DIGITAL DESIGN

ASSIGNMENTREPORT

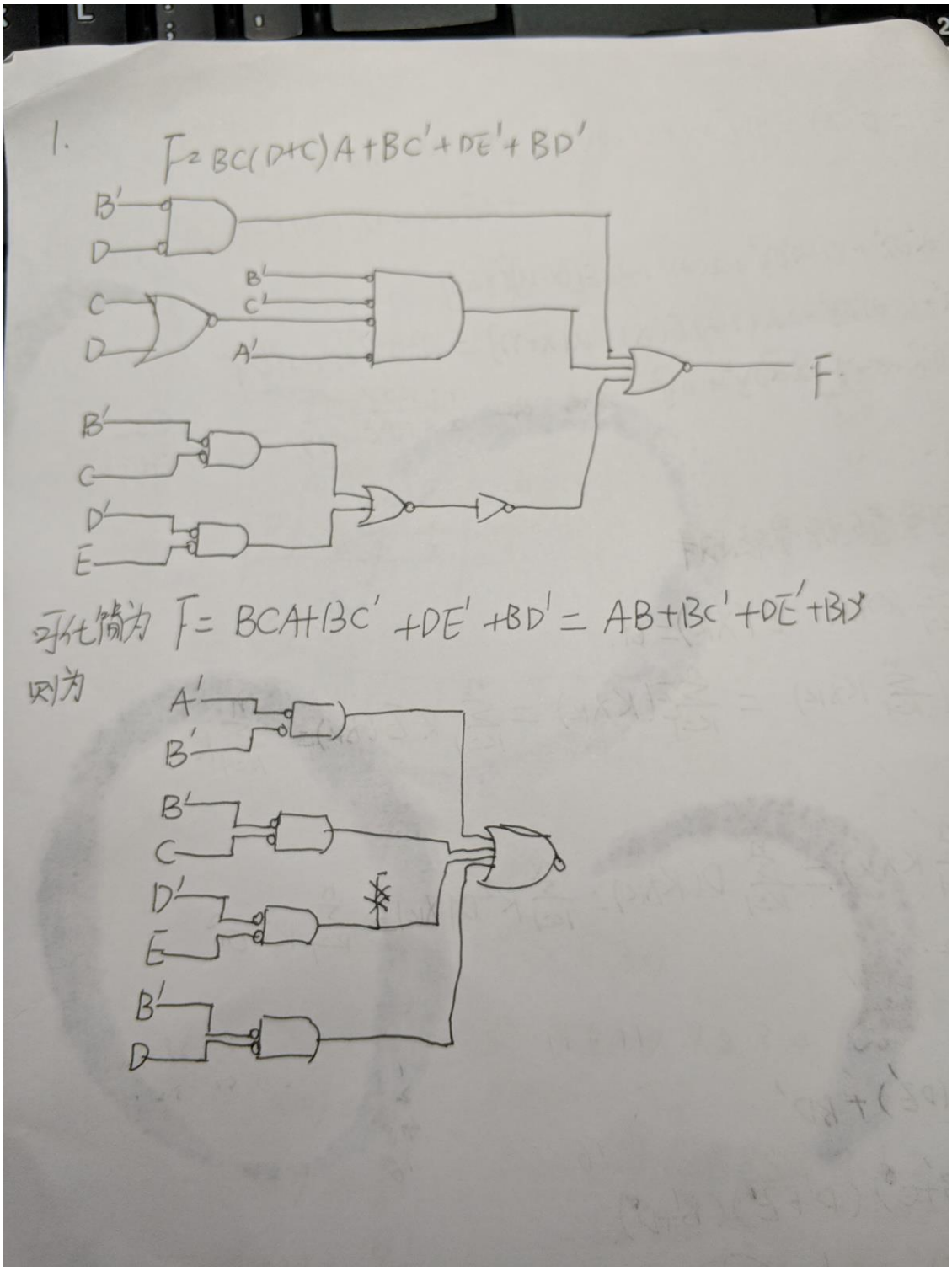
ASSIGNMENT ID : ${Lab_Number}

Student Name:李田所

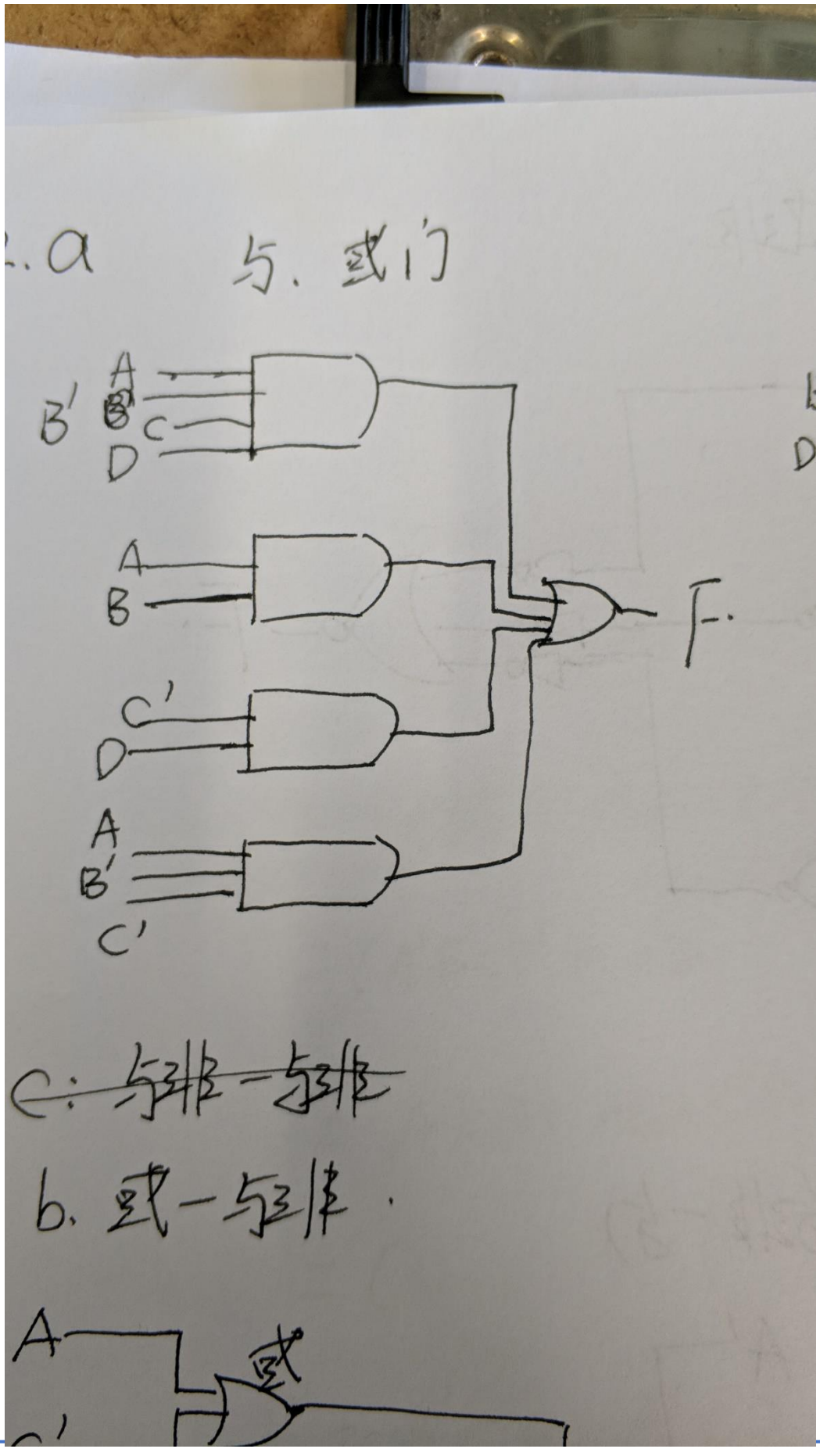
Student ID: 11451419

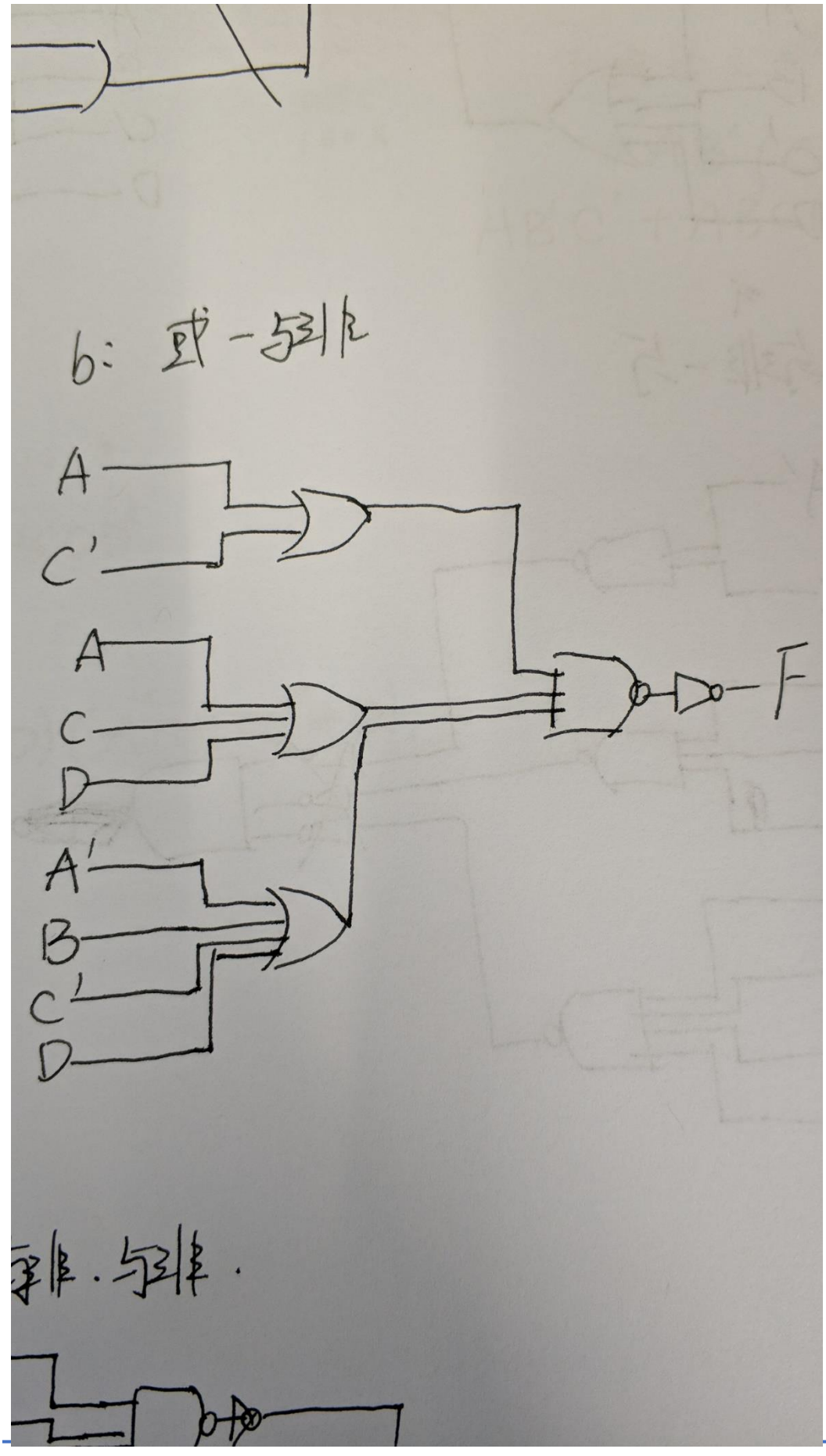
PART 1: DIGITAL DESIGN THEORY

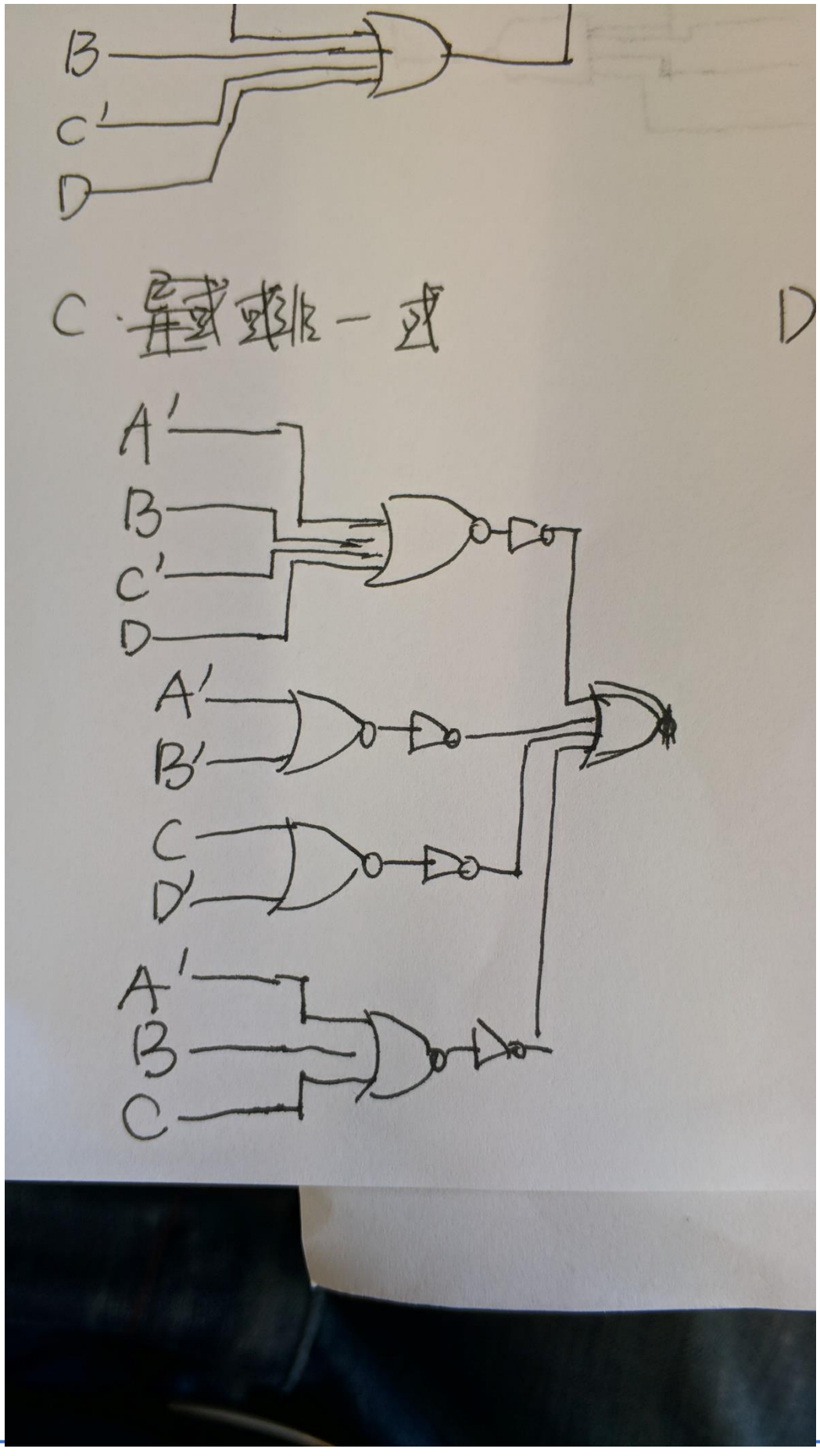
Provide your answers here:

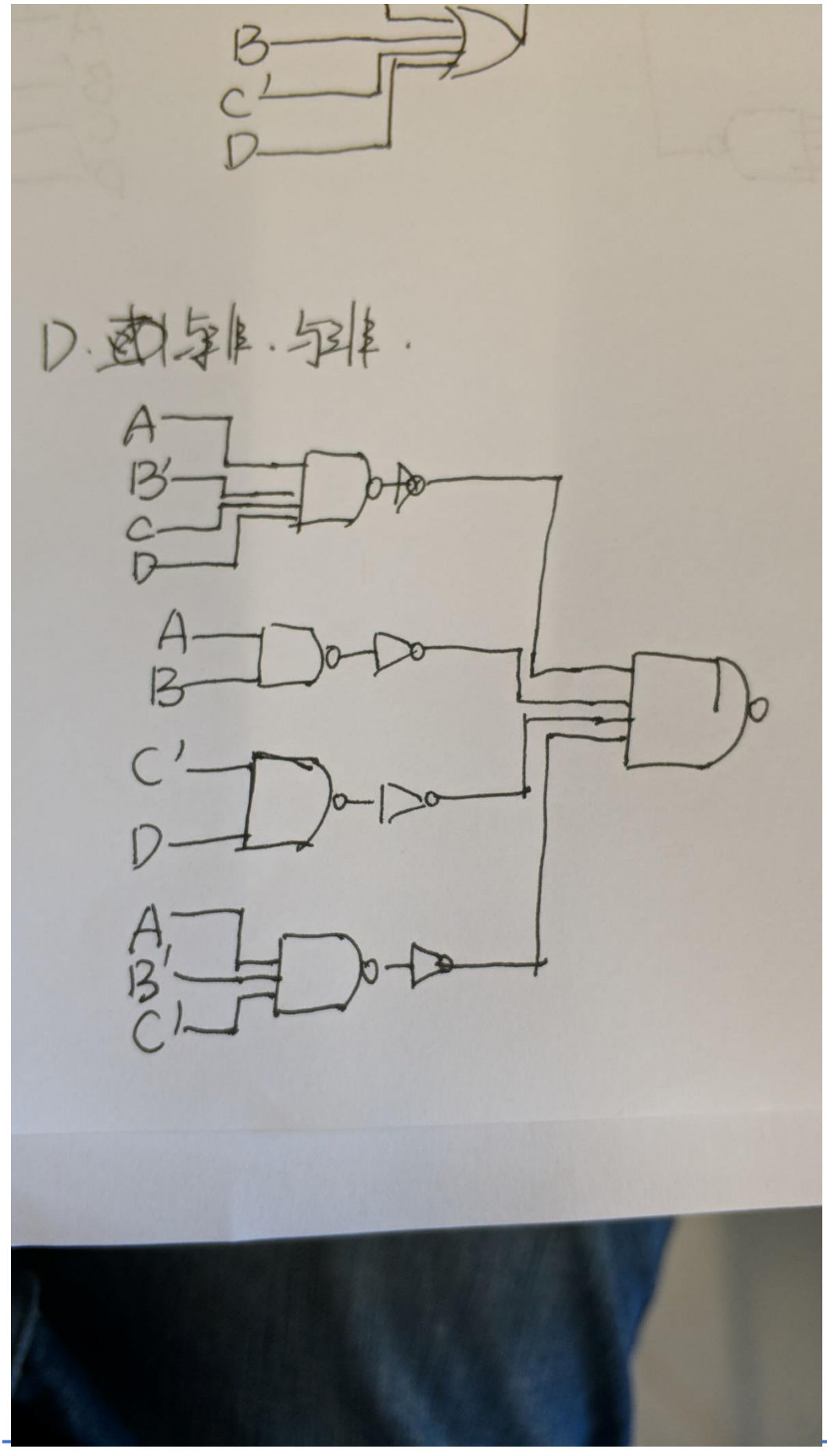
1. 

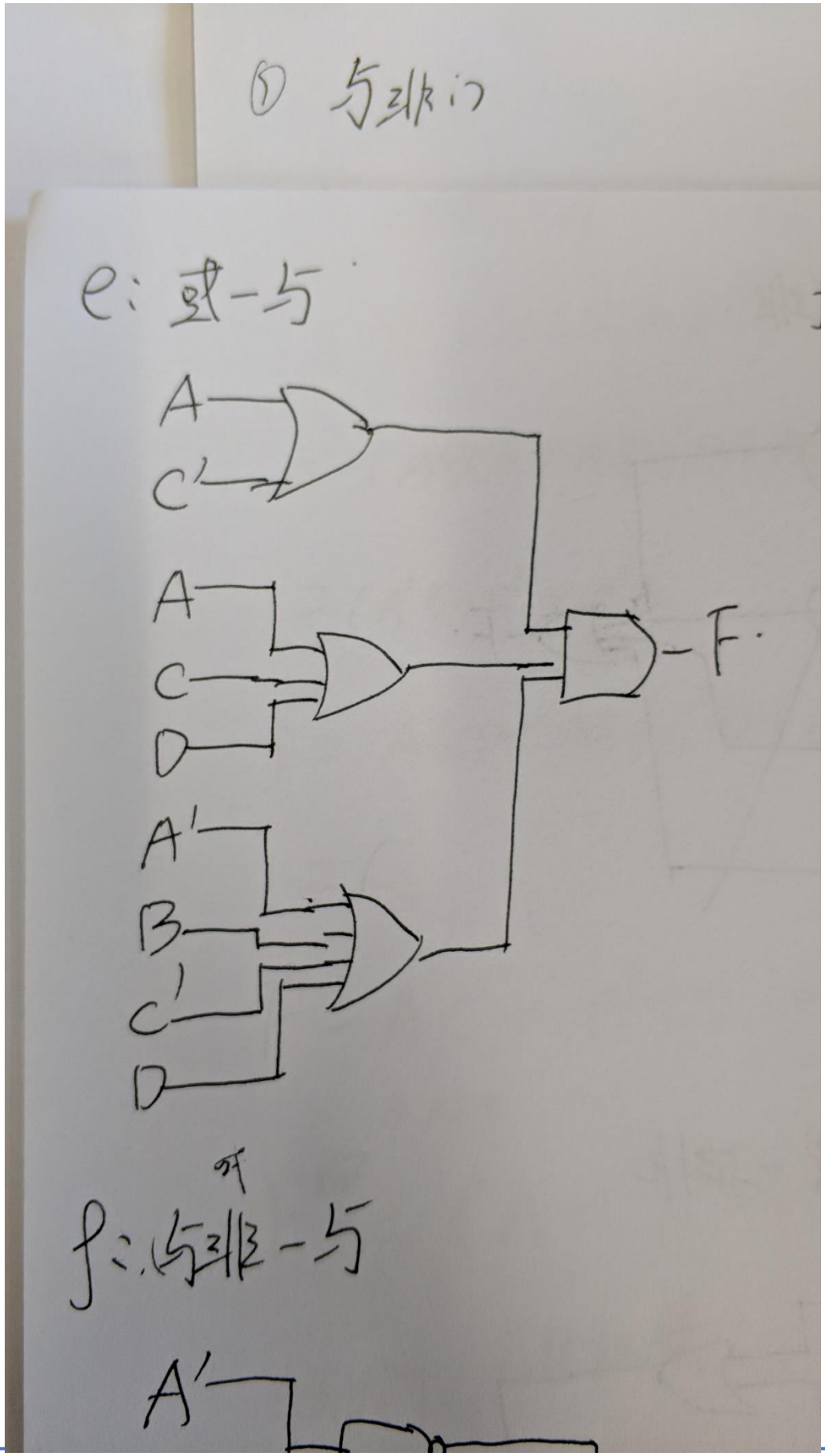
2.

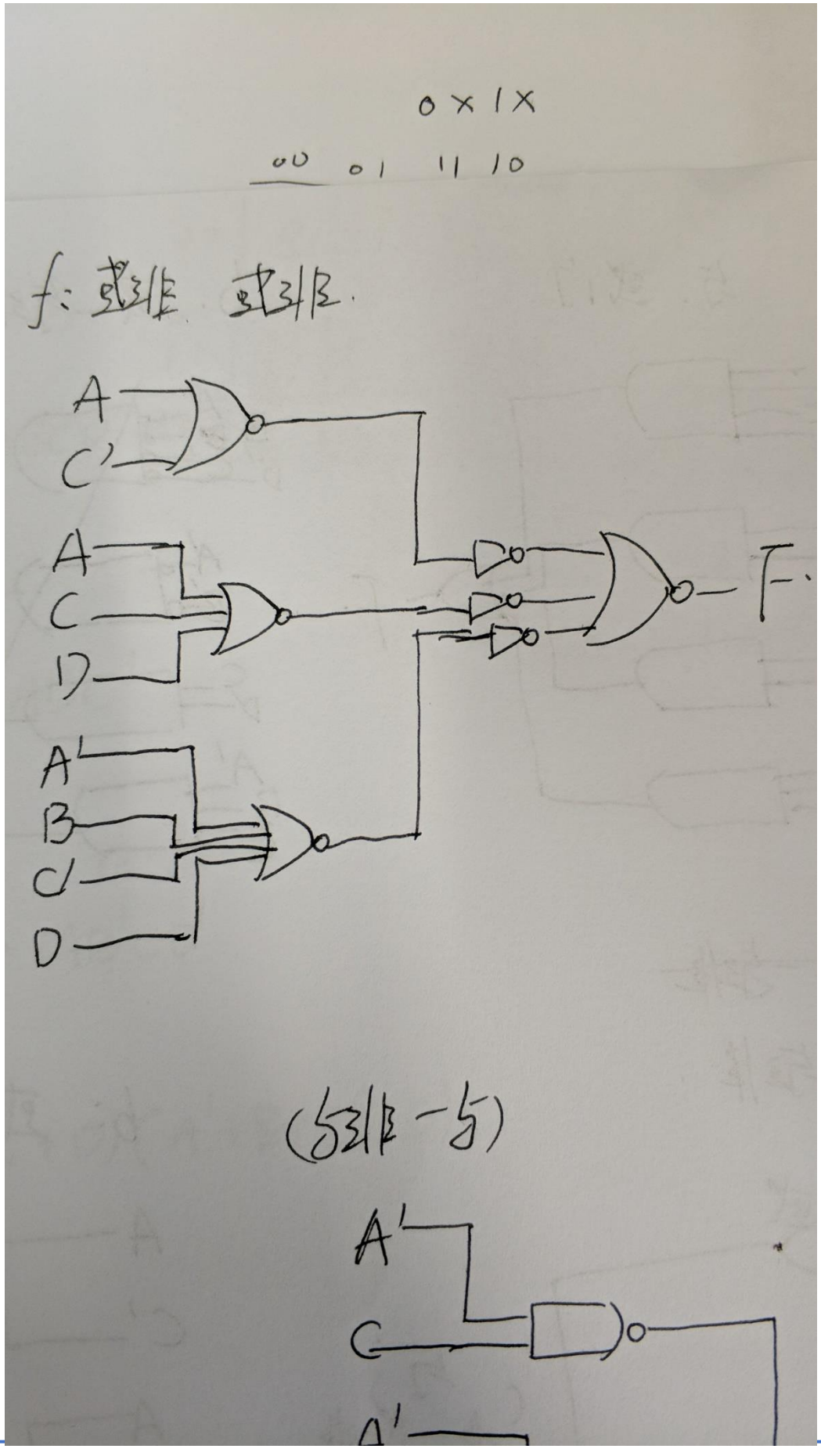

2. a
与. 或门
A
B'
B
C
D
A
B
C'
D
A
B'
C'
F.
c: 与非-与非
b. 或-与非.
A
或

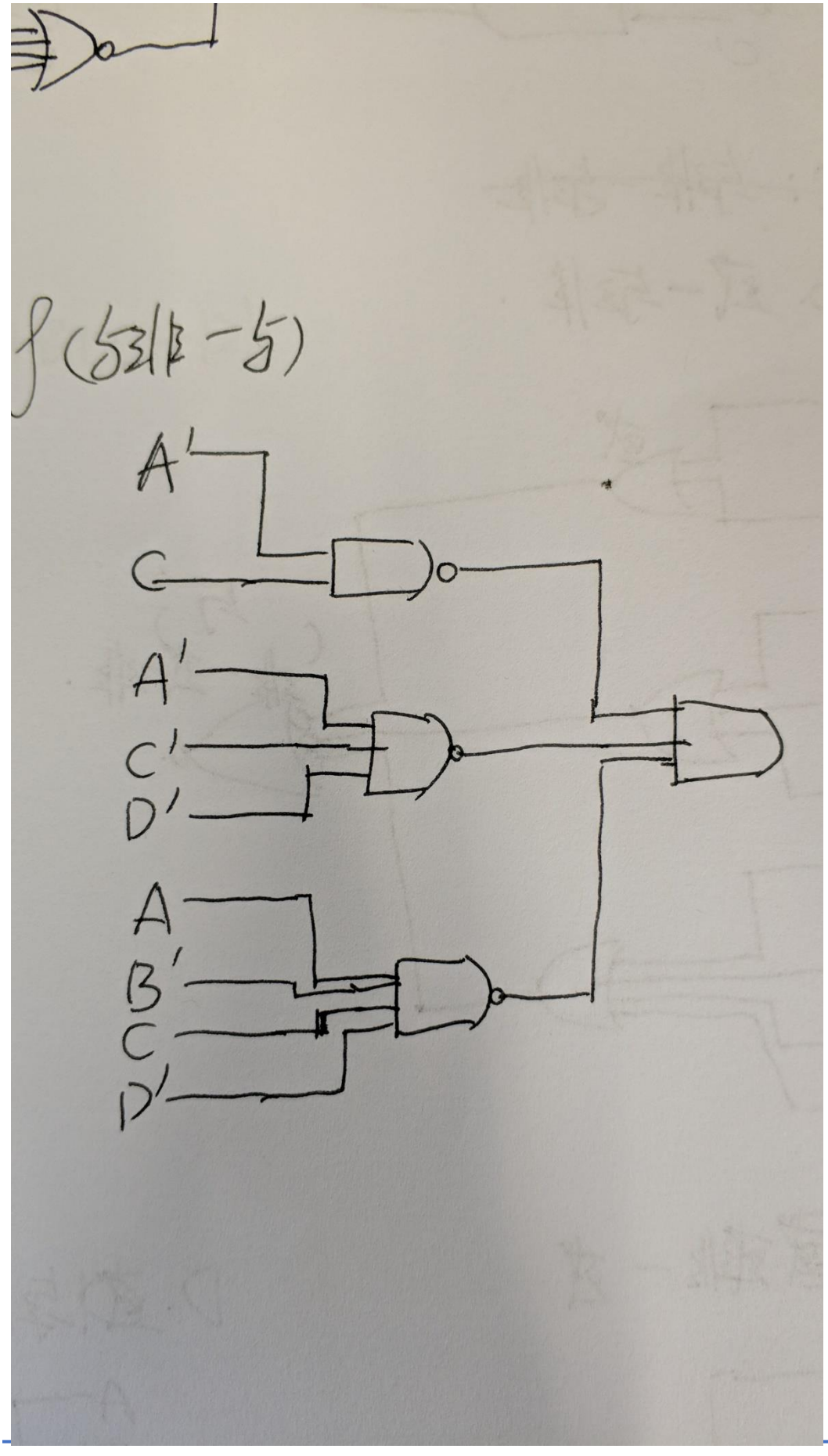

b: 或-与非
A
C'
A
C
D
A'
B
C'
D
F
与非.与非.


