Course Code : 25CS102

Title : Digital Principles and System Design

Karnaugh Map

HDL- Hardware Discripting Language

Verilog HDL - Verification logging

Circuits:

1. Combination Logic circuits
2. Sequential Logic circuits

Latches, flip-flops - SR, D, JK,

Shift Register, Counters, Virtual Lab

Finite state machine:

Model, using D, JK and T flip flops -

state function, state table, Diagram.

Programmable Logic Devices:

Textbook:

1. M. Morris Mano "Digital Design"
2. Samir Palnitkar "Verilog HDL"
3. Ming-Bo Lin "Digital System Design and Practice"

Referrence Book:

1. Jr. Roth Charles H
2. John F Wakerly
3. Donald D. Givone

Boolean Function

$f = abc + a'b + bc' \Rightarrow$ SOP [Sum of Products]

Minimize by

- Boolean Laws - o/p Function
- Karnaugh Map - Unique o/p Function
- Tabulation Method - Proper o/p's

a, b, c - literals, variables

$f = (a+b)(b+c')(a+b') \Rightarrow$ POS [Product of Sums]

Types of System

Analog $\Rightarrow$ Continuous

Digital $\Rightarrow$ Discrete

1. Analog :

Time varying signals that can take any value within a given range.

2. Digitals :

Time varying signals that can take only one of the two discrete values of volts.

Benefits of digital over analog:

* Reproducibility
* Not effected by noise means quality
* Ease of Design
* Data protection
* Programmable
* Speed
* Economy

Representing Information:

Digital electronics represent information (0,1) with only two discrete values.

Information Representation:

(2,1) 2 bits (00, 01, 10, 11)

(4,2,1) 3 bits (000, 001, 010, 011, 100, 101, 110, 111)

(8,4,2,1) 4 bits (0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111)

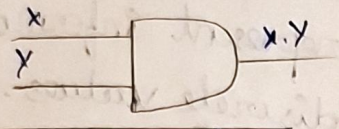
Gates:

* The most basic digital devices are called gates.
* Important gates: AND, OR, NOT
* It has one or more inputs and produces an output depending on the inputs.
* Output depends on the present inputs.

* Gates got their name → allowing or blocking the flow of digital information is called gating.
* It is known as Combinational circuit

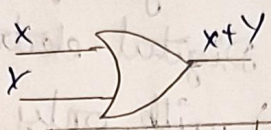
Digital logic:
* Binary system → 0 & 1, low and high
* Basic building blocks - AND, OR, NOT

AND:



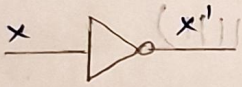
x	y	x.y
0	0	0
0	1	0
1	0	0
1	1	1

OR:



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

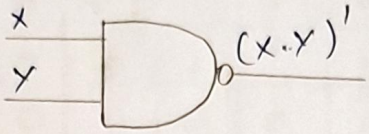
NOT:



x	x'
0	1
1	0

NAND:

$f = (x \cdot y)'$

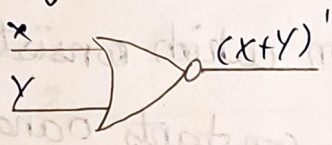


(AND - Inverted)

x	y	(x.y)'
0	0	1
0	1	1
1	0	1
1	1	0

NOR:

$f = (x+y)'$

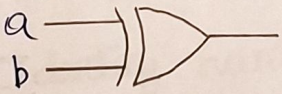


(OR - Inverted)

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

XOR:

$f = a'b + ab'$



a	b	a'	b'	a'b+ab'
0	0	1	1	0
0	1	1	0	1
1	0	0	1	1
1	1	0	0	0

XNOR:

$f = a'b' + ab$

a	b	a'	b'	a'b'+ab
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	1

BOOLEAN ALGEBRA

* Boolean function is described by an algebraic expression which consists of binary variables, the constants 0 and 1, and the logic operation symbols.

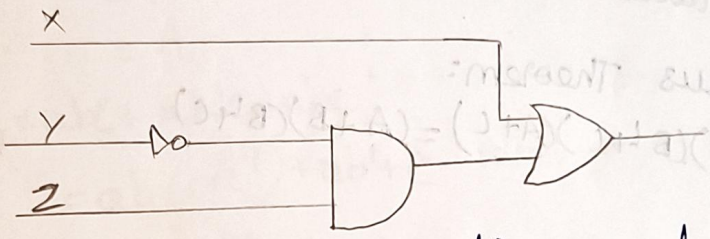
$F = x + y'z$

* It is equal to 1 if x is equal 1 or if both y' and z are equal to 1.
* F is equal to 0 otherwise
* Therefore e.g. $F = 1$
  * if $x = 1$ or if $y = 0$ and $z = 1$

* Boolean function can be represented in a truth table.

X	Y	Z	Y'	Y'Z	F=X+Y'Z
0	0	0	1	0	0
0	0	1	1	1	1
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	1	0	1
1	0	1	1	0	1
1	1	0	0	0	1
1	1	1	0	0	1

Circuit diagram



Identities of OR operation and AND operations:

$A+0=A$ $\quad$ $A\cdot 1=A$

$A+1=1$ $\quad$ $A\cdot A=A$

$A+\bar{A}=1$ $\quad$ $A\cdot 0=0$

$A+A=A$ $\quad$ $A\cdot A'=0$

Double inverse: $(\bar{\bar{A}}=A)$ or $(a')'=a$

Identity name	AND	OR
Identity law	$1\cdot X=X$	$0+X=X$
Null law	$0\cdot X=0$	$1+X=1$
Idempotent law	$X\cdot X=X$	$X+X=X$
Inverse law	$X\cdot\bar{X}=0$	$X+\bar{X}=1$

Identity Name	AND	OR
Commutative law	$XY = YX$	$X+Y = Y+X$
Associative law	$(XY)Z = X(YZ)$	$(X+Y)+Z = X+(Y+Z)$
Distributive law	$X+YZ = (X+Y)(X+Z)$	$X(Y+Z) = XY+XZ$

Identity Name	AND	OR
Absorption law	$X(X+Y) = X$	$X+XY = X$
Demorgan's law	$\overline{(XY)} = \bar{X}+\bar{Y}$	$\overline{(X+Y)} = \bar{X}\bar{Y}$
Double Complement law	$\overline{(\bar{X})} = X$	

Consensus Theorem:

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

Proof.

$$\begin{aligned}
LHS &= (A+B)(B'+C)(A+C) \\
&= (A+B)(B'+C)(A+C+0) \\
&= (A+B)(B'+C)(A+B.B'+C) \\
&= (A+B)(B'+C)(A+B+C)(A+B'+C) \\
&= [(A+B)(A+B+C)][(B'+C)(A+B'+C)] \quad \text{Absorption law} \\
&= (A+B)(B'+C)
\end{aligned}$$

LHS = RHS

1) Solve $a+a'b+ab$

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

$= a+b(1)$

$= a+b$

2) $(A+B)(A+B')$

$= AA+AB'+AB+BB'$

$= A+AB'+AB+0$

$= A+A(B'+B)+0$

$= A+A(1)$

$= A$

3) $f = abc+ab'c+abc'$

$= ab(c+c')+ab'+c$

$= ab+ab'c$

4) $AB+A(B+C)+B(B+C)$

$= AB+AB+AC+BB+BC$

$= AB+AC+B+BC$

$= AB+B+AC+BC$

$= B(A+1)+AC+BC$

$= B+AC+BC$

$= B(C+1)+AC$

$= AC+B$

	Y	X
$Y'.X' \leftarrow$	0	0
$Y.X' \leftarrow$	1	0
$Y'.X \leftarrow$	0	1
$Y.X \leftarrow$	1	1

5) $f = (AB + C)(AB + D)$

$= AB + ABD + ABC + DC$

$= AB + AB(D + C) + DC$

$= AB(1 + (D + C)) + DC$

$= AB + DC$

6) $f = [(a+b+c).d]'$

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

$= a' + b' + c' + d'$

7) $f = [(a+b').(a'+b)]'$

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

$= a'b + ab'$

Canonical Forms

Definition:

* Minterm: a product term
* Maxterm: a sum term

Minterm (AND Terms):

Two literals

x	y	
0	0	$\rightarrow x'.y'$
0	1	$\rightarrow x'.y$
1	0	$\rightarrow x.y'$
1	1	$\rightarrow x.y$

m_i - decimal equivalent of the minterms

a	b	c	f	Three variables xyz
0	0	0	1	$x'.y'.z'$
0	0	1	0	$x'.y'.z$
0	1	0	1	$x'.y.z'$
0	1	1	0	$x'.y.z$
1	0	0	0	$x.y'.z'$
1	0	1	1	$x.y'.z$
1	1	0	0	$x.y.z'$
1	1	1	1	$x.y.z$

Minterms (4 2 1)

m_0, m_2, m_5, m_7

$a'.b'.c'$

$a'.b.c'$

$a.b'.c$

$a.b.c$

Maxterm (OR Terms):

Two Variables

x	y	Minterm	Maxterm
0	0	$x'.y'$	$x+y$
0	1	$x'.y$	$x+y'$
1	0	$x.y'$	$x'+y$
1	1	$x.y$	$x'+y'$

a	b	c	f	Three variables xyz Maxterm
0	0	0	1	$x+y+z$
0	0	1	0	$x+y+z'$
0	1	0	1	$x+y'+z$
0	1	1	0	$x+y'+z'$
1	0	0	0	$x'+y+z$
1	0	1	1	$x'+y+z'$
1	1	0	0	$x'+y'+z$
1	1	1	1	$x'+y'+z'$

421

M_1^0

M_1	$x+y+z'$
M_3	$x+y'+z'$
M_4	$x'+y+z$
M_6	$x'+y'+z$

Question

$F = ab + cd + bc'$

Generate the Minterms and Maxterms

Solution

a	b	c	d	c'	ab	cd	bc'	f = ab+cd+bc'
0	0	0	0	1	0	0	0	0
0	0	0	1	1	0	0	0	0
0	0	1	0	0	0	0	0	0
0	0	1	1	0	0	1	0	1
0	1	0	0	1	0	0	1	1
0	1	0	1	1	0	0	1	1
0	1	1	0	0	0	0	0	0
0	1	1	1	0	0	1	0	1
1	0	0	0	1	0	0	0	0
1	0	0	1	1	0	0	0	0
1	0	1	0	0	0	0	0	0
1	0	1	1	0	0	1	0	1
1	1	0	0	1	1	0	1	1
1	1	0	1	1	1	0	1	1
1	1	1	0	0	1	0	0	1
1	1	1	1	0	1	1	0	1

Mintems:

$m_3, m_4, m_5, m_7, m_{11}, m_{12}, m_{13}, m_{14}, m_{15}$

$a+b+c+d'$, $a+b'+c+d$, $a+b'+c+d'$, $a+b'+c'+d'$, $a'+b'+c+d$, $a'+b+c'+d'$, $a'+b'+c+d$, $a'+b+c+d'$, $a'+b'+c'+d'$

$F = \Sigma m\,(3,4,5,7,11,12,13,14,15)$

$F = \Sigma(m_3, m_4, m_5, m_7, m_{11}, m_{12}, m_{13}, m_{14}, m_{15})$

Maxterms:

$M_0, M_1, M_2, M_6, M_8, M_9, M_{10}$

$(a'b'c'd'), (a'b'c'd), (a'b'cd'), (a'bcd'),$
$(ab'c'd'), (ab'c'd), (ab'cd')$

$F = \Pi_M(0,1,2,6,8,9,10)$

Canonical Forms (unique)

Any boolean function F () can be expressed as a SOP and POS.

* Canonical Sum of Products (Sum of minterms)
* Canonical Product of Sums (Product of maxterm)

Minimizing SOP Expression

A	B	C	F
0	0	0	0
0	0	1	1 ✓
0	1	0	1 ✓
0	1	1	0
1	0	0	0
1	0	1	1 ✓
1	1	0	0
1	1	1	1 ✓

Minterms:

$F = \Sigma_m (1, 2, 5, 7)$

$F = \Sigma (m_1, m_2, m_5, m_7)$

$F = a'b'c + a'bc' + ab'c + abc$

Maxterms

$F = \Pi_M (0, 3, 4, 6)$

$F = \Pi (M_0, M_3, M_4, M_6)$

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

Conversion between canonical forms:

1) $f = a'b' + ab \Rightarrow$ SOP

$= \Sigma_m (0, 3)$

$= \Pi_M (1, 2)$

2) $f = a'bc + abc' + abc \Rightarrow$ SOP

$= \Sigma_m (3, 6, 7)$

$= \Pi_M (0, 1, 2, 4, 5)$

3) $f = a'b'c + abc + a'bc'$

$= \Sigma_m (1, 2, 7)$

$= \Pi_M (0, 3, 4, 5, 6)$

1) $f = \Pi_M (0, 3, 7, 14, 15)$

$f = \Sigma_m (1, 2, 4, 5, 6, 8, 9, 10, 11, 12, 13)$

2) $f = (a+b+c+d)(a+b+c'+d)(a+b'+c'+d)$
$(a'+b'+c'+d')$

$f = \Pi_M(0, 6, 10, 15)$

$f = \Sigma_m(1, 2, 3, 4, 5, 7, 8, 9, 11, 12, 13, 14)$

Conversion of SOP from standard to canonical form:

1) $f = ab + abc'$

$f = ab(c+c') + abc'$

$= abc + abc' + abc'$

$= abc + abc'$

$f = \Sigma_m(6, 7)$

2) $f = a'b + b'c + c'd$

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

$= a'bc + a'bc' + ab'c + a'b'c + ac'd + a'c'd$

$= a'bc(d+d') + a'bc'(d+d') + ab'c(d+d')$
$+ a'b'c(d+d') + ac'd(b+b') + a'c'd(b+b')$

$= a'bcd + a'bcd' + a'bc'd + a'bc'd' + ab'cd +$
$ab'cd' + a'b'cd + a'b'cd' + ab'c'd + ab'c'd$
$+ a'bc'd + a'b'c'd$

$= a'bcd + a'bcd' + a'bc'd + a'bc'd' +$
$ab'cd + ab'cd' + a'b'cd + a'b'cd'$
$+ abc'd + ab'c'd + a'b'c'd$

$f = \Sigma_m(7, 6, 5, 4, 11, 10, 3, 2, 13, 9, 1)$

$f = \Pi_M(0, 8, 12, 14, 15)$

3) $f = a'b + bc' + ab'c'$

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

$= a'bc + a'bc' + abc' + a'bc' + ab'c'$

$f = a'bc + a'bc' + abc' + ab'c'$

$f = \Sigma_m(3,2,6,4)$

$f = \Pi_M(0,1,5,7)$

4) $f = (a+b)(a'+b)(a'+c)$

$= (a+b+cc')(a'+b+cc')(a'+c+bb')$

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

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

$f = \Sigma_m(2,3,7)$

5) $f = (a+b'+c).(b'+c+d)(a'+b+c)(b+c'+d)$

$= (a+b'+c+dd')(b'+c+d+aa')(a'+b+c+dd')$
$(b+c'+d+aa')$

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

$f = \Pi_M(4,5,12,8,9,2,10)$

$f = \Sigma_m(0,1,3,6,7,11,13,14,15)$

$f(a,b,c) = (a+b+c)\cdot(b'+c')\cdot(a'+c')$

$= (a+b+c)(aa'+b'+c')(ba'+c'+bb')$

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

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

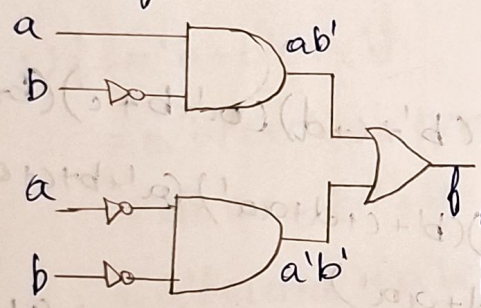
$f = \Pi_M(0,3,7,5) = \Pi_M(0,3,5,7)$

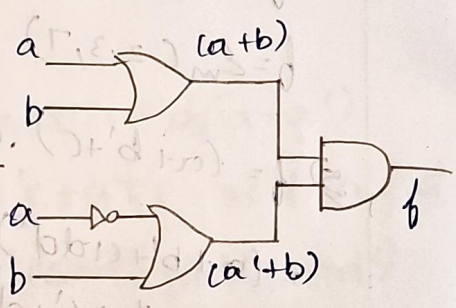
$f = \Sigma_m(1,2,4,6)$

Canonical forms:

- SOP
- POS

$f = ab' + a'b'$ $\qquad$ $f = (a+b)(a'+b)$





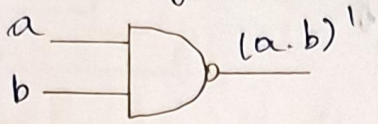
Number System

1. Binary (0,1)
2. Octal (0,1,2,3,4,5,6,7)
3. Decimal (0,1,2,3,4,5,6,7,8,9)
4. Hexadecimal (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F)

Implementation using universal logic Gates

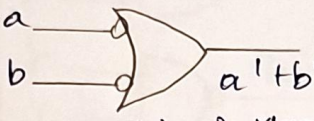
1. NAND Implementation (AND-INV)
2. NOR Implementation (OR-INV)

NAND Diagram:



(AND-INV)

or



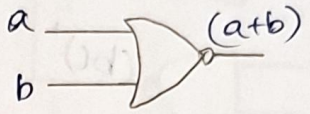
Using De-Morgan's law

$f = (a'+b')$

$f = (a.b)'$

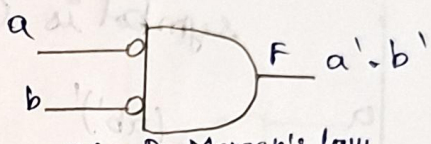
(INV-OR)

NOR Diagram:



(OR-INV)

