

Q. What are the advantages and disadvantages of Digital Technology?

Ans. Digital systems are defined as systems that process discrete information using binary values (0 and 1) to achieve a desired goal.

Advantage:

1. Digital systems are generally easier to design.
2. Information storage is easy
3. Accuracy and precision are easier to maintain throughout the system.
4. Operations can be performed
5. Digital circuits are less affected by noise.
6. More digital circuitry can be fabricated on IC chips.

It is true that analog circuitry has also benefited from the tremendous development of IC technology but its relative complexity and its use of devices that cannot be economically integrated have prevented analog systems from achieving the same high degree

of integration.

Subject:

Date:

Disadvantage:—

1. Use more energy than analog circuits to accomplish the ~~same~~ same tasks, thus producing more heat as well.
2. Digital communications require greater bandwidth.
3. Low life span as more dependencies on outer source.
4. Digital computer manipulates discrete elements of information by means of a binary code.
5. Quantization error during analog signal sampling

Subject: Chapter 02

Date:

1. Explain parity bit method for error correction using a parity bit generator circuit and a checker circuit.

(i) Show even parity bit method works in a communication system.

consider a bit string ~~1011~~ 1011

(ii) what is the limitation of parity bit checker.

Ans: A parity Bit is an extra bit that is attached to a code group that is being transferred from one location to another.

The parity bit is made either 0 or 1 depending on the number of 1s that are contained in the code group two different methods are used.

There are two types of parity bit.

1. Even parity bit &

2. odd parity bit

Even parity:- If the data has even number of 1s, then the parity bit is 0.

example: Data: 11000
with parity bit: 0 11000

Subject:

Date:

If the data has odd number of 1s, then the parity bit is 1.

Example: Data: 010011

with the parity bit : 1 010011

1 1011

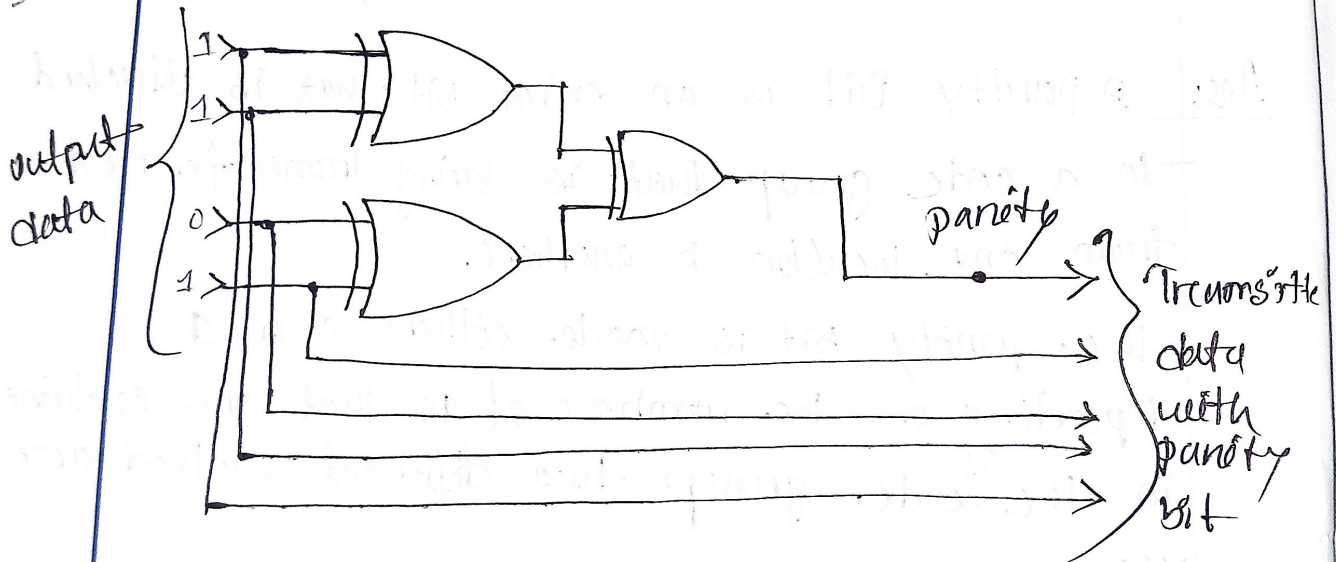


Fig: even-parity generator

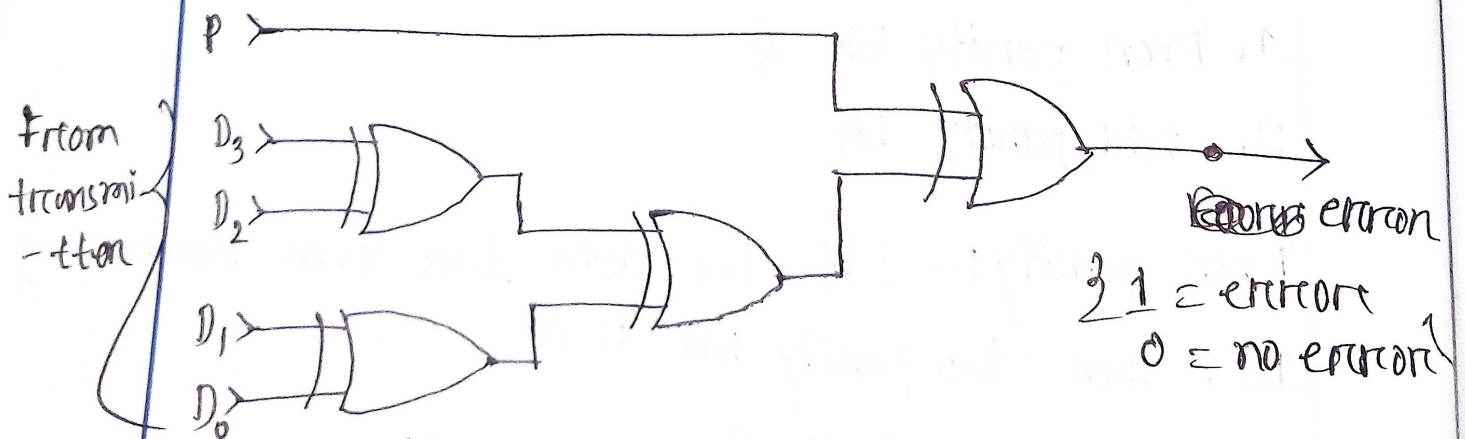


Fig: even-parity checker

Subject:

Date:

Odd parity :- If the data has odd number of 1s, then the parity bit is 0

example: Data: 110001

with parity bit: 0 110001

If the data has even number of 1s, then the parity bit is 1

example: Data: 11000

with parity bit: 1 11000

(ii) The limitation of parity bit checker:-

1. Single-bit Error detection only:-

Detects single-bit errors, but misses even-numbered errors.

2. Cannot correct Errors: Detects errors but lacks the ability to correct them.

3. Ineffective for Burst Error: Poor

detection of errors affecting multiple adjacent

Subject:

Date:

bit,

4. limited to simple transmission scenarios:
Suitable only for low-error-rate environments.

5. ~~Overhead~~ Overhead: Adds extra data to
each transmission, increasing size slightly.

1. (a) 1

(i) 1

(ii) 1

(iii) 1

Ans: 6

(i)

1. (a) Implement the boolean function: -

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

(i) with AND, OR and inverter gates.

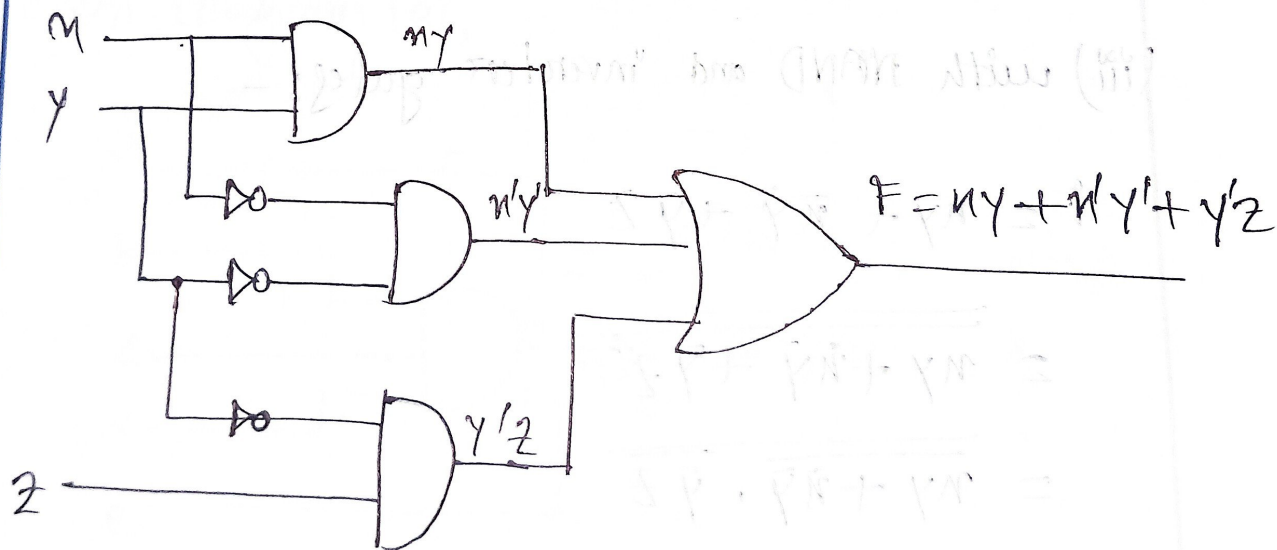
(ii) with NOR, and inverter gates.

(iii) with NAND and inverter gates.

Ans: Given boolean function,

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

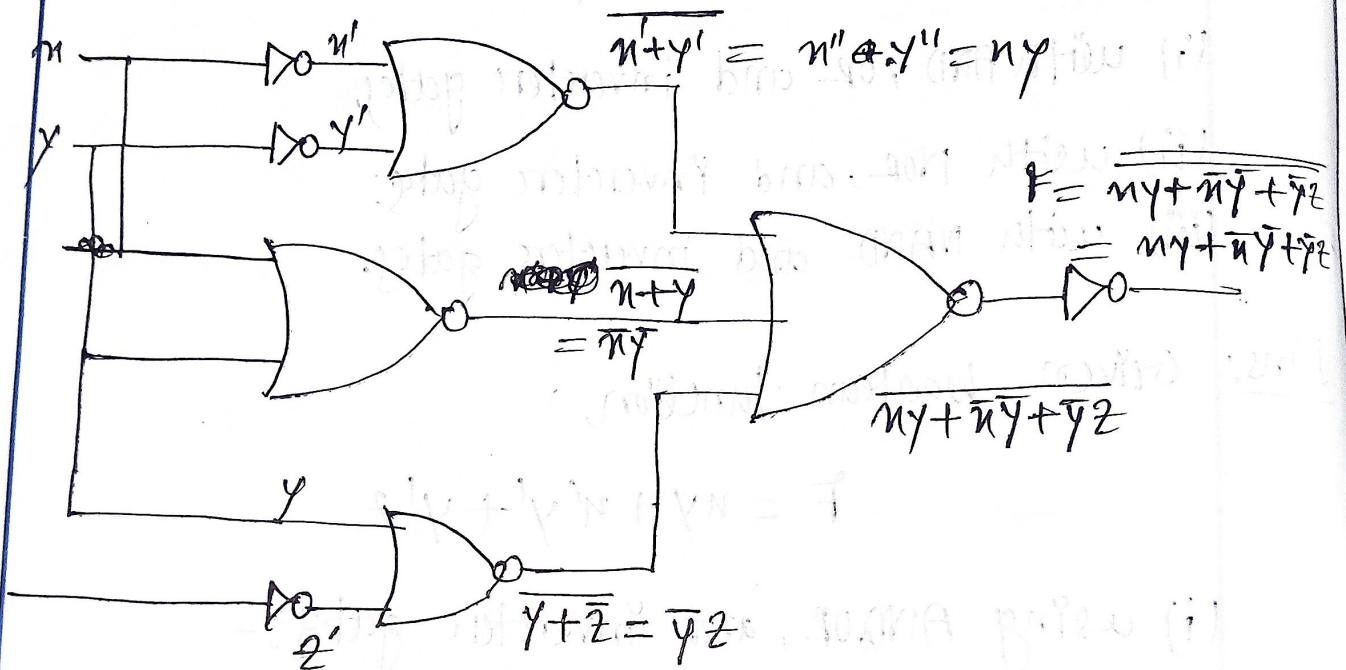
(i) using AND, OR, and inverter gates: -



Subject:

Date:

(ii) with NOR and inverter gates:-



(iii) with NAND and inverter gates:-

$$F = xy + \bar{x}\bar{y} + yz$$

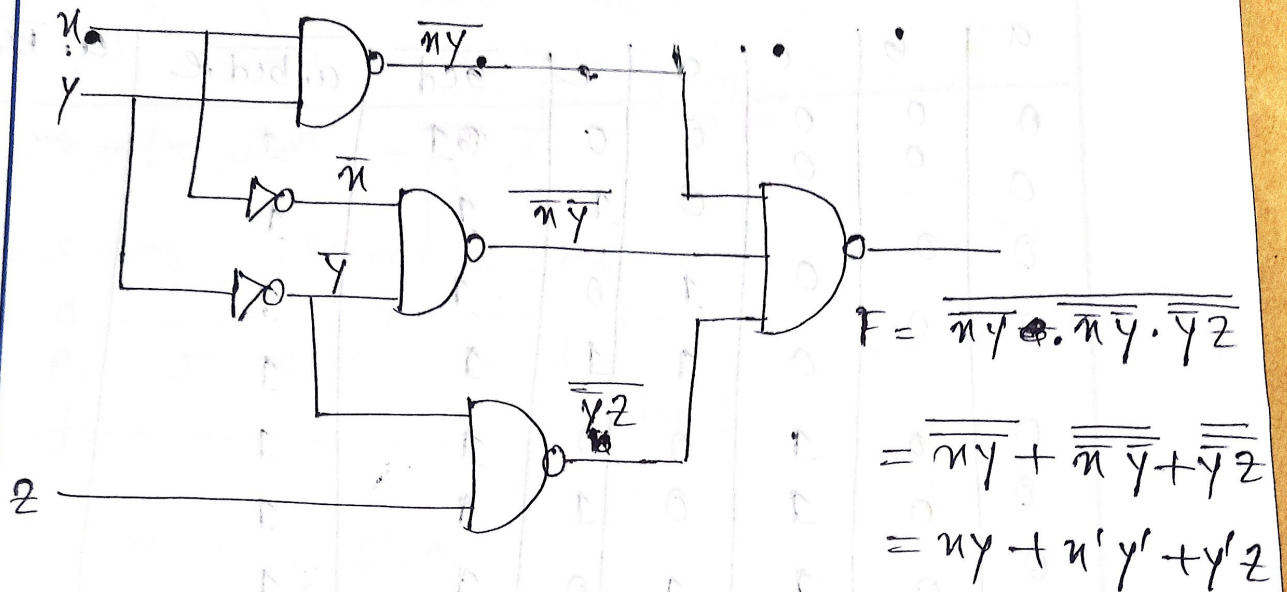
$$= \overline{\overline{xy + \bar{x}\bar{y} + yz}}$$

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

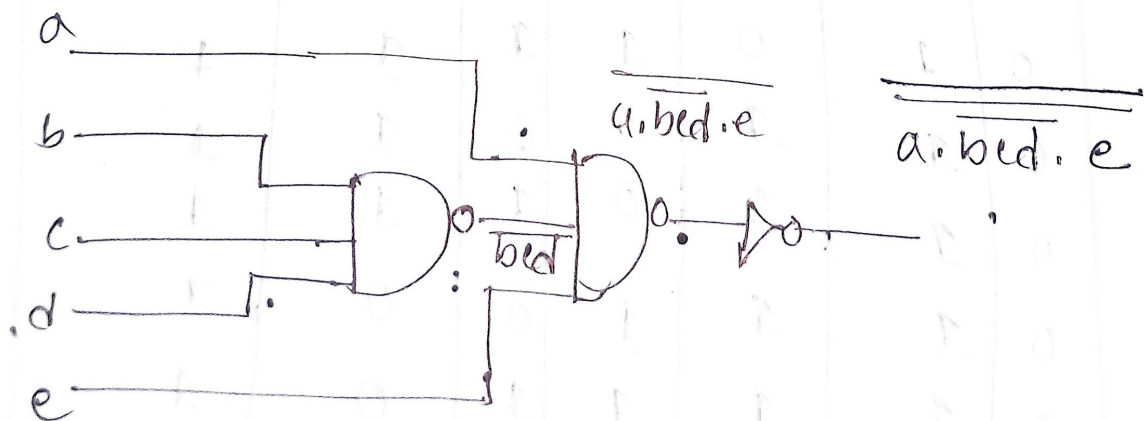
$$= \overline{\overline{xy} \cdot \overline{\bar{x}\bar{y}} \cdot \overline{yz}}$$

