DIGITAL INTEGRATED CIRCUITS ANALYSIS

ELECTRONICS & COMMUNICATION ENGINEERING

unit - 5

HAND NOTES

BY:

P.RAJESH M.TECH.,

ASSISTANT PROFFESOR

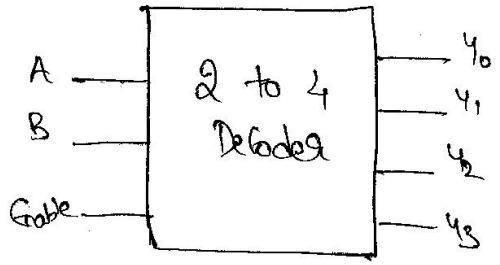
EMAIL: rajesh.crit@gmail.com

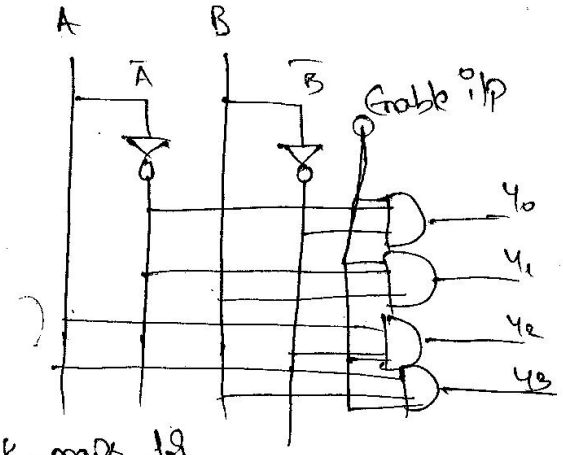
ph: 9989786119

9985786099

CRIT COLLEGE OF ENGG & TECHNOLOGY

ANANTAPURAMU





Truth table:-

Gable	B	A	y_3	y_2	y_1	y_0
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

K-maps for

$y_0 = \bar{A}\bar{B}$

$y_1 = \bar{A}B$

$y_2 = A\bar{B}$

$y_3 = AB$

VHDL Programing:-

i) Data flow Style:- (Selected signal assignment)

```
library ieee;
use ieee.std_logic_1164.ALL;
Entity decoder is
Port (Gable, B, A : in std_logic;
      y : out std_logic_vector (3 downto 0));
End decoder;
Architecture dataflow of decoder is
 Signal yn: std_logic_vector (0 to 7);
 begin
  With A,B,C Select
   yn <= "0001" when "00";
   yn <= "0010" when "01";
   yn <= "0100" when "10";
   yn <= "1000" when "11";
   yn <= "0000" when others;
 End dataflow.
```

If Statement:- (behavioral)

```
library ieee;

Entity decoder is
Port ( n1 : in std_logic;
       y : out std_logic_vector(3 downto 0));
End decoder is
Architecture behavioral of decoder is
begin
Process (n1)
begin
if n1 = 00 then y <= "0001";
else if n1 = 01 then y <= "0010";
else if n1 = 10 then y <= "0100";
else if n1 = 11 then y <= "1000";
else y <= "0000";
End if;
End Process;
End behavioral;
```

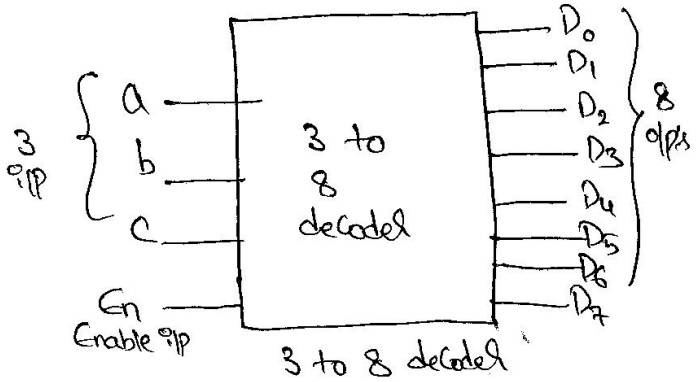
Unit - 7

Combinational Logic Design :-

1) Binary Decoder :-

decoder which takes n-bit binary i/p and 2^n o/p from which only one o/p will be activated at time based i/p lines as known as binary decoder.

→ **3 to 8 decoder :-** 3 i/p and 8 o/p's



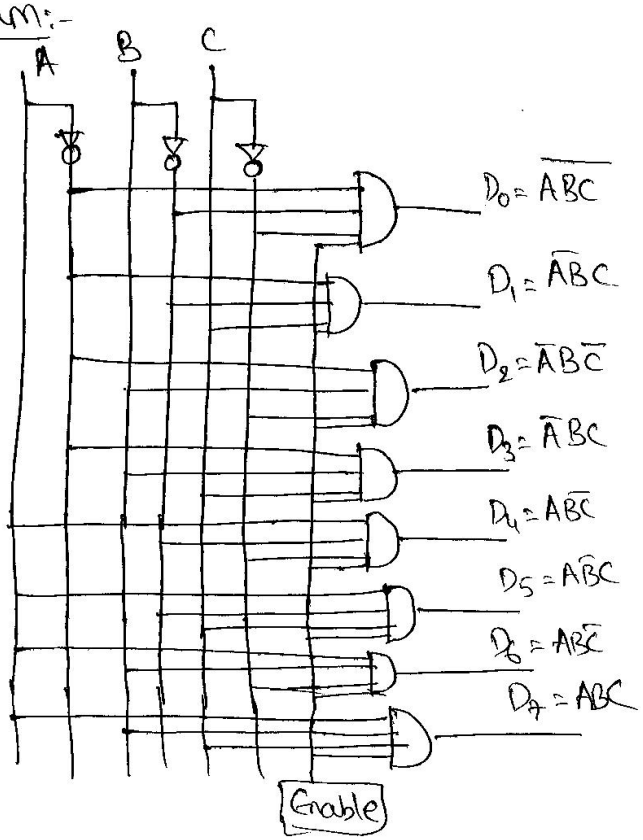
K-map relations

$D_0 = \bar{A}\bar{B}\bar{C}$
$D_1 = \bar{A}\bar{B}C$
$D_2 = \bar{A}B\bar{C}$
$D_3 = \bar{A}BC$
$D_4 = A\bar{B}\bar{C}$
$D_5 = \bar{A}\bar{B}C$
$D_6 = AB\bar{C}$
$D_7 = ABC$

Truth table :-

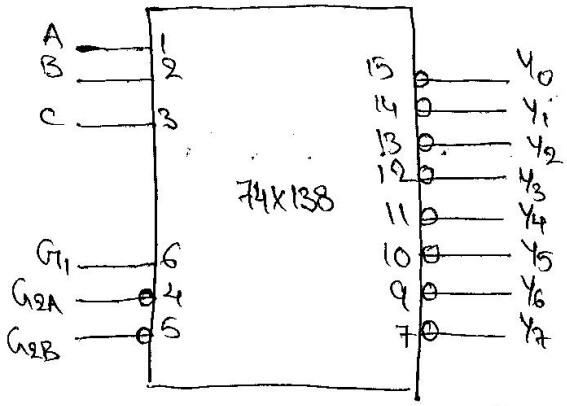
				o/p's							
En	a	b	c	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	x	x	x	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	0
1	0	1	0	0	0	1	0	0	0	0	0
1	0	1	1	0	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	1	0	0	0
1	1	0	1	0	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	0	0	1	0
1	1	1	1	0	0	0	0	0	0	0	1

