DIGITAL DESIGN

ASSIGNMENT REPORT

ASSIGNMENT ID: 2

Student Name: 张嘉浩

Student ID: 12010423

PART 1: DIGITAL DESIGN THEORY

Provide your answers here:

张嘉港 12010423

1. (a) $F_1=\sum m_i$, $F_2=\sum m_j$, $F=F_1+F_2$

$$F=F_1+F_2=\sum m_i+\sum m_j=\sum(m_i+m_j)$$

(b) $G=F_1F_2=\sum m_i \sum m_j$

$$m_i m_j=\begin{cases}0, & i\neq j\\ 1, & i=j.\end{cases}$$

thus only common minterms.

2. (a)

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

$$F(x,y,z)=\sum(3,5,7)=m_3+m_5+m_7$$

~~$F=z'+x'y'$~~ ~~$F=z(x+y)$~~ $F'(x,y,z)=\sum(0,1,2,4,6)$

$=m_0+m_1+m_2+m_4+m_6$.

~~$F(A,B,C)$~~ $F(x,y,z)=(F'(x,y,z))'=(m_0+m_1+m_2+m_4+m_6)'$

$=m_0'm_1'm_2'm_4'm_6'$

$=M_0M_1M_2M_4M_6$

$=\Pi(0,1,2,4,6)$

(b) $F(A,B,C,D)=\Pi(3,5,8,11,12,15)$

$=M_3M_5M_8M_{11}M_{12}M_{15}$

$=m_3'm_5'm_8'm_{11}'m_{12}'m_{15}'$

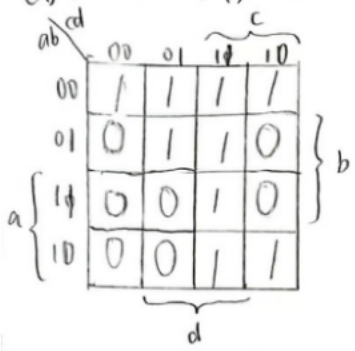
$=(m_3+m_5+m_8+m_{11}+m_{12}+m_{15})'$

~~$=(F(x,y,z))'$~~ $=(F'(A,B,C,D))'$

thus $F(A,B,C,D)=\sum(0,1,2,4,6,7,9,10,13,14)$

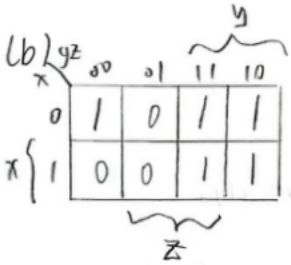
3.

(a) ~~F(a,b,c,d) = (a'b'+b'c+a'd+cd)(a+b'+c)(d+d')~~



$F(a,b,c,d) = a'b' + a'd + cd + b'c$

(b)



$F' = xy'z + xy'$

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

4. (a) $y'z' + yz' + x'z = xy'z' + x'y'z' + xyz' + x'yz' + x'yz + x'y'z$

~~(a) $y'z' + yz' + x'z = z'(y+y') + x'z$~~

$x' + xz' = x'(y+y')(z+z') + x(y+y')z' = x'yz + x'y'z + x'yz + x'y'z' + xyz' + xy'z'$

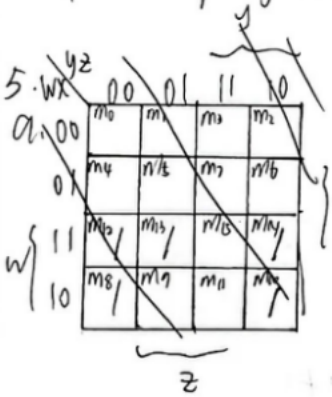
~~$\Rightarrow z' + x'z = zx' + xz'$, false~~

~~(cannot exchange x and z).~~ thus True.

