$F_s \geq 2f_{max}$ → Bir işaret örneklenecekse, işaretin içindeki en yüksek frekansın en az 2 katı ile örnekleme yapılır.

Örnekleme frekansı

Sampling → Quantizing → Encoding

* Kullanılan bit sayısı arttıkça kuanta aralığı azalır dolayısıyla hata miktarı da düşer.

AND OR NOT XOR XNOR NAND NOR

Aksiyom: Doğru olarak kabul edilen önermeler

Teorem: Doğruluğu ispat edilebilen önermeler

Örnek: $ab+\bar{a}c+bc = ab+\bar{a}c$ olduğunu gösteriniz.

$ab+\bar{a}c+bc\underbrace{(a+\bar{a})}_{1} = ab+\bar{a}c+abc+\bar{a}bc$

$= ab\underbrace{(1+c)}_{1}+\bar{a}c\underbrace{(1+b)}_{1}$

$= ab+\bar{a}c$

x	y	Minterm	m_i	Maksterm	M_i
0	0	$\bar{x}\bar{y}$	m_0	xy	M_0
0	1	$\bar{x}y$	m_1	$x\bar{y}$	M_1
1	0	$x\bar{y}$	m_2	$\bar{x}y$	M_2
1	1	xy	m_3	$\bar{x}\bar{y}$	M_3

Minterm → çarpımlar toplamı

Maksterm → toplamlar çarpımı

Minterm: Minimum Terimler Kanonik Biçimi

$f(a,b,c) = ab+\bar{a}c$ ifadesini mintermler cinsinden yazınız.

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

$f(a,b,c) = \bar{a}\bar{b}\bar{c}\,f(0,0,0)+\bar{a}\bar{b}c\,f(0,0,1)+\bar{a}b\bar{c}\,f(0,1,0)+\bar{a}bc\,f(0,1,1)$
$+a\bar{b}\bar{c}\,f(1,0,0)+a\bar{b}c\,f(1,0,1)+ab\bar{c}\,f(1,1,0)+abc\,f(1,1,1)$

$f(a,b,c) = ab+\bar{a}c = \bar{a}\bar{b}c+\bar{a}bc+ab\bar{c}+abc$

$= \Sigma(1,3,6,7) = m_1+m_3+m_6+m_7$ → minterm gösterimi

$= \Pi(0,2,4,5) = M_0+M_2+M_4+M_5$ → maksterm gösterimi

Not: Mintermler ile makstermler aslında birbirinin tümleyenidir. $(m_i = \overline{M_i})$

Örnek:

a	b	c	f(a,b,c)	
0	0	0	1	$\bar{a}\bar{b}\bar{c}$
0	0	1	1	$\bar{a}\bar{b}c$
0	1	0	0	$(a+\bar{b}+c)$
0	1	1	0	$(a+\bar{b}+\bar{c})$
1	0	0	1	$a\bar{b}\bar{c}$
1	0	1	1	$a\bar{b}c$
1	1	0	0	$(\bar{a}+\bar{b}+c)$
1	1	1	0	$(\bar{a}+\bar{b}+\bar{c})$

$= (\bar{a}\bar{b}\bar{c})+(\bar{a}\bar{b}c)+(a\bar{b}\bar{c})+(a\bar{b}c)$

$= [a+\bar{b}+c].[a+\bar{b}+\bar{c}].[\bar{a}+\bar{b}+c].[\bar{a}+\bar{b}+\bar{c}]$

Lojik Fonksiyonların İndirgenmesi

1- Görüşe Dayalı İndirgeme
2- Karnaugh Maps
3- Quine McCluskey

Görüşe Dayalı İndirgenme

KURAL: $a+bc = (a+b).(a+c)$

Örnek 1: $f(a,b,c,d) = \bar{a}b\bar{c}+ad+b\bar{c}\bar{d}$

$= b\bar{c}(\bar{a}+\bar{d})+ad$

$= b\bar{c}.\overline{ad}+ad$

$= (ad+b\bar{c}).\underbrace{(ad+\overline{ad})}_{1}$

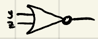
$= ad+b\bar{c}$

Örnek 2: $f(x,y,z) = \Sigma(0,4)$ $4=(100)_2$

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

$= \bar{y}\bar{z}(\bar{x}+x)$

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

$= (y+z)'$ ⇒ 

Karnaugh Maps

Örnek 1: $f(x,y) = x+y$

x	y	f(x,y)
0	0	0
0	1	1
1	0	1
1	1	1

x \ y	0	1
0	0	1
1	1	1

Hamming Distance:

$y = ham(010,111)$
$y = 2$

farklı değer varsa 1 artır.

10110101
11001110 > 6

Örnek 2: $f(x,y,z) = x+y+z$

x	y	z	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

her geçişte 1 bit değişim olmalı, ondan sıra böyle yazılır.