OR



Using De-Morgan's law

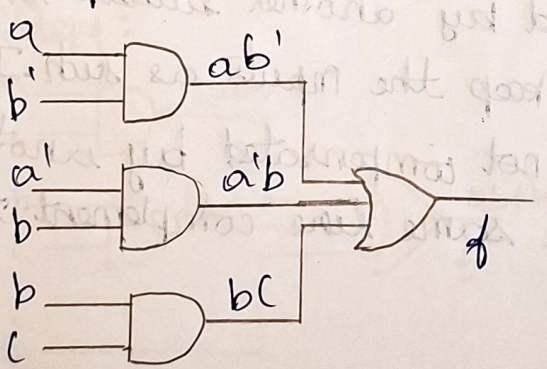
$f = (a'.b')$

$f = (a+b)'$

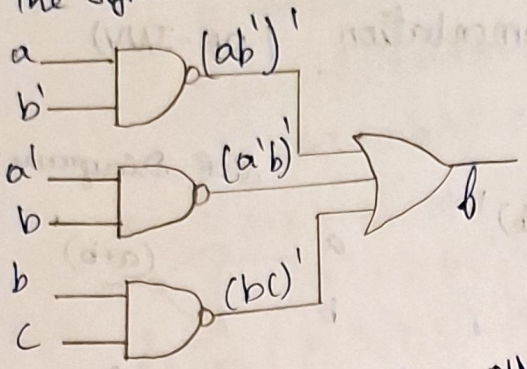
NAND Implementation

$$f = ab' + a'b + bc$$

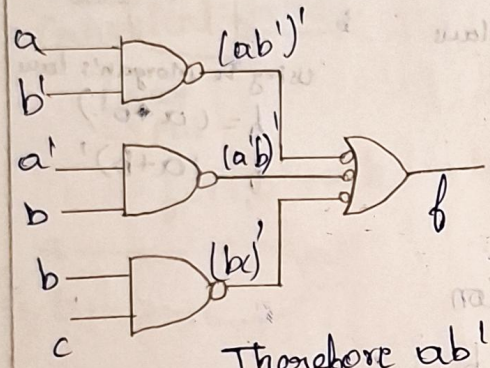
Step 1: (AND, OR)



Step2: Replace AND gate with NAND gate:
The symbol is (AND-INV)



Step3: Replace OR gate with NAND gate. The symbol is (INV-OR)

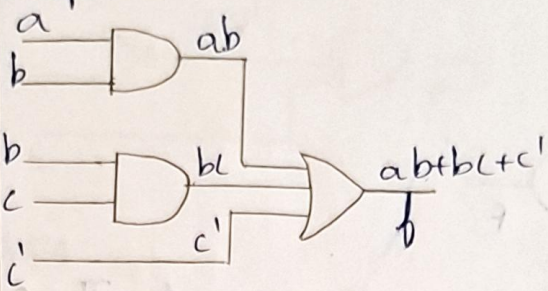


Therefore $ab'+a'b+bc$

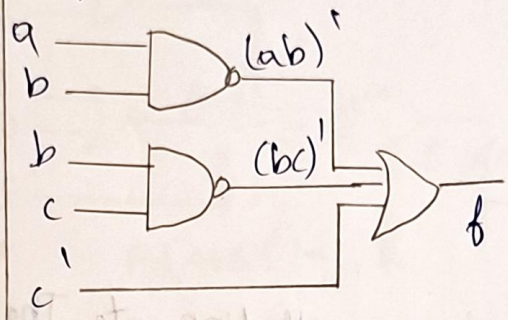
Step4: Check for the bubbles if a bubble is compensated by another bubble in a same line keep the inputs as such. If a bubble is not compensated by another bubble in a same line complement the input

1) $f = ab + bc + c'$

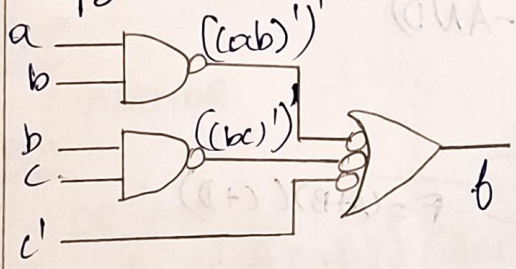
Step1:



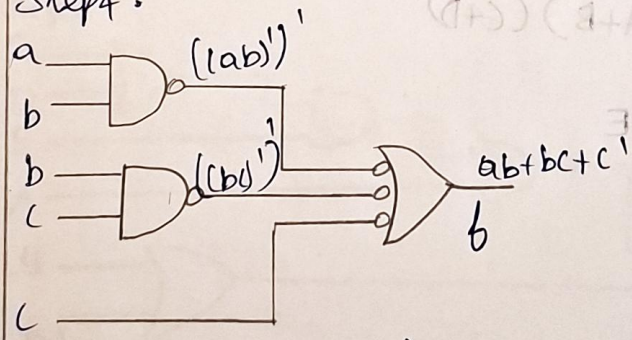
Step2:



Step3:



Step4:

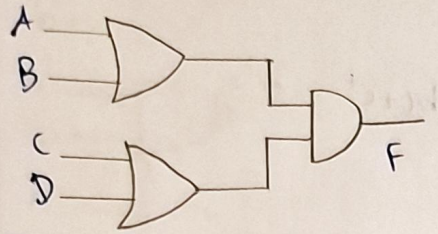


$\therefore ab + bc + c'$

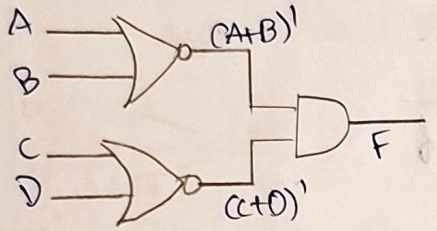
NOR Implementation:

1) $F = (A+B)(C+D)$

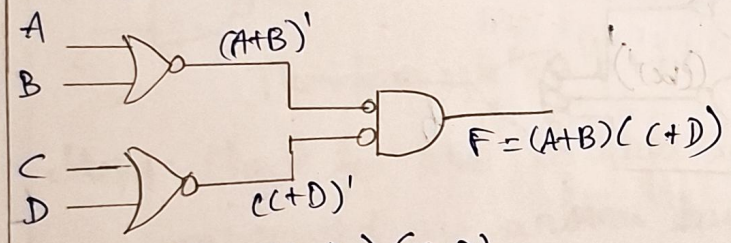
Step 1:



Step 2: Replace OR gate with NOR gate. The symbol is (OR-INV)



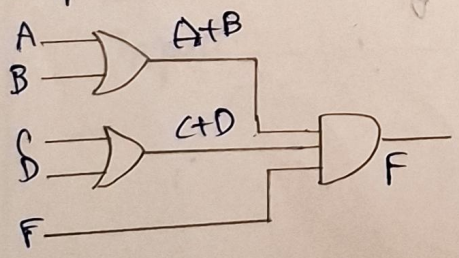
Step 3: Replace AND gate with NOR gate. The symbol is (INV-AND)



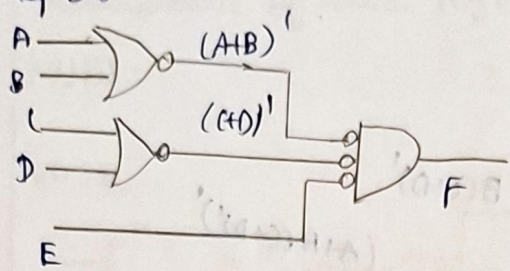
$\therefore F = (A+B)(C+D)$

2) $F = (A+B)(C+D)E$

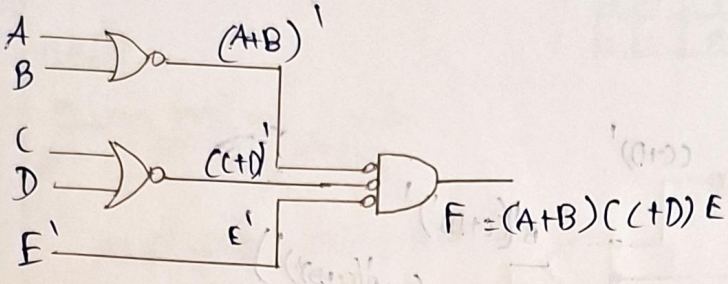
Step 1:



Step 3:



Step 4:



$F = (A+B)(C+D)E$

Two level Implementation

<u>SOP</u>

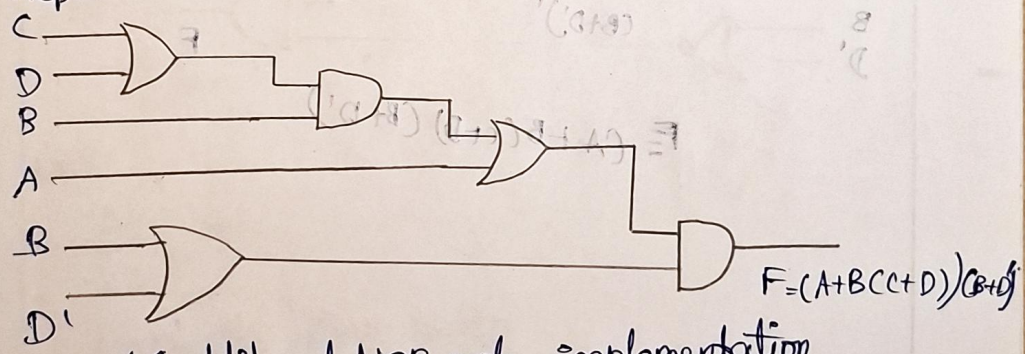
AND-OR

<u>POS</u>

OR-AND

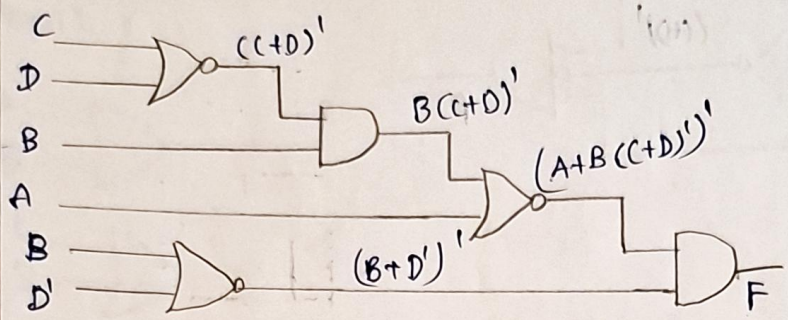
$F = (A+B(C+D))(B+D')$

Step 1:

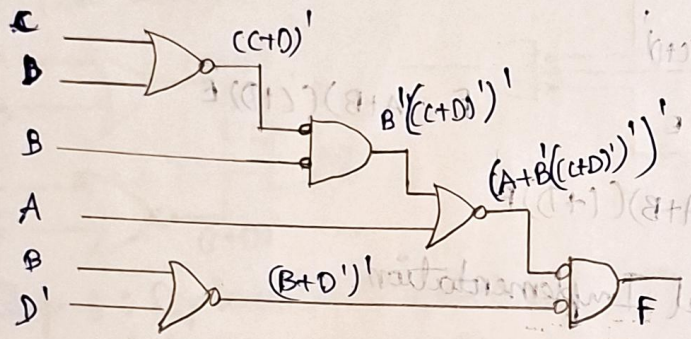


Multilevel NOR gate implementation

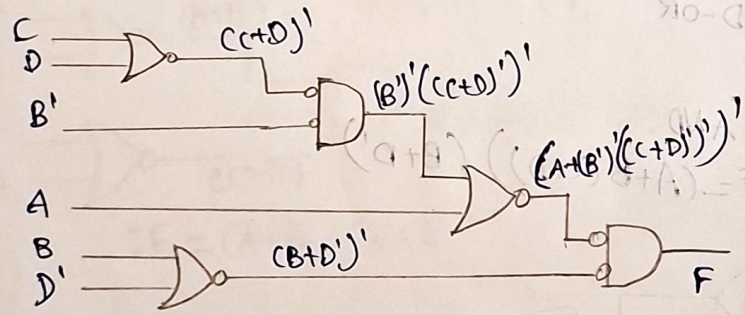
Step 2:



Step 3:



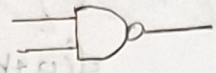
Step 4:



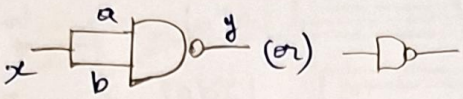
$$F = (A+B(C+D)(B+D')$$

Realization of Basic logic gate using NAND:

NAND :



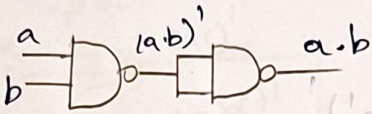
NOT:



Not gate using NAND

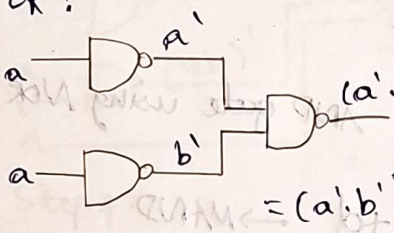
x	a	b	O/P
0	0	0	1 ✓
	0	1	1
	1	0	1
1	1	1	0 ✓

AND:



AND gate using NAND

OR :



$= (a'.b')'$

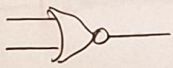
$= ((a+b)')'$

$F = a+b$

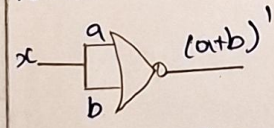
OR gate using NAND

Realization of basic logic gate using NOR

NOR:

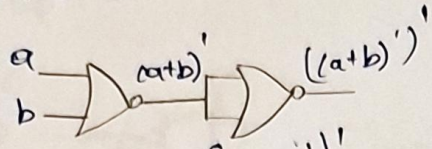


NOT:



NOT gate using NAND

~~AND~~ OR:



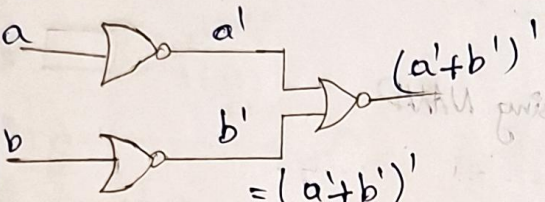
$= ((a+b)')'$

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

$F = a+b$

OR ~~AND~~ gate using NOR

~~OR~~ AND:



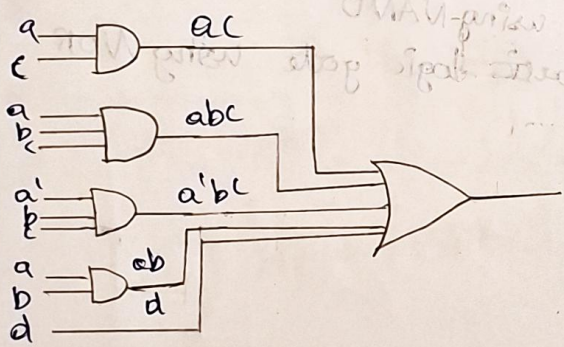
$= (a'+b')'$

$\cancel{= ((a+b)')'}$

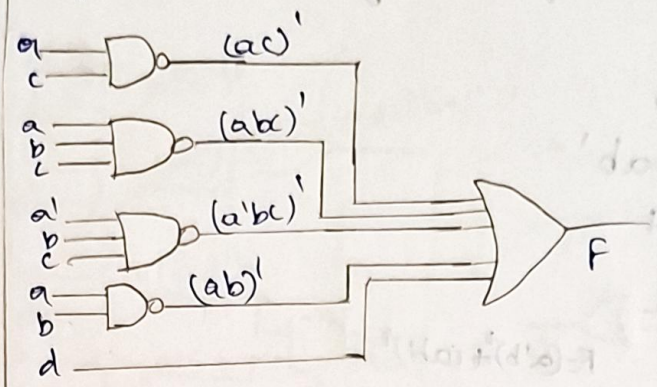
$= (a')'(b')'$ AND gate using NOR

$F = a.b$

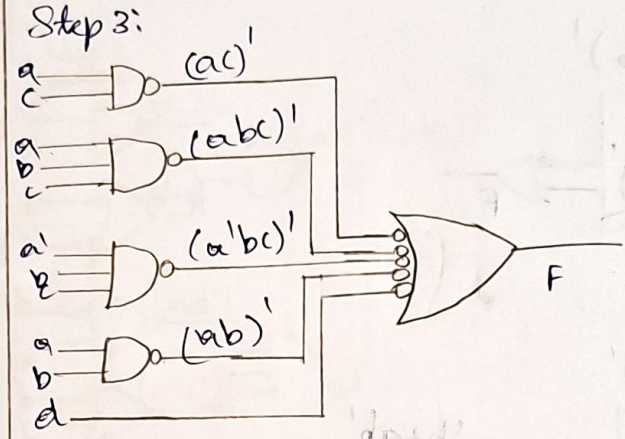
$f = ac + abc + a'bc + ab + d \rightarrow$ NAND



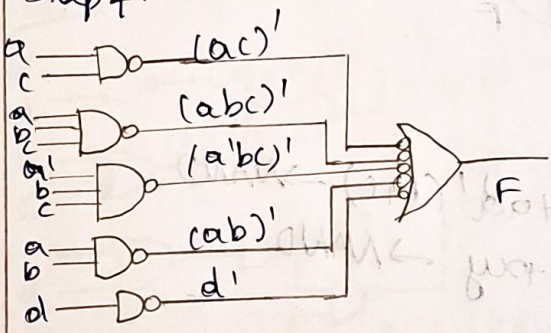
Step 1:



Step 3:



Step 4:

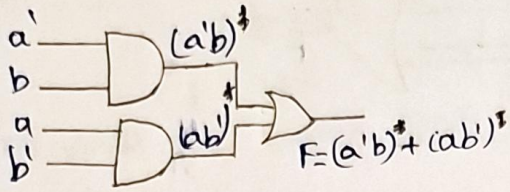


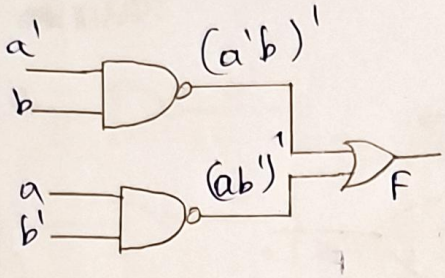
Implement XOR gate using ~~NAN~~ NAND gate.