(b) ~~$(x'y + xz + y'z')' = (xy')(x'z')(yz)$, which is definitely not the complement of $xy' + x'z' + yz$, thus false.~~

$xy'z'$

$x'y + xz + y'z' = x'yz + x'yz' + xyz + xy'z' + \cancel{xyz} + xy'z'$



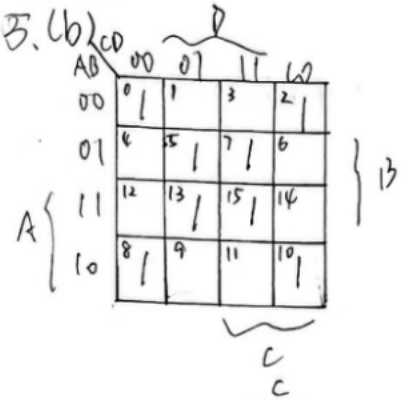
~~f = f(x,~~

~~$F(w,x,y,z) = wz' + wxy'$~~

$xy' + x'z' + yz = xy'z + xy'z' + x'yz' + x'y'z' + xyz + x'yz$

thus True.

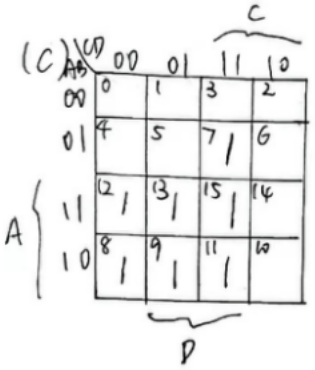
3. (b)



(a) 见下.

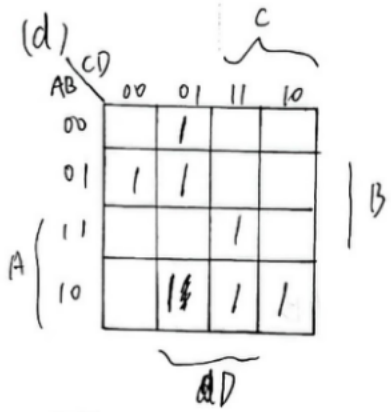
$F(A,B,C,D) = BD + B'D'$

(c)



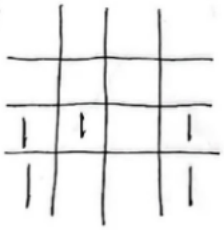
$F(A,B,C,D) = AC' + AD + BCD$

(d)



$F(A,B,C,D) = A'BC' + A'C'D + AB'D + ACD + AB'C$

5. (a)

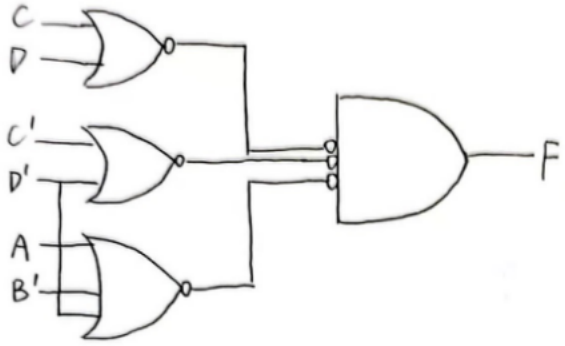


~~$F(A,B,C,D) = AD' + ABC'$~~

$F(W,X,Y,Z) = WZ' + WXY'$

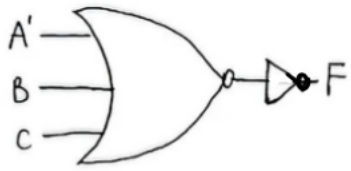
6. (a) $F'(A,B,C,D) = CD + C'D' + A'BD$

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



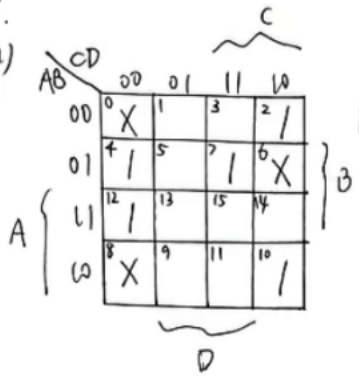
(b) $F'(A,B,C,D) = AB'C'$

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



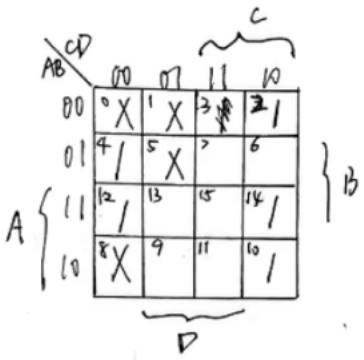
7.

