BINARY NUMBERS

Binary numbers contain only 0's and 1's.

Ex: $(1\,1\,0\,1\,0\,1)_2 = 2^0 + 2^2 + 2^4 + 2^5 = 1 + 4 + 16 + 32 = (53)_{10}$

↑ base

conversion

Summation and Subtraction:

$$\begin{array}{r} 101101 \\ +\ 100111 \\ \hline 1010100 \end{array} \qquad \begin{array}{r} 101101 \\ -\ 100111 \\ \hline 000110 \end{array}$$

Product:

$$\begin{array}{r} 1011 \\ \times\ \ 101 \\ \hline 1011 \\ 0000 \\ +\ 1011 \\ \hline 110111 \end{array}$$

Complements: are used to show negative binary.

1's complement:

1's comp:

1011000 ⟶ 0100111

2's comp: Add 1 to 1's complement.

Ex: 11011000 —2's comp.→ $\begin{array}{r} 00100111 \\ +\ \ \ \ \ \ \ 1 \\ \hline 00101000 \end{array}$

Substraction with Complements:

The substraction of two numbers $M-N$

1) Add M to the 2's comp of N.

2) If $M \geq N$, the sum produces an end carry which is discarded.

3) If $M < N$, the sum does <u>not</u> produce a carry. To obtain the answer, take the 2's complement of the sum and place a negative in front.

Ex: Given

$X = 1010100$ and

$Y = 1000011$, perform the substraction

a) $X-Y=?$ b) $Y-X=?$ by using 2's complement

(a) $X = 1010100$

2's comp of $Y = 0111101$

$$\begin{array}{r} 1010100 \\ +\ 0111101 \\ \hline \cancel{1}0010001 \end{array}$$

end carry ↑ discard

answer $= 0010001$

(b) $Y = 1000011$

2's comp of $X = 0101100$

$$\begin{array}{r} 0101100 \\ +\ 1000011 \\ \hline 1101111 \end{array}$$

if it is 1 it is negative

Therefore the answer is $Y-X = -($ 2's comp of $1101111)$

OR

$= -0010001$

x(-)

Binary Logic

AND op:

Binary variable X and Y.

X.Y means X AND Y.

X	Y	X.Y
0	0	0
1	1	1
0	1	0
1	0	0

↑ truth table

OR op:

X	Y	X+Y
1	0	1
0	1	1
1	1	1
0	0	0

NOT op:

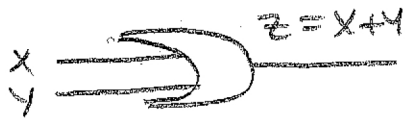
X	$X' = \overline{X}$
0	1
1	0

Logic Gates

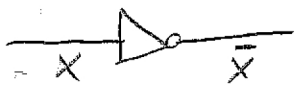
Logic gates are electronic circuits that operate with inputs to produce an output.

AND:

OR:



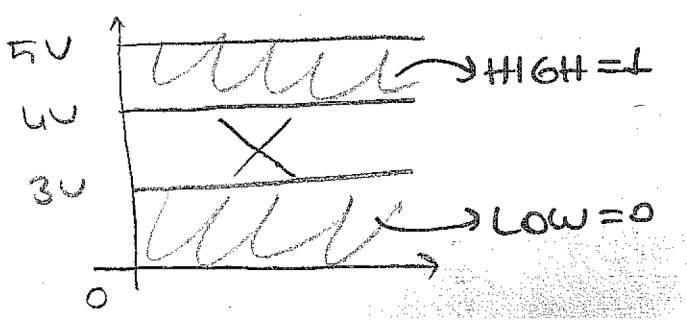
NOT:



Voltage = electrical energy

0 V → 0

5 V → 1



Boolean Algebra and Logic Gates

Binary Algebra

Theorems:

$X + 0 = X$

$X + X' = 1$

$X + X = X$

$X + 1 = 1$

$(X')' = X$

$X + Y = Y + X$

$X + (Y + Z) = (X + Y) + Z$

$X(Y + Z) = XY + XZ$

$(X + Y)' = X'Y'$

$X + XY = X$

$X \cdot 1 = X$

$X \cdot X' = 0$

$X \cdot X = X$

$X \cdot 0 = 0$

$XY = YX$

$X(YZ) = (XY)Z$

$X + YZ = (X + Y)(X + Z)$

$(XY)' = X' + Y'$

$X(X + Y) = X$

Operator Precedence:

Paranthesis → NOT → AND → OR

Boolean Functions:

$F1 = X + Y'Z$

function — input variables: X, Y, Z

Truth Table

X	Y	Z	F1
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Algebraic Manipulations:

Ex: Simplify the problem functions.

1- $X(X'+Y) = \underbrace{X \cdot X'}_{0} + XY = XY$

2- $X + X'Y = \underbrace{(X+X')}_{1}(X+Y) = X+Y$

3- $(X+Y)(X+Y') = XX + XY' + XY + YY'$ (OR, $YY' = 0$)

$X(1 + \underbrace{Y'+Y}_{1}) = X$

4- $XY + X'Z + YZ = XY + X'Z + YZ(X+X')$

$= XY + X'Z + XYZ + X'YZ$

$= XY(\underbrace{1+Z}_{1}) + X'Z(\underbrace{1+Y}_{1}) = XY + X'Z$

Complement of a Function

De Morgan's Theorem:

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

$$(ABCD\cdots F)' = A'+B'+C'+\cdots+F'$$

Ex: Find the complement of the function

$F_1 = X'YZ' + X'Y'Z$ and $F_2 = X(Y'Z' + YZ)$

$F_1' = (X'YZ' + X'Y'Z)' = (X'YZ')'(X'Y'Z)'$

$= (X+Y'+Z)(X+Y+Z')$

$F_2' = [X(Y'Z' + YZ)]' = X' + (Y'Z' + YZ)'$

$= (X' + (Y'Z')')(YZ)' = X' + YZ' + Y'Z$

Minterms and Maxterms

Given;

X	Y	Z	minterm	maxterm
0	0	0	$X'Y'Z' \rightarrow m_0$	$X+Y+Z \rightarrow M_0$
0	0	1	$X'Y'Z \rightarrow m_1$	$X+Y+Z' \rightarrow M_1$
0	1	0	$X'YZ'$	$X+Y'+Z$
0	1	1	$X'YZ$	$X+Y'+Z'$
1	0	0	$XY'Z'$	$X'+Y+Z$
1	0	1	$XY'Z$	$X'+Y+Z'$
1	1	0	XYZ'	$X'+Y'+Z$
1	1	1	XYZ	$X'+Y'+Z'$

Suppose a function

$$f_1 = X'Y'Z + XYZ' + XYZ$$

$$= m_1 + m_4 + m_7$$

If

$$f_1' = X'Y'Z + X'YZ' + X'YZ + XY'Z + XYZ'$$

$$= m_0 + m_2 + m_3 + m_5 + m_6 \quad \text{(sum of min terms)}$$

If

$$(f_1')' = (X+Y+Z)(X+Y'+Z')(X'+Y+Z')(X'+Y'+Z)(X+Y'+Z')$$

$$= M_0 \cdot M_2 \cdot M_3 \cdot M_5 \cdot M_6 \quad \text{(product of max terms)}$$

$$= f_1$$

Thus;

Any function f can be shown as SOP or POS form.

