

# **Digital Systems ( ICT 2154 ) & Digital Systems and Computer Organization (ICT 2171)**

## **4 credit**

Faculties : Mr. Ghanashyama Prabhu & Dr. Rashmi N.R.

I&CT Department, MIT Manipal

[gs.prabhu@manipal.edu](mailto:gs.prabhu@manipal.edu)

[rashmi.naveen@manipal.edu](mailto:rashmi.naveen@manipal.edu)

**Students need to write notes for each class**

# Abstract Syllabus of DSM:

- Introduction, Simplification of Boolean functions – K-map and tabulation method, NAND and NOR implementation, Combinational logic- Design of Adders/Subtractors, Binary Parallel adder[7483], Carry Look ahead Adder [74182], Multiplier using 7483, BCD adder, Magnitude Comparator [7485], Decoder [74138,7442], Combinational logic circuit design using decoders, Encoder [74148], Multiplexers [74157, 74153], Combinational logic circuit design using multiplexers, De Multiplexers, ROMS and Programmable Logic Arrays, Sequential logic –Asynchronous and Synchronous counters, Synchronous counter design, Shift registers, Shift register counters, Analysis and design of clocked sequential circuits, Memory Devices - RAM, ROM, PROM, EPROM, EEPROM, PLD.

# Abstract Syllabus of DSCO:

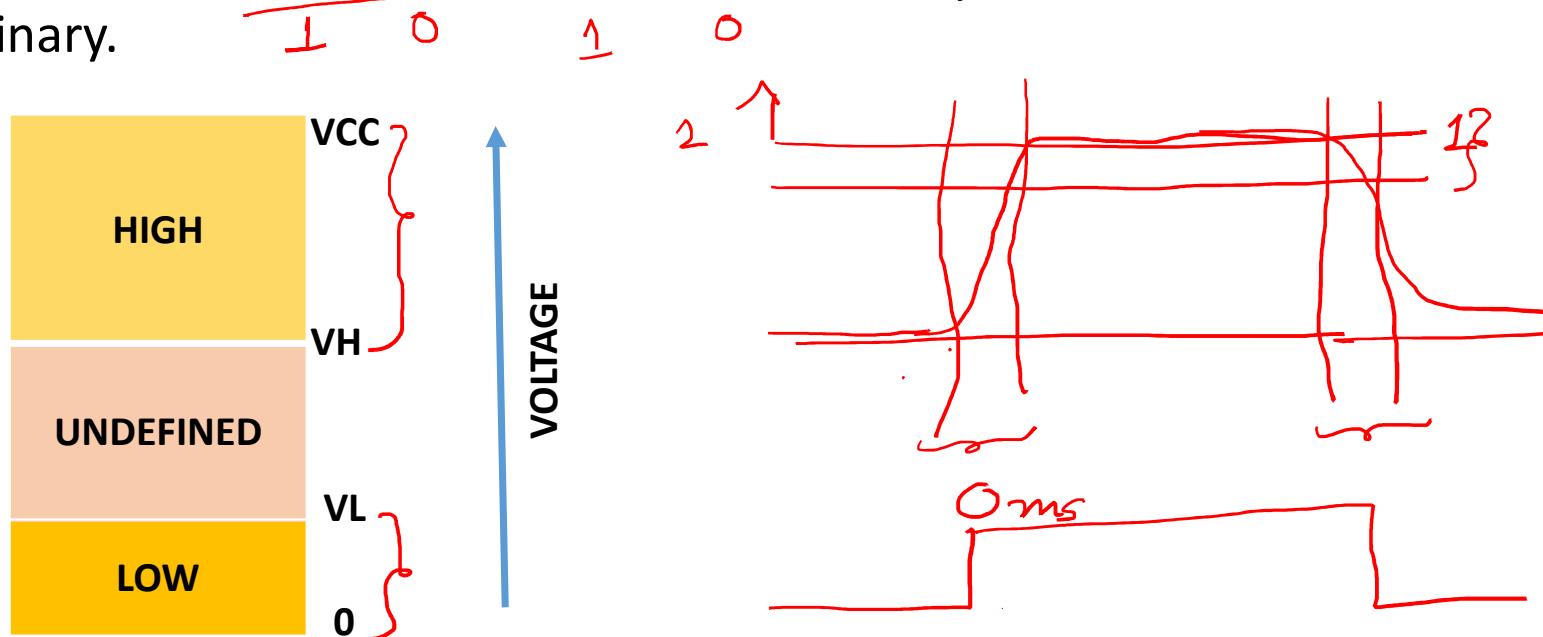
- Introduction, Simplification of Boolean functions – K-map method, NAND and NOR implementation, Combinational logic, Design of Adders/Subtractors, code converters, Application of typical TTL integrated circuit components like Binary Parallel adder[74283], multiplier using 74283, Magnitude Comparator [7485], Decoders [74138,7442], Encoders [74148], Multiplexers [74157], combinational shifter design, De Multiplexers, Sequential logic –counters and shift registers, Computer organization- Introduction, ALU unit, Control unit, Hardwired and Micro – programming approach, Memory unit, Input and Output unit.

# References:

- M. Morris Mano: Digital Design, Third edition, Pearson Education, 2013
- Ronald J. Tocci and Neal S. Widmer : Digital Systems, 9th Edition, Pearson Education, 2007.
- J. F. Wakerly, Digital Design Principles and Practices, 3rd edition, Pearson Education, 2003
- Computer Organization:
  - Mohamed Rafiquzzaman and Rajan Chandra, Modern computer Architecture (3e), Galgotia publications Pvt. Ltd, 2015.
- Additional reference:
  - Digital principles and Design by Donald D. Givone

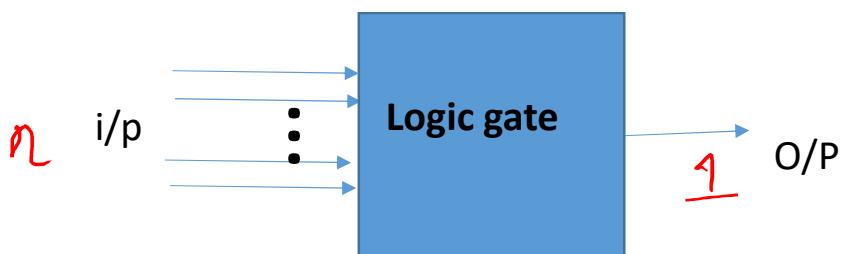
# Difference between Analog and Digital

- Analog information can take any values in between the range
- Digital information takes discrete values. At the basic level, value can be ON/OFF, ONE/ZERO, TRUE/FALSE, HIGH/LOW i.e. only two values and hence the name binary.



# Logic gates

- Building blocks of digital circuit
- Electronic circuits with one or more inputs and one output.



# Basic logic gates

- AND , OR, NOT gates
- Graphic symbols of these 3, 2-i/p gates are shown below

AND



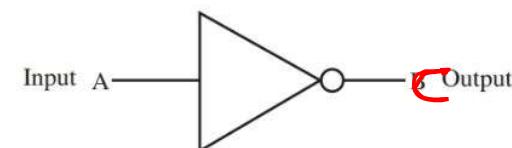
$$\underline{C = A \cdot B} = \underline{\textcircled{AB}}$$

OR



$$\underline{C = A + B}$$

NOT



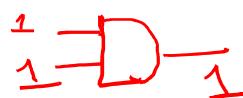
$$\underline{C = A'} = \underline{\overline{A}}$$

# Truth Table

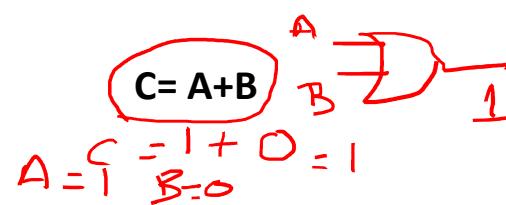
- Truth table of a logic gate or a logic circuit depicts the output for all possible input combinations
- AND

INPUTS		OUTPUT
A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

$$C = A \cdot B$$



INPUTS		OUTPUT
A	B	C
0	0	0
0	1	1
1	0	1
1	1	1



OR

NOT

INPUT	OUTPUT
A	C
0	1
1	0

$$\frac{1}{2} = 2$$

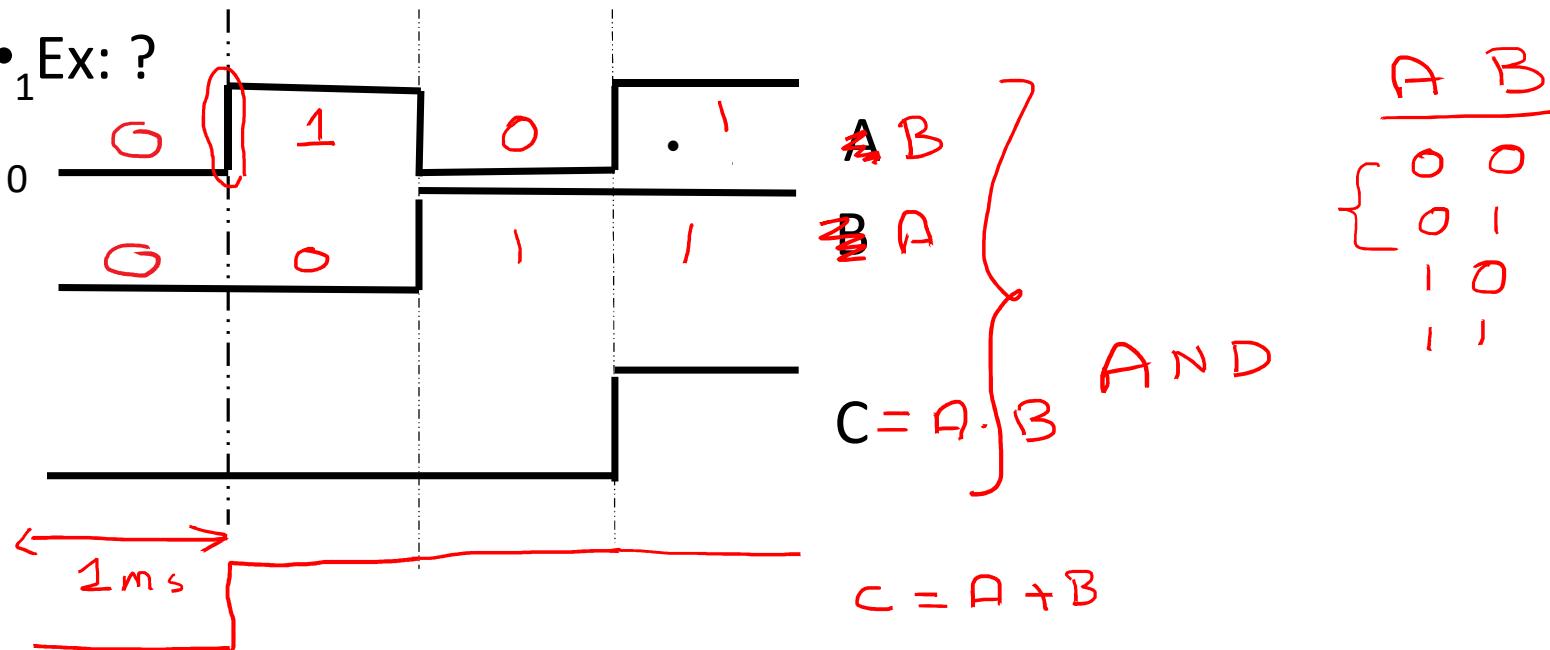
No. of IP variables  
2 = 4

No. of IP combinations

## Timing diagram

- Represents the output of the gate for input signal combinations

- Ex: ?



$$C = A + B$$

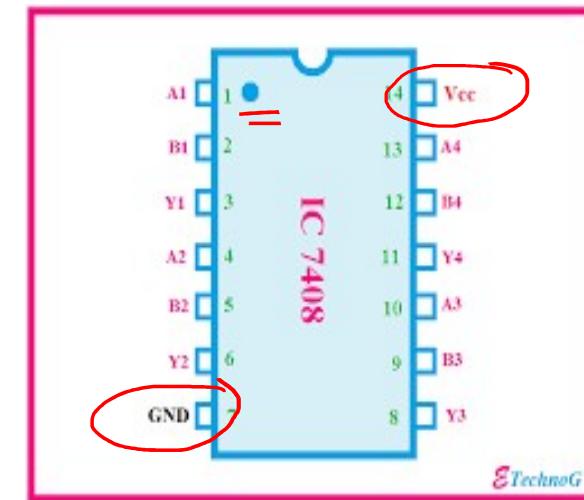
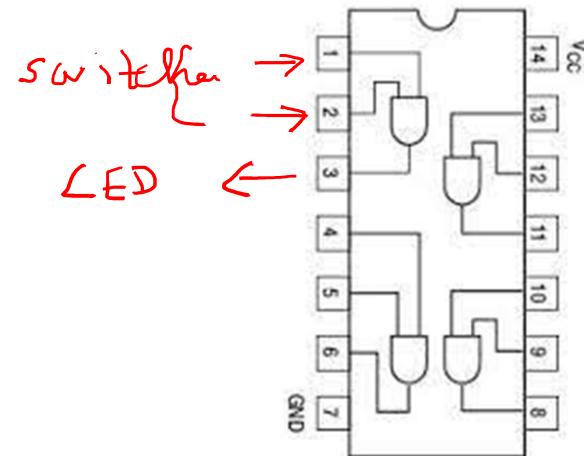
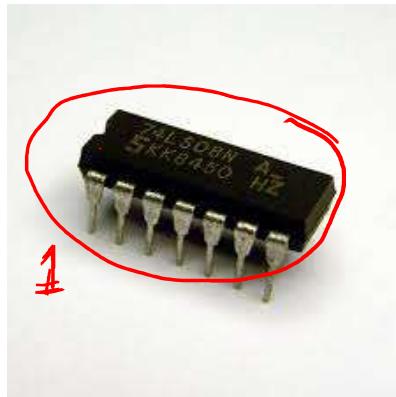
DRAW THE TIMING DIAGRAM FOR OR GATE ✓

$$\underline{C = A + B}$$

**DRAW THE TIMING DIAGRAM FOR OR GATE**

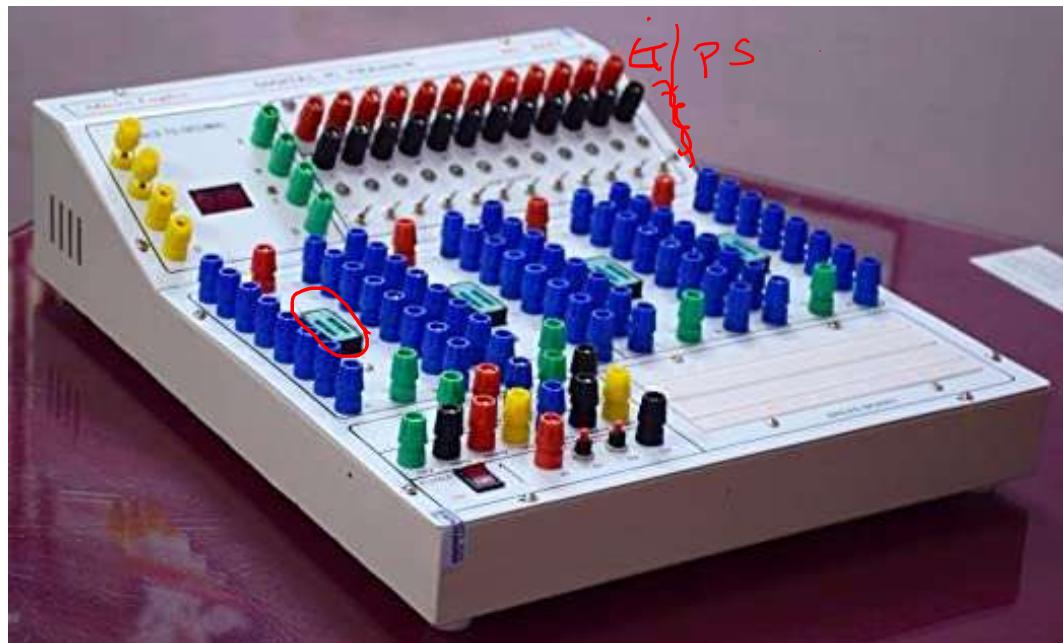
# How do gates look physically

- Gates are available in the form of Integrated Circuits (ICs)



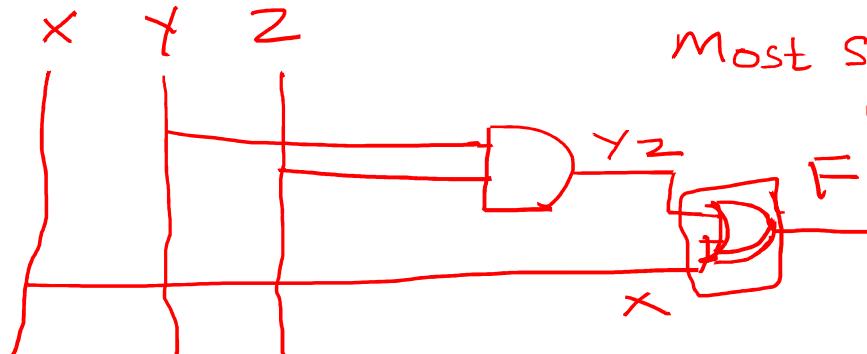
- Write the full form of SSI, MSI, LSI, VLSI, ULSI, ? And differentiate

# Digital Trainer Kit



# Boolean function and Logic circuit

- For the Boolean function  $F(x,y,z) = x + \underline{yz}$ , draw the logic circuit, timing diagram and truth table.



$2^3 = 8$

Most sign. Bit (MSB)

$F = x + yz$

	X	Y	Z	F
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

Least significant bit (LSB)

$$2^3 = 8 \Rightarrow 0 \text{ to } 7$$

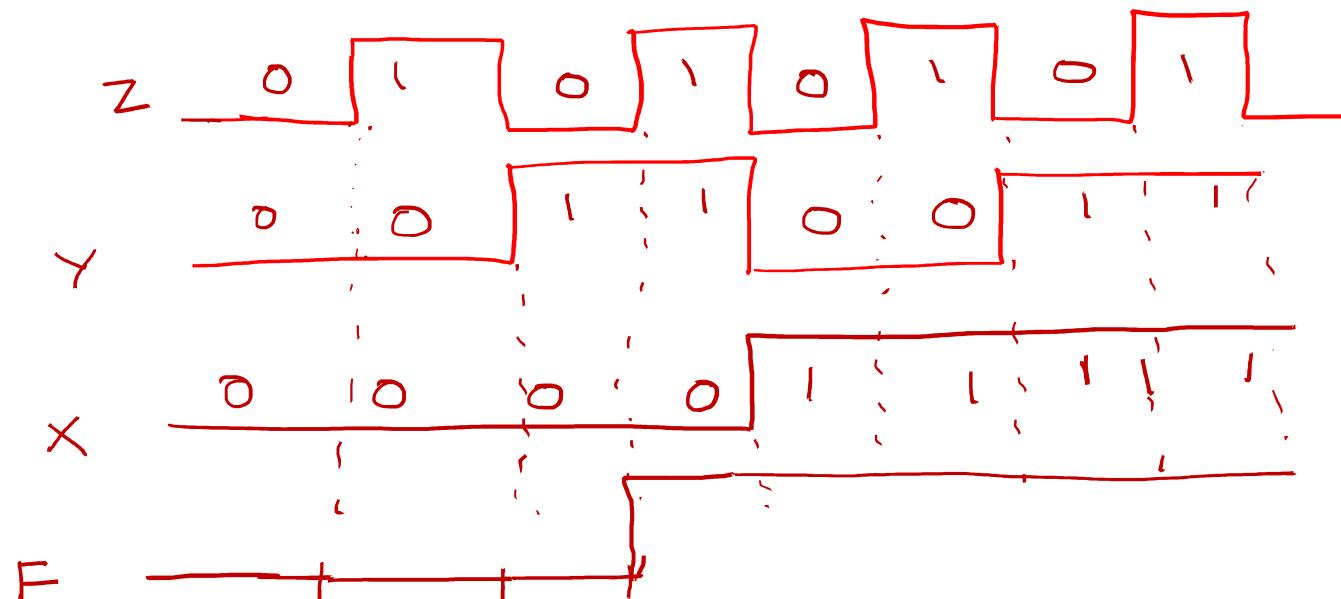
000

(11)

$$4 + 2 + 1 = 7$$

## **Truth Table: $F(x,y,z) = x + yz$**

Timing Waveform:  $F(x,y,z) = x + yz$



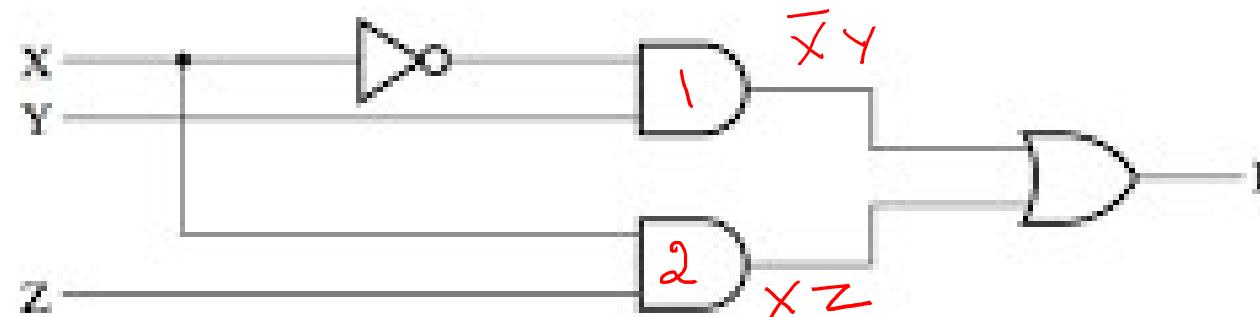
Interchanging x and z  $\rightarrow$  LSB  $\rightarrow$  MSB

	4	2	
	$Z$	$Y$	$X$
	0	0	0
	0	0	1
	0	1	0
	0	1	1
	1	0	0
	1	0	1
	1	1	0
	1	1	1

LSB

MSB

- Write the Boolean function and truth table for the following circuit..



$$F = \underline{\bar{X}Y} + \underline{XZ}$$

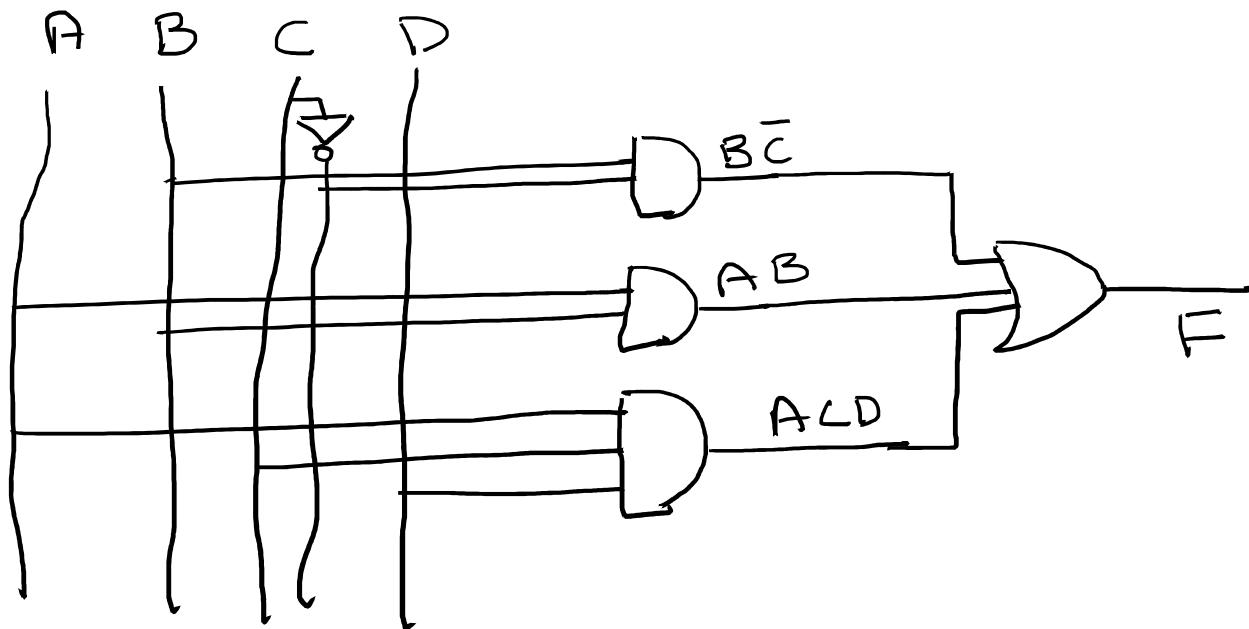
Truth Table for  $F = \overline{x'y} + \overline{xz}$

$x$	$y$	$z$	$\bar{F}$
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

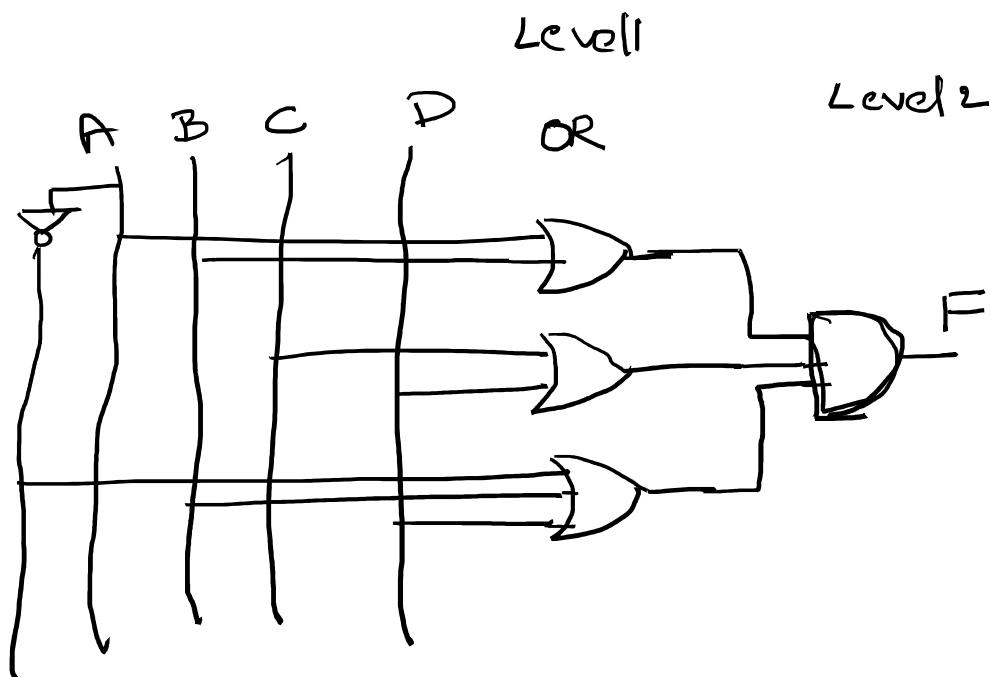
**Draw the logic diagram (without simplification) using basic logic gates**

- $F = \underline{BC}' + \underline{AB} + \underline{ACD}$
- $F = (A+B)(C+D)(A'+B+D)$

$$F = \underline{\underline{BC'}} + \underline{\underline{AB}} + \underline{\underline{ACD}} \Rightarrow \text{Sum of product}$$



$$F = \underline{\underline{A+B}} \underline{\underline{C+D}} \underline{\underline{A'+B+D}} \rightarrow \text{product of Sums.}$$



# Boolean Theorems

**Table 2.1**  
*Postulates and Theorems of Boolean Algebra*

Postulate 2	(a)	$x + 0 = x$		(b)	$x \cdot 1 = x$
Postulate 5	(a)	$x + x' = 1$		(b)	$\underline{x \cdot x'} = 0$
Theorem 1	(a)	$x + x = x$		(b)	$\underline{x \cdot x} = x$
Theorem 2	(a)	$\underline{\underline{x}} + 1 = 1$		(b)	$\underline{\underline{x}} \cdot 0 = 0$
Theorem 3, involution		$\underline{(x')} = x$			
Postulate 3, commutative	(a)	$x + y = y + x$		(b)	$xy = yx$
Theorem 4, associative	(a)	$x + (y + z) = (x + y) + z$		(b)	$x(yz) = (xy)z$
Postulate 4, distributive	(a)	$x(y + z) = xy + xz$		(b)	$x + yz = (x + y)(x + z)$
Theorem 5, DeMorgan	(a)	$\underline{(x + y)} = x'y'$		(b)	$\underline{(xy)} = x' + y'$
Theorem 6, absorption	(a)	$\underline{x + xy} = x$		(b)	$x(x + y) = x$

$$x(1 \cdot y) = x$$

$$x + xy = x(1 + y) = x$$

- Rough

$$x = 0 \\ 1 \quad \text{XOR gate} \quad F = x + x = x$$

$$x \quad \text{XOR gate} \quad x \cdot x = x$$

$$x = 0 \ 0 \\ x = 1 \ 1$$

$$x \cdot 1 = x$$

$$\begin{aligned} & x + x + x + x + x = x \\ & AB \cdot 1 \\ & AB + (ABC) + A\bar{B}\bar{C} \cancel{+ A\bar{B}C} \overline{A\bar{B}C} \\ & AB(1 + C) + AC(B + \bar{B}) \\ & \underline{\underline{AB + AC}} \end{aligned}$$

**Rough**

## Simplify the following Boolean functions

$$\bullet X(X' + Y) = \cancel{X}Y$$

$$\bullet X + X'Y = \cancel{(X+X')} (X+Y) = X+Y$$

$$\bullet X' + XY = ? \quad \cancel{X'} + Y$$

$$\bullet (X + Y)(X + Y') = \cancel{X}$$

$$\left. \begin{array}{l} \text{Ex: } \underbrace{AB}_{X} + \overline{AB} \cdot C = AB + C \\ \quad \quad \quad \overline{X} \cdot C = X + C \\ \text{Ex: } AB + \overline{\cancel{A}\cancel{B}} \cdot C \end{array} \right\}$$

## Simplify following Boolean expressions to one term using Boolean theorems

- $A'B(D'+C'D)+B(A+A'CD)$

$$= \overline{A}B\overline{D} + \underline{\overline{A}B\overline{D}} + AB + \underline{\overline{A}B\overline{D}}$$

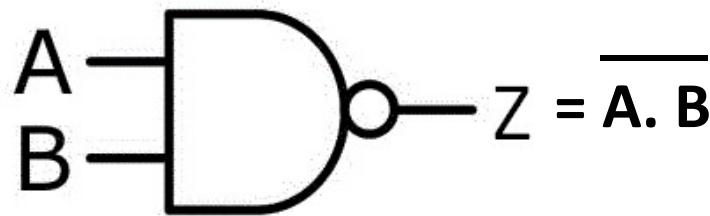
$$= \underline{\overline{A}B\overline{D}} + \overline{A}B\overline{D} + AB$$

$$= \underline{\overline{A}B} + \underline{AB} = B$$

**Simplify following Boolean expressions to one term using Boolean theorems : $A'B(D'+C'D)+B(A+A'CD)$**

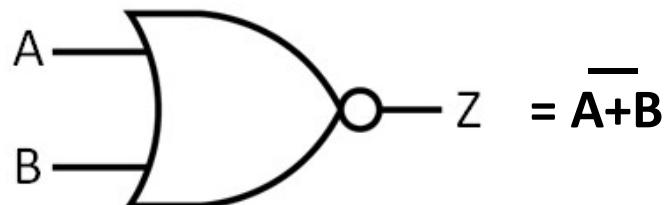
# Universal gates

- NAND gates



A	B	$Z = \overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

- NOR gates



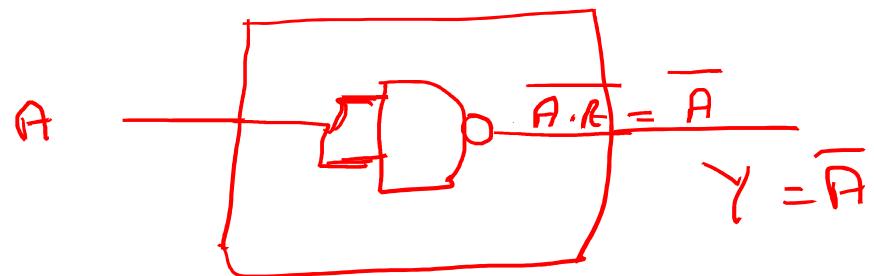
A	B	$Z = \overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

## Implement basic logic gates using NAND gates

- NOT using NAND gate:

$$Y = \overline{A}$$

$$Y = \overline{A \cdot B}$$

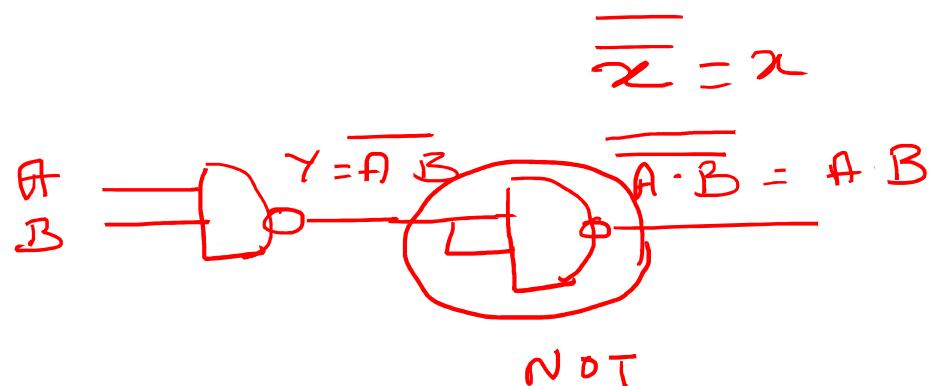


## Implement basic logic gates using NAND gates

- AND using NAND gate:

$$\text{AND } Y = A \cdot B$$

$$\text{NAND: } Y = \overline{\overline{A} \cdot \overline{B}}$$



## Implement basic logic gates using NAND gates

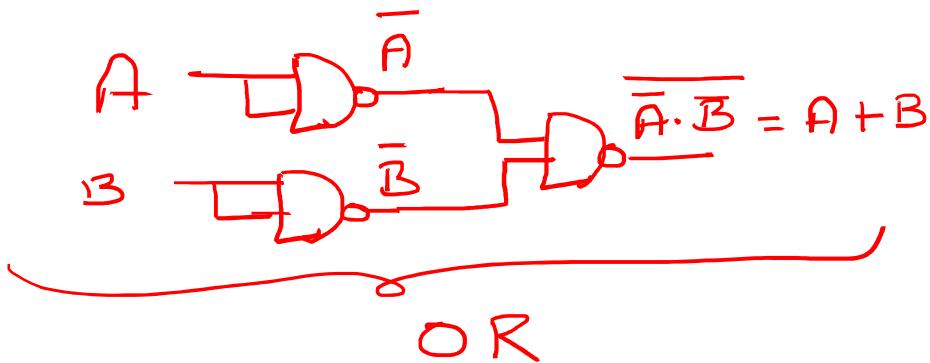
- OR using NAND gate:

$$Y = A + B$$

$$Y = \overline{\overline{A} \cdot \overline{B}} = \overline{\overline{A}} + \overline{\overline{B}}$$

↑ ↑

if we apply  $\overline{\overline{A}}$   $\overline{\overline{B}}$  to NAND  $= \overline{\overline{A} \cdot \overline{B}} = \overline{\overline{\overline{A}}} + \overline{\overline{\overline{B}}} = A + B$

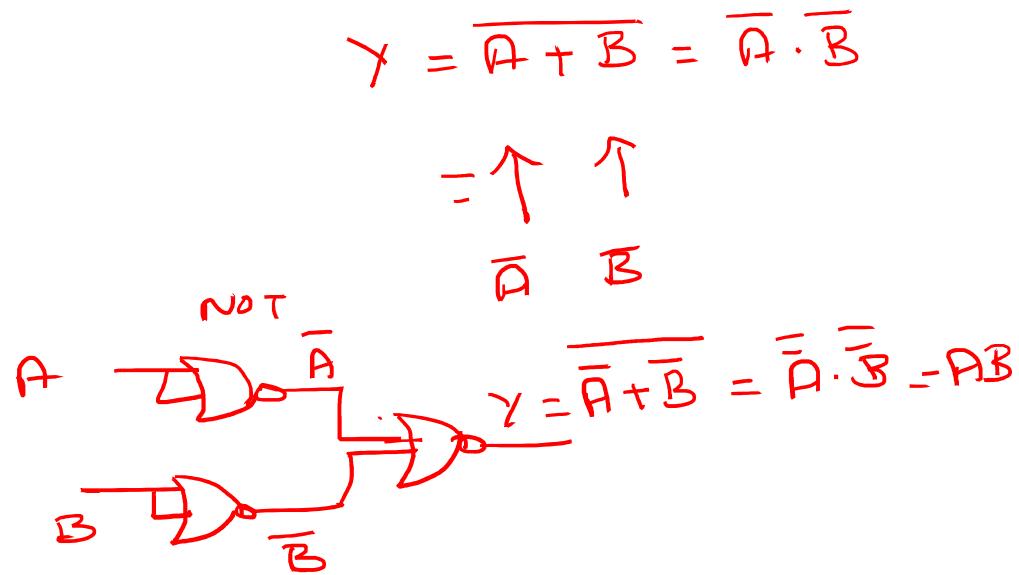


## **Implement basic logic gates using NOR gates**

- NOT using NOR gate:

## Implement basic logic gates using NOR gates

- AND using NOR gates:



## Implement basic logic gates using NOR gates

- OR using NOR gates:

$$Y = \overline{A + B}$$

**Any questions ?**

**III sem B.Tech (IT/CCE)**

**ICT 2154 Digital Systems**  
**&**

**ICT 2171 Digital systems and computer organisation**

- **Topics covered**
- MINTERM, MAXTERM
- Writing Boolean expressions for the given truth table
- Design of a combinational circuit for the given problem
- Realization of a logic circuit using Universal gates

## **MINTERMS & MAXTERMS**

- MIN and MAX are means to represent the inputs through logical AND /OR operations
- Consider a 3-variable Boolean function,  $F(a,b,c)$
- Prepare a table with 6 columns
- In first column, write all possible binary combinations, possible with three input variables, one below the other. Ex:000,001...., ... 111
- In 2<sup>nd</sup> column, write decimal equivalent of all the corresponding input combinations.

	INPUTS A B C	MINTERMS <i>Product</i>	Decimal notation of minterms	Maxterms SUM	Decimal notation of minterms
0	0 0 0	$\bar{A}\bar{B}\bar{C} = 1 \cdot 1 \cdot 1 = 1$	$m_0$	$A + B + C$	$M_0$
1	0 0 1	$\bar{A}\bar{B}C$	$m_1$	$\bar{A} + B + \bar{C}$ $0 + 0 + 0 = 0$	$M_1$
2	0 1 0	$\bar{A}B\bar{C}$	$m_2$	$A + \bar{B} + C$	$M_2$
3	0 1 1	$\bar{A}BC$	$m_3$	$A + \bar{B} + \bar{C}$	$M_3$
4	1 0 0	$A\bar{B}\bar{C}$	$m_4$	$\bar{A} + B + C$	$M_4$
5	1 0 1	$A\bar{B}C$	$m_5$	$\bar{A} + B + \bar{C}$	$M_5$
6	1 1 0	$AB\bar{C}$	$m_6$	$\bar{A} + \bar{B} + C$	$M_6$
7	1 1 1	$ABC$	$m_7$	$\bar{A} + \bar{B} + \bar{C}$	$M_7$

Rough

## Minterms (Standard products) and Maxterms (standard sums)

Row No.	A	B	C	Minterms	Maxterms
0	0	0	0	$A'B'C' = m_0$	$A + B + C = M_0$
1	0	0	1	$A'B'C = m_1$	$A + B + C' = M_1$
2	0	1	0	$A'BC' = m_2$	$A + B' + C = M_2$
3	0	1	1	$A'BC = m_3$	$A + B' + C' = M_3$
4	1	0	0	$AB'C' = m_4$	$A' + B + C = M_4$
5	1	0	1	$AB'C = m_5$	$A' + B + C' = M_5$
6	1	1	0	$ABC' = m_6$	$A' + B' + C = M_6$
7	1	1	1	$ABC = m_7$	$A' + B' + C' = M_7$

## Write the Boolean function for $f(x, y, z)$ given below

x	y	z	f
0	0	0 ✓	1
0	0	1 ✓	1
2	0	1 0	0 ↴
3	0	1 1	0 ↴
	1	0 0	1
	1	0 1	1
7	1	1 0	1
	1	1 1	0 ↴

$$\begin{aligned}
 f(x, y, z) &= \bar{x}\bar{y}\bar{z} + \bar{x}\bar{y}z + x\bar{y}\bar{z} + x\bar{y}z + xy\bar{z} \\
 &= m_0 + m_1 + m_4 + m_5 + m_6 \\
 &= \sum_{m=0}^{101} (0, 1, 4, 5, 6)
 \end{aligned}$$

$$\begin{aligned}
 f(x, y, z) &= \overbrace{(x + \bar{y} + z) \cdot (x + \bar{y} + \bar{z}) \cdot (\bar{x} + \bar{y} + \bar{z})}^{x=0 \ y=0 \ z=0 \quad 1 \cdot 1 \cdot 1 = 1} \\
 &\quad \quad \quad x=1 \ y=1 \ z=1 \Rightarrow f = 1 \cdot 1 \cdot 0 = 0 \\
 &= M_2 \cdot M_3 \cdot M_7 = \pi M(2, 3, 7)
 \end{aligned}$$

Boolean function can be represented as (i) sum of minterms and (ii) product of maxterms.

---

## Sum of Minterms and product of maxterms expressions

x	y	z	f
0	0	0	1
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	0

$$A + \bar{A}B = A + B$$

Sum of minterms : Canonical form ✓

$$\begin{aligned} F(x, y, z) &= \underline{x'y'z'} + \underline{x'y'z} + \underline{xy'z'} + \underline{xy'z} + \underline{xyz} \\ &= m_0 + m_1 + m_4 + m_5 + m_6 \\ &= \Sigma (0, 1, 4, 5, 6) \end{aligned}$$

Sum of product(simplified) : Standard forms

$$F(x, y, z) = \underline{\underline{y}} + \underline{\underline{xy\bar{z}}} = \underline{\underline{y}} + \underline{\underline{x\bar{z}}}$$

Product of maxterms : Canonical form

$$\begin{aligned} F(x, y, z) &= \underline{(x+y'+z)} \cdot \underline{(x+y'+z')} \cdot \underline{(x'+y'+z')} \\ &= M_2 \cdot M_3 \cdot M_7 \\ &= \Pi (2, 3, 7) \end{aligned}$$

Product of sum : Standard forms

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

**Rough**

## Relationship between sum of minterms and product of maxterms

- Sum of minterms : Canonical form

$$\begin{aligned} \bullet F(x, y, z) &= x'y'z' + x'y'z + xy'z' + xy'z + xyz' \\ &= m_0 + m_1 + m_4 + m_5 + m_6 \\ &= \sum_m (0, 1, 4, 5, 6) \quad \checkmark \end{aligned}$$

$\left. \begin{array}{l} f(x, y, z) = \\ \sum_m (0, 1, 4, 5, 6) = \\ \prod_M (2, 3, 7) \end{array} \right\}$

$$F'(x, y, z) = \sum_m (2, 3, 7) = m2 + m3 + m7$$

Taking complement on both sides and applying DeMorgan's theorem

$$\begin{aligned} \bar{F} &= F(x, y, z) = (m2 + m3 + m7)' = m2'.m3'.m7' \\ &= (\underline{x'yz'})' . (\underline{x'yz})' . (\underline{xyz})' = \underline{(x+y'+z)} . \underline{(x+y'+z')} . \underline{(x'+y'+z')} \\ &= M2.M3.M7 = \underline{\underline{\prod (2,3,7)}} \Rightarrow \text{Product of Maxterm for } f \\ &\qquad \qquad \qquad m_j' = M_j \end{aligned}$$

Express the Boolean function  $F(a,b,c) = ab' + c'$  using Sum of minterms and Product of maxterms

$$\overline{c + \bar{c}} = 1$$

- Two methods
  1. Identify the missing term and include them in the expression using the postulates :  $x+0=x$  ,  $x.1=x$  ,  $x+x'=1$ ,  $x.x'=0$
  2. Write the truth table from the given expression and then write sum of minterms and product of maxterms

$$F(a,b,c) = \underline{ab'} + \underline{c'} \text{ using method 1}$$

$$= a\bar{b} \cdot 1 + \bar{c} \cdot 1$$

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

$$= \underbrace{a\bar{b}c + a\bar{b}\bar{c}}_{,,} + a\bar{c} + \bar{a}\bar{c}$$

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

$$= m_4 + m_5 + m_1 + m_0$$

$$= \sum_m (0, 2, 4, 5, 6) \Rightarrow \bar{f}(a, b, c) = \sum_m (1, 3, 7)$$

$$F(a, b, c) = \text{KM}(1, 3, 7) \Rightarrow \bar{f}(a, b, c) = \text{KM}(0, 2, 4, 5, 6)$$

$F(a,b,c) = \overline{ab'} + c'$  using method 2

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

$$f(a,b,c) = \sum m(0, 2, 4, 5, 6) \\ = \prod M(1, 3, 7)$$

Design a combinational circuit that takes 3-bit input and generates an output high whenever the input is a prime number.

Draw the circuit using basic logic gates

$$F(A, B, C)$$

① Truth table

②  $F = \sum m(1, 2, 3, 5, 7) \text{ or } \prod M(0, 4, 6)$

③ Simplified expression for  $F_2$

④ Draw the ext

①	A	B	C	F
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1

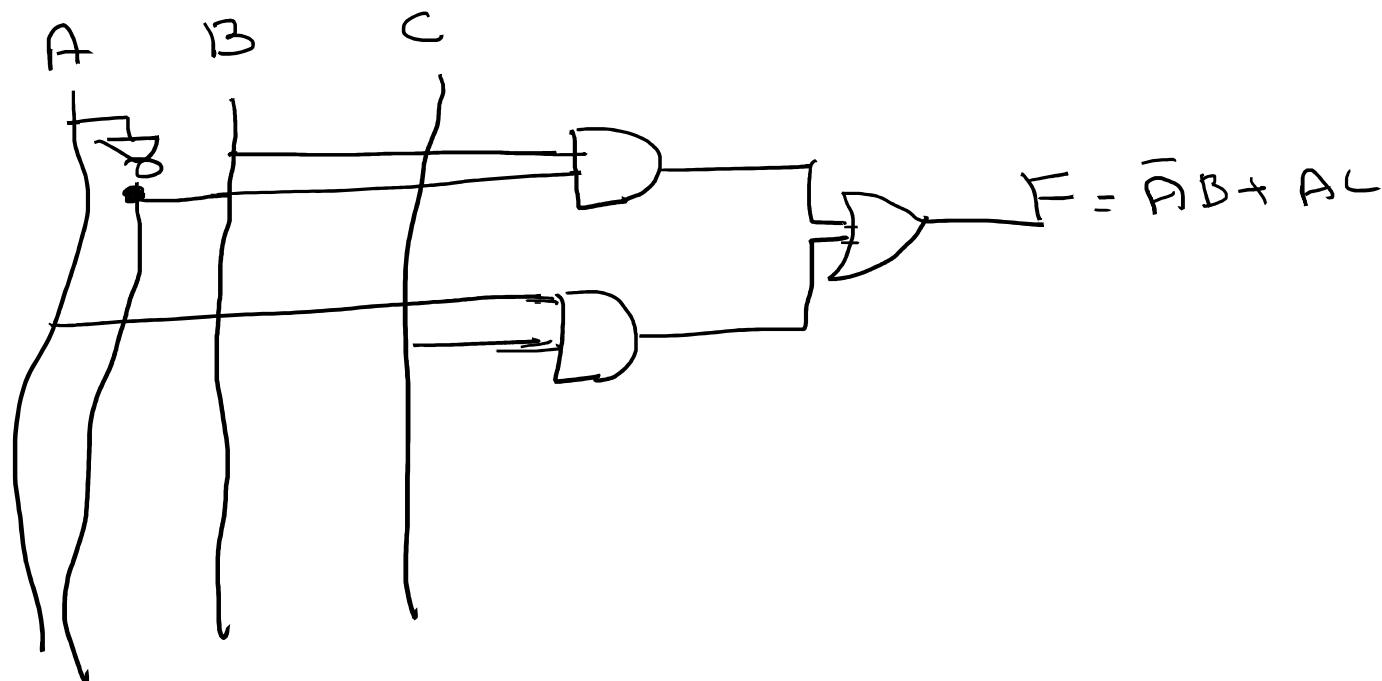
$$\overline{A} \overline{B} \overline{C} + \overline{A} B C$$

$\overline{ABC}$ )

$$\begin{aligned} F(A, B, C) &= \sum m(2, 3, 5, 7) \\ &= \overline{AB} + BC \end{aligned}$$

## Rough

$$F(A, B, C) = \bar{A}B + AC$$



Design a combinational circuit that takes 2-bit input and outputs the square of the input

①

	$A_1, A_0$	$y_3 \ y_2 \ y_1 \ y_0$	MSB	LSB
0	0 0	0 0 0 0		
1	0 1	0 0 0 1		1
2	1 0	0 1 0 0	4	
$\rightarrow 3$	1 1	1 0 0 1	9	

②

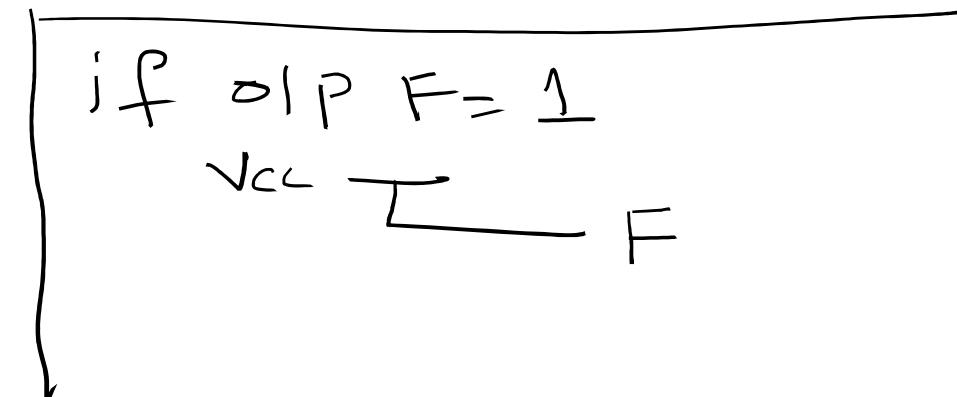
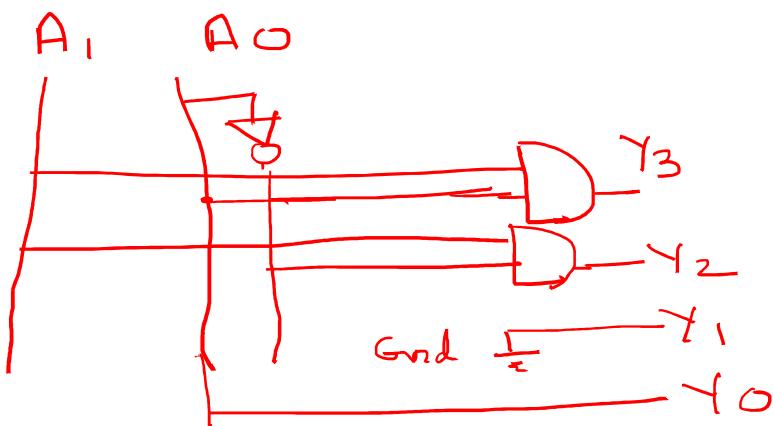
$$y_3 = A_1 \cdot A_0 = \Sigma m(3)$$

$$y_2 = A_1 \cdot \overline{A}_0 = \Sigma m(2)$$

$$y_1 = 0$$

$$\begin{aligned} y_0 &= \overline{A}_1 \cdot \overline{A}_0 + A_1 \cdot \overline{A}_0 = \Sigma m(1, 3) \\ &= A_0 \end{aligned}$$

③



**Rough**

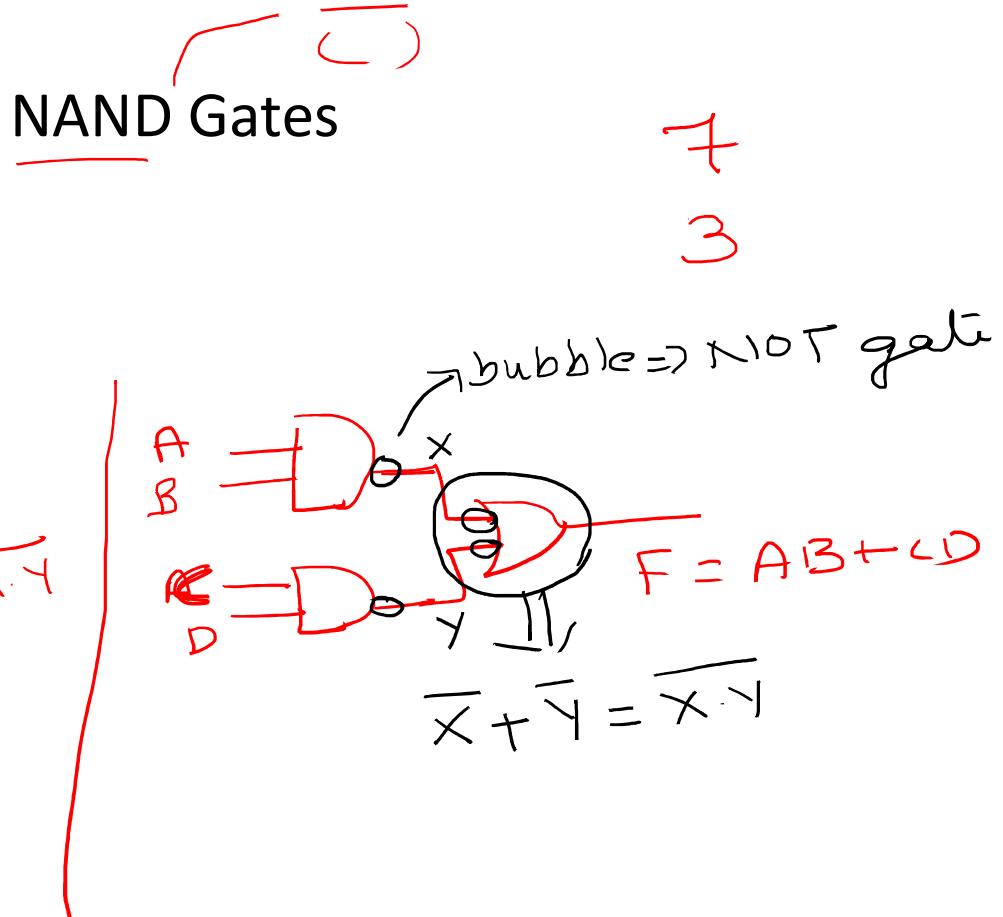
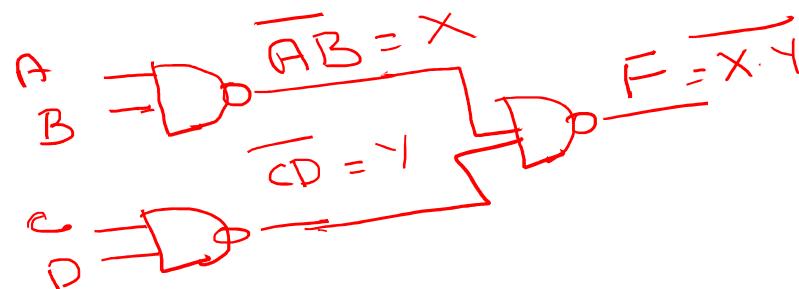
## Drawing the circuit using only universal gates

- 1.  $F(A,B,C,D) = AB + CD$  using only NAND Gates

$$\bar{F} = \overline{\overline{AB} + \overline{CD}} = \overline{\overline{AB}} \cdot \overline{\overline{CD}}$$

$$\bar{F} = \overline{(AB) \cdot (\overline{CD})} = F$$

$X$        $Y$

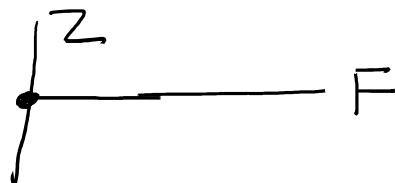


$$A + \bar{A}B = A + B$$

2.  $F(x,y,z) = xz + y'z + x'y'z$  using only NAND gates

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

$$F = z$$



$$= \cancel{xz} +$$

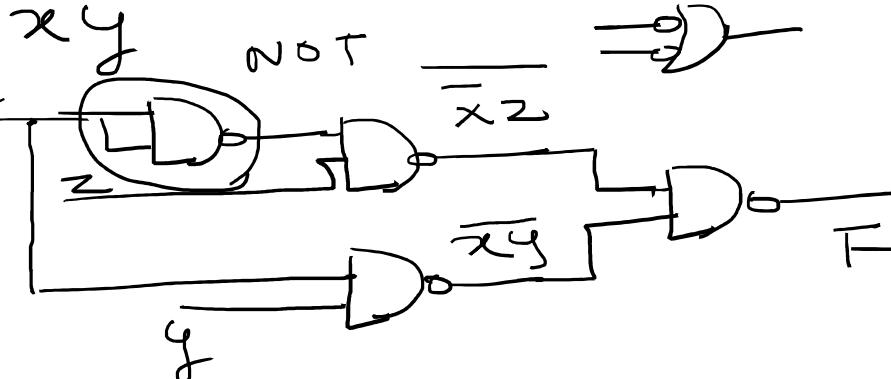
$$= z(x + \bar{y} + \bar{x}) = z(1 + \bar{y}) = z$$

⑤

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

$$\bar{F} = \overline{\bar{x}z \cdot \bar{xy}}$$

$$F = \overline{\bar{F}} = \overline{\bar{x}z \cdot \bar{xy}}$$



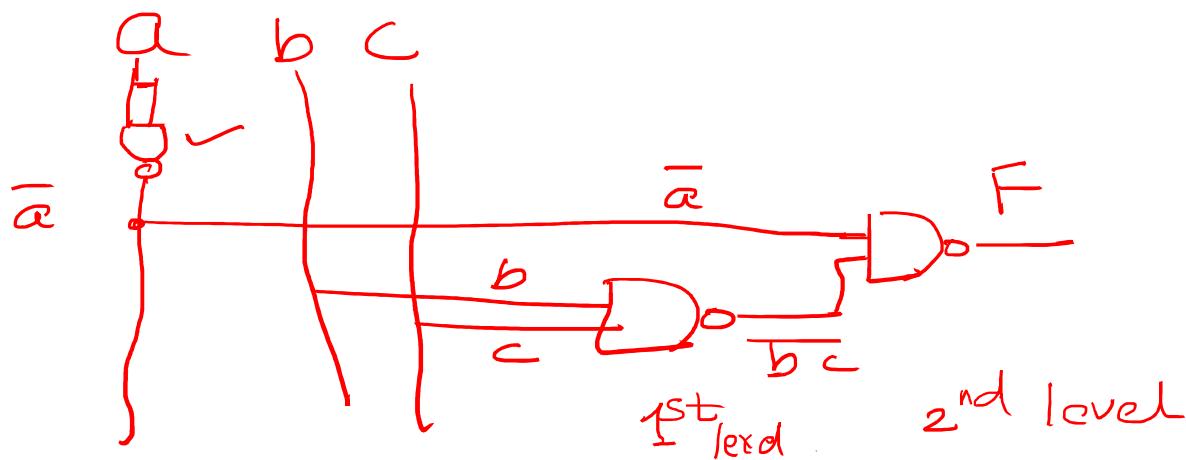
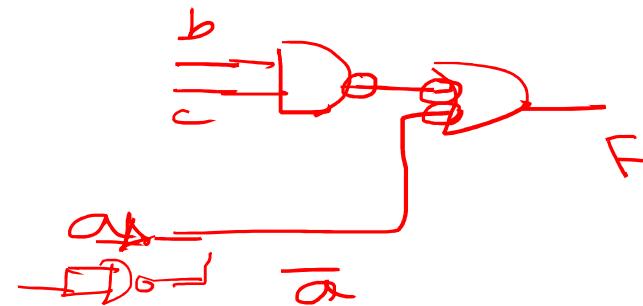
3. Realize the simplified  $F(x,y,z) = \Sigma (1,2,3,4,5,7)$  using only NAND gates

-

4.  $F(a,b,c) = \underline{\underline{a}} + bc$  using only NAND gates

- $\bar{F} = \bar{a} \cdot \bar{bc}$

$$\bar{F} = \overline{\bar{a} \cdot \bar{bc}}$$

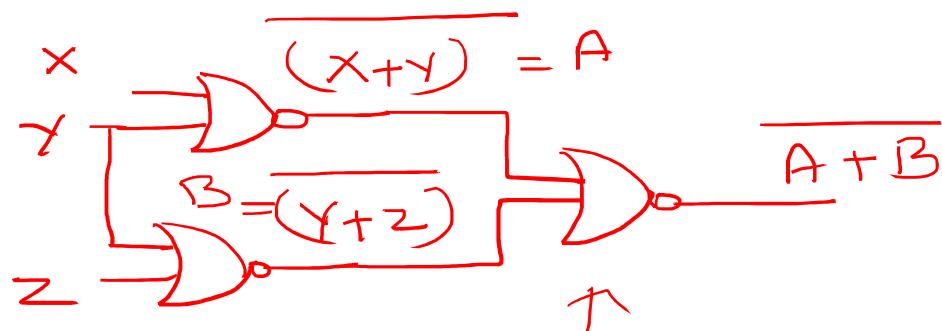


## Drawing the circuits using only NOR gates

- 1.  $F(x,y,z) = \underline{\underline{(x+y)}} \cdot \underline{\underline{(y+z)}}$

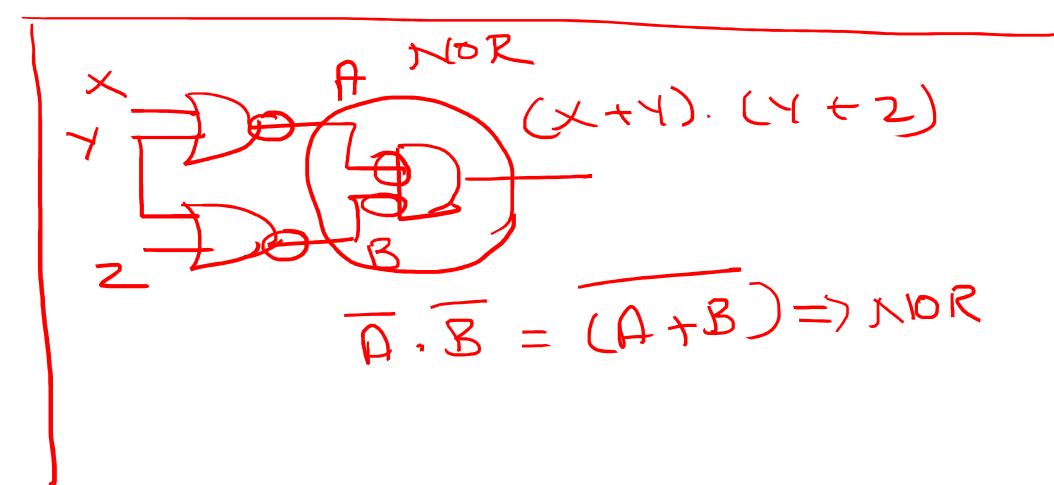
$$\overline{F} = \overline{\underline{\underline{(x+y)}} + \underline{\underline{(y+z)}}}$$

$$\overline{\overline{F}} = \overline{\overline{\underline{\underline{(x+y)}}} + \overline{\underline{\underline{(y+z)}}}}$$

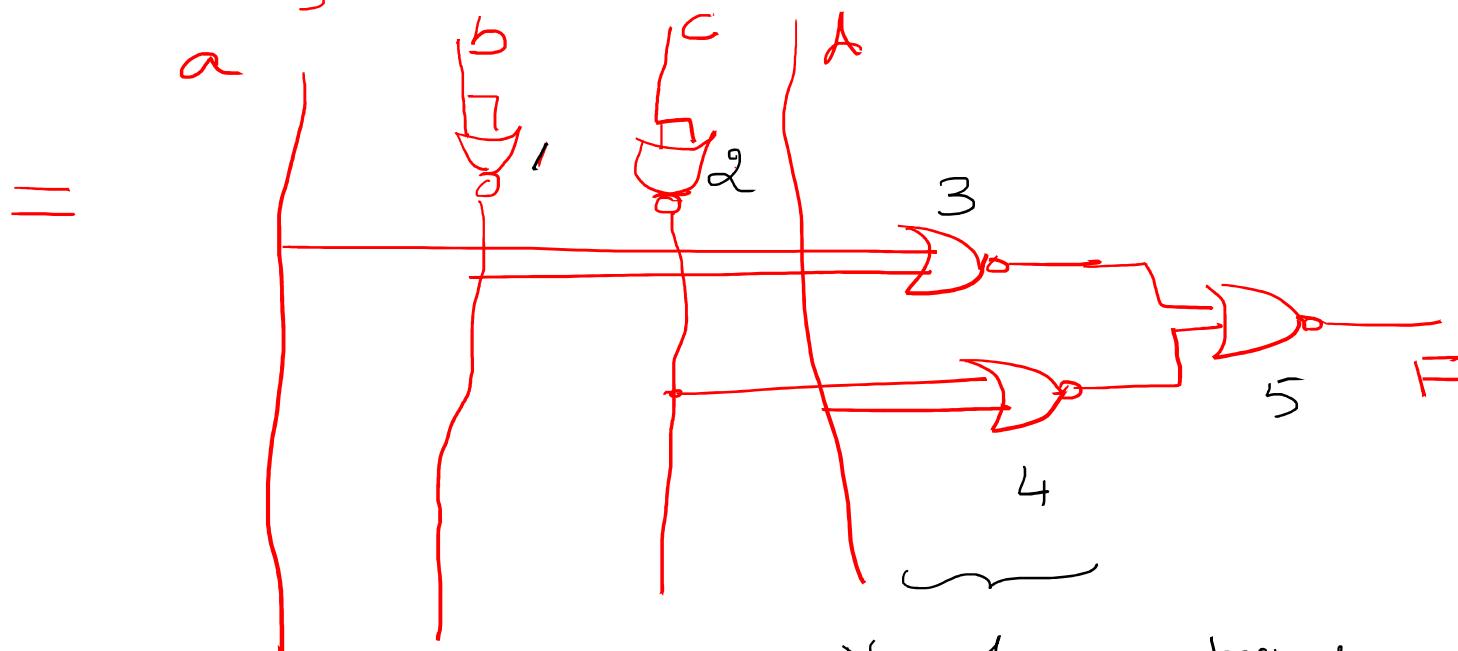


NOT using NOR

$$\begin{aligned} F &= \overline{A + B} \\ &= \overline{\overline{A} + \overline{B}} = \overline{\overline{A}} \end{aligned}$$

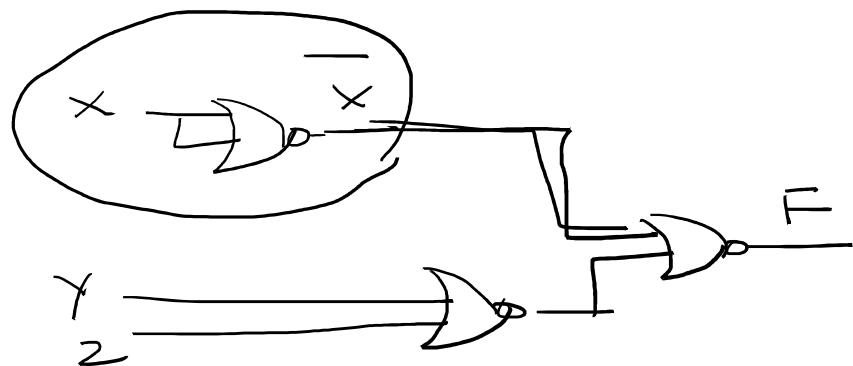


$$2. F(a, b, c, d) = \overbrace{(a+b')}^{\substack{3 \\ 1}} \overbrace{(c'+d)}^{\substack{4 \\ 2}}$$



No. of sum terms

$$3. f(x,y,z) = \underline{\underline{x}} \cdot \underline{\underline{(y+z)}} \Rightarrow f = \overline{\underline{x} + \underline{(y+z)}}$$



$$4. f(a, b, c, d) = a.b.(c+d)$$

$$= \overline{\underline{a}} + \overline{\underline{b}} + \overline{\underline{(c+d)}}^4$$

Any questions

# LECTURE 5 & 6

---

KARNAUGH MAP (K – MAP)

# K - MAP

# Karnaugh-map

- Pictorial form of a truth table.
- Graphical tool to simplify a logical equation by forming groups of cells.

One-variable

$$F = X \quad \text{or} \quad F(x)$$

$$F = \bar{x}$$

Two-variable

$$F(x,y) \quad \text{or} \quad F(A,B)$$

$$2^2 = 4 \text{ variations ilp}$$

$$\begin{array}{ll} \bar{x}\bar{y} & x+y \\ \bar{x}y & x+\bar{y} \\ x\bar{y} & \bar{x}+y \\ xy & \bar{x}+\bar{y} \end{array}$$

minterms      (maxterms)

$$SOP = \sum_{i=1}^3 (0, 1, 2) = \bar{x}\bar{y} + \underline{\bar{x}y + xy} =$$

$$POS = \prod_{i=1}^3 3 = (\bar{x}+\bar{y}) = \text{simplified}$$

Why minimization?

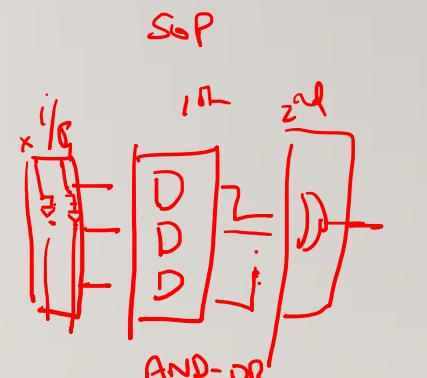
Cost minimization

Cost  $\propto$  Number of 1's per total gates  
+ Number of gates

<u>1<sup>st</sup> level</u>	<u>2<sup>nd</sup> level</u>
Not gate	AND

OR	= 6
----	-----

2	6	3	= 11
2	-	1 OR	= 3
2	-	2	= 4



Cost = 17 unit cost

Cost = 7 unit cost

# K - MAP

3 Variable function $f(x,y,z)$				$2^3 = 8$ combinations	
	<u>min terms</u>	<u>max terms</u>	$\sum$		
0	$\bar{x}\bar{y}\bar{z}$	000 m <sub>0</sub>	M <sub>0</sub>	$x+y+z$	0
1	$\bar{x}\bar{y}z$	001 m <sub>1</sub>	M <sub>1</sub>	$x+y+\bar{z}$	1
2	$\bar{x}yz$	010 m <sub>2</sub>	M <sub>2</sub>	$x+\bar{y}+z$	1
3	$\bar{x}y\bar{z}$	011 m <sub>3</sub>	M <sub>3</sub>	$x+\bar{y}+\bar{z}$	1
4	$x\bar{y}\bar{z}$	100 m <sub>4</sub>	M <sub>4</sub>	$\bar{x}+y+z$	0
5	$x\bar{y}z$	101 m <sub>5</sub>	M <sub>5</sub>	$\bar{x}+y+\bar{z}$	0
6	$xy\bar{z}$	110 m <sub>6</sub>	M <sub>6</sub>	$\bar{x}+\bar{y}+z$	1
7	$xy\bar{z}$	111 m <sub>7</sub>	M <sub>7</sub>	$\bar{x}+\bar{y}+\bar{z}$	0

$m_i$

$M_i$

Boolean law's

Any function involving 3 variables

SOP

$$f(x,y,z) = \sum (1, 2, 3, 6) = \bar{x}\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}y\bar{z} + xy\bar{z}$$

canonical

POS

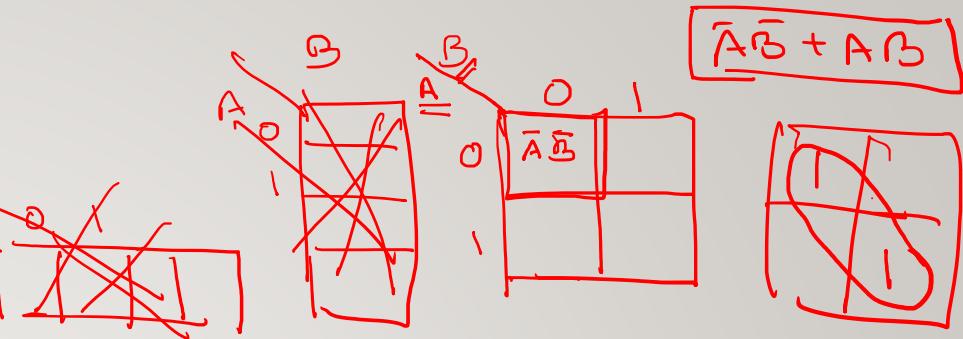
$$f(x,y,z) = \prod (0, 4, 5, 7) = (x+y+z)(\bar{x}+y+z)(\bar{x}+y+\bar{z})(\bar{x}+\bar{y}+\bar{z})$$

=====

$$\boxed{M_i = \overline{m_i}}$$

## TWO VARIABLE K - MAP

$2^2 = 4$  combination  $\Rightarrow$  4 cells

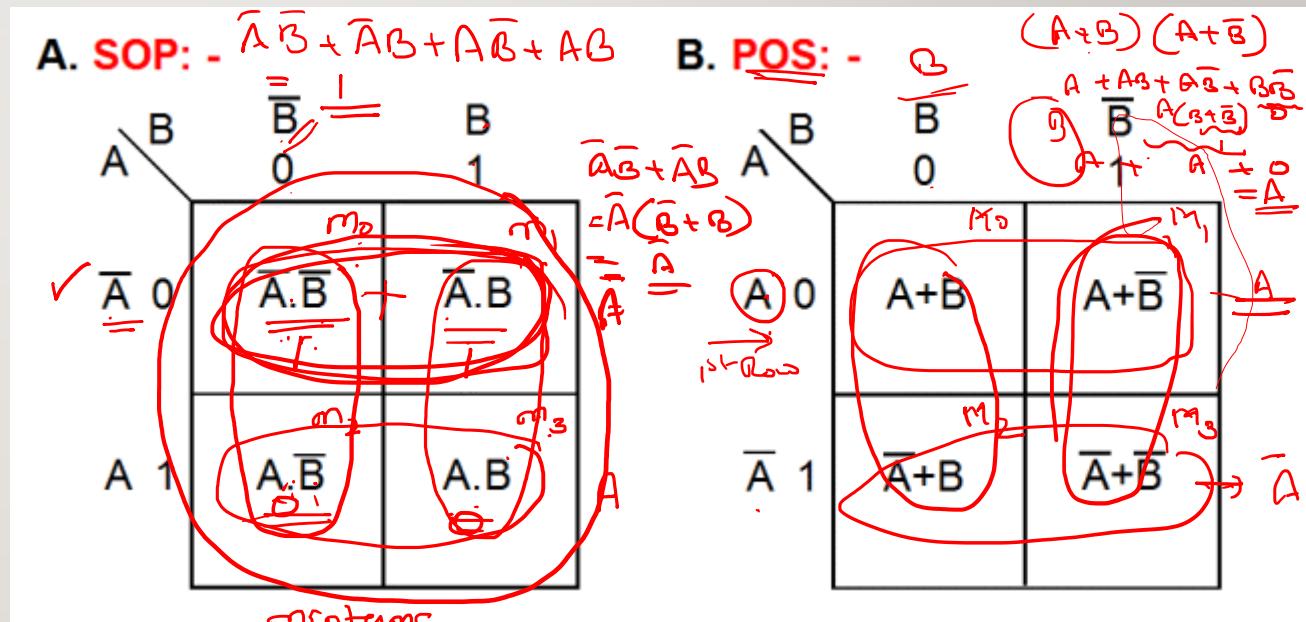


A	B	SOP	POS
0	0	$\bar{A}\bar{B}$	$A + B$
0	1	$\bar{A}B$	$A + \bar{B}$
1	0	$A\bar{B}$	$\bar{A} + B$
1	1	$AB$	$\bar{A} + \bar{B}$

$$F(AB) = \bar{A}\bar{B} + \bar{B}A = \bar{A}\bar{B}$$

$$\bar{A}\bar{B} + A\bar{B} = \cancel{\bar{A}\bar{B}} + \cancel{A\bar{B}} = \underline{\underline{0}}$$

$$\therefore F(AB) = \underline{\underline{0}}$$



$$F(A, B) = \sum_{m_0, m_1, m_3} 1 = \sum_{m_0, m_1, m_3} 1 = F(A, B)$$

$$F(A, B) = \sum_{m_0, m_1, m_3} 1 = \sum_{m_0, m_1, m_3} 1 = F(A, B)$$

$$F = \bar{A}\bar{B} + A\bar{B}$$

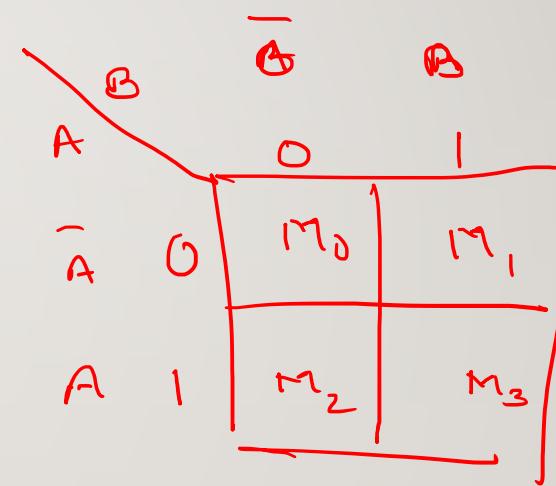
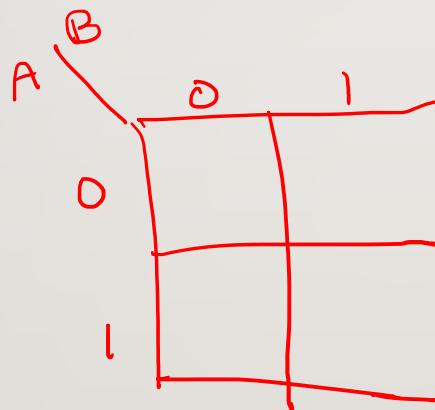
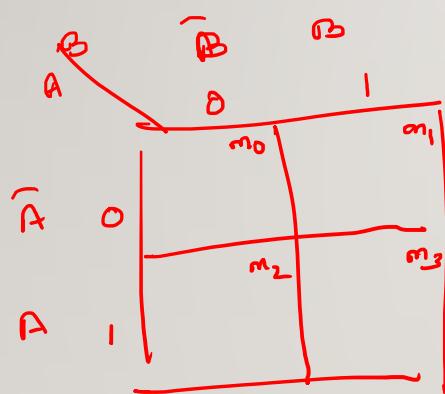
$$F = B\bar{A} + A\bar{B}$$

$$F(A, B) = \sum_{m_0, m_1, m_3} 1 = F(A, B)$$

## TWO VARIABLE K – MAP

---

$f(x, y) \Rightarrow 4$  variation

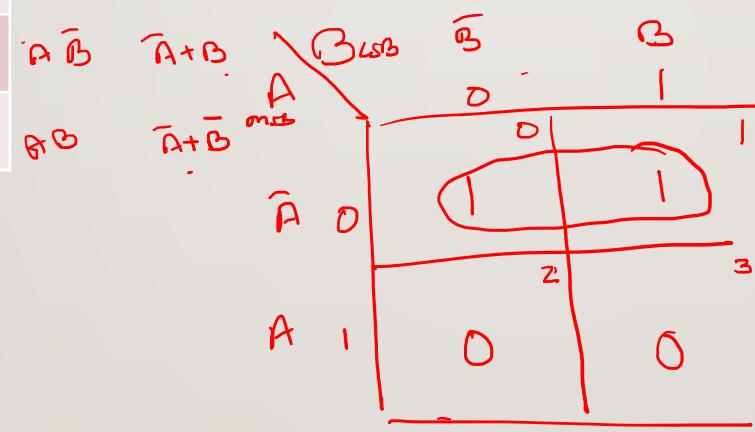


**EXAMPLE I:** Solve using K-map, Draw simplified Circuit  
 Give SOP, POS & simplified Expression

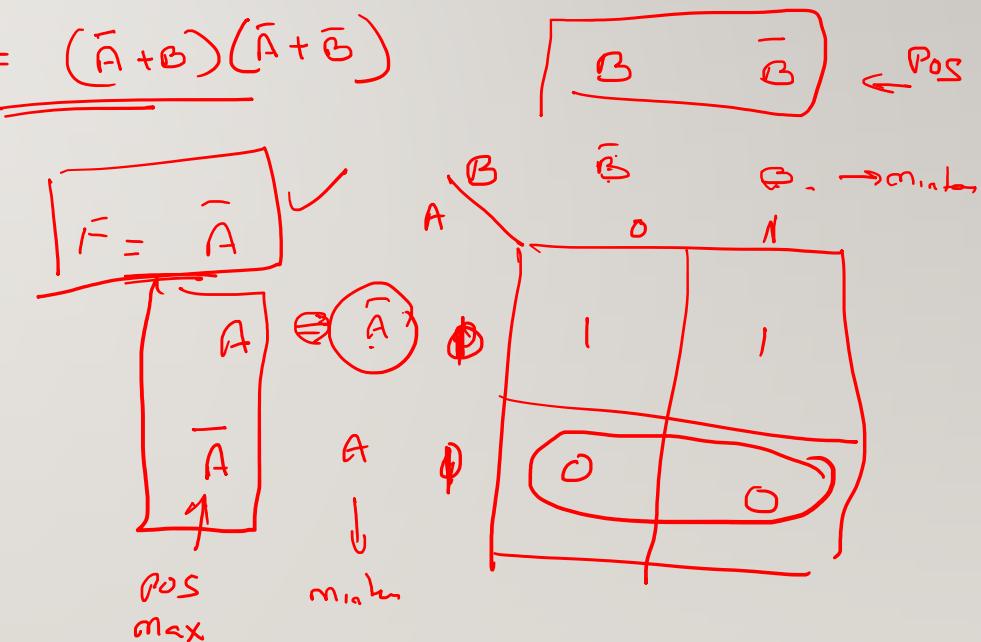
A	B	F
0	0	1
0	1	1
1	0	0
1	1	0

$$F(A, B) \cdot \text{SOP} = \sum(0, 1) = \overline{A}\overline{B} + \overline{A}B$$

$$\text{POS} = \prod_m(2, 3) = (\overline{A}+B)(\overline{A}+\overline{B})$$



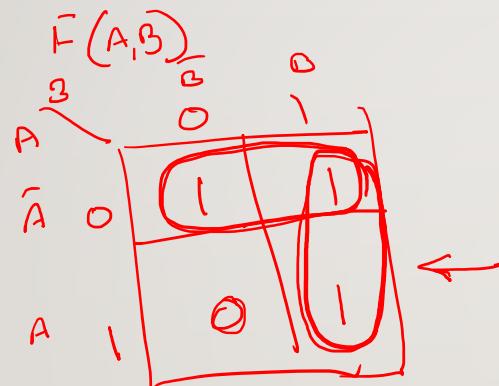
SOP



POS

## EXAMPLE 2:

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



SOP

$$F = \bar{A} + AB \xrightarrow{X} \bar{A} + B$$

$$F = \underline{\bar{A} + B} \quad \checkmark$$

Combination Rule

Note: K-map

\* To combine the max number of 1's SOP  
0's POS

$$\begin{matrix} 2^0 & 2^1 & 2^2 & 2^3 & 2^4 & 2^5 \\ 1 & 2 & 4 & 8 & 16 & 32 \end{matrix} \dots \dots \dots$$

1 2 4 8 16 32 to eliminate variables

$$\begin{aligned} F &= \bar{A}\bar{B} + \underline{\bar{A}B} + \underline{AB} \\ &= (\bar{A}\bar{B} + \bar{A}B) + (\bar{A}B + AB) \\ &= \bar{A}(\bar{B} + B) + (\bar{A} + A)B \\ \boxed{F = \bar{A} + B} \end{aligned}$$

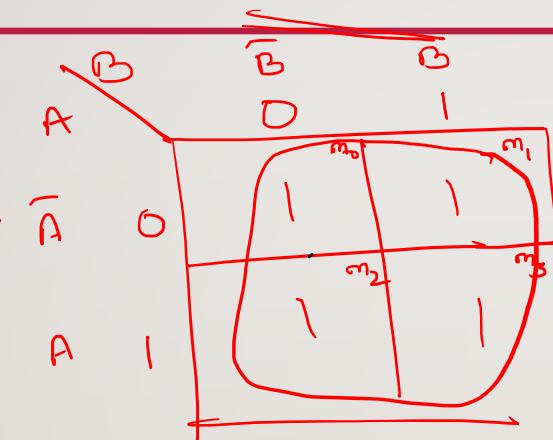
$$x+x+x \dots \dots \xrightarrow{=} x$$

$$12 = 1100 = \bar{f}(A B C D) = A B \bar{C} \bar{D}$$

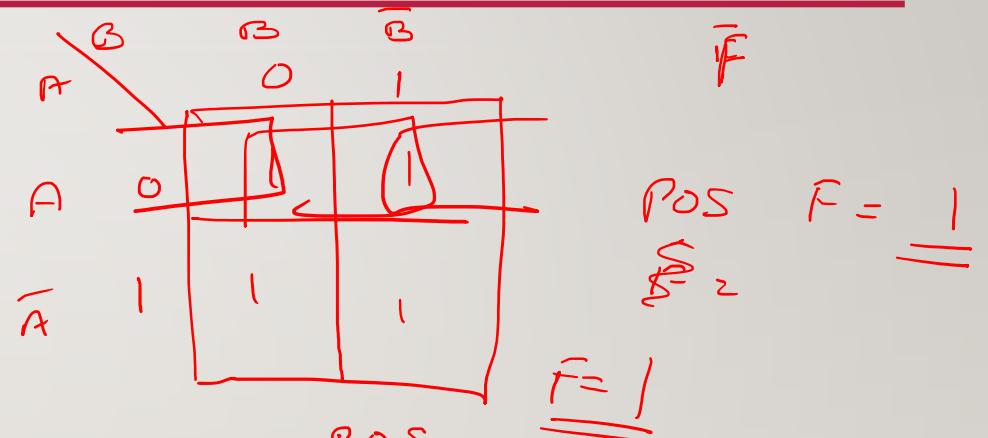
## EXAMPLE 3:

Solve using K-map

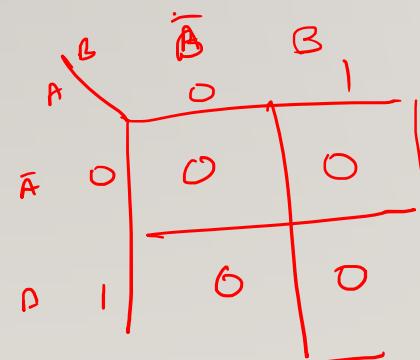
A	B	F
0	0	1
0	1	1
1	0	1
1	1	1



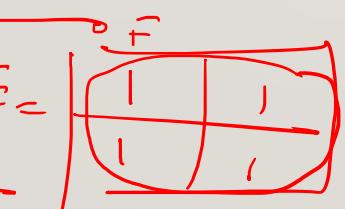
SOP



$$\text{POS} \Rightarrow F = 1$$



$$\text{Hg Lc} \rightarrow T$$



$$\cdot F = P_c \oplus \bar{F} = \bar{P}_c$$

$$\begin{aligned}
 F &= \text{SOP} = \sum_m (m_0, m_1, m_2, m_3) \\
 F &= \text{POS} = \prod_{\text{m}} (m_0, m_1, m_2, m_3)
 \end{aligned}$$

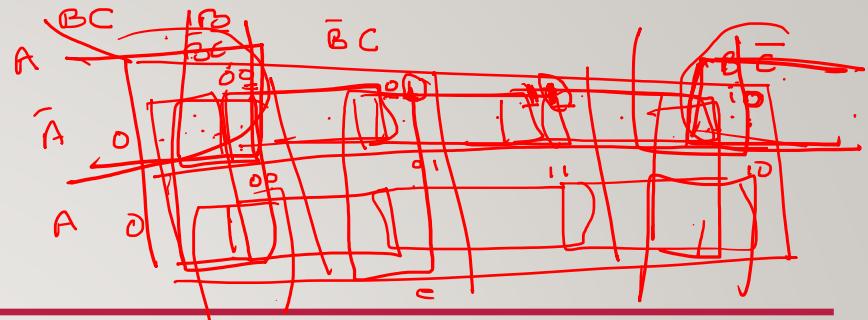
# THREE VARIABLE K – MAP

Variables = 3

Combinations = 8

$\begin{matrix} BC \\ 00 \\ 01 \\ 10 \\ 11 \end{matrix}$

$$\bar{A}\bar{B} + A\bar{B} \quad \bar{A}B + A\bar{B}$$



A	B	C	SOP	POS
0	0	0	$m_0 \bar{A}\bar{B}\bar{C}$	$A + B + C \text{ } m_0$
0	0	1	$m_1 \bar{A}\bar{B}C$	$A + B + \bar{C} \text{ } m_1$
0	1	0	$m_2 A\bar{B}\bar{C}$	$A + \bar{B} + C \text{ } m_2$
0	1	1	$m_3 A\bar{B}C$	$A + \bar{B} + \bar{C}$
1	0	0	$m_4 A\bar{B}\bar{C}$	$\bar{A} + B + C$
1	0	1	$m_5 A\bar{B}C$	$\bar{A} + B + \bar{C}$
1	1	0	$m_6 A\bar{B}\bar{C}$	$\bar{A} + \bar{B} + C$
1	1	1	$m_7 ABC$	$\bar{A} + \bar{B} + \bar{C} \text{ } m_7$

$F(A B C)$

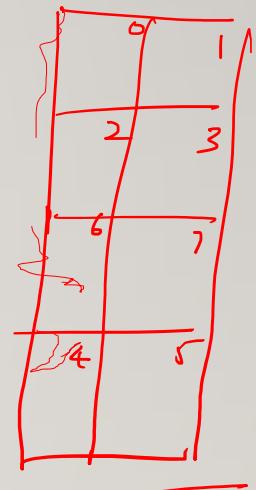
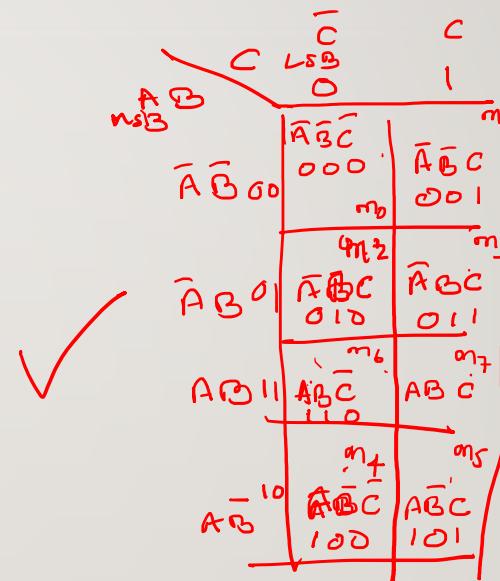
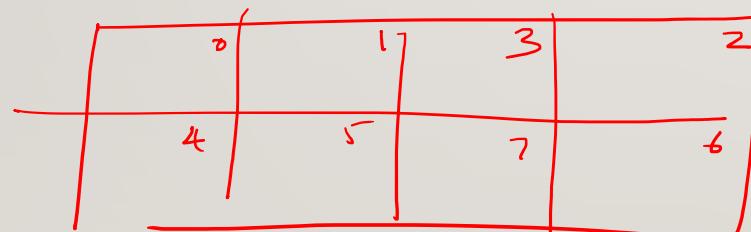
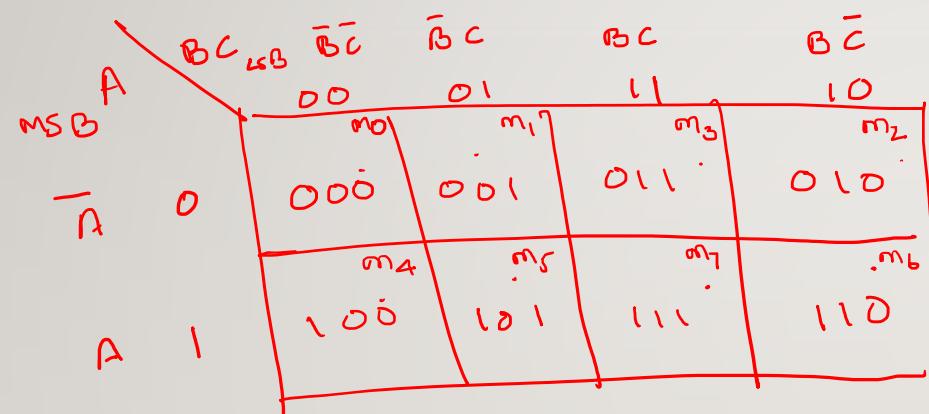
A. SOP: -

	$\bar{B} \bar{C}$	$\bar{B} C$	$B \bar{C}$	$B C$
$\bar{A}$	$\bar{A} \bar{B} \bar{C}$ 0	$\bar{A} B C$ 1	$A \bar{B} C$ 3	$A B \bar{C}$ 2
$A$	$A \bar{B} \bar{C}$ 4	$A \bar{B} C$ 5	$A B C$ 7	$A B \bar{C}$ 6

B. POS: -

	$B+C$	$B+\bar{C}$	$\bar{B}+\bar{C}$	$\bar{B}+C$
$\bar{A}$	$A+B+C$ 0	$A+B+\bar{C}$ 1	$A+\bar{B}+\bar{C}$ 3	$A+\bar{B}+C$ 2
$A$	$\bar{A}+B+C$ 4	$\bar{A}+B+\bar{C}$ 5	$\bar{A}+\bar{B}+\bar{C}$ 7	$\bar{A}+\bar{B}+C$ 6

# 3-VARIABLE K-MAPS



# EXAMPLE 1:

Given the Boolean function:

$$F = \overline{\underline{A}}\overline{\underline{C}} + \overline{\underline{A}}\underline{\underline{B}} + \underline{\underline{A}}\overline{\underline{B}}\overline{\underline{C}} + \underline{\underline{B}}\underline{\underline{C}}$$

3 Variable =  $2^3 = 8$

---

- Express it in Sum of minterms form.
- Find the minimal sum of products expression using k-map.

$$F = \overline{\underline{A}}\overline{\underline{C}} + \overline{\underline{A}}\underline{\underline{B}} + \underline{\underline{A}}\overline{\underline{B}}\overline{\underline{C}} + \underline{\underline{B}}\underline{\underline{C}}$$

$$\begin{aligned} F(A, B, C) &= \overline{\underline{A}}(\overline{\underline{B}} + \underline{\underline{B}})C + \overline{\underline{A}}\overline{\underline{B}}(\overline{\underline{C}} + \underline{\underline{C}}) + \underline{\underline{A}}\overline{\underline{B}}\overline{\underline{C}} + (\overline{\underline{A}} + \underline{\underline{A}})\underline{\underline{B}}\underline{\underline{C}} \\ &= \overline{\underline{A}}\overline{\underline{B}}\overline{\underline{C}} + \overline{\underline{A}}\overline{\underline{B}}\underline{\underline{C}} + \overline{\underline{A}}\underline{\underline{B}}\overline{\underline{C}} + \overline{\underline{A}}\underline{\underline{B}}\underline{\underline{C}} + \underline{\underline{A}}\overline{\underline{B}}\overline{\underline{C}} + \underline{\underline{A}}\overline{\underline{B}}\underline{\underline{C}} + \underline{\underline{A}}\underline{\underline{B}}\overline{\underline{C}} \\ &= \underline{\underline{m}_1} + \underline{\underline{m}_3} + \underline{\underline{m}_2} + \underline{\underline{m}_5} + \underline{\underline{m}_5} + \underline{\underline{m}_3} + \underline{\underline{m}_7} \end{aligned}$$

$$F = \sum_m (1, 2, 3, 5, 7) \quad SOP \quad \bar{F} \Rightarrow SOP = \sum_m (0, 4, 6)$$

$$F = \prod_m (0, 4, 6) \quad POS \quad \bar{F} \quad POS = \prod_m (1, 2, 3, 5, 7)$$

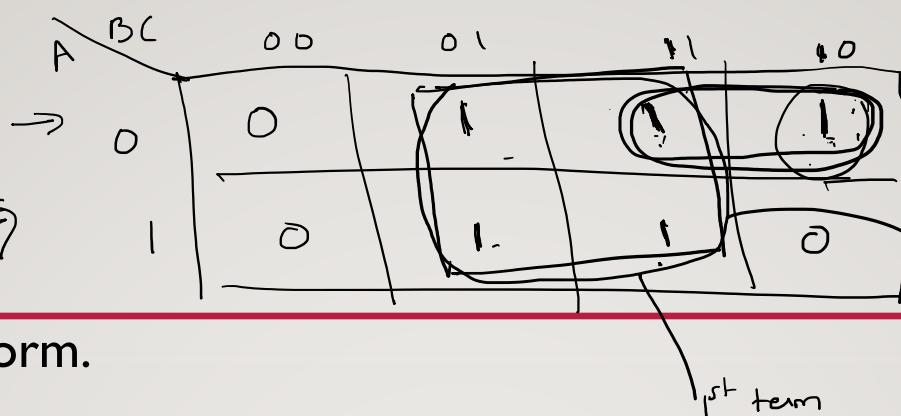
A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

# EXAMPLE 1:

Given the Boolean function:

$$F = \overline{AC} + \overline{AB} + A\overline{BC} + BC$$

- Express it in Sum of minterms form.
- Find the minimal sum of products expression using k-map.

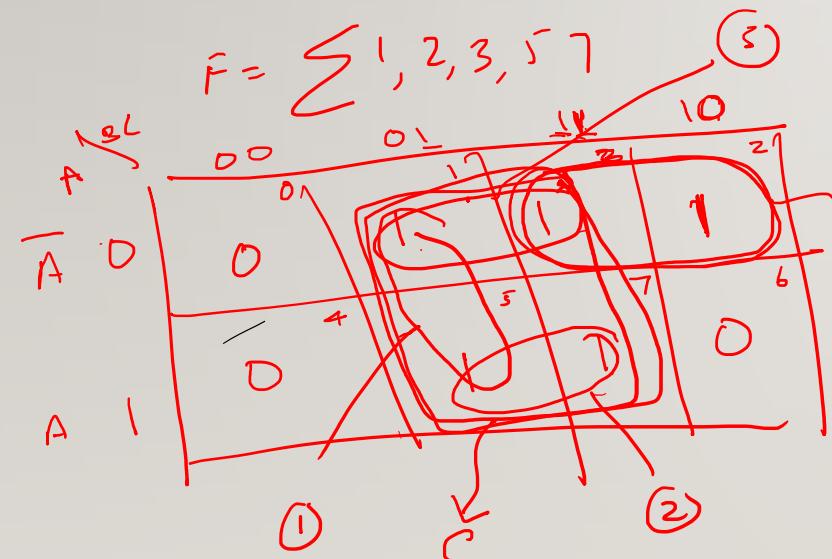


$$\begin{aligned}
 & \overline{ABC} + \overline{A}BC + A\overline{BC} + BC \\
 & \Rightarrow \overline{AC} + AC \\
 & = (\overline{A} + A)C = C \\
 & C + A'BC' \\
 & \overline{ABC} + \overline{A}B\overline{C}
 \end{aligned}$$

$$\begin{aligned}
 & C + A'BC \\
 & \text{Grouping Rule} \\
 & \overline{A}B(C + \overline{C}) \\
 & = \overline{AB}
 \end{aligned}$$

(1) Try combining max no. of Cells

(2) Try to cover unmarked 1's

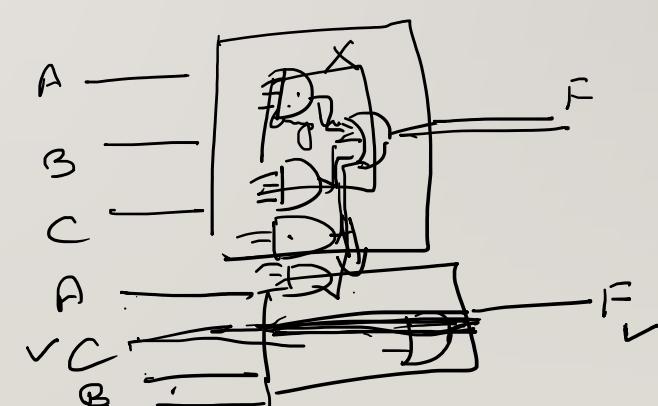


$$\begin{aligned}
 & X + \overline{X}Y = X + Y \\
 & F = C + \overline{A}B + \underbrace{(① + ②) + ③}_{\text{Redundant}}
 \end{aligned}$$

$$\cancel{F = C + \overline{A}B}$$

$$C + \overline{A}B\overline{C}$$

$$\begin{aligned}
 & C + (C \overline{B})\overline{C} \\
 & = \underline{\underline{C + C}} \overline{B}
 \end{aligned}$$

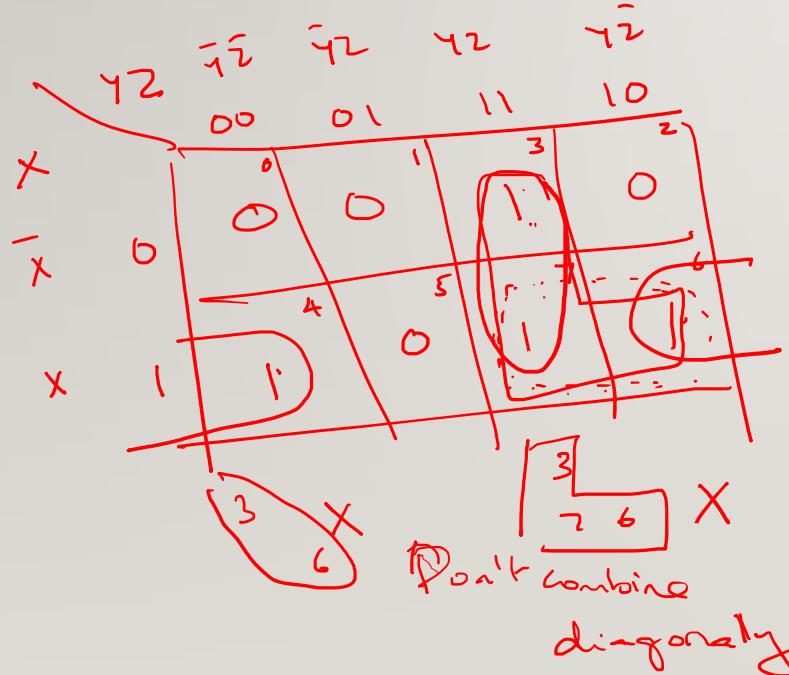


## EXAMPLE 2:

Simplify the Boolean expression:

$$F(x, y, z) = \sum(3, 4, 6, 7)$$

SOP  $F = \sum(3, 4, 6, 7)$



$$\begin{aligned} F &= \cancel{y_2 + x\bar{z}} + (\quad) \\ &\quad \text{Redundant} \\ &= \cancel{\bar{x}y_2 + x\bar{y}_2} \\ &\quad \text{---} \\ &= \cancel{(x+y)\bar{y}_2} \\ &= \cancel{\bar{x}y_2 + \cancel{x\bar{y}_2}} \\ &\quad \text{---} \\ &= \cancel{y_2(\bar{x}z + x\bar{z})} \\ &\quad \text{---} \\ &= \boxed{?} \end{aligned}$$

Annotations:  $\cancel{y_2 + x\bar{z}}$  is crossed out with a large red line.  $\cancel{\bar{x}y_2 + x\bar{y}_2}$  is crossed out with a large red line.  $\cancel{(x+y)\bar{y}_2}$  is crossed out with a large red line.  $\cancel{\bar{x}y_2 + \cancel{x\bar{y}_2}}$  is crossed out with a large red line.  $\cancel{y_2(\bar{x}z + x\bar{z})}$  is crossed out with a large red line. The question mark  $\boxed{?}$  is enclosed in a red bracket.

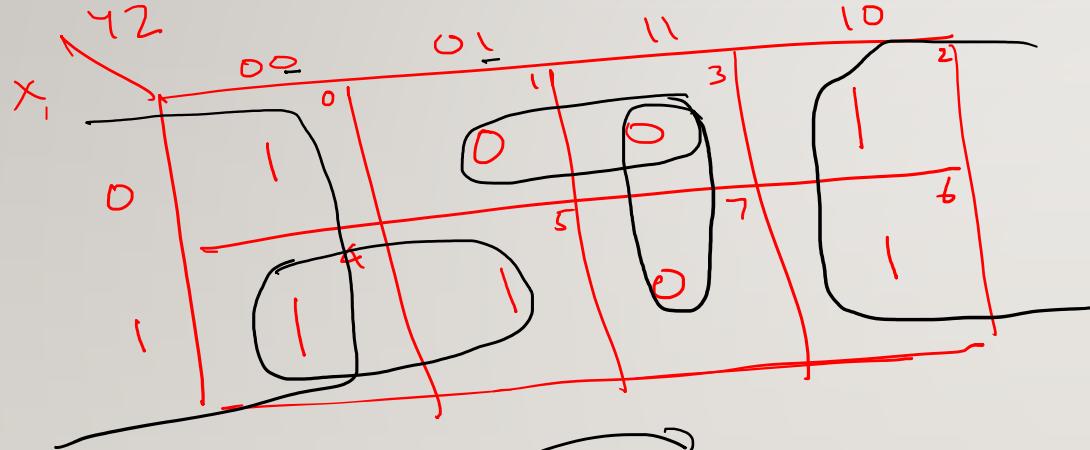
$\stackrel{2}{L}$   $\stackrel{3}{L}$

$\stackrel{2}{L}$   $\stackrel{3}{L}$  p AND  
 $\stackrel{2}{L}$   $\stackrel{3}{L}$  p OR

$\stackrel{2}{L}$   $\stackrel{3}{L}$  p AND  
 $\stackrel{2}{L}$   $\stackrel{3}{L}$  p OR  
 $\stackrel{2}{L}$   $\stackrel{3}{L}$  b AND

## EXAMPLE 3:

$$F(x, y, z) = \sum (0, 2, 4, 5, 6)$$



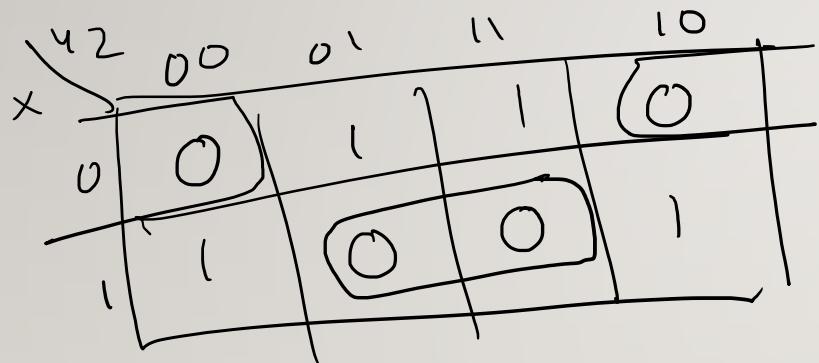
$$\begin{array}{l}
 F = \overline{z} + x\bar{y} \\
 \Leftrightarrow \text{POS} = (\bar{x} + \bar{z})(\bar{y} + \bar{z}) \\
 \Leftrightarrow \text{SOP} = (\bar{z} + x)(\bar{z} + \bar{y})
 \end{array}$$

$m_1$        $m_3$   
 $(\bar{x} + \bar{y} + \bar{z})(x + \bar{y} + \bar{z})$   
 $\bar{y}\bar{z}$

$$\left| \begin{array}{l}
 a \cdot (b + c) = \underline{a \cdot b} + a \cdot c \\
 a + bc = (a + b)(a + c)
 \end{array} \right.$$

## EXAMPLE 4:

$$F(x, y, z) = \prod (0, 2, 5, 7)$$



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

# LECTURE 5 & 6

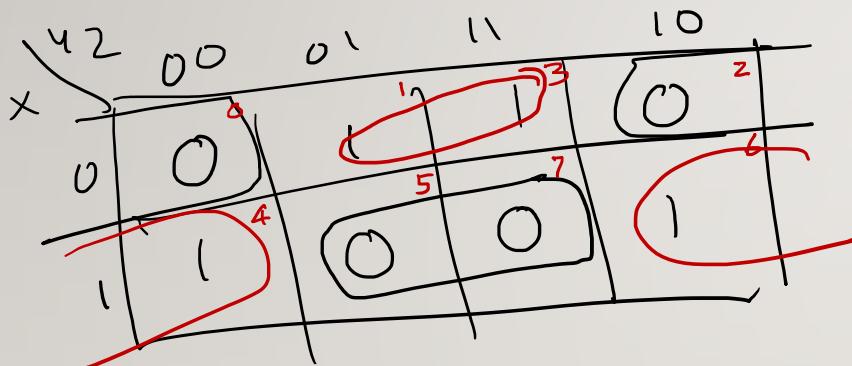
---

KARNAUGH MAP (K – MAP)

## EXAMPLE 4:

$$F(x, y, z) = \prod_{m} (0, 2, 5, 7) = \sum_{S} (1, 3, 4, 6)$$


---



POS

$$\overline{F} = (x + z)(\bar{x} + \bar{z})$$

SOP

$$\overline{F} = \overline{xz} + x\bar{z}$$

NAND

$$\begin{aligned}\overline{\overline{F}} &= \overline{\overline{xz} + \bar{x}\bar{z}} = \overline{F} \\ &= \overline{\overline{xz}} \cdot \overline{\overline{\bar{x}\bar{z}}} = \overline{F}\end{aligned}$$

Draw ckt here  
NAND ONLY

NOR

$$\begin{aligned}F &= \overline{\overline{F}} = \overline{(x+z)(\bar{x}+\bar{z})} \\ &= \overline{\overline{(x+z)} + \overline{(\bar{x}+\bar{z})}} = \overline{\overline{x+z} \cdot \overline{\bar{x}+\bar{z}}}\end{aligned}$$

NOR ONLY

# FOUR VARIABLE K – MAP

4 Variables  $2^4 = 16$  combinations

SOP using minterms

A. SOP:-

	$\bar{C} \bar{D}$	$\bar{C} D$	$C \bar{D}$	$C D$	
$\bar{A} \bar{B}$	$m_0$	$m_1$	$m_3$	$m_2$	
$\bar{A} B$	$m_4$	$m_5$	$m_7$	$m_6$	
$A \bar{B}$	$m_{12}$	$m_{13}$	$m_{15}$	$m_{14}$	
$A B$	$m_8$	$m_9$	$m_{11}$	$m_{10}$	

$\rightarrow$  ABCD       $\rightarrow$  0000  
 $\rightarrow$  0001  
 $\rightarrow$  0010  
 $\rightarrow$  0011  
 $\rightarrow$  0100  
 $\rightarrow$  0101  
 $\rightarrow$  0110  
 $\rightarrow$  0111  
 $\rightarrow$  1000  $m_8$   
 $\rightarrow$  1001  
 $\rightarrow$  1010  
 $\rightarrow$  1011  
 $\rightarrow$  1100  
 $\rightarrow$  1101  
 $\rightarrow$  1110  
 $\rightarrow$  1111

$\bar{C} \bar{D}$	$\bar{C} D$	$C \bar{D}$	$C D$	$C \bar{D}$	$\rightarrow$ ABCD
$\bar{A} \bar{B}$	00	01	11	10	
$\bar{A} B$	0	1	3	2	
$A \bar{B}$	4	5	7	6	
$A B$	12	13	15	14	
$A \bar{B}$	8	9	11	10	

# FOUR VARIABLE K – MAP

---

POS:-		C+D	C+ $\bar{D}$	$\bar{C}+\bar{D}$	$\bar{C}+D$
A+B	$\bar{A}+\bar{B}$	0 0	0 1	1 1	1 0
$\bar{A}\bar{B}$ 0 0	A+B+C+D	A+B+C+ $\bar{D}$	A+B+ $\bar{C}+\bar{D}$	A+B+ $\bar{C}+D$	
	0	1	3	2	
$A+\bar{B}$ 0 1	A+ $\bar{B}$ +C+D	A+ $\bar{B}$ +C+ $\bar{D}$	A+ $\bar{B}$ + $\bar{C}+\bar{D}$	A+ $\bar{B}$ + $\bar{C}+D$	
	4	5	7	6	
$\bar{A}+\bar{B}$ 1 1	$\bar{A}+\bar{B}$ +C+D	$\bar{A}+\bar{B}$ +C+ $\bar{D}$	$\bar{A}+\bar{B}$ + $\bar{C}+\bar{D}$	$\bar{A}+\bar{B}$ + $\bar{C}+D$	
	12	13	15	14	
$\bar{A}+B$ 1 0	$\bar{A}+B$ +C+D	$\bar{A}+B$ +C+ $\bar{D}$	$\bar{A}+B$ + $\bar{C}+\bar{D}$	$\bar{A}+B$ + $\bar{C}+D$	
	8	9	11	10	

*AB' 5 cell*

AB		CD	$\bar{C}D$	$\bar{C}D$	$CD$	$CD$
		00	01	11	10	10
$\bar{A}B$	0	0	$\bar{A}\bar{B}\bar{C}\bar{D}$			
$\bar{A}B$	01	4	5	7	6	
$AB$	11	12	13	15	14	
$A\bar{B}$	10	8	9	11	10	

for SOP

AB		CD	$C+D$	$C+\bar{D}$	$\bar{C}+D$	$\bar{C}+\bar{D}$
		00	01	11	10	10
$A+B$	0	00	$M_0$			
$A+\bar{B}$	01		$A+B(C+D)$	$A+B(C+\bar{D})$		
$\bar{A}+\bar{B}$	11					
$\bar{A}+B$	10					

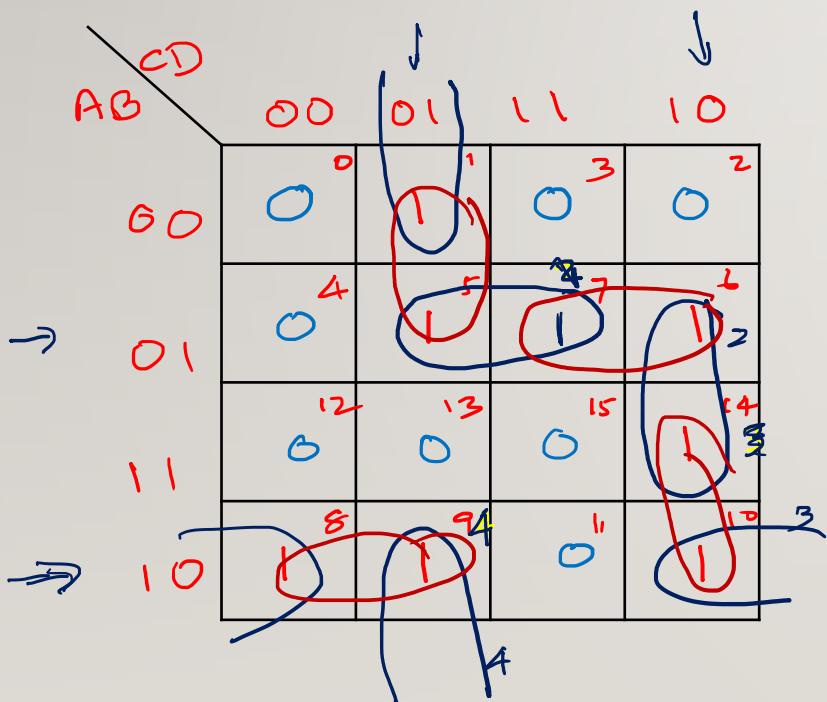
for POS

# EXAMPLE I:

- Simplify the following expression into

- ~~SOP & give NAND realization~~
- ~~POS & give NOR realization~~

$$F(A, B, C, D) = \sum_m (1, 5, 6, 7, 8, 9, 10, 14) = \prod_m (0, 2, 3, 4, 11, 12, 13, 15)$$



SOP    Group 1's

$$F = \bar{A}\bar{C}D + \bar{A}B\bar{C} + A\bar{C}\bar{D} + A\bar{B}\bar{C}\bar{D} \quad \checkmark$$

$$F = \bar{A}BD + BCD + A\bar{B}\bar{D} + \bar{B}\bar{C}D \quad \checkmark$$

$$\bar{A}B\bar{C}D + \bar{A}B\bar{C}D \Rightarrow \bar{A}B(\bar{C} + \bar{C})D = \underline{\bar{A}B\bar{D}}$$

# CONTINUED...

$$F(A,B,C,D) = \sum(1,5,6,7,8,9,10,14) = \bigwedge_m (0, 2, 3, 4, 11, 12, 13, 15)$$

- POS

POS

	$C+D$	$C+D^{\prime}$	$C^{\prime}+D^{\prime}$	$C^{\prime}+D$
$A+B^{\prime}$	00	01	11	10
$A+B$	00	1	0	0
$\bar{A}+\bar{B}$	11	1	1	1
$\bar{A}+\bar{B}^{\prime}$	10	1	1	1

POS

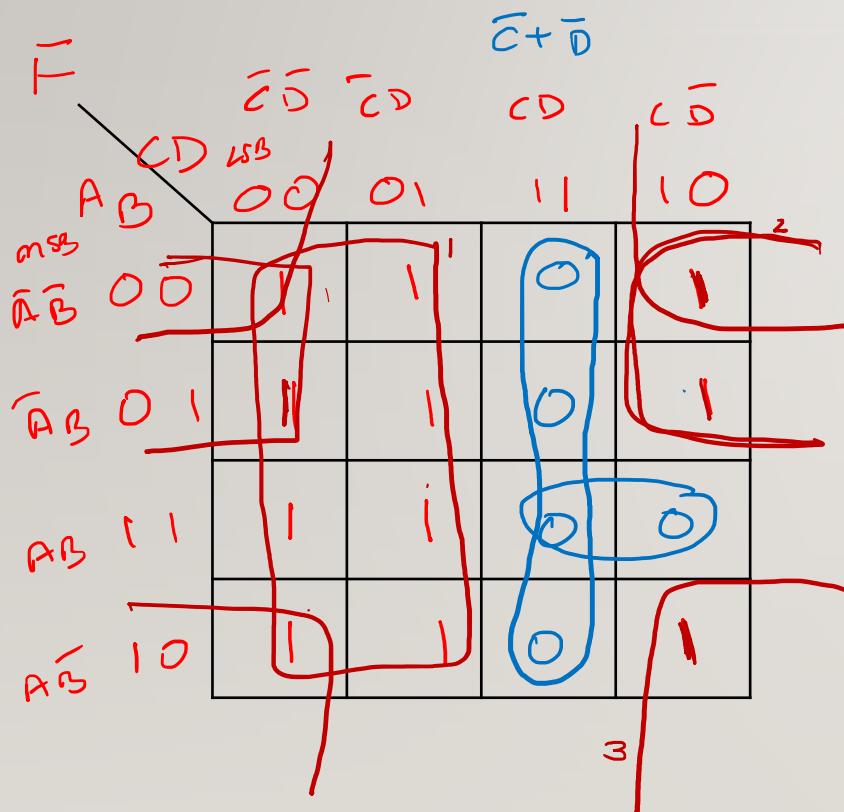
$$F = (A + \underline{C+D}) \cdot (\bar{A} + B + \bar{C}) \cdot (\bar{A} + \bar{B} + C) \cdot (\bar{A} + \bar{C} + \bar{D})$$

OR

$$F = (\bar{B} + C+D) \cdot (\bar{A} + \bar{B} + \bar{D}) \cdot (B + \bar{C} + \bar{D}) \cdot (A + B + \bar{D})$$

## EXAMPLE 2:

$$F(A, B, C, D) = \sum_{m_0}^{m_3} (0, 1, 2, 4, 5, 6, 8, 9, 10, 12, 13)$$



$$F = m_0 + m_1 + m_2 + m_4 + m_5 + m_6 \\ + m_8 + m_9 + m_{10} + m_{12} + m_{13}$$

$x + x + x + x + \dots \Rightarrow x$

Note: A pair of minterms combined  $\rightarrow$  1 variable vanishes

For Ex:

$$m_{10} + m_2$$

$$A\bar{B}C\bar{D} + \bar{A}\bar{B}C\bar{D}$$

$$\rightarrow (A + \bar{A})\bar{B}C\bar{D} = \bar{B}C\bar{D}$$

ONE 3 IP AND

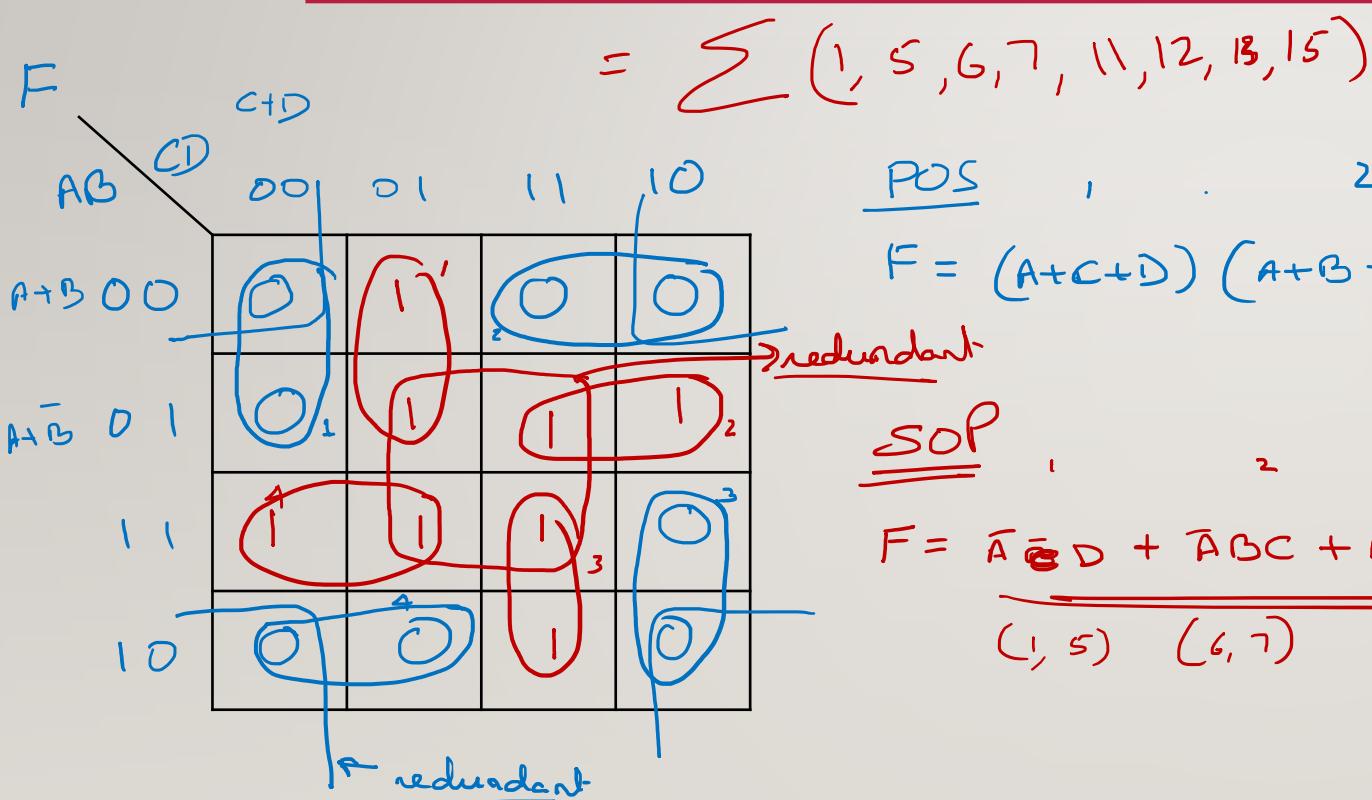
possible

$$\underline{\underline{m_0 + m_2 + m_8 + m_{10}}}$$

2 4 1/p AND gates

## EXAMPLE 3:

$$F(A, B, C, D) = \prod_m (0, 2, 3, 4, 8, 9, 10, 14)$$



POS , . . . 2 . 3 4

$$F = (A+C+D) (A+B+\bar{C}) (\bar{A}+\bar{C}+D) (\bar{A}+B+C)$$

SOP , . . . 2 3 4 redundant

$$F = \bar{A}\bar{C}D + \bar{A}BC + ACD + AB\bar{C} + (\quad)$$

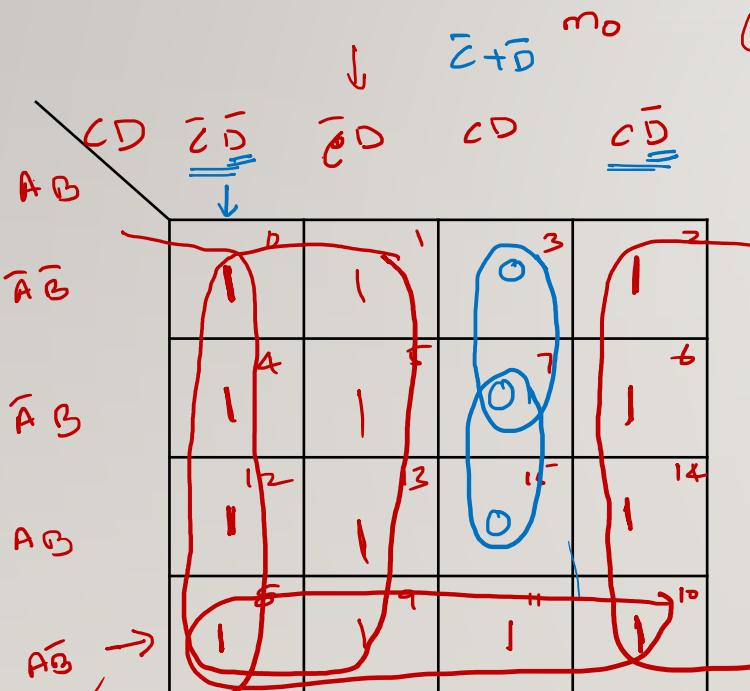
$\overbrace{(1, 5)}^{(1, 5)}, \overbrace{(6, 7)}^{(6, 7)}, \overbrace{(11, 12)}^{(11, 12)}, \overbrace{(13, 15)}^{(13, 15)}$   
 Quad

## EXAMPLE 4:

$$F(A, B, C, D) = \overline{C}(\overline{A}\overline{B}\overline{D}) + D + A\overline{B}C + \overline{D}$$

~~LSG~~

$$= \overline{A}\overline{B}\overline{C}\overline{D} + \overline{C}D + A\overline{B}C + \overline{D}$$



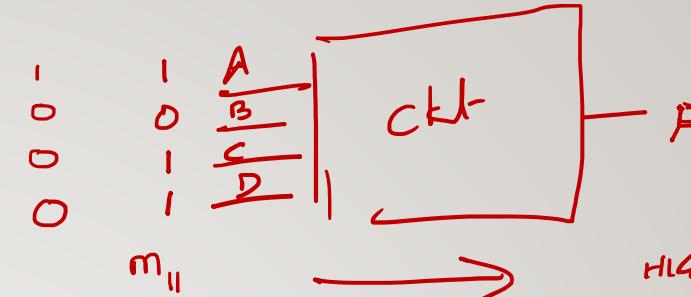
$$F = \sum_s (0, 1, 2, 4, 5, 6, 8, 9, 10, 11)$$

$$\begin{aligned} & \downarrow \quad \overline{C} + \overline{D} \quad m_0 \\ & (\overline{A} + A)(\overline{B} + B)\overline{C}\overline{D} \\ & \quad \quad \quad \quad \quad \overline{C}\overline{D} \\ & m_1 \quad m_5 \quad m_9 \quad m_{13} \end{aligned}$$

$\overline{C}\overline{D} \rightarrow$  A LGB one missing  
Quadr group  
 $A\overline{B}C \rightarrow$  1 variable missing D  
 pair group  $m_{1,0} \text{ & } m_{11}$

$$\begin{aligned} & A\overline{B}C(\overline{D} + D) \\ & = A\overline{B}C\overline{D} + A\overline{B}CD \\ & \quad m_{10} \quad m_{11} \end{aligned}$$

$\overline{D} \rightarrow$  3 variable missing  
 group 8 minterms



$m_{11}$

HIGH

Simplified Expression

$$F = \overline{C} + \overline{D} + A\overline{B}$$

SOP

POS

$$F = (A + \overline{C} + \overline{D})(\overline{B} + \overline{C} + \overline{D})$$

Distributive law

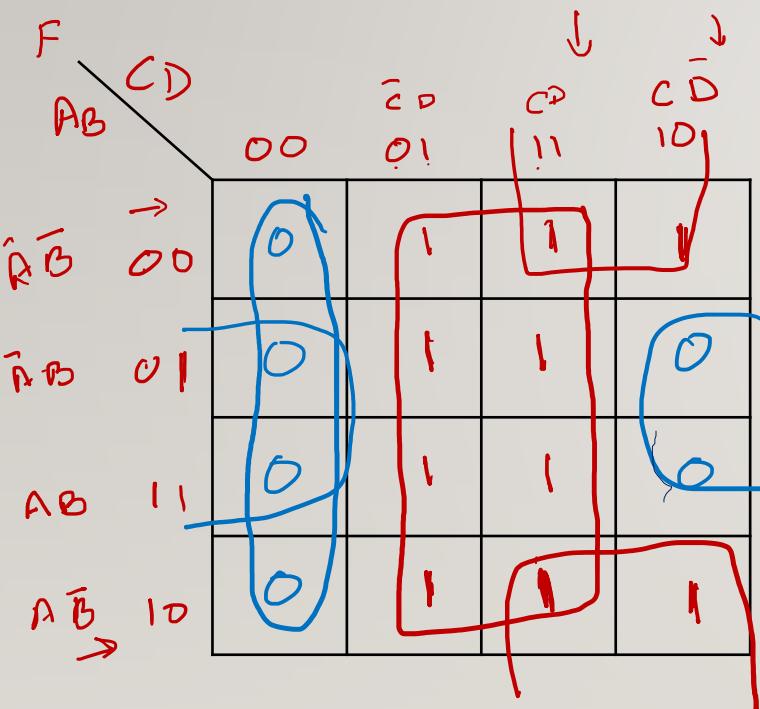
$$\begin{aligned} x + y \cdot z &= (x+y)(x+z) \\ x(y+z) &= xy + xz \end{aligned}$$

$$F = (\overline{C} + \overline{D} + A)(\overline{B} + \overline{C} + \overline{D})$$

# EXAMPLE 5:

$$\begin{aligned}
 F(A,B,C,D) &= D(\bar{A} + B) + \bar{B}(C + AD) \\
 &= \bar{A}D + BD + \bar{B}C + A\bar{B}D
 \end{aligned}$$

$$= \sum_m (1, 2, 3, 5, 7, 9, 10, 11, 13, 15)$$



$$= \prod_m (0, 4, 6, 8, 12, 14)$$

$$A+B \rightarrow \text{POS} = (C+D)(\bar{B}+D)$$

↓ distributive law

$$\text{SOP} = \underline{D+(\bar{B}C)} \rightarrow (\bar{B}+D)(C+D)$$

↓ distributive law

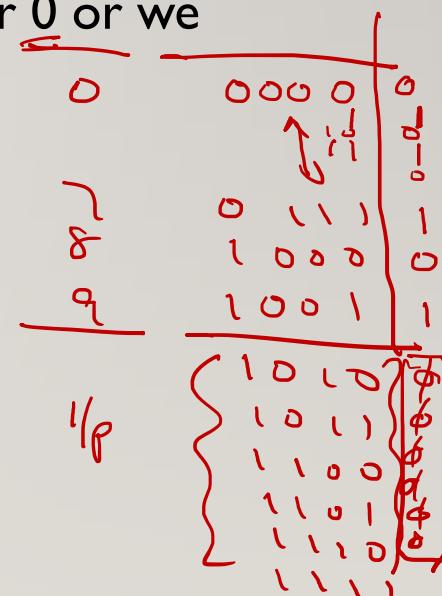
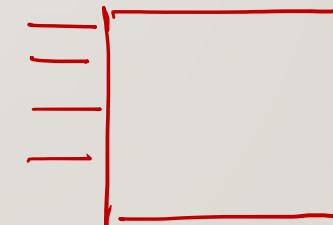
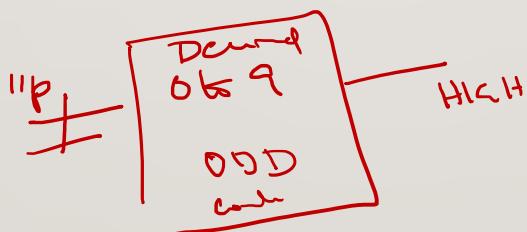
# DON'T CARE CONDITION

---

- The “Don’t Care” conditions indicate the input combinations which are invalid for a particular circuit.

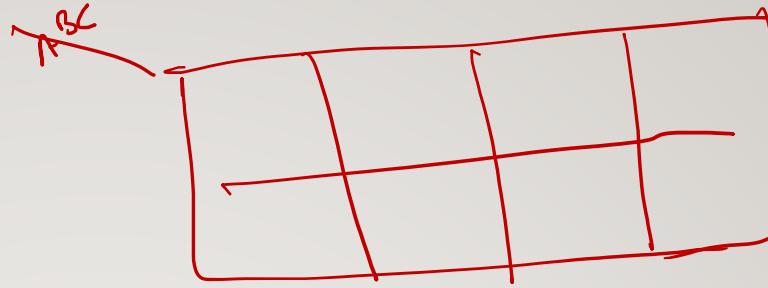
$\phi$       d

- While forming groups of cells, we can consider a “Don’t Care” cell as either 1 or 0 or we can simply ignore that cell.
- Therefore, “Don’t Care” condition are used to form a larger group of cells.

