VIT
Vellore Institute of Technology

School of Computer Science Engineering and Information Systems

Fall Semester 2023-2024

Continuous Assessment Test – I

Programme Name & Branch: BTECH CSE (BCB, BCE, BCI, BCT, BDS, BKT)

Course Name & code: BECE102L Digital Systems Design **Slot: B2+TB2**

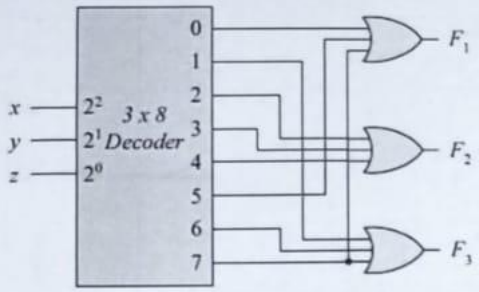
Exam Duration: 90 Min. **Maximum Marks: 50**

General instruction(s):

1. *Answer all the questions*

<table>
<tr><th>Q. No.</th><th>Question</th><th>Max Marks</th><th>CO</th><th>BL</th></tr>
<tr><td>1.</td><td>Simplify the Boolean function "F" using K-map and implement the function using only 2 input NAND gates $F = wxy'z + w'xz + wxyz$</td><td>10</td><td>CO1</td><td>BL3</td></tr>
<tr><td>2.</td><td>a) Reduce the following Boolean function "G" to three literals by Boolean laws/postulates/theorems $G = (x'y' + z)' + z + xy + wz$
b) Draw the schematic of (three literals function/simplified function) "G" using CMOS cells.</td><td>10</td><td>CO1</td><td>BL6</td></tr>
<tr><td>3.</td><td>a. Find out errors in the following verilog program:
<table>
<tr><td>Line</td><td>Error in code</td></tr>
<tr><td>1</td><td>Module 1_mydesign (A,B,C);</td></tr>
<tr><td>2</td><td>input a, b;</td></tr>
<tr><td>3</td><td>output c;</td></tr>
<tr><td>4</td><td>always (*)</td></tr>
<tr><td>5</td><td>begin;</td></tr>
<tr><td>6</td><td>if (a= 1'b01)</td></tr>
<tr><td>7</td><td>assign c <= b;</td></tr>
<tr><td>8</td><td>Else</td></tr>
<tr><td>9</td><td>assign c = a;</td></tr>
<tr><td>10</td><td>end</td></tr>
<tr><td>11</td><td>end module;</td></tr>
</table>
b. Write a verilog code in dataflow level modeling for the following verilog program.
<pre>
module test (in, En, Y);
input [1:0] in;
input En;
output reg [3:0] Y;
always @(*)
begin
    if (En == 1'b1)
        begin
        case(in)
        2'b00: Y = 4'b0001;
</pre></td><td>10</td><td>CO2</td><td>BL3</td></tr>
</table>

```
        2'b01: Y = 4'b0010;
        2'b10: Y = 4'b0100;
        2'b11: Y =4'b1000;
        endcase
        end
    else
    Y = 4'b0000;
end
endmodule
```

4.	Find the Boolean functions from the below decoder circuit. Implement the function F1 using 8:1 MUX(s), F2 using 4:1 MUX(s), F3 using 2:1 MUX(s). 	**10**	**CO 3**	**BL6**
5.	Design the combinational circuit with less number of logic gates for the input A, B, C and D as given in the truth table. Let Y and Z be the outputs of the design. Write the gate level/structural verilog code of the design along with its testbench code to verify it.	**10**	**CO 3**	**BL6**

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

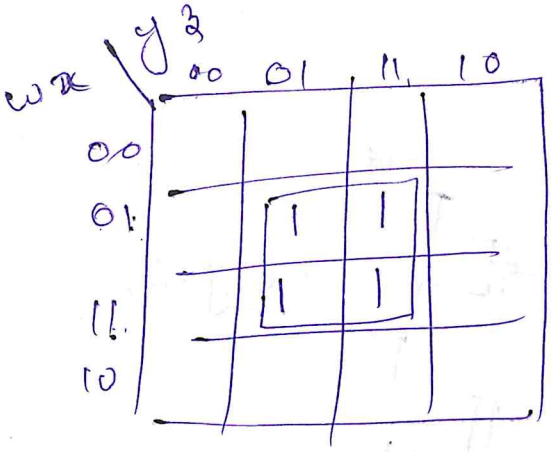
B2 BECEW2L – CA11 – B2 – KEY.

