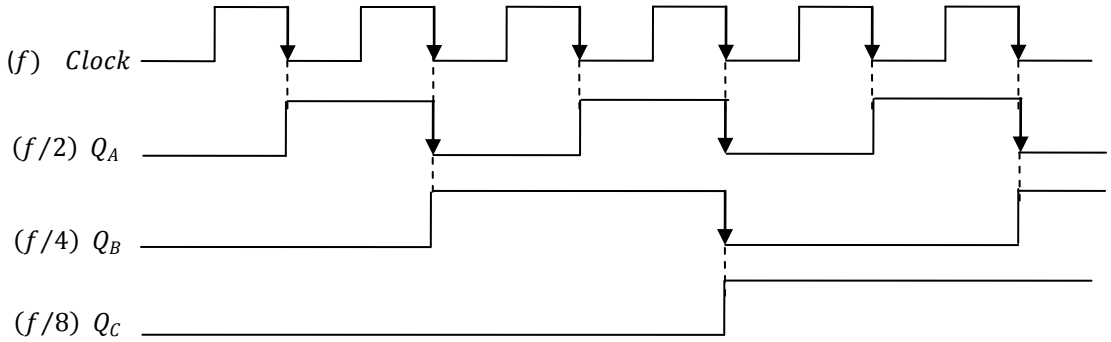


Şekil 11.8. Mod 16 ve Mod10 asenkron sayıcı

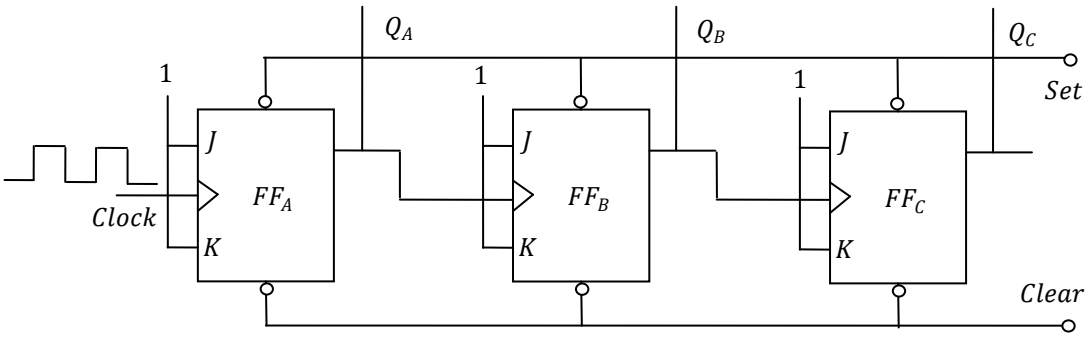
11.8. Sayıcıların Frekans Bölücü Olarak Kullanılması



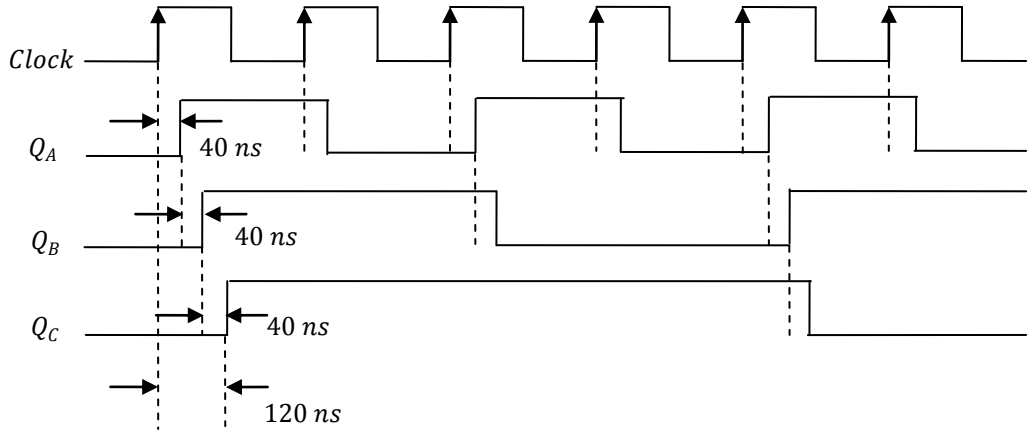
Şekil 11.9. Sayıcıların frekans bölücü olarak kullanılması



