邏輯系統實驗

Lab 11　分析 Testbench

2022/05/24

第 5 組	
組員姓名	學號
林珮玉	E24084096
廖本恩	E24102179
蘇冠誠	E24084143

基礎題：

1. 簡述題目：

選擇一堂之前實驗課中所有實驗的 testbench 來做報告，分析 testbench 的程式碼邏輯、意義(越詳細越高分)

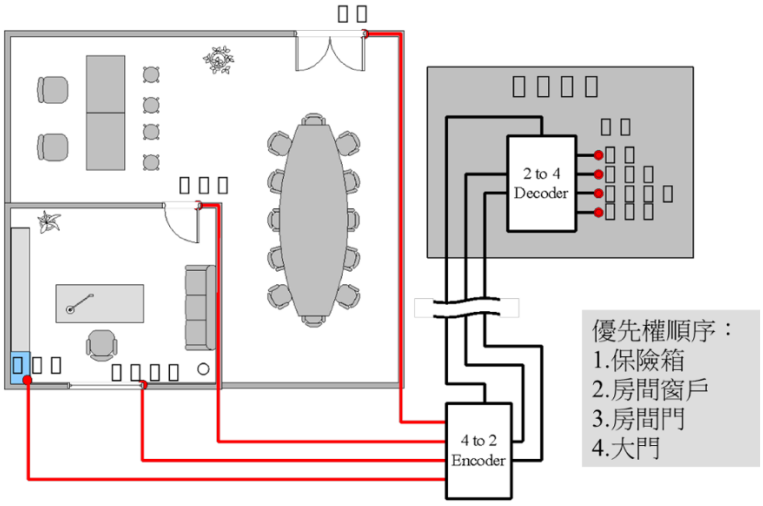
2. 實現方式：

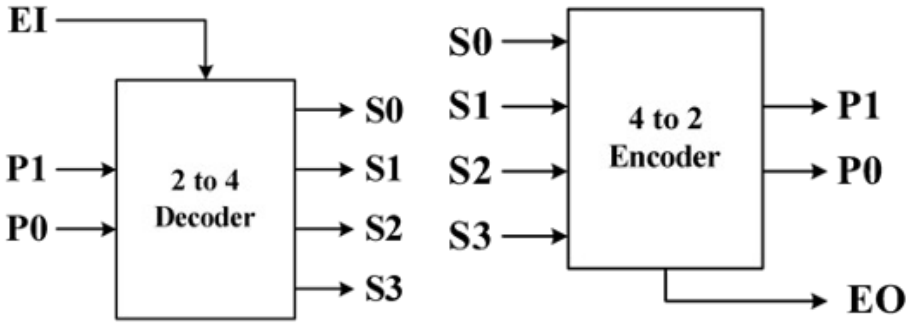
我們選擇 Lab4 的實驗來分析

tb1-1.v

簡述題目：保全系統

利用 Verilog 的 Gate Level 來描述保全系統電路並驗證






```
module tb ();
    // 設定保全門口所需的輸入與輸出之變數
    reg s0,s1,s2,s3;
    wire e,f,g,h;
```

```
    problem1_1 pb1 (
        .s0(s0),
        .s1(s1),
        .s2(s2),
        .s3(s3),
        .e(e),
        .f(f),
        .g(g),
        .h(h)
    );
    // Initialization
    // 利用行為層次語法，設定各個初始區塊
    initial
    begin
        $display("----------------------");
        $display("-- Simulation Start --");
        $display("----------------------");
    end

    //首先，利用 $fopen 打開已經寫好的 golden data，並將檔案讀入
    integer fd;
    initial
    begin
        fd = $fopen("golden.data","r");
    end
    integer i; //arr idx
    integer fail; // count fail times
    integer count; // total iterations
    reg ge,gf,gg,gh; // golden
    initial
    begin
    count = 0;
    fail = 0;
    // 設定 mux 的 input variables
    for( i = 0;i < 16;i = i+1) begin
        s0 = i[0];
        s1 = i[1];
        s2 = i[2];
```