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

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

Örnek 3: $f(x,y,z,w) = xy+wz$

x	y	z	w	f
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

xy \ zw	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	1	1	1	1
10	0	0	1	0

Karnaugh Yönteminde Komşuluk:

n değişkenli bir fonksiyonda 2^k tane terim kısalıp $(n-k)$ çarpanlı bir tek terime dönüşürse bu 2^k tane terim k. mertebeden komşudur.

Teorem 1) Birinci Dereceden Komşuluk: İki terimin birinci dereceden komşu olması için gerek ve yeter koşul, bunlara karşı düşen fonksiyonların alan üzerinde komşu olmasıdır.

$2^k = 2 \Rightarrow k=1$ $n=4 \Rightarrow (n-k)=3$ çarpanlı

xy \ zw	00	01	11	10
00	1	1		1
01		1	1	
11			1	
10		1		

$f(x,y,z,w) = \bar{x}yw+yzw+\bar{x}\bar{y}\bar{w}+\bar{y}\bar{z}w$

Teorem 2) İkinci Dereceden Komşuluk: 4 terimin ikinci dereceden komşu olması için gerek ve yeter koşul bunlara karşı düşen fonksiyonların alan üzerinde bir sıra şeklinde olması veya kare oluşturmasıdır.

Dört değişken için $2^k = 4 \Rightarrow k=2$ $(n-k)=2$ çarpanlı terimler

Not: Değişmeyenleri alıyoruz.

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

bd ab $\bar{b}\bar{d}$
$\bar{b}d$

Teorem 3) Üçüncü Dereceden Komşuluk: 8 terimin üçüncü dereceden komşu olması için gerek ve yeter koşul bunlara karşı düşen fonksiyonların alan üzerinde komşu 2 sıra oluşturmasıdır.

$2^k = 8$ terim $k=3$

$(n-k) = 4-3 = 1$ çarpanlı

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

$f(a,b,c,d) = d+\bar{b}\bar{d}$

$f(a,b,c,d) = d+\bar{b}$

Daha önce kullanılanlar olsa da yine de kullanılabilir.

Teorem 4) Dördüncü Dereceden Komşuluk: 16 terimin dördüncü dereceden komşu olması için gerek ve yeter koşul bunlara karşı düşen fonksiyonların bütün alanı kaplıyor olması gerekir.

$2^k = 16$ $k=4$

$(n-k) = (4-4) = 0$
terim yok

$f(a,b,c,d) = 1$
devamlı 1'dir

Örnek: $f(a,b,c,d) = \overline{a}\overline{b}\overline{c} + \overline{b}c\overline{d} + \overline{a}bc\overline{d} + a\overline{b}\overline{c}$

Bilgi

$\overline{a}\overline{b}\overline{c} = \overline{a}\overline{b}\overline{c}(d+\overline{d})$

$= \overline{a}\overline{b}\overline{c}d + \overline{a}\overline{b}\overline{c}\overline{d}$

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

Çarpımların Toplamı

Minterm

$f(a,b,c,d) = \overline{b}\overline{c} + \overline{b}\overline{d} + \overline{a}c\overline{d}$

↓

Alternatif

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

Toplamların Çarpımı

Maksterm

$f(a,b,c,d) = (\overline{c}+\overline{d})(\overline{b}+c)(\overline{a}+\overline{b})$

Örnek: $f(a,b,c,d) = \Sigma(0,3,4,5,6,7,13,14)$

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

$f = \overline{a}\overline{c}\overline{d} + b\overline{c}d + \overline{a}cd + bc\overline{d}$

Yeşilli olanı yazmaya gerek yok çünkü hepsi zaten eşlendi bu yüzden boşa terim artmış olur.

Eksik Boole İşlemleri

Örnek: $f(a,b,c,d) = \Sigma(3,7,11,12,15) + \Sigma_{\phi}(0,10,13,14)$

ab \ cd	00	01	11	10
00	Ø (0)	(1)	1 (3)	(2)
01	(4)	(5)	1 (7)	(6)
11	1 (12)	Ø (13)	1 (15)	Ø (14)
10	(8)	(9)	1 (11)	Ø (10)

$f(a,b,c,d) = cd + ab$

Maks komşuluk
Min terim

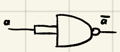
a	b	c	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	Ø

Ø → joker
→ keyfi
→ don't care

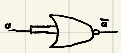
Not: Bir f fonksiyonunda örneğin 1101 durumu hiç oluşmuyorsa, o durum da Ø olarak değerlendirilebilir.

