MODULE – 2

THE COMBINATIONAL LOGIC CIRCUITS

1] Prove the universality of NAND and NOR Gates. [CO2: P3]

2] State Duality theorem. State and prove De Morgan's first & second theorems. [CO2: P4]

3] Determine the minimum sum-of-products for –

a) $f1\ (a, b, c) = \sum(1, 3, 4, 5, 6, 7)$

b) $f2\ (a, b, c) = \Pi\ (2, 4, 7)$

c) $f3\ (a, b, c, d) = b'c'd' + bcd + acd' + a'b'c + a'bc'd$ [CO2: P8 or P12]

4] Determine the minimum product-of-sums for –

a) $f1\ (a, b, c) = \sum\ (0, 1, 2, 3, 4, 6, 7)$

b) $f2\ (a, b, c) = \Pi\ (1, 4, 5)$

c) $f3\ (a, b, c, d) = b'c'd' + bcd + acd' + a'b'c + a'bc'd$ [CO2: P8]

5] Solve for the simplified Boolean expression using K-Map:

a) $f1\ (a,b,c,d) = \bar{a}\bar{c}d + \bar{a}cd + \bar{b}\bar{c}\bar{d} + a\bar{b}c + \bar{a}\bar{b}c\bar{d}$

b) $f2\ (a,b,c,d) = (a + b + \bar{d})(\bar{a} + b + \bar{d})(a + \bar{b} + \bar{c} + d)(\bar{a} + \bar{b} + \bar{c} + \bar{d})(\bar{a} + \bar{b} + \bar{c} + d)$ [CO2: P30-Similar]

6] Find the minimum sum-of-products for –

(a) $f1\ (a, b, c) = m0 + m2 + m5 + m6$ (b) $f2\ (d, e, f) = \sum m\ (0, 1, 2, 4)$

(c) $f3\ (r, s, t) = rt' + r's' + r's)$ (d) $f4\ (x, y, z) = M0\ .\ M5$ [CO2: P8 or P12]

7] Design a 3-input, 1-output, minimal two-level gate combinational circuit; which has an output equal to 1 when majority of its inputs are at logic 1, and has output 0 when majority of inputs are at logic 0. [CO2]

8] Design a minimal sum and minimal product combinational gate circuit to generate the odd parity bit for an 8421 BCD code. [CO2]

9] Design (a) Binary-to-Gray Code Converter, and (b) Gray-to-Binary Code Converter [CO2]

10] A switching circuit has two control inputs (C1 and C2), two data inputs (X1 and X2), and one output (Z). The circuit performs one of the logic operations AND, OR, EQU (equivalence), or XOR (exclusive OR) on the two data inputs. The function performed depends on the control inputs:

C_1	C_2	Function Performed by Circuit
0	0	OR
0	1	XOR
1	0	AND
1	1	EQU

(i) Derive a truth table for Z

(ii) Use a Karnaugh Map to find minimum AND-OR Gate Circuit to realize Z. [CO2]

11] A logic circuit realizing the function f has four inputs a, b, c, d. The three inputs a, b, and c are the binary representation of the digits 0 through 7 with a being the most significant bit. The input d is an odd-parity bit; that is, the value of d is such that a, b, c, and d always contains an odd number of 1's. (For example, the digit 1 is represented by abc = 001 and d = 0, and the digit 3 is represented by abcd = 0111.) The function f has value 1 if the input digit is a prime number. (A number is prime if it is divisible only by itself and 1; 1 is considered to be prime, and 0 is not.)

a. Draw a Karnaugh map for f
b. Find all prime implicants of f
c. Find all minimum sum of products for f
d. Find all prime implicants of f'
e. Find all minimum product of sums for f. [CO2]

12] Using Quine-McCluskey method, simplify;

a) $f(a, b, c, d) = \sum m(3, 4, 5, 7, 10, 12, 14, 15) + \sum d(2)$
b) $f(a, b, c, d) = \sum m(1, 5, 7, 9, 11, 12, 14, 15)$
c) $f(a, b, c, d) = \sum m(0, 1, 3, 5, 6, 7, 8, 10, 14, 15)$
d) $f(a, b, c, d) = \sum m(1, 3, 4, 5, 6, 7, 10, 12, 13) + \sum d(2, 9)$
e) $f(a, b, c, d) = \sum m(9, 12, 13, 15) + \sum d(1, 4, 5, 7, 81 11, 14)$. [CO2: P27 & P28-Similar]

MODULE – 3

COMBINATIONAL LOGIC CIRCUITS

1] Explain briefly;

a) Gate fan-in
b) Propagation delay
c) Transition time
d) Hazards in Combinational logic
   i. Static-1 Hazard
   ii. Static-0 Hazard
   iii. Dynamic Hazard [CO3:, P1, P3 & P4 to P7]

2] How do you detect static hazards in Combinational logic circuits? Explain with examples.

3] Write a brief note on four-valued logic simulator. [CO3: P8 & P9]

4] Consider the logic function: $F(A, B, C, D) = \sum m(0, 2, 3, 5, 6, 7, 8, 9, 13, 15)$.

(a) Find three different minimum AND-OR circuits that implement F. Identify two hazards in each circuit. Then find an AND-OR circuit for F that has no hazards.

(b) There are two minimum OR-AND circuits for F; each has one hazard. Identify the hazard in each circuit, and then find an OR-AND circuit for F that has no hazards. [CO3: P11 & P12-Similar]

5] Consider the logic function: F (A, B, C, D) = ∑ m(0, 2, 5, 6, 7, 8, 9, 12, 13, 15).

(a) Find two different minimum AND-OR circuits which implement F. Identify two hazards in each circuit. Then find an AND-OR circuit for F that has no hazards.

(b) The minimum OR-AND circuit for F has one hazard. Identify it, and then find an OR-AND circuit for F that has no hazards. [CO3: P11 & P12-Similar]

6] Write a brief note on

a) Multiplexers
b) Three-State Buffers
c) Decoders and Encoders
d) Programmable Logic Devices
   i. Programmable Logic Array
   ii. Programmable Array Logic [CO3: P15 to P30]

7] What is multiplexer? Design 4:1 multiplexer and implement using logic gates, write the truth table and explain its working principle. [CO3: P15-Hint]

8] Implement Y (A, B, C, D) = ∑m (0, 1, 6, 7, 8, 9, 10, 11, 12, 14) using 8-to-1 multiplexer and 4-to-1 multiplexer. [CO3: P18-Similar]

9] (a) Prove that a 4:1 MUX can be realized using only 2:1 multiplexers. [CO3: P18-Hint]

(b) Use a 4-to-1 multiplexer and a minimum number of external gates to realize the function F (w, x, y, z) = ∑m (3, 4, 5, 7, 10, 14) + ∑d (1, 6, 15). [CO3: P18-Similar]

