LAB02

1.

Design specification

output cout, s;

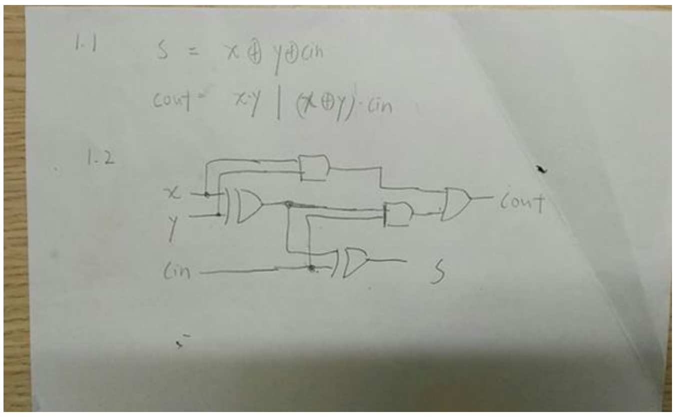
input x, y, cin;

cout = xy | xcin | ycin;

s = x^y^cin

Design implementation

diagram


1.1
$s = x \oplus y \oplus cin$
$cout = x \cdot y \mid (x \oplus y) \cdot cin$
1.2
x
y
cin
cout
s

assign cout = x & y | x & cin | y & cin;

assign s = x ^ y ^ cin;

I/O PINS

```
set_property PACKAGE_PIN U16 [get_ports cout]
set_property PACKAGE_PIN E19 [get_ports s]
set_property PACKAGE_PIN W16 [get_ports cin]
set_property PACKAGE_PIN V17 [get_ports x]
set_property PACKAGE_PIN V16 [get_ports y]
```

Discussion:

其實老師上完還對板子一無所知，我問同學才知道要用板子當 testbench，那個開關是控制 input 的，還有 input 的腳位在開關下方

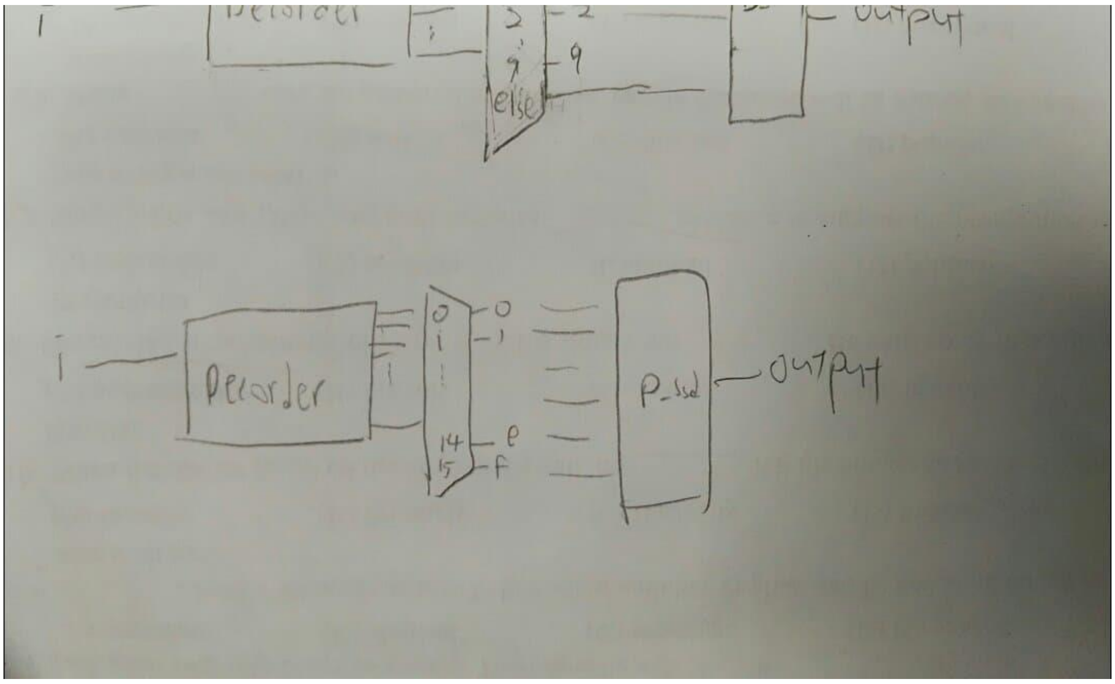
Conclusion

第一次完板子，覺得很酷

2.
Design specification:

```
input [3:0]i;
output [7:0]D_ssd;              // seven segment display
output [3:0]light;              // light of seven segment display
output [3:0]d;                  // led
light = 4'b0000;
d = i;
```

