DIGITAL DESIGN

ASSIGNMENTREPORT

ASSIGNMENT ID : III

Student Name: 郑鑫颖

Student ID: 11912039

PART 1: DIGITAL DESIGN THEORY

Provide your answers here:

Digital Design Assignment III

1: $F(A,B,C,D)=\Sigma(1,3,5,8,9,10,11,13,15)$

AB \ CD	00	01	11	10
00	0	1 1	3 1	2
01	4	5 1	7	6
11	8 12	13 1	15 1	14
10	8 1	9 1	11 1	10 1

(a) AND-OR

$F=C'D+AD+B'D+AB'$

(b) OR-AND

$F=(A+D)\cdot(B'+D)\cdot(A+B'+C')$

(c) NOR-OR

$F=(C+D')'+(A'+D')'+(B+D')'+(A'+B)'$

(d) NOR-NOR

$F=[(A+D)'+(B'+D)'+(A+B'+C')']'$

(e) NAND-NAND

$F=[(C'D)'\cdot(AD)'\cdot(B'D)'\cdot(AB')']'$

(f) NAND-AND

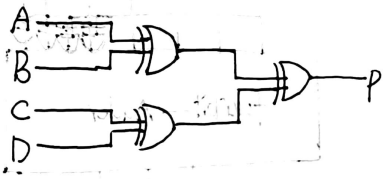
$F=(A'D')'\cdot(BD')'\cdot(A'BC)'$

2: parity generator

A	B	C	D	P
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

$P=\Sigma(1,2,4,7,8,11,13,14)$

$=A\oplus B\oplus C\oplus D$



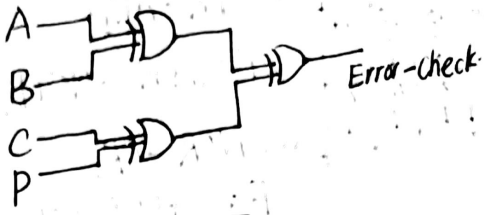
0	1 V	3	2 V
4 V	5	7 V	6
12	13 V	15	14 V
8 V	9	11 V	10

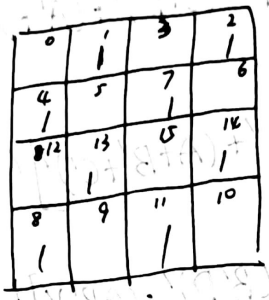
Parity Checker

A	B	C	P	Error-check
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Error-Check = $\Sigma(1,2,4,7,8,11,13,14)$.

$= A \oplus B \oplus C \oplus P$





3.

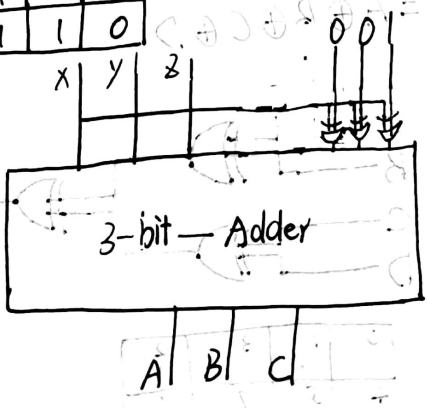
	X	Y	Z	A	B	C
0	0	0	0	0	0	1
1	0	0	1	0	1	0
2	0	1	0	0	1	1
3	0	1	1	1	0	0
4	1	0	0	0	1	1
5	1	0	1	1	0	0
6	1	1	0	1	0	1
7	1	1	1	1	1	0

