Sayısal Sistemler

- Güvenli
- Tekrarlanabilir
- Kalite değişmez, iyileştirilebilir
- Gürültü ve dış etkenlerden az etkilenir
- Ucuzdur.
- Kopyalamada bozulmaz. (iletim)

=) Taban

$(3,42)_n = 3.n^0 + 4.n^{-1} + 2n^{-2}$

$(0,bc)_n = (?)_m$

$0,bc \cdot m \quad a_1a_1a_2 \Rightarrow m_1 = a$

$0,a_1a_2 \cdot m = a_3,a_4a_5 \Rightarrow m_2 = a_3$

$\vdots$

$0,a_na_{n+1} \cdot m = a_{n+2},0$

Tümleme

=) Azaltılmış taban tümleme

$(r^n-1)-N$ — N: r tabanında n digitli bir sayı

=) Taban tümleme

$r^n - N = [(r^n-1)-N] + 1$

Azaltılmış T.T.

Tümleme ile Çıkarma

Taban: $M - N = M + (r^n - N) - r^n$

Taban-1: $M - N = M + [(r^n-1)-N] - r^n + 1$

İşaretli Sayılar

-> ①0..... 1 negatif, 0 pozitif

Örnek
- $1111 = -7$
- $0111 = 7$

* İlk terim negatiflik ifade eder, gerisi aynı hesaplanır.

Signed Magnitude

Taban Tümleme — Tümleyeni

- $1001 = -7$

$-1.8 + 2^0.1 = -7$

- 0111 = 7

-0.8 ...

Signed-2's Complements

Taban-1 Tümleme — Taban-1 Tümleyeni

- $1001 = -6$

$-1.(8-1)+1 = -6$

- $0110 = 6$

Signed-1's Complements

BCD Kod

* Her rakamı 4 bit ifade eder

-> $(21)_{10} = (0010\ 0001)_{BCD}$

Toplama

```
   1000   1000     88
     11   1001     39
+
   1100   0001    127
   0110   0110  ← önemli
+
 1,0010   0111
 1  2      7
```

* 6 eklenme sebebi; 9 rakam var 4 bit 15'e kadar gösterir. $15-9=6$ farkı kapatıyoruz

ASCI-II

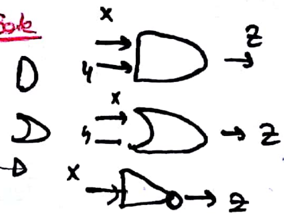
Even / Odd Priority

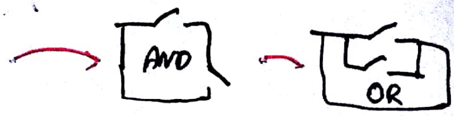
0......
1......

Ikili Lojik

Gate

→ And $\cdot$ $= x.y$
→ Or $+$ $= x+y$
→ Not $'$ $= x'$





Kendi Buldugum

Minterm

⇒ And. 2^n tane olur
- $xy, xy', x'y, x'y'$
- $m_0, m_1 \ldots$

* m için (000) = m_0
0: not
1 = kendisi

$m_2 = (010)$
$x'yz$

MaxTerm

⇒ Or, 2^n tane olur
- $x+y, x+y', x'+y, x'+y'$
- $M_0, M_1 \ldots$

* M için (000) = M_0
0: kendisi
1: not

$M_2 = (010)$
$x+y'+z$

$(M_0)' = m_0$
$(m_0)' = M_0$

→ $m_0 = x'y'z' \stackrel{!}{=} x+y+z = M_0$
→ $m_1 = x'y'z \stackrel{!}{=} x+y+z' = M_1$
→ $m_2 = x'yz' \stackrel{!}{=} x+y'+z = M_2$
→ $m_3 = x'yz \stackrel{!}{=} x+y'+z' = M_3$
→ $m_4 = xy'z' \stackrel{!}{=} x'+y+z = M_4$
→ $m_5 = xy'z \stackrel{!}{=} x'+y+z' = M_5$
→ $m_6 = xyz' \stackrel{!}{=} x'+y'+z = M_6$
→ $m_7 = xyz \stackrel{!}{=} x'+y'+z' = M_7$