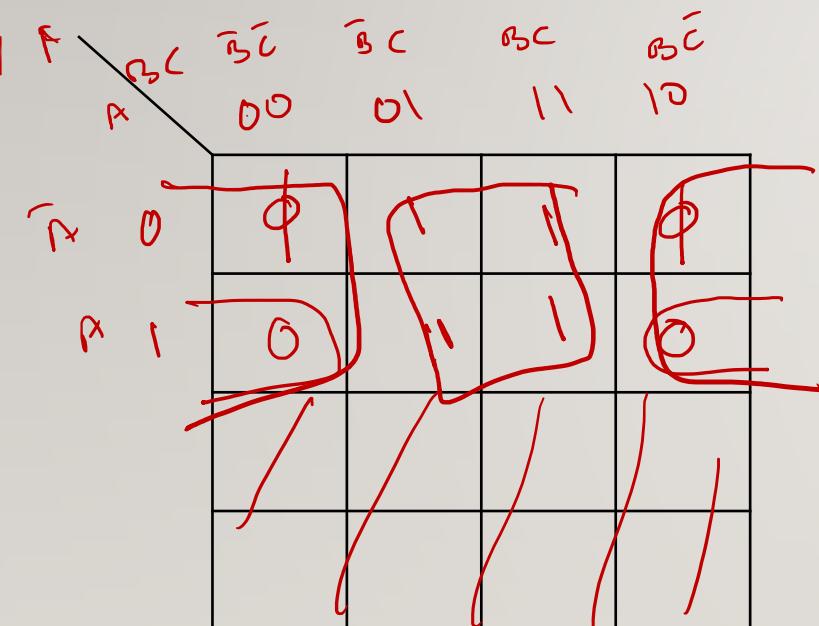


# EXAMPLE I:

$$F(A, B, C) = \sum_m (1, 3, 5, 7) + \sum_d (0, 2)$$



$$= \prod_m (\overline{A} \cdot \overline{B} \cdot C) + \prod_d (\overline{A} \cdot B \cdot \overline{C})$$



$$\begin{array}{|c|} \hline F = C \\ \hline \end{array}$$

SOP  
POS =  $(\overline{A} + C)$   
will don't care

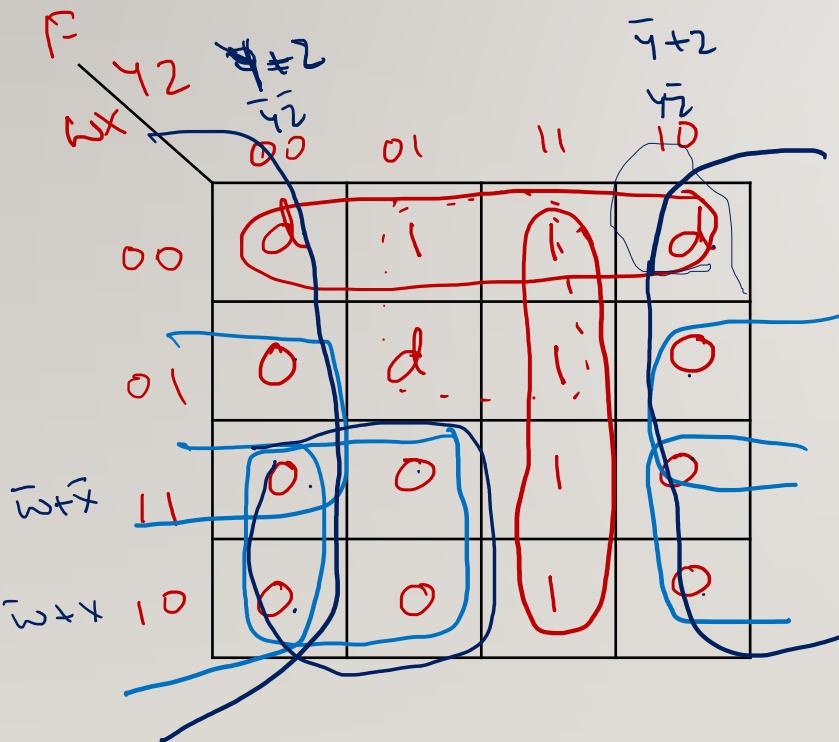
$$\begin{array}{|c|} \hline F = C \\ \hline \end{array}$$

POS

## EXAMPLE 2:

$$F(W, X, Y, Z) = \sum_m (1, 3, 7, 11, 15) + \sum_d (0, 2, 5)$$

$$= \prod_m (4, 6, 8, 9, 10) \cdot \prod_d (0, 2, 5)$$



SOP

$$F = Y_2 + \bar{W}\bar{X}$$

OR

$$F = Y_2 + \bar{W}Z$$

POS  $F = (\bar{W}+Y)(\bar{W}+Z)(\bar{X}+Z)$

Consider clubbing with don't cares

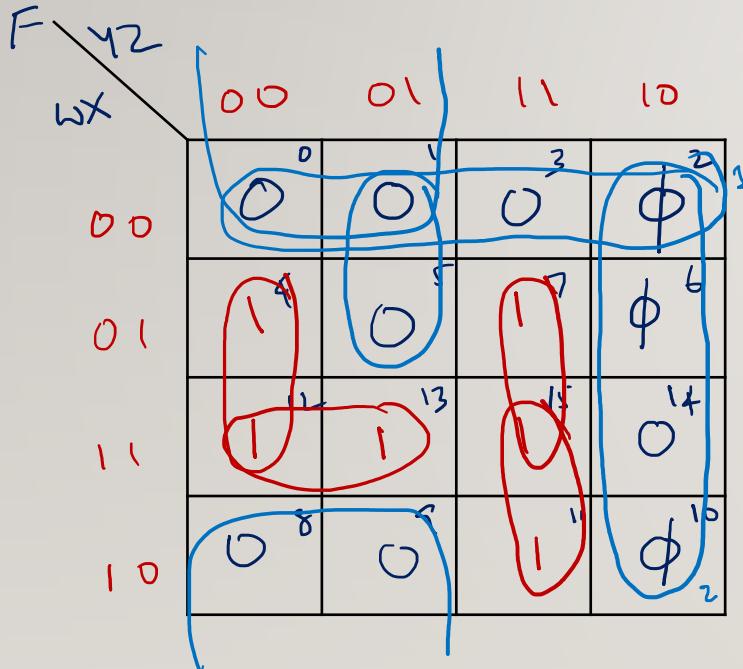
$$F = \underline{Z} + (\bar{W}+Y)$$

$$\underline{YZ} + \underline{\bar{W}Z}$$

## EXAMPLE 3:

$$F(W, X, Y, Z) = \prod_M (0, 1, 3, 5, 8, 9, 14) \cdot \prod_D (2, 6, 10) = \sum_m 4, 7, 11, 12, 13, 15 + \sum_d 2, 5, 10$$


---



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

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

P. Note:

Clubbing rule

1. Club max number of 1's / 0's along with ' $\phi$ '.

to get max combinations

2. Check for uncovered 1's / 0's

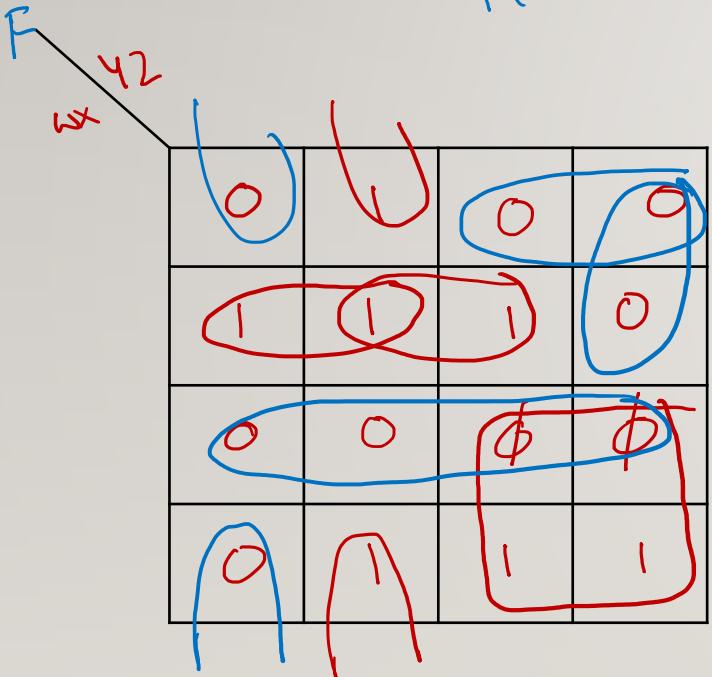
3. If any uncovered 1's / 0's  $\rightarrow$  along with ' $\phi$ 's go for max clubbing

SOP / POS

## EXAMPLE 4:

$$F(W, X, Y, Z) = \sum_m (1, 4, 5, 7, 9, 10, 11) + D(14, 15)$$

~~$= \overline{\bigcup_m (0, 2, 3, 6, 8, 12, 13)} + D(14, 15)$~~



SOP  $\Rightarrow F =$

pls. wdh

POS  $\Rightarrow F =$

pls. wdh

$\phi ( )$   
 $d ( )$   
 $D ( )$

## EXAMPLE 5:

Design a combinational circuit with 4- input lines that represents a decimal digit in BCD and 4- output lines that generates 2's complement of input digit.

---




# EXAMPLE 6:

Design a combinational circuit to check for even parity of 4 bits. A logic '1' output is required when the 4 bits constitute an even parity.

		$wx\bar{y}z$	$wx\bar{y}z$	$wx\bar{y}z$	$wx\bar{y}z$
		00	01	11	10
00	00	0	0	0	0
01	01	0	0	0	1
11	11	0	0	1	0
10	10	0	1	0	1

Ex-OR Gates    EX-NOR



$\Rightarrow$  Check for Even parity

Even parity

$'1' p \rightarrow$  for Even number of '1's in  
 $'1' p$   $\rightarrow$  if Yes  $\rightarrow 1$   
No  $\rightarrow 0$

4-bit

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

$$+ w\bar{x}\bar{y}\bar{z} + w\bar{x}y\bar{z} + \bar{w}\bar{x}y\bar{z} + w\bar{x}yz$$

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

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

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

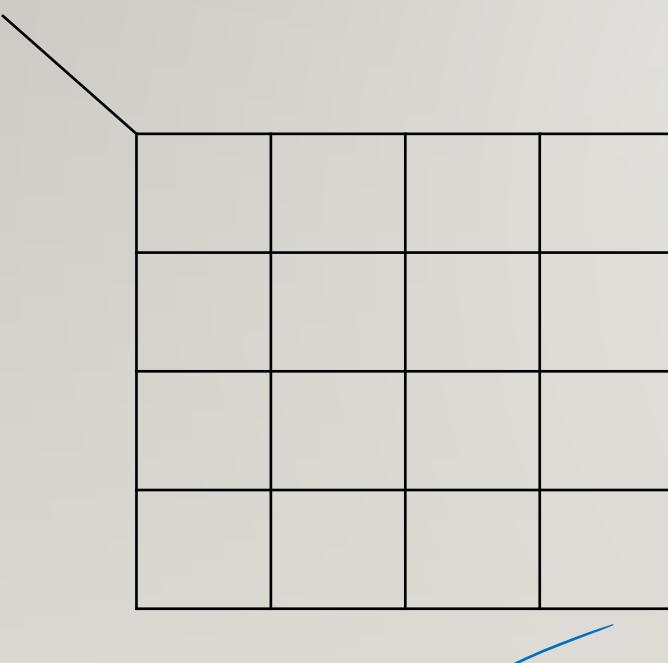
$$= (\cancel{w\bar{x}} + \cancel{\bar{w}x})(y\oplus z) + (\cancel{w\bar{x}} + \cancel{w\bar{x}})(y\oplus z) = \underline{\underline{w\oplus x\oplus y\oplus z}}$$

$w \times y \bar{z}$	$F$	$m_0$
0 0 0 0	0	$m_1$
0 0 0 1	0	$m_L$
0 0 1 0	0	$m_3$
0 0 1 1	1	
0 1 0 0	0	$m_4$
0 1 0 1	1	
0 1 1 0	1	
0 1 1 1	0	
1 0 0 0	0	
1 0 0 1	1	
1 0 1 0	1	
1 0 1 1	0	
1 1 0 0	1	
1 1 0 1	0	
1 1 1 0	0	
1 1 1 1	1	

## EXAMPLE 7:

Design a combinational circuit that multiplies by '5' an input decimal digit represented in BCD. The output is also in BCD.

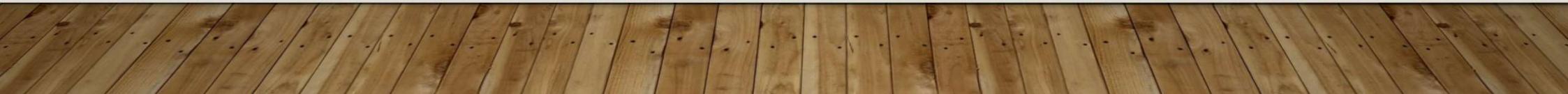
---




# LECTURE 5

---

KARNAUGH MAP (K – MAP)



EXAMPLE 5:

DESIGN A COMBINATIONAL CIRCUIT WITH 4- INPUT LINES THAT REPRESENTS A DECIMAL DIGIT IN BCD AND 4- OUTPUT LINES THAT GENERATES 2'S COMPLEMENT OF INPUT DIGIT.

Decimal digit	8 4 2 1 A B C D	2's Complement $y_3 y_2 y_1 y_0$
0	0000	0000
1	0001	1111
2	0010	1110
3	0011	1101
4	0100	1100
5	0101	1011
6	0110	1010
7	0111	1001
8	1000	0001 1000
9	1001	0111

•

0100  $\Rightarrow$  1's comp =  $1 \overset{1}{\cancel{0}} 11 + \frac{1}{1}$   
 $\uparrow (4)$

Sign bit = 0 = +ve  
 1 = -ve

$\uparrow (-4) \text{ of } (0100)$

1's comp. (1100)  $\leftarrow$  LSB

4-bit  $\therefore P = 2^4 = 16$  input combinations  
 $0000 - 1001$   
 $= 10$  combinations

$(-8)_{10} < 0 \rightarrow (0\overset{1}{\cancel{1}}\overset{1}{\cancel{1}})_2$        $(+7)_{10}$

5-bits  
 $0 \rightarrow$

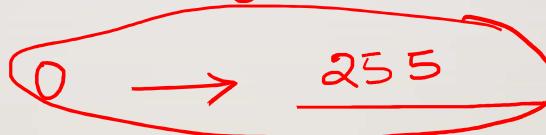
5-bits

Unsigned  $\Rightarrow 00000 - 11111 \Rightarrow (0) \rightarrow (31)$

Signed  $\Rightarrow \begin{array}{c} 10000 \\ \oplus 000 \\ \hline (-16) \end{array} \leftarrow 00000 \rightarrow 01111 (+15)_{10}$

$$2^5 - 1$$

unsigned short int A; 1 byte = 8 bits  $\Rightarrow 2^8 = 256$



①

{  $A = -3;$   
 $A = 257;$  → }

?

To find Q's complement contd:

$$Y_3 = \sum m(0, 1, 2, 3, 4, 5, 6, 7, 8) + d(10, 11, 12, 13, 14, 15)$$

$$Y_2 = \sum m(0, 1, 2, 3, 4, 5, 6, 9) + d(10, 11, 12, 13, 14, 15)$$

$$Y_1 = \sum m(1, 2, 5, 6, 9) + d(10, 11, 12, 13, 14, 15)$$

$$Y_0 = \sum m(3, 5, 7, 9) + d(11)$$

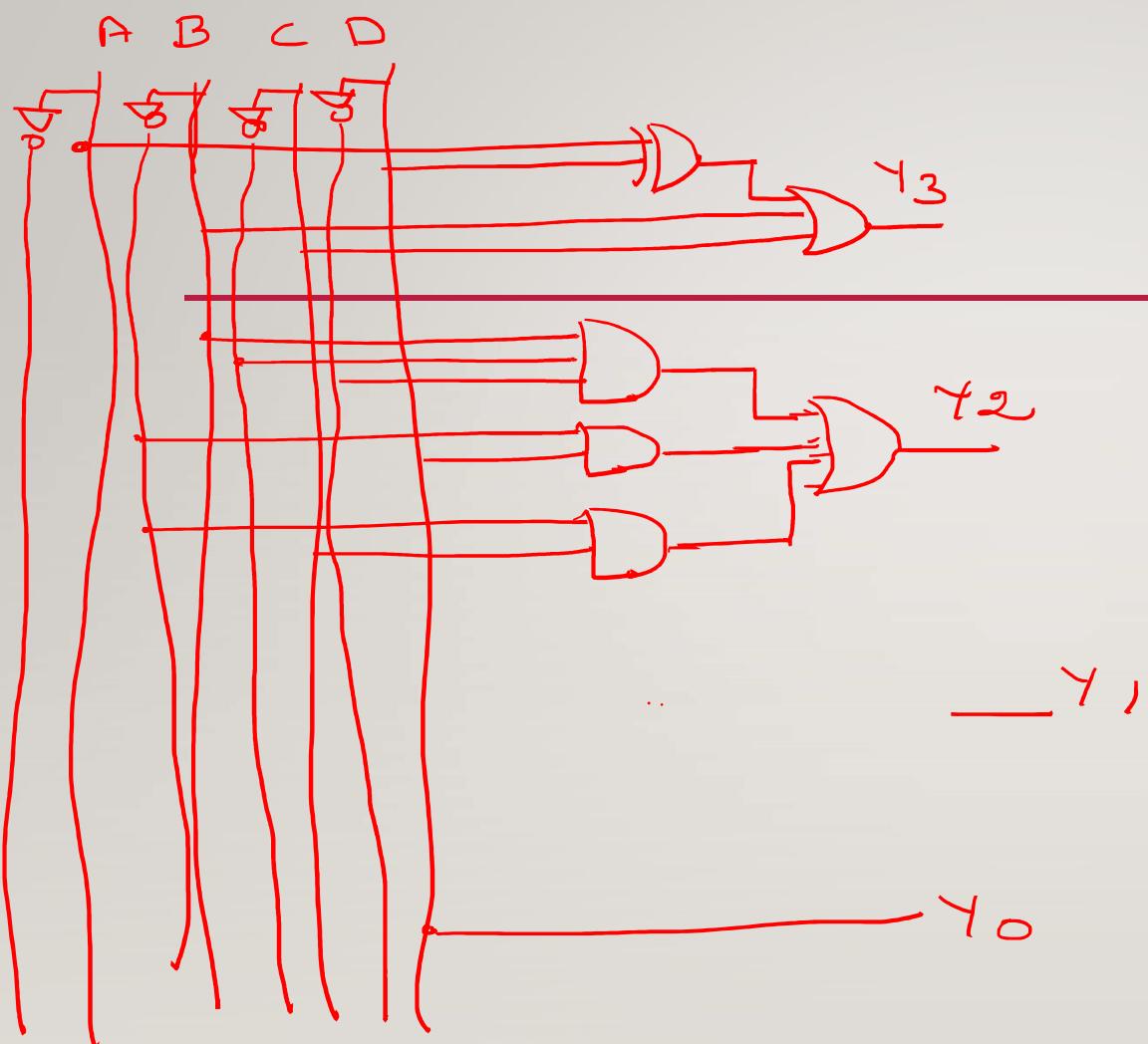
		$\bar{A}D$	$\bar{A}D$	$\bar{A}D$	$\bar{A}D$	
$\bar{A}B$	$\bar{B}$	0	1	1	1	10
$B$	$\bar{B}$	00	01	11	10	10
$A$		d	d	d	d	
		0	0	0	0	

$$Y_3 = B + C + \bar{A}D + A\bar{D}$$

$$Y_2 = \cancel{B} + \bar{B}D + \cancel{\bar{B}C} = B\bar{C}\bar{D} + \bar{B}D + \bar{B}C$$

$$Y_1 = ? \quad \bar{C}D + C\bar{D} = C \oplus D$$

$$Y_0 = D, \text{ from truth table}$$



$$Y_3 = B + C + \underbrace{A\bar{D} + \bar{A}D}_{A \oplus D}$$

$$Y_2 = BC\bar{D} + \bar{B}D + \bar{B}C$$

## EXAMPLE 6:

Design a combinational circuit to check for even parity of 4 bits. A logic '1' output is required when the 4 bits constitute an even parity.

---

## EXAMPLE 7:

Design a combinational circuit that multiplies by '5' an input decimal digit represented in BCD. The output is also in BCD.

---

Input :  $(0)_{10}$  to  $(9)_{10}$        $(0000)$  to  $(1001)$  in BCD  
output :  $(0)_{10}$  to  $(45)_{10}$        $(000000)$  to  $(0100\ 0101)$  in BCD

# CODE CONVERTERS

# Code converters

- A code converter circuit will convert coded information in one form to a different coding form.
- Coded representation for 10 decimal symbols is known as binary coded decimal (or BCD) or decimal codes.
- Minimum 4-bits are required to represent decimal symbol.
- Out of 16 , 4-bit combinations, only 10 combinations are used to represent 10 decimal symbols and remaining 6 will not be used (don't cares)

# Binary and Binary coded Decimal(BCD)

Decimal	Binary	BCD
0	0000	00 00
..	"	..
9	1001	1001
10	1010	0001 0000
11	1011	0001 0001
12	1100	0001 0010
13	1101	0001 0011
67		0110 0111
90		1001 0000
23		0010 0011

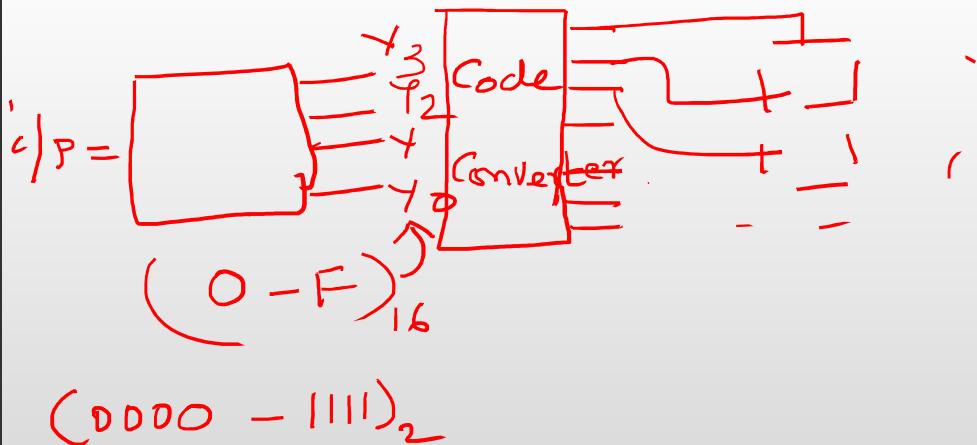
10  
 ↓    ↓  
0001 0000  
 8421 BCD code  
 8 4 2 1  
 1001 = (9) <sub>10</sub>

# Difference between binary and BCD representation

- $(28)_{10}$

Binary representation :  $(11100)_2$

8421 BCD representation :  $(0010\ 1000)_2$



# Introduction to BCD codes (4-bit)

Weighted codes  
 $= 8421, 84-2-1,$   
 $2421$

Decimal digit	8421 (BCD)	Excess 3	84-2-1	2421	Gray code
0	0000	0011	0000	0000	0000
1	0001	0100	0111	0001	0001
2	0010	0101	0110	0010 1000	0011
3	0011	0110 ✓	0101	0011 1001	<del>0010</del>
4	0100	0111 ✓	0100	0100 1010	<del>0110</del>
5	0101	1000 ✓	<del>1011</del>	1011 0101	<del>0100</del> = 0111
6	0110	1001 ✓	1010	1100 0110	0101
7	0111	1010 ✓	1001	1011 0111	0100
8	1000	1011 ✓	1000	1110	1100
9	1001	1100 ✓	1111	1111	1101
Don't cares	1010, 1011 1100, 1101 1110, 1111	0000, 0001 0010, 1101 1110, 1111	0001, 0010, 0011, 1100, 1101, 1110	1000 1001 1010 1010 0101 0110, 0111	0010 0011 0100 1011 1010, 1000 1011, 1001

Self - Complementary codes  
 ↳ Excess -3  
 $84-2-1$   
 $2421$

# Complements

Are used for simplifying the subtraction operation and for logical manipulation.

There are two complements for each base:

- (R-1)'s complement (Diminished radix complement)
- R's complement (Radix complement)

- (R-1)'s complement:

(R-1)'s complement of a number is  $(R^n - 1) - N$

Where  $\underline{\underline{R}} \rightarrow$  base

$\underline{\underline{N}}$  → number whose complement is to be taken

$\underline{\underline{n}}$  → number of digits/bits in the number N

$$R = \begin{matrix} 2 & 1 \\ 2 & 2 \\ 8 & 1 \\ 10 & 1 \\ 16 & 1 \end{matrix}$$

single-digit no :  $n=1$   
 $R=2$        $N=0$        $(R-1)'s = 1's$  compl of  $N=0$

$$(R-1) - N$$
$$\begin{array}{r} 1 - 0 = 1 \\ 1 - 1 = 0 \end{array}$$

## ■ R's complement

R's complement of a number is  $R^n - N$

Where  $R \rightarrow$  base

$$\equiv$$

$N \rightarrow$  number whose complement is to be taken

$n \rightarrow$  number of digits/bits in the number  $N$

Examples:  $R=2, N=0, 2^1$ 's compl. of a no }  $\begin{array}{l} 2^1 \text{'s compl. of a no} \\ R^1 \text{'s compl. of a no} \end{array}$

$$R^n - N \Rightarrow \text{single bit} = n=1$$

$$2^1 - N \Rightarrow N=0,$$

$$\underline{\underline{(R^n-1)-N+1}} = \underline{\underline{R^n-N}}$$

$2^1 \text{'s compl. of } 1 = 1$   
 $10^1 \text{'s compl. of } 1 = 10-1 = 9$

- Any questions?

# CODE CONVERTERS

# Code converters

- A code converter circuit will convert coded information in one form to a different coding form.
- Coded representation for 10 decimal symbols is known as binary coded decimal (or BCD) or decimal codes.
- Minimum 4-bits are required to represent decimal symbol.
- Out of 16 , 4-bit combinations, only 10 combinations are used to represent 10 decimal symbols and remaining 6 will not be used (don't cares)

# Binary and Binary coded Decimal(BCD)

Decimal	Binary	BCD
0	0000	00 00
..	"	. "
9	1001	1001
10	1010	0001 0000
11	1011	0001 0001
12	1100	0001 0010
13	1101	0001 0011
67		0110 0111
90		1001 0000
23		0010 0011

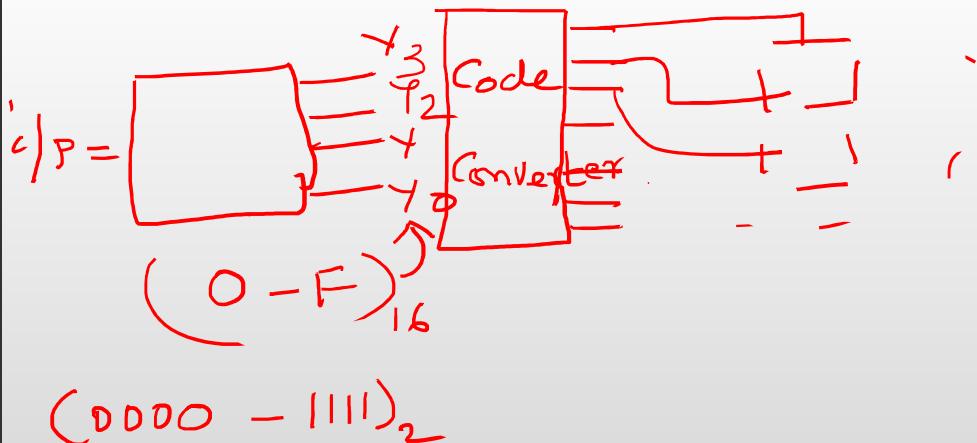
10  
 ↓      ↓  
0001 0000  
 8421 BCD code  
 8 4 2 1  
 1001 = (9) <sub>10</sub>

# Difference between binary and BCD representation

- $(28)_{10}$

Binary representation :  $(11100)_2$

8421 BCD representation :  $(0010\ 1000)_2$



# Introduction to BCD codes (4-bit)

Weighted codes  
 $= 8421, 84-2-1,$   
 $2421$

Decimal digit	8421 (BCD)	Excess 3	84-2-1	2421	Gray code
0	0000	0011	0000	0000	0000
1	0001	0100	0111	0001	0001
2	0010	0101	0110	0010 1000	0011
3	0011	0110 ✓	0101	0011 1001	<del>0010</del>
4	0100	0111 ✓	0100	0100 1010	<del>0110</del>
5	0101	1000 ✓	<del>1011</del>	1011 0101	<del>0100</del> = 0111
6	0110	1001 ✓	1010	1100 0110	0101
7	0111	1010 ✓	1001	1011 0111	0100
8	1000	1011 ✓	1000	1110	1100
9	1001	1100 ✓	1111	1111	1101
Don't cares	1010, 1011 1100, 1101 1110, 1111	0000, 0001 0010, 1101 1110, 1111	0001, 0010, 0011, 1100, 1101, 1110	1000 1001 1010 1010 0101 0110, 0111	0010 0011 0100 1011 1010, 1000 1011, 1001

Self - Complementary codes  
 ↳ Excess -3  
 $84-2-1$   
 $2421$

# Complements

Are used for simplifying the subtraction operation and for logical manipulation.

There are two complements for each base:

- (R-1)'s complement (Diminished radix complement)
- R's complement (Radix complement)

- (R-1)'s complement:

(R-1)'s complement of a number is  $(R^n - 1) - N$

Where  $\underline{\underline{R}} \rightarrow$  base

$\underline{\underline{N}}$   $\rightarrow$  number whose complement is to be taken

$\underline{\underline{n}}$   $\rightarrow$  number of digits/bits in the number N

$$R = \begin{matrix} 2 & 1 \\ 2 & 8 \\ 1 & 8 \\ 1 & 0 \\ 1 & 6 \end{matrix}$$

single-digit no :  $n=1$   
 $R=2$        $N=0$        $(R-1)'s = 1's$  compl of  $N=0$

$$(R-1) - N$$
$$\begin{array}{r} 1 - 0 = 1 \\ 1 - 1 = 0 \end{array}$$

## ■ R's complement

R's complement of a number is  $R^n - N$

Where  $R \rightarrow$  base

$$\equiv$$

$N \rightarrow$  number whose complement is to be taken

$n \rightarrow$  number of digits/bits in the number  $N$

Examples:  $R=2, N=0, 2^1$ 's compl of a no }  $\begin{array}{l} 2^1 \text{'s compl of } 0 = 1 \\ 2^1 \text{'s compl of } 1 = 10-1 = 9 \end{array}$

$R^n - N \Rightarrow$  single bit =  $n=1$

$2^1 - N \Rightarrow N=0,$

$(R^n - 1) - N + 1 = R^n \equiv$

Ex:  $R=10, n=\underline{\underline{1}}$

$$(R-1)'s = 9's$$

$$(R'-1) - N = (9-N) = 9's \text{ compl. of } N$$

$$N=8 \Rightarrow 9-8 = 1$$

$\uparrow$   
9's compl of 8

$$\left. \begin{array}{l} 9's \text{ compl. of } 1 = 8 \\ 10's \text{ compl. of } 1 = 10-1 = 9 \end{array} \right\}$$

# Code converter design steps:

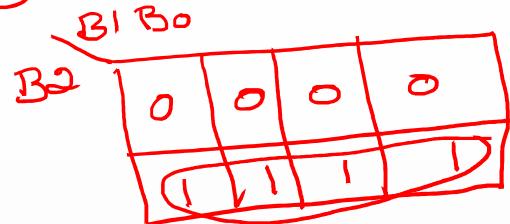
- 1. Write the truth table
- 2. Identify the don't care inputs from input code
- 3. Write the minterms/maxterms for every output variable
- 4. Simplify the expressions for output variables
- 5. Draw the circuit using the specified gates.

1. Design a 3 bit binary to gray code converter.

(a)

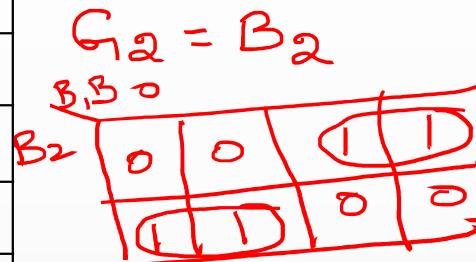
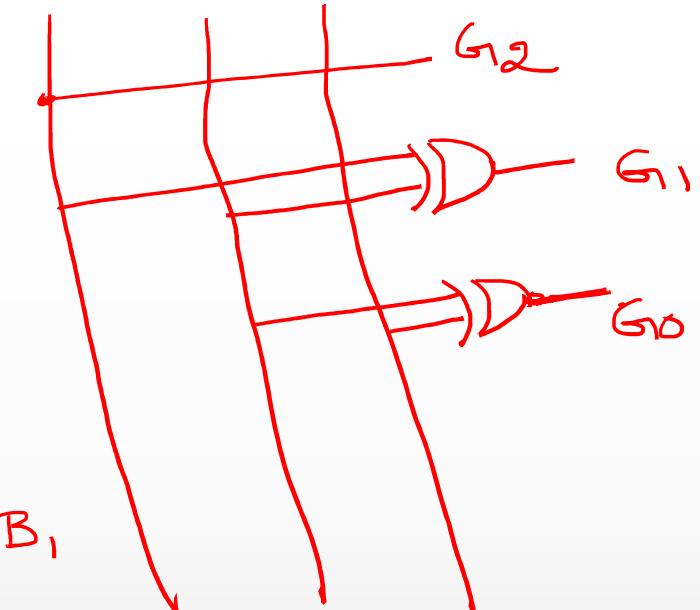
3-bit Binary B <sub>2</sub> B <sub>1</sub> B <sub>0</sub>	Gray G <sub>2</sub> G <sub>1</sub> G <sub>0</sub>
000	000
001	001
010	011
011	010
100	110
101	111
110	101
111	100

(c)

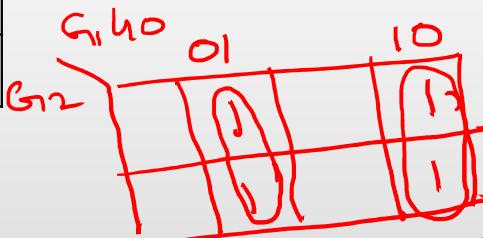


(d)

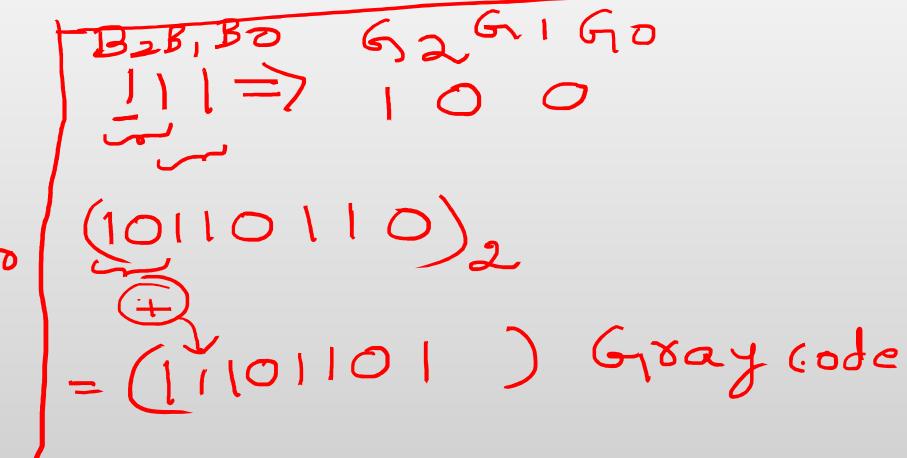
B<sub>2</sub> B<sub>1</sub> B<sub>0</sub>



$$G_1 = \overline{B}_2 B_1 + B_2 \overline{B}_1 = B_2 \oplus B_1$$



$$G_0 = \overline{B}_1 B_0 + B_1 \overline{B}_0 = B_1 \oplus B_0$$



$$\begin{aligned} G_2 &= \Sigma m(4, 5, 6, 7) \\ G_1 &= \Sigma m(2, 3, 4, 5) \\ G_0 &= \Sigma m(1, 2, 5, 6) \end{aligned}$$

*Gray binary*

1. Design a 3 bit ~~binary~~ to ~~gray~~ code converter ~~contd..~~ ← Assignment

2. Design a code converter to convert a decimal digit represented in 8421 code to a decimal digit represented in Excess 3 code.  $\Rightarrow$  NAND gates

(a)

Decimal digit	8 4 2 1 ✓	Excess 3 code ✓
	A B C D	E3 E2 E1 E0
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100
Don't cares	1010, 1011, 1100, 1101, 1110, 1111	-----

10/11/2021

$$8421 \rightarrow \text{Excess-3}$$

$\hookrightarrow$  simplified SOT  $\rightarrow$  grouping '1's

$$E_3 = \sum m(5, 6, 7, 8, 9) + d(10, 11, 12, 13, 14, 15)$$

$$E_2 = \sum m(1, 2, 3, 4, 9) + d(10, 11, 12, 13, 14, 15)$$

$$E_1 = \sum m(0, 3, 4, 7, 8) + d( )$$

From truth table:  $E_0 = \overline{D}$

$E_3 =$			
CD	00	01	11
AB	00	1	1
00	0	1	1
01	0	d	d
11	d	d	d

$$E_3 = A + BC + BD$$

$E_2 =$			
CD	00	01	11
AB	1	1	1
00	0	1	1
01	1	d	d
11	d	d	d

$E_2 =$			
CD	00	01	11
AB	00	1	1
00	0	1	1
01	1	d	d
11	d	d	d

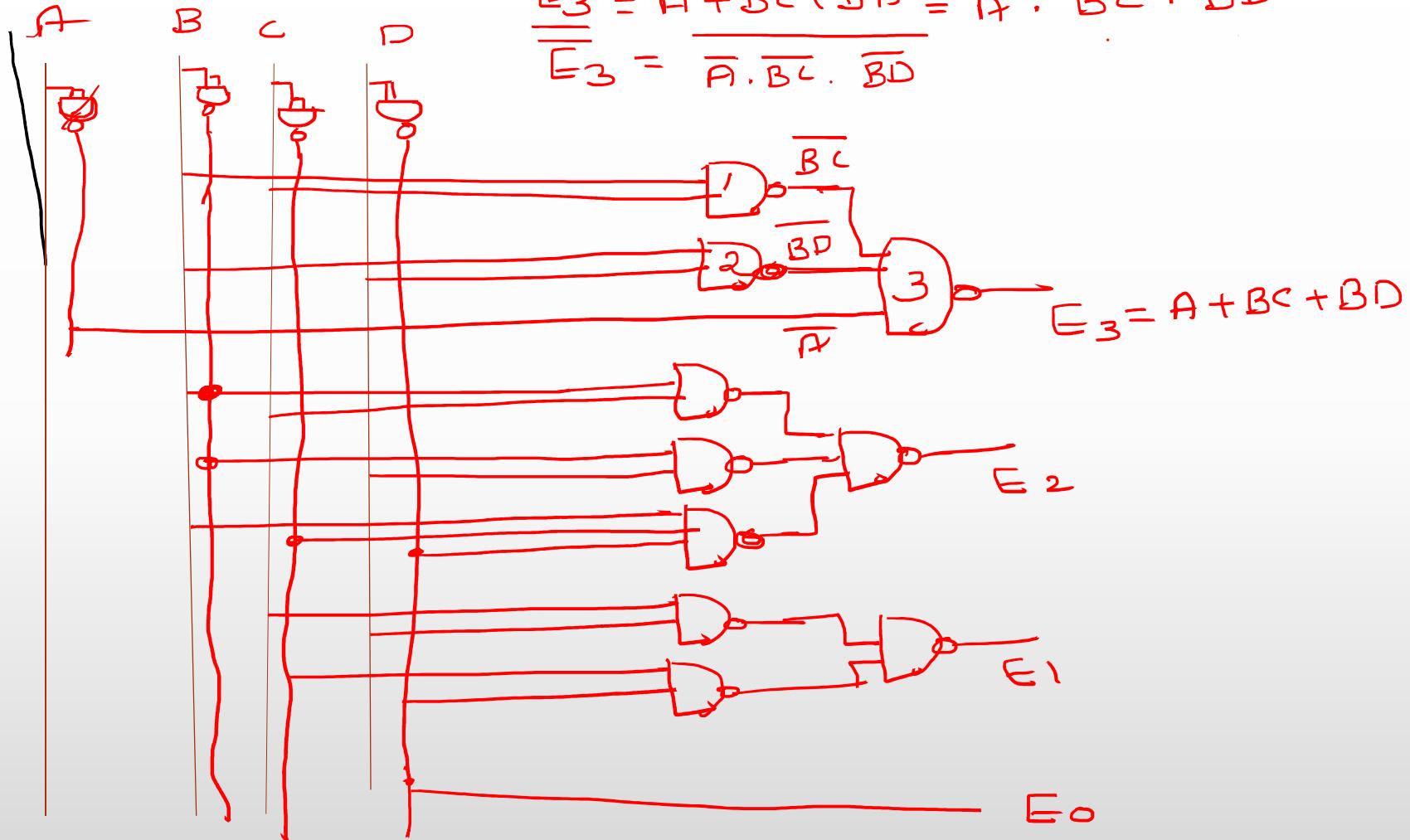
$$E_2 = \overline{B}C + \overline{B}D + B\overline{C}\overline{D}$$

$$E_1 = CD + \overline{C}\overline{D}$$

2. Design a code converter to convert a decimal digit represented in 8421 code to a decimal digit represented in Excess 3 code contd.

$$\overline{E_3} = \overline{A + BC + BD} = \overline{A} \cdot \overline{BC} \cdot \overline{BD}$$

$$\overline{E_3} = \overline{A} \cdot \overline{BC} \cdot \overline{BD}$$



2. Design a code converter to convert a decimal digit represented in 8421 code to a decimal digit represented in Excess 3 code.

---

3. Design a code converter to convert a decimal digit represented in 8 4 2 1 code to a decimal digit represented in 8 4 -2 -1 code.

$\rightarrow \text{NOR gates} \rightarrow \text{POS} \rightarrow \text{grouping } \sum_m$

Decimal digit	8 4 2 1 A B C D	8 4 -2 -1 Y <sub>3</sub> Y <sub>2</sub> Y <sub>1</sub> Y <sub>0</sub>
0	0000	0000
1	0001	0111
2	0010	0110
3	0011	0101
4	0100	0100
5	0101	1011
6	0110	1010
7	0111	1001
8	1000	1000
9	1001	1111
Don't cares	10, 11, 12, 13, 14, 15	

$$Y_3 = \pi M(0, 1, 2, 3, 4) \cdot d(10, 11, 12, 13, 14, 15)$$

$$Y_2 = \pi M(0, 5, 6, 7, 8) \cdot d(10, 11, 12, 13, 14, 15)$$

$$Y_1 = \pi M(0, 3, 4, 7, 8) \cdot d( )$$

$$Y_0 = D$$

only

③ Simplified POS

④ Circuit using NOR gates

3. Design a code converter to convert a decimal digit represented in 8 4 2 1 code to a decimal digit represented in 8 4 -2 -1 code.

4

Design a code converter to convert a decimal digit represented in Excess 3 code to a decimal digit represented in 8 4 -2 -1 code. ~~= NOR gates~~

Decimal digit	EXCESS -3 CODE A B C D	8 4 -2 -1 Y3 Y2 Y1 Y0
0	0011	0000
1	0100	0111
2	0101	0110
3	0110	0101
4	0111	0100
5	1000	1011
6	1001	1010
7	1010	1001
8	1011	1000
9	1100	1111
Don't cares	0000,0001,0010, 1101,1110,1111	-----

10/11/2021

~~Y3 = A~~

$Y_3 = A$

$Y_2 = B$

$Y_1 = \bar{C}$

$Y_0 = \bar{D}$

10/11/2021

(5)

4. Design a code converter to convert a decimal digit represented in 8 4 -2 -1 code to a decimal digit represented in 2 4 2 1 code. = NOR gates

Decimal digit	8 4 -2 -1 A B C D	↓	2 4 2 1 Y3 Y2 Y1 Y0	↓
0	0000 (0)		0000	
1	0111 (1)		0001	
2	0110 (2)		0010	
3	0101 (3)		0011	
4	0100 (4)		0100	
5	1011 (11)		1011	
6	1010 (10)		1100	
7	1001 (9)		1101	
8	1000 (8)		1110	
9	1111 (5)		1111	
Don't cares	0001, 0010, 0011, 1100, 1101, 1110			

10/11/2021

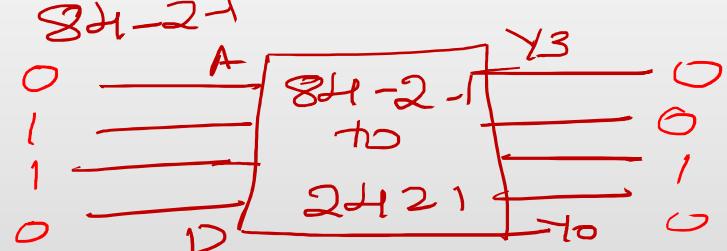
$$Y_3 = \bar{A}$$

$$Y_2 = \sum m(4, 6, 7, 8, 9) + \bar{d}(1, 2, 3, 12, 13, 14)$$

$$Y_1 = \sum m(2, 3, 5, 8, 9) + \bar{d}(2, 3, 12, 13, 14)$$

AB	00	01	11	10
00	0	1	3	2
01	4	5	-	6
10	-	-	-	-

2 4 2 1



4. Design a code converter to convert a decimal digit represented in 8 4 -2 -1 code to a decimal digit represented in 2 4 2 1 code contd  $\rightarrow$  NOR

$$Y_3 = \sum m(8, 9, 10, 11, 15) + d(1, 2, 3, 12, 13, 14) = A$$

$$Y_2 = \sum m(4, 8, 9, 10, 15) + d(1, 2, 3, 12, 13, 14) =$$

$$Y_1 = \sum m(5, 6, 8, 11, 15) + d(11)$$

$$Y_0 = D$$

	$c$	$D$	
$A$	0	$A$	$d$
00	0	0	$d$
01	1	0	0
	$d$	$d$	1
10	1	1	0

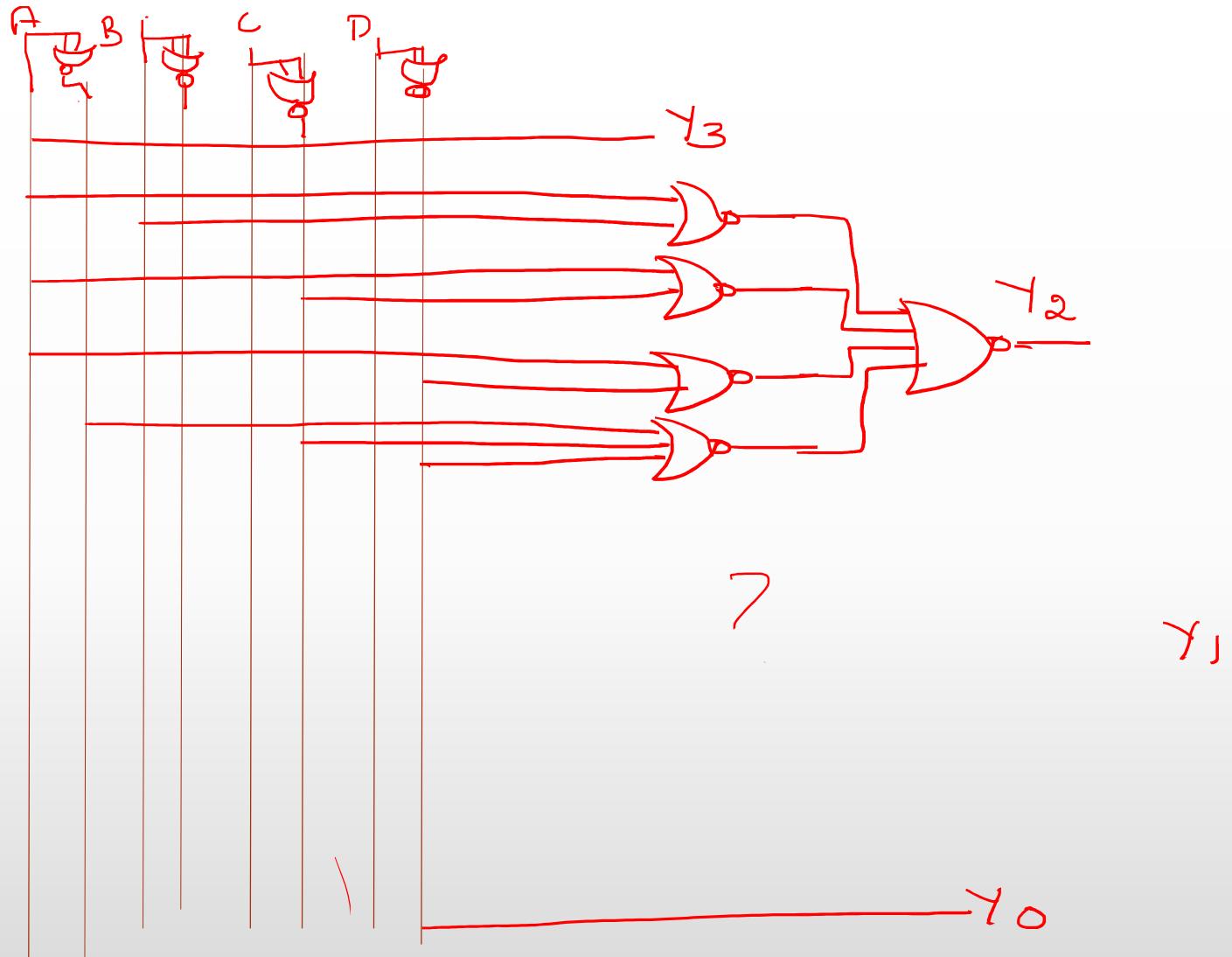
3 Quads, 1 pair

$$Y_2 = (A+B)(\bar{A}+\bar{C})(A+\bar{D})(B+\bar{C}+\bar{D})$$

	$c$	$D$	$d$	$d$	$d$	$Y_1$	$Y_0$
$B$	0	0	$d$	$d$	$d$		
00	0	0	0	0	0	1	1
01	0	1	0	0	0	1	1
	$d$	$d$	1	$d$			
11	$d$	$d$	1	$d$			
10	1	0	1	0	0		

$$Y_1 = (A+B)(\bar{B}+C+D)(\bar{A}+C+\bar{D}) \\ (\bar{A}+\bar{C}+\bar{D})(\bar{A}+\bar{C}+D)$$

$$\bar{A} + \bar{C} + D$$



5. Design a code converter to convert a decimal digit represented in 2 4 2 1 code to a decimal digit represented in gray code.

Decimal digit	2 4 2 1 A B C D	Gray code G3 G2 G1 G0
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	1011	0111
6	1100	0101
7	1101	0100
8	1 1 10	1100
9	1 1 11	1101
	0101, 0110, 0111, 1000, 1001, 1010	

10/11/2021

$$G_3 = \sum m(14, 15) + d(5, 6, 7, 8, 9, 10)$$

$$G_2 = \sum m(4, 11, 12, 13, 14, 15) + d(5, 6, 7, 8, 9, 10)$$

$$G_1 = \sum m(2, 3, 4, 11) + d( )$$

$$G_0 = \sum m(1, 2, 11, 12, 15) + d( )$$

5. Design a code converter to convert a decimal digit represented in 2 4 2 1 code to a decimal digit represented in gray code

- Any questions?

↑ 1 byte

①  $A = 257; \text{ cout} < < A < < endl;$   $A = 1$

$$\begin{array}{r} 1 | 0000\ 0001 \\ \hline 256 \end{array} \Rightarrow 1$$

$\begin{array}{r} 0011 \\ \hline 1001 \end{array}$

②  $A = -3 \Rightarrow ?$  4 bits  $\Rightarrow (-3) = 1001$

$\hookrightarrow 2 \text{ bytes} = \text{unsigned} = 0000 \dots \dots \ 16 \text{ bits}$

16-bits

253      ↓ 111... - . . . 1111

+-----  
?      1111 1111      1111 1101  
?      (65, 533)      unsigned



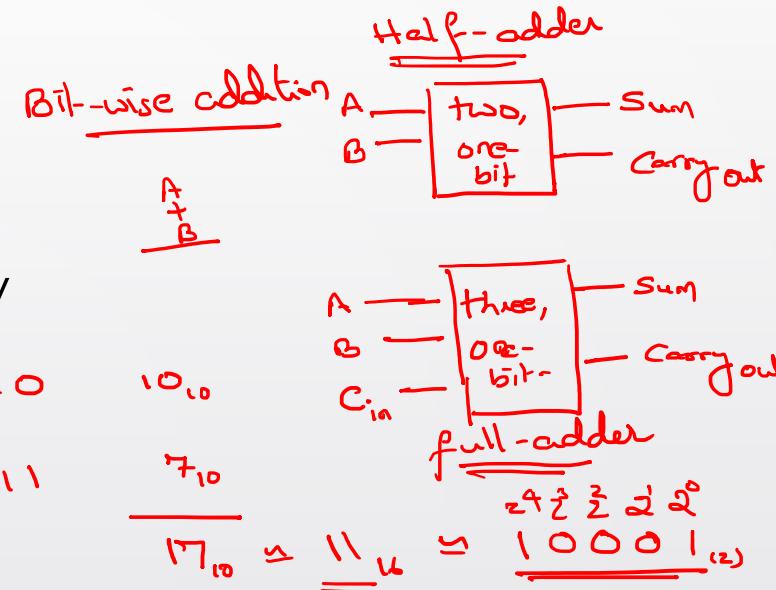
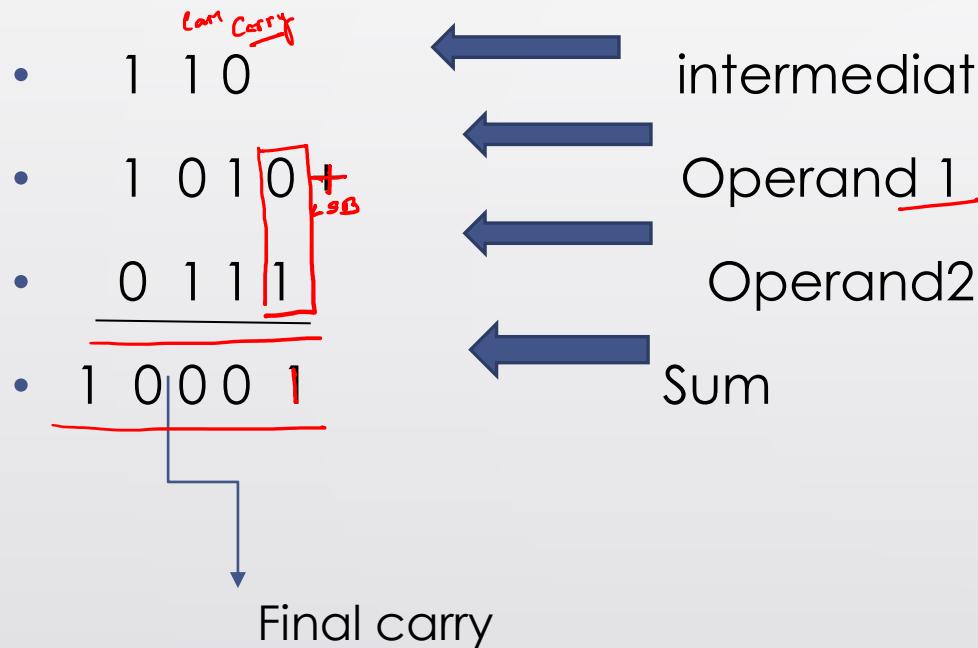
# Binary adders and subtractors

- Half adder, full adder, parallel adder
- Half subtractor , full subtractor, parallel subtractor
- Subtraction using complements, parallel adder/subtractor
- Carry Look ahead adder, Decimal adder

# Binary Addition

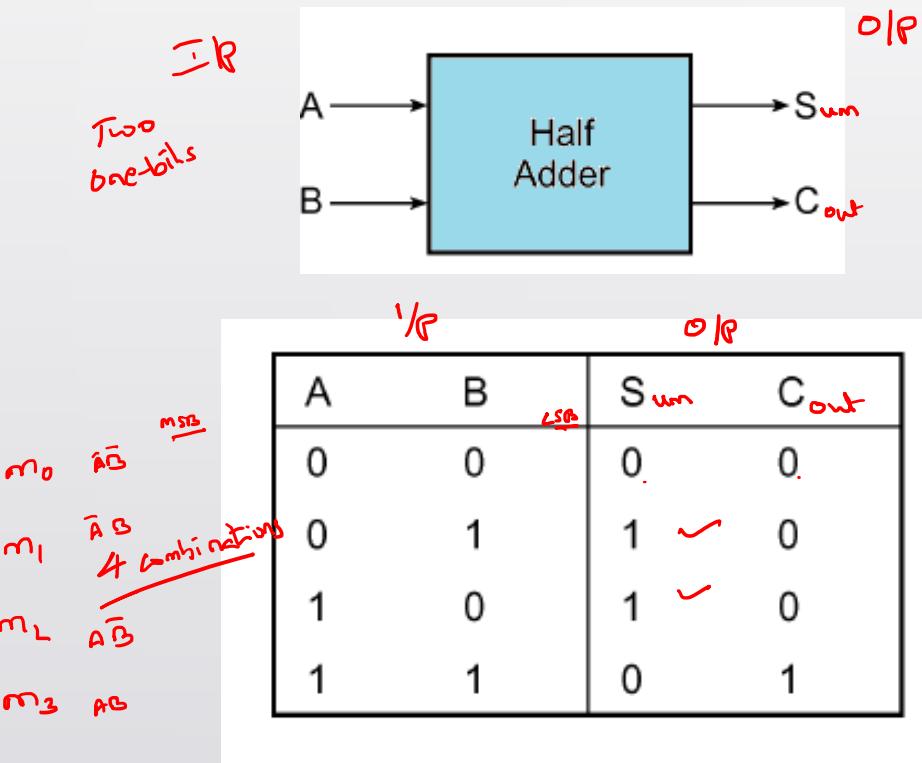
## Arithmetic Operation

Example : Addition of two 4-bit Numbers



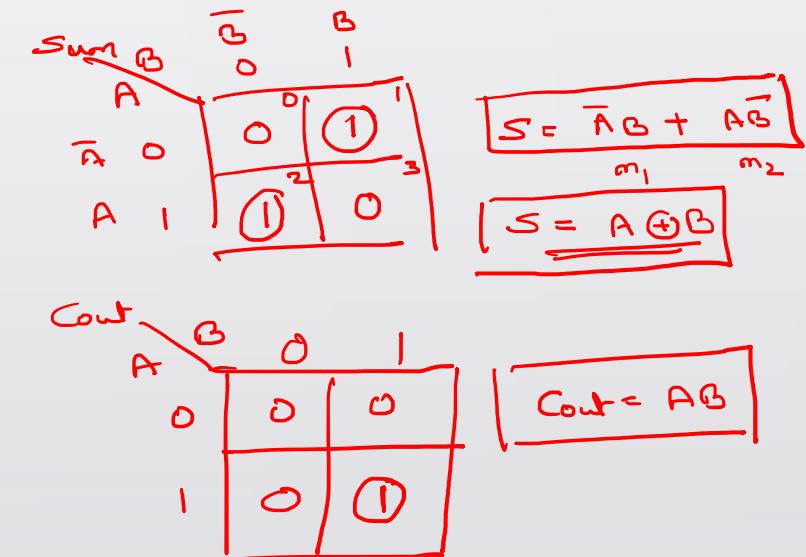
# Half adder(HA)

- Adds 2, 1-bit numbers A and B , generated two outputs sum(S) and carry (C).



Expression for sum and carry :

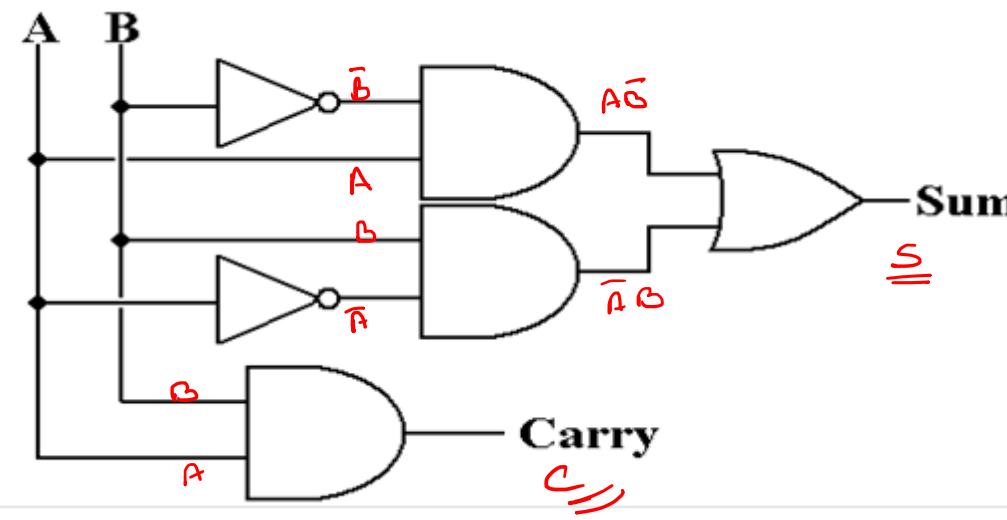
$$\begin{array}{r}
 \begin{array}{c}
 \begin{array}{ccccc}
 A & 0 & 0 & 1 & 1 \\
 + B & & & & \\
 \hline
 00 & 01 & 01 & 10
 \end{array} \\
 = \sum_m 1, 2 = \prod_m 0, 3
 \end{array} \\
 \begin{array}{c}
 C_{out} = \sum_m 3 = \prod_m 1, 2 \\
 \text{SOP} \quad \text{POS}
 \end{array}
 \end{array}$$



# HA circuit

---

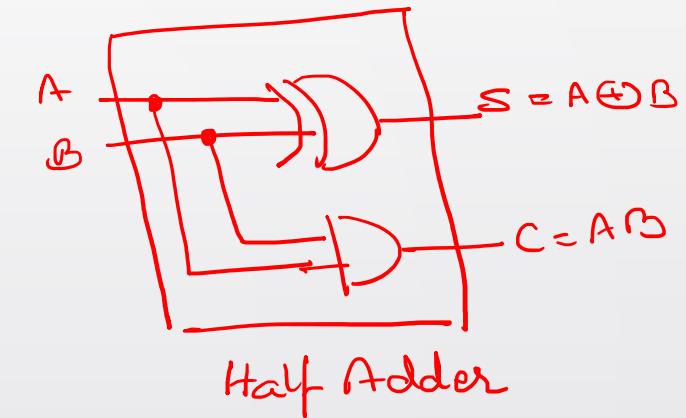
Using basic logic gates



$$S = \overline{A}B + A\overline{B}$$

$$C = AB$$

Using XOR and AND gate

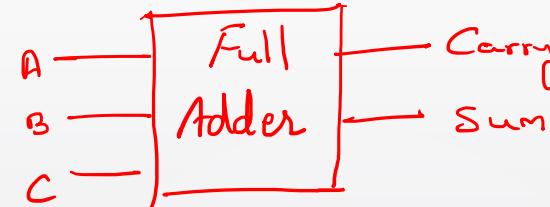


# Full adder

Truth Table

A B C	$m_0$	Carry	Sum
0 0 0	$m_0$	0	0
0 0 1	$m_1$	0	1
0 1 0	$m_2$	0	1
0 1 1	$m_3$	1	0
1 0 0	$m_4$	0	1
1 0 1	$m_5$	1	0
1 1 0	$m_6$	1	0
1 1 1	$m_7$	1	1

$$0+0+1 \rightarrow \\ \begin{array}{r} 00 \\ 01 \\ \hline 01 \end{array}$$



$$\text{Carry} = \sum_m 3, 5, 6, 7 = \prod_m 0, 1, 2, 4$$

$$\text{Sum} = \sum_m 1, 2, 4, 7 = \prod_m 0, 3, 5, 6.$$

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

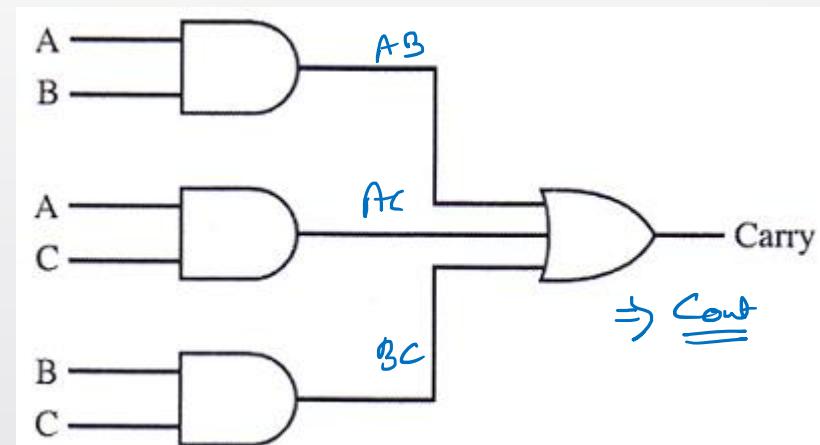
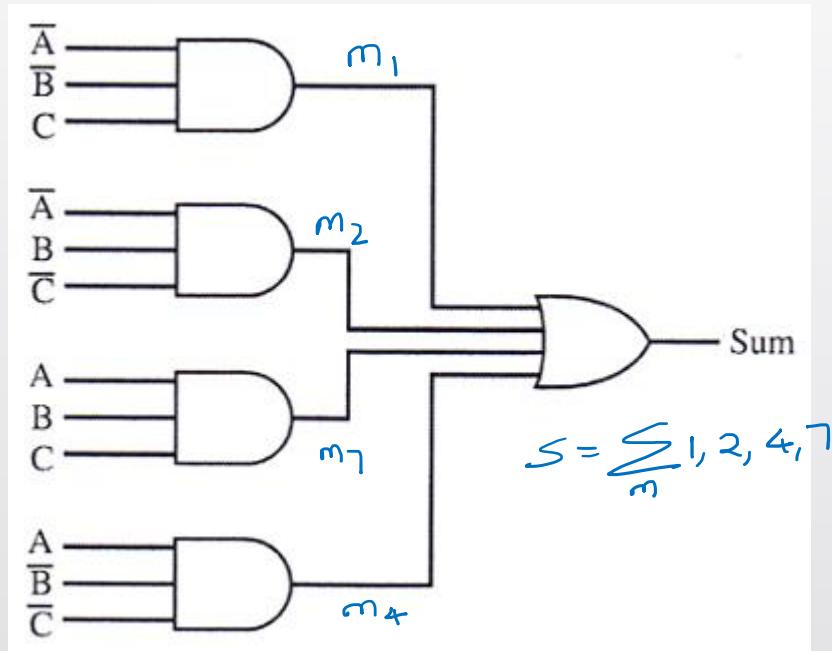
$$\begin{aligned} \text{Sum} &= \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC \\ &= \bar{A}(\bar{B}C + B\bar{C}) + A(\bar{B}\bar{C} + BC) \\ &= \bar{A}(B \oplus C) + A(\bar{B} \oplus C) \\ &= \underline{\underline{A \oplus B \oplus C}} \end{aligned}$$

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

$$\text{Carry} = \underline{\underline{BC + AC + AB}}$$

# FA circuit using basic logic gates

---

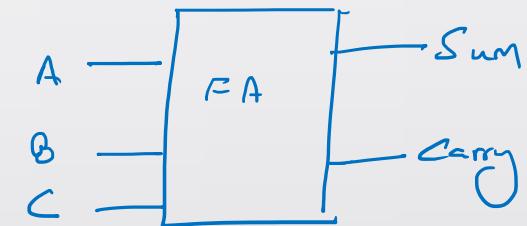
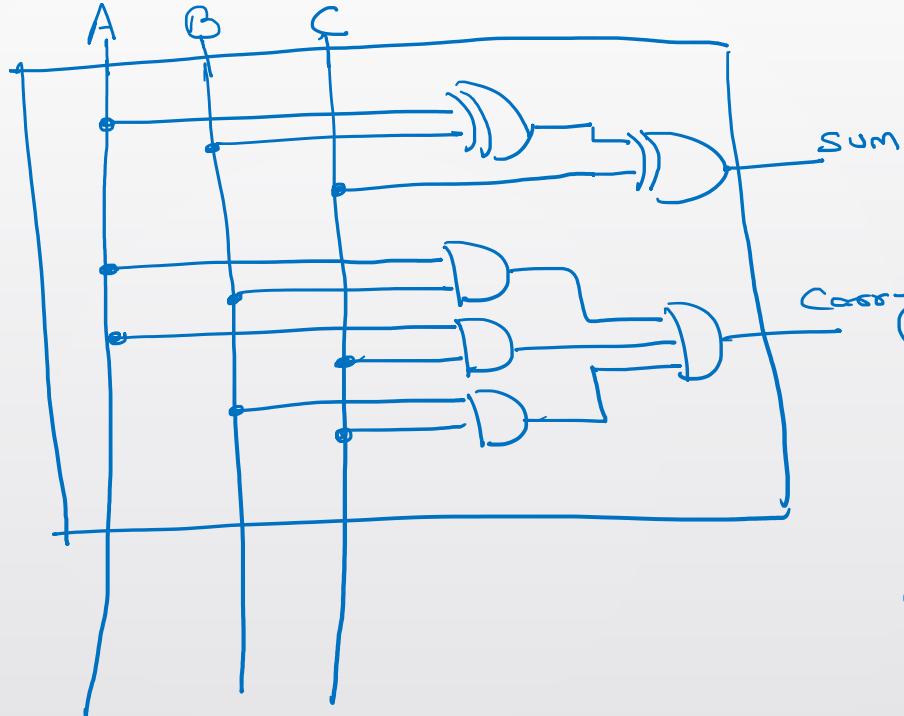


# Full adder circuit using XOR operations

FA will  $\rightarrow$  XOR

$$\text{Sum} = \underline{\underline{A \oplus B \oplus C}}$$

$$\text{Carry} = \underline{\underline{AB + AC + BC}}$$



# FA using 2 HA s and one external gate

Half Adder

$$S = A \oplus B$$

$$C = A \cdot B$$

Full Adder

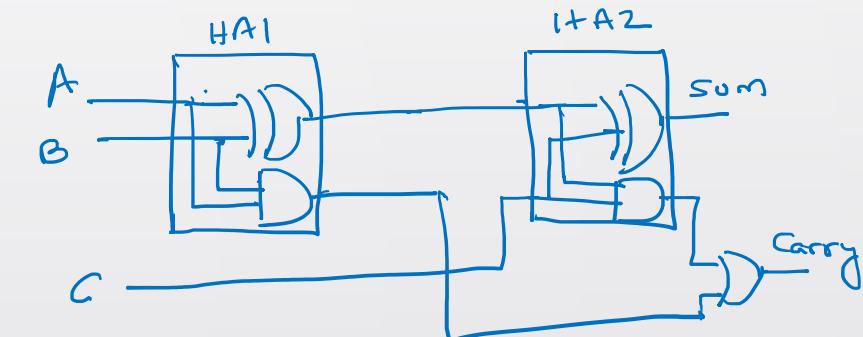
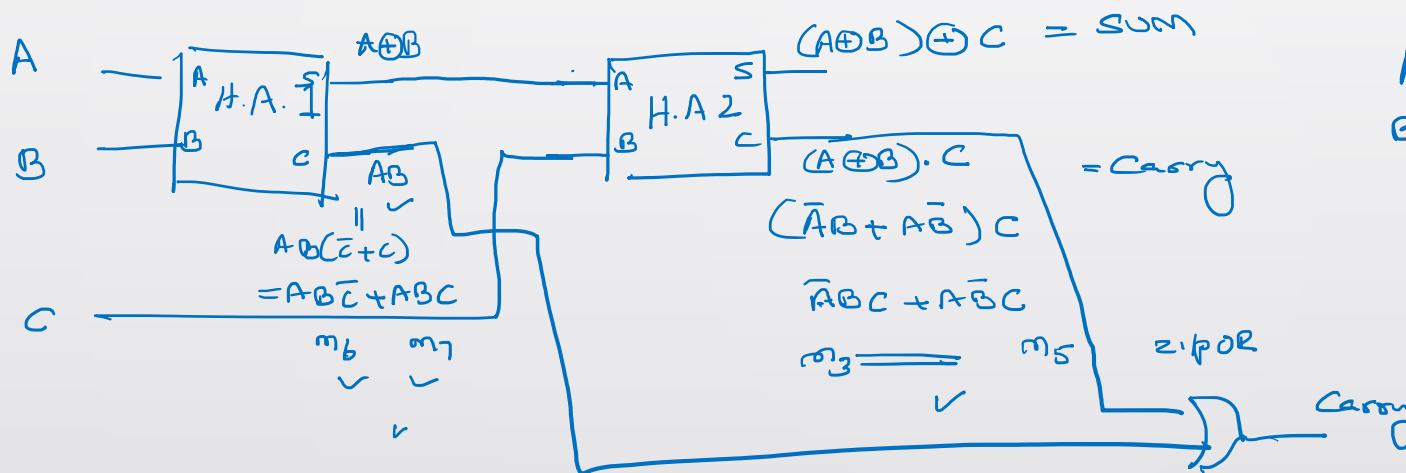
S of HA1

$$\text{sum} = (A \oplus B) \oplus C$$

$$\text{Carry} = \underline{\underline{AB + AC + BC}}$$

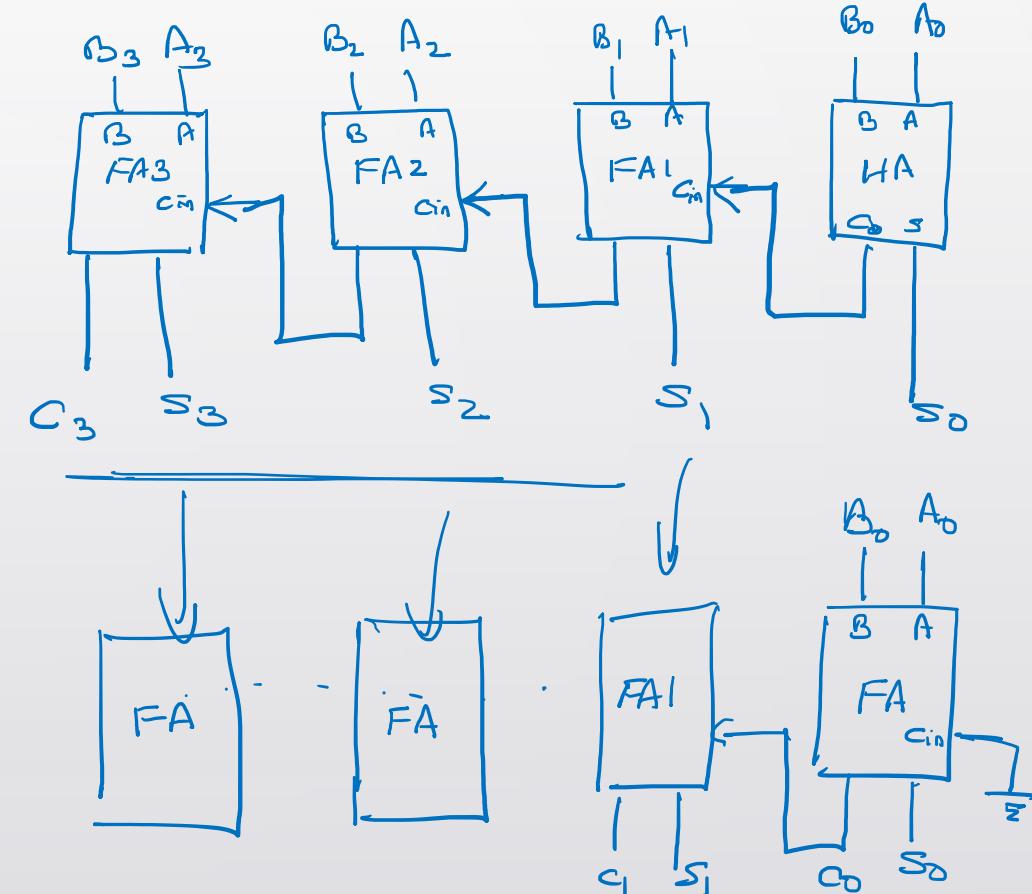
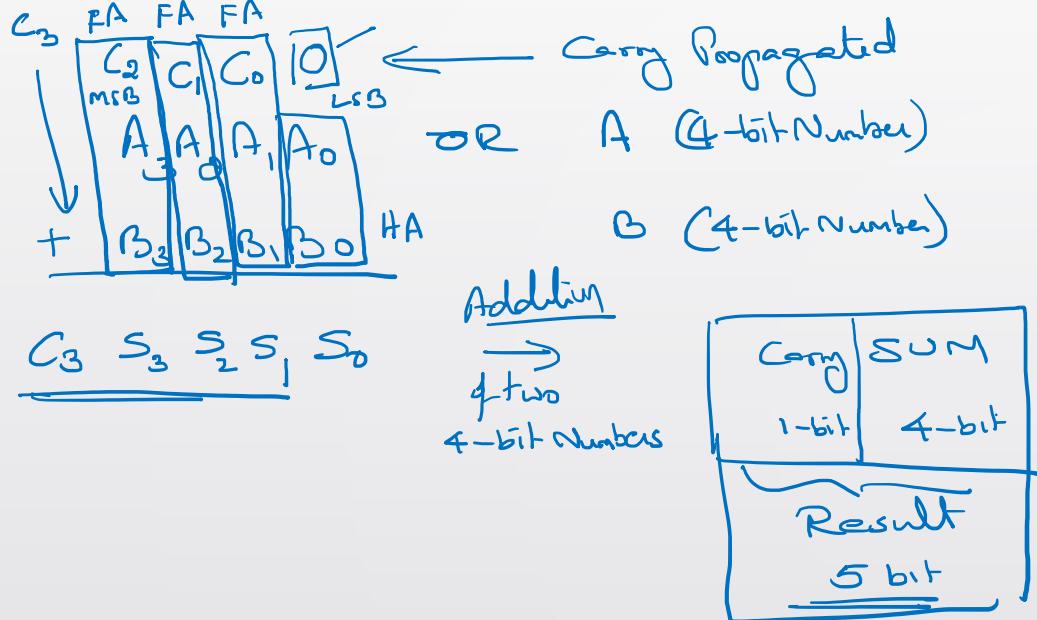
$$= \sum_m 3, 5, 6, 7$$

$$= (\overline{A}BC + A\overline{B}C) \stackrel{\text{OR}}{=} m_3 + (ABC + A\overline{B}C) \stackrel{\text{OR}}{=} m_6 + (AB\overline{C} + A\overline{B}C) \stackrel{\text{OR}}{=} m_7$$



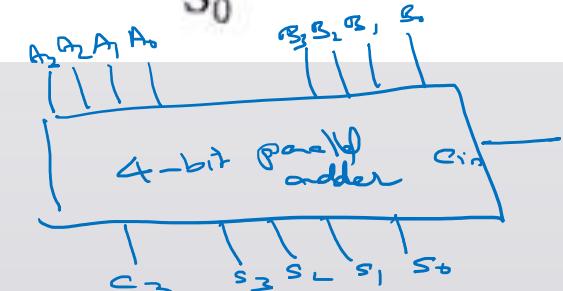
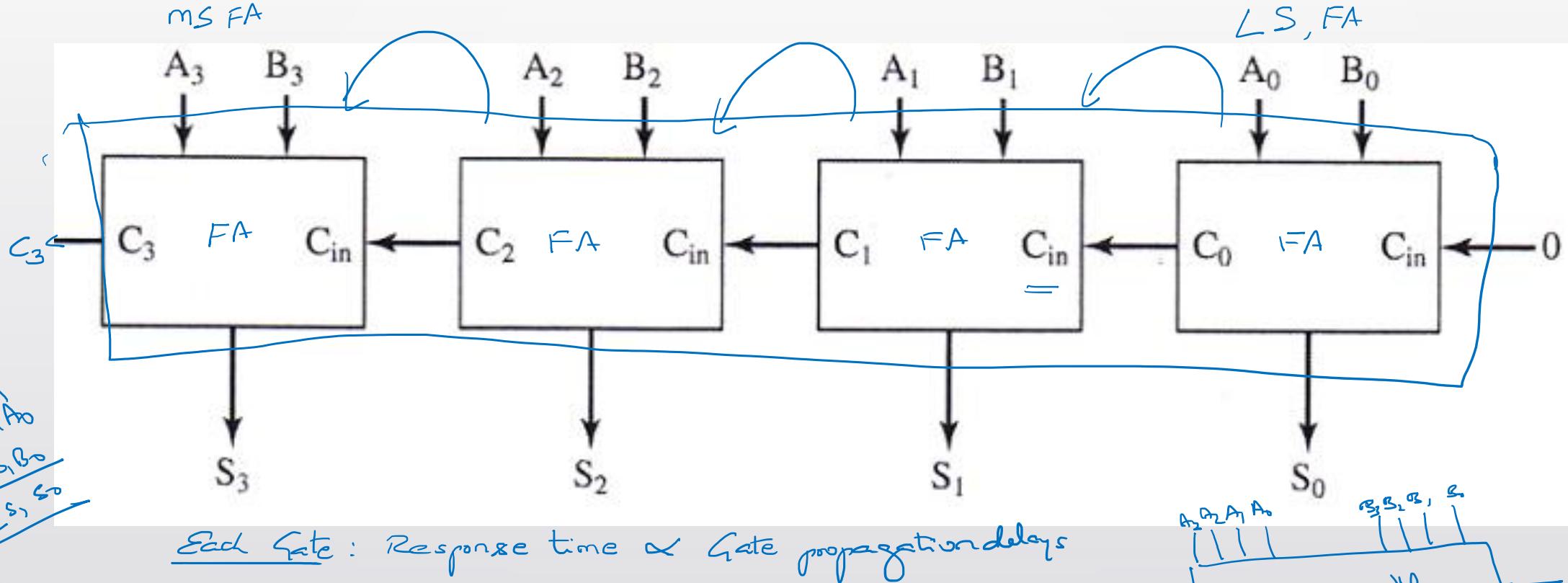
# 4-bit Parallel adder using FA blocks

- Consider addition of 2, 4-bit numbers:  $(A_3 A_2 A_1 A_0)$  and  $(B_3 B_2 B_1 B_0)$



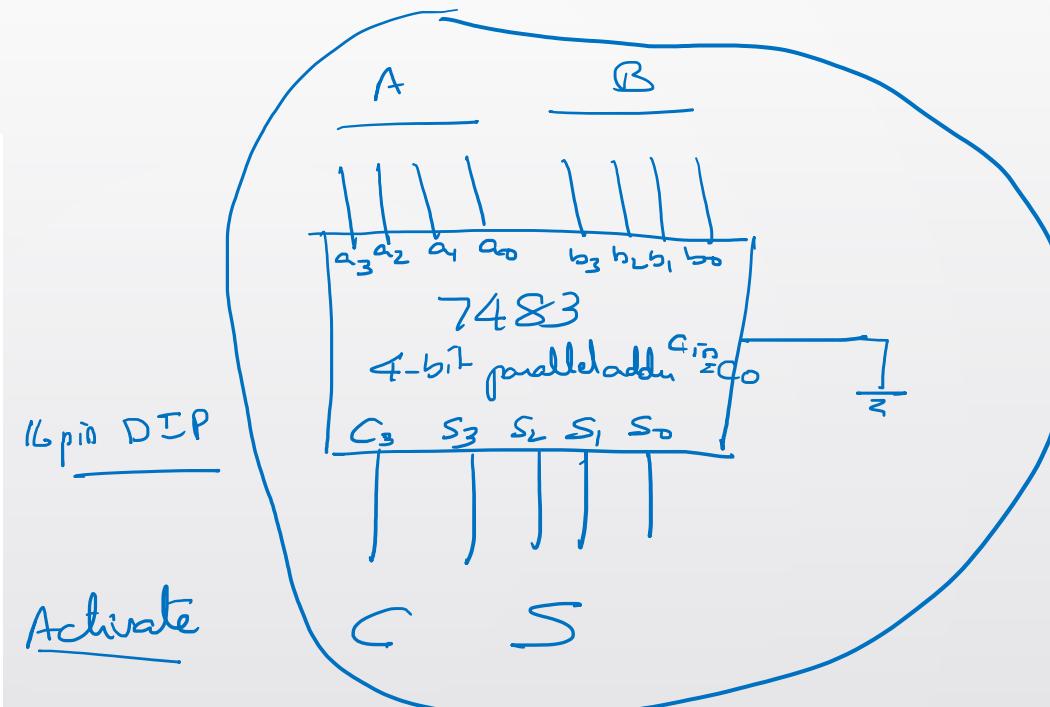
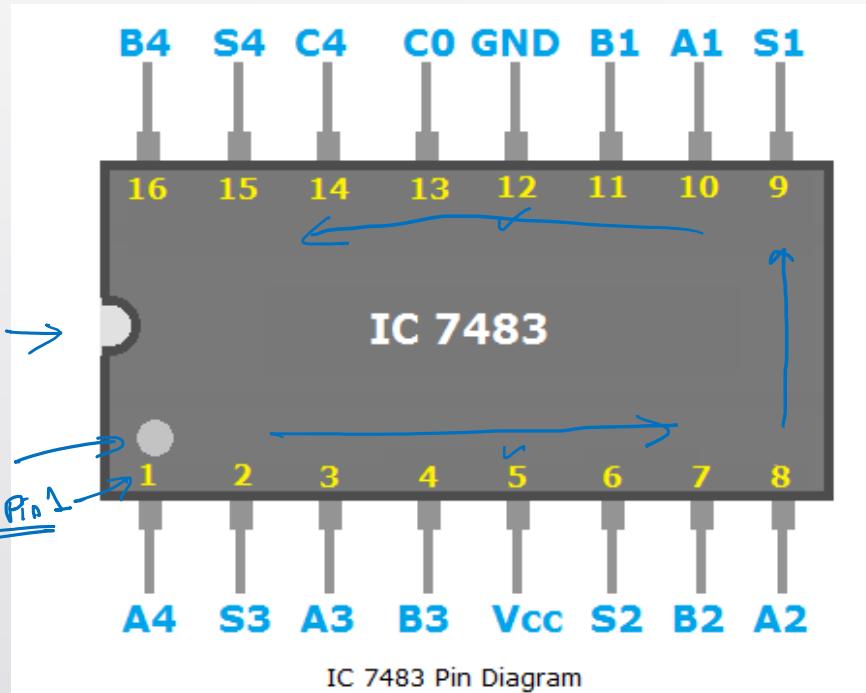
# 4-bit parallel adder

Also called as Carry Propagation Adder (CPA)



# 7483 IC : 4-BIT PARALLEL ADDER

---



# Half subtractor

- Write the truth table and circuit for half subtractor

$$\begin{array}{r}
 A \\
 -B \\
 \hline
 A-B
 \end{array}
 \quad \text{bit wise subtraction} \\
 A \rightarrow 1\text{-bit } 0 \text{ or } 1 \\
 B \rightarrow 1\text{-bit } 0 \text{ or } 1$$

$$\begin{array}{r}
 0 \\
 -0 \\
 \hline
 00
 \end{array}
 \quad
 \begin{array}{r}
 1 \\
 0 \\
 \hline
 11
 \end{array}
 \quad
 \begin{array}{r}
 1 \\
 -0 \\
 \hline
 01
 \end{array}
 \quad
 \begin{array}{r}
 1 \\
 -1 \\
 \hline
 00
 \end{array}$$

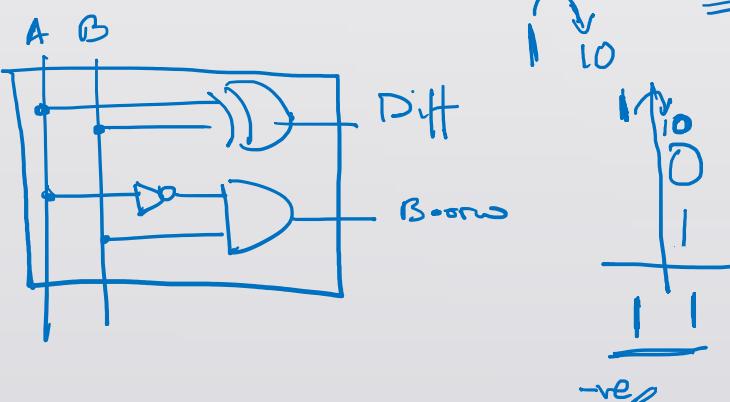
$$\dots$$

$$2^2$$

$$2^2$$

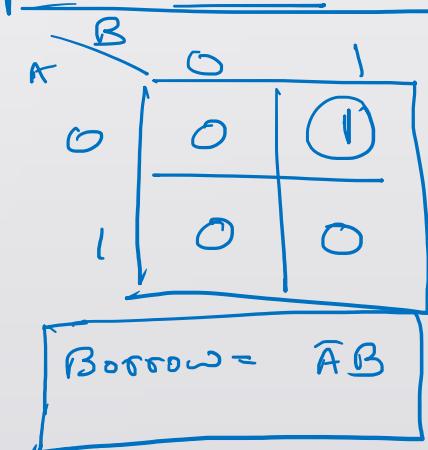
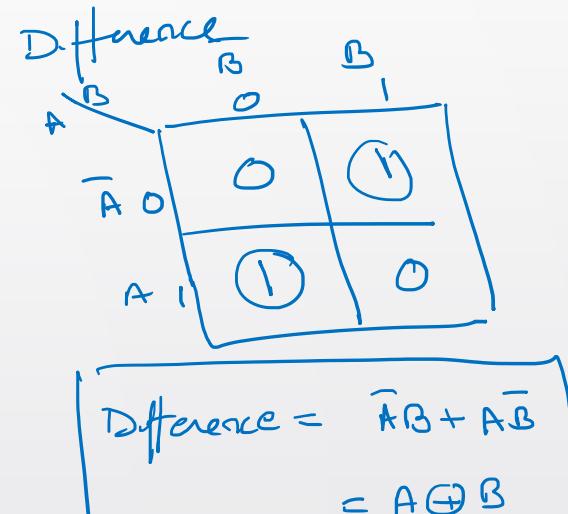
$$10$$

$$10$$



	A	B	Difference	Borrow
m <sub>0</sub>	0	0	0	0
m <sub>1</sub>	0	1	1	1
m <sub>2</sub>	1	0	1	0
m <sub>3</sub>	1	1	0	0

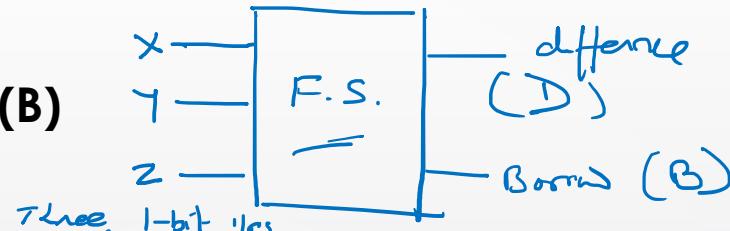
$$\begin{aligned}
 \text{Difference} &:= \sum_{m=0,1,2} 1 = \overline{\prod}_{m=0,1,2} 0, 3 \\
 \text{Borrow} &:= \sum_{m=0,1,2} 1 = \overline{\prod}_{m=0,1,2} 0, 2, 3
 \end{aligned}$$



# Full subtractor

**DIFFERENCE (D) = X-Y-Z, Borrow (B)**

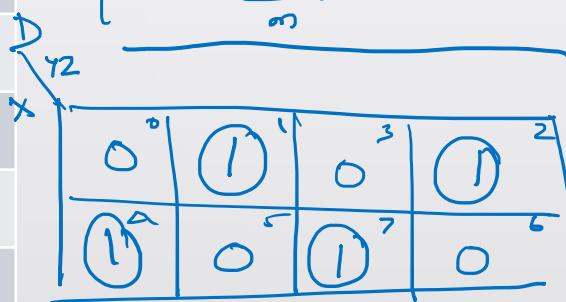
X	Y	Z	D	B
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1



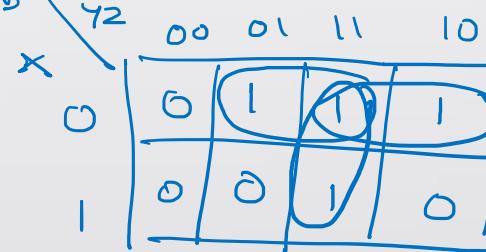
Expressions for D and B:

$$D = \sum_m 1, 2, 4, 7 = \prod_m 0, 3, 5, 6$$

$$B = \sum_m 1, 2, 3, 7 = \prod_m 0, 4, 5, 6$$



$$D = \overline{x}yz + \overline{x}y\bar{z} + x\bar{y}\bar{z} + xy\bar{z}$$



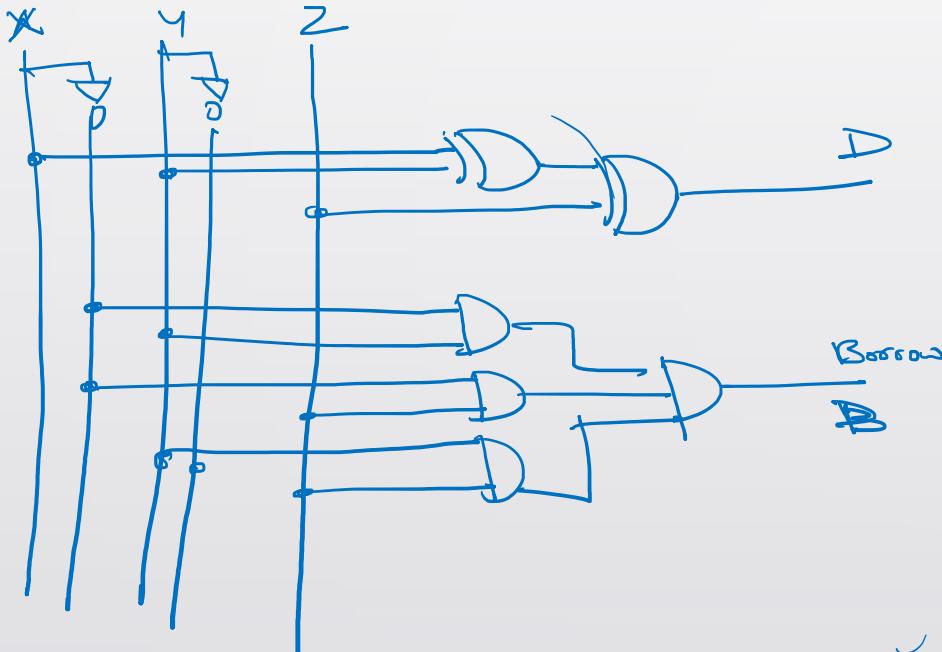
$$B = \overline{x}z + \overline{x}y + yz$$

# FS circuit

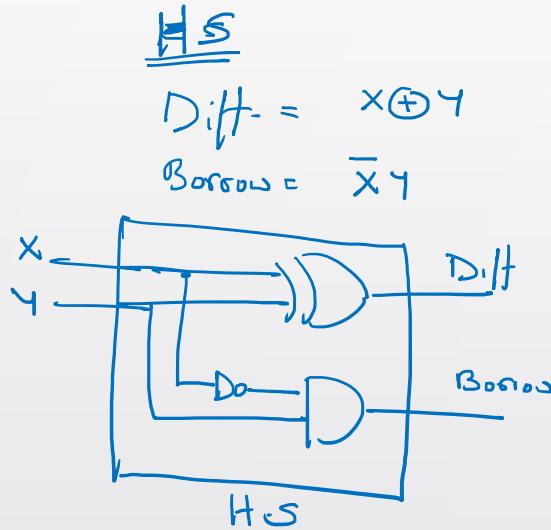
- Draw the circuit for FS using
  - (i) basic logic gates only
  - (ii) XOR and basic logic gates

$$D = X \oplus Y \oplus Z = \sum_{S=1}^7 = \overline{X} \overline{Y} Z + \overline{X} Y \overline{Z} + X \overline{Y} \overline{Z} + X Y Z$$
$$B = \overline{X} Y + \overline{X} Z + Y Z$$

PRACTICE



# Full subtractor using 2 HS s and one external gate



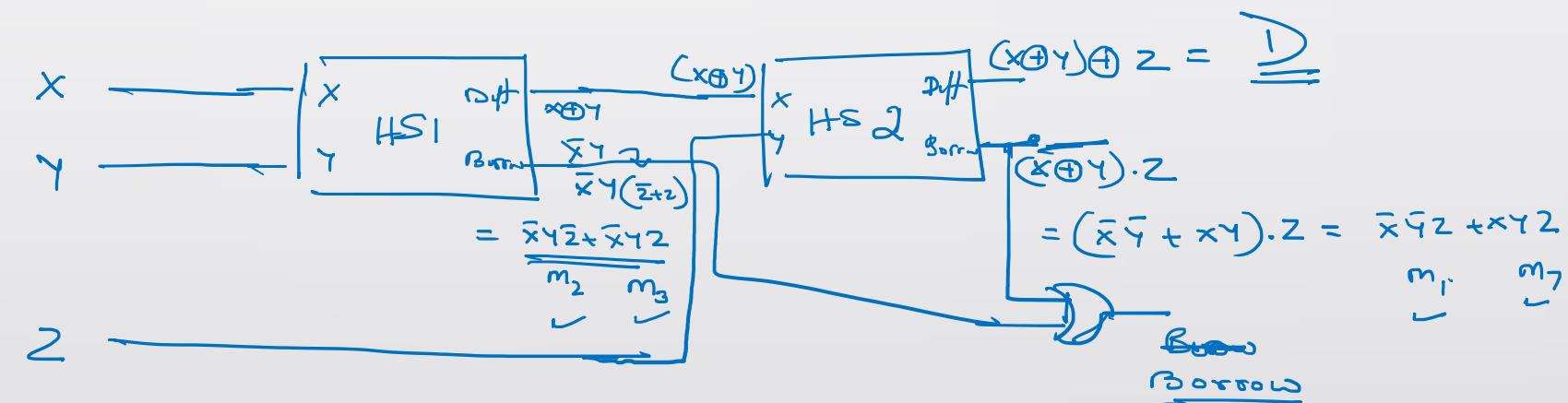
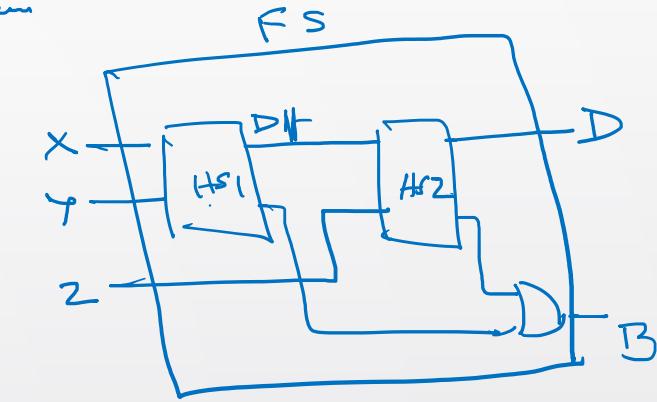
F.S

$$D = x \oplus y \oplus 2 = (x \oplus y) \oplus 2$$

Dif term

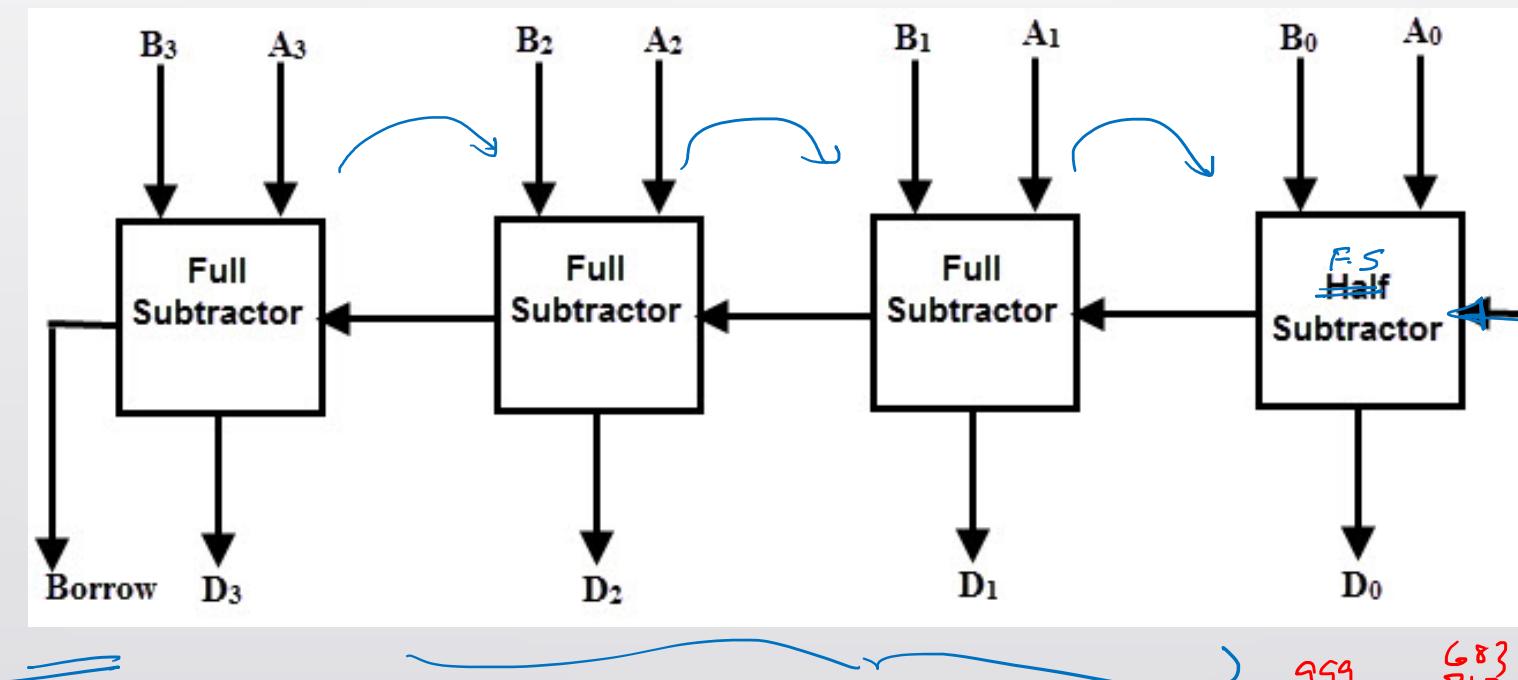
HS1

$$B = \sum_{m=1}^7 = (m_1) + (m_2) + (m_3) + (m_7)$$



# 4-bit parallel subtractor using FS blocks

Consider subtraction of 2, 4-bit numbers: (A<sub>3</sub> A<sub>2</sub> A<sub>1</sub> A<sub>0</sub>) and (B<sub>3</sub> B<sub>2</sub> B<sub>1</sub> B<sub>0</sub>)



$$\begin{array}{r} \text{A}_3 \text{A}_2 \text{A}_1 \text{A}_0 \\ - \text{B}_3 \text{B}_2 \text{B}_1 \text{B}_0 \\ \hline \end{array}$$

$$\begin{array}{r} 1246 \\ 683 \\ \hline -563 \\ \hline 8001 \end{array}$$

$$\begin{array}{r} 1000 \\ 1111 \\ \hline 001 \\ \hline 246 \\ -683 \\ \hline 316 \\ \hline 0563 \\ \hline 437 \end{array}$$

# Subtraction using complements

use 4-bit parallel adders or use 7483

## Using 2's complement method

### Using 1's complement method

Example

$$\begin{array}{r} 8 \\ -2 \\ \hline +6 \end{array}$$

Let us use 2's complement addition

$$\begin{array}{r} 1000 \\ 0010 \\ \hline 1000 \\ 1101 \\ \hline 10110 \end{array}$$

2's complement +1

the sign of the answer  
retain only sum terms  $\rightarrow$  Difference

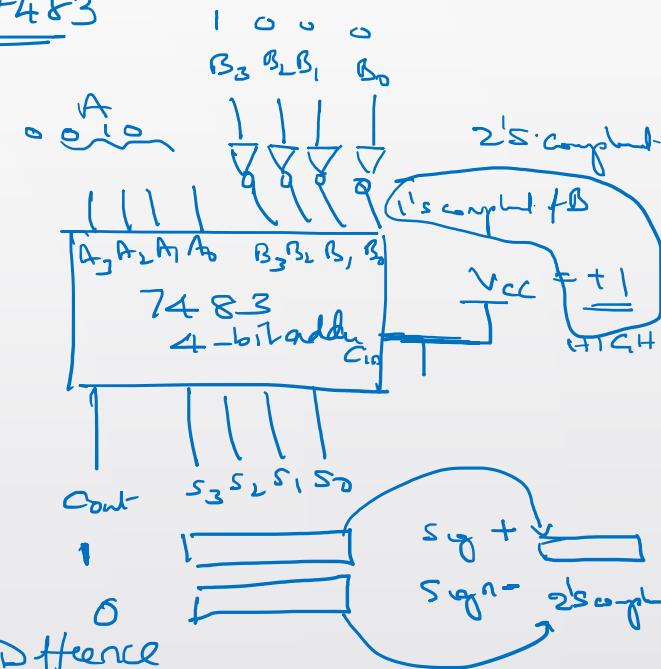
Example

$$\begin{array}{r} 2 \\ -8 \\ \hline -6 \end{array}$$

Retain only sum terms  $\rightarrow$  Difference

$$\begin{array}{r} 0010 \\ 1000 \\ \hline 0111 \\ 0010 \\ \hline 01010 \\ 01010 \\ \hline \end{array}$$

2's comp  
1's comp +1  
2's complement



$$1010 \rightarrow \begin{array}{r} 001 \\ 010 \\ \hline 0110 \end{array}$$

# Subtraction using complements

Using 2's complement method

## Using 1's complement method

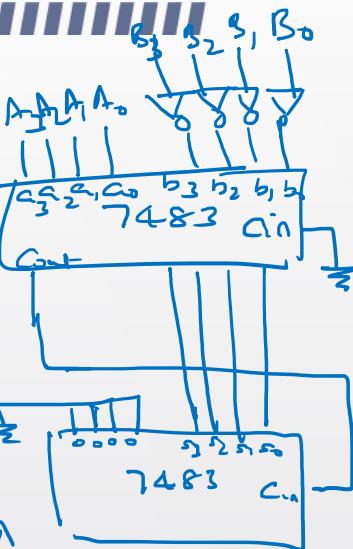
$$\begin{array}{r} 8 \\ - 2 \\ \hline \end{array} \quad \begin{array}{r} 1000 \\ - 0010 \\ \hline \end{array} \quad \begin{array}{r} 1000 \\ 1101 \\ \hline 10101 \\ \text{Carry +ve sign} \\ \hline \end{array}$$
  
$$\begin{array}{r} -8 \\ - 2 \\ \hline \end{array} \quad \begin{array}{r} 0000 \\ 1000 \\ \hline \end{array}$$

$$\begin{array}{r} 2 \\ - 8 \\ \hline \end{array} \quad \begin{array}{r} 0010 \\ - 1000 \\ \hline \end{array} \quad \begin{array}{r} 011 \\ 0010 \\ 0111 \\ \hline 01001 \\ \text{Ans} \end{array}$$

$$\begin{array}{r} 0101 \\ + 1 \\ \hline 0110 \\ \text{Sum} \\ \text{Add Carry to Least bit} \\ \text{also use carry to add to sum} \end{array}$$

Carry Or Overflow  
is 0  
-ve number

Ans is in 1's  
complement form



$$1001 \xrightarrow{\text{Ans}} 0110 = 6$$



**Questions?**



# Binary adders and subtractors

- Half adder, full adder, parallel adder
- Half subtractor , full subtractor, parallel subtractor
- Subtraction using complements, parallel adder/subtractor
- Carry Look ahead adder, Decimal adder

# Subtraction using complements

Using 2's complement method

Using 1's complement method

(1)



$$\begin{array}{r} 1000 \\ -0010 \end{array}$$

$$\begin{array}{r} 1000 \\ 1's\ copl \quad 10 \\ 2's\ comp \end{array}$$

$$\begin{array}{r} 1000 \\ +110 \\ \hline 10110 \end{array}$$

$$\begin{array}{r} A_3 A_2 A_1 A_0 \\ - B_3 B_2 B_1 B_0 \end{array}$$

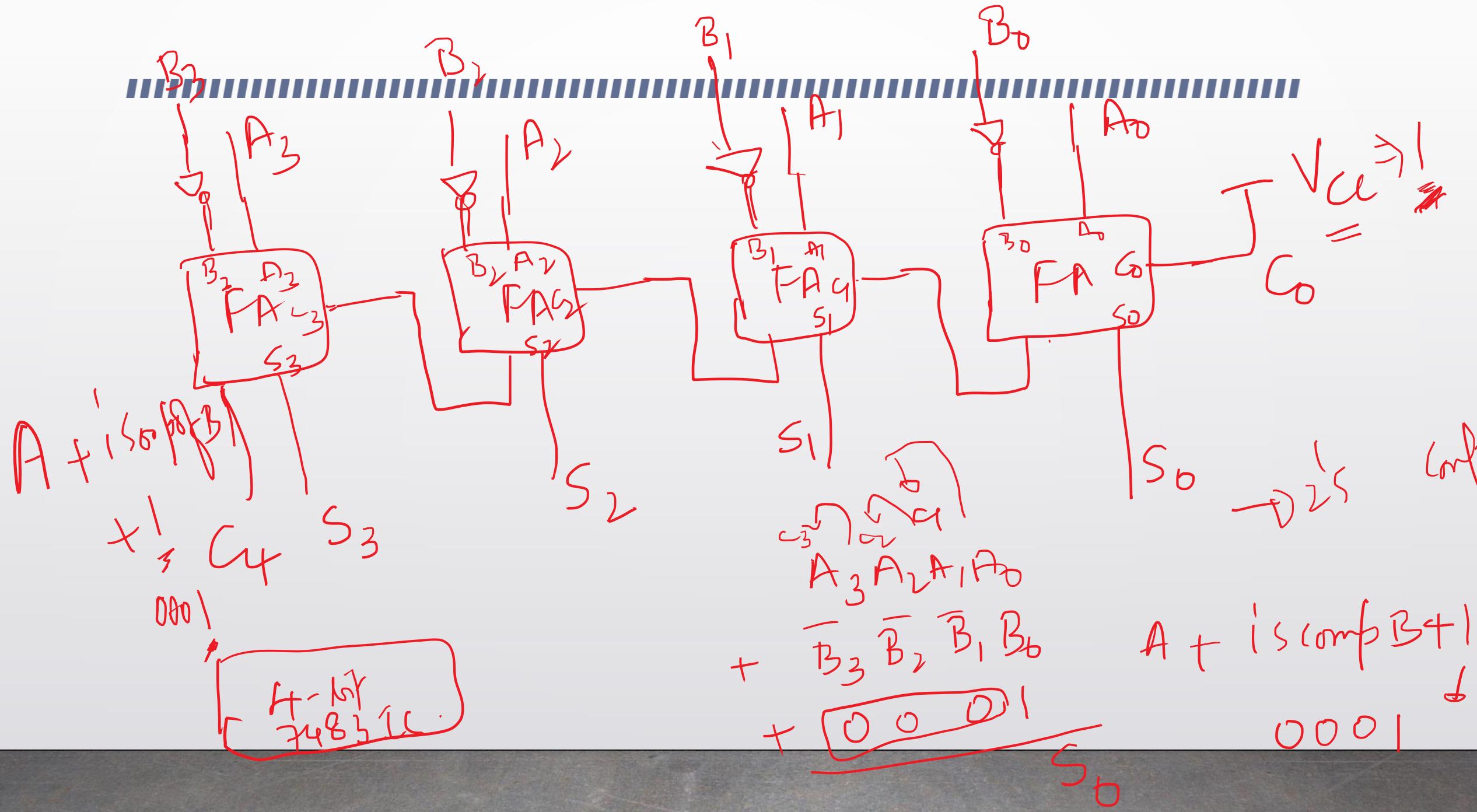
$$\begin{array}{r} 2 \\ -8 \\ \hline -6 \end{array}$$

$$\begin{array}{r} 0010 \\ -1000 \end{array}$$

$$\begin{array}{r} 11 +ve \\ \rightarrow 0010 \\ 0111 \end{array}$$

$$\begin{array}{r} 1010 \\ \hline 0101 \end{array}$$

$$\begin{array}{r} -6 \\ -6 \\ \hline 0101 \end{array}$$



# ~~1's Complement Subtraction~~

$$\begin{array}{r} 1000 \\ -0010 \\ \hline 1101 \end{array}$$

Annotations: A red arrow points from the top '1' of '1000' to the top '1' of '1101'. A red circle highlights the 4th column ('1'). A red bracket labeled '5 + 1' is shown below the 4th column. A red arrow labeled '1' points from the bottom '1' of '1101' to the bottom '1' of the result.

$$\begin{array}{r} 0010 \\ -1000 \\ \hline 0110 \end{array}$$

Annotations: A red arrow points from the top '0' of '0010' to the top '0' of '0110'. A red circle highlights the 4th column ('0'). A red bracket labeled '6' is shown below the 4th column. A red arrow labeled '1' points from the bottom '0' of '0110' to the bottom '0' of the result.

$$\begin{array}{r} 0111 \\ -0101 \\ \hline 1010 \end{array}$$

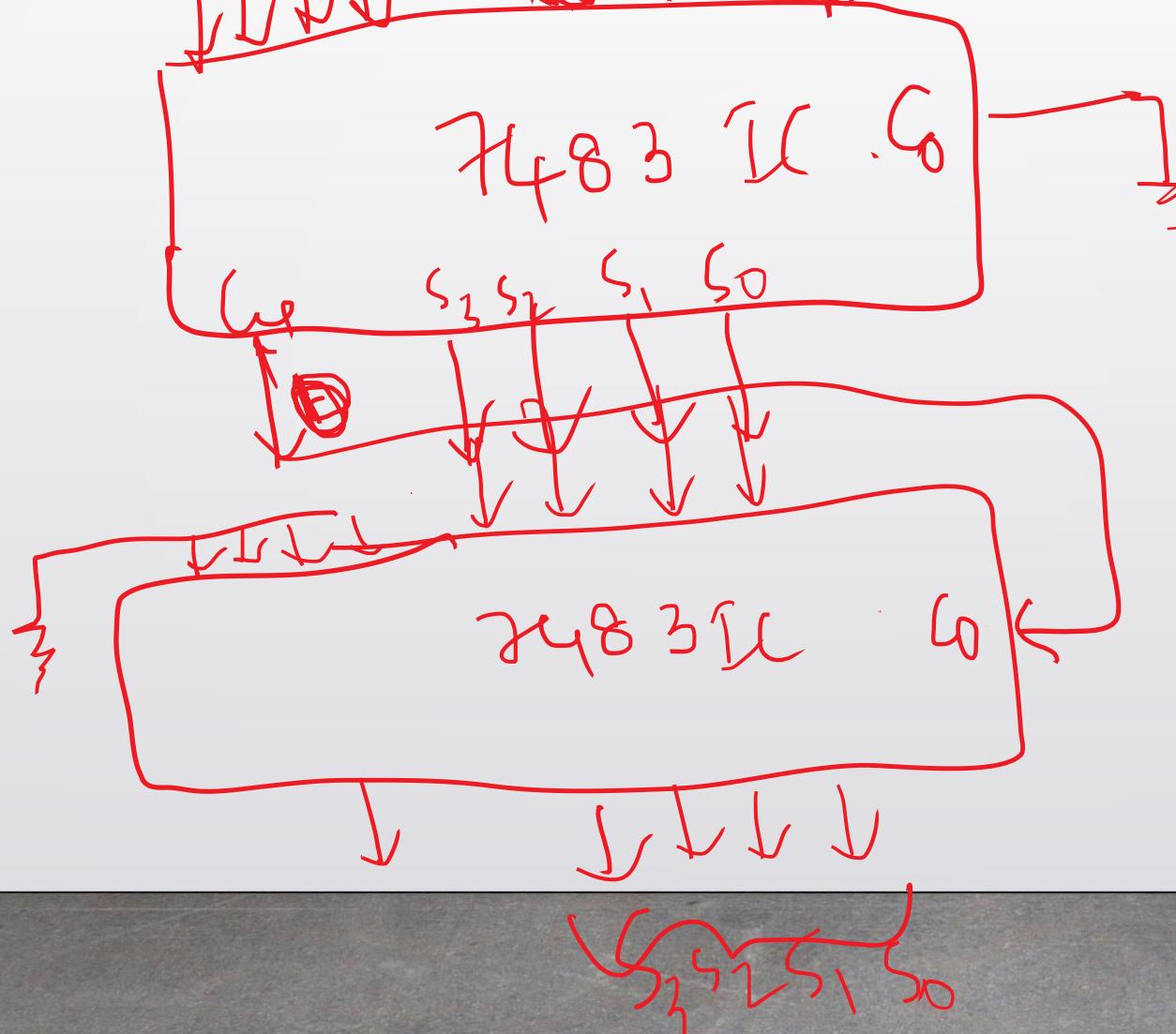
Annotations: A red arrow points from the top '0' of '0111' to the top '1' of '1010'. A red circle highlights the 4th column ('1'). A red bracket labeled '1' is shown below the 4th column. A red arrow labeled '1' points from the bottom '0' of '1010' to the bottom '0' of the result.

$$\begin{array}{r} 0101 \\ -0111 \\ \hline 1000 \end{array}$$

Annotations: A red arrow points from the top '0' of '0101' to the top '1' of '1000'. A red circle highlights the 4th column ('0'). A red bracket labeled '1' is shown below the 4th column. A red arrow labeled '1' points from the bottom '0' of '1000' to the bottom '0' of the result.

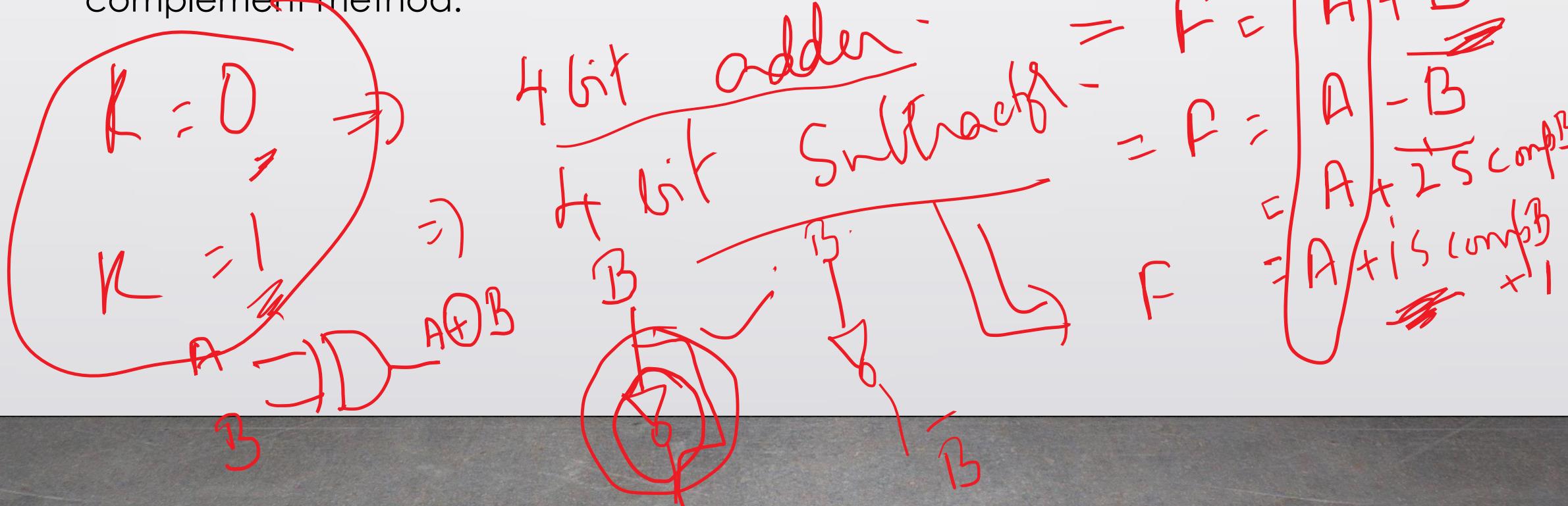
7483 IC

B<sub>3</sub> B<sub>2</sub> B<sub>1</sub> B<sub>0</sub> A<sub>3</sub> A<sub>2</sub> A<sub>1</sub> A<sub>0</sub>

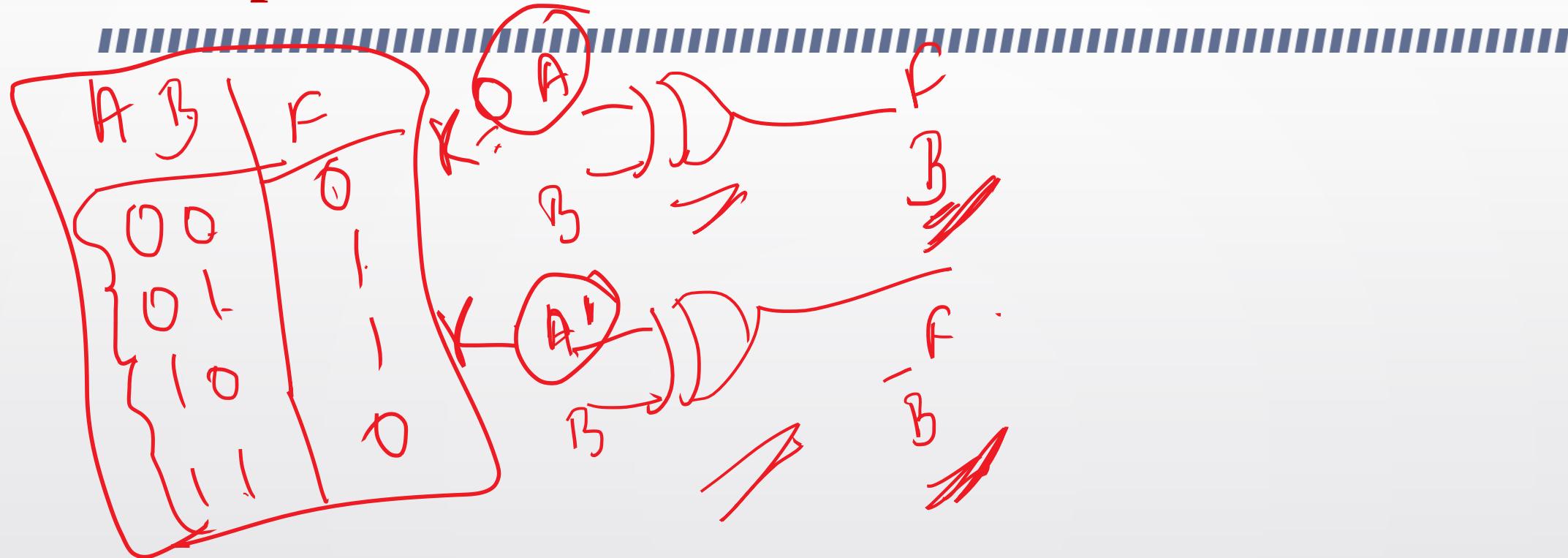


## 4-bit parallel adder/subtractor :

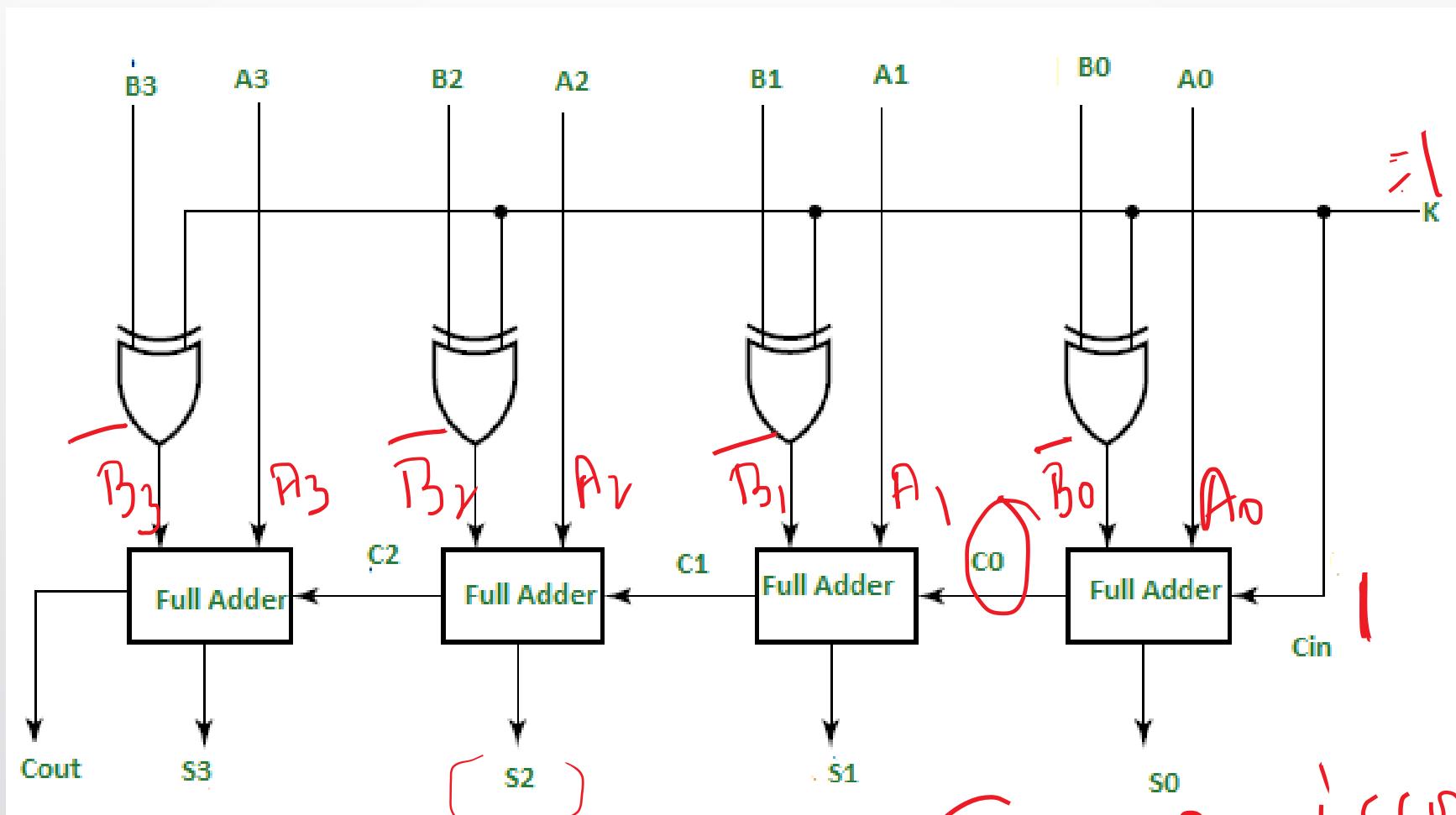
- Design a 4-bit adder/ subtractor using FA blocks or 7483 IC and minimum external gates, i.e. if the control input bit  $K=0$ , the circuit should add the input numbers or if  $K=1$ , the circuit should subtract the two numbers using 2's complement method.
- Note: Unless and otherwise mentioned assume subtraction to be using 2's complement method.



## 4-bit parallel adder/subtractor :



# 4-bit parallel adder/subtractor :



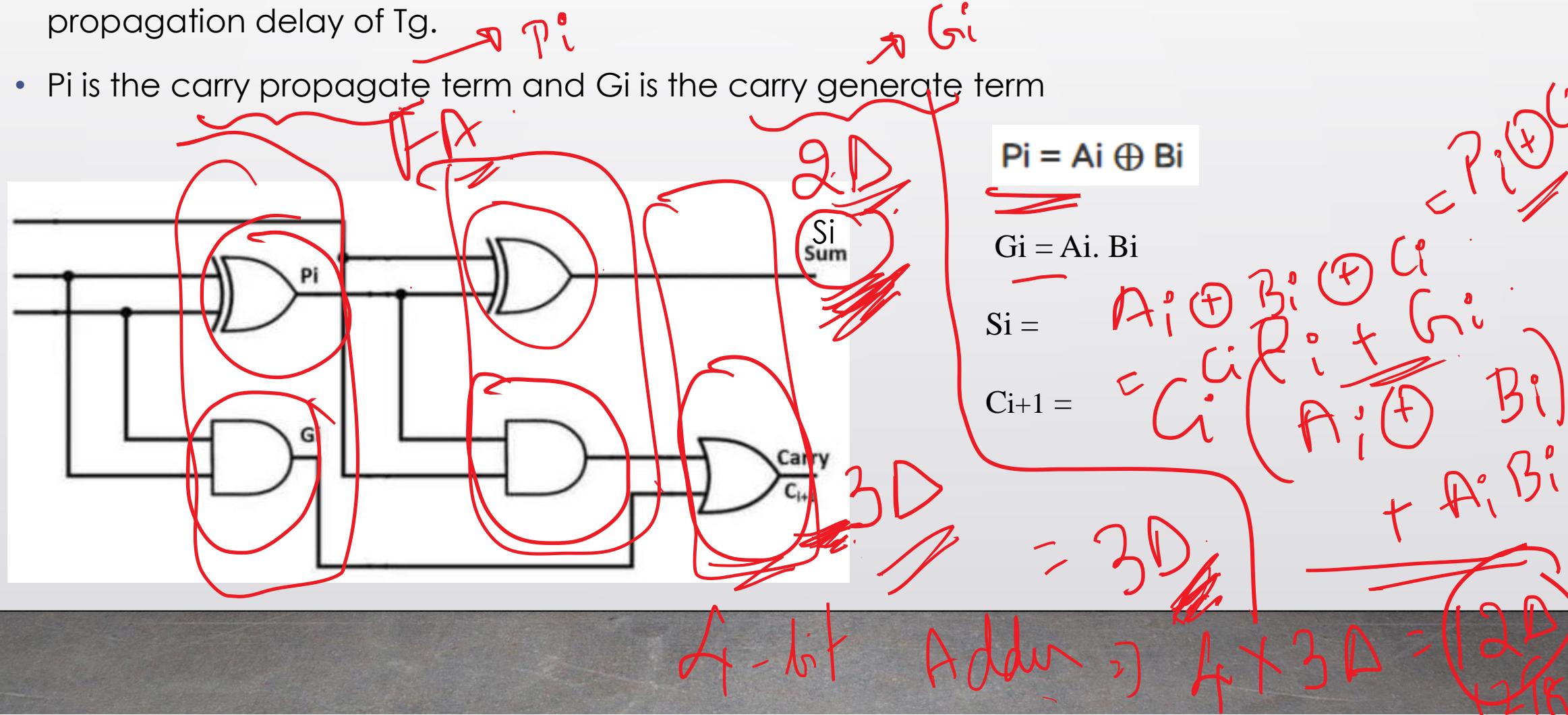
$$S = A + i \text{ swap } B + i - A - B$$

# Carry Look Ahead (CLA) Adder

[Parallel Adder]

[Fast Adder]

- Propagation delay in Full Adder is  $3T_g$  with respect to following circuit, where  $T_g$  is the propagation delay of a gate. All the gates are assumed to have a propagation delay of  $T_g$ .
- $P_i$  is the carry propagate term and  $G_i$  is the carry generate term



# CLA continued

## 4-bit adder

- Carry generation in CLA from  $A_i$ ,  $B_i$ , and  $C_0$

- $C_0 = \text{input carry}$

$$C_1 =$$

$$C_0 P_0 + G_0 = C_0(A_0 \oplus B_0) + A_0 B_0$$

$$C_2 =$$

$$G_1 P_1 + G_1 = (C_0 P_0 + G_0) P_1 + G_1$$

$$C_3 =$$

$$G_2 + P_1 G_0 + P_1 P_0 C_0$$

$$C_2 P_2 + G_2 = (G_1 + P_1 G_0 + P_1 P_0 C_0) P_2 + G_2$$

$$= G_2 + P_2 G_1 + P_1 P_2 G_0$$

$$G_4 = G_3 + P_3 G_2 + P_2 P_3 G_1 + P_1 P_2 P_3 G_0$$

$$+ P_0 P_1 P_2 P_3 C_0$$

# CLA Continued

---

- Expressions for sum

$$S_0 = A_0 \oplus B_0 \oplus C_0 = P_0 \oplus C_0$$

$$S_1 = A_1 \oplus B_1 \oplus C_1$$

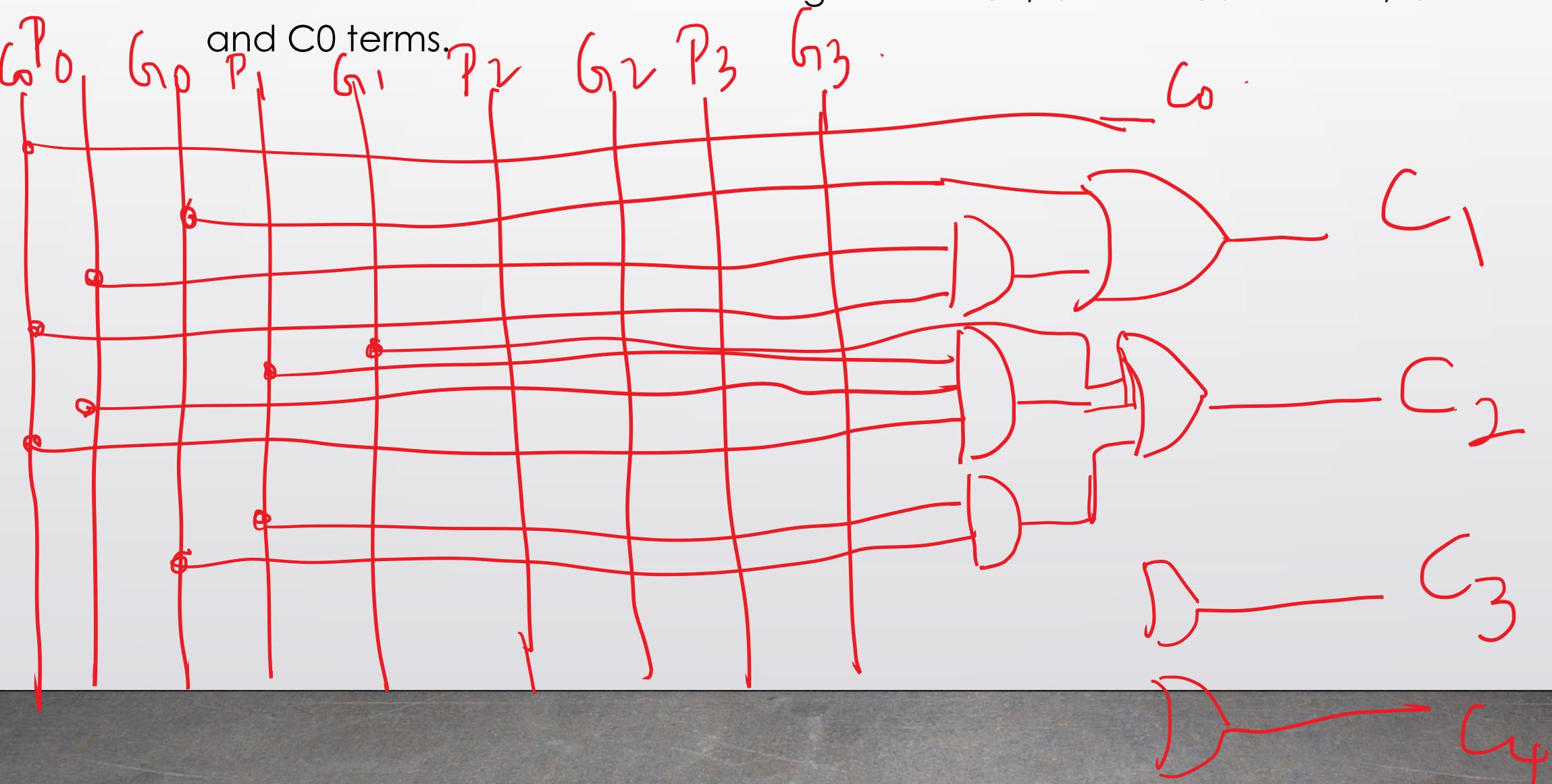
$$S_2 = A_2 \oplus B_2 \oplus C_2$$

$$S_3 = A_3 \oplus B_3 \oplus C_3$$

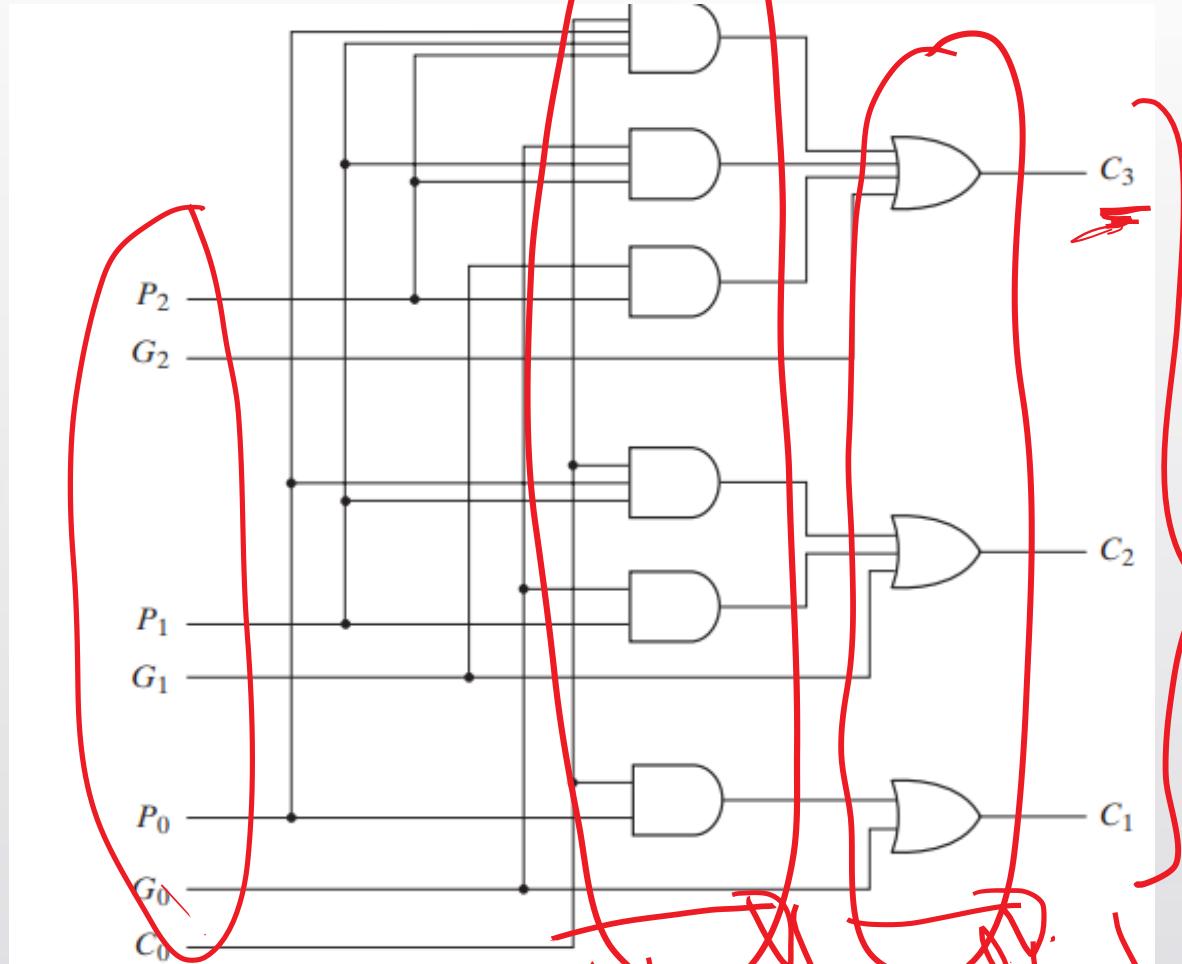
C4 =

# CLA: Carry look ahead generator circuit

- Draw the combinational circuit to generate  $C_1, C_2$  and  $C_3$  from  $P_i, G_i$  and  $C_0$  terms.



