

1. 組別：第 22 組
2. 組員：404415073 電機二 蔡孟勳 404415055 電機二 劉恩瑞
3. 題目名稱：實驗 8 有限狀態機
4. 功能說明：

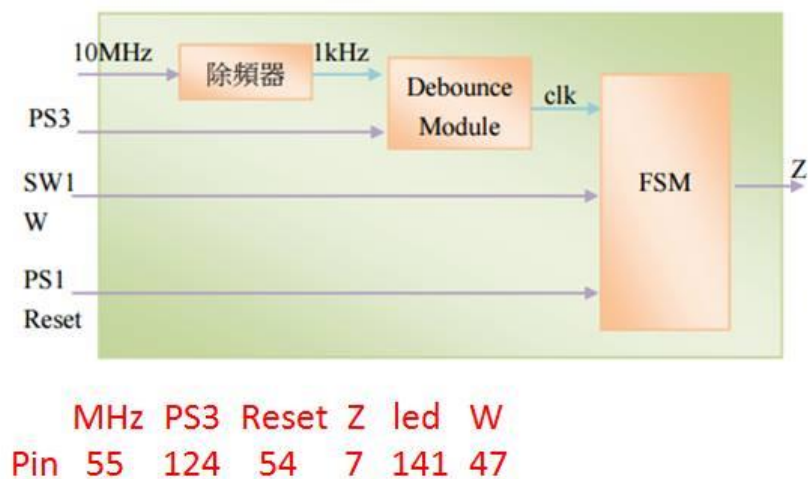
這次的實驗為撰寫一個有限狀態機

輸入訊號為任意 0 或 1 的訊號

輸出訊號為特定狀態下會亮起的 LED 燈

整個電路的功能為：可得知特定狀態的有限狀態機

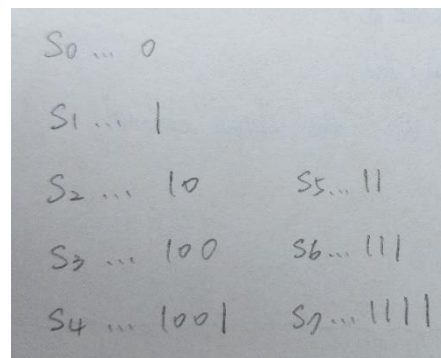
5. 硬體架構圖：



電路設計的想法:

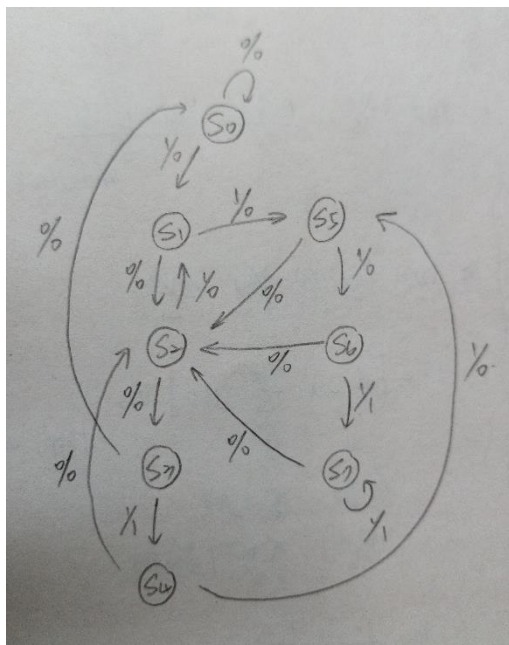
這次的實驗有兩種做法：Mealy-type 和 Moore-type，我們這組選擇 Moore-type 的做法。題目要求為：在連續輸入 1001 或 1111 的情況下，輸出為 1，其餘情況則輸出皆為 0，其實只要把可能的狀態列出來，然後決定目前輸入 0 的話要跳到哪個狀態，輸入 1 的話又該跳到哪個狀態，最後在題目要求的情況下使輸出為 1，即可完成這次的實驗。

State description →

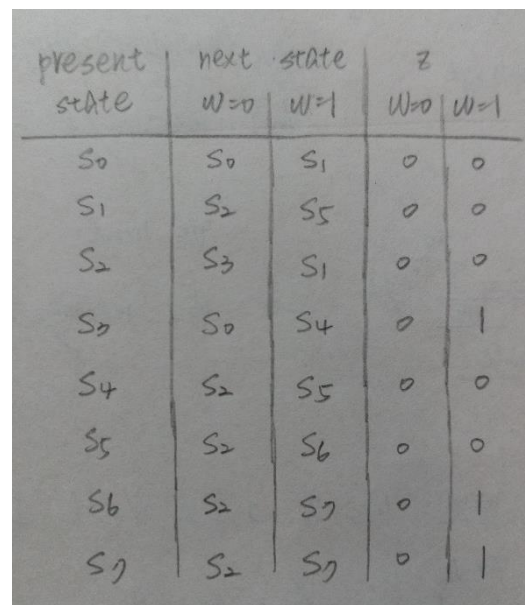


$S_0 \dots 0$	
$S_1 \dots 1$	
$S_2 \dots 10$	$S_5 \dots 11$
$S_3 \dots 100$	$S_6 \dots 111$
$S_4 \dots 1001$	$S_7 \dots 1111$

