B.Tech.

Fourth Semester Examination, May-2011

Digital Electronics (EE-204-F)

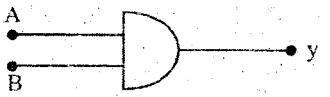
Note : Attempt *five* questions. *Q. No.* **1** *is compulsory* and one question from each of four Sections.

Q. 1. (a) Realize the following using Nand Gate :

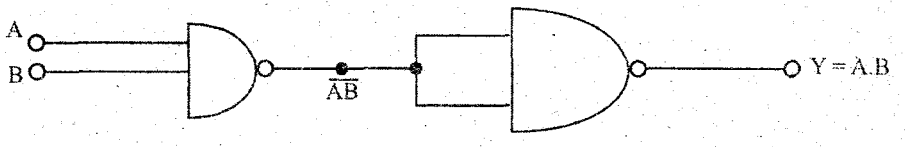
(i) AND gate

(ii) EX-OR gate

Ans. (i) AND Gate : **Desired Gate :**



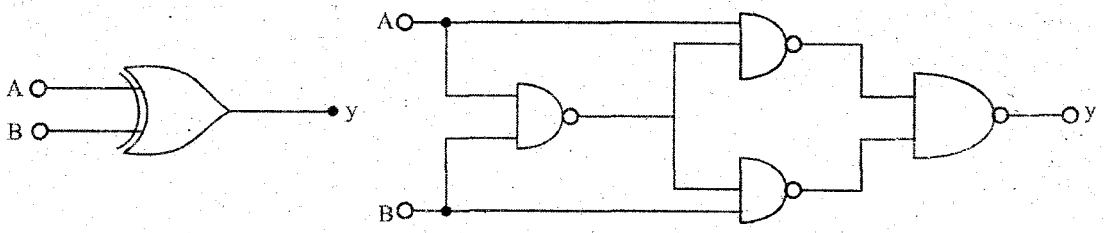
NAND Construction



An AND gate is made by following a NAND gate with a NOT gate as shown above. This gives a NOT NAND, i.e., AND.

(ii) EX-OR Gate :

Desired Gate **NAND Construction**



An EX-OR gate is constructed by using 4 NAND gates such that if both inputs are high, the inputs to the final NAND gate will also be high and the output will be low. This effectively represents the formula

"(A NAND (A NAND B)) NAND (B NAND (A NAND B))"

Q. 1. (b) Simplify :

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

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

Ans. (i) $F = \overline{A}\,(\overline{B} + \overline{C}) + BC + A\overline{C}$

$\Rightarrow \quad \overline{A}\overline{B} + \overline{A}\overline{C} + BC + A\overline{C}$

$\Rightarrow \quad \overline{C}(\overline{A}+A)+\overline{A}\overline{B}+BC$

$\Rightarrow \quad \overline{A}\overline{B}+BC+\overline{C} \qquad [\because A+\overline{A}=1]$

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

$\Rightarrow \quad \overline{A.B+A.C}+\overline{A}B+\overline{C}\,\overline{(A+B)}$

$\Rightarrow \quad \overline{A.B}.\overline{A.C}+\overline{A}B+\overline{A}.\overline{B}.\overline{C}$

$[\because \overline{A+B}=\overline{A}.\overline{B}]$ Dc- Morgan Theorem

$\Rightarrow \quad (\overline{A}+\overline{B})(\overline{A}+\overline{C})+\overline{A}B+\overline{A}\overline{B}\overline{C}$

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

$\Rightarrow \quad \overline{A}+\overline{A}\overline{C}+\overline{B}\overline{C}+\overline{A}\overline{B}\overline{C} \qquad [\because A+A=A]$

$[\because A+1=1]$

$\Rightarrow \quad \overline{A}+[(\overline{A}+\overline{B}+\overline{A}\overline{B})\overline{C}]$

$\Rightarrow \quad \overline{A}+(\overline{A}+\overline{B})\overline{C}$

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

$\Rightarrow \quad \overline{A}+\overline{B}\overline{C} \qquad [\because A+A=A]$

Q. 1. (c) Give truth table for :.

(i) S-R Flip-Flop

(ii) J-K Flip-Flop

Ans. (i) S-R Flip-Flop :

S	R	Q	Q'	
1	0	1	0	(after $S=1, R=0$)
0	0	1	0	
0	1	0	1	(after $S=0, R=1$)
0	0	0	1	
1	1	0	0	

Or

Inputs		Outputs
S_n	R_n	Q_{n+1}
0	0	Q_n
1	0	1
0	1	0
1	1	?

(ii) J-K Flip-Flop

Inputs		Output
J_n	K_n	Q_{n+1}
0	0	Q_n
1	0	1
0	1	0
1	1	$\overline{Q_n}$

Q. 1. (d) Differentiate between :

(i) Latch and Flip-Flop

(ii) Ripple counter and Synchronous counter

Ans.

	Latches	Flip-Flop
(i)	Latches are asynchronous, which means that the output changes very soon after the input changes.	A flip-flop is a synchronous version of the latch i.e., a latch with a clock signal, which means that the outputs of all the sequential circuits change simultaneously to the rhythm of a global clock signal.
(ii)	A flip-flop is an edge-triggered device.	Whereas latch is a level sensitive device.
(iii)	Latch is sensitive to glitches on enable fin.	Whereas flip-flop is immune to glitches.
(iv)	Latch does not have a clock signal.	Have a clock signal.
(v)	Latches are faster than flip-flops.	Slower than latches.

In electronics both are a kind of bistable multi-vibrator.

(ii) Ripple Counter and Synchronous Counter :

Ripple Counter : In a ripple counter, the flip-flop output transition serves as a source for triggering other flip-flops. In other words, the C input of some or all flip-flops are triggered not by the common clock pulses, but rather by the transition that occurs in the other flip-flop outputs. So in ripple or asynchronous counter all the flip-flop are not clocked simultaneously.

Synchronous Counter : In a synchronous counter all the flip-flops are clocked simultaneously and the C inputs of all flip-flops receive the common clock. The ring and the twisted-ring counters are example of synchronous counters.

The ripple counters have the advantage of simplicity (only flip-flops are required) but their speed is low because of ripple action.

The speed of operation of **synchronous** counters improves significantly if all the flip-flops are clocked simultaneously. The resulting circuit is known as a synchronous counter.

Section—A

Q. 2. (a) (i) Multiply 4.75_2 by 3.625_2.

(ii) Divide 50_2 by 5_2.

(iii) Convert 2004_{10} to octal

(iv) Convert 7325_8 to binary

(v) Add $+112_{10}$ and $+65_{10}$ using 2's compliment arithmetic.