Türetilmiş Kapılarla Temel Lojik Kapıların Yapılması

NOT



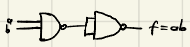
$f = \overline{ab} = \overline{a.a} = \overline{a}$



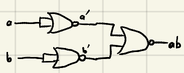
$f = (\overline{a+b}) = (\overline{a+a}) = \overline{a}$

NAND

$f = a.b = (\overline{\overline{a.b}})$

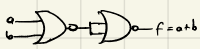


$f = a.b = (\overline{\overline{a.b}}) = (\overline{a'+b'})$

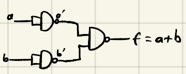


NOR

$f = a+b = (\overline{\overline{a+b}})$



$f = a+b = (\overline{a'b'})$



Quine McCluskey Yöntemi Örnek:

$f(a,b,c,d) = \Sigma(1,4,6,7,8,9,10,11,15)$ fonksiyonunu sadeleştiriniz

1'lerin sayısı	minterm	abcd	işaret
1	1	0001	✓
	4	0100	✓
	8	1000	✓
2	6	0110	✓
	9	1001	✓
	10	1010	✓
3	7	0111	✓
	11	1011	✓
4	15	1111	✓

Grup	abcd	işaret
1-9	-001	AB
4-6	01-0	AB
8-9	100-	✓
6-7	011-	✓
9-11	10-1	AB
10-11	101-	✓
7-15	-111	✓
8-10	10-0	AB
11-15	1-11	AB

Grup	abcd	işaret
8-9-10-11	10--	AB

Grup	abcd	terim	Minterm 1	4	6	7	8	9	10	11	15
1-9	-001	b'c'd	X					X			
4-6	01-0	a'bd'		X	X						
6-7	011-	a'bc			X	X					
7-15	-111	bcd				X					X
11-15	1-11	acd								X	X
8-9-10-11	10--	ab'					X	X	X	X	

$f(a,b,c,d) = b'c'd + a'bd' + ab' + bcd$

İndirgenmiş İfadelerin Aynı Tür Kapılarla Gerçekleştirilmesi

1-) Çarpımlar Toplamı NAND Tasarımı

$f(x,y,z) = xyz + xy' + xz$ $\qquad f = \Sigma\Pi x_i$ → Çarpımların Toplamı

$\overline{f} = \overline{\Sigma\Pi x_i} \qquad \overline{f} = \Pi\overline{\Pi x_i} \qquad \overline{\overline{f}} = \overline{\Pi\overline{\Pi x_i}}$

Örnek: $f(a,b,c) = \overline{a}\overline{b}c + \overline{a}bc + ab\overline{c} + abc$

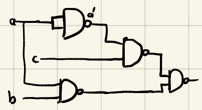
a \ bc	00	01	11	10
0		1	1	
1			1	1

$f(a,b,c) = \overline{a}c + ab$

$\overline{f}(a,b,c) = (\overline{\overline{a}c + ab})$

$\overline{f} = (\overline{\overline{a}c}).(\overline{ab})$

$\overline{\overline{f}} = (\overline{(\overline{a'c}).(\overline{ab})})$



2-) Çarpımlar Toplamı NOR Tasarımı

$f = \Sigma\Pi x_i \qquad \Pi x_i = \overline{\Sigma\overline{x_i}}$

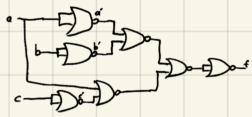
Örnek: $\Pi x_i = \overline{\overline{abc}} \qquad \Pi x_i = (\overline{\overline{a}+\overline{b}+\overline{c}})$

$f = \Sigma\overline{\Sigma\overline{x_i}} \Rightarrow \overline{f} = \overline{\Sigma\overline{\Sigma\overline{x_i}}}$

Örnek: $f(a,b,c) = ab + \overline{a}c$

$= (\overline{\overline{ab}}) + (\overline{\overline{\overline{a}c}}) \Rightarrow (\overline{a'+b'}) + (\overline{a+c'}) = f$

$f' = \overline{(\overline{a'+b'}) + (\overline{a+c'})} \qquad \overline{\overline{f}} = \overline{\overline{(\overline{a'+b'})+(\overline{a+c'})}}$



3-) Toplamlar Çarpımı NOR Tasarımı

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

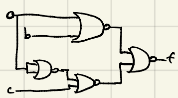
$f = \Pi\Sigma x_i \qquad \overline{f} = \overline{\Pi\Sigma x_i} \Rightarrow \overline{f} = \Sigma\overline{\Sigma x_i}$

$f = \overline{\overline{f}} = \overline{\Sigma\overline{\Sigma x_i}}$

Örnek: $f(a,b,c) = (a+b)(a'+c)$

$\overline{f} = (\overline{a+b}) + (\overline{a'+c})$

$\overline{\overline{f}} = (\overline{(\overline{a+b})+(\overline{a'+c})})$



4-) Toplamlar Çarpımı NAND Tasarımı

$f = \Pi\Sigma x_i \qquad \Sigma x_i = \overline{\Pi\overline{x_i}}$

$\overline{f} = \overline{\Pi\overline{\Pi\overline{x_i}}} \qquad \overline{\overline{f}} = \overline{\overline{\Pi\overline{\Pi\overline{x_i}}}}$