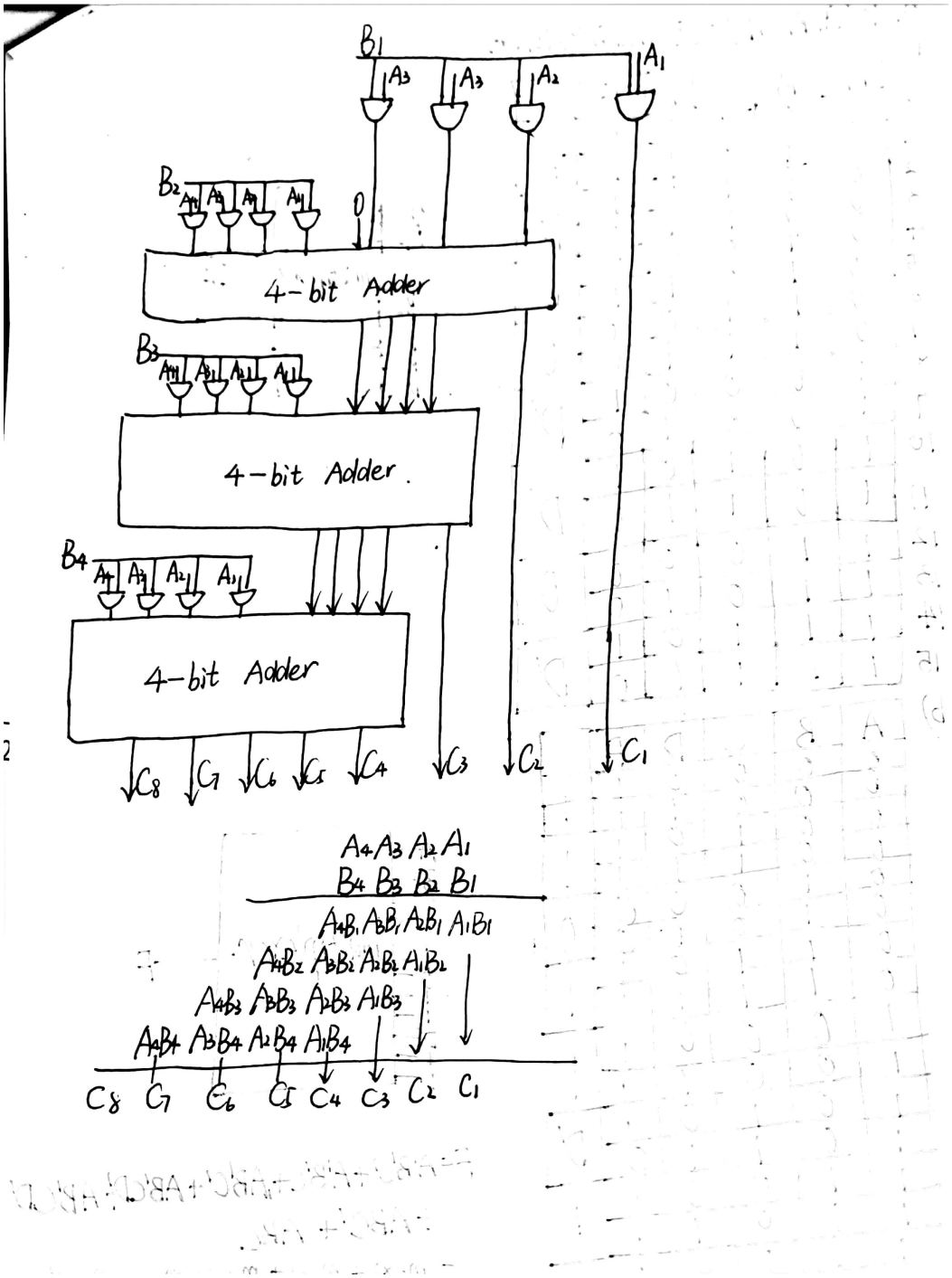
input / output

$A = \Sigma(3,5,6,7) = xz + xy + yz$

$B = \Sigma(1,2,4,7) = x \oplus y \oplus z$

$C = \Sigma(0,2,4,6) = z'$

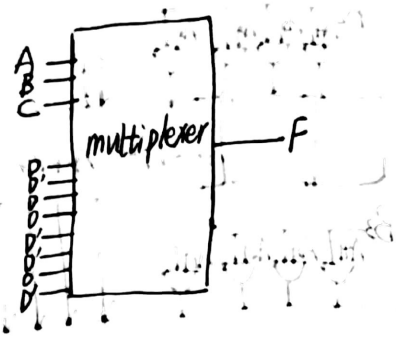



B1
A1
A3
A3
A2
B2
0
4-bit Adder
B3
4-bit Adder
B4
4-bit Adder
C8
C7
C6
C5
C4
C3
C2
C1
A4 A3 A2 A1
B4 B3 B2 B1
A4B1 A3B1 A2B1 A1B1
A4B2 A3B2 A2B2 A1B2
A4B3 A3B3 A2B3 A1B3
A4B4 A3B4 A2B4 A1B4
C8 C7 C6 C5 C4 C3 C2 C1

5. $F(A, B, C, D) = \Sigma(1, 2, 5, 8, 10, 14)$

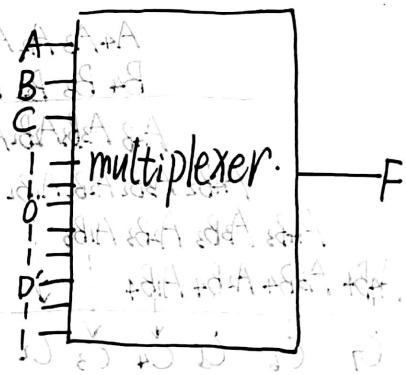
(a)

	A	B	C	D	F	
0	0	0	0	0	0	D
1	0	0	0	1	1	
2	0	0	1	0	1	D'
3	0	0	1	1	0	
4	0	1	0	0	0	D
5	0	1	0	1	1	
6	0	1	1	0	0	0
7	0	1	1	1	0	
8	1	0	0	0	1	D'
9	1	0	0	1	0	
10	1	0	1	0	1	D'
11	1	0	1	1	0	
12	1	1	0	0	0	0
13	1	1	0	1	0	
14	1	1	1	0	1	D'
15	1	1	1	1	0	



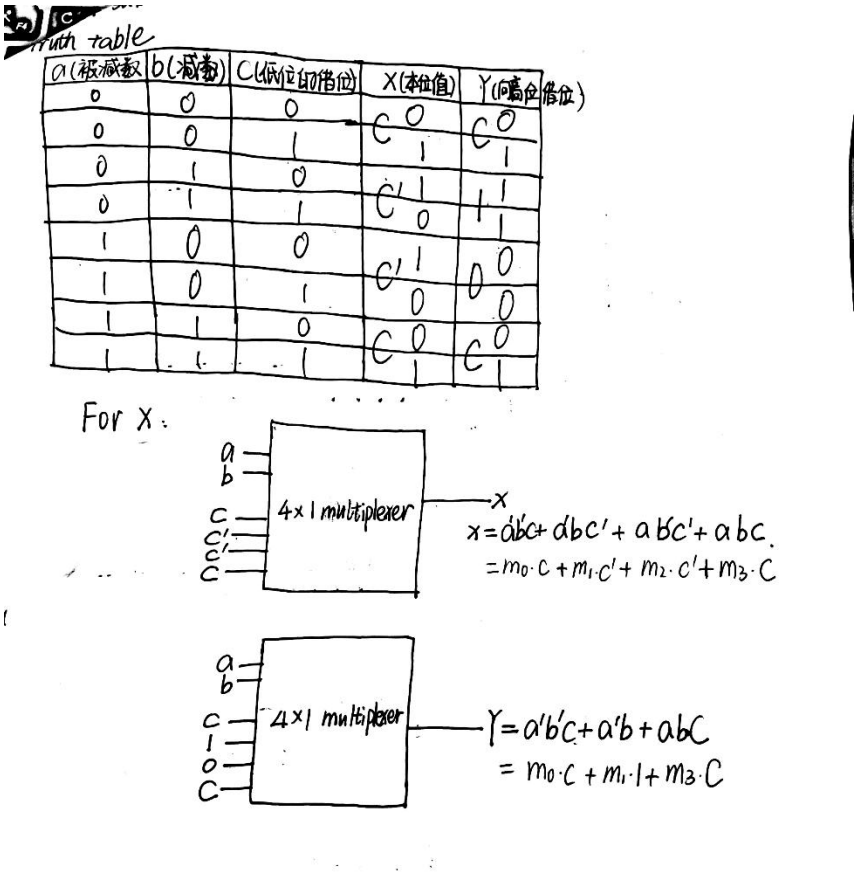
(b)

