

TC.

KARAMANOĞLU MEHMETBEY ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

**MANTIK DEVRELERİ**

Hazırlayan

Beyza KALCI

Bilgisayar Mühendisliği

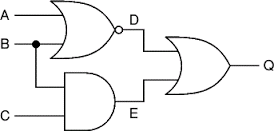
Ara Sınav Tezi

**İÇİNDEKİLER**

* 1. Mantık Devreleri
* 2. Sayı Sistemleri
* 3. Kodlar Ve Kodlamalar
* 4. Lojik Devreler
* 5. Boolean Kuralları Ve Demorgan Teoremi
* 6. Birleşik Mantık Devrelerinin Tasarımı
* 7. Bileşik Mantık Devreleri
* 8. Multiplexer Ve Demultiplexer
* 9. Decoder Ve Encoder
* 10.Karşılaştırıcı ve Aritmetik İşlem Devreleri
* 10.1 Karşılaştırıcılar
* 10.2 Aritmetik İşlem Devreleri
* 10.2.1 Toplayıcı Devreleri
* 10.2.1.1 Yarım Toplayıcı
* 10.2.1.2 Tam Toplayıcı (Full Adder - FA)
* 10.2.2 Çıkarıcı Devreleri
* 10.2.2.1 Yarım Çıkarıcı Devresi
* 10.2.2.2 Tam Çıkarıcı Devresi
* 11. Flip Flop
* 11.1 Flip – Flop Devresi Nasıl Çalışır ?
* 11.2 Flip Flop Türleri
* 11.2.1. RS Flip-Flop Devresi
* 11.2.2 Tetiklemeli - Saatli R- S Tipi Flip- Flop
* 11.2.3 J-K Flip-Flop (JK FF)
* 11.2.4 D Tipi Flip-Flop (D FF)
* 11.2.5 Ana Uydu (Master-Slave) Flip-Flop
* 12. Hafıza Devreleri
* 12.1 Belleklerde Kullanılan Terimler Ve Yapılan Temel İşlemler
* 12.2. Bellekleri Sınıflandırılması – Bellek Çeşitleri
* 12.2.1 Manyetik Bellekler
* 12.2.2 Optik Disk Bellekler
* 12.2.3 Yarı İletken Bellekler
* 12.2.3.1 Yalnızca Okunabilen Bellekler (ROM’lar)
* 12.2.3.2 Yarıiletken Oku / Yaz Bellekler (RWM’lar)
* KAYNAKÇA

1. **Mantık Devreleri**

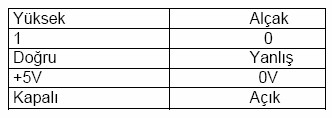
Mantık devreleri kapılardan oluşan ve 1 ve 0 rakamlarını kullanarak işlemlerin gerçekleştirildiği devrelerden meydana gelen yapılardır. Bir diğer adı **lojik devre**dir. Genel gösterimi şu şekildedir;

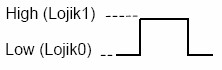


* 1. Mantık Devrelerinin Çalışma Prensibi

Mantık devreleri iki gerilim seviyesine göre çalışırlar. Her mantık devresinin bu iki gerilim seviyesine karşılık gelen değerler vardır. Bu yüzden mantık devreleri ikilik (binary) sistemde kullanılan 1 ve 0 ile tanımlanmak zorundadırlar. Bu sayılar sistemin girdilerinin ikilik koda dönüşmelerini sağlar.

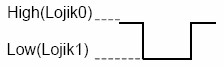
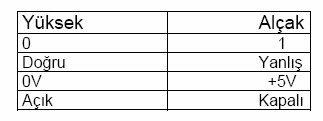
Pozitif Mantık





Pozitif mantık sayısal sinyal

Negatif Mantık



Negatif mantık sayısal sinyal

1. **Sayı Sistemleri**

Sayısal elektronikte dört çeşit sayı sistemi kullanılmaktadır;

# İkilik (binary) Sayı Sistemi

# Onlu (desimal) Sayı Sistemi

# Sekizli (oktal) Sayı Sistemi

# On altılı (hexadesimal) Sayı Sistemi

* 1. İkilik Sayı Sistemi

İkilik sayı sisteminde iki adet sayı bulunur. Bunlar 0 ve 1 dir. İkilik sayı sisteminin tabanı 2’dir. İkilik sayı sisteminde elektronik devreler ve mikroişlemciler üzerinde bir gerilim değeri olan 5 volt “1”, voltajın olmamasını ya da bir başka deyişle 0 volt da “0” olarak kabul edilir. Her sayı dijit olarak ifade edilir ve basamaklar 2'nin kuvveti olarak yazılır. Örneğin 4 dijitten (haneden) oluşan (1011 )2 gibi 4-bitlik bir sayının bit ağırlıkları 2³,2²,2¹,2º 'dır.

* Bir ikili sistem sayısında her bir rakama **"bit"** adı verilir.
* 4 bit'den oluşan sayılara **"nibble"** adı verilir.
* 8 bit'den oluşan sayılara **"byte"** adı verilir.
* 2 byte'den oluşan sayılara **"word"** adı verilir (16 bit).
* 2 word'den oluşan sayılara **"double word"** adı verilir (32 bit).
* Bir sayının en sağında yer alan bit'e **"alçak bit"** (Low bit) adı verilir.
* Bir sayının en solunda yer alan bit'e **"yüksek bit"(İşaret biti)** (High bit) adı verilir.
  1. Onluk Sayı Sistemi
     1. Onlu Tam Sayılar

Onlu sayı sistemi günlük hayatta kullandığımız 0,1,2,3,4,5,6,7,8,9 rakamlarından oluşur. Her sayı bulunduğu basamağa göre değer alır.

**Örnek 1**

128 sayısı;

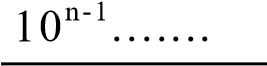
128 = 1x102 + 2x101 + 8x10°

128 = 1x100 +2x10 + 8x1

128 = 100 + 20 + 8 şeklinde tanımlanabilir.

**n. basamak.. ...4. basamak 3. basamak 2. basamak 1. basamak**

**Üstel değer** 10n-1 10 3 10 2 101 10°

**Değer** 100 10 1

**Örnek 2**

2784 sayısını;

2784= 2x103+7x102+8x101+4x10°

2784=2x1000+3x100+8x10+4x1

2784=2000+700+80+4

2784=2784 şeklinde tanımlanabilir.

2.2.2 Ondalıklı Sayılar

Verilen sayı ondalıklı ise bu durumda aynı işlem yapılır fakat ondalıklı kısımda üs kısmı negatif değerlerle ifade edilir.

**Örnek 3**

568,25 sayısı

568,25=5x102+6x101+8x10°+2x10-1 +5x10-2

568,25=5x100+6x10+8x1+2x(1/10) +5x(1/100)

568,25=500+60+8+0,2+0,05 568,25=568,25

şeklinde tanımlanabilir.

* + 1. **İkilik Sayıları Onluk Sayı Sistemine Çevirme**

İkilik sayıları onluk sayı sistemine çevirirken her basamak kendi basamak değeri ile çarpılır ve çarpım sonuçları toplanır.

**Örnek 4**

(1010)2=(?)10

|  |  |
| --- | --- |
| (1010)2 | = 1 x23 + 0x22 + 1 x21+ 0x2° |
| (1010)2 | = 1 x8 + 0x4 + 1 x2 + 0x1 |
| (1010)2 | = 8 + 0 + 2 + 0 |
| (1010)2 | = 10 |

**Örnek 5**

(11001)2 = (?)10

(11001)2 = 1x 24+1x 23+0x 22+0x 21+1x 2°

(11001)2=16 + 8 + 0 + 0 + 1

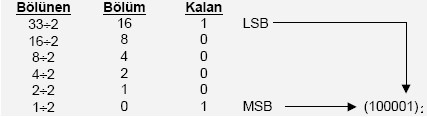
(11001)2 = 25

**2.2.4 Onlu Sayı Sisteminden İkilik Sayı Sistemine Çevirme**

Onlu sayıları ikilik sayılara çevirirken sayı 2 nin kuvvetlerine bölünür ve çıkan sonuç tersinden yazılır.

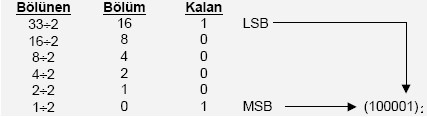
**Örnek 6**

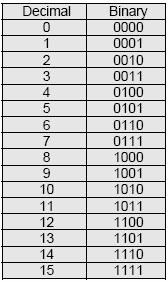
(33)₁₀=(?)₂



**Örnek 7**

(172)₁₀=(?)₂



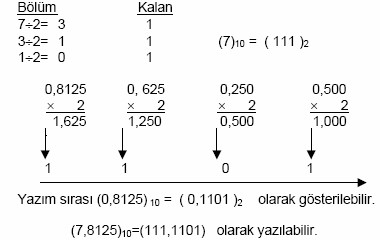


**2.2.5 Ondalıklı Sayıları İkilik Sayı Sistemine Çevirme**

Ondalıklı Sayıların ikilik sistemdeki karşılıkları bulunurken ondalıklı kısma kadar olan bölüm için normal çevirim yöntemi uygulanır. Ondalıklı kısım, kesirli kısmın sıfıra veya sıfıra yakın bir değere ulaşıncaya kadar 2 ile çarpılır.

**Örnek 8**

(7,8125)10 =(?)2



## Sekizlik Sayı Sistemi

Sekizlik sayı sistemi, ikili sayıları temsil etmek için kullanılır. Sekizli sayılar 0’dan 7’ye kadar olan rakamları kullanır ve bu nedenle 8 tabanına göre sayı sistemi denir.

## **n.basamak 4.basamak 3.basamak 2.basamak 1.basamak**

**Üstel değer** 8n-1 83 82 8 1 8°

**Değer**8 n-1 512 64 8 1

Verilen bir sayıyı 8’li karşılığına çevirmek için;

* Sayının en sağından başlayarak sayıyı 3 bitlik gruplara ayırınız.
* Her 3 bitlik grubu tablo 3’ten yararlanarak 8’li karşılığı ile değiştiriniz.

**Örnek 9**

4753 sayısını sekizlik sisteme çevirin.

4753 = 4 . 8**3** + 7 . 8**2** + 5 . 8**1** + 3 . 8**0**

= 4 . 512 + 7 . 64 + 5 . 8 + 3 . 1

= 2048 + 448 + 40 + 3 = 2539

Her basamakta yer alan rakam 8 (mavi renkle gösterilen) sayısının katları (kırmızı renkle gösterilen değerde) ile çarpılır. En sağda yer alan 8 sayısının katı 0 ile başlar ve sol tarafa doğru birer birer artar. Elde edilen değerler toplanarak sayı elde edilir.

2.3.1 Sekizlik Sayı Sistemini İkilik Sayı Sistemine Çevirme

Sekizlik sayıyı ikilik sayıya çevirmek için sekizlik sayının her basamağı  3 bitlik ikilik sayıya çevrilir.  
  
**Örnek 10**  
  
(432)₁₀= ( ? )₂   
  
(702) ₈ sayısının her basamağının ikilik karşılığı 3 bitlik yazılırsa;  
  
  7        0       2  
111   000   010  
  
(702) ₈ = ( 111000010 )₂ bulunur.

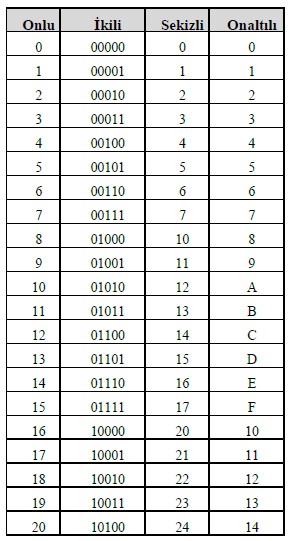
**Örnek 11**  
  
(456)₈ = ( ? )₂   
  
(456)₈ sayısının her basamağının ikilik karşılığı 3 bitlik yazılırsa;  
  
  4        5       6  
100   101   110  
  
(456)₈= ( 100101110 )₂ bulunur.

**2.3.2 İkilik Sayı Sistemini Sekizlik Sayı Sistemine Çevirme**  
  
İkilik sayıyı sekizlik sayıya çevirmek için ikilik sayı sağ taraftan itibaren 3’er 3’er gruplara ayrılır. Her grubun sekizlik karşılığı yazılır.  
  
**Örnek 12**  
  
(011101) ₂ = ( ? ) ₈  
  
(011101) ₂  Sağ taraftan 3’erli gruplara ayırırsak;  
  
011   101  
  3       5

(011101) ₂ = (35) ₈ bulunur.  
  
**Örnek 13**  
  
(11010111) ₂  = ( ? ) ₈  
  
(11010111) ₂  Sağ taraftan 3’er 3’er gruplara ayırırsak;  
  
11   010    111  
 3      2        7  
  
(11010111) ₂ = (327) ₈ bulunur.

* 1. Onaltılık Sayı Sistemi

Onaltılık sayılar 0’dan başlayarak 15’e kadar giden toplam 16 farklı ifadeden oluşan bir sayı sistemidir. Her 4-bit ile 0’dan 15’e kadar sayı değeri ifade edilebildiğinden ve 10’dan 15’e kadar da onluk sistemde iki haneli kullanım söz konusu olduğundan tek haneli(karakterli) gösterim kullanılmıştır. Buna göre onaltılık sayılar; 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F şeklinde gösterilir. Görüldüğü gibi 9’dan sonra 10 yerine A, 11 yerine B, 12 yerine C, 13 yerine D, 14 yerine E ve 15 yerine de F harfi kullanılmaktadır.



2.4.1 Onaltılık Sayı Sistemini İkilik Sayı Sistemine Çevirme

Onaltılık sayıyı ikilik sayıya çevirme işlemi yapılırken onaltılık sayının her basamağındaki sayının ikilik karşılığı 4 basamaklı olarak yazılır.  
  
Böylece onaltılık sayının ikilik karşılığı bulunur.

**Örnek 14**

(9FB)₁₆= ( ? )₂ 

(9FB)₁₆  
  
   9          F         B   
1001   1111   1011  
  
(9FB)₁₆ = (100111111011)₂ olur.

**Örnek 15**

(F16)₁₆ = ( ? )₂

(F16)₁₆  
  
    F           1          6  
1111     0001   0110  
  
(F16)₁₆  = (111100010110)₂  bulunur.

2.4.2 İkilik Sayı Sistemini Onaltılık Sayı Sistemine Çevirme  
  
İkilik sayıyı on altılık sayı sistemine çevirmek için verilen ikili sayı sağdan başlamak üzere 4’er 4’er gruplara ayrılır.  
  
Ayrılan her grubun on altılık karşılığı yazılır.  
  
  
**Örnek 16**

(01111101)₂ = ( ? )₁₆

4’erli gruplara ayırırsak;  
  
 0111   1101  
   7          D  
  
(01111101)₂ = (7D)₁₆ bulunur.

**Örnek 17**

(101001011110)₂ = ( ? )₁₆

4’erli gruplara ayırırsak;  
  
1010   0101   1110  
   A         5          E  
  
(101001011110)₂ = (A5E)₁₆ bulunur.

2.4. Onaltılık Sayı Sistemini Onluk Sayı Sistemine Çevirme

On altılık sayıyı onluk sayıya çevrilirken onaltılık sayının sağdan itibaren ilgili basamağı 16'nın 0, 1, 2, 3, ... üsleri ile çarpılır ve sonra toplanır.  
  
Pratik olarak onaltılık sayının basamakları sağdan itibaren;

1. basamağı 1 ile,  
  
2. basamağı 16 ile,  
  
3. basamağı 256 ile,  
  
4. basamağı 4096 ile çarpılır ve çıkan sonuçlar toplanır.  
  
Çıkan sonuç onaltılık sayının onluk karşılığını verir.  
  
  
**Örnek 18**

(5C8)₁₆ = ( ? )₁₀

(5C8)₁₆ = 8.1 + 12.16 + 5.256  
  
(5C8)₁₆ = 8 + 192 + 1280  
  
(5C8)₁₆ = ( 1480 )₁₀ olarak hesaplanır.

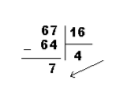
Onaltılık sayılarla hesap yapılırken harf olarak belirtilen sayıların rakama çevrilerek hesap yapılması daha kolay olacaktır.   
  
  
**Örnek 19**  
  
(A19)₁₆ = ( ? )₁₀

(A19)₁₆ = 9.1 + 1.16 + 10.256  
  
(A19)₁₆ = 9 + 16 + 2560  
  
(A19)₁₆= ( 2585 )₁₀ olarak hesaplanır.

* + 1. Onluk Sayı Sistemini Onaltılık Sayı Sistemine Çevirme

Onluk sayıyı, onaltılık sayıya çevirmek için onluk sayı 16’ya bölünür.  
  
Bölme sonunda kalanlar tersten yazılır.

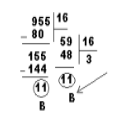
**Örnek 20**  
  
(67)₁₀ = ( ? )₁₆

[](https://4.bp.blogspot.com/-8YJguOErBH4/WvGmks1AcFI/AAAAAAAABr8/Z-Z1NQMvNZgWsnINLQRp2NFVYU49r2VAACLcBGAs/s1600/DeepinEkranG%25C3%25B6r%25C3%25BCnt%25C3%25BCs%25C3%25BC_20180508162942.png)

(67)₁₀ = ( 47 )₁₆    olarak hesaplanır.

**Örnek 21**

(955)₁₀ = ( ? )₁₆

[](https://4.bp.blogspot.com/-QtoQkOTDpbc/WvGnuaB4DzI/AAAAAAAABsM/Jm_6zwwvKg0Snksm2cPA2gEIipD7G0fjQCLcBGAs/s1600/DeepinEkranG%25C3%25B6r%25C3%25BCnt%25C3%25BCs%25C3%25BC_20180508163220.png)

( B=11’dir)  
  
(955)₁₀ = (3BB)₁₆      olarak hesaplanır.

1. Kodlar Ve Kodlamalar

Dijital elektronikte yapılan işlemleri kolaylaştırmak ve hata oranlarını azaltmak  
amacıyla kodlar kullanılır.  
Kodlama, iki küme elemanları arasında karşılığı kesin olarak belirtilen kurallar  
bütünüdür. Diğer bir şekilde ifade edilirse, görünebilen, okunabilen, yazı, sayı ve işaretlerin  
değiştirilmesi işlemine “kodlama” denir. ‘Morse’ alfabesi kodlamaya iyi bir örnektir.

Kodlama işlemi yalnızca onluk sistemdeki sayıları içerebileceği gibi, alfabetik ve alfa sayısal bilgilerin kodlanmasını içerebilir. Farklı bilgileri kodlama ihtiyacı ve değişik alanlarda kodlama gereksinimi çeşitli kodlama yöntemlerini doğurmuştur.

Kodlama işlemi aşağıdaki avantajları sağlar;

* Aritmetik işlemlerde kolaylık sağlar.
* Hataların bulunmasını kolaylaştırır.
* Hataların düzeltilmesi işlemlerini basitleştirir.
* Bellek işlemlerinde verimliliği artırır.
* Bilgilerin işlenmesi işleminin insanlarca kolayca anlaşılmasını sağlar.

Yalnızca sayısal karakterlerin kodlanmasıyla ortaya çıkan kodlara **‘sayısal kodlar’** (BCD kodları) denilirken, alfabetik ve sayısal karakterlerin kodlanmasını içeren kodlama yöntemlerine **‘alfa sayısal kodlar’** denir.

* 1. Sayısal (BCD) Kodlar

Yalnızca sayısal karakterlerin kullanıldığı sayısal kodlama sistemlerinin çok geniş uygulama alanı olması nedeni ile, çok farklı sayısal kodlama yöntemleri kullanılmaktadır. Sayısal kodlama yöntemlerine örnek olarak;

* BCD kodu
* Bar kodu
* +3 kodu
* Aiken kodu
* 5’te 2 kodu
* Gray kodu
* Eşitlik (Parity) kodu

kodlama yöntemleri verilebilir.

* + 1. BCD Kodu (8421)

Onluk sistemdeki bir sayının, her bir basamağının ikilik sayı sistemindeki karşılığının dört bit şeklinde yazılması ile ortaya çıkan kodlama yöntemine denir. Onluk sayı sistemi 0 ile 9 arasındaki sayıları içerdiğinden, her basamaktaki sayının ikili sistemde kodlanması için 4 bite ihtiyaç vardır. Onluk bir sayıyı BCD kodu olarak yazmak için, onluk sayının her bir basamağı 4 bitlik ikili sayı grupları şeklinde yazılır. Yazılan gruplar bir araya getirilince BCD kodlu sayı elde edilir.

**BCD (8421)**

**0 0000**

**1 0001**

**2 0010**

**3 0011**

**4 0100**

**5 0101**

**6 0110**

**7 0111**

**8 1000**

**9 1001**

İki veya daha fazla basamaktan oluşan onluk sayılar için tek basamaklı

onluk sayıların ikilik kodları yan yana konur.

**Örnek 22**

(23)₁₀ =>(2)₁₀ = (0010)BCD ve (3)₁₀ = (0011) BCD

=> (23)₁₀ = (0010 0011) BCD

Bu kodlamanın en yüksek basamak değeri (23)₈ , üçüncü basamak (22)₄ , ikinci

basamak (21)₂ ve en düşük basamak ağırlığı (20)₁ olarak belirlenmiştir ve bu kodlama her

bir onluk sistemdeki sayının dört bitlik karşılığı yazılarak tamamlanır.

