DIGITAL DESIGN

ASSIGNMENT REPORT

ASSIGNMENT ID : 2

Student Name: 杨钰城

Student ID: 12112323

PART 1: DIGITAL DESIGN THEORY

Provide your answers here:

Assignment 2

—Part 1.

Q1 we assume a four-bit input a, b, c, d, $x \oplus y = x'y + y'x$

assume that output = A, B, C, D to express the bit place

a	b	c	d	A	B	C	D
0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1
0	0	1	0	1	1	1	0
0	0	1	1	1	1	0	1
0	1	0	0	1	1	0	0
0	1	0	1	1	0	1	1
0	1	1	0	1	0	1	0
0	1	1	1	1	0	0	1
1	0	0	0	1	0	0	0
1	0	0	1	0	1	1	1
1	0	1	0	0	1	1	0
1	0	1	1	0	1	0	1
1	1	0	0	0	1	0	0
1	1	0	1	0	0	1	1
1	1	1	0	0	0	1	0
1	1	1	1	0	0	0	1

A's k-map

ab \ cd	00	01	11	10
00	0	1	1	1
01	1	1	1	1
11	0	0	0	0
10	1	0	0	0

$$A = a'b + a'd + a'c + ab'c'd'$$
$$= a'(b+c+d) + a(b+c+d)'$$
$$= a \oplus (b+c+d)$$

B's k-map

ab \ cd	00	01	11	10
00	0	1	1	1
01	1	0	0	0
11	1	0	0	0
10	0	1	1	1

$$B = b'd + b'+c + bc'd'$$
$$= b'(c+d) + b(c+d)'$$
$$= b \oplus (c+d)$$

$$C = c'd + cd$$
$$= c \oplus d$$

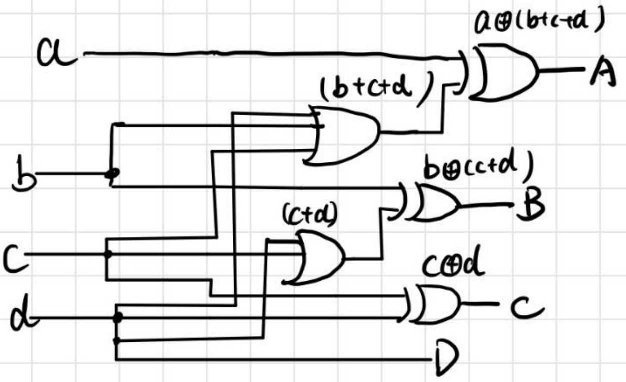
C's k-map

ab \ cd	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	0	1	0	1
10	0	1	0	1

D's k-map

ab \ cd	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	1	1	0
10	0	1	1	0

$$D = d$$



Q2

The truth table:

x	y	z	A	B	C
0	0	0	0	1	0
0	0	1	0	1	1
0	1	0	1	0	0
0	1	1	0	1	0
1	0	0	0	1	1
1	0	1	1	0	0
1	1	0	1	0	1
1	1	1	1	1	0

A's k-map

x\yz	00	01	11	10
0	0	0	0	1
1	0	1	1	1

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

$= xz + yz'$

B's k-map

x\yz	00	01	11	10
0	1	1	1	0
1	1	0	1	0

$= (y+z')(y'+z) = y'z' + yz$

C's k-map

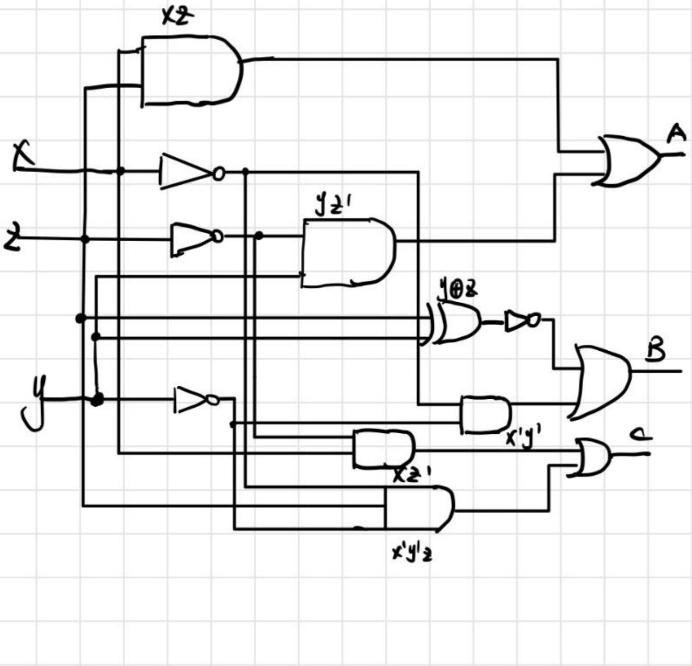
x\yz	00	01	11	10
0	0	1	0	0
1	1	0	0	1

$= xz' + x'y'z$

$A = xz + yz'$

$B = y'z' + yz + x'y' = (y \oplus z)' + x'y'$

$C = xz' + x'y'z$

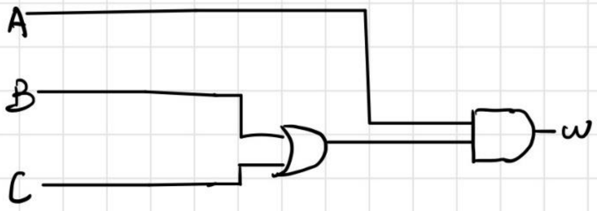


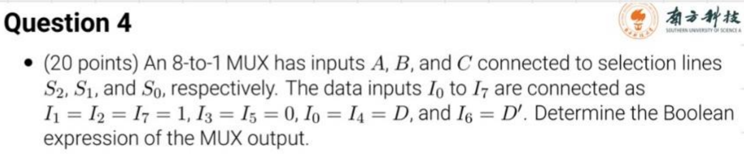
Q3

$F(A,B,C,D)=\sum C(10,11,12,13,14,15) = AB + AC$

$= A(B+C)$

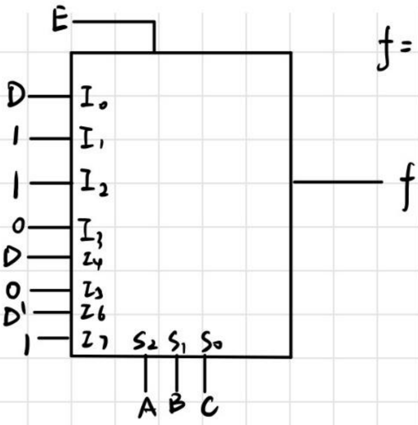
AB \ CD	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	0	0	1	1





Question 4

- (20 points) An 8-to-1 MUX has inputs A, B, and C connected to selection lines S_2, S_1, and S_0, respectively. The data inputs I_0 to I_7 are connected as $I_1 = I_2 = I_7 = 1$, $I_3 = I_5 = 0$, $I_0 = I_4 = D$, and $I_6 = D'$. Determine the Boolean expression of the MUX output.

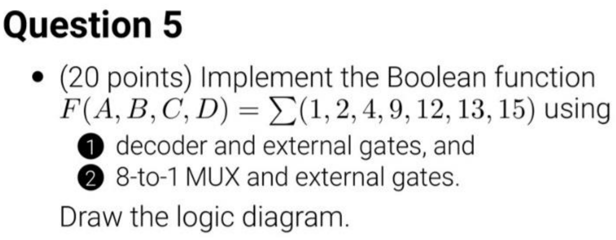


$f = A'B'C'D + A'B'C + A'BC' + AB'C'D + ABC'D' + ABC$

AB \ CD	00	01	11	10
00	0	1	1	1
01	1	1	0	0
11	1	0	1	1
10	0	1	0	0

$f = BC'D' + B'C'D + A'BC' + A'B'C + ABC$

$= C'(B \oplus D + A'B) + C(A \oplus B)'$



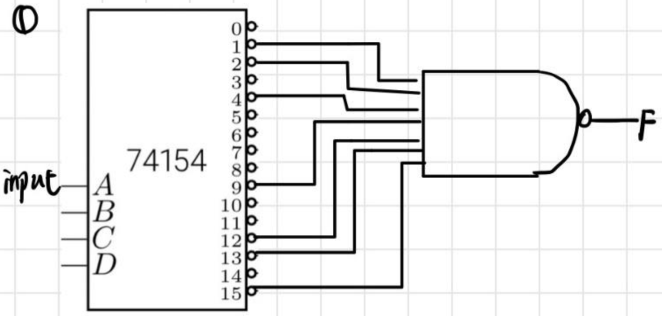
Question 5

- (20 points) Implement the Boolean function $F(A, B, C, D) = \sum(1, 2, 4, 9, 12, 13, 15)$ using
  1. decoder and external gates, and
  2. 8-to-1 MUX and external gates.

  Draw the logic diagram.

