Bölüm 6. KOMBİNASYONEL DEVRELER

- **Toplayıcılar**
  - **Yarım Toplayıcı**
  - **Tam Toplayıcı**
  - **Paralel Toplayıcılar**

- **Karşılaştırıcılar**

- **Kod Çözücüler**

- **Kodlayıcılar**

Toplayıcılar

Yarım Toplayıcı (Half Adder - HA):

Daha önce ifade edildiği gibi ikili toplama işlemi şu şekildeydi;

0+0 = 0,

0+1 = 1,

1+0 = 1,

1+1 = 10 (buradaki 1, eldeyi temsil etmektedir)

Tek bit üzerinde yerine getirilen bu işlemi icra eden devreler yarım toplayıcı olarak bilinir. Yani yarım toplayıcı, bir bitlik iki sayının toplamını bulan ve çıkışında toplam ve elde bitlerini veren kombinasyonel bir devredir.

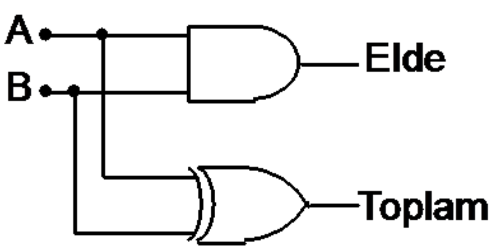
Toplayıcılar

Yarım toplayıcının doğruluk tablosu;

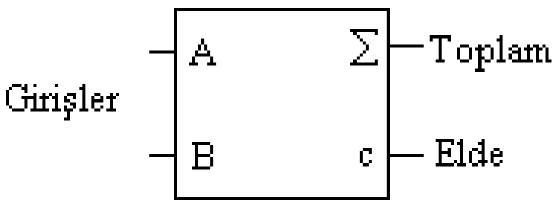
A B	Toplam	Elde
0 0	0	0
0 1	1	0
1 0	1	0
1 1	0	1

Toplam = $A \oplus B$

Elde = A.B



Yarım toplayıcının devre



Yarım toplayıcının grafiksel gösterimi

Toplayıcılar

Tam Toplayıcı (Full Adder - FA):

Tam toplayıcı bir devre, 1’er bitlik 2 girişi ve elde bitini alır, çıkışında toplam ve elde bitlerini oluşturur. Yarım toplayıcıya ek olarak elde girişi de vardır.

Tam toplayıcının doğruluk tablosu;

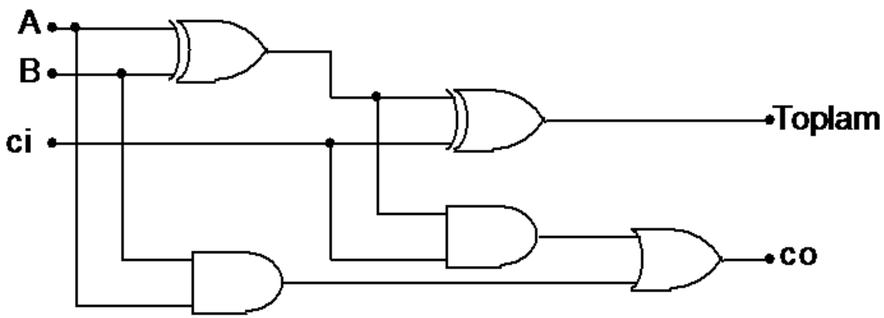
Girişler			Çıkışlar	
A	**B**	**ci**	**Toplam**	**co**
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$\begin{aligned} \text{Toplam} &= A'.B'.ci+A'.B.ci'+A.B'.ci'+A.B.ci \\ &= ci.(A'.B'+A.B)+ci'.(A'.B+A.B') \\ &= ci.(A\oplus B)'+ci'.(A\oplus B) \\ &= A\oplus B\oplus ci \end{aligned}$$

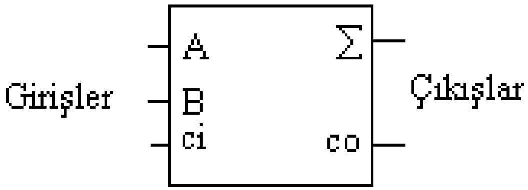
$$\begin{aligned} co &= A'.B.ci+A.B'.ci+A.B.ci'+A.B.ci \\ &= ci.(A\oplus B)+A.B \end{aligned}$$