**Örnek 23**

(49)₁₀ = ( ? ) BCD

Dönüştürme işlemi onluk sistemdeki her bir rakamın dört bitlik BCD

karşılığı yazılarak bulunur;

4 9

(0100) (1001)

(49)₁₀ = ( 0100 1001 ) BCD

**Örnek 24**

(0001 1001 0010) BCD = ( ? )₁₀

Dönüştürme işlemi her bir dört bitlik BCD rakamın onluk sistemdeki

karşılığı yazılarak bulunur;

1 9 2

0001 1001 0010

(0001 1001 0010) BCD = ( 192 )₁₀

**Örnek 25**

(263)10 = ( ? ) BCD

Her bir basamaktaki sayının ikili karşılığı 4 bit olarak yazılırsa;

2 6 3

0010 0110 0011

(263)10 = (001001100011)BCD

**Örnek 26**

(1001 0011 0110)BCD = ( ? )₁₀

(1001 0011 0110)BCD

9 3 6

(100100110110)BCD = (936)10

**Örnek 27**

(10100110)2 =( ? ) BCD

2+4+32+128=166

(10100110)2= (166)10

BCD koduna çevirirsek;

(000101100110)𝐵𝐶𝐷

**Örnek 28**

(1010110101)2 =( ? ) BCD

1+4+16+32+128+512=693

(1010110101)2= (693)10

BCD koduna çevirirsek;

(011010010011)𝐵𝐶𝐷

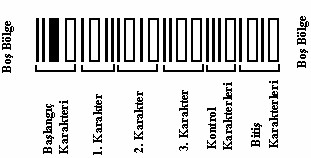
#### Bar (Çubuk) Kodu

Onlu sayıların farklı şekilde düzenlenmiş çubuklarla ifade edildiği kodlama sistemi **bar kod** olarak isimlendirilir. Kare veya dikdörtgen biçimlerde basılabilir ve 2 boyutludur. Kare veya dikdörtgen şeklinde olan bu yapının genel adı ise Data Matrixtir.

Diğer bir deyişle, karakterlerin (rakam veya harf) farklı kalınlıktaki çizgiler ve boşluklar ile temsil edildiği kodlama sistemi barkod olarak adlandırılır.

Klavyeye alternatif olarak kullanılan bar kodu yöntemi, veri giriş / çıkışının kolay olması nedeniyle özellikle stoklama işlemlerinde ve marketlerde yaygın olarak kullanılmaktadır.

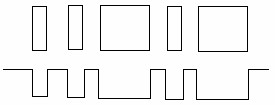
Karakterleri temsil etmek için kullanılan çizgilerin uygun araçlarla okunup çözümlenmesi ve bilgisayara aktarılması için çeşitli barkod yöntemleri ve barkodları okuyacak farklı teknolojiler bulunmaktadır.



Şekilde gösterilen kombinasyon 39 bar koduna bir örnektir. Bu örnekte toplam 9 çubuk / aralık bulunmaktadır. Bunlardan 1. ve 8. sıralardaki çubuklar ile 3. sıradaki aralık lojik ‘1’ değerini, diğer çubuk ve aralıklar ‘0’ değerlerini temsil etmektedir.

1 0 1 0 0 0 0 1 0

Barkodlu bilgiler barkod okuyucu ile okunur. Bir barkod tarandığında, kodda bulunan çizgiler ve boşluklar elektrik sinyallerine dönüştürülürler. Boşlukların (veya çizgilerin) elektriksel olarak 0V’a, siyah çizgilerin 1V’a karşılık gelir. Elde edilen sinyalin bilgisayar tarafından anlaşılabilecek sayısal değerlere dönüştürülmesi gerekir. Bu dönüşüm işlemi kuralları önceden belirlenen bir kod çözücü ile yapılır.



**Barkod**

**Sayısal İşaret**

Barkodun sahip olduğu değerler, genellikle barkodun altına yazılır ve barkodun herhangi bir nedenle okunamaması durumunda barkodun değerinin el ile bilgisayara girilebilmesine imkan tanımak için kullanılır.



Karekodlar da barkodlara bir örnektir. Karekod olarak isimlendirilen kod, 5’te 2 kodunun çubuklarla ifade edilen şeklidir. Bu kodda bilgiler yalnızca çubuklarla ifade edilir, aralıklar bir anlam içermez. Her bilgi 5 çubuk ile oluşturulur ve bunlardan yalnızca 2 tanesi geniştir. İnce çubuklar '0', kalın çubuklar '1' anlamına gelir.



**Örnek 29**

Onlu sistemdeki ‘0’ ve ‘6’ değerlerini 25 bar kodu ile gösterelim.

dizisi '0' değerini temsil ederken ,

dizisi onlu '6' değerini temsil eder.

* + 1. **+3 Kodu**

Artı 3 kodu (+3 Code), BCD kodu ile ilgilidir ve belirli aritmetik işlemlerde işlem kolaylığı nedeniyle BCD kodu yerine kullanılır. Bir onlu sayının Artı 3 kodundaki karşılığı, onlu sayının karşılığı olan ikili sayıya 3 eklenmiş halidir. Bu nedenle bu kodlama yöntemi, ‘3 fazlalık kodu’ olarak ta isimlendirilir. Artı 3 kodundaki sayılar, BCD kodunda olduğu gibi dört bitlik ikili sayılar şeklinde ifade edilir.

Hesaplama yapmada ve hataları düzeltmelerde sağladığı kolaylıklara rağmen, tümleyenini almadaki güçlükler nedeniyle son zamanlarda nadiren kullanılmaktadır.

**Örnek 30**

(48)10 =(?)+3

+ 3 + 3 her bir basamağa 3 eklenir,

7 11 bulunan sonuç 4 bitlik ikili sayıya çevrilir.

0111 1011

Bulunan sayılar yan yana yazılarak Artı 3 kodlu sayı elde edilir. Buna göre;

(48)10=(01111011)+3

**Örnek 31**

(11000110)+3 = (?)10

Sayı dörder bitlik gruplara ayrılır ve her bir grubun karşılığı olan onlu sayı bulunur.

(11000110)+3 1100 0110 = (12 6)+3

Bulunan her bir sayıdan 3 çıkarılırsa;

12 6

- 3 - 3

9 3

(93)10 sayısı bulunur.

(10100110)+3=(93)10

**Örnek 32**

(59)₁₀ = ( ? ) +3

Aynı soru (59)₁₀ = ( ? )Xs-3 code olarakta gösterilebilir.

Dönüştürme işlemi her bir onluk sistemdeki rakamın dört bitlik BCD karşılığı yazıldıktan ve her bir sayıya 3 (0011) ilave edildikten sonra bulunabileceği gibi, onluk sistemdeki her bir sayıya üç eklenip en son yeni elde edilen her bir sayıyı BCD’ye çevirerek de yapılabilir.

(59)= 0101 1001

+ 0011 0011

1000 1100

(59)₁₀ = ( 1000 1100 ) +3

**Örnek 33**

(1386)₁₀ = ( ? ) +3

Dönüştürme işlemi her bir onluk rakamın dört bitlik BCD karşılığı yazılır ve her bir basamağa 3 (0011) ilave edilir;

(1386) = 0001 0011 1000 0110

+ 0011 0011 0011 0011

0100 0110 1011 1001

(1386)₁₀ = ( 0100 0110 1011 1001) +3

* + 1. **Aiken Kodu**

Aiken kodu 4 basamaklı ve basamak değerlerinin ‘2421’ şeklinde ifade edildiği bir kodlama şeklidir. Onlu sistemde 5’e kadar olan sayıları kodlamak için sağ taraftaki basamaklar kullanılırken, 5’den büyük değerleri ifade etmek için sol taraftaki bitler kullanılır.

Bu kodlama şekli simetrik kodlamaya bir örnektir. (0-4) arasındaki sayılar için normal ikili sayılar kullanılırken, (5-9) arasındaki sayılar için başlangıçtaki sayıların simetriği kullanılır.

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

**Örnek 34**

(3)10 ve (7)10 sayılarını Aiken Koduna göre kodlayınız.

(3)10 sayısı ‘2421’ basamak değerleri göz önünde bulundurularak yazılırsa;

(3) = (0011)Aiken

(7)10 sayısı basamak değerleri göz önünde bulundurularak yazılırsa;

(7)10 = (1101)Aiken

* + 1. **5’te 2 Kodu**

5’de 2 kodunda, her onlu sayı, içinde mutlaka iki tane '1' bulunan 5 bitlik ikili sayı ile temsil edilir. Bütün sayılarda mutlaka iki tane '1' bulunduğundan hataların kolayca bulunmasını sağlar. Sayılar ikili sistemde ifade edilirken basamak değerleri '7 4 2 1 0' şeklinde sıralanır. (0)10 sayısını 5’te 2 kodunda ifade etmek için (11000) kombinasyonu kullanılır.

**Örnek 35**

(6)10 = (?)5’de 2

Basamak değerleri 74210 olduğundan ve mutlaka 2 tane 1 bulunması gerektiğinden;

(6)10 = (01100)5’de 2

**Örnek 36**

(01010 10100)5’te 2 =(?)10

Her bir basamaktaki sayı 5 bit ile ifade edildiğinden, sayı 5 bitlik gruplara ayrılıp her bir grubun karşılığı olan onlu sayı yazılırsa;

(01010 10100)

|  |  |
| --- | --- |
|  | **5’te 2 Kodlu** |
| **Onluk Sayı** | **Sayı** |
| **7 4 2 1 0** |
| 0 | 1 1 0 0 0 |
| 1 | 0 0 0 1 1 |
| 2 | 0 0 1 0 1 |
| 3 | 0 0 1 1 0 |
| 4 | 0 1 0 0 1 |
| 5 | 0 1 0 1 0 |
| 6 | 0 1 1 0 0 |
| 7 | 1 0 0 0 1 |
| 8 | 1 0 0 1 0 |
| 9 | 1 0 1 0 0 |

5 9

(0101010100)5’te 2 = (59)10

* + 1. **Gray Kodu**

Geçişler sırasında hatayı minimuma indirmek için geliştirilmiş bir koddur. İsmini mucidi Frank Gray'den alır. Gray kodlama yöntemi, basamak ağırlığı olmayan bir kodlama yöntemidir. Basamak ağırlığının olmaması, her bir basamaktaki sayıların basamak ağırlıklarına göre karşılıklarının olmamasıdır. Özelliği, Art arda iki sayı arasında sadece tek bit değişikliğidir.

Normal ikilik kodunda ardışık sayılarda çoğu kez birden fazla bit değişikliği söz konusudur. Devrelerde 0-1 durumları arasındaki değişim sırasında okuyucu farklı değerler okuyabilir.

Gray kodunda sadece tek bit değiştiğinden, ve ikilik sayı olduğundan, gray kodunda bu hata ortadan kaldırılmıştır.

Sayısal elektronik ve bilgisayar giriş-çıkış işlemlerinde kullanılan gray kodlama yöntemi, minimum değişimli kodlar sınıfı içerisinde yer alır. Bunun nedeni bir sayıdan diğerine geçerken yalnızca bir bitin konum değiştirmesidir.

**Örneğin; yalın ikilik kodlamada (3)10 = (0011)2  değerinden (4)10 = (0100)2  değerine geçerken üç bitin değeri değişirken, gray kodlamada yalnızca bir bitin değeri değişir.**

Gray kodlanmış sayılarda basamak değeri olmadığından, bu kodlama yönteminin aritmetik işlemlerin olduğu yerlerde kullanılması mümkün değildir. Ancak sütun esasına göre çalışan cihazlardaki hatayı azalttığından, giriş / çıkış birimlerinde ve analog - dijital çeviricilerde tercih edilirler. Yalnızca, 9’dan 0’a geçişte çok sayıda bit konum değiştirir.

Onluk sayıların karşılığı olan ikili sayıları Gray kodlanmış olarak ifade etmek için, bir sayıdan diğerine geçişte tek bir bitin değer değiştirmesi esas alınır.

**İkilik Sayıların Gray Koduna Çevrilmesi 1. Yöntem;**

* İkilik olarak verilen ilk bit aşağıya indirilir.
* İlk bit ile ikinci bitin toplamı aşağıdaki bitin sağ tarafına yazılır.
* İkinci bit ile üçüncü bitin toplamı aşağıdaki diğer bitlerin sağına yazılır.
* Bitler bitene kadar iki bitin toplamı sağ bitin altına gelecek şekilde işleme
* devam edilir.
* “Toplama işleminde 1+1 =0 olmalıdır”.

**İkilik Sayıların Gray Koduna Çevrilmesi 2. Yöntem;**

* İkili sistemde verilen ikilik sayının en yüksek öncelikli bitinin (MSB) önüne (en soluna) “0” sıfır konur.
* En düşük öncelikli bitten (LSB) başlayarak her bit sol yanındaki bit ile kıyaslanmaya başlar.
* Kıyaslanan iki bit birbirine eşit ise (her ikisi “1” ve ya “0”) gray kod hanesi “0” sıfır yazılır.
* Kıyaslanan iki bit birbirine eşit değil ise (biri “1” diğeri “0”) gray kod hanesi “1” bir yazılır.

|  |  |  |
| --- | --- | --- |
| **Onluk Değer** | **İkilik Değer** | **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** |

**Örnek 37**

(101110101)2 =(?) 𝑔𝑟𝑎𝑦

* + - 1. 1 0 1 1 1 0 1 0 1 İkilik Sayı

Başlama biti

* + - 1. 1 1 0 0 1 1 1 1 Gray kodlu sayı

(101110101)2 = (111001111) 𝑔𝑟𝑎𝑦

**Örnek 38**

(1000101)2 =(?) 𝑔𝑟𝑎𝑦

* + 1. 1 0 0 0 1 0 1 Binary Sayı

* + 1. 1 0 0 1 1 1 Gray kodlu sayı

(1000101)2 = (1100111) 𝑔𝑟𝑎𝑦

**Gray Kodunun İkilik Koda Çevrilmesi;**

* Gray kodlu ifadedeki ilk bit aşağı indirilir.
* İkinci bit ile aşağıya indirilen ilk bitin toplamı aşağıya indirilen bitin yanına yazılır.
* Üçüncü bit, aşağıya indirilen ikinci bitle toplanır ve ikinci bitin yanına yazılır.
* Gray kodlu bitler bitene kadar işleme devam edilir.

**Örnek 39**

(111001111)GRAY =(?)2

1 1 1 0 0 1 1 1 1 Gray kodlu sayı

1 0 1 1 1 0 1 0 1 İkilik sayı

(111001111)GRAY = (101110101)2

**Örnek 40**

(1100111)GRAY =(?)2

1 1 0 0 1 1 1 Gray kodlu sayı

1 0 0 0 1 0 1 İkilik Sayı

**Örnek 41**

(01011101)𝑔𝑟𝑎𝑦 =(?)2

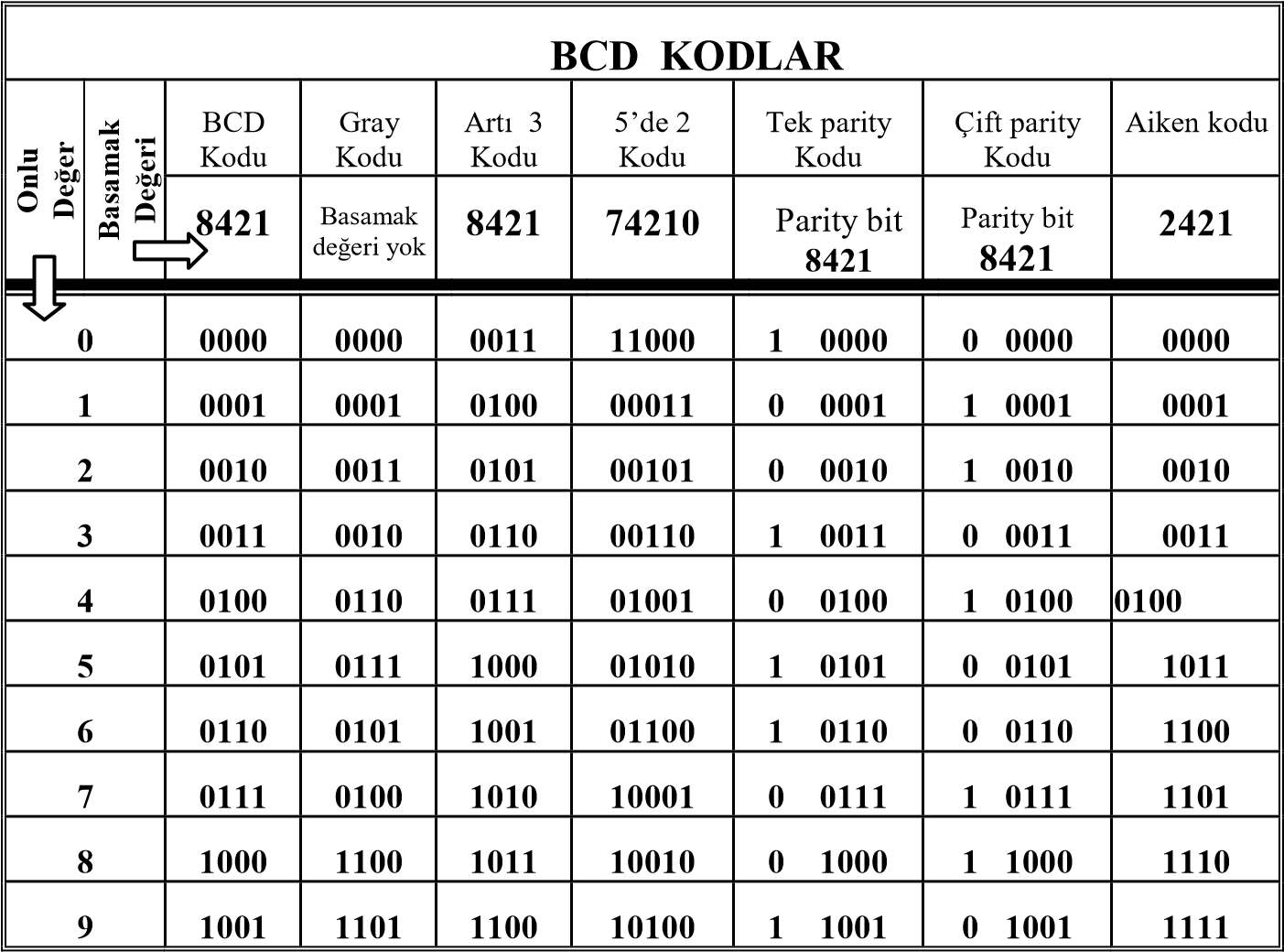
(01011101)𝑔𝑟𝑎𝑦 =(01101001)2

* İlk sayı aynen aşağı alınır,
* Sağındaki diğer sayılar
* yukarıdaki sayı 1 olduğunda rakamın değiştiğine,
* 0 olduğunda değişmediği kabul edilerek değerlendirilir.

**Örnek 42**

(10101100111)𝑔𝑟𝑎𝑦=(?)2

(10101100111)𝑔𝑟𝑎𝑦 =(11001000101)2



**3.1.7 Eşitlik (Parity) Kodu**

İkilik sayı sistemindeki bilginin bir yerden başka bir yere taşınması dijital sistemlerde sıkça karşılaşılan bir olaydır. Bilginin bir yerden başka bölgeye taşınması sırasında, değişik nedenlerden dolayı gürültü oluşması ve oluşan gürültünün iletilen bilgiyi bozması zaman zaman karşılaşılan sorunlardır. Bilgi iletimi sırasında bu şekilde oluşan hataları tespit etmek ve mümkünse düzeltmek sayısal sistemlerin özelliklerindendir.

Hataları tespit etmede kullanılan en yaygın ve en kolay yöntem eşitlik biti kodlama (parity code) yöntemidir. Bu yöntemde, hataların ortaya çıkarılmasını sağlamak amacıyla BCD kodlu sayının sağındaki veya solundaki basamağa **eşitlik biti** (parity bit) eklenir. Eşitlik biti, kodlanan veride 1 yada 0 ların tek mi, çift mi olduğunu belirtir. İki türlü eşitlik biti yöntemi bulunmaktadır;

* **Çift eşitlik (even parity)**
* **Tek eşitlik (odd parity).**

**Çift Eşitlik Yöntemi;**

Eşitlik bitinin değeri, kodlanacak bilgideki 1’lerin toplam sayısı (eşitlik biti dahil) çift olacak şekilde seçilir. Kodlanacak sayıdaki 1’lerin sayısı tek ise, eşitlik biti olarak ‘1’ eklenir. Kodlanacak bilgideki 1’lerin sayısı çift olması durumunda ise, eşitlik biti olarak ‘0’ eklenir.

**Örnek 43**

(1000011)2 sayısına çift eşitlik biti yöntemine göre eşitlik biti ekleyelim.

Kodlanacak bilgide (1000011) üç adet ‘1’ bulunduğundan, bilgideki 1’lerin sayısını çift yapmak için eşitlik biti olarak ‘1’ eklenir ve sonuç;

(11000011)

**Örnek 44**

(1000001)2 sayısını çift eşitlik yöntemine göre kodlayalım.

Verilen sayıda çift sayıda ‘1’ bulunduğundan, eşitlik biti olarak ‘0’ eklenir ve kodlama işlemi sonuç;

‘01000001’

**Tek Eşitlik Bit Yöntemi;**

Aynı mantığa göre düzenlenir. Tek fark kodlanan bilgideki 1’lerin sayısı tek olmalıdır.

**Örnek 45**

(1000001)2 sayısına tek eşitlik biti yöntemini uygulayalım.

‘1000001’ sayısında çift sayıda ‘1’ bulunduğundan, eşitlik biti değeri ‘1’ olur ve kodlaşmış bilgi;

‘11000001’

**Örnek 46**

(1000011)2 sayısına tek eşitlik biti ekleyelim.

Verilen sayıda tek sayıda ‘1’ bulunduğundan, eklenecek eşitlik biti ‘0’ olur ve sonuç;

‘01000011’

**Eşitlik kodunda unutulmaması gereken nokta, çift veya tek eşitlik biti yönteminde eklenen bitin bilginin bir parçası olduğudur. Normalde 7 bit olarak ifade edilen bilgiler, eşitlik bitinin eklenmesiyle 8 bitlik bilgiler haline dönüşür. Eşitlik kodlama yönteminin avantajı, bilginin iletilmesi sırasında bir bitin değerinin değişmesi ihtimali olan yerlerde hatanın alıcı tarafından kolayca tespit edilebilmesidir.**

1. Lojik Devreler

Sayısal devrelerin tasarımında kullanılan temel devre elemanlarına Lojik kapılar adı verilir. Bir lojik kapı bir çıkış, bir veya birden fazla giriş hattına sahiptir. Çıkışı, giriş hatlarının durumuna bağlı olarak Lojik-1 veya Lojik-0 olabilir. Bir Lojik kapının girişlerine uygulanan sinyale bağlı olarak çıkışının ne olacağını gösteren tabloya doğruluk tablosu (truth table) adı verilir.