B
C'
D
C 异或非-或
D
A'
B
C'
D
A'
B'
C
D
A'
B
C

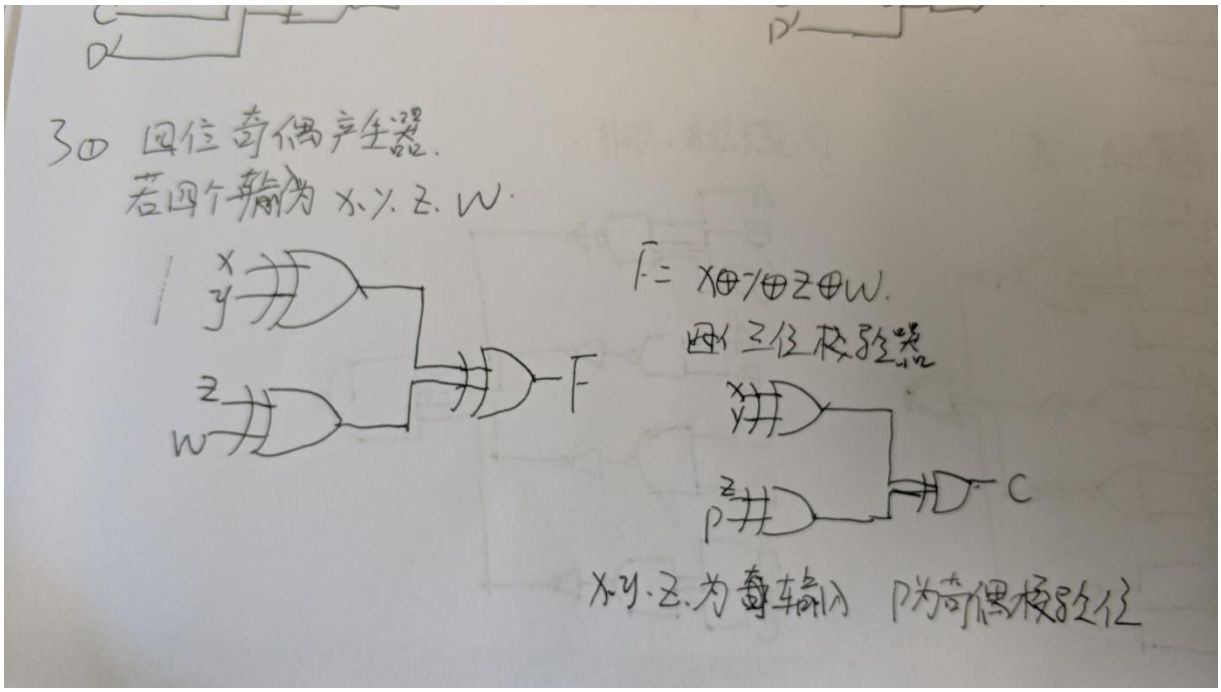

B
C'
D
D. 或非. 与非.
A
B'
C
D
A
B
C'
D
A
B'
C


e: 或-与
A
C'
A
C
D
F
A'
B
C'
D
f: 与非-与
A'


0×1X
00 01 11 10
f: 或非 或非
A
C'
A
C
D
F
A'
B
C
D
(与非-与)
A'
C
A'


A'
C
A'
C'
D'
A
B'
C
D'

3.



4. A = xy+xz+yz’ B = x’y’ + yz + y’z’ C = x’y’z + xz’

5. In the seven output, 1 means output and 0 is no output.

10_base _number	A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1

9	1	0	0	1	1	1	1	1	0	1	1

So a =B’C’D’ + AB’C’ + A’BC’D + A’C

	00	01	11	10
00	A’B’C’D’ 0 √	A’B’C’D1	A’B’CD3 √	A’B’CD’2 √
01	A’BC’D’4	A’BC’D5 √	A’BCD7 √	A’BCD’6 √
11	ABC’D’12	ABC’D13	ABCD15	ABCD’14
10	AB’C’D’8 √	AB’C’D9 √	AB’CD14	AB’CD’10

: b = B’C’ + A’C’D’ + A’B’ + A’CD

	00	01	11	10
00	A’B’C’D’ 0 √	A’B’C’D1 √	A’B’CD3 √	A’B’CD’2 √
01	A’BC’D’4 √	A’BC’D5	A’BCD7 √	A’BCD’6
11	ABC’D’12	ABC’D13	ABCD15	ABCD’14
10	AB’C’D’8 √	AB’C’D9 √	AB’CD11	AB’CD’10

: c = A’B + A’C’ + B’C’ + A’CD

	00	01	11	10
00	A’B’C’D’ 0 √	A’B’C’D1 √	A’B’CD3 √	A’B’CD’2
01	A’BC’D’4 √	A’BC’D5 √	A’BCD7 √	A’BCD’6 √
11	ABC’D’12	ABC’D13	ABCD15	ABCD’14
10	AB’C’D’8 √	AB’C’D9 √	AB’CD11	AB’CD’10

: d = B’C’D’ + AB’C’ + A’BC’D + A’B’C + A’CD’

	00	01	11	10
00	A’B’C’D’ 0 √	A’B’C’D1	A’B’CD3 √	A’B’CD’2 √
01	A’BC’D’4	A’BC’D5 √	A’BCD7	A’BCD’6 √
11	ABC’D’12	ABC’D13	ABCD15	ABCD’14

10	AB’C’D’8 √	AB’C’D9 √	AB’CD11	AB’CD’10

:e = B’C’D’ + A’CD’

	00	01	11	10
00	A’B’C’D’ 0 √	A’B’C’D1	A’B’CD3	A’B’CD’2 √
01	A’BC’D’4	A’BC’D5	A’BCD7	A’BCD’6 √
11	ABC’D’12	ABC’D13	ABCD15	ABCD’14
10	AB’C’D’8 √	AB’C’D9	AB’CD11	AB’CD’10

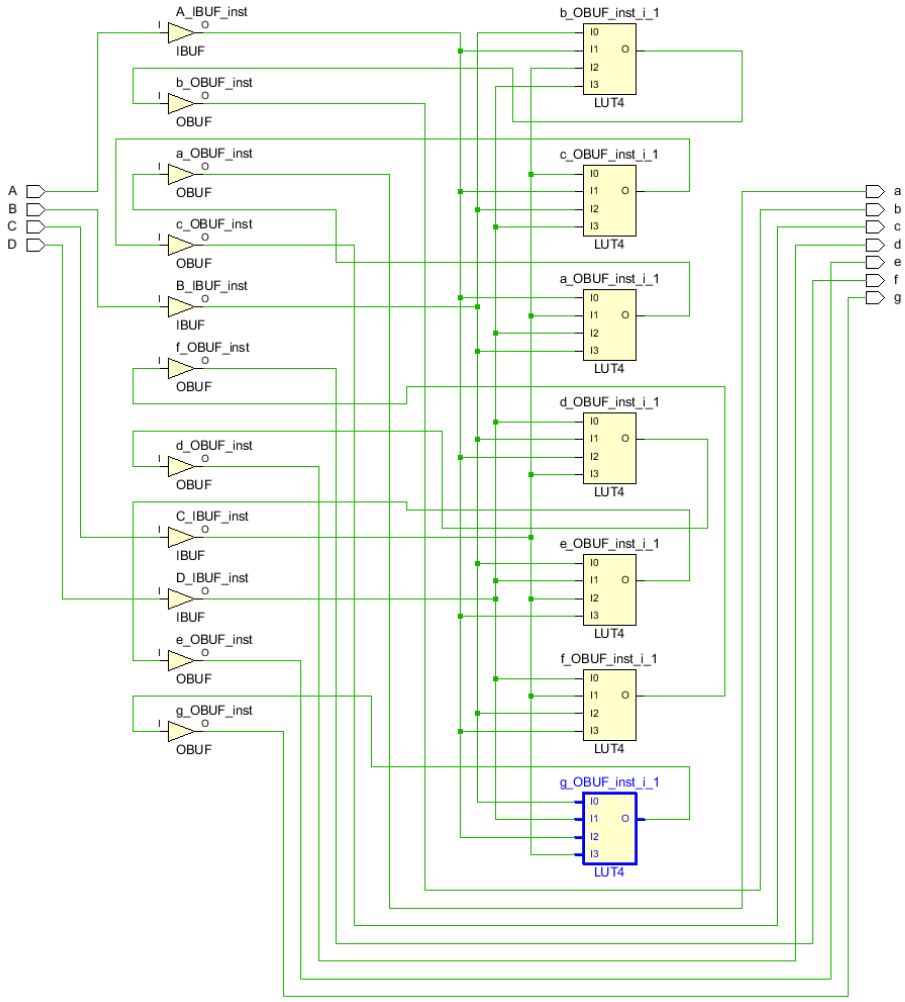
: f = A’C’D’ + B’C’D’ + A’BC’ + AB’C’ + A’BD’

	00	01	11	10
00	A’B’C’D’ 0 √	A’B’C’D1	A’B’CD3	A’B’CD’2
01	A’BC’D’4 √	A’BC’D5 √	A’BCD7	A’BCD’6 √
11	ABC’D’12	ABC’D13	ABCD15	ABCD’14
10	AB’C’D’8 √	AB’C’D9 √	AB’CD11	AB’CD’10

: g = A’BC’ + A’B’C + A’CD’ + AB’C’

	00	01	11	10
00	A’B’C’D’ 0	A’B’C’D1	A’B’CD3 √	A’B’CD’2 √
01	A’BC’D’4 √	A’BC’D5 √	A’BCD7	A’BCD’6 √
11	ABC’D’12	ABC’D13	ABCD15	ABCD’14
10	AB’C’D’8 √	AB’C’D9 √	AB’CD11	AB’CD’10

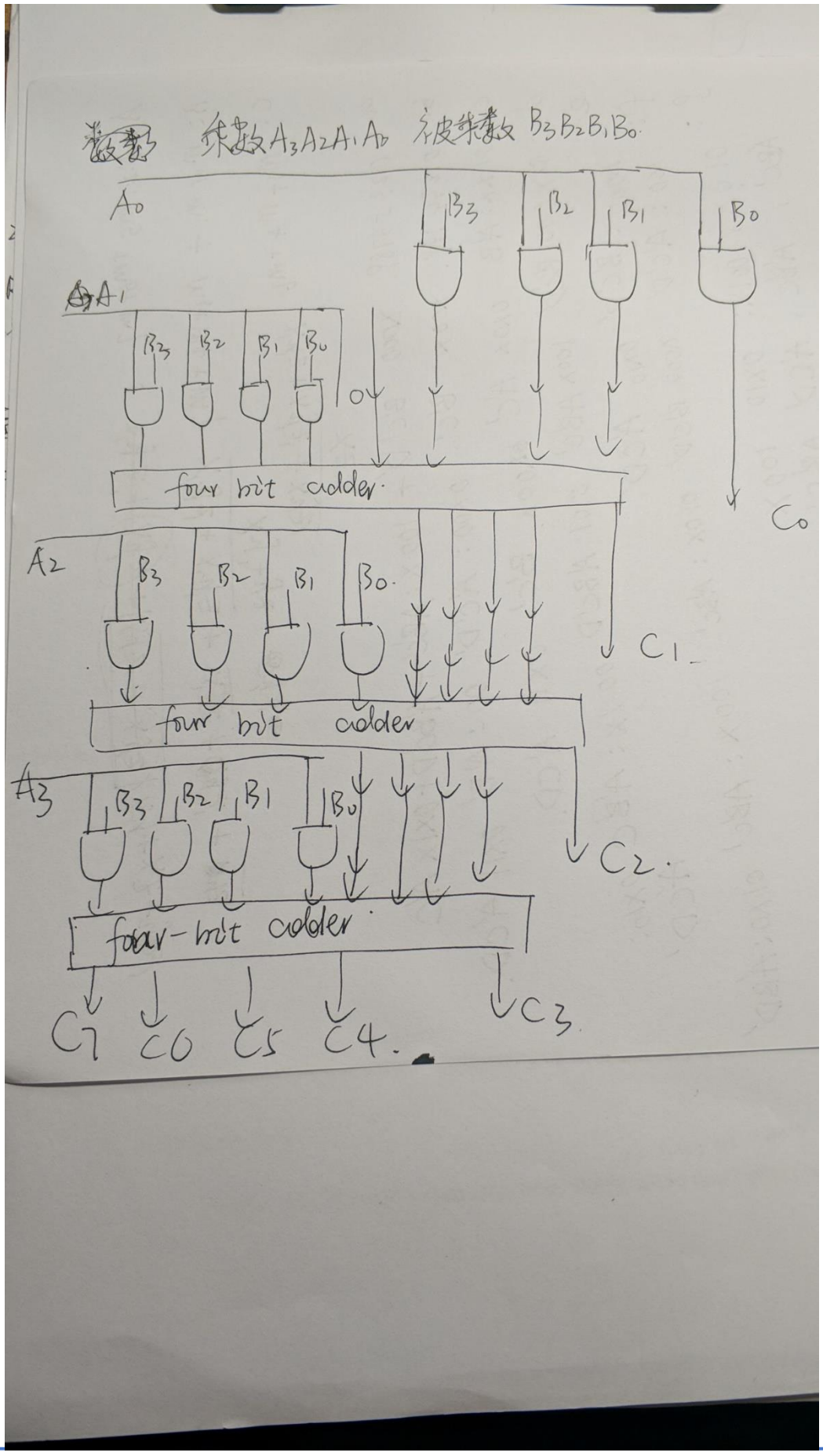
So all of the output can express as the sum of products, what we need is 10 four input and many output **and** gate ,four input variable and four reverse input variables, & 10 **or** gate which has different input. The sum of the gates is 28.



The code is:

```
`timescale 1ns / 1ps

module temp1(

input A,B,C,D,

output a,b,c,d,e,f,g

    );

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

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

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

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

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

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

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

endmodule
```


乘数 $A_3A_2A_1A_0$ 被乘数 $B_3B_2B_1B_0$
A_0
B_3 B_2 B_1 B_0
A_1
B_3 B_2 B_1 B_0
0
four bit colder
C_0
A_2
B_3 B_2 B_1 B_0
C_1
four bit colder
A_3
B_3 B_2 B_1 B_0
C_2
four-bit colder
C_7 C_6 C_5 C_4
C_3

6.

a. for a binary multiplier that multiplies two unsigned four-bit numbers. 16 and gate and three four-bit-adder are needed to make the circuit.

the graph of the multiplier that multiplies two unsigned four-bit numbers is:

four_bit_multiplier_code:

```
`timescale 1ns / 1ps
   // 4 is the higest
module four_bit_multply(
input firstMulti1,
input firstMulti2,
input firstMulti3,
input firstMulti4,
input secondMulti1,
input secondMulti2,
input secondMulti3,
input secondMulti4,
output output0,
output output1,
output output2,
output output3,
output output4,
output output5,
output output6,
output output7);
      wire temp1;
      wire temp2;
```

```
wire temp3;
wire temp4;
wire temp5;
wire temp6;
wire temp7;
wire temp8;
  assign output0 = firstMulti1 & secondMulti1;
  four_bit_adder fout_bit_adder1(
  (firstMulti2 & secondMulti1),(firstMulti2 & secondMulti2),
  (firstMulti2 & secondMulti3),(firstMulti2 & secondMulti4),
  (firstMulti1 & secondMulti2),(firstMulti1 & secondMulti3),
  (firstMulti1 & secondMulti4),0,
  temp1,temp2,temp3,temp4,output1);

  four_bit_adder fout_bit_adder2(
  (firstMulti3 & secondMulti1),(firstMulti3 & secondMulti2),
  (firstMulti3 & secondMulti3),(firstMulti3 & secondMulti4),
  temp4,temp3,temp2,temp1,
  temp5,temp6,temp7,temp8,output2);

  four_bit_adder fout_bit_adder3(
    (firstMulti4 & secondMulti1),(firstMulti4 & secondMulti2),
    (firstMulti4 & secondMulti3),(firstMulti4 & secondMulti4),
    temp8,temp7,temp6,temp5,
    output7,output6,output5,output4,output3);
```

endmodule

four_bit_adder_code:

```
`timescale 1ns / 1ps
module four_bit_adder(
input firstAdd1,
input firstAdd2,
input firstAdd3,
input firstAdd4,
input secondAdd1,
input secondAdd2,
input secondAdd3,
input secondAdd4,
output isEn,
output output4,
output output3,
output output2,
output output1
);
wire temp1;
wire temp2;
wire temp3;
two_bit_adder add1(firstAdd1,secondAdd1,0,temp1,output1);
two_bit_adder add2(firstAdd2,secondAdd2,temp1,temp2,output2);
two_bit_adder add3(firstAdd3,secondAdd3,temp2,temp3,output3);
two_bit_adder add4(firstAdd4,secondAdd4,temp3,isEn,output4);
```

```
endmodule
```

two_bit_adder_code:

```
`timescale 1ns / 1ps
module two_bit_adder(
    input x,
    input y,
    input en,
    output isEn,
    output result
);
assign result = ( ~x & ~y & en) | (~x & y & ~en) |    ( x & ~y & ~en)    | (x & y & en);
assign isEn    = ( x & y) |( y & en) | (en & x);

endmodule
```

the test_sim_code:

```
`timescale 1ns / 1ps
    module four_bit_multi_sim(

        );
        reg sim1M1,sim1M2,sim1M3,sim1M4,sim2M1,sim2M2,sim2M3,sim2M4;
        wire
simoutput1,simoutput2,simoutput3,simoutput4,simoutput5,simoutput6,simoutput7,simoutput8
;
        four_bit_multply temp1(
        .firstMulti1(sim1M1),
        .firstMulti2(sim1M2),
```

```
.firstMulti3(sim1M3),
.firstMulti4(sim1M4),
.secondMulti1(sim2M1),
.secondMulti2(sim2M2),
.secondMulti3(sim2M3),
.secondMulti4(sim2M4),
.output0(simoutput1),
.output1(simoutput2),
.output2(simoutput3),
.output3(simoutput4),
.output4(simoutput5),
.output5(simoutput6),
.output6(simoutput7),
.output7(simoutput8)
);
initial
    begin
    {sim1M1,sim1M2,sim1M3,sim1M4} = 4'b0000;
    {sim2M1,sim2M2,sim2M3,sim2M4} = 4'b0000;
    repeat (15)
        begin
        #1
        {sim1M1,sim1M2,sim1M3,sim1M4} =
{sim1M1,sim1M2,sim1M3,sim1M4} + 1;
          repeat (16)
```

```
                begin
                #1
                  {sim2M1,sim2M2,sim2M3,sim2M4} =
{sim2M1,sim2M2,sim2M3,sim2M4} +1;
                  $display(sim1M1 + " " +sim1M2+ " "    +sim1M3+ " "
+sim1M4+ " "    +sim2M1 + " " +sim2M2+ " "    +sim2M3+ " "    +sim2M4 );
                end
            end
        end
    endmodule
```

we can also use the more directly way:

the .v_code:

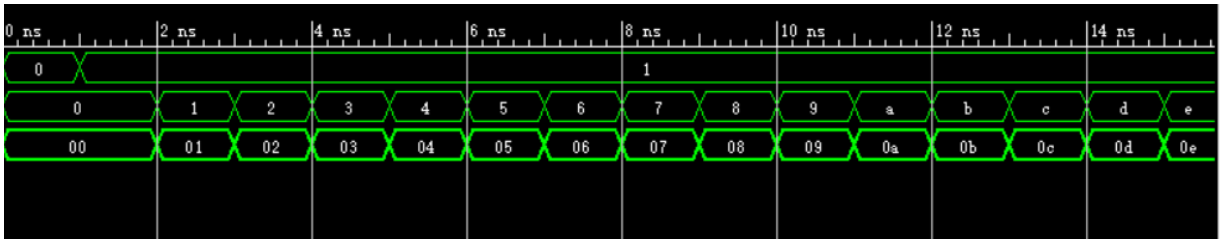
```
`timescale 1ns / 1ps
module more_directly(
    input [3:0]m1,
```

```
        input [3:0]m2,
        output [7:0] output1
        );
        assign output1 = m1 * m2;
endmodule
```

the sim_code:

```
`timescale 1ns / 1ps
module more_directly_sim();
reg [3:0]m1sim;
reg [3:0]m2sim;
wire [7:0]output1sim;
more_directly test1(
.m1(m1sim),
.m2(m2sim),
.output1(output1sim));
        initial
                begin
                m1sim = 4'b0000;
                m2sim = 4'b0000;
                repeat (15)
                                begin
                                #1
                                m1sim=    m1sim + 1;
                                 repeat (16)
                                        begin
```

```
                #1
                m2sim =    m2sim+1;
                   $display(m1sim +" " + m2sim );
                end
            end
    end
endmodule
```



7. EI: enable the input. EO:enable the output GS:gate spread

EI	0	1	2	3	4	5	6	7	2	1	0	EO	GS
1	x	x	x	x	x	x	x	x	1	1	1	1	1
0	x	x	x	x	x	x	x	0	0	0	0	1	0
0	x	x	x	x	x	x	0	1	0	0	1	1	0
0	x	x	x	x	x	0	1	1	0	1	0	1	0
0	x	x	x	x	0	1	1	1	0	1	1	1	0
0	x	x	x	0	1	1	1	1	1	0	0	1	0
0	x	x	0	1	1	1	1	1	1	0	1	1	0
0	x	0	1	1	1	1	1	1	1	1	0	1	0
0	0	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	0	1

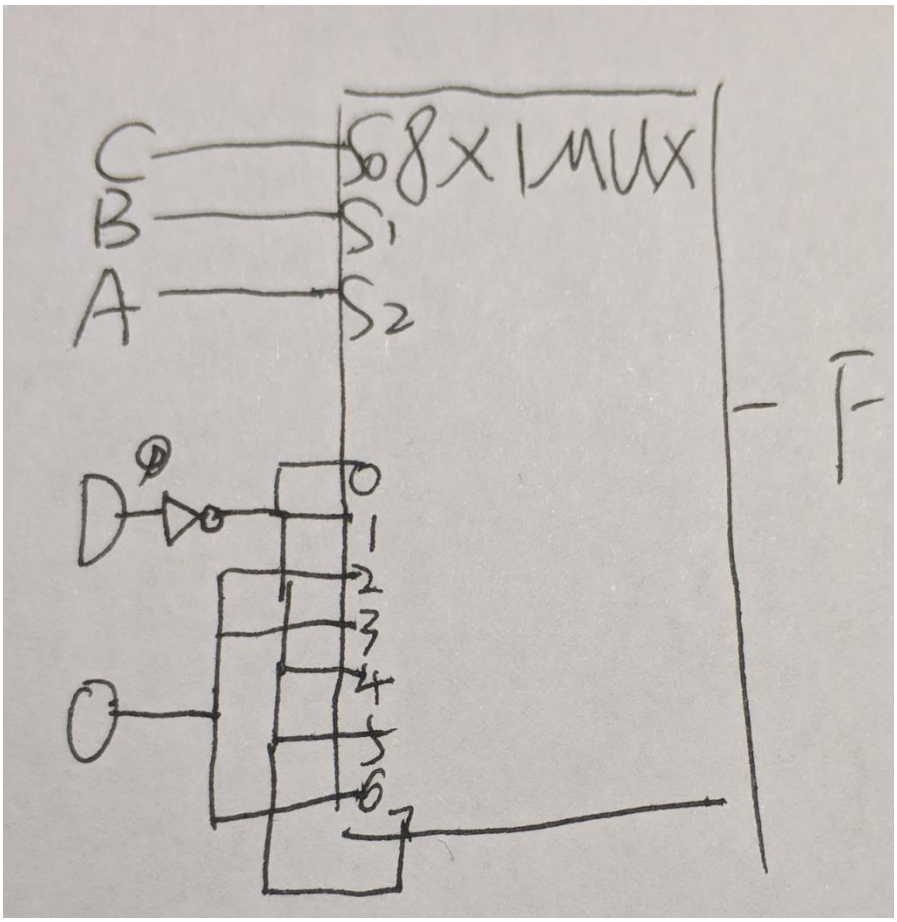
If the input 2&6 is 1 and others are 0(included the EI), then the output is $0_2\ 0_1\ 1_0\ 1_{EO}\ 0_{GS}$

8.

a. F(A, B, C, D) =

∑ (0, 2, 5, 8, 10, 14) = A’B’C’D’ + A’B’CD’ + A’BC’D+AB’C’D’ + AB’CD’ + ABCD’

A	B	C	D	F	
0	0	0	0	1	F= D’
0	0	0	1	0	
0	0	1	0	1	F= D’
0	0	1	1	0	
0	1	0	0	0	F= 0
0	1	0	1	1	
0	1	1	0	0	F= 0
0	1	1	1	0	
1	0	0	0	1	F= D’
1	0	0	1	0	
1	0	1	0	1	F= D’
1	0	1	1	0	
1	1	0	0	0	F= 0
1	1	0	1	0	
1	1	1	0	1	F= D’
1	1	1	1	0	



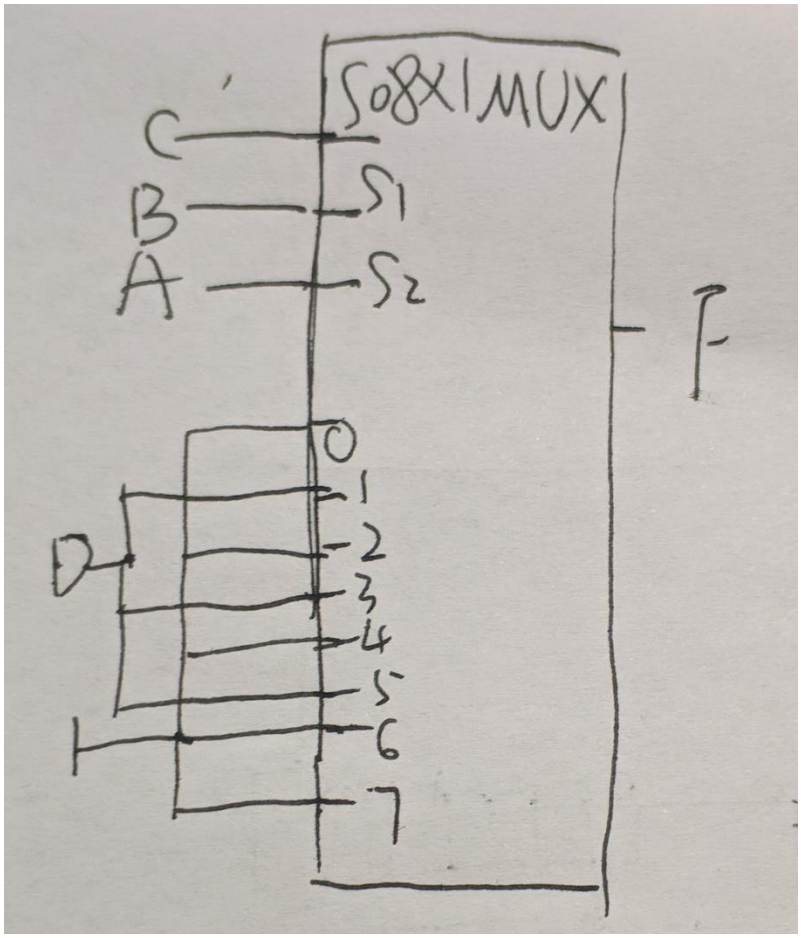
b. F(A, B, C, D) =

$\prod$ (2, 6, 11) = $\sum$(0,1,3,4,5,7,8,9,10,12,13,14,15)

= A'B'C'D' + A'B'C'D + A'B'CD + A'BC'D' + A'BC'D + A'BCD + AB'C'D' + AB'C'D + AB'CD' + ABC'D' + ABC'D + ABCD' + ABCD

A	B	C	D	F	
0	0	0	0	1	F= 1
0	0	0	1	1	
0	0	1	0	0	F= D
0	0	1	1	1	
0	1	0	0	1	F= 1
0	1	0	1	1	

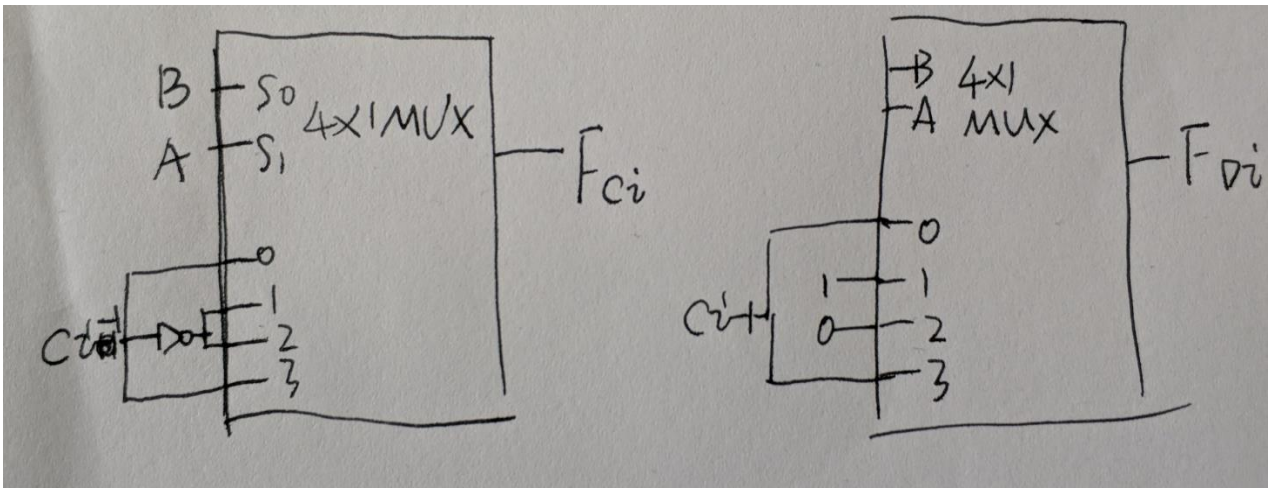
0	1	1	0	0	F= D
0	1	1	1	1	
1	0	0	0	1	F= 1
1	0	0	1	1	
1	0	1	0	1	F= D’
1	0	1	1	0	
1	1	0	0	1	F= 1
1	1	0	1	1	
1	1	1	0	1	F=1
1	1	1	1	1	



9. Every full subtracter have three input and two output: the minuend ,the subtrahend, is the last borrow digit are input and is this digit will borrow digit and the result in this bit are double output.

