ALU

module ALU(data\_operandA, data\_operandB, ctrl\_ALUopcode, ctrl\_shiftamt, data\_result, isNotEqual, isLessThan, overflow);

input [31:0] data\_operandA, data\_operandB;

input [4:0] ctrl\_ALUopcode, ctrl\_shiftamt;

output [31:0] data\_result;

output isNotEqual, isLessThan,overflow;

// YOUR CODE HERE //

wire cin,cout;

wire [31:0] B;

////2 to 1 Mux, decision for add/sub,and solve cin

m21 m1(cin,0,1,ctrl\_ALUopcode[0]);

MUX2 mux2(B[31:0],data\_operandB[31:0],~data\_operandB[31:0],ctrl\_ALUopcode[0]);

//do add or sub

CSA32 csa1(data\_operandA[31:0],B[31:0],cin,data\_result[31:0],cout);

//information signal value:

xnor xnor1(overflow,data\_operandA[31],B[31],cout,~data\_result[31]);

endmodule

XOR:

module xnor32(C,A,B);

input [31:0] A,B;

output [31:0] C;

genvar i;

generate

for (i=0;i<32;i=i+1) begin: generate\_compare

xnor xnor1(C[i],A[i],B[i]);

end

endgenerate

endmodule

32-bit MUX

module MUX2(X, A0, A1, S);

parameter WIDTH=32; // How many bits wide are the lines

output [WIDTH-1:0] X; // The output line

input [WIDTH-1:0] A1; // Input line with id 1'b1

input [WIDTH-1:0] A0; // Input line with id 1'b0

input S; // Selection bit

assign X = (S == 1'b0) ? A0 : A1;

endmodule // multiplexer\_2\_1

RCA\_8

module CSA8 (a,b,cin,s,cout);

input [7:0] a,b;

input cin;

output [7:0] s;

output cout;

wire c;

wire [3:0] s0,s1;

RCA a1(a[3:0],b[3:0],cin,s[3:0],cs);

RCA a2(a[7:4],b[7:4],0,s0[3:0],c0);

RCA a3(a[7:4],b[7:4],1,s1[3:0],c1);

assign s[7:4] = (cs == 1'b0) ? s0[3:0] : s1[3:0];

assign cout =(cs == 1'b0) ? c0 : c1;

endmodule