* $\Sigma(0,1) = m_0 + m_1$
* $\Pi(0,2) = M_0 . M_2$

Kanonik form

Minterm Maxterm

⇒ Eğer $f: \Sigma(0,2,4,6) = \Pi(1,3,5,7)$

De Morgan's Law → $f' = \Pi(0,2,4,6) = \Sigma(1,3,5,7)$ (Maxterm, Minterm)

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

/* $f = x + yz$

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

$= (x+y+z.z') . (x+yy'+z)$ (1) (2)

(1) $(x+y+z).(x+y+z')$ (2) $(x+y+z).(x+y'+z)$

$x.x = x$ */

$= (x+y+z)(x+y+z')(x+y'+z)$
000 001 010

$= \Pi(0,1,2)$ */

* Kendi Formulum *

örnek (1) (2) → + olduğu için (1) ve (2) Σ (minterm) şeklinde olmalı

$f = x + y.z$

(1) x...... Minterm (m) ⇒ $x=1$, $y=?$, $z=?$ $(1??) = \begin{cases} 100 \\ 101 \\ 110 \\ 111 \end{cases}$ $\Sigma(4,5,6,7)$

* Tek eleman Σ de Π de olur.

0: x'
1: x

Maxterm (M) 0: kendi 1: not ⇒ $x=0$, $y=?$, $z=?$ $(0??) \begin{cases} 001 \\ 010 \\ 011 \\ 000 \end{cases}$ $\Pi(0,1,2,3)$

De Morgan's Law

(2) .. y.z Minterm ⇒ $x=?$, $y=1$, $z=1$ $(?11) = \begin{cases} 011 \\ 111 \end{cases}$ $\Sigma(3,7)$

* And olduğu için minterm olur (sadece) / [redacted]

0: not
1: kendisi

Maxterm (M) y.z ⇒ olduğundan M işlemi olmaz.

$\Sigma(3,7) = \Pi(0,1,2,4,5,6)$

De Morgan's Law

$f = (1) + (2) = \Sigma(4,5,6,7) + \Sigma(3,7)$

([redacted]) $= \Sigma(3,4,5,6,7) = \Pi(0,1,2)$

De Morgan's Law

* $\Sigma(7,7) = \Sigma(7)$

* Kendi Formülüm

örnek

$f = \Sigma(0,1,2,4,5,6)$

6 tane $\rightarrow 6 \geq 2^{2}$

$6 < 2^{3} \rightarrow$ 3 bit

3-2=1 harfli eleman mevcut olabilir.

(1,2,3)

* 1 varsa 1,2,3... harfli elemanlar olabilir.
2 varsa 2,3,....
3 varsa 3,....

* Analiz yapıldı. şimdi işleme geçiyoruz.

Mintermdeki bit sayısı - 1 harfli eleman sayısı
3-1

$\Rightarrow \Sigma(0,1,2,4,5,6) =$ (000) (001) (010) (100) (101) (110)

minterm olduğu için
0: not
1: kendisi

* 1 harfli için 2^{2} tane aynı bit lazım
* 2 harfli için 2^{1} tane aynı bit lazım
3-2=1
⋮

* Tüm terimleri bulana kadar arama işlemi (ortak bit bulma) devam edecek.
* Mümkün olduğunca fazla terimde bit ortaklığı bulunmaya çalışılır.
Sırasıyla; (3 bit için)
→ 4 terimde 1 ortak bit
2 terimde 2 ortak bit
1 terim 3 ortak bit

0 (000)
2 (010)
4 (100)
6 (110)
(??0) = z'

0 (000)
1 (001)
4 (100)
5 (101)
(?0?) = y'