Logic diagram:-



*) 74x138 Decoder :-

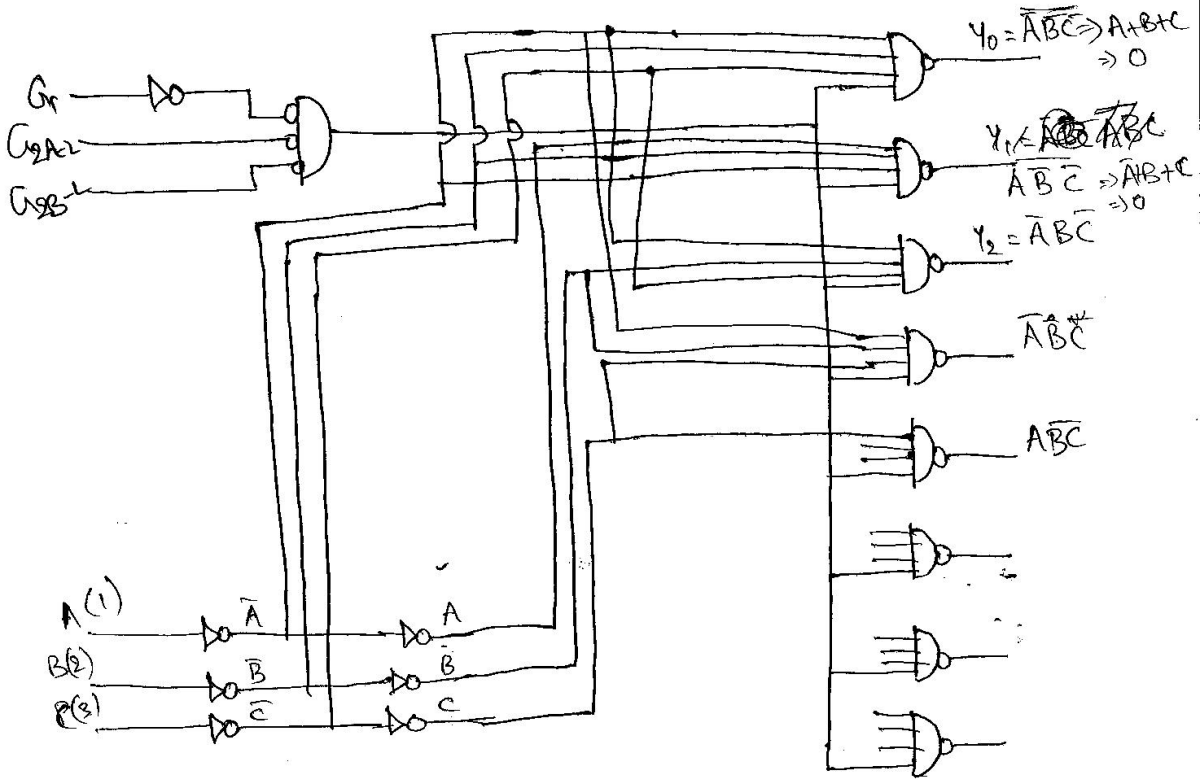
It is a 3 to 8 decoder that available for commercial purpose.



It has ABC are three binary i/p. Two active low Enable i/ps ($\overline{G_{2A}}$, $\overline{G_{2B}}$) and one active high G_1 i/p.

It has eight decoded o/p (Y_0 to Y_7).

I/p						O/p							
G_1	$\overline{G_{2A-L}}$	$\overline{G_{2B-L}}$	C	B	A	$\overline{Y_7}$	$\overline{Y_6}$	$\overline{Y_5}$	$\overline{Y_4}$	$\overline{Y_3}$	$\overline{Y_2}$	$\overline{Y_1}$	$\overline{Y_0}$
0	X	X	X	X	X	1	1	1	1	1	1	1	1
X	1	X	X	X	X	1	1	1	1	1	1	1	1
X	X	1	X	X	X	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	1	1	1	0
1	0	0	0	0	1	1	1	1	1	1	1	0	1
1	0	0	0	1	0	1	1	1	1	1	0	1	1
1	0	0	0	1	1	1	1	1	1	0	1	1	1
1	0	0	1	0	0	1	1	1	0	1	1	1	1
1	0	0	1	0	1	1	1	0	1	1	1	1	1
1	0	0	1	1	0	1	0	1	1	1	1	1	1
1	0	0	1	1	1	0	1	1	1	1	1	1	1



*) write a VHDL Program for 74x138 (3 to 8 dec) using Case Statement

Sol:- Logic diagram from '2' question.

Program:-

```
library IEEE;
use IEEE.Std_Logic_1164.ALL;
entity dec 3to 8 is
Port (A : in Std_Logic_vector (0 to 7);
      y : out Std_Logic_vector (7 down to 0));
end dec 3to 8;
Architecture behavioural of dec 3to8 is
begin
  Process (A)
  begin
    Case A is
when 0 => y <= "00000001";
when 1 => y <= "00000010";
when 2 => y <= "00000100";
when 3 => y <= "00001000";
when 4 => y <= "00010000";
when 5 => y <= "00100000";
when 6 => y <= "01000000";
when 7 => y <= "10000000";
  end Case;
  end Process;
  end behavioral;
```

*) Write a VHDL Program for 74x138 (3 to 8 dec) using Structural design method.

Sol) Logic diagram

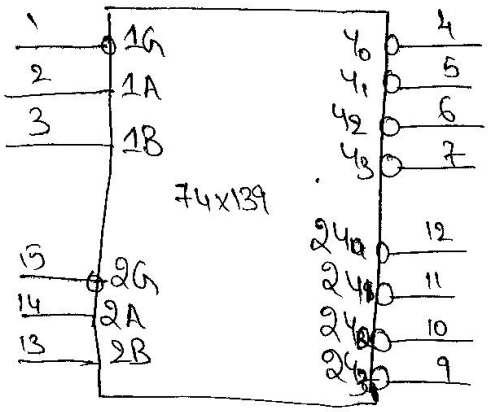
Program:-

```
library ieee;
use ieee.Std_logic_1164.ALL;
```

```
entity 74x138 is
Port ( A, B, C, G1, G2A, G2B : in Std_logic;
       Y0_L, Y1_L, Y2_L, Y3_L, Y4_L, Y5_L, Y6_L, Y7_L : out Std_logic);

end 74x138;
Architecture Structural of 74x138 is
signal NOTA, NOTB, NOTC, NOTG2A, NOTG2B, E : Std_logic;
Component NOT
  Port ( A : in Std_logic;
         ABar : out Std_logic);
  end Component;
Component NOT B
Port ( B : in Std_logic;
       BBar : out Std_logic);
  end component;
Component NOT C
   Port (C : in Std_logic;
         CBar : out Std_logic); end Component;
Component AND 3
     Port ( A, B, C : in Std_logic;
            y : out Std_logic);
       end Component;
Component NAND
     Port ( G1, G2A, G2B : in Std_logic;
            y : out Std_logic);
     end Component;

begin
   N1 : NOT A  Port map (A, A Bar);
   N2 : NOT B  Port map (B, B Bar);
   N3 : NOT C  Port map (C, C Bar);
   A3 : AND 3  Port map (A, B, C, A Bar, B Bar, C Bar);
   Nand : NAND) Port map ( G1, G2A, G2B, G1, G2A Bar, G2B Bar);
    :    :    :

end 74x138;
```

*) The * 74X139 * Dual * 2-to-4 Decoder :-

74X139 Consists of two independent and identical 2-to-4 decoder The Enable i/p and o/p of IC 74X139 are active Low.



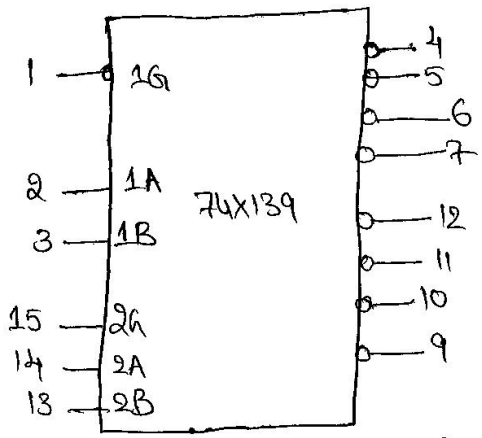
The below truth table of first half 2-to-4 decoder is similar for second half

$\overline{G}$	1A	1B	$\overline{Y_0}$	$\overline{Y_1}$	$\overline{Y_2}$	$\overline{Y_3}$
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

Second half:-

$2\overline{G}$	2A	2B	$2\overline{Y_0}$	$2\overline{Y_1}$	$2\overline{Y_2}$	$2\overline{Y_3}$
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

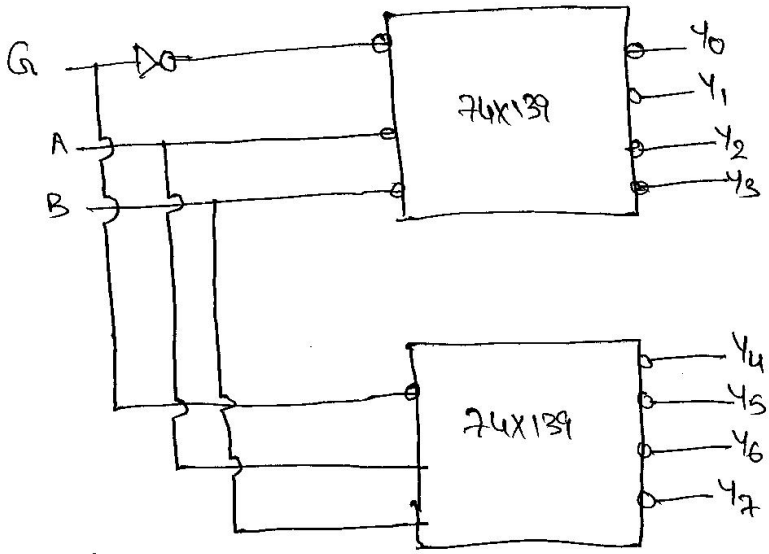
*) 74x139 IC :-



IC 74x139 has active low o/p's. The pin 'G' acts as active low enable i/p.