Ans. (i) Multiply 4.75_2 by 3.625_2 :

$$
\begin{array}{r}
4.75 = 100.110 \\
3.625 = \times 11.101 \\
\hline
100110 \\
000000 \\
\hline
100110 \\
100110 \\
\hline
10001.001110 = (10001.00111)_2
\end{array}
$$

$$\Rightarrow \quad (17.21875)_{10}$$

$$\Rightarrow \quad \left(17\frac{7}{32}\right)_{10} \quad \textbf{Ans.}$$

(ii) Divide 50_2 by 5_2 :

i.e., $(110010)_2$ by $(101)_2$

$$
\begin{array}{rl}
 & 101 \quad \leftarrow \text{Quotient} \\
\text{Divisor} \rightarrow 101 \overline{)} & 110010 \quad \leftarrow \text{Dividend} \\
 & 101 \\
\hline
 & 00101 \\
 & 101 \\
\hline
 & 0
\end{array}
$$

Ans. $\boxed{\textbf{101}}$ i.e., $(101)_2$

(iii) Convert 2004_{10} to Octal :

i.e., $(2004)_{10}$ to $(\quad)_8$

Quotient

$$\frac{2004}{8} \rightarrow 250$$

$$\frac{250}{8} \rightarrow 31$$

$$\frac{31}{8} \rightarrow 3$$

$$\frac{3}{8} \rightarrow 0$$

Remainder

4 → 2 → 7 → 3

3 7 2 4

Thus, $(2004)_{10} = (3724)_8$

(iv) Convert 7325_8 to Binary :

i.e., $(7325)_8$ to ($)_2$

Octal numbers can be converted into equivalent binary numbers by replacing each octal digit by its 3-bit equivalent binary.

i.e., $7 = 111$

$3 = 011$

$2 = 010$

$5 = 101$

Thus, $(7325)_8 = (111011010101)_2$

(v) Add + 112_{10} and + 65_{10} using 2's Compliment Arithmetic :

2's complement representation of

$(+112)_{10} = 01110000$

2's complement representation of

$(+65)_{10} = 01000001$

$(+112) + (+65)$

(+ 112)	0 1 1 1 0 0 0 0
+ (+ 65)	0 1 0 0 0 0 0 1
+ 177	①0 1 1 0 0 0 1

The final carry bit acts as a sign bit for the answer if it is 1 then the answer is +ve so then the answer is negative.

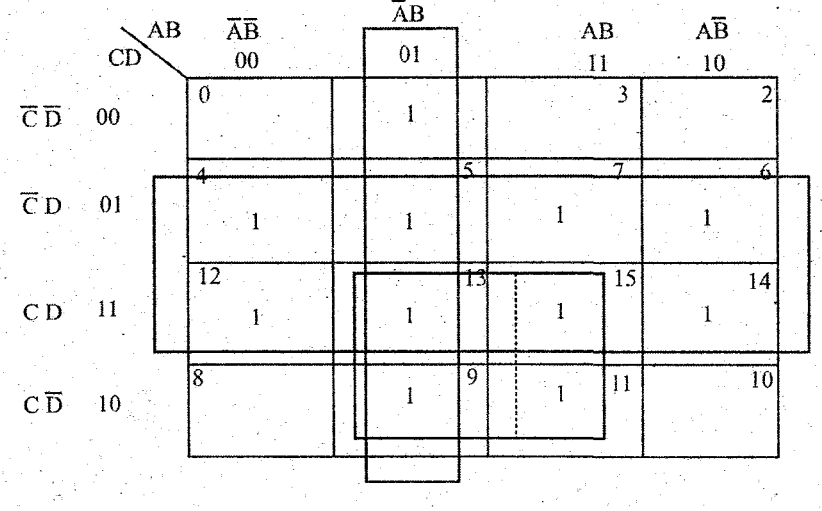
In our case final carry is 1 so the answer is positive i.e., + 177.

Q. 2. (b) Use K-map to simplify :

(i) $F = \bar{A}B\bar{C} + A\bar{C}D + BC + \bar{A}BD + A\bar{B}CD$

(ii) $F = (AC + A\bar{C}D)(AD + AC + BC)$

Ans. (i) K-map for $F = \bar{A}B\bar{C} + A\bar{C}D + BC + \bar{A}BD + A\bar{B}CD$



$\Rightarrow \quad \bar{A}B\bar{C}(\bar{D}+D)+A\bar{C}D(\bar{B}+B)+(\bar{A}+A)(\bar{D}+D)BC+\bar{A}\bar{B}D(\bar{C}+C)+A\bar{B}CD$

$\Rightarrow \quad \bar{A}B\bar{C}\bar{D}+\bar{A}B\bar{C}D+A\bar{C}D\bar{B}+A\bar{C}DB+\bar{A}\bar{D}BC+\bar{A}DBC+A\bar{D}BC$

$+ADBC+\bar{A}\bar{B}D\bar{C}+\bar{A}\bar{B}DC+A\bar{B}CD \qquad [\because A+\bar{A}=1]$

The simplified expression by using K-map is $Y = D+\bar{A}B+BC$

(ii) $F = (AC+A\bar{C}D)(AD+AC+BC)$

$\Rightarrow \quad A.A.CD+A.A.C.C+ABC.C+A.A\bar{C}D.D+A\bar{C}CD+A\bar{C}DBC$

$\Rightarrow \quad ACD+AC+ABC+A\bar{C}D \qquad [\because A.\bar{A}=0 \ \& \ A.A=1]$

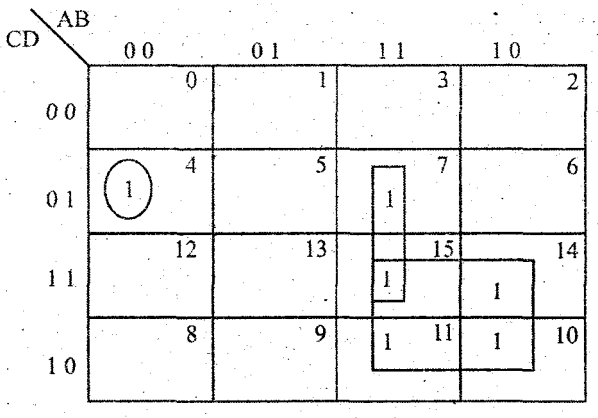
$\therefore \quad \Rightarrow \quad AC(D+1)+ABC+A\bar{C}D$

$\Rightarrow \quad AC+ABC+A\bar{C}D \qquad [\because A+1=1]$

$\Rightarrow \quad AC(\bar{B}+B)(\bar{D}+D)+ABC(\bar{D}+D)+A\bar{C}D(\bar{B}+B)$

$\Rightarrow \quad ABC\bar{D}+ABCD+AC\bar{B}\bar{D}+AC\bar{B}D+ACBD+ACBD+\bar{A}C\bar{D}\bar{B}+A\bar{C}DB$

$\Rightarrow \quad ABC\bar{D}+ABCD+A\bar{B}C\bar{D}+A\bar{B}CD+\bar{A}\bar{B}C\bar{D}+AB\bar{C}D \qquad [\because \ \bar{A}+\bar{A}=\bar{A}]$



The simplified expression by using K-map is

$$Y = AC+ABD+\bar{A}\bar{B}\bar{C}D$$

Q. 3. (a) Write short notes on :

(i) Error detecting and correcting codes

(ii) Excess-3 code and gray code

Ans. (i) Error Detecting and Correcting Codes :

Error-Detecting : Error-detecting is the detection of errors caused by noise or other impairments during transmission from the transmitter to the receiver. When a message is transmitted, it has the potential to get scrambled by noise. This is certainly true of voice messages, and is also true of the digital messages that are sent to and from computers. Now even sound and video are being transmitted in this manner. By a digital message, we mean a sequence of 0's & 1's which encodes a given message. So we will seek to do is to add more data to a given binary message : adding such data is called an error-detecting code.

A common-type of error-detecting code is called a parity check.

Error-Correcting Codes : We will also try to add data to the original message so that we can detect if errors were made in transmission, and also to figure or what the original message was from the possibly correct message that was received. This type of code is an **error-correcting code.**

In general, a code is said to be error-correcting code if the correct code word can be deduced from the erroneous word. Hamming code is an error-correcting code.