1) $F = wxy'z + w'xz + wxyz$

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

0 1 1 1

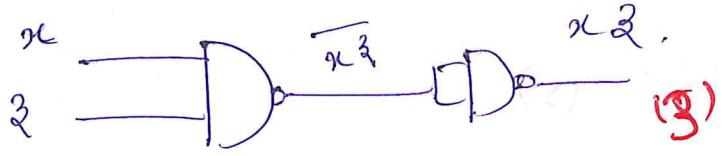
(3)



(4)

$F = xz$

wx yz
01 01
11 11



(3)

2) a) $G = (x'y' + z)' + z + xy + wz$.

$= (x'y')'z' + z + xy + wz$

$= ((x')' + (y')')z' + z + xy + wz$

$= (x + y)z' + z + xy + wz$

* Using $AB + BC = (A+B)(A+C)$

$= (x+y+z)(z+z') + xy + wz$

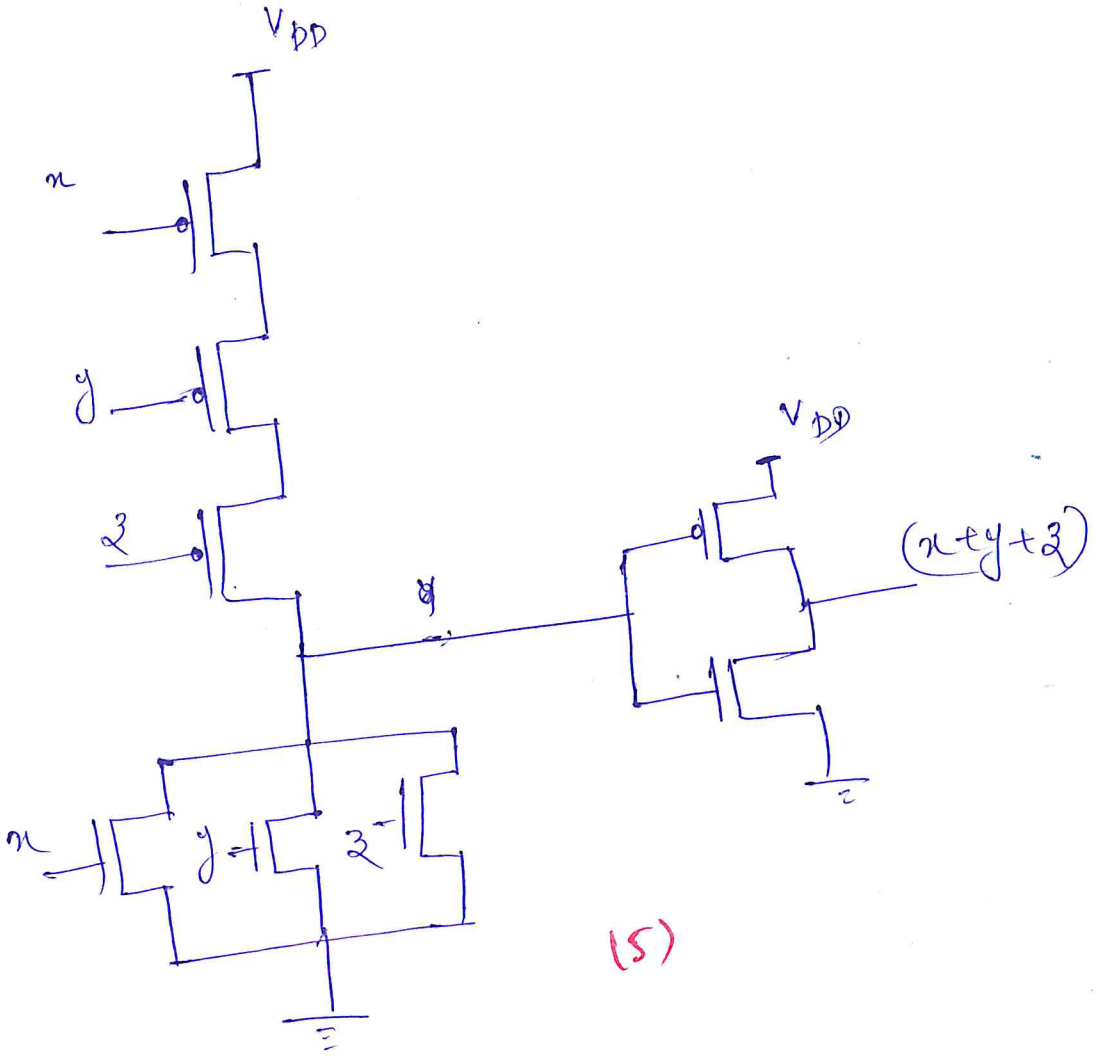
$= (x+y+z) + xy + wz$

$= z(1+w) + x + y + xy$

$= z + x(1+y) + y$ (5)

$z + xy = x + y + z$

b)



(5)

3. i) Module !_ mydesign (A, B, C);

Error: Module → module (capital M)

module name should start with a letter not with a number.

---

2) input a,b;

Inputs are not defined in capital letters.