(a)



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

(b)



$F(A,B,C,D) = \cancel{AD' + CD' + A'BD}$

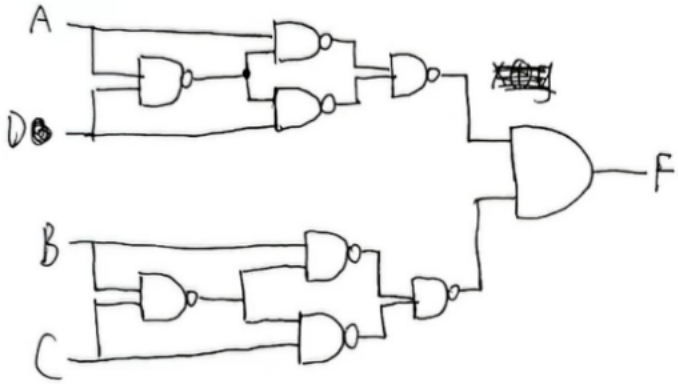
$ACD' + B'CD' + C'D'$

8. $F = A'BC'D + AB'CD' + ABC'D' + A'B'CD$

$= AD'(B'C + BC') + A'D(BC' + B'C)$

$= (AD' + A'D)(B'C + BC')$

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



PART 2: DIGITAL DESIGN LAB (TASK1)

DESIGN

(This part is always REQUIRED)

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

- *Verilog design (provide the Verilog code)*

```
3  module lab7_7_segment_tube(
4      input[0:0] order,
5      input [3:0] in,
6      output reg [7:0] seg_out,
7      output  [7:0] seg_en    );
8
9      reg [3:0] show;
10     always @(*)
11     begin
12       //递减
13       if (order[0])
14         if(in[0] == 1)        show=4'h0;
15         else if(in[1] == 1)   show=4'h1;
16         else if(in[2] == 1)   show=4'h2;
17         else if(in[3] == 1)   show=4'h3;
18         else                  show=4'hf;
19       //递增
20       if (order[1])
21         if(in[3] == 1)    show=4'h3;
22         else if(in[2] == 1)   show=4'h2;
23         else if(in[1] == 1)   show=4'h1;
24         else if(in[0] == 1)   show=4'h0;
25         else                  show=4'hf;
26     end
27
28
29     assign seg_en=8'b01111111;
30     always @(*)
31     begin
32       case(show)
33       4'h0: seg_out=8'b1100_0000;
34       4'h1: seg_out=8'b1111_1001;
35       4'h2: seg_out=8'b1010_0100;
36       4'h3: seg_out=8'b1011_0000;
37       4'h4: seg_out=8'b1001_1001;
38       4'h5: seg_out=8'b1001_0010;
39       4'h6: seg_out=8'b1000_0010;
40       4'h7: seg_out=8'b1111_1000;
41       4'h8: seg_out=8'b1000_0000;
42       4'h9: seg_out=8'b1001_0000;
43       4'ha: seg_out=8'b1000_1000;
44       4'hb: seg_out=8'b1000_0011;
45       4'hc: seg_out=8'b1100_0110;
46       4'hd: seg_out=8'b1010_0001;
47       4'he: seg_out=8'b1000_0110;
48       4'hf:  seg_out=8'b1111_1111;
49     default: seg_out = 8'b0000_0000;
50     endcase
51     end
52 endmodule
```

CONSTRAINT FILE AND THE TESTING

(This part is optional depending on the requirement of the lab task)

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

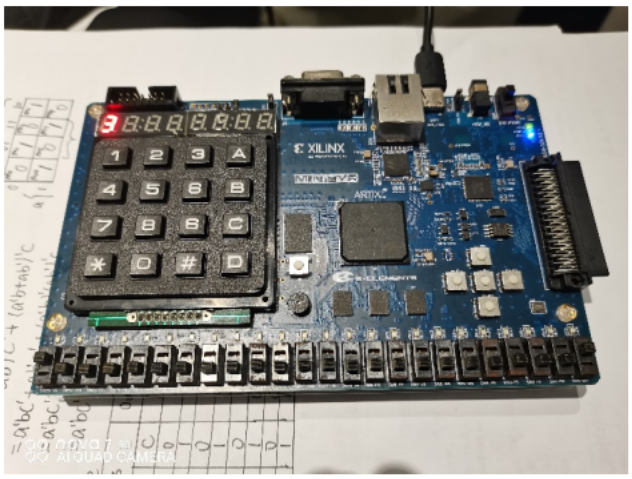
- *Constraint file (provide the screen shots on the feature of a pin and the binding info between pins and the input /output ports)*