Diagram



Design implementation

```
display U0(i, D_ssd);

    assign light = 4'b0000;
    assign d = i;

`define SS_0 8'b00000011
`define SS_1 8'b10011111
`define SS_2 8'b00100101
`define SS_3 8'b00001101
`define SS_4 8'b10011001
`define SS_5 8'b01001001
```

```
`define SS_6 8'b01000001
`define SS_7 8'b00011111
`define SS_8 8'b00000001
`define SS_9 8'b00001001

module display(in, segs);
        input [3:0]in;
        output reg [7:0]segs;

        always @*
                case (in)
                4'd0: segs = `SS_0;
                4'd1: segs = `SS_1;
                4'd2: segs = `SS_2;
                4'd3: segs = `SS_3;
                4'd4: segs = `SS_4;
                4'd5: segs = `SS_5;
                4'd6: segs = `SS_6;
                4'd7: segs = `SS_7;
                4'd8: segs = `SS_8;
                4'd9: segs = `SS_9;
                default: segs = 8'b01110001;
          endcase
endmodule
```

I/O PINS

```
set_property PACKAGE_PIN V19 [get_ports {d[3]}]
set_property PACKAGE_PIN U19 [get_ports {d[2]}]
set_property PACKAGE_PIN E19 [get_ports {d[1]}]
set_property PACKAGE_PIN U16 [get_ports {d[0]}]
set_property PACKAGE_PIN W7 [get_ports {D_ssd[7]}]
set_property PACKAGE_PIN W6 [get_ports {D_ssd[6]}]
set_property PACKAGE_PIN U8 [get_ports {D_ssd[5]}]
set_property PACKAGE_PIN V8 [get_ports {D_ssd[4]}]
set_property PACKAGE_PIN U5 [get_ports {D_ssd[3]}]
set_property PACKAGE_PIN V5 [get_ports {D_ssd[2]}]
set_property PACKAGE_PIN U7 [get_ports {D_ssd[1]}]
set_property PACKAGE_PIN V7 [get_ports {D_ssd[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {i[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {i[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {i[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {i[0]}]
set_property PACKAGE_PIN W4 [get_ports {light[3]}]
set_property PACKAGE_PIN V4 [get_ports {light[2]}]
set_property PACKAGE_PIN U4 [get_ports {light[1]}]
set_property PACKAGE_PIN U2 [get_ports {light[0]}]
```

Disscussion:

　這次沒遇到沒什麼問題，唯一的就是引用另一個 module 時，我沒有給他一個名子，我原本寫 display (i, D_ssd); 然後有錯，才改成 display U0(i, D_ssd);

Conclusion:

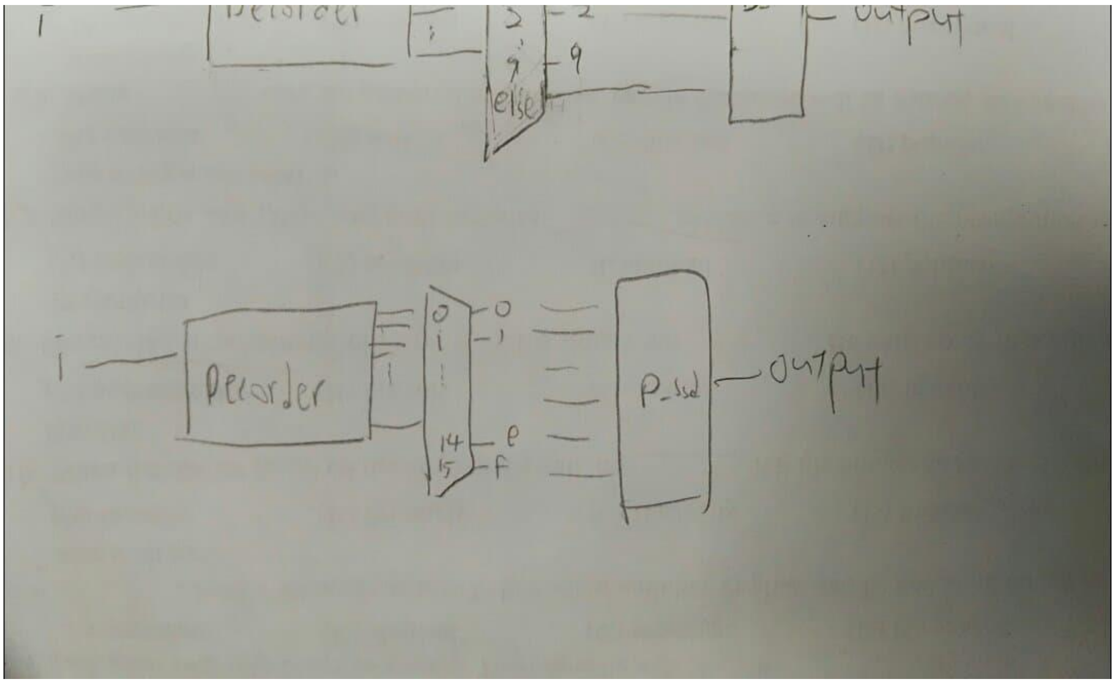
　這次感覺很新奇，居然可以讓燈亮，希望以後能讓 4 個燈亮不同的字。

3.