Subject:

Date:



2. ~~It~~ construct truth table & Boolean expression, given diagram (a),



boolean expression,

$$y = \overline{a.bcd.e} = a.bcd.e$$

Subject:

Date:

truth tables

a	b	c	d	e	$\overline{bcd}$	$a \cdot \overline{bcd} \cdot e$	$a \cdot bcd \cdot e$
0	0	0	0	0	1	1	
0	0	0	0	1	1	1	
0	0	0	1	0	1	1	
0	0	0	1	1	1	1	
0	0	1	0	0	1	1	
0	0	1	0	1	1	1	
0	0	1	1	0	1	1	
0	0	1	1	1	1	1	
0	1	0	0	0	1	1	
0	1	0	0	1	1	1	
0	1	0	1	0	1	1	
0	1	0	1	1	1	1	
0	1	1	0	0	1	1	
0	1	1	0	1	1	1	
0	1	1	1	0	0	1	
0	1	1	1	1	0	1	
1	0	0	0	0	1		
1	0	0	0	1	1		

Give

a

b

c

d

e

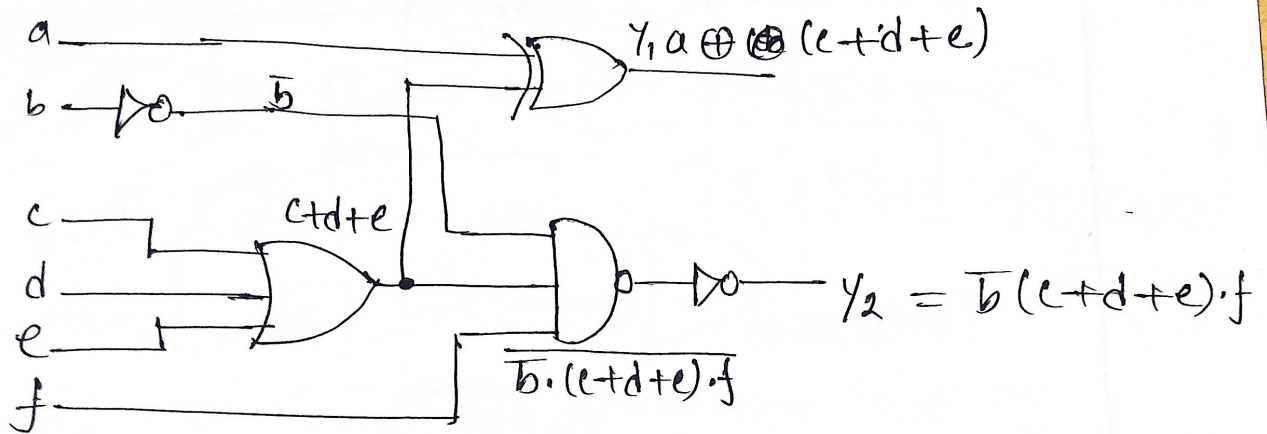
f

Bo

Subject:

Date:

Given diagram (b),



Boolean expression,

$$Y_1 = a \oplus (c+d+e)$$

$$Y_2 = \bar{b} \cdot (c+d+e) \cdot f$$

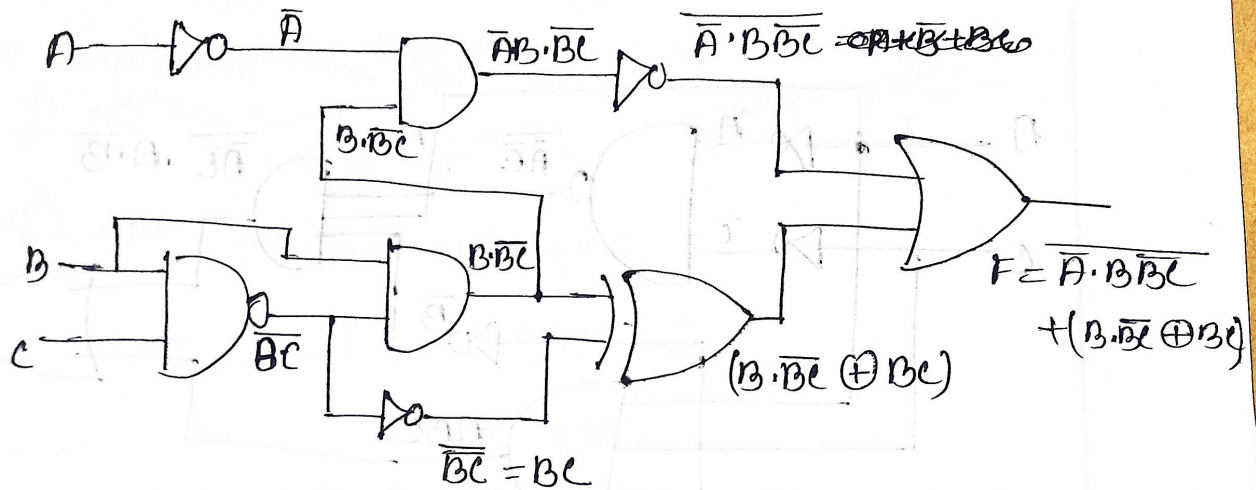
Simplify, $Y_1 = a \cdot (c+d+e) + \bar{a} \cdot (c+d+e)$

Simplify the circuit :-

Subject:

Date:

Ans



Boolean Function,

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

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

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

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

$$= A + \bar{B} + \bar{C} + (\bar{B} \cdot \bar{C} + B \cdot C) \quad [\because \bar{B} + B = 1]$$

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

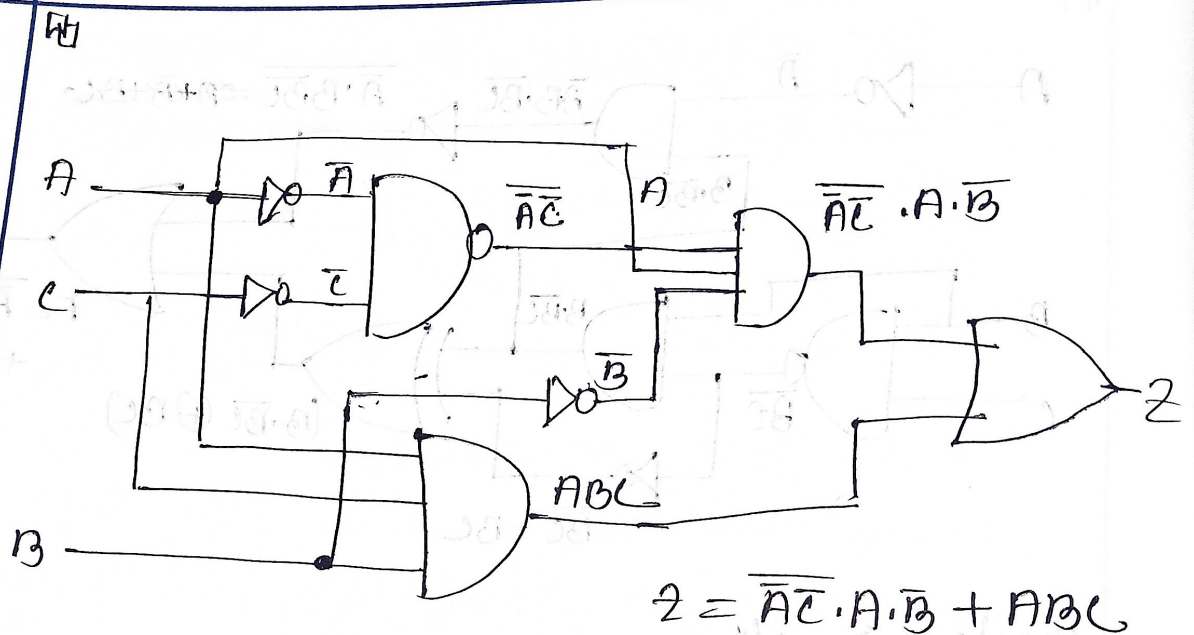
$$= A + \bar{B} + \bar{C} + 0 + B \cdot C + 0 + B \bar{C} \quad [\because B \cdot B = 0]$$

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

$$= A + \bar{B} + B(C + \bar{C}) = A + \bar{B} + B = A + 1 = 1 \quad [\because C + \bar{C} = 1, B + \bar{B} = 1]$$

Subject:

Date:



Boolean expression,

$$\begin{aligned}
 Z &= \overline{A}\overline{C} + A.B \\
 &= (\overline{A} + \overline{C}) \cdot A.B + A.BC \\
 &= (A + \overline{C}) \cdot A.B + A.BC \\
 &= A.A\overline{B} + A\overline{B}C + A.BC \\
 &= A\overline{B} + A\overline{B}C + A.BC \\
 &= A\overline{B}(C + 1) + A.BC \\
 &= A\overline{B} + A.BC \\
 &= A(\overline{B} + BC) \\
 &= A(\overline{B} + C)
 \end{aligned}$$

$$\begin{aligned}
 &\because A + 1 = 1 \\
 &\overline{A} + A.B = \overline{A} + B
 \end{aligned}$$

Subject:.....

Date:.....

$$= AB + BC$$

Q. Implement the function using NAND gate:-

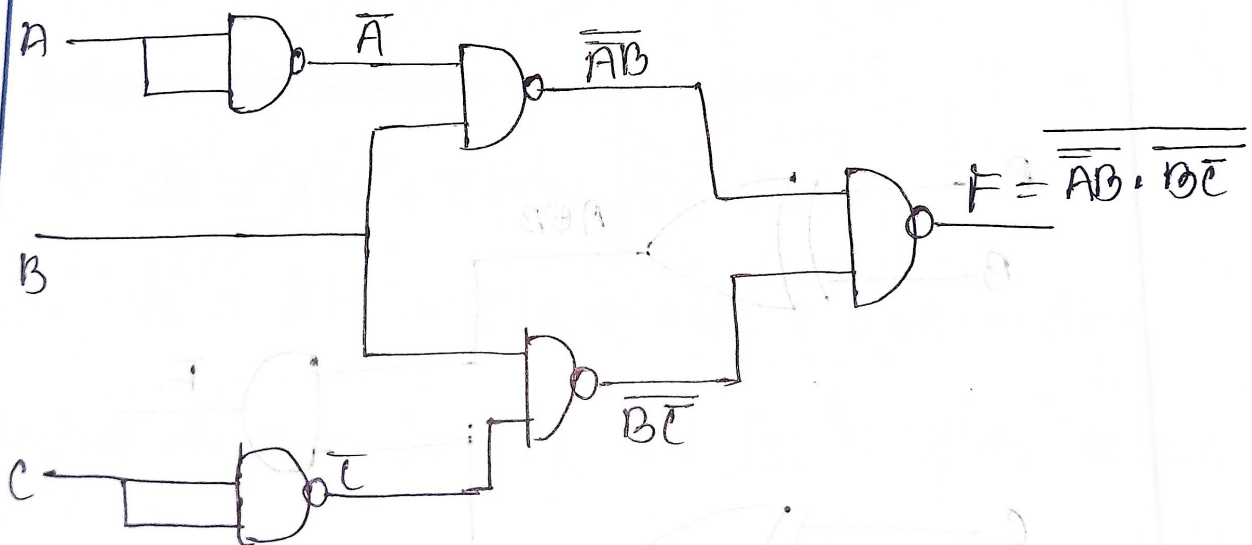
$$F = A'B + BC'$$

Solⁿ: Give boolean expression,

$$F = A'B + BC'$$

$$= \overline{AB} + \overline{BC}$$

$$= \overline{\overline{AB}} \cdot \overline{\overline{BC}}$$



Subject:

Date:

Q Implement the following Boolean Expression with exclusive-OR and AND gates:-

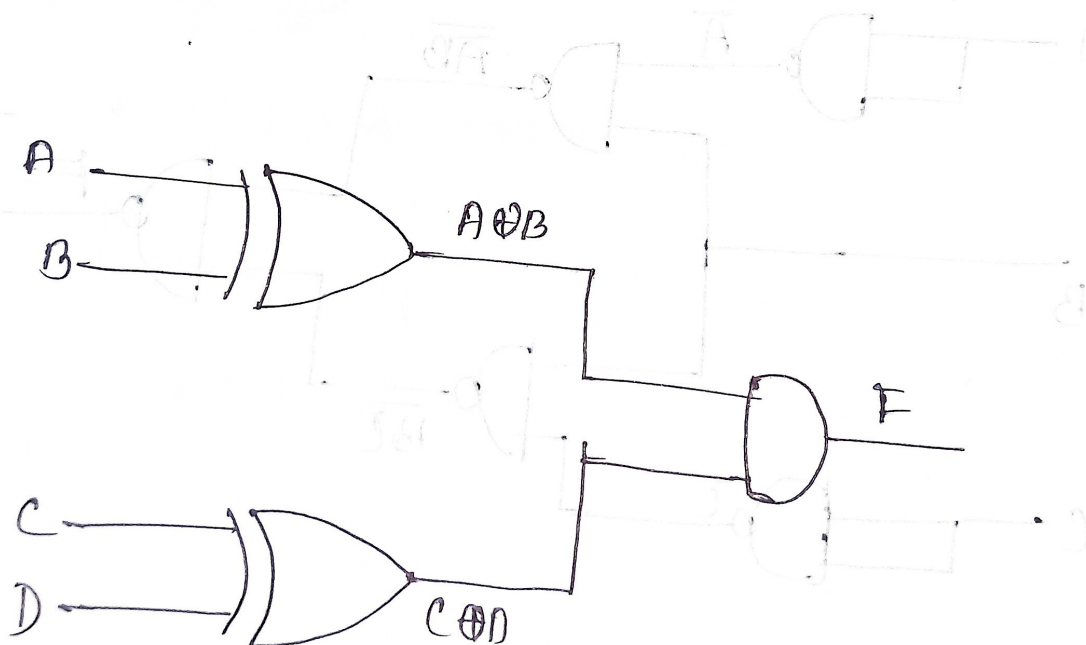
$$F = AB'CD' + A'B'CD' + AB'C'D + A'BC'D$$

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

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

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

$$= (A \oplus B)(C \oplus D)$$



At Boolean expression & draw
logic circuit

Subject:.....

Date:.....