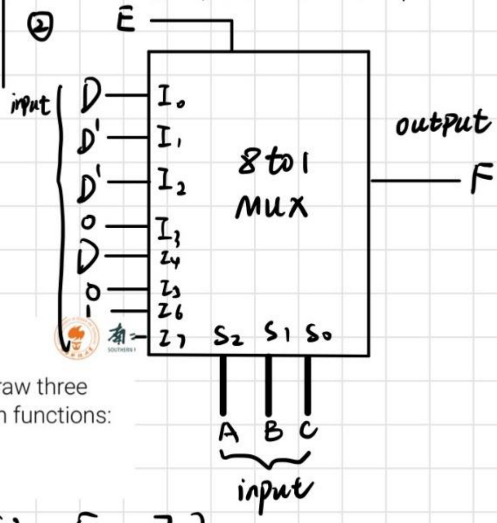
K-map

AB \ CD	00	01	11	10
00	0	1	0	1
01	1	0	0	0
11	1	1	1	0
10	0	1	0	0



A	B	C	D	F	I
0	0	0	0	0	D
0	0	0	1	1	
0	0	1	0	1	D'
0	0	1	1	0	
0	1	0	0	1	D'
0	1	0	1	0	
0	1	1	0	0	0
0	1	1	1	0	
1	0	0	0	0	D
1	0	0	1	1	
1	0	1	0	0	0
1	0	1	1	0	
1	1	0	0	1	1
1	1	0	1	1	
1	1	1	0	0	D
1	1	1	1	1	

$f = A'B'C'D + AB'CD' + A'BC'D' + ABC' + ABCD$
$= \Sigma(1, 2, 4, 9, 12, 13, 15)$



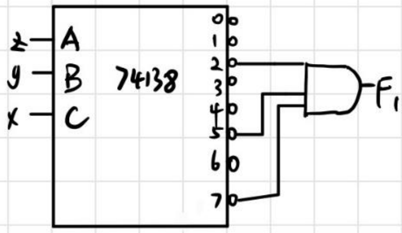
Question 6

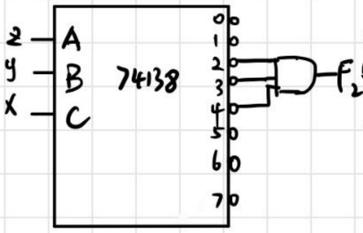
- (10 points) Using decoders and external gates, design and draw three combinational circuits defined by the following three Boolean functions: $F_1 = x'yz' + xz$, $F_2 = xy'z' + x'y$, $F_3 = x'y'z' + xy$.

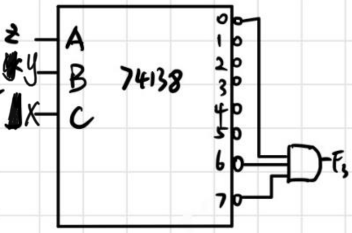
$F_1 = x'yz' + xz = x'yz' + xyz + xy'z = \Sigma(2, 5, 7)$

$F_2 = xy'z' + x'y = xy'z' + x'yz' + x'yz = \Sigma(2, 3, 4)$

$F_3 = x'y'z' + xy = x'y'z' + xyz + xyz' = \Sigma(0, 6, 7)$





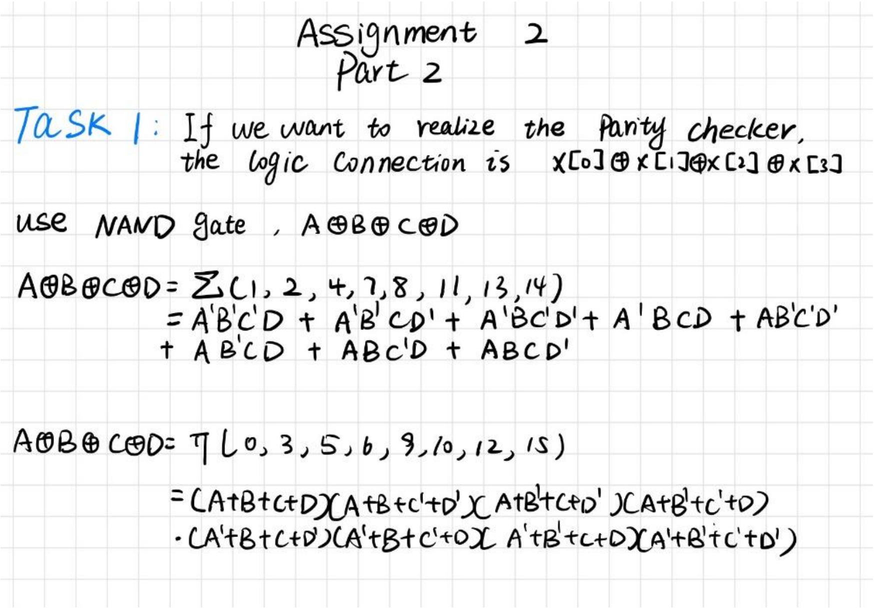


PART 2: DIGITAL DESIGN LAB (TASK1)

DESIGN

Describe the design of your system by providing the following information:

- *Verilog design (provide the Verilog code)*
- *Truth-table*



Assignment 2
Part 2

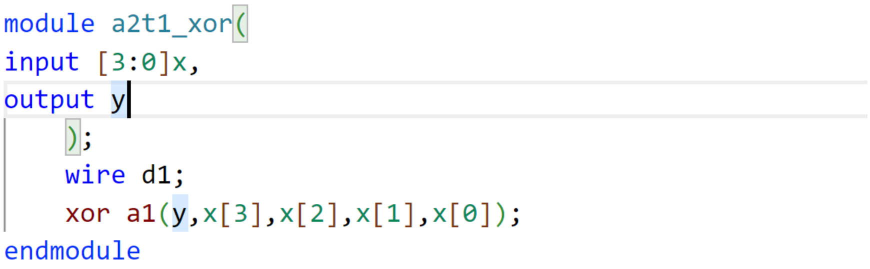
Task 1: If we want to realize the Parity checker, the logic Connection is x[0] ⊕ x[1] ⊕ x[2] ⊕ x[3]

use NAND gate , A ⊕ B ⊕ C ⊕ D

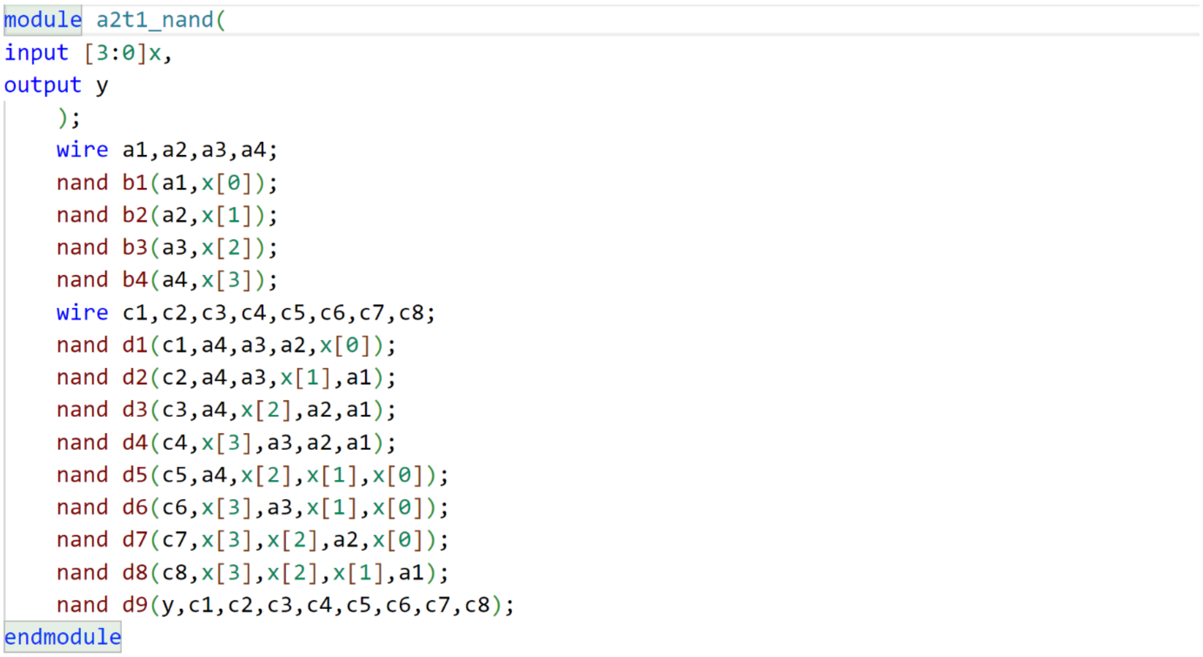
$$A \oplus B \oplus C \oplus D = \sum(1,2,4,7,8,11,13,14)$$
$$= A'B'C'D + A'B'CD' + A'BC'D' + A'BCD + AB'C'D' + AB'CD + ABC'D + ABCD'$$

$$A \oplus B \oplus C \oplus D = \prod(0,3,5,6,9,10,12,15)$$
$$= (A+B+C+D)(A+B+C'+D')(A+B'+C+D')(A+B'+C'+D) \cdot (A'+B+C+D')(A'+B+C'+D)(A'+B'+C+D)(A'+B'+C'+D')$$

-



```
module a2t1_xor(
input [3:0]x,
output y
    );
    wire d1;
    xor a1(y,x[3],x[2],x[1],x[0]);
endmodule
```

