

Furkan EKİCİ - 2017555017

DIGITAL DESIGN

Converting Binary to Decimal

$$0 \quad 1 \quad , \quad 0 \quad 1 \\ \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\ 2^1 \times 0 \quad 2^0 \times 1 \quad , \quad 2^{-1} \times 0 \quad 2^{-2} \times 1 \Rightarrow 1,25$$

Converting Decimal to Binary

* Sırekli ikiye böl ve kalanları yaz. (Altınan İste).

Kalan

$$\begin{array}{r} 30/2 = 15 & 0 \\ 15/2 = 7 & 1 \\ 7/2 = 3 & 1 \\ 3/2 = 1 & 1 \\ 1/2 = 0 & 1 \end{array} \Rightarrow 1110$$

+ Virgülle olursa , 0,375

$$0,375 \times 2 = 0,750$$

$$0,750 \times 2 = 1,500 \Rightarrow 110.$$

$$1,500 \times 2 = 3,00 \\ 2 = 1 \text{ kalan}$$

! Bu lkez ostten
altta doğru.

Base 8 and 16

Oct

Hex

0 0

1 1

2 2

3 3

4 4

5 5

6 6

7 7

10 8

11 9

12 A

13 B

14 C

15 D

16 E

17 F

Octal to Binary

$$(2 \quad 6 \quad 1 \quad , \quad 3 \quad 5)_8 \rightarrow (010 \quad 110 \quad 001 \quad , \quad 011 \quad 101)_2$$

Her sayı 3 bitle yazılır.

Hex to Binary

$$(2 \quad 6 \quad 1 \quad , \quad 3 \quad 5)_{16} \rightarrow (0010 \quad 0110 \quad 0001 \quad , \quad 0011 \quad 0101)_2$$

Her sayı 4 bitle yazılır.

Binary to Octal

$$(101 \quad 011, \quad 010)_2 \rightarrow (5 \quad 3, \quad 2)_8$$

* 3'lu grulere at, her grubu rakamlandır.

$$(5 \quad 3, \quad 2)_8$$

Tüm kısımda base, virgülden sonrası
kısımda sona "0" koysabilirsin.

Binary to Hex

$$(0010 \quad 1011, \quad 0100)_2 \rightarrow (2 \quad B, \quad 4)_{16}$$

* 4'lu grulere at, her grubu rakamlandır.

$$(2 \quad B, \quad 4)_{16}$$

BOOLEAN

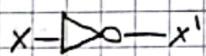
Basic Boolean Operations

x	y	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1



AND GATE

x	x'
0	1
1	0



NOT GATE

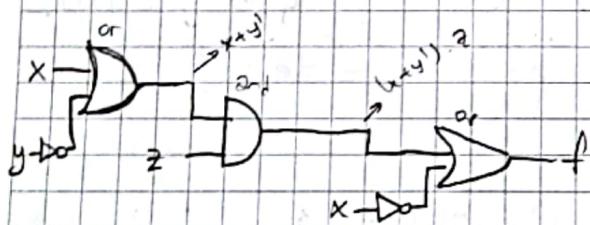
x	y	$x+y$
0	0	0
0	1	1
1	0	1
1	1	1



OR GATE

! İşlem öncelikleri: Parantez > Not > And > Or

example: $f(x,y,z) = (x+y') \cdot z + x'$



Truth Table:

Olası tüm giriş kombinasyonlarına göre
fikslarının oluşturduğu tablodur.

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

Truth table.

Boolean Algebra

$$\bullet x+0 = x$$

$$\bullet x+1 = 1$$

$$\bullet x+x = x$$

$$\bullet \underline{x+x' = 1}$$

$$\bullet x+y = y+x$$

$$\bullet x+(y+z) = (x+y)+z$$

$$\bullet x(y+z) = xy+xz$$

$$\bullet x+x'y = x \quad \begin{matrix} ? \\ (x \cdot 1 + x \cdot y \equiv x(y+1) \equiv x) \end{matrix}$$

$$\bullet x \cdot 1 = x$$

$$\bullet x \cdot 0 = 0$$

$$\bullet x \cdot x = x$$

$$\bullet x \cdot x' = 0$$

$$\bullet x \cdot y = y \cdot x$$

$$\bullet x \cdot (y \cdot z) = (x \cdot y) \cdot z$$

$$\bullet x+y \cdot z = (x+y) \cdot (x+z)$$

$$\bullet x \cdot (x+y) = x \quad \begin{matrix} \nearrow \\ (x+0) \cdot (x+y) \equiv x+y \cdot 0 \equiv x \end{matrix}$$

* Sadece solu öğren ve çarpı (\cdot) işaretinin yerine artı ($+$),
arti işaretinin yerine çarpı (\cdot), 0 ve bir (1) işaretinin yerine (0),
sıfır işaretinin yerine bir yazarak eşdeğerlik elde et!

De Morgan:

$$\bullet (x')' = x$$

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

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

Example: $x' \cdot y' + xy \cdot z + x'y$ sadeleştir.

$$x'(y'+y) + xy \cdot z$$

$$x' + xy \cdot z$$

$$\underbrace{(x'+x)}_1 \cdot (x'+yz)$$

$$\boxed{x'+yz}$$

Example: $f(x,y,z) = x(y'z' + yz)$ ise $f'(x,y,z) = ?$

$$(x(y'z' + yz))' \equiv x' + (y'z' + yz)' \equiv x' + (y'z') \cdot (y \cdot z)'$$
$$\equiv \boxed{x' + (y+z) \cdot (y'+z')}$$

Karnugh Maps:

Garpimlarin toplamı (sum of product) seklinde en sade ifadeyi elde etmemizi saglayan tablolardir.

Example:

$$x'y' + x'y \quad \text{En sade hali nedir?}$$

x' depismed
 $y' \rightarrow y$ oldu
depsit
(Depismeden, Y22)

	y'	0	1
x'	$x'y'$	$x'y$	
0	xy'	xy	

$$x'y' + x'y \equiv x'$$

Proof: $x'(\underbrace{y' + 1}_1) \equiv x' \cdot 1 \equiv x'$

★ ikinin kuvvetleri seklinde grup olustur. (2,4,8)

Example: $x'y' + x'y + xy$

	y'	0	1
x'	$x'y'$	$x'y$	
0	xy'	xy	

sabit olan x'

sabit olan y

$$x'y' + x'y + xy \equiv \boxed{x' + y}$$

Example: $x'y'z + x'y z$

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

x' deşifremedi.

$y' \rightarrow y$ oldu.

z' deşifremedi.

$$x'y'z + x'y z = x'z$$

(tek grup olduğu için)
(tek garipim var)

! Sıralayın, karıştırın! (00, 01, 11, 10)

Example: $x'y'z' + x'y'z + x'y z' + x'y z$

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

Köşelerde de
grup oluşturabilir.

ilk satırda = $x'z'$ (sabitolar)

ikinci satırda = xz' (sabitolar)

$$x'z' + xz'$$

$$\overbrace{z'(x' + x)}^1 = z' \cdot 1 = \boxed{z'}$$

* Tek grup olduğu için

tek garip olmali 1. garip

$(x'z')$ gibi bir ifade ($xz' + yx$),
gibi deşil 2. garip

! Satır satır baktığınız
için iki garip var.

Sadece z' 'nın deşifre
deşifremiyor.

Example: Doğruluk tablosu verilen bir donkleme Karnugh map

yardımı ile bulma.

X	Y	Z	F
0	0	0	0 m_0
0	0	1	1 m_1
0	1	0	0 m_2
0	1	1	0 m_3
1	0	0	0 m_4
1	0	1	1 m_5
1	1	0	1 m_6
1	1	1	1 m_7

1 olan
yerleri
bul.

Buraya
yerlestir.

F için

X\YZ	00	01	11	10
0	m_0	m_1	1	m_2
1	m_4	m_5	m_7	m_6

$y'z + xy$

* Karnugh map'taki m_k depekerini bulmak için depekeri (x, y, z) binary olarak yazarıp topla. Örneğin m_7 (binary 111)'dır.

* 4 depeşken olduğunda.

wx\yz	00	01	11	10
00	m_0	m_1	m_3	m_2
01	m_4	m_5	m_7	m_6
11	m_{12}	m_3	m_{15}	m_{14}
10	m_8	m_9	m_{11}	m_{10}

Burda da yine

közelce grup olabilir,
(m_0, m_2, m_4, m_8)

$$x'z' + xy'z$$

!	0	1	0	1
	0	1	1	0

Bu da doğru

$$y'z + yz' + xy$$

0	1	0	0
0	1	1	1

Bu da doğru

$$y'z + yz' + xz$$

istediğiniz yere grupperlendirilebiliriz.

Bunu



Temel Asal Çarpan! (Araştır) ?

$wx'y'z'$	00	01	11	10
00	1	1		
01	1	1		
11		1	1	
10			1	1

$$w'y' + x'y'z + wxz + wxy$$

Eğer grupta, başka bir gruba dahil olmamış bir minterm varsa ongruba temel asal çarpan denir.

Sadeleşmiş ifadede bulunacak olanlar kırmızıyla çizilecek oldığından bunlara temel asal çarpan denir.

I don't care

İmkânsız olan durumlarda cikis'a "X" yazılır. Gruplama yaparken isimizde gelirse 1, isimizde gelirse 0 alırız.

Example:

00	01	11	10
00	1	0	0
01	1	X	0
11	0	X	1
10	1	0	X

$$x'z' + w'x'y' + wxy$$

Dikkat!

- 1-) Tablonun sırasına dikkat et (Tek değişken $\rightarrow 0, 1$)
(iki değişken $\rightarrow 00, 01, 11, 10$)
- 2-) Mungkin olduğunda büyük grup oluştur.
- 3-) Mungkin olduğunda 22 grub kullan.

Example: (I don't care) Seven Segment Display

$f \overline{f}$
 $\underline{e} \overline{e}$
 $\underline{d} \overline{d}$

"e" nin yanıp yanmama durumu için;

A	B	C	D	e
0	0	0	0	1
1	0	0	1	0
2	0	0	1	0
3	0	0	1	0
4	0	1	0	0
5	0	1	0	0
6	0	1	0	1
7	0	1	1	0
8	1	0	0	1
9	1	0	0	0
X	.	.	.	X
X	.	.	.	X
X	.	.	.	X
X	.	.	.	X
.

CD
AB	00	01	11	10	11
00	1	0	0	0	1
01	0	0	0	0	1
11	X	X	X	X	1
10	1	0	X	X	X

$$CD' + B'D'$$

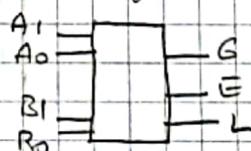
Basic Circuit Design

- 1-) Devrenin girişlerini ve çıkışlarını belirle.
- 2-) Doğruluk tablosu veya bool cebri ile fonksiyonu ifade et.
- 3-) Karnugh veya bool cebri ile sadeleştir.
- 4-) Devrede göster.

Example: 2-bitlik, sayıları karşılaştırın devreyi tasarla.

Step - 1: Giriş - çıkış belirle.

Inputs = 00, 01, 10, 11 (4 tane)



OUTPUT = Greater (G), Equal (E), Lesser (L)

Step -2 : Dögrülük tablosu yap.

A ₁	A ₀	B ₁	B ₀	G	E	L
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	0	0	0
1	1	1	1	0	0	0

Step -3 : Karnaugh map ile sadelestir

B ₁ B ₀				B ₁ B ₀				B ₁ B ₀			
A ₁ A ₀		A ₁ A ₀		A ₁ A ₀		A ₁ A ₀		A ₁ A ₀		A ₁ A ₀	
00	01	11	10	00	01	11	10	00	01	11	10
0	0	0	0	0	1	0	0	0	1	1	1
0	1	1	0	0	0	1	0	0	0	1	1
1	1	1	0	0	0	0	1	0	0	0	0
1	0	1	0	1	0	0	0	1	0	0	0
1	0	1	1	0	0	1	0	0	0	1	0
1	1	0	0	1	0	1	0	0	0	0	0
1	1	0	1	0	1	1	0	0	0	0	0
1	1	1	0	0	1	0	1	0	0	0	0
1	1	1	1	0	1	0	1	0	0	0	0

$$G(A_1, A_0, B_1, B_0) =$$

$$A_1 A_0 B_0' +$$

$$A_0 B_1' B_0' +$$

$$A_1 B_1'$$

$$E(A_1, A_0, B_1, B_0) =$$

$$A_1' A_0' B_1' B_0' +$$

$$A_1' A_0 B_1' B_0 +$$

$$A_1 A_0 B_1 B_0 +$$

$$A_1 A_0' B_1 B_0'$$

$$L(A_1, A_0, B_1, B_0) =$$

$$A_1' A_0' B_0 +$$

$$A_0' B_1 B_0 +$$

$$A_1' B_1$$

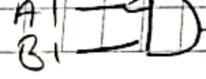
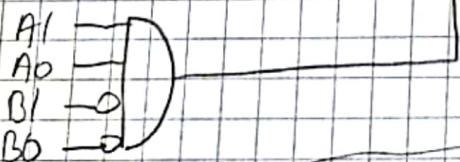
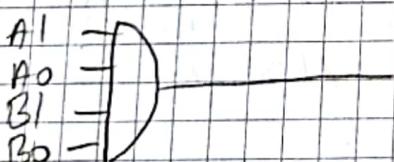
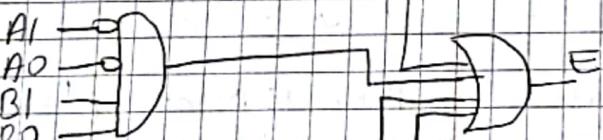
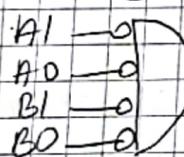
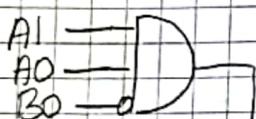
Step -4: Tasarla

$$G = A_1 A_0 B_0' + A_0 B_1' B_0' + A_1 B_1'$$

$$E = A_1' A_0' B_1' B_0' + A_1 A_0 B_1' B_0 + A_1 A_0 B_1 B_0 + A_1 A_0' B_1 B_0'$$

$$L = A_1' A_0' B_0 + A_0' B_1 B_0 + A_1' B_1$$

* Bunları yaptıktan sonra
da test et!



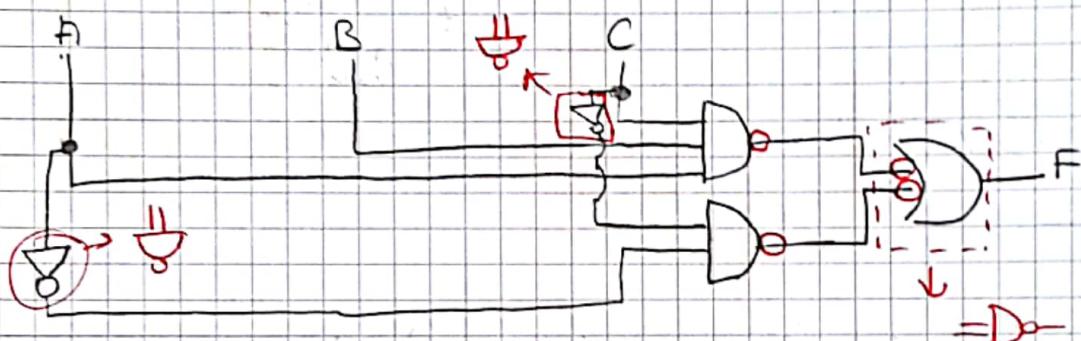
NAND GATE

Example: $F = ABC + A'B'C' + A'B'C'$ Sadece NAND kapıları

Kullanarak bir devre tasarlayın.

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

$$A'C' + ABC$$



Girişlerinin hepsi 0 olursa OR kapıları NAND'e dönüştür.

And'in sonuna, Or'un başına NOT koymaya gerek

Maxterm:

A	B	C	F	F'
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	0	1
1	1	1	0	1

Maxterm için F si 0 olanları ($F'=1$) al.
 A, B, C' 'nin 0 olması maxtermde olumlu bir şey
yani $A=0$ ise A yazılır.
 $A=1$ ise A' yazılır.

Maxterm:

$$\rightarrow A + B + C$$

$$\rightarrow A + B' + C'$$

$$\rightarrow A' + B' + C$$

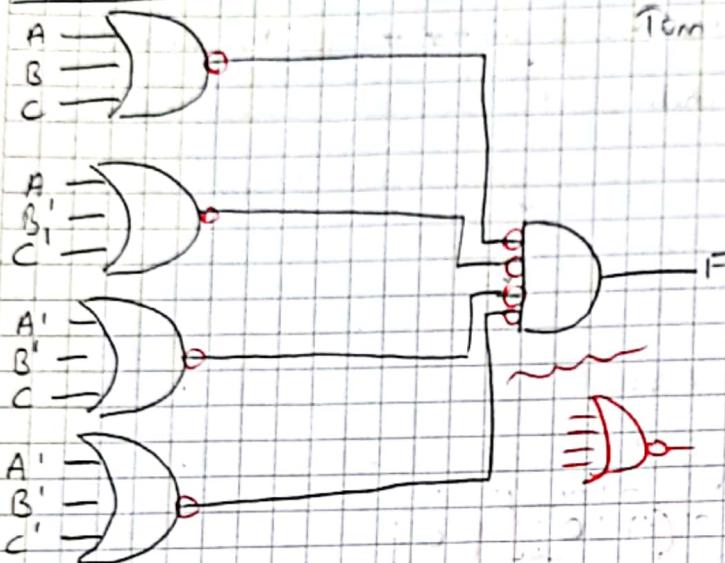
$$\rightarrow A' + B' + C'$$

$$(F')' = (A+B+C) \cdot (A+B'+C') \cdot (A'+B+C) \cdot (A'+B'+C')$$

$$F = \underbrace{\text{Toplamların}}_{\text{Toplamların}} \underbrace{\text{Harçımı}}_{\text{Harçımı}}$$

! Maxterm kavramı, sadece OR kullanarak devre yapınız.
pibi sorularda kolaylık sağlar.

