

YILDIZ TEKNİK ÜNİVERSİTESİ
Elektronik ve Haberleşme Mühendisliği Bölümü

LOJİK DEVRELER DERS NOTLARI

Prof. Dr. Ertuğrul ERİŞ
Devreler ve Sistemler Anabilimdalı

İSTANBUL, 2001

BÖLÜM 1 SAYISAL DİZGELER	1
1.1 Sayısal Dizgelerin (sistem) Temel Kavramları	1
1.2 Bilginin İletilmesi	4
1.3 Bilginin Saklanması	4
1.4 Bilginin İşlenmesi	5
1.5 Konumsal Sayı Dizgeleri	6
1.5.1 Taban dönüştürülmesi.....	7
1.5.2 Aritmetik	9
BÖLÜM 2 BOOLE CEBRİNE GİRİŞ	12
2.1 Matematiksel Genel Kavramlar.....	12
2.2 Boole Cebri.....	14
2.3 Sonlu Kümeler Arasında Tanımlanan Fonksiyonlar	18
2.4 Boole ve Boole Olmayan Fonksiyonlar	19
2.5 İki Değerli Boole Cebri ve Anahtar Fonksiyonları	25
2.6 Özet	27
BÖLÜM 3 SAYISAL DİZGELER VE BOOLE CEBRİ.....	28
3.1 Giriş.....	28
3.2 Önergeler Cebri ve İki Değerli Boole Cebri	28
3.3 Kapı Elemanlarından Oluşan Devreler ve İki Değerli Boole Cebri.....	29
3.4 Özet	38
BÖLÜM 4 KOMBİNEZONSAL DEVRE SENTEZİ.....	39
4.1 Giriş.....	39
4.2 Tablo Yöntemiyle Minimal Fonksiyonların Bulunması	41
4.2.1 Tablo yöntemi.....	41
4.2.2 Asal Bileşenlerin Bulunması.....	42
4.2.3 Asal bileşenler tablosu ve minimal fonksiyonların bulunması.....	44
4.2.4 Bütün İndirgenemeyen ve minimal fonksiyonların belirlenmesi.....	45
4.3 Karnaugh Diyagramlarıyla Fonksiyonların minimalleştirilmesi.....	50
4.3.1 Karnaugh diyagramları.....	50
4.3.2 Karnaugh diyagramlarında küpler ve bunlara karşı düşen çarpımlar(toplamlar) terimleri.....	52
4.3.3 Karnaugh diyagramlarıyla minimal fonksiyon(ların) bulunması	53
4.3.4 Kısmen belirlenmiş fonksiyonlar ve karnaugh diyagramlarıyla minimalleştirilmeleri	56

4.4	Kombinezonsal devre sentezi	58
4.4.1	Paralel toplayıcının sentezi	59
4.4.2	Kod dönüştürücü sentezi.....	61
BÖLÜM 5 KOMBİNEZONSAL MSI VE LSI ELEMANLARI		63
5.1	GİRİŞ	63
5.2	Paralel toplayıcı.....	63
5.3	Karşılaştırma devresi.....	68
5.4	Çözücüler (Decoders).....	69
5.5	Multiplexer (Çoğullayıcı) devreler	72
5.6	Read-Only Memory (ROM)	75
5.7	PLA (Programable Logic Array)	77
BÖLÜM 6 ARDIŞIL MANTIK DEVRELERİ		81
6.1	Giriş.....	81
6.2	Ardışıl Makinelerin Modellendirilmesi	82
6.3	Ardışıl Makinelerin Sınıflandırılması	85
6.4	Bellek Elemanları	86
6.5	Bellek elemanlarının tetiklenmesi	91
6.6	Tüm devre olarak bellek elemanları yardımcı girişleri	98
BÖLÜM 7 ARDIŞIL DEVRELERİN ANALİZİ		99
7.1	Saatli ardışıl devrelerin analizi	99
7.2	Örnek bir Moore makinesi analizi.....	99
7.2.1	Matematiksel model ve analiz	100
7.2.2	Durum tablosu ve analiz	101
7.2.3	Durum diyagramı ve analiz	102
7.2.4	Zaman diyagramı (Timing diagram).....	102
7.3	Örnek bir Mealy makinesi analizi	103
BÖLÜM 8 ARDIŞIL DEVRELERİN SENTEZİ		107
8.1	Sentez yönteminin genel adımları.....	107
8.2	Modulo 8 ileri sayıcı.....	110
8.3	Özel bir sayıcı devresi	112
8.4	Dizi detektörü	115

BÖLÜM 9 ARDIŞIL DEVRELERDE DURUM İNDİRGENMESİ VE DURUM KODLAMASI 118

9.1	Durum indirgemesi (state reduction).....	118
9.1.1	Eşleştirme tablosu ile durum indirgemesi yöntemi.....	120
9.1.2	Eşdeğer durum sınıfları ile durum indirgemesi yöntemi.....	122
9.2	Durum kodlaması (state assignment).....	124
9.2.1	Kod sayısı ve eşdeğer durum kodları	124
9.2.2	Durum kodlaması	127

BÖLÜM 10 YAZMAÇLAR, SAYICILAR, BELLEKLER.....130

10.1	Yazmaçlar.....	130
10.1.1	Paralel yüklemeli yazmaçlar.....	130
10.1.2	Ötelemeli yazmaçlar.....	133
10.2	Sayıcılar (Counters).....	136
10.2.1	İkili Asenkron Sayıcılar (Binary ripple counter)	136
10.2.2	İkili kodlanmış onlu asenkron sayıcı (BCD binary coded decimal ripple counter)	137
10.2.3	İkili senkron sayıcılar.....	139
10.2.4	Paralel yüklemeli ikili senkron sayıcılar.....	140
10.3	Sayıcılarının uygulaması olarak, zamanlama dizileri üreten devreler.....	142
10.3.1	Kelime-zamanı darbesinin üretilmesi	143
10.3.2	Sayısal sistemlerde zaman işaretleri	143
10.3.3	Johnson sayıcılar ve 8 li zaman işareti üreteçleri.....	145
10.4	Bellek birimi	146

EKLER 148

EK 1	Kaynakça.....	148
EK 2	Örnek Sınav Soruları.....	149

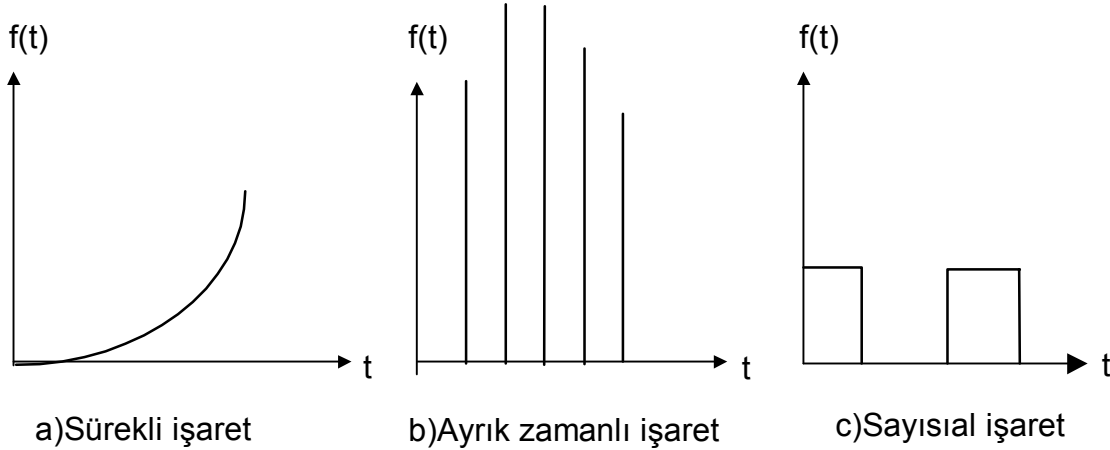
BÖLÜM 1 SAYISAL DİZGELER

1.1 Sayısal Dizgelerin (sistem) Temel Kavramları

Bilgi saklanır, iletilir veya işlenirken çeşitli fiziksel yapılarda görülür: harfler (yazaçlar), sayılar, elektriksel büyüklükler, ışık, duman, kimyasal yapılar gibi. Biz elektriksel yapıda (akım, gerilim) olan işaretlerle ilgileneceğiz. Bu işaretlerin zamana göre değişimi üç farklı biçimde görülebilir.

a. **Sürekli işaretler:** R reel sayılar kümesi olmak üzere $f:R \rightarrow R$ ye tanımlanan fonksiyonlardır. Her bir t anına belli bir reel sayı karşı düşer (Şekil.1.1.a).

b. **Ayrık zamanlı (discrete) işaretler:** N doğal sayılar kümesi olmak üzere $f:N \rightarrow R$ ye tanımlanan fonksiyonlardır. Yalnızca belli t ler için fonksiyon tanımlanmıştır. Sayısal süzgeçlerdeki örneklenmiş işaret, ayrık zamanlı işarete bir örnektir. (Şekil.1.1.b)



Şekil 1.1 Farklı biçimlerde görülen işaretler

c. **Sayısal (digital) işaretler:** S , N nin sonlu bir alt kümesi olmak koşuluyla $f:N \rightarrow S$ ye tanımlanan fonksiyonlardır. Bu fonksiyonların ayrık olan fonksiyonlardan farkı, değer kümesinin, R reel sayılar kümesi yerine, N doğal sayılar kümesinin sonlu bir alt kümesi olmasıdır (Şekil.1.1.c). S kümesindeki eleman sayısı (m), kullanılan mantığı belirler, m -li mantık gibi. Sayısal dizgelerde $S=(0,1)$ **ikili mantığı** yaygın bir biçimde kullanılmaktadır. Yarıiletken elemanların güvenilir çalışması olan iletimde ve kesimde çalışma 0,1 simgelerine karşı düşürülür. Bu dersin kapsamı içinde yalnızca ikili mantıkla ilgileneceğiz.

Günlük yaşantımızda bilgi alış verişi (saklama, iletme ve işleme) harfler (yazaçlar), noktalama işaretleri, rakamlar gibi simgeler aracılığıyla yapmaktayız. Bu farklı simgelerden oluşan kümeye **abece** (alfabe), abecenin her bir elemanına da **sayamak** (digit) diyeceğiz. Alışlagelen abecenin, sayısal dizgeye ilişkin sayısal işaretin değer kümesi olan, $S=(0,1)$ abecesine dönüştürülmesine **kodlama (düşümlleme)** denir.

Bilişim kuramında bilgi birimi **ikili (bit)** dir. Bir soruya ilişkin sorana göre aynı olasılıklı iki yanıt, bir bitlik bilgi taşıyor denir. Örneğin (evet, hayır); (var, yok); (1,0) abeceleri

birer bitlik bilgi taşırlar. Buna göre m sayamaklı bir abecenin her bir sayamağı ($\log_2 m$) bitlik bilgi taşır. Bir başka deyişle m sayamaklı abecenin her bir sayamağına, (0,1) li simgeler karşı düşürsek (kodlarsak), $k \geq \log_2 m$ bağıntısını sağlayan en küçük tam sayı olan k tane (0,1) simgesine gereksinim vardır. Örneğin m=4 sayamaklı (A,B,C,D) abecesinin her bir elemanı $\log_2 4 = 2$ bitlik bilgi taşır, yani A=00 B=01 C=10 D=11 biçiminde kodlanabilir ve her bir sayamağı (0,1) li iki tane simge karşı düşürülmüş olur. Bu da bit sayısına eşittir. Yani bu abece 2 bitlik bilgi taşır.

Pratikte kullanılan, harfler, rakamlar ve noktalama işaretlerinden oluşan alıştığımız abeceyi, ikili abaceye dönüştüren en yaygın **alphanumeric (abece ve sayılar)** kod, **ASCII (American Standard Code for Information Interchange)** kodudur. Alıştığımız abecedeki her bir sayamak 7 bitle temsil edilebilir. Aşağıdaki Tablo 1.1 de bu kod verilmiştir. Örneğin A ve a için kodlar

$b_7 b_6 b_5 b_4 b_3 b_2 b_1$
A 1 0 0 0 0 1
a 1 1 0 0 0 1 dır.

Tablo 1.1 ASCII kodu

		b_7	b_6	b_5	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b_4	b_3	b_2	b_1									
0	0	0	0	0	NUL	DLE	SP	0	@	P	,	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	\$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	,	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	A	LF	SUB	.	:	J	Z	j	z
1	0	1	1	B	VT	ESC	+	;	K	[	k	{
1	1	0	0	C	FF	FS	,	<	L	\	l	
1	1	0	1	D	CR	GS	-	=	M	]	m	}
1	1	1	0	E	SO	RS	.	>	N	^	n	~
1	1	1	1	F	SI	US	/	?	O	ş	o	DEL

ASCII koduna alternatif olarak kullanılan **EBCDIC (Extended Binary Coded Decimal Interchange Code)** kodu vardır, genellikle IBM bilgisayarlarda kullanılır.

Yalnızca nümerik olan kodlar da vardır. 10 lu abecedeki sayamakların ikili abecedeki karşılıklarını veren kodlar, genellikle **BCD (Binary Coded Decimal)** olarak anılırlar. $\log_2 10 = 3$ ile 4 arasında bir sayı olduğundan, bu kodlamada 4 bit'e gereksinim vardır. Bu durumda 10 kod kullanılacak diğer 6 kod ise atıl kalacaktır. Aşağıdaki Tablo 1.2 de BCD kodlar verilmiştir. Koyu bölmedeki rakamlar konumların ağırlıklarını

göstermektedir. Örneğin 7421 kodunda 8 rakamı $1.7+0.4+0.2+1.1$ olarak bulunur. Son sütunda ki kodlama 8421 koduna 3 eklenerek bulunur. 1. sütundaki kod yaygın bir biçimde kullanılır ve BCD kod dendiği zaman bu kod anlaşılır.

Tablo 1.2 Değişik BCD kodlar

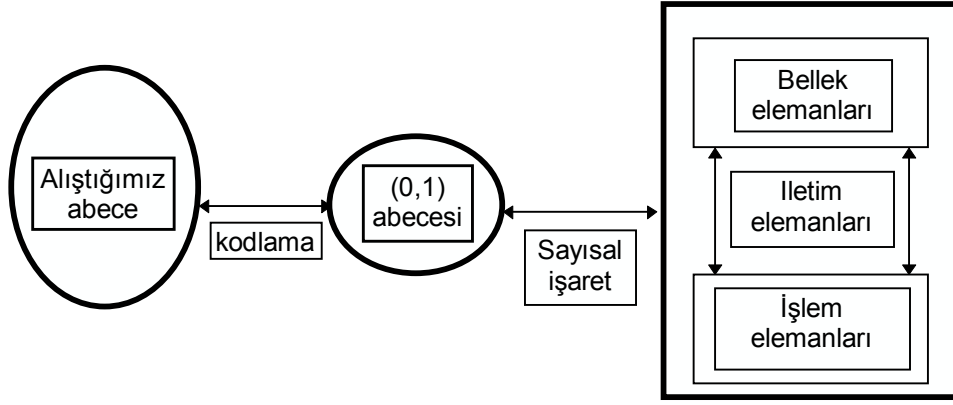
Onlu ayamak	8421 odu	5421 odu	7421 odu	Excess kodu
	8421	5421	7421	8421
0	0000	0000	0000	0011
1	0001	0001	0001	0100
2	0010	0010	0010	0101
3	0011	0011	0011	0110
4	0100	0100	0100	0111
5	0101	0101	0101	1000
6	0110	0110	0110	1001
7	0111	0111	1000	1010
8	1000	1011	1001	1011
9	1001	1100	1010	1100

Tablo 1.2 Gray kodu

0	000
1	001
2	011
3	010
4	110
5	111
6	101
7	100

Gray kodu bir başka sıkça kullanılan koddur. Bu koda dönüştürülecek abecedeki sıralı elemanların ikili kodlanmasında komşu sıralardaki kodlarda yalnızca bir bitte değişiklik vardır. Örneğin yukarıda Tablo 1.3 te verilen 3 bitli Gray kodunda 3 ve 4 gibi komşu elemanlara verilen kodlarda, (010, 110) yalnızca en soldaki bit değişmiş diğerleri değişmemiştir.

Şimdiye kadar, Şekil 1.2 de canlandırıldığı gibi, sayısal dizgenin girişine gelen (0,1) abeceli sayısal işaretin, alışlagelen abece'den kodlama ile nasıl üretildiğini gördük. Şimdide sayısal dizgenin iç yapısını, alt dizgelerini görelim. Sayısal dizge bilginin iletimine yarayan iletim elemanları, bilginin saklanmasına yarayan bellek elemanları ve bilginin işlenmesi, yeni bilgi üretilmesine yarayan işlem elemanlarından oluşur (Şekil.1.2). Bu elemanlar arasında eşgüdümü sağlayan komut işaretleri, neredeki bilginin ne kadar saklanacağı veya ne zaman neredeki bilginin nasıl işleneceğini belirler.



Şekil.1.2. Sayısal dizge ve iç yapısı.

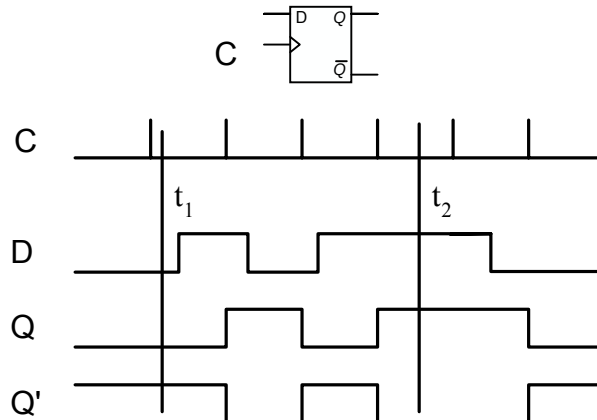
1.2 Bilginin İletilmesi

Bilginin iletilmesi iletim elemanlarıyla olur: İletken tel, elektromagnetik alan gibi. Tellerle yapılan iletimde kodlama sonucu elde edilen n bitlik bilginin her biti bir telle iletilebilir, buna **paralel iletim** denir. Ya da bilgiye ilişkin bitler bir tel üzerinde zaman içinde sıralı olarak iletilebilir, buna da **seri iletim** denir. Paralel iletimin daha hızlı olacağı açıktır, bedeli tel maliyetidir. Bu nedenle uzak mesafelerdeki iletimde, genellikle seri iletim; kısa mesafelerdeki iletimde ise paralel iletim seçilir.

1.3 Bilginin Saklanması

Bilgi, devingen (dinamik) bellek elemanlarıyla tekrarlanarak saklandığı gibi, durağan (statik) bellek elemanlarıyla da uzun süreler saklanabilir. Örneğin disketler, compact diskler (CD) , hard diskler gibi

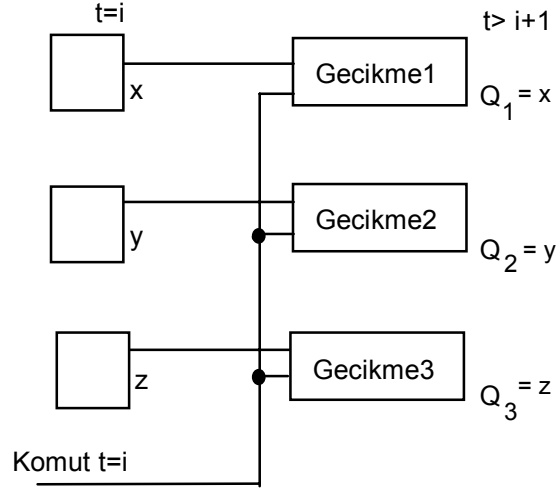
Devingen bellek elemanları, yeni bir komut gelene kadar bir bitlik bir bilgiyi saklayan elemanlardır. Çeşitli türleri vardır. Şekil.1.3 de bir bellek elemanına ilişkin giriş ve çıkışlar gösterilmiştir. C komut darbelerinin giriş ucudur, bu darbeler geldiğinde bellek elemanı D girişini kabul eder. Q çıkıştır. Q' , Q nun tümleyeni diye adlandırılır ve $Q=0$ iken $Q'=1$, $Q=1$ iken $Q'=0$ olur. Gecikme elemanı olarak bilinen bu eleman, komut işareti varken girişine gelen sayısal işareti çıkışına ötelenmiş olarak vermektedir. Girişle çıkış arasındaki ilişkiyi $Q(i+1)=D(i)$ biçiminde



Şekil.1.3 Gecikme bellek elemanı ve giriş çıkış ilişkisine bir örnek.

ifade edebiliriz. Bu da i anındaki girişin $(i+1)$ anında çıkışta görülmesidir. Şekil.1.3 de $t = t_1$ ve $t=t_2$ anlarında, aynı girişler için çıkışların farklı olduğu gözlemlenir. Çıkış, girişler cinsinden yegane olarak belirlenememektedir. Bu özellik bellek elemanlı devrelerde görülen genel bir özelliktir. Böyle elemanları içeren mantık devrelerine **ardışıl (sequential) devreler** denir.

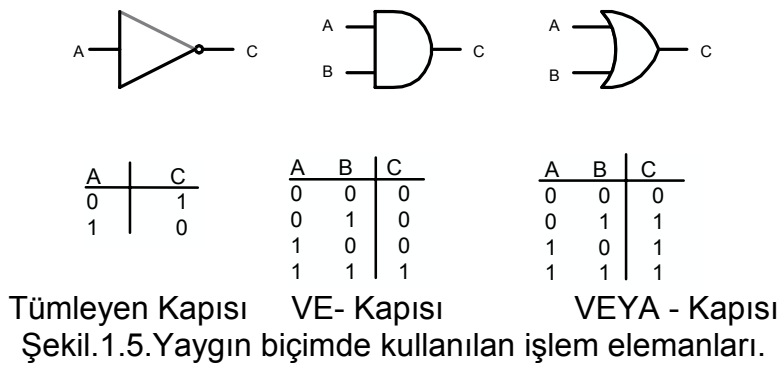
Şekil.1.4 te verilen gecikme elemanlarından oluşmuş mantık devresi, bilginin depolanmasına ilişkin bir örnek devredir. Başka elemanların çıkışları olan X Y Z uçlarındaki bilgi, komuta işaret geldiğinde gecikme elemanlarının çıkışlarında görülecektir. Komuta yeni bir işaret gelene kadar çıkış aynı kalacak, yani bilgi depolanmış olacaktır . Bu devreye yazmaç (register) devresi denir.



Şekil.1.4. Bir yazmaç devresi.

1.4 Bilginin İşlenmesi

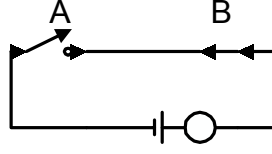
Bilginin işlenmesi işlem elemanlarıyla yapılır. Basit ikili işlem elemanları **kapılardır** (Gate). Belli başlı işlem elemanları ve giriş çıkış bağıntıları Şekil.1.5 de verilmiştir.



Şekil.1.5. Yaygın biçimde kullanılan işlem elemanları.

Tümlen kapısı çıkışında girişinin zıddını verir. VE (AND) kapısında A girişine, kapıyı yöneten işaret gözüyle bakılabilir. Şöyle ki $A=1$ iken $B=C$ yani kapı açık ve $A=0$ iken $C=0$ yani kapı kapalıdır. VEYA(OR) kapısı da girişlerden en az biri veya ikisi birlikte 1 olduğunda çıkış 1 olur, aksi halde 0 olur. Konuşma dilinde ve / veya' ya karşı düşer. Bu işlem elemanları yeni bilgi üretirler, yani çıkış bilgileri giriş bilgilerinden farklıdır ve çıkışlar girişler cinsinden yegane olarak belirlenir. Böyle elemanlardan oluşan mantık devrelerine **kombinezonsal (combinational) devreler** denir.

Kombinezonsal bir devre örneği olarak bir alarm devresi tasarlayalım. Görevli birinin kasayı açmasında bir zilin çalmasını, yabancı birinin aynı kasayı açmasında ise zilin çalmasını sağlayan basit bir anahtarlı devre Şekil.1.6 da verilmiştir.



Şekil.1.6. Kombinezonsal bir alarm devresi.

A ve B anahtarlarının kapalı olması mantık 1'e, açık olması mantık 0'a, zilin çalması mantık 1'e çalması da mantık 0'a karşı düşsün. A anahtarı görevlinin işe geldiğinde açtığı (0) ve gidişinde kapadığı (1) bir anahtar olsun. B anahtarı da birinin kasayı açmasında 1, aksi halde 0 olsun. Olası bütün durumlar: A=0 (görevli var), B=0 (kasa kapalı) ise Z=0 dır. zil çalmaz; A=0 (görevli var), B=1 (kasa açılmış) ise Z=0 dır, zil çalmaz -görevli varken yabancının gelmeyeceğini varsayalım- A=1 (görevli yok), B=0 (kasa kapalı) ise Z=0 dır, zil çalmaz; A=1 (görevli yok), B=1 (kasa açılmış) ise Z=1dir, zil çalar. Sözünü ettiğimiz bütün giriş kombinezonları ve çıkış Şekil.1.6 da ki tabloda verilmiştir. Görüldüğü gibi VE kapısının giriş çıkış ilişkisi ile bu tablo aynıdır. A ve B bilgileri zilin çalıp çalmasına komuta eden bilgiyi (Z), ya şekildeki devre ile veya bir VE kapısıyla üretirler.

1.5 Konumsal Sayı Dizgeleri

On tabanlı bilinen abecenin, ilgilendiğimiz iki tabanlı (0,1) abecesine dönüştürülmesi ve bu abecede aritmetik işlemlerin yapılışını sırasıyla inceleyelim. Sayı dizgelerindeki sayamaklara "rakam"da denir. Kullanıla gelen sayı dizgeleri, taban (radix) denen bir pozitif R tam sayısı ile, R sayamaklı (rakamlı) sıralı bir abecedan oluşan konumsal sayı dizgeleridir. Örneğin onlu sistemde R=10 olup sıralı abece $R=\{0,1,2,3,4,5,6,7,8,9\}$ kümesidir. Her konumun (basamağın) R ile belirlenen bir ağırlığı vardır. Simgesel olarak R tabanlı bir konumsal sayı dizgesinde

$$N = d_m \dots d_2 d_1 d_0 d_{-1} d_{-2} \dots d_m$$

ile ifade edilen N sayısının 10 tabanındaki karşılığı

$$N|_R = \sum d_i R^i = d_m R^m + \dots + d_2 R^2 + d_1 R^1 + d_0 R^0 + d_{-1} R^{-1} + d_{-2} R^{-2} + \dots d_m R^m \dots (1)$$

formülüyle belirlenir, $d_i \in R$ olup işlemler on tabanında yapılır. Görüldüğü gibi sayamakların bulundukları konumların tabana bağlı ağırlıkları vardır. Bu nedenle bu dizgelere konumsal sayı dizgeleri denilmektedir.

Örneğin R=10 $R=\{0,1,2,3,4,5,6,7,8,9\}$ için 10 tabanındaki 547 sayısı

$$547|_{10} = 5 \cdot 10^2 + 4 \cdot 10^1 + 7 \cdot 10^0 = 547|_{10} \text{ dur.}$$

Öte yandan R=2 $R=\{0,1\}$ için 2 tabanındaki 1101 sayısı

$$1101|_2 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 13|_{10} \text{ dur.}$$

Bir sayamağın bir ilerletilmesi: sıralı abecede bir sonraki sayamaktır. Örneğin 1 in bir ilerletilmiş 2, 9 un bir ilerletilmiş 0 dır.

Konumsal sayı dizgelerinde pozitif tam sayıların yazılması yöntemi

1) ilk tam sayı (0) dır

2) ikinci ve sonraki sayılar şöyle yazılır: En küçük ağırlıklı basamaktaki sayı bir ilerletilir. Eğer ilerletme sonucu yeni sayı (0) oluyorsa bu basamağa (konuma) (0) yazılır ve bir soldaki basamaktaki sayı bir ilerletilir. Böyle devam edilir. Tam sayıların bu biçimde yazılması saymaya karşı düşer.

1.5.1 Taban dönüştürülmesi

Yaygın biçimde kullanılan tabanlar onlu, ikili, sekizli (octal) ve 16 tılı (hexagonal) tabanlardır.

Herhangi bir tabandaki bir sayının on-tabanına dönüştürülmesi (1) formülüyle yapılır. On tabanında verilen herhangi bir sayıyı R tabanındaki sayıya dönüştürmek için tutulan yol ise; verilen sayıyı R ye (on-tabanında) bölüp kalanı tutmak, çıkan bölümü tekrar R ye bölmek kalanı tutmak ve bu işleme bölüm sıfır oluncaya kadar ($N_{i+1} = 0$) devam etmektir. Kalanlar geriye doğru ard arda yazıldığında, R tabanındaki sayı elde edilir. Çünkü yukarıdaki (1) ifadesindeki d_i ler burada kalanlara karşı düşmektedir.

Tanıt:

N_{10} : 10 tabanındaki sayı; R: istenen taban (bölen); N_i :Bölüm; d: kalan olsun. İşlemler 10 tabanında yapılmaktadır.

$$\frac{N_{10}}{R} = N_1 + \frac{d_0}{R}$$

$$\frac{N_{10}}{R^2} = \frac{N_1}{R} + \frac{d_0}{R^2} = N_2 + \frac{d_1}{R} + \frac{d_0}{R^2}$$

.

.

$$\frac{N_{10}}{R^i} = N_i + \frac{d_{i-1}}{R} + \frac{d_{i-2}}{R^2} + \dots + \frac{d_0}{R^i}$$

$$\frac{N_{10}}{R^{i+1}} = \frac{N_i}{R} + \frac{d_{i-1}}{R^2} + \frac{d_{i-2}}{R^3} + \dots + \frac{d_0}{R^i} = N_{i+1} (=0) + \frac{d_i}{R} + \frac{d_{i-1}}{R^2} + \frac{d_{i-2}}{R^3} + \dots + \frac{d_0}{R^{i+1}}$$

$$N_{10} = \left(N_i = d_i \right) R^i + d_{i-1} R^{i-1} + d_{i-2} R^{i-2} + \dots + d_0 R^0$$

Bu açınımında d_i en çok ağırlıklı bit **MSB (Most Significant Bit)** d_0 ise en az ağırlıklı bit **LSB (Least Significant Bit)** olarak adlandırılırlar. Aşağıda 10 tabanındaki 3545 sayısının 6 tabanında karşılığının bulunması bir örnek olarak verilmiştir.

$$\begin{array}{l}
3545:6 = 590 + \frac{5}{6} \\
590:6 = 98 + \frac{2}{6} \\
98:6 = 16 + \frac{2}{6} \\
16:6 = 2 + \frac{4}{6} \\
2:6 = 0 + \frac{2}{6}
\end{array}
\Rightarrow 3545|_{10} = 24225|_6$$

On tabanında ondalıklı bir sayının herhangi bir R tabanında karşılığının bulunması: Sayı R ile çarpılır (10 tabanında), tam kısmı tutulur ; bulunan yeni ondalıklı sayı tekrar R ile çarpılır ve tam kısmı tutulur. Böyle istenildiği kadar devam edilir. Tutulan rakamlar ondalıklı olarak ard arda yazılarak istenilen R tabanındaki ondalıklı sayı bulunur. Tanıt bir önceki tanıtı benzer biçimde yapılır. Aşağıda ondalıklı on tabanındaki 0.8906 sayısının 6 tabanına karşılığının bulunması bir örnek olarak verilmiştir.

$$\begin{array}{l}
0.8906 \times 6 = \underline{5}.3436 \\
0.3436 \times 6 = \underline{2}.0616 \\
0.0616 \times 6 = \underline{0}.3696 \\
0.3696 \times 6 = \underline{2}.2176
\end{array}
\Rightarrow 0.8906|_{10} = 0.5202|_6$$

Tam ve ondalıklı kısmı olan bir sayı için dönüşüm tam ve ondalıklı kısmı ayrı ayrı düşünülerek iki adımda yapılır.

10 tabanında 16 ya kadar sayıların diğer bazı tabanlardaki karşılıkları aşağıda Tablo 1.3 te verilmiştir. 16 tabanında gerekli olan ilave sayamalar yazaç(harf) olarak kullanılır.

Tablo 1.3 Yaygın tabanlar arası dönüşümler

10 tabanı	2 tabanı	8 tabanı	16 tabanı
0	0000	00	0
1	0001	01	1
2	0010	02	2
3	0011	03	3
4	0100	04	4
5	0101	05	5
6	0110	06	6
7	0111	07	7
8	1000	10	8
9	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

Görüldüğü gibi taban dönüştürürken, her hangi bir tabandan 10 tabanına dönüşü (1) ifadesini kullanarak hemen yapabiliriz. 10 tabanından herhangi bir tabana dönüşümde ise bölme işlemleri yapmamız ve kalanları saptamamız gerekir. Herhangi bir tabandan herhangi bir başka tabana dönüşümü ise önce 10 tabanına, sonra varmak istediğimiz tabana, dönüştürme işlemleri yapmak gerekmektedir. Örneğin 7 tabanından 11 tabanına dönüşüm yapmak için önce 7 tabanındaki sayıyı 10 tabanına, sonrada bulunan 10 tabındaki sayıyı 11 tabanına dönüştürmek gerekir.

2 tabanından 8 ve 16 tabanlarına geçişler için kısa yollar verilebilir. İki tabanındaki sayı LSB ten başlayarak üçer üçer gruplandırılır ve her üçlü grubun 8 tabanındaki karşılığı yazılırsa 8 tabanındaki sayı bulunur. Ters taban dönüşümünde 8 tabanındaki her bir sayıya karşı gelen üç bitlik iki tabanındaki sayılar yan yana yazılarak bulunur. Tanıtını düşününüz. Örneğin

$$101\ 001\ 111\ 010|_2 = 5172|_8 \text{ dir.}$$

Benzer biçimde 16 tabanına geçişte de, dörtlü gruplar yapmak gerekir, örneğin

$$1010.0111\ 1010|_2 = A7A|_{16} \text{ dır.}$$

1.5.2 Aritmetik

Herhangi bir tabanlı konumsal sayı dizgesinde aritmetik işlemler, onlu sistemdekine benzer biçimde yapılır.

Toplama: P ve Q tam sayılarının toplamı (P+Q): P tam sayısının, en küçük basamağındaki rakamından başlayarak -bulunulan tabanda- Q defa ilerletilmesidir. Buda sayma işlemidir. Ancak ilgili tabana ilişkin toplama tablosu kullanılarak, bu işlem daha basit yapılabilir. İki ve sekiz tabanlı sayı dizgeleri için toplama tablosu Tablo.1.4 de verilmiştir.

Tablo.1.4. İki ve sekiz tabanlı için toplama tabloları.

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1		2	3	4	5	6	7	10
2			4	5	6	7	10	11
3				6	7	10	11	12
4					10	11	12	13
5						12	13	14
6							14	15
7								16

+	0	1
0	00	01
1	01	10

elde

Tablolarda görüldüğü gibi toplam iki basamaklı bir sayı olabilir. Soldaki basamaktaki sayıya (rakama) "elde" denir. Herhangi bir tabanda iki sayı toplanırken, önce en küçük ağırlıklı basamaktaki rakamların toplamından başlanır. Eğer bu rakamların toplamı, tabloda tek basamaklı ise, bu rakam toplamda ki aynı ağırlıklı basamak yerine konur. Elde nin 0 olduğu düşünölmüştür. Eğer en küçük ağırlıklı basamaktaki rakamlar

toplamı, tabloda iki basamaklı ise, birinci basamaktaki rakam toplamda ki en küçük ağırlıklı basamağa yazılır, ikinci basamaktaki rakam ise "elde" olarak bir soldaki basamaktaki rakamlar toplamına eklenir.

Bu işleme sola doğru devam edilir. Sonuç toplam, toplanan sayılardan bir basamak daha sola kayabilir. İki örnek aşağıda verilmiştir.

$$\begin{array}{r} 1101,01|_2 \\ +1011,11|_2 \\ \hline 11001,00|_2 \end{array}$$

$$\begin{array}{r} 13,25|_{10} \\ +11,75|_{10} \\ \hline 25,00|_{10} \end{array}$$

$$\begin{array}{r} 772|_8 \\ +647|_8 \\ \hline 1641|_8 \end{array}$$

$$\begin{array}{r} 506|_{10} \\ +423|_{10} \\ \hline 929|_{10} \end{array}$$

Çıkarma: Çıkarma işlemi çıkarma tablosundan yararlanarak yapılır. Tablo.1.5 de iki tabanı için verilen tabloda iki rakamın farkı olan sayılar iki basamaklıdır. Sağdaki farkı, soldaki ise çıkarılan sayıda bir önceki basamaktan alınan ödünç gösterir. Böylece ödünç alınan basamaktaki rakam bir küçültülmüş olur.

Tablo.1.5. İki tabanı için çıkarma tablosu

a \ b	0	1
0	00	01
1	11	00 → a-b farkı

↓
ödünç

Bir örnek aşağıda verilmiştir.

$$\begin{array}{r} 11010|_2 \\ -111|_2 \\ \hline 10011|_2 \end{array}$$

$$\begin{array}{r} 772|_8 \\ -647|_8 \\ \hline 123|_8 \end{array}$$

Çarpma: Bu işlem tekrarlanmış toplama işlemidir. Çarpma tablosundan yararlanarak daha çabuk sonuca varılır.

Tablo.1.6 da iki ve sekiz tabanı için çarpma tabloları verilmiştir. İki rakamın çarpımında bulunan, iki basamaklı sayının -tek basamaklı ise soldaki basamağında (0) olduğu düşünülür - birinci basamağındaki rakam çarpımda ilgili basamağa yazılır, ikinci basamaktaki rakam ise çarpımda bir soldaki basamağa elde olarak aktarılıp, eklenir. Bu işlemlere on tabandakine benzer biçimde devam edilir.

Tablo.1.6. İki ve sekiz tabanları için çarpım tabloları

x \ y	0	1
0	0	0
1	0	1

x \ y	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1		1	2	3	4	5	6	7
2			4	6	10	12	14	16
3				11	14	17	22	25
4					20	24	30	34
5						31	36	43
6							44	52
7								61

Çarpmaya ilişkin bir örnek aşağıda verilmiştir.

$$\begin{array}{r}
 1101|_2 \\
 \times 1011|_2 \\
 \hline
 1101 \\
 1101 \\
 0000 \\
 + 1101 \\
 \hline
 10001111
 \end{array}
 \qquad
 \begin{array}{r}
 772|_8 \\
 \times 647|_8 \\
 \hline
 6726 \\
 3750 \\
 + 5734 \\
 \hline
 642026|_8
 \end{array}$$

Bölme: Herhangi bir tabandaki bölme işlemi, on tabanındakine benzer biçimde yukarıda anlatılan çarpma ve çıkarma işlemleriyle yapılır. İki tabanı için iki örnek aşağıda verilmiştir.

$$\begin{array}{r}
 1101|1011 \\
 -1011 \quad 1,001 \\
 \hline
 0010000 \\
 -1011 \\
 \hline
 0101
 \end{array}
 \qquad
 \begin{array}{r}
 1110|10 \\
 -10 \quad 111 \\
 \hline
 11 \\
 -10 \\
 \hline
 10 \\
 -10 \\
 \hline
 ..
 \end{array}
 \qquad
 14: 2 = 7$$

Sorular

1. Bir konumsal sayı dizgesindeki sayamak sayısının fazla olmasının olumlu ve olumsuz birer yönünü tartışınız.
2. 1.5.1 bölümünde anlatılan 8 ve 16 tabanları ile iki tabanı arasındaki geçiş kolaylığının neden öyle olduğunu kanıtlayınız.
3. Bilgi iletişim sistemleri ile teknoloji arasındaki ilişkiyi tartışınız.
4. İkili mantık yerine daha üst değerde bir mantıkla çalışan sayısal sistemde bilgi iletişimde ne yarar sağlanabilir açıklayınız.
5. Analog ve sayısal işaretlerin saklanabildiği ortamları tartışınız.

BÖLÜM 2 BOOLE CEBRİNE GİRİŞ

Boole cebrinin aksiyomatik tanımına geçmeden önce kullanacağımız matematiksel kavramların tanımlarını anımsayalım.

2.1 Matematiksel Genel Kavramlar

Tanım 2.1. Kartezyen çarpım: A ve B kümelerinin kartezyen çarpımı $A \times B$; $x \in A$, $y \in B$ olmak üzere bütün (x, y) sıralı çiftlerinden oluşan kümedir.

Örneğin $A = \{a_1, a_2\}$, $B = \{b_1, b_2, b_3\}$ olmak üzere $A \times B = \{(a_1, b_1) (a_1, b_2) (a_1, b_3) (a_2, b_1) (a_2, b_2) (a_2, b_3)\}$ dür.

Tanım 2.2. Bağntı (Relation): A dan B ye bağntı $\langle A, B, R \rangle$ üçlüsüdür. Öyle ki A tanım kümesi (domen), B değer kümesi, R bağntı olmak üzere $R \subseteq A \times B$ dir. Hemen görülüyor ki R bir sıralı çiftler kümesidir. $R : A \rightarrow B$ ile gösterilir.

Örneğin A ve B yukarıdaki örnekte verilen kümeler olmak üzere, $R = \{(a_1, b_2) (a_2, b_3)\}$ kümesi bir bağntıdır.

Tanım 2.3. Tamamen belirlenmiş bağntı (Completely specified relation) : R bağntısında, tanım kümesindeki elemanların tamamı görünüyorsa bu bağntıya tamamen belirlenmiş denir.

Yukarıdaki örnekte verilen R bağntısı tamamen belirlenmiştir.

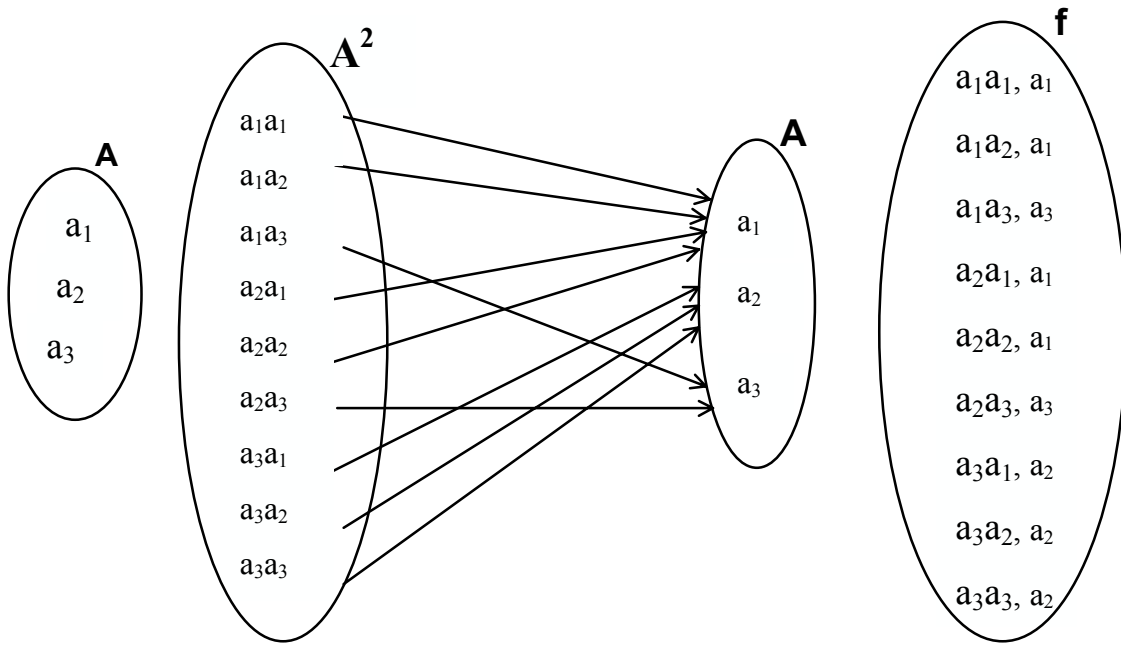
Tanım 2.4. Fonksiyonel bağntı (functional relation): R bağntısı tanım kümesindeki bir elemana değer kümesinde bir ve yalnız bir eleman karşı düşüyorsa bu bağntıya fonksiyoneldir denir.

$A = \{a_1, a_2, a_3\}$ ve $B = \{b_1, b_2\}$ kümeleri için $R = \{(a_1, b_1) (a_1, b_2) (a_3, b_2)\}$ bağntısı fonksiyonel değil, fakat $R = \{(a_1, b_2) (a_2, b_3) (a_3, b_2)\}$ bağntısı fonksiyoneldir.

Tanım 2.5. Fonksiyon: Tamamen belirlenmiş fonksiyonel bağntıya fonksiyon denir. $F : A \rightarrow B$ ile gösterilir. Yukarıda verilen ikinci bağntı bir fonksiyondur.

Tanım 2.6. n-lik işlem (n-ary operation) : A^n , A nın kendi üzerinden n defa kartezyen çarpımı olan küme olmak üzere $A^n \rightarrow A$ ya olan fonksiyona n-lik işlem denir. Yada n-değişkenli fonksiyon denir.

Şekil.2.1 de verilen diyagramla gösterilen fonksiyon bir 2-li işlemdir.



Şekil.2.1. 2-li işleme (iki değişkenli bir fonksiyona) örnek.

Tanım 2.7. **Eşdeğerlik bağıntısı (equivalence relation):** $R: A \rightarrow A$ ya bir bağıntı aşağıdaki özelliklerin üçünü birden sağlıyorsa bu bağıntıya eşdeğerlik bağıntısı denir, ve " $=$ " simgesi ile gösterilir.

1.Yansıma (refleksive) özelliği:

$\forall a \in A$ için $(a,a) \in R$ dir.

2.Simetri özelliği:

$(a,b) \in R$ ise $(b,a) \in R$ dir.

3.Geçişlilik (transitive) özellik:

$(a,b) \in R$ ve $(b,c) \in R$ ise $(a,c) \in R$ dir.

Böyle bir R bağıntısının özellikleri $a = a$; $a = b$ ise $b = a$; $a = b$, $b = c$ ise $a = c$ biçiminde de ifade edilebilir.

Örneğin $A = \{a_1, a_2, a_3\}$ kümesi için $R = \{(a_1, a_1) (a_2, a_2) (a_3, a_3) (a_1, a_2) (a_2, a_1)\}$ bir eşdeğerlik bağıntısıdır.

Tanım 2.8. **Sıra bağıntısı (order relation):** $A \rightarrow A$ ya bir bağıntı yalnızca yansıma ve geçişlilik özelliklerini sağlıyor ve $(a,b) \in R$, $(b,a) \in R$ için $a = b$ ise bu bağıntıya sıra bağıntısı denir ve " $<$ " simgesi ile gösterilir.

Örneğin $A = \{a_1, a_2, a_3\}$ kümesi için $R = \{(a_1, a_1) (a_2, a_2) (a_3, a_3) (a_1, a_2) (a_2, a_3) (a_1, a_3)\}$ bir sıra bağıntısıdır.

2.2 Boole Cebri

Tanım 2.9. **Boole Cebri**: Boole cebri $M = (S, R, A)$ üçlüsü ile verilen bir matematiksel sistemdir. Öyle ki

S: Sonlu sayıda elemandan oluşan bir kümedir.

R: 'Toplama' ve 'çarpma' denen ve tanımlanan iki 2-li işlem ile 'tümleyen' denen ve tanımlanan bir 1-li işlem den oluşan kümedir.

A: S nin elemanlarıyla R deki işlemlere ilişkin aksiyomlar kümesidir.

Toplama, çarpma ve tümleyen işlemleri sırasıyla $\vee$, $\wedge$, ' simgeleriyle gösterilir. İşlem tanımı uyarınca bu matematiksel sistem kapalıdır.

Boole cebri aksiyomları

A.1 Yer değiştirme (comutative) aksiyomu:

- $a, b \in S$ için 1. $a \vee b = b \vee a$
2. $a \wedge b = b \wedge a$ dır.

A.2. Dağılma (distributive) aksiyomu

- $a, b, c \in S$ için 1. $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$
2. $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$ dir.

A.3. Birim elemanları aksiyomu

$\forall a \in S$ için geçerli olan öyle bir 0 elemanı vardır ki a ile toplandığında a nın kendisini verir. Yine $\forall a \in S$ için öyle bir 1 elemanı vardır ki a ile çarpıldığında a nın kendisini verir.

- Yani 1. $a \vee 0 = 0 \vee a = a$
2. $a \wedge 1 = 1 \wedge a = a$ dır.

A.4. Tümleyen elemanı aksiyomu

$\forall a \in S$ için öyle bir tümleyen denen a' elemanı vardır ki

1. $a \vee a' = a' \vee a = 1$
2. $a \wedge a' = a' \wedge a = 0$ dır.

Bundan sonra $\vee$, $\wedge$ simgeleri yerine sırasıyla alışagelen $(+)$, $(.)$ simgelerini kullanacağız.

Teorem 2.1. Boole cebri aksiyomlarından çıkarılan bir teorem veya cebirsel eşitlikten $(+)$ ile $(.)$ nın ve 1 ile 0 ın yer değiştirmeleriyle, yani $(+)$ görülen yere $(.)$, $(.)$ görülen yere $(+)$ ve 1 görülen yere 0, 0 görülen yere 1 koyarak, geçerli olan ve dual denen ikinci bir teorem veya cebirsel eşitlik bulunur.

Tanıt: Her aksiyomun 1. ve 2. adımlarından birinden diğerine yukarıda sözü edilen yer değiştirmelerle geçilir. Bunlara dual aksiyomlar da denir. Öyleyse var olan teorem veya cebirsel eşitlik dual aksiyomlar kullanılarak tanıtlanır.

Tanım 2.10. **Sabit (değişmez)**: Boole cebirindeki S kümesinin her bir elemanı bir sabittir.

Tanım 2.11. **Değişken**: S kümesindeki herhangi bir elemanı simgeleyebilen bir simgedir.

Tanım 2.12. **Üreysel (generic) değişken:** Aksiyoamlarda kullanılan değişkenler olup, hem herhangi bir eleman (değişmez veya değişken) hem de cebirsel ifade olabilirler.

Teorem 2.2. Tümleyen elemanı tektir.

Tanıt: Varsayalım ki a ya ilişkin farklı iki tümleyen elemanı a'_1 ve a'_2 olsun. Yani $a+a'_1=1$, $a.a'_1=0$ ve $a+a'_2=1$, $a.a'_2=0$ olsun.

$$\begin{aligned}
 a'_1 &= 1.a'_1 && \text{A.3.2} \\
 &= (a+a'_2).a'_1 && \text{A.4.1} \\
 &= a'_1.(a+a'_2) && \text{A.1.2} \\
 &= a'_1.a+a'_1.a'_2 && \text{A.2.2} \\
 &= 0+a'_1.a'_2 && \text{A.4.2} \\
 &= a.a'_2+a'_1.a'_2 && \text{varsayımımızdan} \\
 &= a'_2.(a+a'_1) && \text{A.1.2 ve A.2.1} \\
 &= a'_2.1 && \text{A.4.1} \\
 &= a'_2 && \text{A.3.2}
 \end{aligned}$$

a' 'nın tümleyenini farklı iki a_1 ve a_2 olamaz, çünkü eşittirler. Yani a' tektir. Tanıt olmayana ergi yöntemiyle yapılmıştır.

Teorem 2.3.

1. $a+1 = 1$
2. $a.0 = 0$ dır.

Tanıt:

$$\begin{aligned}
 a+1 &= 1.(a+1) && \text{A.3.2} \\
 &= (a+a').(a+1) && \text{A.4.1} \\
 &= a+a'.1 && \text{A.2.2} \\
 &= a+a' && \text{A.3.2} \\
 &= 1 && \text{A.4.1}
 \end{aligned}$$

2.eşitlik düale teoremince tanıtlanır.

Teorem 2.4. Birim elemanlarının tümleyenleri $0'=1$ ve $1'=0$ dır.

Tanıt: A.4.1 aksiomunda a elemanını 0 olarak alırsak $0+0'=1$ dır. A.3.1 gereğince sol taraf $0'$ dür, o halde $0'=1$ dir.

$1'=0$ olduğu dual aksiyoamlardan yararlanarak tanıtlanır.

Teorem 2.5. Eşdeğer kuvvet (idempotent) teoremi $a+a = a$ ve $a.a = a$ dır.

Tanıt:

$$\begin{aligned}
 a+a &= 1.(a+a) && \text{A.3.2} \\
 &= (a+a').(a+a) && \text{A.4.1} \\
 &= a+a'.a && \text{A.2.2} \\
 &= a+0 && \text{A.4.2} \\
 &= a && \text{A.3.1}
 \end{aligned}$$

dual eşitlik benzer biçimde tanıtlanır.

Teorem 2.6. Herhangi bir a elemanının tümleyeninin tümleyenini, kendisine eşittir. Yani $(a')'=a$ dır.

Tanıt:

$$\begin{aligned}(a')' &= (a')'.1 && \text{A.3.2} \\ &= (a')' (a+a') && \text{A.4.1} \\ &= (a')'.a+(a')'.a' && \text{A.2.1} \\ &= (a')'.a+0 && \text{A. 4.2} \\ &= (a')'.a+a'.a && \text{A.4.2} \\ &= a. [(a')'+ a'] && \text{A.2.2} \\ &= a.1 && \text{A.4.1} \\ &= a && \text{A.3.2}\end{aligned}$$

Teorem 2.7. Yutma teoremi: a, b Boole cebrinin elemanları olmak üzere

$$a+a.b = a$$

$$a.(a+b) = a$$

eşitlikleri geçerlidir.

Tanıt:

$$\begin{aligned}a+a.b &= a.1+a.b && \text{A.3.2} \\ &= a(1+b) && \text{A.2.2} \\ &= a.1 && \text{Teorem 2.3} \\ &= a && \text{A.3.2}\end{aligned}$$

Diğer eşitlik dualite teoremince tanıtlanır.

Teorem 2.8. Boole cebrinin herhangi iki elemanı arasında

$$a+a' b = a+b$$

$$a.(a'+b) = ab$$

eşitlikleri sağlanır.

Tanıt:

$$\begin{aligned}a+a'b &= (a+a') (a+b) && \text{A.2.1} \\ &= 1.(a+b) && \text{A.4.1} \\ &= a+b && \text{A.3.2}\end{aligned}$$

ikinci eşitlik, yukarıda tanıtladığımız eşitliğin duali alınarak yapılır.

Teorem 2.9. Boole cebri (+) ve(.) işlemleri için gruplandırılabilir, yani associative dir.

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

$$a.(b.c) = (a.b) .c \text{ dir.}$$

Tanıt: $X=a+(b+c)$ $Y=(a+b)+c$ olsun. $X = Y$ olduğunu tanıtlayacağız.

$$\begin{aligned}aX &= aX \\ &= a[a+(b+c)] \\ &= a && \text{Teorem 2.7}\end{aligned}$$

ve

$$\begin{aligned}aY &= aY \\ &= a [(a+b)+c] \\ &= a (a+b)+a.c && \text{A. 2.2} \\ &= a+a.c && \text{Teorem 2.7} \\ &= a \text{ dır}\end{aligned}$$

O halde eşdeğerlik bağıntısının geçişlilik özelliği gereğince

$$aX=aY=a \quad (1) \text{ dır.}$$

Diğer taraftan

$$\begin{aligned} a' X &= a' [a + (b+c)] \\ &= a' a + a' (b+c) && \text{A.2.2} \\ &= 0 + a' (b+c) && \text{A.4.2} \\ &= a' (b+c) && \text{A.3.1} \end{aligned}$$

ve

$$\begin{aligned} a' Y &= a' [(a+b)+c] \\ &= a' (a+b) + a' c && \text{A.2.2} \\ &= a' a + a' b + a' c && \text{A. 2.2} \\ &= 0 + a' (b+c) && \text{A.4.2 ve A.2.2} \\ &= a' (b+c) && \text{A.3.1} \end{aligned}$$

geçişlilik özelliğiyle

$$a' X = a' Y = a' (b+c) \quad (2) \text{ dir.}$$

1 ve 2 eşitliklerinden

$$\begin{aligned} aX + a' X &= aY + a' Y \\ (a+a')X &= (a+a')Y && \text{A.2.2} \\ X &= Y && \text{A.4.1} \end{aligned}$$

Yani

$$a+(b+c) = (a+b)+c \text{ dir.}$$

Dual eşitlik benzer biçimde tanıtlanır.

Teorem 2.10. De Morgan teoremi: Boole cebrinin birer elemanı olan a ve b arasında

$$(a+b)' = a' b'$$

$$(ab)' = a'+b'$$

bağıntıları vardır.

Tanıt:

$$\begin{aligned} (a+b)+a' b' &= [(a+b)+a'] [(a+b)+b'] && \text{A.2.a} \\ &= [(a+a')+b] [a+(b+b')] && \text{Teorem 2.9} \\ &= (1+b) (a+1) && \text{A.4.1} \\ &= 1.1 && \text{Teorem 2.3} \\ &= 1 && \text{Teorem 2.5} \end{aligned}$$

Diğer taraftan

$$\begin{aligned} (a+b) . (a' b') &= aa' b'+ba' b' && \text{A.2.a} \\ &= 0 b'+a'0 && \text{A.4.b} \\ &= 0 \end{aligned}$$

dir.

O halde aksiyom 4 gereğince $(a+b)' = a' b'$ dür. Dual olan ikinci ifade benzer biçimde tanıtlanır. De Morgan teoremi, matematiksel tümevarımla

$$(a_1+a_2+\dots+a_n)' = (a'_1 \cdot a'_2 \cdot a'_3 \cdot \dots \cdot a'_n)$$

ve

$$(a_1 \cdot a_2 \cdot \dots \cdot a_n)' = a'_1 + a'_2 + \dots + a'_n$$

biçiminde genelleştirilebilir.

Teorem 2.11. Consensus teoremi

$$a.b + a' c + b.c = a.b + a' c$$

Tanıt:

$$a.b + a' c + b.c = a.b + a' c + b.c.1$$

$$\begin{aligned}
&= a.b + a'.c + b.c.(a + a') \\
&= a.b + a'.c + a.b.c + a'.b.c \\
&= a.b(1 + c) + a'.c(1 + b) \\
&= a.b + a'.c
\end{aligned}$$

Teorem 2.2. m-elemanlı bir I kümesinin bütün alt kümelerinden oluşan bir kuvvet kümesi S, kümeler kuramının ($\cup$) birleşim, ($\cap$) kesişim, tümleyen işlemleriyle bir Boole cebri oluşturur.

Tanıt kümeler teorisinin aksiyomları ile Boole Cebri aksiyomları arasındaki karşılıklıktan yararlanarak yapılabilir.

Örnek 2.1. $Z=\{A,B\}$ kümesinin alt kümelerinden oluşan $S= \{ (\emptyset), (A),(B),(AB) \}$ kümesi, kümeler kuramının kesişim, birleşim, tümleyen işlemleriyle bir Boole cebri oluşturur. S kümesinin elemanları için birleşim, kesişim ve tümleyen işlemlerine ilişkin tanım tabloları aşağıda verildiği gibidir. (Tablo 2.1). 1 ve 0 birim elemanları sırasıyla (A,B) ve $\emptyset$ kümeleri alınarak, bağımsız aksiyomların sağlandığı görülür

Tablo 2.1. S kümesine ilişkin işlemlerin tanım bağıntıları.

$\cup$	$(\emptyset)$	(A)	(B)	(A,B)
$(\emptyset)$	$(\emptyset)$	(A)	(B)	(A,B)
(A)	(A)	(A)	(A,B)	(A,B)
(B)	(B)	(A,B)	(B)	(A,B)
(A,B)	(A,B)	(A,B)	(A,B)	(A,B)

$\cap$	$(\emptyset)$	(A)	(B)	(A,B)
$(\emptyset)$	$(\emptyset)$	$(\emptyset)$	$(\emptyset)$	$(\emptyset)$
(A)	$(\emptyset)$	(A)	$(\emptyset)$	(A)
(B)	$(\emptyset)$	$(\emptyset)$	(B)	(B)
(A,B)	$(\emptyset)$	(A)	(B)	(A,B)

$'$	$(\emptyset)$	(A)	(B)	(A,B)
$(\emptyset)$	(A,B)	$(\emptyset)$	$(\emptyset)$	$(\emptyset)$
(A)	$(\emptyset)$	(A)	$(\emptyset)$	(A)
(B)	$(\emptyset)$	$(\emptyset)$	(B)	(B)
(A,B)	$(\emptyset)$	(A)	(B)	(A,B)

a) Birleşim işlemi

b) Kesişim işlemi

c)Tümleyen

2.3 Sonlu Kümeler Arasında Tanımlanan Fonksiyonlar

Teorem 2.13. $f : A \rightarrow B$ tanımlanan farklı fonksiyon sayısı, A'nın kardinalitesi $\#A=p$ ve B nin kardinalitesi $\#B=N$, olduğuna göre, N farklı işaretin tekrarlanabilir p li permütasyonları (devşirim) sayısı olan N^p dir. Yani her bir permütasyon bir fonksiyona karşı düşer.

Tanıt tekrarlanabilen elemanlı permütasyon tanımıyla kolayca yapılır.

Örnek 2.2. $A = (a_1, a_2, a_3)$ ve $B = (b_1, b_2)$ olsun. $\#A = p = 3$ ve $\#B = N = 2$ dir. fonksiyon sayısı $N^p = 2^3 = 8$ dir. Bütün fonksiyonlar aşağıda verilmiştir.

	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
a_1	b_1	b_2	b_1	b_2	b_2	b_2	b_1	b_1
a_2	b_1	b_2	b_2	b_1	b_2	b_1	b_2	b_1
a_3	b_1	b_2	b_2	b_2	b_1	b_1	b_1	b_2

Teorem 2.14. $S^n \rightarrow S$ ya tanımlanabilen farklı n-değişkenli fonksiyon sayısı, S in kardinalitesi ($\#S = N$) N olmak üzere $N^{(N^n)}$ dir.

Tanıt: Tanım 2.6 da verilen n-değişkenli fonksiyon tanımı uyarınca, S in n kez kartezyen çarpımı olan $S^n = S \times S \times \dots \times S$ kümesindeki eleman sayısı N^n yani N

tekrarlanabilen farklı elemanın n li permütasyonları sayısı kadardır ($\#S^n = N^n$). Öyleyse Teorem 2.12 den dolayı $S^n \rightarrow S$ 'e fonksiyon sayısı $N^{(N^n)}$ dir.

Örnek 2.3. $S = (S_1, S_2, S_3)$ olduğuna göre $f: S^2 \rightarrow S$ e tanımlanan iki değişkenli fonksiyon sayısı, $N = 3$, $n = 2$ olduğundan $N^{(N^n)} = 3^{(3^2)} = 3^9 = 19683$ dür. S^2 kümesinde $N^n = 3^2 = 9$ eleman vardır ve tablo 2.2. de gösterilmiştir. İlk iki sütun sırasıyla x_1, x_2 bağımsız değişkenlerinin aldığı değerlere karşı düşer. Diğer sütunlardan her biri bir fonksiyona karşı düşer.

Tablo 2.2. Üç elemanlı bir kümeye ilişkin tanımlanabilen iki değişkenli fonksiyonlar.

$x_1 x_2$	$f_1(x_1 x_2)$	$f_2(x_1 x_2)$	$f_3(x_1 x_2)$	.	.	$f_{19683}(x_1 x_2)$
$a_1 a_1$	a_1	a_1	.	.	.	a_3
$a_1 a_2$	a_1	a_1	.	.	.	a_3
$a_1 a_3$	a_1	a_1	.	.	.	a_3
$a_2 a_1$	a_1	a_1	.	.	.	a_3
$a_2 a_2$	a_1	a_1	.	.	.	a_3
$a_2 a_3$	a_1	a_1	.	.	.	a_3
$a_3 a_1$	a_1	a_1	.	.	.	a_3
$a_3 a_2$	a_1	a_1	.	.	.	a_3
$a_3 a_3$	a_1	a_2	.	.	.	a_3

2.4 Boole ve Boole Olmayan Fonksiyonlar

Tanım 2.13. n -değişkenli **Boole fonksiyonu**: $x_1, x_2, \dots, x_n$ değişkenlerine (+), (.), (') işlemleri uygulayarak, S^n kümesindeki her bir elemana S kümesinde bir eleman karşı düşüren fonksiyona denir. Bir başka yaklaşımla formül biçiminde ifade edilebilen $S^n \rightarrow S$ e fonksiyonlardır. Boole olan fonksiyonların dışındaki bütün fonksiyonlara **Boole olmayan fonksiyon** denir.

Tanım 2.14. **Boole fonksiyonunun uzunluğu**: Boole fonksiyonunun formülle verilen ifadesinde tekrarlanmış veya tekrarlanmamış sabitler, tekrarlanmış veya tekrarlanmamış değişkenler ile tekrarlanmış veya tekrarlanmamış işlemlerin toplam sayısıdır.

$F = a$, $f = x_1$ 1 uzunluklu fonksiyona, $f = x_1'$ 2 uzunluklu, $f = x_1 + x_2$ ise 3 uzunluklu fonksiyona birer örnektir. Bir fonksiyon, yazılış biçimine göre farklı uzunlukta olabilir. Örneğin $f_1 = (x_1 + x_2)'$ ve $f_2 = x_1' \cdot x_2$ fonksiyonları eşdeğer olup birincisi 4 uzunluklu ikincisi ise 5 uzunlukludur.

Teorem 2.15. $f(x_1 x_2 \dots x_i \dots x_n) |_{x_i=0} = f(x_1, x_2, \dots, a, x_n)$ olmak üzere n-değişkenli bir

Boole
fonksiyonu,

$$f(x_1 \dots x_n) = f(x_1 \dots x_i \dots x_n) |_{x_i=0} x'_i + f(x_1 \dots x_i \dots x_n) |_{x_i=1} x_i \quad (1)$$

veya duali

$$f(x_1 \dots x_n) = [(x_1 \dots x_i \dots x_n) |_{x_i=1} + x'_i] [f(x_1 \dots x_i \dots x_n) |_{x_i=0} + x_i] \quad (2)$$

biçimlerinde yazılabilir.

Tanıt: Önce ilk ifadeyi matematiksel tümevarımla sonra ikinci ifadeyi dual olarak tanıtlayalım. $L=1$ uzunluklu Boole fonksiyonları için (1) ifadesi geçerlidir. $L=1$ uzunluklu Boole fonksiyonları $f(x_1 \dots x_n)=a$, $f(x_1 \dots x_n)=x_i$ dir.

$f(x_1 \dots x_n)=a$ için

$$f(x_1 \dots x_i \dots x_n) |_{x_i=0} = a, f(x_1 \dots x_i \dots x_n) |_{x_i=1} = a \text{ olup}$$

$$f(x_1 \dots x_i \dots x_n) |_{x_i=0} x'_i + f(x_1 \dots x_i \dots x_n) |_{x_i=1} x_i = a x'_i + a x_i = a(x'_i + x_i) = a.1 = a \text{ dır.}$$

$f(x_1 \dots x_n)=x_i$ için

$$f(x_1 \dots x_i \dots x_n) |_{x_i=0} = 0, f(x_1 \dots x_i \dots x_n) |_{x_i=1} = 1 \text{ olup}$$

$$f(x_1 \dots x_i \dots x_n) |_{x_i=0} x'_i + f(x_1 \dots x_i \dots x_n) |_{x_i=1} x_i = 0 x'_i + 1 x_i = 0 + x_i = x_i \text{ dir.}$$

$f(x_1 \dots x_n)=x_j$ için

$$f(x_1 \dots x_i \dots x_n) |_{x_i=0} = x_j, f(x_1 \dots x_i \dots x_n) |_{x_i=1} = x_j \text{ olup}$$

$$f(x_1 \dots x_i \dots x_n) |_{x_i=0} x'_i + f(x_1 \dots x_i \dots x_n) |_{x_i=1} x_i = x_j x'_i + x_j x_i = x_j(x'_i + x_i) = x_j.1 = x_j$$

dir.

Şimdi $L \leq k$ için (1) ifadesinin geçerli olduğunu varsayalım ve $L = k+1$ uzunluklu bir Boole fonksiyonu için bu ifadenin geçerliliğini tanıtlayalım. $(k+1)$ uzunluklu bir Boole fonksiyonu k veya daha küçük uzunluklu g ve h fonksiyonları cinsinden ancak aşağıdaki üç farklı biçimde yazılabilir.

a) $f(x_1 \dots x_n) = g(x_1 \dots x_n).h(x_1 \dots x_n)$

b) $f(x_1 \dots x_n) = g(x_1 \dots x_n)+h(x_1 \dots x_n)$

c) $f(x_1 \dots x_n) = [g(x_1 \dots x_n)]'$

Bu olası üç farklı durumu ayrı ayrı inceleyelim.

a) g ve h nin uzunlukları L veya daha küçük uzunluklu olduklarına göre

$$g(x_1 \dots x_n) = g(x_1 \dots x_i \dots x_n) |_{x_i=0} x'_i + g(x_1 \dots x_i \dots x_n) |_{x_i=1} x_i$$

$$h(x_1 \dots x_n) = h(x_1 \dots x_i \dots x_n) |_{x_i=0} x'_i + h(x_1 \dots x_i \dots x_n) |_{x_i=1} x_i$$

yazılabilir. Öyleyse

$$\begin{aligned} g(x_1 \dots x_i \dots x_n) \cdot h(x_1 \dots x_i \dots x_n) &= g(x_1 \dots x_i \dots x_n)|_{x_i=0} h(x_1 \dots x_i \dots x_n)|_{x_i=0} x'_i \\ &\quad + g(x_1 \dots x_i \dots x_n)|_{x_i=1} h(x_1 \dots x_i \dots x_n)|_{x_i=1} x_i \\ &= f(x_1 \dots x_i \dots x_n)|_{x_i=0} x'_i + f(x_1 \dots x_i \dots x_n)|_{x_i=1} x_i \end{aligned}$$

b)

$$\begin{aligned} g(x_1 \dots x_i \dots x_n) + h(x_1 \dots x_i \dots x_n) &= [g(x_1 \dots x_i \dots x_n)|_{x_i=0} + h(x_1 \dots x_i \dots x_n)|_{x_i=0}] x'_i \\ &\quad + [g(x_1 \dots x_i \dots x_n)|_{x_i=1} + h(x_1 \dots x_i \dots x_n)|_{x_i=1}] x_i \\ &= f(x_1 \dots x_i \dots x_n)|_{x_i=0} x'_i + f(x_1 \dots x_i \dots x_n)|_{x_i=1} x_i \end{aligned}$$

c)

$$\begin{aligned} [g(x_1 \dots x_n)]' &= [g(x_1 \dots x_i \dots x_n)|_{x_i=0} x'_i + g(x_1 \dots x_i \dots x_n)|_{x_i=1} x_i]' \\ &= [(g'(x_1 \dots x_i \dots x_n)|_{x_i=0} + x'_i) \cdot (g'(x_1 \dots x_i \dots x_n)|_{x_i=1} + x_i)] \\ &= g'(x_1 \dots x_i \dots x_n)|_{x_i=0} g'(x_1 \dots x_i \dots x_n)|_{x_i=1} + g'(x_1 \dots x_i \dots x_n)|_{x_i=0} x'_i + g'(x_1 \dots x_i \dots x_n)|_{x_i=1} x_i \end{aligned}$$

x_i
=1

Consensus teoremi gereğince

$$\begin{aligned} &= g'(x_1 \dots x_i \dots x_n)|_{x_i=0} \cdot x'_i + g'(x_1 \dots x_i \dots x_n)|_{x_i=1} x_i \\ &= f(x_1 \dots x_i \dots x_n)|_{x_i=0} x'_i + f(x_1 \dots x_i \dots x_n)|_{x_i=1} x_i \end{aligned}$$

$L = k+1$ uzunluklu f fonksiyonu için de verilen bağıntı geçerli olduğuna göre her zaman geçerlidir. (2) eşitliği (1) eşitliğinin duali olduğundan, geçerlidir.