*) Designing a (3 to 8 decoder) using 74x139 IC

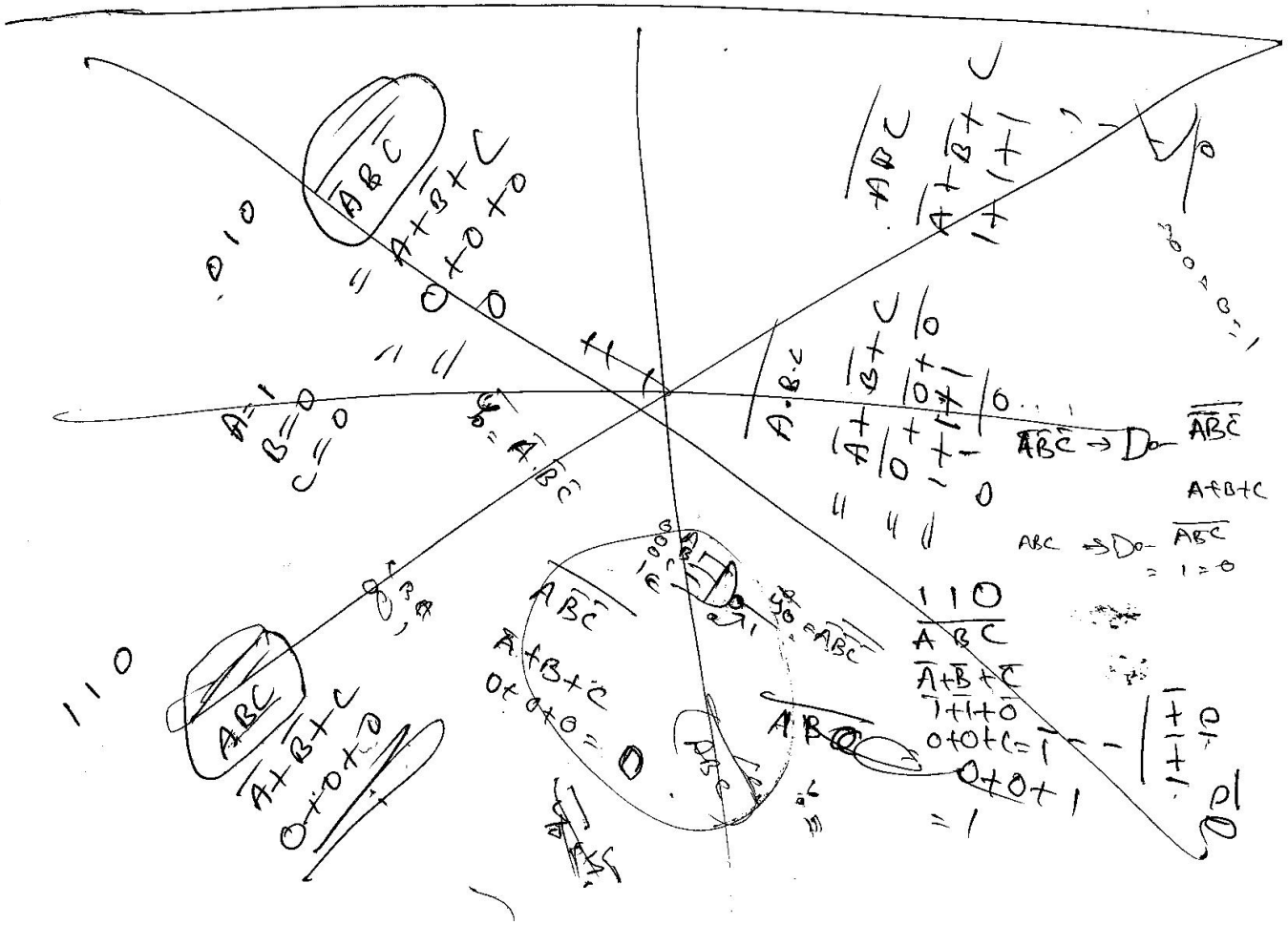
The two 74x139 IC's are cascaded to obtain (3 to 8 decoder)



The Enable i/p G_{2A-L}, and G_{2B-L} are active low and G_1 is active high.

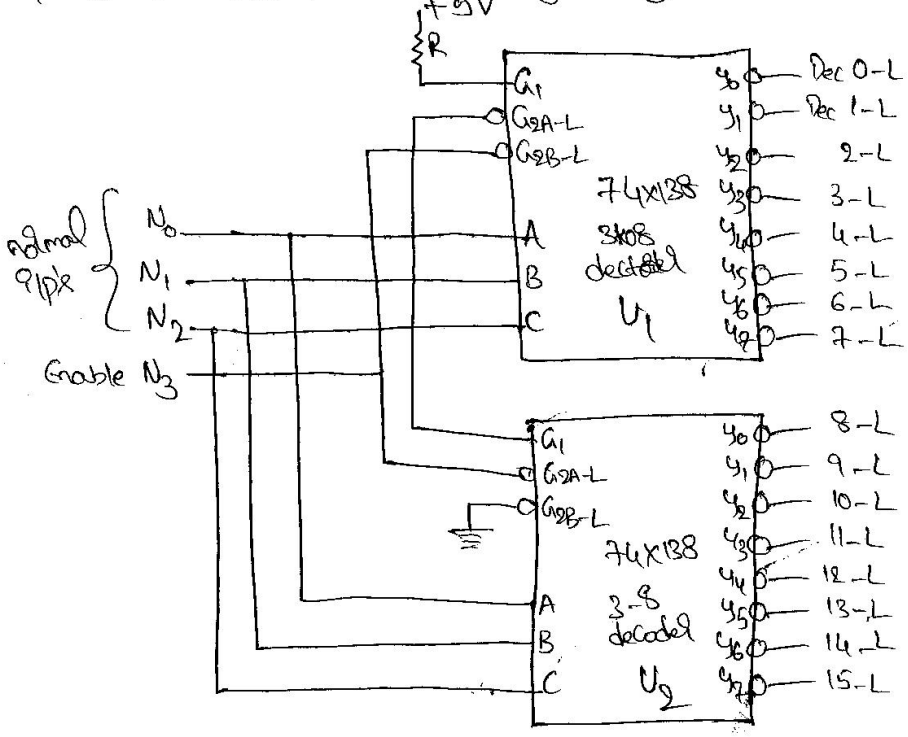
G	B	A	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	0	0	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	0	1
0	1	0	1	1	1	1	1	0	1	1
0	1	1	1	1	1	1	0	1	1	1
1	0	0	1	1	1	0	1	1	1	1
1	0	1	1	1	0	1	1	1	1	1
1	1	0	1	0	1	1	1	1	1	1
1	1	1	0	1	1	1	1	1	1	1

From truth table :- If all 3-ip A, B, G are low then all dp except Y_0 are high.



*) Design 4 to 16 decoder using two 74X138 decoder write a data flow style VHDL Program.