(ii) Excess-3 Code and Gray Code :

Excess-3 Code : This is another form of BCD codes in which each decimal digit is coded into a 4-bit binary code. The code for each decimal digit is obtained by adding decimal 3 to the natural BCD code. **e.g.,** decimal 2 is coded as 0010 to 011 = 0101 in excess-3 code. It is not a weighted code. This code is a self-complementing code, which means 1's complement of the coded number yields 9's complement. For example, excess-3 code of decimal 2 is 0101, its 1's complement is 1010 which is excess-3 code for decimal 7, which is 9's complement of 2.

Gray Code : It is very useful code in which a decimal number is represented in binary form in such a way so that each gray-code number differs from the preceding and the succeeding number by a single bit. For example, the Gray code for decimal number 5 is 0111 and for 6 is 0101. These two codes differ by only one bit position (third from the left). This code is extensively for shaft encoders because of this property. It is not a weighted code.

Q. 3. (b) Give simplified large equation of Table by Quine-McClusky Method.

Table

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

Ans. The logic function of table is written as

$$Y(A, B, C) = \Sigma m(2, 6, 7)$$

Arrange all the minterms of the function in binary form in a table according to the number of ones; one 1 & two 1's & so on.

i.e.,

→ Group	Minterm	Variables A	B	C	Check
1	2	0	1	0	✓
2	6	1	1	0	✓
3	7	1	1	1	✓

→ Group	Minterm	Variables A	B	C	Check
1	2, 6	–	1	0	✓
2	6, 7	1	1	–	✓

Now we check group of four can be made

Group	Minterm	Variables A	B	C
1	2, 6, 6, 7	No		

All non-checked minterm groups are the prime implicants of the function. The function can be written as,

$$\boxed{Y(A,B,C)=AB+B\overline{C}}$$

Section—B

Q. 4. (a) Show how 1.6 to 1 multiplexes can be obtained using 4 to 1 multiplexes.

Ans. Since 1.6 inputs are there, the first four input are applied to the first 4 : 1 MUX. The second four inputs to the second 4 : 1 MUX & so on. Select inputs C & D are applied to S_1 & S_0. The outputs of these Muxes are connected as data inputs to the fifth 4 : 1 MUX & select inputs A and B are applied to S_1 & S_0 of that MUX.

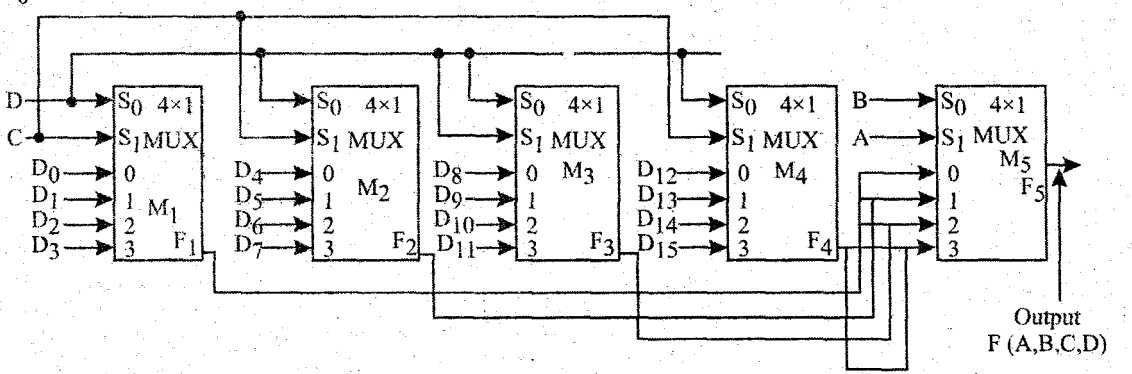
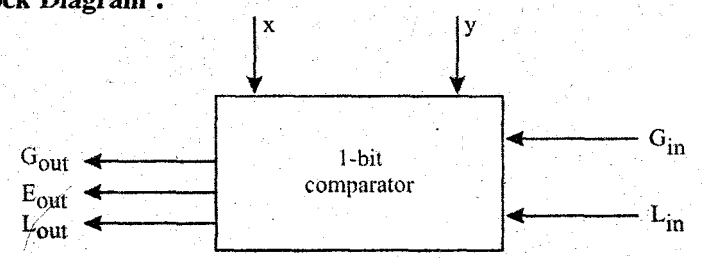


Fig. 16 : 1 MUX using 4 : 1 MUXS

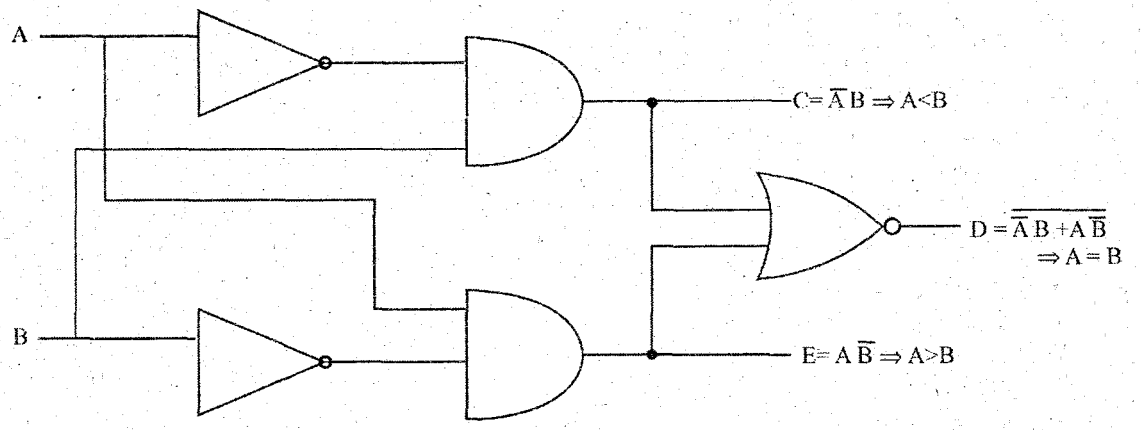
Q. 4. (b) Draw and explain :
(i) Block diagram (ii) Truth Table (iii) Circuit for 1-bit comparator
Ans. (i) Block Diagram :



(ii) Truth Table :

Inputs		Outputs		
B	A	A>B	A=B	A<B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

(iii) Circuit for 1-bit Comparator :



Q. 5. (a) Give truth table, Boolean equation and circuit diagram for :

(i) Full Adder

(ii) Subtractor.

Ans. (i) Full Adder :

Full adder has 3 inputs and two outputs.