Teorem 2.15. Boole fonksiyonlarının **birinci tip kanonik açılımı**: $f : S^n \rightarrow S$ e n -değişkenli bir Boole fonksiyonu "**çarpımlar terimleri toplamı**"

$$f(x_1 \dots x_n) = f(0,0,\dots,0) x'_1 x'_2 \dots x'_n + f(0,0,\dots,0,1) x'_1 x'_2 \dots x_n + f(1,1,\dots,1) x_1 x_2 \dots x_n$$

x_n

biçiminde ifade edilebilir.

Tanıt: Verilen fonksiyonun, teorem 2.15 e göre açılımı ard ardına n kez uygulanarak

$$\begin{aligned} f(x_1 \dots x_n) &= x'_1 f(0, x_2, \dots, x_n) + x_1 f(1, x_2, \dots, x_n) \\ &= x'_1 f(0, x_2, \dots, x_n) + x_1 f(1, x_2, \dots, x_n) \\ &= x'_1 [x'_2 f(0, 0, \dots, x_n) + x_2 f(0, 1, \dots, x_n)] + x_1 [x'_2 f(1, 0, \dots, x_n) + x_2 f(1, 1, \dots, x_n)] \\ &= \dots \\ &= f(0,0,\dots,0) x'_1 x'_2 \dots x'_n + f(0,0,\dots,0,1) x'_1 x'_2 \dots x_n + f(1,0,\dots,1) x_1 x'_2 x_3 \dots x_n + \dots \\ &\quad \dots + f(1,1,\dots,1) x_1 x_2 \dots x_n \end{aligned}$$

bulunur.

1. tip kanonik açılım bir başka biçimde

$$f(x_1 x_2 \dots x_n) = \sum_{i=0}^{2^n-1} f(i) m_i(x_1 x_2 \dots x_n)$$

yazılabilir, öyle ki

$$f(0) = f(0, 0, \dots, 0)$$

$$f(1) = f(0, 0, \dots, 1)$$

.

$$f(2^n-1) = f(1, 1, \dots, 1)$$

ve

$$m_0(x_1 x_2 \dots x_n) = x'_1 x'_2 \dots x'_n$$

$$m_1(x_1 x_2 \dots x_n) = x'_1 x'_2 \dots x_n$$

.

.

.

$$m_{2^n-1}(x_1 x_2 \dots x_n) = x_1 x_2 \dots x_n \text{ dir.}$$

$f(i)$ ve m_i deki i , $0 \leq i \leq 2^n-1$ olan 10 tabanında bir sayıdır. $f(i)$ ye ilişkin eşitliğin sağında fonksiyonun değişkenlerinin aldıkları değerlerin saptanması: i sayısının iki tabanındaki karşılığında, basamaklar değişkenlere karşı düşürüldüğünde, basamaklarda ki sayamalar ilgili değişkenlerin aldıkları değerlerdir. m_i lere ilişkin eşitliğin sağındaki çarpım terimlerindeki değişkenlerin saptanması: i sayısının iki tabanındaki karşılığında, basamaklar değişkenlere karşı düşürüldüğünde, bir basamaktaki sayamak 0 ise ilgili değişken tümleyen (x') olarak, sayamak 1 ise ilgili değişken kendi (x) olarak çarpımlar teriminde görülür.

Tanım 2.15 Minterim: Bir Boole fonksiyonuna ilişkin 1.tip kanonik açınımda çarpımlar terimlerine minterim denir

Teorem 2.16 Boole Fonksiyonunun ikinci tip kanonik açılımı: $f: S^n \rightarrow S$ e n -değişkenli bir Boole fonksiyonu "**toplamlar terimleri çarpımı**"

$$\begin{aligned} f(x_1, x_2, \dots, x_n) = \\ = [f(0, 0, \dots, 0) + x_1 + x_2 + \dots + x_n] [f(0, 0, \dots, 1) + x_1 + x_2 + \dots + x'_n] \dots [f(1, 1, \dots, 1) + x'_1 + x'_2 + \dots + x'_n] \end{aligned}$$

biçiminde ifade edilebilir.

Tanıt: Teorem 2.15 in duali alınarak tanıt yapılır.

2. tip kanonik açılım bir başka biçimde

$$f(x_1 x_2 \dots x_n) = \prod_{i=0}^{2^n-1} [f(i) + M_i(x_1 x_2 \dots x_n)]$$

yazılabilir, öyle ki $f(i)$ 1. tip kanonik açınımda tanımlandığı gibidir M_i ler ise

$$\begin{aligned} M_0(x_1 x_2 \dots x_n) &= x_1 + x_2 + \dots + x_n \\ M_1(x_1 x_2 \dots x_n) &= x_1 + x_2 + \dots + x'_n \\ &\vdots \\ M_{2^n-1}(x_1 x_2 \dots x_n) &= x'_1 + x'_2 + \dots + x'_n \text{ dir.} \end{aligned}$$

M_i deki i $0 \leq i \leq 2^n-1$ olan 2^n tabanında bir sayıdır. M_i ye ilişkin eşitliğin sağındaki toplam terimlerindeki değişkenlerin saptanması: i sayısının iki tabanındaki karşılığında basamaklar değişkenlere karşı düşürüldüğünde, bir basamaktaki sayı 0 ise ilgili değişken kendi (x) olarak, sayı 1 ise ilgili değişken tümleyen (x') olarak toplam terimlerinde görülür.

Tanım 2.16 Makterim: Bir Boole fonksiyonuna ilişkin 2. tip kanonik açınımda toplam terimlerine Makterim denir.

Teorem 2.17. Bir Boole fonksiyonuna ilişkin 1. ve 2. tip açınımlar yeganedir.

Tanıt: Varsayalım ki bir Boole fonksiyonun farklı iki 1. tip açılımı X ve Y olsun. X ve Y farklı olduğuna göre, en az bir minterimin katsayısı X ve Y de farklı olmak zorundadır. Bu demektir ki fonksiyonun aynı bir tanım elemanına farklı iki değer elemanı karşı düşer, fonksiyon tanımıyla çelişir. O halde farklı iki 1. tip açılım olamaz. 2. tip açılım içinde benzer tanıt yapılır.

Sonuç 2.1. Herhangi bir n -değişkenli Boole fonksiyonuna ilişkin 0,1 simgelerinin n -li permütasyonları sayısı (2^n) kadar olan tanım elemanına karşı düşen değer elemanları biliniyorsa, bu fonksiyon yegane olarak belirlenmiş olur. Bir başka deyişle herhangi bir Boole fonksiyonu 2^n özel değeriyle yegane olarak belirlenir. Bu sonuç 1. ve 2. tip kanonik açınımlardan hemen görülmektedir.

Örnek 2.4. Örnek 2.1 de verilen S kümesi için $f: S^2 \rightarrow S$ e tanımlanan iki değerli bir Boole fonksiyonun $2^2 = 2^2 = 4$ adet 0,1 simgelerinden oluşan tanım elemanı için aldığı değerler $f(0,0) = A$; $f(0,1) = 1$; $f(1,0) = 0$; $f(1,1) = B$ olsun. Bu Boole fonksiyonuna ilişkin 1. tip kanonik açılım:

$$f(x_1 x_2) = f(0,0)x'_1 x'_2 + f(0,1)x'_1 x_2 + f(1,0)x_1 x'_2 + f(1,1)x_1 x_2 = Ax'_1 x'_2 + x'_1 x_2 + Bx_1 x_2 \text{ dir.}$$

2. tip kanonik açılım

$$f(x_1x_2) = [f(0,0)+x_1+x_2] [f(0,1)+x_1+x'_2] [f(1,0)+x'_1+x_2] [f(1,1)+x'_1+x'_2]$$

$$= (A+x_1+x_2) (x'_1+x_2) (B+x'_1+x'_2)$$

dür. S^2 kümesindeki diğer tanım elemanları için Boole fonksiyonunun aldığı değer elemanları yukarıdaki herhangi bir açılım formülünden hesaplanarak bulunabilir. Bu fonksiyon tablo biçiminde tablo 2.3 de verilmiştir. Örneğin $x_1x_2 = AB$ domen elemanı için Boole fonksiyonunun aldığı değer. örnek 2.1 deki işlem tanım tablolarından yararlanarak

$$f(x_1x_2) = Ax_1x_2 + x'_1x_2 + Bx_1x'_2 = A A' B' + A' B + B A B = 0.B' + B.B + 0.B = B x_1 = B$$

$$x_1 = A, x_2 = B$$

bulunur.

Tablo2.3. Örnek Boole fonksiyonu için $S^2 \rightarrow S$ ilişkisi

S^2	00	01	10	11	0A	0B	A0	B0	1A	1B	A1	B1	AA	AB	BA	BB
S	A	1	0	B	A	1	0	A	0	B	B	1	0	B	A	1

Sonuç 2.2. Verilen bir fonksiyonun Boole olup olmadığına, 1. veya 2. tip kanonik açınımlardan yararlanarak, karar verilebilir.

Verilen $f: S^2 \rightarrow S$ fonksiyon Boole varsayılarak, 0,1 den oluşmuş 2^n tanım elemanı için fonksiyonun aldığı değerden yararlanarak 1. (2.) tip kanonik açılım yazılır. n: değişken sayısı $N=\#S$ olduğuna göre, tanım kümesinde arta kalan (N^n-2^n) tanım elemanı için fonksiyonun aldığı değer, 1 (2.) tip kanonik açılım formülüyle bulunan değerlerden farklı ise fonksiyon Boole olmayan fonksiyon, aynı ise fonksiyon Boole fonksiyondur.

Örnek 2.5. Örnek 2.1 de yerilen $S=\{\emptyset, (A), (B), (AB)\}$ kümesi ile $(+), (.), (')$ işlemlerinin oluşturduğu Boole cebirinde $S^2 \rightarrow S$ e tanımlanan 2 değişkenli iki fonksiyon tablo 2.4 te verilmiştir. $N=4, n=2$ dir f_1 fonksiyonu örnek 2.4 te verilen fonksiyon olup bir Boole fonksiyondur. Çünkü tablo 2.4 te çift çizgi ile ayrılan birinci bölgedeki verilerle f_1 e ilişkin kanonik açınımlar, ikinci bölgedeki f_1 e ilişkin verileri sağlar. f_2 fonksiyonu ise 0-1 den oluşmuş $2^n = 2^2 = 4$ tanım elemanı için f_1 in aldığı değerleri almaktadır. f_2 , Boole fonksiyonu olsaydı f_1 in aynı olacaktı. Oysa $f_1(0,A)=A$ olduğu halde $f_2(0,A)=0$ dir. O halde f_2 Boole fonksiyonu olamaz, Boole olmayan bir fonksiyondur. Tablo 2.4 te, çift çizgi ile ayrılan 1. bölgede f_1 in aldığı değerleri alan, fakat ikinci bölgede en az bir yerde f_1 den farklı olan Boole olmayan çok sayıda fonksiyon verilebilir.

Tablo.2.4. Boole olan ve olmayan fonksiyonlara örnek.

S^2	00	01	10	11	0A	0B	A0	B0	1A	1B	A1	B1	AA	AB	BA	BB
f_1	A	1	0	B	<u>A</u>	1	0	A	0	B	B	1	0	B	A	1
f_2	A	1	0	B	<u>0</u>	1	0	A	0	B	B	1	0	B	A	1

Sonuç 2.3. # $S = N$ olduğuna göre $f: S^n \rightarrow S$ n -değişkenli Boole fonksiyonu sayısı $N^{(2^n)}$ dir. $f: S^n \rightarrow S$ e n -değişkenli Boole olmayan fonksiyon sayısı ise $[N^{(N^n)} - N^{(2^n)}]$ dir.

1.veya 2. kanonik açınımlar gereğince, Boole fonksiyonları için 0,1 den oluşmuş 2 tanım elemanına karşı düşen değer elemanları bağımsız olarak seçilebilir. Farklı seçim sayısı, $N^{(N^n)}$ dir. -N farklı işaretin 2^n li permütasyonları sayısı- Diğer tanım elemanlarına karşı düşen değer elemanları bağımsız seçilemez, çünkü açınım formülüyle yegane olarak belirlenirler. O halde Boole fonksiyonu sayısı $N^{(2^n)}$ dir. $S^n \rightarrow S$ e toplam fonksiyon sayısı, teorem 2.13 uyarınca $N^{(N^n)}$ olduğuna göre Boole olmayan fonksiyon sayısı $[N^{(N^n)} - N^{(2^n)}]$ dir.

Sonuç 2.4. $S = \{0,1\}$ kümesi için tanımlanan $S^n \rightarrow S$ e n -değişkenli fonksiyonların tamamı Boole fonksiyonudur.

$$N^{(N^n)} = N^{(2^n)} = 2^{(2^n)} \text{ dir.}$$

2.5 İki Değerli Boole Cebri ve Anahtar Fonksiyonları

Tanım 2.17. **İki değerli Boole Cebri:** 0 ve 1 birim elemanlarından oluşan $S=\{0,1\}$ kümesinin, Tablo 2.5 te verilen toplama, çarpma, tümleyen işlemleriyle oluşturduğu Boole cebrine iki değerli Boole cebri denir.

Tablo 2.5. $S=\{0,1\}$ kümesi için tanımlanan toplama, çarpma, tümleyen işlemleri.

+	0	1		.	0	1			'
0	0	1		0	0	0		0	1
1	1	1		1	0	1		1	0

İleride ki konularda iki değerli Boole cebrinin bir matematiksel model olarak temsil ettiği fiziksel sistemler üzerinde duracağız.

Tanım 2.8 . **Anahtar fonksiyonları:** İki değerli Boole cebrinde tanımlanan fonksiyonlara anahtar fonksiyonları denir.

Sonuç 2.4 uyarınca ikili Boole cebrinde tanımlanan bütün fonksiyonlar Boole fonksiyonu olduğuna göre, bütün anahtar fonksiyonları Boole fonksiyonu olup formül biçiminde ifade edilebilirler.

Tanım 2.19. **Doğru ve yanlış minterimler:** İkili Boole cebrinde tanımlanan anahtar fonksiyonlarının 1. tip kanonik açınımlarında katsayısı 1 olan minterimlere doğru, 0 olan minterimlere ise yanlış minterim denir.

Tanım 2.20. **Doğru ye yanlış makterimler:** İkili Boole cebrinde tanımlanan anahtar fonksiyonlarının 2. tip kanonik açınımlarında katsayısı 1 olan makterimlere yanlış, 0 olan makterimlere ise doğru makterim denir.

Anahtar fonksiyonlarının 1. tip kanonik açınımlarında yalnızca doğru minterimleri, 2. tip kanonik açınımda yalnızca doğru makterimlerin düşünülmesi yeter. Çünkü 1. tip

açınımında 1, 2. tip açınımında 0 olurlar ve fonksiyonun ifadesinde görülmezler. O nedenle 1. ve 2. tip kanonik açınımları

$$f(x_1, x_2, \dots, x_n) = \sum \text{Doğru minterimler veya}$$

$$f(x_1, x_2, \dots, x_n) = \prod \text{Doğru makterimler}$$

biçiminde yazılabilir.

Anahtar fonksiyonlarının bir gösterilimi de

$$f(x_1, x_2, \dots, x_n) = \sum \text{Doğru minterimlerin 10 tabanındaki indisleri, veya}$$

$$f(x_1, x_2, \dots, x_n) = \prod \text{Doğru makterimlerin 10 tabanındaki indisleri}$$

biçiminde olur. Örneğin $f(x_1, x_2, x_3) = \sum 2,4,5,7$ ile gösterilen anahtar fonksiyonu $f(010) = f(100) = f(101) = f(111) = 1$ ve $f(000) = f(001) = f(011) = f(110) = 0$ olan bir fonksiyondur.

Tanım 2.21 **Doğruluk tablosu**: İki değerli Boole cebirinde bir fonksiyona ilişkin tanım ve değer kümesi elemanlarını gösteren tabloya doğruluk tablosu denir.

Örnek 2.6. Aşağıdaki Tablo 2.6 da bir anahtar fonksiyonuna ilişkin doğruluk tablosu ve minterim ve makterimler verilmiştir.

Tablo 2.6. Bir Boole fonksiyonuna ilişkin doğruluk tablosu. ve fonksiyona ilişkin minterim ve makterimler.

i	x_1	x_2	x_3	$f(x_1, x_2, x_3)$	Minterimler	makterimler
	2^2	2^1	2^0	$f(i)$	M_i	M_i
0	0	0	0	0	$m_0 = x'_1 x'_2 x'_3$	$M_0 = x_1 + x_2 + x_3$
1	0	0	1	0	$m_1 = x'_1 x'_2 x_3$	$M_1 = x_1 + x_2 + x'_3$
2	0	1	0	1	$m_2 = x'_1 x_2 x'_3$	$M_2 = x_1 + x'_2 + x_3$
3	0	1	1	0	$m_3 = x'_1 x_2 x_3$	$M_3 = x_1 + x'_2 + x'_3$
4	1	0	0	1	$m_4 = x_1 x'_2 x'_3$	$M_4 = x'_1 + x_2 + x_3$
5	1	0	1	1	$m_5 = x_1 x'_2 x_3$	$M_5 = x'_1 + x'_2 + x'_3$
6	1	1	0	0	$m_6 = x_1 x_2 x'_3$	$M_6 = x'_1 + x'_2 + x_3$
7	1	1	1	1	$m_7 = x_1 x_2 x_3$	$M_7 = x'_1 + x'_2 + x'_3$

Örnek fonksiyonun 1. tip kanonik açılımı

$$f(x_1, x_2, x_3) = \sum 2,4,5,7$$

$$= f(0)m_0 + f(1)m_1 + f(2)m_2 + f(3)m_3 + f(4)m_4 + f(5)m_5 + f(6)m_6 + f(7)m_7$$

$$= 0.m_0 + 0.m_1 + 1.m_2 + 0.m_3 + 1.m_4 + 1.m_5 + 0.m_6 + 1.m_7$$

$$= m_2 + m_4 + m_5 + m_7$$

$$= x'_1 x_2 x'_3 + x'_1 x'_2 x'_3 + x_1 x'_2 x_3 + x_1 x_2 x_3 \text{ dir.}$$

2. tip kanonik açılım ise

$$f(x_1, x_2, x_3) = \prod 0,1,3,6$$

$$= [f(0)+M_0] [f(1)+M_1] [f(2)+M_2] [f(3)+M_3] [f(4)+M_4] [f(5)+M_5] [f(6)+M_6] + f(7)+M_7$$

$$= [0+M_0] [0+M_1] [1+M_2] [0+M_3] [1+M_4] [1+M_5] [0+M_6] [1+M_7]$$

$$=M_0.M_1.M_3.M_6$$

$$=(x_1+x_2+x_3)(x_1+x_2+x'_3)(x_1+x'_2+x'_3)(x'_1+x'_2+x_3) \text{ dır}$$

2.6 Özet

Bu bölümde en genel anlamda Boole cebri ve bu cebirde tanımlanan Boole olan fonksiyonlarla Boole olmayan fonksiyonlar incelenmiştir. Boole olan fonksiyonlara ilişkin genel 1. ve 2. tip kanonik açınımlar verilmiştir. Son olarak ta ikili Boole cebri, bu cebirde tanımlanan anahtar fonksiyonları ve bu fonksiyonların minterim ve makterimler cinsinden nasıl yazıldığı gösterilmiştir. Bundan sonraki bölümlerde yalnızca ikili Boole cebri kullanılacaktır.

BÖLÜM 3 SAYISAL DİZGELER VE BOOLE CEBRİ

1.1 Giriş

Bu bölümde sırasıyla **önergeler cebri** ve kapı elemanlarından oluşan sayısal dizgelerin, ikili Boole cebri ile matematiksel olarak nasıl modellendirileceği incelenecektir. Boole cebriindeki $\{S\}$ kümesinin 1 ve 0 elemanları, fiziksel dizgelerdeki elektriksel büyüklüklerin (akım veya gerilim) olup olmasına karşı düşürüleceklerdir.

Tanım 3.1. **Akım mantığı**: Bir ikili sayısal dizgede 1 ve 0 simgeleri akımın olup olmasını simgeliyorsa, bu dizge akım mantığı ile çalışıyor denir.

Tanım 3.2. **Gerilim mantığı**: Bir ikili sayısal dizgede 1 ve 0 simgeleri gerilimin olup olmasını simgeliyorsa, bu dizge gerilim mantığı ile çalışıyor denir.

Tanım 3.3. **Pozitif mantık**: İkili sayısal dizgede herhangi bir noktada 1 in simgelediği gerilim (akım) 0 in simgelediği gerilimden (akımdan) büyükse dizge pozitif mantıkla çalışıyor denir.

Tanım 3.4. **Negatif mantık**: İkili sayısal dizgede herhangi bir noktada 1 in simgelediği gerilim (akım) 0'ın simgelediği gerilimden (akımdan) küçükse dizge negatif mantıkla çalışıyor denir.

3.2 Önergeler Cebri ve İki Değerli Boole Cebri

Bilindiği gibi önermeler cebri dilin matematiksel betimlemesinde kullanılır. Bu betimlemede önermelerin özelliği ya doğru ya da yanlış olmasıdır. Buna göre "Bu gün Gümüşsuyunda yağmur yağabilir" bir önerme değildir ama "Bugün Gümüşsuyunda yağmur yağdı" bir önermedir. Dilde kullandığımız bazı cümleler, önermeler cebri ($\vee$) birleşim, ($\wedge$) kesişim, ($\neg$) olumsuz işlemleriyle aşağıdaki tablo 3.1 deki gibi betimlenebilirler. X ve Y önerme değişkenleri olup D(doğru) ve Y(yanlış) değerlerinden birini alırlar.

Tablo 3.1. Dildeki cümleler, önermeler cebri ve Boole cebri karşılaştırması.

Dildeki ifade	Önergeler Cebrinde	Boole Cebrinde
olumsuz x	$\neg x$ Olumsuz	x'
x veya y veya ikisi de	$x \vee y$ Birleşim	$x+y$
x ve y	$x \wedge y$ Kesişim	$x.y$
Eğer x öyleyse y	$x \rightarrow y$ x,y' yi gerektirir	$x'+y$
x eğer ve yalnız y	$x=y$	$xy+x'y'$

Teorem 3.1. $S=\{ D(\text{doğru}), Y(\text{yanlış}) \}$ kümesi ve aşağıda tablo 3.2 te tanımlanan toplama($\vee$), çarpma($\wedge$), olumsuz($\neg$) işlemleri, bir Boole cebri oluşturur.

Tablo 3.2. Önergeler cebri işlemlerinden toplama, çarpma ve olumsuz işlemlerinin tanım tablosu.

x	y	$X \vee Y$	$X \wedge Y$	$\neg x$
D	D	D	D	Y
D	Y	D	Y	Y
Y	D	D	Y	D
Y	Y	Y	Y	D

Tanıt: $D \rightarrow 1$, $Y \rightarrow 0$ a karşı düşürülüp, Boole cebri aksiyomları sağlatılarak tanıt yapılabilir.

Önerme fonksiyonu bir Boole fonksiyonu ile temsil edilebilir. Örneğin aşağıdaki koşullardan en az biri (VE/VEYA) sağlandığında doğru(D), aksi halde yanlış(Y) olan bir F önerme fonksiyonu düşünelim. Bu fonksiyon bir asansörün güvenilir çalışabilmesi için uyarı işareti üreten fonksiyonda olabilir. F önermesi,

a) Asansör çalışmakta, içi boş ve çağrılmamışsa ve/veya

b) Asansör çalışmakta ve kapısı açık

koşulları altında doğru aksi halde yanlış olsun.

Şimdi sırasıyla U,V,Y,Z değişkenlerini aşağıdaki önermelerle tanımlayalım.

U: Asansör çalışıyor

V: Asansör boş

Y: Asansör çağrılmış ; Y': Asansör çağrılmamış

z: Kapı açık ; z': Kapı kapalı

Koşullardan herhangi biri sağlandığında $F = 1$, sağlanmadığında ise $F=0$ olacaktır. Bu lojik toplama işlemidir. Yani F, koşulların toplamı biçiminde görülecektir. Herhangi bir koşul ise belli değişkenlerin birlikte 1 değeri almalarında (1); herhangi bir değişkenin istenmeyen bir değer almasında (0) olacaktır. Bu işlem lojik çarpma işlemine karşı düşer. Netice olarak F önerme fonksiyonu, değişkenlerin çarpımıyla elde edilen koşulların, toplamı biçiminde görülecektir.

Buna göre F önermesi:

$$F = UVY' + UZ$$

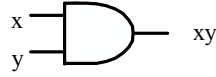
biçiminde bir Boole fonksiyonu ile ifade edilebilir. Bu Boole fonksiyonu üzerinde yapılacak çeşitli işlemler sonucu sözle anlatım farklı biçimlerde ifade edilebilir. Örneğin yukarıdaki fonksiyondaki terimler U parantezine alınabilir.

3.3 Kapı Elemanlarından Oluşan Devreler ve İki Değerli Boole Cebri

Önce yaygın biçimde kullanılan kapı elemanlarını tanıyalım.

1.VE kapısı (AND gate): Şekil 3. 1 de giriş çıkış ilişkisinin doğruluk tablosu ve gösterilişi verilmiştir.

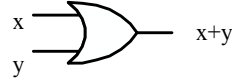
x	y	xy
0	0	0
0	1	0
1	0	0
1	1	1



Şekil 3.1 VE kapısının doğruluk tablosu ve gösterilişi

2. **VEYA kapısı (OR gate):** Şekil 3.2 de giriş çıkış ilişkisinin doğruluk tablosu ve gösterilişi verilmiştir.

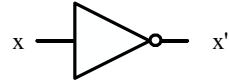
x	y	x+y
0	0	0
0	1	1
1	0	1
1	1	1



Şekil 3.2 VEYA kapısının doğruluk tablosu ve gösterilişi

3. **TÜMLEYEN kapısı (NOT gate):** Şekil 3.3 te giriş çıkış ilişkisinin doğruluk tablosu ve gösterilişi verilmiştir.

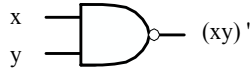
x	x'
0	1
1	0



Şekil 3.3. Tümlleyen kapısının doğruluk tablosu ve gösterilişi

4. **TÜVE kapısı (NAND Gate):** Şekil 3.4 te giriş çıkış ilişkisinin doğruluk tablosu ve gösterilişi verilmiştir.

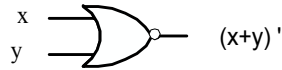
x	y	(xy)'
0	0	1
0	1	1
1	0	1
1	1	0



Şekil 3.4. TÜVE kapısının doğruluk tablosu ve gösterilişi

5. **TÜVEYA kapısı (NOR gate):** Şekil 3.5 te giriş çıkış ilişkisinin doğruluk tablosu ve gösterilişi verilmiştir.

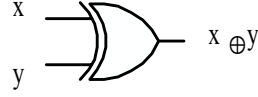
x	y	(x+y)'
0	0	1
0	1	0
1	0	0
1	1	0



Şekil3.5. TÜVEYA kapısının doğruluk tablosu ve gösterilişi.

6. **Ayrıcalı-VEYA kapısı (Exclusive-OR gate):** Şekil 3.6 da giriş çıkış ilişkisinin doğruluk tablosu ve gösterilişi verilmiştir.

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0



Şekil 3.6 Ayricalı-VEYA kapısına ilişkin doğruluk tablosu ve gösterilişi.

Bir kombinezonsal devreye ilişkin Boole fonksiyonu, kapılara ilişkin giriş - çıkış bağıntısı (tanım bağıntısı) formül biçiminde yazılarak ta kolayca bulunabilir.

Ayricalı-VEYA kapısının VEYA kapısından ayrılması 11 girişi için de çıkışın 0 olmasıdır. Yani Ayricalı-VEYA kapısı girişler farklı işaretli ise 1 çıkışını, aynı işaretli ise 0 çıkışını verir.

Kapıların çıkışında, tümleyen işlemi için kullanılan küçük yuvarlak (o) işareti; kapıların girişlerinde, değişkenlerin tümleyenleri için de kullanılabilir.

Tanım 3.14. Giriş yelpazesi (Fan in) Bir kapı elemanına uygulanabilecek maksimum giriş sayısına denir.

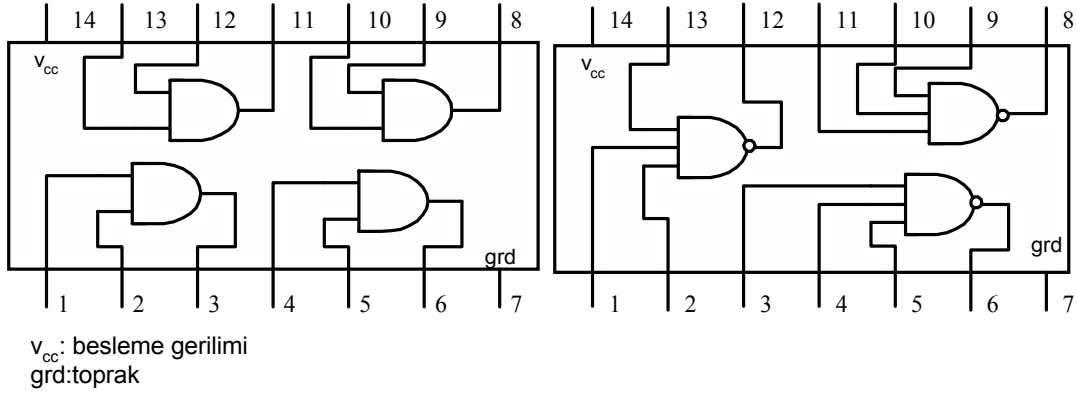
Tanım 3.5. Çıkış yelpazesi (Fan out) Bir kapı elemanından alınabilecek maksimum çıkış sayısına denir.

Tanım 3.6. Mantık devresi: Sonlu sayıda kapı elemanlarından oluşan ve aynı bir girişine birden fazla çıkışın bağlı olmadığı birleşik devreye denir.

Mantık devreleri, pozitif (negatif) akım (gerilim) mantıkları $S=\{ 0,1 \}$ kümesine VE, VEYA, TÜMLEYEN kapıları çarpma, toplama, tümleyen işlemlerine karşı düşürülerek, iki değerli bir Boole cebri ile matematiksel olarak modellendirilebilirler.

Bu dersin kapsamı içerisinde kapı elemanları, giriş-çıkış ilişkisi verilerek tanımlanmıştır. Her bir kapı elemanının gerçekleştirilmesi için transistörlerden yararlanılır. Transistörlerle yapılan gerçekleştirilmelerde çeşitli teknolojiler kullanılmaktadır (TTL, MOS, CMOS gibi) Bu teknolojiler ve kapı elemanlarının iç yapısı bu dersin kapsamı dışında tutulmuştur.

Kapı elemanları yukarıda gösterildiği gibi tek başlarına ticari olarak satılmazlar, tümleştirilmiş devre olarak satılırlar. Lojik devrelerde kullanılan tümleştirilmiş devreleri dört gruba ayırmak mümkündür. **SSI** (Single Scale Integrated Circuits) Küçük ölçekli tümleşik devreler. SSI elemanlar içlerinde 10 kadar kapı elemanı bulunduran devrelerdir. İki giriş ve bir çıkışlı kapı elemanlarından bir SSI elemanda normal olarak dört tane bulunur. Şekil 3.7 de iki SSI elemanı örneği verilmiştir. Tek giriş ve tek çıkışlı olan tümleyen elemanından ise 8 tanesi ticari bir tümleşik devrede görülür.



Şekil 3.7 Pratikte kullanılan iki SSI tümleşik devre şeması

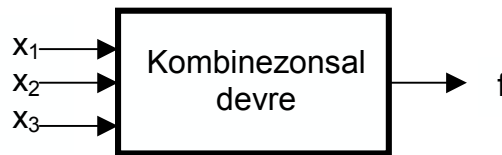
MSI (Medium Scale Integrated Circuits) Orta ölçekli tümleşik devreler, içersinde 10-100 kadar kapı elemanı bulunduran devrelerdir ve bu dersin kapsamında incelenecektir. LSI (Large Scale Integrated Circuits) Büyük ölçekli tümleşik devrelerde, 100-1000 kadar kapı elemanı bulunabilir, yine bu dersin kapsamında incelenecektir. VLSI (Very Large Scale Integrated Circuits) Büyük ölçekli tümleşik elemanlar binlerce kapı elemanını kapsayan devrelerdir bu dersin kapsamında incelenmeyecektir. Örneğin mikroişlemciler bir VLSI elemandır.

Tanım 3.7. Kombinezonsal devre: çıkışları, girişleriyle yegane olarak belirlenen mantık devrelerine denir. Buna göre aynı girişe, farklı çıkışlar veren mantık devresi kombinezonsal değildir. İlerde göreceğimiz gibi bu devreler ardışıldır.

Her bir kombinezonsal devreye bir Boole fonksiyonu (analiz), her bir Boole fonksiyonuna da en az iki kombinezonsal devre (sentez) karşı düşürülebilir.

Verilen bir kombinezonsal devreye karşı düşen Boole fonksiyonunun bulunması, devrenin analiz edilmesine karşı düşer (Giriş/çıkış ilişkisinin analitik ifadesi). Analiz işlemi, bilindiği gibi, devreler teorisinde, çeşitli yöntemlerle yapılmaktadır- örneğin çevre, düğüm yöntemleri gibi- Oysa lojik devrelerde **analiz işlemi**, her bir çıkışa ilişkin bir Boole fonksiyonun bulunmasıyla yapılabilir. Bu fonksiyonların bulunması devreler teorisindekine göre çok daha basit ve kolaydır.

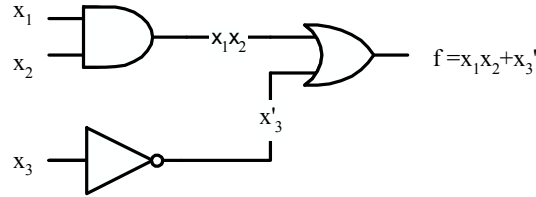
Analizi yapılacak devre giriş ve çıkışları belirlenmiş bir kapalı kutu olarak verilmiş olsun. Kombinezonsal devrenin çıkışı, tanım gereğince girişler cinsinden yegane olarak belirlendiğine göre, her bir girişe bir bağımsız değişken karşı düşürülüp fonksiyon olarak bulunur. Her bir giriş permütasyonu ve buna ilişkin çıkış, devrede gözlenerek bir doğruluk tablosu oluşturulur. Bu tabloya karşı düşen Boole fonksiyonu, analiz için aranan fonksiyondur. Örneğin şekil 3.8 de üç girişli ve tek çıkışlı bir kombinezonsal devre ve buna ilişkin doğruluk tablosu verilmiştir. İlerde nasıl elde edebileceğimizi göreceğimiz gibi, bu fonksiyon $f = x_1x_2 + x'_3$ Boole fonksiyonuna eşdeğerdir.



Girişler bağımsız değişkenler			Bağımlı değişkenler			
x_1	x_2	x_3	f	x_1x_2	x_3'	$x_1x_2+x_3'$
0	0	0	1	0	1	1
0	0	1	0	0	0	0
0	1	0	1	0	1	1
0	1	1	0	0	0	0
1	0	0	1	0	1	1
1	0	1	0	0	0	0
1	1	0	1	1	1	1
1	1	1	1	1	0	1

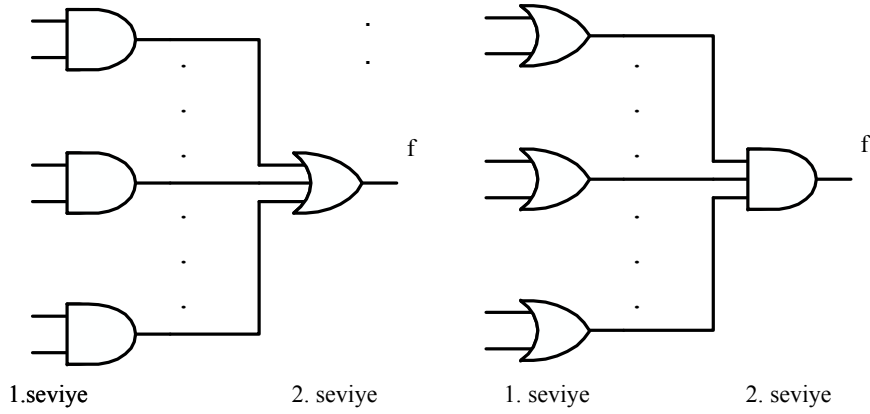
Şekil 3.8 Örnek kombinezonsal devreye ilişkin doğruluk tablosu ve karşı düşen Boole fonksiyonu.

Analizi yapılacak kombinezonsal devre, bir kapalı kutu biçiminde değil de, kapı elemanları ve bağlantılarıyla verilmiş olsun. Bu durumda analiz için gerekli olan fonksiyon, doğruluk tablosuna gerek olmadan, formül biçiminde bulunabilir. Tutulacak yol, her kapının tanım bağıntısını kullanarak, kapı çıkışının girişleri cinsinden yazılması ve bu işleme devrenin çıkışına gelinceye kadar devam edilmesidir. Şekil 3.9 de bir örnek verilmiştir.



Şekil 3.9 Bir kombinezonsal devreye karşı düşen Boole fonksiyonunun bulunmasına örnek

Giriş ve çıkışları belli (doğruluk tablosu veya fonksiyonu belli) bir kutuya ilişkin, kapı elemanlarında oluşmuş devrenin bulunması, **sentez işlemi**, lojik devrelerde en az iki çözümlüdür. n -değişkenli bir Boole fonksiyonuna, 1. ve 2. tip kanonik açınımlardan yararlanarak, her birine birer tane olmak üzere, en az iki kombinezonsal devre karşı düşürülebilir. 1. tip kanonik açınım için çarpımlar terimleri, n -girişli VE kapılarıyla; çarpımlar terimlerinin toplamı olan fonksiyon da, n -girişli VE kapılarının çıkışlarının bir VEYA kapısının girişine verilmesiyle, VEYA kapısının çıkışında, iki seviyeli olarak elde edilir. Sonuç çarpımlar terimlerinin toplamı VE-VEYA devresidir. VE kapıları 1., VEYA kapısı 2. Seviye ye karşı düşer.



(a) çarpımlar terimlerinin toplamı

Birinci tip Kanonik açınım

$$f = \sum f(i)m_i$$

(b) toplamlar terimlerinin çarpımı

İkinci tip kanonik açınım

$$f = \prod (f(i) + M_i)$$

Şekil 3.10 Bir Boole fonksiyonuna ilişkin iki iki-seviyeli en genel biçimde kombinezonsal devre.

2. tip kanonik açınım için de, benzer biçimde, önce VEYA kapıları sonra VE kapısı kullanılarak, iki seviyeli bir kombinezonsal devre elde edilir. Sonuç toplamlar terimlerinin çarpımı VEYA-VE devresidir. VEYA kapıları 1., VE kapısı 2. Seviye ye karşı düşer.

Şekil 3.10 a da bir Boole fonksiyonunun 1. tip kanonik açınımına ilişkin, şekil 3.10 b de de aynı fonksiyonun 2. tip kanonik açınımına ilişkin iki seviyeli kombinezonsal devreler en genel biçimde verilmiştir.

Şekilde görüldüğü gibi, İki-seviyeli gerçekleştirilmelerde devrenin cevap verme süresinin, iki-kapı elemanı gecikmesinin toplamı kadar olduğu, genel olarak söylenebilir.

Örneğin

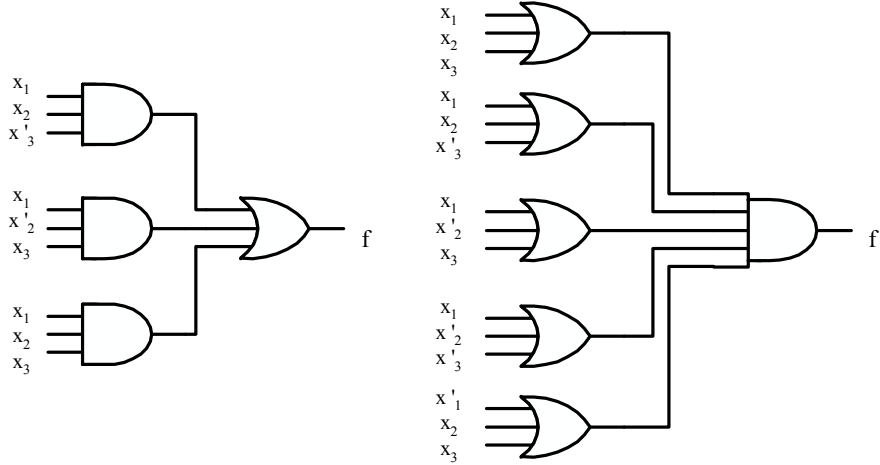
$$f(x_1x_2x_3) = \sum 5,6,7$$

$$= x_1x_2'x_3 + x_1x_2x_3' + x_1x_2x_3$$

$$= \prod 0,1,2,3,4$$

$$= (x_1+x_2+x_3) (x_1+x_2+x_3') (x_1+x_2'+x_3) (x_1+x_2'+x_3') (x_1+x_2+x_3)$$

fonksiyonuna ilişkin iki iki seviyeli kombinezonsal devre Şekil 3.11 de verilmiştir.



Şekil 3.11. Bir örnek Boole fonksiyonuna ilişkin iki iki-seviyeli kombinezonsal devre.

Tanım 3.8. Bütün işlemler (elemanlar): Herhangi bir Boole fonksiyonu, belli işlemlerle (elemanlarla) ifade edilebiliyorsa (gerçekleştirilebiliyorsa), bu işlemler (elemanlar) bir bütün oluşturuyor denir.

Örneğin toplama, çarpma, tümleme işlemleri (VE, VEYA, TÜMLEYEN kapı elemanları) bir bütün oluştururlar. Bütün Boole fonksiyonlarının 1. ve 2. kanonik açınımları vardır ve bu açınımlar toplama, çarpma, tümleme işlemleriyle ifade edilirler.

Teorem 3.3. Toplama ve tümleyen işlemleri (VEYA, TÜMLEYEN elemanları) ile çarpma ve tümleyen (VE, TÜMLEYEN işlemleri) bir bütün oluştururlar.

Tanıt:

İlki için, eksik olan çarpma işlemi, tümleyen ve toplama işlemleriyle

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

biçiminde elde edilir. Burada yalnız toplama ve tümleyen işlemleri vardır.

İkincisi için, eksik olan toplama işlemi, tümleyen ve çarpma işlemleriyle

$$x + y = (x' \cdot y')'$$

biçiminde elde edilir. Burada yalnız çarpma ve tümleyen işlemleri vardır. O nedenle bu işlemler birer bütün oluştururlar.

Tanım 3.9. Minimal bütün işlemler (elemanlar): Bütün işlemlerin (elemanların) sayısı minimum ise bu işlemler (elemanlar) bir minimal bütün oluşturuyor denir.

Teorem 3.4. TÜVE veya TÜVEYA işlemleri (elemanları) birer minimal bütün oluştururlar.

Tanıt:

$$x \cdot y = [(xy)' (xy)']'$$

$$x + y = [(xx)' (yy)']'$$

$$x' = (xx)'$$

olduğundan TÜVE işlemi bir minimal bütün oluşturur.

TÜVEYA işleminin minimal bütünlüğü, benzer biçimde tanıtlanabilir. Şöyle ki

$$x \cdot y = [(x+x)' + (y+y)']'$$

$$x + y = [(x+y)' (x+y)']'$$

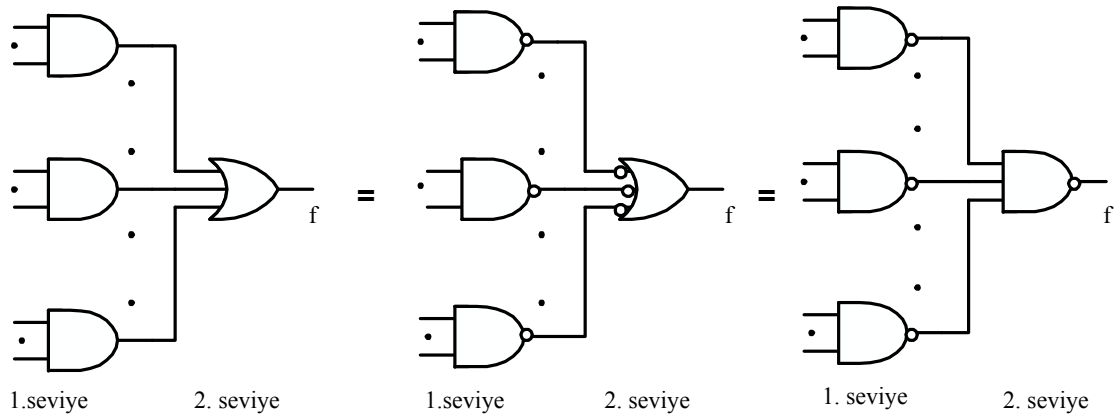
$$x' = (x+x)'$$

dır.

Şekil 3.8 a da verilen iki seviyeli devrede, VE kapılarının çıkışlarının tümleyenlerini ve VEYA kapılarının girişlerinin de tümleyenlerini aldığımızı düşüncemize, giriş çıkış açısından sonuç değişmeyecektir. Teorem gereğince tümleyen tümleyen kendisine eşittir. Öte yandan bütün girişleri tümleyen olan bir VEYA kapısı bir TVE kapısına eşdeğerdir, çünkü De Morgan teoremi gereğince

$$(x'_1 + x'_2 + x'_3 + \dots + x'_n) = (x_1 \cdot x_2 \cdot x_3 \cdot \dots \cdot x_n)'$$

dir. Bu sonuca göre 1.tip kanonik açınım karşı düşen iki-seviyeli gerçeğe, bütün kapıları TVE kapılarıyla yer değiştirdiğimizde; elde edilen devre ilk devreye eşdeğer, üstelik bütün elemanları aynı tip kapı elemanıdır. Bakınız Şekil 3.12.



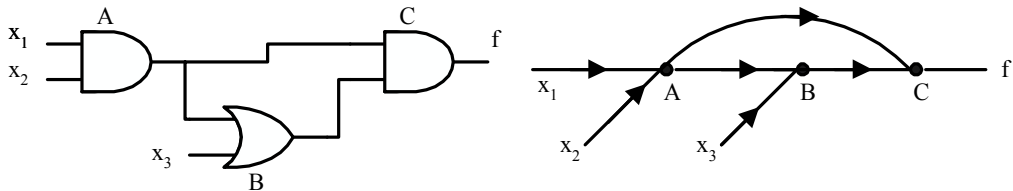
Şekil 3.12 Yalnızca TVE kapıları kullanılarak bir Boole fonksiyonunun en genel gerçekleştirilişi

Benzer biçimde 2 kanonik açınım için de yalnızca TVEYA kapılarıyla (bütün kapıları TVEYA kapılarıyla değiştirilerek) en genel bir devre verilebilir.

Görülüyor ki bütün Boole fonksiyonlarını, gerektiğinde aynı tip kapı elemanlarıyla gerçekleştirmek olasıdır. Tümleşik devre teknolojisi açısından bu sonuç yararlıdır.

Tanım 3.10. Bir **mantık devresine ilişkin graf (çizge)**: Her bir kapı elemanına bir düğüm, her bir giriş ve çıkışa da işaretin akışına göre yönlendirilmiş bir topolojik eleman karşı düşürülerek, elde edilen çizgeye denir.

Örneğin şekil 3.12a daki mantık devresine ilişkin çizge, Şekil 3.12 b de verilmiştir.



a) Örnek mantık devresi.

b) Örnek mantık devresine ilişkin çizge

Şekil.3.12. Bir mantık devresi ve çizgesi.

Tanım 3.11. **Yol**: Bir çizgeye ilişkin bir A düğümünden hareketle, topolojik elemanların yönleri doğrultusunda ilerleyerek, bir B düğümüne varılıyorsa, A ve B düğümleri arasında bir yol vardır denir.

Tanım 3.12. **Çevre**: İki ucu birleşik yola çevre denir.

Tanım 3.13. **İleri beslemeli mantık devresi**: Çizgesi, bir çevre içermeyen mantık devresine denir.

Örneğin Şekil 3.12 b de verilen mantık devresi ileri beslemelidir.

Teorem 3.5. İleri beslemeli bir mantık devresi kombinezonsaldır.

Tanıt: Önce "ileri beslemeli bir mantık devresinde, yalnızca devre girişlerini veya bir alt kümesini giriş kabul eden, en az bir kapı elemanı vardır" önermesinin doğru olduğunu tanımlayalım. Varsayalım ki bu önerme yanlış. Herhangi bir kapı elemanı seçelim, girişlerinden en az biri, varsayım gereğince, bir başka kapı elemanının çıkışıdır. Bu yeni kapı elemanı da, varsayım gereğince, bir başka kapı elemanının çıkışından beslenmek zorundadır. Böylece devam edilirse, kapı elemanları sonlu ve devre birleşik olduğundan, aynı bir devre elemanı iki defa hesaba katılmak zorundadır. Bu devrenin ileri beslemeli olmadığını gösterir, varsayımla çelişir. O halde varsayım yanlış, yani ileri beslemeli bir mantık devresinde, girişleri yalnızca devrenin girişleri olan, en az bir kapı elemanı vardır.

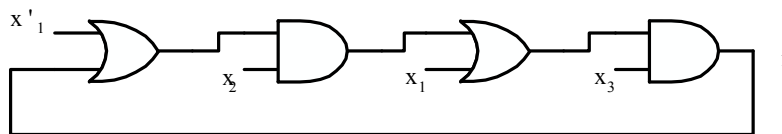
Yukarıda sözü edilen kapılara 1 kapıları diyelim. Öyle kapılar vardır ki girişleri $x_1, \dots, x_n$ ve 1 kapılarının çıkışları olsun. Bunlara da 2 kapıları diyelim. 1 ve 2 kapı çıkışları yalnızca $x_1, \dots, x_n$ girişleri cinsinden ifade edilebilir. Böyle devam edilerek devrenin çıkışlarına varılır. Her bir kapı çıkışı daha öncesine giderek yegane olarak $x_1, \dots, x_n$ girişleriyle belirlendiğine göre, çıkışta $x_1, \dots, x_n$ le yegane olarak belirlenir. O halde devre kombinezonsaldır.

Tanım 3.24. **Geri beslemeli mantık devresi**: Bir mantık devresine ilişkin çizge, bir çevre içeriyorsa, bu devreye geri beslemeli devre denir.

Teorem 3.6. Geri beslemeli mantık devreleri, kombinezonsal veya kombinezonsal olmayan (ardışıl) olabilir.

Tanıtı bir örnek üzerinden yapabiliriz. Şekil 3.13 de verilen geri beslemeli mantık devresi kombinezonsaldır. f çıkış fonksiyonu, girişler cinsinden yegane olarak belirlenir ve

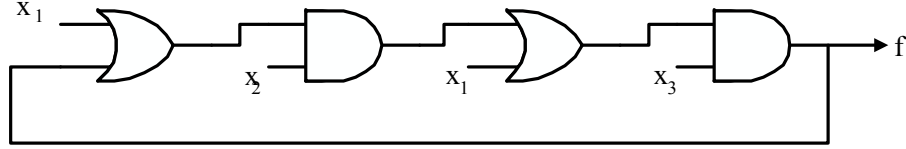
$$f = [(x'_1 + f) x_2 + x_1] x_3 = [(x'_1 + f + x_1) (x_2 + x_1)] x_3 = (x_1 + x_2) x_3 \text{ dür}$$



Şekil 3.13. Kombinezonsal olan bir geri beslemeli mantık devresi.

Şekil 3.14 de verilen, geri beslemeli devre; kombinezonsal değil, ardışıldır.

$f = x_1x_3 + x_2x_3f$ olup sırasıyla t_0, t_1, t_2, t_3 anları için $x_1x_2x_3 = 000, 011, 111, 011$ girişleri uygulandığında, çıkışlar 0,0,1,1 dir. Aynı 011 girişi için çıkışlar farklı olduğundan, devre kombinezonsal olamaz. Uygulanan giriş dizisi ayrıca tablo biçiminde verilmiştir.



$$f = [(x_1 + f)x_2 + x_1]x_3 = (x_1 + f)(x_1 + x_2)x_3 = (x_1 + x_2f)x_3 = x_1x_3 + x_2x_3f$$

	x_1	x_2	x_3	f
t_0	0	0	0	0
t_1	0	1	1	0
t_2	1	1	1	1
t_3	0	1	1	1

Şekil 3. 14.Kombinezonsal olmayan (ardışıl) bir geri beslemeli mantık devresi.

3.4 Özet

Bu bölümde iki değerli Boole cebrinin uygulandığı önermeler cebri ve kapı elemanlarından oluşan mantık devreleri incelenmiştir. Bu devrelerin davranışlarının Boole fonksiyonlarıyla temsil edilebileceği gösterilmiştir. Belli bir davranış biçimi betimleyen Boole fonksiyonlarının, kapı elemanlarından oluşan devrelerle nasıl gerçekleştirilebileceği konuları ele alınmıştır.

BÖLÜM 4 KOMBİNEZONSAL DEVRE SENTEZİ

Önceki bölümde, kapı devrelerine ilişkin Boole fonksiyonlarının bulunması, analiz sorunu incelenmişti. Ayrıca bir Boole fonksiyonuna ilişkin, kapı devresinin bulunması, sentez sorunu da ele alınmış; 1. ve 2. tip kanonik açınımlara ilişkin genel iki seviyeli devreler verilmiştir. Bu bölümde ise sentezin, daha az sayıda kapı elemanlarıyla, gerçekleştirilmesi üzerinde durulacaktır. Bundan sonraki adım da, pratikte kombinezonsal devre sentezinin, nasıl yapılacağı örneklerle incelenecektir.

4.1 Giriş

Amaç, verilen bir fonksiyona eşdeğer, fakat karmaşıklığı daha az olan, fonksiyonların bulunmasıdır. Bir Boole fonksiyonunun karmaşıklığı veya ona karşı gelen devrenin karmaşıklığı, tanımlanması oldukça zor bir kavramdır. Değişik kriterlere göre bir fonksiyonun veya devrenin karmaşıklığı değişmektedir. Örneğin fonksiyonun karmaşıklığı, gerçekleştirilmesinde kullanılan toplam eleman sayısına göre, eleman girişleri sayısına (giriş yelpazesi), kolay onarılabilir olmasına, hata saptamasının kolaylığına, farklı eleman grupları kullanılmasına göre farklı biçimlerde tanımlanabilir. İdeal olanı, bütün kriterlere cevap yerebilen, bir karmaşıklık tanımıdır. Oysa çeşitli kriterler için, gerçekleştirilen fonksiyonların devreleri, birbiriyle çelişen sonuçlar verebilir. Örneğin eleman sayısı az olan basit bir devre, hata saptanması bakımından karmaşık olabilir. İstenen ideal tanımın yapılmasındaki güçlük nedeniyle, tutulan yol, belli koşullar altında, minimal karmaşıklık tanımının verilmesidir.

Tanım 4.1. Minimal (indirgenmiş) fonksiyon: Bir g fonksiyonu, f fonksiyonuna eşdeğer, çarpımlar toplamı (toplamlar çarpımı) biçiminde, minimum sayıda minimum değişkenli terimden oluşuyorsa; g ye f in minimal fonksiyonu denir.

$$f(x_1x_2x_3) = \Sigma 0,1,5,6,7$$

$$f(x_1x_2x_3) = x'_1x'_2x'_3 + x'_1x'_2x_3 + x_1x'_2x_3 + x_1x_2x'_3 + x_1x_2x_3$$

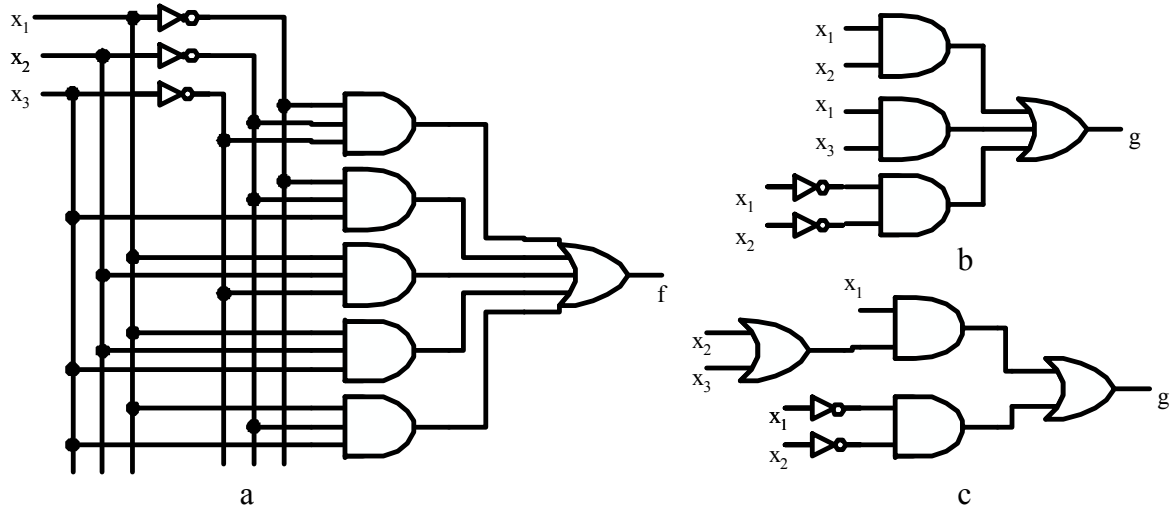
fonksiyonuna eşdeğer

$$g_1(x_1x_2x_3) = x_1x_2 + x_1x_3 + x'_1x'_2$$

fonksiyonu tanıma uygun bir minimal fonksiyondur. Oysa

$$g_2(x_1x_2x_3) = x_1(x_2 + x_3) + x'_1x'_2$$

biçiminde yazılan, g_2 eşdeğer fonksiyonu, çarpımlar toplamı biçiminde olmadığından, tanım gereğince minimal değildir. Her bir fonksiyona ilişkin devreler Şekil.4.1 de verilmiştir.



(a) $f = x_1'x_2'x_3' + x_1'x_2'x_3 + x_1x_2x_3' + x_1x_2x_3 + x_1x_2'x_3$

(b) $g = x_1x_2 + x_1x_3 + x_1'x_2'$

(c) $g = x_1(x_2 + x_3) + x_1'x_2'$

Şekil.4.1.Tanıma uygun minimal ve minimal olmayan fonksiyonlar ve devreleri.

Minimal fonksiyona ilişkin devre, genel gerçekleştirilmeye göre hem kapı sayısı hem de kapılara ilişkin giriş yelpazeleri bakımından daha basittir. Gerçekleştirmede terimler kapı elemanlarına ve terimlerdeki değişken sayısı da giriş yelpazelerine karşı düşer. 2. seviyedeki elemanın giriş yelpazesi terim sayısı kadardır.

Tanım 4.2. İndirgenemeyen (irreducible, irredundant) fonksiyon: Bir f fonksiyonuna eşdeğer indirgenemeyen g fonksiyonu, çarpımlar toplamı biçiminde olan ve herhangi bir değişkeni kaldırıldığında f ile eşdeğerliği bozulan fonksiyondur.

Örneğin

$$f(x_1x_2x_3) = \sum 0,2,3,4,5,7$$

$$f(x_1x_2x_3) = x_1'x_2'x_3' + x_1'x_2x_3' + x_1'x_2x_3 + x_1x_2'x_3' + x_1x_2'x_3 + x_1x_2x_3$$

fonksiyonuna eşdeğer

$$g_1(x_1x_2x_3) = x_1'x_3' + x_2'x_3' + x_2x_3 + x_1x_3$$

fonksiyonu indirgenemeyendir, fakat minimal değildir.

Sonuç 1. Her minimal fonksiyon indirgenemeyendir, karşıtı doğru değildir.

Sonuç 2. Aynı bir fonksiyona ilişkin birden fazla minimal veya indirgenemeyen fonksiyon olabilir. Bir başka deyişle minimal veya indirgenemeyen fonksiyonlar yegane değildir.

Önceki örnekteki f e eşdeğer

$$g_2(x_1x_2x_3) = x_1'x_3' + x_1x_2' + x_2x_3 \text{ ile}$$

$$g_3(x_1x_2x_3) = x_1'x_2 + x_2'x_3' + x_1x_3$$

fonksiyonları hem indirgenemeyen hem de minimal fonksiyonlardır.

1 tip açınım için, verilen minimal fonksiyona ilişkin devrede, VE kapılarının sayısı, VE kapılarının giriş yelpazesi, VEYA kapısının giriş yelpazesi azalır. Benzer biçimde 2. tip açınım için verilen minimal fonksiyona ilişkin devrede, VEYA kapılarının sayısı, VEYA, kapılarının giriş yelpazesi ve VE kapısının giriş yelpazesi azalır.

Minimal fonksiyonların bulunmasına ilişkin iki yöntem verilecektir. Birincisi, Quine McCluskey yöntemi diye de bilinen, tablo yöntemi olup, genel ve bilgisayar programlamasına elverişlidir. İkincisi Karnaugh diyagramı yöntemidir. Bu yöntem, bağımsız değişken sayısı az olan (3,4,5, gibi) fonksiyonların minimalleştirilmesinde, kolaylık sağlar.

4.2 Tablo Yöntemiyle Minimal Fonksiyonların Bulunması

Bu bölümde önce yöntemi tanıtır, sonra adımlarını ayrı ayrı inceleyeceğiz.

4.2.1 Tablo yöntemi

Önce 1. tip sonra benzer yolla 2. tip kanonik açınım için minimalleştirmeyi inceleyeceğiz.

Tanım 3.4. Örtten (cover) fonksiyon: g fonksiyonun 1 değerini aldığı her giriş için f fonksiyonu da 1 değerini alıyorsa, g ye f için gerekir veya f, g yi örtüyor denir. Sırasıyla $g \rightarrow f$ veya $f \supseteq g$ ile gösterilir.

Tablo 4 1de.verilen $h = x'_1x'_2 + x_1x_2x_3$ fonksiyonu $f = x_1+x'_2$ için gerekir ($h \rightarrow f$), veya f , h yı örter ($f \supseteq h$).

Tablo 4.1. Gerekir veya örtten fonksiyonlara örnek ($f \supseteq h$, $h \rightarrow f$)

x_1	x_2	x_3	h	f
0	0	0	1	1
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	0	1
1	0	1	0	1
1	1	0	1	1
1	1	1	0	1

Tanım 4.4. Asal bileşen (prime implicant) Eğer p çarpımlar terimi, f tarafından örtülüyor ($p \rightarrow f$) ve p deki herhangi bir değişken kaldırıldığında f, p yi örtmüyorsa; p ye f in asal bileşenidir denir.

Örneğin $f(x_1x_2x_3) = x'_1x_2 + x_1x_3 + x'_2x'_3$ fonksiyonu için $p = x'_1x_2$ bir asal bileşendir. Çünkü $p = x'_1x_2$ de , x'_1 ve x_2 kaldırıldığında, sırasıyla geriye kalan x_2 ve x'_1 , f tarafından örtülmemektedir.

Teorem 4.1. Bir fonksiyona ilişkin, indirgenemeyen fonksiyondaki çarpımlar terimleri asal bileşenlerdir.

Tanıt: f e eşdeğer ve indirgenemeyen bir fonksiyon f^* olsun. (α) da, f in asal bileşeni olmayan, bir çarpımlar terimi olsun ve f^* da görülsün. (α) asal bileşen olmadığından en

azından bir değişken silindiğinde, f^* ile f in eşdeğerliği bozulmaz, fakat f^* indirgenemez olmaktan çıkar. Varsayımımızla çelişiyor o halde (α) asal bileşen olmak, zorundadır. Bir başka deyişle f^* daki bütün çarpımlar terimleri, f in bir asal bileşeni olmak zorundadır.

Bu teoremden, bir fonksiyonun minimalleştirilmesinde önce bütün asal bileşenlerin bulunması gerektiği ortaya çıkmaktadır. Sonra da bulunan asal bileşenlerden, toplamaları minimal fonksiyonu veren, minimum sayıda asal bileşenler, alt kümesinin bulunması gerekecektir.

Tanım 4.5. Temel asal bileşen (Essential prime implicant) Bir asal bileşen, bir doğru minterimi örtüyor ve başka hiçbir asal bileşen aynı doğru minterimi örtmüyorsa, bu asal bileşene Temel asal bileşen denir.

f fonksiyonunun bütün doğru minterimlerinin, f e eşdeğer indirgenemeyen veya minimal g fonksiyonu tarafından örtülmesi gerektiğinden, f in bütün temel asal bileşenleri, g de görülmek zorundadır.

Yukarıdaki açıklamaların ışığında, tablo yöntemiyle bir fonksiyonun minimalleştirilmesini üç adımda özetleyebiliriz.

1. Bütün temel asal bileşenler bulunup, minimal fonksiyona hepsi birlikte alınır.
2. Temel asal bileşenlerin örttüğü asal bileşenler, minimal fonksiyonda görülmezler, yani atılırlar.
3. Birinci adımda, f in bütün doğru minterimleri temel asal bileşenler tarafından örtülüyorsa; minimal fonksiyon yeganedir ve temel asal bileşenler toplamına eşittir.
4. Aksi halde bulunan temel asal bileşenlere, öyle minimum sayıda asal bileşen eklemeliyiz ki temel asal bileşenler ve seçilen bu asal bileşenler birlikte, f fonksiyonunu örtsünler ve ona eşit olsunlar. Bu dördüncü adım, direkt olarak hemen yapılamayabilir, sistematik, kuralı ileride verilecektir.

4.2.2 Asal Bileşenlerin Bulunması

Belli koşullar altında, k değişkenli iki çarpımlar terimi, $(k-1)$ değişkenli bir çarpımlar terimine dönüşebilir. (k) değişkenli iki çarpımlar teriminde, $(k-1)$ değişken, değişkenler ve tümleyenleri bakımından tamamen aynı ve yalnızca bir değişken terimlerden birinde kendisi, diğerinde tümleyen biçiminde görülüyorsa; bu iki terim birleşir. Bu birleşme sonunda, farklı olan değişken kaybolur. Geriye, her iki terimde ortak ve aynı $(k-1)$ değişkenlerden oluşan, bir terim kalır. Matematiksel olarak

$$Aa' + Aa = A(a+a') = A.1 = A$$

bağıntısı düşünülmektedir. Öyle ki A , $(k-1)$ tane, değişkenler ve tümleyenleri bakımından tamamen aynı, değişkenden oluşur. (a) ise yalnızca bir değişkendir. Bu işleme "**birleştirme işlemi**" diyeceğiz. Birleşme sonucu, iki tane k değişkenli çarpımlar terimi yerine, bir tane, $(k-1)$ değişkenli, bir çarpımlar terimi bulunur. Örneğin

$$\begin{aligned} x'_1 x_2 x_3 x'_4 + x'_1 x_2 x_3 x_4 &= \\ &= x'_1 x_2 x_3 (x_4 + x'_4) \\ &= x'_1 x_2 x_3 . 1 \end{aligned}$$

$$= x'_1 x_2 x_3$$

dür. $(x'_1 x_2 x_3 x'_4)$ ve $(x'_1 x_2 x'_3 x_4)$ çarpımlar terimleri, yukarıda açıklandığı anlamda birleştirilmeye uygun değildir. Çünkü $x_3 x_4$ gibi, iki değişken, her iki terimde farklı biçimler de görülmektedir: birinde $x'_3 x_4$ diğerinde $x_3 x'_4$ biçimindedir.

Doğru minterimler üzerinden yapılan birleştirme işlemi, doğru minterimlerin ikili tabanda temsil edilişlerinden yararlanarak, daha kolay yapılabilir. Örneğin

$$x'_1 x_2 x_3 x'_4 \text{ minteriminin iki tabanındaki temsili } 0110 \text{ ile}$$

$$x'_1 x_2 x_3 x_4 \text{ minteriminin iki tabanındaki temsili } 0111 \text{ ile}$$

yapıldığında (bakınız bölüm 2.5. Anahtar fonksiyonlarının kanonik açınımları) görülüyor ki yalnız en az ağırlıklı basamaktaki sayamalar farklı, diğer basamaklardaki sayamalar ise aynıdır. O halde birleştirme sonucu (011-)e karşı gelen terim $x'_1 x_2 x_3$ çıkar, (-) kaybolan değişkenin bulunduğu basamakta görülür. Sonuç olarak, 1 sayısı (0 sayısı) yalnızca bir fark eden, iki tabanındaki sayıların temsil ettiği doğru minterimler karşılaştırılmadır.

Birleştirme işlemi, ard arda yapılarak, değişken ve terim sayısı daha da azaltılabilir. Örneğin aşağıdaki dört çarpımlar terimine, birleşme işlemi ard arda uygulanarak, 2 değişkenli, bir çarpımlar terimi elde edilir.

$$x_1 x_2 x'_3 x'_4 + x_1 x_2 x_3 x'_4 + x_1 x_2 x'_3 x_4 + x_1 x_2 x'_3 x'_4 = x_1 x_2 x'_3 (x'_4 + x_4) + x_1 x_2 x_3 (x'_4 + x_4) = x_1 x_2 (x'_3 + x_3) = x_1 x_2$$

Bu işlemlerin sistematik, bir biçimde yapılabilmesi için tutulacak yol adım adım aşağıdaki gibi özetlenebilir:

1. Doğru minterimler öyle gruplandırılır ki aynı grupta olan doğru minterimlerin, ikili tabanda temsili olan sayılardaki 1 sayıları aynı olsun. Bulunan gruplar, içerlerindeki 1 sayısına göre, dizinlenerek (indislenerek) sıralandırılır.
2. En küçük dizinli (indisli) gruptaki her bir iki tabanlı sayı, bir sonraki dizinli grupta ki her bir sayı ile karşılaştırılır. Birleştirilmeye uygun çiftler varsa birleştirilir. Bu işleme, i. dizinli gruptaki her bir iki tabanlı sayı ile (i+1) dizinli gruptaki her bir iki tabanlı sayı, teker teker karşılaştırılıp, olası olanlar birleştirilerek, bütün gruplar bitinceye kadar devam edilir. Komşu gruplardan, iki tabanında temsil edildiklerinde, yalnızca bir basamaktaki sayamağı fark eden, terimler birleştirilir. Elde edilen sonuç terimde, birleştirilen iki terimin, karşılıklı basamaklardaki ortak sayamalar, aynen saklanır, farklı olan sayamaların bulunduğu basamağa (-) konur. Böyle birleşmeye giren terimler $\checkmark$ ile işaretlenir. Aynı bir terimin birden fazla birleşmeye girebileceği gözden kaçırılmamalıdır. Sonuçta yine aynı dizinli gruplar oluşturulur.
3. Bu adımda yapılan işlem öz olarak ikinci adımdakinin aynıdır. Komşu gruplardaki terimler, yalnızca bir basamaktaki sayamaları farklı ve aynı basamaklarda (-) işareti taşıyorlarsa, birleştirilebilir. İkinci adımdaki işlemler tekrarlanır. İşleme yeni bir birleştirilebilen terim bulunamayana kadar devam edilir. $\checkmark$ ile işaretlenmiş terimler asal bileşenlerdir. Çünkü her biri f için gerekir ve başka bir çarpımlar terimi tarafından örtülmezler.

$f = \sum 0,1,2,5,7,8,9,10,13,15$ fonksiyonunun asal bileşenlerini bulalım. 1. adıma ilişkin tablo, Tablo 4.2a da ve sonraki adımlara ilişkin tablolarda Tablo 4.2.b c de verilmiştir.

