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

```
module timer(
            input clk,
            input reset,
            input enable,
            input [4:0] value,
            input valid,
            output trigger,
            output[4:0] count);
    reg [4:0] out;
    // reg [4:0] MUX1_next, MUX2_next;
    wire [4:0] MUX1, MUX2, out_next;
    wire complete;

    timer_fsm timer_fsm_inst(
         .clk(clk),
         .reset(reset),
         .enable(enable),
         .complete(complete),
         .trigger(trigger));

    assign count = out;
    assign complete = ~(|out);

    assign MUX1 = (~complete & enable)?(out - 'd1):(out);
    assign MUX2 = (valid)?(value):(MUX1);
    assign out_next = (reset)?(5'h0):(MUX2);

    always @(posedge clk) begin
        out <= out_next;
    end
endmodule

```

timer_fsm.v

```verilog
module timer_fsm(
            input clk,
            input reset,
            input enable,
            input complete,
            output trigger);

    reg [1:0] state, state_next;
    parameter IDLE = 'd0;
    parameter COUNTING = 'd1;
    parameter PAUSED = 'd2;
    parameter DONE = 'd3;

    assign trigger = (state == DONE);

    always @(posedge clk) begin
        if(reset) begin
            state <= IDLE;
        end else begin
            state <= state_next;
        end
    end

    always @(*) begin
        state_next = state;

        case(state)
            IDLE: begin
                if((enable==0) & (complete==0)) begin
                    state_next = PAUSED;
                end else if((enable==1) & (complete==0)) begin
                    state_next = COUNTING;
                end
            end
            COUNTING: begin
                if((enable==0) & (complete==1)) begin
                    state_next = IDLE;
                end else if((enable==1) & (complete==1)) begin
                    state_next = DONE;
                end else if((enable==0) & (complete==0)) begin
                    state_next = PAUSED;
                end
            end
            PAUSED: begin
                if((enable==1) & (complete==1)) begin
                    state_next = DONE;
                end else if((enable==1) & (complete==0)) begin
                    state_next = COUNTING ;
                end else if((enable==0) & (complete==1)) begin
                    state_next = IDLE;
                end
            end
            DONE: begin
                if(complete==1) begin
                    state_next = IDLE;
                end else if((enable==1) & (complete==0)) begin
                    state_next = COUNTING;
                end else if((enable==0) & (complete==0)) begin
                    state_next = PAUSED;
                end
            end
        endcase
    end
endmodule

```

VPL