第三次實驗報告

題目:3-1~3-4

姓名:羅名志

學號:0813228

繳交日期:2022/3/16

一、3-1

(1)實驗程式碼:

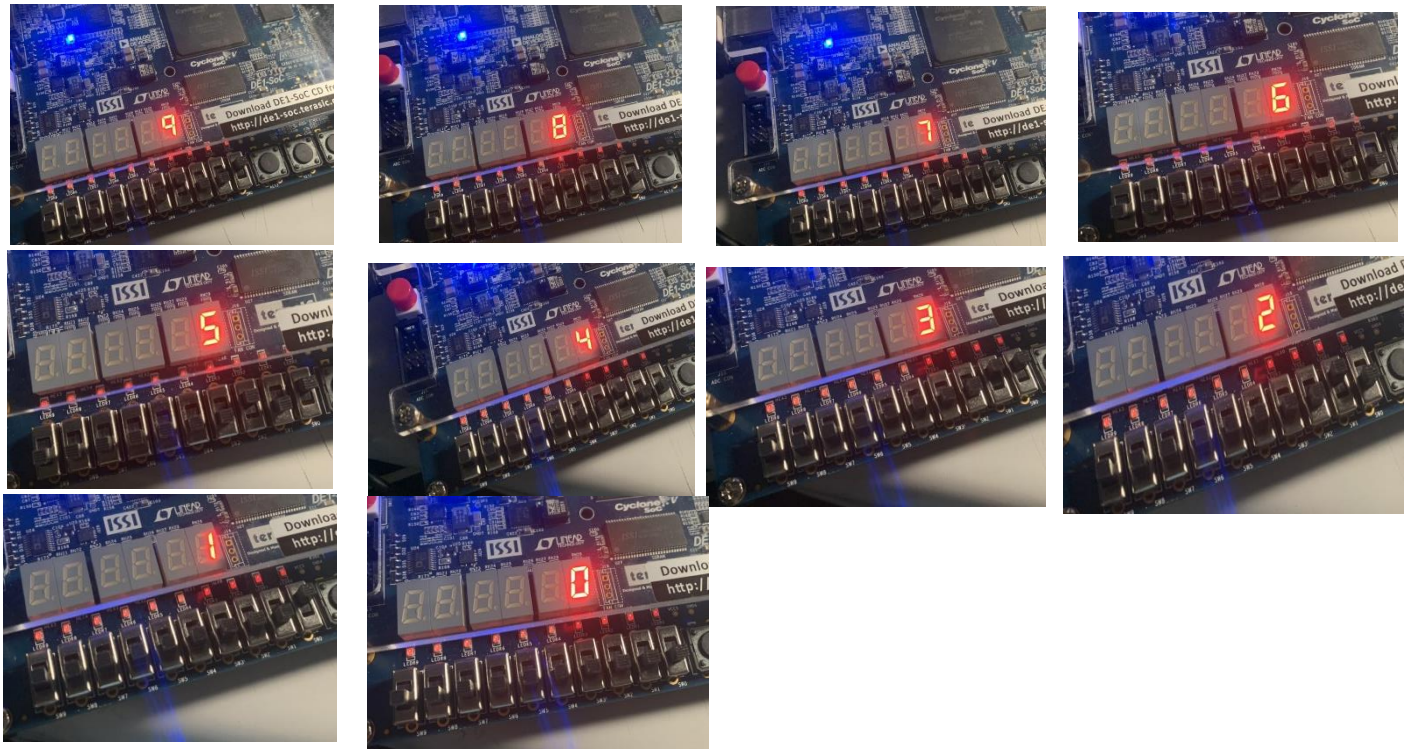
```
module labthree1(SW,HEX0);
input [3:0]SW;
output [6:0]HEX0;

assign HEX0[0]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)|(~SW[3]&~SW[2]&~SW[1]&SW[0])|(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)|(~SW[3]&~SW[2]&SW[1]&SW[0]&0)|
assign HEX0[1]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)|(~SW[3]&~SW[2]&~SW[1]&SW[0]&0)|(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)|(~SW[3]&~SW[2]&SW[1]&SW[0]&0
assign HEX0[2]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)|(~SW[3]&~SW[2]&~SW[1]&SW[0]&0)|(~SW[3]&~SW[2]&SW[1]&~SW[0])|(~SW[3]&~SW[2]&SW[1]&SW[0]&0)|
assign HEX0[3]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)|(~SW[3]&~SW[2]&~SW[1]&SW[0])|(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)|(~SW[3]&~SW[2]&SW[1]&SW[0]&0)|
assign HEX0[4]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)|(~SW[3]&~SW[2]&~SW[1]&SW[0])|(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)|(~SW[3]&~SW[2]&SW[1]&SW[0])|(~
assign HEX0[5]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)|(~SW[3]&~SW[2]&~SW[1]&SW[0])|(~SW[3]&~SW[2]&SW[1]&~SW[0])|(~SW[3]&~SW[2]&SW[1]&SW[0])|(~SW
assign HEX0[6]=(~SW[3]&~SW[2]&~SW[1]&~SW[0])|(~SW[3]&~SW[2]&~SW[1]&SW[0])|(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)|(~SW[3]&~SW[2]&SW[1]&SW[0]&0)|(~

endmodule
```

```