Tablo 4.2. Örnek fonksiyona ilişkin asal bileşenlerin bulunmasındaki tablolar.

		x_1	x_2	x_3	x_4
G_1	$\sqrt{0}$	0	0	0	0
	$\sqrt{1}$	0	0	0	1
G_2	$\sqrt{2}$	0	0	1	0
	$\sqrt{8}$	1	0	0	0
	$\sqrt{5}$	0	1	0	1
G_3	$\sqrt{9}$	1	0	0	1
	$\sqrt{10}$	1	0	1	0
	$\sqrt{7}$	0	1	1	1
G_4	$\sqrt{13}$	1	1	0	1
G_5	$\sqrt{15}$	1	1	1	1

(a)

		x_1	x_2	x_3	x_4
	$\sqrt{0,1}$	0	0	0	-
G'_1	$\sqrt{0,2}$	0	0	-	0
	$\sqrt{0,8}$	-	0	0	0
	$\sqrt{1,5}$	0	-	0	1
	$\sqrt{1,9}$	-	0	0	1
G'_2	$\sqrt{2,10}$	-	0	1	0
	$\sqrt{8,9}$	1	0	0	-
	$\sqrt{8,10}$	1	0	-	0
	$\sqrt{5,7}$	0	1	-	1
G'_3	$\sqrt{5,13}$	-	1	0	1
	$\sqrt{9,13}$	1	-	0	1
G'_4	$\sqrt{7,15}$	-	1	1	1
	$\sqrt{13,1}$	1	1	-	1

(b)

		x_1	x_2	x_3	x_4
G''_1	0,1,8,9	-	0	0	-
	0,2,8,10	-	0	-	0
G''_2	1,5,9,13	-	-	0	1
G''_3	5,7,13,15	-	1	-	1

(c)

Örneğin Tablo 4.2a da gruplaşmalar yapıldıktan sonra G_2 deki 1 ile G_3 deki 5 birleştirilmiş, (0-01) bulunmuş, Tablo 4.2b deki tabloya (1,5) (0-01) olarak aktarılmış, 1. ve 5. terimler $\sqrt{}$ ile işaretlenmiştir. Kaybolan 2. basamaktaki sayamak yerine (-) gelmiştir. İşleme benzer biçimde devam edilmiştir. sonuçta $\sqrt{}$ ile işaretlenmemiş (-00-), (-0-0), (- -01), (-1-1) lere yani sırasıyla $x'_2x'_3$; $x'_2x'_4$; x'_3x_4 ; x_2x_4 asal bileşenlerine varılmıştır.

4.2.3 Asal bileşenler tablosu ve minimal fonksiyonların bulunması

Sütunlarında, fonksiyona ilişkin doğru minterimlerin, satırlarında ise asal bileşenlerin bulunduğu tabloya, asal bileşenler tablosu diyeceğiz. Eğer bir satırdaki asal bileşen, bir sütundaki doğru minterimi örtüyorsa, bu satır ve sütunun kesiştiği yere x işareti konur. Örneğin Tablo 4.2. de asal bileşenlerini bulduğumuz fonksiyona ilişkin, asal bileşenler tablosu Tablo 4.3. de verilmiştir. Bulunan asal bileşenlerin öyle bir minimal alt kümesi seçilmelidir ki bütün doğru minterimler örtülsün. Yani tablodaki her bir sütun, seçilen asal bileşenlere ilişkin satırlarda, en az bir x işareti bulundurmalı ve asal bileşenlerdeki değişken sayısı olabildiğince minimal olmalıdır.

Asal bileşenler tablosunun bir sütununda, yalnızca bir x işareti varsa, bu işaretin bulunduğu satıra ilişkin asal bileşen, temel asal bileşendir. Çünkü sütundaki doğru minterim, yalnızca bu asal bileşen tarafından örtülmektedir. Temel asal bileşenin bulunduğu satıra temel satır denir. Temel asal bileşenleri ayırt etmek için, x işareti, $\otimes$ biçimine dönüştürülüp, ilgili temel asal bileşen $\checkmark$ ile işaretlenir. Temel asal bileşenler saptandıktan sonra, bunların örttükleri bütün minterimler $\checkmark$ ile işaretlenir. Örneğin x_2x_4 ve $x'_2x'_4$ temel asal bileşeni; 0,2,8 ve 10 sütunlarının temsil ettikleri doğru minterimleri örter. Eğer temel asal bileşenler doğru minterimlerin tamamını örtmüyorsa, başka asal bileşenler eklemek gerekecektir. örneğimizde x_2x_4 ve $x'_2x'_4$ temel asal bileşenleri 0,2,5,7,8,10,13,15 doğru minterimlerini örter, fakat 1 ve 9 doğru minterimlerini örtmezler. Temel asal bileşenlere ek olarak x'_3x_4 veya $x'_2x'_3$ asal bileşenlerini alırsak bütün doğru minterimler örtülmüş olur. O halde iki minimal fonksiyon

Tablo 4.3. Örnek fonksiyona ilişkin asal bileşenler tablosu.

			x'_1	x'_1	x'_1	x'_1	x'_1	x_1	x_1	x_1	x_1	x_1	x_1
			x'_2	x'_2	x'_2	x_2	x_2	x'_2	x'	x'_2	x_2	x_2	x_2
			x'_3	x'_3	x_3	x'_3	x_3	x'_3	x'	x_3	x'_3	x_3	x_3
			x'_4	x_4	x'_4	x_4	x_4	x'_4	x'	x'_4	x_4	x_4	x_4
			0	1	2	5	7	8	x_4	10	13	15	
			$\checkmark$		$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	9	$\checkmark$	$\checkmark$	$\checkmark$	
Temel satır	0,1,8,9	$x'_2x'_3$	x	x				x	x				
	0,2,8,10 $\checkmark$	$x'_2x'_4$	x		$\otimes$			x		$\otimes$			
Temel satır	1,5,9,13	x'_3x_4		x		x			x		x		
	5,7,13,15 $\checkmark$	x_2x_4				x	$\otimes$				x	$\otimes$	

$$f = x_2x_4 + x'_2x'_4 + x'_3x_4 \text{ ve}$$

$$f = x_2x_4 + x'_2x'_4 + x'_2x'_3$$

dür.

4.2.4 Bütün İndirgenemeyen ve minimal fonksiyonların belirlenmesi

Asal bileşenler tablosunda, temel asal bileşenlerin örttükleri doğru minterimlerin bulunduğu sütunlar ile temel asal bileşenlerin bulunduğu satırların silinmesiyle bulunan tabloya indirgenmiş asal bileşenler tablosu diyeceğiz.

Tablo 4.4 te $f = \sum 0,1,3,4,7,13,15,19,20,22,23,29,31$ fonksiyonu için asal bileşenler tablosu verilmiştir. MSB (En çok ağırlıklı bit) x_1 değişkenine, LSB (En az ağırlıklı bit) ise x_5 değişkenine karşı düşürülmüştür.

Tablo.4.4 Örnek fonksiyona ilişkin asal bileşenler tablosu.

	0	1	3 ✓	4	7 ✓	13 ✓	15 ✓	19 ✓	20	22	23 ✓	29 ✓	31 ✓
$A=x_2x_3x_5$						⊗	x					⊗	x
$B=x_3x_4x_5$					x		x				x		x
$C=x'_2x_4x_5$			x		x			⊗			x		
$D=x_1x'_2x_3x_4$										x	x		
$E=x_1x'_2x_3x'_5$									x	x			
$F=x'_2x_3x'_4x'_5$				x					x				
$G=x'_1x'_2x'_3x_5$		x	x										
$H=x'_1x'_2x'_4x'_5$	x			x									
$I=x'_1x'_2x'_3x'_4$	x	x											

Görüldüğü gibi bu tabloda A ve C temel asal bileşenlerdir, Temel asal bileşenlerin örttükleri doğru minterimler ise 3, 7, 13, 15, 19, 23, 29, 31 dir. Temel asal bileşenlerin bulundukları satırların, bu temel asal bileşenlerin örttükleri doğru minterimleri bulundukları sütunların, silinmesiyle bulunan, indirgenmiş asal bileşenler tablosu Tablo 4.5 te verilmiştir.

Tablo.4.5.Örnek fonksiyona ilişkin indirgenmiş asal bileşen tablosu.

	0	1	4	20	22
D					x
E				x	x
F			x	x	
G		x			
H	x		x		
I	x	x			

İndirgenmiş asal bileşenler tablosundan, minimal fonksiyonların bulunmasının genel bir yolu, indirgenmiş tabloda, asal bileşenleri değişken alan, bir önermeler fonksiyonunun tanımlanmasıdır. 3. Bölümde, belli koşulların sağlanmasına karşı düşen, çarpımlar toplamı biçiminde olan, örnek bir önerme fonksiyonu görmüştük. Burada ise, toplamlar çarpımı biçiminde görünen, bir p önerme fonksiyonunu elde edeceğiz.

İndirgenmiş asal bileşenler tablosundaki, f indirgeyeceğimiz fonksiyona ait bir doğru minterimin, asal bileşenler tarafından örtülmesini, toplamlar terimi olarak düşünebiliriz. Öyle ki ilgilenilen doğru minterimin, sütunundaki (x) lara karşı düşen, bir ve/veya daha fazla asal bileşen değişkeninin (1) olması, bu doğru minterimin örtüldüğünü gösterir.

Örneğin Tablo 4.5 teki (0) doğru minteriminin örtülmesini, (H+I) ile ifade edebiliriz. Bu demektir ki H ve I dan herhangi biri veya ikisi de (1) olduğunda, (0) doğru minterimi örtülür, aksi taktirde $H+I=0$ olur ve (0) doğru minterimi örtülmez.

İndirgenmiş asal bileşenler tablosundaki, bütün doğru minterimler için benzer toplamlar terimleri oluşturulur. Minimal fonksiyonda bulunan asal bileşenleri belirleyen p önerme fonksiyonunun, söz konusu edilen toplamlar terimlerinin çarpımı biçiminde olması gerekir. Ancak o zaman, indirgenmiş asal bileşenler tablosundaki, bütün doğru minterimler örtülmüş olur.

Buna göre Tablo 4.5 teki indirgenmiş asal bileşenler tablosu için p önerme fonksiyonu

$$P = (H+I) (G+I) (F+H) (E+F) (D+E)$$

bulunur. p önerme fonksiyonu çarpımlar toplamı biçiminde düzenlenirse

$$p = EHI+EFI+DFI+EGH+DFGH$$

olur. Bu ifadeden indirgenmiş asal bileşenler tablosunun kapsanabilmesi için en az üç satıra gereksinim olduğu anlaşılmaktadır. Örneğin E,H,I, satırlarındaki asal bileşenler, tablodaki bütün doğru minterimleri kapsar. p deki her bir çarpımlar terimini, aradığımız asal bileşen grubu alarak, A ve C temel asal bileşenlerini de ekleyip 5 farklı, indirgenemeyen fonksiyon bulunur. p nin ifadesindeki sonuncu terimdeki değişken sayısı diğerlerinden farklı olduğu için, indirgenemeyen 5 fonksiyon arasından aşağıdaki 4 tanesi minimaldir.

$$f(x_1x_2x_3x_4x_5) = A+C+E+H+I = x_2x_3x_5 + x'_2x_4x_5 + x_1x'_2x_3x'_5 + x'_1x'_2x'_4x'_5 + x'_1x'_2x'_3x'_4$$

$$f(x_1x_2x_3x_4x_5) = A+C+E+F+I = x_2x_3x_5 + x'_2x_4x_5 + x_1x'_2x_3x'_5 + x'_2x_3x_4x'_5 + x'_1x'_2x'_3x'_4$$

$$f(x_1x_2x_3x_4x_5) = A+C+D+F+I = x_2x_3x_5 + x'_2x_4x_5 + x_1x'_2x_3x_4 + x'_1x'_2x'_4x'_5 + x'_1x'_2x'_3x'_4$$

$$f(x_1x_2x_3x_4x_5) = A+C+E+G+H = x_2x_3x_5 + x'_2x_4x_5 + x_1x'_2x_3x'_5 + x'_1x'_2x'_3x'_5 + x'_1x'_2x'_4x'_5$$

Bu yöntemle bütün minimal fonksiyonlar ve indirgenemeyen fonksiyonlar bulunmaktadır.

Göz önünde tutulması gereken bir konu, minimal fonksiyonu, önermeler fonksiyonundan yararlanarak bulurken, seçilen asal bileşenlerdeki değişken sayısı minimum olmalıdır. Örneğin D, E, H, I, G asal bileşenlerinden, varsayalım G nin değişken sayısı 4 değilde 3 olsaydı, yukarıda bulduğumuz 4 minimal fonksiyondan, yalnızca sonuncusu minimal olurdu, diğerleri olmazdı.

Şimdiye kadar, asal bileşen tablosunun incelenerek, bütün minimal fonksiyonların nasıl bulunacağı bir örnek üzerinde verildi. Şimdi de asal bileşen tablosu karmaşık olduğu zaman ve bütün minimal fonksiyonları değil de, yalnızca bir tanesinin bulunması istendiğinde kullanılabilecek bir yöntem vereceğiz. Bu amaçla tutulacak sistematik bir yol, örnek üzerinden verilecektir. Örnek fonksiyonumuz:

$f(x_1x_2x_3x_4x_5) = \sum 1,3,4,5,6,7,10,11,12,13,14,15,18,19,20,21,22,23,25,26,27$ ve buna ilişkin asal bileşen tablosu Tablo 4.6a da verilmiştir.

Tablo 4.6 Örnek fonksiyon ve ona ilişkin asal bileşen ve indirgenmiş asal bileşen tabloları

	1	3	4	5	6	7	10	11	12	13	14	15	18	19	20	21	22	23	25	26	27
	✓	✓	✓	✓	✓	✓			✓	✓	✓	✓			✓	✓	✓	✓	✓		✓
$\sqrt{A=x'_2x_3}$			x	x	x	x									⊗	⊗	x	x			
$\sqrt{B=x'_1x_3}$			x	x	x	x			⊗	⊗	x	x									
$C=x'_1x'_3x_4$													x	x						x	x
$D=x'_1x'_2x_4$													x	x			x	x			
$E=x'_2x'_3x_4$							x	x												x	x
$F=x'_1x'_2x_4$							x	x			x	x									
$G=x'_3x_4x_5$		x						x						x							x
$H=x'_2x_4x_5$		x				x								x				x			
$I=x'_1x_4x_5$		x				x		x				x									
$\sqrt{J=x'_1x'_2x_5}$	⊗	x		x		x															
$\sqrt{K=x'_1x_2x'_3x_5}$																			⊗		x

a) Asal bileşenler tablosu.

	10	11	18	19	26
C			x	x	x
D			x	x	
E	x	x			x
F	x	x			
G		x		x	
H				x	
I		x			

b) İndirgenmiş asal bileşenler tablosu.

	10	11	18	19	26
C			x	x	x
E	x	x			
G		x		x	

c) Satır indirgemesi yapılmış asal bileşenler tablosu

Temel asal bileşenler A,B,J ve K olup, bunların örttüğü minterimler ✓ ile işaretlenmiştir. Temel asal bileşenlere ilişkin satırlar ve örtükleri doğru minterim sütunlarının kaldırılmasıyla elde edilen indirgenmiş tablo Tablo 4.6 b de verilmiştir. Bu tabloda temel asal bileşen satırı yoktur ama, bazı satırları kaldırmak olasıdır. örneğin H satırı yalnız 19 da x bulundururken G satırı 11 ve 19 da x bulundurmaktadır. H ve G nin uzunlukları aynı olduğuna ve yalnızca bir minimal fonksiyon bulmak istendiğine göre H satırını kaldırabiliriz. Çünkü G asal bileşeni K nın örttüğü 19 doğru mintermini de örtmektedir.

Genel olarak, bir V asal bileşeninin örttüğü, bütün doğru minterimler; bir U asal bileşenince de örtülüyorsa, U asal bileşenine karşı gelen satır, V asal bileşenine karşı gelen satırı örtüyor denir. Gereksiz olan V asal bileşenine ilişkin satır kaldırılır. Tersii doğru değildir.

Örnekte G, H' yı; E, F ve I' yı; C, D' yi kapsadığından ve hepsi de aynı değişken sayısında olduğundan H, F, I ve D satırları kaldırılarak, Tablo 4.6 c deki, sonuç indirgenmiş asal bileşenler tablosu bulunur. Bu tabloda C ve E temel asal bileşenlerdir, ve minimal ifade de görülmek zorundadır. Bütün (10,11,18,19,26) doğru minterimler C ve E tarafından örtüldüğünden minimal fonksiyon, daha önce bulduğumuz temel asal bileşenlerle

$$f(x_1x_2x_3x_4x_5) = x'_2x_3 + x'_1x_3 + x'_1x'_2x_5 + x_1x_2x'_3x_5 + x_1x'_3x_4 + x_2x'_3x_4 \text{ dır.}$$

İndirgenmiş asal bileşenler tablosunda, sütunların da indirgenmesi söz konusudur. örnekte 10. sütunundaki doğru minterimin örtülmesi için E veya F asal bileşenleri seçilmek zorundadır. 11. sütunundaki doğru minterim de E ve F tarafından örtüldüğünden, 11 sütunu silinebilir.

Bir asal bileşenler tablosunun i. sütununda x olan her yerde, j sütununda da x varsa j sütunu i sütununu örter denir. Bu i ve j sütunlarının birlikte bulunduğu bir tablodan elde edilen minimal fonksiyon ile, yalnızca i. sütununu bulunduran indirgenmiş tablodan elde edilen minimal fonksiyon aynıdır. Çünkü i. sütundaki doğru minterim düşünülürken, aynı zamanda j. sütundaki doğru minterim de düşünülmüş olmaktadır. O halde j sütunu silinebilir. İndirgenmiş asal bileşenler tablosunda bir örten sütunlar varsa, bu sütun silinebilir.

Satır veya sütun indirgemesinin sırasının değişmesi, bir minimal fonksiyonun bulunması açısından, sonucu etkilemez. O nedenle, satır sütun indirgemeleri (silmeleri) yaparak, basitleşen asal bileşen tablosunda, doğru minterimnleri örten minimum sayıda asal bileşen seçimi kolayca yapılabilir.

Tablo yöntemiyle bir minimal fonksiyonun bulunması adım adım aşağıdaki gibi özetlenebilir.

1. Doğru olan minterimlere ilişkin iki tabanındaki sayılar, bulundurdıkları 1 sayısına göre gruplandırılır. Aynı sayıda 1 bulunduran iki tabanındaki sayılar aynı gruptadırlar. Bu gruplar içerilerindeki 1 sayısına göre sıralandırılırlar.
2. i. gruptaki iki tabanındaki herhangi bir sayı, (i+1) gruptaki herhangi bir sayıyla sıra ile karşılaştırılır. Bu karşılaştırılan sayılar birleştirilebiliyorsa, birleştirilir ve $\checkmark$ ile işaretlenir. Kaybolan değişken yerine - işareti konulur. Karşılaştırmaya artık birleştirmenin olası olmadığı duruma kadar devam edilir.
3. Sonuçta birleştirilmemiş $\checkmark$ ile işaretli olmayan bütün terimler asal bileşenlerdir. Bunlara göre asal bileşenler tablosu düzenlenir.
4. Sütununda yalnızca bir tane x işareti bulunan doğru minterimleri örten asal bileşenler, temel asal bileşenlerdir. Bunların örttükleri bütün doğru minterimler $\checkmark$ ile işaretlenir.
5. Temel asal bileşenlerin örttükleri doğru minterimlerin dışında kalan doğru minterimleri ($\checkmark$ ile işaretlenmemiş olanlar), örten minimum sayıda asal bileşen, hemen görülüyorsa, bunlar seçilerek minimal fonksiyon(lar) bulunur.
6. Aksi halde yukarıda anlatılan önermeler cebri kullanılarak bütün minimal fonksiyonlar bulunur.
7. Yalnızca bir minimal fonksiyonun bulunması yeterli oluyorsa, satır ve sütun indirgemesi yaparak minimal fonksiyon bulmaktır.
 - $\checkmark$ ile işaretli doğru minterimlerin bulunduğu sütunlar ve temel asal bileşen satırları silinerek indirgenmiş asal bileşenler tablosu bulunur.
 - İndirgenmiş asal bileşenler tablosunda, bir satır bir başka satırı örtüyorsa, örtülen satır silinir.

- İndirgenmiş asal bileşenler tablosunda, bir sütunun örttüğü bir başka sütun varsa, örten sütun silinerek sonuç indirgenmiş tablo bulunur.
- Sonuç tabloda bulunan doğru minterimlerin tamamını örten, minimum sayıda ve minimum değişkenli asal bileşenler seçilir.

8. 7. adımda seçilen asal bileşenlerle, 4. adımda bulunan temel asal bileşenlerin toplamı minimal fonksiyonu verir.

Her ne kadar bu bölümde, yalnızca 1. tip kanonik açılımı verilen fonksiyonların, minimalleştirilmesi esas alınmışsa da, benzer akıl yürütme, 2. tip kanonik açılımı verilen fonksiyonlar için de yapılabilir. Bu durumda doğru olan maxterimler düşünülür. Birleştirme işlemi ise

$$(A+a)(A+a') = A+aa' = A$$

olarak alınır.

Örneğin Tablo 4.6 da verilen fonksiyon

$$g = \sum 1,3,4,5,6,7,10,11,12,13,14,15,18,19,20,21,22,23,25,26,27$$

biçiminde verilseydi, örnekteki A,B,J,K temel asal bileşenler ve C,E asal bileşenlerine, sırasıyla $(x_2 + x'_3)$, $(x'_2 + x_3)$, $(x_1 + x_2 + x'_5)$, $(x'_1 + x'_2 + x_3 + x'_5)$, $(x'_1 + x_3 + x'_4)$, $(x'_2 + x_3 + x'_4)$ toplamlar terimleri karşı düşecekti. Minimal fonksiyon da bu terimlerin çarpımı yani

$$g = (x_2 + x'_3) \cdot (x'_2 + x_3) \cdot (x_1 + x_2 + x'_5) \cdot (x'_1 + x'_2 + x_3 + x'_5) \cdot (x'_1 + x_3 + x'_4) \cdot (x'_2 + x_3 + x'_4)$$

olacaktı.

Burada dikkat edilmelidir ki toplamlar çarpımı biçiminde bulunan g fonksiyonu , daha önce çarpımlar toplamı biçiminde bulunan f fonksiyonunun tümleyenidir. Başlangıçta verilen f fonksiyonunun 2. tip kanonik açılımına ilişkin minimal fonksiyonu bulmak için

$$f = \prod 0,2,8,9,16,17,24,28,29,30,31$$

$$(=\sum 1,3,4,5,6,7,10,11,12,13,14,15,18,19,20,21,22,23,25,26,27)$$

düşünmek gerekirdi.

4.3 Karnaugh Diyagramlarıyla Fonksiyonların minimalleştirilmesi

Daha öncede belirtildiği gibi, az değişkenli fonksiyonlar için, minimal ifadelerin daha hızlı bulunması yöntemi, Karnaugh diyagramı yöntemidir. Şimdi bu yöntemi inceleyelim.

4.3.1 Karnaugh diyagramları

Dört veya beş değişkenliye kadar fonksiyonların minimalleştirilmesini hızlandıran bir yöntem, Karnaugh diyagramı yöntemidir. Karnaugh diyagramı, Boole fonksiyonun doğruluk tablosunun, amaca yardımcı olacak biçimde, düzenlenmesidir. Üç ve dört değişkenli fonksiyonlar için Karnaugh diyagramları Tablo 4.7 de verilmiştir.

Tablo 4.7 Üç ve dört değişkenli fonksiyonlara ilişkin Karnaugh diyagramları.

		x_1x_2			
		00	01	11	10
x_3	0	0	2	6	4
	1	1	3	7	5

		x_1x_2			
		00	01	11	10
x_3x_4	00	0	4	12	8
	01	1	5	13	9
	11	3	7	15	11
	10	2	6	14	10

Üç değişkenli fonksiyonlara ilişkin diyagramlarda, sütunlar x_1x_2 bağımsız değişkenlerinin ikili değer permütasyonlarını (00, 01, 11, 10), satırlar ise x_3 bağımsız değişkeninin değerlerini (0,1) göstermektedir. i. satır ile j. sütunun kesiştiği kareye, sütun ve satırlar permütasyonlarının oluşturduğu tanım elemanına karşı düşen, fonksiyonun değeri yazılır. örneğin 3. sütun ve 2. satırın kesiştiği yere $f(x_1x_2x_3) = f(1,1,1)$ için fonksiyonun aldığı değer yazılır. 4 değişkenli diyagramın da, bir fonksiyonu temsil etmesi, benzer biçimde yapılır. Fonksiyonun, diyagrama taşınmasında kolaylık olsun diye, karelerin içine, sütun satır permütasyonu olan, iki tabanındaki sayının, on tabanındaki karşılığı geçici olarak yazılmıştır. Bu sayılar her zaman yazılmaz, karelere yalnızca fonksiyonun aldığı değerler yazılır. örneğin Tablo 4.8 de doğruluk tablosu ile verilen fonksiyonlar, yanlarında Karnaugh diyagramlarıyla temsil edilmişlerdir.

Tablo 4.8 a) Üç değişkenli $f(x_1x_2x_3) = \sum 1,2,3,7 = x'_1x'_2x_3 + x'_1x_2x'_3 + x'_1x_2x_3 + x_1x_2x_3$ fonksiyonun Karnaugh diyagramında gösterilişi.

$x_1x_2x_3$	f
0 0 0 0	0
1 0 0 1	1
2 0 1 0	1
3 0 1 1	1
4 1 0 0	0
5 1 0 1	0
6 1 1 0	0
7 1 1 1	1

x_1x_2	00	01	11	10
0	0	1 ²	6	4
1	1 ¹	1 ³	1 ⁷	5

Tablo 4.8.b) Dört değişkenli

$f(x_1x_2x_3x_4) = \sum 2,3,7,11,12,15 = x'_1x'_2x_3x'_4 + x'_1x'_2x_3x_4 + x'_1x_2x_3x_4 + x_1x'_2x_3x_4 + x_1x_2x'_3x_4 + x_1x_2x_3x_4$
Fonksiyonunun doğruluk tablosu ve Karnaugh diyagramı

$x_1x_2x_3x_4$	f
0 0 0 0	0
1 0 0 0	0
2 0 0 1	0
3 0 0 1	0
4 0 1 0	1
5 0 1 0	1
6 0 1 1	1
7 0 1 1	1
8 1 0 0	0
9 1 0 0	0
10 1 0 1	0
11 1 0 1	0
12 1 1 0	1
13 1 1 0	1
14 1 1 1	0
15 1 1 1	1

x_1x_2	00	01	11	10
00	0	4	12	8
01	1	5	13	9
11	1 ³	1 ⁷	1 ¹⁵	1 ¹¹
10	1 ²	6	14	10

↓
 $x'_1x'_2x_3x'_4$

Görüldüğü gibi 1. Fonksiyonun, 1 olduğu (1,2,3,7) karelerine 1, diğerlerine 0 konmuş; 2. fonksiyonun da, 1 olduğu (2,3,7, 11,12,15) karelerine 1, diğer karelere de 0 konmuştur. Yaygın kullanılan biçim, fonksiyonun 1 olduğu karelerin doldurulması ve 0 olduğu karelerin boş bırakılmasıdır.

Karnaugh diyagramlarındaki kare sayısı, $f : S^n \rightarrow S$ e bir fonksiyon için, S^n kümesindeki eleman sayısı (2^n) kadardır. Örneğin 3 değişkenli fonksiyonlara ilişkin Karnaugh diyagramlarında bu sayı, $2^3 = 8$, 4 değişkenli için $2^4 = 16$ dır.

Doğruluk tablosu verilen bir fonksiyonu, Karnaugh diyagramına aktarmak kolay olduğu gibi; diyagramdan, fonksiyona veya doğruluk tablosuna geçmekte kolaydır. Şöyle ki diyagramda fonksiyonun 1 değerini aldığı, doğru minterimler, bulundukları kareye ilişkin permütasyonlardan (satır, sütun) yararlanılarak bulunur: Eğer bir basamakta 1 sayamağı varsa; o basamağa, ilgili değişkenin kendisi, aksi halde tümleyeni yazılır. Bu doğru minterimlerin toplamı fonksiyonu verir (Tablo 4.8).

Karnaugh diyagramlarının sütun ve satırlarının kodlanmasındaki sıra önemlidir. Ortak kenarları olan karelere ilişkin minterimlerde, bütün değişkenler, bir değişken dışında aynıdır, birleştirilebilirler. Böyle karelere “**komşu kareler**” denir. Diyagramın kenar karelerinin komşuları, diyagramı önce bir silindirik biçimine dönüştürüp sonra alt ve üst tabanları birleştirilerek elde edilen tor yüzeyi düşünülerek, daha kolayca görülebilir. Bu nedendir ki, komşu sütunların kodlarında, yalnızca bir basamaktaki sayamaklar farklıdır, diğerleri aynıdır. Tablo 4. 9 a, b ve c de sırasıyla m_7 , m_6 , ve m_{10} a komşu minterimler (kareler) gösterilmiştir.

Diyagramdaki bir kareye, komşu kareler sayısı, değişken sayısı kadardır. örnekte görüldüğü gibi m_7 , m_6 , m_{10} a komşu kareler sırasıyla m_3 , m_5 , m_6 , m_{15} ; m_2 , m_4 , m_7 , m_{14} ; m_2 , m_8 , m_{11} , m_{14} dır.

Tablo 4.9.Dört değişkenli fonksiyona ilişkin Karnaugh diyagramındaki komşu karelere örnekler.

$x_3x_4 \backslash x_1x_2$	00	01	11	10
00				
01		m_5		
11	m_3	m_7	m_{15}	
10		m_6		

$x_3x_4 \backslash x_1x_2$	00	01	11	10
00		m_4		
01				
11		m_7		
10	m_2	m_6	m_{14}	

$x_3x_4 \backslash x_1x_2$	00	01	11	10
00				m_8
01				
11				m_{11}
10	m_2		m_{14}	m_{10}

4.3.2 Karnaugh diyagramlarında küpler ve bunlara karşı düşen çarpımlar(toplamlar) terimleri

Komşu iki karede, fonksiyon 1 değerini alıyorsa, bu doğru minterimler birleştirilerek kısaltılırlar. Örneğin m_6 ve m_7 de fonksiyon 1 değerini alıyorsa, iki doğru minterim yerine

$$x'_1x_2x_3x'_4 + x'_1x_2x_3x_4 = x'_1x_2x_3$$

bir çarpımlar terimi yazılabilir.

Tanım 4.6. **m-mertebeden alt küp**: Her biri m tane kareye komşu, 1 (0) değerini alan karelerden oluşmuş, 2^m kareye m . mertebeden alt küp denir

Örneğin Tablo 4.10 a da 1'lerden oluşmuş her bir kare, iki ayrı kareyle komşudur ve 4 ($2^2 = 2^2 = 4$) tanedir. O nedenle bu dört 1, 2. mertebeden alt küp oluşturur.

Her m. mertebeden alt küp, n bağımsız değişken sayısı olmak üzere, (n-m) değişkenli bir çarpımlar terimiyle ifade edilebilir. Çünkü alt küpte bulunan 2^m terimde, ortak (n-m) değişken parçası ile m değişkenin bütün permütasyonları görülür. Tablo 4.10 a da Karnaugh diyagramı ile verilen fonksiyon için, x_1 ve x_4 değişkenleri bütün karelere ilişkin terimlerde ortaktır. x_2 ve x_3 ün bütün permütasyonları da söz konusu terimlerde görülür. Yani

$$x'_1x'_2x'_3x_4 + x'_1x'_2x_3x_4 + x'_1x_2x'_3x_4 + x'_1x_2x_3x_4 = x'_1x_4(x'_2x'_3 + x'_2x_3 + x_2x'_3 + x_2x_3) = x'_1x_4 \text{ dır.}$$

Parantez içindeki ifade, sanki bütün minterimleri doğru olan, bir 2 değişkenli fonksiyon -1.tip kanonik açılımında bütün terimleri 1 olan- gibi düşünülebilir. Bu 2 değişkenli fonksiyon, 1 e eşit, bir sabittir.

Bir alt küpe karşı düşen çarpımlar terimlerinin bulunması:

Alt küpe ilişkin doğru minterimler için

- x_i değişkeninin aldığı değerler, hem 0 hem de 1 olabiliyorsa, x_i değişkeni çarpımlar teriminde görülmez.
- x_i değişkeninin aldığı değerler değişmeyip, hep 1 kalıyorsa, bu değişken x_i olarak çarpımlar teriminde görülür.
- x_i değişkeninin aldığı değerler değişmeyip, hep 0 kalıyorsa, bu değişken x'_i olarak çarpımlar teriminde görülür.

Örneğin Tablo 4.10. daki 1. Karnaugh diyagramındaki alt küpteki doğru minterimler için x_1 değişkeni, hep 0 değerini almaktadır ve x'_1 , çarpımlar terimi x'_1x_4 te görülür. x_2 ve x_3 değişkenleri, bu alt küpteki doğru minterimler için, hem 0 hem de 1 değerini aldıkları için, kaybolurlar, yani çarpımlar teriminde görülmezler. x_4 değişkeni ise, söz konusu alt küpteki doğru minterimler için, yalnızca 1 değerini almaktadır ve çarpımlar teriminde, x_4 olarak görülür.

Tablo 4.10 a, b, c de 2, 1 ve 3 mertebeden küpler ve ilgili çarpımlar terimleri gösterilmiştir.

Tablo 4.10. Çeşitli mertebeden alt küpler ve onlara karşı gelen çarpımlar terimlerine örnekler.

x_1x_2	00	01	11	10
x_3x_4 00				
01	1	1		
11	1	1		
10				
	x'_1x_4			

x_1x_2	00	01	11	10
x_3x_4 00		1		
01		1		
11				
10	1			1
	$x'_1x_2x'_3$		$x'_2x_3x_4$	

x_1x_2	00	01	11	10
x_3x_4 00	1		1	1
01	1		1	1
11	1		1	1
10	1		1	1
	x'_2		x'_1	

4.3.3 Karnaugh diyagramlarıyla minimal fonksiyon(ların) bulunması

Karnaugh diyagramıyla verilen bir fonksiyona eşdeğer, minimal fonksiyon, seçilmiş çeşitli alt küplerden yararlanılarak bulunabilir. Örneğin Tablo 4.11 de karnaugh diyagramıyla verilen $f = x'_1x'_2x_3 + x'_1x'_2x'_3 + x_1x'_2x'_3$ fonksiyonu, iki 1. mertebeden alt küp

için $f = x_1'x_2' + x_2'x_3$ minimal fonksiyonuna dönüşür. Çünkü $a + a = a$ özelliğinden yararlanılarak

$$\begin{aligned} f &= x_1'x_2'x_3 + x_1'x_2'x_3' + x_1x_2'x_3 \\ &= x_1'x_2'x_3 + x_1'x_2'x_3' + x_1'x_2'x_3 + x_1x_2'x_3 \\ &= x_1'x_2'(x_3 + x_3') + x_2'x_3(x_1 + x_1') \\ &= x_1'x_2' + x_2'x_3 \end{aligned}$$

olur.

Tablo.4.11. Örnek fonksiyon.

x_1x_2	00	01	11	10
x_3				
0	1			1
1	1			

$\rightarrow x_2'x_3$
 $\rightarrow x_1'x_2'$

Aynı bir doğru minterimin birden fazla alt küpte görülmesi, $a+a=a$ teoremi dolayısıyla, olasıdır ve yararlı olabilir. Görüldüğü gibi eşdeğer bir fonksiyon , 1 lerden oluşan alt küplere karşı düşen çarpımlar terimleri toplamı biçiminde yazılabilir. Çarpımlar terimleri sayısı, seçilen alt küpler sayısı kadardır. Her bir çarpımlar terimindeki değişken sayısı da, ilgili alt küpün mertebesiyle belirlenir. O halde minimal fonksiyon, minimum sayıda, olası en büyük mertebeden, bütün 1 li kareleri örten alt küpler seçilerek bulunur. Daha büyük bir alt küp tarafından kapsanan, bir alt küp, hiç bir zaman, minimal fonksiyon için seçilmemelidir.

Karnaugh diyagramlarından yararlanılarak minimal fonksiyonun bulunması işlemi adım adım şöyle özetlenebilir.

1. Hiç bir alt küpe veya 0. Mertebeden altküpe giren, kendisi) girmeyen doğru minterimler temel asal bileşenlerdir ve minimal ifadeye alınırlar.

2. Bir doğru minterimin, birinci mertebeden yalnızca bir alt küpe girdiğini varsayalım. Ayrıca bu doğru minterim, daha büyük mertebeden bir başka alt küp tarafından örtülmesin. Böyle doğru minterimlere ilişkin, 1. Mertebe alt küpe karşı düşen, birleştirilmiş çarpımlar terimleri, bir temel asal bileşendir ve minimal fonksiyona alınırlar.

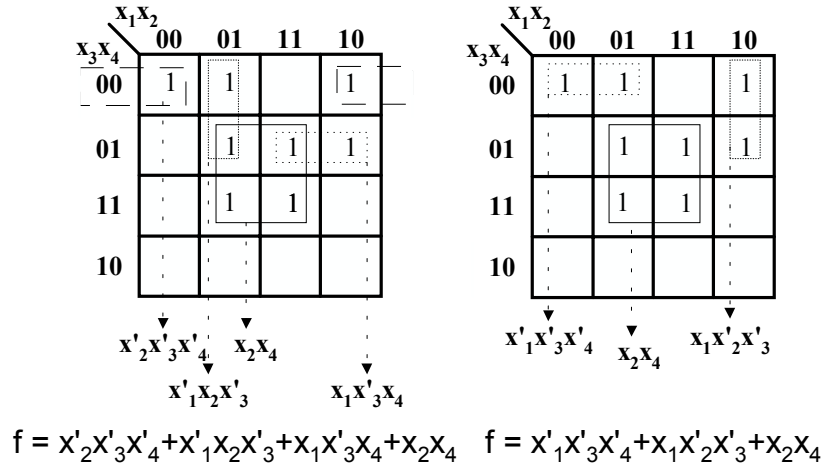
3. Bir doğru minterimin, ikinci mertebeden yalnızca bir alt küpe girdiğini varsayalım. Ayrıca bu doğru minterim, daha büyük mertebeden bir başka alt küp tarafından örtülmesin. Böyle doğru minterimlere ilişkin, 2. Mertebe alt küpe karşı düşen, birleştirilmiş çarpımlar terimleri, bir temel asal bileşendir ve minimal fonksiyona alınırlar.

4. Üçüncü ve dördüncü mertebeden alt küplere ilişkin temel asal bileşenler varsa ,minimal fonksiyona, 2 ve 3. adımlarda olduğu gibi, alınırlar.

5. Temel asal bileşenlere karşı düşen alt küplerin alınmasından sonra, henüz örtülmemiş doğru minterimleri örten , minimum sayıda, maksimum mertebeli asal bileşenler ilave edilerek minimal fonksiyon bulunur. Fonksiyonun bütün doğru minterimlerinin, ilave asal bileşenlerle, örtülmüş olması gerekir; aksi taktirde bulunan minimal fonksiyon, verilen fonksiyona eşdeğer olmaz.

Yöntemin daha iyi anlaşılabilmesi için aşağıda iki örnek verilmiştir. Tablo 4.12 de Karnaugh diyagramıyla verilen fonksiyon, ilk diyagramda seçilen alt küpler için

Tablo.4.12. Aynı bir fonksiyona ilişkin farklı indirgemeler.



aşağıdaki biçimde

$$f = x_2'x_3x_4 + x_1x_2x_3 + x_1x_3x_4 + x_2x_4$$

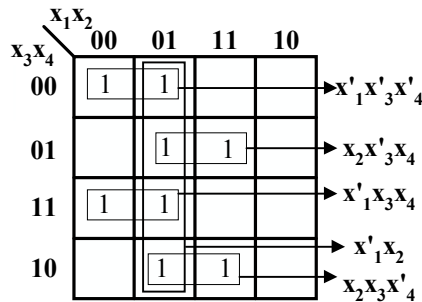
yazılabilir.

Aynı fonksiyon 2. tabloda gösterilen alt küpler için minimal olarak

$$f = x_1'x_3x_4 + x_1x_2'x_3 + x_2x_4$$

biçiminde elde edilir. Adım adım tutulacak yolla da bu sonuca varılabilir. Örneğin $f(0111) = 1$ ve $f(1111) = 1$ doğru minterimleri yalnızca (x_2x_4) ikinci mertebeden alt küpü tarafından örtülmektedir ve başka daha yüksek mertebeden bir alt küp tarafından örtülmemektedir. O nedenle (x_2x_4) temel asal bileşendir ve minimal fonksiyonda görülmelidir. Bu temel asal bileşene ilave etmemiz gereken, minimum sayıdaki, en yüksek mertebeden alt küplere karşı düşen asal bileşenler $x_1'x_3x_4$, $x_1x_2'x_3$ dür.

Tablo.4.13. Örnek fonksiyon.



Tablo 4.13 deki örnekte de hemen büyük mertebeli alt küplerin seçilmemesi gerektiği gösterilmektedir. Adım adım gidildiğinde çizikli olarak gösterilen alt küpten gelen x_1x_2 teriminin gereksiz oluşu anlaşılır. Yalnızca temel asal bileşenlerden minimal fonksiyon

$$f = x_1'x_3x_4 + x_2x_3x_4 + x_1'x_3x_4 + x_1'x_2$$

olarak bulunur.

Bir fonksiyonun, çarpımlar toplamı için verilen bu yöntem, benzer biçimde, toplamlar çarpımına da uygulanabilir. Bu halde fonksiyonun 0 değerini aldığı doğru maxterimler

için alt küpler düşünülür. Tablo 4.14 de verilen fonksiyon, gösterilen alt küpler için toplamlar çarpımı biçiminde minimal olarak

$$f = (x'_1 + x'_2 + x_4) (x_1 + x_2 + x'_4) (x_2 + x'_3) (x'_3 + x_4)$$

elde edilir. İkinci tip kanonik açınımına ilişkin minimal fonksiyon bulunurken, (0) lardan oluşan bir alt küpe karşı gelen toplamlar çarpımı, birinci tip kanonik açınım dakine benzerdir. Alt küpe ilişkin kareler için değişken (0) olarak sabit kalıyorsa, toplamlar teriminde bu değişkenin kendisi, (1) olarak sabit kalıyorsa, toplamlar teriminde bu değişkenin tümleyeni görünür. Birici kanonik açınım için durum tersi idi.

Tablo.4.14. 2. tip açınımına ilişkin minimal fonksiyona örnek.

$x_1 x_2$		00	01	11	10	
$x_3 x_4$	00			0		$x'_1 + x'_2 + x_4$
	01	0				$x_1 + x_2 + x'_4$
	11	0			0	$x_2 + x'_3 + x'_4$
	10	0	0	0	0	$x'_3 + x_4$

1. ve 2. Tip kanonik açınımlara ilişkin minimal fonksiyonlarının hangisinin tercih edileceği genelleştirilemez.

4.3.4 Kısmen belirlenmiş fonksiyonlar ve karnaugh diyagramlarıyla minimalleştirilmeleri

Tanım 4.7. Kısmen belirlenmiş fonksiyonlar (Don't care functions): $S^n \rightarrow S$ e bir fonksiyon, tanım kümesi S^n deki bazı elemanlar için, S kümesindeki herhangi bir elemanı alabiliyorsa, bu fonksiyona kısmen belirlenmiştir denir. Bir başka deyişle fonksiyon bazı tanım elemanı için keyfi (0 da 1 de olabiliyorsa) ise, bu fonksiyona kısmen belirlenmiştir denir. İkinci bölümde tanımlanan tamamen belirlenmiş bağıntı ile bu tanım karıştırılmamalıdır. Kısmen belirlenmiş fonksiyon dememize rağmen fonksiyon tamamen belirlenmiş bir bağıntıdır ancak bazı domen elemanları için fonksiyonun aldığı değer 0 da 1 de olabilir(keyfi); bir başka deyişle keyfi sayısı k ise kısmen belirlenmiş fonksiyondan, olası 2^k fonksiyonlardan herhangi biri kastedilmektedir.

Kısmen belirlenmiş fonksiyonlar, belirlenmiş doğru mintermlerle, keyfi minterimler ayrı iki grupta belirtilerek gösterilir. örneğin

$$f_1 = \sum 1, 3, 5 + \sum_k 7, 8, 12, 14 \text{ ve}$$

$$f_2 = \prod 3, 7, 9 . \prod_k 5, 13, 14, 15$$

ile verilen fonksiyonlara ilişkin Karnaugh diyagramları Tablo.4.15 de verildiği gibidir.

Tablo.4.15. Kısmen belirlenmiş fonksiyonlara örnek.

$x_3x_4 \backslash x_1x_2$	00	01	11	10
00			k	k
01	1	k		
11	1	k		
10			k	

$x_3x_4 \backslash x_1x_2$	00	01	11	10
00				
01		k	k	0
11	0	0	k	
10			k	

$$f_1 = \sum 1, 3, 5 + \sum_k 7, 8, 12, 14$$

$$f_2 = \prod 3, 7, 9 \prod_k 5, 13, 14, 15$$

k keyfi değer sayısı olmak üzere, kısmen belirlenmiş bir fonksiyon gerçekte 2^k fonksiyonu temsil eder. Örneğin Tablo .15 te $2^5 = 32$ tane fonksiyon temsil edilmektedir. Minimal ifadenin bulunmasında keyfi değerler belirlenerek 2^k lık gruptan minimal olan seçilmiş olur.

örnek: 10 tabanında (0-9) a kadar rakamlara ilişkin ikili kodu (BCD), +3 koduna dönüştüren 4 girişli, 4 çıkışlı bir kombinezon devreyi temsil eden fonksiyonlara ilişkin doğruluk tablosu Tablo 4.16 da verilmiştir.

Tablo.4.16. Kısmen belirlenmiş fonksiyonlara ilişkin örnek.

	x_1	x_2	x_3	x_4	f_1	f_2	f_3	f_4	
0	0	0	0	0	0	0	1	1	3
1	0	0	0	1	0	1	0	0	4
2	0	0	1	0	0	1	0	1	5
3	0	0	1	1	0	1	1	0	6
4	0	1	0	0	0	1	1	1	7
5	0	1	0	1	1	0	0	0	8
6	0	1	1	0	1	0	0	1	9
7	0	1	1	1	1	0	1	0	10
8	1	0	0	0	1	0	1	1	11
9	1	0	0	1	1	1	0	0	12
10	1	0	1	0	k	k	k	k	k
11	1	0	1	1	k	k	k	k	k
12	1	1	0	0	k	k	k	k	k
13	1	1	0	1	k	k	k	k	k
14	1	1	1	0	k	k	k	k	k
15	1	1	1	1	k	k	k	k	k

Çıkış fonksiyonları sırasıyla

$$f_1 = \sum 5, 6, 7, 8, 9 + \sum_k 10, 11, 12, 13, 14, 15$$

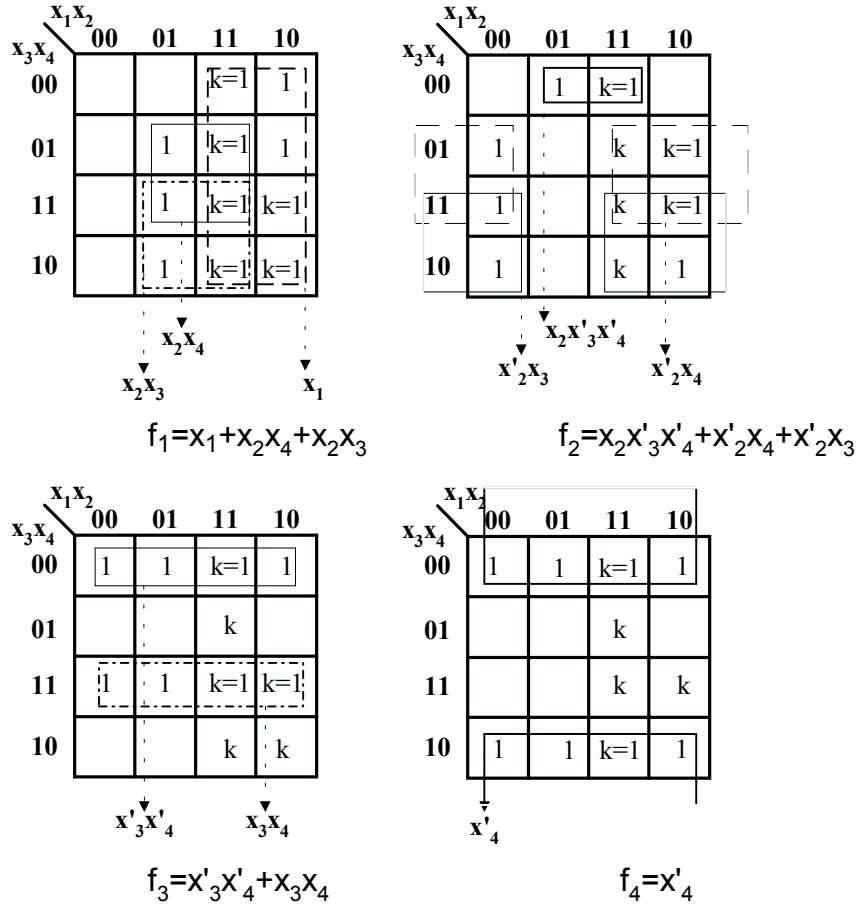
$$f_2 = \sum 1, 2, 3, 4, 9 + \sum_k 10, 11, 12, 13, 14, 15$$

$$f_3 = \sum 0, 3, 4, 7, 8 + \sum_k 10, 11, 12, 13, 14, 15$$

$$f_4 = \sum 0, 2, 4, 6, 8 + \sum_k 10, 11, 12, 13, 14, 15$$

Karnaugh diyagramında uygun alt küplerin seçimi sonucu varılan minimal fonksiyonlar Tablo 4.17 de verilmiştir.

Tablo.4.17.Kısmen belirlenmiş fonksiyonların Karnaugh diyagramında minimalleştirilmesi.



Alt küplerin seçimi maksimum mertebeli minimum sayıda küp oluşturabilmek için keyfi olan değerler saptanarak yapılmıştır. Artık bu minimal fonksiyonlar tamamen belirlenmiş fonksiyonlardır.

4.4 Kombinezonsal devre sentezi

Bu bölümün başlangıcında, bir Boole fonksiyonuna eşdeğer minimal fonksiyonun -ki gerçekleştirildiğinde minimal sayıda kapı elemanı içerir- bulunmasına ilişkin yöntemler üzerinde duruldu. Oysa sentez işlemi indirgeme işleminin dışında başka adımları da içerir. Bir kombinezonsal devrenin sentezine ilişkin adımları şöyle sıralayabiliriz.

- Problemin sözle tanımı, giriş çıkış ilişkisinden yararlanarak giriş değişkenlerinin ve çıkış değişken(leri)nin [fonksiyon(lar)nun] sayısının saptanması,
- Her bir çıkış değişkeninin (fonksiyonunun), bağımsız giriş değişkenlerince belirlenen, doğruluk tablosunun bulunması,
- Doğruluk tablosuyla belirlenen fonksiyon(lar)un indirgenmesi ve minimal fonksiyon(lar)un bulunması,
- Minimal fonksiyon(lara) a karşı düşen devrenin çizilmesi.

Sentezi istenen devreden beklenen, başlangıçta da değinildiği gibi, yalnızca kapı sayısının minimum olması değildir. Bu nedenle, önce minimal eşdeğer fonksiyona karşı düşen minimum elemanlı devreler gerçekleştirilip, sonra diğer gereksinimleri de

olabildiğince karşılayabilecek biçimde, ilaveler yapılması gerekir. Şimdi iki örneği, paralel toplayıcı ve kod dönüştürücülerinin minimal elemanlı sentezini inceleyelim. Göreceğiz ki seçeceğimiz devreler, yukarıda açıkladığımız minimal fonksiyona karşı düşen devreye daha iyi birer alternatif olacaklardır.

4.4.1 Paralel toplayıcının sentezi

İki tabanında iki sayının toplamı seri veya paralel gerçekleştirilebilir. Paralel toplayıcıda bütün bitler aynı anda uygulanır ve çıkışa ilişkin bütün bitler aynı anda görülür(devrenin gecikme süresi kadar sonra). Seri toplayıcıda ise, iki tabanındaki sayıların önce en az ağırlıklı bitleri devrenin girişine gelir ve bunların toplamı olan bit çıkışta görülürken, bir sonraki ağırlıklı bitlere eklenecek olan elde de bir başka çıkış olanak görülür. Bu nedenle seri toplama yapabilmek için, her bir bite ilişkin elde' nin bir yerde tutularak, bir sonraki bitlerin toplamına eklenmesi gerekir. Bu tutma işi bellek elemanı ile yapılır. O nedenle seri toplayıcılar ardışıl devre olarak gerçekleştirilebilir.

Paralel toplamada, bitlerin toplanmasını, ayrı birer hücre olarak düşünebiliriz. En az ağırlıklı bitlerin toplanmasında, iki giriş iki çıkış vardır. Girişler, toplanacak sayıların en az ağırlıklı bitleri; çıkışlar ise, bu bitlere ilişkin toplam ve bir sonraki bitlerin toplamında kullanılacak eldedir. Ara bitlerin toplamını ise, hücresel olarak üç girişli ve iki çıkışlı düşünebiliriz. Girişler, ilgilenilen ağırlıklı bitler ve bir evvelki bitten gelen eldedir. Çıkışlar ise, ilgilenilen bitlerin toplamı ve bir sonraki bitlerin toplamında kullanılacak eldedir. Birinci olarak düşündüğümüz hücre **yarı toplayıcı** (half adder) ikinci düşündüğümüz hücre ise **tam toplayıcı** (full adder) olarak adlandırılır. Göreceğiz ki tam toplayıcıları yarı toplayıcılardan elde edebilir.

Yarı toplayıcı

Yukarıda tanımlandığı gibi iki giriş ve iki çıkışlı olan yarı toplayıcının doğruluk tablosu aşağıdaki Tablo 4.18 deki gibidir.

Tablo 4.18 Yarı toplayıcının doğruluk tablosu

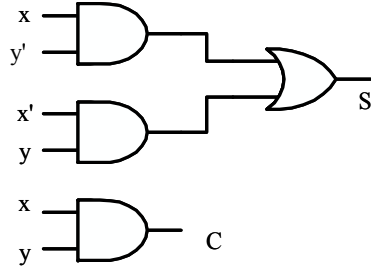
x	y	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Bu tabloya göre çıkış fonksiyonları

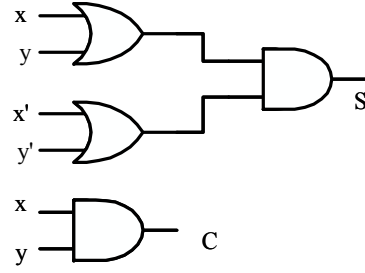
$$S = x' y + xy'$$

$$C = xy \text{ dir.}$$

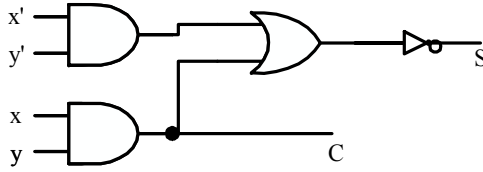
Bu fonksiyonların gerçekleştiren beş alternatif devre Şekil 4.2 de verilmiştir. Görüldüğü gibi en basit bir devrenin gerçekleştirilmesinde, tasarımcı için çeşitli alternatifler ortaya çıkabilmektedir.



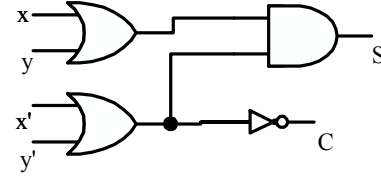
a. S ve C için birinci tip açınım $S=xy'+x'y$
 $S=(x+y)(x'+y')$ $C=xy$



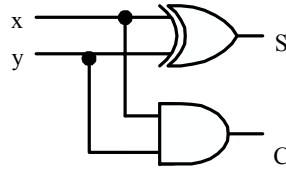
b. S için ikinci tip açınım
C için birinci tip açınım $C=xy$



c. S ve C ortaklığı $S=(C+x'y')$
 $C=xy$



d. S ve C ortaklığı $S=(x+y)(x'+y')$
 $C=(x'+y')$



e. $S=x \oplus y$, $C=xy$

Şekil 4.2 Yarı toplayıcı için birbirinin alternatifi olabilecek dört devre

Şekil 4.2 a da fonksiyonların 1. tip açınımları, b de S nin ikinci C nin birinci tip açınımları düşünülmüş ve gerçekleştirilmiştir. (c) de S ve C nin ortaklığı, $S' = (x'y+xy')' = x'y+xy$ bağıntısından yararlanılmıştır. (d) de yine S ve C nin ortak kısımları değerlendirilmiştir. (e) de ise en basit yapı görülmektedir. Fonksiyonları farklı biçimlerde ifade ederek daha fazla alternatif gerçekleştirmeler yapmak mümkündür. Hangileri olabileceğini düşününüz.

Görüldüğü gibi bu gerçekleştirilmede yukarıda işlediğimiz minimal fonksiyona karşı gelen devre seçilmemiştir.

Tam toplayıcı

Üç girişli iki çıkışlı tam toplayıcının ne olduğunu daha önce tanımlamıştık, buna göre bir sonraki basamağa için elde ve toplam çıkış fonksiyonlarına ilişkin doğruluk tablosu aşağıda Tablo 4.19 da verilmiştir. x ve y toplanacak bitler z te bu bitlere ilişkin elde yi göstermektedir. S toplanacak bitlerin toplamı, C ise bir sonraki bitlerin toplamında kullanılacak eldeyi gösterir.

Tablo 4.19. Tam toplayıcının çıkış fonksiyonlarına ilişkin doğruluk tablosu

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

S ve C fonksiyonlarına ilişkin Karnaugh diyagramları ve indirgenmiş fonksiyonlar Tablo 4.20 de verilmiştir.

Tablo 4.20 Tam toplayıcıya ilişkin indirgenmiş çıkış fonksiyonları

xy	00	01	11	10
z	00	01	11	10
0		1		1
1	1		1	

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

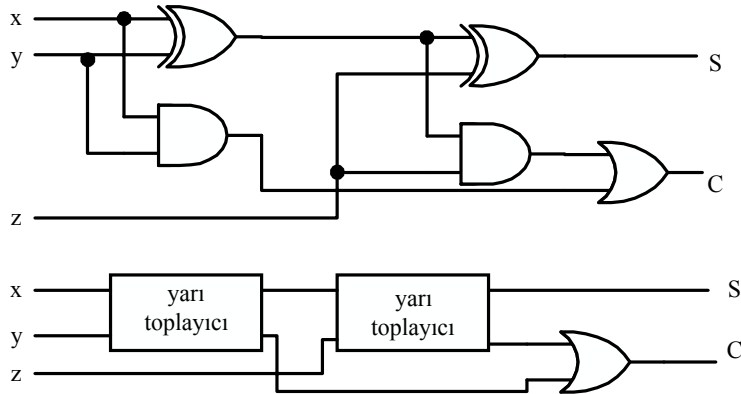
xy	00	01	11	10
z	00	01	11	10
0			1	
1		1	1	1

$$C = xy + xz + yz$$

Bulduğumuz fonksiyonlar iki seviyeli olarak gerçekleştirilebilir. S için üç girişli dört VE kapısına ve dört girişli bir VEYA kapısı gerekir. C için ise, iki girişli üç VE kapısı ile üç girişli bir VEYA kapısına gereksinimi vardır. Oysa aynı fonksiyonları aşağıdaki bağıntılardan yararlanarak iki yarı toplayıcı ve iki girişli bir VEYA kapısıyla Şekil 4.3 teki gibi elde etmek mümkündür.

$$S = z'(xy' + x'y) + z(xy' + x'y) = z'(xy' + x'y) + z(x'y' + xy) = xy'z' + x'yz' + xyz + x'y'z = (x \oplus y) \oplus z$$

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



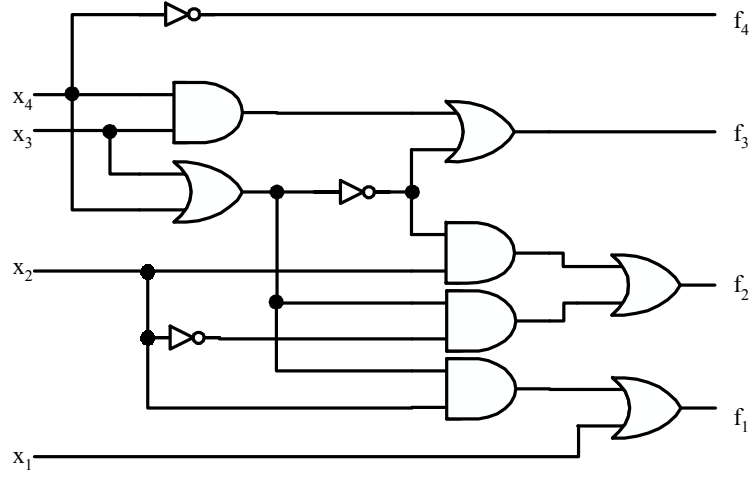
Şekil 4.3 İki yarı toplayıcı ve bir VEYA kapısı ile gerçekleştirilen tam toplayıcı

4.4.2 Kod dönüştürücü sentezi

Kısmen belirlenmiş Boole fonksiyonların indirgenmesine ilişkin verdiğimiz örnek ikili kodlanmış ondalıklı sayıları +3 koduna dönüştürüyordu. Doğruluk tablosu ve indirgenmiş çıkış fonksiyonları aşağıdaki gibi bulunmuştu.

$$\begin{aligned}
f_1 &= x_1 + x_2x_4 + x_2x_3 \\
f_2 &= x_2x'_3x'_4 + x'_2x_4 + x'_2x_3 \\
f_3 &= x'_3x'_4 + x_3x_4 \\
f_4 &= x'_4
\end{aligned}$$

Bu fonksiyonların gerçekleştirilmesi, ortak olabilecek kapı elemanları da düşünülerek, Şekil 4.4 te verilmiştir.



Şekil 4.4 BCD kodunu +3 koduna dönüştüren devre

BÖLÜM 5 KOMBİNEZONSAL MSI VE LSI ELEMANLARI

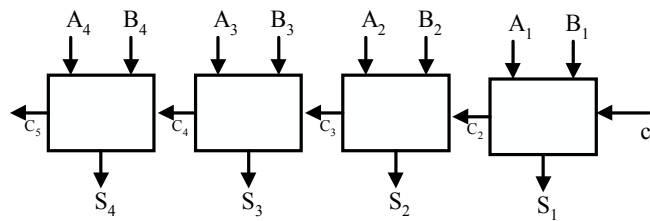
Bundan önceki bölümde, bir Boole fonksiyonun minimum sayıda kapı elemanı ile, iki seviyeli gerçekleştirilmesine ilişkin yöntem üzerinde durmuştuk. Kullanılan kapı elemanları, piyasada, SSI (**S**ingle **S**cale Integrated circuits) yongalar (chip) olarak satılmaktadır. VE, VEYA gibi kapı elemanlarından bir yongada 4 tane TÜMLEYEN kapısından ise 8 tane bulunmaktadır. Bunların dışında piyasada MSI (**M**edium **S**cale Integrated circuits) ve LSI (**L**arge **S**cale Integrated circuits) olarak sırasıyla 10-100 ve 100-1000 kapı elemanı içeren yongalarda vardır. MSI ve LSI elemanları sayısal sistemlerin tasarımında kullanılırlar. MSI elemanlarından beklenen, minimum sayıda kapı elemanı içermesinden ziyade dış bağlantıların minimum olmasıdır. Çünkü yongaların maliyetine tesir eden en önemli öğelerden biri bacak (uç) sayısıdır. Giriş değişkenleri fazla olan Boole fonksiyonların tasarımının daha önce anlatılan klasik yöntemlerle yapılması pratik olmamaktadır. Çünkü 8 değişkenli bir Boole fonksiyonunun doğruluk tablosu $2^8 = 256$ satır gerektirir. O nedenle bu gibi fonksiyonların gerçekleştirilmesi bir algoritmaya dayandırılır ve tasarımcının yaratıcılığına bağlı olarak alternatif devreler oluşturulabilir. Yegane olabilecek bir yöntemden bahsedilemez. MSI devre tasarımı genellikle algoritmalara dayandırılarak yapılır.

5.1 GİRİŞ

Bu bölümde, pratikte kullanılan MSI ve LSI elemanların lojik devrelerinin, tasarımına ilişkin algoritmalar incelenecek ve bu devrelerin istenileni yapıp yapmadığı, analiz edilerek, gösterilecektir. MSI elemanı olarak paralel girişli toplama devreleri, karşılaştırma devreleri, kod çözücü (decoder) devreler, Çoğullayıcı (multiplexer) devreler, demultiplexer devreleri ele alınacaktır. LSI elemanları olarak ta ROM (**R**ead **O**nly **M**emory) devreleri ve PLA (**P**rogrammable **L**ojik **A**rray) devreleri incelenecektir.

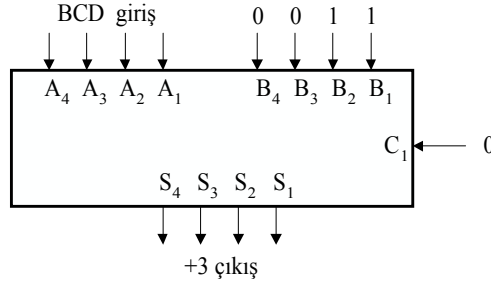
5.2 Paralel toplayıcı

Bundan önceki bölümde, bir bitlik tam toplayıcı devresinin, SSI elemanları kullanarak nasıl gerçekleştirildiğini görmüştük. Bu devreyi bir hücre olarak kullanarak tasarlanan, 4 bitlik bir MSI tam toplayıcı devresi Şekil 5.1 de verilmiştir. Pratikte kullanılan 74283 dört bitlik bir MSI tam toplayıcıdır. Toplanacak sayıların aynı ağırlıklı bitleri birer hücre girişi olarak alınmıştır. Çıkışlar her bir hücre girişlerinin toplamı ve bir sonraki hücrenin girişi olacak eldelerdir. Birinci hücrenin elde girişi 0'dır. Bir MSI elemanı içine sığdırılabilen dört tam toplayıcı ile 4 bitlik sayıların toplamı yapılabilir. Daha fazla biti olan sayıların toplamı için burada verilen MSI elemanlarının kaskad bağlanması gereklidir.



Şekil 5.1 Dört bitlik MSI toplayıcı lojik devresi

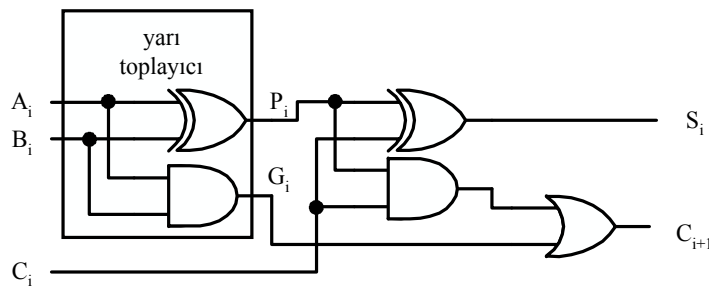
Daha önce incelediğimiz BCD den +3 koda dönüştürücüyü, burada anlattığımız MSI toplayıcı ile, hiçbir ilave kapı elemanı kullanmadan, kolayca gerçekleyebiliriz. Böyle bir gerçeklemeyle kazancımız, bağlantıları yapma işleminin olmayışı, yani işçilik ve bacak sayısının az olması yani maliyettir. Sözünü ettiğimiz kod dönüştürücünün MSI toplayıcı ile gerçekleşişi Şekil 5.2 de verilmiştir.



Şekil 5.2 BCD den +3 e dönüştürücünün MSI toplama devresi ile gerçekleştirilişi

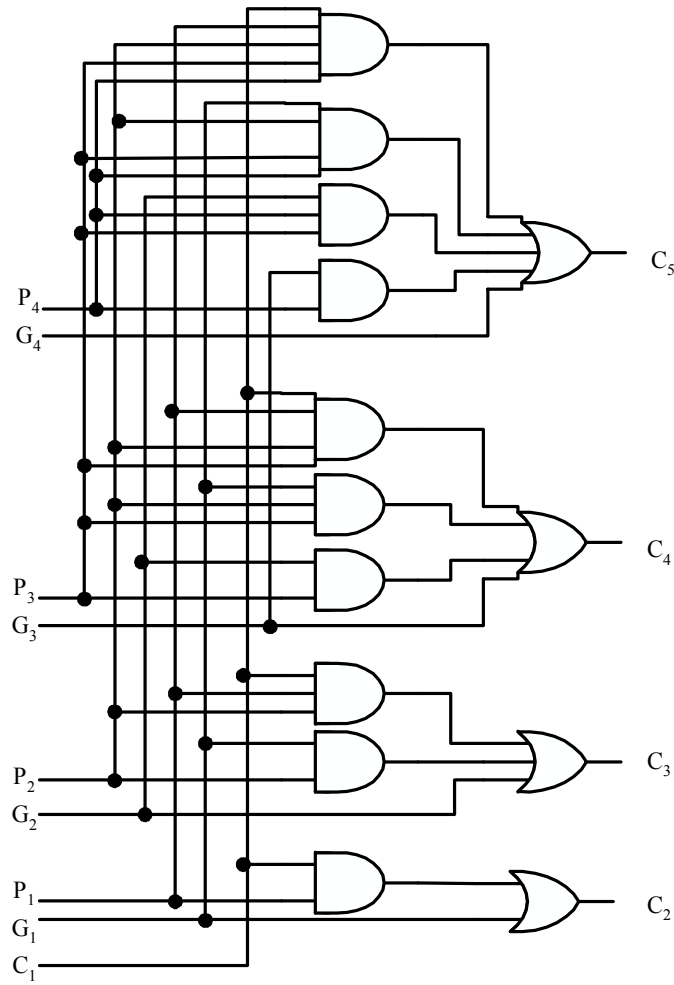
Eldenin yayılması

Yukarıda incelediğimiz paralel toplayıcının, belli bir girişe ilişkin çıkışını görebilmek için, c_5 elde sinin ne olduğunu bilmek gerek. Buda C_5 in elde edilmesine kadar geçen süre olan, yayılma zamanına denktir. Şimdi bu zamanın ne olduğunu saptamaya çalışalım. C_5 çıkışının görülebilmesi için C_4 ün dolayısıyla C_3 , C_2 nin belirlenmesi gerektiğini Şekil 5.1 deki MSI toplayıcıdan görebiliriz. Şekil 5.3 te, daha önce incelediğimiz, tam toplayıcı devresinin lojik devresi görülmektedir. C_{i+1} elde edilmesi için gerekli süre VE ve VEYA kapılardakini gecikme ile, P_i ve G_i çıkışlarını veren yarı toplayıcıdaki gecikmenin toplamı kadardır. Kaskad bağlı dört tam toplayıcılı MSI elemanındaki gecikme ise bir yarı toplayıcı gecikmesi ile, 8 kapı elemanındaki gecikme toplamı kadardır. Nedeni Yarı toplayıcıların paralel, eldelerin ise seri bağlı kaskad bir yapı göstermesidir. Sözünü ettiğimiz gecikme süresi, bit sayısı daha fazla olan sayıların toplamında, her bir bit için iki kapı elemanı gecikmesi kadar artacaktır. Bu nedenle gecikme süresini azaltmak için " hızlı elde (look ahead carry)" denen ilave devre kullanılır. Şimdi bu devreyi inceleyelim.