11.9. Sayıcılarda Propagasyon (Yayılma) Gecikmesi



Şekil 11.10. Sayıcılarda propagasyon gecikmesi



Clock frekansı toplam yayılma gecikmesine göre hesaplanır. Bir flip-flop' un yayılma gecikmesi yaklaşık 40 ns' dir. Bir lojik kapının ise yaklaşık 20 ns' dir. Asenkron sayıcılarda

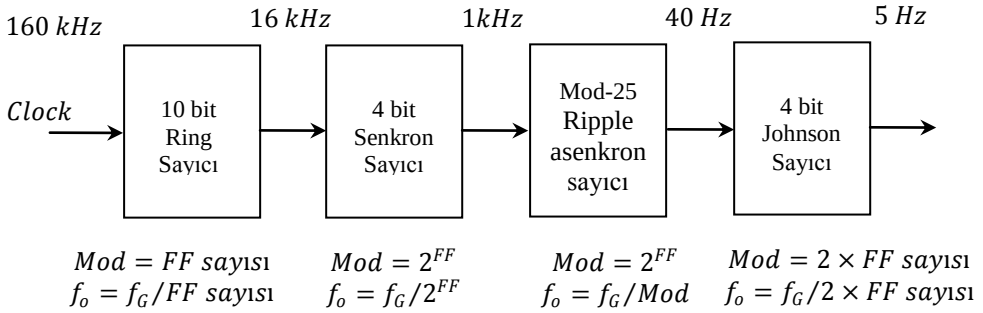
$$Toplam\ yayılma\ gecikmesi = FF\ sayısı \times Bir\ FF'nin\ \ yayılma\ gecikmesi + Sistemdeki\ lojik\ kapıların\ yayılma\ gecikmesi$$

şeklinde hesaplanır. Senkron sayıcılarda ise

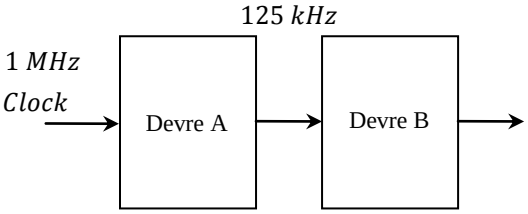
$$Toplam\ yayılma\ gecikmesi = Bir\ FF'nin\ \ yayılma\ gecikmesi + Sistemdeki\ lojik\ kapıların\ yayılma\ gecikmesi$$

eşitliği ile hesaplanabilir.

Örnek 11.1.



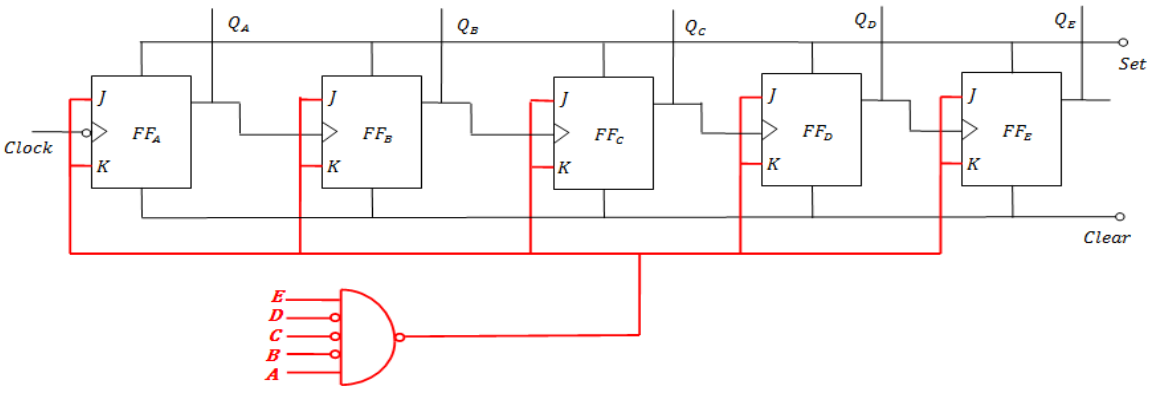
Örnek 11.2.