Example:



Tüm kapılıları NOR ile deşifre edilebilir



AND before
OR sonuna
NOT
lçey.

XOR GATE:

A B | F Girisler aynıysa "0" farklıysa "1".

0	0	0
0	1	1
1	0	1
1	1	0

$$A \oplus B = A'B + AB'$$

Üçlüde tek sayıda 1 veya çıkış 1 diğer durumlarda 0.

A	B	C	F \rightarrow $F = A \oplus B \oplus C$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Fan-out
Fan-in.

XNOR GATE:

A	B	F
0	0	1
0	1	0
1	0	0
1	1	1

$$(A \oplus B)' = A'B' + AB$$

0	0	1
0	1	0
1	0	0
1	1	1

Soru: Kırmızı (K), Sarı (S), Yeşil (Y) bir trafik işaretindeki lambalar olsun. Bu sistemdeki hatalı durumları Sezip gizinde "1" işaret on devre tasarılayın
(Hatalı Durumlar: Herhangi bir lambanın yanmaması, veya birden fazla lambanın aynı anda yanması.)

A-) Sadık NAND ile tasarıla.

B-) Sadık NOR ile tasarıla.

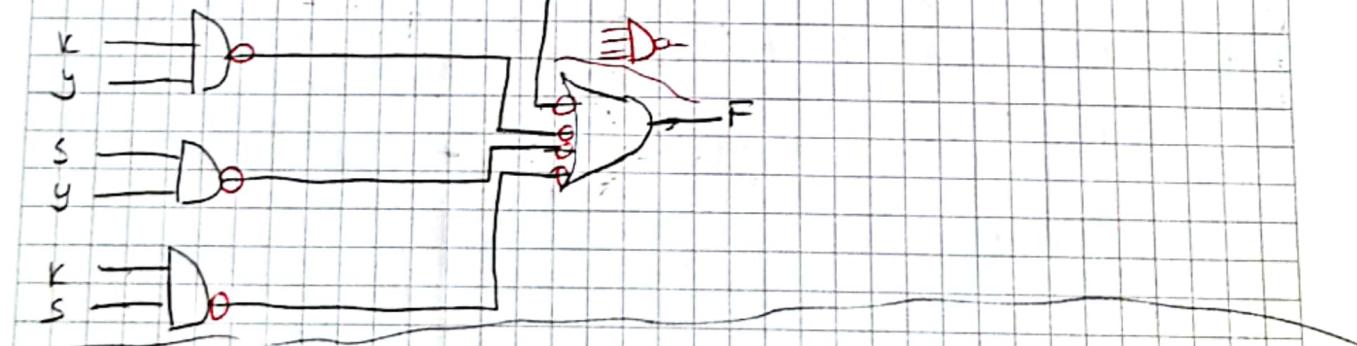
K	S	Y	H
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

K	S	Y	00	01	11	10
0	0	1	1	0	1	0
1	0	0	0	1	1	1

$$H = (K'S'Y') + (KY) + (SY) + (KS)$$

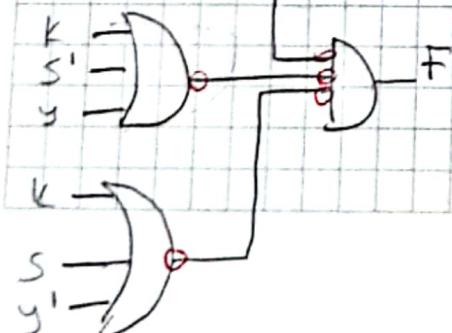


HEPSİ NAND.



HEPSİ NOR,

NOR yaparken 0'ları dikkate al
 (karmıfta)



$$F = (K' + S + Y) \cdot (K + S' + Y) \cdot (K + S + Y')$$

$x' \cdot y' \cdot z'$

011

Example:

X	Y	Z	F	F'
0	0	0	1	0
0	0	1	1	0
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

$$F = \sum(0, 1, 3, 4, 7)$$

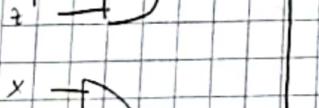
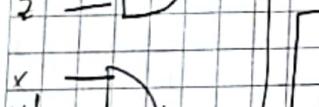
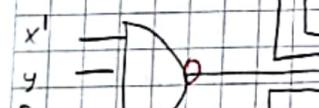
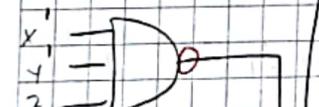
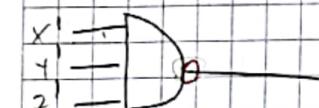
$$F' = \sum(2, 5, 6)$$

$$F = \prod(2, 5, 6)$$

$$F' = \prod(0, 1, 3, 4, 7)$$

Sadece NAND ile.

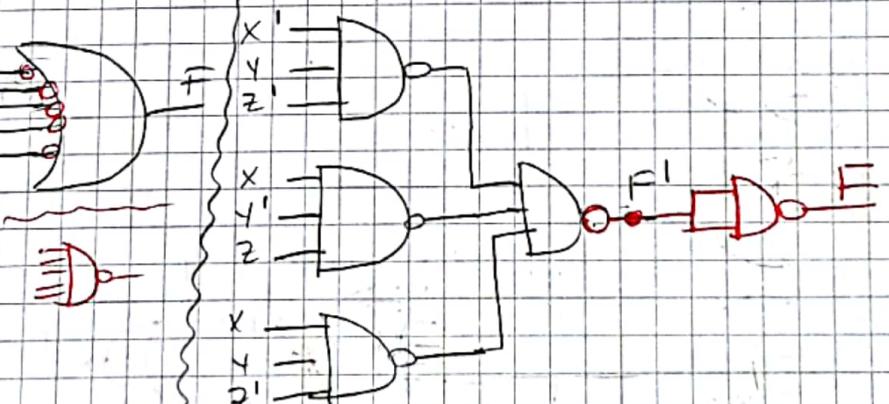
$$F = x'y'z' + x'y'z + x'yz + xy'z' + xyz$$



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

$$(F')' = (x'y'z')' \cdot (xy'z)' = (xyz')'$$

Hepsinin NAND oldugu.



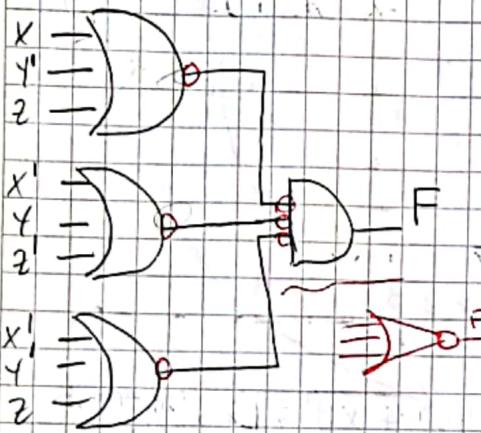
$$(A, A')'$$

$$(A, A'')$$

DEUAM!

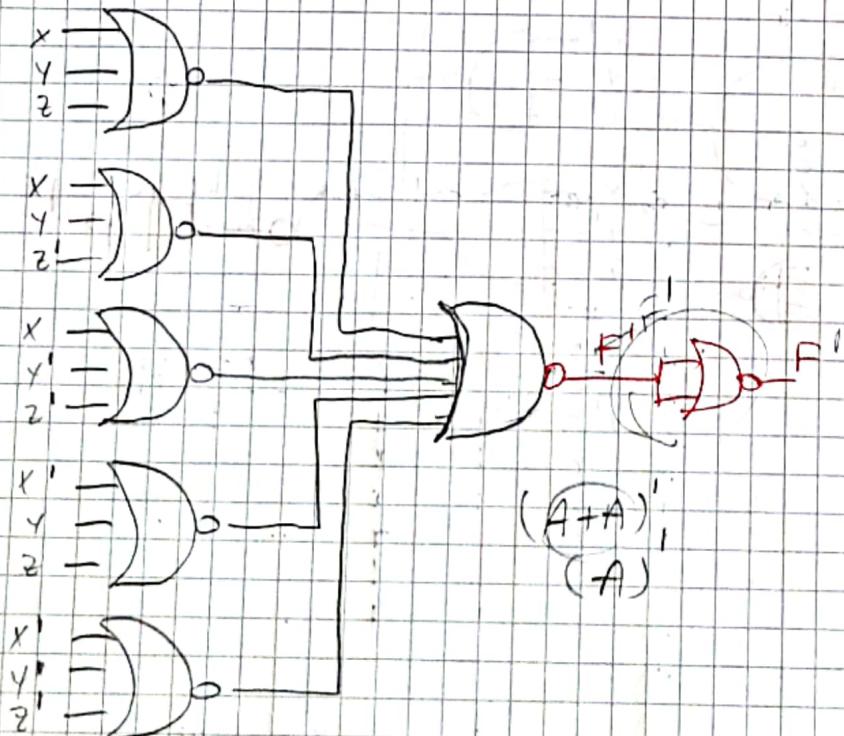
\int ADECE NOR ile

$$F = \overline{\Pi}(2, 5, 6) \Rightarrow F = (x+y+z) \cdot (x'+y+z') \cdot (x'+y'+z)$$



$$\pi' = \Pi(0, 1, 3, 4, 7)$$

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



DECODERS

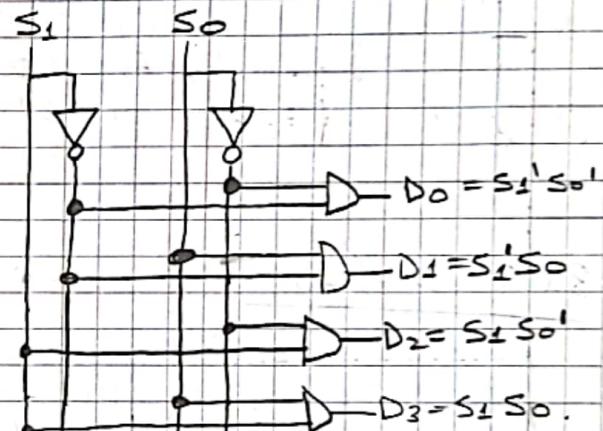
Kodlanmış bilgileri anlayabilen kodlardan gelenen deurelere kod çözücü (decoder) denir.

*) n giriş varsa 2^n çıkış vardır. ($2-4$, $3-8$ gibi).

Örneğin; S_1, S_0 giriş, D_3, D_2, D_1, D_0 ise çıktılarımız olsun.

S_1	S_0	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

(Dögrülük Tablosu)



Enable inputs.

Decoderde EN (enable) ucuna 1 deperi verilmezse çıkışlı hep 0 üretir.

EN	S_1	S_0	D_0	D_1	D_2	D_3
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	0	1	0	0
1	0	1	0	0	1	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

Gördüğün gibi
timlik 12 0.

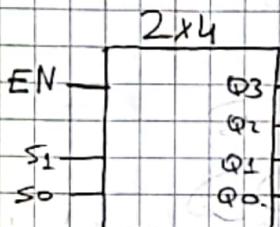
Kirmizi işaretli yerin tamamına sunuda yazabildik:

EN	S_1	S_0	D_0	D_1	D_2	D_3
0	X	X	0	0	0	0

$\neg(\neg S_0 \wedge \neg S_1 \wedge \neg S_2 \wedge \dots \wedge \neg S_n)$

* $S_0, S_1, S_2, \dots, S_n$ 'in 0 ve 1 ile girişleri yapılıp, onluk tabanda sonuçlar elde ediliyor. (örneğin $S_0=1, S_1=0, S_2=1$ olsun. 0 zaman $(101)_2 = (5)_{10}$ olduğundan Q5 galisir).

2x4 Decoder



$$\begin{aligned}Q_3 &= S_1 \cdot S_0 \\Q_2 &= S_1 \cdot S_0' \\Q_1 &= S_1' \cdot S_0 \\Q_0 &= S_1' \cdot S_0'\end{aligned}$$

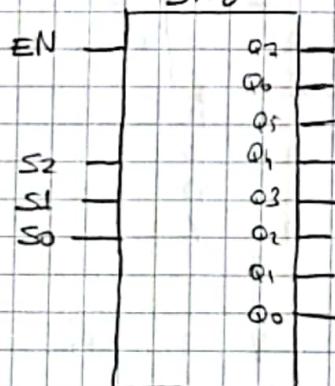
3x8 Decoder

S_2	S_1	S_0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

3x8 Decoder

$$(7)_{10} = (11111)_2$$

$$\begin{aligned}Q_7 &= S_2 \cdot S_1 \cdot S_0 \\Q_6 &= S_2 \cdot S_1 \cdot S_0' \\Q_5 &= S_2 \cdot S_1' \cdot S_0 \\Q_4 &= S_2 \cdot S_1' \cdot S_0' \\Q_3 &= S_2' \cdot S_1 \cdot S_0 \\Q_2 &= S_2' \cdot S_1 \cdot S_0' \\Q_1 &= S_2' \cdot S_1' \cdot S_0 \\Q_0 &= S_2' \cdot S_1' \cdot S_0'\end{aligned}$$



* Decodeler minterm tıretici olarakta biliniir.

Örnek: Decoder kullanarak 3 tane 1 bitlik sayıyi toplayan bir devre tasarlayınız.

→ Öncelikle doğruluk tablosu yapmalıyız. (Sayılarımıza x, y, z olsun)

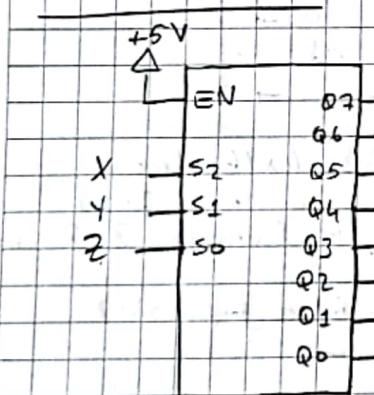
X	Y	Z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$C(x,y,z) = (3, 5, 6, 7)$$

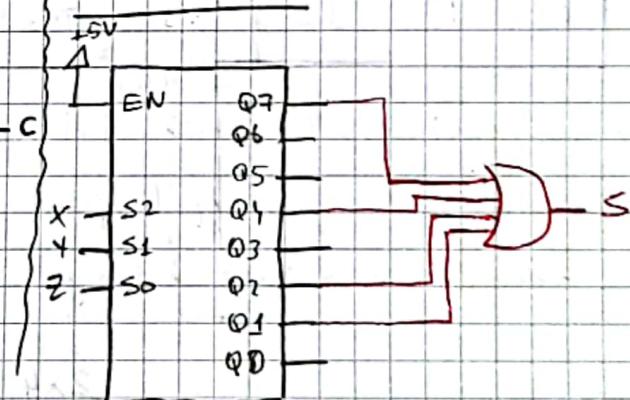
$$S(x,y,z) = (1, 2, 4, 7)$$

* Decoder = mintermlerin toplamı

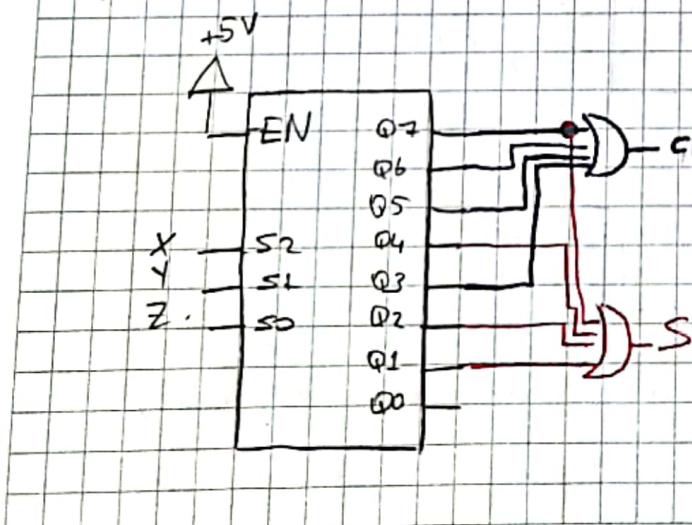
C devresi



S devresi



Veya iki devreyi birlestirebiliriz :

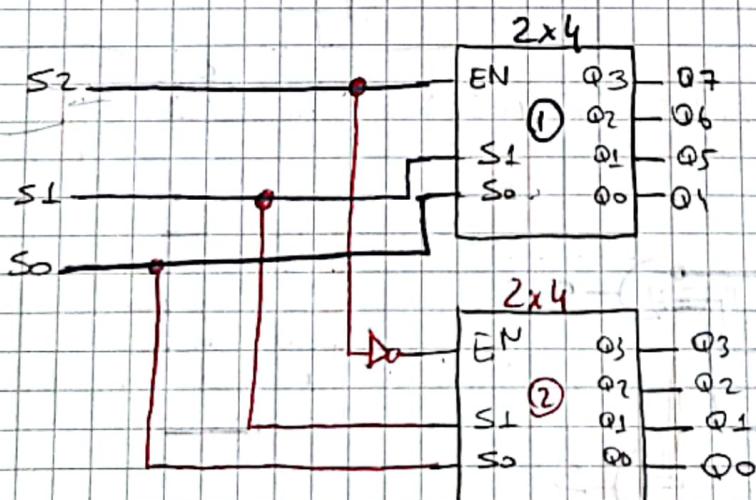


3x8 Decoder tasarlamak

2 adet 2x4 decoder ile bir adet 3x8 decoder tasarlanabilir. İlk 2x4 decoderimiz Q₀-Q₃ arası çıkışları, ikinci 2x4 decoder ise Q₄-Q₇ arası çıkışları sağlamak için kullanılabilir.