Şekil 5.3 Tam toplayıcı devresi

Şekil 5.3 teki yarı toplayıcının çıkışlarını P (carry propagation) elde yayılması, G yi ise elde üreticisi (carry generate) olarak isimlendirelim. P ve G çıkışları, MSI toplayıcıya, girişler uygulanır uygulanmaz, yarı toplayıcı gecikme süresi sonunda hemen oluşur. Bundan sonraki amacımız, bütün elde çıkışlarının, yarı toplayıcı yayılma süresine ilave olarak, yalnızca iki kapı elemanı gecikme süresi sonunda, oluşmasını sağlamaktır. Buda Şekil 5.2 de verilen, MSI devredeki 8 kapılık gecikmenin, 2 kapılık gecikmeye inmesi, yani 1/4 üne inmesi demektir.



Şekil 5.4 İleri bakmalı elde devresi

$$P_i = A_i \oplus B_i$$

$$G_i = A_i \cdot B_i$$

biçiminde genel olarak yazılabilir. Diğer yandan çıkışlar, bu tanımlanan P_i, G_i değişkenleri ve elde cinsinden genel olarak

$$S_i = P_i \oplus C_i$$

$$C_{i+1} = G_i + P_i C_i$$

biçiminde yazılabilir. Bir önceki C_i ler cinsinden eldeler

$$C_2 = G_1 + P_1 C_1$$

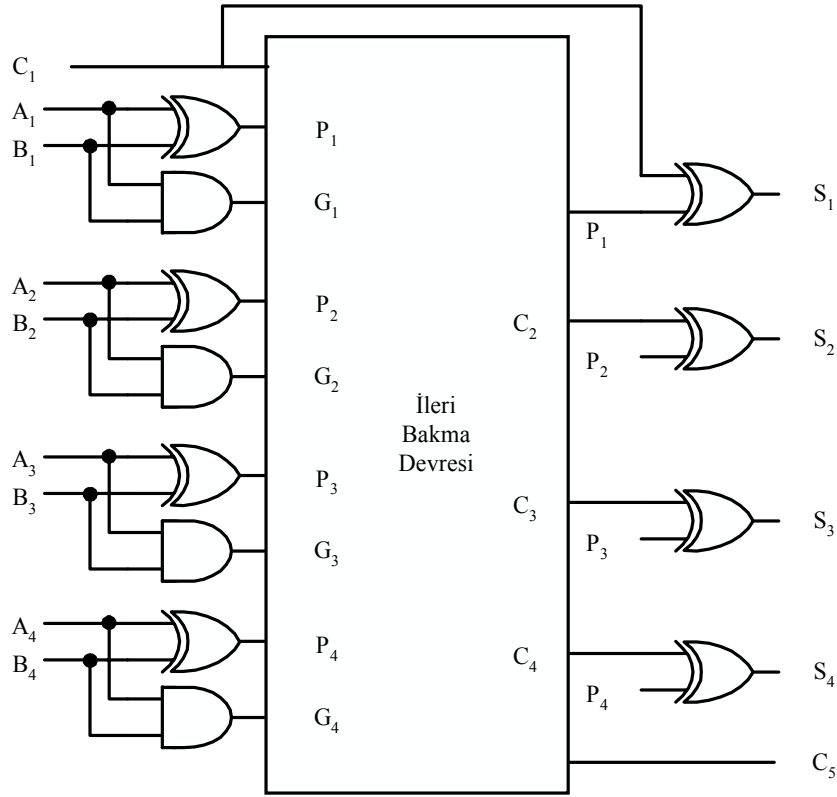
$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 (G_1 + P_1 C_1) = G_2 + P_2 G_1 + P_2 P_1 C_1$$

$$C_4 = G_3 + P_3 C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 C_1$$

$$C_5 = G_4 + P_4 C_4 = G_4 + P_4 G_3 + P_4 P_3 G_2 + P_4 P_3 P_2 G_1 + P_4 P_3 P_2 P_1 C_1$$

dir. Çarpımlar toplamı biçiminde olan bu ifadeler, iki seviyeli gerçekleştirilerek, iki kapı elemanı gecikmesi ile, sonucu verirler. Buda bizim varmak istediğimiz sonuç idi. Devre Şekil 5.4 te verilmiştir.

Şimdi ileri bakmalı paralel toplayıcı(Look ahead carry) MSI in lojik devresini Şekil 5.4 teki devre ile birlikte oluşturabiliriz Bu yapı, uygulamadaki 74182 tümleşik devresinin temelidir ve Şekil 5.5 te verilmiştir.



Şekil 5.5 İleri bakmalı bir paralel toplayıcı lojik devresi

İkili kodlanmış onlu toplayıcılar (Binary coded decimal adder)

Toplama işlemi, her zaman iki tabanında yapıлып, sonra on tabanına dönüş biçiminde olmaz. Alternatif olarak, on tabanındaki bir sayının, her bir sayamağı (dijiti) iki tabanında (ikili kodlanmış onlu, binary coded decimal) toplanıp, elde si de bulunarak yapılır. Bu durumda iki sayamağın (dijitin) toplamı 10 dan fazla ise sonuçtan 10 çıkarıp elde oluşturmak gerekir. Bu yöntemin seçilmesinin nedeni BCD ve iki tabanındaki sayıların birbirine dönüştürülmesinin, on ve iki tabanlarının dönüştürülmesine göre çok daha kolay olmasıdır. Örneğin 999 sayısı iki tabanındaki karşılığı, birinci bölümde anlatıldığı gibi, ard arda 2 ye bölme işlemi yapıp kalanların yazılması sonucunda, 1111100100 biçiminde bulunur. Oysa 999 BCD karşılığı 9 un iki tabanındaki karşılığı olan 1001 in yan yana üç defa yazılmışı 1001 1001 1001 dir. Görüldüğü gibi BCD için dönüşüm işleminin kolaylaşmasına karşılık bit sayısı iki fazladır. BCD sonuçların on tabanında gösterilmesi, her bir dijiti için yedi bölümlü gösterici (seven segment display) kullanılarak yapılır.

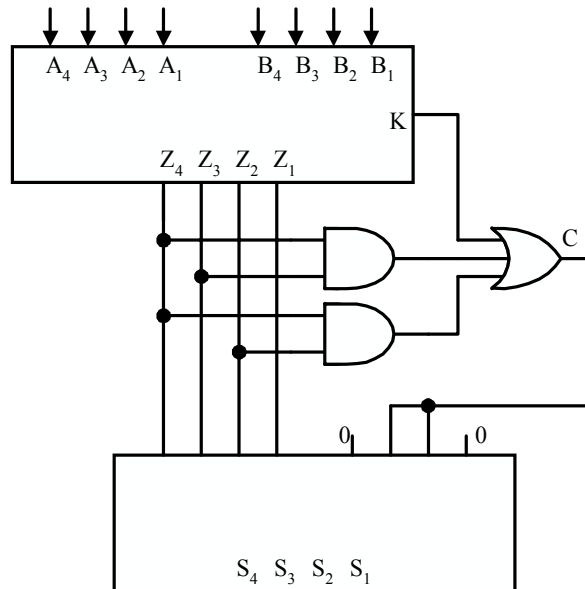
BCD iki sayının toplamı en fazla $9+9+1=19$ sonucu verebilir. Sonucu 1, bir önceki sayamaktan (dijitten) gelebilecek elde için konulmuştur. Şimdi daha önce gördüğümüz MSI toplayıcının girişlerine BCD sayılar uyguladığımızda çıkışlarının 0 dan 19 kadar olabileceğini düşünerek, çıkış sayısının BCD sayıya nasıl dönüştürülebileceğini inceleyelim. Yani iki sayamağın (dijitin) toplamı 10 dan fazla ise, sonuçtan 10 çıkarıp kalanı toplam sayamak olarak yazıp, bir sonraki sayamağa elde (1) taşımak gerekir. Bu dönüşüm aşağıda tablo biçiminde verilmiştir. İkili çıkışlar toplam Z, elde K ile, BCD çıkışlar toplam S ve elde C ile gösterilmiştir. 9 a kadar çıkışlar ikili ve BCD de aynıdır ve

çift çizgi ile ayrılmıştır. 10 ve daha yukarı olan çıkışlarda BCD nin aldığı değerlerden 10 çıkarılmış sonuç yazılmış ve elde 1 olmuştur. Amaca ulaşmamız için $C=1$ olduğu durumları, K ve Z ler cinsinden ifade etmek ve bu durumda Z ten 10 çıkarmaktır. En büyük sayı en fazla 19 olabileceğine göre, (20-31) arasındaki girişler hiç uygulanmayacaktır. O nedenle bu girişlere karşı düşen çıkışlar K (keyfi) alınmıştır.

K	Z ₄	Z ₃	Z ₂	Z ₁	C	S ₄	S ₃	S ₂	S ₁	Desimal
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 Z+6
0	1	0	1	1	1	0	0	0	1	11 Z+6
0	1	1	0	0	1	0	0	1	0	12 Z+6
0	1	1	0	1	1	0	0	1	1	13 Z+6
0	1	1	1	0	1	0	1	0	0	14 Z+6
0	1	1	1	1	1	0	1	0	1	15 Z+6
1	0	0	0	0	1	0	1	1	0	16 Z+6
1	0	0	0	1	1	0	1	1	1	17 Z+6
1	0	0	1	0	1	1	0	0	0	18 Z+6
1	0	0	1	1	1	1	0	0	1	19 Z+6

Görüldüğü gibi $K=1$ iken $C=1$ dir. Bunun dışında $Z=10-15$ arasında da $C=1$ olmalıdır. Z'ler bağımsız değişkenler olmak üzere, 10-15 te (1) olan diğer yerlerde (0) olan bir fonksiyon düşünsek, bu fonksiyonun indirgenmiş hali $Z_4Z_3 + Z_4Z_2$ olur. O halde $C = K + Z_4Z_3 + Z_4Z_2$ dir. Bu sonucu beş değişkenli bir Karnaugh diyagramıyla da görebilirdik.

$C=1$ iken toplamdan 10 çıkarma işlemi, S çıkışlarına 6 ilave etme işlemine denktir; yani aynı sonucu verir. 6 ilave etme için ayrıca bir ikili toplayıcı kullanılabiliriz. Şekil 5.6 da iki ikili toplayıcı ve elde yi veren üç kapı elemanı ile bir BCD toplayıcı devrenin lojik devresi verilmiştir.



Şekil 5.6 BCD toplayıcı

Yukarıda incelediğimiz temel alınarak, ikinci ikili toplayıcı daha da basitleştirilerek, tek bir tümleşik devre, 74283 tasarlanmıştır. Yani yukarıda verilen gibi iki MSI elemanı ve üç kapı elemanı tek bir MSI olarak oluşturulmuştur.

Burada incelediğimiz BCD toplama devresini, klasik yöntemle yapmaya kalksaydık, 9 girişli (dördü bir sayıyı, diğer dördü diğer sayıyı ve biride elde) devreler gerçekleştirecektik. Buna karşı gelen Boole fonksiyonların doğruluk tablosu $2^9 = 512$ satırdan oluşacaktı. O nedenle algoritmik hücresel çözümler daha avantajlıdır.

5.3 Karşılaştırma devresi

Karşılaştırma devresi, sekiz girişindeki 4 bitlik iki sayıyı karşılaştırarak, üç çıkışında küçük, büyük ve eşitlik durumlarını belirlerler. Girişler $A=A_1A_2A_3A_4$ ve $B=B_1B_2B_3B_4$ karşılaştırılacak dörder bitlik sayılar, çıkışlar da $(A > B)$ için $\zeta_1 = 1$ $\zeta_2 = 0$ $\zeta_3 = 0$; $(A < B)$ için $\zeta_1 = 0$ $\zeta_2 = 1$ $\zeta_3 = 0$; ve $A = B$ iken $\zeta_1 = 0$ $\zeta_2 = 0$ $\zeta_3 = 1$ dir. Klasik anlamda probleme bakarsak $2^8 = 256$ farklı giriş vardır, ve bildiğimiz yöntemlerle bu devreye karşı düşen fonksiyonun indirgemesini elle yapmak pratik değildir. Bilgisayarla yapmak mümkün olsa bile , algoritmik olan hücresel yapı tüm devre üretimine daha uygun olacaktır.

A ve B sayılarının eşitliğini yakalayabilmek için

$$X_i = A_i B_i + A'_i B'_i \quad i = 1,2,3,4$$

ile tanımlanan x değişkenlerini incelemek yeter. Çünkü A ve B ye ilişkin karşılıklı bitlerin eşit olması durumunda bu ifade 1, aksi halde 0 değerini alır. (x) lerin dördünün birden 1 olması $A = B$ olduğunu gösterir Yani

$$\zeta_3 = x_1 x_2 x_3 x_4 = 1$$

ise $A=B$ dir. Üçüncü çıkış belirlenmiş oldu.

$A > B$ için ise önce en ağırlıklı bitleri karşılaştırmalıyız, eğer A nın 1, B ninki 0 ise ($A_4 B'_4 = 1$) $A > B$ dir. En ağırlıklı bitler birbirine ($x_4 = 1$) eşit ve ($A_3 B'_3 = 1$) ise yine $A > B$ dir. Bu yapı en kötü ihtimalle, benzer biçimde, en az ağırlıklı bitlerin karşılaştırılmasına kadar devam eder. Yani

$$\zeta_1 = A_4 B'_4 + x_4 A_3 B'_3 + x_4 x_3 A_2 B'_2 + x_4 x_3 x_2 A_1 B'_1$$

ifadesi 1 ise $A > B$ dir. Bir başka deyişle $A > B$ nin sağlanması için

i. $A_4 = 1, B_4 = 0$ ($A_4 B'_4 = 1$) veya

ii. $A_4 = B_4$ ve $A_3 = 1, B_3 = 0$ ($A_3 B'_3 = 1$) veya

iii. $A_4 = B_4$ ve $A_3 = B_3$ ve $A_2 = 1, B_2 = 0$ ($A_2 B'_2 = 1$) veya.

iv. $A_4 = B_4$ ve $A_3 = B_3$ ve $A_2 = B_2$ ve $A_1 = 1, B_1 = 0$ ($A_1 B'_1 = 1$)

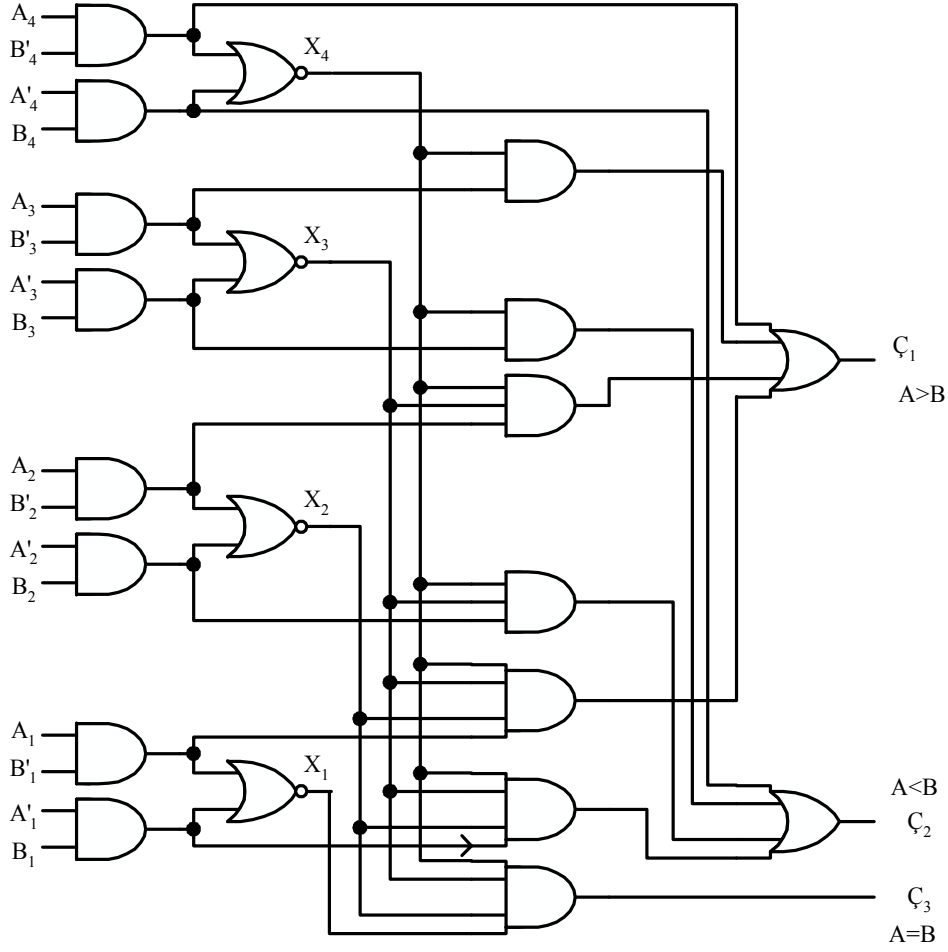
dir.

Benzer akıl yürütme ile $A < B$ çıkışı için

$$\zeta_2 = A'_4 B_4 + x_4 A'_3 B_3 + x_4 x_3 A'_2 B_2 + x_4 x_3 x_2 A'_1 B_1$$

ifadesi bulunabilir.

Şekil 5.7 de yukarıda ifadelere karşı düşen devre verilmiştir.

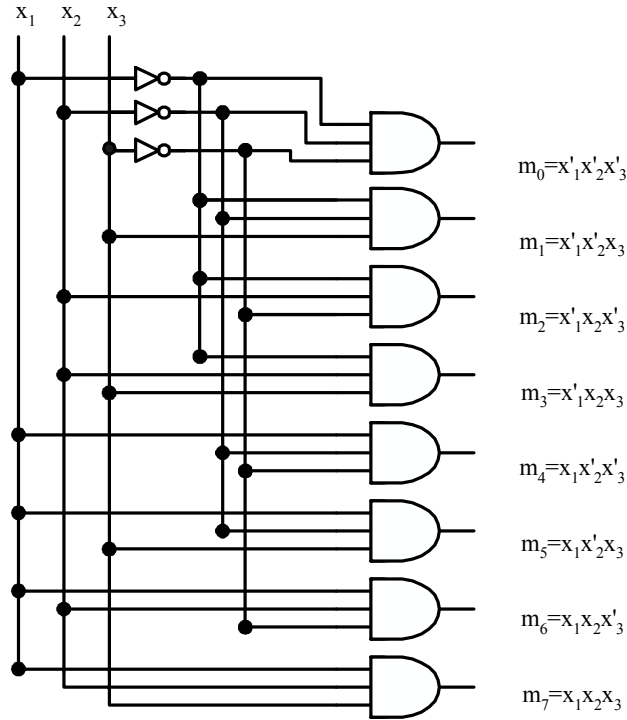


Şekil 5.7 Karşılaştırma devresi

5.4 Çözücüler (Decoders)

Çözücüler n girişli 2^n çıkışlı devrelerdir. n girişin alabileceği her bir giriş permütasyonu için çıkışlardan yalnızca biri 1, diğerleri ise 0 olur. Bir başka deyişle çıkışlar mintermleri belirlemektedir.

Şekil 5.8 de 3×8 (3×2^3) lik bir çözücü devresi verilmiştir.



Şekil 5.8 3X8 çözücü devresi

Boole fonksiyonlarının çözücüler kullanılarak gerçekleştirilmesi

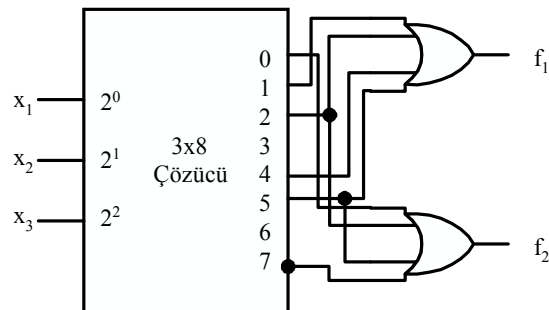
$n \times 2^n$ lik bir çözücü, olabilecek bütün 2^n minterimi oluşturacağına göre; herhangi bir n değişkenli Boole fonksiyonu, iki seviyeli olarak çözücülerle gerçekleştirilebilir. n değişkenli, doğru minterm sayısı k olan bir Boole fonksiyonu, $n \times 2^n$ lik bir çözücü ve k giriş yelpazesi olan bir VEYA kapısıyla gerçekleştirilebilir. Hatta genelleştirme yapılarak, n değişkenli m tane Boole fonksiyonu bir tane $n \times 2^n$ çözücü ve m tane VEYA kapısıyla gerçekleştirilebilir.

Bir örnek olarak üç değişkenli, iki Boole fonksiyonunu bir MSI çözücü ile gerçekleştirelim. Fonksiyonlar

$$f_1 = \sum 1, 2, 4, 5$$

$$f_2 = \sum 0, 2, 5, 7$$

olsun. Her birinde 4 doğru minterm olduğuna göre dört girişli iki VEYA kapısına gerek vardır. Şekil 5.9 da devre verilmiştir.

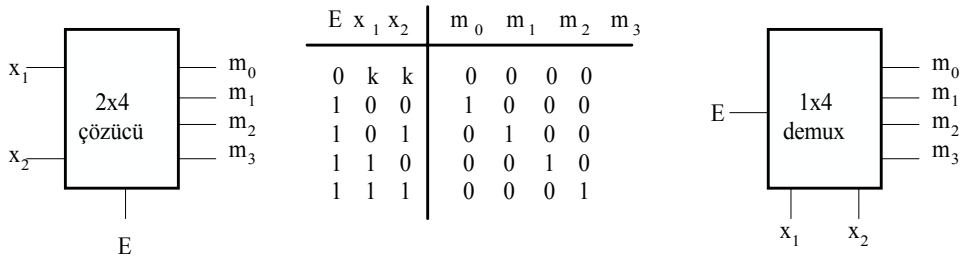


Şekil 5.9 İki fonksiyonun bir çözücü kullanılarak gerçekleştirilmesi

Fonksiyonların çözücüler kullanılarak gerçekleştirilmesinde bir indirgeme sorunu yoktur. Buna karşılık çözücünün içinde kapı ve giriş / çıkış yelpazesi açısından da hiçbir tasarruf yapılmamıştır. SSI elemanlarıyla yapılacak sentezde ise, indirgeme ile, sözü edilen tasarruflar göz önünde tutulmuştur. SSI veya MSI elemanlarından hangisinin tercih edilerek gerçekleştirilme yapılacağı koşullara bağlıdır. Örneğin aynı fonksiyondan çok sayıda üretilecekse, SSI sentezdeki indirgeme yapılarak bulunan lojik devrenin, tüm devre olarak gerçekleştirilmesi daha uygun olur. Eğer yalnızca üç beş tane fonksiyon gerçekleştirilmesi söz konusu ise, mevcut MSI elemanlarını kullanmak hem zaman hem de işçilik tasarrufu sağlar.

Yanlış minterim sayısı doğru minterim sayısından az ise, f' (f in tümleyeni) gerçekleştirilip VEYA yerine TÜVEYA kapısı kullanılarak; VEYA kapısı giriş yelpazesinde tasarruf sağlanabilir.

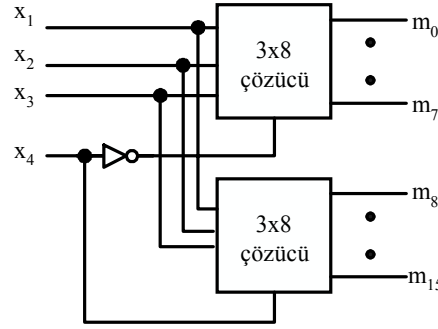
Pratikte kullanılan MSI çözücülerin, bir de kontrol girişi "enable" vardır ve MSI elemanının çalışmasını kontrol eder. Bu giriş 0 iken minterim çıkışlarının hepsi 0 olur, aksine bu giriş 1 iken ise çözücü normal çalışma işlevini yerine getirir. Devrede bütün VE kapılarının girişlerine bu kontrol girişi bağlanarak istenen işlev yerine getirilebilir. Şekil 5.10 da böyle bir çözücü ve doğruluk tablosu verilmiştir.



Şekil 5.10 Kontrol girişli bir çözücü ve alternatif yorumu: demultiplekser

Bu çözücüyü, sanki E normal giriş, x_1 ve x_2 girişleri de sanki kontrol girişleriymiş gibi de düşünebiliriz. Bu durumda kontrol girişinin karşı düştüğü minterim çıkışı E girişinin değerini alır. Bir başka deyişle E girişine gelen bilgi dört çıkıştan istenenine verilebilmektedir. Bu işlev haberleşmede kullanılan demultiplekser (demultiplexer) işlevidir.

Çözücülerdeki kontrol girişi, giriş sayısı az olanlar çözücüler kullanarak, giriş sayısı daha fazla olan çözücülerin elde edilmesine de yarar. Şekil 5.11 de iki tane 3x8 lik çözücü kullanılarak gerçekleştirilen, 4x16 lık bir çözücü devresi verilmiştir. $x_4 = 0$ iken yukarıdaki çözücü çalışır ve m_0 dan m_7 ye kadar minterimler çıkışında gözlenir. Bu sırada alttaki çözücü çalışmaz ve bütün çıkışları 0 dır. $x_4 = 1$ olduğunda ise alttaki çözücü çalışır ve m_8 den m_{15} e kadar minterimler, bu çözücü çıkışında gözlenir. Bu sırada üstteki çözücü çalışmaz ve çıkışları 0 dır.



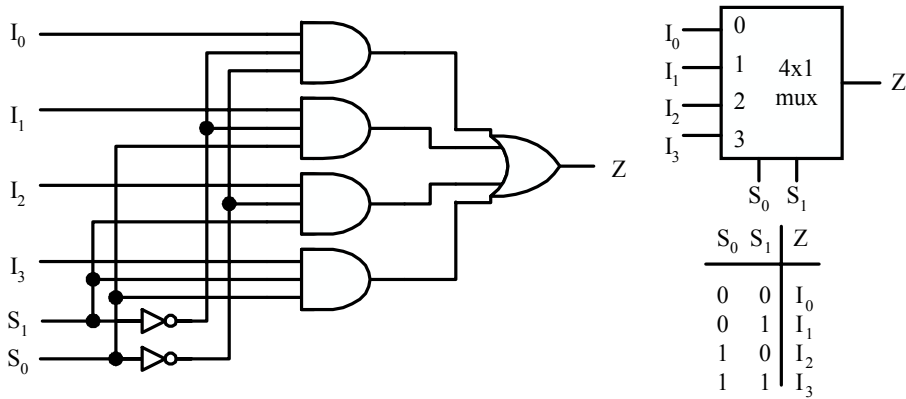
Şekil 5.11 İki 3x8 lik çözücüden elde edilen 4x16 lık çözücü devresi

Kod çözücünün tersi işlevi yerine getiren devrelere kodlayıcı (encoder) denir. Bu devrelerin giriş sayısı 2^n , çıkış sayısı ise n dir. Girişlerden biri 1 iken, diğerleri 0 olmak zorundadır. Hangi girişe (1) gelmişse on girişin iki tabanındaki karşılığı çıkışta görülür. Örneğin 6. girişe 1 gelmişse, çıkış 110 dır. Böyle bir devreyi bir uygulama olarak gerçekleyiniz.

5.5 Multiplexer (Çoğullayıcı) devreler

Genel olarak çoğullayıcılar, birden fazla fiziksel bilgi kaynağındaki bilgiyi, istenen sırada, tek bir fiziksel bilgi kanalına aktarmaya yararlar. Lojik devre kapsamı içinde çoğullayıcı 2^n girişteki bilgiyi n kontrol girişi (select) ile istenen sırada tek çıkışa aktarmaya yarayan kombinezonsal devrelerdir. Kısaca mux diye bilinirler.

Şekil 5.12 de 4x1 ($2^2 \times 1$) lik bir mux'ın lojik devresi verilmiştir. seçme girişlerinin her bir permütasyonu VE kapılarıyla oluşturulurken bu kapıların girişlerine 2^n mux girişleri ayrı ayrı bağlanmıştır. Görüldüğü gibi bu yapıda $n \times 2^n$ ($n = 2$)lik bir dekoderin, 2^n sayıdaki VE kapısına n tane bağımsız giriş bağlanmıştır. 2^n tane VE çıkışı, mux ın tek bir çıkışına, bir VEYA kapısıyla alınmıştır.

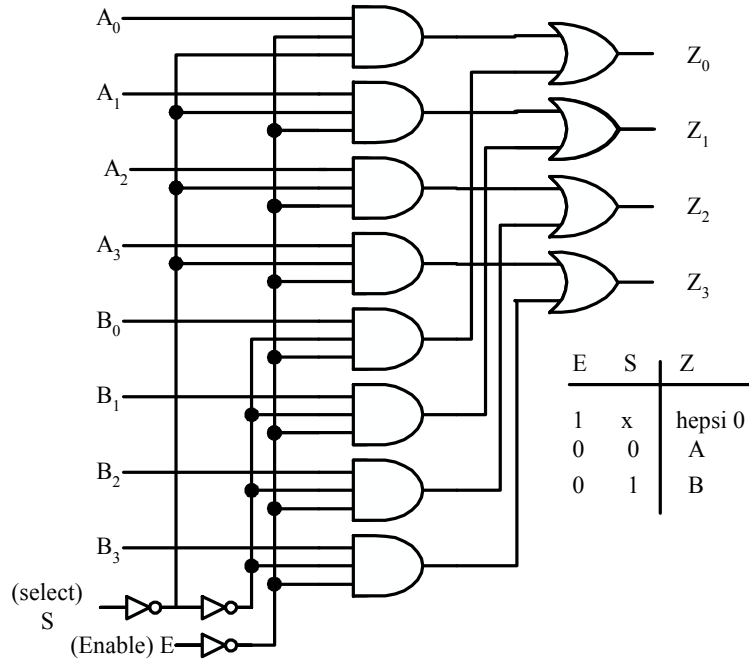


Şekil 5.12 4x1 mux lojik devresi, blok diyagramı ve doğruluk tablosu

Pratikte kullanılan mux ların, dekoderler de olduğu gibi, çalışmayı kontrol eden bir enable girişi (strobe da denir) vardır. Enable'ın belli bir giriş değeri için çıkışların hepsi 0 (bazen 1) olur, diğer giriş değeri için ise mux normal çalışma işlevini yerine getirir. Bu kontrol girişi bir üst mertebeden mux ın bir alt mertebeden iki mux kullanılarak elde edilmesini de sağlar. Söz konusu yapı decoderler de olduğu gibidir. Örneğin 8x1 lik bir

mux, 2 tane 4x1 lik mux in enable girişleri kullanılarak elde edilebilir. Çıkışa gelecek kapının ne olması gerektiğini düşününüz.

Pratikte birden fazla mux in, seçme ve enable girişleri ortak olmak üzere aynı MSI elemanında kullanıldığı görülür. Örneğin içersinde dört tane 2x1 mux in bulunduğu MSI elemanları (74157) ticari olarak vardır. Şekil 5.13 te böyle bir MSI elemanının lojik devresi verilmiştir. A ve B girişlerini farklı kaynaklardan gelen dörder bitlik enformasyon olarak düşünürsek, sanki bu devre, A veya B enformasyonundan birini isteği bağlı olarak seçip aynı bir yoldan (bus) bir başka yere göndermektedir. Böyle bir işlev bilgisayarların alt birimleri arasında veri alış verişinin, ortak bir hat üzerinde yapılmasında faydalı olur.



Şekil 5.13 dört 2x1 lik mux tan oluşan lojik devre

Mux larla Boole fonksiyonlarının gerçekleştirilmesi

Bundan önceki bölümlerde gördüğümüz gibi, dekoder ve mux larda mintermler oluşturulmakta, dekoderlerde ayrıca bir VEYA kapısı kullanılarak, herhangi bir fonksiyon gerçekleştirilmekteydi. Mux larda ise bu VEYA kapısı tümleşik devrenin içinde bulunmaktadır. Amacımız mux ları kullanarak bir Boole fonksiyonunun nasıl gerçekleştirilebileceğini göstermek. n seçicili $2^n \times 1$ lik bir mux ile n değişkenli bir fonksiyonu gerçekleştirelim. Değişkenleri seçiciler seçelim. Seçicilerin alacağı her bir değer permütasyonu bir mintermi belirler ve buna karşı düşen mux ta bir giriş vardır. Eğer gerçekleştireceğimiz fonksiyon, bu minterm için 1 ise (doğru minterm) buna karşı düşen girişe 1, 0 ise (yanlış minterm) bu girişe 0 uygulayalım. Aynı işlev bütün giriş değer permütasyonları için tekrarlandığında mux çıkışında istediğimiz fonksiyonu gerçekleştirmiş oluruz.

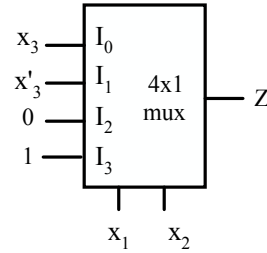
Yukarıda açıklananndan bir adım daha ileri giderek, $(n+1)$ değişkenli bir fonksiyon n seçicili $2^n \times 1$ lik bir mux la gerçekleştirilebilir. Bunun için n değişken seçiciler olarak alınıp, girişlere de 0, 1, x_n veya x'_n den biri uygulanır. Bir örnek üzerinden gerçekleştirmenin nasıl yapılacağını görelim. Şekil 5.13 te bir örnek fonksiyon ve devresi verilmiştir. Fonksiyon üç değişkenli $f = \sum 1, 2, 6, 7$ olduğuna göre 4x1 lik iki seçicili bir mux a gereksinim vardır. x_1, x_2 değişkenlerini seçiciler olarak seçelim. Doğruluk tablosundan görüldüğü gibi x_1, x_2

değişkenlerinin bir değer permütasyonu için x_3 in aldığı 0 ve 1 olmak üzere iki değer vardır. Bu değerler için fonksiyonun aldığı değerler 0, 1, x_3 veya x'_3 olabilir. Buna göre tabloyu tekrar düzenlersek (Şekil 5.13 te gölgelendirilerek belirtildi). , mux girişlerine ne bağlayacağımız ortaya çıkar. Bu tablodaki sonuca, bir başka düzenlemeyle daha çabuk ulaşabiliriz. Öyle ki sütunları, seçicilerin belirlediği mintermlere karşı düşen girişler; satırları da x_3 değişkeni ve tümleyenini gösterecek biçimde bir tablo oluşturulsun. Bu durumda satır ve sütunlara karşı gelen yerler, olabilecek bütün minterimlerdir. Doğru minterimleri bir yuvarlak içine alalım, yuvarlak içinde olmayanlar yanlış minterimler olsun. Buna göre

- i bir sütundaki (bir seçici girişini belirler) mintermler yuvarlak içinde değilse, bu sütuna ilişkin giriş 0,
- ii. bir sütundaki (bir seçici girişini belirler) mintermler yuvarlak içindeyse, bu sütuna ilişkin giriş 1,
- iii. bir sütundaki (bir seçici girişini belirler) mintermlerden x_3 satırına karşı gelen yuvarlak içinde x'_3 satırına karşı gelen yuvarlak içinde değilse bu sütuna ilişkin giriş x_1 ,
- iv (iii) durumun tersi ise ilgili giriş x'_1 olmalıdır. (Şekil 5.13)

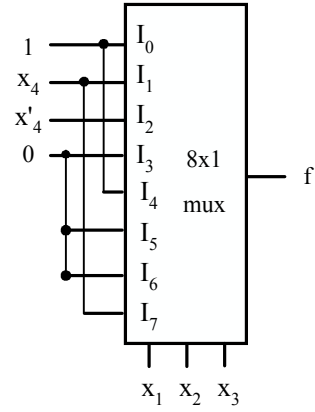
minter	x_1	x_2	x_3	f	
0	0	0	0	0	
1	0	0	1	1	$f = x_3$
2	0	1	0	1	$f =$
3	0	1	1	0	x'_3
4	1	0	0	0	$f = 0$
5	1	0	1	0	
6	1	1	0	1	$f = 1$
7	1	1	1	1	

	$x_1 x_2$			
	00	01	10	11
	I_0	I_1	I_2	I_3
$(0)x'_3$	0	②	4	⑥
$(1)x_3$	①	3	5	⑦
	x_3	x'_3	0	1



Şekil 5.13 dört değişkenli bir fonksiyonun 4x1 mux la gerçekleştirilmesi

	$x_1x_2x_3$							
	000	001	010	011	100	101	110	111
	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
$(0)x'_4$	①	2	④	6	⑧	10	12	14
$(1)x_4$	①	③	5	7	⑨	11	13	⑮
	1	x_4	x'_4	0	1	0	0	x_4



Şekil 5.14 $f = \sum 0,1,3,4,8,9,15$ fonksiyonunun 8x1 mux kullanarak gerçekleştirilmesi

Şekil 5.14 te dört değişkenli bir örnek fonksiyonun 8x1 lik bir mux ile gerçekleştirilmesi gösterilmiştir.

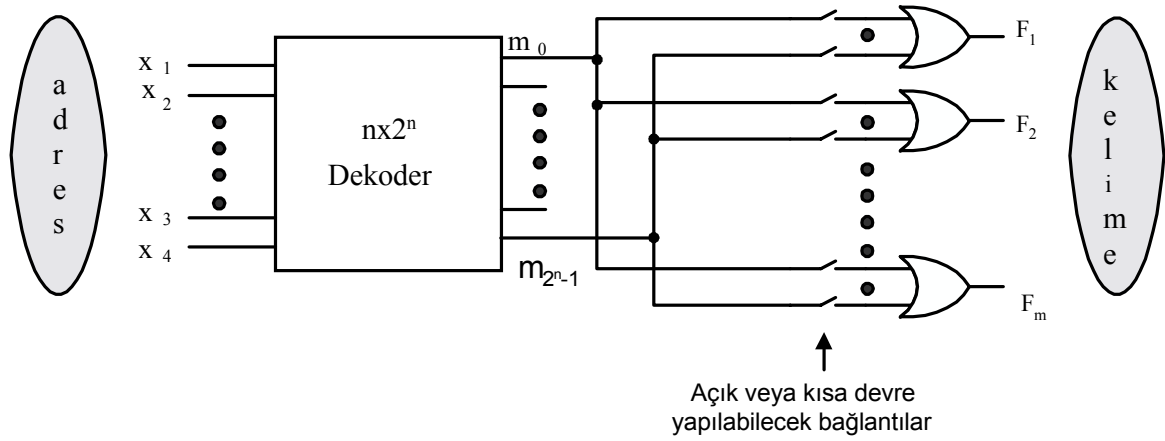
Yukarıda örnek üzerinde açıklanan gerçekleştirme yöntemi genelleştirilebilir. Burada en çok ağırlıklı bitlere karşı gelen değişkenlerin dışındaki değişkenler seçici girişleri olarak seçilmiştir, mux girişlerine de 0,1 veya en ağırlıklı değişkenin kendisi veya tümleyeni gelmektedir. halbuki herhangi n-1 değişken seçici, kalan değişken giriş seçilerek te, benzer akıl yürütmeye gerçekleştirme yapılabilir.

5.6 Read-Only Memory (ROM)

Bu ve bundan sonraki kısımda, ROM (**R**ead **O**nly **M**emory) ve PLA (**P**rogrammable **L**ogic **A**rray) elemanları göreceğiz. Bu elemanlar içinde 100-1000 civarında kapı elemanları bulunduran LSI (Large Scale Integrated Circuits) devrelerdir.

ROM elemanlarını, gördükleri işlevler açısından, iki farklı biçimde inceleyebiliriz.

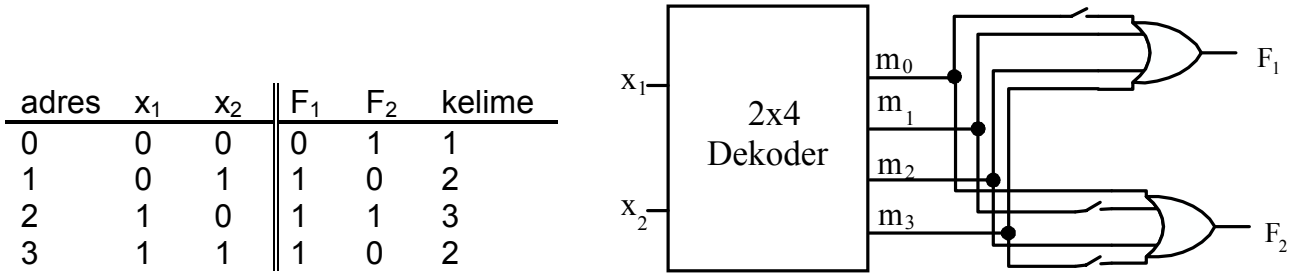
Birincisi birden fazla Boole fonksiyonunun, bir tümleşik devre ile gerçekleştirilmesini sağlarlar. ROM LSI elemanının içinde bir dekodere ile gerçekleştirilecek fonksiyon sayısı kadar VEYA kapısı vardır. VEYA kapısı sayısı m olsun. $n \times 2^n$ lik dekodere 2^n VE kapısının çıkışını her bir VEYA kapısına bağlayan, $m \times 2^n$ tane bağlantı vardır. Bu bağlantılar, isteğe göre, açık veya kısa devre yapılabilirler. Diyelim m tane Boole fonksiyonu gerçekleştireceğiz. 1. fonksiyonda hangi doğru minterimler varsa, onlara ilişkin VE kapılarının çıkışları, 1. VEYA kapısının girişlerine, bağlantılar kısa devre yapılarak birleştirilir. Yanlış minterimlere karşı gelen VE kapılarının çıkışları ise açık devre yapılarak VEYA kapısının girişine bağlanmaz. Böylece 1. fonksiyon gerçekleştirilmiş olur. Diğer fonksiyonlar da benzer biçimde gerçekleştirilir. Bağlantıların açık veya kısa devre yapılmaları, üretici firmanın belirlediği yöntemle özel aletler kullanılarak yapılır. **n** giriş değişkenli **m** çıkışlı bir ROM elemanın iç yapısı Şekil 5.15 te verildiği gibidir. Böyle bir ROM $2^n \times m$ lik ROM olarak adlandırılır.



Şekil 5.15 Genel bir ROM elemanının devresi

Yukarı açıkladığımız ROM' u, yalnızca okunabilen bellek işlevini yerine getiren bir eleman olarak ta görebiliriz. Yalnızca okunabilen bellek elemanından kastedilen, adres bitleri dediğimiz giriş değişkenlerinin belli bir permütasyonuna karşı gelen belli bir yerde, kelime (word) dediğimiz m bitlik belli bir bilginin tutulmasıdır. $2^n \times m$ lik ROM da n giriş değişkeni ile 2^n farklı adres tanımlanabilir, bu adreslere de m bitlik kelimeler (word) saklanabilir. Yukarıda açıkladığımız ROM da, girişler sanki adresleri (minterimleri) vermekte; girişlere karşı gelen m fonksiyonun aldığı değer de, çıkışları sanki o adresteki bilgi(word)yi vermektedir diye düşünülebilir (Şekil 5.15).

Şimdi bir örnek üzerinde ROM un iki farklı yorumunu yapalım. Ele alacağımız örnek, çizim ve anlatım kolaylığı açısından iki giriş ve iki çıkışlı seçilmiştir. Pratikte 7-8 giriş ve 8 çıkış değişkenliler yaygındır. Şekil 5.16 da verilen örnek ROM, iki iki-değişkenli fonksiyon olarak düşünülebilir. Aynı ROM, dört adresinde, iki bitlik dört kelimenin bulunduğu, bir bellek elemanı olarak ta düşünülebilir.

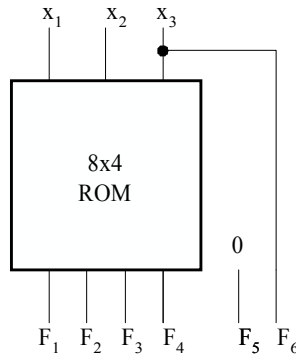


Şekil 5.16 Örnek bir ROM ve iki yorumu

Şimdi pratik bir örnek yapalım. Sıfırdan yediye kadar sayıların karelerini alan bir kombinezon devreyi ROM kullanarak gerçekleyelim. Yedi ye kadar sayıları temsil edebilmemiz için üç giriş değişkenine, sıfırdan kırk dokuza kadar sayıları temsil edebilmemiz için de altı çıkış fonksiyonuna gereksinim vardır ($2^5 = 32$, $2^6 = 64$). Doğruluk tablosu ve ROM devresi Şekil 5.17 verilmiştir. Doğruluk tablosundan görüldüğü gibi; 5. çıkış her giriş için 0, 6. çıkış ise 3. girişle aynıdır o nedenle ayrıca bir ROM çıkışı

gerektirmezler. 3 adres bitli 4 çıkışlı bir ROM, 0 dan 7 ye kadar sayıların karelerini almak için yeterli olur.

	x_1	x_2	x_3	F_1	F_2	F_3	F_4	F_5	F_6
0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	1
2	0	1	0	0	0	0	1	0	0
3	0	1	1	0	0	1	0	0	1
4	1	0	0	0	1	0	0	0	0
5	1	0	1	0	1	1	0	0	1
6	1	1	0	1	0	0	1	0	0
7	1	1	1	1	1	0	0	0	1



Şekil 5.17 Kare alma işlemi yapan bir ROM devresi

ROM türleri

ROM' lardaki bağlantılar iki biçimde yapılabilir: mask ve programlama. Mask bağlantılar üretim sırasında direkt olarak yapılır. Programlama da ise üretimden sonra; kullanıcı tarafından, özel cihazlarda katalog bilgisine göre programlanarak **"PROM"**, yapılır. Her ne kadar programlanarak yapılıyor denirse de bağlantılar hardware olarak sağlanmaktadır.

ROM lara konan bilgiler, ROM'un türüne göre, ya bir daha hiç değiştirilemezler veya silinip yeni bilgilerle değiştirilebilirler. İkinci tür olanlara **"silinebilir PROM " EPROM (Erasable PROM)** denir. Silme işlemi elektronik olarak yapılanlara **"elektronik olarak silinebilen ROM"(EAROM:Electronically alterable ROM)** denir. Silme işlemi mor ötesi ışınla yapılanlara ise **"silinebilen ROM" EPROM (erasable ROM)** denir.

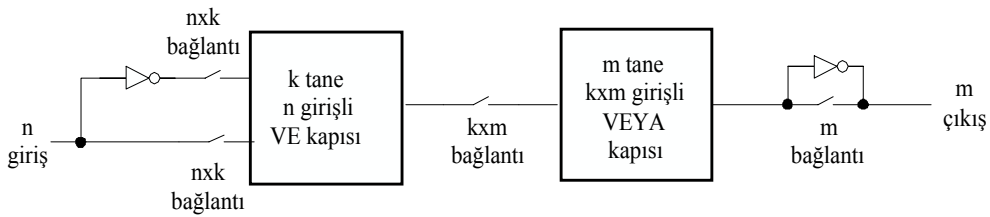
ROM lar genel olarak çok girişli bir çok fonksiyonun gerçekleşmesinde yaygın olarak kullanılabilirler, örneğin ASCII kodundan EBIDIC koduna dönüştürücüler gibi. Diğer taraftan ROMlar yalnızca okunan bellek elemanları olarak software kontrol birimlerinde yaygın biçimde kullanılırlar.

5.7 PLA (Programable Logic Array)

Bundan önceki kısımda ROM larla fonksiyon gerçekleştirirken gördük ki, fonksiyonun 1 değerini aldığı doğru minterimler için, ROM un içinde n girişli bir VE kapısı vardır. Yani ROM larda 2^n VE kapısı varken, gerçekleştirmede biz bunlardan yalnız fonksiyondaki 1

sayısı kadar olanını kullanmaktayız. Sağlanan avantaj, indirgeme düşünmeden gerçekleşmeye geçilebilmesi oldu. Buna karşılık ROM daki bir çok VE kapısı da kullanılmadı. Oysa biliyoruz ki indirgeme yaptığımızda kullanacağımız VE kapısı sayısı fonksiyondaki 1 sayısından çok daha az olacaktır. İşte bu olumsuzluğa cevap verebilmek için PLA (Programable Logic Array) LSI elemanları geliştirilmiştir.

PLA' in yapısı ROM yapısına benzerdir. ROM lu gerçeklemelerde fonksiyondaki her bir 1 e karşılık ROM daki 2^n kapıdan biri seçilir. ROM larda n giriş değişkeni sayısı olmak üzere n giriş yelpazeli 2^n VE kapısı varken; PLA de 2^n den çok daha küçük sayıda n giriş yelpazeli VE kapısı vardır. İkinci seviyedeki VEYA kapılarının giriş yelpazesi, ROM larda 2^n iken PLA de 2^n den çok daha küçük VE kapısı sayısı kadardır. Örneğin Pratikte kullanılan bir PLA de $n = 16$ giriş değişkeni , 46 n -girişli VE kapısı (ROM olsa idi bu sayının $2^{16} = 256$ olması gerekirdi) ve 8 çıkış (8 bitlik kelimeler) vardır. 82S100 PLA nin yapısı bu yapıdadır



Şekil 5.18 Bir PLA elemanın blok diyagramı

Şekil 5.18 de genel olarak bir PLA elemanın blok diyagramı gösterilmiştir. Giriş sayısı n dir. $k \ll 2^n$ olan k tane VE kapısı vardır . Öyle ki her bir kapının giriş yelpazesi n dir; ancak $2n$ bağlantı vardır. (n) tanesi değişkenin kendisi diğer (n) tanesi ise değişkenin tümleyeni için gereklidir. Gerçekleştirilecek fonksiyona göre, bu VE kapılarının girişlerine ilişkin $2n \times k$ bağlantının açık devre yapılması veya kısa devre olarak tutulması gerekir. Çıkış sayısı m kadar, VEYA kapısı vardır. Bu kapılara k tane VE kapısı, $k \times m$ bağlantı ile bağlanmıştır. Gerçekleştirilecek fonksiyonda kaç çarpımlar terimi varsa, onlara ilişkin bağlantılar kısa devre olarak tutulmalı, diğerlerine ilişkin bağlantılar ise açık devre kılınmalıdır. (m) tane çıkışta fonksiyonun kendisi veya tümleyeni elde edilebilir. Bir çıkış fonksiyonunun kendisi çıkışa aktarılacak isteniyorsa ona ilişkin bağlantı kısa devre olarak bırakılmalı, tümleyeni elde edilmek isteniyorsa ilgili bağlantı açık devre yapılmalıdır. Çıkışta ki m bağlantı bize çıkış fonksiyonlarının kendileri veya tümleyenlerini çıkışa almamızı sağlar. Bunun faydasını bundan sonraki bölümde PLA in programlanmasında göreceğiz.

Şimdi bir örnek olarak basit iki fonksiyonun PLA ile gerçekleştirilmesinde, bağlantıların nasıl seçileceğini inceleyelim. Pratikte böyle bir gerçekleştirme PLA elemanı ile yapılmaz, anlaşılması kolay olsun diye bu örnek yapılmaktadır. Pratikte PLA elemanları çok sayıda fonksiyonların tek bir elemanla gerçekleştirilmesinde yaygın olarak kullanılır.

Gerçekleştirilecek İki-değişkenli iki fonksiyonun doğruluk tablosu ve Karnaugh diyagramları Şekil 5.19 da verilmiştir. Görüldüğü gibi gerçekleştirilecek fonksiyonlarda ortak olan çarpımlar terimi için aynı VE kapısı kullanılmaktadır. Bir amaç ta böyle ortak olan çarpımlar terimleri sayısını maksimize etmektir.

x_1	x_2	x_3	F_1	F_2	Çarpım terimleri	Girişler	çıkışlar
0	0	0	0	0		x x x	F F
0	0	1	0	0	1 $x_1x'_2$	1 0 -	1 -
0	1	0	0	0	2 x_1x_3	1 - 1	1 1
0	1	1	0	1	3 x_2x_3	- 1 1	- 1
1	0	0	1	0			
1	0	1	1	1			
1	1	0	0	0			
1	1	1	1	1			

PLA program tablosu

x_1x_2	00	01	11	10
x_3				
0				1
1			1	1

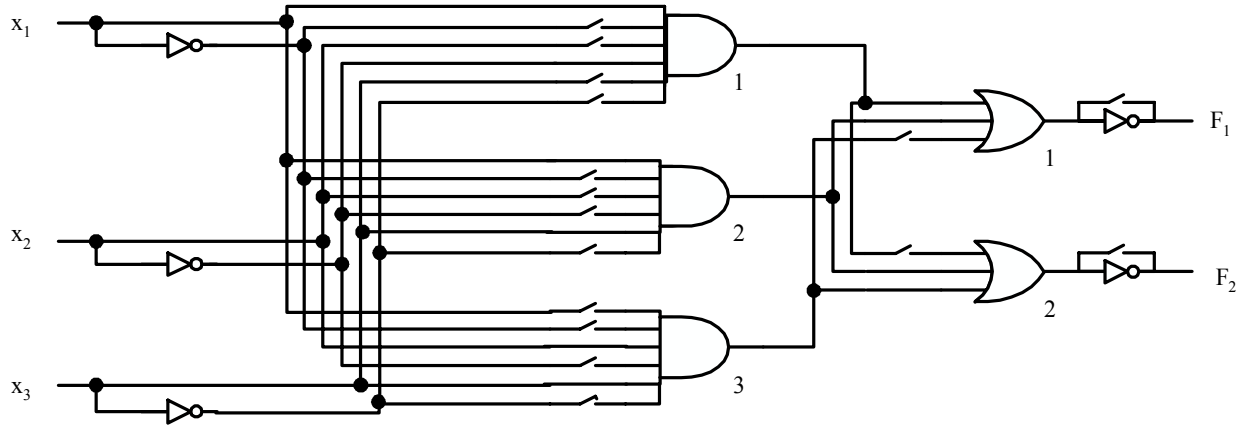
$$F_1 = x_1x'_2 + x_1x_3$$

x_1x_2	00	01	11	10
x_3				
0				
1		1	1	1

$$F_2 = x_1x_3 + x_2x_3$$

Şekil 5.19 PLA elemanı ile gerçekleştirilecek örnek iki değişkenli iki fonksiyon

Bu örnek fonksiyonların PLA ile gerçekleştirilmesinde, hangi bağlantıların açık devre yapılacağı, hangi bağlantıların ise kısa devre olarak bırakılacağı, Şekil 5.20 de gösterilmiştir.



Şekil 5.20 Örnek iki fonksiyonun PLA gerçekleştirilmesi

PLA program tablosu

Yukarıda örnek fonksiyonunun PLA elemanı ile gerçekleştirilmesinde önce fonksiyonlar indirgendi. ROM larla gerçeklemede fonksiyonlar indirgenmemişti. Ondan sonra bağlantıların açık/kısa devre yapılmasına geçildi. Bağlantıların sistematik bir biçimde nasıl yapılacağını gösteren tabloya PLA program tablosu denir. Bu tabloda satırlarda, indirgenmiş fonksiyonlardaki farklı çarpımlar terimleri gösterilir. Birinci sütunda her bir çarpımlar terimi için seçilen VE kapılara numaralandırılarak gösterilir. Bundan sonraki sütunlarda değişkenlere ayrılmıştır. 1. satırdaki VE kapısının girişleri için değişkenin kendisi varsa 1, tümleyeni varsa 0, değişken yoksa (-) konulur. Bu işlem bütün VE

kapıları için tekrarlanır. Şekil 5.18 deki blok diyagramında $2n \times k$ bağlantının açık devre mi, kısa devre mi yapılacağı belirlenmiş olur. Son sütunlarda ise çıkış fonksiyonlarının her biri temsil edilir. Bir çıkış fonksiyonunda hangi VE kapıları varsa ilgili yere 1, olmayan VE kapılarına ilişkin yere ise (-) konur. Şekil 5.18 deki blok diyagramında $k \times m$ bağlantılarından hangilerinin açık hangilerinin kısa devre yapılacağı belirlenmiş olur. Bundan önceki örneğe ilişkin PLA program tablosu Şekil 5.19 da verilmiştir.

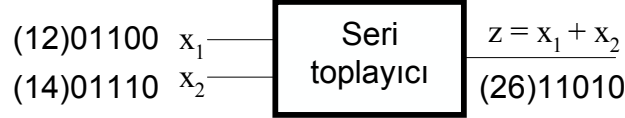
PLA lerde fonksiyonun kendisinde tümleyeni de gerçekleştirilebilmektedir. O nedenle ortak çarpımlar terimleri sayısını maksimumlaştırabilmek için gerçekleştirilmesini istediğimiz fonksiyonların tümleyenlerini de düşünebiliriz.

BÖLÜM 6 ARDIŞIL MANTIK DEVRELERİ

Şimdiye kadar ki beş bölümde, çıkışları girişleriyle yegane olarak belirlenebilen kombinezonsal mantık devrelerinin, analiz ve sentezini inceledik. Bu bölümde de çıkışları giriş değişkenlerinden başka, durum değişkenleri dediğimiz bellek elemanlarına ilişkin değişkenlerle birlikte ancak belirlenebilen **ardışıl mantık** devrelerini inceleyeceğiz. Ardışıl mantık devreleri kısaca **ardışıl devreler** olarak adlandırılır. Yaygın olarak bu devrelere **ardışıl makineler** da denir. Ardışıl makine incelememizde sırasıyla ardışıl makinelerin modellendirilmesi, sınıflandırılması, analiz ve sentezi ile sentezde karşılaşılan sorunları ele alacağız. Daha sonra kombinezonsal devrelerde olduğu gibi, ardışıl MSI ve LSI elemanlarının yapı ve işlevlerini inceleyeceğiz.

6.1 Giriş

Şekil 6.1 deki seri toplayıcı devresini düşünelim. x_1 , x_2 girişleri en az ağırlıklı basamaktan en çok ağırlıklı basamağa doğru, zamanda sıralı, iki tabanındaki iki sayıyı gösterebilir. z çıkışı da bu iki tabanındaki sayıların toplamı olan sayıyı, basamak ağırlığına göre zamanda sıralı olarak gösterebilir. varsayalım ki $x_1 = 12$ ve $x_2 = 14$ sayıları toplanacak. Giriş ve çıkış aşağıdaki Şekil 6.1 deki gibidir.



Şekil.6.1. Seri toplayıcı.

		2^4	2^3	2^2	2^1	2^0
		t_4	t_3	t_2	t_1	t_0
(12)	x_1	0	1	1	0	0
(14)	x_2	0	1	1	1	0
(26)	z	1	1	0	1	0

t_0 ve t_4 anlarındaki girişler (00) aynı olduğu halde çıkışlar farklıdır. Benzer biçimde t_2 ve t_3 anlarındaki girişler (11) aynı olduğu halde çıkışlar farklıdır. Görülüyor ki çıkış yalnızca girişler tarafından belirlenememektedir. O halde devre kombinezonsal değil, ardışıldır. Çıkışın belirlenebilmesi için her t_i anında, bir evvelki t_{i-1} anına ilişkin toplamın "elde" sinin bilinmesi gerekmektedir. Bu "elde"yi, makinenin herhangi bir t anındaki durumunu belirleyen bir değişken (y) olarak düşünelim. Öyle ki değişkenin t_i anında aldığı değer (Y), t_{i-1} anında bu değişkenle (y), giriş değişkenlerinin aldığı değerler tarafından belirlenmiş olsun. Yani $Y = f(x_1, x_2, y)$. (y) ye şimdiki durum, (Y) ye de bir sonraki durum, y değişkenine de durum değişkeni denir. Örneğimiz için toplam

elde(y)	1	0	0	0
x_1	0	1	1	0
x_2	0	1	1	1
z	1	1	0	1

biçiminde bulunur. Artık çıkış değişkeni giriş değişkenleri ve 0 andaki durum değişkeni tarafından yegane olarak belirlenmiştir.

6.2 Ardışıl Makinelerin Modellendirilmesi

Ardışıl makineler matematiksel, tablo ve diyagram olmak üzere üç farklı biçimde modellendirilebilirler.

Matematiksel model : Yukarıda örnekteki seri girişli toplama devresi gibi, sonlu durumlu ardışıl makineleri, $M = (G, Ç, D, Y, Z, D^0)$ ile verilen bir matematiksel modelle temsil edebiliriz.

G: Makinenin farklı girişlerinden oluşan giriş kümesidir. $\#G = p$ ise girişleri temsil edebilmek için gerekli olan giriş değişkeni sayısı n ; $n \geq \log_2 p$ eşitsizliğini sağlayan en küçük tam sayıdır. $\mathbf{X} = \{x_1, x_2, \dots, x_n\}$ **bağımsız giriş değişkenleri** kümesi olsun.

Ç: Makinenin çıkışında görülen, farklı çıkışlardan oluşan çıkış kümesidir. $\#Ç = g$ ise, çıkışları temsil edebilmek için gerekli olan çıkış değişkeni sayısı m ; $m \geq \log_2 g$ eşitsizliğini sağlayan en küçük tam sayıdır. $\mathbf{Z} = \{z_1, z_2, \dots, z_m\}$ çıkış değişkenleri kümesidir.

D: Makinenin farklı durumlarından oluşan kümedir. $\#D = \mu$ ise, gerekli olan durum değişkeni sayısı r , $r \geq \log_2 \mu$ eşitsizliğini sağlayan en küçük tam sayıdır. $\mathbf{y} = \{y_1, y_2, \dots, y_r\}$ **bağımsız durum değişkenleri** kümesidir.

$\mathbf{Y} = \{Y_1, Y_2, \dots, Y_r\}$ bir sonraki durum değişkenleri kümesidir. Bu küme elemanları her biri $Y_i^{t+1} = Y_i(x_1^t, x_2^t, \dots, x_n^t, y_1^t, y_2^t, \dots, y_r^t) = Y_i(\mathbf{X}^t, \mathbf{y}^t) : S^{n+r} \rightarrow S, S = \{0,1\} \quad i = 1, 2, \dots, r$ biçiminde tanımlanan r tane **bir sonraki durum fonksiyonudur**. t şimdiki anı, $t+1$ bir sonraki anı göstermektedir.

$\mathbf{Z} = \{z_1, z_2, \dots, z_m\}$ bağımlı çıkış değişkenleri kümesidir. Bu küme elemanlarının her biri $Z_i^t = Z_i(x_1^t, x_2^t, \dots, x_n^t, y_1^t, y_2^t, \dots, y_r^t) = Z_i(\mathbf{X}^t, \mathbf{y}^t) : S^{n+r} \rightarrow S, S = \{0,1\} \quad i = 1, 2, \dots, m$ biçiminde tanımlanan m tane **çıkış fonksiyonudur**.

Z_i fonksiyonları $x_1 \dots x_n$ bağımsız giriş ve $y_1, \dots, y_r$ bağımsız durum değişkenlerinin fonksiyonu ise makineye **"Mealy makinesi"**, Z_i fonksiyonları yalnızca, $y_1, \dots, y_r$ durum değişkenlerinin fonksiyonu ise makineye **"Moore makinesi"** denir.

$\mathbf{Y}$ fonksiyonlarında, fonksiyonların aldığı değer $(t+1)$ anındaki değer; bağımsız değişkenlerin aldığı değer ise, t anındaki değerlerdir. $\mathbf{Z}$ fonksiyonlarında ise değişkenler ve fonksiyonların aldığı değerler aynı t anı için geçerlidir. Bundan sonra fonksiyonların t ile ilişkisi ayrıca simgelenmeyecektir.

D^0 : Başlangıç durumudur. Yani bağımsız durum değişkenlerinin t_0 anındaki aldıkları değerlere karşı gelen $(y_1^0, y_2^0, \dots, y_r^0)$ durumudur.

Örnek 6.1. Seri toplayıcı için M matematiksel modelini çıkaralım.

G giriş kümesi ve değişkenleri: $G = \{G_1(x_1 = 0, x_2 = 0); G_2(x_1 = 0, x_2 = 1); G_3(x_1 = 1, x_2 = 0); G_4(x_1 = 1, x_2 = 1)\}$ dir.

Ç çıkış kümesi ve değişkenleri $Ç = \{Ç_1(z_1 = 0), Ç_2(z_1 = 1)\}$ dir.

D durum kümesi ve değişkenleri $D = \{D_1(y=0) \text{ elde } 0, D_2(y=1) \text{ elde } 1\}$ dir. (y) bağımsız durum değişkenidir.

$\mathbf{Y}$ bir sonraki durum ve $\mathbf{Z}$ çıkış fonksiyonları Tablo 6.1. de doğruluk tablosuyla verilen

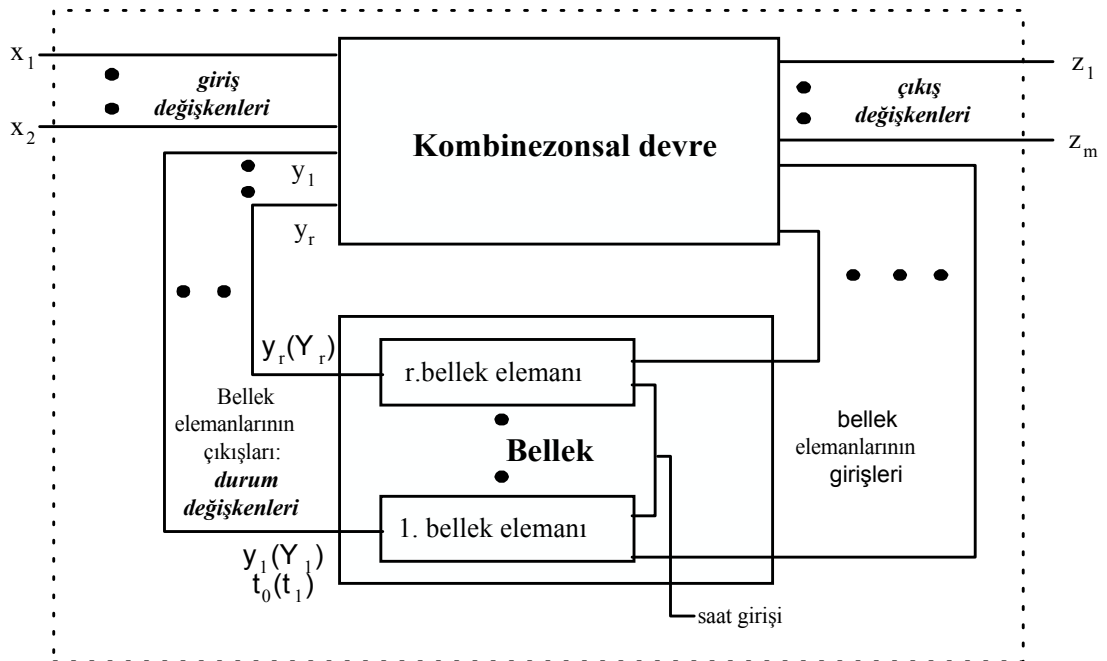
$$Y^{t+1} = y' x_1 x_2 + y' x_1' x_2 + y x_1 x_2' + y x_1' x_2' \text{ ve}$$

$$Z = y' x_1' x_2 + y' x_1 x_2' + y x_1' x_2' + y x_1 x_2 \text{ fonksiyonlarıdır.}$$

Tablo.6.1.Seri toplayıcıya ilişkin bağımsız değişkenler ve bir sonraki durum ve çıkış fonksiyonlarının doğruluk tablosu.

giriş x_1	giriş x_2	durum y	Y bir sonraki durum fonksiyonu.	Z çıkış fonksiyonu
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

Matematiksel olarak $M = (G, \mathcal{C}, D, Y, Z, D^0)$ ile verilen bir ardışıl makine, genel olarak



Şeki1.6.2.deki bir devreye karşı düşürülebilir. Devre iki temel parçadan oluşur.

Şekil.6.2. Bir ardışıl makinenin en genel yapısı.

Birincisi kombinezonsal parça ikincisi ise bellek parçasıdır. Kombinezonsal parçanın girişleri, sistemin giriş değişkenleri ($x_1, x_2, \dots, x_n$) ile bağımsız durum değişkenleri ($y_1, y_2, \dots, y_r$) dir. Kombinezonsal parçanın çıkışları da makinenin çıkış değişkenleri ($z_1, z_2, \dots, z_m$) ile bellek elemanlarının giriş fonksiyonlarıdır.

Bellek elemanlarının çıkışları olan iletkenlerde, zaman ekseninde önce y bağımsız değişkenlerinin sonra bağımlı bir sonraki durum değişkenlerinin aldığı değerler gözlenir. Ard arda gelen iki girişten, ikincisine ilişkin şimdiki durum, birinci girişe ilişkin bir sonraki durumdur. Seri toplayıcı örneğinde t_0 anındaki girişler için olan bir sonraki durum 0, t_1 anındaki girişler için şimdiki durumdur. Benzer biçimde t_2 anındaki şimdiki durum 0; bir sonraki durum (elde) ise 1 dir. Bu yeni elde t_3 anındaki girişler için şimdiki durumdur. Bellek elemanları bundan sonraki alt bölümde detaylı olarak incelenecektir.

Matematiksel olarak temsil edilen ardışıl makine, durum diyagramları veya durum tablolarıyla da temsil edilebilir.

Durum tablosu (Geçiş tablosu, transition table): Satırlarında durumlar (D kümesinin elemanları), sütunlarında girişlerin (G kümesinin elemanları) bulunduğu matristir. i. satır ve j. sütunun kesiştiği yerde; makinenin i. satıra ilişkin durumunda (D_i), j. sütuna ilişkin giriş (G_j) geldiğinde ki bir sonraki durum (D_k) ile o andaki çıkış (C_k) yazılıdır (Tablo 6.2a). Özel olarak örnek 6.1 deki seri toplayıcı için durum tablosu Tablo 6.2b de verilmiştir. Bu tabloda girişler ve durumlar ilgili değişkenler cinsinden ifade edilmiştir.

Tablo.6.2. Ardışıl makineye ilişkin durum tablosu.

	G_1	G_2	G_j	G_p
D_1				
D_2				
D_i			D_k, C_k	
D_μ				

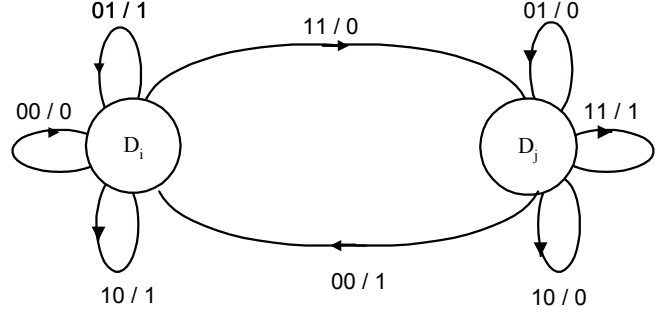
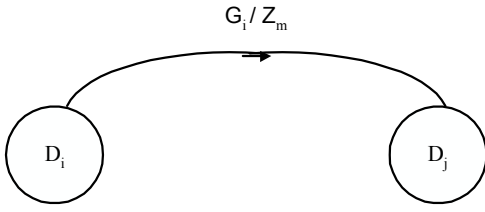
(a) Bir ardışıl makineye ilişkin genel durum tablosu

	x_1x_2 00 giriş	X_1x_2 01 giriş	x_1x_2 11 giriş	x_1x_2 10 giriş
y=0 şimdiki durum	Y,Z 0,0	Y,Z 0,1	Y,Z 1,0	Y,Z 0,1
y=1 şimdiki durum	Y,Z 0,1	Y,Z 1,0	Y,Z 1,1	Y,Z 1,0

(b) Seri toplayıcıya ilişkin durum tablosu

Durum diyagramı, düğümler ve topolojik elemanlardan oluşmuş bir diyagramdır. Düğümler durumlara karşı düşer. Makine D_i durumunda iken G_j girişi altında D_j bir sonraki durumuna gidiyor ve çıkış z_m ise, D_i düğümünden D_j düğümüne yönlendirilmiş bir topolojik eleman çizilir ve üzerine G_j / z_m yazılır. Şekil 6.3 a da durum diyagramının tipik

bir parçası gösterilmiştir. Örnek 6.1 deki seri toplayıcı için durum diyagramı Şekil 6.3b de verildiği gibidir. Girişler ve durumlar ilgili değişkenler cinsinden verilmiştir.



