

第六次實驗報告

題目:6-1~6-3

姓名:羅名志

學號:0813228

繳交日期:2022/4/15

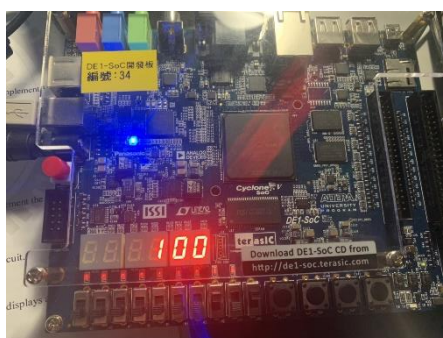
一、6-1

(1)實驗程式碼:

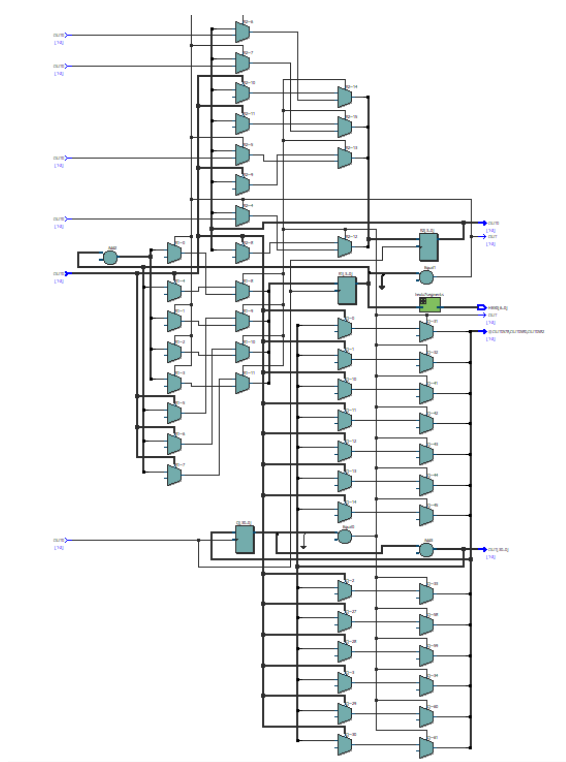
```
1 module labsix1(KEY,CLOCK_50,HEX0,HEX1,HEX2);
2   input [0:0]KEY;
3   input  CLOCK_50;
4   output [6:0]HEX0;
5   output [6:0]HEX1;
6   output [6:0]HEX2;
7   reg [30:0]Q;
8   reg [3:0]R1,R2,R3;
9
10
11 always @(posedge CLOCK_50)begin
12   if(Q == 50000000)begin
13     Q <= 0;
14     if(R1 == 9)begin
15       R1 <= 0;
16       if(R2 == 9)begin
17         R2<=0;
18         R3<=R3+1;
19       end
20     else
21       R2<=R2+1;
22     end
23     else
24       R1 <= R1+1;
25     end
26     else begin
27       Q <= Q+1;
28       if(KEY[0]==0) begin
29         R1<=0;
30       end
31     end
32   end
33 end
34
35 hexto7segment s(R1,HEX0);
36 hexto7segment ss(R2,HEX1);
37 hexto7segment sss(R3,HEX2);
38
39 endmodule
40
41 module hexto7segment (
42   input [3:0] iDIG,
43   output reg [6:0] oSEG
44 );
45
46 always@(iDIG) begin
47   case(iDIG)
48     4'b0001: oSEG = 7'b1111001;
49     4'b0010: oSEG = 7'b0100100;
50     4'b0011: oSEG = 7'b0110000;
51     4'b0100: oSEG = 7'b0011001;
52     4'b0101: oSEG = 7'b0010010;
53     4'b0110: oSEG = 7'b0000010;
54     4'b0111: oSEG = 7'b1111000;
55     4'b1000: oSEG = 7'b0000000;
56     4'b1001: oSEG = 7'b0011000;
57     4'b0000: oSEG = 7'b1000000;
58   endcase
59 end
60
61 endmodule
```