Toplayıcılar

Toplam = A⊕B⊕ci co = ci.(A⊕B)+A.B

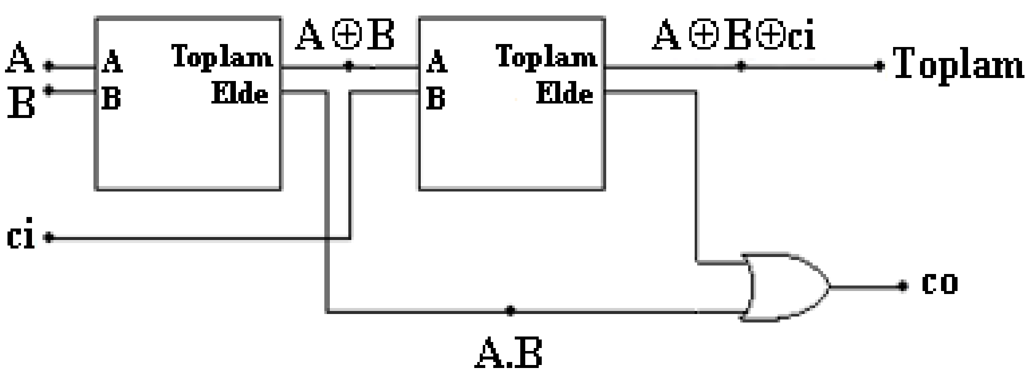


Tam toplayıcının devre şeması



Tam toplayıcının grafik gösterimi

2 yarım toplayıcı kullanarak tam toplayıcı elde etmek mümkündür:



Yarım toplayıcıda;

Elde = A.B

Toplam = A ⊕ B

Toplayıcılar

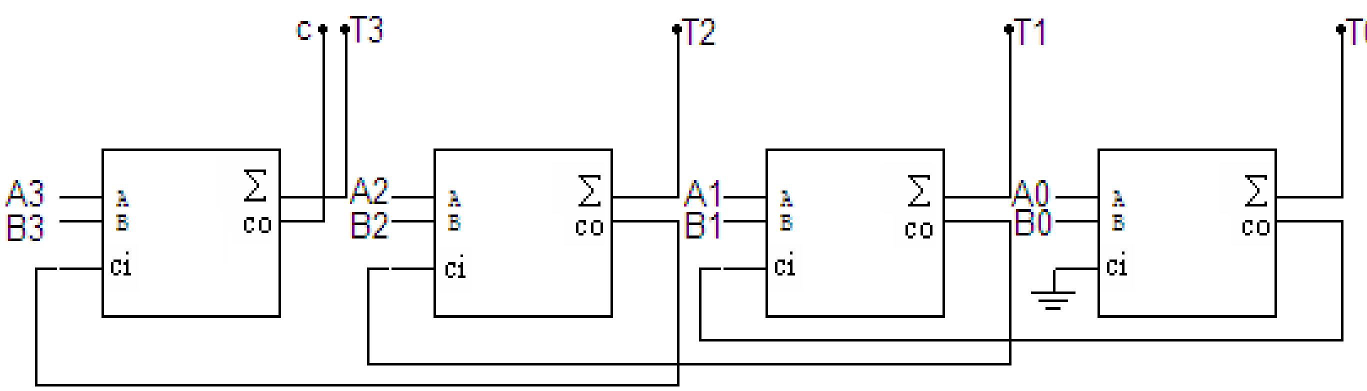
Paralel İkili Toplayıcılar:

Yarım ve tam toplayıcı devreler, tek bitlik sayıların toplama işleminde kullanıldı. Çok sayıda bit içeren ikili sayıların toplanması için, tam toplayıcılar kaskat bağlanmak suretiyle paralel toplayıcılar geliştirilmiştir. Örneğin 4 bitlik iki sayının toplama işlemini 4 adet tam toplayıcıyla gerçekleştirebiliriz. Bu gerçekleştirimde, her bir tam toplayıcının elde çıkışı bir sonraki tam toplayıcının elde girişine verilir. En düşük değerli bitleri toplarken elde girişi 0 alınır. En yüksek değerli bitlerin toplamı neticesinde c elde biti oluşur.

