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 7 計時器

4. 功能說明：

這次的實驗為撰寫一個計時器

輸入訊號為

(1)　一個 10MHz 的 clock (當正緣觸發時可執行程式碼)、

(2)　一個脈波 (使計時器歸零並重新開始…即 Reset)、

(3)　兩個 0 或 1 的訊號…即 Enable 和 Speed

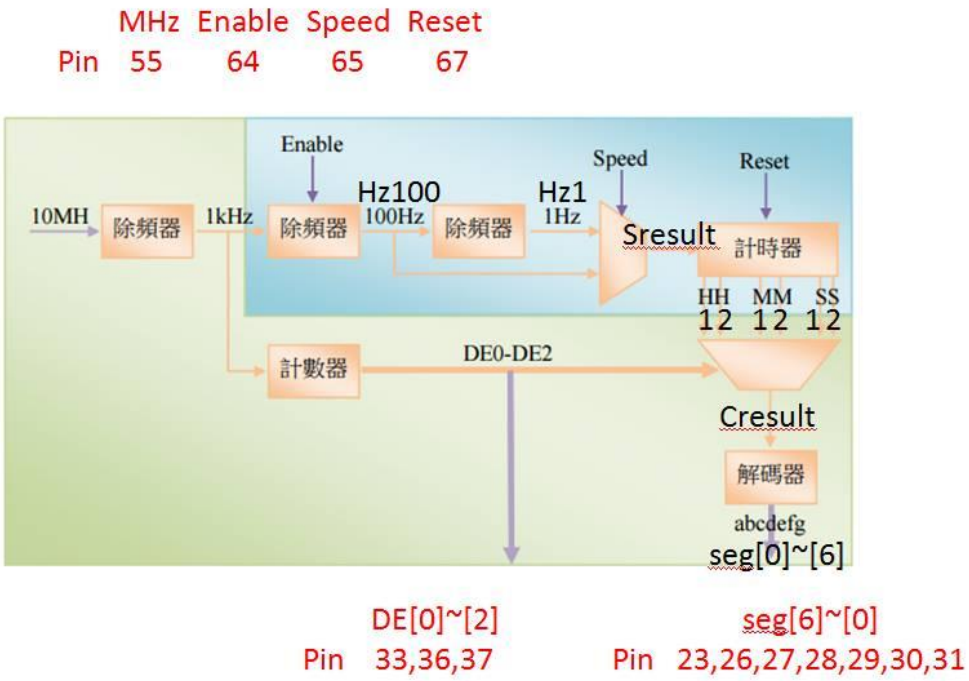
(分別控制計時器是否啟動與計時速度的快慢)

輸出訊號為

(1)　6 個七段顯示器 (分別用來顯示時鐘的時、分、秒)

整個電路的功能為：可調整速度快慢的計時器

5. 硬體架構圖：



電路設計的想法:

這次的實驗其實可以概略分為兩個區域：顯示區和計時區。先說計時區，其中包含三個部分：除頻器、多工器、計時器，根據題目要求，我們先設計兩個除頻器分別得到 1Hz 和 100Hz 的 clock，接著設計多工器，可以讓使用者選擇何種速度計時，然後就是整個實驗最難的部分，何時要進位、何時要歸零，都必須要仔細思考，最後的結果就是會根據 clock 的觸發來計時；另一個區域是顯示區，這部分就比較單純，隨著 clock 的觸發而計數，再根據計數的結果來決定要哪個七段顯示器亮,但由於視覺暫留,會感覺好像 6 個七段顯示器同時亮,以達到顯示完整的時鐘(時、分、秒)。

