Q1] a) Convert (451.43)10 into octal, binary and hexadecimal and base 7.

(04)

Solution :-

Decimal to octal

8	451	
8	56	3
8	7	0
	0	7

451 = 703

0.43 x 8 = 3.44   3
0.44 x 8 = 3.52   3
0.52 x 8 = 4.16   4
0.43 = (0.334)
$(451.43)_{10} = (703.334)_8$

Decimal to binary

2	451	
2	225	1
2	112	1
2	56	0
2	28	0
2	14	0
2	7	0
2	3	1
2	1	1
	0	1

451 = 111000011
0.43 x 2 = 0.86   0

0.86 x 2 = 1.72   1
0.43 = 0.01
(451.43)₁₀ = (111000011.01)₂

Decimal to hexadecimal

16	451	
16	28	3
16	1	12
	0	1

451 = (1 12 3)
0.43 x 16 = 6.88   6
0.88 x 16 = 14.08   14
0.08 x 16 = 1.28   1
0.43 = (0.6 14 1)
(451.43)₁₀ = (1 12 3.6 14 1)₁₆

Decimal to base 7

7	451	
7	64	3
7	9	1
7	1	2
	0	1

451 = (1 2 1 3)
0.43 x 7 = 3.01   3
0.01 x 7 = 0.07   0
0.43 = (0.30)
(451.43)₁₀ = (1213.30)₇

Q1] b) Subtract using 1's and 2's complement method (73)10 – (49)10.

(04)

 Solution :-

1's complement method
Binary representation of 49 is 0110001
1's complement of 49 is 1001110

```
      1001110

    + 1001001
  _____
      10010111

    +           1
  _____
      10011000
```

Therefore $(73)_{10} - (49)_{10} = 11000 = (24)_{10}$.

2's complement method.

Binary representation of 49 is 0110001

1's complement of 49 is 1001110

2's complement of 49 is 1's complement + 1

```
      1001110

    +        1
  _____
      1001111
```

Now adding 73 and 2's complement of 7

```
      1001001

    + 1001111
  _____
    1 00 11000
```

Discard carry 1

Therefore $(73)_{10} - (49)_{10} = 11000 = (24)_{10}$.

Q1] c) Perform $(52)_{10} - (68)_{10}$ in BCD using 9's complement.

(04)

 Solution :-

First subtract 68 from 99 that is find 9's complement of 68

```
      9 9

    -  6 8
  _____
      3 1
```

We then add 52 and 9's complement of 68

```
      5 2
```

```
    +      3 1
   _____
           8 3
```
(52)10 - (68)10 = (83)

Q1] d) State De Morgan's theorem. Prove OR-AND configuration is equivalent to NOR-NOR configuration.

(04)

Solution :-

Theorem 1:

Statement: The complement of the product of two or more variables is equal to the sum of the components of the variables.

Logical expression: $\overline{A.B} = \overline{A} + \overline{B}$

NAND is equal to BUBBLED OR.

A	B	A.B	$\overline{A.B}$	$\overline{A}$	$\overline{B}$	$\overline{A} + \overline{B}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

From the above truth table we can see that $\overline{A.B} = \overline{A} + \overline{B}$ Hence proved.

Theorem 2 :-

Statement: The complement of the sum of two or more variables is equal to the product of the complement of the variables

Logical expression: $\overline{A + B} = \overline{A}.\overline{B}$

NOR is equal to BUBBLED AND.

A	B	A + B	$\overline{A + B}$	$\overline{A}$	$\overline{B}$	$\overline{A}.\overline{B}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

From the above truth table we can see that $\overline{A + B} = \overline{A}.\overline{B}$ Hence proved.

Q1] e) Encode the data bits 111010001 using Hamming code.

(04)

Solution :-

For given data of 9 bit, hamming code is given as

D13	D12	D11	D10	D9	P8	D7	D6	D5	P4	D3	P2	P1

Given 9 bit is 111010001 therefore it can be written as

1	1	1	0	1	P8	0	0	0	P4	1	P2	P1

P1: For P1 consider P1, D3, D5, D7, D9, D11, D13
        D3, D5, D7, D9, D11, D13 = 100111
        For even parity set P1 = 0
P2: For P2 consider P2, D3, D6, D7, D10, D11
        D3, D6, D7, D10, D11 = 10001
        For even parity set P2 = 0
P4: For P1 consider P4, D5, D6, D7, D12, D13
        D5, D6, D7 = 00011
        For even parity set P4 = 0
P8: For P1 consider P8, D9, D10, D11, D12, D13
        D9, D10, D11, D12, D13= 10111
        For even parity set P4 = 0

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

Therefore, hamming code for 111010001 is given as 1110100000100.

Q1] f) Explain SOP and POS and solve the following using K-Map
F(A,B,C,D)=πM(1,3,5,6,7,10,11)+d(2,4)

(04)

Solution :-

The sum-of-products (SOP) form is a method (or form) of simplifying the Boolean expressions of logic gates. In this SOP form of Boolean function representation, the variables are operated by AND (product) to form a product term and all these product terms are ORed (summed or added) together to get the final function.