A	B	C_{i-1}	C_i	F_1 =	D_i	F_2 =
0	0	0	0	$F = C_{i-1}$	0	$F = C_{i-1}$
0	0	1	1		1	
0	1	0	1	$F = C_{i-1}'$	1	F = 1
0	1	1	0		1	
1	0	0	1	$F = C_{i-1}'$	0	F = 0
1	0	1	0		0	
1	1	0	0	$F = C_{i-1}$	0	$F = C_{i-1}$
1	1	1	1		1	

So the graph of the subtracter is

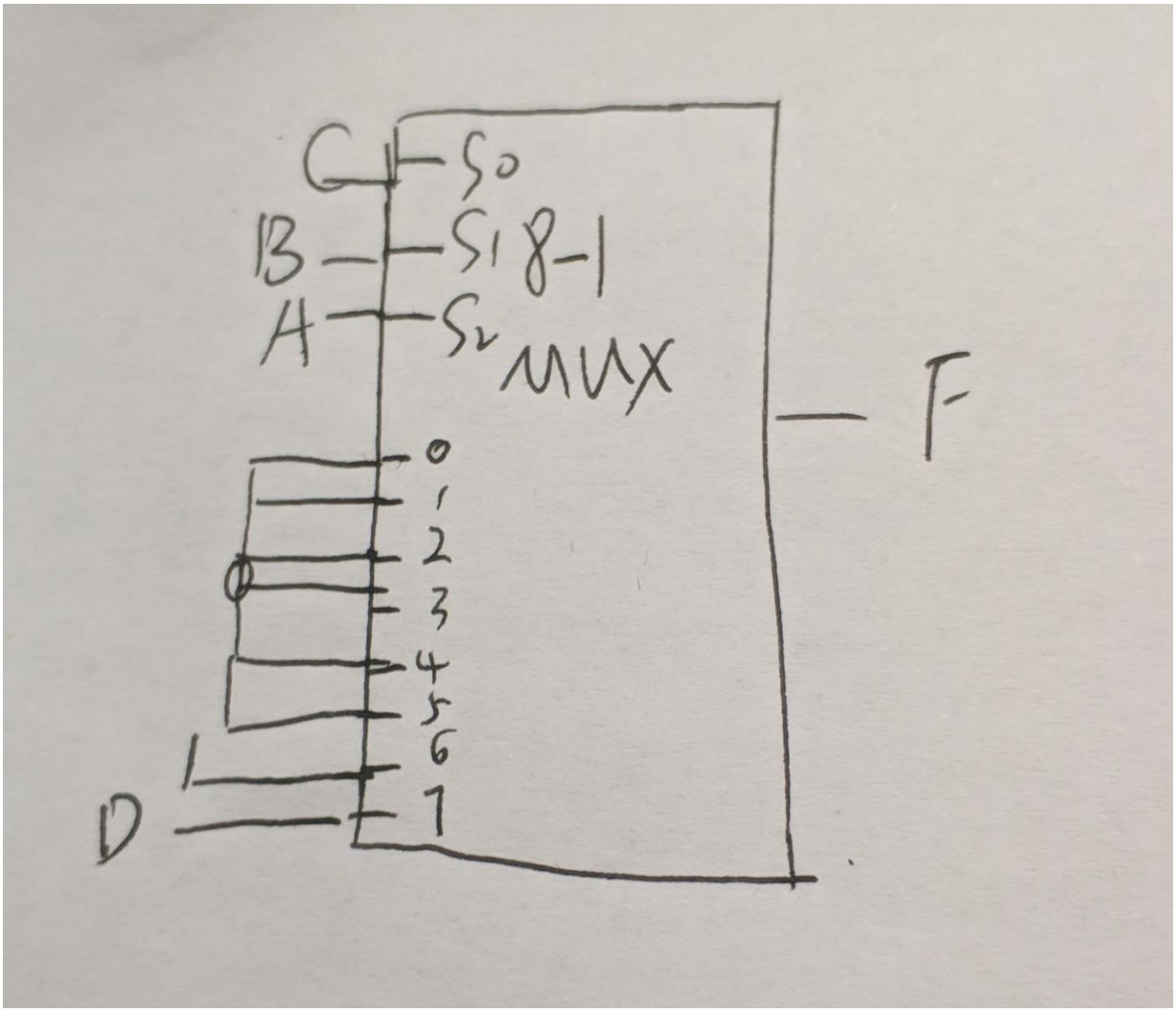


10.

A	B	C	D	F	
0	0	0	0	0	F= D
0	0	0	1	1	
0	0	1	0	1	F= 1

0	0	1	1	1	
0	1	0	0	1	F= 1
0	1	0	1	1	
0	1	1	0	0	F= 0
0	1	1	1	0	
1	0	0	0	1	F= D’
1	0	0	1	0	
1	0	1	0	1	F= D’
1	0	1	1	0	
1	1	0	0	0	F= D
1	1	0	1	1	
1	1	1	0	0	F= 0
1	1	1	1	0	

So F = A’B’C’D + A’B’CD’ + A’B’CD + A’BC’D’ + A’BC’D + AB’C’D’ + AB’CD’ + ABC’D.



PART 2: DIGITAL DESIGN LAB (TASK1)

The words before codes and graph

1. The first way is write the truth table and write the code, using four_bit_input_variable ,two eight_bit_output_variable and a four_bit_output_variable.
2. Using one repeat to make every state be used,add the xdc file with setted seg-gate
3. Then do the synthesis,set the I/Oports.
4. Run implementation
5. Generate the bitstream and use the borad to take photos.

DESIGN

```
`timescale 1ns / 1ps

module Ass_3_1_v(

    input [15:0]input1,

    output reg [7:0]    seg_out,

    output [7:0] seg_en,

    output [15:0] output1

     );

      assign seg_en = ~8'hff;

      assign output1 = input1;

      always @*

      begin

        casex(input1)

        16'bxxxxxxxxxxxxxxx1: seg_out = 8'b01000000;//0

        16'bxxxxxxxxxxxxxx10: seg_out = 8'b01111001;//1

        16'bxxxxxxxxxxxxx100: seg_out = 8'b00100100;//2

        16'bxxxxxxxxxxxx1000: seg_out = 8'b00110000;//3

        16'bxxxxxxxxxxx10000: seg_out = 8'b00011001;//4

        16'bxxxxxxxxxx100000: seg_out = 8'b00010010;//5

        16'bxxxxxxxxx1000000: seg_out = 8'b00000010;//6
```

```
  16'bxxxxxxxx10000000: seg_out = 8'b01111000;//7

  16'bxxxxxxx100000000: seg_out = 8'b00000000;//8

  16'bxxxxxx1000000000: seg_out = 8'b00010000;//9

  16'bxxxxx10000000000: seg_out = 8'b00001000;//A

  16'bxxxx100000000000: seg_out = 8'b00000011;//B

  16'bxxx1000000000000: seg_out = 8'b01000110;//C

  16'bxx10000000000000: seg_out = 8'b00100001;//D

  16'bx100000000000000: seg_out = 8'b00000110;//E

  16'b1000000000000000: seg_out = 8'b00001110;//F

  default:seg_out = 8'b0000000;

endcase
```

Ass_3_1_v.v × Ass_3_1_xdc.xdc × Ass_3_1_sim.v * ×

C:/Users/37682/XilinxProject/Ass_3_1/Ass_3_1.srcs/sources_1/new/Ass_3_1_v.v

```
`timescale 1ns / 1ps
module Ass_3_1_v(
  input [15:0]input1,
  output reg [7:0]  seg_out,
  output [7:0] seg_en,
  output [15:0] output1
   );
   assign seg_en = ~8'hff;
   assign output1 = input1;
   always @*
   begin
    casex(input1)
    16'bxxxxxxxxxxxxxxx1: seg_out = 8'b01000000;//0
    16'bxxxxxxxxxxxxxx10: seg_out = 8'b01111001;//1
    16'bxxxxxxxxxxxxx100: seg_out = 8'b00100100;//2
    16'bxxxxxxxxxxxx1000: seg_out = 8'b00110000;//3
    16'bxxxxxxxxxxx10000: seg_out = 8'b00011001;//4
    16'bxxxxxxxxxx100000: seg_out = 8'b00010010;//5
    16'bxxxxxxxxx1000000: seg_out = 8'b00000010;//6
    16'bxxxxxxxx10000000: seg_out = 8'b01111000;//7
    16'bxxxxxxx100000000: seg_out = 8'b00000000;//8
    16'bxxxxxx1000000000: seg_out = 8'b00010000;//9
    16'bxxxxx10000000000: seg_out = 8'b00001000;//A
    16'bxxxx100000000000: seg_out = 8'b00000011;//B
    16'bxxx1000000000000: seg_out = 8'b01000110;//C
    16'bxx10000000000000: seg_out = 8'b00100001;//D
    16'bx100000000000000: seg_out = 8'b00000110;//E
    16'b1000000000000000: seg_out = 8'b00001110;//F
    default:seg_out = 8'b0000000;
   endcase
   end
```

end

all of the code and waves is correct, after my calculate all of them are right.

SIMULATION

During the simulation, because the amount of the case are too much,every step just stay in unit 1 time. But there still can not act enough cases.

Code:

```
`timescale 1ns / 1ps

module Ass_3_1_sim( );

reg [15:0]input1sim;

wire [7:0]seg_out_sim;

wire [7:0]seg_en_sim;

wire [15:0]output1sim;

Ass_3_1_v test(
```

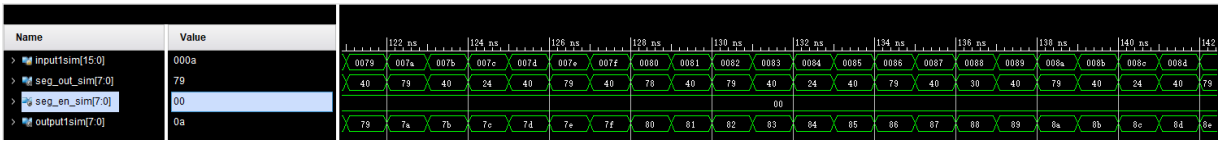
```
    .input1(input1sim),
    .seg_out(seg_out_sim),
    .seg_en(seg_en_sim),
    .output1(output1sim));
        initial
        begin
        input1sim = 16'b0000000000000000;
        repeat(65536)
            begin
            #1
            input1sim = input1sim + 1;

$display($time,"{input1,seg_out,seg_en,output1}:%d %d %d %d",input1sim,seg_out_sim,seg_en_sim,output1sim);
            end
        end
    endmodule
```

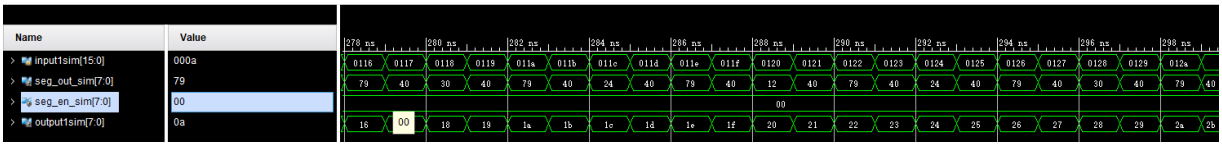
Ass_3_1_v.v | Ass_3_1_xdc.xdc | Ass_3_1_sim.v | Untitled 8

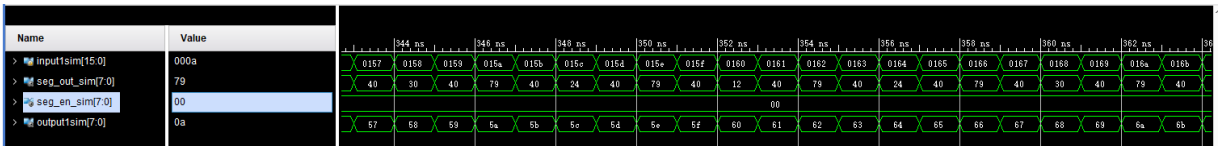
C:/Users/37682/XilinxProject/Ass_3_1/Ass_3_1.srcs/sim_1/new/Ass_3_1_sim.v

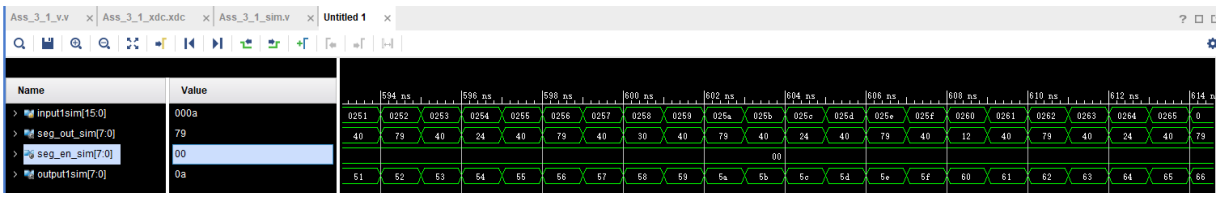
```
1   `timescale 1ns / 1ps
2   module Ass_3_1_sim( );
3   reg [15:0]input1sim;
4   wire [7:0]seg_out_sim;
5   wire [7:0]seg_en_sim;
6   wire [15:0]output1sim;
7   Ass_3_1_v test(
8   .input1(input1sim),
9   .seg_out(seg_out_sim),
10  .seg_en(seg_en_sim),
11  .output1(output1sim));
12      initial
13      begin
14      input1sim = 16'b0000000000000000;
15      repeat(65536)
16          begin
17          #1
18          input1sim = input1sim + 1;
19           $display($time, "{input1, seg_out, seg_en, output1}:%d %d %d %d", input1sim, seg_out_sim, seg_en_sim, output1sim);
20          end
21      end
22  endmodule
```

Because there are so many case that I just can take a few case to express it.



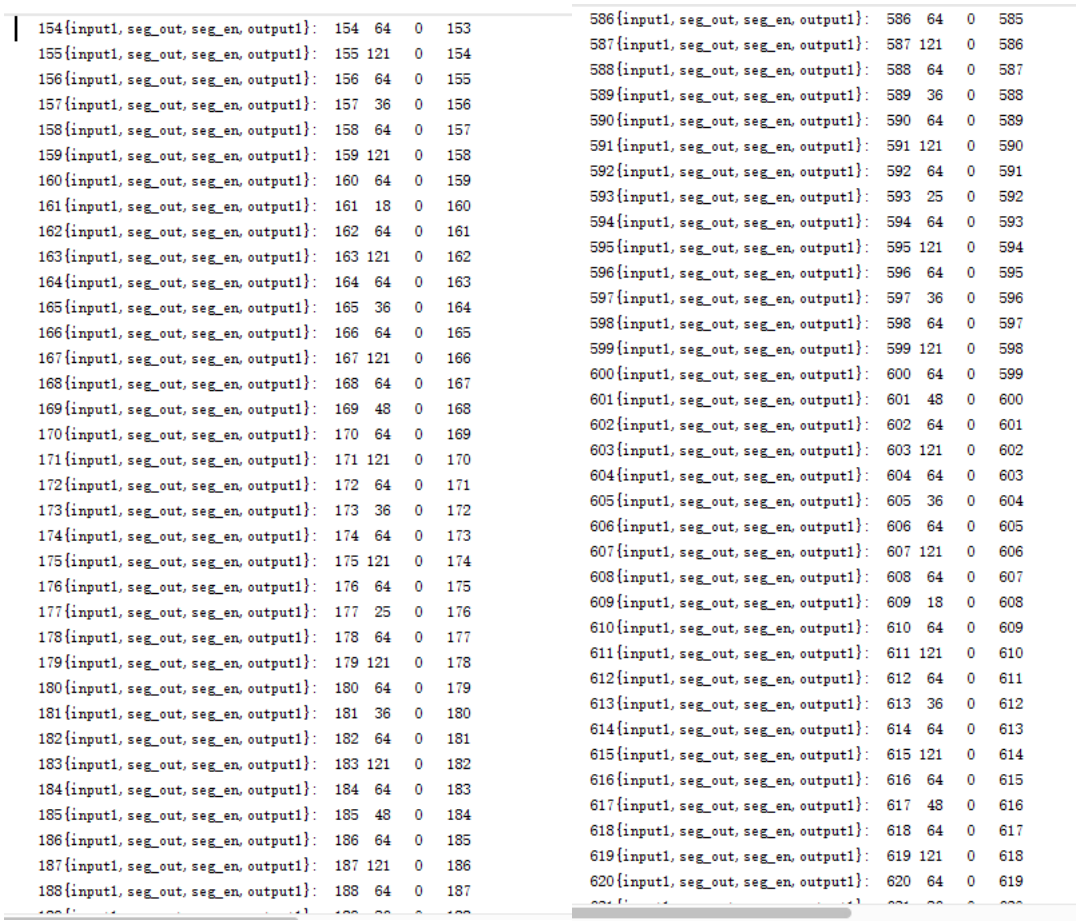






And this is some display , there are too much display there just put some

This is two example of the simualation value in system task



```
154{input1, seg_out, seg_en, output1}:  154  64   0  153
155{input1, seg_out, seg_en, output1}:  155 121   0  154
156{input1, seg_out, seg_en, output1}:  156  64   0  155
157{input1, seg_out, seg_en, output1}:  157  36   0  156
158{input1, seg_out, seg_en, output1}:  158  64   0  157
159{input1, seg_out, seg_en, output1}:  159 121   0  158
160{input1, seg_out, seg_en, output1}:  160  64   0  159
161{input1, seg_out, seg_en, output1}:  161  18   0  160
162{input1, seg_out, seg_en, output1}:  162  64   0  161
163{input1, seg_out, seg_en, output1}:  163 121   0  162
164{input1, seg_out, seg_en, output1}:  164  64   0  163
165{input1, seg_out, seg_en, output1}:  165  36   0  164
166{input1, seg_out, seg_en, output1}:  166  64   0  165
167{input1, seg_out, seg_en, output1}:  167 121   0  166
168{input1, seg_out, seg_en, output1}:  168  64   0  167
169{input1, seg_out, seg_en, output1}:  169  48   0  168
170{input1, seg_out, seg_en, output1}:  170  64   0  169
171{input1, seg_out, seg_en, output1}:  171 121   0  170
172{input1, seg_out, seg_en, output1}:  172  64   0  171
173{input1, seg_out, seg_en, output1}:  173  36   0  172
174{input1, seg_out, seg_en, output1}:  174  64   0  173
175{input1, seg_out, seg_en, output1}:  175 121   0  174
176{input1, seg_out, seg_en, output1}:  176  64   0  175
177{input1, seg_out, seg_en, output1}:  177  25   0  176
178{input1, seg_out, seg_en, output1}:  178  64   0  177
179{input1, seg_out, seg_en, output1}:  179 121   0  178
180{input1, seg_out, seg_en, output1}:  180  64   0  179
181{input1, seg_out, seg_en, output1}:  181  36   0  180
182{input1, seg_out, seg_en, output1}:  182  64   0  181
183{input1, seg_out, seg_en, output1}:  183 121   0  182
184{input1, seg_out, seg_en, output1}:  184  64   0  183
185{input1, seg_out, seg_en, output1}:  185  48   0  184
186{input1, seg_out, seg_en, output1}:  186  64   0  185
187{input1, seg_out, seg_en, output1}:  187 121   0  186
188{input1, seg_out, seg_en, output1}:  188  64   0  187
```

```
586{input1, seg_out, seg_en, output1}:  586  64   0  585
587{input1, seg_out, seg_en, output1}:  587 121   0  586
588{input1, seg_out, seg_en, output1}:  588  64   0  587
589{input1, seg_out, seg_en, output1}:  589  36   0  588
590{input1, seg_out, seg_en, output1}:  590  64   0  589
591{input1, seg_out, seg_en, output1}:  591 121   0  590
592{input1, seg_out, seg_en, output1}:  592  64   0  591
593{input1, seg_out, seg_en, output1}:  593  25   0  592
594{input1, seg_out, seg_en, output1}:  594  64   0  593
595{input1, seg_out, seg_en, output1}:  595 121   0  594
596{input1, seg_out, seg_en, output1}:  596  64   0  595
597{input1, seg_out, seg_en, output1}:  597  36   0  596
598{input1, seg_out, seg_en, output1}:  598  64   0  597
599{input1, seg_out, seg_en, output1}:  599 121   0  598
600{input1, seg_out, seg_en, output1}:  600  64   0  599
601{input1, seg_out, seg_en, output1}:  601  48   0  600
602{input1, seg_out, seg_en, output1}:  602  64   0  601
603{input1, seg_out, seg_en, output1}:  603 121   0  602
604{input1, seg_out, seg_en, output1}:  604  64   0  603
605{input1, seg_out, seg_en, output1}:  605  36   0  604
606{input1, seg_out, seg_en, output1}:  606  64   0  605
607{input1, seg_out, seg_en, output1}:  607 121   0  606
608{input1, seg_out, seg_en, output1}:  608  64   0  607
609{input1, seg_out, seg_en, output1}:  609  18   0  608
610{input1, seg_out, seg_en, output1}:  610  64   0  609
611{input1, seg_out, seg_en, output1}:  611 121   0  610
612{input1, seg_out, seg_en, output1}:  612  64   0  611
613{input1, seg_out, seg_en, output1}:  613  36   0  612
614{input1, seg_out, seg_en, output1}:  614  64   0  613
615{input1, seg_out, seg_en, output1}:  615 121   0  614
616{input1, seg_out, seg_en, output1}:  616  64   0  615
617{input1, seg_out, seg_en, output1}:  617  48   0  616
618{input1, seg_out, seg_en, output1}:  618  64   0  617
619{input1, seg_out, seg_en, output1}:  619 121   0  618
620{input1, seg_out, seg_en, output1}:  620  64   0  619
```

all of the code and waves is correct, after my calculate all of them are right.

Decibel of wave:

oh!

What a beautiful wave you are!

You repeat dozens of times

Every time obey the same law.

solemn and respectful, dignified and stately

a wonderful product of vivado!

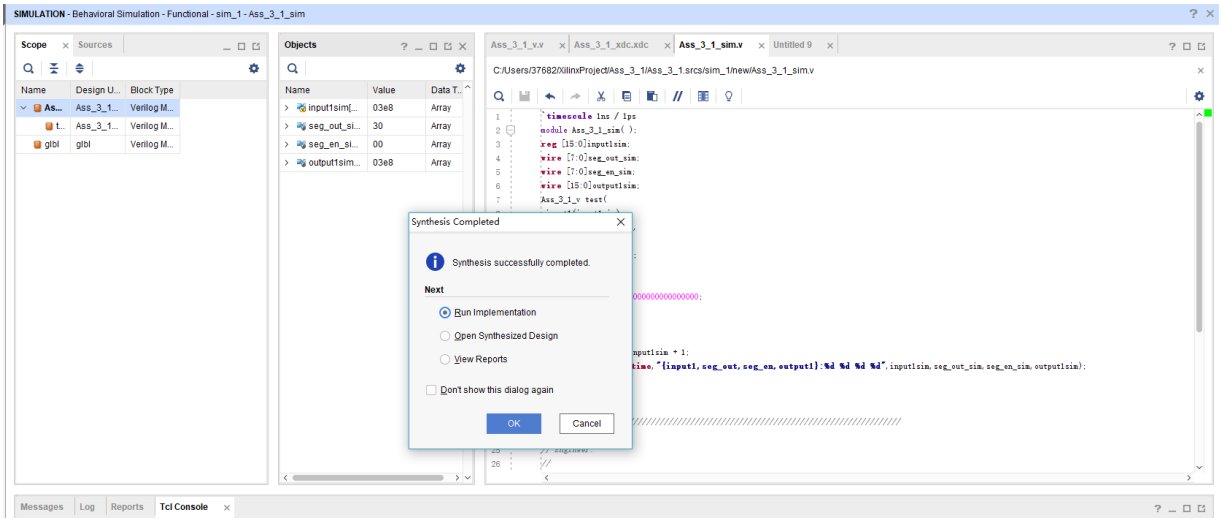
The waves are correct after verification.

All of the result is correct after my checking.

FINALLY all of the code and waves is correct, after my calculate all of them are right.

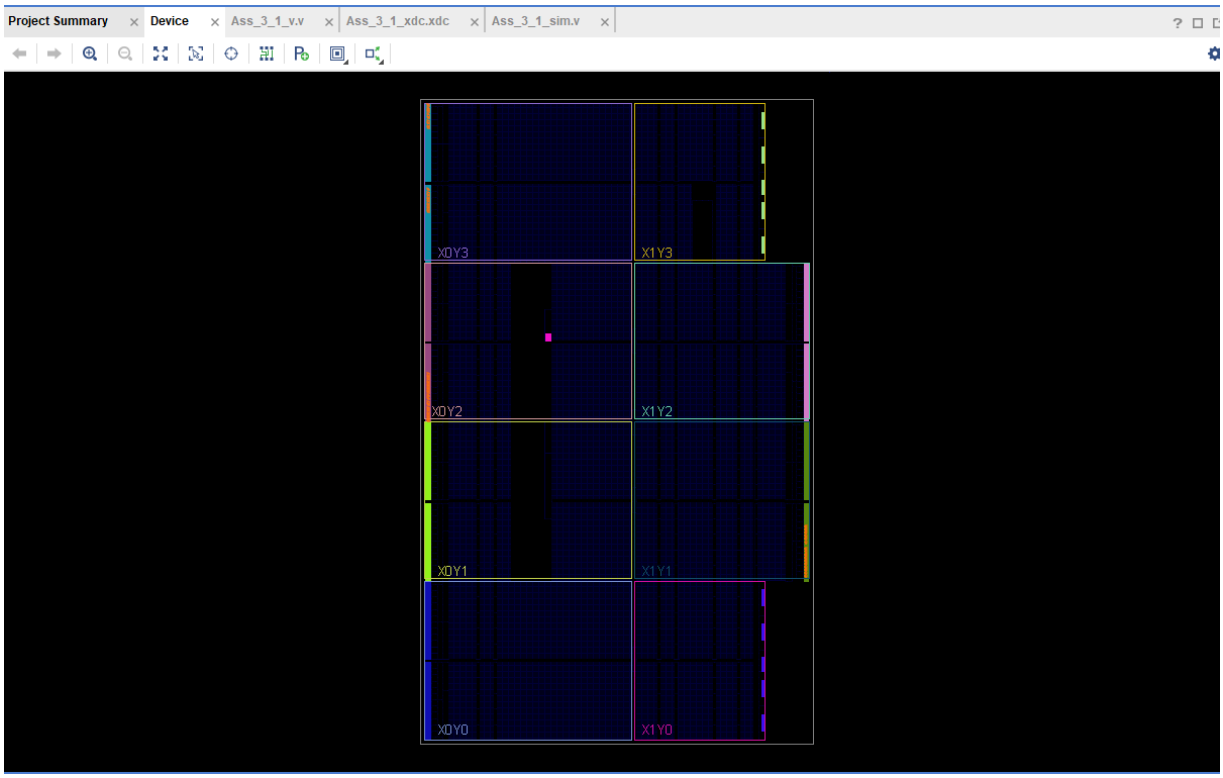
CONSTRAINT FILE AND THE TEST

In this part I will do the synthesis, implementation(include the I/O ports set), generate bitstream and burn it in the board.