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



$F = A'B'C' + A'B'C + A'BC + AB'C' + AB'CD' + ABC' + ABC$

$= m_0 \times 1 + m_1 \times 1 + m_3 \times 1 + m_4 \times 1 + m_5 \times D' + m_6 \times 1 + m_7 \times 1$



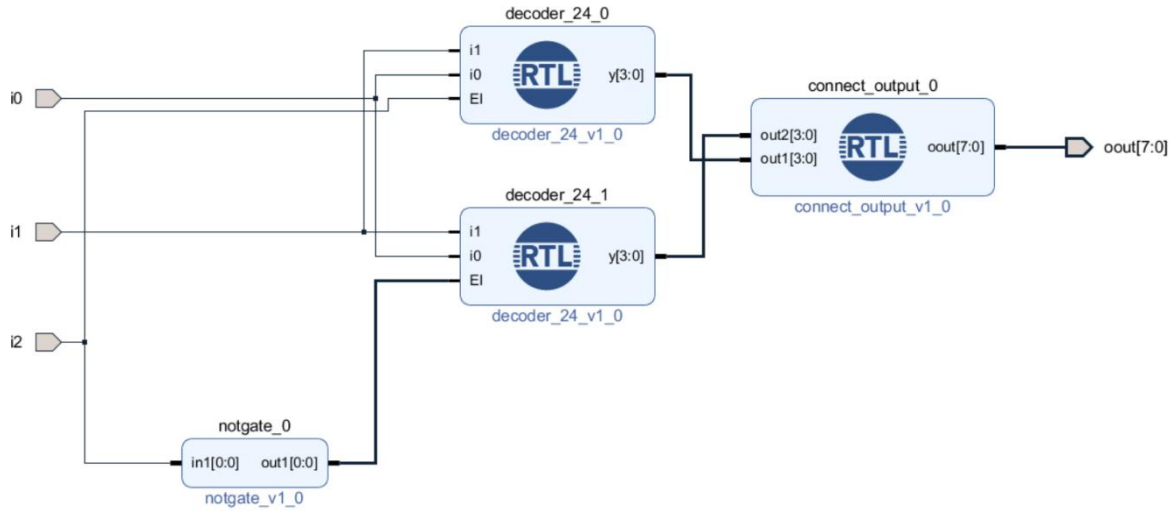
PART 2: DIGITAL DESIGN LAB (TASK1)

DESIGN

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

3-8 decoder

- *Verilog design (provide the Verilog code)*



2-4 decoder

```
module decoder_24(
    input i1,
    input i0,
    input EI,
    output reg[3:0] y
    );
    always@*
    begin
    if(~EI)
    casex({i1,i0})
        2'b00:y=4'b0001;
        2'b01:y=4'b0010;
        2'b10:y=4'b0100;
        2'b11:y=4'b1000;
        endcase
    else y=4'b0000;
    end
endmodule
```

- *Truth-table for*

i2	*i1*	*i0*	*oout[7]*	*oout[6]*	*oout[5]*	*oout[4]*	*oout[3]*	*oout[2]*	*oout[1]*	*oout[0]*
0	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*
0	*0*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*0*
0	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*0*	*0*

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

4-16 decoder

```
module decoder4_16(
    input i0,
    input i1,
    input i2,
    input i3,
    output reg [15:0] out
    );

    wire[15:0] out0;
    decoder_38_wrapper v(i0,i1,i2,{out0[7],out0[6],out0[5],out0[4],out0[3],out0[2],out0[1],out0[0]});

    always @* begin
    if(~i3)
    out={8'b0,out0[7],out0[6],out0[5],out0[4],out0[3],out0[2],out0[1],out0[0]};
    else
    out={out0[7],out0[6],out0[5],out0[4],out0[3],out0[2],out0[1],out0[0],8'b0};
    end

endmodule
```

Truth-table for4-16

I0	I1	I2	I3	Out0[15]	Out0[14]	Out0[13]	Out0[12]	Out0[11]	Out0[10]	Out0[9]	Out0[8]	Out0[7]	Out0[6]	Out0[5]	Out0[4]	Out0[3]	Out0[2]	Out0[1]	Out0[0]
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