Sol) 4 to 16 decoder can be designed by combining two 3 to 8 decoders

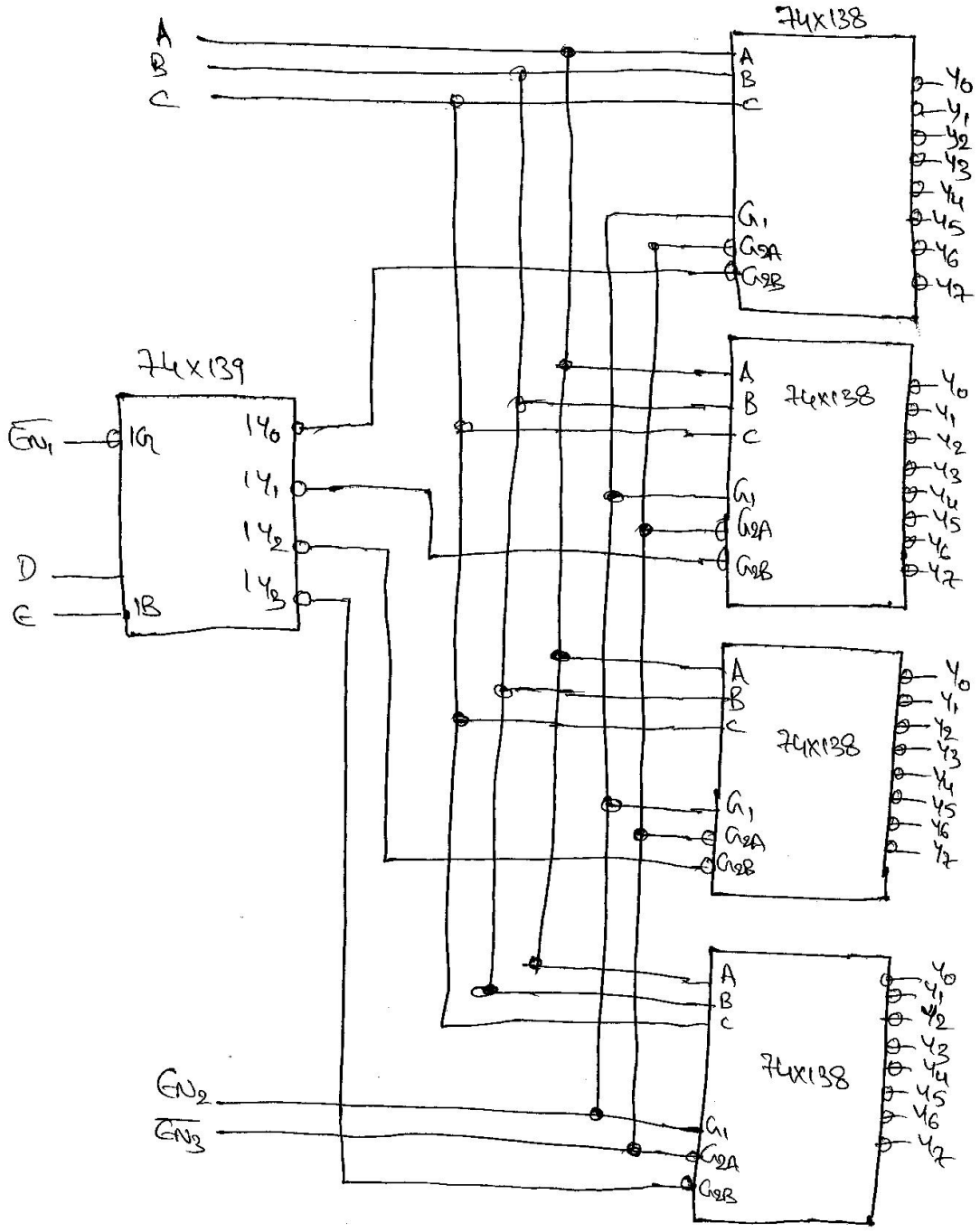


VHDL Program:-

```
Library ieee;
use ieee.std_logic_1164.ALL;
use ieee.std_logic_ARITH.ALL;
use ieee.std_logic_UNSIGNED.ALL;

entity dec 4X16 is
Port ( G1, G2A, G2B : in std_logic;
       A : in std_vector (2 down to 0);
       y_L : out std_logic_vector (0 to 15));

end dec4x16;
Architecture data flow of 4X16 dec is
signal y_L_i : std_logic_vector (0 to 15);
begin with no select y_L_iL =
```

```
y_L_1L <= "1000 0000 0000 0000" when "0000";
        = "0100 0000 0000 0000" when "0001";
        = "0010 0000 0000 0000" when "0010";
        = "0001 0000 0000 0000" when "0011";
        = "0000 1000 0000 0000" when "0100";
        = "0000 0100 0000 0000" when "0101";
        = "0000 0010 0000 0000" when "0110";
        = "0000 0001 0000 0000" when "0111";
        = "0000 0000 1000 0000" when "1000";
        = "0000 0000 0100 0000" when "1001";
        = "0000 0000 0010 0000" when "1010";
        = "0000 0000 0001 0000" when "1011";
        = "0000 0000 0000 1000" when "1100";
        = "0000 0000 0000 0100" when "1101";
        = "0000 0000 0000 0010" when "1110";
        = "0000 0000 0000 0001" when "1111";
        = "0000 0000 0000 0000" when others;
y_L_1L <= y_L_1 when (G1 and not G2A and not G2B) = '1'
          else "0000 0000 0000 0000";

end data flow;
```

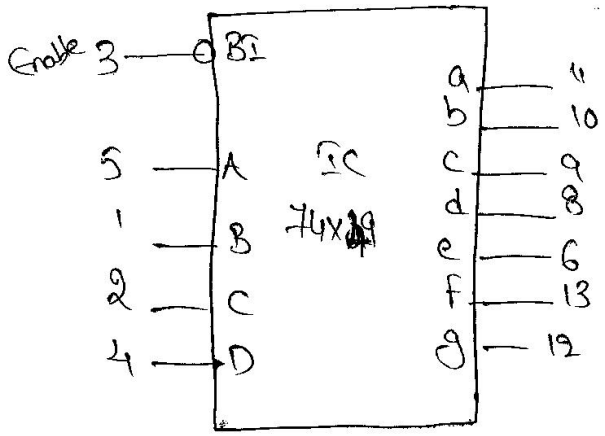
*) Design of 5-32 decoder using 74x138 :-



74x138 is (3 to 8 decoder), so it accepts three i/p i.e A, B, C and gives eight active low o/p's.

In order to design 5-32 decoder we need 4-such decoders one of these enable i/p to each of for 74x138 comes from 74x139 o/p

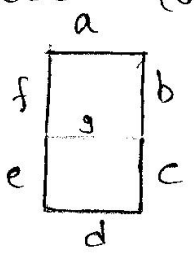
✱ IC 74x49 Seven Segment decoder:- (for Decimal & non Decimal Display)



The truth table of 74x49 seven segment decoder.

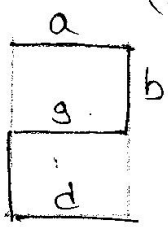
BI	D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	0	1	0	1	1	0	0	0	0
0	0	0	1	0	1	1	0	1	1	0	1
0	0	0	1	1	1	1	1	1	0	0	1
0	0	1	0	0	0	1	1	0	0	1	1
0	0	1	0	1	1	0	1	1	0	1	1
0	0	1	1	0	1	0	1	1	1	1	1
0	0	1	1	1	1	1	1	0	0	0	0
0	1	0	0	0	1	1	1	1	1	1	1
0	1	0	0	1	1	1	1	1	0	1	1
0	1	0	1	0	1	1	1	0	1	1	1
0	1	0	1	1	0	0	1	1	1	1	1
0	1	1	0	0	1	0	0	1	1	1	0
0	1	1	0	1	0	1	1	1	1	0	1
0	1	1	1	0	1	0	0	1	1	1	1
0	1	1	1	1	1	0	0	0	1	1	1
1	X	X	X	X	0	0	0	0	0	0	0

0000 (0)

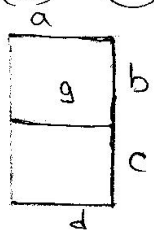


0001 (1)

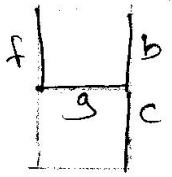
0010 (2)



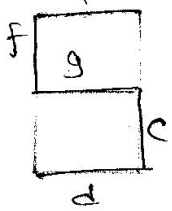
0011 (3) (16)



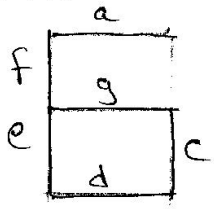
0100 (4)



0101 (5)

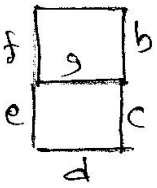


0110 (6)

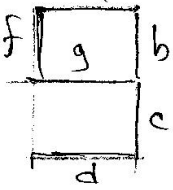


0111 (7)

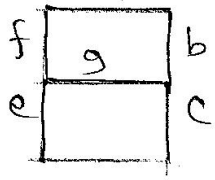
1000 (8)



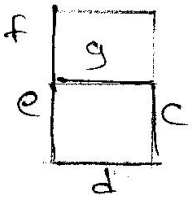
1001 (9)



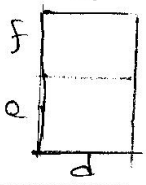
1010 (A)



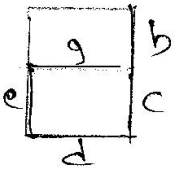
1011 (b)



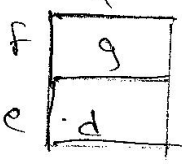
1100 (C)



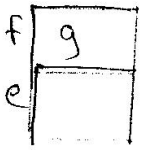
1101 (d)



1110 (E)



1111 (F)



* Write a VHDL Program for BCD to 7 segment decoder?

```
Library ieee;
use ieee.std_logic_1164.ALL;
Entity 7segment is
Port ( A,B,C,D : in std_logic;
       BI      : in std_logic;
       Leds    : out std_logic_vector (1 to 7));
End 7segments;
Architecture dataflow of 7segment is
begin Signal S : std_logic_vector(0 to 15);
```

```
With ABCD Select
    LEDS <= "1111110" when "0000";
    LEDS <= "0110000" when "0001";
    LEDS <= "1101101" when "0010";
    LEDS <= "1111001" when "0011";
    LEDS <= "0110011" when "0100";
    LEDS <= "1011011" when "0101";
    LEDS <= "1011111" when "0110";
    LEDS <= "1110000" when "0111";
    LEDS <= "1111111" when "1000";
    LEDS <= "1111011" when "1001";
    LEDS <= "1110111" when "1010";
    LEDS <= "0011111" when "1011";
    LEDS <= "1001110" when "1100";
    LEDS <= "0111101" when "1101";
    LEDS <= "1001111" when "1110";
    LEDS <= "1000111" when "1111";
    LEDS <= "0000000" when others;
    S <= LEDS when BS = '1'
         Else "0000000";

End dataflow;
```