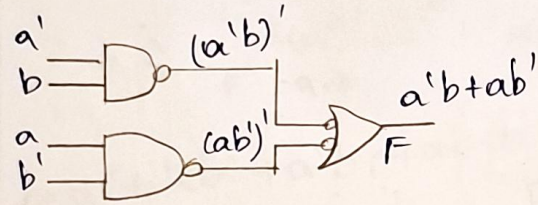
XOR

$f = a \oplus b$

$= a'b + ab'$





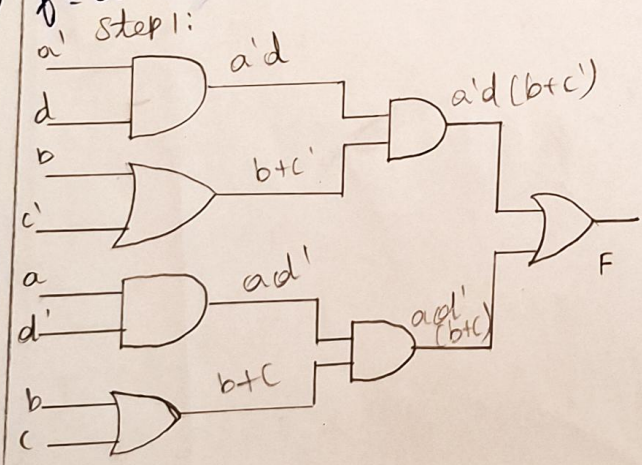


1) $f = a'd(b+c') + ad'(b+c)$ → NAND

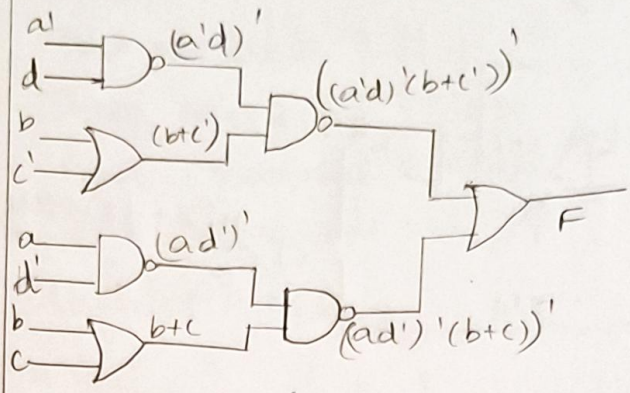
2) $f = w(x+yz) + xy$ → NAND

1) $f = a'd(b+c') + ad'(b+c)$

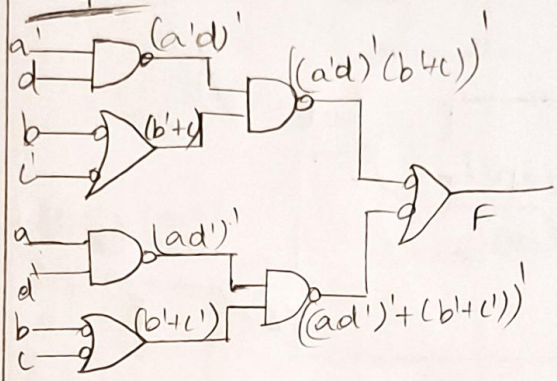
Step 1:



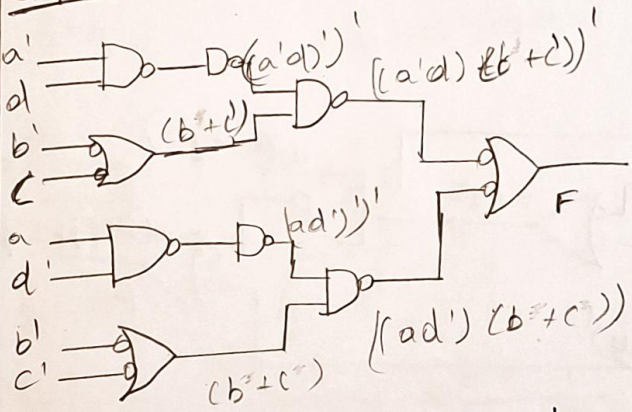
Step 2



Step 3



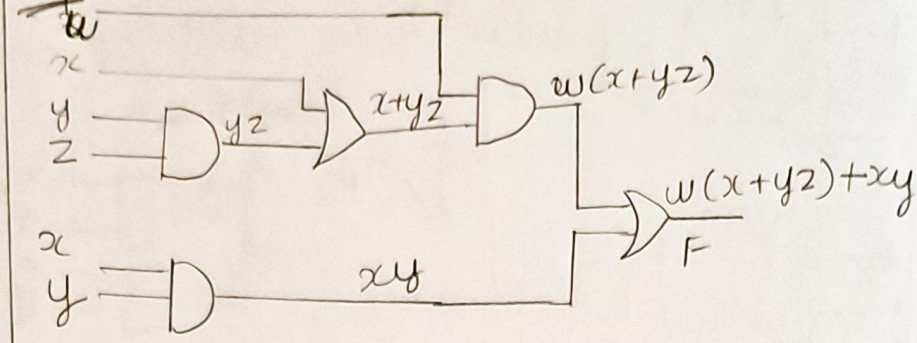
Step 4



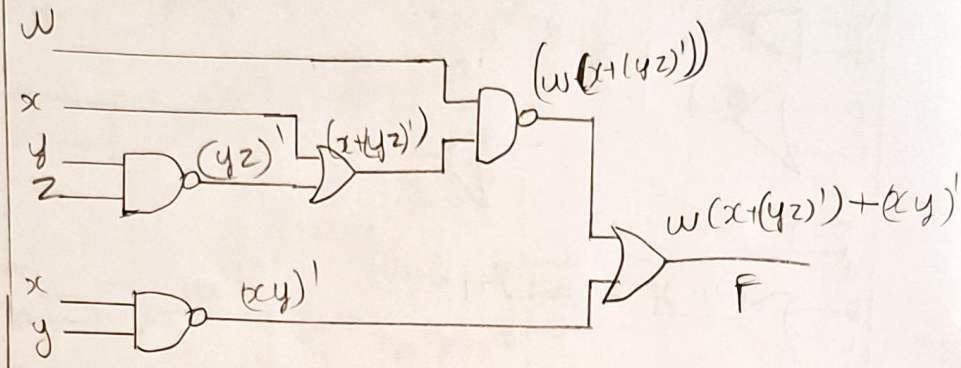
$$F = a'd(b+c') + ad'(b+c)$$

2) $f = w(x+yz)+xy$

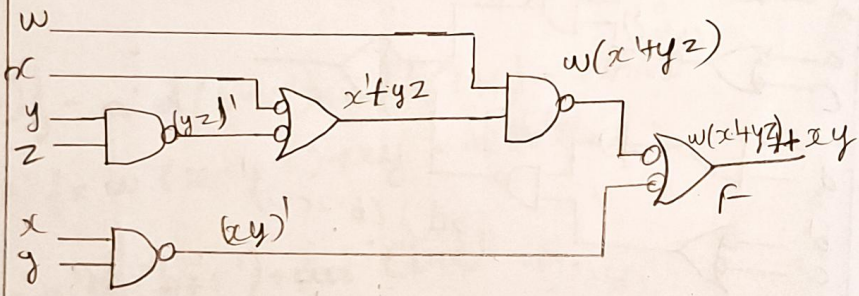
Step 1 :



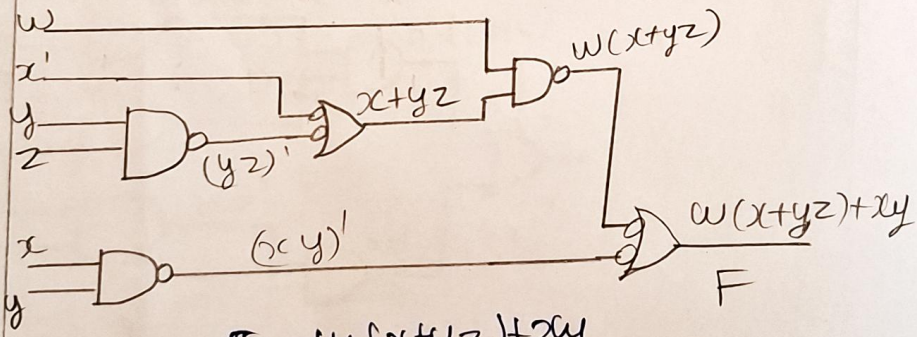
Step 2 :



Step 3 :



Step 4 :

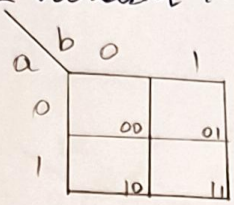


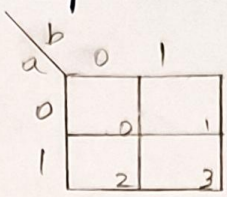
$F = w(x+yz)+xy$

Karnaugh-Map (K-Map)

Two variable KMap
Three variable KMap
Four variable K-Map
Five variable K-Map

2-Variable K-Map



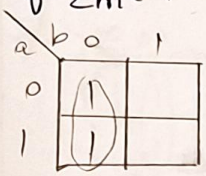


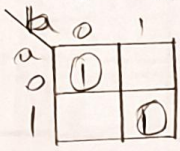
8 4 2 1

a	b
0	0
0	1
1	0
1	1

$f = \Sigma m(0,2)$

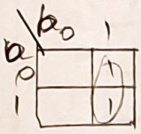
$f = \Sigma m(0,3)$

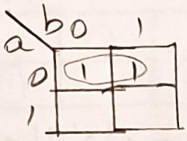




$f = b'$

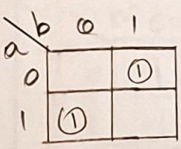
$f = a'b' + ab$

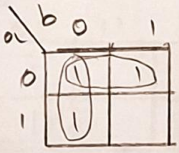


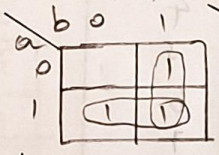


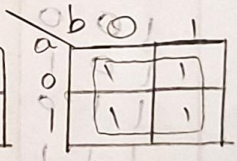
$f = b$

$f = a'$









$f = a'b + ab'$ $\quad$ $f = b' + a'$ $\quad$ $f = a + b$ $\quad$ $f = 1$

$f = \Sigma m(1,2)$ $\quad$ $f = \Sigma m(0,1,2)$ $\quad$ $f = \Sigma m(1,2,3)$ $\quad$ $f = \Sigma m(0,1,2,3)$

Write truth table for AND gate and derive the boolean expression using KMap.

A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

$f = \Sigma m(3)$

a \ b	0	1
0		
1		1

$f = (ab)$

Three variable KMap:

a \ bc	00	01	11	10
0	0	1	3	2
1	4	5	7	6

a	b	c	o/p
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

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

a \ bc	00	01	11	10
0	1 (0)	1 (1)	(3)	(2)
1	1 (4)	1 (5)	(7)	(6)

$f = b'$

1)

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

$f = b' + c'$

2)

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

$f = b'c' + bc$

3)

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

$f = a'b'c' + ab'c + a'bc + abc'$

4)

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

$f = b' + c'$

5)

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

$f = ac + bc + ab$

6)

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

$f = a'b' + ab$

7)

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

$f = \cancel{a'b'c'} + bc + a'c'$

8)

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

$f = a'c + ac'$

9)

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

$f = a'c' + ab'$

$f = \Sigma m(0,3,5,7)$

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

$f = ac + bc + a'b'c'$

4-Variable K-Map

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

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

$f = c'd' + cd$

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

$f = d$

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

$f = a'b'c' + a'bc + abc' + ab'c$

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

$f = d'$

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

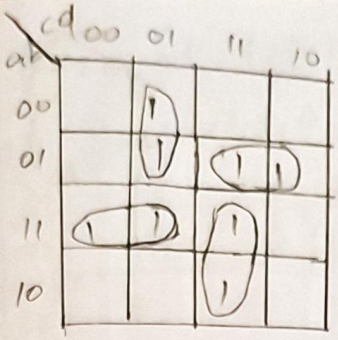
$f = cd + bd + b'c'd'$

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

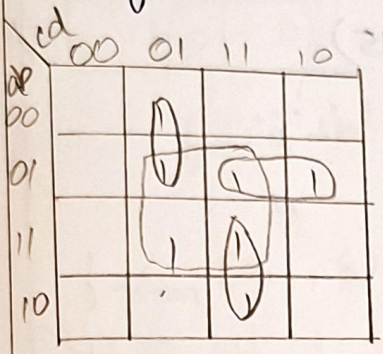
$f = a'b + a'c' + a'd$

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

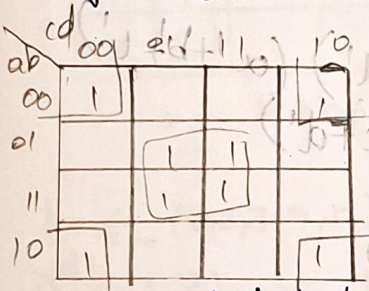
$f = c' + d'$



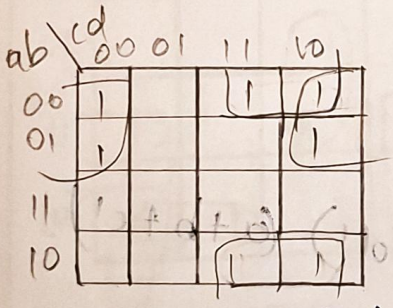
$f = a'c'd + abc' + a'bc + acd$



$f = bd + a'c'd + a'bc + acd$



$f = b'd' + bd$



$f = \cancel{ab'c + a'b'c +}$
$a'd' + b'c$

POS

$f = \Pi_M (0,2,5)$

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

$f = (a+c) \cdot (a'+b+c')$

$f = \Pi_M (0,1,5,7,9,12,14,15)$

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

$f = (a+b+c)(a+b'+d')(a'+b'+c')$
$(a'+b'+d)(b+c+d')$

$f = \Pi_M (0,2,3,8,9,12,13,15)$

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

$f = (a'+c+d')(a'+b'+d')(a+b+c')$
$(b+c+d)$

$f = \Sigma m\,(0,4,5,7,8)$

$f = \Pi M\,(1,2,3,6,9,10,11,12,13,14,15)$

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

$f = (a'+b')(b+d')(c'+d)$

$f = \Sigma m\,(0,1,5) + \Sigma d\,(3,4)$

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

K-Map with Don't care

Don't care (X)

a	b	f
0	0	0
0	1	1
1	0	1
1	1	X → 1/0

$f = \Sigma m\,(0,2,7,9,12) + \Sigma d\,(8,10,14,15)$

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

$f = \cancel{(bcd)} + \cancel{ab'c'} + \cancel{a'b'd'} + \cancel{abd'}$

$f = bcd + ad' + ab'c' + b'd'$

1) $f = \Sigma_m(0,7,8,9,10,12) + \Sigma_d(2,5,13)$

2) $f = \Sigma_m(1,3,4,6,11) + \Sigma_d(8,10,12,13)$

3) $f = \Sigma_m(0,2,4,5,6,8,10,15) + \Sigma_d(7,13,14)$

4) $f = \Sigma_m(0,3,4,7,8,10,12,14) + \Sigma_d(2,6)$

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

$$f = a'bd + ac' + b'd'$$

2) sol^n

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

$$f = a'b'd + a'bd' + ab'c$$

3)

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

$$\cancel{f = bd + b'd' + a'bd'}$$

$$f = b'd' + \cancel{bd'} + a'b + abd$$

4)

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

$f = (d)(a+c')$

5-Variable Map (a, b, c, d, e) 0 – 31

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

a = 0

bc \ de	00	01	11	10
00	16	17	19	18
01	20	21	23	22
11	28	29	31	30
10	24	25	27	26

a = 1

i)

a = 0

bc \ de	00	01	11	10
00				
01		1	1	
11		1	1	
10				

a = 1

bc \ de	00	01	11	10
00				1
01		1	1	1
11		1	1	
10				1

	16	8	4	2	1
	a	b	c	d	e
0 →	0	0	0	0	0
8 →	0	1	0	0	0
15 →	0	1	1	1	1
16 →	1	0	0	0	0
24 →	1	1	0	0	0
31 →	1	1	1	1	1

