Digital Design I

CS 223 - 01

Lab 05

Uğur Erdem Seyfi

21801744

09.12.2019

Trainer Pack No:

**1. Timer module implementation**

module timer(

input logic clk, load, enable, [31:0] M,

output logic clkP

);

logic[31:0] dummyC;

counter C(clk, load, enable, M, clkP, dummyC);

endmodule

2. **Counter module implementation**

module counter(

input logic clk, load, enable,[31:0] countNo,

output logic tc, [31:0] OUT

);

always\_ff @( posedge clk) begin

if (load | OUT >= countNo - 1)

OUT <= 0;

else if (enable)

OUT <= OUT + 1;

tc <= (OUT == countNo - 1) ? 1 : 0;

end

endmodule

**3. Master process implementation**

module master\_process(

input logic clk, load, enable,

output logic ledCM

);

logic[31:0] A = 2000000;

logic clk\_20ms;

timer T20(clk, load, enable, A, clk\_20ms);

logic[31:0] B = 50;

logic[31:0] dummy;

logic outflow;

counter C(clk\_20ms, load, enable, B, outflow, dummy);

always\_ff @(posedge clk\_20ms) begin

ledCM <= outflow ? ~ledCM : ledCM;

end

endmodule

**4. Register implementation**

module register(

input logic clk, reset, load, [15:0] data,

output logic [15:0] Q

);

always\_ff @(posedge clk, posedge reset, posedge load) begin

if (reset)

Q <= 0;

else if(load)

Q <= data;

end

endmodule

**5. Master/slave selection** is done in master\_slave module and it can be seen in part c

**6, 7, 8. Slave process**

module slave\_process(

input logic clk, load, enable,

output logic ledSM

);

logic[31:0] A = 2000000;

logic clk\_20ms;

timer T20(clk, load, enable, A, clk\_20ms);

logic[31:0] B = 25;

logic[31:0] dummy;

logic outflow;

counter C(clk\_20ms, load, enable, B, outflow, dummy);

always\_ff @(posedge clk\_20ms) begin

ledSM <= outflow ? ~ledSM : ledSM;

end

endmodule

**part c, master\_slave (full implementation) module**

module master\_slave(

input logic clk, M, S, Em, Es, reset,

output logic M\_LED, S\_LED, CM\_LED, CS\_LED,

s0, s1, s2, s3, s4, s5, s6, dp, [3:0] an

);

assign M\_LED = M | ((~M & ~S) & Em);

assign S\_LED = (~M & S) | ((~M & ~S) & (~Em & Es)) ;

master\_process MP(clk, 0, Em & M\_LED, CM\_LED);

slave\_process SP(clk, 0, Es & S\_LED, CS\_LED);

logic [15:0] C\_1 = 16'd160;

// register(clk, reset, CM\_LED, C\_1 + 1, C\_1);

SevSeg\_4digit(clk, C\_1[3:0], C\_1[7:4], C\_1[11:8], C\_1[15:12], s0, s1, s2, s3, s4, s5, s6, dp, an);

endmodule

**testbench for master\_slave**

module testbench();

logic clk, M, S, Em, Es, reset;

logic M\_LED, S\_LED, CM\_LED, CS\_LED, s0, s1, s2, s3, s4, s5, s6, dp;

logic [3:0] an;

master\_slave(clk, M, S, Em, Es, reset,

M\_LED, S\_LED, CM\_LED, CS\_LED, s0, s1, s2, s3, s4, s5, s6, dp, an);

initial begin

M = 0;

S = 0;

Em = 0;

Es = 0;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

Es = 1;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

Em = 1;

Es = 0;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

Es = 1;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

S = 1;

Em = 0;

Es = 0;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

Es = 1;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

Em = 1;

Es = 0;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

Es = 1;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

M = 1;

S = 0;

Em = 0;

Es = 0;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

Es = 1;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

Em = 1;

Es = 0;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

Es = 1;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

S = 1;

Em = 0;

Es = 0;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

Es = 1;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

Em = 1;

Es = 0;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

Es = 1;

repeat(10) begin

clk = 1; #10;

clk = 0; #10;

end

end

endmodule;