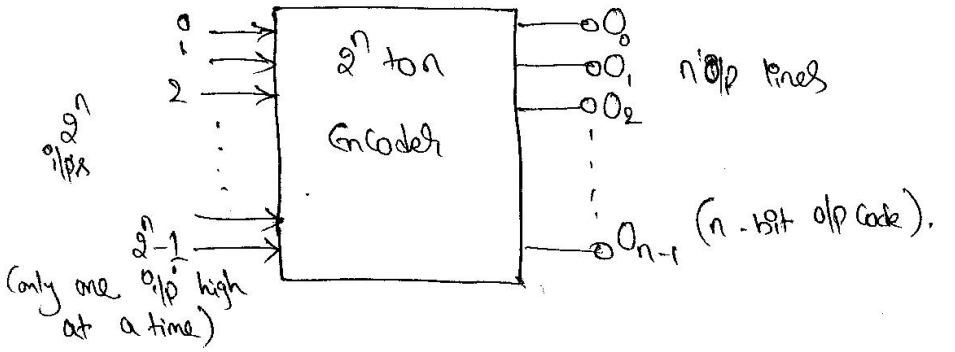
*) Write a VHDL Program of 7 Segment using Case statement?

```
Library ieee;
Use ieee.std_logic_1164.ALL;
Entity 7seg is
  Port ( A : in std_logic_vector(3 downto 0);
         Leds : out std_logic_vector(1 to 7));
End 7seg;
```

```
Architecture behavioral of 7 segment is
  begin
    Process (A)
     begin
     Case A is
        When "0000" => Led <= "1111110";
        When "0001" => Led <= "0110000";
        When "0010" => Led <= "1101101";
        When "0011" => Led <= "1111001";
        When "0100" => Led <= "0110011";
        When "0101" => Led <= "1011011";
         When "0110" => Led <= "1011111";
         When "0111" =>  Led <= "1110000";
         When "1000" => Led <= "1111111";
         When "1001" => Led <= "1111011";
         When "1010" => Led <= "1110111";
         When "1011" => Led <= "0011111";
         When "1100" => Led <= "1001110";
          When "1101" => Led <= "0111101";
          When "1110" => Led <= "1001111";
          When "1111" => Led <= "1000111";
          When "others => Led <= "-------";

End Case;
End Process;
End behavioral;
```

*) **Encoders:-** An Encoder Converts an active i/p signal into Coded o/p signal. The simplest Encoder is 2^n to n line decoder It has 2^n i/p lines, only one of them is active at a time ~~with n-bits~~. Logic circuitry within Encoder Converts this active i/p to a Coded binary o/p with n-bits. 2^n to n Encoder is Supply a Collection of OR gates.

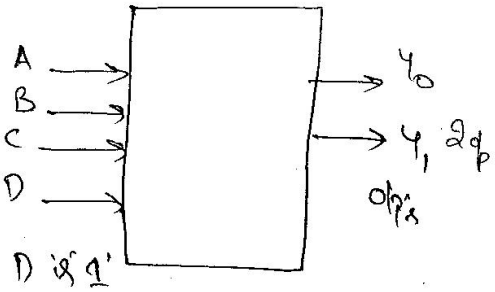
In Some Cases, more than one i/p Can be active, a special Encoder is Called Priority Encoder. Priority Encoder ensures that the o/p Code corresponds to highest i/p line if more than one i/p is active



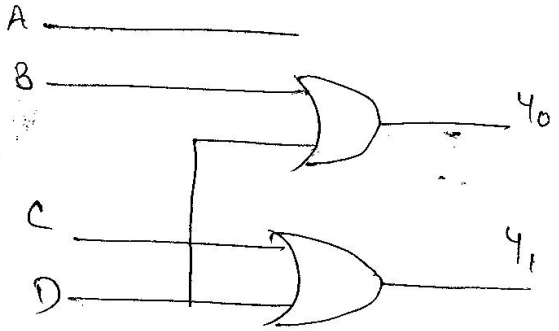
(1) **4 to 2 Encoder:-**

One of the i/p signal is asserted at a time and o/p has two bits. $n=2$ $2^2=4$ i/p

The o/p Y_0 is 1 when either i/p B (or) D is one and o/p Y_1 is 1 when i/p C (or) D is '1'



D	C	B	A	Y_1	Y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



P) VHDL Program for 4-2 decoder using Case statement?

```
library ieee;
use ieee.std_logic_1164.ALL;
Entity Encoder is
Port ( A,B,C,D : in std_logic;
       d : out std_logic_vector (1 down to 0));
End Encoder;
Architecture behavioral of Encoder is
begin
Process (A,B,C,D)
begin
Case A,B,C,D is
when "1000" => d <= "00";
when "0100" => d <= "01";
when "0010" => d <= "10";
when "0001" => d <= "11";
When others => d <= "ZZ";
End case;
End Process;
End behavioral;
```

P) VHDL Program for 4-2 decoder using if statement?

```
library ieee;
use ieee.std_logic_1164.ALL;
Entity Encoder is
Port ( A,B,C,D : in std_logic;
       d : out std_logic_vector (1 down to 0));
End Encoder;
Architecture behavioral of Encoder is
signal A0 : in std_logic_vector (3 downto 0);
begin
Process (A0, B, C, D) is
```

```
begin
if (A0 = "1000" then d <= "00";
elsif A1 = "0100" then d <= "01";
elsif A2 = "0010" then d <= "10";
elsif A3 = "0001" then d <= "11";
else d <= "ZZ";
End if;
End process;
End behavioral;.
```

P) VHDL Program of 4 to 2 Encoder using dataflow Style ?

```
library ieee;
use ieee.std_logic_1164.ALL;
Entity Encoder is
Port ( A0 : in std_logic (3 downto 0);
       d : out std_logic_vector (1 downto 0));
End Encoder;
Architecture dataflow of Encoder is
begin
    with A0 select  d <= "00" when "1000";
                    d <= "01" when "0100";
                    d <= "10" when "0010";
                    d <= "11" when "0001";
                    d <= "ZZ" when others;

End dataflow;
```

(ii) Octal to Binary Encoder :-

This accepts a 3-bit i/p code and activates one of 8 outputs lines corresponding to that code. It has 8 i/p lines and produces a 3-bit o/p code corresponding to activated i/p.

								o/p		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	A	B	C
(1)	0	0	0	0	0	0	0	0	0	0
0	(1)	0	0	0	0	0	0	0	0	1
0	0	(1)	0	0	0	0	0	0	1	0
0	0	0	(1)	0	0	0	0	0	1	1
0	0	0	0	(1)	0	0	0	1	0	0
0	0	0	0	0	(1)	0	0	1	0	1
0	0	0	0	0	0	(1)	0	1	1	0
0	0	0	0	0	0	0	(1)	1	1	1

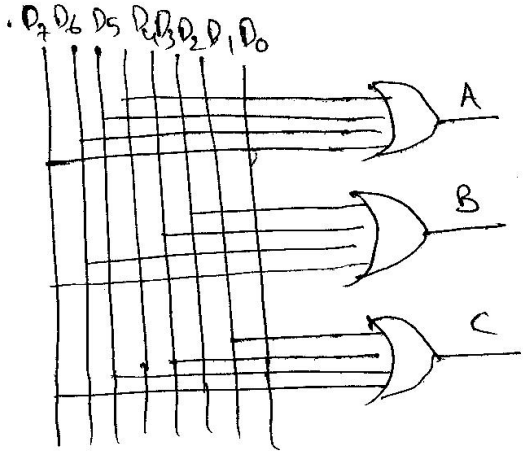
from truth table logic expression

$$A = D_4 + D_5 + D_6 + D_7$$

$$B = D_2 + D_3 + D_6 + D_7$$

$$C = D_1 + D_3 + D_5 + D_7$$

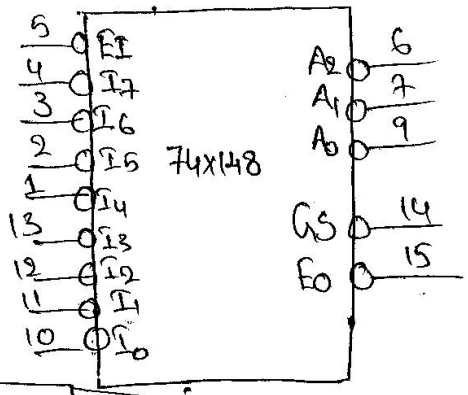
where D_0 is not represented in any expression hence it is don't care



```
Entity 0 to B Encoder is
Port ( D[7:0] : in Std_logic (7 downto 0);
       A,B,C : out Std_logic);
End 0toB Encoder;
Architecture dataflow of 0 to B Encoder is
begin
A <= (D[4] & D[5] & D[6] & D[7]);
B <= (D[2] & D[3] & D[6] & D[7]);
C <= (D[1] & D[3] & D[5] & D[7]);
End data flow of 0 to B Encoder;
```