Truth Table

Inputs			Outputs	
A_n	B_n	C_{n-1}	S_n	C_n
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

Boolean Equation :

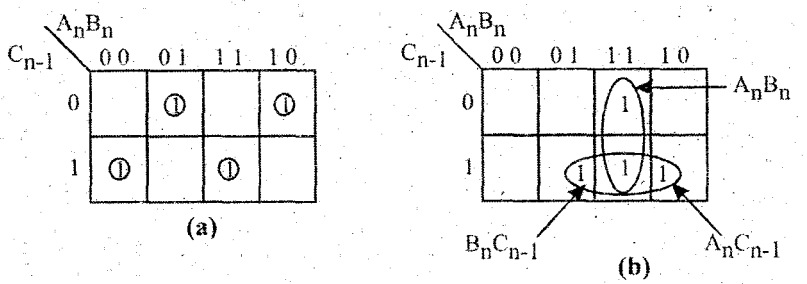
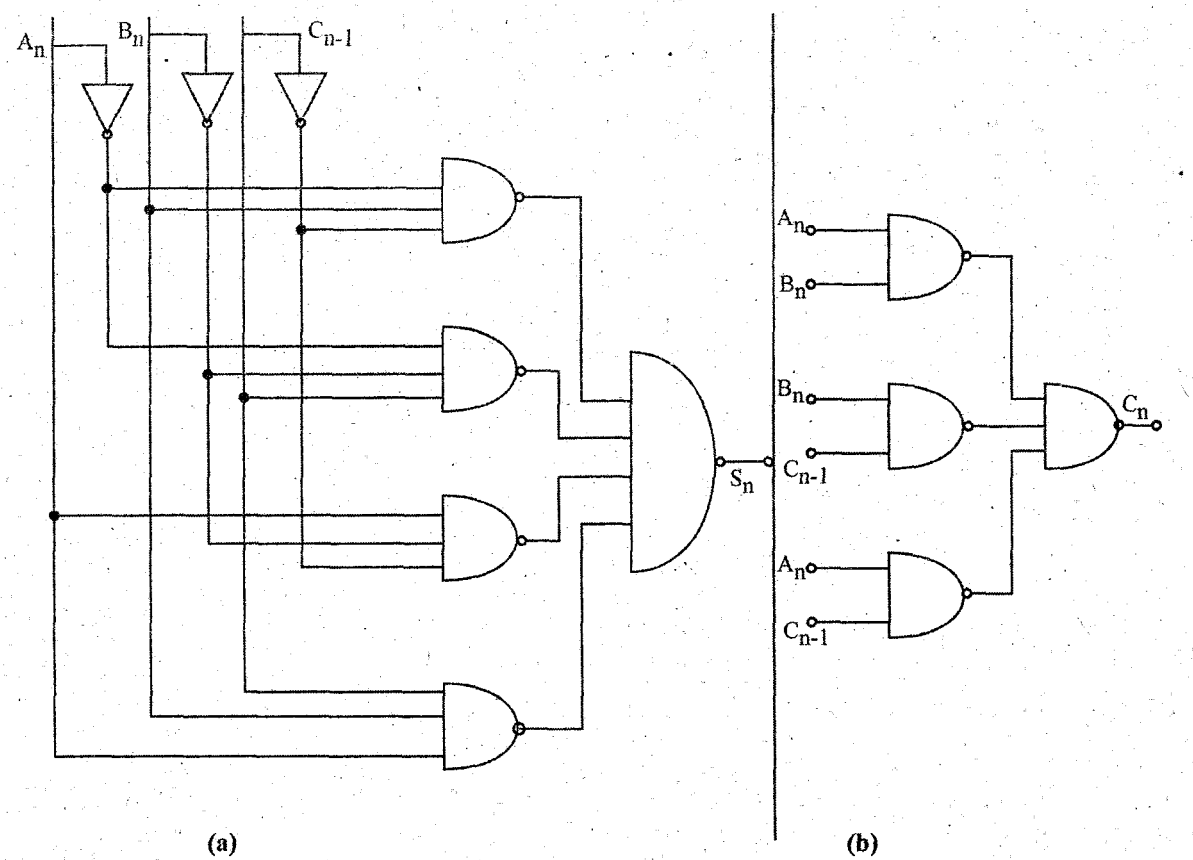


Fig. K-maps for S_n & C_n

$$\therefore \quad S_n = \text{Sum} = \overline{A}_n B_n \overline{C}_{n-1} + \overline{A_n}\ \overline{B_n} C_{n-1} + A_n \overline{B_n}\ \overline{C_{n-1}} + A_n B_n C_{n-1}$$

$$C_n = A_n B_n + B_n C_{n-1} + A_n C_{n-1}$$

Circuit Diagram :



NAND-NAND realization of (a) sum (S_n) & (b) carry (C_n)

(b) Subtractor : Just like a full-adder we require a full-subtractor circuit for performing multibit subtraction wherein a borrow from the previous bit position may also be there.

Truth Table

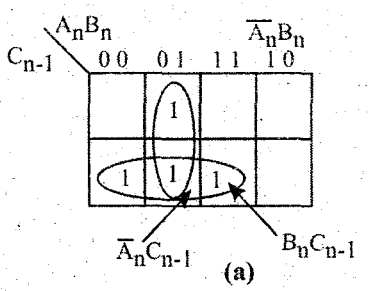
Borrow from the previous stage

Subtrahend

Minuend

Inputs			Outputs	
A_n	B_n	C_{n-1}	D_n	C_n
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

Equation :

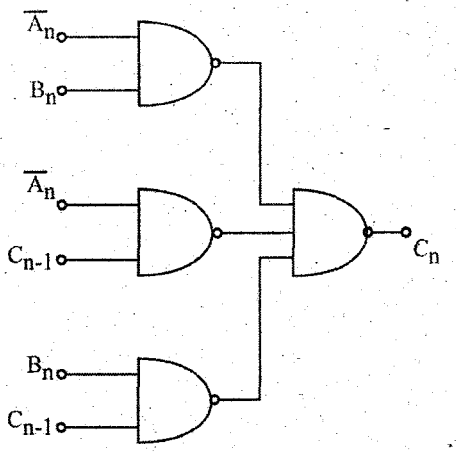


(a)

K-map for C_n

The K-map for the output D_n is exactly the same as the K-map for S_n of the full adder circuit & therefore its realization is same.

Circuit for Subtractor



Q. 5. (b) What is a demultiplexer. Explain with help of suitable block diagram and logic circuit of 1 to 16 demuliplexer.

Ans. The demultiplexer performs the reverse operation of a MUX. It accepts a single input & distribute it over several outputs.

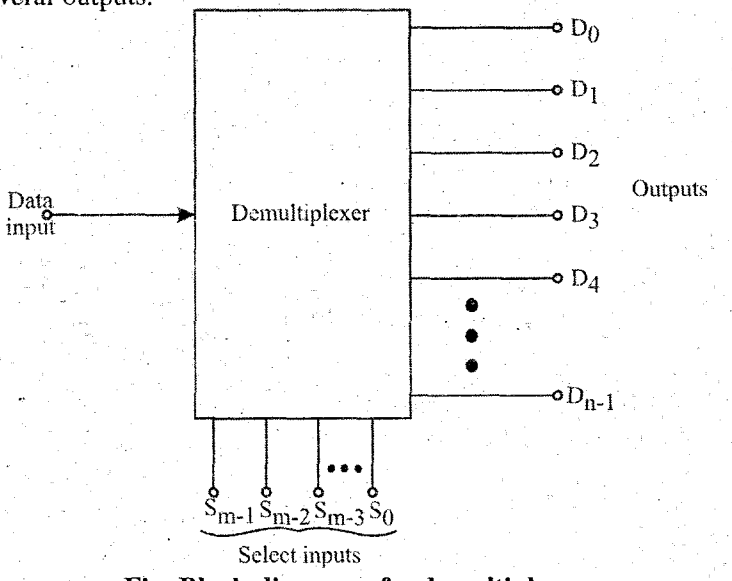
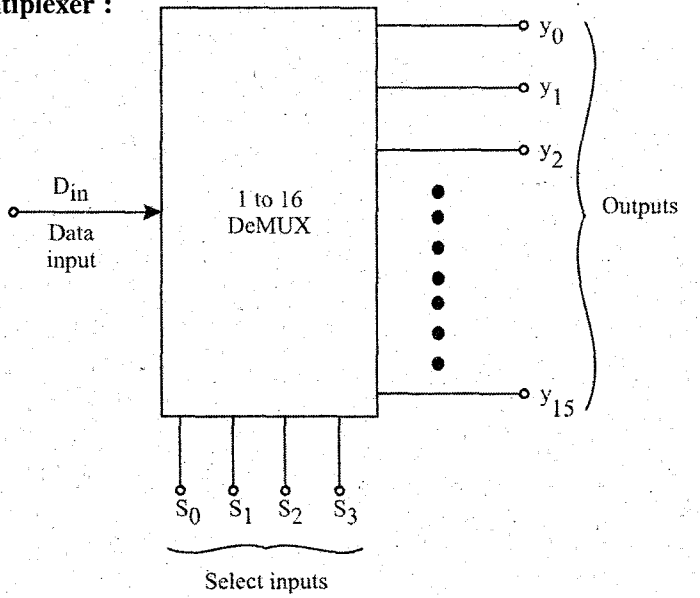


Fig. Block diagram of a demultiplexer

The select input code determines to which output the data Input will be transmitted.