* VE(AND)
* VEYA(OR)
* DEĞİL(NOT)
* VEDEĞİL(NAND)
* VEYADEĞİL(NOR)
* ÖZELVEYA(EXOR)
* ÖZELVEYA DEĞİL(EXNOR)

temel lojik kapılardır.

**4.1. Doğruluk Tabloları (Truth Table)**

Doğruluk tabloları sayısal devrelerin tasarımında ve analizinde kullanılan en basit ve faydalı yöntemdir. Doğruluk tablosu giriş değişkenlerinin alabileceği olası bütün durumlar için çıkış ifadesinin ne olduğunu gösteren tablodur.

Bir doğruluk tablosunda eğer n sayıda giriş değişkeni varsa bu değişkenler olası 2n sayıda değişik durum alabilirler.

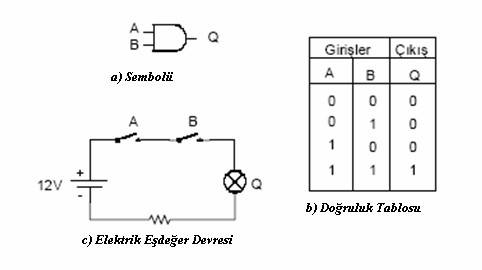
**Örneğin bir sayısal devrenin iki (n=2) giriş değişkeni varsa bu değişkenlerin alabileceği durum sayısı 22=4 iken, üç giriş değişkeni (n=3) için 23=8 farklı durum yazılabilir.**



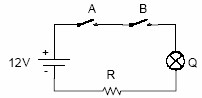
4.2 Mantık Kapıları

4.2.1 Ve Kapısı

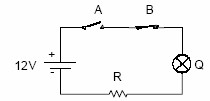
Bir çıkış, iki veya daha fazla giriş hattı bulunur. Eşdeğer devresini ise birbirine seri iki anahtar olarak düşünebiliriz. İki anahtar da kapalı olmadan yani tüm girişlerin değeri "1" olmadan, elde edilen çıkış "1" olmaz. "**Y = A . B"** ile gösterilir.



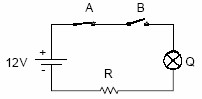
* A ve B anahtarları açık ise (A=0, B=0) lamba yanmayacaktır (Q=0).

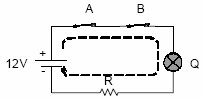


* Eğer A anahtarı açık (A=0), B anahtarı kapalı(B=1) ise, lamba yanmayacaktır. (Q=0) .



* Eğer A anahtarı kapalı (A=1),B anahtarı açık (B=0) ise, lamba yanmayacaktır **(Q=0).**



* Eğer A ve B anahtarları kapalı (A=1,B=1) ise, lamba yanacaktır (Q=1).

**Örnek 47**

Üç-girişli bir VE kapısına ait Lojik ifadeyi yazarak doğruluk tablosunu oluşturunuz.

Girişlere A,B,C dersek (n=3) oluşturulacak doğruluk tablosunda 23 = 8 farklı durumun yazılması gerekir.

|  |  |  |  |
| --- | --- | --- | --- |
|  | GİRİŞLER |  | ÇIKIŞ  Q |
| A | B | C |

0 0 0 0

0 0 1 1

1. 1 0 0

0 1 1 1

1 0 0 0

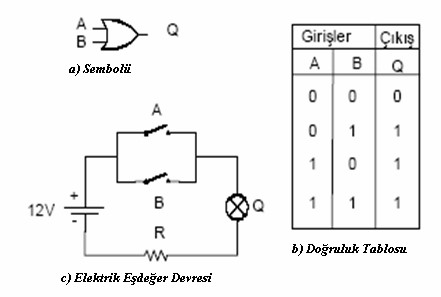
1 0 1 1

1 1 0 0

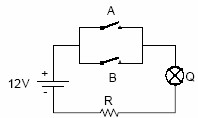
1 1 1 1

4.2.2 Veya Kapısı

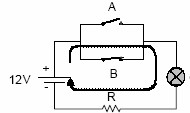
Bir çıkış, iki veya daha fazla giriş hattı bulunur. Eşdeğer devresini birbirine paralel iki anahtar olarak düşünebiliriz. Bu yüzden çıkıştan "1" elde etmek için herhangi bir giriş değerinin "1" olması yeterlidir. Tüm girişlerin "1" olmasının sonucu da yine "1" olarak kabul edilir. "**Y = A + B"** ile gösterilir.



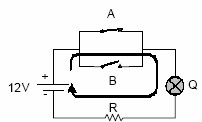
* Eğer A ve B anahtarları açık ise (A=0, B=0) lamba yanmayacaktır Q=0.



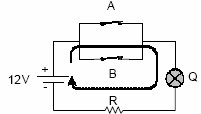
* Eğer A anahtarı açık (A=0), B anahtarı kapalı (B=1) ise, lamba yanacaktır Q=1.



* Eğer A anahtarı kapalı (A=1), B anahtarı açık (B=0) ise, lamba yanacaktır Q=1.



* Eğer A ve B anahtarları kapalı (A=1,B=1) ise, lamba yanacaktır Q=1.



**Örnek 48**

Üç-girişli bir VEYA kapısına ait Lojik ifadeyi yazarak doğruluk tablosunu oluşturunuz.

Girişlere A,B,C dersek (n=3) oluşturulacak doğruluk tablosunda 23 = 8 farklı durumun yazılması gerekir.

|  |  |  |  |
| --- | --- | --- | --- |
|  | GİRİŞLER |  | ÇIKIŞ  Q |
| A | B | C |

0 0 0 1

0 0 1 1

1. 1 0 0

0 1 1 0

1 0 0 1

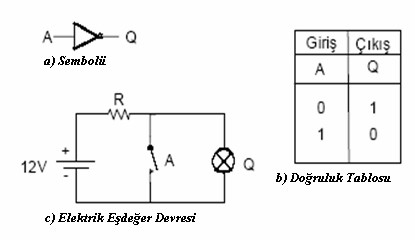
1 0 1 0

1 1 0 1

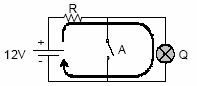
1 1 1 0

4.2.3 Değil Kapısı

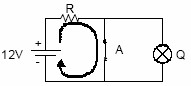
Bir çıkış, bir giriş hattı bulunur. Çıkış işareti, giriş işaretinin tam tersi (devriği) olur. "0" girişli bir devrenin çıkışı "1" olurken, "1" girişli bir devrenin çıkışı "0" olur.**" Y = A' "**ile gösterilir.



* Eğer A anahtarı açıksa (A=0) akım devresini Q lambası üzerinden tamamlayacağından lamba yanacaktır(Q=1).

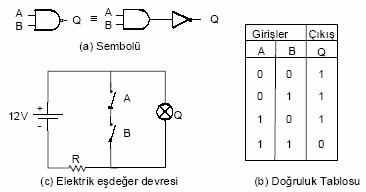


* Eğer A anahtarı kapalı ise (A=1) akım devresini A anahtarı üzerinden tamamlayacağından lamba yanmayacaktır (Q=0).

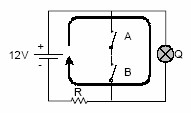


4.2.4 Ve Değil Kapısı

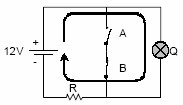
Bir çıkış, iki veya daha fazla giriş hattı bulunur. "VE" fonksiyonunun "DEĞİL" ile birleşimi olarak düşünebiliriz. Bu doğrultuda "VE" fonksiyonundan elde edilen çıkışların tam tersi elde edilir. Girişlerin hepsi "1" olursa çıkış "0" olur ancak bunun dışındaki tüm durumlarda "1" çıkışı elde edilir.  " **Y = (A . B)' "**ile gösterilir.



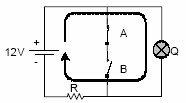
* Eğer A ve B anahtarları açık (A=0,B=0) ise akım devresini Q lambası üzerinden tamamlar lamba yanar (Q=1).



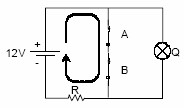
* Eğer A anahtarı açık(A=0), B anahtarı kapalı(B=1) ise akım devresini Q lambası üzerinden tamamlar lamba yanar(Q=1).



* Eğer A anahtarı kapalı(A=1), B anahtarı açık (B=0) ise akım devresini Q lambası üzerinden tamamlar lamba yanar (Q=1).

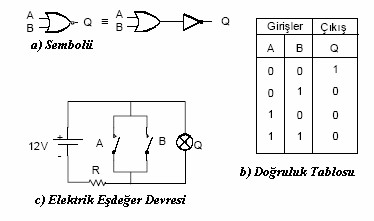


* Eğer A ve B anahtarları kapalı ise(A=1,B=1) ise akım devresini anahtar üzerinden tamamlar Q lambası yanmaz (Q=0).

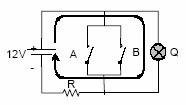


4.2.5 Veya Değil Kapısı

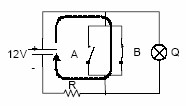
Bir çıkış, iki veya daha fazla giriş hattı bulunur. "VEYA" fonksiyonunun "DEĞİL" ile birleşimi olarak düşünebiliriz. Bu yüzden "VEYA" fonksiyonundan elde edilen çıkışların tam tersi elde edilir. Girişlerin hepsi "0" olursa çıkış "1" olur ancak bunun dışındaki tüm durumlarda "0" çıkışı elde edilir. **" Y = (A + B)' "** ile gösterilir.



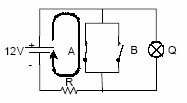
* Eğer A ve B anahtarları açık (A=0, B=0) ise akım devresini Q lambası üzerinden tamamlar lamba yanar Q=1.



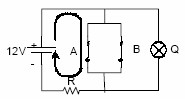
* Eğer A anahtarı açık(A=0), B anahtarı kapalı(B=1) ise akım devresini B anahtarı üzerinden tamamlar Q lambası yanmaz Q=0.



* Eğer A anahtarı kapalı (A=1), B anahtarı açık ise akım devresini A anahtarı üzerinden tamamlar Q lambası yanmaz Q=0.

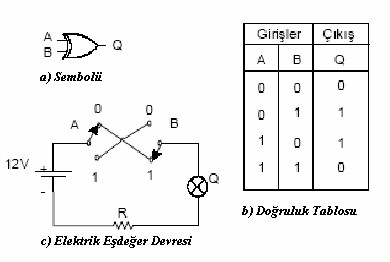


* Eğer A ve B anahtarları kapalı ise (A=1,B=1) ise akım devresini anahtar üzerinden tamamlar Q lambası yanmaz Q=0.

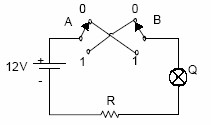


4.2.6 Özel Veya Kapısı

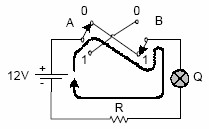
Bir çıkış, iki veya daha fazla giriş hattı bulunur. Tek bir özellik dışında "VEYA" kapısı ile birebir aynıdır. Bu özellik de; girişlerin hepsi "1" olursa çıkış değeri "1" yerine "0" olur. " **Y= A' . B + A . B' "** ile ifade edilebilir.



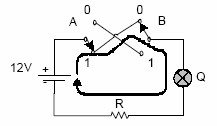
* Eğer A ve B anahtarları açık (A=0,B=0) ise akım devresini tamamlamaz ve lamba yanmayacaktır Q=0.

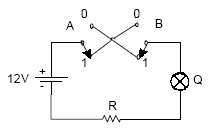


* Eğer A anahtarı açık(A=0), B anahtarı kapalı(B=1) ise akım devresini tamamlar Q lambası yanar Q=1.



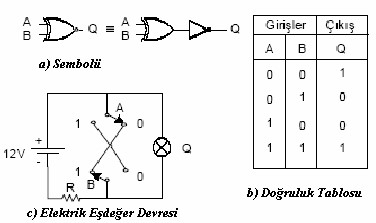
* Eğer A anahtarı kapalı (A=1), B anahtarı açık (B=0) ise akım devresini tamamlar Q lambası yanar Q=0.



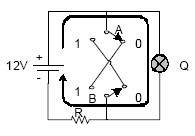
* Eğer A ve B anahtarları kapalı ise(A=1,B=1) ise akım devresini anahtar üzerinden tamamlar Q lambası yanmaz Q=0.

4.2.7 Özel Veya Değil Kapısı

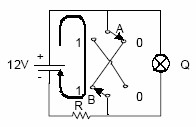
Bir çıkış, iki veya daha fazla giriş hattı bulunur. "ÖZELVEYA" fonksiyonunun tam tersi çıkış verir. "VEYA" kapısıdan tek bir özellik ile ayrılır ve bu özellik de; girişlerin hepsi "0" olursa çıkış değeri "0" yerine "1" olur. **" Y= A' . B' + A . B "** ile ifade edilebilir.



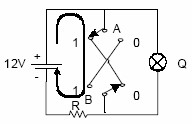
* Eğer A ve B anahtarları "0" konumunda ise akım devresini lamba üzerinden tamamlar Q=1.



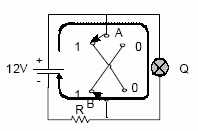
* Eğer A anahtarı "0"konumunda, B anahtarı "1" konumunda ise akım devresini anahtarlar üzerinden tamamlar Q lambası yanmaz Q=0.



* Eğer A anahtarı kapalı(A=1), B anahtarı açık (B=0) ise akım devresini tamamlar Q lambası yanar Q=0.



* Eğer A ve B anahtarları "1" konumunda ise akım devresini lamba üzerinden tamamlar Q=1.



1. Boolean Kuralları Ve Demorgan Teoremi
   1. Boolean Matematiği

Boolean aritmetiği ikilik sayı sisteminin matematiğidir. Matematikte gördüğümüz işlemlerin tamamı boolean aritmetiğinde de vardır. Fakat değişkenlerin alabileceği sadece iki değer vardır. Değişken olarak sadece “1” ve “0” ya da doğru ve yanlış vardır. Boolean aritmetiğinin 12 adet kanunu vardır, genellikle harfler kullanılarak bu kanunlar ifade edilir.

Boolean aritmetiğinde kullanılan sembollerin bazıları matematikteki ile aynı olmasına rağmen farklı anlam taşıyabilir. Harfin üzeri çizili ise bunun anlamı **değil** demektir, bazı kaynaklarda üst çizgi yerine ' kesme işareti kullanılır (A'= A).

* + işareti VEYA işlemini,
* ● işareti VE işlemini gösterir.

Nokta işareti genellikle kullanılmaz, A•B yerine AB kullanılır.

* + 1. Toplama İşlemi

Boolean toplama VEYA işlemine eşittir. Basit olarak toplamanın kuralını şöyle özetleyebiliriz;

* 0+0=0
* 0+1=1
* 1+0=1
* 1+1=1

Boolean aritmetiğinde toplama toplama dahil olan değerlerinin toplamıdır. Sadece VEYA işlemidir.

* A+B
* A+B'
* A+B+C’
* A’+B+C+D’

Toplam giriş literallerinden en az biri 1 olduğunda 1, aksi halde 0’dır.

* + 1. Çarpma İşlemi

Boolean çarpma VE işlemine eşdeğerdir. Kurallarını şöyle özetleyebiliriz;

* 0.0=0
* 0.1=0
* 1.0=0
* 1.1=1

Boolean aritmetiğinde çarpım giriş değerlerinin çarpımıdır.

Çarpım değerlerinden biri sıfır olduğunda 0, tümü 1 olduğunda 1’dir.

* 1. Boolean Kanunları
     1. Değişme Özelliği

VEYA işleminde veya VE işleminde girişlerin sırası önemli değildir.

* A+B=B+A veya AB=BA



A



B



X=AB



≡



B



A



X=B+A



A



B



X=A+B



≡



B



A



X=BA

* + 1. Birleşme Özelliği
* A+(B+C)=(A+B)+C

Bu özellik ikiden fazla değişkenin VEYA işleminin sırasının önemli olmadığını gösterir. Aynı özellik VE işleminde de geçerlidir.

* A(BC)=(AB)C



≡



B



C



X=B+C



A



X=A+(B+C)



B



C



X=A+B



A



X=(A+B)+C



Toplamada Birleşme özelliği.



≡



X=B



B



C



A



X=A(BC)



X=(AB)C



X=A

B



B



C



A



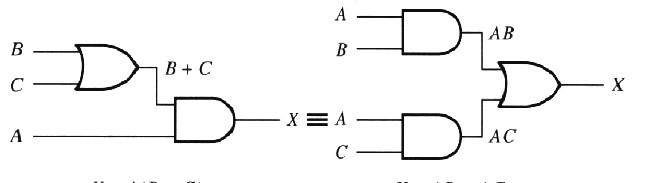
Çarpmada birleşme özelliği

* + 1. Dağılma Özelliği

A,B,C giriş değişkenlerini göstermek üzere *dağılma kanunu* aşağıdaki gibi yazılır.

* A . (B+C)=A . B + A. C

VEYA' lanmış B,C değişkenlerinin A ile VE işlemi ile elde edilen ifade, A değişkeninin B, C değişkenleri ile VE işlemi sonucu VEYA işleminden elde edilen ifadeye eşittir.



-

* 1. Boolean Kuralları

|  |  |
| --- | --- |
| 1. | A+0=A |
| 2. | A+1=1 |
| 3. | A•0=0 |
| 4. | A•1=A |
| 5. | A+A=A |
| 6. | A+A’=1 |
| 7. | A•A=A |
| 8. | A•A’=0 |
| 9. | *A*=*A* |
| 10. | A+AB=A |
| 11. | *A*+*AB*=*A*+*B* |
| 12. | (A+B)(A+C)=A+BC |

**KURAL 1 A+0=A KURAL 2 A+1=1**



A=0



0



X=0



A=1



0



X=1



X=A+0=A



1



X=1



1



X=1



X=A+1=1



A=1



A=0

**KURAL 3 A.0=0 KURAL 4 A.1=A**



A=0



1



X=0



A=1



1



X=1



X=A.1=A



A=0



0



X=0



A=1



0



X=0



X=A.0=0

**KURAL 5 A+A=A KURAL 6 A+A’=1**



A=0



A=0



X=0



A=1



A=1



X=1



X=A+A=A



1



X=1



0



X=1



X=A+A’=1



A=1



A=0

**KURAL 7 A.A=A** **KURAL 8 A.A’=0**



A=1



A=0



X=0



A=0



A=1



X=0



X=A.A=0



A=1



A=1



X=1



A=0



A=0



X=0



X=A.A=A

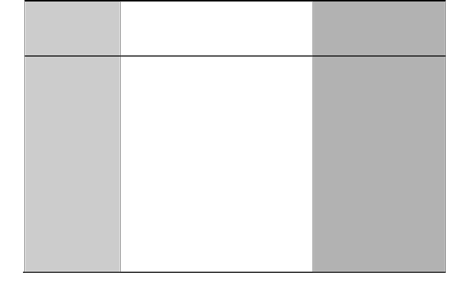
**KURAL 10**

Bu kural dağılma özelliği kullanılarak açıklanabilir.

A+AB = A(1+B) Dağılma özelliği.

= A.1

= A



A

B

AB

A+AB

0

0

0

0

0

1

0

0

1

0

0

1

1

1

1

1



Eşit

**KURAL 11**

A+A’B=A+B

A+A’B=(A+AB)+AB’

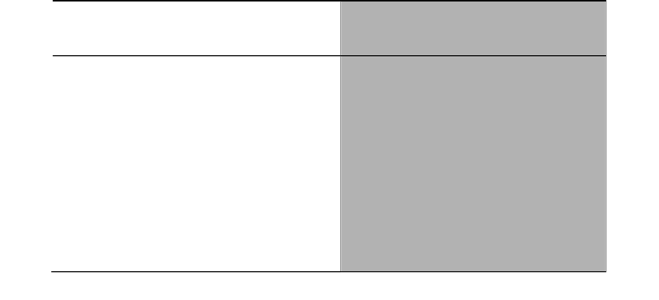
= (AA+AB)+AB

= AA + AB + AA’ + A’B

=(A+A’) + (A+B)

=1. (A+B)

=A+B



**A**

**B**

**AB**

**A+AB**

**A+B**

0

0

0

0

0

0

1

1

1

1

1

0

0

1

1

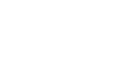
1

1

0

1

1



Eşit

**KURAL 12**

(A+B)(A+C)=AA+AC+AB+BC

=A+AC+AB+BC

=A(1+C)+AB+BC

=A.1+AB+BC

=A(1+B)+BC

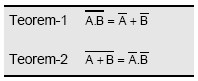
=A.1+BC

=A+BC

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **A+B** | **A+C** | **(A+B)(A+C)** | **BC** | **A+BC** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |  |

**5.4 DeMorgan Teoremleri**

DeMorgan teoremleri Boolean matematiğinin en önemli teoremleridir. İki değişken için DeMorgan teoremleri aşağıdaki gibi yazılır.

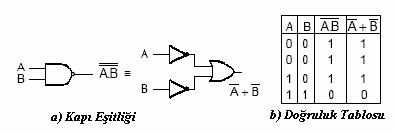


**5.4.1 Teorem-1**

**"Boolean matematiğinde çarpma işleminin değerleri toplama işlemine eşittir."**

A, B gibi iki değişkenin VEDEĞİL kapısına uygulanması ile elde edilen ifade bu iki değişkenin değilinin alınmasından sonra VEYA işlemi ile elde edilen ifadeye eşittir.