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

SIMULATION

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

- *Using Verilog (provide the Verilog code)*

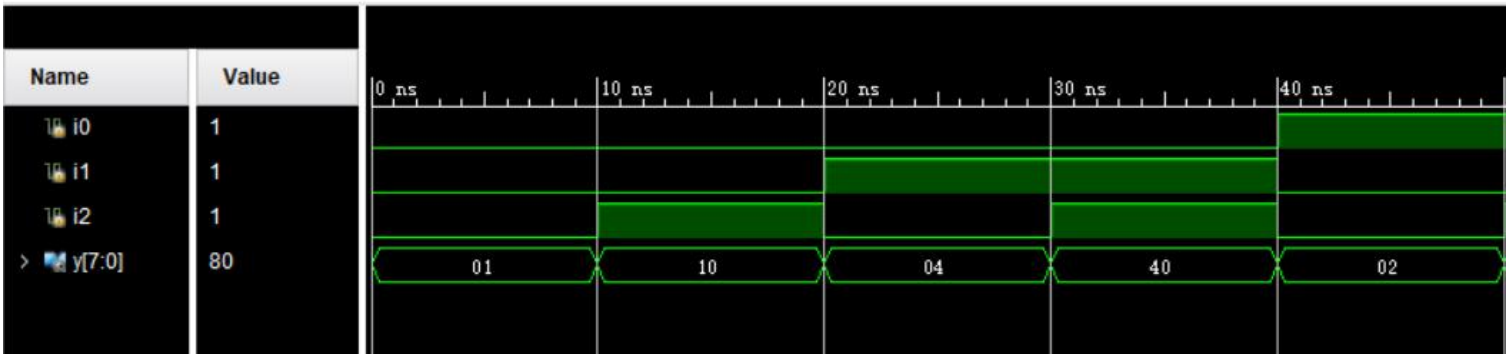
  3-8decoder

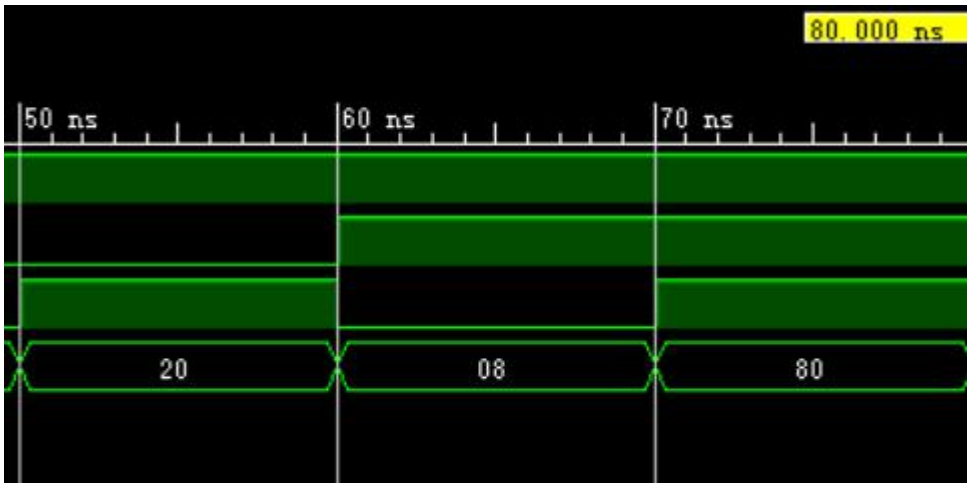
```
module decoder_sim();
  reg i0;
  reg i1;
  reg i2;
  wire[7:0] y;

 decoder_38_wrapper v(i0,
     i1,
     i2,
    y);
   /* decoder4_16 v(i0,i1,i2,i3,y);*/

  initial begin
  {i0,i1,i2}=0;
  repeat(7) #10 {i0,i1,i2}={i0,i1,i2}+1;
  #10 $finish;
  end

endmodule
```

4-16decoder

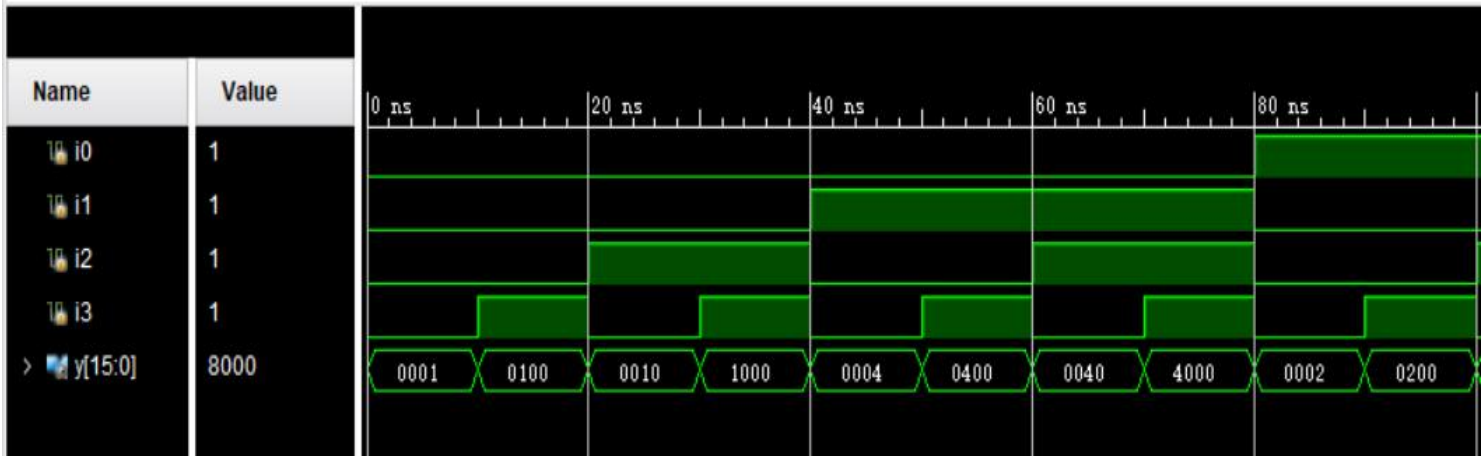
```
module decoder_sim();
  reg i0;
  reg i1;
  reg i2;
  reg i3;
  wire[15:0] y;

 /*decoder_38_wrapper v(i0,
     i1,
     i2,
    y);*/
    decoder4_16 v(i0,i1,i2,i3,y);

  initial begin
  {i0,i1,i2,i3}=0;
  repeat(15) #10{i0,i1,i2,i3}={i0,i1,i2,i3}+1;
  #10 $finish;
  end

endmodule
```