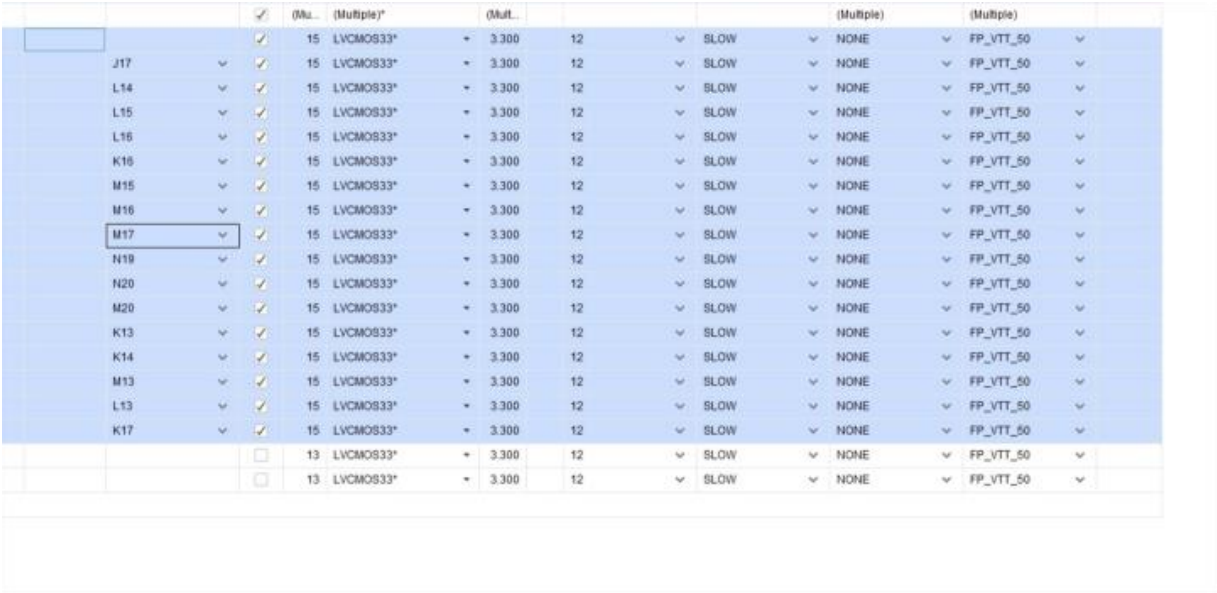
In this part choose to open synthesized design

.

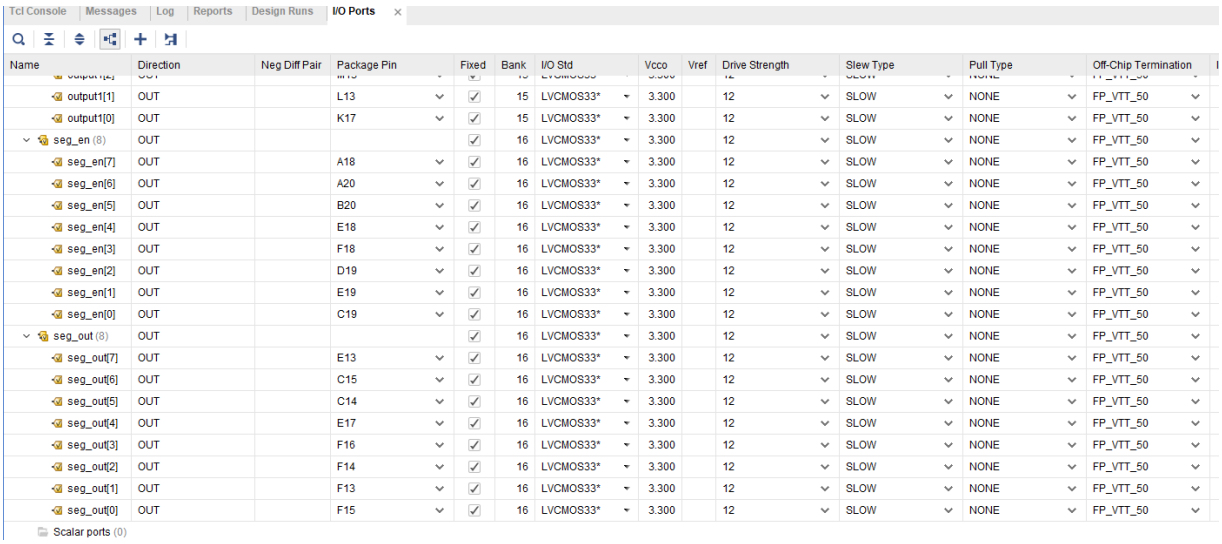


Tcl Console | Messages | Log | Reports | Design Runs | I/O Ports

Name	Direction	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Drive Strength	Slew Type	Pull Type	Off-Chip Termination	IN_TERM
All ports (48)													
input1 (16)	IN			☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[15]	IN		V5	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[14]	IN		R6	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[13]	IN		T6	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[12]	IN		Y6	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[11]	IN		AA6	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[10]	IN		V7	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[9]	IN		AB7	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[8]	IN		AB6	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[7]	IN		V9	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[6]	IN		V8	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[5]	IN		AA8	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[4]	IN		AB8	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[3]	IN		Y8	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[2]	IN		Y7	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[1]	IN		W9	☑	34	LVCMOS33*	3.300				NONE	NONE	
input1[0]	IN		Y9	☑	34	LVCMOS33*	3.300				NONE	NONE	
output1 (16)	OUT			☑	15	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
output1[15]	OUT		J17	☑	15	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
output1[14]	OUT		L14	☑	15	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	



			(Mu...	(Multiple)*	(Mult...				(Multiple)	(Multiple)
		✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	J17	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	L14	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	L15	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	L16	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	K16	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	M15	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	M16	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	M17	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	N19	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	N20	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	M20	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	K13	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	K14	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	M13	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	L13	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
	K17	✓	15	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
			13	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50
			13	LVCMOS33*	3.300	12	SLOW		NONE	FP_VTT_50



Tcl Console | Messages | Log | Reports | Design Runs | I/O Ports

Name	Direction	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Drive Strength	Slew Type	Pull Type	Off-Chip Termination
output1[1]	OUT		L13	✓	15	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
output1[0]	OUT		K17	✓	15	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_en (8)	OUT			✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_en[7]	OUT		A18	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_en[6]	OUT		A20	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_en[5]	OUT		B20	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_en[4]	OUT		E18	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_en[3]	OUT		F18	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_en[2]	OUT		D19	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_en[1]	OUT		E19	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_en[0]	OUT		C19	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_out (8)	OUT			✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_out[7]	OUT		E13	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_out[6]	OUT		C15	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_out[5]	OUT		C14	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_out[4]	OUT		E17	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_out[3]	OUT		F16	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_out[2]	OUT		F14	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_out[1]	OUT		F13	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
seg_out[0]	OUT		F15	✓	16	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50
Scalar ports (0)												

And the xdc files code is:

```
set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[7]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[6]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[5]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[4]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[3]}]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[2]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[1]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[0]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[7]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[6]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[5]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[4]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[3]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[2]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[1]}]

set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[0]}]

set_property PACKAGE_PIN C19 [get_ports {seg_en[0]}]

set_property PACKAGE_PIN E19 [get_ports {seg_en[1]}]

set_property PACKAGE_PIN D19 [get_ports {seg_en[2]}]

set_property PACKAGE_PIN F18 [get_ports {seg_en[3]}]

set_property PACKAGE_PIN E18 [get_ports {seg_en[4]}]

set_property PACKAGE_PIN B20 [get_ports {seg_en[5]}]

set_property PACKAGE_PIN A20 [get_ports {seg_en[6]}]

set_property PACKAGE_PIN A18 [get_ports {seg_en[7]}]
```

```
set_property PACKAGE_PIN F15 [get_ports {seg_out[0]}]

set_property PACKAGE_PIN F13 [get_ports {seg_out[1]}]

set_property PACKAGE_PIN F14 [get_ports {seg_out[2]}]

set_property PACKAGE_PIN F16 [get_ports {seg_out[3]}]


set_property PACKAGE_PIN E17 [get_ports {seg_out[4]}]

set_property PACKAGE_PIN C14 [get_ports {seg_out[5]}]

set_property PACKAGE_PIN C15 [get_ports {seg_out[6]}]

set_property PACKAGE_PIN E13 [get_ports {seg_out[7]}]


set_property IOSTANDARD LVCMOS33 [get_ports {input1[15]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[14]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[13]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[12]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[11]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[10]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[9]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[8]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[7]}]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports {input1[6]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[5]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[4]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[3]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[2]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[1]}]

set_property IOSTANDARD LVCMOS33 [get_ports {input1[0]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[15]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[14]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[13]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[12]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[11]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[10]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[9]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[8]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[7]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[6]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[5]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[4]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[3]}]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports {output1[2]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[1]}]

set_property IOSTANDARD LVCMOS33 [get_ports {output1[0]}]

set_property PACKAGE_PIN V5 [get_ports {input1[15]}]

set_property PACKAGE_PIN R6 [get_ports {input1[14]}]

set_property PACKAGE_PIN T6 [get_ports {input1[13]}]

set_property PACKAGE_PIN Y6 [get_ports {input1[12]}]

set_property PACKAGE_PIN AA6 [get_ports {input1[11]}]

set_property PACKAGE_PIN V7 [get_ports {input1[10]}]

set_property PACKAGE_PIN AB7 [get_ports {input1[9]}]

set_property PACKAGE_PIN AB6 [get_ports {input1[8]}]

set_property PACKAGE_PIN V9 [get_ports {input1[7]}]

set_property PACKAGE_PIN A8 [get_ports {input1[6]}]

set_property PACKAGE_PIN AA8 [get_ports {input1[5]}]

set_property PACKAGE_PIN AB8 [get_ports {input1[4]}]

set_property PACKAGE_PIN Y8 [get_ports {input1[3]}]

set_property PACKAGE_PIN Y7 [get_ports {input1[2]}]

set_property PACKAGE_PIN W9 [get_ports {input1[1]}]

set_property PACKAGE_PIN Y9 [get_ports {input1[0]}]

set_property PACKAGE_PIN K17 [get_ports {output1[0]}]
```

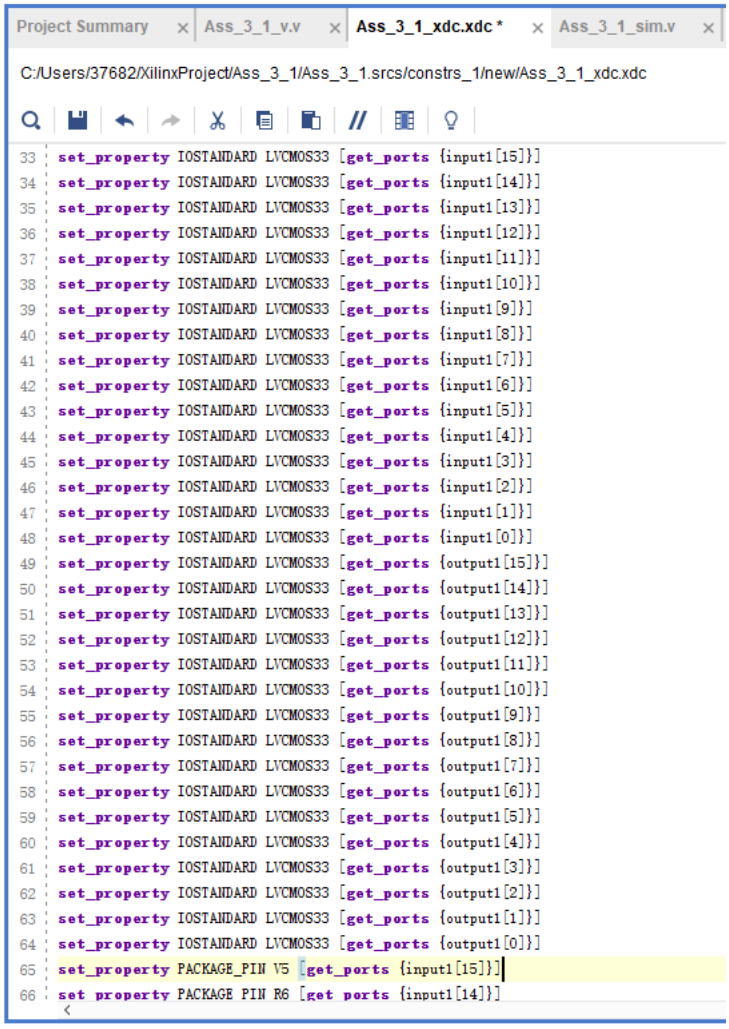
```
set_property PACKAGE_PIN L13 [get_ports {output1[1]}]

set_property PACKAGE_PIN M13 [get_ports {output1[2]}]

set_property PACKAGE_PIN K14 [get_ports {output1[3]}]

set_property PACKAGE_PIN K13 [get_ports {output1[4]}]

set_property PACKAGE_PIN M20 [get_ports {output1[5]}]

set_property PACKAGE_PIN N20 [get_ports {output1[6]}]

set_property PACKAGE_PIN N19 [get_ports {output1[7]}]

set_property PACKAGE_PIN M17 [get_ports {output1[8]}]

set_property PACKAGE_PIN M16 [get_ports {output1[9]}]

set_property PACKAGE_PIN M15 [get_ports {output1[10]}]

set_property PACKAGE_PIN K16 [get_ports {output1[11]}]

set_property PACKAGE_PIN L16 [get_ports {output1[12]}]

set_property PACKAGE_PIN L15 [get_ports {output1[13]}]

set_property PACKAGE_PIN L14 [get_ports {output1[14]}]

set_property PACKAGE_PIN J17 [get_ports {output1[15]}]
```

And the Graph:

Project Summary × Ass_3_1_v.v × **Ass_3_1_xdc.xdc *** × Ass_3_1_sim.v ×

C:/Users/37682/XilinxProject/Ass_3_1/Ass_3_1.srcs/constrs_1/new/Ass_3_1_xdc.xdc

```
1  set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[7]}]
2  set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[6]}]
3  set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[5]}]
4  set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[4]}]
5  set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[3]}]
6  set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[2]}]
7  set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[1]}]
8  set_property IOSTANDARD LVCMOS33 [get_ports {seg_en[0]}]
9  set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[7]}]
10 set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[6]}]
11 set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[5]}]
12 set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[4]}]
13 set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[3]}]
14 set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[2]}]
15 set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[1]}]
16 set_property IOSTANDARD LVCMOS33 [get_ports {seg_out[0]}]
17 set_property PACKAGE_PIN C19 [get_ports {seg_en[0]}]
18 set_property PACKAGE_PIN E19 [get_ports {seg_en[1]}]
19 set_property PACKAGE_PIN D19 [get_ports {seg_en[2]}]
20 set_property PACKAGE_PIN F18 [get_ports {seg_en[3]}]
21 set_property PACKAGE_PIN E18 [get_ports {seg_en[4]}]
22 set_property PACKAGE_PIN B20 [get_ports {seg_en[5]}]
23 set_property PACKAGE_PIN A20 [get_ports {seg_en[6]}]
24 set_property PACKAGE_PIN A18 [get_ports {seg_en[7]}]
25 set_property PACKAGE_PIN F15 [get_ports {seg_out[0]}]
26 set_property PACKAGE_PIN F13 [get_ports {seg_out[1]}]
27 set_property PACKAGE_PIN F14 [get_ports {seg_out[2]}]
28 set_property PACKAGE_PIN F16 [get_ports {seg_out[3]}]
29 set_property PACKAGE_PIN E17 [get_ports {seg_out[4]}]
30 set_property PACKAGE_PIN C14 [get_ports {seg_out[5]}]
31 set_property PACKAGE_PIN C15 [get_ports {seg_out[6]}]
32 set_property PACKAGE_PIN E13 [get_ports {seg_out[7]}]
33 set_property IOSTANDARD LVCMOS33 [get_ports {input1[15]}]
34 set_property IOSTANDARD LVCMOS33 [get_ports {input1[14]}]
```


Project Summary
Ass_3_1_v.v
Ass_3_1_xdc.xdc *
Ass_3_1_sim.v
C:/Users/37682/XilinxProject/Ass_3_1/Ass_3_1.srcs/constrs_1/new/Ass_3_1_xdc.xdc
33 set_property IOSTANDARD LVCMOS33 [get_ports {input1[15]}]
34 set_property IOSTANDARD LVCMOS33 [get_ports {input1[14]}]
35 set_property IOSTANDARD LVCMOS33 [get_ports {input1[13]}]
36 set_property IOSTANDARD LVCMOS33 [get_ports {input1[12]}]
37 set_property IOSTANDARD LVCMOS33 [get_ports {input1[11]}]
38 set_property IOSTANDARD LVCMOS33 [get_ports {input1[10]}]
39 set_property IOSTANDARD LVCMOS33 [get_ports {input1[9]}]
40 set_property IOSTANDARD LVCMOS33 [get_ports {input1[8]}]
41 set_property IOSTANDARD LVCMOS33 [get_ports {input1[7]}]
42 set_property IOSTANDARD LVCMOS33 [get_ports {input1[6]}]
43 set_property IOSTANDARD LVCMOS33 [get_ports {input1[5]}]
44 set_property IOSTANDARD LVCMOS33 [get_ports {input1[4]}]
45 set_property IOSTANDARD LVCMOS33 [get_ports {input1[3]}]
46 set_property IOSTANDARD LVCMOS33 [get_ports {input1[2]}]
47 set_property IOSTANDARD LVCMOS33 [get_ports {input1[1]}]
48 set_property IOSTANDARD LVCMOS33 [get_ports {input1[0]}]
49 set_property IOSTANDARD LVCMOS33 [get_ports {output1[15]}]
50 set_property IOSTANDARD LVCMOS33 [get_ports {output1[14]}]
51 set_property IOSTANDARD LVCMOS33 [get_ports {output1[13]}]
52 set_property IOSTANDARD LVCMOS33 [get_ports {output1[12]}]
53 set_property IOSTANDARD LVCMOS33 [get_ports {output1[11]}]
54 set_property IOSTANDARD LVCMOS33 [get_ports {output1[10]}]
55 set_property IOSTANDARD LVCMOS33 [get_ports {output1[9]}]
56 set_property IOSTANDARD LVCMOS33 [get_ports {output1[8]}]
57 set_property IOSTANDARD LVCMOS33 [get_ports {output1[7]}]
58 set_property IOSTANDARD LVCMOS33 [get_ports {output1[6]}]
59 set_property IOSTANDARD LVCMOS33 [get_ports {output1[5]}]
60 set_property IOSTANDARD LVCMOS33 [get_ports {output1[4]}]
61 set_property IOSTANDARD LVCMOS33 [get_ports {output1[3]}]
62 set_property IOSTANDARD LVCMOS33 [get_ports {output1[2]}]
63 set_property IOSTANDARD LVCMOS33 [get_ports {output1[1]}]
64 set_property IOSTANDARD LVCMOS33 [get_ports {output1[0]}]
65 set_property PACKAGE_PIN V5 [get_ports {input1[15]}]
66 set_property PACKAGE_PIN R6 [get_ports {input1[14]}]

Project Summary | Ass_3_1_v.v | Ass_3_1_xdc.xdc * | Ass_3_1_sim.v

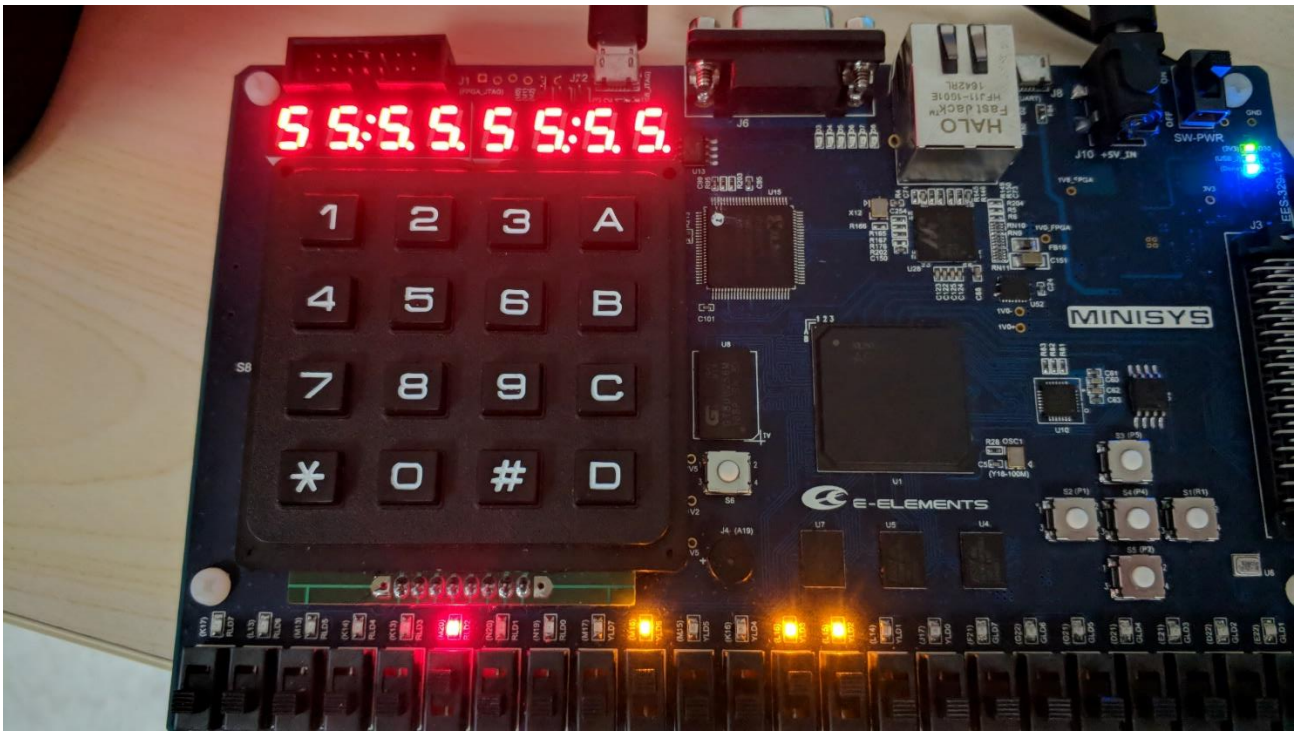
C:/Users/37682/XilinxProject/Ass_3_1/Ass_3_1.srcs/constrs_1/new/Ass_3_1_xdc.xdc

```
set_property IOSTANDARD LVCMOS33 [get_ports {output1[0]}]
set_property PACKAGE_PIN V5 [get_ports {input1[15]}]
set_property PACKAGE_PIN R6 [get_ports {input1[14]}]
set_property PACKAGE_PIN T6 [get_ports {input1[13]}]
set_property PACKAGE_PIN Y6 [get_ports {input1[12]}]
set_property PACKAGE_PIN AA6 [get_ports {input1[11]}]
set_property PACKAGE_PIN V7 [get_ports {input1[10]}]
set_property PACKAGE_PIN AB7 [get_ports {input1[9]}]
set_property PACKAGE_PIN AB6 [get_ports {input1[8]}]
set_property PACKAGE_PIN V9 [get_ports {input1[7]}]
set_property PACKAGE_PIN A8 [get_ports {input1[6]}]
set_property PACKAGE_PIN AA8 [get_ports {input1[5]}]
set_property PACKAGE_PIN AB8 [get_ports {input1[4]}]
set_property PACKAGE_PIN Y8 [get_ports {input1[3]}]
set_property PACKAGE_PIN Y7 [get_ports {input1[2]}]
set_property PACKAGE_PIN W9 [get_ports {input1[1]}]
set_property PACKAGE_PIN Y9 [get_ports {input1[0]}]
set_property PACKAGE_PIN K17 [get_ports {output1[0]}]
set_property PACKAGE_PIN L13 [get_ports {output1[1]}]
set_property PACKAGE_PIN M13 [get_ports {output1[2]}]
set_property PACKAGE_PIN K14 [get_ports {output1[3]}]
set_property PACKAGE_PIN K13 [get_ports {output1[4]}]
set_property PACKAGE_PIN M20 [get_ports {output1[5]}]
set_property PACKAGE_PIN N20 [get_ports {output1[6]}]
set_property PACKAGE_PIN N19 [get_ports {output1[7]}]
set_property PACKAGE_PIN M17 [get_ports {output1[8]}]
set_property PACKAGE_PIN M16 [get_ports {output1[9]}]
set_property PACKAGE_PIN M15 [get_ports {output1[10]}]
set_property PACKAGE_PIN K16 [get_ports {output1[11]}]
set_property PACKAGE_PIN L16 [get_ports {output1[12]}]
set_property PACKAGE_PIN L15 [get_ports {output1[13]}]
set_property PACKAGE_PIN L14 [get_ports {output1[14]}]
set_property PACKAGE_PIN J17 [get_ports {output1[15]}]
```

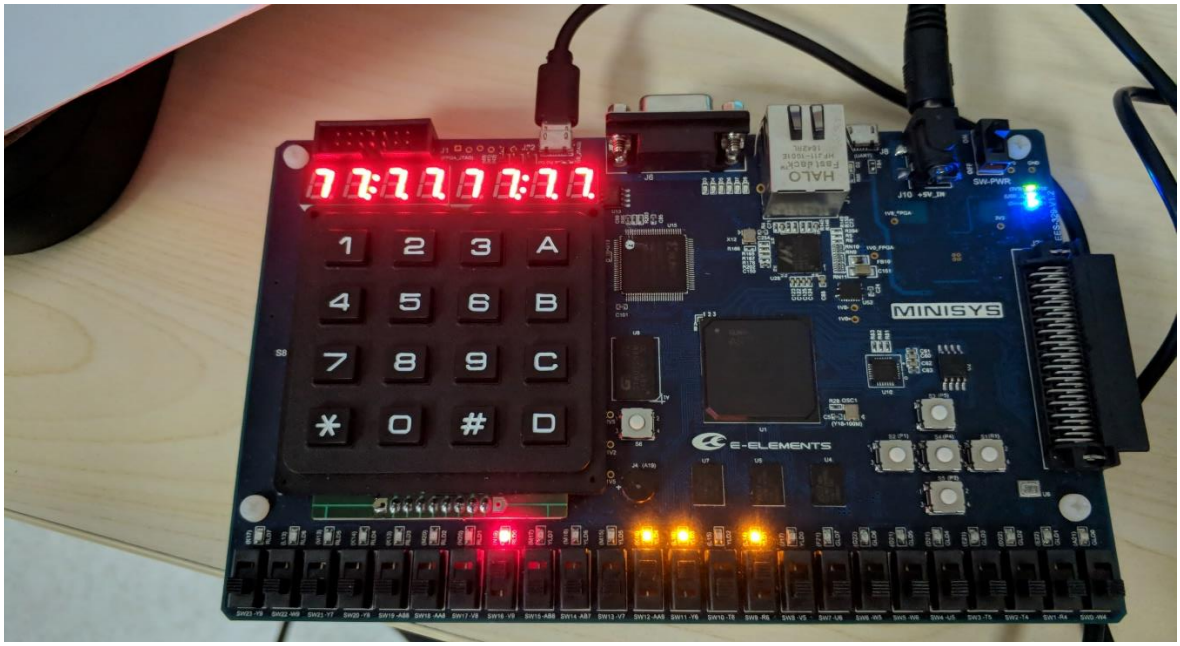
Then burn it in the board and get the result