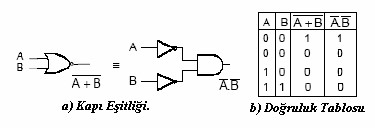


**5.4.2 Teorem-2**

**"Boolean matematiğinde toplama işleminin değerleri çarpma işlemine eşittir."**

A, B gibi iki değişkenin VEYA DEĞİL kapısına uygulanması ile elde edilen ifade bu iki değişkenin değilinin alınmasından sonra, girişler VE işlemi ile elde edilen ifadeye eşittir.





**Örnek 49**

Aşağıdaki Lojik ifadeye DeMorgan teoremini uygulayınız.



Lojik ifade içindeki işlemleri farklı bir değişken kullanarak tanımlayalım;



Basitleştirilmiş eşitlik;



BU ifadeye DeMorgan teoremini uygularsak olacaktır.

X ve Y değişkenlerini fonksiyona tekrar yazarsak Q eşitliği;

olur.

ifadesinde ve dönüşümü yapılırsa;

olacaktır.

Q ifadesi ise;

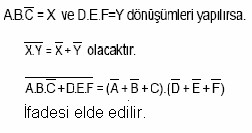
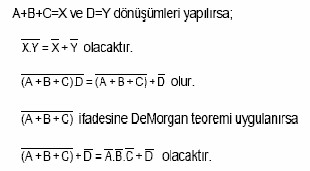


**Örnek 50**

 Aşağıdaki lojik ifadelere DeMorgan teoremini uygulayınız.

a)

b)

a) b)

**5.5 Boolean İfadesinden Sayısal Devrelerin Çizilmesi**

Devre tasarlanırken ilk önce Boolean ifadesinde kaç tane giriş değişkenin olduğu, daha sonra bu değişkenlerin hangi Boolean işlemine uygulandığı bulunmalıdır. Çizim sırasında Boolean matematiği işlem sırası takip edilmelidir.

**İşlem sırası parantez, DEĞİL, VE, VEYA şeklindedir.**

**Örnek 51**

Q = A.B + C ifadesini gerçekleştirecek sayısal devreyi tasarlayınız.

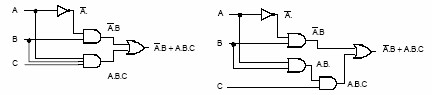
İşlemin gerçekleştirilmesine Boolean çarpma ile başlanır. Boolean çarpma işlemi VE kapısı ile gerçekleşeceğinden, ilk adımda A ile B değişkenlerinin VE kapısına uygulanması gerekir. Boolean çarpma işlemi ile elde edilen ifade (A. B), diğer giriş değişkeni ile Boolean toplama işlemi yapılır. Boolean toplama işlemi VEYA kapısı ile gerçekleşeceğinden A.B ifadesi C ile VEYA kapısına uygulanır.



**Örnek 52**

Q = Ā.B + A.B.C ifadesini gerçekleştirecek sayısal devreyi tasarlayınız.

Verilen Boolean ifadesinin çizimine öncelikle VE kapıları ile ifade edilen Boolean çarpma işlemi ile başlanır. Ancak VE kapılarına uygulanacak değişkenlerden DEĞİL olan varsa, öncelikle bu değişken DEĞİL kapısına uygulanarak bu işlem (Ā) gerçekleştirilir. DEĞİL'i alınan değişken diğer değişken(B) ile VE kapısına (Ā.B) uygulanır. Elde edilmek istenen A.B.C ifadesinde üç değişkenin VE kapısına uygulanması gerektiğinden üç girişli bir VE kapısı ve iki girişli iki VE kapısının ardı ardına bağlanması ile bu işlem gerçekleştirilir. Elde edilen bu iki ifade VEYA kapısına uygulanarak devrenin çizimi tamamlanır.



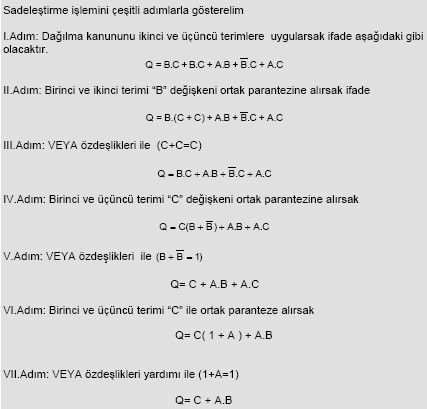
**5.6 Boolean İfadelerinin Sadeleştirilmesi**

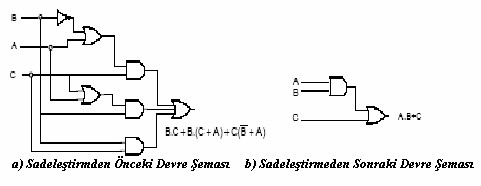
Çoğu zaman sayısal bir devre için elde edilen Boolean ifadesi uzun ve karmaşık olabilir. Devreyi bu haliyle tasarlamak işlemin maliyetinin artmasını ve hata yapma olasılığını beraberinde getirmektedir. Boolean teorem, kural ve kanunular yardımı ile ifadeler sadeleştirilerek daha az sayıda mantık kapısı ile sayısal devreler tasarlanabilir.

**Örnek 53**



İfadesini Boolean teoremleri yardımı ile indirgeyiniz.



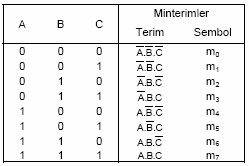


**5.7 Boolean Açılımları**

Boolean ifadeleri fonksiyonun doğruluk tablosundan elde edilen iki temel açılımdır. Bu ifadeler eğer bir sadeleştirme işlemi uygulanmazsa az sayıda değişken içermesi ender olarak karşılaşılan bir durumdur. Boolean ifadelerinin yazıldığı iki temel açılım minterimlerin toplamı ve maxterimlerin çarpımı olarak gösterilebilirler.

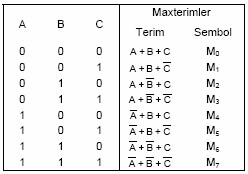
**5.7.1 Minterim ve Maxterim**

İkili bir değişken Boolean ifadesi olarak değişkenin kendisi (A) veya değişkenin değili ( A’ ) şeklinde gösterilebilir. VE kapısına uygulanan A ve B değişkenlerinin iki şekilde Boolean ifadesi yazılabileceğinden bu değişkenlerin alabileceği dört durum söz konusudur. Bu dört durum **minimum terim veya standart çarpım** adını alır. Benzer şekilde n sayıda değişken için 2n kadar minimum terimyazılabilir.



Üç değişkenin alabileceği sekiz (23) durum olduğundan 0'dan 7'ye kadar olan onluk sayıların ikilik karşılıkları, yazılabilecek durumları vermektedir. Her bir değişken ikilik sayıda eğer "0" ise değili"1" ise değişkenin kendisiyazılarak bulunur. Minimum terim Boolean ifadesini "1" yapan terimdir. Her bir minimum terim mjşeklinde gösterilir. Burada j indisi ilgili ikilik sayının onluk karşılığıdır.

Benzer biçimde n kadar değişken için değişkenin kendisi ve değili olmak üzere VEYA işlemini ile birleştirilmiş 2n kadar durum yazılabilir. VEYA işlemi ile birleştirilmiş bu durumlar ise maksimum terimler veya standart toplama adını alırlar. Her maxterim üç değişkenin VEYA işlemi ile birleştirilmiş halinden elde edilir ve burada ikilik sayıda değişken 0 ise değişkenin kendisi,1 ise değişkenin değiliyazılarak bulunabilir.

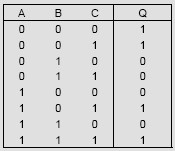


**5.7.1.1 Minterimlerin Toplamı**

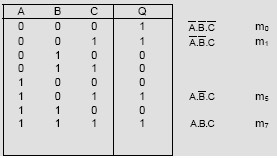
Boolean fonksiyonunu minterimlerin toplamı (çarpımların toplamı) cinsinden ifade edebilmek için fonksiyonun olduğu her durum için minimum terimler bulunur. Bulunan bu minimum terimler VEYA işlemi yapılarak fonksiyon minterimlerin toplamı (çarpımların toplamı) cinsinden yazılabilir.

**Örnek 54**

Aşağıdaki doğruluk tablosundan lojik ifadeyi minterimler cinsinden bulunuz.



Doğruluk tablosunda çıkış ifadesinin '1' olduğu her duruma ait minterim bulunduktan sonra bu terimler VEYA işlemi yapılarak lojik ifade elde edilir.



Yazılan minterimlerin her birisinin çıkışı 1 yapan terimler olduğu doğruluk tablosundan görülmelidir. Minterimlerin VEYA işlemi yapılması ile elde edilen ifade çıkışın 1 olduğu tüm durumları kapsayacaktır.

veya



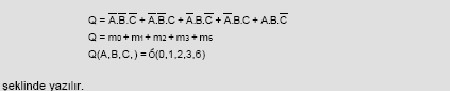
 şeklinde yazılabilir

Çoğu durumda doğruluk tablosunu vermek yerine aşağıdaki gösterimde kullanılabilir.



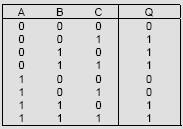
**5.7.1.2 Maxterimlerin Çarpımı**

Boolean fonksiyonları maxterimlerin çarpımı olarak da ifade edilebilirler. n sayıda değişkene ait 2n sayıda maxterim yazılabilir. Bu maxterimler fonksiyonun '0' olmasını sağlayan terimlerdir. Boolean fonksiyonunu maxterimlerin çarpımı formunda yazmak için fonksiyonun '0' olduğu her duruma ait maxterimler bulunur. Bulunan bu maxterimler VE ilemi yapılarak fonksiyon maxterimlerin çapımı formunda yazılabilir.

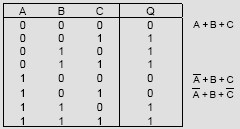


**Örnek 55**

Aşağıdaki doğruluk tablosundan lojik ifadeyi maxterimler cinsinden bulunuz.



Doğruluk tablosunun çıkış ifadesinin 0 olduğu her duruma ait maksimum terim bulunduktan sonra bu terimler VE işlemi yaparak lojik ifade elde edilir.



Yazılan minimum terimlerin çıkışın '0' olmasını sağlayan terimler olduğu doğruluk tablosundan görülmelidir.

 yazılabilir.

Çoğu durumda doğruluk tablosu yerine Şeklinde fonksiyon verilebilir

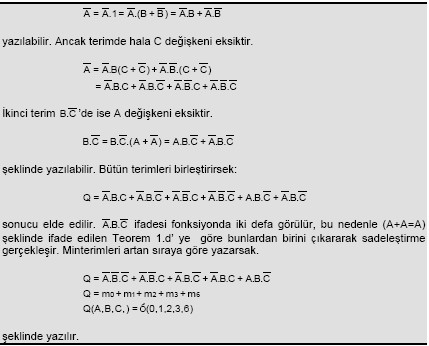


Boolean fonksiyonların maxterimlerin çarpımı (toplamların çarpımı) olarak ifade edebilmek için fonksiyonu VEYA terimleri haline getirmek gerekir.

Bu işlem:

(A+B).(A+C) =A+B.C

dağılma kanunu kullanılarak gerçekleştirilir.Daha sonra her bir VEYA teriminde eksik değişken varsa, A eksik değişkeni göstermek üzere, terim A.Ā ile VEYA işlemi yapılır.



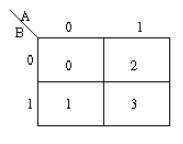
## Birleşik Mantık Devrelerinin Tasarımı

Birleşik mantık devresinin çıkışı giriş değişkenleri tarafından belirlenir. Bu tür devrelerin tasarımında öncelikle doğruluk tablosu elde edilir. Doğruluk tablosundan elde edilen ifade Karnaugh haritasına yerleştirilerek en sade ifade yazılır. Sadeleştirmenin amacı devreyi daha az sayıda geçit kullanarak gerçeklemektir. Karnaugh haritası 4 ve daha az değişken olduğunda kullanılışlıdır. Daha fazla değişkene sahip olduğunda Boolean kanunları kullanılarak ifadeler sadeleştirilebilir.

6.1 Karnough Haritası

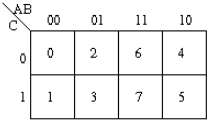
**6.2 2 Değişkenli Karnaugh Haritası**

Bu kutu tipi dijital devrede iki tane giriş olduğu zaman kullanılır. Ayrıca karnaugh haritası 2^2 =4 adet kutucuğa sahiptir.



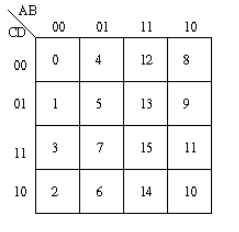
**6.3 3 Değişkenli Karnaugh Haritası**

Bu kutu tipi dijital devrede üç tane giriş olduğu zaman kullanılır. Ayrıca karnaugh haritası 2^3 =8 adet kutucuğa sahiptir.



**6.4 4 Değişkenli Karnaugh Haritası**

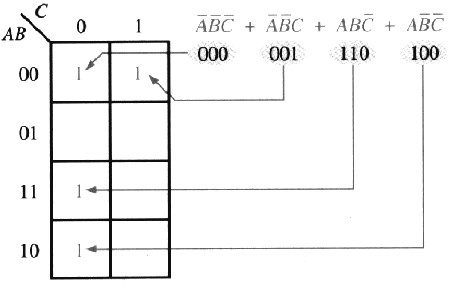
Bu kutu tipi dijital devrede dört tane giriş olduğu zaman kullanılır. Ayrıca karnaugh haritası 2^4 =16 adet kutucuğa sahiptir.



**ÖRNEK 56**

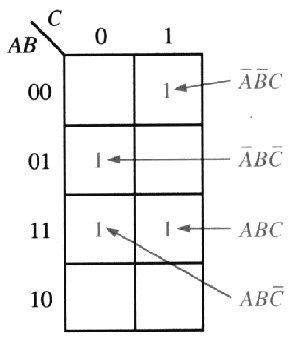
A’B’C+ A’B’C’+ABC’+AB’C’ ifadesini haritaya yerleştirin.

Öncelikle ikilik karşılıklarını yazalım



**Örnek 57**

A’B’C+ A’BC’+ABC+ABC’ ifadesini haritaya yerleştirin.



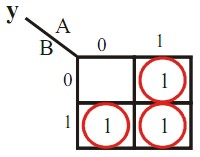
###### Karnough Haritasında Gruplandırma

Karnough haritalarının amacı var olan çıkış ifadelerini en az kapı kullanmak suretiyle aynı işi yapabilecek lojik ifadeyi elde etmektir. Çünkü Karnough ile ifade sadeleşirken aynı zamanda gereksiz yere kullanılabilecek lojik kapıları da azaltmış olmaktayız. Böylece elde edilecek devrenin hem fiziksel boyutu küçülecek hem de maliyeti düşecektir.

Gruplama yaparken şunlara dikkat edilir;

* Gruplama yaparken sadece “1” ler dikkate alınır. Boş olan yerler “0” demektir ve buraların gruplama yaparken önemi yoktur.
* Karnough haritalarında hedef en çok “1” i gruplamaktır.
* Hiçbir “1” açıkta kalmamalıdır.
* Gruplar 1, 2, 4, 8, 16 gibi iki ve ikinin üs katları şeklinde olmalıdır.
* Karnough haritaları üzerinde çapraz gruplama yapılamaz. Gruplar yan yana ya da alt alta olmalıdır.

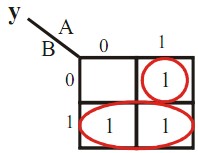
Aşağıda çeşitli gruplama şekilleri gösterilmektedir;



**YANLIŞ** bir gruplamadır.

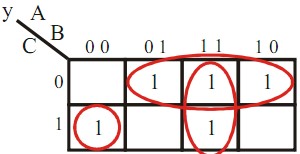
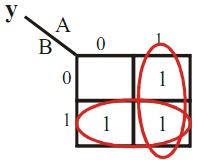
Çünkü en büyük grup oluşturacak şekilde gruplama yapılmamıştır.

Üç ayrı grup yapılarak çıkış ifadesi gereksiz yere uzatılmıştır.



**YANLIŞ** bir gruplamadır.

Grup yapılabiliyorsa tek başına 1 bırakılmamalıydı.



Bu **DOĞRU** bir gruplamadır.

Hiçbir “1” açıkta kalmamıştır.

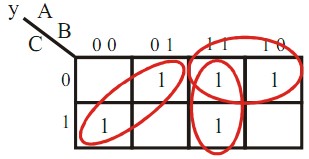
En büyük sayıdaki gruplar alınmıştır**.**

AB (11) hücresindeki 1 her iki gruba da dâhil edilebilir.

Bu **YANLIŞ** bir gruplamadır.

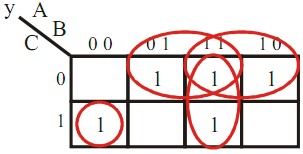
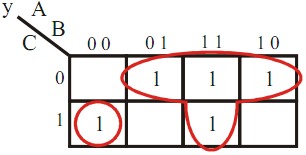
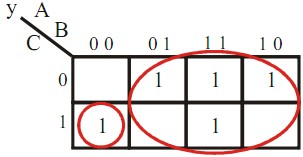
Çünkü 3 adet “1” ile gruplama yapılamaz.

Grup sayısı 1, 2, 4, 8.... olmalıdır.



Bu **YANLIŞ** bir gruplamadır.

Çünkü çapraz grup yapılamaz.



Bu **YANLIŞ** bir gruplamadır.

Boş kutular gruba dâhil edilemez.

Bu **YANLIŞ** bir gruplamadır.

Çünkü grup içinde hem alt alta hem yan yana “1” olamaz.

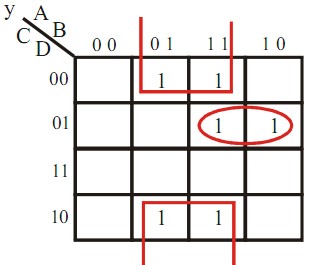
Grup ya yan yana ya alt alta olmalıdır.

Bu **DOĞRU** bir gruplamadır.

Burada 4 adet grup bulunmaktadır

Karnough haritasında en yukardan en aşağıya veya en sağdan

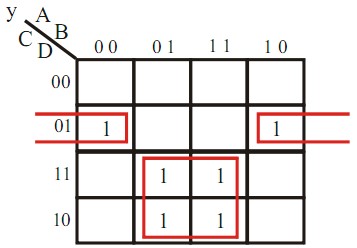
en sola geçiş vardır.



Bu **DOĞRU** bir gruplamadır.

Burada dörtlü ve ikili olmak üzere 2 adet grup vardır. Ucu açık

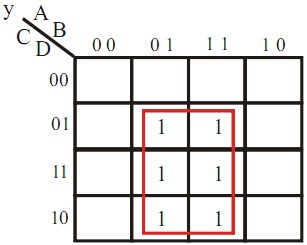
olan çizgiler birleşerek dörtlü grubu oluşturmaktadır.



Bu **DOĞRU** bir gruplamadır.

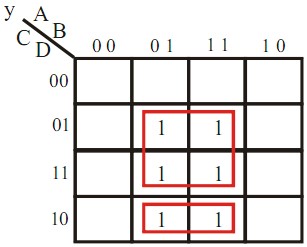
Burada biri ikili biri dörtlü olmak üzere 2 adet grup vardır.

Ucu açık olan çizgiler içinde ikili grup bulunmaktadır.



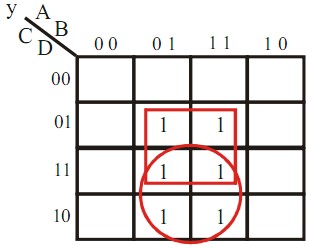
Bu **YANLIŞ** bir gruplamadır.

Çünkü altılı grup yapılamaz.



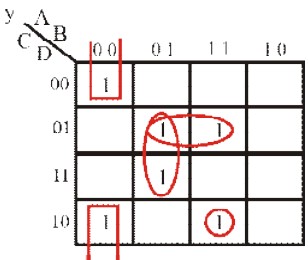
Bu **YANLIŞ** bir gruplamadır.

Dörtlü iki grup yapılabilirdi.

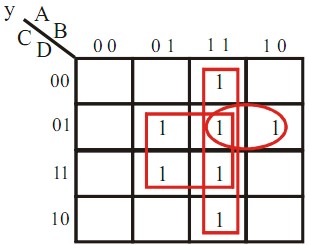


Bu **DOĞRU** bir gruplamadır.

Kare içinde ve daire içinde olmak üzere 2 adet dörtlü grup vardır.



Bu **DOĞRU** bir gruplamadır.



Bu **DOĞRU** bir gruplamadır.

###### Karnough Haritasından Sadeleşmiş İfadeyi Yazma

Karnough haritalarında ifadelere göre gruplandırmalar yapıldıktan sonra önemli olan nokta her grubu doğru bir şekilde ifadelendirmektir.

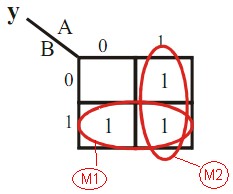
|  |  |  |  |
| --- | --- | --- | --- |
| A | B | Y | İfade |
| 0 | 0 | 0 | **.** |
| 0 | 1 | **1** | **.B** |
| 1 | 0 | **1** | **A.** |
| 1 | 1 | **1** | **A.B** |

Doğruluk tablosuna göre Y denklemini Y=1 olanlara göre yazarsak

**Y=A’.B +A. B’ + A.B** olur. Yazılan ifade çarpımların toplamı şeklindedir.

Bu ifadeye göre devre şemasını çizmek gerekir. Eğer devre şeması çizilecek olsa üç adet VE kapısı ve bir adet VEYA kapısı ayrıca A ve B’nin değillerini alacak değil kapısı çizilmek zorunda. Karnough haritasını kullanarak aynı işi gören daha kısa bir ifadeyi elde edilebilir ve elde edilen ifade kısa olduğu için yapılan devre daha az elemanla yapılmış olacaktır.

Eğer bu doğruluk tablosuna göre Karnough haritası hazırlanırsa;



Bu şekilde yapılan gruplandırma doğrudur.