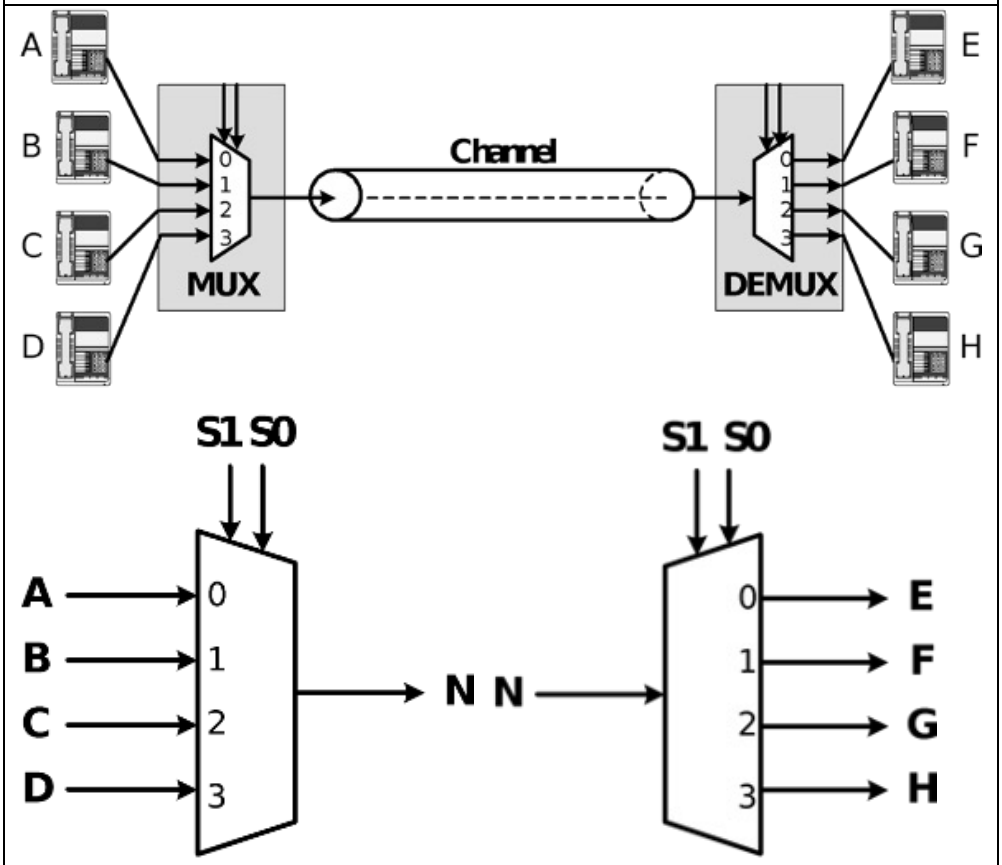
```
        s3 = i[3];
        count = count +1;
        // 利用 $fscanf 讀取四行 golden data，並各別存到四個變數中
        $fscanf(fd,"%d",ge);
        $fscanf(fd,"%d",gf);
        $fscanf(fd,"%d",gg);
        $fscanf(fd,"%d",gh);
        #10; // 讀完一排 golden data 後，延遲十秒，再做判斷
        // 若如果 file 讀進來的數值與 wire 不同，則 fail 就加一，並顯示錯誤。
        if((e !== ge) || (f !== gf) || ( g !== gg) || (h !== gh)) begin
          fail = fail + 1;
          $display("Case %d:\n\tInput s0,s1,s2,s3 : %d,%d,%d,%d\n\tExpect
e,f,g,h : %d,%d,%d,%d\n\tOutput e,f,g,h :
%d,%d,%d,%d\n",count,s0,s1,s2,s3,ge,gf,gg,gh,e,f,g,h);
        end
    end
        // 如果都沒有錯誤，則顯示通過，否則顯示失敗
        if (fail === 0)
        begin
            $display("\n");
            $display("\n");
            $display("      ***************************   _._    _,-'\"\"`-._      ");
            $display("      ** Congratulations !!   **  (,-.`._,'(      |\\\`-/|    ");
            $display("      ** Simulation1 PASS!!   **      `-.-' \\ )-`( , o o)    ");
            $display("      ***************************          `-   \\\`_`\"'-   ");
            $display("\n");
        end
        else
        begin
            $display("\n");
            $display("\n");
            $display("      ***************************       |\\     _,,,---,,_    ");
            $display("      ** OOPS!!             **  ZZZzz /,`.-'`'    -.  ;-;;,_ ");
            $display("      ** Simulation1 Failed!! **      |,4-  ) )-,_. ,\\ (  `'-'");
            $display("      ***************************     '---''(_/--'  `-'\\_)    ");
            $display("       Totally has %d errors                ", fail);
            $display("\n");
        end
```

```
        // 完成後利用 $fclose 關閉檔案，並加上 $finish 結束模擬運算。
        $fclose(fd);
        $finish;
    end
endmodule
```