RESULTS

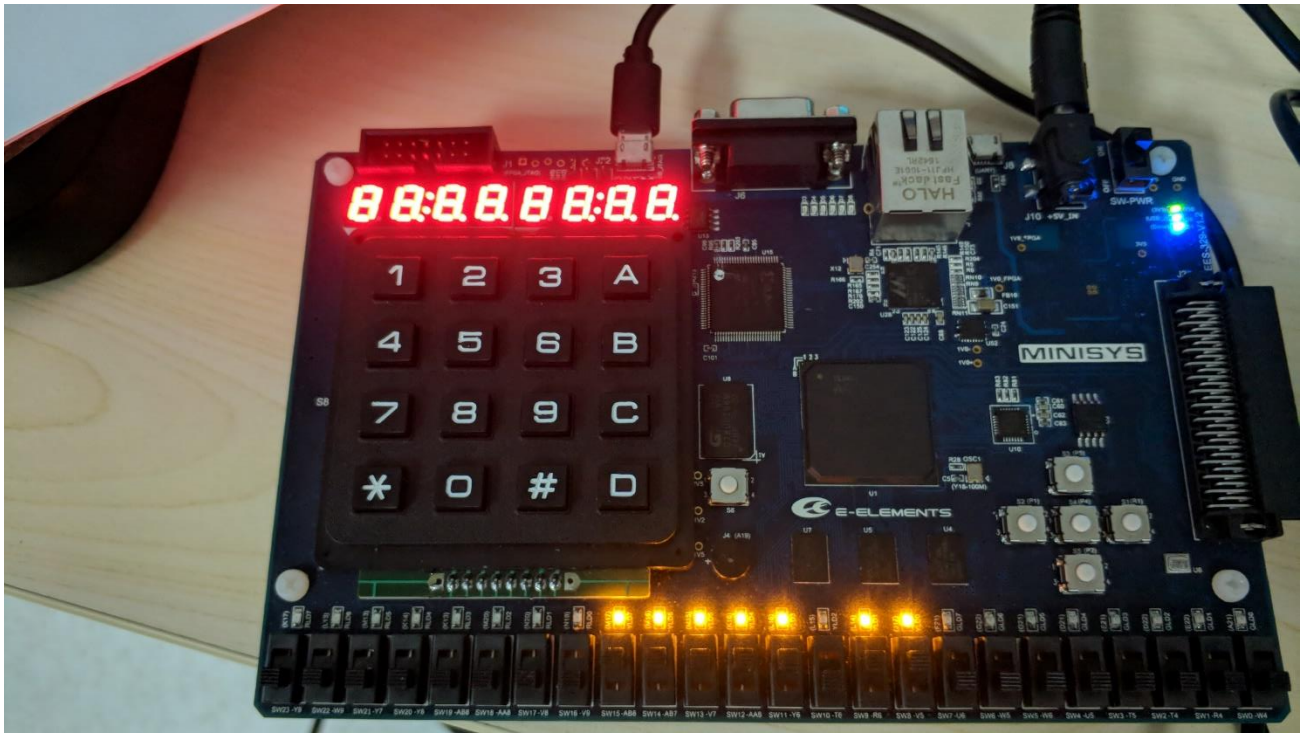
5(AA8) + 9(AB7) +C(Y6) + D(T6) → 5



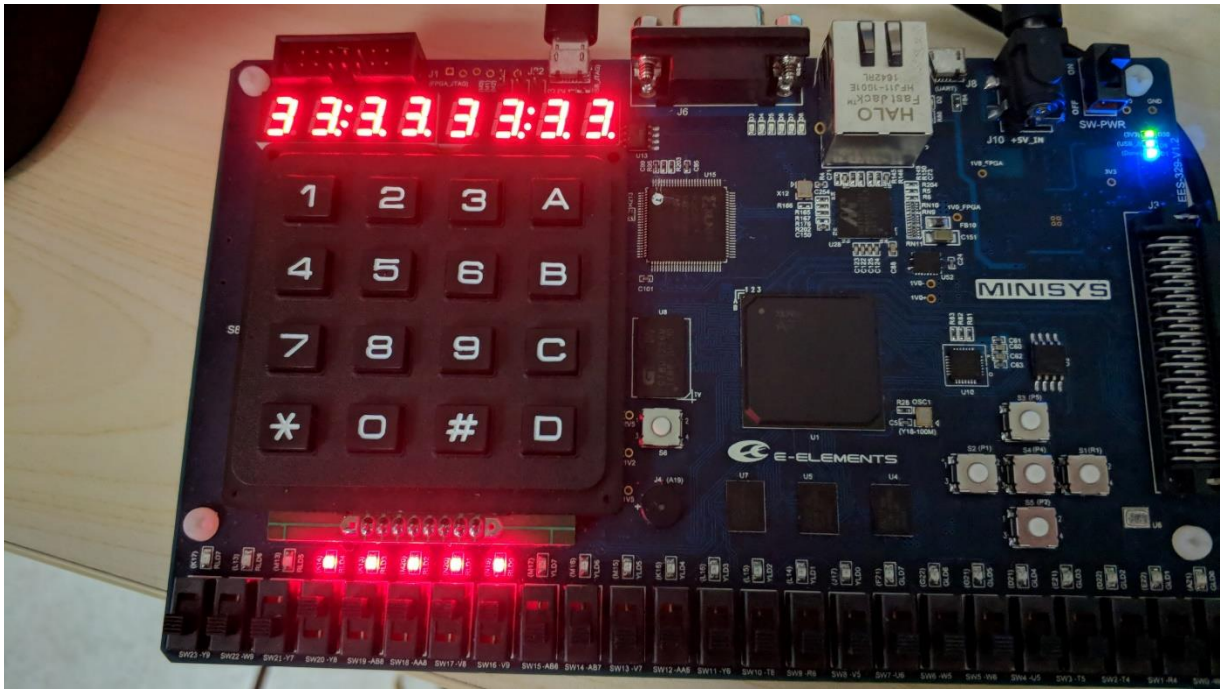
7(V9) + B(AA6) + C(Y6) +E(R6) → 7



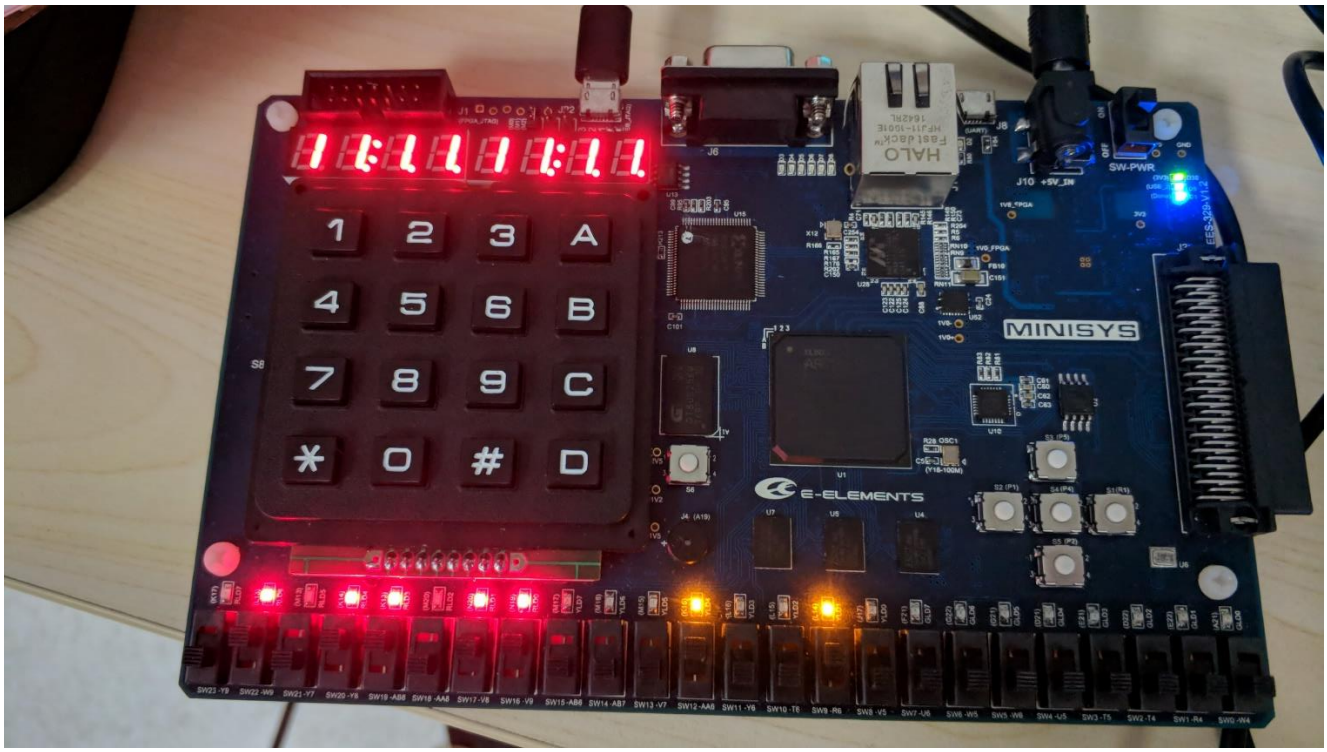
8(AB6)+9(AB7)+A(V7)+B(AA6)+C(Y6)+E(R6)+F(V5)→8



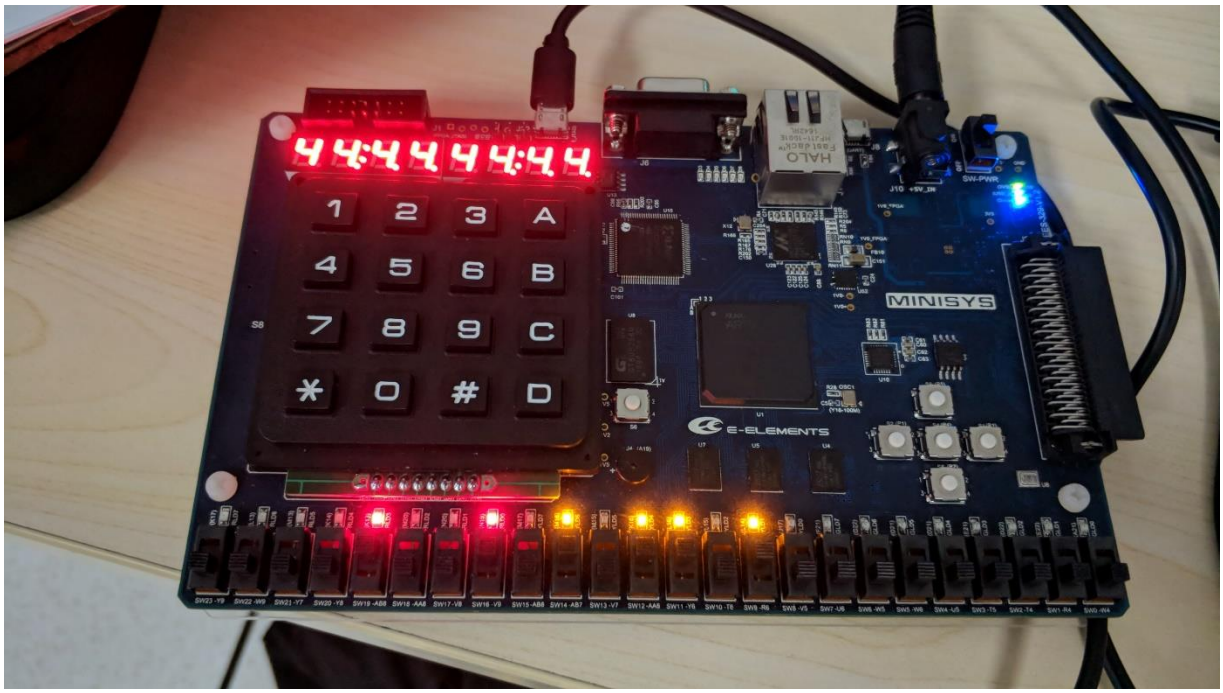
3(Y8)+4(AB8)+5(AA8)+6(V8)+7(V9)→3



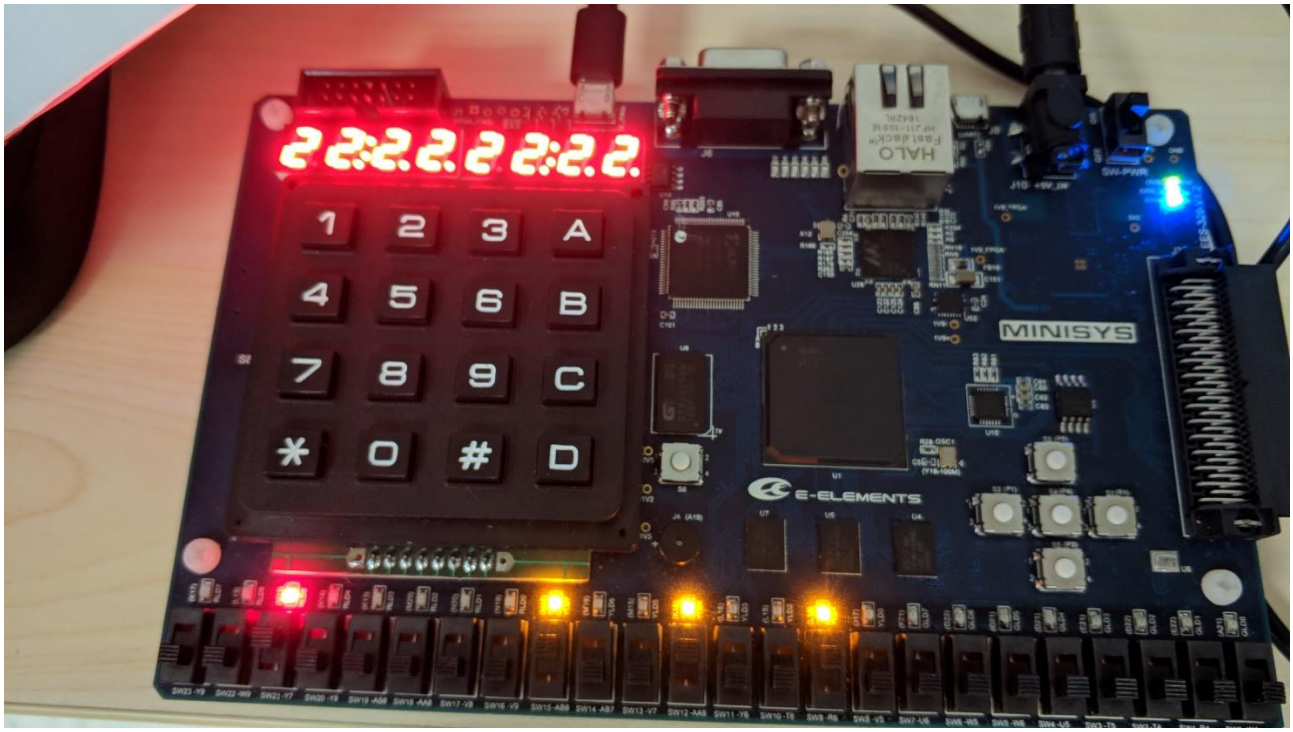
1(W9)+3(Y8)+4(AB8)+6(V8)+7(V9)+B(AA6)+E(R6)→1



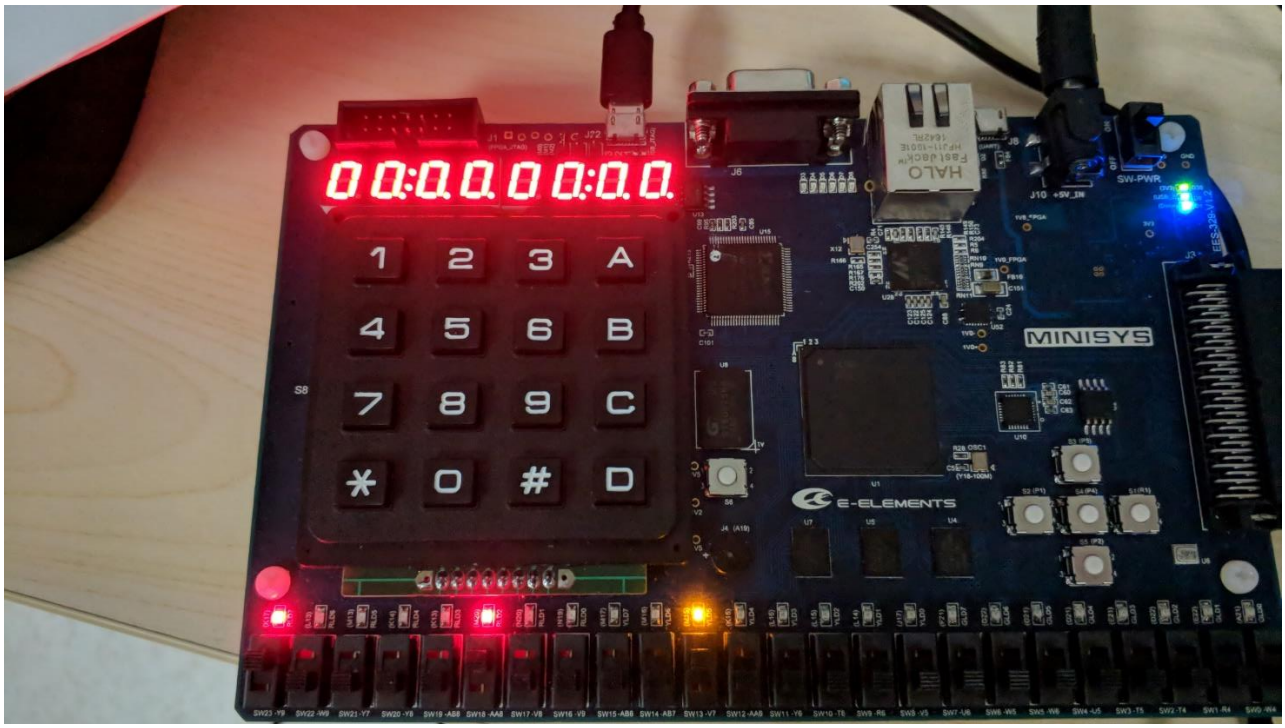
4(AB8)+7(V9)+9(AB7)+B(AA6)+C(Y6)+E(R6)→4



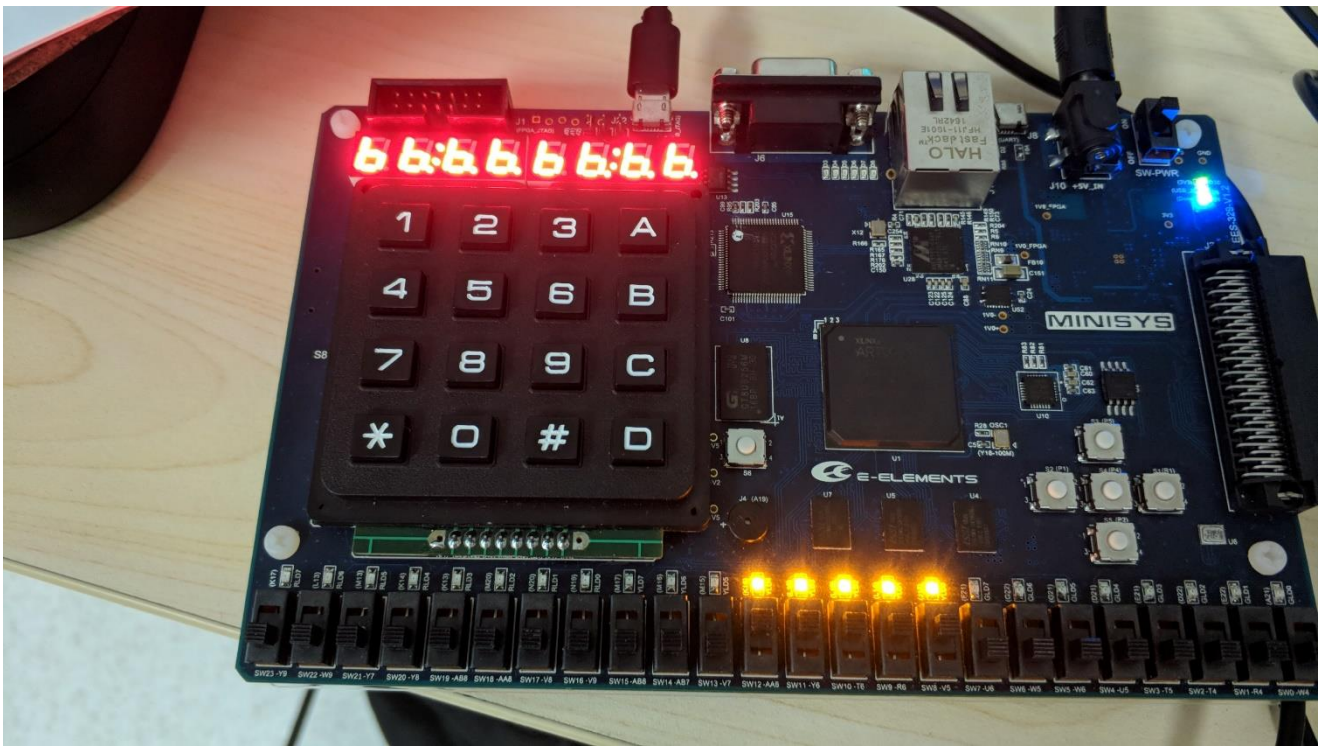
2(Y7)+8(AB6)+B(AA6)+E(R6)→2



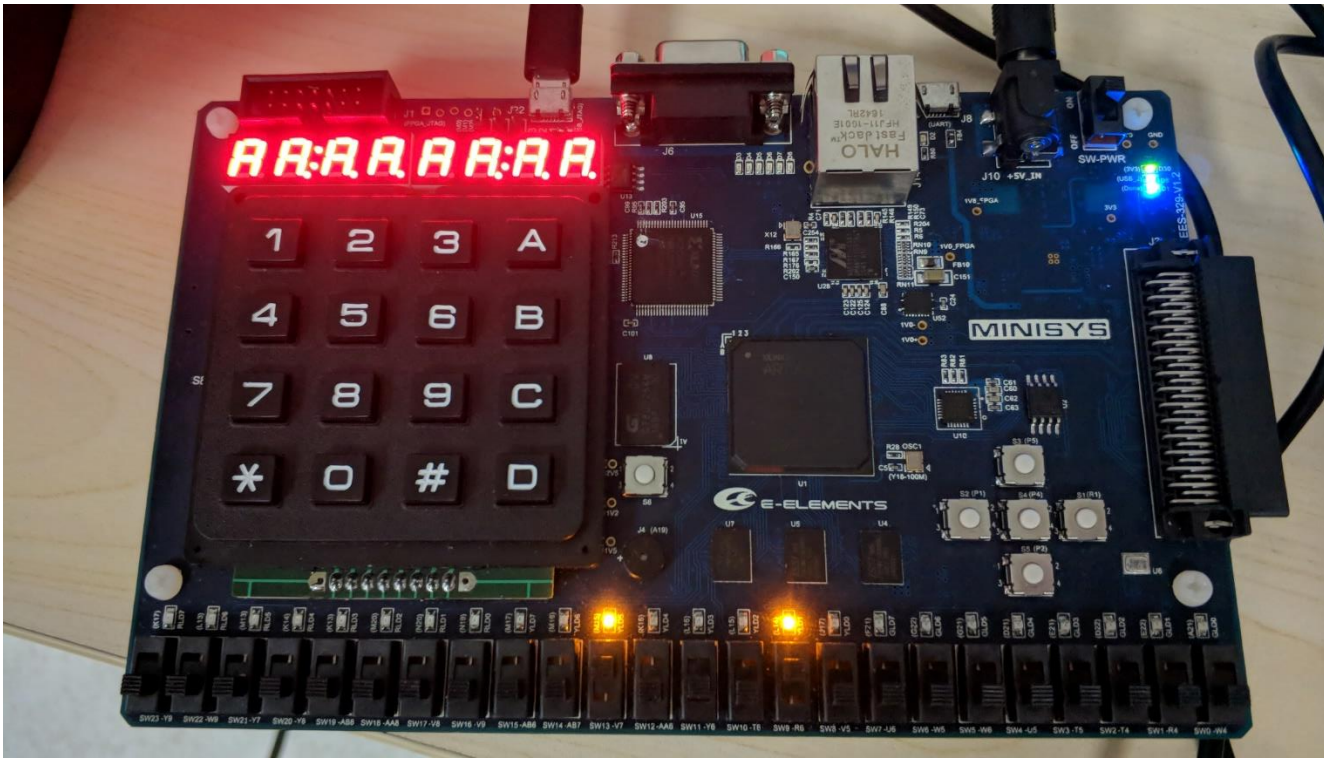
0(Y9)+5(AA8)+A(V7)→0



B(AA6)+C(Y6)+D(T6)+E(R6)+F(V5)→b



A(V7)+E(R6)→A



PROBLEMS AND SOLUTIONS

Task1:

1. In the task1's I/O ports it is hard to find how to full the nunbers, then first we can use the file that provide to produce an implement in another file and obverse what will happen then do the same as that one, or can just add the 16_bit input and output, just add the xdc_file in our own file.
2. Sometimes it will happen some strange things

   Solutions: close vivado and open it again or even reboot computer

ADDITION

Describe of waves and codes in Simulation is provide

Problems and solution is provided

Description of waves are provided

PART 2: DIGITAL DESIGN LAB (TASK2)

The words before codes and graph

1. The sop of the function is

   F = ABCD + AB'C'D+ AB'C'D' + AB'CD' + ABC'D +

   A'BCD + A'BCD'+ A'BC'D+ A'BC'D'+ A'B'CD+ A'B'CD'+ A'B'C'D+ A'B'C'D'

   = A' + AB'C' + AB'D'+ABD

   And the truth table is

A	B	C	D	F	
0	0	0	0	1	F= 1

0	0	0	1	1	
0	0	1	0	1	F= 1
0	0	1	1	1	
0	1	0	0	1	F= 1
0	1	0	1	1	
0	1	1	0	1	F= 1
0	1	1	1	1	
1	0	0	0	1	F= 1
1	0	0	1	1	
1	0	1	0	1	F=1
1	0	1	1	1	
1	1	0	0	0	F= 0
1	1	0	1	0	
1	1	1	0	0	F= D
1	1	1	1	1	

So the graph should like that

And in this part we first creat a 74151_ 8_to_1_line_multiplexer

Then we creat a center .v file to achieve the Boolean function by using the 74151_ 8_to_1_line_multiplexer.

And then write the testbench, do synthesis and run implementation and Create the constraint file.

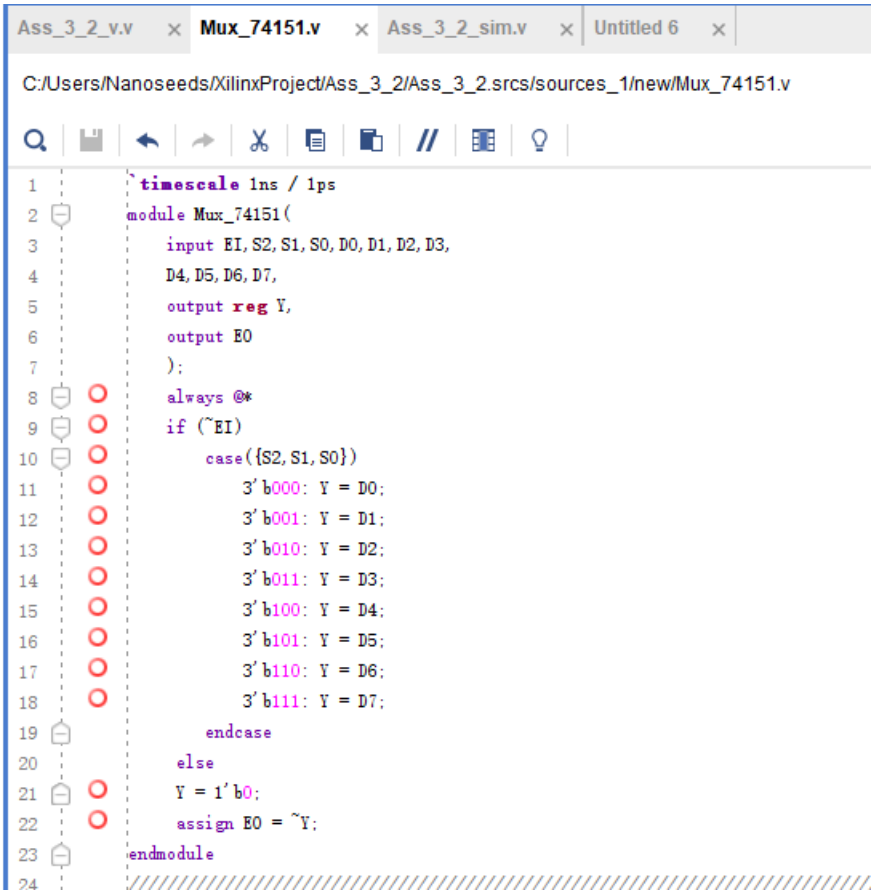
Finally burn the code into the develop board and take photos.