S ₂	S ₁	S ₀	Q ₀	Q ₁	Q ₂	Q ₃	Q ₄	Q ₅	Q ₆	Q ₇
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	0	1
1	1	1	0	0	0	0	0	0	0	1

Yapılan dec'in (2x4) çıkış sayısına göre (4)
 Yapılan dec'in (2x4) çıkış sayısına göre (4)
 Yapılmış isteniydi 4'ü grupla ayıracaktı. Bunada 3x8 dec'
 2x4 dec ile yapmamız isteniyor. Nine 4 16 grupta yapacağınız.
 2x4 dec ile yapmamız isteniyor. Nine 4 16 grupta yapacağınız.

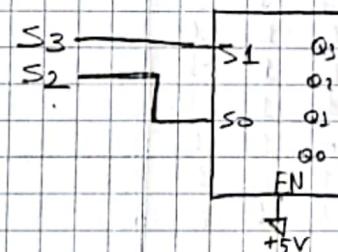


2. dec'de istenilen tabloda kırmızının üstündeki alan,
 1. dec'de istenilen tabloda kırmızının altındaki alan
 yapılmıştır.

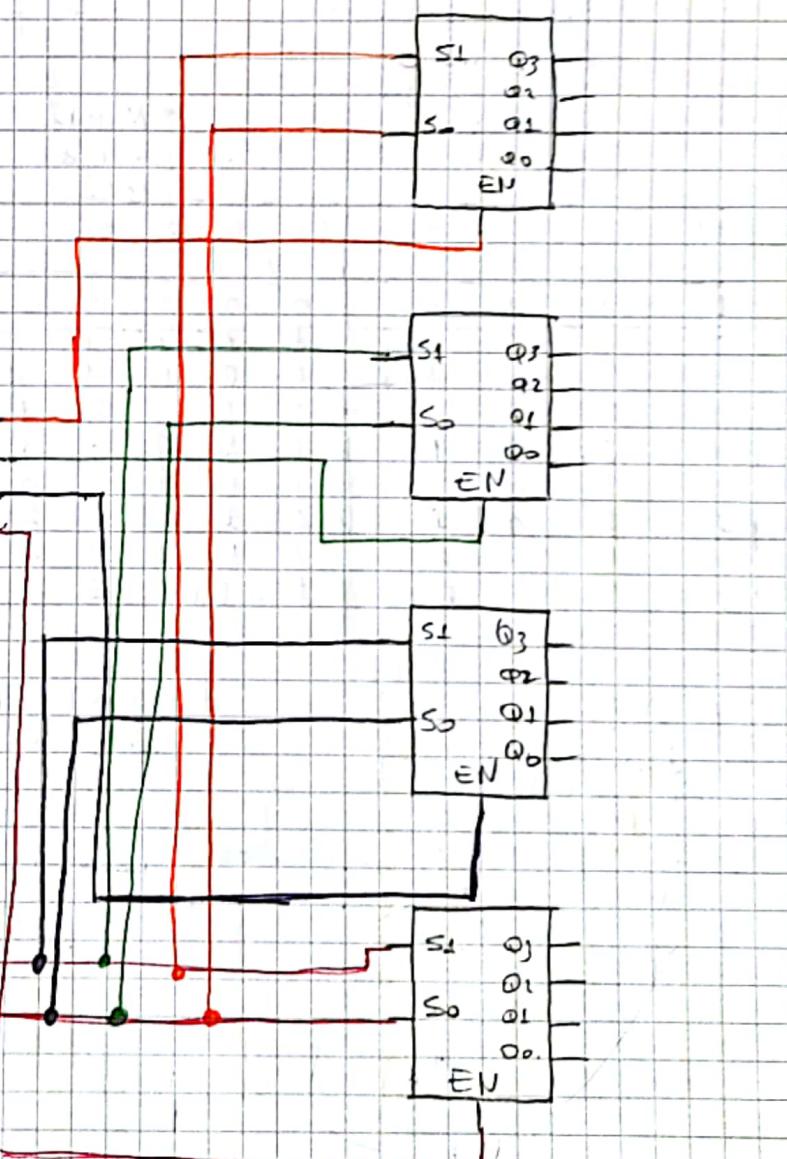
4x16 Decoder Tasarlamak

S_3	S_2	S_1	S_0
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	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

(4x16 dec dörtlük)
Tablosu



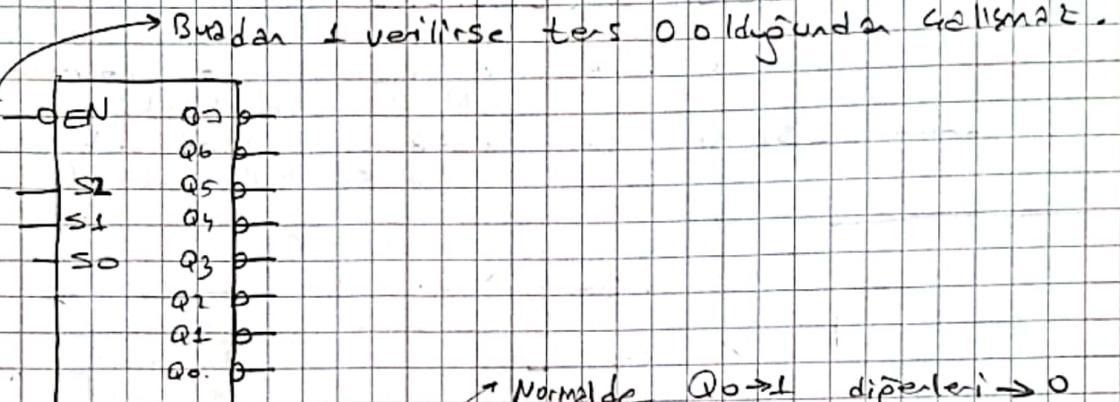
Buradaki mantık S_1 ve S_0 'ı
her decoderde (4 tane) kullanıp -
 EN değeri bir black box bir decode ile.
 S_3 ve S_2 kullanarak 4 dec'e de
döğütmektedir.



* Eğer bu devre (4x16) 3x8 dec ile yapılmak istense.
2 dec (3x8) kullanılırdu ve EN kısımlarına birlikte S_3 'ün
kendisi diperde S_3' bağlanırdu ve 16 çıkış elde
edilirdi.

Active-Low Decoder

* Decoderi muktemekin çarpımı olarak kullanabiliriz.



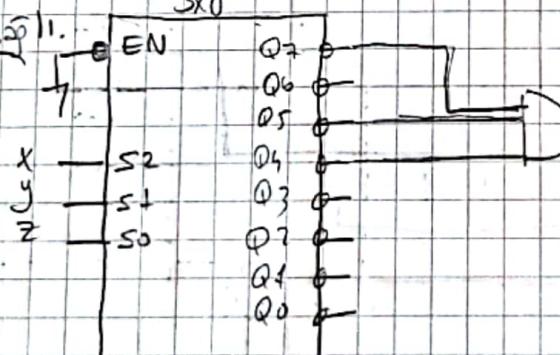
EN	S ₂	S ₁	S ₀	Q ₀	Q ₁	Q ₂	Q ₃	Q ₄	Q ₅	Q ₆	Q ₇
1	X	X	X	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1
0	1	1	0	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	0

Örneğin :

3x8

$$f(x, y, z) = \prod M(4, 5, 7) \text{ olsun.}$$

Toppa tablosu:



$$f = \sum m(0, 1, 2, 3, 6)$$

$$f' = \sum (4, 5, 7) = m^4 + m^5 + m^7$$

$$(f')' = (m^4 + m^5 + m^7)'$$

$$f = m^4 \cdot m^5 \cdot m^7'$$

$$\Rightarrow f = M_4 \cdot M_5 \cdot M_7$$

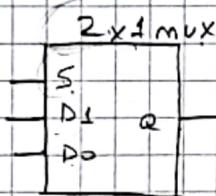
$$f = \sum m(0, 1, 2, 3, 6), \quad f' = \sum m(4, 5, 7)$$

$$f = \prod M(4, 5, 7)$$

$$f' = \prod M(0, 1, 2, 3, 6)$$

Multiplexers :

N-tane giriş içerişinden ($D_0, D_1, D_2, D_3 \dots$) seçme ucuları ($S_0, S_1 \dots$) yardımıyla veri seçer ve bir tane çıkış üretir. $\rightarrow n$ adet seçme ucu varsa. 2^n adet giriş vardır.



$$Q = S'D_0 + S \cdot D_1$$

Eğer $S=0$ ise D_0 aktifdir.

Eğer $S=1$ ise D_1 aktifdir.

(Formülle göre tablo)

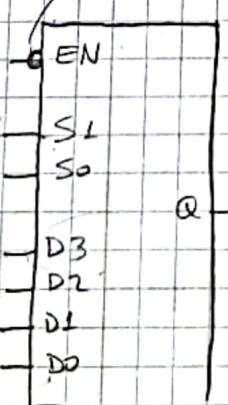
(Aşağından dolayı)

S	D1	D0	Q
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

S	Q
0	D_0
1	D_1

4x1 MUX :

Teslendirici var.

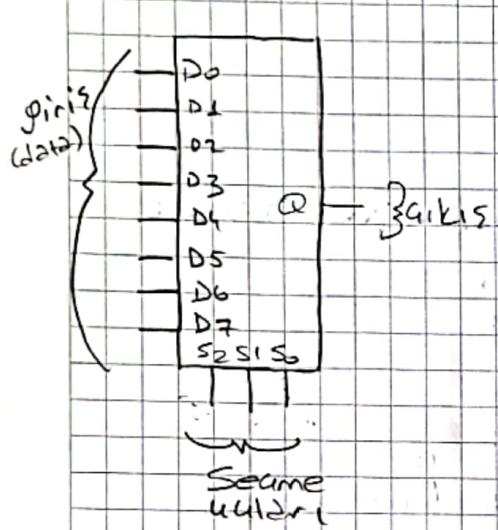


EN'	S1	S0	Q
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	X	X	1

EN ucunda teslendirici vardır. 0 verilince doğrudan aktarılır.

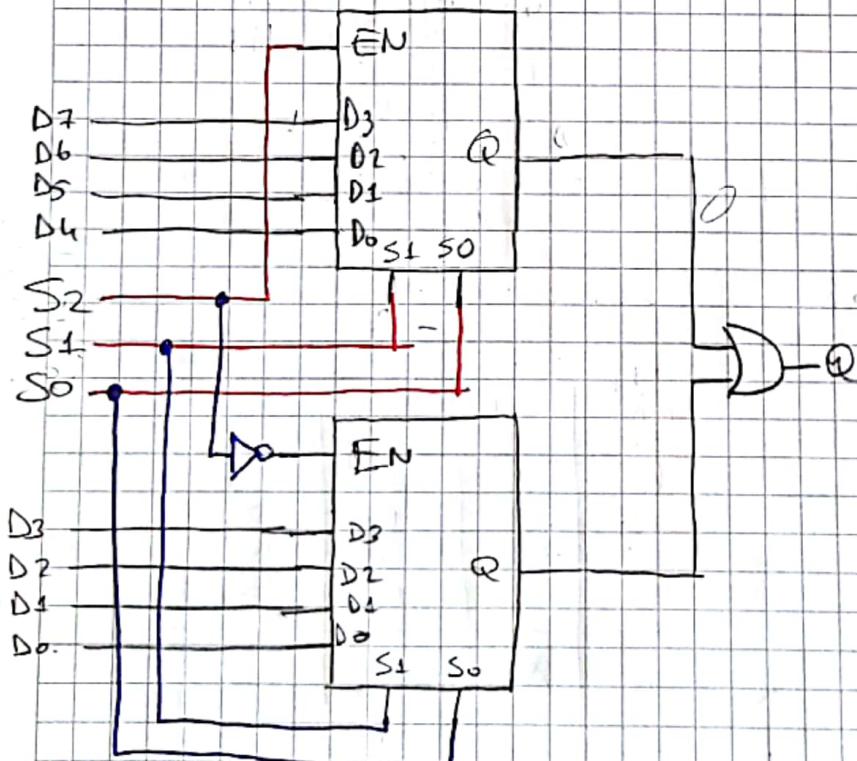
1 verilirse sonu4 hep 1 aktar.

8x1 mux:



S_2	S_1	S_0	Q
0	0	0	D_0
0	0	1	D_4
0	1	0	D_2
0	1	1	D_3
1	0	0	D_1
1	0	1	D_5
1	1	0	D_6
1	1	1	D_7

Soru: 4x1 mux kullanarak 8x1 mux elde et.



Örnek: $f(x_1, x_2, x_3) = \sum m(1, 2, 6, 7)$ fonksiyonunu muxiləf təsərləyin.

Öncelikle doğruluk tablosu oluştur.

Eğer 8×1 muxilə dəstənmək istiyorsak ki;

X	Y	Z	f		
0	0	0	0	1	D7
0	0	1	1	2	D6
0	1	0	1	2	D5
0	1	1	0	4	D4
1	0	0	0	3	D3
1	0	1	0	2	D2
1	1	0	1	6	D1
1	1	1	1	7	D0

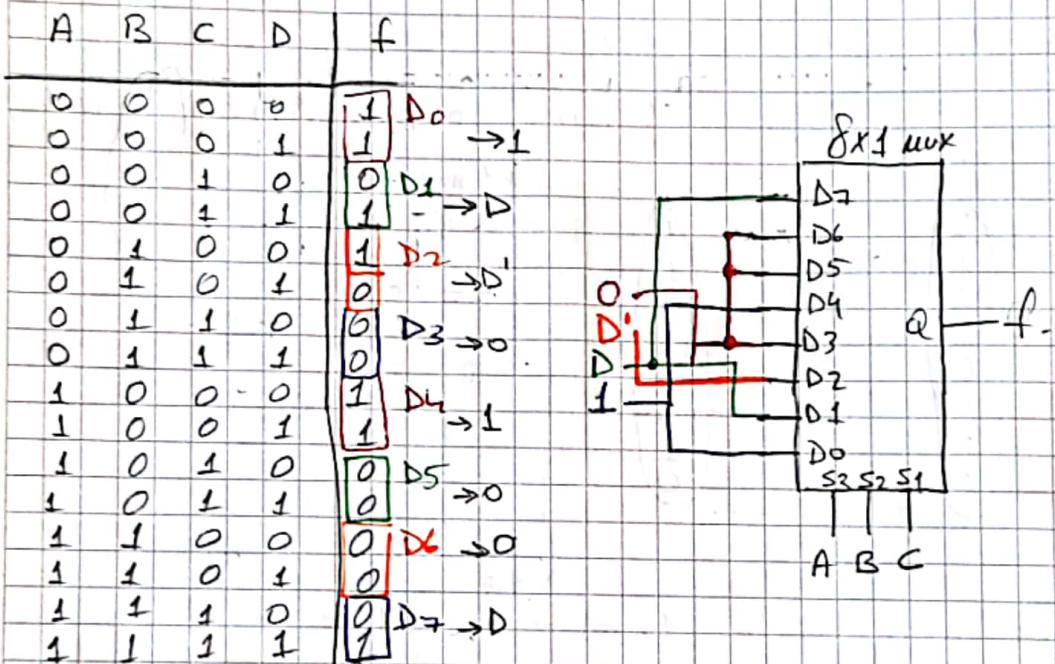
Aşağıdaki 0. elemanı Do'ya 1. elemanı Di'ye aktardık.

Eğer txt MUX ile oluşturmak isteseydik

X	Y	Z	f	$\bar{z} \cdot \bar{n} \cdot \bar{i} \cdot \bar{s} \cdot \bar{l}$
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	1	1
1	1	1	1	1

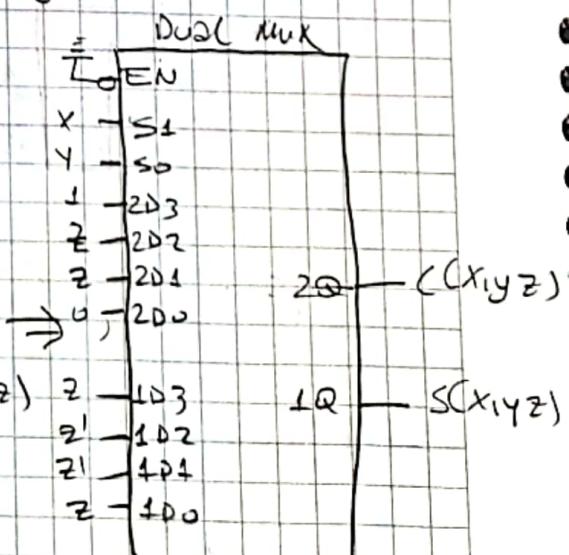
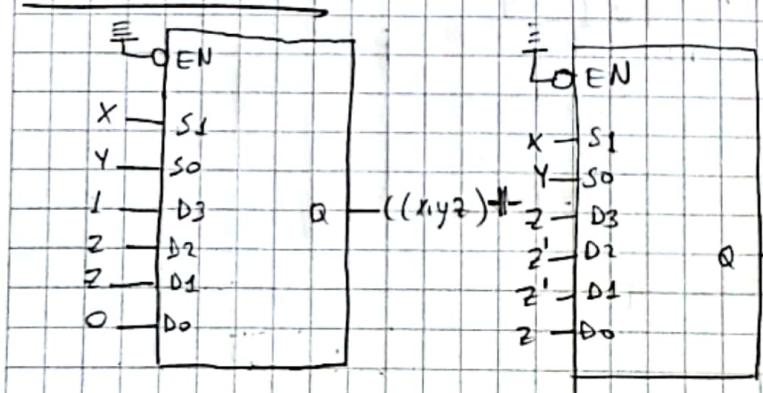
→ Bu kez ikinci grupluyoruz f'i.