EX: $f_1 = \sum(1,4,7)$ (SOP)

or

$f_2 = \Pi(0,2,3,5,6)$ (POS)

Conversion;

$$\boxed{m_j' = M_j}$$

Standart foms: (SOP), (POS)

EX: In SOP form

$F_1 = y' + xy + x'y'z'$

find the minterms:

$F_1 = y'(\underbrace{xz + x'z + xz' + x'z'}) + xy(z+z') + x'y'z'$

$\underbrace{z(x+x')}_{1} + z'\underbrace{(x+x')}_{1}$

$\underbrace{z+z'}_{1}$

OR

$= \underbrace{xy'z}_{m_5} + \underbrace{x'y'z}_{m_1} + \underbrace{xy'z'}_{m_4} + \underbrace{x'y'z'}_{m_0} + \underbrace{xyz}_{m_7} + \underbrace{xyz'}_{m_6} + x'y'z'$

$F_1 = \sum(0,1,4,5,6,7)$

Exclusive OR

$$X+Y = XY' + X'Y$$

NOR

$(X+Y)'$

NAND

$(X \cdot Y)'$

Logic Gates

AND

OR

NOT
(inverter)

NAND

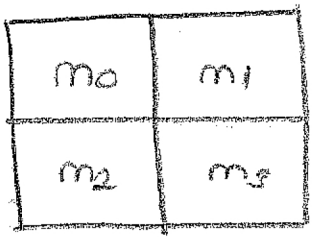
NOR

EX-OR

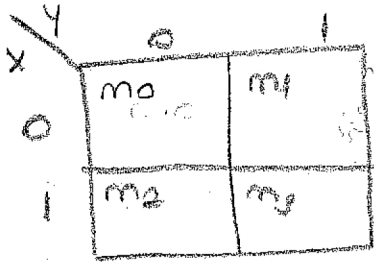
Gate Level Minimization
The Map Method

K-Map: Karnaugh Map

2-variable K-map:

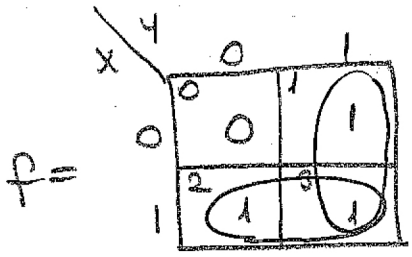


Inputs: X and Y



EX: Simplify the following function by K-map.

$$f = X'Y + XY' + XY = X+Y = \Sigma(1,2,3)$$



Grouping

1. Adjeisand cells
2. Power of 2. 2^n, n=# of groups
3. Large groups

$f = X + Y$

minterm olduklan için (+)

3_Variable K-map:

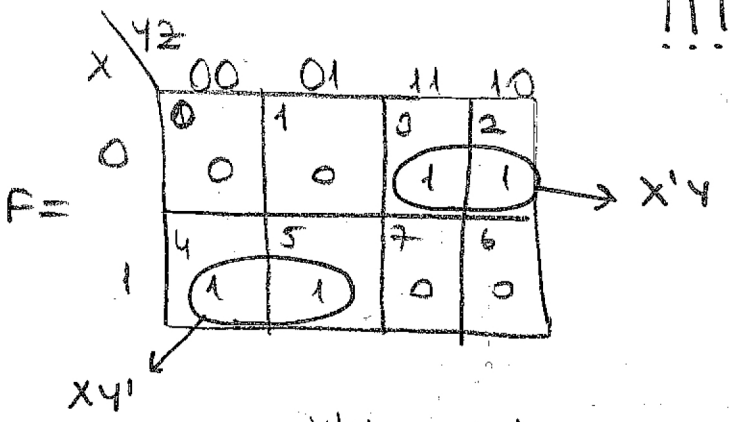
Ex: $F(X,Y,Z) = \Sigma(3,4,5)$ → minterm

simplify by K-map

Solution:

(at the given minterm you put 1.)

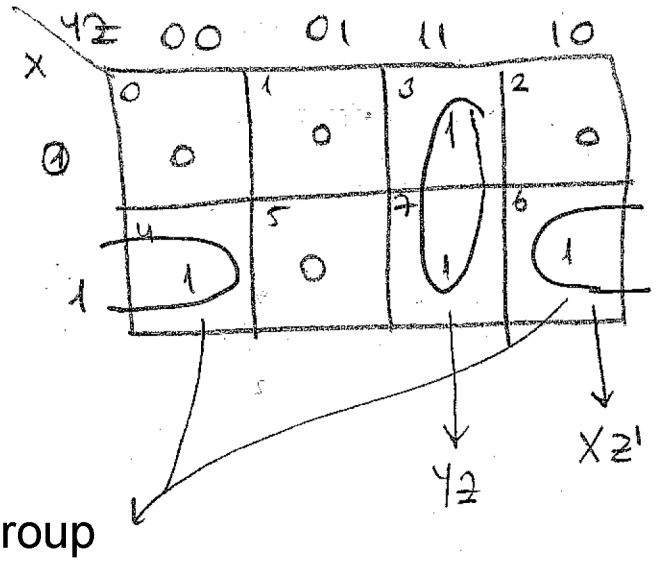
!!!!!



$= X'Y + XY' = X \oplus Y$

Ex: $F(X,Y,Z) = \Sigma(3,4,6,7)$

simplify by K-map.



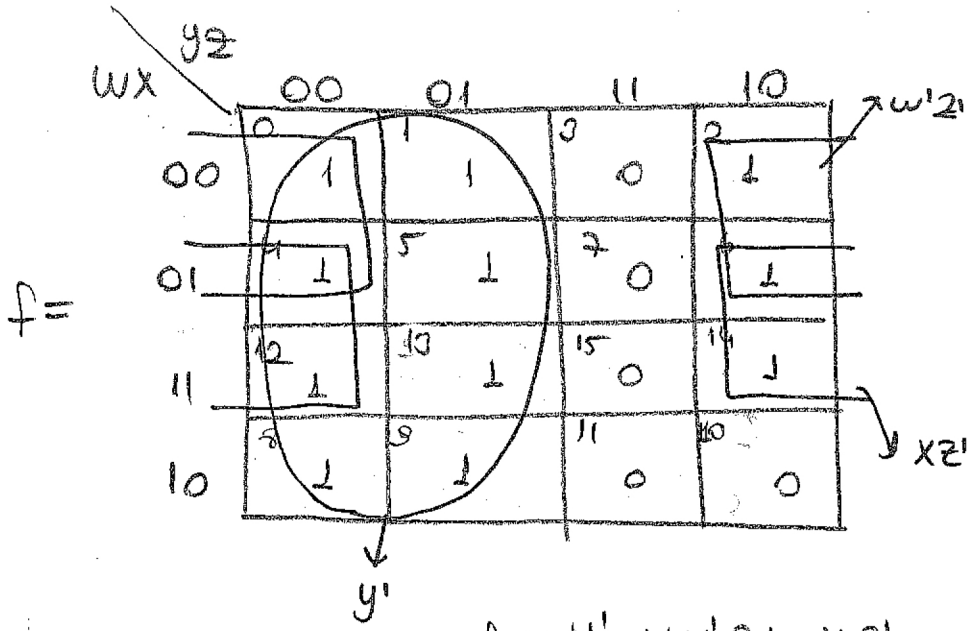
$= XZ' + YZ$

Group

4 - Variable K-map:

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

Simplify by K-map.