10] Implement $Z(A,B,C,D) = A'C' + A'BD' + AB'D'$ using a 4-to-1 MUX and minimum n umber of external gates, taking

a) A and B as select inputs
b) C and D as select inputs. [CO3: P18-Similar]

11] Design an 8-to-1 multiplexer using four 2-to-1 multiplexers and one 4-to-1 multiplexer. [CO3: P18-Hint]

12] Show how

(a) two 2-to-1 multiplexers (with no added gates) could be connected to form a 3-to-1 MUX. Input selection should be as follows:

If AB = 00, select I_0

If AB = 01, select I_1

If AB = 1– (B is a don't-care), select I_2 [CO3: P6 of M4-Hint]

(b) two 4-to-1 and one 2-to-1 multiplexers could be connected to form an 8-to-1 MUX with three control inputs. [CO3: P18-Similar]

ANALOG AND DIGITAL ELECTRONICS – REVIEW QUESTIONS

(c) four 2-to-1 and one 4-to-1 multiplexers could be connected to form an 8-to-1 MUX with three control inputs. [CO3: P18-Similar]

(d) to implement a 32-to-1 multiplexer using two 16-to-1 multiplexers and a 2-to-1 multiplex [CO3: P18]

13] Implement the function $R = ab'h' + bch' + eg'h + fgh$ using –

(a) 2-to-1 MUXes

(b) only tri-state buffers. [CO3: P20-Hint]

14] Implement a full adder

(a) using two 8-to-1 MUXes. Connect X, Y, and Cin to the control inputs of the MUXes and connect 1 or 0 to each data input.

(b) using two 4-to-1 MUXes and one inverter. Connect X and Y to the control inputs of the MUXes, and connect 1's, 0's, Cin, or Cin to each data input. [CO3: P18-Similar]

15] A car safety alarm considers four inputs: door closed (D), key in (K), seat pressure (S), and belt closed (B). The alarm (A) should sound if –

· The key is in and the door is not closed (or)

· The door is closed, the key is in, the driver is in the seat and the seat belt is not fastened.

(i) Construct the truth table (ii) Implement the above function using 8:1 MUX. [CO3]

16] Define decoder. Describe the working principle of a 3:8 decoder. Draw the logic diagram of 3:8 decoder with enable input. Realize the following Boolean expressions using a 3:8 decoder and multi-input OR gates: $F1(A, B, C) = \sum m(1, 3, 7)$ $F2(A, B, C) = \sum m(2, 3, 5)$. [CO3: P23-Similar]

17] Illustrate with neat diagram

a) A 3-to-8 line decoder

b) A 4-to-10 line decoder. [CO3: P22-Hint]

18] Realize a full adder using a 3-to-8 line decoder and

(a) two OR gates, (b) two NOR gates. [CO3: P23-Hint]

19] Derive the logic equations for a 4-to-2 priority encoder. [CO3: P25-Hint]

20] Implement the following equations using PLA – [CO3: P27 & P28]

$$X = AB'D + AC' + BC + C'D'$$

$$Y = A'C' + AC + C'D'$$

$$Z = CD + A'C' + AB'D$$

21] Implement a full subtractor using a PAL. [CO3: P30-Similar]

By: MAHESH PRASANNA K.,

DEPT. OF CSE, VCET.

_____*********_____

****

MODULE – 4

VHDL, LATCHES AND FLIP-FLOPS

1] Write VHDL code for – [CO3: P5 & P6-Hint]

a) 2:1 MUX using VHDL statement and conditional assignment statement

Answer:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
    entity MUX2to1 is
            port (  d0: in STD_LOGIC;
                    d1: in STD_LOGIC;
                    A: in STD_LOGIC;
                    y: out STD_LOGIC);
        end MUX2to1
        architecture behavioral of MUX2to1 is
            begin
          y <= (not A and d0) or (A and d1);  // y <= d0 when A = '0' else d1;
        end behavioral
```

b) 4-to-1 MUX using VHDL statement and conditional assignment statement.

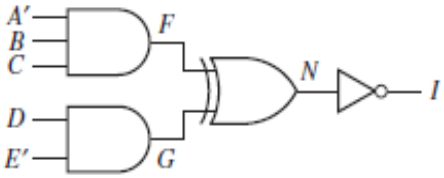
Answer:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
        entity MUX4to1 is
            port (  d0, d1, d2, d3: in STD_LOGIC;
                    A, B: in STD_LOGIC;
                    y: out STD_LOGIC);
        end MUX4to1
        architecture multiplexer of MUX4to1 is
            begin
                y <= (not A and not B and I0) or (not A and B and I1) or
                     (A and not B and I2) or (A and B and I3);
        end multiplexer
    // architecture multiplexer of MUX4to1 is     // architecture multiplexer of MUX4to1 is
        // sel <= A&B;                                  // begin
            // begin                                      // y <= d0 when A = '0' and B = '0'
                // with sel select                       // else d1 when A = '0' and B = '1'
                    // y <= d0 when "00",                // else d2 when A ='1' and B = '0'
                       // d1 when "01",                  // else d3;
                       // d2 when "10",               // end multiplexer
                       // d3 when "11";
        // end multiplexer
```

2] Write VHDL statements that represent the following circuit: [CO3: P7-Hint]

a) Write a statement for each gate.