The product of sums form is a method (or form) of simplifying the Boolean expressions of logic gates. In this POS form, all the variables are ORed, i.e. written as sums to form sum terms.

F(A,B,C,D)=πM(1,3,5,6,7,10,11)+d(2,4)

Fig.1.1 K map representation.

Equation form is

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

Q1] g) Explain lockout condition. How can it be avoided?

(04)

Solution :-

In counters, Lockout condition is that condition wherein a counter gets onto a forbidden state and rather than coming out of it to another acceptable state or initial state, the counter switches to another forbidden state and gets stuck up in the cycle of forbidden states only.
If by chance, the counter happens to find itself in any one of the unused states, its next state would not be known. It may just be possible that the counter might go from one unused state to another and never arrive at a used state. A counter whose unused states have this feature is said to suffer from LOCK OUT.

To avoid lock out and make sure that at the starting point the counter is in its initial state or it comes to its initial state within few clock cycles, external logic circuitry is to be provided and so we design the counter assuming the next state to be the initial state, from each unused states.
In order to prevent lock out condition, use PRESET option and clear option in a flip flop and hence the registers will be cleared and we can go for the smooth functioning of counter circuits.

Q2] a) Reduce equation using Quine McCluskey method and realize circuit using (10) basic gates.
-F(A,B,C,D) = Σm (1,5,6,12,13,14) + d(2,4)

(10)

Solution :-

Table 1

Group	Minterm	Binary Representation			
		A	B	C	D
0	m1	0	0	0	1
	m2*	0	0	1	0
	m4*	0	1	0	0
1	m5	0	1	0	1
	m6	0	1	0	1
	m12	0	1	1	0
2	m13	1	1	0	1
	m14	1	1	1	0

Table 2

Group	Minterm	Binary Representation			
		A	B	C	D
0	m1-m5	0	_	0	1
	m2-m6*	0	_	1	0
	m4-m5*	0	1	0	_
	m4-m12*	_	1	0	0
1	M5-m13	_	1	0	1
	M6-m14	_	1	1	0
	M12-m13	1	1	0	_
	M12-m14	1	1	_	0

Table 3

Group	Minterm	Binary Representation			
		A	B	C	D
0	M4-m5-m12-m13*	_	1	0	_
	M4-m12-m5-m13*	_	1	0	_
	M4-m12-m6-m14*	_	1	_	0

PI	Decimal of PI	1	5	6	12	13	14	2	4
$B\bar{C}$	M4 m5 m12 m13		⊗		X	⊗			X
$B\bar{D}$	M4 m6 m12 m14			⊗	X		⊗		X

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

Q2] b) Design 4-bit BCD subtractor using IC 7483.

(10)

 Solution :-

4 Bit BCD subtractor

IC 7483 performs the addition of two 4-bit BCD numbers A3 A2 A1 A0 and B3 B2 B1 B0 and carry input to give the output S3 S2 S1 S0 and carry out. So for adding the two numbers A3 A2 A1 A0 and B3 B2 B1 B0, the two numbers are given to input terminals 1,3,8,10 and 16,4,7,11 of the IC 7483 and carry in the terminal 13 is set to zero. To subtract two numbers by two's complement method, we are adding the 2's complement of the second number to each of the four bits of the first numbers. The final carry is neglected and the difference is taken from S3 S2 S1 S0.

In the circuit we set mode control such that when the mode control is zero, addition is performed and subtraction is performed when the mode control is one. We use XOR gates to feed the input so that when control is one, the complement of each of the four bits are fed and mode control is zero, the input as such is fed.

In 1's complement subtraction, the complement of the subtraction is taken and added with the other number. The final carry is then added to LSB of the result. In case there is no carry, the complement of the result is taken and this will be a negative number. This indicates that, subtraction is performed from a smaller number.
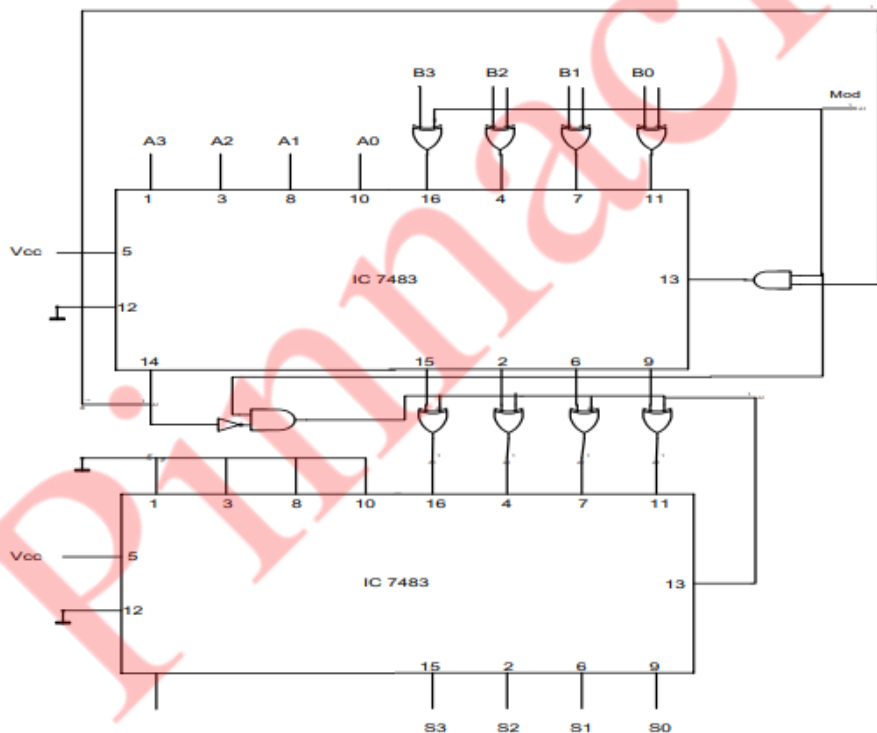