Örnek: $F(A, B, C, D) = \sum(0, 1, 3, 4, 8, 9, 15)$ fonksiyonunu
8x1 mux ile tasarlayınız.



④ Eğer 16x1 mux ile tasarlanıysa, sadece 0 ve 1 ları (f' teli) yerine koyması beklenir. 8x1 dediği için 2'li gruplar halinde ele alındır.

Dual Mux (Carry-Sum) (3 tane 1 bitlik)



ADDITION AND MULTIPLICATION

Sayı toplamak, çarpmak ve çıkarmak için kullanılan devreler.

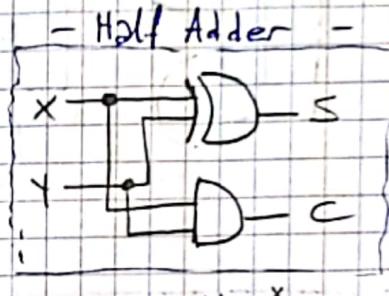
Örneğin 2 tane 1 bitlik sayıyı birbirine ekleyelim. ($x \cdot y$)

x	y	c	s	sum
0	0	0	0	
0	1	0	1	
1	0	0	1	
1	1	1	0	

$$c = x \cdot y$$

$$s = x'y + xy'$$

$$\Rightarrow s = x \oplus y$$



C iin

x/y	0	1
0		
1	1	

$$x \cdot y$$

S iin

x/y	0	1
0		
1	1	1

$$x'y + xy'$$



Adding Three Bits

x	y	Cin	$Cout$	S	$S = \Sigma m(1, 2, 4, 7)$
0	0	0	0	0	$= x'y'cin + x'y'cin' + xy'cin' + xy.cin$
0	0	1	0	1	$= x'(y'.cin + y.cin') + x(y'.cin + y.cin)$
0	1	0	0	1	$= x'(y \oplus cin) + x \cdot (y \oplus cin)'$
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	<u>$= x \oplus y \oplus cin$</u>

$$= x'y'cin + x'y'cin' + xy'cin' + xy.cin$$

$$= x'(y'.cin + y.cin') + x(y'.cin + y.cin)$$

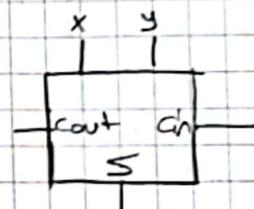
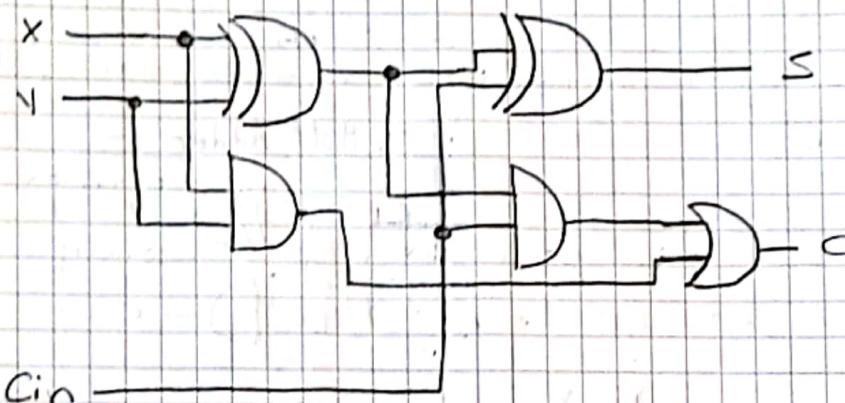
$$= x'(y \oplus cin) + x \cdot (y \oplus cin)'$$

$$= \boxed{x \oplus y \oplus cin}$$

$$Cout = \Sigma m(3, 5, 6, 7) = x'y.cin + xy'.cin + xy.cin' + xy.cin$$

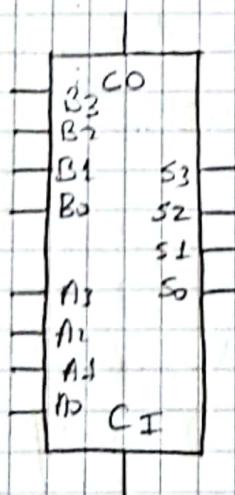
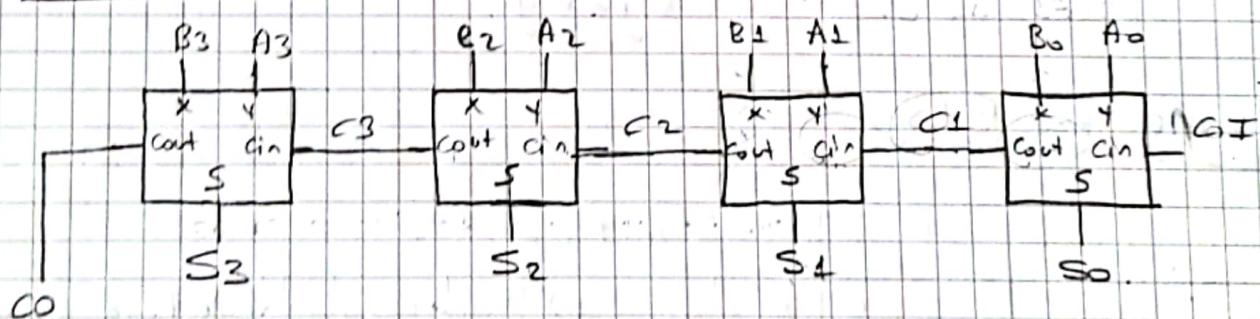
$$= Cin(x'y + xy') + xy \underbrace{(cin' + cin)}_{1} = \boxed{Cin(x \oplus y) + xy}$$

$$S = X \oplus Y \oplus \text{Cin} \quad C = \text{Cin}(X \oplus Y) + X \cdot Y$$



Full Adder

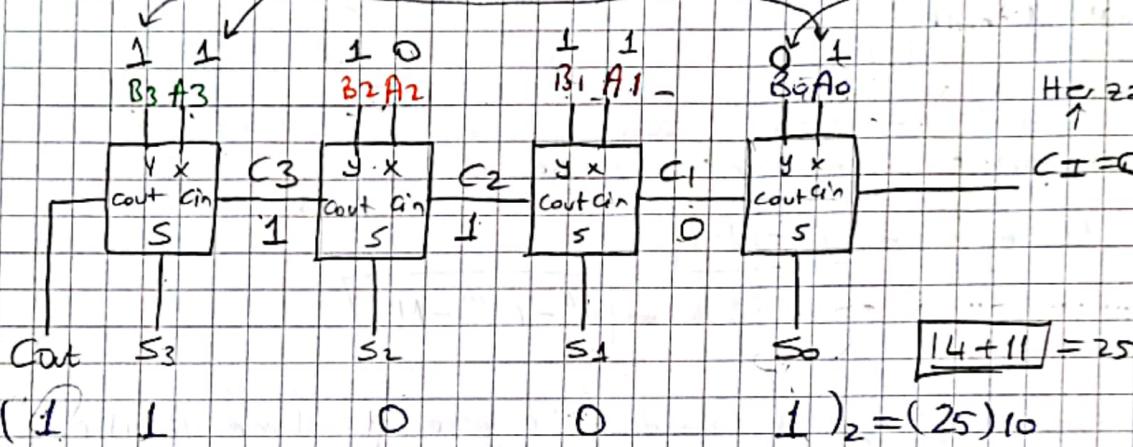
4 bit Adder



Yukarıdaki devre.

Örnek: 2 tane 4 bitlik sayıyi (11 ve 14) birbirine ekleyen devreyi çiziniz.

$$A = (11)_{10} = (1011)_2, \quad B = (14)_{10} = (1110)_2$$

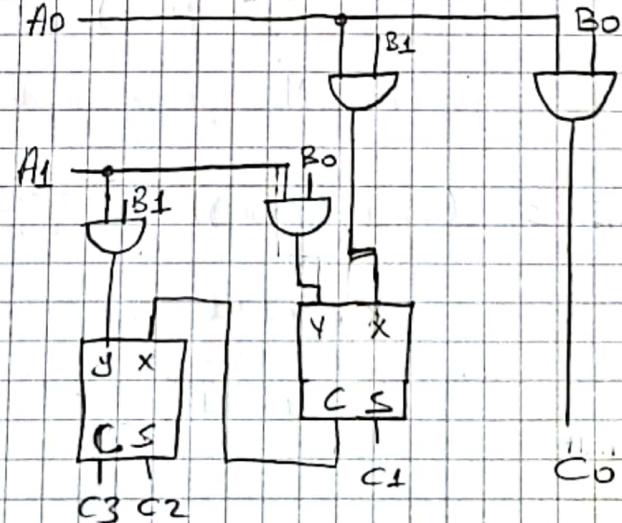


* $C_I + A_0 + B_0$ yap C_1 ve S_0 'ı bul.

* $C_I + B_1 + A_1$ yap C_2 ve S_1 'ı bul

Multiplication (Çarpma)

$$\begin{array}{r}
 & B_1 & B_0 \\
 & \times & \\
 A_1 & & A_0 \\
 \hline
 A_1.B_1 & A_1.B_0 \\
 \hline
 + & C_3 & C_2 & C_1 & C_0
 \end{array}$$



Subtraction (Çıkarma)

Tümleyen Yöntemi:

$(r-1)^n$ e'ye Tümleyen:

$r \rightarrow$ basamak tabanı

$n \rightarrow$ basamak sayısı

$N \rightarrow$ n basamaklı r tabanında bir sayı ise

$$(r-1)^n$$
 e'ye Tümleyeni = $\boxed{r^n - N - 1}$

$$M$$
 ondalık (virgül) sayı ise = $\boxed{r^n - r^m - N}$

* 10'lu tabanda 9'a, 2'lik tabanda 1'e şere tümleme yapılır.

Örneğin:

* $(52520)_{10}$ olsun bunun 9'a tümleyeni = $\begin{array}{r} 99999 \\ - 52520 \\ \hline 47479 \end{array}$
veya formüle uygulsat $(10^5 - 52520 - 1) \rightarrow \boxed{147479}$

* $(101100)_2$ olsun. Bunun 1'e tümleyeni = $\begin{array}{r} 11111 \\ - 101100 \\ \hline 010011 \end{array}$
veya formüle uygulsat $(2^6 - N - 1) \rightarrow$

! 1'e tümleme yaparken N sayısının içindeki 0'lar + 1'ler 0 olur.

* $(0,3267)_{10}$ olsun. (ondalık (virgül)) $\rightarrow \begin{array}{r} 0,9999 \\ - 0,3267 \\ \hline 0,6732 \end{array}$ veya formüle $(10^0 - 10^{-4} - N)$

* $(0,0110)_2$ olsun. $\rightarrow \begin{array}{r} 0,1111 \\ - 0,0110 \\ \hline 0,1001 \end{array}$ veya formüle $(2^0 - 2^{-4} - N)$

$(r-1)$ 'e Tümlenen ile Çıkarma:

$(M-N)$ 'yi bulmak için:

1-) N sayısının $(r-1)$ 'e göre tümlenenini al.

2-) Sonucu M sayısı ile topla.

3-) Eğer elde (taşıma) varsa toplamın sonucuna eklenir. (Sonuç pozitif)

4-) Eğer elde (taşıma) yoksa bulunan sayısının $(r-1)$ 'e göre tümleneni alınır ve başına " $-$ " konur.

Örneğin:

$$\begin{array}{r} N \\ \overline{(72532)}_{10} \end{array}$$

$\underline{(03250)}_{10}$ işlemi sorulsun.

1. adım N in 9'a göre tümleneni =

$$\begin{array}{r} 99999 \\ - 03250 \\ \hline 96749 \end{array}$$

2. adım: 96749

$$\begin{array}{r} + 72532 \\ \hline \text{Elde } \boxed{169281} \end{array}$$

3. adım: $\begin{array}{r} 1 \\ \boxed{69282} \end{array}$ Sonuç

$$\begin{array}{r} M \\ \overline{(03250)}_{10} \\ \overline{- (72532)}_{10} \end{array}$$

} işlemi sorulsun.

1. adım N 'nin 9'a göre tümleneni $\rightarrow \boxed{27467}$

2. adım 27467

$$\begin{array}{r} + 03250 \\ \hline \text{Elde } \boxed{30717} \end{array}$$

3. adım: Sonucun 9'a göre tümleneni = -69282

Basına eksi troy

Örnek 1: $(\begin{array}{cccc} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{array})$ $\xrightarrow{\substack{M \\ N}}$ işlemi sorulsun.

1. adım N'in 1'e gide tamlayıcı.(kolay yöntem \Rightarrow O'ları 1, 11eri 0)
011 1011

2-2dim Nintcmayen'i (lie pse) ile M'itopla.

$$\begin{array}{r}
 & 1010100 \\
 & 0111011 \\
 + & \boxed{1}0001111 \\
 \hline
 & 1 \\
 & + \\
 \hline
 \boxed{0010000} & \rightarrow \text{Sonuç}
 \end{array}$$

3-2dim

Örnek 5: $(1000100)_2$ $\overset{M}{\overbrace{}}$ $\overset{N}{\overbrace{(1010100)_2}}$ } işlemi sorulsun.

1. adım N'ın 1'e göre tamleyeni = 0101011

$$\begin{array}{r} \text{2-adim} & 1000 \ 100 \rightarrow M \\ & + 0101 \ 011 \hline & 1101111 \rightarrow M' \text{in 110 goes to m'leyen}. \\ \text{elde} & \leftarrow \\ \text{gör} & \end{array}$$

4.-adım Sonuçun timleyeninin ac (*I'ye göre*, *basisindan*-*"kay*).

1-00 10000

(r'ye t̄mleyen)

$(r - N) \rightarrow N$ 'in r'ye t̄mleyeninin formülü -

Örneğin: $(52520)_{10}$.

$$10^4$$
'a t̄mleyeni. \Rightarrow
$$\begin{array}{r} 100000 \\ - 52520 \\ \hline 47480 \end{array}$$

Örneğin: $(101100)_2 \xrightarrow{N}$

2'ye t̄mleyenini bulmak için 0'ları 1, 1'leri 0 yap ve alttan sayıya + ekle.

$$(101100)_2 \rightarrow 010011$$

$$\begin{array}{r} + 1 \\ \hline 010100 \end{array}$$

$\rightarrow N$ 'in 2'ye göre t̄mleyeni

(Two's complement)

r'ye t̄mleyen yöntemi ile Çıkarma:

$(M - N)$ işlemini yapmak için:

1-) N 'in r'ye göre t̄mleyenini al.

2-) Bu t̄mleyeni M ile topla.

3-) Eğer elde ($t̄zma$) varsa \Rightarrow Eldeyi at. (Sonuç pozitiftir).

4-) Eğer elde ($t̄zma$) yoksa \Rightarrow Sonucun r'ye göre t̄mleyenini al basına "-" boy.

Örneğin $72532 - 53250$ } islemi sonulsun.

$$\begin{array}{r} 100000 \\ - 03250 \\ \hline 96750 \end{array}$$

$$\begin{array}{r} 72532 \\ + 53250 \\ \hline \text{elde} \leftarrow 1169282 \end{array}$$

, 3-) Elde atılır.

$$\boxed{69282}$$

Örneğin $\begin{array}{r} 03250 \\ - 72532 \\ \hline \end{array}$ ^M _N } işlemi sorulsun.