)|(~SW[3]&SW[2]&~SW[1]&~SW[0])|(~SW[3]&SW[2]&~SW[1]&SW[0]&0)|(~SW[3]&SW[2]&SW[1]&~SW[0]&0)|(~SW[3]&SW[2]&SW[1]&SW[0]&0)|
&0)|(~SW[3]&SW[2]&~SW[1]&~SW[0]&0)|(~SW[3]&SW[2]&~SW[1]&SW[0])|(~SW[3]&SW[2]&SW[1]&~SW[0])|(~SW[3]&SW[2]&SW[1]&SW[0]&0)|
)|(~SW[3]&SW[2]&~SW[1]&~SW[0]&0)|(~SW[3]&SW[2]&~SW[1]&SW[0]&0)|(~SW[3]&SW[2]&SW[1]&~SW[0]&0)|(~SW[3]&SW[2]&SW[1]&SW[0]&0
)|(~SW[3]&SW[2]&~SW[1]&~SW[0])|(~SW[3]&SW[2]&~SW[1]&SW[0]&0)|(~SW[3]&SW[2]&SW[1]&~SW[0]&0)|(~SW[3]&SW[2]&SW[1]&SW[0])|(S
(~SW[3]&SW[2]&~SW[1]&~SW[0])|(~SW[3]&SW[2]&~SW[1]&SW[0])|(~SW[3]&SW[2]&SW[1]&~SW[0]&0)|(~SW[3]&SW[2]&SW[1]&SW[0])|(SW[3]
SW[3]&SW[2]&~SW[1]&~SW[0]&0)|(~SW[3]&SW[2]&~SW[1]&SW[0]&0)|(~SW[3]&SW[2]&SW[1]&~SW[0]&0)|(~SW[3]&SW[2]&SW[1]&SW[0])|(SW[
(~SW[3]&SW[2]&~SW[1]&~SW[0]&0)|(~SW[3]&SW[2]&~SW[1]&SW[0]&0)|(~SW[3]&SW[2]&SW[1]&~SW[0]&0)|(~SW[3]&SW[2]&SW[1]&SW[0])|(S
```

```
&SW[0]&0)|(SW[3]&~SW[2]&~SW[1]&~SW[0]&0)|(SW[3]&~SW[2]&~SW[1]&SW[0]&0)|(SW[3]&(SW[2]|SW[1]));
&SW[0]&0)|(SW[3]&~SW[2]&~SW[1]&~SW[0]&0)|(SW[3]&~SW[2]&~SW[1]&SW[0]&0)|(SW[3]&(SW[2]|SW[1]));
1]&SW[0]&0)|(SW[3]&~SW[2]&~SW[1]&~SW[0]&0)|(SW[3]&~SW[2]&~SW[1]&SW[0]&0)|(SW[3]&(SW[2]|SW[1]));
&SW[0])|(SW[3]&~SW[2]&~SW[1]&~SW[0]&0)|(SW[3]&~SW[2]&~SW[1]&SW[0])|(SW[3]&(SW[2]|SW[1]));
0])|(SW[3]&~SW[2]&~SW[1]&~SW[0]&0)|(SW[3]&~SW[2]&~SW[1]&SW[0])|(SW[3]&(SW[2]|SW[1]));
W[0])|(SW[3]&~SW[2]&~SW[1]&~SW[0]&0)|(SW[3]&~SW[2]&~SW[1]&SW[0]&0)|(SW[3]&(SW[2]|SW[1]));
&SW[0])|(SW[3]&~SW[2]&~SW[1]&~SW[0]&0)|(SW[3]&~SW[2]&~SW[1]&SW[0]&0)|(SW[3]&(SW[2]|SW[1]));
```

