Computation II: embedded system design (5EIB0)

Part 2d - 2024-04-12 Finite State Machine (Implement FSM Moore UART Frame Odd Parity Detector - 6pt)

Grade

Reviewed on Monday, 15 April 2024, 4:06 PM by Automatic grade
Grade: 6.00 / 6.00

Assessment report [-]
[+]**Summary of tests**
Submitted on Friday, 12 April 2024, 3:04 PM (Download)

uart_parity_odd.v

```
module uart_parity_odd(
        input clk,
        input reset,
        input signal,
        output error,
        output valid);

    parameter BREAK = 'd0;
    parameter IDLE = 'd1;
    parameter START = 'd2;

    parameter BIT1_EVEN = 'd3;
    parameter BIT1_ODD = 'd4;
    parameter BIT2_EVEN = 'd5;
    parameter BIT2_ODD = 'd6;
    parameter BIT3_EVEN = 'd7;
    parameter BIT3_ODD = 'd8;
    parameter BIT4_EVEN = 'd9;
    parameter BIT4_ODD = 'd10;
    parameter PAR_EVEN = 'd11;
    parameter PAR_ODD = 'd12;
    parameter STP_EVEN = 'd13;
    parameter STP_ODD = 'd14;
    reg [3:0] state, state_next;

    assign valid = (state==STP_ODD);
    assign error = (state==STP_EVEN);

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

    always @(*) begin
        state_next = state;

        case(state)
            BREAK: begin
                if(signal == 1) begin
                    state_next = IDLE;
                end
            end
            IDLE: begin
                if(signal == 0) begin
                    state_next = START;
                end
            end
            START: state_next = (signal == 1)?(BIT1_ODD):(BIT1_EVEN);

            BIT1_EVEN: state_next = (signal == 1)?(BIT2_ODD):(BIT2_EVEN);
            BIT1_ODD: state_next = (signal == 1)?(BIT2_EVEN):(BIT2_ODD);

            BIT2_EVEN: state_next = (signal == 1)?(BIT3_ODD):(BIT3_EVEN);
            BIT2_ODD: state_next = (signal == 1)?(BIT3_EVEN):(BIT3_ODD);

            BIT3_EVEN: state_next = (signal == 1)?(BIT4_ODD):(BIT4_EVEN);
            BIT3_ODD: state_next = (signal == 1)?(BIT4_EVEN):(BIT4_ODD);

            BIT4_EVEN: state_next = (signal == 1)?(PAR_ODD):(PAR_EVEN);
            BIT4_ODD: state_next = (signal == 1)?(PAR_EVEN):(PAR_ODD);

            PAR_EVEN: state_next = (signal == 1)?(STP_EVEN):(BREAK);
            PAR_ODD: state_next = (signal == 1)?(STP_ODD):(BREAK);

            STP_EVEN: state_next = (signal == 1)?(IDLE):(START);
            STP_ODD: state_next = (signal == 1)?(IDLE):(START);
        endcase
    end
endmodule

```

VPL