$f = (e + ade'$

$f = \Sigma m(0, 2, 4, 6, 9, 11, 13, 15, 17, 21, 25, 27, 29, 31)$

a = 0

bc \ de	00	01	11	10
00	1			1
01	1			1
11		1	1	
10		1	1	

a = 1

bc \ de	00	01	11	10
00		1		
01		1		
11		1	1	
10		1	1	

$f = be + ab'e' + ad'e$

$f = \Sigma_m(1,3,6,10,11,12,14,15,17,19,20,22,24,29,30)$

a=0

bc \ de	00	01	11	10
00		1	1	
01				1
11	1		1	1
10			1	1

a=1

bc \ de	00	01	11	10
00		1	1	
01	1			1
11		1		1
10	1			

$f = b'c'e + a'bd + a'bce' + ab'ce' + abcd'e + abc'd'e'$

$f = \Sigma_m(0,1,4,5,16,17,21,25,29)$

a=0

bc \ de	00	01	11	10
00	1	1		
01	1	1		
11				
10				

a=1

bc \ de	00	01	11	01
00	1	1		
01		1		
11		1		
10		1		

$f = a'b'd' + ad'e + b'c'd'$

$f = \Sigma_m(0,2,4,6,9,13,21,23,25,29,31)$

a=0

bc \ de	00	01	11	10
00	1			1
01	1			1
11		1		
10		1		

a=1

bc \ de	00	01	11	10
00				
01		1	1	
11		1	1	
10		1		

$f = a'b'e' + bd'e + ace + \cancel{abd'e}$

$f = \Sigma m\,(0,1,2,4,5,6,8,9,12,13,14)$

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

$f = c' + a'd' + b\,d'$

$f = a'b'c' + b'cd' + a'bcd' + ab'c'$

$= a'b'c'(d+d') + b'cd'(a+a') + a'bcd' + ab'c'(d+d')$

$f = a'b'c'd + a'b'c'd' + ab'cd' + a'b'cd' + a'bcd' + ab'c'd + ab'c'd'$

$= a'b'c'd + a'b'c'd' + ab'cd' + a'b'cd' + a'bcd' + ab'c'd + ab'c'd'$

$f = \Sigma m\,(0,1,10,2,6,9,8)$

$f = \Pi M\,(3,4,5,7,11,12,13,14,15)$

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

$f = b'c' + b'd' + a'cd'$

1) $f = \Sigma_m (0,1,2,5,8,9,10)$

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

$f = b'c' + b'd' + a'c'd$

$f = \Pi_M (3,4,6,7,11,12,13,14,15)$

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

$f = (c+d')(a+b')(b+c+d)$

---

1) $f(A,B,C) = \Sigma_m (0,2,4) + \Sigma_d (1,5)$

2) $f(A,B,C) = \Sigma_m (0,1,3,7) + \Sigma_d (2,5)$

3) $f(A,B,C,D) = \Sigma_m (1,3,7,11,15) + \Sigma_d (0,2,4)$

4) $f(A,B,C,D) = \Sigma_m (5,6,7,12,13) + \Sigma_d (4,9,14,15)$

5) $f(A,B,C,D) = \Sigma_m (0,7,8,9,10,12) + \Sigma_d (2,5,13)$

Answers:

1)

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

$f = a'c' + b'$

2)

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

$f = a' + c$

3)

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

$f = a'b' + cd$

4)

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

~~$f = a'b + ab$~~ $f = b$

5)

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

$f = ac' + b'd' + a'bd$

1)

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

$a \oplus b$ XOR $= a'b + ab'$

$(a \oplus b)'$ XNOR $= a'b' + ab$

$f = a'b'c' + ab'c + a'bc + abc'$

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

$= a'(b \oplus c)' + a(b \oplus c)$

$b \oplus c = x$

$= a'x' + ax$

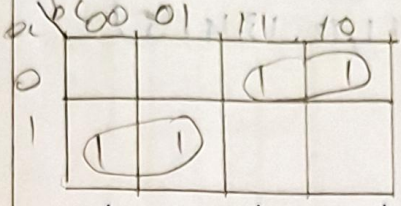
$= (a \oplus x)'$

$f = (a \oplus b \oplus c)'$

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

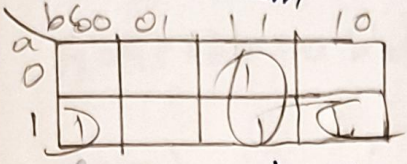
$f =$

1) $F(A,B,C) = \sum m(2,3,4,5)$



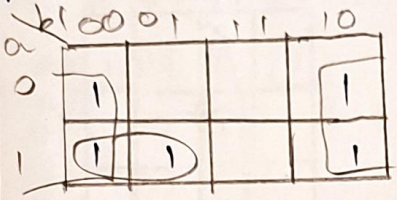
$f = a'b + ab'$

2) $F(A,B,C) = \sum m(3,4,6,7)$



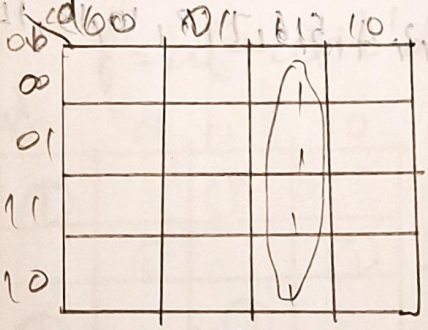
$f = ac' + bc$

3) $f(A,B,C) = \sum m(0,2,4,5,6)$



$f = c' + ab'$

4) $F(A,B,C,D) = \sum m(3,7,11,15)$



$f = cd$

5) $F(A,B,C,D) = \Sigma m(9,10,11,12,13,14,15)$

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

$f = ab + bd + ac$

6) $F(A,B,C,D) = \Sigma m(0,2,4,6,8,10,12,14)$

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

$f = d'$

7) $F(A,B,C,D) = \Sigma m(0,2,4,5,6,7,8,10,13,14)$

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

$f = a'b + cd' + b'd' + b'd$

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

$= x'y' + xy$

$\therefore f = (a \oplus b \oplus c \oplus d)'$

8) $F(A,B,C,D) = \Sigma_m(1,3,4,5,10,11,12,13,14,15)$

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

$f = ac + bc' + a'b'd$

9) $F(A,B,C,D) = \Sigma_m(0,2,3,5,7,8,10,11,13,15)$

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

$f = bd + cd + b'd'$

10)

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

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

$f = a'b'c'd' + a'b'cd + a'bc'd + a'bcd' + ab'c'd' + ab'cd' + abc'd' + abcd + ab'c'd + abcd'$

$= a'b'(c'd'+cd) + a'b(c'd+cd') + ab(c'd'+cd) + ab'(c'd+cd')$

$= a'b'(c \oplus d)' + a'b(c \oplus d) + ab(c \oplus d)' + ab'(c \oplus d)$

$x = (c \oplus d)$

$f = a'b'x' + a'bx + abx' + ab'x$

Combinational Logic

Definition:

In the case of combinational logical circuits the output is depends only on the present input.

~~ADDERS:~~

Step 1: Write the truth table
Step 2: Put the K-Map and get the corresponding boolean equation.
Step 3: Draw the logical circuit.

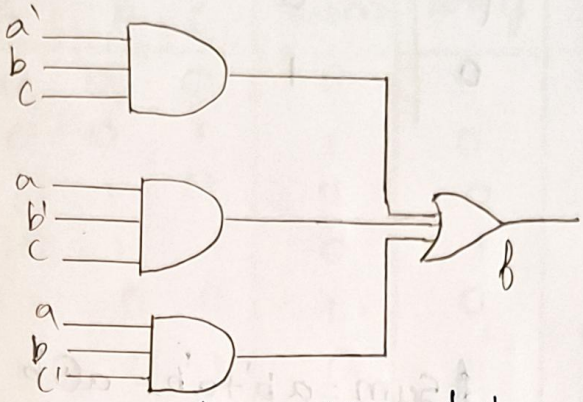
Design a combinational logic circuits with three inputs x, y, z and one output f. The output is equal to one (1) if the number of ones with binary combination of your inputs x, y, z are even.

XYZ	F
0 0 0	0
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	0

$$f = \Sigma m\,(3, 5, 6)$$

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

$f = a'bc + ab'c + abc'$



$f = a'bc + ab'c + abc'$

ADDERS:

1) Half adder (2 bits)
2) Full adder (3 bits)

Output:
Sum, carry

Bin addition:

	S	C
0+0	0	0
0+1	1	0
1+0	1	0
1+1	0	1

1+1=2 (0 1)
1+1+1=3 (1 1)

8	4	2	1
		1	0
		1	1

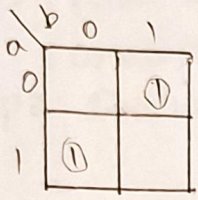
Half adder:

Truth Table:

a	b	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

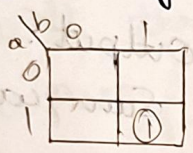
KMap

(i) Sum

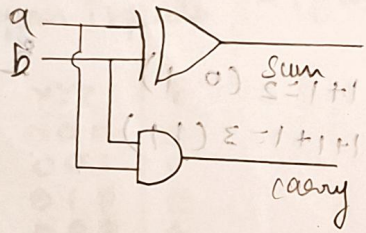


Sum $= ab' + a'b = a \oplus b$

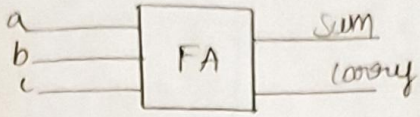
(ii) Carry



Carry $= a \cdot b$



Full adder:



Truth Table

a	b	c	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

K Map:

(i) Sum

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

$$Sum = a'b'c + a'bc' + ab'c' + abc$$

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

$x = a \oplus b$

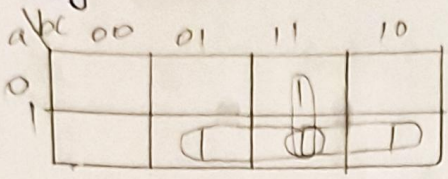
$$= c(a \oplus b)' + c'(a \oplus b)$$

$$= cx' + c'x$$

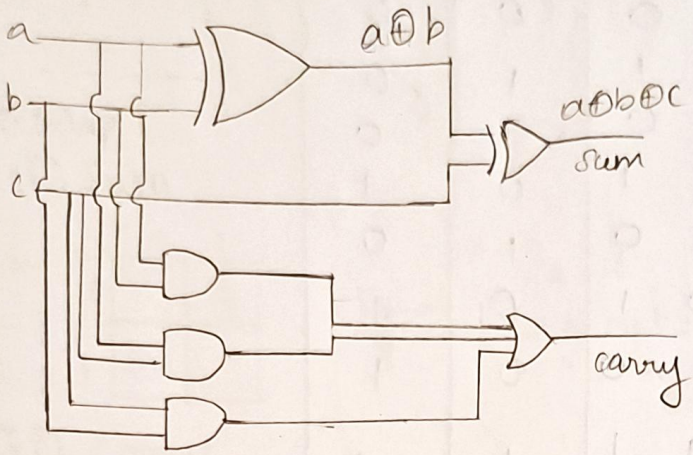
$$= c \oplus x$$

$$Sum = a \oplus b \oplus c$$

(ii) Carry :

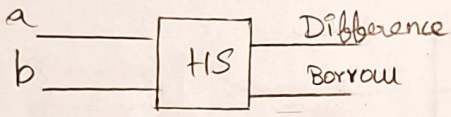


$$Carry = ac + bc + ab$$



Subracters:

Half Subracter:



a b	D	B
0 − 0 =	0	0
0 − 1 =	1	1
1 − 0 =	1	0
1 − 1 =	0	0

B
1 0 (2)
0 1 (1)
1

Truth Table

a	b	D	B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

K Map (Difference):

a \ b	0	1
0		1
1	1	

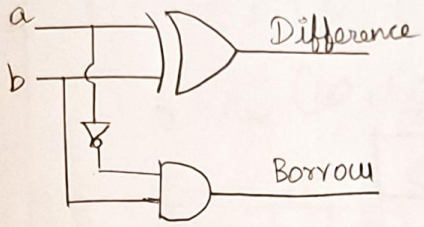
Difference $= a'b + ab'$

$D = a \oplus b$

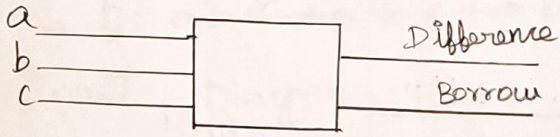
K Map (Borrow):

a \ b	0	1
0		1
1		

Borrow $= a'b$



Full Subracter:



Truth Table:

a	b	c	D	B
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

KMap:

i) Difference:

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

$$\text{Difference} = a'b'c + a'bc' + ab'c' + abc$$

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

$$= c(a \oplus b)' + c(a \oplus b)$$

$x = a \oplus b$

$$= cx' + cx$$

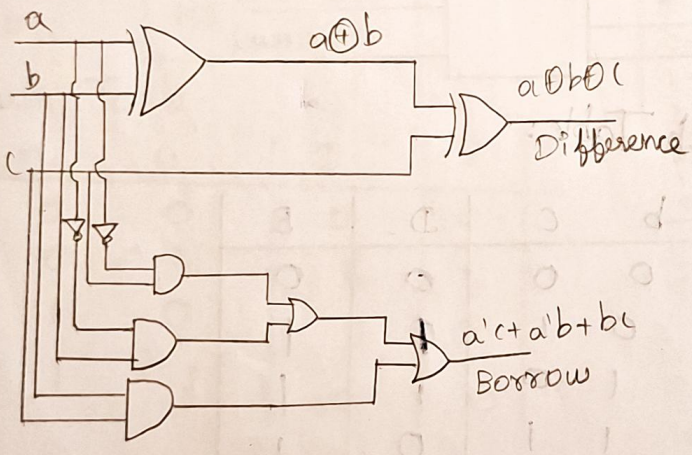
$$= c \oplus x$$

$$= a \oplus b \oplus c$$

ii) Borrow:

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

$$\text{Borrow} = a'c + a'b + bc$$



Design a combinational circuit with three inputs and one output in which the output is 1 when the binary value of the input is less than 3. The output is 0, otherwise

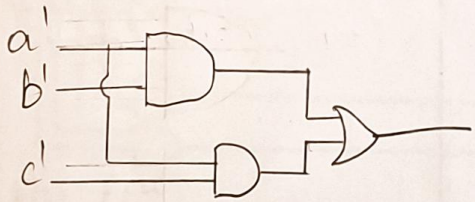
a	b	c	f
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

$f = \Sigma_m (0,1,2)$

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

$f = \cancel{a'b'c' + a'b'c + a'bc'}$ $a'b' + a'c'$

Circuit Diagram:



2) ABCD (Binary coded Decimal) number with 4 bits (w, x, y, z) appear at the input of a circuit. The output of the circuits is 1 if the input values are 0, 2, 3, 5 or 8. Obtain sum of products and product of sums expression of the circuit. Implement using NAND gate only.

Decimal	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

B A 2 1

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

$f = \Sigma_m(0,2,3,5,8)$

$f = \Pi_M(1,4,6,7,9)$

1) For SOP:

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

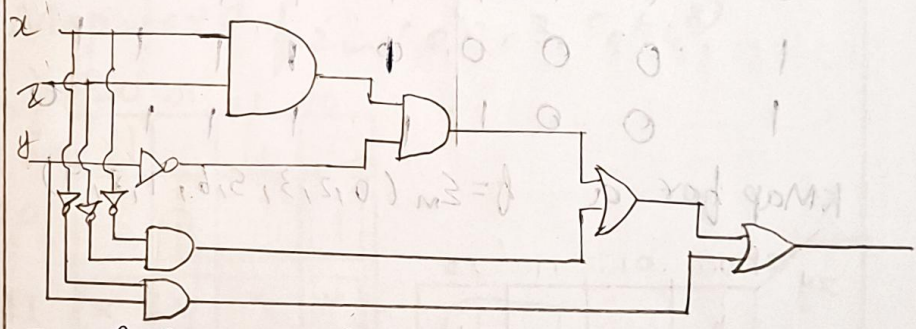
$$f = x'z' + xy'z + x'y$$

2) For POS:

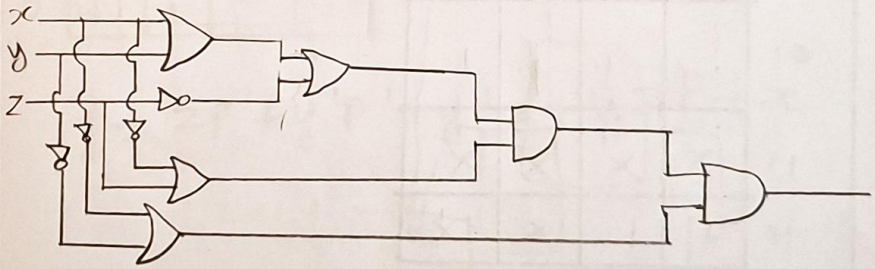
wx \ yz	00	01	11	10
00		0		
01	0		0	0
11	x	x	x	x
10		0	x	x

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

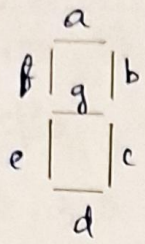
For SOP:



For POS:



Seven Segment Display:



x	y	z	p	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1

KMap for a $f = \Sigma_m(0,2,3,5,6,7,8,9)$

xy \ zp	00	01	11	10
00	1		1	1
01		1	1	1
11	X	X	X	X
10	1	1	X	X

$f = z + y'p' + x + yp$

KMap for b: $f = \Sigma m(0,1,2,3,4,7,8,9)$

xy \ zp	00	01	11	10
00	1	1	1	1
01	1		1	
11	X	X	X	X
10	1	1	X	X

$f = z + z'p' + zp + y'$

KMap for c: $f = \Sigma m(0,1,3,4,5,6,7,8,9)$

xy \ zp	00	01	11	10
00	1	1	1	
01	1	1	1	1
11	X	X	X	X
10	1	1	X	X

$f = z' + p' + y$

KMap for d $f = \Sigma m(0,2,3,5,6,8)$

xy \ zp	00	01	11	10
00	1		1	1
01		1		1
11	X	X	X	X
10	1	1	X	X

$f = zp' + y'p' + y'z + yz'p + x$

K Map for e $f = \Sigma m(0,2,6,8)$

xy \ zp	00	01	11	10
00	1			1
01				1
11	x	x	x	x
10	1		x	x

$$f = zp' + y'p'$$

KMap for f : $f = \Sigma m(0,4,5,6,8,9)$

xy \ zp	00	01	11	10
00				
01	1	1		1
11	x	x	x	x
10	1	1	x	x

$$f = x + yz' + yp' + z'p'$$

KMap for g $f = \Sigma m(2,3,4,5,6,8,9)$

xy \ zp	00	01	11	10
00			1	1
01	1	1		1
11	x	x	x	x
10	1	1	x	x

$$f = zp' + yz' + x + y'z$$

Design FA using 2 HA:

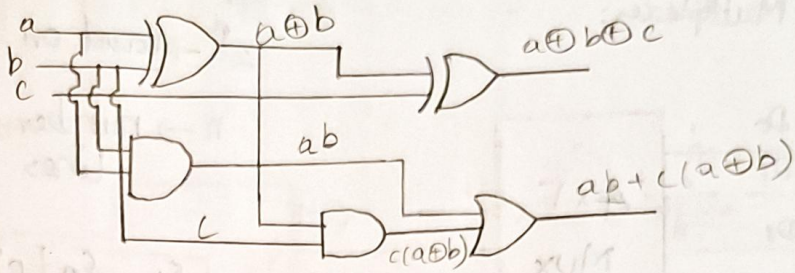
Sum $= a \oplus b \oplus c$

carry $= ab + a'bc + ab'c$

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

$= ab + c(a \oplus b)$

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



Multiplexers and Demultiplexers

Multiplexers:

Many to one, Data Selector

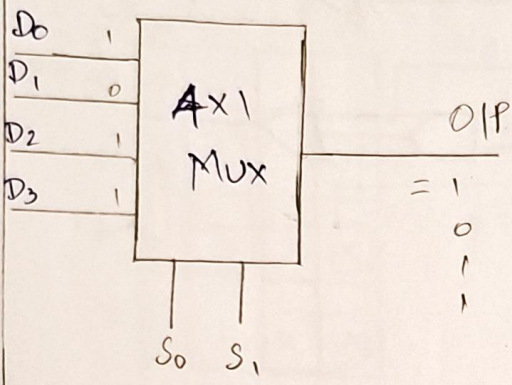
Demultiplexers:

One to many, Data Distributor

Multiplexer:

$2^n \rightarrow$ Based on inputs

$n \rightarrow$ number of selection lines



S_1	S_0	O/P
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

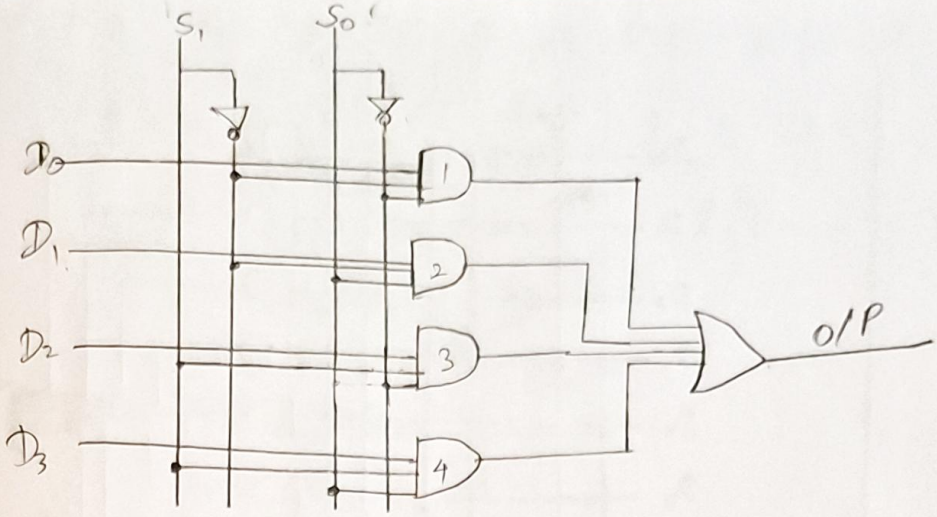
O/P

$D_0 \rightarrow S_1' \; S_0'$

$D_1 \rightarrow S_1' \; S_0$

$D_2 \rightarrow S_1 \; S_0'$

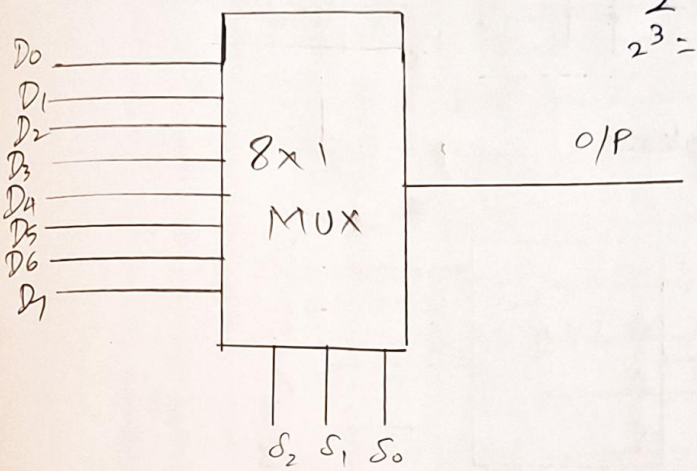
$D_3 \rightarrow S_1 \; S_0$



8×1 MUX

2^n $n=3$

$2^3 = 8$ inputs



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

$D_0 \rightarrow S_2' S_1' S_0'$

$D_1 \rightarrow S_2' S_1' S_0$

$D_2 \rightarrow S_2' S_1 S_0'$

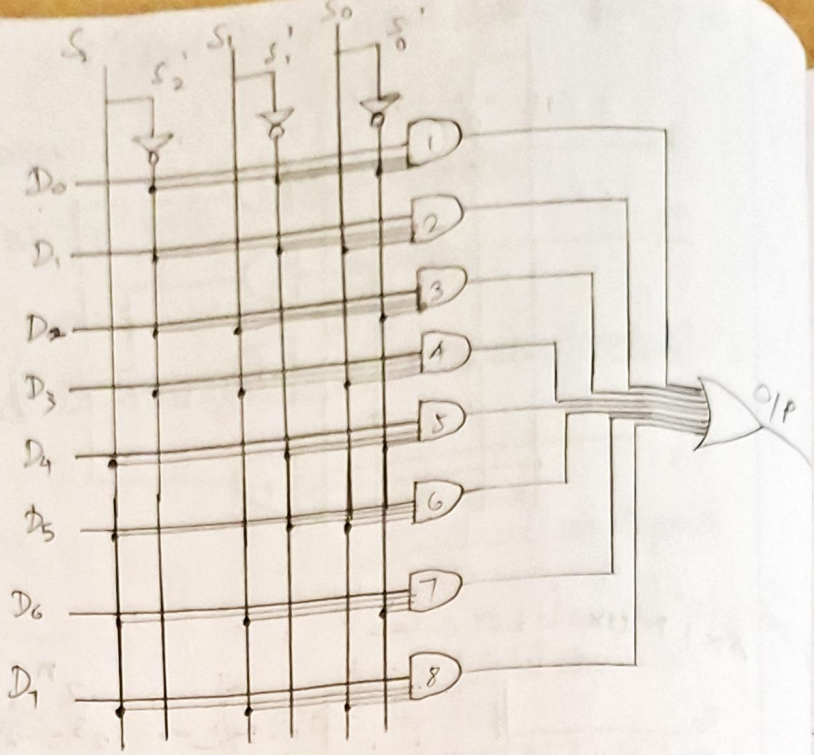
$D_3 \rightarrow S_2' S_1 S_0$

$D_4 \rightarrow S_2 S_1' S_0'$

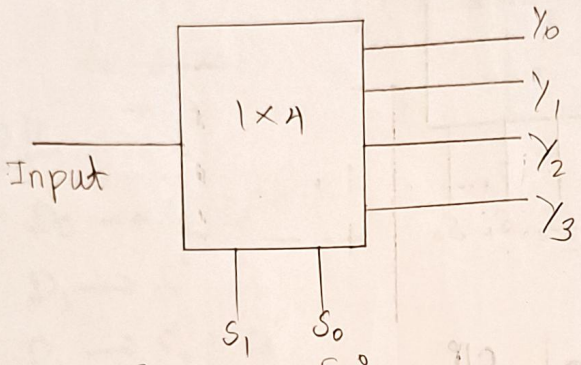
$D_5 \rightarrow S_2 S_1' S_0$

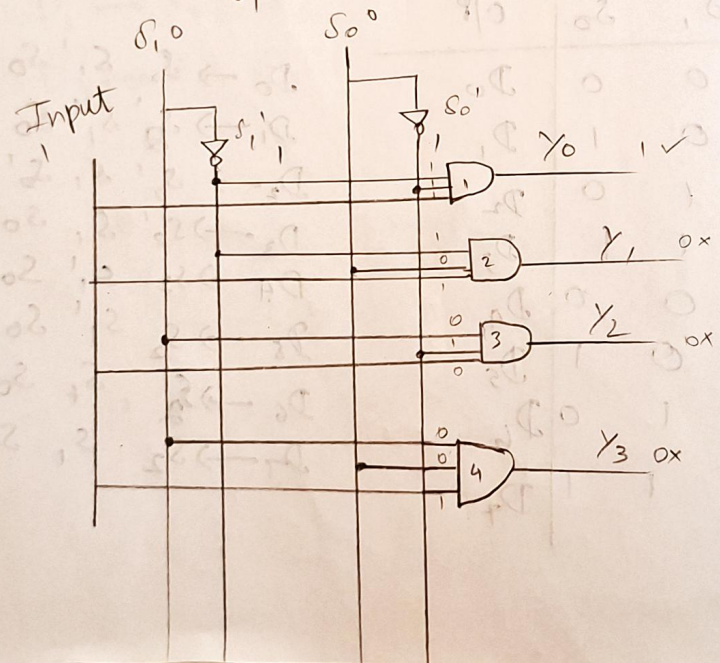
$D_6 \rightarrow S_2 S_1 S_0'$

$D_7 \rightarrow S_2 S_1 S_0$

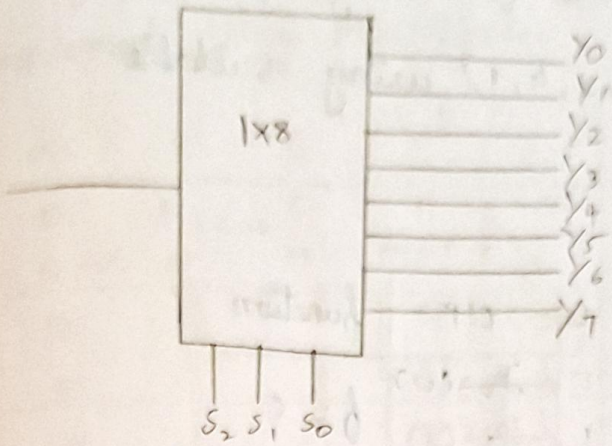


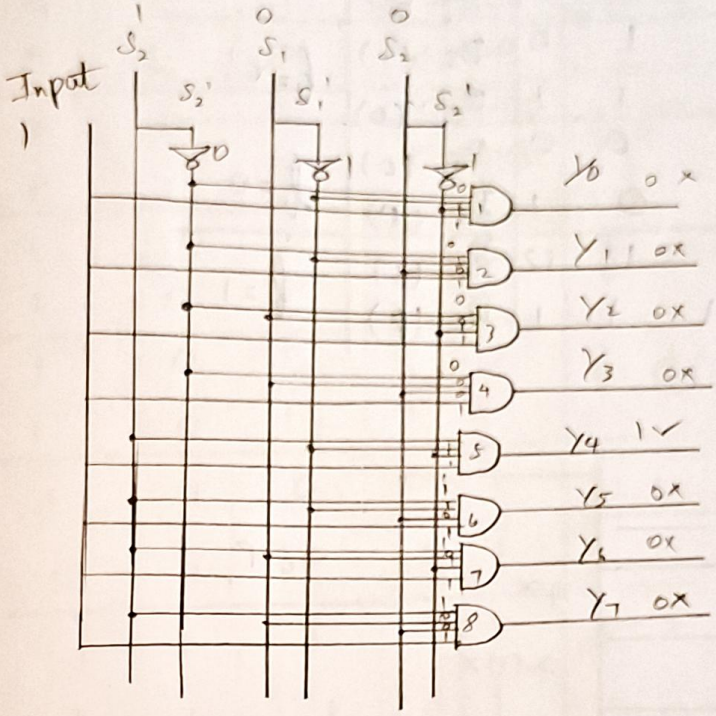
Demultiplexer:





1x8 DEMUX:

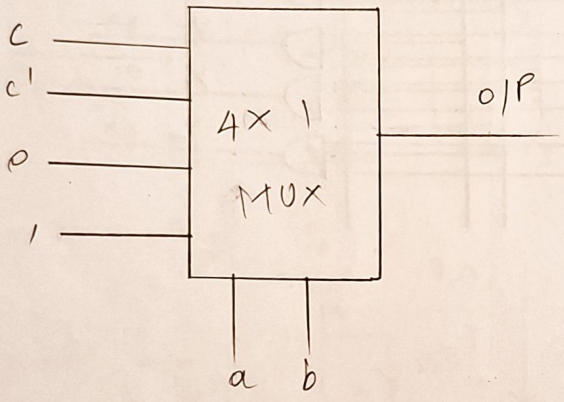

1x8
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7
S2 S1 S0


Input
1
S2
S1
S0
S2'
S1'
S0'
Y0 0 ×
Y1 0 ×
Y2 0 ×
Y3 0 ×
Y4 1 ✓
Y5 0 ×
Y6 0 ×
Y7 0 ×

Implement the following function:

$f = \Sigma_m (1, 2, 6, 7)$ using suitable multiplexer

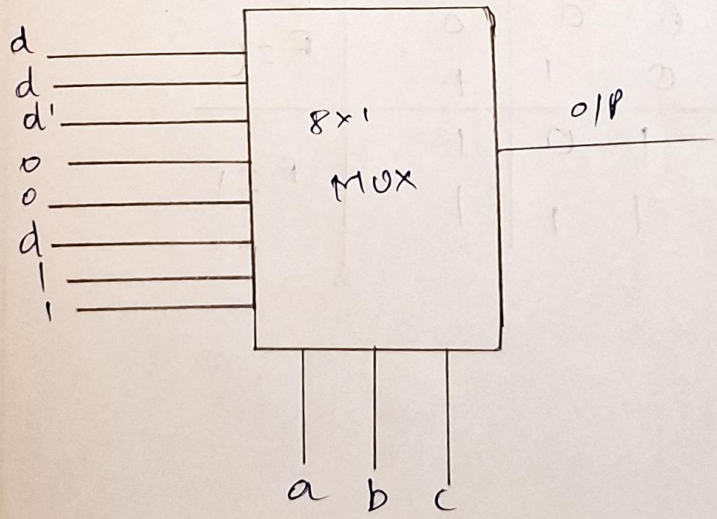
a	b	c	O/P	function
0	0	0	$\leftarrow D_0 \rightarrow (0)$	$f = c$
0	0	1	$\leftarrow D_1 \rightarrow (1)$	
0	1	0	$\leftarrow D_2 \rightarrow (1)$	$f = c'$
0	1	1	$\leftarrow D_3 \rightarrow (0)$	
1	0	0	$\leftarrow D_4 \rightarrow (0)$	$f = 0$
1	0	1	$\leftarrow D_5 \rightarrow (0)$	
1	1	0	$\leftarrow D_6 \rightarrow (1)$	$f = 1$
1	1	1	$\leftarrow D_7 \rightarrow (1)$	



$f = \Sigma_m (1,3,4,11,12,13,14,15)$

Implement Using Multiplexer.