State graph ↓



State table ↓



present state	next state		Z	
	W=0	W=1	W=0	W=1
S_0	S_0	S_1	0	0
S_1	S_2	S_5	0	0
S_2	S_3	S_1	0	0
S_3	S_0	S_4	0	1
S_4	S_2	S_5	0	0
S_5	S_2	S_6	0	0
S_6	S_2	S_7	0	1
S_7	S_2	S_7	0	1

6. 程式碼&註解：

```
1 module exp8(Z,led,MHz,PS3,W,Reset); //有限狀態機模組宣告
2   input MHz,PS3,W,Reset; //輸入為MHz,PS3,W,Reset
3   output Z,led; //輸出為Z,led
4   wire kHz,clk; //有限狀態機的邏輯電路圖中會用到2條線連接不同模組
5   reg [3:0]state,STATE; //保留state[0]~[3],STATE[0]~[3]的值到下一次指定新值
6
7   //定義固定的值s0為0,s1為1,s2為2,s3為4,s4為9,s5為3,s6為7,s7為15
8   parameter s0=4'd0,s1=4'd1,s2=4'd2,s3=4'd4,s4=4'd9,s5=4'd3,s6=4'd7,s7=4'd15;
9
10  div10000 d1 (kHz,MHz); //使用除頻器1的模組，輸出為kHz，輸入為MHz
11  debounce d2 (clk,PS3,kHz); //使用消除彈跳模組，輸出為clk，輸入為PS3,kHz
12
13  always@(state,W) //當state,w的值有改變時，底下的Behavioral Model的敘述會被執行
14  case(state) //state的值為true時，執行以下敘述
15    s0:STATE=W?s1:s0; //當state為s0時，W=1,STATE=s1，W=0,STATE=s0
16    s1:STATE=W?s5:s2; //當state為s1時，W=1,STATE=s5，W=0,STATE=s2
17    s2:STATE=W?s1:s3; //當state為s2時，W=1,STATE=s1，W=0,STATE=s3
18    s3:STATE=W?s4:s0; //當state為s3時，W=1,STATE=s4，W=0,STATE=s0
19    s4:STATE=W?s5:s2; //當state為s4時，W=1,STATE=s5，W=0,STATE=s2
20    s5:STATE=W?s6:s2; //當state為s5時，W=1,STATE=s6，W=0,STATE=s2
21    s6:STATE=W?s7:s2; //當state為s6時，W=1,STATE=s7，W=0,STATE=s2
22    s7:STATE=W?s7:s2; //當state為s7時，W=1,STATE=s7，W=0,STATE=s2
23    default:STATE=s0; //state為其他值時，STATE=s0
24  endcase //結束case
25
26  always@(posedge clk or negedge Reset) //當clk正緣觸發或Reset負緣觸發，底下的Behavioral Model的敘述會被執行
27  if(~Reset) //如果Reset反相為true，執行以下敘述
28    state<=s0; //state的值變為s0
29  else //其他情況
30    state<=STATE; //state的值變為STATE的值
31
32  assign Z=(state==s4 || state==s7); //宣告Z的值為state==s4且state==s7的邏輯判斷
33  assign led=1'b1; //宣告led的值為1
34
35  endmodule //結束模組
36
37 module div10000(out,in); //除頻器除以10000的模組宣告
38   input in; //輸入為in
39   output reg out; //保留輸出out的值到下一次指定新值
40   reg [12:0]counter; //保留輸出counter[0]~[12]的值到下一次指定新值
41
42   always@(posedge in) //當in正緣觸發時，底下的Behavioral Model的敘述會被執行
43   if(counter==4999) //如果counter的值等於4999時，執行以下敘述
44     begin //開始
45       counter<=0; //counter的值變為0
46       out<=~out; //out反相
47     end //結束
48   else //其他情況
49     counter<=counter+13'd1; //counter的值+1
50
51  endmodule //結束模組
52
53 module debounce(out,in,kHz); //消除彈跳的模組宣告
54   input kHz,in; //輸入為kHz,in
55   output out; //輸出為out
56   reg [10:0]d; //保留d[0]~d[10]的值到下一次指定新值
57
58   always@(posedge kHz) //當kHz正緣觸發時，底下的Behavioral Model的敘述會被執行
59   begin //開始
60     d[10]<=d[9]; //d[10]的值變為d[9]的值
61     d[9]<=d[8]; //d[9]的值變為d[8]的值
62     d[8]<=d[7]; //d[8]的值變為d[7]的值
63     d[7]<=d[6]; //d[7]的值變為d[6]的值
64     d[6]<=d[5]; //d[6]的值變為d[5]的值
65     d[5]<=d[4]; //d[5]的值變為d[4]的值
66     d[4]<=d[3]; //d[4]的值變為d[3]的值
67     d[3]<=d[2]; //d[3]的值變為d[2]的值
68     d[2]<=d[1]; //d[2]的值變為d[1]的值
69     d[1]<=d[0]; //d[1]的值變為d[0]的值
70     d[0]<=in; //d[0]的值變為in的值
71   end //結束
72
73   and(out,d[10],d[9],d[8],d[7],d[6],d[5],d[4],d[3],d[2],d[1],d[0]); //邏輯AND，輸入為d[0]~d[10]；輸出為out
74
75  endmodule //消除彈跳的模組結束
```

7. 心得：

404415073 蔡孟勳

這次的實驗真的滿簡單的，state graph 和 state table 寫出來後，只要轉為程式碼即可，可是之後的實驗應該會跟數位系統設計密切相關，可能要先理解那堂課所教的內容，才比較能清楚知道實驗的原理，程式也才比較能打得出來。