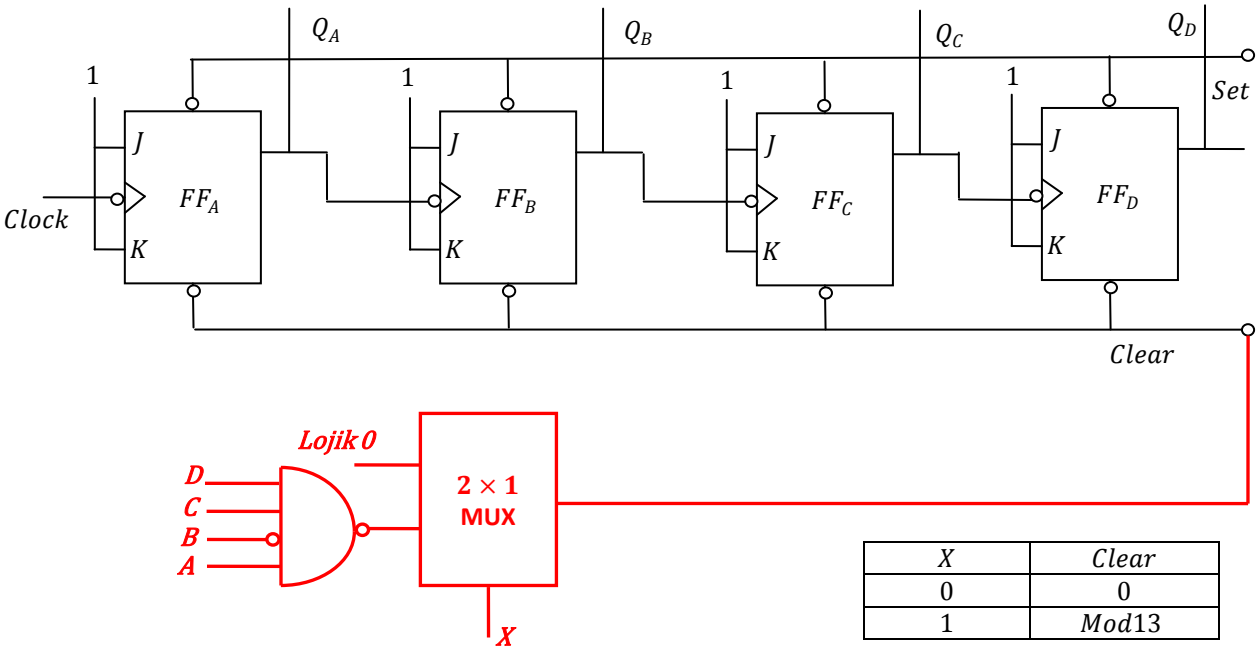
$$Mod = {}^{1\ MHz}/_{125\ kHz} = 8$$

$$Devre\ A \Longrightarrow Mod\ 8\ sayıcı$$

Örnek 11.3. 17' e kadar sayan ve 17' de duran asenkron sayıcıyı çiziniz (JK-FF kullanılacak olup, FF' lerin set ve clear uçları inversli değildir). $17 \Longrightarrow EDCBA = 10001$

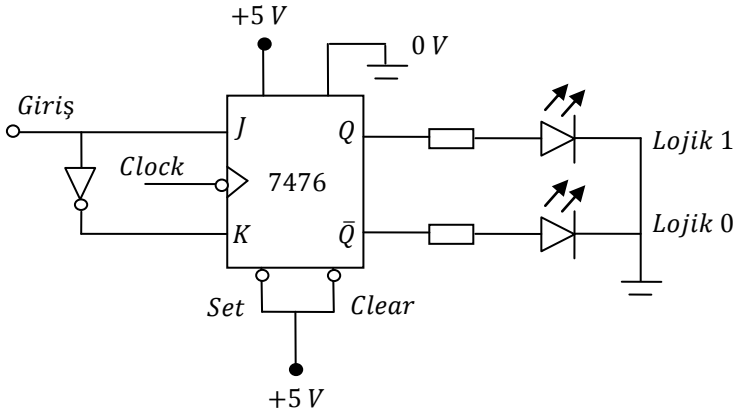


Örnek 11.4. Asenkron bir sayıcı tasarlanacaktır. Bu sayıcıya eklenecek X kontrol ucu sayesinde sayıcı "0" olabilecek ve istenen modda maksimum sayıyı gösterebilecektir ($X = 0$' da sayıcı sıfırlanacak, $X = 1$ olduğunda maksimum sayıda duracaktır). Böyle bir sisteme uygun Mod 13 asenkron sayıcıyı X kontrol ucuna bağlı olarak gerçekleştiriniz (Set ve clear uçları inversli JK flip-flop kullanılacaktır).

