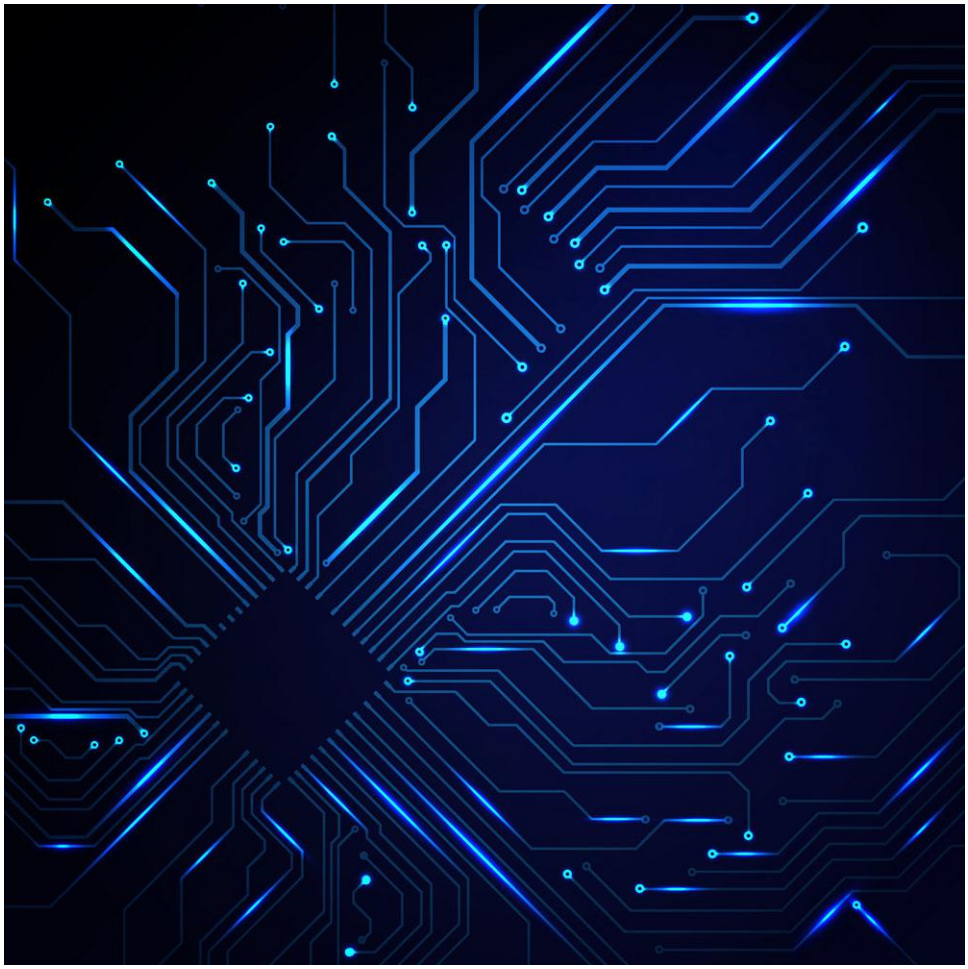


注意

1. 繳交時一律轉 PDF 檔
2. 繳交期限為
   上完課後
   當週五晚上 12 點前
3. 一人繳交一份
4. 檔名：學號_HW?.pdf
   檔名請按照作業檔名格式進行填寫
   未依照格式不予批改