b) *Write one statement for the whole circuit.*



Answer:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
        entity FIG2A is
                port (  A,B, C: in STD_LOGIC;
                        F: out STD_LOGIC);
        end FIG2A
        architecture gates of FIG2A is
                begin
                    F <= ((not A) and B and C);          // G <= (D and (not E));
                    // N <= (F xor G);                   // I <= (not N);
        end gates
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
        entity FIG2B is
                port (  A, B, C, D, E: in STD_LOGIC;
                        I: out STD_LOGIC);
        end FIG2B
        architecture gates of FIG2B is
        signal F, G, N: bit;
            begin                           // I = (A'BC ⊕ DE')'
                    I <= (not ((not A) and B and C) xor (D and (not E)));
end gates
```

3] *Draw the circuit represented by the following VHDL statements:* [CO3]

F <= E ***and*** *I;*

I <= G ***or*** *H;*

G <= A ***and*** *B;*

H <= ***not*** *C* ***and*** *D;*

Answer *(Hint)****:***

$F = EI$

$= E (G + H)$

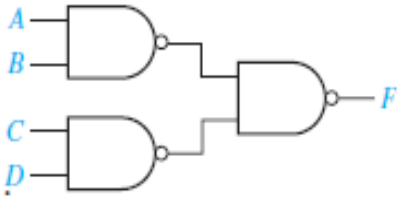
$= E [(AB) + H]$

$= E (AB + C'D)$

4] *Write* [CO3: P7-Hint]

a) *a complete VHDL module for a two-input NAND gate with 4-ns delay.*

b) *Write a complete VHDL module for the following circuit that uses the NAND gate module of Part (a) as a component.*



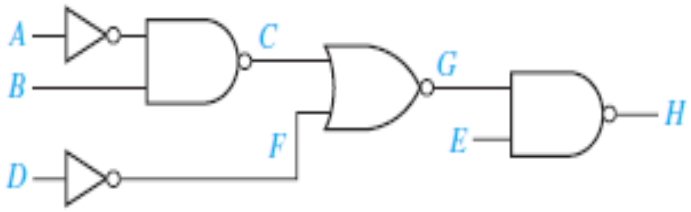
Answer:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
        entity NAND2GATE is
                port (   X, Y: in STD_LOGIC;
                         Z: out STD_LOGIC);
        end NAND2GATE
        architecture gates of NAND2GATE is
                begin
                    Z <= X nand Y after 4ns;
        end gates
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
        entity FIG4B is
                port (   A, B, C, D: in STD_LOGIC;
                         F: out STD_LOGIC);
        end FIG4B
        architecture gates of FIG4B is
        component NAND2GATE
                port (   X, Y: in STD_LOGIC;
                         Z: out STD_LOGIC);
        end component
        signal P, Q: bit;          // P – Output of 1st NAND gate; Q – Output of 2nd NAND gate.
                begin
                    NG1: NAND2GATE port map (A, B, P);
                    NG2: NAND2GATE port map (C, D, Q);
                    NG3: NAND2GATE port map (P, Q, F);
        end gates
```

5] Draw the circuit represented by the following VHDL statements: [CO3]

*T1 <= **not** A **and not** B **and** I0;*	***Answer** (Hint):*	*T1 = A'B'I0*
*T2 <= **not** A **and** B **and** I1;*		*T2 = A'BI1*
*T3 <= A **and not** B **and** I2;*		*T3 = AB'I2*
*T4 <= A **and** B **and** I3;*		*T4 = ABI3*
*F <= T1 **or** T2 **or** T3 **or** T4;*		*F = T1 + T2 + T3 + T4*

6] Write a single concurrent VHDL statement to represent the following circuit: [CO3: P7-Hint]



Answer:

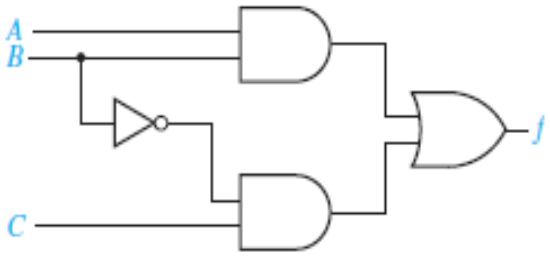
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
        entity FIG6 is
                port (   A, B, D, E: in STD_LOGIC;
                         H: out STD_LOGIC);
        end FIG6
        architecture gates of FIG6 is
            begin                          // H = (GE)' = ((C + F)'E)' = (((A'B)'+D')'E)'
                    H <= (((not A) nand B) nor (not D)) nand E;
end gates
```

7] Write the VHDL code for [CO3: P7, P8-Hint]

a) *A full adder*

b) *A full subtractor.*

8] In the following circuit, all gates, including the inverter, have an inertial delay of 10 ns. [CO3]



a) *Write VHDL code that gives a dataflow description of the circuit. All delays should be inertial delays.*

b) *Using the Direct VHDL simulator simulate the circuit. (Use a View Interval of 100 ns.) Initially set A = 1, B = 1 and C = 1, then run the simulator for 40 ns. Change B to 0, and run the simulator for 40 ns. Record the waveform [Even you can draw waveform for given conditions]*

c) *Change the VHDL code of Part (a) so that the inverter has a delay of 5 ns.*

d) *Repeat Part (b)*

e) *Change the VHDL code of Part (c) so that the output OR gate has a transport delay rather than an inertial delay*

f) *Repeat Part (b)*

g) *Explain any differences between the waveforms for Parts (b), (d), and (f).*

Answer:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
        entity FIG8A is
                port (   A,B, C: in STD_LOGIC;
                         f: out STD_LOGIC);
        end FIG8A
        architecture dataflow of FIG8A is
        signal p, q, r: bit;        // p – output of upper AND gate, q – output of lower AND gate,
                begin                       // r – output of NOT gate.
                    p <=A and B after 10ns;
                    r <= nor B after 10ns;
                    q <= r and C after 10ns;
                    f <= p or q after 10ns;
        end dataflow
```

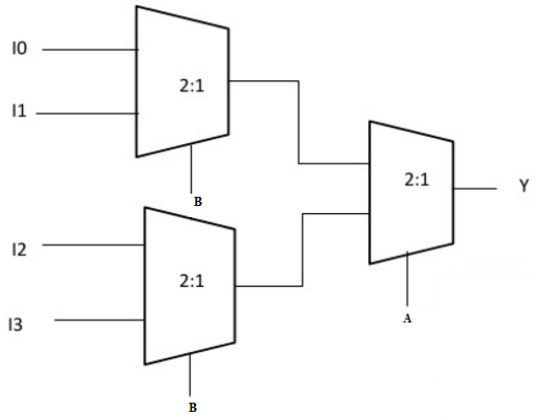
9] Design a 4-to-1 MUX using only three 2-to-1 MUXes. Write an entity-architecture pair to implement a 2-to-1 MUX. Then write an entity-architecture pair to implement a 4-to-1 MUX using three instances of your 2-to-1 MUX. *[CO3: P7 to P9-Hint]*

Answer:

Expression for 2:1 MUX – $F = \bar{A}I_0 + AI_1$

Expression for 4:1 MUX – $Y = \bar{A}\bar{B}I_0 + \bar{A}BI_1 + A\bar{B}I_2 + ABI_3$

$= \bar{A}(\bar{B}I_0 + BI_1) + A(\bar{B}I_2 + BI_3)$

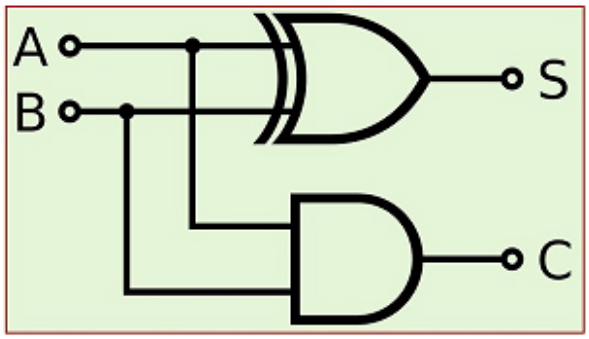



```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
        entity MUX2to1 is
                port (   A, I0, I1: in STD_LOGIC;
                         F: out STD_LOGIC);
        end MUX2to1
        architecture multiplexer of MUX2to1 is
                begin
                    F <= (not A and I0) or (A and I1);
        end multiplexer
```