The no. of output lins is n & the no. of select lines is m, where $n = z^m$. This circuit can also be used as binary-to-decimal decoder with binary Inputs applied at the select Input lines & the output will be obtained on the corresponding line. The data output line is to be connected to logic level.

1 to 16 Demultiplexer :



Block diagram of 1 : 16 De MUX

Circuit Diagram of 1 to 16 De mux :

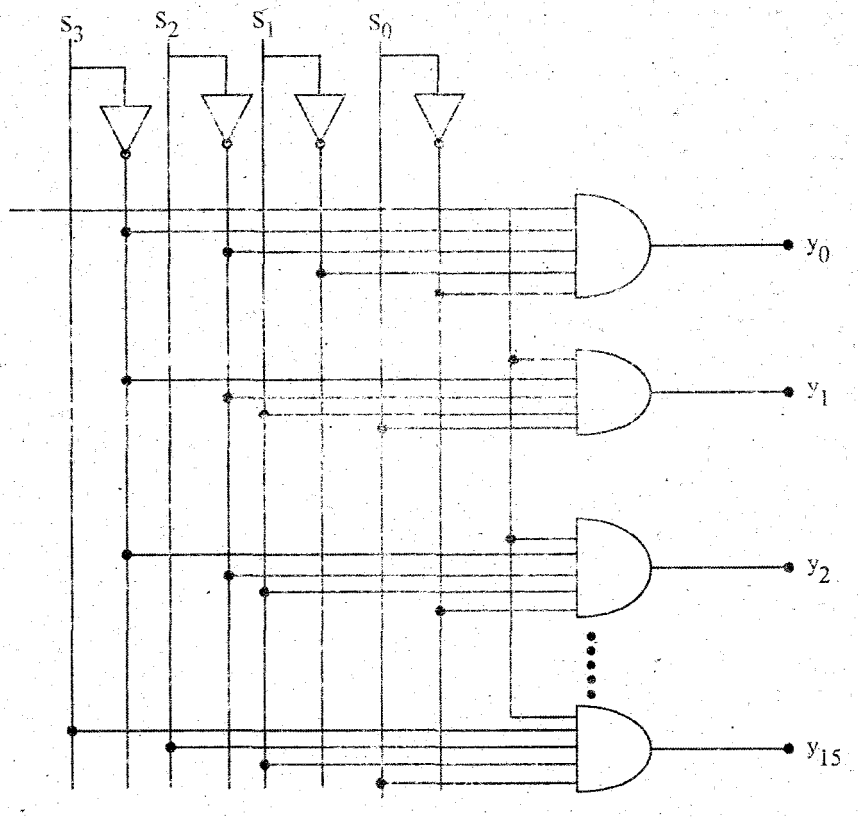
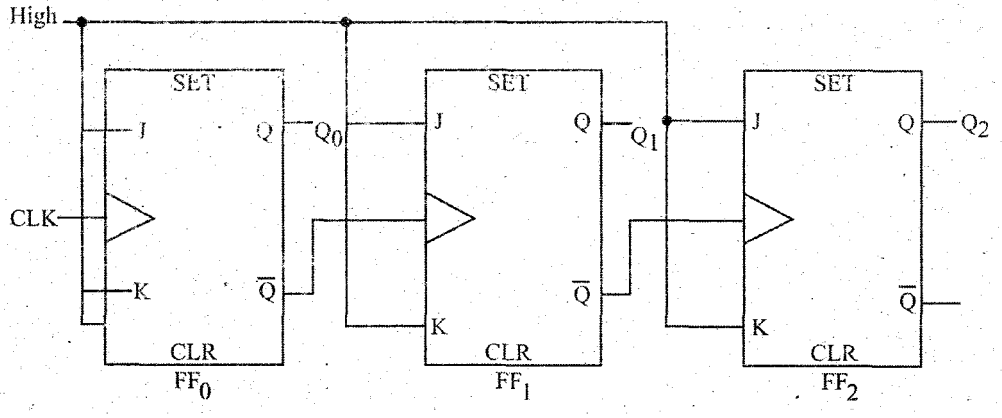
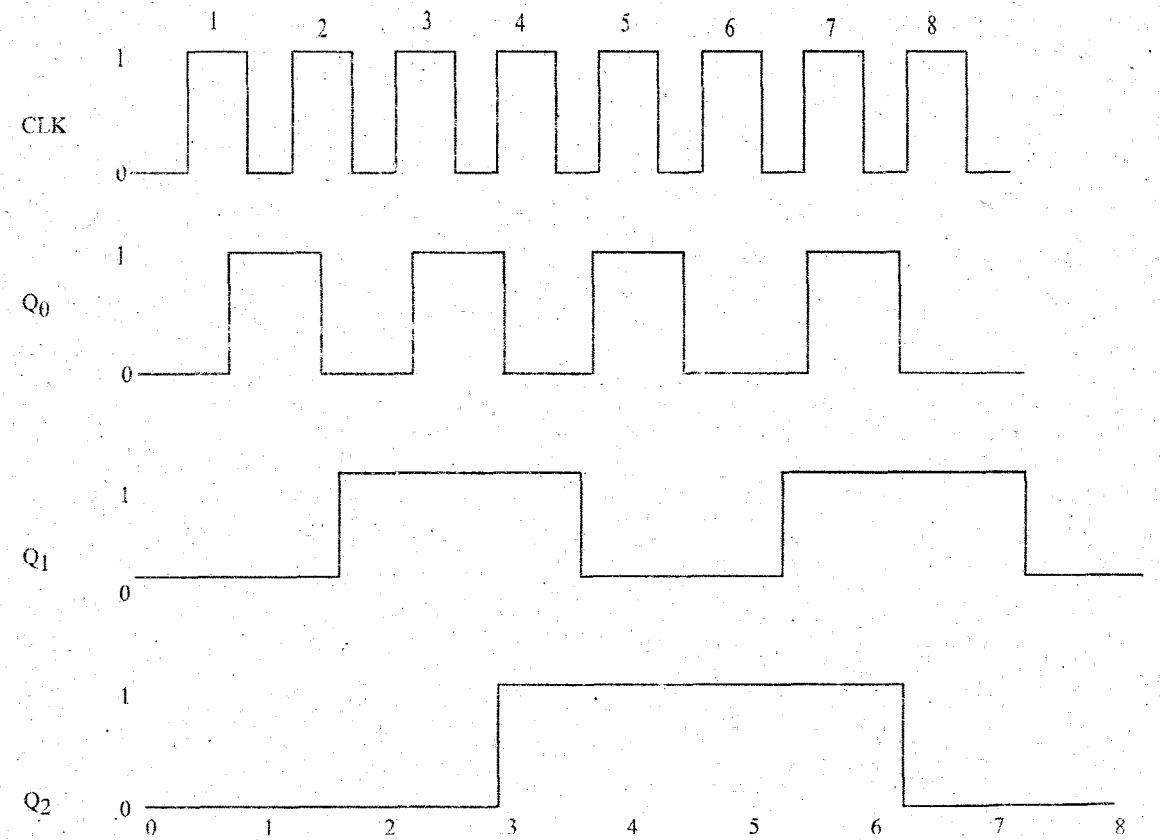


Fig. : 1 : 16 DEMUX

Section—C

Q. 6. Design :
(i) MOD-8 ripple counter
(ii) Decode synchronous counter using J-K Flip-Flops
Ans. (i) MOD-8 Ripple Counter : MOD-8 Ripple counter use 3-flip-flops.