tb1-2.v

簡述題目：接線生

利用 Verilog 的 Gate Level 來描述保全系統電路並驗證




```
module tb ();
    // 設定多工器與解多工器所需的變數
    // Register
    reg s0,s1,s2,s3,a,b,c,d;
    wire e,f,g,h;
    problem1_2 pb2 (
        .mux_s0(s0),
        .mux_s1(s1),
        .demux_s0(s2),
        .demux_s1(s3),
```

```
        .a(a),
        .b(b),
        .c(c),
        .d(d),
        .e(e),
        .f(f),
        .g(g),
        .h(h)
    );
    // Initialization
    initial
    begin
        $display("----------------------");
        $display("-- Simulation Start --");
        $display("----------------------");
    end
    //READ file
    // 利用 $fscanf 讀取四行 golden data，並各別存到四個變數中
    integer fd;
    initial
    begin
        fd = $fopen("golden.data","r");
    end
    integer i,sel; //arr idx
    integer fail; // count fail times
    integer count; // total iterations
    reg ge,gf,gg,gh; // golden
    initial
    begin
    count = 0;
    fail = 0;
    // 設定 mux 的 input variables
    for( i = 0;i < 16;i = i+1) begin
        a = i[0];
        b = i[1];
        c = i[2];
        d = i[3];
        for (sel = 0 ;sel < 16 ;sel = sel + 1) begin
```