(a) Genel olarak durum diyagramı

(b) Toplayıcıya ilişkin durum diyagramı

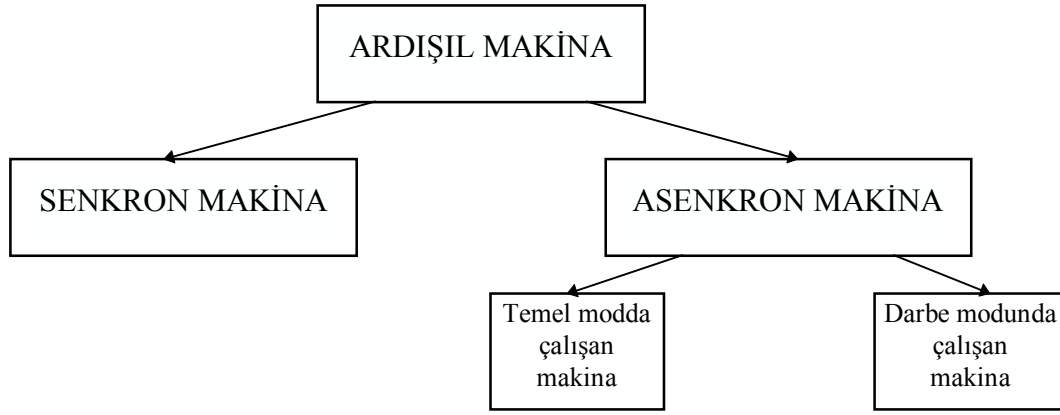
Şekil.6.3. Durum diyagramları.

6.3 Ardışıl Makinelerin Sınıflandırılması

Ardışıl makineler senkron (saatli) ve asenkron (saatsiz) makineler olmak üzere iki ana sınıfa toplanırlar. **Senkron makinelerde** periyodik saat darbeleri üreten bir merkezi saat (master-clock generator) vardır. Bütün bellek elemanlarının saat girişlerine, bu merkezi saat çıkışı bağlanmıştır (bakınız Şekil 6.2). Makinenin durum değiştirmesi, saat darbesi geldiği zaman mümkündür. Yeni bir saat darbesi gelene kadar, makinenin durumunda bir değişiklik olmaz. Makinenin kombinezonsal kısmı ise saat darbesinden bağımsız çalışır. Her bir giriş için bir saat darbesi uygulanmalıdır. Makinenin çalışma hızı saat darbelerinin periyoduyla sınırlanmıştır. örneğin seri toplayıcı devre bir senkron ardışıl makinedir.

Asenkron makinelerde saat darbeleri yoktur. Temel modda (fundamental mode) ve darbe modunda (pulse mode) çalışanlar olmak üzere iki farklı türden olabilirler. Temel modda çalışan asenkron makinelerde girişler seviye biçimindedir, yani yeni bir giriş gelene kadar eski girişler durarak sistemin kararlı (stabil) bir duruma gelmesini sağlar. Yarış problemi olmaması için birden fazla giriş değişkeni aynı anda değişmez. Darbe modunda çalışan makinelere gelince, girişleri darbe biçimindedir. Herhangi bir anda giriş değişkenlerinden yalnız biri 1 olur, diğerleri 0 olmak zorundadır. Farklı girişler arasında bütün girişler birlikte 0 olmak zorundadırlar. Şekil 6.4 de ardışıl makinelerin sınıflandırılması gösterilmiştir.

Bu dersin kapsamı içinde bilgisayarlarda kullanılan senkron makineler ağırlıklı olarak incelenecek, çoğunlukla kontrol sistemlerinde kullanılan asenkron makineler ise incelenmeyecektir.

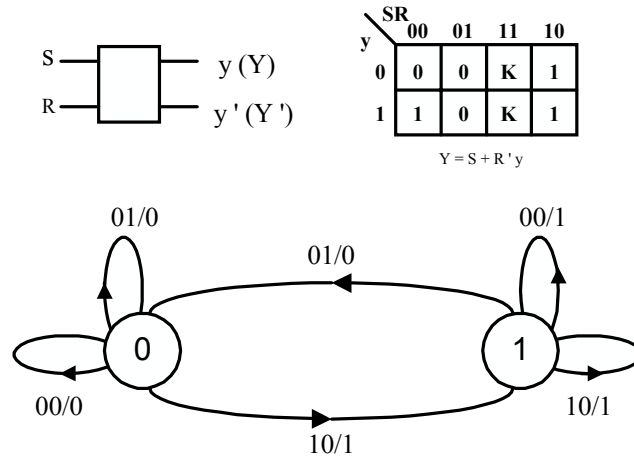


Şekli.6.4. Ardışıl makinelerin sınıflandırılması.

6.4 Bellek Elemanları

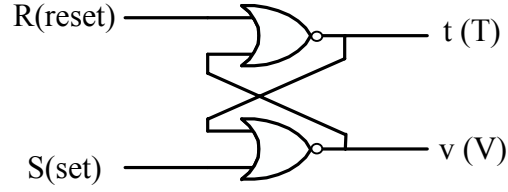
Ardışıl devrede her bir durum değişkenine ilişkin bir **bellek (flip flop)** elemanı vardır. Bundan dolayı bellek elemanları 1 bitlik bir bilginin saklanmasını sağlarlar. Çıkış şimdiki durum değişkeni y ve aynı zamanda bir sonraki durum fonksiyonu olan Y dir. Ayrıca durum değişkeninin veya bir sonraki durum değişkeninin tümleyeni de çıkış olarak vardır. Şimdi belli başlı bellek elemanlarını teker teker inceleyelim.

SR bellek elemanı: iki girişli S(set), R(reset) ve iki çıkışlıdır (y, y') y' f devrenin yapısı dolayısıyla doğrudan elde edildiğinden, ikinci bir çıkış değerlendirilmiştir. Bütün diğer bellek elemanlarında da durum aynıdır, yani y ve y' çıkışları vardır. SR bellek elemanın tanımı bir sonraki durum fonksiyonu, durum tablosu ve durum diyagramı olarak şekil 6.5 te verilmiştir. (11) girişinin hiçbir zaman uygulanmayacağı varsayılarak çıkışlar k =keyfi olarak alınmıştır. SR bellek elemanının çalışması, girişler birbirinin zıttı iken 'S nin dediğini yapıyor' biçiminde de yorumlanabilir.



Şekil 6.5 SR bellek elemanı

Şimdi Şekil 6.6 te verilen devreyi analiz ederek, Şekil 6.5 te tanımlanan SR flip flopuna karşı düştüğünü gösterelim. Şekil 6.6 teki devrenin çıkışlarını, başlangıçta sanki biri diğerinin tümleyeni değilmiş gibi düşünüp, sonra bu çıkışların birbirlerinin tümleyeni olduklarını gözleyelim



a) TÜVEYA kapılarıyla gerçekleştirilen SR flip flopu

t	v	S	R	T	V
0	0	0	0	0	1
				1	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	0	0
1	1	0	0	0	1
				1	0
1	1	0	1	0	1
1	1	1	0	1	0
1	1	1	1	0	0
0	1	0	0	0	1
0	1	0	1	0	1
0	1	1	0	1	0
0	1	1	1	0	0
1	0	0	0	1	0
1	0	0	1	0	1
1	0	1	0	1	0
1	0	1	1	0	0

TÜVEYA kapılarının gecikmelerine bağlı olarak iki hal (01 veya 10) olabilir, belirsizlik var

TV(00) ara durumundan sonra TÜVEYA kapılarının gecikmelerine bağlı olarak iki hal (01 veya 10) olabilir, belirsizlik var

b) Devrenin en genel durumda analizi

t	v	S	R	T	V
0	1	0	0	0	1
0	1	0	1	0	1
0	1	1	0	1	0
1	0	0	0	1	0
1	0	0	1	0	1
1	0	1	0	1	0

c) Devrenin girişlerine 11 gelmeyeceği koşulu altında sadeleşmiş doğruluk tablosu

y	S	R	Y	Y'
0	0	0	0	1
0	0	1	0	1
0	1	0	1	0
1	0	0	1	0
1	0	1	0	1
1	1	0	1	0

d) Sonuçta varılan doğruluk tablosu

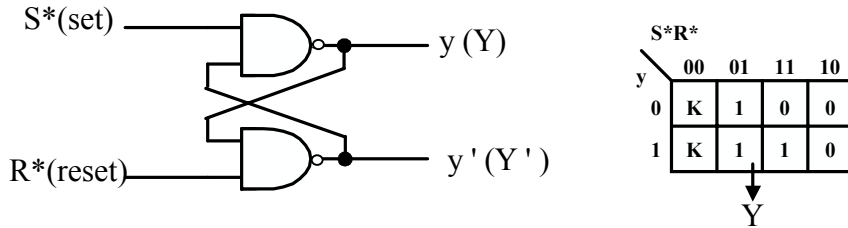
Şekil 6.6 SR belek elemanın devresi

Şekil 6.6 b deki tabloda devrenin girişine hiçbir zaman SR (11) uygulanmayacağı düşünülürse TV nin hiçbir zaman TV(00) durumuna gelmeyeceği görülür. Bu varsayım altında, tablodaki ilk dört satır silinebilir. Diğer taraftan TV sütunlarında 11 durumu hiç görünmemektedir. O nedenle TV(11) e karşı gelen, tablodaki ikinci dörtlü satır da silinebilir. Girişine hiçbir zaman 11 gelmeme koşulu altında basitleştirilmiş tablo Şekil 6.6 c de verilmiştir. Bu tabloda ki çıkışların her biri bir diğerinin tümleyenidir. O halde $V=T'$ dır. Şekil 6.6 d de varılan sonuç tablo, SR flip flopunun tanım bağıntısına denktir.

SR belek elemanının çalışması, girişler birbirinin zıttı iken S nin dediğini yapıyor biçiminde de yorumlanabilir.

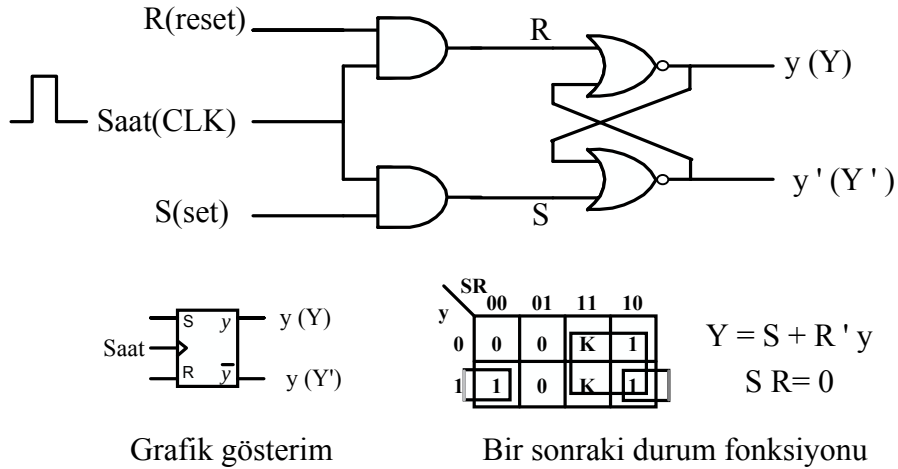
TÜVEYA kaplılarıyla gerçekleştirilen devreye benzer TÜVE kaplılarıyla gerçekleştirilen bir devre Şekil.6.7 de verilmiştir. TÜVEYA' lı devredeki 11 girişinin özelliği TÜVE li devrede 00 girişi altında oluşur ve S ile R yer değiştirmiş gibi görülür. Bu elemanı S^*R^* ile gösterelim. S^*R^* belek elemanının çalışması, girişler birbirinin zıttı iken 'R nin dediğini yapıyor' biçiminde de yorumlanabilir.

TÜVE' li devre ilerde tanımlayacağımız gecikme (Delay) belek elemanının gerçekleştirilmesinde kullanılacaktır. Şekil 6.5 ve 6.7 de verilen devrelere **temel belek** elemanları denir.



Şekil 6.7 TÜVE kaplılarıyla gerçekleştirilen bir temel belek elemanı

Saatli SR belek elemanı: Yukarıda tanıttığımız SR belek elemanı asenkron makinelerde kullanılır. Saatli (Senkron) makinelerde ise saatli belek elemanları kullanılır. Şimdi saatli SR belek elemanını inceleyelim. SR girişlerini birer uçları saat darbesine bağlı VE kapılarından geçirdiğimizde, Şekil 6.8 görülen ve tanım tablosu verilen, saatli SR belek elemanını elde ederiz. Saat girişi 0 olduğunda VE kapılarının çıkışları 0 olacak ve belek elemanı bulunduğu durumunu koruyacaktır. Saat girişi 1 olduğunda ise SR girişleri VE kapılarının çıkışlarına aktarılır. Belek elemanı girişlerin ve durumun aldığı değerlere bağlı olarak çalışır.



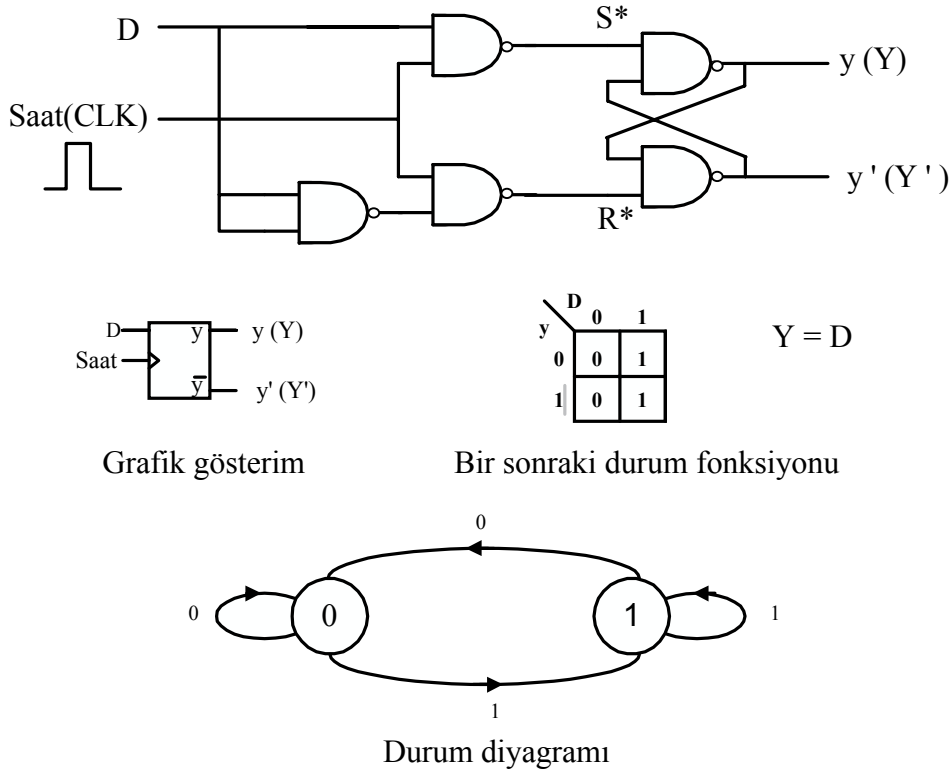
Grafik gösterim

Bir sonraki durum fonksiyonu

Şekil 6.8 Saatli SR belek elemanı devresi, grafik gösterimi ve tanım tablosu

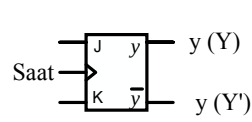
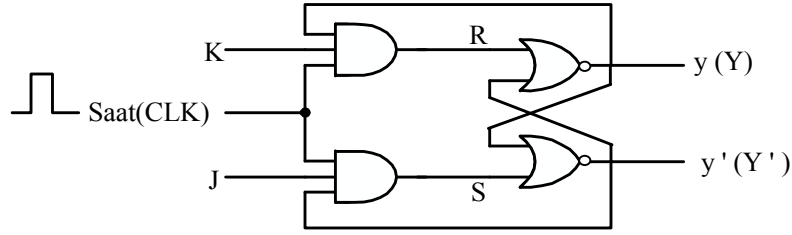
Tanım fonksiyonundaki y , saatten önceki şimdiki durumu; Y ise saatten sonraki bir sonraki durumu göstermektedir. O nedenle tanım fonksiyonunda t değişken olarak görülmemektedir. Tanım fonksiyonun ifadesi $SR=11$ girişlerinin hiçbir zaman uygulanmayacağı varsayımıyla bulunmuştur. $SR=0$ bağıntısı bu koşulu ifade etmektedir. Pratikte saatli SR bellek elemanı nadiren üretilir, ilerde göreceğimiz saatli D ve JK bellek elemanları daha yaygın bir biçimde üretilmektedir.

Saatli Gecikme D (delay) bellek elemanı: TÜVE kapılarıyla daha önce oluşturduğumuz devreyi temel alarak Şekil 6.9 da oluşturulan devre, D bellek elemanıdır. Girişine gelen işareti, iki saat darbesi aralığında çıkışında tutar. Bir başka deyişle giriş geciktirilerek, çıkışta tutulur. Birinci saat darbesinden sonra gelen girişler çıkışa etkili olmaz. Saat darbesi yokken (0) geri beslemeli TÜVE kapılarının girişleri 11 olup durum değişmez, yani bellek elemanı çalışmaz. Saat darbesi varken (1) $D = 1$ ise çıkış da 1, $D = 0$ ise çıkış 0 olur. Şekil 6.9 de Gecikme bellek elemanın devresi tanım tablosu (bir sonraki durum fonksiyonu) ve durum diyagramı verilmiştir.



Şekil.6.9. Gecikme bellek elemanı devresi, grafik gösterilimi, tanım tablosu ve durum diyagramı

Saatli JK bellek elemanı: SR bellek elemanı gibidir; yalnızca $JK=11$ girişi altında belirsizlik olmayıp, şimdiki durum zıddına dönüşür. Şekil 6.10 da devresi tanım tablosu, grafik gösterimi, ve durum diyagramı verilmiştir. Görüldüğü gibi TÜVEYA kapılarının çıkışları 11, biri diğerinin tümleyeni, olamayacağı için TÜVEYA kapılarının girişleri de hiçbir zaman 11 olmayacaktır, yani SR de görülen belirsizlik burada görülmeyecektir.

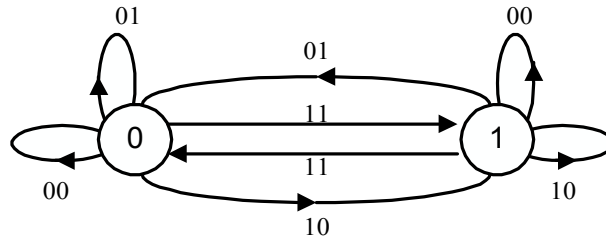


Grafik gösterim

JK	00	01	11	10
y	0	0	1	1
1	1	0	0	1

$$Y = J y' + K' y$$

Bir sonraki durum fonksiyonu

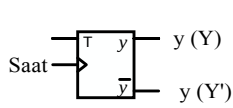
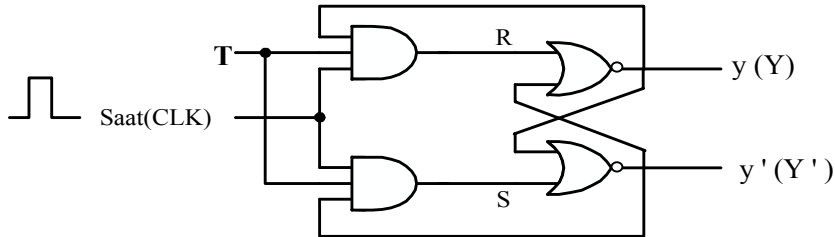


Durum diyagramı

Şekil 6.10 JK bellek elemanı devresi, tanım tablosu, grafik gösterilimi, durum diyagramı

Saatli T tetikleme bellek elemanı

Tek girişli bir bellek elamanıdır. Girişine 0 geldiğinde durumunu korur, 1 geldiğinde ise durumunu zıddına dönüştürür (togling). Bu elemana ilişkin bilgiler Şekil 6.11 da verilmiştir.

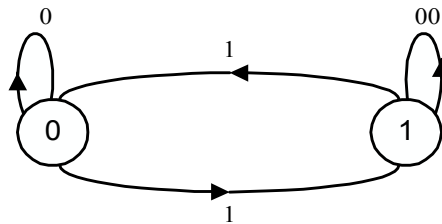


Grafik gösterim

T	0	1
y	0	1
1	1	0

$$Y = T y' + T' y$$

Bir sonraki durum fonksiyonu



Durum diyagramı

Şekil 6.11. T tekikleme bellek elemanı devresi ,tanım tablosu, grafik gösterimi ve durum diyagramı

6.5 Bellek elemanlarının tetiklenmesi

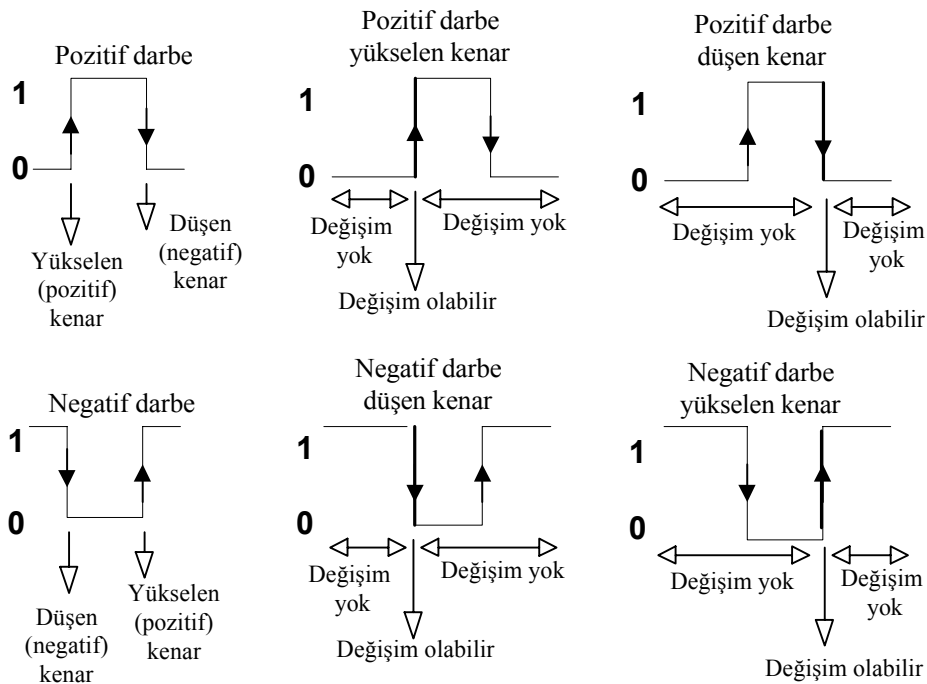
Bellek elemanlarının durumunun değişmesi, girişlerindeki anlık değişimlerle olmaktadır. Bu durum değişmelerine bellek elemanın tetiklenmesi (trigering) denir. Asenkron belleklerde bu değişimler seviye biçimindedir, giriş seviyesindeki değişimler durum değişimlerine neden olurlar. Saatli bellek elemanlarında ise tetikleme saat darbeleriyle sağlanır. Yani giriş işaretlerine bağlı olarak, ancak saat darbesi varken durum değişimi olabilir.

Saatli bellek elemanlarında (makinelere) tetikleme iki türdür. Saat işareti 1 değerini aldığı anda tetikleme (durum değişimi) olur, 0 değerini aldığı anda ise girişler değişse bile tetikleme olmaz. Böyle çalışan saatli bellek elemanlarına **latch** denir.

ikinci tür tetikleme, kenar tetiklemesidir. Saatli bellek elemanlarında (makinelere) saat darbesinin yükselen veya alçalan kenarında tetikleme (durum değişimi) olur, bunun dışındaki bütün aralıklarda (0 veya 1) ise girişler değişse bile tetikleme olmaz.

Saat darbesi Şekil 6.12 de görüldüğü gibi **pozitif** veya **negatif** olabilir. **Pozitif darbenin** 0 dan 1 e geçiş kenarına **yükselen** veya **pozitif kenar (leading , positive edge)**, 1 den 0 a geçiş kenarına da **düşen** veya **negatif kenar (trailing, negative edge)** denir. Benzer biçimde **negatif darbenin** 0 dan 1 e geçiş kenarına **yükselen** veya **pozitif kenar (leading , positive edge)**, 1 den 0 a geçiş kenarına da **düşen** veya **negatif kenar (trailing, negative edge)** denir.

Bir saatli kenar tetiklemeli bellek elemanı, ya yükselen kenarda yada düşen kenarda tetiklenebilir. Şekil 6 12 de pozitif ve negatif saat darbeleri için tetikleme kenarları gösterilmiştir.



Şekil 6.12 Saat darbeleri ve tetikleme kenarları

Şimdi saatsiz, saatli bellek elemanları ve saatli ardışıl devrelerde karşılaşılan tetikleme sorunlarını inceleyelim.

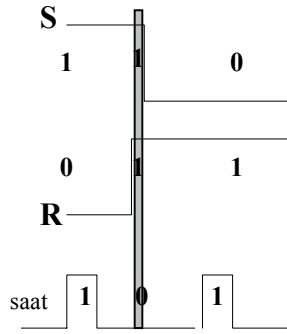
İki girişli bellek elemanlarında, ard arda iki giriş uygulandığını düşünelim. Öyleki iki giriş değişkenin de değişmesini gereksin. Örneğin 01-10 girişleri gibi. Pratikte bu girişlerin ard

arda uygulanması mümkün olmaz, çünkü herhangi bir giriş değişkenindeki gecikme 10-00-10 veya 01-11-10 dizisinin uygulanması sonucunu doğurur. Yani istemediğimiz halde arada 00 veya 11 girişi uygulanmış olur.

Şekil 6.6 ve 6.7 asenkron temel bellek elemanları için sırasıyla 11 ve 00 girişlerinin uygulanmaması gereken girişler olduğu belirtilmişti. Ortaya bir sorun çıkmaktadır. Çözüm Şekil 6.6 daki asenkron bellek elemanı için her bir girişten sonra 00 girişinin uygulanmasıdır. 01-10 giriş dizisi yerine 01-00-10 giriş dizisini uygularsak giriş değişkenlerinin ikisi birden aynı anda değişmek zorunda kalmaz ve istenmeyen 11 girişi arada gelemmez. Diğer taraftan bu bellek elemanı için 00 girişi altında durum değişmemektedir, sanki 01 den sonra 10 uygulanmış gibi olmaktadır. Şekil 6.7 deki asenkron temel bellek elemanı için benzer çözüm her bir girişten sonra 11 girişinin uygulanmasıdır.

Asenkron bellek elemanlarında tetikleme süresince durumun devamlı değişmesi de bir tetikleme sorunu yaratır. Örneğin asenkron JK bellek elemanı 11 girişi altında 0 ve 1 durumları arasında devamlı durum değiştirir. Bu olaya çevrim (cycling) denir. O nedenle uygulamada iki girişli asenkron bellek elemanı olarak JK değil SR bellek elemanı kullanılır.

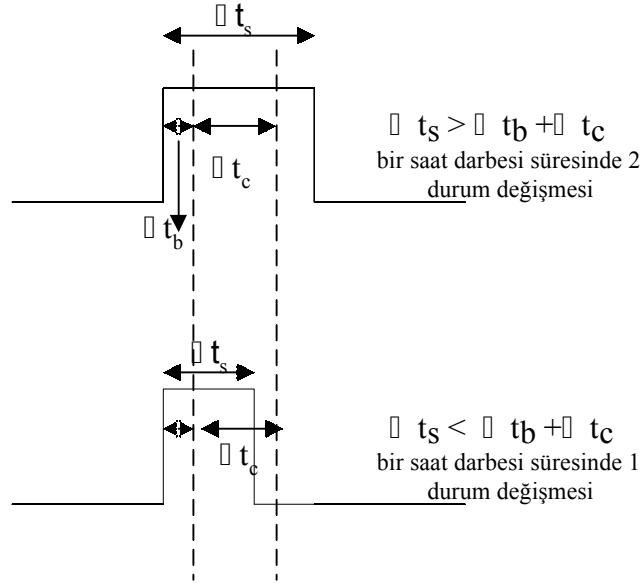
Yukarıda asenkron bellek elemanlarında istenmeyen girişler ve çevrim için tetikleme sorunlarını inceledik, şimdi de senkron bellek elemanları için inceleyelim. istenmeyen girişler saat darbelerinin olmadığı aralığa düşürülürse, saat darbeleri olmadığında bellek elemanı çalışmayacağından çözüme ulaşılır. Şekil 6.13 te bu çözüm canlandırılmıştır.



Şekil 6.13 Senkron SR bellek elemanı için istenmeyen girişlere ilişkin tetikleme sorununu çözümü

Senkron JK bellek elemanında 11 girişi altında görülen çevrimin oluşmasını önlemek için saat darbesinin yükselen veya düşen kenarında tetikleme yapılır. Tetikleme anlık olacağından, durum ancak bir defa değişebilecektir. Kenar tetiklemeli bellek elemanlarının yapısını ilerde ayrıca göreceğiz.

Bellek elemanlarında karşılaşılan tetikleme sorunlarının benzerleri en genel hali Şekil 6.2 de verilen ardışıl devreler içinde söz konusudur. Asenkron ardışıl devreler (makinelere) bu dersin kapsamı dışında tutulduğundan incelenmeyecek yalnızca senkron makineler incelenecektir.



Şekil 6.14 Senkron ardışıl devrede saat darbe uzunluğu

Senkron ardışıl devrelerde saat darbe uzunluğu (t_s), bellek elemanları gecikmesinden (t_b) daha fazla, bellek (t_b) ve kombinezonsal devredeki gecikmesi (t_c) toplamından ($t_b + t_c$) daha küçük olmalıdır. Yani $t_b < t_s < t_b + t_c$ olmalıdır. Bellek elemanlarının çıkışında yeni bir durumu gözleyebilmek için $t_b < t_s$ olmalıdır. $t_s > t_b + t_c$ olursa ise bellek elemanları 1 saat darbesi zaman aralığında iki veya daha fazla çalışmış olur (Şekil 6.14). Oysa istenen her bir saat darbesi için, belli bir giriş altında, yalnızca bir durum değişikliğinin olmasıdır. Bu bağıntının sağlanması için senkron bellek elemanlarını, kenar tetiklemeli yapmak uygun olur. Tetikleme darbenin bir kenarında olacağından, t_s süresi minimuma inecek ve $t_s < t_b + t_c$ eşitsizliği her zaman sağlanacaktır. Kenar tetiklemeli D bellek elemanı için $t_b < t_s$ ilişkisi, bir örnek olarak, bu bölümde ayrıca incelenecektir.

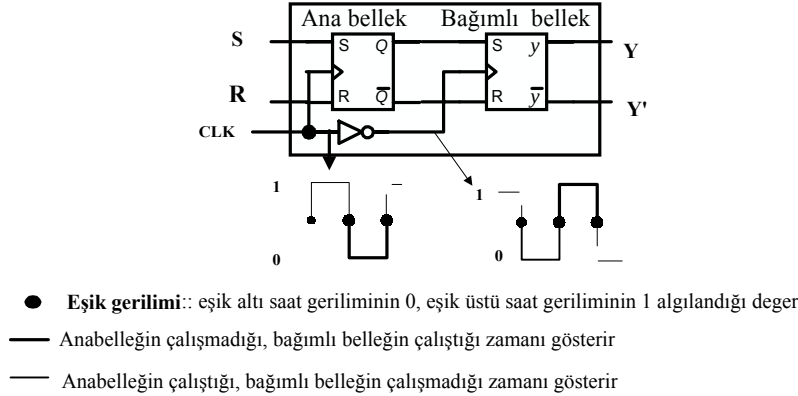
Senkron ardışıl devrelerde karşılaşılan bir başka tetikleme sorunu da, birden fazla durum değişkeninin aynı anda değer değiştirmesidir. Bellek elemanın çıkışının gecikmelerindeki farklılıklar dolayısıyla devre istenmeyen durumlardan geçebilir. Örneğin iki durum değişkenli bir makede 01 durumundan 10 durumuna geçerken arada 00 (veya 11) istenmeyen durumundan geçilebilir. Kenar tetiklemeli bellek elemanları kullanıldığında bu sorunla karşılaşılmaz. Nedeni istenmeyen durum oluştuğunda tetikleme işleminin tamamlanmış olmasıdır. Bellek elemanları, yeni bir tetikleme oluncaya kadar (yeni bir darbe kenarına kadar) çalışmazlar.

Bellek elemanlarının saat darbelerinin kenarlarında tetiklenmesi, üç farklı biçimde yapılabilir. Birincisi bir RC devresiyle keskin darbeler (spike) elde etmektir. ikincisi ana ve bağımlı (master slave) , üçüncüsü ise kenar tetiklemeli bellek elemanlarıdır. Son ikisini ayrı ayrı inceleyeceğiz.

Saat darbelerinin kenarında tetiklenen bellek elemanların bulunduğu ardışıl devrelerin daha hızlı çalıştığı söylenemez. Devrenin çalışma hızına tesir eden saat darbesinin 1 olma süresi değil, saat darbelerinin frekansıdır.

Ana-bağımlı (Master slave) bellek elemanı:

Ana-bağımlı bellek elemanları, ardarda bağlı iki bellek elemanı ve bir tümleyen kapısından oluşur. Birinci bellek elemanına aa, ikincisine ise bağımlı bellek elemanları denir. SR nin ana-bağımlı yapısı Şekil 6.15 de verilmiştir.



Şekil 6.15 Ana-bağımlı (master-slave) SR bellek elemanı

Bellek elemanları, saat girişlerini, saat darbesinin yükselen ve düşen kenarlarında, belli bir eşik geriliminin altında 0, üstünde ise 1 olarak algılanırlar (Şekil 6.15). Ana- bağımlı bellek elemanının saat girişindeki pozitif saat darbesi, tümleyen kapısı nedeniyle bağımlı bellek elemanının saat girişinde negatif darbe oluşturur.

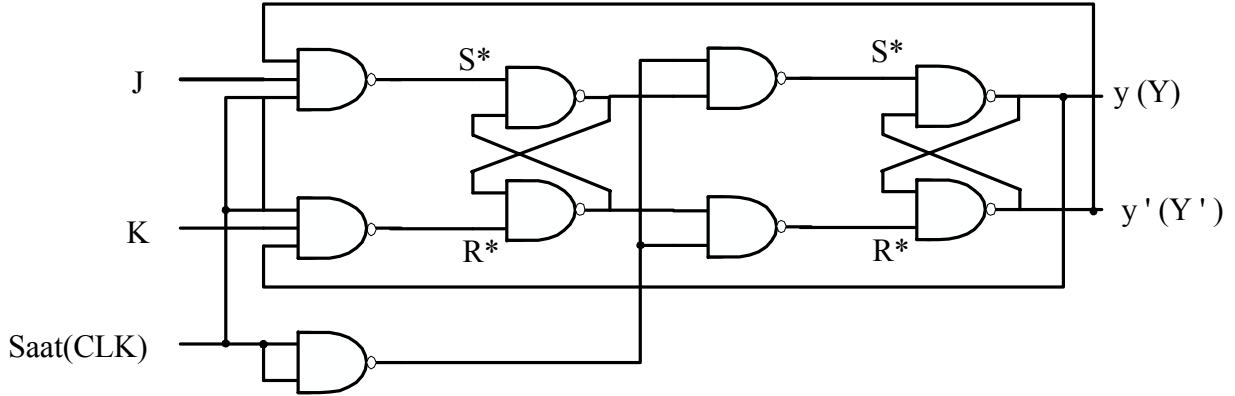
Buna göre girişteki saat darbesi 0 dan 1 e geçtiğinde ve 1 olduğunda (şekilde ince çizgi) bağımlı bellek elemanı çalışmaz, çünkü bağımlı bellekteki saat girişi (şekilde ince çizgi) 0 olmuştur. Ana bellek elemanı ise çalışır, girişlerine bağlı olarak, SR tanımına uygun çıkışları verir. Girişteki saat darbesi 1 den 0 a düşerken ve 0 olduğunda (şekildeki kalın çizgi) ana bellek çalışmaz. Bağımlı bellek çalışır çünkü bağımlı bellek girişindeki darbe 0 dan 1 e yükselir ve 1 değerini alır (şekildeki kalın çizgi). Bağımlı bellek girişi Q, Q' olduğundan tanım gereğince $Y = Q$ ve $Y' = Q'$ olur.

Şimdi ana-bağımlı bellek devresinin çalışmasını düşündüğümüzde; girişteki saat darbesinin yükselen kenarında, tanım uyarınca SR girişlerine göre ana bellek elemanın çıkışları oluşur . Bu çıkış bağımlı bellek elemanın girişinde kalır çünkü ikinci bellek elemanı çalışmamaktadır. Saat darbesinin düşen kenarında ise ikinci bellek elemanın girişleri aynen Y, Y' çıkışlarında birlikte görülecektir, çünkü SR için girişler birbirinden farklı ise tanım gereğince $Y=S$ dir. Yani ana-bağımlı bellek elemanı saat darbesinin düşen kenarında, girişler için SR bellek elemanı tanımına uygun çıkışları vermektedir.

Ana-bağımlı bellek elemanının saat darbesinin yükselen kenarında çalışmasını saat girişine ilave bir tümleyen kapısı bağlayarak sağlamak mümkündür. Bu durumda saat darbeleri negatif darbeler olarak düşünülmelidir (Şekil 6.12); düşen kenarda ana bellek elemanı yükselen kenarda ise bağımlı bellek elemanı ve dolayısıyla SR ana-bağımlı bellek elemanı çalışacaktır.

Yukarıda SR bellek elemanı için incelediğimiz ana-bağımlı yapı, diğer tip bellek elemanları için de benzer biçimde gerçekleştirilir. Örneğin D bellek elemanının çıkışlarını bir SR bellek elemanı girişlerine bağlayarak ana-bağımlı bir D bellek elemanı elde edilebilir. Ancak JK için benzer işlem yapılamaz çünkü $J = K = 1$ girişi için saatin 1 olduğu sürece ana bellek elemanı çevirime girer. O nedenle ana-bağımlı JK bellek elemanı için çözüm olacak bir devre Şekil 6.16 te verilmiştir. Bu devrenin saat darbesinin

düşen kenarında tetiklenen bir JK bellek elemanı olarak çalıştığını S^*R^* tanımından yararlanarak gösteriniz.

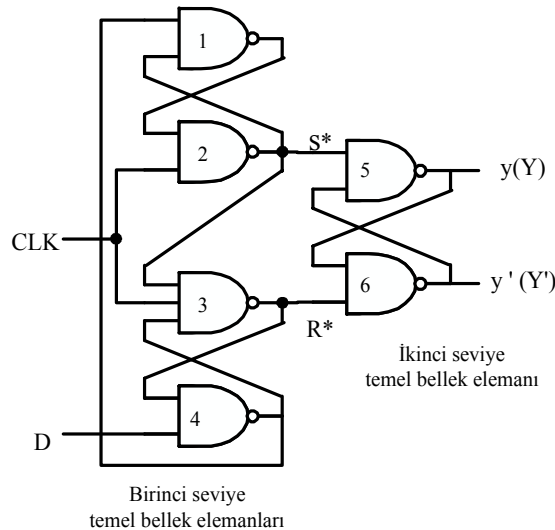


Şekil 6.16 Saat darbesinin düşen kenarında tetiklenen JK bellek elemanı

Kenar tetiklemeli bellek elemanları:

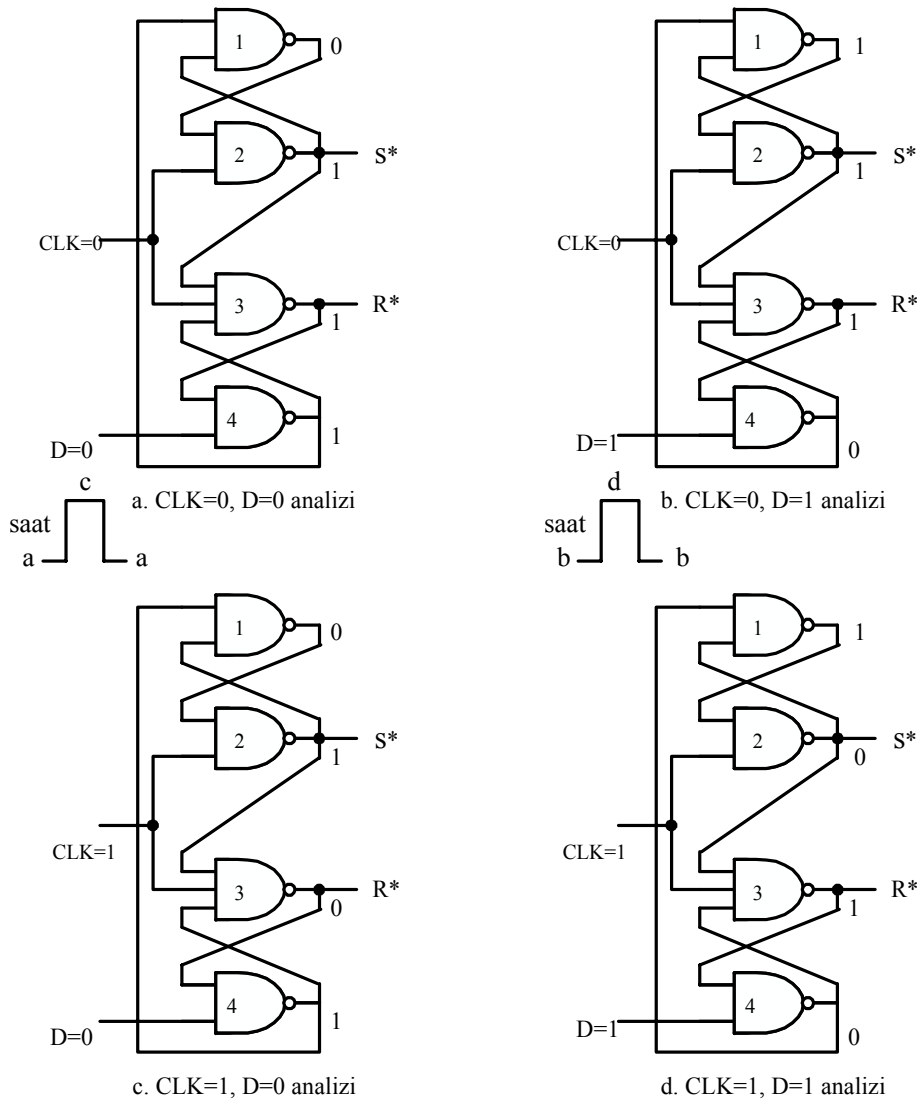
Bellek elemanlarının saat darbelerinin geçişlerinde çalışmasını sağlayan üçüncü alternatif, kenar tetiklemeli bellek elemanlarıdır. Saat darbesinin seviyesi belli bir eşik değerini aşarken bellek elemanı çalışır ; ondan sonra aynı eşik değere aynı yönden (yükselen veya düşen) ikinci defa gelene kadar bellek elemanları, girişlere karşılık vermezler, çalışmazlar.

Darbe kenarı ile tetiklenen **gecikme bellek elemanı**na ilişkin lojik devre Şekil 6.17 te verilmiştir. Bu devrede temel bellek elemanı (ikinci seviye) girişlerine birer temel bellek elemanı (birinci seviye) bağlanmıştır. Birinci seviyedeki temel bellek elemanları, darbenin düşen kenarında, ikinci seviyedeki temel belleğin durumunu koruyucu girişleri (11) sağlar. Aynı bellek elemanları darbenin yükselen kenarında ise; ikinci seviyedeki temel belleğin girişinin, çıkışına aktarılmasını sağlar.



Şekil 6.17 Kenar tetiklemeli gecikme bellek elemanın lojik devresi

Şimdi devrenin analizini yapalım. Saat darbesi 0 iken $D = 0$ ve $D = 1$ girişleri için birinci seviyedeki temel bellek elemanlarının çıkışları şekil 6.18 a ve b de verilmiştir. ikinci seviyedeki (çıkıştaki) temel bellek elemanı gösterilmemiştir.



Şekil 6.18 Darbe kenarı tetiklemeli gecikme bellek elemanı lojik devresinin analizi

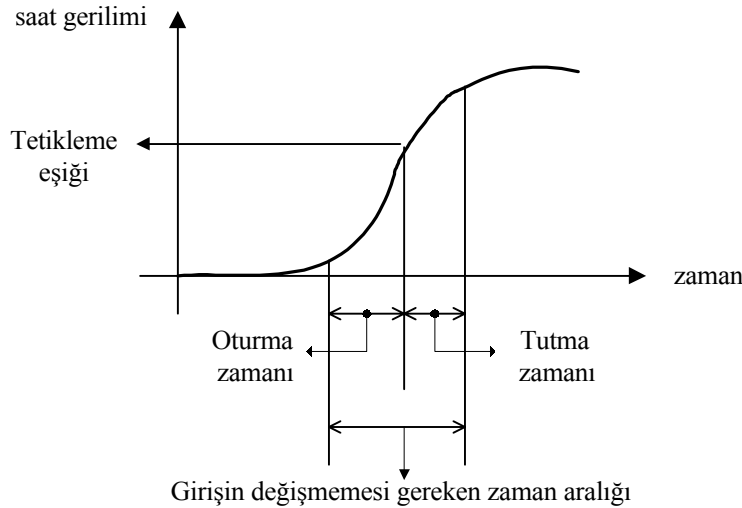
Saat = 0 iken D girişi ne olursa olsun $S^* = R^* = 1$ dir. Buda çıkıştaki ikinci seviye temel bellek elemanın durumunu değiştirmemesi yani çalışmaması demektir . 4 kapısı D ' yi gösterirken, 1 kapısı D yi gösterir (Şekil 6.18 a ve b).

Şekil 6.18 a da saat darbesi 0 dan 1 e çıkarıldığında, Şekil 6.18 c görülen değerler gözlenir. Üç kapısının çıkışı 0 olacak, diğer kapı çıkışları ise eski değerlerini koruyacaktır. $S^*R^* = 10$ girişi için çıkıştaki temel belek elemanı çıkışı, $R^* = 0$ olduğundan, D = 0 olan girişe eşit olacaktır. Benzer biçimde Şekil 6.18 b deki devrenin 0 olan saat girişi, 1 kılındığında; Şekil 6.18 d deki değerler gözlenir. Bu sefer 2 kapısının çıkışı 0 olacak, ama diğer kapıların çıkışları değişmeyecektir. $S^*R^* = 01$ girişi için çıkıştaki temel belek elemanı çıkışı, $R^* = 1$ olduğundan, D=1 olan girişe eşit olacaktır. Sonuç olarak, Şekil 6.17 te verilen kenar tetiklemeli bellek elemanı, saat darbesinin yükselen kenarında, D bellek elemanın tanımına uygun çıkışlar verir. Yani saat darbesinin yükselen kenarında giriş çıkışa aktarılmaktadır.

Şekil 6.18 c de saat darbesi 1 den 0 a düştüğünde; Şekil 6.18 a daki değerler gözlenir. 3 kapısının çıkışı 1 olacak, diğer kapı çıkışları ise eski değerlerini koruyacaktır. $S^*R^* = 11$ olduğundan, çıkıştaki temel bellek elemanı . Benzer biçimde Şekil 6.18 d de saat

darbesi 1 den 0 a düştüğünde; Şekil 6.18 b daki değerler gözlenir. 2 kapısının çıkışı 1 olacak, diğer kapı çıkışları ise eski değerlerini koruyacaktır. $S^*R^* = 11$ olduğundan, çıkıştaki temel bellek elemanı çalışmayacak yani durumu değişmeyecektir. Sonuç olarak, Şekil 6.17 te verilen kenar tetiklemeli bellek elemanı, saat darbesinin düşen kenarında çalışmayacaktır. Yani şekil 6.17 de verilen devre, saat darbesinin yükselen kenarda tetiklenen bir D bellek elemanıdır.

Şimdi de saat darbesinin yükselme süresinde, çalışma koşullarını daha detaylı olarak analiz edelim. Şekil 6.18 a daki devrede $D = 0$ iken saat darbesini 1'e çıkarırsak; 4 ve 1 kapılarının çıkışları değişmezken, $S^* = 1$ ve $R^* = 0$ olur (Şekil 6.17 c). Buda Şekil 6.17 teki devrede $Y=0$ olması, yani çıkışın D girişine eşit olması ve saat darbesinin yükselen kenarında tetiklenmesi demektir. Saat darbesi 0 dan 1 e yükselirken belli bir süre önce, ki ona **oturma zamanı (set up time)** denir, D girişinin değişmemesi gerekir. Oturma zamanı 4 ve 1 kapılarındaki toplam gecikme kadardır. Bu süre $D = 0$ girişi için 1 kapı çıkışının 0 da kalma süresidir ve ancak o zaman saat 0 dan 1 e geçerken $D = 0$ girişi algılanabilir. Bir başka deyişle D bellek elemanı tarafından $D = 0$ girişinin algılanması isteniyorsa; tetikleme eşiğinden önce, en az oturma süresi kadar bir zaman için, $D = 0$ girişinin girişte tutulması gerekir. Oturma süresi içinde $D = 1$ yapılırsa, bu algılama gerçekleşmez (Bakınız Şekil 6.19).



Şekil 6.19 Yükselen kenarda tetikleyen darbenin değişimi ve giriş sınırlaması

D girişi, saat darbesi tetikleme eşiğini aştıktan sonra da, tutma süresi (holding time) denen zaman aralığında, değişmemelidir. Bu tutma süresi, 3 kapı çıkışı (R^*) nin 1 den 0 a geçme süresidir ve ancak o zaman saat 0 dan 1 e geçerken $D = 0$ girişi algılanabilir. Bir başka deyişle $D = 0$ girişinin D bellek elemanı tarafından algılanması isteniyorsa ; $D = 0$ girişinin, tetikleme eşiğinden sonra en az tutma süresi kadar, girişte tutulması gerekir. Tutma süresi içinde $D = 1$ yapılırsa, bu algılama gerçekleşmez (Bakınız Şekil 6.19).

Sonuç olarak $D=0$ girişinin, D bellek elemanı tarafından algılanabilmesi için en az oturma +tutma zamanı kadar bir zaman aralığında, girişte tutulması zorunludur. (Bakınız Şekil 6.19)

Yukarıda $D=0$ girişi için saat darbesinin yükselen kenarındaki durumu inceledik, benzer biçimde, $D=1$ girişinde de zamanlama koşulları aynıdır. Yani oturma ve tutma süreleri toplam süresince $D=1$ değeri değişmemelidir.

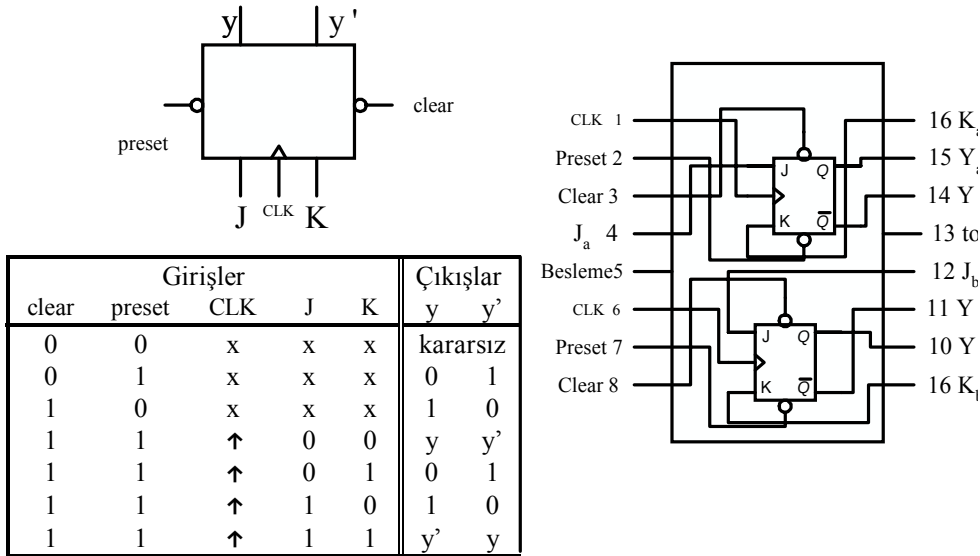
Burada yalnızca D belek elemanın kenar tetiklemeli lojik devresi verilmiş ve analiz edilmiştir. Diğer bellek elemanları için de benzer biçimde çalışan devreleri vardır ve genellikle kataloglarda verilir.

6.6 Tüm devre olarak bellek elemanları yardımcı girişleri

Bir ardışıl devrenin başlangıç durumunun (D^0) istenildiği gibi seçilebilmesi gerekir. Buda her bir bellek elemanına karşı düşen, bağımsız durum değişkeni (y) nin, başlangıçta istenildiği gibi seçilebilmesi demektir. Bir başka deyişle bellek elemanlarının başlangıç durumunu, 0 veya 1 den herhangi birine getirebilmemiz gerekir. O nedenle başlangıç durumunu 0 a getiren bir **Clear** ile, başlangıç durumunu 1 e getiren bir **preset** yardımcı girişlerine gereksinim vardır. Bu girişler asenkron olarak çalışırlar, yani saat ve giriş değişkenlerinden bağımsız olarak başlangıç durumunu 0 veya 1 e getirirler.

Bellek elemanları, iki tanesini bir arada bulunduran, küçük ölçekli (SSI) tümleşik devreler olarak satılırlar. Yardımcı girişleri de içeren bir JK bellek elemanı tümleşik devresinin şematik gösterilimi ve tanım tablosu Şekil 6.20 da verilmiştir. Clear ve Preset girişlerinde bir yuvarlak varsa; bu girişlerin 0 olması halinde, clear ve preset işlevleri yerine getirilir. Clear ve Preset girişlerinde bir yuvarlak yoksa; bu girişlerin 1 olması halinde, clear ve preset işlevinin yerine getirilir. Saat darbesinde aşağıya doğru bir ok düşen kenarda tetiklenen bir bellek elemanını, yukarıya doğru bir ok ise yükselen kenarda tetiklenen bir bellek elemanını gösterir.

Bir 7476 JK tümleştirilmiş devresindeki bağlantılar ve uçların ne oldukları Şekil 6.20 da verilmiştir. Normal olarak bu tüm devreler 5 veya 10 voltluk (V_{CC}) doğru gerilimle beslenirler. Clear ve preset girişli olan bellek elemanları olduğu gibi yalnızca clear veya yalnızca preset girişleri olan bellek elemanları da piyasada bulunabilir.



Şekil 6.20 Belek elemanlarının çeşitli girişleri ve bir JK tümleştirilmiş devresi

BÖLÜM 7 ARDIŞIL DEVRELERİN ANALİZİ

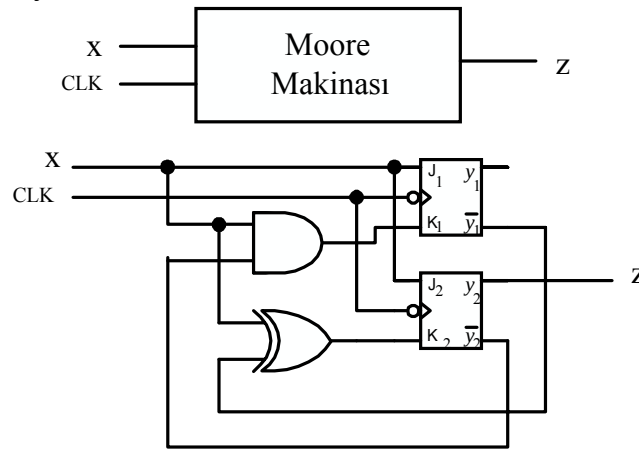
Bundan önceki bölümde ardışıl devrelerin senkron(saatli) ve asenkron olarak sınıflandırıldığını gördük, bu ve bundan sonraki bölümlerde yalnızca saatli (senkron) ardışıl devreler üzerinde duracağız.

7.1 Saatli ardışıl devrelerin analizi

Bilindiği gibi analiz, elemanlar ve bağlantılar belli iken, girişlere karşı düşen çıkışların ne olduğunu bulmaktır. Bir başka deyişle devrenin girişlere cevabını belirlemektir. Üç analiz yöntemini inceleyeceğiz, öyle ki her biri daha önce verdiğimiz bir modele ilişkin olacak. Birinci yöntem, matematiksel model olarak tanıtılan, **bir sonraki durum ve çıkış fonksiyonlarının** bağımsız değişkenler (giriş ve şimdiki durum değişkenleri) cinsinden bulunmasıdır. İki ve üçüncü yöntemler, daha önce tanıtılan **durum tablosu** ve **durum diyagramına** dayandırılan yöntemlerdir. Bu üç yöntemden birinden diğerine kolayca geçilebilir. Üç farklı yöntemin kullanılması, analizin amacına bağlı olarak birinin diğerinden daha çabuk sonuç verebilmesidir. Örneğin bir sonraki durum ve çıkış fonksiyonları ile analiz yöntemi, bir lojik devrenin bilgisayar simülasyonu açısından, diğerlerine göre daha uygundur. Diğer taraftan belli bir giriş dizisi için çıkış dizisinin ne olduğu isteniyorsa, durum diyagramı veya durum tablosu yöntemleri daha kolay sonuca götüreceğinden tercih edilir. Bu yöntemleri **Moore** ve **Mealy** makineleri üzerinde birer örnekle inceleyelim.

7.2 Örnek bir Moore makinesi analizi

Daha önce tanımladığı gibi Moore makinesinde çıkışlar yalnızca durum değişkenlerinin fonksiyonudur. Tek giriş ve tek çıkışlı bir Moore makinesi örneği Şekil 7.1 de verilmiştir. Görüldüğü gibi çıkış yalnızca bir durum değişkeninin fonksiyonudur, giriş değişkenine doğrudan bağlı değildir, yani Moore makinesidir.



Şekil 7.1 Analizi için örnek olarak seçilmiş bir Moore makinesi

1.1.1 Matematiksel model ve analiz

Şekilden görüldüğü gibi $Z = y_2$ dir. Birinci JK bellek elemanına ilişkin bir sonraki durum fonksiyonunu, JK nın tanım bağıntısından yararlanarak bulabiliriz. Birinci bellek elemanın tanım bağıntısı

$$Y_1 = y_1 K'_1 + y'_1 J_1 \text{ dir.}$$

Diğer taraftan birinci bellek elemanına ilişkin girişler

$$J_1 = x$$

$$K_1 = x \oplus y'_2 \text{ dir.}$$

Bunları birinci bellek elemanının yukarıdaki tanım bağıntısında yerine korsak; birinci bellek elemanına ilişkin bir sonraki durum fonksiyonunu, bağımsız giriş ve durum değişkenleri cinsinden ifade etmiş oluruz.

$$Y_1 = y_1 (x \oplus y'_2)' + y'_1 (x)$$

$$= y_1 (x' + y_2) + y'_1 x$$

$$Y_1(x, y_1, y_2) = y_1 x' + y_1 y_2 + y'_1 x$$

Benzer biçimde ikinci bellek elemanına ilişkin bir sonraki durum fonksiyonunu da

$$Y_2 = y_2 K'_2 + y'_2 J_2 \text{ dir.}$$

$$J_2 = x$$

$$K_2 = x \oplus y'_1 = x' y'_1 + x y_1$$

$$Y_2 = y_2 (x \oplus y'_1)' + y'_2 x$$

$$Y_2(x, y_1, y_2) = x' y_1 y_2 + x y'_1 y_2 + x y'_2 \text{ biçiminde buluruz.}$$

Görüldüğü gibi iki bellek elemanı olduğu için iki bir sonraki durum fonksiyonu vardır. Olabilecek durumlar: **00** ($Y_1 = 0, Y_2 = 0$), **01** ($Y_1 = 0, Y_2 = 1$), **10** ($Y_1 = 1, Y_2 = 0$), **11** ($Y_1 = 1, Y_2 = 1$) olmak üzere dört tanedir.

Örneğimizdeki Moore makinesinin matematiksel modeli $M = \{ G = \{0,1\}, Ç = \{0,1\}, D = \{00,01,10,11\}, Y = \{Y_1 = y_1 x' + y_1 y_2 + y'_1 x, Y_2 = x' y_1 y_2 + x y'_1 y_2 + x y'_2\}, Z = \{z = y_2\}, D^0 = \{\text{bağımsız seçilebilen } 00,01,10,11 \text{ den biridir}\} \}$ dir. Bu matematiksel modelden yararlanarak $D^0 = 00$ başlangıç durumunda $x = 101$ giriş dizisi için devrenin analizini yapalım.

1. saat darbesi düşen kenarından önce: $x = 1$, şimdiki durum (D^0) = 00 ($y_1 = 0, y_2 = 0$), $z = 0$ dir.

1.saat darbesinin düşen kenarından sonra: $x = 1$, bir sonraki durum = 11, $Y_1(x = 1, y_1 = 0, y_2 = 0) = 1, Y_2(x = 1, y_1 = 0, y_2 = 0) = 1, z = 1$ dir.

2. saat darbesinin düşen kenarından önce $x = 0$, şimdiki durum = 11 (1. saat darbesi için bir sonraki durum), $z = 1$ dir.

2.saat darbesinin düşen kenarından sonra: $x = 0$, bir sonraki durum = 11, $Y_1(x = 0, y_1 = 1, y_2 = 1) = 1, Y_2(x = 0, y_1 = 1, y_2 = 1) = 1, z = 1$ dir.

3. saat darbesinin düşen kenarından önce $x = 1$, şimdiki durum = 11 (2. saat darbesi için bir sonraki durum), $z = 1$ dir.

3.saat darbesinin düşen kenarından sonra: $x = 1$, bir sonraki durum = 10, $Y_1(x = 1, y_1 = 1, y_2 = 1) = 1, Y_2(x = 1, y_1 = 1, y_2 = 1) = 0, z = 0$ dir.

Buna göre $x = 101$ dizisine karşılık, çıkışta 110 dizisi gözlenecektir.

7.2.2 Durum tablosu ve analiz

Durum tablosu satırlarında şimdiki durumlar, sütunlarında bütün olabilecek girişler, elemanlarında da bir sonraki durum ve çıkışların bulunduğu bir matristir. Satır ve sütunlardaki durum ve girişleri, Karnaugh diyagramı sırasında almak kolaylıklar sağlar. Buna göre Moore makinesi örneğimizdeki durum tablosunu, daha önce bulduğumuz, bir sonraki durum ve çıkış fonksiyonlarını tabloya taşıyarak elde edebiliriz. Durum değişkeni sayısı iki olduğundan, olabilecek bütün durumlar dört tanedir yani dört satır vardır. Giriş değişkeni de bir tane olduğundan, iki farklı giriş yani iki sütun vardır. Önce bütün durumları Karnaugh diyagramındaki sıralanışa uygun olarak satırlara, bütün girişleri de benzer biçimde sütunlara Şekil 7.2 deki gibi yerleştirelim. Tabloyu Karnaugh diyagramı biçimde oluşturduğumuzdan, taşıyacağımız fonksiyonların her bir çarpımlar teriminin kaçınıcı mertebeden bir küp oluşturduğuna bakıp, gerekli olan doğru minterimleri uygun yerlere koyarak durum tablosunu bulabiliriz. Bu amaçla, daha kolay görülsün diye Y_2Y_1 fonksiyonlarını önce ayrı ayrı düşünüp, sonra birleştirebiliriz (Şekil 7.2). Örnek olarak Y_1 fonksiyonunu tabloya taşıyalım. Y_1 fonksiyonundaki birinci terim y_1x' , y_2 değişkenini eleyen birinci mertebeden bir alt küp oluşturur ve üçüncü ile dördüncü satırlarla birinci sütunun kesiştikleri yerlerde birer doğru mintermi içerir. Diğer çarpımlar terimleri için aynı işlem tekrarlanarak Y_1 ve Y_2 ayrı ayrı bulunabilir. Doğru olmayan minterimler 0 alınarak; bulduğumuz Y_1 ve Y_2 fonksiyonları, yan yana durum tablosuna şekilde görüldüğü gibi taşınırlar.

Çıkış fonksiyonu direkt olarak y_2 ye eşit olduğundan hemen yazılabilir. Moore makinesinin çıkışları yalnızca durum değişkenlerinin fonksiyonu olduğundan, durum tablosunda belli bir satıra karşı gelen satırların hepsinde çıkışlar aynıdır. Nedeni belli satır için durum değişkenleri değişmez dolayısıyla Moore makinesinin çıkışı değişmez. Bu belli satıra karşı gelen farklı sütunlar için giriş değişkenleri değişse bile Moore makinesi çıkışına etki etmezler. Bu nedenle Moore makinesinin çıkışları Şekil 7.2 deki gibi ayrı bir sütunda gösterilirler.

x y_1y_2	0	1
00	0	1
01	0	1
11	1	1
10	1	0
	Y_1	

x y_1y_2	0	1
00	0	1
01	0	1
11	1	0
10	0	1
	Y_2	

x y_1y_2	0	1	z
00	00	11	0
01	00	11	1
11	11	10	1
10	10	01	0
	Y_1Y_2, z		

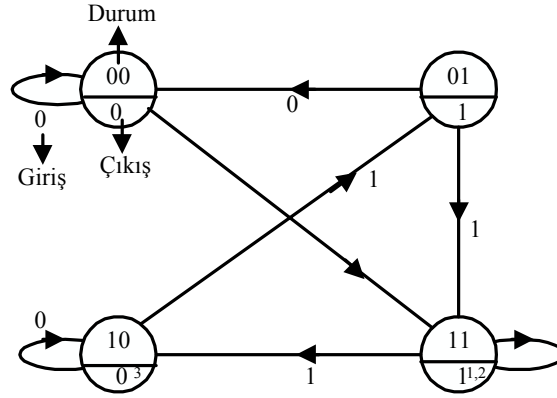
Şekil 7.2 Örnek Moore makinesine ilişkin durum tablosunun çıkarılışı

Şimdi $x = 101$ giriş dizisine karşılık çıkış dizisini, 00 başlangıç durumunda, durum diyagramından bulalım. Durum tablosunun 1. satırından görüldüğü gibi 00 durumunda

çıkış, girişten bağımsız, 0 dir. Şimdiki 00 durumunda giriş 1 iken, 1. saat darbesinin düşen kenarında bir sonraki durum, tablonun 1. satır ve 2. sütunundan görüldüğü gibi, 11 dir. Bu durumda, tablodaki 3. satır , **çıkış 1 dir**. 11 durumunda giriş 0 iken, 2.saat darbesinin düşen kenarında, bir sonraki durum ve çıkış değişmez. Bir sonraki durum 11 (Tablodaki 3. satır 1. sütun.), **çıkış ise 1 dir**. (Tablodaki 3. satır.) Şimdiki 11 durumunda giriş 1 iken, 3. saat darbesinin düşen kenarında, bir sonraki durum 10 olur. Tablodaki 3. satır 2. sütun. **Çıkış ise 0 dir**. Tablodaki 4. satır . O halde çıkış dizisi 110 dir.

7.2.3 Durum diyagramı ve analiz

Durum tablosu bulunduktan sonra, durum diyagramına geçiş kolaydır. Önce her bir durum için, birer çember biçiminde daire çizilir. Sonra her bir durum için; bütün girişler altında gidilen yeni durum , yönlendirilmiş topolojik elemanlarla belirtilir. Eksiklik olmaması için; her bir durumdan, giriş sayısı kadar okun çıkması kontrol edilebilir. Birden fazla giriş için bir sonraki durum aynıysa, bu girişlere ilişkin oklar bazen birleştirilerek te gösterilir. Moore makinesinde çıkışlar yalnız durum değişkenlerine bağlı olduğu için çıkışlar durumlara ilişkin çember düğümün içine yazılır. incelediğimiz örneğe ilişkin durum diyagramı Şekil 7.3 de verilmiştir.



Şekil 7.3 Örnek Moore makinesine ilişkin durum diyagramı

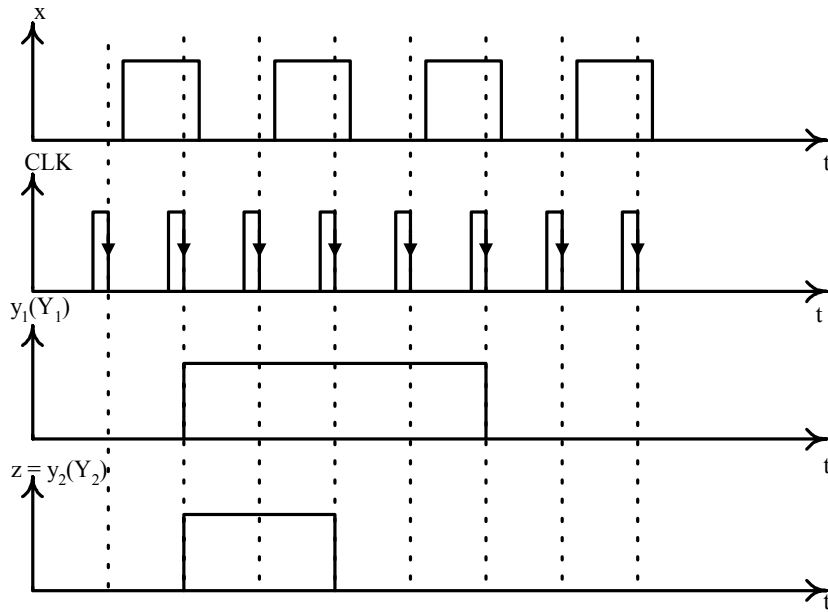
Şimdi 00 başlangıç durumu ve $x = 101$ giriş dizisi için örnek Moore makinesinin analizini yapalım. Şekil 7.3 teki durum diyagramında gözlenecek çıkışların üzerine kaçınca saat darbesinden sonra gözleneceği yazılmıştır. Yani çıkış dizisi $z = 110$ dir.

7.2.4 Zaman diyagramı (Timing diagram)

Analog devrelerde girişler ve çıkışlar zamana bağlı birer fonksiyondur. Sayısal sistemlerde ise giriş ve çıkışlar zamana bağlı olarak birer dizi oluşturmaktadırlar. Lojik devrelerde, giriş dizisi ve başlangıç durumu verildiğinde, bir sonraki durum ve çıkış fonksiyonların zamana göre değişimini gösteren diyagramlara **zaman diyagramı (timing diagram)** denir. İncelemekte olduğumuz Moore makinesi için Başlangıç durumu 00, giriş dizisi de 010101 olsun. Giriş dizisinin ilk elemanının 0 olduğunu, yani dizinin zamana göre soldan sağa sıralandığını varsayalım. Bellek elemanları saat darbesinin negatif kenarında tetiklendiği için durum değişiklikleri yalnızca bu düşen kenarda olabilecektir. incelediğimiz makine Moore makinesi olduğundan, yani çıkışlar yalnızca durum değişkenlerinin fonksiyonu olduğundan, çıkışlar da yalnızca saat darbelerinin düşen kenarında

değişecektir. Mealy makinesinde bundan sonraki bölümde göreceğimiz gibi böyle olmamaktadır.