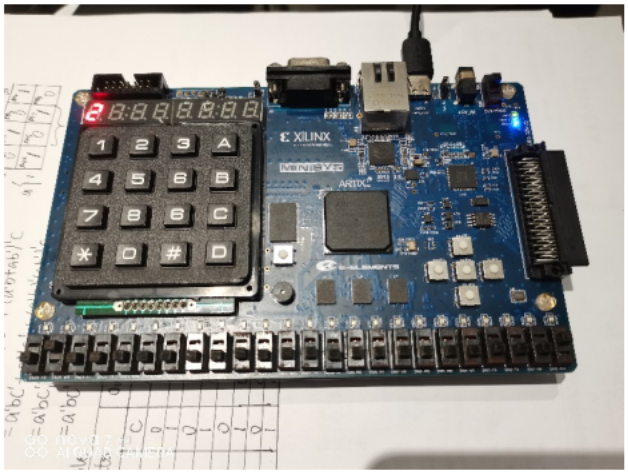
```
set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {in[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {in[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {in[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {in[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {order[0]}]
set_property PACKAGE_PIN Y9 [get_ports {in[3]}]
set_property PACKAGE_PIN W9 [get_ports {in[2]}]
set_property PACKAGE_PIN Y7 [get_ports {in[1]}]
set_property PACKAGE_PIN Y8 [get_ports {in[0]}]
set_property PACKAGE_PIN W4 [get_ports {order[0]}]
set_property PACKAGE_PIN A18 [get_ports {seg_en[7]}]
set_property PACKAGE_PIN A20 [get_ports {seg_en[6]}]
set_property PACKAGE_PIN B20 [get_ports {seg_en[5]}]
set_property PACKAGE_PIN E18 [get_ports {seg_en[4]}]
set_property PACKAGE_PIN F18 [get_ports {seg_en[3]}]
set_property PACKAGE_PIN D19 [get_ports {seg_en[2]}]
set_property PACKAGE_PIN E19 [get_ports {seg_en[1]}]
set_property PACKAGE_PIN C19 [get_ports {seg_en[0]}]
set_property PACKAGE_PIN E13 [get_ports {seg_out[7]}]
set_property PACKAGE_PIN C15 [get_ports {seg_out[6]}]
set_property PACKAGE_PIN C14 [get_ports {seg_out[5]}]
set_property PACKAGE_PIN E17 [get_ports {seg_out[4]}]
set_property PACKAGE_PIN F16 [get_ports {seg_out[3]}]
set_property PACKAGE_PIN F14 [get_ports {seg_out[2]}]
set_property PACKAGE_PIN F13 [get_ports {seg_out[1]}]
set_property PACKAGE_PIN F15 [get_ports {seg_out[0]}]

```

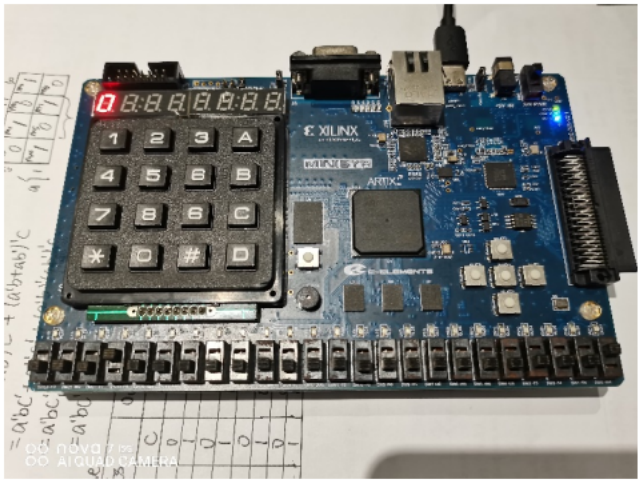
- *The testing result (provide the screen shots (at least 3 testing scene) to show state of inputs and outputs along with the related descriptions.*