Kullanılan Mintermler
0,1,2,4,5,6 ⟶ hepsi

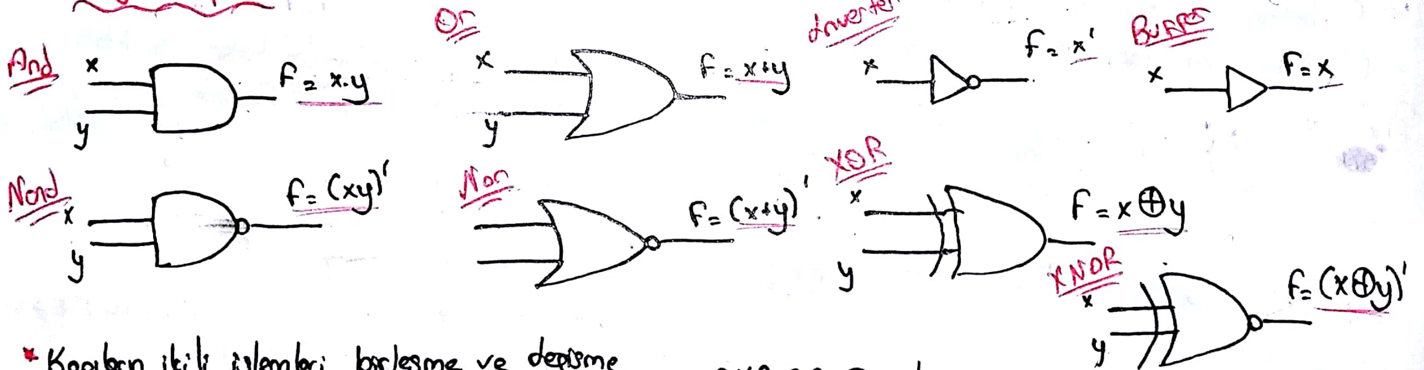
* Aynı min/max term birden fazla kullanılabilir.

$f = \Sigma(0,1,2,4,5,6) = \underline{z' + y'}$

Bit Sayısı

(4 bit için)
→ 8 terim 1 ortak 2^{3} (1)
→ 4 terim 2 ortak 2^{2} (2)
→ 2 terim 3 ortak 2^{1} (3)
→ 1 terim 4 ortak 2^{0} (4)

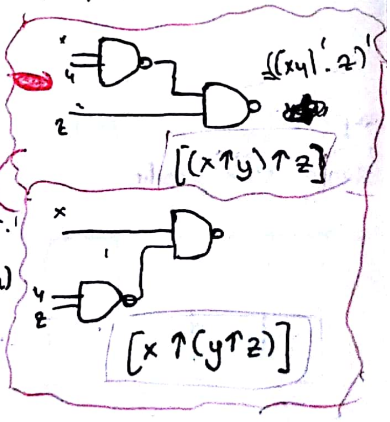
Lojik Kapılar



* Kapıların ikili işlemleri birleşme ve değişme özelliği sağlıyorsa [xy = yx , x.(x+y) = xx+xy] kapılar genişletilebilir.

→ AND, OR, İnverter, Buffer, ~~NAND, NOR~~ XOR, XNOR genişletilebilir.

$(x \downarrow y) \downarrow z \neq x \downarrow (z \downarrow y)$ genişletilmez.

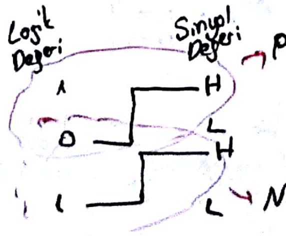


2'si farklıdır! (genişletilmez)

Genişletme

* XOR teklik belirler. (xyz) için XOR = 1 ise x,y,z 'den sadece 1 tanesi 1'dir.

(001)
(010)
(100)
$\Rightarrow$ XOR = 1



→ Pozitif Lojik (H:1 L:0)

→ Negatif Lojik (H:0 L:1)

Pozitif Or kapısı

negatif OR kapısı

The Map Method

x \ y	0	1
0		
1		

Two-Variable

x \ yz	00	01	11	10
0				
1				

Three-Variable

xy \ zt	00	01	11	10
00				
01				
11				
10				

Four-Variable

n-variable

* 2^n tane yanyana 1 → 1
  - Değişmeyen varsa yazılır. En sona kalan 1'ler yazılır
* 0 tane yanyana 1 (Hiç 1 yoksa) → 0

2^2 tane 1 → 1

2^1 tane 1 → (Değişmeyen Yazılır)

2^0 tane 1 → "

yanyana (* 0 tane 1 = 0)

örnek

x \ y	0	1
0	1	1
1		

→ 2^1 tane yanyana 1

* değişmez ve $x=0$ (Ama içeride 1 yazıyor)

⇒ $f = x'$

örnek

x \ y	0	1
0	1	
1		1

→ 2^0 tane yanyana 1 değişmeyen yok

$f = x'y' + xy$ (1) (2)

(1) $x=0$, $y=0$ 1 ⇒ $x'y'$ (2) $x=1$, $y=1$ 1 ⇒ xy

örnek

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

* Minterm için 1
* Maxterm için 0 (veya işlem yok)

ilk sütun.

