第六次實驗報告

題目:6-1~6-3

姓名:羅名志

學號:0813228

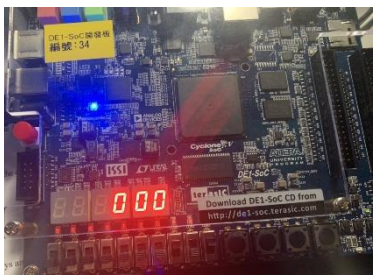
繳交日期:2022/4/15

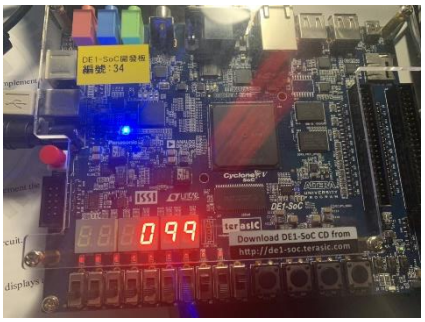
一、6-1

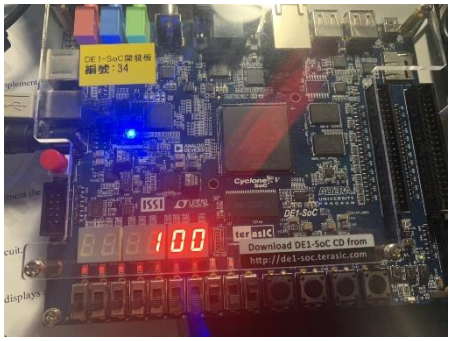
(1)實驗程式碼:

```
module labsix1(KEY,CLOCK_50,HEX0,HEX1,HEX2);
   input [0:0]KEY;
   input CLOCK_50;
   output [6:0]HEX0;
   output [6:0]HEX1;
   output [6:0]HEX2;
   reg [30:0]Q;
   reg [3:0]R1,R2,R3;


   always @(posedge CLOCK_50)begin
      if(Q == 50000000)begin
         Q <= 0;
         if(R1 == 9)begin
            R1 <= 0;
            if(R2 == 9)begin
            R2<=0;
            R3<=R3+1;
            end

            else
               R2<=R2+1;

         end
         else
            R1 <= R1+1;
      end
      else begin
         Q <= Q+1;
         if(KEY[0]==0) begin
            R1<=0;
   end
   end
   end


   hexto7segment s(R1,HEX0);
   hexto7segment ss(R2,HEX1);
   hexto7segment sss(R3,HEX2);

   endmodule

module hexto7segment (
   input [3:0] iDIG,
   output reg [6:0] oSEG
    );

 always@(iDIG) begin
   case(iDIG)
     4'b0001: oSEG = 7'b1111001;
     4'b0010: oSEG = 7'b0100100;
     4'b0011: oSEG = 7'b0110000;
     4'b0100: oSEG = 7'b0011001;
     4'b0101: oSEG = 7'b0010010;
     4'b0110: oSEG = 7'b0000010;
     4'b0111: oSEG = 7'b1111000;
     4'b1000: oSEG = 7'b0000000;
     4'b1001: oSEG = 7'b0011000;
     4'b0000: oSEG = 7'b1000000;
   endcase
 end

endmodule
```

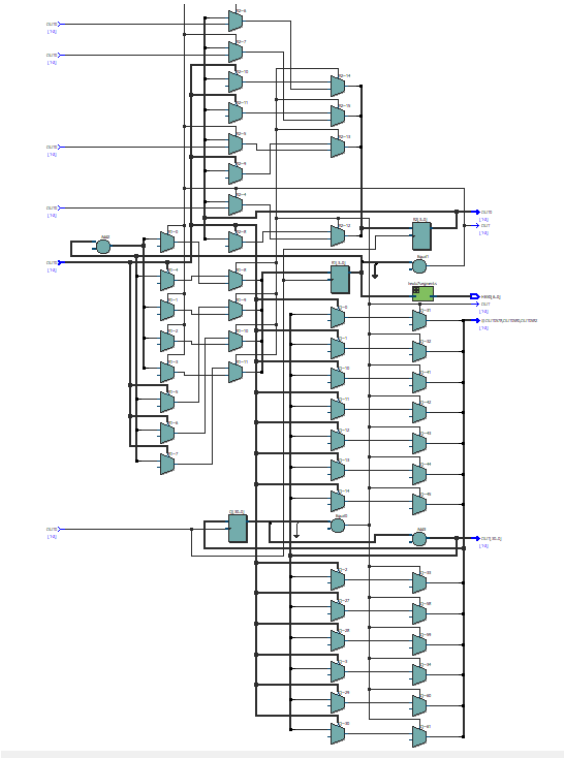
(2)實驗結果:







(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。

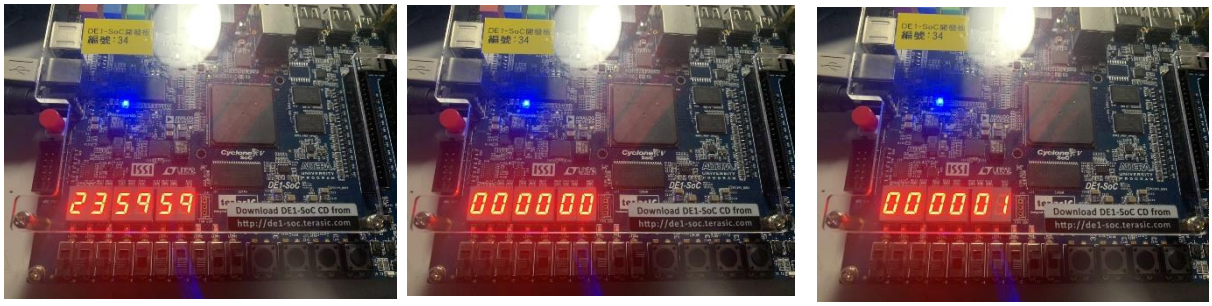
二、6-2

(1)實驗程式碼

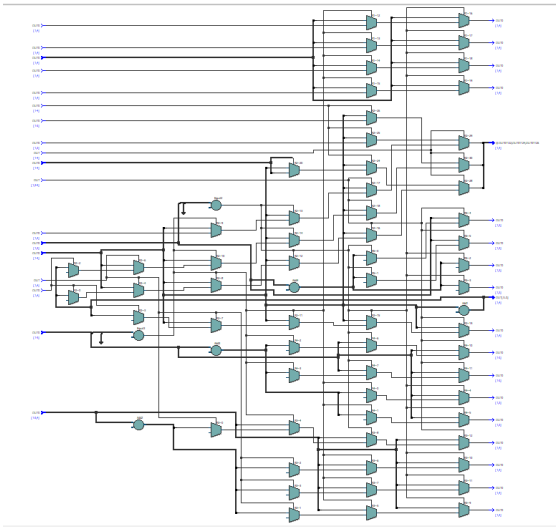
```
module labsix2(SW,KEY,CLOCK_50,HEX0,HEX1,HEX2,HEX3,HEX4,HEX5);
   input [0:0]KEY;
   input CLOCK_50;
   output [6:0]HEX0;
   output [6:0]HEX1;
   output [6:0]HEX2;
   output [6:0]HEX3;
   output [6:0]HEX4;
   output [6:0]HEX5;
   reg [30:0]Q;
   reg [3:0]R1,R2,R3,R4,R5,R6;
   input [9:0]SW;
   reg temp;


   always @(posedge CLOCK_50)begin
      if(Q == 50000000)begin
         Q <= 0;
         if(R6==9) begin
            R6<=0;
            if(R5==5)begin
               R5<=0;
               if(R4==9) begin
                  R4<=0;
                  if(R3==5)begin
                     R3<=0;
                     if(R2==9 || (R2==3 & R1==2)) begin
                        R2<=0;
                        if(R2==3 & R1==2) begin
                           R1=0;
                        end
                        else begin
                           R1<=R1+1;
                        end
                     end
                     else begin
                        R2<=R2+1;
                     end
                  end
                  else begin
                     R3<=R3+1;
                  end
               end
               else begin
                  R4<=R4+1;
               end
            end
            else begin
               R5<=R5+1;
            end
         end

         else begin
            R6<=R6+1;
         end
      end
      else begin
         Q <= Q+1;
         if(temp & SW[8]==1) begin
            if(SW[9]==1) begin
               temp<=0;
               R2<=SW[3:0];
               R1<=SW[7:4];
               R5<=0;
               R6<=0;
               Q<=0;
            else begin
               temp<=0;
               R4<=SW[3:0];
               R3<=SW[7:4];
               R5<=0;
               R6<=0;
               Q<=0;
            end
         end

         else begin
         if(~temp & ~SW[8]) begin
            temp<=1;
         end
         end
      end

end

   hexto7segment s(R6,HEX0);
   hexto7segment s1(R5,HEX1);
   hexto7segment s2(R4,HEX2);
   hexto7segment s3(R3,HEX3);
   hexto7segment s4(R2,HEX4);
   hexto7segment s5(R1,HEX5);

   endmodule

module hexto7segment (
   input [3:0] iDIG,
   output reg [6:0] oSEG
    );

```

```
always@(iDIG) begin
  case(iDIG)
    4'b0001: oSEG = 7'b1111001;
    4'b0010: oSEG = 7'b0100100;
    4'b0011: oSEG = 7'b0110000;
    4'b0100: oSEG = 7'b0011001;
    4'b0101: oSEG = 7'b0010010;
    4'b0110: oSEG = 7'b0000010;
    4'b0111: oSEG = 7'b1111000;
    4'b1000: oSEG = 7'b0000000;
    4'b1001: oSEG = 7'b0011000;
    4'b0000: oSEG = 7'b1000000;
  endcase
end

endmodule
```

(2)實驗結果:



(3) RTL 布局



(4)問題與討論:

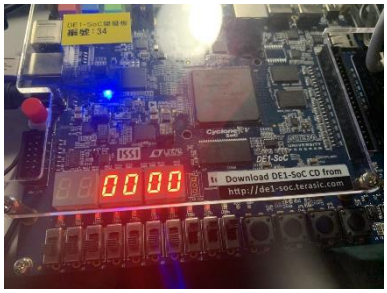
Posedge clock、set time 處理蠻久的，因為不能使用兩個 always 同時改動同個 reg，因此耗費了很多時間在處理這個 bug，但後來多利用一個變數表示 SW[8] 的 posedge 訊號，然後包在同一個 always 內就能順利解決，算是這次 lab 學到最多的地方。

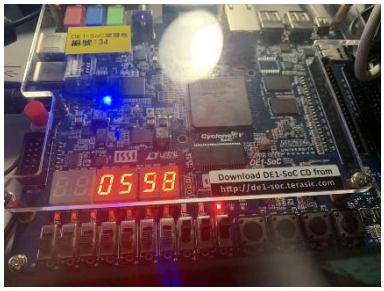
三、6-3

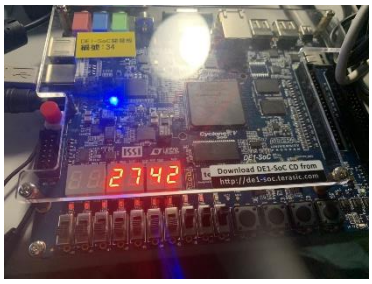
(1)實驗程式碼:

```
module labsix3(LEDR,SW,KEY,CLOCK_50,HEX0,HEX1,HEX2,HEX3);
   input [3:0]KEY;
   output reg [0:0] LEDR;
   input CLOCK_50;
   output [6:0]HEX0;
   output [6:0]HEX1;
   output [6:0]HEX2;
   output [6:0]HEX3;
   reg [30:0]Q,D;
   reg [3:0]s,start,sig,sig2,R4,R3,R2,R1;
   input [9:0]SW;


   always @(posedge CLOCK_50)begin
   if(start==1)begin
      if(Q == s*50000000)begin
         Q <= 0;
         sig<=1;
         LEDR[0]<=1;

      end
      else begin
      Q<=Q+1;

      end

      end
   if(start==0) begin
      LEDR[0]<=0;
      sig<=0;
   end

   if(sig==1)begin

       if(D == 50000)begin
          D <= 0;
          if(R1==9) begin
             R1<=0;
             if(R2==9) begin
                R2<=0;
                if(R3==9) begin
                   R3<=0;
                   R4<=R4+1;
                end

                else begin
                   R3<=R3+1;
                end
             end
             else begin
                R2<=R2+1;
             end
          end

          else begin
             R1<=R1+1;
          end
          end
       else begin
          D<=D+1;
       end
          end
    if(sig==0 && sig2==0) begin
             R1<=0;
             R2<=0;
             R3<=0;
             R4<=0;
          end


       end


   always@(negedge KEY[0], negedge KEY[3]) begin
      if(KEY[0]==0) begin
         start<=1;
         s<=SW[3:0];
         sig2=0;
      end
      if(KEY[3]==0) begin
         start<=0;
         sig2<=1;
      end
   end

   hexto7segment s2(R4,HEX3);
   hexto7segment s3(R3,HEX2);
   hexto7segment s4(R2,HEX1);
   hexto7segment s5(R1,HEX0);

   endmodule

module hexto7segment (
   input [3:0] iDIG,
   output reg [6:0] oSEG
    );

 always@(iDIG) begin
   case(iDIG)
      4'b0001: oSEG = 7'b1111001;
      4'b0010: oSEG = 7'b0100100;
      4'b0011: oSEG = 7'b0110000;
      4'b0100: oSEG = 7'b0011001;
      4'b0101: oSEG = 7'b0010010;
      4'b0110: oSEG = 7'b0000010;
      4'b0111: oSEG = 7'b1111000;
      4'b1000: oSEG = 7'b0000000;
      4'b1001: oSEG = 7'b0011000;
      4'b0000: oSEG = 7'b1000000;
   endcase
  end

endmodule
```

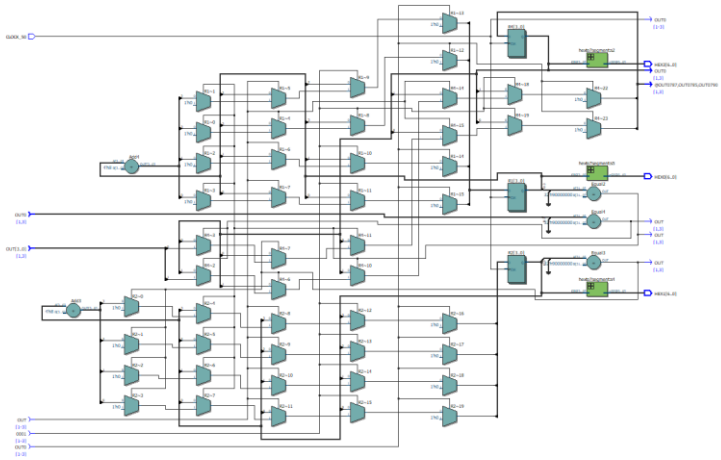
(2)實驗結果:







(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。