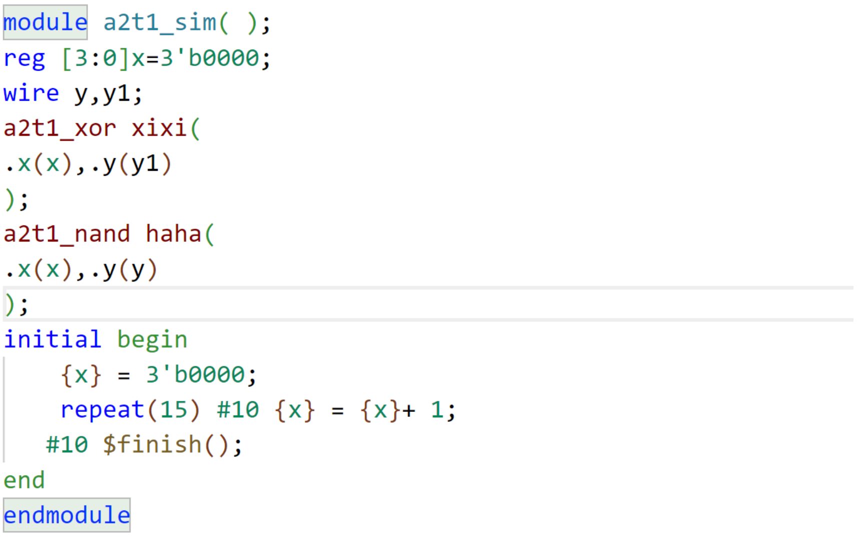


```
module a2t1_nand(
input [3:0]x,
output y
    );
    wire a1,a2,a3,a4;
    nand b1(a1,x[0]);
    nand b2(a2,x[1]);
    nand b3(a3,x[2]);
    nand b4(a4,x[3]);
    wire c1,c2,c3,c4,c5,c6,c7,c8;
    nand d1(c1,a4,a3,a2,x[0]);
    nand d2(c2,a4,a3,x[1],a1);
    nand d3(c3,a4,x[2],a2,a1);
    nand d4(c4,x[3],a3,a2,a1);
    nand d5(c5,a4,x[2],x[1],x[0]);
    nand d6(c6,x[3],a3,x[1],x[0]);
    nand d7(c7,x[3],x[2],a2,x[0]);
    nand d8(c8,x[3],x[2],x[1],a1);
    nand d9(y,c1,c2,c3,c4,c5,c6,c7,c8);
endmodule
```

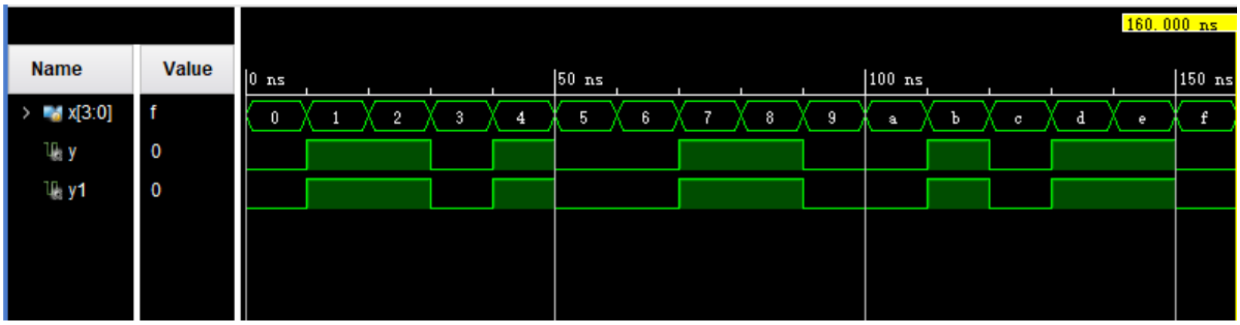
SIMULATION

Describe how you build the test bench and do the simulation.

- *Using Verilog(provide the Verilog code)*
- *Wave form of simulation result (provide screen shots)*
- *The description on whether the simulation result is same as the truth-table, is the function of the design meet the expectation.*



```
module a2t1_sim( );
reg [3:0]x=3'b0000;
wire y,y1;
a2t1_xor xixi(
.x(x),.y(y1)
);
a2t1_nand haha(
.x(x),.y(y)
);
initial begin
    {x} = 3'b0000;
    repeat(15) #10 {x} = {x}+ 1;
   #10 $finish();
end
endmodule
```



- *Through the simulation result ,we can see the result is obviously right as the truth-table,and we use truth-table to design this logic gate,so we can think that the function of the design meet the expectation.*

THE DESCRIPTION OF OPERATION

Describe the problem occurred while in the lab and your solution. Any suggestions are welcomed.

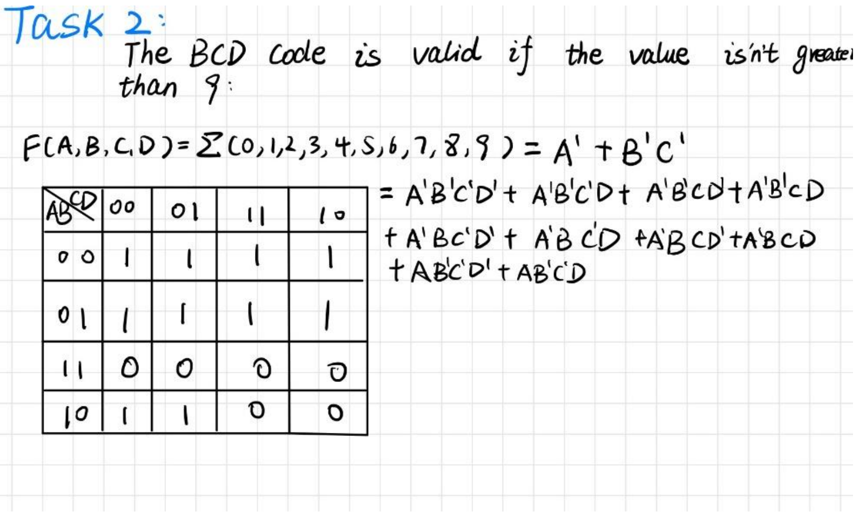
- *Problems and solutions*

PART 2: DIGITAL DESIGN LAB (TASK2)

DESIGN

Describe the design of your system by providing the following information:

- *Verilog design while* using primitive gates on Sum-of-Minterms
- *Verilog design while using* primitive gates on Product-of-Maxterms
- *Truth-table*



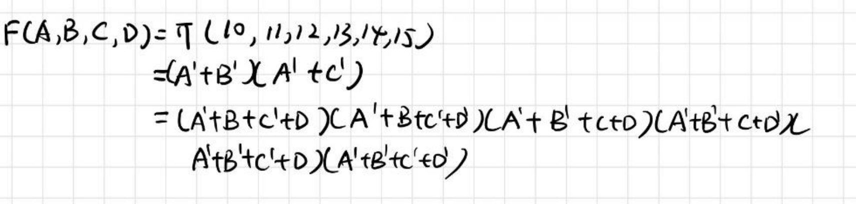
Task 2:

The BCD code is valid if the value isn't greater than 9:

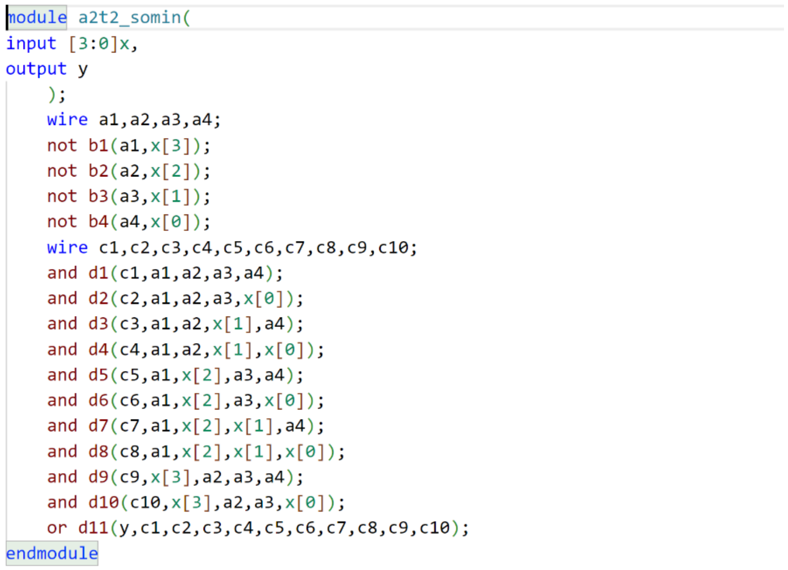
$F(A,B,C,D) = \Sigma(0,1,2,3,4,5,6,7,8,9) = A' + B'C'$