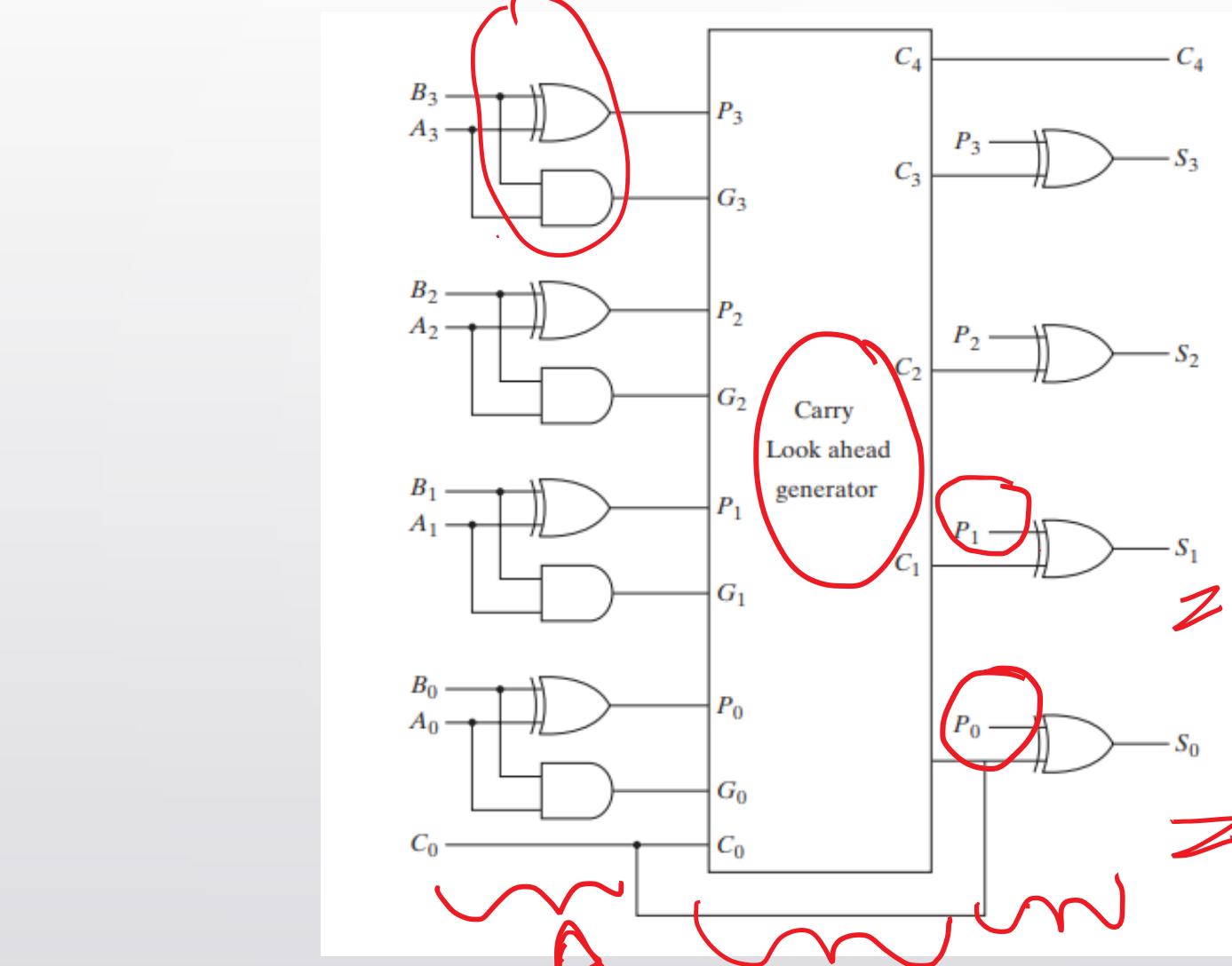
# CLA: Carry look ahead generator circuit



Time required to generate  $C_1, C_2, C_3$   
in terms of  $T_g$ ?

2D of  $2^T_g$  for  $C_3$ ,  
 $C_2$ ,  $C_1$ ,  $C_0$

# 4-bit CLA



Time required to generate  
S1,S2,S3,C4 in terms of Tg?

$T_g + 2T_g + T_g$   
 $T_g - T_g + T_g$   
 $T_g + T_g + T_g$

# Comparison

---

- CLA or CPA...which is better?

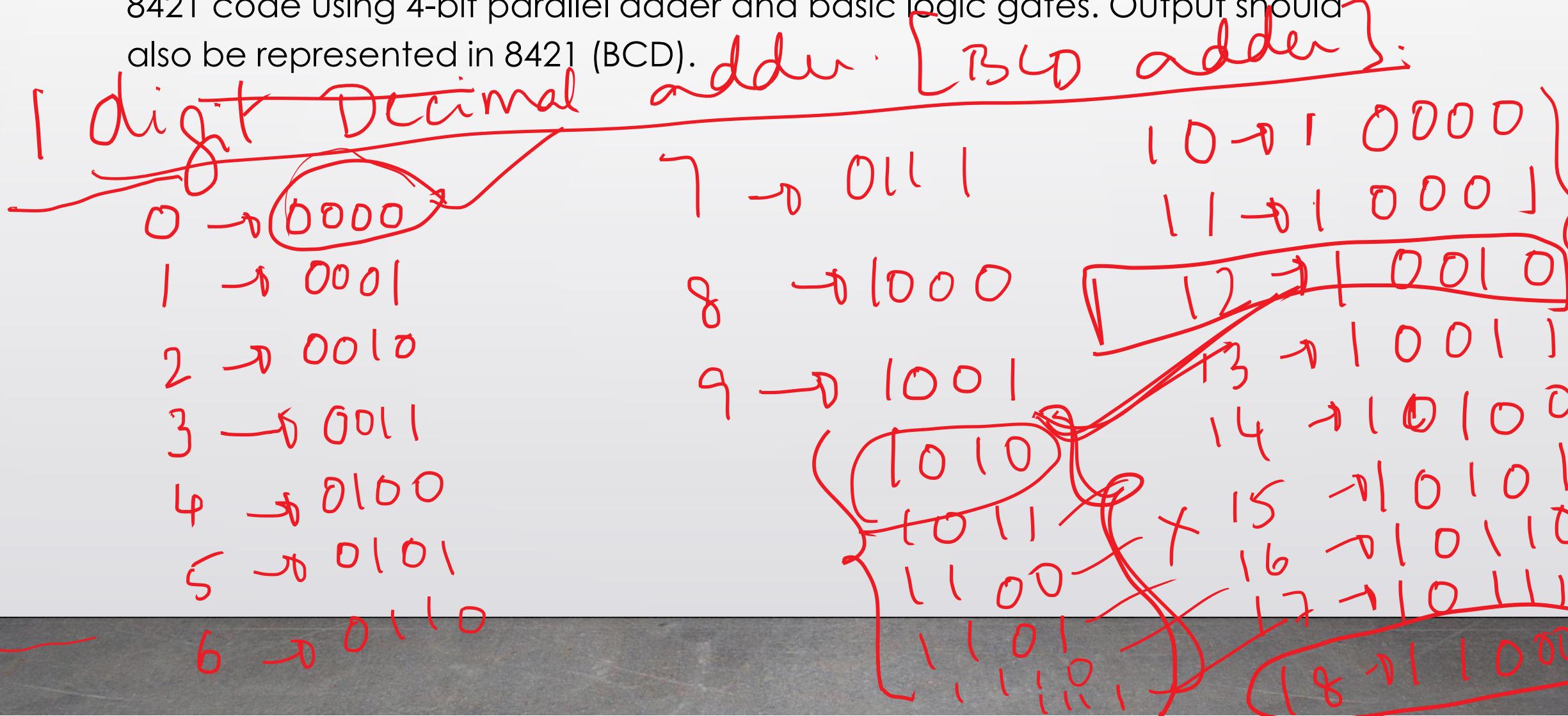
4-bit CLA is 3 times faster than  
4-bit Ripple Carry Adder

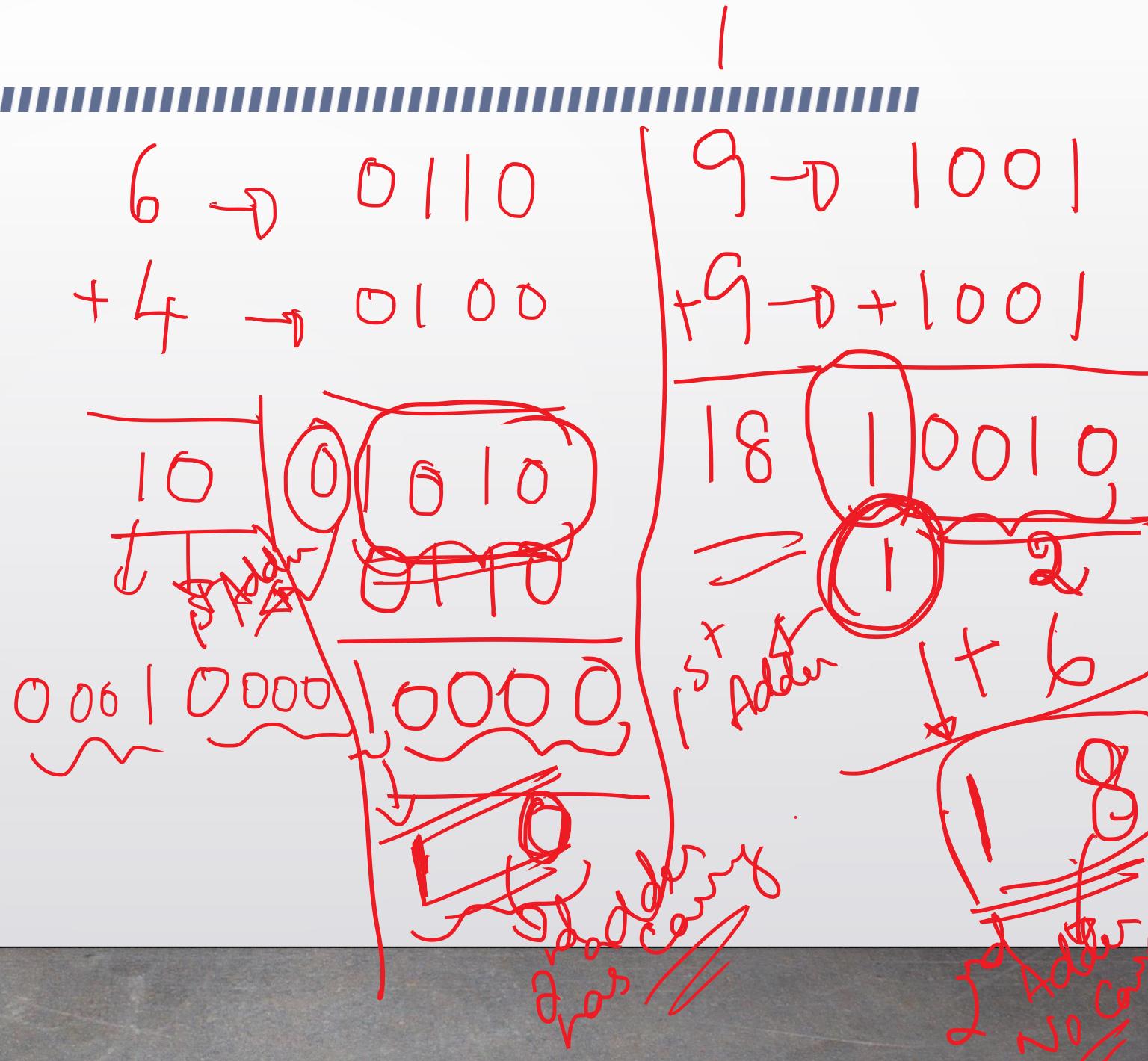
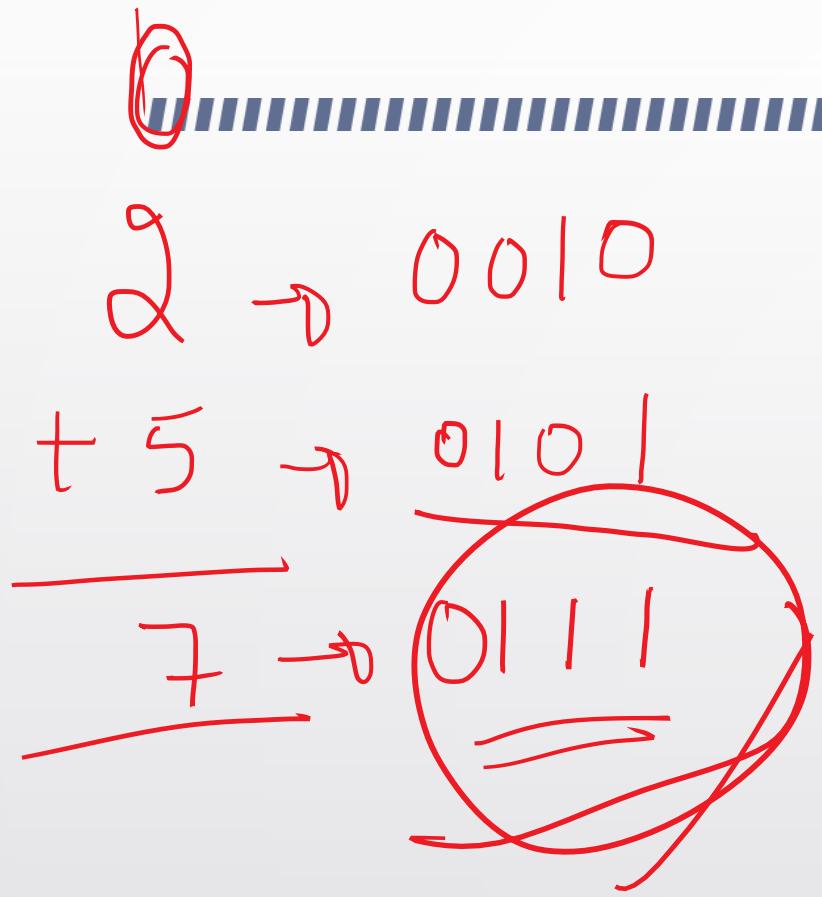
BCD adder

**Decimal adder:** Used to add decimal numbers  
represented in binary coded form

# Decimal adder:

- Design a decimal adder to add two, single digit decimal numbers input in 8421 code using 4-bit parallel adder and basic logic gates. Output should also be represented in 8421 (BCD).





# BCD ADDER: TRUTH TABLE

K	Binary Sum				C	BCD Sum				Decimal
	Z <sub>8</sub>	Z <sub>4</sub>	Z <sub>2</sub>	Z <sub>1</sub>		S <sub>8</sub>	S <sub>4</sub>	S <sub>2</sub>	S <sub>1</sub>	
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9
{ } 0 1 0 1 0					1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19

2 <sup>8</sup>	2 <sup>4</sup>	2 <sup>2</sup>	2 <sup>0</sup>	11	00	01	10
00	00	00	00	11	00	00	00
01	00	00	00	11	00	00	00
11	11	11	11	11	11	11	11
10	01	01	01	11	01	01	11

$$F_1 = Z_8 Z_4 + Z_8 Z_2$$

$$F = K + F_1$$

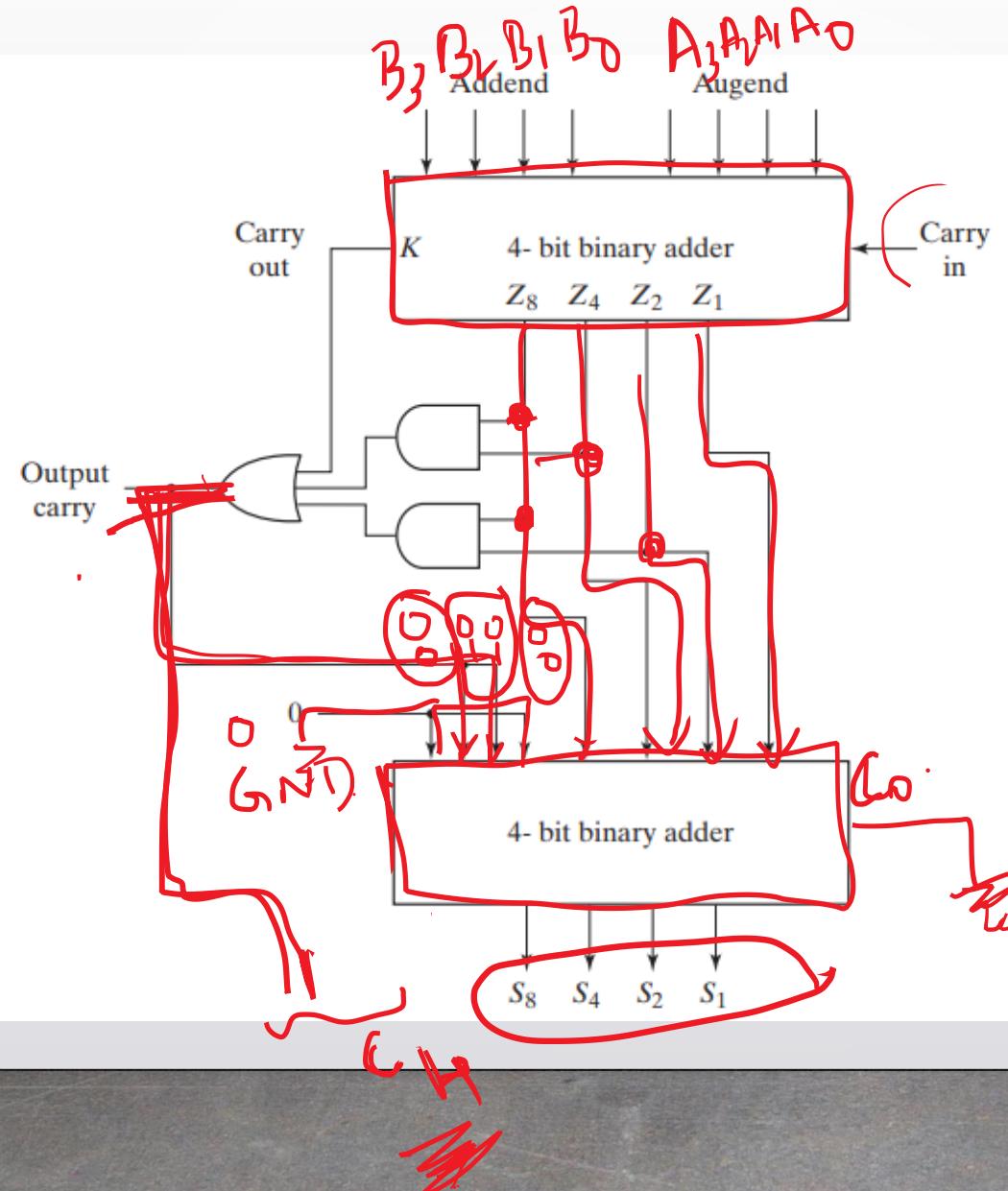
$$F = K + Z_8 Z_4 + Z_8 Z_2$$

# BCD ADDER

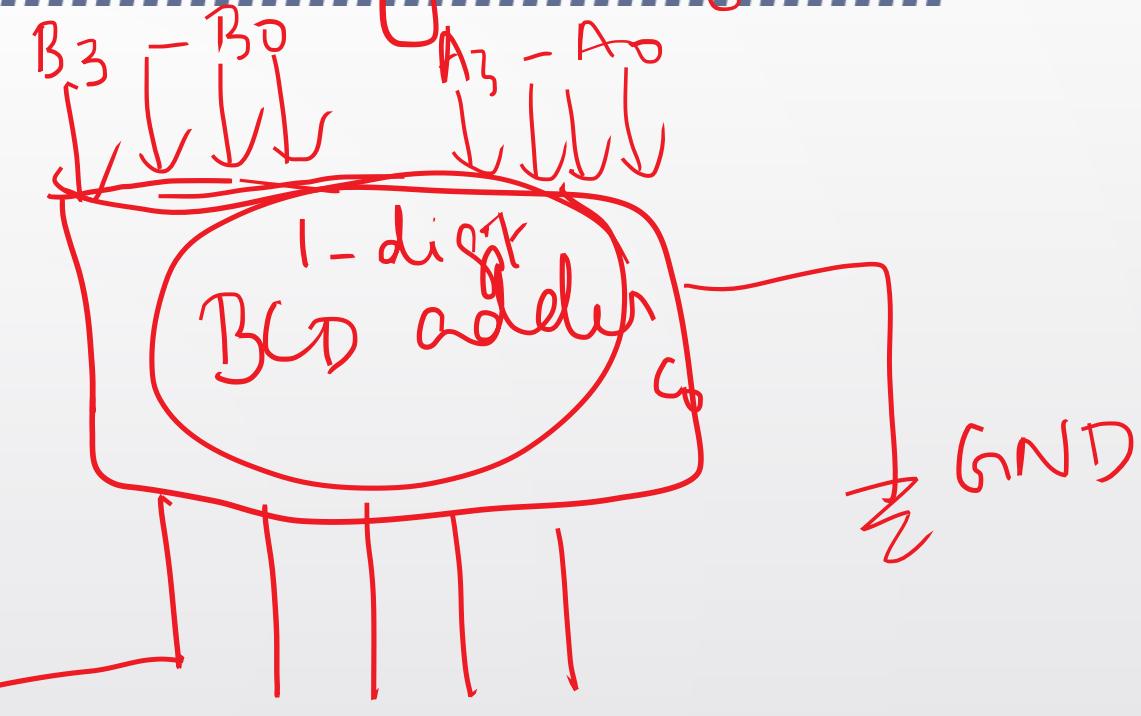
---

- Binary sum can be converted to BCD by adding 6 to binary sum.
- 6 needs to be added only when binary sum is  $> 9$  or  $(1001)_2$
- Referring the truth table, write the expression for F such that,  $F=1$  if binary sum is  $> 9$  else  $F=0$

# Block diagram of BCD adder



2-digit BCD adder using 1-digit BCD adder



Design 2-digit BCD adder using 7483 ICs & external gates.

(4-bit binary adder)

# Reference:



- Digital design , third edition by morris mano, chapter 4
  - . Slides are used only as a supporting material to teach the subject.
  - . Students should write down the notes and read the text book.



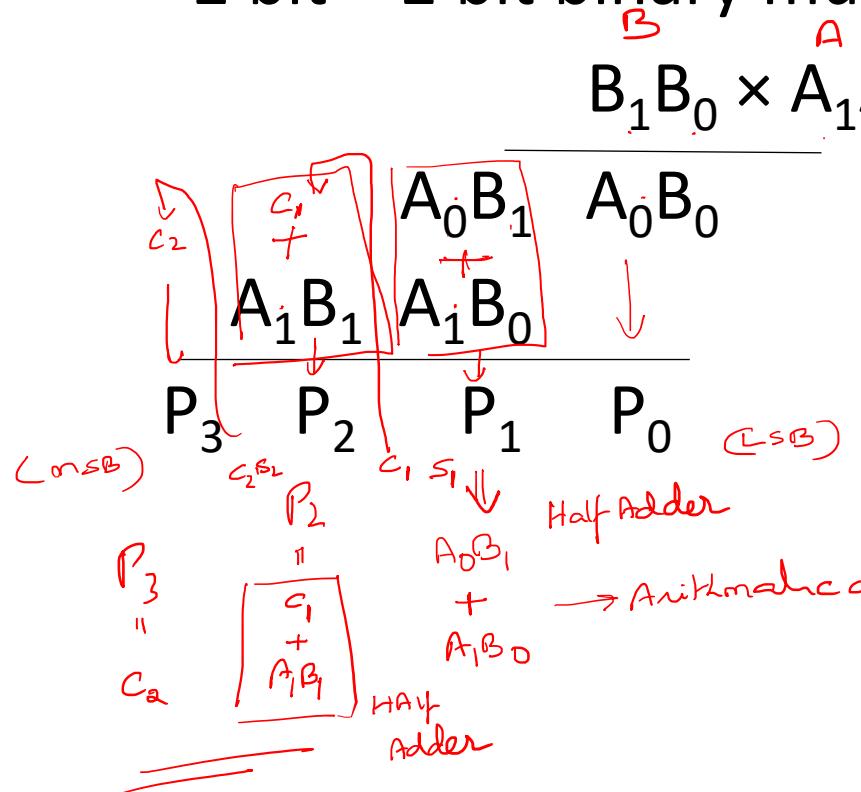
**Questions?**

# Multipliers and Magnitude comparators

---

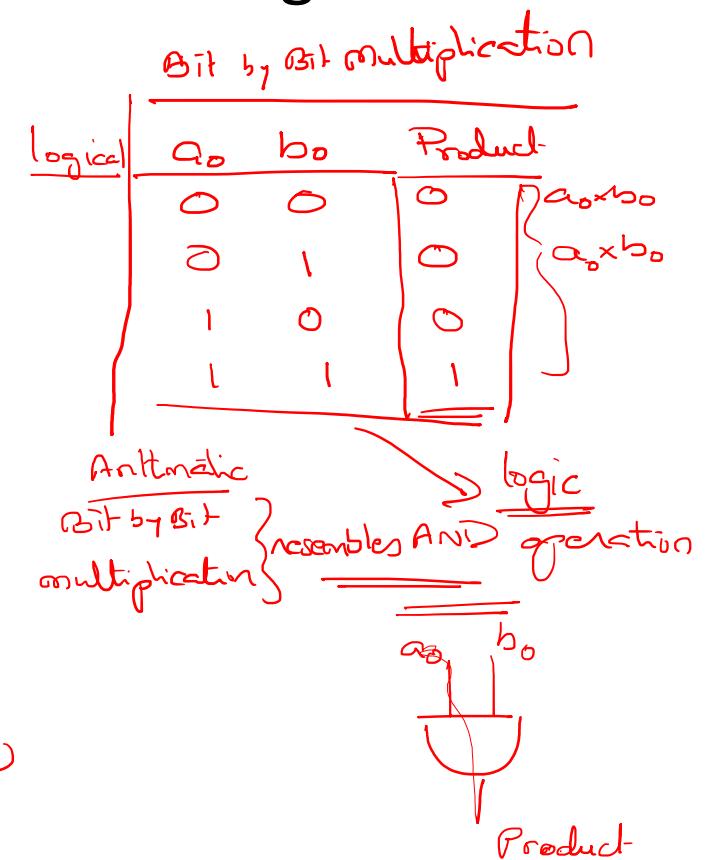
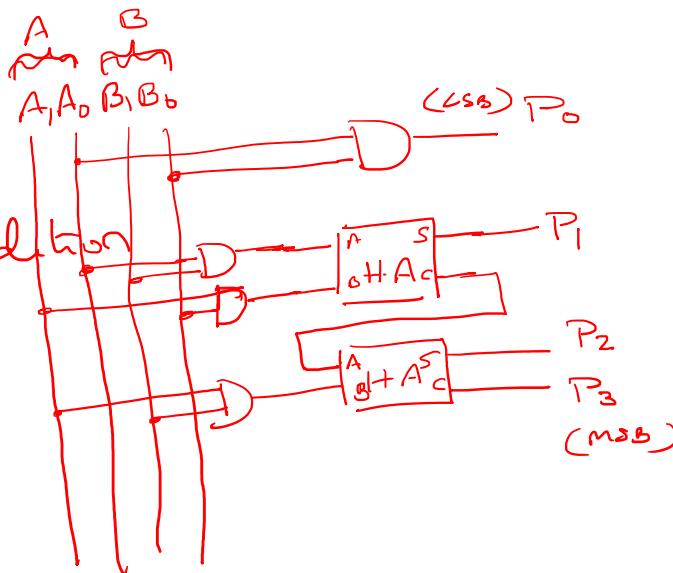
# Binary Multiplier

- 2 bit  $\times$  2 bit binary multiplier using adders and external gates.



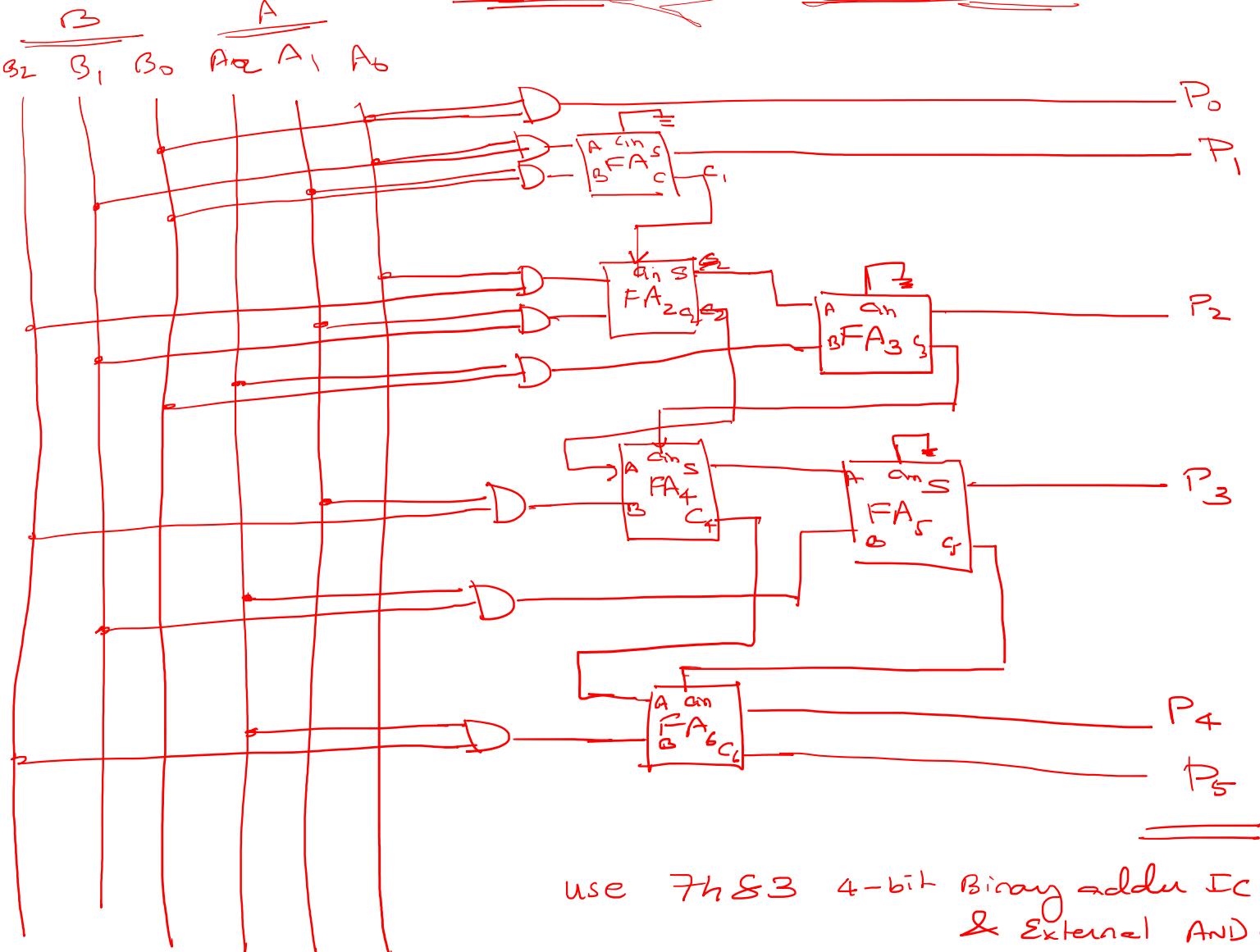
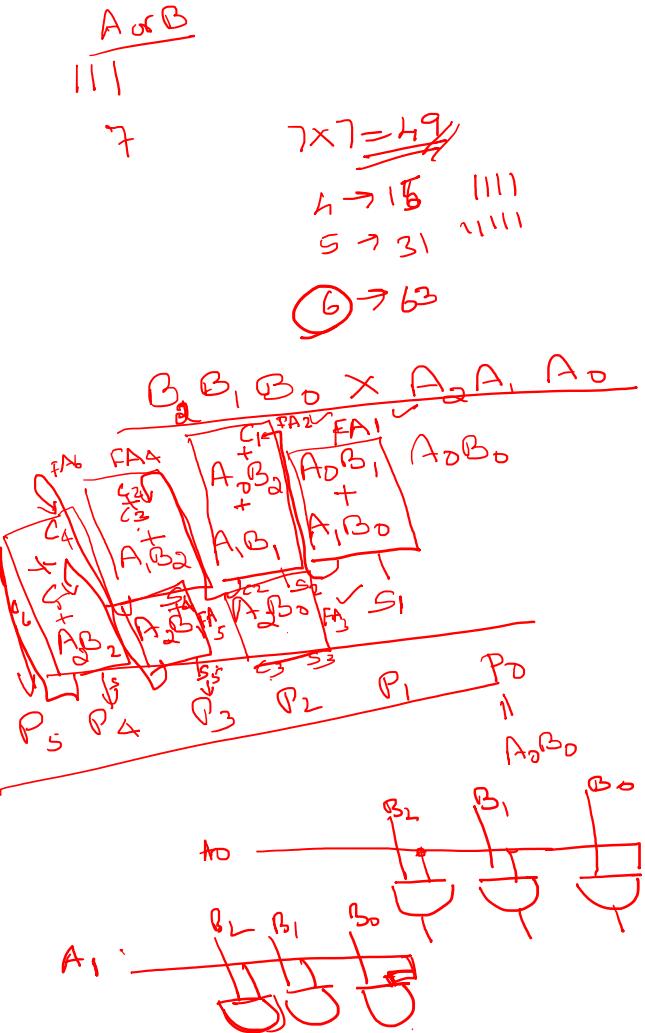
Ex:  $10 \times 11$

$$\begin{array}{r} b_1 b_0 \\ 10 \\ \times 11 \\ \hline b_1 a_1 & b_0 a_1 \\ 1 \times 1 & 0 \times 1 \\ \hline b_0 a_0 & 0 \times 1 \\ 1 \times 1 & 0 \times 1 \\ \hline & 0 \times 1 \end{array}$$





- Design a 3 bit  $\times$  3 bit binary multiplier using Full adders and external AND gates.



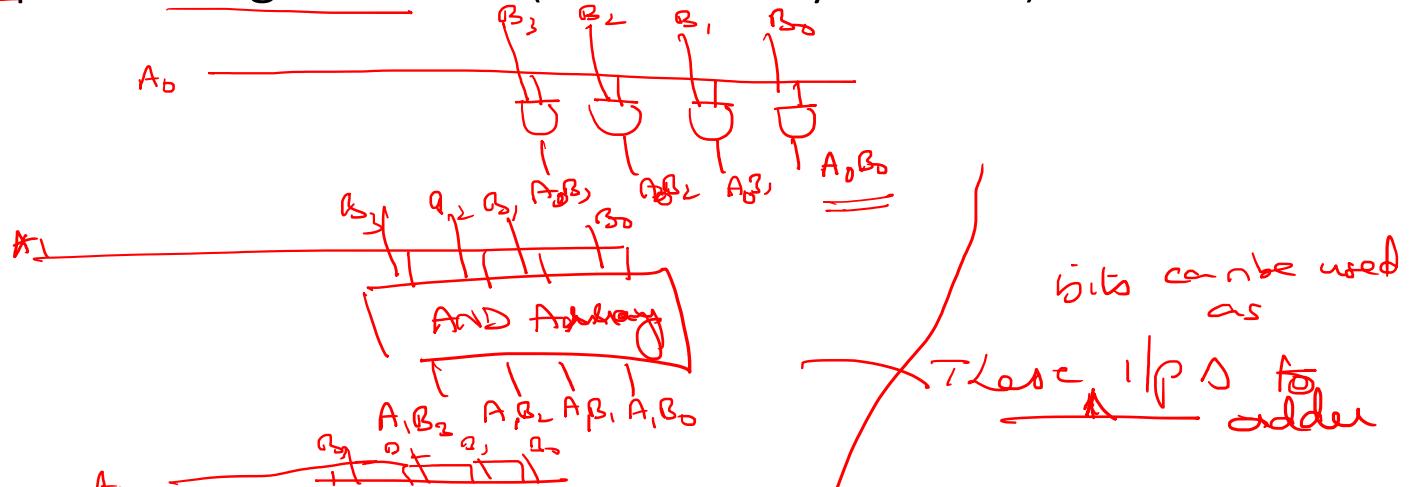
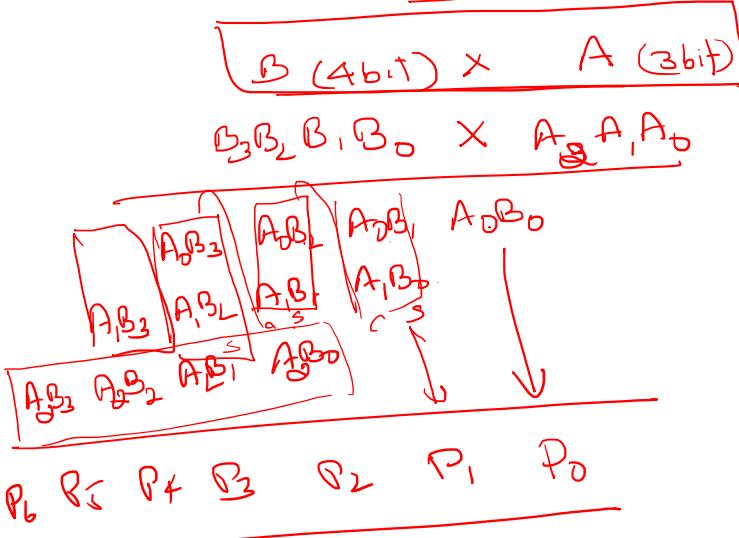
use 74LS3 4-bit Binary adder IC  
& External AND gates  
to realize  $3 \times 3$  bit  
multiplication.



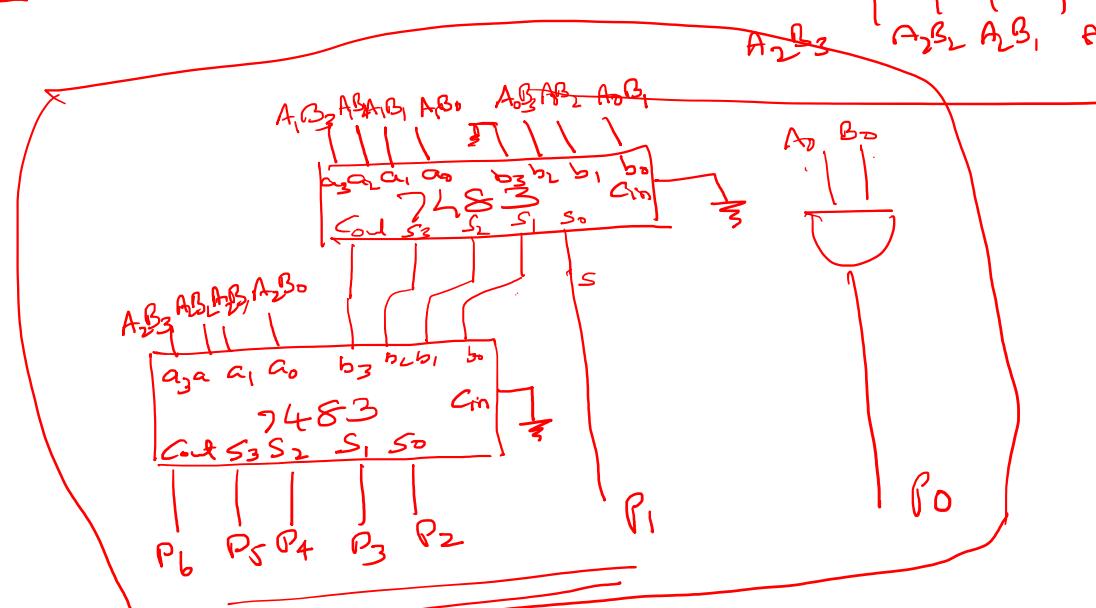
$$1111 \times 111 = \underline{105} \quad \text{minimum } \underline{7 \text{ bits}}$$

B      A

Design a 4 bit  $\times$  3 bit binary multiplier using 7483 ICs (4 bit binary adders) and external AND gates.

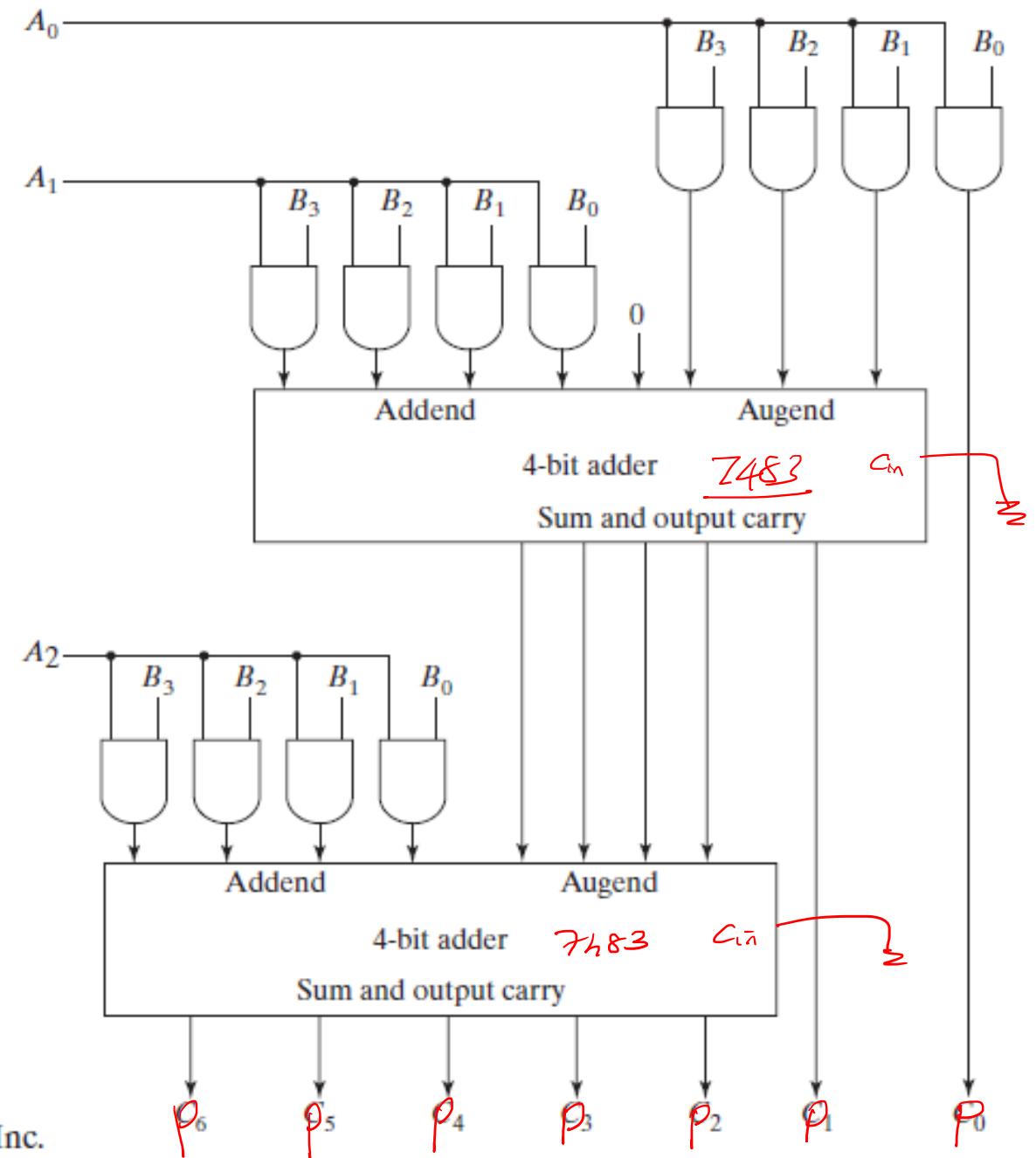


bits can be used  
as  
 $A \rightarrow D$  to  
adder





## 4-bit by 3-bit binary multiplier



# Magnitude Comparator

- 1 bit Magnitude comparator

*Truth Table*

Input		Output		
A	B	A < B	A = B	A > B
m <sub>0</sub>	0	0	1	0
m <sub>1</sub>	0	1	0	0
m <sub>2</sub>	1	0	0	1
m <sub>3</sub>	1	1	0	0

*for Example*

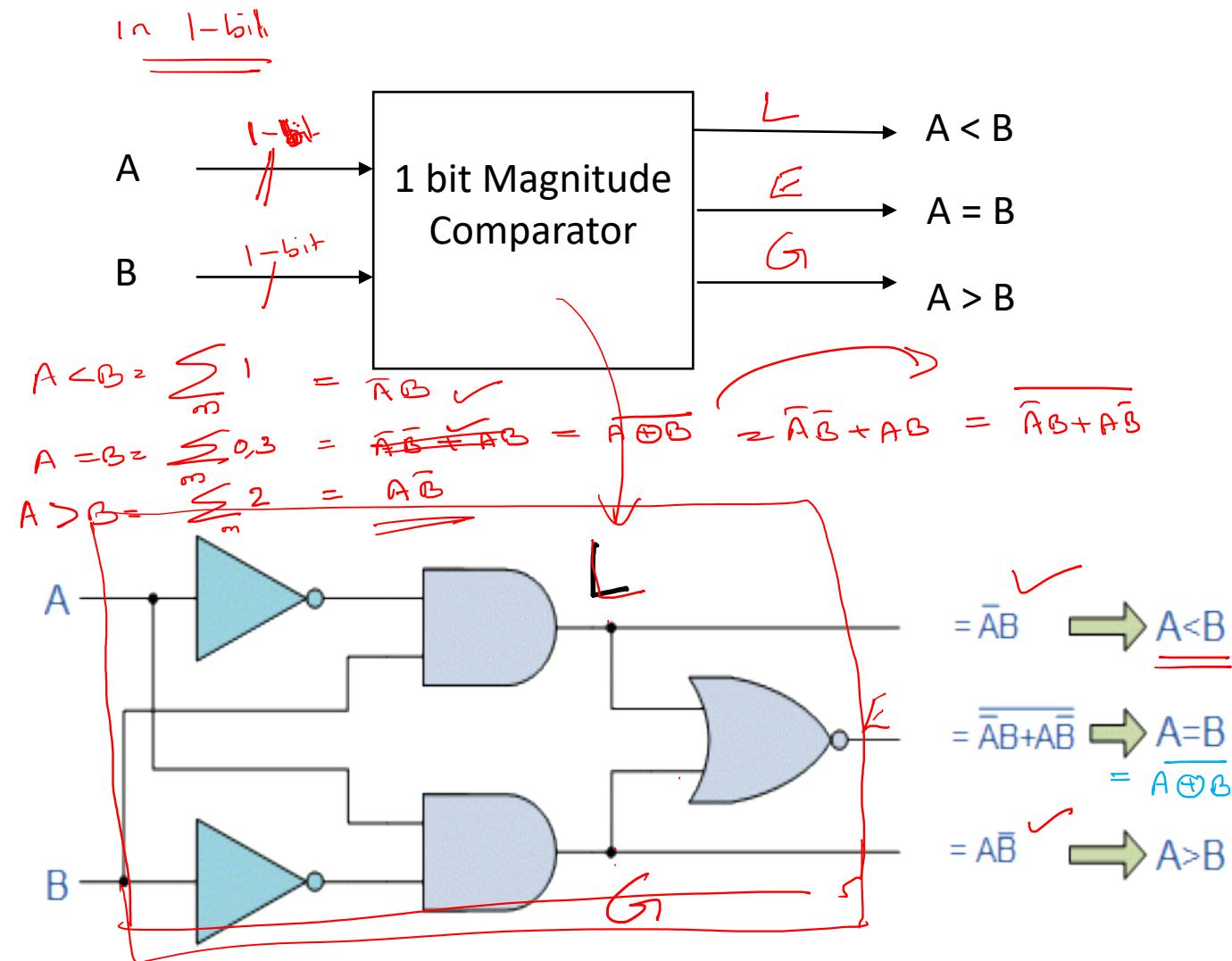
if to compare A & B  $\Rightarrow$  whether  $A > B$   
 $A = B$   
 $A < B$

① Draw V/P logic relation  
 using combinational logic

② subtraction  $A - B$

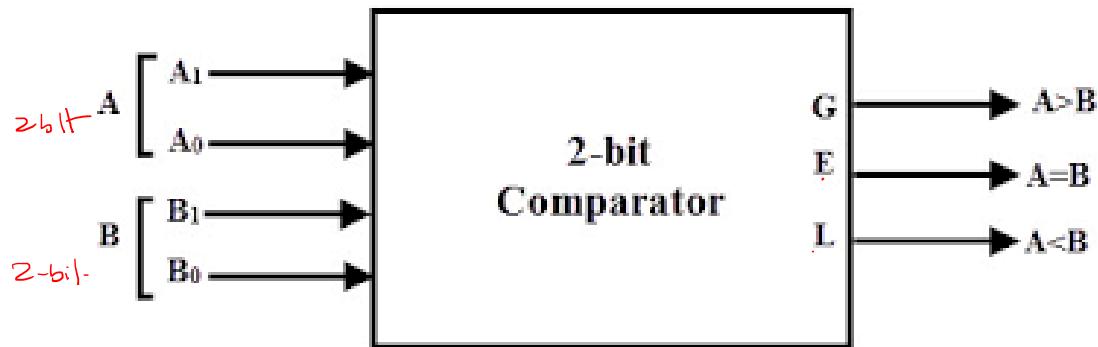
$$\begin{aligned} \frac{-A}{B} \\ \text{Ans} \neq 0 \text{ & } +ve \\ \neq 0 \text{ & } -ve \\ = 0 \end{aligned}$$

$$\begin{aligned} A > B \\ A \neq B \\ A < B \end{aligned}$$



## • 2 bit magnitude comparator

Inputs				Outputs		
$A_1$	$A_0$	$B_1$	$B_0$	$A > B$	$A = B$	$A < B$
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	0	1	0
1	1	1	1	0	1	0



$$G = A > B = \sum_m{4, 8, 9, 12, 13, 14}$$

$$E = A = B = \sum_m{0, 5, 10, 15}$$

$$L = A < B = \sum_m{1, 2, 3, 6, 7, 11}$$

$$\begin{matrix} A \\ A_1 \\ A_0 \\ B_1 \\ B_0 \end{matrix}$$

$$\begin{matrix} A \\ 48 \\ A \\ 43 \\ B \end{matrix}$$

$A_1, A_0$	$B_1, B_0$	00	01	11	10
00	0	0	0	0	0
01	0	0	0	0	0
11	0	0	0	0	0
10	1	1	1	0	0

$$G = \sum_m{4, 8, 9, 12, 13, 14}$$

$$G = A_1 \bar{B}_1 + A_0 \bar{B}_1 \bar{B}_0 + A_1 A_0 \bar{B}_0$$

$$G = A_1 \bar{B}_1 + A_0 \bar{B}_0 [A_1 + \bar{B}_1]$$

$$= A_1 \bar{B}_1 + \bar{A}_1 A_0 \bar{B}_1 \bar{B}_0 + [A_1 A_0 B_1 \bar{B}_0]$$

$$= \bar{A}_1 A_0 \bar{B}_1 + A_0 \bar{B}_0 (\bar{A}_1 \bar{B}_1 + A_1 \bar{B}_1)$$

$$= A_1 \bar{B}_1 + A_0 \bar{B}_0 (A_1 \oplus B_1)$$

$$A > B = (A_1 > B_1) + (A_1 = B_1)(A_0 > B_0)$$

$$= A_1 \bar{B}_1 + (A_1 \oplus B_1)(A_0 \bar{B}_0)$$

$$E = A_1 = B_1$$

$$L = A_0 < B_0$$

# 2-bit magnitude comparator

$$A = B \Rightarrow E = (\overline{A_1 = B_1}) \cdot (\overline{A_0 = B_0}) = (\overline{A_1} \oplus \overline{B_1}) \cdot (\overline{A_0} \oplus \overline{B_0})$$

$E_1 \cdot E_0$

$$A < B \Rightarrow L = (\overline{A_1 < B_1}) + (\overline{A_1 = B_1}) \cdot (\overline{A_0 < B_0})$$

$$L = (\overline{A_1} \cdot \overline{B_1}) + (\overline{A_1} \oplus \overline{B_1}) \cdot (\overline{A_0} \cdot \overline{B_0})$$

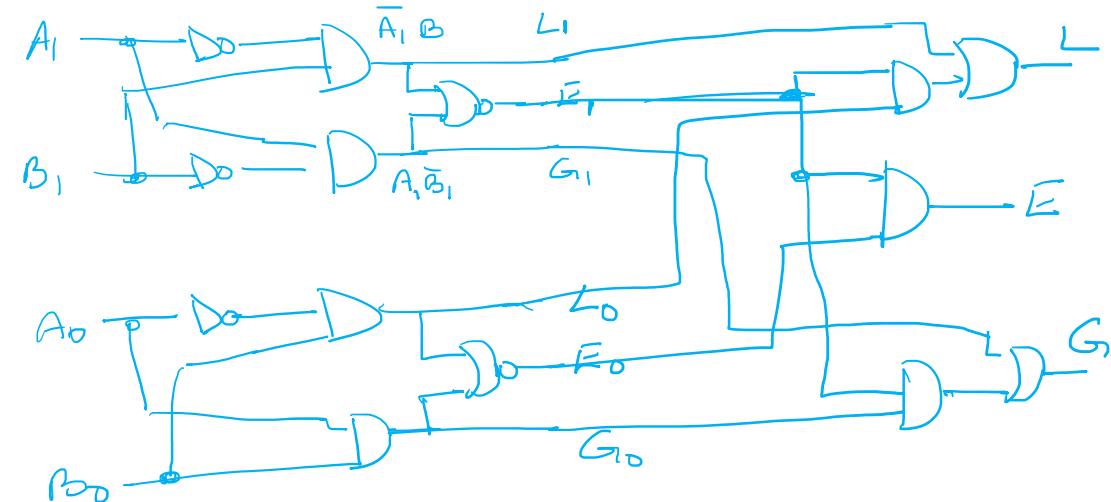
$L_1 + E_1 \cdot L_0$

$$A > B = G = (\overline{A_1 > B_1}) + (\overline{A_1 = B_1}) \cdot (\overline{A_0 > B_0})$$

$$G = \overline{A_1} \cdot \overline{B_1} + (\overline{A_1} \oplus \overline{B_1}) \cdot (\overline{A_0} \cdot \overline{B_0})$$

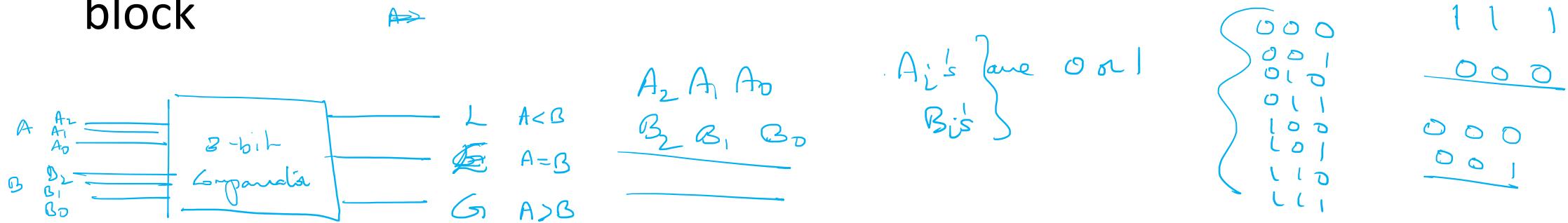
$G_1 + E_1 \cdot G_0$

$$\begin{array}{l} A_1, A_0 \\ B_1, B_0 \\ \hline \overline{A_1 \oplus B_1} = \overline{\overline{A_1} \cdot \overline{B_1} + A_1 \cdot \overline{B_1}} \end{array}$$



Two-bit magnitude comparator  
using 1-bit magnitude comparators

- 3 bit magnitude comparator using 1-bit magnitude comparator block



$$G \text{ or } A > B = G_2 + E_2 G_1 + E_2 E_1 G_0 = A_2 \bar{B}_2 + (\bar{A}_2 \oplus B_2) \cdot A_1 \bar{B}_1 + (\bar{A}_2 \oplus B_2) (\bar{A}_1 \oplus B_1) A_0 \bar{B}_0$$

$$(A_2 > B_2) + (A_2 = B_2)(A_1 > B_1) + (A_2 = B_2)(A_1 = B_1)(A_0 > B_0)$$

$$E \text{ or } A = B = E_2 \cdot E_1 \cdot E_0 = (\bar{A}_2 \oplus B_2) \cdot (\bar{A}_1 \oplus B_1) \cdot (\bar{A}_0 \oplus B_0)$$

$$(A_2 = B_2) \cdot (A_1 = B_1) \cdot (A_0 = B_0)$$

$$L \text{ or } A < B = L_2 + E_2 L_1 + E_2 E_1 L_0 = \bar{A}_2 B_2 + (\bar{A}_2 \oplus B_2) \cdot \bar{A}_1 B_1 + (\bar{A}_2 \oplus B_2) (\bar{A}_1 \oplus B_1) \cdot \bar{A}_0 B_0$$

$$(A_2 < B_2) + (A_2 = B_2)(A < B) + (A_2 = B_2)(A = B_1)(A < B_0)$$

- 4 bit magnitude comparator



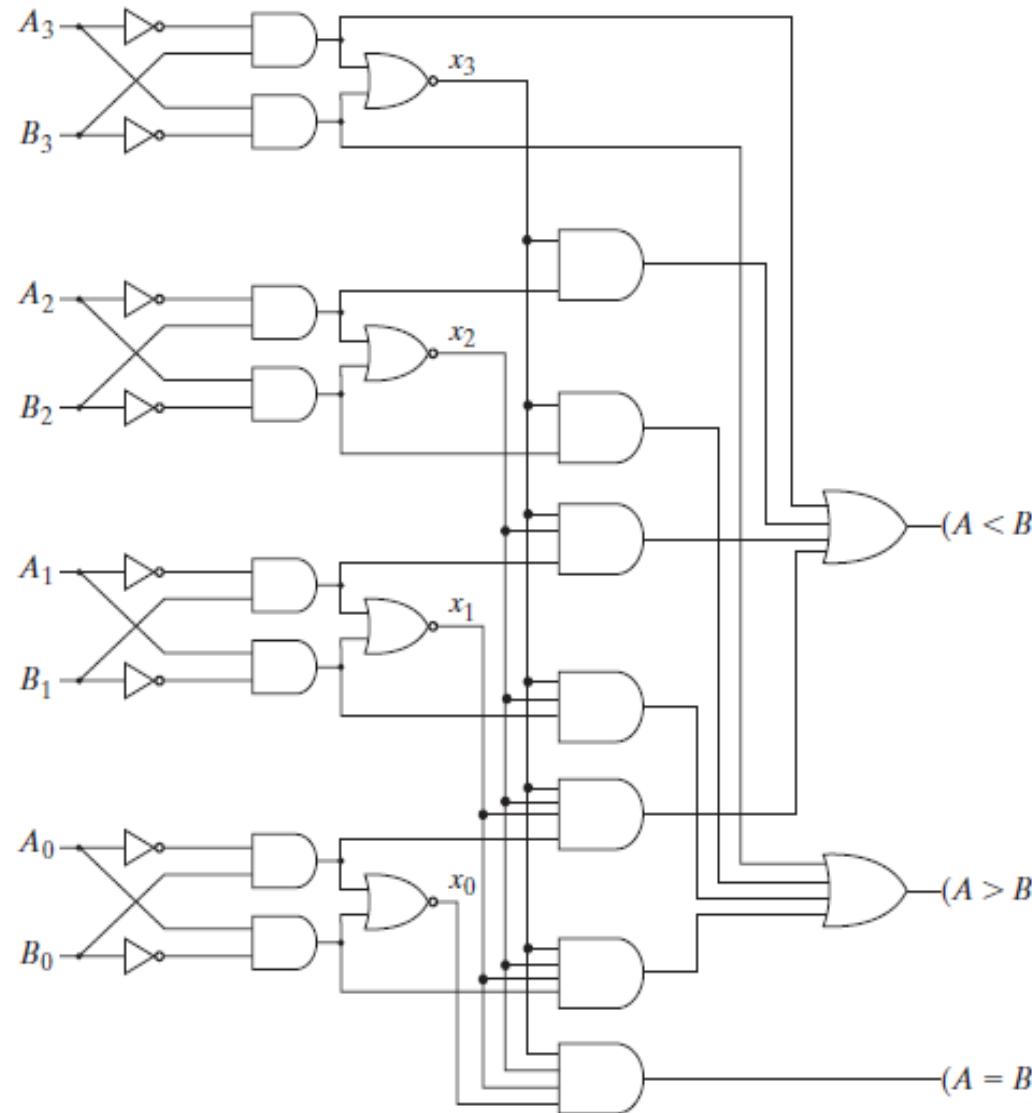
$A_3 A_2 A_1 A_0$   
 $B_3 B_2 B_1 B_0$

$$G_1 = A > B = \overline{G_3} + \overline{E_3} \overline{G_2} + \overline{E_3} \overline{E_2} G_1 + \overline{E_3} \overline{E_2} \overline{E_1} G_0 \\ \quad + \overline{A_3} \overline{B_3} + (\overline{A_3} \oplus \overline{B_3}) \overline{A_2} \overline{B_2} + \dots - \dots - \dots - \dots$$

$$E = A = B \Rightarrow \overline{E_3} \cdot \overline{E_2} \cdot \overline{E_1} \cdot \overline{E_0}$$

$$L = A < B \Rightarrow \overline{L_3} + \overline{E_3} \overline{L_2} + \overline{E_3} \overline{E_2} L_1 + \overline{E_3} \overline{E_2} \overline{E_1} L_0$$

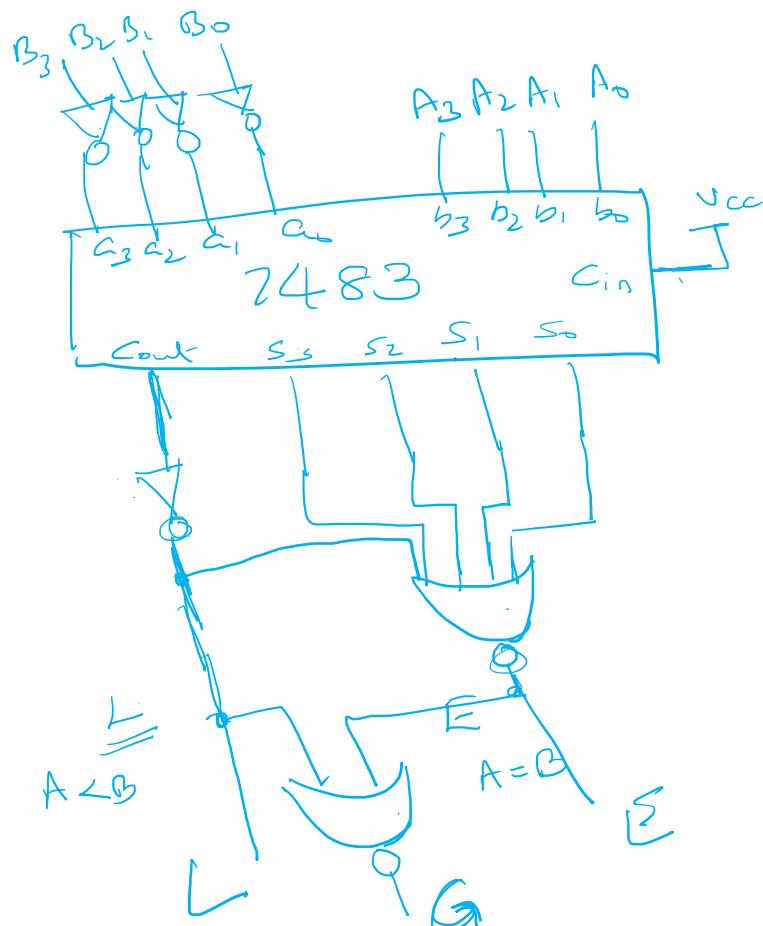
# 4-bit magnitude comparator



only NOR

Design a 4-bit magnitude comparator using 7483 IC and external gates

What is 7483 IC?



?  
4-bit Binary Adder

$$\begin{array}{r} A \\ -B \\ \hline \end{array}$$

+ 2's Comp B

EQUAL ?

S3

$$G = \overline{L} \cdot \overline{E}$$

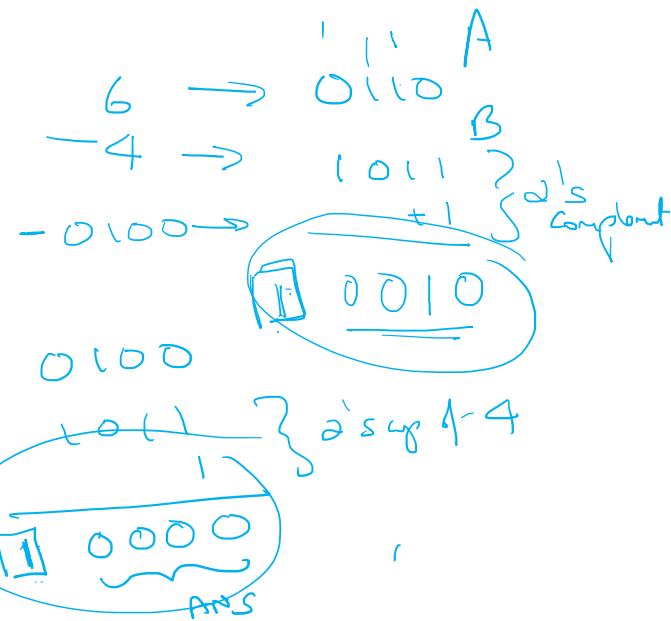
$$L + E = \overline{L} \cdot \overline{E}$$

-4 - 0110  
Carry  $\rightarrow$  +ve

no carry  $\rightarrow$  -ve

$$\begin{array}{r} 0100 \\ 100 \\ \hline 1110 \end{array}$$

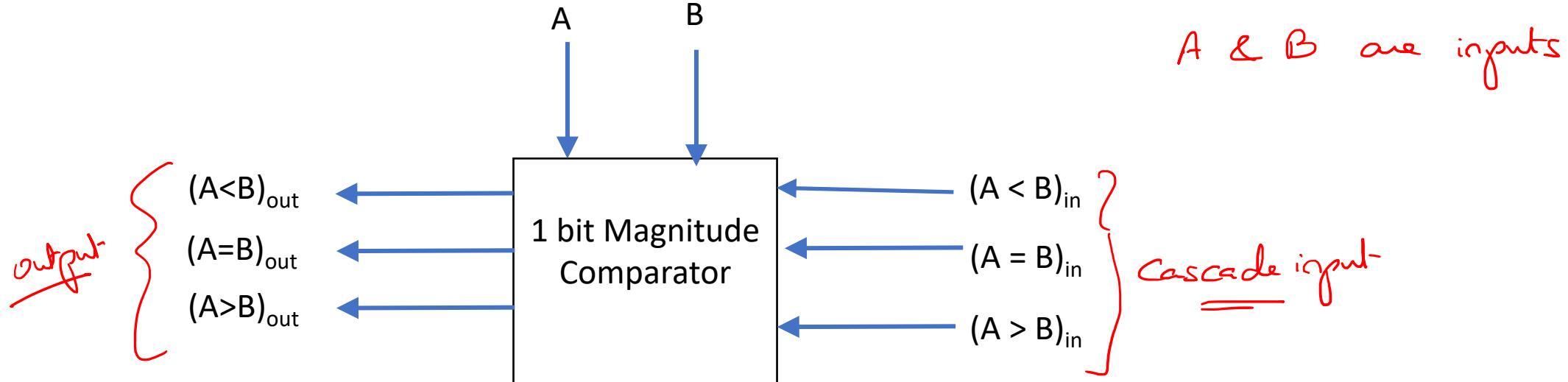
Ans - 2's comp



# Multipliers and Magnitude comparators

---

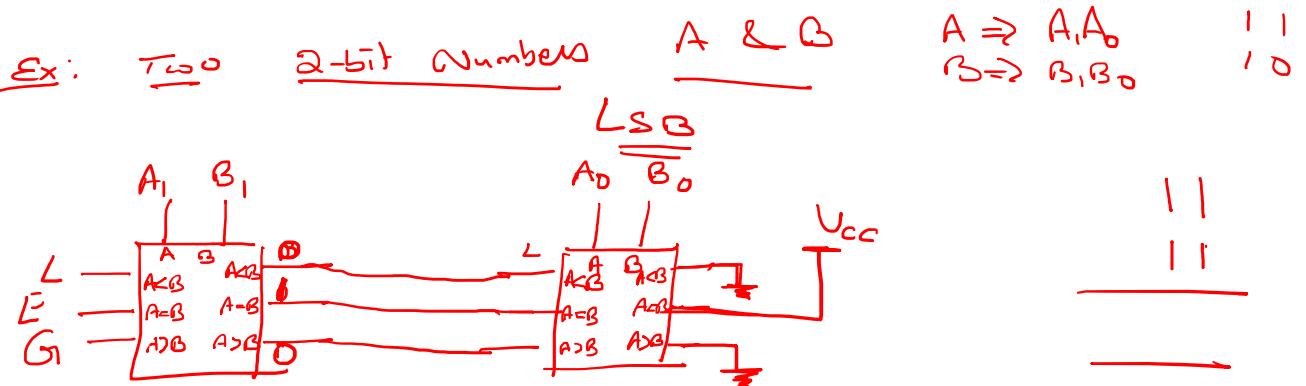
- 1-bit magnitude comparator with cascading input:



$$G \quad (A > B)_{out} = (A > B) + (A = B)(A > B)_{in}$$

$$L \quad (A < B)_{out} = (A < B) + (A = B)(A < B)_{in}$$

$$E \quad (A = B)_{out} = (A = B)(A = B)_{in}$$

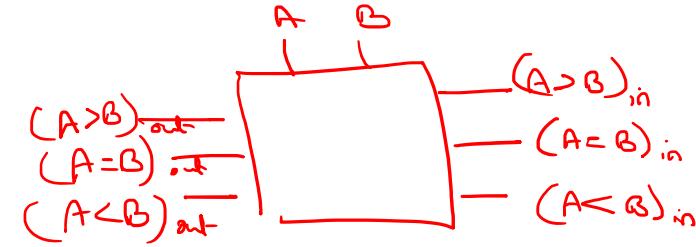
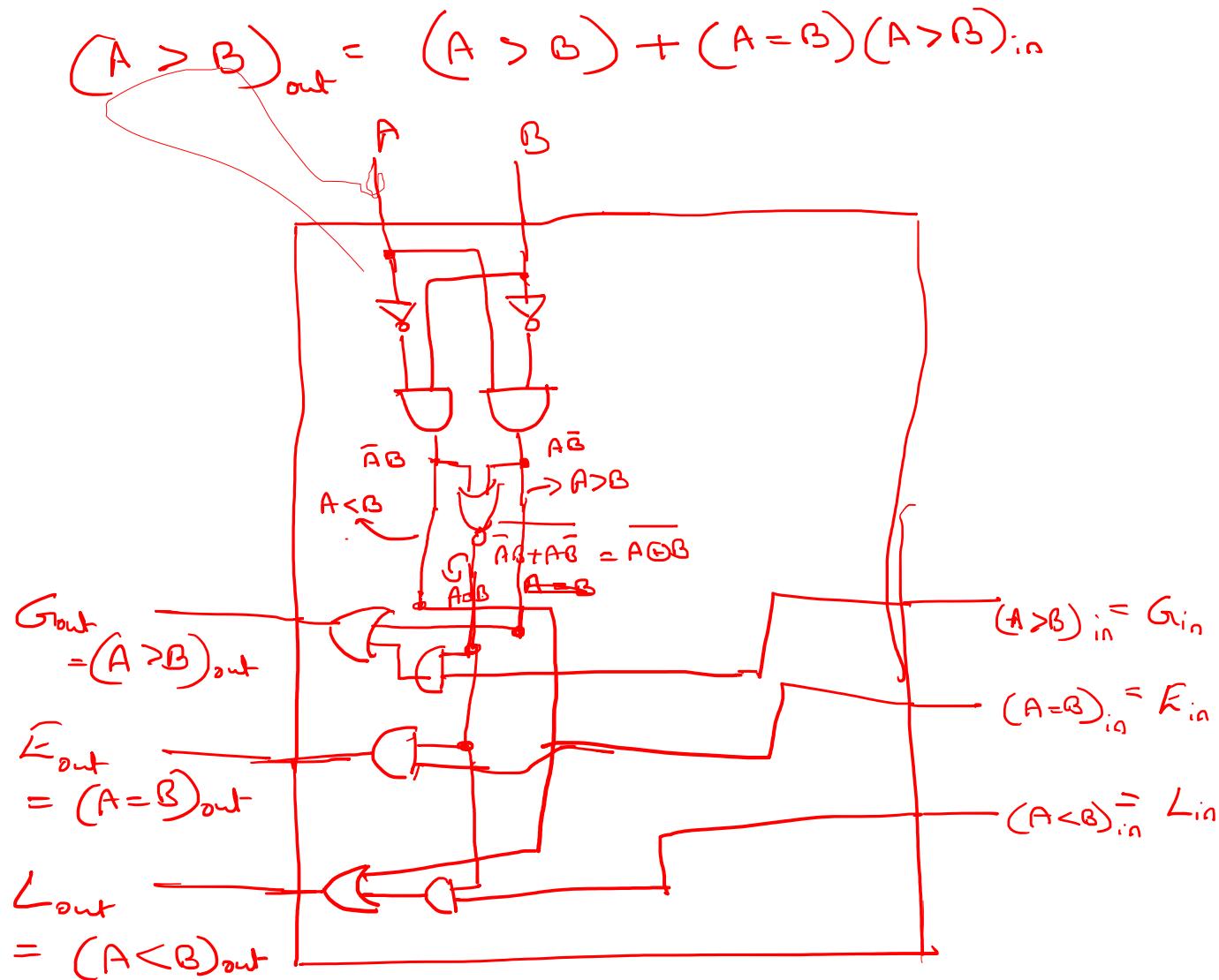


$$G \quad A > B = (A_1 > B_1) + (A_1 = B_1) \cdot (A_0 > B_0) \Rightarrow G = G_1 + E_1.G_0$$

$$L \quad A < B = (A_1 < B_1) + (A_1 = B_1) \cdot (A_0 < B_0) \Rightarrow L = L_1 + E_1.L_0$$

$$E \quad (A = B) = (A_1 = B_1) \cdot (A_0 = B_0) \Rightarrow E = E_1 \cdot E_0$$

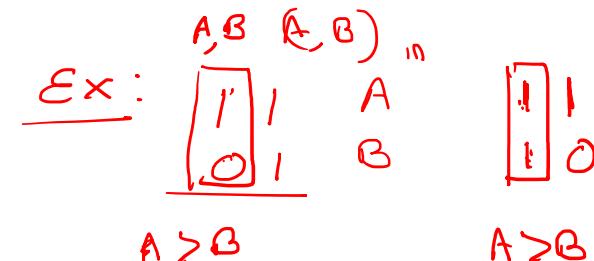
- Design of 1-bit magnitude comparator with cascading input:



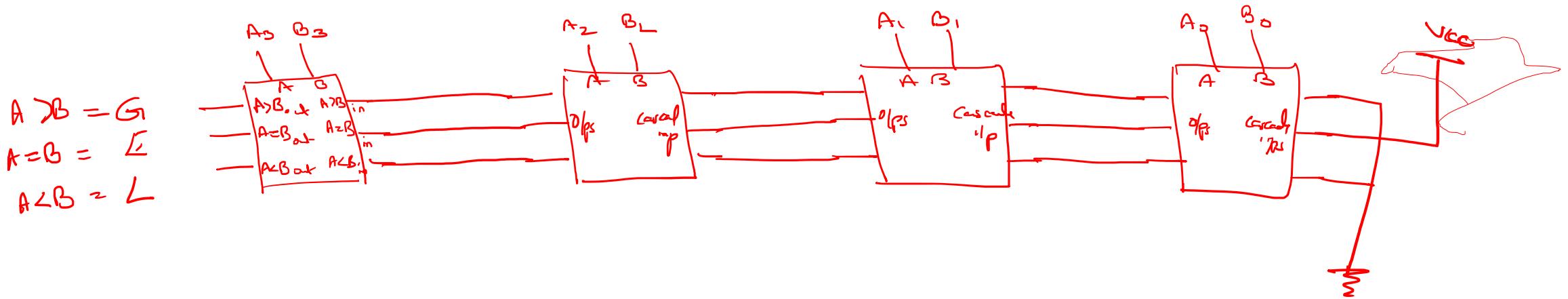
$$(A > B)_{out} = (A > B) + (A = B)(A > B)_{in}$$

$$(A = B)_{out} = (A = B) \cdot (A = B)_{in}$$

$$(A < B)_{out} = (A < B) + (A = B)(A < B)_{in}$$

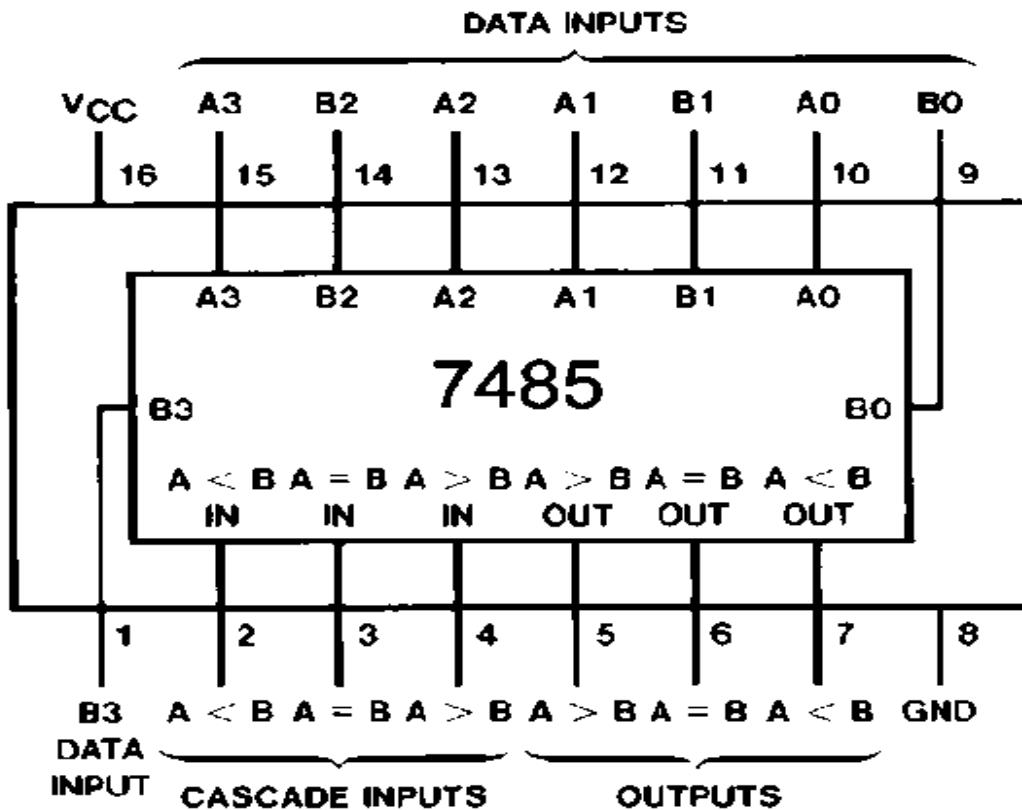


- Design a 4-bit magnitude comparator using 1-bit magnitude comparator with cascading inputs.

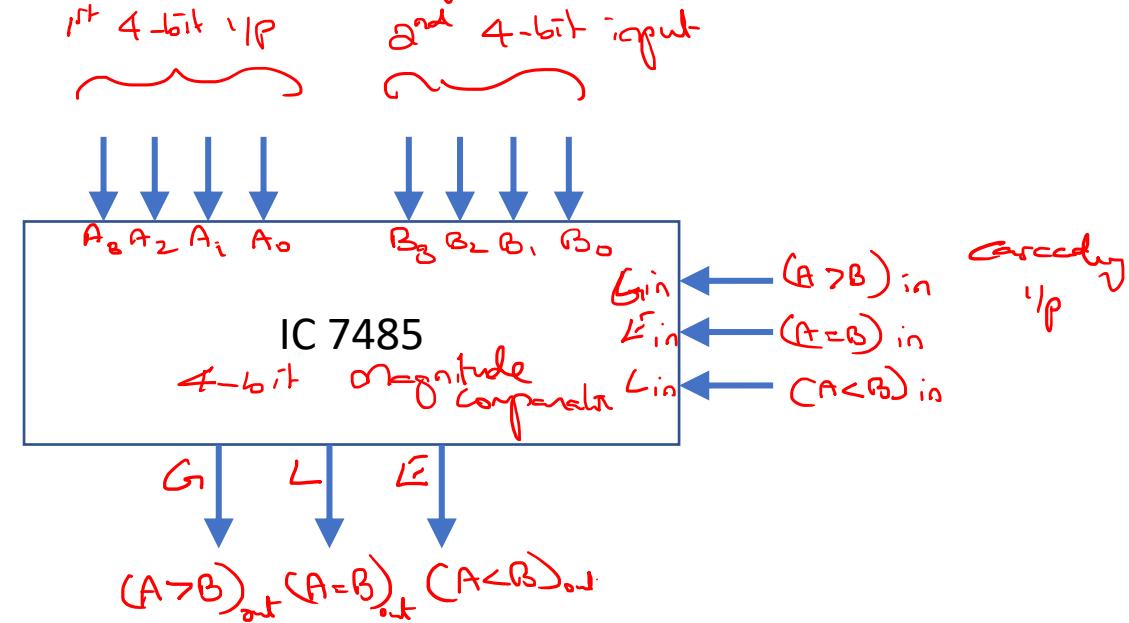


- 7485 IC ( 4-bit magnitude comparator with cascading inputs)

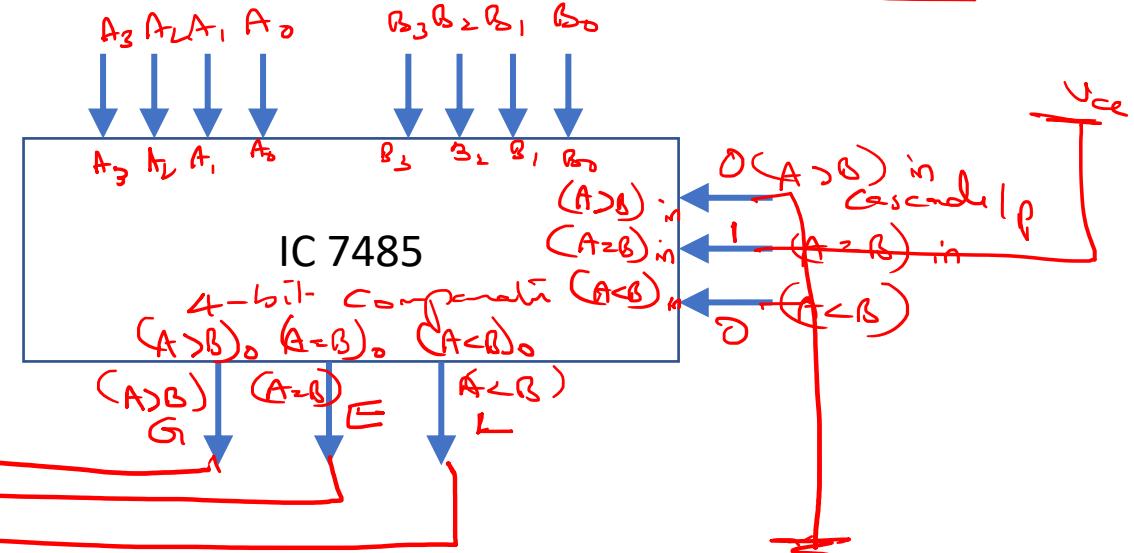
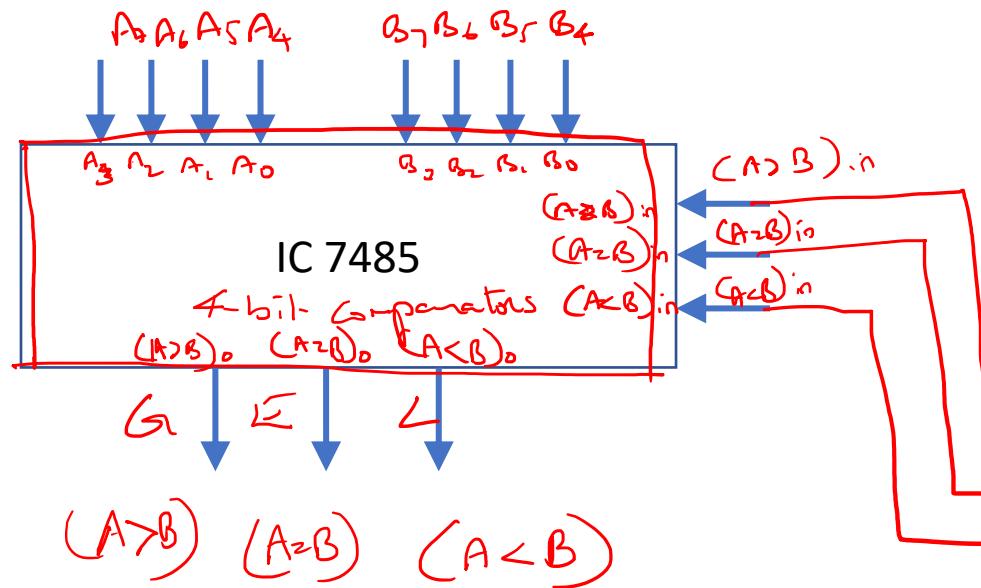
IC - Pin diagram



Schematic Diagram



- Design 8-bit magnitude comparator using 7485 ICs



$$\begin{aligned} A &= A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0 \\ B &= B_7 B_6 B_5 B_4 B_3 B_2 B_1 B_0 \end{aligned}$$

$\overbrace{\hspace{10em}}$

$t_o$  Clock  $A > B$   
 $A = B$   
 $A < B$

Don't forget

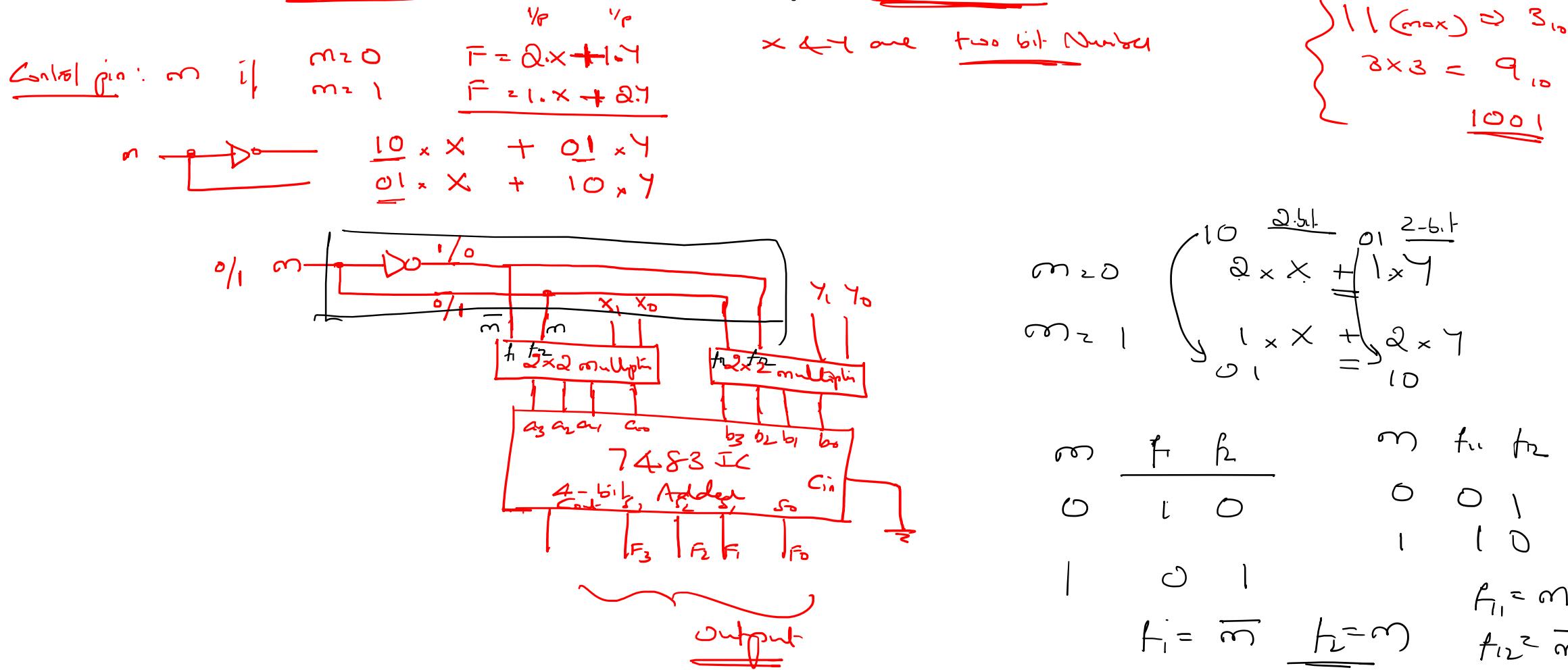
- Design a combinational circuit using 7483 IC and external gate to perform the arithmetic operations as shown below

4-bit Adder

Only one gate

$$F = 2X + Y \text{ when } m=0, \quad \& F = X + 2Y \text{ when } m=1$$

Where  $X, Y$  are two bit numbers and output  $F$  is a 4 bit number.



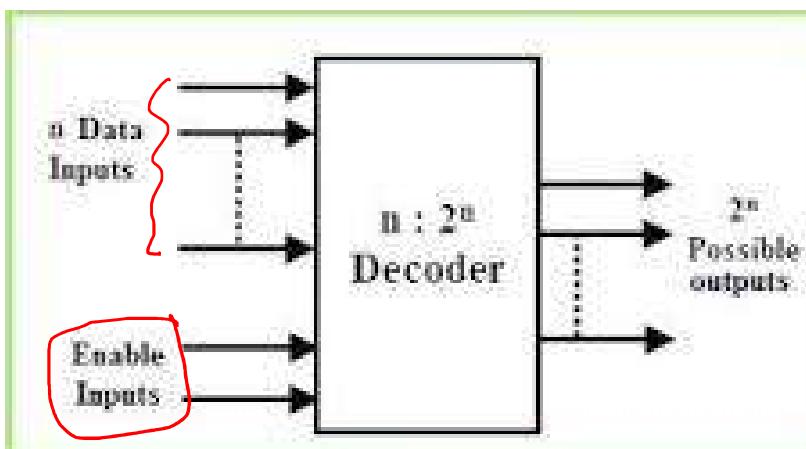
- Questions?

# DECODERS AND ENCODERS

Students are advised to write down the notes for every lecture

# Decoder:

- Decoder is a combinational circuit
- Converts a binary information from n-input lines to a maximum of  $2^n$  unique output lines ( n-to-  $2^n$  line decoder) and one or more enable inputs.Ex: 2-to-4 line, 3-to-8 line..etc
- In standard decoders, only one output line will be active at a time corresponding to the input binary combination.



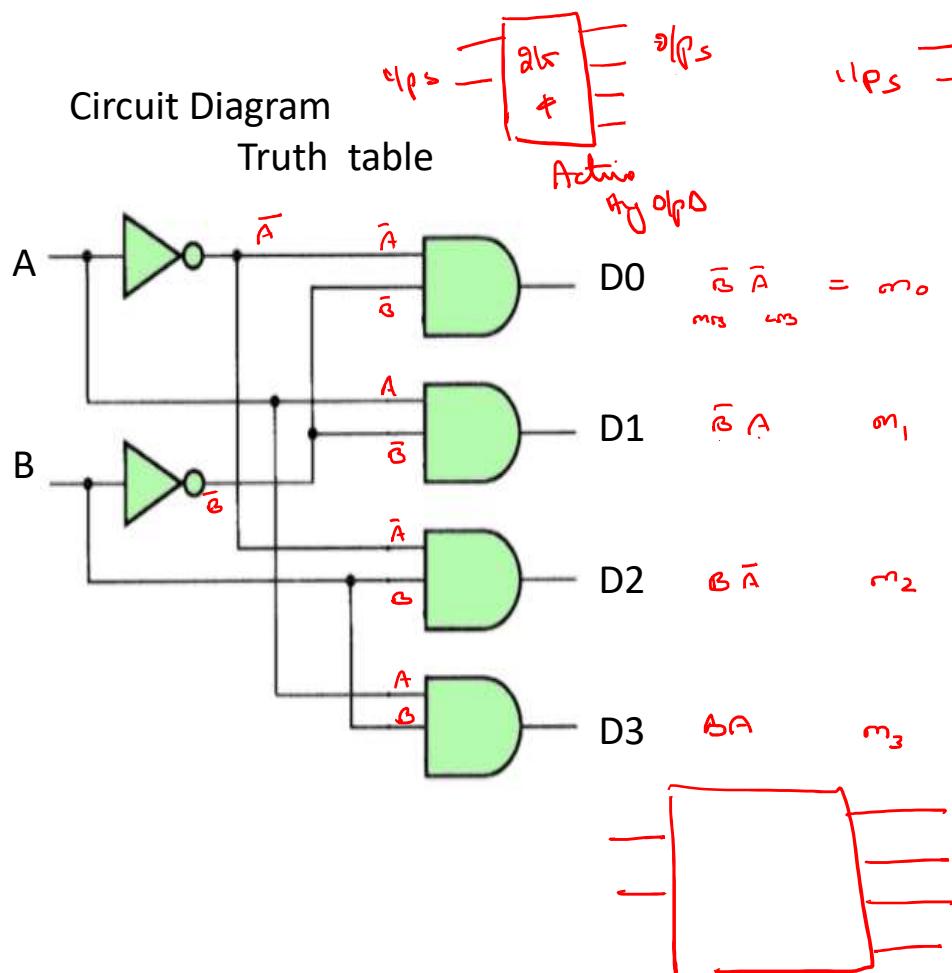
2<sup>n</sup>

E.g. 2-61  
2: a decoder

	D <sub>0</sub>	D <sub>1</sub>	D <sub>2</sub>	D <sub>3</sub>
00	x	x	x	✓
01	x	x	✓	x
10	x	✓	x	x
11	✓	x	x	x

1 : 2 line  
2 : 4 line  
3 . 8  
4 : 16 decoder  
⋮  
 $bil = 2^n$  lines decoder

# 2-to-4 line decoder



NAND Gates at output array

S	A	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
0	0	1	1	1	0
0	1	*	1	0	1
1	0	1	0	1	1
1	1	0	1	1	1

M<sub>3</sub> M<sub>2</sub> M<sub>1</sub> M<sub>0</sub>

Active low o/p

Truth table for Active HIGH o/p

$\bar{A}\bar{B}$	$\bar{A}B$	$A\bar{B}$	$AB$	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
0	0	0	0	0	0	0	1
0	1	0	0	0	0	1	0
1	0	0	1	0	1	0	0
1	1	1	0	1	0	0	0

m<sub>3</sub> m<sub>2</sub> m<sub>1</sub> m<sub>0</sub>

Active HIGH Outputs

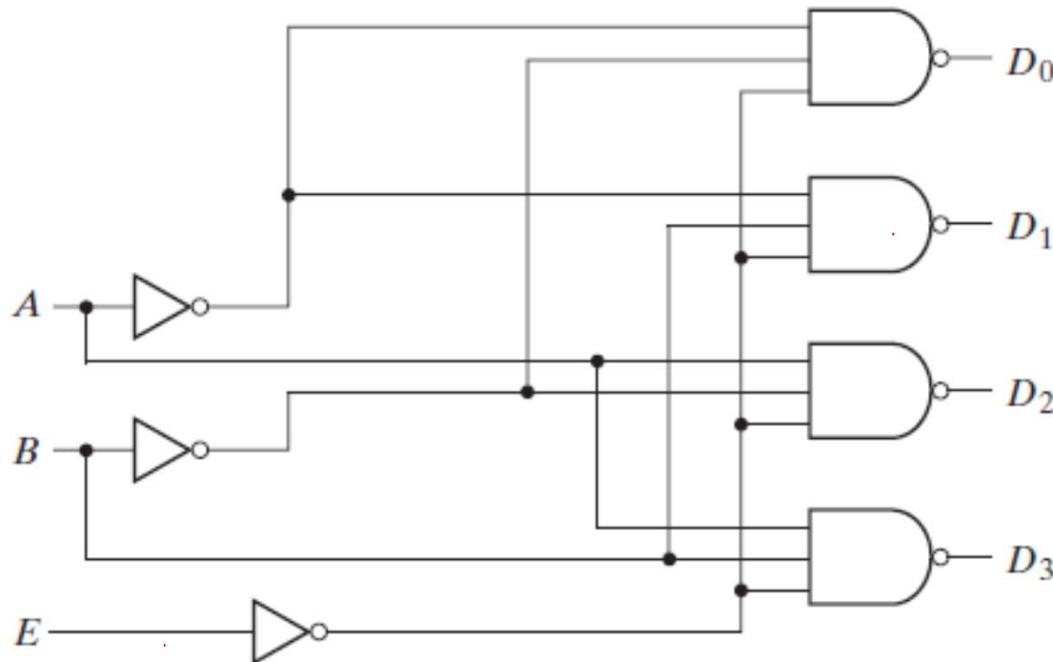
with outputs HIGH

Active HIGH output

with NAND Gates at outputs → Active LOW o/p

Note:  $m_2 = m_3$

2-to-4 line decoder with active low output &  
 enable <sup>input</sup> active low



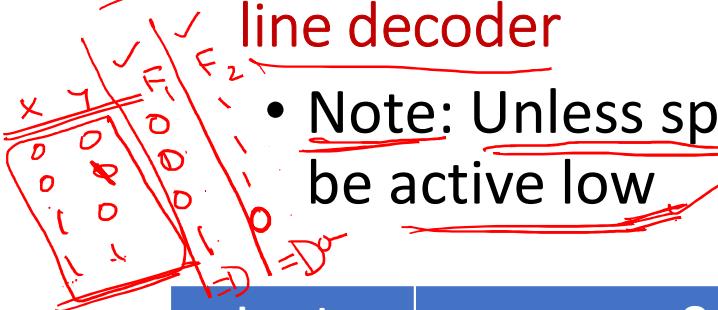
(a) Logic diagram

$E$	$A$	$B$	$D_0$	$D_1$	$D_2$	$D_3$
1	$X$	$X$	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

(b) Truth table



Ques: Write the truth table, logic diagram and block diagram of 3-to-8 line decoder

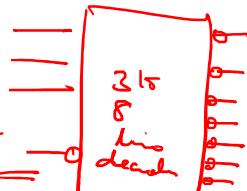


- Note: Unless specified, assume the output and enable input to be active low

Inputs			Outputs								
E	A	B	C	D0	D1	D2	D3	D4	D5	D6	D7
1	x	xx		*	*	*	*	*	*	*	*
0	0	0	0	0	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1
0	0	1	0	1	1	1	1	1	1	1	1
0	0	1	1	1	1	0	1	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1
0	1	1	0	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	0

Expressions for o/p variables

Question should arise and are

- Q1. Are there any enable inputs? E =    
 Q2. If enables are there, then are they active low / active HIGH?

- Q3. Whether o/p lines are active low/high?

$$D_0 = \overline{E} \overline{A} \overline{B} \overline{C}$$

$$D_1 = \overline{E} \overline{A} \overline{B} C$$

$$D_2 = \overline{E} \overline{A} B \overline{C}$$

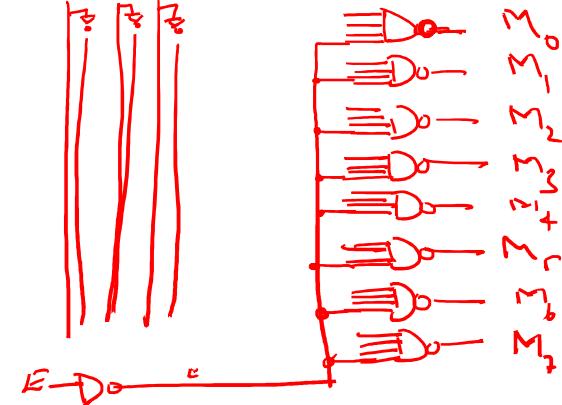
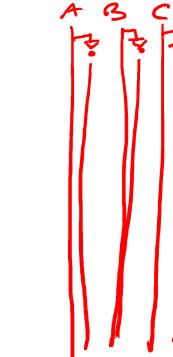
$$D_3 = \overline{E} \overline{A} B C$$

$$D_4 = \overline{E} A \overline{B} \overline{C}$$

$$D_5 =$$

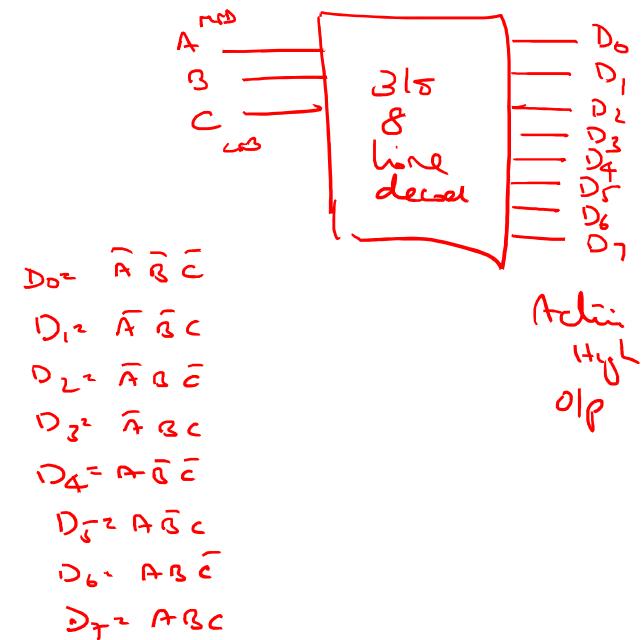
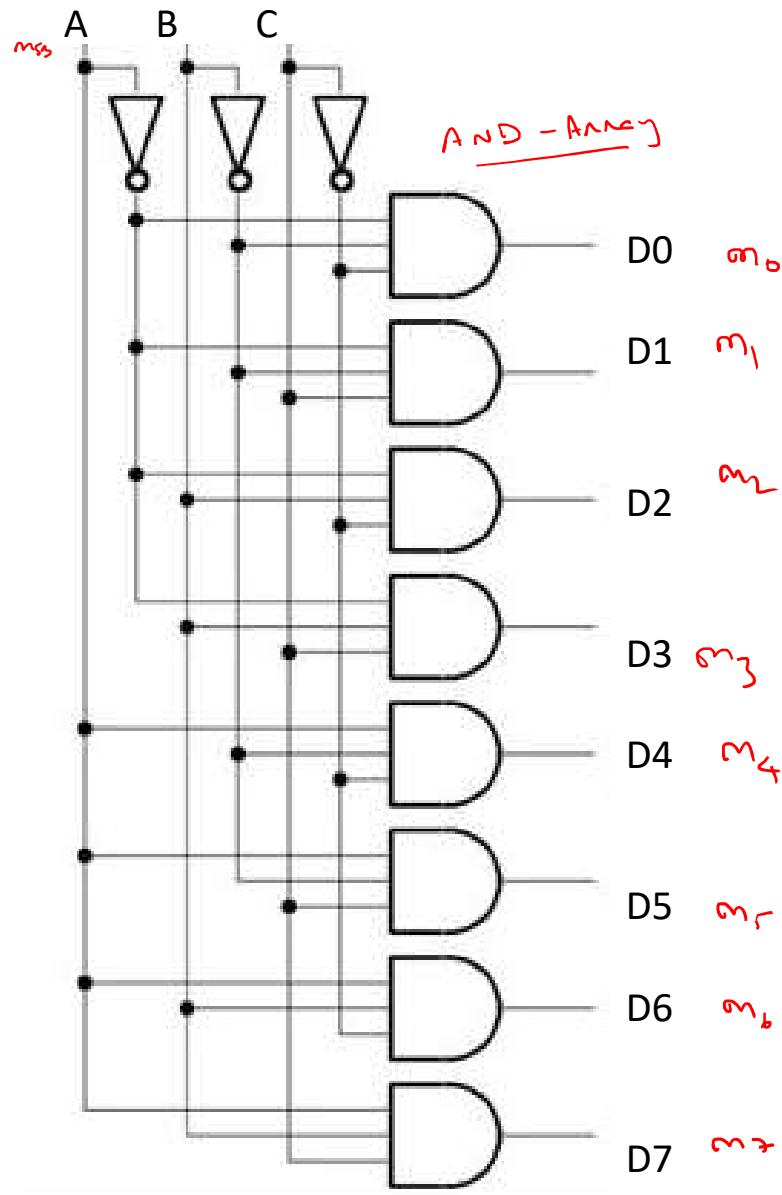
$$D_6 =$$

$$D_7 =$$

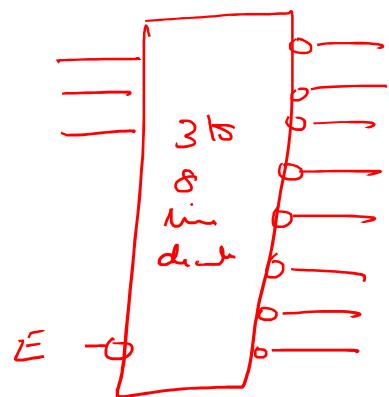
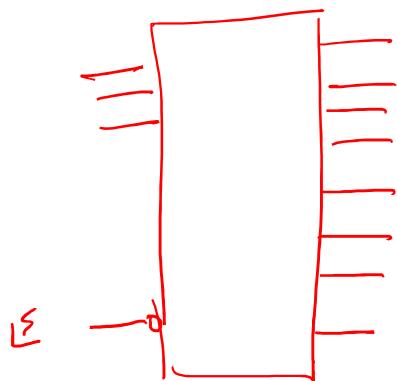
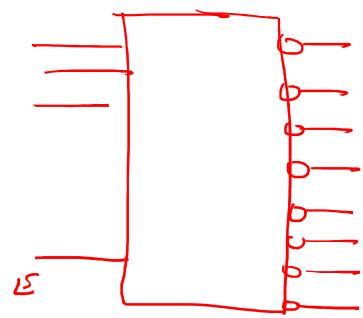
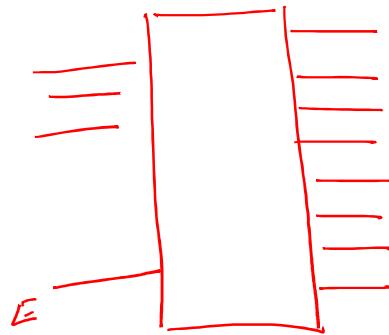


Action High O/P's

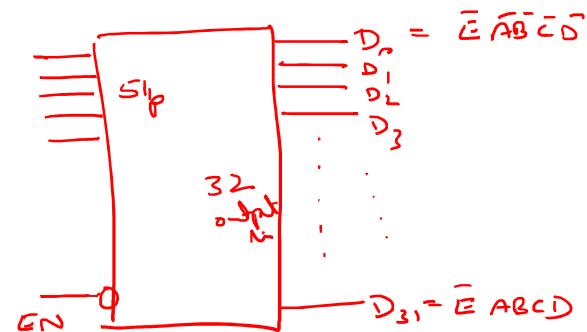
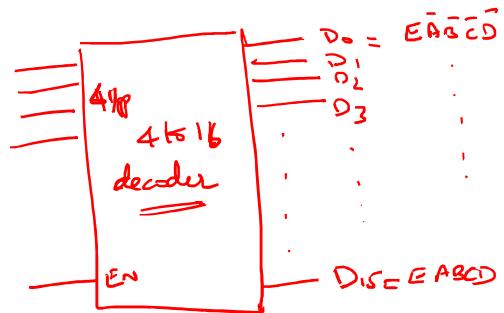
How do you input enable E  
to the circuit?



## Block diagram of 3-to-8 line decoder



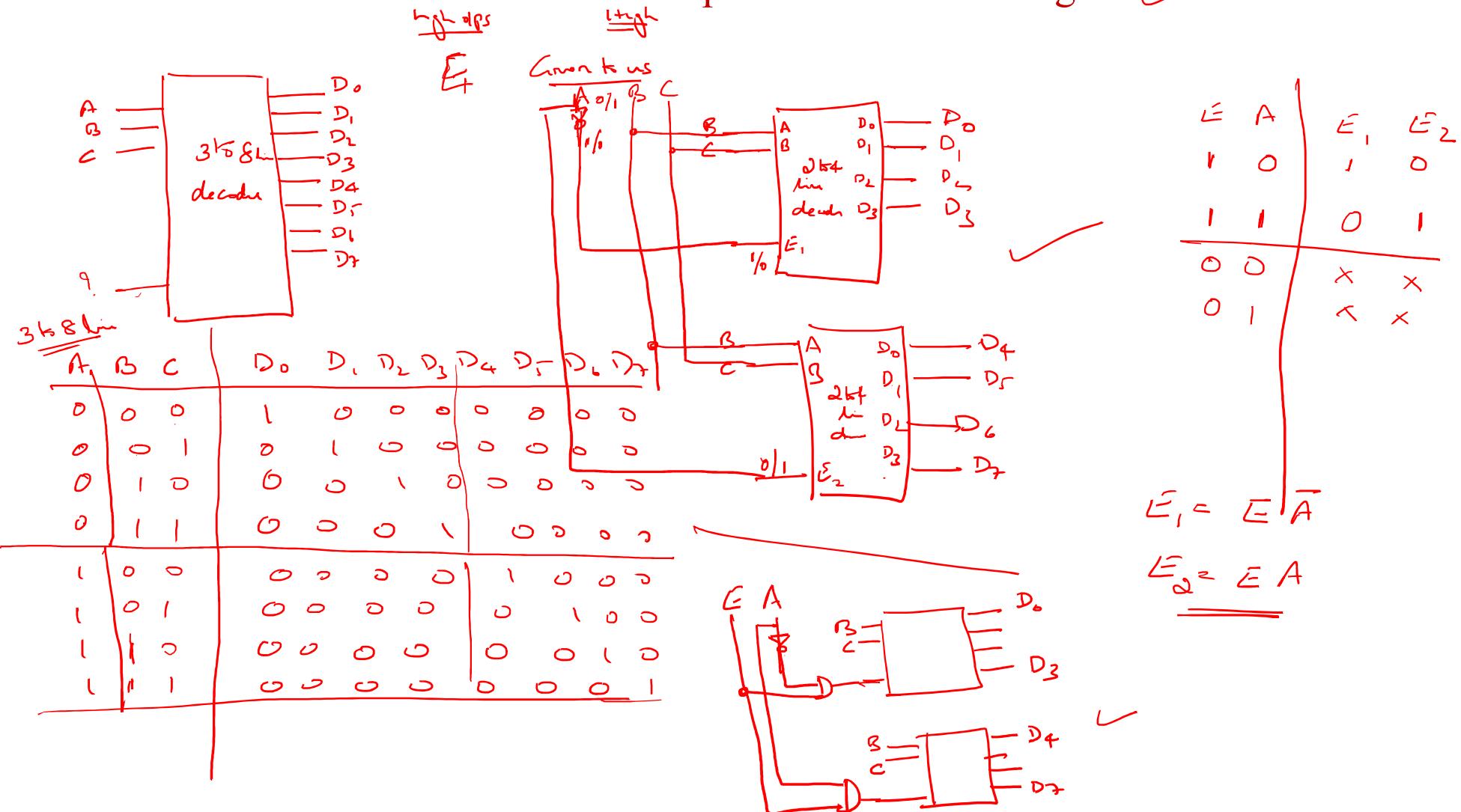
# Discussion of 4-to-16 and 5-to-32 line decoders



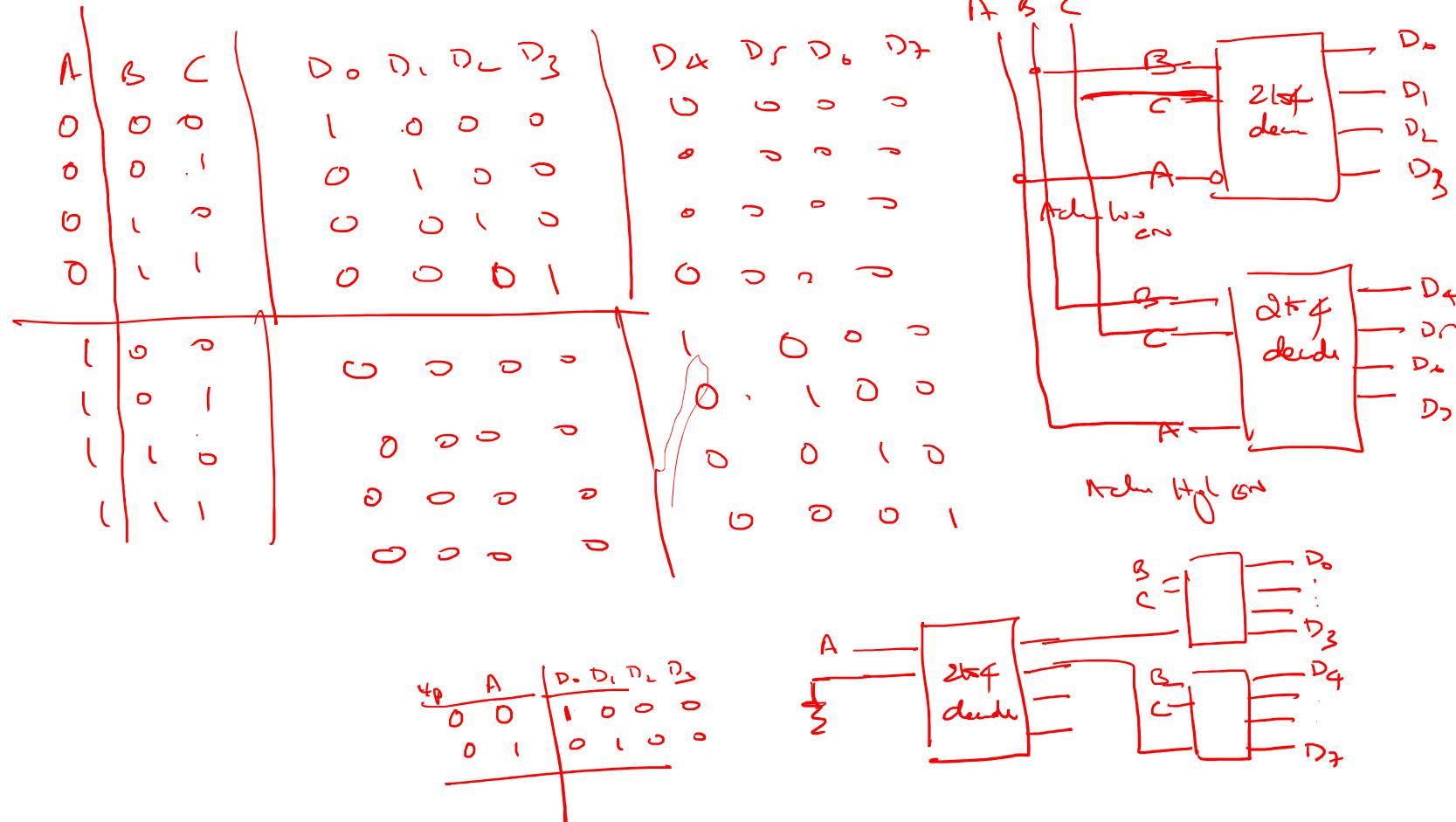
Design 3-to-8 line decoder using minimum number of:

1. 2-to-4 decoders with enable input and one external gate
2. 2-to-4 decoders with enable inputs only

Design 3-to-8 line decoder using minimum number of:  
2-to-4 decoders with enable input and one external gate ✓



# Design 3-to-8 line decoder using minimum number of 2-to-4 decoders only



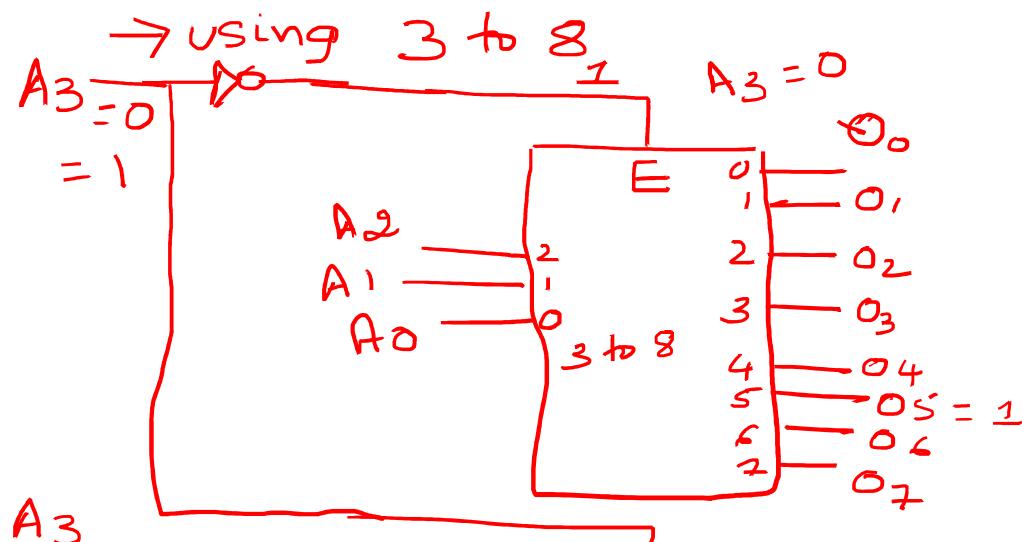
Design 4-to-16 line decoder using minimum number of

- 1. 3-to-8 decoders with enable input and one external gate
- 2. Only 3-to-8 and 2-to-4 line decoders with enable inputs
- 3. Only 2-to-4 line decoders with enable inputs

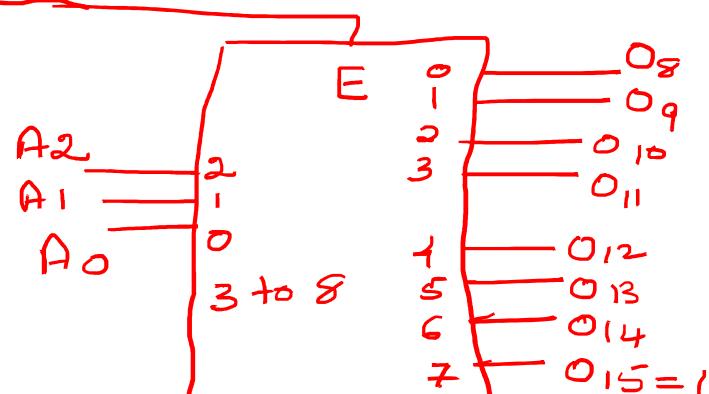
# Design 4-to-16 line decoder

# Design 4-to-16 line decoder

1



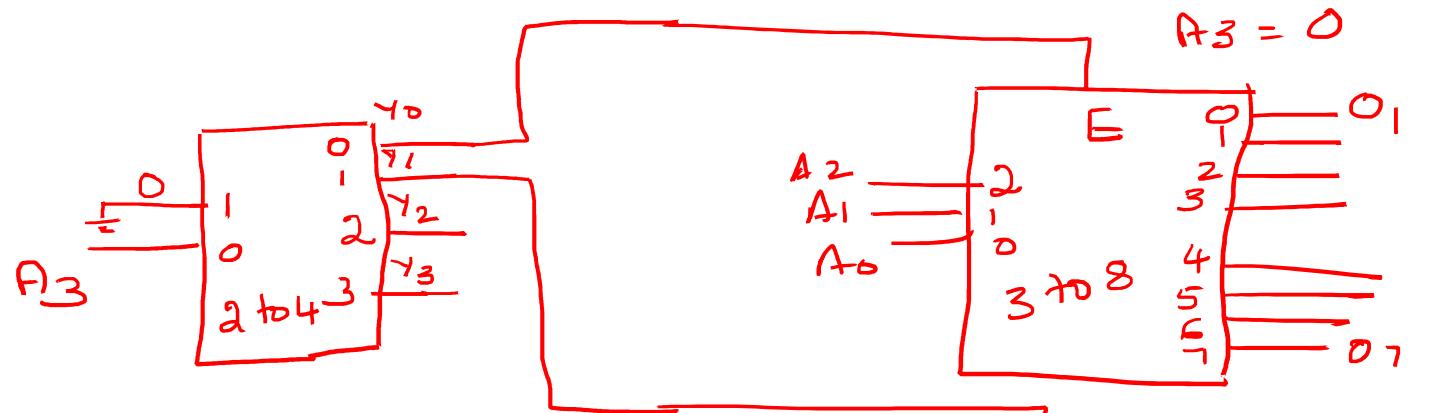
$$\begin{aligned}
 &A_3 \ A_2 \ A_1 \ A_0 \\
 &0 \ 1 \ 0 \ 1 \Rightarrow O_5 \\
 &1 \ 1 \ 1 \ 1 \Rightarrow O_{15}
 \end{aligned}$$



$$A_3 = 1$$

# Design 4-to-16 line decoder

(2) 3-to-8 decoder & 2-to-4 decoder



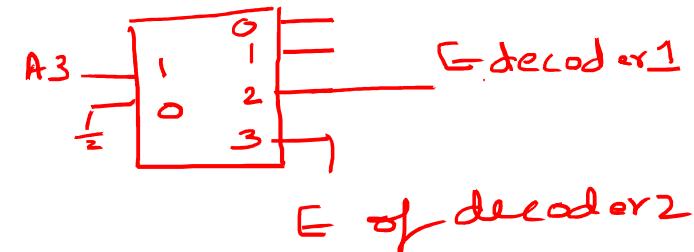
$$\begin{array}{c} 0 \ A_3 \\ \hline 0 \ 0 - Y_0 \\ 0 \ 1 - Y_1 \end{array}$$

$1 \ 0 \times$   
 $1 \ 1 \times$

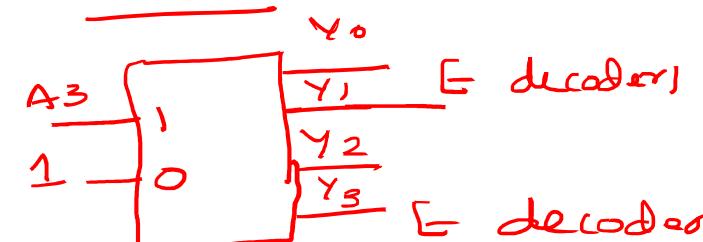
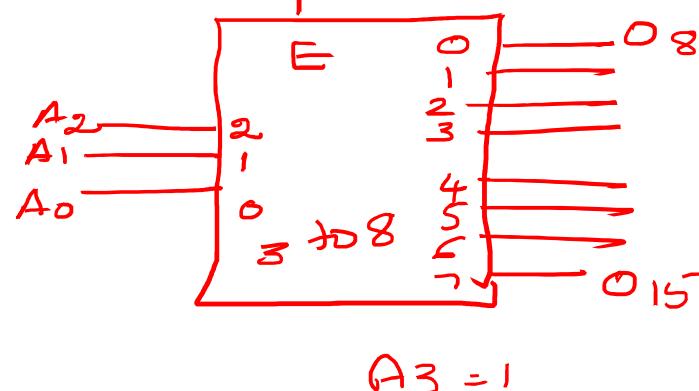
will never occur

$$Y_2 = Y_3 = 0$$

Soln 2

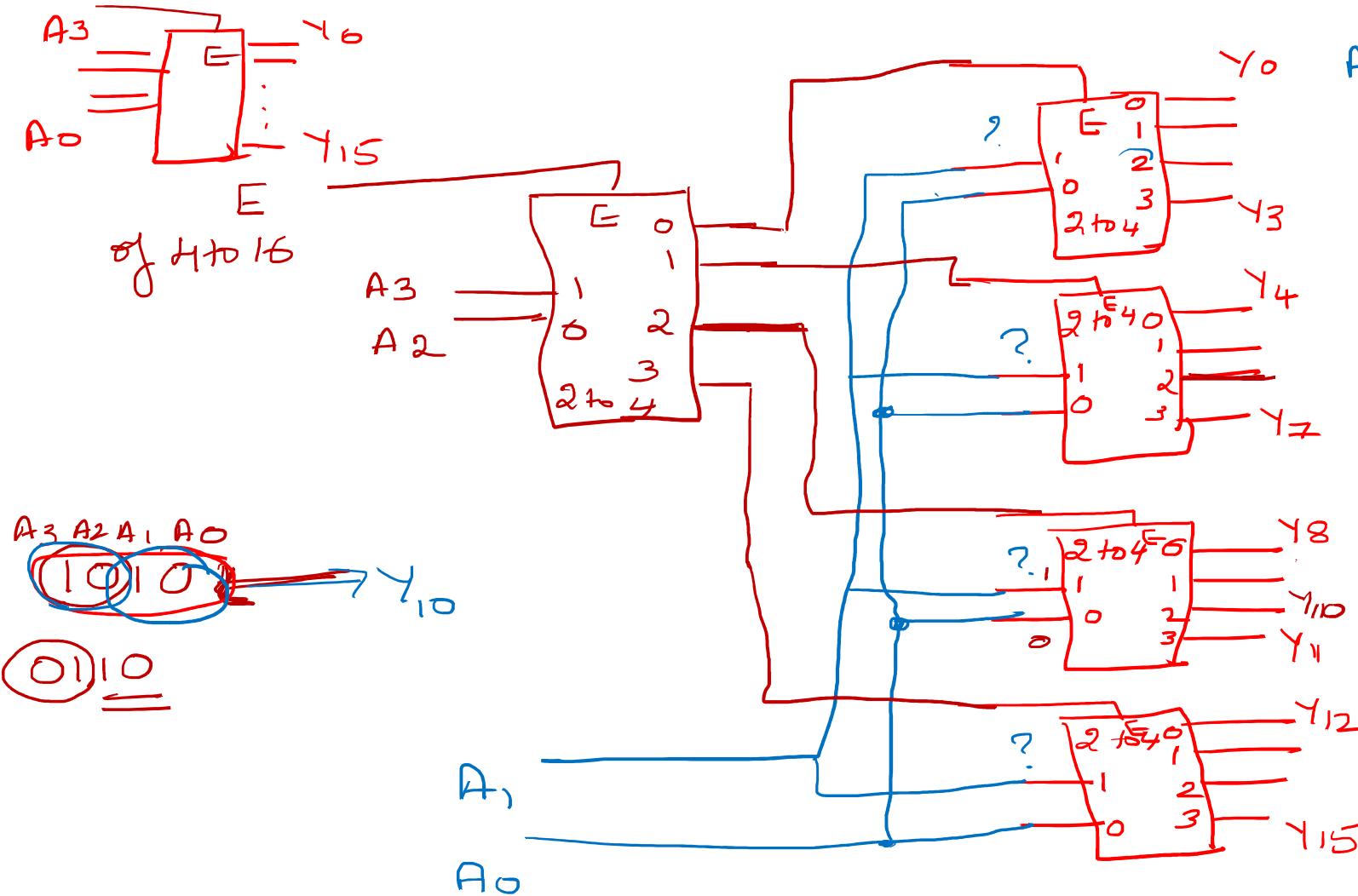


Soln 3



$$\begin{array}{c} \text{two inputs} \\ \hline 0 \ 1 - Y_1 \\ 1 \ 1 - Y_3 \end{array}$$

4 to 16 using only 2 to 4 decoders



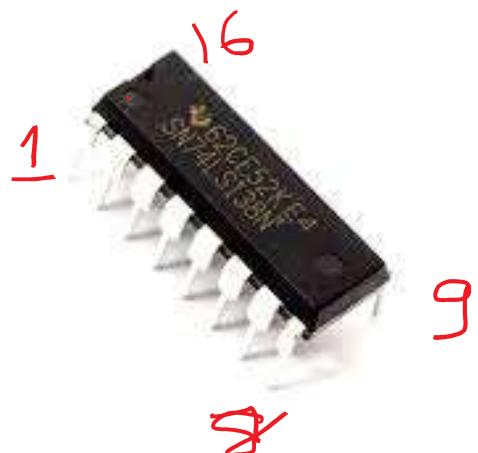
$$A_3 A_2 = 00 \\ \textcircled{0} \textcircled{0} \textcircled{0} \textcircled{0} \rightarrow \textcircled{0} \textcircled{0} \textcircled{1} \textcircled{1}$$

$$A_3 A_2 = 01 \\ \textcircled{0} \textcircled{1} \textcircled{0} \textcircled{0} - \textcircled{0} \textcircled{1} \textcircled{1} \textcircled{1}$$

$$A_3 A_2 = 10 \\ \textcircled{1} \textcircled{0} \textcircled{0} \textcircled{0} - \textcircled{1} \textcircled{0} \textcircled{1} \textcircled{1}$$