Design specification

```
input [3:0]i;
output [7:0]D;
output [3:0]d;
output [3:0]light;

    light = 4'b0000;
    d = i;
```



Design implementation

```
display U0(i, D);
assign light = 4'b0000;
assign d = i;
```

```
`define SS_0 8'b00000011
`define SS_1 8'b10011111
`define SS_2 8'b00100101
`define SS_3 8'b00001101
`define SS_4 8'b10011001
`define SS_5 8'b01001001
```

```
`define SS_6 8'b01000001
`define SS_7 8'b00011111
`define SS_8 8'b00000001
`define SS_9 8'b00001001
`define SS_10 8'b00010001
`define SS_11 8'b11000001
`define SS_12 8'b11100101
`define SS_13 8'b10000101
`define SS_14 8'b01100001
`define SS_15 8'b01110001

module display(in, DEC);
        input [3:0]in;
        output reg [7:0]DEC;
        always @*
        case (in)
          4'd0: DEC = `SS_0;                    // decorder
          4'd1: DEC = `SS_1;
          4'd2: DEC = `SS_2;
          4'd3: DEC = `SS_3;
          4'd4: DEC = `SS_4;
          4'd5: DEC = `SS_5;
          4'd6: DEC = `SS_6;
          4'd7: DEC = `SS_7;
          4'd8: DEC = `SS_8;
          4'd9: DEC = `SS_9;
          4'd10: DEC = `SS_10;
          4'd11: DEC = `SS_11;
          4'd12: DEC = `SS_12;
          4'd13: DEC = `SS_13;
          4'd14: DEC = `SS_14;
          4'd15: DEC = `SS_15;
          default: DEC = 8'b11111111;
        endcase
endmodule
```

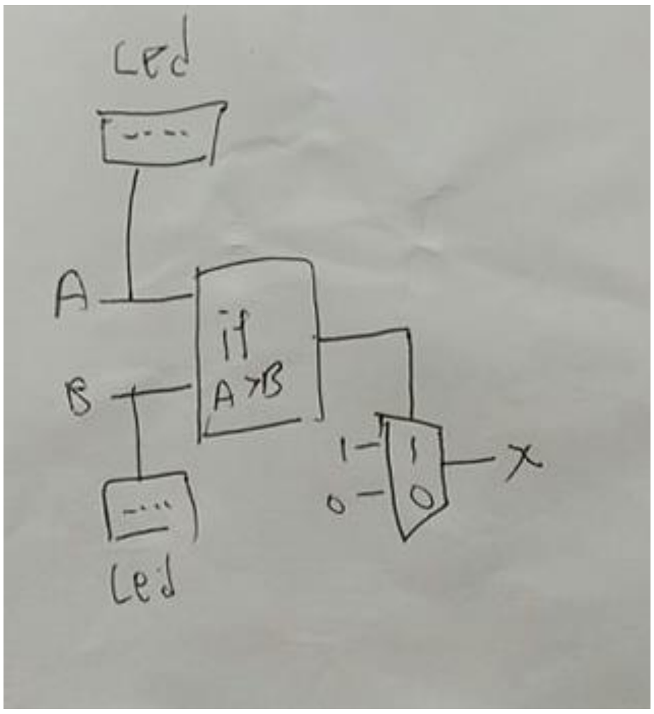
Disscussion:

這題跟第二題差不多，在多加幾個 output 型式就好

4.

Design specification

```
input [3:0]A, B;
output reg X;
output [3:0]a;
output [3:0]b;
a = A;
b = B;
if (A > B) x = 1;
else x = 0;
```



Design implementation

```
always @* begin
        if (A > B) X = 1;
        else X = 0;
    end
assign a = A;
assign b = B;
```

discussion

這次的想法引簡單，如果 a > b 就 output 1，否則 output 0;

Conclusion

　這次本來以為會很難，因為以前畫比較器很複雜，但他有大於這個功能真的很方便，減少了一大堆線路。