SAYI SİSTEMLERİ

8765 gibi bir ondalık sayı 8 tane binlik, 7 tane yüzlük, 6 tane onluk ve 5 tane birlikten oluşur. Görüldüğü üzere, on tabanındaki bir sayı 10 ve 10'un kuvvetlerinin belirli katsayı ile çarpılmasından oluşmaktadır.

$$10^3 \times 8 + 10^2 \times 7 + 10^1 \times 6 + 10^0 \times 5 = 8765$$

Genelde ondalık kesirli bir sayı aşağıdaki gibi gösterilir.

$$a_j a_{j-1} ......a_5 a_4 a_3 a_2 a_1 a_0 .a_{-1} a_{-2} a_{-3} a_{-4} ....$$

a_j katsayıları 0 ile 9 arasında değişen tam sayılarıdır. Ayrıca j değerleri 10'un hangi kuvvetinin alınacağını da gösterir.

$$a_5 \times 10^5 + a_4 \times 10^4 + a_3 \times 10^3 + a_2 \times 10^2 + a_1 \times 10^1 + a_0 \times 10^0 + a_{-1} \times 10^{-1} + a_{-2} \times 10^{-2} + a_{-3} \times 10^{-3} + a_{-4} \times 10^{-4}$$

Onlu sayı sistemi 10 ve 10'un katları taban ya da baz aldığından dolayı bu sayılara on tabanında sayılar denir. İkili sayı sisteminde ise sadece 0'lar ve 1'ler söz konusudur. Burada taban ikidir. Dolayısıyla 2 ve 2'nin kuvvetleri söz konusudur. Örneğin, 1010.11 sayısının onlu tabandaki karşılığı,

$$(1010.11)_2 = (\quad ? \quad)_{10}$$

$$= 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 0\times 2^0 + 1\times 2^{-1} + 1\times 2^{-2}$$

$$= 8 + 2 + 0.5 + 0.25 = 10.75$$

Genel olarak, r tabanındaki bir sayının onlu tabandaki karşılığı

$$...a_n r^n + a_{n-1} r^{n-1} + ... + a_3 r^3 + a_2 r^2 + a_1 r^1 + a_0 r^0 + a_{-1} r^{-1} + a_{-2} r^{-2} + a_{-3} r^{-3} + ...$$

Buradaki a katsayıları 0 ile r–1 arasında değişir. Bir sayı grubunun hangi tabanda olduğunu belirtmek için $(\text{sayı})_{\text{taban}}$ düzeni kullanılır. Eğer taban belirtilmez ise sayı onlu tabandadır. $(5427)_8$, $(274)_{10}$, $(1011)_2$, $(5645)_7$ gibi örnekler verilebilir. Eğer kullanılan taban 10'dan büyük ise bu durumda 10 ve 10'dan büyük rakamlar için alfabenin harfleri kullanılır. Örneğin, $(AB2C)_{16}$ sayısının 10'lu tabandaki karşılığı,

$$(AB2C)_{16} = 16^3 \times A + 16^2 \times B + 16^1 \times 2 + 16^0 \times C$$

Buradaki harfler 10 ve 10'dan büyük olan sayılara karşılık gelir. Şöyle ki;

A → 10	D → 13
B → 11	E → 14
C → 12	F → 15

İkili işlemler üzerine bazı örnekler;

$$\begin{array}{r} 10111 \\ +\ 01101 \\ \hline 100100 \end{array} \qquad \begin{array}{r} 10111 \\ -\ 01101 \\ \hline 01010 \end{array} \qquad \begin{array}{r} 1011 \\ \times \quad 101 \\ \hline 1011 \\ 0000 \\ +\ 1011 \\ \hline 110111 \end{array}$$

Sayı Tabanlarında Dönüşüm

- ✓ $(1010.011)_2 = (\ ?\ )_{10}$

  $= 2^3 \times 1 + 2^2 \times 0 + 2^1 \times 1 + 2^0 \times 0 + 2^{-1} \times 0 + 2^{-2} \times 1 + 2^{-3} \times 1$

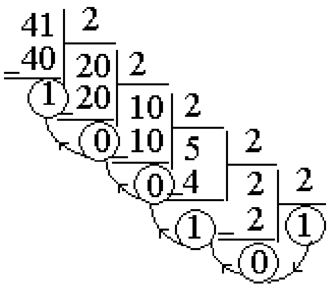
  $= (10.375)_{10}$

- ✓ $(630.4)_8 = (\ ?\ )_{10}$

  $= 8^2 \times 6 + 8^1 \times 3 + 8^0 \times 0 + 8^{-1} \times 4$

  $= (408.5)_{10}$

- ✓ $(41)_{10} = (\ ?\ )_2$



Bölüm, dönüştürülmek istenen tabandan küçük bir sayı kalıncaya kadar dönüştürülmek istenen tabana bölünür. Daha sonra en son bölümden başlamak üzere kalan sayılar yan yana yazılarak dönüşüm tamamlanmış olur.

$(41)_{10} = (101001)_2$

- ✓ $(0.6875)_{10} = (\ ?\ )_2$

$$\begin{aligned} 0.6875 \times 2 &= \boxed{1}.375 \Rightarrow a_{-1} = 1 \\ 0.375 \times 2 &= \boxed{0}.750 \Rightarrow a_{-2} = 0 \\ 0.750 \times 2 &= \boxed{1}.500 \Rightarrow a_{-3} = 1 \\ 0.500 \times 2 &= \boxed{1}.000 \Rightarrow a_{-4} = 1 \end{aligned}$$

$(0.6875)_{10} = (0.1011)_2$

✓ $(0.513)_{10} = (\ ?\ )_8$

$0.513 \times 8 = 4.104$	$a_{-1} = 4$
$0.104 \times 8 = 0.832$	$a_{-2} = 0$
$0.832 \times 8 = 6.565$	$a_{-3} = 6$
$0.656 \times 8 = 5.248$	$a_{-4} = 5$
$0.248 \times 8 = 1.984$	$a_{-5} = 1$
.... = ...	..... = ..

$(0.513)_{10} = (0.40651.....)_8$

Sayı hangi tabana dönüştürülecek ise o tabanla çarpılır ve tam kısmı kesirli kısımdan ayrılarak kesirli kısım sıfır oluncaya çarpma işlemine devam edilir.

✓ $(41.6875)_{10} = (\quad ? \quad)_2$

Tam kısmı ve kesirli kısmı ayrı ayrı dönüştürülerek dönüşüm işlemi tamamlanır.

$(41.6875)_{10} = (101001.1011)_2$

Sekizli ve Onaltılı Sayılar

Sayısal bilgisayarlar, ikili sayı mantığını kullandığı için ikili, sekizli ve onaltılı sayıların iki yönlü dönüşümleri oldukça önemlidir. 2^3=8 ve 2^4=16 olduğu için ikili sayı grubu içindeki her üç hane bir sekizliye, her dört hane de bir onaltılıya karşılık düşer. İkiliden sekizliye dönüşüm, noktadan başlayarak sağa ve sola doğru grupların üçer üçer ayrılması ile gerçekleştirilir. Sonra bu haneler karşılık düşen değerler yazılarak işlem sona erdirilir.

$(10111011101.1010111)_2 = (\ ?\ )_8$

$$\underbrace{10}_{2}\ \underbrace{111}_{7}\ \underbrace{011}_{3}\ \underbrace{101}_{5}\ .\ \underbrace{101}_{5}\ \underbrace{011}_{3}\ \underbrace{100}_{4} = 2735.534$$

$$\underbrace{101}_{5}\ \underbrace{1101}_{D}\ \underbrace{1101}_{D}\ .\ \underbrace{1010}_{A}\ \underbrace{1110}_{E} = (5DD.AE)_{16}$$

Neden ikili sayı sistemi değil de sekizli ya da onaltılı sayı sistemi? Şöyle bir örnek ile rahatlıkla açıklanabilir: sayısal bilgisayarlarla iletişim kurmak için veri almak ya da yazmak gereklidir. Örneğin iki tabanında 1111111111111111 sayısını iletişim verisi olarak düşünelim. Bu verinin sayısal bilgisayara iletilebilmesi için 16 tane haneye ihtiyaç vardır oysaki sekiz tabanında bu sayı 377777 gibi altı hane ile onaltı tabanında ise FFFF gibi dört hane ile ifade edilebilir. İkili gösterime göre 3 ya da 4 kat daha az hane gerektirdiğinden bu sayı tabanları sayısal bilgisayarlarda tercih edilir.

Tablo. Farklı tabanların dönüşümü

TABAN 10	TABAN 2	TABAN 8	TABAN 16
00	0000	00	0
01	0001	01	1
02	0010	02	2
03	0011	03	3
04	0100	04	4
05	0101	05	5
06	0110	06	6
07	0111	07	7
08	1000	10	8
09	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Sayıların Tümleyeni

Sayıların tümleyenleri bilgisayarlarda çıkarma işlemini basitleştirmek ve lojik işlemleri yürütmek için kullanılır. *r* tabanındaki bir sayı için iki tümleyen vardır: birisi taban tümleyeni diğeri de taban–1 tümleyeni. Bunlar sırasıyla, *r'*nin tümleyeni ve *r*-1'in tümleyeni olarak bilinir.

Taban–1 tümleyeni:

r tabanında n haneli bir N sayısı verildiğinde N'in *r*-1 tümleyeni (r^n-1)-N olarak verilir. 10 tabanında *r*=10 ve *r*-1=9 olup 9'un tümleyeni 10^n-1-N olur. 10 tabanında 4 haneli bir sayı için 9'a tümleyeni 9999'dan sayının çıkartılması ile elde edilir. Buradan da görüldüğü üzere 10 tabanında bir sayının 9'a tümleyeni her bir hanenin 9'dan çıkartılması ile elde edilir.

İkili sayılar için *r*=2 ve *r*-1=1 olup herhangi iki tabanındaki bir N sayısının 1'e tümleyeni (2^n-1)-N'dir. Örneğin, iki tabanında dört haneli bir sayı için n=4 olup $2^4=(10000)_2$ ve $2^4–1=1111$. İkili sayıların her bir basamağının 1'den çıkartılması 1'lerin 0, 0'ların 1 olması anlamına gelmektedir.

10001010'in 1'e tümleyeni 01110101'dir.

Sekizli ve onaltılı sayıların (*r*-1)'e tümleyeni ise sırasıyla 7 ve F'den çıkartılması ile elde edilir.

Taban Tümleyeni

n haneli bir N sayısının taban tümleyeni N≠0 için r^n-N ve N=0 için 0'dır. (r-1)'e göre tümleyen ile r'ye göre tümleyen karşılaştırıldığında r^n-N=[(r^n-1)-N]+1 olup r'ye göre tümleyen r-1'e göre tümleyene 1 eklenerek elde edilir. Örneğin, onlu tabandaki 2356 sayısının r-1'e göre tümleyeni 7643, r'ye göre tümleyeni 7644'dür. Bir başka değişle, onlu tabandaki sayının en düşük anlamlı rakamı 10'dan diğerleri de 9'dan çıkartılarak 10'a (taban) tümleyeni bulunur. Eğer en düşük anlamlı rakam sıfır ise sola doğru sıfırdan farklı ilk elemandan 10 çıkartılır diğerlerinden yine 9 çıkartılarak tümleyeni bulunur. Benzer şekilde ikili tabanda en düşük anlamlı rakamlar 0 ise ilk 1'de dâhil olmak üzere bunlar değiştirilmeden bırakılıp diğer değerler 1 ise 0, 0 ise 1 yapılarak 2'ye göre tümleyeni bulunur. Eğer en düşük anlamlı rakam 1 ise bu değiştirilmez diğer rakamlar 1 ise 0, 0 ise 1 yapılarak tümleyen bulunur. Eğer sayı kesirli ise nokta geçici olarak kaldırılır. Tümleyeni bulunduktan sonra nokta eski konumuna yerleştirilerek tümleyeni bulunmuş olur. Bir sayının tümleyeninin tümleyeni yine kendisine eşittir. Örneğin, N sayısının r'ye göre tümleyeni r^n-N, r^n-N sayısının tümleyeni r^n-(r^n-N)=N'dir.

Taban Tümleyeni Kullanılarak Çıkarma İşlemi

Sayısal bilgisayarlar çıkarma işlemini yapmak için burada anlatılacak mantığı kullanır. n haneli ve işaretsiz iki sayıdan oluşan r tabanındaki M-N gibi bir çıkarma işlemi;

- Çıkarılan M sayısı, çıkan N sayısının r'ye göre tümleyenine eklenir. Böylece M+(r^n-N)=M-N+r^n işlemi gerçekleşmiş olur.
- M ≥ N ise toplama sonucundan r^n eldesi atılarak M-N işleminin sonucu elde edilir.
- M < N ise toplama sonucunda elde oluşmaz ve sonuç r^n-(N-M)'e eşit olur. Bu aslından M-N'in r'ye göre tümleyenidir ve sonuç negatiftir.

Örnek: 10'a tümleyen kullanarak 72532–3250 işlemini gerçekleştiriniz.

M	=	72532	
N'in 10'a göre tümleyeni	=	96750	(10^5–3250 sayı 4 haneli ama haneler eşit olmak zorunda olduğu için n=5 seçildi.)
		+	
Toplam	=	169282	
M ≥ N olduğu için atılacak son elde 10^5	=	− 100000	
Sonuç	=	69282	

N'in 10'a tümleyeni ile en anlamlı hane 9 olur. Toplamda yeni bir hanenin oluşması atılacak bir elde olduğu ya da işlem sonucunun pozitif olduğu anlamına gelir.

Örnek: 10'a tümleyen kullanarak 3250-72532 işlemini gerçekleştiriniz.

M	=	03250
N'in 10'a göre tümleyeni	=	27468
Toplam	=	30718
Atılacak elde yoktur.		
Sonuç:-(30718'in 10'a tümleyeni)	=	-69282

N'in 10'a tümleyeni alınıp toplandığında herhangi bir hane oluşmadığından atılacak bir elde yoktur ve işlem sonucu negatiftir.

Örnek: X=1010100 ve Y=1000011 ikili sayılarını 2'ye tümleyenlerini kullanarak (a) X-Y ve (b) Y-X işlemlerini gerçekleştiriniz.

(a)	X	=	1010100	
	Y'nin 2'ye tümleyeni	=	+ 0111101	
	Toplam	=	10010001	(hane oluştuğu için elde var)
	Atılan elde 2^7	=	−10000000	
	Sonuç (X-Y)	=	00010001	
(b)	Y	=	1000011	
	X'nin 2'ye tümleyeni	=	+ 0101100	
	Toplam	=	1101111	(hane oluşmadığı için elde yok)
	Atılan elde	=	Yok	
Sonuç:-(Y-X)Toplamın 2'ye tümleyeni		=	-0010001	

Taban–1 tümleyeni kullanılarak çıkarma işleminin yapılması

İşaretsiz sayıların çıkarma işlemi (r-1)'e tümleyen kullanılarak da yapılabilir. (r-1)'e tümleyenin r'ye tümleyenden bir eksik olduğu hatırlanırsa çıkan tümleyeni ile çıkarılanın toplamı sonucunda elde oluşuyor ise sonuç, eldenin kaldırılarak toplama eklenmesi ile bulunur.

Örnek: X=1010100 ve Y=1000011 ikili sayılarını 1'e tümleyenlerini kullanarak (a) X-Y ve (b) Y-X işlemlerini gerçekleştiriniz.

(a) X = 1010100

Y’nin 1’e tümleyeni	=	+ 0111100	
Toplam	=	$\boxed{1}$0010000	(hane oluştuğu için elde var)
Elde aktarması	=	+ 1	
Sonuç (X-Y)	=	0010001	

(b)

Y	=	1000011	
X’nin 1’e tümleyeni	=	+ 0101011	
Toplam	=	1101110	(hane oluşmadığı için elde yok)
Atılan elde	=	Yok	
Sonuç:-(Y-X)Toplamın 1’e tümleyeni	=	-0010001	

İKİLİ KODLAR

Bilgisayarlar sadece birler ve sıfırlar üzerinde işlem yapabildikleri için veriler ikili olarak kodlanmış olarak kaydedilirler. Kaydedilen kodların her bir hanesi bir bittir. Bit, tanıma göre ikili bir hanedir. İkili bir hane de ya 0 ya da 1’den oluşmuş bir büyüklüktür. n bitlik bir sayı dizisi ya da kod 2^n farklı gruptan oluşan bir büyüklüğü temsil eder. Örneğin, 2 bitlik bir kodda 00, 01, 10 ve 11 olmak üzere 2^2=4 tane farklı durum belirtilebilir.

Onlu sayılar en az dört bit ile ifade edilir. Dört ya da daha fazla biti farklı kombinasyonlarda düzenleyerek çeşitli kodlar elde edilebilir. Kullanılan birkaç kod aşağıdaki tabloda verilmiştir.

Tablo. Onlu sayılar için ikili kodlar

Onlu Sayı	**BCD 8421**	**3 Artıklı**	**84-2-1**	**2421**	**İki Beşli 5043210**
0	0000	0011	0000	0000	0100001
1	0001	0100	0111	0001	0100010
2	0010	0101	0110	0010	0100100
3	0011	0110	0101	0011	0101000
4	0100	0111	0100	0100	0110000
5	0101	1000	1011	1011	1000001
6	0110	1001	1010	1100	1000010
7	0111	1010	1001	1101	1000100
8	1000	1011	1000	1110	1001000
9	1001	1100	1111	1111	1010000

BCD (**B**inary **C**oded **D**ecimal)(ikili kodlanmış onlu), her onlu sayının yukarıdaki tabloda verildiği gibi bir koda karşılık gelmesi ile elde edilir. BCD kodunda ağırlıklar 8, 4, 2 ve 1’dir. Örneğin, onlu tabandaki 6, BCD kodunda 0110 değerine karşılık gelir. 0×8+1×4+2×1+0×1=6 şeklinde onlu tabana çevrilir. Bu onlu sayının ikili tabana çevrilmesi gibi görünmekle birlikte tamamen farklıdır. 0 ile 9 arasındaki sayıların ikili tabanı ile BCD kodları aynı olmakla beraber 9’dan büyük sayılar için farklıdır. Örneğin, onlu 27 sayısını düşünelim, 27 sayının ikili dönüşümü, 11011 iken BCD kodu 0010 0111'dir. Her bir onlu basamağın ayrı ayrı ikili karşılıkları yazılarak sayı BCD olarak kodlanır. 3-artıklı kodlamada sayının BCD koduna 3 eklenerek elde edilir. Eski bilgisayarlarda kullanılan bir kodlama türüdür. 84-2-1 kodlamada ağırlıklar 8, 4, -2 ve -1 olarak değişir. 1011 sayısının onlu karşılığı, $1\times8+0\times4+1\times(-2)+1\times(-1)=5$ olarak bulunur. Tabloda gösterilen diğer ağırlıklı kodlar ise 2421 ve 5043210 'dır.

Bu verilen kodlamalardan en doğal olanı ve en çok kullanılanı BCD kodlamadır. Listedeki diğer dört bitli kodlamalarda ise BCD'de olmayan bir özellik vardır. 3 artıklı kod, 84-2-1 ve 2421 kodlamaları kendilerini simetrik olarak tümleyen kodlardır, yani onlu sayının 9'a tümleyeni 1'leri 0, 0'ları 1 yapmakla kolayca bulunabilir. 604 sayısının 2421 kodunda 110000000100 olarak kodlanır. Bu sayının 9'a tümleyeni 001111111011 olup onlu karşılığı 395 'dir. Verilen diğer kod olan iki beşli kodu hata bulma özelliği olan bir koddur. Bu kodda sayı iki tane 1 ve beş tane 0 ile temsil edilir. Elektriksel olarak bu bilginin iletilmesi sırasında bu 1 ve 0'lar belirli elektriksel sinyaller ile ifade edilirler. Veri bir yerden bir yere iletilirken bozulmalar söz konusu olabilir. Bu kod yardımı ile alınan veri yukarıda verilen tablodaki kombinasyonlara uymuyorsa bir hata saptanır.

Tablo. Eşitlik biti

Tek Eşitlik		Çift Eşitlik	
Mesaj	**P**	**Mesaj**	**P**
0000	1	0000	0
0001	0	0001	1
0010	0	0010	1
0011	1	0011	0
0100	0	0100	1
0101	1	0101	0
0110	1	0110	0
0111	0	0111	1
1000	0	1000	1
1001	1	1001	0
1010	1	1010	0
1011	0	1011	1
1100	1	1100	0
1101	0	1101	1
1110	0	1110	1
1111	1	1111	0

Hata Bulma Kodu

İkili bir bilginin iletilmesi sırasında bilgide gürültüden dolayı bazı kayıplar söz konusu olabilir. Gönderilen verideki 1'lerin sayısını tek ya da çift yapacak bir ek bit mesaja eklenmesi ile elde edilir. İletim hattı boyunca bir bilgi kaybolması söz konusu olmuş ise bu bit yardımı ile hata olduğu anlaşılır. Bu kodlamada hata tespit edilir fakat düzeltilemez. Ayrıca çift eşitlik kodlamada iki tane bit değişirse bu hata da tespit edilemez ancak ilave hata bulma yöntemlerine ihtiyaç duyulur.

Gray Kodu

Sürekli sinyallerin sayısal ya da ayrık sinyallere dönüştürülmesinde kullanılan bir

Tablo. Gray Kodu

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

kodlama türüdür. Bu koddaki avantaj bir birini takip eden sayıların bitlerinin tek tek değişmesidir. Örneğin, ikili 7 sayısı 8’e geçerken dört bit değer değiştirir. Bu kodda ise 7, 0100 ile gösterilir 8 ise 1100 ile gösterilir. Görüldüğü gibi sadece bir bitte değişim olmuştur. Hızlı değişen sayılarda ikili kodlarda birden fazla bitin değişiminden doğacak gecikme hataları olabilir. Ama gray kodunda sadece bir bit değişeceğinden bu tip geçiş hatalarından kurtulabiliriz. Gray kodunun en tipik uygulaması sürekli mil konumunun belirlenmesidir. Bu uygulamada mil konumları parçalara ayrılarak bu parçalara numaralar verilir. Peş peşe verilen numaralar ile geçişlerdeki hatalar ortadan kaldırılır. İkili sayının gray kodu karşılığı için:En yüksek değerlikli (MSB) bit aşağı indirilir. Her bit solundaki bitle elde dikkate alınmaksızın toplanır. Bu işlem en düşük değerlikli (LSB) bite kadar devam eder. Elde edilen sayı, Binary sayının Gray kod karşılığıdır.

ASCII Karakter Kodları

Bilgisayarlarda veriler sadece rakamlardan oluşmaz. Harf ya da bazı özel karakterlerden oluşabilir. Bu karakterler ve kodları aşağıda verilmiştir.

Tablo. ASCII Karakter Tablosu

				$b_7b_6b_5$				
$b_4b_3b_2b_1$	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
0000	NUL	DLE	SP	0	@	P	‘	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	“	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	‘	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	ı	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

İKİLİ LOJİK

İkili lojik, iki ayrık değeri olan değişkenler ve mantık alanındaki işlemlerle ilgilenir. Değişkenlerin aldığı iki değer olup bunlar Evet-Hayır ya da Doğru-Yanlış gibi isimler ile adlandırılır. Ama bit olarak düşüldüğünde 1 ya da 0 değerlerini bu değişkenlere atamak daha uygun olacaktır. Burada anlatılacak olan ikili lojik Boole Cebri diye adlandırılan cebir türüne eşdeğerdir. Boole cebri, ikili işlemler ve bunları gerçekleştiren sayısal devrelerin tasarımı arasında bir köprü görevi görür.

İkili lojik iki değişken ve lojik işlemlerden oluşur. Sadece 1 ve 0 olan bu değişkenlere A, B, C, x, y, z… gibi alfabeden değişken isimleri atamak kullanılan en yaygın yoldur. VE, VEYA ve DEĞİL olmak üzere üç tane temel lojik işlem vardır.

1- VE: Bu işlem iki lojik değişkenin arasına bir nokta konularak ya da herhangi bir işaretçi kullanılmadan temsil edilir. Örneğin x.y=z ya da xy=z gibi. Bu işlem "*x ve y eşittir z*" şeklinde okunur. VE işlemi x ve y 1 olduğunda 1 değerini verirken diğer tüm durumlarda çıkış 0 olarak gerçekleşir. İkili tabandaki çarpma işlemine benzemektedir.

VE işlemi için doğruluk tablosu

x	**y**	**x.y**
0	0	0
0	1	0
1	0	0
1	1	1

2- VEYA: bu işlem (+) işareti ile temsil edilir. İki lojik değişken arasındaki VEYA işlemi x+y=z ile gösterilir ve "*x veya y eşittir z*" şeklinde okunur. Bu işlem ikili toplamadan farklıdır. İkili toplamada 1+1=10 iken ikili lojikte 1+1=1'dir. İlki, "*bir artı bir eşittir iki*" diye okunurken ikincisi "*bir veya bir eşittir bir*" diye okunur.

VEYA işlemi için doğruluk tablosu

x	**y**	**x+y**
0	0	0
0	1	1
1	0	1
1	1	1

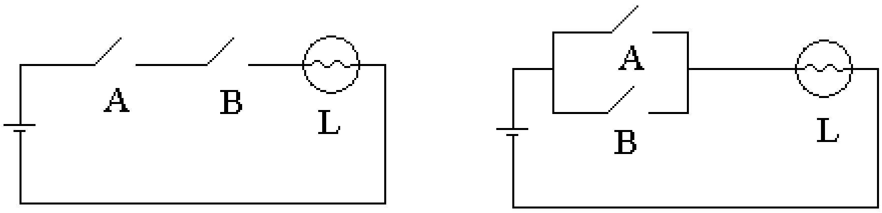
3- DEĞİL: üssü (') ya da üst çizgi (–) ile temsil edilir. $x' = z$ ya da $\bar{x} = z$ şeklinde yazılırken "*x'in değili eşittir z*" olarak okunur.

Değil işlemi için doğruluk tablosu

x	**x'**
0	1
1	0

Anahtarlama Devreleri ve İkili İşaretler

Şekildeki devreleri göz önüne alalım.



(a) (b)

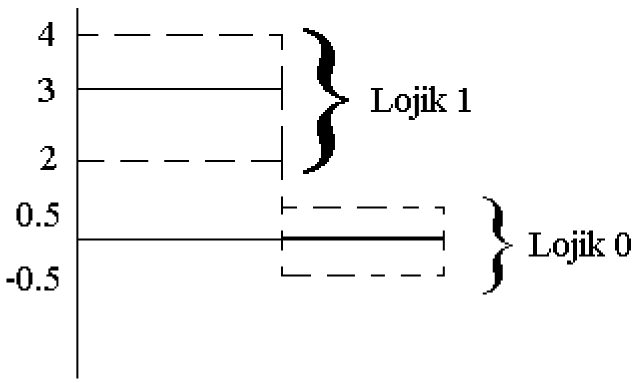
A ve B anahtarları açıkken 0, kapalı iken 1'i temsil etsin. Benzer şekilde L lambası yanıyorken 1, sönükken 0'ı temsil etsin. Anahtarlar seri bağlı iken ***A ve B*** lambalarının kapalı olması durumunda L lambası yanar. Anahtarlar paralel bağlı iken ***A veya B*** anahtarlarının kapalı olması durumunda lamba yanar. Bu iki devre için VE ve VEYA işlemleri ile aşağıdaki bağıntılar çıkartılabilir.

L=A.B (a)

L=A+B (b)

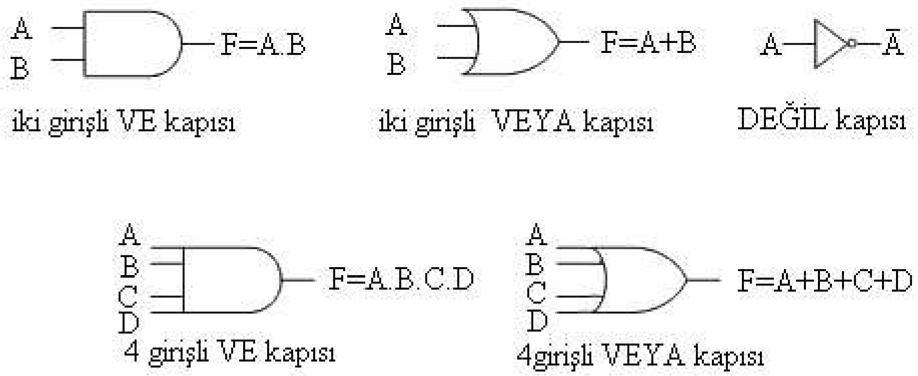
Elektronik sayısal devreler, transistör gibi aktif bir elemanın anahtar gibi davranmasından dolayı anahtarlamalı devreler olarak da adlandırılır. Transistörün iletimde olması anahtarın kapalı, kesimde olması ise anahtarın açık olması durumlarına karşılık gelir. Bu anahtarları elle kapatmak yerine ikili işaretleri kullanarak transistör iletim ya da kesim durumlarının birinde çalıştırılabilir. Bütün sayısal sistemlerde akım ya da gerilim gibi elektriksel büyüklükler iki belirli değerden birini alır. Bunlar lojik 1 ya da lojik 0 olarak adlandırılan iki gerilim seviyesidir. Lojik 1, 2 ile 4V arasında değişen bir gerilim seviyesi iken lojik 0 ise, 0.5 ile -0.5V arasında değişir.

İzin verilen aralıklar kesikli çizgiler ile sınırlandırılmış olan bölgelerdir. Bu bölgeler dışında sadece durum geçişleri sırasında karşılaşılır.

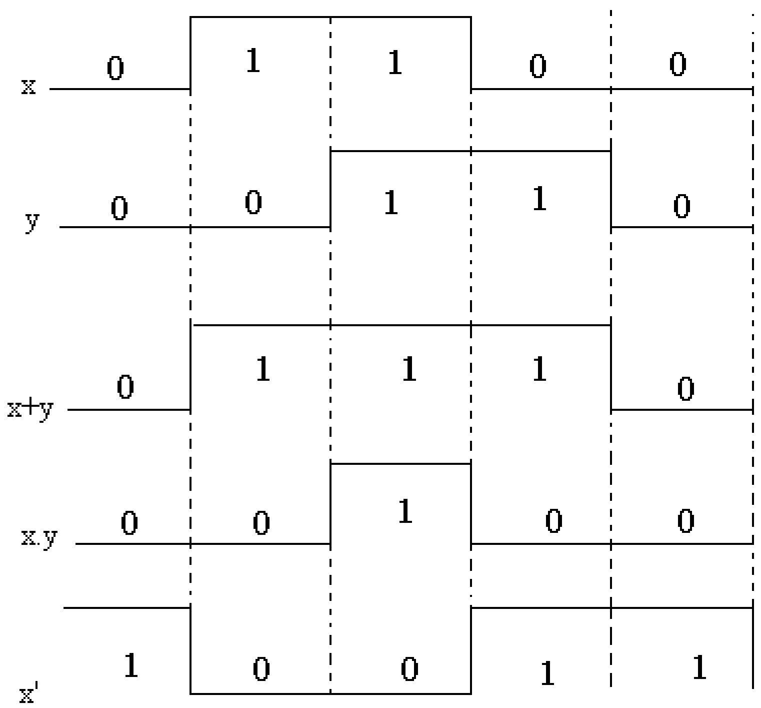


Lojik Kapılar

Aşağıda VE, VEYA ve DEĞİL lojik işlemlerini gerçekleştiren lojik kapı sembolleri görülmektedir. Kapı adı verilen bu devreler, girişteki lojik şartlar sağlandığında Lojik 1 ya da Lojik 0 işareti üreten devre elemanlarıdır.



İkili işaretler şeklinde düşünülecek olursa VE, VEYA ve DEĞİL işlemleri sonucunda iki girişli kapıların çıkış sinyalleri aşağıdaki gibi olur.



BOOLE CEBRİ VE LOJİK KAPILAR

Boole cebri, sonuç çıkarma prensibine dayalı diğer matematiksel sistemler gibi bir eleman kümesi, bir işlem kümesi ve belirli sayıda kanıtlanmamış aksiyomdan (postulat) oluşur. Bir S kümesinin elemanları S={a,b,c,d} olsun. $a\in S$, $b\in S$ iken * şeklinde tanımlı bir ikili işlem a*b=e olsun. $e\notin S$ olduğu için bu * işlemi bir ikili işlem değildir.

Matematiksel bir sisteme ait en genel cebirsel postulatlar şunlar.

1. Kapalılık Özelliği: İkili bir işlem, S'deki her eleman çiftine yine S'de bir elemana düşecek şekilde tanımlı ise S kümesi bu işleme göre kapalıdır denir. Örneğin N={1,2,3,…}doğal sayı kümesi (+) işlemine göre kapalıdır. 2+5=7 yine N kümesinin bir elemanıdır. Ancak, (-) işlemine göre kapalı değildir. ($2-3=-1\notin N$)
2. Birleşme Özelliği: x, y, $z\in S$ için ,

   (x*y)*z=x*(y*z) yazılabiliyorsa bu ikili işlemin birleşme özelliği vardır denir.
3. Değişme Özelliği: x, $y\in S$ için,

   x*y=y*x yazılabiliyorsa bu ikili işlemin değişme özelliği vardır denir.
4. Birim Eleman: $x\in S$ için,

   x*e=e*x=x $e\in S$ olan ve * ikili işlemine göre birim eleman e'dir denir.
5. Ters (Inverse): * işlemine göre birim elemanı e olan bir S kümesinin $\forall$ $x\in S$ için x*y=e olmasını sağlayan $y\in S$ elemanı mevcut ise x'in tersi vardır denir.
6. Dağılma Özelliği: * ve . S kümesi üzerinde bir ikili işlem olsun.

   x*(y.z)=(x*y).(x*z) özelliğine * ikili işleminin, . ikili işlemi üzerinde dağılma özelliği vardır denir.

Yukarıdaki maddeler özetlenecek olursa;

+ işlemi toplamayı tanımlar.

Toplamanın birim elemanı 0'dır.

Toplamanın tersi çıkarmadır.

. ikili işlemi çarpmayı temsil eder.

Çarpmanın birim elemanı 1'dir.

a'nın çarpmaya göre tersi bölmeyi tanımlar. (a).(1/a)=1

Sadece çarpmanın toplama üzerinde dağılma özelliği vardır.

a.(b+c)=(a.b)+(a.c)

Boole Cebrinin Özellikleri(Huntington Postulatları)

1.a. + işlemine göre kapalılık

b. . işlemine göre kapalılık

2.a. + işlemine göre 0 birim elemandır. x+0=0+x=x

b. . işlemine göre 1 birim elemandır. x.1=1.x=x

3.a. + işlemine göre değişme özelliği vardır. x+y=y+x

b. . işlemine göre değişme özelliği vardır. x.y=y.x

4.a. . işleminin + işlemi üzerinde dağılma özelliği vardır. x.(y+z)=(x.y)+(x.z)

b. + işleminin **.** işlemi üzerinde dağılma özelliği vardır. x+(y.z)=(x+y).(x+z)

5. $\forall$ x$\in$B için bir x'$\in$B elemanı (x'in tümleyeni) vardır.

a. x+x'=1

b. x.x'=0

6. x$\neq$y olacak şekilde en az iki tane x,y$\in$B vardır.

Boole Cebri ile Sıradan Cebir Arasındaki Farklar

1. + işleminin . işlemi üzerindeki dağılma kuralı yani x+(y.z)=(x+y).(x+z) sıradan cebirde geçerli değildir.
2. Boole cebrinde toplama ve çarpmaya göre elemanın tersi olmadığından çıkarma ve bölme yoktur.
3. Postulat 4 sıradan cebirde olmayan bir işlemi tanımlar.
4. Sıradan cebir sonsuz elemanlı bir küme üzerinde çalışırken Boole cebri iki elemanlı B={0,1} kümesi üzerinde çalışır.

Boole cebri bazı işlemlerde sıradan cebire benzediği için + ve . işaretleri seçilmiştir. Burada incelenecek olan Boole cebrini kapı tipi devre uygulamasıdır.

İki Değerli Boole Cebri

İkili + ve . işlemlerine ait tablolar aşağıda gösterilmiştir. Görüldüğü üzere, bu tablolar daha önce anlatılan VE ve VEYA işlemleri ile aynıdır.

x	**y**	**x.y**
0	0	0
0	1	0
1	0	0
1	1	1

x	**y**	**x+y**
0	0	0
0	1	1
1	0	1
1	1	1

x	**x'**
0	1
1	0

Bu durumda, Huntington postulatlarının B={0,1} kümesi ve daha önce anlatılan ikili işlem için geçerli olduğu gösterilmelidir.

1. Kapalılık: Tablodan da görüldüğü üzere her bir işlemin sonucu 0 veya 1 olup $0,1 \in B$'dir
2. **a.** 0+0=0 0+1=1+0=1

   b. 1.1=1 1.0=0.1=0

   Postulat 2'de tanımlandığı gibi 0 toplam, 1 de çarpım için birim elemandır.
3. Değişme Kuralı: Tablodaki simetriden de görülebileceği gibi her iki işlemin de değişme özelliği vardır.
4. **a.** x.(y+z)=(x.y)+(x.z) dağılma kuralı bir doğruluk tablosu ile kolaylıkla görülebilir.

x	y	z	y+z	x.(y+z)	x.y	x.z	(x.y)+(x.z)
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

b. x+(y.z)=(x+y).(x+z) dağılma kuralı bir doğruluk tablosu ile kolaylıkla görülebilir.

x	y	z	(y.z)	x+(y.z)	(x+y)	(x+z)	(x+y).(x+z)
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

5. Tümleyen tablosu kullanılarak

a. 0+0'=0+1=1 ve 1+1'=1+0=1 olduğundan x+x'=1 olduğu

b. 1.1'=1.0=0 ve 0.0'=0.1=0 olduğundan x.x'=0 olduğu gösterilebilir.

6. iki değerli Boole cebrinin 0 ve 1 gibi iki değeri olduğu ve $0 \neq 1$ olduğundan dolayı postulat 6 sağlanır.

Görüldüğü üzere iki değerli Boole cebri daha önce anlatılmış olan VE ve VEYA işlemlerine, tümleyen işleminin ise DEĞİL işlemine eşdeğer olduğu görülür.

BOOLE CEBRİNE İLİŞKİN TEMEL TEOREM VE ÖZELLİKLER

Dualite

Huntington postulatları (a) ve (b) şıkları halinde bir çift olarak verilmişti. İkili işlemler ve birim elemanları kendi aralarında değiştirilerek bir şıktan diğeri elde edilebilir. Buna Boole cebrinin dualite prensibi denir. Cebirsel bir ifadenin duali istendiğinde VEYA ve VE işlemleri değiştirilir, 1'ler yerine 0, 0'lar yerine 1 konur.

Temel Teoremler

Aşağıda Boole cebrine ilişkin tabloda, altı teorem ve dört postulat yer almaktadır. Bu teorem ve postulatlar Boole cebrinin öğrenilmesi gereken en temel bağıntılardır. Teoremlerde postulatlar gibi çiftler halinde verilmiştir. Sağdaki ifadeler soldaki ifadenin dualidir. Postulatlar, cebirsel yapıların temel aksiyomları olup kanıtlanmasına gerek yoktur. Teoremler, postulatlar kullanılarak ispatlanır.