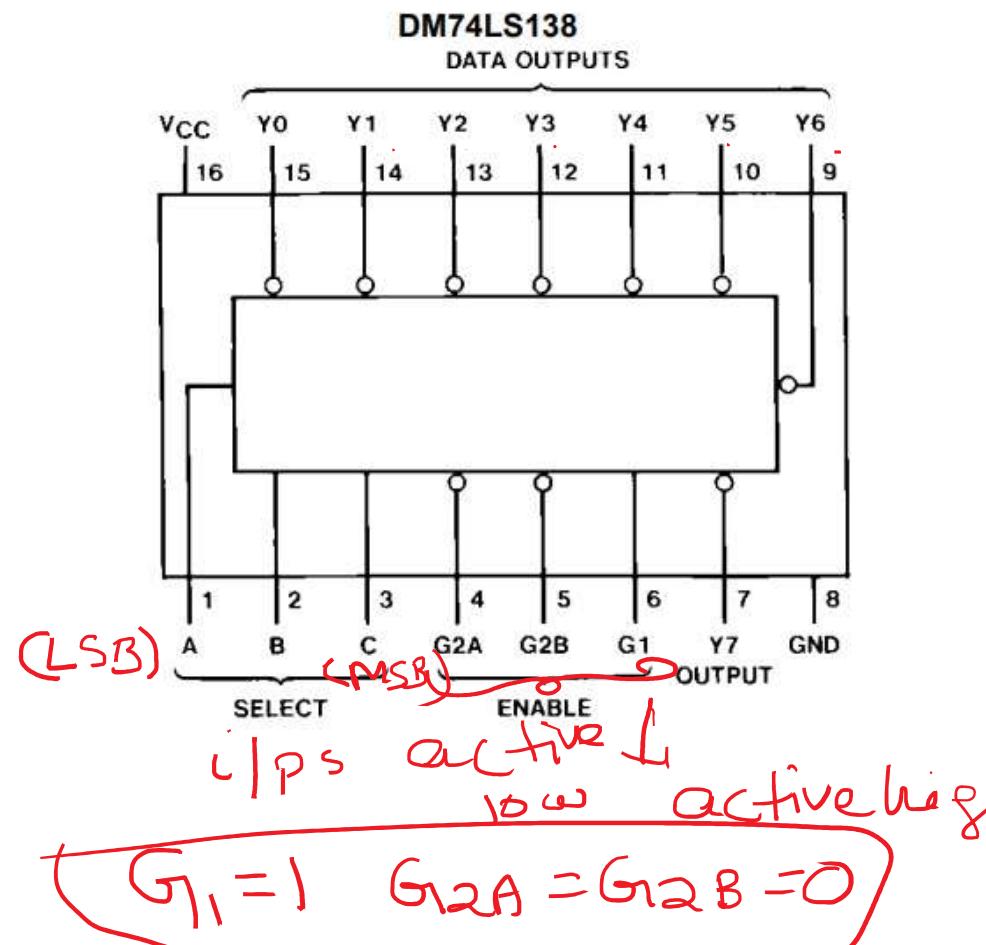
$$A_3 A_2 = 11 \\ \textcircled{1} \textcircled{1} \textcircled{0} \textcircled{0} - \textcircled{1} \textcircled{1} \textcircled{1} \textcircled{1}$$

# 74138 IC: 3-to-8 line decoder with active low output

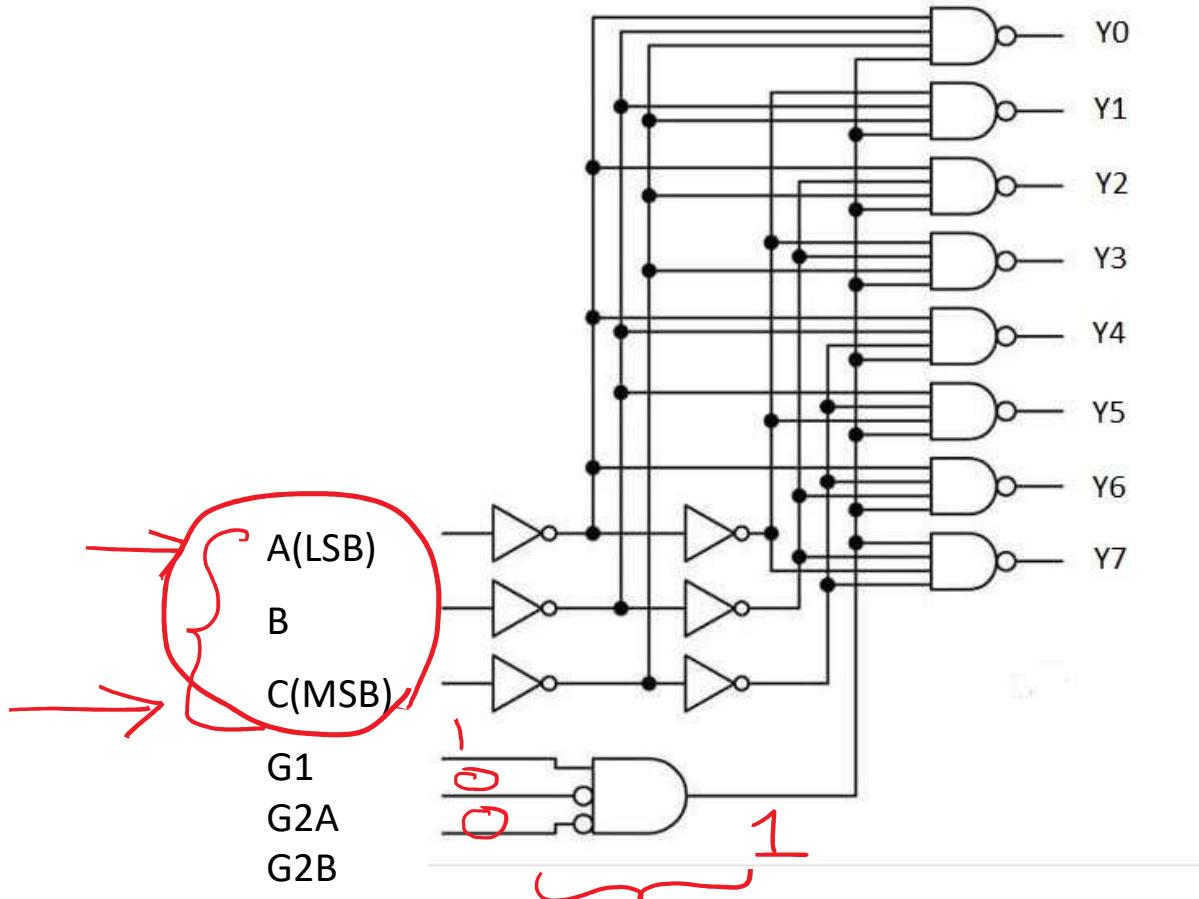


$$000 \Rightarrow Y_0 = 0$$

$$Y_1 \text{ to } Y_7 = 1$$

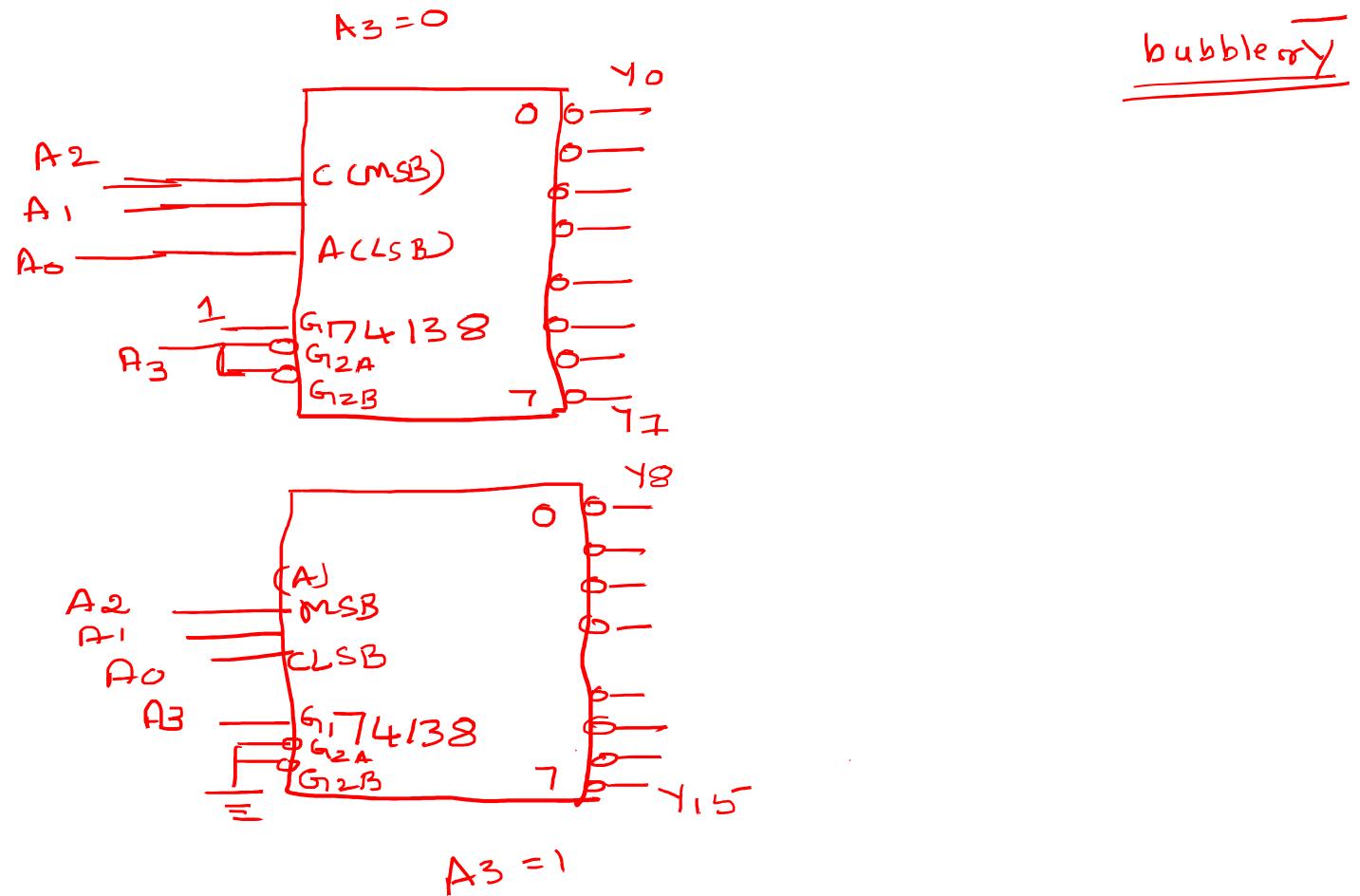


# 74138 IC internal diagram



# Design 4-to-16 decoder using minimum of 74138 ICs ONLY

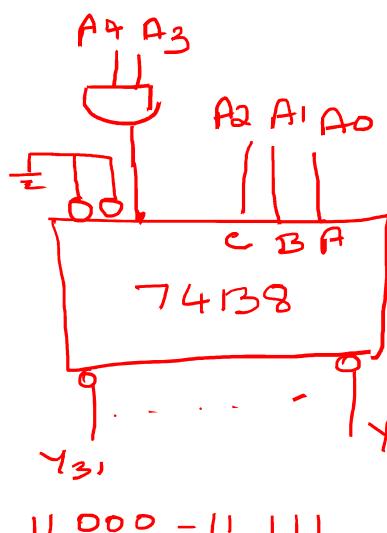
$A_3 A_2 A_1 A_0$



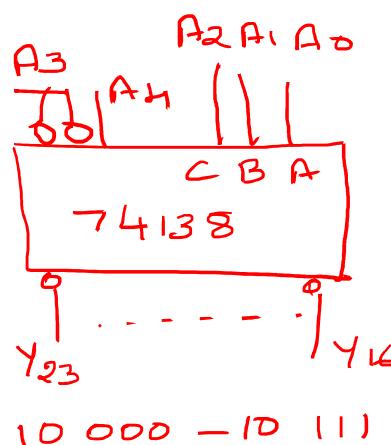
Design 5-to-32 decoder with active low output  
 using minimum of 74138 ICs and one external gate  
**ONLY**

$$\underline{A_4 A_3 A_2 A_1 A_0 \Rightarrow Y_0 \text{ to } Y_{31}}$$

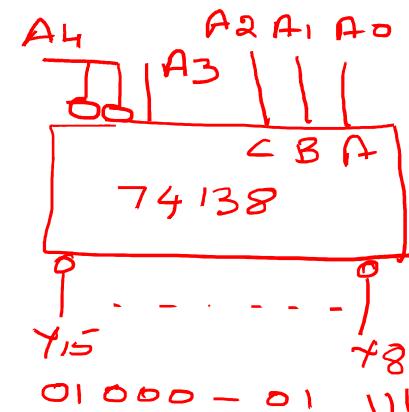
$$A_4 A_3 = 11$$



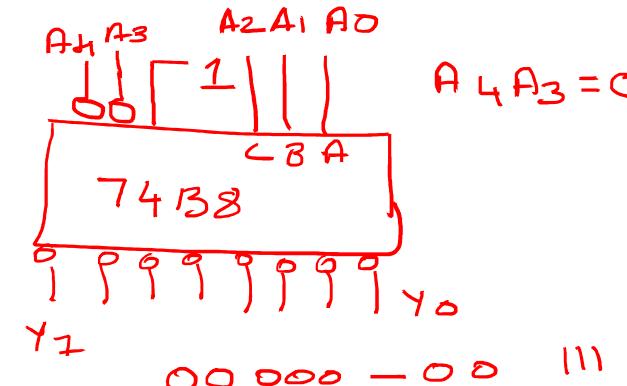
$$A_4 A_3 = 10$$



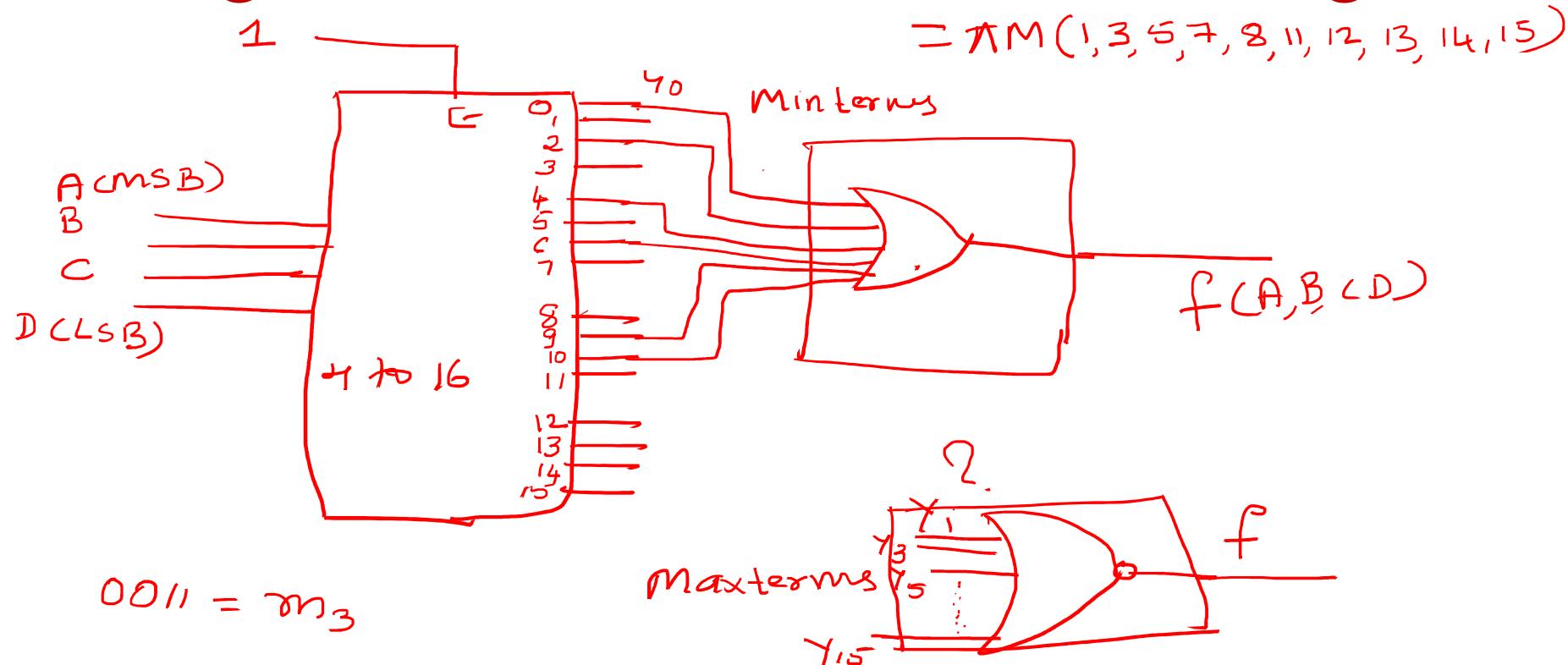
$$A_4 A_3 = 01$$



$$A_4 A_3 = 00$$



Implement  $f(A, B, C, D) = \sum m(0, 2, 4, 6, 9, 10)$  ✓  
 using suitable decoder and external gates



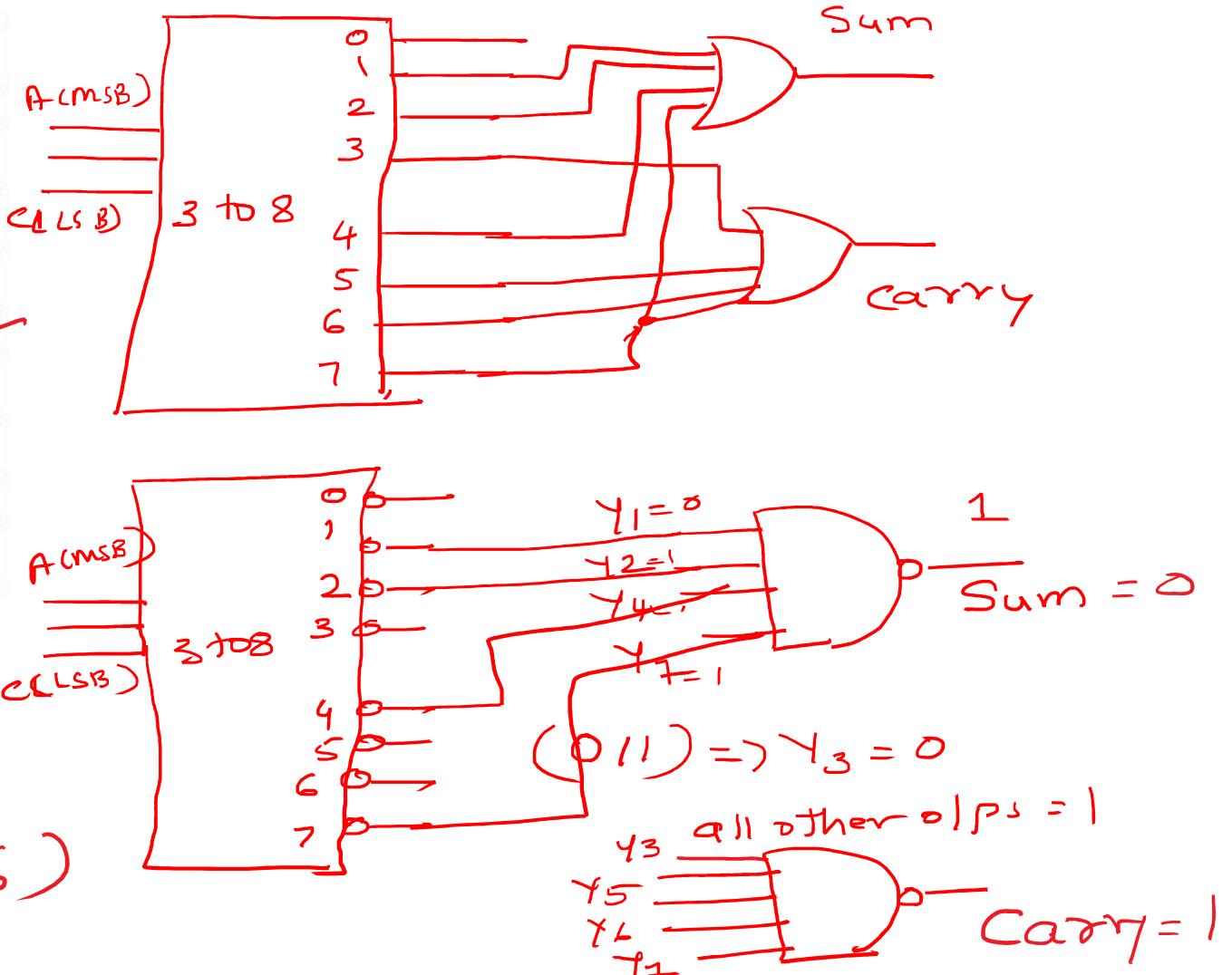
Design a full adder using 3-to-8 line decoder and external gates

Inputs			Outputs	
A	B	$C_{in}$	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

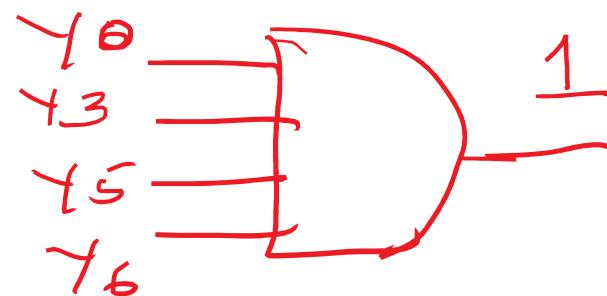
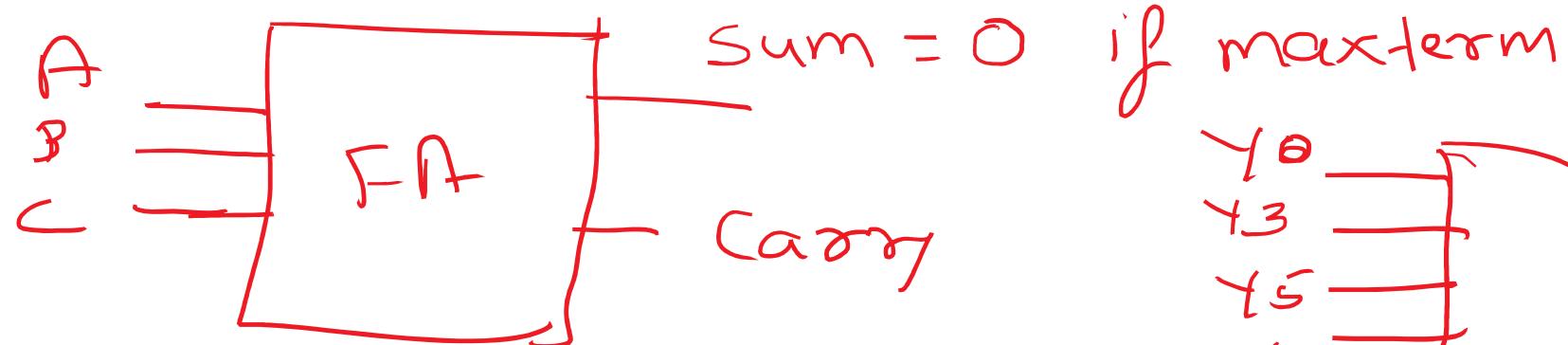
$$\text{Sum} = \sum m(1, 2, 4, 7)$$

$$\text{Carry} = \sum m(3, 5, 6, 7)$$

$$\text{Sum} = \pi m(0, 3, 5, 6)$$



$\text{Sum} = \text{AM}(0, 3, 5, 6) \quad Y_0 \quad Y_3 \quad Y_5 \quad Y_6$



Decoder	$\Sigma_m$	$\pi_M$
Active-high	OR	NOR
Active-low	NAND	AND

Realize  $f_1(x,y,z) = \prod M(1,3,6,7)$  using

- a. 3-to-8 line decoder with active high output and suitable gates
- b. 74138 decoder and suitable gates

•

Design a code converter to convert a decimal digit represented in 84-2-1 code to a decimal digit represented in excess-3 code using 74138 decoder and external gates..

Code converter design....

Code converter design...

Design  $f(x,y,z) = x' + y'z$  using 3-to-8 line decoder and external gates.

## Encoder:

- Combinational circuit that performs inverse operation of a decoder.
- Encoder has  $2^n$  (or fewer) input lines and n output lines. Ex: 4-to-2 line, 8-to-3 line...etc
- 4-to-2 encoder is given below:

Truth table

Inputs				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

Circuit

Write the truth table and circuit for 8-to-3 line encoder

Design a 4 –to-2 line priority encoder

4-to-2 line priority encoder contd...

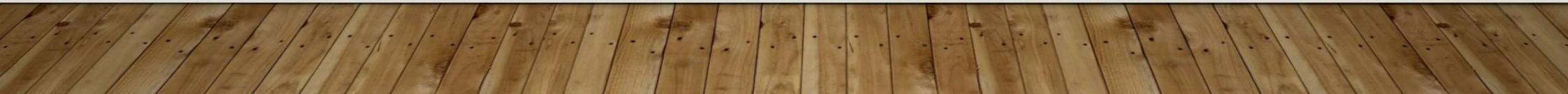
- Any questions?

## **DECODERS AND ENCODERS CONTD**

---

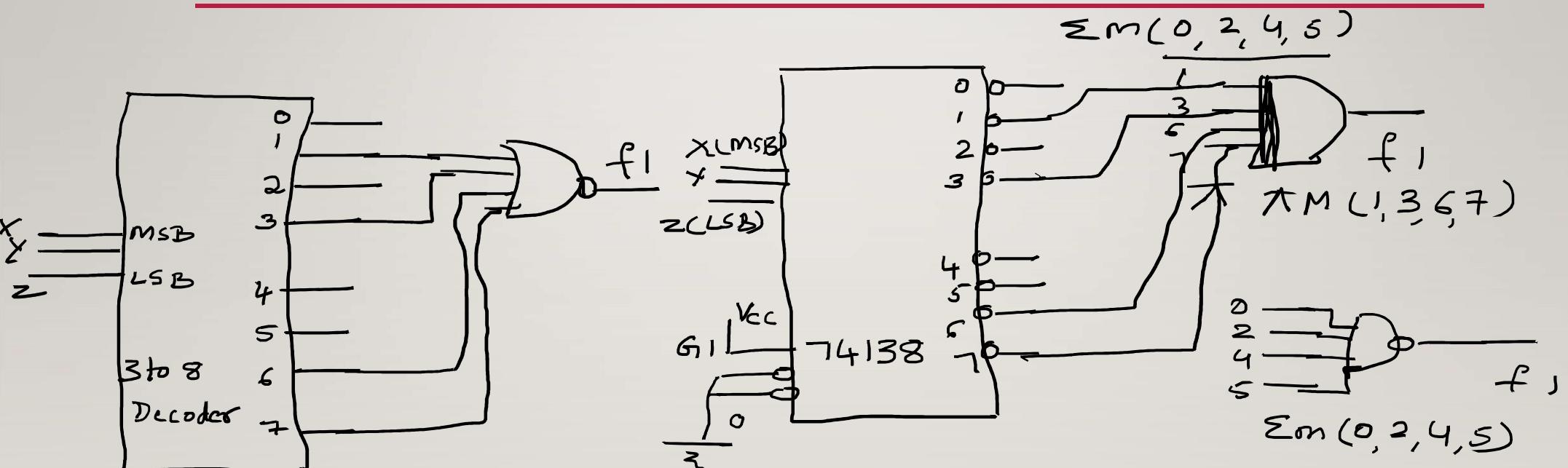
**STUDENTS ARE ADVISED TO WRITE DOWN THE NOTES FOR EVERY LECTURE**

**NOV. 2<sup>ND</sup> 2021 (10.30 AM TO 12.30 PM)**



Realize  $f_1(x,y,z) = \Pi M(1,3,6,7)$  using  $= \sum m(0,2,4,5) \Rightarrow OR$

- 3-to-8 line decoder with active high output and suitable gates
- 74138 decoder and suitable gates



Design a code converter to convert a decimal digit represented in 84-2-1 code to a decimal digit represented in excess-3 code using 74138 decoder and external gates..

→ complete the truthtable

84-2-1 A B C D	Excess-3 e3 e2 e1 e0
0 0 0 0 0	0 0 0 1 1
0 1 1 1 7	0 1 0 0 0
0 1 1 0 6	0 1 0 1 0
0 1 0 1 5	0 1 1 0 0
0 1 0 0 4	0 1 1 1 0
1 0 1 1 11	{ 1 0 0 0 0
1 0 1 0 10	{ 1 0 0 0 1
1 0 0 1 9	{ 1 0 1 0 0
1 0 0 0 8	{ 1 0 1 1 0
1 1 1 1 15	{ 1 1 0 0 0

$$e_3 = A$$

$$e_3 = \sum m(0, 1, 2, 3, 4) = \pi m (5 \text{ terms})$$

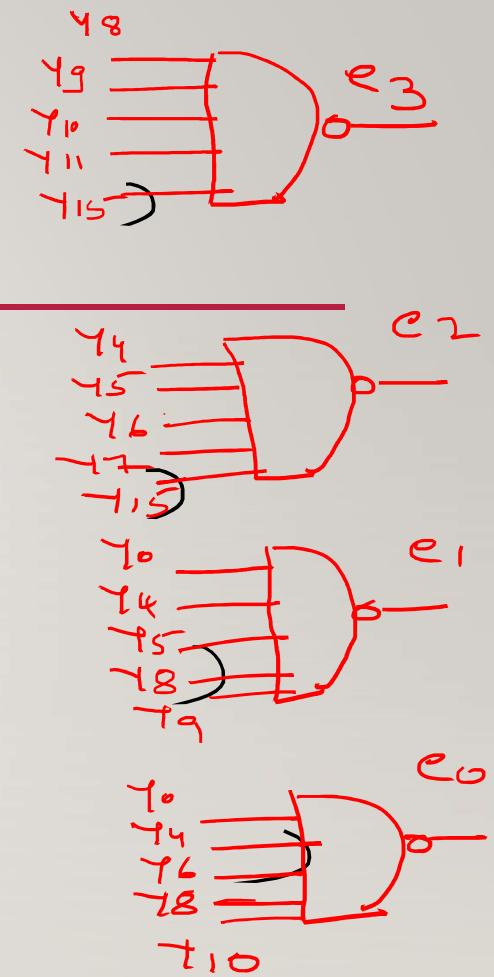
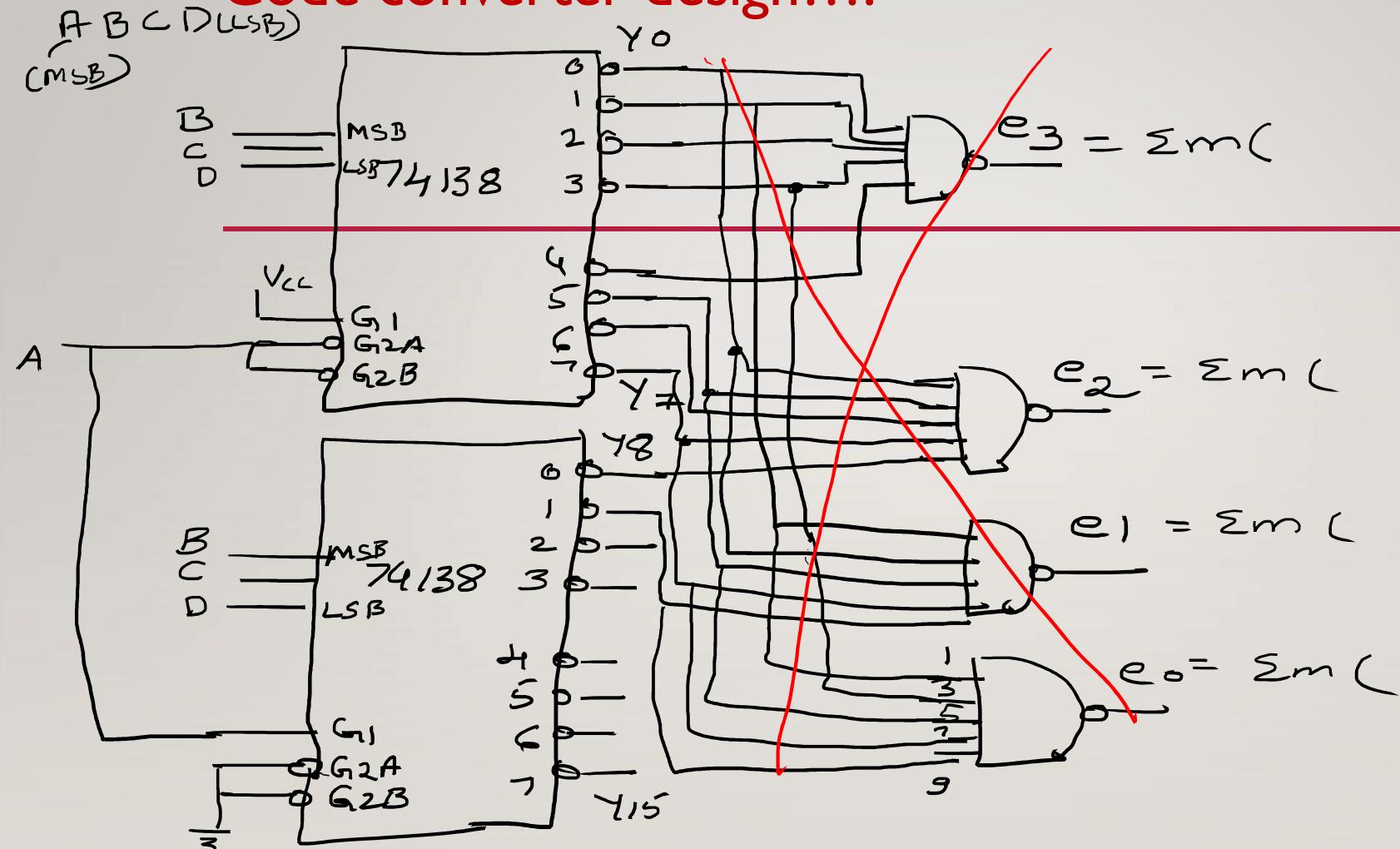
$$e_2 = \sum m(0, 5, 6, 7, 8) = B$$

$$e_1 = \sum m(0, 4, 5, 8, 9) = \bar{C}$$

$$e_0 = \sum m(1, 3, 5, 7, 9) = \bar{D}$$

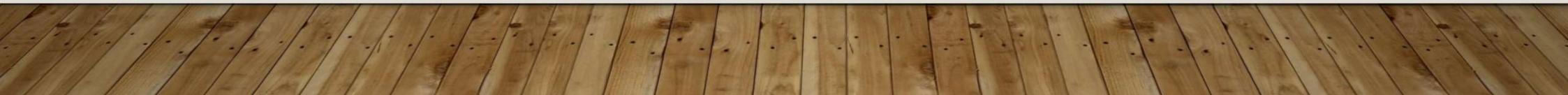
$$e_3(A, B, C, D)$$

## Code converter design....



# Code converter design...

---



Design  $f(x,y,z) = x' + y'z$  using 3-to-8 line decoder and external gates.

$$f(x,y,z) = \overline{\quad}$$

assumed as active high output

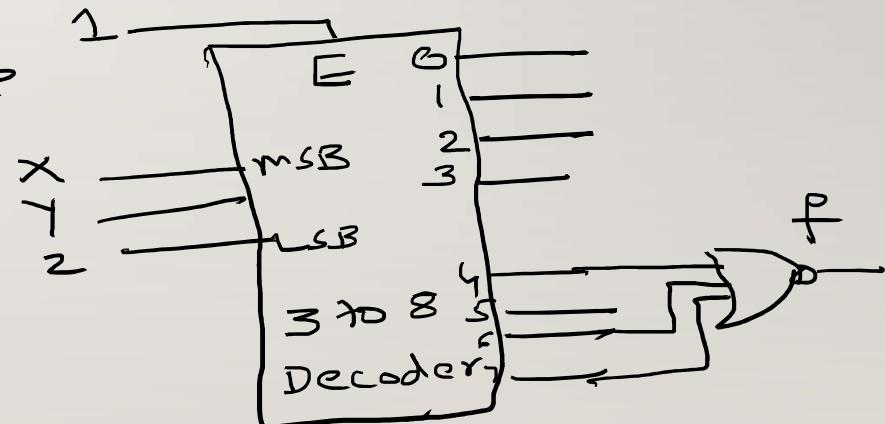
Minterms for  $x'$  ( $x=0, y=2$  - don't care) =  $\Sigma m(0, 1, 2, 3)$

$$\begin{array}{ccc} x & \bar{y} & z \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{array} \quad \bar{y}z \text{ (x=0 or 1, y=0 & z=1)} = 1, 5$$

001, 101

$$f(x,y,z) = \bar{x} + \bar{y}z - \Sigma m(0, 1, 2, 3, 5) = \pi m(4, 6, 7)$$

Can also be written from truth table



## ENCODER:

- Combinational circuit that performs inverse operation of a decoder.
- Encoder has  $2^n$  (or fewer) input lines and n output lines. Ex: 4-to-2 line, 8-to-3 line...etc
- 4-to-2 encoder is given below:

$2^4 \rightarrow 2$

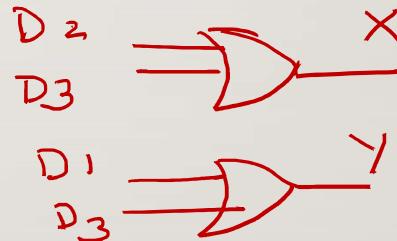
Truth table

Inputs				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 ✓

$$x = D_2 + D_3$$

$$y = D_1 + D_3$$

Circuit



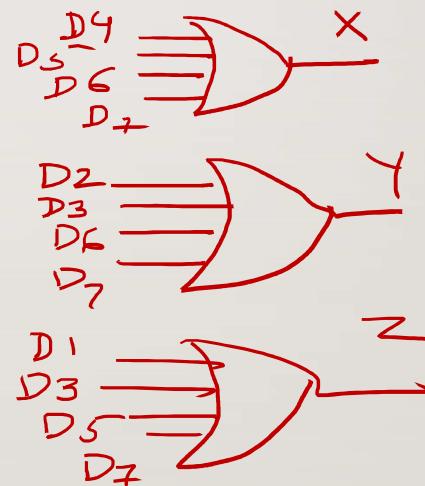
Write the truth table and circuit for 8-to-3 line encoder

$D_0$	$D_1$	$D_2$	$D_3$	$D_4$	$D_5$	$D_6$	$D_7$	$X$	$Y$	$Z$
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	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

$$X = D_4 + D_5 + D_6 + D_7$$

$$Y = D_2 + D_3 + D_6 + D_7$$

$$Z = D_1 + D_3 + D_5 + D_7$$



## Design a 4-to-2 line priority encoder

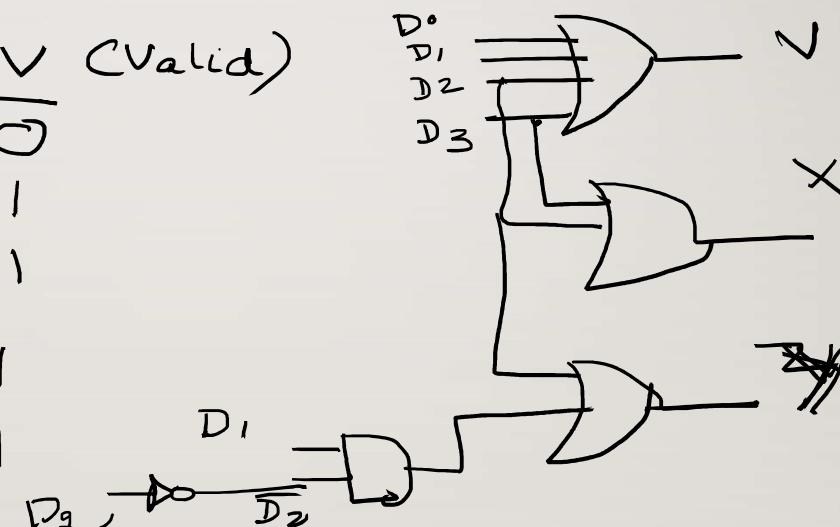
$$D_0 < D_1 < D_2 < D_3$$

$D_0$	$D_1$	$D_2$	$D_3$	$X$	$Y$	$V$	$C_{Valid}$
0	0	0	0	0	0	0	1
1	0	0	0	0	0	1	1
✓	0	1	0	0	0	1	1
✓	0	0	1	1	0	1	1
0	0	0	0	0	0	0	0
0	0	1	0	0	1	1	1
0	1	0	0	1	0	1	1
0	0	0	1	0	1	1	1

$$V = D_0 + D_1 + D_2 + D_3$$

$$X = D_3 + \overline{D_2} \overline{D_3} = D_2 + D_3$$

$$Y = \overline{D_1} \overline{D_2} \overline{D_3} + D_3 = D_3 + D_1 \overline{D_2}$$



4-to-2 line priority encoder contd...

---

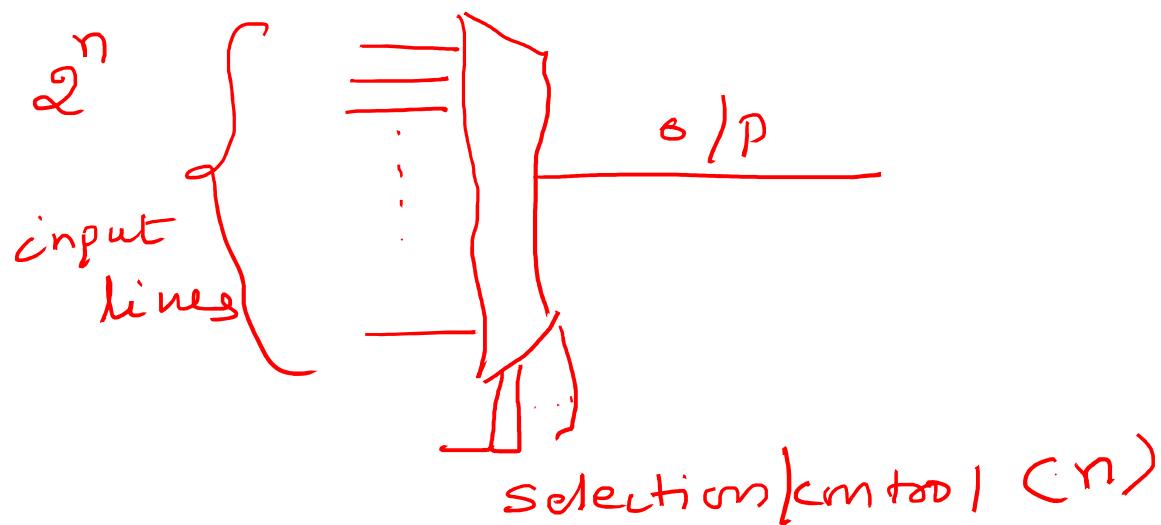
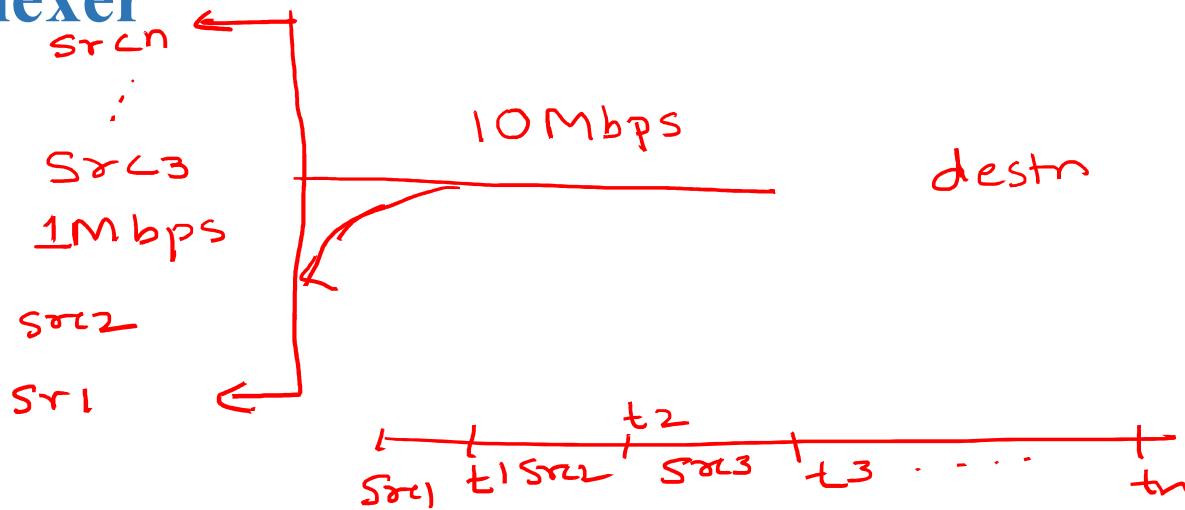
---

- Any questions?

# Multiplexers

Students are advised to write down the notes for every lecture

## Multiplexer



## Multiplexer

- Multiplexer is an useful MSI device and are also called as data selectors.
- Multiplexer selects one of its  $2^n$  input line and directs it to a single output line .
- n-bit select lines decide which input line is to be selected.
- Examples: 2-to-1 line MUX, 4-to-1 line MUX, 8-to-1 line MUX, 16-to-line MUX.

$2^n$  :  $\chi$ ,  $n \Rightarrow$  select lines

2 : 1

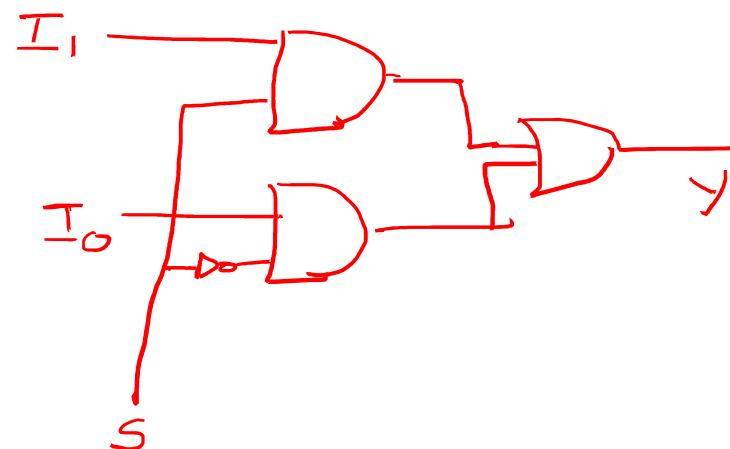
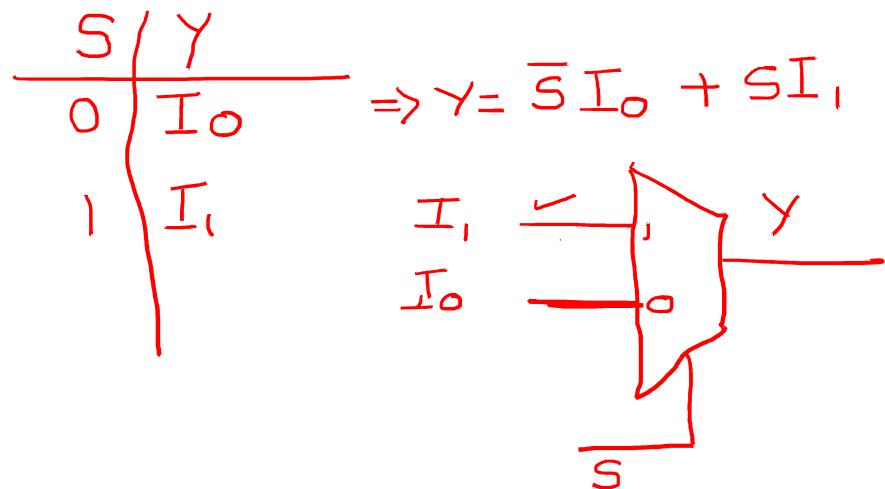
4 : 1

8 : 1

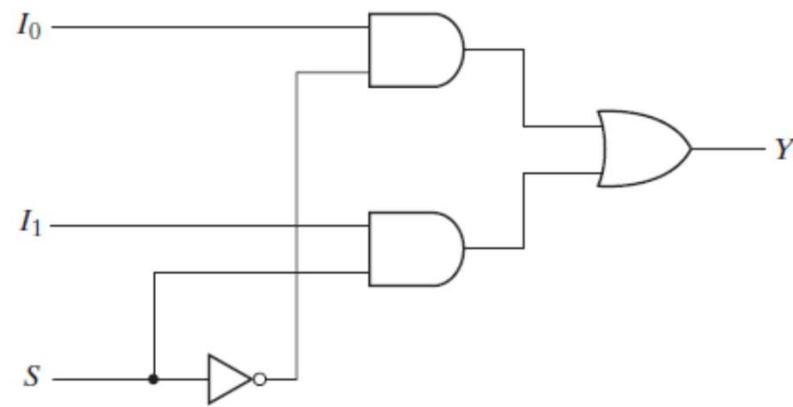
16 : 1

## 2-to-1 line MUX

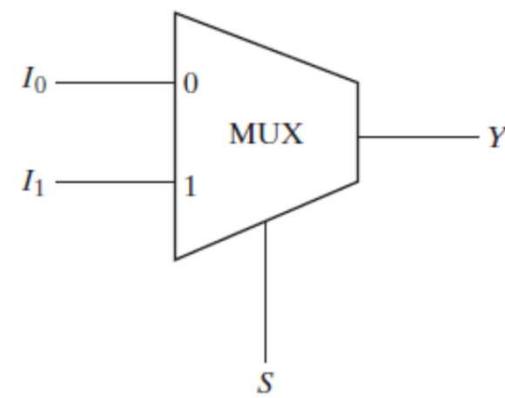
- S- selection input, y is the output, I<sub>1</sub> and I<sub>0</sub> are inputs
- Symbol/block diagram, function table, output expressions and circuit is given :



## 2-to-1 line MUX



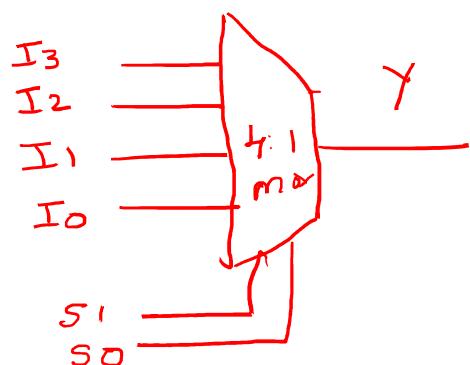
(a) Logic diagram



(b) Block diagram

## 4-to-1 line multiplexer

- Write the Symbol/block diagram, Function table, output expressions and circuit .



$S_1$	$S_0$	$Y$
0	0	$I_0$
0	1	$I_1$
1	0	$I_2$
1	1	$I_3$

Assume

$$I_3 = 1$$

$$\boxed{I_2} = 1$$

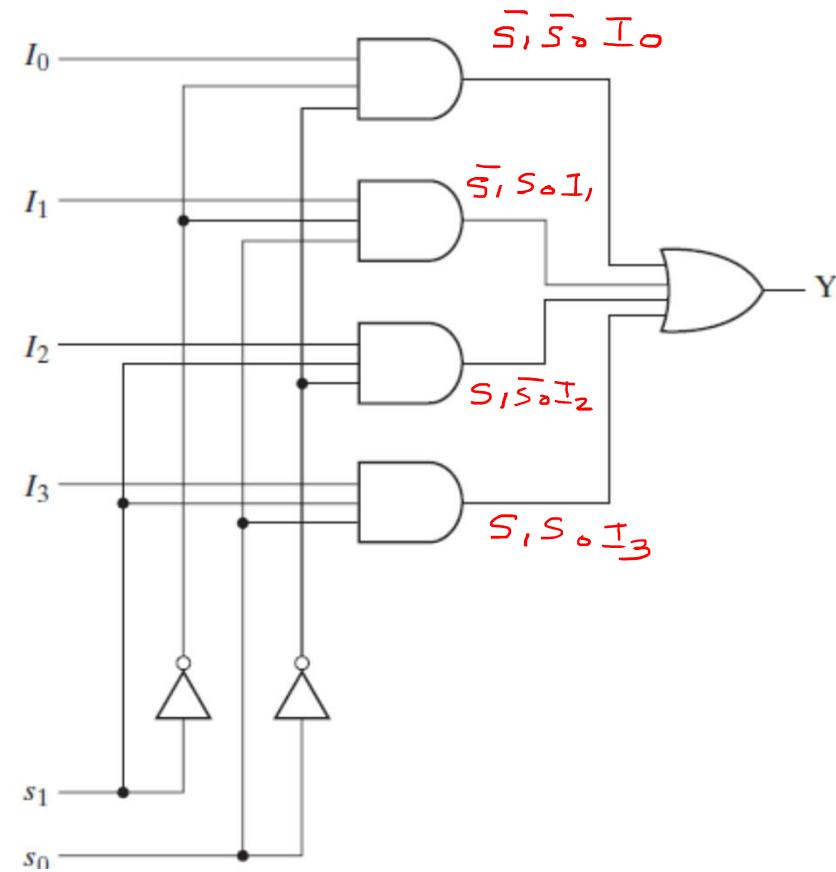
$$I_1 = 0$$

$$I_0 = 0$$

$$S_1, S_0 = 10, \quad Y = ? \underline{1}$$

$$Y = \bar{S}_1 \bar{S}_0 \cdot I_0 + \bar{S}_1 S_0 \cdot I_1 \\ + S_1 \bar{S}_0 \cdot I_2 + S_1 S_0 \cdot I_3$$

## 4-to-1 line multiplexer



## 8-to-1 line multiplexer

- Write the Symbol/block diagram, function table, output expressions and circuit

Select lines:  $S_2 S_1 S_0$

$$Y = \bar{S}_2 \bar{S}_1 \bar{S}_0 I_0 + \bar{S}_2 \bar{S}_1 S_0 I_1 + \bar{S}_2 S_1 \bar{S}_0 I_2 + \bar{S}_2 S_1 S_0 I_3 \\ S_2 \bar{S}_1 \bar{S}_0 I_4 + S_2 \bar{S}_1 S_0 I_5 + S_2 S_1 \bar{S}_0 I_6 + S_2 S_1 S_0 I_7$$

→ Draw 8:1 mux circuit

## Realize 4:1 using only 2:1 MUXs (Multiplexer tree)

For 4:1 mux

$$y = \bar{s}_1 \bar{s}_0 I_0 + \bar{s}_1 s_0 I_1 + s_1 \bar{s}_0 I_2 + s_1 s_0 I_3$$

For 2:1 mux

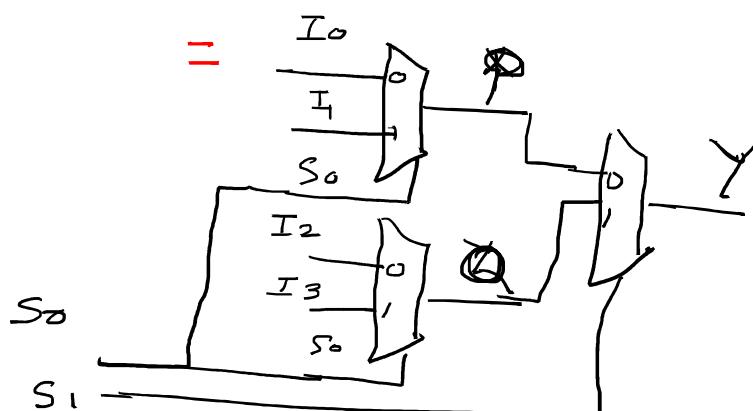
$$y = \bar{s}_0 I_0 + s_0 I_1$$

$$\rightarrow y = \bar{s}_1 (\bar{s}_0 I_0 + s_0 I_1) +$$
  

2:1 mux

$$s_1 (\bar{s}_0 I_2 + s_0 I_3) = \bar{s}_1 \cdot P + s_1 \cdot Q$$
  

2:1 mux



3, 2:1 muxs

## Realize 8:1 using only 2:1 MUXs (Multiplexer tree)

( $4+2+1$ ), 2:1 MUXs

$2^n$ :1 mux using  
2:1 mux  
how many 2:1 mux?

$2^n - 1$  ← Ans?

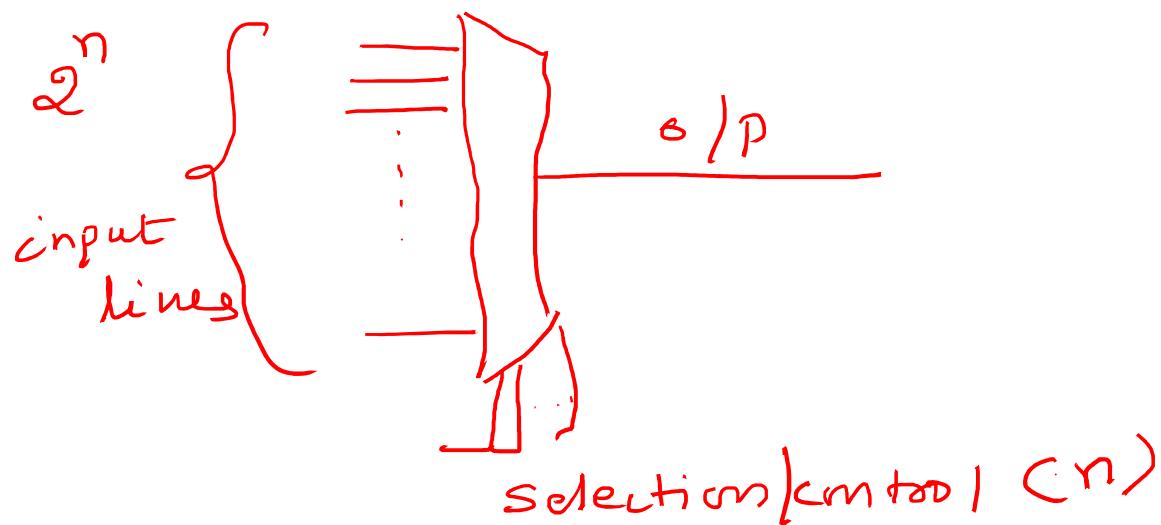
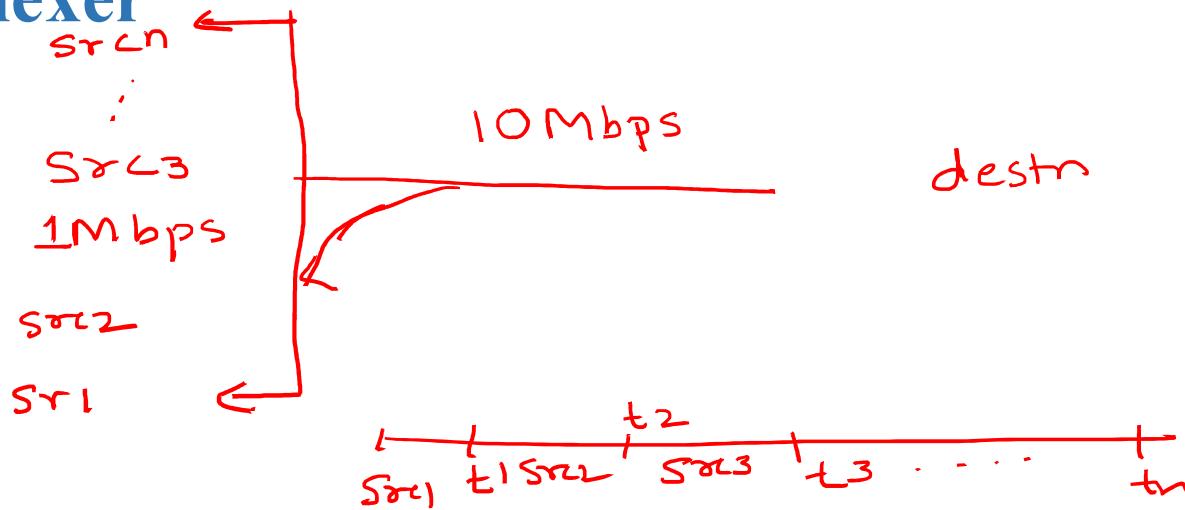
**Realize 8:1 using only 4:1 MUXs and 2:1 MUX**

**Question?**

# Multiplexers

Students are advised to write down the notes for every lecture

## Multiplexer



## Multiplexer

- Multiplexer is an useful MSI device and are also called as data selectors.
- Multiplexer selects one of its  $2^n$  input line and directs it to a single output line .
- n-bit select lines decide which input line is to be selected.
- Examples: 2-to-1 line MUX, 4-to-1 line MUX, 8-to-1 line MUX, 16-to-line MUX.

$2^n$  :  $\chi$ ,  $n \Rightarrow$  select lines

2 : 1

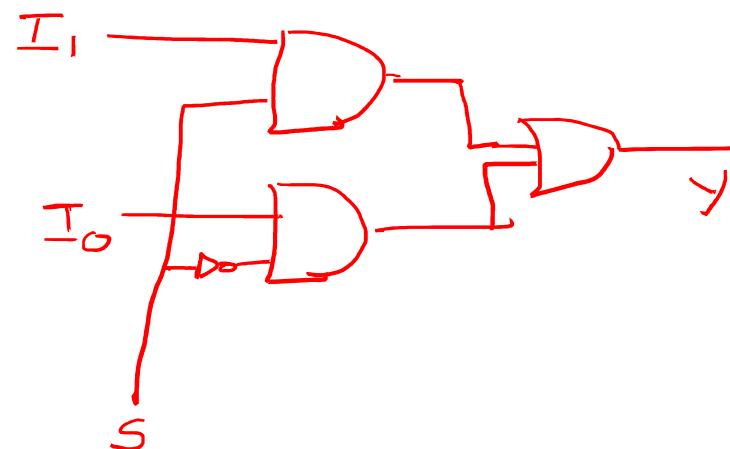
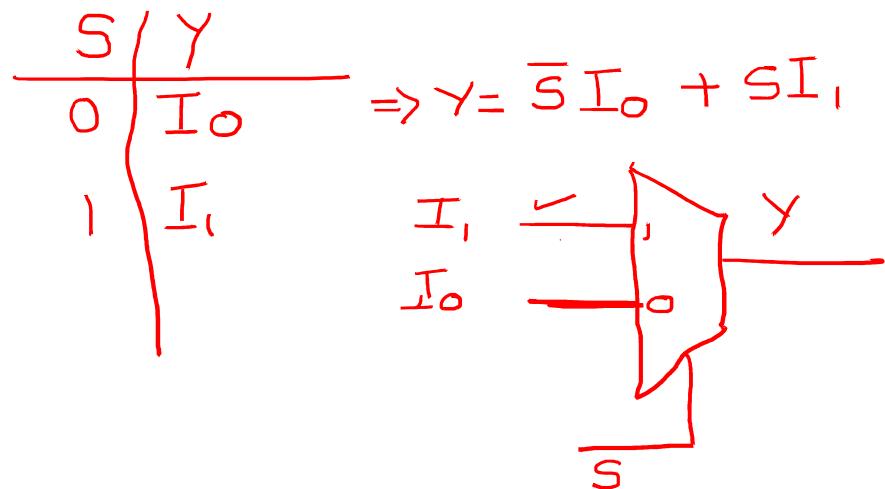
4 : 1

8 : 1

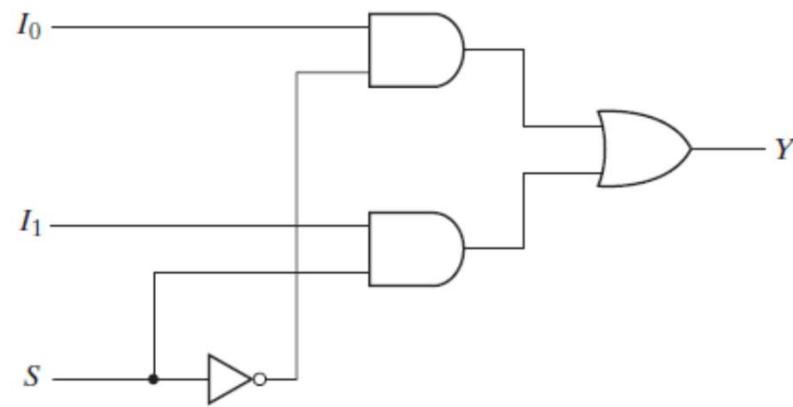
16 : 1

## 2-to-1 line MUX

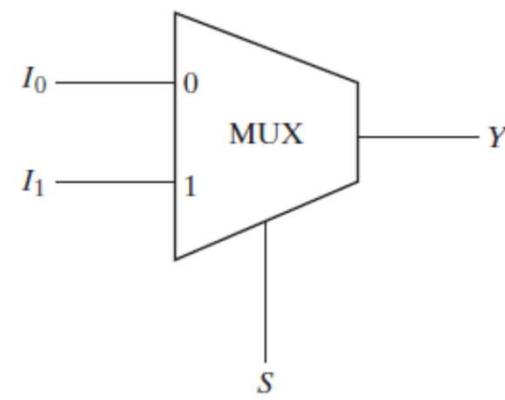
- S- selection input, y is the output, I<sub>1</sub> and I<sub>0</sub> are inputs
- Symbol/block diagram, function table, output expressions and circuit is given :



## 2-to-1 line MUX



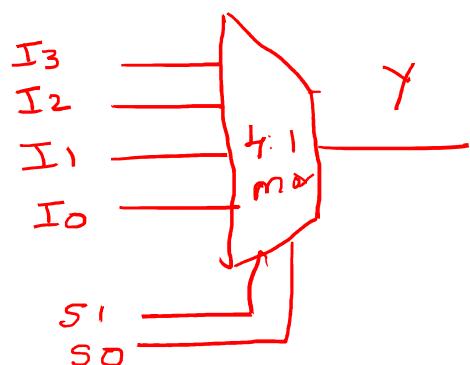
(a) Logic diagram



(b) Block diagram

## 4-to-1 line multiplexer

- Write the Symbol/block diagram, Function table, output expressions and circuit .



$S_1$	$S_0$	$Y$
0	0	$I_0$
0	1	$I_1$
1	0	$I_2$
1	1	$I_3$

Assume

$$I_3 = 1$$

$$\boxed{I_2} = 1$$

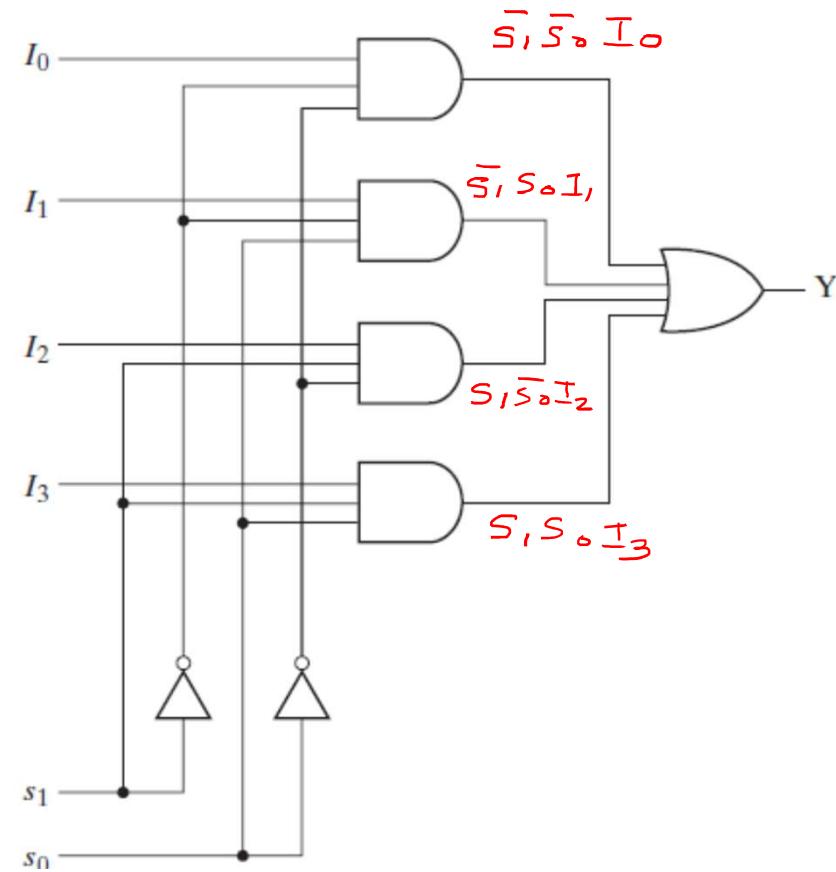
$$I_1 = 0$$

$$I_0 = 0$$

$$S_1, S_0 = 10, \quad Y = ? \underline{1}$$

$$Y = \bar{S}_1 \bar{S}_0 \cdot I_0 + \bar{S}_1 S_0 \cdot I_1 \\ + S_1 \bar{S}_0 \cdot I_2 + S_1 S_0 \cdot I_3$$

## 4-to-1 line multiplexer



## 8-to-1 line multiplexer

- Write the Symbol/block diagram, function table, output expressions and circuit

Select lines:  $S_2 S_1 S_0$

$$Y = \bar{S}_2 \bar{S}_1 \bar{S}_0 I_0 + \bar{S}_2 \bar{S}_1 S_0 I_1 + \bar{S}_2 S_1 \bar{S}_0 I_2 + \bar{S}_2 S_1 S_0 I_3 \\ S_2 \bar{S}_1 \bar{S}_0 I_4 + S_2 \bar{S}_1 S_0 I_5 + S_2 S_1 \bar{S}_0 I_6 + S_2 S_1 S_0 I_7$$

→ Draw 8:1 mux circuit

## Realize 4:1 using only 2:1 MUXs (Multiplexer tree)

For 4:1 mux

$$y = \bar{s}_1 \bar{s}_0 I_0 + \bar{s}_1 s_0 I_1 + s_1 \bar{s}_0 I_2 + s_1 s_0 I_3$$

For 2:1 mux

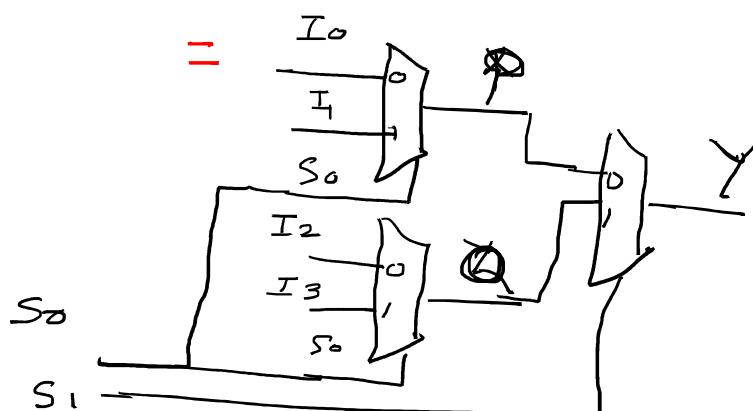
$$y = \bar{s}_0 I_0 + s_0 I_1$$

$$\rightarrow y = \bar{s}_1 (\bar{s}_0 I_0 + s_0 I_1) +$$
  

2:1 mux

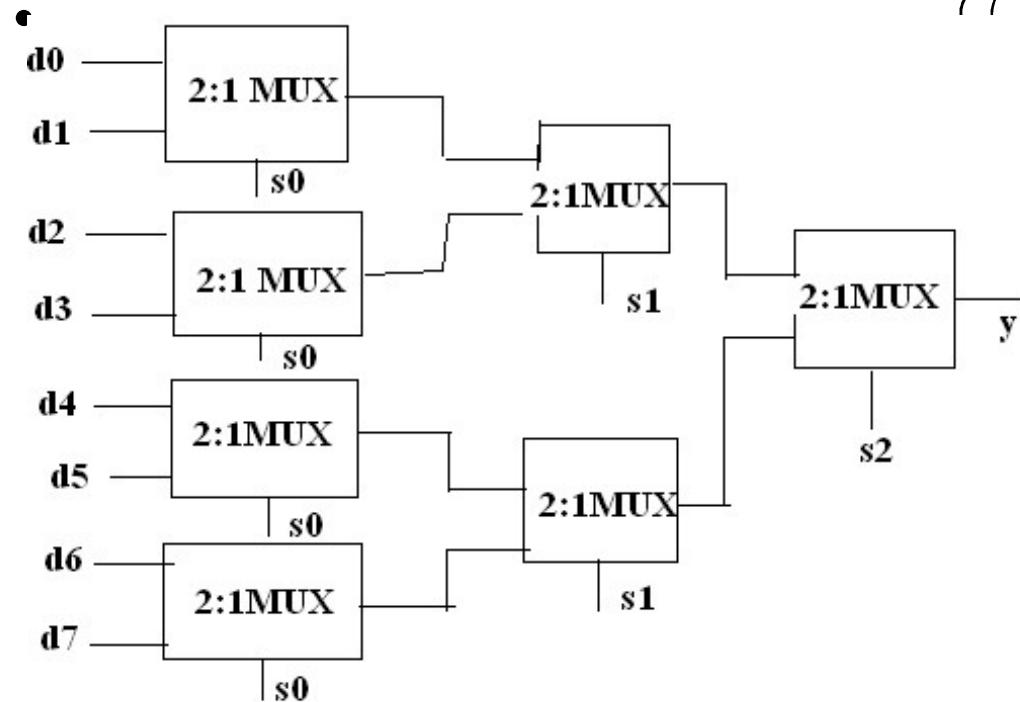
$$s_1 (\bar{s}_0 I_2 + s_0 I_3) = \bar{s}_1 \cdot P + s_1 \cdot Q$$
  

2:1 mux



3, 2:1 muxs

## Realize 8:1 using only 2:1 MUXs (Multiplexer tree)



( $4 + 2 + 1$ ), 2:1 MUXs

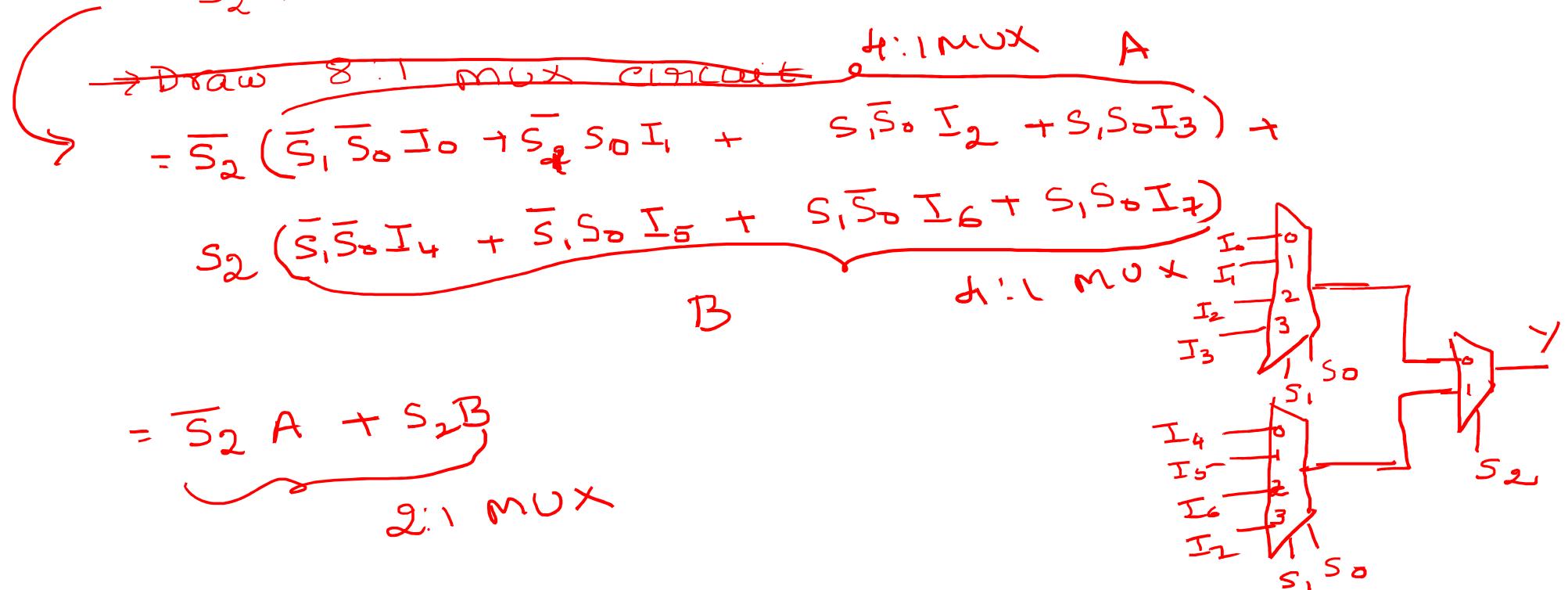
$2^n$ :1 mux using  
2:1 mux  
how many 2:1 mux?

$2^n - 1$  ← Ans?

## Realize 8:1 using only 4:1 MUXs and 2:1 MUX

- Select lines:  $S_2 S_1 S_0$

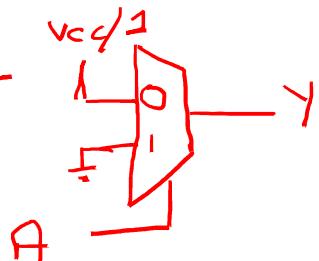
$$Y = \bar{S}_2 \bar{S}_1 \bar{S}_0 I_0 + \bar{S}_2 \bar{S}_1 S_0 I_1 + \bar{S}_2 S_1 \bar{S}_0 I_2 + \bar{S}_2 S_1 S_0 I_3 + \\ S_2 \bar{S}_1 \bar{S}_0 I_4 + S_2 \bar{S}_1 S_0 I_5 + S_2 S_1 \bar{S}_0 I_6 + S_2 S_1 S_0 I_7$$



## Realize each of the basic logic gates using 2:1 MUX

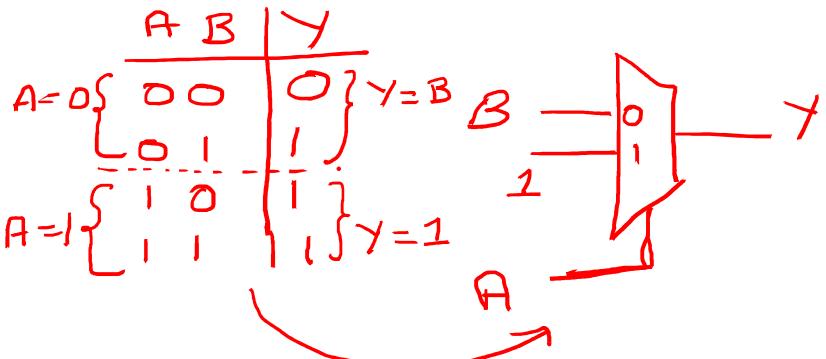
$$\text{NOT} \Rightarrow Y = \bar{A}$$

A	Y
0	1
1	0



$$\text{OR gate } Y = A + B$$

if  $A=0$   $Y=B$



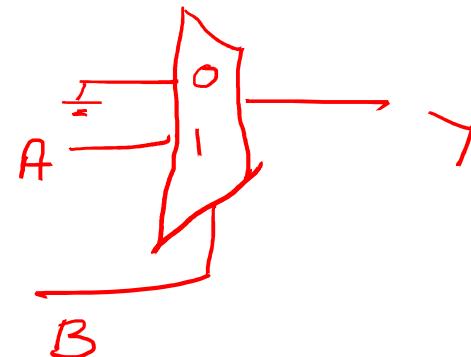
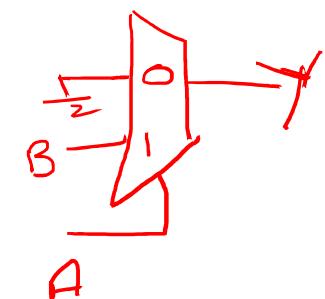
$$\text{AND}$$

$$Y = AB$$

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

$\{ \begin{matrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{matrix} \} \quad Y=0$

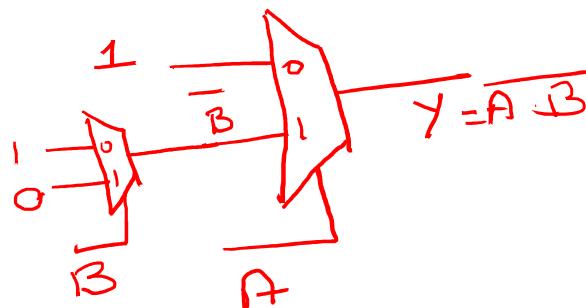
$\{ \begin{matrix} 1 & 1 \end{matrix} \} \quad Y=B$



NAND gate using 2:1 MUXs

$$Y = \overline{A \cdot B}$$

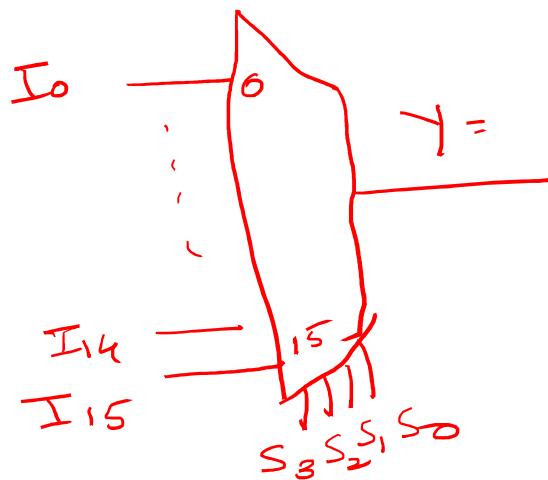
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0



Realize NOR, XOR

## 16-to-1 line multiplexer----Exercise for you

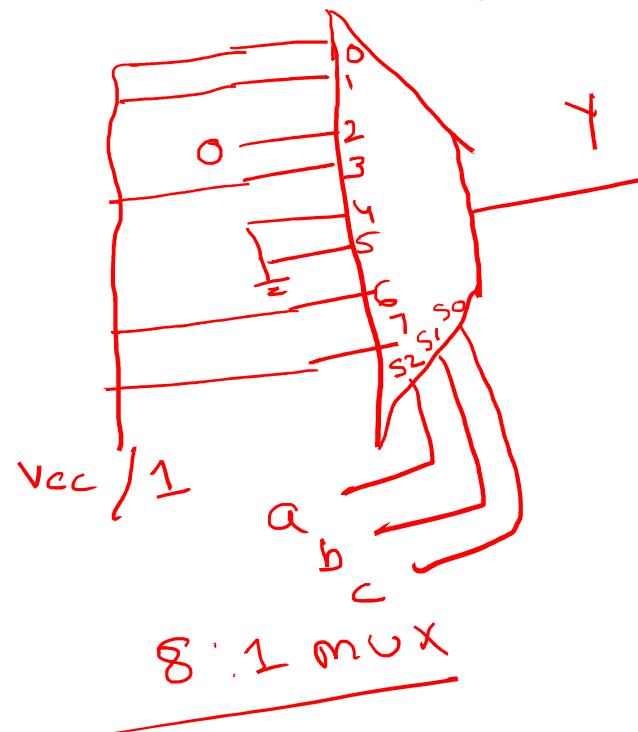
- Write the Symbol/block diagram, function table, output expressions and circuit



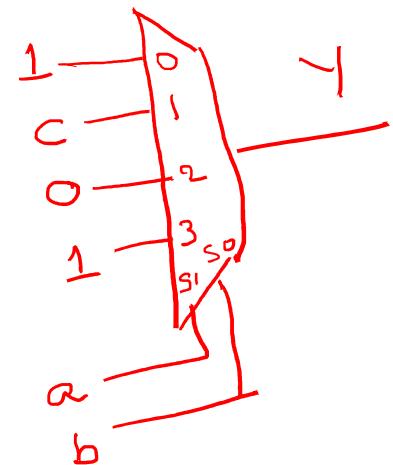
# Multiplexer application in logic design

- Implement the function  $F(a,b,c) = \sum m(0,1,3,6,7)$  using
  - 8:1 MUX only ✓ *L equal to the no. of select lines*
  - Minimum no. of 4:1 MUXs only ~~except 4:1~~

$a \ b \ c$	$F$
$ab=00$	0 0 0        } $F=1$
	0 0 1      -   } $F=1$
$ab=01$	0 1 0      0 } $F=c$
	0 1 1      -   } $F=1$
$ab=10$	1 0 0      0 } $F=0$
	1 0 1      -   } $F=0$
$ab=11$	1 1 0      1 } $F=1$
	1 1 1      1 } $F=1$



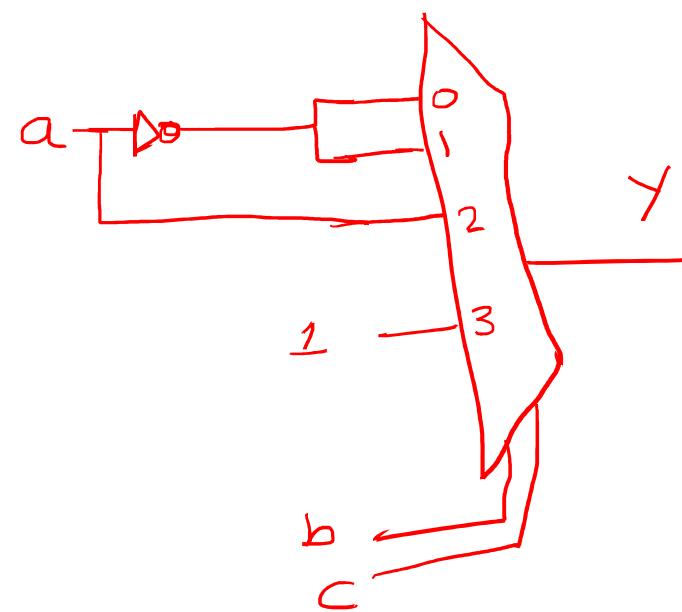
4:1 MUX



Implement the function  $F(a,b,c) = \sum m(0,1,3,6,7)$  using  $bc$  as selection lines for 4:1 mux

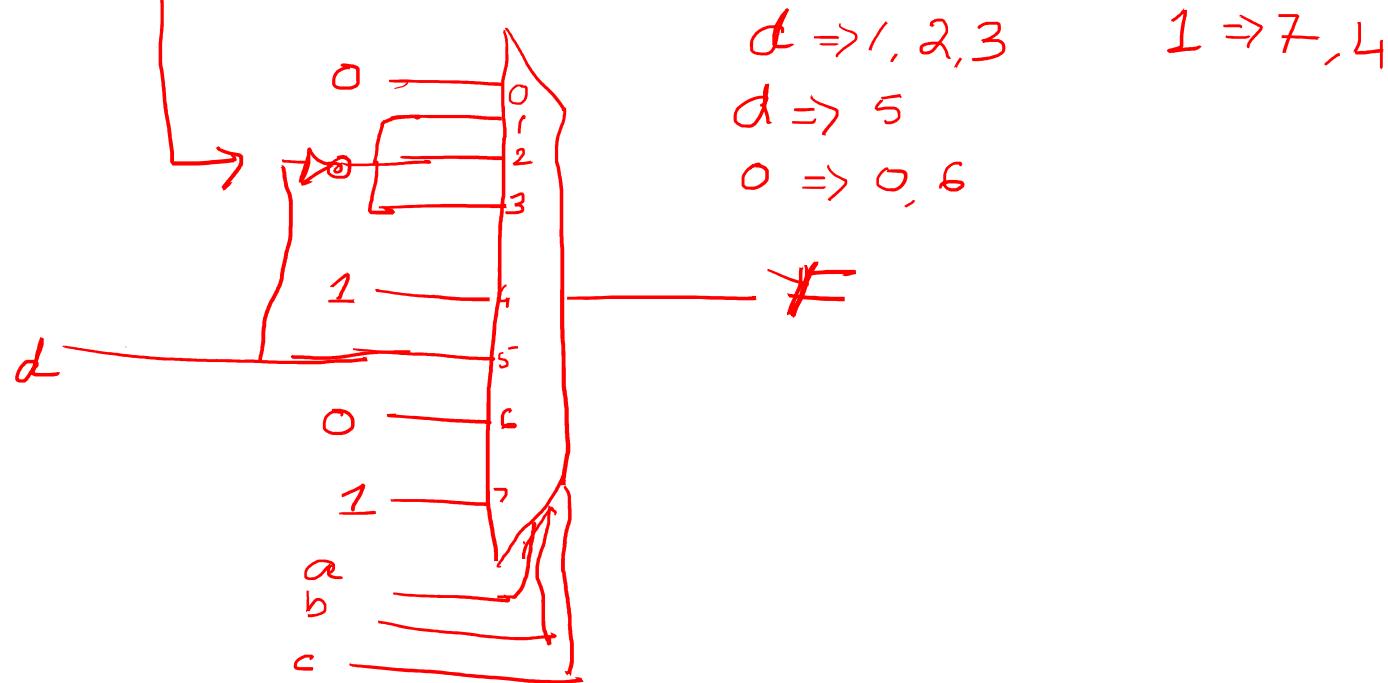
	$\bar{b}c$	$\bar{b}c$	$\bar{b}c$	$bc$
$\bar{a}_0$	0 1	1 1	2 0	3 1
$a=1$	4 0	5 0	6 1	7 1
	$b_c = 00$	01	10	11
$y = \bar{a}$	$\bar{a}$	$a$	1	

$\uparrow$   
 $T\bar{T}$  is newritten



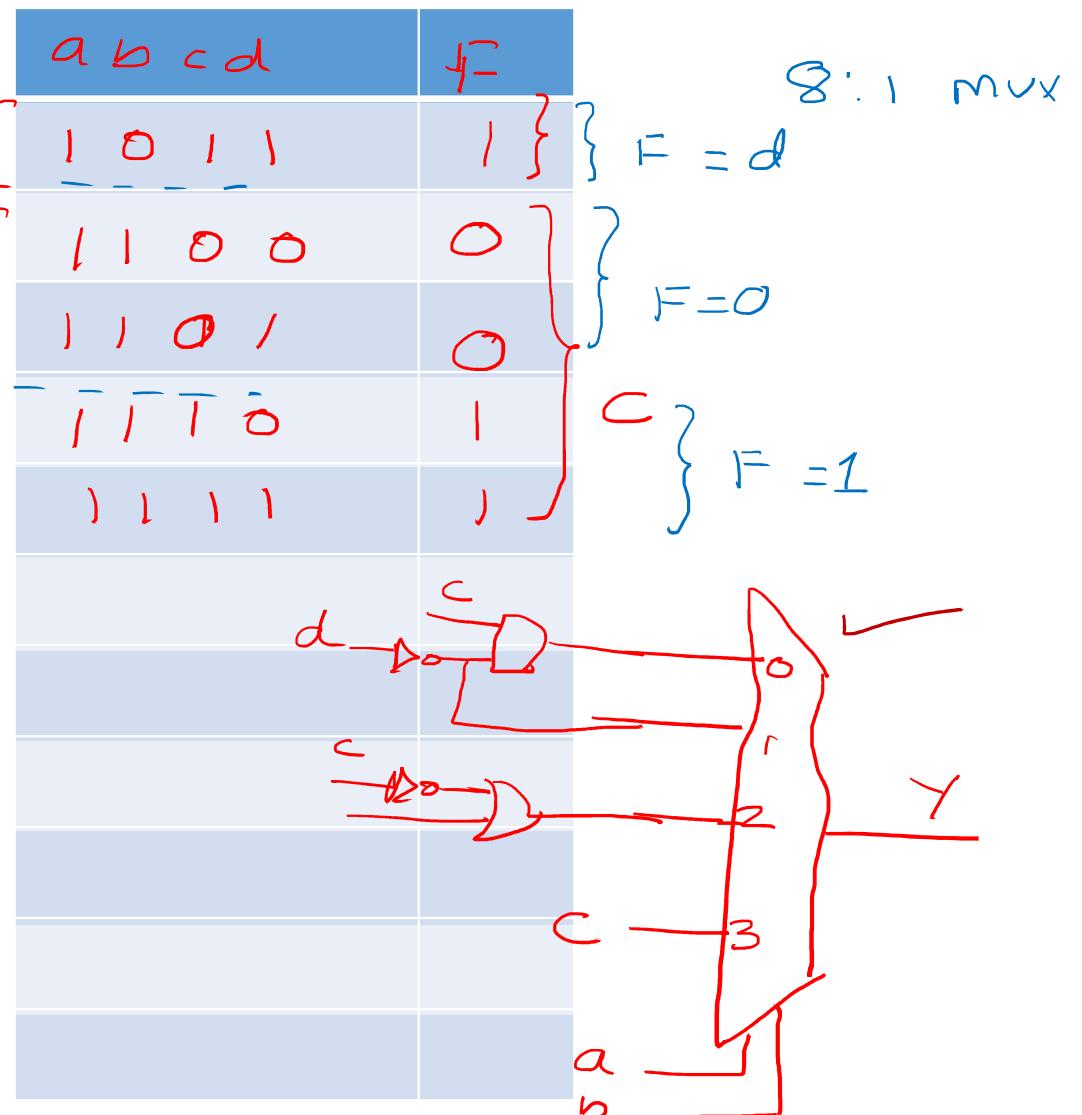
## Implement the function $F(a,b,c,d) = \sum m(2,4,6,8,9,11,14,15)$

- using
  - ✓ 8:1 MUX and one NOT gate  $\sqrt{a,b,c}$  as the selection inputs
  - . 4:1 MUX and external gates
  - . 4:1 MUX and 2:1MUXs only



$$F(a,b,c,d) = \Sigma m(2,4,6,8,9,11,14,15)$$

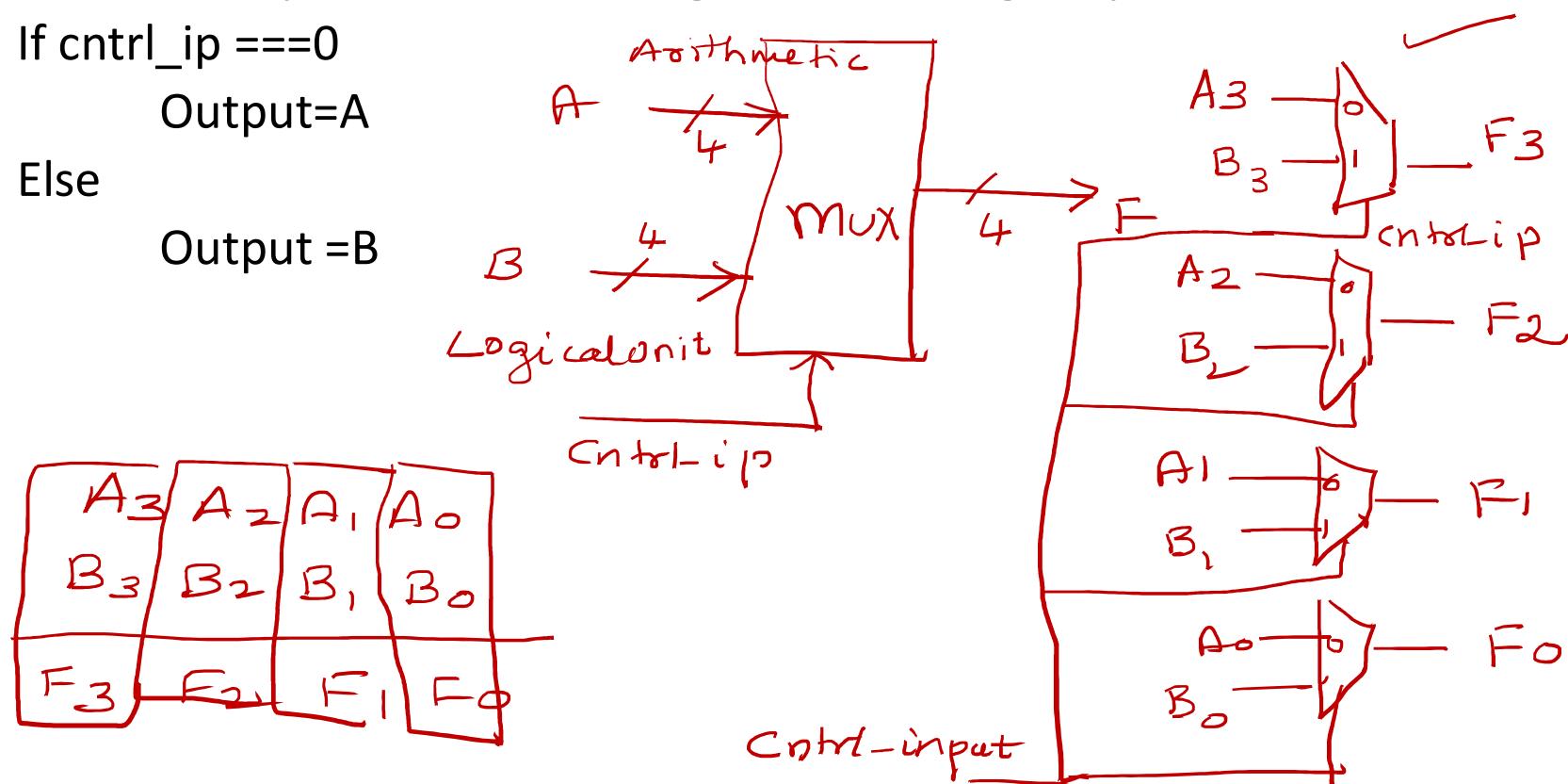
$a$	$b$	$c$	$d$	$F$
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

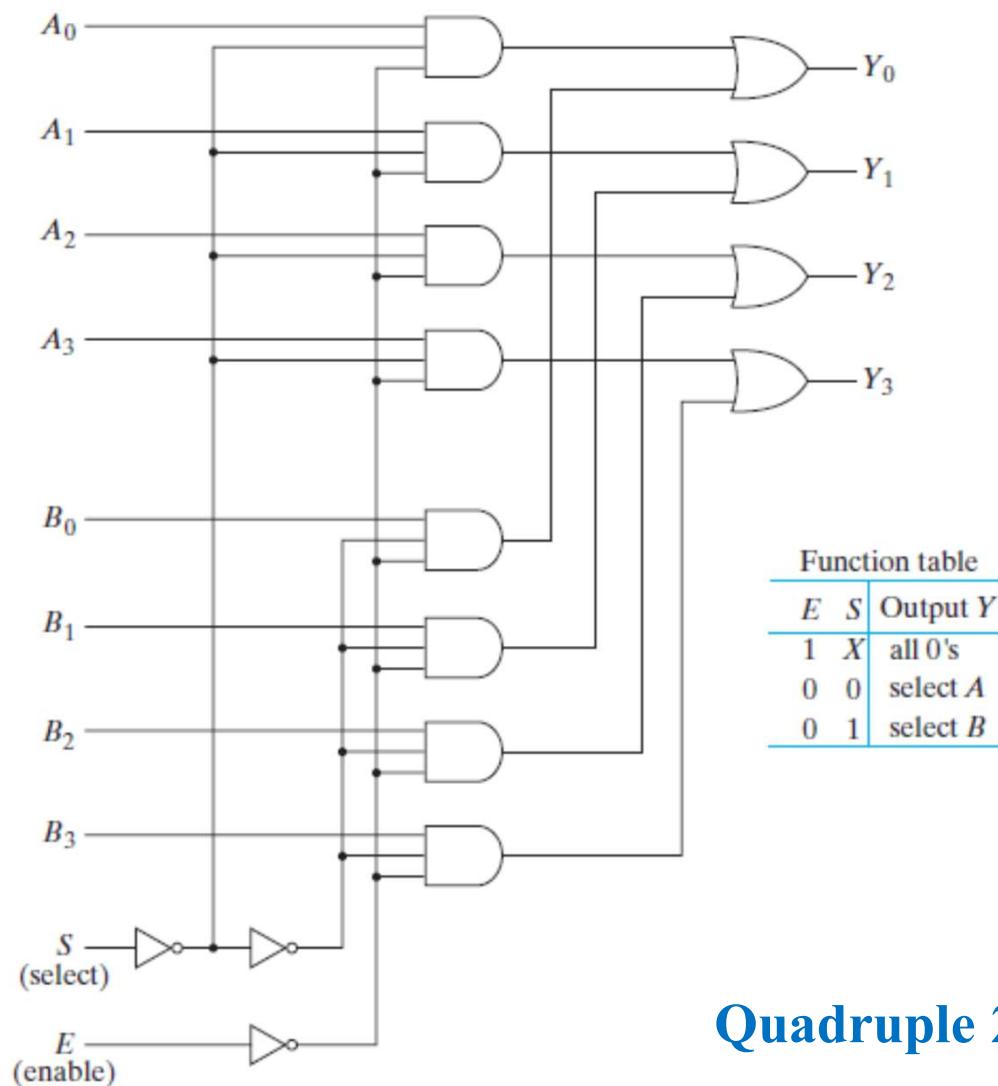


**F(a,b,c,d) = Σm(2,4,6,8, 9,11,14,15) realization contd...**

## Nibble Multiplexer

- A and B are two 4-bit numbers. Design a combinational circuit using suitable multiplexers according to following requirements.
  - If  $\text{ctrl\_ip} == 0$   
Output = A
  - Else  
Output = B

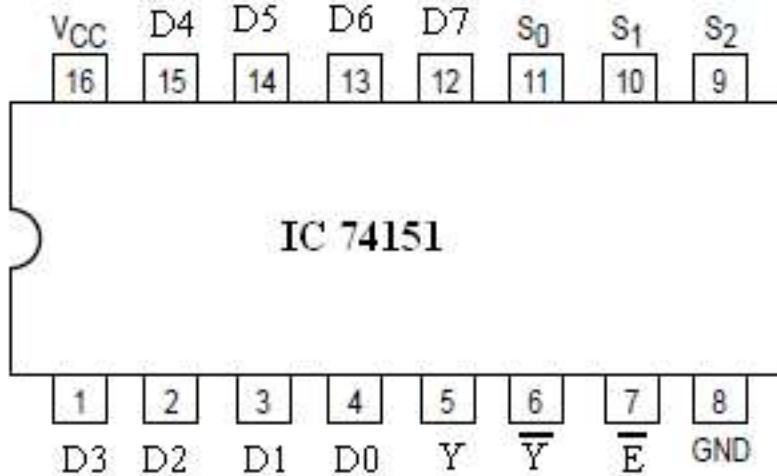




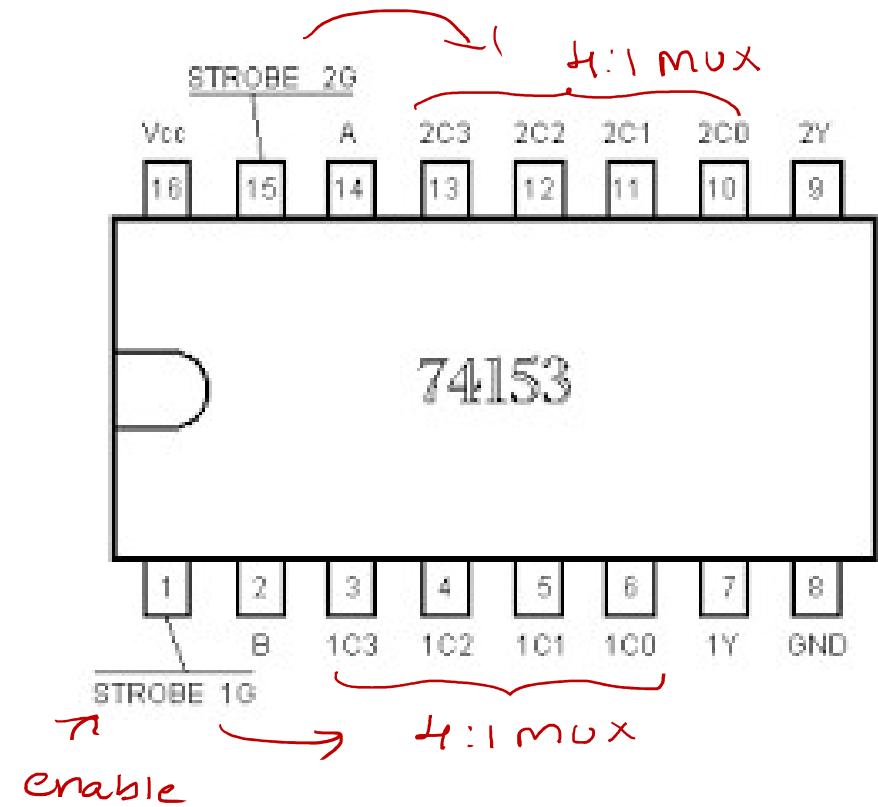
**Quadruple 2:1 MUX or Nibble MUX**

## Multiplexer ICs :

74151- 8:1 MUX

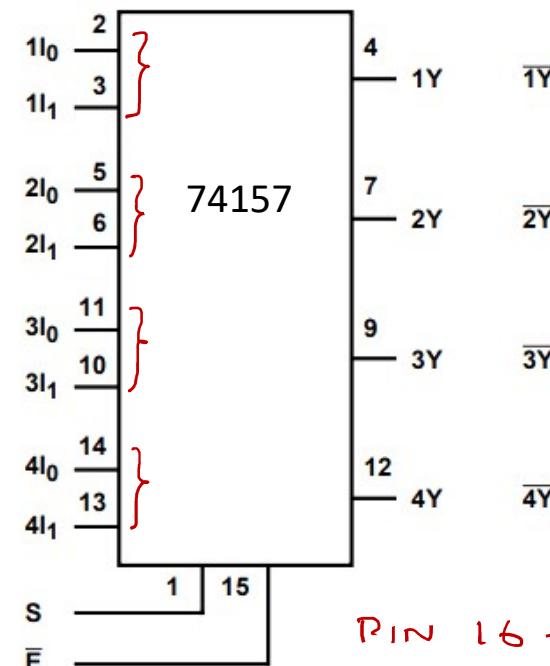
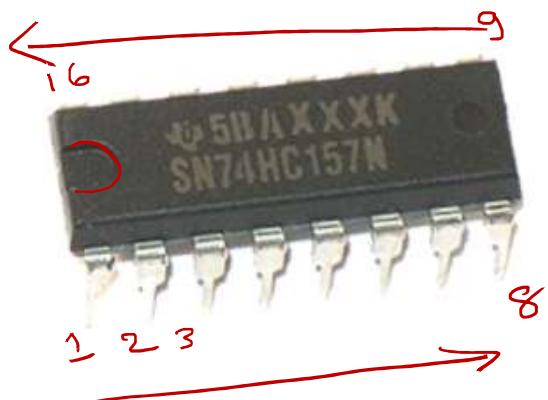


74153- 4:1 MUXs



# 74157 Quad 2:1 MUX IC

4,



PIN 16 -  $V_{CC}$   
8 - Gnd

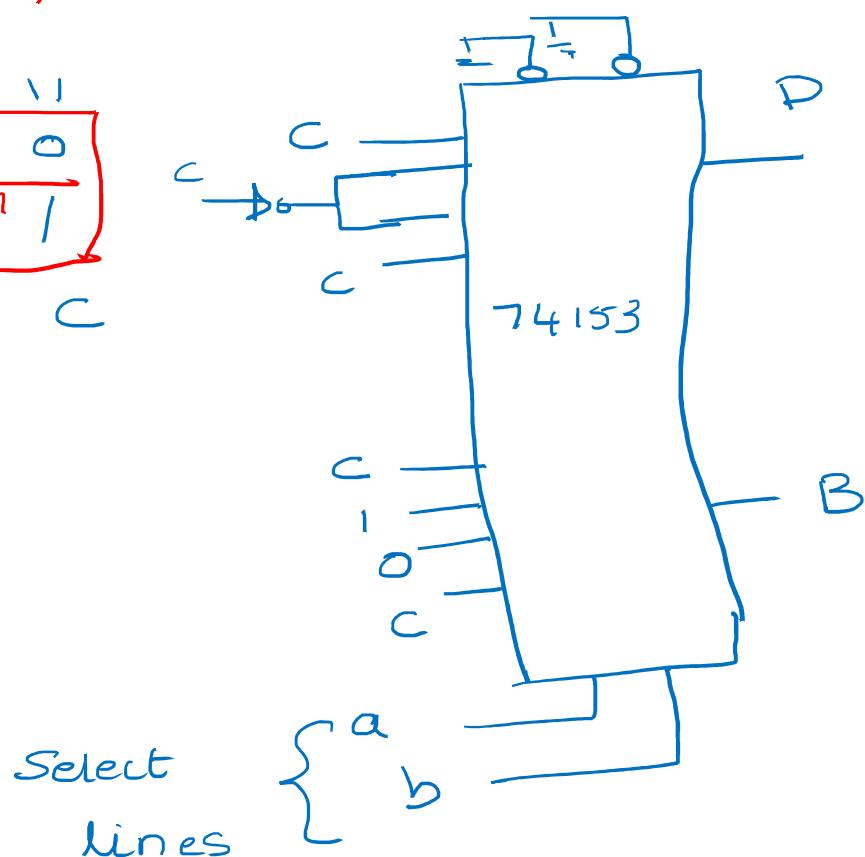
# Design a full subtractor using 74153 IC and one external gate.

(a, b, c)

$$D = \Sigma m(1, 2, 4, 7) \quad B = \Sigma m(1, 2, 3, 7)$$

$a$	$b$	$c=0$	$c=1$	$D$
0	0	0	1	0
1	0	1	1	1
0	1	0	1	0
1	1	1	0	1

$a$	$b$	$c=0$	$c=1$	$B$
0	0	0	1	0
1	0	1	1	1
0	1	0	1	0
1	1	1	0	1



**Full subtractor using 74153 IC and one external gate  
contd...**

## Design a full adder/ subtractor using 74153 IC and two external gates

if  $m=0$       adder  $F = a + b + C_{in} = s$

else

subtractor  $F = a - b$   
 ~~$= a + \bar{b}$~~

$$FA \Rightarrow S = \sum m(1, 2, 4, 7)$$

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

$$FS \Rightarrow D = \sum m(1, 2, 4, 7)$$

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

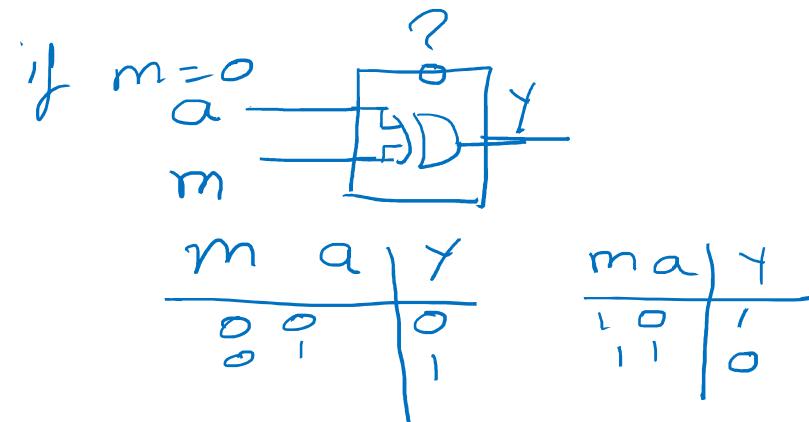
## Full adder/subtractor using 74153 IC and two external gates contd...

choose  $b_c$  as selection

$b_c$	00	01	10	11
$a=0$	0	1	1	0
$a=1$	1	0	0	1
Sum/Diff =	$a$	$\bar{a}$	$\bar{a}$	$a$

$C/P$

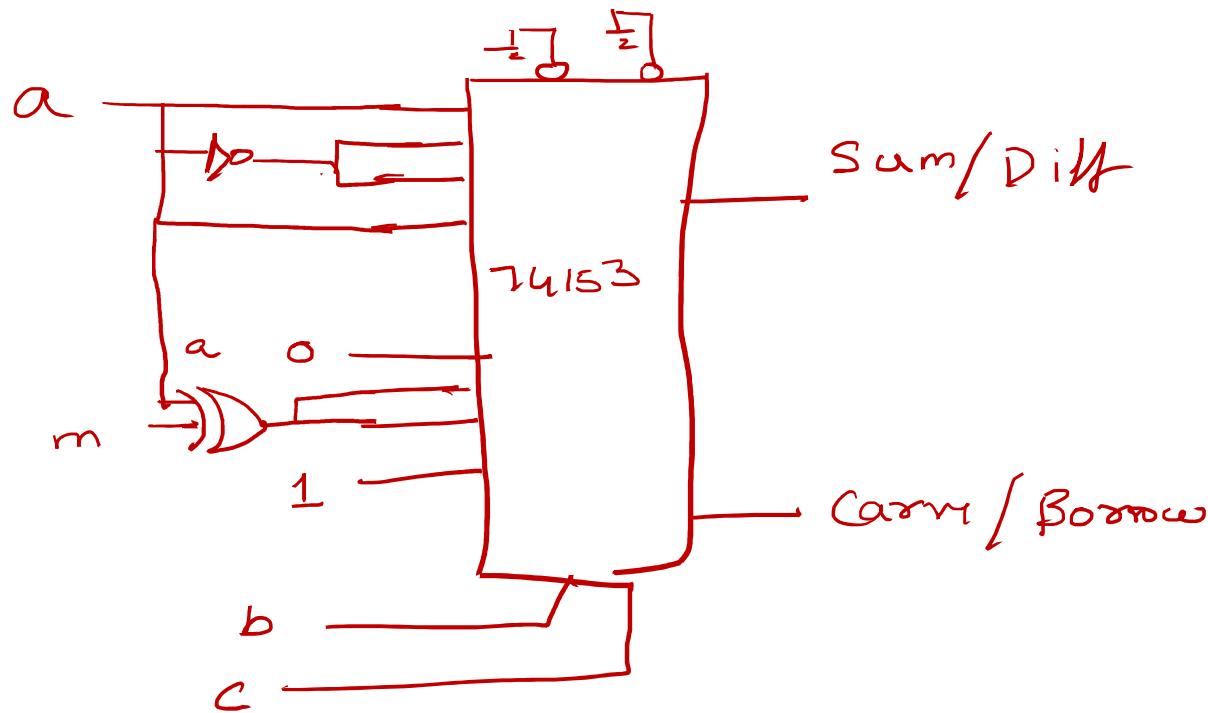
$b_c$	00	01	10	11
$a=0$	0	0	0	1
$a=1$	0	1	1	1
Carry =	<u>0</u>	<u>a</u>	<u>a</u>	<u>1</u>



$b_c$

$b_c$	00	01	10	11
$a=0$	0	1	1	1
$a=1$	0	0	0	1
Borrow =	<u>0</u>	<u><math>\bar{a}</math></u>	<u><math>\bar{a}</math></u>	<u>1</u>

## Full adder/subtractor using 74153 IC and two external gates



## **Exercise:**

- Realize the Boolean expression  $f(w,x,y,z) = \Sigma m(4,5,7,8,10,12,15)$  using
  - 74151 multiplexer and external gate
  - 74153 IC and external gates
- Refer the prescribed book for additional problems

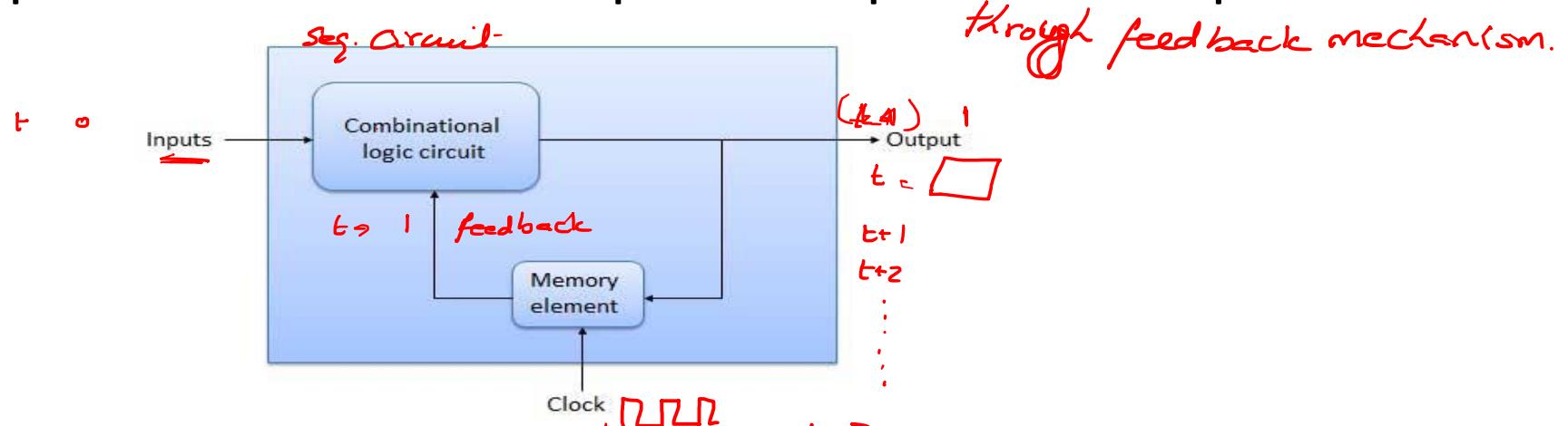
**Question?**

# Sequential circuits

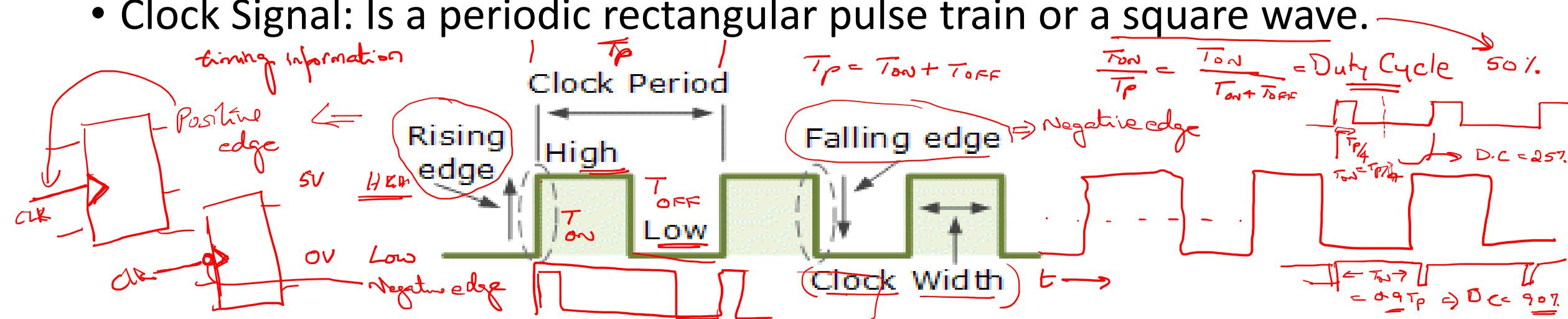
- NAND Latch
- NOR Latch
- SR, D, JK, T flip flop

# Sequential circuits:

- Outputs are dependent on current inputs and previous outputs.



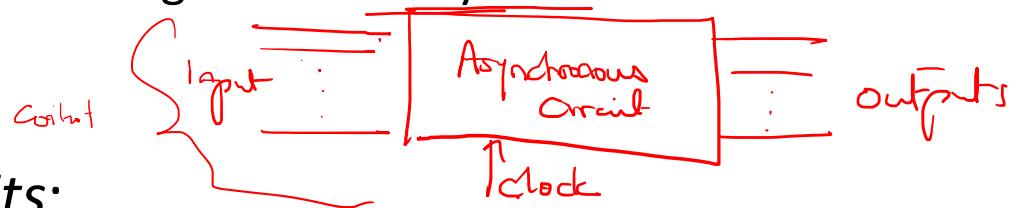
- Storage elements : Devices capable of storing binary information *in terms of bits*
- Clock Signal: Is a periodic rectangular pulse train or a square wave.



- Two types of sequential circuits:

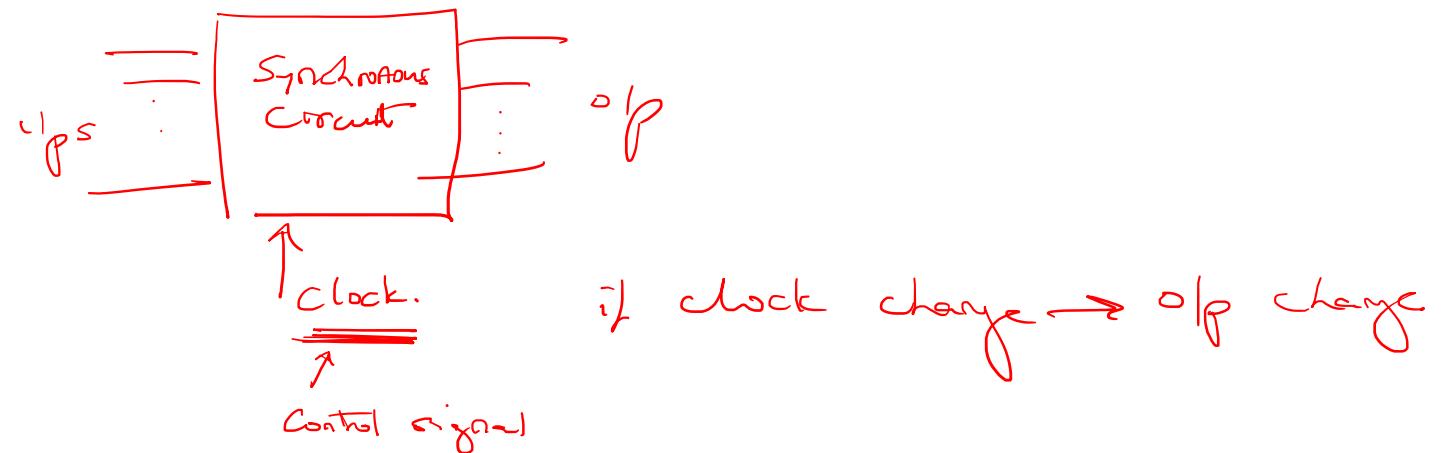
- *Asynchronous sequential circuits:*

The output of the logic circuits can change state at any time when one or more of the inputs change.

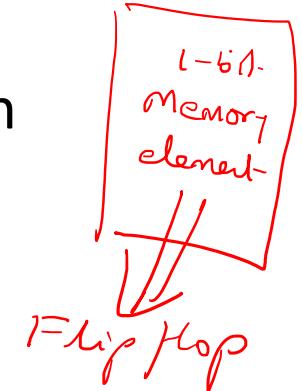


- *Synchronous sequential circuits:*

The exact times at which any output can change states are determined by a signal called the clock.



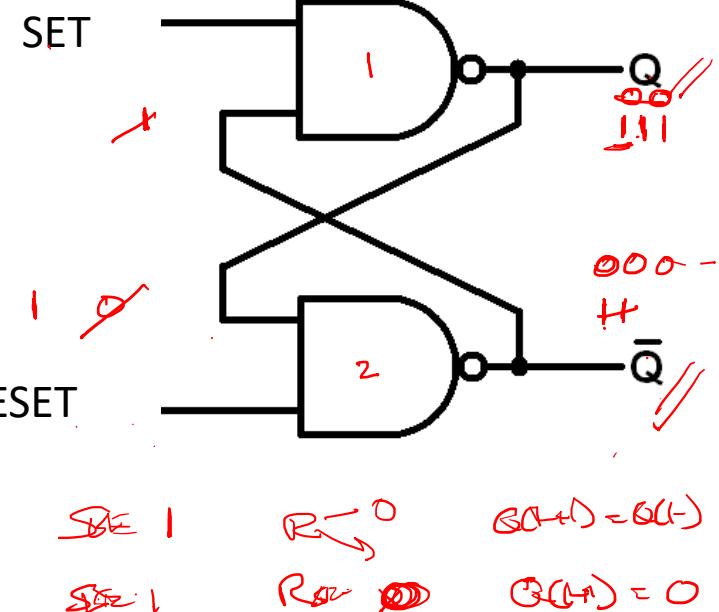
- Flip Flop:
  - Is a binary storage element capable of storing one bit of information
- Latch: <sup>basic</sup> ~~Circuit within flip flop~~
  - Basic type of flip flop is referred as Latch



$$\underbrace{(\text{Latch} + \text{control})}_{\text{ }} = \text{flip flop}$$

In latches : Circuit is built using AND or NOR gates.

# NAND Latch (Active Low latch)



SET	RESET	$Q(t)$	$Q(t+1)$	$Q'(t+1)$
0	0	0	1	1
0	0	1	1	1
0	1	0	1	0
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

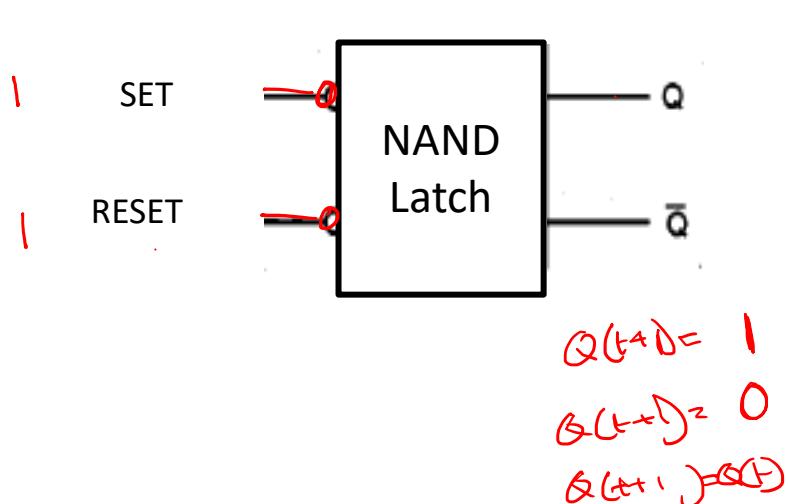
INVALID or Indeterminate state OR

Retained / No change

Step → time count

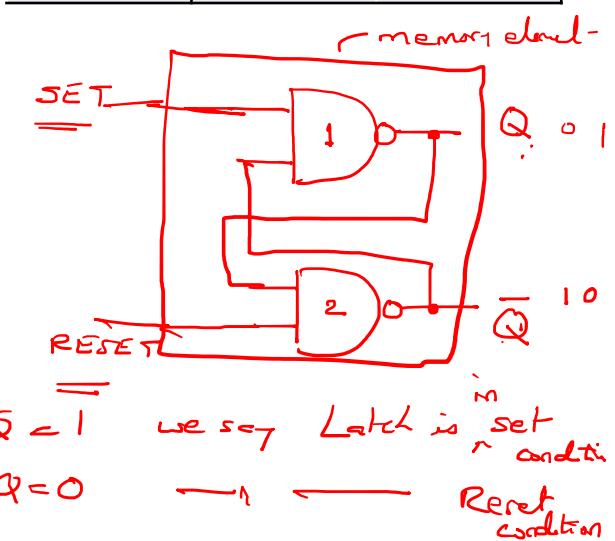
Previous State → Present State

$Q(t-1)$	$Q(t)$	$Q'(t)$
0	1	1
1	1	1
0	1	0
1	1	0
0	0	1
1	0	1
0	0	1
1	1	0

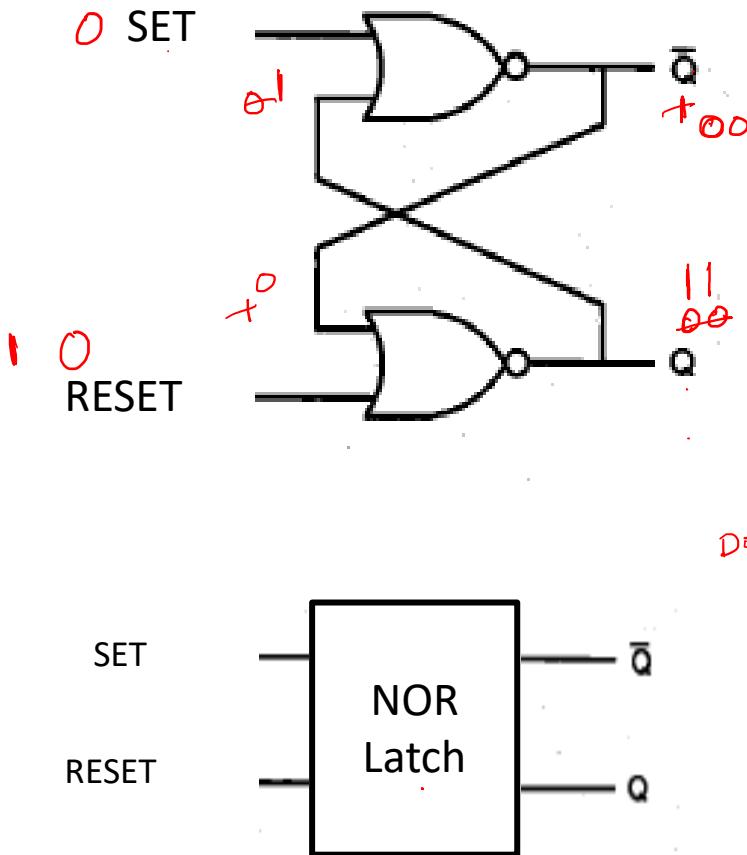


Function table

SET	RESET	Output
0	0	Invalid state Indeterminate state
0	1	Set $Q=1$
1	0	Reset $Q=0$
1	1	No Change



# NOR Latch (Active high Latch)



SET	RESET	$Q(t)$	$Q(t+1)$	$Q'(t+1)$
0	0	0	0	1 previous output
0	0	1	1	0
0	1	0	0	Reset
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	0	0
1	1	1	0	0

*Note: Handwritten annotations include 'Set True' and 'Reset' circled in red.*

Function table				
SET	RESET	Output		
0	0	No Change in state		
0	1	Reset		
1	0	Set		
1	1	Indeterminate		

$Q(t-1)$	$Q(t)$	$Q'(t)$
0	0	1
1	1	0
0	0	1
1	0	1
0	1	0
1	1	0
0	0	0
1	0	0

*Same table*

$$Q(t+1) = Q(t)$$

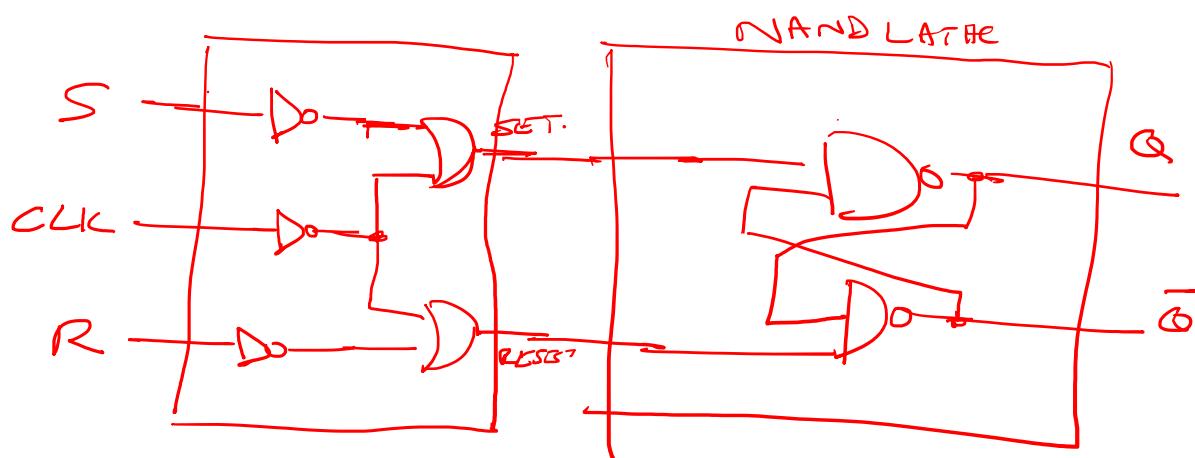
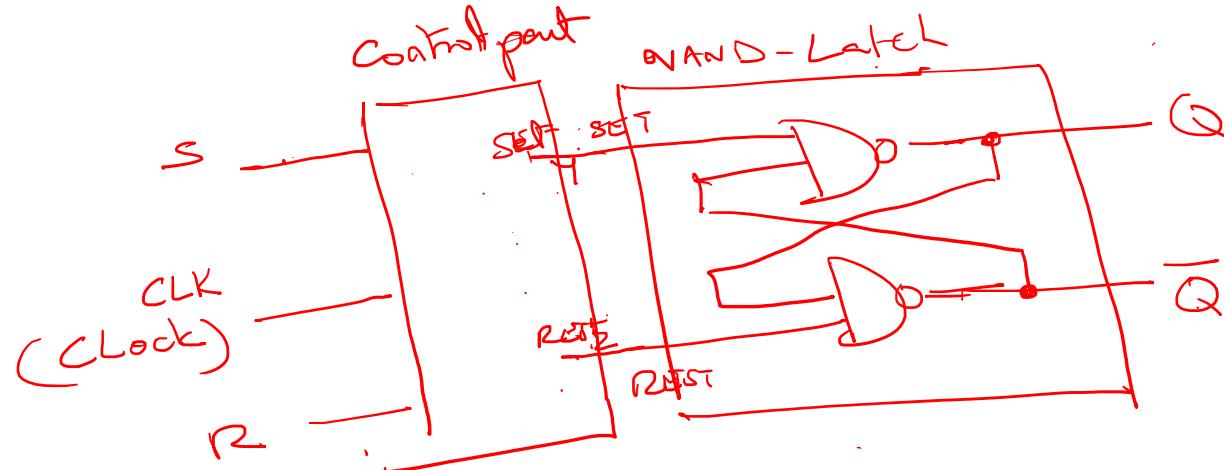
$$\rightarrow Q(t+1) = 0$$

$$Q(t+1) = 1$$

*Invalid state  $Q(t+1) = X$  Don't care*

# SR Flip Flop using NAND latch

Flip-Flop is a 1-bit storing circuit employs a basic latch and a control input-ckt.

