

Computation II: embedded system design (5EIB0)

[Dashboard](#) / [My courses](#) / [5EIB0](#) / [Final Exam 2024-04-12](#) / [Part 2e - 2024-04-12 Implement a counter with FSM trigger \(6pt\)](#)

Part 2e - 2024-04-12 Implement a counter with FSM trigger (6pt)

 [Description](#)

 [Submission view](#)

Grade

Reviewed on Monday, 15 April 2024, 3:58 PM by Automatic grade

Grade: 6.00 / 6.00

Assessment report [-]

[\[+\]](#)Summary of tests

 Submitted on Friday, 12 April 2024, 4:03 PM ( [Download](#))

timer.v

```
1 module timer(  
2     input clk,  
3     input reset,  
4     input enable,  
5     input [4:0] value,  
6     input valid,  
7     output trigger,  
8     output[4:0] count);  
9     reg [4:0] out;  
10    // reg [4:0] MUX1_next, MUX2_next;  
11    wire [4:0] MUX1, MUX2, out_next;  
12    wire complete;  
13  
14    timer_fsm timer_fsm_inst(  
15        .clk(clk),  
16        .reset(reset),  
17        .enable(enable),  
18        .complete(complete),  
19        .trigger(trigger));  
20  
21    assign count = out;  
22    assign complete = ~(|out);  
23  
24    assign MUX1 = (~complete & enable)?(out - 'd1):(out);  
25    assign MUX2 = (valid)?(value):(MUX1);  
26    assign out_next = (reset)?(5'h0):(MUX2);  
27  
28    always @(posedge clk) begin  
29        out <= out_next;  
30    end  
31 endmodule  
32
```

timer_fsm.v



```
1 module timer_fsm(  
2     input clk,  
3     input reset,  
4     input enable,  
5     input complete,  
6     output trigger);  
7  
8     reg [1:0] state, state_next;  
9     parameter IDLE = 'd0;  
10    parameter COUNTING = 'd1;  
11    parameter PAUSED = 'd2;  
12    parameter DONE = 'd3;  
13  
14    assign trigger = (state == DONE);  
15  
16    always @(posedge clk) begin  
17        if(reset) begin  
18            state <= IDLE;  
19        end else begin  
20            state <= state_next;  
21        end  
22    end  
23  
24    always @(*) begin  
25        state_next = state;  
26  
27        case(state)  
28            IDLE: begin  
29                if((enable==0) & (complete==0)) begin  
30                    state_next = PAUSED;  
31                end else if((enable==1) & (complete==0)) begin  
32                    state_next = COUNTING;  
33                end  
34            end  
35            COUNTING: begin  
36                if((enable==0) & (complete==1)) begin  
37                    state_next = IDLE;  
38                end else if((enable==1) & (complete==1)) begin  
39                    state_next = DONE;  
40                end else if((enable==0) & (complete==0)) begin  
41                    state_next = PAUSED;  
42                end  
43            end  
44            PAUSED: begin  
45                if((enable==1) & (complete==1)) begin  
46                    state_next = DONE;  
47                end else if((enable==1) & (complete==0)) begin  
48                    state_next = COUNTING;  
49                end else if((enable==0) & (complete==1)) begin  
50                    state_next = IDLE;  
51                end  
52            end  
53            DONE: begin  
54                if(complete==1) begin  
55                    state_next = IDLE;  
56                end else if((enable==1) & (complete==0)) begin  
57                    state_next = COUNTING;  
58                end else if((enable==0) & (complete==0)) begin  
59                    state_next = PAUSED;  
60                end  
61            end  
62        endcase  
63    end  
64 endmodule  
65
```

[VPL](#)

You are logged in as Thomas Stirling Valdez (Log out)

5EIB0

Data retention summary