AB\CD	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	0	0	0	0
10	1	1	0	0

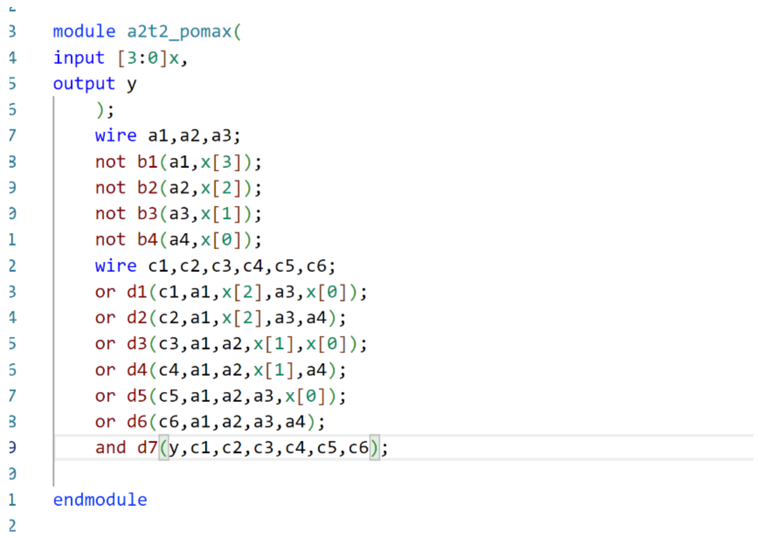
$= A'B'C'D' + A'B'C'D + A'B'CD' + A'B'CD$
$+ A'BC'D' + A'BC'D + A'BCD' + A'BCD$
$+ AB'C'D' + AB'C'D$



$F(A,B,C,D) = \Pi(10,11,12,13,14,15)$
$= (A'+B')(A'+C')$
$= (A'+B+C'+D)(A'+B+C'+D')(A'+B'+C+D)(A'+B'+C+D')($
$A'+B'+C'+D)(A'+B'+C'+D')$



```
module a2t2_somin(
input [3:0]x,
output y
    );
    wire a1,a2,a3,a4;
    not b1(a1,x[3]);
    not b2(a2,x[2]);
    not b3(a3,x[1]);
    not b4(a4,x[0]);
    wire c1,c2,c3,c4,c5,c6,c7,c8,c9,c10;
    and d1(c1,a1,a2,a3,a4);
    and d2(c2,a1,a2,a3,x[0]);
    and d3(c3,a1,a2,x[1],a4);
    and d4(c4,a1,a2,x[1],x[0]);
    and d5(c5,a1,x[2],a3,a4);
    and d6(c6,a1,x[2],a3,x[0]);
    and d7(c7,a1,x[2],x[1],a4);
    and d8(c8,a1,x[2],x[1],x[0]);
    and d9(c9,x[3],a2,a3,a4);
    and d10(c10,x[3],a2,a3,x[0]);
    or d11(y,c1,c2,c3,c4,c5,c6,c7,c8,c9,c10);
endmodule
```

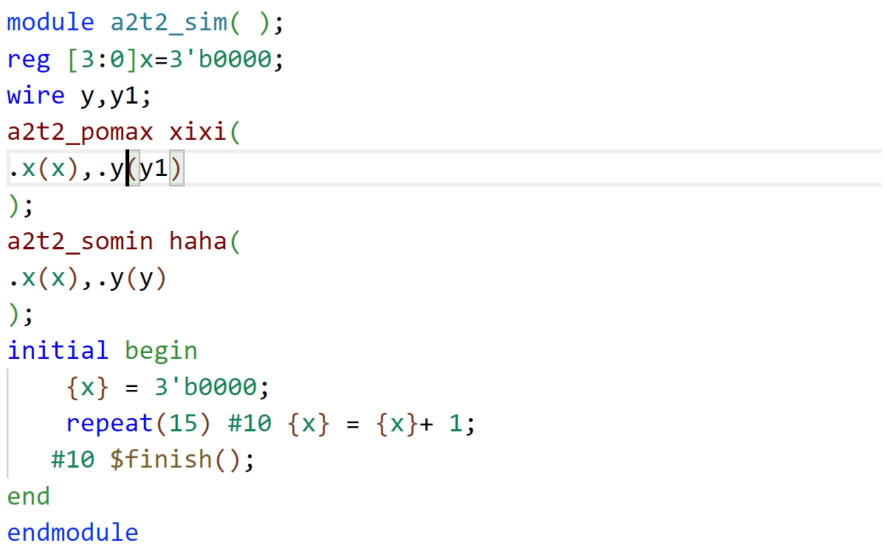


```
module a2t2_pomax(
input [3:0]x,
output y
    );
    wire a1,a2,a3;
    not b1(a1,x[3]);
    not b2(a2,x[2]);
    not b3(a3,x[1]);
    not b4(a4,x[0]);
    wire c1,c2,c3,c4,c5,c6;
    or d1(c1,a1,x[2],a3,x[0]);
    or d2(c2,a1,x[2],a3,a4);
    or d3(c3,a1,a2,x[1],x[0]);
    or d4(c4,a1,a2,x[1],a4);
    or d5(c5,a1,a2,a3,x[0]);
    or d6(c6,a1,a2,a3,a4);
    and d7(y,c1,c2,c3,c4,c5,c6);

endmodule
```

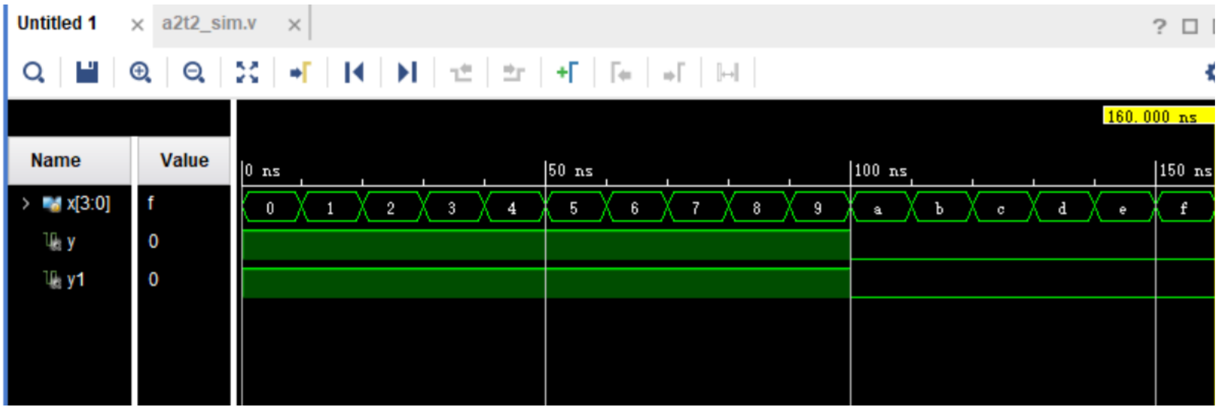
SIMULATION

Describe how you build the test bench and do the simulation.

- *Using Verilog (provide the Verilog code)*
- *Wave form of simulation result (provide screen shots)*
- *The description on whether the simulation result is same as the truth-table, is the function of the design meet the expectation*



```
module a2t2_sim( );
reg [3:0]x=3'b0000;
wire y,y1;
a2t2_pomax xixi(
.x(x),.y(y1)
);
a2t2_somin haha(
.x(x),.y(y)
);
initial begin
    {x} = 3'b0000;
    repeat(15) #10 {x} = {x}+ 1;
   #10 $finish();
end
endmodule
```



- *Through the simulation result ,we can see the result out1equals to out2 and out3 equals to out4,which proves DeMorgan of 2 bits is right and the result is right as the truth-table. we use truth-table to design this logic gate,so we can think that the function of the design meet the expectation.*

THE DESCRIPTION OF OPERATION

Describe the problem occurred while in the lab and your solution. Any suggestions are welcomed.