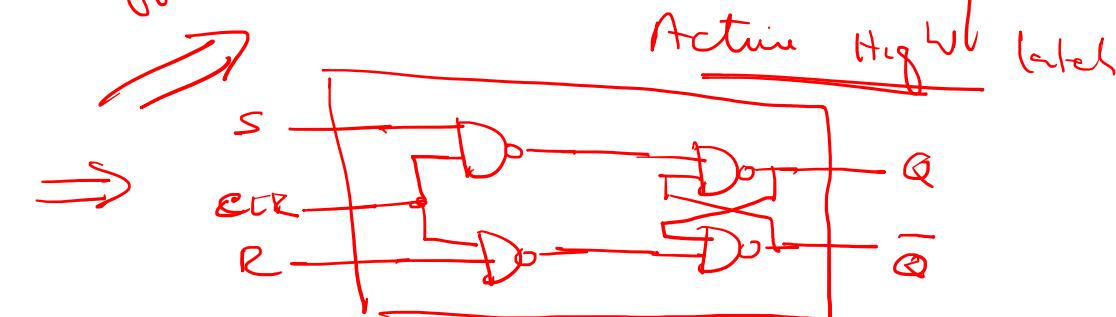


Active High Flip-flop

Requirements for SR flip flop

Clk	S	R	Output
0	x	x	No change
1	0	0	No change
1	0	1	Reset
1	1	0	Set
1	1	1	Indeterminate

level trigger



# SR Flip Flop using NAND latch

Function Table for SR Flip Flop			Output
Clk	S	R	
0	X	X	No Change ✓
1	0	0	No Change
1	0	1	Reset
1	1	0	Set
1	1	1	Indeterminate

Clk	S	R	Q(t)	Q(t+1)	SET	RESET
0	0	0	0	0	1	X
0	0	0	1	1	X	1
0	0	1	0	0	1	X
0	0	1	1	1	X	1
0	1	0	0	0	1	X
0	1	0	1	1	X	1
0	1	1	0	0	1	X
0	1	1	1	1	X	1
1	0	0	0	0	1	X
1	0	0	1	1	X	1
1	0	1	0	0	1	X
1	0	1	1	1	0	0
1	1	0	0	1	0	1
1	1	0	1	0	1	0
1	1	1	0	X	X	X
1	1	1	1	X	X	X

Equation for SET and RESET

$SET = \overline{R} \cdot \overline{S} = \overline{CLR} \cdot S$

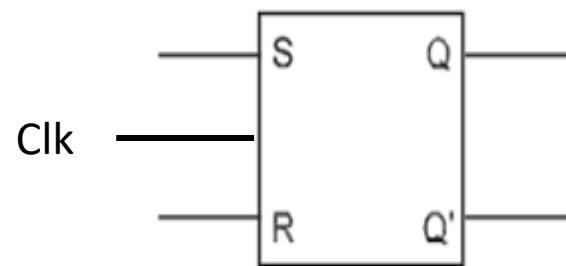
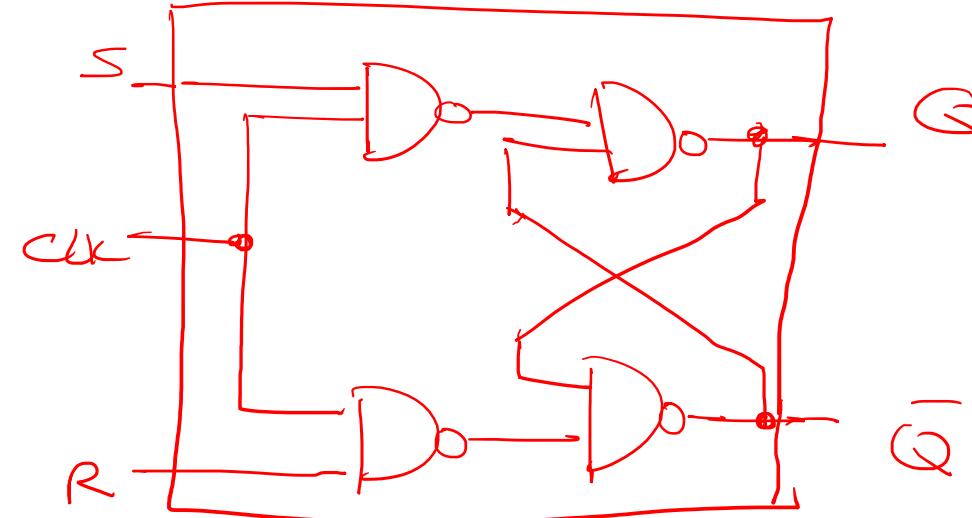
$RESET = \overline{C} \cdot \overline{R} = \overline{CLR} \cdot R$

$\overline{C} \cdot \overline{L} + \overline{R} \cdot \overline{S} = \overline{SET}$

$\overline{C} \cdot \overline{L} + \overline{C} \cdot \overline{R} = \overline{RESET}$

# Circuit:

Clk	S	R	$Q(t+)$	$\bar{Q}(t+)$
0	X	X	No change	
1	0	0	No change $Q(t)$ $\bar{Q}(t)$	
1	0	1	0	1
1	1	0	1	0
1	1	1	Indeterminate	



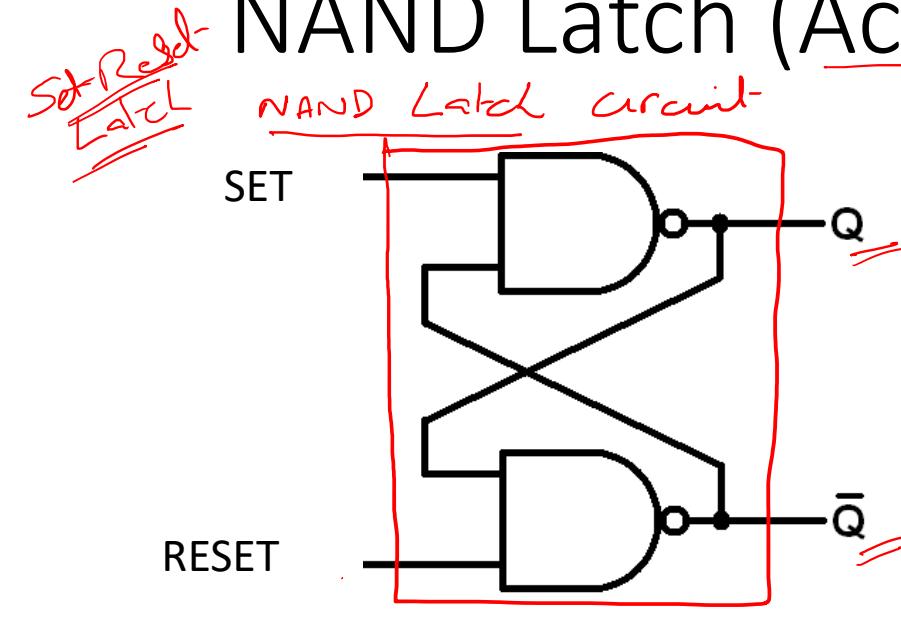
Active high SR flip flop

# Sequential circuits

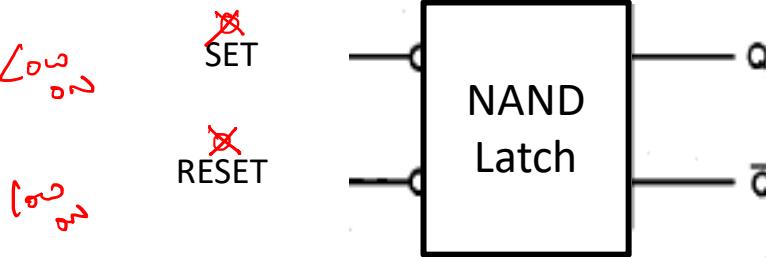
Continued

- NAND Latch
- NOR Latch
- SR, D, JK, T flip flop

# NAND Latch (Active Low latch)



*functionalities of NAND Latch*



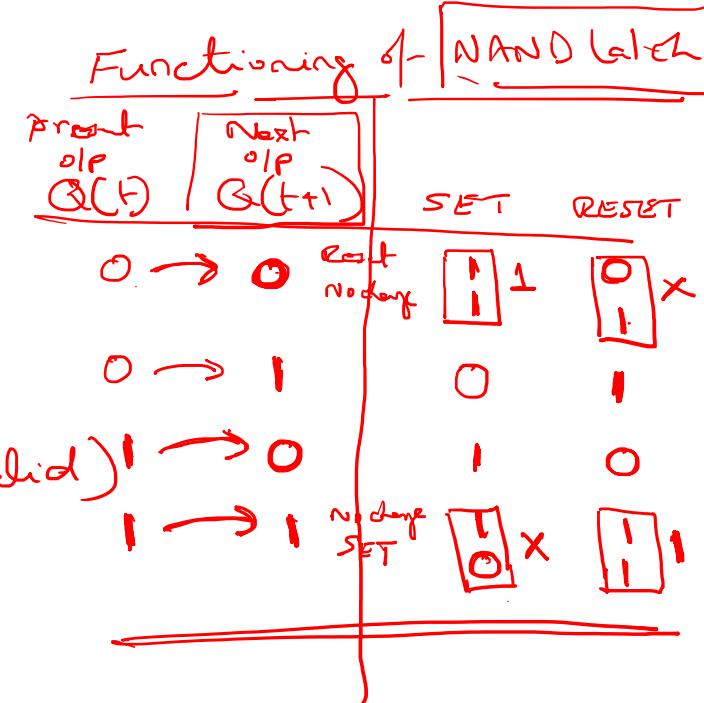
SET	RESET	Q(t)	Q(t+1)	$Q'(t+1)$
0	0	0	1	1
0	0	1	1	1
0	1	0	1	0
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

*not desired*  
*o/p*

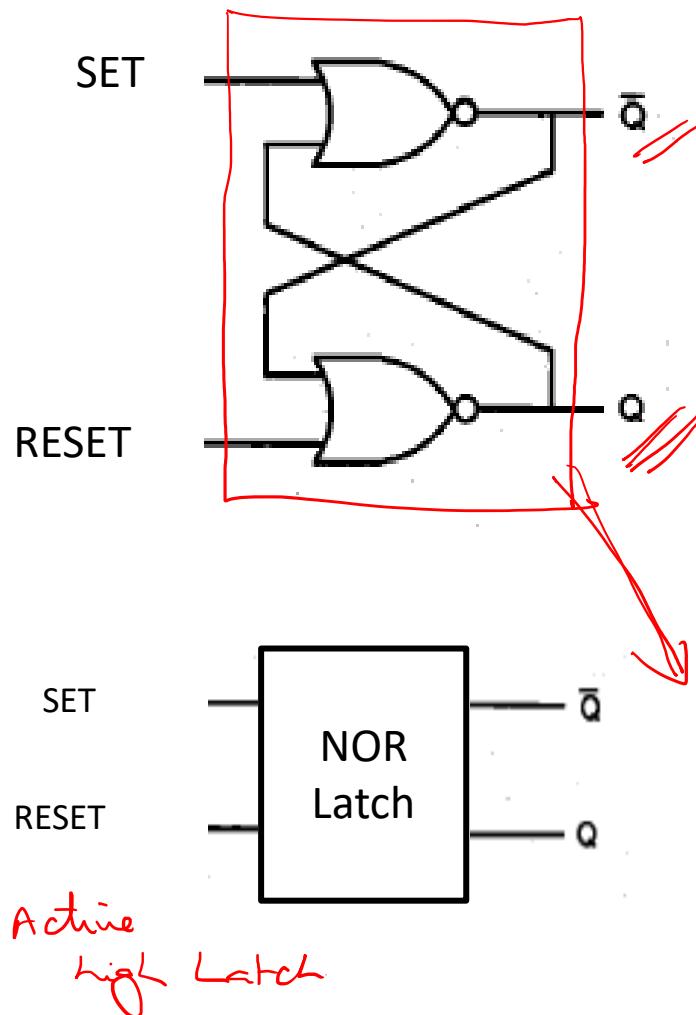
*Function table*

SET	RESET	Output	$Q(t+1)$
0	0	Indeterminate (Invalid)	
0	1	Set	1
1	0	Reset	0
1	1	No Change	$Q(t)$

*Not to use at o/p  
& NAND latch*



# NOR Latch (Active high Latch)



SET	RESET	$Q(t)$	$Q(t+1)$	$Q'(t+1)$
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	0	0
1	1	1	0	0

Function table

SET	RESET	Output	$Q(t+1)$
0	0	No Change	$Q(t)$
0	1	Reset	0
1	0	Set	1
1	1	Indeterminate	X

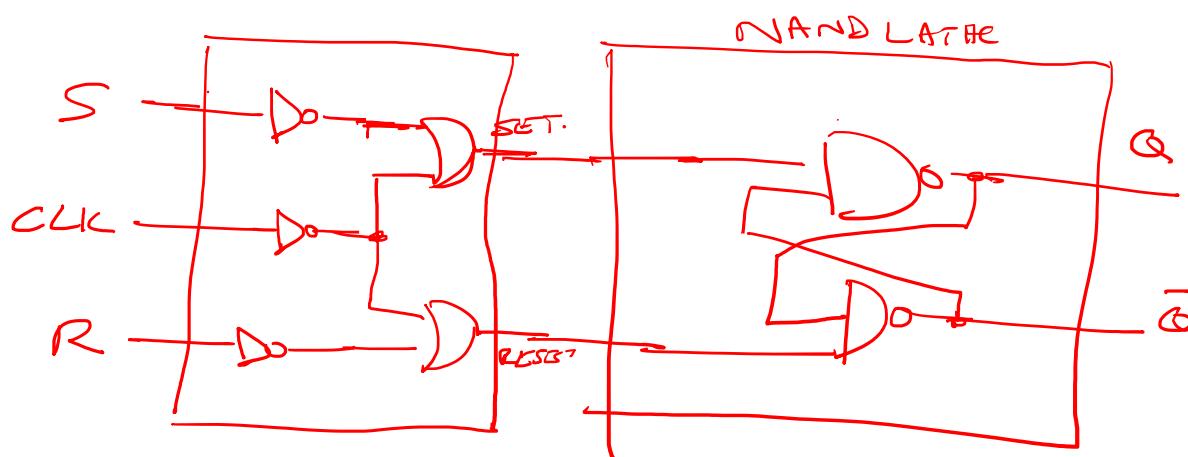
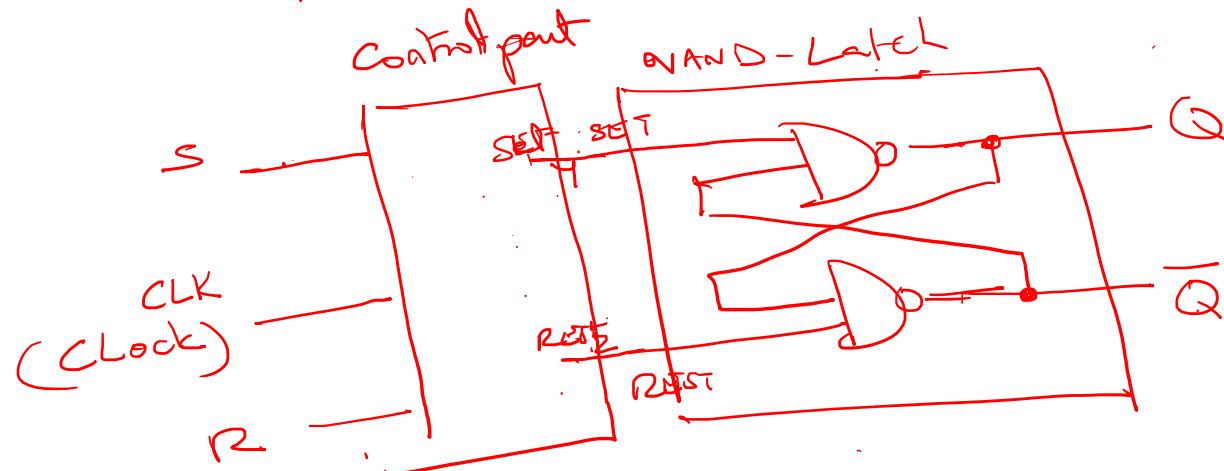
*Don't use '1's on NOR Latch if '1's*

*functionality*

$Q(t)$	$Q(t+1)$	SET	RESET
0	0	retain	0
0	1	Transition	1
1	0	Transition	0
1	1	Retain	1
		Set	X
		Reset	0

# SR Flip Flop using NAND latch

Flip-Flop is a 1-bit storing circuit employs a basic latch and a control input-ckt.

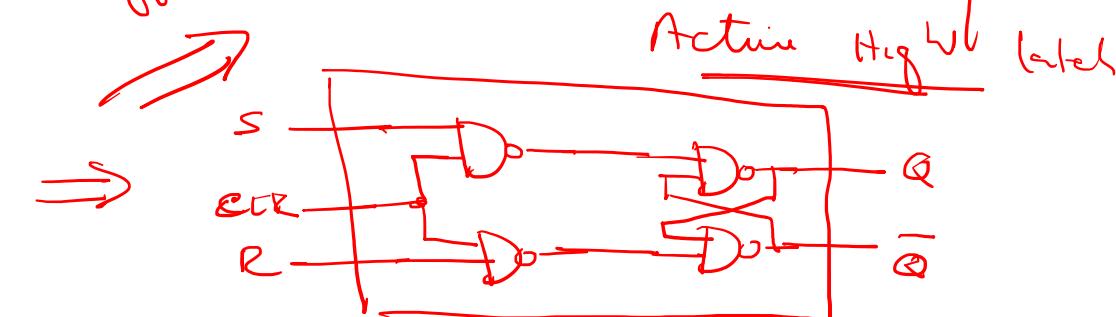


Active High Flip-flop

Requirements for SR flip flop

Clk	S	R	Output
0	x	x	No change
1	0	0	No change
1	0	1	Reset
1	1	0	Set
1	1	1	Indeterminate

level trigger



# SR Flip Flop using NAND latch

Requirement of SR

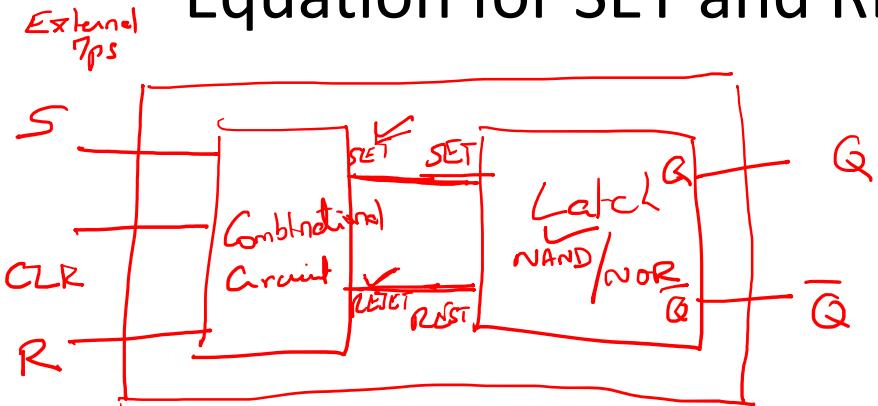
Function Table

of - SR Flip Flop

Clk	S	R	Output
0	X	X	No Change
1	0	0	No Change
1	0	1	Reset $Q(t) = 0$
1	1	0	Set $Q(t) = 1$
1	1	1	Indeterminate

level  
low  
high

Equation for SET and RESET



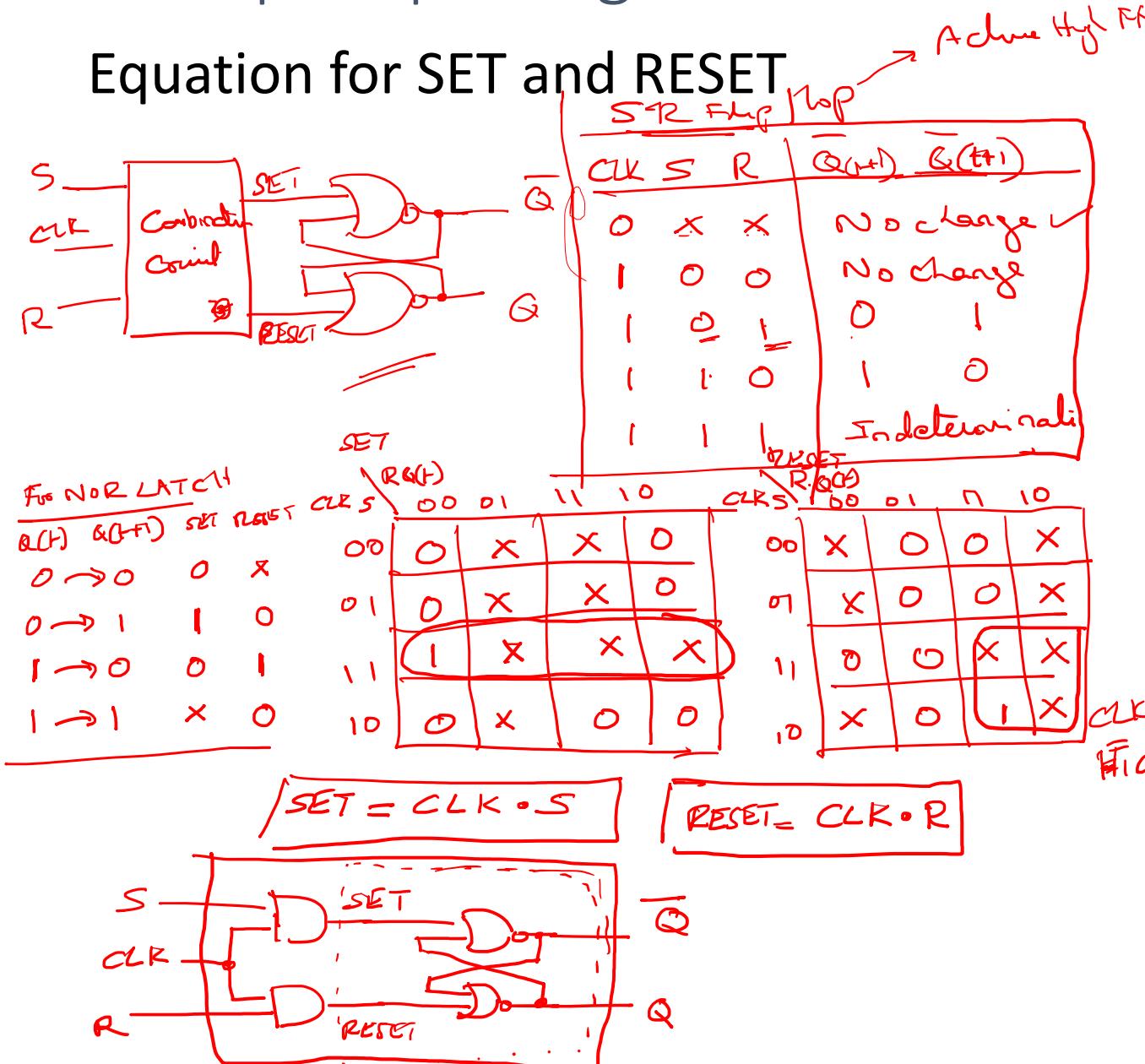
Need to clock any fb from off is required?

Excitation Table

Clk	S	R	Q(t)	Q(t+1)	SET	RESET
0	0	0	0	0	1	X
0	0	0	1	1	X	1
0	0	1	0	0	1	X
0	0	1	1	1	X	1
0	1	0	0	0	1	X
0	1	0	1	1	X	1
0	1	1	0	0	1	X
0	1	1	1	1	X	1
1	0	0	0	0	1	X
1	0	0	1	1	X	1
1	0	1	0	0	1	X
1	0	1	1	1	X	1
1	1	0	0	1	0	1
1	1	0	1	1	X	1
1	1	1	0	X	X	X
1	1	1	1	X	X	X

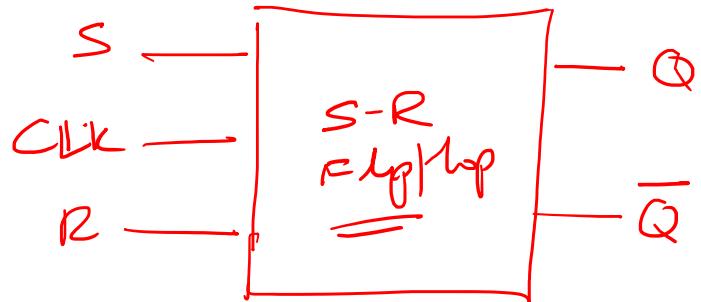
# SR Flip Flop using NOR latch

Equation for SET and RESET



Excitation Table						
Clk	<u>S</u>	<u>R</u>	<u>Q(t)</u>	<u>Q(t+1)</u>	SET	RESET
0	0	0	0	0	0	X
0	0	0	1	1	X	0
0	0	1	0	0	0	X
0	0	1	1	1	X	0
0	1	0	0	0	0	X
0	1	0	1	1	X	0
0	1	1	0	0	0	X
0	1	1	1	1	X	0
1	0	0	0	0	0	X
1	0	0	1	1	X	0
1	0	1	0	0	0	X
1	0	1	1	1	X	0
1	1	0	0	0	0	1
1	1	0	1	1	1	0
1	1	1	0	X	X	X
1	1	1	1	X	X	X

# Circuit and Block Diagram



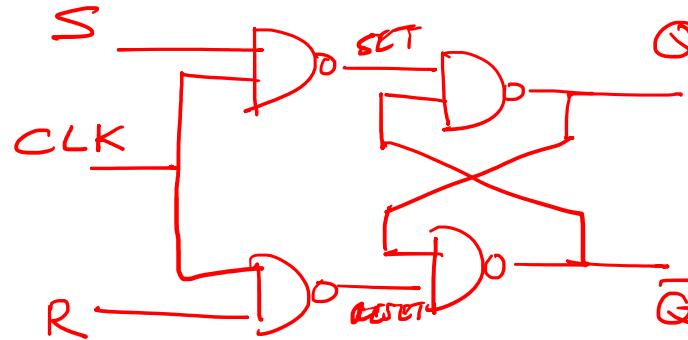
Block diagram

of SR flip flop

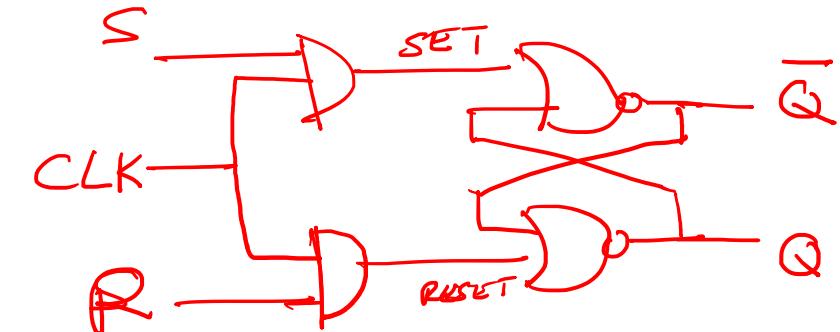
Function table

CLK	S	R	Output
0	X	X	No change
1	0	0	No change
1	0	1	Reset
1	1	0	Set
1	1	1	Indeterminate

① NAND LATCH

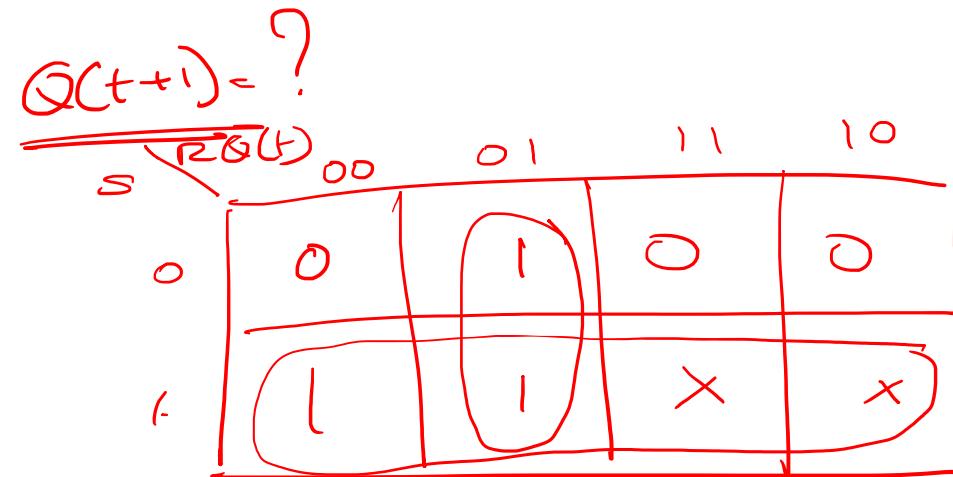


② NOR Latch



Characteristic equation:  $S-R$  flip flop

Characteristic Table			
S	R	Q(t)	Q(t+1)
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	X
1	1	1	X



$$Q(t+1) = S + \bar{R} Q(t)$$

Rider or not to use

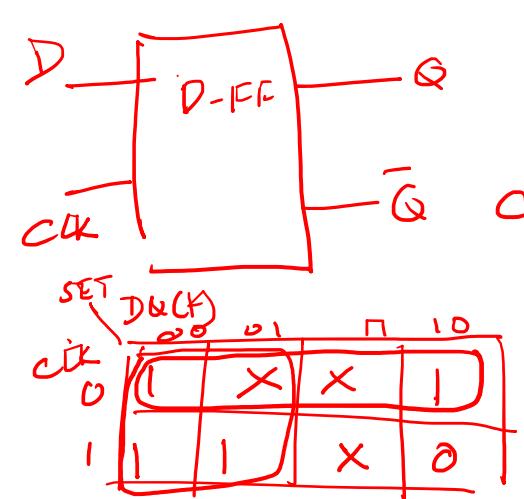
$S=1, R=1$  Not to use.

# D Flip Flop using NAND latch

Function Table		
Clk	D	Output $Q(t+1)$
0	X	No change
1	0	0
1	1	1

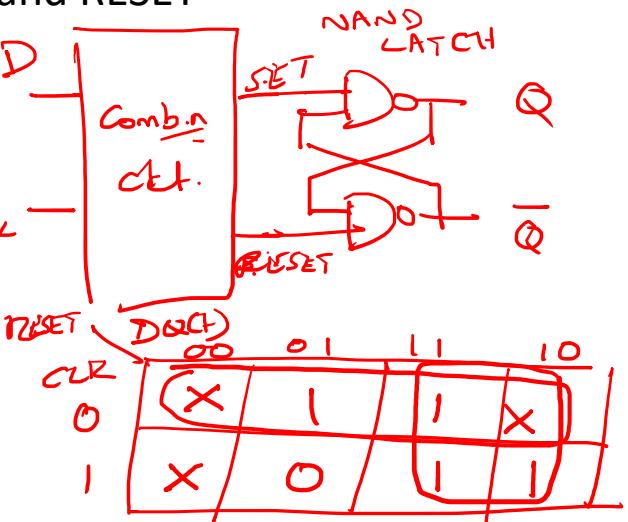
Level

Equation for SET and RESET



$$\text{SET} = \overline{\text{CLR}} + \overline{D} = \overline{\text{CLR}} \cdot \overline{D}$$

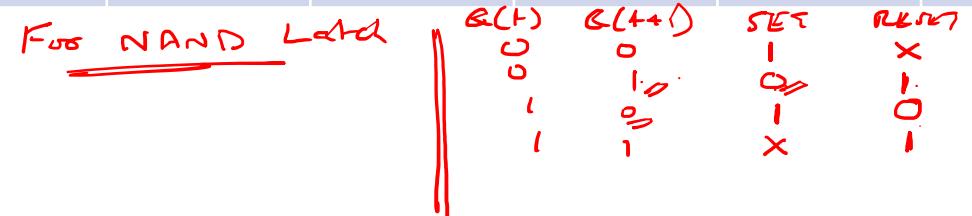
$$\text{RESET} = \overline{\text{CLK}} + D$$



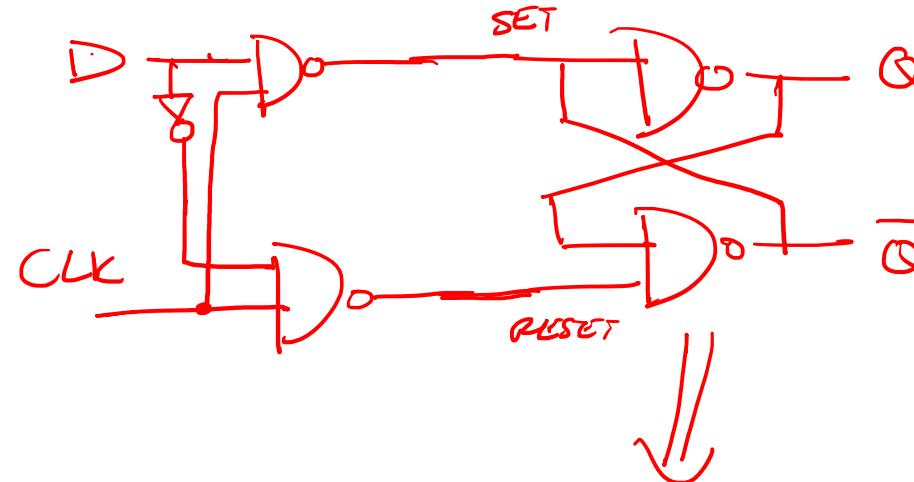
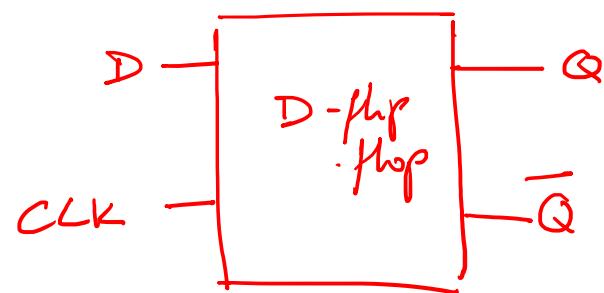
$$\text{SET} = \overline{\text{CLK}} \cdot D$$

$$\text{RESET} = \overline{\text{CLK}} \cdot \bar{D}$$

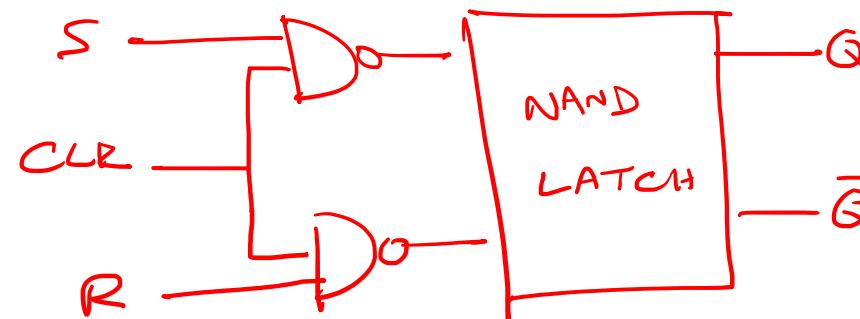
Excitation Table					
Clk	D	$Q(t)$	$Q(t+1)$	SET	RESET
0	0	0	0	1	X
0	0	1	1	X	1
0	1	0	0	1	X
0	1	1	1	X	1
1	0	0	0	1	X
1	0	1	0	1	0
1	1	0	0	1	0
1	1	1	1	0	1



# Circuit and block diagram of D Flip Flop

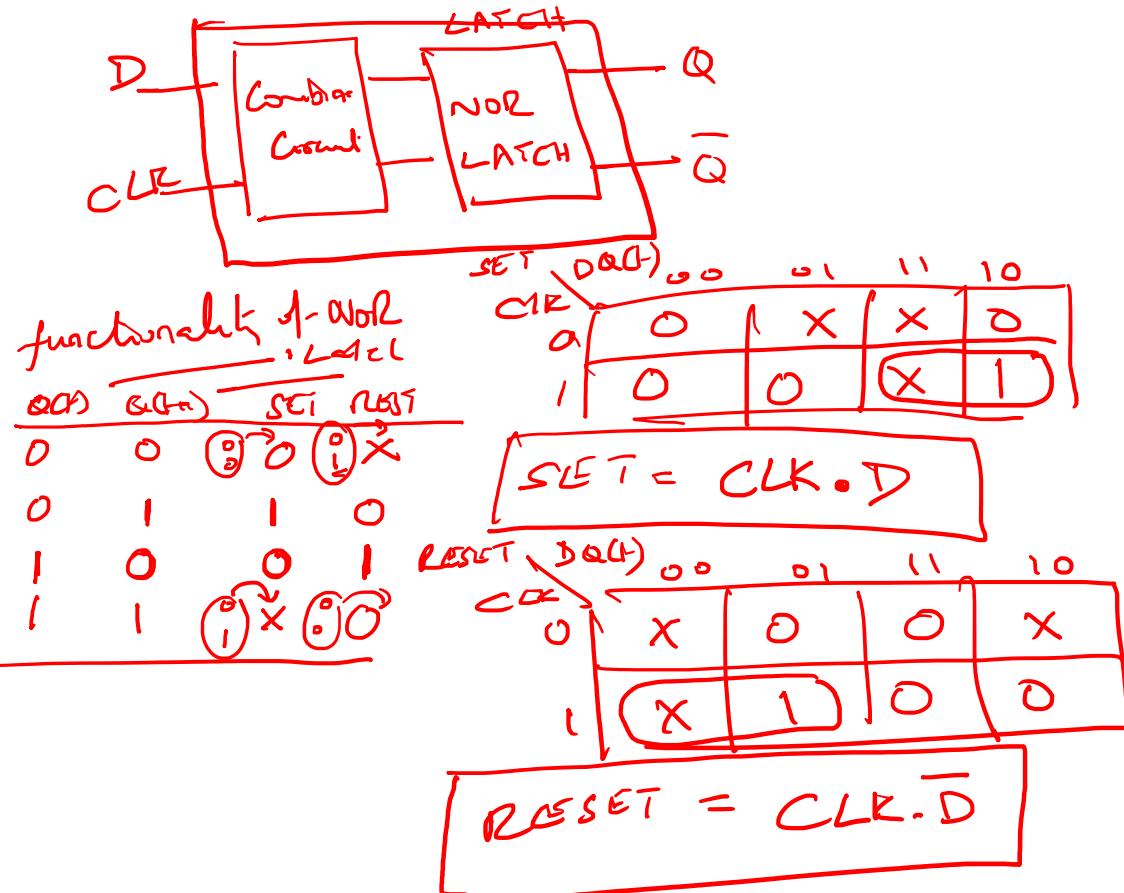


S-R flip flop w/t  
NAND LATCH



# D Flip Flop using NOR Latch

Equation for SET and RESET



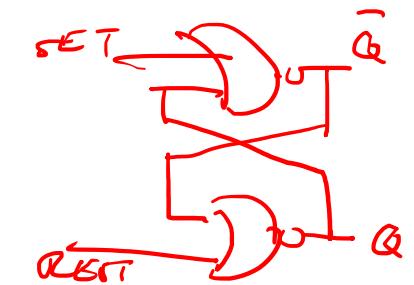
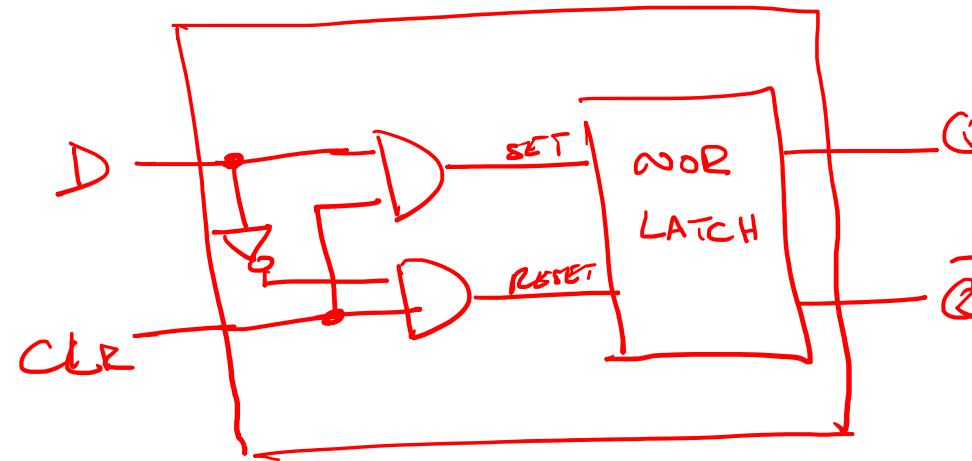
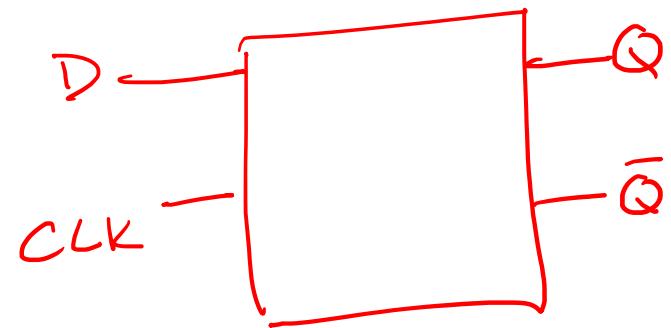
Excitation Table

Clk	D	Q(t)	Q(t+1)	SET	RESET
0	0	0	0	0	X
0	0	1	1	X	0
0	1	0	0	0	X
0	1	1	1	X	0
1	0	0	0	0	X
1	0	1	0	0	1
1	1	0	1	1	0
1	1	1	1	X	0

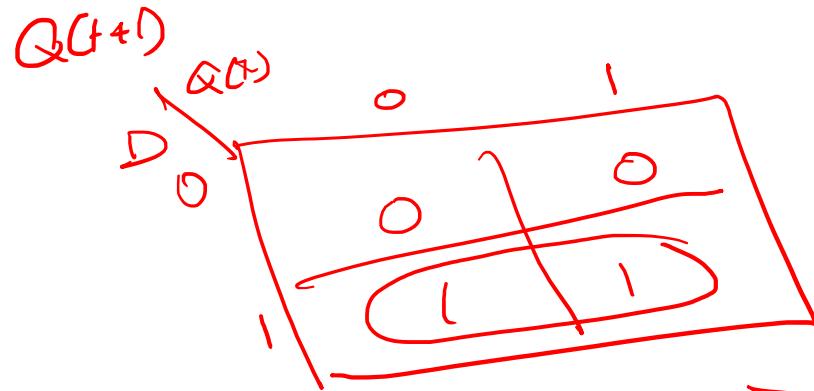
# Circuit and block diagram of D Flip Flop

$$SET = CLK \cdot D$$

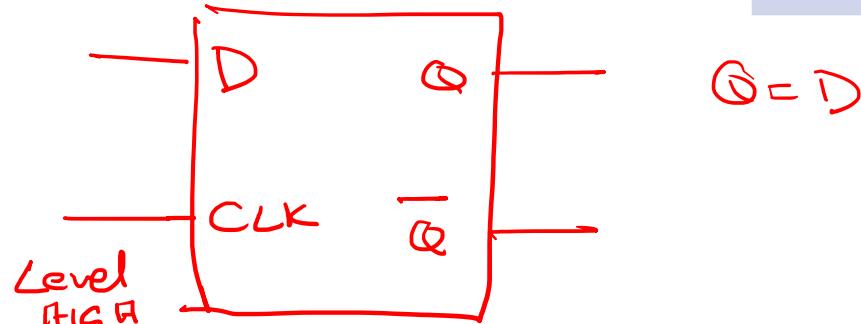
$$QRESET = CLK \cdot \bar{D}$$



Characteristic equation:



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



$$Q = D$$

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

$$Q(t+1) = \bar{CLK} Q(t) + CLK D$$

Characteristic Table for D FF		
D	Q(t)	Q(t+1)
0	0	0
0	1	0
1	0	1
1	1	1

CLK	D	Q(t)	Q(t+1)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

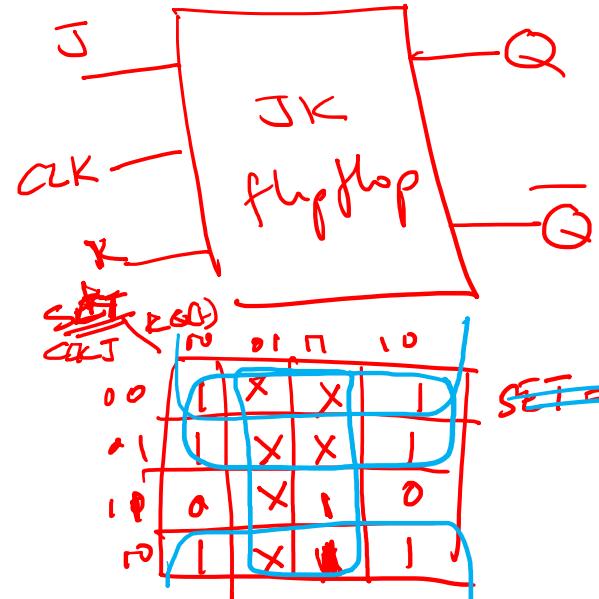
# JK JK Flip Flop using NAND latch

Function Table of JK flip flop

Clk	J	K	Output $Q(t+1)$
0	X	X	No Change or Retain previous state
1	0	0	No Change = $Q(t)$
1	0	1	Reset $Q(t) = 0$
1	1	0	Set $Q(t) = 1$
1	1	1	Toggle $Q(t)$

Equation for SET and RESET

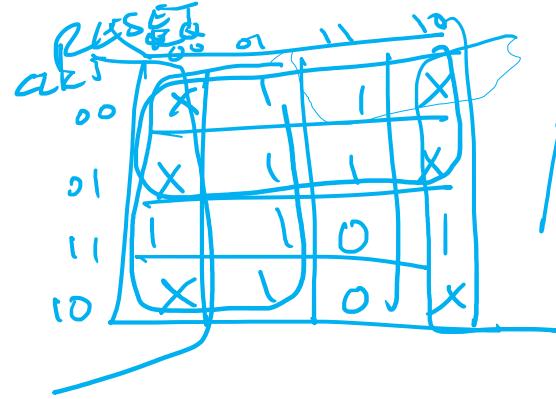
NAND LATCH Function			
$Q(t)$	$Q(t+1)$	SET	RESET
0	1	1	X
0	0	0	1
1	0	1	0
1	1	X	1



$$\begin{aligned} SET &= Q(t) \\ SET &= \overline{Q(t)} \cdot \overline{CLK} \cdot J \\ SET &= \overline{Q(t)} \cdot CLK \cdot J \end{aligned}$$

Clk	J	K	Q(t)	Q(t+1)	SET	RESET
0	0	0	0	0	1	X
0	0	0	1	1	X	1
0	0	1	0	0	1	X
0	0	1	1	1	X	1
0	1	0	0	0	1	X
0	1	0	1	1	X	1
0	1	1	0	0	1	X
0	1	1	1	1	X	1
1	0	0	0	0	1	X
1	0	0	1	1	X	1
1	0	1	0	0	1	X
1	0	1	1	0	1	X
1	1	0	0	1	0	1
1	1	0	1	0	1	X
1	1	1	0	1	0	1
1	1	1	1	0	1	0

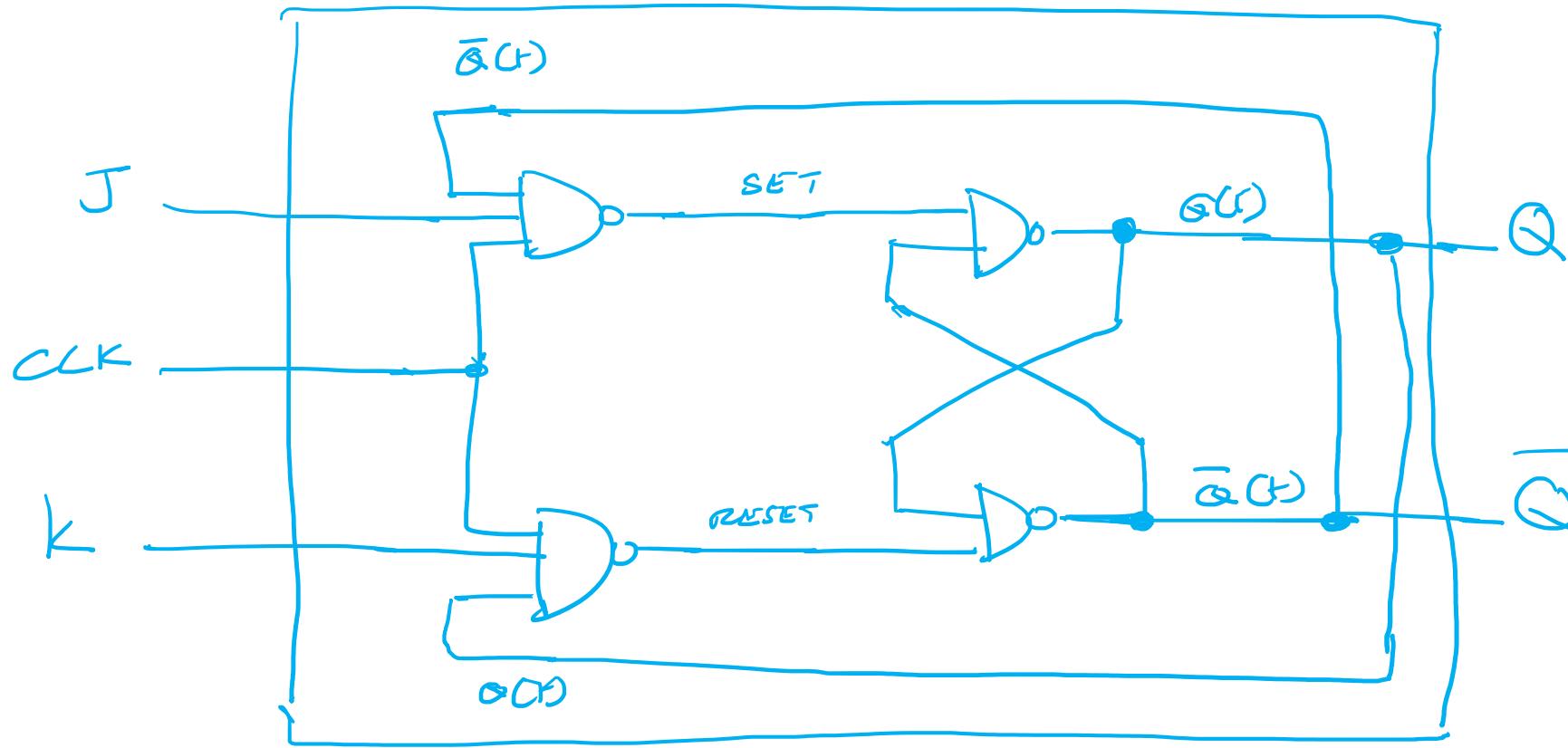
# Circuit and block diagram of JK flip flop:



$$\text{RESET} = \bar{K} + \bar{C}LK + \bar{Q}(t)$$

$$\text{RESET} = \bar{CLK} \cdot K \cdot Q(t)$$

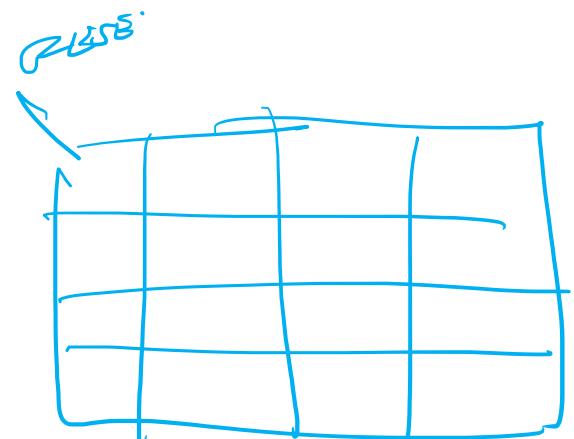
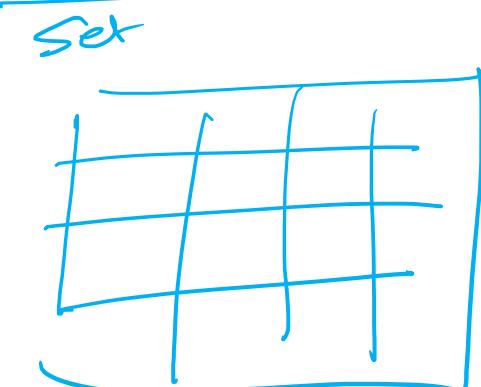
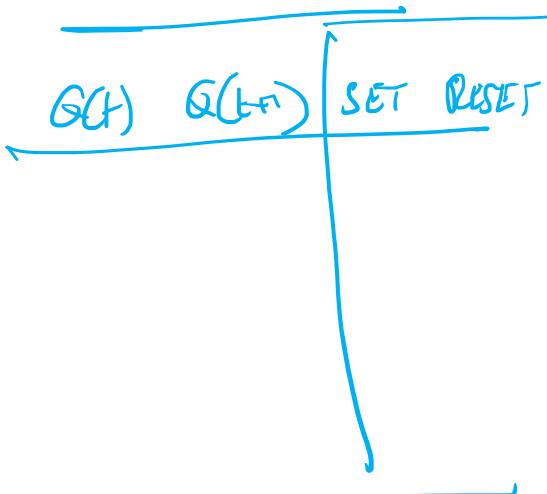
$$\text{SET} = \bar{CLK} \cdot J \cdot \bar{Q}(t)$$



# JK Flip Flop using NOR latch

## Equation for SET and RESET

functionality of NOR LATCH



Functional Table of JK

CLR	J	K	output
0	x	x	No change
1	0	0	No change
1	0	1	Reset
1	1	0	SET
1	1	1	Toggle

**Excitation Table**

Clk	J	K	Q(t)	Q(t+1)	SET	RESET
0	0	0	0	0	0	
0	0	0	1	1		
0	0	1	0	0	0	
0	0	1	1	1	1	
0	1	0	0	0	0	
0	1	0	1	1	1	
0	1	1	0	0	0	pls. Enter
0	1	1	1	1	1	
1	0	0	0	0	0	
1	0	0	1	1	1	
1	0	1	0	0	0	RESET
1	0	1	1	1	1	
1	1	0	0	0	1	SET
1	1	0	1	1	1	
1	1	1	0	1	1	
1	1	1	1	0	0	

# Circuit and Block Diagram

Circuit-

& Block diagram

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

Characteristic equation:

$$\bar{J} \bar{Q}(t) + \bar{K} Q(t) = 0$$

Characteristic Equation

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

# SEQUENTIAL CIRCUITS

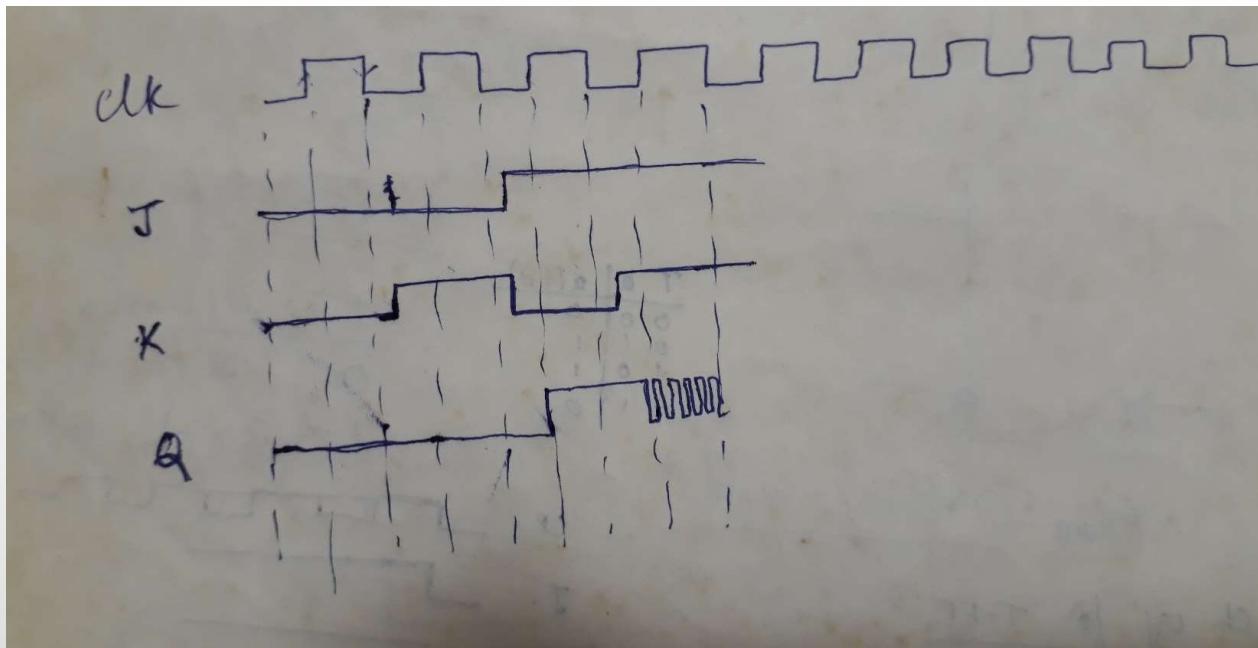
- Race around condition
- Master slave model
- Flip flop conversion

# Race Around Condition

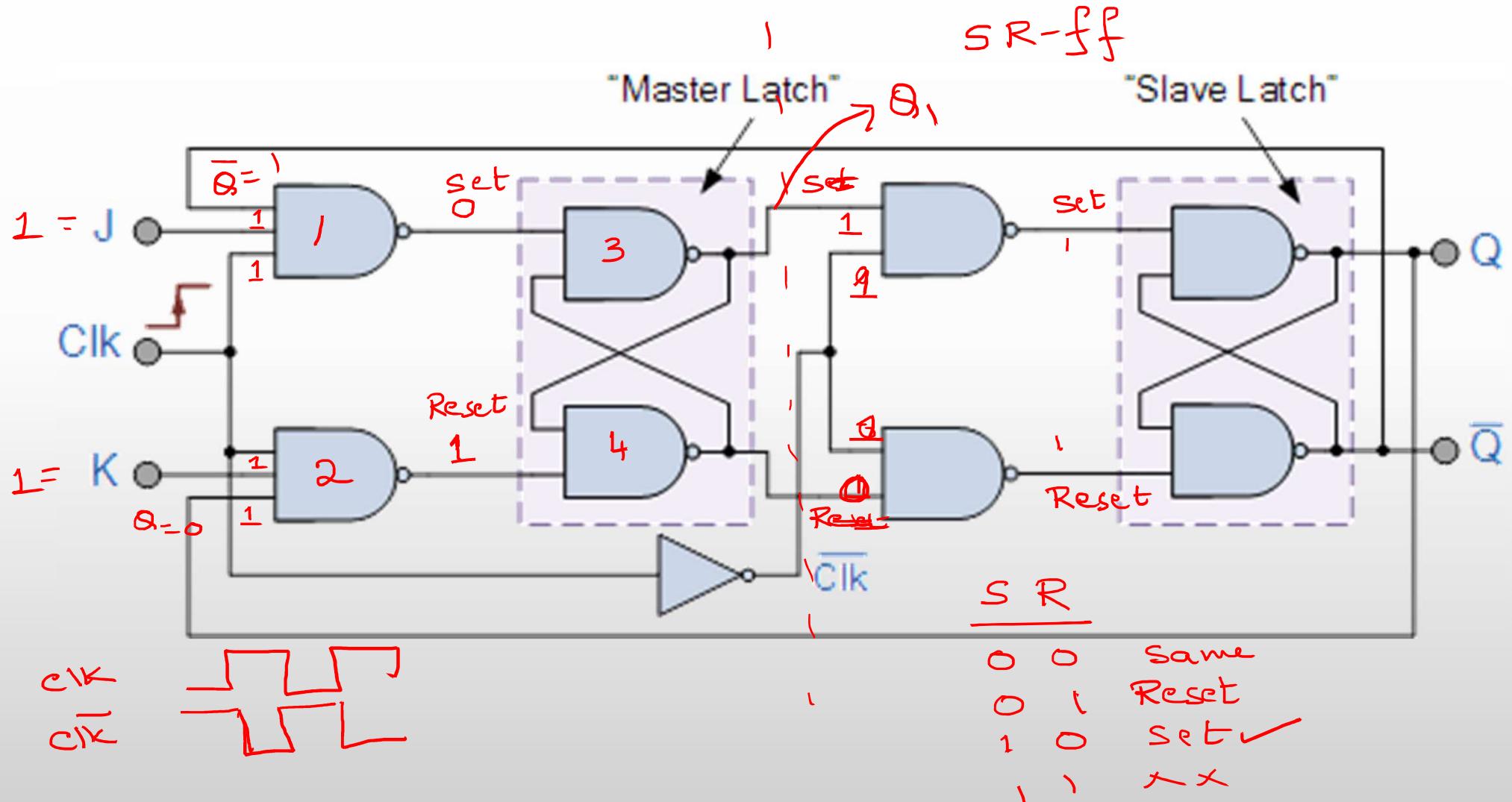
- For JK flip flop if J, K and Clock are equal to 1 the state of flip-flop keeps on toggling which leads to uncertainty in determining the output of the flip-flop. This problem is called **Race around the condition**.

# Race Around Condition

- For JK flip flop if J, K and Clock are equal to 1 the state of flip-flop keeps on toggling which leads to uncertainty in determining the output of the flip-flop. This problem is called **Race around the condition**.

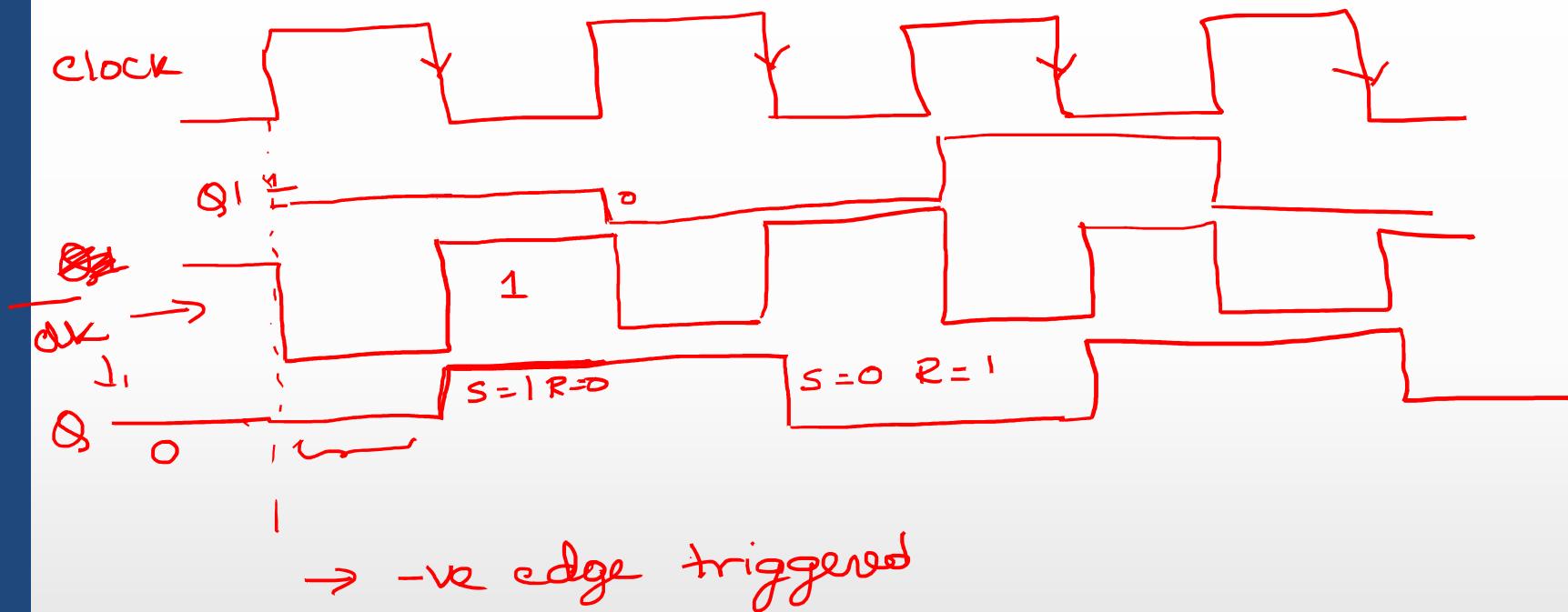


# Master Slave JK Flip Flop

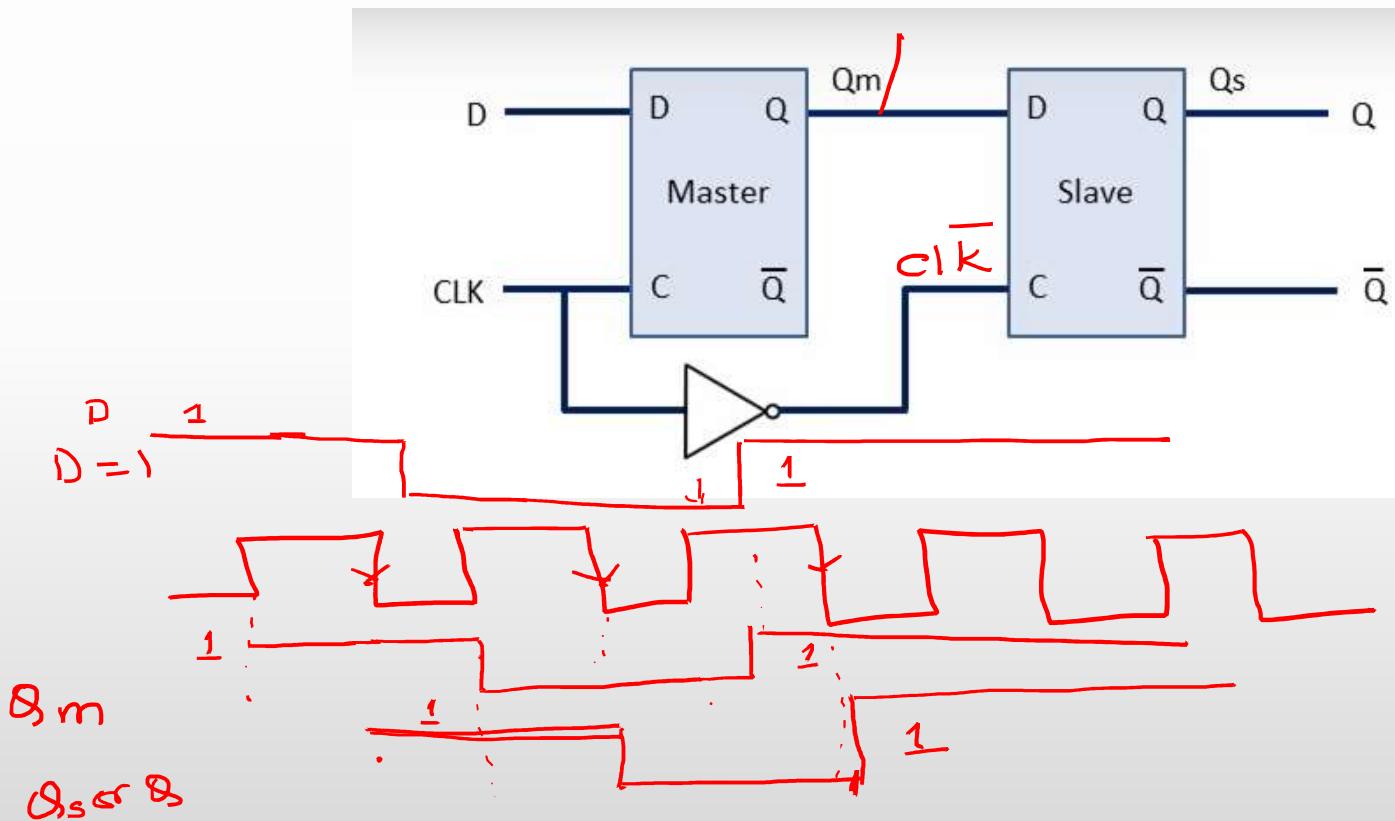


# Master Slave JK Flip Flop

$J = K = 1$

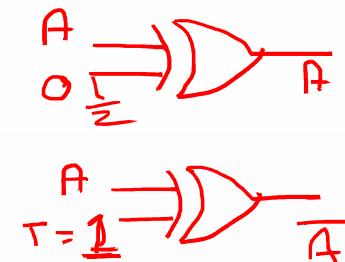
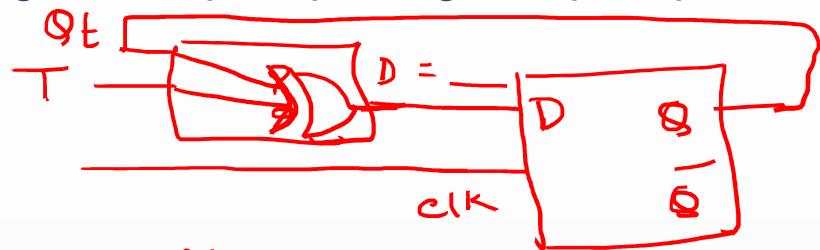


# Master Slave D Flip Flop



# Flip Flop Conversion

1. Design a T Flip Flop using D Flip Flop



TR of T-ff

Refer to excitation table of D-ff

T	$Q_t$	$Q_{t+1}$	D
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

$D(T, Q_t) = T \oplus Q_t$

2. Convert a given SR flip flop to work as a JK flip flop

$J$	$K$	$Q_t$	$Q_{t+1}$	$S$	$R$
0	0	0	0	0	0
0	0	1	1	0	0
0	1	0	0	0	0
0	1	1	0	0	1
1	0	0	1	0	0
1	0	1	1	0	0
1	1	0	1	0	0
1	1	1	0	0	0

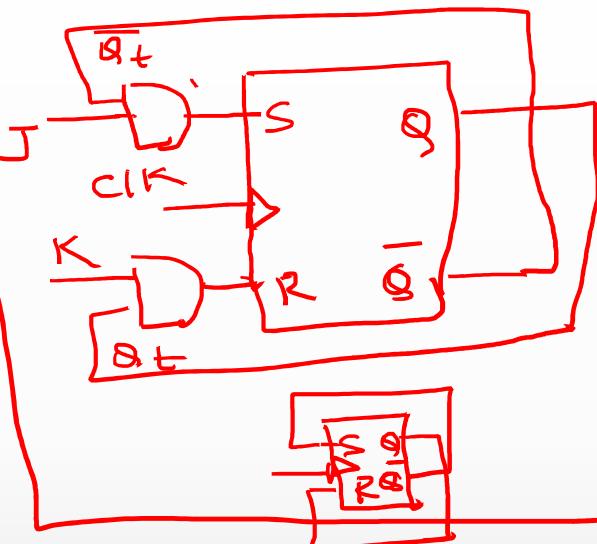
$J$	$K$	$Q_t$	$Q_{t+1}$	$\bar{Q}_{t+1}$
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	0	1

$$S = J \bar{Q}_t$$

$J=0$	$K$	$Q_t$	$Q_{t+1}$	$\bar{Q}_{t+1}$
0	0	0	0	1
0	0	1	1	0

$$R = K Q_t$$

Final ckt



Toggle for every  
clk cycle.

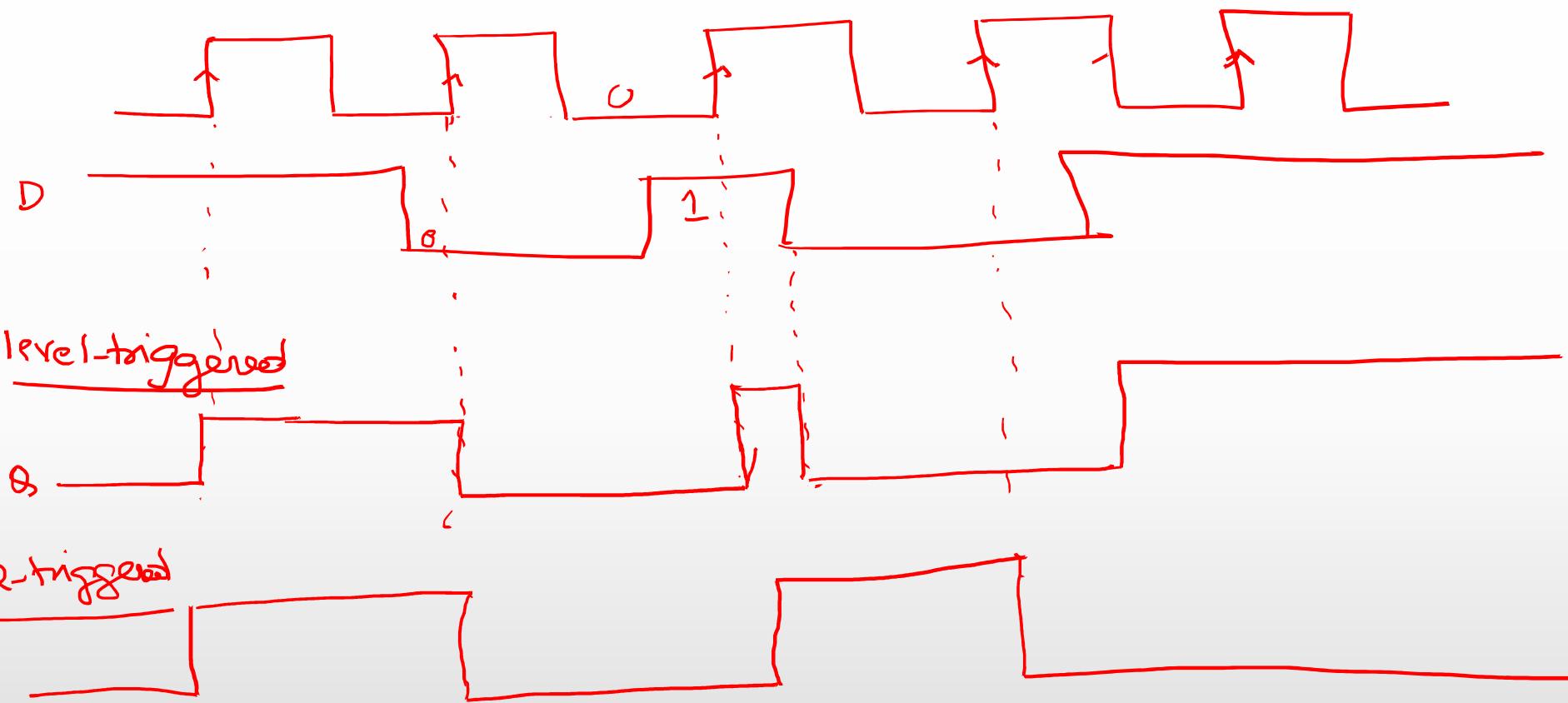
Excitation table of SR

$Q_t$	$Q_{t+1}$	$S$	$R$
0	0	0	0
0	1	1	0
1	0	0	1
1	1	1	1

$$S=0 \quad R=0 \quad / \quad S=0 \quad R=1$$

(No change/Reset)

$$\begin{array}{ll} \text{Set} & \\ S=0 \quad R=0 & S=1 \quad R=0 \\ (\text{No change/Set}) & \end{array}$$

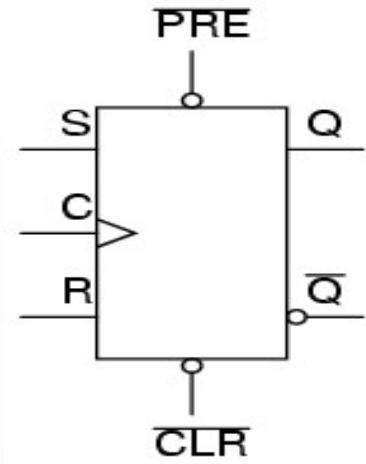


3. Design a D flip flop using an AB flip flop whose function table is given below.

A	B	Output
0	0	RESET
0	1	No Change
1	0	Toggle
1	1	SET

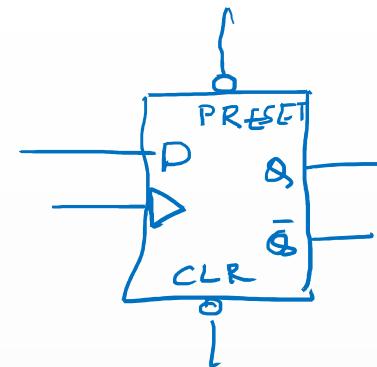
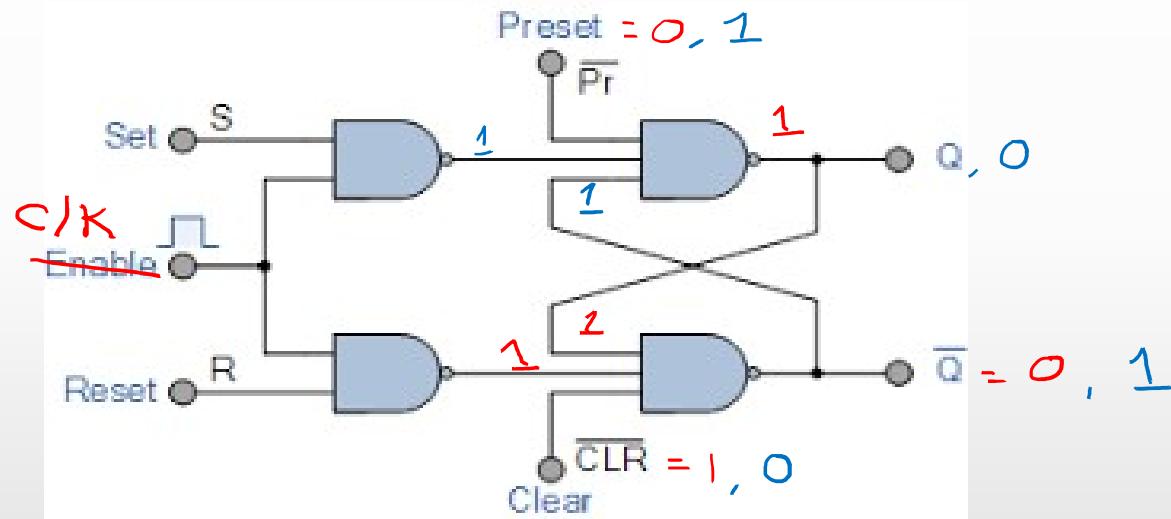
3. Design a D flip flop using an AB flip flop CONTD.

# Asynchronous Inputs:



PRESET	CLEAR	FF Response
0	0	Indeterminate <span style="color:red">X</span>
{ 0 ✓	1 ✓	SET
1	0	CLEAR
1	1	<span style="color:red">Clocked operation</span>

## Asynchronous Inputs:



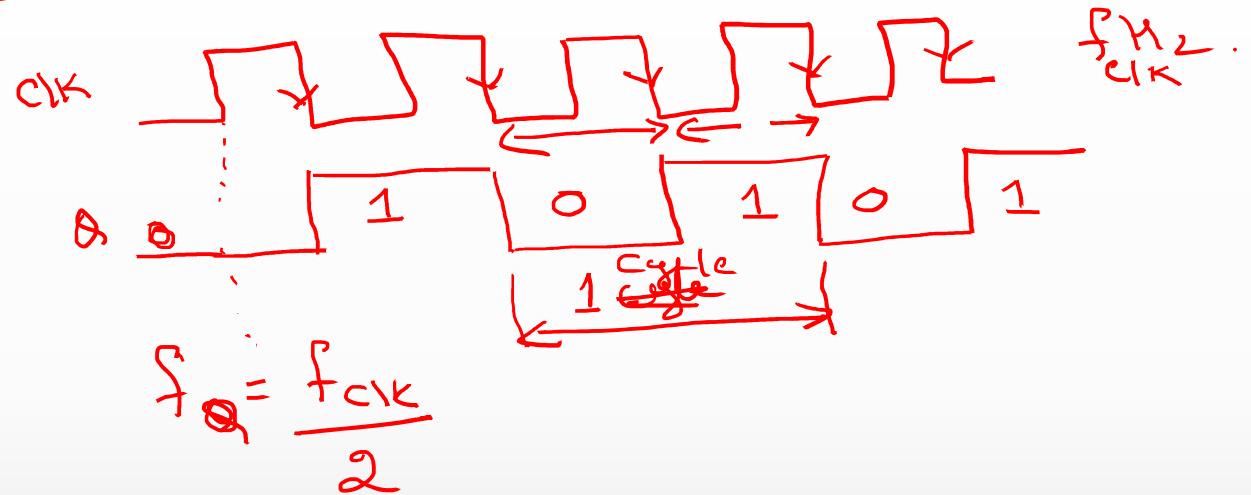
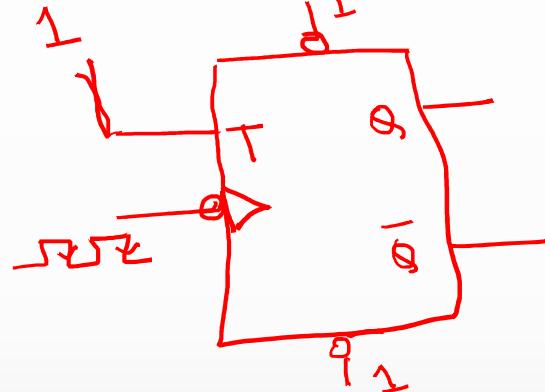
- Any Questions?

# ASYNCHRONOUS COUNTER(RIPPLE COUNTER)

# Counters:

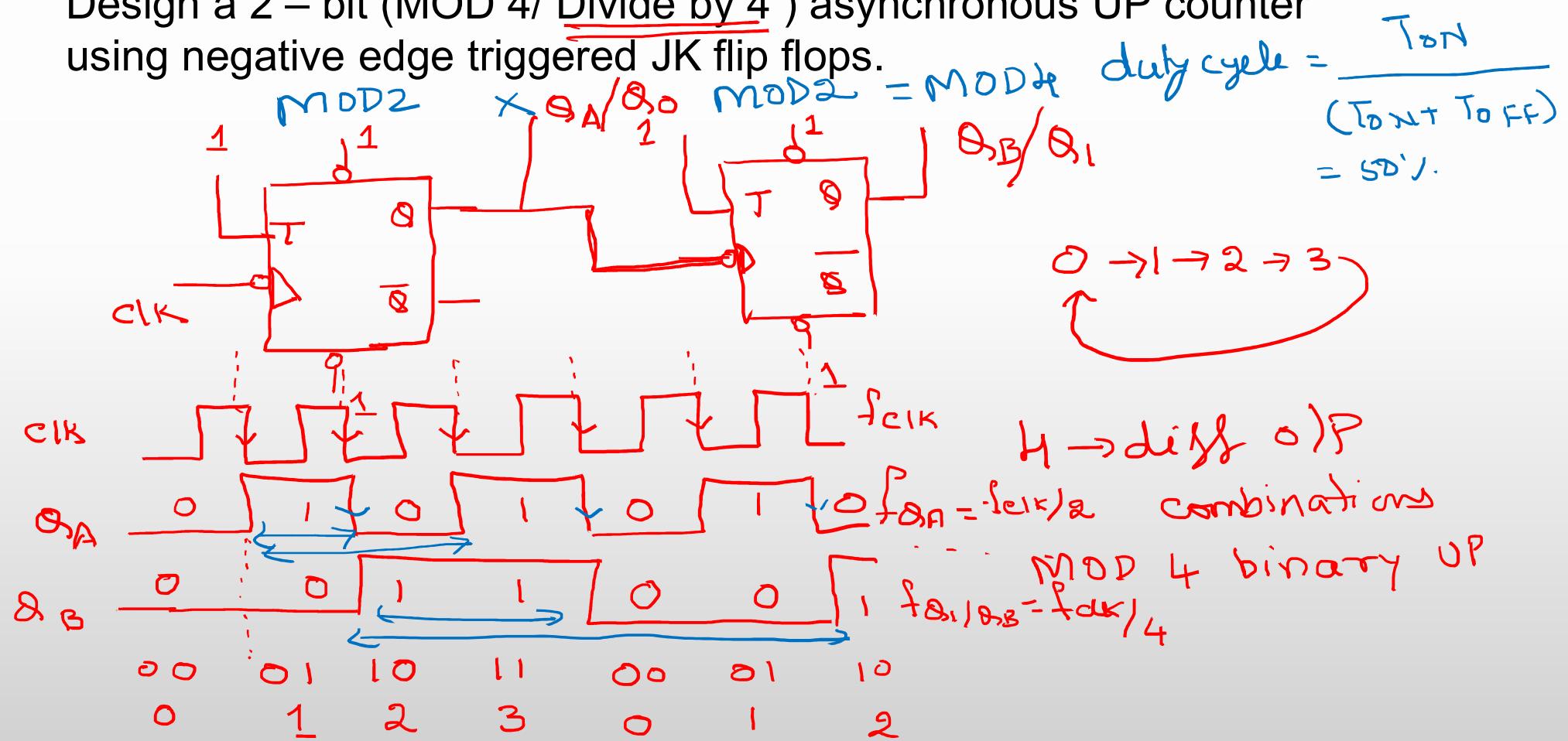
- Register that goes through prescribed sequence of states upon the application of input pulses is called a counter.
- There are 2 types of counters:
  - *Asynchronous counters (Ripple counters)* : Clock inputs are triggered by transitions of other flipflop.
  - *Synchronous counters* : The clock inputs of all flip flops receive common clock.

## MOD 2 or divide by 2 Counter using T ff



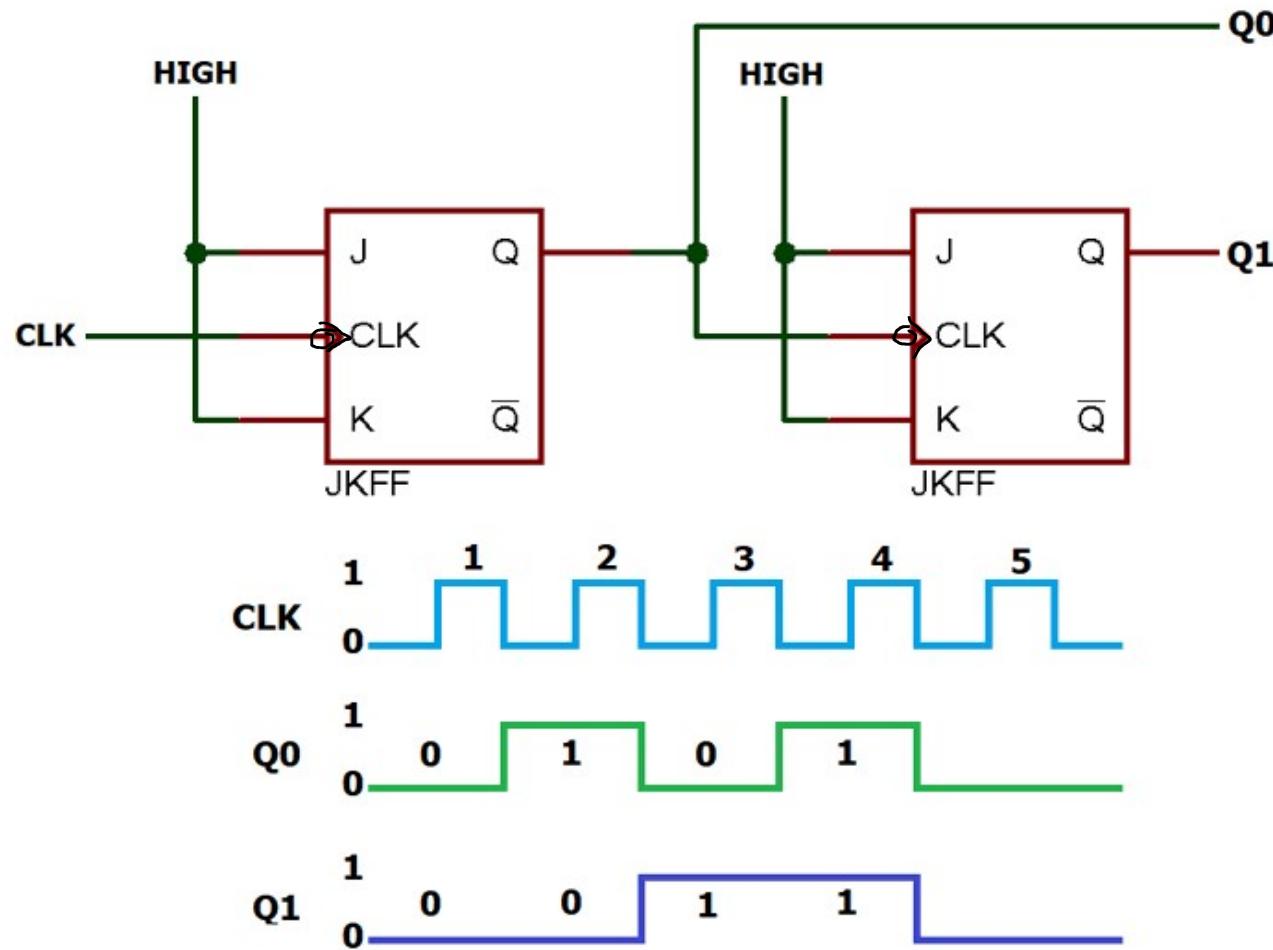
# Asynchronous counters (Ripple counters)

Design a 2 – bit (MOD 4/ Divide by 4 ) asynchronous UP counter using negative edge triggered JK flip flops.

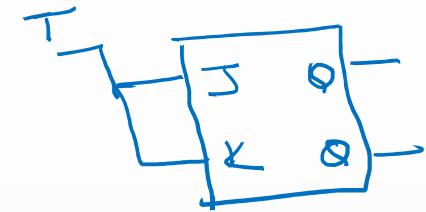
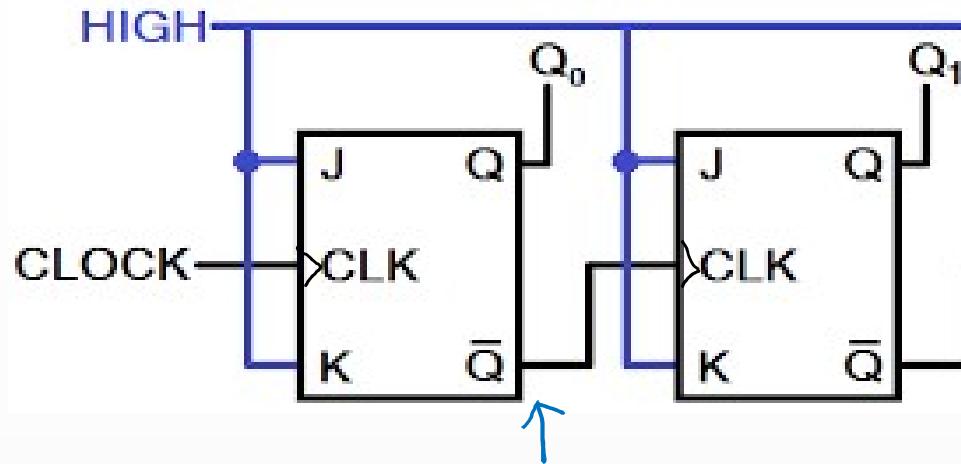


# Asynchronous counters (Ripple counters)

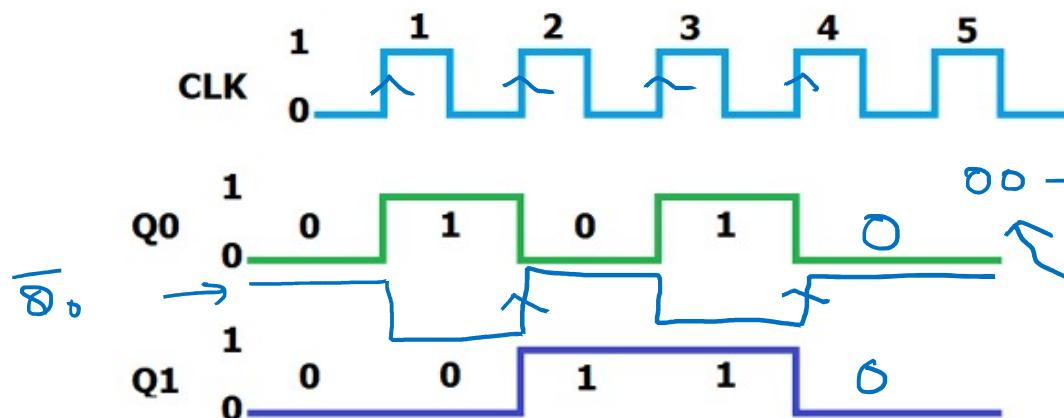
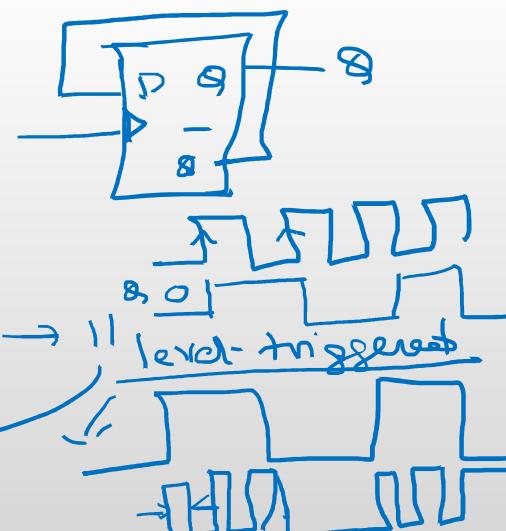
- Design a 2 – bit (MOD 4/ Divide by 4 ) Asynchronous UP counter using negative edge triggered JK flip flops.



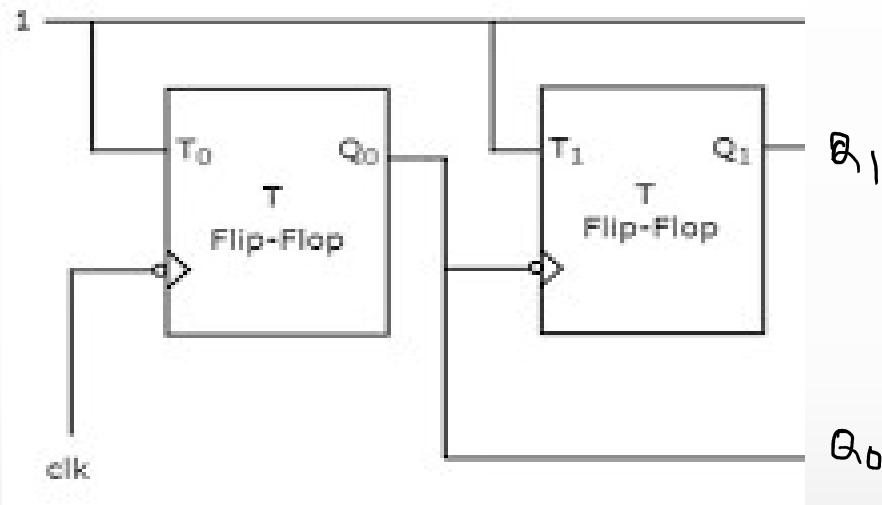
- Design a 2 – bit Asynchronous UP counter using positive edge triggered JK flip flops.



D-flip to T-flip

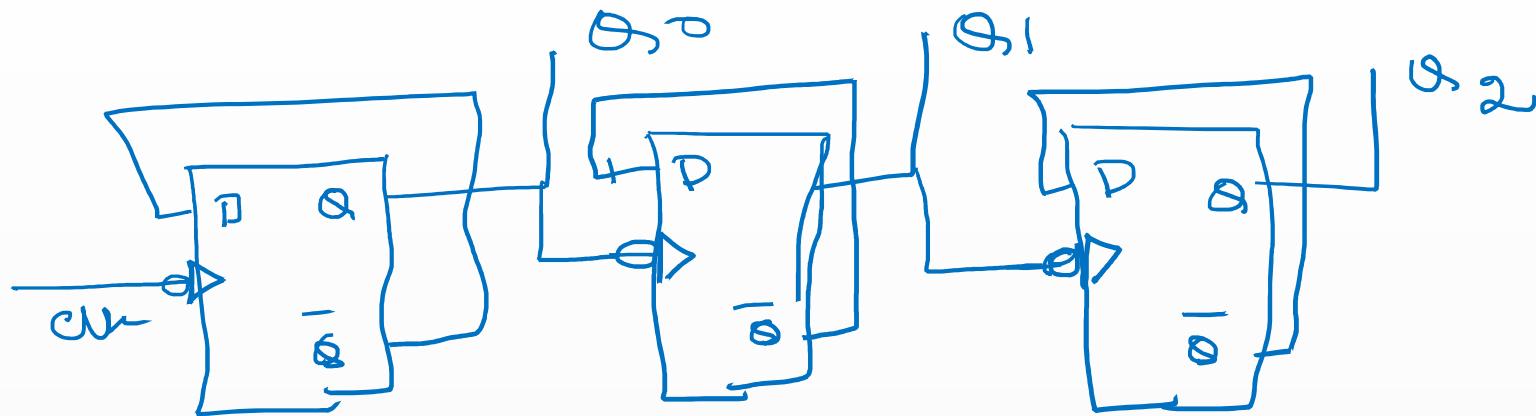


- Design a 2 – bit Asynchronous UP counter using negative edge triggered T flip flops.



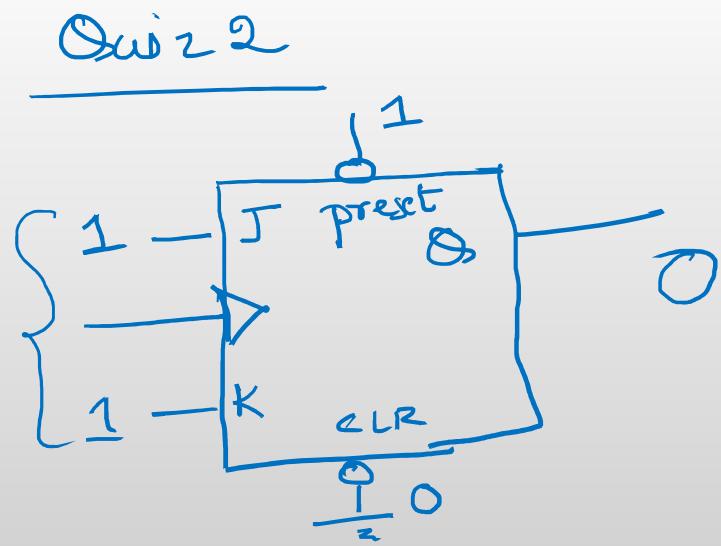
- Design a 2 – bit Asynchronous UP counter using positive edge triggered T flip flops.

- Design a 3 – bit (MOD 8/Divide by 8) Asynchronous UP counter using negative edge triggered D flip flops.



- Design a 3 – bit Asynchronous UP counter using positive edge triggered D flip flops.

Draw the circuit in your note book

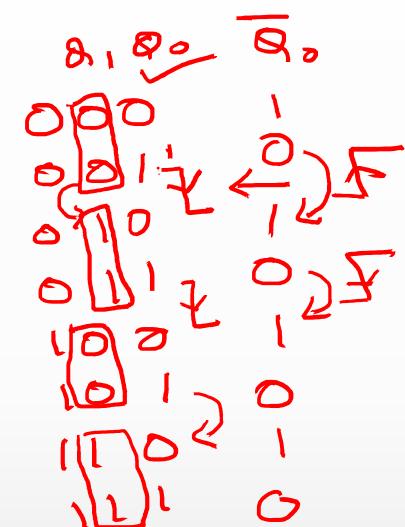
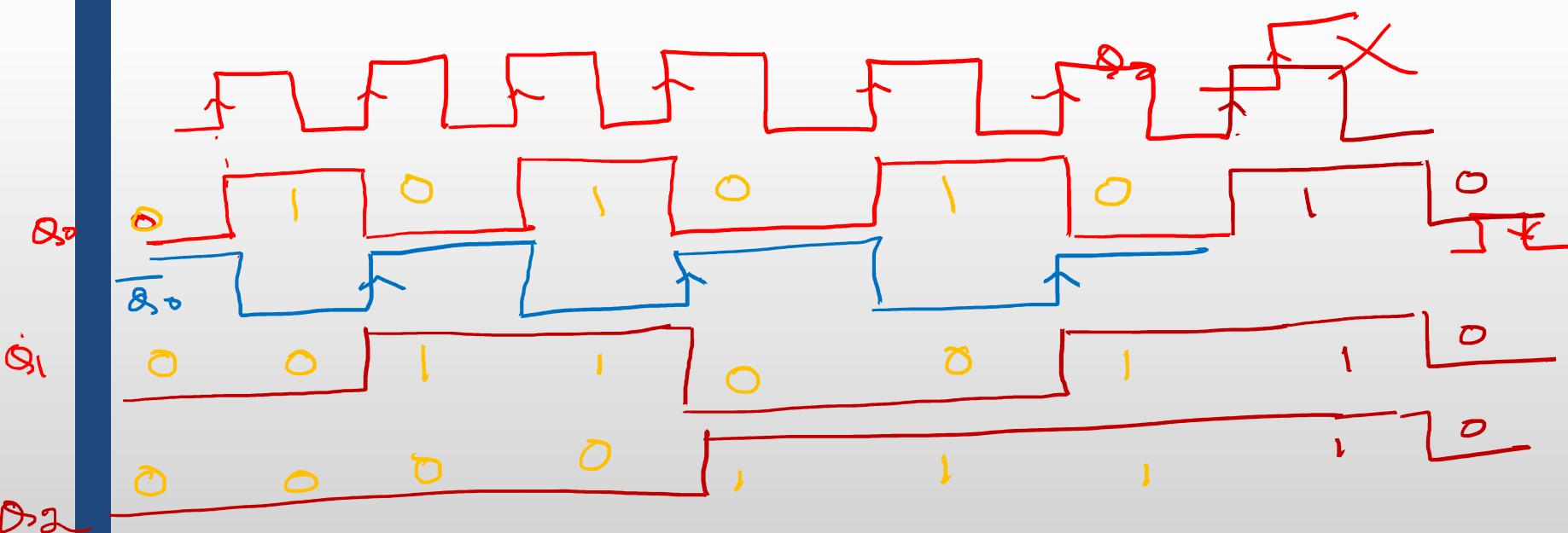
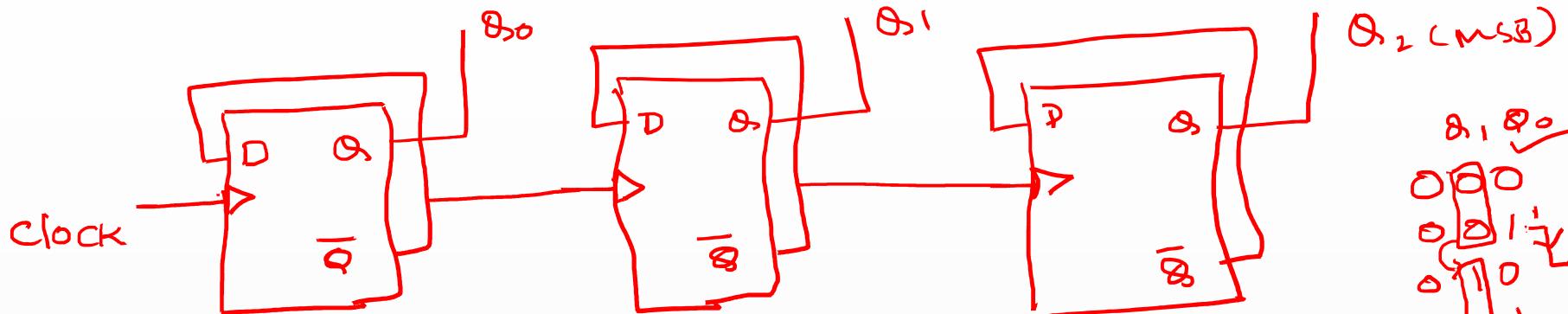


# ASYNCHRONOUS COUNTER(RIPPLE COUNTER)

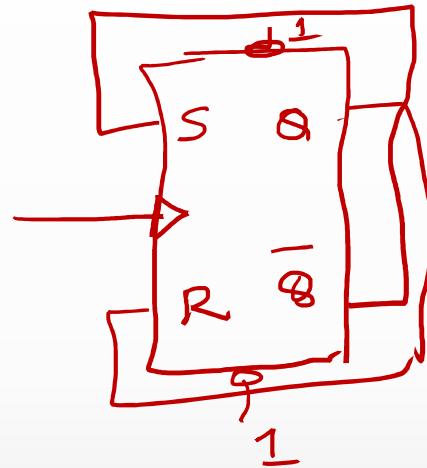
# Counters:

- Register that goes through prescribed sequence of states upon the application of input pulses is called a counter.
- There are 2 types of counters:
  - *Asynchronous counters (Ripple counters)* : Clock inputs are triggered by transitions of other flipflop.
  - *Synchronous counters* : The clock inputs of all flip flops receive common clock.

- Design a 3 – bit Asynchronous UP counter using positive edge triggered D flip flops.



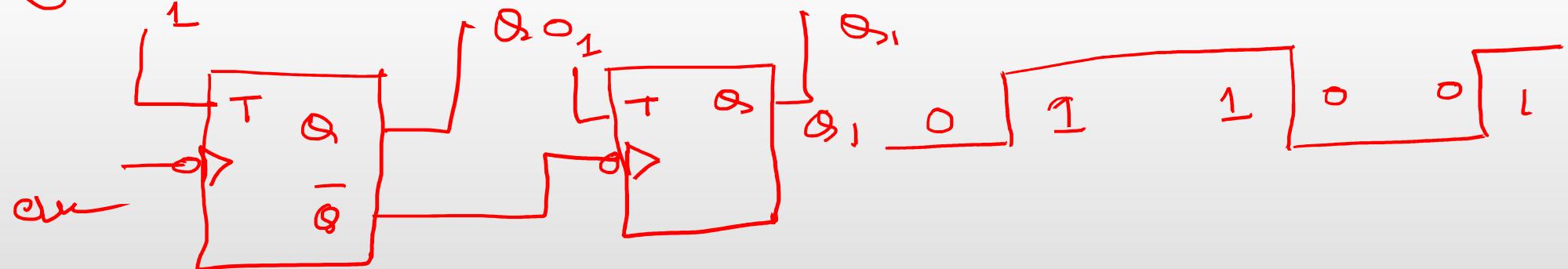
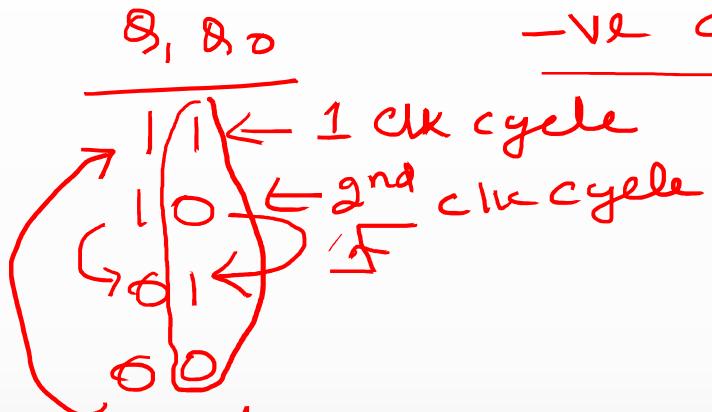
- Design a 4 – bit ( MOD 16/ Divide by 16) Asynchronous UP counter using negative edge triggered SR flip flops.
- Design a 4 – bit Asynchronous UP counter using positive edge triggered SR flip flops.



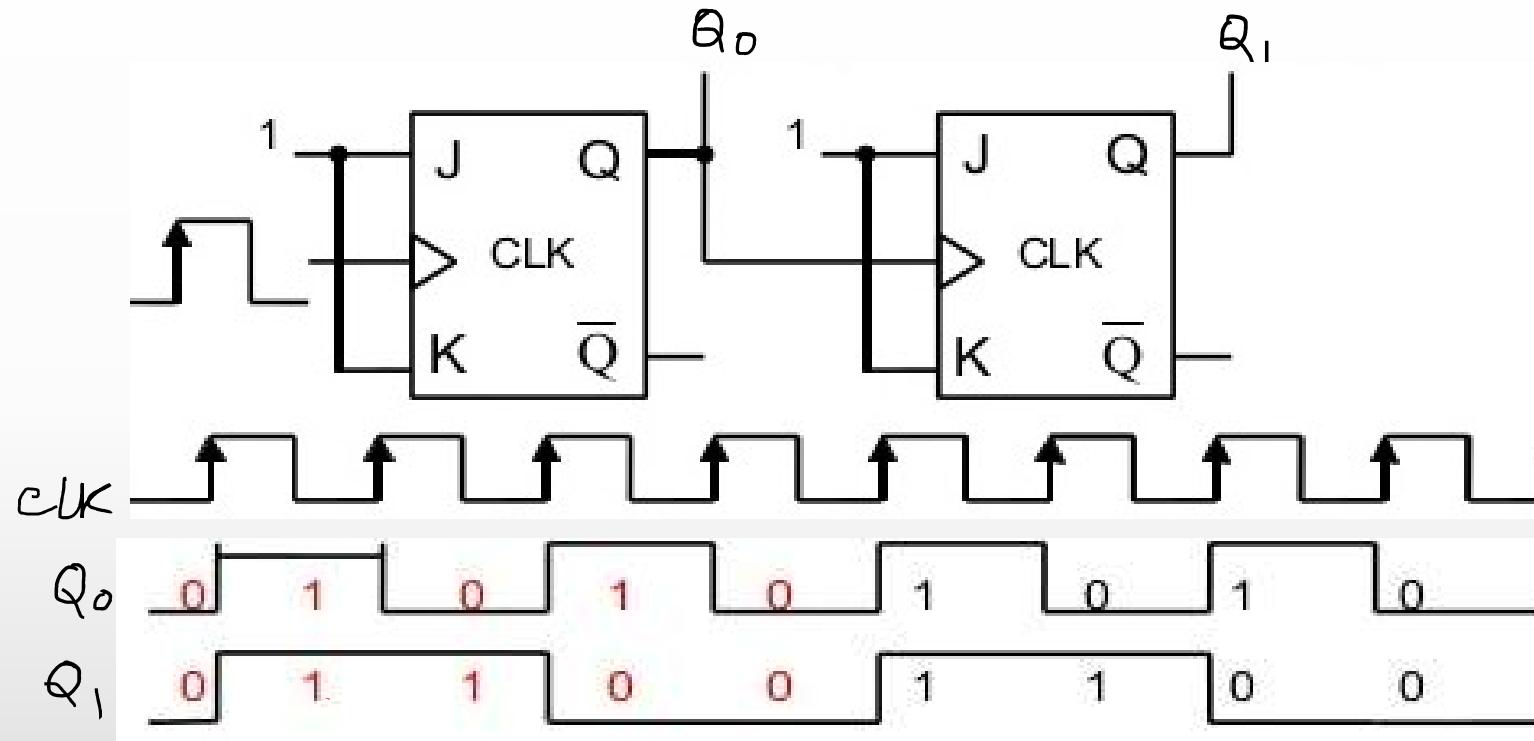
# **DOWN COUNTERS**

MOD4 down  $\Rightarrow$  2-bit

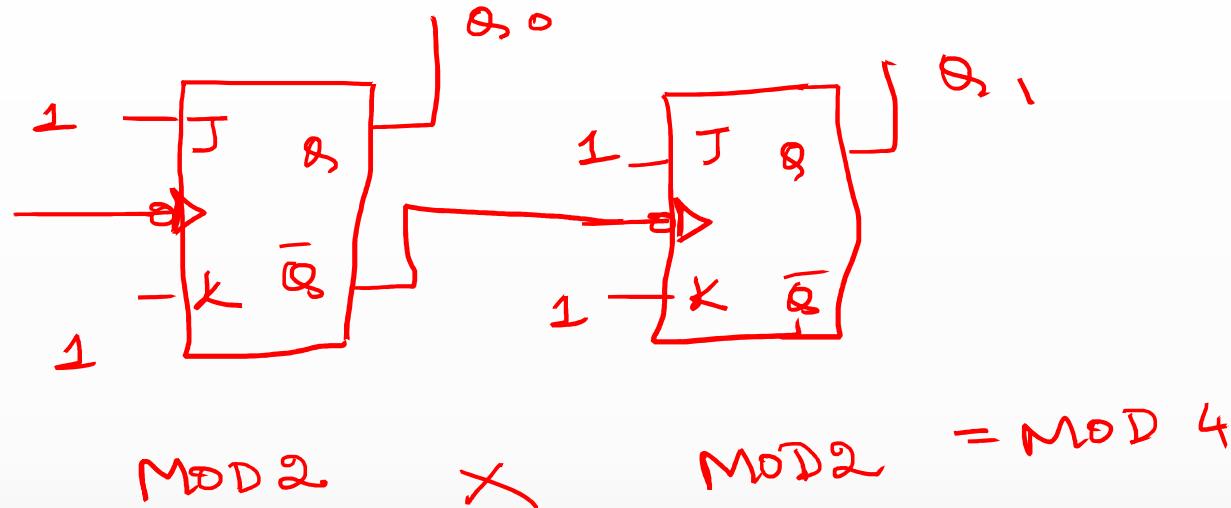
-ve edge triggered (JK, T, SR, D)



- Design a 2 – bit Asynchronous DOWN counter using positive edge triggered JK flip flops.



- Design a 2 – bit Asynchronous DOWN counter using negative edge triggered JK flip flops.



- Design a 2 – bit Asynchronous DOWN counter using negative edge triggered T flip flops.

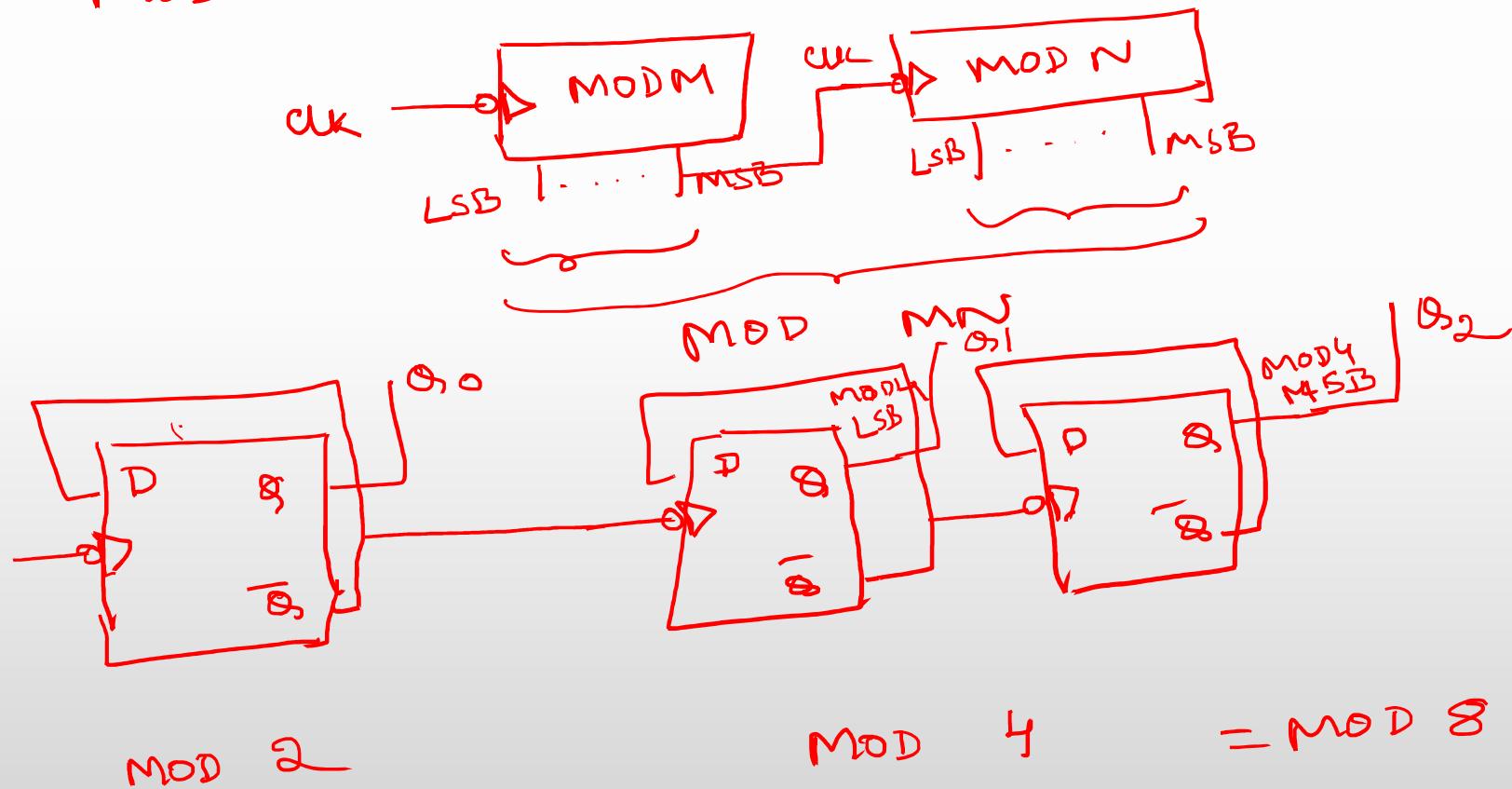
- Design a 2 – bit Asynchronous DOWN counter using positive edge triggered T flip flops.

Draw the circuit

- Design a 3 – bit Asynchronous DOWN counter using negative edge triggered D flip flops.

$$\text{MOD } 8 \Rightarrow \text{MOD } 4 \times \text{MOD } 2 = \underline{\text{MOD } 2 \times \text{MOD } 4} \quad \checkmark$$

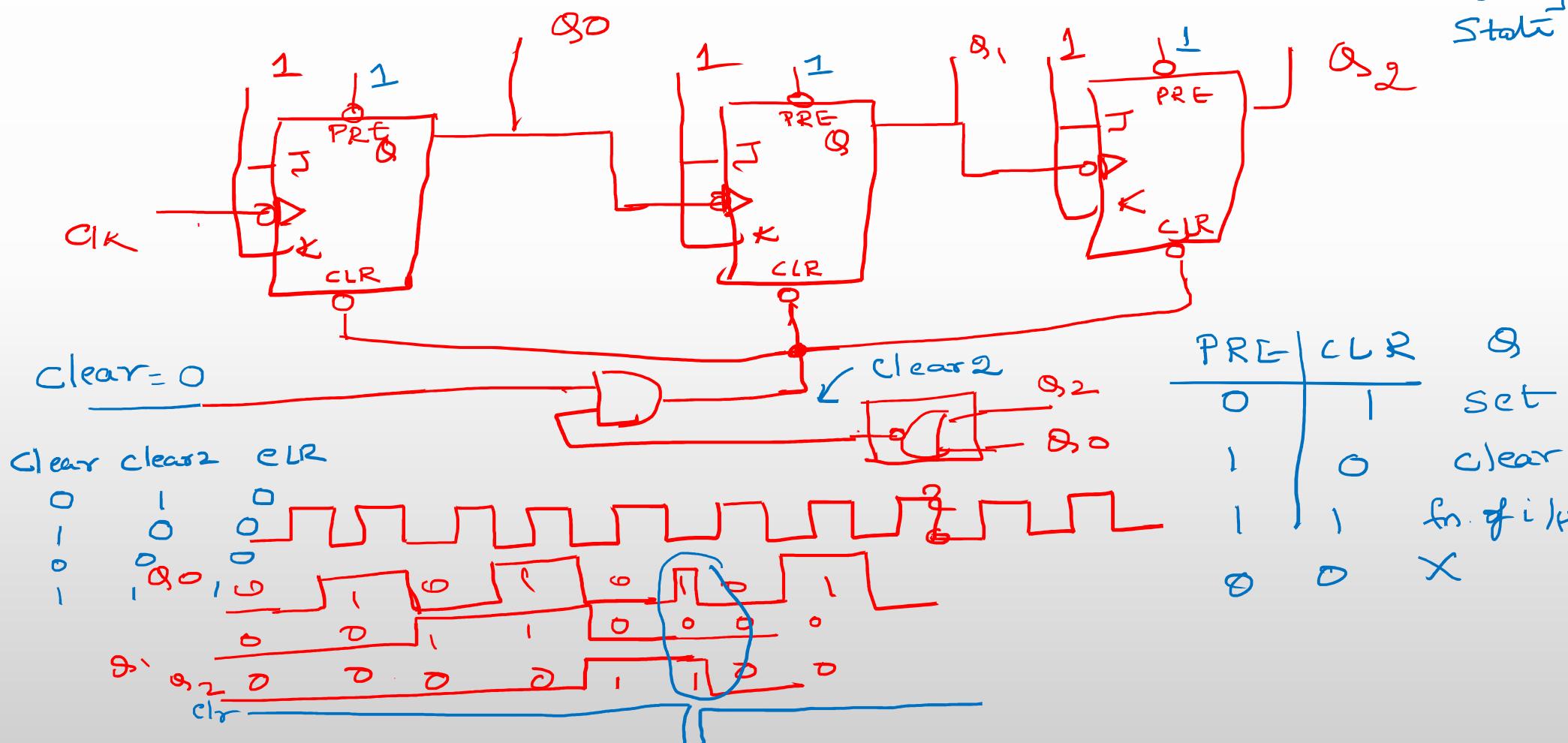
$$\text{MOD } MN \Rightarrow \text{MOD } M \times \text{MOD } N$$



- Design a 3 – bit Asynchronous DOWN counter using positive edge triggered D flip flops.

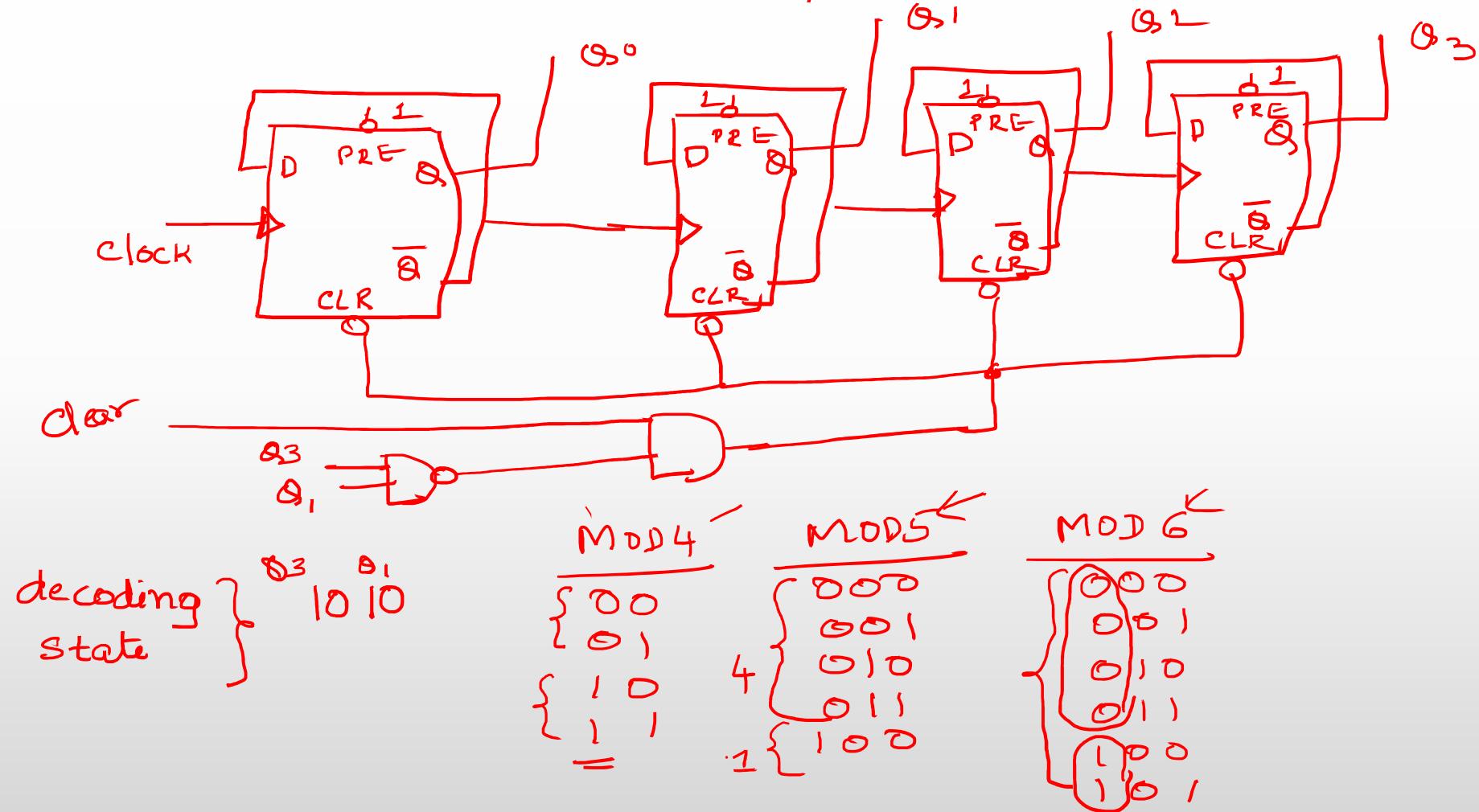
- Design a MOD 5 Asynchronous UP counter using negative edge triggered JK Flip Flops.

$0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 0$        $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 0$



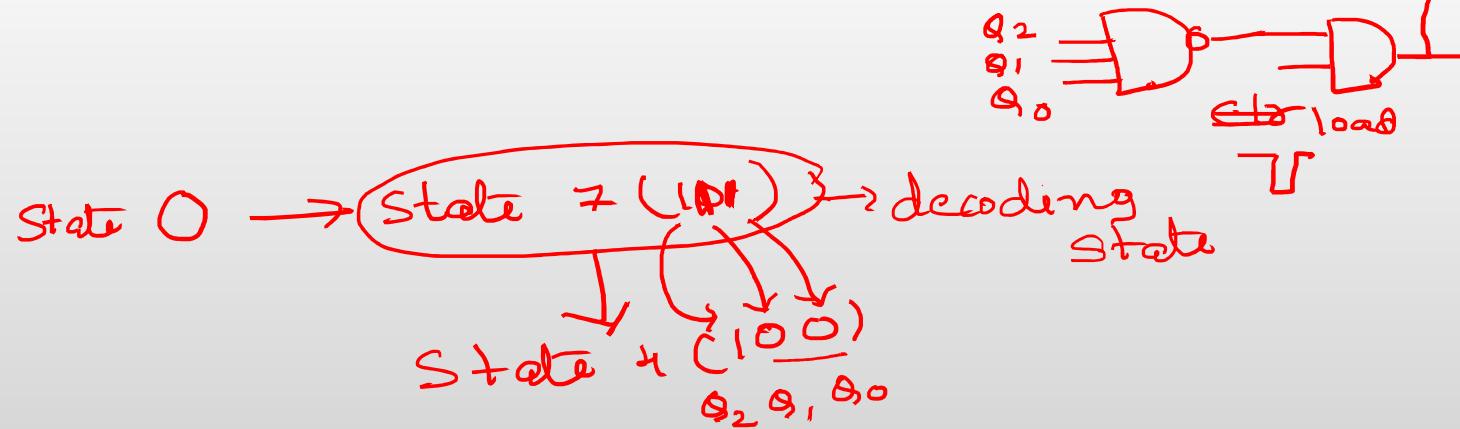
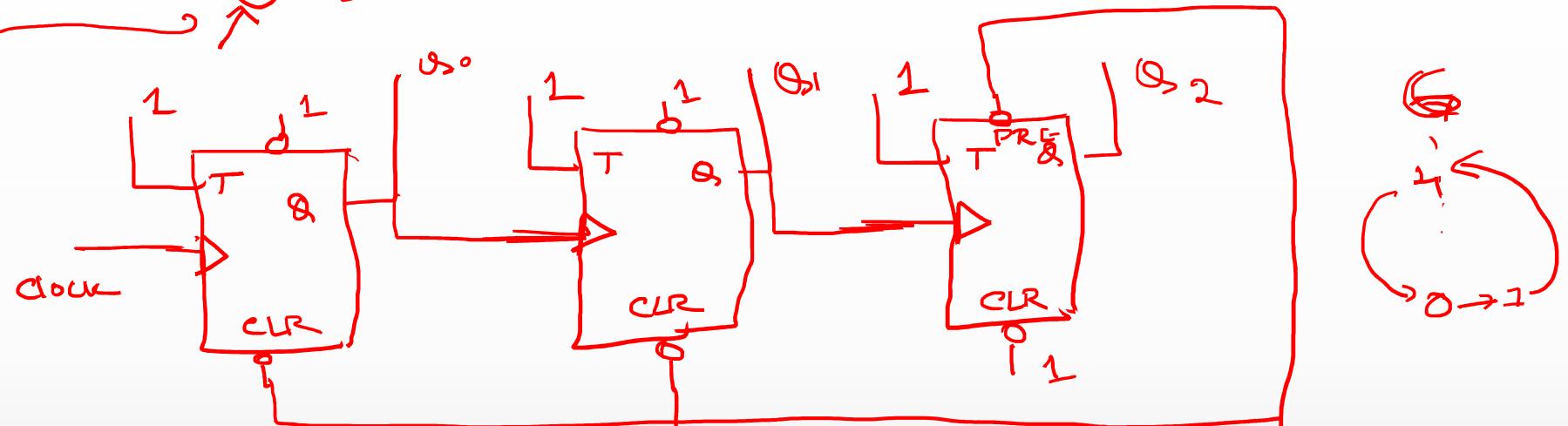
- Design a MOD 10 Asynchronous UP counter using positive edge triggered D Flip Flops.

$0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 8 \rightarrow 9 \rightarrow 0$       *Decade / Decimal*

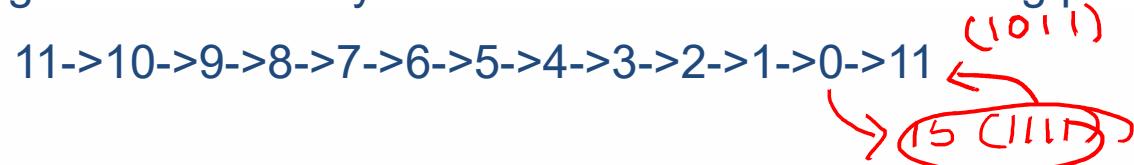


Design a MOD 5 Asynchronous DOWN counter using positive edge triggered T Flip Flops.

4->3->2->1->0->4       $\overline{100} \rightarrow 3-2-1-0 \rightarrow \dots$



- Design a MOD 12 Asynchronous DOWN counter using positive edge triggered T Flip Flops.

11->10->9->8->7->6->5->4->3->2->1->0->11  
  
15 (1111)

- Design a MOD 18 Asynchronous DOWN counter using positive edge triggered JK Flip Flops.

- Design a 3 bit Asynchronous UP/DOWN counter using negative edge triggered JK Flip Flops.

Design a counter to obtain a 1KHz clock signal from a 10KHz clock signal with 50% duty cycle using negative edge triggered JK Flip Flops.

UP counter

Square

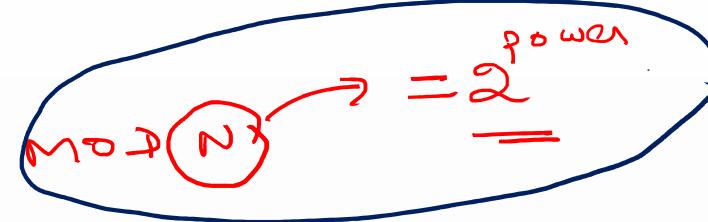
$$\text{MOD } 10 = \text{MOD } 5 \times \underbrace{\text{MOD } 2}, \quad 4 \text{ JFs}.$$

Dec. count	$Q_3\ Q_2\ Q_1\ Q_0$	Clock cycle
0	0 0 0 0	-1
1	0 0 0 1	-2
2	0 0 1 0	-3
3	0 0 1 1	-4
4	0 1 0 0	-5
5	0 0 0 0	-6
6	0 0 0 1	-7
7	0 0 1 0	-8
8	0 0 1 1	-9
9	0 1 0 0	-10
10	0 1 0 1	
11	0 1 1 0	
12	0 1 1 1	
	0 0 0 0	
	0 0 0 1	
		:

Design a MOD 6 Asynchronous counter with 50% duty cycle using negative edge triggered D Flip Flops.