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

$= \Pi(3,7,10,11,15)$ → Sabitler

1 1 1 1 } $y=0$ ⇒ y'

1 1 1 1 } $z=0$, $w=0$ ⇒ $z'w'$

1 1 1 1 } $z=0$, $x=1$ ⇒ $z'x$

* Kullanılmamış 1 kalmayana kadar, seçilebilen en fazla yanyana 1 seçilir. Seçimler ($2^0, 2^1, 2^2$ ....) tane 1 şeklinde olur.

$f = y' + z'x + z'w$

* f'i bulmak için table methodu 1 için Σ verir
* F' bulmak için table methodu 0 için Π verir

Aynı örnek için

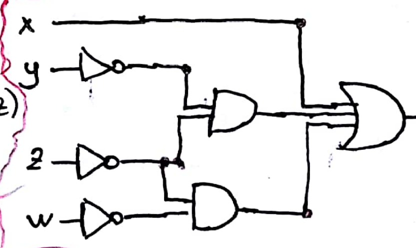
0 0 } $w=0$, $y=1$, $z=1$ ⇒ $(w+y'+z')$

0 0 } $w=1$, $y=1$, $z=1$ ⇒ $(w'+y'+z')$

0 0 } ⇒ $w'+x+y'+z$

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

$= \Pi(3,7,10,11,15)$



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

* F' bulmak için 0'lar 1, 1'ler 0 olur sonra işlem yapılır.

x \ yz	00	01	11	10
0	0	1 (001)	1 (011)	0
1	1 (100)	0	0	1 (110)

$f = x'z + xz'$

⇒

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

ilk sütun

f' için 1 → 0, 0 → 1

$x=0$, $z=0$ } $x'z'$

$x=1$, $z=1$ } xz

$f' = xz + x'z'$

$f' = \Sigma(0,2,5,7) = \Pi(1,3,4,6)$

→ Değerleri tabloya koyuyoruz.

$x'z = \{(001), (011)\}$ $x=0$, $z=1$

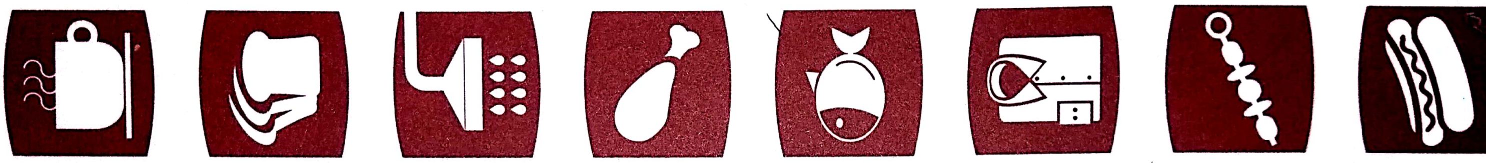
$xz' = \{(100), (110)\}$ $x=1$, $z=0$

tabloda 1 koyuyoruz

Tablodan çıkarıyoruz

m_1, m_3, m_4, m_6

$f = \Sigma(1,3,4,6) = \Pi(0,2,5,7)$



$f = \sum(0,1,2,4,5,6)$

0	(000)
1	(001)
2	(010)
4	(100)
5	(101)
6	(110)