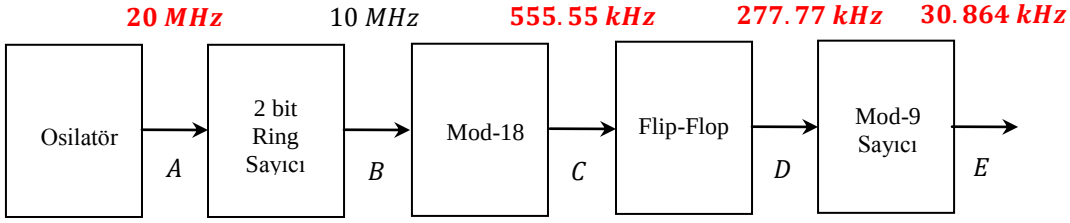


X	$Clear$
0	0
1	$Mod13$

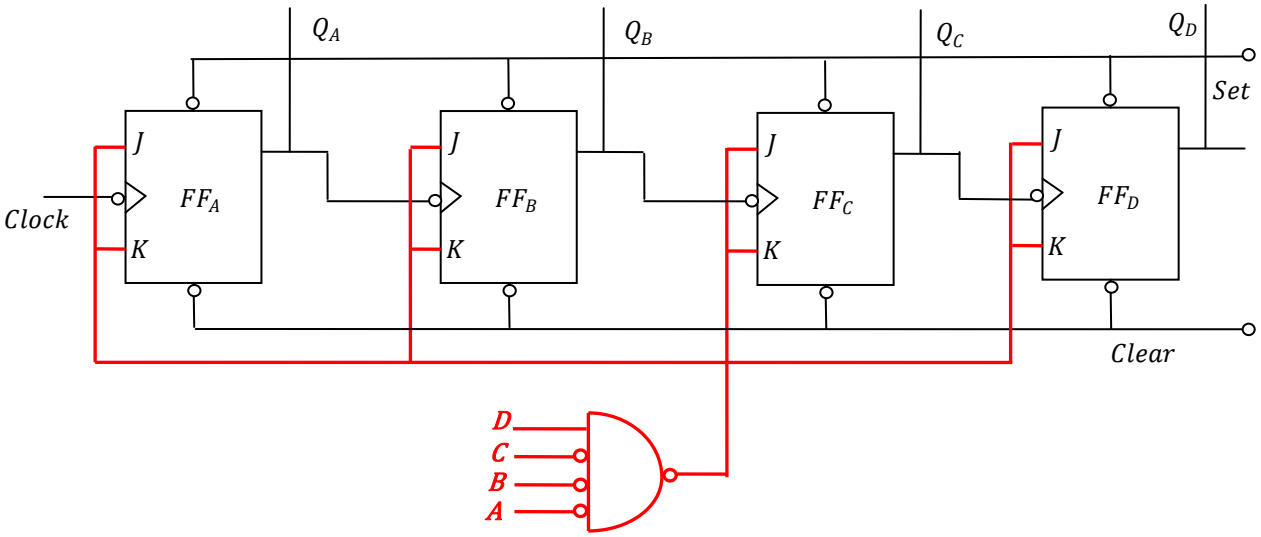
Örnek 11.5. 7476 JK flip-flop' u kullanarak bir lojik prob tasarlayıp devre şemasını eksiksiz, çalışır durumda çiziniz.



Örnek 11.6. B' deki sinyal 10 MHz ise A, C, D ve E noktalarındaki sinyalin frekanslarını bulunuz.



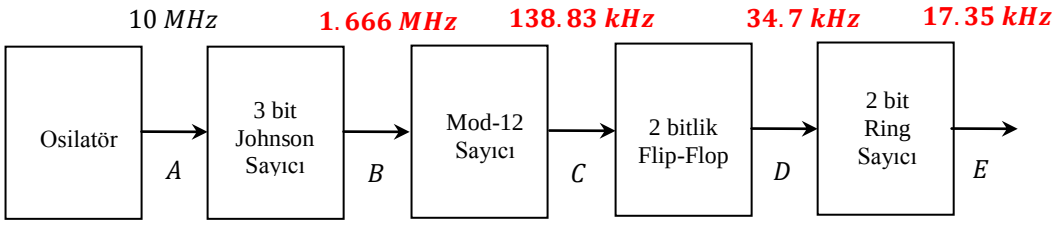
Örnek 11.7. 8' e kadar sayan ve 8' de duran asenkron sayıcıyı JK kullanarak tasarlayınız. (Set ve Clear uçları inversli JK flip-flop kullanarak tasarlayınız.)



Örnek 11.8. ***(ÖDEV)*** Üç girişi ve 1 çıkış olan bir çoğunluk dedektör devresi (kombinasyonel devre) tasarlanacaktır. Girişlerin çoğunda "1" olduğunda çıkış lojik 1, girişlerin çoğunda "0" olduğunda çıkış Lojik 0 olacaktır. Doğruluk tablosunu oluşturarak, Karnaugh diyagramı yardımıyla maksimum terimler kanonik açılımı şeklinde gerekli lojik fonksiyonu elde ediniz. Bu fonksiyonu gerçekleştirecek lojik devreyi çiziniz.