The Mux_74151's code:

```
`timescale 1ns / 1ps
module Mux_74151(
    input EI,S2,S1,S0,D0,D1,D2,D3,
    D4,D5,D6,D7,
    output reg Y,
    output EO
    );
    always @*
    if (~EI)
        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 EO = ~Y;



the center code:

`timescale 1ns / 1ps

module Ass_3_2_v(

input EI,

input A,B,C,D,

output outputa,outputb,outputc,outputd,

output F,

output EO

);

```
assign {outputa,outputb,outputc,outputd} = {A,B,C,D};
Mux_74151 Mux1(
.EI(EI),
.S2(A),
.S1(B),
.S0(C),
.D0(1'b1),
.D1(1'b1),
.D2(1'b1),
.D3(1'b1),
.D4(1'b1),
.D5(1'b1),
.D6(1'b0),
.D7(D),
.Y(F),
.EO(EO));
Endmodule
```

Ass_3_2_v.v | Mux_74151.v | Ass_3_2_sim.v | Untitled 6

C:/Users/Nanoseeds/XilinxProject/Ass_3_2/Ass_3_2.srcs/sources_1/new/Ass_3_2_v.v

```
1   `timescale 1ns / 1ps
2   module Ass_3_2_v(
3   input EI,
4   input A,B,C,D,
5   output outputa,outputb,outputc,outputd,
6   output F,
7   output EO
8       );
9       assign {outputa,outputb,outputc,outputd} = {A,B,C,D};
10      Mux_74151 Mux1(
11      .EI(EI),
12      .S2(A),
13      .S1(B),
14      .S0(C),
15      .D0(1'b1),
16      .D1(1'b1),
17      .D2(1'b1),
18      .D3(1'b1),
19      .D4(1'b1),
20      .D5(1'b1),
21      .D6(1'b0),
22      .D7(D),
23      .Y(F),
24      .E0(E0));
25  endmodule
26  //////////////////////////////////////////////////////////////////////////////////
```

all of the code and waves is correct, after my calculate all of them are right.

SIMULATION

I Code:

`timescale 1ns / 1ps

module Ass_3_2_sim(

);

reg EIsim,sima,simb,simc,simd;

wire outputasim,outputbsim,outputcsim,outputdsim;

```
wire Fsim,EOsim;
Ass_3_2_v test(
.EI(EIsim),
.A(sima),
.B(simb),
.C(simc),
.D(simd),
.outputa(outputasim),
.outputb(outputbsim),
.outputc(outputcsim),
.outputd(outputdsim),
.F(Fsim),
.EO(EOsim));
initial
    begin
    EIsim = 0;
    {sima,simb,simc,simd} = 4'b0000;
        repeat(16)
            begin
            #10
              {sima,simb,simc,simd} =    {sima,simb,simc,simd} +1;
```

```
$display($time,"{A,B,C,D,F,EO}:%d %d %d %d %d %d",sima,simb,simc,simd,Fsim,EOsim)
;

                    end

                EIsim = 1;

                    repeat(16)

                        begin

                        #10

                        {sima,simb,simc,simd} =    {sima,simb,simc,simd} +1;
$display($time,"{A,B,C,D,F,EO}:%d %d %d %d %d %d",sima,simb,simc,simd,Fsim,EOsim)
;

                        end

            end
```

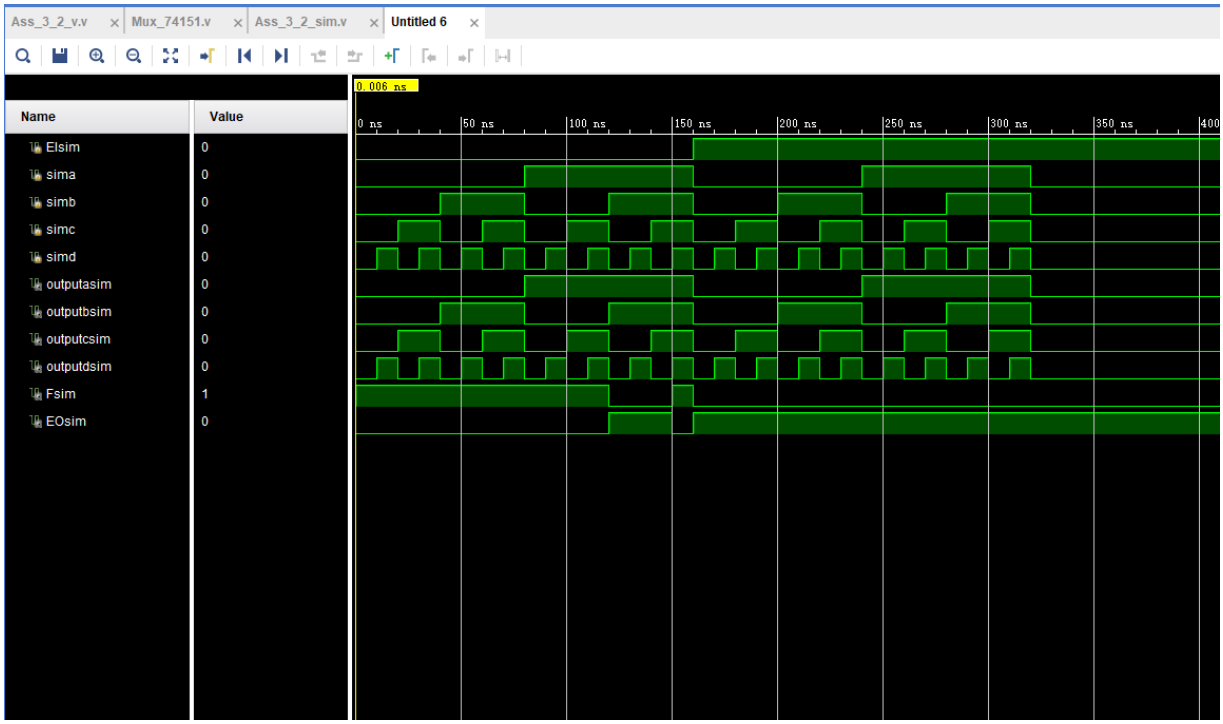
endmodule

Ass_3_2_v.v | Mux_74151.v | Ass_3_2_sim.v | Untitled 7

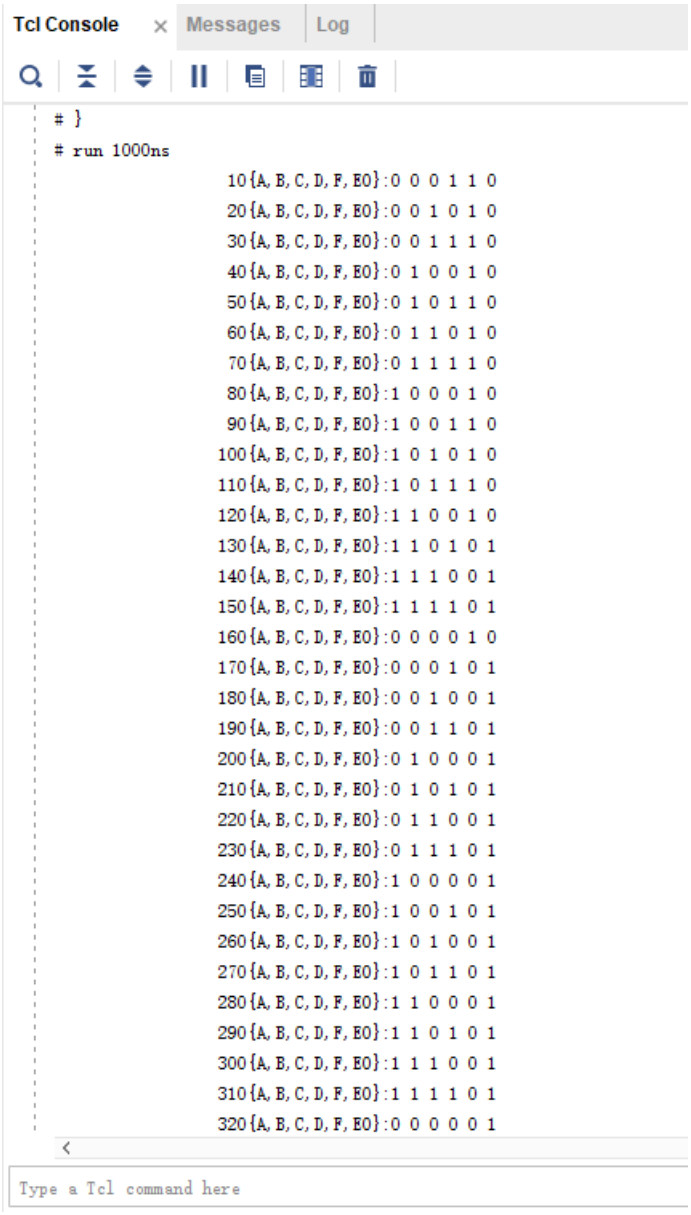
C:/Users/Nanoseeds/XilinxProject/Ass_3_2/Ass_3_2.srcs/sim_1/new/Ass_3_2_sim.v

```
1  `timescale 1ns / 1ps
2  module Ass_3_2_sim(
3      );
4
5      reg EIsim, sima, simb, simc, simd;
6      wire outputasim, outputbsim, outputcsim, outputdsim;
7      wire Fsim, EOsim;
8      Ass_3_2_v test(
9      .EI(EIsim),
10     .A(sima),
11     .B(simb),
12     .C(simc),
13     .D(simd),
14     .outputa(outputasim),
15     .outputb(outputbsim),
16     .outputc(outputcsim),
17     .outputd(outputdsim),
18     .F(Fsim),
19     .EO(EOsim));
20     initial
21         begin
22         EIsim = 0;
23         {sima, simb, simc, simd} = 4'b0000;
24             repeat(16)
25                 begin
26                 #10
27                  {sima, simb, simc, simd} =  {sima, simb, simc, simd} +1;
28                 $display($time, "{A,B,C,D,F,EO}:%d %d %d %d %d %d", sima, simb, simc, simd, Fsim, EOsim);
29                  end
30             EIsim = 1;
31                 repeat(16)
32                   begin
33                   #10
34                   {sima, simb, simc, simd} =  {sima, simb, simc, simd} +1;
35                 $display($time, "{A,B,C,D,F,EO}:%d %d %d %d %d %d", sima, simb, simc, simd, Fsim, EOsim);
36                    end
37         end
38 endmodule
39 //////////////////////////////////////////////////////////////////////////////////
40 // Company:
41 // Engineer:
```

Graph:



This is the simualation value in system task



The waves is correct, after my calculate all of them are right

Observer and Describe of wave:

old

What a beautiful wave you are!

You repeat dozens of times

Every time obey the same law.

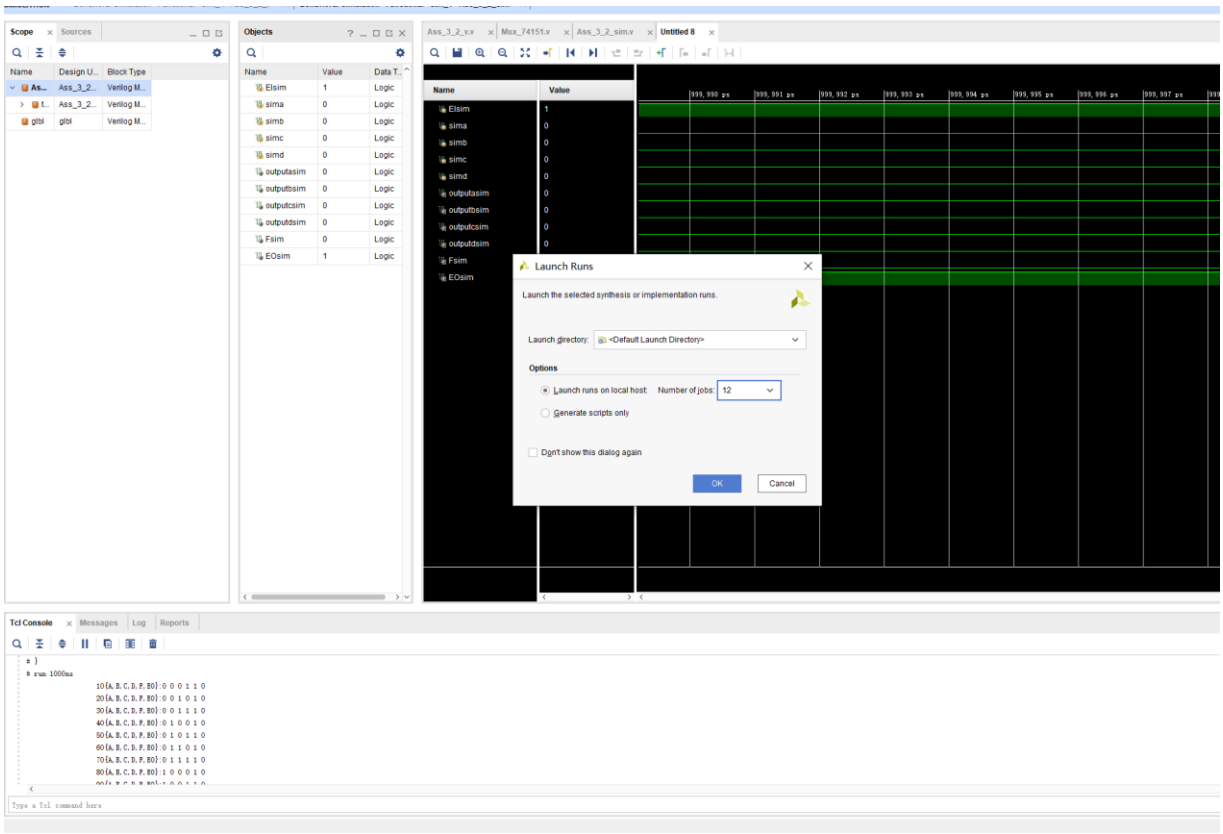
solemn and respectful, dignified and stately

a wonderful product of vivado!

The waves are correct after verification.

CONSTRAINT FILE AND THE TESTING

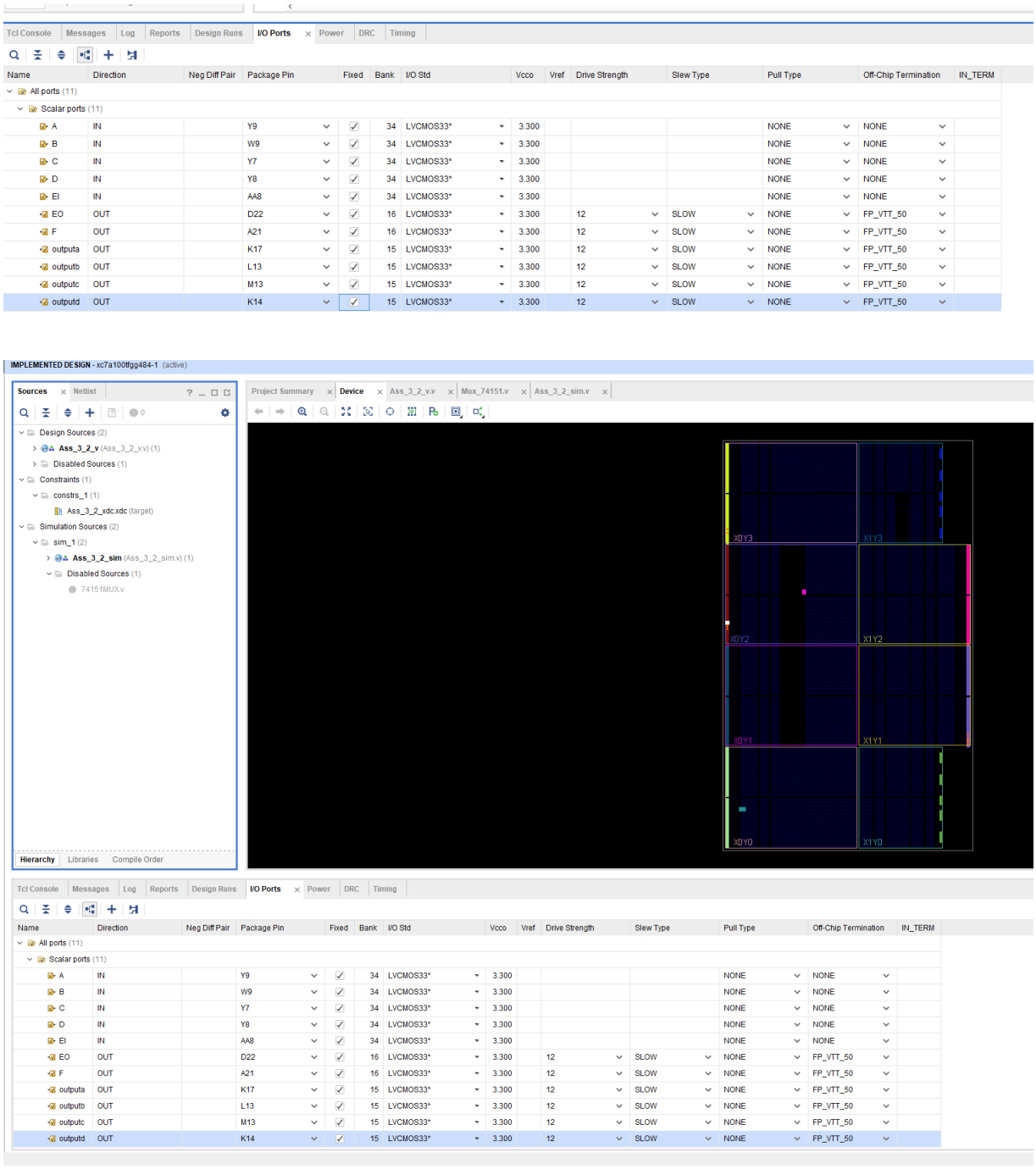
First of all is run synthesis and implementation:



In the I/O port step, we have five input and six output

A,B,C,D use the most left position, and EI input use sixth position, four of the output is used to show the input more directive so they use the lights corresponding to the switch.and the output F use the most right position's light. The EO output use the 3rd form the right light.

All of them use the LVCMOS33 that the vlotage is 3.3 volt



and the xdc files is

Codes:

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 EI]

set_property IOSTANDARD LVCMOS33 [get_ports EO]

set_property IOSTANDARD LVCMOS33 [get_ports F]

set_property IOSTANDARD LVCMOS33 [get_ports outputa]

set_property IOSTANDARD LVCMOS33 [get_ports outputb]

set_property IOSTANDARD LVCMOS33 [get_ports outputd]

set_property IOSTANDARD LVCMOS33 [get_ports outputc]

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 AA8 [get_ports EI]

set_property PACKAGE_PIN D22 [get_ports EO]

set_property PACKAGE_PIN A21 [get_ports F]

set_property PACKAGE_PIN K17 [get_ports outputa]

set_property PACKAGE_PIN L13 [get_ports outputb]
```

set_property PACKAGE_PIN M13 [get_ports outputc]

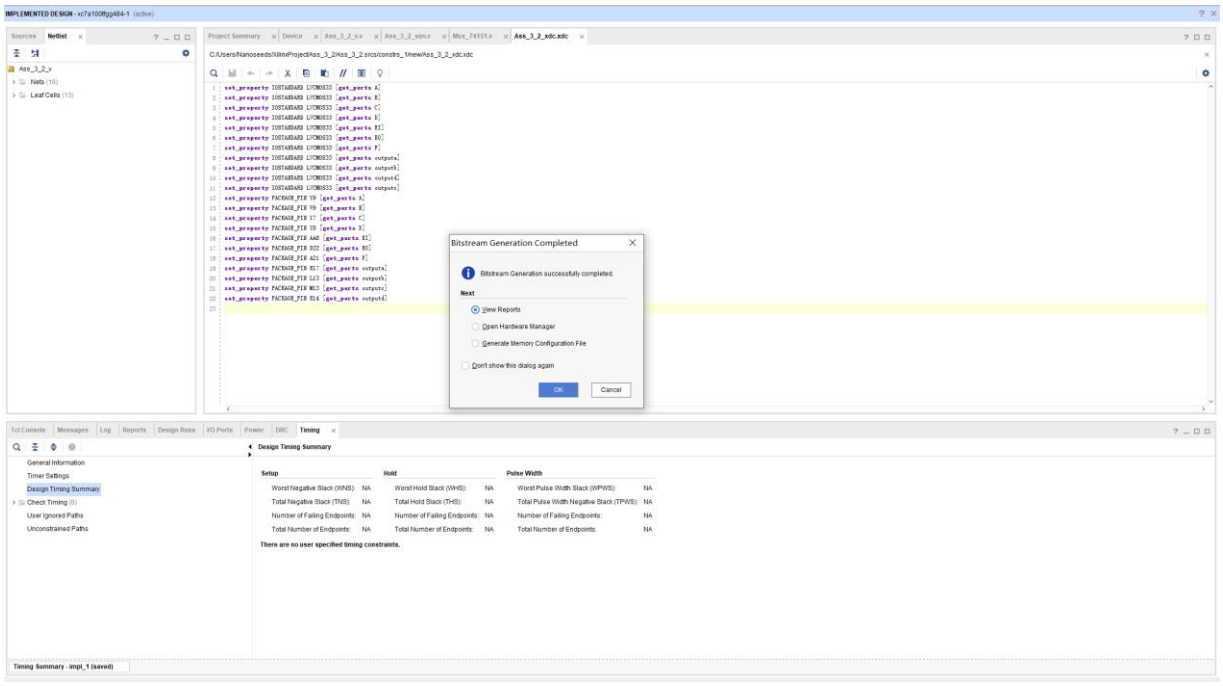
set_property PACKAGE_PIN K14 [get_ports outputd]

Project Summary | Device | Ass_3_2_v.v | Mux_74151.v | Ass_3_2_sim.v | **Ass_3_2_xdc.xdc**

C:/Users/Nanoseeds/XilinxProject/Ass_3_2/Ass_3_2.srcs/constrs_1/new/Ass_3_2_xdc.xdc

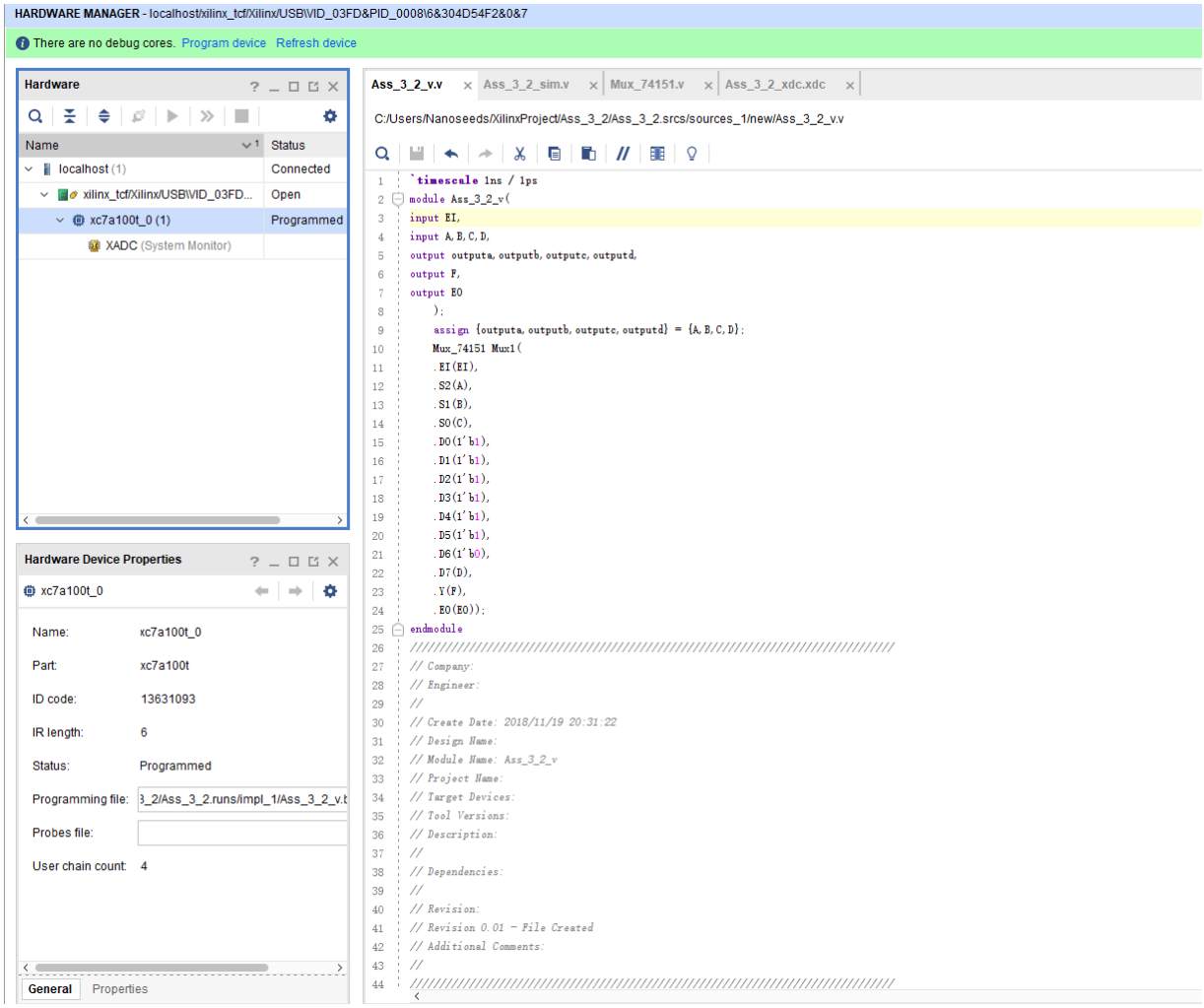
```
1  set_property IOSTANDARD LVCMOS33 [get_ports A]
2  set_property IOSTANDARD LVCMOS33 [get_ports B]
3  set_property IOSTANDARD LVCMOS33 [get_ports C]
4  set_property IOSTANDARD LVCMOS33 [get_ports D]
5  set_property IOSTANDARD LVCMOS33 [get_ports EI]
6  set_property IOSTANDARD LVCMOS33 [get_ports EO]
7  set_property IOSTANDARD LVCMOS33 [get_ports F]
8  set_property IOSTANDARD LVCMOS33 [get_ports outputa]
9  set_property IOSTANDARD LVCMOS33 [get_ports outputb]
10 set_property IOSTANDARD LVCMOS33 [get_ports outputd]
11 set_property IOSTANDARD LVCMOS33 [get_ports outputc]
12 set_property PACKAGE_PIN Y9 [get_ports A]
13 set_property PACKAGE_PIN W9 [get_ports B]
14 set_property PACKAGE_PIN Y7 [get_ports C]
15 set_property PACKAGE_PIN Y8 [get_ports D]
16 set_property PACKAGE_PIN AA8 [get_ports EI]
17 set_property PACKAGE_PIN D22 [get_ports EO]
18 set_property PACKAGE_PIN A21 [get_ports F]
19 set_property PACKAGE_PIN K17 [get_ports outputa]
20 set_property PACKAGE_PIN L13 [get_ports outputb]
21 set_property PACKAGE_PIN M13 [get_ports outputc]
22 set_property PACKAGE_PIN K14 [get_ports outputd]
23
```

then do the generate bitstream



and program the device

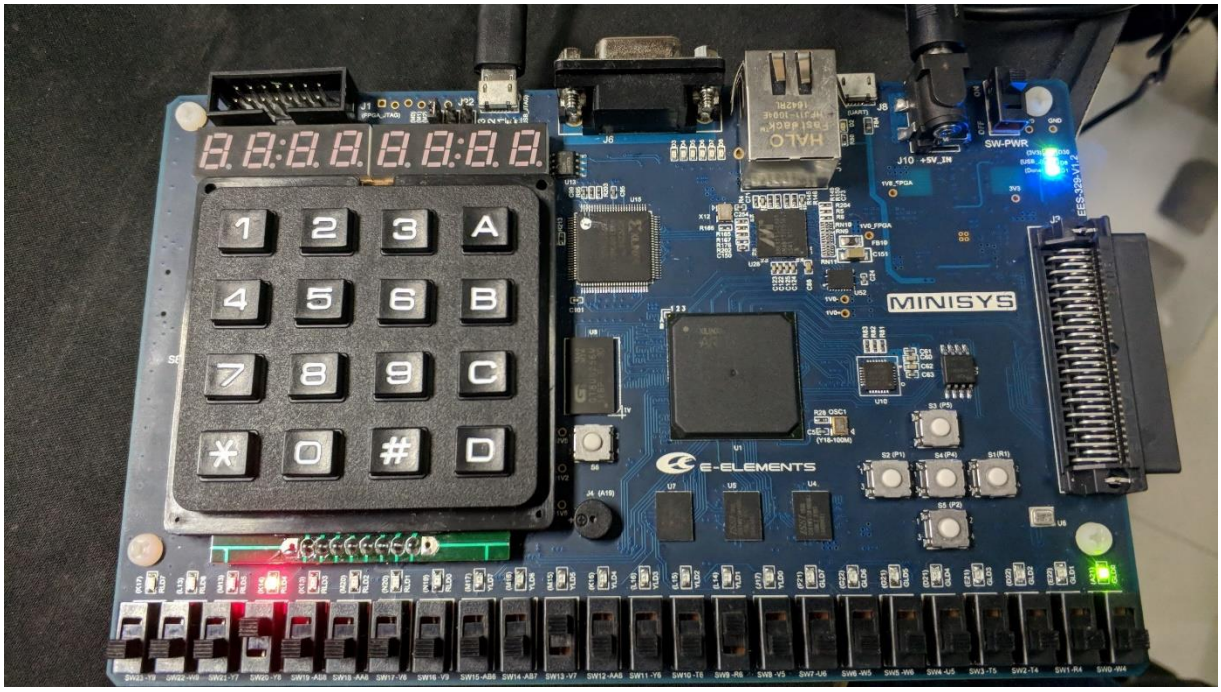
after many times of turn on and turn off the device finally be found by the vivado