A3 A2 A1 A0
+ B3 B2 B1 B0

---

c T3 T2 T1 T0



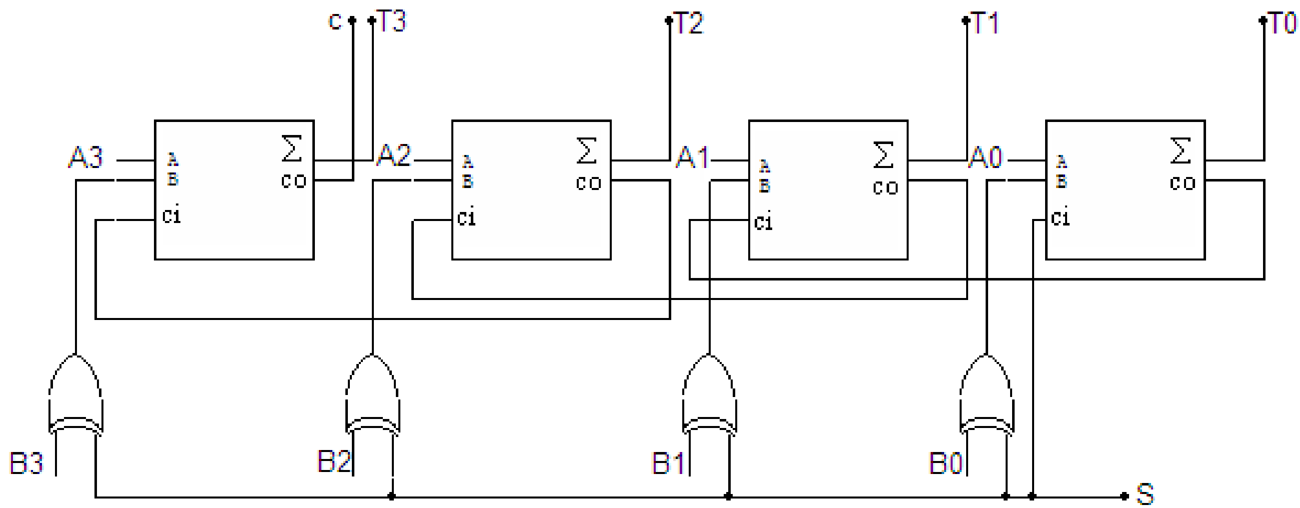
Toplayıcılar

Çıkarma işlemi, 2’ye tümleyen ile toplama işlemine dönüşebildiğinden, aşağıdaki devre yardımıyla hem toplama hem de çıkarma işlemleri yapılabilmektedir. 2’ye tümleyeni alınan sayı $(B3B2B1B0)_2$ ikili sayısıdır.

A3 A2 A1 A0
± B3 B2 B1 B0

---

c T3 T2 T1 T0

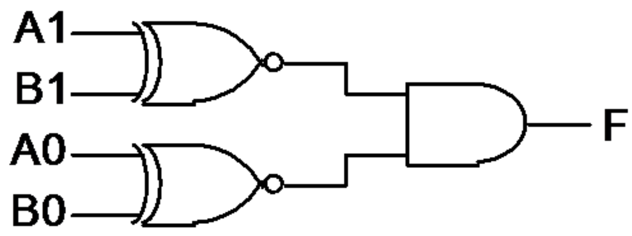


S = 0 için toplama işlemi, S = 1 için de çıkarma işlemi yapılır.

Karşılaştırıcılar

Karşılaştırıcılar, iki tane ikili sayının büyüklüklerini karşılaştırmak için kullanılırlar. Özellikle sayıların eşit veya eşit olmama durumlarını tespit etmek için EXOR veya EXNOR kapıları kullanılabilir.

Örneğin 2 bitlik iki sayının (A1A0 ve B1B0) karşılaştırması için aşağıdaki devre kullanılabilir.



A ve B ikili sayıları birbirine eşit olduğunda çıkış 1, diğer durumlarda ise çıkış 0 olur. N adet bit içeren sayılar için de bu devre genişletilebilir.