Tablo. Boole Cebrine ilişkin postulatlar ve teoremler

Teorem 1	**(a)** x+x=x	**(b)** x.x=x
Teorem 2	**(a)** x+1=1	**(b)** x.0=0
Teorem 3, ters alma	$(x')' = x$	
Teorem 4, birleşme	**(a)** x+(y+z)=(x+y)+z	**(b)** x.(y.z)=(x.y).z
Teorem 5, DeMorgan	**(a)** (x+y)'=x'.y'	**(b)** (x.y)'=x'+y'
Teorem 6, yutma	**(a)** x+x.y=x	**(b)** x.(x+y)=x
Postulat 2	**(a)** x+0=x	**(b)** x.1=x
Postulat 3, değişme	**(a)** x+y=y+x	**(b)** x.y=y.x
Postulat 4, dağılma	**(a)** x.(y+z)=(x.y)+(x.z)	**(b)** x+(y.z)=(x+y).(x+z)
Postulat 5	**(a)** x+x'=1	**(b)** x.x'=0

Teorem 1.(a) x+x=x olduğunu ispat edelim.

x+x= (x+x).1 **Postulat 2.b**

= (x+x).(x+x') **Postulat 5.a**

= x+(x.x') **Postulat 4.b**

= x+0 **Postulat 5.b**

= x **Postulat 2.a**

Teorem 1.(b) x.x=x olduğunu ispat edelim.

x.x= x.x+0 **Postulat 2.a**

= x.x+x.x' **Postulat 5.b**

= x.(x+x') **Postulat 4.a**

= x.1 **Postulat 5.a**

= x **Postulat 2.b**

Teorem 2.(a) $x+1=1$ olduğunu ispat edelim.

$x+1=(x+1).1$ **Postulat 2.b**

$=(x+1).(x+x')$ **Postulat 5.a**

$=x+(x'.1)$ **Postulat 4.b**

$=x+x'$ **Postulat 2.b**

$=1$ **Postulat 5.a**

Teorem 2.(b) $x.0=0$ Dualiteden açıkça görülür. İspatlanacak olursa da;

$x.0=x.0+0$ **Postulat 2.a**

$=x.0+x.x'$ **Postulat 5.b**

$=x.(x'+0)$ **Postulat 4.a**

$=x.x'$ **Postulat 2.a**

$=0$ **Postulat 5.b**

Teorem 3 $(x')'=x$, x'in tek bir tümleyeni vardır. $x+x'=1$ ve $x.x'=0$ olup hem x'in hem de x' 'nün tümleyenini göstermektedir. Dolayısıyla x' 'nün tümleyeni hem x hem de $(x')'$ olarak gösterilebilir.

Teorem 6.a $x+xy=x$ olduğunu ispat edelim.

$x+xy=x.1+xy$ **Postulat 2.b**

$=x(y+1)$ **Postulat 4.a**

$=x.1$ **Teorem 2.a**

$=x$ **Postulat 2.b**

Teorem 6.b $x(x+y)=x$ olduğunu dualite kullanmadan gösteriniz.

$x(x+y)=(x+0)(x+y)$ **Postulat 2.a**

$=x+(y.0)$ **Postulat 4.b**

$=x+0$ **Teorem 2.b**

$=x$ **Postulat 2.a**

Boole cebrine ait teoremler doğruluk tabloları yardımı ile de doğrulanabilir. Doğruluk tablolarında değişkenlerin olası kombinasyonları aranarak aralarındaki ilişkilere bakılır. Yukarıda verilmiş olan Teorem 6.a'nın doğruluk tabloları kullanılarak ispatı aşağıda verilmiştir.

x	y	xy	x+xy
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

Birleşme kuralı ve DeMorgan teoreminin ispatı oldukça uzun olup burada sadece doğruluk tabloları ile gösterilecektir.

(x+y)'=x'y' 1. DeMorgan teoremine ilişkin doğruluk tablosu

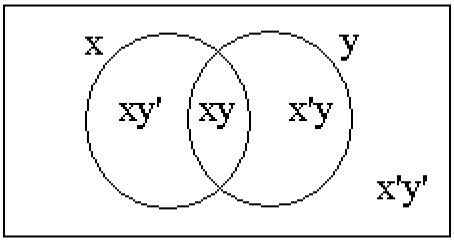
x	y	x+y	(x+y)'	x'	y'	x'.y'
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

İşlem Önceliği

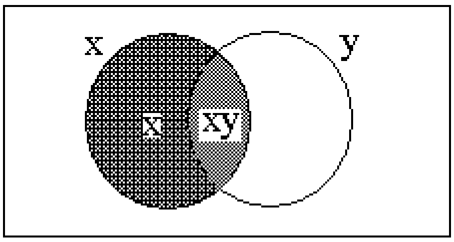
Boole cebrinde işlem önceliği sırası, Parantez (1), DEĞİL (2), VE (3), VEYA (4) şeklindedir. Bir Boolen işlemi sadeleştirileceği ya da sonucu hesaplanacağı zaman önce parantez içindeki ifadeler hesaplanmalı sonra diğer Boole işlemleri yapılmalıdır.

Venn Diaygramları

Boole aritmetiğindeki değişkenler arasındaki ilişkilerin daha kolay kavranması için kullanılan yöntemlerden bir tanesi de Venn diyagramlarıdır. Aşağıdaki şekilde x kümesine dâhil olan bölgeler x=1, dışındaki bölgelerde ise x=0 olarak kabul edilir. x'e ait olup y'ye ait olmayan bölgeler xy', y'ye ait olup x'e ait olmayan bölgeler x'y, hem x hem de y'ye ait olmayan bölgeler x'y', hem x hem de y'ye ait olan bölge ise xy ile gösterilir.



Örneğin, F=x+yx=x ifadesini göz önüne alalım. Postulatlar kullanılarak bu ifade x olarak sadeleştirilebiliyordu. Bunu, Venn diyagramları kullanarak göstermeye çalışalım.



xy bölgesi hem x hem de y'ye ait olan kesişim bölgesidir. x ile bu bölgenin toplamı (birleşimi) yine x'i vereceği diyagramdan kolaylıkla görülmektedir.

Aşağıdaki tabloda Boole cebrinin diğer gösterime türlerindeki karşılıkları görülmektedir.

Boole Cebri Gösterimi	**Venn Diyagramı Gösterimi**	**Önerme Lojiği Gösterimi**	**Anahtarlama Devresi Karşılığı**
x+y	$x \cup y$	VEYA (OR)	Paralel Devre
xy	$x \cap y$	VE (AND)	Seri Devre
x'	x' ya da x^0	DEĞİL (NOT)	Ters
0	$\varnothing$	Yanlış (FALSE)	Açık
1	$\cup$	Doğru (TRUE)	Kapalı
=	=	Lojik eşit(Logical equal)	Eşdeğer Devre

Boole Fonksiyonları

Aşağıdaki üç değişkenli Boole fonksiyonları göz önüne alalım.

F_1=xyz'

F_2=x+y'z

F_3=x'y'z+x'yz+xy'

F_4=xy'+x'z

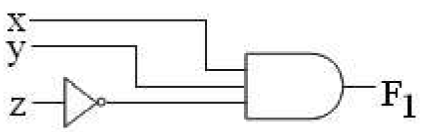
Bu verilen fonksiyonların giriş ve çıkışları arasındaki ilişkiler doğruluk tabloları yardımı ile yazılabilir.

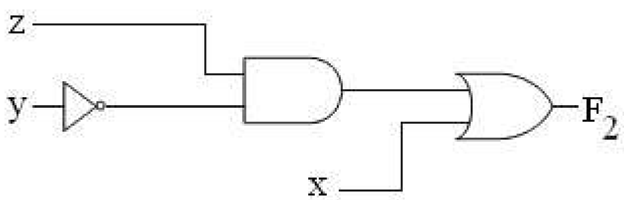
x	y	y	F_1	F_2	F_3	F_4
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	0	0	0	0
0	1	1	0	0	1	1
1	0	0	0	1	1	1
1	0	1	0	1	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	0

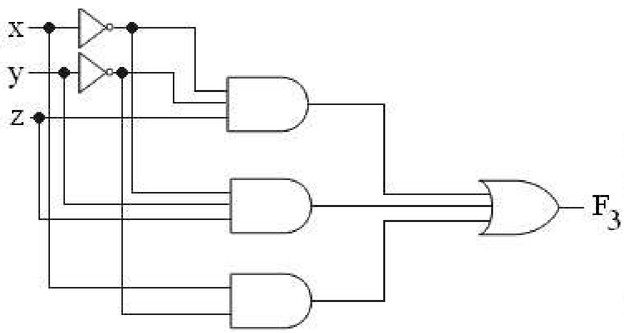
Boole cebri işlemleri verilen bir Boole fonksiyonunu sadeleştirmek ya da daha az elemanla temsil etmek için kullanılır. F_3 ve F_4'ün doğruluk tablosundaki değerlerine bakılacak olursa her ikisinin de aynı çıkışları verdiği görülür. Eğer F_3 sadeleştirilecek olursa F_4 elde edilir.

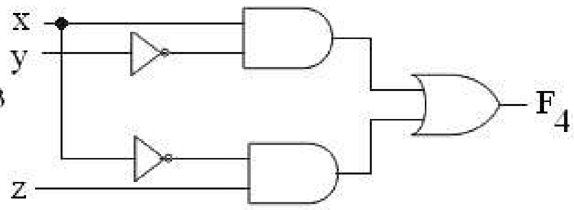
F_3=x'y'z+x'yz+xy'= x'z(y'+y)+xy'= x'z.1+xy'= x'z+xy'

Şimdi bu Boole fonksiyonlarının lojik kapılar ile gerçekleştirilmesini görelim.









Boole fonksiyonlarının minimuma indirgenmesi üzerine bazı örnekler.

Örnek: F=x+x'y Boole fonksiyonunu minimum sayıdaki değişkene indirgeyerek sadeleştiriniz.

F=x+x'y=(x'+x)(x+y)

=1.(x+y)

= x+y

Örnek: F=x(x'+y) Boole fonksiyonunu minimum sayıdaki değişkene indirgeyerek sadeleştiriniz.

F=x(x'+y)= x.x'+x.y

= 0+x.y

= xy

Örnek: F=x'y'z+x'yz+xy' Boole fonksiyonunu minimum sayıdaki değişkene indirgeyerek sadeleştiriniz.

F=x'y'z+x'yz+xy'= x'z(y'+y)+xy'

= x'z+xy'

Örnek: F=xy+x'z+yz Boole fonksiyonunu minimum sayıdaki değişkene indirgeyerek sadeleştiriniz.

F=xy+x'z+yz= xy+x'z+yz(x'+x)

= xy+x'z+x'yz+xyz

= xy(z+1)+x'z(y+1)

= xy+x'z

Örnek: F=(x+y)(x'+z)(y+z) Boole fonksiyonunu minimum sayıdaki değişkene indirgeyerek sadeleştiriniz.

Bir önceki örneğin duali olup yukarıdaki sonucun dualinin alınması ile bulunabilir.

F=(x+y)(x'+z)(y+z)= (x+y)(x'+z)

Fonksiyonun Tümleyeni

Bir F fonksiyonunun tümleyeni F' olup doğruluk tablosunda 1'lerin yerine 0, 0'ların yerine 1 konulması ile elde edilir. Bir fonksiyonunun tümleyeni DeMorgan teoremi kullanılarak cebirsel olarak da elde edilebilir. İki değişkenli Boole cebri için DeMorgan teoremi daha önce verilmişti. Bu teorem ikiden fazla değişkenli cebirsel ifadeler için de kullanılabilir.

$$
\begin{aligned}
(A+B+C)' &= (A+X)' \\
&= A'.X' \qquad X=B+C \\
&= A'.(B+C)' \\
&= A'.B'.C'
\end{aligned}
$$

Daha çok değişkenli fonksiyonlar için de aynı mantık uygulanacak olursa,

$(A+B+C+D+E+F)'=A'.B'.C'.D'.E'.F'$

$(A.B.C.D.E.F)'= A'+B'+C'+D'+E'+F'$

DeMorgan teoreminin genelleştirilmiş haline göre önce VE ve VEYA işlemleri kendi aralarında yer değiştirir, sonra her bir ifadenin tümleyeni alınır.

Örnek: $F_1=x'yz'+x'y'z$ ve $F_2=x(y'z'+yz)$

$F_1' = x'yz'+x'y'z=(x'yz')+(x'y'z)=(x+y'+z)(x+y+z')$

$F_2' = x(y'z'+yz)=(x)+(y'z'+yz)=x'+[(y+z).(y'+z')]$

Duallerini alıp elemanların tümleyenlerini alacak olursak;

$F_1' = (x'yz')+(x'y'z) \Rightarrow (x'+y+z')(x'+y'+z) \Rightarrow (x+y'+z)(x+y+z')$

$F_2' = x(y'z'+yz)=(x)+(y'+z')(y+z) \Rightarrow x'+[(y+z)(y'+z')]$

KANONİK VE STANDART BİÇİMLER

Minterim ve Maksterim

İkili değişkenler normal (x) ya da tümleyen (x') formunda bulunurlar. VE işlemi ile birleştirilmiş x ve y ikili değişkenleri xy, x'y, xy' ve x'y' kombinasyonlarından birini alırlar. Bu terimlere minterim ya da standart çarpım denir. n adet değişken birleştirilerek 2^n adet minterim elde edilebilir. Yöntem 1'li açılım olarak da bilinir. Önce, doğruluk tablosu oluşturulur doğruluk tablosunda sonucu 1 olan satırlardaki değişkenler eğer 0 ise tümleyenleri (değilleri), 1 ise kendisi alınarak değişkenler birbiri ile çarpılır (VE'lenir) sonra bu çarpımların tümü toplanır (VEYA'lanır).

Maksterimler bulunurken, sonucu 0 olan satırlardaki değişkenlerin değeri 0 ise değiştirilmeden, değeri 1 ise tümleyenleri alınarak toplanır ve her satır toplamı bir biri ile çarpılır. Bu yöntem maksterimler, standart toplam ya da 0'lı açılım olarak da bilinir. Her bir maksterimin kendisine karşı düşen minterimin tümleyeni olup, aynı şekilde bunun tersi de doğrudur.

Üç değişkenli maksterim ve minterim tablosu

			Minterimler		Maksterimler	
x	y	z	Terim	Sembol	Terim	Sembol
0	0	0	x'y'z'	m_0	x+y+z	M_0
0	0	1	x'y'z	m_1	x+y+z'	M_1
0	1	0	x'yz'	m_2	x+y'+z	M_2
0	1	1	x'yz	m_3	x+y'+z'	M_3
1	0	0	xy'z'	m_4	x'+y+z	M_4
1	0	1	xy'z	m_5	x'+y+z'	M_5
1	1	0	xyz'	m_6	x'+y'+z	M_6
1	1	1	xyz	m_7	x'+y'+z'	M_7

Üç değişkenli iki fonksiyon

x	y	z	f_1	f_2
0	0	0	0	0
0	0	1	1	0
0	1	0	0	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

Üç değişkenli f_1 ve f_2 fonksiyonlarının doğruluk tablolarını göz önüne alalım. f_1 fonksiyonunu 1 yapan satırlar x'y'z, xy'z', xyz 'dir. Bu minterimlerden her biri 1 sonucunu verir. Dolayısıyla, f_1= x'y'z + xy'z' + xyz $=m_1+m_4+m_7$ ayrıca $f_1=M_0.M_2.M_3.M_5.M_6$ şeklinde de yazılır.

Benzer şekilde f_2 fonksiyonu için;

f_2=x'yz+xy'z+xyz'+xyz$=m_3+m_5+m_6+m_7$ ve ayrıca $f_2=M_0.M_1.M_2.M_4$ şeklinde de ifade edilebilir.

Her Boole fonksiyonu minterimlerin toplamı olarak ifade edilebilir.

Bu f_1 ve f_2 fonksiyonlarının tümleyenlerini bulmak için değeri 0 olan satırlara göre minterimler oluşturulur ve bunlar VEYA'lanır.

$f_1' = x'y'z' + x'yz' + x'yz + xy'z + xyz' = m_0+m_2+m_3+m_5+m_6$

DeMorgan teormine göre tümleyenin tümleyeni yine kendisi olduğuna göre

$f_1 =(x+y+z)(x+y'+z)(x+y'+z')\ (x'+y+z)\ (x'+y'+z)=M_0.M_2.M_3.M_5.M_6$

Benzer şekilde f_2 fonksiyonun tümleyeni,

$f_2' = m_0+m_1+m_2+m_4$

DeMorgan teormine göre tümleyenin tümleyeni yine kendisi olduğuna göre

$f_2=M_0.M_1.M_2.M_4$

Bu örneklerden de görüldüğü üzere her fonksiyon maksterimlerin çarpımı ile gösterilebilir. Bu gösterimlere kanonik gösterim adı verilir. Herhangi bir fonksiyonu minterimlerin toplamı şeklinde ifade etmek için, eksik olan terimin x olduğu varsayılırsa ifade (x+x') ile çarpılarak eksik terimler giderilir ve fonksiyon minterimlerin toplamı şeklinde ifade edilir. Aşağıdaki örnekte bu durum daha iyi bir şekilde açıklanmıştır.

Örnek:

F=A+B'.C fonksiyonunu minterimlerin toplamı şeklinde yazınız.

A değişkeninde eksik olan değişkenler B ve C 'dir.

A=A.(B+B')=AB+AB'

A=A.B.(C+C')+AB'(C+C')=ABC+ABC'+AB'C+AB'C'

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

$F=A+B'C= ABC+ABC'+AB'C+AB'C'+ \cancel{AB'C}+A'B'C$

AB'C terimi iki kere tekrarlandığı için (x+x=x olduğundan dolayı) bunu bir kere yazmak gerekir. Tekrar düzenlenecek olunursa,

$F=A+B'C= A'B'C + AB'C' +AB'C+ABC'+ ABC =m_1+m_4+m_5+m_6+m_7$

Boole fonksiyonu minterimlerin toplamı şeklinde verilmiş ise aşağıdaki şekilde daha kısa bir biçimde gösterilebilir.

$$F(A,B,C) = \sum(1,4,5,6,7)$$

$\sum$ sembolü terimlerin VEYA'landığı anlamına gelir, izleyen sayılar ise fonksiyonların minterimleridir. Parantez içindeki değişkenler ise sıra ile verilmiş olup minterimin VE işlemine dönüştürülmesi sırasında kullanılacak olan sırayı verir.

Boole fonksiyonlarının minterimlerini elde etmenin bir yolu da doğrudan fonksiyonda değerleri yerine koymaktır. F=A+B'C fonksiyonunu göz önüne alalım. B'C ifadesi ancak B=0 ve C=1 iken 1 olur F ifadesi A=1 olduğu sürece 1 ‘dir. Bunları göz önüne alarak doğruluk tablosu kolaylıkla oluşturulabilir.

Verilen bir fonksiyonu maksterimlerin çarpımı şeklinde yazabilmek için fonksiyon, önce VEYA’lı terimlerin çarpımı haline getirilmelidir. Bunun için x+yz=(x+y)(x+z) dağılma kuralı uygulanır. Daha sonra eksik terimler yerine de x.x' terimi konur.

Örnek:

F=x.y+x'.z fonksiyonunu maksterimlerin çarpımı şeklinde ifade ediniz.

Önce dağılma kuralı uygulanarak fonksiyonu VEYA’lı terimlere dönüştürelim.

F= x.y+x'.z= (xy+x')(xy+z)	(dağılma kuralı)
= (x'+xy)(z+xy)	(yer değiştirme kuralı)
= (x'+x)(x'+y)(z+x)(z+y)	(dağılma kuralı)
= (x'+y)(z+x)(z+y)	(x+x'=1)

Her bir terimde eksik olan değişkenler eklenirse,

(x'+y)=(x'+y)+z.z'=(x'+y+z)(x'+y+z')

(x+z)= (x+z)+y.y'=(x+y+z)(x+y'+z)

(y+z)= (y+z)+x.x'=(x+y+z)(x'+y+z)

Aynı olan terimler x.x=x olduğunda dolayı tekrarlanan terimlerden sadece bir tanesi alınır.

F= x.y+x'.z= (x+y+z) (x+y'+z) (x'+y+z) (x'+y+z')=$M_0.M_2.M_4.M_5$

Bu fonksiyonu daha kısa bir şekilde aşağıdaki gibi elde edebiliriz.

$$F(x,y,z)=\prod(0,2,4,5)$$

Kanonik Yapılar Arasında Dönüşüm

Minterimlerin toplamı şeklinde ifade edilen bir fonksiyonun tümleyeni, orijinal fonksiyonda olmayan minterimlerin toplamına eşittir.

$F(A,B,C)=\sum(1,4,5,6,7)$ fonksiyonunun tümleyeni $F'(A,B,C)=\sum(0,2,3)$ ’tür. F' ‘nün tümleyeni F, DeMorgan teoremi kullanılarak aşağıdaki gibi elde edilir.

$$F=(m_0+m_2+m_3)'=m_0'.m_2'.m_3'=M_0M_2M_3=\prod(0,2,3)$$

Buradan $m_j'=M_j$ olduğu görülür.

$$\sum(1,4,5,6,7)=\prod(0,2,3)$$

Diğer Lojik İşlemler

n=2 için VE ve VEYA lojik işlemlerinden başka 14 tane daha lojik işlem vardır. İki değişkene göre bu lojik işlemler arasındaki bağıntılar tablodaki gibi verilir.

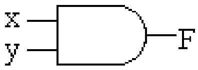
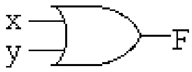
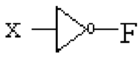
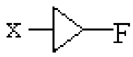
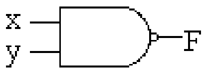
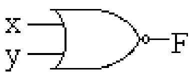
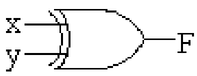
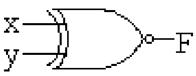
İki değişkene ait 16 fonksiyonun Boole ifadeleri

Boole Fonksiyonları	İşlem Sembolü	İsim	Açıklama
$F_0=0$		SIFIR	İkili sabit sıfır
$F_1=xy$	$x.y$	VE	x ve y
$F_2=xy'$	x/y	Yasaklama	x fakat y değil
$F_3=x$		İletim	x
$F_4=x'y$	y/x	Yasaklama	y fakat x değil
$F_5=y$		İletim	y
$F_6=xy'+x'y$	$x \oplus y$	Özel VEYA	x veya y fakat ikisi de değil
$F_7=x+y$	$x + y$	VEYA	x veya y
$F_8=(x+y)'$	$x \downarrow y$	VEYA DEĞİL	VEYA- değil
$F_9=xy+x'y'$	$x \odot y$	Eşdeğer	x eşit y
$F_{10}=y'$	y'	Tümleyen	y değil
$F_{11}=x+y'$	$x \subset y$	İçerme	y, x’ten sonra ise
$F_{12}=x'$	x'	Tümleyen	x değil
$F_{13}=x'+y$	$x \supset y$	İçerme	x, y’den sonra ise
$F_{14}=(xy)'$	$x \uparrow y$	VE DEĞİL	VE- değil
$F_{15}=1$		Bir	ikili sabit bir

İki değişkenli bu 16 fonksiyona ait doğruluk tablosu aşağıdaki gibi verilir.

x	y	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
İşlem Sembolü			.	/		/		$\oplus$	$+$	$\downarrow$	$\odot$	'	$\subset$	'	$\supset$	$\uparrow$	

Yasaklama ve içerme fonksiyonları bilgisayar lojiğinde çok nadiren kullanılır, sadece mantık bilimi ile uğraşanlar tarafından kullanılır. Ayrıca bu iki fonksiyonun yapısı dağılma ve birleşme kuralına uymadıklarından dolayı lojik kapılar ile gerçekleştirilmeye uygun değildirler. Aşağıdaki şekilde elektronik devre elemanları ile gerçekleştirilebilen 8 tane lojik kapı ve doğruluk tabloları görülmektedir. Tampon devresi olarak adlandırılan eleman sinyalin iletimi sırasında zayıflamaması için kullanılır. İki tane eviricinin peş peşe bağlanması ile elde edilir. VEDEĞİL ve VEYADEĞİL kapılarının VE ve VEYA kapılarından daha fazla kullanılır. Bunun nedeni, bu kapıların transistör ile kolayca gerçekleştirilmeleridir.

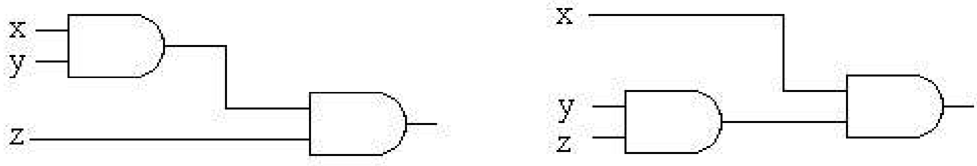
İsim	Grafik Sembol	Cebirsel Fonksiyon	Doğruluk Tablosu	
VE (AND)		$F=xy$	x y 0 0 0 1 1 0 1 1	F 0 0 0 1
VEYA (OR)		$F=x+y$	x y 0 0 0 1 1 0 1 1	F 0 1 1 1
DEĞİL (NOT)		$F=x'$	x 0 1	F 1 0
TAMPON (BUFFER)		$F=x$	x 0 1	F 0 1
VE DEĞİL (NAND)		$F=(xy)'$	x y 0 0 0 1 1 0 1 1	F 1 1 1 0
VEYA DEĞİL (NOR)		$F=(x+y)'$	x y 0 0 0 1 1 0 1 1	F 1 0 0 0
ÖZEL VEYA (XOR- EOR)		$F=xy'+x'y$ $=x\oplus y$	x y 0 0 0 1 1 0 1 1	F 0 1 1 0
ÖZEL VEYA DEĞİL (XNOR)		$F=xy+x'y'$ $=x\odot y$	x y 0 0 0 1 1 0 1 1	F 1 0 0 1

Kullanılan bu kapıların girişleri birden fazla olabilir. Örneğin üç girişli bir VE kapısı düşünelim.

$x.(y.z)=(x.y).z$ birleşme özelliği

$x.y=y.z$ değişme özelliği

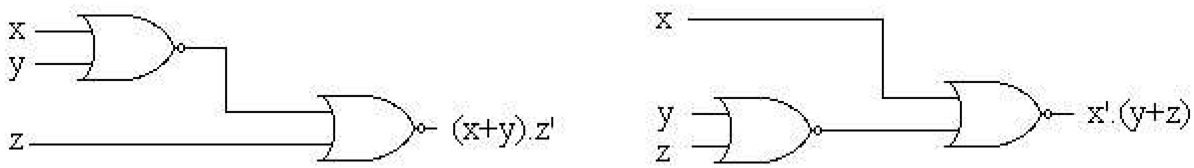
VE işlemi bu iki özelliği de gösterdiği için bağlantı sırasındaki herhangi bir değişikliğin önemi yoktur.



Fakat VEDEĞİL ve VEYADEĞİL kapıları dikkate alınacak olursa VEYADEĞİL için, $x\downarrow$ $(x\downarrow y)\downarrow z \neq x\downarrow(y\downarrow z)$ özelliğini sağlamaz.

$(x\downarrow y)\downarrow z=[(x+y)'+z]'=(x+y).z'$

$x\downarrow(y\downarrow z)=[x+(y+z)']'=x'(y+z)$



Dolayısıyla parantezlerin yerleri oldukça önemlidir ve dikkat edilmelidir.

$x\downarrow y\downarrow z=(x+y+z)'$

aynı şekilde

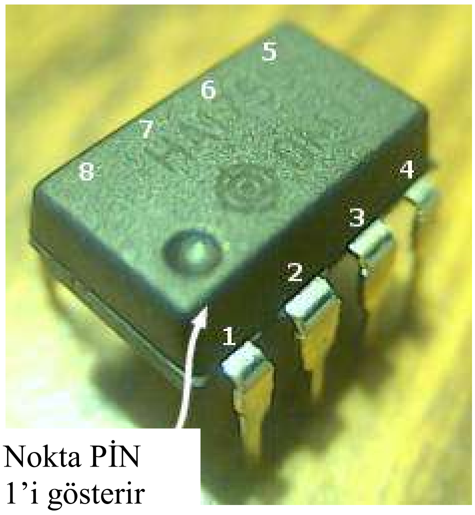
$x\uparrow y\uparrow z=(xyz)'$

DeMorgan teoreminden de görüldüğü üzere çarpımların toplamı şeklindeki ifadeler VEDEĞİL kapıları ile gerçekleştirilebilmektedir.

$F=[(ABC)'(DE)']'=ABC+DE$

ÖZELVEYA fonksiyonu tek bir fonksiyondur. Girişlerinde tek sayıda 1 varsa çıkışı 1 seviyesindedir. Çok girişli ÖZELVEYA kapılarının kullanımı pek yaygın değildir. Hatta iki girişli yapılar bile diğer kapılarla oluşturulur.

Sayısal devreler tümdevreler ile oluşturulur. Tümdevreler(IC), yonga (chip) adı verilen ve silisyumdan yapılmış küçük yarıiletken kristallerdir. Değişik tipte kapılar istenilen devreyi oluşturmak üzere bir araya getirilerek seramik ya da plastik bir kılıfa monte edilir ve devrelerin giriş çıkışı uçları olarak adlandırılan bacakları da (pin) bu kılıftan dışarıya çıkartılır. Aşağıdaki şekilde bir tümdevre ve bunun pinlerinin nasıl numaralandırıldığı görülmektedir.

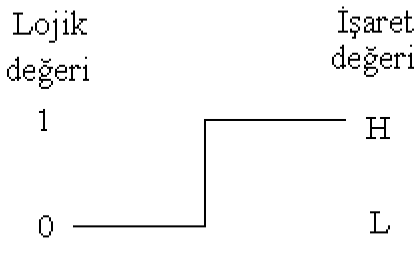


Sayısal tümdevreler dört tümleşiklik seviyesine ayrılırlar. Bu tümleşiklik seviyelerine yongada kullanılan kapı sayılarına göre karar verilir.

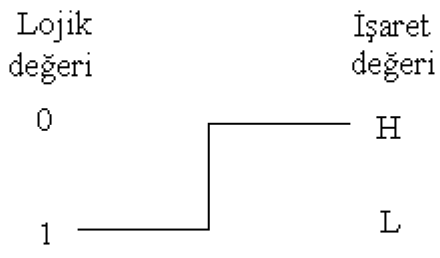
Küçük Ölçekte Tümleşiklik (*Small Scale Integration*, SSI) SSI tümdevreleri bir elemanda birden fazla bağımsız kapı içerir. Kapıların giriş ve çıkışları pinler yardımı ile doğrudan dışarıya çıkartılmıştır. Kapı sayısı genellikle 10'dan az olduğu için bacak sayıları sınırlıdır.
Orta Ölçekte Tümleşiklik (*Medium Scale Integration*, MSI) MSI tümdevreleri bir elemanda 10 ila 100 arasıda kapı içerirler. Genellikle kod çözücü, toplama elemanı, veya veri seçici (multiplexer) gibi belli temel işlevleri yerine getirirler.
Yüksek Ölçekte Tümleşiklik (*Large Scale Integration*, LSI) LSI tümdevreleri 100 ile birkaç bin kapı içerirler. Bu elemanlar genellikle, işlemci, bellek ve programlanabilir lojik elemanları gibi sayısal elemanlar içerirler.
Çok Yüksek Ölçekte Tümleşiklik (*Very Large Scale Integration*, VLSI) VLSI tümdevreleri ise tek bir elemanda binlerce kapı içerirler. Bu elemanlara örnek olarak karmaşık mikrobilgisayarlar gösterilebilir.
Sayısal tümdevreler sadece karmaşıklıklarına göre değil ayrıca yapıldıkları teknolojiye göre de sınıflandırılırlar. Bu devre teknolojisi sayısal lojik aile olarak adlandırılır. Her lojik ailenin kendine özgü bir temel elektronik devresi vardı ve daha karmaşık aileler buradan hareketle gerçekleştirilir. Her teknolojide temel devre VEDEĞİL, VEYADEĞİL ve TAMPON

kapısıdır. Temel devrenin yapısında kullanılan elektronik elemanlar genellikle söz konusu sayısal aileye de adını verir.
TTL- Transistör - Transistör Lojik
ECL- Emetör Kuplajlı Lojik
MOS- Metal Oksit Yarıiletken
CMOS Tamamlayıcı Metal Oksit Yarıiletken

Transistör-transistör lojik ailesi VEDEĞİL kapısını transistör ve diyottan oluşan bir teknolojiden geliştirilmiş olan bir türdür. Devre ilk önceleri DTL olarak adlandırılırken devredeki diyotlar, devrenin çalışmasını iyileştirmek için transistörlerle değiştirilmiş ve aile TTL olarak adlandırılmıştır.
Emetör kuplajlı lojik (ECL), tümdevre halindeki sayısal lojik aileler içindeki en hızlı olanlardır. ECL, yüksek hızın gerekli olduğu süper bilgisayarlar ve gerçek zamanlı sayısal işaret işleyicilerde kullanılırlar. ECL teknolojisinde kullanılan transistörler doymaya sokulmazlar. Dolayısıyla 1 ya da 2 Ns zaman gecikmeleri sağlanır.
Metal oksit yarıiletkenler sadece bir tip taşıyıcı hareketine dayalı transistörlerdir. Bu taşıyıcılar elektron (n-kanallı) ya da delik (p-kanallı) olabilir. Bunlar NMOS ve PMOS olarak adlandırılırlar. Tümleyici MOS da, bir tane NMOS ve bir tane PMOS kullanılarak elde edilir. MOS'ların bipolar transistörlere göre en büyük avantajları, üretim sırasında daha az teknik gerektirmesi, az güç kayıplarının olması ve yüksek yoğunluk kapasiteleri göstermeleridir.
Lojik işaretler pozitif ve negatif lojik olmak üzere ikiye ayrılırlar. Aşağıdaki şekilde pozitif ve negatif lojik işaretler görülmektedir.



Pozitif Lojik



Negatif Lojik

Negatif lojik bir işaret VEYA kapısının girişine uygulandığında VE kapısına pozitif lojik bir işaret uygulanmış gibi olur. Bir başka değişle, birbirlerine göre tümleyenlerdir.

BOOLE FONKSİYONLARININ SADELEŞTİRİLMESİ

Boole fonksiyonlarını oluşturan sayısal lojik kapıların karmaşıklığı doğrudan doğruya fonksiyonu oluşturan cebirsel ifadenin karmaşıklığına bağlıdır. Her fonksiyonun doğruluk tablosu gösterimi tektir fakat cebirsel olarak birçok farklı yolla gösterilebilir. Boole fonksiyonları daha önce anlatılan cebirsel yollarla sadeleştirilebilir ama bu yöntemin sistematik kuralları olmadığından dolayı kullanışlı değildir. Diyagram metodu, Boole fonksiyonlarının minimizasyonu için kullanılan basit bir yöntemdir. Bu yöntem doğruluk tablosunun şekilleştirilmiş bir biçimi ya da Venn diyagramının gelişmiş bir şekli olarak da düşünülebilir. Bu diyagram metodu Veitch ya da Karnaugh diyagramı olarak da bilinir.

Diyagram karelerden oluşmuştur ve her bir kare minterimi temsil eder. Her Boole fonksiyonu minterimlerin toplamı şeklinde ifade edilebildiği için fonksiyonun içerdiği minterimleri kapsayan karelerden hareketle fonksiyon elde edilebilir. Aslında bu diyagram bir fonksiyonun standart formda ifade edilebileceği tüm şekilleri sunan görsel bir diyagramdır. Bu diyagram yardımıyla alternatif cebirsel ifadeler elde edilerek bunların en basit olanı seçilebilir. Burada, en basit cebirsel ifadenin minimum sayıda değişkenin çarpımlarının toplamı ya da toplamlarının çarpımı şeklinde ifade edileceği varsayılacaktır.

İki ve Üç Değişkenli Diyagramlar

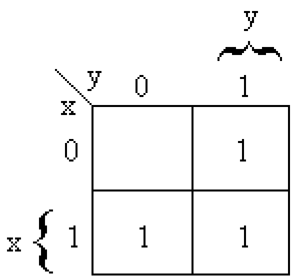
Aşağıdaki tablo minterimlerin yerleştirilmesi ile oluşturulmuş bir diyagramdır.

m_0	m_1
m_2	m_3

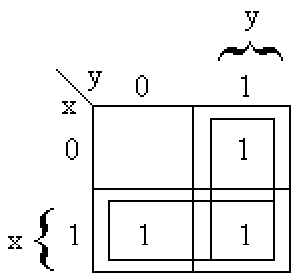
Bu minterimler yerleştirilirken satır ve sütunlar aşağıdaki gibi lojik değişkenler ile temsil edilirler.

x \ y	0	1 (y)
0	x'y'	x'y
x { 1	xy'	xy

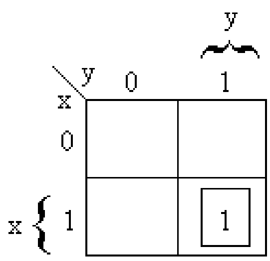
x+y lojik işlemini göz önüne alalım. $x+y=x'y+xy'+xy=m_1+m_2+m_3$ ile minterimlerin toplamı şeklinde temsil edilir. Diyagram oluşturulurken fonksiyonun doğruluk tablosunda lojik 1 olan yerlere bu minterimler aşağıdaki gibi yerleştirilir.



Burada bir birine komşu olan ve 1'lerden oluşan ikili ya da dörtlü kareler göz önüne alınmalıdır. Bu komşuluklar aşağıda dikdörtgen içine alınarak gösterilmiştir. Daha sonra bu komşuluklardan yararlanarak fonksiyon sadeleştirilebilir. İkinci satırı düşünelim. Bu satırın değişkenleri arasındaki ilişki şu şekildedir. İkinci satır birinci sütun elemanı xy' temsil ederken ikinci satır ikinci sütun elemanı da xy 'yi temsil eder. Bu ikisinin toplamı xy'+xy=x(y'+y)=x.1=x olur. Aynı şekilde, ikinci sütun birinci satır x'y temsil ederken ikinci sütun ikinci satır ise xy 'yi temsil eder. bu ikisinin toplamı x'y+xy=y(x'+x)=y olur. Bu iki ifadenin toplamı x+y lojik ifadenin minimum toplamlar şeklindeki en basit ifadesidir. Genel olarak bir yöntem ortaya konulacak olursa; yatay satır için, x değişmezken y değişkeni değişiyor ve toplama kendisi ve değili geldiği için toplamı 1 olur dolayısıyla yatay satırda x değişkeninin sadece kendisi kalır, değeri de 1 olduğu için doğrudan kendisi alınır. Dikey sütun için ise y'nin değeri değişmezken x ve x'in değili toplama girdiğinden x etkisiz kalır ve sadece y 'nin kendisi işleme girer o sütundaki y değeri de 1 olduğu için direkt olarak y değeri alınır ve bu iki değer toplanarak x+y şeklinde lojik ifadenin en sade hali elde edilir.



x.y lojik işlemini göz önüne alalım. Doğruluk tablosundan aşağıdaki diyagram elde edilir.



İkinci sütun ve satır elemanı sadece lojik 1 olduğundan sadece bu eleman 1 olup x ve y değişmediğinden ve değerli bir olduğundan bu ikisinin VE 'lenmesi ile fonksiyon sadeleştirilebilir.

Üç değişkenli diyagramlar için değişkenler yerleştirilirken sıralı ikililer yerine aşağıdaki tablolarda görüldüğü gibi bunların gray kodları kullanılır.

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6

			y	y
$x \backslash xz$	00	01	11	10
0	$x'y'z'$	$x'y'z$	$x'yz$	$x'yz'$
x 1	$xy'z'$	$xy'z$	xyz	xyz'
		z	z	

Birbirine komşuluğu olan karelerden geçişlerde değişkenin değili göz önüne alındığı için değeri değişen değişkenin, fonksiyonun sadeleştirilmesi sırasında herhangi bir etkisi olmaz. Örneğin, m_1 ve m_3 karelerini gözüne alalım. x'y'z ve x'yz ile belirlenir. Minterimler toplamı şeklinde belirtecek olursak x'y'z+x'yz=x'z(y+y')=x'z ile belirlendiği görülür. Dolayısıyla, komşu kareler arasındaki geçişlerde değişken değeri değişiyorsa bu değişkenin etkisi (x+x')=1 olduğu için herhangi bir etkisi olmayacaktır. Değişken değişmiyorsa, örneğin m_1 ve m_3 kareleri için düşünecek olursak, ilk satırda x değişkeni lojik 0 olduğu için bunun fonksiyona etkisi elemanın değili olarak gelecektir.