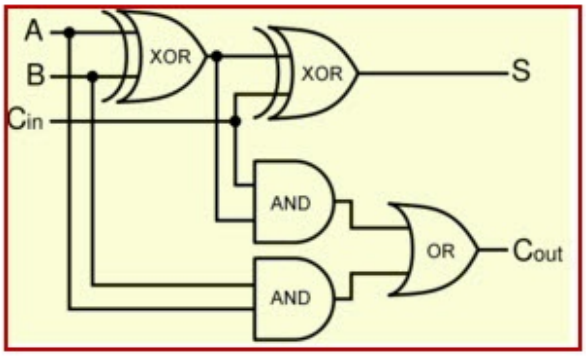
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
        entity MUX4to1 is
                port (  A, B, I0, I1, I2, I3: in STD_LOGIC;
                        Y: out STD_LOGIC);
        end MUX4to1
        architecture multiplexers of MUX4to1 is
        component MUX2to1
                port (  A, I0, I1: in STD_LOGIC;
                        F: out STD_LOGIC);
        end component
        signal P, Q: bit;          // P – Output of 1st 2:1 MUX & Q – Output of 2nd 2:1 MUX.
                begin
                    M1: MUX2to1 port map (B, I0, I1, P);
                    M2: MUX2to1 port map (B, I2, I3, Q);
                    M3: MUX2to1 port map (A, P, Q, Y);
        end multiplexers
```

10] Write an entity-architecture pair to implement the half adder. Then write an entity-architecture pair to implement a full adder using two instances of your half adder and an OR gate. [CO3]

Answer:

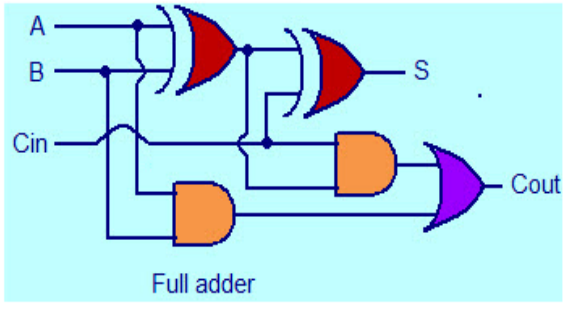


$S = A \oplus B \qquad C = AB$



$S = A \oplus B \oplus C_{in} \qquad Cout = AB + BC + CA$

$S = A \oplus B \oplus Cin$



$$Cout = (AB) + Cin.(A \oplus B)$$
$$= AB + Cin(A'B + AB')$$
$$= AB + CinA'B + CinAB'$$
$$= AB + AB + CinA'B + CinAB' \ (\because X + X = X)$$
$$= A(B + B'Cin) + B(A + A'Cin)$$
$$= A(B + Cin) + B(A + Cin) \ (\because X + X'Y = X + Y)$$
$$= AB + BCin + CinA$$

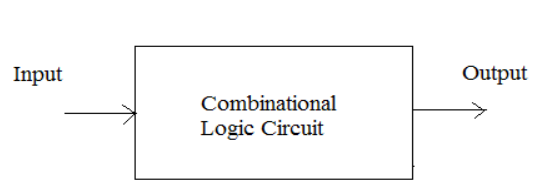
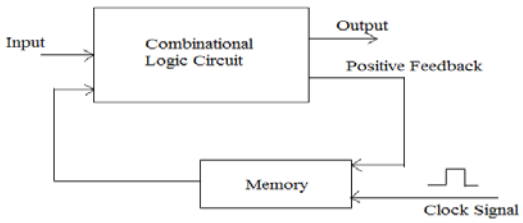
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
```

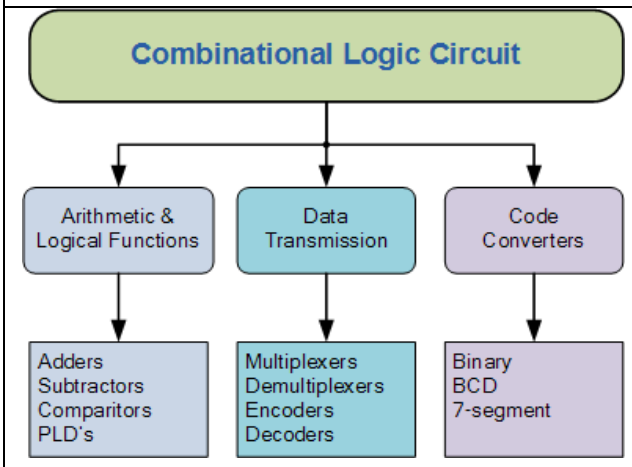
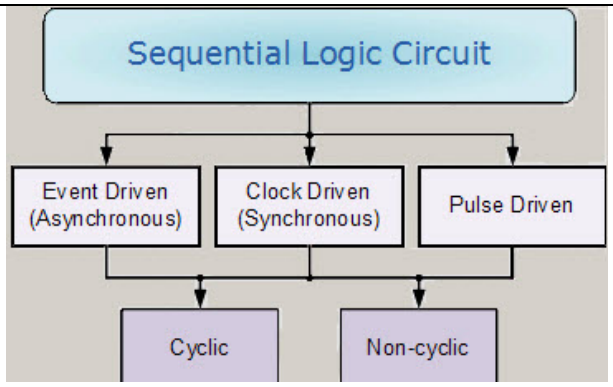
```
entity HA is
        port (  A, B: in STD_LOGIC;
                S, C: out STD_LOGIC);
end HA
architecture gates of HA is
        begin
            S <= A xor B;
            C <= AB;
end gates
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
        entity FA is
                port (  A, B, Cin: in STD_LOGIC;
                        Sum, Cout: out STD_LOGIC);
        end FA
        architecture gates of FA is
        component HA
                port (  A, B: in STD_LOGIC;
                        S, C: out STD_LOGIC);
        end component
        signal s1, c1, c2: bit;     // s1 – Output of 1st XOR gate, c1 – Output of 1st AND gate,
                begin                       // c2 – output of 2nd AND gate.
                    HA1: HA port map (A, B, s1, c1);
                    HA2: HA port map (s1, Cin, Sum, c2);
                    Cout <= c1 or c2;
        end gates
```

10] Write a VHDL module for a 4-bit adder. [CO3: P8 & P9-Similar]

11] Differentiate combinational and sequential circuits. [CO3]

Answer:

Combinational Circuits	Sequential Circuits
	
Output is a function of the present inputs (Time Independent Logic); it simply outputs the input according to the logic designed. *Example: Adder [1+0=1; Dependency only on present inputs i.e., 1 and 0].*	*Output is a function of clock, present inputs and the previous states of the system; it involves feedback from output to input that is stored in the memory for the next operation.* *Example: Counter [Previous O/P +1=Current O/P; Dependency on present input as well as*

	previous state].
Do not have the ability to store data.	*Have memory to store the present states that is sent as control input (enable) for the next operation.*
Logic gates are the elementary building blocks; and hence, used mainly for Arithmetic and Boolean operations.	*Flip flops (binary storage device) are the elementary building unit; and hence, used for storing data (and hence used in RAM).*
Independent of clock; and hence does not require triggering to operate.	*Clocked (Triggered for operation with electronic pulses).*
	

12] Differentiate Latches and Flip-Flops. [CO3]

Answer:

Latches	***Flip-Flops***
Latch is transparent – because input is directly connected to output when enable is high. It means Latch is sensitive to pulse duration – level triggered (also called soft barrier).	*Flip-flop is a pair of latches (master and slave flop). Flip-flop is sensitive to pulse transition. The signal only propagates through on the rising/falling edge – edge triggered (also called hard barrier).*
Latches are the building blocks of sequential circuits and Latches can be built from logic gates.	*Flip-Flops are also building blocks of sequential circuits; but Flip-Flops are built from Latches.*
Latch is sensitive to the duration of the pulse and can send or receive data only when the switch is on.	*Flip-Flop is sensitive to a signal change. They can transfer data only at the signal instant and the data cannot be changed until next signal change.*
Latches are asynchronous devices; and require less power.	*Flip-Flops are synchronous devices; and require more power.*
Latch based design is noisy.	*Flip-Flop based design is robust.*

13] Write brief note on [CO3]

a) Set-Reset Latch

b) Applications of S-R Latch

c) *Gated Latch*

d) *Edge Triggered D Flip-Flop*

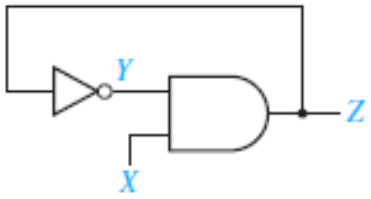
e) *S-R Flip-Flop*

f) *JK Flip-Flop*

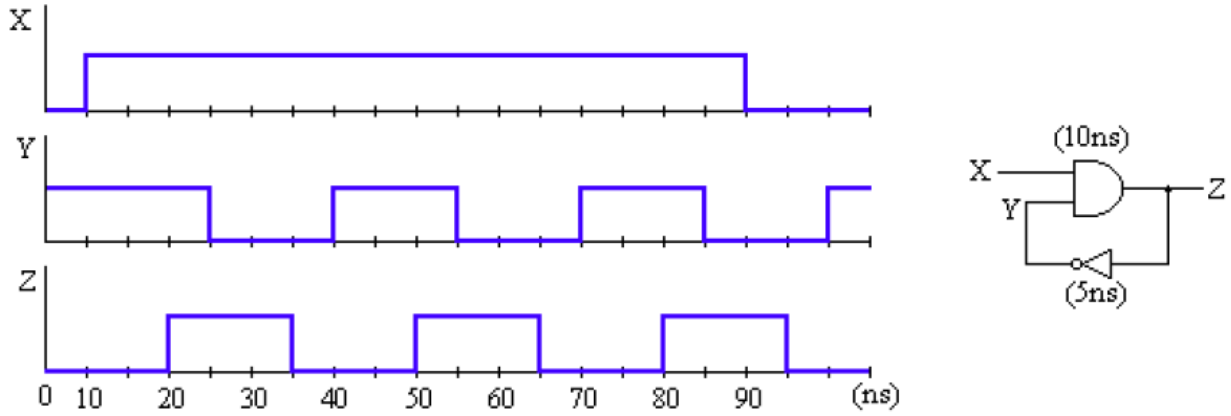
g) *T Flip-Flop*

h) *Asynchronous Sequential Circuits (Self Study Topic).*

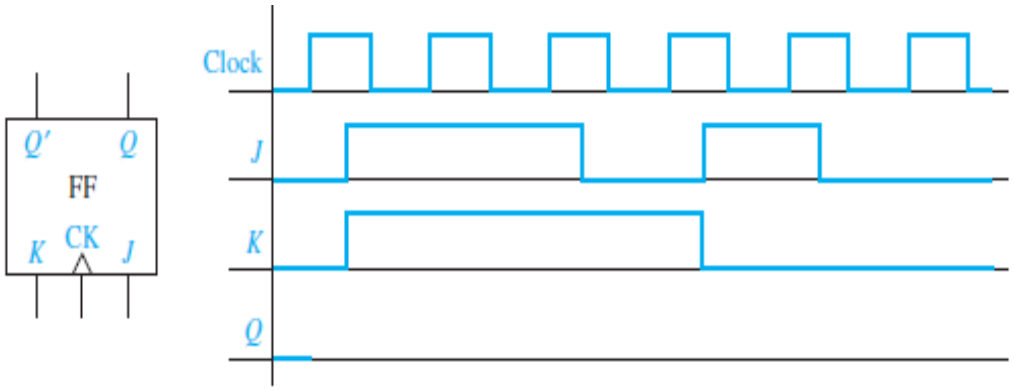
14] Assume that the inverter in the given circuit has a propagation delay of 5 ns and the AND gate has a propagation delay of 10 ns. Draw a timing diagram for the circuit showing X, Y, and Z. Assume that X is initially 0, Y is initially 1, after 10 ns X becomes 1 for 80 ns, and then X is 0 again. [CO3]



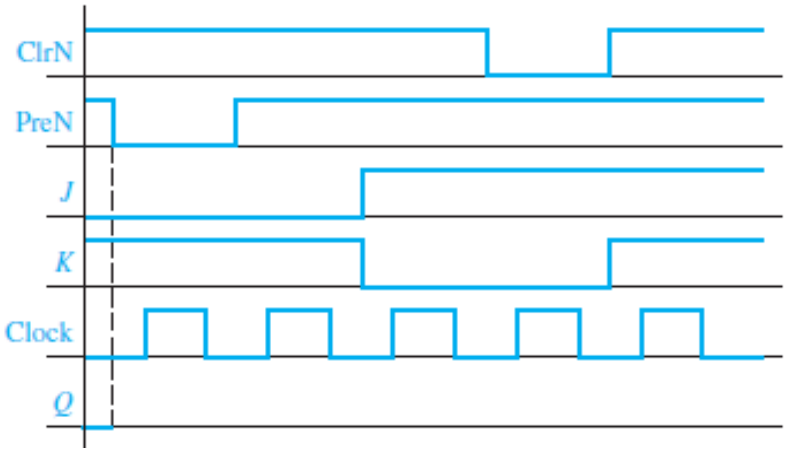
Answer:



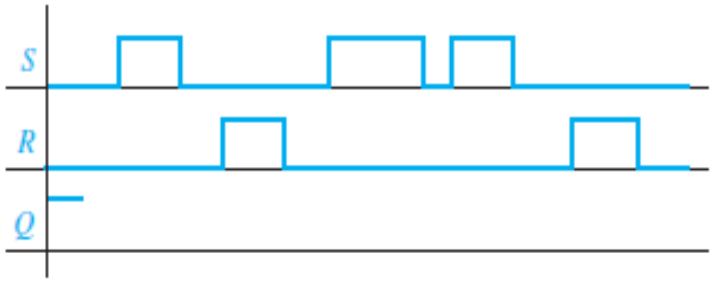
15] Complete the following timing diagram for the flip-flop: [CO3]



16] Complete the following timing diagram for a J-K flip-flop with a falling-edge trigger and asynchronous ClrN and PreN inputs. [CO3]



17] Complete the following timing diagram for an S-R latch. Assume Q begins at 1. [CO3]



18] Convert by adding external gates: (a) a D flip-flop to a J-K flip-flop; (b) a T flip-flop to a D flip-flop; (c) a T flip-flop to a D flip-flop with clock enable. [CO3]

Answer:

(a) a D flip-flop to a J-K flip-flop

Truth Table of JK Flip-flop

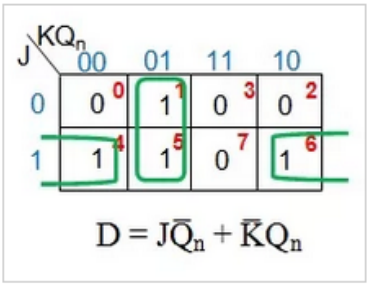
Inputs		Outputs	
		Present State	Next State
J	K	Q_n	Q_{n+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

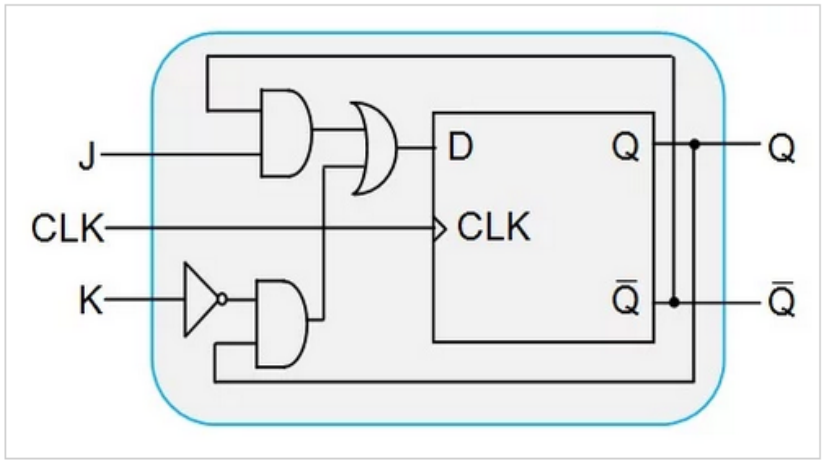
JK Inputs		Outputs		D Input
		Present State	Next State	
J	K	Q_n	Q_{n+1}	D
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0

Excitation Table of D Flip-flop

Outputs		Input
Present State	Next State	
Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

D to JK Conversion Table





(b) a T flip-flop to a D flip-flop

Truth Table of D Flip-flop

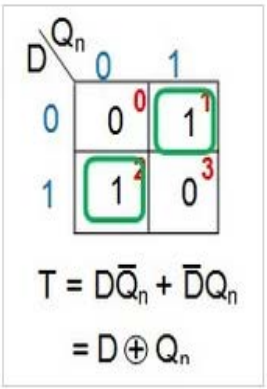
Input	Outputs	
	Present State	Next State
D	Q_n	Q_{n+1}
0	0	0
0	1	0
1	0	1
1	1	1

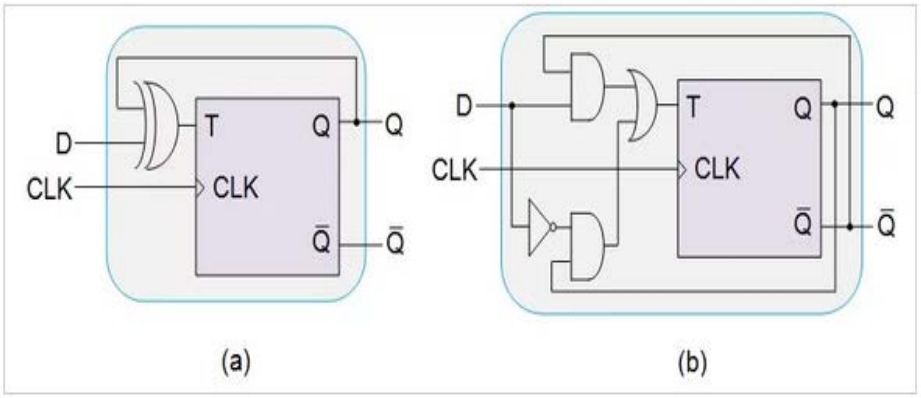
D Input	Outputs		T Input
	Present State	Next State	
D	Q_n	Q_{n+1}	T
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Excitation Table of T Flip-flop

Outputs		Input
Present State	Next State	
Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

T to D Conversion Table





By: MAHESH PRASANNA K.,

DEPT. OF CSE, VCET.

_____*********_____

****

MODULE – 5

REGISTERS AND COUNTERS

1] Differentiate Registers & Counters. [CO4: P1-Hint]

2] Explain with neat diagram, a 4-bit D flip-Flop Register with Data, Load, Clear, and Clock inputs. [CO4: P1]

3] With neat diagram, explain how data can be transferred from the output of one of two registers into a third register using tri-state buffers. [CO4: P2]

4] Describe the working of a parallel adder with accumulator. [CO4: P4]

5] What is a Shift Register? Explain a 4-bit right shift SISO register with neat timing diagram, when the shift register initially contains 0101 and the serial input sequence is 1, 1, 0, 1. [CO4: P6]

6] Explain a 4-bit PIPO shift register with a neat diagram. [CO4: P8]

7] Explain the working of a 3-bit Johnson counter with a neat sketch. [CO4: P9, P10]

8] What is a counter? Design a 3-bit straight binary sequence counter using – [CO4: P12 to 14]

(a) T flip-flop (b) D flip-flop (c) S-R flip-flop (d) J-K flip-flop.