2022/10/19

實驗五

FSM 練習

姓名：王嘉羽　　學號：00957116

班級：資工 3B

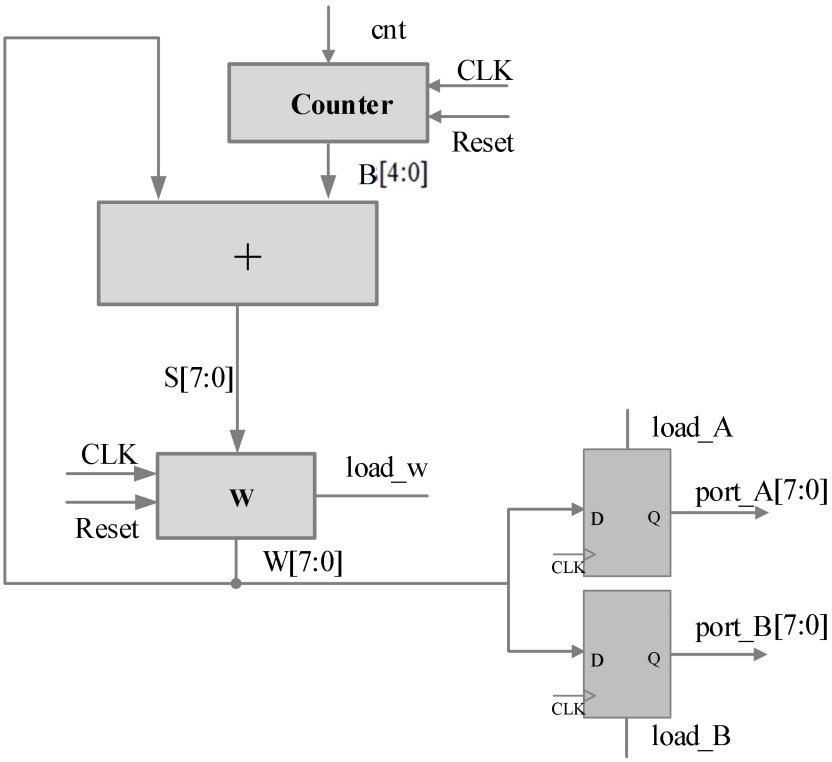
E-mail：vayne20011125@gmail.com

一、 Counter + register

● 實驗說明：

1. counter 數到 20 做累加運算，counter 加到 10 的時候將值傳到 port_A，加到 20 的時候傳到 port_B
2. 輸入：clk, reset
3. 輸出：port_A[7:0]、port_B[7:0]

● 系統硬體架構方塊圖（接線圖）：



● 系統架構程式碼、測試資料程式碼與程式碼說明(.sv 檔及.do 檔都要截圖)

截圖請善用 win+shift+S

◆ hw_1017_1.sv

```
1  `timescale 1ns/10ps
2  module hw_1017_1(
3      input clk,                      //時脈
4      input reset,                    //reset
5      output logic [7:0] port_A,      //output
6      output logic [7:0] port_B
7  );
8
9      logic load_w;             //load_w控制線
10     logic load_A;             //load_A控制線
11     logic load_B;             //load_B控制線
12     logic [4:0] ps,ns;        //現在狀態 下一個狀態
13     logic [7:0] B;            //計數器 中間的值
14     logic [7:0] S;            //S = W+B
15     logic [7:0] W;
16     logic cnt;
17
18     always_ff @(posedge clk)        //fsm
19     begin
20        if(reset)
21           ps <= #1 0;
22        else
23           ps <= #1 ns;
24     end
26     always_ff @(posedge clk)        //W載入器
27     begin
28        if(reset)
29           W  <= #1 0;
30        else if(load_w)
31           W  <= #1 S;
32     end
33
34     always_ff @(posedge clk)        //A載入器
35     begin
36        if(reset)
37           port_A   <= #1 0;
38        else if(load_A)
39           port_A   <= #1 W;
40     end
41
42     always_ff @(posedge clk)        //B載入器
43     begin
44        if(reset)
45           port_B   <= #1 0;
46        else if(load_B)
47           port_B   <= #1 W;
48     end
50     always_ff @(posedge clk)        //counter
51     begin
52        if(reset)
53           B  <= #1 0;
54        else if(cnt)
55           B  <= #1 B + 1;
56     end
57
58     assign S = W + B;
59
60     always_comb
61     begin
62     ns = 0;
63     load_w = 1;
64     load_A = 0;
65     load_B = 0;
66     cnt = 1;
67     case(ps)
68           0:
69           begin
70              ns = 1;
71           end
72
73           1:
74           begin
75              ns = 2;
76           end
```

```
78              2:
79              begin
80                  ns = 3;
81              end
82
83              3:
84              begin
85                  ns = 4;
86              end
87
88              4:
89              begin
90                  ns = 5;
91              end
92
93              5:
94              begin
95                  ns = 6;
96              end
97
98              6:
99              begin
100                 ns = 7;
101             end
103             7:
104             begin
105                 ns = 8;
106             end
107
108             8:
109             begin
110                 ns = 9;
111             end
112
113             9:
114             begin
115                 ns = 10;
116             end
117
118             10:
119             begin
120                 ns = 11;
121             end
122
123             11:
124             begin                   //在ns = 10時，w會得到(1+~+10)的結果
125                 load_A = 1;         //所以load_A要在n=11時load，才能拿到w的值
126                 ns = 12;
127             end
129             12:
130             begin
131                 ns = 13;
132             end
133
134             13:
135             begin
136                 ns = 14;
137             end
138
139             14:
140             begin
141                 ns = 15;
142             end
143
144             15:
145             begin
146                 ns = 16;
147             end
148
149             16:
150             begin
151                 ns = 17;
152             end
```

```
159             18:
160             begin
161                 ns = 19;
162             end
163
164             19:
165             begin
166                 ns = 20;
167             end
168
169             20:
170             begin
171                 ns = 21;
172             end
173
174             21:
175             begin
176                 load_B =1;          //在ns = 20時，w會得到(1+~+20)的結果
177                 cnt = 0;            //所以load_B要在n=21時load，才能拿到w的值
178                 ns = 22;
179             end
180
181             22:                     //停止狀態
182             begin
183                 load_w = 0;
184                 cnt = 0;
185                 ns = 22;
186             end
187         endcase
188     end
189
190
191 endmodule
192
```