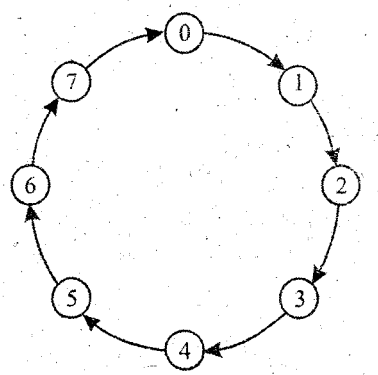


Waveform of MOD-8 Ripple counter

MOD-8 has eight states due to the third flip-flop.

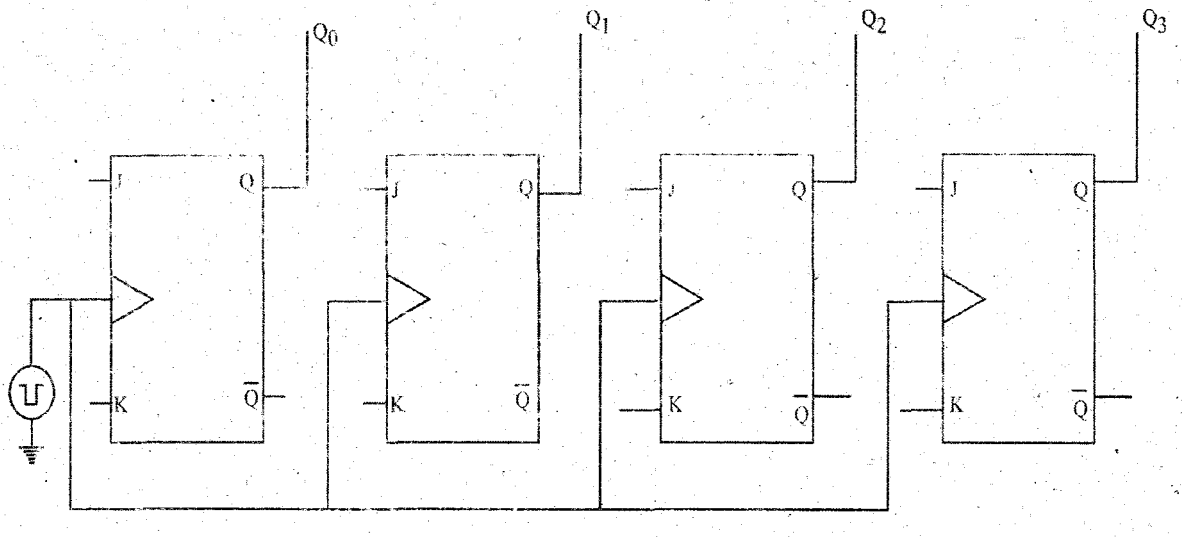
Present State			Next State		
C	**B**	**A**	**C**	**B**	**A**
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

State Diagram



(ii) Synchronous Counter Using J-K Flip-Flops :

A synchronous counter, in contrast to an asynchronous counter is one whose outputs bits change state simultaneously with no ripple. The only way we can build such a counter circuit from J-K flip-flop is to connect all the clock inputs together, so that each and every flip-flop receives the exact same clock pulse at the exact same time.



For synchronous counters, all the flip-flops are using the same clock signal. Thus, the output would change synchronously.

Procedure to Design :

Step 1 : Obtain the state diagram.

Step 2 : Obtain the excitation table.

Step 3 : Obtain & simplify K-map.

Step 4 : Draw the circuit.

Q. 7. (a) What is a shift register ? Draw circuit diagram for :

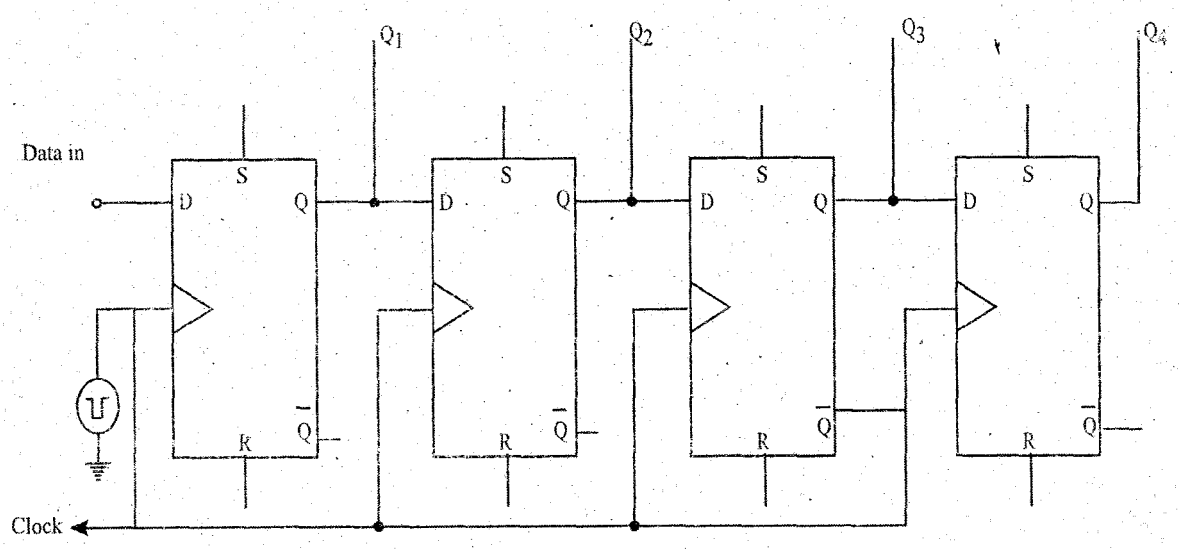
(i) Serial in/parallel out

(ii) Parallel in/serial out shift registers using J-K Flip-Flops.

Ans. In a digital circuits, a shift register is a cascade of flip-flops, sharing the same clock, which has the output of any one but the last flip-flop connected to the "data" input of the next one in the chain resulting in a circuit that shifts by one position in "bit-array", shifting in the data present at its inputs and shifting out the last bit in the array when enabled to do so by a transition of the clock input.

(i) Serial in/Parallel Out :

This conversion allows conversion from serial to parallel format. Data is input serially (SISO).