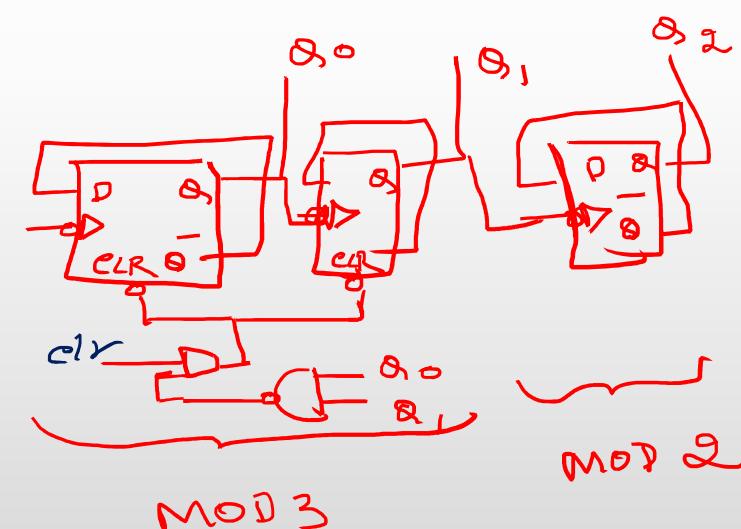
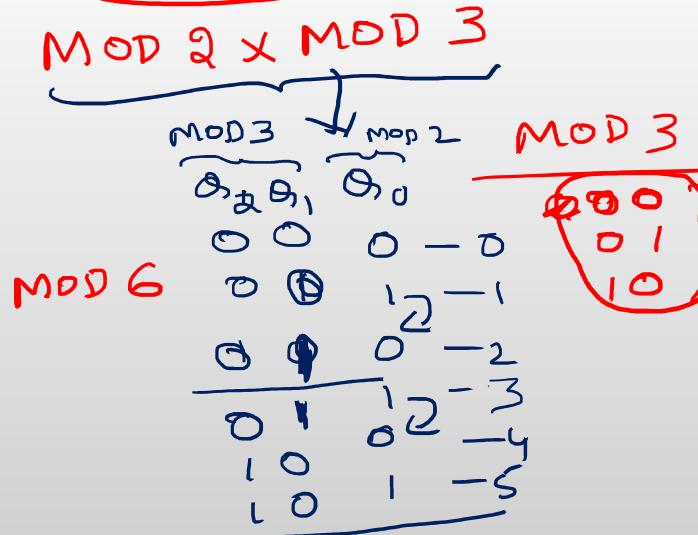
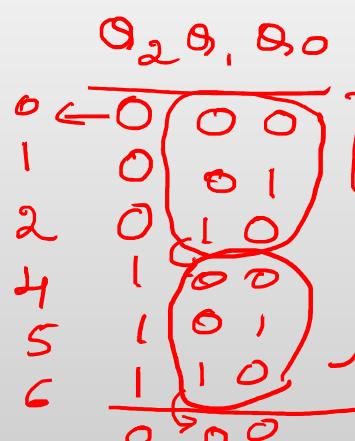
$$f_{\text{output}} = \frac{f_{\text{clk}}}{6}$$

$$\text{MOD } MN = \text{MOD } M \times \text{MOD } N$$



$$MN \neq 2^{\text{power}}$$

$$\text{MOD } 6 = \text{MOD } 3 \times \text{MOD } 2 \rightarrow 50\%$$

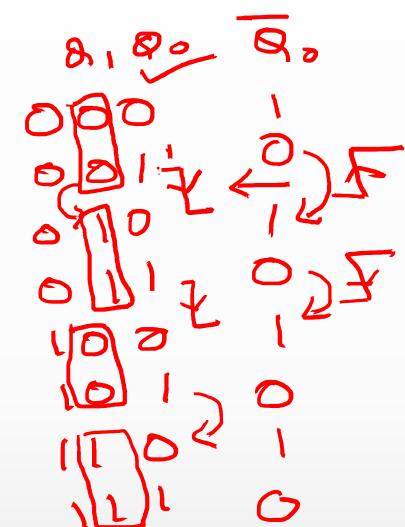
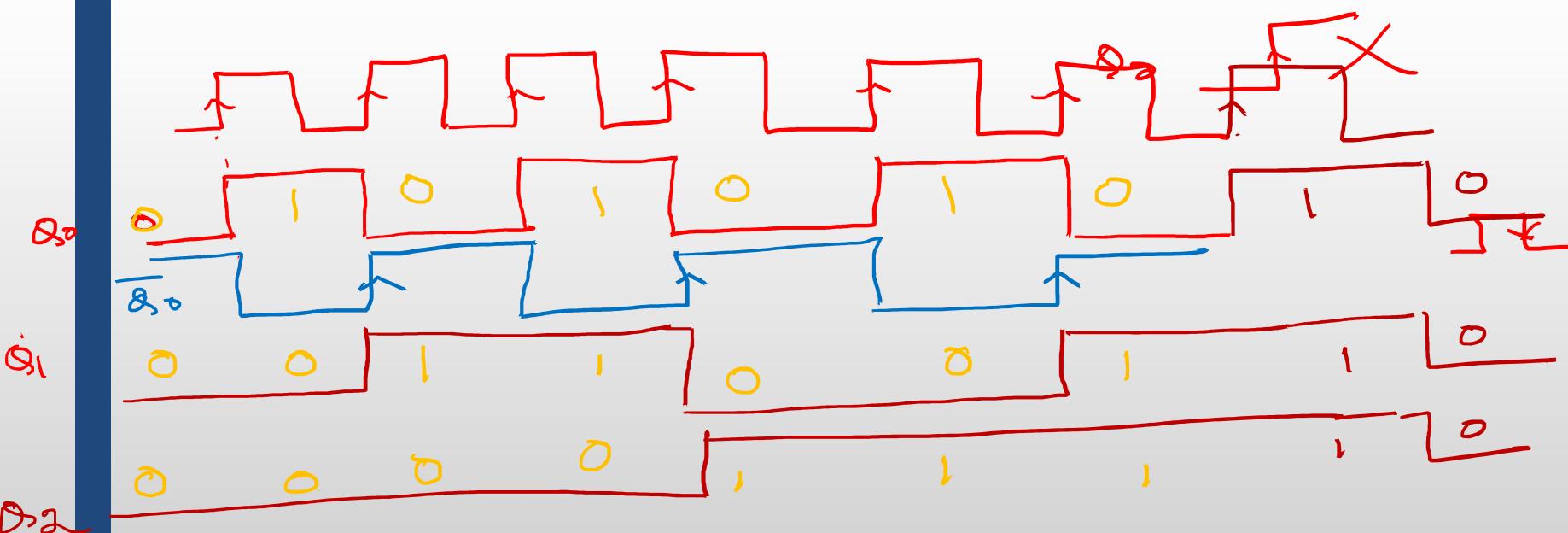
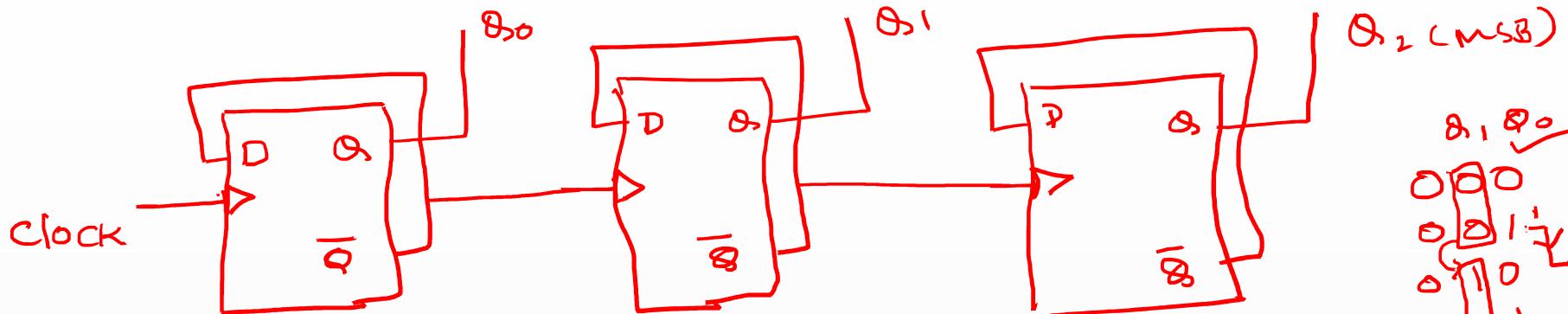


# ASYNCHRONOUS COUNTER(RIPPLE COUNTER) CONTD..

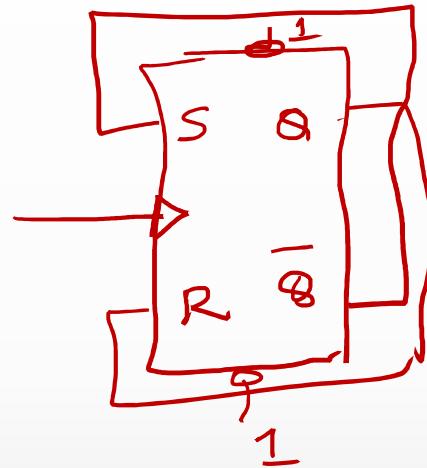
# Counters:

- Register that goes through prescribed sequence of states upon the application of input pulses is called a counter.
- There are 2 types of counters:
  - *Asynchronous counters (Ripple counters)* : Clock inputs are triggered by transitions of other flipflop.
  - *Synchronous counters* : The clock inputs of all flip flops receive common clock.

- Design a 3 – bit Asynchronous UP counter using positive edge triggered D flip flops.



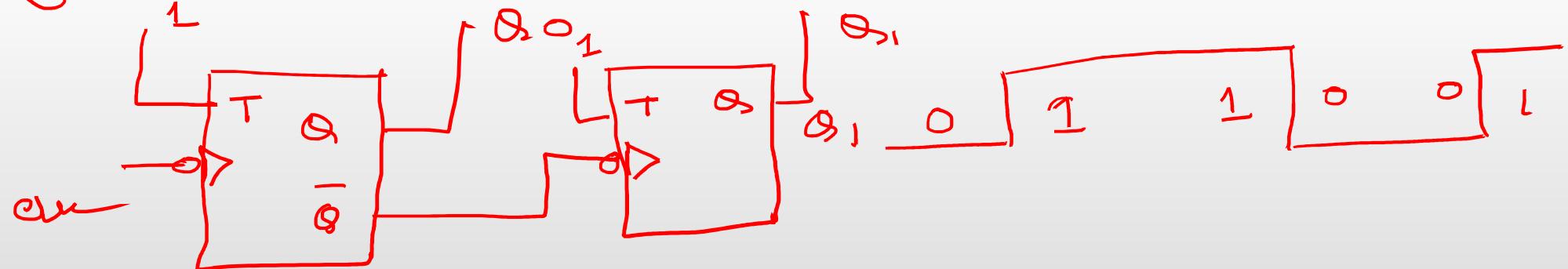
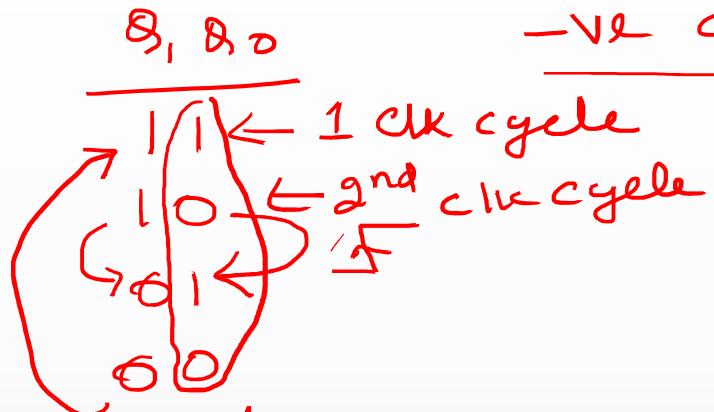
- Design a 4 – bit ( MOD 16/ Divide by 16) Asynchronous UP counter using negative edge triggered SR flip flops.
- Design a 4 – bit Asynchronous UP counter using positive edge triggered SR flip flops.



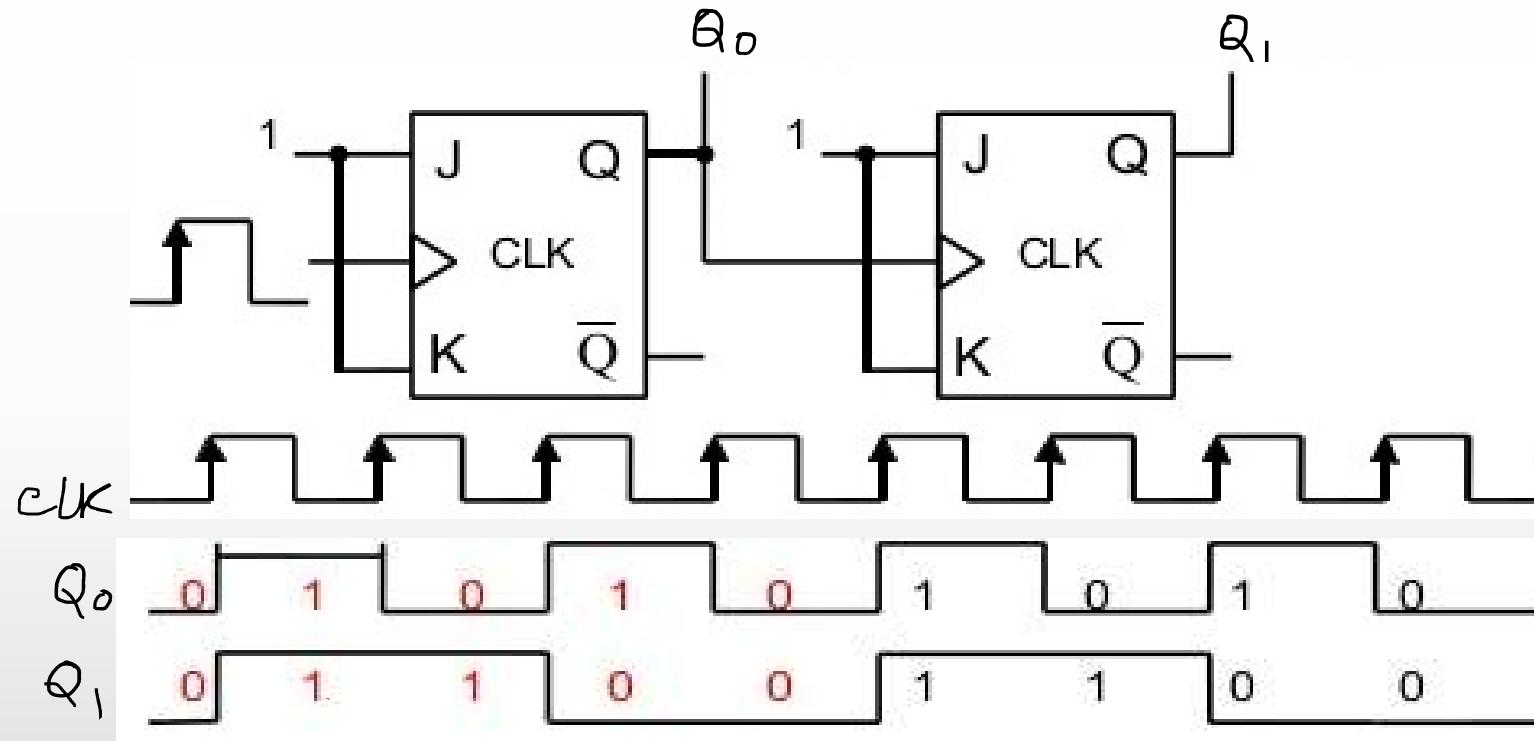
# **DOWN COUNTERS**

MOD4 down  $\Rightarrow$  2-bit

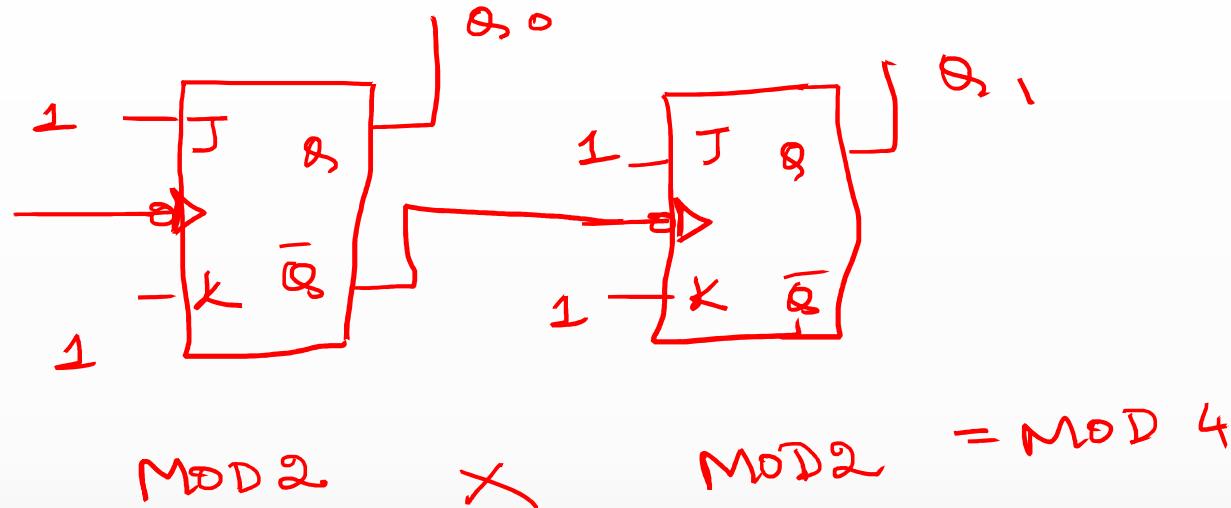
-ve edge triggered (JK, T, SR, D)



- Design a 2 – bit Asynchronous DOWN counter using positive edge triggered JK flip flops.



- Design a 2 – bit Asynchronous DOWN counter using negative edge triggered JK flip flops.



- Design a 2 – bit Asynchronous DOWN counter using negative edge triggered T flip flops.

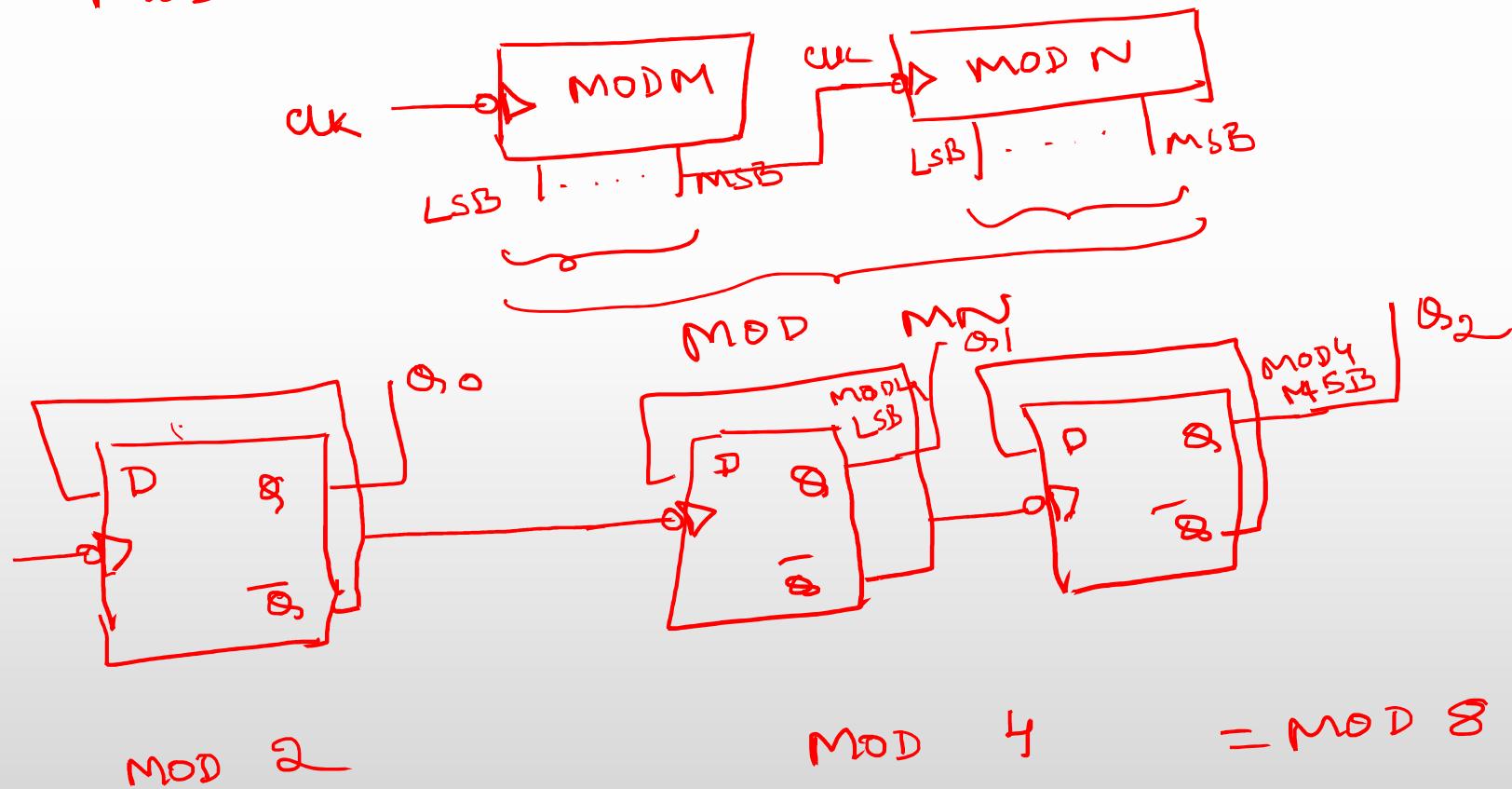
- Design a 2 – bit Asynchronous DOWN counter using positive edge triggered T flip flops.

Draw the circuit

- Design a 3 – bit Asynchronous DOWN counter using negative edge triggered D flip flops.

$$\text{MOD } 8 \Rightarrow \text{MOD } 4 \times \text{MOD } 2 = \underline{\text{MOD } 2 \times \text{MOD } 4} \quad \checkmark$$

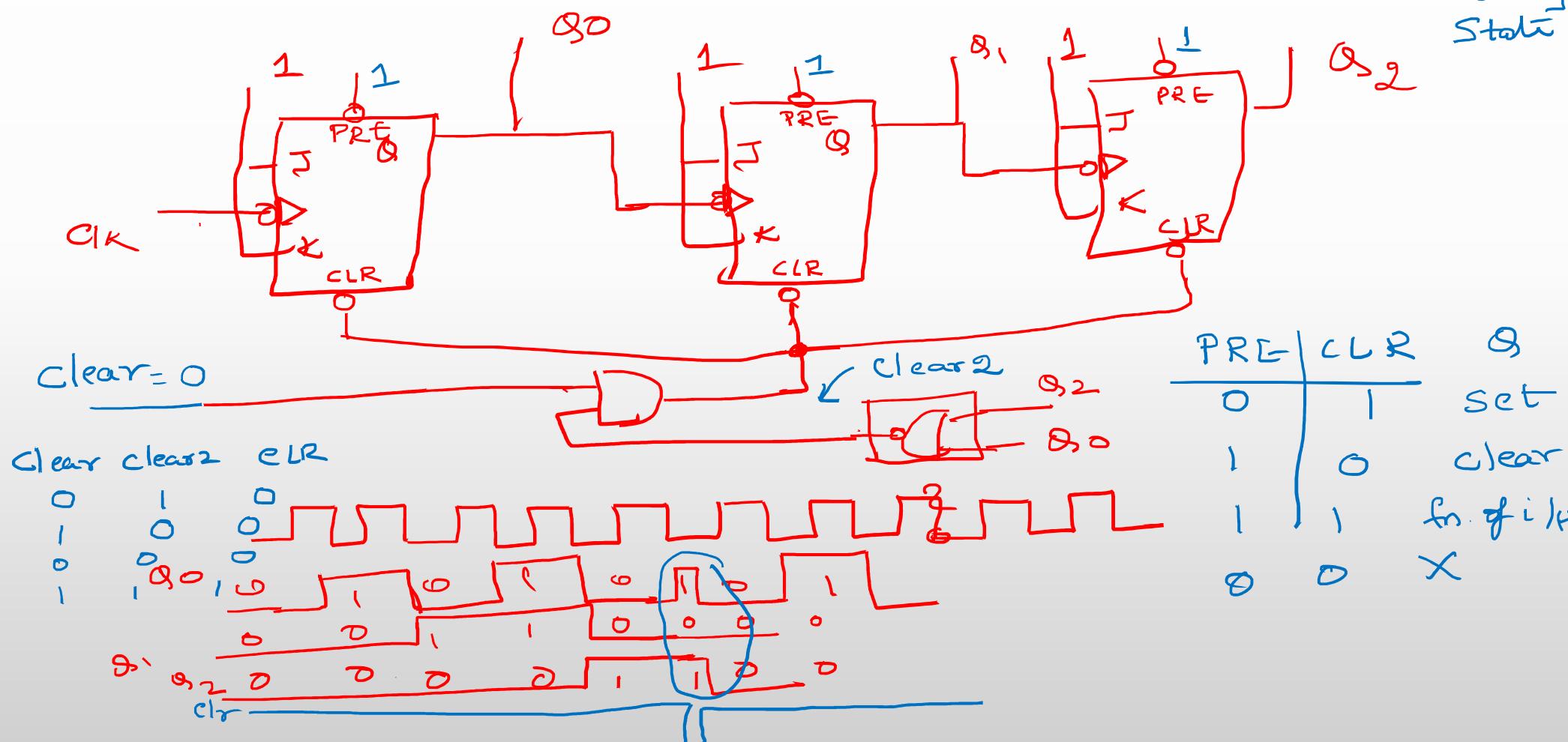
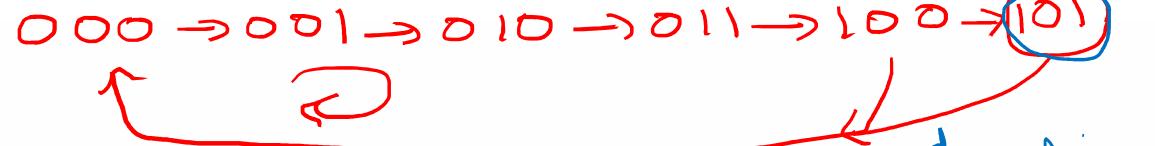
$$\text{MOD } MN \Rightarrow \text{MOD } M \times \text{MOD } N$$



- Design a 3 – bit Asynchronous DOWN counter using positive edge triggered D flip flops.

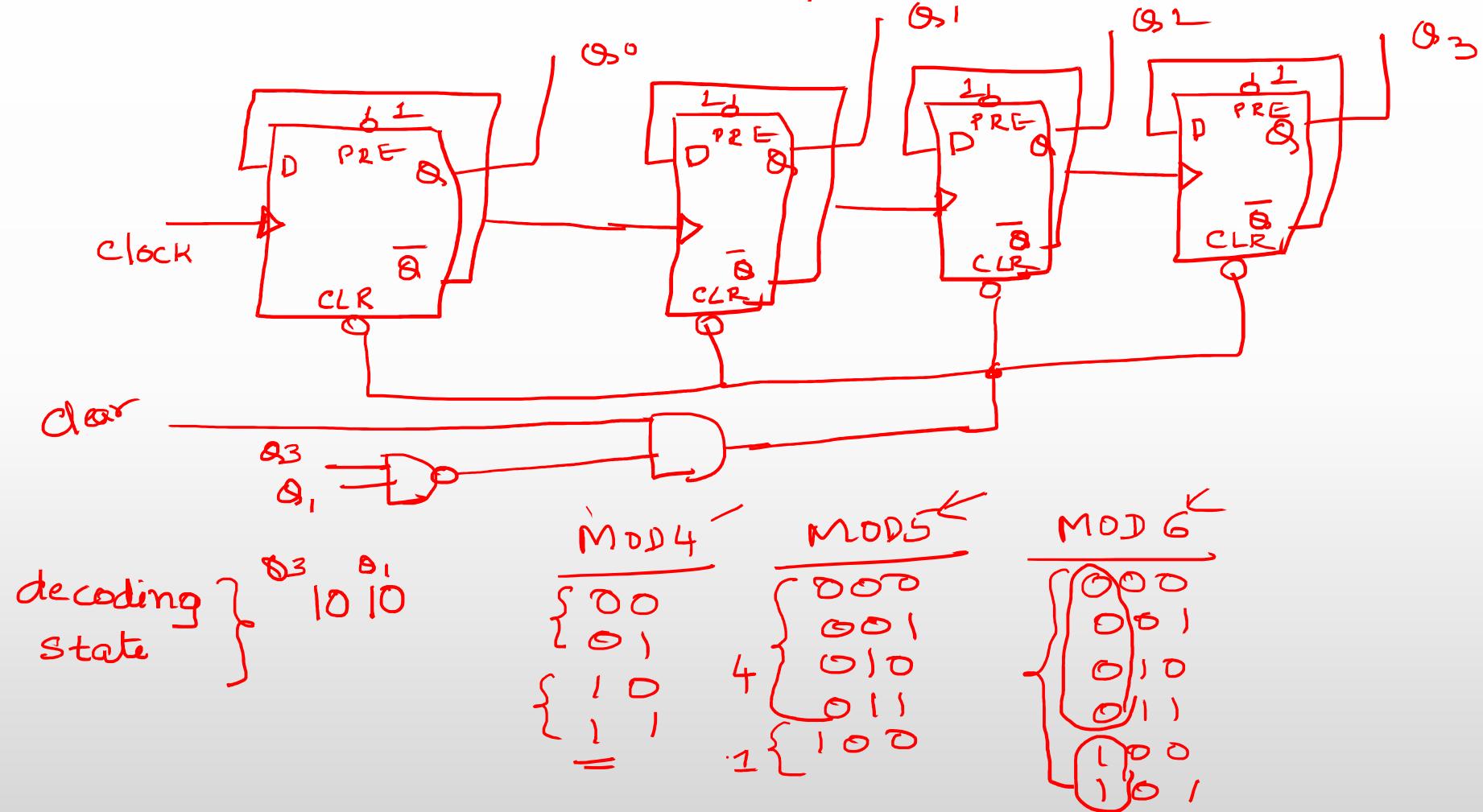
- Design a MOD 5 Asynchronous UP counter using negative edge triggered JK Flip Flops.

$0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 0$        $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 0$



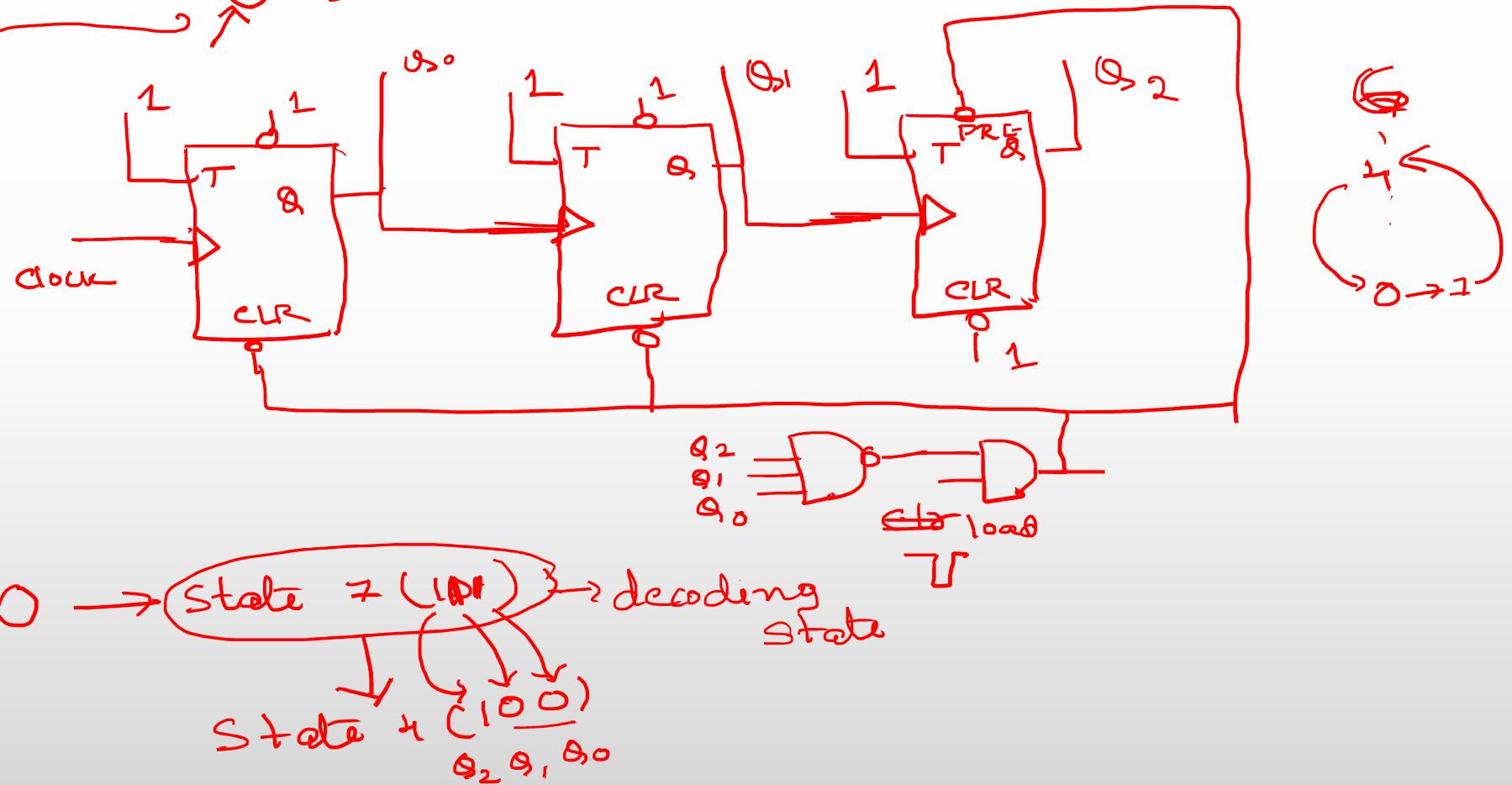
- Design a MOD 10 Asynchronous UP counter using positive edge triggered D Flip Flops.

$0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 8 \rightarrow 9 \rightarrow 0$       *Decade / Decimal*

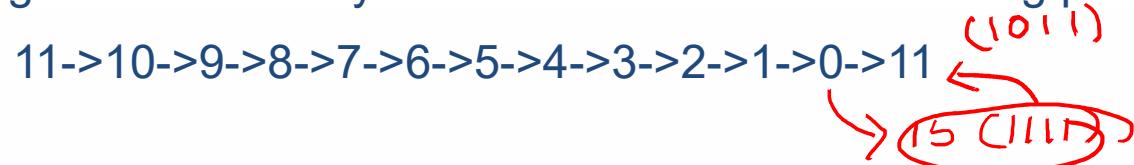


Design a MOD 5 Asynchronous DOWN counter using positive edge triggered T Flip Flops.

4->3->2->1->0->4  
 $4 \rightarrow 3 \rightarrow 2 \rightarrow 1 \rightarrow 0 \rightarrow 4 \rightarrow 3-2-1-0 \rightarrow \dots$



- Design a MOD 12 Asynchronous DOWN counter using positive edge triggered T Flip Flops.

11->10->9->8->7->6->5->4->3->2->1->0->11  
  
15 (1111)

- Design a MOD 18 Asynchronous DOWN counter using positive edge triggered JK Flip Flops.

- Design a 3 bit Asynchronous UP/DOWN counter using negative edge triggered JK Flip Flops.

Design a counter to obtain a 1KHz clock signal from a 10KHz clock signal with 50% duty cycle using negative edge triggered JK Flip Flops.

UP counter

Square

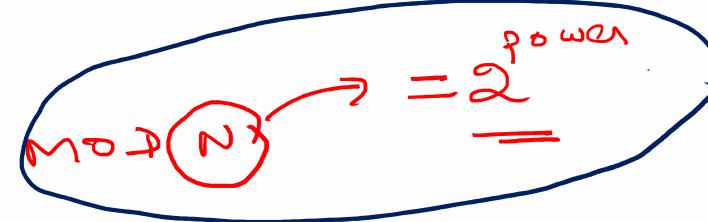
$$\text{MOD } 10 = \text{MOD } 5 \times \underbrace{\text{MOD } 2}_{\text{4 JFs}},$$

Dec. count	$Q_3 Q_2 Q_1 Q_0$	Clock cycle
0	0 0 0 0	-1
1	0 0 0 1	-2
2	0 0 1 0	-3
3	0 0 1 1	-4
4	0 1 0 0	-5
5	0 0 0 0	-6
6	0 0 0 1	-7
7	0 0 1 0	-8
8	0 0 1 1	-9
9	0 1 0 0	-10
10	0 1 0 1	0 0 0
11	0 1 1 0	0 0 1
12	1 0 0 0	:

Design a MOD 6 Asynchronous counter with 50% duty cycle using negative edge triggered D Flip Flops.

$$f_{\text{output}} = \frac{f_{\text{clk}}}{6}$$

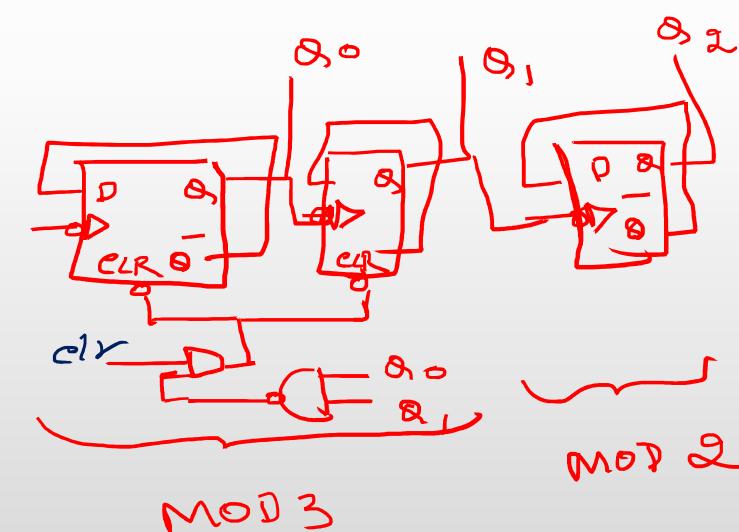
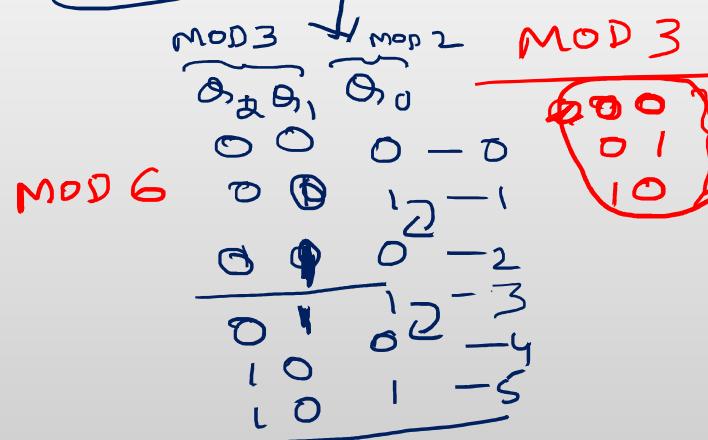
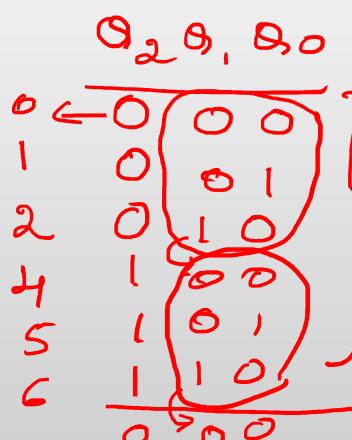
$$\text{MOD } MN = \text{MOD } M \times \text{MOD } N$$



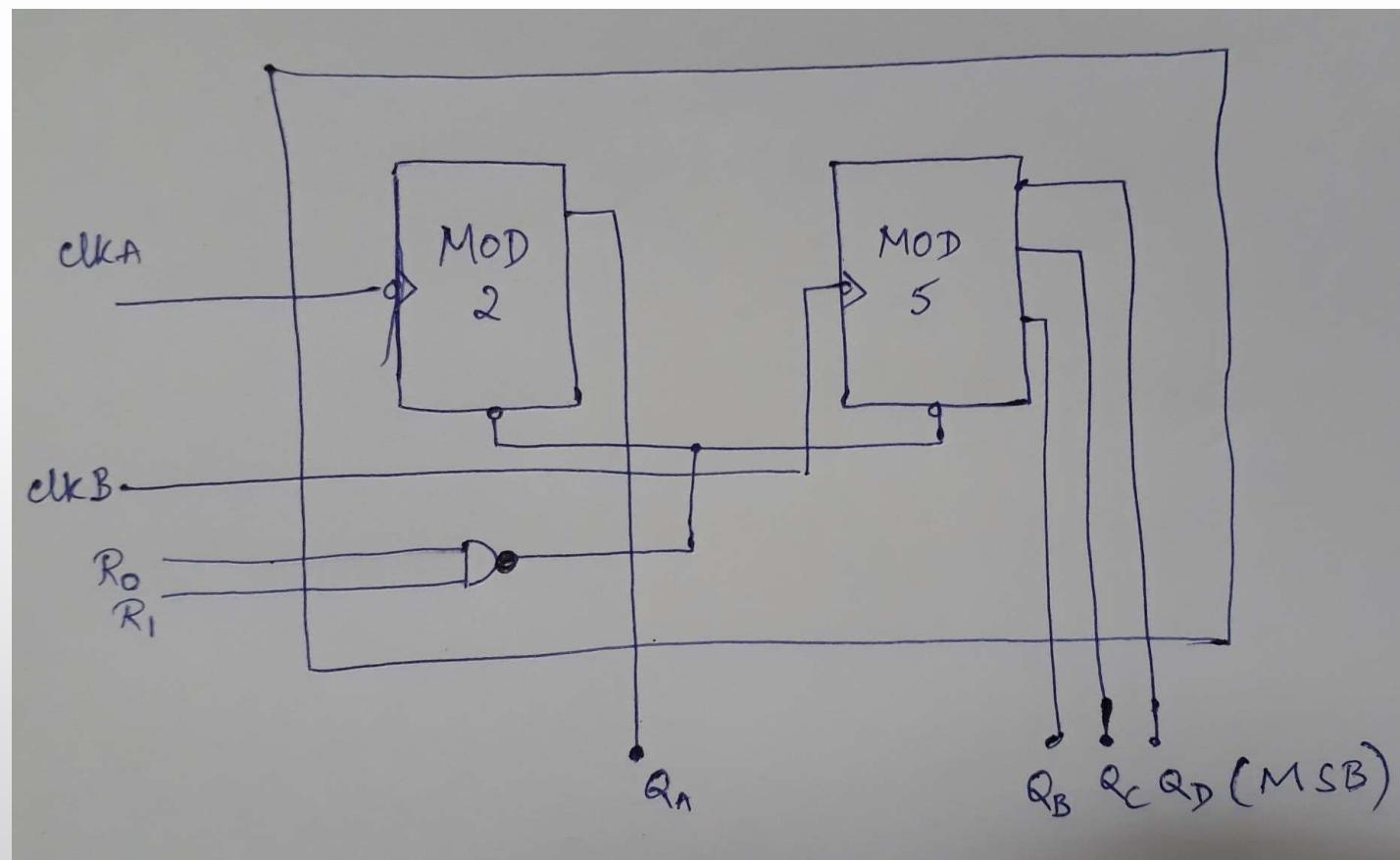
$$MN \neq 2^{\text{power}}$$

$$\text{MOD } 6 = \text{MOD } 3 \times \text{MOD } 2 \rightarrow 50\%$$

$$\text{MOD } 2 \times \text{MOD } 3$$



# 7490 IC (MOD 10 Asynchronous UP Counter)



CKB	1	14	CKA
R0(1)	2	13	NC
R0(2)	3	12	QA
NC	4	11	QD
VCC	5	10	GND
R9(1)	6	9	QB
R9(2)	7	8	QC

Note: Connect pin number 6 & 7 to GND

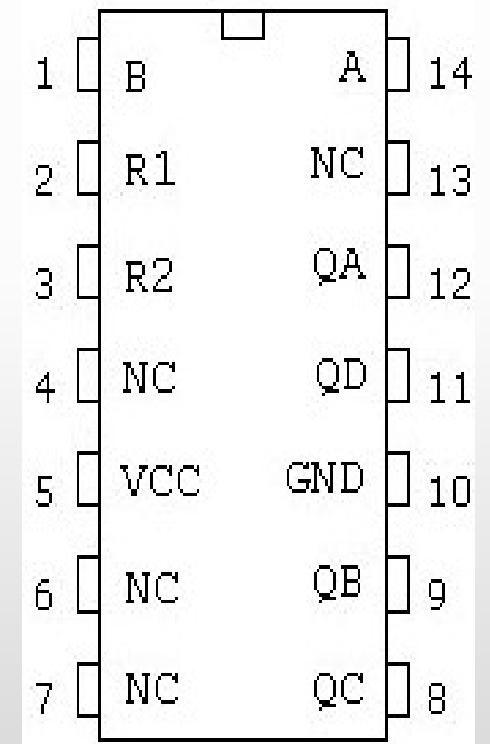
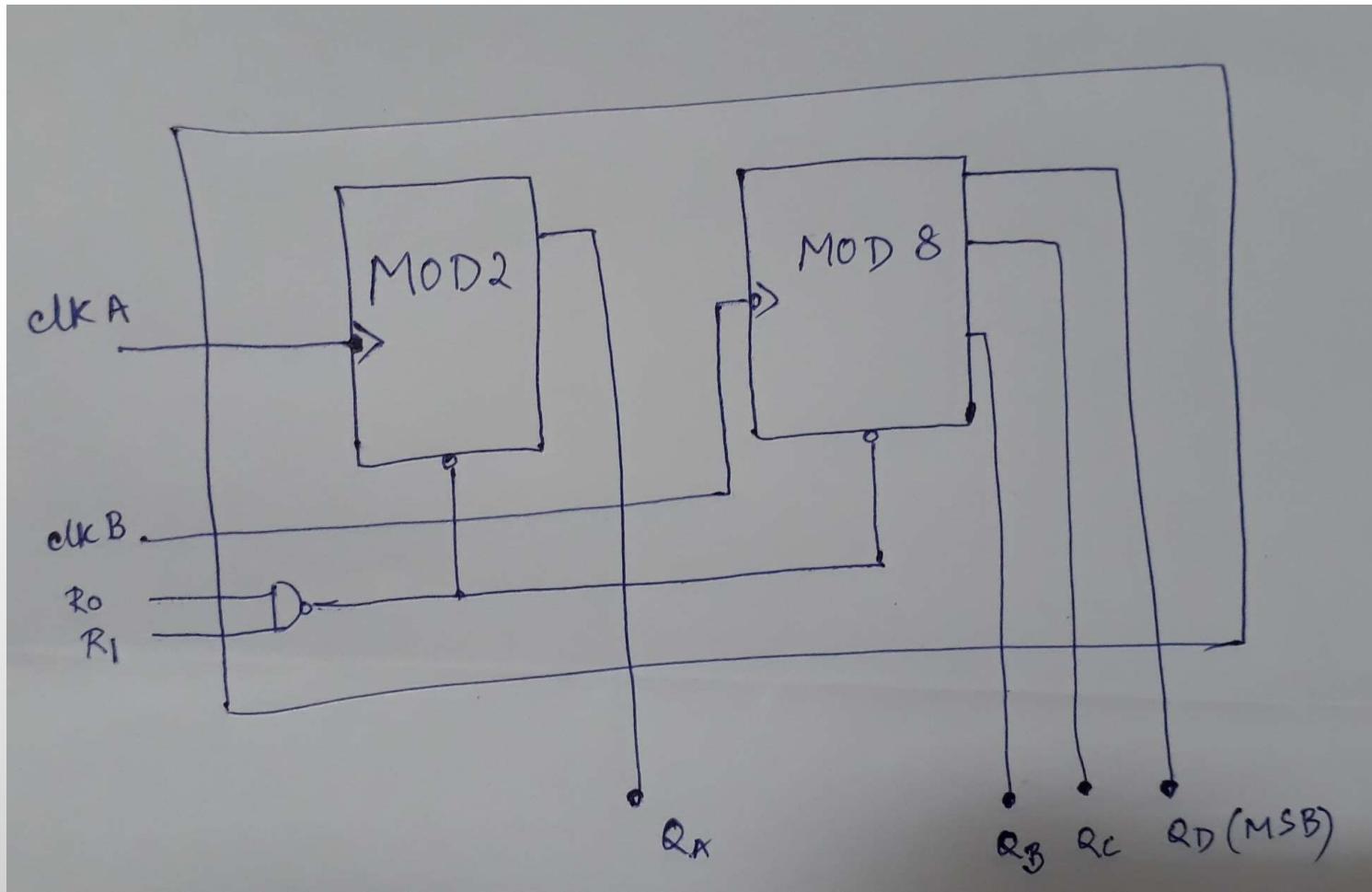
- Design MOD 10 counter using 7490IC

- Design MOD 6 counter using 7490IC and external gate

- Design a counter that performs UP count from 00 to 99 using 7490ICs

- Design a counter to perform a count from 00 to 24 using 7490ICs and external gate if required.

# 7493 IC (MOD 16 Asynchronous UP Counter)



- Design MOD 16 counter using 7493IC

- Design MOD 12 counter using 7493 IC and external gate if required

- Design a counter that performs UP count from 00H to FFH using 7493ICs

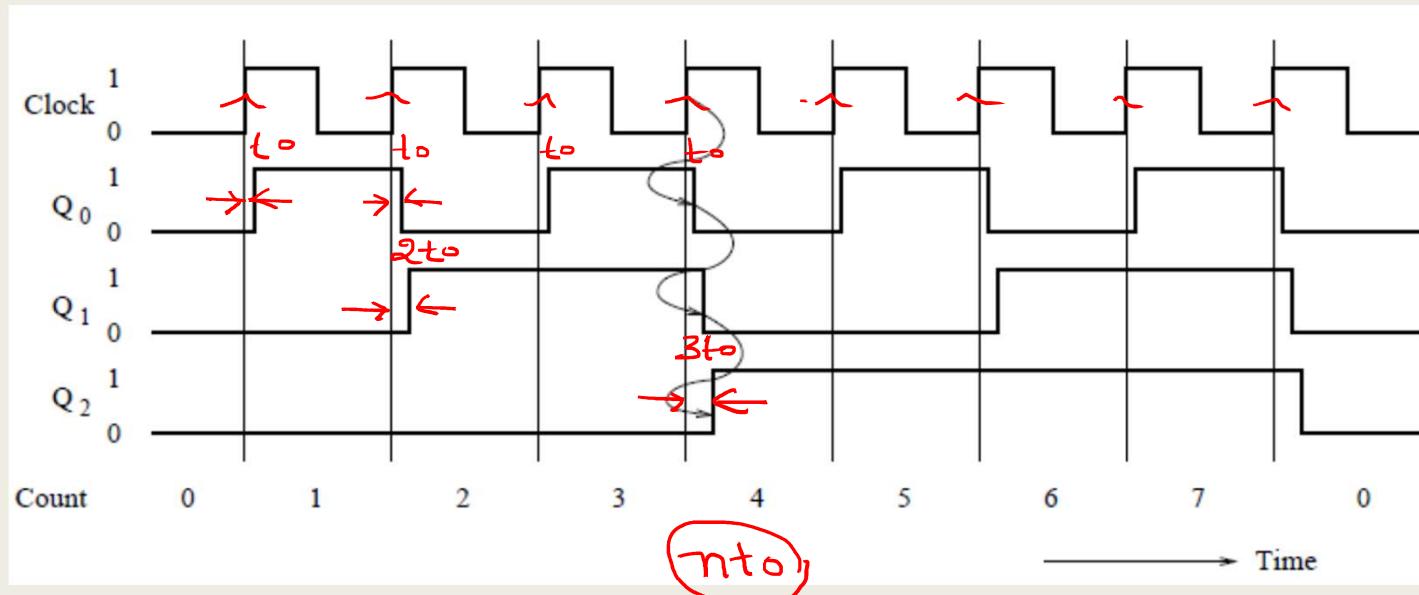
- Design a counter to perform a count from 00 to 65H using 7493ICs and external gate if required.

# Synchronous counters

- Limitations of asynchronous counters
- Synchronous counter design
- Synchronous counter ICs

# Limitation of asynchronous counters

RIPPLE ~~asynchronous~~ counters



- Limitations: 1. Not suitable for high frequency applications ✓  
2. Not suitable for higher mod counters

==

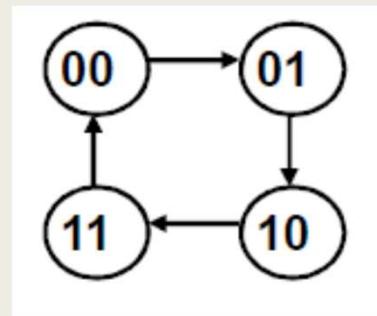
# Synchronous counter design

- All the flip flops are clocked simultaneously using same clock.
- Suitable for high frequency applications
- Designed using sequential circuit design process which can be used for any synchronous circuit designs

# 1. Design 2-bit synchronous binary UP counter using T ffs

- Counter states:  $00 \rightarrow 01 \rightarrow 10 \rightarrow 11$

State diagram



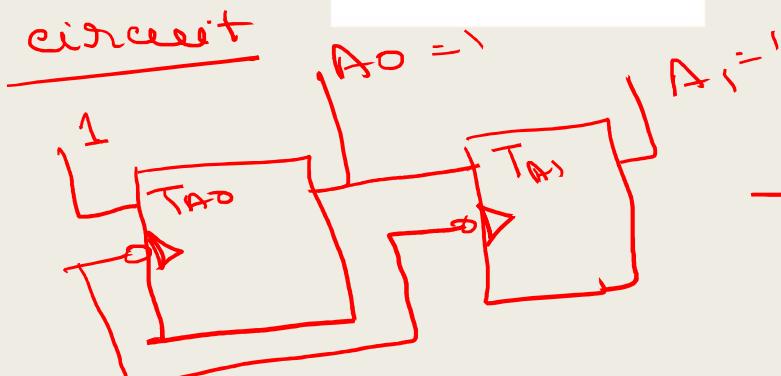
State table

Present state		Next state		Flip-flop inputs	
$A_1$	$A_0$	$A_1^+$	$A_0^+$	$TA_1$	$TA_0$
0	0	0	1	0	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	0	0	1	1

ff excitation

$$TA_1 = A_0$$

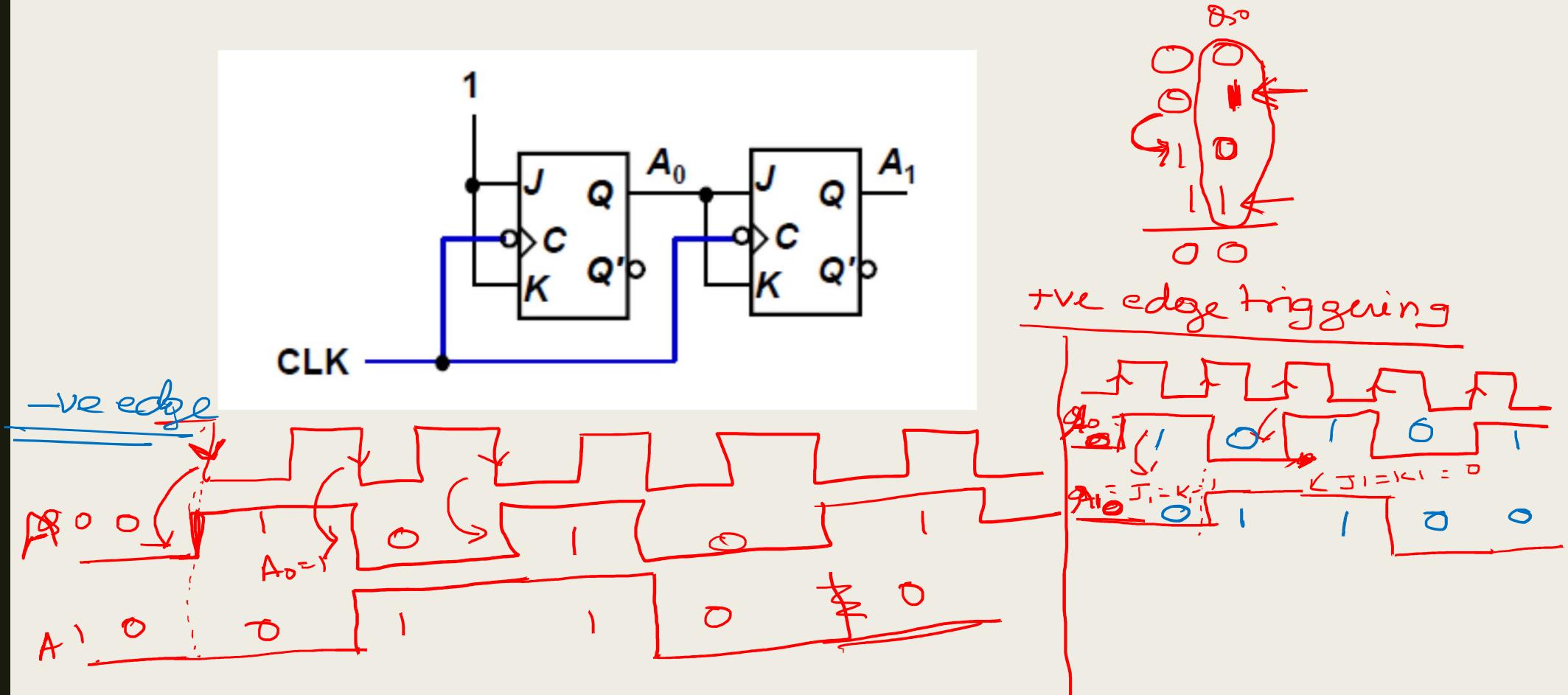
$$TA_0 = 1$$



$$TA_1(A_1, A_0) = A_0$$

$$TA_0 = 1$$

## 2-bit synchronous binary UP counter using JK ffs : circuit diagram



2. Design 2-bit synchronous binary UP counter using JK ffs

## 3-bit synchronous binary UP counter using JK ffs contd..



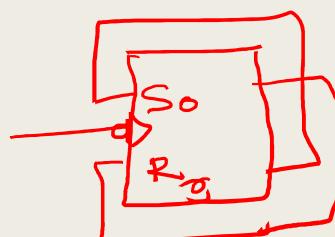
$$\left. \begin{array}{l} J_0 = K_0 = 1 \\ J_1 = K_1 = Q_0 \\ J_2 = K_2 = Q_1, Q_0 \end{array} \right\}$$

Solve & Verify

### 3. Design 3-bit synchronous binary UP counter (MOD 8) using SR ffs

$000 \rightarrow 001 \rightarrow 010 \rightarrow 011 \rightarrow 100 \rightarrow 101 \rightarrow 110 \rightarrow 111$

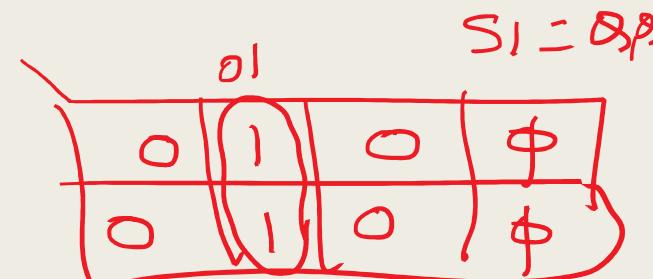
Present State	Next State
$Q_2 Q_1 Q_0$	$Q_2^+ Q_1^+ Q_0^+$
$000$	$001$
$001$	$010$
$010$	$011$
$011$	$100$
$100$	$101$
$101$	$110$
$110$	$111$
$111$	$000$



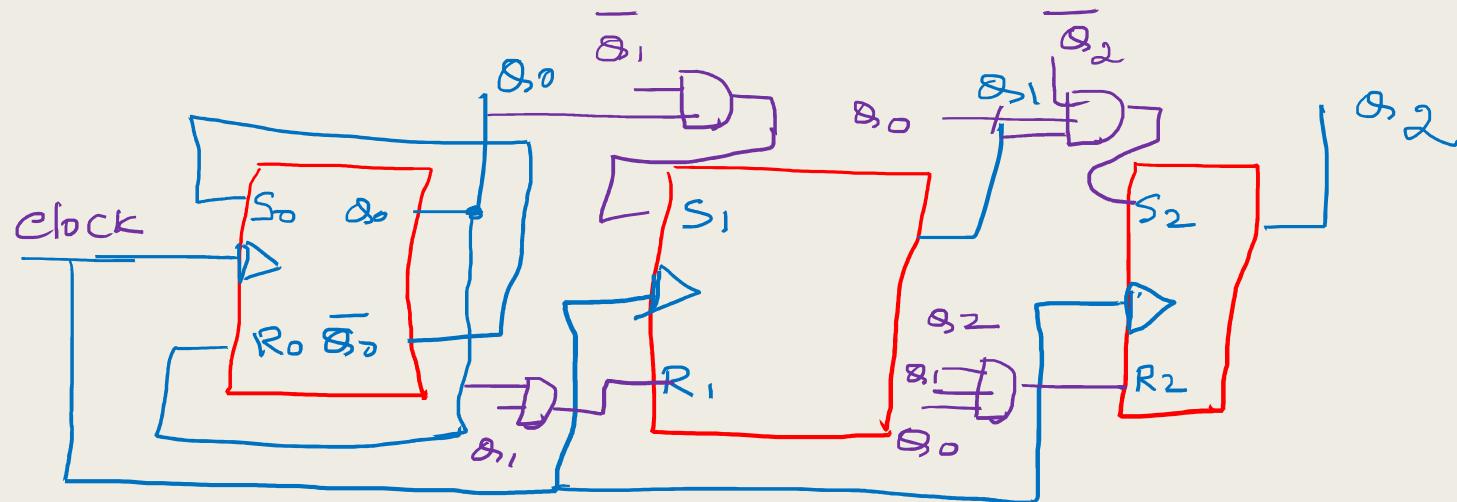
$S, R$	$S, R$
$0, 0$	$0, \phi$
$0, 1$	$1, 0$
$1, 0$	$0, 1$
$\phi, \phi$	$\phi, 0$

$S=0, R=1 / S=0, R=0 \rightarrow$  Reset / No change  
 $S=1, R=0 / S=0, R=0 \rightarrow$  Set / No change

$$\begin{aligned}
 S_2 &= \overline{Q_2} Q_1 Q_0 \\
 R_2 &= Q_2 Q_1 Q_0 \\
 S_1 &= \overline{Q}_1 Q_0 - \\
 R_1 &= Q_1 Q_0 - \\
 S_0 &= \overline{Q}_0 - \\
 R_0 &= Q_0 -
 \end{aligned}$$



## 3-bit synchronous binary UP counter (MOD 8) using SR ffs



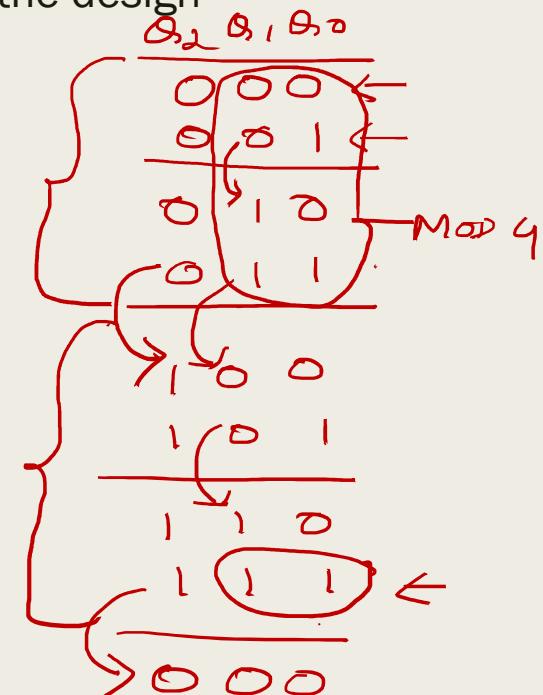
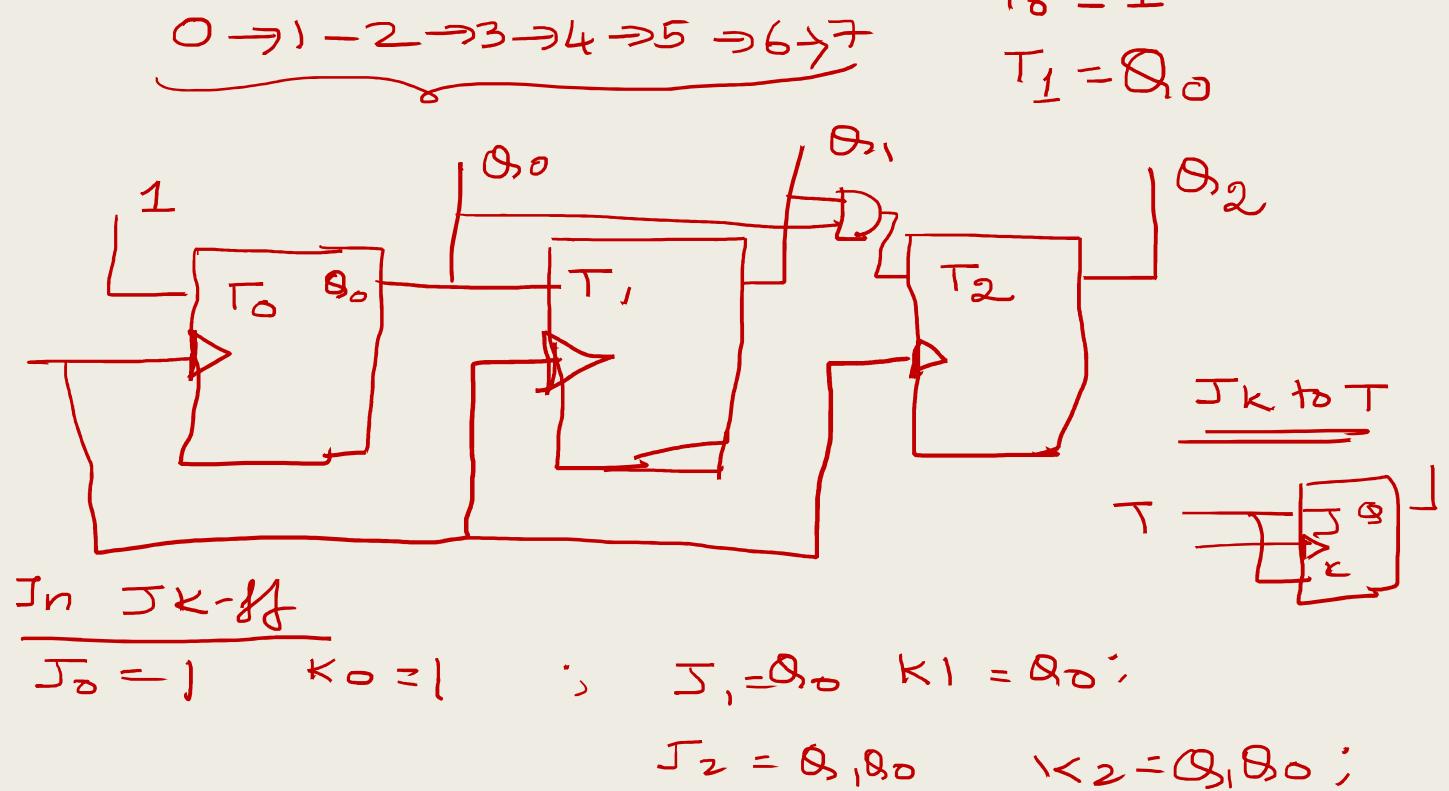
$$S_0 = \overline{Q_1} \quad R_0 = Q_0$$

$$S_1 = \overline{Q_2} \overline{Q_1} \quad R_1 = Q_1 \overline{Q_0}$$

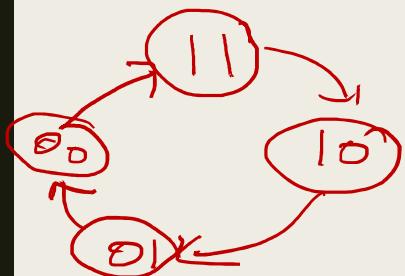
$$S_2 = \overline{Q_2} \overline{Q_1} \overline{Q_0} \quad R_2 = \overline{Q_2} \overline{Q_1} \overline{Q_0}$$

#### 4. Draw the circuit of 3-bit synchronous binary up counter (MOD 8) using T ffs

- Analyse the previous examples and draw the circuit directly without the design steps.



## 5. Design 2-bit synchronous binary down counter (MOD 4) using D ffs



State diagram

State table

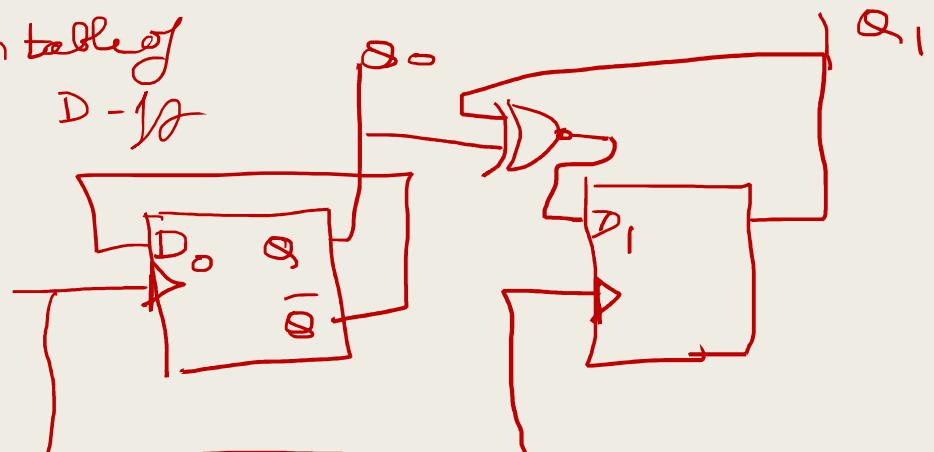
Present state $Q_1, Q_0$	Next state $Q'_1, Q'_0$		$D_1, D_0$
	$Q'_1$	$Q'_0$	
0 0	1	1	1 1
0 1	0	0	0 0
1 0	0	1	0 1
1 1	1	0	1 0

Excitation table of D - 12

$$D_1(Q_1, Q_0) = \overline{(Q_1 + Q_0)}$$

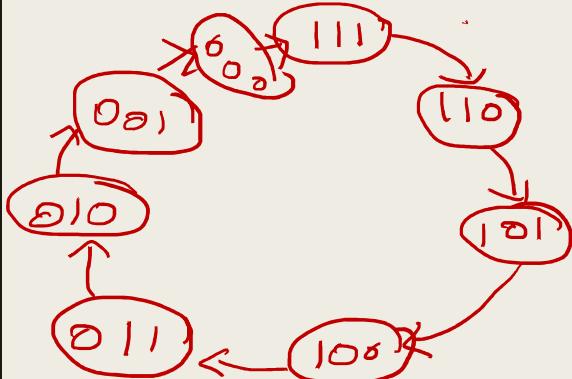
$$D_0(Q_1, Q_0) = \overline{Q_0}$$

$$Q_1, Q_0 = 10 \Rightarrow Q_1$$



**2-bit synchronous binary down counter (MOD 4) using D ffs**

## 6. Design 3-bit synchronous binary down counter (MOD 8) using D ffs



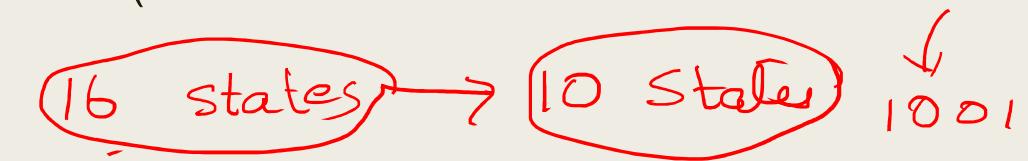
Present state			Next state			$S_2^+$	$S_1^+$	$S_0^+$
$Q_2$	$Q_1$	$Q_0$	$Q_2^+$	$Q_1^+$	$Q_0^+$	$D_2$	$D_1$	$D_0$
0	0	0	1	1	1	1	1	0
0	0	1	0	0	1	0	0	0
0	1	0	1	0	0	0	0	1
0	1	1	0	1	1	0	1	0
1	0	0	1	0	0	0	1	1
1	0	1	0	1	0	1	0	0
1	1	0	1	1	0	1	0	0

Inputs:

- $D_2 = \bar{S}_1 S_0$
- $D_1 = \bar{S}_2 \bar{S}_0 + S_2 S_0$   
 $= S_2 \bar{S}_0 S_0$
- $D_0 = \bar{S}_2 S_0$

## 7. Design decade (BCD) synchronous up counter (MOD 10) using T ffs $\rightarrow$ 4 ffs

- (0  $\rightarrow$  1  $\rightarrow$  2  $\rightarrow$  3  $\rightarrow$  4  $\rightarrow$  5  $\rightarrow$  6  $\rightarrow$  7  $\rightarrow$  8  $\rightarrow$  9  $\rightarrow$  0)



10 to 15  $\rightarrow$  Not required, unused

## Decade (BCD) synchronous up counter contd..

<u>Present state</u>	<u>Next-state</u>	
$Q_3 Q_2 Q_1 Q_0$	$Q_3^+ Q_2^+ Q_1^+ Q_0^+$	
0000	$\rightarrow 0001$	$T_3 T_2 T_1 T_0$
0001	$\rightarrow 0010$	0 0 0 1
0010	$\rightarrow 0011$	0 0 1 1
0011	$\rightarrow 0100$	0 1 1 1
0100	$\rightarrow 0101$	0 0 0 1
0101	$\rightarrow 0110$	0 0 1 1
0110	$\rightarrow 0111$	0 0 0 1
0111	$\rightarrow 1000$	1 1 1 1
1000	$\rightarrow 1001$	0 0 0 1
1001	$\rightarrow 0000$	1 0 0 1
1010	—xx xx	x x x x
1111	—xxx x	x x x x

$Q_3 + Q_2 +$	$Q_1 + Q_0 +$	T
0 0	0 0	0
0 0	0 1	1
1 0	1 0	1
1 1	1 1	0

$$T_0 = 1$$

$$T_1 = \frac{Q_1 Q_0}{Q_3 Q_2 Q_0}$$

1	1	1	
1	1	1	
d	d	d	d
0	1	1	d

$$T_1 = \Sigma m(1, 3, 5, 7) + d(10, 11, 12, 13, 14, 15)$$

$$= \overline{Q}_3 Q_0$$

$$T_2 = Q_1 Q_0 \Rightarrow (3, 7, \textcircled{11}, \textcircled{15})$$

$$T_3 = \frac{Q_0 Q_1 Q_2}{(7, 15)} + \frac{Q_0 \overline{Q}_2 Q_3}{(3, 13) \text{ or } (3, 11)}$$

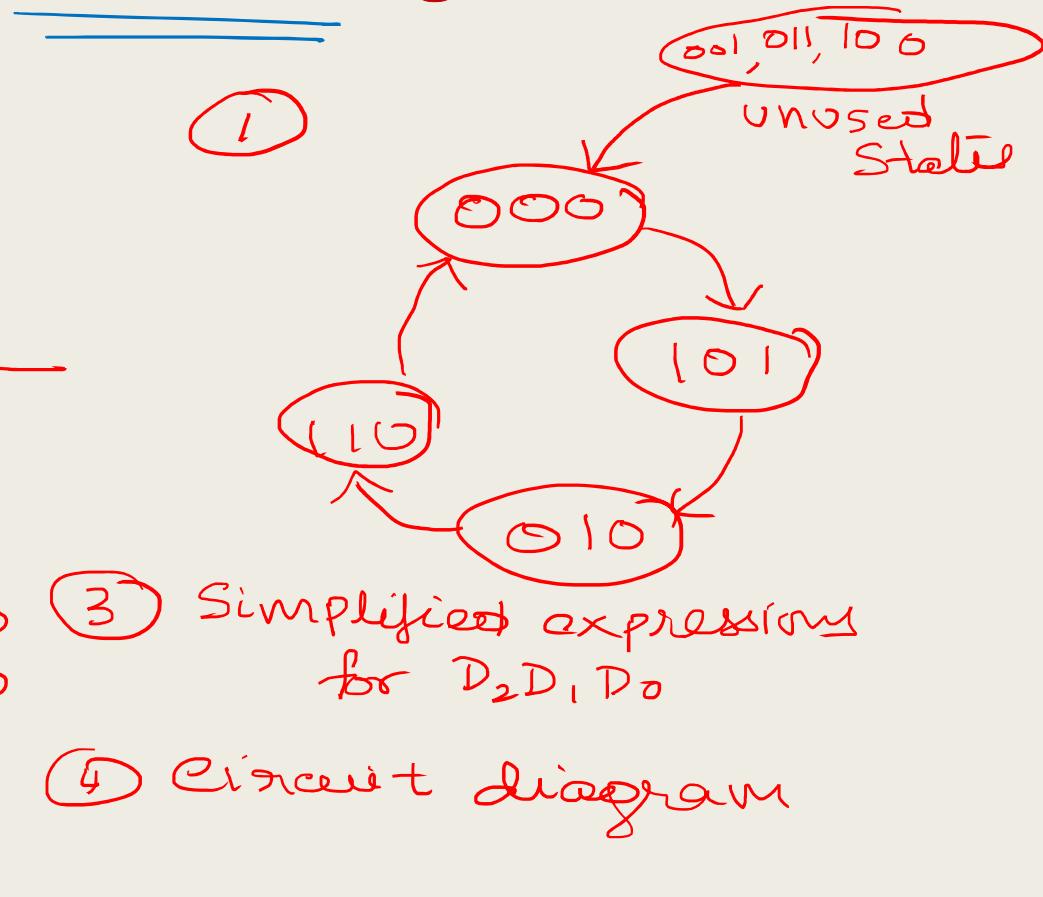
$$\begin{array}{|c|c|} \hline 0 & 0 \\ \hline 1 & 0 \\ \hline \end{array}$$

Design 3-bit synchronous counter to count according to the given sequence using D ffs. All the undefined/unused states should go to state 0.--Self correcting counters.

- $0 \rightarrow 5 \rightarrow 2 \rightarrow 6 \rightarrow 7 \rightarrow 0$

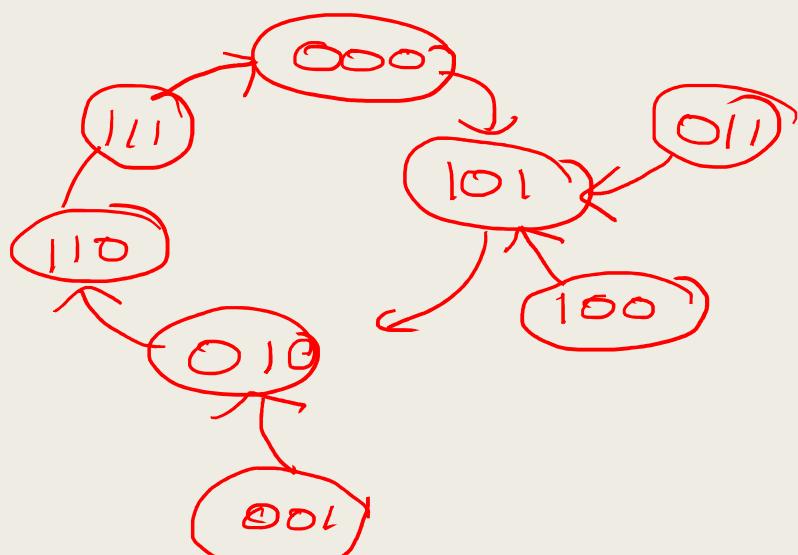
② Present States      Next State

$S_2 S_1 S_0$	$S_2 T_2 T_1 T_0$	$D_2 D_1 D_0$
0 0 0	1 0 1	1 0 1
0 0 1	0 0 0	0 0 0
0 1 0	1 1 0	1 1 0
0 1 1	0 0 0	0 0 0
1 0 0	0 0 0	0 0 0
1 0 1	0 0 0	0 1 0
1 1 0	1 1 1	1 1 1
1 1 1	0 0 0	0 0 0



Design 3-bit synchronous counter to count according to the given sequence using D ffs. All the undefined states should go to next valid/defined state.

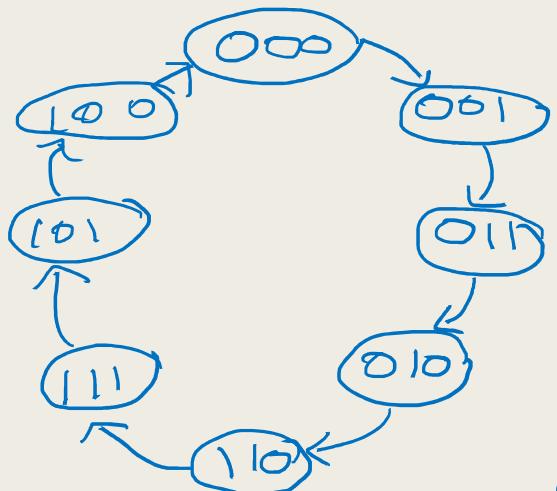
- $0 \rightarrow 5 \rightarrow 2 \rightarrow 6 \rightarrow 7 \rightarrow 0$



Complete Simplification & Circuit

Present state $Q_2 Q_1 Q_0$	Next state $Q_2^+ Q_1^+ Q_0^+$			$D_2 D_1 D_0$
	$Q_2^+$	$Q_1^+$	$Q_0^+$	
000	1	0	1	1 0 1
001	0	1	0	0 1 0
010	1	1	0	1 1 0
011	1	0	1	1 0 1
100	1	0	1	1 0 1
101	0	1	0	0 1 0
110	1	1	1	1 1 1
111	0	0	0	0 0 0

## 8. Design 3-bit gray code counter using JK ffs



Present st. $Q_2 Q_1 Q_0$	Next state $Q_2' Q_1' Q_0'$	J <sub>2</sub> K <sub>2</sub> J <sub>1</sub> K <sub>1</sub> J <sub>0</sub> K <sub>0</sub>					
		J <sub>2</sub>	K <sub>2</sub>	J <sub>1</sub>	K <sub>1</sub>	J <sub>0</sub>	K <sub>0</sub>
000	001	0	d	0	d	1	d
001	011	0	d	1	d	d	0
010	110	1	d	d	0	0	d
011	010	0	d	d	0	d	d
100	000	d	1	0	d	0	d
101	100	d	0	d	d	d	d
110	101	d	0	d	d	d	d
111	110	d	0	d	d	d	d

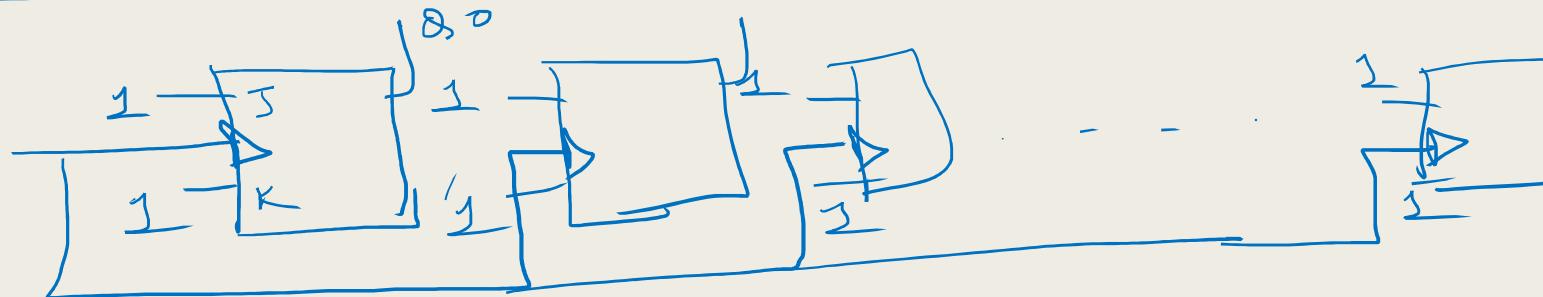
$Q_2 Q_1 Q_0$	J	K	Set/Reset
00	0	d	p
01	1	d	
10	d	1	
11	d	0	



$$\begin{aligned}
 J_2 &= \overline{Q}_2 Q_0 \quad (1, 3) \\
 K_2 &= \overline{Q}_1 \overline{Q}_0 \rightarrow Q_2 \oplus Q_1 \\
 J_1 &= \overline{Q}_2 Q_0 \quad (5, 7) \\
 K_1 &= Q_2 Q_0 \quad (5, 7) \\
 J_0 &= \overline{Q}_2 \overline{Q}_1 + Q_2 Q_1 \\
 K_0 &= \boxed{\begin{array}{cccc} d & 0 & 1 & d \\ d & 1 & 0 & d \\ 0 & 1 & 0 & d \\ d & 0 & d & 1 \end{array}} \\
 Q_1 \oplus Q_2 & = \underline{\quad} + \underline{\quad}
 \end{aligned}$$

## ~~3-bit gray code counter using JK ffs~~

Quiz of pre class

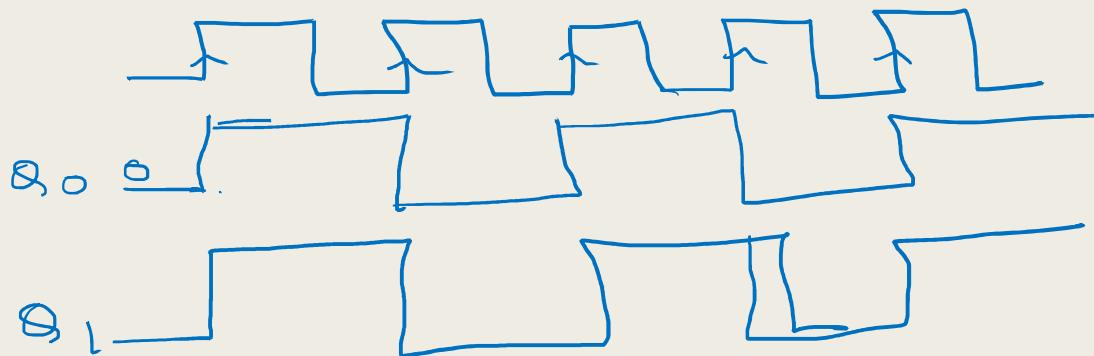


$\rightarrow 00 \dots 0$

$\rightarrow 111 \dots 1$

000 - - -

111 - - -



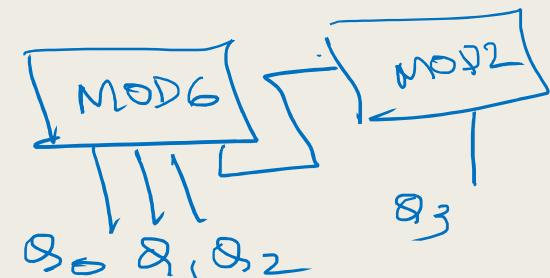
(2) (2)

MOD 12  $\Rightarrow$  MOD M  $\times$  MOD N

0	-	0000
1	-	0001
2	-	0010
3	-	0011
4	-	0100

1	000	-8	2
1	001	-9	
1	010	-10	
1	011	-11	
1	100	-12	

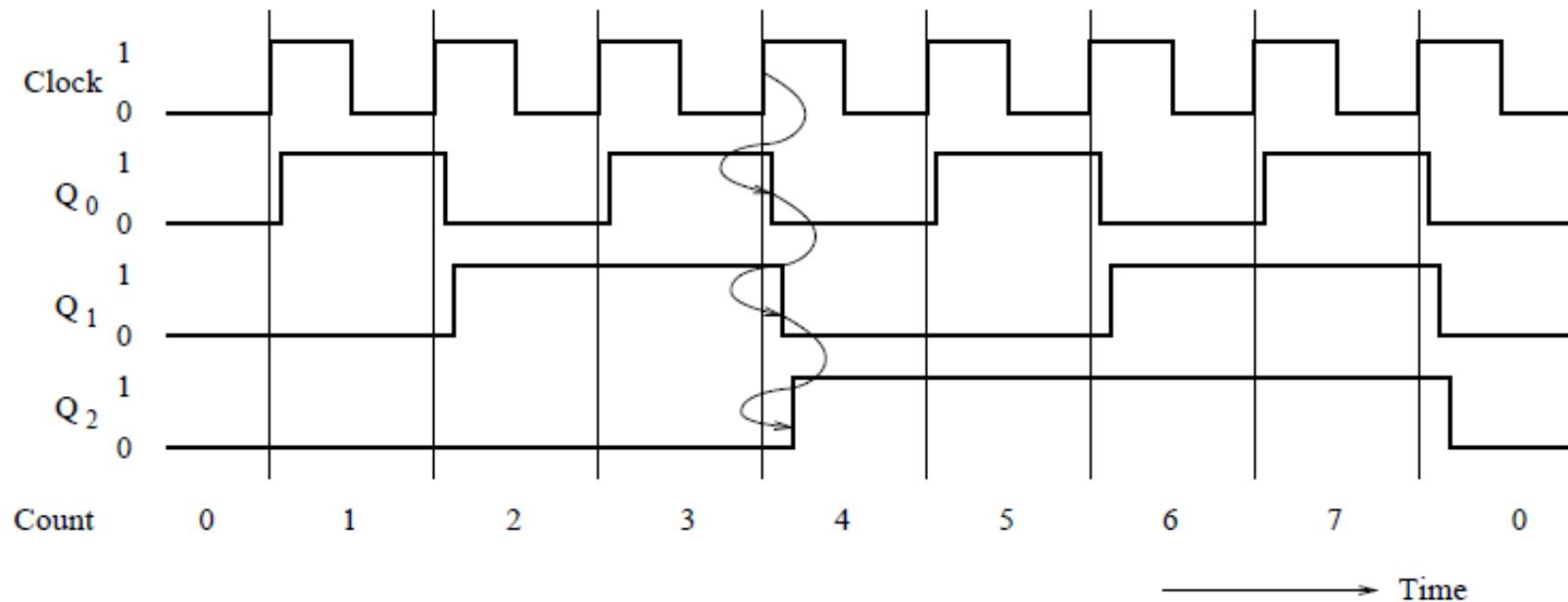
6 +  
14, 15



# Synchronous counters

- Limitations of asynchronous counters
- Synchronous counter design
- Synchronous counter ICs

# Limitation of asynchronous counters



Limitations:

1. Not suitable for high frequency applications
2. Not suitable for higher mod counters

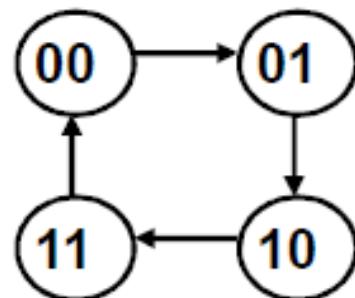
# Synchronous counter design

- All the flip flops are clocked simultaneously using same clock.
- Suitable for high frequency applications
- Designed using sequential circuit design process which can be used for any synchronous circuit designs

# 1. Design 2-bit synchronous binary UP counter using T ffs

- Counter states:  $00 \rightarrow 01 \rightarrow 10 \rightarrow 11$

State diagram



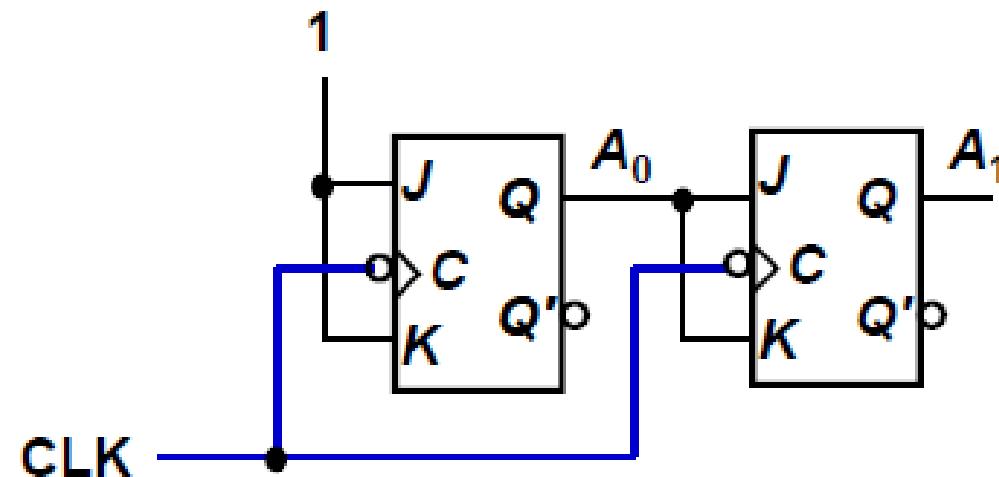
State table

Present state		Next state		Flip-flop inputs	
$A_1$	$A_0$	$A_1^+$	$A_0^+$	$TA_1$	$TA_0$
0	0	0	1	0	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	0	0	1	1

$$TA_1 = A_0$$

$$TA_0 = 1$$

## 2-bit synchronous binary UP counter using JK ffs : circuit diagram



2. Design 2-bit synchronous binary UP counter using JK ffs

2-bit synchronous binary UP counter using JK ffs contd..

3. Design 3-bit synchronous binary UP counter (MOD 8)using SR ffs

3-bit synchronous binary UP counter (MOD 8)using SR ffs

#### 4. Draw the circuit of 3-bit synchronous binary up counter (MOD 8) using T ffs

- Analyse the previous examples and draw the circuit directly without the design steps.

5. Design 2-bit synchronous binary down counter  
(MOD 4) using D ffs

# **2-bit synchronous binary down counter (MOD 4) using D ffs**

6. Design 3-bit synchronous binary down counter  
(MOD 4) using D ffs

## 7. Design decade (BCD) synchronous up counter (MOD 10)using T ffs

- (0→1→2→3→4→5→6→7→8→9→0)

# **Decade (BCD) synchronous up counter contd..**

# Synchronous counters

- Limitations of asynchronous counters
- Synchronous counter design
- Synchronous counter ICs

# Synchronous counter design

- All the flip flops are clocked simultaneously using same clock.
- Suitable for high frequency applications
- Designed using sequential circuit design process which can be used for any synchronous circuit designs

# 3-bit gray code counter using JK ffs

$T_2$	$Q_2$	$\bar{Q}_2$	$D_2$	$Q_1$	$\bar{Q}_1$	$D_1$	$Q_0$	$\bar{Q}_0$	$D_0$
0	0	1	0	0	1	0	0	1	0
1	1	0	1	1	0	1	1	0	1

$$J_2 = Q_1 \bar{Q}_0$$

$J_1$	$Q_2$	$\bar{Q}_2$	$D_2$	$Q_1$	$\bar{Q}_1$	$D_1$	$Q_0$	$\bar{Q}_0$	$D_0$
0	0	1	1	0	1	0	0	1	0
1	0	0	0	1	0	1	1	0	1

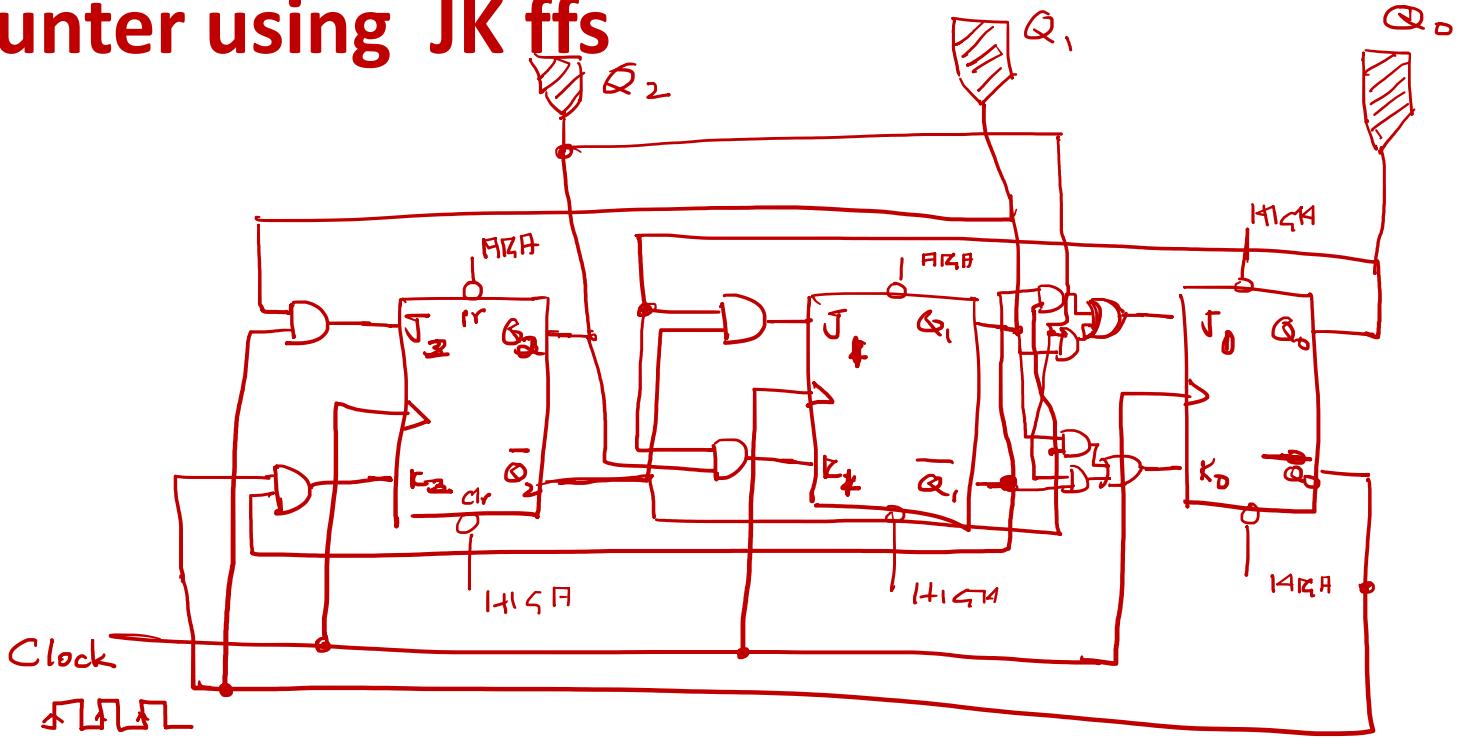
$$J_1 = \bar{Q}_2 Q_0$$

$J_0$	$Q_2$	$\bar{Q}_2$	$D_2$	$Q_1$	$\bar{Q}_1$	$D_1$	$Q_0$	$\bar{Q}_0$	$D_0$
1	X	X	0	0	1	X	0	1	0
0	X	X	1	1	0	X	1	0	1

$$J_0 = \bar{Q}_2 \bar{Q}_1 + Q_2 Q_1$$

$K_0$	$Q_2$	$\bar{Q}_2$	$D_2$	$Q_1$	$\bar{Q}_1$	$D_1$	$Q_0$	$\bar{Q}_0$	$D_0$
X	0	1	1	0	1	0	0	1	0
X	1	0	0	1	0	1	1	0	1

$$K_0 = \bar{Q}_2 Q_1 + Q_2 \bar{Q}_1$$



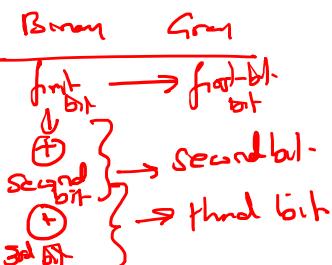
Tutorial or Work-out question for you

Design 3-bit Gray code down counter using JK flip-flop

## 8. Design 3-bit gray code counter using JK ffs

Next State Table

Decimal Number	Binary	3-bit Gray Code
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100
8	Cannot be represented using only 3 bits	
9		



1. up counting  
2. down counting

3-bit  $\rightarrow$  3 JK flipflops

msb      lsb

present state	$Q_2 Q_1 Q_0$	next state		
		$Q_2 Q_1 Q_0$	$J_2 K_2$	$J_1 K_1$

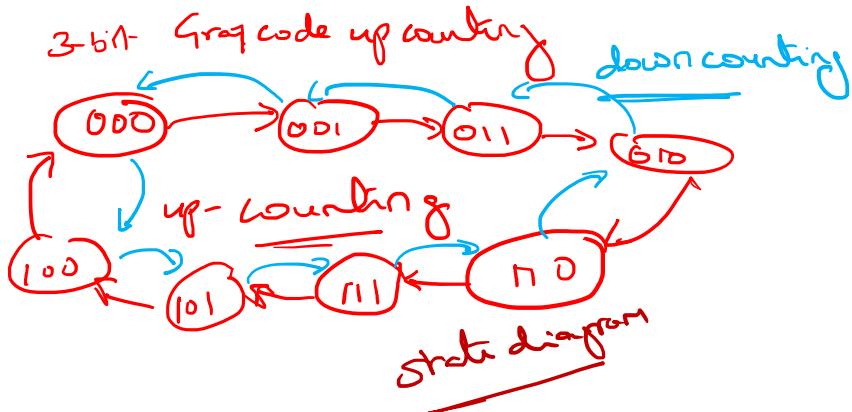
$m_0$	0 0 0	0 0 1	0 X	0 X	1 X ✓
$m_1$	0 0 1	0 1 1	0 X	1 X	X 0 ✓
$m_3$	0 1 1	0 1 0	0 X	X 0	X 1 ✓
$m_2$	0 1 0	1 1 0	1 X	X 0	0 X ✓
$m_6$	1 1 0	1 1 1	X 0	X 0	1 X ✓
$m_7$	1 1 1	1 0 1	X 0	X 1	X 0 ✓
$m_5$	1 0 1	1 0 0	X 0	0 X	X 1 ✓
$m_4$	1 0 0	0 0 0	X 1	0 X	0 X ✓

From function table of JK

pre	<u>S</u>	<u>K</u>	from function table of JK
0 $\rightarrow$ 0	0	0	0 0 0
0 $\rightarrow$ 1	1	0	0 0 1
1 $\rightarrow$ 0	x	0	0 1 0
1 $\rightarrow$ 1	x	0	0 1 1

pre	<u>S</u>	<u>K</u>	from function table of JK
0 $\rightarrow$ 0	0	0	0 0 0
0 $\rightarrow$ 1	1	0	0 0 1
1 $\rightarrow$ 0	x	0	0 1 0
1 $\rightarrow$ 1	x	0	0 1 1

pre	<u>s</u>	<u>d</u>	<u>r</u>	<u>next state</u>	<u><math>J_2 K_2</math></u>	<u><math>J_1 K_1</math></u>	<u><math>J_0 K_0</math></u>
$m_0$	0 0 0	0 0 1	0	0 0 0	0 X	0 X	1 X
$m_1$	0 0 1	0 1 1	0	0 0 1	0 X	1 X	X 0
$m_2$	0 1 0	1 1 0	1	0 1 1	1 X	X 0	0 X
$m_3$	0 1 1	0 1 0	0	0 1 0	0 X	X 0	X 1
$m_4$	1 0 0	0 0 0	X	1 1 0	X 1	0 X	0 X
$m_5$	1 0 1	1 0 0	0	1 1 1	X 0	0 X	X 1
$m_6$	1 1 0	1 1 1	0	1 0 1	X 0	X 1	1 X
$m_7$	1 1 1	1 0 1	1	1 0 0	X 1	X 0	X 0



# Design 3-bit synchronous counter to count according to the given sequence using D ffs.

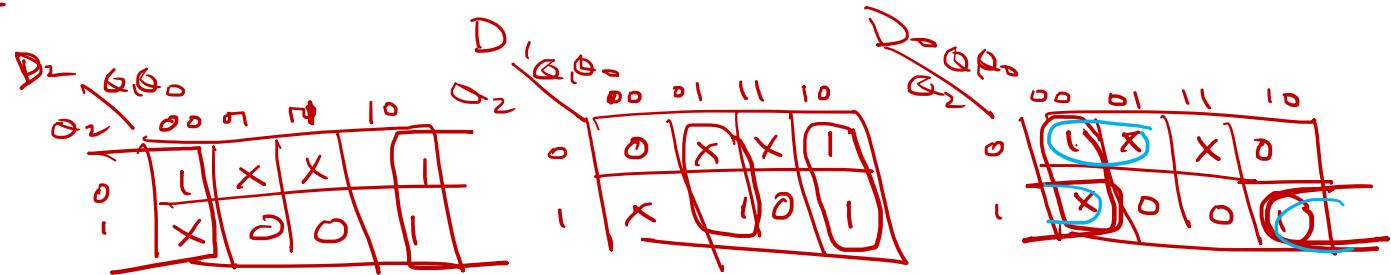
used states & counting

- $0 \rightarrow 5 \rightarrow 2 \rightarrow 6 \rightarrow 7 \rightarrow 0$

unused states

states: 1, 3, 4,

PS	Q1S	D <sub>2</sub> D <sub>1</sub> D <sub>0</sub>
Q <sub>2</sub> Q <sub>1</sub> Q <sub>0</sub>	Q <sub>2</sub> Q <sub>0</sub>	
m <sub>0</sub> 0 0 0	1 0 1	1 0 1
m <sub>1</sub> 0 0 1	X X X	X X X
m <sub>2</sub> 0 1 0	1 1 0	1 1 0
m <sub>3</sub> 0 1 1	X X X	X X X
m <sub>4</sub> 1 0 0	X X X	X X X
m <sub>5</sub> 1 0 1	0 1 0	0 1 0
m <sub>6</sub> 1 1 0	1 1 1	1 1 1
m <sub>7</sub> 1 1 1	0 0 0	0 0 0



$$D_2 = \overline{Q}_3$$

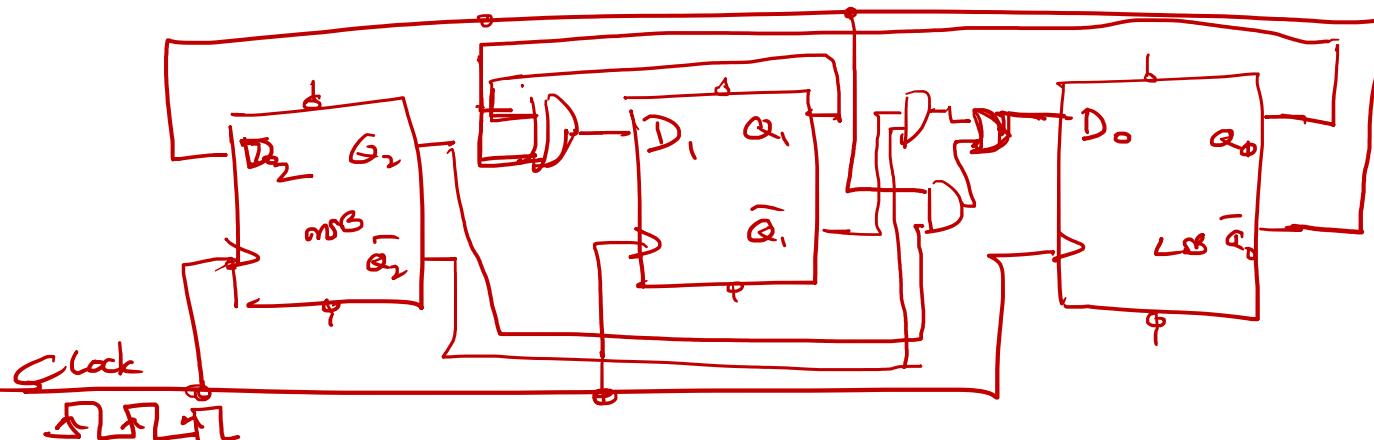
$$D_1 = \overline{Q}_1 \overline{Q}_0 + \overline{Q}_1 Q_0$$

$$D_1 = Q_1 \oplus Q_0$$

$$D_0 = \overline{Q}_1 \overline{Q}_0 + Q_2 \overline{Q}_0$$

or

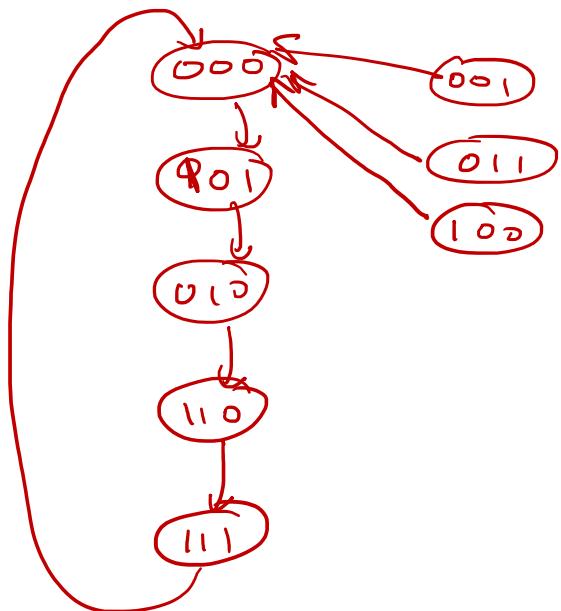
$$D_0 = \overline{Q}_2 \overline{Q}_1 + Q_2 \overline{Q}_0$$



Ex.

Design 3-bit synchronous counter to count according to the given sequence using D ffs. All the undefined states should go to state 0.---Self correcting counters.

- $0 \rightarrow 5 \rightarrow 2 \rightarrow 6 \rightarrow 7 \rightarrow 0$



P.S	N.S
$Q_2 Q_1 Q_0$	$Q_2 \bar{Q}_1 \bar{Q}_0$
0 0 0	1 0 1
0 0 1	<u>0 0 0</u>
0 1 0	1 1 0
0 1 1	<u>0 0 0</u>
1 0 0	<u>0 0 0</u>
1 0 1	0 1 0
1 1 0	1 1 0
1 1 1	0 0 0

Generation

D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	1
0	0	0
-	-	-
0	0	0
0	0	0
0	1	0
0	1	0
0	0	0

$$D_0 = \bar{Q}_2 \bar{Q}_1 \bar{Q}_0$$
$$D_1 = m_2 + m_5 + m_6$$

$Q_2$	0	0	0	1	1
0	0	0	0	0	1
1	0	1	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0

$$D_1 = Q_2 \bar{Q}_0 + Q_2 \bar{Q}_1 \bar{Q}_0$$
$$D_2 = Q_1 \bar{Q}_0 + \underline{\bar{Q}_2 \bar{Q}_0}$$

$Q_2$	0	0	1	1	0
0	0	0	0	0	1
1	0	0	0	1	1
0	1	0	0	0	0
1	0	1	0	0	0

Please Draw the circuit

$$D_0 = \bar{Q}_2 \bar{Q}_1 \bar{Q}_0$$

$$D_1 = Q_1 \bar{Q}_0 + Q_2 \bar{Q}_1 Q_0$$

$$\underline{\underline{D_2 = Q_1 \bar{Q}_0 + \bar{Q}_2 \bar{Q}_0}}$$

Design 3-bit synchronous counter to count according to the given sequence using D ffs. All the undefined states should go to next valid/defined state.

- $0 \rightarrow 5 \rightarrow 2 \rightarrow 6 \rightarrow 7 \rightarrow 0$

next Valid code for unused code

$$\begin{array}{ccc} 001 & \rightarrow & 010 \\ 011 & \rightarrow & 101 \\ 100 & \rightarrow & 101 \end{array}$$

Active High Asynchronous input

Next Valid Code

Assuming we are going upwards

Downwards

$$\begin{array}{c} 000 \\ 010 \\ 010 \end{array}$$

What is

Defined state

Ex. all unused codes go to

Valid Code

say

110

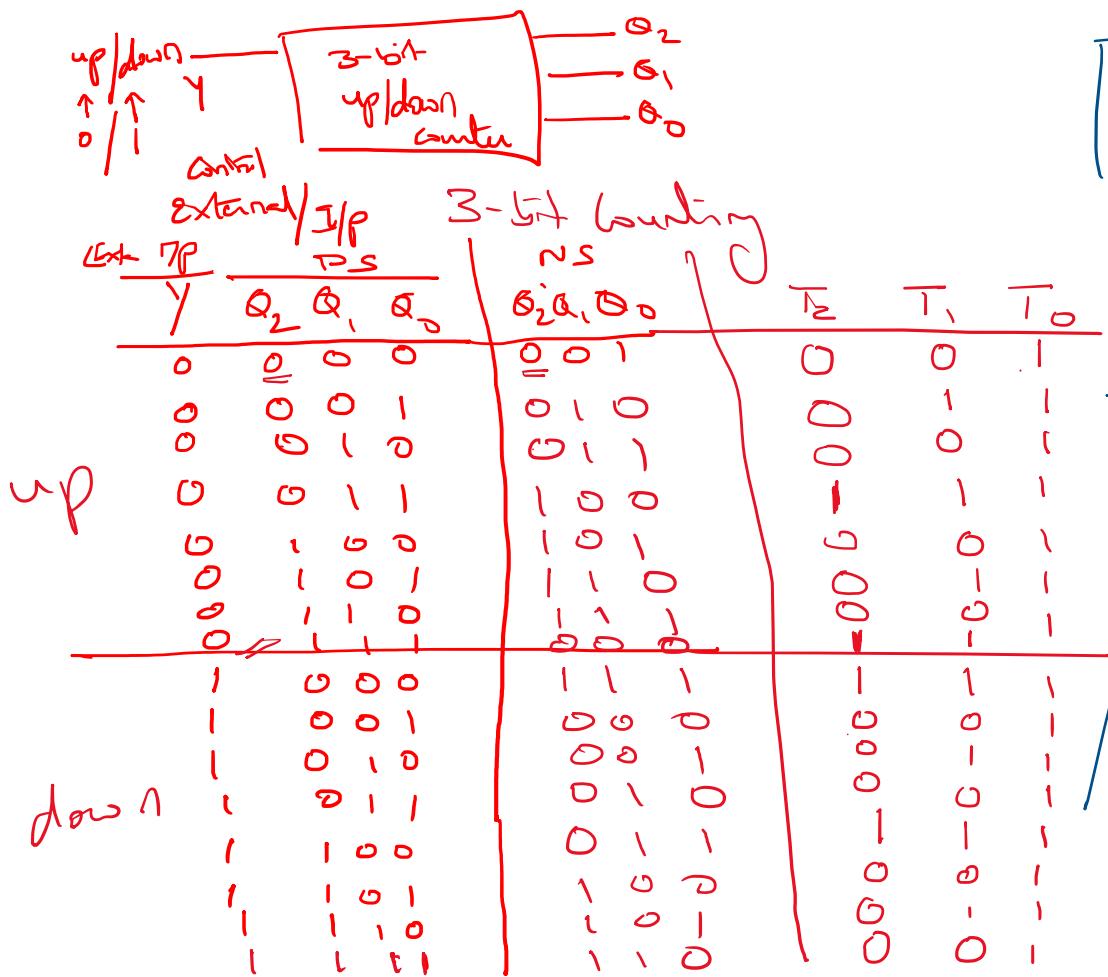
$$\begin{array}{ccc} 001 & \rightarrow & 110 \\ 011 & \rightarrow & 110 \\ 100 & \rightarrow & 110 \end{array}$$

## 9. Design a 3-bit (up/down) synchronous binary counter using T ffs

*Active low / Active High*

*up down  
0 / 1*

- If control input up/down = 0, counter should count upwards from the present count or else it has to count downwards from the present count.



$$\bar{T}_0 = \bar{1}$$

$$\bar{T}_1 = \bar{1} \oplus Q_0 + \bar{1} \bar{Q}_0$$

$$T_1 = \bar{1} \oplus Q_0$$

$$T_2 = \bar{1} Q_1 Q_0 + \bar{1} \bar{Q}_1 Q_0$$

$T_1$   $\bar{Q}_1 Q_0$

$T_2$   $\bar{Q}_2 \bar{Q}_1 Q_0$

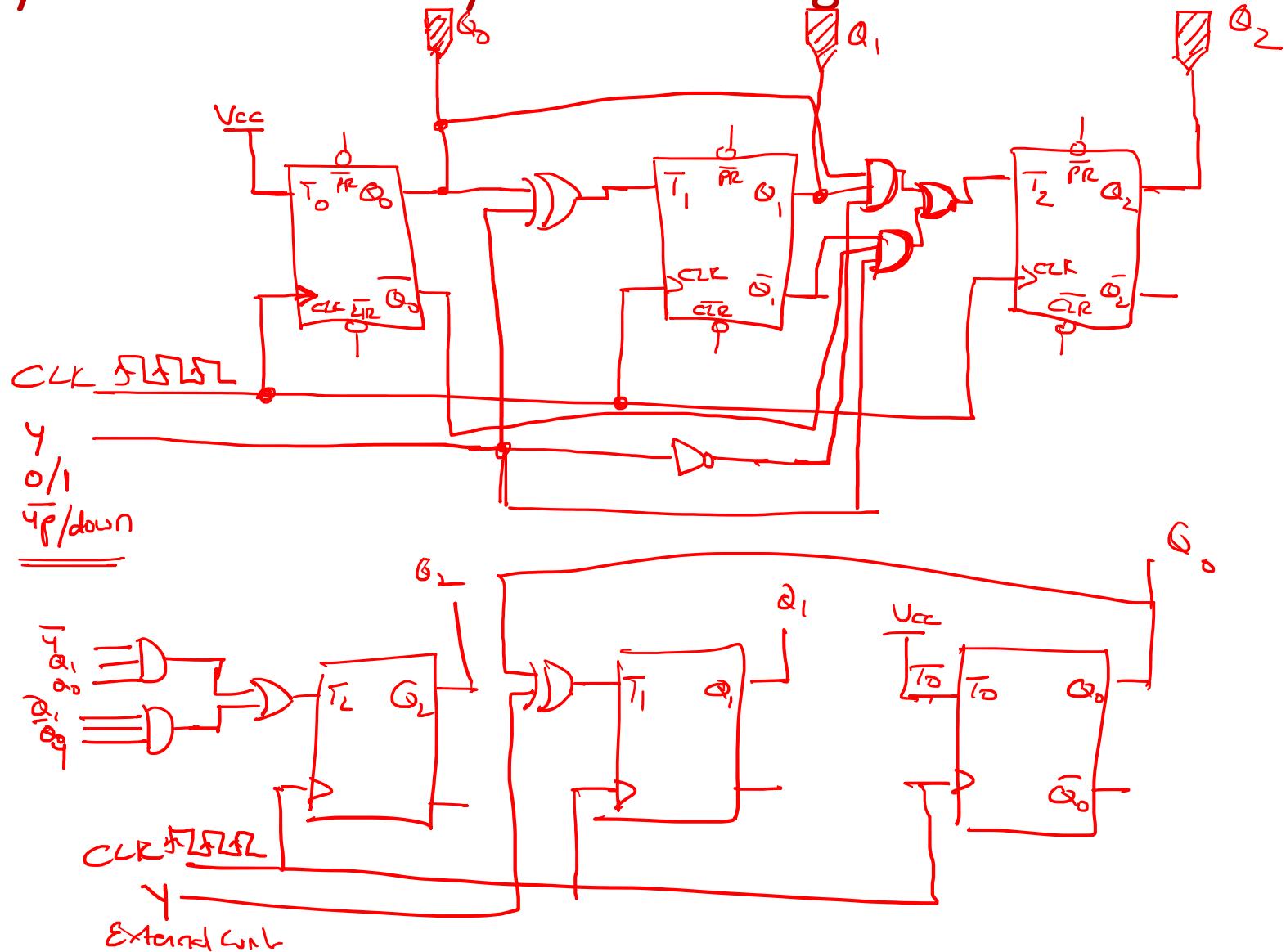
	00	01	11	10
00	0	1	0	0
01	0	1	0	0
11	1	0	0	1
10	1	0	0	1

$T_2$   $\bar{Q}_2 \bar{Q}_1 Q_0$

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

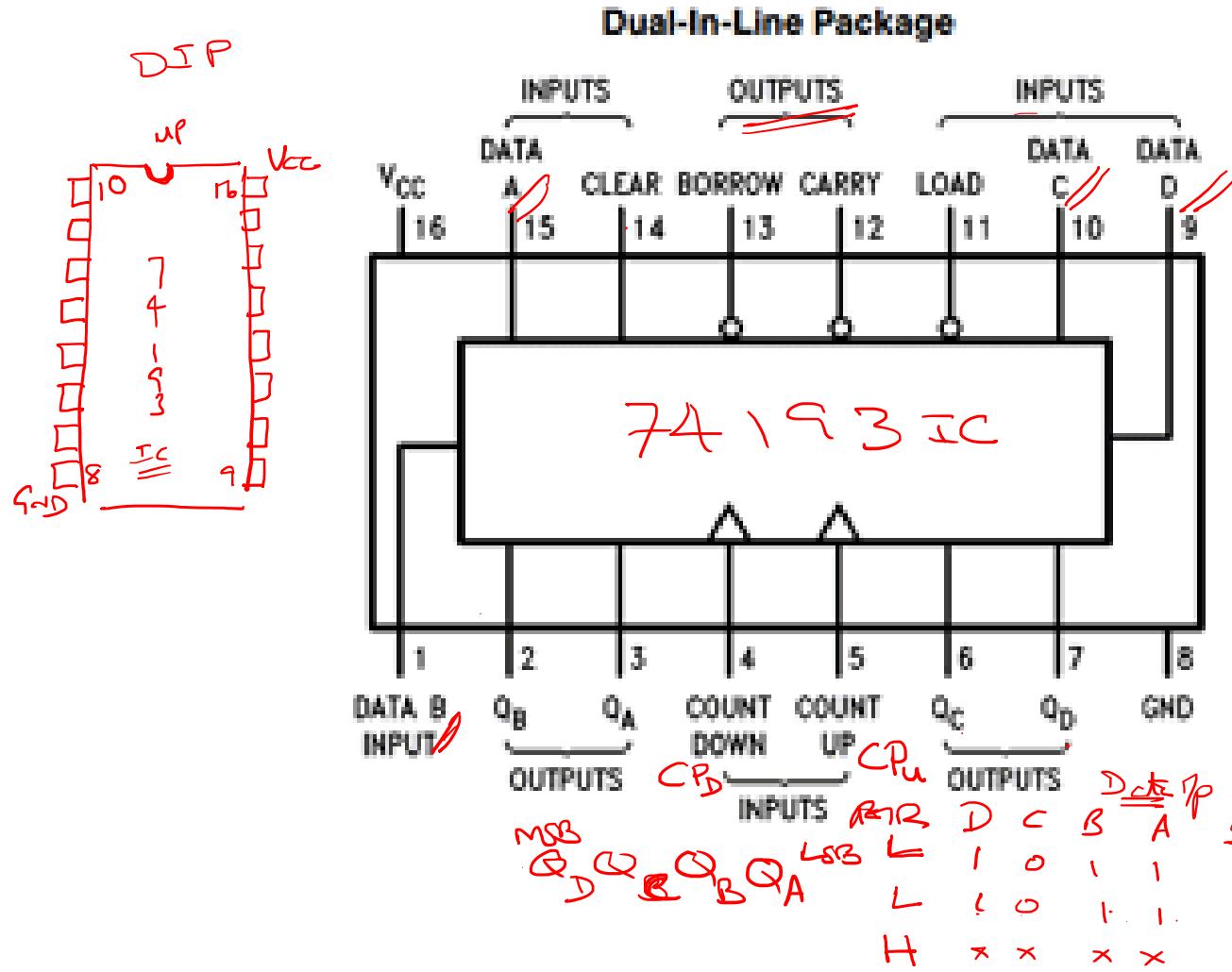
# 3-bit up/down synchronous binary counter using T ffs

$$\begin{aligned}
 T_0 &= 1 \\
 \bar{T}_1 &= \bar{Q}_0 + Q_0 \\
 &= \bar{Q} \oplus Q_0 \\
 \bar{T}_2 &= \bar{Q}_1 Q_0 + \bar{Q}_1 \bar{Q}_0
 \end{aligned}$$



# 74193 IC: 4-bit up/down synchronous counter

MOD-16 counter      0000  
1111      ↑↑



Q<sub>D</sub> is the MSB and Q<sub>A</sub> is the LSB

Count up/CP<sub>U</sub>: count up clock input

Count down/CP<sub>D</sub>: count down clock input

LOAD: Asynchronous parallel load (active low)

CLEAR: Asynchronous master reset input (Active High)

CARRY: Terminal count up output

Borrow: Terminal count down output

Data inputs: Parallel data inputs

	Q <sub>D</sub>	Q <sub>C</sub>	Q <sub>B</sub>	Q <sub>A</sub>
H	x	x	x	x
L	0	0	1	1
	0	1	1	1

T<sub>CU</sub>

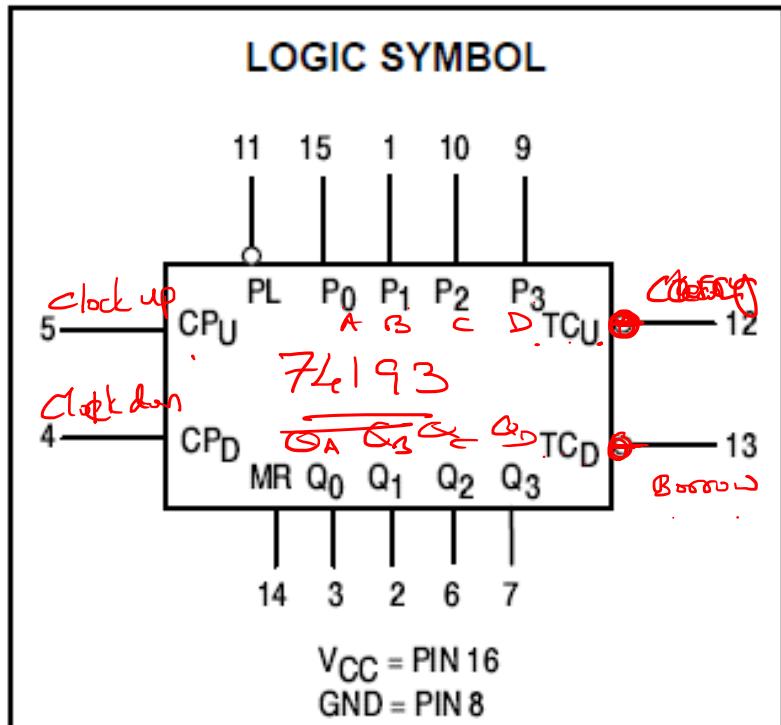
T<sub>CD</sub>

LSB      MSB

A      B      C      D

# 74193 IC: 4-bit up/down synchronous counter

Operation



Highest priority

Active low

For Normal operation  
MR → Low

MODE SELECT TABLE

MR	PL	CPU	CPD	MODE
H	X	X	X	Reset (Asyn.)
L	L	X	X	Preset (Asyn.)
L	H	H	H	No Change
L	H	H	H	Count Up
L	H	H	H	Count Down

D C B A  
P<sub>3</sub> P<sub>2</sub> P<sub>1</sub> P<sub>0</sub> located

L = LOW Voltage Level

H = HIGH Voltage Level

X = Don't Care

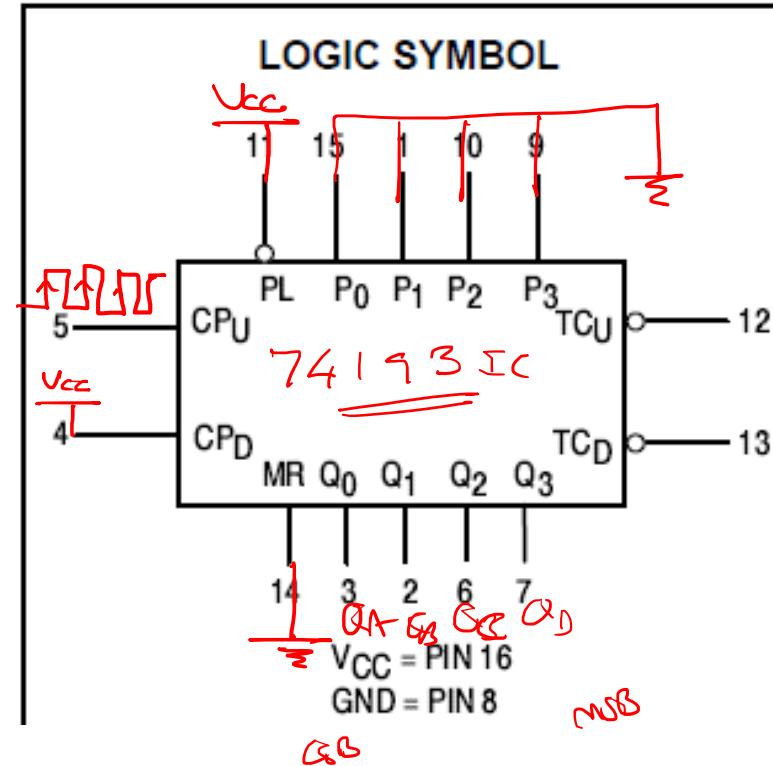
⊓ = LOW-to-HIGH Clock Transition

# Design a MOD-16 binary UP counter using 74193 IC

4-bit binary UP

0000  
0001  
:  
1111  
0000

$\overline{MR} = \text{HIGH/Low}$  ✓  
 $\overline{PL} = \text{HIGH/Low}$  ✗



Design a MOD 16 binary DOWN counter using 74193 IC

(4-bit binary down counting) Clock should be given CPD

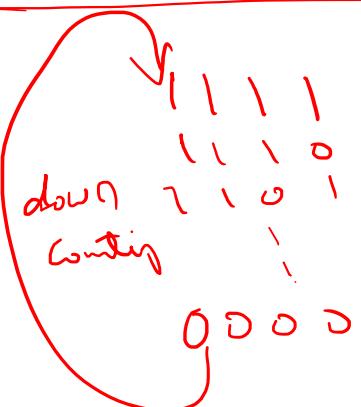
Please Note

A HIGH Level on MR Resets the output to

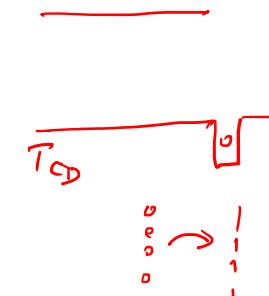
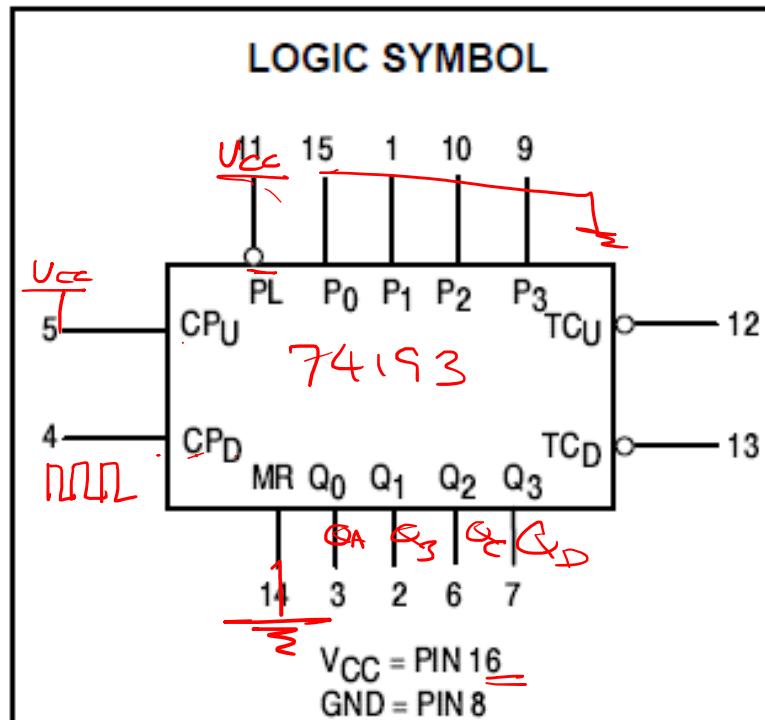
$Q_3 Q_2 Q_1 Q_0 A$   
0 0 0 0

So for Normal Counting

Set MR to low



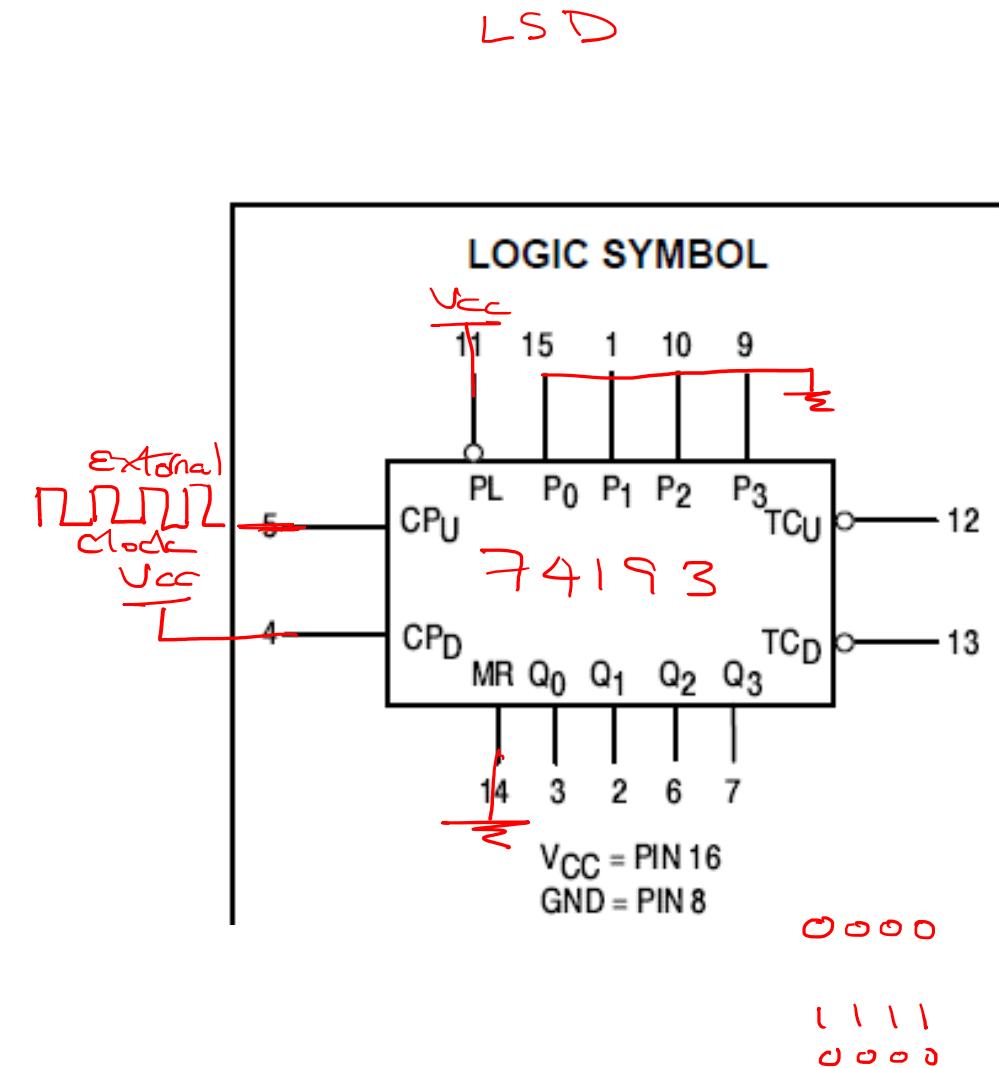
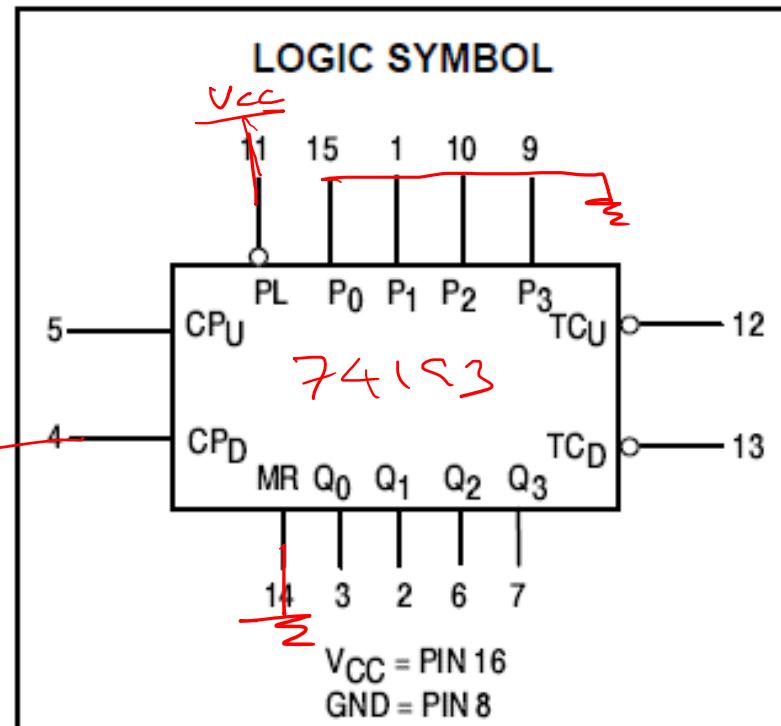
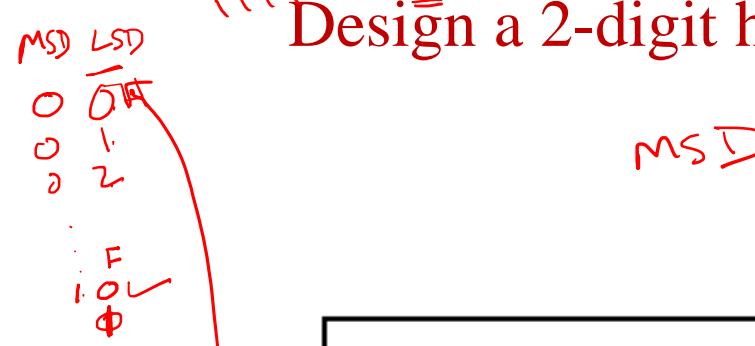
$\bar{PL} \rightarrow \text{HIGH}$



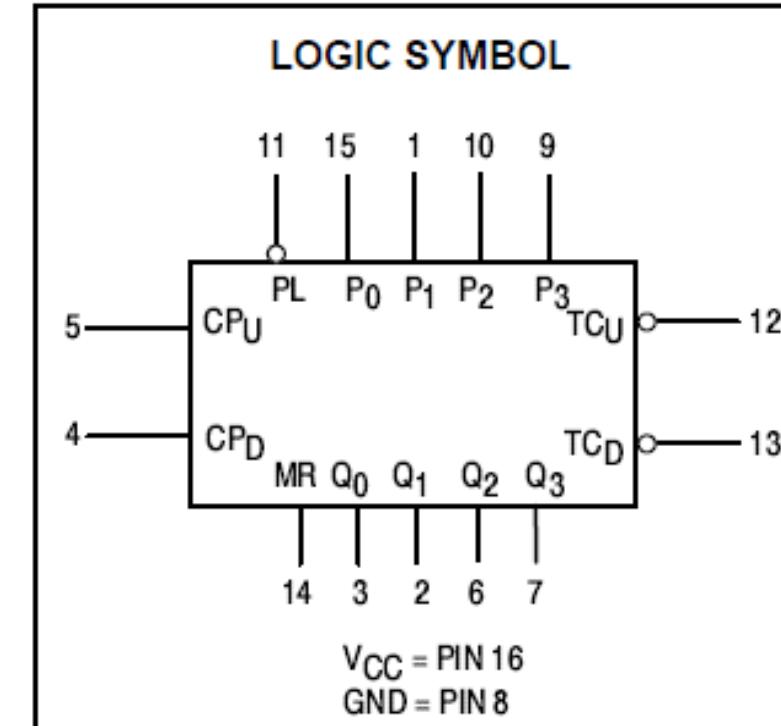
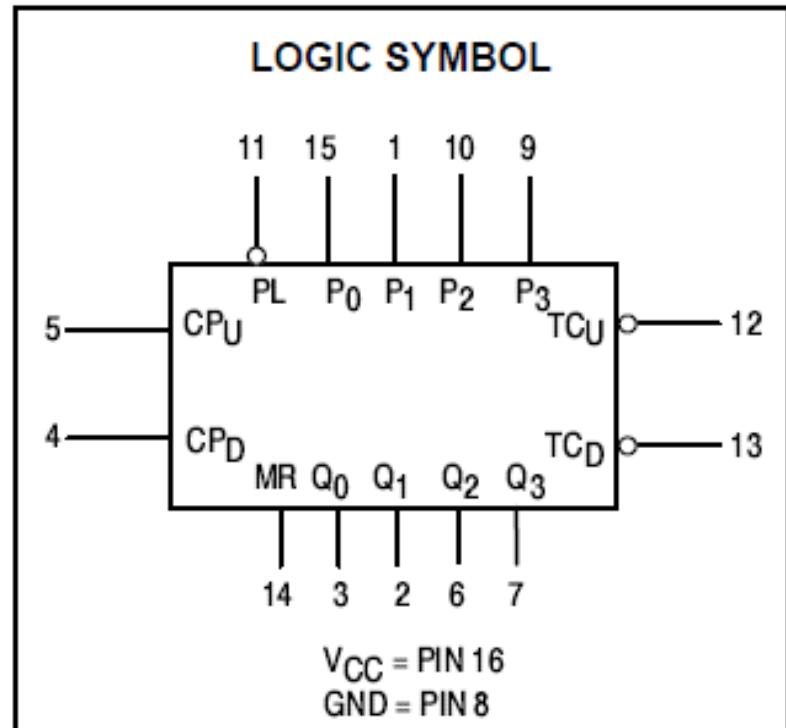
0000 → 01

↑↑↑ → F<sup>H</sup>

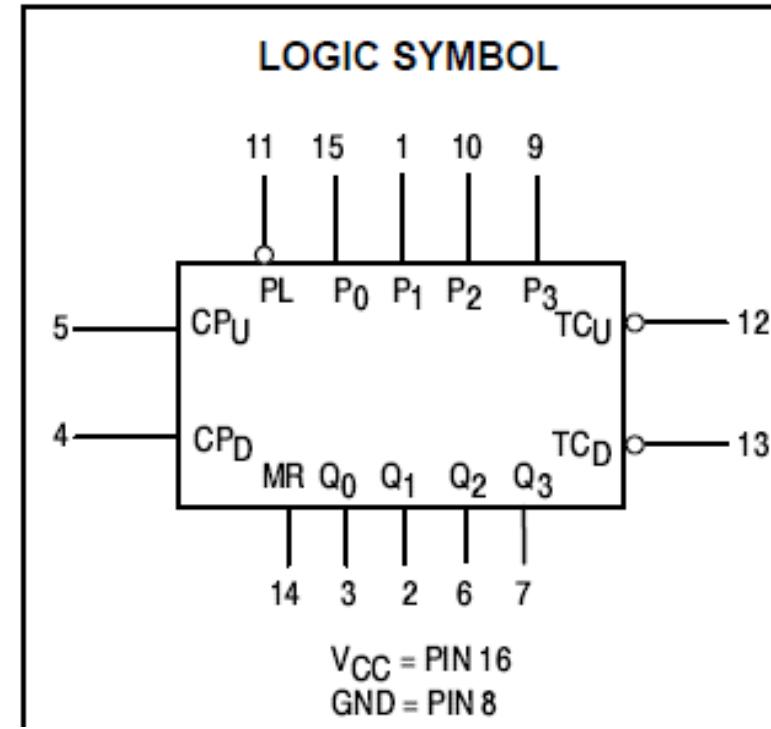
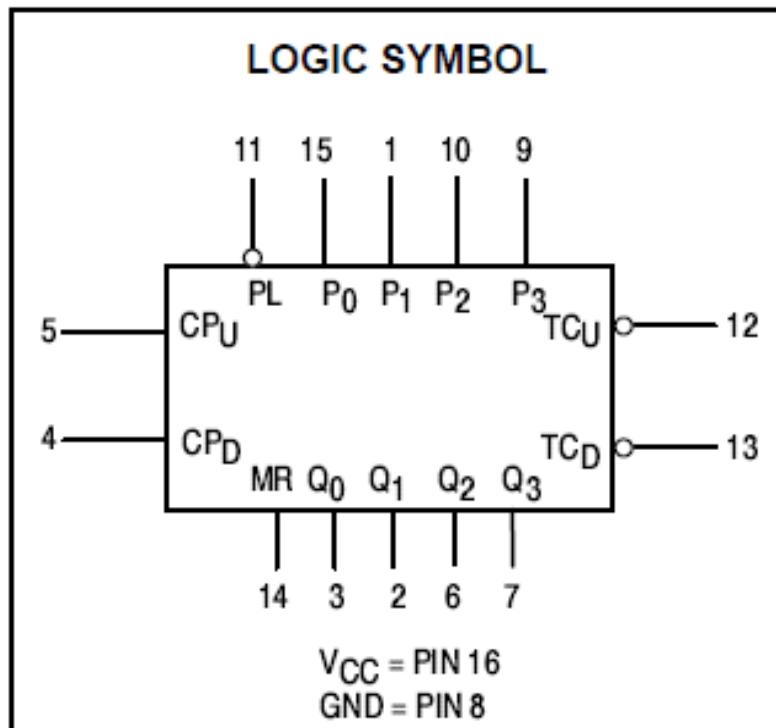
Design a 2-digit hexadecimal UP counter (00-FF H) using 74193 IC



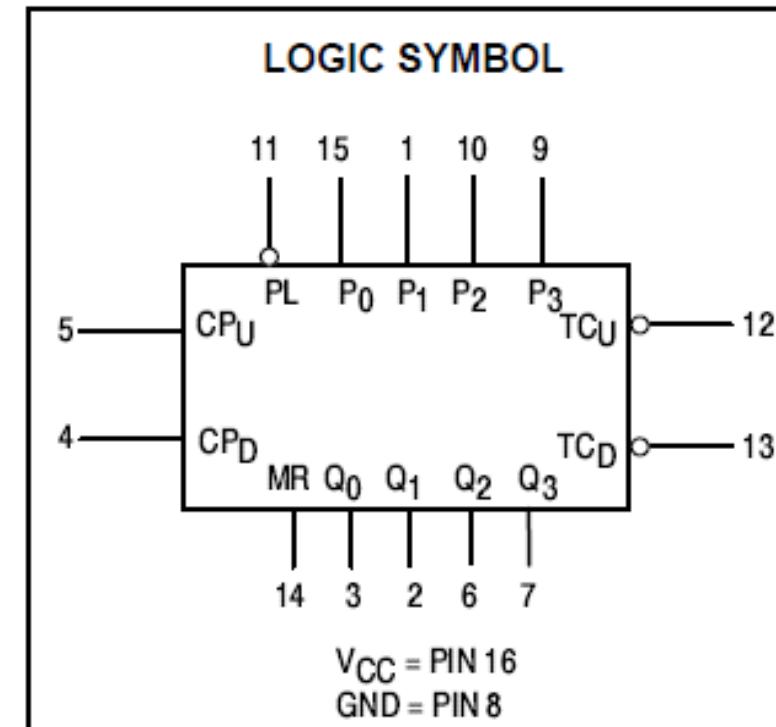
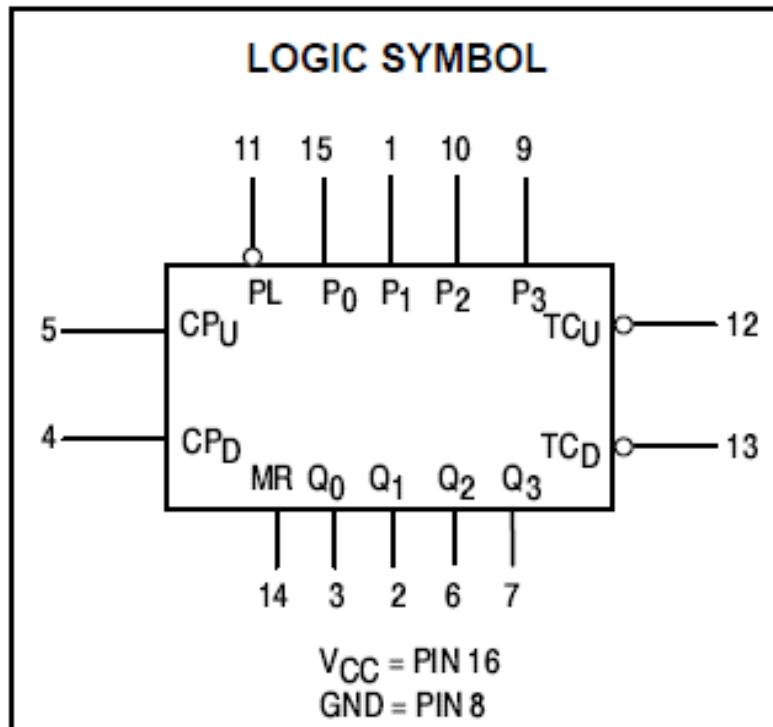
## Design a 2-digit hexadecimal UP counter (00-FF H) using 74193 IC



Design a 2-digit hexadecimal UP counter (18H-99 H) using 74193 IC



# Design a 2-digit hexadecimal DOWN counter (A1H-23 H) using 74193 IC



## Counters with parallel load(Presettable)

- Flip flops will have an additional asynchronous input referred as ‘load’, which may be active high or low.
- For active low ‘load’ input, if load =0, flip flop should be loaded with external input bit P else output should change according to other synchronous/asynchronous inputs.
- This feature enables the counter to have parallel load capability for transferring an external input /data /count into the counter.

