module p3c(out, zero\_flag, alu1, alu2, data\_in, clk, write\_enable, selOP);

output reg[7:0] out, alu1, alu2;

output reg zero\_flag;

input clk, write\_enable;

input[7:0] data\_in;

input[2:0] selOP;

reg [2:0] selA, selB, selD;

reg [1:0] curr\_state, next\_state;

reg [7:1] register[7:0];

initial

begin

curr\_state =2'b00;

register[1]=8'h00;

register[2]=8'h00;

register[3]=8'h00;

register[4]=8'h00;

register[5]=8'h00;

register[6]=8'h00;

register[7]=8'h00;

out = 8'h00;

end

always @(posedge clk)

begin

case(curr\_state)

2'b00:

begin

next\_state <= 2'b01;

selA = 3'b000;

selB = 3'b010;

selD = 3'b001;

end

2'b01:

begin

next\_state <= 2'b10;

selA = 3'b100;

selB = 3'b000;

selD = 3'b010;

end

2'b10:

begin

next\_state <= 2'b00;

selA = 3'b001;

selB = 3'b010;

selD = 3'b111;

end

endcase

if(selA==0)

alu1 <=data\_in;

else

alu1 <=register[selA];

if(selB==0)

alu2 <=data\_in;

else

alu2 <=register[selB];

case(selOP)

3'b000 : out=alu1+alu2;

3'b001 : out=alu1-alu2;

3'b010 : out=alu1\*alu2;

3'b011 : out=alu1/alu2;

3'b100 : out=alu1&alu2;

3'b101 : out=alu1|alu2;

3'b110 : out=alu1^alu2;

3'b111 : out=alu1;

endcase

if(write\_enable && selD!==0)

register[selD]<=out;

if(out==0)

zero\_flag <=1;

else

zero\_flag <=0;

curr\_state<=next\_state;

end

endmodule

module tb\_p3c();

wire[7:0] out, alu1, alu2;

wire zero\_flag;

reg[7:0] data\_in;

reg clk, write\_enable;

reg[2:0] selOP;

integer i;

initial

begin

clk=1;

write\_enable=1'b1;

selOP=3'b000;

for (i=0; i<=15; i=i+1)

begin

data\_in=$urandom%2\*\*8;

#10;

end

end

always #5 clk=~clk;

p3 UUT(out, zero\_flag, alu1, alu2, data\_in, clk, write\_enable, selOP);

endmodule