9] With a neat sketch and relevant expressions, explain a 3-bit binary up-down counter. [CO4: P15, P16]

10] How a 2-to-1 multiplexer can be used for construction of a 3-bit loadable counter? Explain with diagram, truth-table, and relevant expressions. [CO4: P16, P17]

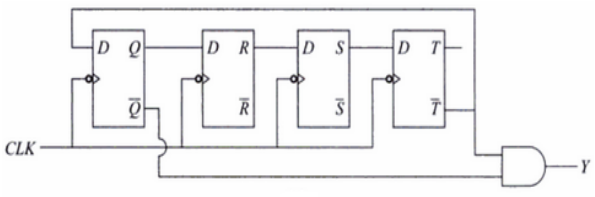
11] Design a 3-bit binary counter to generate the sequence 0, 4, 7, 2, 3, 0, . . . using –

(a) T flip-flop (b) D flip-flop (c) S-R flip-flop (d) J-K flip-flop. [CO4: P18 to P23]

12] Construct a 4-bit Johnson counter using – [CO4-P9 & P10 – Hint]

(a) D flip-flops (b) J-K flip-flops.

Answer: *(a) D flip-flops*



Clock	Serial in = $\bar{T}$	Q	R	S	T	Y = $\bar{Q}\bar{T}$
0	1	0	0	0	0	1
1	1	1	0	0	0	0
2	1	1	1	0	0	0
3	1	1	1	1	0	0
4	0	1	1	1	1	0
5	0	0	1	1	1	0
6	0	0	0	1	1	0
7	0	0	0	0	1	0
8	1	0	0	0	0	1
9	1	1	0	0	0	0 repeats

ANALOG AND DIGITAL ELECTRONICS – REVIEW QUESTIONS

13] Design a 3-bit counter which counts in the sequence: [CO4-P18 to P23 – Similar]

001, 011, 010, 110, 111, 101, 100, (repeat) 001, . . .

(a) Use D flip-flops (b) Use T flip-flops

(c) Use S-R flip-flops (d) Use J-K flip-flops.

14] Design a 3-bit counter which counts in the sequence: [CO4-P18 to P23 – Similar]

001, 100, 101, 111, 110, 010, 011, 001, . . .

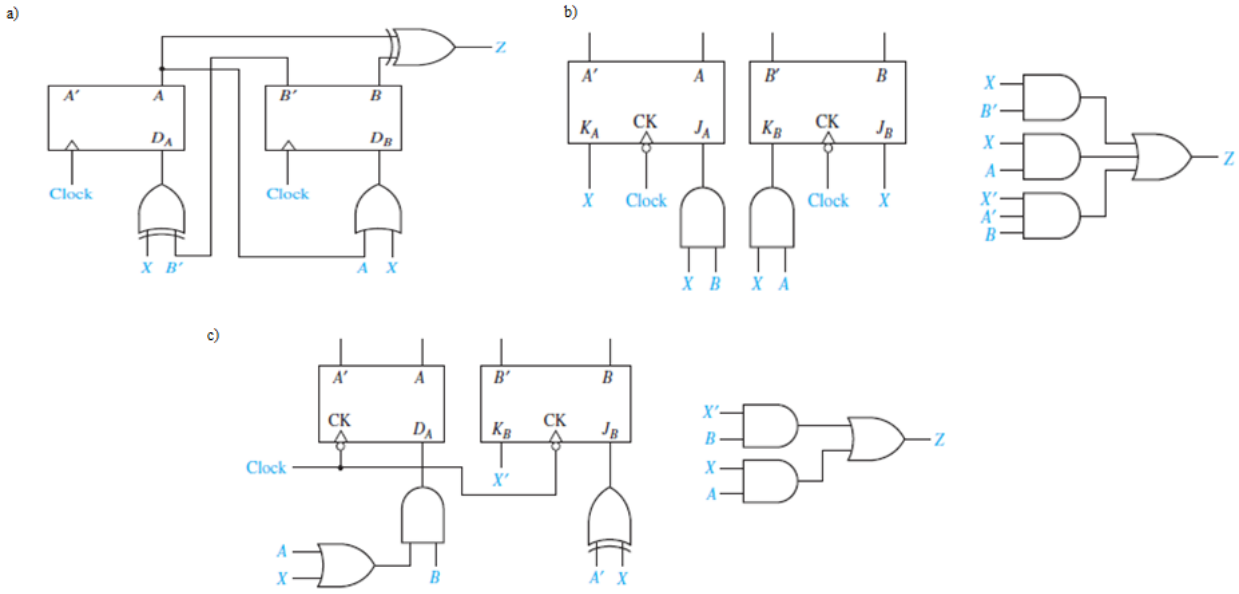
(a) Use D flip-flops (b) Use J-K flip-flops

(c) Use T flip-flops (d) Use S-R flip-flops

(e) What will happen if the counter of (a) is started in state 000?

15] What is mean by parity? Define odd parity and give examples. Design an odd parity checker for serial data input with state graph, state table, and relevant waveforms. [CO4: P24, P25 & P26]

16] Derive the state table for the circuits of the following figures: [CO4: P27 onwards]



17] Construct a state graph for the serial adder. [CO4: P31 & P32]

18] Construct a state graph for the shift register shown. (X is the input, and Z is the output). Is this a Mealy or Moore machine? [CO4: P27 onwards - Similar]

Fig. Q18

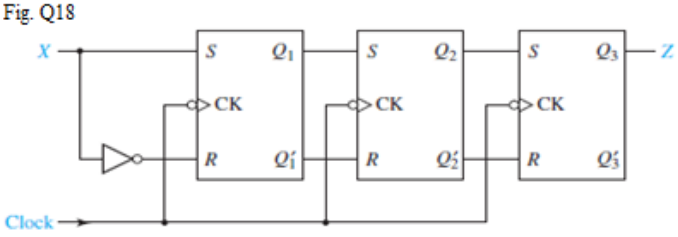


Fig. Q19

ABC	$A^+B^+C^+$ X = 0	X = 1
000	011	010
001	000	100
010	100	100
011	010	000
100	100	001

19] Below is a state transition table with the outputs missing. The output should be Z = X'B' + XB.

(a) Is this a Mealy machine or Moore machine? [CO4: P27 onwards - Similar]

(b) Fill in the outputs on the state transition table.

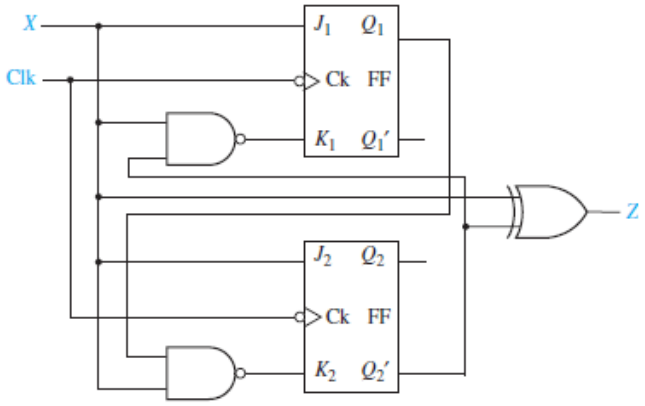
(c) Give the state graph.