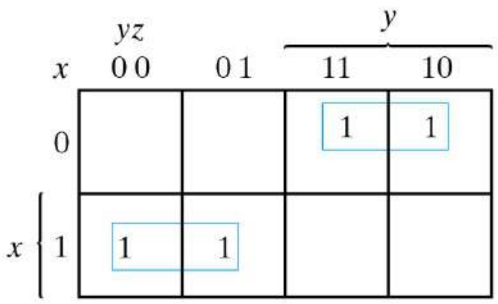
Örnek: Aşağıdaki Boole fonksiyonunu sadeleştiriniz.

$F(x, y, z) = \sum(2,3,4,5)$

Önce fonksiyonu temsil eden her minterime karşılık gelecek şekilde 1 konur. Bunlar, m_2, m_3, m_4 ve m_5 minterimleridir.

			y	y
$x \backslash yz$	00	01	11	10
0			1	1
x 1	1	1		

1'lerden oluşmuş en az iki tane 1'li grupları bulunur.



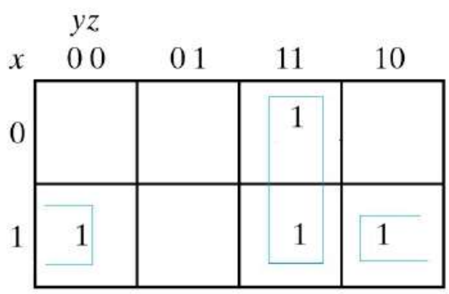
İlk satırdaki dikdörtgen ile belirtilen 1'ler şu işleme karşılık gelir. x'yz+x'yz'=x'y(z+z')=x'y 'ye karşılık gelir. Başka bir değişle, ilk satırda değişen eleman z olduğu için göz önüne alınmaz sadece y dikkate alınır, ilk satırda diğer değişmeyen eleman x olduğu için ve x değişkenin bu satırdaki değeri lojik 0 olduğu için değili alınır ve bu iki değişkenin VE 'lenmesi ile bu karelerin fonksiyonu bulunur. İkinci satır için ise değişmeyen eleman y ve değeri lojik 0 olduğu için y' diğer eleman x ve değeri lojik 1 olduğu için kendisi alınarak VE 'lenerek xy' elde edilir. Elde edilen iki fonksiyonun VEYA 'lanması ile yukarıdaki fonksiyonun en sade hali elde edilir.

$F(x,y,z)=\sum(2,3,4,5)=x'y+xy'$

Örnek: Aşağıdaki Boole fonksiyonunu sadeleştiriniz.

$F(x,y,z)=\sum(3,4,6,7)$

Önce fonksiyonu temsil eden her minterime karşılık gelecek şekilde 1 konur. Bunlar, m_3, m_4, m_6 ve m_7 minterimleridir.



İlk sütun ile son sütün arasında da bir komşuluk ilişki vardır. Buradaki örnekte olduğu gibi, xy'z'+xyz'=xz(y+y')=xz'. En son sütunla ilk sütunu komşu olarak düşünecek olursak, değişen eleman y olduğundan etkisi göz önüne alınmaz. Değişmeyen eleman z ve değeri lojik 0 olduğundan z', satır elemanı x olduğundan ve değeri lojik 1 olduğundan x bu ikisi VE 'lenirse xz' elde edilir. Diğer kare düşünülecek olursa, y ve z değer değiştirmiyorlar ve değerleri lojik 1 dolayısıyla, yz olarak elde edilir, x değer değiştirdiği için göz önüne alınmaz ve bu karenin fonksiyonu yz olarak elde edilir. x'yz+xyz=yz(x+x')yz. Yukarıdaki fonksiyonun en sade hali bu iki minterimin VEYA 'lanması ile elde edilir ve xz'+yz olur.

$$F(x,y,z)=\sum(3,4,6,7)=xz'+yz$$

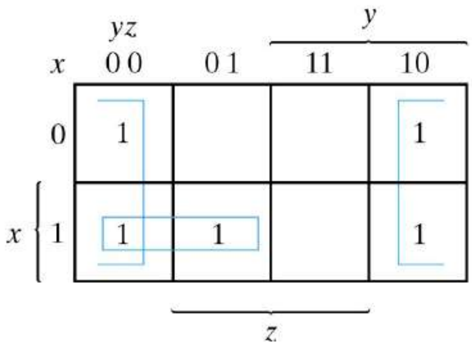
Gruplandırılan komşu karelerin sayısı daima 1, 2, 4, ve 8 gibi ikinin kuvvetleri şeklinde olmalıdır. Bu komşu karelerin sayısı arttıkça elde edilen çarpım terimindeki değişken sayısı azalır.

- Her kare bir minterime karşılık düşer ve üç değişken ile temsil edilir.
- Komşu iki kare iki değişkenli bir terime karşılık düşer.
- Komşu dört kare ise tek değişkene karşılık düşer.
- Sekiz komşu kare tüm diyagramı kapsadığı için sürekli olarak 1'e eşit bir fonksiyonu verir.

Örnek: Aşağıdaki Boole fonksiyonunu sadeleştiriniz.

$$F(x,y,z)=\sum(0,2,4,5,6)$$

Önce fonksiyonu temsil eden her minterime karşılık gelecek şekilde 1 konur. Bunlar, m_0, m_2, m_4, m_5 ve m_6 minterimleridir.



Dörtlü ikili komşuluğunda değişmeyen eleman sadece z dir ve değeri lojik 0 olduğu için z' ile temsil edilir. Diğer ikili kare y'x ile temsil edilir. Bu iki değerin toplamı fonksiyonun değerini verir.

$$F(x,y,z)=\sum(0,2,4,5,6)=xy'+z'$$

Örnek: Aşağıda verilen Boole fonksiyonunu göz önüne alalım.

F=A'C+A'B+AB'C+BC

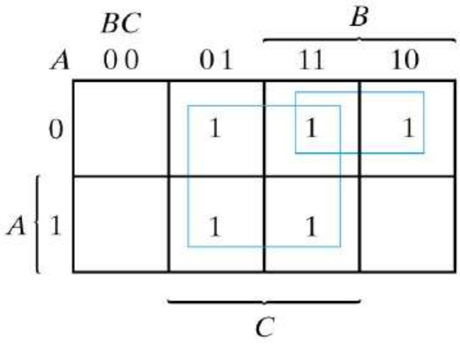
Eksik terimleri tamamlayacak olursak;

F=A'BC+A'B'C+~~A'BC~~+A'BC'+AB'C+ABC elde edilir.

Minterimler şeklinde ifade edecek olursak;

F(A,B,C)=m_3+m_1 +m_5+m_2+m_7

$$F(A,B,C)=\sum(1,2,3,5,7)$$



Dörtlü birliler tek bir değişkeni temsil eder. Tek değişmeyen eleman C ‘dir. Diğer ikili grup ise iki değişkeni temsil eder. Değişmeyen eleman B ve sütunda değişmeyen eleman A ve değeri lojik 0 olduğundan değili alınarak VE ‘lenir ve A'B elde edilir. Her ikisinin toplamı ile A'B+C elde edilir.

$$F(A,B,C) = \sum(1,2,3,5,7) = A'B + C$$

Dört Değişkenli Diyagramlar

Burada, yine sıralı ikililer yerine gray kodları kullanılacaktır. Minterimlerin diyagrama yerleşimleri ve karelerin değişkenlerle olan ilişkisi aşağıdaki gibidir.

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}

wx \ yz	00	01	11	10
00	$w'x'y'z'$	$w'x'y'z$	$w'x'yz$	$w'x'yz'$
01	$w'xy'z'$	$w'xy'z$	$w'xyz$	$w'xyz'$
11	$wxy'z'$	$wxy'z$	$wxyz$	$wxyz'$
10	$wx'y'z'$	$wx'y'z$	$wx'yz$	$wx'yz'$

Üç değişkenli diyagramlar için uygulanan yöntemin aynısı uygulanır.

Bir kare dört değişkenli bir minterimi temsil eder.

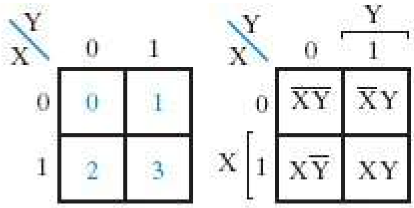
İki komşu kare üç değişkenli bir minterimi temsil eder.

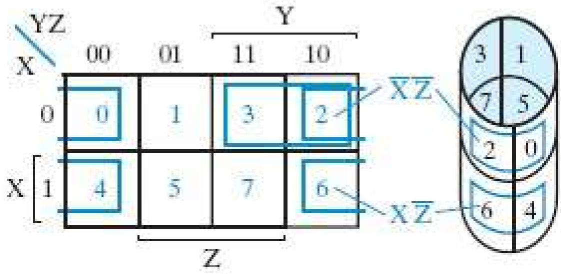
Dört komşu kare iki değişkenli bir minterimi temsil eder.

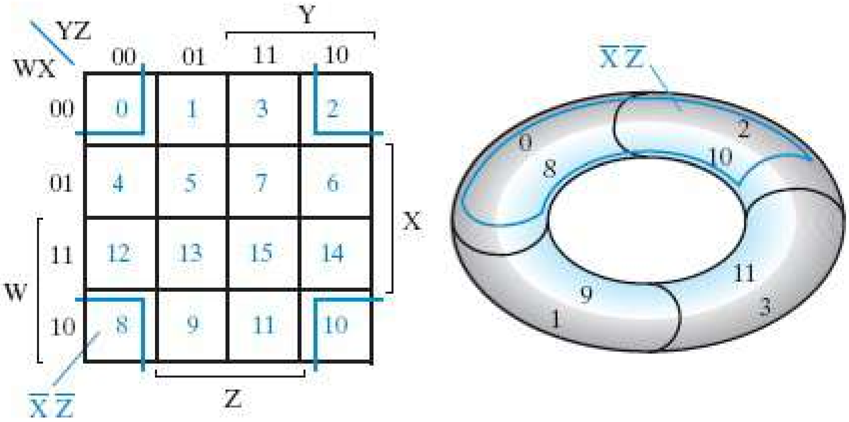
Sekiz komşu kare tek değişkenli bir minterimi temsil eder.

On altı komşu kare 1’e eşit bir fonksiyonu temsil eder.

4’lü, 8’lü ve 16’lı karaler için komşuluklar aşağıdaki şekilde gösterilmiştir.

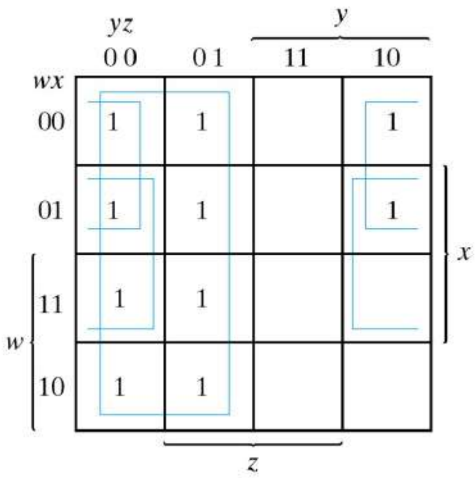






Örnek: Aşağıdaki Boole fonksiyonunu basitleştiriniz.

$$F(w, x, y, z) = \sum(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14)$$



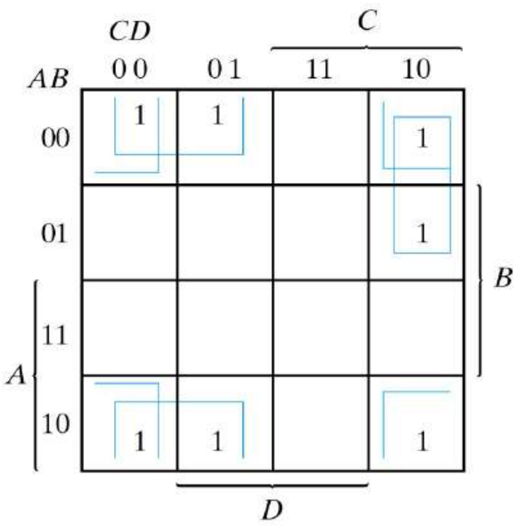
Sekizli komşu birliler tek bir değişkene karşılık gelir. Satır değişkenleri w ve x ile sütun değişkeni z değiştiği için bu sekizli komşudan sadece y fonksiyona gelir onunda değeri lojik 0 olduğu için y' olarak değer alır. İlk sütundaki ikili ile son sütundaki ikili dörtlü bir komşuluk oluşturup iki değişken ile ifade edilirler. Geçişlerde x ve y değer değiştiği için göz önüne alınmazken w ve z değer değişmediklerinden bunların çarpımı ile bir terim gelir. w değeri ve z değeri lojik 0 olduğundan değilleri alınarak çarpılır (w'z'). Diğer dörtlü de y ve w değer değiştirdiklerinden işleme katılmazlar. z ve x işleme girerler. z değişkenin o karelerdeki değeri lojik 0 olduğundan z' olarak dikkate alınır x ise değişmeden işleme girer (xz').

F(w, x, y, z)=y'+xz'+w'z'

Örnek: Aşağıdaki Boole fonksiyonunu sadeleştiriniz.

F=A'B'C'+B'CD'+A'BCD'+AB'C'

İlk minterim iki elemanlı bir kareyi temsil eder. Bu minterimde bulunmayan eleman D ‘dir. Dolayısıyla D ‘nin 0’dan 1’e geçtiği satırlarda bu minterim temsil edilir. Aynı mantıkla diğer üç değişkenli terimlerin de bulundukları kareler bulunabilir. Dört değişkenli minterim ise tek bir kare ile temsil edilir. Bu kareler aşağıdaki tabloda gösterilmiştir.

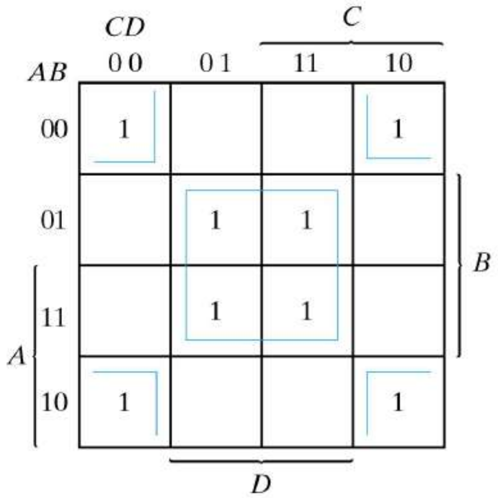


F=B'C'+A'CD'+B'D'

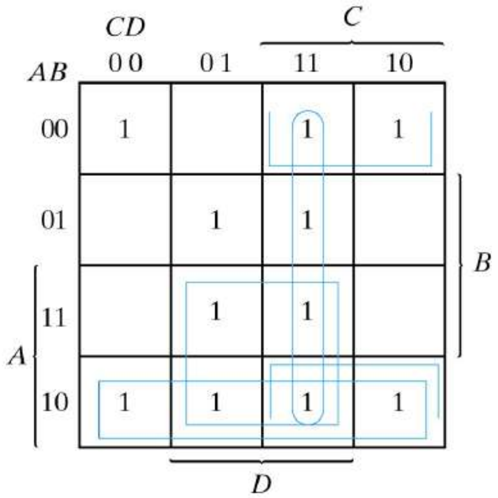
Asal Çarpanlar

Diyagram üzerinde komşu kareler seçilirken, kareler birleştirildiğinde fonksiyonun tüm minterimleri kapsadığından emin olunmalıdır. Aynı zamanda, ifadedeki terim sayının minimum olmasına ve daha önce kullanılmış terimlerin gerekmiyorsa tekrar kullanılmamasına dikkat edilmelidir. Bazen basitleştirme kriterini sağlayan iki veya daha çok sayıda ifade olabilir. Diyagramdaki kareleri gruplandırma asal çarpan ve temel asal çarpan olarak adlandırılan terimlerin anlaşılması ile daha sistematik olarak yapılabilir. Asal çarpan, diyagramda olası maksimum sayıda komşu karenin birleştirilmesi ile elde edilen bir çarpım terimidir. Bir karedeki minterim sadece bir asal çarpan tarafından kapsanıyorsa buna temel asan çarpan denir. Bu asal çarpanlar, bir Boole fonksiyonunun diyagram kullanılarak olası basitleştirilmiş ifadelerinin bulunmasında kullanılacak olup ileride detaylı olarak anlatılacaktır.

$F = (A, B, C, D) = \sum(0, 2, 3, 5, 7, 8, 9, 10, 11, 13, 15)$ Boole fonksiyonunu göz önüne alalım.



(a)



(b)

Yukarıdaki (a) şeklinde gösterilen BD ve B'D' temel asal çarpanlardır. Şekilde (b)'de ise bu temel asal çarpanlar ile örtülebilen mümkün tüm asal çarpanlar görülmektedir. Bu asal çarpanlar CD, B'C, AD ve AB' 'dır. O halde bu F fonksiyonunu farklı dört şekilde ifade etmek mümkündür.

F=BD+B'D'+CD+AD

=BD+B'D'+CD+AB'

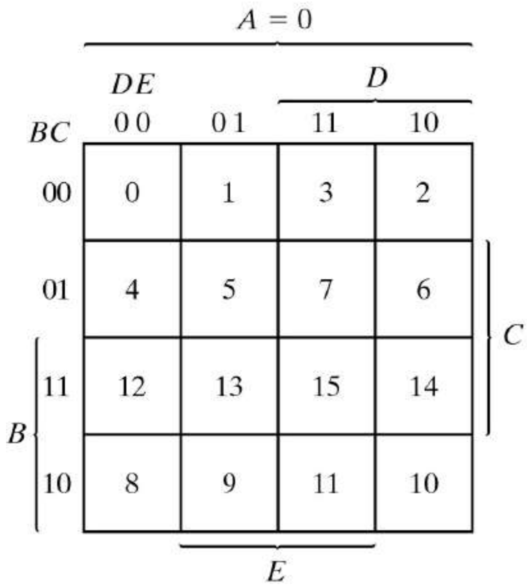
=BD+B'D'+B'C+AD

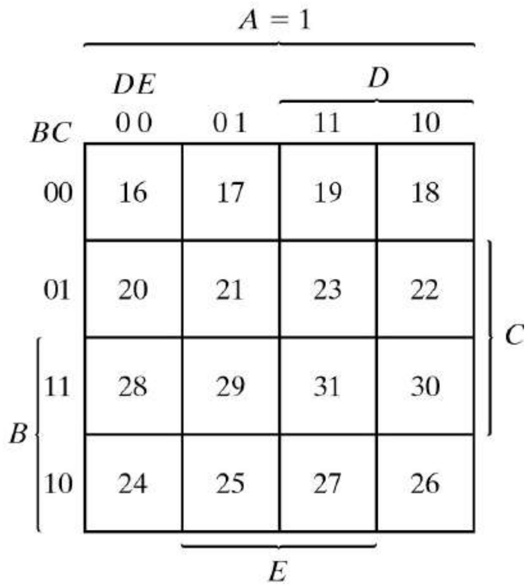
=BD+B'D'+B'C+AB'

Yukarıdaki örnekten de anlaşılabileceği gibi asal çarpanlar ifadenin sadeleştirilmesi için seçeneklerin belirlenmesinde kullanılır.

Beş Değişkenli Diyagramlar

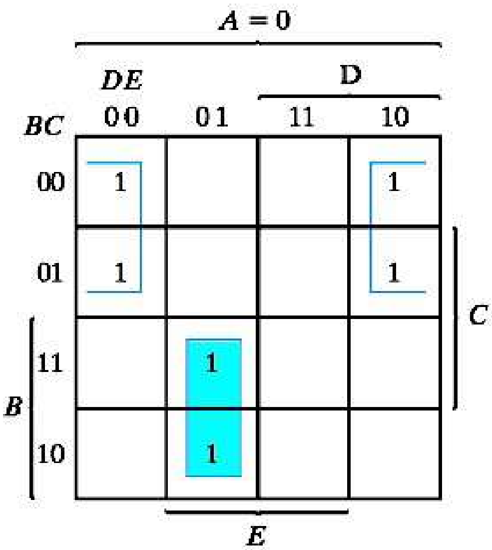
Dörtten fazla değişkenli diyagramları kareler ile ifade etmek oldukça zordur. Beş değişkenli bir diyagramda 32, altı değişkenli bir diyagramda 64 kare bulunur. Dolayısıyla, bu kareler arasındaki komşulukları da bulmak zorlaşacaktır. Beş değişkenli bir diyagramda değişkenlerden bir tanesinin değeri sürekli 0 ve 1 alınarak iki tane diyagram oluşturulur. Aşağıdaki diyagramlarda görüldüğü gibi A değişkeni diyagramlardan birinde sürekli olarak 0, diğerinde ise 1 olarak seçilmiştir. Bu diyagramlar hem kendi içlerinde hem de karşılıklı olarak komşuluklar içerirler. Örneğin m_4 minterimi m_{20} minterimine, m_{15} minterimi de m_{31} minterimine komşudur. Altı veya daha fazla değişkenli diyagramları kullanmak ve çözmek oldukça zordur. Bu nedenle Boole fonksiyonlarının basitleştirilmesi için hazırlanmış özel bilgisayar programları kullanmak gerekecektir.

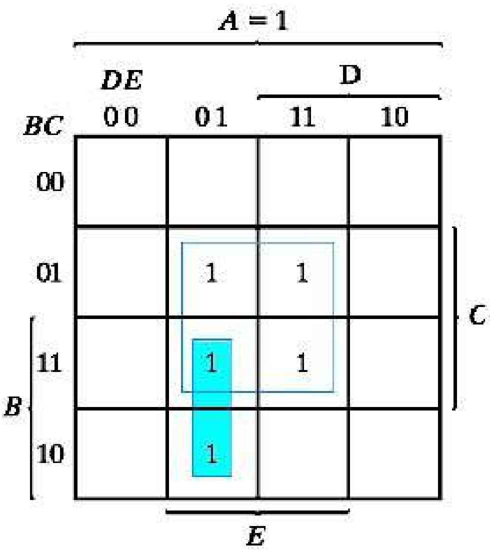




Örnek: Aşağıdaki Boole fonksiyonunu basitleştiriniz.

$F(A, B, C, D, E) = \sum(0, 2, 4, 6, 9, 13, 21, 23, 25, 29, 31)$





Diyagramlar yukarıdaki gibi oluşturulur. Soldaki diyagram basitleştirilecek olursa, A'B'E'+A'BD'E olur, sağdaki diyagram için ise ACE+ABD'E olur. Noktalı kareler her iki diyagramda komşu olduklarından bir terimden A', diğer terimden A geleceği aşikârdır. A+A'=1 olduğundan bu tür komşuluklarda A değişkeni ve terimlerden biri göz önüne alınmaz yani ABD'E ve A'BD'E terimlerindeki A değişkenleri göz önüne alınmamalı ve BD'E göz önüne alınmalıdır.

F= A'B'E'+BD'E+ ACE

Toplamların Çarpımının Basitleştirilmesi

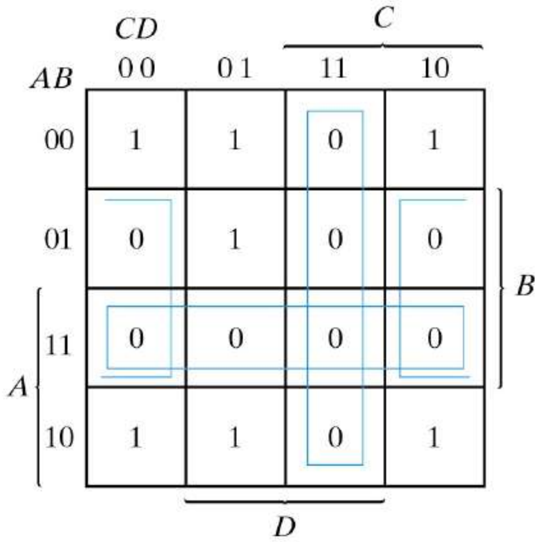
Şimdiye kadar Boole fonksiyonları çarpımların toplamı şeklinde ifade edilmişti. Bu fonksiyonları toplamların çarpımı şeklinde de ifade etmek mümkündür. Bunun için genelleştirilmiş DeMorgan teoremi kullanılır. Fonksiyonun tümleyeninin tümleyeni kendisini verdiği için fonksiyonu basitleştirirken fonksiyonun tümleyeninin basit hali bulunur. Bunun için diyagramlarda boş kalan karelere 0 konarak bu kareler arasındaki komşuluklar kullanılır ki bu da fonksiyonun tümleyenini verir. Bu fonksiyona DeMorgan teoremi uygulanırsa, ifade toplamların çarpımı haline gelmiş olur.

Örnek: Aşağıdaki Boole fonksiyonunu

(a) Çarpımların toplamı

(b) Toplamların çarpımı şeklinde basitleştiriniz.

$$F(A, B, C, D) = \sum(0,\ 1,\ 2, 5,\ 8,\ 9, 10)$$



1’ler ile temsil edilen karelerden çarpımların toplamı şeklinde aşağıdaki F fonksiyonu elde edilir.

F=B'D'+B'C'+A'C'D

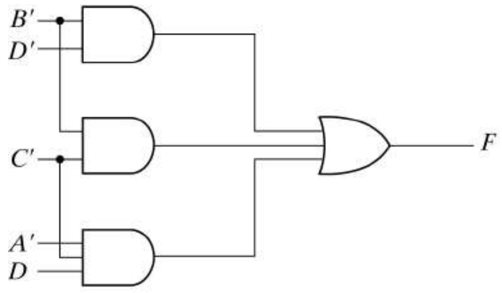
Boş kalan kareler 0’lar ile doldurulur. Bu 0’lar ile temsil edilen karelerden ise çarpımların toplamı şeklinde aşağıdaki F' tümleyen fonksiyonu elde edilir.

F'=AB+CD+BD'

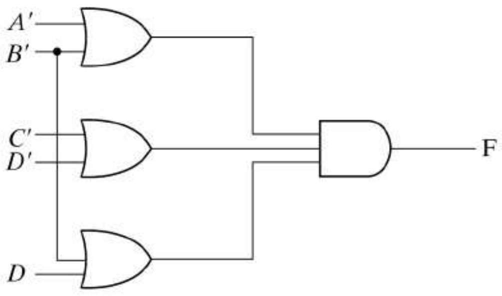
Tümleyenin tümleyenini alırsak F ’in toplamların çarpımı şeklindeki fonksiyonu elde edilir.

F=(A'+B')(C'+D')(B'+D)

Aşağıdaki şekilde görüldüğü gibi her iki fonksiyonda kapılarla gerçekleştirilmiştir. Çarpımların toplamında tüm girişler VE ‘lendikten sonra bir VEYA kapısı ile sonlandırılır. Diğer gerçekleştirmede ise tüm girişler VEYA ‘landıktan sonra bir VE kapısı ile sonlandırılır. Bu gerçekleştirmelere *iki kademeli* gerçekleştirme denir.



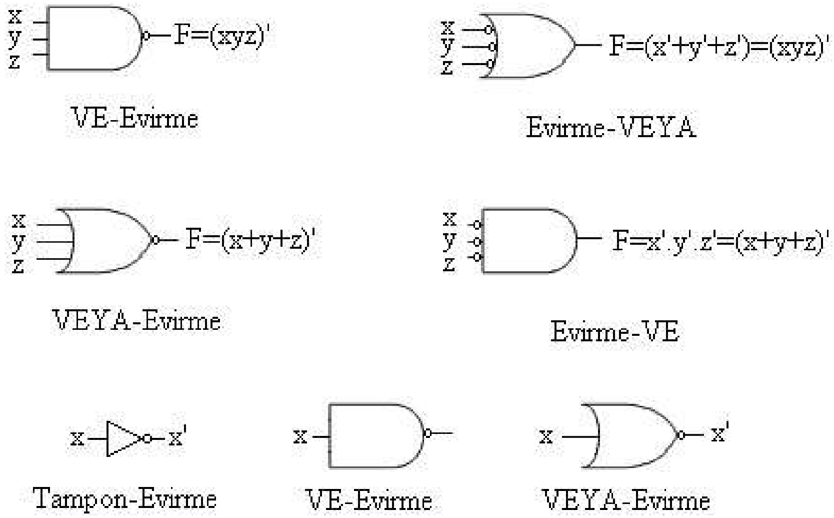
(a) $F = B'D' + B'C' + A'C'D$



(b) $F = (A' + B')(C' + D')(B' + D)$

VEDEĞİL ve VEYADEĞİL Kapılarıyla Gerçekleme

Elektronik devreler gerçekleştirilmeleri kolay olduğu için, sayısal devreler VE, VEYA, DEĞİL kapılarıyla gerçekleştirilmesi yerine VEDEĞİL, VEYADEĞİL kapılarıyla gerçekleştirilir. VEDEĞİL ve VEYADEĞİL kapılarının dönüşümü için aşağıdaki şekilleri incelemek gerekecektir.



Bu gösterimlerdeki giriş çıkış ilişkileri DeMorgan teoremi kullanılarak elde edilmiştir. Küçük daireler değişkenin tümleyeninin alındığı anlamına gelir. Her iki çıkışın da birbirinin aynı olduğu görülmektedir. Bu yapıların elektronik devreler ile gerçekleştirilmesi kolay olduğu için tasarımlarda bu tip yapılar seçilir. Tek girişli gösterilen yapılar ise bir evirici olarak çalışırlar.

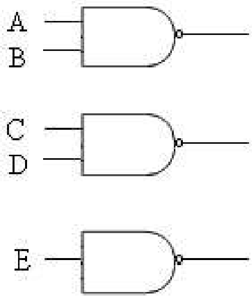
VEDEĞİL Gerçeklemesi

Boole fonksiyonlarını VEDEĞİL gerçeklemesi ile gösterebilmek için fonksiyonu çarpımların toplamı şeklinde ifade etmek gerekecektir.

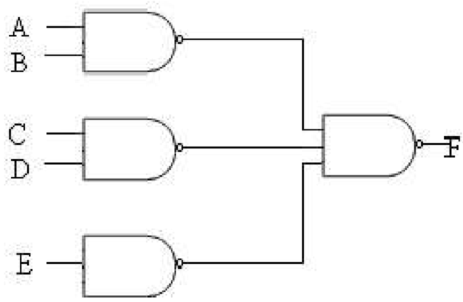
F=AB+CD+E ifadesini göz önüne alalım.

İzlenecek yol aşağıdaki gibidir.

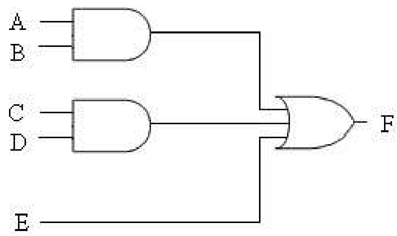
1- Önce fonksiyonu basitleştirip çarpımların toplamı olarak ifade edin.
   F=AB+CD+E
2- En az iki değişkeni olan fonksiyonun her çarpım terimi için bir VEDEĞİL kapısı alarak değişkenleri bu VEDEĞİL kapısının girişi olacak şekilde aşağıdaki gibi bağlayınız.



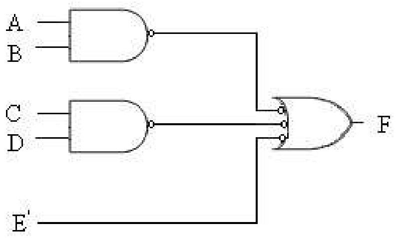
3- VE-Evirme ya da Evirme-VEYA kapılarını kullanarak girişlerine birinci kademedeki kapıların çıkışları gelecek şekilde bir VEDEĞİL kapısının girişine aşağıdaki gibi bağlayınız.



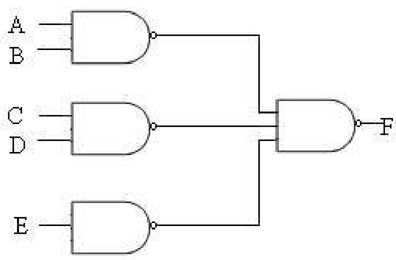
Aşağıdaki bu fonksiyonun farklı şekildeki gerçeklemeleri görülmektedir.



VE-VEYA



VEDEĞİL Evirme-VEYA



VEDEĞİL-VEDEĞİL

4- Tek değişkenli bir terim ise birinci kademede gösterilir ve bir evirici ile gerçeklenir. Ya da tümleyeni alınarak VEDEĞİL kapısının girişine bağlanır.

Diğer bir yolda, basitleştirilmiş fonksiyonu bulmak için 1'ler yerine 0'lar kullanılır. Dolayısıyla fonksiyonun tümleyeni bulunur. Bu fonksiyon gerçekleştirildikten sonra çıkışın tümleyeni alınarak fonksiyonun kendisi elde edilir. Bu gerçekleme iki çıkışlı olur ve bazen tasarımcı fonksiyonun kendisine ya da tümleyenine gerek duyabilir.

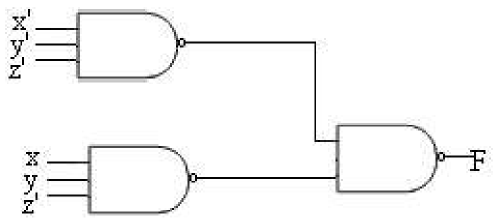
Örnek: Aşağıdaki fonksiyonu VEDEĞİL kapıları ile gerçekleştiniz.

$F(x, y, z) = \sum(0,\ 6)$

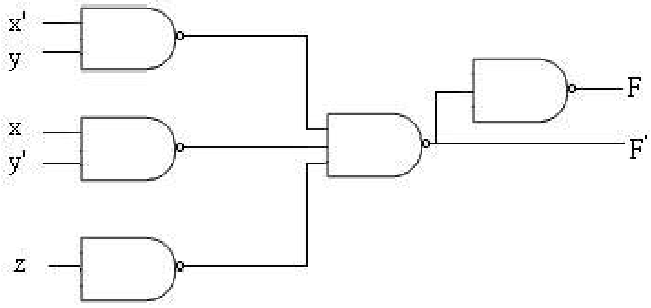
x \ yz	00	01	11	10
0	1	0	0	0
1	0	0	0	1

F'=x'y+xy'+z

F=<x'y'z'+xyz'



F=x'y'z'+xyz'



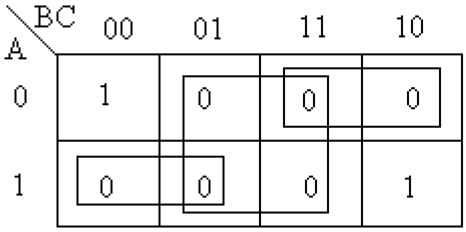
F'=x'y+xy'+z

VEYADEĞİL Gerçeklemesi

VEYADEĞİL fonksiyonu VEDEĞİL fonksiyonunun dualidir. Bu nedenle, VEDEĞİL fonksiyonundaki kural ve yöntemlerin duali VEYADEĞİL fonksiyonu için de geçerlidir. Gerçekleme için fonksiyonun toplamların çarpımı şeklinde olması gerekmektedir. Bunun için fonksiyondaki 0'lar dikkate alınarak öncelikle fonksiyonun tümleyeni bulunur. Fonksiyonun kendisi için ise elde edilen ifadenin tümleyeni alınarak ifade toplamların çarpımı şekline getirilir.

Örnek: Aşağıdaki fonksiyonu VEYADEĞİL kapıları ile gerçekleştiniz.

$F(A, B, C) = \sum(0,\ 6)$

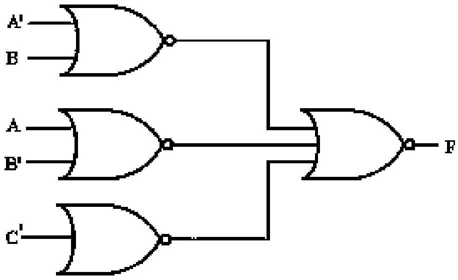


0’lardan faydalanarak

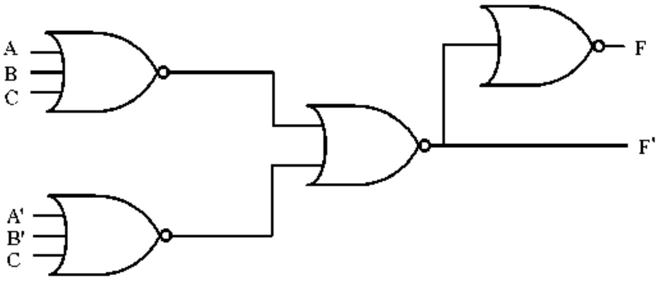
F'=A'B+AB'+C ise DeMorgan teoreminden F=(A+B')(A'+B)C'

1’lerden faydalanarak

F=A'B'C'+ABC' bulunur. DeMorgan teoreminden F'=(A+B+C)(A'+B'+C) olarak bulunur.



F=(A+B')(A'+B)C'



F'=(A+B+C)(A'+B'+C)

Aşağıdaki tabloda şimdiye kadar anlatılanlar özetlenmektedir.

DURUM	BASİTLEŞTİRİLECEK FONKSİYON	KULLANILACAK STANDART FORM	NASIL TÜRETİLECEĞİ	UYGULAMA	F İÇİN SEVİYE SAYISI
a	F	Çarpımların Toplamı	Diyagramdaki 1’lerin toplanması	VEDEĞİL	2
b	F'	Çarpımların Toplamı	Diyagramdaki 0’ların toplanması	VEDEĞİL	3
c	F	Toplamların Çarpımı	b’deki F' ‘nün tümleyeni	VEYADEĞİL	2
d	F'	Toplamların Çarpımı	a’daki F ‘in tümleyeni	VEYADEĞİL	3

Diğer İki Kademeli Gerçeklemeler

VE-VEYA-EVİRME ve VEYA-VE-EVİRME gerçeklemeleri de olup standart formları korurlar.

Etkisiz (Don’t Care) Koşullar

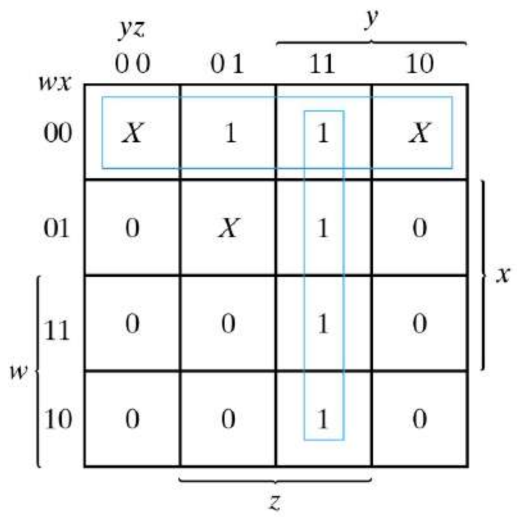
Pratikte bazı şartlarda belirsiz durumlar söz konusu olabilir. Şimdiye kadar incelenen fonksiyonlarda lojik 1 olarak belirtilen durumların dışındaki durumlar 0 olarak alınmıştı. Örneğin, dört bit ile belirlenmiş onlu sayıların belirsiz sayılan altı kombinasyonu söz konusudur. Eğer tanımlanmamış bir durum söz konusu ise bu durumun 1 ya da 0 olmasının herhangi bir önemi yoktur. Bu nedenle, fonksiyonun belirlenmemiş minterimlerine etkisiz koşullar denir. Fakat bu etkisiz koşullar fonksiyonun sadeleştirilmesi için kullanılabilir. Bu etkisiz koşullar diyagrama yerleştirilirken 1 ya da 0 olarak belirlenmez bunların yerine X işareti konulur. Diyagram sadeleştirilirken bu etkisiz elemanlar 1 ya da 0 olarak göz önüne alınabilir. Şöyle ki, eğer X işaretli kare bir sadeleştirme sağlıyorsa 1 eğer herhangi bir sadeleştirme sağlamıyorsa 0 olarak göz önüne alınır.