# Flip flop with parallel load: design

Design an presetable asynchronous counter to count between the limits  $(3)_{16}$  to  $(B)_{16}$  using T-ffs

Design an presettable synchronous counter to count between the limits  $(3)_{16}$  to  $(B)_{16}$  using T-ffs

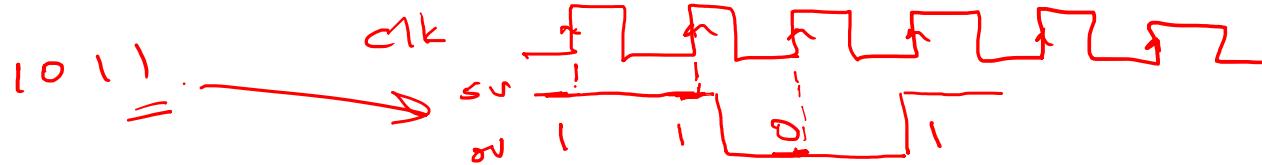
Design an asynchronous counter to count between the limits  $(3)_{16}$  to  $(N)_{16}$ , N is a single digit hexadecimal number with value  $> (3)_{16}$ . Using 7485 IC, 4-bit asynchronous counter (with parallel load) block diagram and external gates..

- Questions: ?



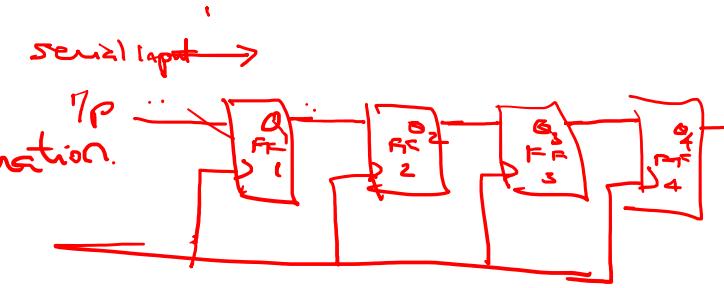
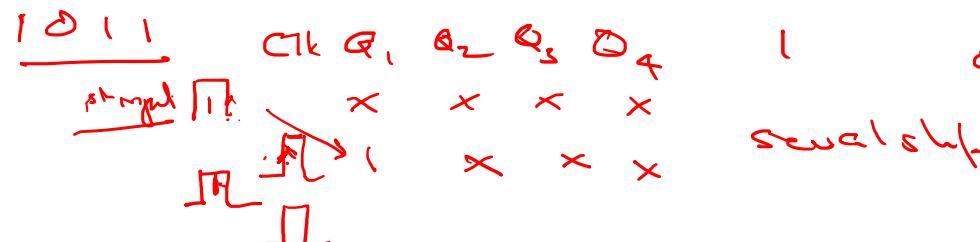
Shift registers and shift register  
counters

# Registers



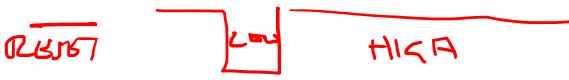
- Register is a group of flip flops (D, or SR, or JK)
- Each flip flop can store one-bit data. n-bit register can hold n-bit data.
- There are two ways to shift the data into the register and two ways to shift the data out of the register.
- Accordingly: 4 categories of shift register are
  - Serial in Serial out (SISO)
  - Serial In parallel out (SIPO)
  - Parallel in Serial out (PISO)
  - Parallel in parallel out (PIPO)

Example: 4-bit shifting operation.



Simple 4-bit shift Register

# SISO and SIPO Register



- Simple 4-bit shift register is shown below (Reset is nothing but clear input)
- Register is cleared using reset/clear input.
- It can be used as Serial-in serial out (at  $Q_D$ ) and Serial-in parallel out shift register.

Little endian concept  
LSB is first

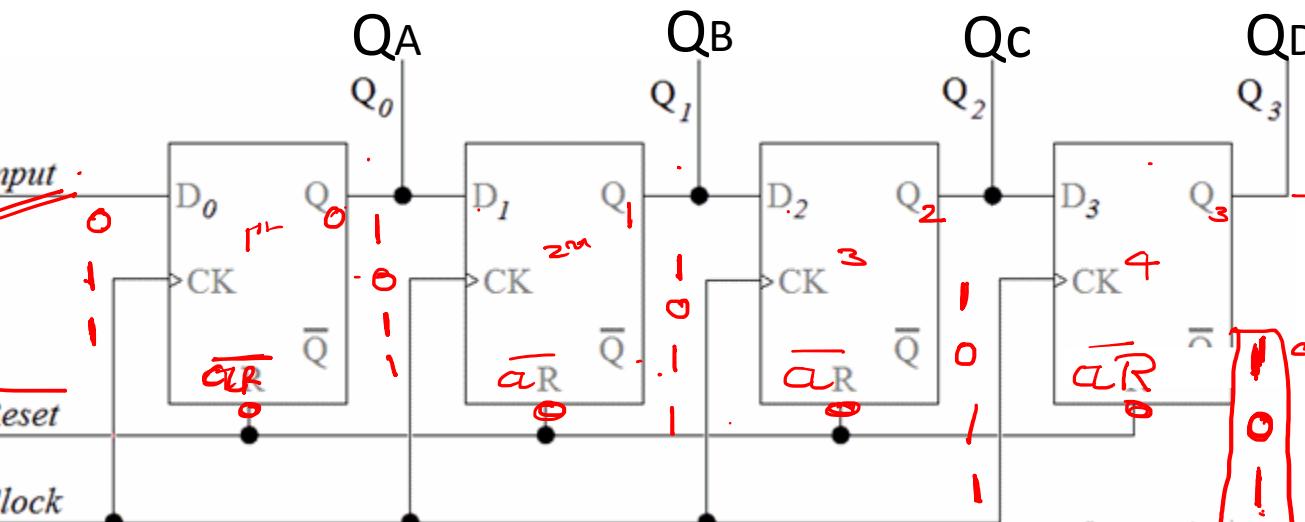
n-bit      MSB    ...    LSB

MSB    LSB  
101

Ex. Lat

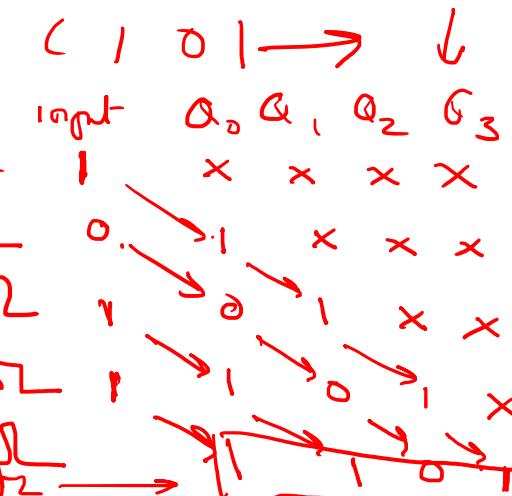
In a n-bit shift register  
the requires  $n$ -clock pulses  
to store the  $n$ -bit data  
serially

at 1st clock pulse  
n-bit (LSB) is available  
parallel edge  
of required bits  
serially



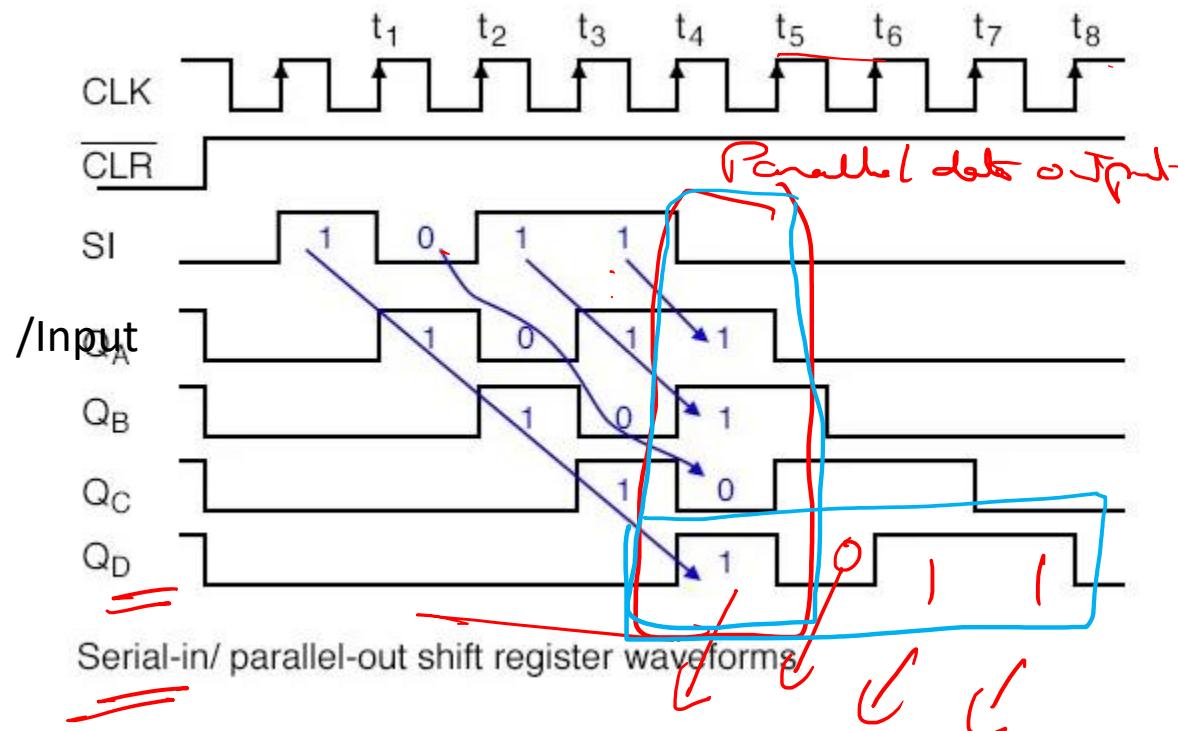
These requires  $(2^n - 1)$

clock pulses  
for completely shifting out  
data serially at  
parallel input

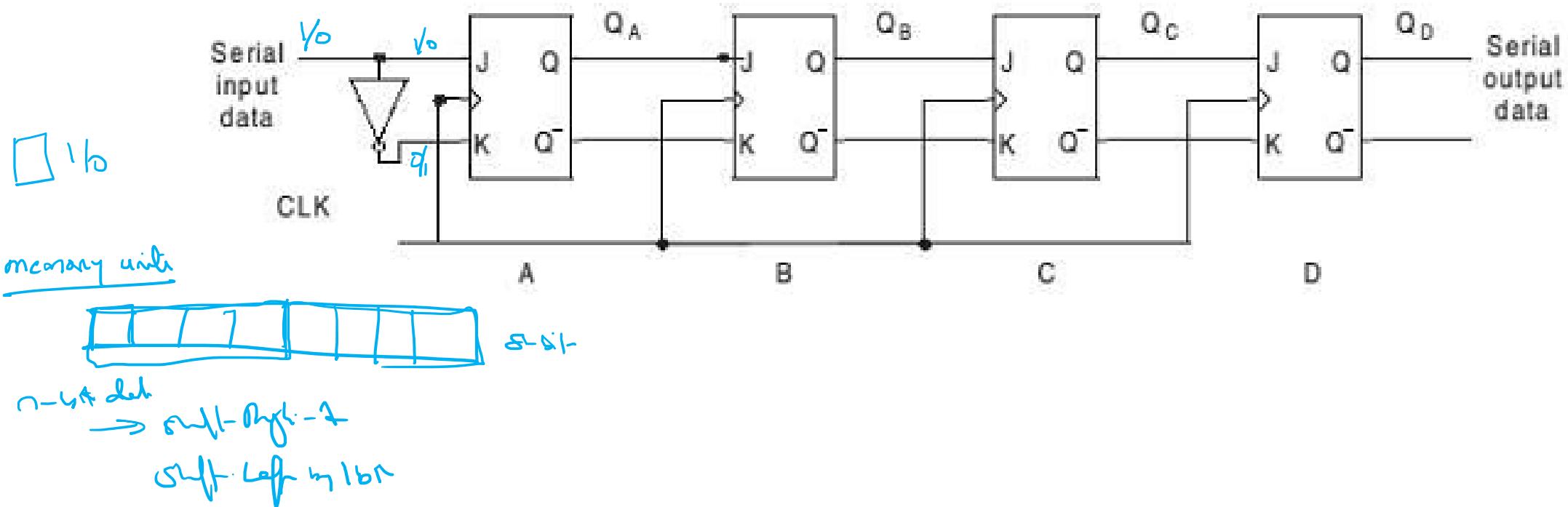


© www.petervis.com

# 4-bit shift register (Serial-in)

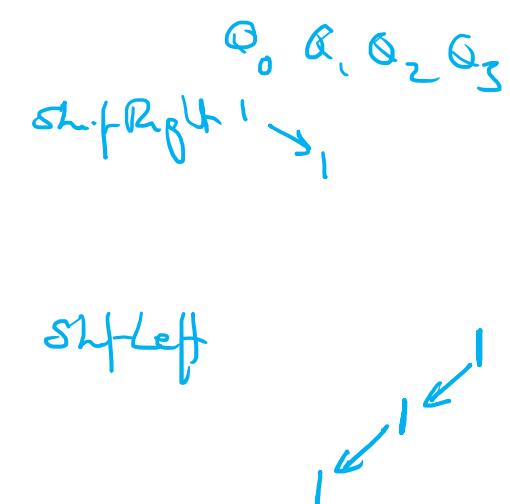
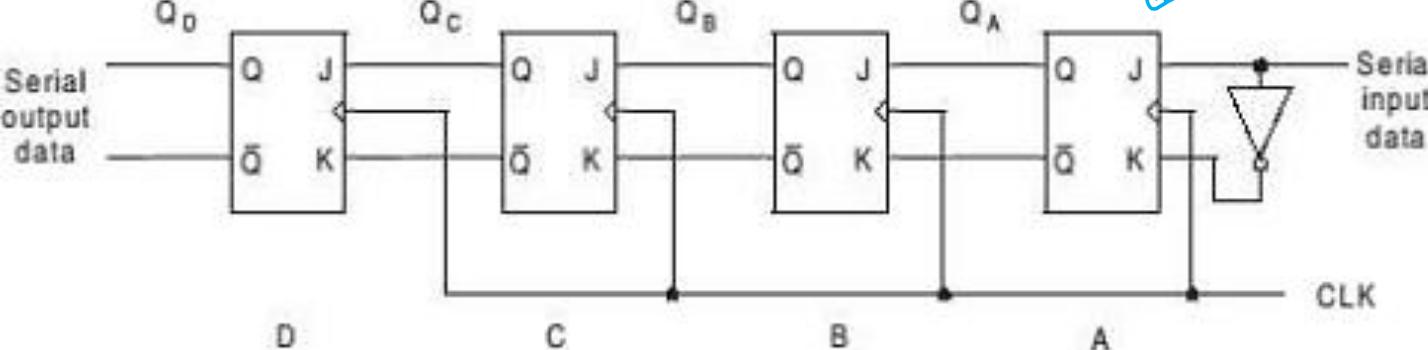
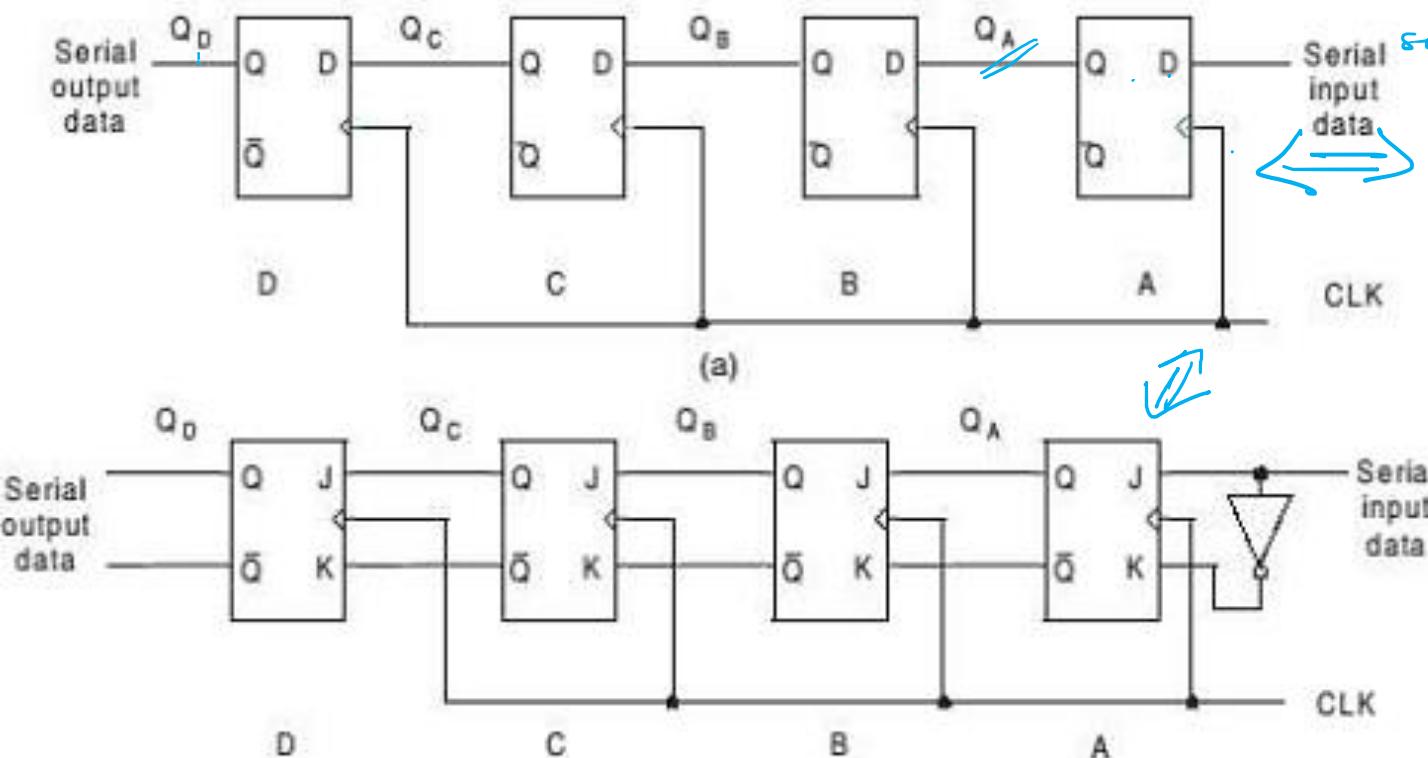
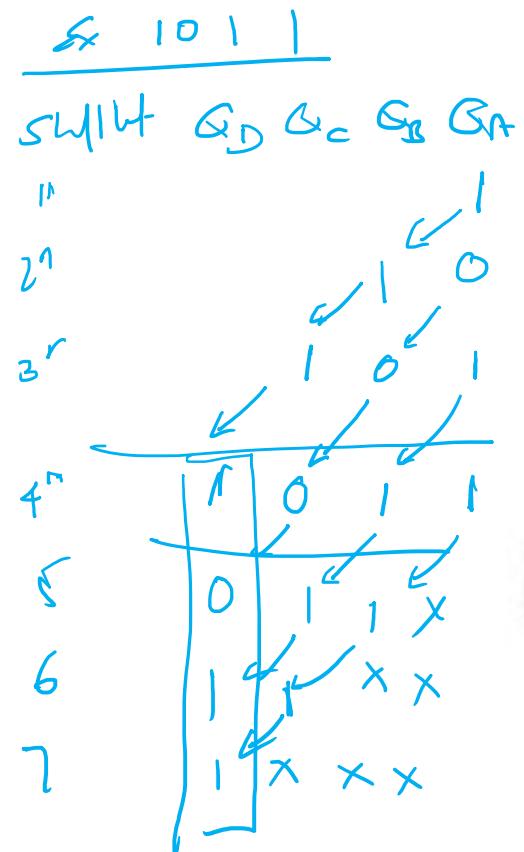


# SISO and SIPO Register using JK ffs



$1011 \rightleftharpoons$  shf Right  
shf Left  
LSB front → MSB  
MSB from

# SISO and SIPO register with shift left

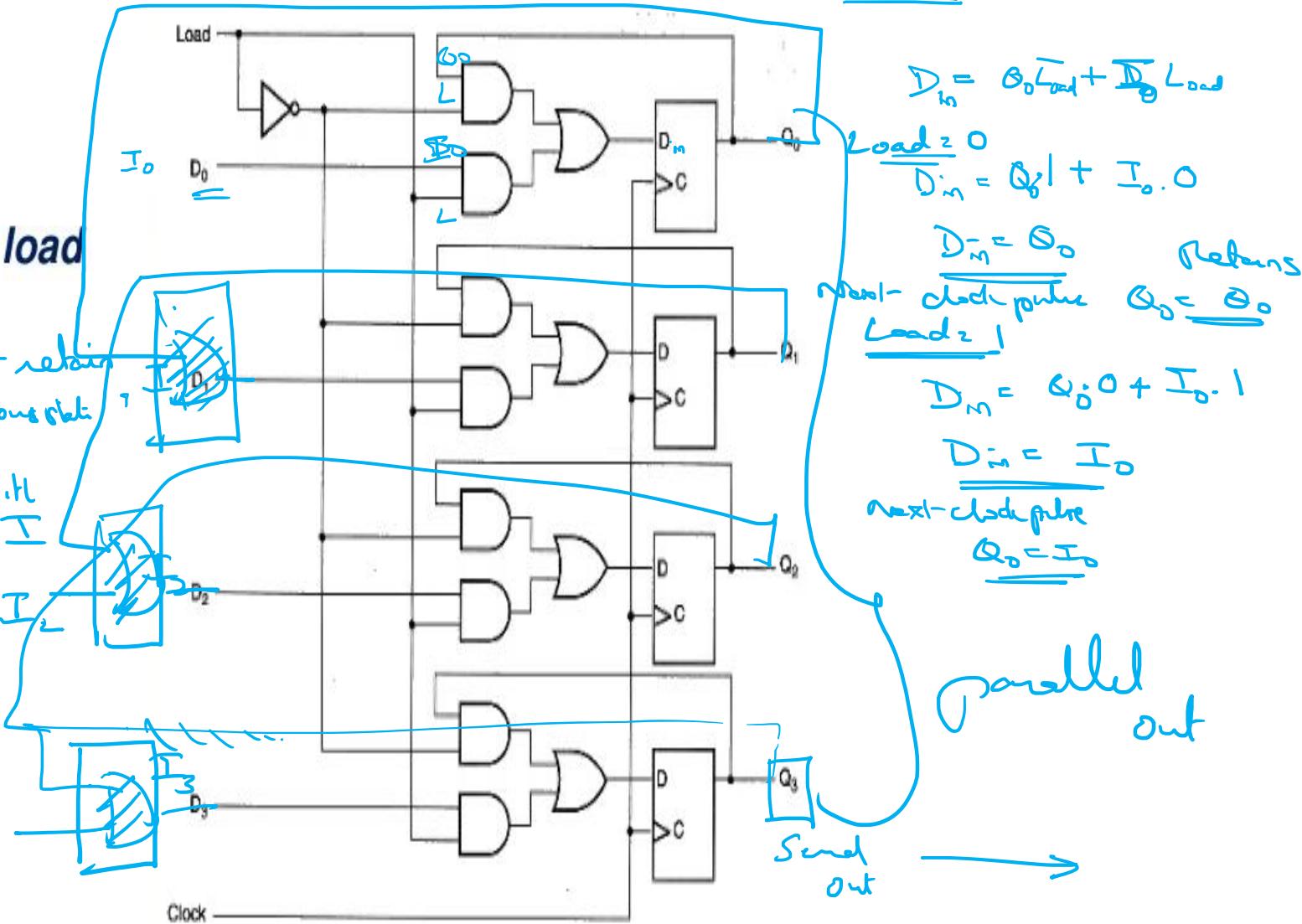
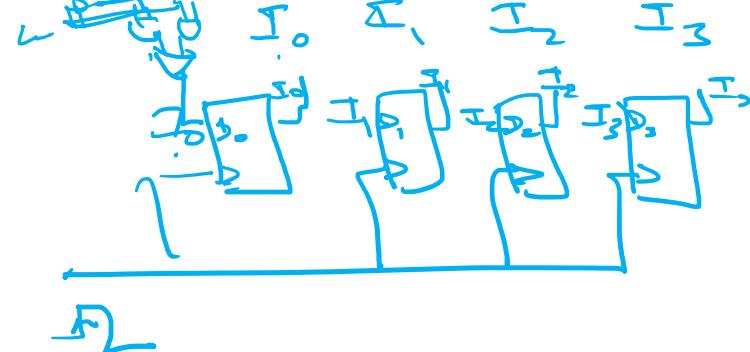


# Register with parallel load (PISO or PIPO)

*4-bit register with parallel load*

If Load = 0 No loading ff's must retain previous state

If Load = 1 ff's should be loaded with input-data I



# 4-bit bidirectional shift register using D/SR/JK ffs

- If shift left/right =0, shift right else shift left...

left/right

$$0 \rightarrow \text{right}$$

$$1 \rightarrow \text{left}$$

- **Solve it by yourself...**

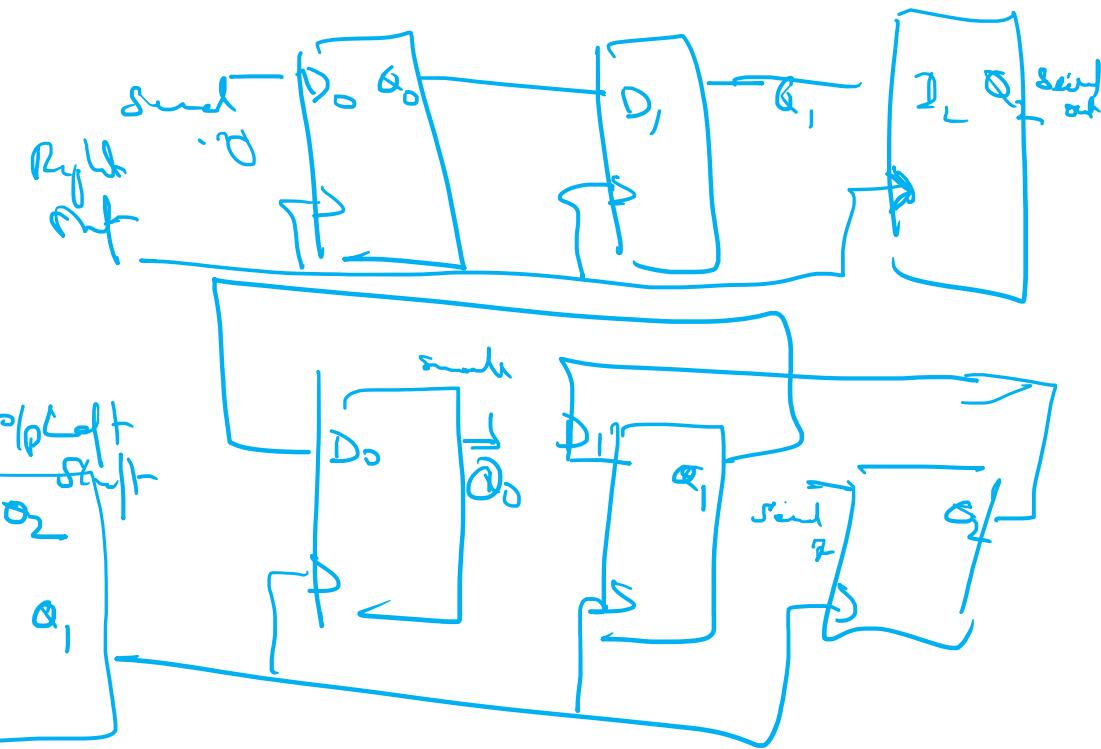
left/right

left/right

left/right

left/right

$$\begin{array}{cccccc} D_0 & D_1 & D_2 & \text{shift} \\ I_0 & Q_0 & Q_1 & Q_2 \\ Q_1 & Q_2 & I_0 & Q_1 \end{array}$$

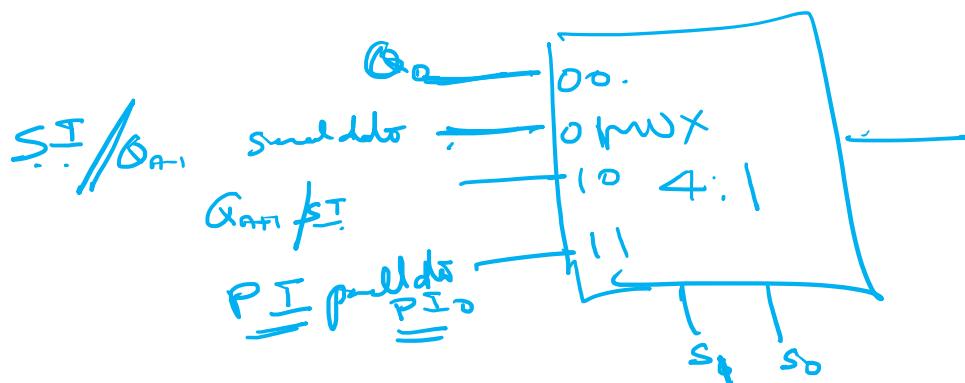


# 4-bit universal shift register

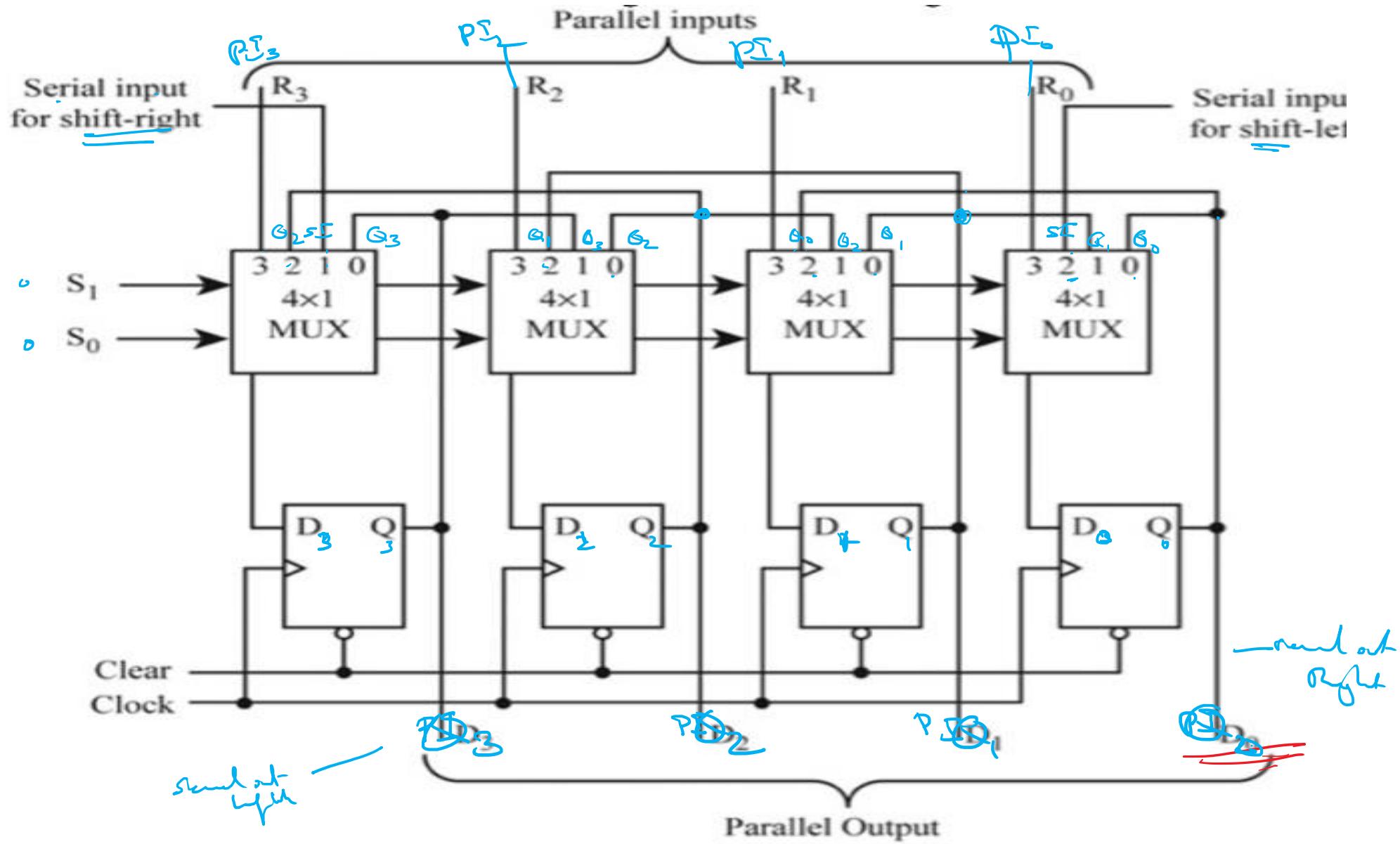
External Control input

Mode Control		Register Operation
S1	S0	
0	0	No change or previous state
0	1	Shift right
1	0	Shift left
1	1	Parallel load

External parallel Data loaded onto the n-bit register

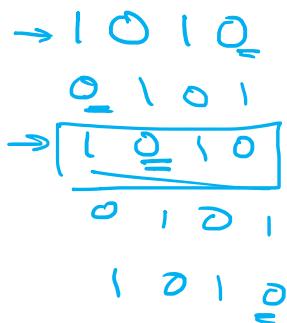


# 4-bit universal shift register



# Shift register counters:

- Ring counter
- Johnson counter /Twisted-ring counter



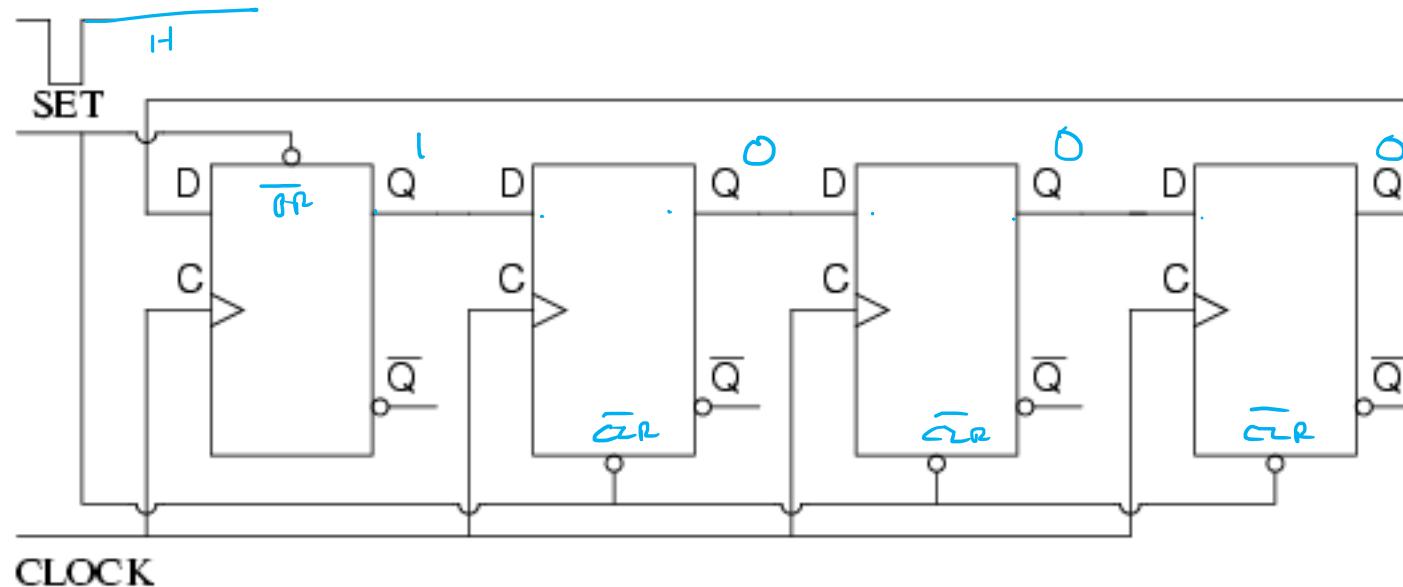
Initial state stored in registers  
are shifted in a circular  
manner

n-bit shift register  
with n-clock pulses  
original sequence/state  
reappears

However there are sides

0000  
1111

# Ring counter



Set one stage, clear three stages

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

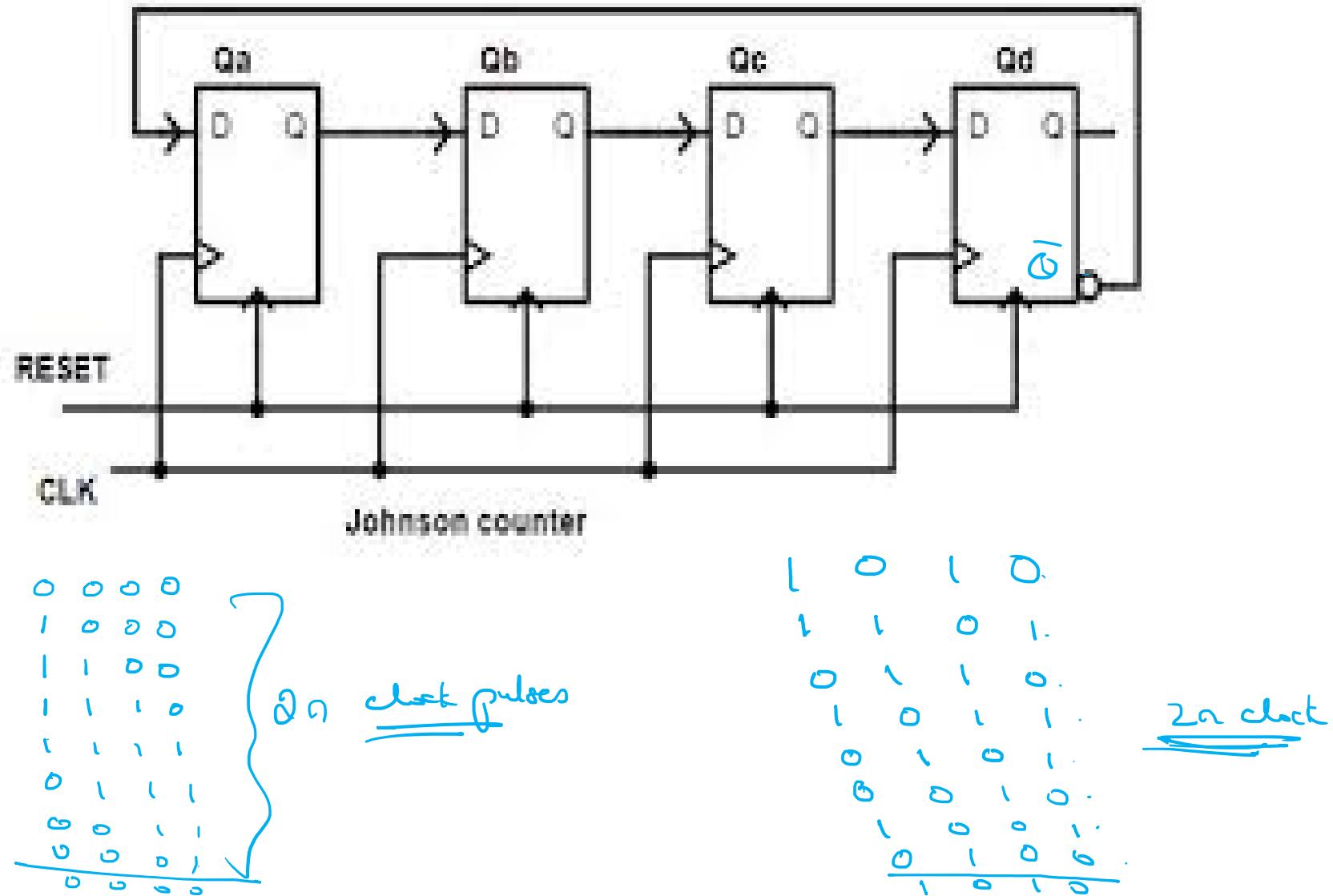
Not to use  
0000  
1111  
1010  
0101

Clock Cycle	Q <sub>1</sub>	Q <sub>2</sub>	Q <sub>3</sub>	Q <sub>4</sub>
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	0	0	0
6	0	1	0	0
.	.	.	.	.
.	.	.	.	.

Bit-pattern repeats for every 4 clock cycles

# Johnson counter /Twisted-ring counter

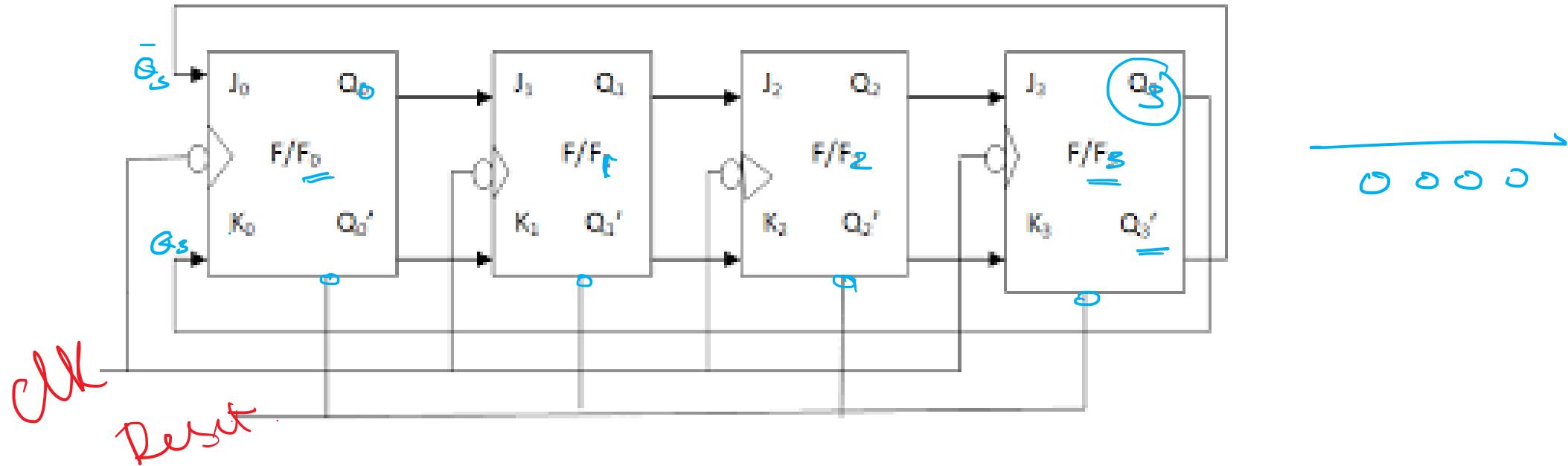
$Q_4$	$Q_3$	$Q_2$	$Q_1$
0	0	0	0
1	0	0	0
1	1	0	0
1	1	1	0
1	1	1	1
0	1	1	1
0	0	1	1
0	0	0	1
repeat			



# Johnson counter /Twisted-ring counter using JK ffs

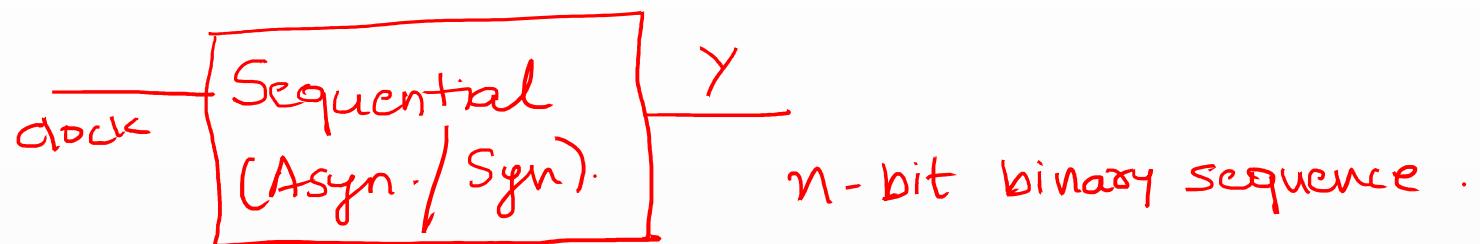
0 0 0 0

Qn



# SEQUENCE GENERATORS, SHIFT REGISTERS AND SHIFT REGISTER COUNTERS

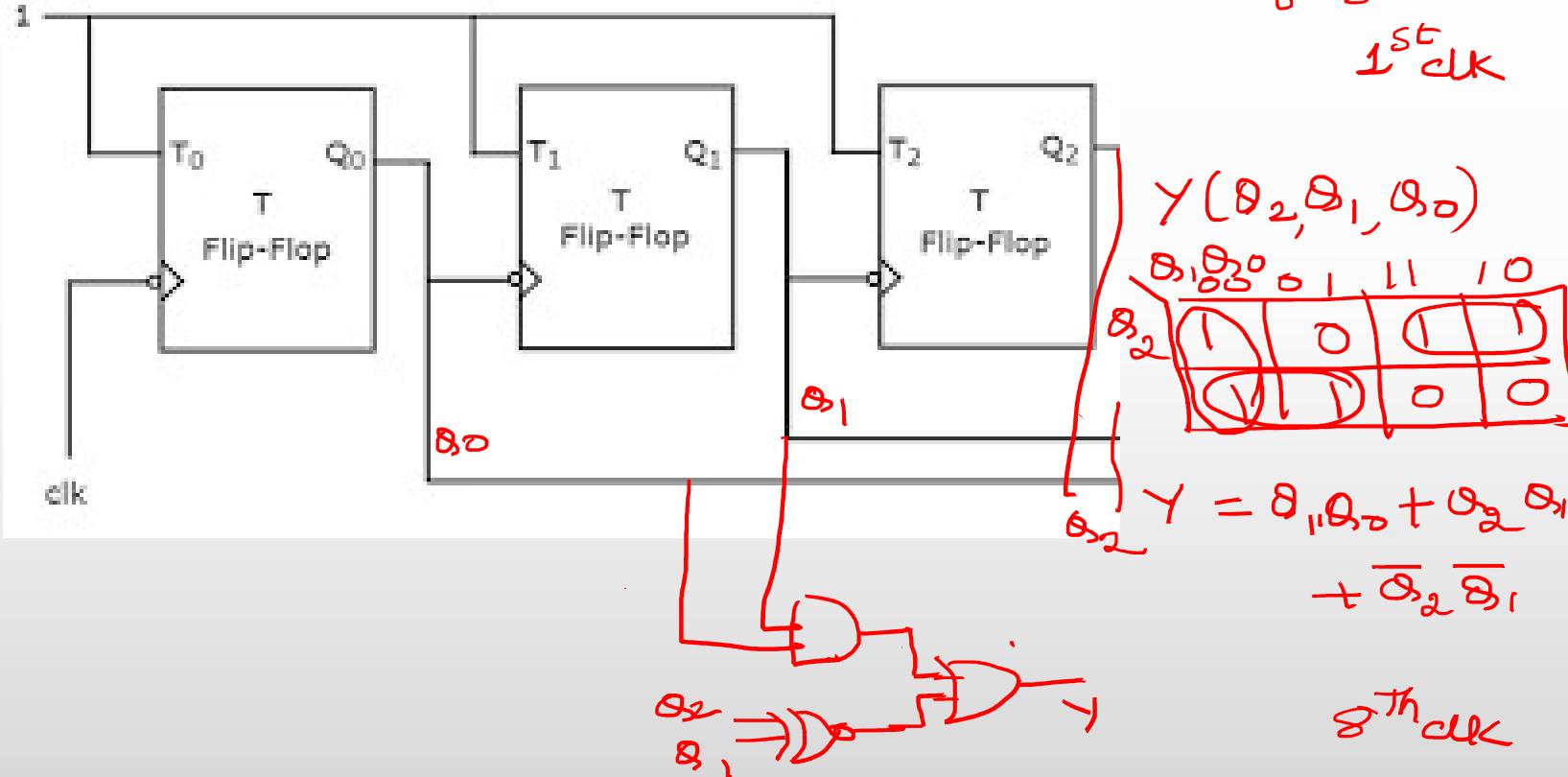
# Sequence generator:



# Sequence Generator

- Design a sequence generator circuit to generate the sequence 10111100 using Asynchronous counter. Use negative edge triggered T flip flop for the design.

- MOD 8 counter

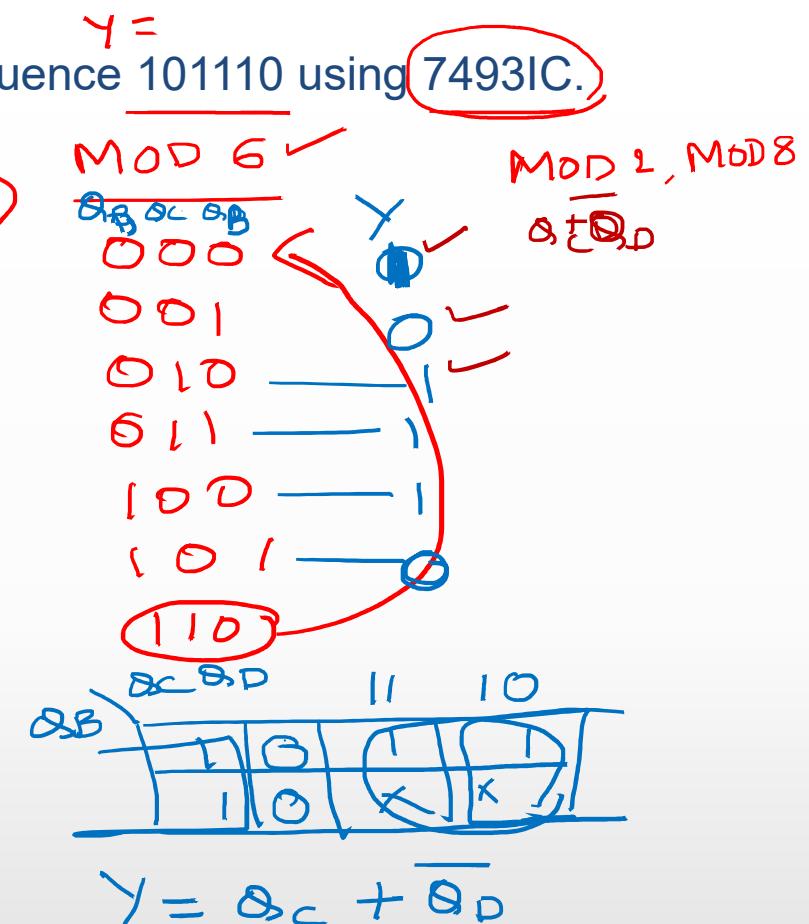
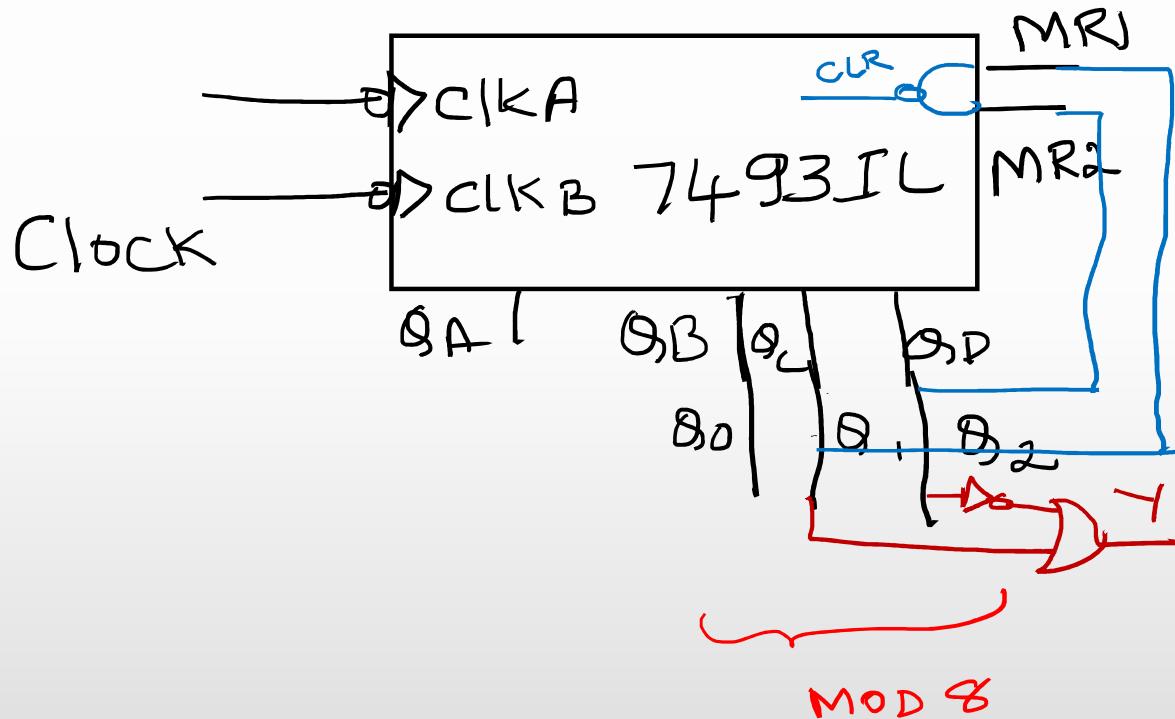


Q2	Q1	Q0	Y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

- Design a sequence generator circuit to generate the sequence  $101110$  using 7493IC.

① MOD 8 → MOD 6 ✓

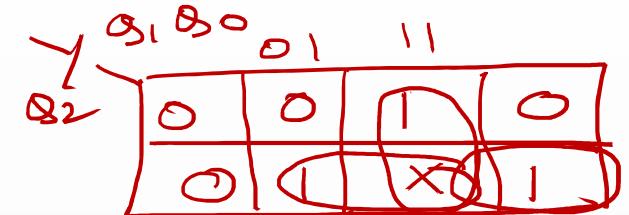
② Find the expression for  $y(Q_B, Q_C, Q_D)$



- Design a sequence generator circuit to generate the sequence 0001011 using synchronous counter. Use D flip flop for the design.

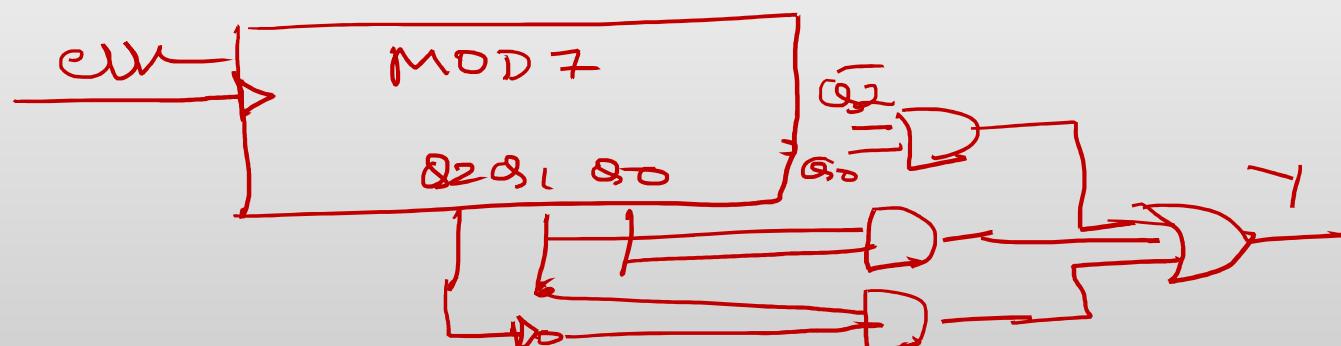
Present	Next State	$D_2$	$D_1$	$D_0$	$Y$
000	001	0	0	1	0
001	010	0	1	0	0
010	011	0	1	1	0
011	100	1	0	0	1
100	101	1	0	1	0
101	110	1	1	0	1
110	000	0	0	0	1
111	000	0	0	0	X

~~3-bit~~ MOD 7 0-1-2...6



$$Y = Q_1 Q_0 + \bar{Q}_2 Q_0 + \bar{Q}_2 Q_1$$

Derive the expressions  
for  $D_2 D_1 D_0$



Generate the sequence: 101011100 using  
TRIG3IC & external gates

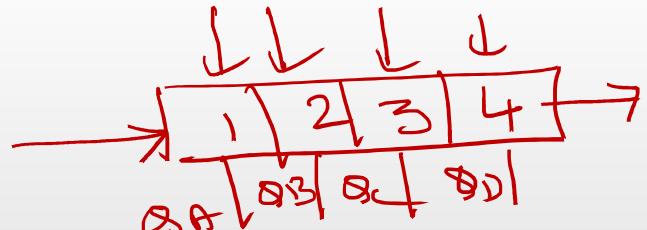
Tog

# Registers

$$C = A + B$$

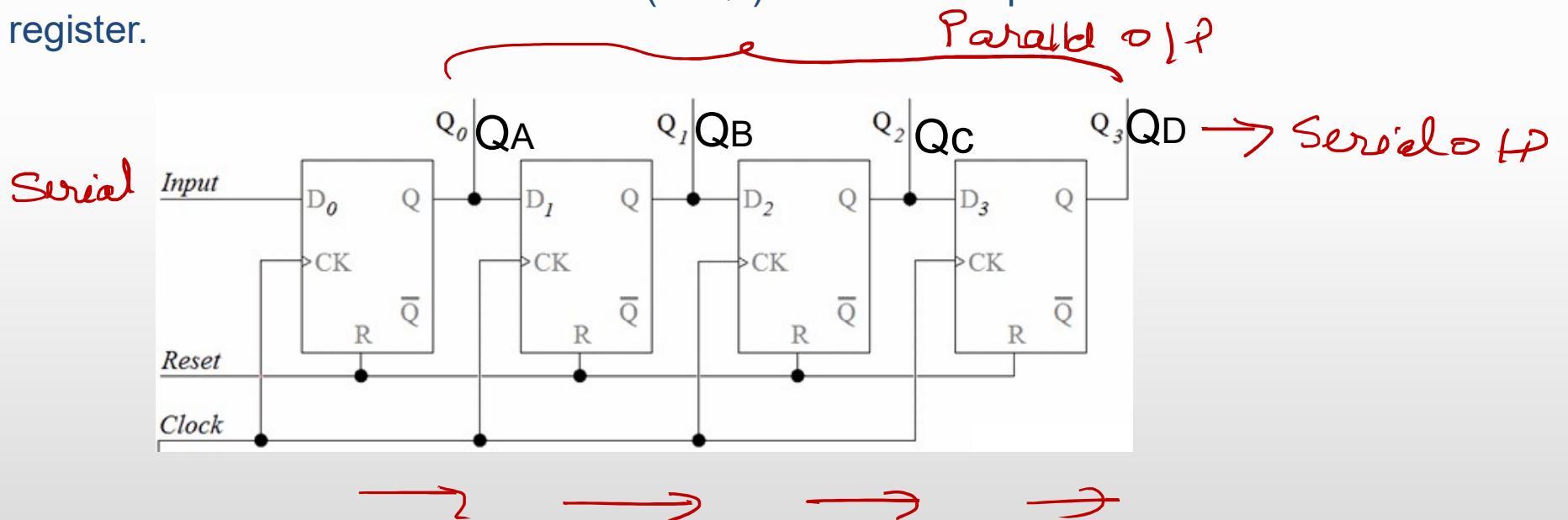
n-bit register

- Register is a group of flip flops (D, or SR, or JK)
- Each flip flop can store one-bit data. n-bit register can hold n-bit data.
- There are two ways to shift the data into the register and two ways to shift the data out of the register.
- Accordingly: 4 categories of shift register are
  - *Serial in Serial out (SISO)*
  - *Serial In parallel out (SIPO)*
  - *Parallel in Serial out (PISO)*
  - *Parallel in parallel out (PIPO)*

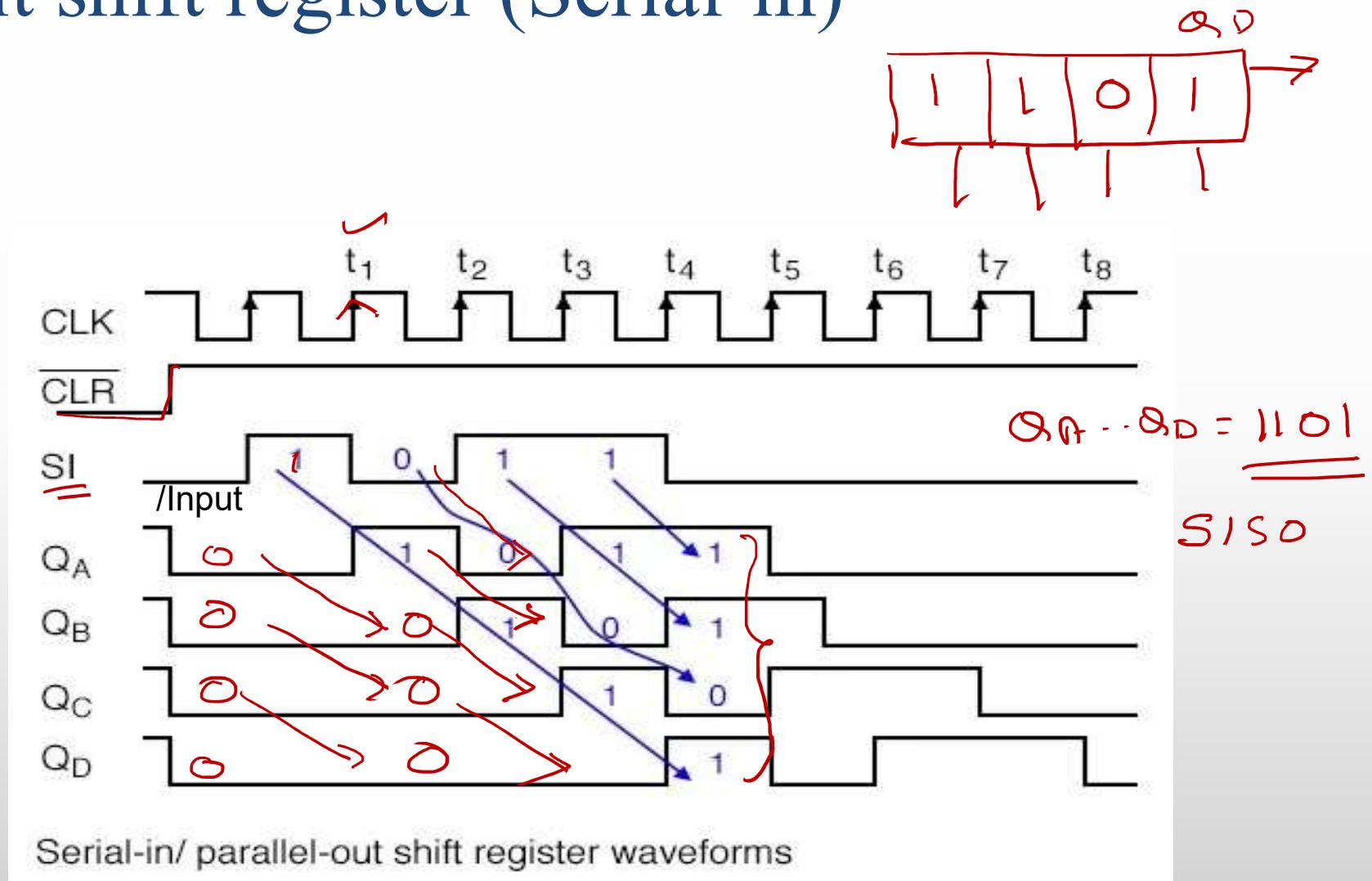


# SISO and SIPO Register

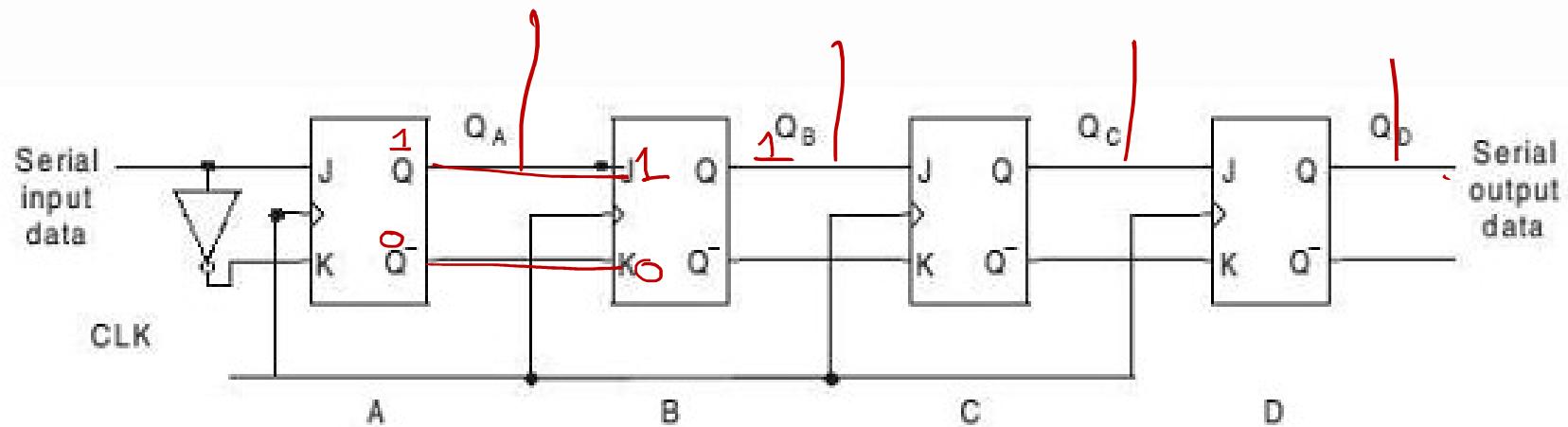
- Simple 4-bit shift register is shown below (Reset is nothing but clear input)
- Register is cleared using reset/clear input.
- It can be used as Serial-in serial out (at Q<sub>D</sub>) and Serial-in parallel out shift register.



# 4-bit shift register (Serial-in)

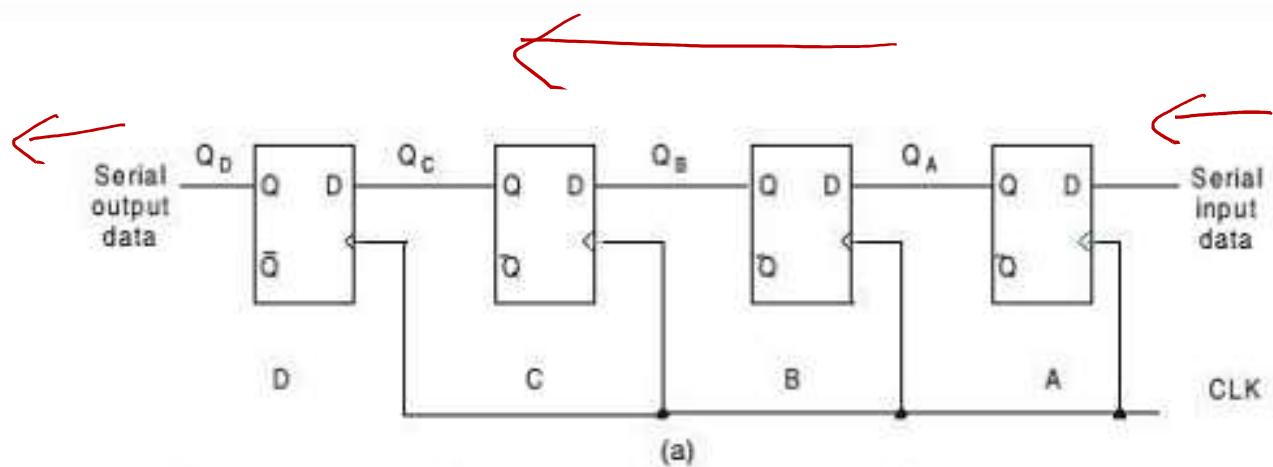


# SISO and SIPO Register using JK ffs

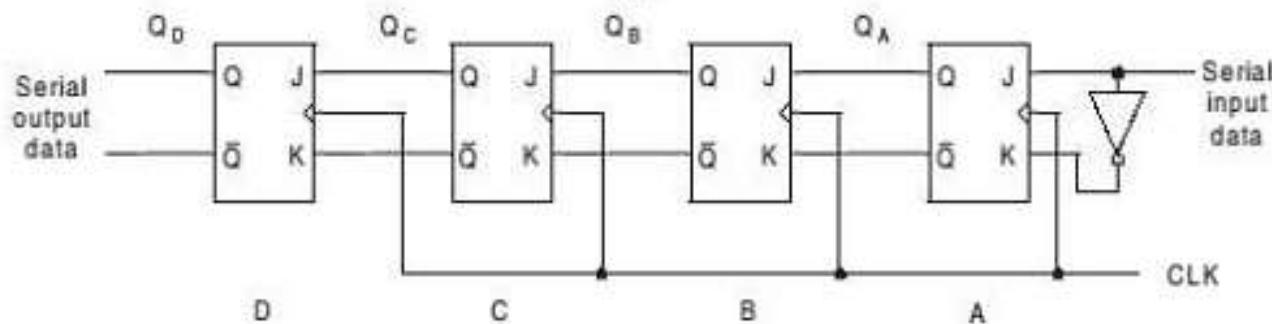


*SR-ffs*

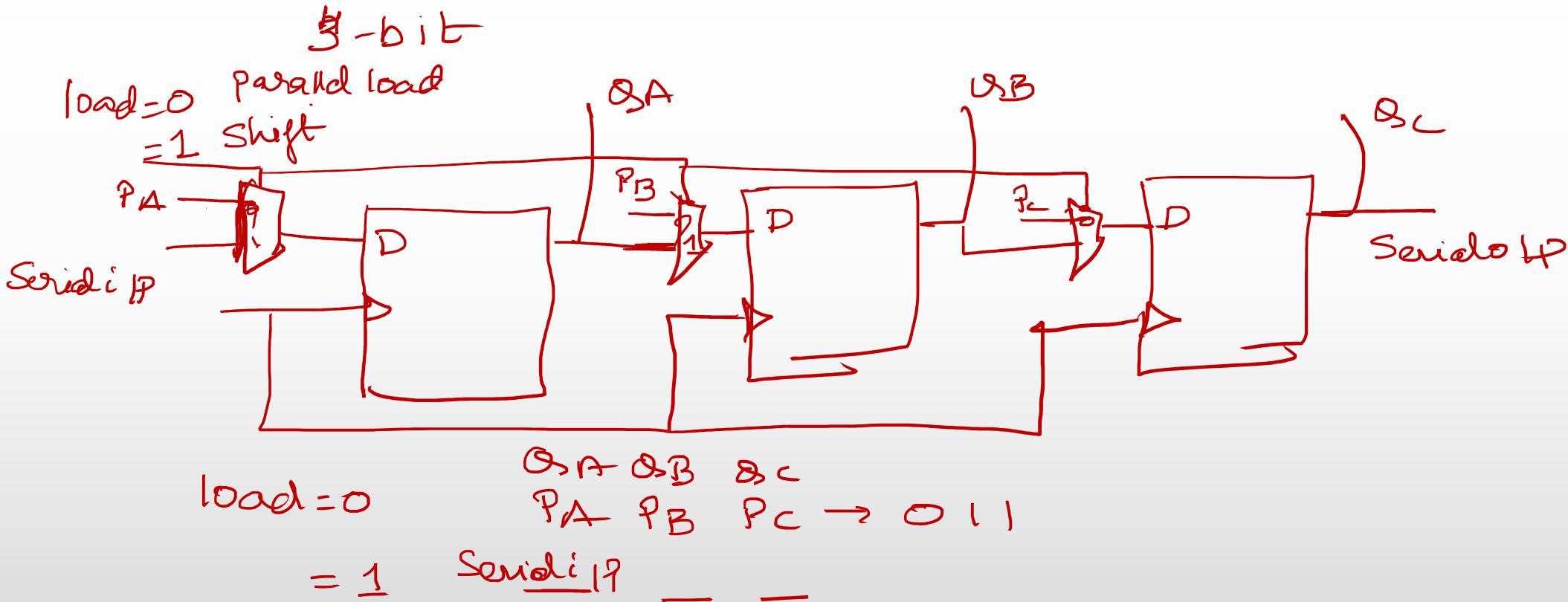
# SISO and SIPO register with shift left



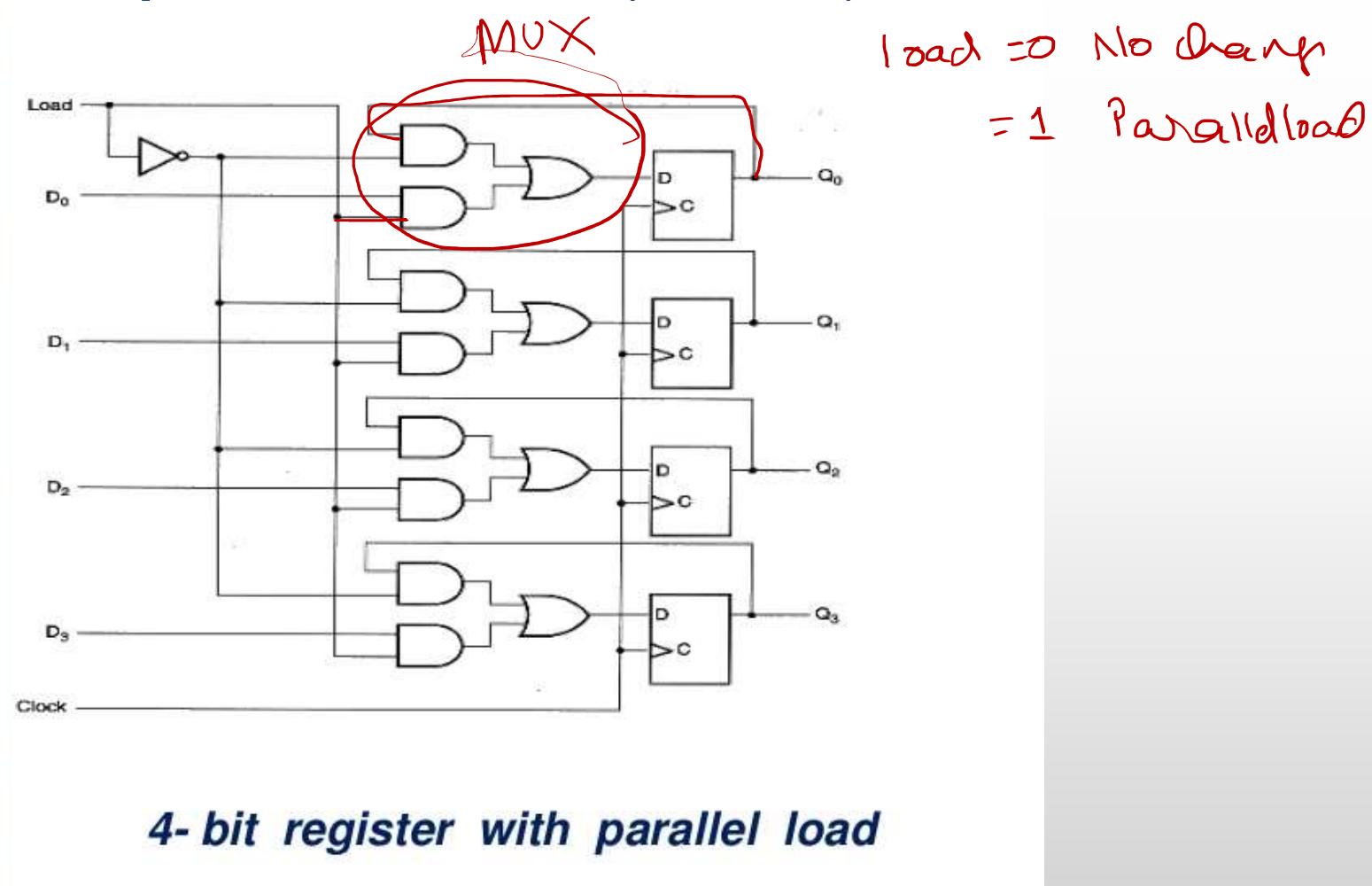
(a)



# Register with parallel load (PISO or PIPO)



# Register with parallel load (PIPO)

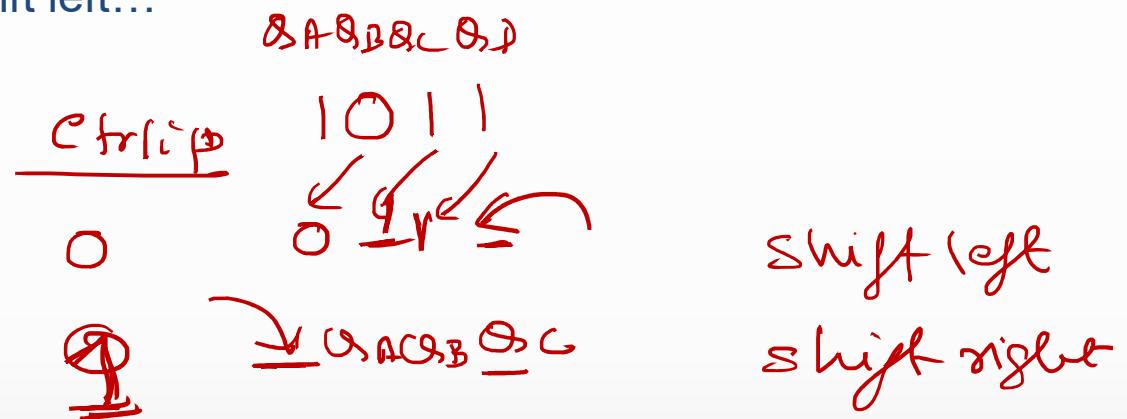


## 4-bit bidirectional shift register using D/SR/JK ffs

- If shift left/right =0, shift right else shift left...

- Solve it by yourself...



# 4-bit universal shift register

Mode Control		Register Operation
S1	S0	
0	0	No change
0	1	Shift right
1	0	Shift left
1	1	Parallel load

$Q_A Q_B Q_C Q_D$

$D_A D_B D_C D_D$

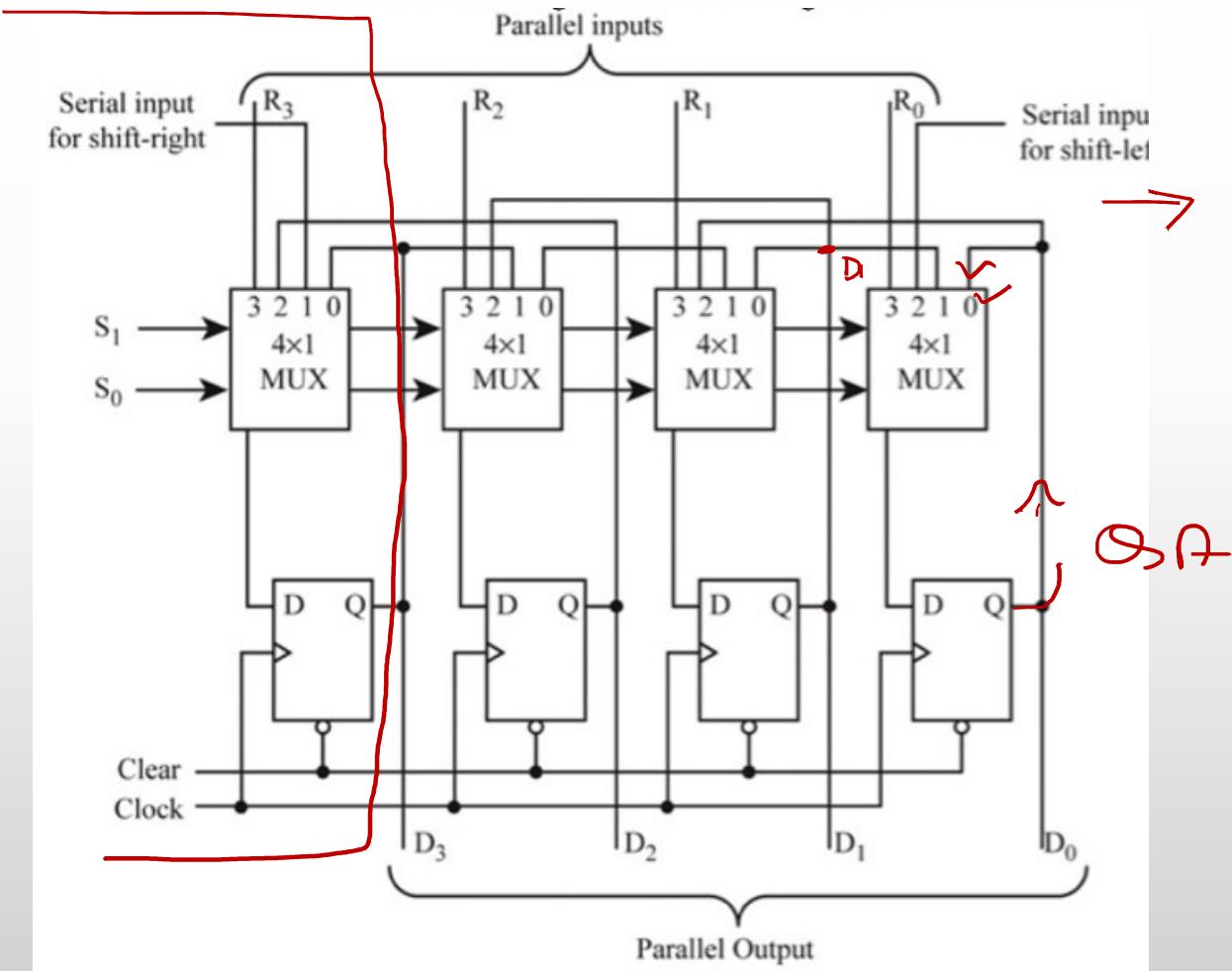
$\underline{Q_A} \underline{Q_B} \underline{Q_C} \underline{Q_D}$

$\underline{Q_A} \underline{Q_B} \underline{Q_C}$

$\underline{Q_B} \underline{Q_C} \underline{Q_D}$

$P_A P_B P_C P_D$

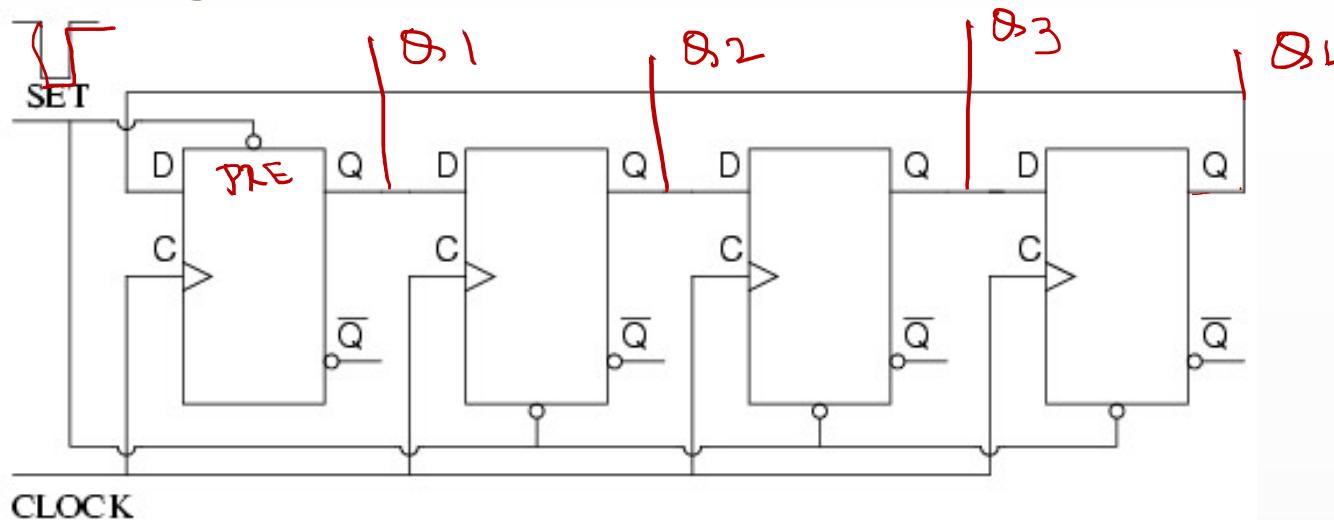
# 4-bit universal shift register



# Shift register counters:

- Ring counter
- Johnson counter /Twisted-ring counter

# Ring counter



right  
SISO - ~~gate~~

MOD 4 Ring  
counter

Set one stage, clear three stages

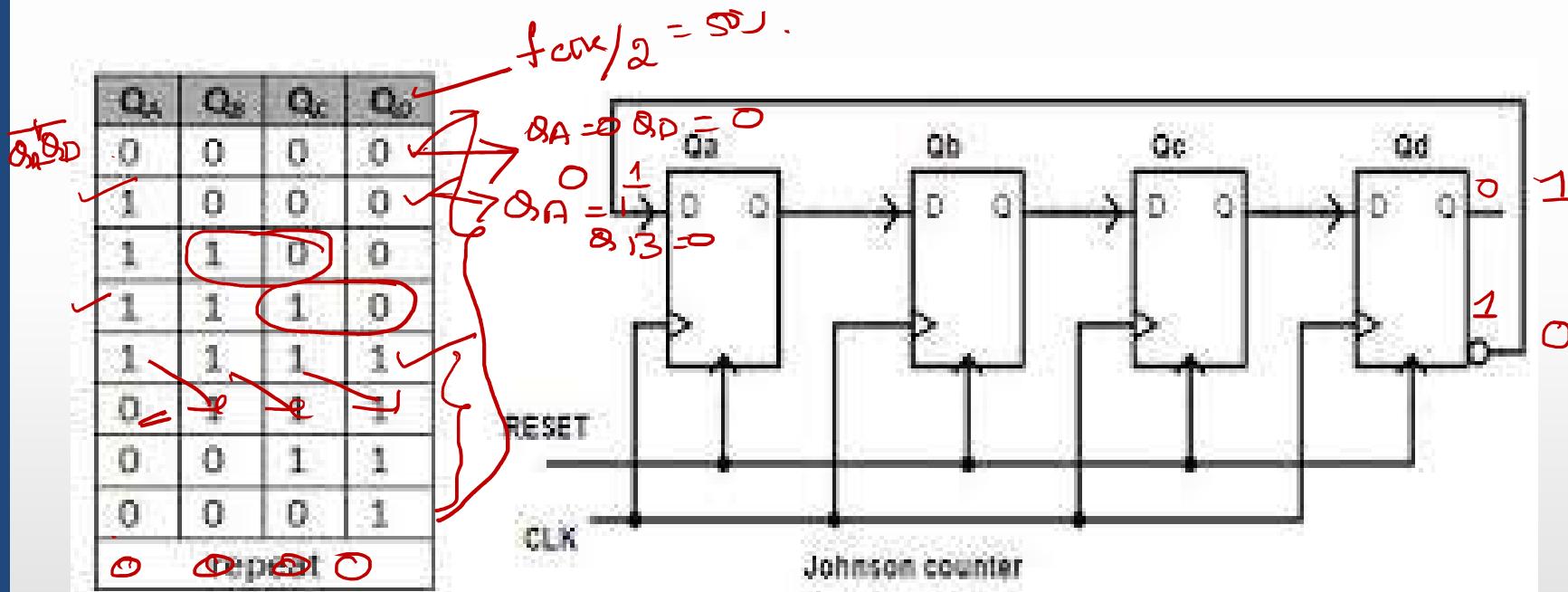
N-bit Ring counter  
MOD-N

Clock Cycle	Q <sub>1</sub>	Q <sub>2</sub>	Q <sub>3</sub>	Q <sub>4</sub>
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	0	0	0
6	0	1	0	0

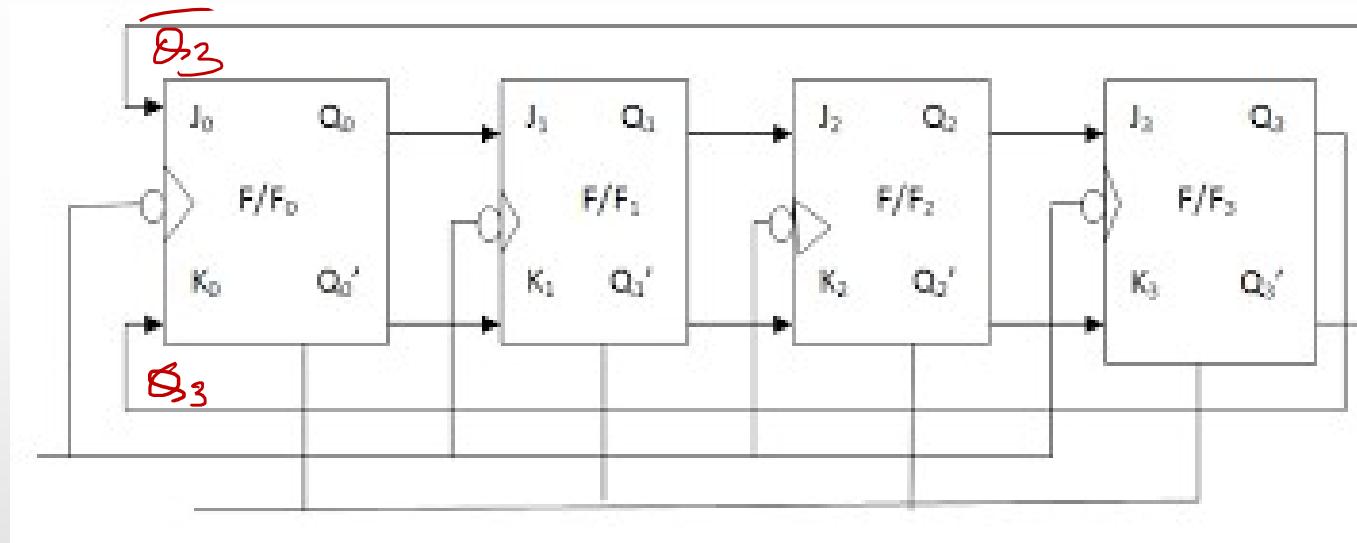
Bit-pattern repeats for every 4 clock cycles

# Johnson counter /Twisted-ring counter

$4 \text{ ffs} = 8 \text{ states} = \text{MOD } 8 \text{ counter}$



# Johnson counter /Twisted-ring counter using JK ffs



Design a sequence generator to generate the sequence "00100"  
using:

5 - 88

- a. Ring counter and external gates ✓
- b. Johnson counter and external gates  $\Rightarrow$  generate  
 $10101$   
 $3 \text{ yrs} \Rightarrow \text{MOD } 6$       ^      5-bits MOD 6

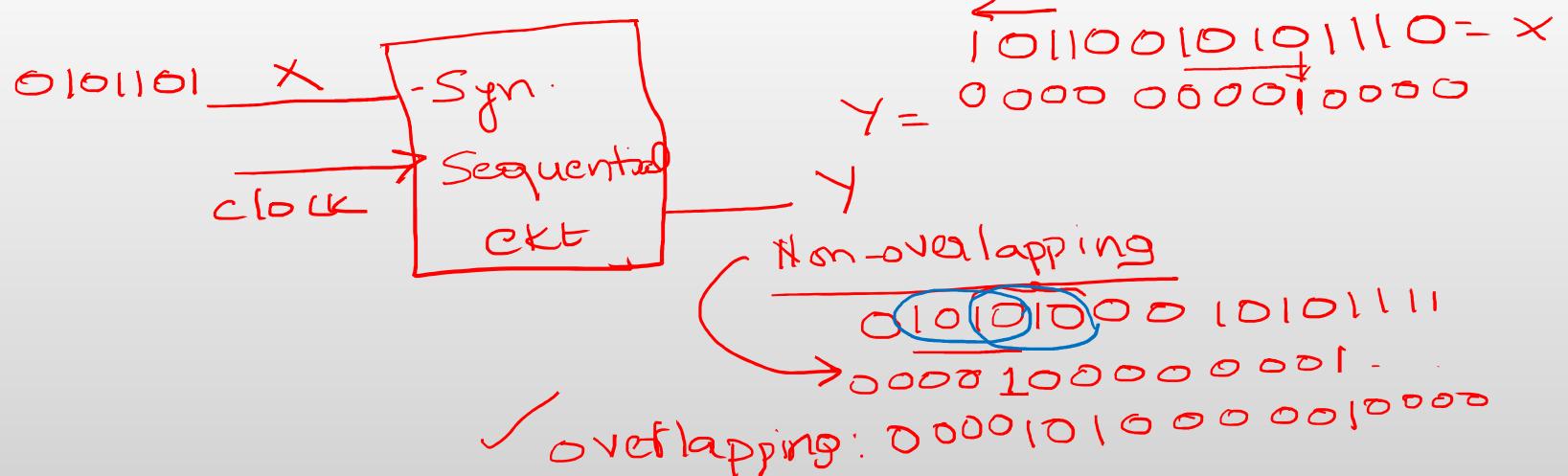
**For you to try !**

# SEQUENCE DETECTOR

# Sequence Detector

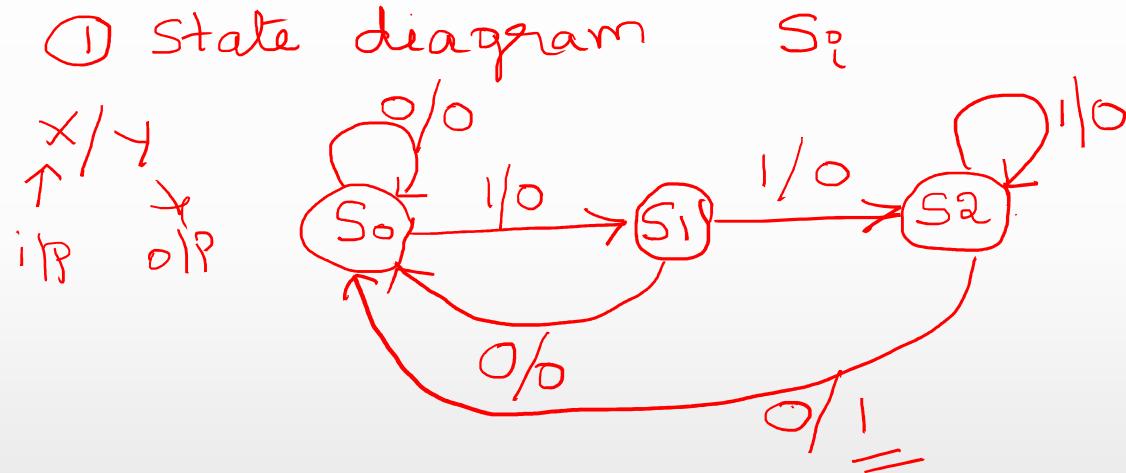
A sequence detector is a sequential circuit which takes string of bits as inputs and generates an output 1 whenever the target sequence has been detected.

- ✓ ■ Mealy Model: Output is a function of present state and input.
- ✓ ■ Moore model: Output is a function of the present state only.



# Mealy Model

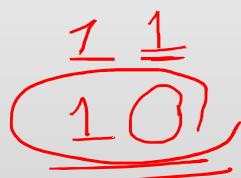
Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 110 and output 0 all other time. Use T Flip flops for the design.



abcde~~fghi~~j k

ghi

111110



$$S_2 = \underline{11} \underline{0} \quad \underline{11} \underline{0}$$
$$S_2 = \underline{\underline{11}} \underline{\underline{0}} + 1$$

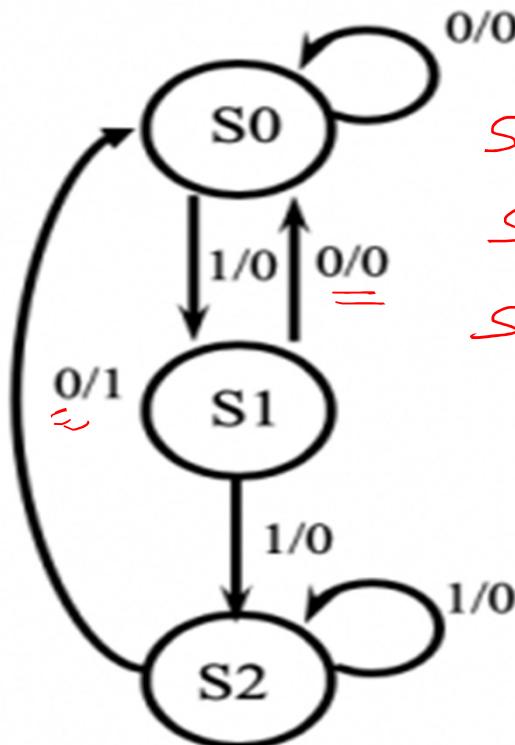
Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 110 and output 0 all other time. Use T Flip flops for the design.

Step 2: No. of states :  $s_0, s_1, s_2$ , 3 states, MOD 3 counter

$$\rightarrow \text{3 ffs}, \quad \begin{cases} s_0 = 00 \\ s_1 = 01 \\ s_2 = 10 \end{cases}$$

# Mealy Model

- Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 110 and output 0 all other time. Use T Flip flops for the design.



$$S_0 = 00$$

$$S_1 = 01$$

$$S_2 = 10$$

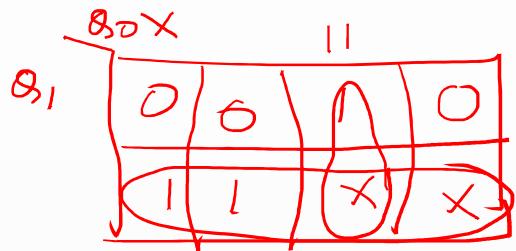
Present state  $\times$  Next state  $\xrightarrow{\quad}$  Expressions for  $T_1, T_0, Y$

$Q_1$	$Q_0$	$\text{Y}$	$Q_1^+$	$Q_0^+$	$T_1$	$T_0$	$\text{Y}$
0	0	0	0	0	$S_0$	0	0
0	0	1	0	1	$S_1$	1	0
0	1	0	0	0	$S_0$	1	0
0	1	1	1	0	$S_2$	0	0
1	0	0	0	0	$S_0$	1	1
1	0	1	1	0	$S_2$	1	0
1	1	0	X	X	X	X	X
1	1	1	X	X	X	X	X

$$T_1 =$$

$$T_0 =$$

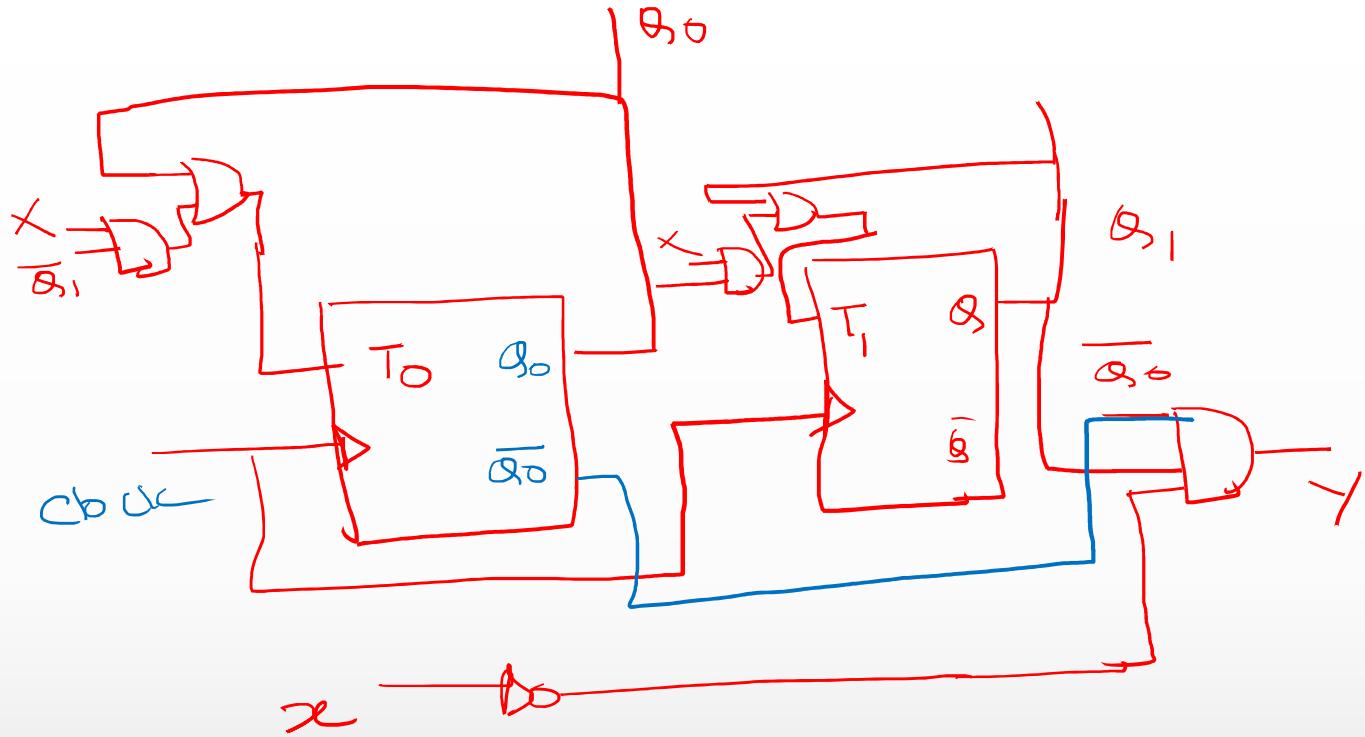
$$Y =$$



$$T_1 = Q_1 + Q_0 X$$

$$T_0 = Q_0 + \bar{Q}_1 X$$

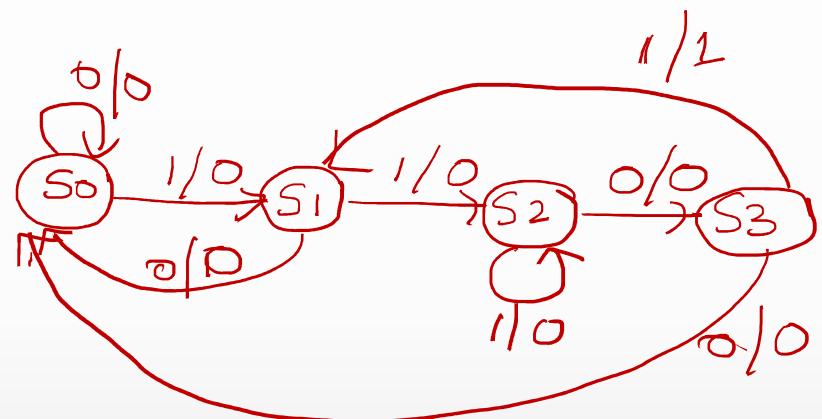
$$Y = Q_1 \bar{Q}_0 \bar{X}$$



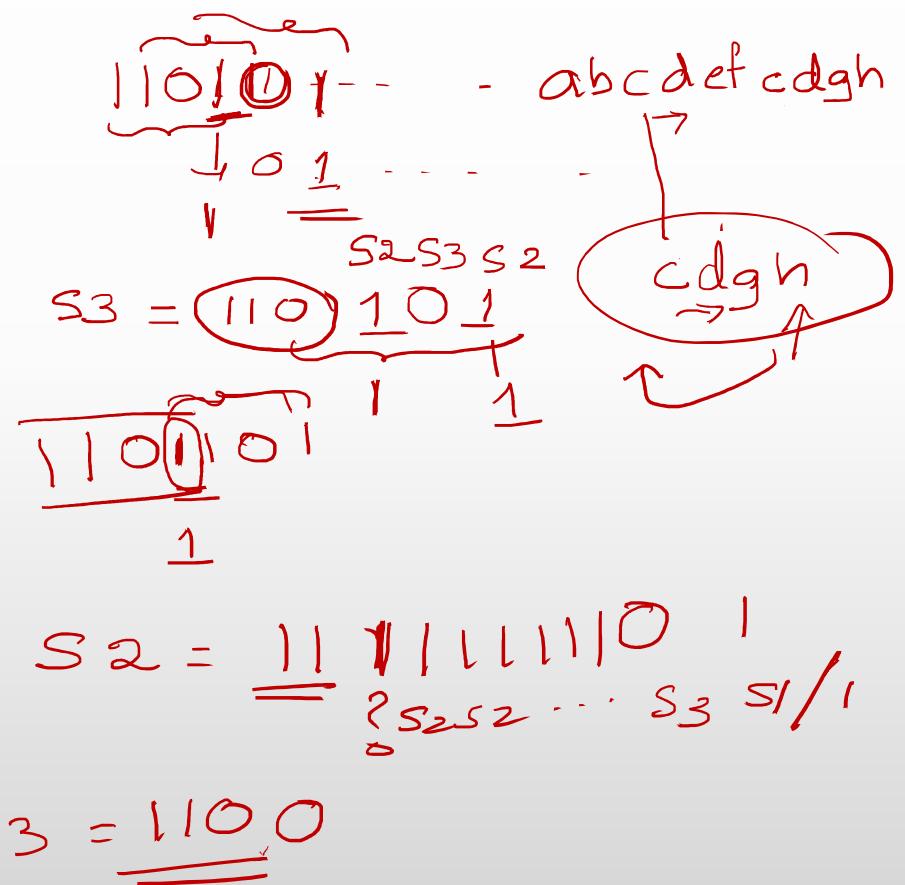
00 - 01 - 10

## Example 2

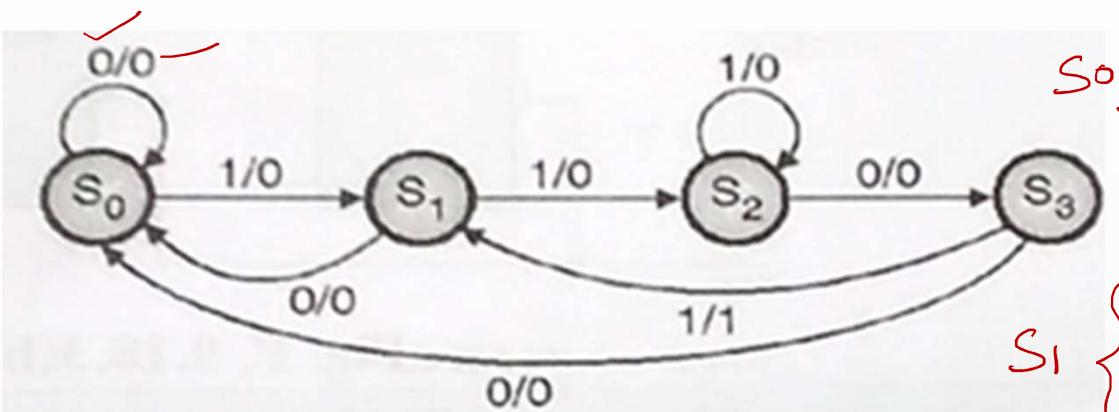
Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 1101 and output 0 all other time. Use JK Flip flops for the design.



$$\begin{array}{cccc} & s_3 & s_1 & s_2 \\ \underline{110} & \underline{110} & 1 & 1 \end{array}$$



- Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 1101 and output 0 all other time. Use JK Flip flops for the design.



$$S_0 = 00$$

$$S_1 = 01$$

$$S_2 = 10$$

$$S_3 = 11$$

$S_0 S_1 / JK$

$00 \rightarrow 0\phi$

$01 \rightarrow 1\phi$

$\phi 1 \rightarrow 10$

$11 \rightarrow \phi 0$

$S_2 \{$

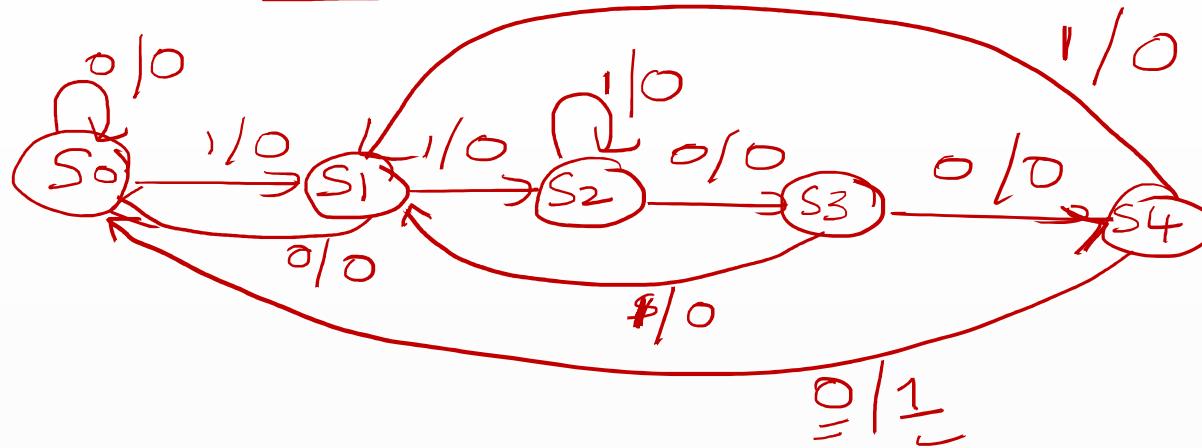
$S_3 \}$

$\gamma = S_1, S_0 X$

Q1	Q0	X	Q1 <sup>+</sup>	Q0 <sup>+</sup>	J1	K1	J0	K0	Y	
Present			Next state							
0	0	0	0	0	$S_0$	0	$\phi$	0	$\phi$	0
0	0	1	0	1	0	$\phi$	1	$\phi$	0	0
0	1	0	0	0	0	$\phi$	$\phi$	1	0	0
0	1	1	1	0	1	$\phi$	$\phi$	$\phi$	1	0
1	0	0	1	1	$\phi$	0	1	$\phi$	0	0
1	0	1	1	0	$\phi$	0	0	$\phi$	0	0
1	1	0	0	0	$\phi$	1	$\phi$	1	0	0
1	1	1	0	1	$\phi$	1	$\phi$	0	1	1

Write K-maps for  $J_1, K_1, J_0, K_0$  and get the simplified expressions

- Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 11000 and output 0 all other time. Use D Flip flops for the design.



5-bit sequence  $x$   
5 states

$$\begin{aligned} S_0 &= 0000 \\ S_1 &= 001 \\ S_2 &= 010 \\ S_3 &= 011 \\ S_4 &= 100 = \underline{\underline{Q}_2 \bar{Q}_1 \bar{Q}_0} \end{aligned}$$

$$S_4 = \underline{\underline{11000}}$$

$$S_2 = \underline{\underline{1100}}$$

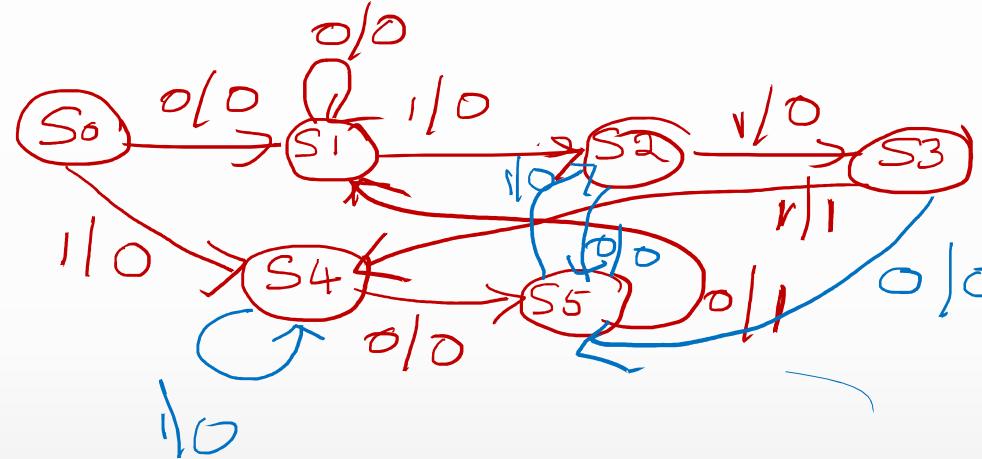
$$S_3 = \underline{\underline{110}} \underline{\underline{1}}$$

$$S_4 = \underline{\underline{1100}} \underline{\underline{1000}}$$

$$\begin{aligned} Y &= \text{if } S_4 \text{ is the present state } \& x = 0 \\ &= \underline{\underline{Q_2 \bar{Q}_1 \bar{Q}_0 \bar{x}}} \end{aligned}$$

Get simplified expressions for  $I_2$ ,  $I_3$  & complete the circuit

- Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 0111 or 100 and output 0 all other time. Use D Flip flops for the design.



$$S_5 = \underline{101}$$

$$S_4 = 11$$

$$\begin{aligned} S_1 &= 0\underline{0} \\ S_2 &= \underline{010} \end{aligned}$$

$$S_3 = \underline{01100}$$

6 states = MOD 6 = 3-bit  $\alpha\beta\gamma$

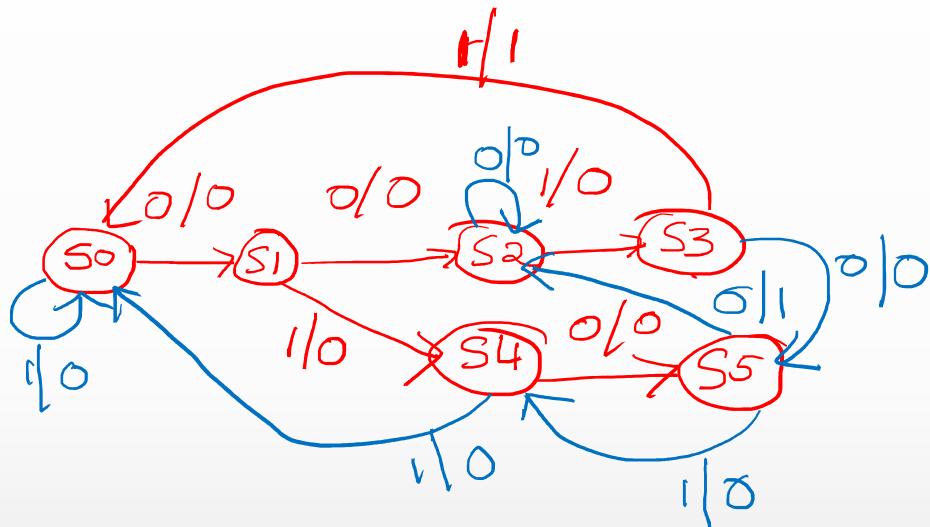
$\gamma = (\text{case i}) \text{ or } (\text{case ii})$

$$\begin{aligned} &= (S_3 \text{ state }, x=1) \text{ or } (S_5 \text{ state }, x=0) \\ &= \underline{\alpha_2}\underline{\alpha_1}\underline{\alpha_0}x + \underline{\alpha_2}\underline{\alpha_1}\underline{\alpha_0}\bar{x} \end{aligned}$$

$$\begin{aligned} X &= 011100111001110 \\ Y &= 00010100010000 \end{aligned}$$

$$\begin{aligned} S_0 &- 000 \\ S_1 &- 001 \\ S_2 &- 010 \\ S_3 &- 011-\underline{\alpha_2}\underline{\alpha_1}, \underline{\alpha_0} \\ S_4 &- 100 \\ S_5 &- 101 \end{aligned}$$

- Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 0011 or 0100 and output 0 all other time. Use JK Flip flops for the design.



$$S_4 = \underline{\underline{011}}$$

$$S_5 = \underline{\underline{010}}$$

$$S_5 = \underline{\underline{0100}}$$

$$S_2 = \underline{\underline{000}} \quad 0 \quad \underline{\underline{11}}$$
  

$$\underline{\underline{000000}} \quad \underline{\underline{11}}$$

$$S_3 = \underline{\underline{0010}}$$

## Excercises

- ① state diagram: 1111, 0000, 1010
- ② " : @ 1100 and 0011
- ③ 1010 and 1011

# 5-VARIABLE K-MAPS AND TABULATION METHOD

## 5 variable K-Maps (Method I): Layered structure

$F(V,W,X,Y,Z)$ , V is the MSB and Z is the LSB

$00000 - 0111$   
 $V=0 (m_0 \text{ to } m_5)$

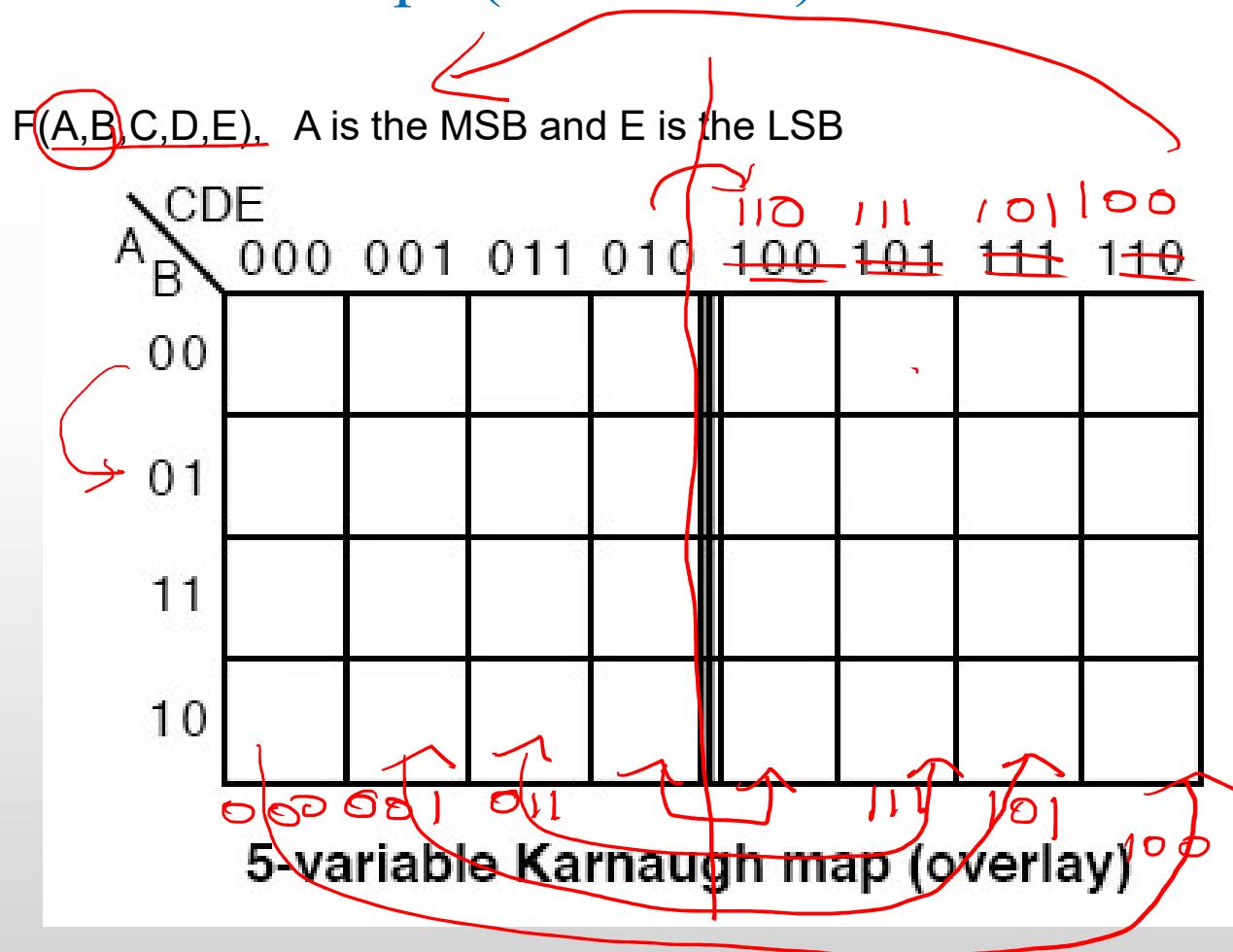
	Y	Z	00	01	11	10
W	X		$m_0$	$m_1$	$m_3$	$m_2$
00			$m_4$	$m_5$	$m_7$	$m_6$
01			$m_{12}$	$m_{13}$	$m_{15}$	$m_{14}$
11			$m_8$	$m_9$	$m_{11}$	$m_{10}$
10						

$10000 - 1111$   
 $V=1 (m_{16} \text{ to } m_{31})$

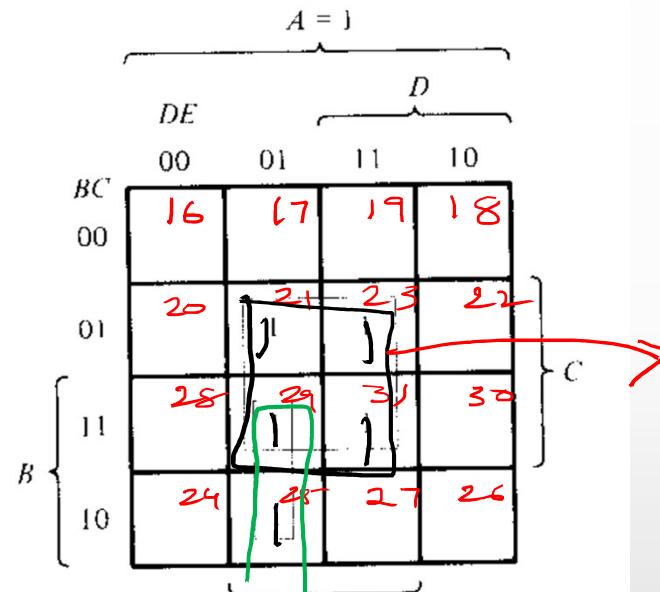
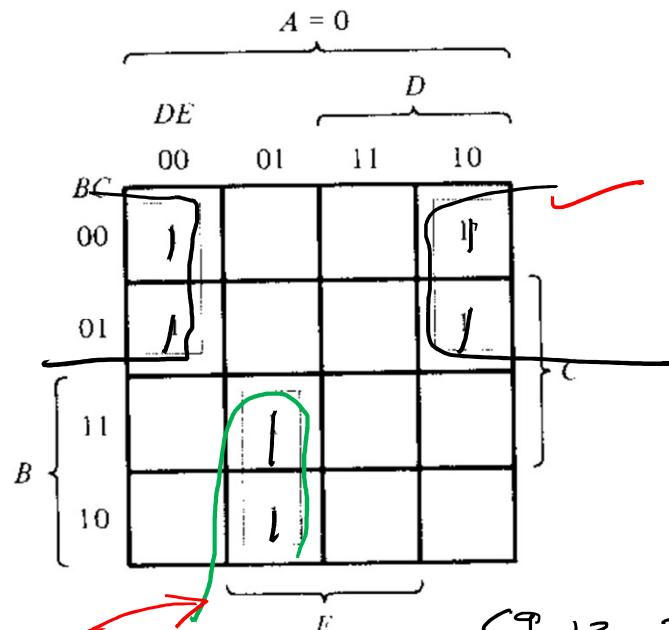
	Y	Z	00	01	11	10
W	X		$m_{16}$	$m_{17}$	$m_{19}$	$m_{18}$
00			$m_{20}$	$m_{21}$	$m_{23}$	$m_{22}$
01			$m_{28}$	$m_{29}$	$m_{31}$	$m_{30}$
11			$m_{24}$	$m_{25}$	$m_{27}$	$m_{26}$
10						

Method I is used for all the examples

## 5 variable K-Maps (Method II): Reflective structure



Solved Example:  $F(A,B,C,D,E) = \sum m(0,2,4,6,9,13,21,23,25,29,31)$

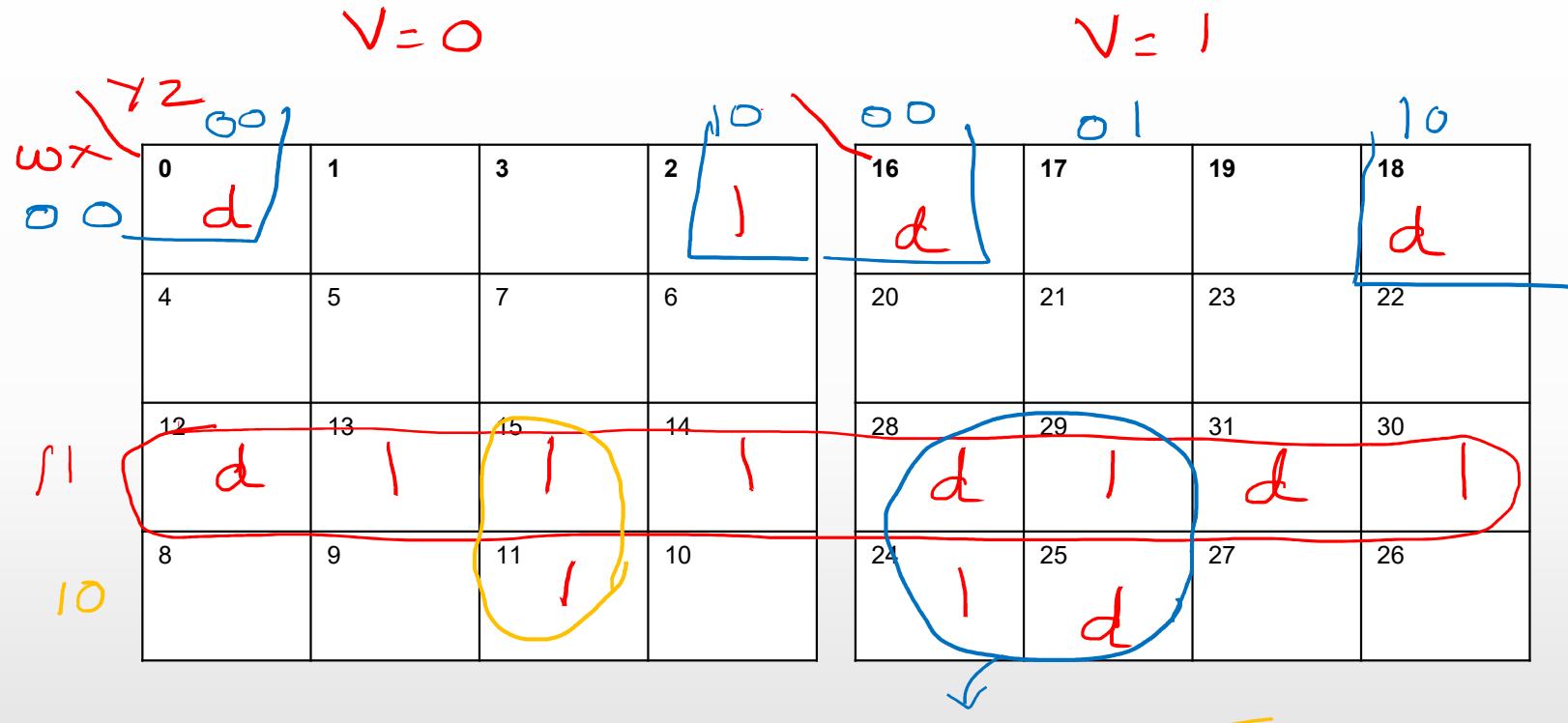


$$(0, 2, 4, 6, 9, 13, 21, 23, 25, 29) - \text{Quad } E = B\bar{C}D$$

$$F = B\bar{C}D + \bar{A}\bar{B}\bar{E} + ACE$$

Example1 :

$$F(v,w,x,y,z) = \sum m(2,11,13,14,15,24,29,30) + \sum d(0,12,16,18,25,28,31)$$



$$F = wx + \bar{w}\bar{x}\bar{z} + v\bar{w}\bar{y} + \bar{v}wyz$$

Example2 :

$$F(v,w,x,y,z) = \Pi M(4,5,9,11,12,14,15,27,30) \cdot D(1,17,25,26,28,31)$$

$V=0$								$V=1$																		
0	1	3	2	16	17	19	18	0	11	10	20	21	23	22	28	29	31	30	27	26	24	25	2	1	0	10
			<i>d</i>												<i>d</i>			<i>d</i>								
				0	1	3	2																			
				4	5	7	6																			
				12	13	15	14																			
				8	9	11	10																			

Annotations:

- $\bar{w}x\bar{y}$  (purple) covers rows 0, 1, 3, 2.
- $\bar{w}xz\bar{y}$  (yellow) covers columns 4, 5, 7, 6.
- $\bar{w}\bar{x}z$  (orange) covers row 12, column 13.
- $\bar{w}\bar{x}z$  (orange) covers row 14, column 15.
- $\bar{w}xy$  (purple) covers columns 28, 29, 31, 30.
- $\bar{w}\bar{x}y$  (red) covers row 9, column 10.
- $\omega x z$  (red) covers row 25, column 27.

$$F = \text{pos}$$

$$F = \text{pos} = (\bar{\omega} + x + \bar{z})(\bar{\omega} + \bar{x} + \bar{y})(\bar{\omega} + \bar{x} + z)(V + \bar{\omega} + \bar{x} + y)$$

Example 3 :

$$F(A, B, C, D, E) = \overline{A'B'CE'} + \overline{A'B'C'D'} + \overline{B'D'E'} + \overline{B'CD'} + \overline{CDE'} + \overline{BDE'}$$

Write the SOP expression

$$\begin{array}{r} 000000 \oplus 000001 \\ \hline 0, 4, 16, 20 \\ \hline A = 0 \\ A = 1 \end{array}$$

		DE		A = 0	
		B	C	D	E
B		00	01	11	10
00	01	1	1	0	0
01	10	1	1	0	0
10	11	0	0	1	1
11	10	0	0	1	0
10	01	0	0	0	1
01	00	0	0	0	0

		DE		A = 1	
		B	C	D	E
B		00	01	11	10
16	17	1	0	0	0
20	21	1	1	0	0
23	24	0	0	1	1
28	29	0	0	1	0
30	31	0	0	1	1
24	25	0	0	0	1
27	26	0	0	1	0

$$A = \overline{B} \overline{D} \overline{E}$$

$$\overline{B} \overline{C} \overline{D} =$$

$$CD\bar{E} = 14,$$

$$(4, 6, 20, 22)$$

$$\overline{B} \overline{C} \overline{E}$$

redundant

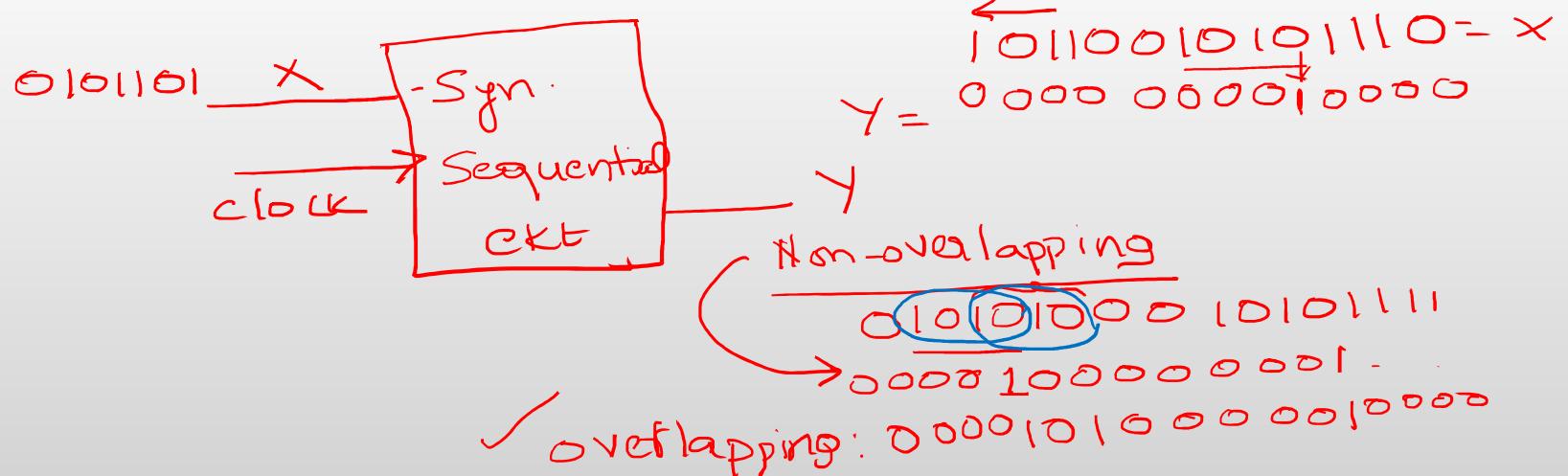
$$\begin{aligned} F = & CD\bar{E} + BD\bar{E} + \overline{B}C\bar{D} + \overline{B}\bar{D}\bar{E} + \\ & + \overline{AB}\bar{D} + \cancel{\overline{ABC}} \end{aligned}$$

# SEQUENCE DETECTOR

# Sequence Detector

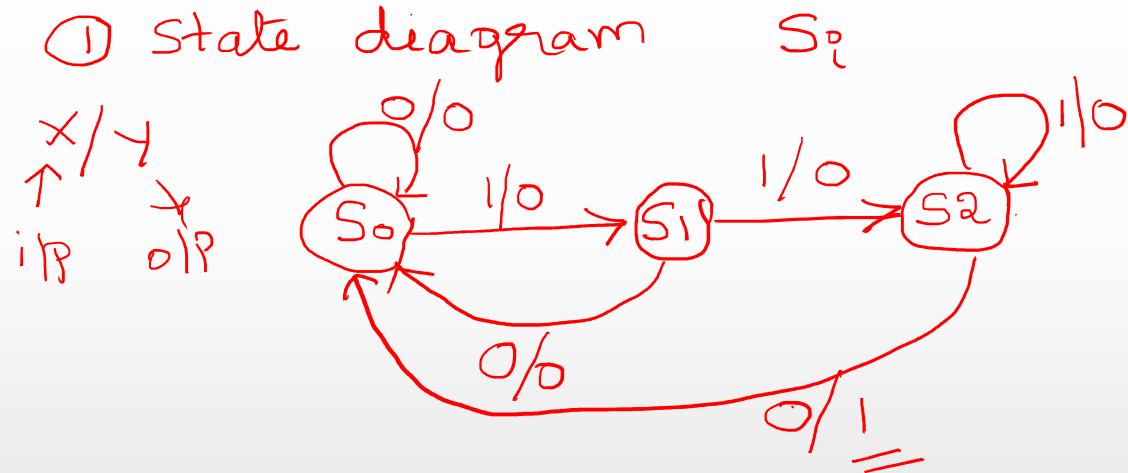
A sequence detector is a sequential circuit which takes string of bits as inputs and generates an output 1 whenever the target sequence has been detected.