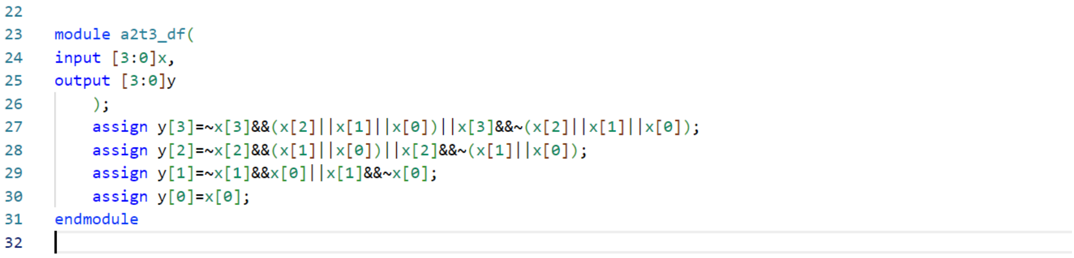
- *Problems and solutions*

PART 2: DIGITAL DESIGN LAB (TASK3)

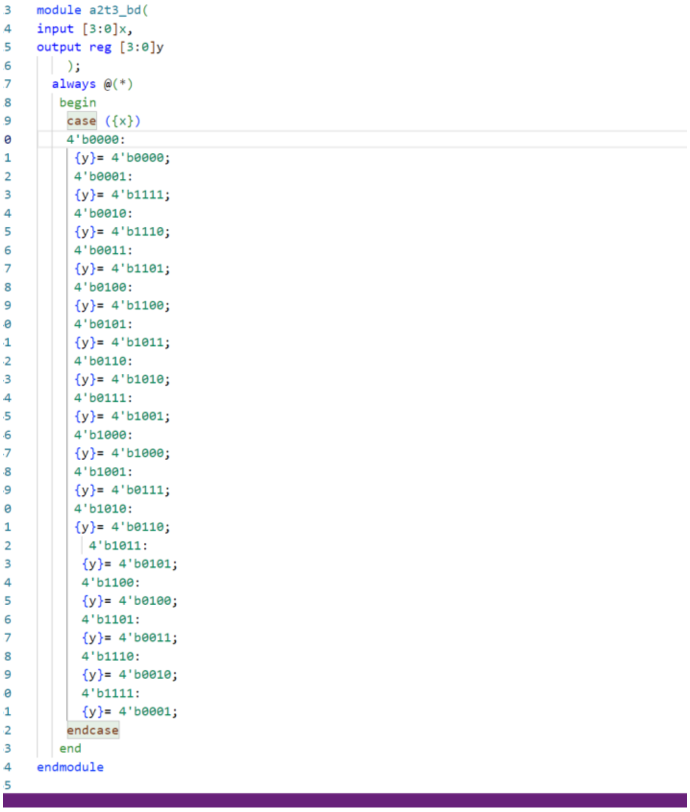
DESIGN

Describe the design of your system by providing the following information:

- *Verilog design while using data flow (provide the Verilog code)*
- *Truth-table*



```
module a2t3_df(
input [3:0]x,
output [3:0]y
    );
    assign y[3]=~x[3]&&(x[2]||x[1]||x[0])||x[3]&&~(x[2]||x[1]||x[0]);
    assign y[2]=~x[2]&&(x[1]||x[0])||x[2]&&~(x[1]||x[0]);
    assign y[1]=~x[1]&&x[0]||x[1]&&~x[0];
    assign y[0]=x[0];
endmodule
```

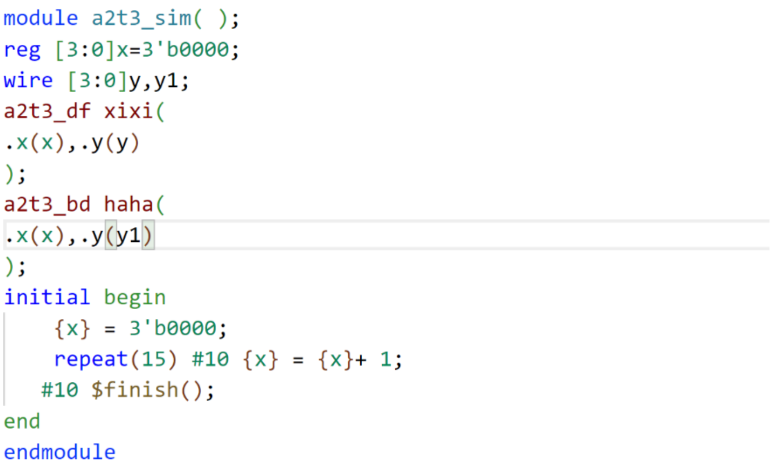


```
module a2t3_bd(
input [3:0]x,
output reg [3:0]y
    );
  always @(*)
   begin
    case ({x})
    4'b0000:
     {y}= 4'b0000;
     4'b0001:
     {y}= 4'b1111;
     4'b0010:
     {y}= 4'b1110;
     4'b0011:
     {y}= 4'b1101;
     4'b0100:
     {y}= 4'b1100;
     4'b0101:
     {y}= 4'b1011;
     4'b0110:
     {y}= 4'b1010;
     4'b0111:
     {y}= 4'b1001;
     4'b1000:
     {y}= 4'b1000;
     4'b1001:
     {y}= 4'b0111;
     4'b1010:
     {y}= 4'b0110;
       4'b1011:
      {y}= 4'b0101;
      4'b1100:
      {y}= 4'b0100;
      4'b1101:
      {y}= 4'b0011;
      4'b1110:
      {y}= 4'b0010;
      4'b1111:
      {y}= 4'b0001;
    endcase
   end
endmodule
```

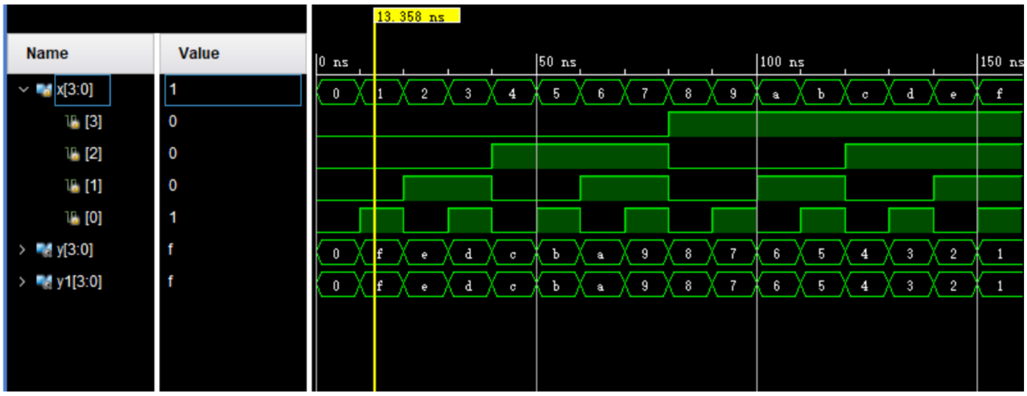
SIMULATION

Describe how you build the test bench and do the simulation.

- *Using Verilog (provide the Verilog code)*
- *Wave form of simulation result (provide screen shots)*
- *The description on whether the simulation result is same as the truth-table, is the function of the design meet the expectation*



```
module a2t3_sim( );
reg [3:0]x=3'b0000;
wire [3:0]y,y1;
a2t3_df xixi(
.x(x),.y(y)
);
a2t3_bd haha(
.x(x),.y(y1)
);
initial begin
    {x} = 3'b0000;
    repeat(15) #10 {x} = {x}+ 1;
   #10 $finish();
end
endmodule
```



```
Name        Value
x[3:0]      1
  [3]       0
  [2]       0
  [1]       0
  [0]       1
y[3:0]      f
y1[3:0]     f
```

13.358 ns
0 ns
50 ns
100 ns
150 ns
0 1 2 3 4 5 6 7 8 9 a b c d e f
0 f e d c b a 9 8 7 6 5 4 3 2 1
0 f e d c b a 9 8 7 6 5 4 3 2 1

- *Through the simulation result ,we can see the result is obviously right as the truth-table,and out1 equals to out2 and out3,which proves that three expressions are logical same. we use truth-table to design this logic gate,so we can think that the function of the design meet the expectation.*

THE DESCRIPTION OF OPERATION

Describe the problem occurred while in the lab and your solution. Any suggestions are welcomed.