Örnek: Etkisiz kombinasyonları

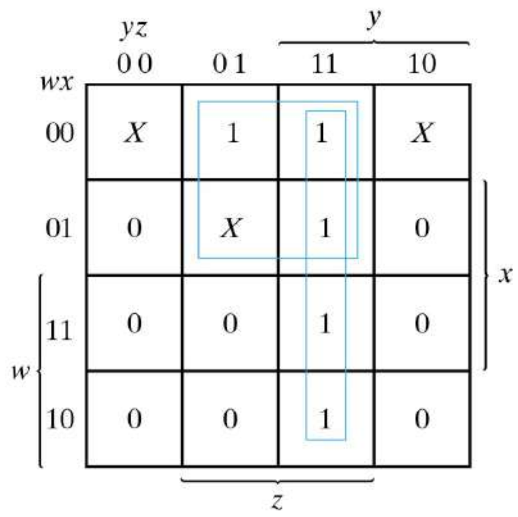
$d(w,x,y,z)=\sum(0,2,5)$

şeklinde olan

$F(w,x,y,z)=\sum(1,3,7,11,15)$ Boole fonksiyonunu sadeleştiriniz.



(a) $F = yz + w'x'$



(a) $F = yz + w'z$

İlk ifade de 0 ve 2 etkisiz elemanları 1 olarak düşünülmüştür. İkinci ifadede de ise sadece 5 etkisiz elemanı 1 olarak dikkate alınmıştır.

$F(w,x,y,z)=yz+w'z=\sum(1,3,5,7,11,15)$

Eğer toplamların çarpımı şeklinde bir ifade elde etmek istiyorsak bu durumda 0'lar göz önüne alınmalıdır.

wx \ yz	00	01	11	10
00	X	1	1	X
01	0	X	1	0
11	0	0	1	0
10	0	0	1	0

F'=z'+wy' ise F=z(w'+y)=Σ(1,3,7,11,15)

Tablo Yöntemi

Değişken sayısının beşi geçmediği durumlarda diyagram yöntemi oldukça kullanışlıdır. Fakat, diyagramda komşu kareler arasında ilişkiyi görmek ve minimum değişken ile fonksiyonu ifade etmek için standartlar yoktur. Tamamen kişinin tecrübelerine ve görüş kabiliyetine bağlıdır. Eğer değişken sayısı artarsa oldukça zor ve hataya açık durumlar söz konusu olacağı aşikardır. Bu nedenle, kullanıcı hatalarından uzak kalabilmek için tablo yöntemi kullanmak gerekecektir. Bu yöntem iki aşamadan oluşur. Önce, asal çarpanlar bulunur sonra, en az sayıda değişkeni verecek ifade seçilir.

Asal Çarpanların Belirlenmesi

Fonksiyonu belirleyen minterimlerin alınması işleme başlanır.

İzlenecek adımlar şu şekildedir.

Eşleme işlemi uygulanarak temel çarpanlar bulunur. Bu adım şu şekildedir: her minterim diğer minterimlerle karşılaştırılır. İki minterim birbirinden tek değişken ile ayrılıyorsa bu değişken kaldırılarak terimden bir değişken eksiltilir. Bu eşleme işlemi tüm minterimler için yapıldıktan sonra ikinci minterim ve diğerleri için tekrarlanır. Geriye kalan terimlerle eşleşmeyen terimler asal çarpanları oluşturur.

Örnek: Tablo yöntemi kullanarak aşağıdaki Boole fonksiyonunu basitleştiriniz.

$$F(w, x, y, z) = \sum(0, 1, 2, 8, 10, 11, 14, 15)$$

(a)

	w	x	y	z	
0	0	0	0	0	√
1	0	0	0	1	√
2	0	0	1	0	√
8	1	0	0	0	√
10	1	0	1	0	√
11	1	0	1	1	√
14	1	1	1	0	√
15	1	1	1	1	√

(b)

	w	x	y	z	
0,1	0	0	0	–	
0,2	0	0	–	0	√
0,8	–	0	0	0	√
2,10	–	0	1	0	√
8,10	1	0	–	0	√
10,11	1	0	1	–	√
10,14	1	–	1	0	√
11,15	1	–	1	1	√
14,15	1	1	1	–	√

(c)

	w	x	y	z
0,2,8,10	–	0	–	0
0,8,2,10	–	0	–	0
10,11,14,15	1	–	1	–
10,14,11,15	1	–	1	–

<u>Adım 1.</u> Önce minterimler temsil ettikleri ikili karşılıklarındaki 1'lerin sayısına göre gruplandırılırlar. Kullanılacak olan minterimler m_0, m_1, m_2, m_8, m_{11}, m_{14} ve m_{15} olduğuna göre

bunların ikili karşılıklarında kaç adet 1 varsa gruplandırılırlar. 0'da 0 tane, 1,2 ve 8'de 1 tane, 10'da iki tane, 11 ve 14'de 3 tane ve 15'de 4 tane bir vardır. Her bir grup altındaki grupla ilişkilendirilebilir. Çünkü komşu gruplar arasında sadece bir tane 1 farklıdır. Dolayısıyla, tek bir bitin yer değiştirmesi söz konusu olabilir fakat iki altındaki grupla 1'lerin sayısı farklı olduğu için herhangi bir benzerlik aramak mümkün değildir.

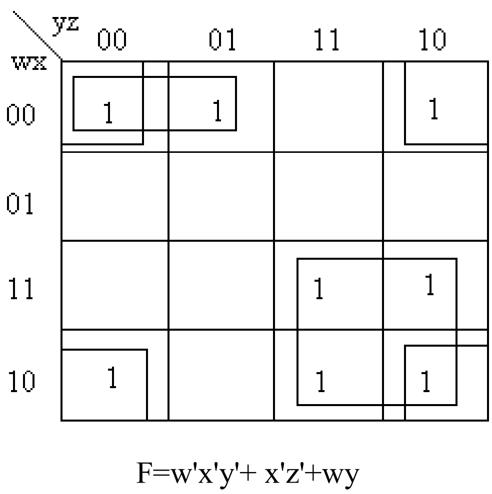
Adım 2. İki grup arasında sadece bir bit farklıysa bu iki grup eşleştirilebilir. Örneğin (a) sütunundaki 0 ile 1 arasında sadece ilk bitler farklı olduğu için bu iki 0,1 şeklinde bir grup oluştururlar ve farklı olan bit – ile gösterilir. Bir başka deyişle, değişen bit çıkartılır. O bu sütundaki diğer iki eleman olan 2 ve 8 ile de karşılaştırılır ve tek bit değişiyorsa bunlarda bir grup oluştururlar. Oluşan gruplar 0,1 0,2 ve 0,8 gruplarıdır. Daha sonra 1,2 ve 8 grubu 10 grubu ile karşılaştırılır. Oluşan gruplar 2,10 ve 8,10 gruplarıdır. Bu işleme tüm gruplar arasında devam edilerek ikinci sütun oluşturulur.

Adım 3. (b) sütunundaki elemanların artık üç değişkeni kalmıştır. Değeri 1 ise elemanın kendisi değeri 0 ise elemanın üslü değerinin alınacağı anlamına gelir. Burada gruplar arasında Adım 2'deki gibi bir eşleme işlemi uygulanır. 0,1 elemanı herhangi bir grupla eşleşmediği için yanında (√) işareti konmamıştır. Herhangi bir grupla eşleşmeyen grubun yanına (√) konmaz. Bu da bu grubun daha fazla sadeleştirilemeyeceği anlamına gelir. Eşleşmeyen gruplar bu şekilde belirlenir. Burada incelenen örnek üçüncü sütunda sonlanmıştır. İşleme eşleşmeyen grup kalmayıncaya kadar devam edilir.

Adım 4. Tabloda yanında (√) işareti olmayan terimler asal çarpanları oluşturur. Bu örnekte, (b) sütunundaki 0,1 grubu, (c) sütunundaki 0,2,8,10 ve 10,11,14,15 terimleri asal çarpanlardır. 000-, -0-0 ve 1-1- gruplarıdır. Bunlar, w'x'y', x'z' ve wy terimlerine karşılık gelir. Asal çarpanların toplamı fonksiyonun basitleştirilmiş halini verir.

F= w'x'y'+x'z'+wy

Bu örneği, yöntemi doğrulamak için diyagram yöntemi ile de çözüp görelim.



F=w'x'y'+ x'z'+wy

Tablo yöntemin yapılması gereken uzun ve sıkıcı işlemler, karşılaştırmanın ikili sayılar yerine onlu sayılarla yapılması ile azaltılabilir. İkili sayılar içindeki her bir, ikinin bir kuvveti ile çarpılacak bir katsayıyı temsil eder. Önce sayılar bir önceki örnekte olduğu gibi sıralanır. Daha sonra gruplar birbiri ile karşılaştırılır. Alt gruptaki sayı üst gruptaki sayıdan ikinin kuvveti kadar büyükse bu grup (b) sütununa taşınır. Ve parantez içine de 2'nin kuvveti olan fark yazılır. İlk gruptaki 0, ikinci gruptaki 1, 2 ve 8'den ikinin kuvveti kadar küçüktür. Dolayısıyla bunlar şeklinde 0,1 (1), 0,2 (2) ve 0,8 (8) bir grup oluştururlar. Üçüncü grup 10 ise, 2 ve 8 ile 2,10 (8) ve 8,10 (2) şeklinde iki grup oluşturur. Bu şartları sağlayabilmesi için alt gruptaki sayı üst gruptaki sayıdan büyük olmalıdır. (b) sütunu oluşturulduktan sonra artık eşlemeler parantez içindeki sayılara göre yapılır. Aynı sayı grupları eşleştirilir. Eşleşen tüm terimlerin yanına da √ işareti konur. (c) sütunu da bu şekilde oluşturulur. Asal çarpanlar tabloda yanında √ işareti olmayan terimlerdir. Bu terimler onlu verildikleri için ikiliye dönüştürülmeli ve yanındaki parantez içindeki rakamlar ise hangi basamaktaki elemana – konacağını belirtir. Örneğin, 0,2,8,10 (2,8), 1010 ikili rakamındaki 2. ve 4. rakamların – olacağı anlamına gelir. Yani terim, -0-0 olarak göz önüne alınacaktır. 10,11,14,15 (1,4) grubunda ise 1111 ifadesinde 1. ve 3. rakamlar – olarak göz önüne alınacaktır. Terim 1-1- olarak kabul edilecektir.

(a)	(b)	(c)
0 √	0,1 (1)	0,2,8,10 (2,8)
	0,2 (2) √	0,2,8,10 (2,8)
1 √	0,8 (8) √	10,11,14,15 (1,4)
2 √		10,11,14,15 (1,4)
8 √	2,10 (8) √	
	8,10 (2) √	
10 √		
11 √	10,11 (1) √	
14 √	10,14 (4) √	
15 √	11,15 (4) √	
	14,15 (1) √	

Örnek: Aşağıdaki fonksiyonun asal çarpanlarını belirleyiniz.

$$F(w,x,y,z)=\sum(1,4,6,7,8,9,10,11,15)$$

(a)	(b)	(c)
0001 1 √	1,9 (8)	8,10,9,11 (1,2)
0100 4 √	4,6 (2)	
1000 8 √	8,9 (1) √	
0110 6 √	8,10 (2) √	
1001 9 √	6,7 (1)	
1010 10 √	9,11 (2) √	
0111 7 √	10,11 (1) √	
1011 11 √	7,15 (8)	
1111 15 √	11,15 (4)	

Yanında √ işareti olmayan 1,9 (8) 4,6 (2) 6,7 (1) 7,15 (8) 11,15 (4) ve 8,9,10,11 (1,2) terimleri asal çarpanlardır.

O halde;

1,9 (8)=1001=-001 =x'y'z

4,6 (2)=0110=01-0 =w'xz'

6,7 (1)=0111=011- =w'xy

7,15 (8)=1111=-111=xyz

11,15 (4)=1111=1-11=wyz

8,9,10,11 (1,2)=1011=10--=wx'

F= wy'z+w'xz'+w'xy+xyz+wyz+wx'

Bu yaklaşımla minimum değişken içeren ifade elde edilmesi garanti altına alınmaz. Bu örnek için diyagram metodunu inceleyelim.

wx \ yz	00	01	11	10
00		1		
01	1		1	1
11			1	
10	1	1	1	1

F=wx'+xyz+w'xz'+x'y'z

Görüldüğü üzere tablo yöntemi şu hali ile minimum fonksiyonu vermez. O halde minimumu veren başka bir tablo yöntemi kullanmak gerekecektir.

Asal Çarpanların Seçimi

Minimum fonksiyonu veren asal çarpanların seçimi asal çarpanlar tablosundan yapılır. Önce elde edilen terimler bir sütuna yazılır. Yanındaki sütuna bu terimin hangi minterimlerle ilişkili olduğu yazılır. Diğer sütunlara ise tüm minterimler yazılır. Her terim hangi minterimi ihtiva ediyorsa onun bulunduğu satır ve sütuna X işareti konur. Sütununda tek X olan minterimleri bünyesinde bulunduran terim temel asal çarpandır. Öncelikle bu temel asal çarpanlar bulunur ve yanlarına √ işareti konur. Sonra bu asal çarpanların içerindeki diğer minterimler diğer tüm terimlerde aranır ve bulunanların en alt sütununa √ işareti konur. En alt sütununda √ işareti olmayan minterimleri bulunduran terim ise asal çarpan olarak fonksiyona katılır. Bunu bir örnekle açıklayalım.

Örnek: Aşağıdaki fonksiyonu tablo yöntemi kullanarak sadeleştiriniz

$$F(w, x, y, z) = \sum(1, 4, 6, 7, 8, 9, 10, 11, 15)$$

F= wy'z+w'xz'+w'xy+xyz+wyz+wx' şeklinde asal çarpanları daha önce bulunmuştu.

			1	4	6	7	8	9	10	11	15
√	x'y'z	1,9	X					X			
√	w'xz'	4,6		X	X						
	w'xy	6,7			X	X					
√	xyz	7,15				X					X
	wyz	11,15								X	X
√	wx'	8,9,10,11					X	X	X	X	
			√	√	√		√		√	√	

Tek sütununda çarpı olan elemanlar 1, 4, 8 ve 10'dur. Bu minterimlerin en az birini içeren terimler temel asal çarpanlar olup x'y'z, w'xz ve wx' şeklinde elde edilirler. Daha sonra bu temel asal çarpanlardaki diğer minterimlere bakılır. Örneğin, x'y'z terimindeki diğer minterim 9'dur onun en alt sütununa da √ işareti konur. w'xz' temel asal çarpanındaki diğer minterim 6 dır 6 sütunun altına da √ işareti konur. wx' temel asal çarpanındaki minterimler 10 ve 11 'dir onların da altına √ işareti konur. Altına işareti olmayan minterimleri içeren (7 ve 15- xyz) terim de asal çarpan olarak fonksiyona katılarak minimum değişkenli fonksiyon elde edilir.

F=wx'+xyz+w'xz'+x'y'z

Toplamların şeklindeki bir ifadeyi elde edebilmek için sadece 0'lar göz önüne alınır. Fonksiyonun tümleyeni bulunur. Daha sonra bu ifadenin tümleyeni alınarak fonksiyon toplamların çarpımı şeklinde elde edilir. Etkisiz koşullar ise fonksiyonun minterimleri şeklinde göz önüne alınarak minimum değişkenli fonksiyon elde edilebilir.

Etkisiz Koşulların Değerlendirilmesi

Etkisiz koşullar başlangıçta birer minterim gibi gözönüne alınır. Gerekli sadeleştirme işlemleri yapılarak asal çarpanların seçimine geçilir. Asal çarpanlar seçilirken minterim sütununa etkisiz koşullar geçirilmez ve sadece etkisiz koşullarla ifade edilmiş bir terim buraya alınmaz.

Örnek: F(A, B, C, D)=∑(0, 1, 2, 8, 12) ve d(A, B, C, D)=∑(3, 7, 10, 14) şeklinde verilmiş bir Boole fonksiyonunun en sade halini Tablo yöntemini (Quine-McCluskey) kullanarak bulunuz.

ABCD	ABCD	ABCD
0 √	0,1 (1) √	0,1,2,3 (1,2)
	0,2 (2) √	0,2,8,10 (2,8)
	0,8 (8) √	
1 √	1,3 (2) √	8,10,12,14 (2,4)
2 √	2,3 (1) √	
8 √	2,10 (8) √	
	8,10 (2) √	
	8,12 (4) √	
3 √	3,7 (4)	
10 √	10,14 (4) √	
12 √	12,14 (2) √	
7 √		
14 √		

	0	1	2	8	12
A'B' 0,1,2 √	X	X	X		
B'D' 0,2,8	X		X	X	
AD' 8, 12 √				X	X
	√	√	√	√	√

Örnek:F(w, x, y, z)=∑(0, 1, 3, 5, 13, 15) ve d(w, x, y, z)=∑(2, 6, 10, 11, 12) şeklinde verilmiş bir Boole fonksiyonunun en sade halini Tablo yöntemini (Quine-McCluskey) kullanarak bulunuz. Gerekli işlemlerden sonra bulunan terimler aşağıdaki gibi olur.

		0	1	3	5	13	15
(√)	w'y'z 1, 5		X		X		
	wxy' 13					X	
(√)	xy'z 5,13				X	X	
	wyz 15						X
√	wxz 13,15					X	X
√	w'x' 0,1,3	X	X	X			
	x'z 3			X			
		√	√	√			

Parantez içindeki ifadelerden sadece bir tanesi alınacak

Sütunuda tek X olan 0'dır. İçerdiği terimler 1 ve 3'tür. Bu terimlerin sütununa da √ işareti konur. İşaret olmayan sütunlar 5, 13, ve 15'tir. Sadece bu üç terim tarafından aynı anda kapsanan terimlere bakılır. 13 ve 15 terimlerini kapsayan wxz 13,15 terimini alırsak açıkta kalan 4 terim mevcuttur. Bu 4 terimden wyz 15 ve wxy' 13 terimleri zaten 13 ve 15 tarafından kapsanmaktadır. 5 ile ilgili olan terimlere bakılacak olursa w'y'z 1, 5 terimi alınırsa açıkta kalan hiç terim kalmıyor. Aynı şekilde xy'z 5,13 terimi alındığında da açıkta kalan terim kalmıyor. Dolayısıyla istenilen iki terimden birisi alınabilir.

Örnek: *(Petrick algoritmasının uygulanışı)* F(w, x, y, z)=∑(1, 5, 7, 8, 10, 14) ve d(w, x, y, z)=∑(0, 6, 9, 11, 13, 15) şeklinde verilmiş bir Boole fonksiyonunun en sade halini Tablo yöntemini (Quine-McCluskey) kullanarak bulunuz.

Gerekli işlemlerden sonra bulunan terimler aşağıdaki gibi olur.

		1	5	7	8	10	14
A	1, 8 (1,8)	X			X		
B	1, 5 (4,8)	X	X				
C	8, 10(1,2)				X	X	
D	5, 7 (2,8)		X	X			
E	7, 14 (1,8)			X			X
F	10, 14 (1,4)					X	X

Görüldüğü üzere her terimde iki tane (X) var ve asal çarpanların seçimi mümkün görülmemektedir. Bu durumda *Petrick algoritması* uygulanarak minimum değişken ile örtülen en sade fonksiyonun bulunması gerekecektir. İlk sutunda yazan harfler (A, B, C,...) o değişkenin hangi minterimler ile örtüldüğünü göstermektedir. Algoritmaya göre yazılacak ifade şu şekilde olmalıdır:

1- Örneğin; 1 nolu minterim A+B ile örtülmektedir. 5 nolu minterim B+D ile örtülmektedir. Bu şekilde tüm terimler bulunur.
2- Minterimleri örten harfler toplamların çarpımı formunda elde edilir. Yani (A+B)(B+D)(D+E)(A+C)(C+F)(E+F) şeklinde bulunur.
3- Toplamların çarpımı formundaki bu ifade düzenlenerek çarpımların toplamı formuna getirilir. BCE+ABEF+BCDF+ACDE+ADF
4- Bu terimler içerisinde en az terimle ifade edilen BCE ve ADF terimleridir. Bu terimler alınır.
5- BCE terimi 1,5 (4,8), 8,10 (1,2) ve 7, 14 (1,8) ile ifade edilen terimlerin alınacağını gösterir. Bu durumda F=y'z+wx'+xy olarak basitleştirilmiş ifade elde edilir.
6- ADF terimi 1,8 (1,8), 5,7 (2,8) ve 10,14 (1,4) ile ifade edilen terimlerin alınacağını gösterir. Bu durumda F=x'y'+xz+wy olarak basitleştirilmiş ifade elde edilir.
7- Bu iki basitleştirilmiş ifadeden istenilen kullanılabilir.

Kombinezonal Lojik

Sayısal sistemlerdeki lojik devreler, kombinezonal ya da ardışıl olabilir. Kombinezonal bir devrede çıkışlar, o andaki girişlere bağlı lojik kapılardan oluşur. Kombinezonal devre, bir Boole fonksiyonu kümesi ile belirlenmiş mantıksal bir bilgi işleme operasyonunu gerçekleştirir. Ardışıl devrelerde ise, bu lojik kapılara ek olarak bellek elemanları da kullanılır. Çıkışlar, giriş ve bu bellek elemanlarının durumları ile belirlenir. Bellek elemanların çıkışları bir önceki durumla ilişkili olabilir. Bu nedenle, ardışıl devrelerde çıkışlar o an ki durum için değil daha önceki durumlarla da ifade edilebilir.

Bu bölümün amacı, daha önce anlatılanlar ışığında kombinezonal devrelere ilişkin sistematik tasarım ve analiz yöntemleri geliştirmektir.

Kombinezonal bir devre, giriş değişkenleri, lojik kapılar ve çıkış değişkenlerinden oluşur. Lojik kapılar girişlerden işaretleri alıp çıkışlar için işaret üretirler. Bu işlem belirli bir giriş verisinden istenen çıkış verisine ikili bilgi dönüştürmedir. Kombinezonal bir devre ait blok diyagramı aşağıda görülmektedir.

n giriş söz konusu olduğuna göre 2^n tane olası ikili değer kombinasyonu vardır. Her bir giriş kombinasyonu için tek bir çıkış kombinasyonu mevcuttur. Bu girişlerde değişkenlerin kendileri ya da tümleyenlerinin olduğu varsayılacaktır. Eğer girişte hem değişkenin kendisi hem de tümleyeni mevcut ise evirici kullanmadan daha sonra anlatılacak olan flip-flop devreleri kullanılacaktır.

Tasarım Yöntemi

Kombinezonal devrenin tasarımı önce sözle başlar ve lojik diyagramlardan elde edilecek lojik devre diyagramı veya Boole fonksiyonu elde edilmesi ile sonuçlanır. Yöntem aşağıdaki adımlardan oluşur.

1. Problem sözle ifade edilir.

2. Mevcut giriş ve çıkış değişkenlerinin sayısı belirlenir.
3. Giriş ve çıkış değişkenlerine harf sembolleri atanır.
4. Giriş ve çıkış arasındaki ilişkileri tanımlayan doğruluk tablosu oluşturulur.
5. Her lojik çıkış için basitleştirilmiş Boole fonksiyonu elde edilir.
6. Lojik devre çizilir.

Kombinezonal bir devrenin doğruluk tablosu giriş ve çıkış sütunlarından oluşturulur. Giriş sütunlarındaki 1 ve 0'lar n adet giriş değişkeni için mevcut 2^n ikili kombinasyonundan elde edilir. Çıkışlara ilişkin değerler problemin incelenmesi ile elde edilir. Her geçerli giriş değeri için çıkış 1 ya da 0 olabilir. Eğer girişte belirsiz durumlar varsa ya da girişin değerleri için çıkışın durumları belirsiz ise bu gibi durumlar etkisiz koşullar olarak göz önünde tutulurlar.

Doğruluk tablosundan elde edilen Boole fonksiyonu çıkışları cebirsel işlem, diyagram ya da tablo yöntemi ile basitleştirilir. Pratik uygulamaya uygun bir tasarım yönteminde (1) minimum kapı sayısı, (2) kapıların minimum giriş sayısı, (3) devre boyunca işaretin minimuma yayılma süresi, (4) minimum ara bağlantı sayısı ve (5) her bir kapının sürme yeteneğine ilişkin sınırlamalar şeklinde başlar. Genellikle, Boole fonksiyonunu basitleştirme işlemi ile başlanır ve diğer performans kriterlerinin sağlanması için sistem geliştirilir.

Toplayıcılar

Sayısal bilgisayarlar, bilgi işleme görevi sırasında çeşitli aritmetik işlemleri yerine getirmek durumunda kalırlar. Bu aritmetik işlemlerin en temeli olan iki sayının toplanmasıdır. Bu basit toplama, 0+0=0, 0+1=1, 1+0=1 ve 1+1=10 şeklinde dört olası işlemdir. Burada, ilk üç toplam tek haneli bir ikili toplamı verir ama dördüncü toplam iki haneden oluşmuştur. İkinci hane yüksek anlamlı bit olup bir sonraki toplama elde olarak geçecektir. İki bitin toplamını gerçekleştiren kombinezonal devreye yarı toplayıcı denir. Üç bitin toplamını (iki bit toplam bir bit gelen elde) veren kombinezonal devreye ise tam toplayıcı denir.

Yarı Toplayıcı

Girişler x ve y, toplam S ve oluşan elde ise C ile gösterilsin. Doğruluk tablosu aşağıdaki gibi elde edilir.

x	y	C	S
0	0	0	0
0	1	0	1

1	0	0	1
1	1	1	0

Toplam sonucuna ilişkin diyagram,

x \ y	0	1
0		1
1	1	

elde bitinin sonucuna ilişkin diyagram

x \ y	0	1
0		
1		1

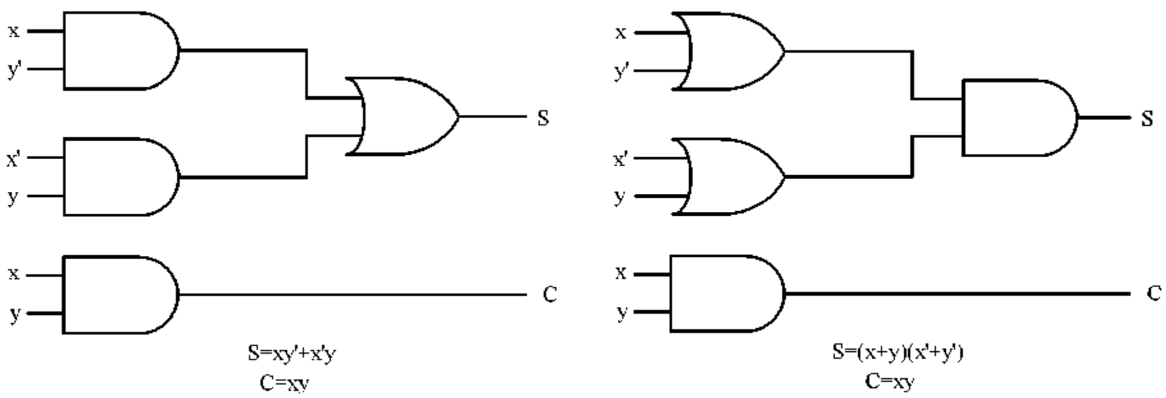
şeklinde elde edilir.

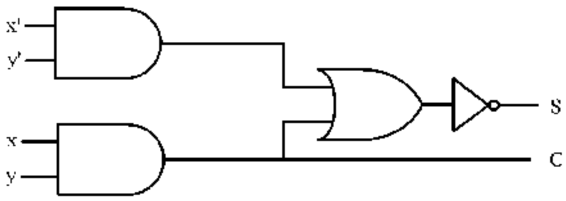
Toplama ilişkin Boole fonksiyonu

S=x'y+xy'

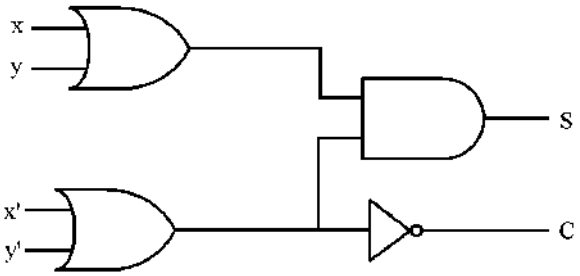
elde bitine ilişkin Boole fonksiyonu

C=xy şeklinde elde edilir.

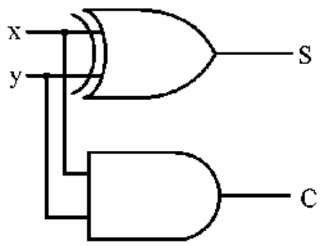




$S=(C+x'y')'$
$C=xy$



$S=(x+y)(x'+y')$
$C=(x'+y')'$



$S=x\oplus y$
$C=xy$

İlk şekil çarpımların toplamı şeklinde elde edilmiştir. İkincisi ise, diyagramda 0'lar göz önüne alınarak toplamların çarpımı şeklinde elde edilmiştir. Üçüncüsü ise toplamların çarpımı şeklinde elde edilen ifadenin tümleyeni alınarak elde edilmiş ve S tekrar tümleyeni alınarak fonksiyonun kendisi bulunmuştur. Dördüncü şekilde ise elde bitinin tümleyeni bulunmuş ve sonra tekrar tümleyeni alınmıştır. Yarı toplayıcı aslında bir ÖZEL-VEYA fonksiyonu ile tanımlanır. Elde bitinin ise bir VE kapısı ile elde edildiği şekil beşincide görülmektedir.

Tam Toplayıcı

Tam toplayıcı, üç giriş bitinin aritmetik toplamını oluşturan bir kombinezonal devredir. Üç giriş ve iki çıkıştan oluşur. x ve y ile gösterilenler giriş, z ile gösterilen düşük anlamlı hanelerden gelen elde biti, C toplam sonucu oluşan elde ve S ise toplam sonucudur. Tam toplayıcıya ait doğruluk tablosu aşağıda gösterildiği gibidir.

x	y	z	C	S
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

Biri elde bitine diğeri de toplam sonucuna ilişkin iki tane diyagram oluşturulmalıdır.

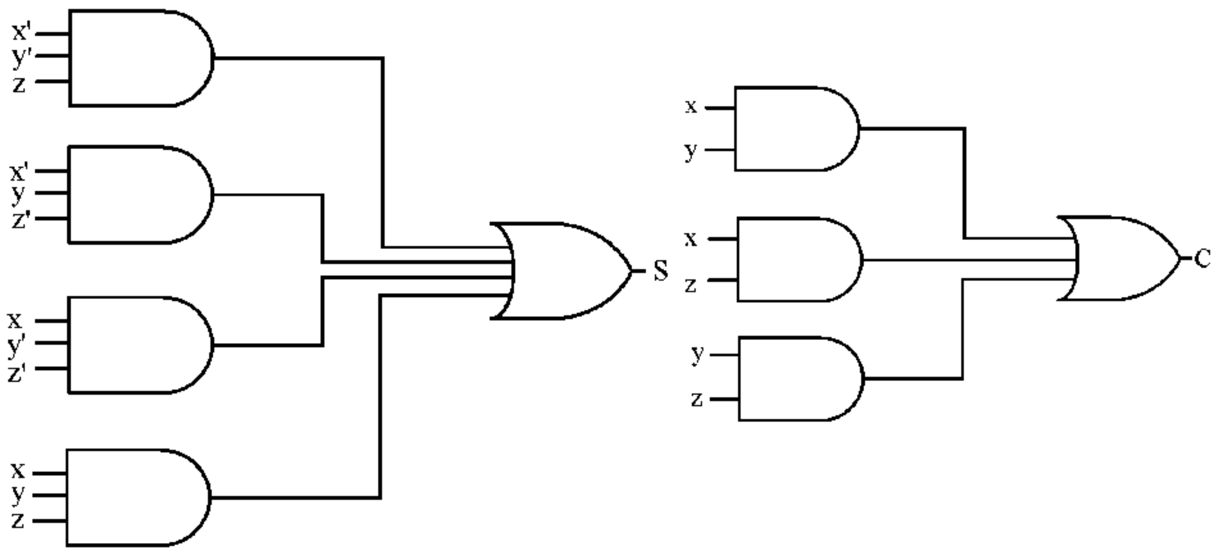
x \ yz	00	01	11	10
0		1		1
1	1		1	

S=xy'z'+x'y'z+xyz+x'yz'

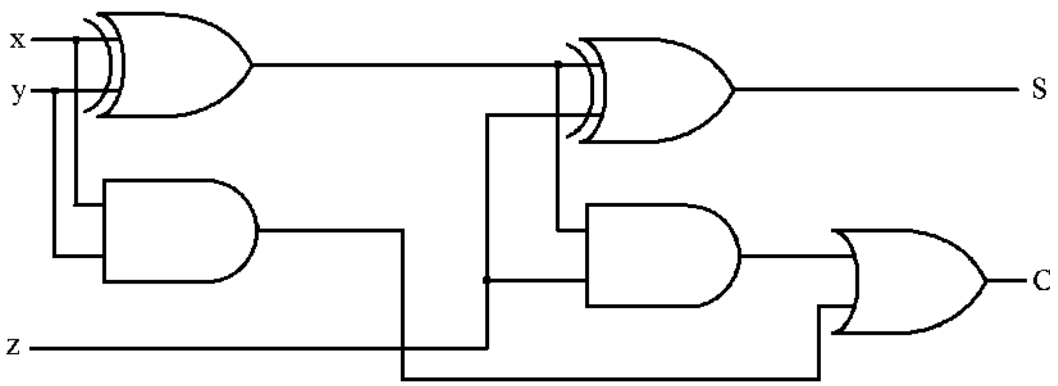
x \ yz	00	01	11	10
0			1	
1		1	1	1

C=xz+yz+xy

Şeklinde diyagramlardan elde edilirler. VE, VEYA kapıları ile aşağıdaki gibi bir lojik kapı devresi oluşturulabilir.



Bir tam toplayıcı iki yarı toplayıcı ile oluşturulabilir.



S=z⊕(x⊕y) a=x⊕y=x'y+xy' olsun

=z⊕a=z'a+za'

=z'(x'y+xy')+z(x'y+xy')' (x'y+xy')'=(x'y)'.(xy')'=(x+y')(x'+y)=xy+x'y'

$=x'yz'+xy'z'+xyz+x'y'z$

$C=z(x\oplus y)+xy$

$=z(x'y+xy')+xy$

$=x'yz+xy'z+xy$

ÇIKARICILAR

Daha önce anlatıldığı gibi, iki ikili sayının çıkarma işlemi, çıkan sayının tümleyeninin alınıp bunun çıkarılan sayıya eklenmesi ile alınır. Bu şekilde çıkarma işlemi tam toplayıcıları gerektiren bir toplama işlemine dönüşmüş olur. Çıkarma işleminde, çıkan sayı çıkarılan sayıdan büyük ise bir üst basamaktan bir bit ödünç alınır ve işlem bir üst bite geçilerek devam edilir.

Yarı Çıkarıcı

İki biti birbirinden çıkaran bir kombinezonal devredir. Ayrıca ödünç bit alınıp alınmadığını gösteren bir de çıkış mevcuttur. Ödünç alınan bit, bir üst basamaktan alındığı için 1 değil 10 olarak düşünülecektir. x ve y çıkan ve çıkarılan sayılar, B ödünç(borrow) biti, D ise fark bitini göstermek üzere aşağıdaki doğruluk tablosu elde edilebilir.

x	y	B	D
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

Diyagram metodu ile aşağıdaki Boole fonksiyonları elde edilir.

$D=x'y+xy'$ $\qquad B=x'y$

Dikkat edilirse buradaki D ‘ye ait ifade yarı toplayıcıdaki ifade ile aynıdır.

Tam Çıkarıcı

Tam çıkarıcı, yarı çıkarıcıya ek olarak bir önceki basmağa verilen ödünç biti de göz önüne alan kombinezonal devredir. Örneğin, x=0, y=0 ve z=1 olsun, x-y-z işlemi için üst basamaktan bir ödünç alınmalıdır. 10-0-0-1=1 olur. Bu durumda B=1 ve D=1 olmuş olur. Doğruluk tablosu aşağıdaki gibidir.

x	y	z	B	D
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

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

D=xy'z'+x'y'z+xyz+x'yz'

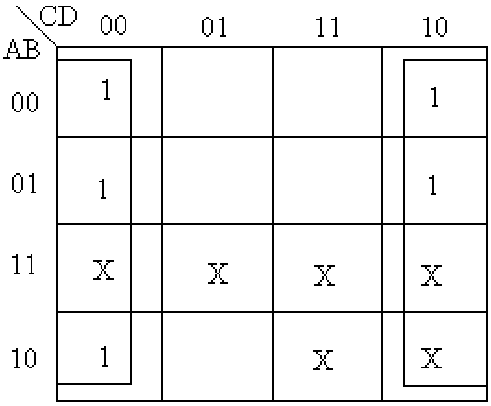
x \ yz	00	01	11	10
0		1	1	1
1			1	

B=x'z+yz+x'y

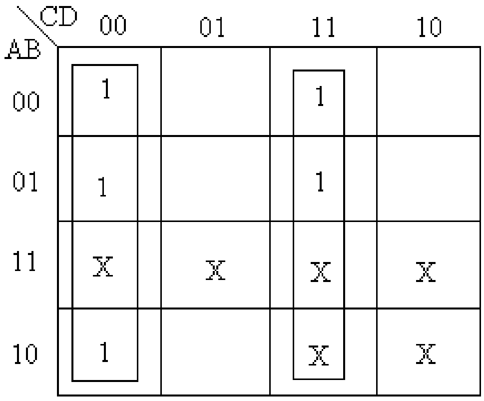
Kod Dönüştürme

Farklı sayısal sistemlerde farklı kodlar kullanıldığı için bu kodların bir birilerine dönüştürülmesi gerekebilir. Burada, bu dönüşüm BCD kodunun 3-fazlalık koduna dönüştürülmesi şeklinde bir örnek ile gösterilecektir. A, B, C, D BCD girişleri, w, x, y, z ise üç fazlalık kod çıkışları olmak üzere aşağıdaki dönüşüm tablosu elde edilir.