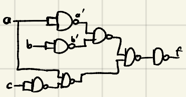
Örnek $f(a,b,c) = [a+b][a'+c]$

$f' = (\overline{a+b}) + (\overline{a'+c})$

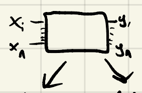
$f' = (a'b') + (ac')$

$f'' = (\overline{a'b'}).(\overline{ac'})$

$f'' = \overline{\overline{(\overline{a'b'}).(\overline{ac'})}}$



Kombinezonal Devreler (Statik, belleksiz)



Kombinezonal Devreler

$x_i(t) \to y_i(t)$

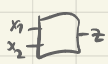
Ardışıl Devreler

$x_i(t) \to$ ⊕ $\to y_i(t)$
$d_i(t) \to$
↳ hafıza

a) Tasarımı

1) Problem tanıtımı
2) $x_i(t)$, $y_i(t)$ sayıları belirlenir.
3) Doğruluk tablosu
4) İndirgeme
5) Kapı seçimi
6) Devre lojik şeması çizilir

Örnek:

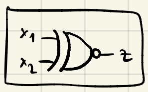


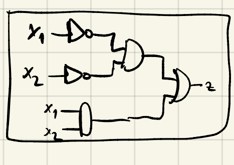
giriş	çıkış
aynı	1
farklı	0

x_1	x_2	z
0	0	1
0	1	0
1	0	0
1	1	1

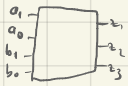
x_1 \ x_2	0	1
0	1	0
1	0	1

$z = \overline{x_1}\overline{x_2} + x_1x_2$





Örnek 2: $\underbrace{a_1 a_0}_{A}$ $\underbrace{b_1 b_0}_{B}$ iki bitlik genlik karşılaştırıcı devreyi tasarlayın



$A = B \Rightarrow z_2 = 1$
$A > B \Rightarrow z_1 = 1$
$A < B \Rightarrow z_3 = 1$

a_1	a_0	b_1	b_0	z_1	z_2	z_3
0	0	0	0		1	
0	0	0	1			1
0	0	1	0			1
0	0	1	1			1
0	1	0	0	1		
0	1	0	1		1	
0	1	1	0			1
0	1	1	1			1
1	0	0	0	1		
1	0	0	1	1		
1	0	1	0		1	
1	0	1	1			1
1	1	0	0	1		
1	1	0	1	1		
1	1	1	0	1		
1	1	1	1		1	

z_1 için karno

b_1b_0 \ a_1a_0	00	01	11	10
00		1	1	1
01			1	1
11				
10			1	

$z_1 = a_1\bar{b}_1 + \bar{b}_1\bar{b}_0 a_0 + a_1 a_0 \bar{b}_0$

Kombinezonsal Tümleşik Devreler

1-) Aritmetik Toplama Çıkarma Devreleri
- Yarı toplayıcı
- Tam toplayıcı
- Yarı çıkarıcı
- Tam çıkarıcı
- Toplama ve çıkarma devresi

2-) Seçiciler (Multiplexer / MUX)

3-) Dağıtıcılar (Demultiplexer / DEMUX)

4-) Kod Çözücüler / kodlayıcılar (Decoder/Encoder)

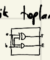
5-) Aritmetik Lojik Birimler (ALU)

6-) 7-Segment Display

a) Aritmetik Toplama ve Çıkarma Devreleri

1-) Yarı toplayıcı / Half adder: 1 bitlik toplama gerçekleştirir. 2 giriş 2 çıkışı var. Elde çıkış olarak vardır. Ancak girişi yok. (Girişi olan tam t.)

a	b	T	E (elde)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1
		xor	and



2-) Tam toplayıcı / Full adder:

E_g	a	b	T	E
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

T için

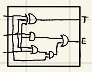
0	1	0	1
1	0	1	0

$T = a \oplus b \oplus c$

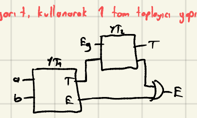
E için

0	0	1	0
0	1	1	1

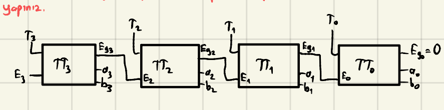
$E = ab + E_g(a+b)$



Soru: 2 yarı t. kullanarak 1 tam toplayıcı yapılır mı? Evet



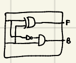
Soru: 4 bitlik iki sayının toplamını bulan paralel toplayıcı devrenin tasarımını TT kullanarak yapınız.



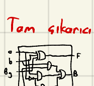
$$\begin{array}{r} E_3 \; a_3 a_2 a_1 a_0 \quad E_{g0}=0 \\ + \; b_3 b_2 b_1 b_0 \\ \hline T_3 T_2 T_1 T_0 \end{array}$$

3-) Yarı çıkarıcı:

a	b	F (fark)	B (borç)
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0
		xor	$B = \bar{a}b$



4-) Tam çıkarıcı:



a	b	B_g (b gibi düşün)	F	B
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Not: Alınan borç 2 olarak gelir.