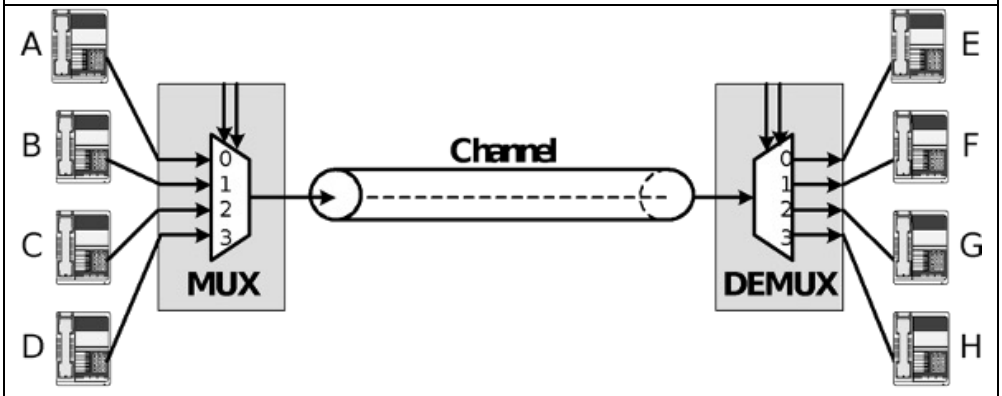
```
            // 設定 demux 的 input variables
            s0 = sel[0];
            s1 = sel[1];
            s2 = sel[2];
            s3 = sel[3];
            count = count + 1;
            $fscanf(fd,"%d %d %d %d",ge,gf,gg,gh);
            #10;  // 讀完一排 golden data 後，延遲十秒，再做判斷
            // 若 file 讀進來的數值與 wire 不同，則 fail 就加一，並顯示錯誤。
            if((e !== ge) || (f !== gf) || ( g !== gg) || (h !==
gh)) begin
                $display("Case %d :\n\t Input a,b,c,d :
%d,%d,%d,%d\n\tMux s1,s0 : %d,%d\n\tDemux s1,s0 : %d,%d\n\tExpect
e,f,g,h : %d,%d,%d,%d\n\tOutput e,f,g,h :
%d,%d,%d,%d\n",count,a,b,c,d,s1,s0,s3,s2,ge,gf,gg,gh,e,f,g,h);
                fail = fail + 1;
            end
        end
    end
        // 如果都沒有錯誤，則顯示通過，否則顯示失敗
        if (fail === 0)
        begin
            $display("\n");
            $display("\n");
            $display("      ***************************   _.-    _,-'\"\"`-._      ");
            $display("      **  Congratulations !!   **  (,-.`._,'(      |\\\`-/|    ");
            $display("      **  Simulation1 PASS!!   **      `-.-' \\ )-`( , o o)    ");
            $display("      ***************************           `-   \\\`_`\"'-   ");
            $display("\n");
        end
        else
        begin
            $display("\n");
            $display("\n");
            $display("      ***************************       |\\     _,,,---,,_    ");
            $display("      **  OOPS!!             **  ZZZzz /,`.-'`'    -.  ;-;;,_  ");
            $display("      **  Simulation1 Failed!! **      |,4-  ) )-,_. ,\\ (  `'-'");
            $display("      ***************************     '---''(_/--'  `-'\\_)    ");
```

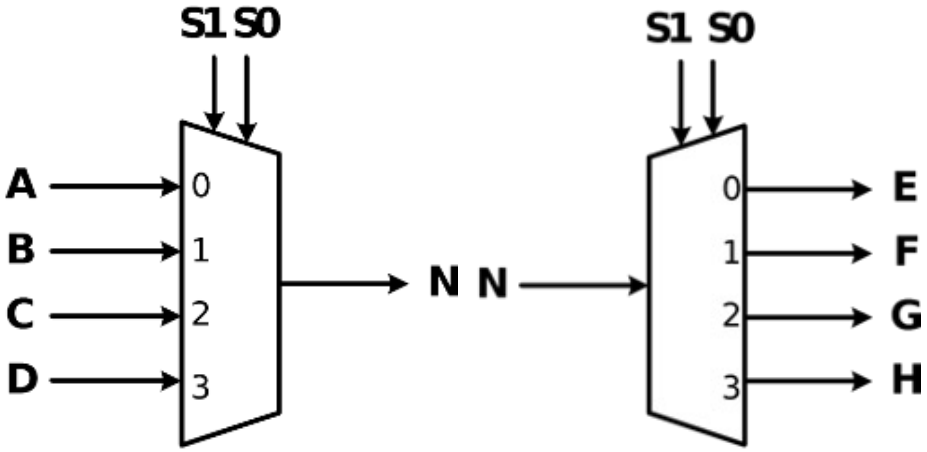
```
            $display("       Totally has %d errors                ", fail);
            $display("\n");
        end
        // 完成後利用 $fclose 關閉檔案，並加上 $finish 結束模擬運算。
        $fclose(fd);
        $finish;
    end
endmodule
```

tb1-2.v

簡述題目：接線生

利用 Verilog 的 Gate Level 來描述保全系統電路並驗證






```
module tb ();
    // 設定多工器與解多工器所需的變數
    // Register
    reg s0,s1,s2,s3,a,b,c,d;
    wire e,f,g,h;
    problem1_2 pb2 (
        .mux_s0(s0),
        .mux_s1(s1),
        .demux_s0(s2),
```