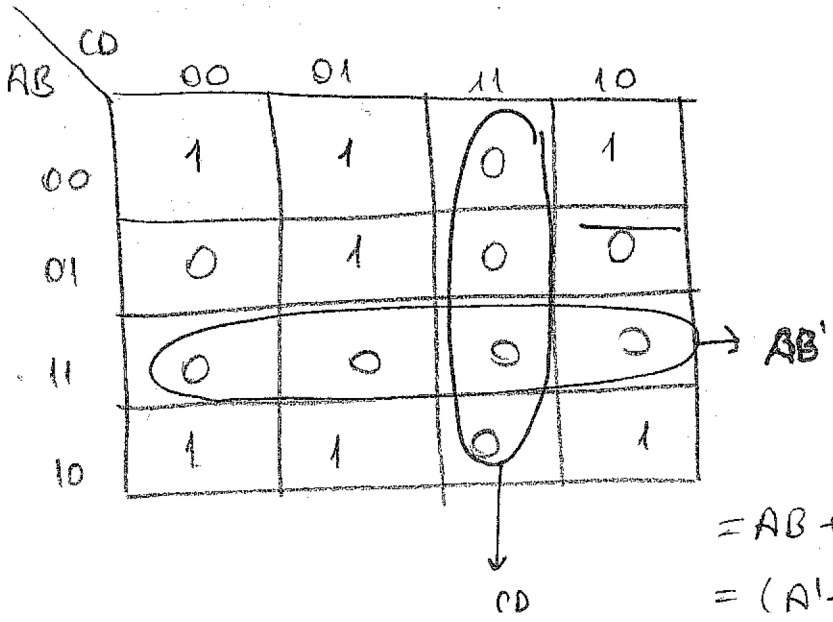


$f = y' + w'z' + xz'$

Ex: Simplify

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

maxterm in K-map



$= AB + CD + BD'$

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

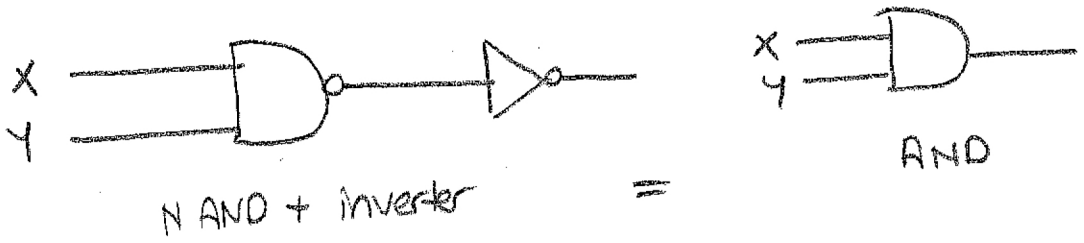
DON'T CARE

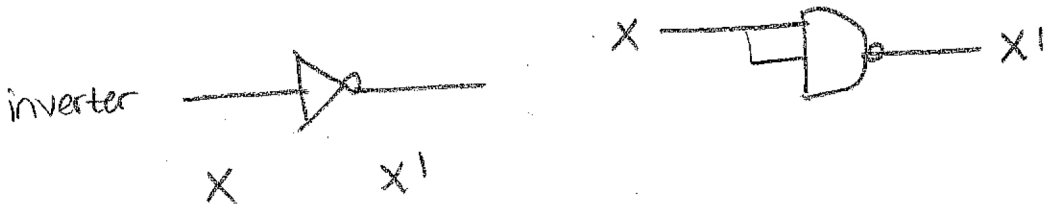
If the function value does not matter then we use don't care (x)

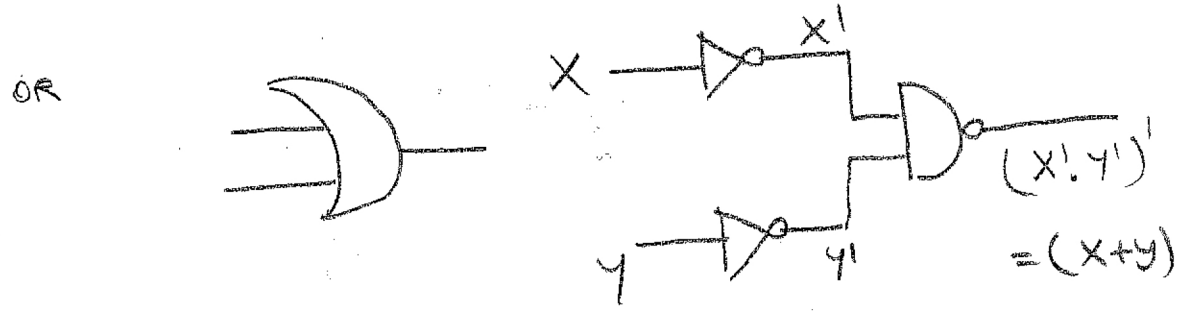
don't care can be grouped as you wish.

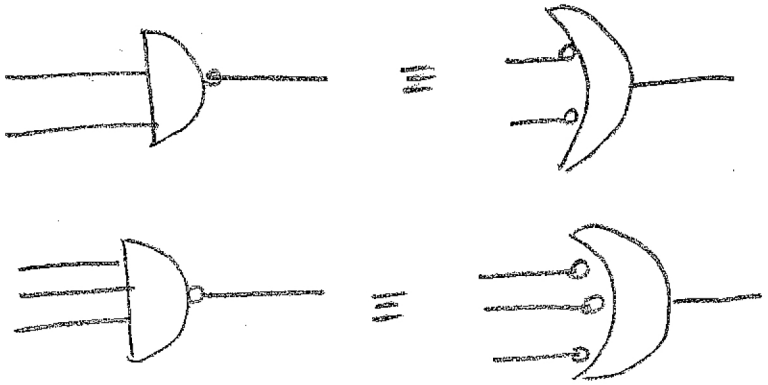
NAND Implementation

Circuits:





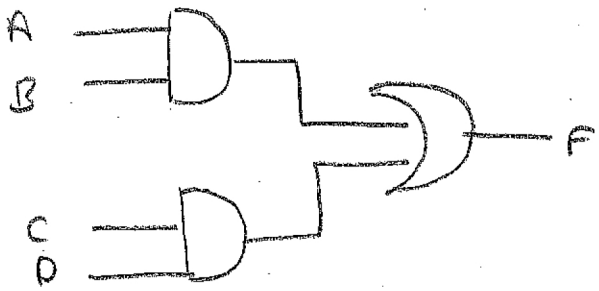


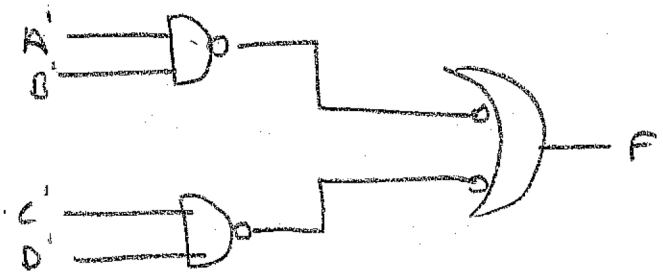


Ex: Implement the following function using NAND logic

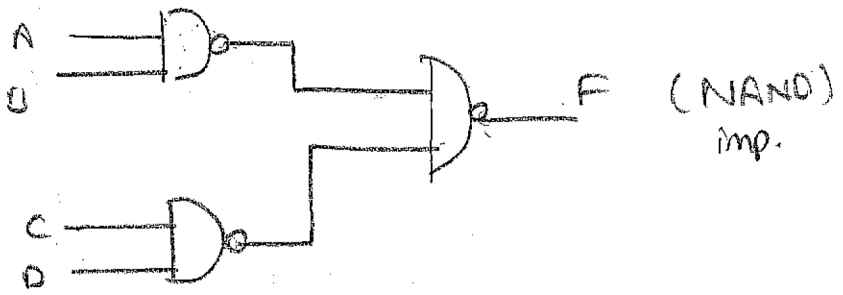
$$f = AB + CD$$

$$f = [(AB)' \cdot (CD)']' = [(A'.B')' \cdot (C'D')']$$

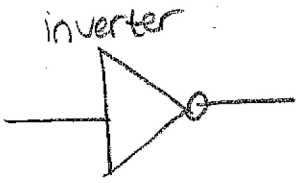


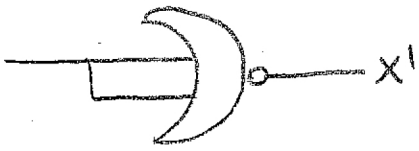


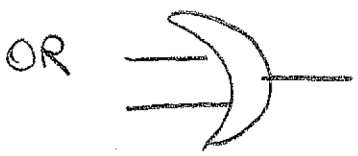
=

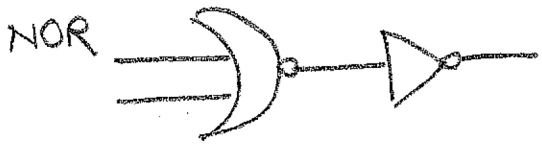


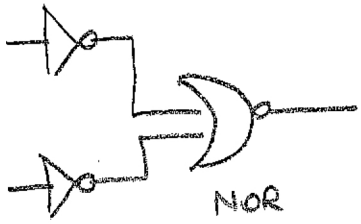
NOR implementation











Exclusive OR function

$$X \oplus y = Xy + X'y$$

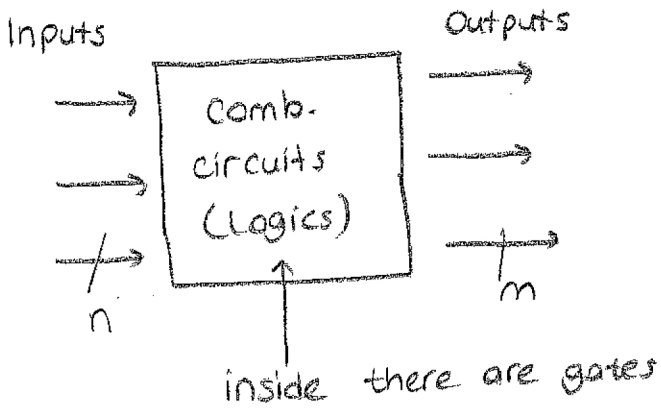
X	y	X+y
0	0	0
0	1	1
1	0	1
1	1	0

X	y	z	X+y+z
0	0	0	0
0	0	1	1
0	1	0	1
1	0	0	1
0	1	1	0
1	1	0	0
1	1	1	1
1	0	1	0

rule = even numbers=0

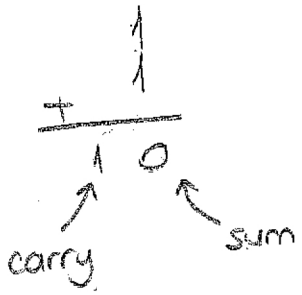
Combinational Logic

Consists of inputs, gates and outputs.



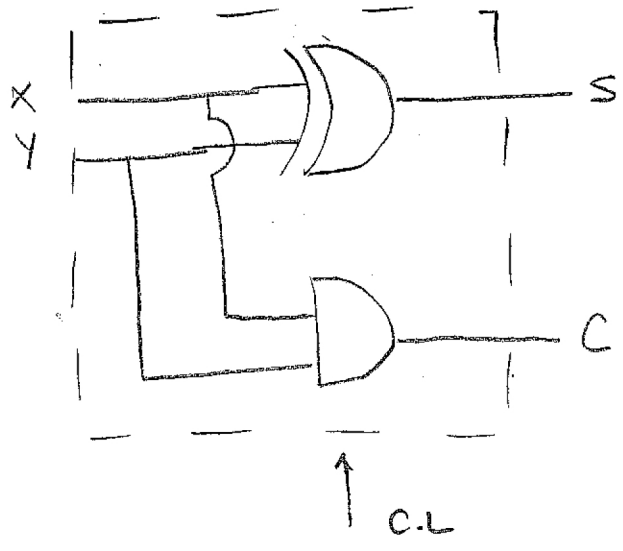
Half-Adder: (1 bit binary adder)

X	Y	Carry C	Sum S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



$S = X \oplus Y$ (OR)

$C = X \cdot Y$ (AND)



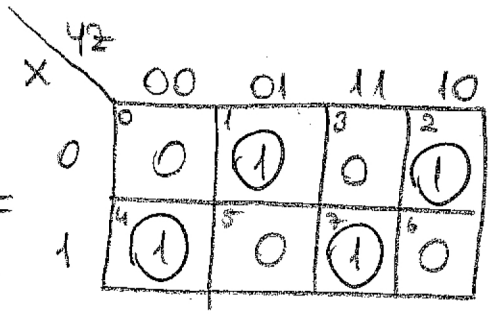
Full-Adder:

Add 3 1-bit binary number

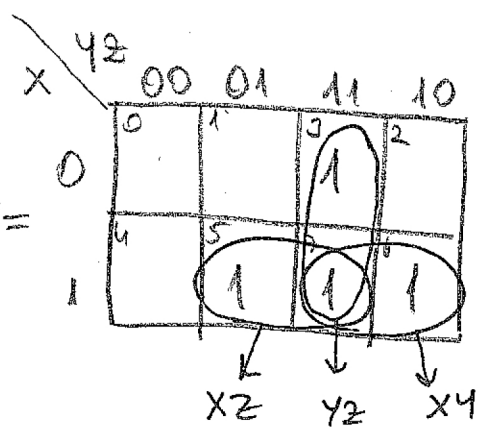
①

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

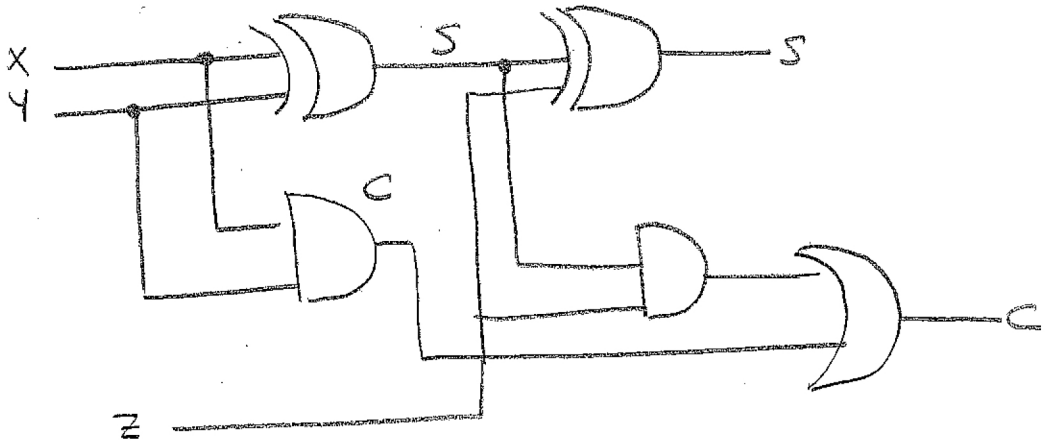
② Simplification:



$= X'Y'Z + X'YZ' + XY'Z' + XYZ$



$= XZ + YZ + XY$

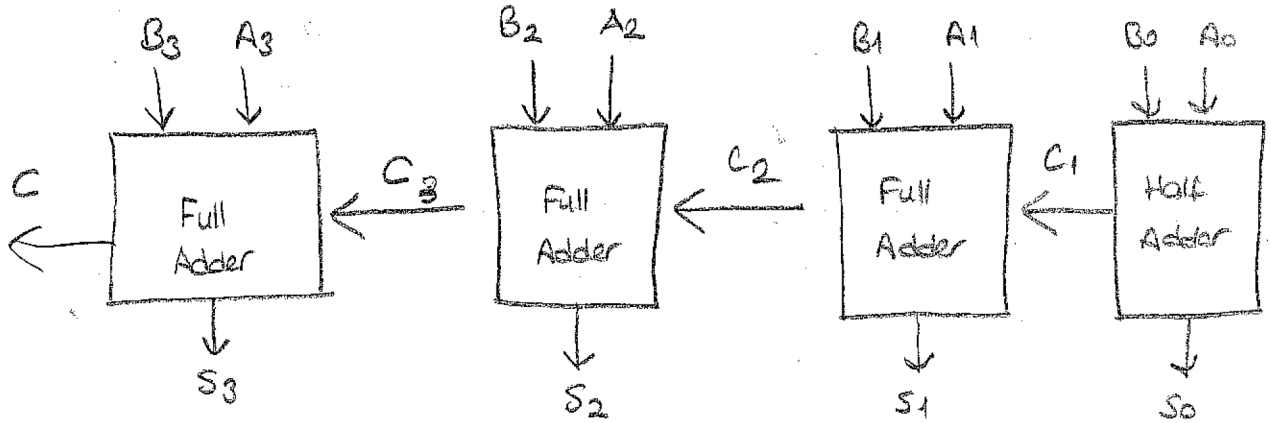


$$\begin{array}{r} X \\ Y \\ + \\ \hline C\ S \\ Z \\ + \\ \hline \end{array}$$

Addition of 2 4-bit Binary Numbers:

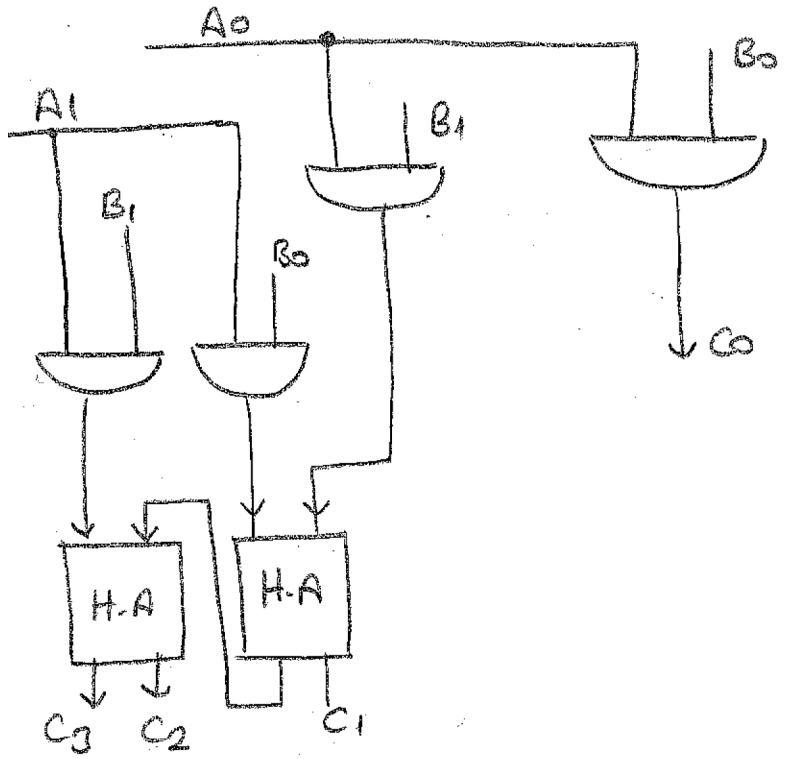
$$\begin{array}{r} C\ \ C\ \ C \\ A_3\ A_2\ A_1\ A_0 \\ B_3\ B_2\ B_1\ B_0 \\ + \\ \hline C\ S_3\ S_2\ S_1\ S_0 \end{array}$$

full adder — half adder



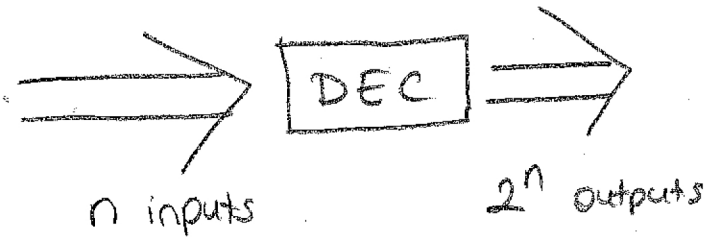
Binary Multiplier:

$$\begin{array}{r} B_1\ B_0 \\ \times\quad A_1\ A_0 \\ \hline A_0B_1\quad A_0B_0 \\ A_1B_1\quad A_1B_0 \\ \hline C_3\quad C_2\quad C_1\quad C_0 \end{array}$$



DECODERS

* It is a C.L circuit that converts n_inputs to max 2^n outputs.

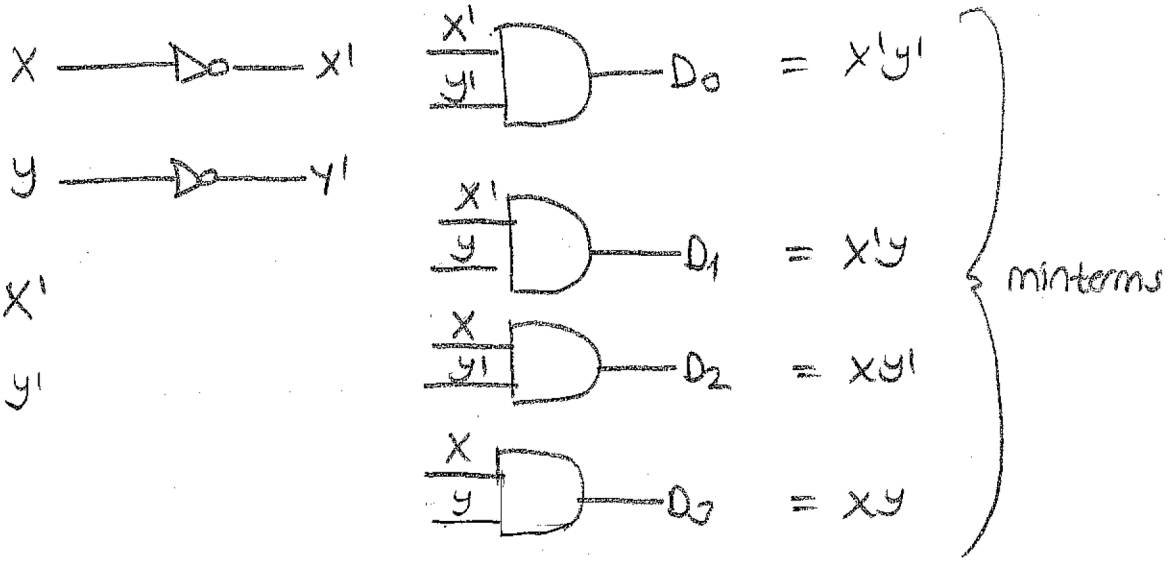


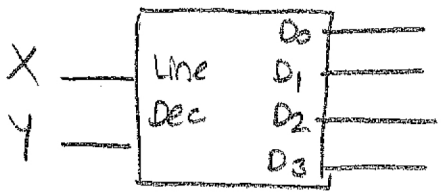
EX: Line-Decoders

Generate minterms

Given 2 inputs design a line-decoder

Ans:





$X_1\ X_2 \dots X_n$	$y_1 \dots y_m$
0 0 0 ...	...

C.L Implementation by Decoders:

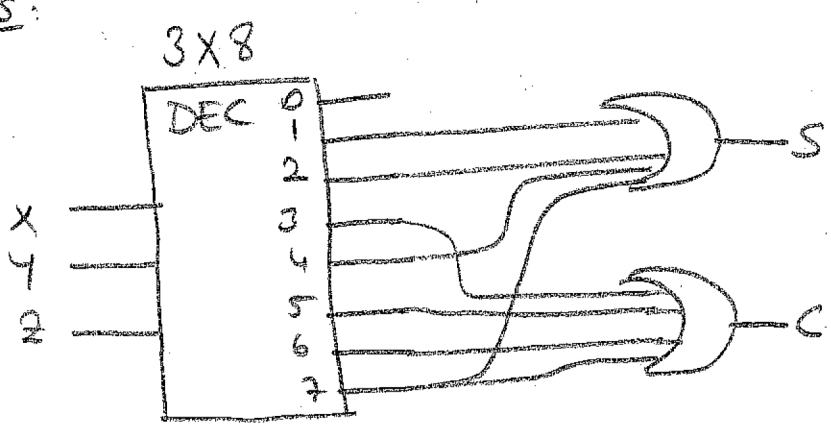
EX: Implement the C.L for

$$S(X,Y,Z) = \sum(1,2,4,7)$$

$$C(X,Y,Z) = \sum(3,5,6,7)$$

by using a line-decoder

Ans:



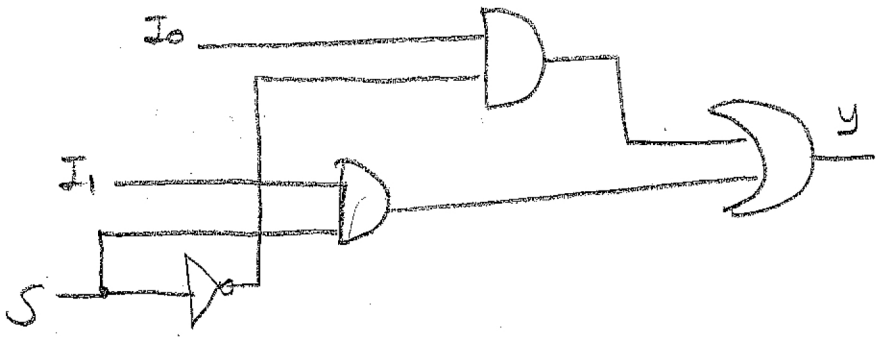
ENCODER:

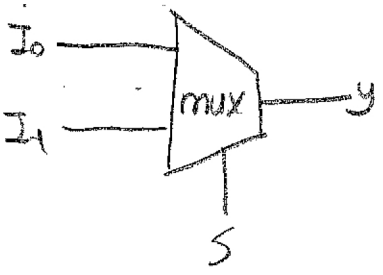
It's the opposite of decode

Multiplexer (mux):

It's a C.L. circuit that selects one of many inputs into a single output

Ex: 1X1 mux





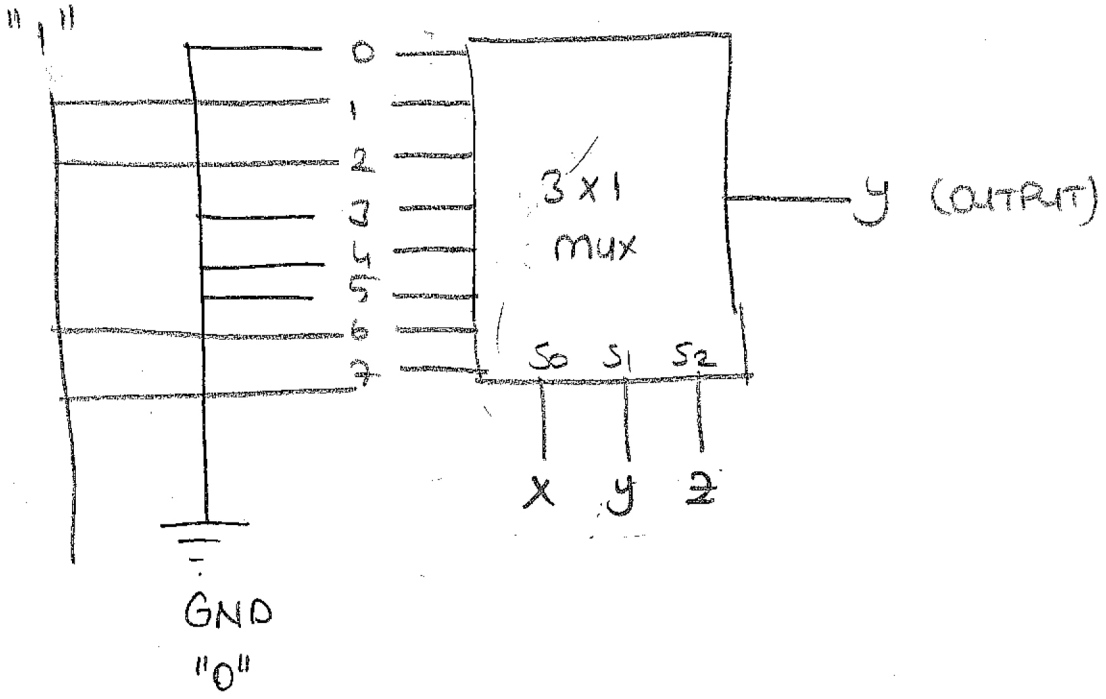
S	y
0	I_0
1	I_1

Boolean function Implementation

EX: (3) x (1) mux
selective output

implement

$F(x, y, z) = \sum (1, 2, 6, 7)$ by using a multiplexer



NAND Implementation:

EX:

$f_1 = AB' + A'C + A'BC'$

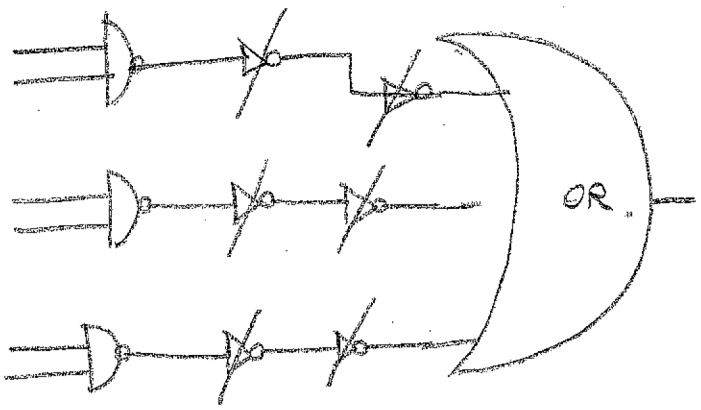
$= AB' + A'(C + BC')$

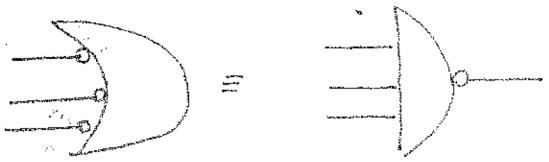
$(C+B)(C+C')$, $C+C' = 1$

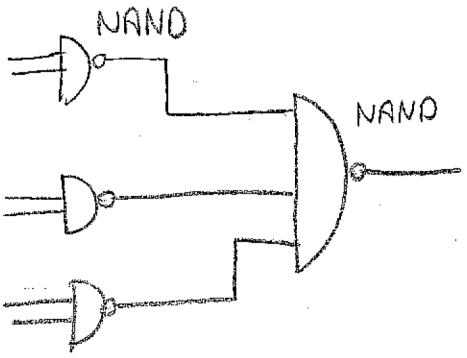
$= AB' + A'C + A'B = [(AB')' \cdot (A'C)' \cdot (A'B)']'$

(SOP)

if in sop form: then replace all the gates by the NAND gate.





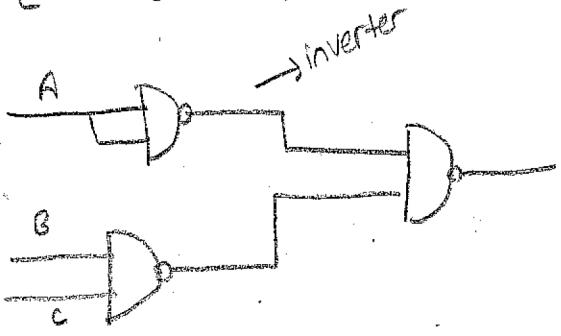


$f_2 = A + BC$

$= A.A + B.C$

$= [(A.A)' . (B.C)']'$

$= [A' . (B.C)']'$



$f_3 = (A'+C)(A+B')$ (POS form)

$= AA' + A'B' + CA + CB'$

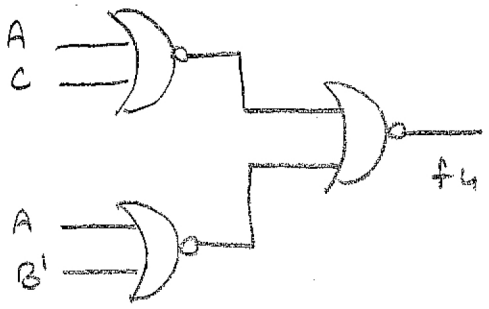
$= 0$

= Then proceed as usual.

NOR implementation:

$f_4 = (A+C).(A+B')$ (POS)

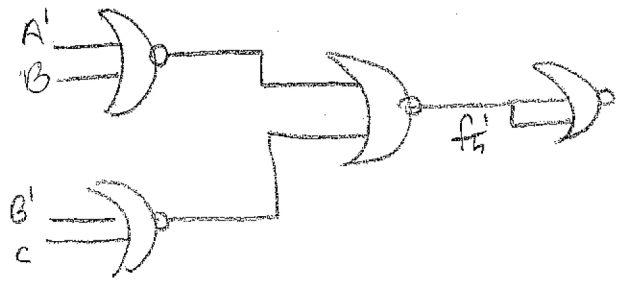
$f_4 = [(A+C)' + (A+B')']'$



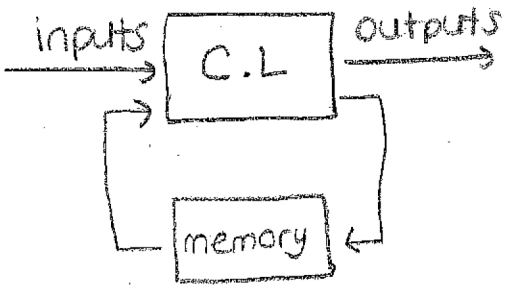
$f_5' = [AB' + BC']'$

$= (A.B')' . (BC')' = (A'+B)(B'+C)$ (POS)

$f_5' = [[(A'+B)' + (B'+C)']']'$ ← last inv.



_ Sequential Logic _



sequantial circuits contain memory.

Two type of sequencial circuits

Synchronous

All outputs are generated at the same time determined by a "clock signed".

Asyncr

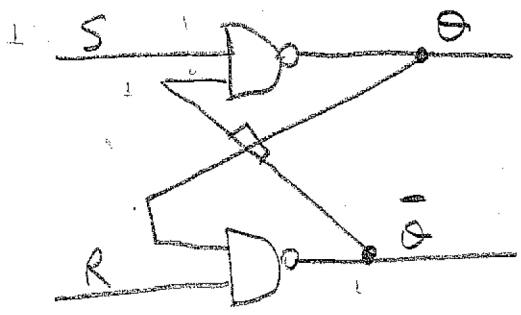
Not all outputs are generated at the same time

The storage element (memory) is flip-flop.

Latches: (First form of flip-flop)

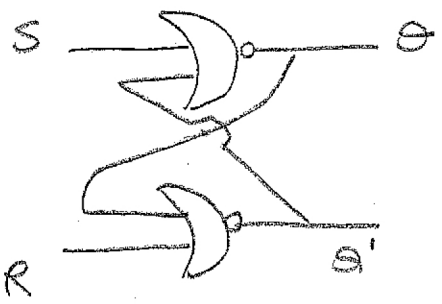
SR Latch.

Set Reset



S	R	Q	$\bar{Q}$	
1	0	0	1	(Set)
1	1	0	1	(Latch)
0	1	1	0	(Reset)
1	1	1	0	(Latch)
0	0	1	1	(forbidde

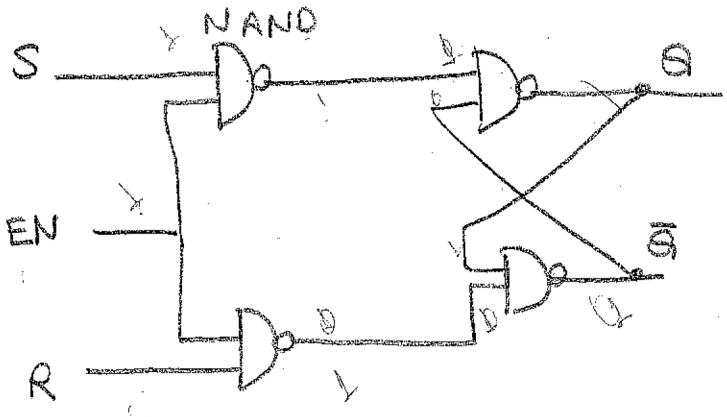
SR - NOR - Latch :