	1		1
1		1	

$F = a \oplus b \oplus c$

	1		
1	1	1	

$B = \bar{a}b + \bar{a}B_g + bB_g$

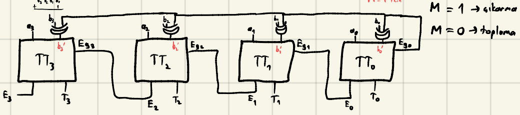
5-) Toplama çıkarma devresi:

$$\begin{array}{r} a_3 a_2 a_1 a_0 \\ - \; b_3 b_2 b_1 b_0 \end{array} \quad \text{ise} \rightarrow \begin{array}{r} b_3' b_2' b_1' b_0' \\ + \quad 1 \end{array} \; \text{2' complement}$$

$$\begin{array}{r} a_3 a_2 a_1 a_0 \\ b_3' b_2' b_1' b_0' \\ + \quad 1 \end{array}$$

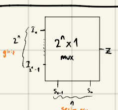
a	b	f
0	0	0
0	1	1
1	0	1
1	1	0

a=0 iken f=b
a=1 iken f=b'



M = 1 → çıkarma
M = 0 → toplama

b) Seçiciler (MUX)



Örnek: Doğruluk tablosu verilen devreyi x_3 ve x_1 seçme girişleri olacak şekilde 4x1 mux ile gerçekleyiniz.

x_1	x_2	x_3	x_4	f
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

$f = x_1\bar{x}_2 + \bar{x}_2 x_3 + \bar{x}_1 x_2 x_4$

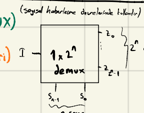
$f = x_3x_4x_1\bar{x}_2 + \bar{x}_3x_4x_1\bar{x}_2 + x_3\bar{x}_4x_1\bar{x}_2 + \bar{x}_3\bar{x}_4x_1\bar{x}_2 + x_3x_4\bar{x}_2 + x_3\bar{x}_4\bar{x}_2 + x_3x_4\bar{x}_1 + \bar{x}_3x_4\bar{x}_1$

$f = \bar{x}_3\bar{x}_4\cdot(x_1\bar{x}_2) + \bar{x}_3x_4\cdot(x_1\bar{x}_2 + \bar{x}_1) + x_3\bar{x}_4\cdot(x_1\bar{x}_2 + \bar{x}_2) + x_3x_4(x_1\bar{x}_2 + \bar{x}_2 + \bar{x}_1)$ $\quad \bar{x}_2(x_1 + 1) + \bar{x}_1 = \bar{x}_2 + \bar{x}_1$

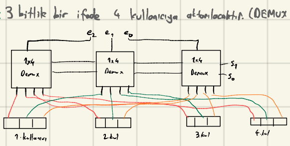
$I_0 = x_1\bar{x}_2 \qquad I_1 = (\bar{x}_1 + \bar{x}_2) \qquad I_2 = \bar{x}_2 \qquad I_3 = \bar{x}_1 + \bar{x}_2$

c-) Dağıtıcılar (DEMUX)

(Sayısal haberleşme devrelerinde kullanılır)

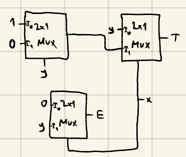


Örnek: 3 bitlik bir ifade 4 kullanıcıya aktarılacaktır. (DEMUX ile yapın) $e_2 e_1 e_0$



★ Vize ★

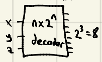
Soru: Yarı toplayıcı devreyi 3 adet 2x1'lik MUX ile oluşturunuz.

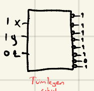


d-) Kod Çözücüler (Decoder)

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

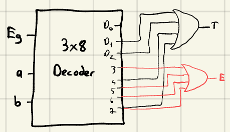
n=3 için





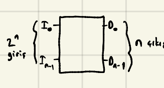
Örnek: Tam toplayıcıyı kod çözücü kullanarak tasarlayınız.

$T(E_g, a, b) = \Sigma(1,2,4,7)$
$E(E_g, a, b) = \Sigma(3,5,6,7)$



e-) Kodlayıcı (Encoder)

(Sayı sistemlerinde dönüşüm)

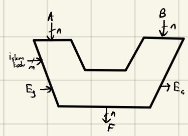


En yüksek biti detect etmede kullanılır.