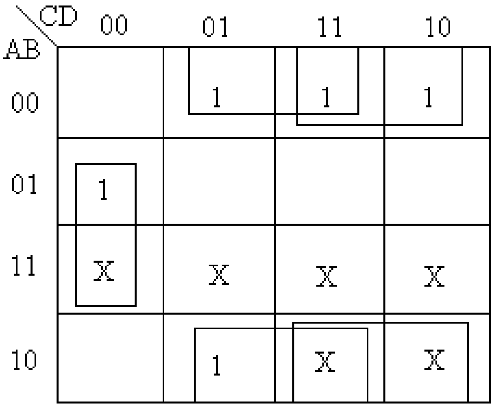
Giriş BCD				Çıkış 3-fazlalık kodu			
A	B	C	D	w	x	y	z
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0



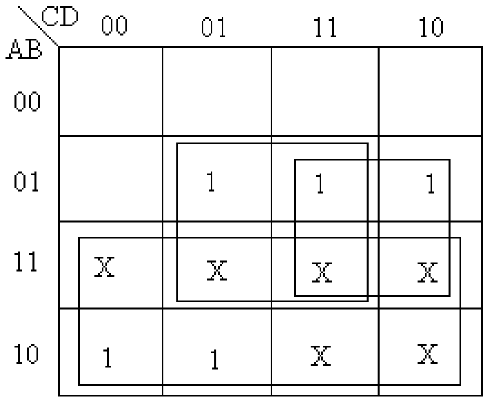
z=D'



y=CD+C'D'



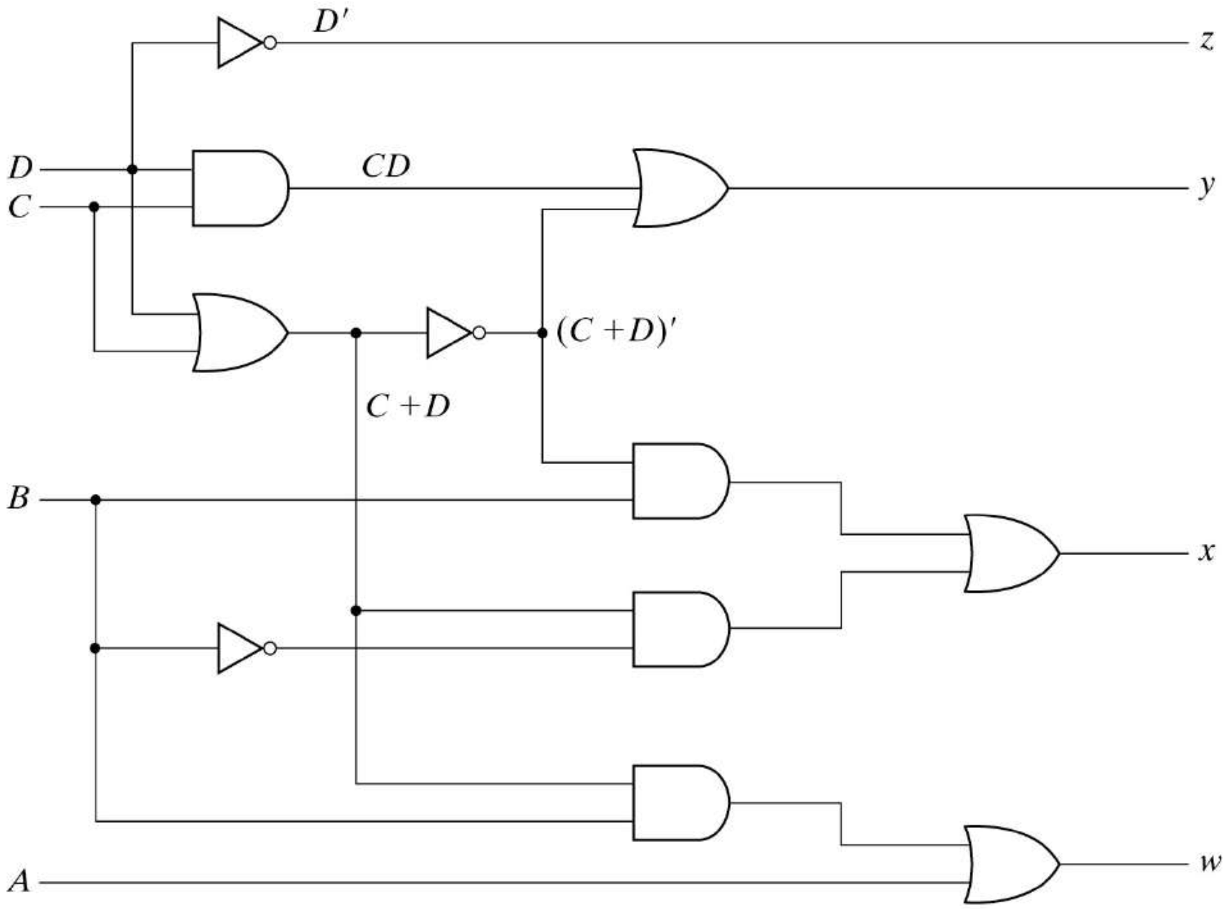
x=B'C+B'D+BC'D'



w=A+BD+BC

z=D'
y=CD+C'D'=CD+(C+D)'
x=B'C+B'D+BC'D'=B'(C+D)+B(C+D)'
w=A+BD+BC=A+B(C+D)

Tabloya dikkat edilirse sadece girişin on değeri için çıkış tanımlıdır. Diyagram oluşturulurken girişin tanımlı olmadığı çıkışlar keyfi olarak seçilecektir. Her bir çıkış için doğruluk tablosundan oluşturulan diyagramlar yukarıdaki gibidir.



ANALİZ YÖNTEMİ

Bir kombinezonal devrenin tasarımı, istenen fonksiyonun sözel tanımı ile başlar, bir Boole fonksiyon kümesi ya da bir lojik devre ile sonlanır. Bir kombinezonal devrenin analizi ise bunun tersi bir süreçle yapılır. Verilen lojik devre ile başlar bir Boole fonksiyon kümesi, doğruluk tablosu veya devrenin çalışmasına ait sözel bir ifade ile sonlanır.

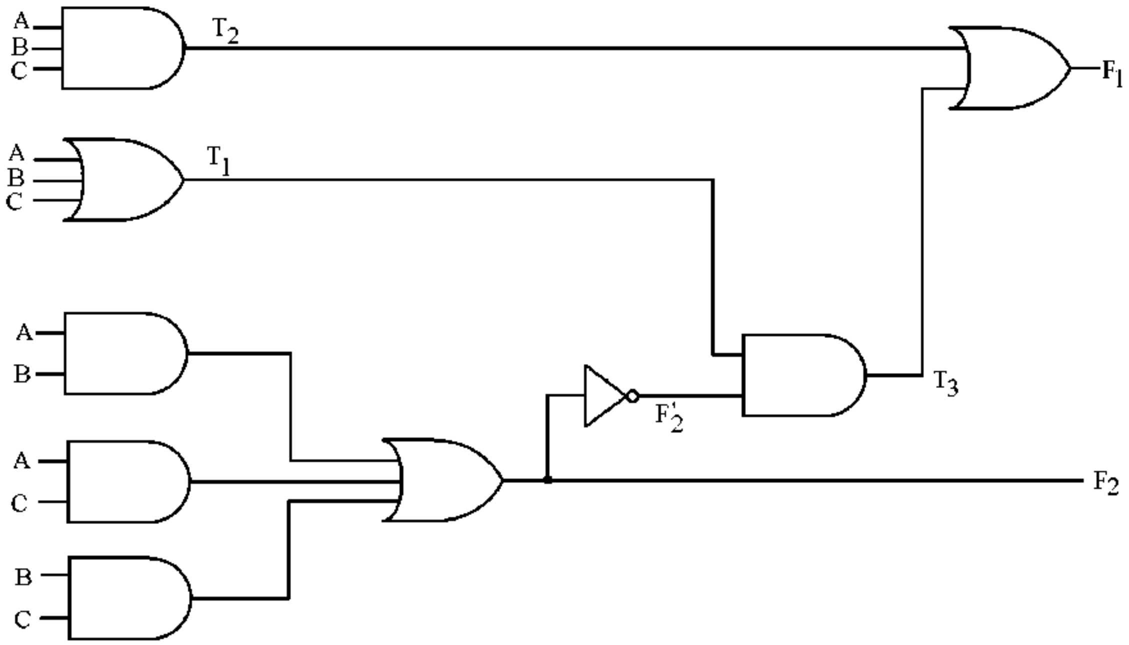
Analizin ilk adımı, verilen devrenin kombinezonal bir devre olup ardışıl bir devre olmadığının emin olunması ile başlar. Kombinezonal devrelerin lojik kapılarında geri besleme ya da bellek elemanları bulunmamaktadır. Daha sonra eğer devrenin çalışması sözle ifade edilmişse kapıların durumlarına göre çalışması doğrulanabilir. Verilmemişse aşağıdaki adımlar uygulanmalıdır.

1- Giriş değişkenlerinin fonksiyonu olan tüm kapı çıkışlarına keyfi semboller verilerek Boole fonksiyonları yazılır.
2- Giriş değişkenlerine keyfi semboller verilerek bu kapılar için Boole fonksiyonları bulunur.

3- 2. adımda anlatılan işlem tüm çıkışlar için tekrarlanır.

4- Daha önce tanımlanmış Boole fonksiyonları tekrar yerlerine konularak çıkışlar giriş değişkenleri cinsinden elde edilir.

Bunu bir örnekle açıklamaya çalışalım.



Çıkışlar F_1 ve F_2 olarak isimlendirilmiştir. Ara durumlar ise T_1, T_2 ve T_3 olarak belirlenmiştir.

F_2=AB+AC+BC

T_1=(A+B+C)

T_2=ABC

$T_3=F'_2T_1$=(A'+B')(A'+C')(B'+C')(A+B+C)

$F_1=T_3+T_2$=(A'+B')(A'+C')(B'+C')(A+B+C)+ABC

=AB'C'+A'BC'+A'B'C+ABC

Bu devre bir tam toplayıcı devredir.

Devrenin çıkış fonksiyonunu elde etmek için başka bir yöntem daha vardır. Önce girişler ve çıkışlar keyfi olarak isimlendirilirler. Sonra girişlere 0'dan 2^n-1'e kadar değerler verilerek doğruluk tabloları hazırlanır ve bu doğruluk tablolarından diyagram metoduna geçilerek Boole fonksiyonları elde edilir.

A	B	C	F_2	F'_2	T_1	T_2	T_3	F_1
0	0	0	0	1	0	0	0	0
0	0	1	0	1	1	0	1	1
0	1	0	0	1	1	0	1	1
0	1	1	1	0	1	0	0	0
1	0	0	0	1	1	0	1	1
1	0	1	1	0	1	0	0	0
1	1	0	1	0	1	0	0	0
1	1	1	1	0	1	1	0	1

Buradan aşağıdaki diyagramlar ve Boole fonksiyonları elde edilir.

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

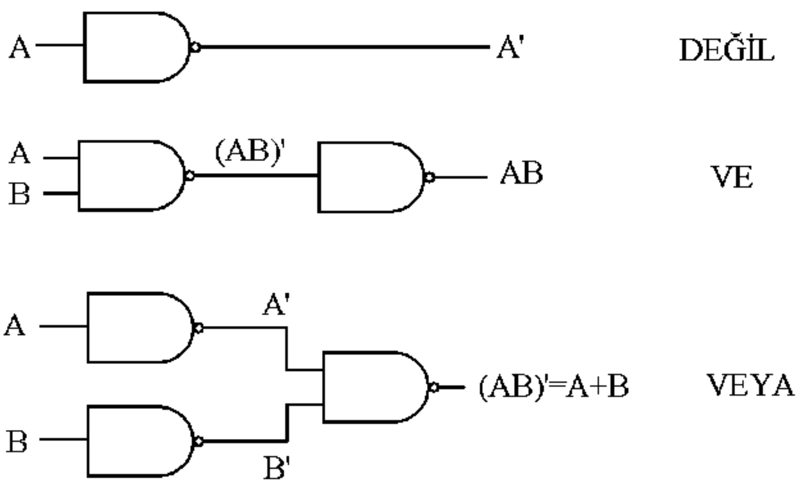
F_1=AB'C'+A'BC'+A'B'C+ABC

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

F_2=AB+AC+BC

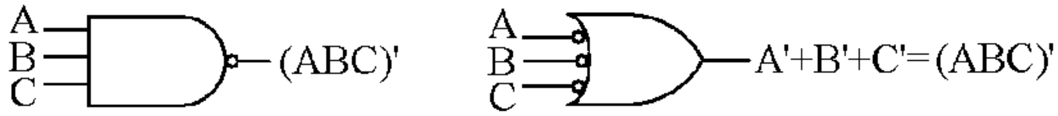
Çok Kademeli VEDEĞİL Devreleri

Daha önce de bahsi geçtiği gibi lojik devreler VE, VEYA kapıları yerine VEDEĞİL ve VEYADEĞİL kapıları ile gerçekleştirilir.



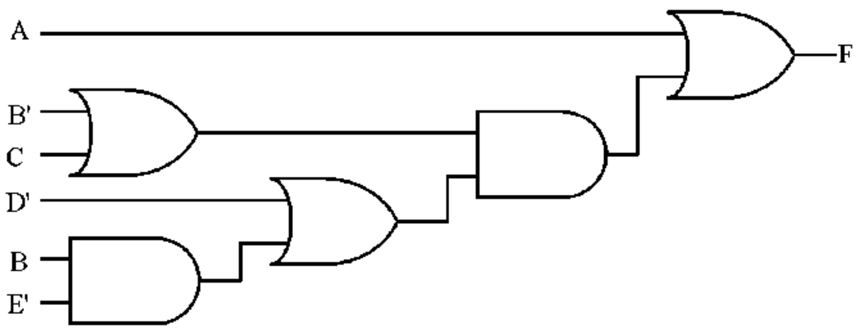
Boole Fonksiyonlarının Gerçeklenmesi

Öncelikle VE, VEYA ve DEĞİL kapıları ile gerçeklenmiş devrede bazı basit teknikler kullanılarak devre VEDEĞİL kapıları ile gerçeklenebilir. Gerçeklenirken, aşağıda gösterilen iki kapı elemanı kullanılır.

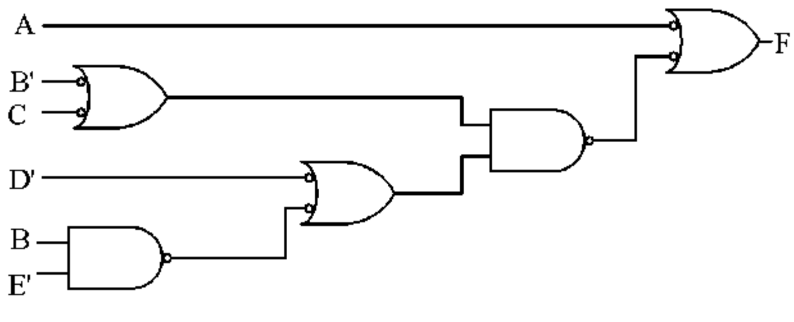


Öncelikle tüm VE kapıları VEDEĞİL kapısı ile değiştirilir. Daha sonra tüm VEYA kapıları EVİRME-VEYA kapısı ile değiştirilerek küçük daireler incelenir. Direkt olarak girişin herhangi bir değişkeni küçük daireye geliyorsa bu değişkenlerin DEĞİL'leri alınarak diğer EVİRME-VEYA kapıları VEDEĞİL kapıları ile yer değiştirilir.

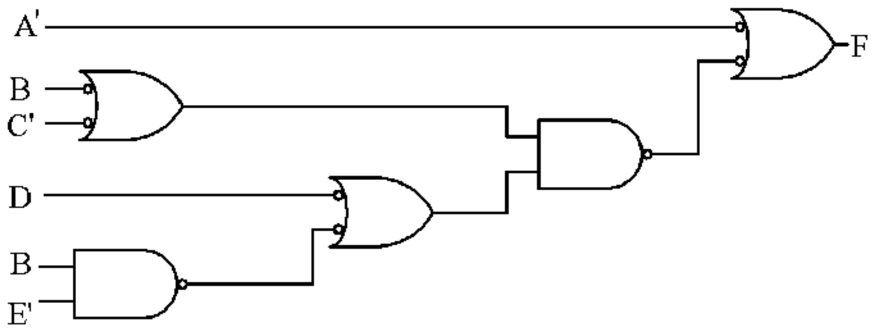
F=A+(B'+C)(D'+BE') fonksiyonunu göz önüne alalım.



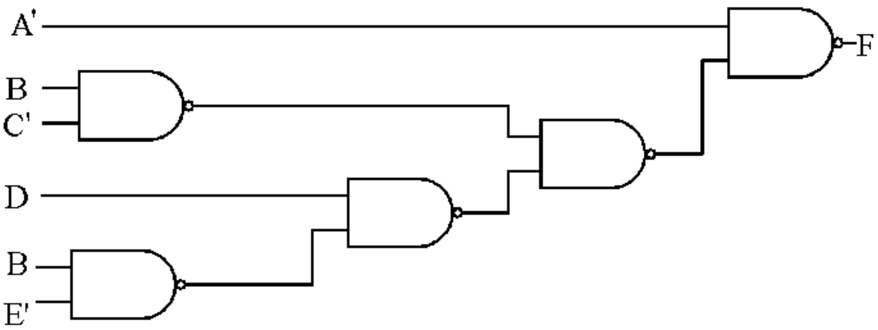
Tüm VE kapıları VEDEĞİL, VEYA kapıları EVİRME-VEYA ile yer değiştirir.



Daha sonra yuvarlaklar takip edilerek doğrudan girişe gelen yuvarlakların etkisi girişin DEĞİL'i alınarak giderilir.



Son olarak da EVİRME-VEYA kapıları VEDEĞİL kapıları ile değiştirilir.

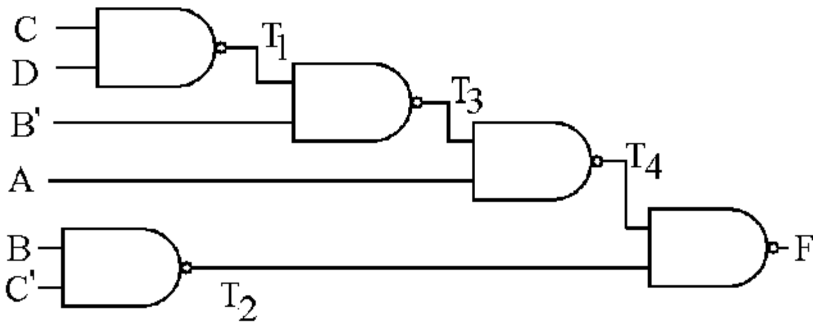


Fonksiyonun kendisinin mi? Tümleyeninin mi? Çıkışta oluştuğunu anlamak için yuvarlaklara bakılır. Eğer bir hat üzerinde takip edilen yuvarlakların sayısı değişkenin fonksiyondaki değerini veriyorsa çıkışın tümlenmesi gerekli değildir. Aksi halde çıkış tek bir girişli VEDEĞİL kapısı ile sonlandırılmalıdır. Örneğin, üsteki B giriş değişkeninin çıkışa doğru hattı üzerinde üç tane tümleme işlemi söz konusudur. Dolayısıyla bu eleman fonksiyonda B' ile temsil edilecektir ve çıkışın tümlenmesine gerek yoktur.

Soru:F=(CD+E)(A+B') Boole fonksiyonunu VEDEĞİL kapıları ile gerçekleştiriniz.

Analiz Yöntemi

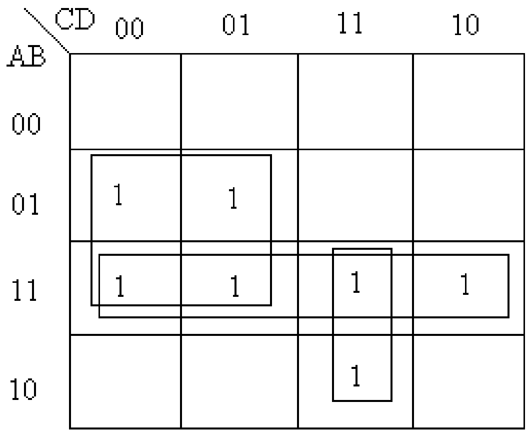
Analizde VEDEĞİL kapıları ile gerçekleştirilmiş olan bir lojik devrenin VE-VEYA lojik kapıları ile nasıl gerçekleştirildiği, Boole fonksiyonu ve doğruluk tablosunun elde edilmesi gerçekleştirilecektir. Öncelikle tüm kapı girişlerine, çıkışlarına keyfi değişken isimleri verilir. Tüm kapı çıkışları için Boole fonksiyonları elde edilir. Ve çıkış bulunur. Aşağıdaki örneği inceleyelim.



$T_1=(CD)'=C'+D'$
$T_2=(BC')'=B'+C$
$T_3=(T_1B')'=T'_1+B=CD+B$
$T_4=(T_3A)'=T'_3+A'=(C'+D')B'+A'$
$F=(T_2T_4)'=T'_2+T'_4=BC'+A(CD+B)$

Doğruluk tablosu elde edilirken tüm kapı çıkışları girişin alacağı tüm kombinasyonlara göre hesaplanır.

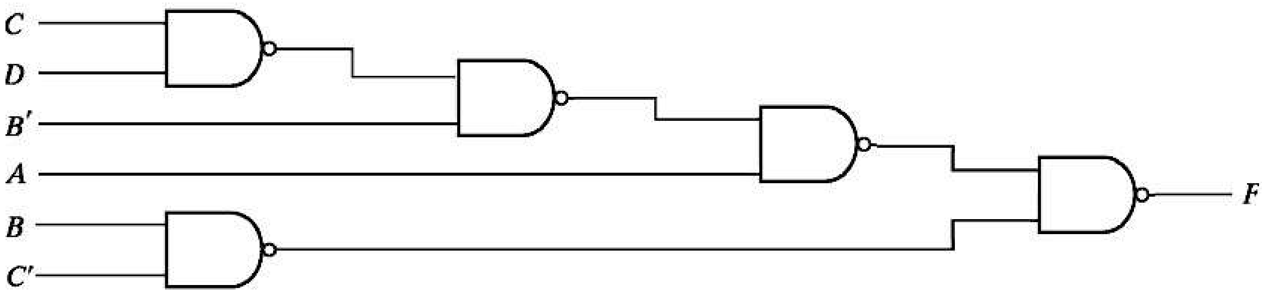
A	B	C	D	T_1	T_2	T_3	T_4	F
0	0	0	0	1	1	0	1	0
0	0	0	1	1	1	0	1	0
0	0	1	0	1	1	0	1	0
0	0	1	1	0	1	1	1	0
0	1	0	0	1	0	1	1	1
0	1	0	1	1	0	1	1	1
0	1	1	0	1	1	1	1	0
0	1	1	1	0	1	1	1	0
1	0	0	0	1	1	0	1	0
1	0	0	1	1	1	0	1	0
1	0	1	0	1	1	0	1	0
1	0	1	1	0	1	1	0	1
1	1	0	0	1	0	1	0	1
1	1	0	1	1	0	1	0	1
1	1	1	0	1	1	1	0	1
1	1	1	1	0	1	1	0	1



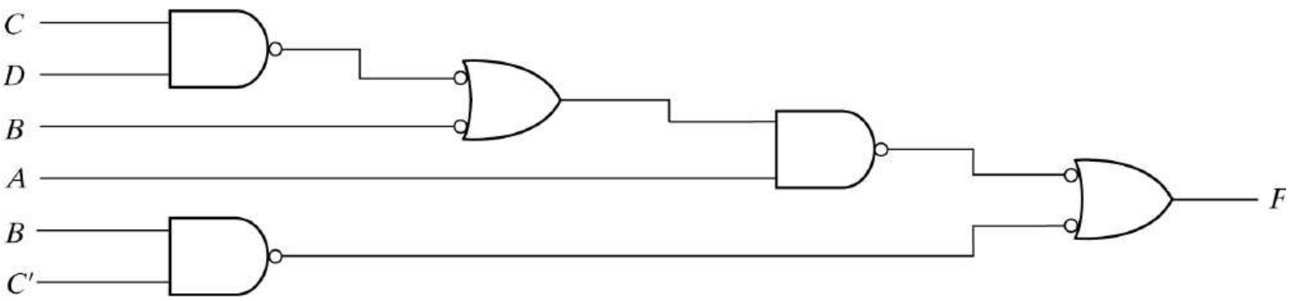
F=BC'+AB+ACD

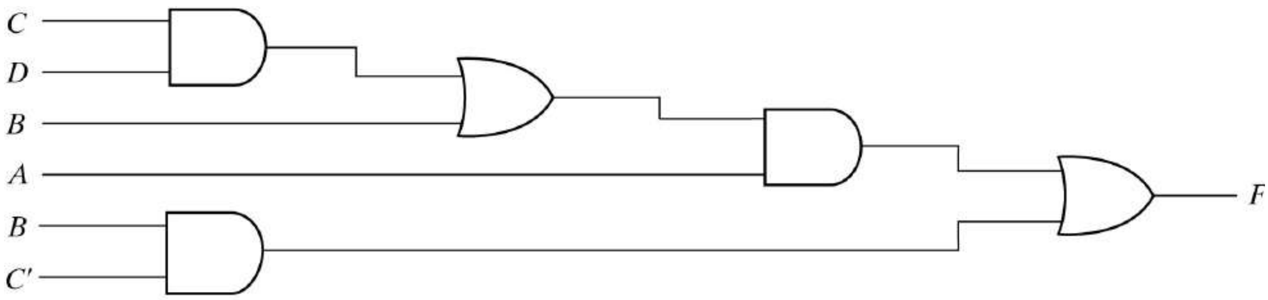
VE-VEYA Diyagramına Dönüştürme

VEDEĞİL lojik kapıları ile oluşturulmuş bir devre VE-VEYA kapıları şekline dönüştürülürken en sondan başlanır. En sondaki VEDEĞİL kapısı bir EVİRME-VEYA kapısına dönüştürülür. Bir örnekle açıklayalım:



Tüm küçük daireleri ortadan kaldıracak şekilde EVİRME-VEYA kapıları yerleştirilir.

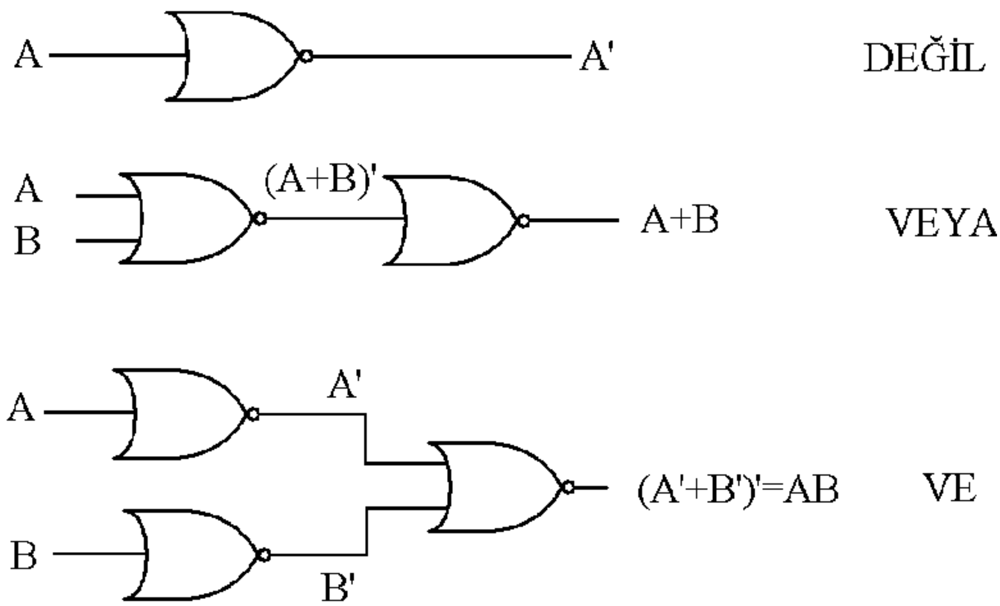




Devre VE-VEYA lojik kapıları ile gerçekleştirilmiş olur.

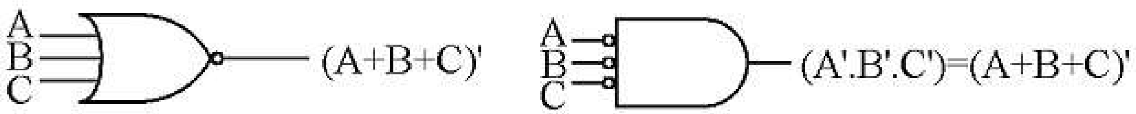
Çok Kademeli VEYADEĞİL Devreleri

VEYADEĞİL fonksiyonu VEDEĞİL fonksiyonun dualidir, bu nedenle VEDEĞİL lojiği için verilmiş olan ifadelerin tümünün duali VEYADEĞİL içinde geçerlidir.



Boole Fonksiyonlarının Gerçeklenmesi

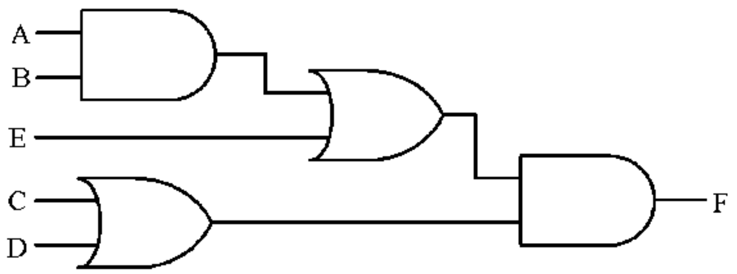
Aşağıda gösterilen iki lojik kapı DeMorgan teoremine göre birbirinin eşdeğeridir.



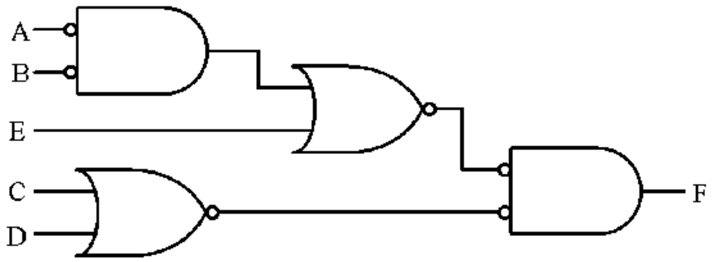
Daha önce anlatılan VEDEĞİL lojiğine dönüşüm yöntemi ile aynıdır.

Öncelikle tüm VEYA kapıları VEYA-EVİRME sembolü ile VEYADEĞİL kapısı ile değiştirilir. Daha sonra tüm VE kapıları EVİRME-VE kapısı ile değiştirilerek küçük daireler incelenir. Direkt olarak girişin herhangi bir değişkeni küçük daireye geliyorsa bu değişkenlerin DEĞİL'leri alınarak diğer EVİRME-VE kapıları VEYADEĞİL kapıları ile yer değiştirilir. Bunu bir örnek üzerinde inceleyelim.

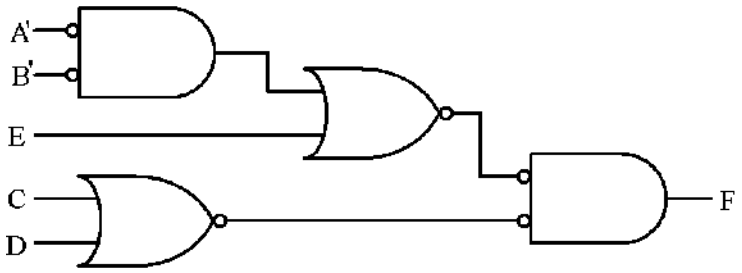
F=(AB+E)(C+D)



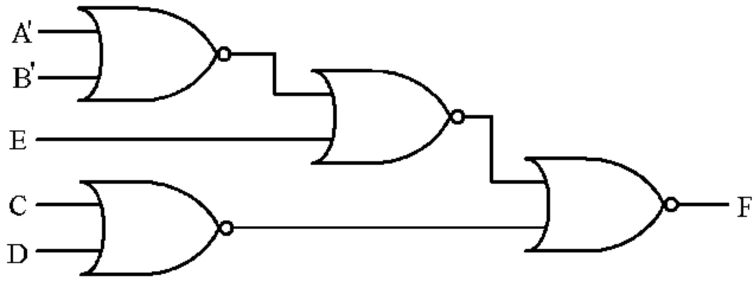
VE kapıları EVİRME-VE, VEYA kapıları VEYADEĞİL ile yer değiştirilir.



Küçük daireler incelenerek direkt girişe geliyorlarsa girişin DEĞİL'i alınır.



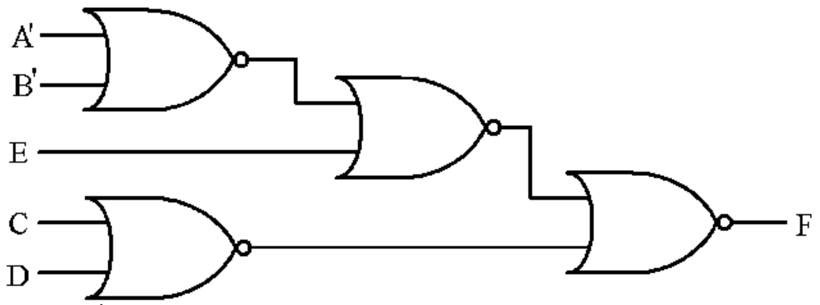
Daha sonra EVİRME-VE kapıları VEYADEĞİL kapısına dönüştürülür.



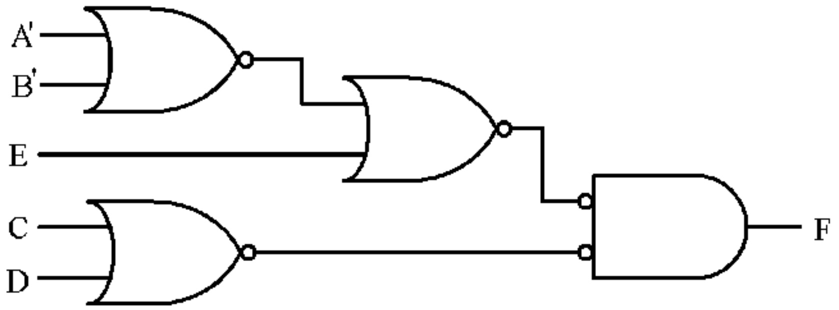
Analiz Yöntemi

VEYADEĞİL lojiğinden VE-VEYA lojiğine geri dönmek için en sondaki VEYADEĞİL kapısı EVİRME-VE kapısına dönüştürülür ve aradaki tüm küçük daireler ortadan kalacak şekilde ara katlara bakılarak uygun EVİRME-VE kapıları yerleştirilir.

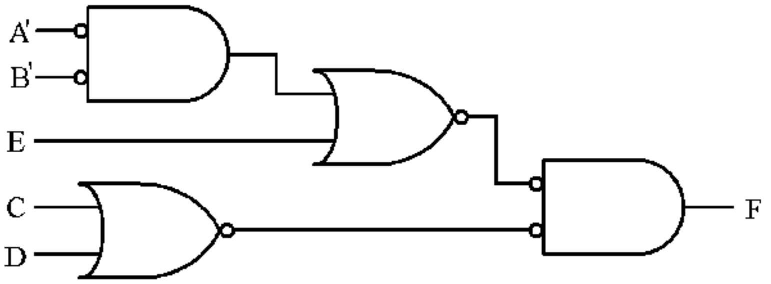
Bir önceki örneği ele alalım.



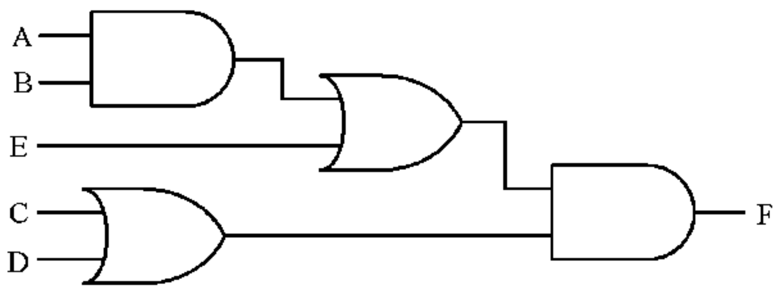
En son kata bir tane EVİRME-VE kapısı yerleştirilir.



Bir tane EVİRME-VE kapısı da A' ve B' girişlerindeki VEYADEĞİL kapısının yerine konmalıdır.



Tüm küçük daireler ortadan kaldırılmış olur. Girişteki değişkenlerin de tümleyenlerinin alındığına dikkat ediniz.



VEYADEĞİL kapıları ile dizayn edilmiş bir lojik devrenin Boole fonksiyonunu ya da doğruluk tablosunu elde etmek için tüm kapı çıkışlarına keyfi semboller verilerek VEDEĞİL lojiğinde olduğu gibi Boole fonksiyonu elde edilebilir.

ÖZEL VEYA Fonksiyonu

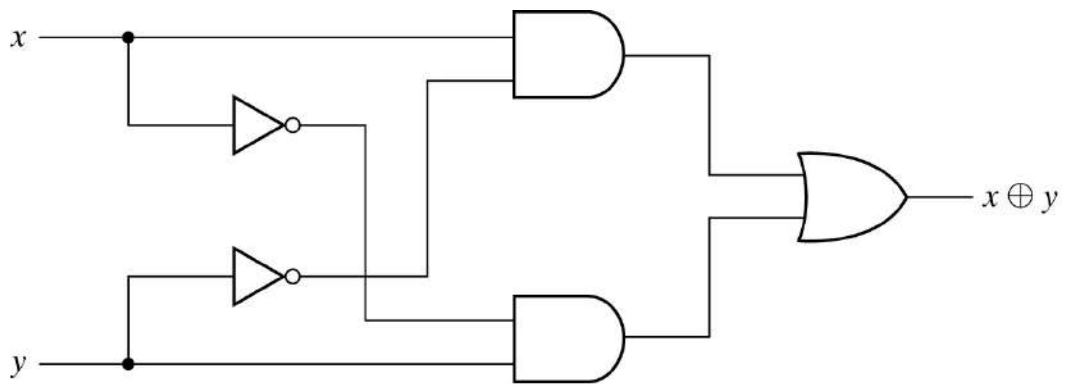
ÖZEL VEYA (XOR) ⊕ sembolü ile gösterilir ve sadece girişlerden birinin 1 olması durumunda çıkış 1 olur. Aşağıdaki işlemi yerine getiren lojik bir operatördür.

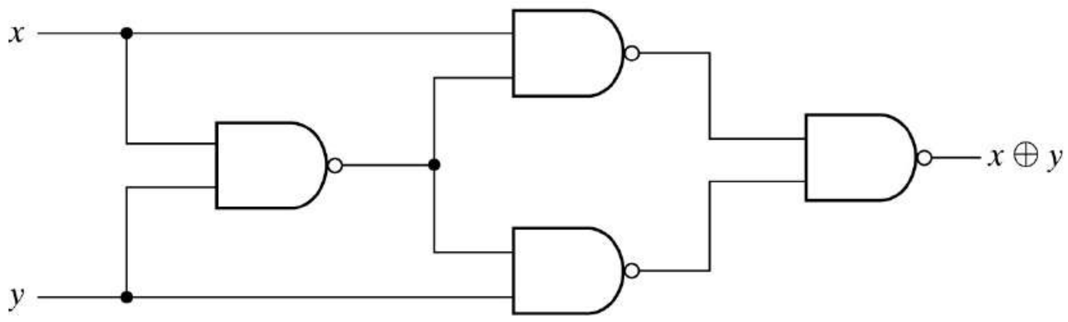
$x \oplus y=x'y+xy'$

ÖZEL VEYA fonksiyonunun tümleyeni ÖZEL VEYA DEĞİL (XNOR) olarak bilinir ve tanım fonksiyonu aşağıdaki gibidir. Her ikisinin de aynı değere sahip olduğu durumda çıkış 1'dir.

$(x \oplus y)'=(x'y+xy')'=(x'y)'(xy')'=(x+y')(x'+y)=xy+x'y'$

Aşağıda ÖZEL VEYA fonksiyonunu gerçekleştiren lojik devreler görülmektedir.





Tek Fonksiyon

Üç girişli bir ÖZEL VEYA kapısı düşünelim.

$A \oplus B \oplus C = (A'B+AB') \oplus C = (A'B+AB')C' + (A'B+AB')'C = A'BC'+AB'C'+ABC+A'B'C = \sum(1,2,4,7)$

Tek sayıda değişkenin 1 olması durumunda çıkış bir olduğu için bu tip fonksiyonlara tek fonksiyonlar denir. Tümleyenin de ise çift sayıda değişken 1 ise çıkış 1 olduğundan dolayı bu tip fonksiyonlara da çift fonksiyon denir. Genel olarak n-değişkenli bir ÖZEL VEYA fonksiyonu $2^n/2$ tane minterimin toplamından oluşan bir tek fonksiyondur.