◆ testbench.sv

simulation > tb > testbench.sv

```
1   `timescale 1ns/10ps
2   module testbench;
3
4       logic reset;                    //重置
5       logic clk;                      //時脈
6       logic [7:0] port_A;             //輸出
7       logic [7:0] port_B;
8
9       hw_1017_1 hw_1017_11(
10          .reset(reset), //()內的變數為tb的變數，"."後面為hw_1017_1.sv的變數，將2者對應起來
11          .clk(clk),
12          .port_A(port_A),
13          .port_B(port_B)
14      );
15
16      always #10 clk = ~clk;
17
18      initial begin
19          reset = 1;clk = 0; //一開始先reset，將時脈歸0
20          #15 reset = 0;
21          #1000 $stop;
22      end
23  endmodule
```

◆ sim.do

simulation > modelsim > sim.do

```
vsim -voptargs=+acc work.testbench
view structure wave signals

do wave.do

log -r *
run -all

```

◆ compile.do

simulation > modelsim > compile.do

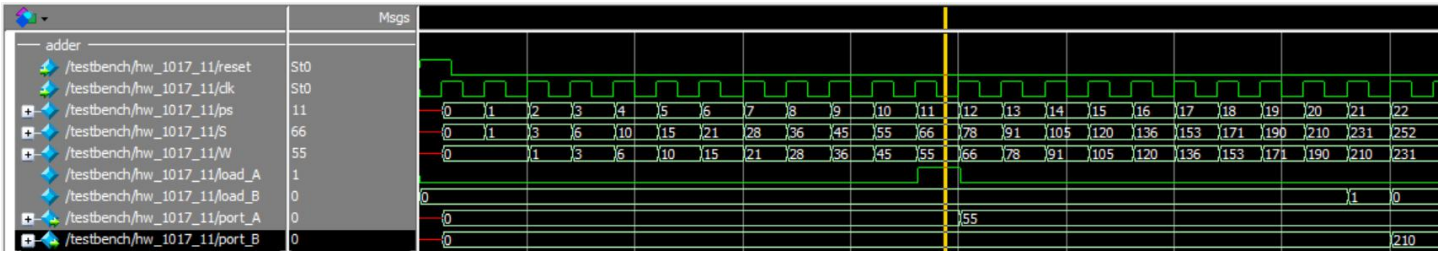
```
#vlib work



# ----------------------------------------------------------
vlog ../tb/testbench.sv
vlog ../../design/hw_1017_1.sv









# ----------------------------------------------------------
```