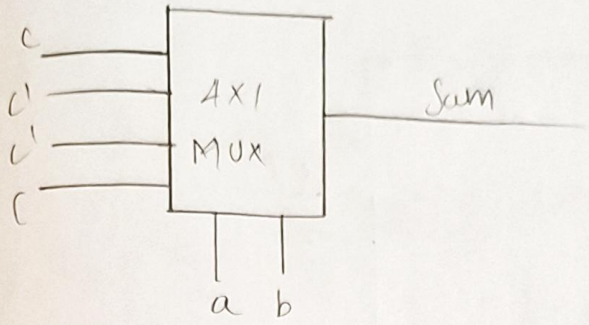
a	b	c	d	o/p	function
0	0	0	0	0	$f = d$
0	0	0	1	1	
0	0	1	0	0	$f = d$
0	0	1	1	1	
0	1	0	0	1	$f = d'$
0	1	0	1	0	
0	1	1	0	0	$f = 0$
0	1	1	1	0	
1	0	0	0	0	$f = 0$
1	0	0	1	0	
1	0	1	0	0	$f = d$
1	0	1	1	1	
1	1	0	0	1	$f = 1$
1	1	0	1	1	
1	1	1	0	1	$f = 1$
1	1	1	1	1	

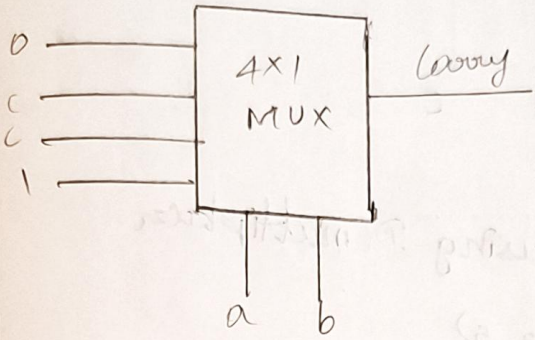


Implement full adder using appropriate multiplexer.

a	b	c	Sum	~~Carry~~ F
0	0	0	0	F=C
0	0	1	1	
0	1	0	1	F=C'
0	1	1	0	
1	0	0	1	F=C'
1	0	1	0	
1	1	0	0	F=C
1	1	1	1	

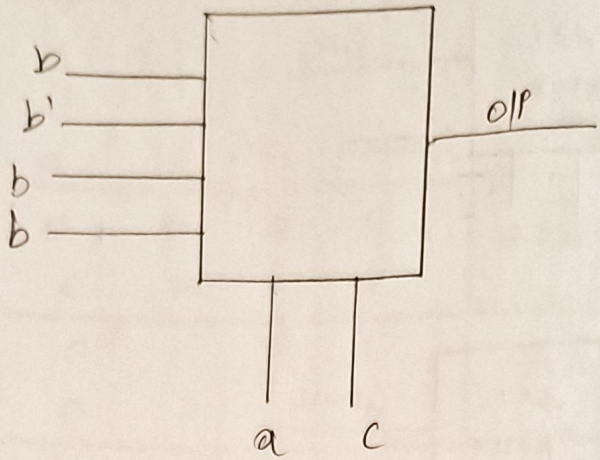
a	b	c	Carry	F
0	0	0	0	F=0
0	0	1	0	
0	1	0	0	F=C
0	1	1	1	
1	0	0	0	F=C
1	0	1	1	
1	1	0	1	F=1
1	1	1	1	





Selection line a & c

a	b	c	O/P	F
0	0	0	0	F=b
0	0	1	1	
0	1	0	1	F=b'
0	1	1	0	
1	0	0	0	F=b
1	0	1	0	
1	1	0	1	F=b
1	1	1	1	

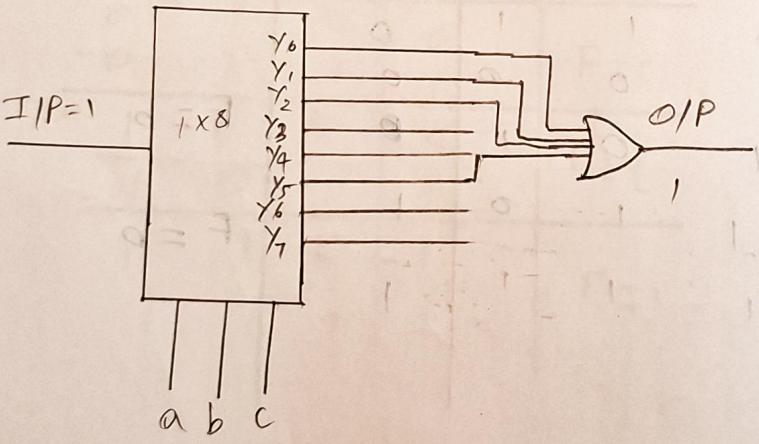


Implementation using Demultiplexer

$f = \Sigma_m (0, 1, 2, 5)$

a , b , c → selection line

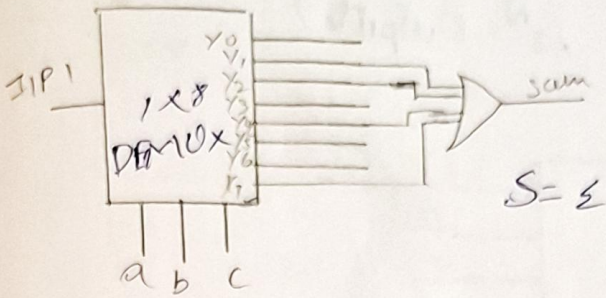
$$\left.\begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{matrix}\right\} = 1$$



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

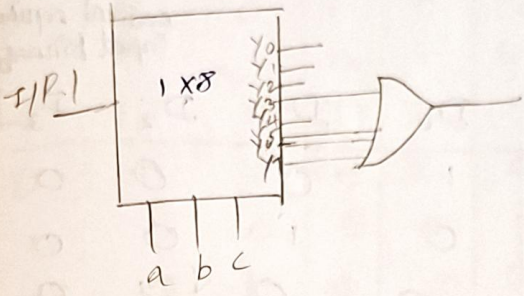
$Carry = \Sigma m(3,5,6,7)$

1.



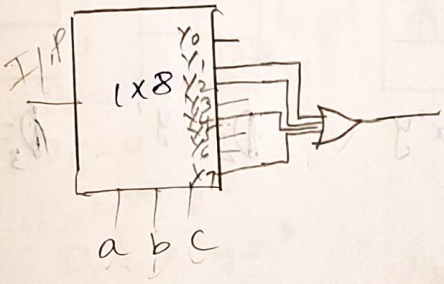
$S = \Sigma m(1,2,4,7)$

2.



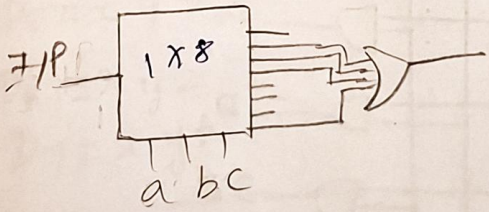
$C = \Sigma m(3,5,6,7)$

3.



$D = \Sigma m(1,2,4,7)$

4.

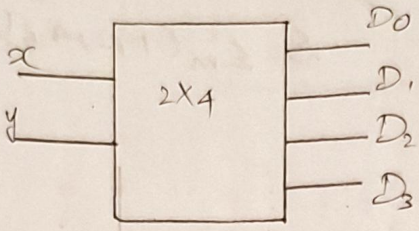


$B = \Sigma m(1,2,3,7)$

Endcoder and Decoder

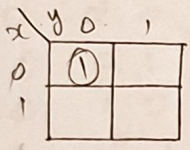
Decoder:

(N- Inputs , 2^N- Outputs)

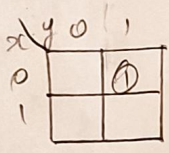


Decimal equivalent of input binary value

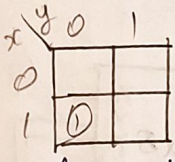
x	y	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1



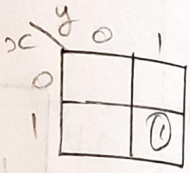
$D_0 = x'y'$



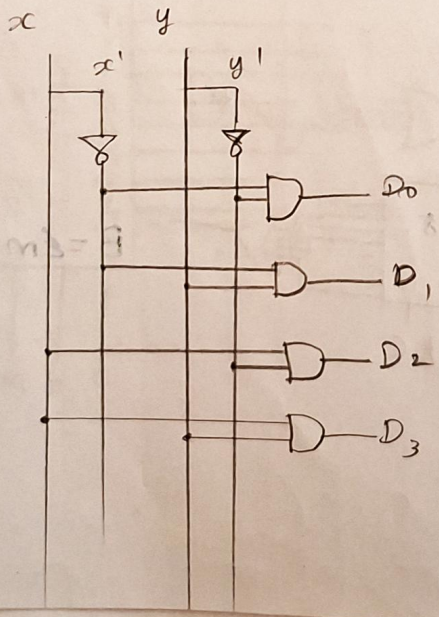
$D_1 = x'y$



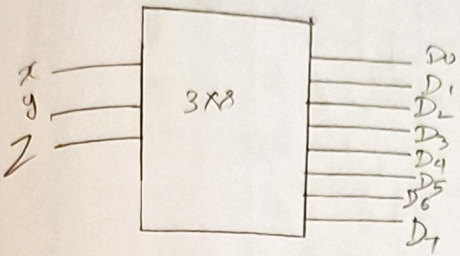
$D_2 = xy'$



$D_3 = xy$



3-Input



x	y	z	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

$D_0 = x'y'z'$

$D_1 = x'y'z$

$D_2 = x'yz'$

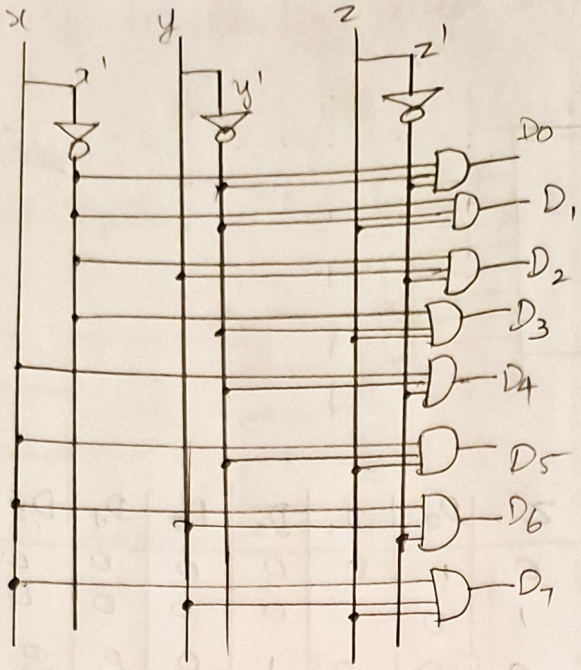
$D_3 = x'yz$

$D_4 = xy'z'$

$D_5 = xy'z$

$D_6 = xyz'$

$D_7 = xyz$

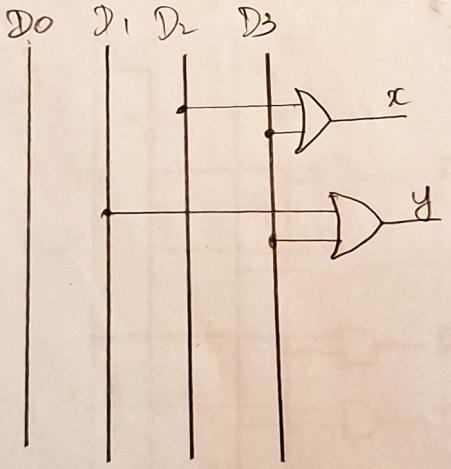


ENDCODERS: (2^n - inputs, n - outputs)

D_0	D_1	D_2	D_3	x	y
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

$x = D_2 + D_3$

$y = D_1 + D_3$



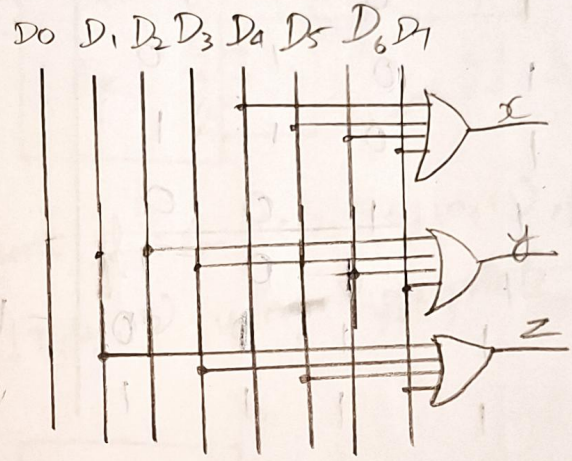
8 X 3

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

$x = D_4 + D_5 + D_6 + D_7$

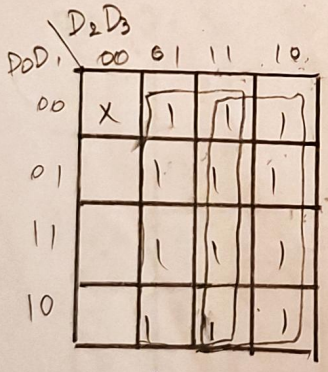
$y = D_2 + D_3 + D_6 + D_7$

$z = D_1 + D_3 + D_5 + D_7$

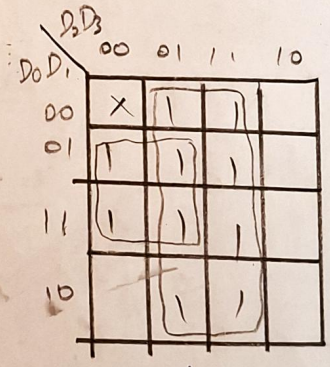


Priority encoder (1 variable)

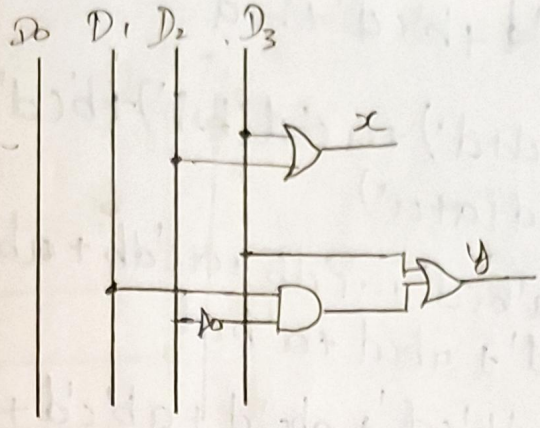
D_0	D_1	D_2	D_3	x	y
0	0	0	0	X	X
0	0	0	1	1	1
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	1	1	1
0	1	1	0	1	0
0	1	1	1	1	1
1	0	0	0	0	0
1	0	0	1	1	1
1	0	1	0	1	0
1	0	1	1	1	1
1	1	0	0	0	1
1	1	0	1	1	1
1	1	1	0	1	0
1	1	1	1	1	1



$x \Rightarrow f = D_3 + D_2$

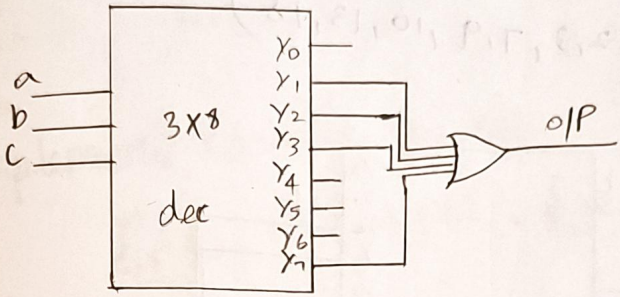


$y \Rightarrow f = D_3 + D_1 D_2'$



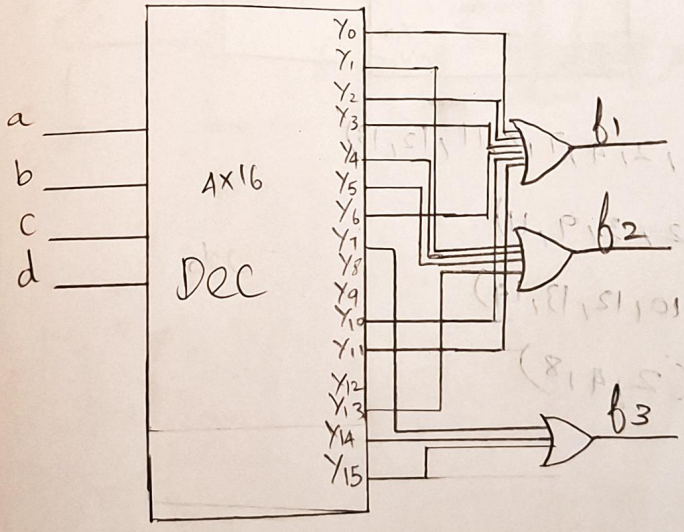
Implement

$f = \Sigma m\ (1,2,3,7)$ using appropriate decoder.



Implement $f_1 = \Sigma m\ (0,2,3,6,10,11)$, $f_2 = \Sigma m\ (1,4,5,13)$

$f_3 = \Sigma m\ (7,14,15)$ using appropriate decoder.



1) $f = a'b'c + ac'd + b'cd' + bcd$

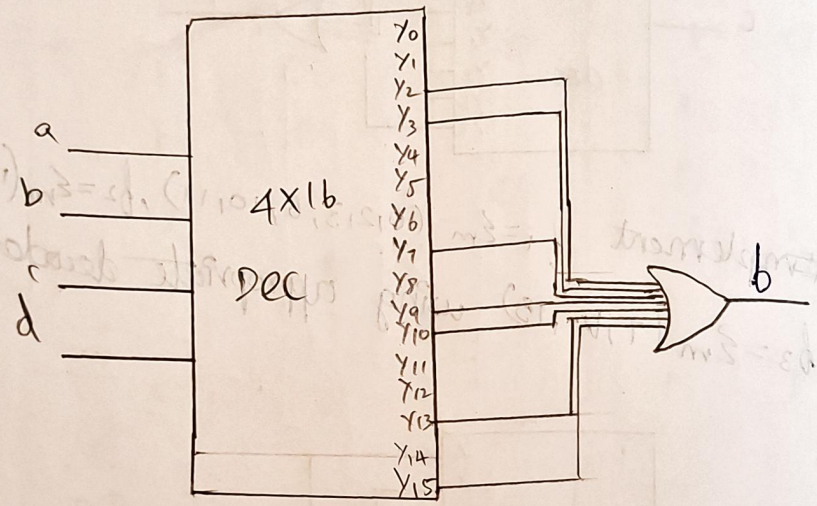
$= a'b'c(d+d') + ac'd(b+b') + b'cd'(a+a') + bcd(a+a')$