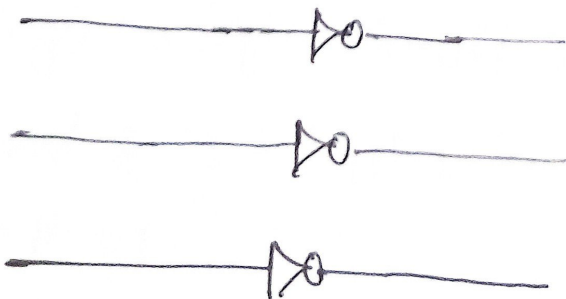
f_1	f_2	a	b	c
1	0	0	0	0
0	0	0	0	1
0	1	0	1	0
1	1	0	1	1
0	1	1	0	0
0	1	1	0	1
1	1	1	1	0
1	0	1	1	1

Boolean expression,

$$f_1 = \bar{a}\bar{b}\bar{c} + \bar{a}bc + ab\bar{c} + abc$$

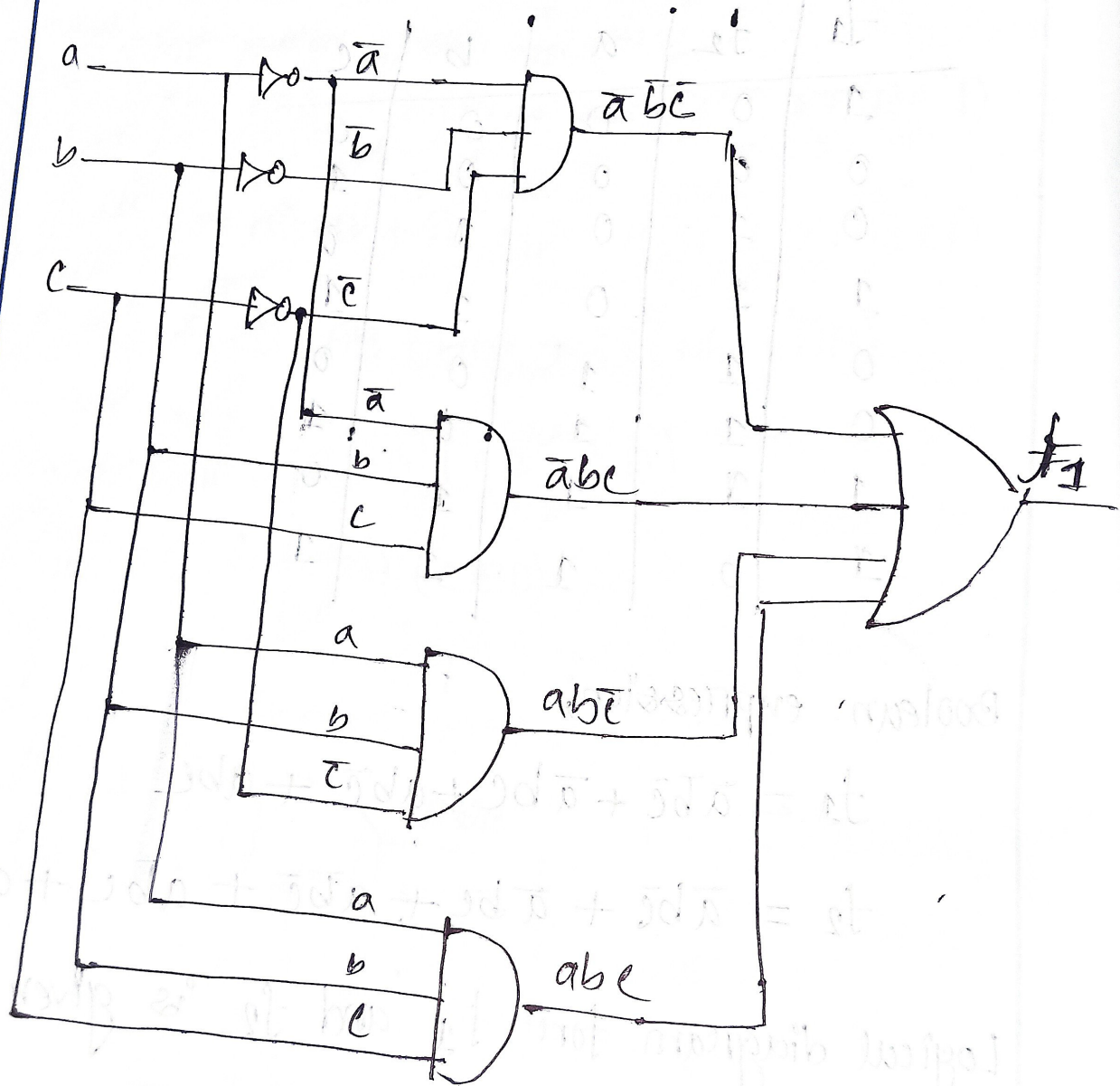
$$f_2 = \bar{a}b\bar{c} + \bar{a}bc + a\bar{b}\bar{c} + a\bar{b}c + ab\bar{c}$$

Logical diagram for f_1 and f_2 is given below,



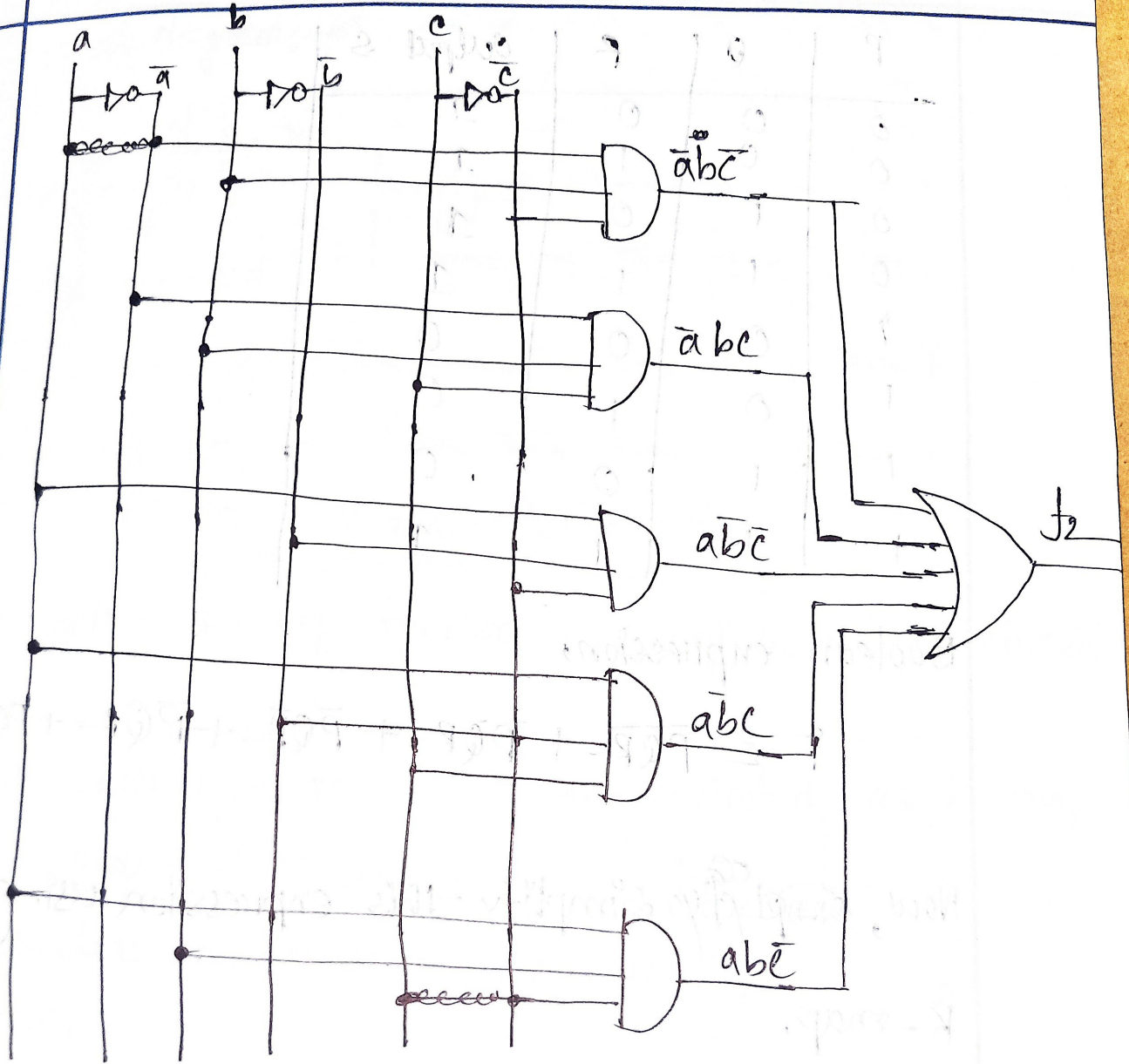
Subject:

Date:



Subject:

Date:



Q1 Design a logic circuit with inputs P, Q, R so that output S is HIGH whenever P is '0' or, whenever $Q = R = 1$

Subject:

Date:

P	Q	R	output S
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Boolean expressions

$$F = \bar{P}\bar{Q}\bar{R} + \bar{P}\bar{Q}R + \bar{P}Q\bar{R} + \bar{P}QR + PQ\bar{R}$$

Now, ~~simplify~~ simplify this expression using K-map.

P \ QR	$\bar{Q}\bar{R}$ 00	$\bar{Q}R$ 01	QR 11	$Q\bar{R}$ 10
($\bar{P}$) 0	1 0	1 1	1 3	1 2
(P) 1			1 7	
	4	5	6	

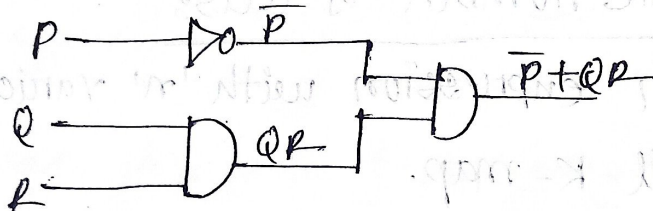
~~Handwritten scribbles~~

$$F = \bar{P} + QR$$

Subject:.....

Date:.....

logical diagram: +



k-map problems

Q Explain k-map method to simplify a logic circuit.

solⁿ:

The karnaugh map, often abbreviated as k-map is a graphical method used for simplifying Boolean expressions and consequently simplifying logical circuit.

It's particularly useful for circuits with a small number of inputs.

Here's a step by step explanations of how the k-map method works:-

1. Create the Karnaugh Map:-

Determine the number of cells:-

For a boolean expression with 'n' variables, create a 2^n cell K-map.

Label Rows and columns:-

Label the rows and columns with binary representation of the input variables.

2. Fill in the K-map:-

Write down the function values:- For each cell in the K-map, write down the corresponding function value obtained from the boolean expression.

3. Group ones:-

Identify group of 1's:- Group adjacent cells containing 1's (power of 2: 1, 2, 4, 8, ...))

4. Create Implicants:-

Create implicants from groups: Each group

Subject:.....

Date:.....

group represents a term in the simplified Boolean expression. write down the implicants for each group.

5. From simplified expression:-

combine implicants:- combine the implicants to form the simplified boolean expression.

Q7 List the limitation of k-map:-

1. limited variable scalability
2. Inefficiency with many terms
3. Difficulty with don't care condition.
4. Limited for non-square Functions.
5. Limited for non-mid term Expressions.
6. Human Error in manual construction.
7. Insensitivity to physical Implementation.
8. Not ideal for sequential circuits.
9. Lack of Direct support for x -or and NOR operations.

Q. Simplify the following Boolean Function

Subject:

Date:

Sol: Given Boolean expression,

$$F = B'DE' + A'BE + B'C'E' + A'BC'D'$$

$$D = BDE' + CD'E'$$

A	B	C	D	E	F
0	0	0	0	0	1
0	0	0	0	1	
0	0	0	1	0	
0	0	0	1	1	
0	0	1	0	0	
0	0	1	0	1	
0	0	1	1	0	
0	0	1	1	1	
0	1	0	0	0	
0	1	0	0	1	
0	1	0	1	0	
0	1	0	1	1	
0	1	1	0	0	
0	1	1	0	1	
1	1	1	1	0	
1	1	1	1	1	
1	0	0	0	0	
1	0	0	0	1	
1	0	0	1	0	
1	0	0	1	1	

A B C D E F

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

Subject:.....

Date :.....

k-maps-

	000	001	011	010	100	101	111	110
00								
01								
11								
10								

Subject:.....

Date:.....

using K-map : -
when, $A = 0$

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

Subject:.....

Date:.....

Mention the Duality Principle:—

The principle states that any algebraic equality derived from these axioms will still be valid whenever the OR and AND operators, and identity elements 0 and 1, have been interchanged.

changing every OR into AND and vice versa, and every 0 into 1 and vice versa.

Q The dual of the exclusive-OR is also its complement.

Sol: Let a Boolean expression for X-OR

$$x \oplus y = \bar{x}y + x\bar{y}$$

Complement of X-OR gate expression,

$$\overline{x \oplus y} = \overline{\bar{x}y + x\bar{y}}$$

$$= \overline{\bar{x}y} \cdot \overline{x\bar{y}}$$

$$= (\bar{\bar{x}} + \bar{y}) \cdot (\bar{x} + \bar{\bar{y}})$$

$$= (x + \bar{y})(\bar{x} + y) \quad \text{--- (1)}$$

[Using De-morgan's theorem]

Subject:.....

Date:.....

$$\begin{aligned}\text{Dual of } x \oplus y &= \text{Dual of } (\bar{x}y + x\bar{y}) \\ &= (\bar{x} + y) \cdot (x + \bar{y}) \quad \text{--- (ii)}\end{aligned}$$

$$\therefore (x \oplus y) = \text{Dual of } (x \oplus y)$$

[Showed]

De Morgan's Theorem:

We use De Morgan's theorem to solve the expression of Boolean Algebra. It is a very powerful tool used in Digital design.

This Theorem explains that the terms are equal to the products of all terms are equal

This theorem explains that the complements of the products of all the terms are equal to the sums of the complements of each and every term.

Subject:

Date:

Like wise, the complements of the sums of all the terms are equal to the products of the complements of each and every term.

Theorem 1: $\overline{A \cdot B} = \overline{A} + \overline{B}$

Theorem 2: $\overline{A + B} = \overline{A} \cdot \overline{B}$

Q. What are different methods to write a logical expression from truth table? Explain with an Example

Solⁿ: There are several methods to write a logical expression from truth table.

Two common method are the sum of products (sop) and product of sum (pos) forms.

Sum of products (sop): -

1. Identify the rows where the output is 1
2. For each row, write a product term (AND) of literals where each literal corresponds to an input variable or its complement.

Subject:

Date:

3. Combine these product terms with sum (or)

Example:

A	B	C	output
0	0	0	1
0	1	1	1
1	0	1	1
1	1	0	0

$$\text{sum of products} = \bar{A}\bar{B}\bar{C} + \bar{A}BC + A\bar{B}C$$

Product of sum (POS):

(1) Identify the rows where the output is 0.

(2) For each row, write a sum term (or) of literals where each literal corresponds to an input variable or its complement.

(3) Combine these sum terms with a product (AND).

Subject:.....

Date:.....

Examples:—

A	B	C	output
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	0

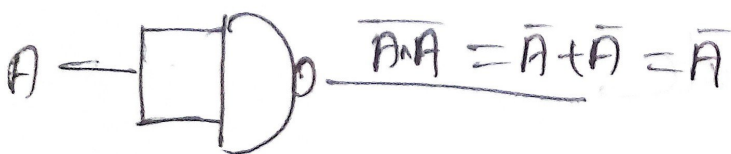
product of sum: $(\bar{A} + B + C)(A + \bar{B} + C)(A + B + \bar{C})$

Q. what do you understand by Universality of logic gate? Show that the NOR gate is a universal gate.

Sol: A universal gate is a gate which can implement any boolean function without need to use any other gate type.

The NAND and NOR gates are called universal gates because they can be combined to produce any of the other gates like OR, AND and NOT gates.

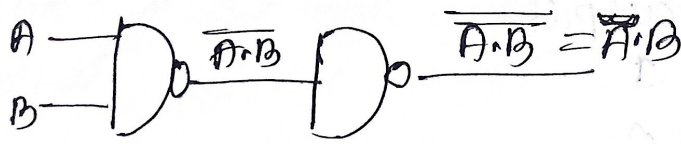
Implementation of NOT, AND and OR gates by NAND gates:—



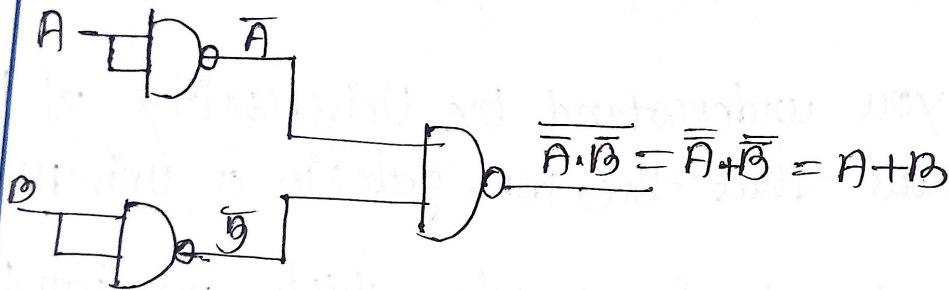
(A) NOT gate

Subject:

Date:

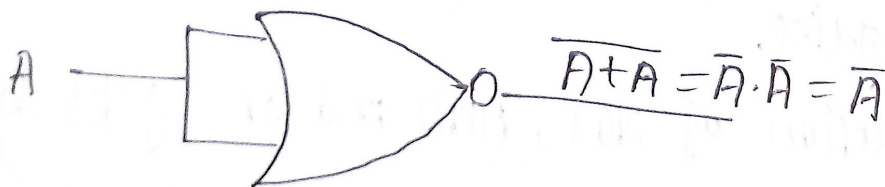


(b) AND gate implementation



(c) OR gate implementation.