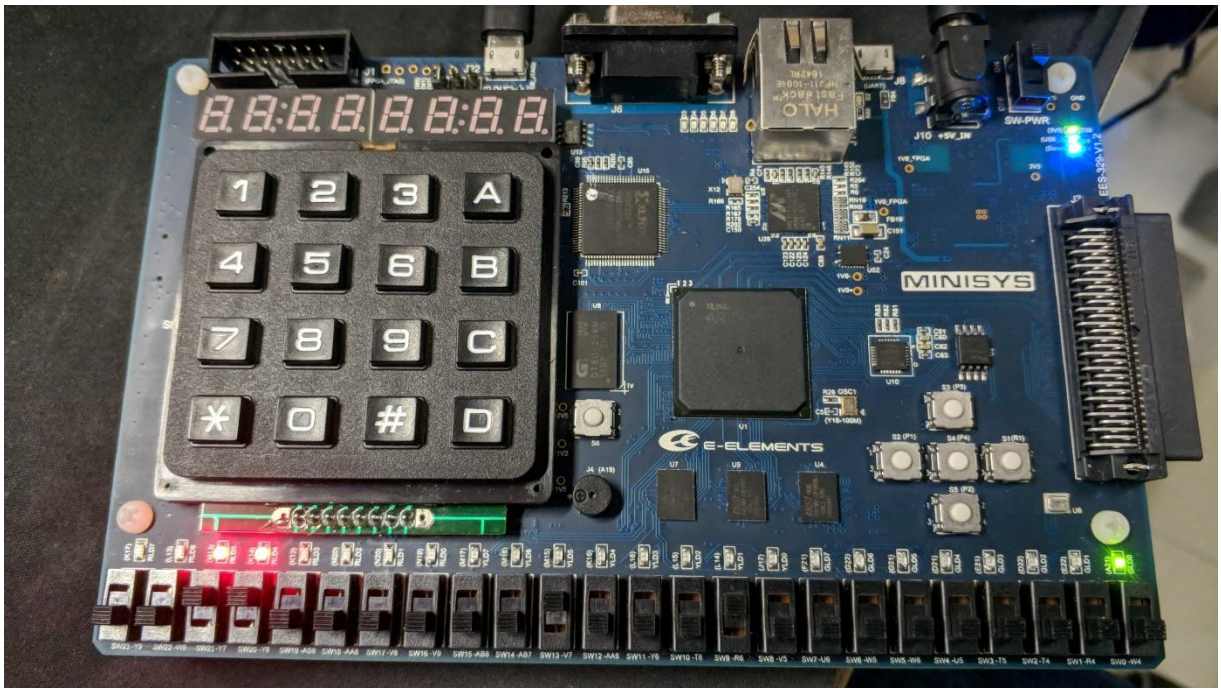
CONSTRAINT FILE AND THE TESTING

Next pictures are the develop board test cases

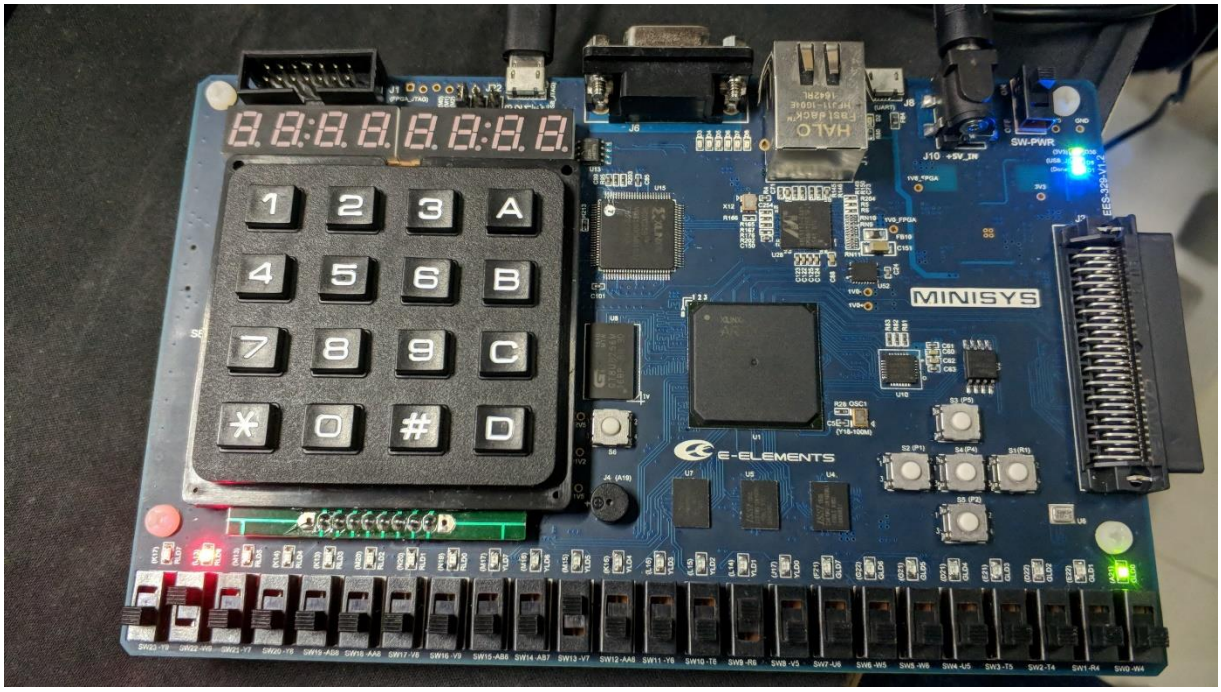
0(Y9)+0(W9)+0(Y7)+1(Y8) +0(AA8)→0(T4)+1(W4)



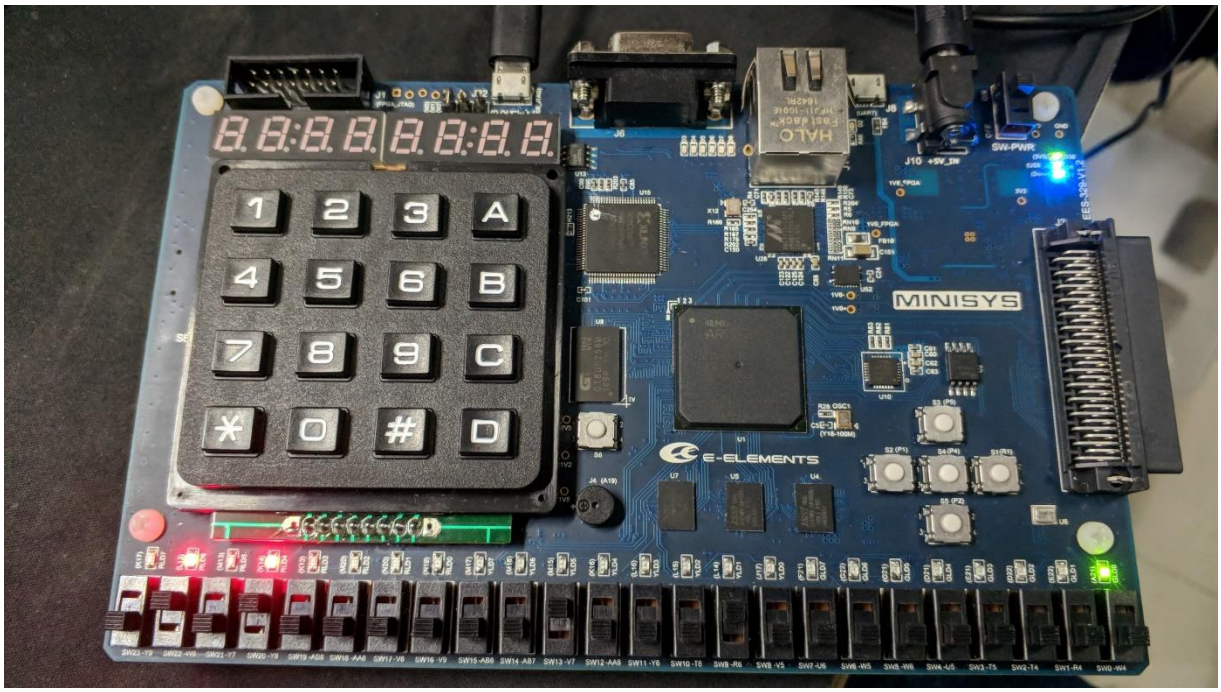
0(Y9)+0(W9)+1 (Y7)+0(Y8) +0(AA8) →0(T4)+1(W4)



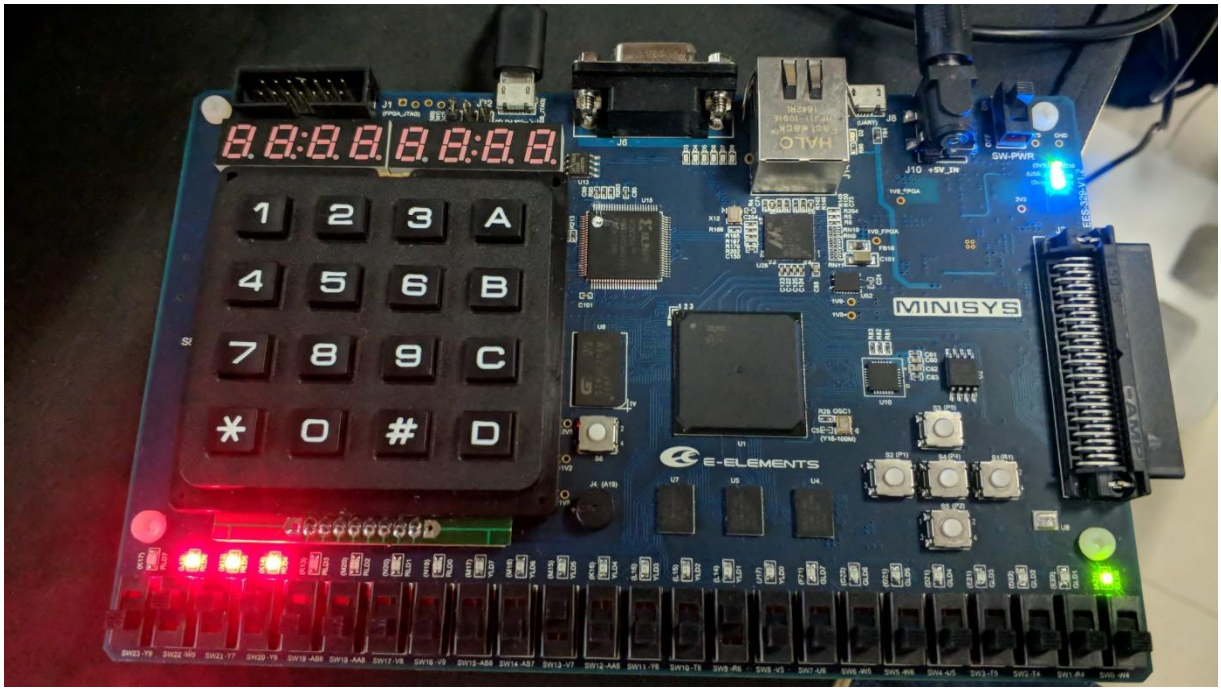
0(Y9)+1(W9)+0(Y7)+0(Y8) +0(AA8) →0(T4)+1(W4)



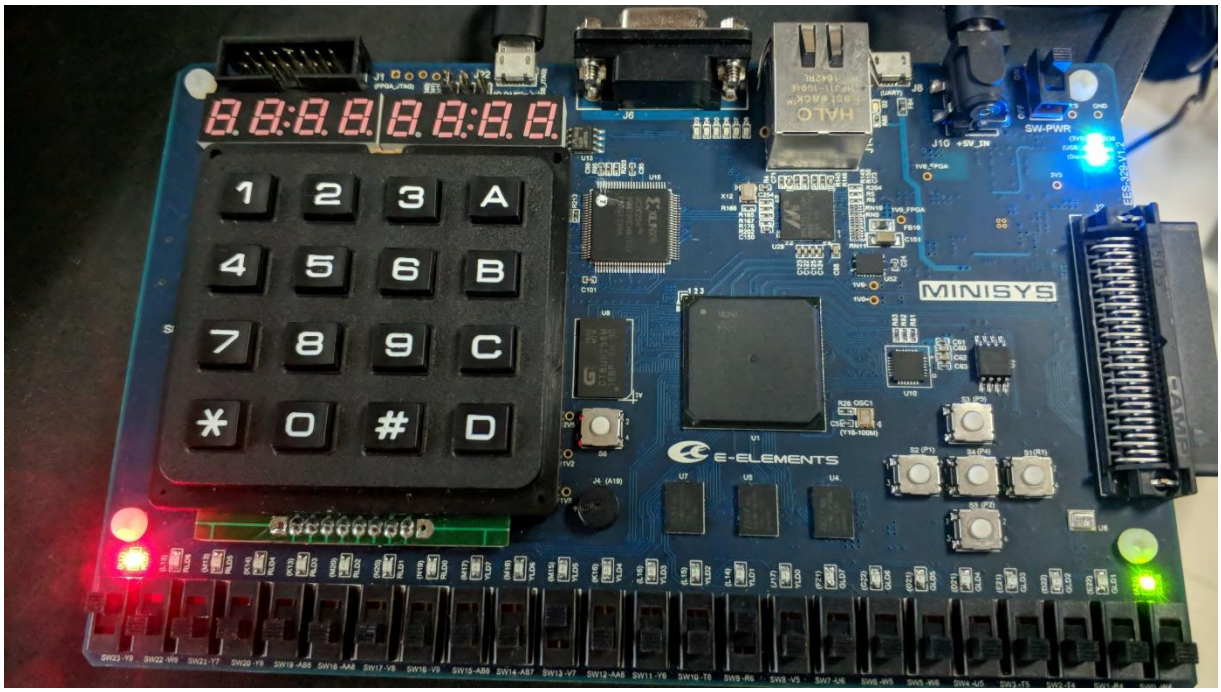
0(Y9)+1(W9)+0(Y7)+1(Y8) +0(AA8)→0(T4)+1(W4)



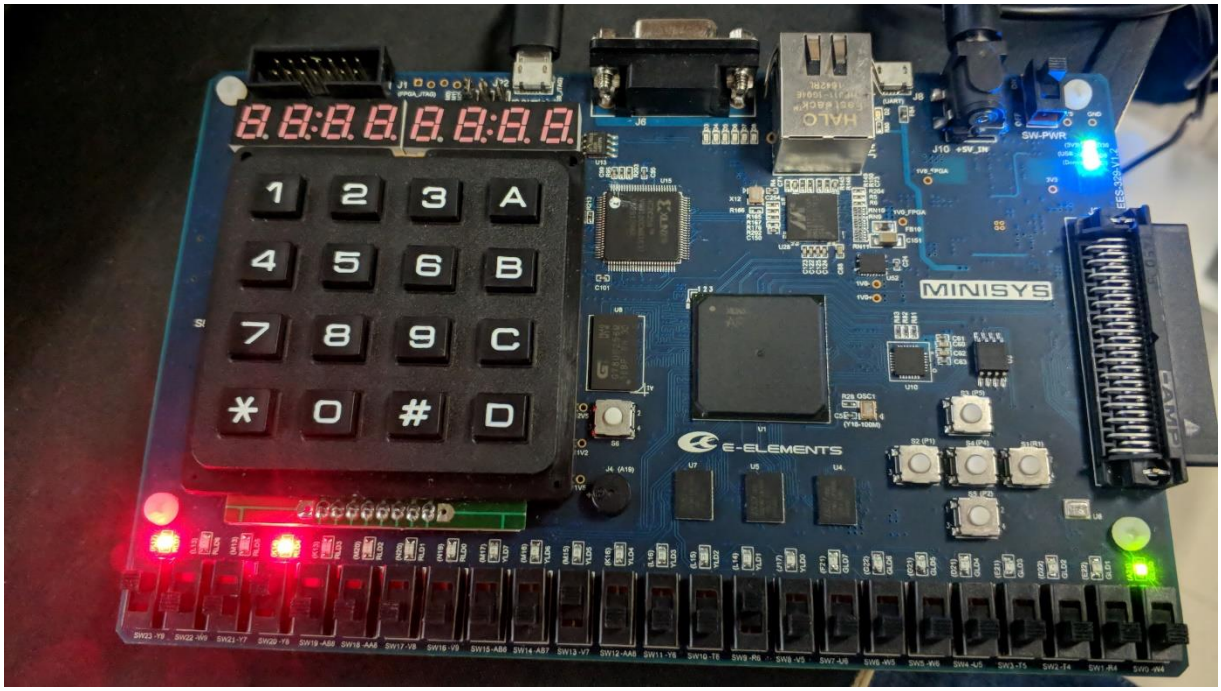
0(Y9)+1(W9)+1(Y7)+1(Y8) +0(AA8)→0(T4)+1(W4)



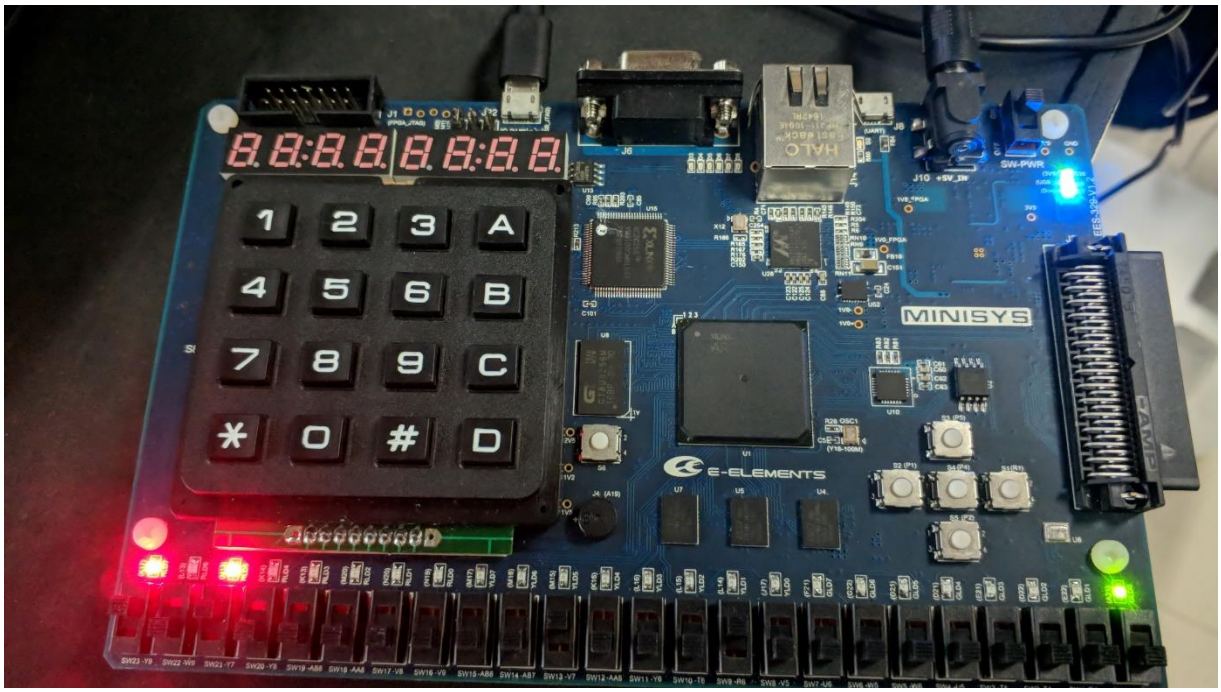
1(Y9)+0(W9)+0(Y7)+0(Y8) +0(AA8)→0(T4)+1(W4)



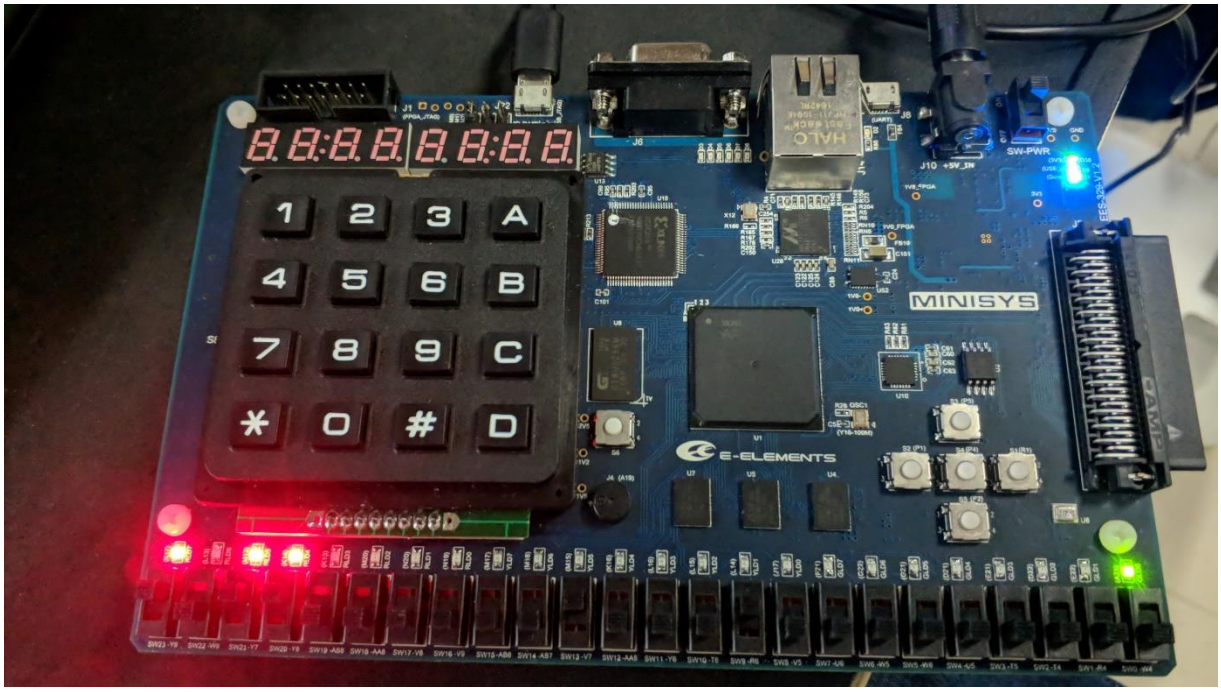
1(Y9)+0(W9)+0(Y7)+1(Y8) +0(AA8) →0(T4)+1(W4)



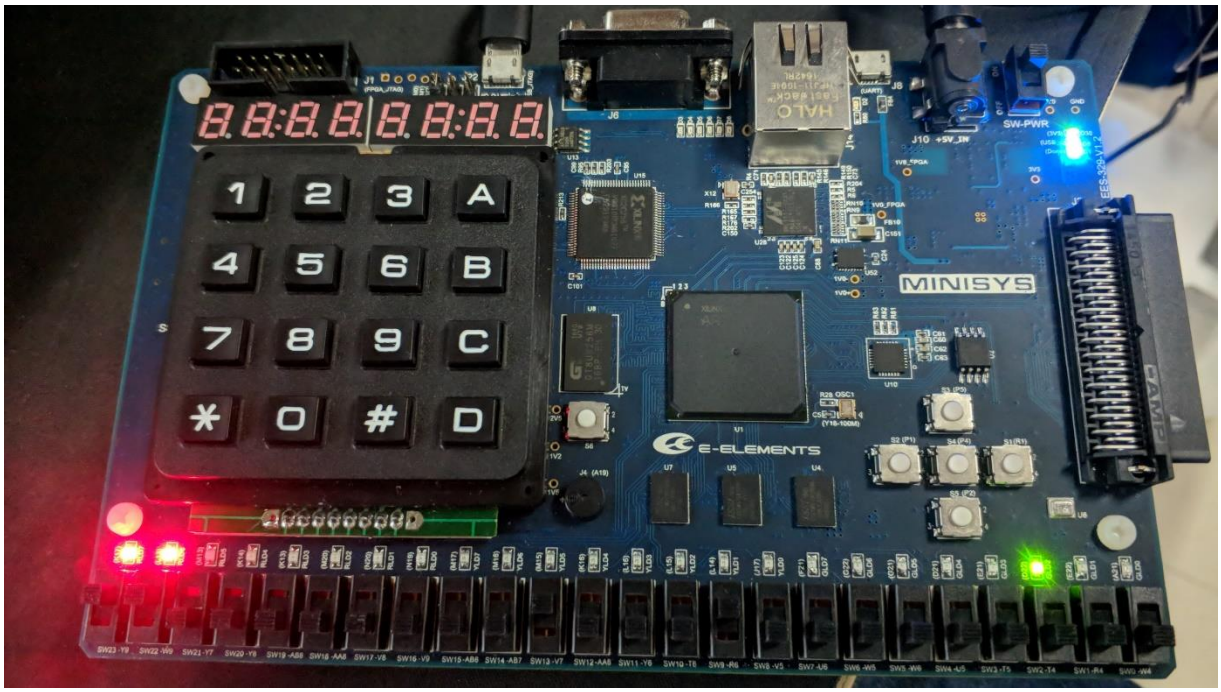
1(Y9)+0(W9)+1(Y7)+0(Y8) +0(AA8)→0(T4)+1(W4)



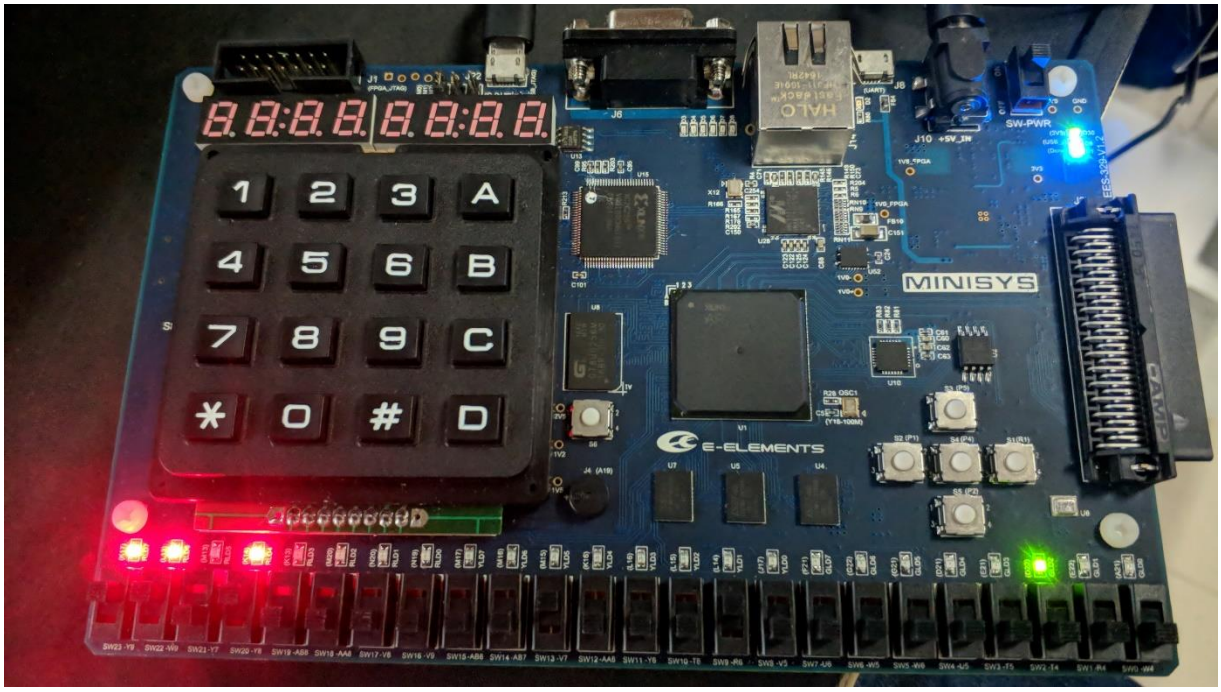
1(Y9)+0(W9)+1(Y7)+1(Y8) +0(AA8)→0(T4)+1(W4)