6. 程式碼&註解：

```
module exp7(DE, seg, MHz, Enable, Speed, Reset); //計時器模組宣告
 input MHz, Enable, Speed, Reset; //輸入為MHz,Enable,Speed,Reset
 output reg [2:0]DE; //輸出為3-bits共3個數DE[0]~[2]，並且保留DE[0]~[2]的值到下一次指定新值
 output reg [6:0]seg; //輸出為7-bits共7個數seg[0]~[6]，並且保留seg[0]~[6]的值到下一次指定新值
 wire kHz, Hz1, Hz100, Sresult; //計時器的邏輯電路圖中會用到4條線連接不同模組
 reg H1; //保留H1的值到下一次指定新值
 reg [3:0]Cresult; //保留Cresult[0]~[3]的值到下一次指定新值
 reg [3:0]S2, M2, H2; //保留S2[0]~[3],M2[0]~[3],H2[0]~[3]的值到下一次指定新值
 reg [2:0]S1, M1; //保留S1[0]~[2],M1[0]~[2]的值到下一次指定新值

 div10000 d1(kHz, MHz); //使用除頻器1的模組，輸出為kHz，輸入為MHz
 div10    d2(Hz100, kHz); //使用除頻器2的模組，輸出為Hz100，輸入為kHz
 div100   d3(Hz1, Hz100); //使用除頻器3的模組，輸出為Hz1，輸入為Hz100

 always@(posedge kHz) //當kHz正緣觸發時，底下的Behavioral Model的敘述會被執行
  if(DE == 5) //如果DE的值為5，執行以下敘述
   DE<=0; //DE的值變為0
  else //其他情況
   DE<=DE+3'd1; //DE的值+1

 assign Sresult = Speed ? Hz1 : Hz100; //當Speed=1時,Sresult=Hz1，當Speed=0時,Sresult=Hz100

 //當Sresult正緣觸發或Reset負緣觸發時，底下的Behavioral Model的敘述會被執行
 always@(posedge Sresult or negedge Reset)
  begin //開始
   if(~Reset) //如果Reset反相為true，執行以下敘述
    begin //開始
     S1<=0; //S1的值變為0
     S2<=0; //S2的值變為0
     M1<=0; //M1的值變為0
     M2<=0; //M2的值變為0
     H1<=0; //H1的值變為0
     H2<=0; //H2的值變為0
    end //結束
   else //其他情況
    begin //開始
     if(~Enable) //如果Enable反相為true，執行以下敘述
      begin //開始

       //如果S1的值為5且S2的值為9，執行以下敘述
       if(S1 == 5 && S2 == 9)
        begin //開始
         S1<=0; //S1的值變為0
         S2<=0; //S2的值變為0
        end //結束
       else //其他情況
        begin //開始
         if(S2 == 9) //如果S2的值為9，執行以下敘述
          begin //開始
           S2<=0; //S2的值變為0
           S1<=S1+3'd1; //S1的值+1
          end //結束
         else //其他情況
          S2<=S2+4'd1; //S2的值+1
        end //結束

       //如果M1的值為5且M2的值為9且S1的值為5且S2的值為9，執行以下敘述
       if(M1 == 5 && M2 == 9 && S1 == 5 && S2 == 9)
        begin //開始
         M1<=0; //M1的值變為0
         M2<=0; //M2的值變為0
        end //結束
       else //其他情況
        begin //開始
         if(M2 == 9 && S1 == 5 && S2 == 9) //如果M2的值為9且S1的值為5且S2的值為9，執行以下敘述
          begin //開始
           M2<=0; //M2的值變為0
           M1<=M1+3'd1; //M1的值+1
          end //結束
         else if(S1 == 5 && S2 == 9) //其他如果S1的值為5且S2的值為9，執行以下敘述
          M2<=M2+4'd1; //M2的值+1
        end //結束
```

```
    //如果H1的值為0且H2的值為9且M1的值為5且M2的值為9且S1的值為5且S2的值為9，執行以下敘述
    if(H1 == 0 && H2 == 9 && M1 == 5 && M2 == 9 && S1 == 5 && S2 == 9)
     begin //開始
      H2<=0; //H2的值變為0
      H1<=H1+1'd1; //H1的值+1
     end //結束
    else //其他情況
     begin //開始

       //如果H1的值為1且H2的值為1且M1的值為5且M2的值為9且S1的值為5且S2的值為9，執行以下敘述
      if(H1 == 1 && H2 == 1 && M1 == 5 && M2 == 9 && S1 == 5 && S2 == 9)
       begin //開始
        H1<=0; //H1的值變為0
        H2<=0; //H2的值變為0
       end //結束

       //其他如果M1的值為5且M2的值為9且S1的值為5且S2的值為9，執行以下敘述
      else if(M1 == 5 && M2 == 9 && S1 == 5 && S2 == 9)
       H2<=H2+4'd1; //H2的值+1
     end //結束
   end //結束

       case(DE) //DE的值為true時，執行以下敘述
         3'd0:Cresult<=H1; //DE的值為0，Cresult的值變為H1的值
         3'd1:Cresult<=H2; //DE的值為1，Cresult的值變為H2的值