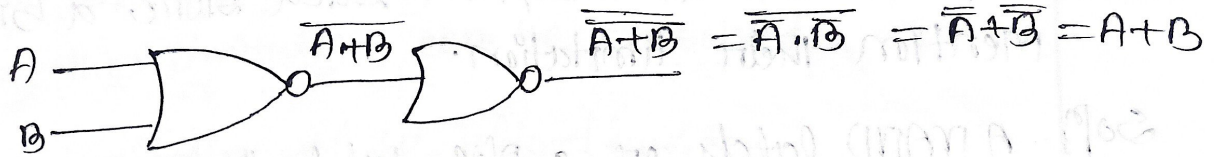
Implementation of NOT, AND and OR gates by NOR gates: -



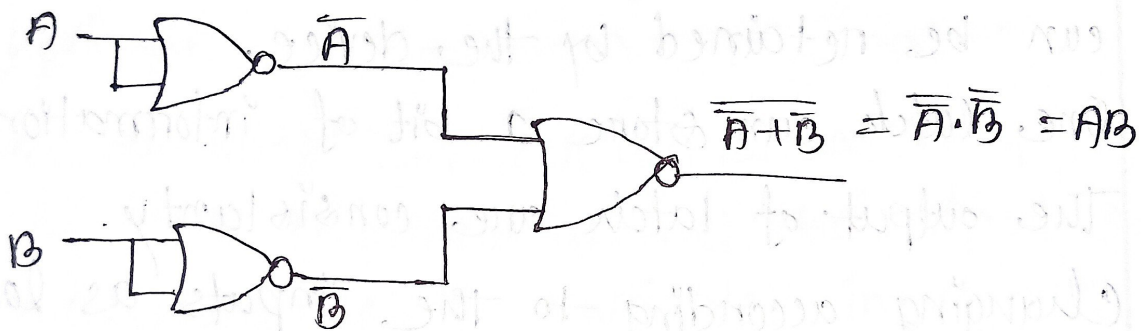
(a) NOT gate

Subject:.....

Date :.....



(b) OR gate



(c) AND gate

NAND gate is a series combination of AND and NOT gates.

NOR gate is a series combination of OR and NOT gates.

1. Explain how a NAND/NOR latch stores a bit. Mention their limitation.

Soln: A NAND latch or a NOR latch is a simple digital circuit that can store one bit of information, either a '0' or a '1'.

A latch has a feedback path, so information can be retained by the device.

One latch can store 1 bit of information.

The output of latch are constantly changing according to the inputs as long as the latch is enabled. In other words, the content of a latch changes immediately when the inputs change when it is enabled.

- A latch which is made up of only NAND gates is called a NAND latch.
- A latch which is made up of only NOR gates is called NOR latch.
- * latch is a digital circuit which converts its outputs according to its inputs instantly.

Subject:.....

Date:.....

2. What is the limitation of S-R flip-flop? Explain how the limitation can be resolved.

Sol: The limitation of SR Flip flop:-

1. Forbidden state: It can enter a forbidden state when both the set (S) and Reset (R) inputs are active simultaneously, leading to unpredictable behaviour.

2. Race conditions: In certain situations, there can be race conditions where the output state is uncertain due to variable in signal propagation delays.

The limitations of SR Flip Flops can be resolved through various means:-

1. Clock Inputs: Introducing clock inputs can help synchronize the transitions, reducing the risk of forbidden states.

This is commonly seen in edge-triggered flip-flop like the D-flip-flop, where changes only occur on clock edges.

Subject:.....

Date:.....

2. JK-Flip-flop: using JK flip-flop, which have J(set) & K(Reset) inputs, allows for a more controlled and predictable behaviour. JK flip-flops include a toggle functionality that helps avoid the forbidden state and race conditions.

3. Master Slave Configuration:-

Employing master slave configuration can mitigate race conditions. In master-slave flip-flops, two sets of latches work alternately, reducing the chances of unpredictable states during transitions.

Q1

Explain a clocked SR Flip-flop with wave forms

Subject:

Date:

Solⁿ: An SR Flip-flop also known as Set-Reset flip-flop is a fundamental digital circuit element used to store a single bit of information (0 or 1)

clocked SR Flip-flop:- The type of ~~SR~~ flip-flop ~~which~~ which has two inputs namely S and R is termed as an SR is called a clocked SR flip-flop if the S and R inputs of the flip-flop control its outputs when a clock pulse is present, (goes from either low to high or high to low).

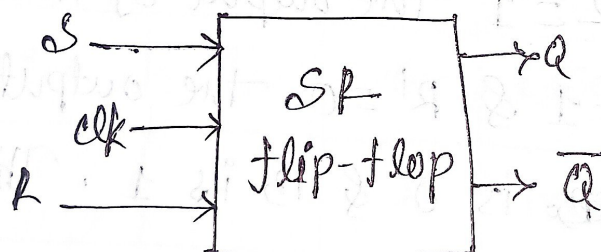
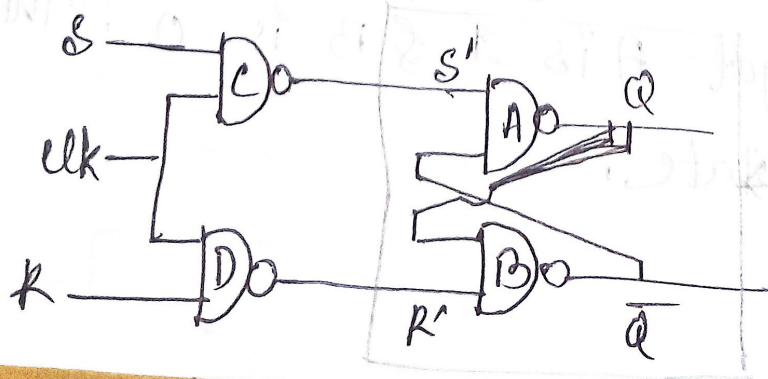


Fig: Clocked SR Flip-flop

The logic circuit diagram:-



Subject:

Date:

Explain:-

when the clock signal is not applied the SR Flip-flop circuits remain inactive and there is no change in the outputs.

when the clock signal is applied the flip-flop circuit becomes active and operates as below:

1. when $S=0$ & $R=0$ the output of NAND gates C & D are $S'=1$ & $R'=1$, so the outputs of the NAND gates A & B remains unchanged.

This is called Hold state.

2. when $S=0$ & $R=1$ the output of NAND gates C & D are $S'=1$ & $R'=0$ the output of the NAND gate A ~~is~~ is 0 & B is 1. This is called Reset state.

3. when $S=1$ & $R=0$ the output of the NAND gates C & D are $S'=0$ & $R'=1$ the output of the NAND gate A is 1 & B is 0. This is called Set State.

Subject:.....

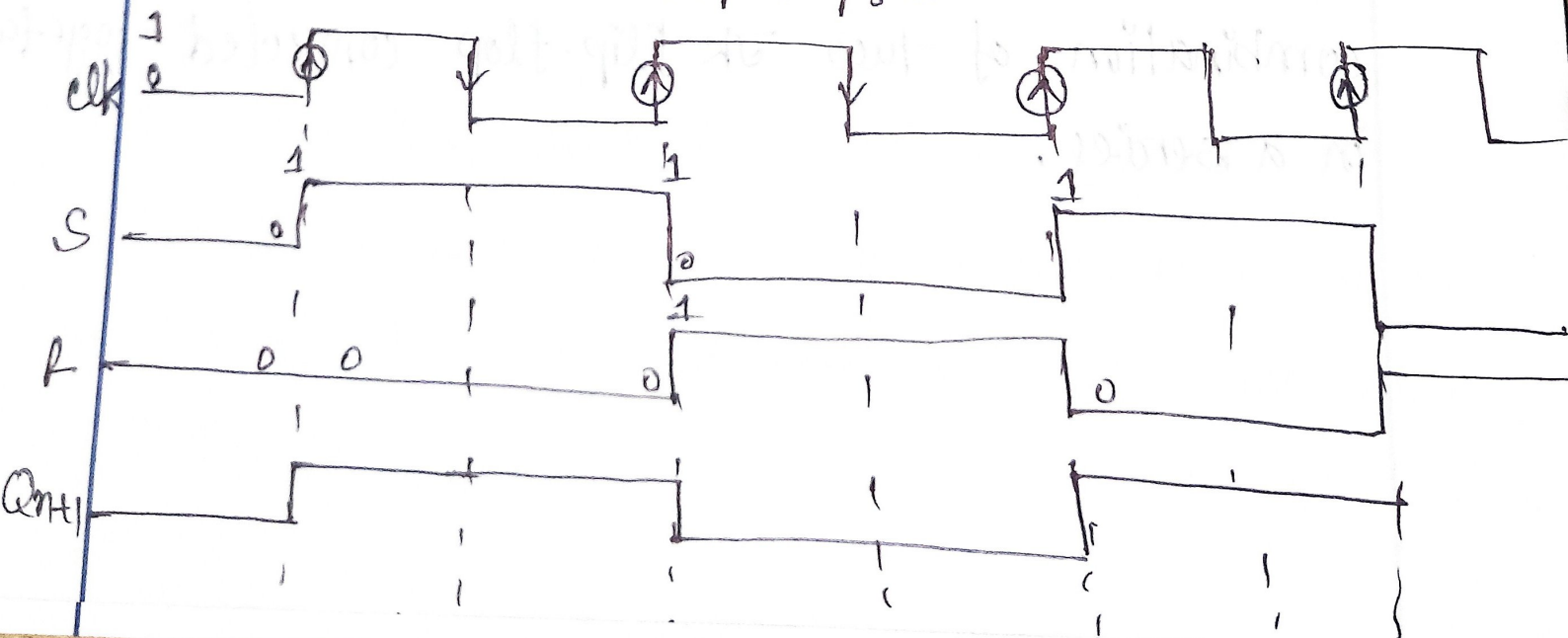
Date:.....

4. when $S=1$ & $R=1$ the output of the NAND gates C & D are $S'=0$ & $R'=0$ the output of the NAND gates A and B try to become 1, which is not possible. This is called Forbidden state.

Truth table of SR Flip-flop:—

clock	S	R	Q_{n+1}
Not Triggered	X	X	Q_n
Triggered	0	0	Hold
"	0	1	0 (Reset)
"	1	0	1 (Set)
"	1	1	Invalid

waveform of SR flip-flop:—



1
Q1 Explain how a SR Flip flop stores a bit.

Subject:.....

Date:.....

Q2 Define race Around condition

In JK Flip-flop, how can you overcome the problem? Explain with appropriate figure & waveform.

Sol:- Race Around condition in J-K flip-flop:-

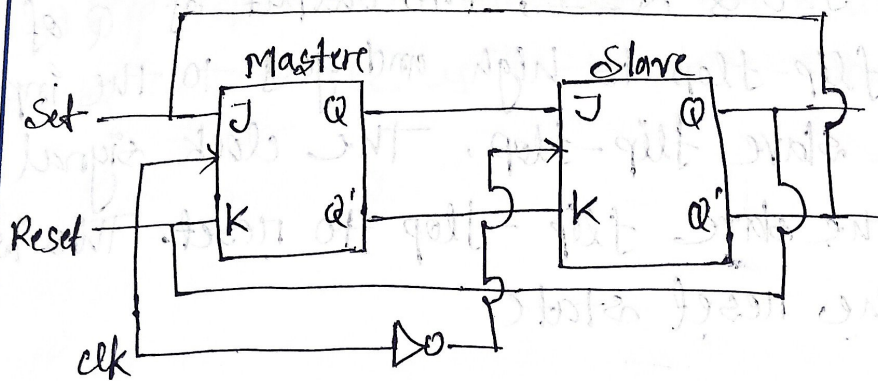
In digital circuits, a race around condition in JK flip flop occurs when both J & K inputs are high, causing the flip-flop to toggle rapidly and unpredictably. This can lead to instability and undefined output state.

To overcome the problem of race around condition in JK Flip-flop we can use the Master Slave JK flip flop.

Master slave JK Flip-flop is basically a combination of two JK Flip-flop connected together in a series.

Subject:

Date:



Truth table :-

J	K	Q_{n+1}
0	0	0 No change / Hold
0	1	0
1	0	1
1	1	Toogle

Explains

1. when $J=0$ & $K=0$, JK flip-flop remains inactive & the output Q remains unchanged. This is called the hold state of the Master slave JK flip-flop.

Subject:.....

Date:.....

2. when $J=0$ & $K=1$, the output of Q of master flip-flop is high and goes to the input K of the slave flip-flop. The clock signal forces the slave flip-flop to reset. This is called the reset state.

3. when $J=1$ & $K=1$ the output of Q of the master flip-flop is high and goes to the input J of the slave flip-flop. The negative transition of the clock signal sets the slave flip-flop.

4. When $J=1$ & $K=1$, the master flip-flop toggles on the positive transition of the clock pulse and the slave flip-flop toggles on the negative transition of the clock pulse.

Hence the problem of the race around condition of the JK Flip-flop is solved.

Q1] What is the limitation of a parallel adder?

Subject: Chapter: 06

Date:

Sol: A parallel adder is a digital circuit that adds multiple binary numbers simultaneously. While parallel adders are commonly used in digital systems, they do have certain limitations. Here are some information.

1. Complexity and size :-

As the number of bits to be added increases, the complexity and size of the parallel adder circuit also increased. This can lead to larger and more intricate circuitry, consuming more resources and potentially increasing power consumption.

2. Propagation delay :-

In a parallel adder, there can be variations in the time it takes for the carry signal to propagate through different stages of the adder. This is known as propagational delay.

As the number of bit increases, the propagational delay also increases potentially impacting the overall speed and performance of the adder.

Subject:

Date:

3. Hardware cost:

The implementation of a parallel adder can require a significant amount of hardware resources, including gates, flip-flops and other components.

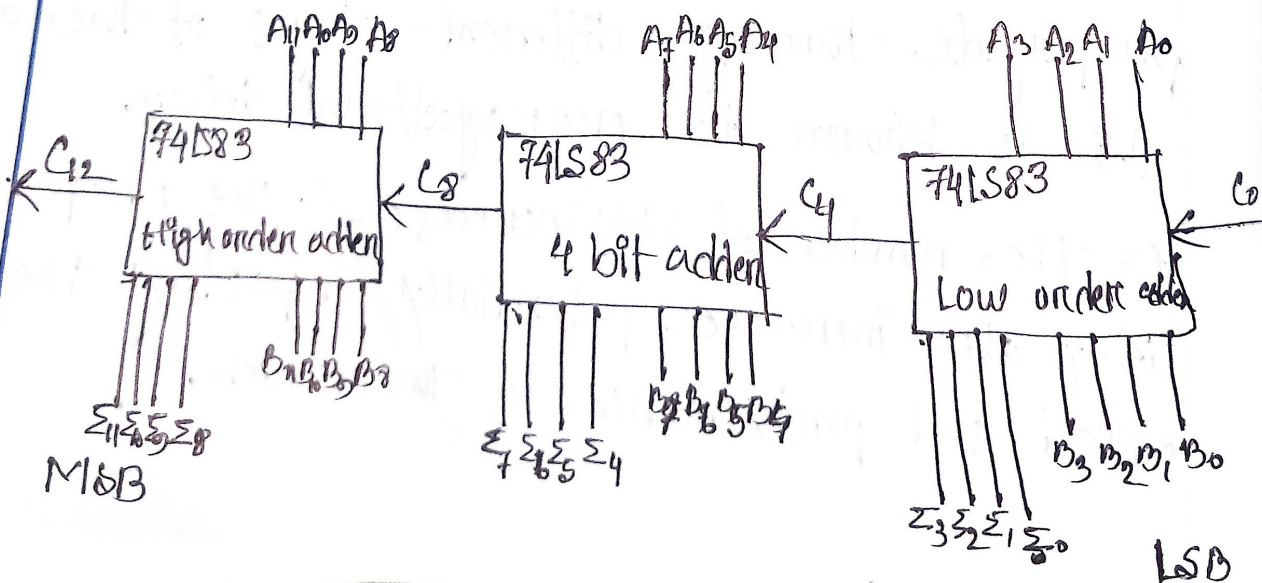
This can increase the overall cost of the digital system.

4. More power consumption.

5. Signal integrity.

Q Using 74LS83 IC's draw a 12 bit parallel adder, 16 bit parallel adder.