Önce şimdiki durum ve giriş altında, bir sonraki durum ve çıkış, durum diyagramı veya tablosundan yararlanılarak bulunur. Bulunan değerler, ard arda iki saat darbesinin düşen kenarları arasındaki zaman aralığına, yerleştirilir. Bu işlem her saat darbesi için tekrarlanarak zaman diyagramı çizilir. (i). saat darbesi için bir sonraki durum (i+1). saat darbesi için şimdiki durumdur. Örneğimizde birinci saat darbesinin düşen kenarından hemen önceki durum (başlangıç durumu) $y_1y_2 = 00$, giriş ise 0 dır. Bir sonraki durumun $Y_1Y_2 = 00$ olduğu, Durum tablosundan, görülmektedir. O nedenle. Şekil 7.4 teki zaman diyagramında, birinci ve ikinci saat darbelerinin düşen kenarları arasına Y_1 ve Y_2 değişkenleri 0 genlikli olarak çizilmişlerdir. Yeni durum $Y_1Y_2 = 00$, ikinci saat darbesi için şimdiki durum, yani $y_1y_2 = 00$ dır. ikinci saat darbesi için giriş 1 dir. Birinci saat darbesi için yapılan işlemler ikinci ve daha sonraki saat darbeleri için tekrarlanarak zaman diyagramı Şekil 7.4 teki gibi tamamlanır. Görüldüğü gibi giriş değişkeninin aldığı değer, iki saat darbesinin düşen kenarları arasında değişse bile; bir sonraki durum değişkenlerine ve çıkışa etkimez. Çünkü bellek elemanları kenar tetiklemelidir ve makine Moore makinesidir.



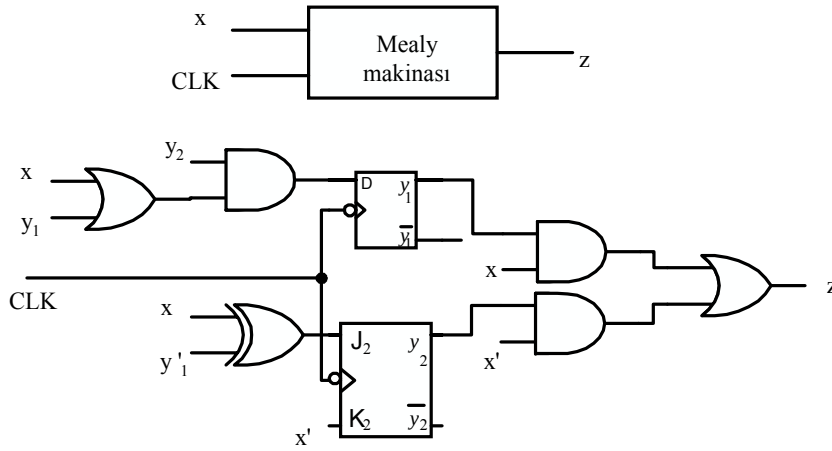
Şekil 7.4 Örnek Moore makinesine ilişkin bir zaman diyagramı

Zaman diyagramının çiziminde, durum tablosu veya diyagramından yararlandık. Matematiksel modeli (Bir sonraki durum fonksiyonları ve çıkış fonksiyonlarını) kullanarak ta, aynı sonuca varabilirdik ama daha zor olurdu.

7.3 Örnek bir Mealy makinesi analizi

Mealy makinesinin analizi de Moore Makinesi analizi gibidir. Mealy makinesinde çıkış fonksiyonları, yalnızca durum değişkenleri değil fakat giriş değişkenlerine de bağlıdır. Bir kombinezonsal devrenin çıkışı olan bu fonksiyonlar, saat darbeleriyle değişen durum değişkenlerinden etkilendiği gibi, saat darbelerinin dışında değişen giriş değişkenlerinden de, etkilenirler. O nedenle çıkışta hatalı çıkış diye adlandırılan istenmeyen çıkışlar

görülebilir. Şimdi Şekil 7.5 de verilen bir Mealy makinesi örneğinde bir sonraki durum fonksiyonları ve çıkış fonksiyonunun, zaman diyagramlarını inceleyerek hatalı çıkışları görelim.



Şekil 7.5 Analizi için örnek olarak seçilmiş bir Mealy makinesi

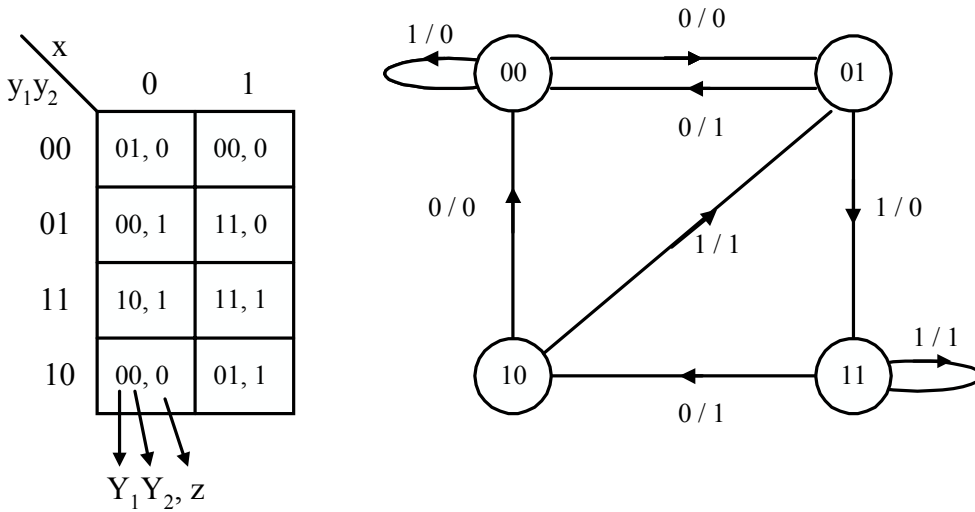
Bir sonraki durum fonksiyonları ve çıkış fonksiyonu

$$Y_1 = D = (x + y_1) \cdot y_2$$

$$Y_2 = y_2 K'_2 + y'_2 J_2 = y_2 (x')' + y'_2 (x \oplus y_1') = y_2 x + y'_2 x y_1 + y'_2 x' y'_1$$

$$= x y_2 + x y_1 + x' y_1 y_2$$

$$Z = x' y_2 + x y_1 \text{ dir.}$$



Şekil 7.6 Örnek Mealy makinesinin analizi için durum tablo ve diyagramı

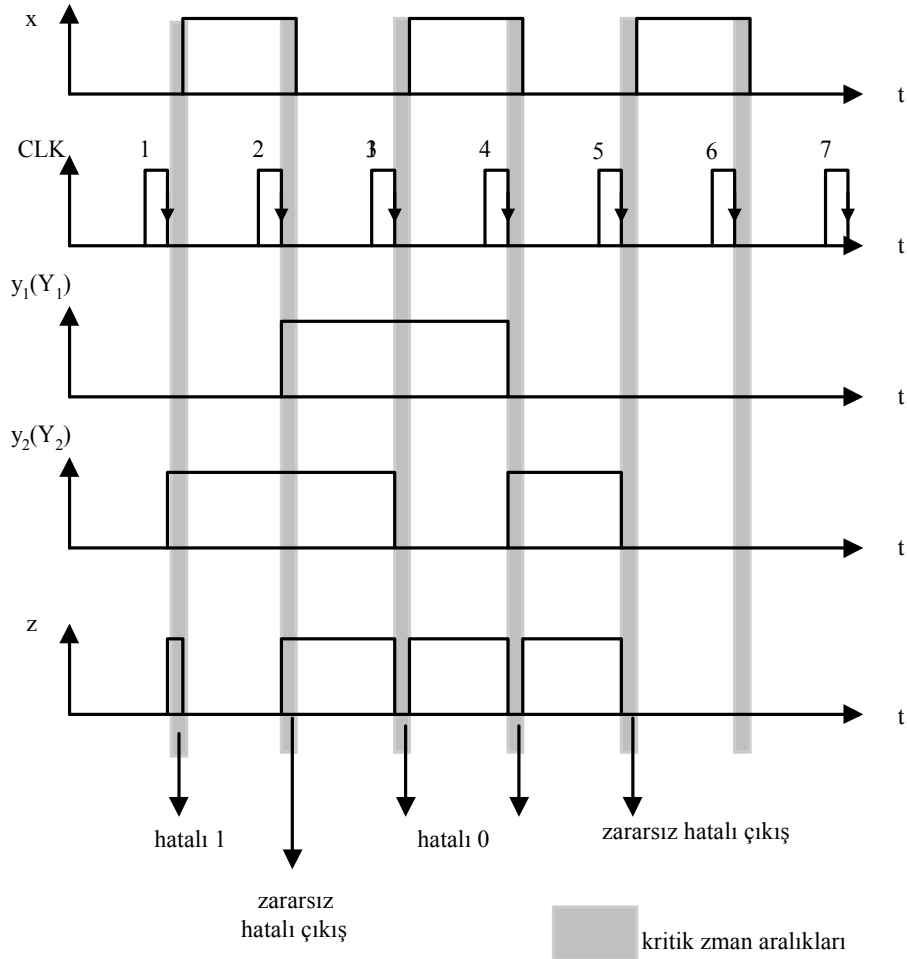
Şekil 7.6 da örnek Mealy makinesinin durum tablo ve diyagramı, Moore makinesinde anlatılan biçimiyle bulunarak, verilmiştir.

Şimdi durum tablosu ve diyagramından yararlanarak zaman diyagramlarını çıkaralım. Bir sonraki durum fonksiyonlarına ilişkin zaman diyagramları, saat darbesinin düşen kenarında değişecektir ve Moore makinesinde olduğu gibi çizilebilir.

Çıkış fonksiyonu, durum değişkeninin değiştiği saat darbelerinin düşen kenarları dışında, giriş değişkeninin değişmesiyle de değişebilir. O nedenle **çıkış fonksiyonuna**

ilişkin zaman diyagramını çizerken, Şekil 7.6 da görüldüğü gibi, saat darbesinin düşen kenarı (durumların değişebileceği an) ve giriş değişkeninin değiştiği anları ayrı ayrı incelemek gerekmektedir. **Çıkış fonksiyonları** için saat darbesinin düşen kenarı ile girişin değiştiği zaman aralığı, kritik bir zaman aralığıdır. Çünkü bu kritik zaman aralığında yeni gelen durumda, birinci saat darbesine ilişkin giriş, etkisini sürdürerek çıkışı değiştirebilir. Bu istenmeyen bir durumdur, çünkü her bir giriş için 1 saat darbesi uygulanmalı ve yeni gidilen durumda (saat darbesinin düşen kenarından hemen sonra) yeni giriş ideal olarak hemen uygulanmalıdır. Bu sağlanamadığından, sanki yeni gidilen durumda eski giriş uygulanmış gibi olur.

Örnek Mealy makinesinin durum ve çıkış (kritik zaman aralığı da göz önüne alınarak) değişkenlerine ilişkin zaman diyagramları Şekil 7.7 da verilmiştir. Önce durum değişkenlerine ilişkin zaman diyagramları Moore makinesinde olduğu gibi kolayca çizilebilir.



Şekil 7.7 Örnek Mealy makinesine ilişkin bir zaman diyagramı ve hatalı çıkışlar

Şimdi çıkış fonksiyonuna ilişkin zaman diyagramını adım adım detaylı olarak inceleyelim. 00 başlangıç durumu ve 0 girişi için, durum diyagramı veya tablosundan

görüldüğü gibi, çıkış 0 dır. Kritik zaman aralığında, yeni durum 01 ve eski giriş 0 halen devam ettiği için çıkış 1 dir. Kritik zaman aralığından sonra 01 durumu devam ederken giriş değişip 1 olduğundan çıkış 0 olmuştur. 00 durumu ve 0 girişi altında 0 olan çıkış, saat darbesinden sonra 01 durumu ve 1 girişi altında 0 değerindedir ama kritik zaman aralığında kısa bir süre 1 değerini almıştır. Bu istenmeyen hatalı bir çıkıştır çünkü eski ve yeni durumlarda 0 çıkışını vermesi gereken makine kısa bir süre içinde olsa 1 çıkışı vermiştir. Kritik zaman aralığında çıkışın aldığı değer, bu zaman aralığının dışında alınan değerlerden birini alsaydı farkına varılmayacak, yani zararsız bir hatalı çıkış olacaktı. Bu olayın nedeni, girişin saat darbesinin düşen kenarından sonra değişmesidir. Benzer akıl yürütme aynen diğer saat darbeleri içinde tekrarlanarak şekil 7.6 daki zaman diyagramı bulunur. Görüldüğü gibi 2, 5, 6. saat darbelerinden sonraki kritik zaman aralıklarındaki hatalı çıkışlar zararsızdır. 1. saat darbesinden sonraki kritik zaman aralığına ilişkin hatalı çıkış 1 iken, 3. ve 4. saat darbelerinden sonraki hatalı çıkışlar 0 dır.

Genel olarak doğru çıkışlar saat darbelerinden hemen önceki çıkışlardır. Bu nedenle 0101010 giriş dizisine ilişkin çıkış dizisi 0011100 dır. Z çıkışında yalnızca doğru çıkışların gözlenmesi isteniyorsa, Z çıkışın bir D bellek elemanın girişine bağlanır ve D bellek elemanın saati de devrenin saatinden beslenir. Bu durumda D bellek elemanının çıkışında yalnızca doğru çıkışlar gözlenir, nedenini düşününüz.

Hatalı çıkışlar ancak Mealy makinesinde gözlenebilir.

BÖLÜM 8 ARDIŞIL DEVRELERİN SENTEZİ

Sentez bilindiği gibi girişleri ve çıkışları sözel olarak tanımlanan bir makinenin yaptığı işi yapan devrenin, yani elemanlar ve aralarındaki bağlantıların, bulunmasıdır. Şekil 6.2 de bir ardışıl devrenin genel yapısının ne olduğu verilmişti. Bu genel yapıdan görüldüğü gibi, sentezde bellek elemanlarının sayısını saptayıp, kombinezonsal devreyi oluşturursak ardışıl devreyi gerçekleyebiliriz. Bu bölümde önce ardışıl devrelerin sentezinde tutulacak yola ilişkin adımları sonra bu adımların uygulamasını çeşitli örnekler üzerinde göreceğiz.

8.1 Sentez yönteminin genel adımları

Sentez yönteminin 6 genel adımı aşağıda verilmiştir.

1. İstenen makinenin **sözle tanımlamasından (word description)**, uygun durum seçimleri yaparak, durum tablosu veya diyagramı çıkarılır. Bu adım için tutulacak yolun bir yöntemi yoktur, sezgisel olarak yapılır. Seçilen durumların ne anlama geldikleri saptanmalıdır. Durumların seçimi bazen kolay bazen ise zor olabilir. Genel olarak durumlar, istenen işlevi yerine getirebilmesi için, makinenin belleğinde tutması gereken bilgilerdir diye düşünülebilir. Durumlar seçildikten sonra sözle tanımlamaya uygun olarak durum tablosu ve diyagramı çizilir.

2. Birinci adımda durumları sezgisel olarak saptadığımızdan, gereğinden fazla durum seçmiş olabiliriz. Durum sayısını, durum değişkenlerinin sayısını belirler. Bu nedenle durum tablosunda eşdeğer durumlar varsa, bunların **indirgenmesi (state reduction)** yapılır. Durum sayısının azalması, durum değişkenlerinin ve dolayısıyla bellek elemanlarının sayısının ve kapı elemanlarının giriş yelpazesinin azalmasına yol açabilir. Farklı durumların sayısı μ ve farklı durum değişkenleri sayısı da r ise $r, r \geq \log_2 \mu$ olan en küçük tam sayıdır. Durum değişkeni sayısı aynı zamanda bellek elemanları sayısıdır. Durum değişkenlerinin sayısının minimumlaştırılması, kombinezonsal devreye ilişkin bağımsız değişkenlerin sayısının azalmasıdır. Durum indirgemesi sonucu durum değişkeni sayısı azalmasa bile kullanılmayacak durumlar ortaya çıkacaktır. Kullanılmayan durumlarda, gerçekleştireceğimiz fonksiyonların aldığı değerler keyfi olacaktır. Bu keyfi değerler de, dolaylı olarak, kombinezonsal devrenin basitleşmesini mümkün kılar. Durum indirgeme kuralları geliştirilmiştir ve ayrı bir başlık altında incelenecektir.

3. Seçilen **durumların kodlanması (state assignment)**: Durumlar birinci ve ikinci adımlarda sözle tanımlanmışlardı. "Durumlara hangi kodlar verilirse gerçekleştirilecek kombinezonsal devre daha basit olur problemine" bu adımda cevap verilir. Farklı yöntemler geliştirilmiştir, fakat birinin diğerlerine göre üstünlükleri olabildiğinden ideal genel bir yöntem verilememiştir. Bu notların kapsamı içinde basit bir iki yöntem ayrı bir bölümde incelenecektir. (r) durum değişkeniyle belirlenen bir durum, birbirinden farklı 2^r sayıda kodlanabilir

4. Kodlama sonucu bir sonraki durum ve çıkış fonksiyonları, bağımsız değişkenler olan giriş ve durum değişkenlerinin fonksiyonu olarak, durum tablosunda belirlenirler. Bu adıma kadar bellek elemanlarının sayısı ve her bir bellek elemanına ilişkin bir sonraki

durum değerinin, belli girişler ve şimdiki durum değişkenleri için ne olduğu belirlenmiş oldu. Şekil 6.2 deki genel yapıda görüldüğü gibi kombinezonsal devrenin çıkışlarının bir kısmı bellek elemanlarının giriş fonksiyonlarıdır. Bu fonksiyonlar saptandığında kombinezonsal devrenin bir kısmı gerçekleştirilebilecektir. Kombinezonsal devrenin diğer kısmı, çıkış fonksiyonlarıdır. Bellek elemanlarının giriş fonksiyonlarını saptayabilmek için seçilecek bellek elemanlarının ters tanım (excitation functions) bağıntılarından yararlanılır.

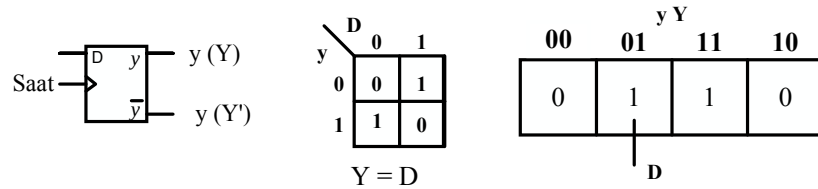
Bellek elemanlarının ters tanım bağıntıları: Bellek elemanlarının ters tanım bağıntıları; bellek girişlerinin, belleğin şimdiki ve bir sonraki durum değerleri belli iken, alması gereken değerleridir. Şimdi her bir bellek elemanına ilişkin ters tanım bağıntılarını, tanım bağıntılarından yararlanarak teker teker çıkaralım.

Gecikme bellek elemanın ters tanım bağıntısı: Tanım bağıntısında $y = 0$ iken $Y = 0$ olması için girişin zorunlu olarak 0 olması gerektiği hemen görülür. Benzer biçimde $y = 0$ iken $Y = 1$, $y = 1$ iken $Y = 0$ ve $y = 1$ iken $Y = 1$ olması için girişin sırasıyla 1, 0 ve 1 olması gerektiği görülür. Sonuç Şekil 8.1.a da gösterilmiştir. Ters tanım tablosunun sütunlarına önce y sonra Y değerleri yazılmış, bunlara ilişkin giriş değerleri ise bir altlarındaki karelerde gösterilmiştir.

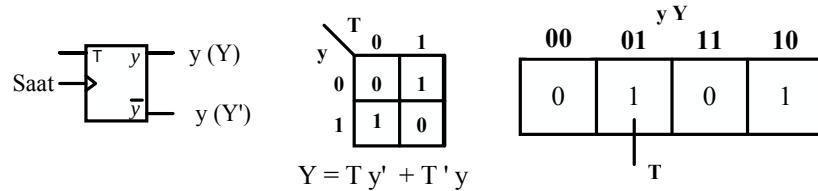
Tetikleme bellek elemanın ters tanım bağıntısı: Gecikme bellek elemanında yapılanlar burada da tekrarlanarak Şekil 8.1 b de görülen ters tanım bağıntısı bulunur.

SR bellek elemanın ters tanım bağıntısı: $y = 0$ dan $Y = 0$ a geçebilmek için, bu elemanın tanım bağıntısından görüldüğü gibi iki alternatif giriş vardır: $S = 0$ $R = 0$, $S = 0$ $R = 1$. Bu iki girişi $S = 0$, $R = K$ (keyfi,) biçiminde ifade edebiliriz. Çünkü durumun 0 dan 0 a geçişini sağlamak için $S = 0$ olması zorunludur, R ise 0 da 1 de olabilir yani K dir. Benzer biçimde diğer geçişlerde saptanarak Şekil 8.1 c deki ters tanım tablosu bulunur. Bu ters tanım tablosundan, girişlerin hiçbir zaman 11 olmadığı görülür, hatırlanacağı gibi SR bellek elemanının tanımını verirken 11 girişlerinin hiçbir zaman uygulanılmayacağı varsayılmıştı. Yani tanım ve ters tanım bağıntıları uyumludur.

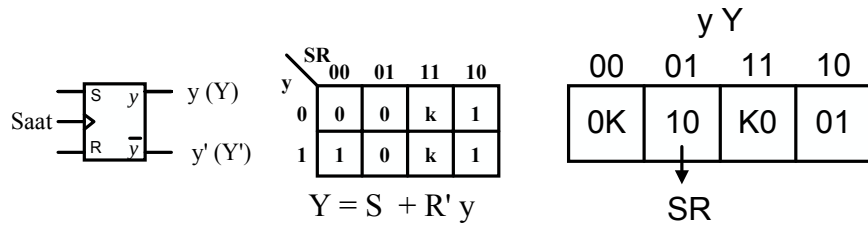
JK bellek elemanın ters tanım bağıntısı: Şekil.8.1 d de görüldüğü gibi, SR bellek elemanına benzer biçimde, JK bellek elemanının ters tanım bağıntısı da bulunabilir.



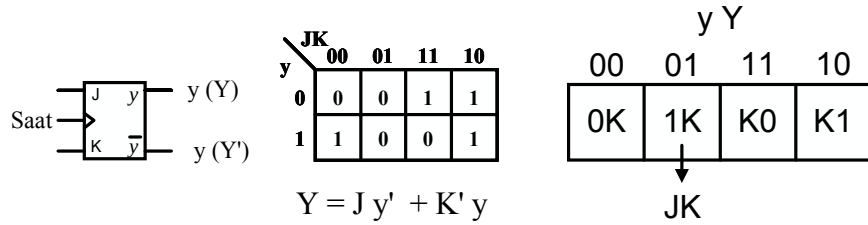
a. D gecikme elemanın ters tanım bağıntısı



b. T tetikleme elemanın ters tanım bağıntısı



c. SR bellek elemanının ters tanım bağıntısı



d. JK bellek elemanının ters tanım bağıntısı

Şekil 8.1 Bellek elemanlarına ilişkin ters tanım bağıntıları

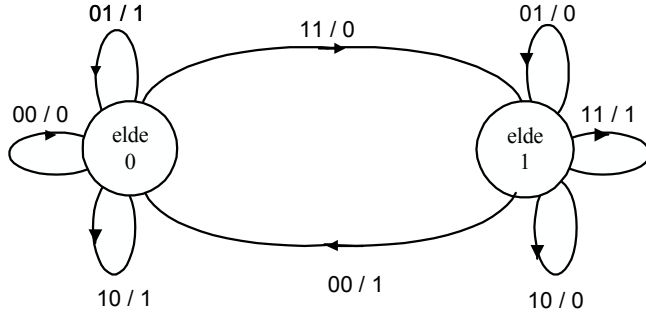
Üçüncü adımda bir sonraki durum fonksiyonlarının doğruluk tablosu, durum tablosunda oluşturulmuştu. Bellek elemanlarının giriş fonksiyonlarının doğruluk tablosu da, durum tablosundaki durum değişkeninin y ve Y değerleri ve bellek ters tanım bağıntısından yararlanarak, bulunur.

5. Çıkış fonksiyonları da, durum tablosundan, bağımsız giriş ve durum değişkenlerinin fonksiyonu olarak bulunur.

6. Bellek elemanlarının giriş fonksiyonları ve çıkış fonksiyonları minimalleştirilir. Bu minimal fonksiyonlara karşı gelen devre Ardışıl devrenin kombinezonsal kısmını oluşturur. Seçilen bellek elemanlarıyla bellek kısmı gerçekleştirilerek ardışıl devrenin sentezi tamamlanır.

Şimdi adım adım anlattığımız sentez işleminin, çeşitli örneklere uygulamasını görelim.

Seri toplayıcı: Şekil 6.1 de verilen seri toplayıcı için, durum tablosu ve diyagramı sırasıyla Şekil 6.2 b ve Şekil 6.3 b de çıkarılmıştı. Durumlar iki tane olup elde 0 ve 1 i göstermektedir, bir durum değişkeni yeterlidir ($\log_2 2=1$). Sentez işleminin ilk üç adımı daha önce yapıldığından dördüncü adımdan başlayacağız. Şekil 8.2 de, durum tablosu veya durum diyagramından yararlanarak, bir sonraki durum ve çıkış fonksiyonlarının doğruluk tablosu gösterilmiştir. Bellek elemanı olarak SR elemanını seçelim, bir durum değişkeni olduğuna göre bir bellek elemanı yeterlidir. Bu bellek elemanının ters tanım bağıntısından SR giriş fonksiyonlarının doğruluk tablosu Şekil 8.2 daki gibi bulunur. Şimdi artık S, R ve Z fonksiyonlarını, Karnaugh diyagramlarına taşıyıp, indirgeyerek, ardışıl devreyi Şekil 8.3 deki gibi gerçekleyebiliriz.



x_1	x_2	y	Y	Z	S	R
0	0	0	0	0	0	K
0	0	1	0	1	0	1
0	1	0	0	1	0	K
0	1	1	1	0	K	0
1	0	0	0	1	0	K
1	0	1	1	0	K	0
1	1	0	1	0	1	0
1	1	1	1	1	K	0

Şekil 8.2. Seri toplayıcı için, bellek girişlerine ve çıkışa ilişkin fonksiyonların doğruluk tablosu.

x_1x_2	00	01	11	10
0			1	
1		k	k=1	k

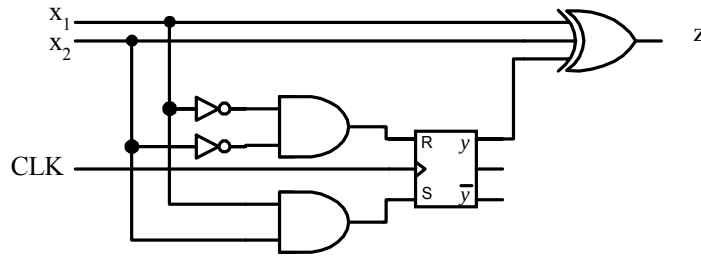
$S = x_1x_2$

x_1x_2	00	01	11	10
0	k=1	k		k
1	1			

$R = x'_1x'_2$

x_1x_2	00	01	11	10
0		1		1
1	1		1	

$z = y \oplus x_1 \oplus x_2$



Şekil 8.3 Bellek elemanı giriş fonksiyonları, çıkış fonksiyonları, indirgenmiş ifadeleri ve seri toplayıcının lojik devresi

Genel olarak Sayıcılar, girişine gelen darbelerle daha önce belirlenen durumlardan geçen devrelerdir. Giriş darbeleri saat darbeleri olabileceği gibi, ayrı bir giriş te olabilir. Sayıcının bir özel hali de saymayı iki tabanında yapanlardır.

8.2 Modulo 8 ileri sayıcı

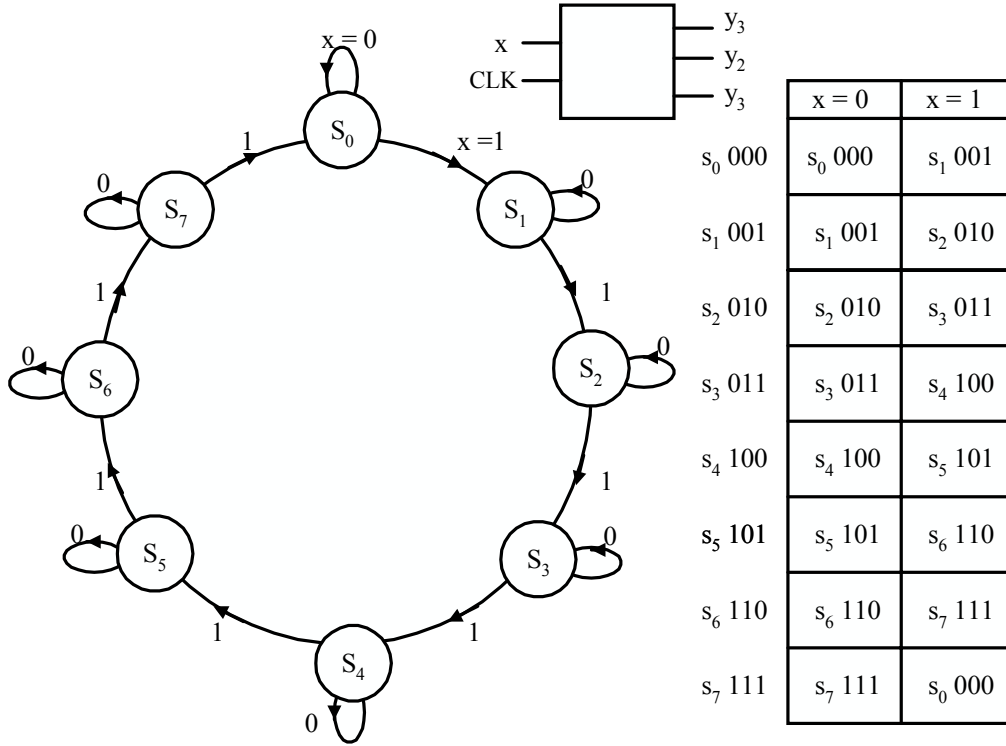
x girişine gelen darbeleri iki tabanında sayan, sekizinci darbe ile başlangıç durumu olan 0 a dönen bir sayıcı, modulo 8 ileri sayıcı, tasarlayalım.

1.Sözle tanım : Tek girişli ve üç çıkışlı bir makinedir. Girişine gelen darbeleri iki tabanında sayar. Her 8 darbeden sonra 0 başlangıç durumuna geçer. Bu işi yapacak 8 durumlu bir ardışıl devre düşünelim öyle ki $S_0, S_1, S_2, S_3, S_4, S_5, S_6, S_7$, durumları sırasıyla makinenin 8, 1, 2, 3, 4, 5, 6, 7 darbe almış olduğunu gösterebilir. Buna göre makineye ilişkin durum tablo ve diyagramı şekil 8.4 deki gibi bulunur. Bir sonraki durum fonksiyonlarının aynı zamanda çıkış fonksiyonları olarak alıp, bir Moore makinesi oluşturabiliriz.

2. Durum indirgemesi Durum indirgemesini bundan sonraki anlatacağız, varsayalım ki indirgeme işlemi yaptık ve eşdeğer durumlar yok. O halde sekiz durum olduğuna göre

üç durum değişkenine ve dolayısıyla üç bellek elemanına gereksinim vardır. Bellek elemanı olarak ta tetikleme elemanını seçelim.

6. **Durum kodlaması** Sekiz durumu $S_0, S_1, S_2, S_3, S_4, S_5, S_6, S_7$, sırasıyla 000, 001, 010, 011, 100, 101, 110, 111 biçiminde kodlayalım. MSB (en çok ağırlıklı bit) bit y_1 , en az ağırlıklı bit te (LSB) y_3 olsun. Bu biçimde kodlama bellek elemanları çıkışlarının, aynı zamanda sayıcının çıkışları olmasını sağlar. Kodlanmış durum tablosu Şekil 8.4 te verilmiştir.

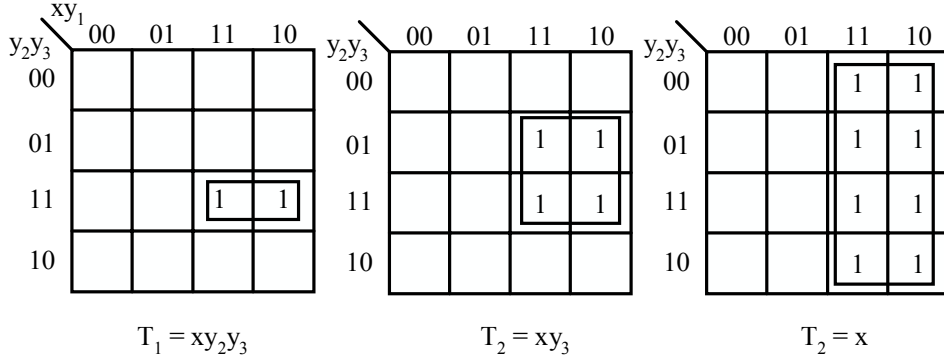


Şekil 8.4 Modulo 8 ileri sayıcının durum diyagramı ve tablosu

4, 5. Bellek elemanları giriş fonksiyonlarının ve çıkış fonksiyonlarının bulunması ve indirgenmesi Tetikleme bellek elemanlarının giriş fonksiyonları ve makinenin çıkış fonksiyonları, durum diyagramı veya tablosu ile tetikleme elemanının ters tanım bağıntılarından yararlanarak, Şekil 8.5 deki gibi elde edilir. Ters tanım bağıntıları kullanılırken, bellek elemanı giriş fonksiyonu, y ve Y indislerinin aynı olmasına dikkat edilmelidir. T bellek elemanlarının giriş fonksiyonlarının Karnaugh diyagramında indirgenmiş ifadeleri, Şekil 8.5 te verilmiştir.

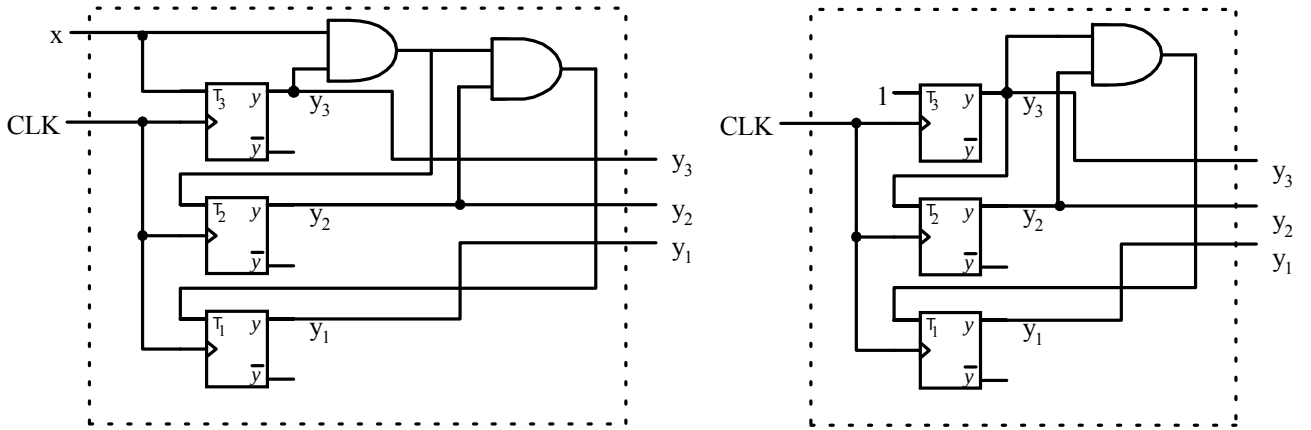
x	y_1	y_2	y_3	Y_1	Y_2	Y_3	T_1	T_2	T_3
0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0	0	0
0	0	1	0	0	1	0	0	0	0
0	0	1	1	0	1	1	0	0	0
0	1	0	0	1	0	0	0	0	0
0	1	0	1	1	0	1	0	0	0
0	1	1	0	1	1	0	0	0	0
0	1	1	1	1	1	1	0	0	0

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



Şekil 8.5 Bellek elemanları giriş fonksiyonlarının indirgenmiş ifadeleri

6.Lojik devrenin gerçekleştirilmesi Son adımda bulduğumuz indirgenmiş fonksiyonlarla, modulo 8 ileri sayıcı devreyi, Şekil 6 2 de verilen genel ardışıl devre yapısında, Şekil 8.6 a daki gibi çizebiliriz.



Şekil 8.6 Modulo 8 sayıcıya ilişkin lojik devreler

Yukarıda incelediğimiz sayıcı x girişine gelen darbeleri değil de doğrudan saat girişine gelen darbeleri sayabilirdi. O zaman x girişleri kaybolacak ve bir sonraki durum fonksiyonlarında ki bağımsız değişken sayısı bir azalmış olacaktı. Bu durum da bulunacak devre Şekil 8.6 b de verilmiştir.

8.3 Özel bir sayıcı devresi

Saat girişine gelen, saat darbeleriyle 0, 2, 4, 7 ve tekrar 0, 2.. sayan bir sayıcı tasarlayalım. Böyle bir makineyi iki biçimde düşünebiliriz. Birincisi üç durum değişkenli 8

durumlu ve çıkışları durum değişkenleri olan makinedir. ikincisi iki durum değişkenli dört durumlu ayrıca üç çıkışı olan bir makinedir.

ilk makineye ilişkin durum tablosu Şekil 8.7 deki gibidir. JK bellek elemanlarıyla gerçekleştirme yapmak istersek; bu bellek elemanlarının ters tanım bağıntılarından yararlanarak bulunan bellek elemanları giriş fonksiyonları da Şekil 8.7 de verilmiştir. Görüldüğü gibi 0, 2, 4, 7 durumlarının dışındaki 1, 3, 5, 6 durumlarına makine hiç gelmemektedir. O halde hiç gelinmeyen durumlara ilişkin bir sonraki durum fonksiyonlarının aldığı değerler ve dolayısıyla bellek elemanları giriş fonksiyonlarının aldığı değerler keyfidir, 0 veya 1 seçilebilir. Bulunan fonksiyonların indirgemesi ve bunlara karşı düşen devre şekil 8.7 de verilmiştir. Görülüyor ki çıkışlar doğrudan bellek elemanlarının çıkışları olarak alınmıştır.

Örneğimizde kullanılmayan durumlar için bir sonraki durumlar ve çıkışlar keyfi alındı. Bu durum ile genelde de karşılaşılabılır. Eğer gerekli olan durum değişkeni sayısı n , durum sayısı m ise $2^n < m$ koşulu altında $(2^n - m)$ durum için bir sonraki durum ve çıkışlar keyfi alınır. Örneğin $m = 6$ durumlu bir makine gerçeklemek için, $n = 3$ durum değişkenine gereksinim vardır, o halde $(2^3 - 6) = 2$ durum için bir sonraki durum ve çıkış fonksiyonları keyfi alınmalıdır.

y_1	y_2	y_3	Y_1	Y_2	Y_3	J_1	K_1	J_2	K_2	J_3	K_3
0	0	0	0	1	0	0	k	1	k	0	k
0	1	0	1	0	0	1	k	k	1	0	k
1	0	0	1	1	1	k	0	1	k	1	k
1	1	1	0	0	0	k	1	k	1	k	1
0	0	1	k	k	k	k	k	k	k	k	k
0	1	1	k	k	k	k	k	k	k	k	k
1	0	1	k	k	k	k	k	k	k	k	k
1	1	0	k	k	k	k	k	k	k	k	k

y_1y_2	00	01	11	10
y_3				
0	0	1	k	k
1	k	k	k	k

$J_1 = y_2$

y_1y_2	00	01	11	10
y_3				
0	1	k	k	1
1	k	k	k	k

$J_2 = 1$

y_1y_2	00	01	11	10
y_3				
0	0	0	k	1
1	k	k	k	k

$J_3 = y_1$

y_1y_2	00	01	11	10
y_3				
0	k	k	k	0
1	k	k	1	k

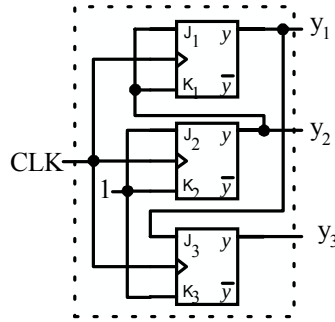
$K_1 = y_2$

y_1y_2	00	01	11	10
y_3				
0	k	1	k	k
1	k	k	1	k

$K_2 = 1$

y_1y_2	00	01	11	10
y_3				
0	k	k	k	k
1	k	k	1	k

$K_3 = 1$



Şekil 8.7 (0-2-4-7) sayan bir sayıcının Moore makinesi olarak gerçekleştirilmesi

Yukarıdaki örnekte (0-2-4-7) sayan sayıcıyı, bellek elemanı sayısını minimum tutarak ta gerçekleştirebiliriz. Bu makinenin, bir önceki durumunu (çıkışı) hatırlaması, yeni saat darbesi ile de istenen çıkışı vermesi yeter. Durumları

A durumu (kodu 00):: Daha önceki saat girişi ile (111) çıkışının görüldüğü durum

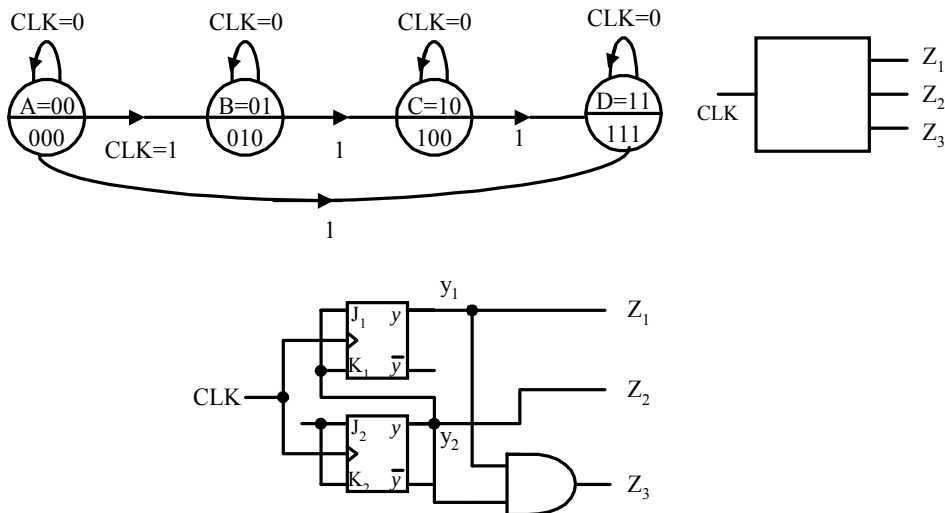
B durumu (kodu 01):: Daha önceki saat girişi ile (000) çıkışının görüldüğü durum

C durumu (kodu 10):: Daha önceki saat girişi ile (010) çıkışının görüldüğü durum

D durumu (kodu 11):: Daha önceki saat girişi ile (100) çıkışının görüldüğü durum

biçiminde tanımlarsak, dört durumlu (iki durum değişkenli) dolayısıyla iki bellek elemanlı, istediğimiz sayıcı gerçekleştirilebilir. Oysa aynı sayıcı için yukarıda üç bellek elemanı kullanmıştık. Tanımladığımız durumlar için durum diyagramı, tablosu, ters tanım bağıntısından yararlanarak JK bellek elemanlarının giriş fonksiyonları ve bunlara karşı düşen devre aşağıdaki Şekil 8.8 de verilmiştir.

y_1	y_2	Y_1	Y_2	Z_1	Z_2	Z_3	J_1	K_1	J_2	K_2
0	0	0	1	0	0	0	0	k	1	k
0	1	1	0	0	1	0	1	k	k	1
1	0	1	1	1	0	0	k	0	1	k
1	1	0	0	1	1	1	k	1	k	1
				$Z_1=y_1$	$Z_2=y_2$	$Z_3=y_1y_2$	$J_1=y_2$	$K_1=y_2$	$J_2=1$	$K_2=1$



Şekil 8.8 dört durumlu 0,2,4,7 sayan sayıcı

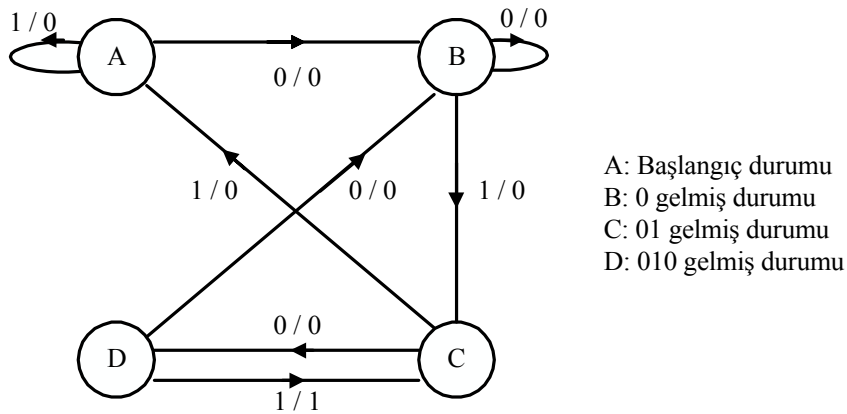
8.4 Dizi detektörü

Tek girişli tek çıkışlı bir makinedir. Girişine 0101 (0 ilk gelen) dizisi geldiğinde çıkışında 1, aksi halde 0 verir. örneğin 010101 girişi için çıkış 000101 dir. Makinenin çıkışı, son 4 girişine gelen işarete göre, belirlenmektedir (Şekil.8.9). Yani iç içe girmiş diziler de değerlendirilmektedir.



Şekil.8.9. Dizi detektörü.

Dört durumlu bir durum diyagramı düşünelim. A başlangıç durumu, B makineye 0 gelmiş durumu, C makineye 01 gelmiş durumu ve D makineye 010 gelmiş durumu olsun. Buna göre A durumunda makineye 1 gelirse makine yine A ya dönmelidir. Çünkü makinenin diziyi tanıması için girişine önce 0 gelmelidir. Makine A dayken 0 geldiğinde B ye geçer. B de 0 gelirse, makine durumunu korur, çünkü son gelen 0 dır. B de 1 girişi gelirse son iki giriş 01 olacağından C durumuna geçer. C de 1 gelirse dizi bozulacağından A başlangıcına, 0 geldiğinde ise son üç giriş 010 olacağından D ye gider. Makine D durumundayken 0 gelirse son gelen 0 olduğundan B ye, 1 geldiğinde ise 1 çıkışı verip, son gelen iki giriş 01 olduğundan C durumuna gider. Durum diyagramını Şekil 8.10 da verildiği gibidir.



Şekil 8.10. Dizi detektörü için durum diyagramı.

Makine 4 durumlu olduğuna göre iki durum değişkeni vardır. Durumları A = 00, B = 01, C = 11, D = 10 biçiminde kodlayalım. Bellek elemanı olarak, gecikme elemanını seçelim. Diyagramdan yararlanarak bağımsız değişkenler, bağımlı bir sonraki durum değişkenleri ve çıkış fonksiyonunu doğruluk tablosu biçiminde verebiliriz (Şekil 8.11).

x	y ₁	y ₂	Y ₁	Y ₂	Z
0	0	0	0	1	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	0	0	0

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

Şekil 8.11. Dizi detektörü için bir sonraki durum ve çıkış fonksiyonlarının doğruluk tablosu.

Gecikme elemanın bir sonraki durum fonksiyonu giriş fonksiyonuna eşit olduğundan $D_1 = Y_1$ ve $D_2 = Y_2$ dir Bu fonksiyonlara ilişkin Karnaugh diyagramları, indirgenmiş fonksiyonlar ve dizi detektörünün lojik devresi Şekil 8.12 de verilmiştir. D_1 ve D_2 fonksiyonları birlikte düşünüldüğünde, üç-girişli üç VE, iki-girişli üç VE, üç-girişli iki VEYA kapısı yerine, iki-girişli bir AYRICALI VEYA, iki-girişli iki VE, üç-girişli bir VE, iki -girişli iki VEYA kapısıyla gerçekleştirilebileceğinin tartışınız.

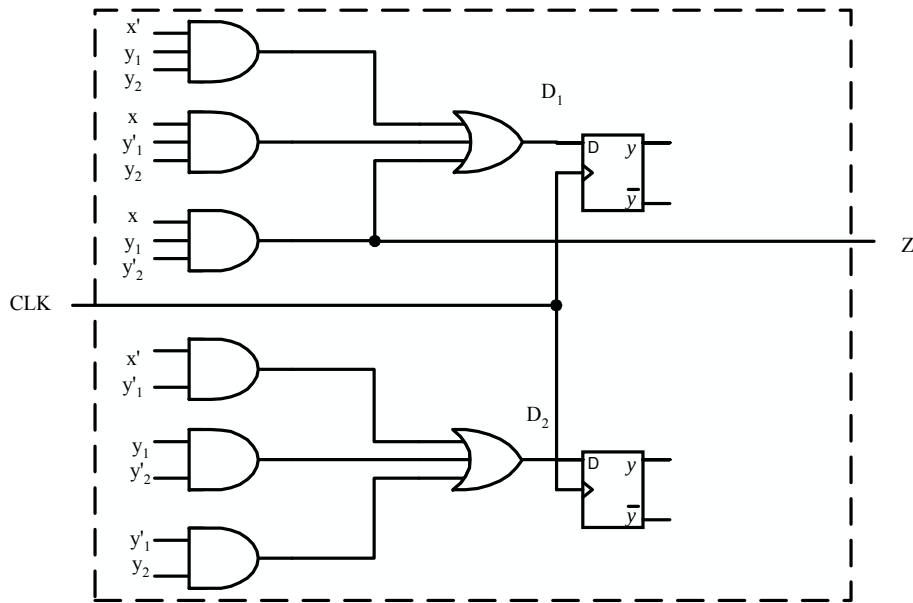
$y_1 y_2$	00	01	11	10
x				
0			1	
1		1		1

$y_1 y_2$	00	01	11	10
x				
0	1	1		1
1		1		1

$y_1 y_2$	00	01	11	10
x				
0				
1				1

$$D_1 = Y_1 = x' y_1 y_2 + x y_1' y_2 + x y_1 y_2' \quad D_2 = Y_2 = x' y_1' + y_1' y_2 + y_1 y_2'$$

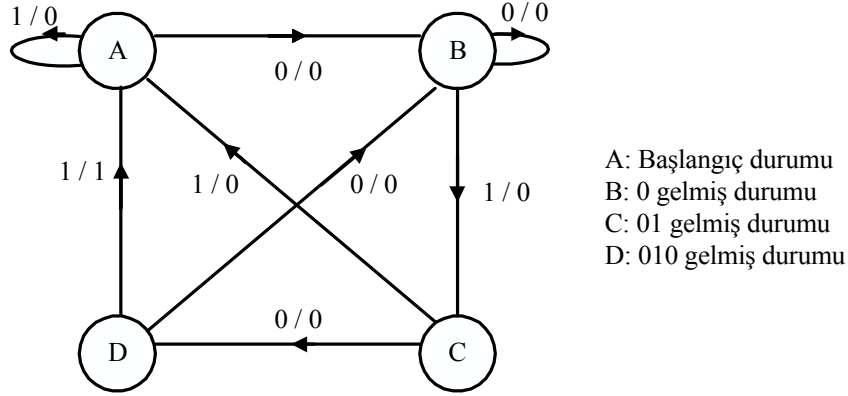
$$Z = x y_1 y_2'$$



Şekil 8.12 Dizi detektörü için bir sonraki durum ve çıkış fonksiyonlarının Karnaugh diyagramları, indirgenmiş fonksiyonlar ve dizi detektörünün lojik devresi

Yukarıda incelediğimiz dizi detektöründe gözlemlediğimiz diziler birbirinin içine girebilmektedir. Örneğin 010101 girişi için 000101 çıkışı elde edileceğine göre birinci

gözlemlenen dizinin son iki elemanı ikinci gözlemlenen dizinin ilk iki elemanı olmaktadır. O nedenle çıkışta ilk 1 gözlemledikten sonra 0 ve tekrar 1 gözlemlenmektedir. Girişine gelen belli uzunluktaki dizileri, birbirinin içine girmeyen ayrı ayrı paketler olarak algılayan dizi detektörü de olabilir. Yani makine bir diziyi detekte ettikten sonra başlangıç durumuna dönüp yeni bir dizi detekte etmeye başlayabilirdi. Bu durumda örnek makinede giriş 0101 0101 ise çıkışta 0001 0001 veya giriş 0101 0110 0101 ise çıkışta 0001 0000 0001 gözlenecekti. Dizi detektörü sentezi yapılırken, istenenin hangisine uyduğu göz önünde tutulmalıdır. İkinci anlattığımız dizi detektörüne ilişkin durum diyagramı Şekil 8.13 te verilmiştir. Görüldüğü gibi bu yeni dizi detektörü için D durumunda, 1 girişi altında makine C durumuna gideceği yerde A durumuna dönmekte ve 1 çıkışı vermektedir.



Şekil 8.13 Birbirinin içine girişmeyen 0101 dizilerini detekte eden bir makineye ilişkin durum diyagramı

Üç uzunluklu bir dizi detektörü tasarlasaydık, üç durum fakat iki durum değişkeni gerekecekti. O zamanda gereksiz olan durum için bir sonraki durum ve dolayısıyla bellek elemanları giriş fonksiyonları k alınabilecekti. Bu dizi detektörü bir Mealy makinesidir. Acaba gereksiz olan durumu da kullanarak, istediğimiz üç uzunluklu diziyi detekte eden makineyi, bir Moore makinesi olarak nasıl gerçekleştirebiliriz, düşününüz.

BÖLÜM 9 ARDIŞIL DEVRELERDE DURUM İNDİRGENMESİ VE DURUM KODLAMASI

Sentez bölümünde sentezin adımlarını incelerken detayına inmeden iki temel problemden söz etmiştik. Şimdi bu problemlerin çözüm yollarının arayacağız. Durum indirgemesi problemi iyi tanımlanmış ve yöntemleri geliştirilmiştir.

Durum kodlaması problemi ise bizi Boole fonksiyonlarının karmaşıklığının tanımına götürür. Bütün gereksinimleri karşılayabilecek Boole fonksiyonları karmaşıklığı tanımlanamamıştır. O nedenle çeşitli karmaşıklık tanımları için çeşitli yöntemler verilmiştir. Bu dersin kapsamı içinde basit fakat sınırlı bir yöntem üzerinde durulacaktır.

1.1 Durum indirgemesi (state reduction)

Bir ardışıl makinede eşdeğer durumların saptanıp elenmesinin, bu makinenin gerçekleştirilmesinde iki yararı olabilir:

1. Durum indirgemesi, durum değişkenlerinin sayısının ve dolayısıyla bellek elemanı sayısının azalmasına neden olabilir.

Örneğin üç durum değişkeni (dolayısıyla üç bellek elemanı) gerektiren beş durumlu bir makinede, bir eşdeğer durum olsun. Eşdeğer durum elendiğinde makine dört durumlu olur ve iki durum değişkeni gerektirir. Bu makine üç bellek elemanı yerine iki bellek elemanı ile gerçekleştirilebilir.

2. Durum indirgemesi sonucunda durum değişkeni azalmasa bile eşdeğer durum sayısı kadar durum için, bir sonraki durumlar ve çıkışlar keyfi olur. Bu eşdeğer durumlara ilişkin bir sonraki durumların keyfi olması, eşdeğer durumlarda bellek elemanları giriş fonksiyonlarının keyfi değer alması demektir. Buda makinenin gerçekleştirilmesinde, kombinezonsal kısmın daha basitleşmesine neden olabilir. Ayrıca eşdeğer durumlar için çıkış fonksiyonlarının keyfi değer alması da benzer biçimde makinenin kombinezonsal devre kısmının basitleşmesine katkıda bulunur.

Örneğin üç durum değişkeni gerektiren sekiz durumlu bir makinede üç durum eşdeğer olsun. Eşdeğer durumlar elendiğinde beş durumlu bir eşdeğer makine bulunur. Durum değişkeni sayısı ve dolayısıyla bellek elemanı sayısı üçtür, değişmez. Ama üç eşdeğer durum için bir sonraki durum keyfi olacağından, bellek elemanları giriş fonksiyonları da bu eşdeğer durumlar için keyfi değerler alacaktır. Ayrıca çıkış fonksiyonları da söz konusu eşdeğer durumlar için keyfi değer olacaktır. Bellek elemanları giriş fonksiyonlarının ve çıkış fonksiyonlarının gerçekleştirilmesine karşı düşen ardışıl devrenin kombinezonsal kısmı, keyfi değerlerin istenildiği gibi seçilebilmesi dolayısıyla basitleşecektir.

Yukarıda sözü edilen faydalardan dolayı durum indirgemesi dediğimiz eşdeğer durumların saptanması ve elenmesi, ardışıl devre sentezinin ağırlıklı bir adımıdır.

Önce eşdeğer durum tanımı verip sonra bu durumların bulunması için yöntemler verelim. **Eşdeğer durum tanımı:** Birbirinin aynı veya farklı N ve N^* gibi iki ardışıl devre ve bunlara ilişkin A ve B durumlarını düşünelim (Şekil 9.1). X herhangi bir uzunluktaki giriş dizisi, Z ve Z^* da sırasıyla A durumundaki N ve B durumundaki N^* makinelerinin X giriş dizisine karşı düşen çıkış olsun. X , Bölüm 5.2 de tanıtılan G kümesinin elemanlarından oluşan dizidir. Z ve Z^* dizileri, Bölüm 5.2 de tanıtılan $\mathcal{C}$ kümesinin

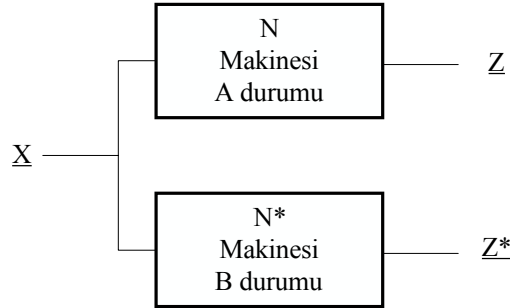
elemanlarından oluşan dizidir. Olabilecek bütün $\underline{X}$ giriş dizileri için $\underline{Z}$ ve $\underline{Z}^*$ dizileri aynı ise A ve B durumları eşdeğerdir denir. Yani

$$\underline{Z} = f(A, \underline{X})$$

$$\underline{Z}^* = f^*(B, \underline{X})$$

dizileri aynıdır. f ve f^* birer çıkış fonksiyon grubunu gösterirler ve aynı makineye ilişkin iseler $f = f^*$ dır.

Bu tanıma uygun olarak iki durumun eşdeğerliğini göstermek, sonsuz giriş dizisi için deneme yapıp çıkışları gözlemeyi gerektirir. O nedenle tanıma uygulayarak eşdeğer durumları bulmak pratik olarak mümkün değildir. Aşağıdaki teorem eşdeğer durumların bulunmasına ilişkin yöntemlerin temelini oluşturur.



Şekil 9.1 Eşdeğer durumların belirlenmesi

Teorem: Aynı bir ardışıl devreye ilişkin A ve B durumlarının eşdeğer olması için gerek ve yeter koşul, bir uzunluklu bütün $\underline{X}$ girişleri için çıkışların aynı olması ve bir sonraki durumların eşdeğer olmasıdır.

Buna göre A ve B durumlarına ilişkin bütün 1 uzunluklu $\underline{X}$ giriş dizileri için hem bir uzunluklu $\underline{Z}$ çıkışları

$$f(A, X) = f(B, X)$$

hem de

$$g(A, X) \equiv g(B, X)$$

olmalıdır. (g) ler bir sonraki durumu simgelemektedir.

Gerek ve yeter koşulları iki girişli, iki çıkışlı ve sekiz durumlu bir makine örneği için yazarak anlaşılmasının kolaylaştırılalım.

$$\begin{array}{ll} f_i(A, 00) = f_i(B, 00) \quad i=1,2 & g(A, 00) = g(B, 00) \\ f_i(A, 01) = f_i(B, 01) \quad i=1,2 & g(A, 01) = g(B, 01) \\ f_i(A, 10) = f_i(B, 10) \quad i=1,2 & g(A, 10) = g(B, 10) \\ f_i(A, 11) = f_i(B, 11) \quad i=1,2 & g(A, 11) = g(B, 11) \end{array}$$

Tanıt: Önce gerek koşul kanıtlayalım. A, B ye eşdeğerse

$$f(A, X) = f^*(B, X) \quad (1)$$

$$g(A, X) \equiv g(B, X) \quad (2)$$

bağıntıları vardır. A'nın B ye eşdeğerliği tanım bağıntısından

$\underline{Y}$ herhangi bir uzunlukta giriş dizisi olmak üzere

$$f(A, \underline{Y}) = f(B, \underline{Y}) \text{ dır.}$$

Bu bağıntı $\underline{Y} = \underline{X}$ özel hali için (1) bağıntısını verir. Öte yandan $\underline{Y}$ dizisi önce $\underline{X}$ sonra $\underline{X}$ olarak seçilirse

$$f[g(A, X), \underline{X}] = f[g(B, X), \underline{X}]$$

bulunur, buda $g(A, X)$ ve $g(B, X)$ in eşdeğerlik tanım bağıntısıdır. (2) bağıntısını verir.

Şimdi (1) ve (2) bağıntıları varken $A \equiv B$ yani $f(A, \underline{Y}) = f(B, \underline{Y})$ olduğunu gösterelim.
(2) bağıntısı

$$f[g(A, X), \underline{X}] = f[g(B, X), \underline{X}]$$

dır. (1) bağıntısıyla birlikte düşünüldüğünde $\underline{Y}$ nin önce X sonra $\underline{X}$ den oluştuğu varsayılırsa

$$f(A, \underline{Y}) = f(B, \underline{Y}) \text{ bulunur.}$$

9.1.1 Eşleştirme tablosu ile durum indirgemesi yöntemi

Eşleştirme tablosuyla durum indirgeme yöntemini, Şekil 9.2 de durum tablosunu verilen Mealy makinesi örneği üzerinden, adım adım inceleyelim.

1. Durumları ikişer ikişer eşleştirmemize yardımcı olacak Şekil 9.3 deki tablo çizilir. Satırlarında a durumu dışında bütün durumlar, sütunlarında ise g durumu dışında bütün durumlar vardır. Bulduğumuz tabloda bir satır ile sütunun kesiştiği kare için, satıra ilişkin durum P sütuna ilişkin durum Q olmak üzere, aşağıdaki işlemleri yapalım.

- bir uzunluklu en az bir giriş için, P ve Q durumları farklı çıkışlar veriyorsa, ilgili kareye X işareti koyalım. Bu işaretin anlamı; P ve Q nun, teoremin birinci koşulu nedeniyle, eşdeğer olamayacağıdır.

Örneğimizde birinci satır ve sütunun kesiştiği kareye X konmuştur çünkü a ve b durumları $x = 1$ girişi için sırasıyla 0 ve 1 farklı çıkışlarını vermektedir.

- Bir uzunluklu bütün girişler için P ve Q durumları, aynı çıkışları ve aynı bir sonraki durumları veriyorsa ilgili kareye $\checkmark$ koyalım. Anlamı teorem gereğince P ve Q durumlarının eşdeğer olduğudur.

Örneğimizde dördüncü satır ile birinci sütunun kesiştiği kareye $\checkmark$ konmuştur. Çünkü ilgili a ve e durumları $x = 0$ girişi için aynı a durumuna, $x = 1$ için ise aynı b durumuna gitmektedir.

- Bir uzunluklu herhangi bir giriş için aynı çıkışı veren (PQ) durum çiftinin, gittiği bir sonraki durum çiftindeki durumlar farklı ise, PQ karesine gidilen durum çifti yazılır. işlem bütün 1 uzunluklu girişler için tekrarlanır. Amaç teoremin iki koşulunun sağlanmasıdır. Bunun anlamı teorem gereğince, PQ durum çiftinin eşdeğer olması için gidilen bir sonraki durum çiftinin eşdeğer olması gerektiğidir.

Örneğimizde üçüncü satır ile birinci sütunun kesiştiği kareye (ag) ve (bf) konmuştur. Çünkü ilgili durumlar a ve d $x = 0$ girişi için aynı 0 çıkışı ve (ag) bir sonraki durumlarını, $x = 1$ girişi için ise aynı 0 çıkışını ve (bf) bir sonraki durumlarını vermektedir.

Tablodaki bütün kareleri yukarıdaki kurallara göre dolduralım (Şekil 9.3).

	x=0	x=1
a	a, 0	b, 0
b	d, 0	e, 1
c	e, 0	b, 1
d	g, 0	f, 0
e	a, 0	b, 0
f	d, 0	g, 1
g	d, 0	f, 1

b					
c		d e			
d	a g	b f			
e				g a	
f		e g	e d	b g	
g		e f	e d	b f	g f
a					
b					
c					
d					
e					
f					

Şekil 9.2 Durum indirgemesi uygulanacak tablo ve eşleştirme tablosu ilk adımı

2. Bu adımda Şekil 9.2 de bir durum çiftinin eşdeğer olması için, eşdeğer olması gereken durumların bulunduğu kareler incelenir. Örneğin üçüncü satır ve birinci sütunun kesiştiği kare ye ilişkin a ve d durumlarının eşdeğer olması için ag ve bf durumlarının eşdeğer olması gerekir ve yeter. Bakalım öylemi? a ve g, kesiştiği karede X olduğundan, eşdeğer değildir. O halde a ve d de eşdeğer olamaz, a ve d nin kesiştiği kareye X konulmalıdır (Şekil 9.3). ag ve bf nin incelenmesi sonucunda bunların kendi aralarında eşdeğer olduğunu görseydik, a ve d nin de eşdeğer olduğunu söyleyebilecek ve ✓ ile işaretleyecektik.

Bu işleme bütün kareler X veya ✓ ile işaretlene kadar devam edilir. Örneğimizde ae ve fg durum çiftlerinin eşdeğer olduğu Şekil 9.3 görülmektedir.

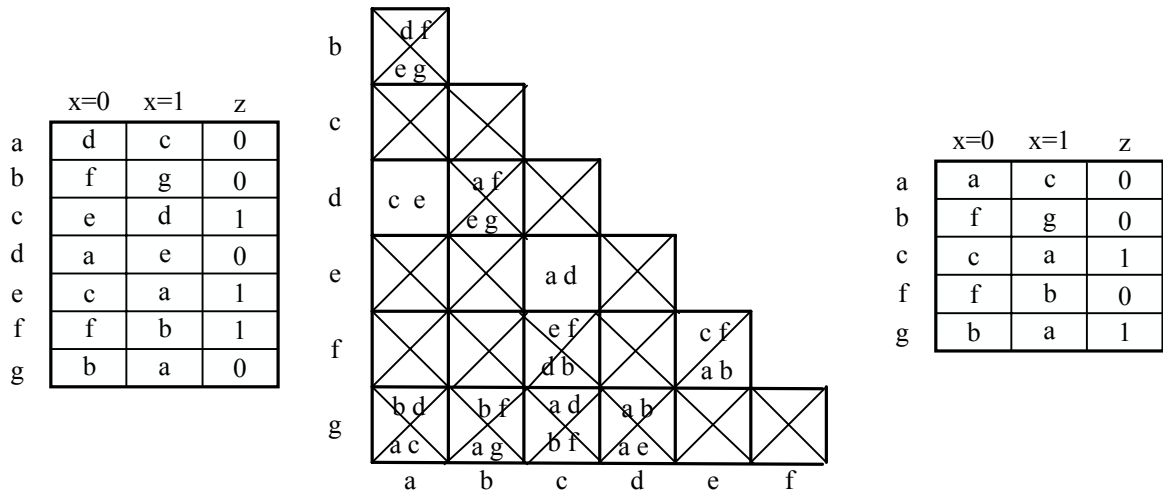
3. Şekil 9.2 te verilen durum tablosunda, a ya eşdeğer e durumunun bulunduğu satır ve f ye eşdeğer g durumunun bulunduğu satırlar atılarak, eşdeğer makineye ilişkin durum tablosu bulunur. Satır atıldıktan sonra bulunan yeni tabloda g görülen yere f, e görülen yere a yazarak, durum indirgemesi yapılmış makinenin durum tablosuna geçilebilir (Şekil 9.3).

b	X					
c	X	de				
d	ag	eb	X			
e	✓			ga		
f		eg	ed	bf	X	
g		ef	ed	bf		✓
	a	b	c	d	e	f

	x=0	x=1
a	a, 0	b, 0
b	d, 0	a, 1
c	a, 0	b, 1
d	f, 0	f, 0
f	d, 0	f, 0

Şekil 9.3 Örnek durum tablosu için eşleştirme yöntemiyle varılan tablo ve buna karşı düşen indirgenmiş durum tablosu

Şimdide bir Moore makinesine ilişkin durum indirgeme örneği yapalım. Şekil 9.4 de örnek Moore makinesi ve eşleştirme tablosu verilmiştir. Eşleştirme tablosundaki X ikinci ve daha sonraki adımlarda elde edilen eşdeğer olmayan durum çiftlerini göstermektedir. Görüldüğü gibi (ad) ve (ce) çiftleri birbirlerinin eşdeğer olmalarını gerektirmektedirler. Eşdeğer olmadıkları kanıtlanamayan bu durum çiftleri eşdeğer olmak zorundadır. Bu sonuca göre indirgenmiş durum diyagramı Şekil 9.4 da verilmiştir.



Şekil 9.4 Bir Moore makinesine ilişkin durum indirgeme örneği

9.1.2 Eşdeğer durum sınıfları ile durum indirgemesi yöntemi

Durum indirgemesi için kullandığımız eşleştirme tablosu yönteminden başka **eşdeğer durum sınıfları yöntemi** vardır. Bu yöntem, giriş dizisi uzunluklarına bağlı olarak, ayırt edilemeyen durumları aynı sınıfa ve ayırt edilebilen durumları da ayrı sınıflara toplamaya dayanır.

Son örneğimizde bu yöntemi açıklayalım. Hiç bir giriş dizisi uygulanmadığında, durumları birbirinden ayırt edemeyiz. Onun için hepsini bir S_0 sınıfın içine koyalım, $S_0 = \{(abcdefg)\}$.

Şekil 9.4 teki durum tablosundan, (abdg) durumlarının, bir uzunluklu girişler altında, çıkışı gözleyerek, birbirlerinden ayırt edilemeyeceğini söyleyebiliriz. Çünkü hepsi aynı 0 çıkışı verirler. Benzer biçimde (cef) durumlarını da, bir uzunluklu bir giriş dizisiyle birbirinden ayırt edemeyiz. Çünkü her biri için çıkış 1 dir. (abdg) ve (cef) bir uzunluklu girişler için eşdeğer durum sınıflarıdır. Biri (abdg) diğeri (cef) sınıflarından olan durum çiftlerinin 1 uzunluklu giriş dizisiyle ayırt edilebileceklerini söyleyebiliriz çünkü 1 uzunluklu giriş için birinci sınıf 0, ikinci sınıf ise 1 çıkışı verir. Buna göre bir bölmeleme yapabiliriz $S_1 = \{(cef) (abdg)\}$.