◆ wave.do

simulation > modelsim > wave.do

```
onerror {resume}
quietly WaveActivateNextPane {} 0

#add wave -noupdate -divider {TOP LEVEL INPUTS}

#add wave -noupdate -format Logic /testbench/clk
#add wave -noupdate -format Logic /testbench/rst



add wave -noupdate -divider {adder}

add wave -noupdate -format logic    /testbench/hw_1017_11/reset
add wave -noupdate -format logic    /testbench/hw_1017_11/clk
add wave -noupdate -format Literal -radix Unsigned      /testbench/hw_1017_11/ps
add wave -noupdate -format Literal -radix Unsigned      /testbench/hw_1017_11/S
add wave -noupdate -format Literal -radix Unsigned      /testbench/hw_1017_11/W
add wave -noupdate -format logic    /testbench/hw_1017_11/load_A
add wave -noupdate -format Literal -radix Unsigned      /testbench/hw_1017_11/load_B
add wave -noupdate -format Literal -radix Unsigned      /testbench/hw_1017_11/port_A
add wave -noupdate -format Literal -radix Unsigned      /testbench/hw_1017_11/port_B
```

● 模擬結果與結果說明：



要在 conuter 加到 10 的時候把值 load 到 port_A，但 load 到 port_A 之前，要先 load 到 W，所以是 ps = 10 -> s 加到 10 -> w 得到值 ; ps = 11 時在讓 port_A 去對 W 做 load，所以會在 n=12 時取得 w 的值。同理 port_B;

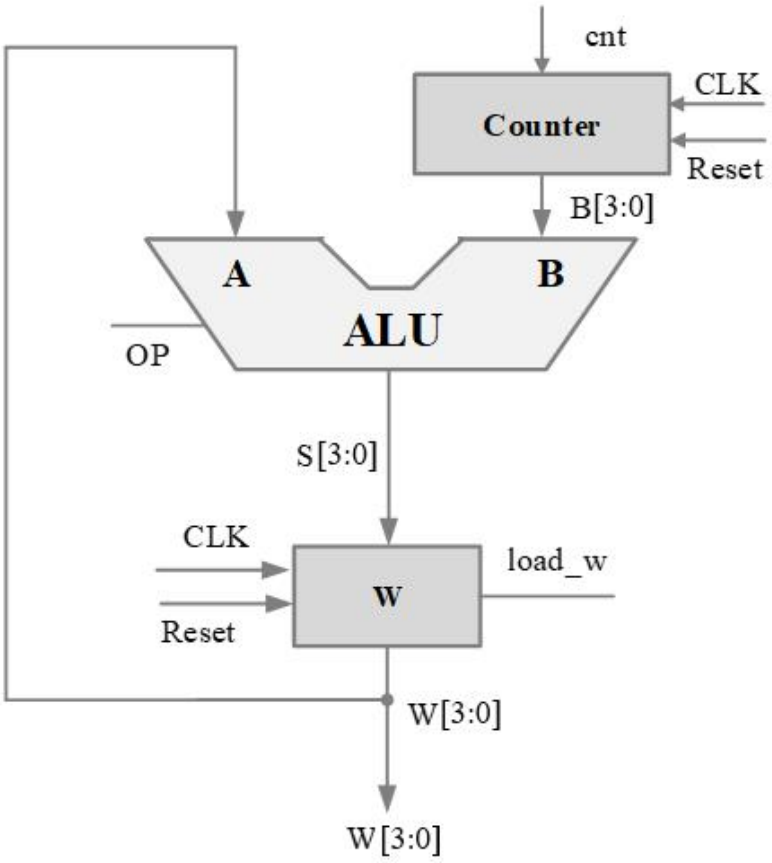
二、 Counter+ALU+register

● 實驗說明：

1. 設計一電路，此電路由 counter、ALU、register 組成，Counter 由 0 數到 10
   S = 0 + 1 | 2 * 3 - 4 / 5 + 6 + 7 ^ 8 + 9 & 10，依序計算，並將計算結果 S 依序存入 W 暫存器
2. 輸入：clk, reset
3. 輸出：W[3:0]

op[2:0]	ALU 運算	註解
3'b000	S = A + B	相加
3'b001	S = A - B	相減
3'b010	S = A * B	相乘
3'b011	S = A / B	除法
3'b100	S = A & B	AND
3'b101	S = A | B	OR
3'b110	S = A ^ B	XOR