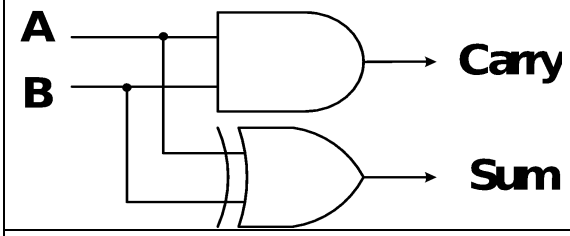
```
        .demux_s1(s3),
        .a(a),
        .b(b),
        .c(c),
        .d(d),
        .e(e),
        .f(f),
        .g(g),
        .h(h)
    );
    // Initialization
    initial
    begin
        $display("----------------------");
        $display("-- Simulation Start --");
        $display("----------------------");
    end
    //READ file
    //首先，利用 $fopen 打開已經寫好的 golden data，並將檔案讀入
    integer fd;
    initial
    begin
        fd = $fopen("golden.data","r");
    end
    integer i,sel; //arr idx
    integer fail; // count fail times
    integer count; // total iterations
    reg ge,gf,gg,gh; // golden
    initial
    begin
    count = 0;
    fail = 0;
    // 設定 mux 的 input variables
    for( i = 0;i < 16;i = i+1) begin
        a = i[0];
        b = i[1];
        c = i[2];
        d = i[3];
```

```
        for (sel = 0 ;sel < 16 ;sel = sel + 1) begin
            // 設定 demux 的 input variables
            s0 = sel[0];
            s1 = sel[1];
            s2 = sel[2];
            s3 = sel[3];
            count = count + 1;
            // 利用 $fscanf 讀取四行 golden data，並各別存到四個變數中
            $fscanf(fd,"%d %d %d %d",ge,gf,gg,gh);
            #10;  // 讀完一排 golden data 後，延遲十秒，再做判斷
            // 若 file 讀進來的數值與 wire 不同，則 fail 就加一，並顯示錯誤。
            if((e !== ge) || (f !== gf) || ( g !== gg) || (h !==
gh)) begin
                $display("Case %d :\n\t Input a,b,c,d :
%d,%d,%d,%d\n\tMux s1,s0 : %d,%d\n\tDemux s1,s0 : %d,%d\n\tExpect
e,f,g,h : %d,%d,%d,%d\n\tOutput e,f,g,h :
%d,%d,%d,%d\n",count,a,b,c,d,s1,s0,s3,s2,ge,gf,gg,gh,e,f,g,h);
                fail = fail + 1;
            end
        end
    end
    // 如果都沒有錯誤，則顯示通過，否則顯示失敗
        if (fail === 0)
        begin
            $display("\n");
            $display("\n");
            $display("      ***************************   _._    _,-'\"\"`-._      ");
            $display("      ** Congratulations !!   **  (,-.`._,'(      |\\\`-/|    ");
            $display("      ** Simulation1 PASS!!   **      `-.-' \\ )-`( , o o)    ");
            $display("      ***************************          `-   \\\`_`\"'-   ");
            $display("\n");
        end
        else
        begin
            $display("\n");
            $display("\n");
            $display("      ***************************       |\\     _,,,---,,_    ");
            $display("      ** OOPS!!            **  ZZZzz /,`.-'`'    -.  ;-;;,_ ");
```

```
            $display("      ** Simulation1 Failed!! **      |,4-  ) )-,_. ,\\ (  `'-'");
            $display("      ***************************     '---''(_/--'  `-'\\_)    ");
            $display("       Totally has %d errors                ", fail);
            $display("\n");
        end
        // 完成後利用 $fclose 關閉檔案，並加上 $finish 結束模擬運算。
        $fclose(fd);
        $finish;
    end
endmodule
```