(iii) Priority Encoder:-

The 74X148 is having 8-i/p Priority Encoder with the logic symbol



Priority Encoder:-(4-bit)

In Priority Encoder if two or more i/p are Equal to '1' at the same time, the i/p having the highest Priority will take Precedence

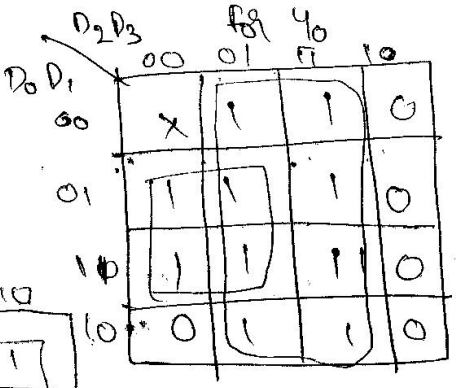
→ i/p's ← | → o/p's

D_0	D_1	D_2	D_3	Y_1	Y_0	Z
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1

from the truth table assume that D_0 has lowest Priority and D_3 has highest Priority the o/p's of Y_1 and Y_0 are o/p bits that identifies the highest Priority i/p set of '1'. ~~other wise~~ the o/p 'z' is chosen such that it is '1' when at least one(or)more i/p is '1'; otherwise it is set '0'.

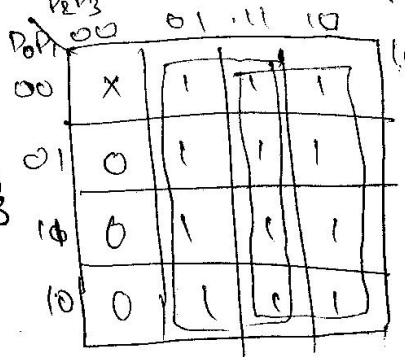
From truth table i/p $D_3 = 1$ then the o/p $Y_1Y_0 = 11$. This is due to the highest Priority of D_3 and the values of D_0, D_1, and D_2 are irrelevant in this case and has been indicated by 'x' in the truth table. The second row when $D_2 = 1$, the o/p are $Y_1Y_0 = 10$ but only when $D_3 = 0$. Similarly i/p D_1 causes the o/p's to be set to $Y_1Y_0 = 00$ only when i/p $D_0 = 0$ is asserted.

K-map from the truth table for Y_0



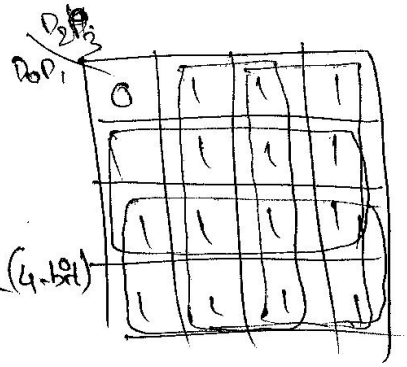
$$Y_0 = D_3 + D_1 \cdot \overline{D_2}$$

K-map for Y_1



$$Y_1 = D_2 + D_3$$

K-map for z

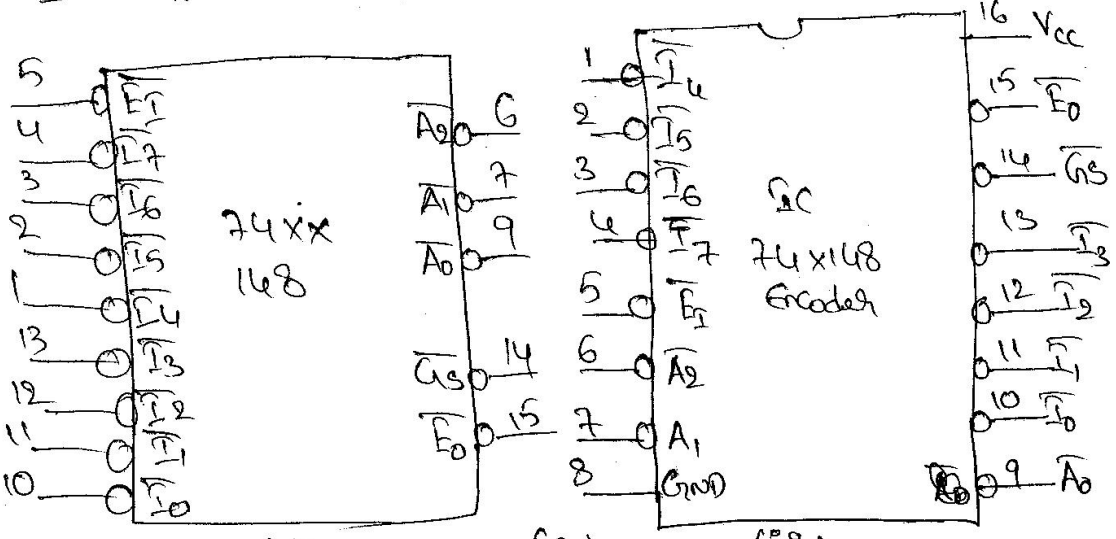


$$z = D_0 + D_1 + D_2 + D_3$$

P) Write a VHDL Program of Priority Encoder (4-bit) using dataflow style.

```
library ieee;
use ieee.std_logic_1164.ALL;
Entity Priority_Encoder is
  Port ( D : in std_logic_vector (3 downto 0);
         Y : out std_logic_vector (1 downto 0);
         Z : out std_logic);
End Priority;
Architecture dataflow of Priority_Encoder is
begin
  Y <= "11" when D(3) = '1' Else
  Y <= "10" when D(2) = '1' Else
  Y <= "01" when D(1) = '1' else
  Y <= "00" when others;
  Z <= '0' when D = "0000" Else '1';
End Priority;
```

*) Priority Encoder IC (74xx148):-



(a) (81) ("ii)

$\overline{E_I}$	$\overline{I_0}$	$\overline{I_1}$	$\overline{I_2}$	$\overline{I_3}$	$\overline{I_4}$	$\overline{I_5}$	$\overline{I_6}$	$\overline{I_7}$	$\overline{A_2}$	$\overline{A_1}$	$\overline{A_0}$	$\overline{G_S}$	$\overline{E_O}$
1	X	X	X	X	X	X	X	X	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	1	1	0	0	0
0	X	0	1	1	1	1	1	1	1	1	0	0	0
0	X	X	0	1	1	1	1	1	1	0	1	0	0
0	X	X	X	0	1	1	1	1	1	0	0	0	0
0	X	X	X	X	0	1	1	1	0	1	1	0	0
0	X	X	X	X	X	0	1	1	0	1	0	0	0
0	X	X	X	X	X	X	0	1	0	0	1	0	0
0	X	X	X	X	X	X	X	0	0	0	0	0	0

It is an 8-i/p Priority Encoder. It accepts data from 8-active low i/p's and provides a binary representation on three active low o/p's ($\overline{A_2}, \overline{A_1}, \overline{A_0}$); $\overline{E_I}$ is active low Enable i/p

A Group Select (or) Got Something (or) Group signal ($\overline{GS}$) is asserted when device is Enabled and one (or) more of i/p to Encoder are active

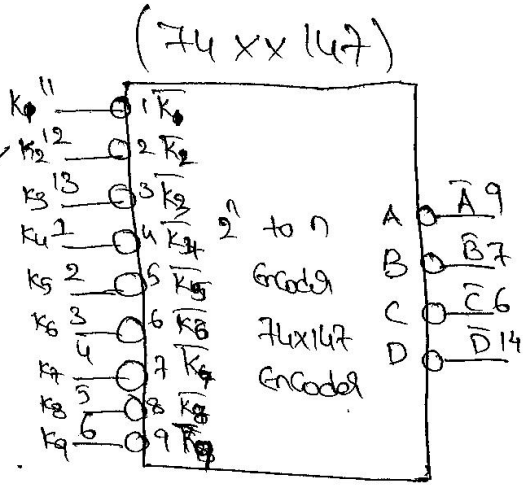
Enable o/p ($\overline{EO}$) is an active low signal that can be used to Cascade several priority Encoder device to form a larger priority Encoding System.

When all the i/p are high i.e none of i/p is active the $\overline{EO}$ goes low indicating that no priority Event Connected to the IC is present. In Cascade priority Encoder $\overline{EO}$ is Connected to ($\overline{EI}$) i/p of the next lower priority Encoder.