0,1	00–
0,2	0–0
0,4	–00 ✓
1,5	–01 ✓
4,5	10–
4,6	1–0

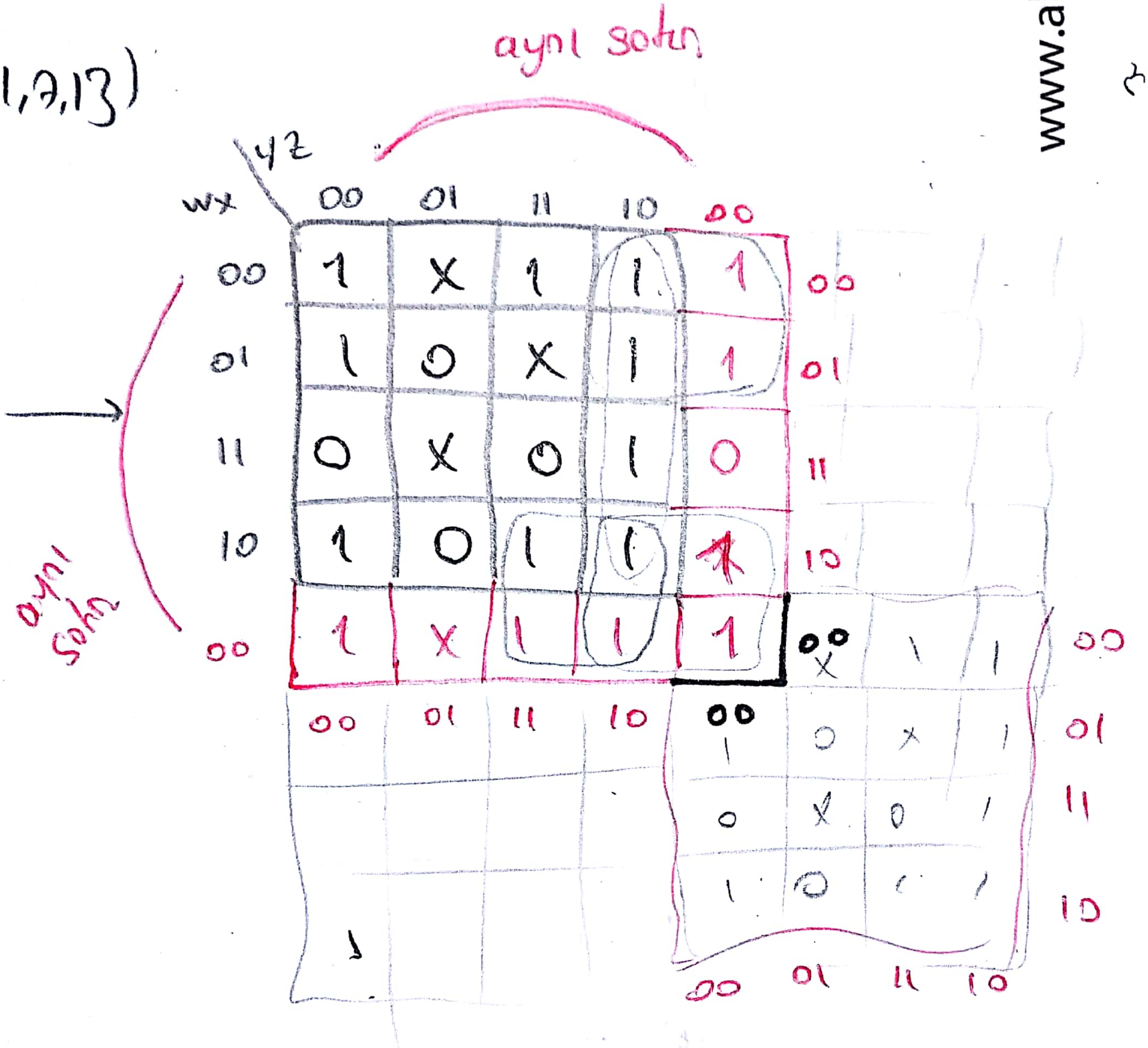
0,1,4,5 –0–
0,4,1,5 –0–
0,2,4,6 ––0

	0	1	2	4	5	6
✓ 0,1,4,5	✓	✓		✓	✓	
✓ 0,2,4,6	✓		✓	✓		✓

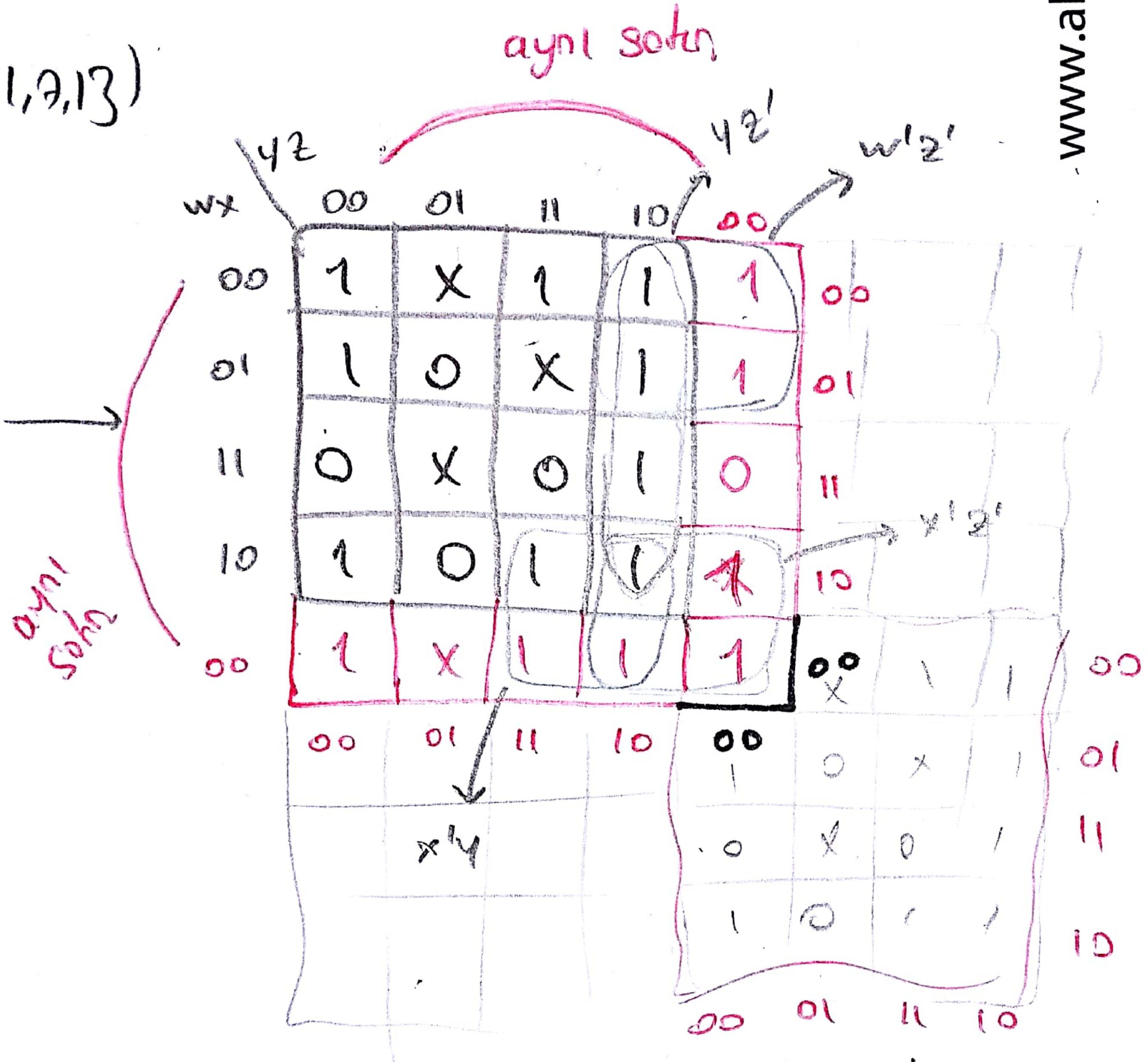
$F(x,y,z) = Y' + Z'$

$z) = \sum(1,7,13)$

2
6
14
10



$-\Sigma(1,7,13)$



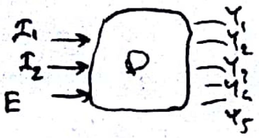
$$F(w,x,y,z) = w'z' + x'y + x'z' + yz'$$

Logic Final

Decoders

→ Kod çözücü

→ Enable ;



$E=0$ için hepsi 0

⇒ $E=1$ veya E'siz için

010 ⇒ $Y_2 = 1$

001 ⇒ $Y_1 = 1$

onluk değeri 1 diğerleri 0 yapar

* Min / Max termlere benziyor

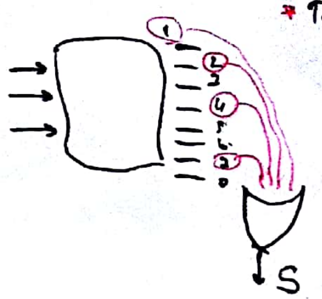
Example: Full Adder

$S = \Sigma(1,2,4,7)$

$C = \Sigma(3,5,6,7)$

* Aynı no'ya sahip Y'leri or kapısına gönderiniz

* Tersi de olur.