         3'd2:Cresult<=M1; //DE的值為2，Cresult的值變為M1的值
         3'd3:Cresult<=M2; //DE的值為3，Cresult的值變為M2的值
         3'd4:Cresult<=S1; //DE的值為4，Cresult的值變為S1的值
         3'd5:Cresult<=S2; //DE的值為5，Cresult的值變為S2的值
       endcase //結束case

       case(Cresult) //Cresult的值為true時，執行以下敘述
        4'd0 : seg = 7'b1111_110; //使七段顯示器顯示0
        4'd1 : seg = 7'b0110_000; //使七段顯示器顯示1
        4'd2 : seg = 7'b1101_101; //使七段顯示器顯示2
        4'd3 : seg = 7'b1111_001; //使七段顯示器顯示3
        4'd4 : seg = 7'b0110_011; //使七段顯示器顯示4
        4'd5 : seg = 7'b1011_011; //使七段顯示器顯示5
        4'd6 : seg = 7'b1011_111; //使七段顯示器顯示6
        4'd7 : seg = 7'b1110_000; //使七段顯示器顯示7
        4'd8 : seg = 7'b1111_111; //使七段顯示器顯示8
        4'd9 : seg = 7'b1111_011; //使七段顯示器顯示9
       endcase //結束case
     end //結束
   end //結束
endmodule //結束模組

module div10000(out, in); //除頻器除以10000的模組宣告
  input in; //輸入為in
  output reg out; //保留輸出out的值到下一次指定新值
  reg [12:0]counter; //保留輸出counter[0]~[12]的值到下一次指定新值
  always@(posedge in) //當in正緣觸發時，底下的Behavioral Model的敘述會被執行
   if(counter == 4999) //如果counter的值等於4999時，執行以下敘述
    begin //開始
     counter<=0; //counter的值變為0
     out<=~out; //out反相
    end //結束
   else //其他情況
    counter<=counter+13'd1; //conuter的值+1
endmodule //結束模組

module div10(out, in); //除頻器除以10的模組宣告
  input in; //輸入為in
  output reg out; //保留輸出out的值到下一次指定新值
  reg [2:0]counter; //保留輸出counter[0]~[2]的值到下一次指定新值
  always@(posedge in) //當in正緣觸發時，底下的Behavioral Model的敘述會被執行
   if(counter == 4) //如果counter的值等於4時，執行以下敘述
    begin //開始
     counter<=0; //counter的值變為0
     out<=~out; //out反相
    end //結束
   else //其他情況
    counter<=counter+3'd1; //conuter的值+1
endmodule //結束模組
```

```
135 module div10(out, in); //除頻器除以10的模組宣告
136   input in; //輸入為in
137   output reg out; //保留輸出out的值到下一次指定新值
138   reg [2:0]counter; //保留輸出counter[0]~[2]的值到下一次指定新值
139   always@(posedge in) //當in正緣觸發時，底下的Behavioral Model的敘述會被執行
140    if(counter == 4) //如果counter的值等於4時，執行以下敘述
141      begin //開始
142       counter<=0; //counter的值變為0
143       out<=~out; //out反相
144      end //結束
145    else //其他情況
146     counter<=counter+3'd1; //conuter的值+1
147  endmodule //結束模組
148
149 module div100(out, in); //除頻器除以100的模組宣告
150   input in; //輸入為in
151   output reg out; //保留輸出out的值到下一次指定新值
152   reg [5:0]counter; //保留輸出counter[0]~[5]的值到下一次指定新值
153   always@(posedge in) //當in正緣觸發時，底下的Behavioral Model的敘述會被執行
154    if(counter == 49) //如果counter的值等於49時，執行以下敘述
155      begin //開始
156       counter<=0; //counter的值變為0
157       out<=~out; //out反相
158      end//結束
159    else //其他情況
160     counter<=counter+6'd1; //conuter的值+1
161  endmodule //結束模組
```

7. 問題討論：

這次的實驗，用到了許多之前實驗課教的東西，除頻器、計數器、多工器、解碼器、七段顯示器等，然而這次也有教到新的語法(task)，但我們在實作時，還是習慣使用呼叫 module 的方式，碰到的問題在於：task 被觸發的條件與整體被觸發的條件不一致，讓我們覺得使用 task 可能會有問題，因此我們還是先用 module 得到結果，再應用到 always 中。或許應該使用好幾個 always，找到相同觸發條件的情況時，再使用 task，這樣也能讓整體程式碼變得比較簡單易懂。

8. 心得：

404415073 蔡孟勳

這次的實驗，雖然有教到 task 的用法，但實作時我們沒有使用，感覺還是習慣呼叫 module 方式，或許下次應該嘗試看看。而整體最困難的部分就是「計時器」，何時進位、何時歸零的邏輯真的需要想清楚再打，我們也是修改了好多次才完成的。