Scenarios:

i. Only one ward's call bell is turned on

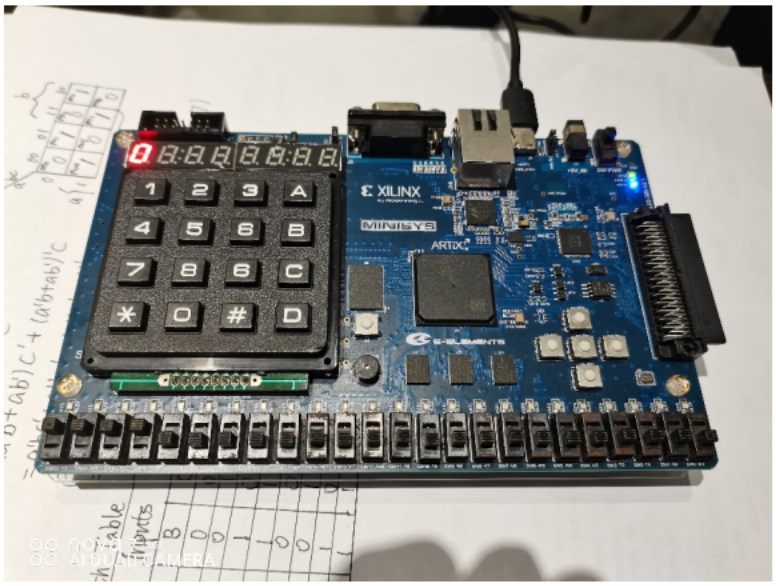




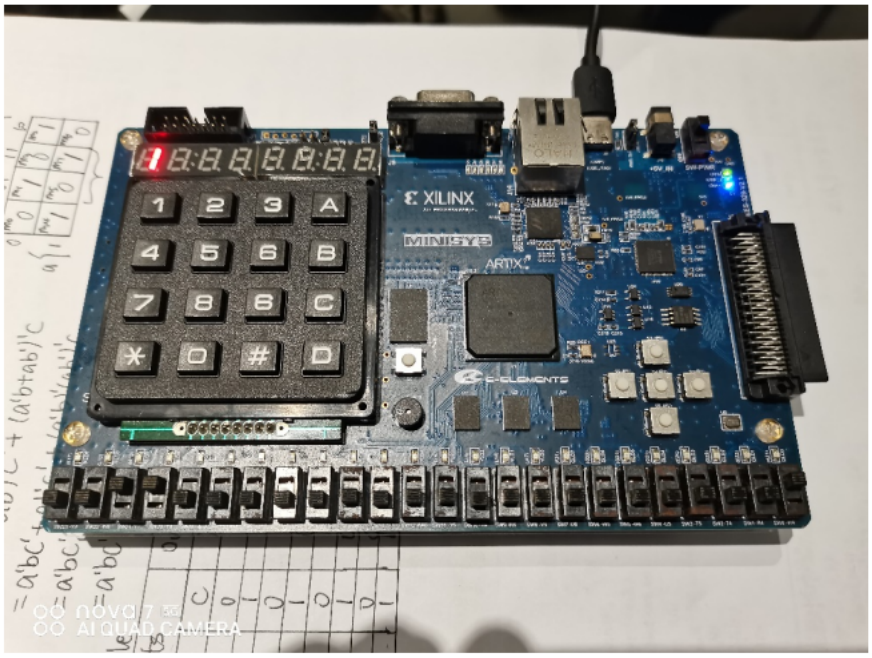


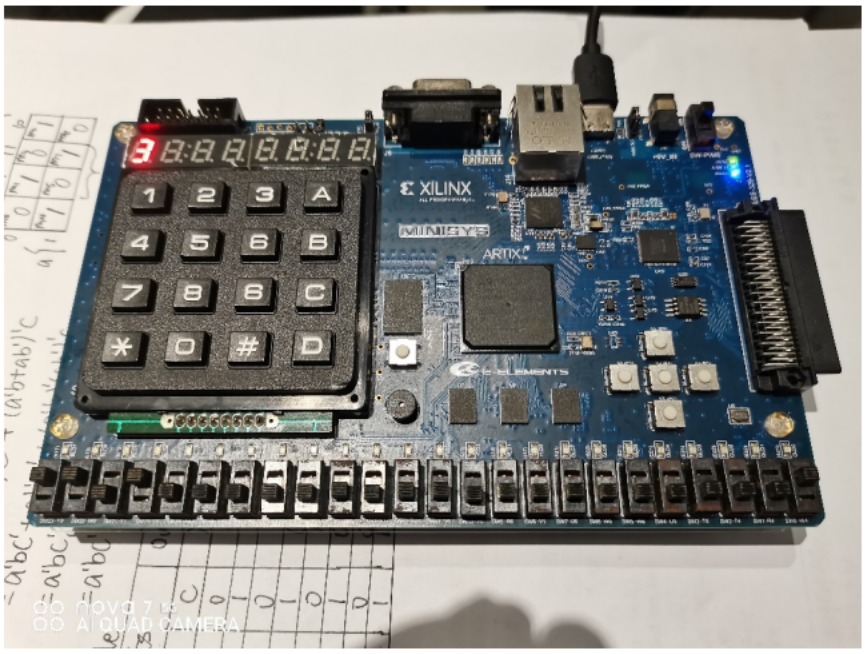
ii. Two or more wards whose call bells are turned on while the priority order is set as priority increases as the number decreases.