**A=0 , B=1 ve A=1, B=1** olduğu hücrelerin ikisi **M1** olarak adlandırılmıştır.

**A=1, B=0 ve A=1, B=1** olduğu hücrelerin ikisi **M2** olarak adlandırılmıştır.

Karnough haritasında çıkacak ifade olarak **Y=M1+M2** yazılacaktır.

Burada M1 için Karnough karşılığını yazılırsa **M1= B** yazılabilir.

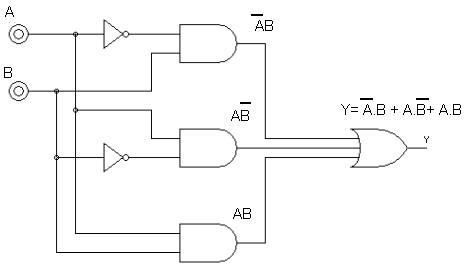
Peki neden B, çünkü M1 gurubuna ait 1’ler B’nin bulunduğu satır içindedir.

A için hiçbir şey yazılamaz. Çünkü A’nın bulunduğu sütunda A=0 ve B=0 hücresi (0), yani **A’** (A değil) bulunduğu sütun 0 iken A’nın bulunduğu sütun 1’dir.

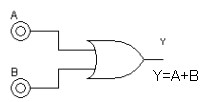
M2 grubu yazılacak olursa M2= A olur. Burada da B ile ilgili hiçbir şey yazılamaz. Çünkü diğer sütun A’nın ve B’nin değilini barındırmaktadır, hem 0 hem de 1 vardır. Bu nedenle seçilmez.

Karnough ile ortaya çıkan ifade **Y=M1+M2 = B + A** olmuştur. Görüldüğü gibi devre aynı işi gören bir adet VEYA kapısıyla yapılabilmektedir.

Devrenin şemasını Karnough haritası kullanmadan yapılırsa aşağıdaki gibi olacaktır.



**Karnough haritası kullanmadan yapılan devre şeması**

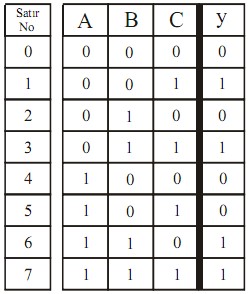


**Karnough haritası kullanılarak yapılan devre şeması**

**Örnek 58**

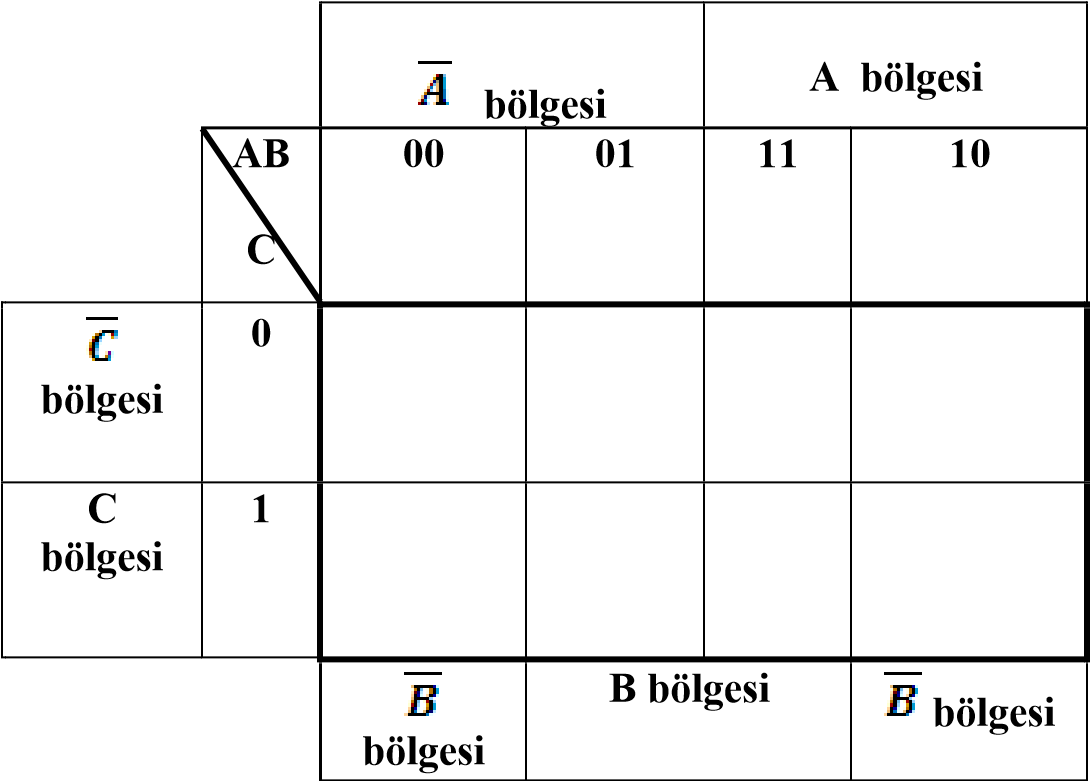
y = A.B.C' + A'.B'.C + B.C şeklinde verilen fonksiyonu Karnough haritası yöntemi ile sadeleştiriniz.

Tablo yapılırsa;

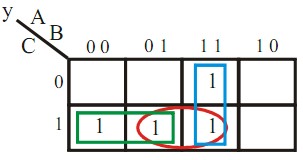


Tablodaki ifadelerin Karnough haritasına aktarılmasına geçmeden önce Karnough haritasında

A,B,C bölgelerinin bilinmesi gerekir. Aşağıdaki tabloda bu bölgelerin hangi hücreleri kapsadığı gösterilmektedir.

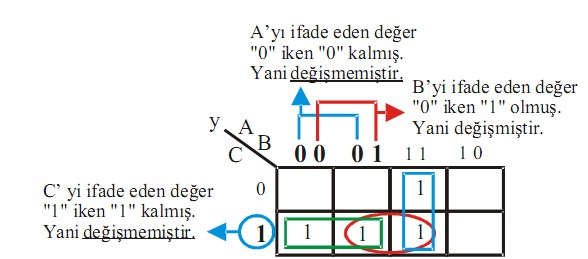


Elde edilen tabloya göre A,B,C değişkenlerine ait Y çıkış ifadesiyle alacak olursak dört adet 1‘den oluşmaktadır. Bu ifade Karnough haritasına yerleştirilirse aşağıdaki gibi olmaktadır.



Karnough haritasına bakıldığında 3 adet grup olduğu görülür. Bu gruplar yeşil, kırmızı ve mavi renkler ile ayrı ayrı gösterilmiştir. Her biri 2 adet “1” içermektedir yani ikili gruptur.

* İndirgenmiş fonksiyon yazılırken her bir gruba ayrı ayrı bakılır.
* Her gruptan çarpım şeklinde 1 ifade çıkar.
* Her gruptan çıkan bu ifadeler toplanınca (yani toplam şeklinde yazılınca) indirgenmiş fonksiyon yazılmış olur. Örnekte 3 adet grup olduğundan y= Y + K + M şeklinde bir ifade oluşacaktır.
* Y ifadesini bulmak için yeşil gruba bakıldığında burada A’nın değişmemiş, B’nin değişmiş, C’nin ise değişmemiştir olduğu görülmektedir. Değişen ifadeler sadeleşen ifadelerdir. Değişmeyen ifadeler ise alınır.



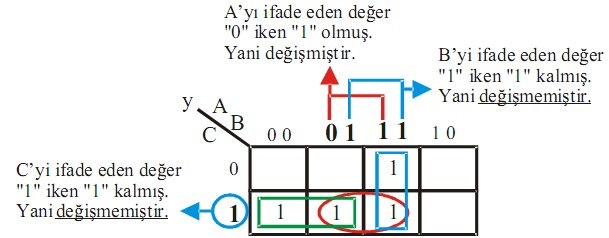
Grup içinde değeri değişenler indirgenmiş demektir ve indirgenmiş fonksiyon yazımında kullanılmaz. Örnekte B’ nin değeri değiştiğinden B yazılmayacaktır.

Değeri değişmeyenler ise çarpım şeklinde alınır. Örnekte A ile C‘nin değeri değişmediğinden çarpım şeklinde yazılacak demektir. Bu ifadeler çarpım şeklinde yazılırken;

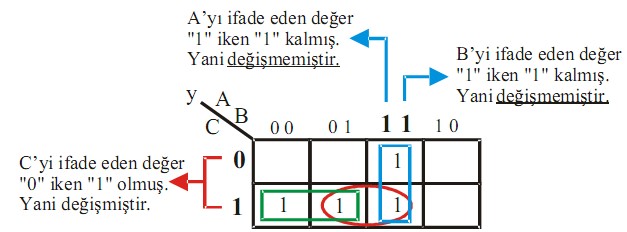
* Değeri “1” olanlar kendileri şeklinde (A, B, C …..) yazılır.
* Değeri “0” olanlar değilleri şeklinde (A', B', C' …...) yazılır.

Bu bilgiler ışığında yeşil gruptan çıkacak sonuç **(A' . C)** olacaktır.

**Kırmızı** gruba bakıldığında bu gruptan çıkacak sonuç **(B . C)** olacaktır.

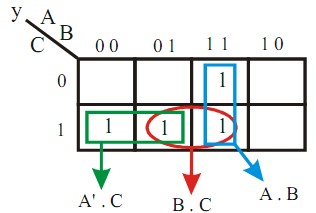


**Mavi** gruba bakıldığında bu gruptan çıkacak sonuç **(A . B)** olacaktır.



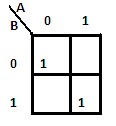
Bu 3 gruptan çarpım şeklinde çıkan sonuçlar ard arda toplandığında denklem ya da ifade ortaya çıkmış olur.

**y = (A' . C) + (B . C) + (A . B) şeklinde olacaktır.**



**Örnek 59**

Aşağıdaki Karnough haritasının çıkış ifadesini yazınız.



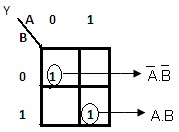
Şekilde görüldüğü gibi 1’ler çapraz olarak gruplandırma yapılamayacağı için ayrı olarak gruplandırılır. A’nın 0, B’nin 0 olduğu grupta A ve B değişiklik göstermediği için etkisiz eleman yoktur. A’nın ve B’nin 0 olduğu (A'.B') kutusudur.

Bunun karşılığı ise **y1=A’.B’** olur.

A’nın 1, B’nin 1 olduğu grupta A ve B değişiklik göstermediği için burada da etkisiz eleman yoktur. A’nın ve B’nin 1 olduğu (A.B) kutusudur.

Bunun karşılığı ise **y2 = A .B** olur.

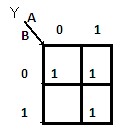
Karnough haritasına ait çıkışı bulmak için gruplar ayrı ayrı toplanır. Grup içi ifadeler çarpım işlemine ve oluşan gruplar toplama işlemine tabi tutulmuştur.



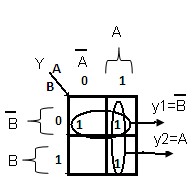
**Y=A’.B’+AB** olacaktır.

**Örnek 60**

Aşağıdaki Karnough haritasının çıkış ifadesini yazınız.



Karnough haritası içinde gruplama işlemi, ikinin katları olacak şekilde ve en fazla 1 kapsayacak şekilde yapılır. Karnough haritası içinde üç adet 1 olduğundan y1 ve y2 olmak üzere iki grup oluşmuştur.



Öncelikli olarak gruplara ait çıkış ifadeleri çarpımlar şeklinde yazılacak ve daha sonra gruplar toplanacaktır. y1 'e ait çıkış ifadesi yazılırsa, burada A değişmiş B ise değişmemiştir.

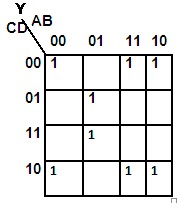
Buna göre çıkış ifadesi **y1= B’**  olur. y2 ye ait çıkış ifadesi yazılırsa; burada ise B değişmiş A değişmemiştir. Buna göre y2’nin çıkış ifadesi **y2 = A** olur.

Karnough haritasına ait çıkışı bulmak için gruplar ayrı ayrı toplanır. Grup içi ifadeler çarpım işlemine ve daha sonra gruplar toplama işlemine tabi tutulmuştur.

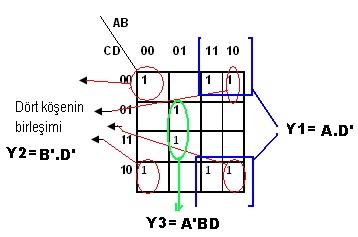
**Y = y 1 + y2 ise Y = A + B olur.**

**Örnek 61**

Aşağıdaki Karnough haritasının çıkış ifadesini yazınız.

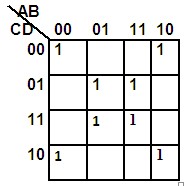


En fazla 1 ‘lerle yapılabilecek grup üç adettir.

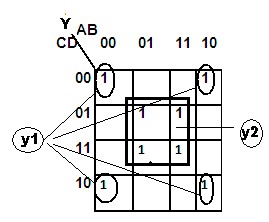


**Örnek 62**

Aşağıdaki Karnough haritasının çıkış ifadesini yazınız.



Karnough haritasının köşeleri bir grup, ortadaki dört adet (1) bir grup yapılır.



**Y= B’.D’ +A.D olur.**

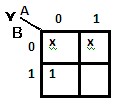
###### Farketmezlere Göre Karnough Haritası

Bazı tasarımlarda gerek giriş gerekse çıkış degişkenlerinin bir önemi yoktur. Bu durumda ifadenin önemsiz olduğunu belirtmek için 0 ve 1 dışında özel bir karakter olan “X” kullanılır. Buna fark etmez, önemsiz vb… gibi adlar verilebilir.

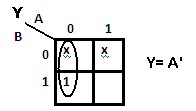
“X” bulunan kutular duruma göre “0” veya “1” kabul edilir. Burada amaç en büyük gruplamayı yapmaktır. Önemsizlerin hepsi kullanılabileceği gibi en büyük gruplama yapabilmek için istenilen “X”i alıp bazı “X” leri grup dışında bırakabiliriz.

**Örnek 63**

Aşağıdaki Karnough haritasının çıkış ifadesini yazınız.

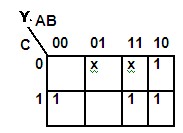


Karnough haritasında farketmezlerden sadece biri gruplamaya "1" olarak dahil edilmiş, diğerinin kullanılmasına gerek duyulmamıştır.

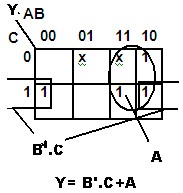


**Örnek 64**

Aşağıdaki Karnough haritasının çıkış ifadesini yazınız.

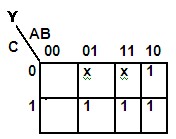


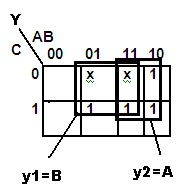
İki grup oluşturulabilir. Farketmezlerden biri guruba dahil edilir.



**Örnek 65**

Aşağıdaki Karnough haritasının çıkış ifadesini yazınız.





6.4 Tablo Yöntemi İle Sadeleştirme (Quine-McCluskey)

Quine-McCluskey'nin algoritması (ya da asal içerenlerin yöntem ) için kullanılan bir yöntemdir en aza indirilmesi arasında Boole fonksiyonları tarafından geliştirilen Willard V Quine  ve genişletilmiş Edward J. McCluskey . İşlevsel olarak Karnaugh eşlemesi ile aynıdır , ancak tablo biçiminde bilgisayar algoritmalarında kullanım daha verimli hale gelir ve ayrıca Boole işlevinin en düşük biçimine ulaşıldığını kontrol etmek için belirleyici bir yol sunulur. Bazen tablolama yöntemi olarak adlandırılır.

**Yöntem iki adımdan oluşur;**

* İşlevin tüm temel sonuçlarını bulma .
* Fonksiyonun temel asal sonuçlarını ve fonksiyonu kapsamak için gerekli diğer asal çarpımları bulma.

1. Bileşik Mantık Devreleri

Dijital sistemlerde kullanılan mantık devreleri, bileşik devreler ve ardışık sıralı devreler şeklinde gruplandırılabilir. Temel lojik kapılardan oluşan ve devrelerin çıkışları doğrudan girişlerin o anki durumlarına göre belirlenen devreler, ‘bileşik mantık devreleri’ olarak adlandırılır. Bileşik devreler bazen, ‘birleşik mantık devreleri’ olarak da isimlendirilir. Bir bileşik devre; giriş değişkenleri, lojik kapılar ve çıkış değişkenlerinden oluşur . Lojik kapı, giriş değişkenlerini alır, bunları işler ve çıkış için bilgi (değişkenler) üretir. Yapılan işlem, ikili giriş verilerin işlenmesi ve uygun çıkış verileri şekline dönüştürülmesidir. Giriş verileri; bir harici kaynaktan gelen ‘n’ sayıda ikili giriş değişkenlerini, çıkış verileri; bir harici devreye doğru yönelmiş ‘m’ sayıda çıkış değişkenlerini içerir. Giriş değişkenlerinin değeri, 2n sayıda farklı ikili giriş kombinasyonundan birisi olabilir ve her bir giriş kombinasyonu için yalnızca bir çıkış kombinasyonu mevcuttur.

Çok farklı uygulama alanları bulunan bileşik mantık devreleri, dört farklı grup altında incelenebilir;

* Kodlama ile İlgili Lojik Devreler: Kodlayıcı (Encoder), Kod çözücü (decoder), Kod değiştirici / çevirici (Code converter).
* Çoklayıcı Devreler, Veri seçiciler (Multiplexer-Data selector).
* Azlayıcı Devreler, Veri dağıtıcılar (Demultiplexer - Data distributor).
* Kıyaslama ve Aritmetik İşlemler ile İlgili Devreler: Karşılaştırıcı (comparator), Toplayıcı (adder), Çıkarıcı (substractor), Çarpıcı (multiplier).
  1. **Bileşik Devre Tasarımı**

Lojik tasarımın içerdiği işlem basamakları aşağıdaki şekilde özetlenebilir;

* Problem belirlenir.
* Giriş değişkenlerinin sayısı ve gerekli çıkış değişkenleri tespit edilir.
* Giriş ve çıkış olarak kullanılacak değişkenlere isim verilir.
* Giriş ve çıkış değişkenleri arasındaki ilişkiyi belirleyen doğruluk tablosu oluşturulur.
* Her bir çıkış için uygun Boolean fonksiyonu yazılır.
* Elde edilen Boolean fonksiyonları sadeleştirilir.
* Lojik devre çizilir.

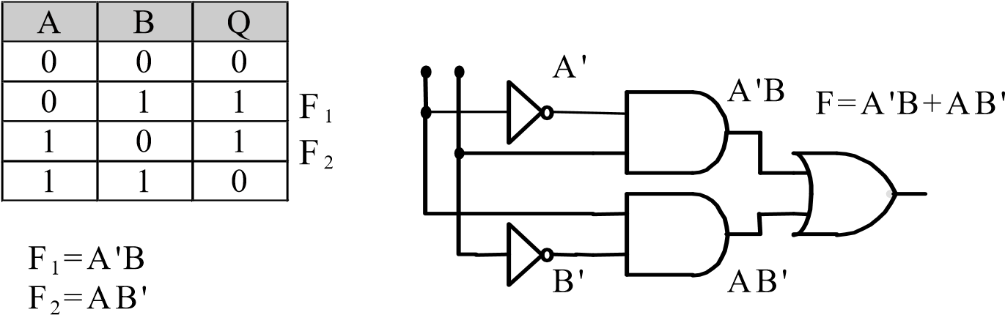
Doğruluk tablosundan elde edilen çıkış eşitliklerini sadeleştirmek için, sadeleştirme yöntemlerinden uygun olan birisi kullanılabilir. Sadeleştirilen eşitliklerin aşağıdaki özellikleri taşıması istenir;

* En az sayıda lojik kapı içermesi.
* Her bir kapının en az sayıda girişe sahip olması
* Devrenin minimum yayılım zamanına sahip olması.
* Devrenin minimum sayıda bağlantı içermesi.
* Her bir kapının, sürme kapasitesi sınırının altında eleman sürmesi.

**Örnek 66**

İki girişli dijital bir sistemde girişlerin farklı olduğu durumlarda çıkışın ‘1’ olmasını sağlayacak lojik devreyi tasarlayalım ve tasarlanan devreyi temel lojik elemanları ile gerçekleştiriniz.

* İki değişkenli doğruluk tablosu oluşturulur.
* Yapılmak istenen işin tanımından; girişlerin farklı olduğu durumlar tespit edilir.
* Çıkışın ‘1’ olduğu kombinasyonlar yazılarak, bu kombinasyonlara ait eşitlik yazılır ve bu eşitlikler birleştirilir. Bu işlem sonucunda F=A'B+AB' eşitliği bulunur.
* Yazılan eşitlikte sadeleştirme yapılamadığından eşitlik doğrudan lojik kapılarla gerçekleştirilir.

 A B

F=F1+F2=A'B+AB'

**Örnek 67**

Üç girişli bir sistemde, girişlerin birden fazlasının lojik ‘1’ olduğu durumlarda çıkışın ‘1’ olmasını sağlayacak lojik devreyi, lojik tasarımda kullanılan işlem sırasına göre gerçekleştiriniz.

Yapılması istenen iş tanımlanır. Bu işlemi gerçekleştirecek doğruluk tablosu oluşturulur.

|  |  |  |  |
| --- | --- | --- | --- |
| **A** | **B** | **C** | **Q** |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

Girişlerden birden fazlasının ‘1’ olduğu durumların tespit edilir. Çıkışın ‘1’ olduğu durumları temsil eden kombinasyonlar yazılır ve yazılan kombinasyonlar birleştirilerek lojik fonksiyon elde edilir.

Yazılan lojik eşitlik sadeleştirilir. Bulunan eşitlikte ABC grubu içerisinde bulunan değişkenlerden en az ikisinin ortak olduğu kombinasyonlarda, ortak olan değişkenler paranteze alınarak eşitlik;

**F=BC(A+A')+AC(B+B')+AB(C+C')**

**1 1 1**

**=BC+AC+AB**

şeklinde ifade edilebilir.