(d) For an input sequence of X = 10101, give a timing diagram for the clock, X, A, B, C, and Z. State changes occur on the rising clock edge. What is the correct output sequence for Z? Change X between rising and falling clock edges so that we can see false outputs, and indicate any false outputs on the diagram.

20] (a) Construct a state table and graph for the circuit shown. *[CO4: P27 onwards - Similar]*

(b) Construct a timing chart for the circuit for an input sequence X = 10111. (Assume that initially Q_1 = Q_2 = 0 and that X changes midway between the rising and falling clock edges.)

(c) List the output values produced by the input sequence.

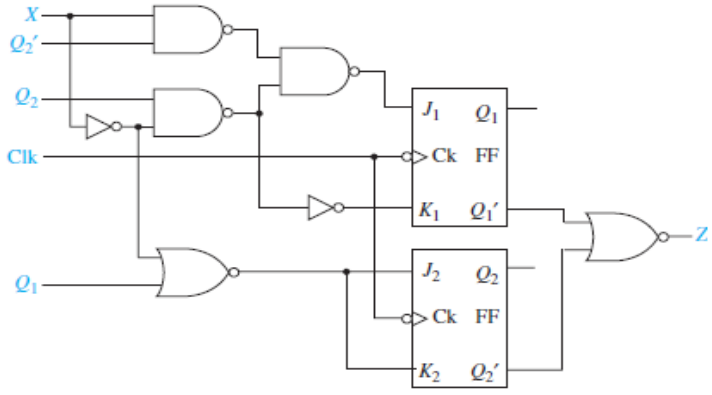


21] Consider the circuit shown. *[CO4: P27 onwards - Similar]*

(a) Construct a state table and graph for the following circuit. Is the circuit a Mealy or Moore circuit? Does the circuit have any unused states? Assume 00 is the initial state.

(b) Draw a timing diagram for the input sequence X = 01100.

(c) What is the output sequence for the input sequence?



By: MAHESH PRASANNA K.,

DEPT. OF CSE, VCET.

_____*********_____

****

MODULE – 1

ANALOG ELECTRONIC CIRCUITS

1] What are Optoelectronic Devices? Explain the construction, working principle, and applications of Photodiodes. *[CO1: P1, to P]*

2] What are Optoelectronic Devices? Explain the construction, working principle, and applications of Light Emitting Diodes. *[CO1: P1, & P4 to P6]*

3] What are Optoelectronic Devices? Explain the construction, working principle, and applications of Photocouplers. *[CO1: P1, P6, P7]*

4] What is Bipolar Junction Transistor? Show base bias (or fixed bias) circuit and explain biasing equations. *[CO1: P7, P8]*

5] What is Bipolar Junction Transistor? Show collector-to-base bias circuit and explain biasing equations. *[CO1: P7, P10]*

6] What is Bipolar Junction Transistor? Show voltage divider bias (or emitter current bias) circuit and explain biasing equations. *[CO1: P7, P11]*

7] How collector-to-base bias circuit provides more stability? Explain. *[CO1: P10]*

8] What is a multivibrator? Explain the working of monostable multivibrator with a neat sketch and drive the equation for 'on' time of the output voltage. *[CO1: P13, P14]*

9] What is a multivibrator? Explain the working of astable multivibrator with a neat sketch and drive the equation for Duty Cycle. *[CO1: P16, P17, P18]*

10] What is an Operational Amplifier? List the characteristics of an ideal Op-Amp & practical Op-Amp. *[CO1: P20]*

11] What is an Operational Amplifier? Differentiate ideal Op-Amp & practical Op-Amp. *[CO1: P20]*

12] Explain Op-Amp Peak Detector circuit with waveform. *[CO1: P21]*

13] With neat diagram and waveforms, explain Non-inverting Schmitt trigger and derive the expression for hysteresis. *[CO1: P22]*

14] With neat diagram and waveforms, explain Inverting Schmitt trigger and derive the expression for hysteresis. *[CO1: P23]*

15] What is a filter? Give the broader classifications of filters. Also, write the significance of active filters? *[CO1: P27]*

16] What are active and passive filters? Give the differences between them. *[CO1: P27, P28]*

17] Write the frequency response curve of – (a) low pass filter, (b) high pass filter, (c) band pass filter, and (d) band reject filter. *[CO1: P28]*

18] With neat sketch and waveforms, explain the working of first order active low-pass filter. *[CO1: P29]*

19] With neat sketch and waveforms, explain the working of second order active low-pass filter.

[CO1: P30]

20] With neat sketch and waveforms, explain the working of first order active high-pass filter.

[CO1: P32]

21] With neat sketch and waveforms, explain the working of second order active high-pass filter.

[CO1: P33]

22] What is band-pass filter? Explain wide band-pass filter with neat sketch and waveforms.

[CO1: P34, P35]

23] What is band-pass filter? Explain narrow band-pass filter with neat sketch and waveforms.

[CO1: P34, P35, P36]

24] What is band-reject (band-stop/ notch) filter? Explain wide band-reject filter with neat sketch and waveforms. [CO1: P36, P37]

25] What is band-reject (band-stop/ notch) filter? Explain narrow band-reject filter with neat sketch and waveforms. [CO1: P36, P38]

26] With a neat sketch explain the working of a non-linear amplifier. Also, by using relevant derivations, prove that output voltage is a logarithmic function of input function. [CO1: P39, P40]

27] Explain the working of Op-Amp relaxation oscillator with a neat diagram. [CO1: P40, P41]

28] Explain voltage-to-current converter with (a) floating load, and (b) grounded load. [CO1: P41, P42]

29] Explain current-to-voltage converter. [CO1: P42]

30] Write a brief note on voltage regulation. [CO1: P43]

31] List and explain the factors affecting the load voltage in a power supply. [CO1: P44]

32] List and explain the performance parameters of a power supply. [CO1: P45]

33] Write a brief note on – (a) Three terminal regulators, and (b) Adjustable voltage regulators.

[CO1: P46, P47, P48]

34] With a neat block diagram, explain a typical A/D and D/A converter. [CO1: P48, P49]

35] Give the symbolic representation of n-bit DAC, and give the expression for analog output voltage. Also, explain the performance parameters of DAC with example. [CO1: P49, P50, P51]

36] Explain binary weighted resistor DAC with a neat diagram. [CO1: P51]

37] Explain R-2R ladder type DAC with a neat diagram. [CO1: P52]

38] Give the functional diagram of ADC and explain a 2-bit flash ADC. [CO1: P54. P55]

39] Explain a 3-bit ADC with neat sketch and relevant truth table. [CO1: P55, P56]

40] With neat diagram, explain successive approximation type ADC. [CO1: P56, P57]

By: MAHESH PRASANNA K.,

DEPT. OF CSE, VCET.

_____*********_____

****