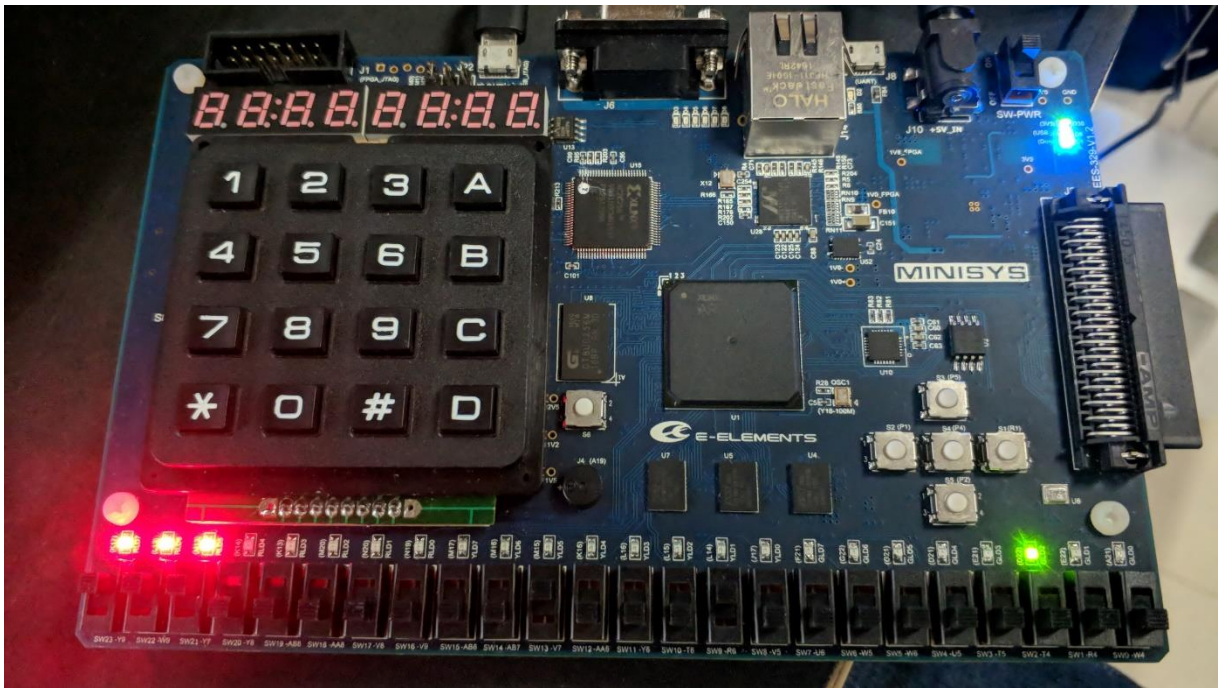
1(Y9)+1(W9)+0(Y7)+0(Y8) +0(AA8)→1(T4)+0(W4)



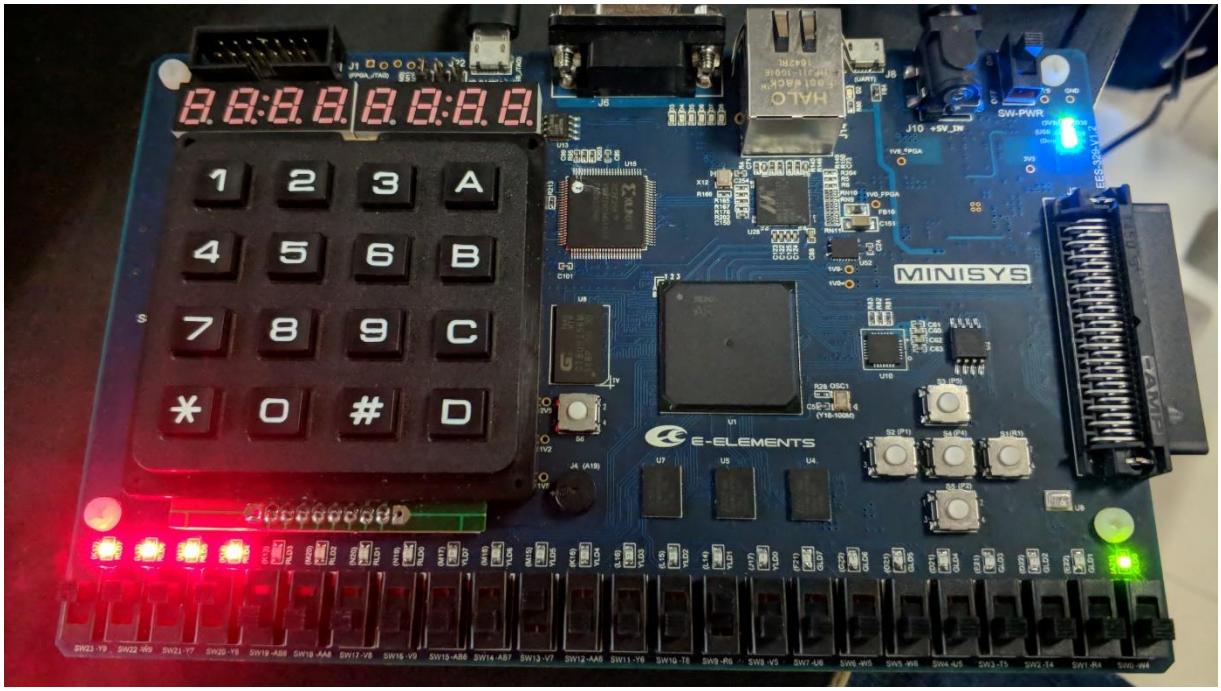
1Y9)+1(W9)+0(Y7)+1(Y8) +0(AA8)→1(T4)+0(W4)



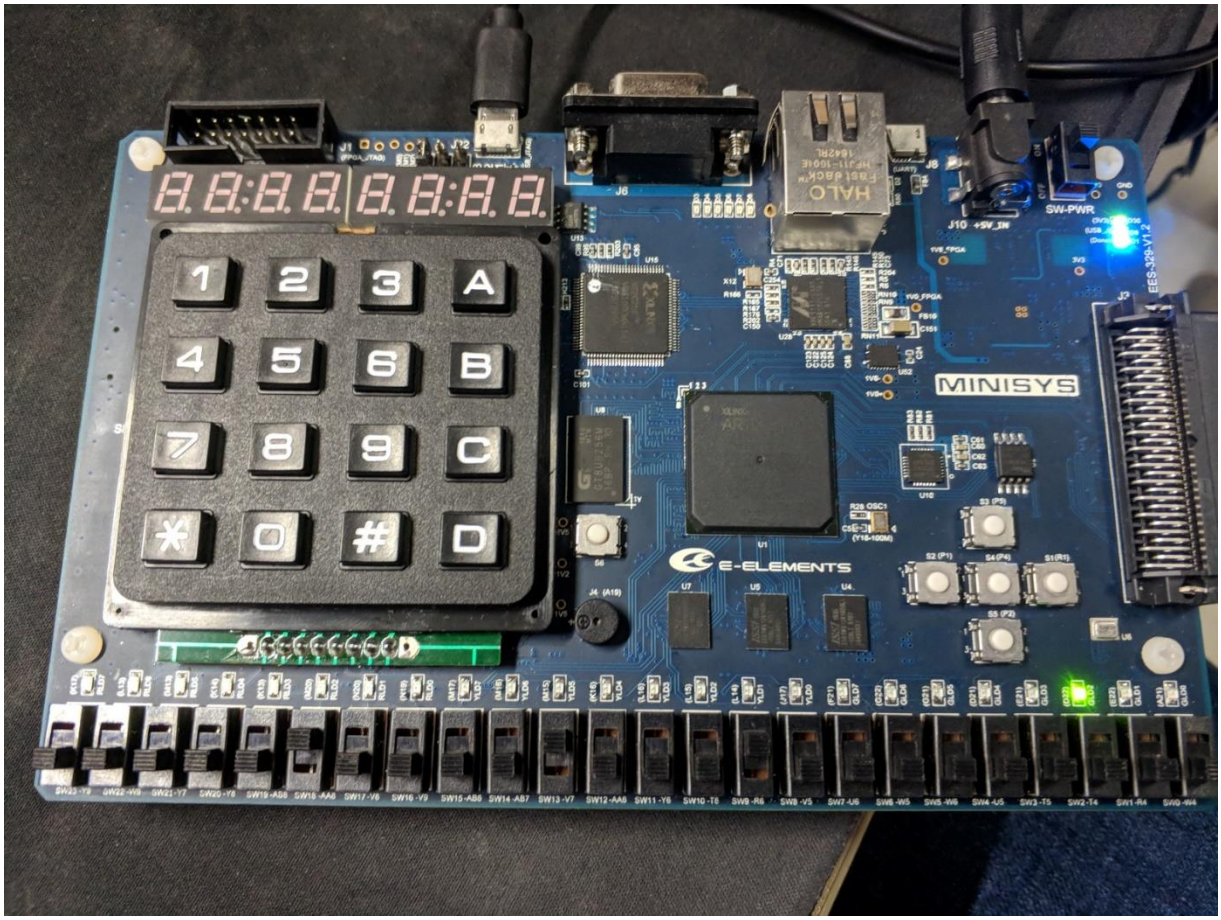
1(Y9)+1(W9)+1(Y7)+0(Y8) +0(AA8)→1(T4)+0(W4)



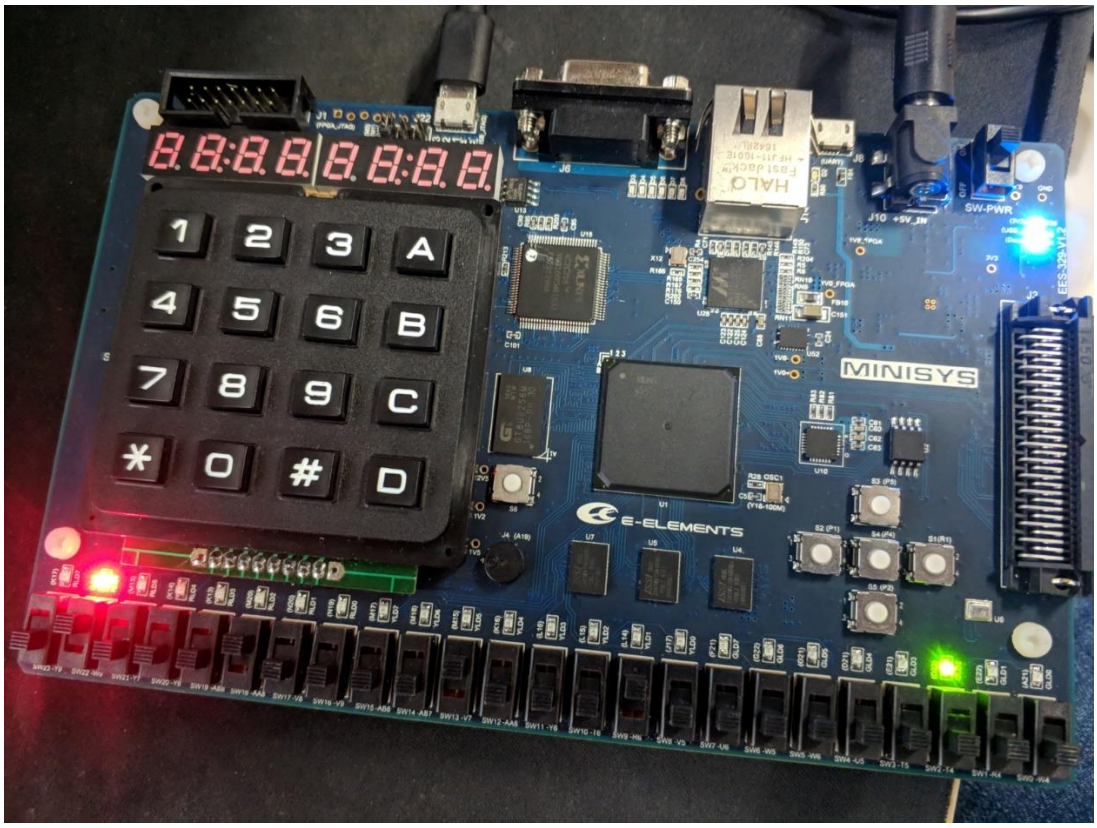
1(Y9)+1(W9)+1(Y7)+1(Y8) +0(AA8)→0(T4)+1(W4)



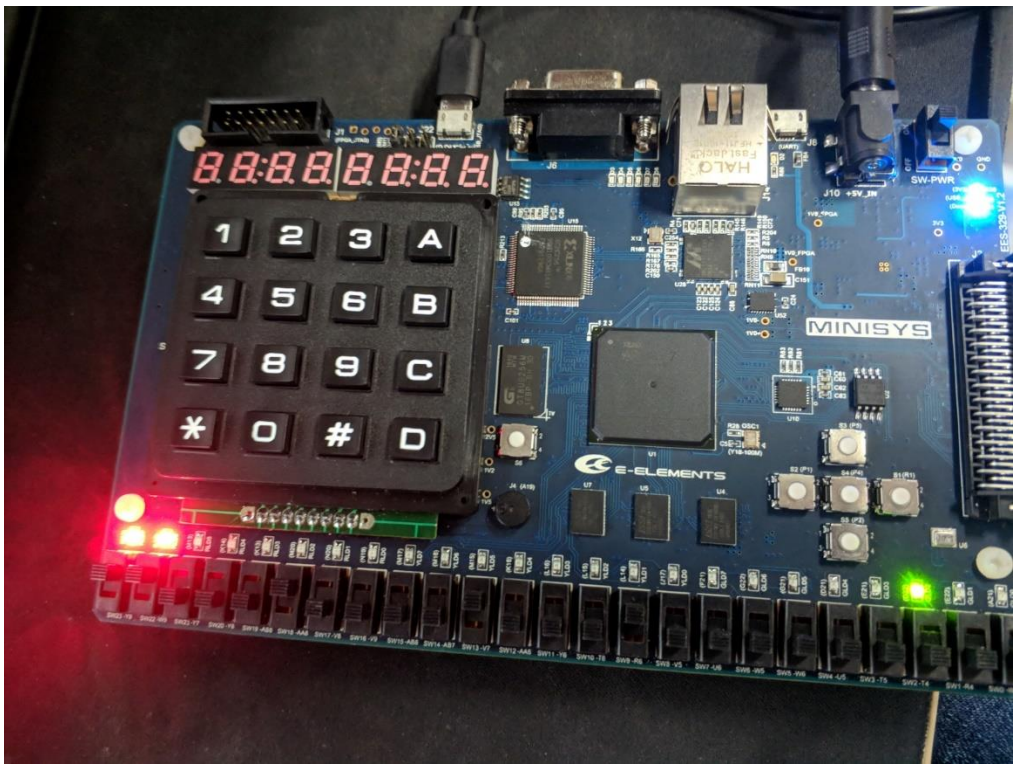
0(Y9)+0(W9)+0(Y7)+0(Y8) + 1(AA8) →1(T4)+0(W4)



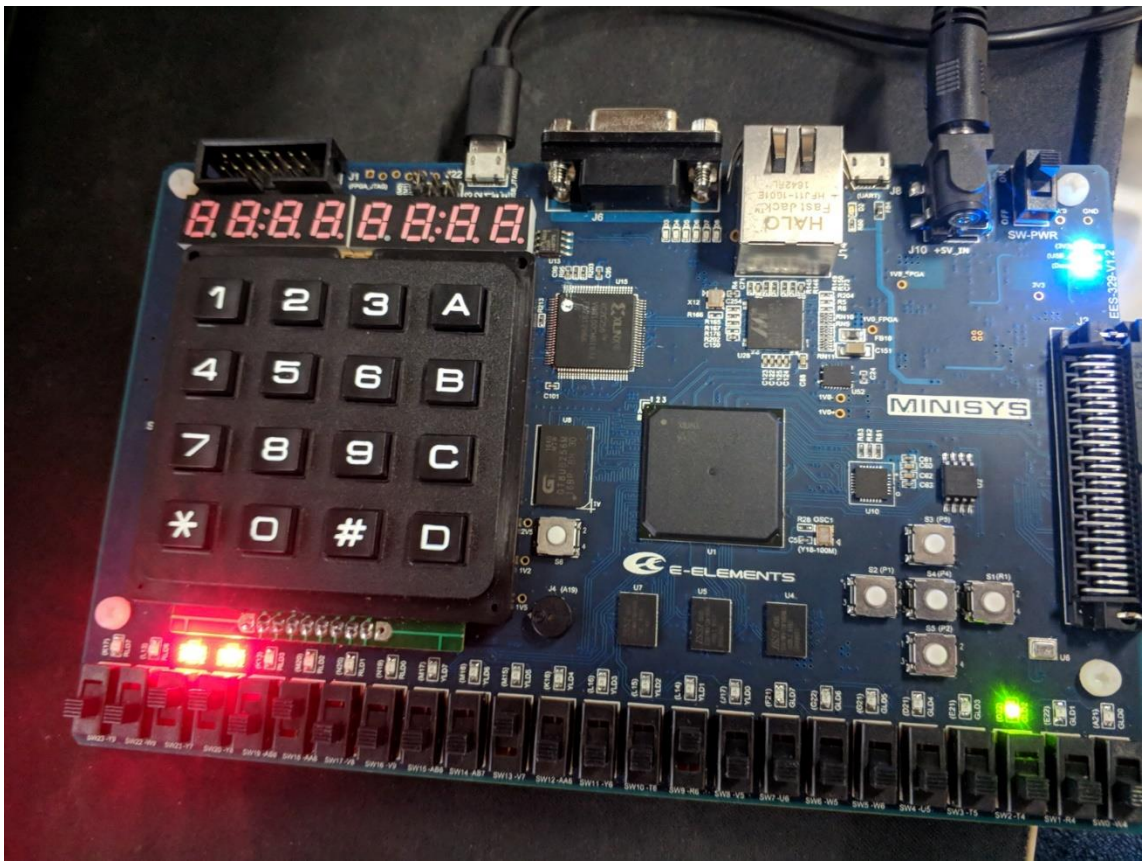
0(Y9)+1(W9)+0(Y7)+0(Y8) + 1(AA8) →1(T4)+0(W4)



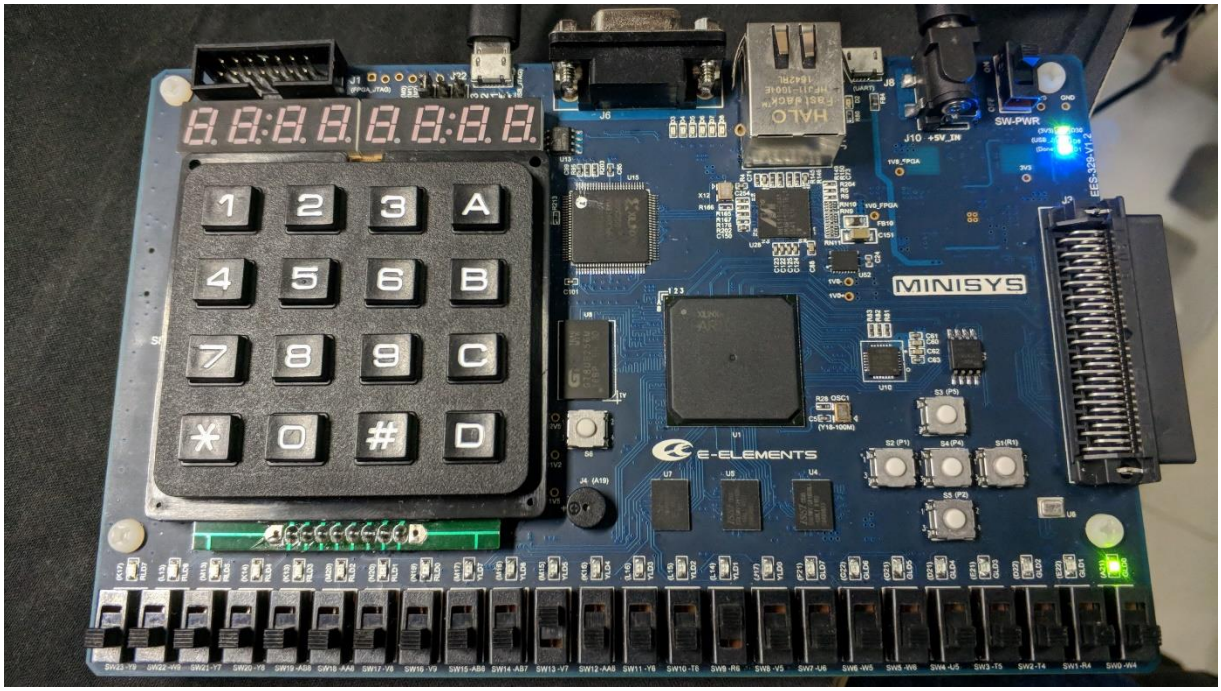
1(Y9)+1(W9)+0(Y7)+0(Y8) + 1(AA8) →1(T4)+0(W4)



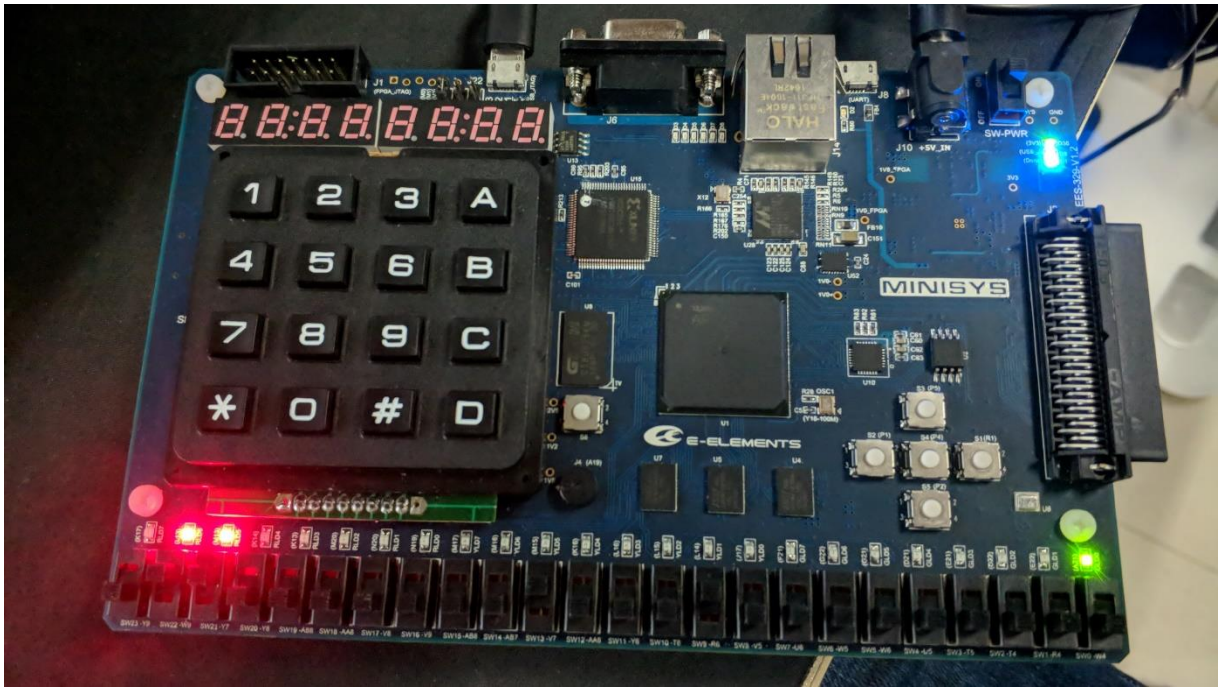
0(Y9)+0(W9)+1(Y7)+1(Y8) + 1(AA8) →1(T4)+0(W4)



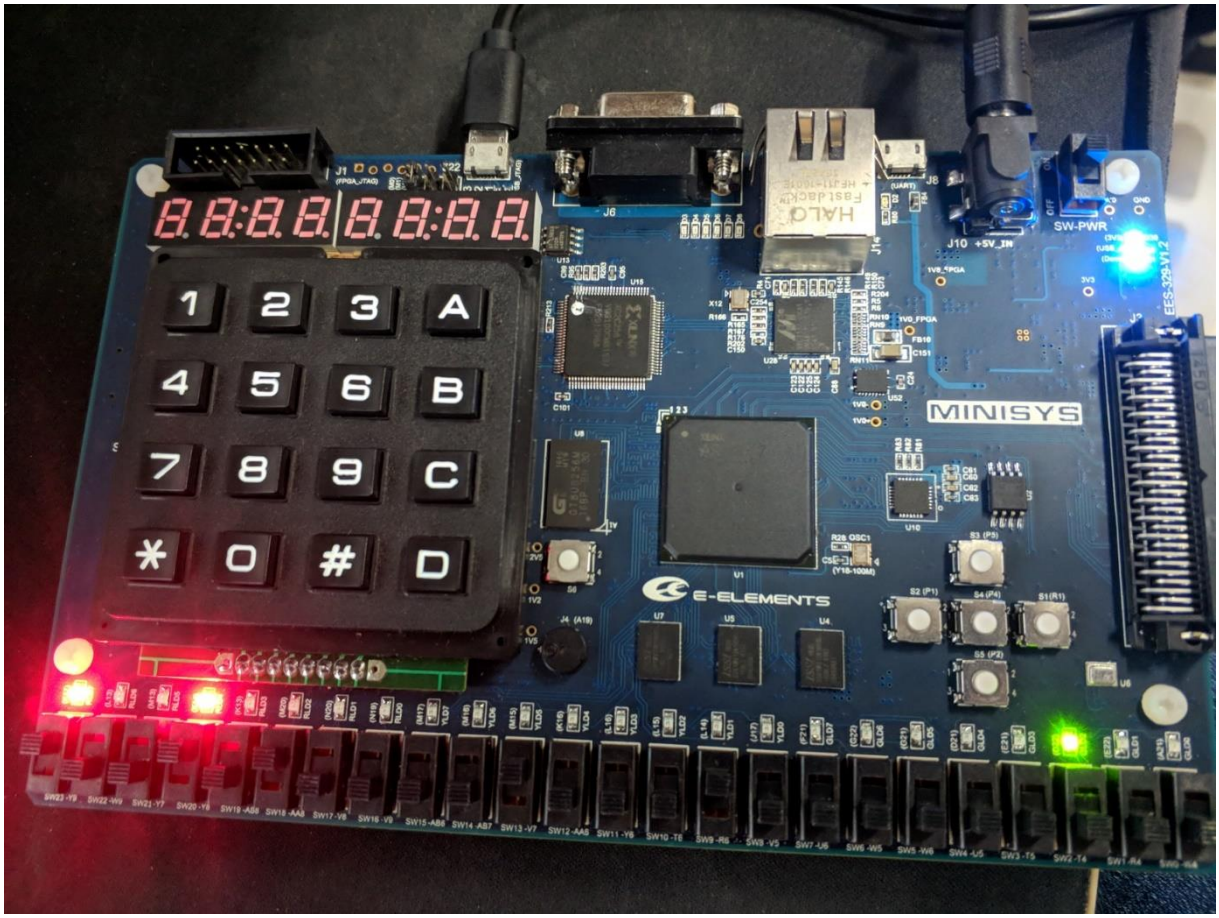
0(Y9)+0(W9)+0(Y7)+0(Y8) + 0(AA8) →0(T4)+1(W4)



0(Y9)+1(W9)+1(Y7)+0(Y8) + 0(AA8) →0(T4)+1(W4)



1(Y9)+0(W9)+0(Y7)+1(Y8) + 0(AA8) →1(T4)+0(W4)



THE DESCRIPTION OF OPERATION

I: build a new project and named it then add source or later choose the artix7fgg484100-1's type

II: then we got in the real surface, we can add a file in the design source and named it with five input and six output, then we use data-flow way in sum of product the last two using five input and six output. four output for show input and the other to show result. The common using the same input and output name for easy to simulation.

III: then we add a simulation file without any input or output but define reg and wire variable then bind the input with reg and output with wire.

IV do the simulation and do the synthesiswill be Smooth sailing, after simulation we will get the simulation graph like ladder. And the synthesis will bring a graph like real chips Sometimes it will happen cannot find ports on this module then we can close vivado and open it once again it will disappear.

V Do the implementation, after that we can define the i/o ports we set input with the switch's code and output with their code at the order. Do not forget the voltage to 3.3V

VI produce the bitstream file and connect the platform. If nothing had been found, connect the usb once again and click it to stop server. Then do auto connect.

VII Finally program the device in the green strip, we will have a programed platform, test it and take photos.

PROBLEMS AND SOLUTIONS

Task1:

1. Sometimes it will happen some strange things

   Solutions: close vivado and open it again or even reboot computer

2. After add others files in the folder, the udp file will go to non-files folder

   Solutions: disabled others file by the right click, then the simulation will be ok to run

3. In the file Ass_3_2_v if use 1 to be the .D*(1),it will have warning.

   Solution: use 1'b1/0 instead of 1/0

4. Sometimes it will appear file is updated and needed reload,

   Solution:then click the reload and just obey it

ADDITION

Describe of waves and codes in Simulation is provide

Description of I/O ports is provide in task two because only task two need it

Problems and solution is provided

Description of waves are provided