tb2-1.v

簡述題目：半加器



A	B	Carry	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

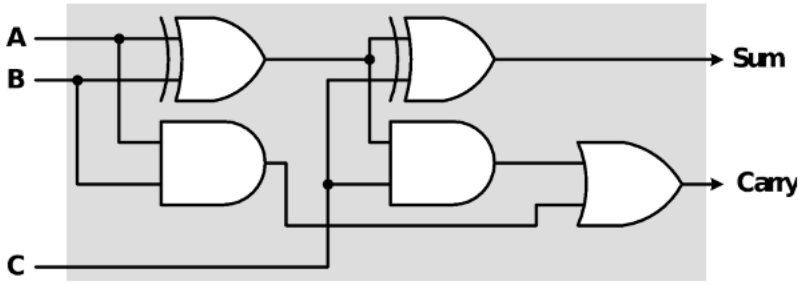
```
module tb ();
    // 設定半加器輸入與輸出所需的變數
    // Register
    reg a,b;
    wire s,c;
    problem2_1 pb3 (
        .a(a),
        .b(b),
        .c(c),
        .s(s)
    );
    // Initialization
    // 利用行為層次語法，設定各個初始區塊
    initial
    begin
        $display("----------------------");
        $display("-- Simulation Start --");
        $display("----------------------");
    end
    //READ file
```

```
    //利用 $fopen 打開我們自行編寫的 golden data，並將檔案讀入
    integer fd;
    initial
    begin
        fd = $fopen("golden.data","r");
    end
    integer i;
    integer fail;
    integer count;
    reg gc,gs;
    initial
    begin
        count = 0;
        fail = 0;
        // 設定 半加器 的 input variables
        for (i = 0; i < 4; i = i + 1) begin
            a = i[0];
            b = i[1];
            // 利用 $fscanf 讀取兩行 golden data，並各別存到兩個變數中
            $fscanf(fd,"%d %d",gc,gs);
            count = count + 1;
            #10; // 讀完一排 golden data 後，延遲十秒，再做判斷
            // 若 file 讀進來的數值與 wire 不同，則 fail 就加一，並顯示錯誤。
            if((gc !== c) || (gs !== s))begin
                fail = fail + 1;
                $display("Case %d :\n\tInput a,b : %d,%d\n\tExpect
c,s : %d,%d\n\tOutput c,s : %d,%d\n",count,a,b,gc,gs,c,s);
            end
        end
        // 如果都沒有錯誤，則顯示通過，否則顯示失敗
        if (fail === 0)
        begin
            $display("\n");
            $display("\n");
            $display("      **************************   _._    _,-'\"\"`-._      ");
            $display("      ** Congratulations !!   **  (,-.`._,'(      |\\\`-/|    ");
            $display("      ** Simulation1 PASS!!   **      `-.-' \\ )-`( , o o)    ");
            $display("      **************************          `-   \\\`_`\"'-   ");
```

```
            $display("\n");
        end
        else
        begin
            $display("\n");
            $display("\n");
            $display("      ****************************       |\\     _,,,---,,_    ");
            $display("      **  OOPS!!             **  ZZZzz /,`.-'`'    -.  ;-;;,_ ");
            $display("      **  Simulation1 Failed!! **      |,4-  ) )-,_. ,\\ (  `'-'");
            $display("      ****************************     '---''(_/--'  `-'\\_)    ");
            $display("       Totally has %d errors                ", fail);
            $display("\n");
        end
        // 完成後利用 $fclose 關閉檔案，並加上 $finish 結束模擬運算。
        $fclose(fd);
        $finish;
    end
endmodule
```