Fig 2 - 4 bit BCD subtractor using IC 7483

Q3] a) Implement the following using only one 8:1 Mux.
F(A,B,C,D) = Σm (0,2,4,6,8,10,12,14)

(05)

Solution :-

Step 1: Draw a table

	D0	D1	D2	D3	D4	D5	D6	D7
A	⓪	1	②	3	④	5	⑥	7
A	⑧	9	⑩	11	⑫	13	⑭	15

If both the values in the column are included than it is connected to 1. If both the values of the column are not included than it is connected to input 0. So, for the given question the mux drawn is given below.
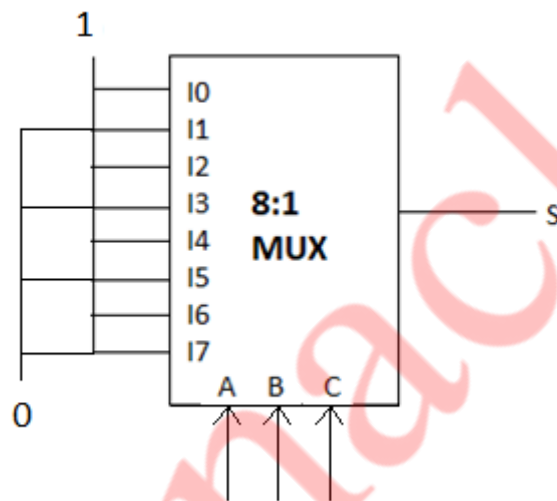
Step 2: MUX implementation

Fig. 3a 8:1 MUX

Q3] b) Design a Full Subtractor using only NAND gates.

(05)

Solution:-

Truth table of full subtractor

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

D = X $\oplus$ Y $\oplus$ Z

X \ YZ	0	1	1	1
	0	0	1	0

B = $\overline{X}$Y + $\overline{X}$ $\overline{Y}$ Z + XYZ

Representation of full subtractor using NAND gates.
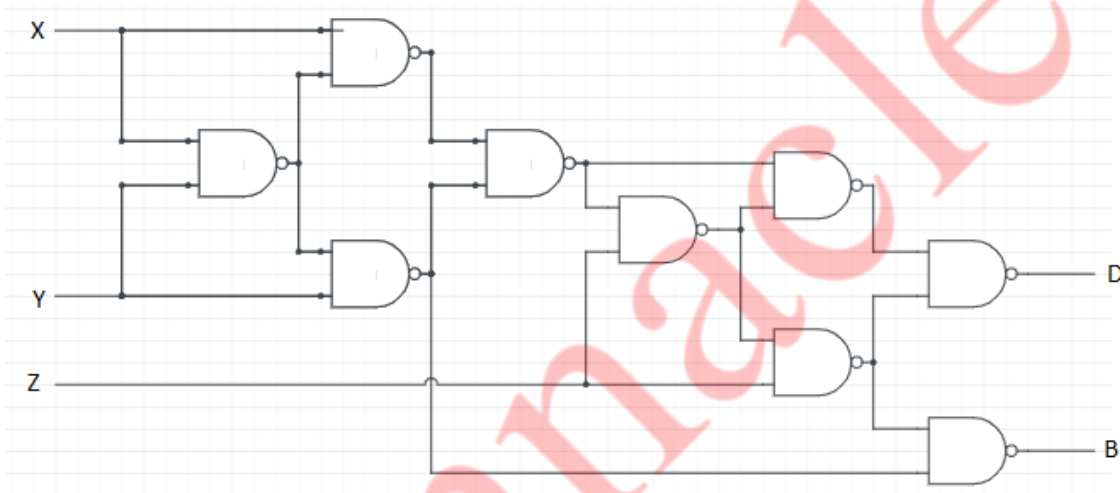
Q3] c) Design a logic circuit to convert 4-bit gray code to its corresponding BCD code.