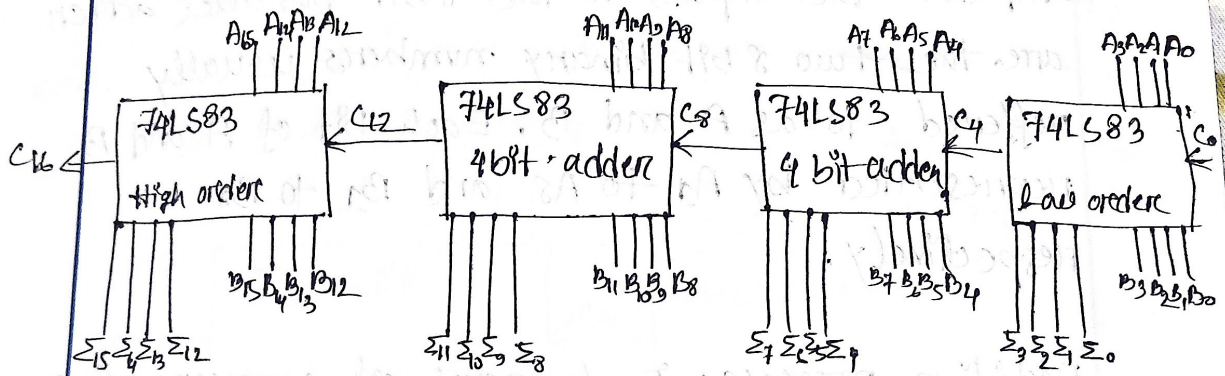
Sol: Drawing a 12 bit parallel adder using 74LS83 IC's



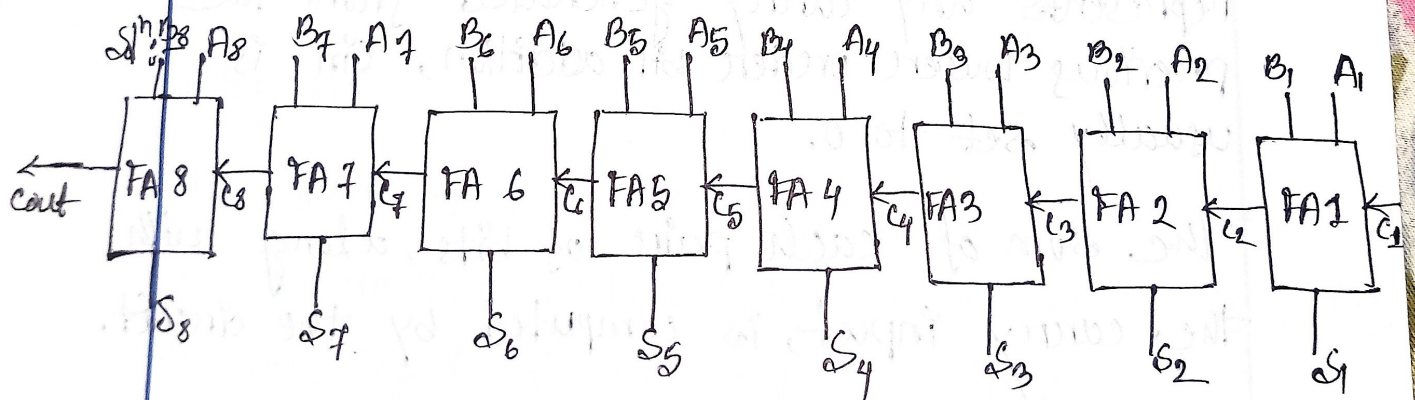
Q16 16 bit parallel adder

Subject:

Date:



Q17 Working principle of a 8-bit parallel adder.



An 8 bit parallel adder is a digital circuit that performs the addition of two 8 bit binary numbers in parallel. It adds each pair of corresponding bits from the two operands. Simultaneously, producing an 8 bit sum output.

Subject:

Date:

Explanation:

Inputs: The inputs to the 8 bit parallel adder are the two 8 bit binary numbers usually referred to as A and B. Each bit of A and B represented by A_1 to A_8 and B_1 to B_8 respectively.

Addition process: Each pair of corresponding bits (A_1 and B_1 , A_2 and B_2 , ..., A_8 and B_8) are added together.

Also, there is a carry input (C_{in}) that represents any carry generated from the previous lower order bit addition, C_{in} is usually set to 0.

The sum of each pair of bits, along with the carry input, is computed by the circuit.

Outputs:

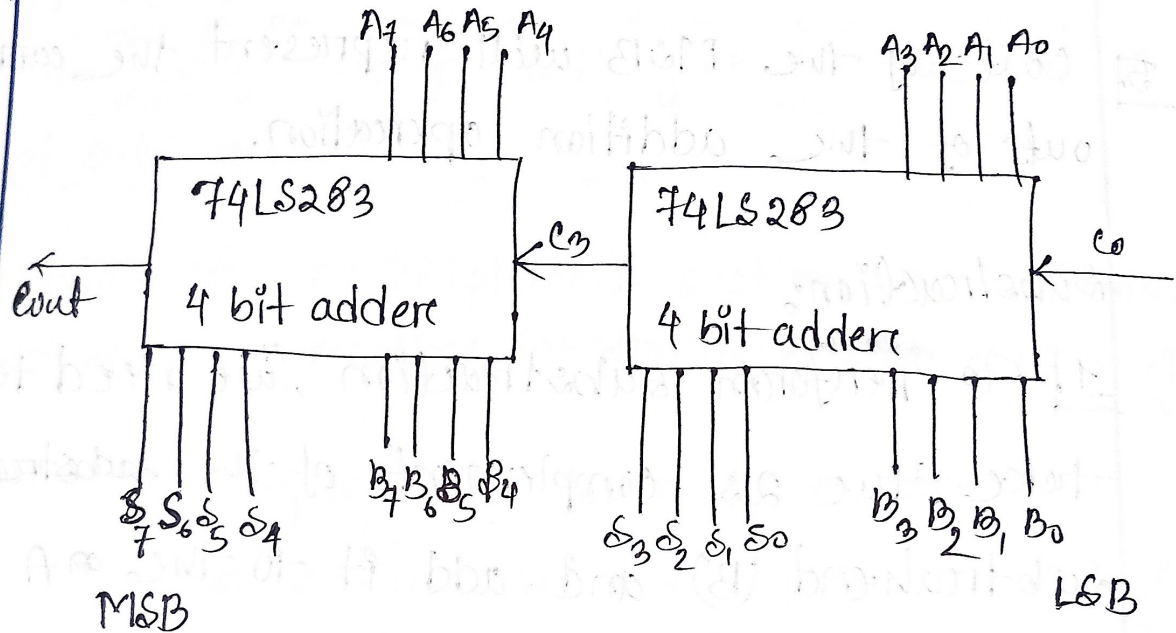
The outputs of the 8 bit parallel adder are the 8-bit sum (S_1 to S_8).

Q. Draw a 8-bit parallel adder using 74LS283 ICs which will be able to perform both subtraction and addition operation.

Subject:

Date:

Sol:



Addition:

1. Connect the corresponding bits (A_0 to A_7 and B_0 to B_7) of the two 8-bit operands (A & B) to the input of the 74LS283 ICs

2. Set the C_{in} of the least significant 74LS283 IC to 0.

3. Connect the C_{out} (carry-out) of each 74LS283 IC to the C_{in} of the next higher order 74LS283 IC.

Subject:

Date:

4] The sum outputs of each stage will give us the binary sum of the operands.

5] Cout. of the MSB will represent the carry out of the addition operation.

Subtraction:

1] To perform subtraction, we need to take the 2's complement of the ~~subtrahend~~ subtrahend (B) and add it to the ~~A~~ A

2] to compute the 2's complement of B, invert all bits of B and add 1.

3] connect the 8-bit A to the A inputs of 74LS283 ICs and the 8-bit 2's complement of the B to the B inputs

4] Set the Cin of the LSB ~~74LS283~~ 74LS283 to 1 (since we are adding the 2's complement)

Subject:.....

Date:.....

5] connect the cout of each stage to the cin of the next higher stage.

The sum outputs will give us the result of subtraction.

Q] Explain how 2's complement system can facilitate arithmetic operations in digital ~~electro~~ computing.

sol:] The 2's complement system is a binary representation scheme used to represent integers in digital computing. In this system, negative numbers are represented by taking the complement (flipping all bits) of the positive counterpart and adding 1 to the result.

The 2's complement system has several advantages, particularly in facilitating arithmetic operations in digital computing.

Addition and Subtraction:-

Addition and subtraction operations can be performed using the same hardware circuitry.

Subject:

Date:

For both signed and unsigned numbers. This simplifies the design of arithmetic circuits. In digital processors, when adding or subtracting in 2's complement the same rules apply for both positive and negative numbers. There is no need for separate circuits to handle signed and unsigned arithmetic.

Simplifies overflow handling:-

In 2's complement arithmetic, overflow conditions are handled seamlessly. Overflow in the most significant bit (MSB) indicates an overflow condition, and it can be easily detected.

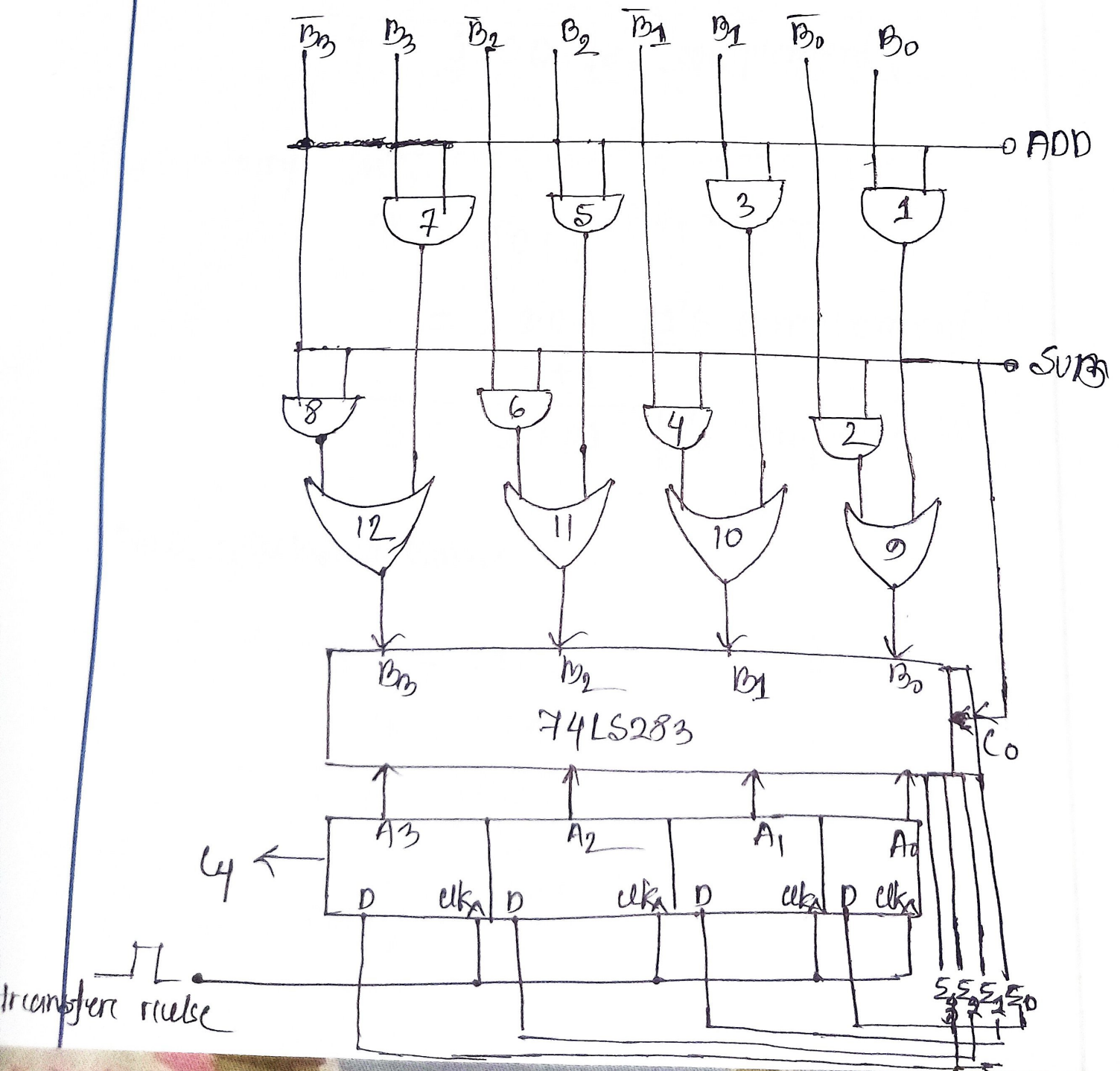
Overflow conditions are treated consistently for both positive and negative numbers making it straightforward to identify when the result of an operation is too large to be represented within the given number of bits.

Subject:

Date:

3. Representation range: The 2's complement representation allows the use of a full range of values for both positive and negative integers within a given number of bits.

□ Parallel adder/subtractor using the 2's complement



Subject:

Date:

Step to subtract 4 from -3 :-

$$-3 - (+4) = -3 - 4$$

we know,

$$\begin{aligned} 4 &= 0100 \quad [4 \text{ bit}] \\ &= 1011 \quad [1's \text{ complement}] \\ &\quad +1 \\ \hline -4 &= 1100 \quad [2's \text{ complement}] \end{aligned}$$

Similarly for,

$$\begin{aligned} 3 &= 0011 \\ &= 1100 \quad [1's \text{ complement}] \\ &\quad +1 \\ \hline -3 &= 1101 \quad [2's \text{ complement}] \end{aligned}$$

Now, ~~subtraction~~

$$\begin{array}{r} -3 = 1101 \\ -4 = 1100 \\ \hline -7 = 101001 \end{array}$$

 / \
 carry bit sign bit

Subject:.....

Date:.....

Q7 Using 2's complement system perform the following operations:-

① $-65 - 88$

$$\begin{aligned}(65)_{10} &= (01000001)_2 \text{ (8 bit)} \\ &= 10111110 \text{ [1's complement]} \\ &\quad +1 \\ \hline &= 10111111 \text{ [2's complement]}\end{aligned}$$

$$(-65)_{10} = (10111111)_2$$

$$\begin{aligned}(88)_{10} &= (01011000)_2 \\ &= (10100111)_2 \text{ [1's complement]} \\ &\quad +1 \\ \hline &= 10101000 \text{ [2's complement]}\end{aligned}$$

$$(-88)_{10} = (10101000)_2$$

Subject:

Date:

Now,

$$-88 = 10101000$$

$$-65 = 10111111$$

$$\underline{-153 = 101100111}$$

↑
carry bit

$$-65 = 10111111$$

$$-88 = 10101000$$

$-65 - 88$'s answer is 101100111

$$(1) \quad 34 + 55$$

$$(34)_{10} = (00100010)_2$$

$$(55)_{10} = (00110111)_2$$

$$34 = 00100010$$

$$55 = 00110111$$

$$\underline{(89)_{10} = 01101001}$$

$$\therefore 34 + 55 = 01011001$$

Subject: chapter 9

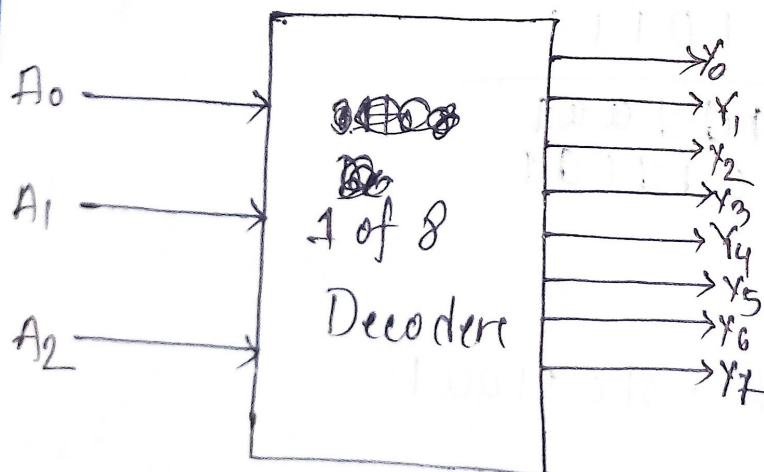
Date:

Q Design a 1 of 8 Decoder using logic gates.

Sol: ~~Design~~ The combinational circuit that change the binary information into 2^N output lines is known as decoders.

The 1 of 8 or ~~3 to 8~~ 3 to 8 line decoder is also known as Binary to octal decoder. There is a total eight outputs $Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7$ and three ~~and~~ inputs A_0, A_1 & A_2 . This circuit has an enable input E , ~~which is not~~

Block Diagram :-



Subject:.....