tb2-2.v

簡述題目：全加器



A	B	C	Carry	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

```
module tb ();
    // 設定全加器輸入與輸出所需的變數
```

```
    reg a,b,c;
    wire s,carry;
    problem2_2 pb4 (
        .a(a),
        .b(b),
        .c(c),
        .s(s),
        .carry(carry)
    );
    // Initialization
    initial
    begin
        $display("----------------------");
        $display("-- Simulation Start --");
        $display("----------------------");
    end
    //READ file
    integer fd;
    initial
    begin
        fd = $fopen("golden.data","r");
    end
    integer i;
    integer fail;
    integer count;
    reg gc,gs;
    initial
    begin
        count = 0;
        fail = 0;
        // 設定 全加器 的 input variables
        for (i = 0; i < 8; i = i + 1) begin
            a = i[0];
            b = i[1];
            c = i[2];
            // 利用 $fscanf 讀取兩行 golden data，並各別存到兩個變數中
            $fscanf(fd,"%d %d",gc,gs);
            count = count + 1;
```

```
            #10; // 讀完一排 golden data 後，延遲十秒，再做判斷
            // 若 file 讀進來的數值與 wire 不同，則 fail 就加一，並顯示錯誤。
            if((gc !== carry) || (gs !== s))begin
                fail = fail + 1;
                $display("Case %d :\n\tInput a,b,c :
%d,%d,%d\n\tExpect c,s : %d,%d\n\tOutput c,s :
%d,%d\n",count,a,b,c,gc,gs,carry,s);
            end
        end
        // 如果都沒有錯誤，則顯示通過，否則顯示失敗
        if (fail === 0)
        begin
            $display("\n");
            $display("\n");
            $display("      ***************************   _._    _,-'\"\"`-._      ");
            $display("      ** Congratulations !!   **  (,-.`._,'(      |\\\`-/|    ");
            $display("      ** Simulation1 PASS!!   **      `-.-' \\ )-`( , o o)    ");
            $display("      ***************************          `-   \\\`_`\"'-   ");
            $display("\n");
        end
        else
        begin
            $display("\n");
            $display("\n");
            $display("      ***************************       |\\     _,,,---,,_    ");
            $display("      ** OOPS!!             **  ZZZzz /,`.-'`'    -.  ;-;;,_ ");
            $display("      ** Simulation1 Failed!! **      |,4-  ) )-,_. ,\\ (  `'-'");
            $display("      ***************************     '---''(_/--'  `-'\\_)    ");
            $display("       Totally has %d errors                ", fail);
            $display("\n");
        end
        // 完成後利用 $fclose 關閉檔案，並加上 $finish 結束模擬運算。
        $fclose(fd);
        $finish;
    end
endmodule
```

心得

組員一 林珮玉 E24084096

這次實驗因為疫情的關係改為線上，原本的 FPGA 進階課程也取消了實作的機會，有一點點可惜，不過我覺得助教這次的實驗作業出的很棒，讓我們更熟悉 Verilog 語法以及 testbench、golden data 和模擬的關係，透過一行一行逐句頗析，讓我更了解 testbench 在做什麼，也發現其實沒有想像中複雜！如果有機會的話，也希望可以自己寫寫看 testbench。

組員二 廖本恩 E24102179

在用了一學期的 testbench 後，終於有機會學習裡面的內容。相對於其他程式語言，要經過 testbench 的驗證應該是 verilog 最特別的一點，以後自己寫其他專案時也會需要寫出 testbench，所以能夠先分析助教的寫法對以後一定是十分有幫助。

組員三 蘇冠誠 E24084143

testbench(tb)這東西過去我有寫過，所以我明白它受不受人歡迎是來自於在它上面耗費的心思，我當時寫 tb 只是簡易地檢查 verilog 的程式有沒有問題而已。換個方式思考，tb 即是寫 verilog 而言不可忽略的部分。