(2)實驗結果:





(3)RTL 布局:



(4)問題與討論:

這部分實驗基本上沒太大難度，利用 BCD 轉成 10 進位的計數器，然後利用 if-else 處理進位的問題，整體上來說算簡單。

二、6-2

(1)實驗程式碼

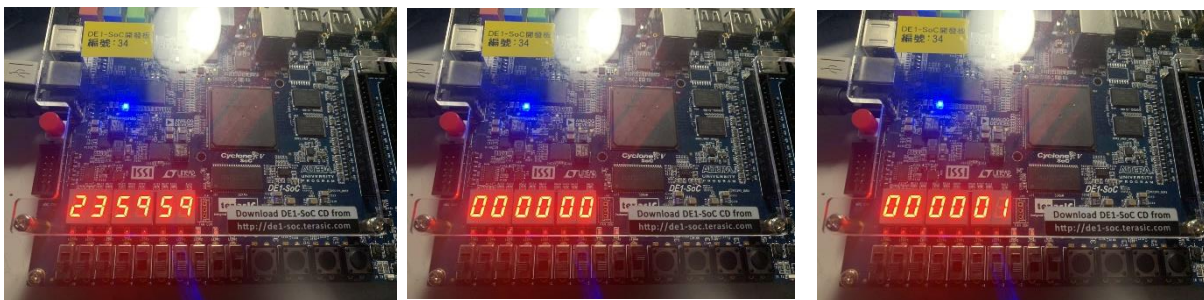
```
1 module labsix2(SW,KEY,CLOCK_50,HEX0,HEX1,HEX2,HEX3,HEX4,HEX5);
2     input [0:0]KEY;
3     input CLOCK_50;
4     output [6:0]HEX0;
5     output [6:0]HEX1;
6     output [6:0]HEX2;
7     output [6:0]HEX3;
8     output [6:0]HEX4;
9     output [6:0]HEX5;
10    reg [30:0]Q;
11    reg [3:0]R1,R2,R3,R4,R5,R6;
12    input [9:0]SW;
13    reg temp;
14
15
16    always @(posedge CLOCK_50)begin
17        if(Q == 50000000)begin
18            Q <= 0;
19            if(R6==9) begin
20                R6<=0;
21                if(R5==5)begin
22                    R5<=0;
23                    if(R4==9) begin
24                        R4<=0;
25                        if(R3==5)begin
26                            R3<=0;
27                            if(R2==9 || (R2==3 & R1==2)) begin
28                                R2<=0;
29                                if(R2==3 & R1==2) begin
30                                    R1=0;
31                                end
32                                else begin
33                                    R1<=R1+1;
34                                end
35                            end
36                            else begin
37                                R2<=R2+1;
38                            end
39                        end
40                        else begin
41                            R3<=R3+1;
42                        end
43                    end
44                    else begin
45                        R4<=R4+1;
46                    end
47                end
48                else begin
49                    R5<=R5+1;
50                end
51            end
52        else begin
53            R6<=R6+1;
54        end
55    end
56    else begin
57        Q <= Q+1;
58        if(temp & SW[8]==1) begin
59            if(SW[9]==1) begin
60                temp<=0;
61                R2<=SW[3:0];
62                R1<=SW[7:4];
63                R5<=0;
64                R6<=0;
65                Q<=0;
66            end
67            else begin
68                temp<=0;
69                R4<=SW[3:0];
70                R3<=SW[7:4];
71                R5<=0;
72                R6<=0;
73                Q<=0;
74            end
75        end
76    end
77    else begin
78        if(~temp & ~SW[8]) begin
79            temp<=1;
80        end
81    end
82    end
83    end
84    end
85    end
86
87    hexto7segment s(R6,HEX0);
88    hexto7segment s1(R5,HEX1);
89    hexto7segment s2(R4,HEX2);
90    hexto7segment s3(R3,HEX3);
91    hexto7segment s4(R2,HEX4);
92    hexto7segment s5(R1,HEX5);
93
94    endmodule
95
96    module hexto7segment (
97        input [3:0] iDIG,
98        output reg [6:0] oSEG
99    );
100
```

```

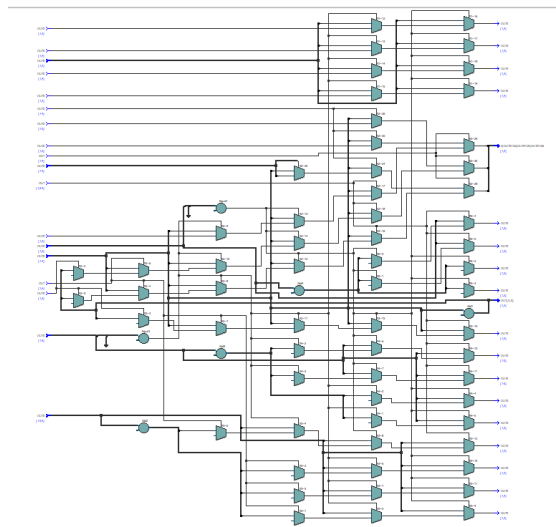
101  always@(idig) begin
102  case(idig)
103      4'b0001: oSEG = 7'b1111001;
104      4'b0010: oSEG = 7'b0100100;
105      4'b0011: oSEG = 7'b0110000;
106      4'b0100: oSEG = 7'b0011001;
107      4'b0101: oSEG = 7'b0010010;
108      4'b0110: oSEG = 7'b0000010;
109      4'b0111: oSEG = 7'b1111000;
110      4'b1000: oSEG = 7'b0000000;
111      4'b1001: oSEG = 7'b0011000;
112      4'b0000: oSEG = 7'b1000000;
113  endcase
114  end
115
116  endmodule