Date:.....

Truth table:-

Enable E	Inputs			Outputs							
	A ₂	A ₁	A ₀	Y ₇	Y ₆	Y ₅	Y ₄	Y ₃	Y ₂	Y ₁	Y ₀
0	X	X	X	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0

Logical expression:-

$$Y_0 = \bar{A}_0 \bar{A}_1 \bar{A}_2$$

$$Y_7 = A_0 A_1 A_2$$

$$Y_1 = \bar{A}_0 \bar{A}_1 A_2$$

$$Y_2 = \bar{A}_0 A_1 \bar{A}_2$$

$$Y_3 = \bar{A}_0 A_1 A_2$$

$$Y_4 = A_0 \bar{A}_1 \bar{A}_2$$

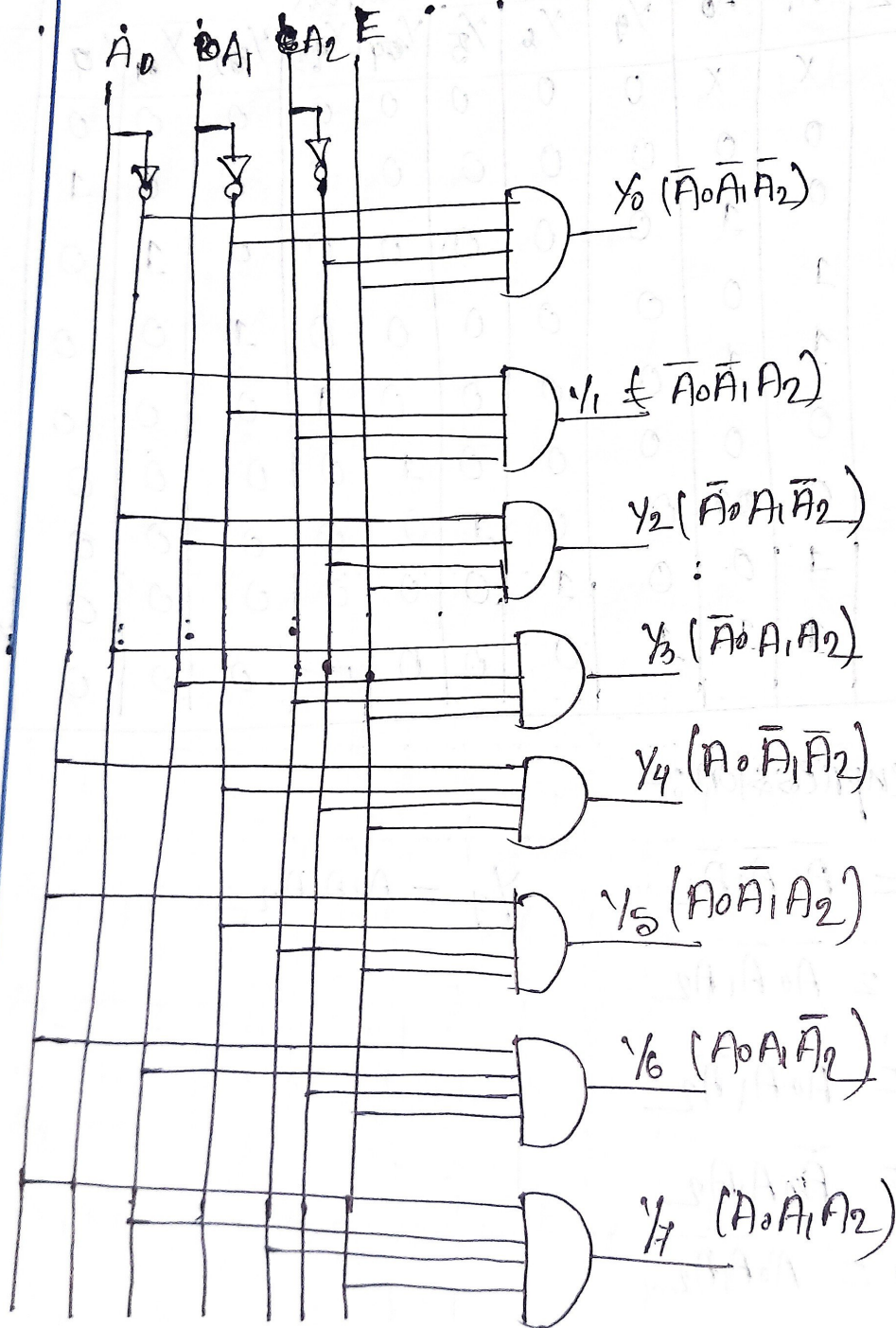
$$Y_5 = A_0 \bar{A}_1 A_2$$

$$Y_6 = A_0 A_1 \bar{A}_2$$

Subject:

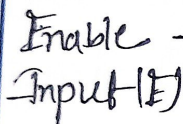
Date:

Logical circuit: -



Date :.....

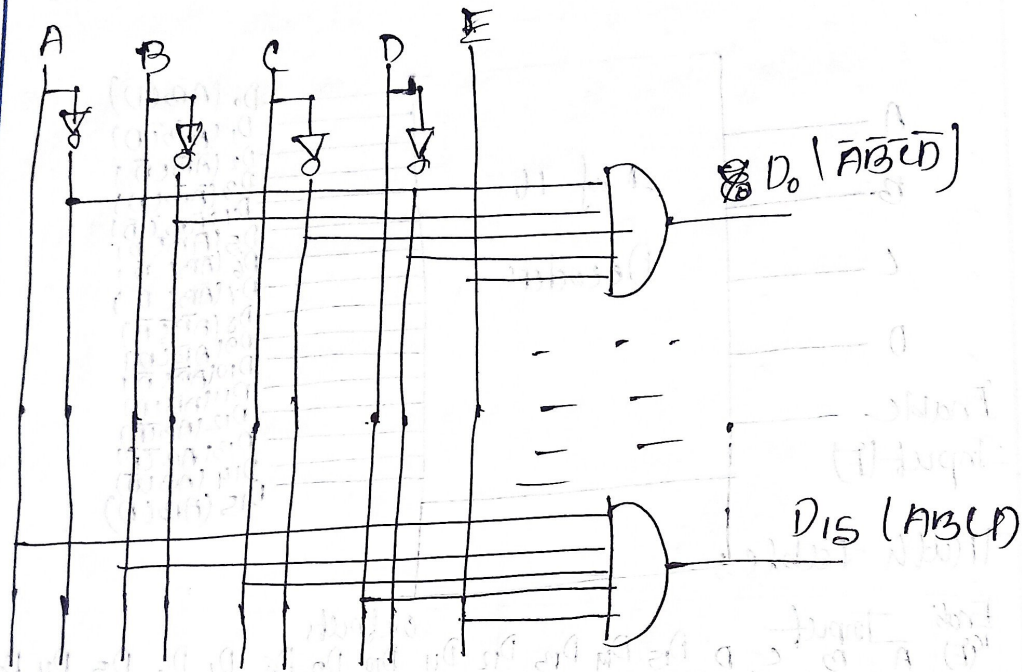
Enable -
Input (E)

[illegible][illegible]

Subject:

Date:

logical circuits



Explanation:— A 4 to 16 Decoder with enable input is a combinational circuit that takes a 4 bit input and activates one of 16 output lines based on the input combination, only if the enable input is active.

Input: It takes 4 bit input (A, B, C, D). This determines which of the 16 output lines will be active.

Subject:

Date:

Enable input: The decoder has also an ~~enable~~ enable input. When the enable input is high, the decoder functions normally. When it is low, all output lines are deactive.

Decoding:-

The 4 bit lines represent a binary number from 0000 to 1111 (0 to 15). Each combination of input lines selects a unique output line.

1. When input is 0000 output 0 is active,

2. When input is 0001, output 1 is active,

and so on

When input 1111, output 15 is active.

4. **Output:** The decoder has 16 output lines, labeled from 0 to 15. The selected output line is ~~active~~ activated when its input combination is present and the enable input is active.

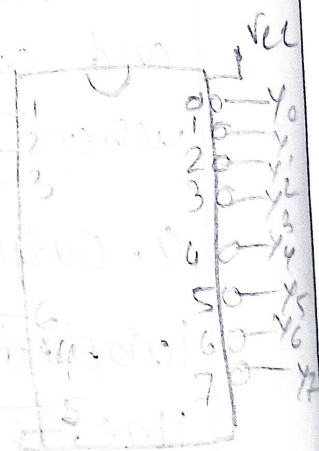
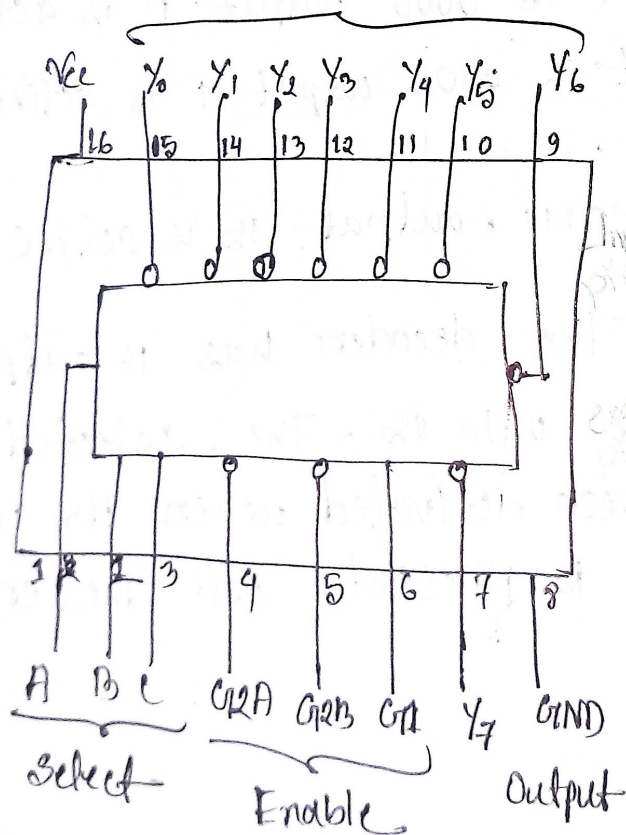
Q Draw the logic Diagram of a 74LS138 decoder IC.
 Also explain its working principle.

Subject:

Date:

Sol: The IC 74LS138 is a 3 to 8 line decoder integrated circuit. The main function of this IC is to decode or otherwise demultiplex the applications. The decoder 74LS138 IC uses advanced technology like silicon (Si) gate TTL technology.

Pin configuration:



Truth	Enab
0	1
1	1
1	1
1	1
1	1
1	1
1	1
1	1
1	1
1	1

we

Subject:

Date:

Truth table:

Inputs		Outputs										
Enable		select										
G_{11}	$G_{12}(\text{Not})$	C	B	A	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
X	1	X	X	X	1	1	1	1	1	1	1	1
0	X	X	X	X	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1	1	1
1	0	1	0	0	1	1	1	1	0	1	1	1
1	0	1	0	1	1	1	1	1	1	0	1	1
1	0	1	1	0	1	1	1	1	1	1	0	1
1	0	1	1	1	1	1	1	1	1	1	1	0

where, $G_{12} = G_{12A} + G_{12B}$

X = don't care

0 = Low

1 = High

$$Y_0 = \overline{CBA}$$

$$Y_1 = \overline{CBA}$$

$$Y_2 = \overline{CBA}$$

$$Y_3 = \overline{CBA}$$

$$Y_4 = \overline{CBA}$$

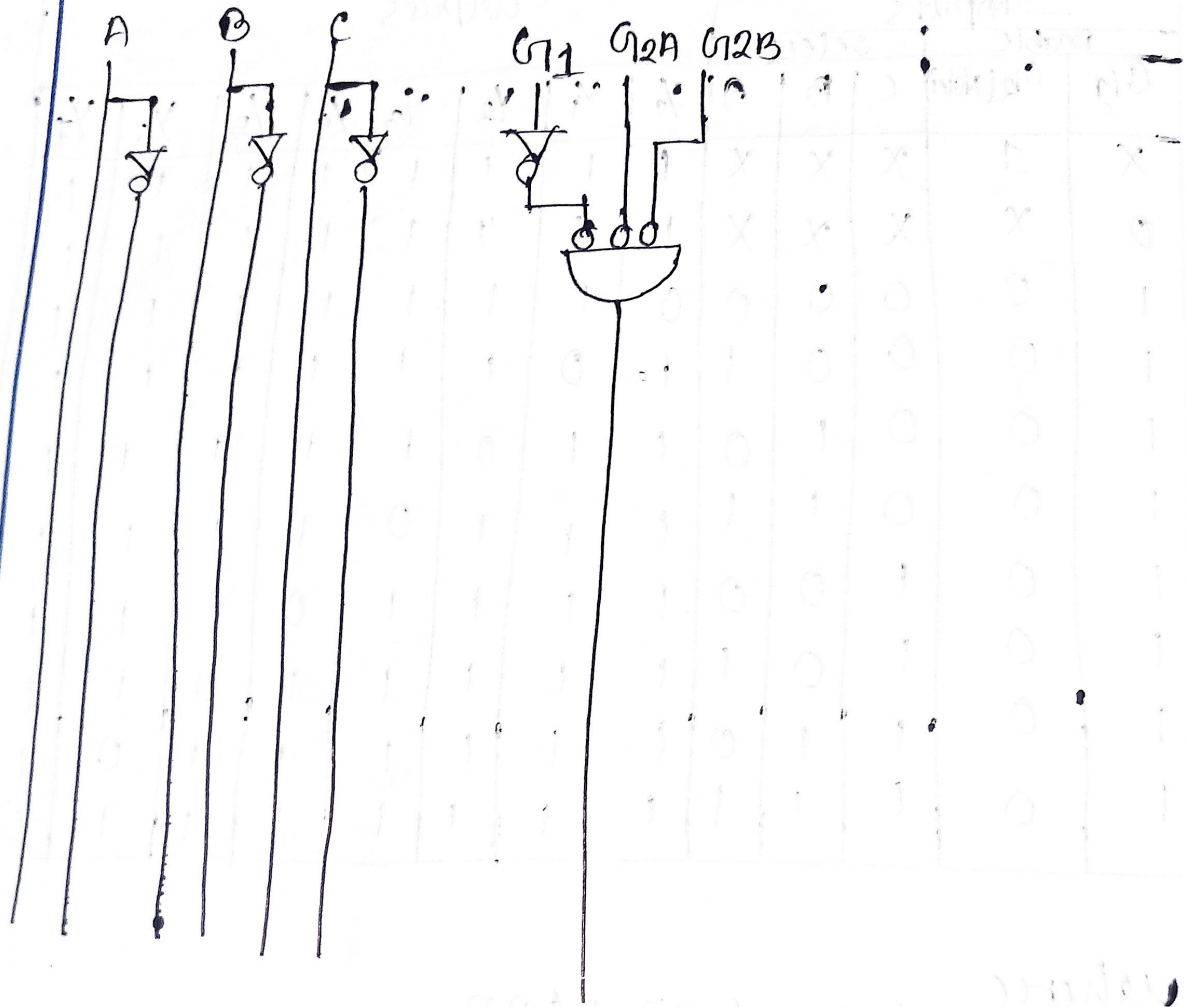
$$Y_5 = \overline{CBA}$$

$$Y_6 = \overline{CBA}$$

$$Y_7 = \overline{CBA}$$

Subject:

Date:

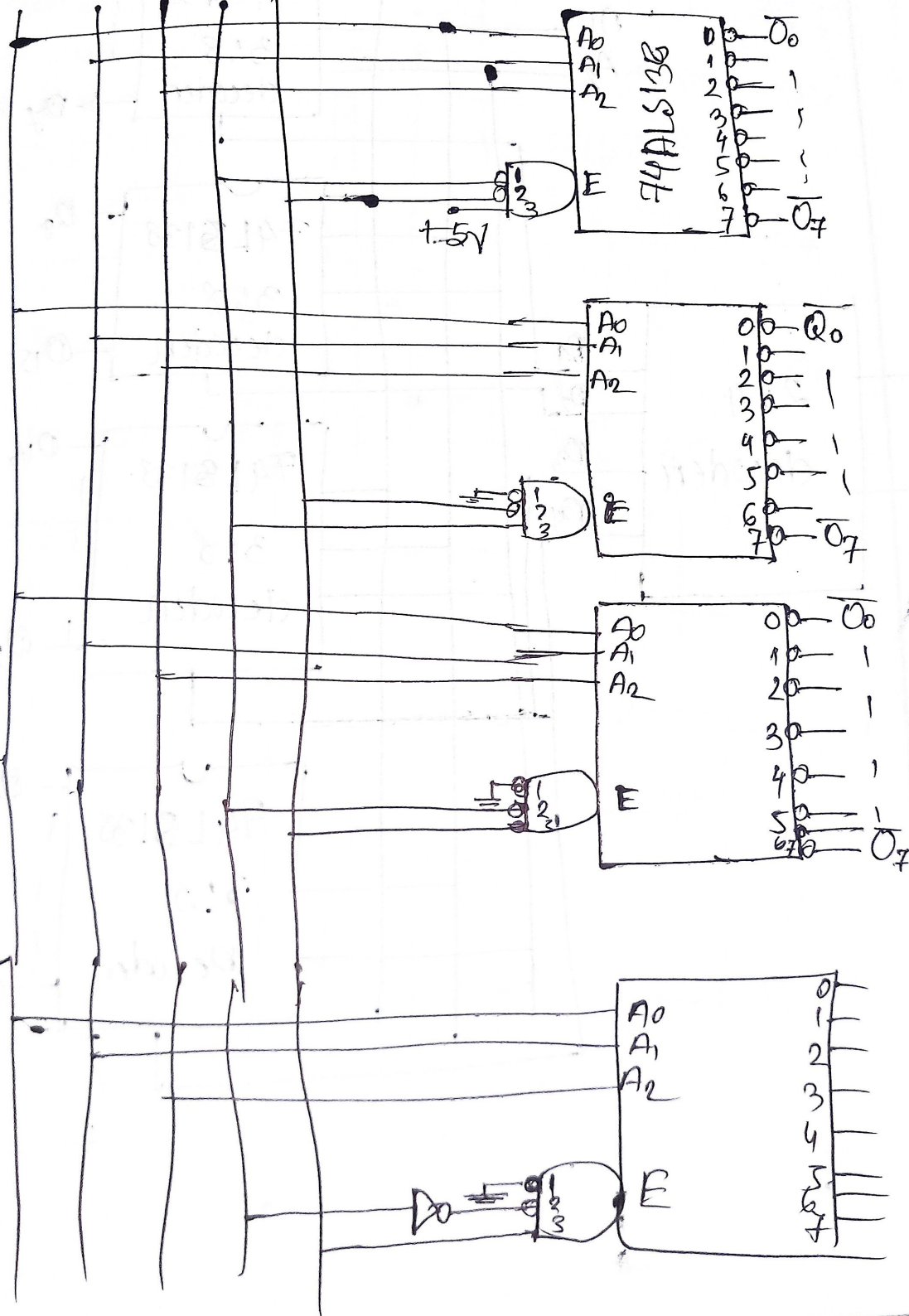


Ex 74LS138 IC to design a
1 of 32 decoder.

Subject:

Date:

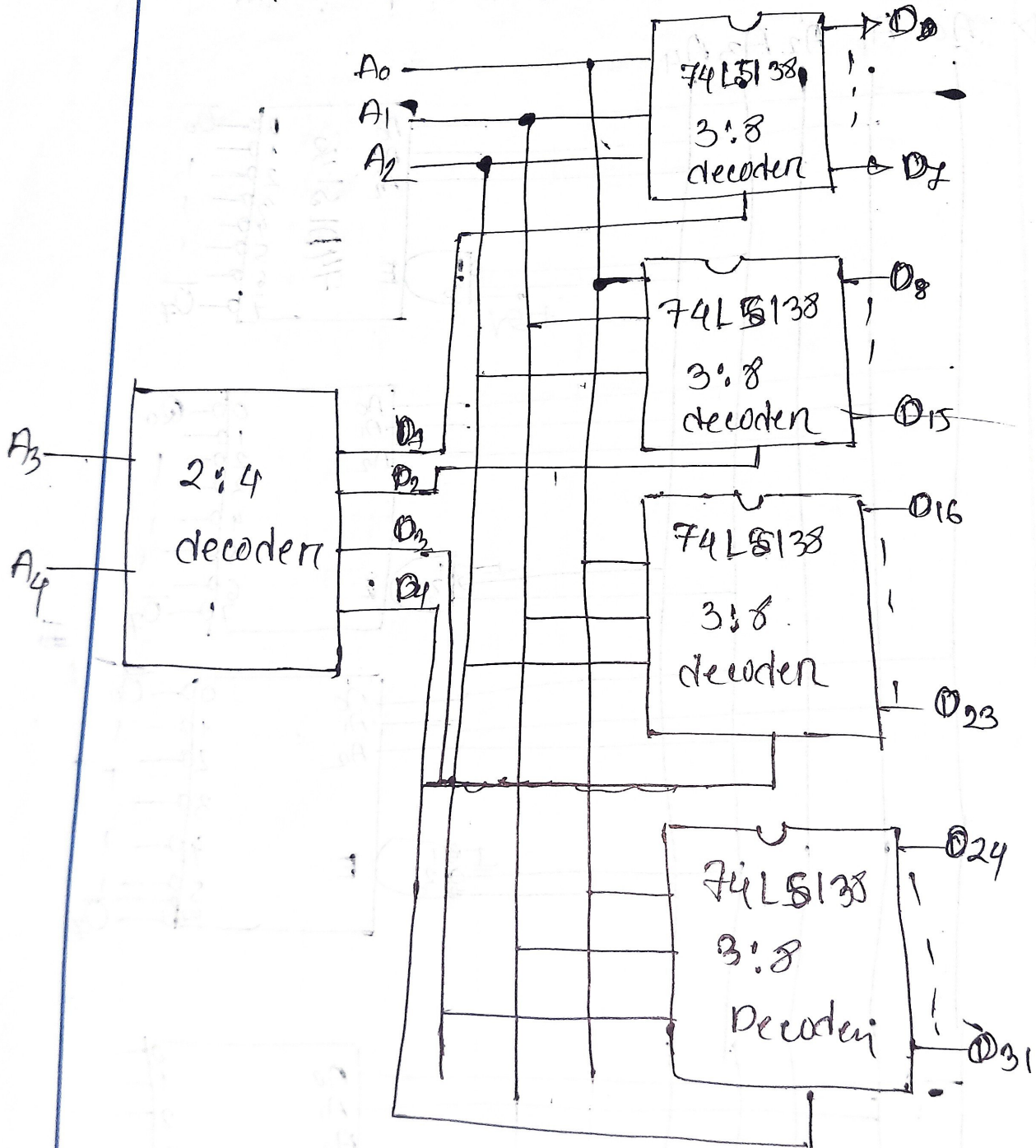
⇒ A₀ A₁ A₂ A₃ A₄



1 to 32

Subject:

Date:



1 to 24

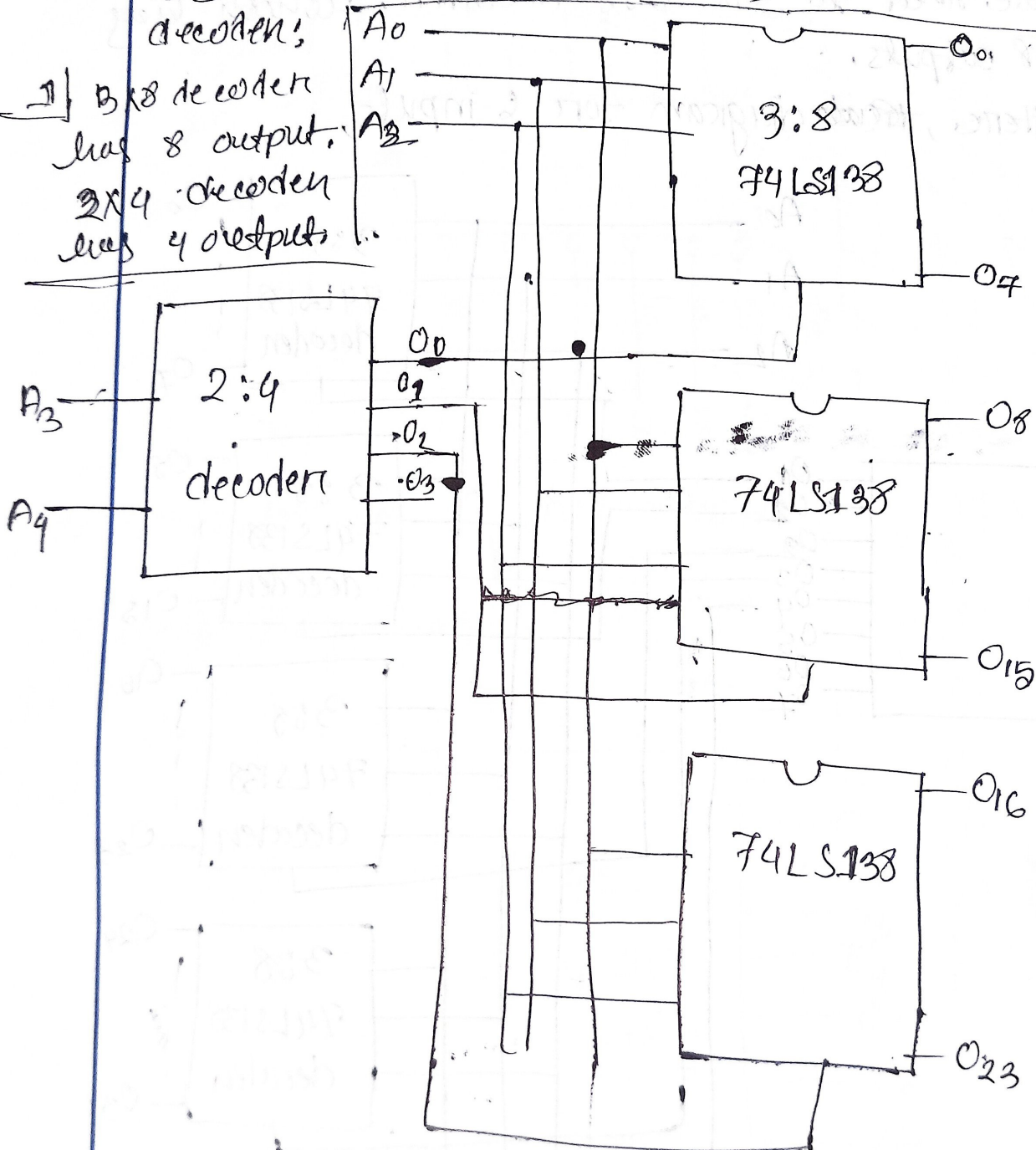
Subject:

Date:

8249

to create 1 to 24 decoder using 3 to 8 decoder
we need ~~to~~ one (2 to 4) and two (3 to 8)
decoder;

1] 3 to 8 decoder
has 8 output.
2] 2 to 4 decoder
has 4 outputs.



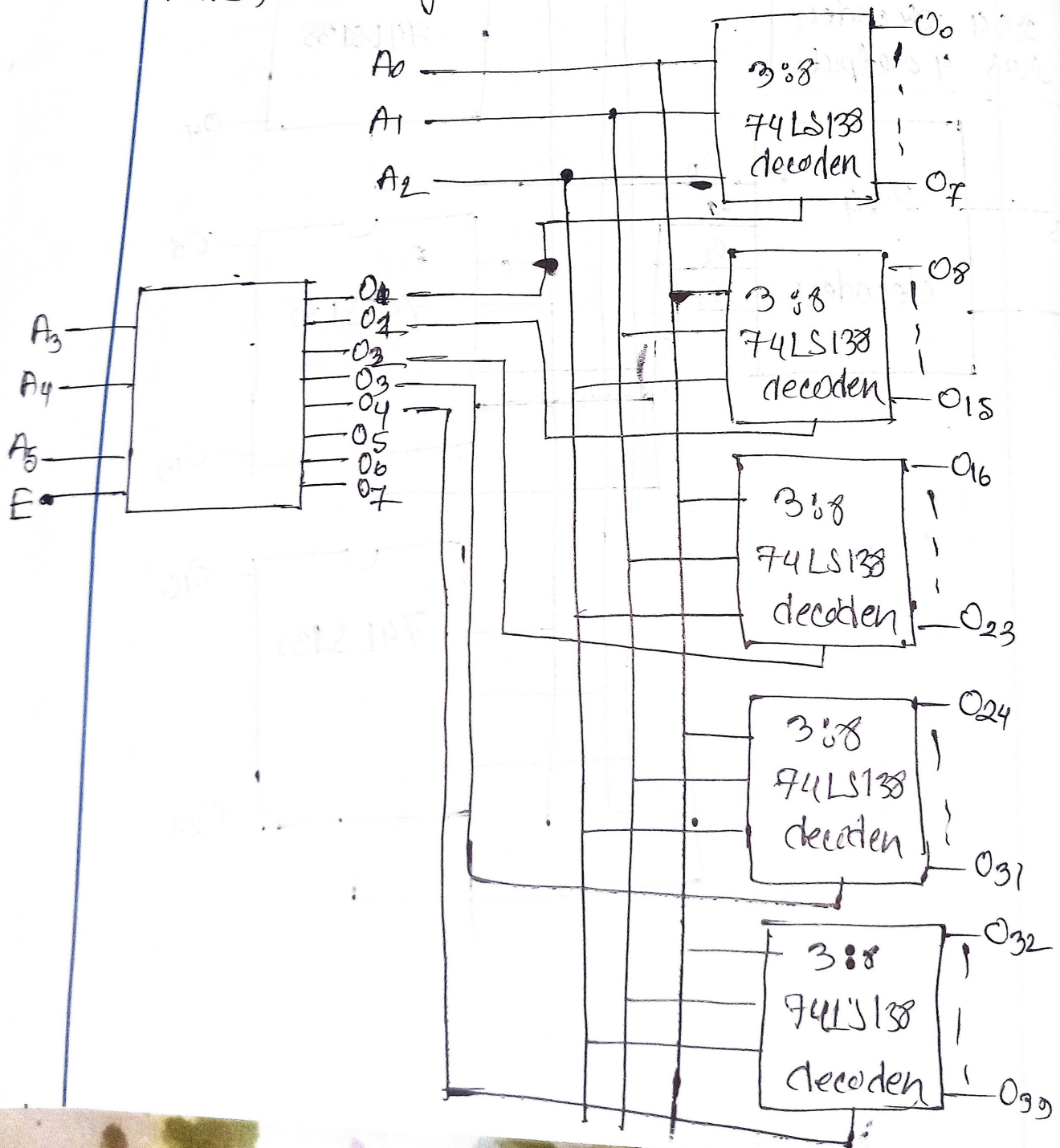
* 1 of 40 decoder using 74LS138 IC.

Subject:

Date:

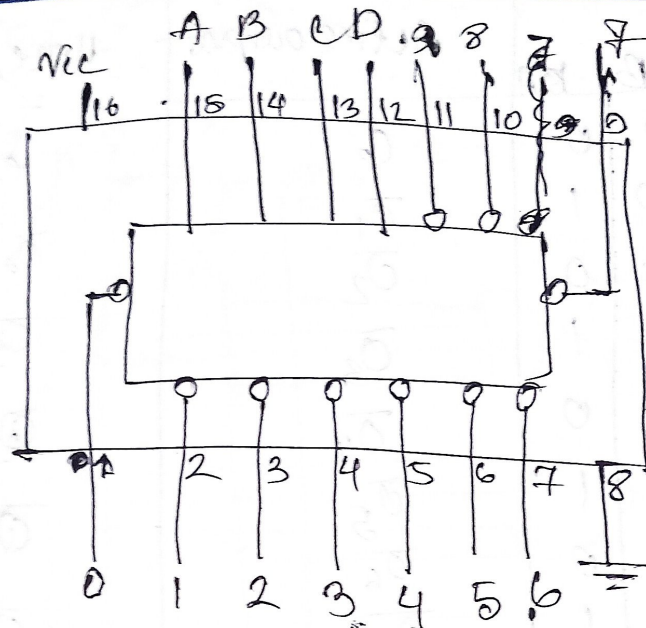
⇒ To create 1 of 40 decoder using 74LS138 IC we need to six 3:8 decoder, which have 8 outputs.