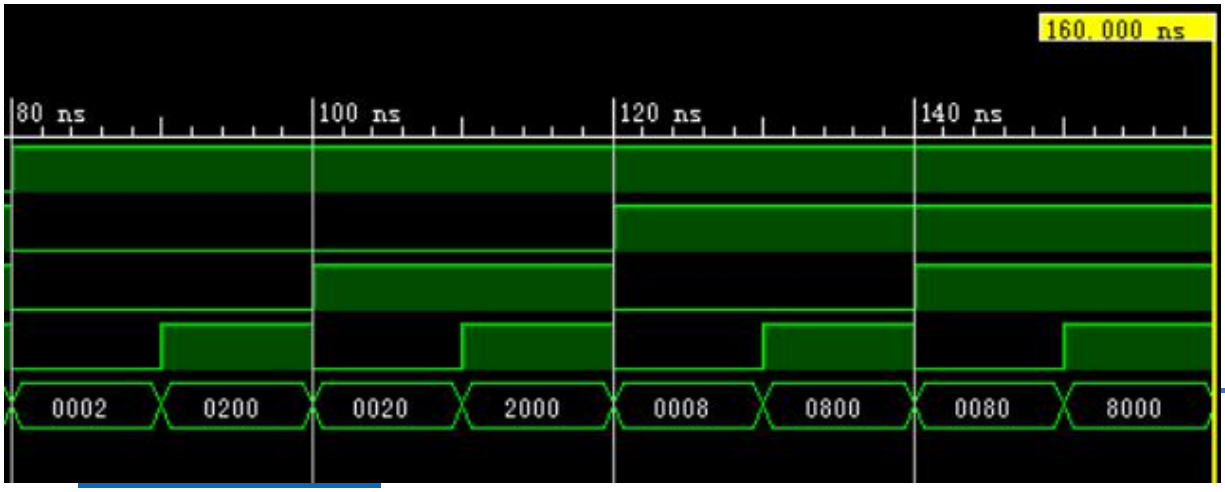
- *Wave form of simulation result (provide screen shots)*
- *3-8decoder*


Name
Value
i0
1
i1
1
i2
1
y[7:0]
80
0 ns
10 ns
20 ns
30 ns
40 ns
01
10
04
40
02


80.000 ns
50 ns
60 ns
70 ns
20
08
80

- *4-16decoder*


Name
Value
i0
1
i1
1
i2
1
i3
1
y[15:0]
8000
0 ns
20 ns
40 ns
60 ns
80 ns
0001
0100
0010
1000
0004
0400
0040
4000
0002
0200


160.000 ns
80 ns
100 ns
120 ns
140 ns
0002
0200
0020
2000
0008
0800
0080
8000

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

  the simulation result is same as the truth-table and meet the expectation

CONSTRAINT FILE AND THE TESTING

Describe how you test your design on the Minisys Practice platform.

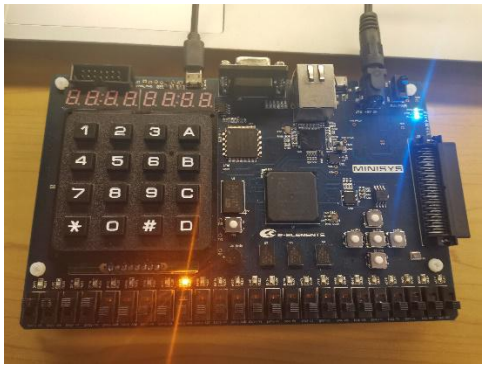
- *Constraint file (provide the screen shots on the feature of a pin and the binding info between pins and the input /output ports)*

```
set_property IOSTANDARD LVCMOS33 [get_ports {i0}]
set_property IOSTANDARD LVCMOS33 [get_ports {i1}]
set_property IOSTANDARD LVCMOS33 [get_ports {i2}]
set_property IOSTANDARD LVCMOS33 [get_ports {i3}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[8]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[9]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[10]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[11]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[12]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[13]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[14]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[15]}]
set_property PACKAGE_PIN Y8 [get_ports {i0}]
set_property PACKAGE_PIN Y7 [get_ports {i1}]
set_property PACKAGE_PIN W9 [get_ports {i2}]
set_property PACKAGE_PIN Y9 [get_ports {i3}]
set_property PACKAGE_PIN A21 [get_ports {out[0]}]
set_property PACKAGE_PIN E22 [get_ports {out[1]}]
set_property PACKAGE_PIN D22 [get_ports {out[2]}]
```