4-Bit Serial in Parallel out shift register

(ii) Parallel in/Serial Out :

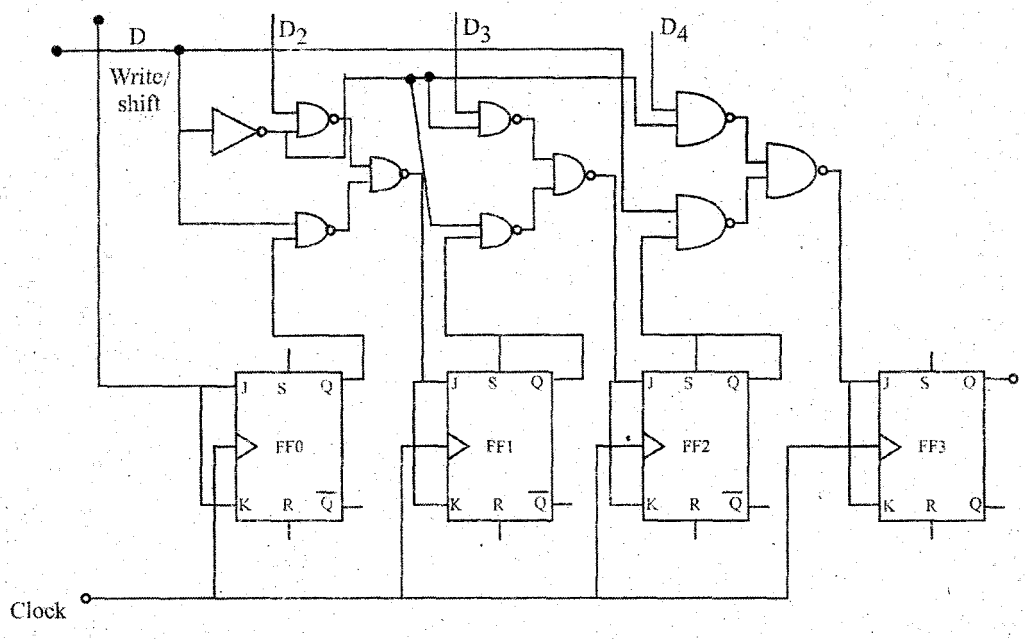


Fig. 4-bit PISO shift register

Q. 7. (b) Convert J-K Flip-Flop into D Flip-Flop.

Ans. We know that the J-K flip-flop has four different transitions from the present state to next state.

i.e.,

Present State	Next State	J-K J_n	J-K K_n
0	0	0	×
0	1	1	×
1	0	×	1
1	1	×	0

×-represent donot care condition, it is either 1 or 0. From the truth table of D flip-flop it is evident that, the value of next state depends on the D input. In simple words the value of D input is retained for the next state of the system.

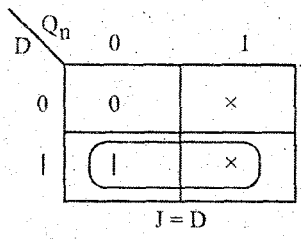
P.S.	N.S.	D-FF D_n
0	0	0
0	1	1
1	0	0
1	1	1

Now from the J-K exitation table & D excitation table. We can deduce the exitation table for the conversion of J-K flip-flop to D flip-flop.

Input	Present State	Next State	Flip-flop Inputs	
0	Q_n	Q_{n-1}	J	K
0	0	0	0	×
0	1	1	×	1
1	0	1	1	×
1	1	1	×	0

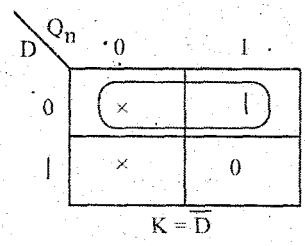
From the above excitation table we draw two separate K-map for J & K inputs respectively.

For J



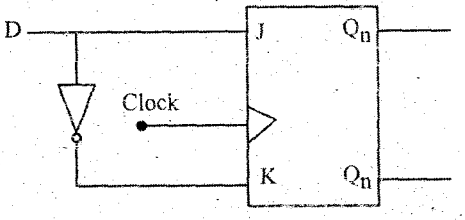
J = D

For K



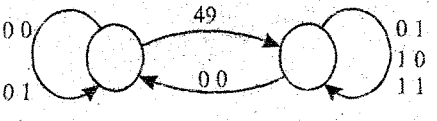
$K = \overline{D}$

From this characteristic equation we can draw the logic circuit i.e., conversion from J-K to D flip-flop.



Section—D

Q. 8. (a) Design an asynchronous sequential logic circuit for state transition diagram shown in figure :

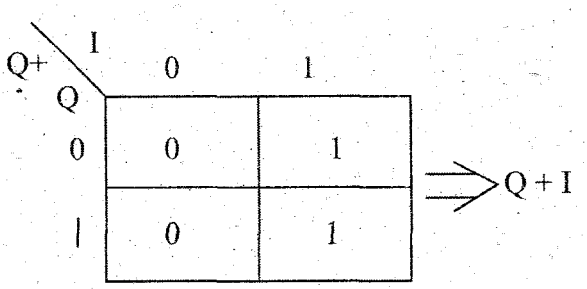


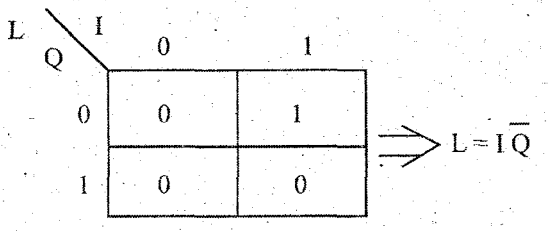
Ans. **State Table**

Present State		Next State				Output	
		X = 0		X = 1		X = 0	X = 1
A	B	A	B	A	B		
0	0	0	1	1	1	0	1
0	1	1	0	0	0	0	0
1	0	1	1	0	1	1	0
1	1	0	0	1	0	1	1

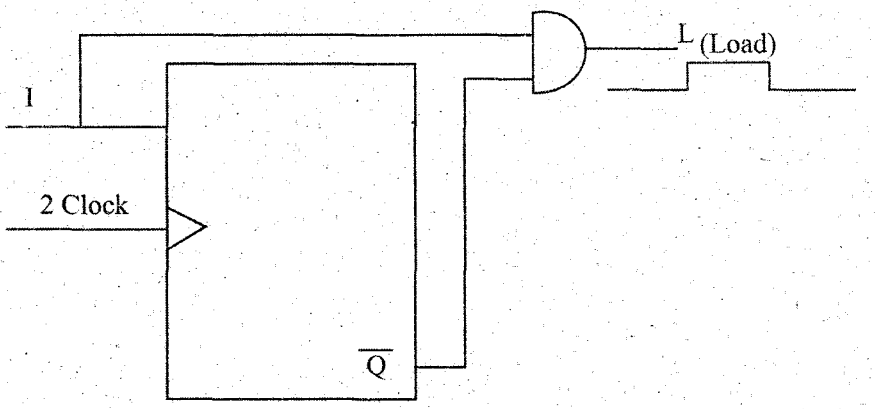
State Assignment Table :

I	Present State	Next State	Present Output
	Q	$Q+1$	L
0	0	0	0
1	0	1	1
1	1	1	0
0	1	0	0





(d) K-map



Circuit Realisation

Q. 8. (b) Implement a full adder using PLA.

Ans. The PLA can be considered as a direct POS (or SOP) implementation of a set of switching functions, with a set of AND functions followed by a set of OR functions. A PLA is often said to have an "AND" plane followed by an "OR" plane. In practice, either NAND or NOR gates are normally used; with the resulting PLA said to be a NAND/NAND or a NOR/NOR. Note that since the full adder does not require the minterm $\overline{A} \cdot \overline{B} \cdot \overline{C}$ this minterm is not included in the "AND" plane of the PLA.

Thus, PLA is efficient device for the implementation of several functions of the same set of variables.

