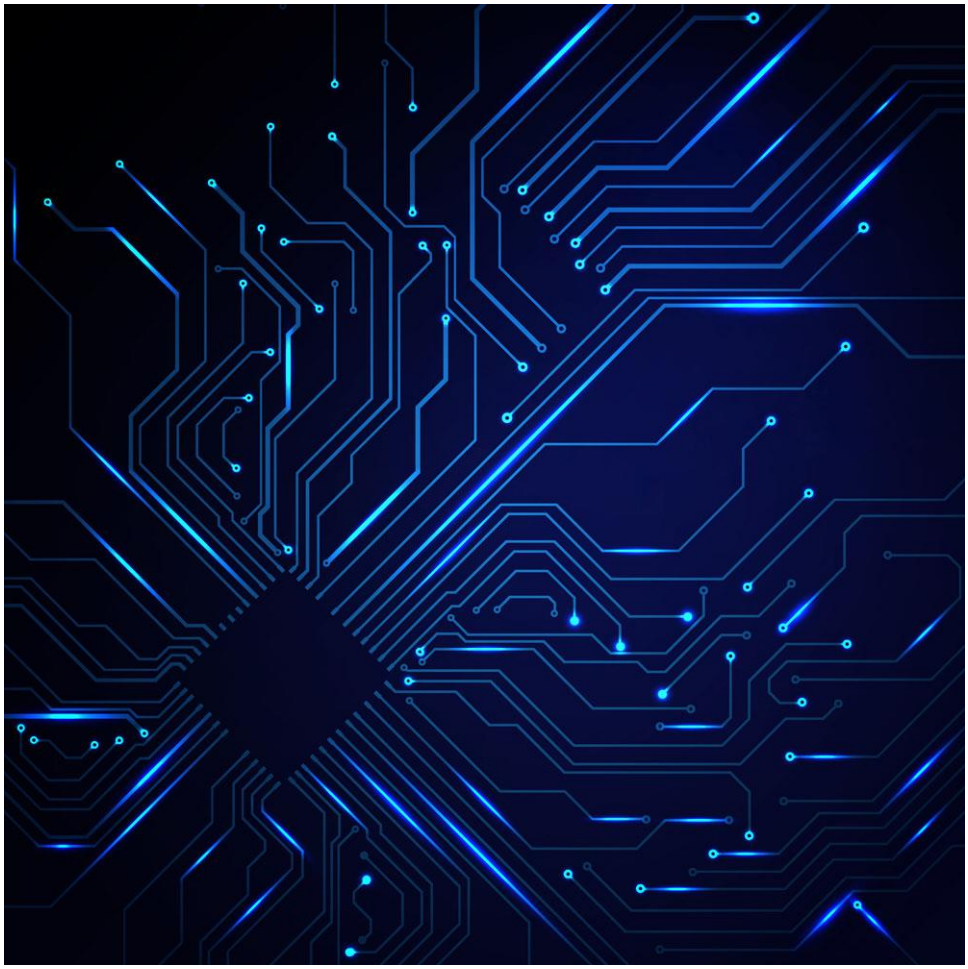


2024/XX/XX

實驗五

姓名：黃文祺　　學號：01057013

班級：資工 3A

E-mail：OOOOOOOO

注意

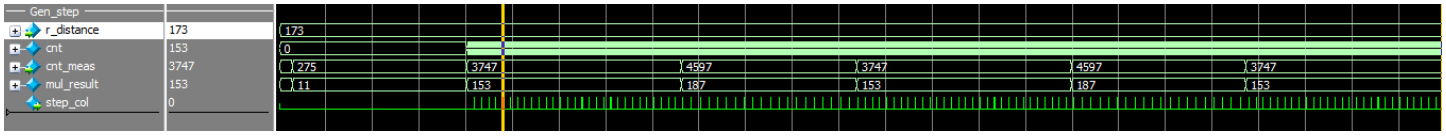
1. 一律將此檔轉成 PDF 檔繳交
2. 繳交期限為
3. 隔週三上午九點
4. 一人繳交一份
5. 檔名：學號_HW?.pdf
6. 檔名請按照作業檔名格式進行填寫
7. 未依照格式不予批改