1-) N'in 10'a göre tamleyeni

$$\begin{array}{r} 100000 \\ - 72532 \\ \hline 27468 \end{array}$$

2-) Mile topla. $\begin{array}{r} 03250 \\ + 27468 \\ \hline \boxed{30718} \end{array}$ ^{oldé yok.}

4-) 30718'in 10'a göre tamleyeni

$$\begin{array}{r} 100000 \\ - 30718 \\ \hline \boxed{-69282} \end{array}$$

^{eksi}

Örneğin: $\begin{array}{r} 1000100 \\ - 1010100 \\ \hline \end{array}$ ^M _N } işlemi sorulsun.

$$\begin{array}{r} 0101011 \\ + 1 \\ \hline 0101100 \end{array}$$

$$\begin{array}{r} 1000100 \\ + 110000 \\ \hline \boxed{1110000} \end{array}$$

^{oldé yok.}

$$\begin{array}{r} 0001111 \\ + 1 \\ \hline \boxed{-0010000} \end{array}$$

^{eksi boy.}

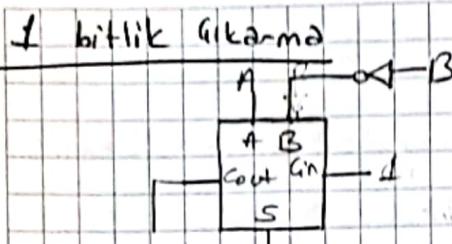
Örneğin: $\begin{array}{r} 1010100 \\ - 1000100 \\ \hline \end{array}$ ^M _N } işlemi sorulsun.

$$\begin{array}{r} 0111011 \\ + 1 \\ \hline 0111100 \end{array}$$

$$\begin{array}{r} 1010100 \\ + 0111100 \\ \hline \boxed{110010000} \end{array}$$

^{oldé}

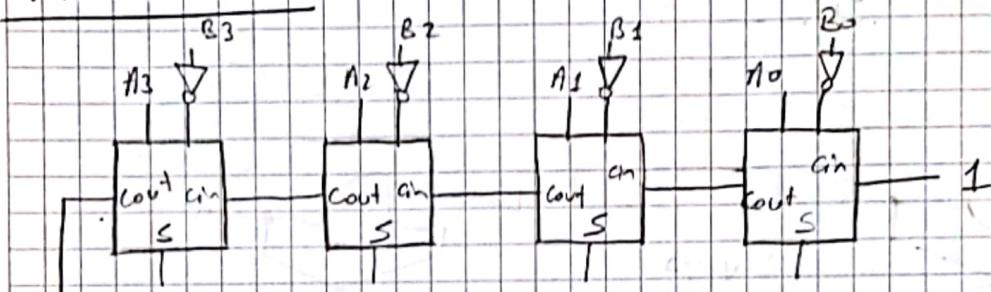
$$\begin{array}{r} 0010000 \\ \boxed{0010000} \end{array}$$



$$A + B' + 1$$

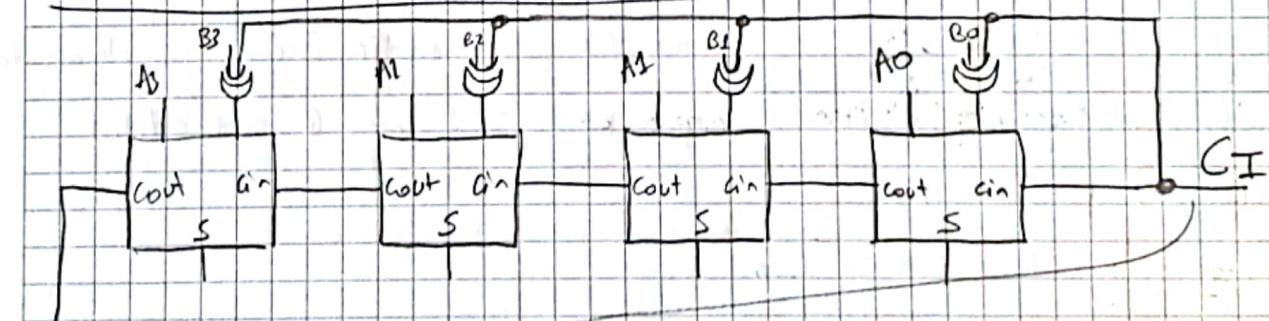
2'ye
tümleyen elmetek bu

4 bitlik Çıkarma



! Bu devre A, B'den büyükse (sonuç pozitifse) doğru çalışır.

4 bitlik Toplama - Çıkarma devresi



C_0 $X \quad Y$ $X \oplus Y$ $\textcircled{*} \quad V$ CI'den 1 verirseki kapılar not

0 0 0 kapısı jaibi davranışır ve çıkarma derresi olur.

0 1 1 (A, B'den büyükse çıkarmayı dizeğin yapar.)

1 0 1 $\textcircled{*}$ CI'den 0 verilirse kapılar B neye o

1 1 0 olur. ve toplama derresi olur.

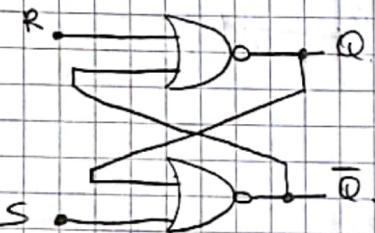
! Kısaca bu devre toplama ve çıkarma yapabir. (CI durumuna göre).

LATCHES (Mandallar)

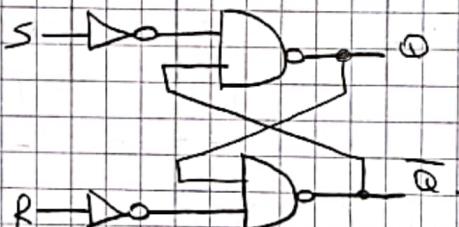
Bir bitlik bilgiyi tutan ve tamleyeni ile beraber bilginin kendisini çıkış olarak veren çift çıkışlı elemanlardır.

Cıktı sadece mevcut giriş seviyelerine değil, bir önceki giriş ve çıkış seviyelerine de bağlıdır.

S-R Latch



veya.



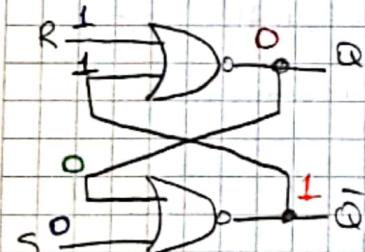
(Bu daha çok kullanılıyor)

! S-R Latch'de aynı anda en fazla bir girişin 1 olmalıdır.

$R \Rightarrow$ Reset (0) , $S \Rightarrow$ Set (1) demektir. Bunu yaparken de \bar{Q} 'ı tam tersine eşitler. (örnegin $R=0$, $S=1$ ise Q set edilip 1 olur \bar{Q} ise 0 olur)

Örnegin : $R=1$, $S=0$ olsun.

Adım 1 : $R=1$, $S=0$ yapılır.



Adım 2: Herhangi bir girişin 1 olan NOR kapısının çıkışını "0" yapacağından $Q=0$ yapıyoruz.

Adım 3: Geri besleme ile Q çıkışı S ile NOR'a sokulur.

Adım 4: 0 ile 0 NOR'a girerse "1" çıkar.

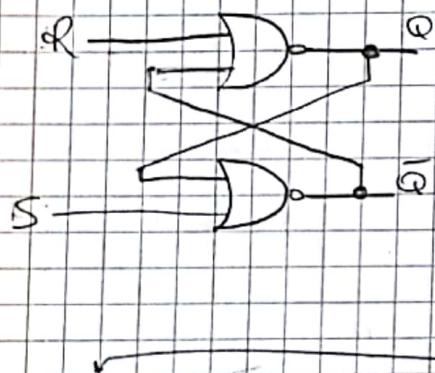
Adım 5: \bar{Q} geri besleme ile R'ın olduğunu göstermektedir.

Böylece Reset(R) işlemi yapılmış olur.

Neyse formül olarak:

$$Q_{\text{next}} = (R + \bar{Q}_{\text{current}})'$$

$$\bar{Q}_{\text{next}} = (S + Q_{\text{current}})' \quad \text{yazılabilir.}$$



S	R	Q
0	0	No change (Mevcut durum devam eder.)
0	1	Reset (0)
1	0	Set (1)
1	1	Yasaklı durum

! NOR kapisi ile yapılan devrelerde $S=1$, $R=1$ ise Q ve \bar{Q} aynı anda "0" olur. (Kendisi ve tersi aynı olamaz)

NAND kapisi ile yapılan devrelerde ise $S=1$, $R=1$ yapılıncaya Q ve \bar{Q} aynı anda "1" oluyor.

* No change demek mevcut durumun korunmaya devam etmesidir.

Örneğin: Q yu set edip 1 yaptıktan sonra ($S=1$, $R=0$ durumu) bit. $S=0$, $R=0$ yaparsak Q hala 1 olmaya devam eder.

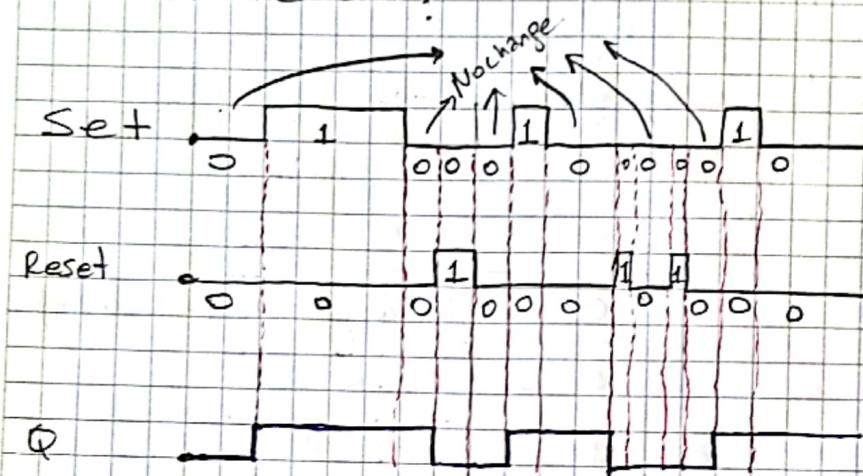
Soru:

*) Burada sadece Q ya ve $S-R$ 'nın durumuna bakarak Q_{next} i yaz. Sonra Q'_{next} i yaz.

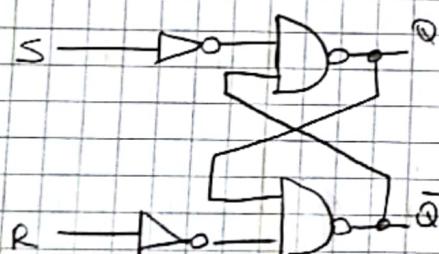
Current Q Q'		S R	Next Q Q'	
0	1	0	0	1
1	0	0	1	0
0	1	0	0	1
1	0	0	0	1
0	1	1	1	0
1	0	1	1	0

| $S=1$, $R=1$ durumu ele alınmadı |

Soru: S-R latchesinin S ve R anahtarları grafikteki gibi seçildiğinde Q çıkışları ne olur?

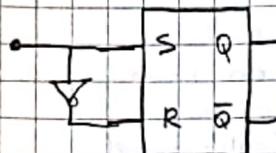


! NAND ile yapılan S-R latch'de;

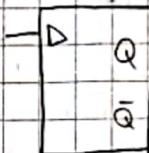


S	R	Q
1	1	No change
1	0	Reset (0)
0	1	Set (1)
0	0	Kullanılmaz

D Latch



S ucuna gelen kaynagini
degili alinarak R
ucuna baglanir.



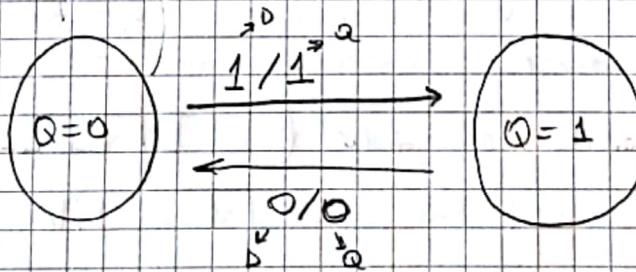
$\rightarrow D \rightarrow 0 \Rightarrow \text{Reset}$

$D \rightarrow 1 \Rightarrow \text{Set}$ durumudur

D	Q	\bar{Q}
0	0	1
1	1	0

1. S-R 'deki yasak durum ($S=1, R=1$) piderilmistir.

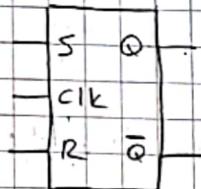
D Latch'inin durum diaigrami euk Sekildedir:



FLIP - FLOPS

Latchlerde saat sinyali yoktur. (Asenkronurlar)

Girişleri, bir başka yetki girişiyle kontrol edilen rastgele flip-flop denir. Bu yetki girişi (clock, Enable) fonksiyonel girişlerin senkronize olarak çıkış etkilemesini (flip-flopu tetiklemesini) sağlar.



Artık clock girişimiz var.

Tetikleme Aşıtları :

Flip-flopların zamanlama sinyalinin seviyesine bağlı olarak durum değiştirmesine tetikleme denir.

Tetikleme sinyali periyodik kare dalgası ($\square \square \square \square$) veya kare dalgası formunda pulsler ($\square \square$) ile yapılır.

Tetikleyici sinyal olan kare dalganın 4 farklı evresi vardır:

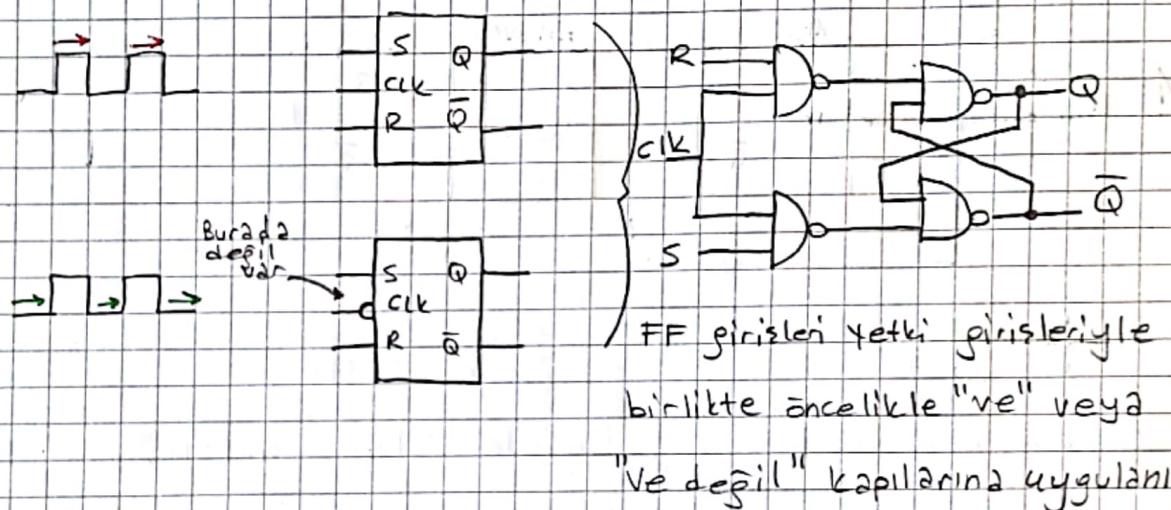
- Pozitif Seviye.
- Negatif Seviye
- Pozitif Kenar
- Negatif Kenar



→ Özellikle pozitif kenar ve negatif kenar tetikleme kullanılır.

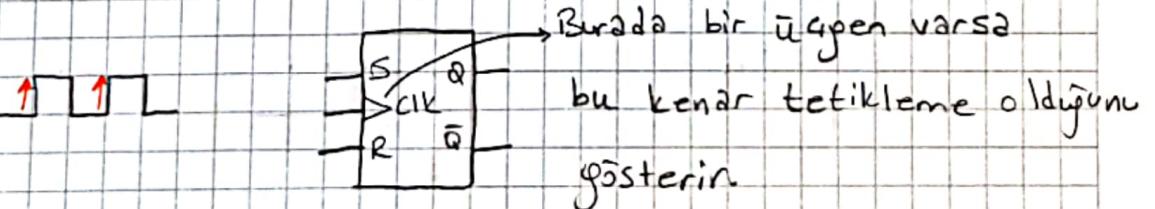
1-) Seviye Tetikleme:

Bu tip tetiklemede CLK girişine uygulanan kare dalganının 1 veya 0 seviyesi süresi boyunca FF girişlerindeki dep̄isimler çıkışları etkiler. Aslında bu bir yetkilendirme işlemidir.



2-) Kenar Tetikleme:

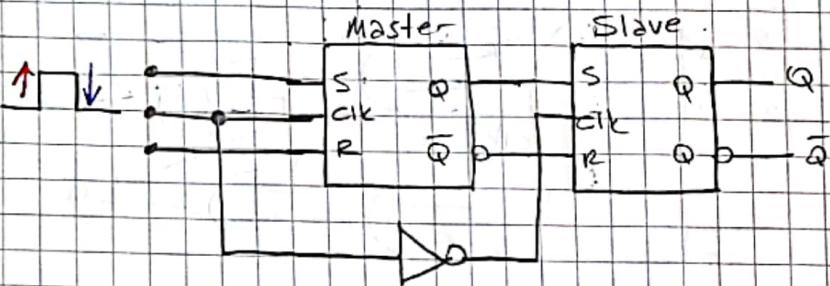
Genelde FF girişlerinin, puls penisiği sürecince çıkış etkilemesi dēgil çok kısa bir anda tetiklemenin yapılp FF'un bir sonraki tetikleme pulsuna kadar kilitli kalması istenir. Çok kısa olan seviyeler arası penisi süresi tetikleme için idealdir.



Master-Slave

FF'ların master-Slave bağlanması kenar tetiklemeyi sağlayan bir tasarım şeklidir.

Negatif Kenar tetikleme

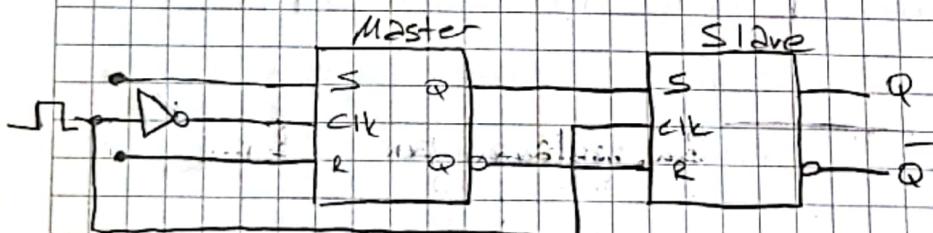


Clokun pozitif kenarı geldiğinde Master,

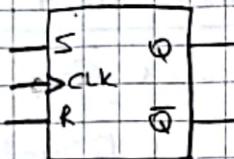
Clokun negatif kenarı geldiğinde Slave çalışır.