Encoder

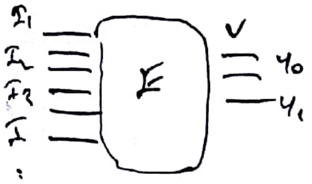
→ Decoder tersi

→ Priority :

$V=0$ ⇒ hepsi 0

$V.1 + Y_0.1 + Y_1.2 = a$

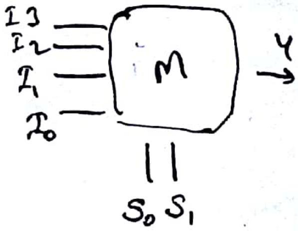
$I_a = 1 \quad I_2 = x \quad (2<a)$



Multiplexers

→ $S_0 + 2.S_1 = a$

$Y = I_a$



Example

$f(x,y,z) = \Sigma(1,2,6,7)$

(0,1) $I_0 = z$

(2,3) $I_1 = z'$

(4,5) $I_2 = 0$

(6,7) $I_3 = 1$

→ 000 ⇒ 0
001 ⇒ 1

Her zaman sonuncuyu alabalı olur.

DeMultiplexers

→ Multiplexers tersi

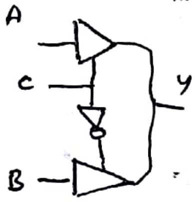
→ DeMUX

* DeMux = Decoder
* Mux = Encoder

Three State Gates

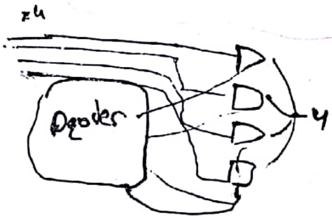
→ C=0 ⇒ Y: Hi-Gz

C=1 ⇒ A



C=0 ⇒ Y=B

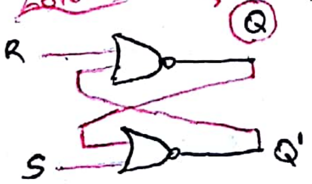
C=1 ⇒ Y=A



* Decoder ile kullanılabilir.

Latches

initial Value



NOR

→ R⊕S = 0 ⇒ Q → Q (no change)

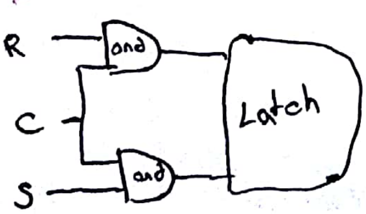
→ R⊕S = 1 ⇒ Q → S (0 = reset, 1 = Set)

→ R∧S = 1 ⇒ Q = Q' = 0 (invalid) (geçersiz)

NAND

→ Değiline ($\bar{R}.\bar{S}$) bakıp NOR tablosuna bakarız.

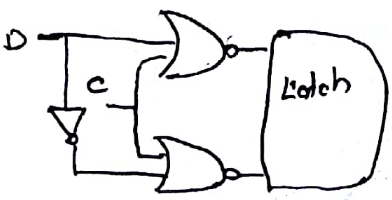
Controlled Latches



→ C=0 ⇒ Q=Q

→ C=1 ⇒ Latch

⇒ D-Latches



→ C=0 ⇒ Q→Q no change

→ C=1 ⇒ Q=D

* Zaman diyagramlarında C=1 durumları incelenir.

→ Latch = $S + Q\bar{R}$ $(S \wedge R \neq 1)$

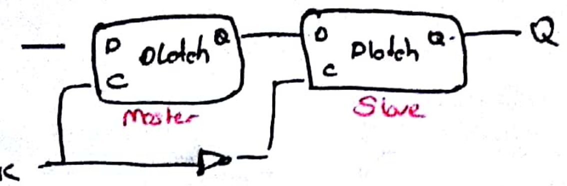
→ C-Latch = $\bar{Q}$ ~~$R + \bar{S}Q$~~ $+ CS$ $(C \wedge S \wedge R \neq 1)$

$\bar{C}Q + CS + \bar{R}\bar{S}Q$

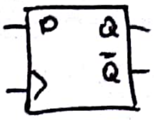
→ D-Latch : $\bar{C}Q + CD$

Flip Flop

-> Master-Slave D Flip-Flop

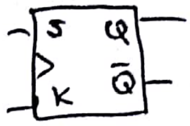


=> Edge-Triggered D flip-flop



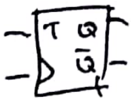
-> $Q(t+1) = D$

=> JK Flip-Flop



-> $J \vee K = 0 \Rightarrow Q(t+1) = Q(t)$
-> $J \oplus K = 1 \Rightarrow Q(t+1) = J$
-> $J \wedge K = 1 \Rightarrow Q(t+1) = \bar{Q}(t)$