(iv) Decimal to BCD Encoder (74XX147) :- (basically 10 i/p's)

It has 9 i/p lines and 4 o/p lines both i/p & o/p are asserted active low. In this there are no i/p lines for decimal zero so we consider only 9 i/p lines

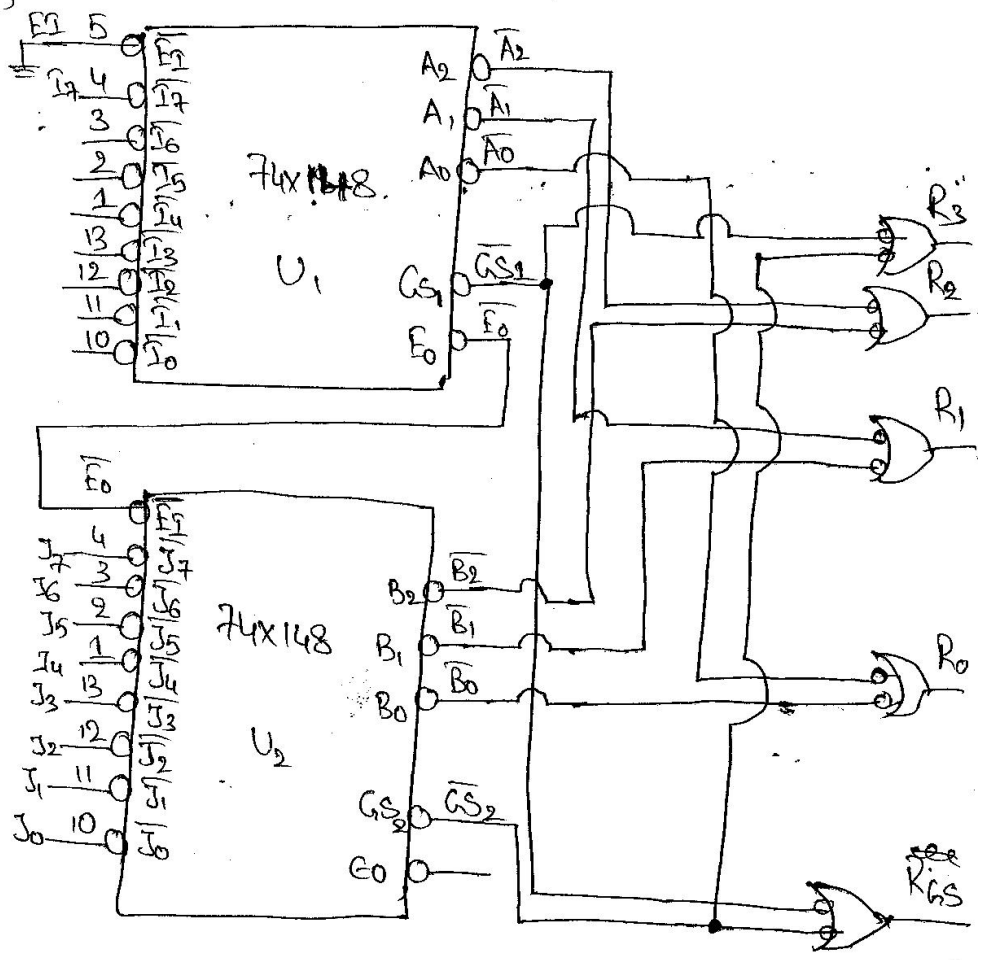
Decimal	$\overline{K_0}$	$\overline{K_1}$	$\overline{K_2}$	$\overline{K_3}$	$\overline{K_4}$	$\overline{K_5}$	$\overline{K_6}$	$\overline{K_7}$	$\overline{K_8}$	$\overline{K_9}$



(27)

Decimal	$\bar{K}_0$	$\bar{K}_1$	$\bar{K}_2$	$\bar{K}_3$	$\bar{K}_4$	$\bar{K}_5$	$\bar{K}_6$	$\bar{K}_7$	$\bar{K}_8$	$\bar{K}_9$	$\bar{D}$	$\bar{C}$	$\bar{B}$	$\bar{A}$
0	0	1	1	1	1	1	1	1	1	1	1	1	1	0
1	X	0	1	1	1	1	1	1	1	1	1	1	0	1
2	X	X	0	1	1	1	1	1	1	1	1	1	0	0
3	X	X	X	0	1	1	1	1	1	1	1	0	1	1
4	X	X	X	X	0	1	1	1	1	1	1	0	1	0
5	X	X	X	X	X	0	1	1	1	1	1	0	0	1
6	X	X	X	X	X	X	0	1	1	1	1	0	0	0
7	X	X	X	X	X	X	X	0	1	1	0	1	1	1
8	X	X	X	X	X	X	X	X	0	1	0	1	1	1
9	X	X	X	X	X	X	X	X	X	0	0	1	1	0

Design a Priority Encoder for 16-i/p using two 74x148 Encoders.



*) VHDL Program for above 8-Priority Encoder :- (74X148) IC

```
library ieee;
use ieee.std_logic_1164.ALL;
use ieee.std_logic_Arith.ALL;
use ieee.std_logic_unsigned.ALL;
Entity Priority Encoder 74x148 is
Port ( EI_L : in std_logic;
       I[7]_L : in std_logic_vector (7 downto 0);
       A_L : out std_logic_vector (2 downto 0);
       GS_L : out std_logic;
       Eo_L : out std_logic);
End Priority Encoder 74x148;
Architecture behavioral of Priority Encoder 74x148 is
  signal EI, GS, Eo : std_logic;
  signal I[7] : in std_logic_vector(7 downto 0);
         A.. : out std_logic_vector (2 downto 0);
  begin
  Process ( EI_L, I[7]_L)
  begin If EI <= '1';
  If I[7] = '1' then A <= "111"; GS <= '1';
  Elsif I[6] = '1' then A <= "110"; GS <= '1';
  Elsif I[5] = '1' then A <= "101"; GS <= '1';
  Elsif I[4] = '1' then A <= "100"; GS <= '1';
  Elsif I[3] = '1' then A <= "011"; GS <= '1';
  Elsif I[2] = '1' then A <= "010"; GS <= '1';
  Elsif I[1] = '1' then A <= "001"; GS <= '1';
  Elsif I[0] = '1' then A <= "000"; GS <= '0';
  Else Eo <= '1';
  End If;
```

```
If EI <= '0' ; else
A <= "000" ; GS <= '0' ; EO <= '0' ;
End If ;
I[I]_L <= not I[I] ; A_L = not A ;
EI_L <= not EI ;
EO_L <= not EO ;
GS_L <= not GS ;
End Process ;
End behavioral ;
```

*) VHDL Program for two 74x148 Priority Encoders :- (16-i/p)

```
library ieee ;
use ieee. Std_logic_1164.ALL ;
Entity Priority two 74x148 is
Port ( EI_L : in Std_logic ;
       I[I]_L : in Std_logic_vector (7 downto 0);
       J[I]_L : in Std_logic_vector (7 downto 0);
       EO_L : out Std_logic ; GS1, GS2 : in out Std_logic ;
       A_L : Buffer Std_logic_vector (2 down to 0);
       B_L : Buffer Std_logic_vector (2 down to 0));
       R : out Std_logic_vector (3 down to 0);
       RGS : out Std_logic ));
End Priority two 74x148 ;
Architecture behavioral of Priority two 74x148 is
begin
signal : EI , I[I] , J[I] , EO , A, B : Std_logic ;
begin Process ( EI_L, I[I], J[I])
  begin
If EI <= '1'
If I[7] = 1 then A <= "111" ; GS <= '1'
```

```
elsif I[6] = '1' then A<="110"; GS1<='1';
elsif I[5] = '1' then A<="101"; GS1<='1';
elsif I[4] = '1' then A<="100"; GS1<='1';
elsif I[3] = '1' then A<="011"; GS1<='1';
elsif I[2] = '1' then A<="010"; GS1<='1';
elsif I[1] = '1' then A<="001"; GS1<='1';
elsif I[0] = '1' then A<="000"; GS1<='1';
else GS1<='0'; Eo<='1';
end if;
if Eo<='1'
```

*) VHDL Program for Priority Encoder 16 i/p (74x148 IC's)

```
library ieee;
use ieee.std_logic_1164.ALL;
entity Pri_Encoder is
port ( EI : in std_logic;
       I[I] : in std_logic_vector (15 downto 0);
       A : out std_logic_vector (3 downto 0);
       GS : out std_logic);
end Pri_Encoder;
architecture behavioral of Pri_Encoder is
begin
process (EI, I)
begin
if EI = '0' then
if I[15] = "xxxxxxxxxxxxxxx0" then A = "0000"; GS='0';
elsif I[14] = "xxxxxxxxxxxxxx01" then A = "0001"; GS='0';
elsif I[13] = "xxxxxxxxxxxxx011" then A = "0010"; GS='0';
elsif I[12] = "xxxxxxxxxxxx0111" then A = "0011"; GS='0';
elsif I[11] = "xxxxxxxxxx01111" then A = "0100"; GS='0';
elsif I[10] = "xxxxxxxxx011111" then A = "0101"; GS='0';
elsif I[9] = "xxxxxxxx0111111" then A = "0110"; GS='0';
```

```
elsif I[8] = "XXXXXXXO111111111" then A = "0111";
elsif I[7] = "XXXXXXO11111111" then A = "1000";
elsif I[6] = "XXXXXO1111111111" then A = "1001";
elsif I[5] = "XXXXO111111111111" then A = "1010";
elsif I[4] = "XXXO11111111111111" then A = "1011";
elsif I[3] = "XXO1111111111111111" then A = "1100";
elsif I[2] = "XO11111111111111111" then A = "1101";
elsif I[1] = "O111111111111111111" then A = "1110";
elsif I[0] = "11111111111111111" then A = "1111";
else EI = '1' then I[7] = "XXXXXXXXXXXXXXXX" then
     A = "1111"; GS = '1'; EO = '1';
End if;
End Process;
End behavioral;
```

*) VHDL Program of Priority Encoder using (74x148) using dataflow?