Şimdi (cef) sınıfını düşünelim $x = 0$ girişi altında cef ($c \rightarrow e, e \rightarrow c, f \rightarrow f$ gider) ye ; $x = 1$ girişi altında ise dab ($c \rightarrow d, e \rightarrow a, f \rightarrow b$) ye gitmektedir. Bir sonraki durumlar (ecf) ve (dab) aynı eşdeğer sınıfındadır, yani iki uzunluklu girişler içinde cef durumları ayırt edilemez. (abdg) sınıfı $x=0$ girişi altında dfab ($a \rightarrow d, b \rightarrow f, d \rightarrow a, g \rightarrow b$) ye gitmektedir. Burada (b) nin gittiği (f) in bulunduğu sınıf ile (adg) nin gittiği (dab) bulunduğu sınıflar farklıdır. Yani iki giriş uzunluklu bir dizi $x = 0$ (veya 1) uygulanıp $z = 0$ gözlendikten sonra $x = 0$ uygulandığında $z = 0$ gözleniyorsa makine (adg) durumlarından birindedir, $z = 1$ gözleniyorsa (b) durumundadır. O halde iki uzunluklu bir dizi (00 veya 10) ile (b) (adg) den ayırt edilebilir, farklı sınıflarda olması gerekir. Diğer taraftan $x = 1$ girişi için abdg cgea ($a \rightarrow c, b \rightarrow g, d \rightarrow e, g \rightarrow a$) ya gitmektedir, yani bir kısmı bir sınıfa diğeri kısmı diğer sınıfa gitmektedir. a ve d, c ve e ye; b ve g, g ve a ya gittikleri için (ad) ve (bg) durumları iki giriş uzunluğunda bir dizi (01 veya 11) ile ayırt edilebilirler. O halde (abdg) sınıfı iki defa bölünerek, $S_2 = \{(cef) (ad) (b)(g)\}$ bölmelemesini elde edilir. Bu bölmeleme şu anlama gelmektedir: (cef) (ad) (b) ve (g) sınıflarının kendi içlerindeki durumlar birbirlerinden iki uzunluklu girişler tarafından ayırt edilemezler. Fakat biri bir sınıftan diğeri diğer bir sınıftan olmak üzere oluşturulan durum çiftleri, iki uzunluklu giriş dizileriyle ayırt edilebilirler.

Örneğin (a) ya 10 (önce 1) veya 11 uygulandığında çıkış 01(önce 0) dir oysa (b) ye aynı 10 veya 11 girişleri uygulandığında çıkış 00 dir.

Üçüncü adımda $x = 1$ girişi altında (cef) sınıfı sırasıyla $c \rightarrow d$, $e \rightarrow a$, $f \rightarrow b$ gitmektedir (b)ve (ad) farklı sınıflarda olduğundan (cef) sınıfı (ce) ve (f) olarak bölmelenir. Yani bu sınıflar üç giriş dizisiyle ayırt edilebilirler. (ad) $x = 0$ ve $x = 1$ girişleri altında aynı sınıf içindeki durumlara gittiğinden üç giriş dizisiyle ayırt edilemezler. Buna göre $S_3 = \{ (ce) (f) (ad) (b)(g) \}$ bölmelemesi elde edilir.

Bölmeleme işlemi, ard arda gelen S eşdeğer sınıflar kümeleri eşit olduğunda, durdurulur. Bizim örneğimizde $S_3 = \{ (ce) (f) (ad) (b) (g) \}$ bölmelemesi, $S_3 = S_4$ olduğundan, son varılan bölmelemedir. Buda gösteriyor ki (ce) ve (ad) durum çiftleri olabilecek bütün giriş uzunlukları için birbirinden ayırt edilemezler, yani eşdeğerdirler. Eşleştirme tablosuyla da bu sonuca varmıştık. Açıkladığımız bölmeleme adımlarına ilişkin bir tablo Şekil 9.5 da verilmiştir. Belli bir giriş altında farklı sınıflara giden bir sınıfın, nasıl bölmelenmesi gerektiğini göstermek için o sınıftaki durumlar koyu veya italik yazılarak farkedirilmiştir.

$$\begin{aligned}
 S_0 &= \{ (a \ b \ c \ d \ e \ f \ g) \} \\
 z &= 0 \ 0 \ 1 \ 0 \ 1 \ 1 \ 0 \\
 S_1 &= \{ (\ c \ e \ f) (a \ b \ d \ g) \} \\
 S_1 &= \{ (\ c \ e \ f) (\mathbf{a} \ \mathbf{b} \ \mathbf{d} \ g) \} \\
 x=0 & (e \ c \ f) (d. \ \mathbf{f}. \ a. \ b) \\
 x=1 & (d \ a \ b) (\mathbf{c} \ \mathbf{g} \ \mathbf{e} \ a) \\
 S_2 &= \{ (\ c \ e \ f) (\mathbf{b}) (\mathbf{a} \ \mathbf{d}) (g) \} \\
 S_2 &= \{ (\ c \ e \ f) (b) (a \ d) (g) \} \\
 x=0 & (e \ c \ f) (f) (d \ a). (b) \\
 x=1 & (d \ a \ \mathbf{b}) (g) (c \ e) (a) \\
 S_3 &= \{ (c \ e) (f) (b) (a \ d) (g) \} \\
 x=0 & (e \ c) (f) (f) (d \ a). (b) \\
 x=1 & (d \ a) (b) (g) (c \ e) (a) \\
 S_3 &= S_4
 \end{aligned}$$

Şekil 9.5 Örnek makinenin durum indirgemesinin eşdeğer durum sınıfları yöntemindeki bölmeleme adımları

Farklı sınıflarda bulunan durumları ayırt edebilmek için uygulanacak giriş dizisini, sondan başa doğru bulabiliriz. Örneğin (c) yi (f) den ayırt etmek için 1kk (önce gelen giriş 1, $k = 0$ veya 1 i gösterir) dizisini uygulamak gerekir. S_3 te c ve f ayrı sınıflardadır. Bunun için s_2 de 1 uygulamak gerekir, bir sonraki durumlar ise d, b dir. Bunları ayırt etmek için S_1 de $x = 0$ veya 1 uygulamak gerek, bir sonraki durumlar (a,f) veya (eg) yi ayırt etmek

için ise S_0 da herhangi bir girişi uygulamak yeter. Şekil 9.6deki tablo, uygulanacak giriş dizisini özetlemektedir.

durum	f	b	f
giriş	1	0(1)	1(0)
çıkış	1	0	1
durum	c	d	a
giriş	1	0(1)	1(0)
çıkış	1	0	0

Şekil 9.6 Örnek makinenin c ve f durumlarının ayırt etmek için uygulanması gereken giriş dizisinin bulunması

Burada eşdeğer durum sınıfları yöntemi Moore makinesi üzerinde incelendi, Mealy makinesi içinde tutulacak yöntem aynıdır.

Genel olarak göstermek mümkündür ki durum sayısı m olan bir makine için en fazla $m-1$ bölmeleme adımıyla sonuç bölmelemeye varılır.

9.2 Durum kodlaması (state assignment)

Ardışıl devre sentezinin durum indirgemesinden sonraki adımı, durum kodlamasıdır. Durumları öyle kodlayalım ki bir sonraki durum fonksiyonları, dolayısıyla bellek elemanları giriş fonksiyonları ve çıkış fonksiyonları gerçeklemesi, olabildiğince basit yani düşük maliyetli olsun. Bu işleme durum kodlaması diyoruz. Durum kodlaması problemi, farklı kodlar için elde edilecek fonksiyonların (bir sonraki durum ve çıkış fonksiyonları) karmaşıklığının karşılaştırılması problemine dönüşür. O halde Boole fonksiyonlarının karmaşıklığı tanımı yapılmalıdır.

Her zaman geçerli olan bir karmaşıklık tanımı verilemediğinden, farklı tanımlar verilmiş ve bunlara ilişkin farklı yöntemler geliştirilmiştir. Genel tanımın verilemeyeşi, bu problemin tek bir çözümünün olmaması demektir. O nedenle farklı yöntemler uygulayıp sonuçları karşılaştırarak seçim yapmak gerekmektedir.

Ayrıca seçilen bellek elemanın tipine göre, bir sonraki durum fonksiyonu karmaşıklığının, bellek elemanının giriş fonksiyonu karmaşıklığına yansımalarının da saptanması gerekmektedir. Genel olarak bunu bellek elemanları ters tanım bağıntılarından göstermek mümkündür.

Önce durum sayısı belli iken, verilebilecek farklı kod sayısının ne olduğunu, bunlar arasında hangilerinin gerçekleştirme açısından eşdeğer olduklarını inceleyeceğiz. Bu incelemenin sonucu görülecek ki 4 duruma kadar seçebileceğimiz farklı kodlar sayısı 3'dür. O nedenle bu üç koda ilişkin gerçekleştirmelerin hepsinin yapılıp, sonuçlar karşılaştırılarak, en uygun durum kodlaması seçilebilir.

Durum sayısı 4'ten fazla olduğunda, seçilebilecek farklı kodlar sayısı hızla artacağından başka çözümler aranmalıdır. Bunun içinde bir sonraki durum fonksiyonlarındaki 1'lerin, olabildiğince komşu olmasını sağlayacak kodun seçilme yöntemi verilecektir.

9.2.1 Kod sayısı ve eşdeğer durum kodları

(μ) durumlu bir makine için $r \geq \log_2 \mu$ bağıntısını sağlayan en küçük tam sayı r (aynı zamanda durum değişkeni sayısı) olmak üzere 2^r farklı kod vardır. (μ) duruma verebileceğimiz kodların sayısı da $2^r / (2^r - \mu)!$ 'dır. Örneğin $\mu = 3$ durumlu makine için $r = 2$ ve kod sayısı $2^2 = 4$ (00, 01, 10, 11) tür. Üç duruma vereceğimiz $4! / (4-3)! = 24$ farklı durum kodu Şekil 9.6 da verilmiştir.

y_1y_2	1	2	3	4	5	6	7	...	19	20	21	22	23	24
a	00	00	00	00	00	00	01		11	11	11	11	11	11
b	01	01	10	10	11	11	00		00	00	01	01	10	10
c	10	11	01	11	01	10	10		01	10	00	10	00	01

Şekil 9.6 Üç durumlu makinenin durum kodlamasına ilişkin olabilecek 24 alternatif durum kodu

Görüldüğü gibi her bir durum kodlaması iki sütundan oluşuyor, birinci sütun bir bellek elemanına diğeri de diğer bellek elemanına ilişkindir. Tabloda bir koda ilişkin sütunların yer değiştirmesiyle bulunan yeni durum kodu, ayrı bir durum kodu gibi görünür, oysa gerçekleştirme bakımından bunlar eşdeğerdir. Bu iki kod için yapılan gerçekleştirmelerde, yalnızca bellek elemanları yer değiştirdiği görülecektir ($y_1 \leftrightarrow y_2$, $Y_1 \leftrightarrow Y_2$). O nedenle bu kod çiftinden biri atılabilir. Örneğin 1 kodunda sütunlar yer değiştirirse 3 kodu bulunur.

Bir koddaki değişkenlerden biri, bir başka kodun değişkeninin tümleyeni biçiminde olsun. Bu kodlara karşı düşen gerçekleştirilmelerde her bir bellek elemanı için aşağıdaki farklılıklar saptanır.

- SR, JK bellek elemanları için giriş fonksiyonları ve Y, Y' çıkışları yer değiştirir (sonuçta ilave bir maliyet gelmez)
- T bellek elemanı için yalnızca Y, Y' yer değiştirir (sonuçta ilave bir maliyet gelmez)
- D bellek elemanı için giriş fonksiyonunun tümleyeninin alınması gerektiğinden ilave bir tümleyen kapısı gelir ve Y, Y' çıkışları yer değiştirir.

Bu sonuca nasıl varıldığını görmek için bellek elemanlarının ters tanım bağıntılarındaki simetriye bakalım. Şekil 9.7 de, Bir A durum kodu ile tümleyen A kodu verildiğinde, şimdiki ve bir sonraki durumlardaki değişiklikler ilk iki satırda görülmektedir. Diğer satırlarda ise bu iki koda ilişkin giriş fonksiyonlarını alacağı değerler görülmektedir. (y_i) lerdeki değişiklikler dışında, giriş fonksiyonlarında görülecek değişiklikler aşağıdaki gibidir.

i. JK ve SR için giriş fonksiyonlarının aldıkları değerlerin yer değiştirmiştir. Yani gerçekleştirmede (J) ile (K) ve (R) ile (S) yer değiştirecektir.

ii T için bir değişiklik yoktur. Yani gerçeklemede aynı giriş fonksiyonu bulunacaktır.

iii. D için ise giriş fonksiyonunun aldığı değer diğerindekinin tümleyenidir. Yani gerçekleştirmede, bir tümleyen kapısı ilave etmek gerekir.

A kodu	$y \rightarrow Y$	$0 \rightarrow 0$	$1 \rightarrow 1$	$0 \rightarrow 1$	$1 \rightarrow 0$
Tümleyen A kodu	$y' \rightarrow Y'$	$1 \rightarrow 1$	$0 \rightarrow 0$	$1 \rightarrow 0$	$0 \rightarrow 1$
A kodu	JK	0K	K0	1K	K1
Tümleyen A kodu	JK	K0	0K	K1	1K
A kodu	SR	0K	K0	10	01
Tümleyen A kodu	SR	K0	0K	01	10
A kodu	T	0	0	1	1
Tümleyen A kodu	T	0	0	1	1
A kodu	D	0	1	1	0
Tümleyen A kodu	D	1	0	0	1

Şekil 9.7 Bellek elemanlarının ters tanım bağıntılarının tümleyen kodlar için karşılaştırılması

Şimdi daha önce gerçekleştirdiğimiz dizi detektörü örneği üzerinde, eşdeğer olan durum kodlamalarını görelim (Şekil 9.8). (II) durum kodlaması (I) kodlamasındaki sütunların yer

değişiminden, (III) kodlaması da (I) kodlamasındaki sütunların tümleyeninden elde edilmiştir. Dizi detektörünün durum tablosu ve üç durum kodu Şekil 9.8 de verilmiştir.

Durum kodu (I)	Durum kodu (II)	Durum kodu (III)	Şimdiki durum	x=0	x=1
00	00	11	A	B,0	A,0
01	10	10	B	B,0	C,0
11	11	00	C	D,0	A,0
10	01	01	D	B,0	C,1

Şekil 9.8 Dizi detektörünün kodlanmasındaki eşdeğer kodlara örnek

(I), (II) ve (III) durum kodları için JK bellek ve D bellek elemanlarının giriş fonksiyonları ve çıkış fonksiyonları, aşağıda Şekil 9.9 da verildiği gibidir. Yani kod sütunlarının yer değiştirmesi bellek elemanlarının adlarının değişmesidir. Bir başka deyişle bellek elemanları giriş fonksiyonlarında ve çıkış fonksiyonlarında $1 \leftrightarrow 2$ indis dönüşümü yapılır. Devrede ise bellek elemanlarının indisleri değiştirilir. Kod sütunlarının tümleyenlerinin alınması JK bellek elemanlarının girişlerinin yer değiştirmesine, D bellek elemanı girişinin ise tümleyeninin alınmasına denk olmaktadır.

(I) durum kodu için	(II) durum kodu için	(III) durum kodu için
$J_1 = x y_2$	$J_1 = x' + y_2$	$J_1 = x' y_2 + x y_2'$
$K_1 = x' y_2' + x y_2$	$K_1 = y_2$	$K_1 = x y_2'$
$J_2 = x' + y_1$	$J_2 = x y_1$	$J_2 = y_1'$
$K_2 = y_1$	$K_2 = x' y_1' + x y_1$	$K_2 = x' + y_1'$
$D_1 = x' y_1 y_2 + x y_1' y_2 + x y_1 y_2'$	$D_1 = x' y_2' + y_1' y_2 + y_1 y_2'$	$D_1 = x y_1' y_2' + x y_1 y_2 + x y_1 y_2'$
$D_2 = x' y_1' + y_1' y_2 + y_1 y_2'$	$D_2 = x' y_1 y_2 + x y_1' y_2 + x y_1 y_2'$	$D_2 = x y_1' y_2' + y_1 y_2$
$Z = x y_1 y_2'$	$Z = x y_1' y_2$	$Z = x y_1' y_2$
	Dönüşüm: $1 \leftrightarrow 2$	Dönüşüm: $y_i \leftrightarrow y_i', J_i \leftrightarrow K_i, D_i \leftrightarrow D_i'$

Şekil 9.9 Eşdeğer kodların, gerçekleştirilecek fonksiyonlara etkisi

Yukarıda açıkladığımız gibi birbirine eşdeğer durum kodları elendiğinde, görülür ki 3 veya 4 durumlu makineler için, gerçekleştirilecek devre üzerinde etkin olan, birbirinde farklı durum kodlaması sayısı 3 tür. (Şekil 9.10)

	Üç durum için kod			Dört durum için kod		
Durumlar	I	II	III	I	II	III
A	00	00	00	00	00	00
B	01	01	11	01	01	11
C	10	11	01	10	11	01
D				11	10	10

Şekil 9.10 Üç ve dört durumlu bir makine için verilebilecek farklı durum kodlamaları

Dört duruma kadar verilebilecek eşdeğer durum kodu üç tane olduğundan, her bir kodlama için elde edilecek devreler karşılaştırılarak, en uygunu seçilebilir. Dört durumdan daha fazla durumlu ardışıl devrelerde, Şekil 9.11 deki tabloda görüldüğü gibi, durum kodlaması sayısı hızla artmaktadır. O nedenle her bir kodlama için devreleri gerçekleyip karşılaştırma yapmak, pratik değildir. Bu amaçla durum sayısı 7-8 civarında olan ardışıl devrelere için bir yöntem, bundan sonraki bölümde incelenecektir. Ayrıca bu dersin kapsamı içine alınmayan ve Boole fonksiyonunun karmaşıklık tanımına bağlı olarak geliştirilmiş genel yöntemler vardır.

Durum sayısı	durum eğişkeni sayısı	eşdeğer lmayan kodlar
2	1	1
3	2	3
4	2	3
5	3	140
6	3	420
7	3	840
8	3	840
9	4	10810800
.	.	.
16	4	$5.5 \cdot 10^{10}$

Şekil 9.11 Durum sayısına bağlı olarak seçilebilecek, farklı durum kodlaması sayısını gösteren tablo

9.2.2 Durum kodlaması

Yöntemin hedefi kodlama ile bir sonraki durum fonksiyonlarında, dolayısıyla bellek elemanları giriş fonksiyonlarında, komşu doğru (yanlış) minterimleri maksimize etmektir. Durum kodlamasını öyle yapalım ki bellek elemanlarının giriş fonksiyonlarına ilişkin 1(0) ler karnaugh diyagramında mümkün olduğunca komşu karelere gelsin.

İki duruma verilen kod, yalnızca bir değışkende fark ediyorsa; bu iki koda **komşu kod** denir. Örneğin 001, 000 a komşudur ama 010 e komşu değildir. Benzer biçimde komşu girişler de tanımlanabilir.

Kodlama işlemini, aşağıdaki kurallara göre adım adım yapabiliriz.

1. Belli bir giriş altında, aynı bir sonraki duruma giden durum çiftlerine komşu kodlar verelim.

Bu demektir ki bir sonraki duruma hangi kod verilirse verilsin, onu temsil eden bir sonraki durum fonksiyonlarının hepsinde, birer 1. mertebeden 1 ve/veya 0 alt küpü kesinlikle vardır.

Belli bir giriş altında öyle 4 durum varsa ki bir sonraki durumları aynı; o zaman bu 4 durumdan her biri en az diğer ikisi ile komşu kodlandırılarak, bir sonraki durum fonksiyonlarının hepsinde, 2. mertebeden 1 ve/veya 0 alt küpleri elde edilir. Örneğin Şekil 9.2 deki durum tablosunda ACEG durumlarının hepsi $x = 0$ girişi altında B durumuna gitmektedir. ACEG durumlarına komşu kodlar verirse bir sonraki durum fonksiyonlarının hepsinde, 2. mertebeden birer alt küp oluşacaktır. Benzer biçimde 2^k lık alt küpler oluşturma kuralları saptanabilir.

2. Bir durumun komşu girişler için gittiği bir sonraki durum çiftine, komşu kodlar verilir.

Bu demektir ki söz konusu durum çiftini temsil eden bir sonraki durum fonksiyonlarının, birisi dışında, hepsinde 1. mertebeden 1 ve/veya 0 alt küpü kesinlikle vardır.

ikişer ikişer komşu dört giriş için (örneğin 00,01,11,10), bir durumun gittiği dört durum, kendi aralarında ikişer ikişer komşu olacak biçimde kodlanabilirler. Bu halde iki tane bir sonraki durum fonksiyonu dışındaki bütün bir sonraki durum fonksiyonlarında, 2. mertebeden birer 1 ve/veya 0 alt küpü kesinlikle oluşur.

3. Belli bir giriş altında aynı çıkışı veren durumlara birbirinin komşusu kodlar verilmelidir. Bu adım 1. adımdaki işlemin, bir sonraki durum fonksiyonu yerine, çıkış fonksiyonu için uygulanmasıdır. Söz konusu komşu kodlu durum çifti, komşu girişler için de aynı çıkışı veriyorsa, tercih edilmelidir.

Yukarıdaki adımlardan anlaşıldığı gibi, yöntem belli koşulları sağlayan durumlara komşu kodlar verilmesine dayanmaktadır. Kuralların getirdikleri yararlar küçükten büyük numaralıya doğru azalmaktadır. O nedenle durum kodlamasına 1. kurala ilişkin kümelerden başlamakta yarar vardır.

Komşu kodların verilmesini, en iyi Karnaugh diyagramı üzerinde yapabiliriz. Şimdi yöntemi Şekil 9.12 te verilen örnek durum tablosu üzerinde inceleyelim

Komşu kodlar verilmesi gereken durum kümeleri her bir adım için aşağıdaki gibidir:

1. $x = 0$ için (D,F), (A,C,E,G) [veya (AC),(AE),(AG),(CE),(CG),(EG)]; $x = 1$ için (A,B,D,F) [veya (AB),(AD),(AF),(BD),(BF),(DF)], (E,G) durumları komşu kodlandırılabilir.

2. $x = 0$ ve $x = 1$ komşu girişleri için (B,C); (C,D); (B,E); $2^*(C,F)$; $2^*(B,G)$ durum çiftleri komşu kodlandırılabilir.

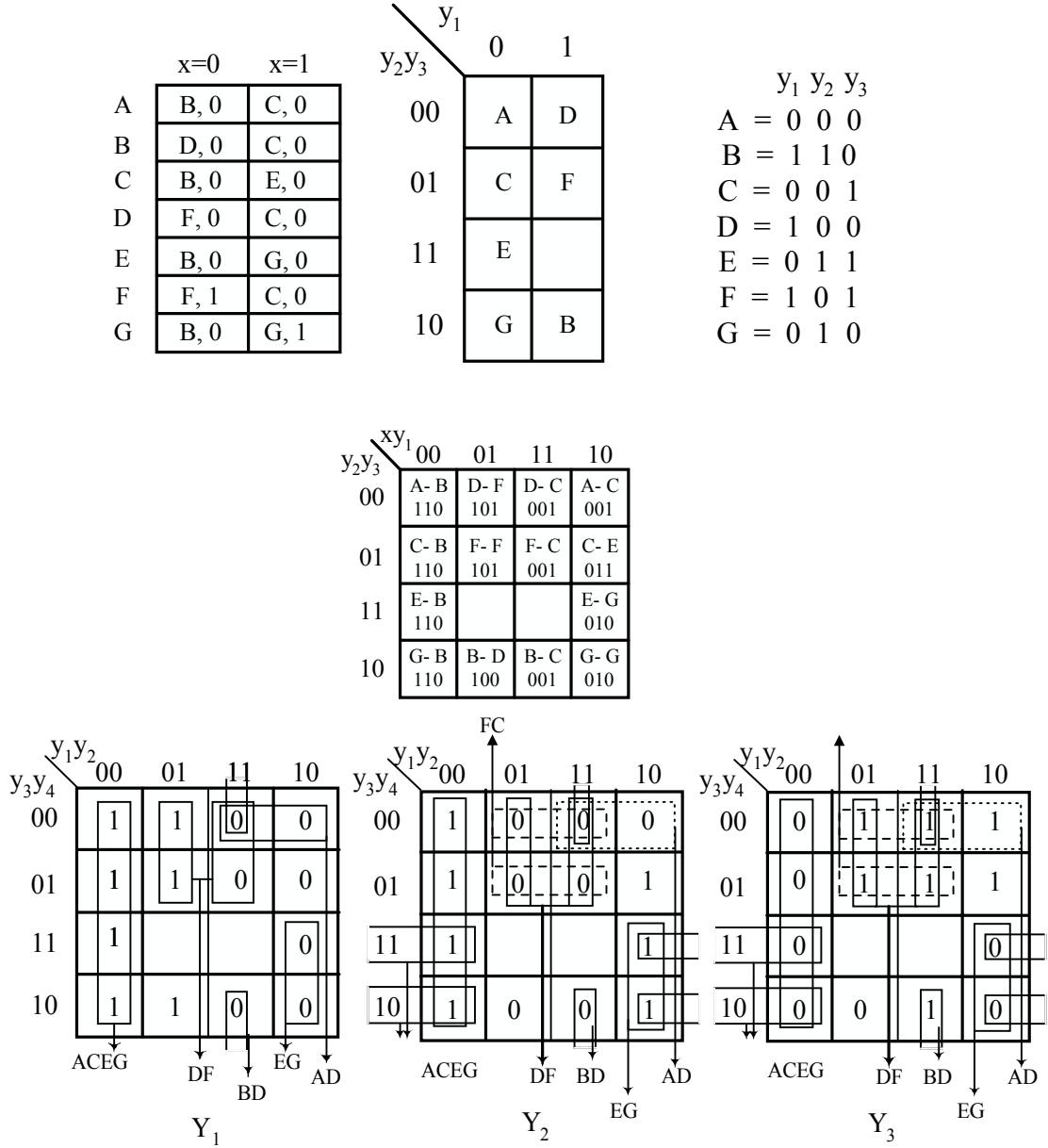
3. $x = 0$ girişi için (A,B,C,D,E,G) grubundaki durumlar, $x=1$ girişi için (ABCDEF) grubundaki durumlar ikişer ikişer komşu kodlandırılabilir.

Komşu kod verilmesi gereken durumlar sayısı 4 veya 8 elemanlı kümeler oluşturuyorsa bunların iki ve üçüncü mertebeden küpler oluşturacak şekilde kodlanması indirgeme işlemine daha fazla katkıda bulunur. Bizim örneğimizde bu özellikte iki küme vardır: (A, B, D, F), (A, C, E, G). Ayrıca tekrarlanan kümeler tercih edilmelidir bizim örneğimizde (EG), (D,F), (C,F), (B,G) ikişer defa komşu kodlanması istenen durum çiftleridir.

Şekil 9.12 verilen kodlamanın seçimini adım adım inceleyelim. (ACEG) ve (ABDF) kümelerinin diğer kümelerle ortaklığı aynı olduğundan rasgele bir tanesi (ACEG) yi seçelim. Bu küme Şekil 9.12 teki Karnaugh diyagramındaki gibi komşu kodlanırsa Y_1, Y_2, Y_3 bir sonraki durum fonksiyonlarının her birinde birer 2. mertebeden küp görülür. Bu küp Z çıkış fonksiyonunda da, 3. kurala ilişkin kümeden dolayı, görünecektir. Artık (ABDF) kümesindeki durumların 2. Mertebeden bir alt küp olarak komşu kodlanması mümkün değildir.

1. kuralda iki defa görülen (DF) çiftine, komşu kodlar Şekil 9.12 teki gibi verilsin. Y_1, Y_2, Y_3 bir sonraki durum fonksiyonlarının her birinde ikişer tane 1. mertebeden alt küp görülür. (DF) nin Karnaugh diyagramındaki yerleşiminden (AD) ve (CF) komşulukları da sağlanmış olur. (AD) çifti 1. kuralı sağladığından Y_1, Y_2, Y_3 bir sonraki durum fonksiyonlarının her birinde birer tane 1. mertebeden alt küp görülür. (CF) çifti ise 2. kuralı 2 defa sağladığından Y_2, Y_3 bir sonraki durum fonksiyonlarının her birinde ikişer tane 1. mertebeden alt küp görülür.

Geriye B nin kodlanması kalmaktadır. B Şekil 9.2 te verildiği gibi kodlansın. (BG) komşuluğu 2. kuralı iki defa sağladığından Y_2, Y_3 bir sonraki durum fonksiyonlarının her birinde ikişer tane 1. mertebeden alt küp görülür. Diğer taraftan (BD) çifti 1. kuralı sağladığından Y_1, Y_2, Y_3 bir sonraki durum fonksiyonlarının her birinde 1. mertebeden birer alt küp görülür. Şekil 9.2 te bütün sonuçlar gösterilmiştir.



Şekil 9.12 Durum kodlaması örneği

BÖLÜM 10 YAZMAÇLAR, SAYICILAR, BELLEKLER

MSI (**M**edium **S**cale **I**ntegrated) devreler birinci bölümde gördüğümüz kombinezonsal oldukları gibi ardışıl da olabilirler. Ardışıl MSI devrelerde yalnız bellek elemanları olabildiği gibi bellek ve kapı elemanları da olabilir. Ardışıl MSI devreler üç ana grupta toplanabilirler: Yazmaçlar (Registers) sayıcılar (Counters) ve rastgele erişimli bellek, RAM ler (**R**andom **A**ccess **M**emory) Bu elemanları ayrı ayrı inceleyeceğiz.

Yazmaçlar, her biri birer bitlik bilgi tutan bellek elemanları ile bilginin işlenmesini denetleyen kapı elemanlarından oluşur.

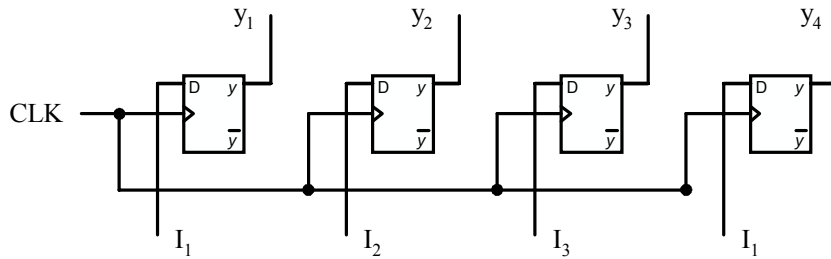
Sayıcılar ard arda gelen saat darbeleriyle daha önce belirlenmiş durumlara sırasıyla geçen ardışıl devrelerdir. Sayıcıları, kapı elemanlarıyla durum geçişleri sağlanan bir yazmaç gibi düşünmek mümkündür. Buna rağmen sayıcılar ayrı bir grup olarak incelenecektir.

Rastgele erişimli bellekler ,RAM (Random Access Memory) ler, hem bellekteki bilginin okunmasını (read) hem de belleğe bir bilginin yazılmasını (write) sağlarlar. Bunlara okuyan ve yazan bellek elemanları da (read-write memory) denir.

Burada sözünü edeceğimiz bellek elemanları bilgisayar tasarımında ve ardışıl devre tasarımında yaygın bir biçimde kullanılırlar.

10.1 Yazmaçlar

En basit bir yazmaç devresi Şekil 10.1 de görüldüğü gibi yalnızca dört D tipi bellek elemanlarından oluşur. Saat girişi geldiğinde girişteki dört bitlik bir bilgi belleğe yazılmış olur ve bir sonraki saat darbesi gelene kadar bu bilgi saklanır. Yazmaçlarda kullanılan bellek elemanları genellikle kenar tetiklemelidirler.



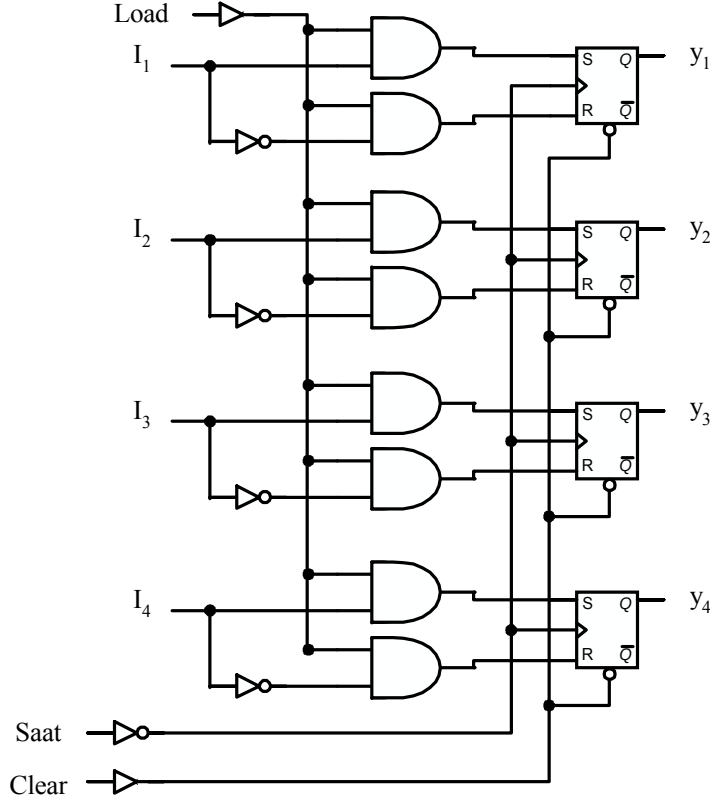
Şekil 10.1 Dört bitlik basit bir yazmaç devresi

10.1.1 Paralel yüklemeli yazmaçlar

Paralel yükleme, bir saat darbesiyle giriş bilgisinin çıkışa aktarılmasıdır. Örneğin dört bitlik bir bilgiyi, paralel yüklemeli bir yazmaçta bir saat darbesiyle, seri toplayıcıda ise ancak dört saat darbesiyle yüklemek mümkündür.

Sayısal sistemlerde sistemin bir ana saati (master clock) vardır ve bütün bellek elemanlarının saat girişine bağlıdır. Oysa bazen belli bellek elemanlarının belli zamanlarda çalıştırılması gereksinimi doğar. Bu nedenle ayrı bir kontrol girişi (load) kullanılır. Şekil 10.2 de load girişli, RS bellek elemanlı, dört bitlik bir paralel yüklemeli

yazmacın lojik devresi verilmiştir. Load girişi saat darbe girişi ile VE lendirilip bellek elemanlarının saat girişlerine bağlanabilirdi. Sayısal sistemlerde saat girişlerinin doğrudan bellek girişlerine bağlanması uygundur, çünkü aksi halde alt sistemlerin birbiriyle senkron çalışmama sorunu çıkabilir. Load girişindeki, tümleyen almayan girişi çıkışı ile aynı olan (**noninverting buffer**) kapı, çıkış yelpazesi (fanout)ni arttırmak içindir. Yani bu kapı 8 VE kapısının girişine yeterli akımı sağlar.



Şekil 10.2 Paralel yüklemeli RS bellek elemanlı yazmaçının lojik devresi

Clear girişi, load girişi gibi, bir buffer kapısından bellek elemanlarının girişine bağlıdır. Giriş 0 olduğunda (bellek elemanlarının girişinde ki küçük yuvarlak bunu göstermek içindir) yazmaçta 0000 bilgisi oluşur. Saat girişi bir tümleyen elemanı üzerinden bellek elemanlarına bağlanmıştır. Tümleyen elemanı, hem gerekli çıkış yelpazesini (fanout) artırır hem de saat darbesinin düşen kenarında tetiklemeyi sağlar.

Şimdi Şekil 10. 3 te doğruluk tablosu verilen paralel yüklemeli yazmacının çalışmasının inceleyelim.

clear	Load	Saat	I ₁	I ₂	I ₃	I ₄	Y ₁	Y ₂	Y ₃	Y ₄
0	x	x	x	x	X	x	0	0	0	0
1	0	x	x	x	X	x	y ₁	y ₂	y ₃	y ₄
1	1	↓	I ₁	I ₂	I ₃	I ₄	I ₁	I ₂	I ₃	I ₄

Şekil 10.3 Paralel yüklemeli yazmaçının doğruluk tablosu

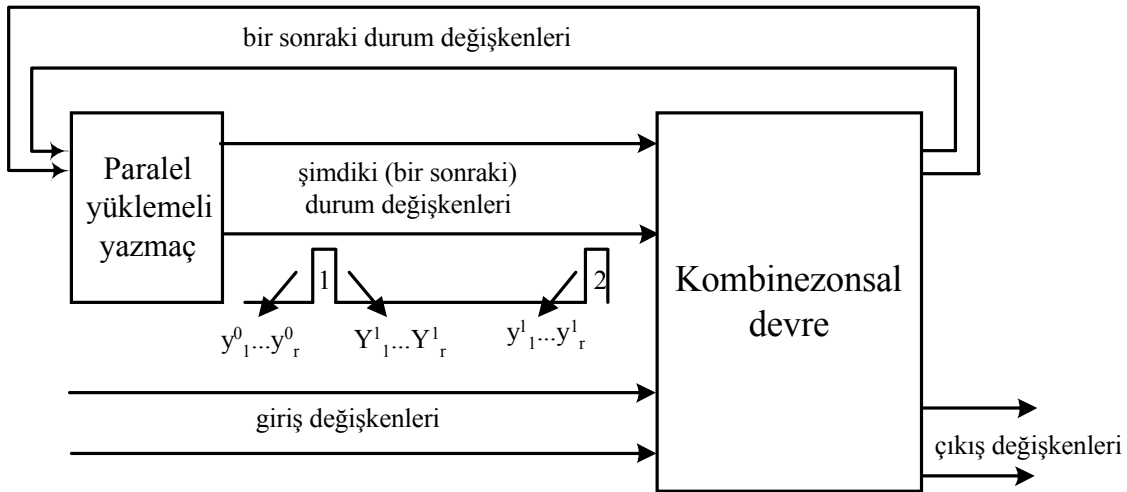
Clear girişi, diğer bütün girişlerden bağımsız olarak, çıkışı sıfırlamaktadır. Clear 1 ve load 0 iken çıkış durumunu korumaktadır. Bu işlem bütün VE kapılarının çıkışlarını dolayısıyla SR bellek elemanlarının girişlerini 00 yapar, SR bellek elemanı tanımı uyarınca da saat darbesi gelse bile çıkışlar değişmez. Clear ve Load 1 iken saat

darbesinin düşen kenarında ise paralel yükleme işlemi olur. Bütün bellek elemanlarının girişleri $S_i = I_i$ ve $R_i = I_i$ bu durumda SR bellek elemanı tanımı gereğince çıkışları $S_i = I_i$ dir.

Yukarıda incelediğimiz paralel yüklemeli yazmaç SR bellek elemanlarıyla gerçekleştirilmiştir, diğer bellek elemanlarıyla da paralel yüklemeli yazmaç gerçekleştirmek mümkündür. Çeşitli firmaların kataloglarında paralel yüklemeli yazmaçların hem lojik devreleri hem de doğruluk tabloları verilmiştir.

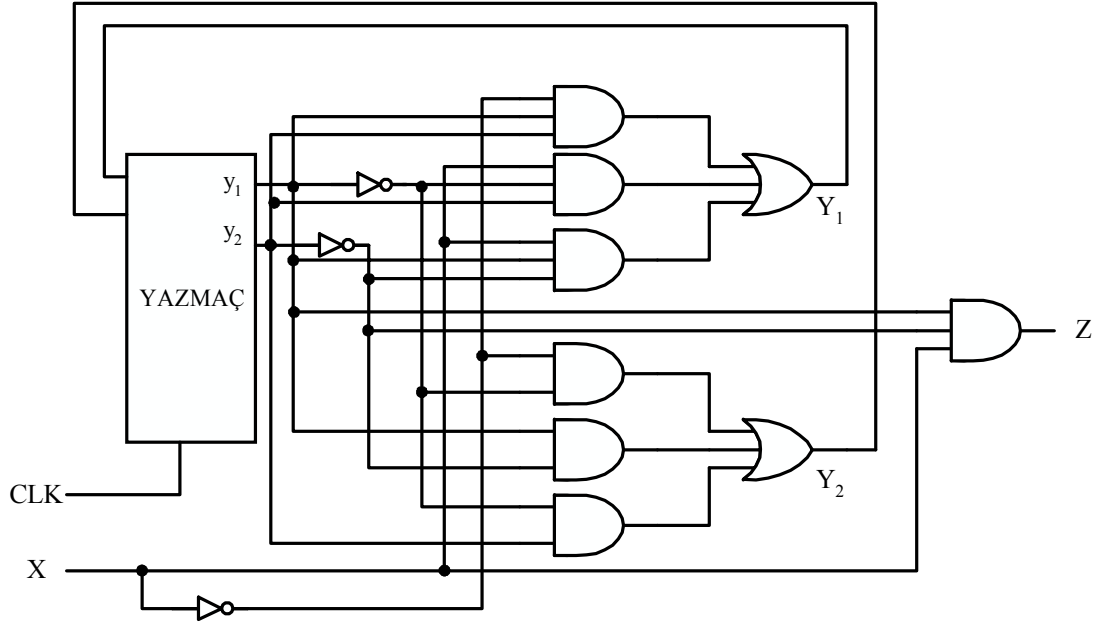
Paralel yüklemeli yazmaçlarla ardışıl devre tasarımı: Paralel yüklemeli yazmaçların iyi bir uygulaması ardışıl devre sentezidir. Daha önce gördüğümüz gibi ardışıl devreler bellek ve kombinezonsal olmak üzere iki parçadan oluşuyordu. Bellek parçası yerine yazmaçlar kullanarak ta sentez Şekil 10.4 te görüldüğü gibi de yapılabilir. Kombinezonsal kısım, kapı elemanlarından oluşturulabileceği gibi ROM larla da gerçekleştirilebilir. Yazmacın çıkışları, saat darbesi öncesi için şimdiki durum değişkenlerine; saat darbesi sonrası için bir sonraki durum değişkenlerine karşı gelir ve kombinezonsal devre girişine bağlanmıştır. Diğer girişler ise ardışıl devrenin bağımsız girişleridir. Kombinezonsal devrenin bir kısım çıkışları, bağımlı bir sonraki durum değişkenlerine karşı gelmektedir ve bir geri besleme ile yazmacın girişlerine bağlanmıştır. Diğer çıkışlar ardışıl devrenin çıkışlarıdır.

Bir MSI yazmaç devresinde 4 bellek elemanının düşünürsek, böyle bir yazmaç ile 16 durumlu bir ardışıl devre gerçekleştirilebilir.



Şekil 10.4 Paralel yüklemeli yazmaçlı ardışıl devrenin genel yapısı

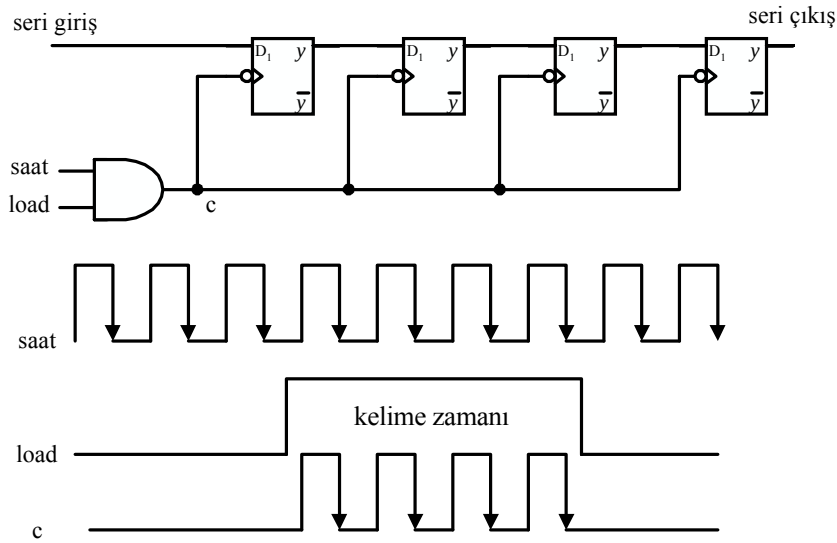
Beşinci bölümdeki dizi detektörünün yazmaç ile gerçekleştirilmesi Şekil 10.5 te verilmiştir.



Şekil 10.5 Dizi detektörünün paralel yüklemeli yazmaç ile gerçekleştirilmesi

10.1.2 Ötelemeli yazmaçlar

içerisindeki bilgiyi her saat darbesi geldiğinde, bir bit öteye kaydıran ve seri girişinden de bir bitlik bilgiyi içine alabilen yazmaçlara ötelemeli yazmaçlar denir. En basit bir ötelemeli yazmaç, çıkışları girişlerine kaskad bağlı D tipi bellek elemanlarından oluşur (Şekil 10.6). Bu devrede her bir saat darbesi geldiğinde bilgi soldan sağa doğru ötelenir. Yazmacın saat darbelerini kontrol eden ikinci bir giriş, load girişi, kullanılabilir. Bunun için saat darbeleri ile load girişi VE elemanı üzerinden saat girişine bağlanır. Load girişi yazmacın bit kapasitesi kadar bir zaman aralığında çalışmasını sağlar. Seri bilgi kelime (word) denilen ard arda bit gruplarından oluşur. Kelimeler 4, 8, 16, 32 bit uzunluklarında olabilirler. Örneğin dört bitlik bir kelime uzunluğu olan yazmacın kapasitesi dört bit yani bir kelimedir. Her seferinde bir kelimelik bilginin ötelenmesi isteniyorsa load girişinin 1 olma uzunluğu, kelime zamanı (word time) . dört saat darbesini kapsayacak kadar olmalıdır (Şekil 10.6).



	serigiriş	D ₁	D ₂	D ₃	D ₄
başlangıç	0	1	0	1	0
1.saat darbesi	1	0	1	0	1
2.saat darbesi	0	1	0	1	0
3.saat darbesi	1	0	1	0	1
4.saat darbesi	0	1	0	1	0

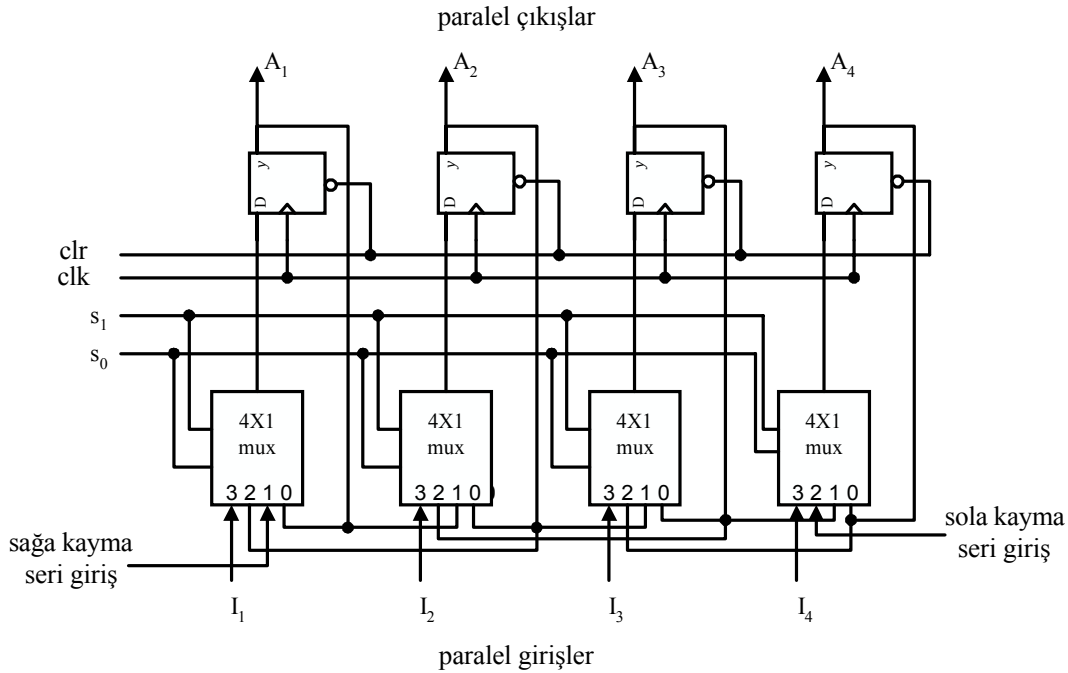
Şekil 10.6 Ötelemeli kaydedici

Bir kelime zamanında saat darbeleri ve giriş çıkışlarda gözlenen işaretler Şekil 10.6 da verilmiştir.

Bilgisayarlarda bilgi seri, Paralel veya ikisinin de olduğu biçimlerde işlenebilir. Seri işlemede her bir saat darbesiyle bir bit işlenirken, paralel işlemede her bir saat darbesiyle bir kelimelik bilgi işlenir. Genellikle bilgisayarlarda paralel işleme söz konusudur. Uzağa bilginin gönderilmesinde ise bilgi, transmisyon ortamının maliyetini azaltmak için, genellikle seri gönderilir.

Çift yönlü seri ve paralel yüklemeli ötelemeli kaydediciler

Bundan önce söz ettiğimiz yazmaçlarda bütün kontrol girişleri birlikte yoktu. Şimdi inceleyeceğimiz de ise sağa, sola öteleme ve paralel yükleme kontrol girişleri olacak. Yazmaçlar sağa sola kaydırma ve paralel yükleme kontrol girişleriyle çok amaçlı olarak kullanılabilirler. Örneğin verinin seriden paralele veya paralelden seriye dönüştürülmesi mümkündür. Şekil 10.7 de dört farklı çalışma modu olan 74194 tipi bir yazmacın lojik devresi verilmiştir.



Şekil 10.7 Kaydırmalı ve paralel yüklemeli yazmaç lojik devresi

Yazmacın s_0 , s_1 girişleriyle kontrol edilen dört çalışma modu vardır:

$s_0s_1 = 00$ ise çıkışta hiçbir değişiklik olmaz,

$s_0s_1 = 01$ ise çıkış sağa ötelenir ve sağa öteleme girişindeki bit A_4 te görülürken A_1 deki bit kaybolur,

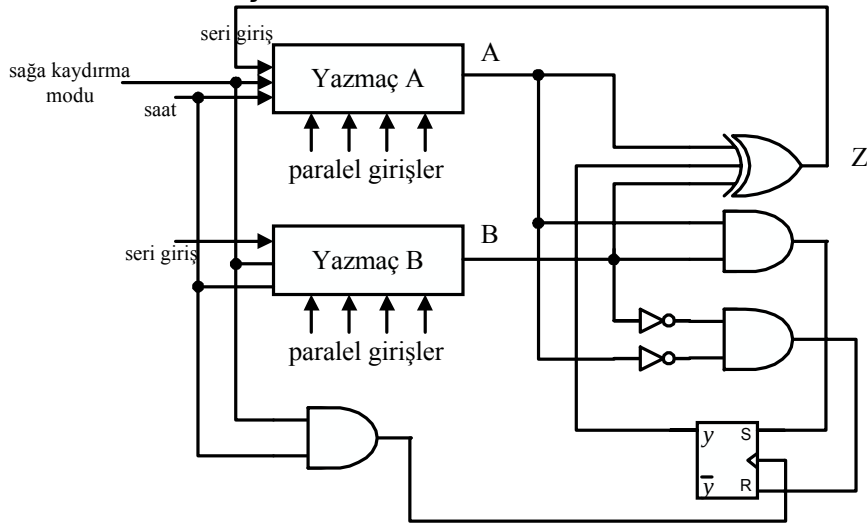
$s_0s_1 = 10$ ise çıkış sola ötelenir ve sola öteleme girişindeki bit A_1 de görülürken A_4 teki bit kaybolur,

$s_0s_1 = 11$ ise paralel girişler çıkışta görülür.

Bu dört çalışma modunu 4X1 lik MUX'ların kontrol girişleriyle sağlamak mümkündür. MUX'ların 0 girişleri bağlı oldukları bellek elemanlarının girişlerine bağlıdır, yani 00 modunda çıkışlar değişmez. MUX'ların 1 girişlerine bir soldaki bellek elemanlarının çıkışları, sağa ötelemeyi sağlamak üzere bağlanmıştır. MUX'ların 2 girişlerine ise bir sağdaki bellek elemanlarının çıkışları, sola ötelemeyi sağlamak için bağlanmıştır. Nihayet MUX'ların 3 girişlerine paralel girişler bağlanmıştır, öyle ki saat darbesi geldiğinde bu paralel girişler çıkışta görülürler.

Böyle bir yazmaçta seri girişlere gelecek 4 bitlik kelimeler dört saat darbesiyle çıkışta paralel olarak görülürler veya çıkıştaki 4 bitlik paralel veri üç saat darbesiyle A_1 veya A_4 ten seri olarak alınabilir.

Bir uygulama olarak yazmaçları, daha önce gördüğümüz (Şekil 7.7) seri toplayıcının giriş ve çıkış verisini toplamada kullanalım. Toplanacak sayılar dört kelime uzunluklu olsun ve birer yazmaçta tutulsunlar. A ve B Yazmaçlarına toplanacak sayılar paralel girilebilir. Çıkıştaki toplam sayıyı da kullandığımız A yazmaçsına seri olarak kaydedelim. Bu durumda çıkışın yazıldığı A yazmacının, üzerindeki toplanacak sayı silinmiş olur. Lojik devre Şekil 10.8 de verilmiştir.



Şekil 10.8 Kaydedicili seri toplayıcı

Paralel yükleme modunda toplayacağımız dört bitlik A ve B sayılarının A ve B yazmaçlarına yüklediğimizi varsayalım. SR bellek elemanının başlangıç durumu 0 olsun A ve B sayılarının en az ağırlıklı bitlerinin toplamı z çıkışında ve dolayısıyla A yazmacının seri girişinde görülecektir. Yazmaçlar öteleme modunda iken, saat darbesi geldiğinde, en az ağırlıklı bitlerin toplamı A yazmacının en sol bitine girer. A ve B sayılarının ikinci ağırlıklı bitleri ise seri toplayıcının girişlerine de gelir. En az ağırlıklı bitlerden gelen elde ise SR bellek elemanının çıkışı y de olacaktır. Böylece z çıkışı, ikinci bitler ile birinci bitlerin eldesinin toplamı olur ve A yazmacının seri girişine gelir. Bu işlem dört saat darbesi boyunca devam ettiğinde A ve B sayılarının toplamı A yazmacında görülür. Bu arada B yazmacına üçüncü bir sayı girilmiş olsun. Bu halde A + B nin üçüncü sayı ile toplamı, yeni bir dört saat darbelik sürede, A yazmacında elde edilebilir.

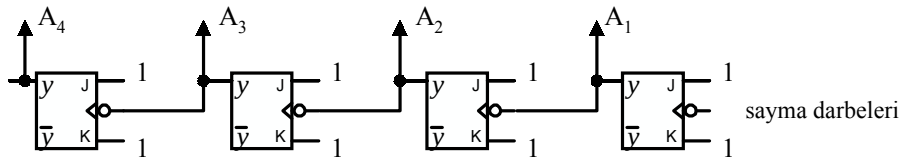
10.2 Sayıcılar (Counters)

MSI sayıcılar iki ana grupta toplanabilir: **asenkron sayıcılar (ripple counters) ve senkron sayıcılar**. Asenkron sayıcılarda birinci bellek elemanı hariç diğer bellek elemanlarının saat girişleri diğer bellek elemanlarının çıkışlarından beslenir. Senkron sayıcılarda ise sayıcıdaki bütün bellek elemanlarının saat girişleri aynı bir saat kaynağından doğrudan beslenir. Asenkron sayıcılarda bellek elemanlarının durum değişimi diğer bellek elemanlarının durum değişimiyle ancak mümkün olur. Senkron sayıcılarda bütün bellek elemanlarının durum değişimi aynı anda olur.

Sayma biçimi olarak sayıcılar **ikili (binary) ve onlu (binary coded decimal BCD)** olarak iki türdür. ikili sayıcılarda sayma iki tabanıdır ve durum sayısı kadar sayma işlemi, ardarda saat darbeleriyle yapılır. Örneğin 16 sayıcısı, 16 durumlu olup 0 dan 15 e kadar sayar sonra tekrar 0 a gelir. ikili kodlu onlu BCD sayıcılarda ise sayıcı iki tabanında 0 dan 9 a kadar iki tabanında her bir saat darbesi geldiğinde sayma yapar, 9 dan sonra 0 a dönüşür. Bu iki türü hem asenkron hem de senkron sayıcılarda göreceğiz.

10.2.1 İkili Asenkron Sayıcılar (Binary ripple counter)

Genellikle Bir MSI sayıcı içinde dört bellek elemanı bulunur. ikili asenkron sayıcılarda ise T ve JK tipi bellek elemanları vardır. ikili asenkron bir MSI sayıcının lojik devresi Şekil 10.9 da verilmiştir. Sayma işlemi yaptıran darbeler en az ağırlıklı biti temsil eden bellek elemanının girişine bağlanmıştır. Birinci bellek elemanının çıkışı ikinci bellek elemanının saat girişine girer. Bütün bellek elemanlarının girişleri 1 (Eğer JK kullanılıyorsa $J=1, K=1$; T bellek elemanı kullanılıyorsa $T=1$) de tutulur. Bilindiği gibi bu girişler altında JK ve T bellek elemanları bulundukları durumların zıtlarına dönüşürler (toggle). Bu yapı üçüncü ve dördüncü bellek elemanlarında tekrarlanır. Saat girişlerindeki yuvarlak çember, saat işaretinin düşen kenarında tetiklemenin olduğunu gösterir.



Şekil 10.9 Asenkron ikili bir sayıcı devresi

Başlangıç durumunda $A_1 A_2 A_3 A_4$ çıkışlarının hepsi 0 olsun. Birinci bellek elemanının saat girişine gelen darbenin düşen kenarında $J = K = 1$ olduğundan A_1 çıkışı $0 \rightarrow 1$ e geçer. Bu geçiş ikinci bellek elemanına etki etmez, çünkü $1 \rightarrow 0$ geçişi gereklidir. Bundan dolayı $A_2 A_3 A_4$ çıkışları değişmez. ikinci saat darbesinin düşen kenarında ise A_1 çıkışı $J = K = 1$ olduğundan $1 \rightarrow 0$ e geçerken ikinci bellek elemanı tetiklenir ve A_2 çıkışı $0 \rightarrow 1$ e geçer. Üçüncü bellek elemanı etkilenmemiştir. $A_1 A_2 A_3 A_4 = 0010$ olup on tabanındaki 2 dir. Üçüncü saat darbesinin düşen kenarında A_1 çıkışı $0 \rightarrow 1$ e geçer, ikinci üçüncü ve dördüncü bellek elemanları etkilenmez. $A_1 A_2 A_3 A_4 = 0011$ olup on tabanındaki 3 dür. Dördüncü sayma saat darbesinin düşen kenarında iki ve üçüncü bellek elemanları etkilenir çünkü A_1 çıkışı $J = K = 1$ olduğundan $1 \rightarrow 0$ e geçerken ikinci bellek elemanı tetiklenir ve A_2 çıkışı $1 \rightarrow 0$ e geçer. Üçüncü bellek elemanı etkilenir ve A_3 çıkışı $0 \rightarrow 1$ e geçer. $A_1 A_2 A_3 A_4 = 0100$ olup on tabanındaki 4 dür. Bu işleme devam edildiğinde görülür ki bir bellek elemanın $0 \rightarrow 1$ geçişi, ancak kendinden önceki bellek elemanlarının tümünün 1 olması halinde mümkündür, zaten iki tabanında saymada böyle olmaktadır. $A_1 A_2 A_3 A_4 =$

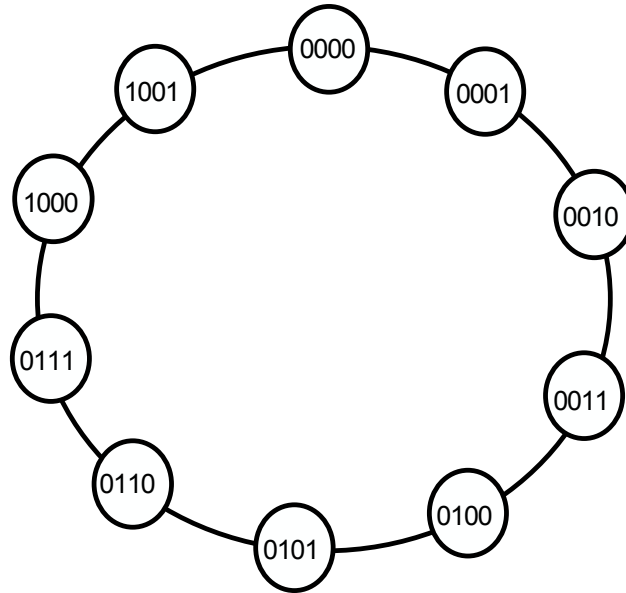
1111 yani 15 olduktan sonra 16. sayma darbesinin düşen kenarında $A_1A_2A_3A_4 = 0000$ olur. A_4 çıkışı, $J = K = 1$ olduğundan, $1 \rightarrow 0$ e geçerken bir sonraki MSI sayıcının birinci bellek elemanını tetikleyerek, saymanın daha büyük sayılara devam ettirilmesini sağlar.

Yukarıda anlattığımız sayıcının geriye sayması isteniyorsa
i. Yükselen kenarda tetiklenen bellek elemanları kullanmak,
ii. Bellek elemanlarının tümleyen çıkışlarını sayıcı çıkışı almak,
iii. Çıkışları Q lardan almak fakat 2,3,ve 4. bellek elemanlarının saat girişlerini bir önceki bellek elemanlarının Q' çıkışlarından beslemek,
yeterli olur.

Bu üç yolu, ilgili lojik devreleri analiz ederek, deneyebilirsiniz.

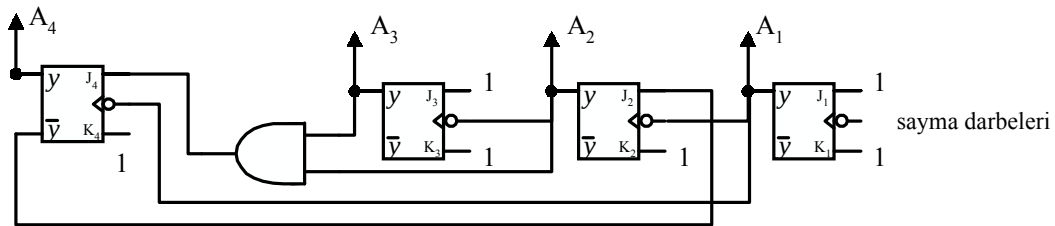
10.2.2 İkili kodlanmış onlu asenkron sayıcı (BCD binary coded decimal ripple counter)

Asenkron sayıcıların sentezi için adım adım takip edilebilecek genelleştirilmiş bir yöntem yoktur. Senkron sayıcılarda ise daha önce adım adım verdiğimiz ardışıl devre sentezi yöntemi kullanılabilir. Asenkron sayıcılar sezgisel olarak sentezi yapıldığından, ikili kodlanmış asenkron sayıcının önce lojik devresini vereceğiz sonra istenen işi yapıp yapmadığını analiz edeceğiz. ikili kodlanmış asenkron sayıcının durum diyagramı Şekil 10.10 da verilmiştir.

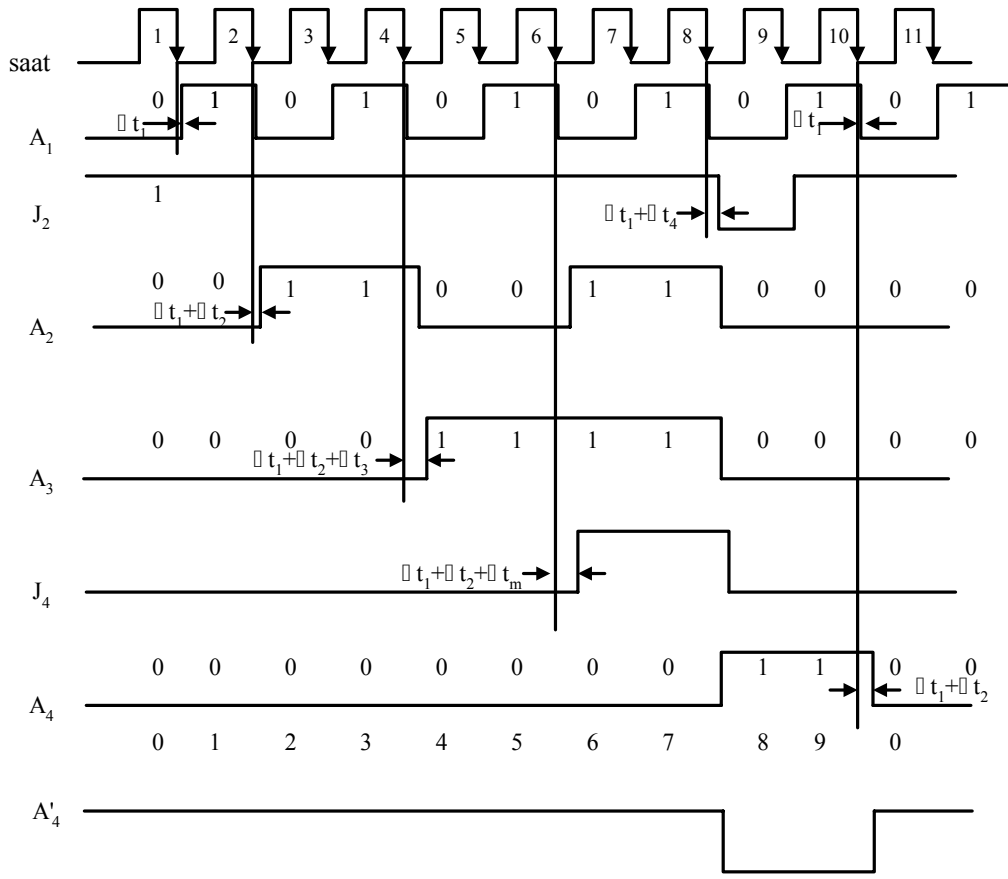


Şekil 10.10 ikili kodlamalı decimal asenkron sayıcı

Şekil 10.10 da durum diyagramı verilen ikili kodlamalı onlu asenkron sayıcının lojik devresi ve zaman diyagramı sırasıyla Şekil 10.11 ve Şekil 10.12 de verilmiştir.



Şekil 10.11 ikili kodlanmış decimal asenkron sayıcının lojik devresi



Şekil 10.12 ikili kodlanmış decimal asenkron sayıcının zaman diyagramı

Saat darbelerinin düşen kenarları ile 1., 2., 3. ve 4. bellek elemanlarının çıkışları arasındaki gecikme sırasıyla t_1 , t_2 , t_3 , t_4 olsun. VE kapısının giriş çıkış gecikmesi ise t_m olsun.

Saat darbesinin düşen kenarına göre A_1 deki değişimin gecikmesi t_1 dir.

Saat darbesinin düşen kenarına göre J_2 deki değişimin gecikmesi $t_1 + t_4$ dir.

Saat darbesinin düşen kenarına göre A_2 ün tetiklenme anı t_1 , A_2 deki değişimin gecikmesi ise $t_1 + t_2$ dir.

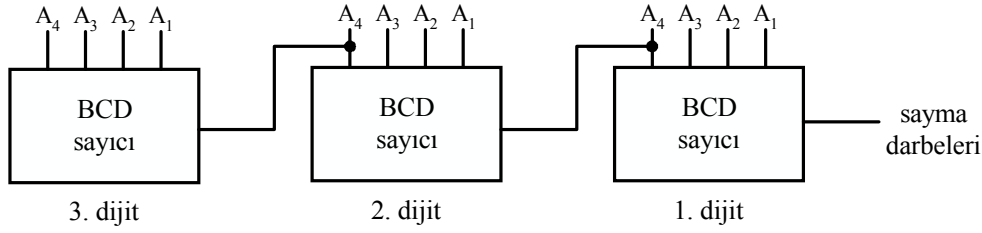
Saat darbesinin düşen kenarına göre A_3 ün tetiklenme anı $t_1 + t_2$, A_3 deki değişimin gecikmesi ise $t_1 + t_2 + t_3$ dir.

Saat darbesinin düşen kenarına göre J_4 teki değişimin gecikmesi ise $t_1 + t_2 + t_m$ dir.

Saat darbesinin düşen kenarına göre A_4 ün tetiklenme anı t_1 , A_4 deki değişimin gecikmesi ise $t_1 + t_2$ dir.

Buna göre Şekil 10.12 de görüldüğü gibi 9 dan sonra 0 olarak sayma yapılmaktadır.

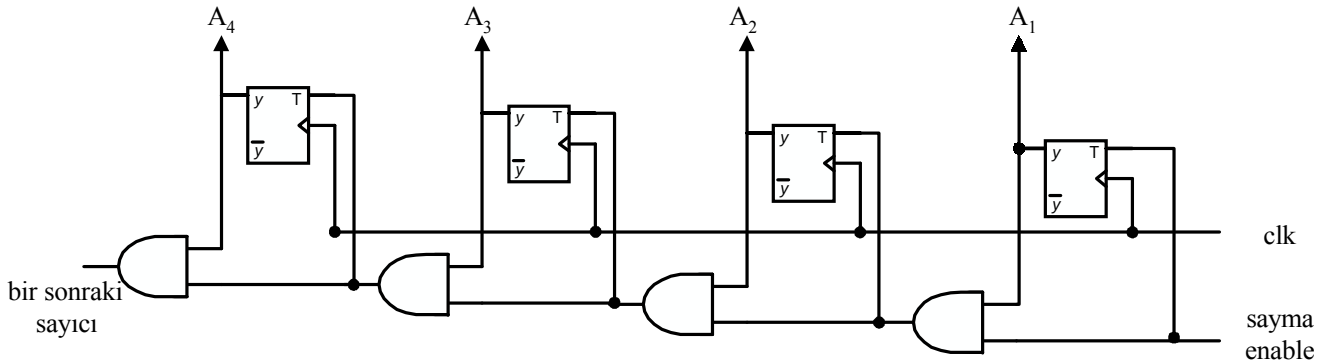
Yukarıda anlatılan sayıcılar yan yana kaskad bağlanarak daha yüksek dijitalere kadar sayma işlemi yapılabilir. Şekil 10.13 te Üç tane kaskad bağlı sayıcı ile 000 dan 999 e kadar saymak mümkün olur . En az ağırlıklı sayıcıya 10 darbe geldiğinde bir sonraki dijitin en az ağırlıklı biti 1 olur, ilk iki sayıcı 99 gösterirken 100 cü darbe geldiğinde de üçüncü sayıcının en az ağırlıklı biti 1 olur ve böyle devam ederek 999 a kadar sayma mümkün olur.



Şekil 10.13 Kaskad bağlı üç dijital bir sayıcı

10.2.3 İkili senkron sayıcılar

Senkron sayıcılarda saat darbeleri, bütün bellek elemanlarının saat girişlerine aynı anda gelirler. Bu nedenle sentezi klasik ardışıl devre sentezi gibi yapılabilir. İkili sayıcılarda sayma her hangi bir bellek elemanının, kendinden önceki bellek elemanlarının hepsi 1 iken, konum değiştirmesi esasına dayanmaktadır. Bu özelliği kullanarak Şekil 10.14 te görüldüğü gibi hücresel yapıda sentez kolayca yapılabilir. Sayma enable 0 iken $T=0$ olacağından sayma işlemi saat darbeleri gelse bile yapılmaz. Sayma enable 1 iken bir bellek elemanının durum değiştirmesi, ancak kendinden önceki bellek elemanların 1 durumunda olması VE kapılarıyla sağlandığında, mümkündür. Aksi halde bu bellek elemanının girişi $T=0$ olacağından durumunu değiştirmez. Bellek elemanı olarak her ne kadar T seçilmişse de JK bellek elemanlarıyla da aynı devre gerçekleştirilebilir, öyle ki VE kapıları çıkışları J ve K girişlerine bağlansın. Bilindiği gibi JK bellek elemanın 00 ve 11 girişleri için davranışı T bellek elemanının 0 ve 1 girişleri için olan davranışıyla aynıdır. Saat girişi yükselen veya düşen kenarın her hangi birinde tetiklenebilir, asenkron sayıcılarda olduğu gibi düşen kenarda tetiklenme zorunluluğu yoktur.