Girişler				Çıkışlar		
I_3	I_2	I_1	I_0	O_1	O_2	Ready
0	0	0	0	∅	∅	0
0	0	0	1	0	0	1
0	0	1	∅	0	1	1
0	1	∅	∅	1	0	1
1	∅	∅	∅	1	1	1

∅ => ne olduğunun önemi yok

f-) ALU (Arithmetic Logic Unit)



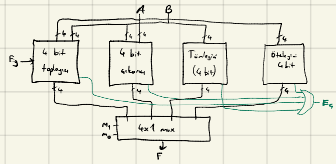
Örnek: 4 bitlik 2 giriş $\underbrace{a_3 a_2 a_1 a_0}_{A}$ $\underbrace{b_3 b_2 b_1 b_0}_{B}$

E_g, $E_ç$

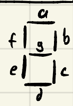
Sonuç $F(f_3, f_2, f_1, f_0)$

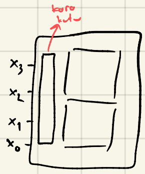
İşlem sayısı = 4 $\quad m_1 m_0$

M_1	M_0	İşlem
0	0	F=A+B
0	1	F=A−B
1	0	F=B'
1	1	F=A>>1



g-) 7-Segment Display

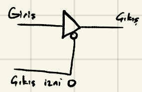




X_3	X_2	X_1	X_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	0	0	1	1	1	1	1	0	1	1

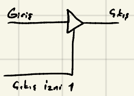
Not: Lojik birimlerin çıkışları birbirine bağlanmaz. Buffer'lar bunun için kullanılır. Açık kolektörlü yapılar da bunun için kullanılır.

Tri-State Bufferler (Üç Durumlu)



Çıkış İzni	Giriş	Çıkış
0	0	0
0	1	1
1	0	HE
1	1	HE

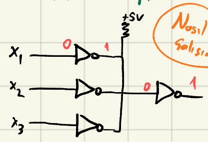
Yüksek empedans



Çıkış İzni	Giriş	Çıkış
0	0	HE
0	1	HE
1	0	0
1	1	1

Yüksek empedans

Açık kolektörlü Yapılar



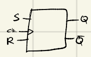
Nasıl çalışır?

Karmaşıklık Hesabı

Eleman karmaşıklığı — Zaman karmaşıklığı

SR Flip-Flop

	S	R	q	Q
Sakla	0	0	0	0
	0	0	1	1
Reset	0	1	0	0
	0	1	1	0
Set	1	0	0	1
	1	0	1	1
Yasak	1	1	0	X
	1	1	1	X



$Q(t+1) = Q(t).\overline{R(t)} + S(t)$
$R(t).S(t) = 0$ için tanımlı
$R(t).S(t) = 1$ için tanımsız

S	R	Q
0	0	q
0	1	0
1	0	1
1	1	X

y \ SR	00	01	11	10
0	0	0	∅	1
1	1	0	∅	1

$Y = S + R'y$

uyarma tablosu

yY	00	01	11	10
SR	0∅	10	∅0	01

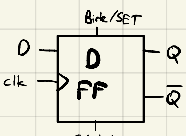
$$Q = S + R'q$$

D Flip-Flop

D	q	Q
0	0	0
0	1	0
1	0	1
1	1	1

D ne ise sonuçta o olur.
Çıkış bir önceki çıkışa bağlı değil

D	Q
0	0
1	1



$Q = D$

y \ D(t)	0	1
0	0	1
1	0	1

$Y = D$

$Q(t+1) = D(t)$
$Q(t+1) = Y$
$Q(t) = y$

Uyarma tablosu

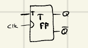
yY	00	01	11	10
D	0	1	1	0

T Flip-Flop

	T	q	Q
korur	0	0	0
	0	1	1
tersine çevirir	1	0	1
	1	1	0

T	Q
0	q
1	q'

$Q = T \oplus q$



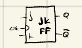
$Q(t+1) = Q(t) \oplus T(t)$

y \ T	0	1
0	0	1
1	1	0

$Y = Ty' + T'y$

$Y = T \oplus y$

JK Flip-Flop



$Q(t+1) = Q(t)\overline{K(t)} + \overline{Q(t)}.J(t)$

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

$Y = J\bar{y} + \bar{K}y$

Uyarma Tablosu

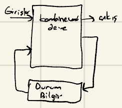
yY	00	01	11	10
JK	0Ø	1Ø	Ø0	Ø1

	J	K	q	Q
SR	0	0	0	0
	0	0	1	1
	0	1	0	0
	0	1	1	0
	1	0	0	1
	1	0	1	1
T	1	1	0	1
	1	1	1	0

J	K	Q
0	0	q
0	1	0
1	0	1
1	1	q'