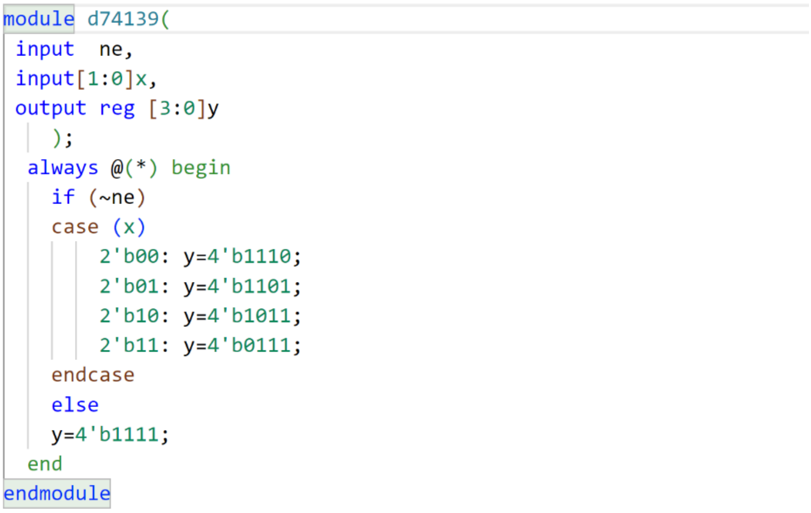
- *Problems and solutions*

PART 2: DIGITAL DESIGN LAB (TASK4)

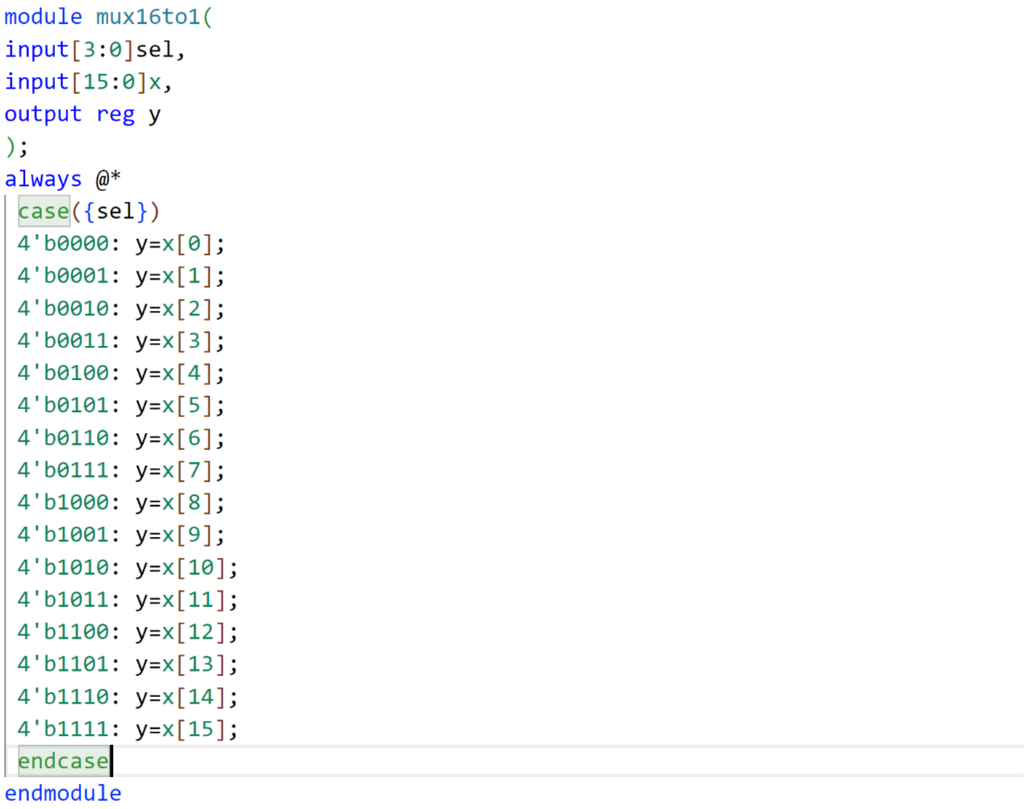
DESIGN

Describe the design of your system by providing the following information:

- *Verilog design while using data flow (provide the Verilog code)*
- *Truth-table*

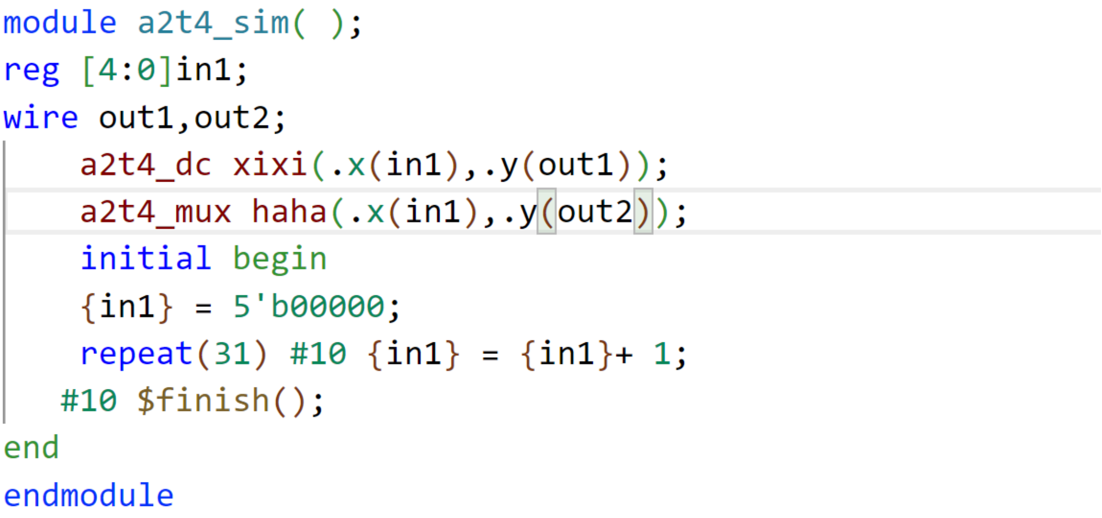


```
module d74139(
 input  ne,
 input[1:0]x,
 output reg [3:0]y
   );
  always @(*) begin
    if (~ne)
    case (x)
        2'b00: y=4'b1110;
        2'b01: y=4'b1101;
        2'b10: y=4'b1011;
        2'b11: y=4'b0111;
    endcase
    else
    y=4'b1111;
  end
endmodule
```



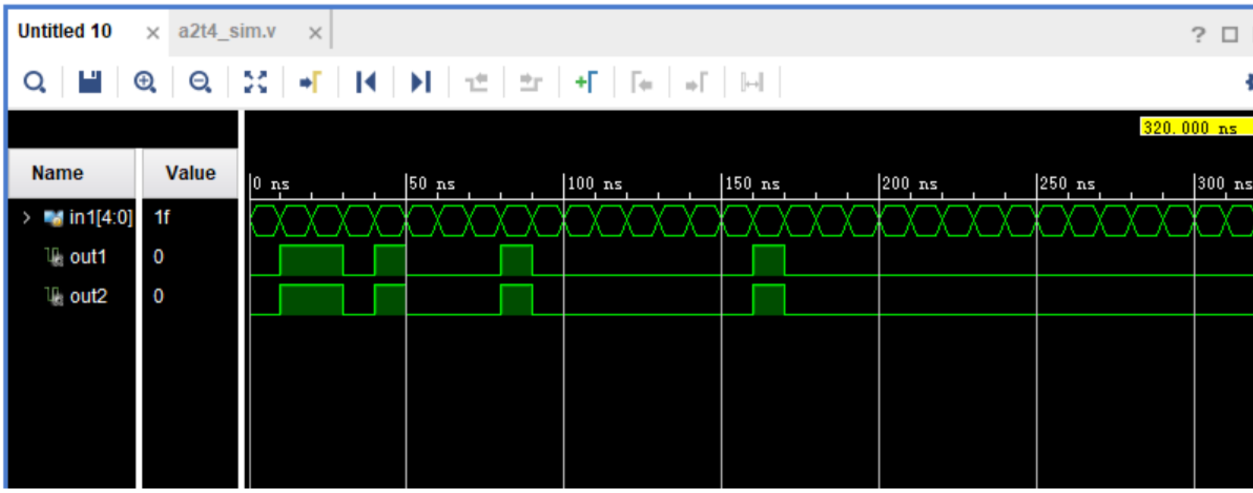
```
module mux16to1(
input[3:0]sel,
input[15:0]x,
output reg y
);
always @*
 case({sel})
 4'b0000: y=x[0];
 4'b0001: y=x[1];
 4'b0010: y=x[2];
 4'b0011: y=x[3];
 4'b0100: y=x[4];
 4'b0101: y=x[5];
 4'b0110: y=x[6];
 4'b0111: y=x[7];
 4'b1000: y=x[8];
 4'b1001: y=x[9];
 4'b1010: y=x[10];
 4'b1011: y=x[11];
 4'b1100: y=x[12];
 4'b1101: y=x[13];
 4'b1110: y=x[14];
 4'b1111: y=x[15];
 endcase
endmodule
```

- *The description on whether the simulation result is same as the truth-table, is the function of the design meet the expectation*