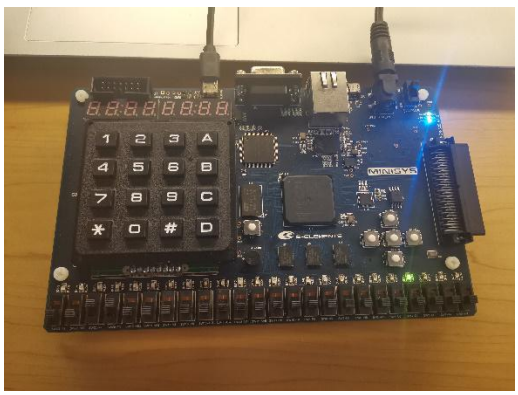
```
set_property PACKAGE_PIN E21 [get_ports {out[3]}]
set_property PACKAGE_PIN D21 [get_ports {out[4]}]
set_property PACKAGE_PIN G21 [get_ports {out[5]}]
set_property PACKAGE_PIN G22 [get_ports {out[6]}]
set_property PACKAGE_PIN F21 [get_ports {out[7]}]
set_property PACKAGE_PIN J17 [get_ports {out[8]}]
set_property PACKAGE_PIN L14 [get_ports {out[9]}]
set_property PACKAGE_PIN L15 [get_ports {out[10]}]
set_property PACKAGE_PIN L16 [get_ports {out[11]}]
set_property PACKAGE_PIN K16 [get_ports {out[12]}]
set_property PACKAGE_PIN M15 [get_ports {out[13]}]
set_property PACKAGE_PIN M16 [get_ports {out[14]}]
set_property PACKAGE_PIN M17 [get_ports {out[15]}]
```

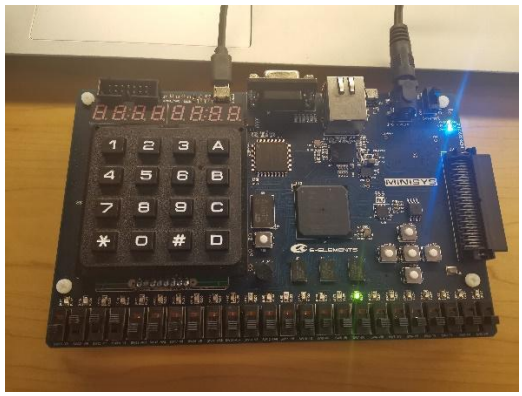
- *The testing result (provide the screen shots (at least 3 testing scene) to show state of inputs and outputs along with the related descriptions.*

15 *0*



3 *7*





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 data flow (provide the Verilog code)*

  Design of 74151

```
module mutiplexer74151(
    input EN,
    input s2,
    input s1,
    input s0,
    input d7,
    input d6,
    input d5,
    input d4,
    input d3,
    input d2,
    input d1,
    input d0,
    output reg y,
    output  w
    );
    always@*
    if(~EN)
    case({s2,s1,s0})
        3'b000:y=d0;
        3'b001:y=d1;
        3'b010:y=d2;
        3'b011:y=d3;
        3'b100:y=d4;
        3'b101:y=d5;
        3'b110:y=d6;
        3'b111:y=d7;
```

```
        endcase
    else
        y=1'b0;

    assign w=~y;
endmodule
```

Truth table

EN	S2	S1	S0	y
0	0	0	0	d0
0	0	0	1	d1
0	0	1	0	d2
0	0	1	1	d3
0	1	0	0	d4
0	1	0	1	d5
0	1	1	0	d6
0	1	1	1	d7
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0

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

Function design

```
module functions(input A,B,C,D,output y1,output y2);
    wire w;
    mutiplexer74151 u(1'b0,A,B,C,D,D,D,1'b0,D,D,1'b0,1'b1,y1,w);

    assign y2=(~A&~B&~C&~D)|(B&~C&D)|(~A&~C&D)|(~A&B&C&D)|(A&C&D);


endmodule
```