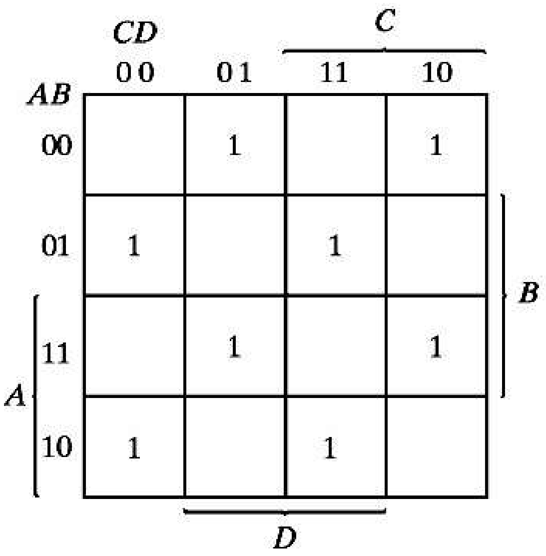
Dört değişkenli bir ÖZEL VEYA fonksiyonu incelendiğinde

$$A \oplus B \oplus C \oplus D = (A'B+AB') \oplus (C'D+CD') = (A'B+AB')' (C'D+CD') + (A'B+AB') (C'D+CD')'$$

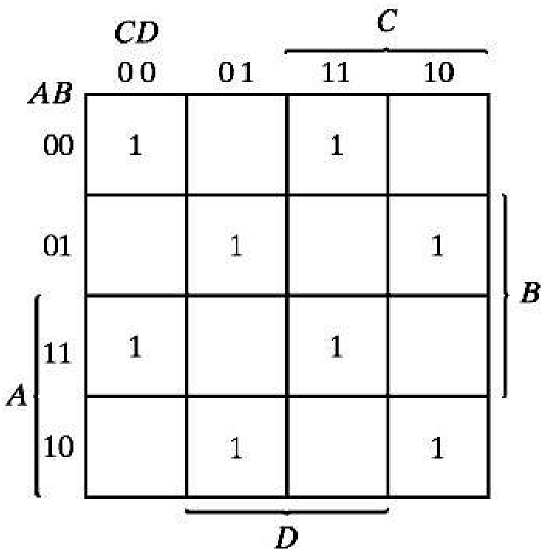
$$= (AB+A'B') (C'D+CD') + (A'B+AB') (CD+C'D')$$

$$= ABC'D+ABCD'+A'B'C'D+ A'B'CD'+ A'BCD+ A'BC'D'+ AB'CD'+ AB'C'D'$$

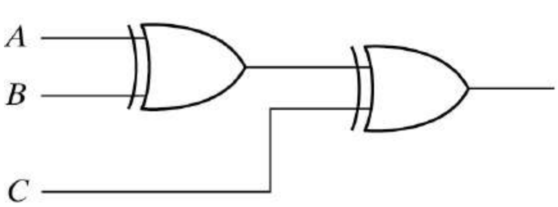
$$= \sum(1,2,4,7,8,11,13,14)$$



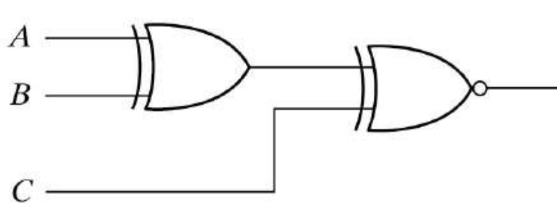
(a) Tek Fonksiyon
$F = A \oplus B \oplus C \oplus D$



(b) Çift Fonksiyon
$F = (A \oplus B \oplus C \oplus D)'$



3-girişli tek fonksiyon



3-girişli çift fonksiyon

Eşlik (Parite) Biti Oluşturulması ve Kontrolü

Hata tespiti ve düzeltme gerektiren sistemlerde ÖZEL VEYA fonksiyonlarından yararlanılır. Bilginin taşınması sırasında meydana gelebilecek hatalarda bir eşlik biti kullanılır. Eşlik biti, ikili mesaj içerisindeki 1'lerin sayısının tek ya da çift yapmak için kullanılan bir bittir. Göndericide eşlik bitini oluşturan devreye eşlik üreteci denir. Alıcıda ise eşlik bitini kontrol eden devreye ise eşlik kontrolörü denir.

Çift eşlikte, mesajdaki 1'lerin toplam sayısı çift olacak şekilde bir eşlik biti mesaja eklenir. Tek eşlik bitinde ise mesajdaki 1'lerin toplam sayısı tek olacak şekilde bir eşlik biti mesaja eklenir. Örneğin 3 bitlik bir mesajdaki 1'lerin sayısı tek ise üç girişli bir ÖZEL VEYA kapısına uygulandığında çıkış, mesajdaki bitlerin sayısı tek ise 1 olacaktır. Bu şekilde bir eşlik biti üreteci yapılabilir. Eşitlik kontrolü ise dört girişli bir ÖZEL VEYA devresi ile yapılır.

Çift eşlik üretecine ilişkin doğruluk tablosu

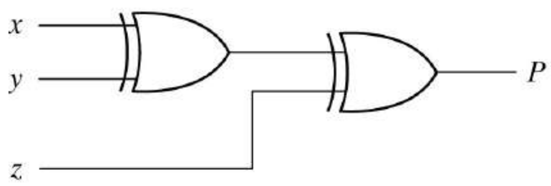
3 Bitlik Mesaj			Eşlik biti
x	y	z	P
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Eşlik üreteci çıkış

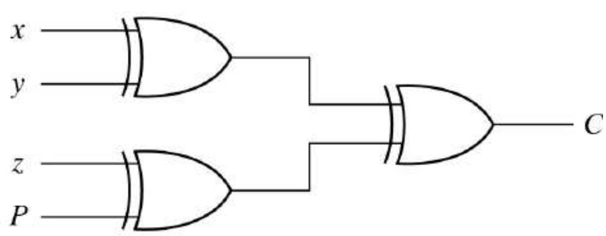
$P = x \oplus y \oplus z$

Eşlik kontrolörü çıkışı

$C = x \oplus y \oplus z \oplus P$



3-bit çift eşlik üreteci



4-bit çift eşlik kontrolü

Kontrolör çıkışı C, eğer hata varsa 1'e eşit olur. Alıcı tarafında tek sayıda bit varsa bu demektir ki taşıma sırasında bir hata olmuştur ve C çıkışı 1 olur. İlgili hatalı ve hatasız durumlara ilişkin tablo aşağıda verilmiştir.

Çift eşlik kontrolü doğruluk tablosu

Alınan 4 bit				Eşli hatası kontrolü
x	y	z	P	C
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Renkli olan bölgeler hatalı olan mesajları göstermektedir.

MSI VE PLD ELEMANLARI

Şimdiye kadar anlatılanlar, herhangi bir lojik fonksiyonu dizayn etmek için kapı sayısını azaltan yöntemler şeklinde idi. Bu maliyeti daha az sayıda kapı devresi kullanarak düşürmek içindi. Fakat devre tasarımında entegre devreler (IC) kullanıldığı zaman bu tip bir yol izlemek her zaman geçerli olmaz.

Tüm devreler devre karmaşıklığına göre

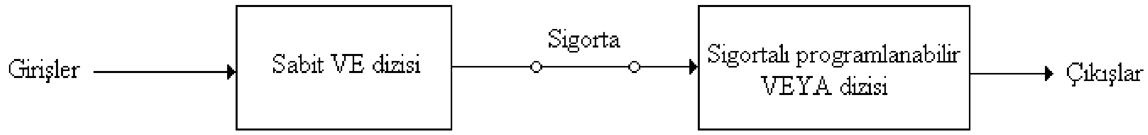
1- Küçük ölçekte tümleşiklik (SSI)
2- Orta ölçekte tümleşiklik (MSI)
3- Büyük ölçekte tümleşiklik (LSI)
4- Çok büyük ölçekte tümleşiklik (VLSI)

Olarak dörde ayrılmıştı. Çok sayıda birbirinden ayrı kapı içeren bir lojik devre bir birinden ayrı kapıları içeren SSI yapısı kullanılarak dizayn edilebilir. Kapı sayısının arttığı düşünülse bile ekonomik olarak tasarımcıya oldukça fazla avantaj sağlar.

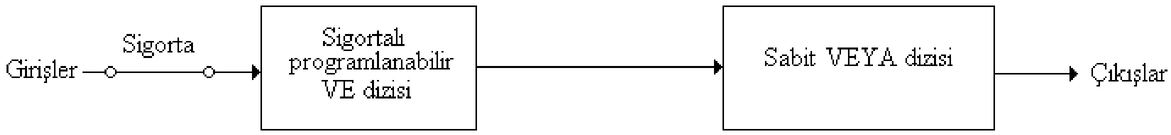
Sayısal devrelerin tasarımında bazı kombinezonal devreler oldukça yaygın olarak kullanılırlar. Bu tip devreler tüm devre şeklinde mevcut olup MSI elemanları olarak sınıflandırılırlar. MSI elemanları sayısal sistemlerde yaygın olarak kullanılan ortak özellikteki bazı özel fonksiyonları gerçekleştirirler. Bu özel fonksiyonlar, toplayıcı, çıkarıcı, karşılaştırıcı, kodlayıcı (decoder), kod çözücü (encoder) ve veri seçicilerdir (multiplexer).

MSI devreleri SSI devrelerine göre maliyeti oldukça fazla aşağı çeker. Programlanabilen Lojik Aygıt (PLD) elemanları ise içinde yer alan lojik kapıların bir birlerine sigortalar yardımıyla bağlı olduğu bir tüm devredir. Aygıtın programlanması, birbirinden ayrılması gereken hatlara ilişkin sigortaların attırılması ile yapılır. Böylece özel bir yapı oluşur. Programlama sözcüğü, devrenin içyapısının belirlenmesine yönelik bir donanım işlevini yerine getirmektir. PLD'deki kapılar VE ve VEYA dizilerinden oluşmaktadır. Bu diziler bir araya gelerek çarpımların toplamını oluştururlar. PLD'deki tüm sigortalar başlangıçta attırılmamış durumdadırlar. Aygıtın programlanması ile gerekli fonksiyona ulaşmak için ilgili sigortalar attırılır.

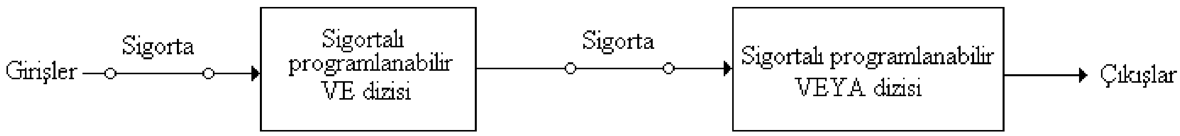
Bu bölümde üç PLD ve sayısal sistemlere uygulanması anlatılacaktır. VE-VEYA dizilerinin yerleşimine göre üç farklı PLD vardır. Bunlar şekilde görüldüğü gibi sabit bir VE dizisi ve çıkışta yer alan VEYA kapılarına ilişkin programlanabilir sigortalara sahiptir. Bu tip PLD'ler programlanabilir salt okunur bellek (PROM) olarak adlandırılırlar. Programlanabilir dizi lojiği (PAL) sigorta ile programlanabilir bir VE dizisi ve çıkışta sabit bir VEYA dizisinden oluşur. VE kapıları Boole fonksiyonundaki çarpım terimlerini oluşturmak üzere kullanılır. PLD'lerin daha esnek yapıda olanları programlanabilir lojik dizisi olarak bilinir. Hem VE hem de VEYA dizileri programlanabilir. Bu PLD elemanları daha sonra ayrıntılı olarak anlatılacaktır.



Programlanabilir Salt Okunur Bellek (PROM)



Programlanabilir Dizi Lojiği (PAL)



Programlanabilir Lojik Dizisi (PLA)

İkili Toplayıcı ve Çıkarıcılar

Daha önce anlatılan tam toplayıcılar kullanılarak elde bitini de göz önüne alarak n bitlik iki sayının toplamı gerçekleştirilebilir. Bunun için aşağıdaki tabloyu incelemek gerekir. A_i toplanılan B_i toplanan C_i giriş eldesi C_{i+1} çıkış eldesi S_i ise toplam olmak üzere dört bitlik iki ikili sayının toplamı (A=1011 B=0011 S=1110)

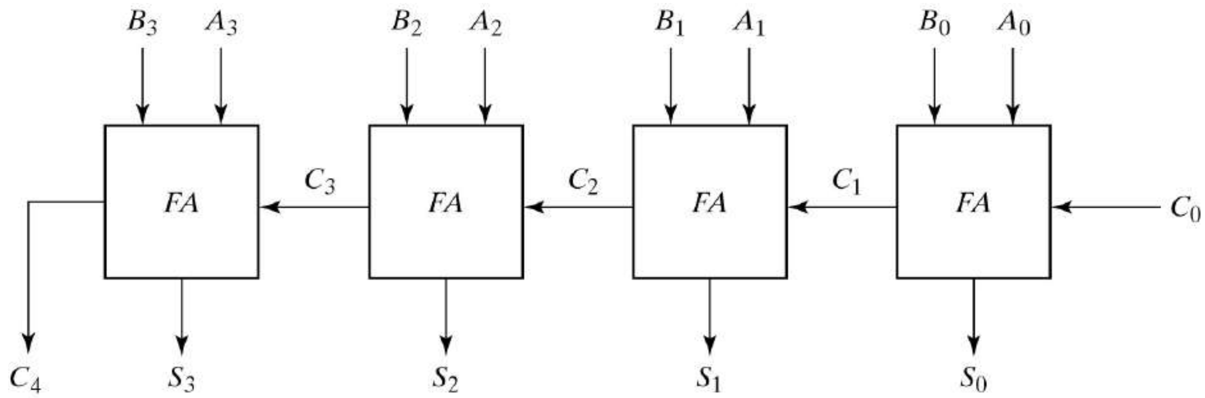
İndis i	3	2	1	0		Tam Toplayıcıdaki Karşılıkları
Giriş Eldesi	0	1	1	0	C_i	z
Toplanılan	1	0	1	1	A_i	x
Toplanan	0	0	1	1	B_i	y
Toplam	1	1	1	0	S_i	S
Çıkış Eldesi	0	0	1	1	C_{i+1}	C

Toplama işlemine en düşük anlamlı bitten başlanır. En düşük konumda yer alan C_0 eldesi sıfır olmalıdır. C_{i+1} tam toplayıcının herhangi bir andaki çıkış eldesidir. Bu kendinden sonra yer alan daha yüksek anlamlı bitin toplamına giriş eldesi olarak gönderilir.

A ve B gibi n bitlik iki sayının toplamı seri ya da paralel olmak üzere iki şekilde yapılır. Seri toplama yönteminde sadece bir tam toplayıcı ve oluşan eldeyi tutmak üzere bir saklama aygıtı kullanılarak yapılır. Toplama işlemini seri olarak gerçekleştirmek için her bir zaman aralığında A ve B ikili sayılarının bir çift bitleri toplayıcıya giriş olarak verilir. Oluşan toplam sonucundaki elde biti ise bir sonraki girişe elde biti olarak aktarılır. Paralel toplama yönteminde ise n adet tam toplayıcı kullanılır. A ve B sayısının tüm bitleri aynı anda toplayıcı girişlerine verilir. Elde bitleri ise tam toplayıcıların çıkış elde bitleri bir sonraki daha yüksek anlamlı bitlerin toplamını yapan toplayıcıya giriş eldesi olarak aktarılır. Eldeler üretilir üretilmez çıkışlardaki toplam değeri ortaya çıkar.

İkili Paralel Toplayıcı

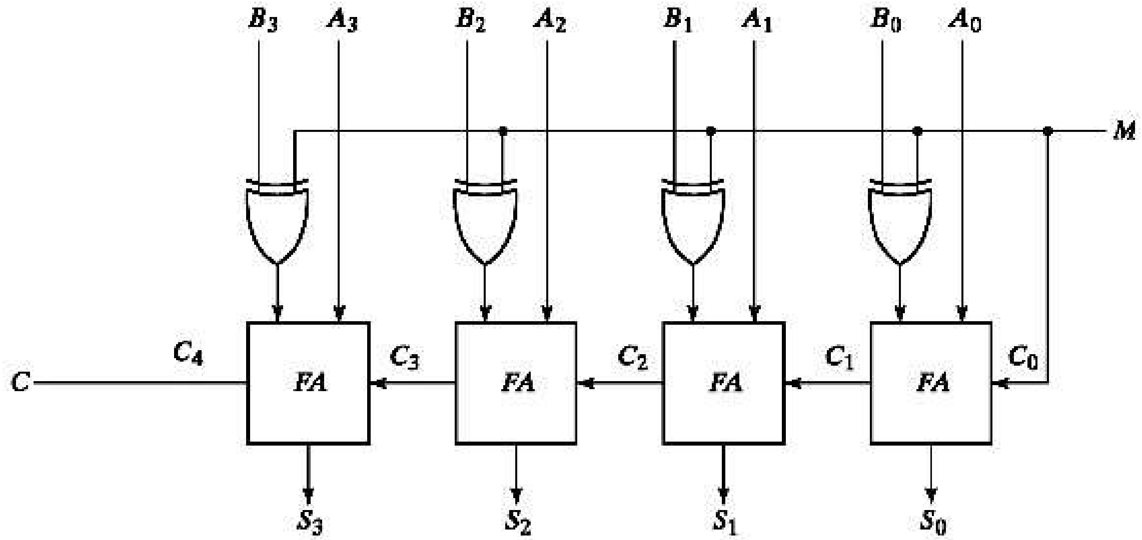
İkili paralel toplayıcı bir birine zincir şeklinde bağlanmış tam toplayıcılardan oluşur. her bir tam toplayıcının çıkış eldesi diğer bir tam toplayıcının giriş eldesine bağlanmıştır. Burada FA ile belirtilen elemanlar birer tam toplayıcı olup entegre tiptedir.



9 girişli bir devrenin tasarımında $2^9=512$ içerikli bir doğruluk tablosu gerekecektir. Bu nedenle MSI tipindeki tüm devreler ile bilinen fonksiyonların peş peşe eklenmesi ile doğruluk tablosu yapmadan fonksiyon gerçekleştirilebilir.

İkili Toplayıcı-Çıkarıcı

Bilindiği üzere sayısal bilgisayarlar çıkarma işlemi yapmaz. Bunun için toplama işlemi kullanılır. A-B işlemini yapmak için B'nin 2'ye tümleyeni alınıp A ile toplanması ile yapılmaktadır. 2'ye tümleyen işi B'nin 1'e tümleyeni alınıp en düşük anlamlı bite 1 eklenmesi ile elde edilir.



Yukarıdaki devrede M seçme biti 1 ise B⊕1=B' olduğu için devre A+B+1 işlemini gerçekleştirir. M girişi aynı zamanda C_0 elde girişine de bağlıdır. Dolayısıyla devre, A ile B'nin 2'ye tümleyeninin toplamını gerçekleştirir. M=0 ise B⊕0=B ve C_0 eldesi de 0 olduğu için toplama işi yapılır.

Eldenin Yayılması

Toplama işleminin paralel olarak yapılabilmesi için eldenin tüm paralel toplayıcıların girişlerinde aynı anda mevcut olması gerekmektedir. C_0 eldesi hesaplanmadan C_4 eldesinin ve diğerlerinin kapı girişlerinde değerleri olsa bile sırayla değerlerine ulaşırlar ve tüm elde değerleri bulunduktan sonra çıkışlarda toplam değeri görülebilir. Bu yayılma gecikmesi kullanılan tümdevrelere bağlı olarak değişir. Yayılma gecikmesi kapı kademe sayısı ve kapının yayılma gecikmesi ile çarpımına eşittir. C_0'dan C_4'e kadar elde biti 2×4 tane VEYA kapısından geçerek gelir. Eldenin yayılma gecikmesi ikili sayıların paralel toplanmasındaki hızı etkileyen en büyük faktörlerden biridir. Eldenin yayılmasını hızlandırmak için daha hızlı elemanlar kullanılabilir fakat fiziksel devrelerin hızları sınırlıdır. Hızı artırmak için diğer bir çözüm ise devre karmaşıklığını arttırarak elde yayılma süresi en aza indirgenebilir. Bu yöntem *elde öngörü* prensibidir. Aşağıdaki devreyi göz önüne alalım. Bu devre bir tam toplayıcı devresidir. Görüldüğü üzere ara katlarda P_i ve G_i olmak üzere iki yeni değişken tanımlanmıştır.

$P_i=A_i \oplus B_i$

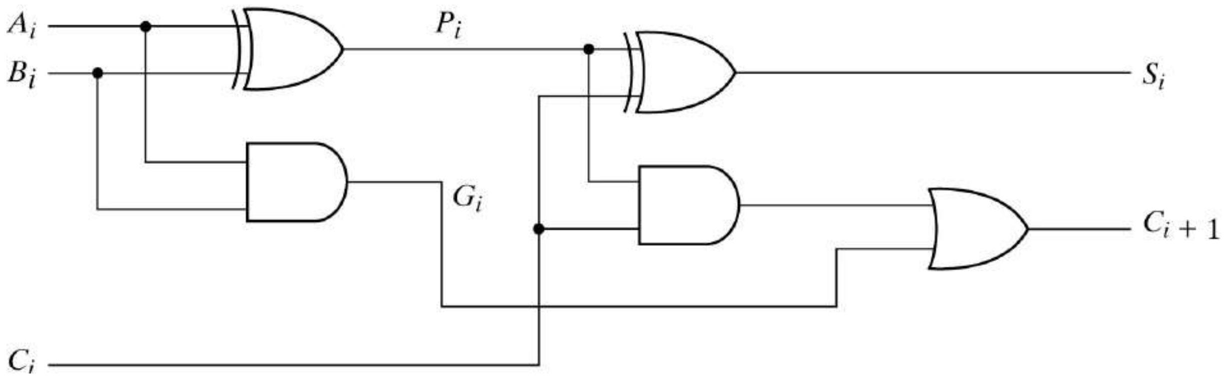
$G_i=A_iB_i$

Toplam ve elde çıkışları ise

$S_i= P_i \oplus C_i$

$C_{i+1}=G_i+P_iC_i$

G_i elde üreteci olarak adlandırılır. A_i ve B_i'den her ikisi de aynı anda 1 ise çıkış elde biti vardır. P_i ise elde yayılması olarak adlandırılır çünkü, hem C_i hem de C_{i+1} yayılması ile ilişkili bir terimdir.

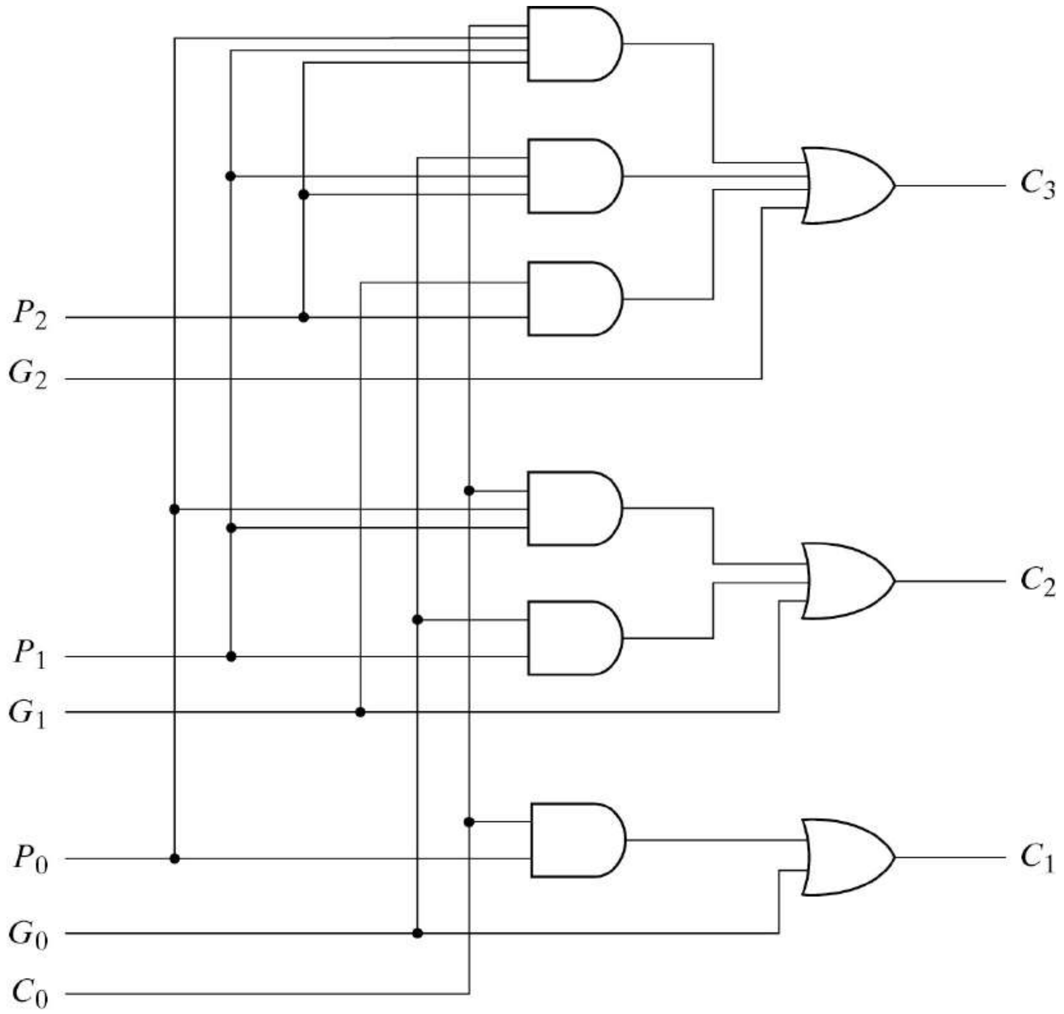


Çıkış elde biti için her kademede Boole fonksiyonları yazılacak olursa ve C_i önceki denklemlerde yerine yazılacak olursa;

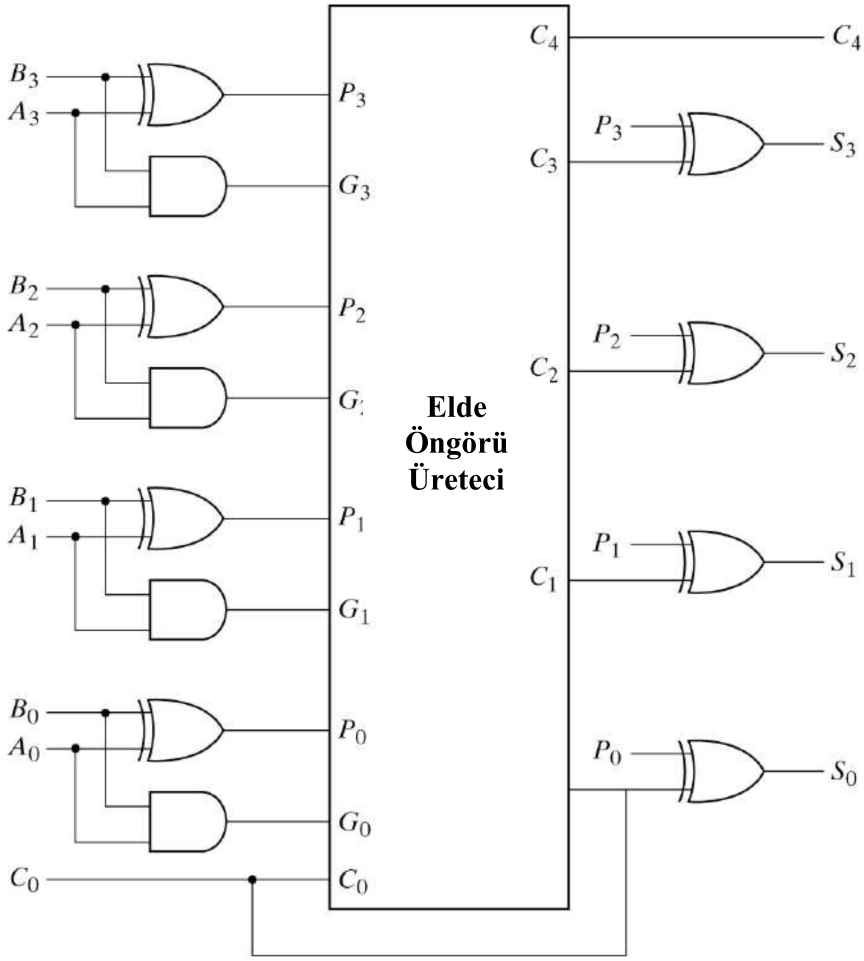
$C_1=G_0+P_0C_0$

$C_2=G_1+P_1C_1=G_1+P_1(G_0+P_0C_0)=G_1+P_1G_1+P_1P_0C_0$

$C_3=G_2+P_2C_2=G_2+P_2(G_1+P_1G_0+P_0P_1C_0)=G_2+P_2G_1+P_2P_1G_0+P_2P_1P_0C_0$



Bu Boole fonksiyonlarını gerçekleştiren lojik devre yukarıda görülmektedir. Bu şekilde tüm eldeler bir birilerinin değerlerini beklemeden oluşturulabilir. C_3, C_1 ve C_2 ile aynı zamanda oluşur.



Elde öngörülü 4 bitlik bir paralel toplayıcının yapısı yukarıda görülmektedir. Burada tüm kapıların gecikme süreleri aynı zamanda olmaktadır. Dolayısıyla tüm toplam çıkışları aynı gecikme sürelerine sahip olduklarından dolayı aynı anda üretilirler. C_4 eldesi ise denklemlerde gösterilmemiştir fakat yerine koyma yöntemi ile elde edilebilir.

$C_4=G_3+P_3C_3=G_3+P_3(G_2+P_2G_1+P_2P_1G_0+P_2P_1P_0C_0)=G_3+P_3G_2+P_3P_2G_1+P_3P_2P_1G_0+P_3P_2P_1P_0C_0$

Onlu Toplayıcı

Bilgisayarlar ve hesap makineleri ikili kodlanmış şekilde onlu gösterimdeki sayıları kullanarak aritmetik işlemleri yürütürler. Onlu toplayıcılar her onlu haneyi kodlamak için 8 giriş ve elde biti için bir giriş ve çıkış için ise 4 bit veri ve bir bit de elde biti için olmak üzere 9 girişli 5 çıkışlı olurlar.

Doğruluk tablosu oluşturarak toplayıcı yapmak için 2^9=512 içerikli bir doğruluk tablosu gerekecektir. Bu doğruluk tablosundaki etkisiz koşullarda göz önüne alındığında tablonun büyük bir kısmı bu etkisiz koşullarla dolacaktır. Bunu bilgisayarla yaptığımızda ise büyük ihtimalle düzensiz kapı desenleri ile karşılaşılacaktır. Buna alternatif bir yol olarak sadece kullanılan kombinasyonların bulunduğu ikili tam toplama devreleri kullanılabilir. Çıkış ise onlu kodların geçerli olduğu ikili kodlar şeklinde düzenlenmelidir.

BCD Toplayıcı

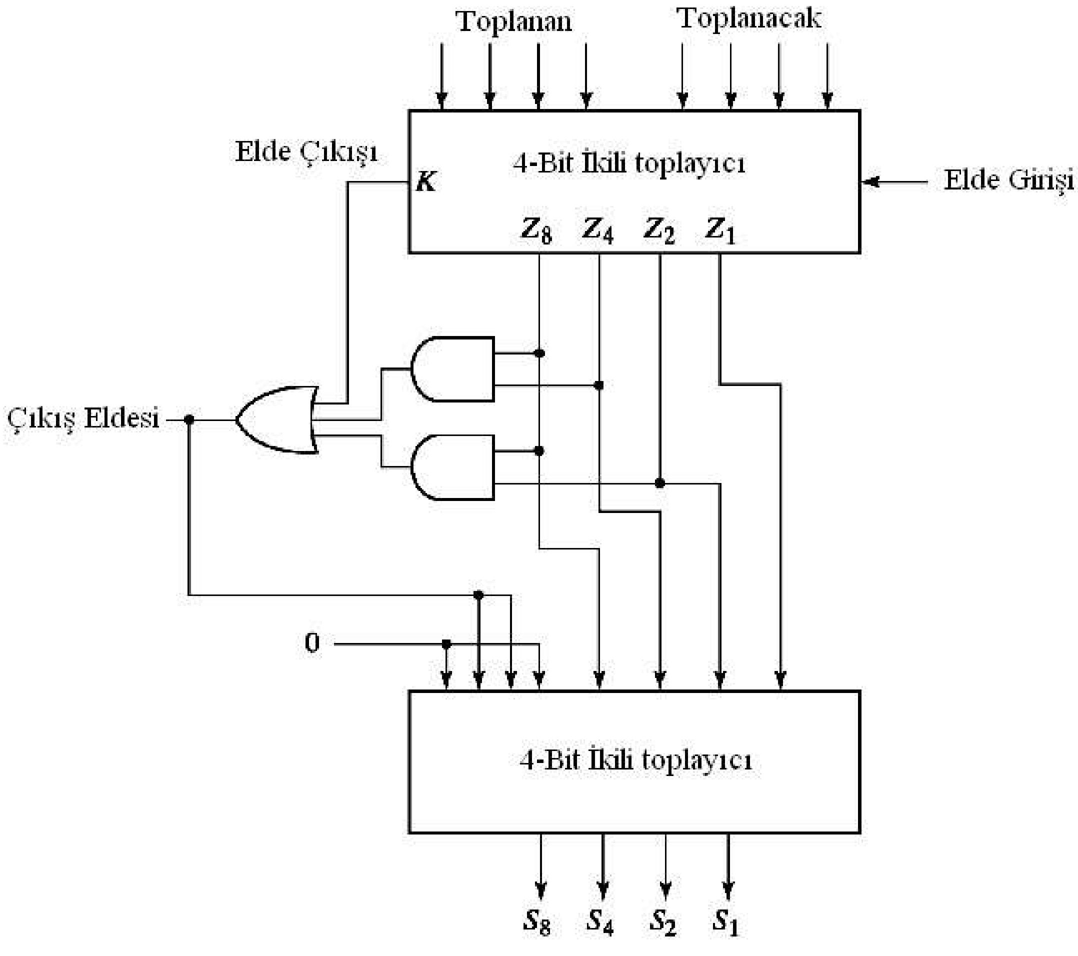
Onlu sayıların 9 ‘u geçemeyeceği düşünülürse ve elde biti de 1 olursa toplam 9+9+1=19 olacaktır. Toplayıcı toplama sonucunu ikili formda oluşturacak ve 0-19 arasında bir sayı üretecektir. Bu ikili sayılar tabloda K, Z_8, Z_4, Z_2 ve Z_1 olarak gösterilmiştir. K eldeyi, Z’nin alt indisleri ise 4 bit karşılık düşen 8, 4, 2, 1 ağırlıklarını göstermektedir.

	İkili Toplam					BCD Toplam				Onlu
K	Z_8	Z_4	Z_2	Z_1	C	S_8	S_4	S_2	S_1	
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9
0	1	0	1	0	1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19

Tabloya dikkat edilecek olursa, ikili toplam sonucunun 9(01001) değerinden sonra değiştiği görülmektedir. BCD’de 9’dan büyük sayılar için herhangi bir gösterim karşılığı yoktur. Bu durumda C ile belirlenen bir çıkış eldesi oluşacaktır. Görüldüğü gibi 9’dan büyük değerler için BCD koduna 6 (0110) eklenmiştir. O halde iki sayı toplandıktan sonra çıkış eldesi olarak belirtilen C’nin bu toplama eklenmesi gerekmektedir. Bir başka deyişle bir düzeltme işlemi gerekmektedir. Toplam işlemi onlu 9 modülüne göre yapıldığı düşünülürse işlemin anlaşılması daha kolaylaşabilir. Eğer Çıkış eldesi C, 1 olursa bu durumda bir düzeltme olacağı anlaşıldığı için ikinci 4 bitlik toplayıcının girişlerine daha önce elde edilen toplam ile beraber düzeltme sayısı 0CC0 eklenmelidir (C yazan yerlere 1 ya da 0 gelecek) 0 gelirse zaten toplam etkilenmeyecek ve herhangi bir düzeltmeye gerek olmadığı anlaşılacaktır. O halde çıkış eldesi C'ye ne zaman gerek duyulacağı araştırılmalıdır. Tablo incelenecek olursa K, Z_8, Z_4, Z_2 ve Z_1'e bağlı olarak C'nin değiştiği görülür. O halde elde edilecek Boole fonksiyonu aşağıdaki gibi olmalıdır. (<u>Bu düzeltme devresinin nasıl elde edilir?</u>)

$C=K+ Z_8Z_4+Z_8Z_2$

Aşağıda bir BCD toplayıcının blok diyagramı görülmektedir.



Genlik Karşılaştırıcısı

İki sayının karşılaştırılması bu iki sayıdan hangisinin büyük, küçük ya da bir birlerine eşit olup olmadıklarının araştırılmasıdır. Bir genlik karşılaştırıcısı A ve B gibi iki sayıyı karşılaştıran kombinezonal bir devredir. Burada, 4 bitlik iki sayının karşılaştırılması için bir algoritma türetme yöntemi incelenecektir. Algoritma, izlendiğinde bir problemin çözümünü veren sonlu adımlar grubuyla belirlenen bir yoldur.

A ve B gibi 4 bitlik iki ikili sayısını göz önüne alalım. Sayıların katsayıları aşağıdaki gibi azalan şekilde olsun.

$A=A_3A_2A_1A_0$

$B=B_3B_2B_1B_0$

Bu hanelerden aynı indeks numarasına sahip olanların değerleri eşit ise sayılar birbirine eşittir. Her bitin eşitlik ilişkisi bir eşdeğer fonksiyonla aşağıdaki gibi ifade edilebilir.

$x_i=A_iB_i+A'_iB'_i \quad i=0,1,2,3$

Burada i indeksli bit değerlerinin her ikisi de bir birine eşitse x_i 1 olacaktır. Bu durumda bu işlem tüm giriş bitlerine uygulanacak ve elde edilen tüm x_i değerleri VE işlemine tabi tutulacaktır. Eğer VE kapısının çıkışı 1 ise A=B demektir.

$(A=B)=x_3x_2x_1x_0$

En düşük anlamlı bitten başlanacak olursa diğer üç hanenin eşit olduğunu düşünelim

$x_3x_2x_1=1$ olduğu durum eğer $A_0B'_0=1$ ise $A_0=1$ ve $B_0=0$ demektir ve A>B ‘dir anlamına gelir. $A'_0B_0=1$ ise $A_0=0$ ve $B_0=1$ olup A<B ‘dir anlamına gelir.

$A>B \ x_3x_2x_1A_0B'_0$

$A<B \ x_3x_2x_1A'_0B_0$

x_0 değerlerinin eşitliği bir önceki adımda kontrol edildi. x_3 ve x_2’nin eşit olduğu düşünülürse

$x_3x_2=1$ bu durumda 1 indeksli hanelere bakmak gerekecektir. $A_1B'_1=1$ ise A>B $A'_1B_1=1$ ise A<B’dir anlamına gelir.

$A>B \ x_3x_2A_1B'_1$

$A<B \ x_3x_2A'_1B_1$

x_0 ve x_1 değerlerinin eşitliği bir önceki adımlarda kontrol edildi. x_3 ‘ün eşit olduğu düşünülürse

x_3=1 bu durumda 2 indeksli hanelere bakmak gerekecektir. $A_2B'_2$=1 ise A>B A'_2B_2=1 ise A<B'dir anlamına gelir.