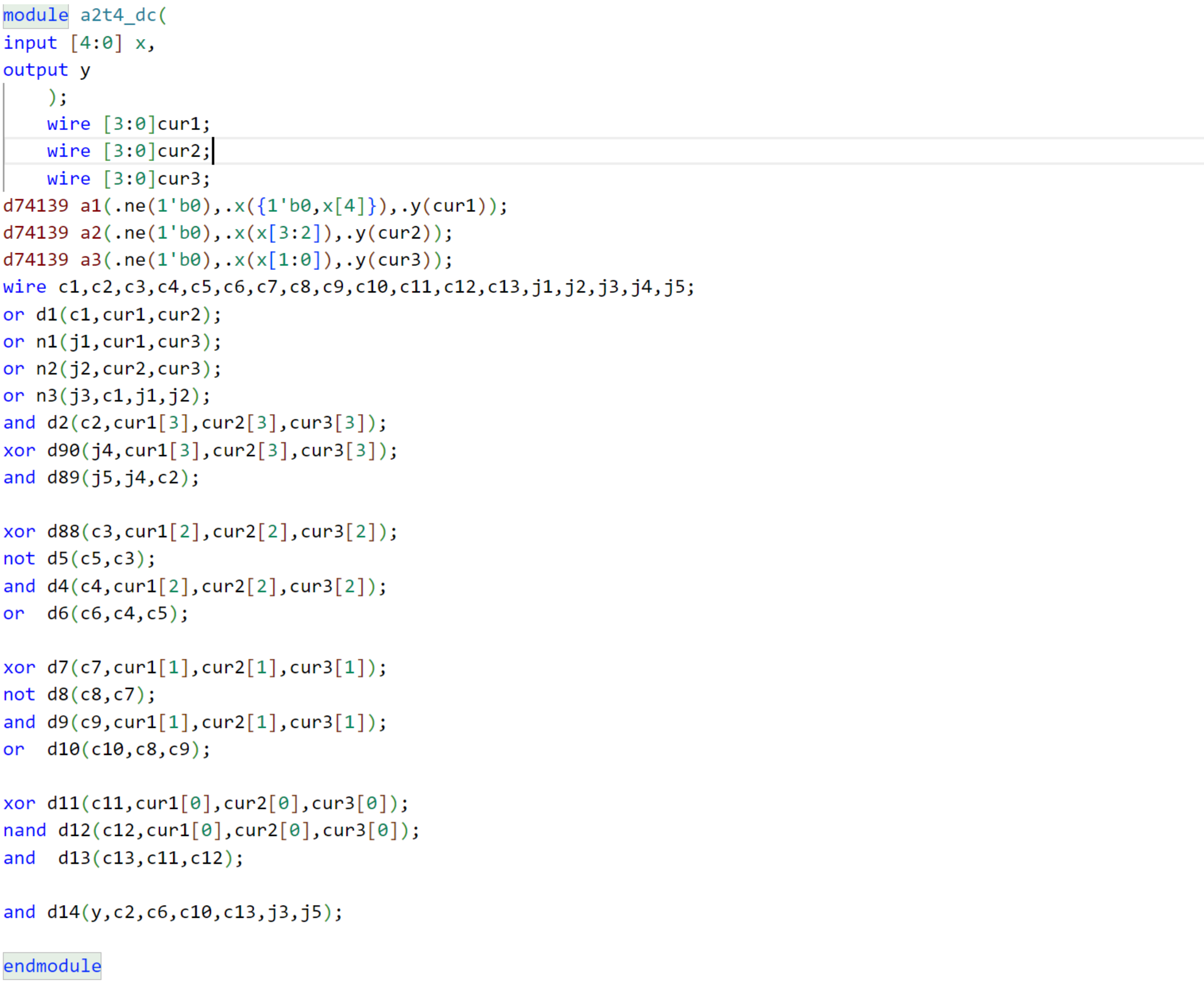


```
module a2t4_sim( );
reg [4:0]in1;
wire out1,out2;
    a2t4_dc xixi(.x(in1),.y(out1));
    a2t4_mux haha(.x(in1),.y(out2));
    initial begin
    {in1} = 5'b00000;
    repeat(31) #10 {in1} = {in1}+ 1;
   #10 $finish();
end
endmodule
```

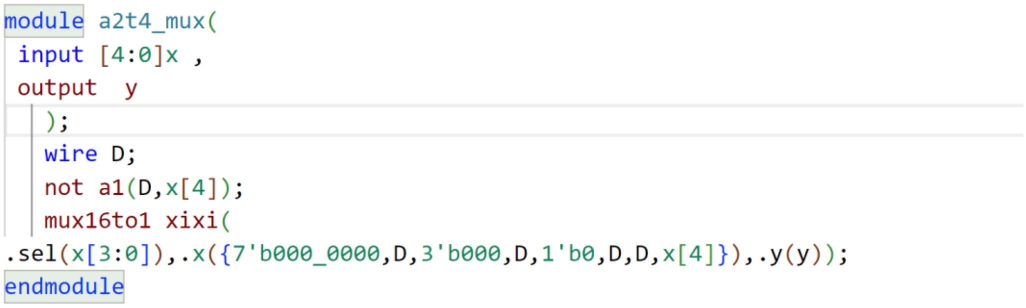
-



-



```
module a2t4_dc(
input [4:0] x,
output y
    );
    wire [3:0]cur1;
    wire [3:0]cur2;
    wire [3:0]cur3;
d74139 a1(.ne(1'b0),.x({1'b0,x[4]}),.y(cur1));
d74139 a2(.ne(1'b0),.x(x[3:2]),.y(cur2));
d74139 a3(.ne(1'b0),.x(x[1:0]),.y(cur3));
wire c1,c2,c3,c4,c5,c6,c7,c8,c9,c10,c11,c12,c13,j1,j2,j3,j4,j5;
or d1(c1,cur1,cur2);
or n1(j1,cur1,cur3);
or n2(j2,cur2,cur3);
or n3(j3,c1,j1,j2);
and d2(c2,cur1[3],cur2[3],cur3[3]);
xor d90(j4,cur1[3],cur2[3],cur3[3]);
and d89(j5,j4,c2);

xor d88(c3,cur1[2],cur2[2],cur3[2]);
not d5(c5,c3);
and d4(c4,cur1[2],cur2[2],cur3[2]);
or  d6(c6,c4,c5);

xor d7(c7,cur1[1],cur2[1],cur3[1]);
not d8(c8,c7);
and d9(c9,cur1[1],cur2[1],cur3[1]);
or  d10(c10,c8,c9);

xor d11(c11,cur1[0],cur2[0],cur3[0]);
nand d12(c12,cur1[0],cur2[0],cur3[0]);
and  d13(c13,c11,c12);

and d14(y,c2,c6,c10,c13,j3,j5);

endmodule
```

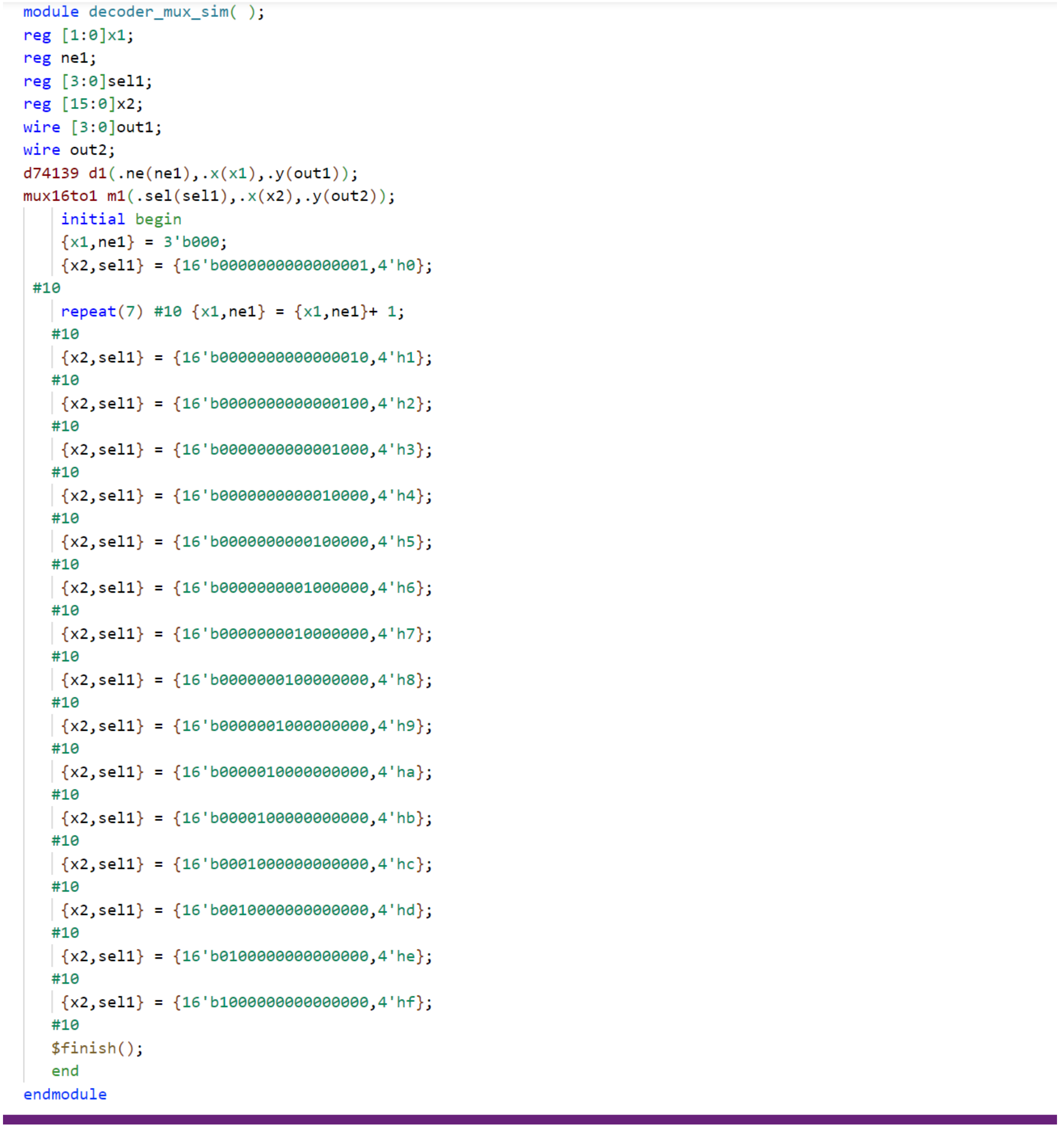


```
module a2t4_mux(
 input [4:0]x ,
 output  y
    );
    wire D;
    not a1(D,x[4]);
    mux16to1 xixi(
.sel(x[3:0]),.x({7'b000_0000,D,3'b000,D,1'b0,D,D,x[4]}),.y(y));
endmodule
```