(10)

Solution:-

Step1: Write the truth table of Gray code and its equivalent binary code is as shown in table:

Gray Code Input				Binary Code Output			
G3	G2	G1	G0	B3	B2	B1	B0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	1	0	0	1	0
0	0	1	0	0	0	1	1
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
0	1	0	1	0	1	1	0
0	1	0	0	0	1	1	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	1	1	0	1	0
1	1	1	0	1	0	1	1
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	0	0	1	1	1	1	0
1	0	0	0	1	1	1	1

Step2: Write K – map for each binary output and get simplified expression The K – map for various binary outputs and the corresponding simplified expression are given below:

For output B3

Fig.3.1 K map for B3

Simplified expression B3=G3
For output B2

Fig 3.2 K map for B2

B2 = G(3)$(\overline{G2})$ + G(2)$(\overline{G3})$
   =G3 $\oplus$ G2

Fig 3.3 K map for B1

B1 = G1 $\overline{G2}\,\overline{G3}$ + G2 $\overline{G3}\,\overline{G2}$ + G3 G2 G1 + G3 $\overline{G1}\,\overline{G2}$

   = $\overline{G1}$ (G2$\oplus$G3) + G1 (G2$\oplus$G3)

   = $\overline{G1}X$ + $\overline{G1}X$ Where X = G2 $\oplus$ G3

    = G1 $\oplus$ G2 $\oplus$ G3

For output B0

Fig 3.4 K map for B0

B0 = G0 $\overline{G1}\,\overline{G2}\,\overline{G3}$ + G2 $\overline{G3}\,\overline{G1}\,\overline{G0}$ + G1 $\overline{G3}\,\overline{G2}\,\overline{G0}$ + G1 $\overline{G3}\,\overline{G2}$ G0 + G2 $\overline{G3}$ G1 G0 + G2 $\overline{G1}$ G3 G0 +

    G2 $\overline{G0}$ G3 G1 + G3 $\overline{G1}\,\overline{G2}\,\overline{G0}$ + G0 $\overline{G2}$ G3 G1

  = $\overline{G0}\,\overline{G1}$ (G2$\oplus$G3) + G0 $\overline{G1}$ ($\overline{G3 \oplus G2}$) + G0 $\overline{G1}$ ($\overline{G3 \oplus G2}$) + G1 $\overline{G0}$ ($\overline{G3 \oplus G2}$)

  = (G2$\oplus$G3) + ($\overline{G1 \oplus G0}$) +(G1$\oplus$G0) + ($\overline{G3 \oplus G2}$)

  = $\overline{Y}$ X + Y $\overline{X}$

Where X = (G2$\oplus$G3) and Y=(G1$\oplus$G0)

  = (G2$\oplus$G3) $\oplus$ (G1$\oplus$G0)

  = G2$\oplus$G3$\oplus$G1$\oplus$G0

Step3: Realization

The Gray to Binary code converter is as shown in figure given below

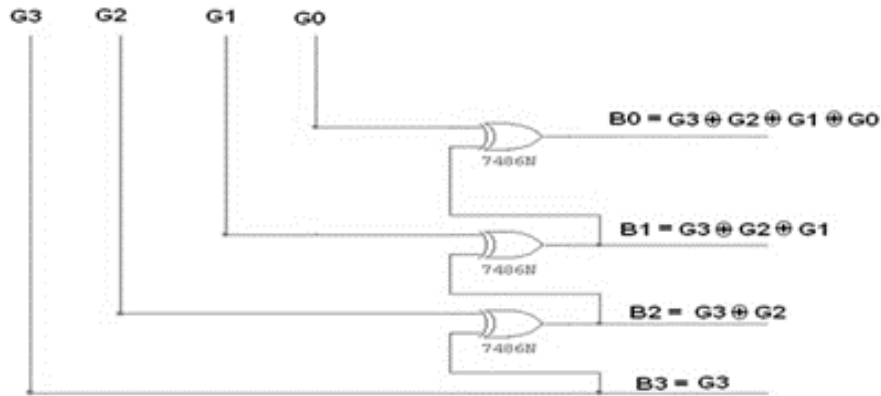Fig. 3.5 Gray code to binary converter

----

Q4] a) Compare different logic families with respect to fan in, fan out, speed, Propogation delay and power dissipation.

(05)

Solution :-

Parameter	ECL	DTL	TTL	IIL	CMOS
Fan in	-	-	12-14	-	Greater than 10
Fan out	25	Medium (8)	High (10)	8-11	50
Speed X Power	100	300	100	Less than 1	70
Propagation Delay (ns)	1-2	30	10	25-2500	70
Power Dissipation (mW)	40-45	8-12	10	5-25	01

Q4] b) Implement 3 bit binary to gray code converter using Decoder.

(05)

Solution :-

Truth table:-

A	B	C	X	Y	Z
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

Formulae:-

Let b2 b1 b0 be the 3-bit binary number and g2 g1 g0 be its equivalent gray code.

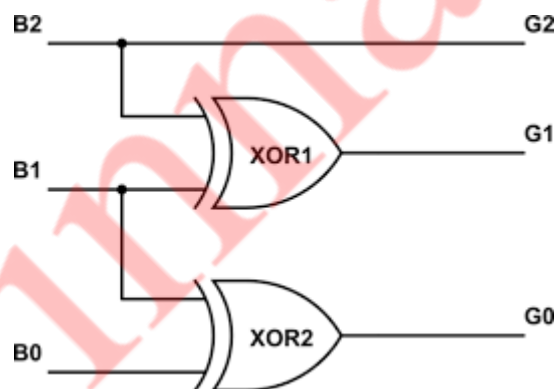Then,

$g2 = b2$

$g1 = b2 \oplus b1$

$g0 = b1 \oplus b0$

Fig. 4.1 Binary to gray code converter.