=> T Flip-Flop

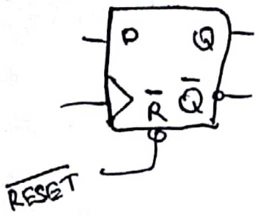


-> $T = 0 \Rightarrow Q(t+1) = Q(t)$
-> $T = 1 \Rightarrow Q(t+1) = \bar{Q}(t)$

C. Equations

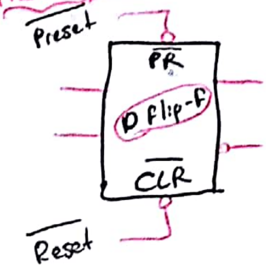
=> D Flip-flop $Q(t+1) = D$
=> JK f-f $Q(t+1) = J\bar{Q} + \bar{K}Q$
=> T f-f $Q(t+1) = T \oplus Q$
(Map Method)

Asynchronous Reset



-> $\bar{R} = 0 \Rightarrow Q(t+1) = 0$ (sıfır)
-> $\bar{R} = 1 \Rightarrow Q(t+1) = D$ (de)
-> $Q(t+1) = R.D$
-> $Q(t+1) = \Sigma(3) = M_3$
(R,D)

Preset and Clear

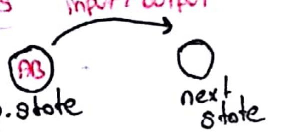


-> $(\overline{PR} \oplus \overline{CLR}) = 1 \Rightarrow Q(t+1) = \overline{CLR}$
-> $\overline{PR} \wedge \overline{CLR} = 1 \Rightarrow Q(t+1) = D$
-> $Q(t+1) = \cancel{\ldots}$
-> $(D, \overline{PL}, \overline{CLR}) = \cancel{\ldots}$

Analysis of Clocked Sequential Circuits

i: input o: output

State Diagrams -> p.state



P.State	Next State		Output	
A,B	i=0	i=1	i=0	i=1
	A,B..	A,B	0 = ...	0 = ...
	⋮			

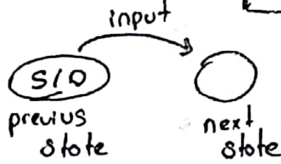
State Diagrams Table

Mealy
-> For the same state;
the output changes with the input.

A	B	input x	output y
0	1	0	1
0	1	1	0

$y(x)$, $y(A,B,x)$
y depended x

Moore State Diagrams ->



S: state (A,B)
O: output

Moore
-> For the same state;
the output does not changed with the input

A	B	input x	output y
0	1	0	1
0	1	1	1

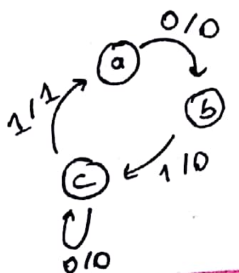
y dont depended x

Ex/ State: a b c c a
Input: 0 1 0 1
Output: 0 0 0 1

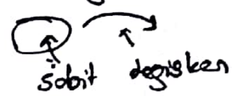
State Reductions

-> Only input and output sequences are important.

* Aynı olan stateleri kaldırıyoruz. (Sonraki kaldırılır.)



* State diagram'ın sabitleri ameri



ÖZET

-> 2'lik -> 10'luk
-> 10'luk -> 2'lik

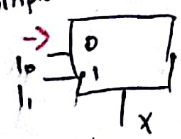
Decoders $\xleftrightarrow{\text{zıt}}$ Encoders

-> $Y_n : m_n$
-> $Y_n : \Sigma . m_n$

-> $Y_1(x,y,z) = \Sigma(2,4)$
-> $Y_2 = \Sigma(1,4)$
-> V varsa $V.Y_1$

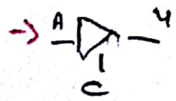
$I_1 I_2$
$(01) = 1$ $Y_1 = 1$
$(11) = 3$ $Y_3 = 1$
diğer = 0

Multiplexer



$x = 0 \Rightarrow I_0$
$x = 1 \Rightarrow I_1$

Three State Gates



$y = C.A$