Digital Design and Computer Organisation Laboratory

UE22CS251A

3rd Semester, Academic Year 2023

Name: **ASHLESHA.T**	SRN:**PES1UG22CS118**	Section: **B**

Week#____4_____ Program Number:_____1____

TITLE: Implement a 16-bit ALU

Design and simulate a 16-bit1 ALU which performs the following core operations.

- **Arithmetic Addition and subtraction**
- **Logic AND along with OR operation**

Generate the simulation waveform using GTKWAVE.

I. alu.v Code Screenshot

ALU.v

```v
module alu_slice(input wire op0,op1,i0,i1,cin,output wire o,cout);
wire [5:0]x;
xor2 xor_1(i1,op0,x[0]);
full_adder fl(x[0],i0,cin,x[1],cout);
and2 ad(i0,i1,x[3]);
or2 od(i0,i1,x[4]);
mux2 mu1(x[3],x[4],cin,x[5]);
mux2 mu2(x[1],x[5],op1,o);
endmodule

module alu(input wire [1:0]op,input wire [15:0]i0,i1,output wire[15:0]o,output wire cout);
wire [16:0]c;
wire cin;
alu_slice a0(op[0],op[1],i0[0],i1[0],op[0],o[0],c[0]);
generate
genvar i;
for(i=1;i<15;i=i+1) begin
   alu_slice a1(op[0],op[1],i0[i],i1[i],c[i-1],o[i],c[i]);
end
endgenerate
alu_slice a2(op[0],op[1],i0[15],i1[15],c[14],o[15],cout);
endmodule
```

```v
module and2(a,b,p);
input a,b;
output p;
assign p=a&b;
endmodule
module or2(a,b,p);
input a,b;
output p;
assign p=a|b;
endmodule
module xor2(a,b,p);
input a,b;
output p;
assign p=a^b;
endmodule
module full_adder(a,b,c,sum,carry);
input a,b,c;
output sum,carry;
wire [4:0]x;
xor2 xor2_1(a,b,x[0]);
xor2 xor2_2(x[0],c,sum);
and2 and2_1(a,b,x[1]);
and2 and2_2(b,c,x[2]);
and2 and2_3(a,c,x[3]);
or2 or2_1(x[1],x[2],x[4]);
or2 or2_2(x[3],x[4],carry);
endmodule
module mux2(input wire d0,d1,s,output wire y);
 assign y=(s==0)?d0:d1;
endmodule
```

```v
module tb;
  reg clk, reset;
  reg [1:0] op; reg [15:0] i0, i1;
  wire [15:0] o; wire cout;
  reg [33:0] test_vecs [0:(`TESTVECS-1)];
  integer i;
  initial begin $dumpfile("tb_alu.vcd"); $dumpvars(0,tb); end
  initial begin reset = 1'b1; #12.5 reset = 1'b0; end
  initial clk = 1'b0; always #5 clk =~ clk;
  initial begin
    test_vecs[0][33:32] = 2'b00; test_vecs[0][31:16] = 16'h0000;test_vecs[0][15:0] = 16'h0000;
    test_vecs[1][33:32] = 2'b00; test_vecs[1][31:16] = 16'haa55;test_vecs[1][15:0] = 16'h55aa;
    test_vecs[2][33:32] = 2'b00; test_vecs[2][31:16] = 16'hffff;test_vecs[2][15:0] = 16'h0001;
    test_vecs[3][33:32] = 2'b00; test_vecs[3][31:16] = 16'h0001;test_vecs[3][15:0] = 16'h7fff;
    test_vecs[4][33:32] = 2'b01; test_vecs[4][31:16] = 16'h0000;test_vecs[4][15:0] = 16'h0000;
    test_vecs[5][33:32] = 2'b01; test_vecs[5][31:16] = 16'haa55;test_vecs[5][15:0] = 16'h55aa;
    test_vecs[6][33:32] = 2'b01; test_vecs[6][31:16] = 16'hffff;test_vecs[6][15:0] = 16'h0001;
    test_vecs[7][33:32] = 2'b01; test_vecs[7][31:16] = 16'h0001;test_vecs[7][15:0] = 16'h7fff;
    test_vecs[8][33:32] = 2'b10; test_vecs[8][31:16] = 16'h0000;test_vecs[8][15:0] = 16'h0000;
    test_vecs[9][33:32] = 2'b10; test_vecs[9][31:16] = 16'haa55;test_vecs[9][15:0] = 16'h55aa;
    test_vecs[10][33:32] = 2'b10; test_vecs[10][31:16] = 16'hffff;test_vecs[10][15:0] = 16'h0001;
    test_vecs[11][33:32] = 2'b10; test_vecs[11][31:16] = 16'h0001;test_vecs[11][15:0] = 16'h7fff;
    test_vecs[12][33:32] = 2'b11; test_vecs[12][31:16] = 16'h0000;test_vecs[12][15:0] = 16'h0000;
    test_vecs[13][33:32] = 2'b11; test_vecs[13][31:16] = 16'haa55;test_vecs[13][15:0] = 16'h55aa;
    test_vecs[14][33:32] = 2'b11; test_vecs[14][31:16] = 16'hffff;test_vecs[14][15:0] = 16'h0001;
    test_vecs[15][33:32] = 2'b11; test_vecs[15][31:16] = 16'h0001;test_vecs[15][15:0] = 16'h7fff;
  end
  initial {op, i0, i1} = 0;
  alu alu_0 (op, i0, i1, o, cout);
  initial begin
    #6 for(i=0;i<`TESTVECS;i=i+1)
      begin #10 {op, i0, i1}=test_vecs[i]; end
    #100 $finish;
  end
endmodule
```

ii)GTKWAVE Screenshot