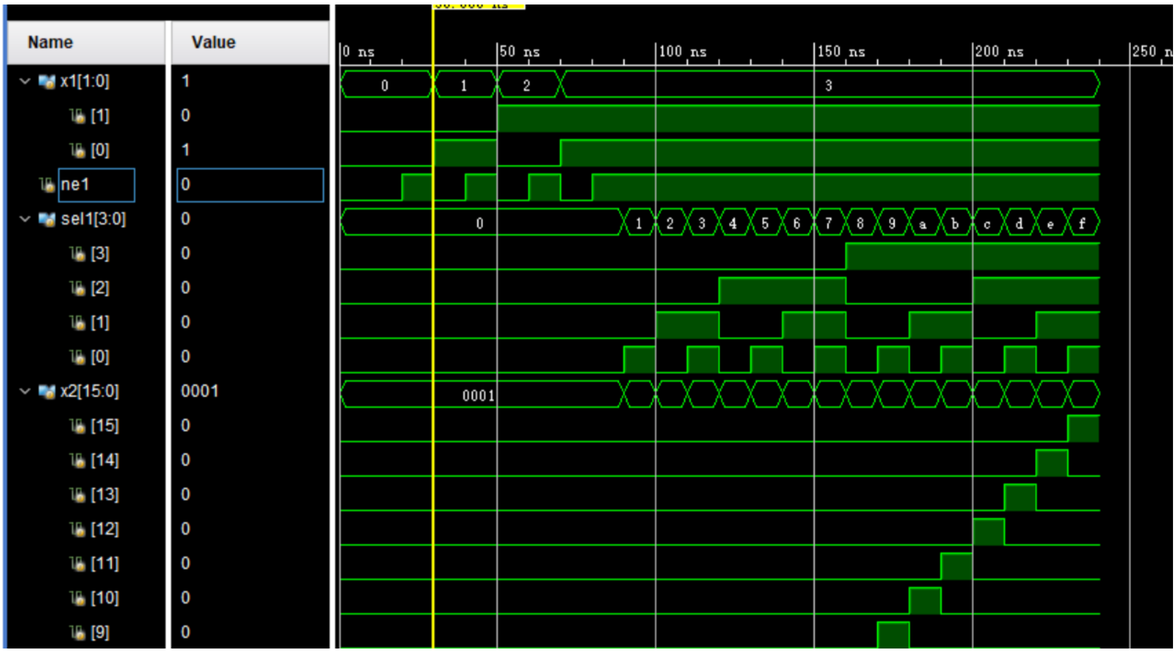
SIMULATION

Describe how you build the test bench and do the simulation.

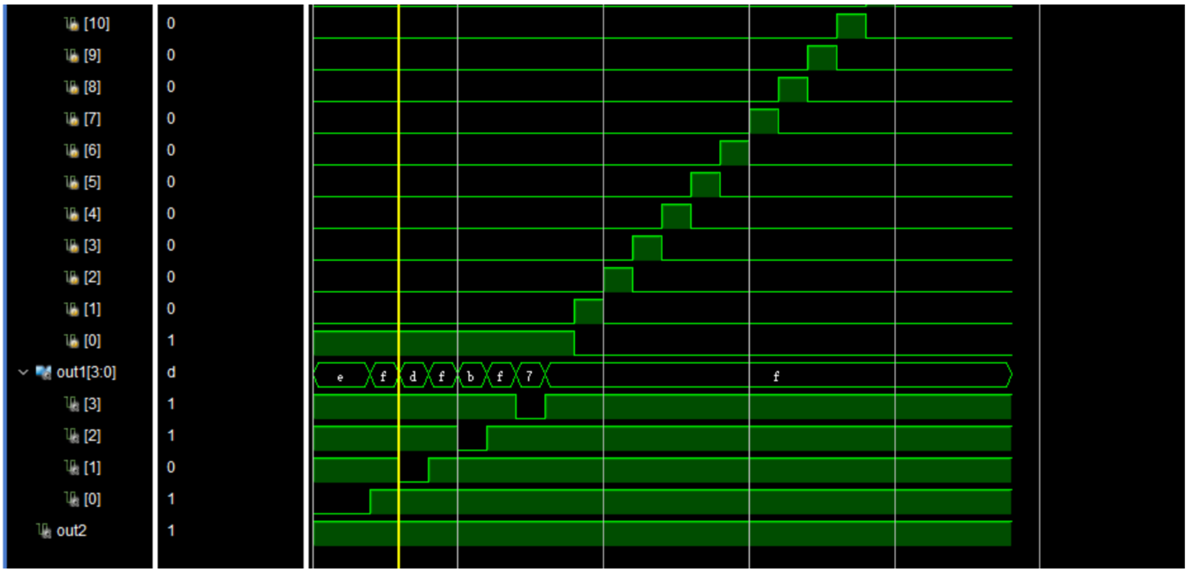
- *Using Verilog (provide the Verilog code)*
- *Wave form of simulation result (provide screen shots)*



```
module decoder_mux_sim( );
reg [1:0]x1;
reg ne1;
reg [3:0]sel1;
reg [15:0]x2;
wire [3:0]out1;
wire out2;
d74139 d1(.ne(ne1),.x(x1),.y(out1));
mux16to1 m1(.sel(sel1),.x(x2),.y(out2));
    initial begin
    {x1,ne1} = 3'b000;
    {x2,sel1} = {16'b0000000000000001,4'h0};
 #10
    repeat(7) #10 {x1,ne1} = {x1,ne1}+ 1;
   #10
    {x2,sel1} = {16'b0000000000000010,4'h1};
   #10
    {x2,sel1} = {16'b0000000000000100,4'h2};
   #10
    {x2,sel1} = {16'b0000000000001000,4'h3};
   #10
    {x2,sel1} = {16'b0000000000010000,4'h4};
   #10
    {x2,sel1} = {16'b0000000000100000,4'h5};
   #10
    {x2,sel1} = {16'b0000000001000000,4'h6};
   #10
    {x2,sel1} = {16'b0000000010000000,4'h7};
   #10
    {x2,sel1} = {16'b0000000100000000,4'h8};
   #10
    {x2,sel1} = {16'b0000001000000000,4'h9};
   #10
    {x2,sel1} = {16'b0000010000000000,4'ha};
   #10
    {x2,sel1} = {16'b0000100000000000,4'hb};
   #10
    {x2,sel1} = {16'b0001000000000000,4'hc};
   #10
    {x2,sel1} = {16'b0010000000000000,4'hd};
   #10
    {x2,sel1} = {16'b0100000000000000,4'he};
   #10
    {x2,sel1} = {16'b1000000000000000,4'hf};
   #10
   $finish();
   end
endmodule
```



Name	Value
x1[1:0]	1
[1]	0
[0]	1
ne1	0
sel1[3:0]	0
[3]	0
[2]	0
[1]	0
[0]	0
x2[15:0]	0001
[15]	0
[14]	0
[13]	0
[12]	0
[11]	0
[10]	0
[9]	0



Name	Value
[10]	0
[9]	0
[8]	0
[7]	0
[6]	0
[5]	0
[4]	0
[3]	0
[2]	0
[1]	0
[0]	1
out1[3:0]	d
[3]	1
[2]	1
[1]	0
[0]	1
out2	1

- *Through the simulation result ,we can see the result is obviously right as the truth-table,and out1 equals to out2 and out3,which proves that three expressions are logical same. we use truth-table to design this logic gate,so we can think that the function of the design meet the expectation.*

THE DESCRIPTION OF OPERATION

Describe the problem occurred while in the lab and your solution. Any suggestions are welcomed.

- *Problems and solutions*