$= a'b'cd + a'b'cd' + ac'db + ac'db' + ab'cd' + a'b'cd' + abcd + a'bcd$

$f = a'b'cd + a'b'cd' + abc'd + ab'c'd + ab'cd' + abcd + a'bcd$

$f = \Sigma m(3, 2, 13, 9, 10, 15, 7)$

$f = \Sigma m(2, 3, 7, 9, 10, 13, 15)$

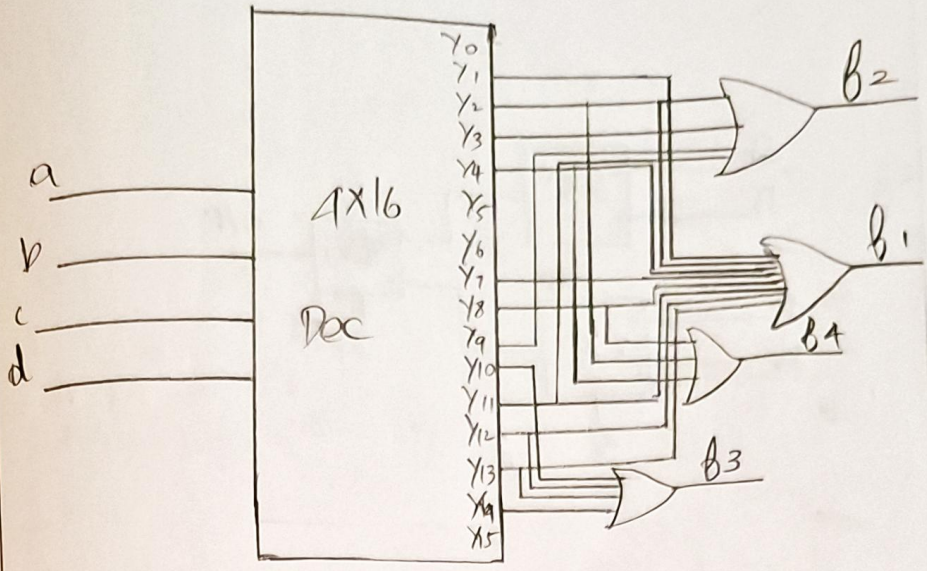


2) $f_1 = \Sigma m(1, 2, 4, 7, 8, 11, 12, 13)$

$f_2 = \Sigma m(2, 3, 9, 11)$

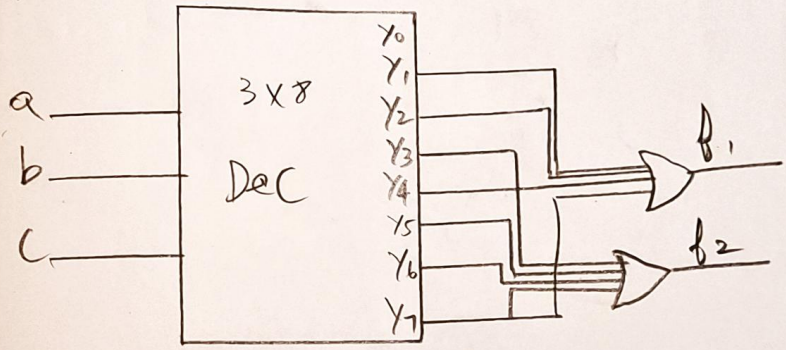
$f_3 = \Sigma m(10, 12, 13, 14)$

$f_4 = \Sigma m(2, 4, 8)$

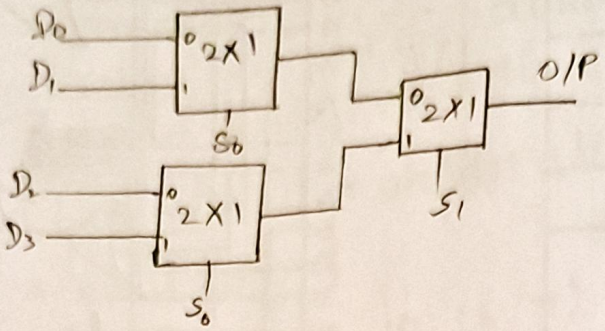


3) Implement FA using decoder

$f_1 = \Sigma m\,(1,2,4,7)$, $f_2 = \Sigma m\,(3,5,6,7)$



1) Implement 4×1 Multiplexer using 2×1 Multiplexer



Sequential logic Circuits

1) latch → NOR, NAND → Store single bit of info (1/0) (holds the value)

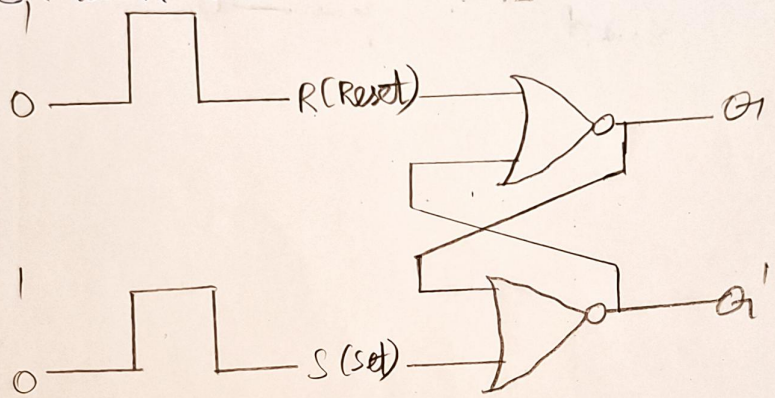
2) flipflops → Stores single bit of info (1/0) (No of bits = No of flipflops)
↓
clk input

In all the latch there is two set of inputs (S and R)

S - Set
R - Reset

Output is Q and Q'

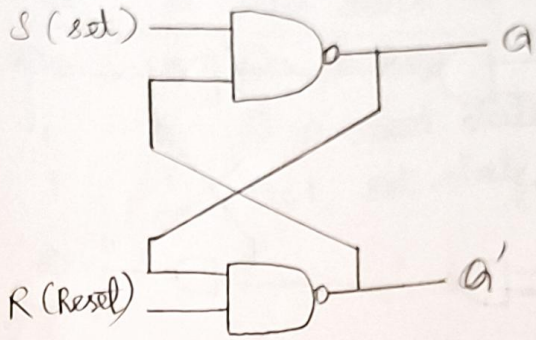
SR latch with NOR Gates:



a	b	O/P
0	0	1
0	1	0
1	0	0
1	1	0

S	R	Q	Q'	
1	0	1	0	
0	0	1	0	(after S=1, R=0) } Set
0	1	0	1	
0	0	0	1	(after S=0, R=1) } Reset
1	1	0	0	(forbidden) or Indeterminate State

SR latch with NAND Gates:

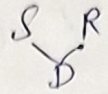


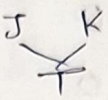
a	b	O/P
0	0	1
0	1	1
1	0	1
1	1	0

S	R	Q	Q'	
1	0	0	1	
1	1	0	1	(after S=1, R=0) } Reset
0	1	1	0	
1	1	1	0	(after S=0, R=1) } Set
0	0	1	1	(forbidden) or Indeterminate State

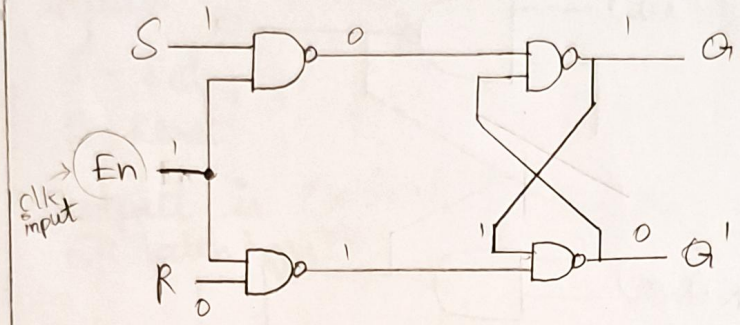
Flipflops:

1) SR f/f } Similar
2) D f/f
3) JK f/f } Similar
4) T f/f



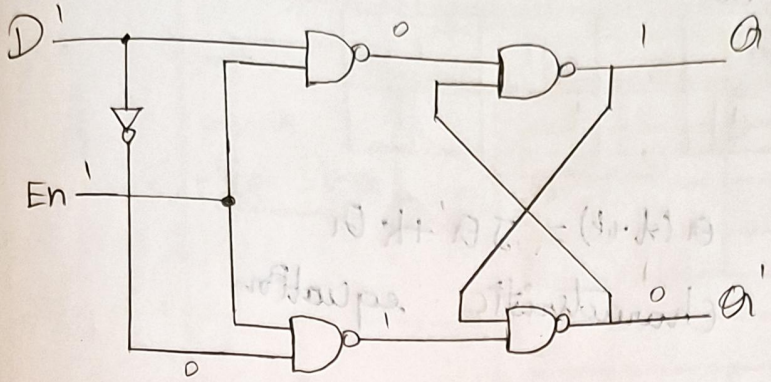


SR Flipflop: (SR latch with control input)



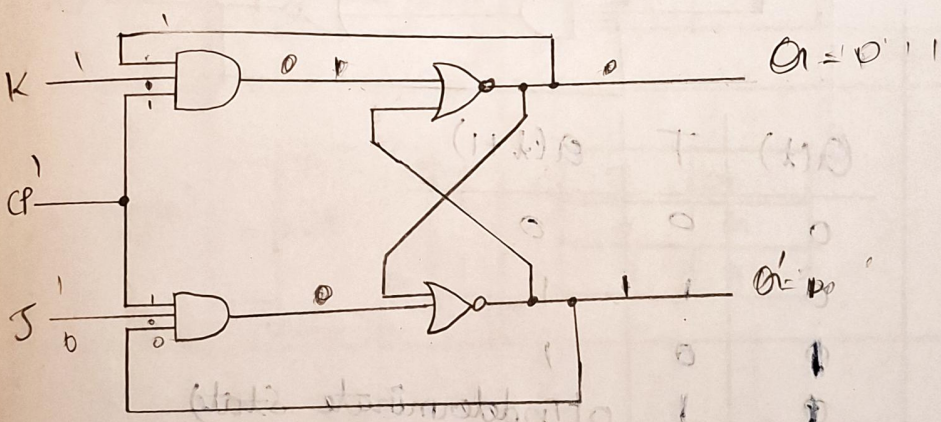
En	S	R	Next state of Q
0	x	x	No change
1	0	0	No change
1	0	1	Q=0 reset state
1	1	0	Q=1 set state
1	1	1	Indeterminate

D flipflop : (D latch)



En	D	Next state of Q
0	x	No change
1	0	Q = 0 reset state
1	1	Q = 1 set state

JK flipflop :



Q(t)	J	K	Q(t+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

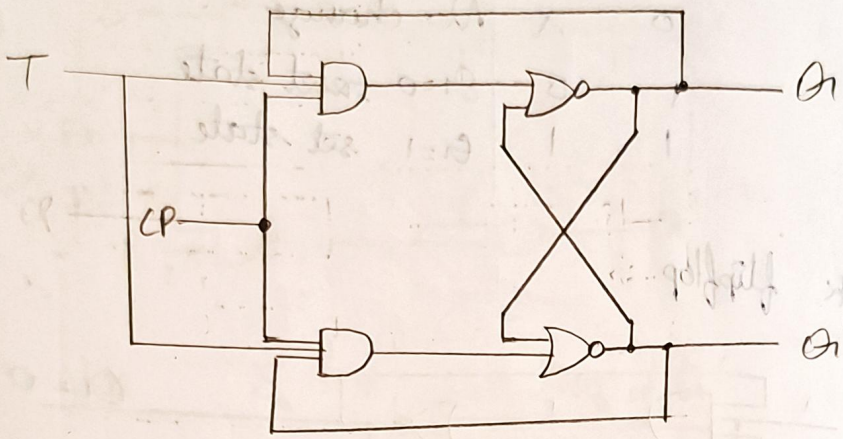
Characteristic Table

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

$$Q(t+1) = JQ' + KQ$$

Characteristic equation

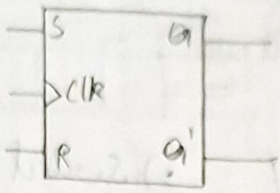
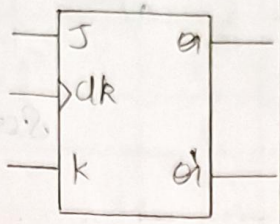
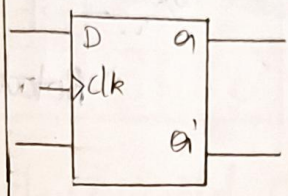
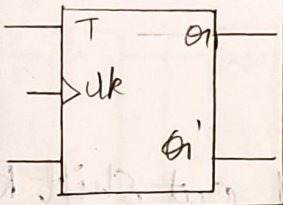
T- flipflop:



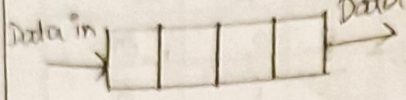
$Q(t)$	T	$Q(t+1)$
0	0	0
0	1	1
1	0	1
1	1	0 (Indeterminate State)

Q \ T	0	1
0		1
1	1	

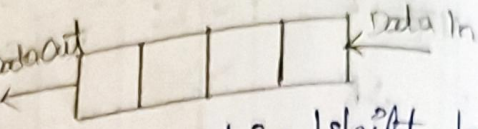
$$Q(t+1) = QT' + Q'T$$
$$= Q \oplus T$$

Flipflop Name	Flipflop Symbol	Characteristic Table
SR		S R Q 0 0 Q 0 1 0 1 0 1 1 1 ?
JK		J K Q 0 0 Q 0 1 0 1 0 1 1 1 Q'
D		D Q 0 0 1 1
T (Toggle)		T Q 0 Q 1 Q'

BASIC MOVEMENT INVOLVED SHIFT REGISTERS

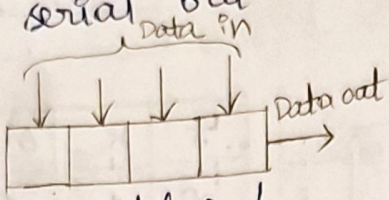


1) Serial in / shift right / serial out



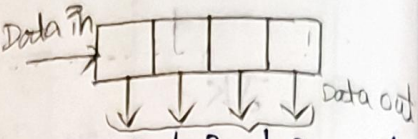
2) Serial in / shift left / serial out

3)



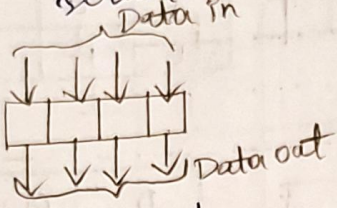
Parallel in / serial out

4)



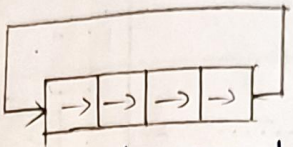
Serial in / Parallel out

5)



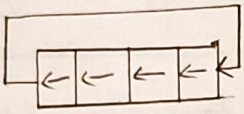
Parallel in / Parallel out

6)



Rotate Right

7)

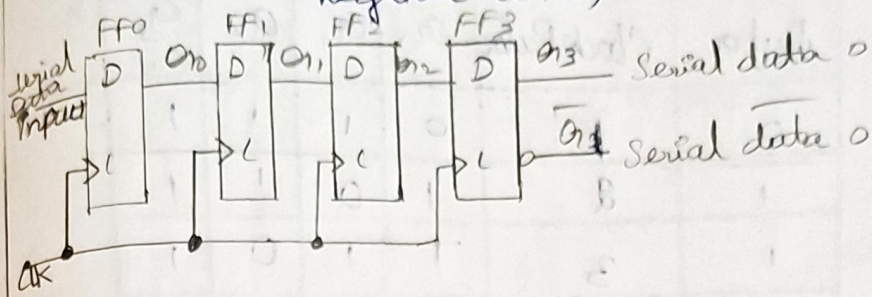


Rotate Left

Type of Register:

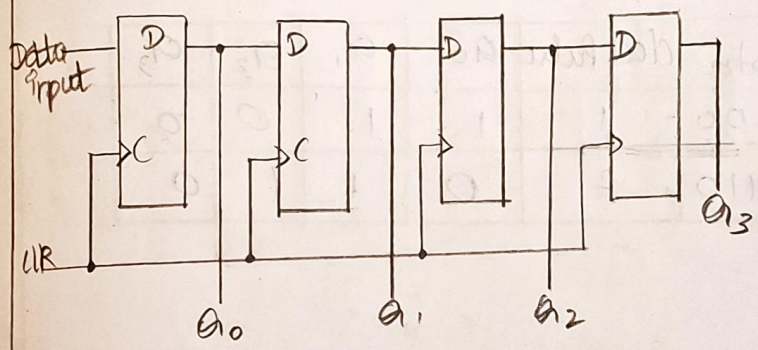
Serial In / Serial Out Shift Register (SISO)
Serial In / Parallel Out Shift Register (SIPO)
Parallel In / Serial Out Shift Register (PISO)
Parallel In / Parallel Out Shift Register (PIPO)

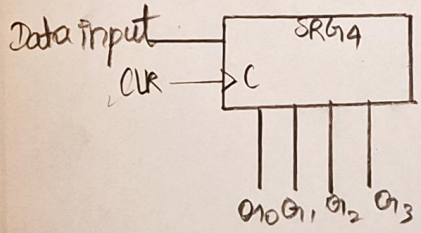
Serial In Serial Out Shift Register (SISO)



Data	Clock Pulse	Q_0	Q_1	Q_2	Q_3
0	0	0	0	0	0
1	1	1	0	0	0
1	2	1	1	0	0
1	3	1	1	1	0
1	4	1	1	1	1
0	5	0	1	1	1
0	6	0	0	1	1

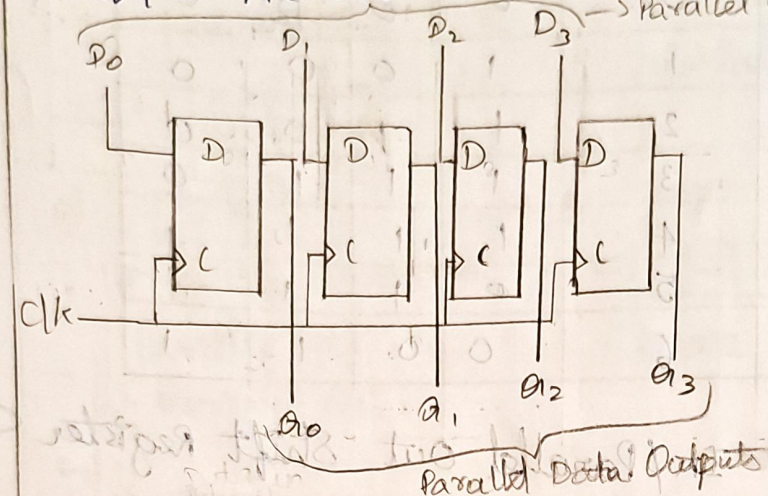
Serial In/Parallel Out Shift Register (SIPO):





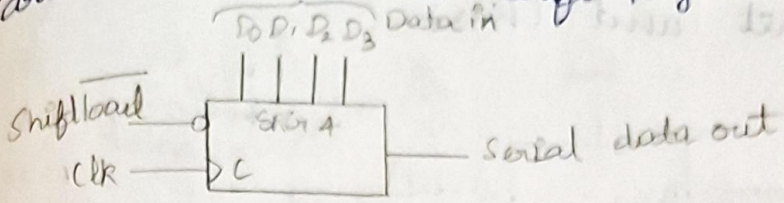
Data	Clock Pulse	Q_0	Q_1	Q_2	Q_3
0	1	0	1	1	1
1	2	1	0	1	1
1	3	1	1	0	1
0	4	0	1	1	0