Negatif kenar tetikleme denmesinin nedeni çıkış yapılan (Slave) FF'nin Ck'nın negatif kenarına bağlı olması.

Pozitif Kenar tetikleme

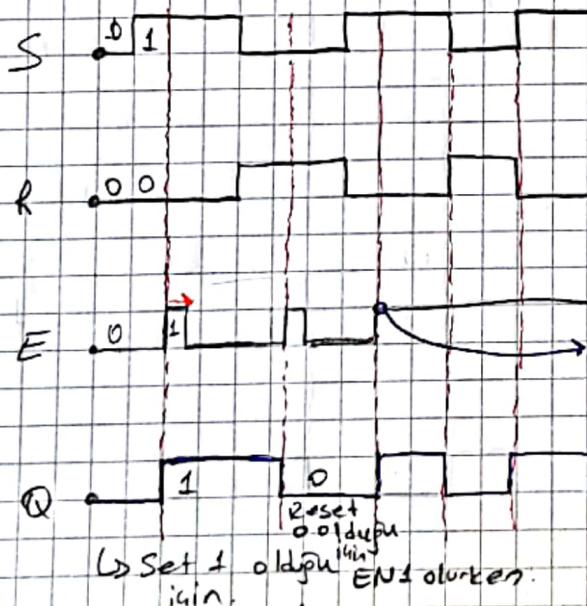


S-R Flip-Flop



S	R	Clk	Q_{n+1}	\bar{Q}_{n+1}
X	X	0	Q_n	\bar{Q}_n (No change)
0	0	1	Q_n	\bar{Q}_n (No change)
0	1	1	0	1 (R set)
1	0	1	1	0 (Set)
1	1	1	1	1 (Yasak durum)

SORU: Enable (yetkilendirilmiş) pırcısı bulunan S-R flip-flop'un zamanla bağlı Q çıkış profiliini çiziniz. (Pozitif seviye)



Bu noktadan sonra Enable (Clk) hep 1 olduğundan yetki S ve R ye geçti, yani S ve R'nin anlık değerini tıkla ettilerdi
(clock durbesi olmadan)

↳ Set 1 olduğunda EN1 olurken.
isin.

Öncelikle Enable (yetki) pırcısının aktive olduğu yerlere bilmeliyiz.
Sonra, yetki pırcısının aktif olduğu sure içinde S ve/veya R'de değişiklik olması mu diye bak. (Turuncu yer)

J-K Flip-Flop



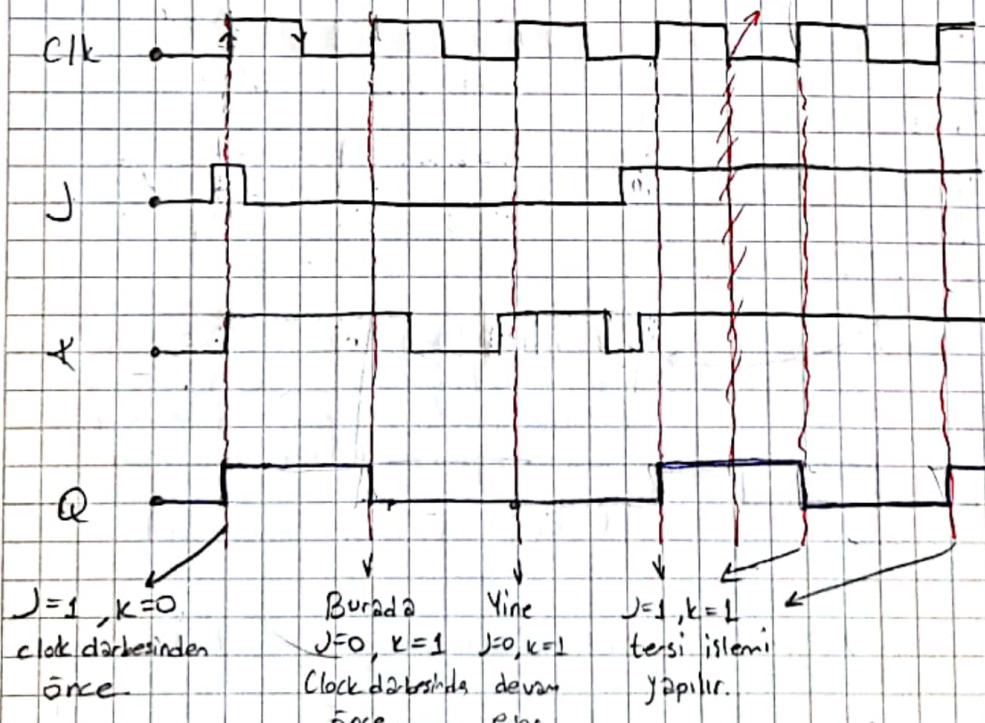
$J \rightarrow \text{Set}$

$K \rightarrow \text{Reset}$.

E	R	Clk	J	K	Q_{next}	\bar{Q}_{next}
		0	X	X	\bar{Q}_{current}	Q_{current} (No change)
		1	0	0	\bar{Q}_{current}	\bar{Q}_{current} (No change)
		1	0	1	0	1 (Reset)
		1	1	0	1	0 (Set)
		1	1	1	\bar{Q}_{current}	Q_{current} (Complement)

Soru: Aşağıda pozitif kenar tetiklemeli J-K flip flop
giriş sinyalleri verilmiştir. (Q çıkış sinyalinin çiziniz.)

Burası olmazacak.

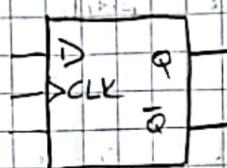


$J=1, K=0$
clock dakisinden önce.

Burada
Yine
 $J=0, K=1$
 $J=0, K=1$
Clock dakisinda devam
önce.
edir.

$J=1, K=1$
tersi işlemi
yapılır.

D- Flip-Flop :



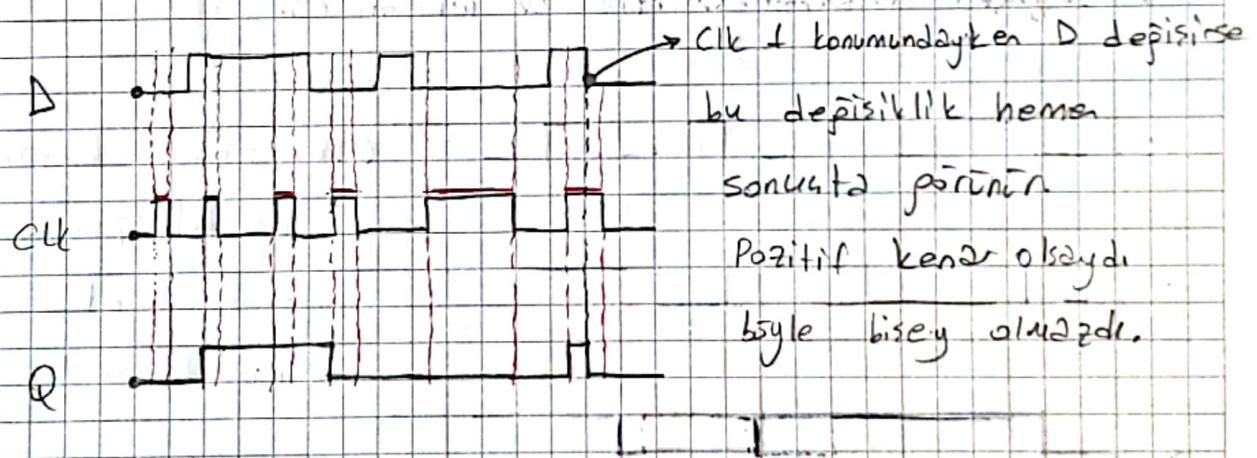
CLK	D	Q _{current}	Q _{next}
0	X	Q _{current}	Q _{current} (No change)
1	0	0	1 (Reset)
1	1	1	0 (Set)

* D_{giris} seviyesi Q_{çıkış} seviyesi

ile aynıdır.

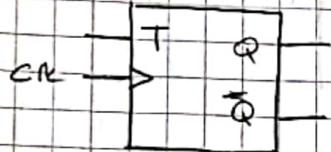
kenar deseydi ($\uparrow \uparrow \downarrow \downarrow$)

Soru: Pozitif seviye ($\uparrow \uparrow$) tetiklemeli D flip-flop için aşağıdaki CLK ve D verilmiştir. Bu na şe : Q'yu çiz.



T Flip-Flop

J-k flip flop'unun girişleri birleştirilek elde edilir.

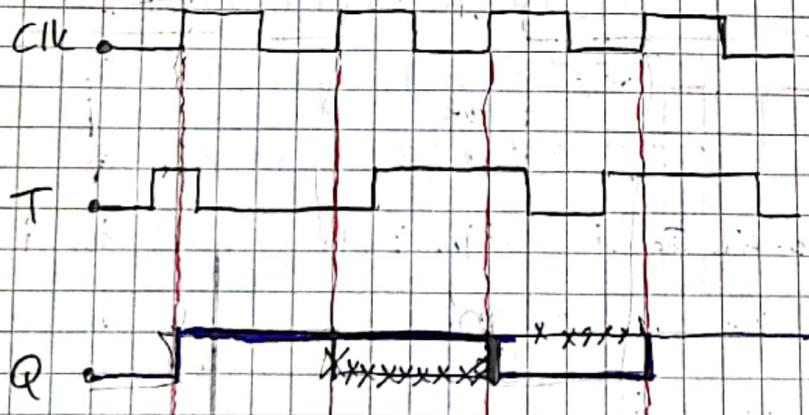


Clk · T	Q next
0 · X	Qcurrent (No change)
1 · 0	Qcurrent (No change)
1 · 1	Qcurrent (Complement.)

1-bitlik sayıcı olabilir (0'dan 1'e)

Soru: Pozitif kenar tetiklemeli T flip-flop'un

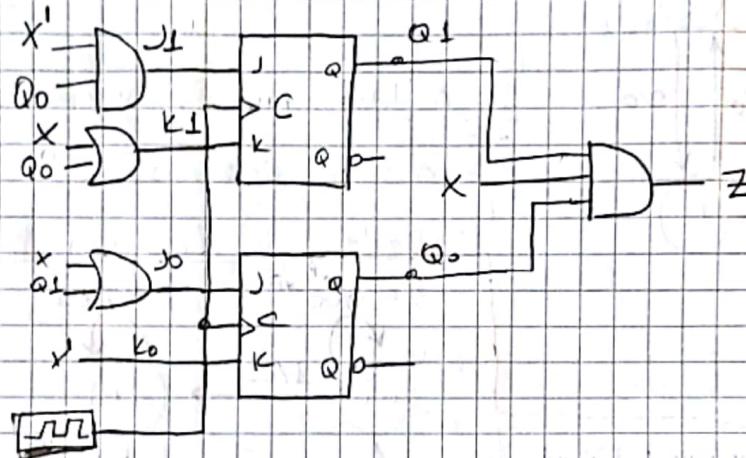
Clk'sı ve T'si verilmiştir. Q çıkışını çiziniz.
(Q başlangıç 0 kabul et)



Clk yükselen kenara
geldiğinde $T=0$ ise
aynen devam et.
 $T=1$ ise durum depistir.

Sequential Circuit Analysis (Ardışık Devre Analizi)

Örnek:



1. Adım: Öncelikle J_1, K_1, J_0, K_0 , ve Z 'nin denklemelerini buluyoruz.

$$J_1 = X' \cdot Q_0, \quad K_1 = X + Q_0, \quad J_0 = X + Q_1, \quad K_0 = X', \quad Z = Q_1 \cdot Q_0 \cdot X$$

2. Adım Tablo oluştur.

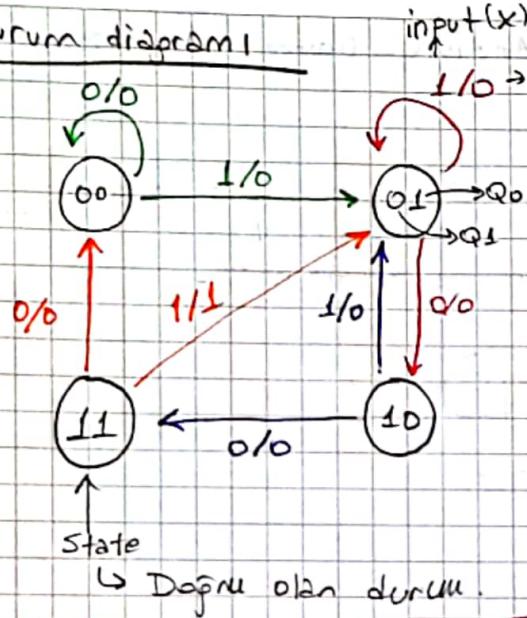
Burayı formüllerle
göre doldur.

Present State		Input	Flip-Flop Input				Next State		Output
Q_1	Q_0	X	J_1	K_1	J_0	K_0	Q_1	Q_0	Z
0	0	0	0	0	0	1	0	1	0
0	0	1	0	1	1	0	0	1	0
0	1	0	1	1	0	1	1	0	0
0	1	1	0	1	1	0	0	1	0
1	0	0	0	0	1	1	1	0	0
1	0	1	0	1	1	0	0	1	0
1	1	0	1	1	1	1	0	0	0
1	1	1	0	1	1	0	0	1	1

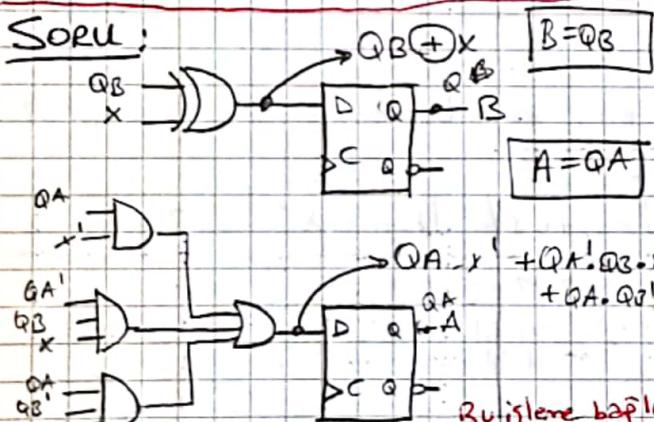
J	K	Q_{next}
0	0	$Q(t)$ No change
0	1	0 Reset
1	0	1 Set
1	1	$Q(t) + 1$ Complement

Nextleri (örneğin $Q_1 next$) bulmak için Q_1 present ve J, K ile birlikte $J-K$ 'nın karakteristik tablosundaki gibi doldurmaliyiz.

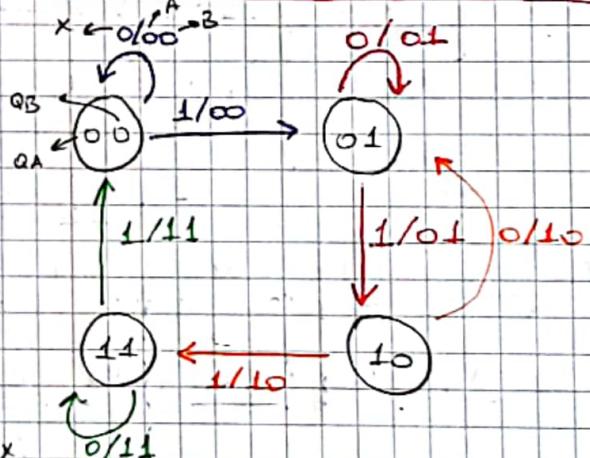
Durum diagramı



Soru:



n tane flip flop varsa 2^n durum
m tane girişi varsa 2^m ok (\uparrow) vardır
(Don't care durumları hariç)



Present QA	Present QB	input X	$Q_B + X$	$Q_A \cdot X + Q_A' \cdot Q_B \cdot X + Q_A \cdot Q_B' \cdot X$	FF Input DB	FF Input DA	Next State QA	Next State QB	Outputs A	Outputs B
0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	1	0	0	1	0	0
0	1	0	1	0	1	0	0	1	0	1
0	1	1	0	1	0	1	1	0	0	1
1	0	0	0	1	0	1	1	0	1	0
1	0	1	1	1	1	1	1	1	1	0
1	1	0	1	1	1	1	1	1	1	1
1	1	1	0	0	0	0	0	0	1	1

İşlemler DA ve DB ye bağlı olduğundan DA ve DB'yi bulurken kolaylık olsun diye tabloya yazdırık.

D karakteristik tablosu

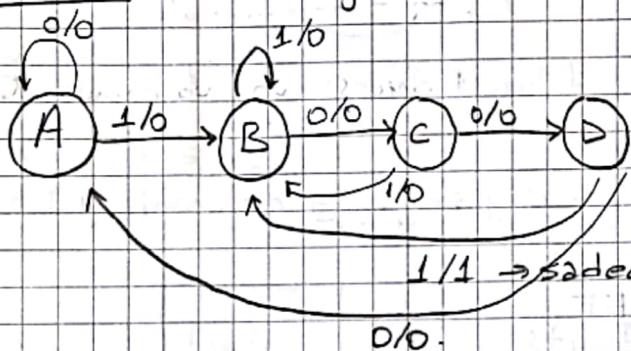
$D=0$ reset, $D=1$ set.