一、 Encoder_2

- **實驗說明:**
- **系統架構程式碼與程式碼說明**

1. 以 50MHz 當作系統時脈，在 600dpi 下量測 1.7μm。
2. 輸入信號：enc，為待測信號。
3. 輸出數值：cnt_meas、mul_result、step_col。

PS. 運用 LPF,Edge detector, counter, FSM 電路設計上述電路



- **系統架構程式碼與程式碼說明**

Low_Pass_Filter_4ENC.sv

這個是使用參考檔案的，跟 testbench_DEMO 一樣用參考檔案

testbench_DEMO.sv

C:\Users\user\Desktop\HW5\signal\simulation\tb\testbench_DEMO.sv - Notepad++

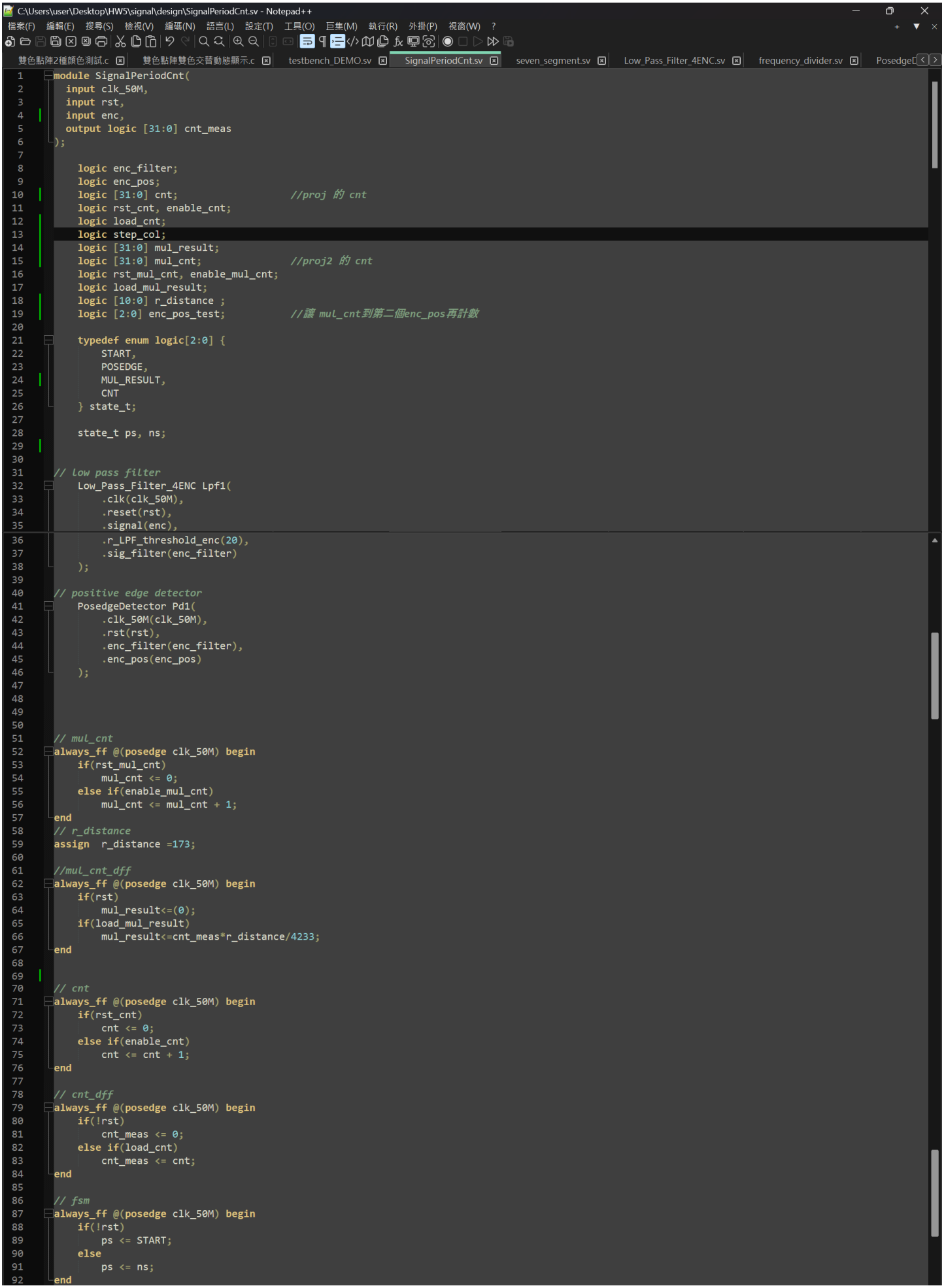
檔案(F) 編輯(E) 搜尋(S) 檢視(V) 編碼(N) 語言(L) 設定(T) 工具(O) 巨集(M) 執行(R) 外掛(P) 視窗(W) ?