Örnek 11.9. ***(ÖDEV)*** Girişine verilen 4 bitlik sayıların tek veya çift olduğunu gösteren lojik devreyi tasarlayınız. Gerekli devre şemasını çiziniz.

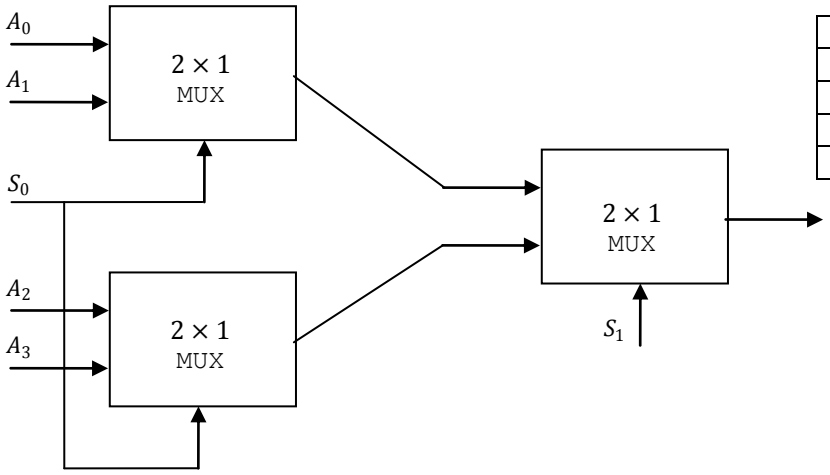
Örnek 11.10.



$f_B = {10\ MHz}/{2 \times 3} = 1.666\ MHz,\ f_C = {1.666\ MHz}/{12} = 138.83\ kHz$

$f_D = {138.83\ kHz}/{2^2} = 34.7\ kHz,\quad f_E = {34.7\ kHz}/{2} = 17.35\ kHz$

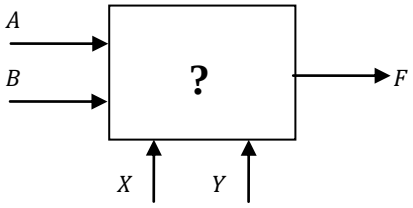
Örnek 11.11. 2×1 MUX kullanarak 4×1 MUX' u blok olarak tasarlayınız. Tasarladığınız MUX' a ait seçme uçlarına ait tabloyu oluşturunuz.



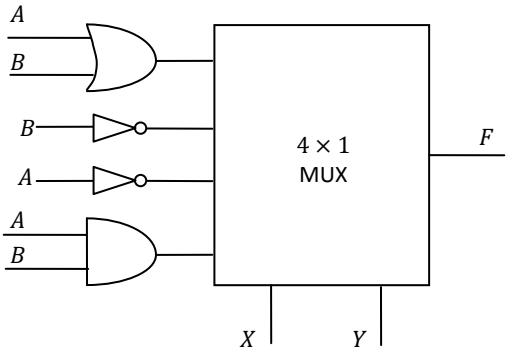
S_1	S_0	$Output$
0	0	A_0
0	1	A_1
1	0	A_2
1	1	A_3

Örnek 11.12.

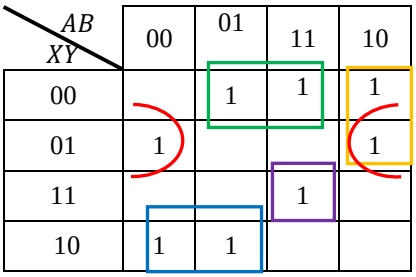
X	Y	F
0	0	$A + B$
0	1	$\bar{B}$
1	0	$\bar{A}$
1	1	$A.B$



X, Y girişlerine bağlı olarak F çıkışının değeri aşağıdaki tabloda verilmiştir. F çıkışına ait fonksiyonu elde ediniz ve devreyi gerçekleyiniz.