*) Design a '10' to '4' Encoder with i/p 1 out of 10 code and o/p's in BCD. Provide the dataflow style of VHDL Program?

Sol:-

												8	4	2	1
Enable	A	B	C	D	E	F	G	H	I	J		W	X	Y	Z
0	0	0	0	0	0	0	0	0	0	0		0	0	0	0
1	1	0	0	0	0	0	0	0	0	0		0	0	0	1
1	0	1	0	0	0	0	0	0	0	0		0	0	1	0
1	0	0	1	0	0	0	0	0	0	0		0	0	1	1
1	0	0	0	1	0	0	0	0	0	0		0	1	0	0
1	0	0	0	0	1	0	0	0	0	0		0	1	0	1
1	0	0	0	0	0	1	0	0	0	0		0	1	1	0
1	0	0	0	0	0	0	1	0	0	0		0	1	1	1
1	0	0	0	0	0	0	0	1	0	0		1	0	0	0
1	0	0	0	0	0	0	0	0	1	0		1	0	0	1
1	0	0	0	0	0	0	0	0	0	1		1	0	1	0

map

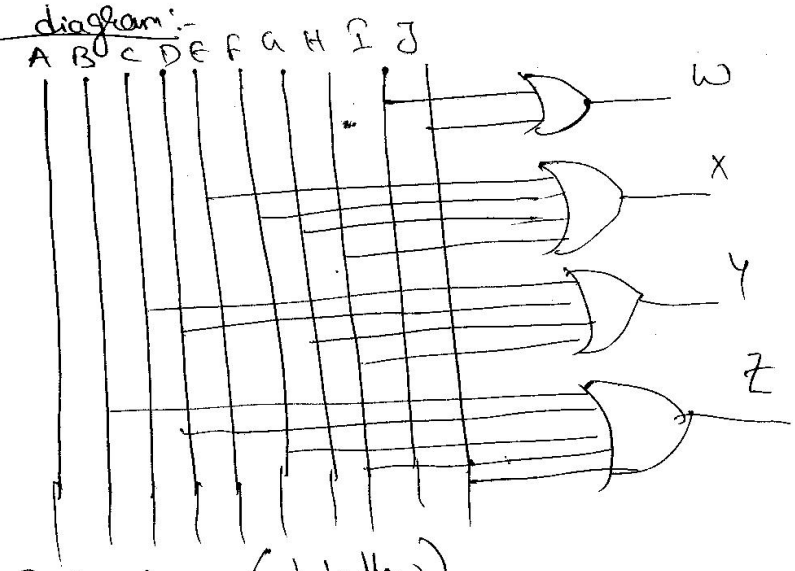
$W = I + J$

$X = E + F + G + H$

$Y = C + D + G + H$

$Z = B + D + F + H + J$

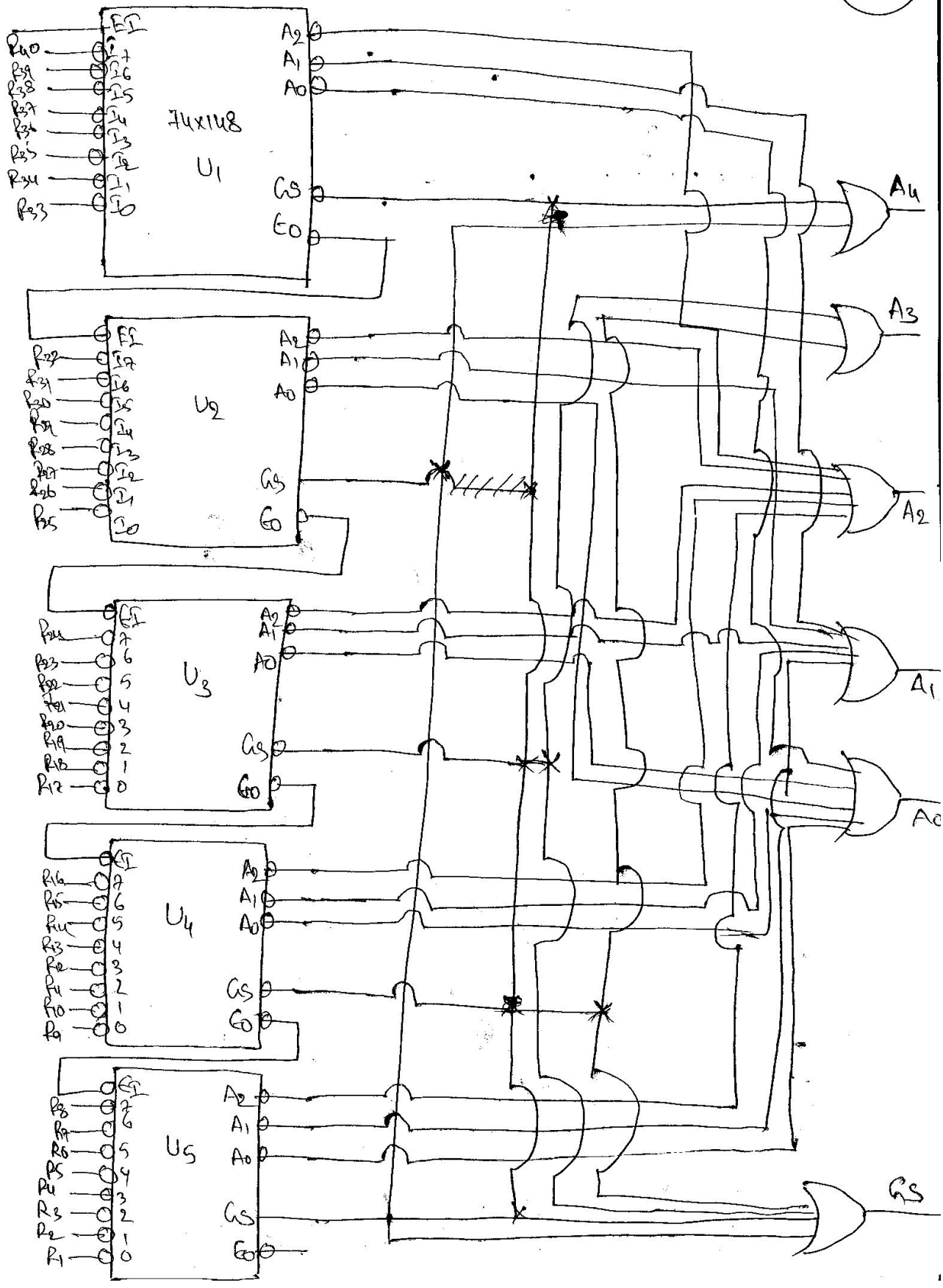
logic diagram:-



VHDL Program:- (dataflow).

```
library ieee;
use ieee. std_logic_1164.ALL;
Entity Encoder 10x4 is
Port (a,b,c,d,e,f,g,h,i,j : in std_logic;
      w x y z : out std_logic);
End Encoder 10x4;
Architecture dataflow of Encoder 10x4 is
begin
with a,b,c,d,e,f,g,h,i,j, select
   w <= (i (or) j);
   x <= (e (or) d (or) g (or) h);
   y <= ( c (or) d (or) g (or) h);
   z <= (b (or) d (or) f (or) h (or) j);
End dataflow;
```

*) Implement the ~~40~~ 32 i/p to 5 o/p Priority Encoder using 74x148 and gates



*) A mechanical disk rotates in a circle in different positions. Two successive positions differ with an angle of 15°. Provide an encoding mechanism for every position of the disk. The disk in the mechanical system o/p's this encoded information to detect the exact position. Design a decoder with an enable i/p to identify the position of the disk.

Sol: One disk rotation = 360°

Two positions differ with an angle = 15°

Total position = $\frac{360^\circ}{15^\circ}$ = 24 positions.