Elde edilen ifade bu şekilde gerçekleştirilebileceği gibi, ortak C parantezine alınarak;

**F = C(A+B)+AB** şeklinde de gerçekleştirilebilir.

A.B+A.C+B.C

A.B

A.C

B.C

A

A.B

A+B

(

A+B).C

AB+C(A+B)

BC

1. **Multiplexer Ve Demultiplexer**

Multiplexer veya veri seçiciler her zaman aralığında girişlerindeki verinin birini tek bir çıkışa bağlar. Demultiplexer ise bu işlemin tersini gerçekleştirir. Aynı tüm devre (dekoder) kod çözücü olarak kullanılabilir. Multiplexer kısaca **MUX**, demultiplexer ise **DEMUX** olarak adlandırılır. Multiplexer ve demultiplexer tüm devreleri kullanılarak birleşik mantık devreleri tasarlanabilir. Eşlik biti üreteci/denetleyici iletişim sistemlerinde oluşan hataları denetlemek için üretilir. Sadece bir bitin değerinin değişmesini algılar. İletilecek olan veri bitleri içerisindeki "1"lerin sayısının tek ya da çift olmasına göre hata denetimi yapılır. Eşlik biti üreteci/denetleyici çift veya tek eşliğe göre çıkışını etkin yapar. Tüm devreler üzerinde hem tek eşlik biti için hem de çift eşlik biti için çıkış bulunur.

## MULTİPLEXER (ÇOKLAYICI)

Multiplexer (MUX) sayısal veriyi birkaç kaynaktan alıp iletim için ortak tek bir yola aktaran bir aygıttır. Temel bir veri seçicide birkaç veri giriş hattı ve tek bir çıkış hattı vardır. Hangi girişin çıkışa aktarılacağını belirleyen birkaç veri seçme hattı da bulunmaktadır.



MUX



0

1

2

3



Y



Veri

çıkışı



S

0

S

1



D

0

D

1

D

2

D

3



Veri

seçme



Veri

girişleri

* İkilik 0 (S1 = 0 ve S0 = 0) veri seçme hatlarına uygulanırsa, D0 girişindeki veri çıkışa aktarılır.
* İkilik 1 (S1 = 0 ve S0 = 1) veri seçme hatlarına uygulanırsa, D1 girişindeki veri çıkışa aktarılır.
* İkilik 2 (S1 = 1 ve S0 = 0) veri seçme hatlarına uygulanırsa, D2 girişindeki veri çıkışa aktarılır.
* İkilik 3 (S1 = 1 ve S0 = 1) veri seçme hatlarına uygulanırsa, D3 girişindeki veri çıkışa aktarılır.

|  |  |  |
| --- | --- | --- |
| **Veri Seçme Girişleri** | |  |
| S1 | S0 | Seçilen veri |
| 0 | 0 | D0 |
| 0 | 1 | D1 |
| 1 | 0 | D2 |
| 1 | 1 | D3 |

Veri seçicinin bu işlemi yapmasını sağlayan mantık devresi incelendiğinde veri çıkışı seçilen veri girişinin durumuna eşittir. Veri çıkışını veri girişleri ve veri seçme girişleri cinsinden yazılmalıdır. Bu şu şekilde yapılabilir;

Yalnızca S1 = 0 ve S0 = 0 olduğunda, Y veri çıkışı D0 veri girişine eşittir;

**Y=D0S1‘S0‘**

Yalnızca S1 = 0 ve S0 = 1 olduğunda, veri çıkışı D1 veri girişine eşittir;

**Y=D1S1‘S0**

Yalnızca S1 = 1 ve S0 = 0 olduğunda, veri çıkışı D2 veri girişine eşittir;

**Y=D2S1S0’**

Yalnızca S1 = 1 ve S0 = 1 olduğunda, veri çıkışı D3 veri girişine eşittir;

**Y=D3 S1 S0**

Bu terimler VEYA işlemi yapıldığında veri çıkışının toplam ifadesi elde edilir;

**Y=D₀S₁’ S₀’ + D₁S₁’ S₀+D₂S₁ S₀’+D₃S₁S₀**

Bu eşitliğin mantık geçitleriyle gerçeklenmesi için 4 adet 3 girişli VE geçidi, 1 adet 4 girişli VEYA geçidi ve S₁ ve S₀‘ın değillerini almak için iki adet çevirici gerekmektedir. Herhangi bir giriş hattından veri seçilebildiğinden bu devreye veri seçici de denilmektedir.

I

1

I

N-1

Veri girişleri

Çıkış

Veri

Girişleri

MUX

Çıkış

I0

Seçme girişleri

S0 S1

Seçme girişleri

## DEMULTİPLEXER (AZLAYICI)

Demultiplexer temel olarak multiplexer’ın yaptığı işin tersini yapar. Bir hattan veriyi alır ve belirli sayıdaki çıkış hatlarından birine aktarır. Bu nedenle, bu aygıta veri çoklayıcı ya da veri dağıtıcı denilir.



S

0

S

1



D

0

D

1

D

2

D

3



Veri

girişi



Seçme

hatları

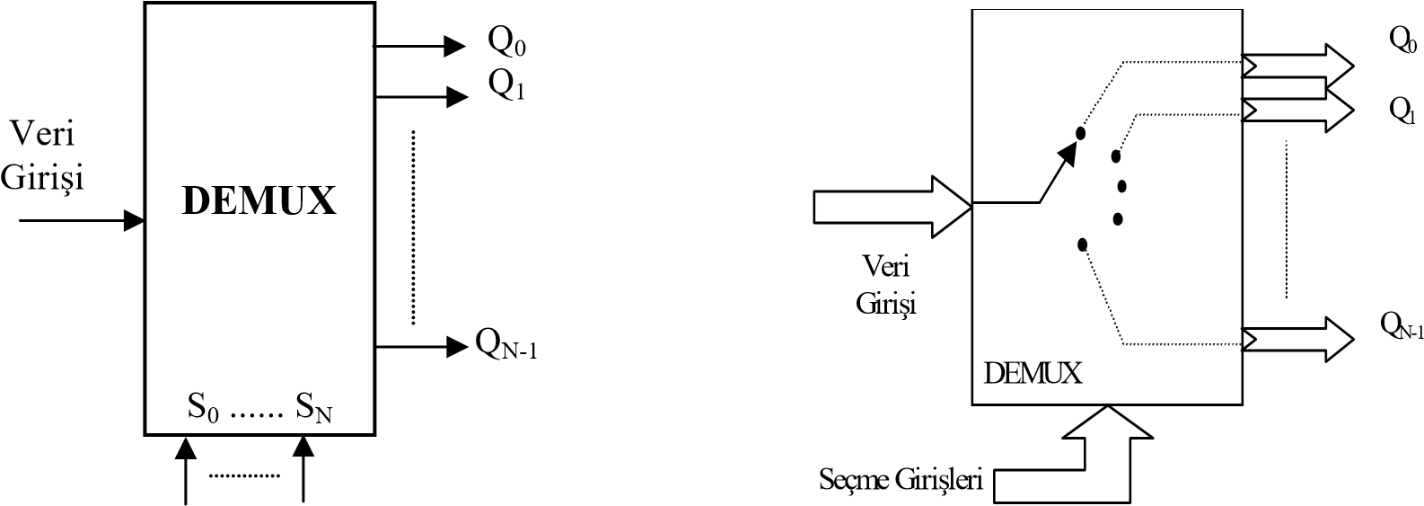


Veri

çıkış

hatları

Veri-giriş hattı tüm VE geçitlerinin girişlerine gitmektedir. İki veri-seçme hattı her seferinde yalnız bir geçide izin vermektedir ve veri-giriş hattındaki veri seçilen geçitten ilgili veri-çıkış hattına aktarılır.



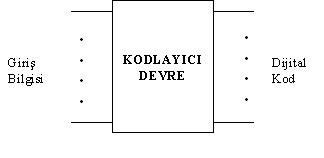
Seçme Girişleri

1. Decoder Ve Encoder

Endüstride, haberleşmede, bilgisayarda ve daha birçok yerde dijital elektronik kullanılır. Elektronik sistemlerde, çiplerin içerisinde mantık devreleri bulunmaktadır. Bu mantık devrelerinde en basitinden kodlayıcı ve kod çözücülerle işlemler yapılır.

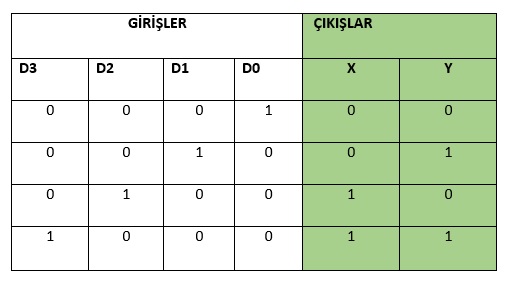
# Encoder (Kodlayıcı) Devreler

**Kodlayıcı**, sayısal bir bilginin, başka bir sayısal bilgiye dönüştürülmesi için kullanılan lojik bir devredir. Örneğin, onluk sayı sisteminde girilen sayısal bilgileri, ikilik sayı sistemlerine dönüştürür. Alfanümerik tuş takımlarında ve klavyede kodlayıcı devreler kullanılır.



# Kodlayıcı Tasarımı

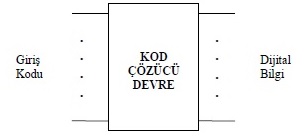
Kodlayıcı tasarlarken giriş sayısını belirlememiz lazım. Bu girişleri kodlayabilecek minimum çıkış sayısını hesaplarız. 12 adet giriş kodlanması gerekiyorsa 2n > 12 formülü yardımıyla n’nin değerinin 4 olduğunu ve devrede 4 tane çıkış olması gerektiğini buluruz. Doğruluk tablosunu oluşturup, devrenin tasarımına geçilir.4 bitlik giriş kodlamak istersek;

2n>4 formülünden n=2   


D0 girişi 1 olunca çıkış 00  
D1 girişi 1 olunca çıkış 01  
D2 girişi 1 olunca çıkış 10  
D3 girişi 1 olunca çıkış 11  
  
Çıkışların 1 olduğu yerlerin denklemi yazılırsa;  
  
X=D2+D3  
Y=D3+D1

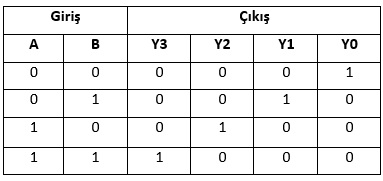
## **Decoder (Kod Çözücü) Devreler**

Kod çözücü (decoder), kodlayıcının tersi işlem yapar. Kodlanmış bilgileri anlaşılır hale dönüştürmek için kullanılır. Örneğin cep telefonumuza gelen mesajları 2’lik sayı sisteminde anlayamayız. Gelen bilgiler çözümlenerek metin formatı haline getirilir. Bilgisayarda anakart, diğer mikroişlemcilerde adresleme amacıyla kullanılır.



## **Kod Çözücü Tasarımı**

Kod çözücü tasarlarken n tane giriş varsa 2n tane çıkış vardır. Doğruluk tablosu oluşturularak sistem tasarlanmış olur.  
  
2X4 Kod çözücü (2 giriş 4 çıkış)

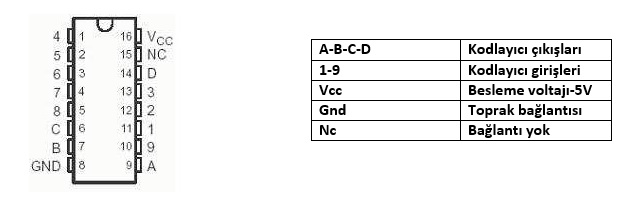
  
  
  
**Y0= A’B  
Y1=A’B  
Y2=AB’  
Y3=AB**

## **Kodlayıcı ve Kod Çözücü Entegreler**

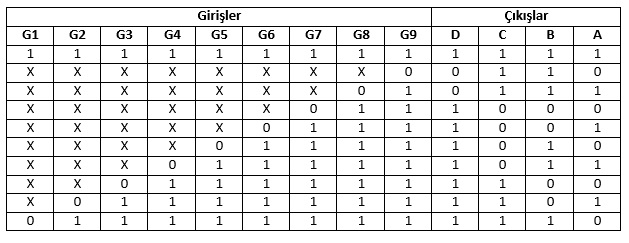
Entegreler devre tasarımını daha kolay ve düşük maliyetli hale getirmektedir. Kodlayıcı ve kod çözücü tasarlarken çeşitli devre elemanları kullanmaktansa tümleşik entegreler tercih edilir. TTL, CMOS gibi entegre türleri çoğunlukla kullanılır.Kod çözücü (decoder) ile bir kodlayıcı (encoder) arka arkaya bağlanırsa, sistemin girdi değeri, çıktı değeri olarak okunur. Kodlayıcı devresini, kapılar kullanarak yapmak da mümkündür.  
Sayısal kodlayıcılar motorların şaftlarına yerleştirilerek servo motor üretiminde kullanılmaktadırlar. Hareketi dijital darbelere (puls) dönüştüren cihazlardır. Elde edilen bu dijital darbeler sayılarak ya da bit şeklinde okunarak hareketin hızını veya cismin o anki konumunu elde etmek mümkündür.

* + 1. Kodlayıcı Entegreler

**74LS147 entegresi**  
Desimal – BCD kodlama için kullanılır. Klavye kodlama ve seçme işlemleri için uygundur.

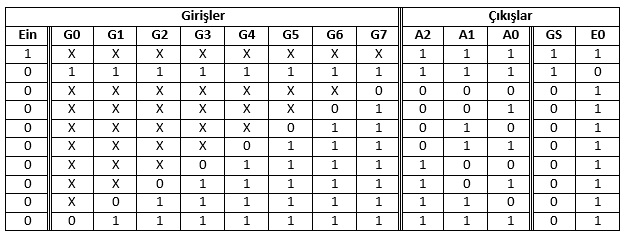
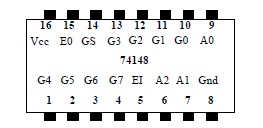
  
  
Normal koşullarda çıkış aktif olduğunda değeri lojik 1 olması gerekir. Bu çıkışa aktif 1 (**Active High**) adı verilir. Eğer ki çıkış aktif olduğunda değeri lojik 0 oluyorsa buna aktif 0 (**Active Low**) adı verilir.

Giriş ve çıkışları aktif 0’dır. Yani girişin aktif olabilmesi için lojik 0 uygulanmalıdır, çıkış aktif olduğunda lojik 0 alınır.  
Öncelikli kodlayıcıdır, yüksek girişin önceliği vardır. En yüksek giriş 9. Giriştir.

  
74LS147 doğruluk tablosu

**74LS148 entegresi**

Octal – BCD kodlama için kullanılır. 74148 entegresi 8 giriş 3 çıkış öncelikli kodlayıcıdır. Bu entegre için, Bu entegrenin girişleri değillenmiştir. Yani girişleri aktif etmek için lojik 0 verilmelidir. Bu entegrenin 8 girişi(G0 – G7) ve bir kontrol ucu(Ein) vardır. Ein=1 olduğunda entegre pasif durumdadır.

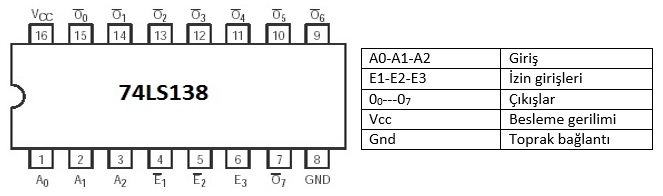


74LS148 doğruluk tablosu

**X değerin 0 veya 1 olmasının fark etmediğini göstermektedir.**

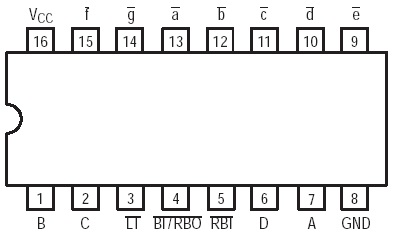
## **Kod Çözücü Entegreler**

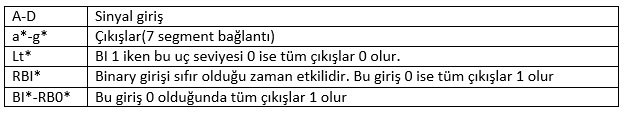
**74LS138 entegresi**  
74LS138 kod çözücü entegresi hafıza sistemlerinde entegre seçiminde ve benzer amaçlarla diğer elektronik devrelerde kullanılır. Demultiplexer elemanı olarak da kullanılabilmektedir.

  
 *74LS138 decoder entegresi ve bacak isimleri*

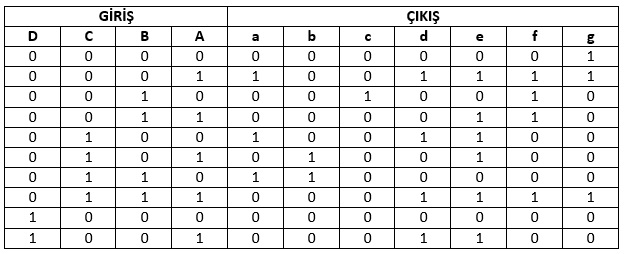
E1 ile E2 uçları lojik-0 ve E3 ucu da lojik-1 olmadığı sürece tüm çıkışlar lojik-1 değerindedir. Bu durumda herhangi bir kod çözme işlemi gerçekleşemez. Bu uçlardan bahsedilen gerekli izin bilgisi girildikten sonra kontrol A0,A1 ve A2 adres girişlerinden yapılır. Bu girişlerden girilen koda göre ilgili çıkış lojik-0 diğer çıkışlarda lojik-1 olur. Değeri 0 olan çıkış aktiftir (Aktif-0), değeri 1 olan çıkış ise pasiftir.

**74LS47 ve 7 segment display**

  
 **74LS47 Decoder entegresi**



Giriş değerlerine göre çıkışta elde edilen 7 led'li göstergeye uygun bilgiler görülmektedir. Entegre çıkışında 7 ledli gösterge değerlerini elde etmek için LT, RBI ve BI/RB0 uçlarının lojik-1’e bağlanması gereklidir.



### **10.Karşılaştırıcı ve Aritmetik İşlem Devreleri**

‘Karşılaştırıcı devreleri’, farklı kaynaklardan gelen bilgileri karşılaştırmak amacıyla düzenlenen devreler olarak düşünebilir. Bileşik lojikte en çok kullanılan devrelerden olan toplayıcı ve çıkarıcı devreler ise, ‘Aritmetik İşlem Devreleri’ olarak isimlendirilir. Karşılaştırıcı ve aritmetik işlem devreleri ‘Kıyaslama Devreleri’ veya ‘Aritmetik Mantık Birimi’ olarak tanımlanır. Diğer bir değişle, kıyaslama devreleri karşılaştırıcılar ve aritmetik lojik devrelerine genelde verilen bir isimdir.

#### **10.1 Karşılaştırıcılar**

#### İki sayıyı karşılaştıran ve büyüklüklerini belirleyen bileşik devreler, **‘büyüklük karşılaştırıcı’** olarak isimlendirilir. Karşılaştırma sonucu; A>B, A=B veya A<B’yi belirleyen üç konum ile belirlenir. En yaygın kullanım yerleri Aritmetik Lojik devrelerdir. Karşılaştırıcı devreleri, girişleri aynı veya farklı iken çıkış veren kontrol devrelerinde ve ikili karşılaştırmanın kullanıldığı adres bulma devrelerinde kullanılır.

En basit karşılaştırıcı devresi, tek bitlik A ve B sayılarının eşitlik durumunu karşılaştıran karşılaştırıcı devresidir. Bu devrede A=B durumunda çıkışlardan birisi ‘1’ olurken, A≠B durumunda diğeri ‘1’ olur.

|  |  |  |  |
| --- | --- | --- | --- |
| **Giri**ş**ler** | | **Çıkı**ş**lar** | |
| **A** | **B** | **A**≠**B** | **A=B** |
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

A

B

A

≠

B

A=B

*Bir bitlik iki sayının karşılaştırması.*

**Örnek 68**

İki bitlik bilgiyi karşılaştıran ve A=B, A>B ve A<B çıkışlarını üreten devreyi tasarlayalım.

Devrenin doğruluk tablosu oluşturulur ve çıkışı temsil eden fonksiyonlar yazılırsa, Şekildeki eşitlikler elde edilir. Elde edilen eşitlikleri temsil eden devrenin çizilmesi ile şekildeki lojik devre oluşur.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | A>B | A=B | A<B |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |

A=B

A>B

A<B

Dört bitlik büyüklük karşılaştırıcı devresi, A girişlerine uygulanan A3A2A1A0 sayıları ile B girişlerine uygulanan B3B2B1B0 sayılarını karşılaştırır. Bütün bitlerde Ai ile Bi değerleri eşitse, iki sayı birbirine eşittir.

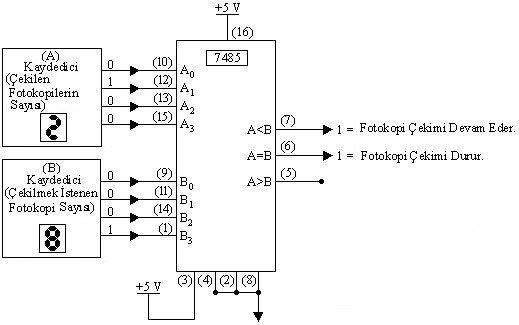
Sayıları karşılaştırma işleminde, önce A3 ve B3 bitlerini karşılaştırır. Eğer A3>B3 ise, diğer bitlerin karşılaştırmasına gerek yoktur. Eğer A3=B3 ise, daha düşük basamak değerine sahip iki bitin karşılaştırılmasına geçilir. Karşılaştırma işlemine, eşit olmayan bir basamak çiftine ulaşıncaya kadar devam edilir. A’nın ilgili hanesi ‘1’ve B’ninki ‘0’ ise A>B sonucuna, A’nın ilgili hanesi ‘0’ ve B’ninki ‘1’ ise, A<B sonucuna varılır. Eğer bütün basamaklardaki değerler birbirine eşitse, A=B sonucuna ulaşılır. Karşılaştırma sonucunda varılan karara göre ilgili çıkış ‘1’ yapılır.