input a,b; → input A, B;

Colon to semi colon

3) output C;

C should be capital, neg is missing

output c; → output ~~reg~~ neg C;

4) always (*) → always @(*)

@ is missing.

5) begin ;

No need of Semicolon.

begin ; → begin

6) if (a = 1'b01) → if (a == 1'b1)

Defined as single bit, but two bits are there.
if (a) is enough.

7) assign c <= b ;

~~no error~~ - can't use assign statement inside always @(

c & b should be in caps.

8) Else → else

E should be small letter.

9) assign c = a ;

~~no error~~ Can't use assign statement inside always @(

c & a should be in caps

10) end

11) end module ; → endmodule

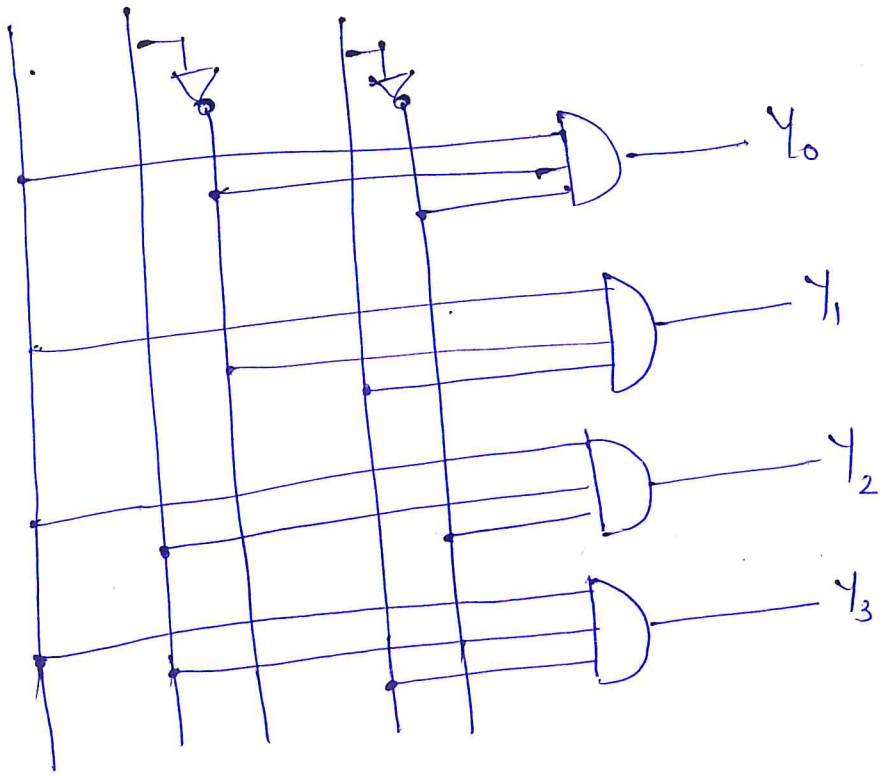
There should not be any space b/w end and module.

No need of Semicolon.