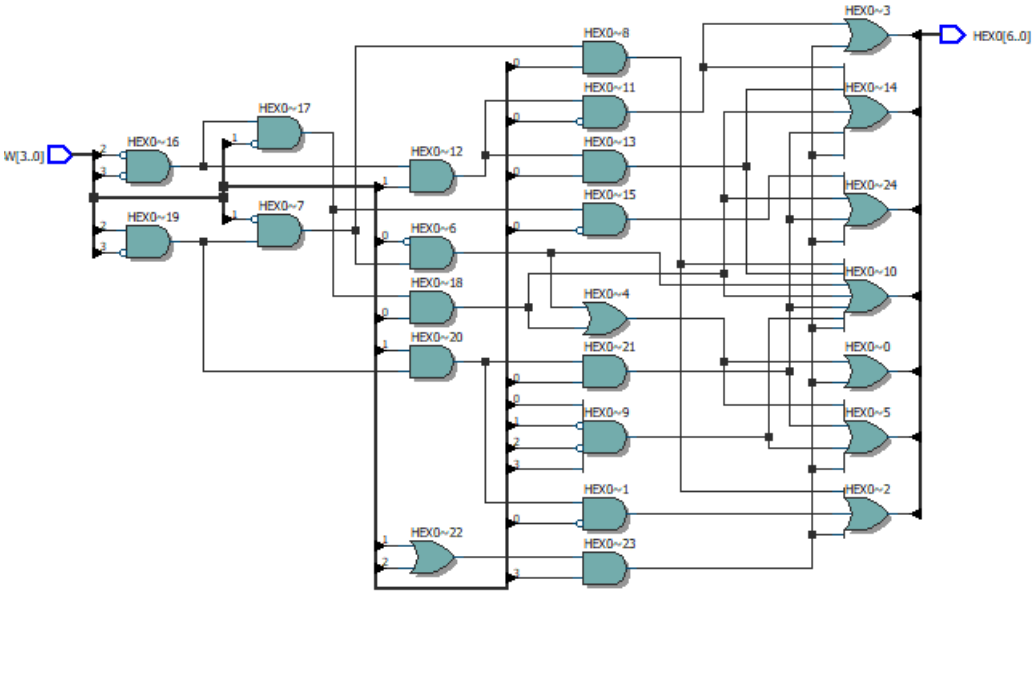
//由左至右分成 3 張圖，將每個組合對應到 hex0

(2)實驗結果:



//上排(由左至右，SW[3:0]):1001/1000/0111/0110
//中排(由左至右，SW[3:0]):0101/0100/0011/0010
//下排(由左至右，SW[3:0]):0001/0000

(3)RTL 布局:



(4)問題與討論:
此實驗較簡單，只要把不同組合分別對應到 HEX0 上就好，剛好可以複習上次的實驗內容。

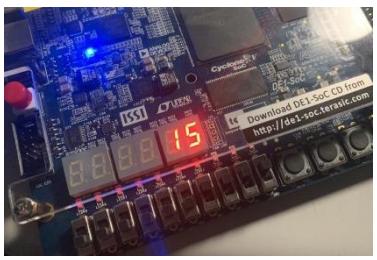
二、3-2

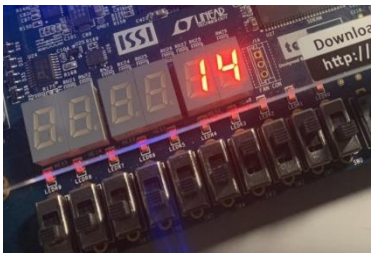
(1)實驗程式碼:

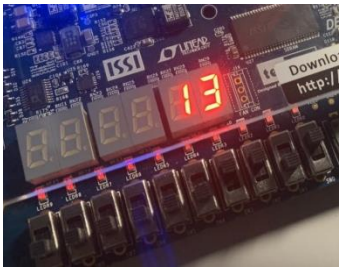
```
module labthree2(SW,HEX1,HEX0);
input [3:0]SW;
output [6:0]HEX1,HEX0;
wire sign ;

wire [3:0]y, v;



comparator comp(SW[3],SW[2],SW[1],SW[0],sign
circuitA a(SW[3:0],sign,v[3:0]);

circuitB b(sign,HEX1);
char_7seg c(v,HEX0 );

endmodule

module comparator(a,b,c,d,sign);
input a,b,c,d;
output sign ;
wire  s5,s4,s3,s2,s1,s0;
assign sign = s5|s4|s3|s2|s1|s0;
assign s5 = a&~b&c&~d;  //10
assign s4 = a&~b&c&d;    //11
assign s3 = a&b&~c&~d;  //12
assign s2 = a&b&~c&d;  //13
assign s1 = a&b&c&~d;  //14
assign s0 = a&b&c&d;    //15
endmodule

module circuitA(SW,sign,v);
input [3:0]SW;
input sign;
wire [2:0]x;
output [3:0]v;

assign x[0]=(~SW[2]&SW[1]&~SW[0]&0)||(~SW[2]&SW[1]&SW[0]&1)||(SW[2]&~SW[1]&~SW[0]&0)||(SW[2]&~SW[1]&SW[0]&1)||(SW[2]&SW[1]&~SW[0]&0)||(SW[2]&SW[1]&SW[0]&1);
assign x[1]=(~SW[2]&SW[1]&~SW[0]&0)||(~SW[2]&SW[1]&SW[0]&0)||(SW[2]&~SW[1]&~SW[0]&1)||(SW[2]&~SW[1]&SW[0]&1)||(SW[2]&SW[1]&~SW[0]&0)||(SW[2]&SW[1]&SW[0]&0);
assign x[2]=(~SW[2]&SW[1]&~SW[0]&0)||(~SW[2]&SW[1]&SW[0]&0)||(SW[2]&~SW[1]&~SW[0]&0)||(SW[2]&~SW[1]&SW[0]&0)||(SW[2]&SW[1]&~SW[0]&1)||(SW[2]&SW[1]&SW[0]&1);

assign v[0]=(~sign&SW[0])||(sign&x[0]);
assign v[1]=(~sign&SW[1])||(sign&x[1]);
assign v[2]=(~sign&SW[2])||(sign&x[2]);
assign v[3]=(~sign&SW[3])||(sign&0);


endmodule

module circuitB(a,hex);
input a;
output [6:0]hex;


assign hex[0]=(a&1);
assign hex[1]=(a&0);
assign hex[2]=(a&0);
assign hex[3]=(a&1);
assign hex[4]=(a&1);
assign hex[5]=(a&1);
assign hex[6]=(a&1)||(~a&1);
endmodule


module char_7seg(SW,HEX);
input [3:0]SW;
output [6:0]HEX;

assign HEX[0]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&~SW[1]&SW[0])||(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&SW[0]&0)||(~SW[3]&SW[2]&~SW[1]&~SW[0])
assign HEX[1]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&~SW[1]&SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&SW[0]&0)||(~SW[3]&SW[2]&~SW[1]&~SW[0
assign HEX[2]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&~SW[1]&SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&~SW[0])||(~SW[3]&~SW[2]&SW[1]&SW[0]&0)||(~SW[3]&SW[2]&~SW[1]&~SW[0]&(
assign HEX[3]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&~SW[1]&SW[0])||(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&SW[0]&0)||(~SW[3]&SW[2]&~SW[1]&~SW[0])
assign HEX[4]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&~SW[1]&SW[0])||(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&SW[0])||(~SW[3]&SW[2]&~SW[1]&~SW[0])||(
assign HEX[5]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&~SW[1]&SW[0])||(~SW[3]&~SW[2]&SW[1]&~SW[0])||(~SW[3]&~SW[2]&SW[1]&SW[0])||(~SW[3]&SW[2]&~SW[1]&~SW[0]&0)||(
assign HEX[6]=(~SW[3]&~SW[2]&~SW[1]&~SW[0])||(~SW[3]&~SW[2]&~SW[1]&SW[0])||(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&SW[0]&0)||(~SW[3]&SW[2]&~SW[1]&~SW[0]&0)
endmodule
```

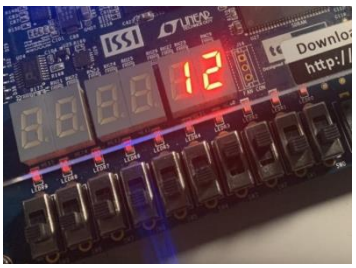
//68-75 行同 LAB 3-1，故部分省略

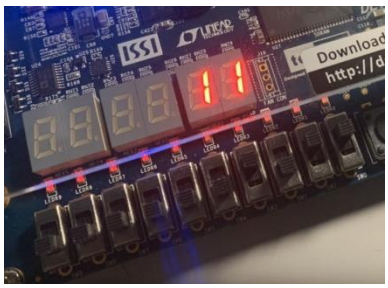
(2)實驗結果:

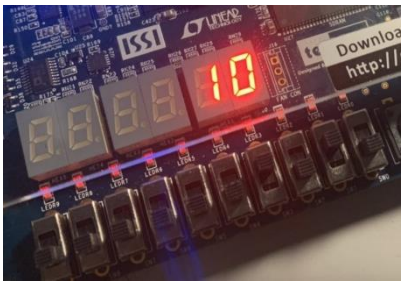


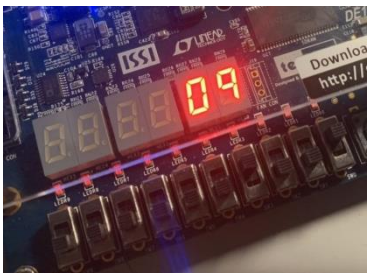


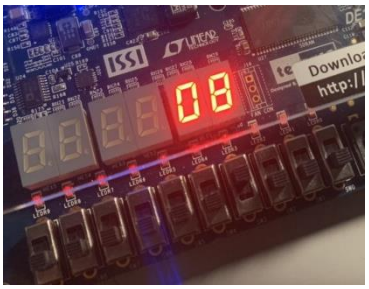


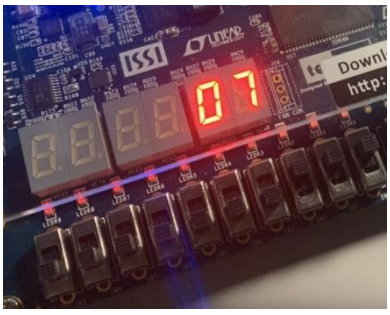


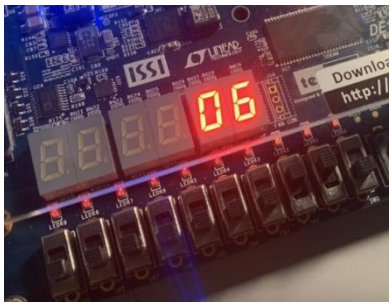


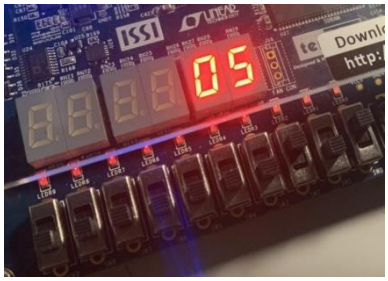


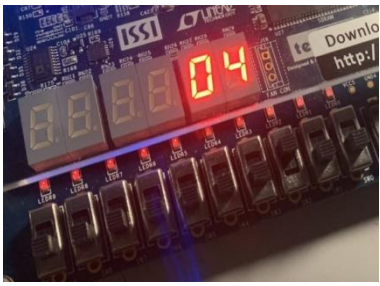


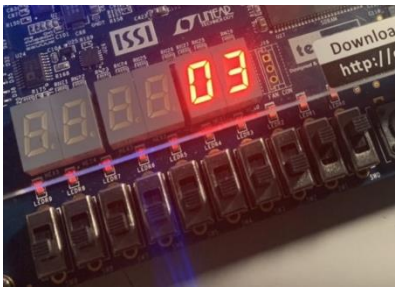


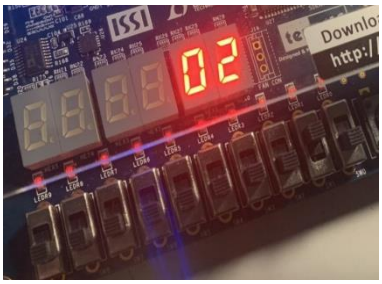


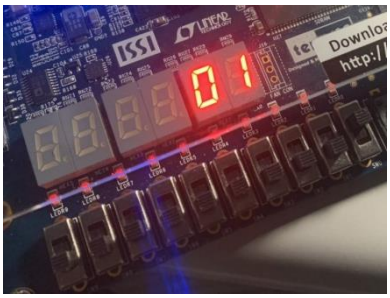


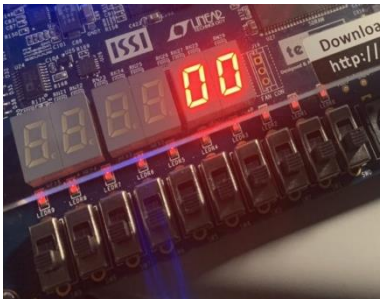






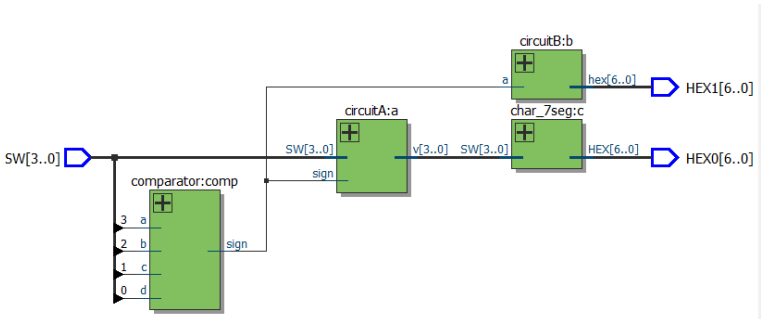






//1 排(由左至右，SW[3:0]):1111/1110/1101/1100
//2 排(由左至右，SW[3:0]):1011/1010/1001/1000
//3 排(由左至右，SW[3:0]):0111/0110/0101/0100
//4 排(由左至右，SW[3:0]):0011/0010/0001/0000

(3)RTL 布局:



(4)問題與討論:

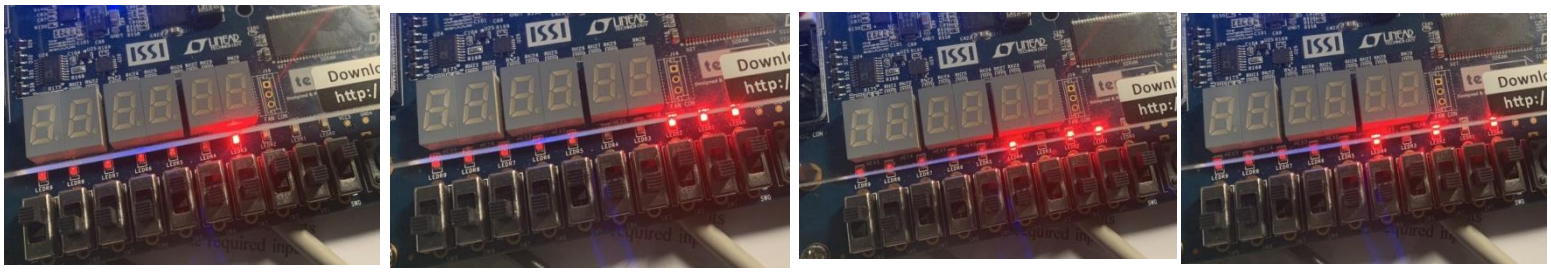
這邊開始有點複雜了起來，想非常久，尤其是 1010-1111 的部分如何轉換成 10-15，同時學到了很多 AND OR 搭配 Wire 的用法，算是這次實驗中最難的部分。

三、3-3

(1)實驗程式碼:

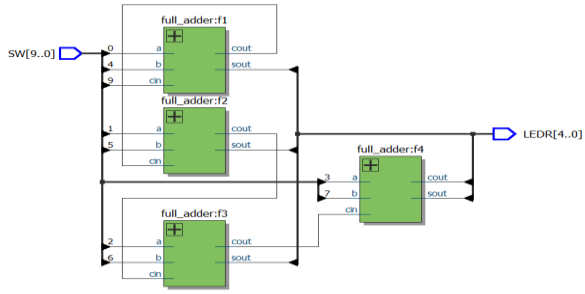
```
module labthree3(SW, LEDR);
input [9:0]SW;

wire [3:0]s;
wire [3:0]c;
output [4:0]LEDR;

full_adder f1(SW[0], SW[4], SW[9], c[0], s[0]);
full_adder f2(SW[1], SW[5], c[0], c[1], s[1]);
full_adder f3(SW[2], SW[6], c[1], c[2], s[2]);
full_adder f4(SW[3], SW[7], c[2], c[3], s[3]);

assign LEDR[0]=s[0];
assign LEDR[1]=s[1];
assign LEDR[2]=s[2];
assign LEDR[3]=s[3];
assign LEDR[4]=c[3];

endmodule

module full_adder(a,b,cin,cout,sout);
input a;
input b;
input cin;
output cout;
output sout;
wire C1, C2, S1;

process(a,b,C1,S1);
process(S1,cin,C2,sout);
assign cout=C1|C2;

endmodule

module process(x,y,C,S);
input x,y;
output C, S;

assign C=(x&y);
assign S=(~x&y)|(x&~y);

endmodule
```

(2)實驗結果:



//(由左至右，sw[9], sw[7:0]):1, 00100101/0, 00100101/1, 11001001/0, 11001001

(3)RTL 布局



(4)問題與討論:

是第一次學到多個加法器的運作，利用 XOR 和 AND 結合，產生出對應的 C、S，覺得是這次實驗中較簡單好懂的部分

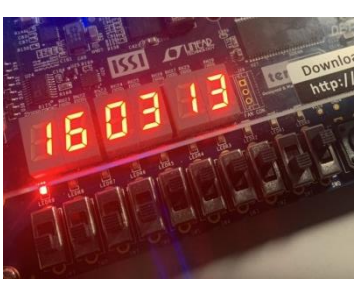
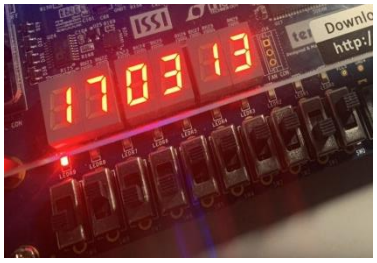
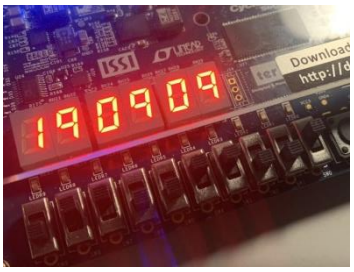
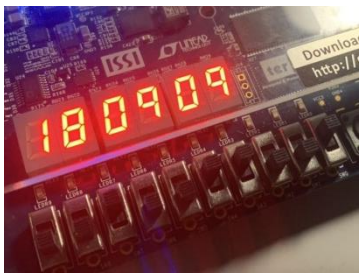
四、3-4