```

(2)實驗結果:



(3) RTL 布局



(4)問題與討論:

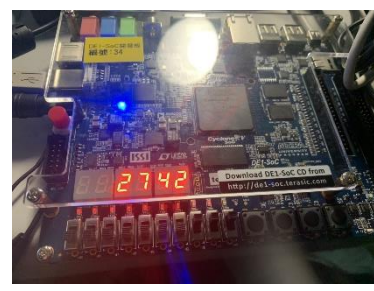
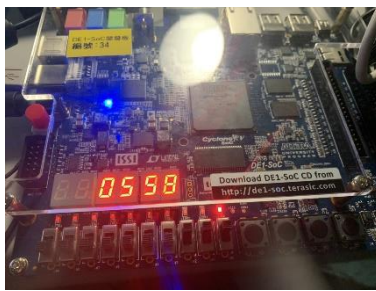
Posedge clock、set time 處理蠻久的，因為不能使用兩個 always 同時改動同個 reg，因此耗費了很多時間在處理這個 bug，但後來多利用一個變數表示 SW[8] 的 posedge 訊號，然後包在同一個 always 內就能順利解決，算是這次 lab 學到最多的地方。

三、6-3

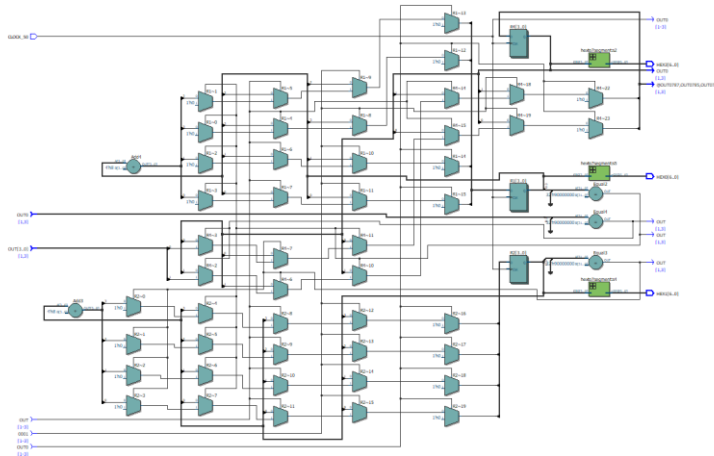
(1)實驗程式碼:

```
1 module labsix3(LED, SW, KEY, CLOCK_50, HEX0, HEX1, HEX2, HEX3);
2   input [3:0] KEY;
3   output reg [0:0] LED;
4   input CLOCK_50;
5   output [6:0] HEX0;
6   output [6:0] HEX1;
7   output [6:0] HEX2;
8   output [6:0] HEX3;
9   reg [30:0] Q, D;
10  reg [3:0] s, start, sig, sig2, R4, R3, R2, R1;
11  input [9:0] SW;
12
13
14  always @(posedge CLOCK_50) begin
15    if (start == 1) begin
16      if (Q == s'50000000) begin
17        Q <= 0;
18        sig <= 1;
19        LED[0] <= 1;
20      end
21    end
22    else begin
23      Q <= Q + 1;
24    end
25  end
26
27  if (start == 0) begin
28    LED[0] <= 0;
29    sig <= 0;
30  end
31
32  if (sig == 1) begin
33    if (D == 50000) begin
34      D <= 0;
35      if (R1 == 9) begin
36        R1 <= 0;
37        if (R2 == 9) begin
38          R2 <= 0;
39          if (R3 == 9) begin
40            R3 <= 0;
41            R4 <= R4 + 1;
42          end
43        end
44      end
45    end
46    else begin
47      R3 <= R3 + 1;
48    end
49  end
50  else begin