Sequential Circuit Design

(Ardısal Devre Tasarımı)

Soru: 1001 girişinde çıkışında 4 tane dijital durumda
oluşan devreyi J-K flip flop ile tasarlayın.

Adım 1: Durum diagramı tasarıla.



Adım 2: Bunları tabloya yerleştirir.

Bunu ek tar

Present State	Input	Next State	Output
Q_1	Q_0	Q_1	Q_0
A	0	A	0
A	1	B	0
B	0	C	0
B	1	B	0
C	0	D	0
C	1	B	0
D	0	A	0
D	1	B	1

Present State	Input	Next State	Output
Q_1	Q_0	X	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Adım 3: Present ve Next State'ler için J-K inputlarını bul.

J-K	Q(t+1)	Q(t+1)	Q(t+2)	J	K
0 0	Q(t) (No change)	0	0	0	X
0 1	0 (Reset)	0	1	1	X
1 0	1 (Set)	1	0	X	1
1 1	Q'(t) (Complement)	1	1	X	0

J 1'den 1 olursa ya No change 0-0

yada Set 1-0
bu yüzden J'nin önemi yok (X)

Present State $Q_1\ Q_0$	Input X	Next State $Q_1'\ Q_0'$	FF Inputs J ₁ K ₁ J ₀ K ₀	Output Z
0 0	0	0 0	0 X 0 X	0
0 0	1	0 1	0 X 1 X	0
0 1	0	1 0	1 X X 1	0
0 1	1	0 1	0 X X 0	0
1 0	0	1 1	X 0 1 X	0
1 0	1	0 1	X 1 1 X	0
1 1	0	0 0	X 1 X 1	0
1 1	1	0 1	X 1 X 0	1

4. Adım Karnaugh map yardımı ile;

J_1, K_1, J_0, K_0 ve Z yi bul.

Örneğin

$$J_1 = Q_0 \cdot X', \quad K_1 = X + Q_0, \quad J_0 = X + Q_1, \quad K_0 = X', \quad Z = Q_1 \cdot Q_0 \cdot X$$

		00	01	11	10
		0	1	X	X
0	0	0	(1)	X	X
1	0	0	0	X	X

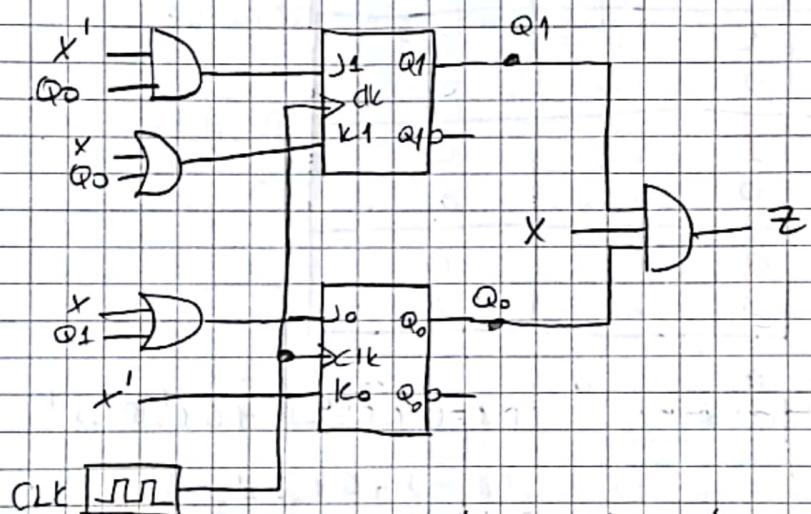
Bu x'li 1'li

dipolelerin
0'lu
ederek

Adım 5 : Devreyi çiz (Bulandırıcılarla devre)

$$\begin{aligned} J_1 &= X^1 \cdot Q_0 & K_1 &= X + Q_0 \\ J_0 &= X + Q_1 & K_0 &= X^1 \end{aligned}$$

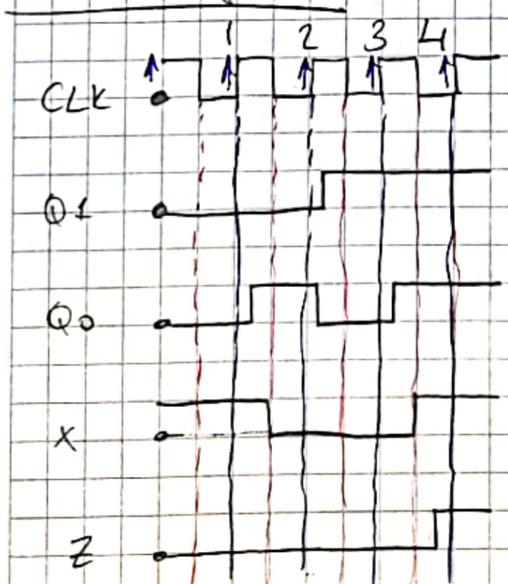
$$Z = \bar{Q}_1 \cdot Q_0 \cdot X$$



1001 girilince (sınasiyla) çıkışında (Z) 1 çıktı.

Devre

Zaman diagramı



Mavili dikkat ol.

Aynı derreyi D flip flop ile tasarlayalım.

Next state'nin ayınlaması.

Present state	Input	Next state	IFF Inputs	Output	D ₁ -D ₀ 'ı bul
Q ₁ Q ₀	X	Q ₁ Q ₀	D ₁ D ₀	Z	
0 0	0	0 0	0 0	0	
0 0	1	0 1	0 1	0	
0 1	0	1 0	1 0	0	
0 1	1	0 1	0 1	0	
1 0	0	1 1	1 1	0	
1 0	1	0 1	0 1	0	
1 1	0	0 0	0 0	0	
1 1	1	0 1	0 1	1	

D₁, D₀, Z yi formüle dek. $D_1 = Q_1 \cdot Q_0' \cdot x' + Q_1' \cdot Q_0 \cdot x'$

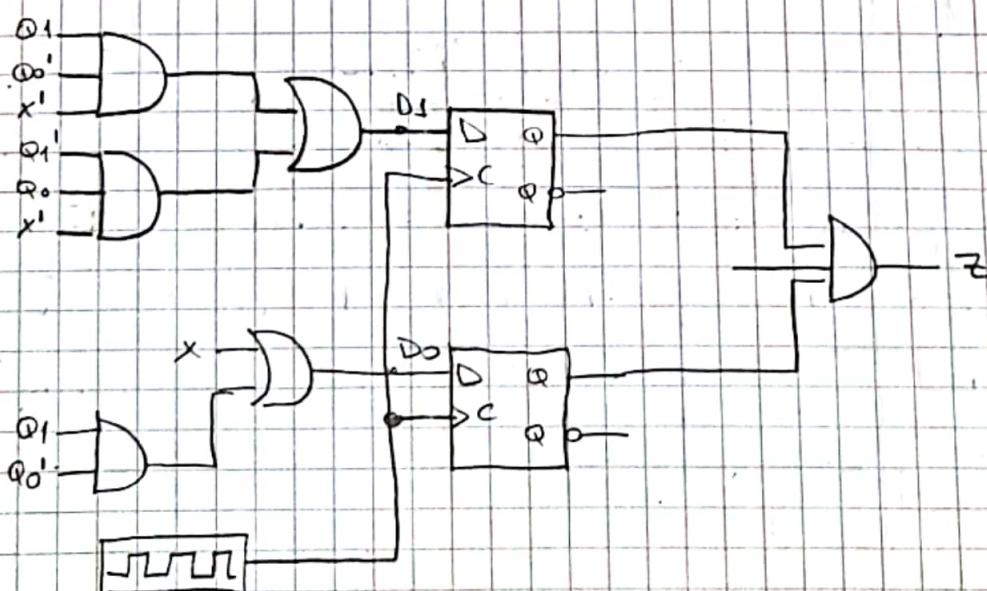
D₁ için

x	00	01	11	10
0	0	1		1
1	1			

$$D_0 = x + Q_1 \cdot Q_0'$$

$$Z = Q_1 \cdot Q_0 \cdot x$$

Devreyi çiz.

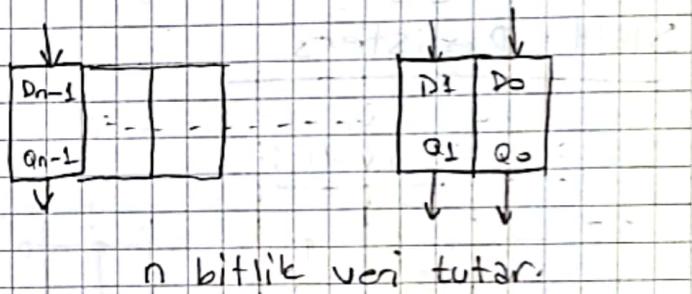
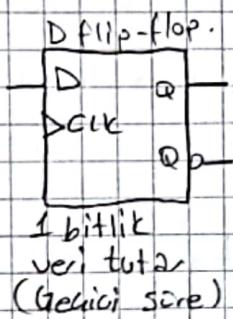


REGISTERS

İkili tabanda kodlanmış verilerin geçici süre ile tutulduğu kayıt ortamlarıdır.

Flip-floplar 1 bitlik hafıza hücresidir.

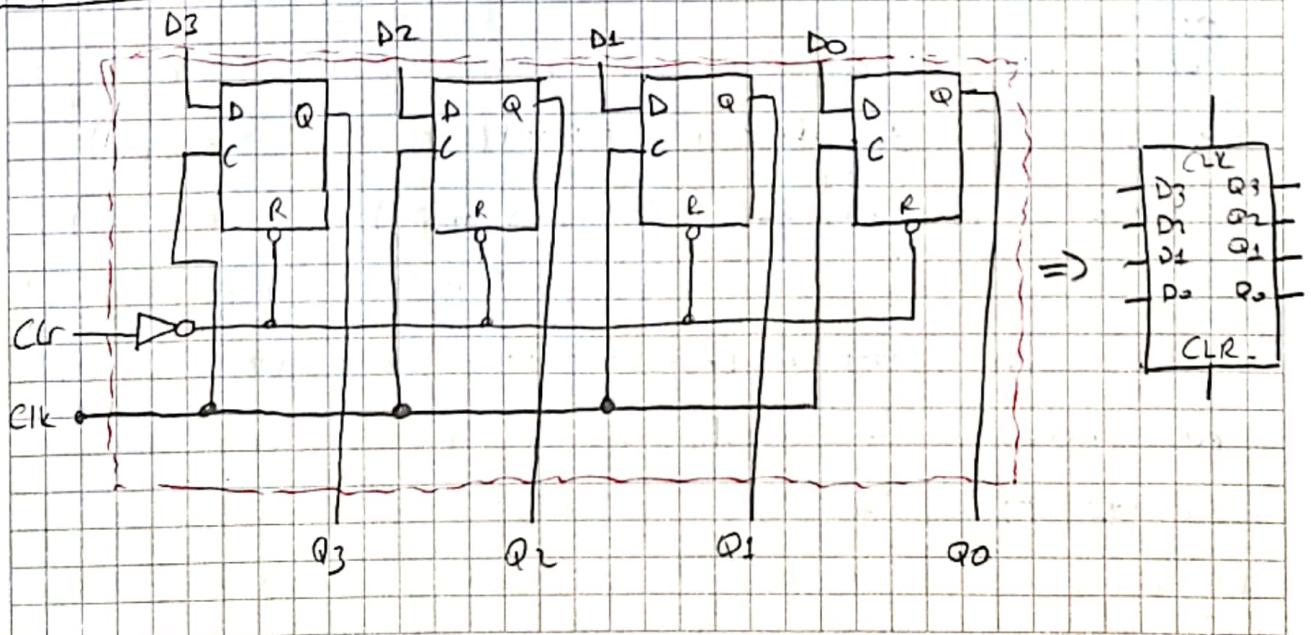
n bit uzunlukta veriyi depostmak için n adet flip flop gereklidir.



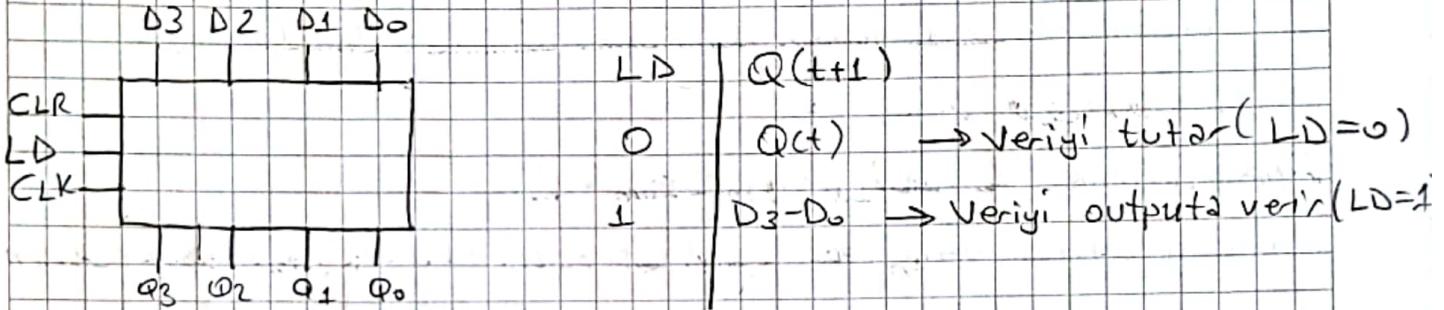
* Genelde D flip-flop kullanılır.

* Mikroişlemci ve mikrodenetleyicilerde, bilgisayar donanımlarında are bellek ve tampon bellek olarak kullanılır.

A Basic Register

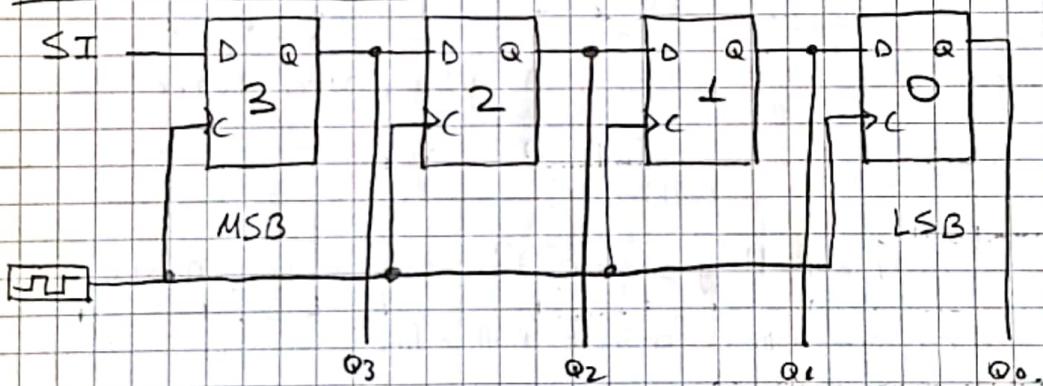


Adding a Parallel Load Operation



Shift Registers

Sağda kaydirmalı:



Örneğin: 1010 sayısı shift register'e kaydedilecek olsun.

İlk önce nereye shift ettiğine bakıyoruz. Bu devre sağda shift ediyor. Bu demektir ki LSB (Least Significant Bit) en düşük ağırlıklı basamak (1'ler basamağı) önce (yani 0) MSB (yani 1) (yükardaki örnek sayıya 1010'da göre) sonra girilecek "0" girilip ilk clock darbesi gerçekleştiğinde veri 3-flip flopa girer. Daha sonra "1" yapılıp bir clock darbesi daha olduğunda $0 \rightarrow 2$ ye, 1 ise $\rightarrow 3$ e girerek sağda doğru kaymaya olurlar. Diğer basamaklarda gırılerken kayıt tamamlanır. (Sola kaydirmalı olsaydı önce MSB on son LSB giriisi idi).

COUNTERS

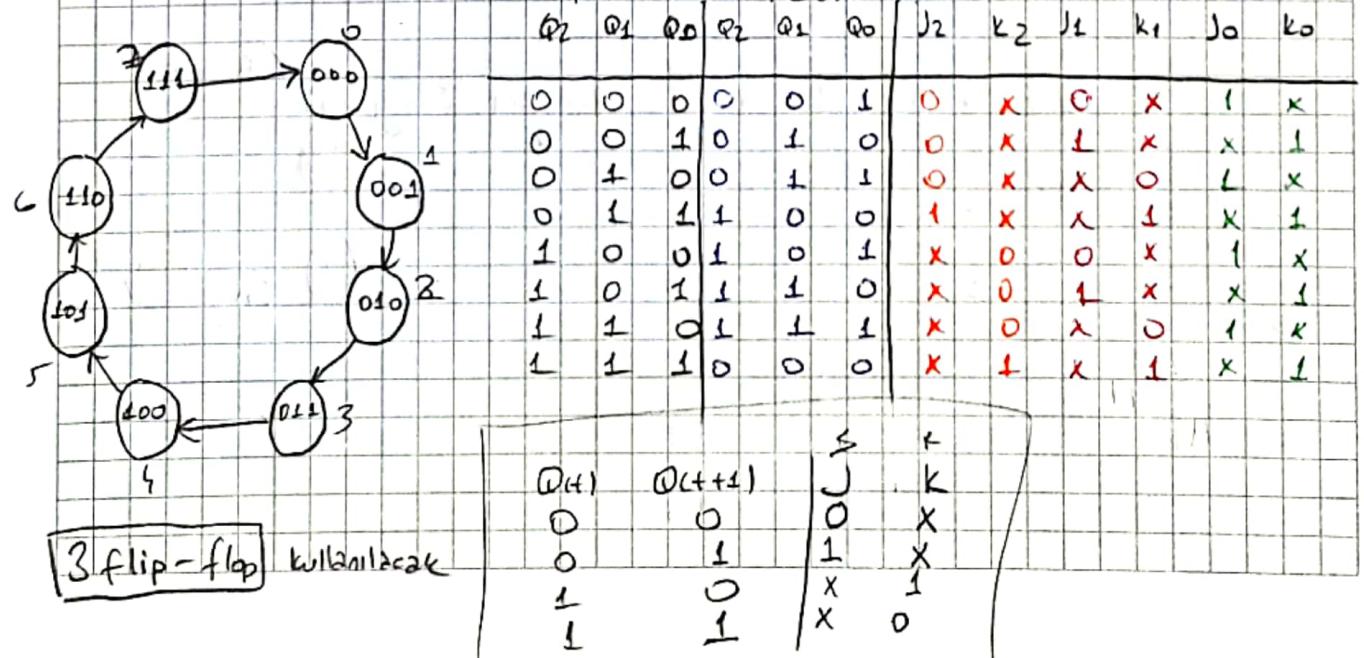
Sayı dizisi denilen önceden belirlenmiş durumları sırasıyla çıkış olarak işareten kaydedici devrelerdir.