3) b) inputs ~~is~~ are Each error (1/2)

Decoder. Net. Max. 10 error's

En. in[0] in[1]




```
module test (in, En, Y);
input [1:0] in;
input En;
output reg [3:0] Y;
assign Y[0]= En & ~ in[0] & ~ in[1];
assign Y[1]= En & ~ in[0] & in[1];
assign Y[2]= En & in[0] & ~ in[1];
assign Y[3]= En & in[0] & in[1];
endmodule.
```

15)

4) $F_1 = \Sigma(0, 5, 7)$

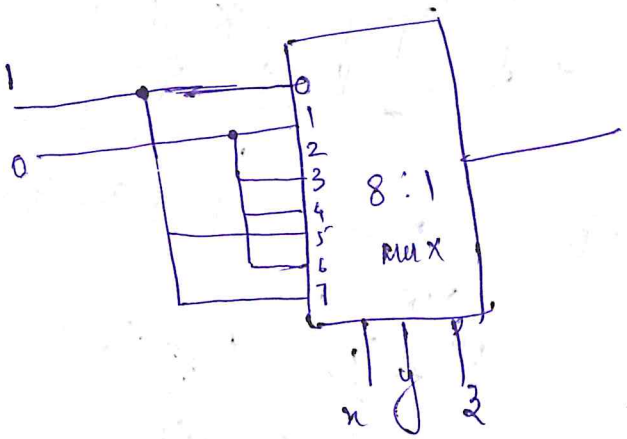
$F_2 = \Sigma(2, 3, 4)$

$F_3 = \Sigma(1, 6, 7)$

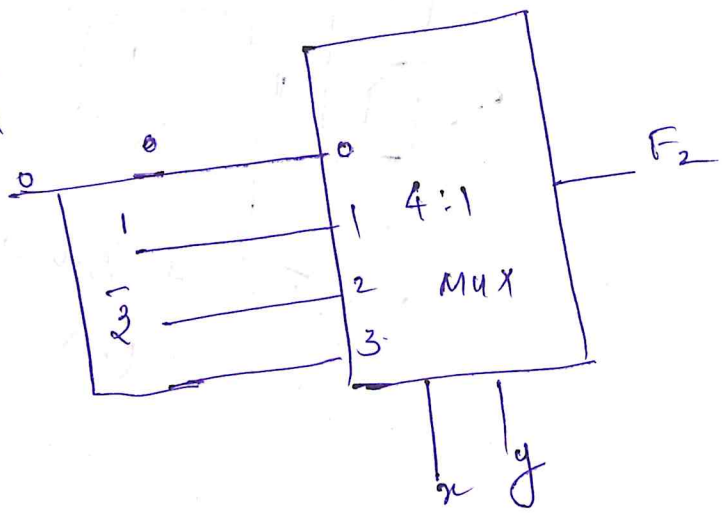
$F_1 \rightarrow$ 8 : 1 mux

x	y	z	F_1	F_2	F_3
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	0	1	0
1	0	1	1	0	0
1	1	0	0	0	0
1	1	1	1	0	1

x & y as select lines. ⟶ $F_2 \rightarrow$ 4:1 mux.



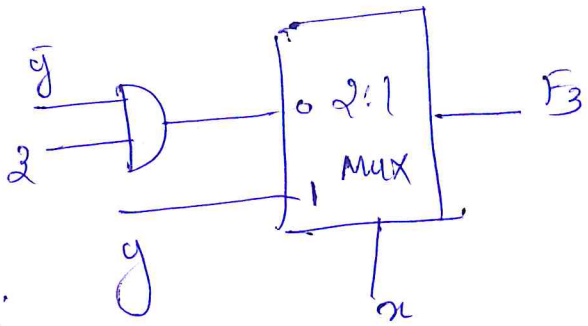
(2)



(4)

x	y	z	F_3
0	0	0	0
0	0	1	1
0	1	0	0
0	0	0	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