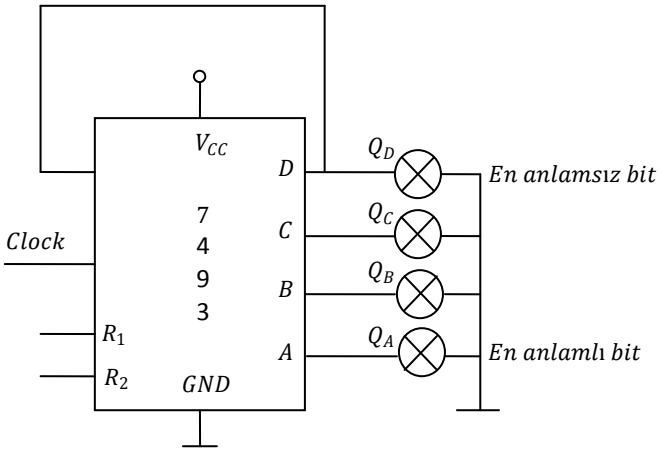


X	Y	A	B	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1



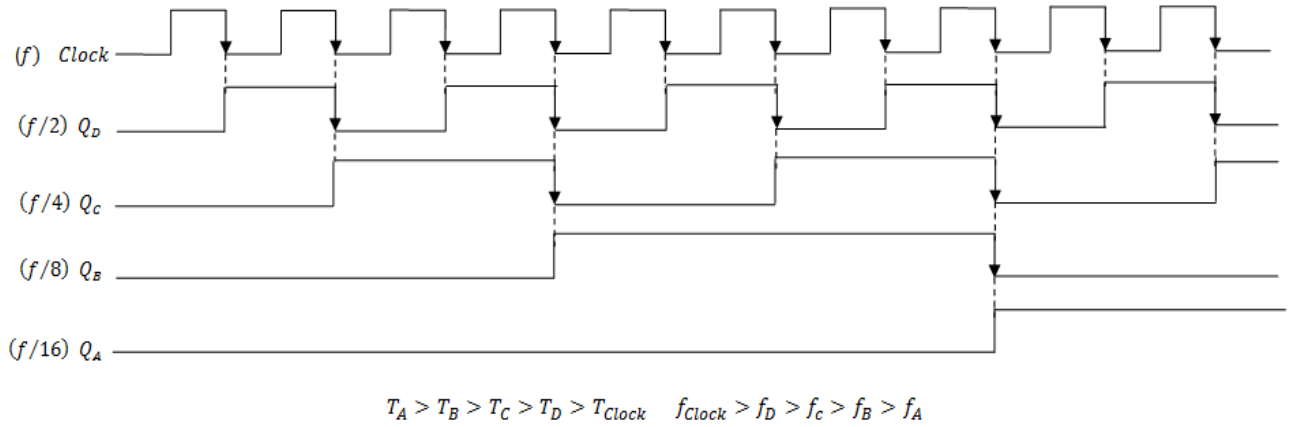
$$F = \bar{X}\bar{Y}B + A\bar{B}\bar{X} + \bar{X}Y\bar{B} + X\bar{Y}\bar{A} + ABXY$$