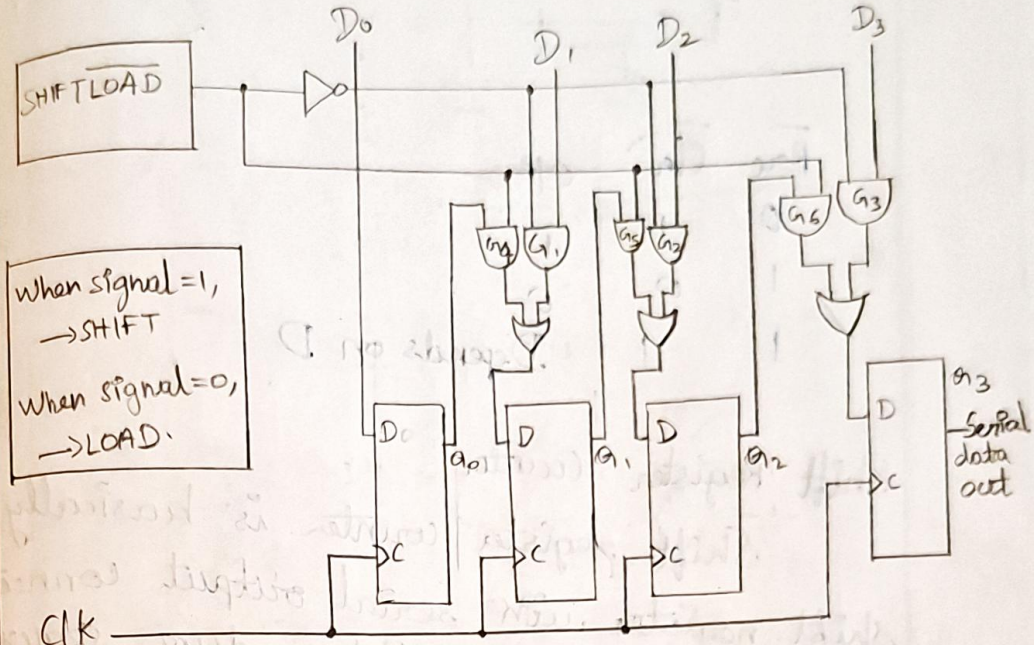
Parallel In / Parallel Out Shift Register (PIPO):



Data	Clock Pulse	Q_0	Q_1	Q_2	Q_3
1100	1	1	1	0	0
0110	2	0	1	1	0

Parallel In / Serial Out Shift Register (PISO):

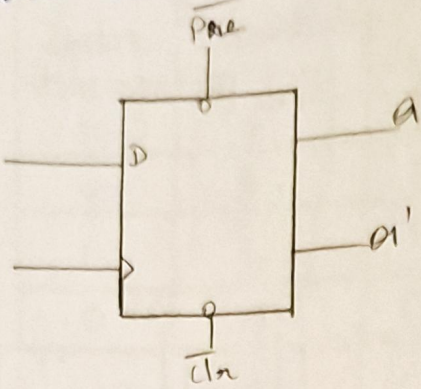




When signal = 1,
→ SHIFT

When signal = 0,
→ LOAD.

Preset and Clear:



$\overline{Pre} = 0$
$\overline{Clr} = 0$
enabled

(Active low)

Preset = 1
Clear = 0

$\overline{Pre}$	$\overline{Clr}$	O/P
0	1	1
1	0	0
1	1	Depends on D

Shift Register Counter

Shift register counter is basically a shift register with serial output connect back to serial input to produce sequences.

Types

1. Johnson Counter

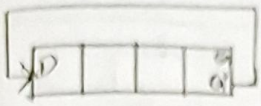
Complement of output of last FF is connected back to D input of 1st FF.

Also known as twisted ring counter.

2. Ring Counter

Output of last FF is connected back to D input of 1st FF.

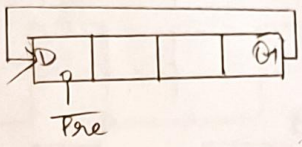
Johnson counter :



clear all inputs

Q_0	Q_1	Q_2	Q_3	Q_3'
0	0	0	0	1
1	0	0	0	1
1	1	0	0	1
1	1	1	0	1
1	1	1	1	0
0	1	1	1	0
0	0	1	1	0
0	0	0	1	0
0	0	0	0	1

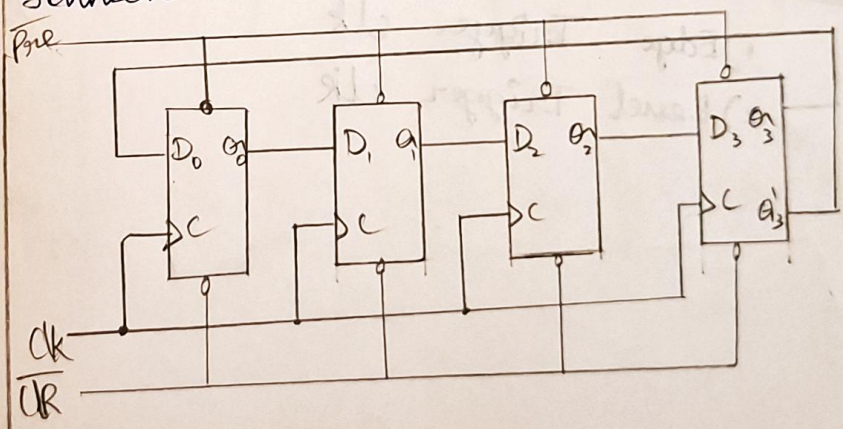
Ring counter



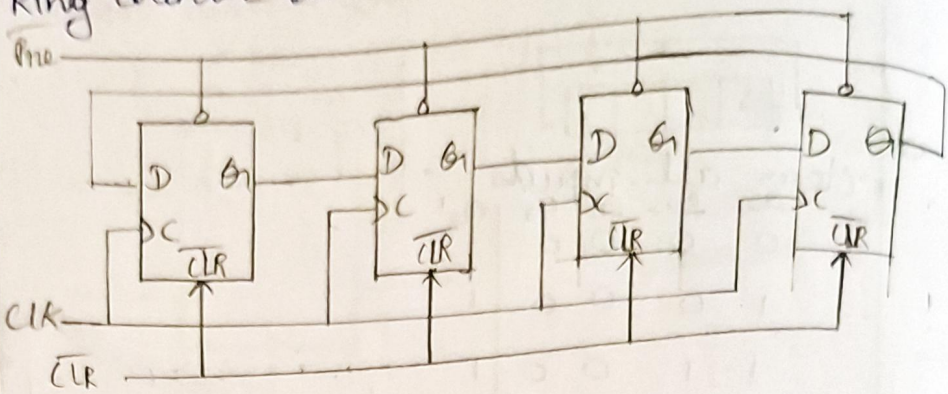
Initial clr of inputs

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

Johnson circuit diagram:

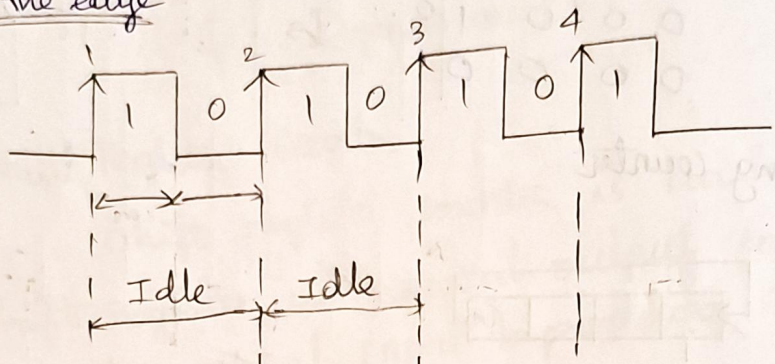


Ring counter circuit diagram:

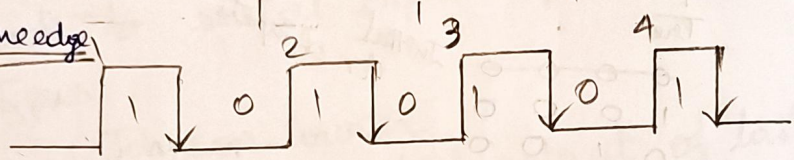


Clk - Edge triggered < +ive edge / -ive edge

+ive edge



-ive edge

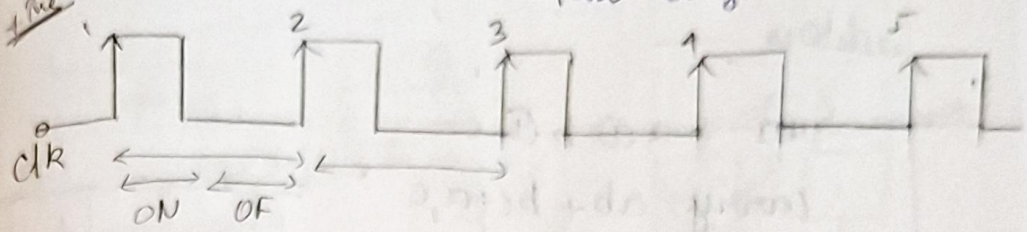


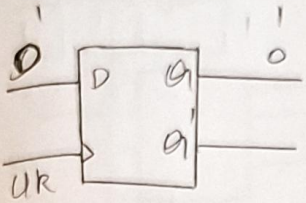
Clk

1) Edge Trigger clk
2) Level Trigger clk

Edge Trigger clk: < +ive edge / -ive edge

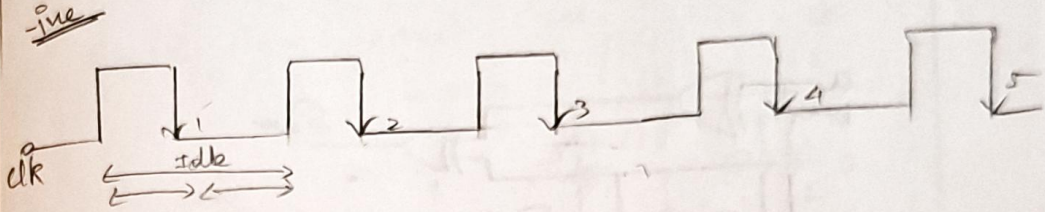
+ive



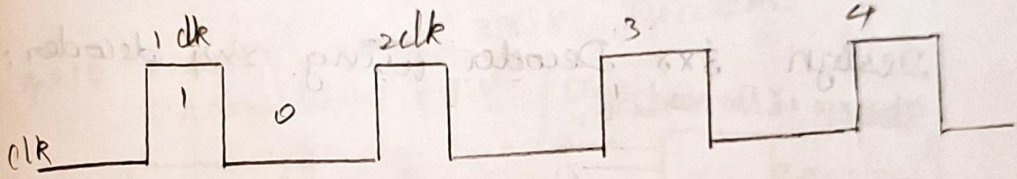


1 clk D=1 O/P=1
2 clk D=0 O/P=0
3 clk D=1 O/P=1

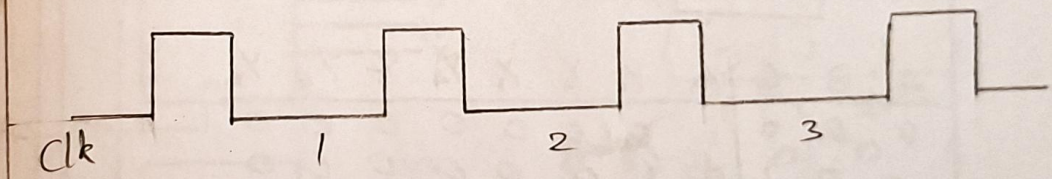
-ive



Level trigger clk:



+ive



Design of full adders using two half adders

(No need)

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

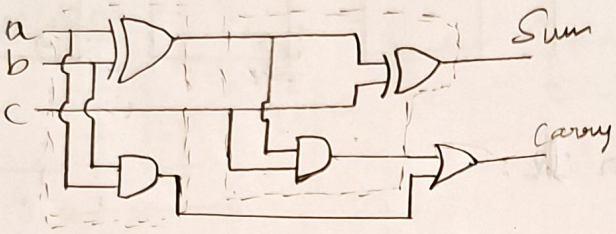
$$Sum = a \oplus b \oplus c$$

$$carry = ab + bc + ac$$

$$carry = ab + a'bc + ab'c$$

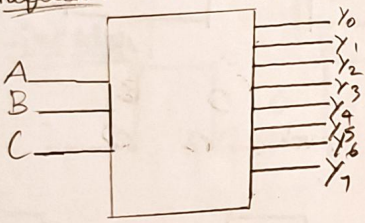
$$= ab + c(a'b + ab')$$

$$= ab + c(a \oplus b)$$



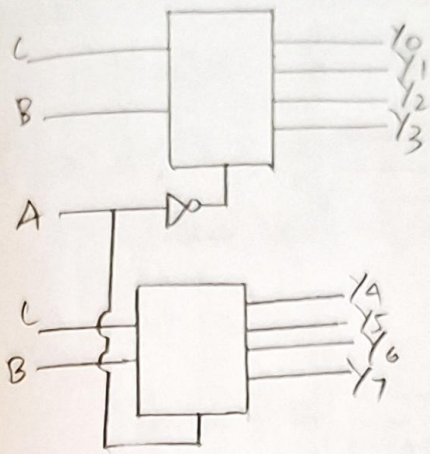
Design 3x8 Decoder using 2x4 decoder:

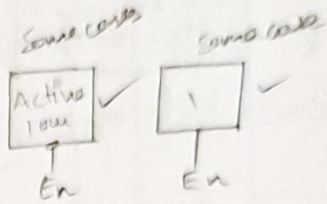
Reference: (No need)



A	B	C	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_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

(X) Imp





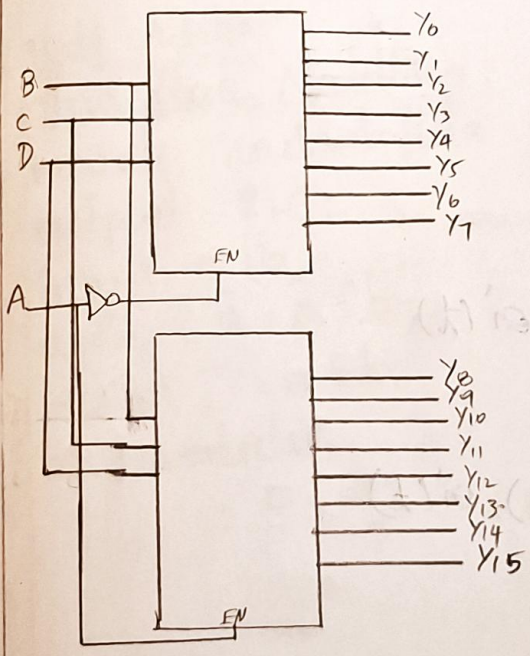
Enable Pin = 1 (Active)

	A	B	C
1st 2x4	0	0	0
	0	0	1
	0	1	0
	0	1	1
2nd 2x4	1	0	0
	1	0	1
	1	1	0
	1	1	1

4x16 Decoder using 3 x 8 Decoder:

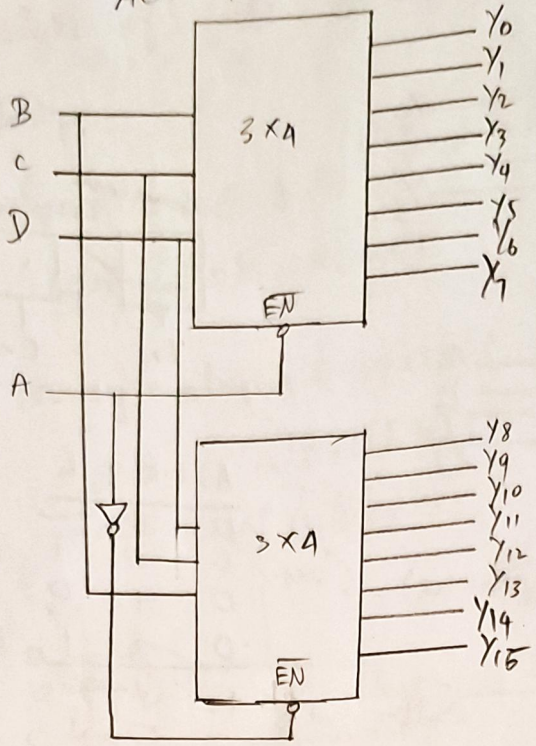
Active high EN=1

Divide active



	a	b	c	d
8 values 1st 3x4	0	0	0	0
	0	0	0	1
	0	1	1	1
8 values 2nd 3x4	1	0	0	0
	1	0	0	1
	1	1	1	1

Active low EN=0 Device active



Counters:

1) Asynchronous counter (T/JK)
2) Synchronous counter (Any #F)

2bit (00, 01, 10, 11)
→ 4 f/f

T f/f

T=1 $Q(t+1) = Q'(t)$

JK f/f

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

Hardware Description Language (HDL)

Types:

1. Verilog - stands for verification Logic
2. VHDL - Hardware Description Language

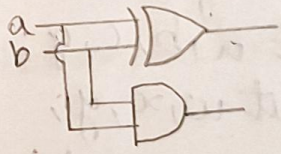
   VHSIC - Very high speed integerated circuits

Verilog:

1) Data flow Modelling
2) Gate level Modelling
3) Behavioral Modelling
4) Structural Modelling

Not = ~
or = |
and = &
xor = ^

Half Adder

Data flow Modelling:

```
module haddataflow (s, c, a, b);
output s, c;
input a, b;
assign s = a ^ b;
assign c = a & b;
endmodule
```