Truth-table

A	B	C	D	Y1	Y2
0	0	0	0	1	1
0	0	0	1	1	1
0	0	1	0	0	0
0	0	1	1	0	0
0	1	0	0	0	0

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

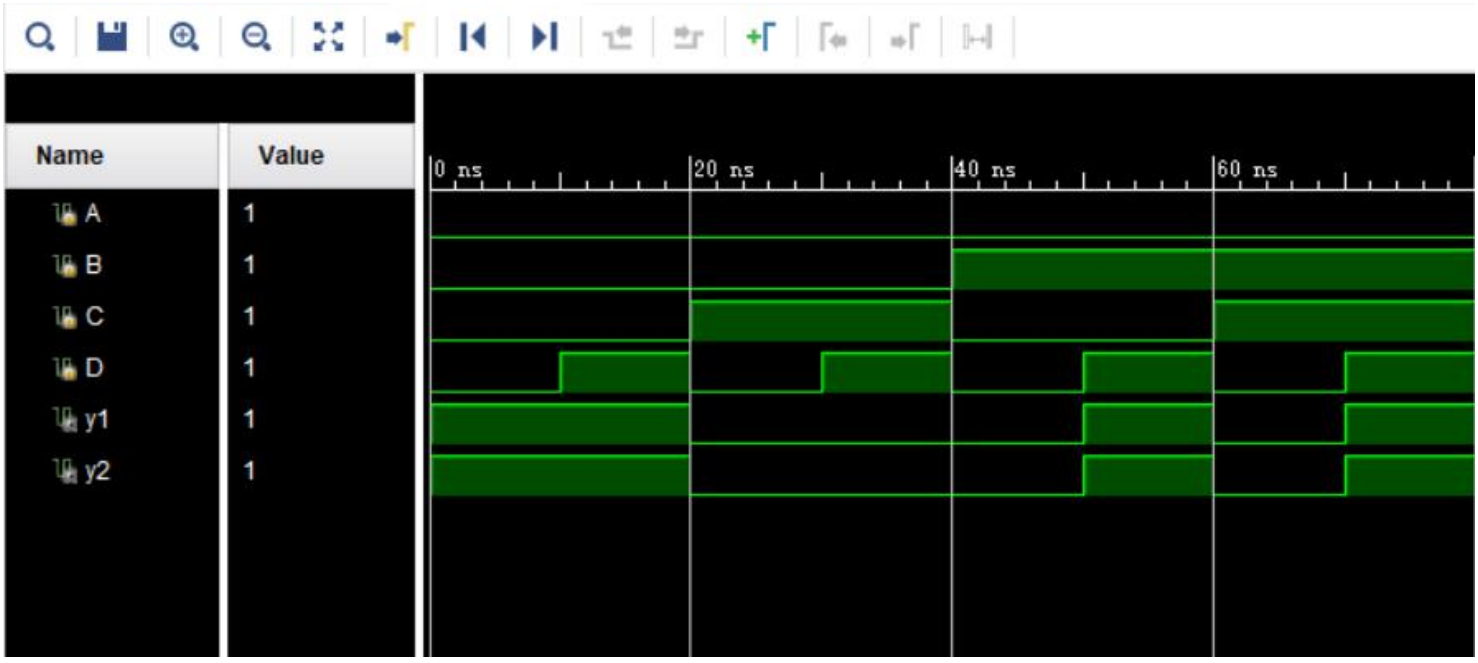
SIMULATION

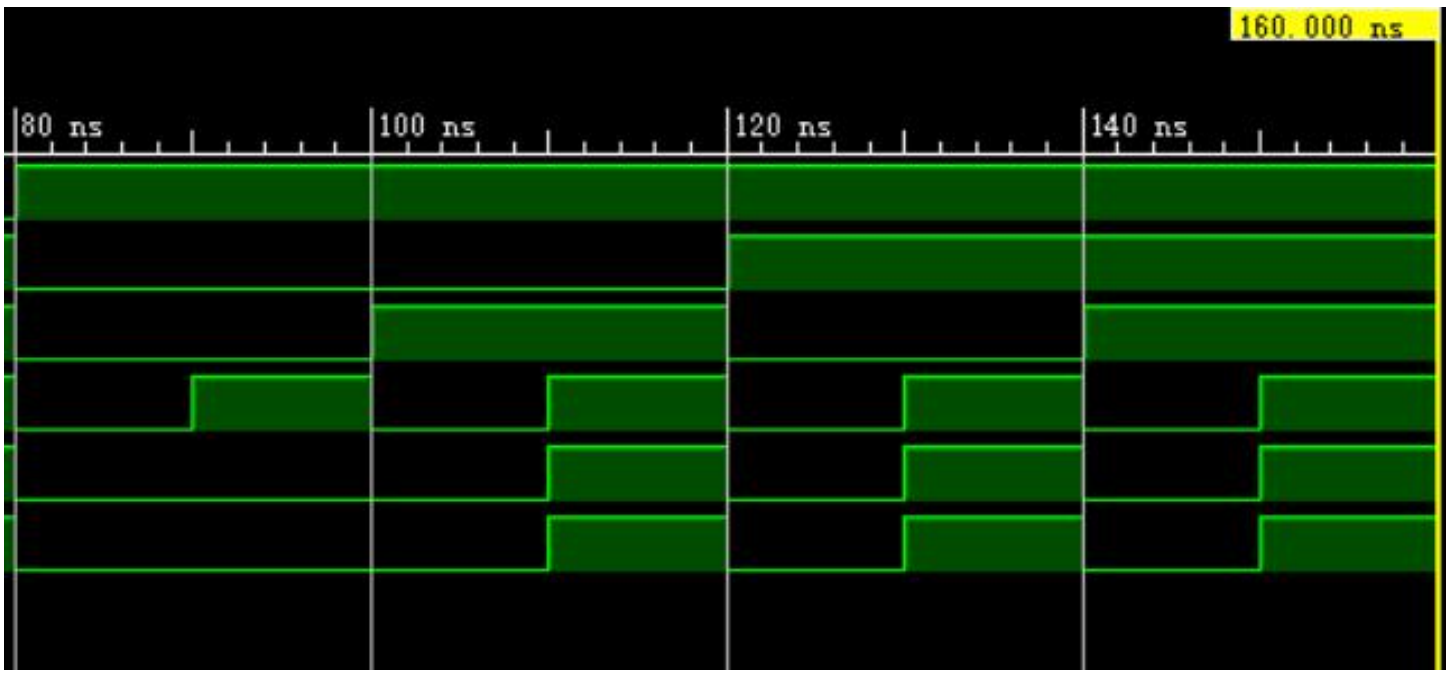
Describe how you build the test bench and do the simulation.

- *Using Verilog (provide the Verilog code)*

```
module funsim();
reg A,B,C,D;
wire y1,y2;

functions f(A,B,C,D,y1,y2);

initial begin
{A,B,C,D}=4'b0;
repeat(15) #10 {A,B,C,D}={A,B,C,D}+1;
#10 $finish;
end
endmodule
```

- *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*

  the simulation result is same as the truth-table, the function of the design meet the expectation.