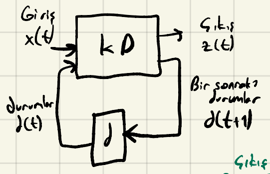
$Q = Jq' + K'q$

Ardışıl Devre Temelleri



Ardışıl Devreler

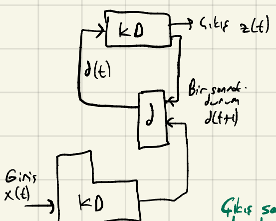
Mealy Makinesi



$z(t) = f(x(t), d(t))$ — çıkış durumlar ve o anki girişe bağlı

$d(t+1) = f(d(t), x(t))$

Moore Makinesi

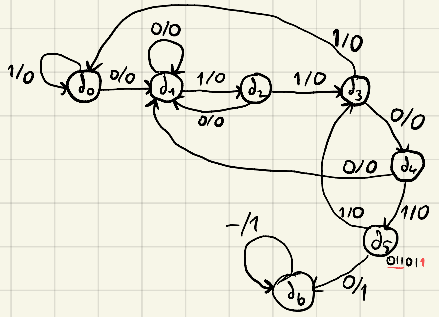


$z(t) = f(d(t))$ — Çıkış sadece duruma bağlı

$d(t+1) = f(x(t), d(t))$

Not: Senkron devreler clock'un frekansı girişin frekansından yüksek seçilir ki bilgi kaybı yaşanmasın.

Örnek: $\overrightarrow{011010}$ şifresini sonlu durum makinesi olarak çiziniz.



durum etiketi	durum d_2 d_1 d_0	giriş	durum$^+$ d_2^+ d_1^+ d_0^+	çıkış
0	0 0 0	0	0 0 1	0
0	0 0 0	1	0 0 0	0
1	0 0 1	0	0 0 1	0
1	0 0 1	1	0 1 0	0
⋮	⋮	⋮	⋮	⋮
6	1 1 0	0	1 1 0	1
6	1 1 0	1	1 1 0	1

Standart Tasarım Birimleri

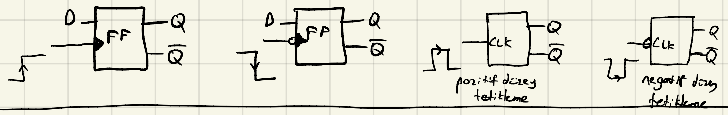
Ardışıl Devreler

Kombinezonsal

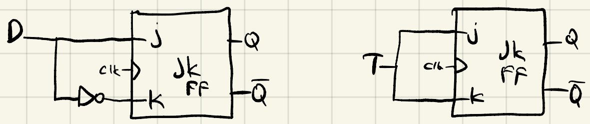
Lojik kapılarla Tasarım

Hafıza (durum)

- Flip-floplar (1 bit saklayıcı)
- Tutucular (Latches) 1 bit
- Sayıcı (Counter) x bit
- RAM
- ROM
- Ardışıl Devreler PSA

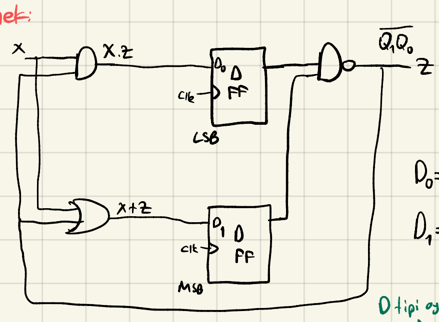


Örnek: JK flip flopu kullanarak D tipi ve T tipi flip flop yapınız.



D tipi FF   T tipi FF

Örnek:



Şekildeki devreyi analiz ediniz.

$D_0 = X.Z = X.(\overline{Q_1.Q_0})$

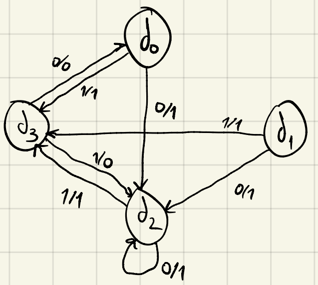
$D_1 = X + Z = X + (\overline{Q_1.Q_0})$

$Y = D$

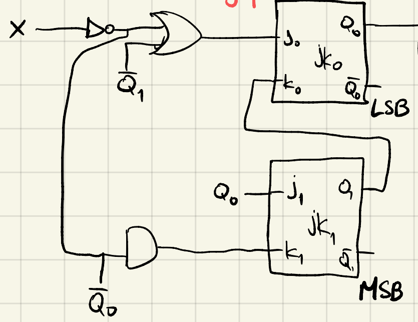
D tipi aynısını yansıtır.

Simgesel Durum	Şimdiki Durumlar Q_1 Q_0	FF girişleri x=0 D_1 D_0	FF girişleri x=1 D_1 D_0	Bir Sonraki Durumlar x=0 Q_1^+ Q_0^+	Bir Sonraki Durumlar x=1 Q_1^+ Q_0^+	Çıkış z
d_0	0 0	1 0	1 1	1 0	1 1	1
d_1	0 1	1 0	1 1	1 0	1 1	1
d_2	1 0	1 0	1 1	1 0	1 1	1
d_3	1 1	0 0	1 0	0 0	1 0	0

Şimdiki durum	Bir Sonraki Durum x=0	Bir Sonraki Durum x=1	Çıkış Z
d_0	d_2	d_3	1
d_1	d_2	d_3	1
d_2	d_2	d_3	1
d_3	d_0	d_2	0