$F_3 = \bar{y}z$

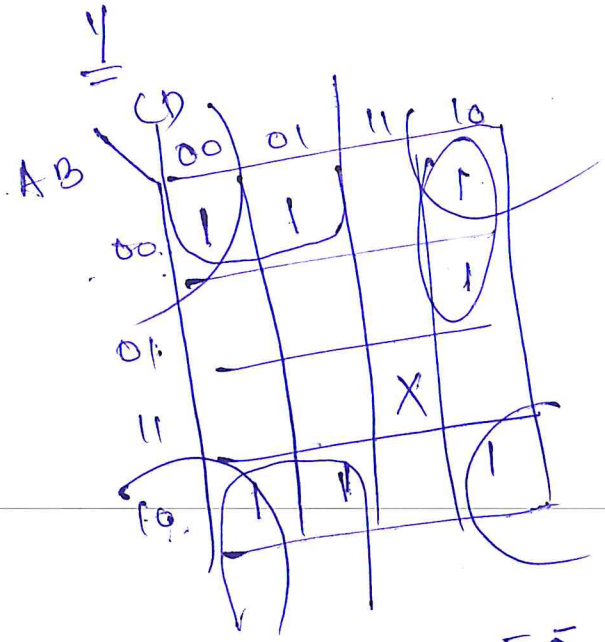


$F_3 = y\bar{z} + yz$

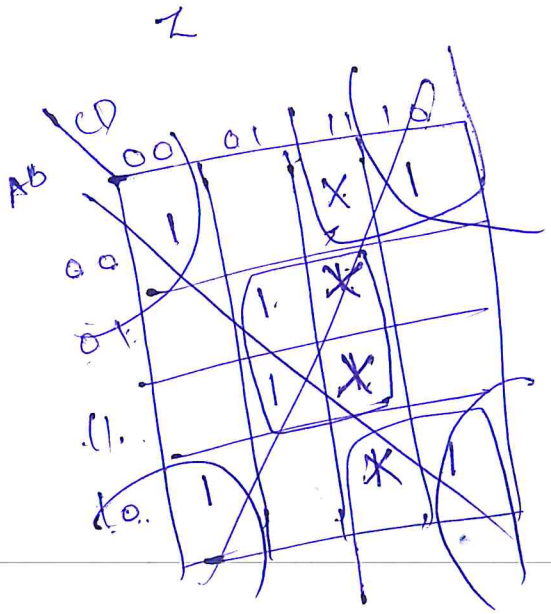
$= y(\bar{z} + z)$

$= y$ (4)

5)



$Y = \bar{B}\bar{C} + \bar{B}\bar{D} + \bar{A}C\bar{D}$ (2)



A B CD
0 0 0 0
1 0 0 1

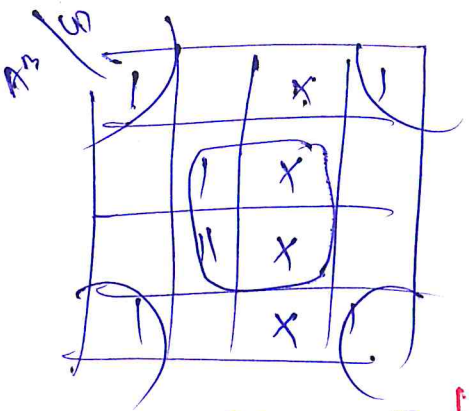
0 0 0 0
1 0 1 0

0 1 0 1
1 1 1 1

0 0 0 0
1 0 1 0

0 0 1 1
1 0 1 0

$Z = BD + \bar{B}\bar{D} + \bar{B}C$



$Z = BD + \bar{B}\bar{D}$ (2)

```
module comb (A, B, C, D, Y, Z)
input A, B, C, D;
output Y, Z;
wire w1, w2, w3, w4, w5, w6, w7, w8;

not n1 (w1, A)
not n2 (w2, B)
not n3 (w3, C)
not n4 (w4, D);

and a1 (w5, w1, C, w4);
and a2 (w6, w2, w4);
and a3 (w7, w2, w3);
and a4 (w8, B, D);

or o1 (Y, w5, w6, w7);
or o2 (Z, w6, w8);

endmodule (3)

module testbench_comb;
reg A, B, C, D;
wire Y, Z;
integer i;
comb tb (A, B, C, D, Y, Z);
initial
```

```
begin
for (i=4'b0; i<16 ; i=i+1)
  begin
  {A, B, C, D} = i;   #10;
    end
    {A, B, C, D} = 4'b0;
    end
    endmodule.
```

(3)