(1) 實驗程式碼:

```
module labthree4(SW,HEX5,HEX4,HEX3,HEX2,HEX1,HEX0,LEDR);
input [8:0]SW;
output [6:0]HEX5,HEX4,HEX3,HEX2,HEX1,HEX0;
wire signa, signb,signc,signd,signe,signf ;

wire [3:0]v,y,z;
output [9:0]LEDR;
comparator compa(0,SW[7],SW[6],SW[5],SW[4],signa,signb);
circuitA a(SW[7:4],signa,signb,v[3:0]);
circuitB b(signa,signb,HEX3);
char_7seg c(v,HEX2);



comparator compb(0,SW[3],SW[2],SW[1],SW[0],signc,signd);
circuitA d(SW[3:0],signc,signd,y[3:0]);

circuitB e(signc,signd,HEX1);
char_7seg f(y,HEX0);

assign LEDR[9]=(signa|signc);
wire [3:0]ss;
wire [3:0]cc;


full_adder f1(SW[0], SW[4], SW[8], cc[0], ss[0]);
full_adder f2(SW[1], SW[5], cc[0], cc[1], ss[1]);
full_adder f3(SW[2], SW[6], cc[1], cc[2], ss[2]);
full_adder f4(SW[3], SW[7], cc[2], cc[3], ss[3]);

comparator compc(cc[3],ss[3],ss[2],ss[1],ss[0],signe,signf);
circuitA g(ss[3:0],signe,signf,z[3:0]);

circuitB h(signe,signf,HEX5);

char_7seg i(z,HEX4);
endmodule

module comparator(top,a,b,c,d,sign,signn);
input a,b,c,d,top;
output sign,signn ;
wire  s5,s4,s3,s2,s1,s0;
assign sign = s5|s4|s3|s2|s1|s0;
assign signn=top;
assign s5 = a&~b&c&~d;
assign s4 = a&~b&c&d;
assign s3 = a&b&~c&~d;
assign s2 = a&b&~c&d;
assign s1 = a&b&c&~d;
assign s0 = a&b&c&d;
endmodule

module circuitA(SS,sign,signn,v);
input [3:0]SS;
input sign,signn;
wire [3:0]x,y;
output [3:0]v;

assign x[0]=(~SS[2]&SS[1]&~SS[0]&0)||(~SS[2]&SS[1]&SS[0]&1)||(SS[2]&~SS[1]&~SS[0]&0)||(SS[2]&~SS[1]&SS[0]&1)||(SS[2]&SS[1]&~SS[0]&0)||(SS[2]&SS[1]&SS[0]&1);
assign x[1]=(~SS[2]&SS[1]&~SS[0]&0)||(~SS[2]&SS[1]&SS[0]&0)||(SS[2]&~SS[1]&~SS[0]&1)||(SS[2]&~SS[1]&SS[0]&1)||(SS[2]&SS[1]&~SS[0]&0)||(SS[2]&SS[1]&SS[0]&0);
assign x[2]=(~SS[2]&SS[1]&~SS[0]&0)||(~SS[2]&SS[1]&SS[0]&0)||(SS[2]&~SS[1]&~SS[0]&0)||(SS[2]&~SS[1]&SS[0]&0)||(SS[2]&SS[1]&~SS[0]&1)||(SS[2]&SS[1]&SS[0]&1);

assign y[0]=(~SS[2]&~SS[1]&~SS[0]&0)||(~SS[2]&~SS[1]&SS[0]&1)||(~SS[2]&SS[1]&~SS[0]&0)||(~SS[2]&SS[1]&SS[0]&1);
assign y[1]=(~SS[2]&~SS[1]&~SS[0]&1)||(~SS[2]&~SS[1]&SS[0]&1)||(~SS[2]&SS[1]&~SS[0]&0)||(~SS[2]&SS[1]&SS[0]&0);
assign y[2]=(~SS[2]&~SS[1]&~SS[0]&1)||(~SS[2]&~SS[1]&SS[0]&1)||(~SS[2]&SS[1]&~SS[0]&0)||(~SS[2]&SS[1]&SS[0]&0);
assign y[3]=(~SS[2]&~SS[1]&~SS[0]&0)||(~SS[2]&~SS[1]&SS[0]&0)||(~SS[2]&SS[1]&~SS[0]&1)||(~SS[2]&SS[1]&SS[0]&1);

assign v[0]=(~sign&~signn&SS[0])||(sign&~signn&x[0])||(~sign&signn&y[0]);

 assign v[1]=(~sign&~signn&SS[1])||(sign&~signn&x[1])||(~sign&signn&y[1]);
 assign v[2]=(~sign&~signn&SS[2])||(sign&~signn&x[2])||(~sign&signn&y[2]);
 assign v[3]=(~sign&~signn&SS[3])||(sign&~signn&0)||(~sign&signn&y[3]);


 endmodule

 module circuitB(a,b,hex);
 input a,b;
 output [6:0]hex;
 wire c;
 assign c=a|b;

 assign hex[0]=(c&1);
 assign hex[1]=(c&0);
 assign hex[2]=(c&0);
 assign hex[3]=(c&1);
 assign hex[4]=(c&1);
 assign hex[5]=(c&1);
 assign hex[6]=(c&1)||(~c&1);
 endmodule


 module char_7seg(SW,HEX);
 input [3:0]SW;
 output [6:0]HEX;

 assign HEX[0]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&~SW[1]&SW[0])||(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&SW[0]&0)||(~SW[3]&SW[2]&~SW[1]&~SW[0]
 assign HEX[1]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&~SW[1]&SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&SW[0]&0)||(~SW[3]&SW[2]&~SW[1]&~SW|
 assign HEX[2]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&~SW[1]&SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&~SW[0])||(~SW[3]&~SW[2]&SW[1]&SW[0]&0)||(~SW[3]&SW[2]&~SW[1]&~SW[0]
 assign HEX[3]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&~SW[1]&SW[0])||(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&SW[0]&0)||(~SW[3]&SW[2]&~SW[1]&~SW[0]
 assign HEX[4]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&~SW[1]&SW[0])||(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&SW[0])||(~SW[3]&SW[2]&~SW[1]&~SW[0])||
 assign HEX[5]=(~SW[3]&~SW[2]&~SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&~SW[1]&SW[0])||(~SW[3]&~SW[2]&SW[1]&~SW[0])||(~SW[3]&~SW[2]&SW[1]&SW[0])||(~SW[3]&SW[2]&~SW[1]&~SW[0]&0)||
 assign HEX[6]=(~SW[3]&~SW[2]&~SW[1]&~SW[0])||(~SW[3]&~SW[2]&~SW[1]&SW[0])||(~SW[3]&~SW[2]&SW[1]&~SW[0]&0)||(~SW[3]&~SW[2]&SW[1]&SW[0]&0)||(~SW[3]&SW[2]&~SW[1]&~SW[0]&
 endmodule

 module full_adder(a,b,cin,cout,sout);
 input a;
 input b;
 input cin;
 output cout;
 output sout;
 wire C1, C2, S1;

 process(a,b,C1,S1);
 process(S1,cin,C2,sout);
 assign cout=C1|C2;

 endmodule

 module process(x,y,C,S);
 input x,y;
 output C, S;

 assign C=(x&y);
 assign S=(~x&y)|(x&~y);

 endmodule
```