A>B $x_3A_2B'_2$

A<B $x_3A'_2B_2$

x_3 dışındaki diğer tüm eşitlikler kontrol edildiğine göre

A>B $A_3B'_3$

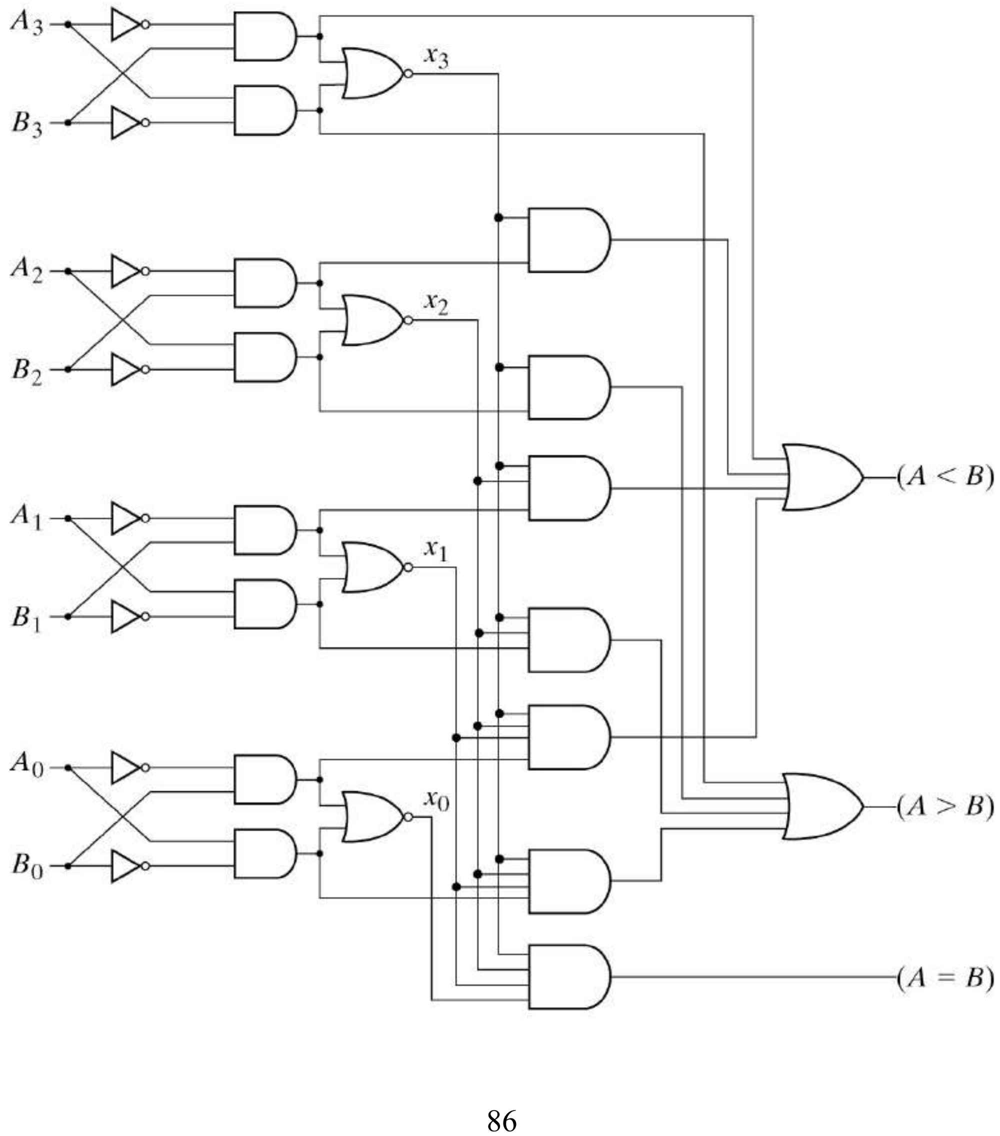
A<B A'_3B_3

Tüm şartların aynı anda sağlanması için elde edilen tüm Boole fonksiyonları toplanmalıdır (VEYA).

$(A>B)=x_3x_2x_1A_0B'_0+x_3x_2A_1B'_1+x_3A_2B'_2+A_3B'_3$

$(A<B)=x_3x_2x_1A'_0B_0+x_3x_2A'_1B_1+x_3A'_2B_2+A'_3B_3$

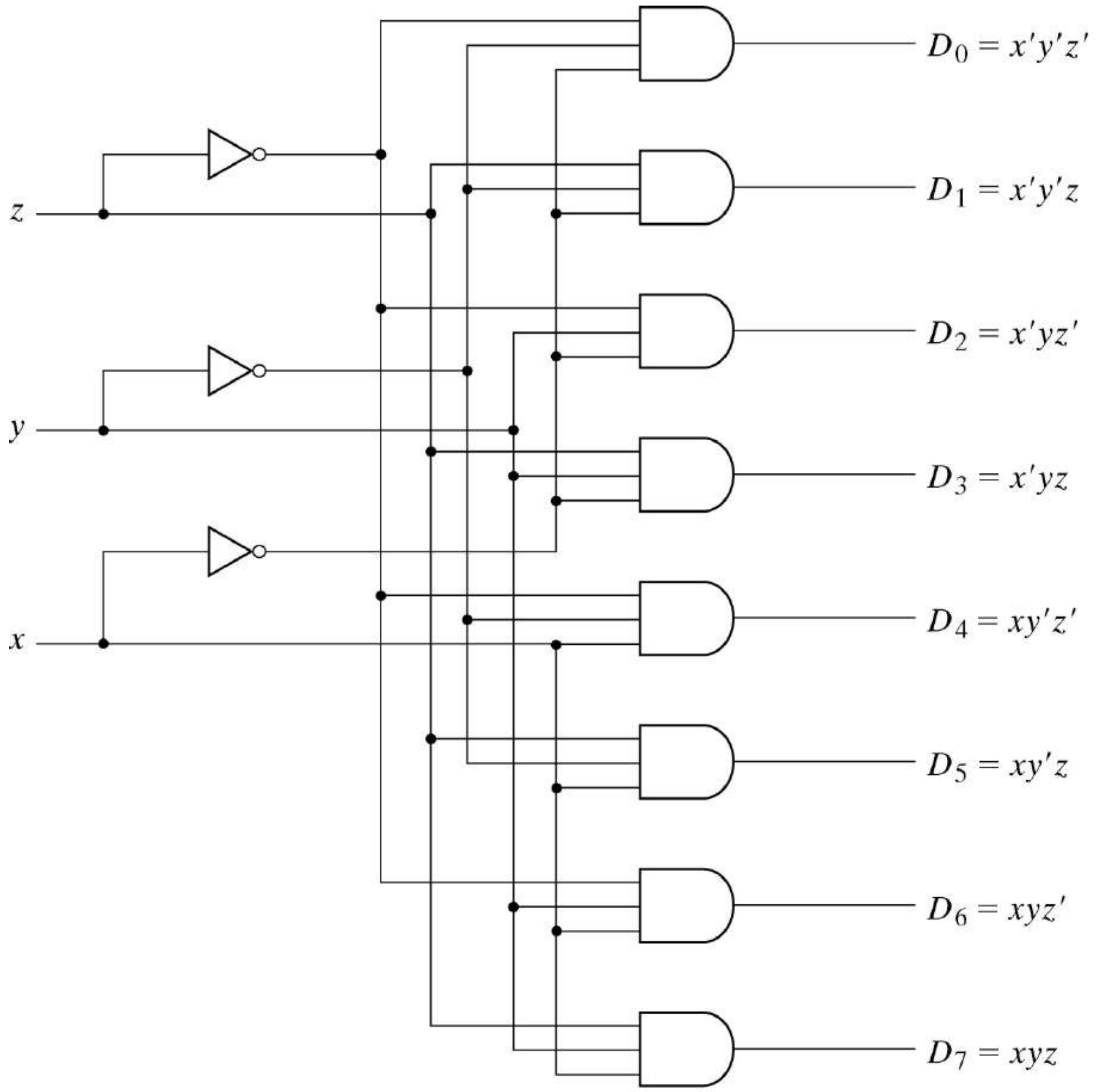
Hangi durum sağlanıyorsa aşağıdaki kombinezonal devrede ilgili çıkış 1 olur.



Görüldüğü üzere devre belirli bir desene sahiptir. Giriş bit sayısının arttığı durumda Boole fonksiyonlarına bakıldığında bu desen rahatlıkla görülmektedir. Bu devrenin girişine BCD formundaki sayılar da uygulanabilir.

Kod Çözücüler ve Kodlayıcılar (Decoders, Encoders)

n bitlik bir ikili bilgi 2^n bağımsız kodlanmış veriyi temsil eder. 3 bitlik bir bilginin $2^3=8$ tane kombinasyonu vardır. Aşağıdaki devre 3'ten 8'e bir kod çözücüdür. Bu 8 tane kombinasyon minterimleri temsil eder. D ile belirtilen çıkışlar indeks numaralarına göre m ile belirtilen minterimlerdir.



3 girişli bir Boole fonksiyonunda olası minterimler x'y'z', x'y'z, x'yz', x'yz, xy'z', xy'z, xyz' ve xyz 'dir. Bu şekilde girişten gelen kombinasyonun hangi minterimi temsil ettiği belirlenebilir.

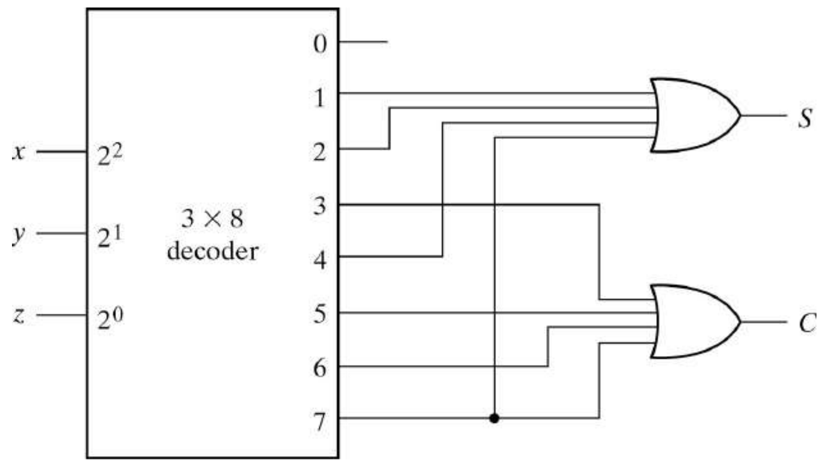
Girişler			Çıkışlar							
x	y	z	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Kombinezonal Lojik Uygulaması

Bir kod çözücü girişine gelen değişkenlere göre minterimler oluşturur. O halde kombinezonal bir devre kod çözücüler ile dizayn edileceği zaman Boole fonksiyonu toplamların çarpımı şeklinde ifade edilmeli ve uygun çıkışlar daha sonra bir VEYA kapısı ile çıkış fonksiyonu olarak kullanılmalıdır.

Örnek: Bir tam toplayıcıyı kod çözücüler kullanarak tasarlayınız.

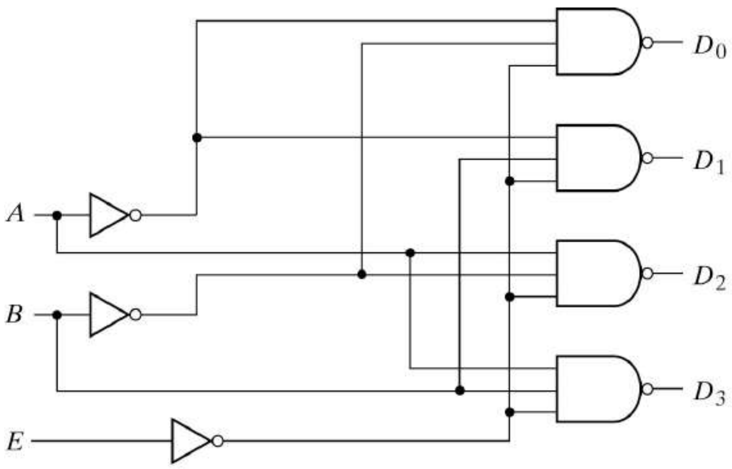
Daha önce tam toplayıcı için toplam Boole fonksiyonu $S(x,y,z)=\sum(1,2,4,7)$ elde çıkış Boole fonksiyonu da $C(x,y,z)=\sum(3,5,6,7)$ olarak bulunmuştu. O halde kod çözücünün 1,2, 4 ve 7 nolu çıkışları bir VEYA kapısının girişi olacak şekilde toplam sonucunu oluşturur ve 3, 5, 6 ve 7 nolu çıkış uçları ise elde çıkışı üretecek şekilde bir VEYA kapsının girişine bağlanır. Bağlantılar aşağıdaki gibidir.



Çok sayıda değişkenden oluşan bir fonksiyonun minterimlerinin toplamını oluşturmak için çok girişli bir VEYA kapısı kullanmak gereklidir. k adet minterimli bir F fonksiyonu yerine 2^n-k minterimli F' tümlenmiş şekli de kullanılabilir. Sadece çıkışta istenilen fonksiyonu türetmek için bir eviriciye gerek vardır. Bir fonksiyonun minterimleri sayısı $2^n/2$'den büyükse (n=3 ise 4'den büyük olmalı) bunun tümleyeni daha az minterim içereceğinden F' 'ne ait minterimlerin toplamı bulunur ve daha sonra bu toplamın tümleyeni alınarak F oluşturulur. Bütün bu işlemleri bir arada yürütmek için bir VEYADEĞİL kapısı kullanılır. Az sayıda minterimle ifade edilen uygulamalarda kod çözücüler en iyi çözümü sağlar.

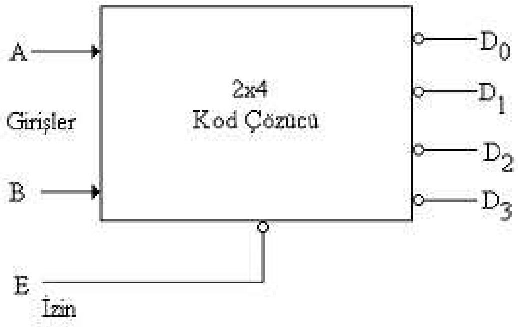
Tekilleyici (Demultiplexer)

Bazı entegre kod çözücüler VEDEĞİL kapıları ile oluşturulurlar. Aşağıdaki devre izin girişli 2'den 4'e kod çözücü olarak bilinir. E girişi 1 olduğu sürece diğer girişlerin değeri ne olursa olsun çıkış sürekli 1 seviyesinde kalır. Eğer E girişi 0 ise kod çözücüye izin verilmiş olur ve bu durumda girişteki kodlara göre tümlenmiş bir kod çözücü gibi çalışır. Kod çözme işlemi E=0 iken yapılır ve çıkışlar 0 durumdayken seçilir.

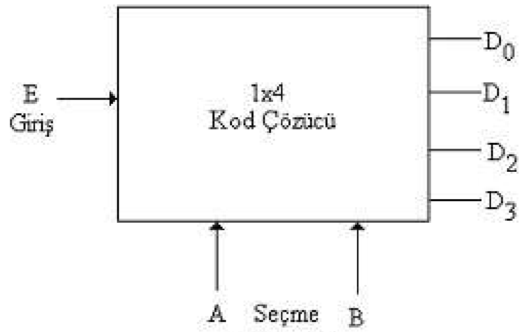


E	A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

Çıkıştaki küçük daireler tüm çıkışların tümleyenin alındığını gösterir. İzin girişli bir kod çözücü bir veri tekilleyici olarak bilinir. Tekilleyici, girişteki veriyi alarak bunu 2^n hattan birine iletir. Belirli bir çıkış hattının seçilmesi n adet seçme hattıyla gösterilir. E giriş değişkenindeki bilgi iki seçme hattı A ve B'nin ikili değerlerine bağlı olarak çıkış hatlarından birine aktarılır. Örneğin, AB=10 için D_2 girişi E ile aynı değerde tutulurken diğer hatların tümü 1 değerinde tutulur.

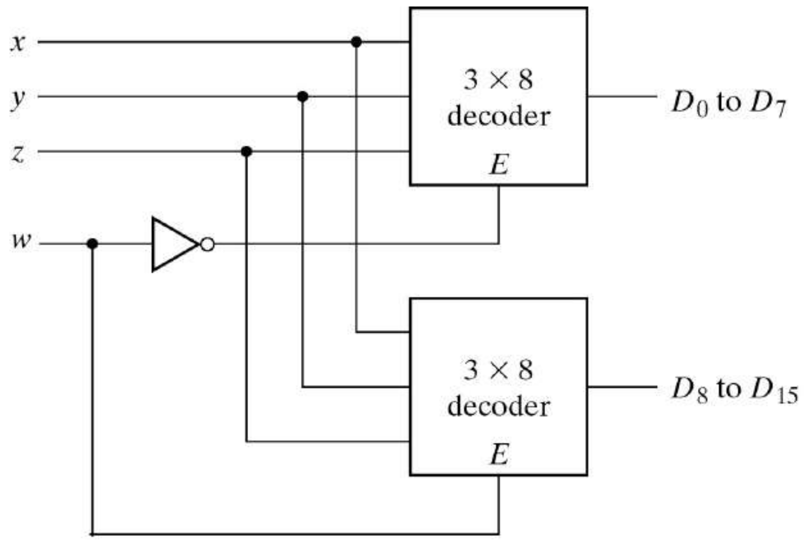


İzin girişli kod çözücü



Tekilleyici

Kod çözücü tekilleyici devreleri daha büyük bir kod çözücü tasarlamak üzere kullanılabilir. Örneğin, iki tane 3x8'lik kod çözücüden bir tane 4x16 kod çözücü elde edilebilir. 4x16 'lı bir kod çözücünün 4 tane girişi vardır. İlk 8 değer için en ağırlıklı bit (aşağıdaki şekilde w) sürekli 0 değerinde kalırken diğer 8 değer için sürekli 1 seviyesinde kalır. Bu bit, izin girişi olarak kullanılırsa aşağıdaki şekilde görüldüğü üzere iki tane 3x8'lik kod çözücü tekilleyiciden bir tane 4x16'lık kod çözücü elde edilebilir.



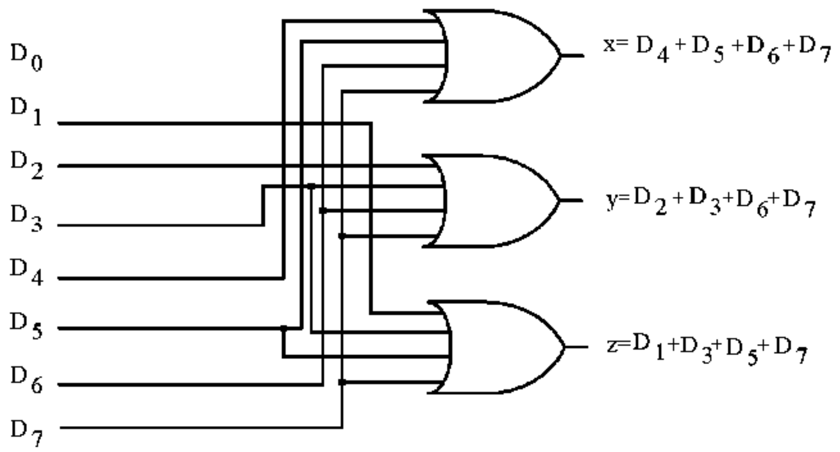
Kodlayıcılar(Encoders)

Bir kodlayıcı bir kod çözücünün tersi işlemi yapan sayısal devredir. 2^n girişten n adet çıkış üretir. Çıkış hatları, giriş değerine karşı düşen ikili kodu üretir. Sekizden ikiliye dönüşüm için doğruluk tablosu aşağıda verildiği gibidir. Görüldüğü üzere herhangi bir anda sadece bir giriş

1 değerine sahiptir. Birden fazla girişin 1 olması durumunda ise devrenin herhangi bir anlamlı çıkışa sahip olmadığı düşünülür.

Girişler								Çıkışlar		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

Bir kodlayıcı, girişleri doğruluk tablosundan elde edilen VEYA kapıları ile gerçekleştirilebilir.



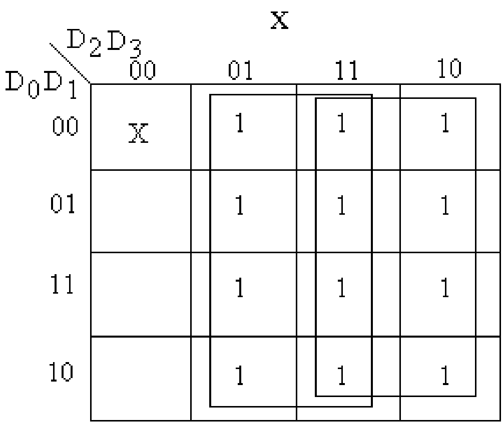
Bu kodlayıcı da bazı belirsiz durumlar söz konusudur. Örneğin, her iki girişe aynı değer geldiği durumu göz önüne alalım. D_3 ve D_6 aynı anda 1'e eşit olursa, aynı anda tanımlı iki giriş olmadığı için kodlayıcı çıkışı 111 olup ne 3'ü ne de 6'yı verir. Bu karmaşayı çözmek için kodlayıcı devrelerde yalnız bir girişin kodlandığından emin olmak için öncelik oluşturulmalıdır. Büyük indeksli girişlerin daha öncelikli olduğu varsayılırsa aynı anda giriş olan D_3 ve D_6 için çıkış 110 olacaktır. Sekizliden ikiliye diğer bir belirsizlik de tüm girişlerin sıfır olduğu durumda tüm çıkışların sıfır olmasıdır. Burada problem D_0 1 olduğu durumda tüm çıkışların sıfır olmasıdır. Bu belirsizlik, girişlerin hiçbirinin aktif olmadığını belirleyen ilave bir çıkış kullanılarak çözülebilir.

Öncelik Kodlayıcısı

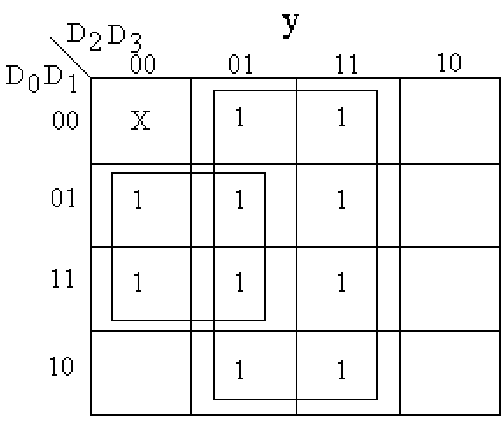
Bir öncelik kodlayıcısı, öncelik fonksiyonunu da içeren bir kodlayıcıdır. Öncelik kodlayıcısının işlevi, iki veya daha fazla giriş aynı anda 1'e eşit olduğunda en yüksek öncelikli girişin göz önüne alınmasını sağlamaktadır. Dört girişli öncelikli bir kodlayıcının doğruluk tablosu aşağıda verilmiştir. X'ler dikkate alınmayan etkisiz koşullardır. D_3 girişi en öncelikli giriştir.

Girişler				Çıkışlar		
D0	D1	D2	D3	x	y	V
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1

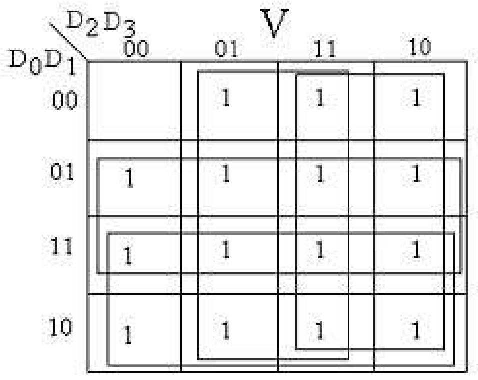
Eğer D_3 girişi 1 ise diğer girişlerin değerlerinin bir önemi yoktur. xy çıkışı 3 (11) değerini alır. D_3 girişi 0 ve D_2 girişi 1 ise diğer iki girişin bir önemi yoktur ve çıkış 10 (2) değerini alır. Burada geçerli çıkış göstergesi V ile gösterilmiş olup sadece bir veya daha fazla giriş 1 değerini aldığında değeri 1 olur. x ve y girişleri için doğruluk tablosu oluşturulursa;



$x = D_3 + D_2$

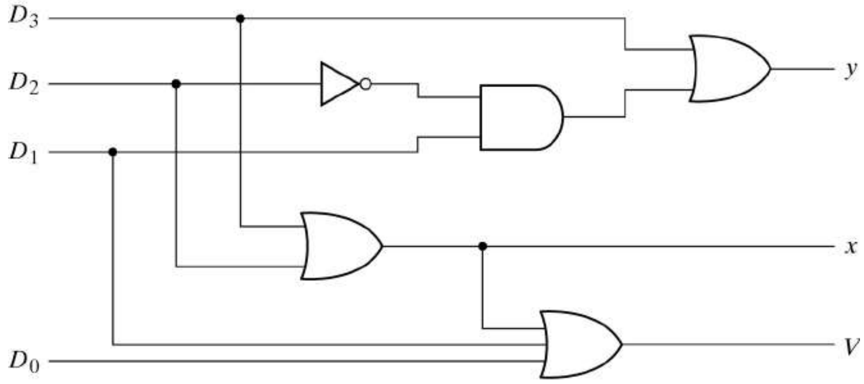


$y = D_3 + D'_2 D_1$



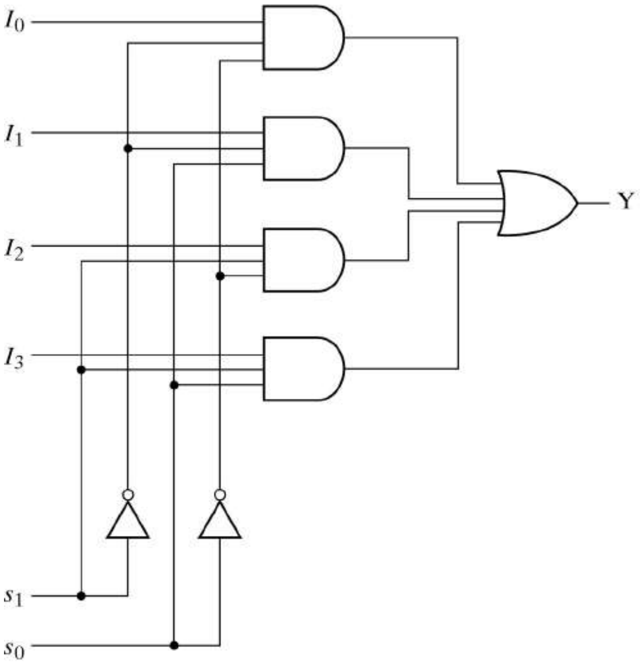
$$V = D_0 + D_1 + D_2 + D_3$$

Yukarıdaki doğruluk tablolarından elde edilen dört girişli öncelik kodlayıcısı aşağıda görülmektedir.



Veri Seçiciler (Multiplexer)

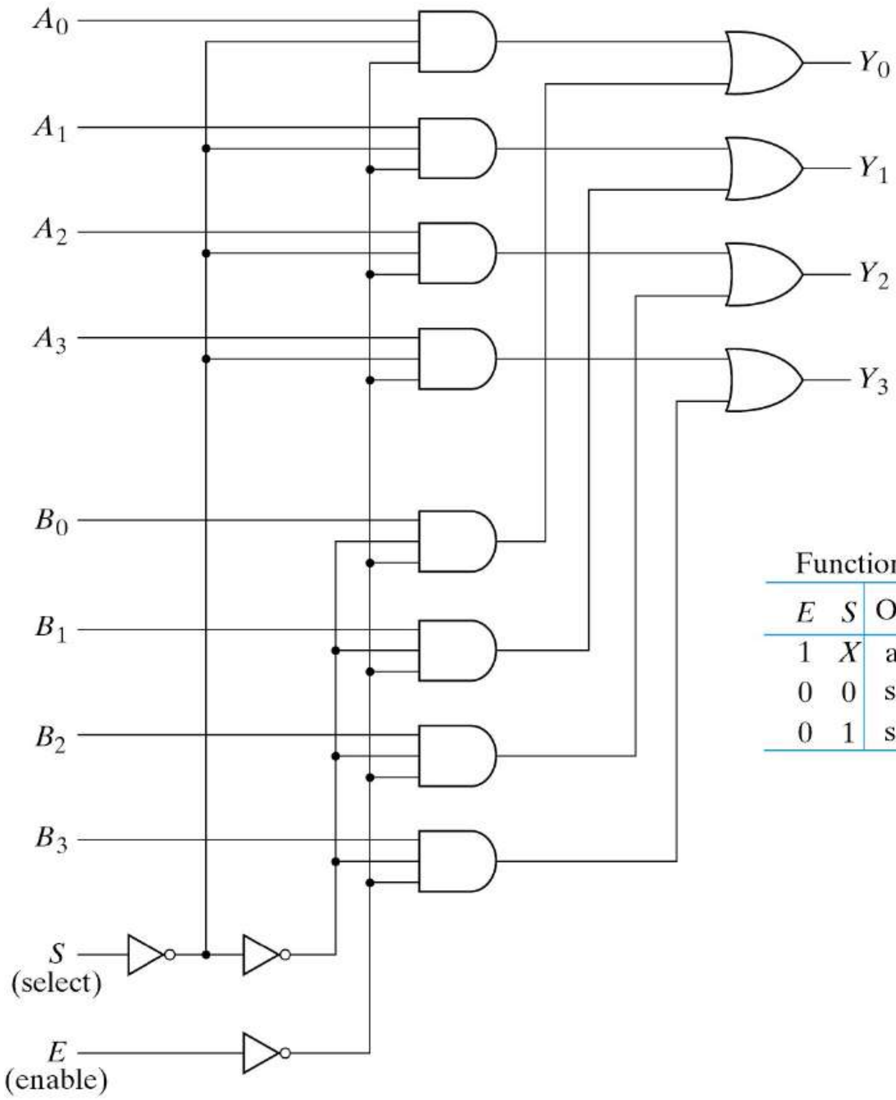
Veri seçmenin anlamı çok sayıdaki bilgi birimlerinin daha az sayıdaki kanallara veya yollara aktarılmasıdır. Sayısal bir veri seçici bir veya daha fazla giriş hattından ikili bilgiyi seçen ve bunu tek giriş hattına bağlayan kombinezonal devredir. Özel bir giriş hattının seçilmesi bir grup seçme hattı ile kontrol edilir. 2^n adet giriş hattı varsa n tane seçme ucu vardır. Aşağıda, 4'ten 1'e veri seçici lojik diyagramı, doğruluk tablosu ve blok diyagramı gösterimi görülmektedir. S_1 ve S_0 veri seçme uçlarına bağlı olarak girişlerden gelen I_0, I_1, I_2 ve I_3 verilerinden bir tanesi çıkışa aktarılır.



s_1	s_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

(b) Function table

Devrenin çalışmasını anlamak için S_1S_0=10 olduğu durumu düşünelim. VE kapılarının girişlerinin en az bir tanesi 0 olursa çıkış 0 olacağı için I_0, I_1 ve I_3 girişlerinin uygulandıkları VE kapılarının bir veya iki girişi 0 olduğu için çıkışları da 0 olacaktır. O halde VEYA ile sonlandırılmış kombinezonal devre doğrudan I_2 girişinin değerini alacaktır. Böylece seçilen bir girişten çıkışa bir yol sağlanmış olacaktır.



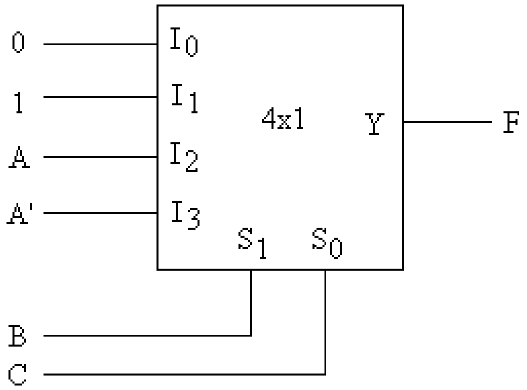
Function table

E	S	Output Y
1	X	all 0's
0	0	select A
0	1	select B

Veri seçicideki VE kapıları ve eviriciler kod çözücü bir devreye benzer ve giriş seçme hatlarının kodunu çözer. Veri seçiciler genellikle multiplexer kelimesinin kısaltması olan MUX ile adlandırılırlar. Tıpkı bir kod çözücüde olduğu gibi veri seçicide de bir izin girişi vardır. İzin girişinin belirli durumlarında tüm çıkışlar yasaklanır, diğer durumlarda ise devre veri seçme işlemini normal bir şekilde yürütür. İzin girişi daha büyük sayısal sistemlerde iki veya daha fazla veri seçme tüm devresinin bir birine bağlandığı durumlarda kullanılır. Aşağıdaki devrede dörtlü 2'den 1'e veri seçme tüm devresi görülmektedir. Eğer E, izin girişi 1 ise veri seçme işlemi yasaklanmıştır demektir. Bu durumda S girişinin aldığı değerlerin bir önemi yoktur. Devrede dört tane veri seçici vardır. Bu çıkışlar ya A ya da B değerlerinden bir tanesini alırlar. Seçme işleminin yapılması için E girişi sürekli 0 seviyesinde tutulmalıdır. Eğer S=0 ise A girişleri, S=1 ise B girişleri seçilecektir demektir.

Boole Fonksiyonlarının Uygulanması

n+1 elemanlı bir Boole fonksiyonu varsa, bu değişkenlerden n tanesi seçme ucu olarak kullanılır, veri seçicinin diğer uçlarına ise kalan değişkenin değili, kendisi, 1 ve 0 uygulanır.



	$\bar{B}\bar{C}$	$\bar{B}C$	$B\bar{C}$	BC
	I_0	I_1	I_2	I_3
A'	0	1	2	3
A	4	5	6	7
	0	1	A	A'

Minterim	A	B	C	F
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

F (A,B,C)=$\sum$(1,3,5,6) Boole fonksiyonunu göz önüne alalım.

3 değişkenli bir Boole fonksiyonu olduğuna göre B ve C değişkenleri seçme uçları olarak seçilir. A, A', 1 ve 0 girişleri ise veri seçicinin girişlerine uygulanır. Kullanılacak olan minterimler m_1 (A'B'C), m_3(A'BC), m_5(AB'C) ve m_6(ABC')'dir. Veri seçicide seçme uçları BC=00 için I_0 girişi, BC=01 için I_1, BC=10 için I_2 ve BC=11 için I_3 girişi seçilir. BC=00 için m_0 ve m_4 minterimleri gerçeklenecek ve her ikisi de 0 çıkışı ürettiği için I_0 0'a bağlanmalıdır. BC=01 için, m_1 ve m_5 minterimleri gerçeklenecek ve her ikisi de 1 çıkışı ürettiği için I_1 1'e bağlanmalıdır. BC=10 için, m_2 ve m_6 minterimleri gerçeklenecek ve m_2 için 0, m_6 için 1 değeri çıkış olduğu için A değeri I_2 girişine bağlanır. BC=11 için, m_3 ve m_7 minterimleri gerçeklenecek ve m_3 için 1, m_7 için 0 değeri çıkış olduğu için A' değeri I_3 girişine bağlanır.

Genel bir yöntem verilecek olursa;

Minterimler için seçilen değişken sıralaması, ABC… olsun. BCD…. Değişkenleri B'den başlanarak en yüksek ağırlıklı seçme ucundan S_0'a doğru bağlanır. Aşağıdaki şekilde daire içine alınmış terimler kullanılan minterimlerdir. Eğer bir sütunda hiçbir terim daire içine alınmamışsa o giriş 0'a, tek bir terim daire içersine alınmış ise daire hangi satırda ise A ya da A' değişkenine, iki terim daire içersine alınmış ise 1'e bağlanmalıdır.

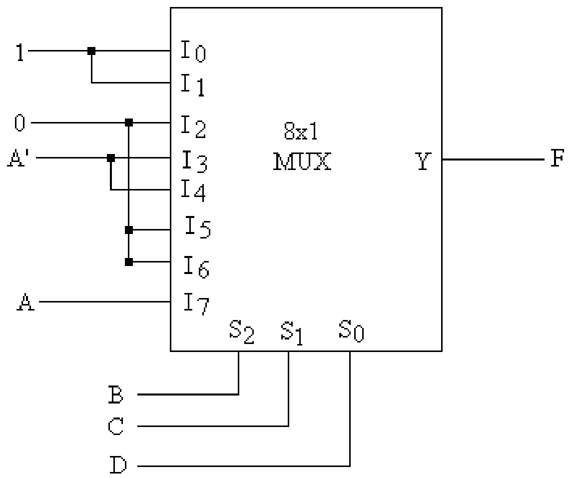
	B'C'	B'C	BC'	BC
	I_0	I_1	I_2	I_3
A'	0	1	2	3
A	4	5	6	7
	0	1	A	A'

Örnek: $F(A,B,C,D)=\sum(0,1,3,4,8,9,15)$ minterimleri ile belirlenmiş Boole fonksiyonunu veri seçiciler kullanarak gerçekleştiriniz.

Seçme uçları $B(S_2)$, $C(S_1)$, $D(S_0)$ olarak belirlenir. n değişkenli olduğuna göre 2^{n-1}x1 'lik bir veri seçici gereklidir. Önce uygulama tablosu oluşturulur.

	B'C'D'	B'C'D	B'CD'	B'CD	BC'D'	BC'D	BCD'	BCD
	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
A'	0	1	2	3	4	5	6	7
A	8	9	10	11	12	13	14	15
	1	1	0	A'	A'	0	0	A

Bu uygulama tablosundan gerekli devre gerçekleştirilir.



Kombinezonal devrelerin uygulanması için kod çözücü ve veri seçici yönteminin karşılaştırılması:

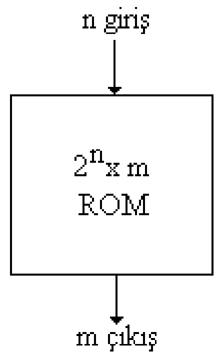
- ✓ Kod çözücü yöntemi her bir fonksiyon çıkışı için bir VEYA kapısı gerektirir. Fakat tüm minterimlerin oluşturulması için bir kod çözücü yeterlidir.

- ✓ Her bir fonksiyonun üretilmesi için ayrı bir veri seçici gereklidir ve çıkışlarının VEYA kapısı ile sonlandırılmasına gerek yoktur.
- ✓ Az çıkışlı bir devre veri seçici ile çok çıkışlı bir devre kod çözücü ile tasarlandığında daha az tüm devre elemanı kullanılır.
- ✓ Her iki eleman da kombinezonal devrelerin tasarımında kullanılabilmelerine rağmen kod çözücüler ikili bilgilerin çözülmesinde, veri seçiciler ise çok girişli ve tek çıkışlı sistemler arasındaki yolun seçilmesinde kullanılırlar.

Salt Okunur Bellek (Read-Only Memory:ROM)

Bir kombinezonal devrenin üretilmesi için minterimlerin toplamlarının üretilmesi yeterlidir. Bir kombinezonal devre, minterimler bir kod çözücü ile elde edilip toplamlar da bir VEYA kapısı kullanılarak gerçekleştirilebilir. Bir salt okunur bellek hem kod çözücünün hem de VEYA kapılarının içerisinde bulunduğu bir tümdevre kılıfıdır. ROM, karmaşık kombinezonal devrelerin tümdevre içinde gerçeklenmesini sağlayan veya ikili bilgiyi sürekli saklayan hafıza elemanı olarak kullanılır.

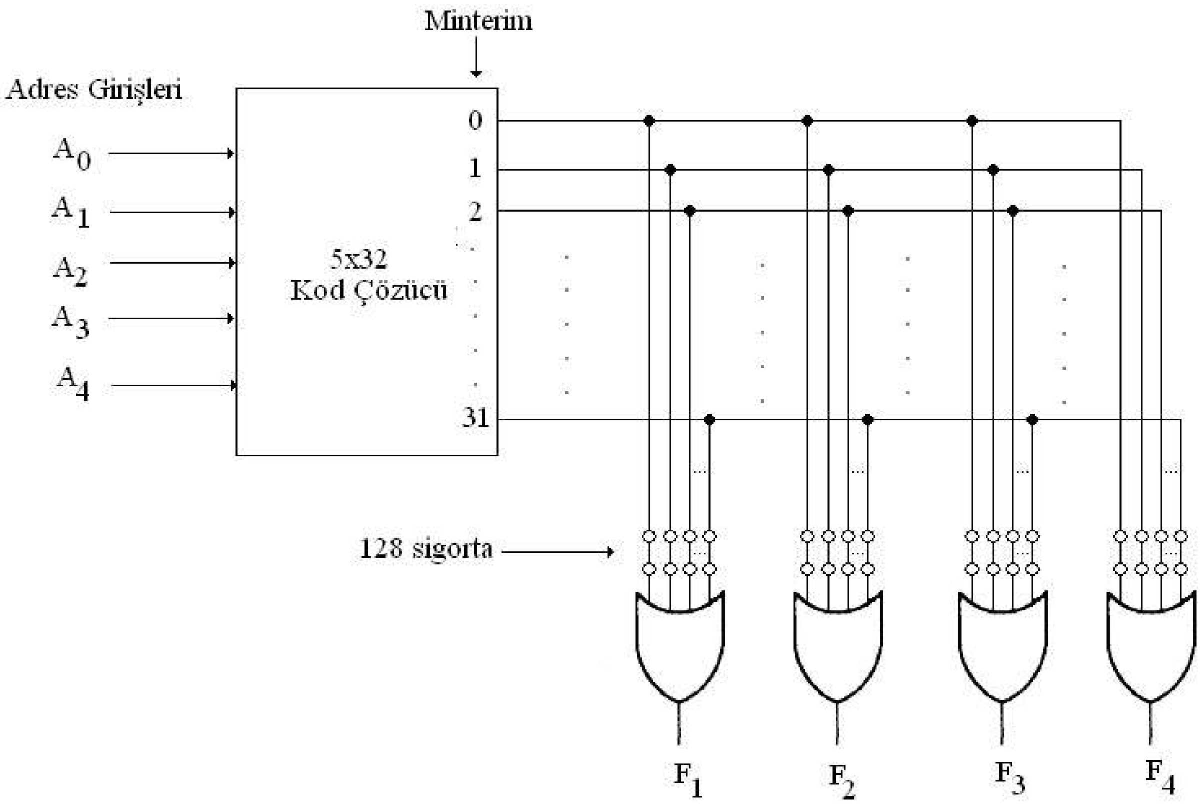
ROM esas olarak içinde ikili bilgileri sürekli olarak saklayan bir hafıza elemanıdır. Saklanacak ikili bilgi tasarımcı tarafından belirlendikten sonra bu ikili bilgiyi temsil eden desen eleman içerisine yerleştirilir. ROM'lar belirli bir yapı için programlanabilen iç elektronik sigortalar ile donatılmış olarak kullanıma hazırdırlar. Desen bir kere oluşturulduktan sonra enerjisi kesilse dahi desen aynı şekilde aygıtın içerisinde kalır. Bir ROM'a ilişkin blok diyagramı aşağıda gösterilmiştir.