11.10. 7493 Entegresi - 4 Bitlik Asenkron Sayıcı

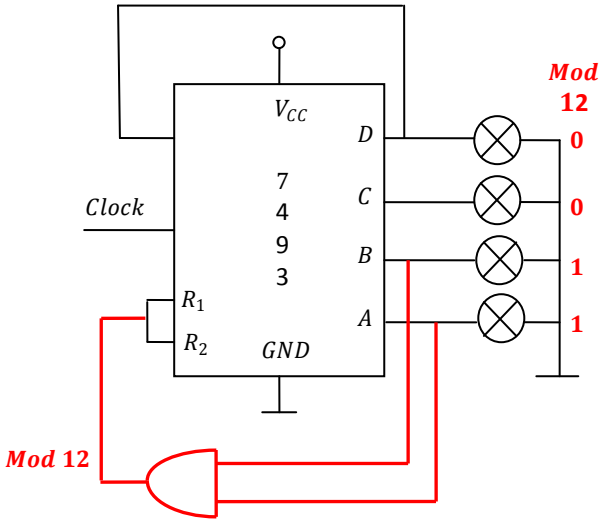


Şekil 11.11. 7493 Entegresi-4 bitlik asenkron sayıcı

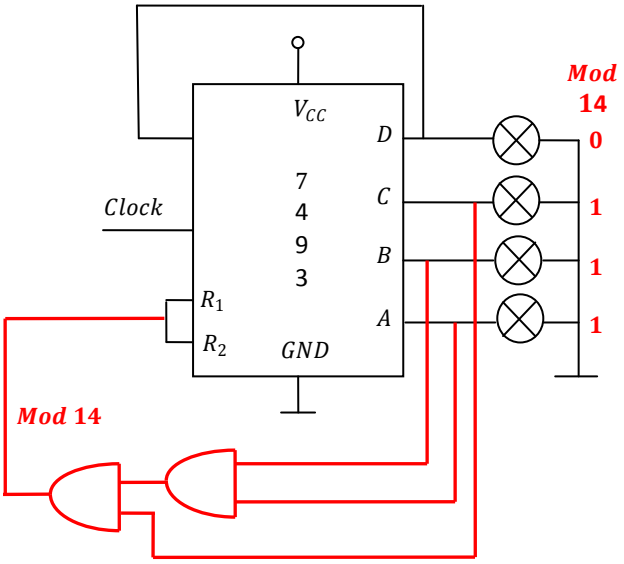
Sayıcının resetlenmesi için R_1 ve R_2 girişlerinin her ikisine birden aynı anda "Lojik 1" verilmesi gerekir.



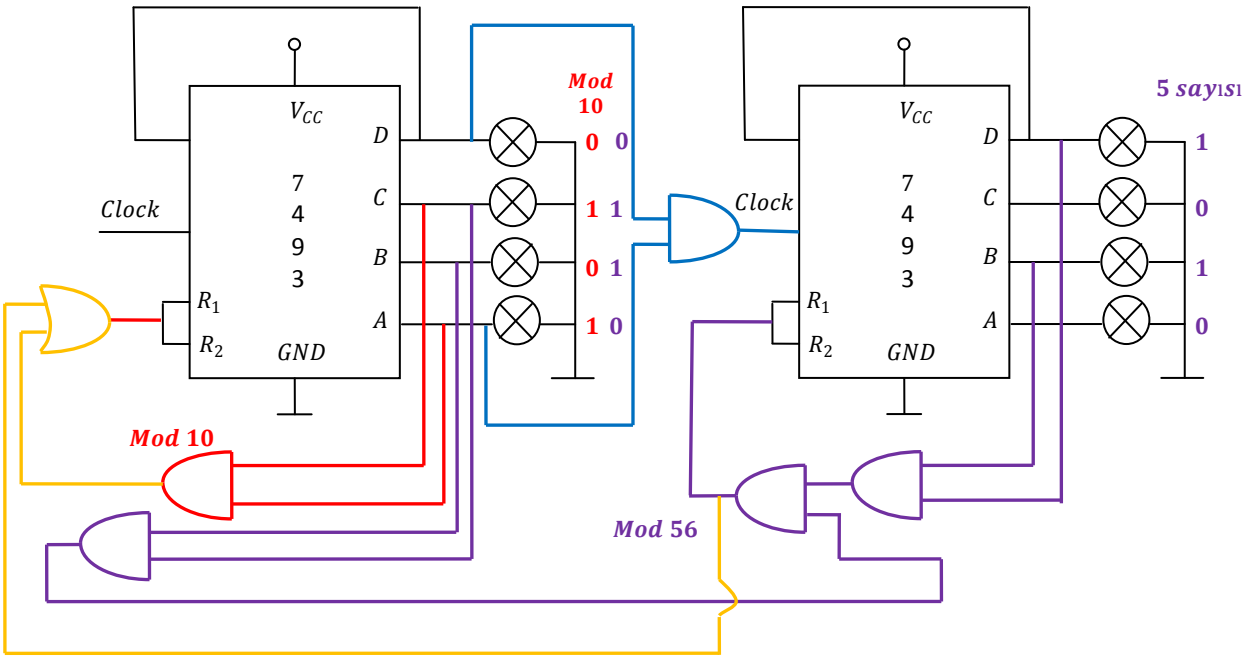
Örnek 11.13. Mod 12 sayıcıyı 7493 entegresi kullanarak tasarlayınız.