Here, ~~read~~ diagram for 6 inputs,

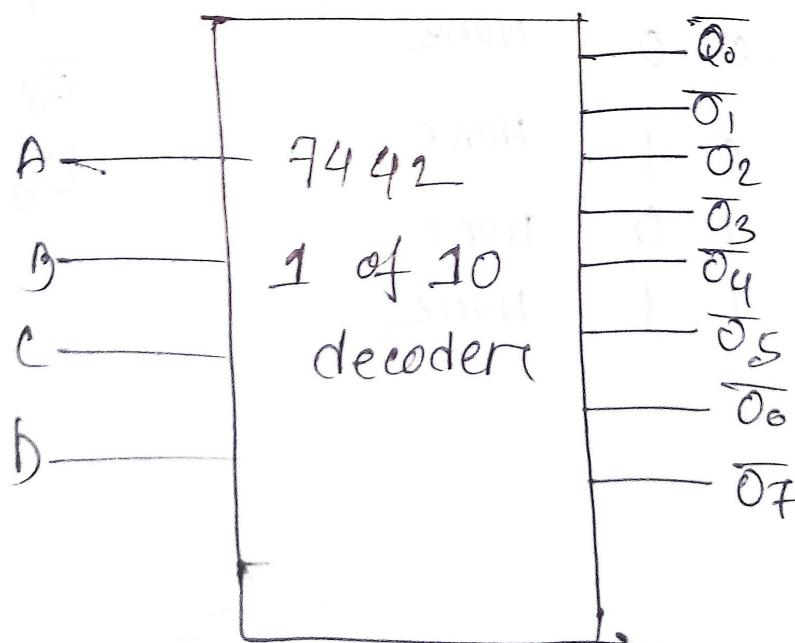


7442 logic connecting
diagram:-

Subject: _____
Date: _____



7442 BCD to decimal decoder:
block diagram:-



truth tables

Subject:

Date:

Inputs				Active output
A	B	C	D	
0	0	0	0	$\overline{O_0}$
0	0	0	1	$\overline{O_1}$
0	0	1	0	$\overline{O_2}$
0	0	1	1	$\overline{O_3}$
0	1	0	0	$\overline{O_4}$
0	1	0	1	$\overline{O_5}$
0	1	1	0	$\overline{O_6}$
0	1	1	1	$\overline{O_7}$
1	0	0	0	$\overline{O_8}$
1	0	0	1	$\overline{O_9}$
1	0	1	0	None
1	0	1	1	None
1	1	0	0	None
1	1	0	1	None
1	1	1	0	None
1	1	1	1	None

Hence, logical expression,

$$\overline{O_0} = \overline{A} \overline{B} \overline{C} \overline{D}$$

$$\overline{O_1} = \overline{A} \overline{B} \overline{C} D$$

$$\overline{O_2} = \overline{A} \overline{B} C \overline{D}$$

$$\overline{O_3} = \overline{A} \overline{B} C D$$

$$\overline{O_4} = \overline{A} B \overline{C} \overline{D}$$

$$\overline{O_5} = \overline{A} B \overline{C} D$$

$$\overline{O_6} = \overline{A} B C \overline{D}$$

$$\overline{O_7} = \overline{A} B C D$$

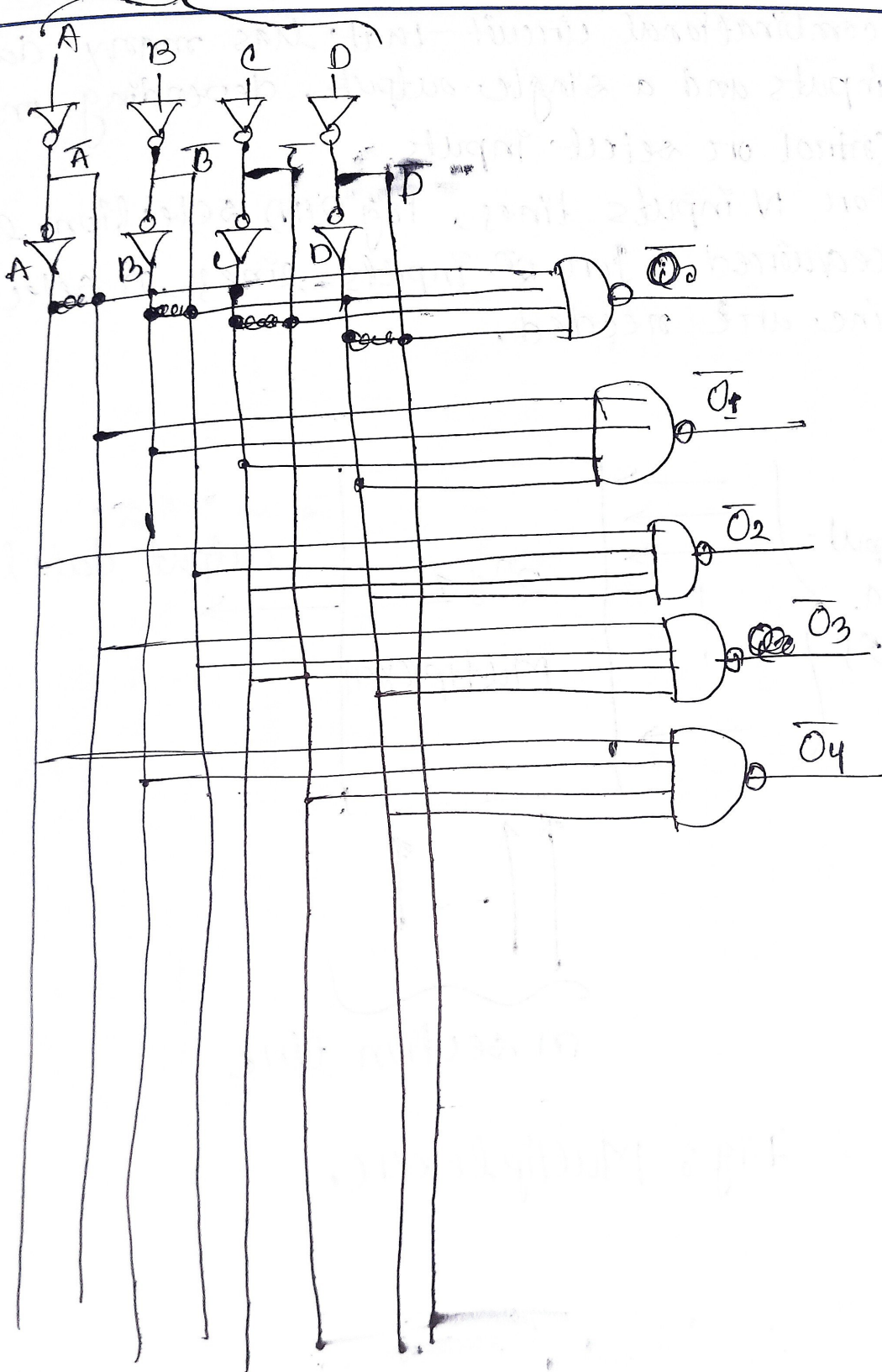
$$\overline{O_8} = \overline{A} \overline{B} \overline{C} D$$

$$\overline{O_9} = \overline{A} \overline{B} C D$$

BCD input
code

Subject:.....

Date:.....



what is multiplexers?

Subject:

Date:

⇒ A multiplexer is a

combinational circuit that has many data inputs and a single output, depending on

control or select inputs;

For N inputs lines, $\log_2(N)$ selection line required, for 2^n inputs lines, n selection line are needed.

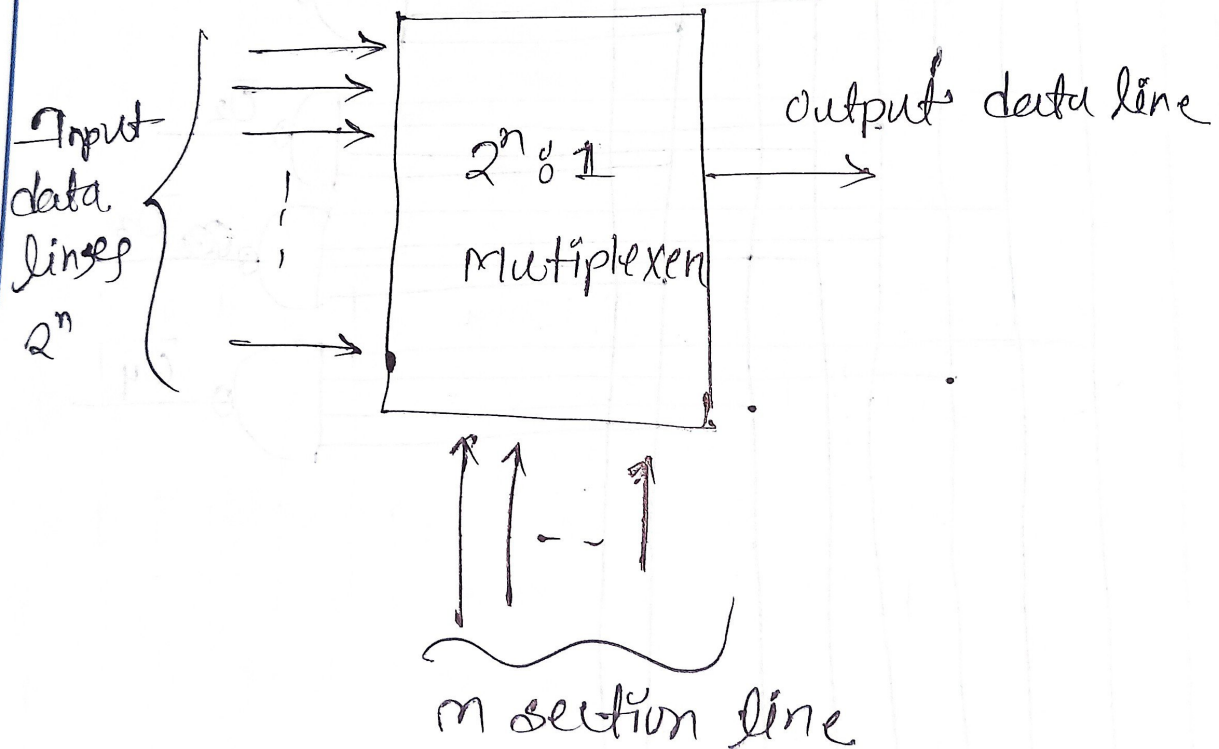


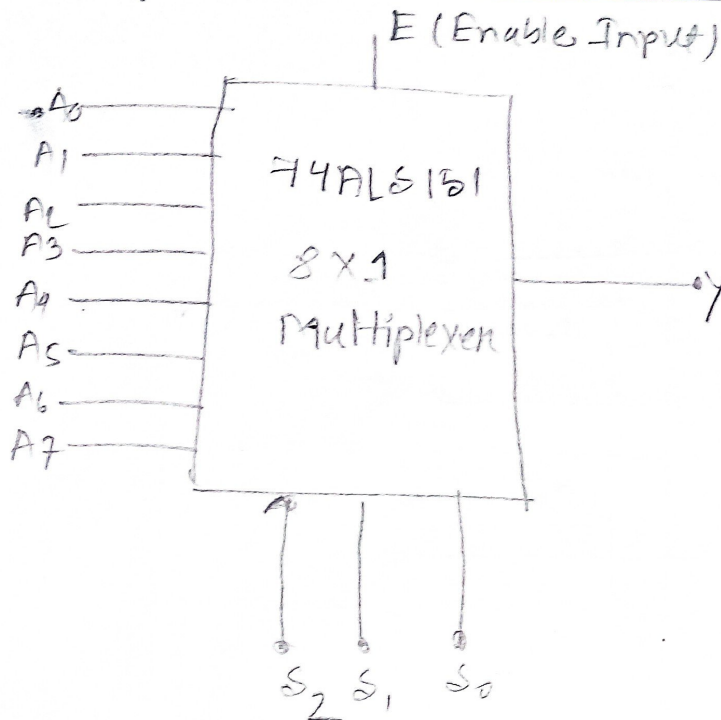
Fig: Multiplexer.

Q 1 8 to 1 Multiplexer using 74ALS151

logical diagram:

Subject: _____

Date: _____



truth table:-

Input			output
S_2	S_1	S_0	Y
0	0	0	A_0
0	0	1	A_1
0	1	0	A_2
0	1	1	A_3
1	0	0	A_4
1	0	1	A_5
1	1	0	A_6
1	1	1	A_7

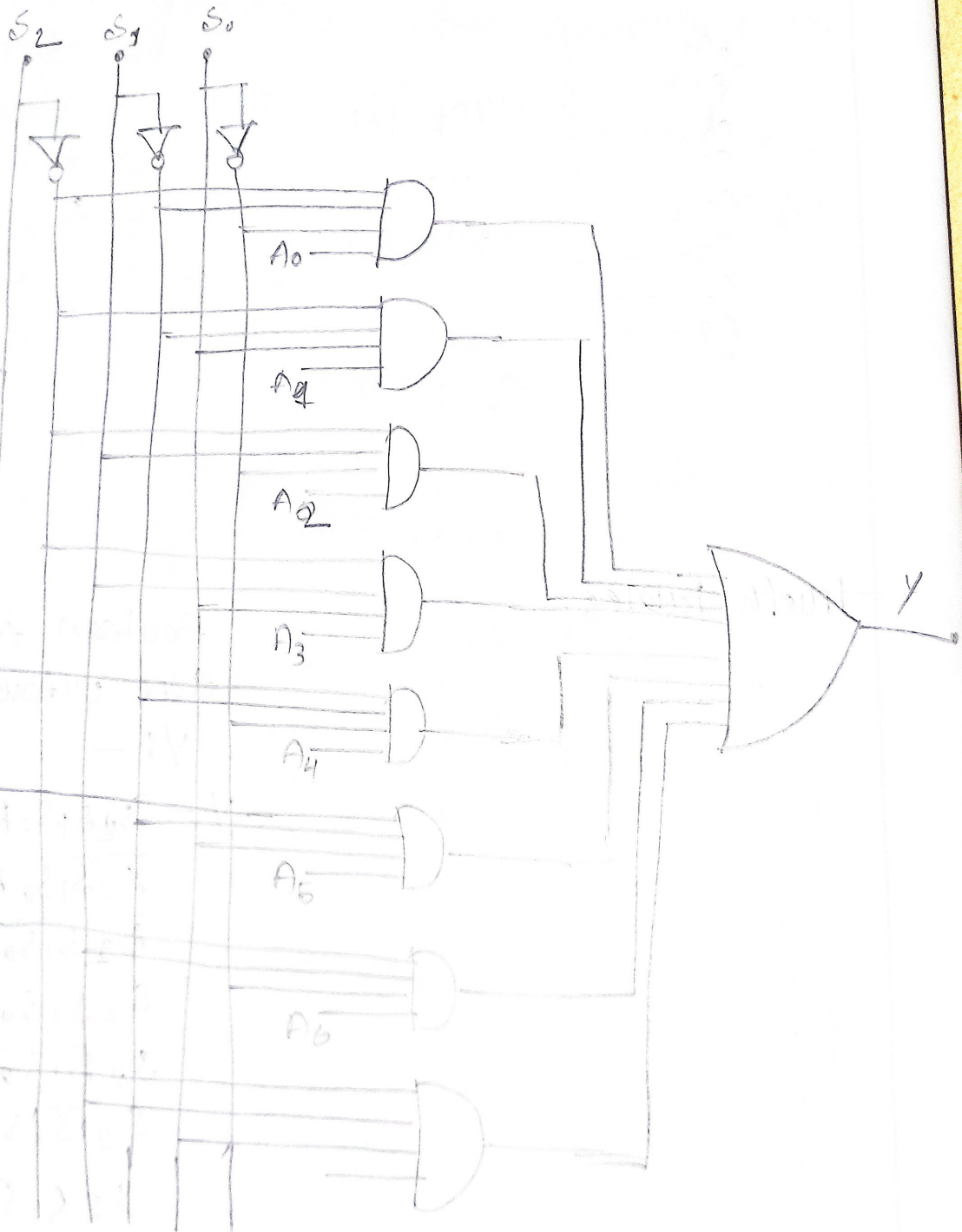
Boolean expression for output variable

Y :-

$$Y = \bar{S}_2 \bar{S}_1 \bar{S}_0 A_0 + \bar{S}_2 \bar{S}_1 S_0 A_1 + \bar{S}_2 S_1 \bar{S}_0 A_2 + \bar{S}_2 S_1 S_0 A_3 + S_2 \bar{S}_1 \bar{S}_0 A_4 + S_2 \bar{S}_1 S_0 A_5 + S_2 S_1 \bar{S}_0 A_6 + S_2 S_1 S_0 A_7$$

Subject:.....

Date:.....



Fig's logical diagram for 74ALS151 Multiplexer