S	R	Q	Q'	
1	0	1	0	(Set)
0	0	1	0	(Latch)
0	1	0	1	(Reset)
0	0	0	1	(Latch)
1	1	0	0	

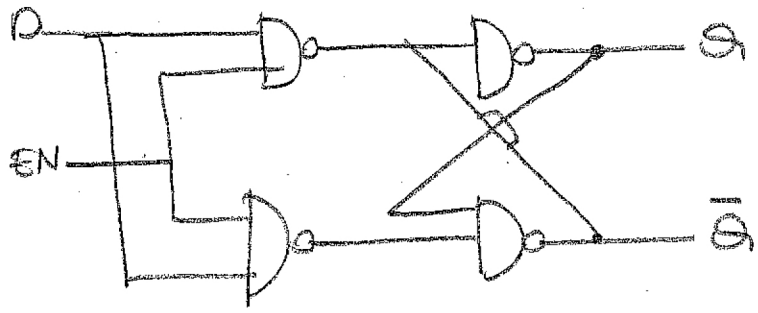
forbidden

SR - Latch with Enable:



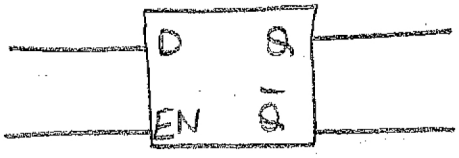
EN	S	R	Q	$\bar{Q}$	
0	X	X			No change (eski degeri tutar)
1	0	0			No change
1	0	1			Q = 0 (Reset)
1	1	0			Q = 1 (set)
1	1	1			Indetermined

D-Latch



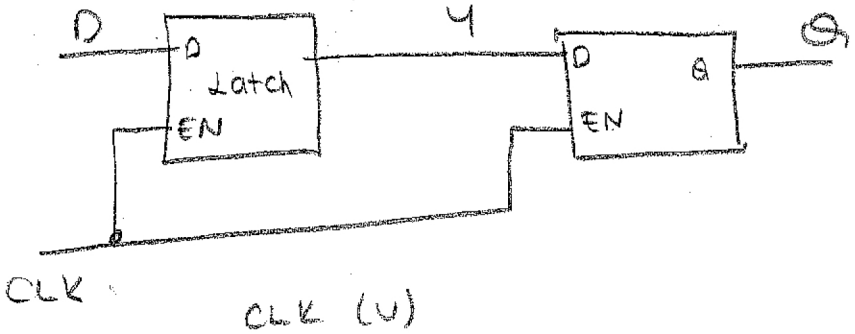
EN	D	Q
0	X	No change
1	0	0 (Reset)
1	1	1 (Set)

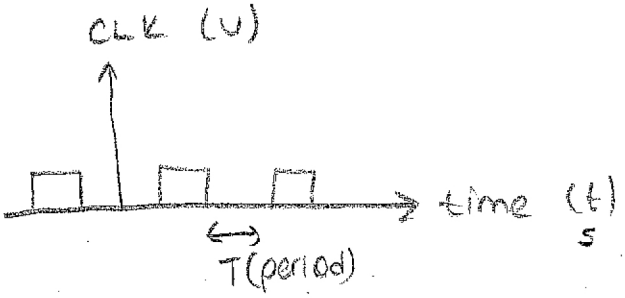
Circuit Symbole for D-Latch:



Circuit Symbole

D-Flip-Flop:



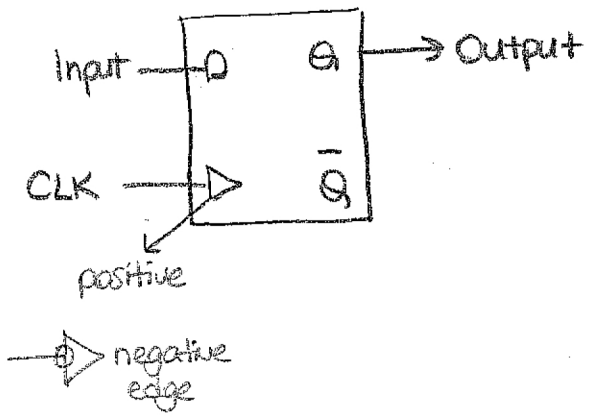


$\frac{1}{T}$ = frequence (Hz)

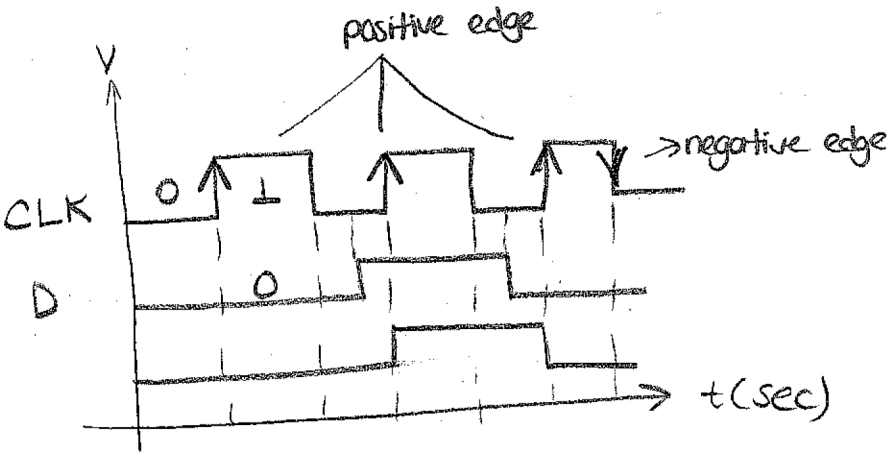
10^3 – kilo (k)	10^{-3} – mili (m)
10^6 – mega (M)	10^{-6} – micro (μ)
10^9 – giga (G)	10^{-9} – nano (n)
10^{12} – Tera (T)	10^{-12} – pico (p)

Edge Trigged D-ff:

D-flip flop



Ex:



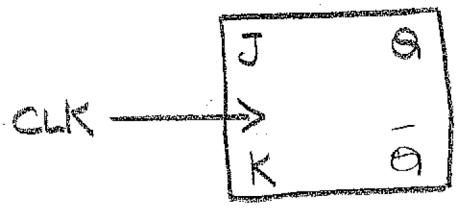
Draw the output (Q) wave to;

D	Q
0	0
1	1

(↑) → only at the raising edge (POS. EDGE)

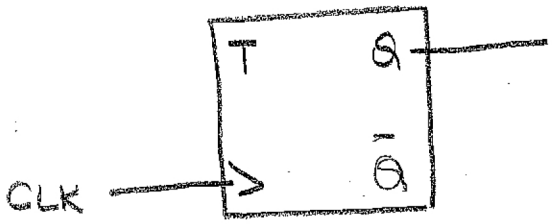
(↓) → (NEG. EDGE)

J-K Flip-Flop:



J	K	$Q(t+1)$ (next value)	
0	0	$Q(t)$ means previous value	NO CHANGE
0	1	0	RESET
1	0	1	SET
1	1	$\bar{Q}(t)$	COMPLEMENT

T-FF: (Toggle)



T	$Q(t+1)$	
0	$Q(t)$	NO CHANGE
1	$\bar{Q}(t)$	COMPLEMENT

- Characteristic Equations -

for D-ff:

$Q(t+1) = D$

for J-K:

$Q(t+1) = J.Q' + K'.Q$

for T-ff:

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

$= TQ' + T'Q$

Ex: Draw the output wave form given. (+ edge D-ff)