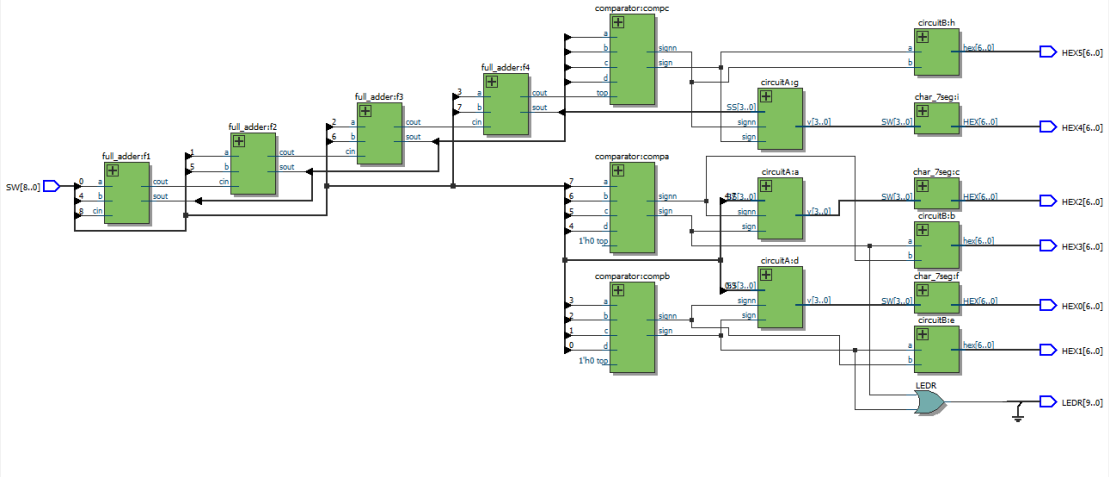
//95-101 同 LAB 3-1，故部分省略

(2)實驗結果:

//由左至右，SW[8:0]:000111101/100111101/110011001/010011001

(3)RTL 布局:



(4)問題與討論:

真的非常複雜，加上我的寫法用了許多 module 和參數，導致 debug 上花很多時間，學到了最重要的就是要把 module 的參數名稱好好定義，否則超容易混淆，另外也算是更熟悉 call 不同 module 的語法和細節了。