- ① Sayıların temel elementi flip-floplardır.
- ② n adet flip-flop içeren bir sayıci en fazla $2^n - 1$ sayısına kadar sayabilir.

Senkron Sayıcılar (Tüm FF'liara aynı anda Clock sinyali piedir)

- 1-) Problem ortaya konur.
- 2-) Durumlar pörsel olarak (durum diagramı) ifade edilir.
- 3-) Giriş ve çıkış sayıları ile kullanılacak flip-flop sayısı ve tipi belirlenir.
- 4-) Flip-flop geçiş tabloları yardımıyla sistemin durum geçiş tablosu doldurulur.
- 5-) Her bir çıkış için, sadeleştirilmiş Boolean fonksiyon elde edilir (Karnaugh ile).
- 6-) Devresi çiz.

SORU: 0'dan 7'ye kadar ileri yönde sayan devreyi J-k ile tasvir et.



$J_2 = Q_0, Q_1$, $J_1 = Q_0$, $J_0 = 1 \rightarrow$ x11nin hepsini 1 setebiliriz.

$K_2 = Q_0, Q_1$, $K_1 = Q_0$, $K_0 = 1$

J_2 için

Q_0	Q_1	00	01	11	10
0		X	X		
1	(1)	X	X		

$$J_2 = Q_0, Q_1$$

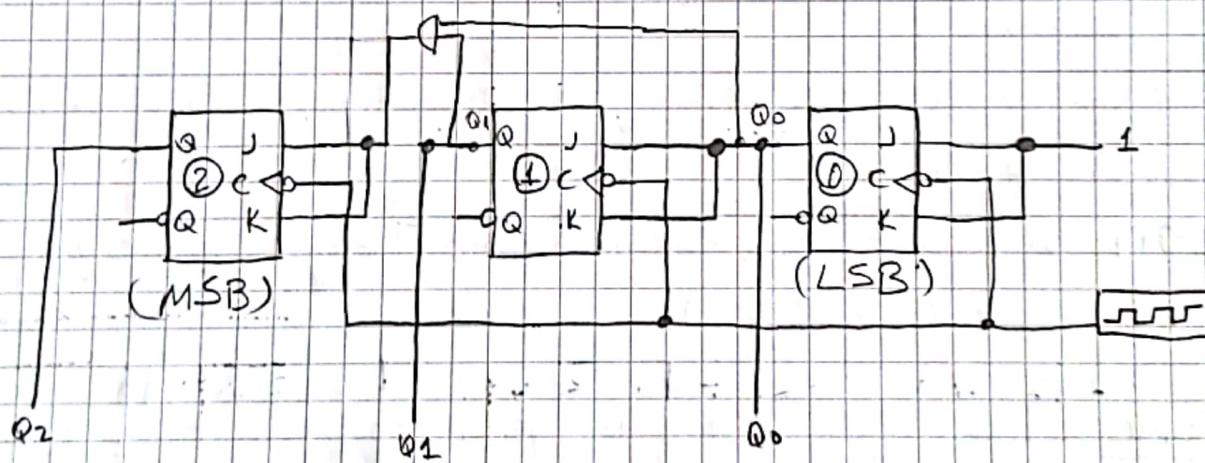
J_1 için

Q_0	Q_1	00	01	11	10
0		X	X		
1	(1)	X	X	1	

$$J_1 = Q_0$$

K_2 ve K_1

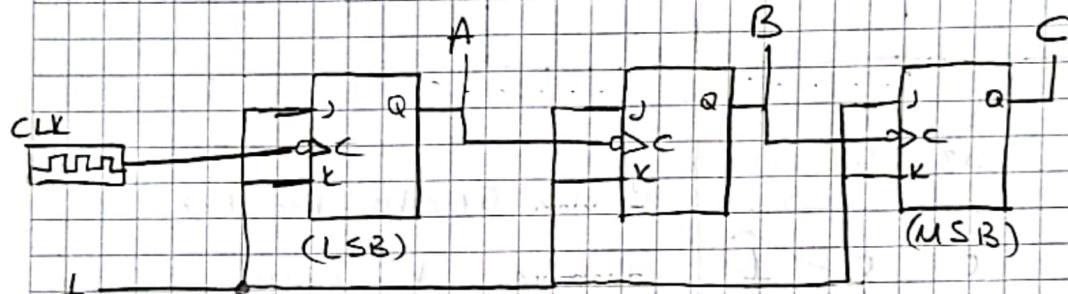
içinde
aynı işlemleri
uyguluyoruz.



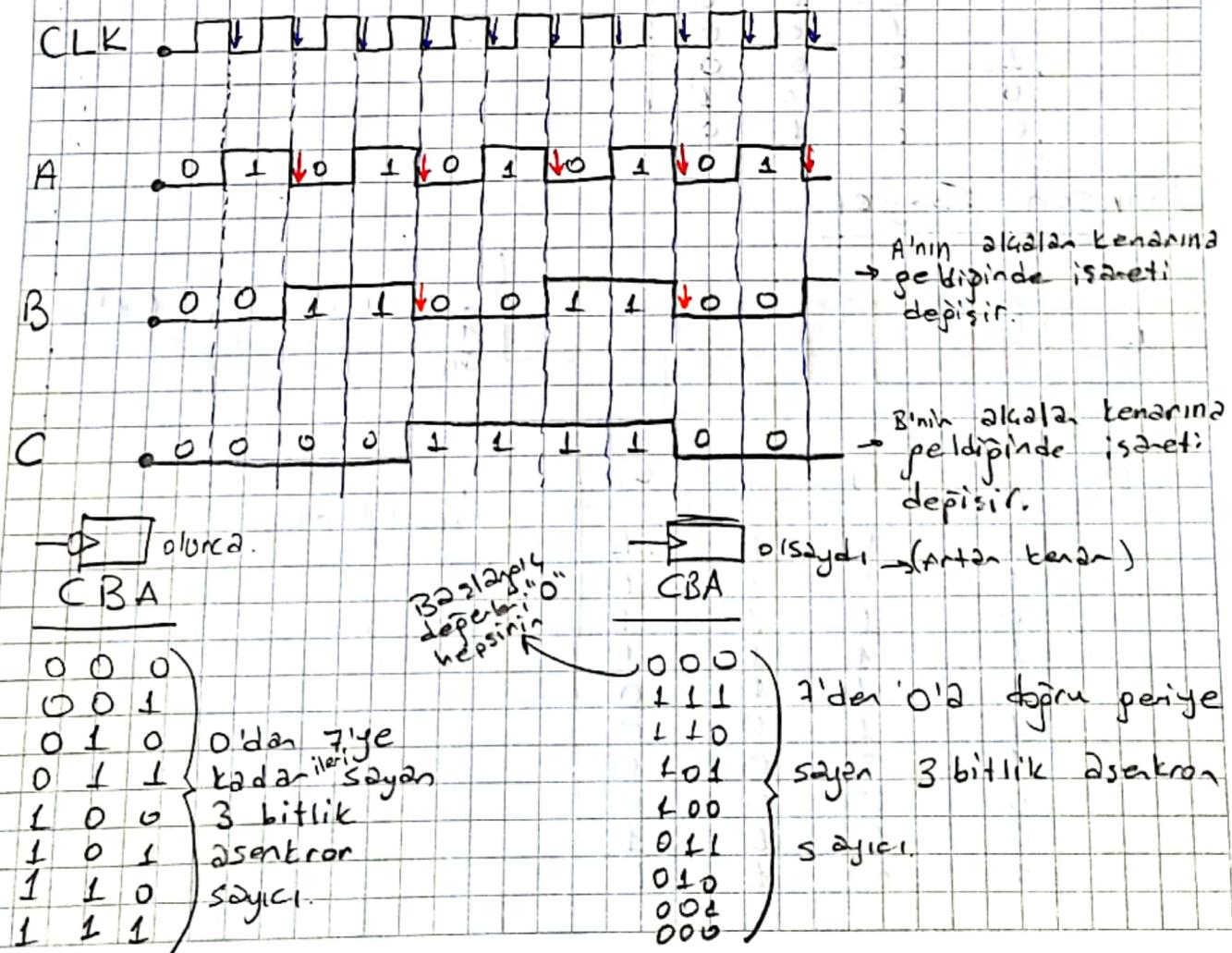
(3 bitlik Senkron Yukarı Sayıcı).

Aşenkron Sayıcılar

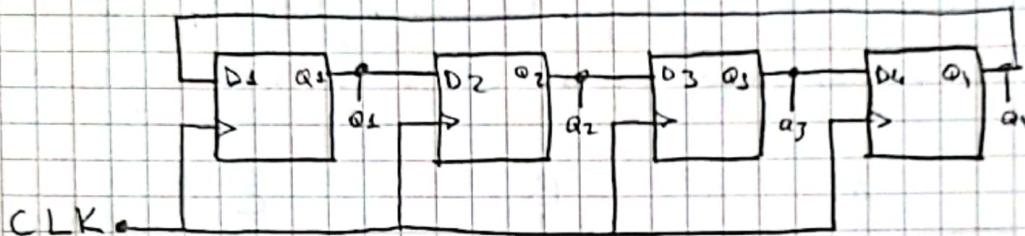
CLK en düşük ağırlıklı basamakta (LSB) başlıdır. Ve Q çıkışından alınan deşer diğer FF'un Clock girişine bağlanır.



(Sayı CBA'dır.) (CLK'inin başlı olduğu yer LSB)
~~→ C~~ olduğu için algalanken dikkate alınır.



Ring Sayıcı

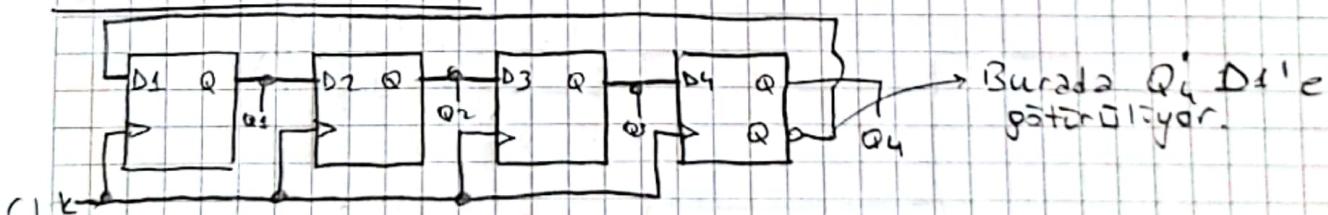


CLK	Q ₄	Q ₃	Q ₂	Q ₁ '
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	1	0	0	0
4	0	0	0	1
5	0	0	1	0
6	0	1	0	0
7	1	0	0	0
8	0	0	0	1
9	0	0	1	0

1 olan Q₁ çıkışının her clock

darbesinde kaydırılıyor.

Johnson Sayıcı



CLK	Q ₄	Q ₃	Q ₂	Q ₁
0	0	0	0	1
1	0	0	1	1
2	0	1	1	1
3	1	1	1	1
4	1	1	1	0
5	1	1	0	0
6	1	0	0	0
7	0	0	0	0
8	0	0	0	1
9	0	0	1	1

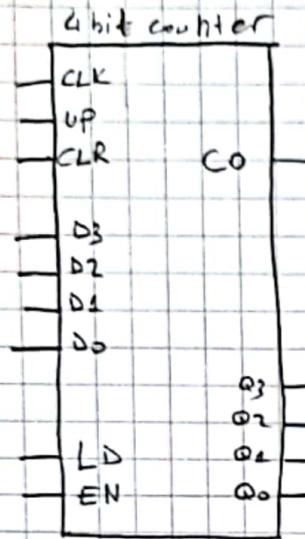
Burada Q₄'ı D₄'e
gönderiyor.

Q₄'ı oldupu kılın
buyle olun.

Q₁ → Q₂, Q₂ → Q₃, Q₃ → Q₄, Q₄ → Q₁

Fotokopide 8 bit counter Lâk

*



CLK → Clock sinyali bağlanır.

UP → 1 ise yukarı, 0 ise aşağı sayar.

CLR → Clear (Sıfırlar)

D₀-D₃ → Parallel veri yüklenmesi (data girisi)

LD → 1 olunca D₀-D₃'deki varileri yükler.

EN → Sayması için 1 olmalı.

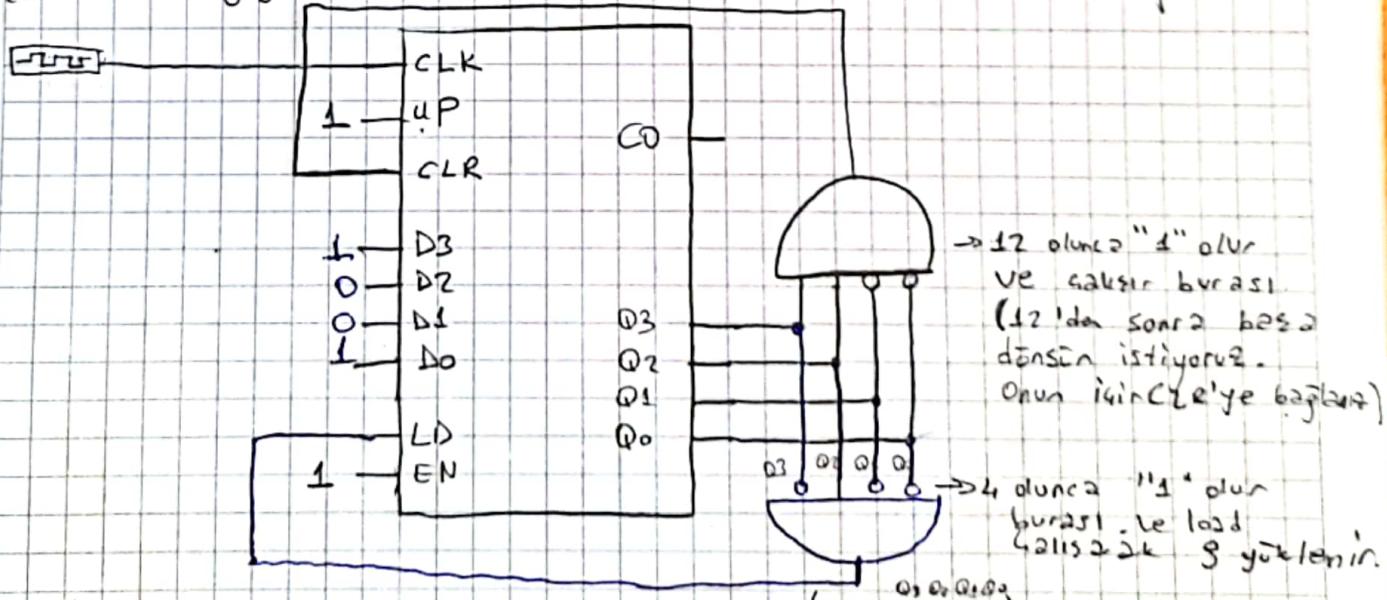
CO → Normalde 1'dir max değere gelince 0 olur.

* Bunların bazılarında testleme devreleri koyabilirler. dikkat et!

Q₀-Q₃ → Veri çıkışları.

SORU 0, 1, 2, 3, 4, 9, 10, 11, 12, 0, 1, 2, 4, 9, 10, 11, 12, 0... şeklinde sayıda bir sayıci yap.

* Yukarı sayıyor (UP=1) ve galisması (2'zim (EN=1) deşenleri verili)



→ 12 olunca "1" olur
ve bu sinyal burası
(12'den sonra başa
densin istiyoruz.
Onun için CO'ye bağlanır)

→ 4 olunca "1" olur
burası ve load
galis 2'lik 9 yüklenir.

1.adım: 12'den sonra sıfırlamasını ayarla. ($12 = 1100$)

2.adım: 4'den sonra ($4 = 0100$) , 9'ya geçmesini istiyoruz. Bunun
için Data girişlerine (D₀-D₃) 9 yazıyoruz ve sayıci 4 oldunda
Load işlemi yaparak 9'u (1001) yüklesin.

— DÖNEM SONU —