● 系統硬體架構方塊圖（接線圖）：



- **系統架構程式碼、測試資料程式碼與程式碼說明(.sv 檔及.do 檔都要截圖)**

  截圖請善用 win+shift+S

  - hw_1017_2.sv

```
1  `timescale 1ns/10ps
2  module hw_1017_2(
3      input clk,                      //時脈
4      input reset,                    //reset
5      output logic [3:0] W     //output
6  );
7
8      logic load_w;             //load_w控制線
9      logic [3:0] ps,ns;        //現在狀態 下一個狀態
10     logic [3:0] A;            // A = W
11     logic [3:0] B;            //計數器，中間的值
12     logic [3:0] S;            // S = A+B
13     logic [2:0] op;           //控制ALU的模式
14     logic cnt;
15
16     always_ff @(posedge clk)        //fsm
17     begin
18        if(reset)
19           ps <= #1 0;
20        else
21           ps <= #1 ns;
22     end
24     always_ff @(posedge clk)         //W載入器
25     begin
26        if(reset)
27           W  <= #1 0;
28        else if(load_w)
29           W  <= #1 S;
30     end
31
32     always_ff @(posedge clk)         //counter
33     begin
34        if(reset)
35           B  <= #1 0;
36        else if(cnt)
37           B  <= #1 B + 1;
38     end
39
40     assign A = W;                   //將W拉到ALU當運算元，取叫A
42     always_comb                      //ALU
43     begin
44        case(op)
45           0:
46              S = A + B;
47           1:
48              S = A - B;
49           2:
50              S = A * B;
51           3:
52              S = A / B;
53           4:
54              S = A & B;
55           5:
56              S = A | B;
57           6:
58              S = A ^ B;
59        endcase
60     end
```

```
62      always_comb                     //狀態切換
63      begin
64      op = 0;                         //預設+
65      ns = 0;                         //次態
66      load_w = 1;                     //always load
67      cnt = 1;                        //預設要加計數器
68      case(ps)
69            0:
70            begin
71               op = 0;
72               ns = 1;
73            end
74
75            1:
76            begin
77               op = 0;
78               ns = 2;
79            end
80
81            2:
82            begin
83               op = 5;
84               ns = 3;
85            end
87             3:
88             begin
89                op = 2;
90                ns = 4;
91             end
92
93             4:
94             begin
95                op = 1;
96                ns = 5;
97             end
98
99             5:
100            begin
101               op = 3;
102               ns = 6;
103            end
104
105            6:
106            begin
107               op = 0;
108               ns = 7;
109            end
111           7:
112           begin
113              op = 0;
114              ns = 8;
115           end
116
117           8:
118           begin
119              op = 6;
120              ns = 9;
121           end
122
123           9:
124           begin
125              op = 0;
126              ns = 10;
127           end
128
129           10:
130           begin
131              op = 4;
132              ns = 11;
133           end
135           11:             //stop state
136           begin
137              cnt = 0;
138              ns = 11;
139              load_w = 0;
140              op = 7;
141           end
142        endcase
143     end
144  endmodule
```

- testbench.sv

simulation > tb > testbench.sv

```
1  `timescale 1ns/10ps
2  module testbench;
3
4      logic reset;                    //重置
5      logic clk;                      //時脈
6      logic [7:0] W;                  //輸出
7
8      hw_1017_2 hw_1017_21(
9          .reset(reset), //()內的變數為tb的變數，"."後面為hw_1017_2.sv的變數，將2者對應起來
10         .clk(clk),
11         .W(W)
12     );
13
14     always #10 clk = ~clk;
15
16     initial begin
17         reset = 1;clk = 0; //一開始先reset，將時脈歸0
18         #15 reset = 0;
19         #1000 $stop;
20     end
21 endmodule
```

