module PC\_control(C, I, F, PC\_in, PC\_out);

input [2:0] C; //condition

input [8:0] I; // offset, immediate and signed

input [2:0] F; // flag NVZ

input [15:0] PC\_in;

output[15:0] PC\_out;

wire [15:0] PC\_Add\_Out , PC\_Add\_Out2;

reg [15:0] pc\_out;

wire Ovfl;

Add\_Sub\_16bit adder(.A(PC\_in), .B(16'h0002), .sub(1'b0), .Sum(PC\_Add\_Out), .Ovfl(Ovfl));

Add\_Sub\_16bit adder2(.A(PC\_Add\_Out), .B({{6{I[8]}}, I, 1'b0}), .sub(1'b0), .Sum(PC\_Add\_Out2), .Ovfl(Ovfl));

always @(\*)

begin

case(C)

3'b000: begin // Z = 0

pc\_out = (~F[0]) ? PC\_Add\_Out2 : PC\_Add\_Out;

end

3'b001: begin // Z = 1

pc\_out = (F[0]) ? PC\_Add\_Out2 : PC\_Add\_Out;

end

3'b010: begin // Z = N = 0

pc\_out = (~F[0] & ~F[2]) ? PC\_Add\_Out2 : PC\_Add\_Out;

end

3'b011: begin // N = 1

pc\_out = (F[2]) ? PC\_Add\_Out2 : PC\_Add\_Out;

end

3'b100: begin // Z = 1 or Z = N = 0

pc\_out = (F[0] | (~F[2] & ~F[0])) ? PC\_Add\_Out2 : PC\_Add\_Out;

end

3'b101: begin // Z = 1 or N = 1

pc\_out = (F[2] | F[0]) ? PC\_Add\_Out2 : PC\_Add\_Out;

end

3'b110: begin // V = 1

pc\_out = (F[1]) ? PC\_Add\_Out2 : PC\_Add\_Out;

end

3'b111: begin // Unconditional

pc\_out = PC\_Add\_Out2;

end

default: begin

end

endcase

end

assign PC\_out = pc\_out;

endmodule

module Add\_Sub\_16bit (

input [15:0] A,

input [15:0] B, //Input values

input sub, // add-sub indicator

output [15:0] Sum, //sum output

output Ovfl //To indicate overflow

);

wire [3:0] C, Ovfls;

wire [15:0] pre\_sat\_sum;

CLA4 cla1(.A(A[3:0]), .B((sub) ? ~B[3:0] : B[3:0]), .Cin(sub), .Sum(pre\_sat\_sum[3:0]), .Cout(C[0]), .Ovfl(Ovfls[0]));

CLA4 cla2(.A(A[7:4]), .B((sub) ? ~B[7:4] : B[7:4]), .Cin(C[0]), .Sum(pre\_sat\_sum[7:4]), .Cout(C[1]), .Ovfl(Ovfls[1]));

CLA4 cla3(.A(A[11:8]), .B((sub) ? ~B[11:8] : B[11:8]), .Cin(C[1]), .Sum(pre\_sat\_sum[11:8]), .Cout(C[2]), .Ovfl(Ovfls[2]));

CLA4 cla4(.A(A[15:12]), .B((sub) ? ~B[15:12] : B[15:12]), .Cin(C[2]), .Sum(pre\_sat\_sum[15:12]), .Cout(C[3]), .Ovfl(Ovfls[3]));

assign Sum = (Ovfls[3]) ? (pre\_sat\_sum[15] ? 16'h7FFF : 16'h8000) : pre\_sat\_sum;

assign Ovfl = Ovfls[3];

endmodule

module CLA4 (

input [3:0] A,

input [3:0] B, //Input values

input Cin, // add-sub indicator

output [3:0] Sum, //sum output

output Ovfl, //To indicate overflow

output Cout

);

wire [3:0] carry;

wire [3:0] p, g;

wire [3:0] CoutTemp;

assign g = A & B;

assign p = A ^ B;

assign carry[0] = Cin;

assign carry[1] = g[0] | (p[0] & Cin);

assign carry[2] = g[1] | (p[1] & g[0]) | (p[1] & p[0] & Cin);

assign carry[3] = g[2] | (p[2] & g[1]) | (p[2] & p[1] & g[0]) | (p[2] & p[1] & p[0] & Cin);

assign Cout = g[3] | (p[3] & g[2]) | (p[3] & p[2] & g[1]) | (p[3] & p[2] & p[1] & g[0]) | (p[3] & p[2] & p[1] & p[0] & Cin);

assign Sum = p^carry;

assign Ovfl = Cout ^ carry[3];

endmodule