Q4] c) Explain 4 bit bidirectional shift register.

(10)

Solution :-

- The binary information (data) in a register can be moved from stage to stage within the register or in or out of the register upon applications of clock pulses.
- This type of bit movement or shifting is essential for certain application like arithmetic and logic operations used in Microprocessors.

- This group of registers are called as 'Shift Register'

4 –bit Bidirectional Shift Register:

- Bidirectional shift register allows shifting of data either to left or to the right side.
- It can be implemented using logic gates circuitry that enables the transfer of data from one stage to the next stage to the right or to the left, depend on the level of control line.
- The RIGHT/LEFT is the control input signal which allows data shifting either towards right or towards left.
- A high on this line enables the shifting of data towards right and low enables it towards left.
- When RIGHT/LEFT is high, gates G1, G2, G3 and G4 are enabled.
- The state of Q output of each flip flop is passed through the D input of the following flip flop.
- When the pulse arrives, the data are shifted one place to the right.
- When the RIGHT/LEFT signal is low, gates G5, G6, G7 are enabled.
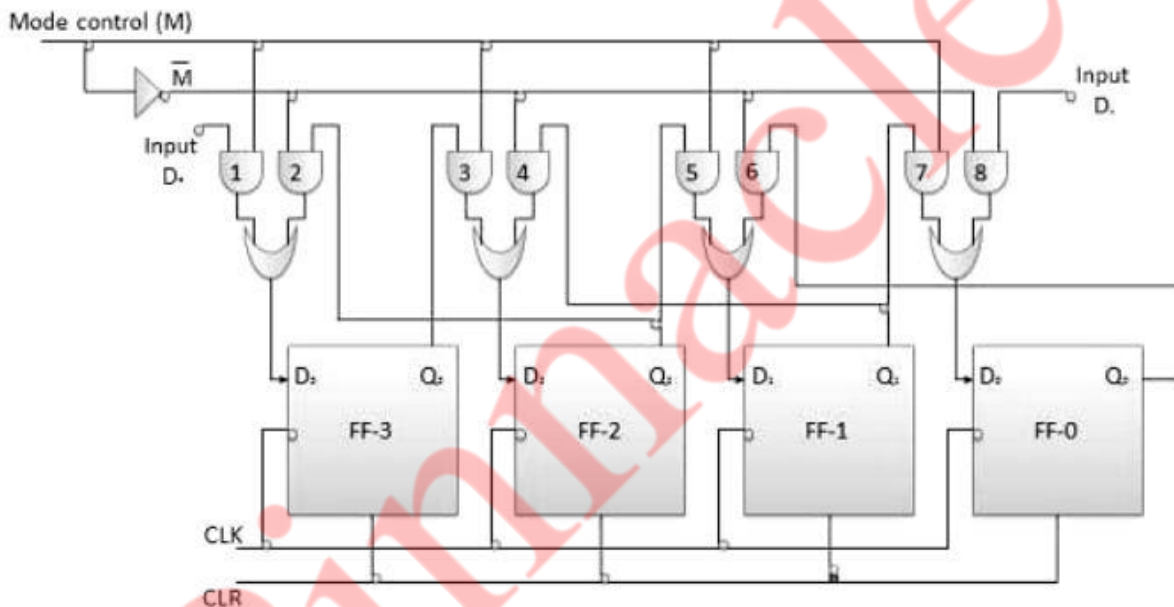- The Q output of each flip-flop is passed through the D input of the preceding flip-flop.

Fig.4.s2 - 4 bit Bidirectional Shift Register

Q5] a) Design mod 13 synchronous counter using T flipflop.

(10)

Solution :-

A synchronous counter is one which has the same clock input for all its flip flops. A MOD 13 synchronous counter counts from 0000 to 1101. Hence it will require four T flip flops. Synchronous counters are designed by using excitation table to determine the combinational logic of inputs to each flip flop. The excitation table for all the four T flip flops is shown:

Present state				Next state				T Flipflop			
Q_D	Q_C	Q_B	Q_A	Q_D^+	Q_C^+	Q_B^+	Q_A^+	T_D	T_C	T_B	T_A
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	1
0	0	1	1	0	1	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	1
0	1	0	1	0	1	1	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	1
0	1	1	1	1	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	1
1	0	0	1	1	0	1	0	0	0	1	1
1	0	1	0	0	0	0	0	1	0	1	0
1	0	1	1	X	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X	X

Fig 5.a Excitation table for MOD 13 synchronous counter

From the above excitation table, we can draw k maps to determine input to every flipflop