- sim.do

simulation > modelsim > sim.do

```
1  vsim -voptargs=+acc work.testbench
2  view structure wave signals
3
4  do wave.do
5
6  log -r *
7  run -all
```

- Wave.do

simulation > modelsim > wave.do

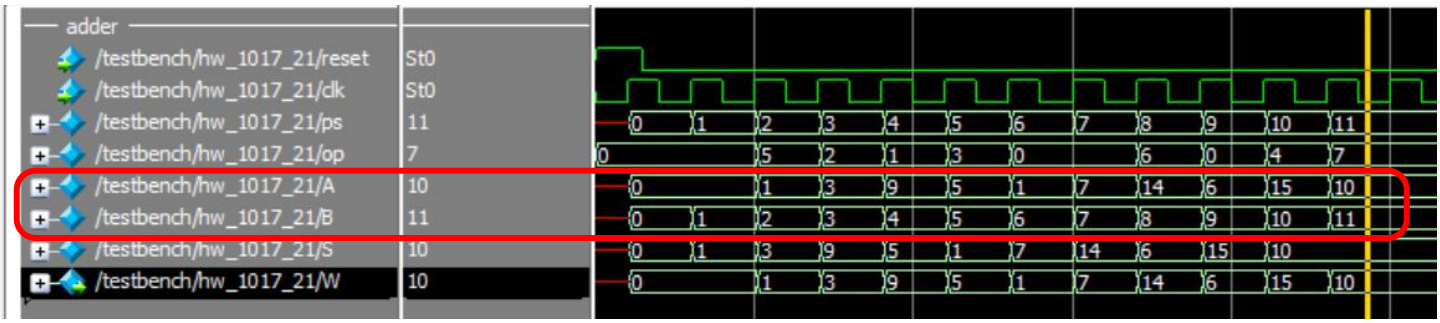
```
1  onerror {resume}
2  quietly WaveActivateNextPane {} 0
3
4  #add wave -noupdate -divider {TOP LEVEL INPUTS}
5
6  #add wave -noupdate -format Logic /testbench/clk
7  #add wave -noupdate -format Logic /testbench/rst
8
9
10
11 add wave -noupdate -divider {adder}
12
13 add wave -noupdate -format logic    /testbench/hw_1017_21/reset
14 add wave -noupdate -format logic    /testbench/hw_1017_21/clk
15 add wave -noupdate -format Literal -radix Unsigned      /testbench/hw_1017_21/ps
16 add wave -noupdate -format Literal -radix Unsigned      /testbench/hw_1017_21/op
17 add wave -noupdate -format Literal -radix Unsigned      /testbench/hw_1017_21/A
18 add wave -noupdate -format Literal -radix Unsigned      /testbench/hw_1017_21/B
19 add wave -noupdate -format Literal -radix Unsigned      /testbench/hw_1017_21/S
20 add wave -noupdate -format Literal -radix Unsigned      /testbench/hw_1017_21/W
```

- compile.do

simulation > modelsim > compile.do

```
1  #vlib work
2
3
4
5  # ----------------------------------------------------------
6  vlog ../tb/testbench.sv
7  vlog ../../design/hw_1017_2.sv
8
```

● 模擬結果與結果說明：



將 A 和 B 根據 op 做運算，存入 S。W 會載入上一個 S 的值。

A 是當前 W 的值。

● 結論與心得：

因為上禮拜停課，所以有些東西都忘光光了，不過好在老師有先複習一點，才慢慢地回憶起來。經過了這一堂課，我對 FSM 又更了解了，我發現現在最大的問題不是寫程式，而是幫程式命名……之前幾堂課像是 ALU,REG 呀，就是一個東西，現在都是做 combine，如果取叫 alu_reg 就會越來越長，所幸我現在決定以後名字都是 hw_上課日期(mmdd)_編號(num)這樣子....