**ALU code**

module ALU(

input logic [7:0] a\_i,

input logic [7:0] b\_i,

input logic [2:0]op\_i,

output logic [7:0] alu\_o

);

localparam OP\_ADD = 3'b000;

localparam OP\_SUB = 3'b001;

localparam OP\_SLL = 3'b001; //shift logic left

localparam OP\_LSR = 3'b001; //logic shift right

localparam OP\_AND = 3'b001;

localparam OP\_OR = 3'b001;

localparam OP\_XOR = 3'b001;

localparam OP\_EQL = 3'b001;

logic carry;

always\_comb begin

case(op\_i)

OP\_ADD: {carry,alu\_o} = {1'b0,a\_i} + {1'b0,b\_i};

OP\_SUB: alu\_o = a\_i - b\_i;

OP\_SLL: alu\_o = a\_i[7:0] << b\_i[2:0];

OP\_LSR: alu\_o = a\_i[7:0] >> b\_i[2:0];

OP\_AND: alu\_o = a\_i[7:0] & b\_i[7:0];

OP\_OR: alu\_o = a\_i | b\_i;

OP\_XOR: alu\_o = a\_i ^ b\_i;

OP\_EQL: alu\_o = {7'h0, a\_i == b\_i};

endcase

end

endmodule

**Test Bench Code**

module tb\_ALU();

logic [7:0] a\_i,b\_i;

logic [2:0]op\_i;

logic [7:0]alu\_o;

ALU day4(.\*);

initial begin

for(int j=0; j<3; j++) begin

for(int i=0; i<7; i++)begin

a\_i = $urandom\_range(0,8'hff);

b\_i = $urandom\_range(0,8'hff);

op\_i = 3'(i);

#5;

end

end

end

initial begin

$dumpfile("day4.vcd");

$dumpvars(0,tb\_ALU);

end

endmodule