Bu ROM n adet giriş, m adet çıkış hattından oluşmuştur. Giriş değişkenlerinin oluşturduğu bit kombinasyonları *adres*, çıkış değişkenlerinin oluşturduğu bit kombinasyonları ise *sözcük* (*kelime*) olarak adlandırılır. Sözcük başına düşen bit sayısı çıkış hattı sayısı olan m'e eşittir. Bir adres aslında n değişkenli minterimlerden birini gösteren bir ikili sayıdır. Dolayısıyla, n giriş değişkenine bağlı 2^n tane adres vardır. Bir çıkış sadece bir adres tarafından seçilebildiği için ROM'da 2^n tane de farklı sözcük vardır. Bu sözcükler birimde saklanır ve herhangi bir anda girişteki adres değerlerine bağlı olarak çıkış hattından elde edilir. Bir ROM'da sözcük sayısı 2^n ile sözcük başına düşen bit sayısı da m ile karakterize edilir.

32x8'li bir ROM'u göz önüne alalım. Birim, her biri 8 bit olan 32 farklı sözcüğe sahiptir. Bunun anlamı, 8 tane çıkış hattı olduğu ve her birimde 32 farklı sözcük saklandığıdır. Bu sözcüklerden herhangi biri çıkış hattına uygulanabilir. 32x8'lik bir ROM'da 2^5=32 olmak üzere 5 giriş vardır. Bu girişler adresleri belirlerler. 00000 girişi için 0 adresi, 11111 girişi için ise 31 adresi seçilip çıkış hattına bu adresteki sözcük iletilebilir.

ROM bir kod çözücü ve sigortalı VEYA kapıları içerir. Çıkıştaki sigortalar isteğe bağlı olarak attırılarak minterimlerin toplamı şeklinde ifade elde edilebilir. Aşağıdaki şekilde 32x4'lük bir ROM görülmektedir. 5 giriş hattı kod çözücü ile adresleri belirlendikten sonra çıkış hattı üzerindeki sigortalı VEYA kapılarında bulunan toplam 128 sigortadan istenilenler attırılıp desen oluşturulur.



Kombinezonal Lojik Uygulamaları

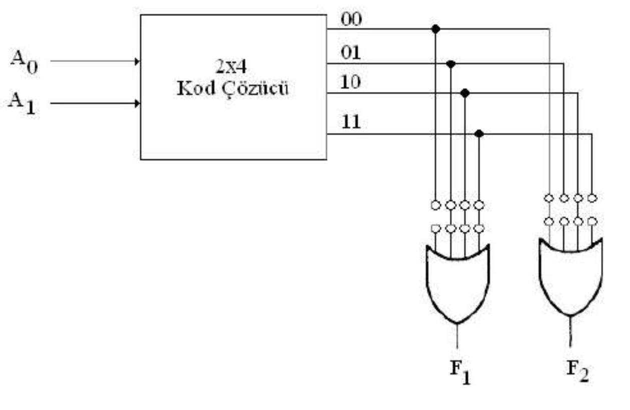
ROM’un lojik diyagramına bakılacak olursa, çıkışların her birinde n adet giriş değişkenine bağlı olarak tüm minterimlerin toplamı yer almaktadır. Daha önceki konulardan hatırlanacağı gibi her bir fonksiyon minterimlerin toplamı şeklinde ifade edilebilir. Elde edilmek istenen fonksiyonda bulunmayan minterimler çıkış hattı üzerindeki sigortalar attırılarak toplamdan çıkartılabilir. Bu şekilde ROM’un çıkışlarında istenen çıkışı temsil eden bir Boole fonksiyonu ifadesi elde edilebilir. n girişli m çıkışlı bir kombinezonal devre için $2^n \times m$ boyutunda bir ROM’a ihtiyaç vardır. Sigortaların attırılması ROM’un programlanması anlamına gelir.

Örnek olarak aşağıdaki devreyi inceleyelim. İki girişli, iki çıkışlı bir kombinezonal devre tasarlanmak istenmektedir. Boole fonksiyonunun minterimleri aşağıdaki gibi verilmiştir.

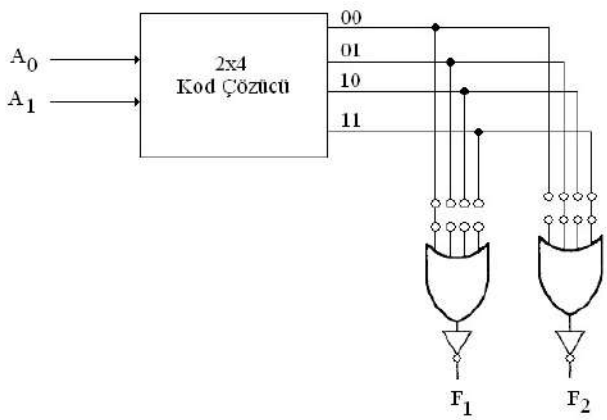
$$F_1\,(A_1,A_0)=\Sigma(1,2,3)$$

$$F_2\,(A_1,A_0)=\Sigma(0,2)$$

A_1	A_0	F_1	F_2
0	0	0	1
0	1	1	0
1	0	1	1
1	1	1	0



VE-VEYA kapıları ile ROM



VE-VEYA-EVİRME kapıları ile ROM

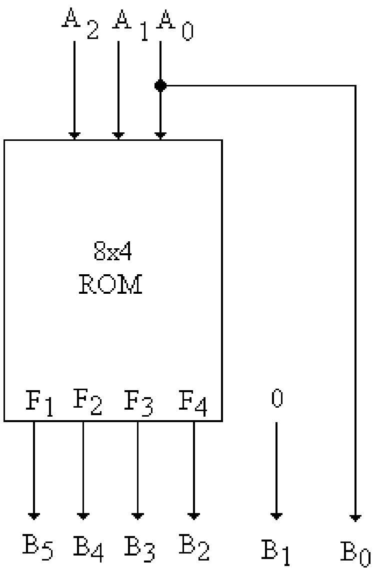
Pratikte böyle bir uygulama için kesinlikle ROM kullanılmaz. Çünkü sistemin yapısı oldukça basittir. ROM’un kullanılmasının avantajı daha karmaşık uygulamalarda ortaya çıkmaktadır. Boole fonksiyonu ile belirlenmiş ifadede iki giriş (n) (A_1,A_0) ve iki çıkış (m)(F_1, F_2) olduğuna göre $2^n \times m$, yani 4×2’lik bir ROM’a ihtiyaç vardır. Aynı zamanda girişte 2×4’lük bir kod çözücüsü olmalıdır. Doğruluk tablosu oluşturmak hangi minterimlerin toplama katılacağını görmek açısından kolaylık sağlar. Şekilde görüldüğü üzere F_1 ve F_2 için minterimlerden faydalanılarak doğruluk tabloları oluşturulmuştur. F_1 için toplamların çarpımı ifadesinde kullanılan minterimler 1, 2 ve 3 olduğuna göre kod çözücünün 01, 10 ve 11 kodlu çıkışları ROM'un çıkışındaki sigortalı VEYA kapısına giriş olarak bağlanmalıdır. Kullanılmayan minterim ise sigortalı VEYA kapısının girişindeki sigorta attırılarak toplama katılması engellenir. F_2 için toplamların çarpımı ifadesinde kullanılan minterimler 0 ve 2 olduğuna göre kod çözücünün 00 ve 10 kodlu çıkışları ROM'un çıkışındaki sigortalı VEYA kapısına giriş olarak bağlanmalıdır. Kullanılmayan minterimler, 01 ve 11 ise sigortalı VEYA kapısının girişindeki sigortalar attırılarak toplama katılması engellenir.

Örnek: Girişine uygulanan 3 bitlik bir sayının karesinin ikili karşılığını çıkışta verebilecek bir kombinezonal devreyi ROM kullanarak tasarlayınız.

İlk olarak kombinezonal devre için doğruluk tablosu oluşturulup kullanılacak olan minterimler bulunmalıdır.

Girişler			Çıkışlar						Onlu
A_2	A_1	A_0	B_5	B_4	B_3	B_2	B_1	B_0	
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1	1
0	1	0	0	0	0	1	0	0	4
0	1	1	0	0	1	0	0	1	9
1	0	0	0	1	0	0	0	0	16
1	0	1	0	1	1	0	0	1	25
1	1	0	1	0	0	1	0	0	36
1	1	1	1	1	0	0	0	1	49

Doğruluk tablosu oluşturmak ROM ile yapılacak uygulamalarda yeterlidir. Bazı durumlarda doğruluk tablosundaki değerlere bağlı olarak ROM'un doğruluk tablosu daha az elemanlı bir tablo olabilir. 3 giriş ve 6 çıkış için istenilen özellikleri sağlayan ROM, 3 girişli olmalı ve 6 çıkışlı olmalıdır. Fakat doğruluk tablosuna bakıldığında çıkışlardan B_0, A_0 giriş değişkeni ile aynı değerleri almaktadır. Dolayısıyla çıkışa A_0 giriş değişkeni verilebilir. Aynı şekilde B_1 çıkışı ise sürekli 0 olduğu için bu çıkışta sürekli lojik 0 seviyesinde tutulması yeterlidir. O halde 3 girişli 4 çıkışlı 8x4'lük bir ROM yeterli olacaktır.



Blok diyagram

A_2	A_1	A_0	F_1	F_2	F_3	F_4
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	1
0	1	1	0	0	1	0
1	0	0	0	1	0	0
1	0	1	0	1	1	0
1	1	0	1	0	0	1
1	1	1	1	1	0	0

Doğruluk Tablosu

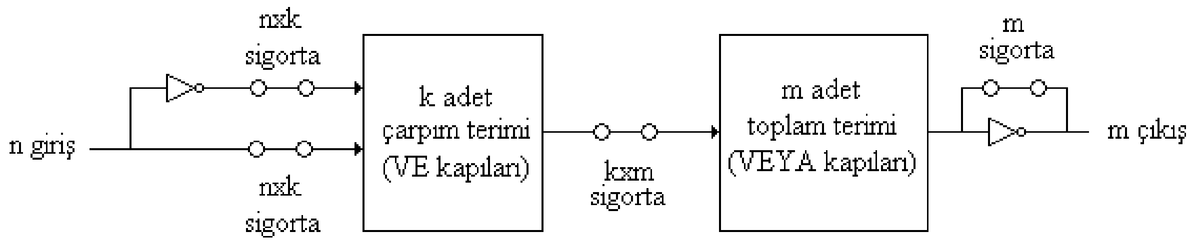
ROM Tipleri

ROM’lar iki şekilde programlanabilir. Birisi maske programlama olarak bilinen ve üretim esnasında üreticiye verilen doğruluk tablosu yardımı ile hazır olarak istenilen fonksiyonu gerçekleştiren bir maske üretilerek ROM bu şekilde programlanır. Bu durumda özel bir üretim olduğu için fiyat da ona göre yüksek olacaktır. Bu durum ancak oldukça fazla üretim gerçekleştiğinde faydalı bir yöntem olacaktır. Daha küçük miktarlar için PROM (Programlanabilir ROM) kullanılır. Attırılmak istenen sigortanın çıkışından bir miktar akım uygulanarak sigorta attırılır. Bu tip elemanlar ROM ve PROM bir kere programlandıktan sonra bir daha kesinlikle desen değişmez ve programlanamaz. Küçük bir hatada artık ROM veya PROM kullanılamaz hale gelir. Bu durumda üçüncü bir eleman olan EPROM (silinebilir-erasable PROM) olarak adlandırılan bir eleman kullanılır. Mor ötesi ışık altına tutulduklarında içerisine yüklenmiş olan desen silinebilir. Bu silme işlemi oldukça uzun süren ve silme işlemini garanti etmeyen bir yoldur. Bu nedenle bu eleman yerine EEPROM olarak adlandırılan ve elektriksel olarak silinebilir PROM’lar geliştirilmiştir.

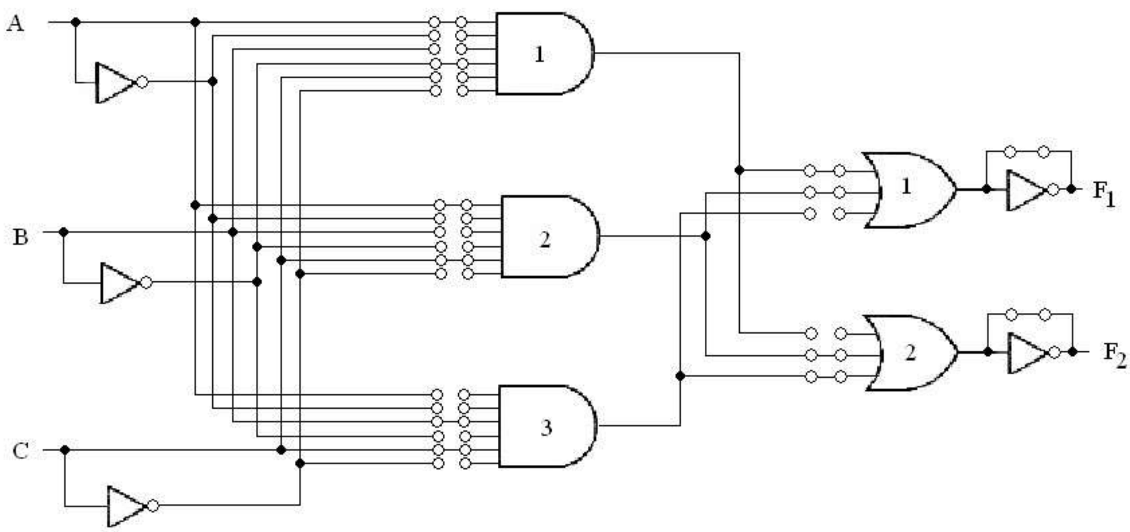
ROM’lar iki amaçla kullanılırlar. Herhangi bir kombinezonal devrenin tasarımında kullanılabildikleri gibi bilgi saklama elemanı olarak da kullanılırlar. Girişteki değişkenler adres değerini belirler ve o adresteki bilgi çıkışa aktarılır.

Programlanabilir Lojik Dizi (PLA)

Etkisiz koşulların olduğu bir kombinezonal devre ROM kullanılarak gerçekleştirilirse bu etkisiz koşullar hiçbir zaman gerçekleşmeyecek bir adres girişine karşılık düşer. Bu adreslerde yer alan sözcüklerin programlanmasına gerek yoktur ve orijinal değerlerinde bırakılabilirler. Etkisiz koşulların fazla olduğu kombinezonal devreleri gerçekleştirmek için ROM yerine programlanabilir lojik dizi veya PLA adı verilen ikinci tip LSI elemanları kullanmak daha ekonomik olacaktır. Bir PLA, ROM'a benzer fakat PLA'da değişkenlerin tümünün kodlanması sağlanmaz ve tüm minterimler üretilmez. Girişinde ROM'da olduğu gibi bir kod çözücü içermez, bunun yerine bir VE kapı dizisi içerir. Bu kapılardan her biri, giriş değişkenlerinden bir çarpım terimi üretilmesi sağlar. PLA içerisinde VE ve VEYA kapıları arasında bir sigorta bulunmaktadır. Çarpımların toplamı şeklindeki ifadelerin şeklindeki özel Boole fonksiyonları uygulandığında istenilen çıkışı sağlayan sigortalar olduğu gibi bırakılırken diğerleri attırılır.



Yukarıdaki şekilde bir PLA'ya ait blok diyagramı görülmektedir. n adet giriş için, k adet sigorta değişkenlerin kendisi, k adet sigorta değişkenin tümleyeni ve k adet VE giriş kapısı, m adet VEYA kapısı girişi için sigorta ve m adet de çıkış hattı ve çıkışın tümleyeni için sigorta bulunur. Aşağıdaki şekilde özel bir PLA yapısı görülmektedir.



Burada kullanılan giriş değişkenleri A, B ve C dir. Girişteki VE kapıları ile AB', AC ve BC çarpım terimleri oluşturulup F_1=AB'+AC, ve F_2=AC+BC çıkış fonksiyonları tanımlanmıştır. Kullanılmayan terimlere ait tüm sigortalar attırılmıştır.

PLA çok girişli ve çok çıkışlı kombinezonal devrelerin yerine düşünülmelidir. Basit bir örnek üzerinde PLA'nın programlanmasını düşünelim. Burada seçilen örnek PLA kullanımına uygun olmayıp sadece çalışmasını ve programlanmasını anlamak üzere seçilmiştir. ROM'da minterimlerin toplamı şeklinde bir yapı göz önüne alınırken PLA'da çarpımların toplamı şeklinde bir ifade göz önüne alınmaktadır. O halde 3 girişli herhangi bir kombinezonal devreye ait doğruluk tablosunun aşağıdaki gibi verildiğini varsayalım.

A	B	C	F_1	F_2
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	1	1
1	1	0	0	0
1	1	1	1	1

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

F_1=AB'+AC

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

F_2=AC+BC

	Çarpım Terimi	Girişler A	B	C	Çıkışlar F1	F2	
AB'	1	1	0	-	1	-	
AC	2	1	-	1	1	1	
BC	3	-	1	1	-	1	
					T	T	T/C

PLA program tablosu

Üç değişkenli bir kombinezonal devrenin PLA ile gerçekleştirilmesi için PLA program tablosunun oluşturulması gerekmektedir. Yukarıdaki program tablosundan üç terimin çarpılacağı anlaşılmaktadır. Dolayısıyla girişte 3 tane VE kapısına ihtiyaç vardır. Girişler sütununda ABC değişkenlerinin altında eğer 1 yazıyorsa değişkenin kendisi, 0 yazıyorsa değişkenin tümleyeni, - yazıyor ise değişkenin çarpım terimi içinde yer almayacağı anlaşılır. Bu çarpım terimlerinin hangisinin ya da hangilerinin çıkışa verileceği ise çıkışlar sütunu altında yer alan çıkış göstergeleri ile belirlenir. Eğer 1 yazıyorsa o terim çıkıştaki VEYA kapısının girişine verilecek, - yazıyor ise o çıkışa verilmeyecektir. O sütunun en altında yer alan T ve C değerleri ise şu anlama gelir:Eğer çıkışta fonksiyonun kendisi üretilecek ise T, tümleyeni üretilecekse C ile gösterilir. T, True anlamında C ise Complement anlamında kullanılmaktadır. Bu da çıkışta bulunan DEĞİL kapılarına paralel bağlı sigortaların attırılıp attırılmayacağını belirtmek için kullanılır. Eğer tasarımda çok sayıda çarpım terimi oluşuyor ise bu durumda değişkenin kendisi değil de tümleyeninin bulunması yoluna gidilir. Çünkü, PLA sınırlı sayıda çarpım terimi üzerinde işlem görür.

Örnek: Bir kombinezonal devre aşağıdaki gibi tanımlanmıştır. Devreyi PLA kullanarak gerçekleştiriniz.

$F_1(A,B,C)=\sum(3,5,6,7)$

$F_2(A,B,C)=\sum(0,2,4,7)$

Minimum değişkenli fonksiyonlar Karnaugh diyagramı yardımıyla bulunacak olursa;

$F_1=AC+AB+BC$ $F'_1=B'C'+A'C'+A'B'$

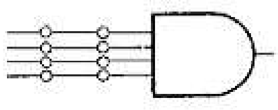
$F_2= B'C'+A'C'+ABC$ $F'_2=B'C+A'C+ABC'$

Eğer F_1 ve F_2 PLA ile gerçekleştirilmek istenirse, toplam 6 tane ve birbirinden farklı terim içerecektir. Bunun yerine F'_1 ile F_2 gerçekleştirilip çıkışta F'_1'nün tümleyeni alınırsa yine aynı işlem yerine getirilmiş olur. Fakat bu durumda gerçekleştirilecek çarpım işlemi ortak terimleri fazla olduğu için 4 olacaktır.

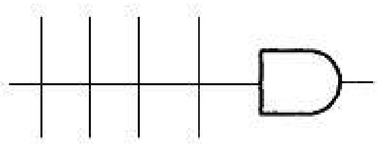
	Çarpım Terimi	Girişler			Çıkışlar		
		A	B	C	F_1	F_2	
B'C'	1	-	0	0	1	1	
A'C'	2	0	-	0	1	1	
A'B'	3	0	0	-	1	-	
ABC	4	1	1	1	-	1	
					C	T	T/C

Programlanabilir Dizi Lojiği (PAL)

Programlanabilir lojik devreler içerisinde sigortalar ile birbirine bağlı yüzlerce kapı devresi içerirler. Böyle devrelerin iç lojik çizimi için özlü bir gösterim olarak dizi lojiği'ni referans almak uygun olacaktır. Alışılagelmiş gösterimde bir kapıya gelen girişler sigortaları ayrı ayrı gösterilerek tüm girişler gösterilir. Fakat dizi lojiği gösteriminde yatay olarak gelen tek bir hat ve girişler ise dikey hatlar ile gösterilir. Yatay hat ile dikey hattın her kesiştiği nokta bir sigortayı tanımlar.

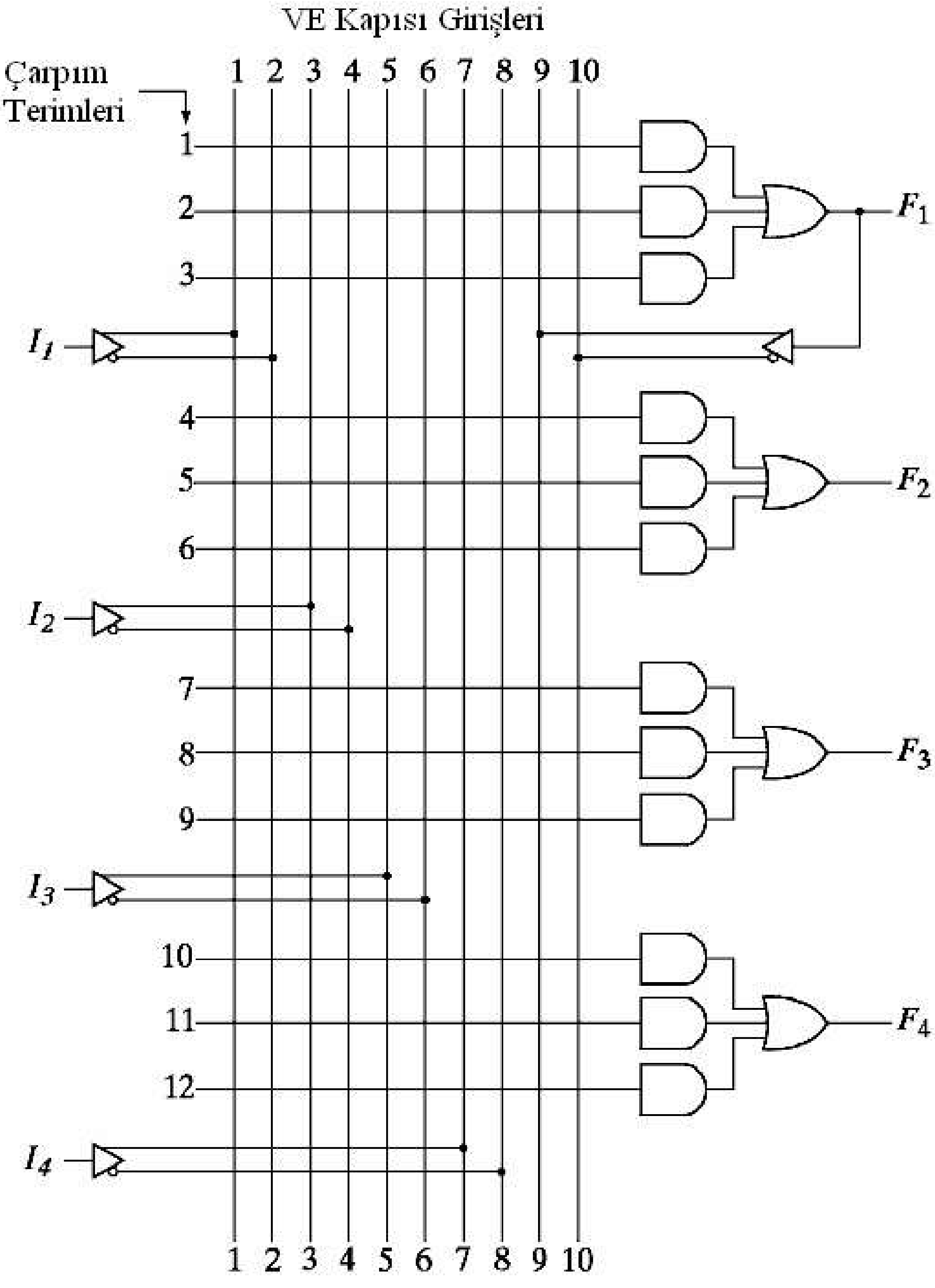


Alışılagelmiş gösterim



Dizi lojiği gösterimi

PAL'lar sabit VEYA kapıları ve programlanabilir VE dizileri ile programlanabilir bir cihazdır. Sadece VE kapıları programlanabilir olduklarından dolayı PAL'ın programlanması daha kolaydır fakat PLA'lar kadar esnek değillerdir. Aşağıda 4 girişli 4 çıkışlı tipik bir PAL dizi lojiği devresi görülmektedir. Her giriş bir tampon ve evirici devresine sahiptir. Bu iki kapı normal ve tümleyen çıkışları birleştirilmiş bir gösterim ile çizilmiştir. Devre toplam dört kattan oluşmaktadır. Dikey çizgiler VE kapılarının girişlerini bir başka deyişle tüm girişleri ve DEĞİL'lerini temsil etmektedir. Her bir VE-VEYA katı üç genişliklidir. Her bir dikey ve yatay çizginin kesiştiği noktada bir sigorta var olduğu kabul edilir eğer sigorta attırılmış ise + ile attırılmamış ise kesişim noktasına X işareti konur. Durumu korunan tüm sigortalar X ile gösterilir. Yatay çizgiler ise VE kapılarının çok girişli olduğu anlamına gelir. Çıkışlarının birisi Tampon-Evirici devresi ve sigortalar ile VE kapılarının girişlerine bağlanmıştır. Normalde PAL devreleri şekilde gösterilenden daha fazla giriş ve çıkışa sahiptirler. Tipik bir PAL devresi her biri 8 girişe ve 8 çıkışa sahip 8 genişlikli VE-VEYA bölgesinden oluşurlar.



PAL kullanılarak yapılan tasarımlarda Boole fonksiyonları öncelikle sadeleştirilmelidir. PLA'daki gibi bir çarpım terimi iki ya da daha fazla VEYA kapısı tarafından paylaşılamaz. Bu nedenle her bir terim ortak terimler göz önüne alınmadan kendi başlarına sadeleştirilmelidir. Yapıdaki VE kapılarının sayısı sabittir eğer çok sayıda terim içeriyorsa çıkışlar ayrı ayrı katlarda hesaplanarak toplanmalıdır.

Bir PAL uygulaması olarak aşağıdaki Boole fonksiyonunu göz önüne alalım.

$w(A, B, C, D)=\sum(2,12,13)$
$x(A, B, C, D)=\sum(7,8,9,10,11,12,13,14,15)$
$y(A, B, C, D)=\sum(0,2,3,4,5,6,7,8,10,11,15)$
$z(A, B, C, D)=\sum(1,2,8,12,13)$

Bu dört fonksiyonun basitleştirilmesi sonucu aşağıdaki ifadeler elde edilir.

w=ABC'+A'B'CD'

x=A+BCD

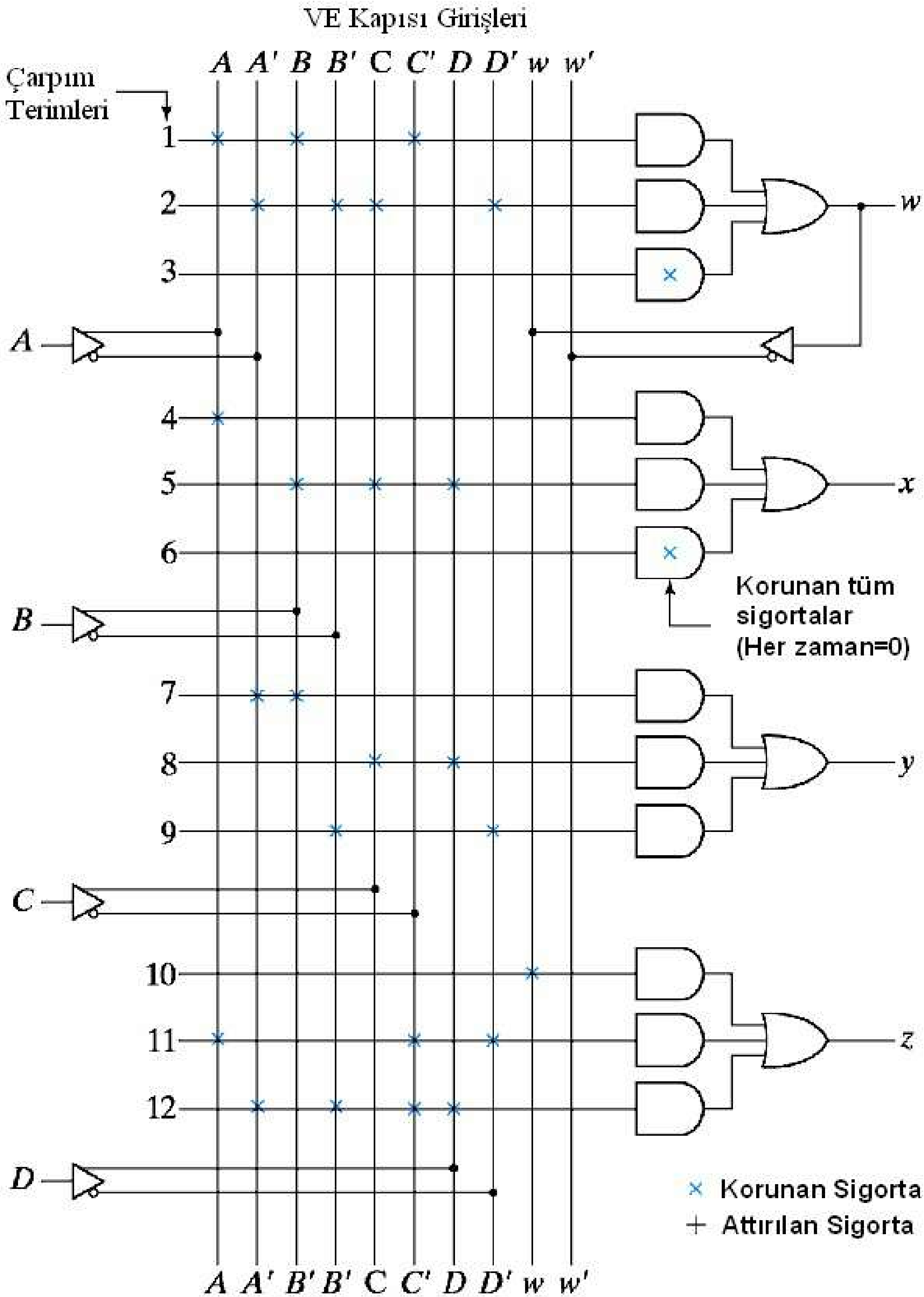
y=A'B+CD+B'D'

z= ABC'+A'B'CD'+AC'D'+A'B'C'D

=w+ AC'D'+A'B'C'D

z terimine dikkat edilecek olursa, w terimini de içerdiği görülmektedir. Bu nedenle bu terimleri tekrar bir çarpım kapısı girişine vermektense w çıkış fonksiyonunu PAL'ın çıkışlarını tümleyen ve normal çıkış veren buffer ucu kullanılırsa tekrar hesaplamaya gerek kalmaz. Aşağıda PAL program tablosu görülmektedir. Eğer bir VE kapsının girişine gelen tüm sigortalar korunur, attırılmazsa değişkenin kendisi ve tümleyeni çarpılacağı için AA' her zaman 0 olacağından kapı çıkışı hep 0 da kalır ve fonksiyonun değeri değişmez.

Çarpım Terimleri	**VE girişleri**					**Çıkışlar**
	A	**B**	**C**	**D**	**W**	
1	1	1	0	-	-	w=ABC'+A'B'CD'
2	0	0	1	0	-	
3	-	-	-	-	-	
4	1	-	-	-	-	x=A+BCD
5	-	1	1	1	-	
6	-	-	-	-	-	
7	0	1	-	-	-	y=A'B+CD+B'D'
8	-	-	1	1	-	
9	-	0	-	0	-	
10	-	-	-	-	1	z=w+ AC'D'+A'B'C'D
11	1	-	0	0	-	
12	0	0	0	1	-	


VE Kapısı Girişleri
A A' B B' C C' D D' w w'
Çarpım Terimleri
1
2
3
4
5
6
7
8
9
10
11
12
A
B
C
D
w
x
y
z
Korunan tüm sigortalar (Her zaman=0)
× Korunan Sigorta
+ Attırılan Sigorta
A A' B' B' C C' D D' w w'

Soru-1) Dört bitlik sayıların 2'ye tümleyenini bulan devreyi tasarlayınız.(Sadece çıkış fonksiyonlarını bulunuz devreyi çizmeyiniz)

Cevap-1)
İki tabanındaki sayıların ikiye sayıların ikiye tümleyen, 2^n-N eşitiliği ile bulunur. 0000 sayısı için ikiye tümleyen 2^4-0000=10000-0000=10000 beş bitlik çıkış olur diğerler için ise dört bitlik bir çıkış olur. Bir başka değişle tüm satırların toplamı 10000(16) olacaktır.

A	B	C	D	F5	F4	F3	F2	F1
0	0	0	0	1	0	0	0	0
0	0	0	1	0	1	1	1	1
0	0	1	0	0	1	1	1	0
0	0	1	1	0	1	1	0	1
0	1	0	0	0	1	1	0	0
0	1	0	1	0	1	0	1	1
0	1	1	0	0	1	0	1	0
0	1	1	1	0	1	0	0	1
1	0	0	0	0	1	0	0	0
1	0	0	1	0	0	1	1	1
1	0	1	0	0	0	1	1	0
1	0	1	1	0	0	1	0	1
1	1	0	0	0	0	1	0	0
1	1	0	1	0	0	0	1	1
1	1	1	0	0	0	0	1	0
1	1	1	1	0	0	0	0	1

F5=A'B'C'D'

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

F4=A'B+A'D+A'C+AB'C'D'

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

F3=BC'D'+B'D+B'C

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

F2=C'D+CD'=C⊕D

Soru-2) 4 bitlik sayıların GRAY kodunu bulan devrenin tasarımı gerçekleştirilecektir.

a) Doğruluk tablosunu elde ediniz.

b) Her bir çıkış için en sade ifadeyi bulunuz.

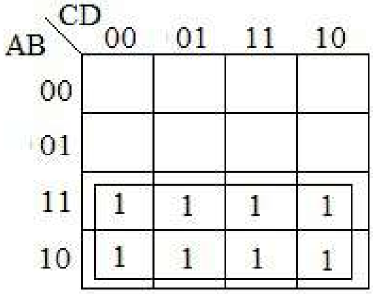
c) Devreyi VE-VEYA kapıları ile gerçekleştiriniz.

d) Devre VE/VEYA kapıları dışında daha az eleman kullanılarak gerçekleştirilebilirse gerçekleştirip bu devreyi çiziniz.

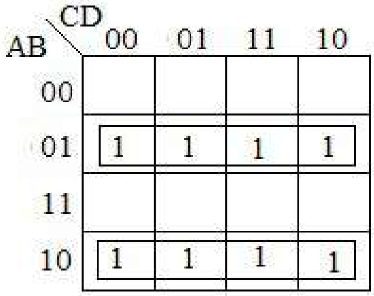
Cevap-2) a)

A	B	C	D	F1	F2	F3	F4
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

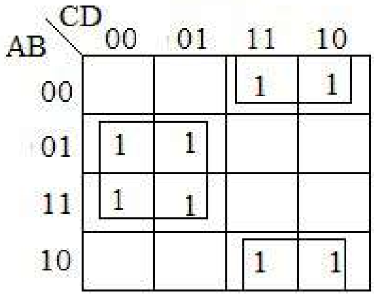
b)



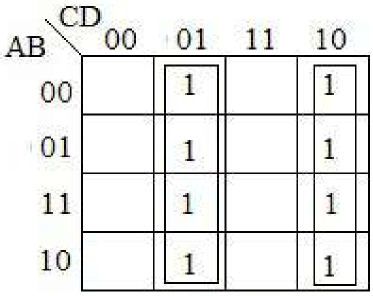
F1=A



F2=A'B+AB'



F3=BC'+B'C



F4=C'D+CD'

c) Devrenin VE/VEYA lojiği ile gerçekleştirilmesi

d) F2, F3 ve F4 çıkışları bir ÖZELVEYA kapısı ile gerçekleştirilebilir. Dolayısıyla devre toplam 3 kapı ile gerçekleştirilebilir.

Soru-3) 4 bitlik sayıların tek mi çift mi olduğunu bulan bir devre tasarlanacaktır

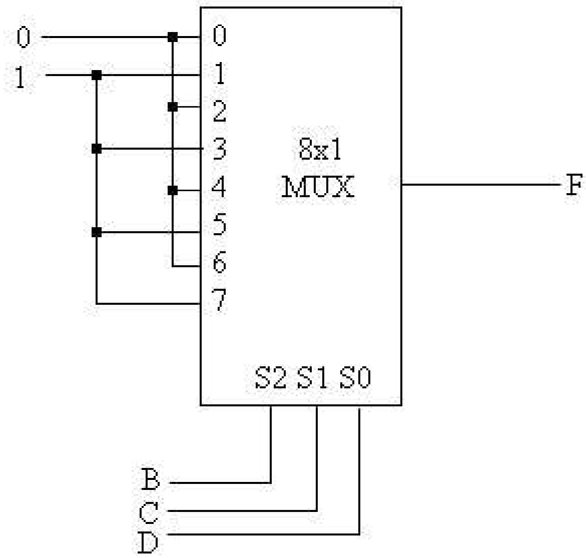
a) Devreyi uygun boyutta MUX ile tasarlayınız ve çiziniz.

b) Devreyi daha küçük bir boyutta bir MUX ile mesela 2x1'lik bir MUX ile tasarlayabilirmisiniz? Cevabınız "Evet" ise tasarımı gerçekleştiriniz ve çiziniz

Cevap-3) a)

A	**B**	**C**	**D**	**F**
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

	I0	I1	I2	I3	I4	I5	I6	I7
A'	0	1	2	3	4	5	6	7
A	8	9	10	11	12	13	14	15
	0	1	0	1	0	1	0	1



b) Bir sayının tek ya da çift olması ilk bitinin 1 ya da 0 olmasına bağlıdır. A, B ve C'nin herhangi bir anlamı yoktur. Sayının tek ya da çift olmasını D biti belirler. O halde sayı, D 1 ise tek 0 ise çifttir. **(10p)**