Örnek: Analizini yapınız



$J_0 = \bar{x} + \bar{Q}_1$

$K_0 = Q_1$

$J_1 = Q_0$

$K_1 = \bar{x} \cdot \bar{Q}_0$

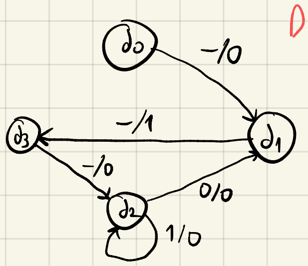
$Z = \bar{Q}_1 \cdot Q_0$

FF girişleri

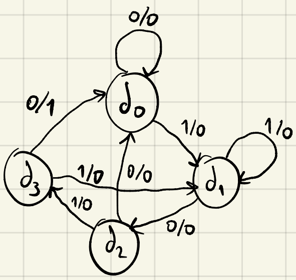
Simgesel Durum	Şimdiki Durumlar Q_1	Q_0	x=0 J_1	K_1	J_0	K_0	x=1 J_1	K_1	J_0	K_0	Sonraki durum x=0 Q_1^+	Q_0^+	x=1 Q_1^+	Q_0^+	Çıkış Z
d_0	0	0	0	1	1	0	0	0	1	0	0	1	0	1	0
d_1	0	1	1	0	1	0	1	0	1	0	1	1	1	1	1
d_2	1	0	0	1	1	1	0	0	0	1	0	1	1	0	0
d_3	1	1	1	0	1	1	1	0	0	1	1	0	1	0	0

Şimdiki Durum y (Q(t))	Bir Sonraki Durum (Q(t+1)) x=0	x=1	Çıkış z
d_0	d_1	d_1	0
d_1	d_3	d_3	1
d_2	d_1	d_2	0
d_3	d_2	d_2	0

Durum diyagramı



Soru: Girişi x, çıkışı z ile ifade edilen bir ardışıl devrenin girişine 1010 dizisi geldiğinde çıkışı 1 yapan aksi durumda 0 yapan devreyi D flip floplarıyla gerçekleyiniz.



Şimdiki Durum y	Bir Sonraki Durum Y x=0	x=1	Çıkış x=0	x=1
d_0	d_0	d_1	0	0
d_1	d_2	d_1	0	0
d_2	d_0	d_3	0	0
d_3	d_0	d_1	1	0

Simgesel Durum	Şimdiki Durum Q_1 Q_0	Giriş x	Bir sonraki durum Q_1^+ Q_0^+	FF_1 D_1	FF_0 D_0	Çıkış z
d_0	0 0	0	0 0	0	0	0
d_0	0 0	1	0 1	0	1	0
d_1	0 1	0	1 0	1	0	0
d_1	0 1	1	0 1	0	1	0
d_2	1 0	0	0 0	0	0	0
d_2	1 0	1	1 1	1	1	0
d_3	1 1	0	0 0	0	0	1
d_3	1 1	1	0 1	0	1	0

yY	00	01	11	10
D	0	1	1	0

D_1 için

x \ Q_1Q_0	00	01	11	10
0	0	1	0	0
1	0	0	0	1

$D_1 = \bar{Q}_1 Q_0 \bar{x} + x Q_1 \bar{Q}_0$

D_0 için

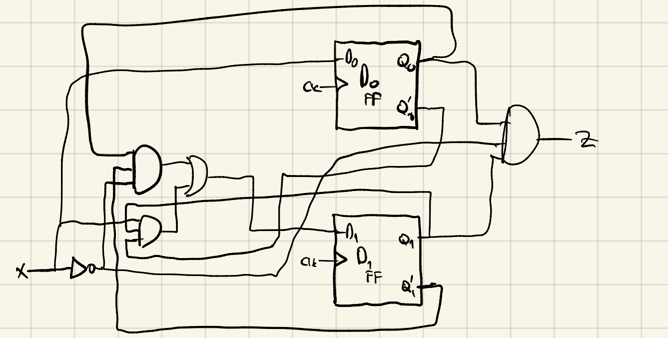
x \ Q_1Q_0	00	01	11	10
0	0	0	0	0
1	1	1	1	1

$D_0 = x$

Z için

x \ Q_1Q_0	00	01	11	10
0	0	0	1	0
1	0	0	0	0

$Z = \bar{x} Q_1 Q_0$



Soru: Hazır 4 bitlik paralel yükleme yapma özelliği olan Modülo-16 sayıcı kullanarak 3-14 arasında sayan bir sayıcı tasarlayın

Şimdiki Durum Q_3 Q_2 Q_1 Q_0	Sonraki Durum Q_3^+ Q_2^+ Q_1^+ Q_0	F
0 0 1 1	0 1 0 0	0
0 1 0 0	0 1 0 1	0
⋮	⋮	⋮
1 1 1 0	0 0 1 1	1