Karşılaştırıcılar

Örneğin 4 bitlik iki sayının (A3A2A1A0 ve B3B2B1B0) büyüklüklerini karşılaştırmak istersek o zaman şu ifadeleri yazabiliriz;

- Şayet A3 = 1 ve B3 = 0 ise A>B,
- Şayet A3 = 0 ve B3 = 1 ise A<B,
- Şayet A3 = B3 ise diğer düşük anlamlı bitlere bakmak gerekir.

Şu eşitlikler yazılabilir;

$$A>B \text{ çıkışı} = A3.B3' + (A3 \otimes B3).A2.B2' + (A3 \otimes B3).(A2 \otimes B2).A1.B1' + (A3 \otimes B3).(A2 \otimes B2).(A1 \otimes B1).A0.B0'$$

$$A<B \text{ çıkışı} = A3'.B3 + (A3 \otimes B3).A2'.B2 + (A3 \otimes B3).(A2 \otimes B2).A1'.B1 + (A3 \otimes B3).(A2 \otimes B2).(A1 \otimes B1)A0'.B0$$

$$A=B \text{ çıkışı} = (A3 \otimes B3).(A2 \otimes B2).(A1 \otimes B1).(A0 \otimes B0)$$

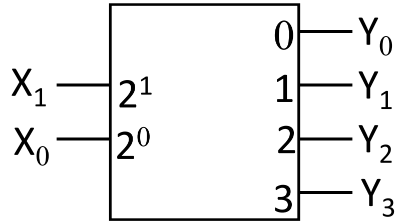
Kod Çözücüler (Decoders)

En genel tanımıyla n adet girişi ve en fazla 2^n adet çıkışı bulunan kombinasyonel devrelerdir. Girişindeki ikili sayının decimal değeri ne ise, o çıkışı aktif olur.

2×4 lük bir kod çözücünün doğruluk tablosu şu şekildedir;

X_1 X_0	Y_0	Y_1	Y_2	Y_3
0 0	1	0	0	0
0 1	0	1	0	0
1 0	0	0	1	0
1 1	0	0	0	1

2×4 Decoder



2×4 decoderin grafiksel gösterimi

$Y_0 = X_1'.X_0'$

$Y_1 = X_1'.X_0$

$Y_2 = X_1.X_0'$

$Y_3 = X_1.X_0$

Kod Çözücüler (Decoders)

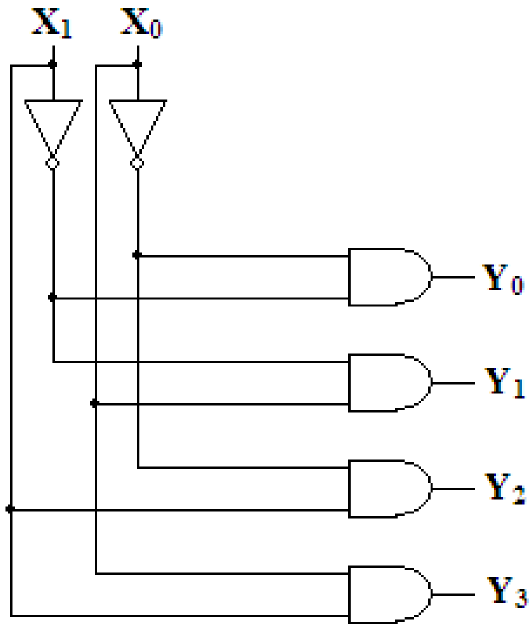
Kod çözücülerde yetki (Enable - EN) girişi de kullanılabilir. EN = 1 iken normal şekilde çalışırken, EN = 0 ise çıkışlar aktif değildir (tüm çıkışlar 0'dır).