雙色點陣2種顏色測試.c | 雙色點陣雙色交替動態顯示.c | testbench_DEMO.sv | SignalPeriodCnt.sv | seven_segment.sv | Low_Pass_Filter_4ENC.sv | frequency_divider.sv | Posedge[

```
1   `timescale 1ns/100ps
2   module testbench_DEMO;
3
4
5
6
7       logic [7:0]w_q;
8       logic reset;
9       logic clk;
10      integer i;
11      logic enc;
12
13      logic [6:0]     HEX0;
14      logic [6:0]     HEX1;
15      logic [6:0]     HEX2;
16      logic [6:0]     HEX3;
17      logic [6:0]     HEX4;
18      logic [6:0]     HEX5;
19      logic [3:0]     KEY;
20      logic [9:0]     SW;
21      logic [9:0]     LEDR;
22
23      logic CLOCK_50;
24      logic rst_n;
25      logic d;
26
27  DE0_CV de0_cv1(
28
29      //////////// CLOCK //////////
30      .CLOCK_50   (CLOCK_50),
31      .CLOCK2_50  (),
32      .CLOCK3_50  (),
33      .CLOCK4_50  (),
34
35      //////////// SEG7 //////////
```

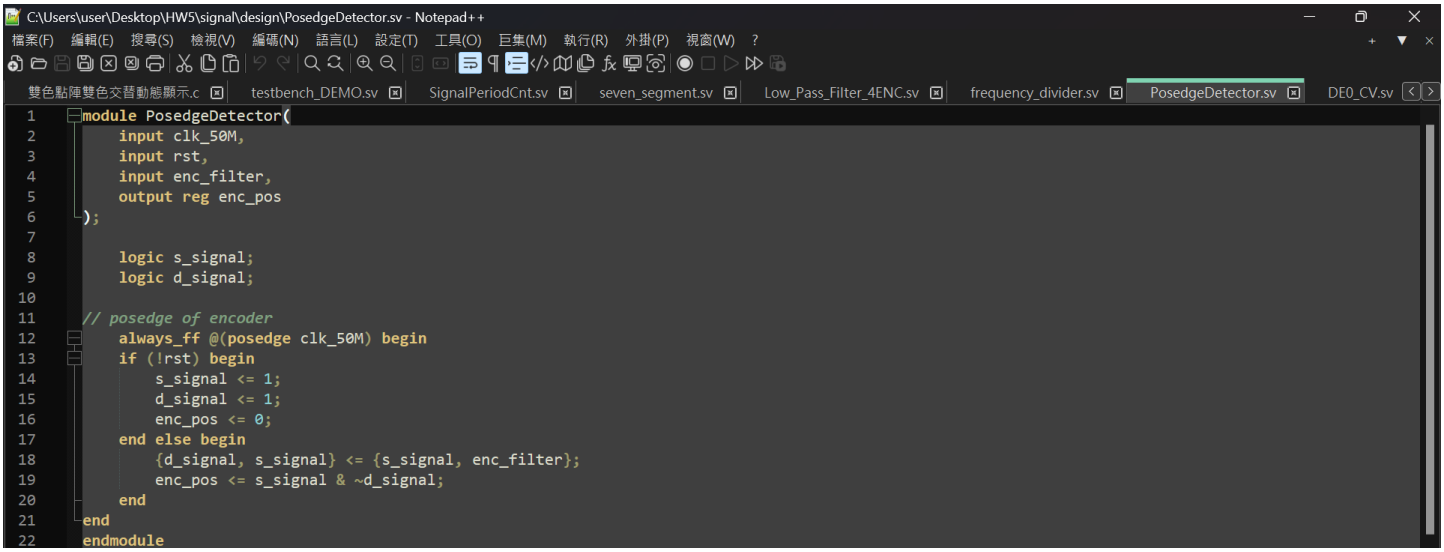
```
    .HEX0(),
    .HEX1(),
    .HEX2(),
    .HEX3(),
    .HEX4(),
    .HEX5(),

    ////////////// KEY //////////
    .KEY        (KEY),
    .RESET_N    (rst_n),

    ////////////// LED //////////
    .LEDR       (LEDR),


    ////////////// SW //////////
    .SW         (SW)
);

    //assign KEY[0] = rst_n;


    task noise_gen; begin
            for (i=0; i<25; i=i+1) begin
                #100    enc = 0;    //noise
                #100    enc = 1;
            end
        end
    endtask



    always #10 CLOCK_50 = ~CLOCK_50;

    always begin

        noise_gen;


        #10000  enc = 0;    // signal

        noise_gen;

        #10000  enc = 1;    // signal

        noise_gen;

        #20000  enc = 0;    // signal
        #20000  enc = 1;
        //
    noise_gen;

        #10000  enc = 0;    // signal

        noise_gen;

        #15000  enc = 1;    // signal

        noise_gen;

        #25000  enc = 0;    // signal
        #27000  enc = 1;



    end

    assign SW[0] = enc;


    initial begin
        CLOCK_50 = 0;
        rst_n = 0;
        enc = 0;

        #20 rst_n = 1;;

        #500000 $stop;
    end


endmodule
```