$T_D = Q_C Q_B Q_A$

$T_C = Q_B Q_A$

$T_B = Q_A + Q_D Q_B$

$T_A = \overline{Q_D} + \overline{Q_B}$
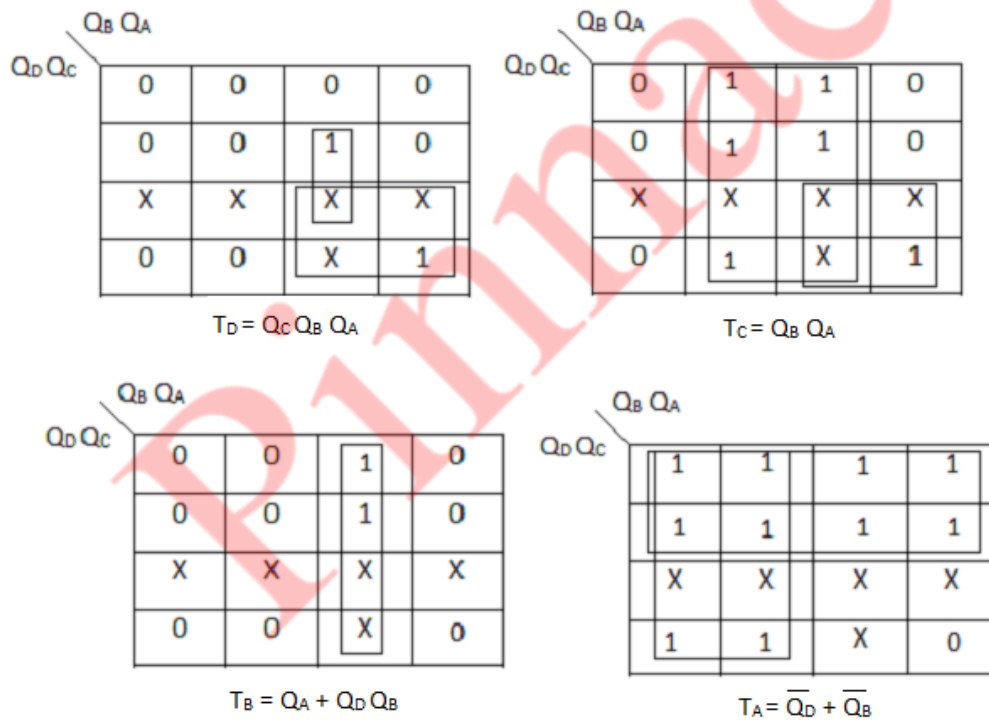
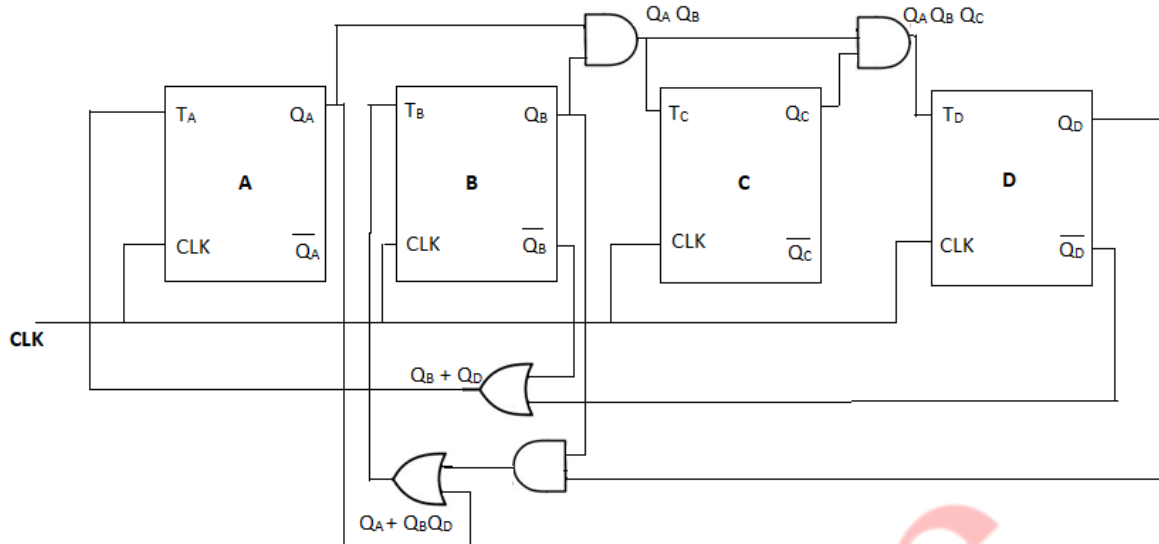Four equations for four T flipflop are obtained. Using them, the MOD 13 synchronous counter is designed as follows:

Fig 5.2 MOD 13 synchronous counter using T flipflop

Q5 b) Convert SR flipflop to JK flipflop and D flipflop.

(10)

Solution:-

SR to JK

a) Excitation Table

Input's		Present State	Next State	Flip Flop Input	
J	K	Qn	Qn+1	S	R
0	0	0	0	0	X
0	1	0	0	0	X
1	0	0	1	1	0
1	1	0	1	1	0
0	0	1	0	0	1
0	1	1	0	0	1
1	0	1	1	X	0
1	1	1	1	X	0

b) K- map simplification
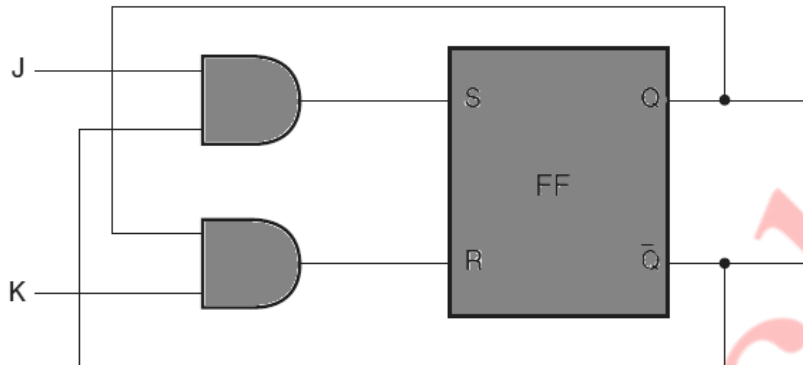
c) Logic diagram