For example, the call bell of ward #1, #2, #3 are turned on, number 1(the highest priority among 1, 2 and 3) is expected to be showed on the 7-seg tub.

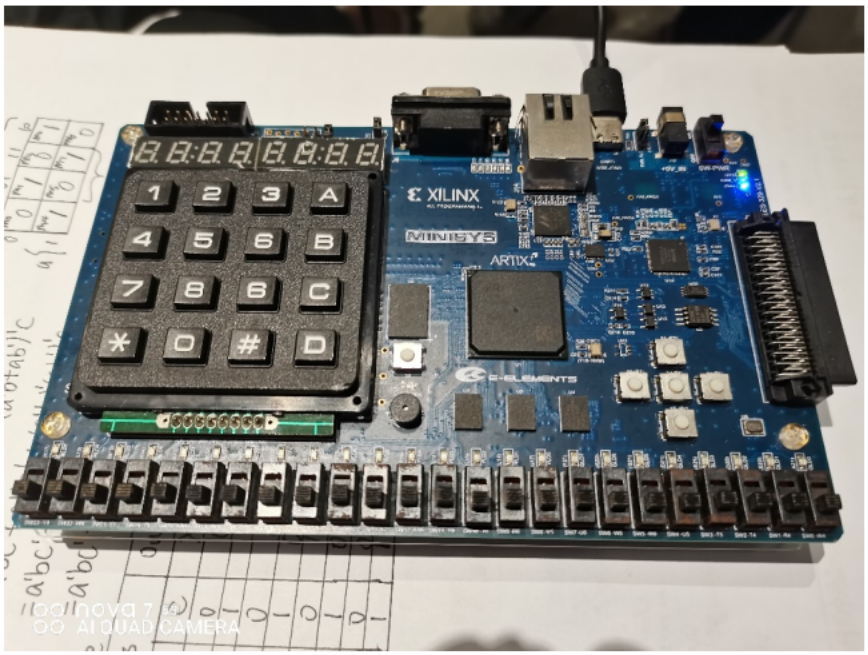


iii. Two or more wards whose call bells are turned on while the priority order is set as priority increases as the number increases. For example, the call bell of ward #1, #2, #3 are turned on, number 3(the highest priority among 1, 2 and 3) is expected to be showed on the 7-seg tub.


1 2 3 A
4 5 6 B
7 8 9 C
* 0 # D
XILINX
MINISYS
ARTIX


1 2 3 A
4 5 6 B
7 8 9 C
* 0 # D
XILINX
MINISYS
ARTIX

iv. No ward's call bell is turned on



THE DESCRIPTION OF OPERATION

(This part is always REQUIRED unless you can get full marks for this lab task)

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

- *Problems and solutions*

  I encountered a lot of problems during this process, and I'm only listing one of them here. For instance, I'm not quite familiar with the constraint files and have no idea how to edit it. But after looking back to the slides of lab and recall what teacher had shown us in the lab, I finally managed to do it.