SignalPeriodCnt.sv (延續之前作業名稱複製的，沒改名稱@@)



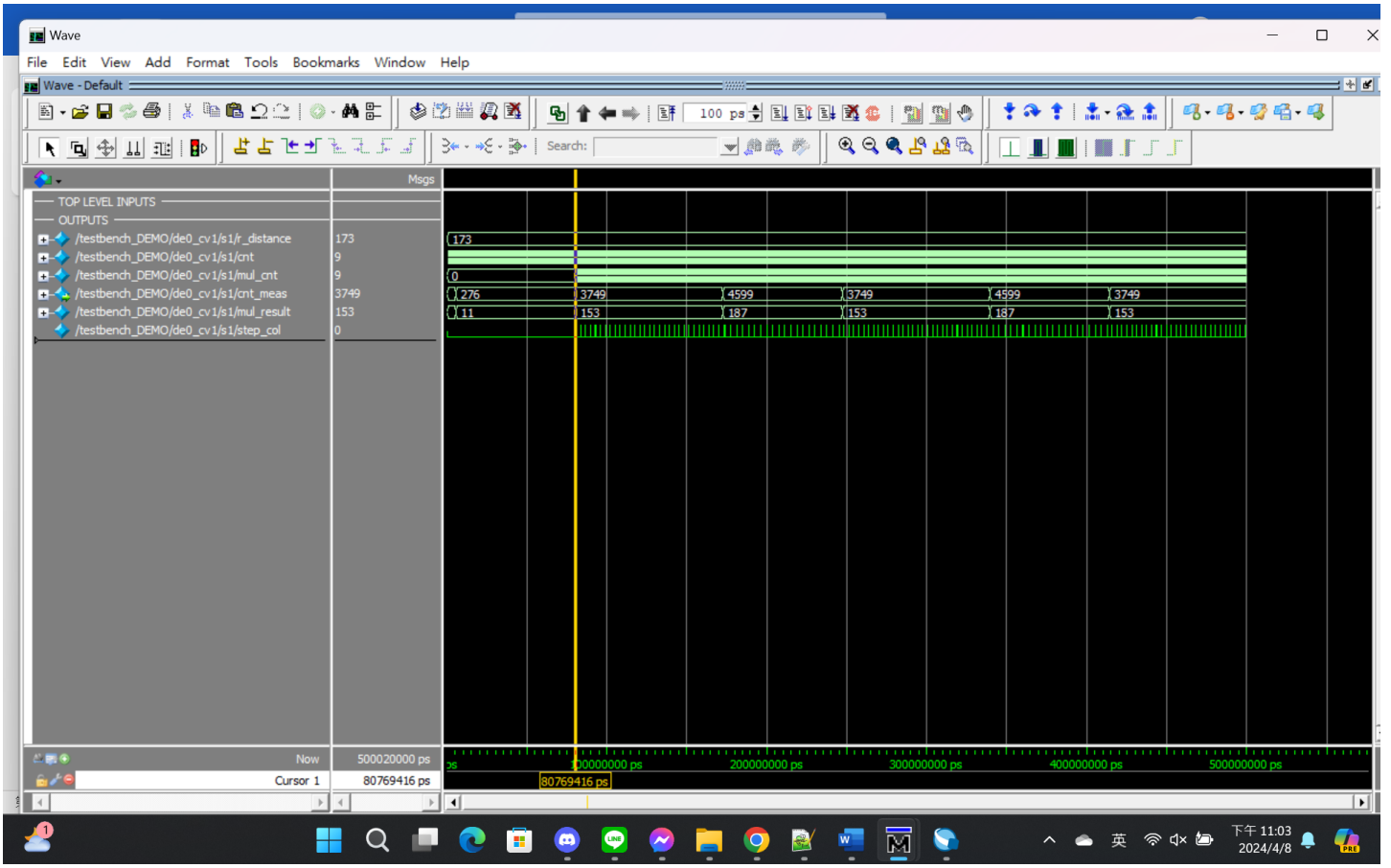
```
93
94  always_comb begin
95          rst_cnt         = 0;
96          load_cnt        = 0;
97          enable_cnt      = 0;
98          step_col        = 0;
99          rst_mul_cnt     = 0;
100         load_mul_result = 1;
101         enable_mul_cnt  = 0;
102
103         if(!rst)
104             enc_pos_test<=0;    //讓 mul_cnt到第二個enc_pos再計數
105
106         ns          = ps;
107
108     case(ps)
109         START: begin
110             rst_cnt     = 1;
111             rst_mul_cnt = 1;
112             ns  = CNT;
113         end
114
115         CNT: begin
116             if(enc_pos)
117                 ns  =   POSEDGE;
118
119             if (enc_pos_test!=2)    //讓 mul_cnt到第二個enc_pos再計數
120                 rst_mul_cnt=1;
121
122             if(mul_cnt>=mul_result & mul_cnt > 0 ) begin
123                 step_col    =1;
124                 rst_mul_cnt =1;
125             end
126
127             enable_cnt      =1;
128             enable_mul_cnt  =1;
129         end
130
131         POSEDGE: begin
132             load_cnt = 1;
133             rst_cnt  = 1;