Büyüklük karşılaştırıcılar, bilgisayarlarda (mikroişlemcili sistemlerde) adres kod çözücü devrelerin bir parçası olarak kullanılır. Bu kullanımda, bir karşılaştırıcı bilgisayarın merkezi işlem biriminin ürettiği adres kodu ile bellekte bulunan adresin kodunu karşılaştırır. Karşılaştırılan kodlar eşitse, istenilen işlemi yapmak üzere ilgili devreyi aktif hale getirir.

|  |  |  |
| --- | --- | --- |
| **A3,B3 A2,B2 A1,B1 A0,B0** | **IA>B IA<B IA=B** | **QA>B QA<B QA=B** |
| A3>B3 X X X  A3<B3 X X X  A3=B3 A2>B2 X X  A3=B3 A2<B2 X X  A3=B3 A2=B2 A1>B1 X  A3=B3 A2=B2 A1<B1 X  A3=B3 A2=B2 A1=B1  A0>B0  A3=B3 A2=B2 A1=B1 A0<B0  A3=B3 A2=B2 A1=B1  A0=B0  A3=B3 A2=B2 A1=B1 A0=B0  A3=B3 A2=B2 A1=B1  A0=B0  A3=B3 A2=B2 A1=B1 A0=B0  A3=B3 A2=B2 A1=B1 A0=B0 | X X X  X X X  X X X  X X X  X X X  X X X  X X X  X X X  1 0 0  0 1 0  X X 1  0 0 0  1 1 0 | 1 0 0   1. 1 0 2. 0 0 3. 1 0 4. 0 0 5. 1 0 6. 0 0 7. 1 0 8. 0 0   0 1 0   1. 0 1 2. 1 0   0 0 0 |

*Dört bitlik karşılaştırma işlemi doğruluk tablosu.*

Karşılaştırıcı devresine örnek olarak şekildeki fotokopi makinası kontrol devresini verebiliriz. 7485 entegresi kullanılarak oluşturulan devrede, çekilmek istenen fotokopi sayısını temsil eden değer ‘B’ girişine, çekilen fotokopileri sayan devrenin çıkışı ise ‘A’ girişine uygulanır. İki sayı birbirine eşit oluncaya kadar A<B çıkışı ‘1’ olur ve fotokopi çekimi devam eder. A ve B girişlerindeki değerlerin aynı olması durumunda; ‘A=B’ çıkışı ‘1’ olur ve fotokopi çekim işlemi durur.



*Fotokopi makinası kontrol devresi.*

#### **10.2 Aritmetik İşlem Devreleri**

Toplama, çıkarma, çarpma, bölme işlemlerini yapan devrelere**, ‘Aritmetik İşlem Devreleri’** denir. Bilgisayarlarda ve hesap makinalarında, temel işlemler toplama ve çıkartma işlemleridir.

**Çarpma işlemi;**

toplama işleminin tekrarlanması,

Bölme işlemi ise;

çıkartma işleminin tekrarlanması ile yapılır.

##### **10.2.1 Toplayıcı Devreleri**

##### Bilgisayarlar ve hesap makinaları, her biri çok sayıda bite sahip iki adet ikili sayıyı toplama işlemini gerçekleştirirler. En basit toplama işlemi dört olası temel işlemi içerir.

**0+0=0,**

**0+1=1,**

**1+0=1,**

**1+1=10, (Elde 1, Toplam = 0)**

İlk üç işlemde tek basamaklı bir sayı elde edilirken, son işlemde ikinci basamak ortaya çıkar ve ikinci basamak **‘elde biti’** (carry bit) olarak isimlendirilir. İki biti toplayan devreler ‘**yarım toplayıcı**’ olarak, üç bitin toplamını yapan devreler ise ‘**tam toplayıcı’** olarak isimlendirilir. Yarım toplayıcı terimi, tam toplayıcıyı oluşturmak için iki tane yarım toplayıcı kullanılmasından gelmektedir.

### **10.2.1.1 Yarım Toplayıcı**

Girişine uygulanan iki biti toplayıp, sonucu toplam ve elde şeklinde veren toplayıcı devresi, ‘**yarım toplayıcı**’ olarak isimlendirilir. Yarım toplayıcı devresi, doğruluk tablosundan elde edilen fonksiyonların lojik devresinin çizilmesi ile oluşturulur. Oluşan devrede, ‘Toplam’ ve ‘Elde’ değerlerini temsil eden iki çıkış bulunur.

|  |  |  |  |
| --- | --- | --- | --- |
| **A** | **B** | **S** | **Co** |
| 0  0  1  1 | 0  1  0  1 | 0  1  1  0 | 0  0  0  1 |

A S (Toplam)

HA

B

Co (Elde)

*Yarım toplayıcı sembolü ve doğruluk tablosu.*

Yarım toplayıcı çıkışlarındaki sadeleştirilmiş fonksiyonlar, S = A'B+AB' ve C = AB şeklinde elde edilir. Girişlerin A ve B, çıkışların S ve C değişkenleri ile ifade edildiği yarım toplayıcı devresi, bir ‘Özel-VEYA’ ve bir ‘VE’ kapısıyla oluşturulabilir.

S

C

A

B

Toplam

Elde

S=A

⊕

B

C=A.B

S=AB’+A’B

C=A.B

A

B

A

B

A

B

Elde

Toplam

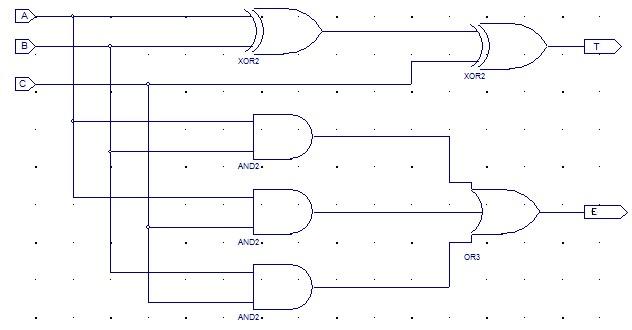
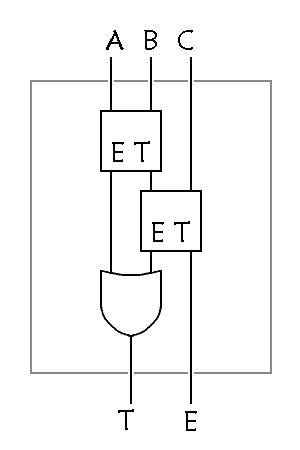
*Yarım toplayıcı devresi lojik şemaları.*

### **Tam Toplayıcı (Full Adder - FA)**

3 [bitlik](http://bilgisayarkavramlari.com/2007/11/24/bit-ikil/) giriş değerlerinin (iki tabanındaki girişler) toplamını veren devredir. Buna göre A, B ve C girişleri için aşağıdaki tablo elde edilir.

A B C E T  
   
0 0 0 0 0  
0 0 1 0 1  
0 1 0 0 1  
0 1 1 1 0  
1 0 0 0 1  
1 0 1 1 0  
1 1 0 1 0  
1 1 1 1 1

verilen çizelgede A, B ve C sayılarının toplamı E ve T değerlerinde gösterilmiştir.

**Bir tam toplayıcının özelveya devreleri (xor) ile tasarımı**  
[](http://bilgisayarkavramlari.com/wp-content/uploads/2007/12/tamtoplayicidevre.jpg)  
**Bir tam toplayıcıyı iki adet**[**yarım toplayıcı**](http://bilgisayarkavramlari.com/2007/12/08/yarim-toplayici-half-adder/)**ile tasarlamak da mümkündür.**  
[](http://bilgisayarkavramlari.com/wp-content/uploads/2007/12/tamtoplayici_yarimtoplayiciile.jpg)

Tam toplayıcı aynı zamanda [kod çözücü (decoder)](http://bilgisayarkavramlari.com/2007/12/09/kod-cozucu-decoder/) ve [çoklayıcı (multiplexer)](http://bilgisayarkavramlari.com/2007/12/09/coklayici-multiplexer/) ile de tasarlanabilir.

#### **10.2.2 Çıkarıcı Devreleri**

Lojik devrelerde yapılan ikinci temel işlem çıkarmadır. İki bitin çıkarmasını yapan devreye ‘**yarım çıkarıcı’**, üç bitin çıkarmasını yapan devreye ise ‘**tam çıkarıcı’** devresi denir.

Yarım

Çıkarıcı

A

B

Fark

(

D

)

Borç(B)

|  |  |
| --- | --- |
| Girişler | Çıkışlar |
| A B | Fark Borç  A-B (B) |
| 0 0   1. 1 2. 0   1 1 | 1. 0 2. 1   1 0  0 0 |

D= AB + AB = A + B

B = A B

*Yarım çıkarıcı sembolü ve doğruluk tablosu.*

### **10.2.2.1 Yarım Çıkarıcı Devresi**

İki bitin çıkarması işlemini yapan çıkarıcı devresinde, iki giriş ve iki çıkış bulunur. Çıkışlardan birisi sayının farkını, diğeri borç bitini gösterir. İki bitin çıkarılması işleminde dört farklı durum oluşur;

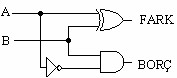
**0– 0 = 0**

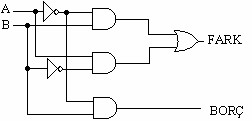
**1– 0 = 1**

**1 – 1 = 0**

**0 – 1 = 1 ( Borç 1)**

A-B işleminde A<B olduğunu zaman ‘0–1’ işlemi oluşur ve bu durumda bir yüksek değerli basamaktan ‘1’ borç alınır. Borç çıkışı, doğruluk tablosunda ayrı bir sütun olarak gösterilir. Yarım toplayıcı devresinde oluşan işlemlerin doğruluk tablosu ve doğruluk tablosuna göre oluşan fonksiyonlar şekilde gösterilmektedir.



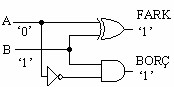
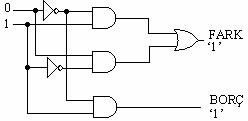


*Yarım çıkarıcı lojik devreleri.*

**Örnek 69**

Yarım çıkarıcı devreleri ile 0-1 işlemini yapalım.

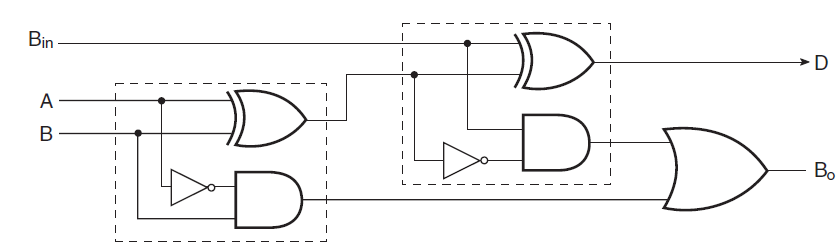
Yarım toplayıcı devre girişlerine A=0 ve B=1 değerleri uygulanırsa şekildeki çıkış değerleri elde edilir.



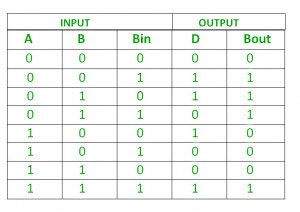
### **Tam Çıkarıcı Devresi**

Daha düşük değerli basamak tarafından ‘1’ borç alınmış olabileceğini dikkate alarak iki biti birbirinden çıkaran bileşik devre, ‘**tam çıkarıcı**’ olarak isimlendirilir. Üç giriş ve iki çıkışa sahip tam çıkarıcı devresinde girişler; çıkarılan, çıkan ve borcu gösterirken, çıkışlardan biri farkı diğeri borcu gösterir.

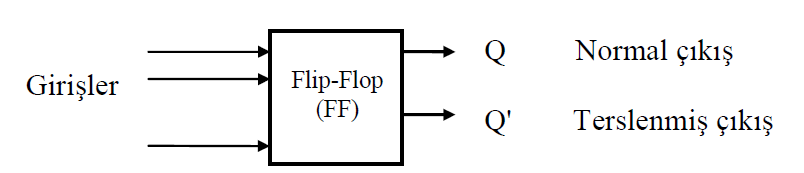
Tam çıkarıcı devre aynı tam toplayıcı gibi iki değeri almanın yanı sıra dışarıdan ödünç değerini de almaktadır. Böylelikle çok basamaklı ikilik sayıları çıkarma imkanı olur. Tam çıkarıcı devrenin devre şeması aşağıdaki gibidir;

[](http://www.lojikprob.com/elektronik/dijital-elektronik-9-birlesik-mantik-devreleri/attachment/full_sub1/)

Burada iki adet yarım çıkarıcı devrenin beraber kullanıldığını fakat NOT kapısının farklı girişlerde uygulanmıştır. Tam çıkarıcının doğruluk tablosu aşağıdaki gibidir;

[](http://www.lojikprob.com/elektronik/dijital-elektronik-9-birlesik-mantik-devreleri/attachment/full-subtractor-truth-table/)

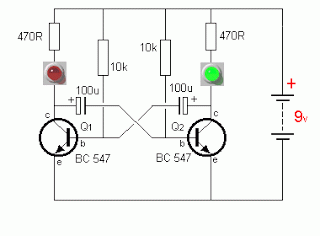
## **Flip Flop**

Flip-flop (FF)’lar girişlerine uygulanan sinyal ile çıkışları iki kararlı olan, elektronik elemanlardır.  
Devreye çalışma gerilimi uygulandığı sürece durumu ve buna bağlı olarak çıkışındaki değeri devamlı olarak koruyabilen multivibratörlere **flip – flop** denir.  
Yapısında lojik kapılar bulunduran, değişken giriş gerilimlerine karşılık farklı çıkış değerlerine sahip olan elemanlardır.  
Flip-flopların bu anlık değişimine**tetiklenme** denmektedir.  
SET (1) ve Reset (0) olmak üzere iki kararlı duruma sahiptir. Bu devre elemanları bu iki kararlı durumdan birisinde kalırlar. Bu nedenle flip-floplar depolama elemanı olarak kullanılırlar.  


Girişlerine uygulanan sinyal değiştirilmediği sürece çıkış durumunu korurlar ve 1 bitlik bilgiyi saklayabilirler.  
Giriş sinyaline göre çıkış Lojik-0 ya da Lojik-1 olur.  
Flip-floplar, belleklerde kullanılan sayıcılarda, kaydedicilerde ve ardışık devrelerde kullanılmaktadır.  
Ardışıl devrelerde bir önceki çıkış, var olan girişlerle birlikte sonraki çıkışı belirler. Bu devrelerin bellek özelliği vardır.  
Yani flip-flopların çıkış değerleri bir önceki çıkışa mutlaka bağlıdır.

## **Flip – Flop Devresi Nasıl Çalışır?**

Devre kondansatörlerin dolarak ters yönde voltaj vermesiyle normalde doyumda çalışan (aktif olan) olan NPN transistörlerin beyzine ters voltaj vererek ledlerin GND ile bağlantısını kesmesiyle oluşan devredir. Devredeki ledlerin yanıp sönme süreleri kondansatörlerle alakalıdır. Daha yüksek faradlı kondansatörler kullanırsak yanıp sönme zamanı uzayacaktır.



Transistörleri iletime sokacak bir gerilim olmadığında LED’ler yanmaz. Kondansatörler normalde boştur. Beslemeyle birlikte kondansatörler çok akım çekmektedirler. İki kondansatörden biri diğerinden daha önce dolar. Üzerinde bulunan gerilimle de bağlandığı transistörü iletime geçirecek ve baz akımından dolayı kondansatörü boşaltacaktır. Bunların ışık verme süreleri direnç ve kondansatörlere bağlıdır. Daha sonra bu sefer diğer kondansatör transistör ve led üçlüsü bu olayı tekrarlar. Ve Flip-Flop devresi tamamlanmış olur.

**11.2 Flip Flop Türleri**

* R-S flip flop (reset-set flip flop  )
* Tetiklemeli (clocked) R-S flip flop
* J-K flip flop
* Master Slave flip flop
* D  flip flop (data flip flop)
* T flip flop (Toggle flip flop)

#### **11.2.1. RS Flip-Flop Devresi**

İki çıkışa sahip FF’de, iki farklı çıkış durumu ortaya çıkar: Q=0, Q'=1 ve Q=1, Q' =0 durumları. Q=0 ve Q'=1 olduğu durum, çıkış ‘0’ veya **‘reset-sıfırla’** durumu olarak tanımlanır. Oluşabilecek ikinci durumunu ifade eden Q=1 ve Q'=0 durumu, çıkış ‘1’ veya **‘set-kur’** durumu olarak adlandırılır.

Bu durumda, FF’nin Q=0 olmasını sağlayan girişi ‘Reset-R’, Q=1 durumunu oluşturan girişi ise ‘Set-S’ olarak düşünülebilir. Çıkışların birbirinin tersi olduğu bu durumlar, normal çalışma durumları olarak kabul edilir. FF girişlerinin ‘set-kur (S)’ ve ‘reset-sil (R)’ olarak isimlendirilmesi ile, R-S FF olarak adlandırılan FF türü ortaya çıkar. RS tipi FF; Q=1 ve Q'=0 iken ‘**set’** (kurma), Q=0 ve Q'=1 olduğunda ise ‘**reset’** (silme) durumundadır.

Temel FF devresi, iki **‘VEDEĞİL’** veya iki **‘VEYADEĞİL’** kapısı ile gerçekleştirilebilir.

**‘VEDEĞİL’** kapılarıyla yapılan devre **‘VEDEĞİL latch’** veya kısaca **‘lanch’** olarak, **‘VEYADEĞİL’** kapıları kullanılarak oluşturulan devre ise **‘VEYADEĞİL lanch’** diye isimlendirilir. Her iki tip **‘lanch’** devresinde, kapılardan birinin çıkışı diğerinin girişine çapraz şekilde bağlanmıştır. Q ve Q' olarak isimlendirilen kapı çıkışları, **‘lanch’** çıkışlarıdır.

**‘VEYADEĞİL’** kapısının girişlerinden birisinin **‘1’** olması, çıkışının **‘0’** olması için yeterlidir. Her iki girişin **‘0’** olması durumunda, çıkış **‘1’** olur.

**‘VEYADEĞİL’** kapılarıyla oluşturulan FF devresinde, FF’nin durumu değiştirilmediği sürece her iki giriş ‘0’ değerine sahiptir.

#### **11.2.2 Tetiklemeli - Saatli R- S Tipi Flip- Flop**

FF’nin konumunun değiştirilmesi işlemi, **‘tetikleme’** olarak adlandırılır. Tetikleme girişinin bulunmadığı, FF durumlarının kontrol girişlerindeki bilginin değişmesi anında değiştiği FF’ler ‘Asenkron FF’ler’ olarak isimlendirilir.

Senkron sistemlerde herhangi bir çıkışın değerinin değişebileceği zamanlar, **‘tetikleme sinyali’** (clock - saat) adı verilen, kare veya dikdörtgen şeklindeki sinyal tarafından belirlenir. Tetikleme sinyali olarak kullanılan sinyalin aktif durum (kenar) değişimi 0’dan 1’e ise buna **‘pozitif kenar de**ğ**i**ş**imi’**, aktif durum değişimi 1’den 0’a doğru ise **‘negatif kenar de**ğ**i**ş**imi’** (NGT) denir.

**‘Tetikleme sinyali’** olarak isimlendirilen sinyal, tetiklemeli FF’lerde bulunan üçüncü girişe uygulanır. Tetikleme işleminin oluşması için gerekli tetikleme sinyalini FF’ye uygulamak amacıyla FF’de oluşturulan üçüncü giriş, **'tetikleme giri**ş**i'** olarak adlandırılır. Tetikleme girişi, ‘clk’ harfleri ile gösterilir ve **‘clk giri**ş**i’** olarak da isimlendirilir. Tetikleme girişinin eklenmesiyle senkron ardışık devre durumuna gelen FF devresinde, R-S girişlerinin değişmesi çıkışları hemen değiştiremez. Çıkışların değişmesi, tetikleme girişine uygulanan işarete bağımlıdır.

Tetiklemeli RS FF’ler, RS FF devresinin girişlerine kapı devreleri eklemek suretiyle elde edilir. Tetikleme girişi eklenmiş R-S FF devresinin sembolü ve **‘VEDEĞİL’** kapılarıyla tetikleme girişi oluşturulması işlemi şekilde görülmektedir. Yapılan işlem, RS FF’nin girişlerine 3. giriş oluşturacak şekilde **‘VEDEĞİL’** kapıları bağlanmasıdır.

#### **J-K Flip-Flop (JK FF)**

J-K Flip-Flop, RS FF’lerin belirsizlik durumunu ortadan kaldırmak amacıyla geliştirilmiş FF çeşididir. Tanımsız durumların tanımlı hale geldiği J-K FF’lerde, J ve K girişleri FF’i kurmak ve silmek için S ve R girişleri gibi davranır. J girişinin S’ye, K girişinin R’ye denk geldiği J-K FF’lerde; J=K=1 durumunda iken, tetikleme sinyalinin her pozitif kenarı ile çıkış bir önceki durumun tersi değerini alır. Yani Q=1 ise 0’a, Q=0 ise 1’e geçiş olur. Bu işlem, **‘toggle’** (ters çevirme) olarak isimlendirilir. Bu işlemin devamlı gerçekleştirilmesi için, J=1 ve K=1 durumunda bırakılır.

#### **11.2.4 D Tipi Flip-Flop (D FF)**

Temel RS FF’den üretilen diğer bir FF çeşidi, tek bir senkron kontrol girişine sahip olan D (Data) tipi FF devresidir. D tipi FF’de oluşan işlemler basittir, Q çıkışı tetikleme sinyalinin gelmesi ile D kontrol girişinin sahip olduğu değeri alır. Bunun anlamı; D=0 iken, tetikleme sinyalinin gelmesi ile Q çıkışı ‘0’ değerine sahip olur demektir. D=0 durumu devam ettiği sürece, **‘Clk’** sinyalinin durumu değişse bile Q=0 değerini korur. D=1 değerini alması durumunda, ilk gelen tetikleme sinyalinin pozitif kenarında Q=1 değerini alır. Q çıkışının yalnızca tetikleme sinyalinin pozitif kenarlarında durum değiştirir.