- ✓ ■ Mealy Model: Output is a function of present state and input.
- ✓ ■ Moore model: Output is a function of the present state only.



# Mealy Model

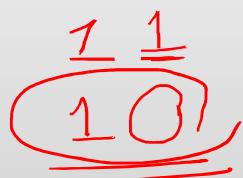
Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 110 and output 0 all other time. Use T Flip flops for the design.



abcde~~fghi~~j k

ghi

111110



$$S_2 = \underline{110} \quad \underline{110}$$
$$S_2 = \underline{\underline{11}} \quad \underline{\underline{10}}$$

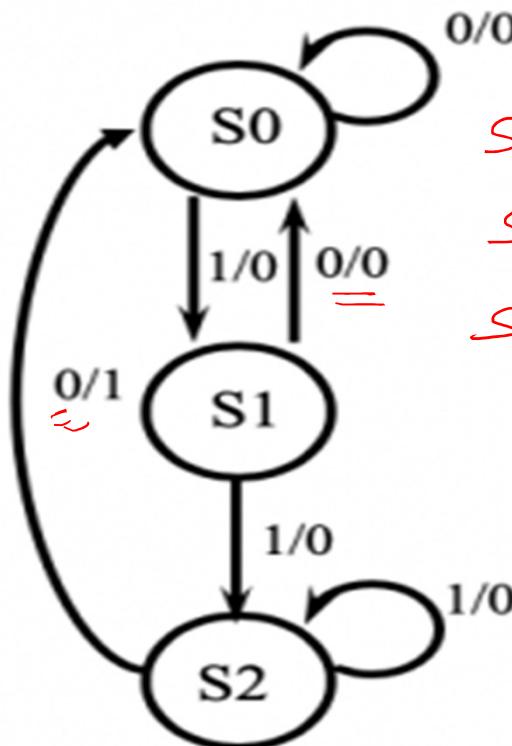
Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 110 and output 0 all other time. Use T Flip flops for the design.

Step 2: No. of states :  $s_0, s_1, s_2$ , 3 states, MOD 3 counter

$$\rightarrow \text{3 ffs}, \quad \begin{cases} s_0 = 00 \\ s_1 = 01 \\ s_2 = 10 \end{cases}$$

# Mealy Model

- Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 110 and output 0 all other time. Use T Flip flops for the design.



$$S_0 = 00$$

$$S_1 = 01$$

$$S_2 = 10$$

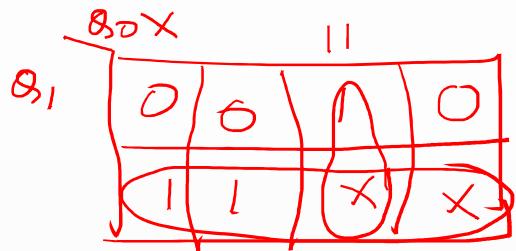
Present state  $\times$       Next State  $\downarrow$       Expressions for  $T_1, T_0, Y$

$Q_1$	$Q_0$	$\text{Y}$	$Q_1^+$	$Q_0^+$	$T_1$	$T_0$	$\text{Y}$
0	0	0	0	0	$S_0$	0	0
0	0	1	0	1	$S_1$	1	0
0	1	0	0	0	$S_0$	1	0
0	1	1	1	0	$S_2$	0	0
1	0	0	0	0	$S_0$	1	1
1	0	1	1	0	$S_2$	1	0
1	1	0	X	X	X	X	X
1	1	1	X	X	X	X	X

$$T_1 =$$

$$T_0 =$$

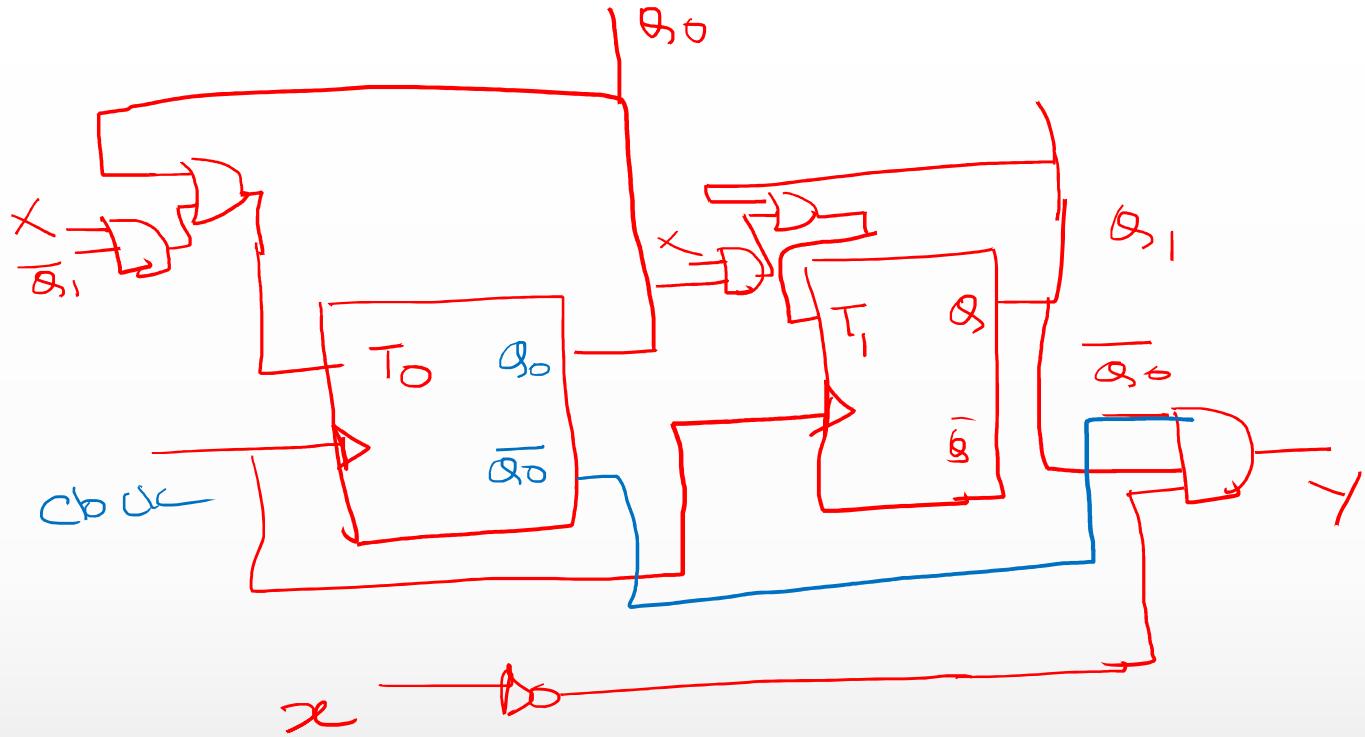
$$Y =$$



$$T_1 = \theta_1 + \theta_0 X$$

$$T_0 = \theta_0 + \bar{\theta}_1 X$$

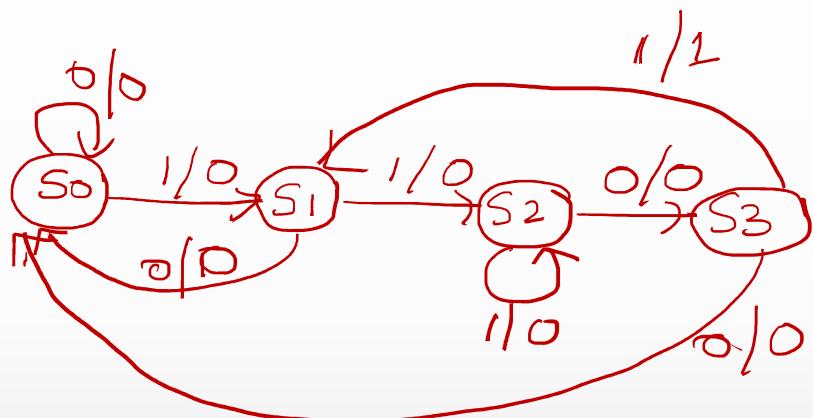
$$Y = \theta_1 \bar{\theta}_0 \bar{X}$$



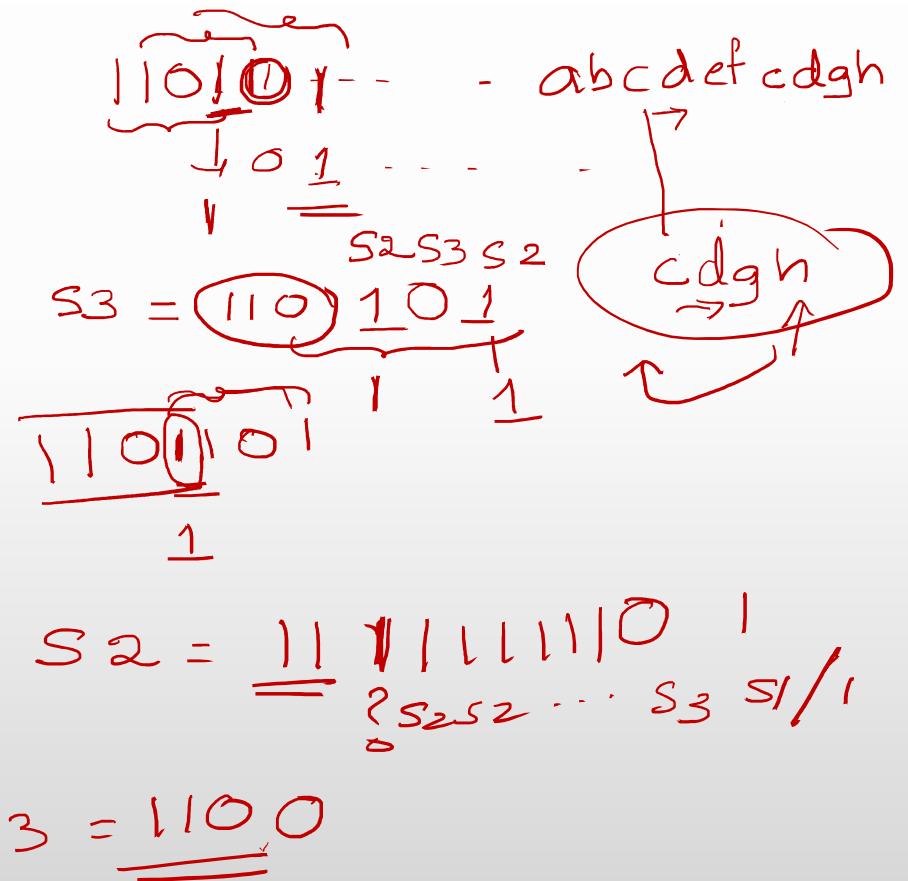
00 - 01 - 10

## Example 2

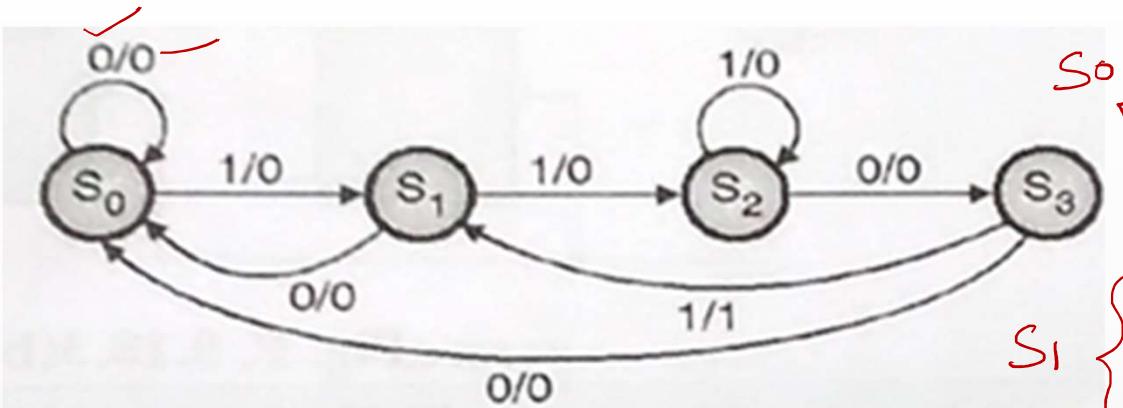
Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 1101 and output 0 all other time. Use JK Flip flops for the design.



$$\begin{array}{c}
 S_3 \quad S_1 \quad S_2 \quad S_3 \quad S_1 \\
 \underline{\quad 1 \quad} \quad \underline{\quad 1 \quad} \quad \underline{\quad 0 \quad} \quad \underline{\quad 1 \quad} \quad \underline{\quad 1 \quad} \\
 \underline{\quad 1 \quad} \quad \underline{\quad 1 \quad} \quad \underline{\quad 0 \quad} \quad \underline{\quad 1 \quad} \quad \underline{\quad 1 \quad}
 \end{array}$$



- Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 1101 and output 0 all other time. Use JK Flip flops for the design.



$$S_0 = 00$$

$$S_1 = 01$$

$$S_2 = 10$$

$$S_3 = 11$$

$Q_1 Q_0 + / J K$

$00 \rightarrow 0\phi$

$01 \rightarrow 1\phi$

$10 \rightarrow \phi 1$

$11 \rightarrow \phi 0$

$S_2 \{$

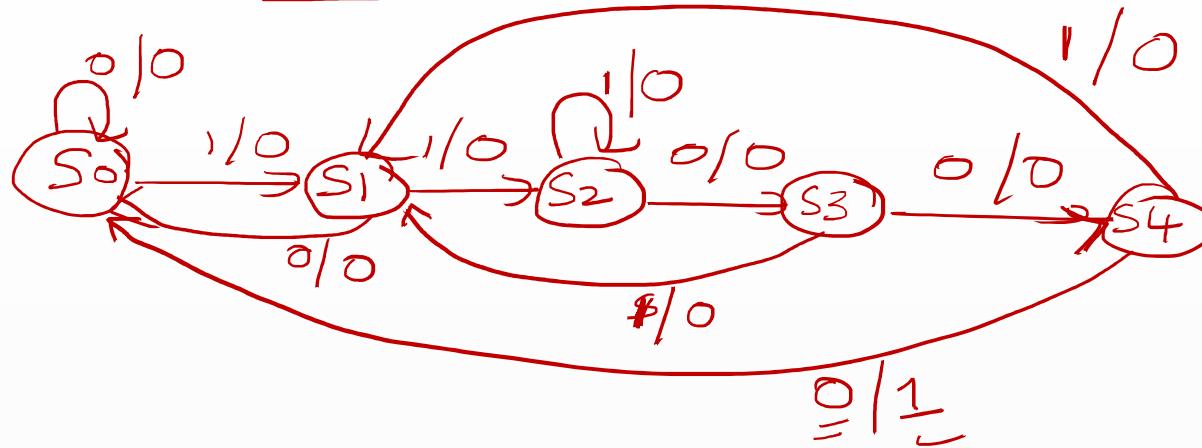
$S_3 \}$

$\gamma = 0, 00 X$

$\overline{Q_1}$	$\overline{Q_0}$	<del>X</del>	$\overline{Q_1^+}$	$\overline{Q_0^+}$	J1	K1	J0	K0	$\gamma$
Present			Next State						
0	0	0	0	0	$S_0$	0	$\phi$	$\phi$	0
0	0	1	0	1	0	$\phi$	1	$\phi$	0
0	1	0	0	0	0	$\phi$	$\phi$	1	0
0	1	1	1	0	1	$\phi$	$\phi$	1	0
1	0	0	1	1	$\phi$	0	1	$\phi$	0
1	0	1	1	0	$\phi$	0	0	$\phi$	0
1	1	0	0	0	$\phi$	1	$\phi$	1	0
1	1	1	0	1	$\phi$	1	$\phi$	0	1

Write K-maps for  $J_1, K_1, J_0, K_0$  and get the simplified expressions

- Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 11000 and output 0 all other time. Use D Flip flops for the design.



5-bit sequence  $x$   
5 states

$$\begin{aligned} S_0 &= 0000 \\ S_1 &= 001 \\ S_2 &= 010 \\ S_3 &= 011 \\ S_4 &= 100 = \underline{\underline{Q}_2 \bar{Q}_1 \bar{Q}_0} \end{aligned}$$

$$S_4 = \underline{\underline{11000}}$$

$$S_2 = \underline{\underline{110}}$$

$$S_3 = \underline{\underline{110}} \underline{\underline{1}}$$

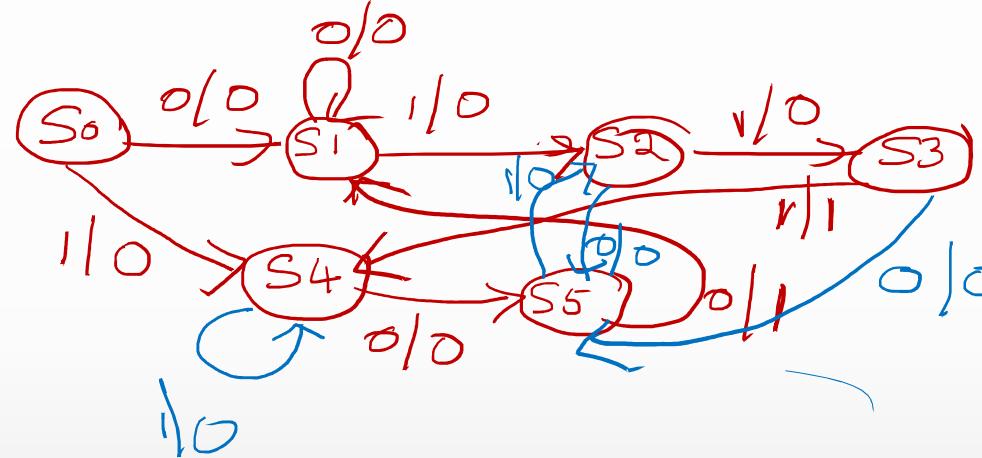
$$S_4 = \underline{\underline{1100}} \underline{\underline{1000}}$$

$$Y = \text{if } S_4 \text{ is the present state } \& x = 0$$

$$= \underline{\underline{Q_2 \bar{Q}_1 \bar{Q}_0 \bar{x}}} \Rightarrow$$

Get simplified expressions for  $I_2$ ,  $I_3$  & complete the circuit

- Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 0111 or 100 and output 0 all other time. Use D Flip flops for the design.



$$S_5 = \underline{101}$$

$$S_4 = 11$$

$$S_3 = \underline{01100}$$

6 states = MOD 6 = 3-bit  $\alpha\beta\gamma$

$\gamma = (\text{case i}) \text{ or } (\text{case ii})$

$$\begin{aligned}
 &= (S_3 \text{ state }, \alpha=1) \text{ or } (S_5 \text{ state }, \alpha=0) \\
 &= \underline{\alpha_2\alpha_1\alpha_0}\alpha + \underline{\alpha_2\bar{\alpha}_1\bar{\alpha}_0}\bar{\alpha}
 \end{aligned}$$

$$\begin{aligned}
 X &= 01110011001100 \\
 Y &= 00010100010000
 \end{aligned}$$

$$S_0 = 000$$

$$S_1 = 001$$

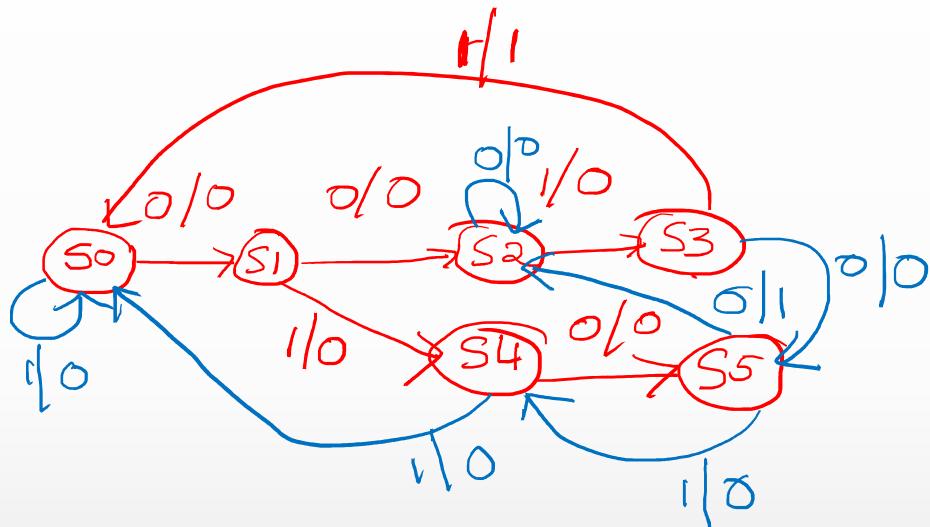
$$S_2 = 010$$

$$S_3 = 011 \quad \underline{\alpha_2\alpha_1\alpha_0}$$

$$S_4 = 100$$

$$S_5 = 101$$

- Design a sequential circuit using mealy model which produces an output 1 whenever it detects a sequence 0011 or 0100 and output 0 all other time. Use JK Flip flops for the design.



$$S_4 = \underline{\underline{011}}$$

$$S_5 = \underline{\underline{010}}$$

$$S_5 = \underline{\underline{0100}}$$

$$S_2 = \underline{\underline{00}} \quad 0 \quad \underline{\underline{11}}$$
  

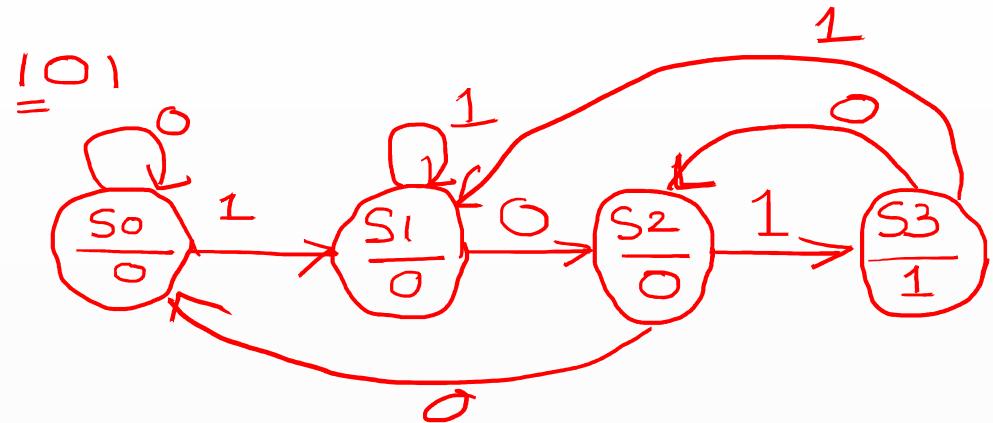
$$\underline{\underline{000000}} \quad \underline{\underline{11}}$$

$$S_3 = \underline{\underline{0010}}$$

## Excercises

- ① state diagram: 1111, 0000, 1010
- ② " : @ 1100 and 0011
- ③ 1010 and 1011

# Moore Model

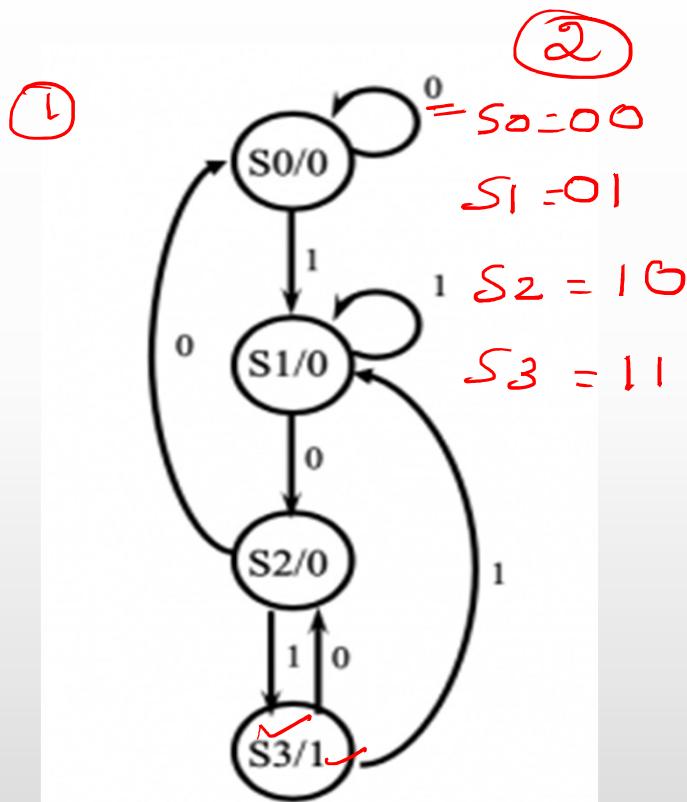


$$S_3 = \underline{1} \underline{0} \underline{1} \underline{0}$$
$$S_3 = \underline{1} \underline{0} \underline{1} \underline{1}_9$$

4 states = 2 ffs

# Moore Model

- Design a sequential circuit using moore model which produces an output 1 whenever it detects a sequence 101 and output 0 all other time. Use T Flip flops for the design.



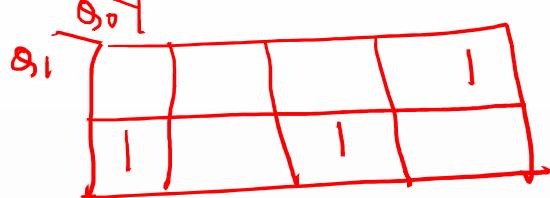
*Present*    *Input*    *Next*

Q <sub>1</sub>	Q <sub>0</sub>	Y	Q <sub>1</sub> <sup>+</sup>	Q <sub>0</sub> <sup>+</sup>	T <sub>1</sub>	T <sub>0</sub>	Z
0	0	0	0	0	<i>S<sub>0</sub></i>	0	0
0	0	1	0	1	<i>S<sub>1</sub></i>	0	1
0	1	0	1	0	<i>S<sub>2</sub></i>	1	0
0	1	1	0	1	<i>S<sub>1</sub></i>	0	0
1	0	0	0	0	<i>S<sub>0</sub></i>	1	0
1	0	1	1	1	<i>S<sub>3</sub></i>	0	1
1	1	0	1	0	<i>S<sub>2</sub></i>	0	1
1	1	1	0	1	<i>S<sub>1</sub></i>	1	0

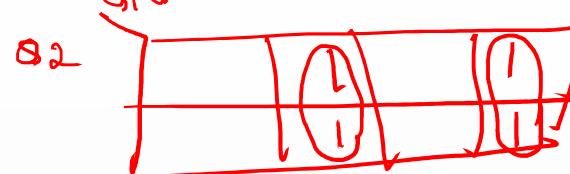
Annotations on the table:

- Braces on the left side group rows by state:  $S_0$ ,  $S_1$ ,  $S_2$ ,  $S_3$ .
- Red circled values:  $S_0$  in the  $Q_1$  column of row 1,  $S_1$  in the  $Q_0$  column of row 2,  $S_2$  in the  $Q_1$  column of row 3,  $S_3$  in the  $Q_0$  column of row 4.
- Red underlined values:  $S_0$  in the  $Q_0$  column of row 1,  $S_1$  in the  $Q_1$  column of row 2,  $S_2$  in the  $Q_0$  column of row 3,  $S_3$  in the  $Q_1$  column of row 4.
- Red crossed-out values:  $S_0$  in the  $Q_1$  column of row 5,  $S_1$  in the  $Q_0$  column of row 6,  $S_2$  in the  $Q_1$  column of row 7,  $S_3$  in the  $Q_0$  column of row 8.

④  $T_1 = \sum m(2, 4, 7)$



$T_0 = \sum m(1, 2, 5, 6)$



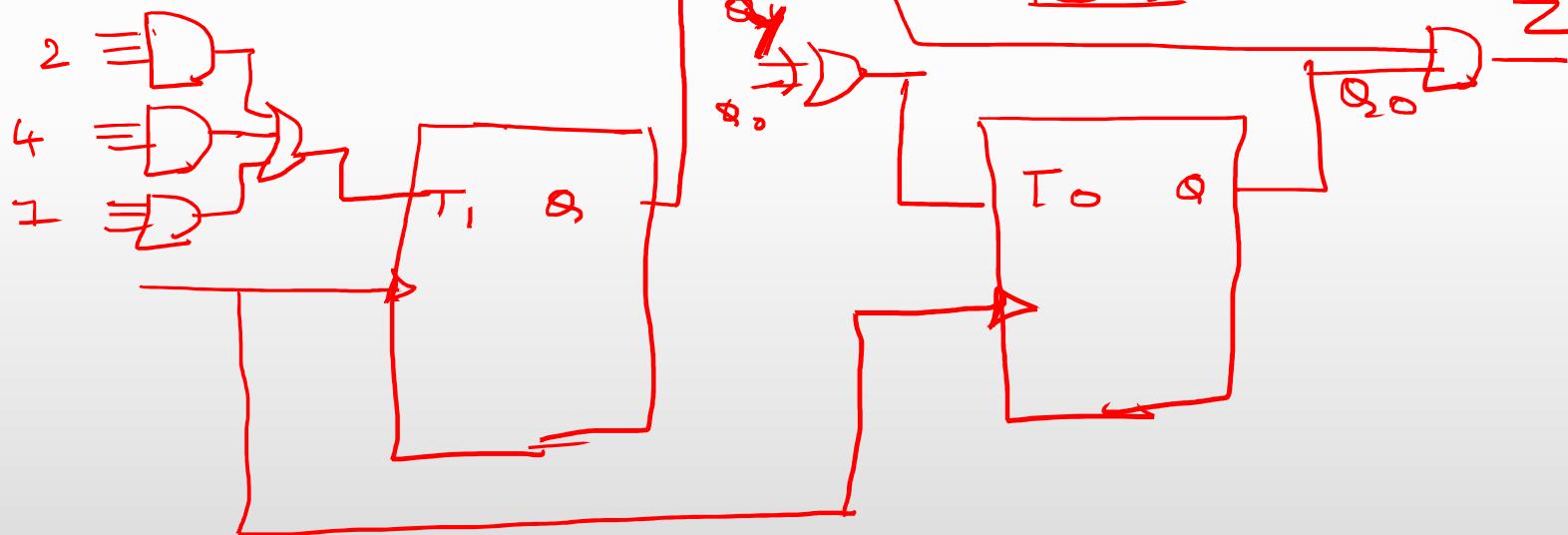
$Z = \sum m(6, 7)$



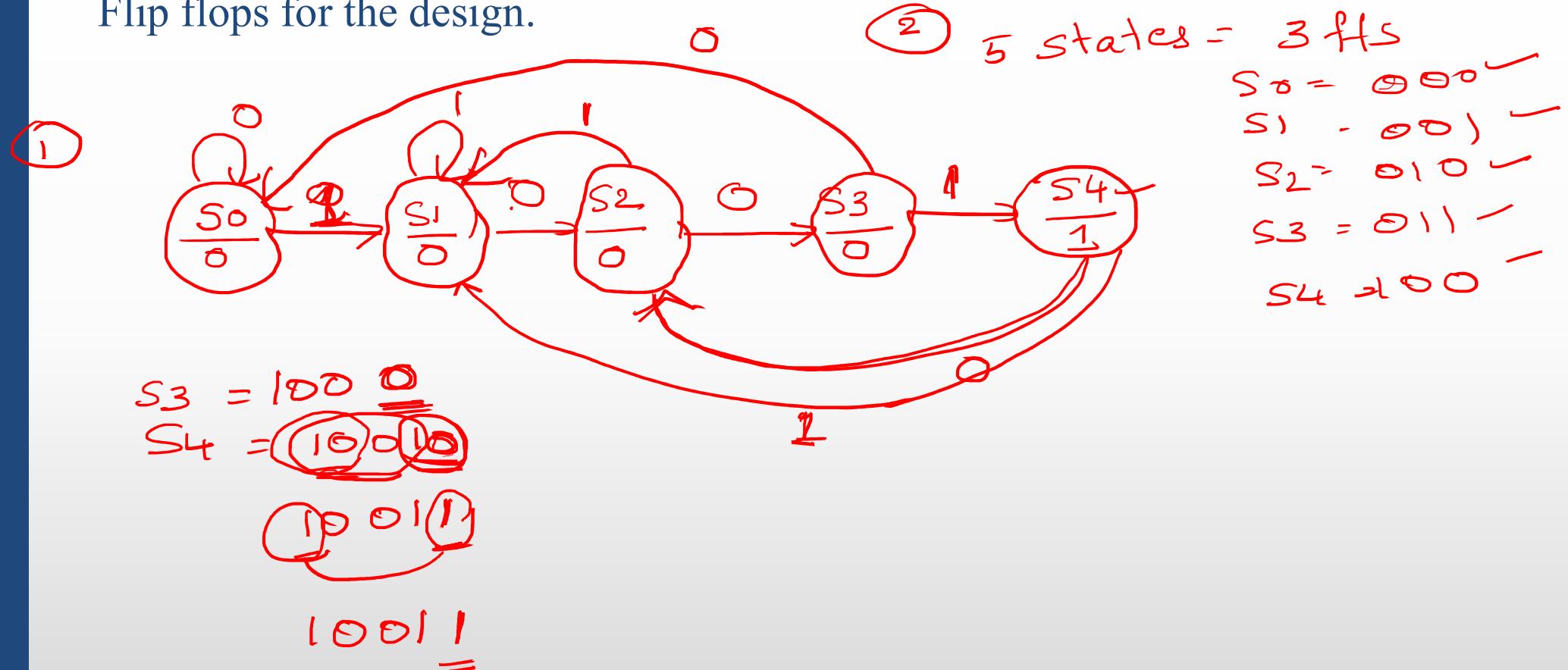
$Z = S_1 S_2$

$Z = S_1 S_2$

⑤

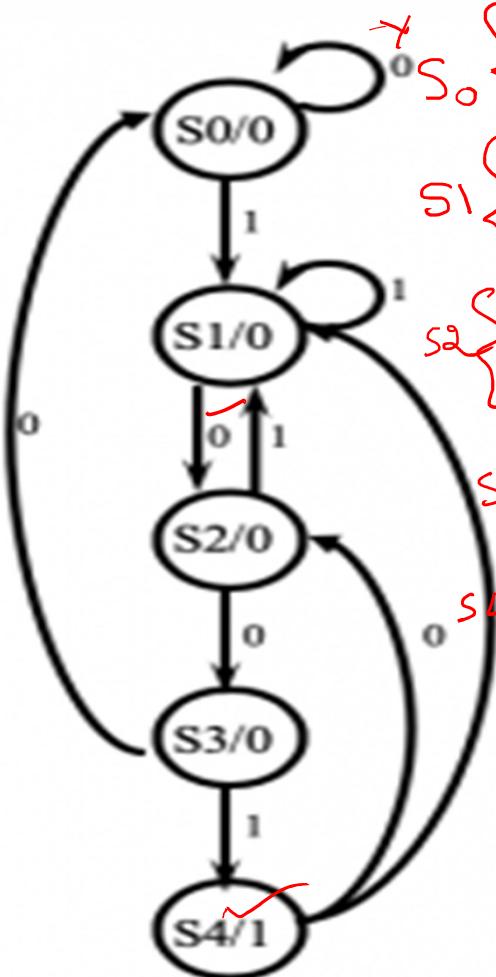


Design a sequential circuit using moore model which produces an output 1 whenever it detects a sequence 1001 and output 0 all other time. Use T flip flops for the design.



Design a sequential circuit using moore model which produces an output 1 whenever it detects a sequence 1001 and output 0 all other time. Use T Flip flops for the design.

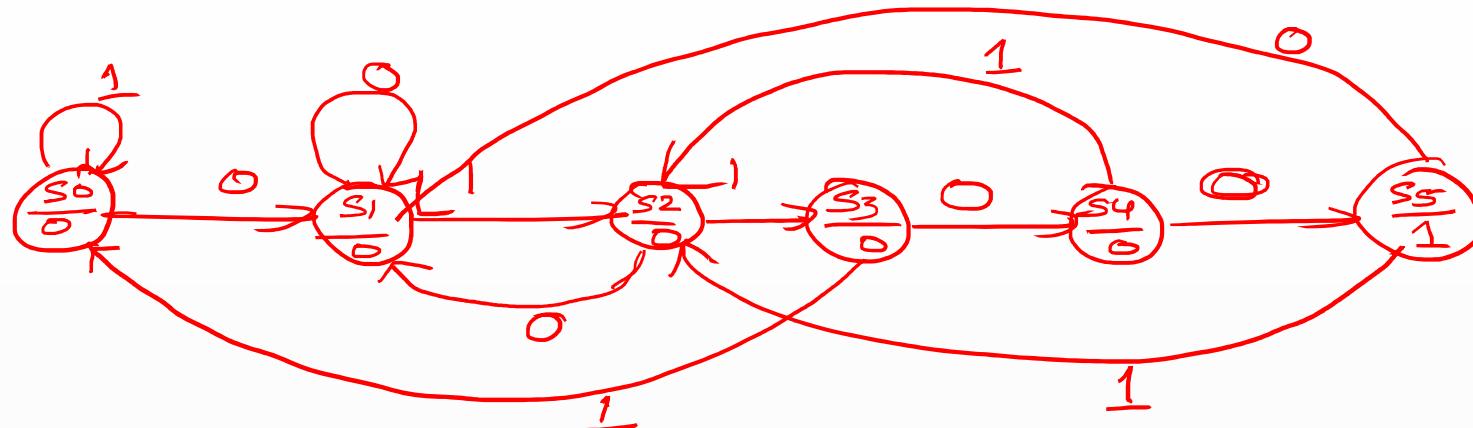
$S_0 =$



Q2	Q1	Q0	Y	Q2 <sup>+</sup>	Q1 <sup>+</sup>	Q0 <sup>+</sup>	T2	T1	T0	Z
0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0	1	1	0
0	0	1	1	0	0	1	0	0	0	0
0	1	0	0	0	1	1	0	0	1	0
0	1	0	1	0	0	1	0	1	1	0
0	1	1	0	0	0	0	0	1	1	0
0	1	1	1	1	0	0	0	1	1	0
1	0	0	0	0	1	0	1	0	0	1
1	0	0	1	0	0	1	0	1	0	1
1	0	1	0	x	x	x				
1	0	1	1	x	x	x				
1	1	0	0	x	x	x				
1	1	0	1	x	x	x				
1	1	1	0	x	x	x				
1	1	1	1	x	x	x				

Simplified expressions for  $T_2$ ,  $\bar{T}_1$ ,  $\bar{T}_0$  & 2  
and complete the circuit

- Design a sequential circuit using moore model which produces an output 1 whenever it detects a sequence 01100 and output 0 all other time. Use T Flip flops for the design.



$$S_2 = \underline{\underline{0}} \underline{\underline{1}} \underline{\underline{0}} \underline{\underline{1}} \underline{\underline{0}} \underline{\underline{0}}$$

6 states : 3ffs  $T_2(\underline{s_2}, \underline{s_1}, \underline{s_0}, Y)$

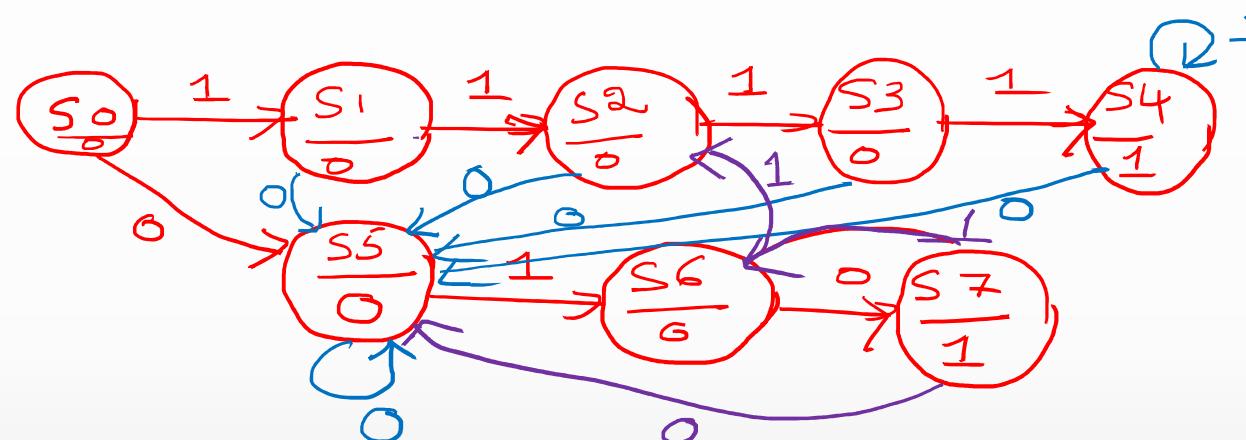
$$S_3 = 0 \ 1 \ 1 \ \underline{\underline{1}}$$

$$S_4 = 0 \ 1 \ 1 \ \underline{\underline{0}}$$

$$S_5 = 0 \ 1 \ 1 \ 0 \ 0 \ \underline{\underline{1}}$$

$$\underline{\underline{0}} \ 1 \ 1 \ 0 \ 0 \ 1$$

- Design a sequential circuit using moore model which produces an output 1 whenever it detects a sequence 1111 or 010 and output 0 all other time. Use D Flip flops for the design.



$$S_1 = \underline{\underline{1010}}$$

$$S_3 = \underline{\underline{1110}}$$

$$S_5 = \underline{\underline{0010}}$$

$$S_2 = \underline{\underline{110}}$$

$$S_4 = \underline{\underline{11110}}$$

$$S_4 = \underline{\underline{11111}} \text{ Rg1} \dots$$

$\overbrace{00000}^{\text{10}} \text{ 10}$   
 $\uparrow \text{ss-ss-ss-ss-ss-}$

8 states = 3 J's

$$D_2(S_2 S_1 S_0 Y) = \underline{\underline{11}} - \text{Kmap}$$

$$S_7 = \underline{\underline{0100}}$$

$$S_7 = \underline{\underline{0101}}$$

$$S_6 = \underline{\underline{011}}$$

- Design a sequential circuit using moore model which produces an output 1 whenever it detects a sequence 1001 or 0100 and output 0 all other time. Use JK Flip flops for the design.

= 9 states  $\Rightarrow$  4 J-K's

Take as an assignment  
S - variables

$J_i (\overbrace{Q_3 Q_2 Q_1 Q_0}^{\text{S - variables}})$

# Programmable Logic devices (PLD)

Dec. 31<sup>st</sup>, 2021

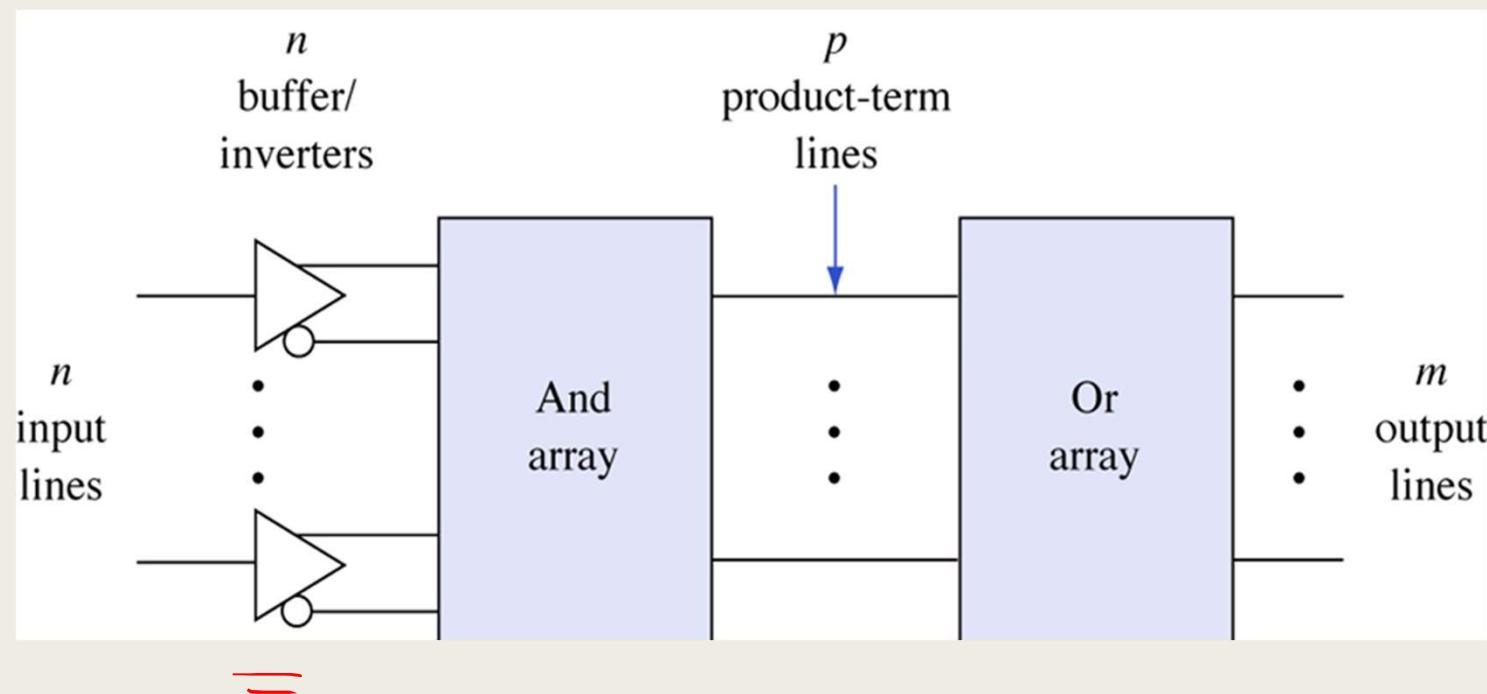
# Programmable Logic devices (PLD)

- Types of PLD ✓
- Combinational function implementation using PLD

# PLD

- PLD is an integrated circuit with internal logic gates (AND and OR) that are connected through electronic paths which behave similar to fuses.
- Programming the devices involve blowing the fuses along the path, making them very well suited for academic and prototyping
- A typical PLD may have hundreds to millions of gates interconnected through hundreds and thousands of internal paths

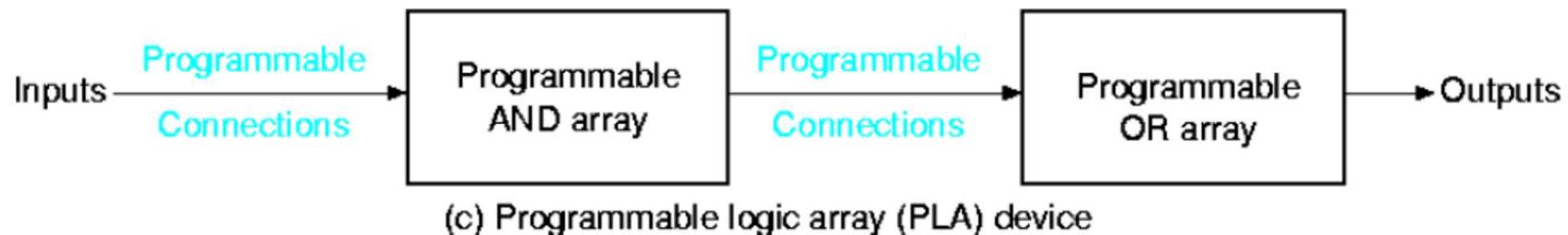
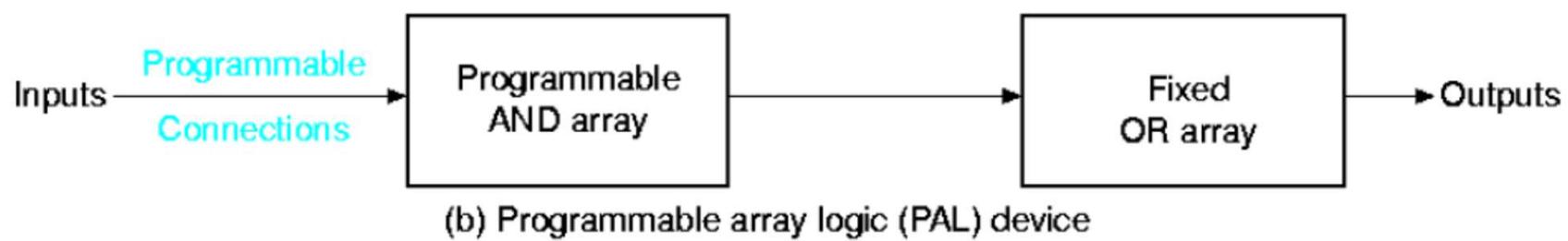
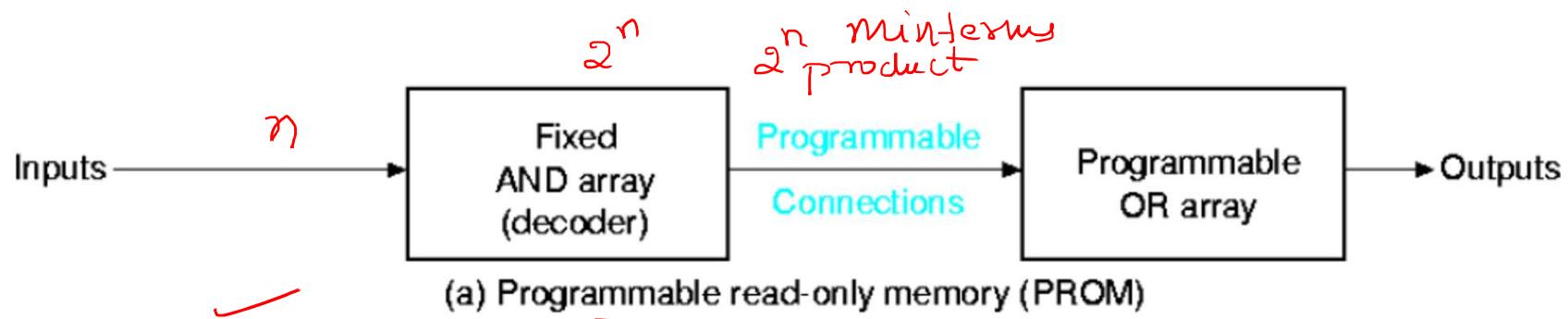
# General Structure of PLDs



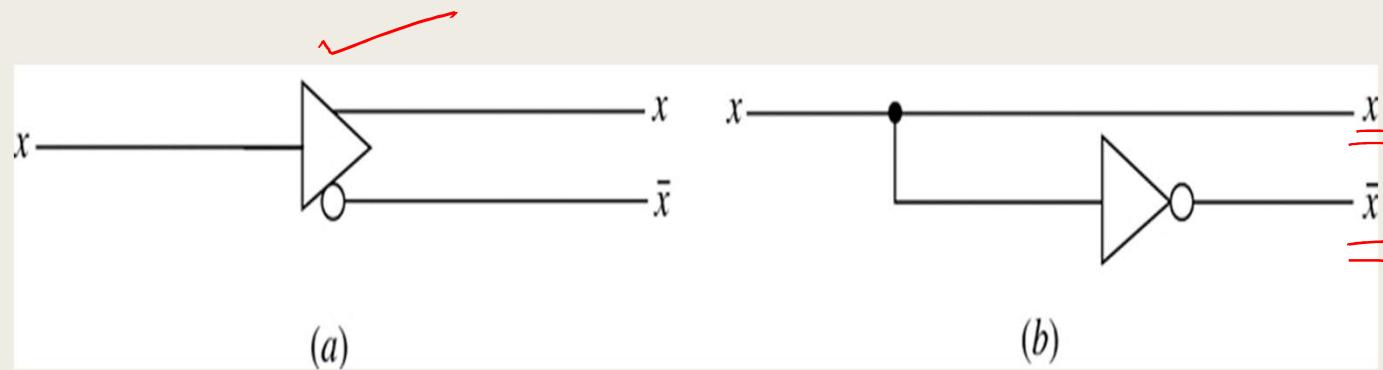
# Types of PLDs

- Read Only Memory (ROM)
- Programmable Logic Array (PLA)
- Programmable Array Logic (PAL)

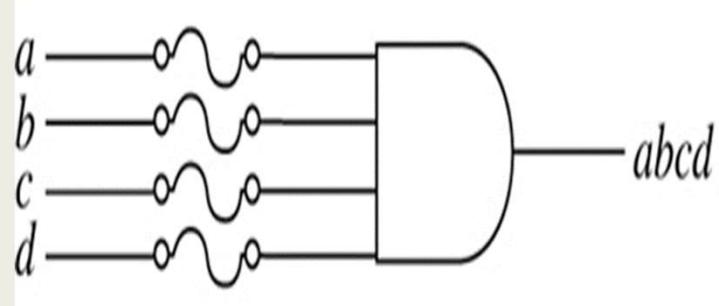
Device	AND-array	OR-array
PROM	Fixed	Programmable
PLA	Programmable	Programmable
PAL	Programmable	Fixed



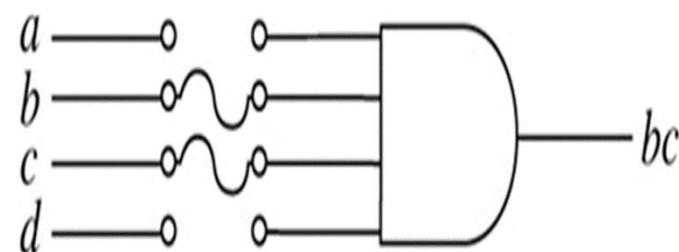
# Buffer /Inverter Symbol:



# Fusible /programmable AND gate symbol:

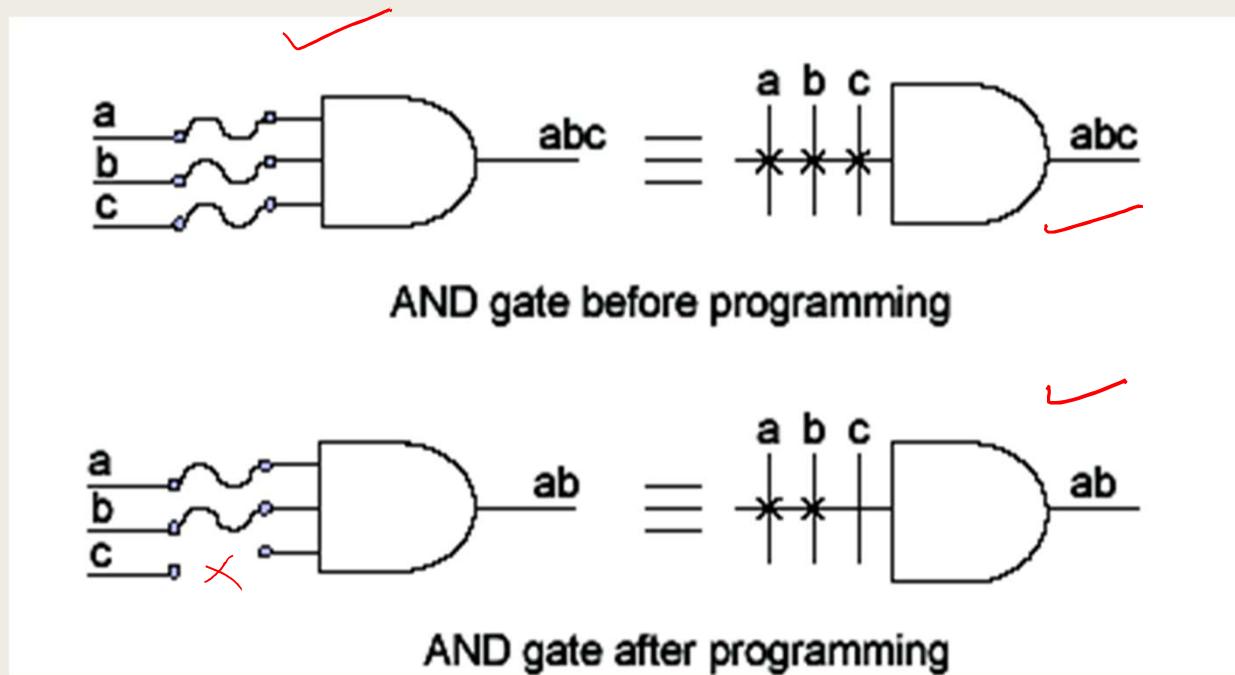


(a)

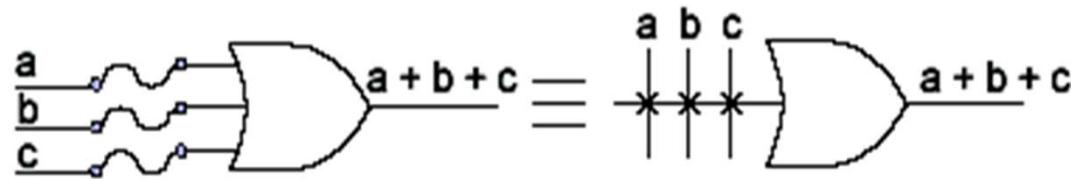


(b)

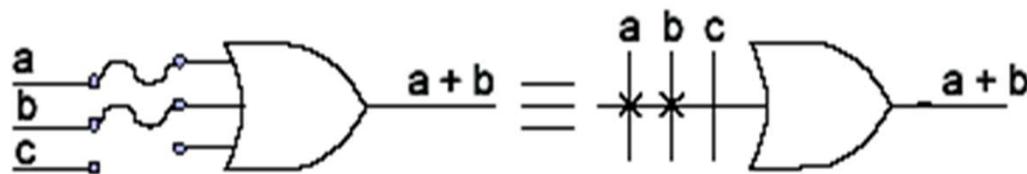
# AND – PLD representation



# OR gate -PLD representation

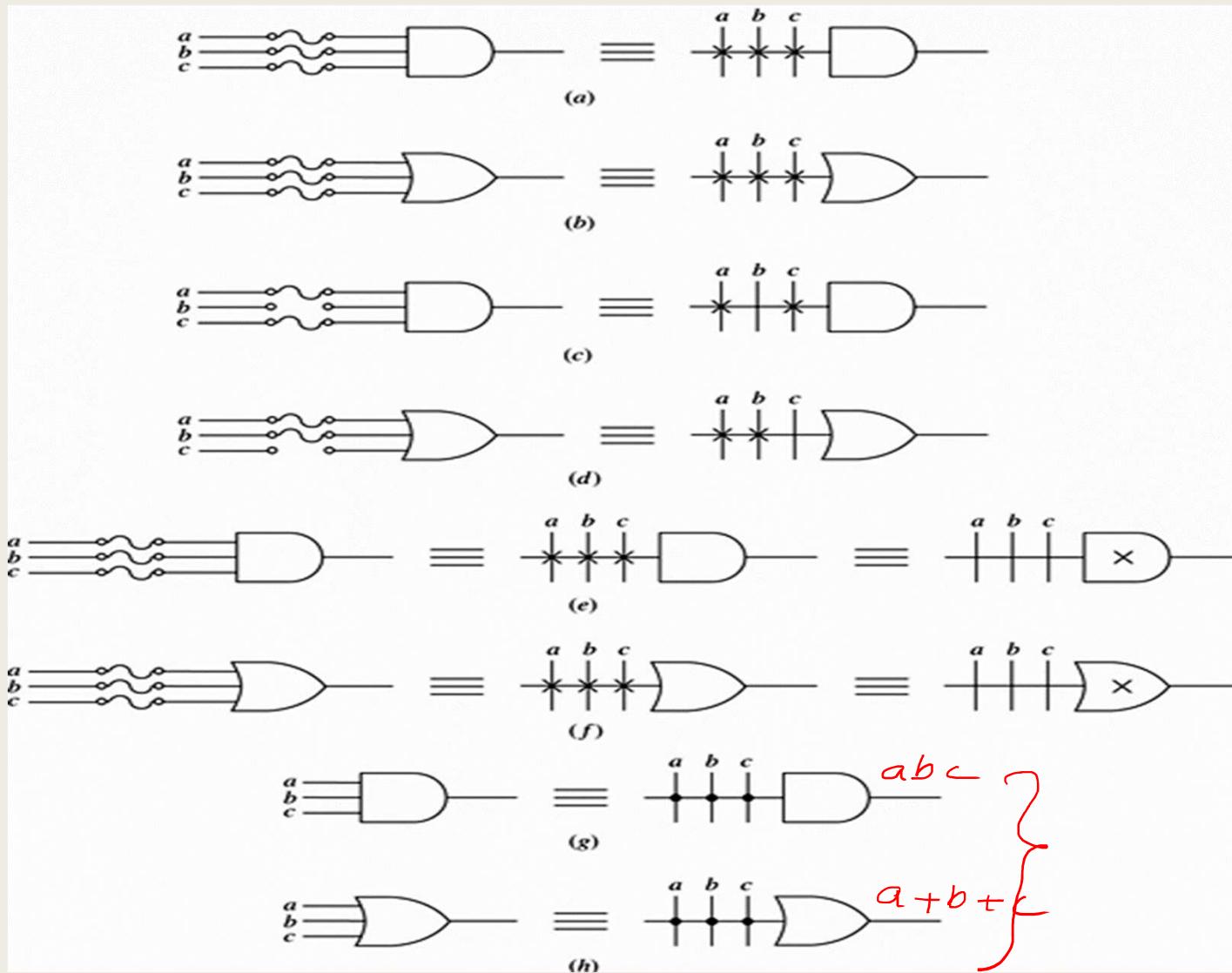


OR gate before programming

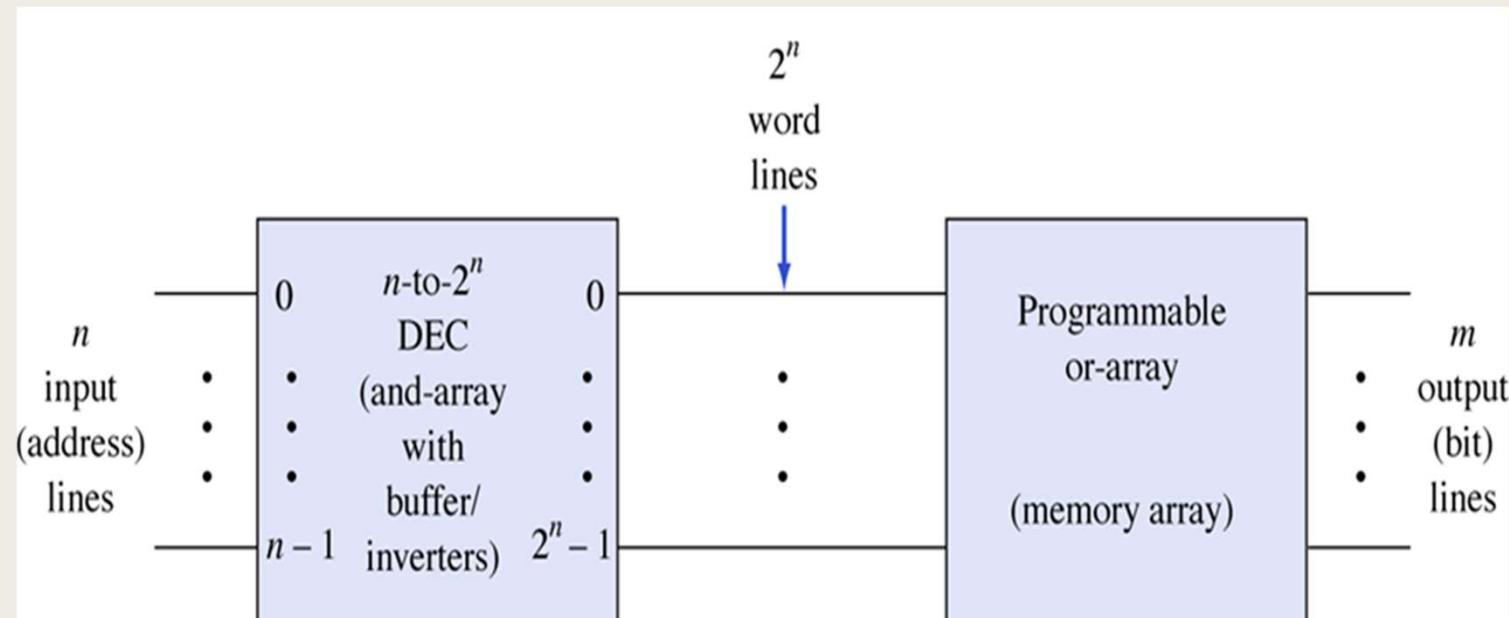


OR gate after programming

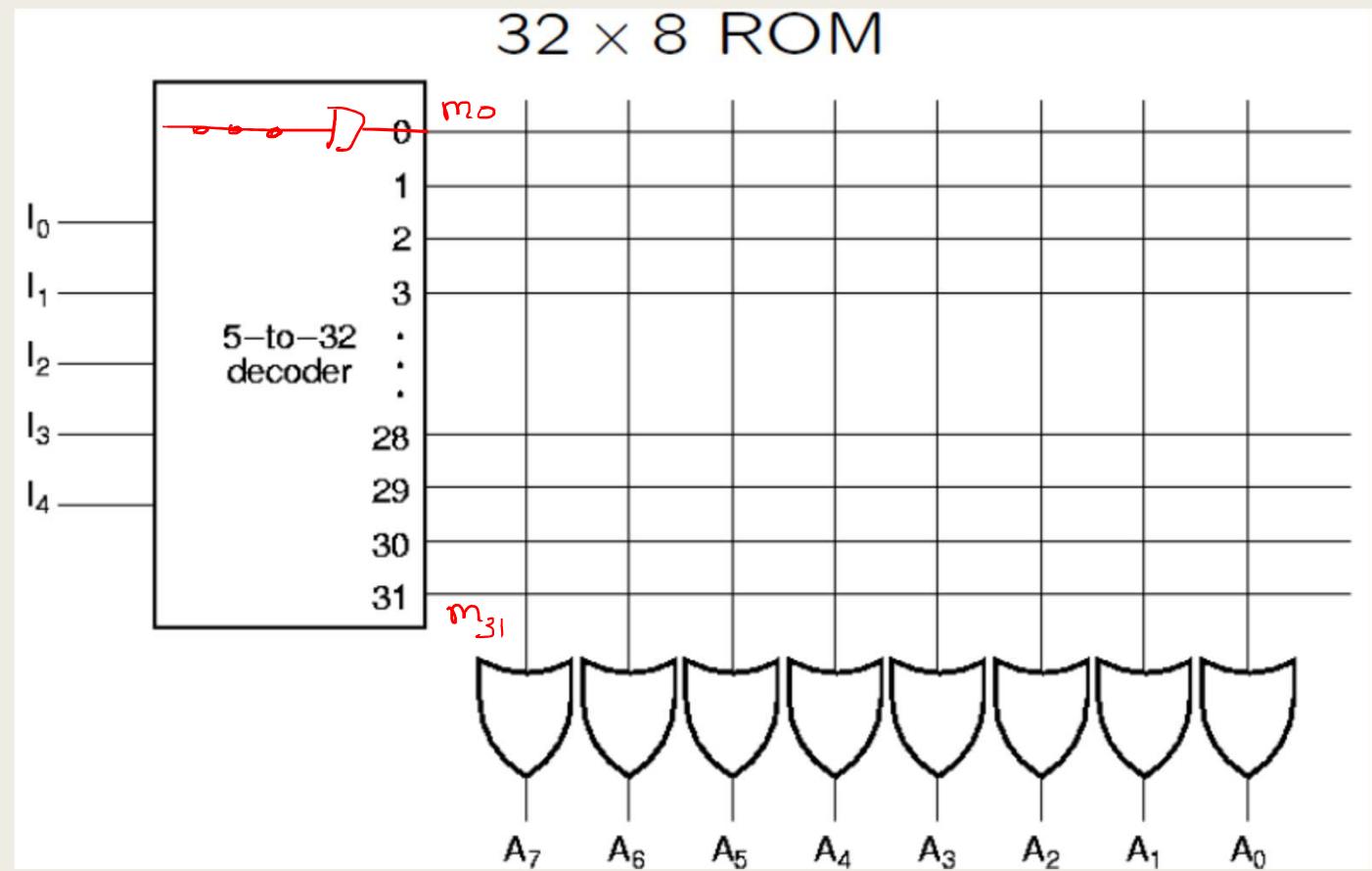
# OR gate -PLD representation



# PROM



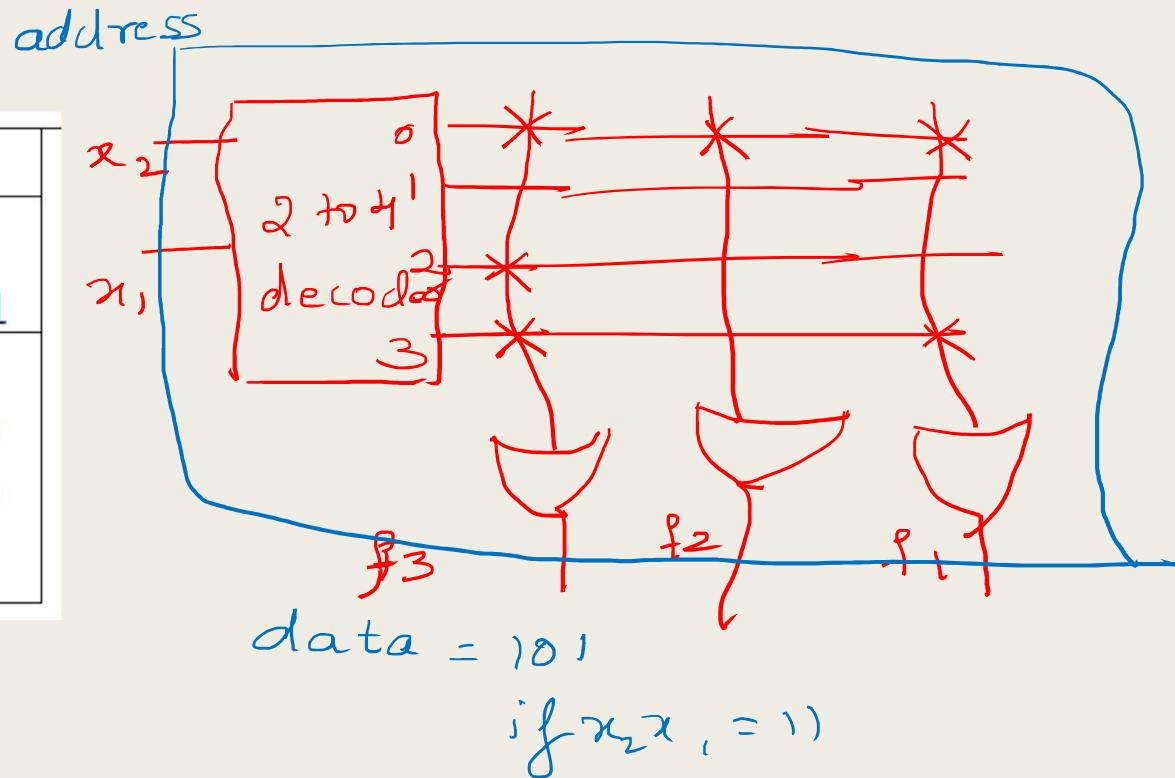
# PROM



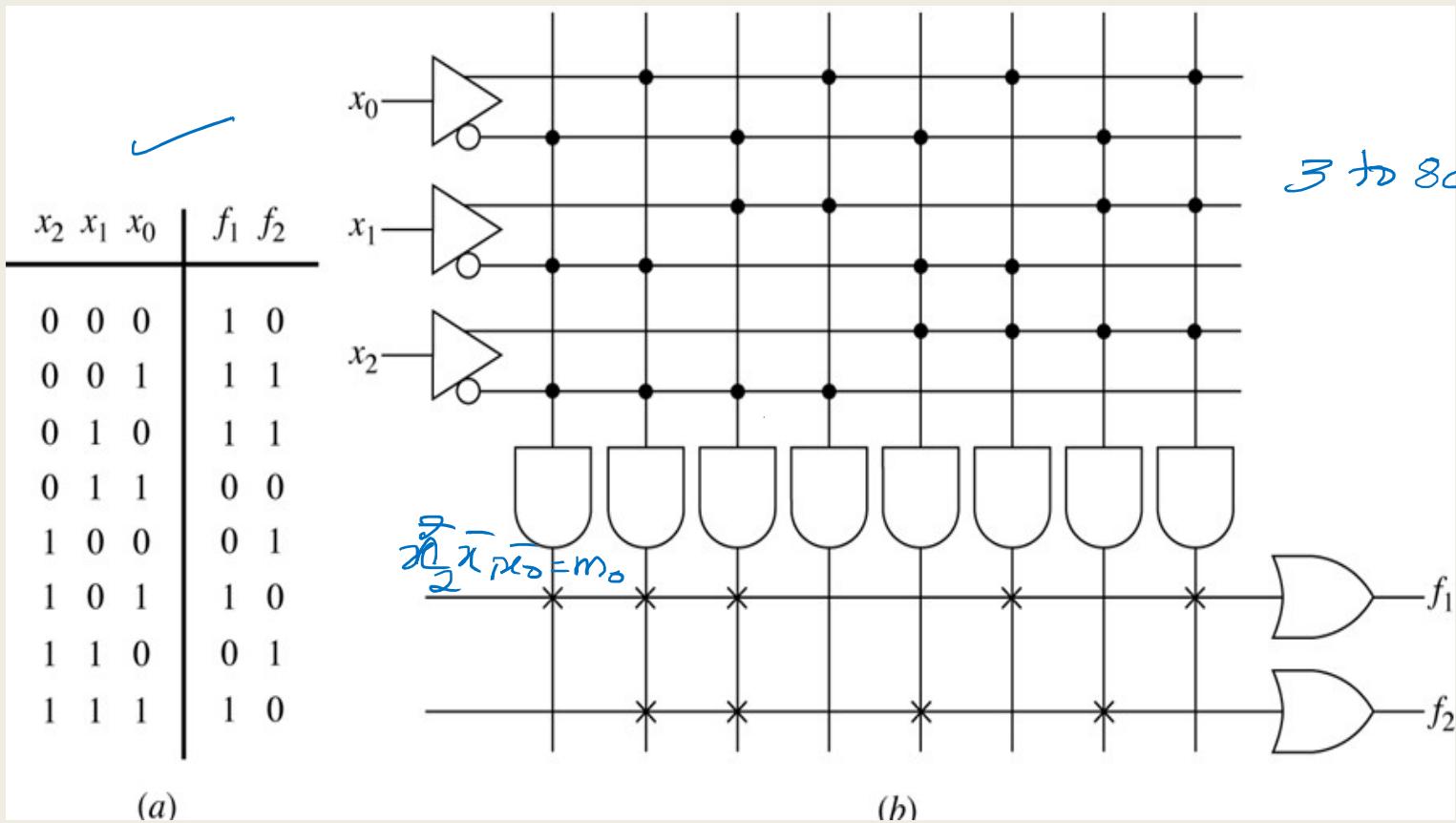
# Programming the ROM to store data

✓

Truth table				
Address		Content		
$x_2$	$x_1$	$f_3$	$f_2$	$f_1$
0	0	1	1	1
0	1	0	0	0
1	0	1	0	0
1	1	1	0	1



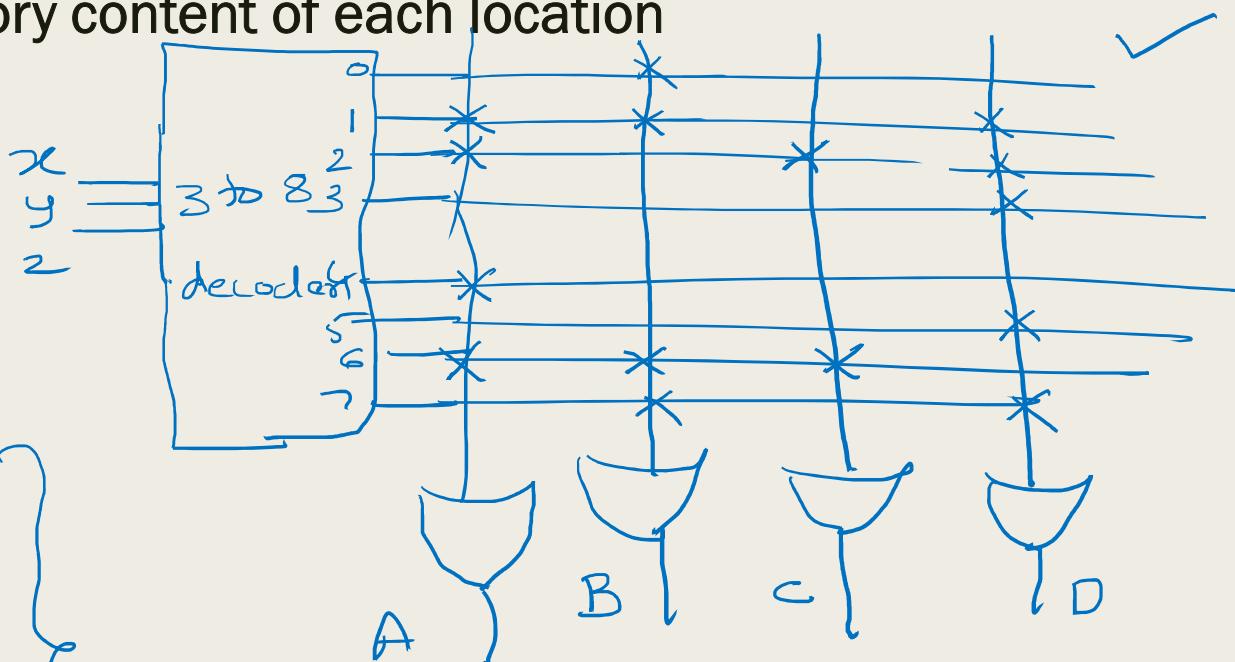
# Using a PROM for logic design : example



Realize the functions using 8x4 ROM .Provide PLD representation.  
Also specify the memory content of each location

- $A(x,y,z) = \Sigma m(1,2,4,6)$
- $B(x,y,z) = \Sigma m (0,1,6,7)$
- $C(x,y,z) = \Sigma m (2,6)$
- $D(x,y,z) = \Sigma m (1,2,3,5,7)$

$x\ y\ z$	A	B	C	D
000	0	1	0	0
001	1	1	0	1
010	1	0	1	1
011	0	0	0	1
100	1	0	0	0
101	0	0	0	1
110	1	1	1	0
111	0	1	0	1



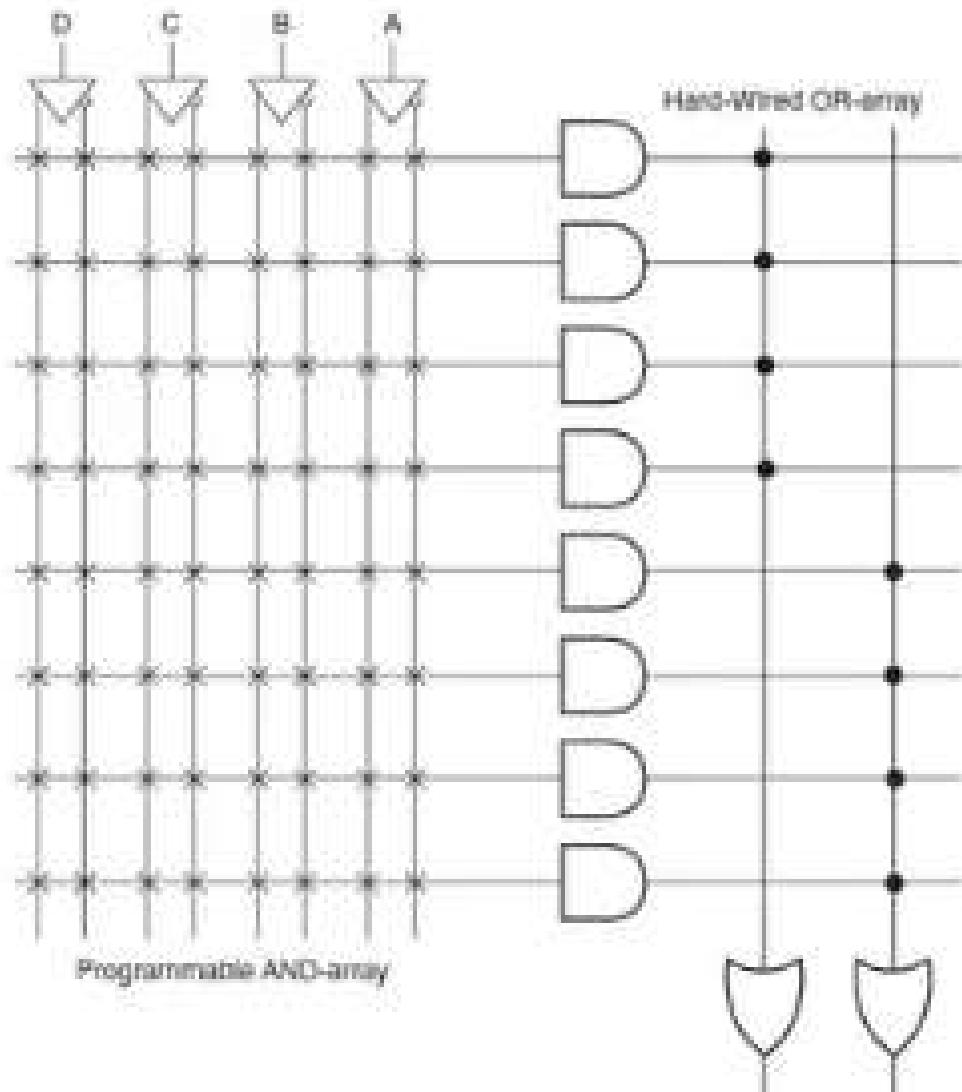
memory content

$$A(x,y,z) = \Sigma m(1,2,4,6) \quad B(x,y,z) = \Sigma m (0,1,6,7)$$

$$C(x,y,z) = \Sigma m (2,6) \quad D(x,y,z) = \Sigma m (1,2,3,5,7)$$

$A(x,y,z) = \Sigma m(1,2,4,6)$     $B(x,y,z) = \Sigma m (0,1,6,7)$     $C(x,y,z) = \Sigma m (2,6)$   
 $D(x,y,z) = \Sigma m (1,2,3,5,7)$

# PAL: Programmable AND array and Fixed OR array :



Realize following functions using PAL shown in previous slide. Write the PLD representation.

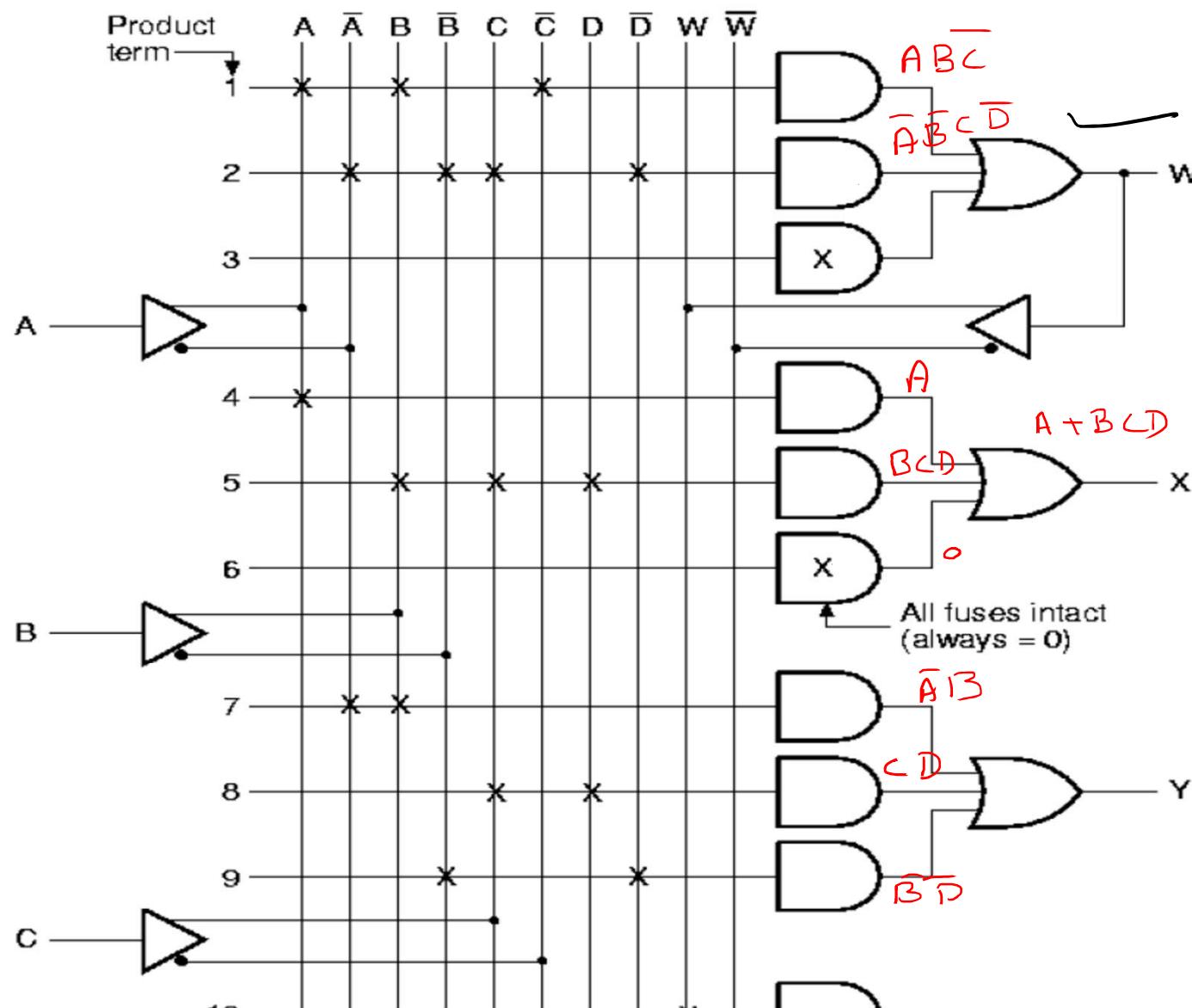
$$w(A,B,C,D) = \Sigma m(2,12,13)$$

$$x(A,B,C,D) = \Sigma m(7,8,9,10,11,12,13,14,15)$$

$$y(A,B,C,D) = \Sigma m(0,2,3,4,5,6,7,8,10,11,15)$$

## PAL: Programmable AND array and Fixed OR array

AND gates inputs



Realize the following truth table (3-i/p, 4-o/p combinational circuit) using PAL.

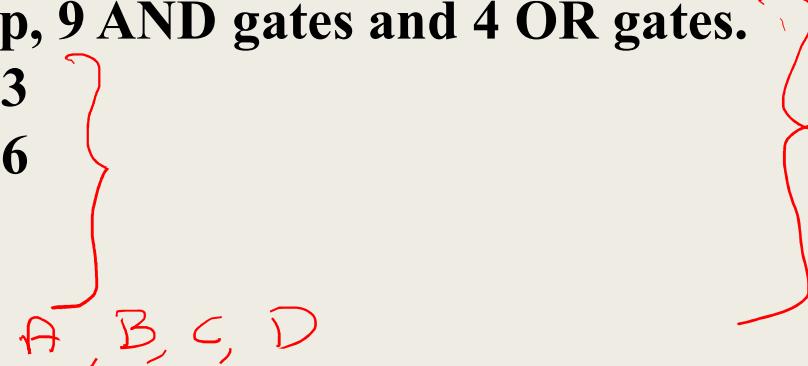
Write the PLD representation. PAL has 4 i/p, 9 AND gates and 4 OR gates.

Input to OR gate1: output of AND gate 1,2,3

Input to OR gate2: output of AND gate 4,5,6

Input to OR gate3: output of AND gate 6,7

Input to OR gate4: output of AND gate 8,9



✓

Inputs			Outputs			
x	y	z	A	B	C	D
0	0	0	0	1	0	0
0	0	1	1	1	1	1
0	1	0	1	0	1	1
0	1	1	0	1	0	1
1	0	0	1	0	1	0
1	0	1	0	0	0	1
1	1	0	1	1	1	0
1	1	1	0	1	1	1

$$A = \Sigma m(1, 2, 4, 6)$$

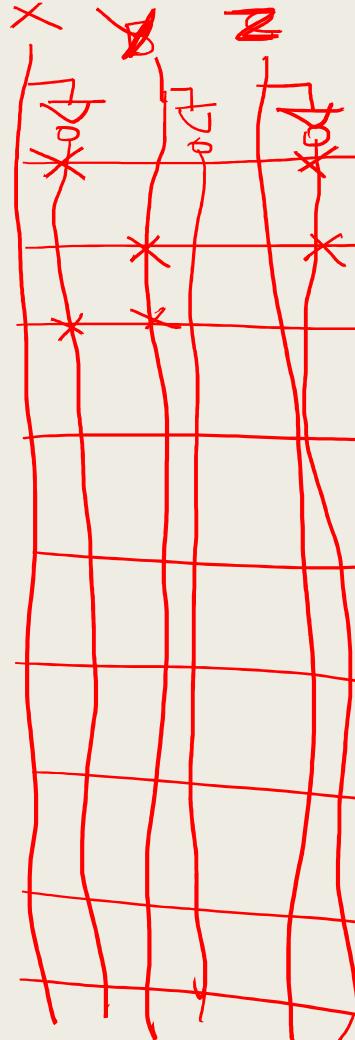
$$B = \Sigma m(0, 1, 3, 4, 5)$$

$$C = \Sigma m(1, 2, 4, 6, 7)$$

$$D = \Sigma m(1, 2, 3, 5, 7)$$

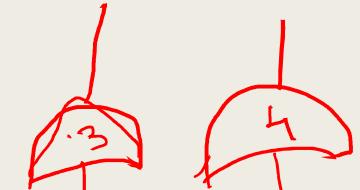
$$A = \bar{x}_0 \bar{z} \bar{x} + \bar{x} y + \bar{x} y$$

$C =$



A

B =



1

2

3

4

5

6

7

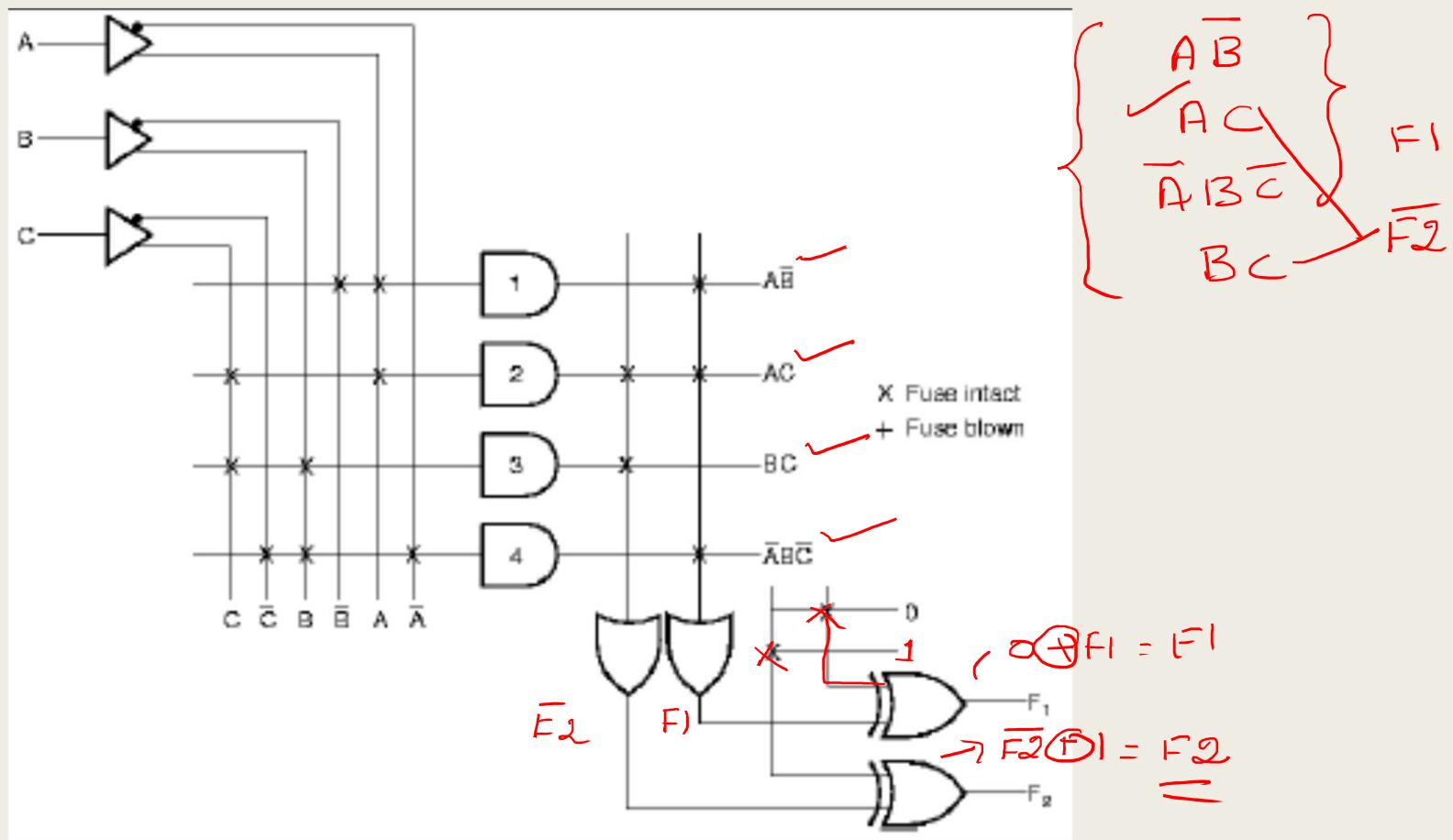
# PLA

- PLA is similar to PROM with decoder replaced by programmable AND gate arrays
- AND array can be programmed to generate any product term
- Outputs of the AND gates are connected as inputs to OR gates
- Out put of OR gate is the input to 2-i/p XOR gates, where the other input can be programmed to receive either 0 or 1
- PLA size is specified as  $k \times n \times m$ , k:number of inputs, n: no. of AND gates, m: no. of outputs *(OR gates)*  
*m no. XOR gates*

PLA: An example to realize  $F_1(A,B,C) = AB' + AC + A'BC'$  AND  $F_2 = (AC + BC)'$  using  
 $\checkmark$   
 $3 \times 4 \times 2$  PLA

$$\checkmark F_2 = \underline{AC} + \underline{BC}$$

- PLA has 3- inputs, 4 AND gates , 2 OR gates and 2 XOR gates.



PLA table for  $F_1(A,B,C) = \overline{AB} + \overline{AC} + \overline{A'}BC'$  AND  $F_2 = (\overline{AC} + BC)'$

$$\overline{F_2} = \overline{AC} + BC$$

- First column lists all product terms, second column specifies the inputs to AND gates and third column specifies the inputs to OR gate
- T (True) / C (Complement) specifies the input to XOR gates

PLA Program Table		Inputs			Outputs	
Term	Term #	A	B	C	$F_1$	$F_2$
$AB$	1	1	0	-	1	-
$AC$	2	1	-	1	1	1
$BC$	3	-	1	1	-	1
$\overline{ABC}$	4	0	1	0	1	-

$\rightarrow$  decide the inputs to OR gate

Realize  $f_1(x,y,z) = \Sigma m(3,6,7)$ ,  $f_2(x,y,z) = \Sigma m(0,1,2,6,7)$ ,  
 $f_3(x,y,z) = \Sigma m(0,1,3,4,5)$  using a 3x5x3 PLA. Also write the PLA table

- Step1: simplify to SOP expression for  $f_1, f_1'$ ,  $f_2, f_2'$ , and  $f_3, f_3'$ . Identify the combination that gives minimum product terms.
- Step2: Represent in PLD notation
- Step3: Write PLA table.

$$f_1 = yz + xy$$


---


$$\bar{f}_1 = \bar{y} + \bar{x}\bar{z}$$

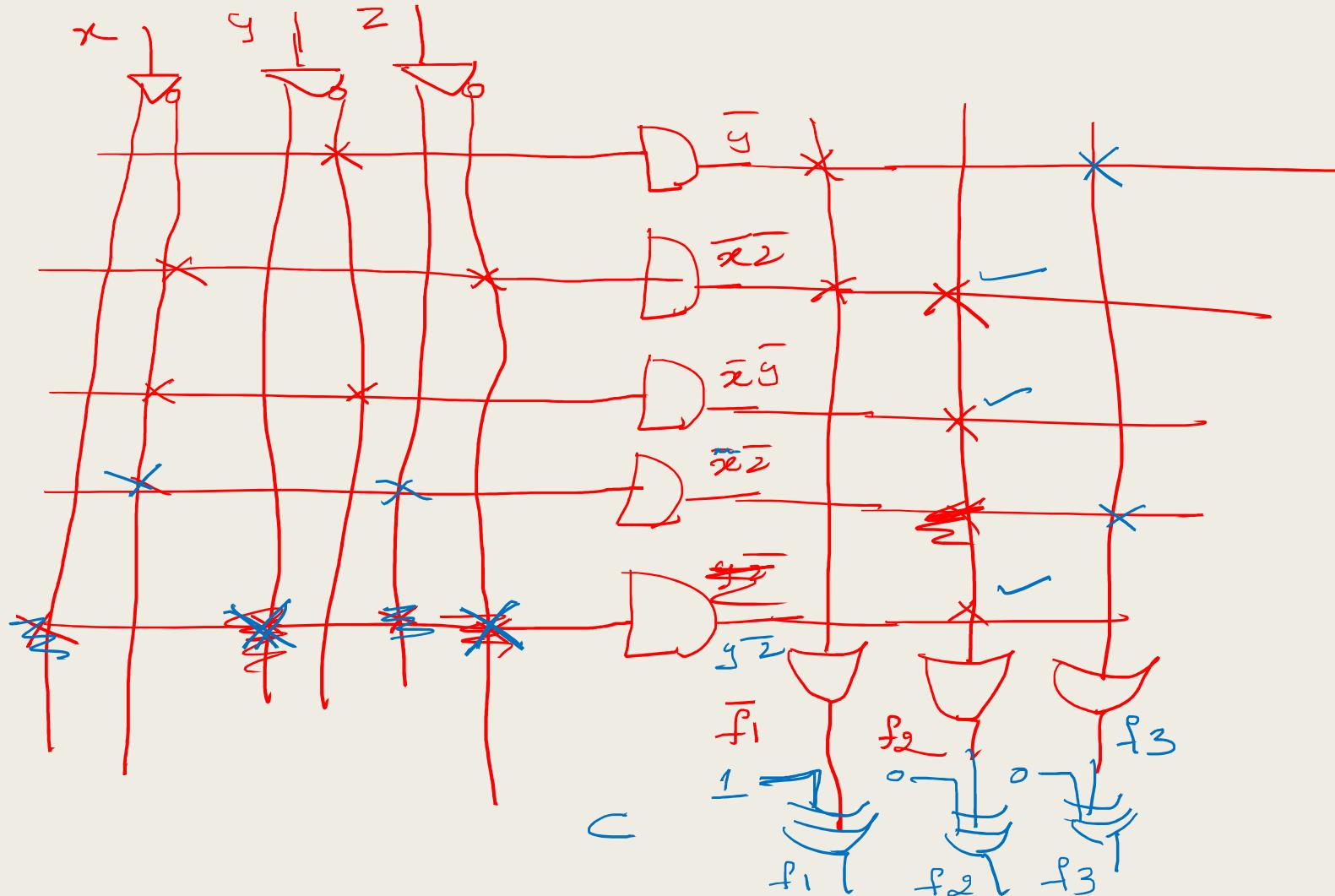
$\hookrightarrow$  AND gates

$$\begin{aligned} \bar{f}_1 \\ f_3 \\ f_2 = \end{aligned}$$

$$\begin{aligned} f_2 &= \bar{x}\bar{y} + \bar{x}\bar{z} + \cancel{y}\bar{z} & f_3 &= \bar{y} + \bar{x}z \\ \bar{f}_2 &= xy + \bar{x}yz & f_3 &= xy + y\bar{z} \end{aligned}$$

$$\left. \begin{array}{c} \bar{y} \\ \bar{x}\bar{z} \\ \bar{x}z \\ \bar{x}\bar{y} \\ y\bar{z} \end{array} \right\} \hookrightarrow \text{AND gates}$$

Realize  $f_1(x,y,z) = \Sigma m(3,6,7)$ ,  $f_2(x,y,z) = \Sigma m(0,1,2,6,7)$ ,  
 $f_3(x,y,z) = \Sigma m(0,1,3,4,5)$  using a  $3 \times 5 \times 3$  PLA. Also write the PLA table



Realize  $f_1(x,y,z) = \Sigma m(3,6,7)$ ,  $f_2(x,y,z) = \Sigma m(0,1,2,6,7)$ ,  
 $f_3(x,y,z) = \Sigma m(0,1,3,4,5)$  using a  $3 \times 5 \times 3$  PLA. Also write the PLA table

PLA table

$$\bar{f}_1 = \bar{y} + \bar{x}\bar{z}$$

Terms	Inputs			Outputs		
	x	y	z	$f_1$	$f_2$	$f_3$
1	-	0	-	1	-	-
2	0	-	0	1	1	1
3	0	-	1	-	-	1
4	0	0	-	-	1	-
5	-	1	0	-	1	-

C T T  
 \_\_\_\_\_

# Additional problems

- Refer the problems at the end of chapter 7: memory and Programmable logic in digital design (3<sup>rd</sup> edition) by M Morris mano

■ Questions: ?

## TABULATION METHOD CONTD..

Ex:  $G(A,B,C,D,E,F) = \Pi M(20,28, 52,60)$

Col1	Col2	Col3	
20 010100	(20,28) 01_100	(20,28,52,60) _1_100	
28 011100	(28,60) _11100		
52 110100	(52,60) 11_100		
60 111100			

$$F' = BDE'F'$$

$$F = (BDE'F')' = B' + D' + E + F$$

Simplify the function  $F(v,w,x,y,z) = \prod M(1,3,6,9,10,11,12,14,15,17,19,20,22,24,25,26,27) + D(4,8,13)$  using tabulation method and draw the circuit for the simplified expression using NOR gates only. Solution in next slide...

Please fill column1 and column2..

col1	Column . 2	Column	Column 3 (quads)	Column 4(octets)
			(1,3,9,11) (2,8) ✓	(17, 19,25,27)(2,8) ✓
			(1,3,17,19)(2,16) ✓	(24,25,26,27)(1,2) ✓
			(1,9,17,25)(8,16) ✓	
			(4,6,12,14)(2,8) PI	
			(4,6,20,22)(2,16) PI	Column 4(octets)
			(8,9,10,11)(1,2) ✓	(1,3,9,11,17,19,25,27)(2,8,16) PI
			(8,9,24,25)(1,16) ✓	(8,9,10,11,12,13,14,15)(1,2,4) PI
			(8,10,12,14)(2,4) PI	(8,9,10,11,24,25,26,27)(1,16,2) PI
			(8,10,24,26)(2,16) ✓	
			(8,12,9,13)(1,4) PI	
			(3,11,19,27)(8,16) ✓	
			(9,11,25,27)(2,16) ✓	
			(10,11,26,27)(1,16) ✓	
			(12,13,14,15)(1,2) ✓	
			(9,11,13,15)(4,2) PI	PI
			(13,15,17,19)(2,4) PI	PI

## Step2: Finding essential prime implicants

	1	3	6	9	10	11	12	14	15	17	19	20	22	24	25	26	27
PI1		X					X	X									
PI2		X											X	X			
PI3				X			X	X									
PI4			X				X										
PI5				X		X				X							
PI6									X	X	X						
PI7	X	X		X		X				X	X				X		X
PI8				X	X	X	X	X	X								
PI9				X	X	X								X	X	X	X

$$\overline{F} = \overline{P}I_2 + \overline{P}I_2 + \overline{P}I_8 + \overline{P}I_9 =$$

Contd..

15 8 4 2 )

A B C D E

$$\blacksquare \quad \text{PI7} = (1, 3, 9, 11, 17, 19, 25, 27)(2, 8, 16) = \underline{\phantom{0}} \underline{\phantom{0}} 0 \underline{\phantom{0}} 1 = \cancel{DE} \bar{C}E$$

$$\blacksquare \quad \text{PI2} = (4, 6, 20, 22)(2, 16) = \underline{\phantom{0}} \underline{\phantom{0}} 0 \underline{\phantom{0}} 1 \underline{\phantom{0}} 0 = \bar{B}C\bar{E}$$

$$\blacksquare \quad \text{PI8} = (8, 9, 10, 11, 12, 13, 14, 15)(1, 2, 4) = \underline{\phantom{0}} \underline{\phantom{0}} \underline{\phantom{0}} 1 = \bar{A}B$$

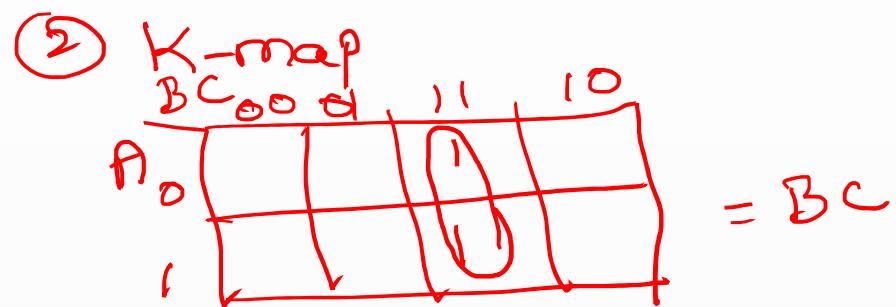
$$\blacksquare \quad \text{PI9} = (8, 9, 10, 11, 24, 25, 26, 27)(1, 16, 2) = \bar{B}\bar{C}$$

$$\checkmark \bar{F} = \bar{C}E + \bar{B}C\bar{E} + \bar{A}B + B\bar{C}$$

$$\blacksquare \quad F = (C + \bar{E})(B + \bar{C} + E)(A + \bar{B}) (\bar{B} + C) \Rightarrow \text{NOR gates}$$

# TABULATION METHOD

①  $\overline{A} \overset{3}{B} C + \underline{A} \overset{7}{B} C = BC (\overline{A} + A) = BC \rightarrow \text{Boolean theorem}$



## Tabulation or Quine McClusky(QM) method

- General simplification method which can be applied to any number of variables
- Includes two steps:
  - Step 1: Obtaining prime implicants  $\rightarrow$  all possible groups
  - Step 2 : Selection of essential prime implicants  $\rightarrow$  eliminate redundant groups/terms

# Tabulation or Quine McClusky(QM) method

- Procedure:

1. Express each minterm by its binary representation

$$\left. \begin{array}{l} 2 = 0010 \rightarrow 1 \\ 3 = 0011 = 2 \text{ logic } 1's \\ 5 = 0101 = " \\ 6 = 0110 = " \\ 8 = 1000 = \text{one logic } 1's \end{array} \right\}$$

2. List the minterm according to increasing number of logic 1's in the binary representation

$$\begin{array}{r} 2 \quad 0010 \\ 8 \quad 1000 \\ \hline 3 \quad -0011 \\ 5 \quad -0101 \\ 6 \quad -0110 \end{array}$$

3. Group the minterm with equal index. (Ex: (1,2,4), (3,5,6)..)

# Tabulation or Quine McClusky(QM) method

4. Compare each term in index  $i$  with each term in index  $(i+1)$ . Form a new term if they differ by exactly one position , with a '-' in place of variables that differ.

Ex:  $\begin{array}{l} 0110 \\ \quad \quad \quad \} \\ 0010 \\ \quad \quad \quad \} \end{array}$

new term is  $0-10$

New term should be written in next column with index  $i+1$ .

$$\begin{array}{r} 2 \quad 0010 \\ - 8 \quad 1000 \\ \hline \end{array} \quad (i)$$
$$\begin{array}{r} 3 \quad 0011 \\ - 5 \quad 0101 \\ \hline \end{array} \quad (i+1)$$
$$\begin{array}{r} 6 \quad 0110 \\ \hline \end{array}$$

(2,3)  
001-  
(2,6)  
0-10

5. Repeat step 4 till last indices. ✓

6. Repeat step 4 and 5 for the new list formed. ✓

7. Terminate the process when no new list is formed.

Example1:  $F(a,b,c,d) = \sum m(0,1,2,5,6,7,8,9,10,14)$

Step 1: Finding prime implicants

Pairs -

Quads

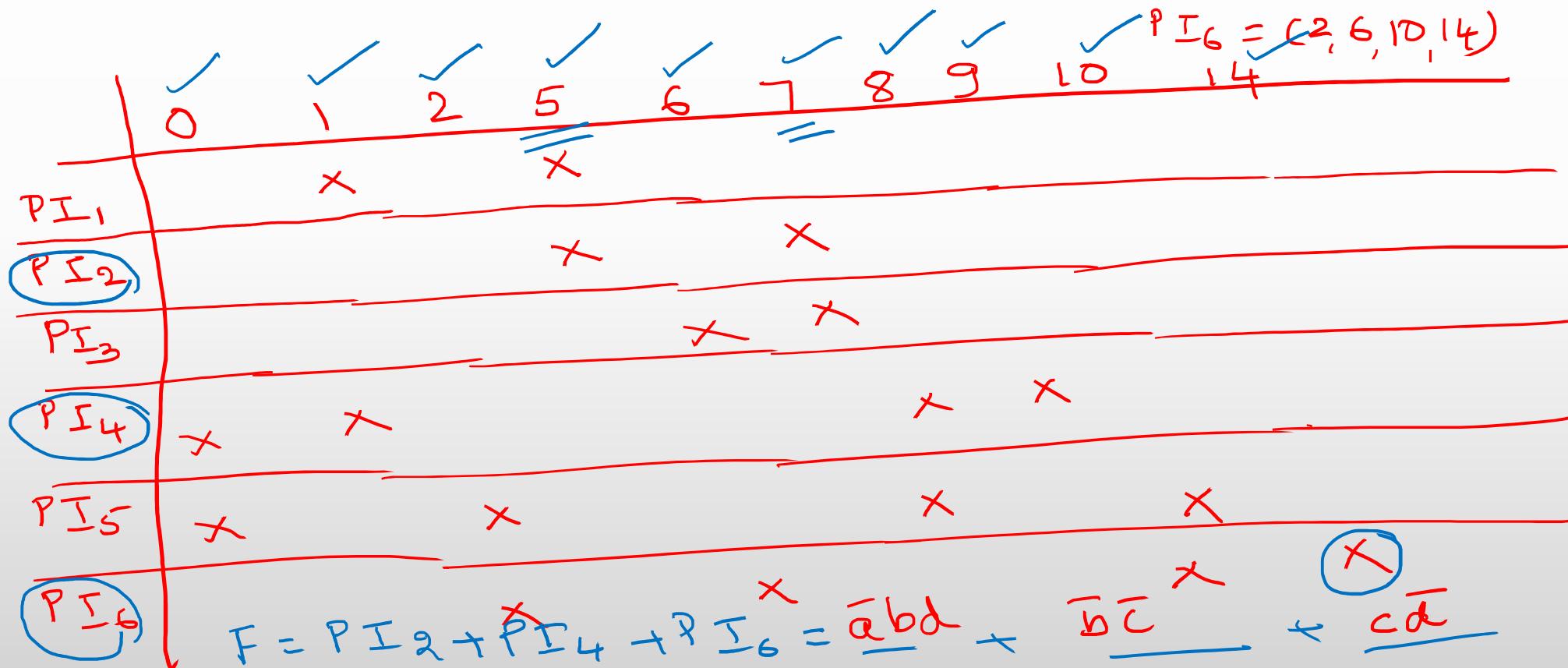
	Column I <i>ab</i> <i>c</i> <i>d</i>	Column II <i>a</i> <i>b</i> <i>c</i> <i>d</i>	Column III <i>a</i> <i>b</i> <i>c</i> <i>d</i>
group 0	0 0000 ✓✓	(0, 1) 000- ✓✓	(0, 1, 8, 9) -00- ✓✓
group 1	{ 1 0001 ✓✓✓	0, 2 00-0 ✓✓	0, 2, 8, 10 -0-0 ✓✓
	2 0010 ✓✓✓	0, 8 -000 ✓✓	0, 8, 1, 9 -00- ✓✓
	8 1000 ✓✓✓	1, 5 0-01 PI1 ✓✓	0, 8, 2, 10 -0-0 ✓✓
	5 0101 ✓✓✓	1, 9 -001 ✓✓	2, 6, 10, 14 --10 ✓✓
group 2	{ 6 0110 ✓✓✓	2, 6 0-10 ✓✓	2, 10, 6, 14 ---10 ✓✓
	9 1001 ✓✓✓	2, 10 -010 ✓✓	
	10 1010 ✓✓✓	8, 9 100- ✓✓	
group 3	{ 7 0111 ✓✓✓	8, 10 10-0 ✓✓	
	14 1110 ✓✓✓	5, 7 01-1 PI2 ✓✓	
		6, 7 011- PI3 ✓✓	
		6, 14 -110 ✓✓	
		10, 14 1-10 ✓✓	

$\text{PI}_4 = -00- = \bar{b}\bar{c}$   
 $\text{PI}_5 = -0-0 = \bar{a}\bar{d}$   
 $\text{PI}_6 = --10 = \bar{c}\bar{d}$

$$\text{Example 1: } F(a,b,c,d) = \sum m(0,1,2,5,6,7,8,9,10,14)$$

■ Step 2:

$$PI_1 = (1, 5) \quad PI_2 = (5, 7) \quad PI_3 = (6, 7) \quad PI_4 = (0, 1, 8, 9) \\ PI_5 = (0, 2, 8, 10) \quad PI_6 = (2, 6, 10, 14)$$





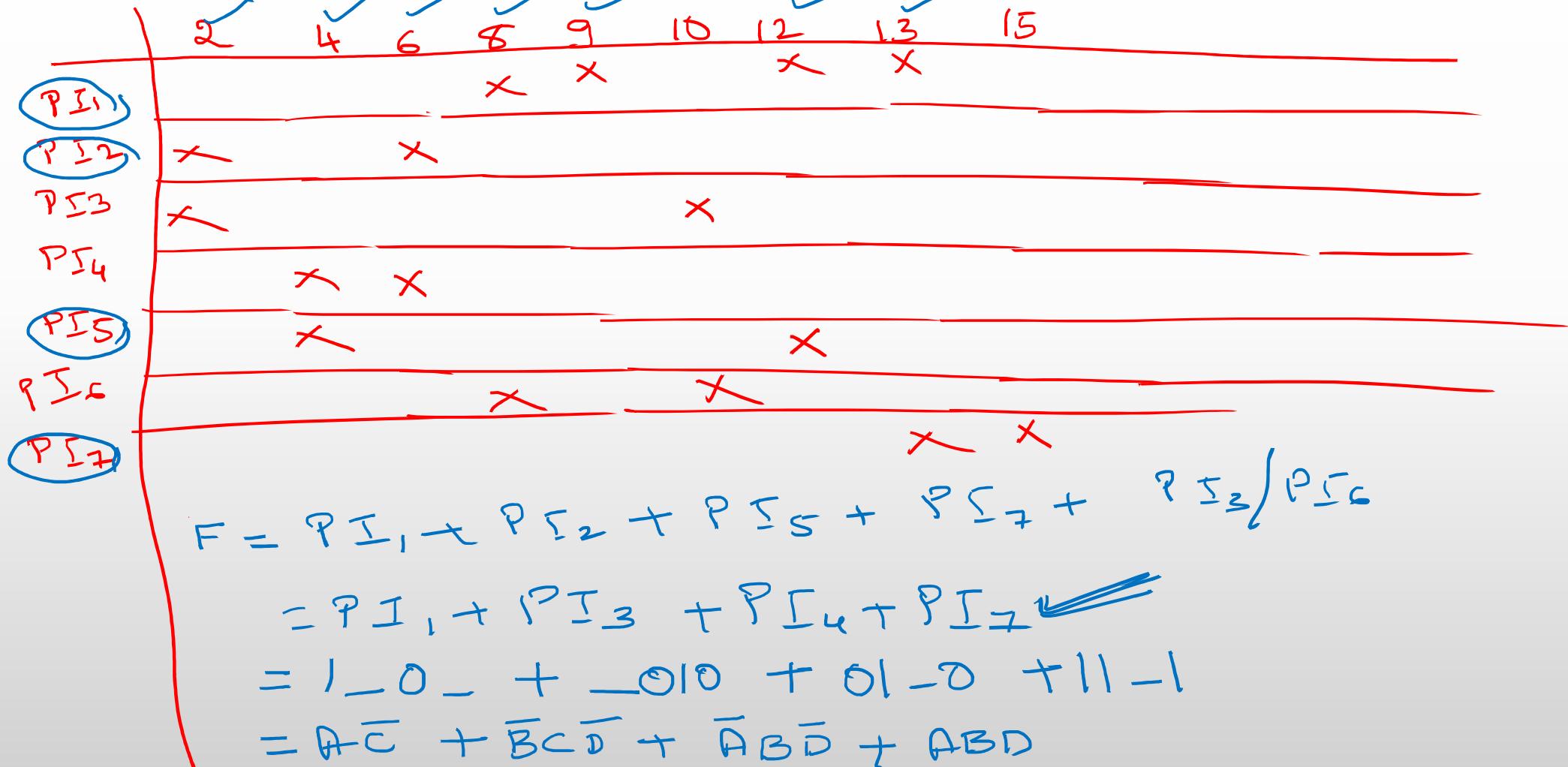
Example 2:

$$f(A, B, C, D) = \sum m(2, 4, 6, 8, 9, 10, 12, 13, 15)$$

List 1			List 2			List 3		
Minterm	ABCD		Minterms	ABCD		Minterms	ABCD	
2	0010	✓	2, 6	0-10	PI <sub>2</sub>	8, 9, 12, 13	1-0-	PI <sub>1</sub>
4	0100	✓	2, 10	-010	PI <sub>3</sub>			
8	1000	✓	4, 6	01-0	PI <sub>4</sub>			
6	0110	✓	4, 12	-100	PI <sub>5</sub>			
9	1001	✓	8, 9	100-	✓			
10	1010	✓	8, 10	10-0	PI <sub>6</sub>			
12	1100	✓	8, 12	1-00	✓			
13	1101	✓	9, 13	1-01	✓			
15	1111	✓	12, 13	110-	✓			
			13, 15	11-1	PI <sub>7</sub>			

$$\begin{aligned}
 f(A, B, C, D) &= PI_1 + PI_3 + PI_4 + PI_7 \\
 &= 1-0- + -010 + 01-0 + 11-1 \\
 &= A\bar{C} + \bar{B}C\bar{D} + \bar{A}B\bar{D} + ABD
 \end{aligned}$$

Step 2 for  $F(A,B,C,D) = \sum m(2,4,6,8,9,10,12,13,15)$



$$\text{Example 1 repeated : } F(a,b,c,d) = \Sigma m(0,1,2,5,6,7,8,9,10,14)$$

Step 1: Finding prime implicants using decimal difference

<del>0</del>	<del>(0,1)</del> (1) ✓	<del>(10,14)</del> (4) ✓
<del>1</del>	<del>(0,2)</del> (2) ✓	
<del>2</del>	<del>(0,8)</del> (8) ✓	Column 3
<del>8</del>	<del>(1,5)</del> (4) — PI1	<del>(0,1,8,9)</del> (1,8) — PI4 <del>(0,2,8,10)</del> (2,8) — PI5
<del>5</del>	<del>(1,9)</del> (3) ✓	<del>(0,8,1,9)</del> (8,1)
<del>6</del>	<del>(2,6)</del> (4) ✓	<del>(0,8,2,10)</del> (8,2)
<del>9</del>	<del>(2,10)</del> (8) ✓	<del>(2,6,10,14)</del> (4,8) — PI6
<del>10</del>	<del>(8,9)</del> (1) ✓	<del>(2,10,6,14)</del> (8,4)
<del>7</del>	<del>(8,10)</del> (2) ✓	
<del>14</del>	<del>(5,7)</del> (2) — PI2	
	<del>(6,7)</del> (1) — PI3	
	<del>(6,14)</del> (8) ✓	

Step 2 remains same

$$\begin{aligned}
 & 8421 \\
 & ABCD \\
 & PI_1(4) - 01 \\
 & I = 0001 \\
 & S = 0101 \\
 & = \bar{A} \bar{C} D \\
 PI_2 &= (S,7)(2) \\
 &\hookrightarrow 0101 \\
 & 0111 \\
 & = 01-1 = \bar{A}BD
 \end{aligned}$$

Step 2 for  $F(a,b,c,d) = \Sigma m(0,1,2,5,6,7,8,9,10,14)$

- Same as before...

Example3 :  $F(v,w,x,y,z) = \sum m(4,5,9,11,12,14,15,27,30) + D(1, 17, 25, 26, 31)$



Col. 1	Col 2	Col 2	Col 3 (Quads)
1 ✓	(1,5)(4) - $\bar{P}\Sigma_6$	(15,31)(16) ✓	
4 ✓	(1,9)(8) ✓	(27,31)(4) ✓	(1,9,17,25)(8,16) - $\bar{P}\Sigma$
5 ✓	(1,17)(16) ✓	(30,31)(1) ✓	
9 ✓	(4,5)(1) - $\bar{P}\Sigma_7$		-(9,11,25,27)(2,16) - $\bar{P}$
12 ✓	(4,12)(8) - $\bar{P}\Sigma_8$		
17 ✓	(9,11)(2) ✓		
11 ✓	(9,25)(16) ✓		
14 ✓	(12,14)(2) - $\bar{P}\Sigma_9$		
25 ✓	(17,25)(8) ✓		
26 ✓	(1,15)(4) ✓		
15 ✓	(11,27)(16) ✓		
27 ✓	(14,15)(1) ✓		
30 ✓	(14,30)(16) ✓		
31 ✓	(25,27)(2) ✓		$F(v,w,x,y,z) = \Sigma m(4,5,9,11,12,14,15,27,30) + D(1, 17, 25, 26)$
	(26,27)(1) ✓		
	(26,30)(4) ✓		

$$F = \underline{PI_2 + PI_4 + PI_8 + PI_7} \rightarrow \text{soln}_1 \quad \text{No don't cares}$$

Simplified expression of  $F(v,w,x,y,z) = \sum m(4,5,9,11,12,14,15,27,30) + D(1, 17, 25, 26, 31)$  is

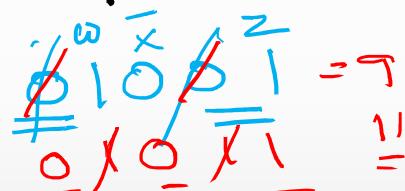
	4	5	9	11	12	14	15	27	30
$\underline{PI_1}$			X						
$\underline{PI_2}$			X	X				X	$\text{Soln}_2 =$
$\underline{PI_3}$				X			X		X
$\underline{PI_4}$						X	X		
$\underline{PI_5}$									
$\underline{PI_6}$			X						
$\underline{PI_7}$	X	X							
$\underline{PI_8}$	X	X				X			
$\underline{PI_9}$					X	X	X	X	$\text{Soln}_1 = PI_2 + PI_4 + PI_8 + PI_7$

TM ( ). DC ( )

Simplified expression of  $F(v,w,x,y,z) = \Sigma m(4,5,9,11,12,14,15,27,30) + D(1, 17, 25, 26, 31)$  is

$$F = PI_2 + PI_4 + PI_8 + PI_6 \text{ or } F = \bar{v}w\bar{x}\bar{y}z$$

$$PI_2 = (9, 11, 25, 27)(2, 16) = \begin{matrix} v & w & x & y & z \\ 16 & 8 & 4 & 2 & 1 \end{matrix}$$

~~prop 1~~  $= \underline{\underline{w\bar{x}z}}$  

$$PI_4 = (14, 15, 30, 31)(1, 16) = \bar{w}xy$$

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

$$PI_8$$

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

$$PI_6 =$$

$$F = \underline{\underline{w\bar{x}z}} + \underline{\underline{wxy}} + \underline{\underline{\bar{v}x\bar{y}\bar{z}}} + \underline{\underline{\bar{v}\bar{w}\bar{y}z}}$$