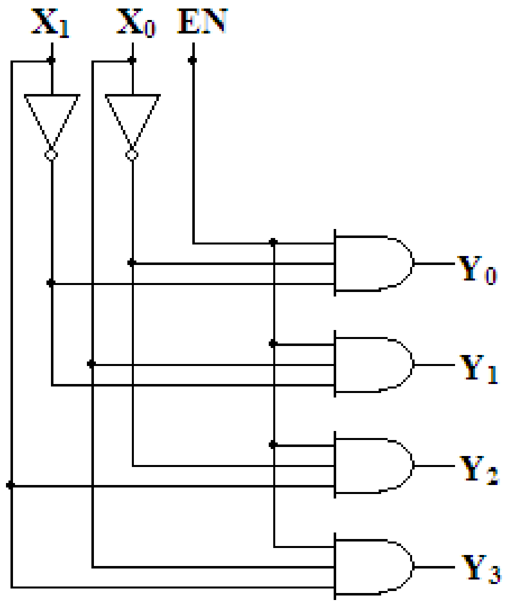
$Y_0 = X_1'.X_0'$

$Y_1 = X_1'.X_0$

$Y_2 = X_1.X_0'$

$Y_3 = X_1.X_0$



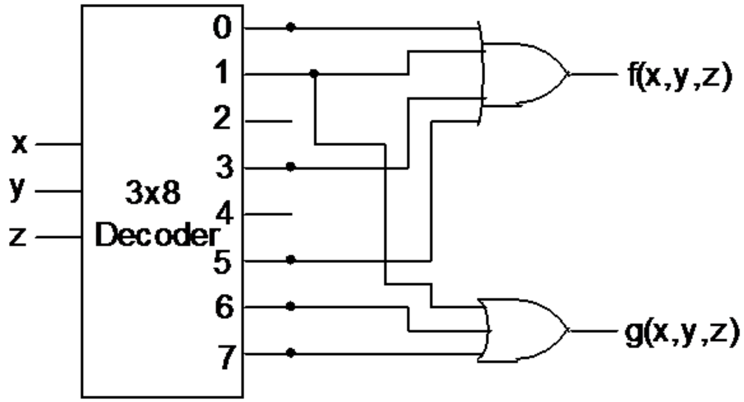


Kod Çözücüler (Decoders)

Decoder'ler, lojik fonksiyonların gerçekleştirilmesinde kullanılabilirler. Kod çözücülerin her bir çıkışı, aslında minterm ifadeleridir.

Genel olarak n adet girişe ve m adet çıkışa sahip bir lojik ifade, $n \times 2^n$ lik bir kod çözücü ve m adet de OR kapısı kullanılarak gerçekleştirilebilir.

Örnek: $f(x,y,z) = \Sigma(0,1,3,5)$ ve $g(x,y,z) = \Sigma(1,6,7)$ lojik ifadelerini bir kod çözücüyle gerçekleyelim.



Kodlayıcılar (Encoders)

Kodlayıcılar, kod çözücülerin gerçekleştirdiğinin tersini yapan kombinasyonel devrelerdir. Genel olarak n adet çıkışı ve en çok 2^n adet de girişi vardır. Normal şartlarda girişlerinin sadece bir tanesinin 1 olması gerekir. Bu durumda çıkışında, hangi girişin 1 olduğunu gösteren ikili kod üretir. 4 girişli 2 çıkışlı bir kodlayıcının doğruluk tablosu aşağıdaki gibidir;

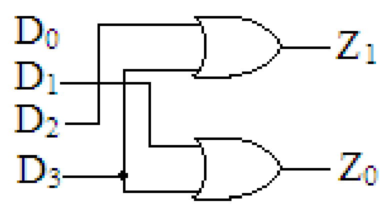
D_0	D_1	D_2	D_3	Z_1 Z_0
1	0	0	0	**0 0**
0	1	0	0	**0 1**
0	0	1	0	**1 0**
0	0	0	1	**1 1**

Aynı anda sadece bir tek girişin 1 olmasına müsaade edildiğinden, Z_1 ve Z_0 çıkışlarını Karnaugh ile hesap ederken, birden fazla girişin 1 olduğu durumları önemsiz durum (don't care) olarak alabiliriz.

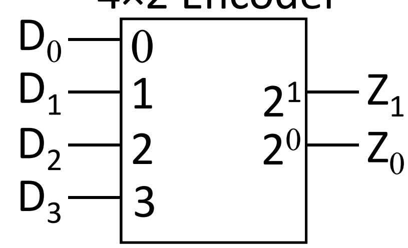
$$Z_1 = D_2 + D_3$$

$$Z_0 = D_1 + D_3$$

$NI = D_0'.D_1'.D_2'.D_3'$ (Giriş yoksa)