PART 2: DIGITAL DESIGN LAB (TASK2)

DESIGN

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

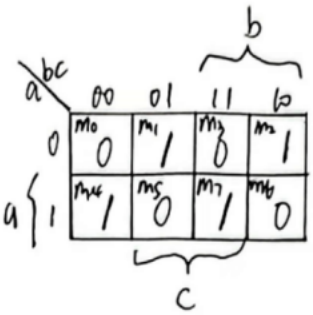
- *Verilog design while using data flow (provide the Verilog code)*

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////////////////////////////////////...
21
22
23 module XOR(a, b, c, x);
24     input a, b, c;
25     output x;
26
27     assign x = ((~a)&(~b)&c) | ((~a)&b&(~c)) | (a&(~b)&(~c)) | (a&b&c);  //m1+m2+m4+m7
28 endmodule
29
30
```

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////////////////////////////////////...
21
22
23 module XOR_Max(a, b, c, x2);
24     input a, b, c;
25     output x2;
26
27     assign x2 = (~a | ~b | c) & (~a | b | ~c) & (a | ~b | ~c) & (a | b | c);  //M0+M3+M5+M6
28 endmodule
29
```

- *Truth-table and K-map*

$$F(a,b,c) = a \oplus b \oplus c$$
$$= (a \oplus b) \oplus c$$
$$= (a'b + ab')\oplus c$$
$$= (a'b + ab')c' + (a'b + ab')'c$$
$$= a'bc' + ab'c' + (a'b)'(ab')'c$$
$$= a'bc' + ab'c' + (a+b')(a'+b)c$$
$$= a'bc' + ab'c' + a'b'c + abc = \sum(1,2,4,7)$$



Truth Table

Inputs			Output
A	B	C	X
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$$F(a,b,c) = \prod(0,3,5,6)$$
$$= M_0 + M_3 + M_5 + M_6$$

SIMULATION

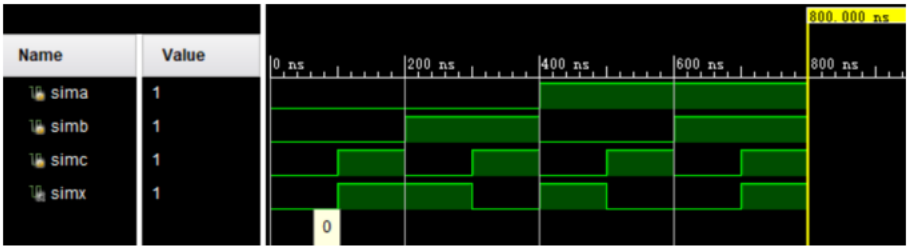
Describe how you build the test bench and do the simulation.

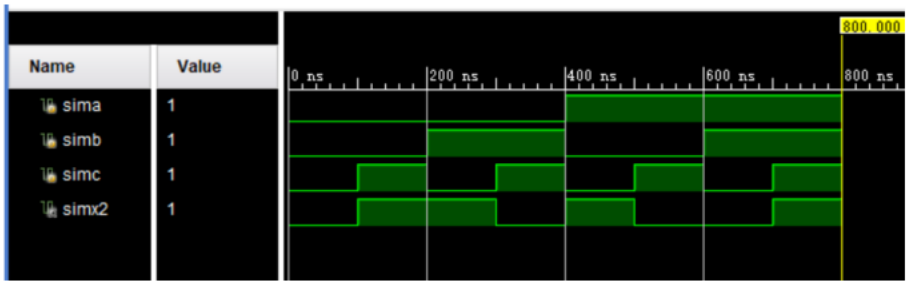
- *Using Verilog (provide the Verilog code)*

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////


module XOR_sim( );
    reg sima, simb, simc;
    wire simx;
    XOR uXOR(.a(sima), .b(simb), .c(simc), .x(simx));

    initial
    begin
        {sima, simb, simc} = 3'b000;
        while ({sima, simb, simc} < 3'b111)
        begin
            #100 {sima, simb, simc} = {sima, simb, simc} + 1;
        end
        #100
        $finish(1);
    end
endmodule
```

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////

module XOR_Max_sim( );
        reg sima, simb, simc;
        wire simx2;
        XOR_Max  uXOR_Max(.a(sima), .b(simb), .c(simc), .x2(simx2));

    initial
    begin
        {sima, simb, simc} = 3'b000;
        while ({sima, simb, simc} < 3'b111)
        begin
            #100 {sima, simb, simc} = {sima, simb, simc} + 1;
        end
        #100
        $finish(1);
    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 exactly the same as the truth-table. For example, when a = 0, b = 1, c = 0, we've got simx = simx2 = 1, which is the same as the truth table above.

THE DESCRIPTION OF OPERATION

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

- *Problems and solutions*

  This task requires knowledge on XOR gate of multiple elements. First, I wasn't that familiar with the XOR gate, especially when it is applied to multiple elements. But after reviewing the lecture slides, I finally learnt to do so.