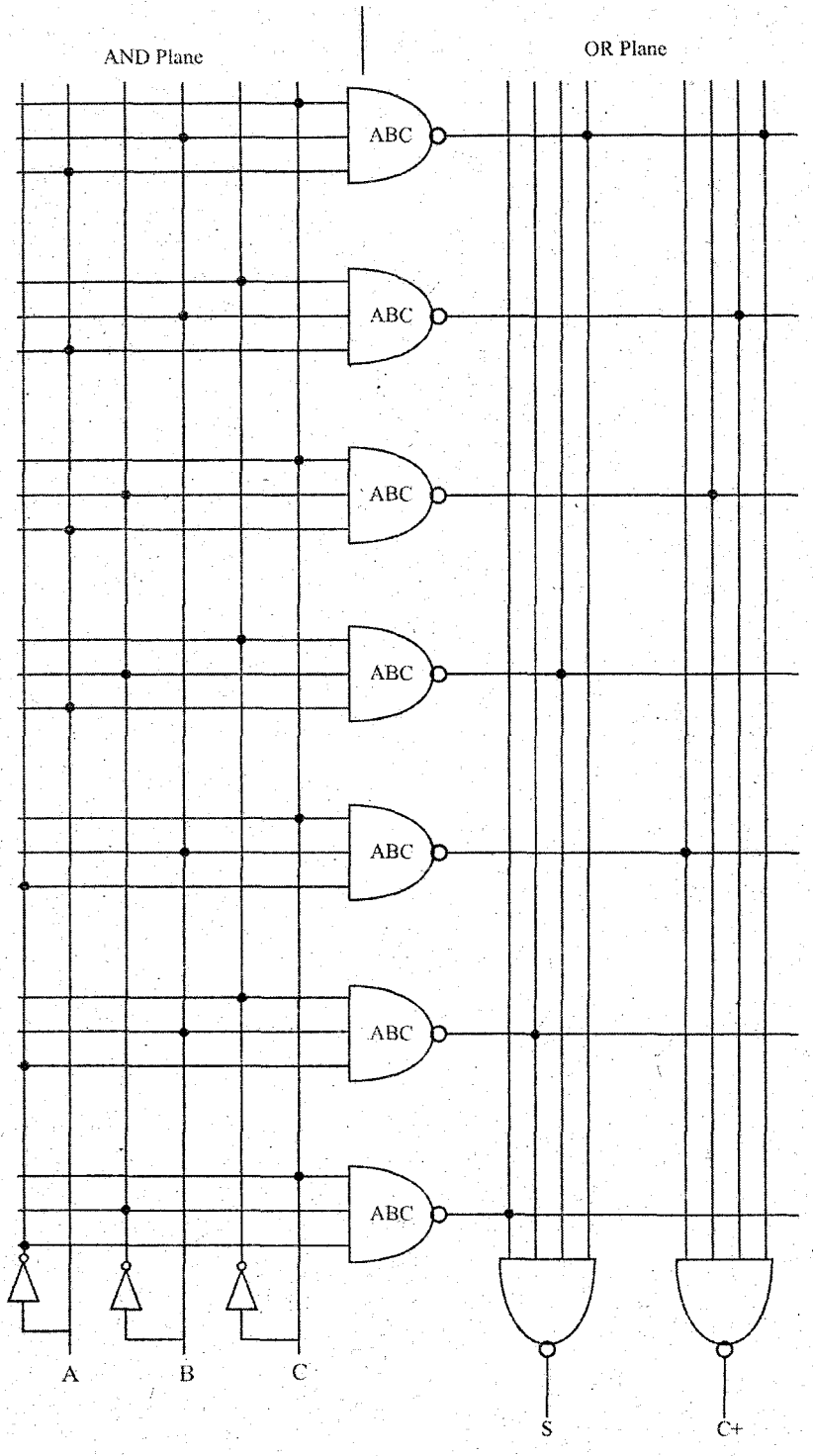


Fig. A PLA implementation of a full adder

Q. 9. Write short notes on :

(i) RAM and ROM **(ii) Algorithm State Machine (ASM)**

Ans. (i) RAM : RAM stands for Read Access Memory. It is a form of computer storage data. Today it takes the form of integrated circuit that allow stored data to be accessed in any order with a worst case performance of constant time. RAM is associated with volatile type of memory in which the information is stored as long as the power is switched on.

Types of RAM : Static RAM (SRAM)
Dynamic RAM (DRAM)

Flip-flop is an example of RAM. In such a memory, data can be put into (writing into the memory) or retrieved from (reading from the memory) in a random fashion & is known as Random-access memory.

(ii) ROM : ROM stands for Read Only Memory. ROM is a class of storage medium used in computers and other electronic devices. Data stored in ROM can't be modified, or can be modified only slowly or with difficulty, so it is mainly used to distribute firmware. In ROM information is stored even if the power is removed. It is a type of non-volatile memory.

Types : PROM (Programmable ROM)
EPROM (Erasable PROM)
EEPROM (Electrically EPROM) or flash ROM
& Ultra-violet ROM.

In a strict sense, ROM refers only to mask ROM (the oldest type of solid state ROM) which is fabricated with the desired data permanently stored in it.

(ii) Algorithm State Machine (ASM) : The algorithm state machine (ASM) is a method for designing finite state machines. It is used to represent diagrams of digital integrated circuits. The ASM diagram is like a state diagram but less formal and thus, easier to understand. An ASM chart is a method of describing the sequential operations of a digital system.

ASM Method :

ASM method is composed of the following steps :

(i) Create an algorithm, using pseudocode, to describe the desired operation of the device.

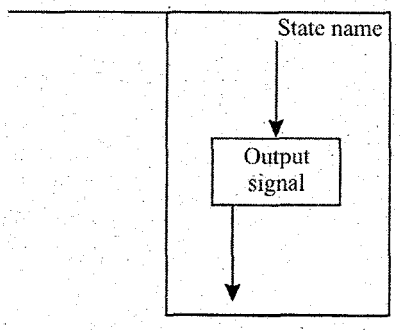
(ii) Convert the pseudocode into an ASM chart.

(iii) Design the data path based on the ASM chart.

(iv) Create a detailed ASM chart based on the data path.

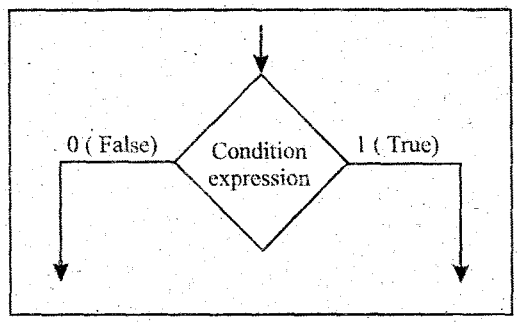
(v) Design the control logic based on the detailed ASM chart.

ASM Chart

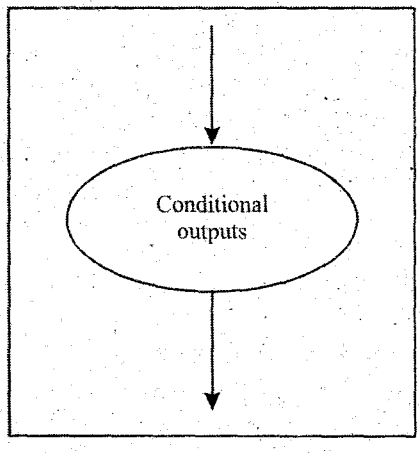


State Box

Data Path



Decision Box



Once the data path is designed, the ASM chart is converted to a detailed ASM chart. The RTL notation is replaced by signals defined in the data path.

Vertices of ASM :