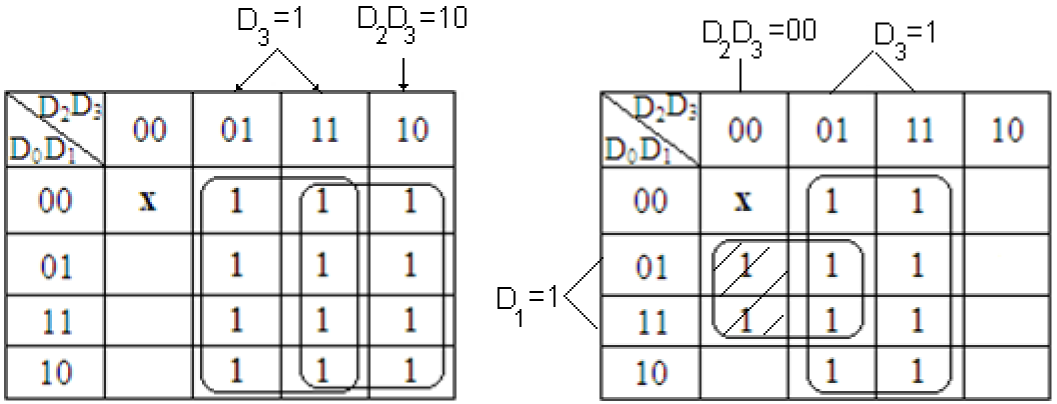
4×2 Encoder



Kodlayıcılar (Encoders)

Bazı durumlarda encoder'in girişlerinden sadece bir tanesinin 1 olması mümkün olmayabilir. Böyle durumlar da düşünülerek öncelikli kodlayıcılar (Priority encoders) geliştirilmiştir. Örnek bir gerçekleştirim, 4 giriş ve 3 çıkış için aşağıda verilmiştir. Öncelik sırası D_3'ten D_0'a doğrudur.

D_0	D_1	D_2	D_3	Z_1 Z_0	NI
0	0	0	0	x x	1
1	0	0	0	0 0	0
x	1	0	0	0 1	0
x	x	1	0	1 0	0
x	x	x	1	1 1	0



$Z_1 = D_2+D_3$

$Z_0 = D_3 + D_2'.D_1$

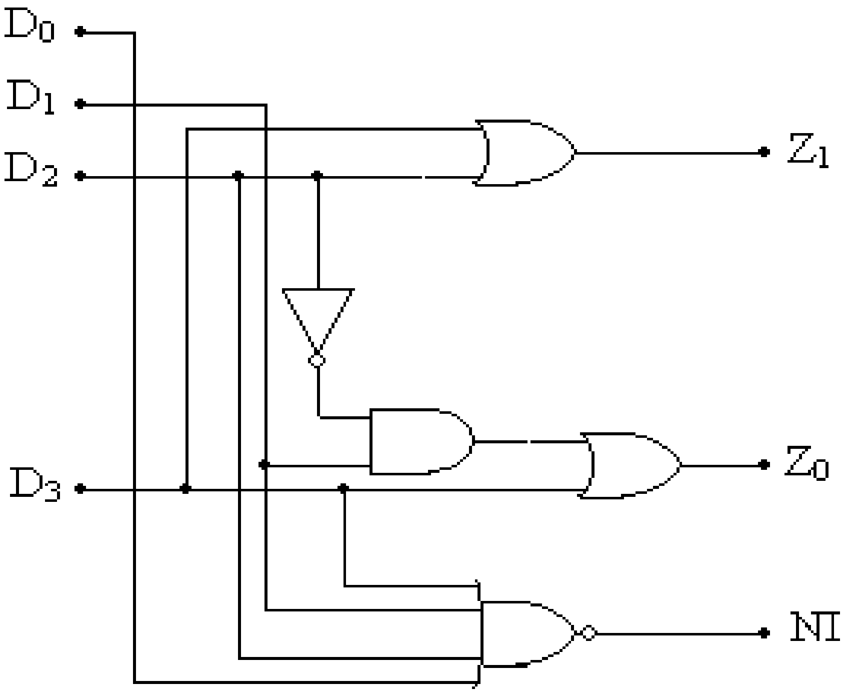
$NI = D_0'.D_1'.D_2'.D_3'$

Kodlayıcılar (Encoders)

$Z_1 = D_2 + D_3$

$Z_0 = D_3 + D_2'.D_1$

$NI = D_0'.D_1'.D_2'.D_3' = (D_0+D_1+D_2+D_3)'$



4×2 öncelikli encoder devresi