134
135             enc_pos_test=enc_pos_test+1;    //------------------------------ //
136             if(enc_pos_test>1)              //讓 mul_cnt到第二個enc_pos再計數 //
137                 enc_pos_test=2;             //------------------------------//
138
139             ns       = CNT;
140         end
141
142     endcase
143 end
144 endmodule
145
```

PosedgeDetector.sv




```
1   module PosedgeDetector(
2       input clk_50M,
3       input rst,
4       input enc_filter,
5       output reg enc_pos
6   );
7
8       logic s_signal;
9       logic d_signal;
10
11  // posedge of encoder
12      always_ff @(posedge clk_50M) begin
13      if (!rst) begin
14          s_signal <= 1;
15          d_signal <= 1;
16          enc_pos <= 0;
17      end else begin
18          {d_signal, s_signal} <= {s_signal, enc_filter};
19          enc_pos <= s_signal & ~d_signal;
20      end
21  end
22  endmodule
```

■ 模擬結果與結果說明：



1. 這次作業的 cnt 是 mul_cnt，第一個 cnt 是 proj1 的 cnt

2. 黃線以前處理乾淨了(cnt 和 mul_cnt 比較)

SignalPeriodCnt 註解 //讓 mul_cnt 到第二個 enc_pos 再計數

■ 結論與心得：

Demo3 結束後其實數據就差不多了，但要注意的就是 cnt 不是作業 4 的 cnt 然後就是 mul_cnt 要稍微做一點改變，才能和作業 5 的模擬範例一樣。 owob