SR to D

a) Excitation Table

Input	Present State	Next State	Flip Flop Input	
D	Qn	Qn+1	S	R
0	0	0	0	X
1	0	1	1	0
0	1	0	0	1
1	1	1	X	0

b) K-map simplification

$S = D$

$R = \overline{D}$
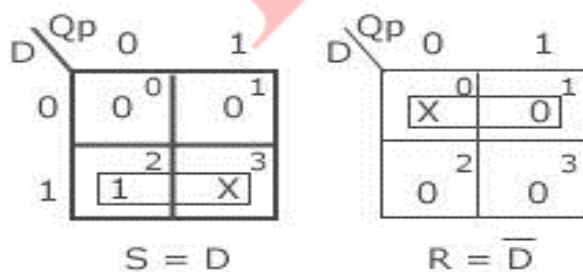
c) Logic diagram
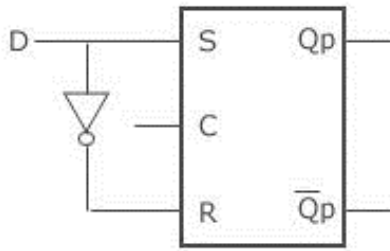
Q6] a) ALU

(05)

Solution :-

The arithmetic logic unit is that part of the CPU that handles all the calculations the CPU may need. Most of these operations are logical in nature. Depending on how the ALU is designed, it can make the CPU more powerful, but it also consumes more energy and creates more heat. Therefore, there must be a balance between how powerful and complex the ALU is and how expensive the whole unit becomes. This is why faster CPUs are more expensive, consume more power and dissipate more heat.

The main functions of the ALU are to do arithmetic and logic operations, including bit shifting operations. These are essential processes that need to be done on almost any data that is being processed by the CPU.

ALUs routinely perform the following operations:

Logical Operations: These include AND, OR, NOT, XOR, NOR, NAND, etc.

Bit-Shifting Operations: This pertains to shifting the positions of the bits by a certain number of places to the right or left, which is considered a multiplication operation.

Arithmetic Operations: This refers to bit addition and subtraction. Although multiplication and division are sometimes used, these operations are more expensive to make. Addition can be used to substitute for multiplication and subtraction for division.

Q6] b) 3 bit Up/Down Asynchronous Counter

(05)

Solution :-

Asynchronous counters are those whose output is free from the clock signal. Because the flip flops in asynchronous counters are supplied with different clock signals, there may be delay in producing output.

By adding up the ideas of UP counter and DOWN counters, we can design asynchronous up /down counter. The 3 bit asynchronous up/ down counter is shown below.
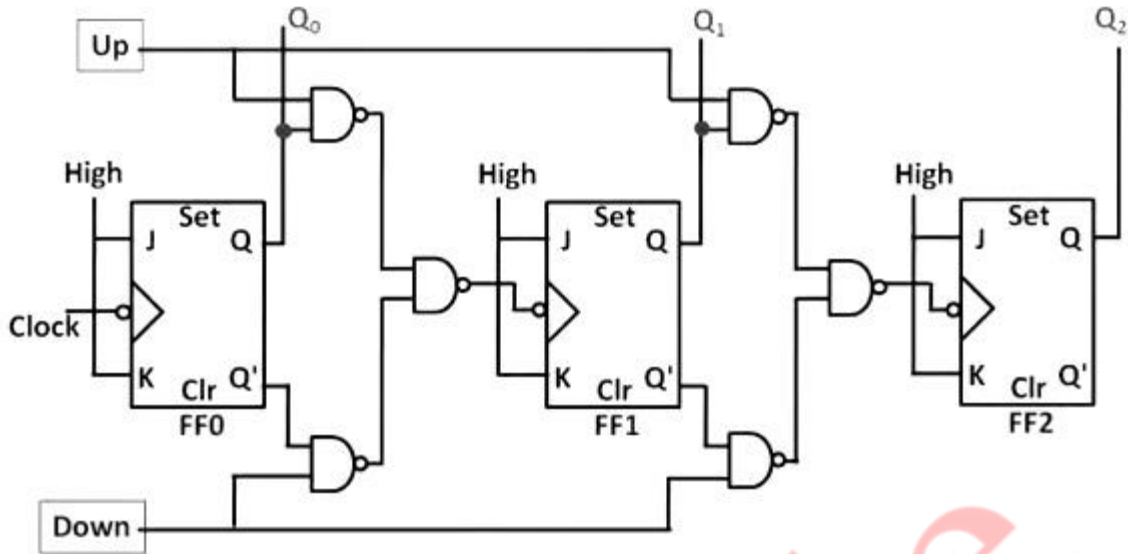
It can count in either ways, up to down or down to up, based on the clock signal input.