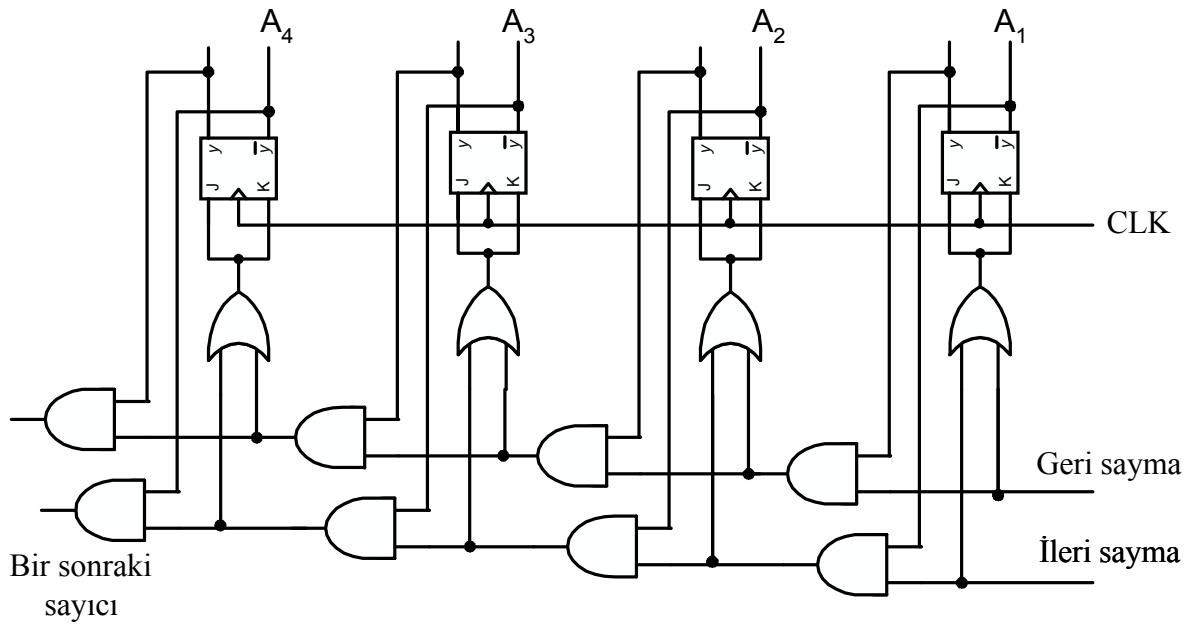


Şekil 10.14 İkili senkron sayıcının lojik devresi

İleri sayıcı gibi geriye sayan sayıcıların yapısı birbirine benzerdir. Geriye sayıcıda bir bellek elemanının durum değiştirmesi, kendinden önceki bütün bellek elemanlarının 0 olması esasına dayanmaktadır. Örneğin 0100 durumunda olan sayıcının geriye sayması halinde A_3, A_2, A_1 bellek elemanlarının durum değiştirmesi gerekir ki buda 0011 sonucunu verir. Yani geriye sayma yapılmaktadır. Geriye saymada bellek elemanlarını girişlerindeki VE kapıları, bir önceki bellek elemanlarının Q çıkışları yerine Q' çıkışlarıyla beslenir.

Hem ileri hem de geri sayma işlemi yapan bir senkron sayıcı lojik devresi Şekil 10.15 te verilmiştir. Aynı devre VEYA kapı çıkışlarını, JK bellek elemanları yerine, T bellek elemanlarının girişlerine bağlayarak ta gerçekleştirilebilir. İleri ve geri sayma girişleri

birlikte 0 ise sayıcı durumunu korur. İleri ve geri sayma girişleri sırasıyla 1,0 ise sayıcı ileri sayar; girişler sırasıyla 0,1 ise sayıcı geriye sayar.



Şekil 10.15 ileri geri sayabilen senkron sayıcı lojik devresi

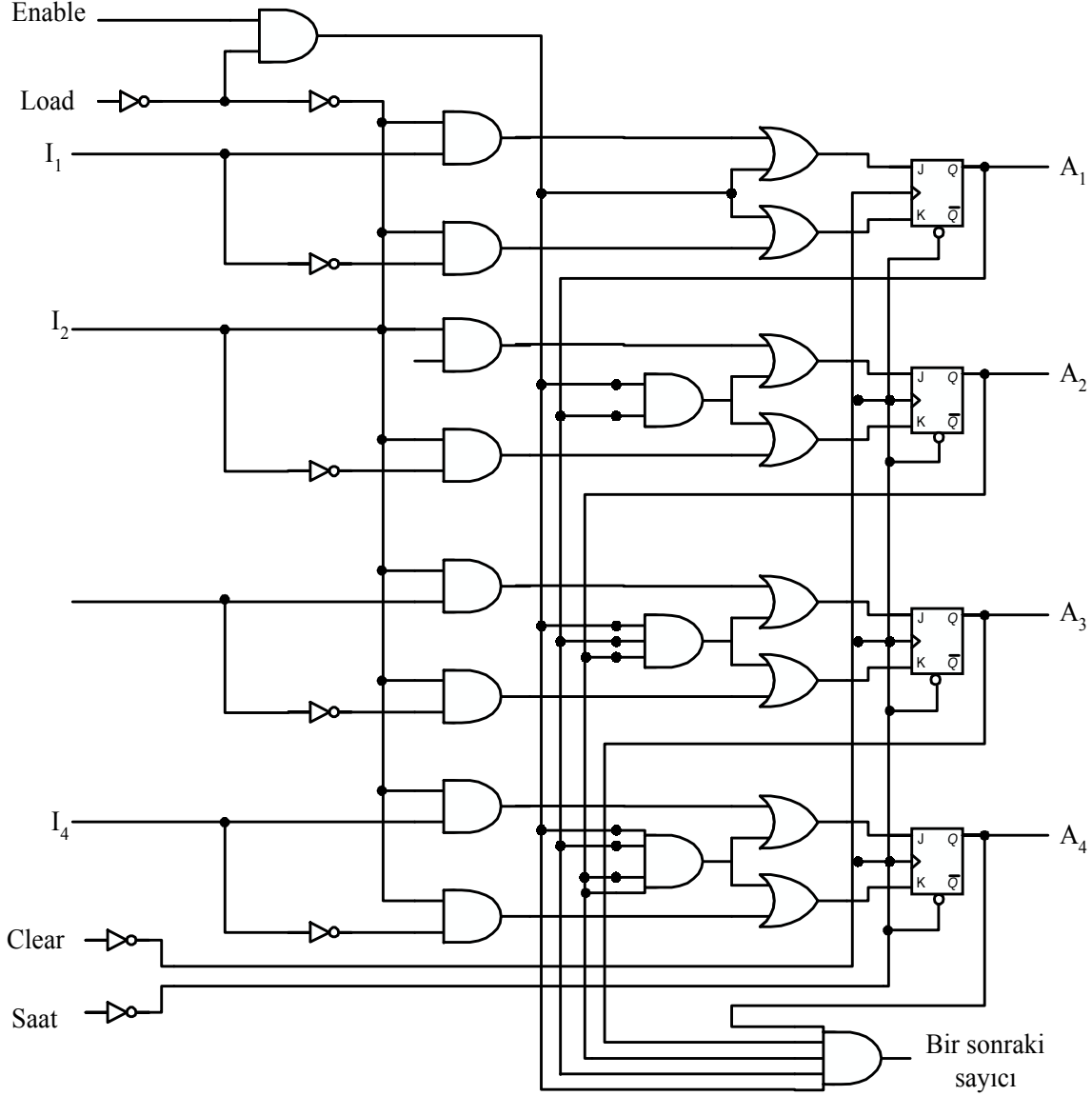
10.2.4 Paralel yüklemeli ikili senkron sayıcılar

ikili kodlanmış desimal senkron sayıcıların lojik devreleri, daha önce gördüğümüz ardışıl devre sentezi yöntemiyle kolayca yapılabilir. Bu nedenle burada ayrıca incelenmeyecektir. Uygulamada kullanılan MSI senkron sayıcıların, paralel yükleme özelliğiyle, istenen sayıdan başlayarak istenen sayıya kadar saymaları kolayca gerçekleştirilebilir.

Sayma, load (yükleme), clear ve saat olmak üzere dört kontrol, dört bitlik paralel giriş, dört bitlik sayı çıkışı, ve bir bitlik elde çıkışı olan 74161 benzeri bir senkron sayıcı devresi Şekil 10.17 de verilmiştir. Kontrol girişlerine göre devrenin davranışına ilişkin doğruluk tablosu aşağıdaki gibidir:

Clear	saat	load	sayma	çıkış
0	X	X	X	hepsi 0
1	X	0	0	değişiklik yok
1	↓	1	X	paralel yükleme
1	↓	0	1	sayma

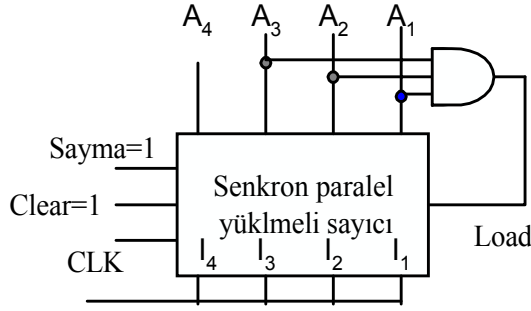
Clear 0 iken diğer kontrol girişleri ve paralel girişler ne olursa olsun, çıkış dört bitte de 0 dır. Load ve sayma 0 ise çıkışta hiçbir değişme gözlenmez, çünkü bellek elemanlarının JK girişleri 00 olmuştur. Clear ve load 1 iken saat darbesinin yükselen kenarında, $J = A$, $K = A'$ olduğundan, çıkışlarda A olacak , yani paralel girişler aynen çıkışta görülecektir. Clear 1 ve load 0 ve sayma 1 iken, bellek elemanlarının JK girişleri 11 olacak yani sayma yapılacaktır.



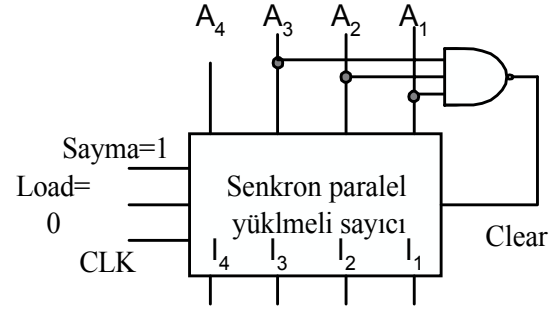
Şekil 10.16 Paralel yüklemeli senkron sayıcının lojik devresi

Yukarıda lojik diyagramı verilen MIS sayıcı ile istenen sırada sayma işlemi yapmak mümkündür. Örneğin modulo N sayıcılar yani 0 dan N-1 e kadar sayan, N inci darbe geldiğinde ise tekrar 0 konumuna dönen sayıcılar tasarlanabilir. Dört bitlik bir Modulo 16 sayıcı 0 dan 15 e kadar sayar 16. darbe geldiğinde 0 a dönüp yeni gelecek darbelerle saymaya devam eder. Daha da geliştirilerek, istenen bir sayıdan başlayarak N-1 darbe ile sayma işleri yapıp N. darbe ile başlangıç sayısına dönen modulo-N sayıcılar da tasarlanabilir. Şekil 10.17 da sırasıyla 0-7, alternatif 0-7, 8-15 ve 3-10 sayan modulo-8 sayıcıların lojik devreleri verilmiştir. Şekil 10.17 a da 0-7 sayan sayıcıda 7 den sonra 0 a

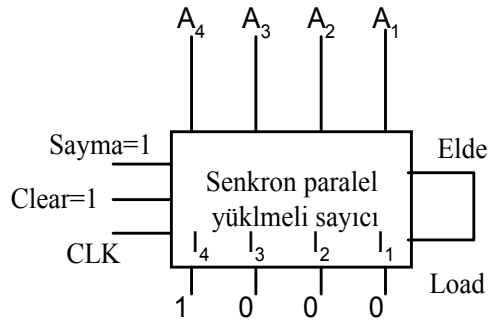
dönüş load girişiyle, paralel girişler üzerinden sağlanmıştır. Şekil 10.17 b de ise aynı sayıcının 7 den 0 a dönüşü clear girişi ile paralel yükleme kullanılmaksızın sağlanmıştır. Şekil 10.17 c deki 8-15 sayıcıda 15 ten 0 a geçiş, son varılan sayı dört bit için maksimum olan 15 olduğundan, elde çıkışının load girişine bağlanması ve 8 in paralel yüklenmesiyle sağlanmıştır. Şekil 10.17 d deki 3-10 sayıcıda 10 dan 3 e geçiş, çıkışta varılması istenen sayı 8 ulaşıldığında, load girişi uyarılarak 3 başlangıcının paralel yüklenmesiyle sağlanmıştır. Paralel yüklemeli ikili kodlanmış onlu 0-9 sayan incelediğimiz sayıcılarla tasarlayabilirsiniz.



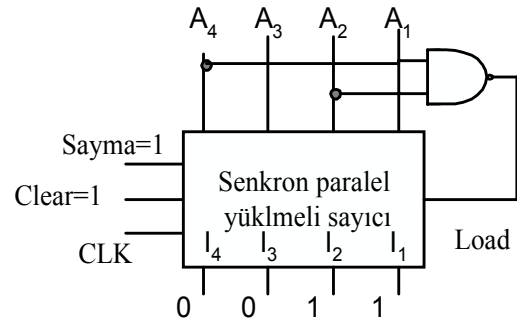
a) 0-7 sayıcı



b) Alternatif 0-7 sayıcı



c) 8-15 sayıcı



d) 3-10 sayıcı

Şekil 10.17 Çeşitli Modulo sayıcıların senkron sayıcılarla gerçekleştirilmesi

Yukarıda incelediğimiz senkron sayıcı genel olarak ardışıl devre sentezinde de bir alternatif olarak kullanılabilir. Öyle ki ard arda gelmeyen durumlar paralel girişlerde oluşturulacak bir sonraki durum fonksiyonları ve load girişleriyle sağlanabilir. Ard arda gelen durumlar ise sayma komutu ile sağlanabilir.

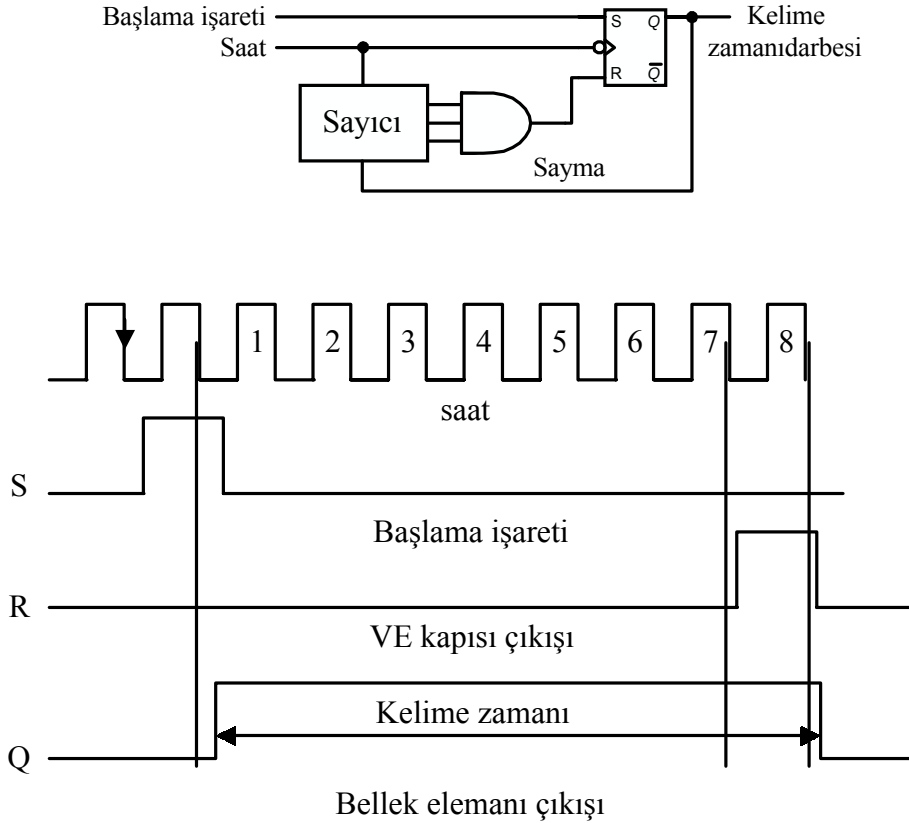
10.3 Sayıcıların uygulaması olarak, zamanlama dizileri üreten devreler

Zaman dizileri özellikle seri çalışan sayısal sistemlerde, bir kelime uzunluğunca çalışmayı sağlayan darbeler olabilirler. Ayrıca sayısal sistemlerde genel olarak görülen zaman dizileri vardır. Sayısal sistemlerdeki **alt devrelerin veya elemanlarının** çalışma aralıklarını kontrol eden kontrol birimleri vardır. Bu birimler ana saatin ürettiği darbelerin belli zaman aralıklarında belli alt devrelerde etkin olmasını sağlarlar. Böylece bir işleve ilişkin işlemler, belli sırada belli alt devrelerin çalışmasıyla tamamlanmış olur. Kontrol

biriminin işlevini yerine getirebilmesi için yazmaçlar veya sayıcılar kullanılabilir. Önce seri çalışan sistemlere ilişkin kelime-zamanı darbesini üreten devreyi inceleyeceğiz. Sonra seri veya paralel çalışan sistemlerde, sözünü ettiğimiz kontrol işlevinin yapılması için gerekli olan zaman dizilerini üreten devreleri inceleyeceğiz.

10.3.1 Kelime-zamanı darbesinin üretilmesi

Kelime zamanının ne olduğunu ötelemeli yazmaçları da tanıtmıştık. Kelime zamanı uzunluklu bir darbenin oluşturulabilmesi için bir başlama işareti geldikten sonra otomatik olarak üretilebilmeli. Şekil 10.18 de 8 bitlik kelime uzunluklu bir darbe üreten devrenin, bir sayıcı, bir bellek elemanı ve bir VE kapısı kullanılarak nasıl gerçekleştirildiği gösterilmiştir. Başlangıçta sayıcı 000, R girişi 0 ve başlama işareti 1 iken ilk saat darbesi geldiğinde, bellek elemanın çıkışı 1 olur ve sayıcı uyarılır sayma işlemi başlar. 8. saat darbesine kadar $S = R = 0$ olacağından bellek elemanın çıkışı 1 de kalır. 8. saat darbesi geldiğinde sayıcı 111 olacağından $R = 1$ olur, $S = 0$ olduğundan bellek çıkışı 0 olacak, sayma duracaktır. Bellek elemanının çıkışı 8 saat darbelerlik süre içinde 1 olarak istenen kelime zamanını verir. ilgili zaman diyagramları şeklin altında verilmiştir.



Şekil 10.18 Seri çalışan sayısal sistemlerde 8 bitlik kelime zamanlı darbe üretimi

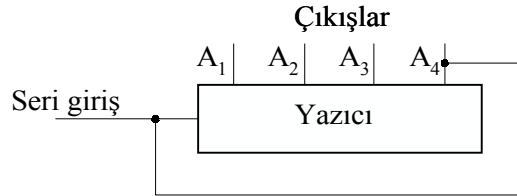
10.3.2 Sayısal sistemlerde zaman işaretleri

Paralel çalışan sistemlerde seri sistemlerde olduğu gibi kelime zamanı devresi üretmek gibi bir sorun olmaz. Çünkü her saat darbesi bir kelimelik bilgiyi aynı anda işleyecektir. Ama her iki sistemde de yapılacak işleme bağlı olarak alt sistemlerin çalışmasını kontrol eden zaman işaretlerine (timing signal) gereksinim vardır. Bu amaçla Şekil 10.19 da verilen yazmaçlı veya sayıcı devreler kullanılabilir.

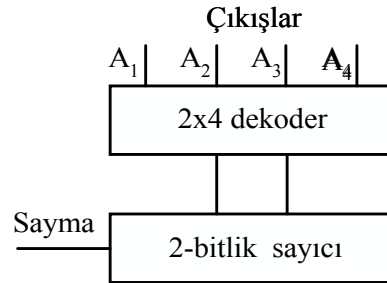
Yazmaç devrenin başlangıç durumu 0001, 0010, 0100, 1000 durumlarından biri olsun. Her bir saat darbesi geldiğinde yazmaçtaki bilgi bir sağa öteleneceği yerde, son bit seri

girişe bağlı olduğundan bu işlem dairesel (ring counter) olarak devam edecektir. Yazmacın çıkışları, biri dışında 0 olur. Yazmacın her bir çıkışının bir alt devreyi kontrol ettiğini düşündüğümüzde, görülür ki bir alt devre çalışırken diğerleri çalışmayacaktır. Bu çözümde 0000 durumu hiç gözlenmeyecektir, ancak ek devrelerle bu sağlanabilir.

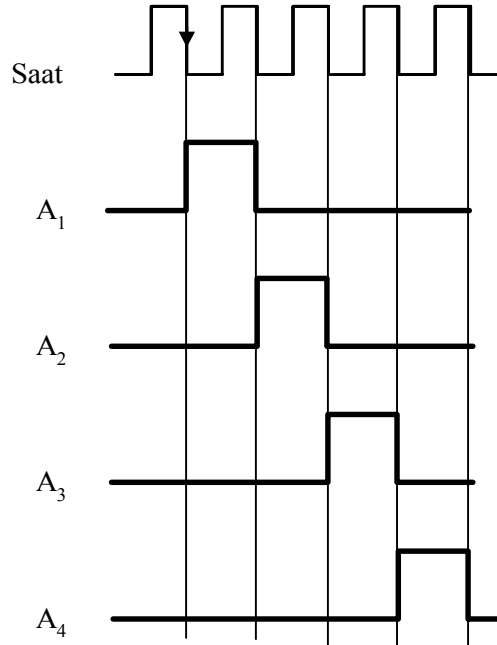
Dört bitlik bir yazmacı kullanarak gerçekleştirdiğimiz bu devrenin işlevini, iki bitlik bir sayıcı ve 2X4 lük bir kod çözücü (dekoder) ile de gerçekleyebiliriz. Kod çözücü, sayıcıdaki sayıya bağlı olarak, yalnızca bir çıkışında 1 diğer çıkışlarında 0 verir. Her iki devreye ilişkin çıkışların zaman diyagramları şekilde gösterilmiştir.



4 lü Yazıcılı zaman işaretüreticisi



4 lü Yazıcılı zaman işaretüreticisi



4 lü zaman işareti

Şekil 10.19 4 bitlik zaman dizileri üreten devreler ve zaman diyagramları

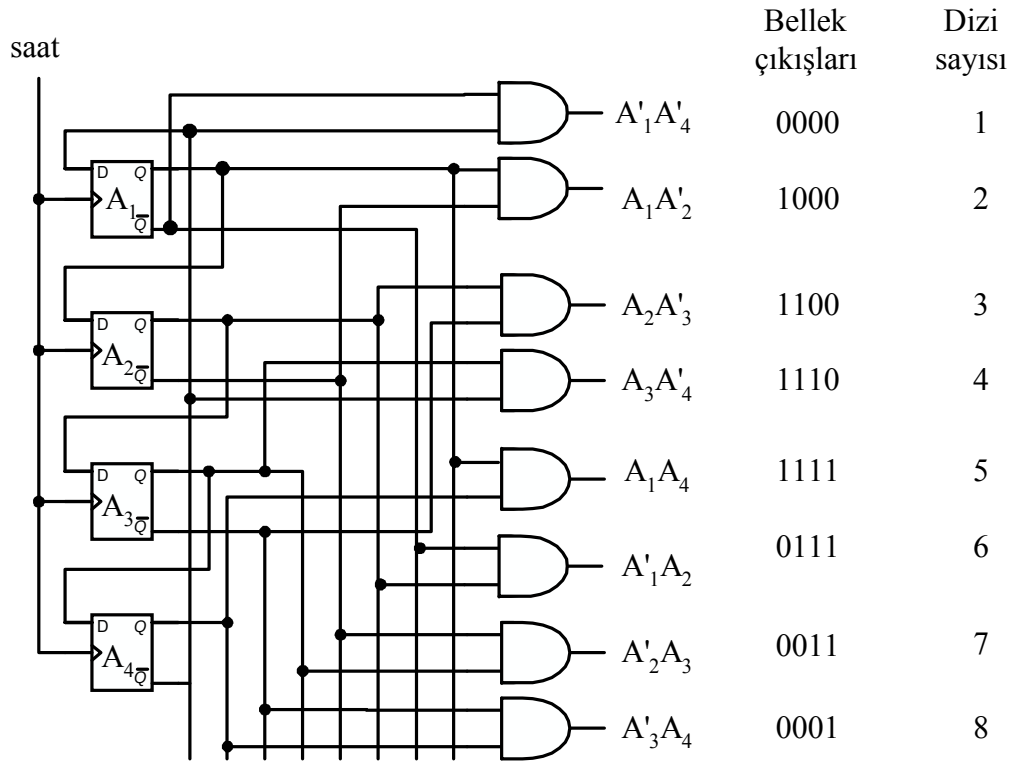
2^n alt devre için kontrol girişleri (zaman işaretleri) üretmek istensin. Yukarıda verilen iki devrenin eleman gereksinimlerini genel olarak şöyle özetleyebiliriz:

1. 2^n bitlik bir ötelemeli yazmaç
2. n -bitlik bir sayıcı ile $n \times 2^n$ lik bir kod çözücüye (n girişli 2^n tane VE kapısı) gereksinim vardır.

Kontrol edilecek alt devre sayısı arttığında karmaşıklık hızla artmaktadır. Bu nedenle daha az elemanla bu işlevi yerine getirebilecek Johnson sayıcılar kullanılır.

10.3.3 Johnson sayıcılar ve 8 li zaman işareti üreticileri

Şekil 10.20 de bir Johnson sayıcı devresi verilmiştir. iki kısımdan oluşmaktadır: sayıcı ve bu sayıcıdan zaman işareti üreten kod çözücü. Sayıcı çıkışları öyledir ki ardarda gelen saat darbeleri sonucunda elde edilen çıkışlarda yalnızca bir değişken değişmekte diğerleri değişmemektedir. ilk yazmaçlı devreden farklı olarak burada başlangıç durumu 0000 alınmış ve 4. bitin tümleyen çıkışı seri girişe bağlanmıştır. Başlangıç ve ardarda gelen sekiz darbenin ilk üçünde A_4 çıkışı 0 diğer dördünde ise 1 dir. Diğer bellek elemanlarının çıkışları bir sonrakine bağlıdır. Bu sayıcıyı bildiğimiz sentez yöntemi ile de gerçekleştirebilirdik.



Şekil 10.20 Johnson sayıcı devresi ile 8 bitlik kelime-zaman darbesi üretilmesi

Çıkışta ki kod çözücü; sayıcı çıkışı bağımsız girişler, çıkışları da zaman işaretleri olan kombinezon devreler düşünülerek gerçekleştirilebilir. 8 VE kapısının çıkışlarından yalnız 1 tanesi 1 iken diğer hepsi 0 dir. O nedenle 8 alt devreyi ayrı ayrı kontrol etmek mümkündür.

Görüldüğü gibi Johnson sayıcıda 4 bellek ve iki-girişli 8 VE kapısı ile yapılan zamanlama devresi, yazmaç ile yapılsaydı 8 bellek elemanı, sayıcıyla yapılsaydı 3 bellek elemanı ve üç girişli 8 VE kapısı gerekecekti.

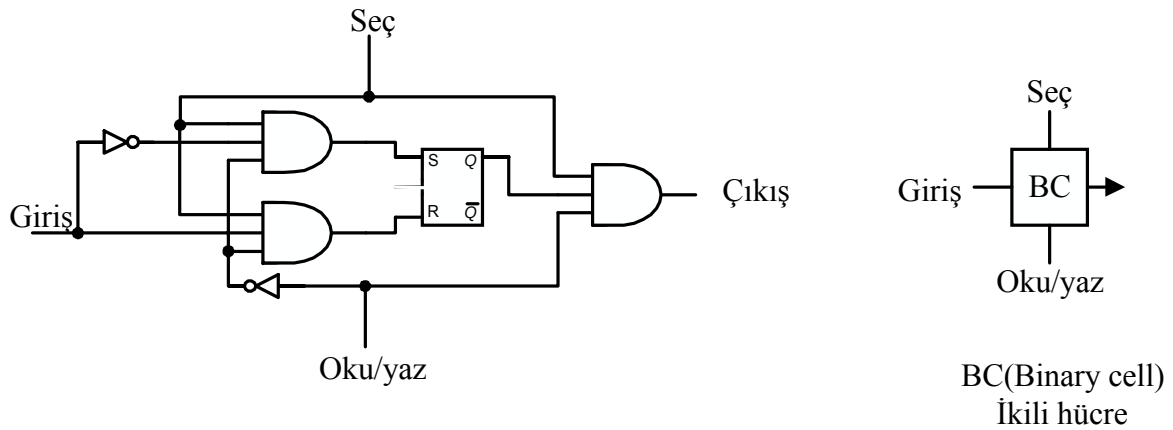
10.4 Bellek birimi

Bir sisteme ilişkin bellek birimi, belli uzunluktaki (kelime uzunluğu) verinin belli adreslere (belli yazmaçlara) kaydedilmesi veya daha önce kaydedilmiş verinin okunması işlevini görürler. Bellek elemanları manyetik core, disk veya ardışıl devre olarak oluşturulabilirler. Biz burada ardışıl olan, okuma yazmalı bellek (Read write memory), rastgele erişimli bellek, RAM (**R**andom **A**ccess **M**emory), bellek birimini inceleyeceğiz.

Uygulamada kullanılan bellek birimleri 8 den 64 bite kadar çeşitli kelime uzunlukları olan binlerce veriyi saklayabilme özelliğine sahiptirler. Burada önce bir hücre tanıtacağız, sonra göstereceğiz ki bu hücrelerin uygun bağlanmasıyla istenilen kapasitede bellek birimleri oluşturulabilir.

İkili hücre (binary cell): Şekil 10.21 de verildiği gibi üç girişli bir çıkışlı bir ardışıl devredir. Girişler giriş (input), oku-yaz (read-write) ve seç (select) tir. Giriş belleğe yazılacak veya bellekten okunacak bir bitlik veridir. Oku-yaz aldığı 0 veya 1 değerine göre bellekten okumamı yoksa belleğe yazmamı yapılacağını belirler. Seç ise bu ikili hücredeki bir bitlik veriye ilişkin, bir işlem yapıp yapılmayacağını belirler. Yazma yapılırken girişteki bir bitlik bilgi belleğe, Q ya aktarılır. Okuma yapılırken ise Q belleğindeki bir bitlik bilgi çıkışa aktarılır. Bu ikili hücreye için giriş çıkış ilişkisi aşağıdaki tabloda özetlenmiştir. Q şimdiki durumu, Q^+ bir sonraki durumu, I ise girişteki veriyi göstermektedir.

Seç	Oku / yaz	Giriş	Q^+	Çıkış
0	x	x	q	0
1	0 (yaz)	I	I	0
1	1(oku)	I	Q	Q

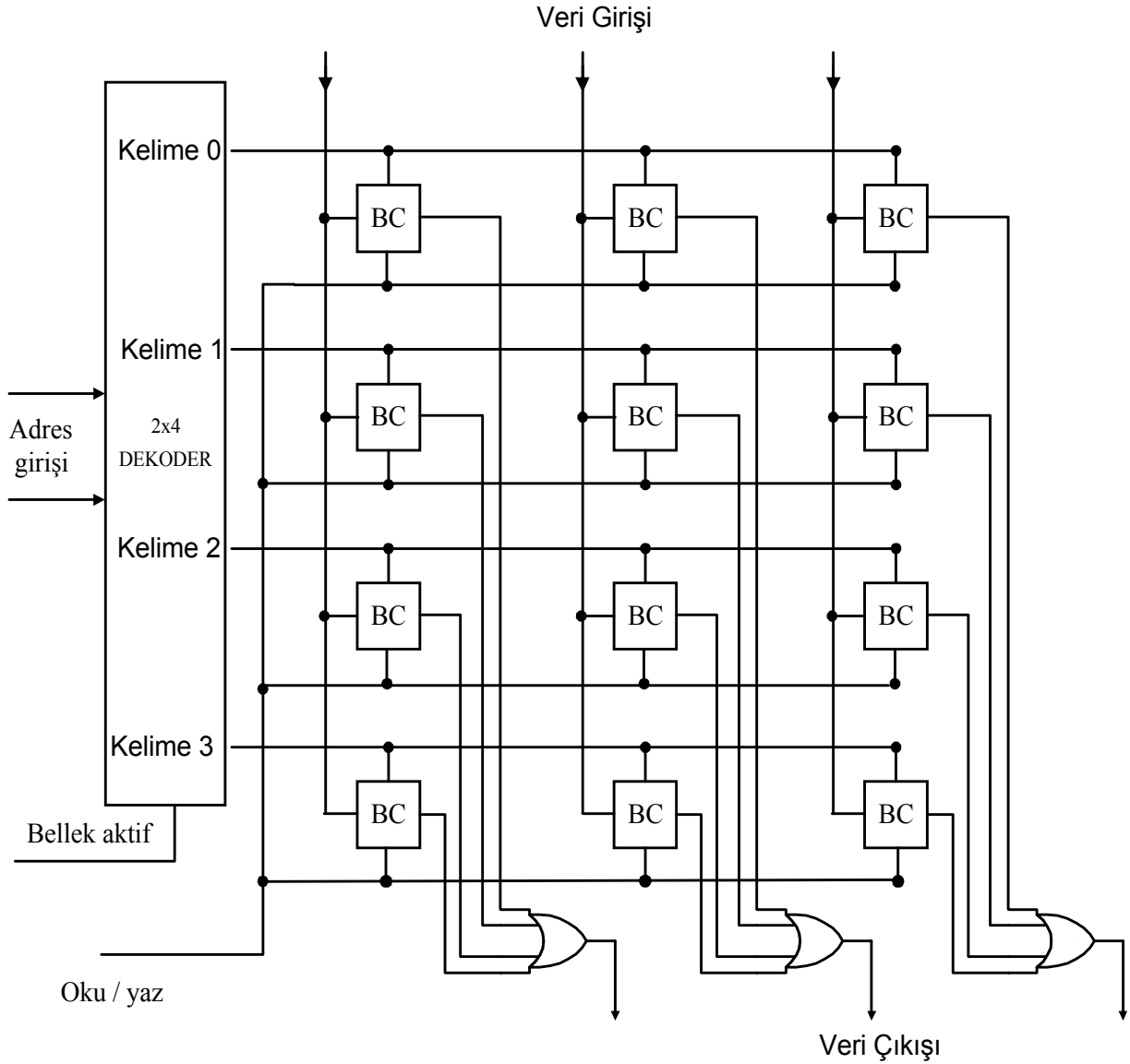


Şekil 10.21 Bir ikili hücrenin lojik devresi ve blok diyagramı

Şekil 10.22 de dört farklı adreste, üç bitlik dört kelime bulunduran bir bellek birimi verilmiştir. 2X4 lük kod çözücü girişindeki iki değişkenin aldığı değere göre yalnızca bir çıkışı 1 olur. Bu çıkış, üç bitlik bir kelime oluşturan üç ikili hücrenin, seç girişlerine bağlanmıştır. Oku /yaz girişi (0) ile Kod çözücünün hangi çıkışındaki kelime uyarılmışsa,

o kelimeyi oluşturan ikili hücelere giriş verisi aktarılır. Oku/Yaz girişi (1) ile ise Kod çözücünün hangi çıkışındaki kelime uyarılmışsa, o kelimeyi oluşturan ikili hücelerdeki veri, çıkış verisi olarak dışarı aktarılır.

Burada her bir adresinde üç bitlik veri bulunan dört adresli bir bellek birimi incelendi. Aynı akıl yürütme ile istenilen adres ve kelime uzunluklu bellek birimleri oluşturulabilir. Uygulamada kullanılan bellek birimleri, genellikle binlerce adresi olan ve her bir adresinde 8, 16, 32, 64 kelime uzunluklu veri olabilen, MSI elemanlardır.



Şekil 10.22 MSI bellek biriminin lojik devresi

EKLER

EK 1 Kaynakça

- Almaini A.:** 'Electronic Logic systems', Prentice-Hall, 1986.
- Davio M., Deschams J., Thayse A.:** 'Digital Systems with Algorirtm Imlementation, John Wiley, 1979.
- Dietmeyer, D.:** 'Logic Desgn of Digital Systems', Allyn and Bacon, 1971.
- Fletcher W.:** 'An Engineering Approach to Digital Design', Prentice-Hall, 1980.
- Givone, D.:** 'Introduction to Switching Circuit Theory', McGraw-Hill, 1970.
- Katz R.H.:** 'Contemporary Logic Design' Benjamin/Cummings, 1994
- Kohavi Z.:** 'Switching and Automata Theory' 1974.
- Lee S.:** 'Modern Switching Theory and Digital Design', Prentice-Hall 1978.
- Mano M.:** 'Sayısal Tasarım', MEB Yayınları, 1994.
- Mano M.:** 'Dgital Logic and Computer Design', Prentice-Hall, 1990.
- Pertman J.:** 'Digital Hardware Design', McGraw-Hill, 1980.
- Roth C.:** 'Fundemantals of Logic Design', West Publishing, 1987.
- Sankur B., Stefanopulos, Y.:**'Bilgisayar ve Mantık Devreleri', Boğaziçi Üniversitesi, 1994.
- Wakerly J.F.:** 'Digital Design' Prentice Hall, 2000
- Wood, P. :** 'Switching Theory', McGraw-Hill, 1968.

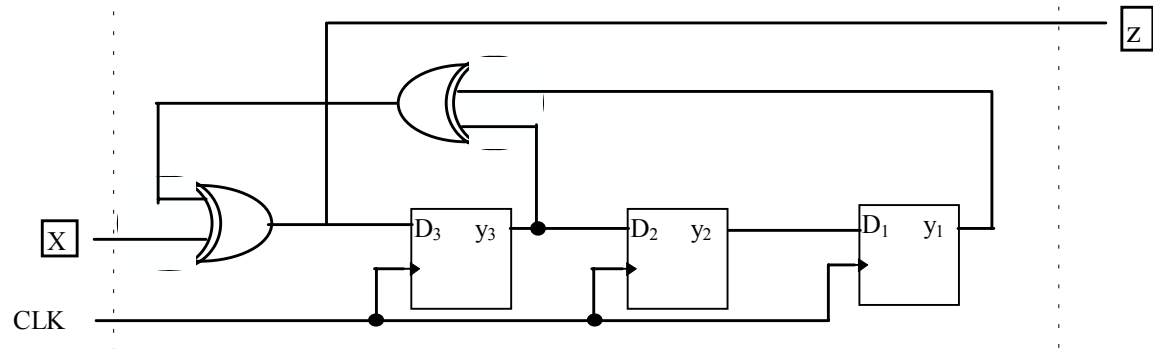
EK 2 Örnek Sınav Soruları

Elektronik ve Haberleşme Bölümü
Devreler ve Sistemler Anabilimdalı

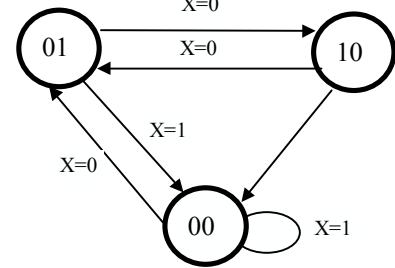
Doç. Dr. Ertuğrul Eriş
6 Aralık 1996

Lojik Devreler Vize Sınavı

- Aşağıda verilen bir giriş ve bir çıkışlı ardışıl devrenin analizi için
 - matematiksel modeli çıkarınız.
 - Durum diyagramını çıkarınız.
 - (00) başlangıç durumunda 101010101 giriş dizisi için çıkış dizisini durum diyagramında çıkışları numaranlandırarak veriniz.



- Durum diyagramı yandaa verilen ardışıl devreyi D tipi bellek elemanları kullanarak gerçekleştiriniz.



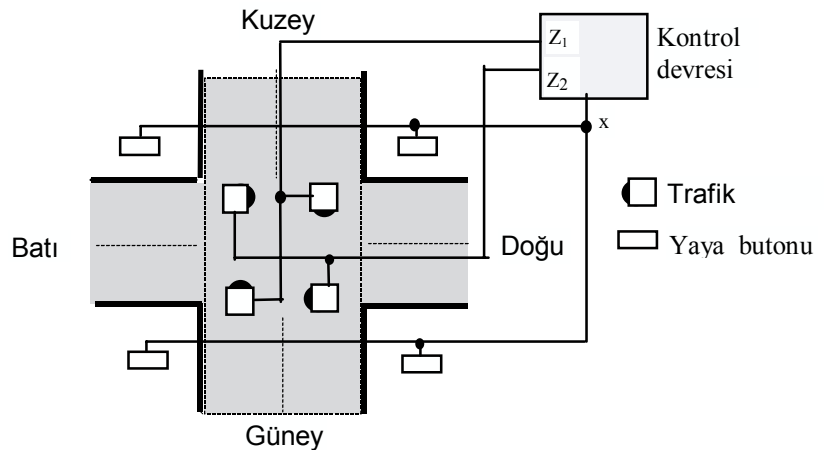
- Aşağıdaki şekilde görülen bir giriş ve iki çıkışlı kontrol devresine ilişkin durum diyagramı çiziniz. Kontrol devresinin sözle tanımı aşağıda verilmiştir.

- Trafik lambaları ilgili olduğu z çıkış değişkeni 1

olduğunda kırmızı yanacak, 0 olduğunda ise yeşil yanacaktır.

- Yayalar karşıdan karşıya geçmek istediklerinde x girişi 1 olacak, istemediklerinde ise 0 olacak. Bu işlem yaya butonu ile sağlanacaktır.

- Yayalardan bir talep gelmedikçe (x girişi 0), yol 1 dakika süreyle dönüşümlü olarak kuzey-güney ve doğu-batı doğrultusunda araç trafiğine açık tutulacaktır. Saat işaretinin periyodunun 1 dakika olduğunu varsayınız.
- Yayalardan bir talep geldiğinde (x girişi 1), yol 1 dakika süreyle yaya trafiğine açık tutulacaktır. Bu durumda bütün trafik lambaları kırmızı yanmalıdır ($z_1=z_2=1$).

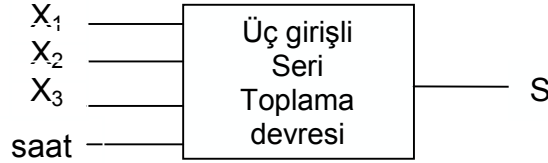


Süre:1.5 saattir. Kitap ve notlara bakılmaz. Puanlar 1) 35 2)35 3)30

Başarılar dilerim.

Lojik Devreler Final Sınavı

1.



		$X_1X_2X_3$							
Y_1y_2		000	001	011	010	110	111	101	100
A		A,0	A,1	B,0	A,1	B,0	B,1	B,0	A,1
B		A,1	B,0	B,1	B,0	B,1	C,0	B,1	B,0
C		B,0	B,1	C,0	B,1	C,0	C,1	C,0	B,1

- a) Durum tablosu yukarıda verilen üç girişli seri toplama devresini D tipi bellek elemanları kullanarak gerçekleyiniz. Durum kodlamasını $A=00$, $B=01$, $C=10$ alınız. Durum tablosu 5 li Karnaugh diyagramına uygun biçimde verilmiştir.
- b) $X_1=7$, $X_2=14$, $X_3=15$ sayılarının toplamına ilişkin saat X_1 , X_2 , X_3 , y_1, y_2 , ve S nin zaman diyagramlarını çiziniz. Başlangıç durumunu 00 seçiniz.

2.

1-7-4-5-1.... biçiminde sayan bir sayıcıyı JK bellek elemanları kullanarak

- a) İki bellek elemanı ile gerçekleyiniz.
- b) Üç bellek elemanı ile gerçekleyiniz.
- c) Sonuçları karşılaştırınız.

3.

- a) Gördüğünüz MSI ardışıl devreleri yazınız.
- b) Senkron Ardışıl devre sentezinin adımlarını yazınız.
- c) Durum indirgemesi yapılmadan gerçekleştirilen devre çalışır mı? Neden?
- d) Durum indirgemesi yapmamakla kaybımız neler olabilir? Neden?
- e) Rastgele Durum kodlaması yapılarak gerçekleştirilen devre çalışır mı? Neden?
- f) Rastgele durum kodlaması yaptığımızda kaybımız ne olabilir? Neden?

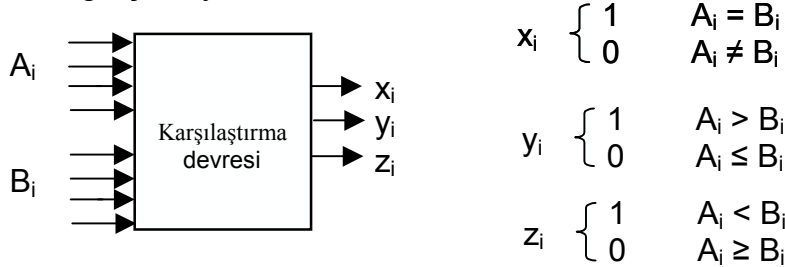
Kitap ve notlara bakılmaz. Süre 2 saattir.
Puanlar 1)40 , 2)40, 3)20.

BAŞARILAR DİLERİM.

Elektronik ve Haberleşme Mühendisliği
Devreler ve Sistemler Anabilim Dalı
Doç.Dr. Ertuğrul Eriş

Lojik Devre Temelleri Final Sınavı

- 1.a) $f = \sum 0,2,4,5,6,11,13,15,18,19,21,22,23,24,26,28,29,30$
fonksiyonuna ilişkin çarpımlar toplamı biçimindeki bütün minimal fonksiyonları bulunuz.
- b) Bulduğunuz minimal fonksiyonlardan birini iki seviyeli gerçekleyiniz
- c) $f = \sum 1,5,6,7,11,15 + \sum_k 2,3,4,10,13$
fonksiyonuna ilişkin bir minimal fonksiyon bulunuz
- d) Bulduğunuz minimal fonksiyonu yalnızca TVE kapılarıyla gerçekleyiniz.
2. $f = x_1 (x_2 + x_3) (x_2 + x_4 + x_5)$
- a) fonksiyonu 1 tane 16x1 lik mux ile gerçekleyiniz.
- b) bu fonksiyonu 8x1 lik bir mux ın enable girişini de kullanarak gerçekleyebilir misiniz? Neden? Nasıl?
- c) Aynı fonksiyonu kaçlık bir dekodele gerçekleyebilirsiniz? Neden? Nasıl?
3. A_1, B_1 en çok ağırlıklı 4 er biti temsil etmek üzere 12 bitlik iki sayı $A=A_1A_2A_3$ ve $B=B_1B_2B_3$ olsun. Bu iki sayıyı aşağıda verilen karşılaştırma devresinden 3 tane kullanarak gerçekleyiniz.

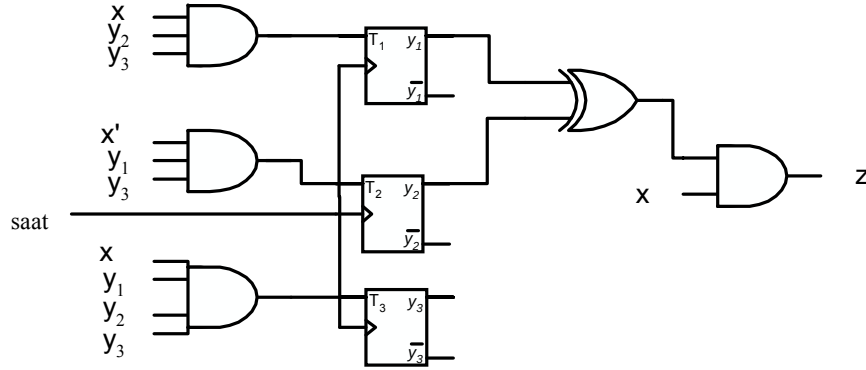


Süre:1.5 saattir. Kitap ve notlara bakılmaz. Puanlar: 1)30, 2)40, 3)30 .

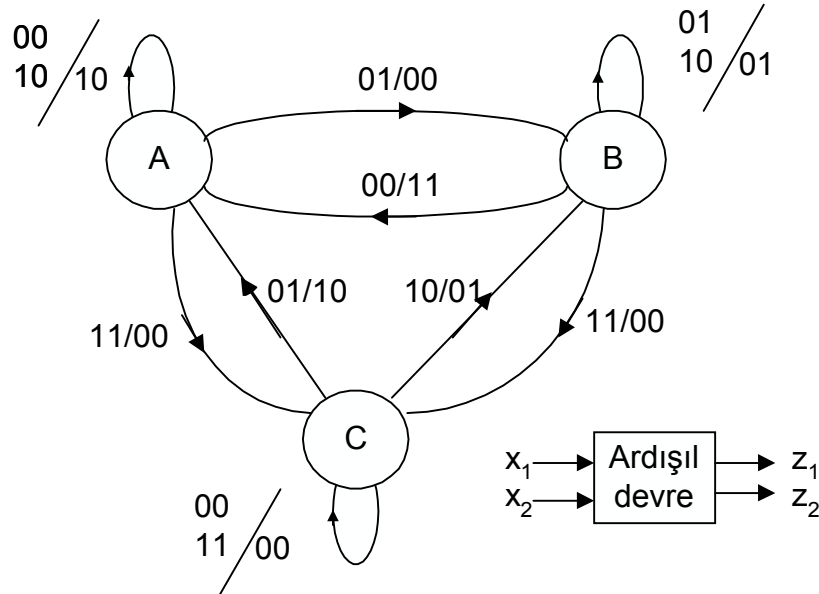
BAŞARILAR DİLERİM

Lojik Devreler Final Sınavı

1.



- c) Yukarıda verilen ardışıl devre daha az sayıda bellek elemanı ile gerçekleştirilebilir mi? durum indirgemesi yaparak gösteriniz?
- d) a daki sonuca göre, gerekiyorsa kotlama yaparak, aynı devreyi D bellek elemanları kullanarak gerçekleyiniz.
- e) Verilen devre ile bulduğunuz devreyi karşılaştırınız.
2. Durum diyagramı verilen senkron ardışıl devreyi



- d) Durum kotlaması yaparak , JK bellek elemanları ile gerçekleyiniz.

e) $X_1 = 001011$

$X_2 = 001011$

Giriş dizisi için çıkış dizisinin zaman diyagramını çizin. Zararlı zararsız çıkışları gösteriniz.

Kitap ve notlara bakılmaz. Süre 2 saattir.

Puanlar 1)60 , 2)40

BAŞARILAR DİLERİM.

26 Mart 1999

Lojik Devre Temelleri Vize 1 sınavı

- a. Boole cebri tanımlayınız.
- b. Boole cebri aksiyomlarını yazınız .

S kümesi $P = \{a,b\}$ kümesinin alt kümelerinden oluşan $S = \{0, 1, A, B\}$ olsun. $0 = \varnothing$, $1 = \{a,b\}$, $A = \{a\}$, $B = \{b\}$. Bu kümeler arasında kümeler teorisinin birleşim ve kesişim ikili işlemleri ile, tümleyen küme birli işlemi. Bu işlemler ve kümeler teorisinin aksiyomları bir Boole cebri oluştursun.

$f(0,0) = a$, $f(0,1) = b$, $f(1,0) = 0$, $f(0,b) = 1$ olmak üzere kaç tane Boole fonksiyonu vardır? Neden? Bu fonksiyonları veriniz.
3 tane Boole olmayan fonksiyon örneği veriniz.

$S = \{0, 1\}$ Boole cebrinde tanımlanan dört değişkenli fonksiyon $f(0000) = f(0001) = f(0100) = f(0110) = f(1000) = f(1011) = f(1111) = 1$ diğer tanım kümesi elemanları için ise 0 olsun

Bu fonksiyonun 1. Tip kanonik açılımını yazınız? Doğru ve yanlış minterimleri belirtiniz.

Bu fonksiyonun 2. Tip kanonik açılımını yazınız? Doğru ve yanlış maxterimleri belirtiniz.

4.

$f_1 = x_1'x_3'x_4 + x_2x_3x_4$, $f_3 = x_1'x_4$ olduğuna göre $f_1 \oplus f_2 = f_3$ bağıntısını sağlayan f_2 fonksiyon(larını) nunu bulunuz.

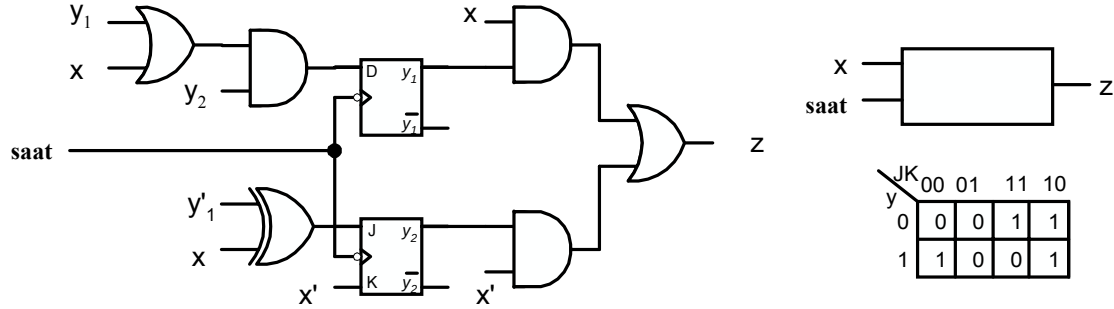
$f_1 = \sum 1,4$; $f_2 = \sum 2,5,6$ ve $f_4 = \sum 1,4,2,5,6$ olduğuna göre $f_1 + f_2 f_3 = f_4$ bağıntısını sağlayan f_3 fonksiyonlarının sayısı kaçtır? Neden? 1 tanesini veriniz.

1) 15 puan, 2) 30 puan, 3) 20 puan, 4)35 puan

Süre 2 saattir. Kitap ve notlara bakılmaz.

Başarılar ve iyi bayramlar dilerim.

1. Aşağıda verilen ardışıl devre için



Matematiksel modeli veriniz.

Durum tablosunu veriniz

Durum diyagramını veriniz.

(00) başlangıç durumu için 0101010 giriş dizisi için çıkış dizisi ve bir sonraki durumlar için zaman diyagramlarını veriniz, hatalı çıkışları da gösteriniz.

2. a. Aşağıda durum tablosu verilen ardışıl devreyi D tipi bellek elemanı kullanarak gerçekleyiniz. Durum kodlaması için A=00, B=01, C=10 ve D=11 alınız.

	X ₁ X ₂			
	00	01	11	10
A	D, 0	A, 0	B, 1	D, 0
B	D, 1	A, 1	C, 0	D, 1
C	B, 0	B, 0	C, 1	B, 0
D	D, 0	A, 0	B, 1	D, 0

A=00 başlangıç durumunda

x₁= 011110

x₂= 010010

Giriş dizisi için çıkış dizisini, durum tablosunda çıkışları numaralandırarak, yazınız. X₂ girişi ile z çıkışı arasındaki ilişki dört temel aritmetik işleminden hangisidir, belirtiniz.

3. Aşağıdaki durum tablosu için durum indirgemesi yapınız, indirgenmiş durum tablosunu çiziniz.

	X ₁ X ₂		
	00	01	10
A	E, 1	C, 0	B, 1
B	C, 0	F, 1	E, 1
C	B, 1	A, 0	D, 1
D	C, 0	F, 1	E, 1
E	A, 0	F, 1	B, 1
F	C, 0	E, 1	C, 1

Bu indirgemenin senteze getireceği yararlar var ise nedenini de açıklayarak belirtiniz.

Bu indirgemenin dışında, bellek elemanları giriş fonksiyonlarına keyfler gelir mi? Gelirse nedenini de açıklayarak yazınız.

Kitap ve notlara bakılmaz. Süre 1,5 saattir. Puanlar 1)40 , 2)40, 3)20

BAŞARILAR DİLERİM.

(30P) 1) İkili işlemi (S^2 -S) aşağıdaki gibi tanımlanmıştır:

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

Aşağıdaki ifadelerin doğru olup olmadığını kanıtlayınız.

(1 A) (A 0) = 1

A B = 1 ise (A + C) (B + C) = 1 dir.

A B = B A

F = $\Sigma 0,1,7,8,9,10,11,13,14,15$ olmak üzere,

(35P) F'e ilişkin çarpımlar toplamı şeklindeki minimal fonksiyon(ları) karnaugh yöntemi ile bulunuz. Bir minimal fonksiyonu iki seviyeli biçimde gerçekleştiriniz.

(35P) F'e ilişkin toplamlar çarpımı şeklindeki minimal fonksiyon(ları) tablo yöntemi ile bulunuz. Bir minimal fonksiyonu TVEYA (NOR) kapıları kullanarak iki seviyeli gerçekleştiriniz.

Süre, 1 saat 15 dakikadır. Kitap ve notlara bakılmaz.
BAŞARILAR DİLERİM

1.(50 Puan)

Verilen durum tablosunda eşdeğer durumlar olup olmadığını inceleyiniz. Varsa alfabetik olarak daha sonra gelen durumları indirgeyerek indirgenmiş durum tablosunu oluşturunuz. Durum diyagramını çiziniz.

Durumlara aşağıdaki gibi kodlar verilmesi halinde oluşacak devreyi JK bellek elemanları kullanarak minimum elemanla gerçekleştiriniz.

a = 010 , b=100 , c=110 , d=101 , e=001 , f=011 , g=000 , h=111

Durum indirgemesi yapmamız bir fayda sağladı mı? Devreyi durum indirgemesi yapmadan gerçekleseydik, ne gibi değişiklikler olurdu? Açıklayınız.

Bu devrede hatalı çıkışlar gözlenebilir mi? Nedenleriyle açıklayınız.

	S	z ₁
	D	z ₂
a	b	00
b	d	01
c	h	01
d	g	00
e	a	10
f	b	00
g	e	01
h	g	00

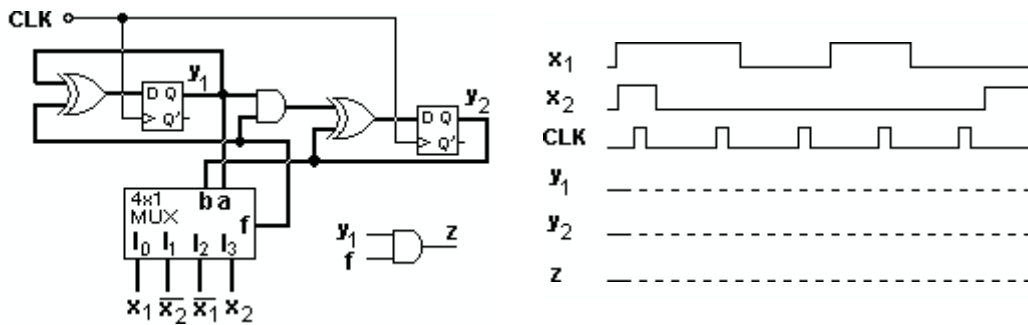
SD:sonraki durum

JK	00	01	11	10
y				
0	0	0	1	1
1	1	0	0	1

yY	00	01	11	10
JK	0k	1k	k0	k1

2.(50 Puan)

Aşağıdaki devrenin analizini yaparak durum tablosunu oluşturunuz, durum diyagramını çiziniz (MUX'un ağırlıksız seçim ucu a'dır).



Devreye 00 başlangıç durumunda aşağıdaki girişlerin uygulanması halinde, çıkışların ve durum değişkenlerinin alacağı değerleri zaman diyagramı üzerinde gösteriniz, zararlı hatalı çıkış var mıdır, açıklayınız.

Süre, 1 saat 15 dakikadır. Kitap ve notlara bakılmaz.
BAŞARILAR DİLERİM

2.2.2001

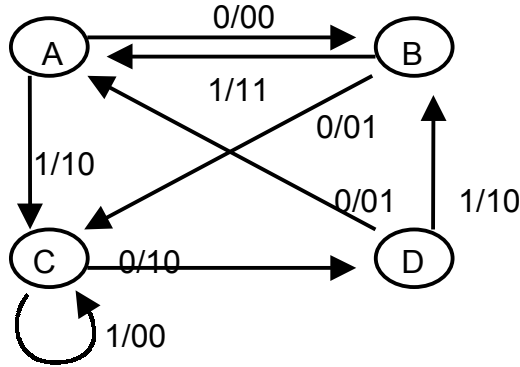
1.(40 Puan)

Yanda durum tablosu verilen ardışıl makinada eşdeğer durumları indirgeyiniz
Durum kodlaması yapınız.

	x=0	x=1
A	B,00	C,11
B	H,01	E,11
C	C,01	F,10
D	H,01	E,11
E	E,11	A,10
F	G,00	F,01
G	D,00	C,11
H	A,10	B,10

2.(40 Puan)

Aşağıda durum diyagramı verilen devreyi JK tipi bellek elemanları ile gerçekleyiniz.

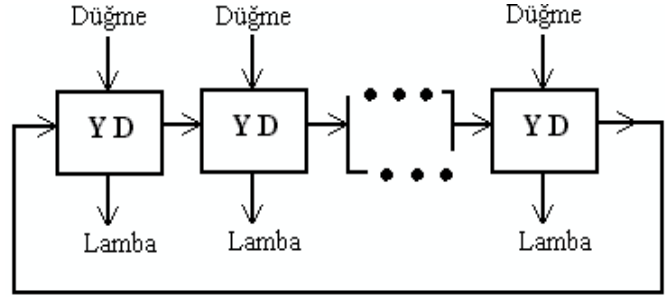


Durum kodları:

A : 00
B : 11
C : 01
D : 10

JK ters tanım bağıntısı:

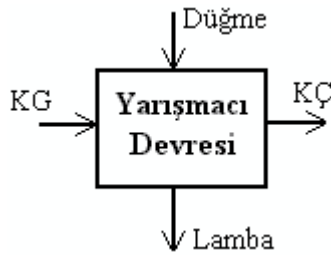
yY	00	01	11	10
jk	0k	1k	k0	k1



3.(20 Puan)

Bilgi yarışmalarında, sorulan soruya sadece düğmesine ilk basanın cevap vermesine olanak tanıyacak bir sistem tasarlanacaktır.

Cevap verme hakkına sahip olan yarışmacının lambası yandıktan sonra, bu yarışmacı da dahil hiçbirinin uygulayacağı giriş durumu değiştirmeyecektir. Sistem her bir yarışmacı için aynı olan birimlerin (yarışmacı devresi) yandaki şekildeki gibi kaskad bağlanması ile oluşturulur.



i.birim (yarışmacı devresi) için ;

Düğme (D): i. yarışmacının kullandığı giriş (düğmeye basıldığında :1 ; Basılmadığında :0)

Komşu Giriş (KG): Herhangi bir düğmeye basıldığı bilgisinin iletildiği giriş.

Komşu Çıkış (KÇ): Herhangi bir düğmeye basıldığı bilgisinin iletildiği çıkış.

Lamba (L): i. Yarışmacının cevap verme hakkı olduğunu gösteren çıkış. (Yanıt verme hakkı var:1 ; yok : 0)

Yarışmacı devresine ait durum diyagramını çiziniz.

NOT: Yukarıdaki sistem yapısı dışında herhangi bir sistem yapısı da tasarlayabilirsiniz.

Süre, 2 saattir. Kitap ve notlara bakılmaz.

BAŞARILAR DİLERİM

abece, 1
 Akım mantığı, 28
 alphanumeric (abece sayısal), 2
 Ana-bağımlı, 94
 Anahtar fonksiyonları, 25
 analiz, 32
 ardışıl (sequential) devreler, 5
 ardışıl devreler, 81
 ardışıl makinalar, 81
 ardışıl mantık, 81
 Asal bileşen (prime implicant), 41
 asal bileşenler tablosu, 44
 ASCII, 2
 asenkron sayıcılar, 136
 Asenkron makinalar, 85
 associative, 16
 Ayrıcılı-VEYA kapısı, 30
 Ayrık zamanlı (discrete) işaret, 1
 Bağıntı (Relation), 12
 BCD (Binary Coded Decimal), 2
 bellek, 86
 binary cell, 146
 Birim eleman, 14
 birinci tip kanonik açınım, 21
 birleşim, 18
 birleştirme işlemi, 42
 Boole Cebri, 14
 Boole fonksiyonu, 19
 Boole fonksiyonunun uzunluğu, 19
 Boole olmayan fonksiyon, 19
 Bütün işlemler, 35
 carry generate, 64
 carry propagation, 64
 cebirine iki değerli Boole cebri, 25
 Clear, 98
 Consensus teoremi, 17
 Counters), 136
 Çıkış yelpazesi (Fan out), 31
 çarpımlar terimleri toplamı, 21
 Çevre, 37
 çevrim (cycling), 92
 Çoğullayıcı, 72
 Dağılma (distributive), 14
 darbe modu, 85
 De Morgan teoremi, 17
 Decoders, 69
 değer kümesi, 12
 Değişken, 14
 devingen (dinamik) bellek eleman, 4
 Devingen bellek elemanı, 4
 Doğru ve yanlış minterimler, 25
 Doğru ve yanlış makterimler, 25
 Doğruluk tablosu, 26
 Don't care functions, 56
 dual, 14
 durağan (statik) bellek eleman, 4
 Durum diyagramı, 84
 Durum tablosu, 84
 durumların kodlanması, 107
 EAROM, 77
 EBCDIC, 2
 Electronically alterable ROM, 77
 elektronik olarak silinebilen ROM, 77
 enable, 72

encoder), 72
 EPROM, 77
 EPROM, 77
 Erasable PROM, 77
 erasable ROM, 77
 Eşdeğer durum, 118
 Eşdeğer kuvvet (idempotent), 15
 Eşdeğerlik bağıntısı (equivalence relation), 13
 flip flop, 86
 Fonksiyon, 12
 Fonksiyonel bağıntı (functional relation), 12
 full adder, 59
 fundemental mode, 85
 Geçiş tablosu, 84
 Geçişlilik (transitive) özellik, 13
 Geri beslemeli mantık devresi, 37
 Gerilim mantığı, 28
 Giriş yelpazesi (Fan in), 31
 Gray kod, 3
 hızlı elde (look ahead carry), 64
 ikili (bit), 1
 ikili hücre, 146
 ikili kodlanmış onlu toplayıcılar, 66
 ikili mantık, 1
 ikinci tip kanonik açınımı, 22
 ileri bakmalı paralel toplayıcı, 66
 ileri beslemeli bir mantık devresi, 37
 indirgenemeyen fonksiyon, 40
 indirgenme, 107
 irreducible, 40
 irredundant, 40
 JK bellek elemanı, 89
 kapı, 5
 kardinalite, 18
 Karnaugh diyagramı, 41
 Kartezyen çarpım, 12
 Kenar tetiklemeli bellek elemanları, 95
 kesişim, 18
 Kısmen belirlenmiş fonksiyonlar, 56
 kodlama(düğümleme), 1
 kodlayıcı, 72
 kombinezonsal (combinational) devreler, 5
 Kombinezonsal devre, 32
 komşu kareler, 52
 konumsal sayı dizgeleri, 6
 kuvvet kümesi, 18
 latch, 91
 Look ahead carry), 66
 LSB(Least Significant Bit), 7
 LSI, 63
 LSI(Large Scale Integrated Circuits), 32
 m. mertebeden alt küp, 52
 Makterim, 23
 Mantık devresi, 31
 mantık devresine ilişkin graf (çizge), 36
 Master slave, 94
 Matematiksel model, 82
 Mealy makinesi, 82
 Minimal (indirgenmiş) fonksiyon, 39
 Minimal bütün işlemler, 35
 Minterim, 22
 Moore makinesi, 82
 MSB(Most Significant Bit), 7
 MSI, 63

MSI (Medium Scale Integrated Circuits), 32
mux, 72
negatif kenar, 91
Negatif mantık, 28
negative edge, 91
n-lik işlem (n-ary operation), 12
önergeler cebri, 28
Örten (cover) fonksiyon, 41
paralel iletim, 4
Paralel yüklemeli ikili senkron sayıcılar, 140
permütasyonları (devşirim), 18
PLA (Programmable Lojik Array), 63
positive edge, 91
pozitif kenar, 91
Pozitif mantık, 28
preset, 98
PROM, 77
pulse mode, 85
Quine Mc- Cluskey yöntemi, 41
ripple counters, 136
ROM, 75
ROM (Read Only Memory), 63
Saatli SR bellek elemanı, 88
Sabit (değişmez), 14
sayamak (digit), 1
Sayısal (digital) işaret, 1
Senkron makinalar, 85
sentez işlemi, 33
seri iletim, 4
seven segment display, 66
Sıra bağıntısı (order relation), 13
siliebilen ROM, 77
silinebilir PROM, 77
Simetri özelliği, 13
sistem, 1
sözle tanımlama, 107
SR bellek elemanı, 86
SSI elemanlar, 31
state assignment, 107

state reduction, 107
strobe, 72
Sürekli işaretli, 1
T tetikleme bellek elemanı, 90
taban (radix), 6
tablo yöntemi, 41
tam toplayıcı, 59
Tam toplayıcı, 60
Tamamen belirlenmiş bağıntı (Completely specified relation), 12
tanım kümesi(domen), 12
Temel asal bileşen (Essential prime implicant), 42
temel bellek elemanları, 88
Temel mod, 85
temel satır, 45
ters tanım bağıntıları, 108
toplamlar terimleri çarpımı, 22
transition table, 84
tümleştirilmiş devre, 31
Tümleyen kapısı, 5
TÜMLEYEN kapısı, 30
TÜVE kapısı, 30
TÜVEYA kapısı, 30
Üreysel (generic) değişken, 15
VE kapısı, 29
VE(AND) kapısı, 5
VEYA kapısı, 30
VEYA(OR) kapısı, 5
VLSI(Very Large Scale Integrated Circuits), 32
word description, 107
Yansıma (refleksive), 13
Yarı toplayıcı, 59
yazmaç (register), 5
yedi bölümlü gösterici, 66
Yer değiştirme (comutative), 14
Yol, 37
Yutma teoremi, 16

KAYNAKÇA

- Almaini A.:** 'Electronic Logic systems', Prentice-Hall, 1986.
- Davio M., Deschams J., Thayse A.:** 'Digital Systems with Algorirtm Imlementation, John Wiley, 1979.
- Dietmeyer, D.:** 'Logic Desgn of Digital Systems', Allyn and Bacon, 1971.
- Fletcher W.:** 'An Engineering Approach to Degital Design', Prentice-Hall, 1980.
- Givone, D.:** 'Introduction to Switching Circuit Theory', McGraw-Hill, 1970.
- Katz R.H.:** 'Contemporary Logic Design' Benjamin/Cummings, 1994
- Kohavi Z.:** 'Switching and Automata Theory' 1974.
- Lee S.:** 'Modern Switching Theory and Digital Design', Prentice-Hall 1978.
- Mano M.:** 'Sayısal Tasarım', MEB Yayınları, 1994.
- Mano M.:** 'Dgital Logic and Computer Design', Prentice-Hall, 1990.
- Pertman J.:** 'Digital Hardware Design', McGraw-Hill, 1980.
- Roth C.:** 'Fundemantals of Logic Design', West Publishing, 1987.
- Sankur B., Stefanopulos, Y.:**'Bilgisayar ve Mantık Devreleri', Boğaziçi Üniversitesi, 1994.
- Wakerly J.F.** 'Digital Design' Prentice Hall, 2000
- Wood, P. :** 'Switching Theory', McGraw-Hill, 1968.