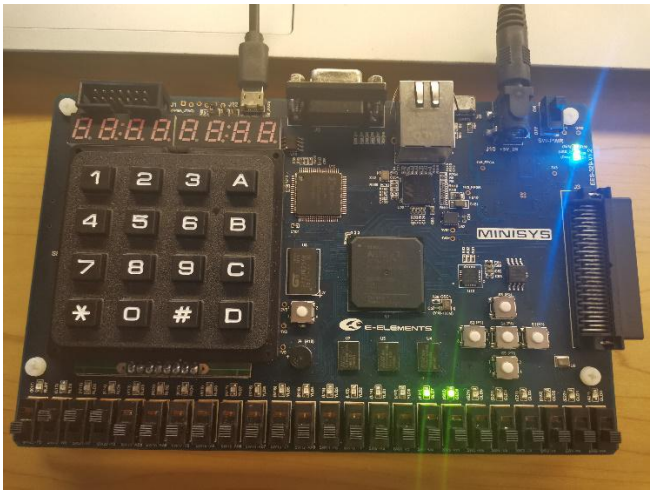
CONSTRAINT FILE AND THE TESTING

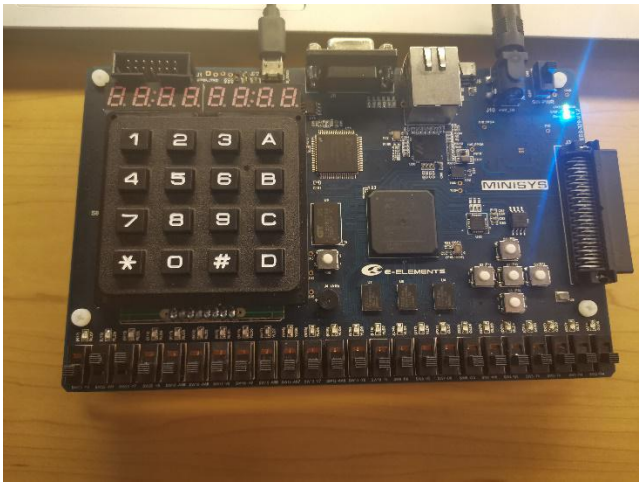
Describe how you test your design on the Minisys Practice platform.

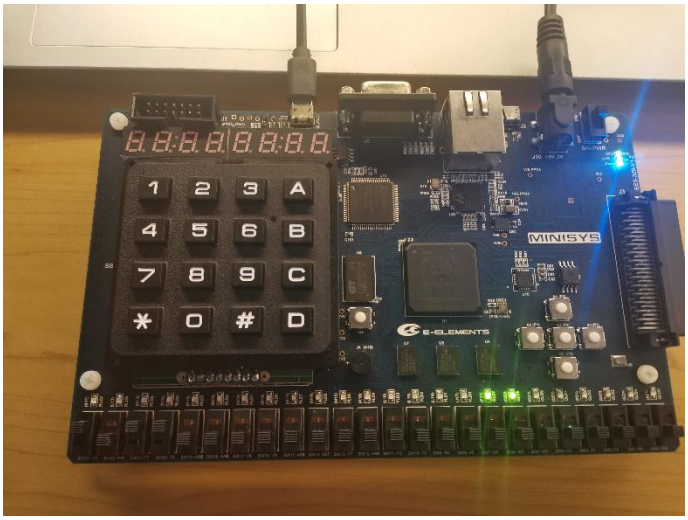
- *Constraint file (provide the screen shots on the feature of a pin and the binding info between pins and the input /output ports)*

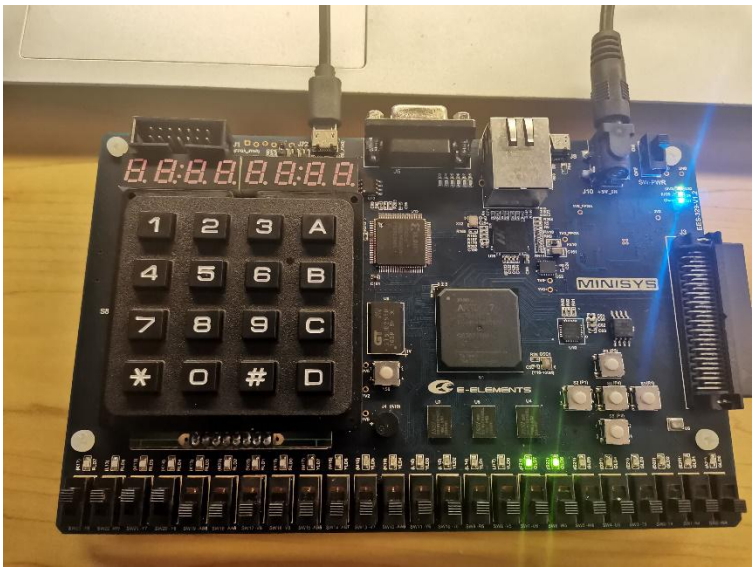
```
set_property IOSTANDARD LVCMOS33 [get_ports {A}]
set_property IOSTANDARD LVCMOS33 [get_ports {B}]
set_property IOSTANDARD LVCMOS33 [get_ports {C}]
set_property IOSTANDARD LVCMOS33 [get_ports {D}]
set_property IOSTANDARD LVCMOS33 [get_ports {y1}]
set_property IOSTANDARD LVCMOS33 [get_ports {y2}]
set_property PACKAGE_PIN Y9 [get_ports {A}]
set_property PACKAGE_PIN W9 [get_ports {B}]
set_property PACKAGE_PIN Y7 [get_ports {C}]
set_property PACKAGE_PIN Y8 [get_ports {D}]
set_property PACKAGE_PIN G22 [get_ports {y1}]
set_property PACKAGE_PIN F21 [get_ports {y2}]
```

- *The testing result (provide the screen shots (at least 3 testing scene)) to show state of inputs and outputs along with the related descriptions.*


1 2 3 A
4 5 6 B
7 8 9 C
* 0 # D
MINISYS
E-ELEMENTS


1 2 3 A
4 5 6 B
7 8 9 C
* 0 # D
MINISYS
E-ELEMENTS


1 2 3 A
4 5 6 B
7 8 9 C
* 0 # D
MINISYS
E-ELEMENTS



THE DESCRIPTION OF OPERATION

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

- *Problems and solutions*