UP Counting

If the UP input and down inputs are 1 and 0 respectively, then the NAND gates between first flip flop to third flip flop will pass the non-inverted output of FF 0 to the clock input of FF 1. Similarly, Q output of FF 1 will pass to the clock input of FF 2. Thus, the UP /down counter performs up counting.

DOWN Counting

If the DOWN input and up inputs are 1 and 0 respectively, then the NAND gates between first flip flop to third flip flop will pass the inverted output of FF 0 to the clock input of FF 1. Similarly, Q output of FF 1 will pass to the clock input of FF 2. Thus, the UP /down counter performs down counting.

The up/ down counter is slower than up counter or a down counter, because the addition propagation delay will be added to the NAND gate network.

Q6] c) Octal to Binary Encoder

(05)

Solution:-

An encoder has 2n or fewer input lines, only one of which is in the "1" state at a particular time and an n-bit code is generated on "n" output lines depending upon which of the input is excited. In other words, an encoder is a circuit in which output lines generate the binary code corresponding to the input value.

An octal to binary encoder has 23 = 8 input lines D0 to D7 and 3 output lines Y0to Y2. Below is the truth table for an octal to binary encoder.

			Input						Output	
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

Fig.6.c Truth Table of octal to binary encoder.

From the truth table, the outputs can be expressed by following Boolean Function.
$Y_0 = D_1 + D_3 + D_5 + D_7$
$Y_1 = D_2 + D_3 + D_6 + D_7$
$Y_2 = D_4 + D_5 + D_6 + D_7$
Above boolean functions are formed by ORing all the input lines for which output is 1. For instance Y_0 is 1 for D_1, D_3, D_5, D_7 input lines. The encoder can therefore be implemented with OR gates whose inputs are determined directly from truth table.

Q6] d) 4-bit Universal shift register

(05)

Solution :-

A Universal shift register is a register which has both the right shift and left shift with parallel load capabilities. Universal shift registers are used as memory elements in computers. A Unidirectional shift register is capable of shifting in only one direction. A bidirectional shift register is capable of shifting in both the directions. The Universal shift register is a combination design of bidirectional shift register and a unidirectional shift register with parallel load provision.
The register operations performed for the various inputs of select lines are as follows:

S1	S0	REGISTER OPERATION
0	0	No changes
0	1	Shift Right
1	0	Shift Left
1	1	Parallel Load

4-bit universal shift register is built with four blocks each constituted of a 4X1 mux and a D-flipflop. All the blocks are essentially identical. Because all the multiplexers in the register are wired similarly,

The L inputs come through port 11, which is why the L inputs are readable only when S1S0 = 11. The feedback Q wires are connected at port 00, so that when S1S0 = 00 the output Q of the D-flipflops

feed back into the flipflops' inputs resulting in no total change in the register content. Port 01 is wired to facilitate right-shifts. In mode S1S0 = 01 only port 01 is active and it takes its value from the previous more significant flipflop and passes it down to the flipflop wired to its mux output. Lastly port 10 is wired to conduce to left-shifts. Being the only active port when S1S0 = 10, it remits the output of the less significant flipflop sourcing into it to the flipflop wired to its mux output. As a consequence of this wiring pattern where each block of the register is an exact replica of any other block, the selector switches are able to align the behavior of all the multiplexers simultaneously. This coincidence of behavior is what we refer to as mode behavior of the universal register.

Q6] e) VHDL

(05)

Solution :-

VHDL stands for very high-speed integrated circuit hardware description language. This language was first introduced in 1981 for the department of Defense (DoD) under the VHSIC program.

VHDL is a hardware description language that can be used a model a digital system at many levels of abstraction, ranging from the algorithmic level to the gate levels.

Structure of VHDL Module:

The Main component of VHDL module consist of following declaration.

      1) Package
      2) Entity
      3) Architecture
      4) Configuration

The following fig. shows the relationship of these basic blocks of VHDL program.

A design may include any number of packages, entity, and Architecture and configuration declaration.

The entity and architecture blocks are compulsory but packages and configuration blocks are optional.
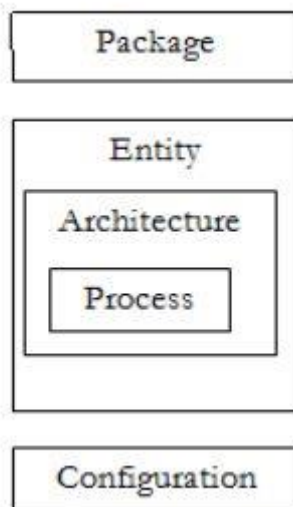

Fig 6.e Relationship of VHDL Design units

Features of VHDL are as follows:

1) Designs are organized hierarchically.
2) Each design element has:
    i)   A well-defined interface.
    ii)   A precise behavioral specification using either algorithmic description or hardware structural description.
3) Models concurrency, timing, and clocking:
    i)   Handles asynchronous and synchronous circuits
    ii)   Designs can be simulated.
4) Language is not case sensitive.
5) A formal language for specifying the behavior and structure digital circuit.