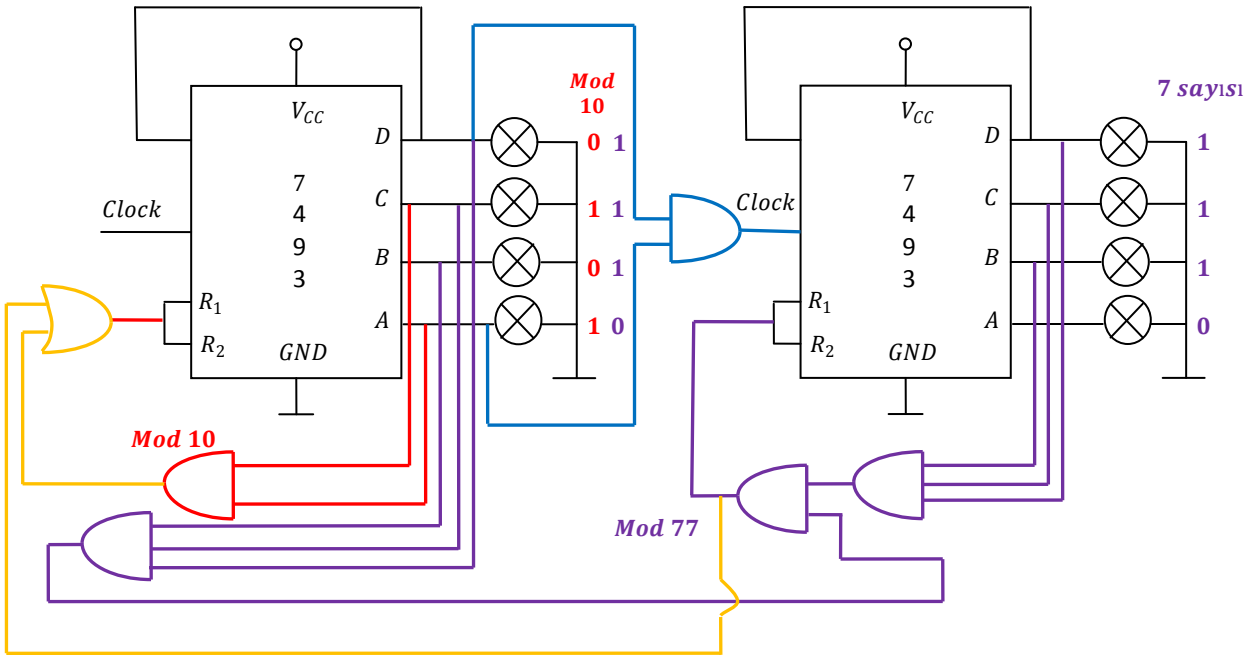
Örnek 11.14. Mod 14 sayıcıyı 7493 entegresi kullanarak tasarlayınız.



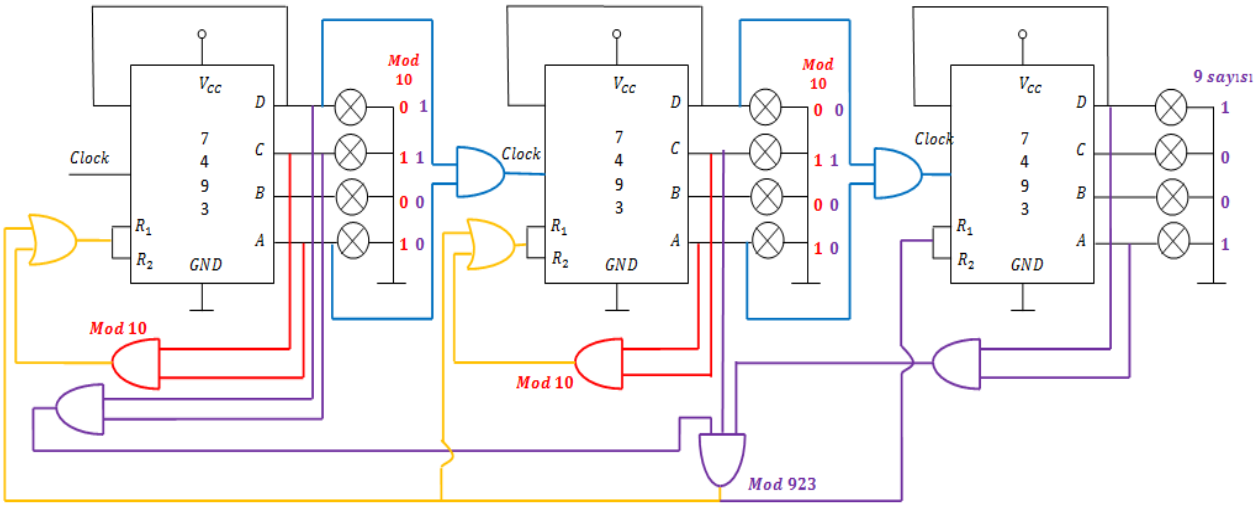
Örnek 11.15. Mod 56 sayıcıyı 7493 entegresi kullanarak tasarlayınız.



Örnek 11.16. Mod 77 sayıcıyı 7493 entegresi kullanarak tasarlayınız.



Örnek 11.17. Mod 923 sayıcıyı 7493 entegresi kullanarak tasarlayınız.



Örnek 11.18. Mod 250 sayıcıyı 7493 entegresi kullanarak tasarlayınız.