D tipi FF devresi, RS FF’nin girişine **‘DEĞİL’** kapısı bağlanarak elde edilebilir. Eklenen **‘DEĞİL’** kapısı, hem RS FF’lerde belirsiz durum oluşturan R=S=1 durumunu ortadan kaldırır, hem de çıkışın D girişini takip etmesini sağlar.

#### **11.2.5 Ana Uydu (Master-Slave) Flip-Flop**

Ana-Uydu (Master-Slave) tipi FF devresi, iki RS Flip-Flop ve bir **‘DEĞİL’** kapısı ile oluşturulur. Devrede kullanılan FF’lerden birisi ana devreyi, diğeri ise ana devreye bağlı olarak çalışan uydu FF’yi oluşturur.

Ana-Uydu FF devresi, **‘Clk’** girişi ‘1’ olduğunda **‘Ana FF’** devresi, ‘0’ olduğunda **‘Uydu FF’** devresi çalışacak şekilde düzenlenmiştir.

Tetikleme girişine ‘0’ uygulandığı zaman, **‘DEĞİL’** kapısının çıkışı uydu FF’nin **‘Clk’** girişini ‘1’ yapar. Bu durumda Q çıkışı Y’ye, Q' çıkışı da Y çıkışına eşittir. Bu anda ana FF’nin tetikleme girişi ‘0’ olduğundan bu FF çalışmaz ve bir önceki konumu korur. Bunun anlamı; uydu FF’nin Q ve Q' çıkış değerlerinin, ana FF’nin bir önceki değerlerine sahip olmasıdır.

Tetikleme girişinin '1' yapılması durumunda, ana FF etkin duruma gelirken, uydu FF etkisizdir. Ana FF’nin çıkışları, girişlerin durumuna göre değerler alır. Devrenin çıkışı, uydu FF’nin durumuna eşittir.

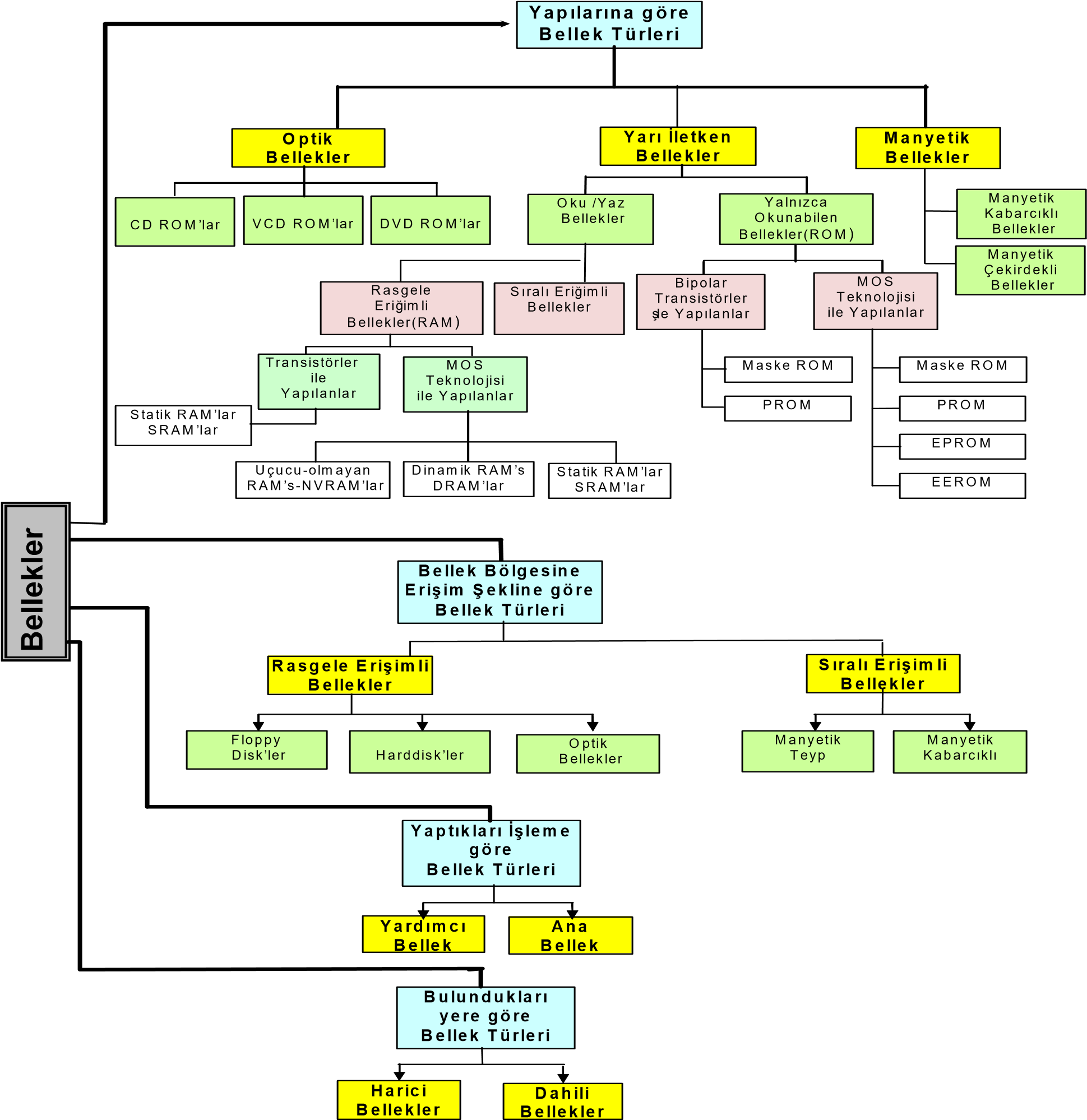
Ana-Uydu Flip-Floplar genelde negatif kenar tetiklemeli olarak çalışırlar. RS Tipi FF’lerle oluşturulan Ana-Uydu FF’nin diğer FF çeşitleri ile oluşturulması mümkündür.

**12. Hafıza Devreleri**

Bilgisayarlarda ve programlar ile çalışan endüstriyel devrelerde kullanılan programların veya programların çalıştırılması sırasında işlenen verilerin ikili bir yapıda saklanması amacıyla kullanılan devreler veya elemanlar, **‘bellek / hafıza devreleri’** olarak isimlendirilir. Diğer bir deyişle; bilgilerin geçici veya sürekli olarak saklandığı birimler ‘bellek devreleri’ olarak isimlendirilir. Bilgisayarın bir elemanı olduğu dijital sistemlerin analog sistemlere üstünlüklerinden birisi, büyük hacimli bilgileri küçük alanlarda uzun süreli saklayabilme özelliğidir.

Bilgisayar merkezi işlem birimi ile doğrudan irtibatlı, bilgisayara dahili olarak yerleştirilen, temelde yarı iletken elemanların oluşturduğu bellek devreleri; yerleştirildikleri yer olarak **‘dahili bellek’**, yaptıkları işlem nedeniyle **‘ana bellek’** olarak isimlendirilir.

Çok büyük hacimli bilgilerin saklanması amacıyla kullanılan ve genelde bilgisayarın dışında oluşturulan bellek elemanları, yerleştirildikleri yer nedeniyle **‘harici bellek’** olarak isimlendirilirken, yaptıkları işlem açısından **‘yardımcı / yedek bellek’** olarak isimlendirilirler. Yardımcı bellekte saklanan bilgiler, bilgisayarın ihtiyacı olduğu anlarda bilgisayara yüklenerek kullanılırlar.



**12.1 Belleklerde Kullanılan Terimler Ve Yapılan Temel İşlemler**

**Bit:** ‘0’ veya ‘1’ değerini alabilen ikili sayı olarak isimlendirilir. İkilik sayı, sayısal sistemlerde en küçük bilgi birimidir.

**Nibble**: Dört bitin bir araya gelerek oluşturduğu bilgi grubu **‘nibble’** olarak isimlendirilir. Bir bayt, dört bitlik iki gruba ayrılabilir ve her grup **‘nibble’** olarak adlandırılır.

**Bayt (Byte):** 8 bitlik bilginin açıklanması için kullanılan özel terim.

Bazı kaynaklarda, nadiren de olsa, bayt terimi farklı uzunluklardaki bitleri tanımlamak için kullanılsa da, genel kabul baytın 8 bit olduğudur.

**Bellek kelimesi (Memory Word):** Aynı tip verileri veya bir komutu temsil etmek amacıyla kullanılan bir bellek içerisindeki bir grup bittir.

16 bitlik bir kelimeyi saklamak amacıyla kullanılan 16 FF’nin oluşturduğu kaydedici, bellek kelimesi için örnek olabilir. Bellek kelimesi uzunluğu, bilgisayarların ölçüsüne bağlı olarak 4 bit– 64 bit arasında değişir. Bununla beraber genel kabul, 16 bit yani iki baytlık bilginin kelime (word) olarak isimlendirilmesi şeklindedir.

**Bellek** **Hücresi (Memory Cell):** Tek bir bitlik bilgiyi (0 veya 1) saklayabilmek için kullanılan eleman, devre veya cihazdır.

**Bellek hücresine örnek olarak; bir FF, şarj edilmiş bir kondansatör, manyetik teyp veya disketteki tek bir spot verilebilir.**

**Bellek Dizisi (Memory Array):** Bellek hücrelerinin bir araya gelerek oluşturduğu yapıdır.

Bellek hücreleri dizileri oluştururken farklı kombinasyonlarda bir araya gelebilirler. Bellek dizilerinde, hücrelerin yan yana bir araya gelerek oluşturduğu yapı **‘satır’**, düşey doğrultuda bir araya gelerek oluşturduğu yapı **‘sütun’** olarak isimlendirilir.

**Kapasite (Capacity):** Tüm bellek sisteminde veya belirli bir elemanda saklanabilecek bit sayısını belirtmek için kullanılan terimdir.

**Adresler (Address):** Bellek dizisinde bir hücrenin veya kelimenin bulunduğu bölgeyi / yeri ifade eden sayıdır.

**Yollar / Taşıtlar (Buses):** Mikroişlemcili bir sistem içerisindeki birimler veya mikroişlemcili sistem ile çevre elemanları arasında iletişimi sağlayan hatlardır.

**Okuma İşlemi (Read Operation):** Belirli bir bellek adresinde saklanan ikili bilginin belirlenip, farklı bir elemana aktarılması işlemidir.

**Yazma İşlemi (Write Operation):** Belirli bir bellek bölgesine yeni bir bilginin yerleştirilmesi veya bir adres bölgesinde saklanan bilgilerin yeni bilgilerle yer değiştirmesi işlemidir.

**Erişim Zamanı (Access Time):** Bir bellek elemanının işlem hızı ölçüsüdür ve bir okuma işlemini gerçekleştirmek için gerekli toplam zamanı ifade eder. Diğer bir deyişle; bir belleğin yeni bir adres bilgisi alması ile verinin bellek çıkışında hazır olduğu durum arasında geçen süre, **‘Erişim zamanı - tACC’** olarak isimlendirilir.

**Uçucu Bellek (Volatile Memory):** Saklama işlemi için elektrik enerjisinin gerektiği bellek tipidir. Elektrik kaynağının kesilmesi ile bellekte saklanan tüm bilgi silinir. Yarı iletken belleklerin çoğu uçucu iken, tüm manyetik bellekler uçucu olmayan özelliğe sahiptirler.

**Rastgele Erişimli Bellek (Random Access Memory - RAM):** Okuma veya yazma işlemi sırasında bellek bölgesine erişim için, çalışılan bellek bölgesinin konumunun hiçbir etkisinin olmadığı bellek türüdür. Yani; tüm bellek bölgeleri için okuma veya yazma işlemi erişim zamanının aynı olduğu bellek çeşididir. Yarı iletken ve manyetik belleklerin çoğu RAM’dır.

**Sıralı Erişimli Bellek (Sequential Access Memory – SAM):** Erişim zamanının çalışılan bellek bölgesinin konumuna bağlı olarak değişim gösterdiği bellek türüdür. Bir bilgi, bilginin saklandığı adrese kadar ki tüm bellek bölgelerinin sırası ile kat edilmesi sonucunda bulunur. Bu işlem, RAM tipi belleklere göre çok daha uzun bir erişim zamanı gerektirir.

**Oku / Yaz Bellekler (Read / Write Memory - RWM):** Okuma ve yazma işlemlerinin benzer işlem aşamaları ile gerçekleştirildiği her hangi bir bellek türüdür. Bilgilerin yazılıp, daha sonra okunabildikleri tüm bellek çeşitlerine verilen genel addır. RAM bellekler bu gruba girer.

**Yalnızca Okunabilen Bellekler (Read Only Memory – ROM):** Yazma işleminin yalnızca bir kere ve üreticiler tarafından gerçekleştirildiği bellek türüdür. ROM bellekteki bilgiler istenilen sayıda okunabilir. Yazma işleminin bir kereden fazla gerçekleştirilebildiği ROM tiplerinde; yazma işlemi okuma işlemine göre çok daha karmaşık olduğundan, yazma işlemi çok sık olarak yapılmaz. Çok değişik türlerin bulunduğu ROM belleklerin hepsi uçucu olmayan tiptedir ve elektrik enerjisi kesilse dahi bilgiler saklanmaya devam eder.

**Statik Bellek Elemanları (Static Memory Devices):**Saklanan bilginin elektrik enerjisi uygulandığı sürece yeniden yazmaya (tazelemeye) gerek olmadan saklandığı yarı iletken bellek elemanlarıdır.

**Dinamik Bellek Elemanları (Dinamic Memory Devices):** Saklanan bilginin uzun süre sabit olarak saklanamadığı, elektrik enerjisi uygulanması yanında, bilginin belirli aralıklarla tazelenmesini (refresh) gerektiren bellek türüdür.

**Dahili / Ana Bellek (Internal / Main Memory):** Komutların ve CPU’nun üzerinde çalıştığı verilerin saklandığı, bilgisayarın ana belleğidir (Main memory). Dahili bellekler, bilgisayarlarda bulunan en hızlı belleklerdir ve genellikle yarı iletken elemanlarından yapılırlar. Bilgisayarlarda bulunan RAM bellekler, dahili belleklerdir.

**Harici / Yedek Bellek (External or Auxiliary Memory):** Genellikle bilgisayarın dışında bulunan ve yavaş çalışan bellek türüdür. Yardımcı bellek olarak da isimlendirilen bu tip bellekte, dahili belleğe göre çok daha büyük hacimli bilgiler saklanabilir. Dahili belleklere göre daha yavaş olan bu bellekler, uçucu olmayan özelliktedirler. Manyetik teypler ve diskler yaygın olarak kullanılan yardımcı bellek türleridir.

**Adres Girişleri:** Bellek 32 kelimeyi depoladığından dolayı, 00000’dan 11111’e kadar ki ikili sayılarla ifade edilen 32 farklı adrese, diğer bir deyişle 32 bilgi saklama bölgesine sahiptir. Adres bölgelerinden herhangi birisine erişmek için, A0-A4 ile temsil edilen adres girişlerine 5 bitlik adres kodu uygulanır.

**R/W′ Girişi:** R/W′ giriş hattı, bellekte yapılacak oku / yaz işlemini belirler. Bazı entegrelerin ayrı-ayrı girişler ile belirlediği bu işlemler, tek bir girişin kullanıldığı durumlarda; R/W**I**=1 olması ile okuma işlemi, R/W**I**=0 olması ile yazma işlemi gerçekleştirir.

**Bellek Yetkilendirme:** Birçok bellek sistemi, sistemin tümünün veya bir kısmının girişlere karşı tepkisiz kalmasını sağlayan bellek yetkilendirme (Memory Enable-ME) girişine sahiptir. Farklı bellek entegrelerinde farklı isimlerle temsil edilen bu girişe uygun sinyal uygulanması ile, belleğin R/W**I** ve adres girişlerine karşı tepkisiz olması sağlanır. Yetkilendirme girişi, belleklerin bir araya gelerek büyük bellek sistemlerini oluşturduğu düzeneklerde önemli bir işleve sahiptir.

**Adres Yolu (Address Bus)**: Tek yönlü bu adres yolu ile, CPU’nun ikili adres çıkışları bellek entegresine taşınır.

**Veri Yolu (Data Bus):** Çift yönlü bu veri yolu ile, CPU ile bellek arasında veri akışını sağlar.

**Kontrol Yolu (Control Bus):** Kontrol yolu ile, kontrol sinyalleri (R/W**I**, ME, vb.) CPU’dan bellek entegrelerine taşınır.

**Örnek 70**

5Mx8 ve 1Mx16 şeklinde ifade edilen belleklerin, hangisinde daha çok bilgi saklanabileceğini bulalım.

**5Mx8 = 5x1.048.576x8 = 41.943.040 bit**

**1Mx16 = 1.048.576x16 = 16.777.216 bit**

Bu durumda, 5Mx8 kapasiteli bellek daha çok bilgi saklayabilir.

## **12.2. Bellekleri Sınıflandırılması – Bellek Çeşitleri**

Bellekler yapılarına göre;

* Manyetik bellekler,
* Optik bellekler,
* Yarı iletken bellekler,

olarak üç grup altında incelenebilir.

## **12.2.1 Manyetik Bellekler**

Manyetik bellekler, saklama işleminin manyetik olarak gerçekleştirildiği ve bilgilerin geçici olmayacak şekilde tutulduğu bellek türleridir.

### **12.2.2 Optik Disk Bellekler**

Optik disk bellekler, yeni bir bellek teknolojisidir ve piyasada etkin olmaya aday gözükmektedir. Çalışma prensibi, çok ince lazer ışınının disk üzerinde yansıması ve **‘scattering’** işlemine dayanır. Disket üzerinde yanma olayı sonucu oluşan mikroskobik çukurlar ‘1’ değerini, düzlükler ise ‘0’ değerini temsil etmek için kullanılır. Spiral olarak hareket eden izler ile temsil edilen veriler, eski plakalarda olduğu gibi, lazer ışınları tarafından pikaplardaki iğnenin sesi okuması gibi okunur. Okunan bilginin yönü içeriden dışarıya doğrudur.

### **12.2.3 Yarı İletken Bellekler**

Bilgisayarlarda ana bellek olarak kullanılan ve çeşitli entegreler şeklinde imal edilen yarı iletken bellekleri başlıca iki grupta toplanabilir;

* Yalnız okunabilen bellekler (Read Only Memories – ROM)
* Okunabilen / yazılabilen bellekler (Read / Write Memories – RWM)

**Yarı**

**İletken**

**Bellekler**

Oku /Yaz

Bellekler

Rastgele

Erişimli

Bellekler (RAM)

Transistörler

ile

Yapılanlar

S

O

M

Teknolojisi

Yapılanlar

ile

Statik RAM’lar

SRAM’lar

Dinamik RAM’lar

DRAM’lar

Uçucu-olmayan

RAM’lar-NVRAM’lar

Statik RAM’lar

SRAM’lar

Yalnızca

Okunabilen

Bellekler (ROM)

Bipolar

Transistörler

ile Yapılanlar

Maske ROM

PROM

Maske ROM

PROM

EPROM

EEROM

MOS

Teknolojisi

ile Yapılanlar

Erişimli

Sıralı

Bellekler

**12.2.3.1 Yalnızca Okunabilen Bellekler (ROM’lar)**

Adından da anlaşılacağı üzere, yalnızca bilgi okunması yapılabilen belleklerdir. Bilginin sabit olarak tutulması istenilen ve sık olarak değişmesine gerek olmayan durumlarda / yerlerde kullanılırlar. Bilgisayarlar, yazar kasalar, güvenlik sistemleri ve ev aletleri ROM’ların kullanım yerlerine örnek olarak verilebilir.

ROM belleklere bilgi, üretici tarafından üretim sırasında veya daha sonra elektriksel olarak yüklenir. Yüklenen bilgiler uçucu değildir ve elektrik kesilse dahi silinmez. Bilgi yükleme işlemi, ROM’un programlaması veya yakılması olarak isimlendirilir.

### **12.2.3.2 Yarıiletken Oku / Yaz Bellekler (RWM’lar)**

Lojik bir devreye yerleştirildikten sonra sonsuz sayıda okuma veya yazma işleminin gerçekleştirebildiği bellekler, **‘oku / yaz (RWM) bellekler’** olarak isimlendirilir. Bilgilerin sabit olarak kaydedildiği ROM’lardan farklı olarak, oku/yaz belleklerde bilgiler geçici olarak saklanırlar. Bir bellek bölgesine bilginin saklanması ‘yazma’ işlemine karşılık gelirken, bir bellek bölgesinde saklanan bilginin alınması işlemi ‘okuma’ işlemini temsil eder.

Oku/yaz bellekler genel olarak, sıralı erişimli bellekler ve rasgele erişimli bellekler olarak iki gruba ayrılır. RWM’ler daha çok rasgele erişimli bellekler (RAM) olarak tanınırlar.

KAYNAKÇA

* http://megep.meb.gov.tr/mte\_program\_modul/moduller\_pdf/Temel%20Mantık%20Devreleri.pdf
* https://teslaakademi.com/mantik-devreleri
* https://merttopuz.com
* https://www.elektrikport.com
* http://kisi.deu.edu.tr//ozlem.karaca/sunumlar/lojik\_devreler.pdf
* https://www.rapidtables.com/calc/math/base-calculator.html
* https://www.learnabout-electronics.org/Digital/dig21.php
* https://medium.com/@mehyalas/dijital-sayı-sistemleri-bedf99f6b8d0
* https://slideplayer.biz.tr/slide/10194885/
* https://dilekylmzr.files.wordpress.com
* http://bilgicelbesi.blogspot.com
* https://abs.mehmetakif.edu.tr
* https://www.elektrikport.com
* https://diyot.net