51    R2 <= R2 + 1;
52  end
53  end
54  else begin
55    R1 <= R1 + 1;
56  end
57  end
58  else begin
59    D <= D + 1;
60  end
61  end
62  if (sig == 0 && sig2 == 0) begin
63    R1 <= 0;
64    R2 <= 0;
65    R3 <= 0;
66    R4 <= 0;
67  end
68
69
70
71
72
73
74
75  always @(negedge KEY[0], negedge KEY[3]) begin
76    if (KEY[0] == 0) begin
77      start <= 1;
78      s <= SW[3:0];
79      sig2 <= 0;
80    end
81    if (KEY[3] == 0) begin
82      start <= 0;
83      sig2 <= 1;
84    end
85  end
86
87  hexto7segment s2(R4, HEX3);
88  hexto7segment s3(R3, HEX2);
89  hexto7segment s4(R2, HEX1);
90  hexto7segment s5(R1, HEX0);
91
92  endmodule
93
94  module hexto7segment (
95    input [3:0] iDIG,
96    output reg [6:0] oSEG
97  );
98
99  always @(iDIG) begin
100    case (iDIG)
101      4'b0001: oSEG = 7'b1111001;
102      4'b0010: oSEG = 7'b0100100;
103      4'b0011: oSEG = 7'b0110000;
104      4'b0100: oSEG = 7'b0011001;
105      4'b0101: oSEG = 7'b0010010;
106      4'b0110: oSEG = 7'b0000010;
107      4'b0111: oSEG = 7'b1111000;
108      4'b1000: oSEG = 7'b0000000;
109      4'b1001: oSEG = 7'b0011000;
110      4'b0000: oSEG = 7'b1000000;
111    endcase
112  end
113
114 endmodule
```

(2)實驗結果:



(3) RTL 布局



(4)問題與討論:

我其實寫完後不知道自己寫了甚麼，因為為了解決上個 part 提到的不能在不同的 `always` 指定 `reg` 值，因此設定了一堆變數去處理 `KEY` 的訊號和秒、毫秒的啟動時間，寫完非常複雜，但也更了解邏輯設計的基本觀念。