Name: Sriram R	SRN: PES1UG20CS435	Section: H
	Date: 22-09-2021	Week Number: 3

1	Construct a 16 bit ALU using basic gates and full adder modules

Program:

alu.v :

```
module xor2 (input wire i0, i1, output wire o);
  assign o = i0 ^ i1;
endmodule

module and2 (input wire i0, i1, output wire o);
  assign o = i0 & i1;
endmodule

module or2 (input wire i0, i1, output wire o);
  assign o = i0 | i1;
endmodule

module mux2 (input wire i0, i1, j, output wire o);
  assign o = (j==0)?i0:i1;
endmodule

module fulladd(input wire a, b, cin, output wire sum, cout);
wire [4:0] t;
    xor2 x0(a, b, t[0]);
    xor2 x1(t[0], cin, sum);

    and2 a0(a, b, t[1]);
    and2 a1(a, cin, t[2]);
    and2 a2(b, cin, t[3]);

    or2 o0(t[1], t[2], t[4]);
    or2 o1(t[3], t[4], cout);
endmodule

// Write code for modules you need here

module alu1(input wire [1:0] op, input wire i0, i1, cin, output wire o, cout);

// Declare wires here
```

```
        wire xorOut;
        wire fullAddSum;
        wire andOut;
        wire orOut;
        wire mux1Out;
        wire mux2Out;

// Instantiate modules here
        xor2 x1(i1, op[0], xorOut);
        fulladd f1(i0, xorOut, op[0], fullAddSum, cout);
        and2 a1(i0, i1, andOut);
        or2 o1(i0, i1, orOut);
        mux2 m1(andOut, orOut, op[0], mux1Out);
        mux2 m2(mux1Out, fullAddSum, op[1], o);

endmodule

module alu(input wire [1:0] op, input wire [15:0] i0, i1, output wire [15:0] o, output wire
cout);

// Declare wires here
        wire [14:0]c;

// Instantiate modules here
        alu1 a1(i0[0],i1[0],op[0],op[1],o[0],c[0]);
        alu1 a2(i0[1],i1[1],c[0],op[1],o[1],c[1]);
        alu1 a3(i0[2],i1[2],c[1],op[1],o[2],c[2]);
        alu1 a4(i0[3],i1[3],c[2],op[1],o[3],c[3]);
        alu1 a5(i0[4],i1[4],c[3],op[1],o[4],c[4]);
        alu1 a6(i0[5],i1[5],c[4],op[1],o[5],c[5]);
        alu1 a7(i0[6],i1[6],c[5],op[1],o[6],c[6]);
        alu1 a8(i0[7],i1[7],c[6],op[1],o[7],c[7]);
        alu1 a9(i0[8],i1[8],c[7],op[1],o[8],c[8]);
        alu1 a10(i0[9],i1[9],c[8],op[1],o[9],c[9]);
        alu1 a11(i0[10],i1[10],c[9],op[1],o[10],c[10]);
        alu1 a12(i0[11],i1[11],c[10],op[1],o[11],c[11]);
        alu1 a13(i0[12],i1[12],c[11],op[1],o[12],c[12]);
        alu1 a14(i0[13],i1[13],c[12],op[1],o[13],c[13]);
        alu1 a15(i0[14],i1[14],c[13],op[1],o[14],c[14]);
        alu1 a16(i0[15],i1[15],c[14],op[1],o[15],cout);

endmodule
```

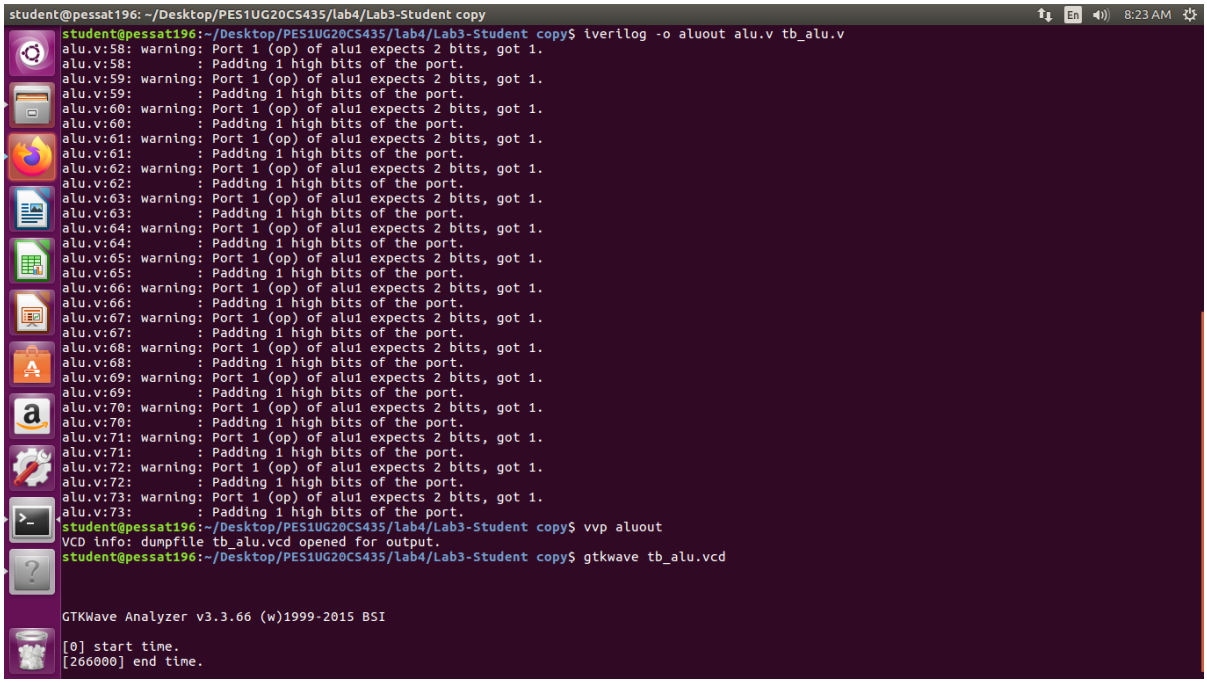
alu_tb.v :

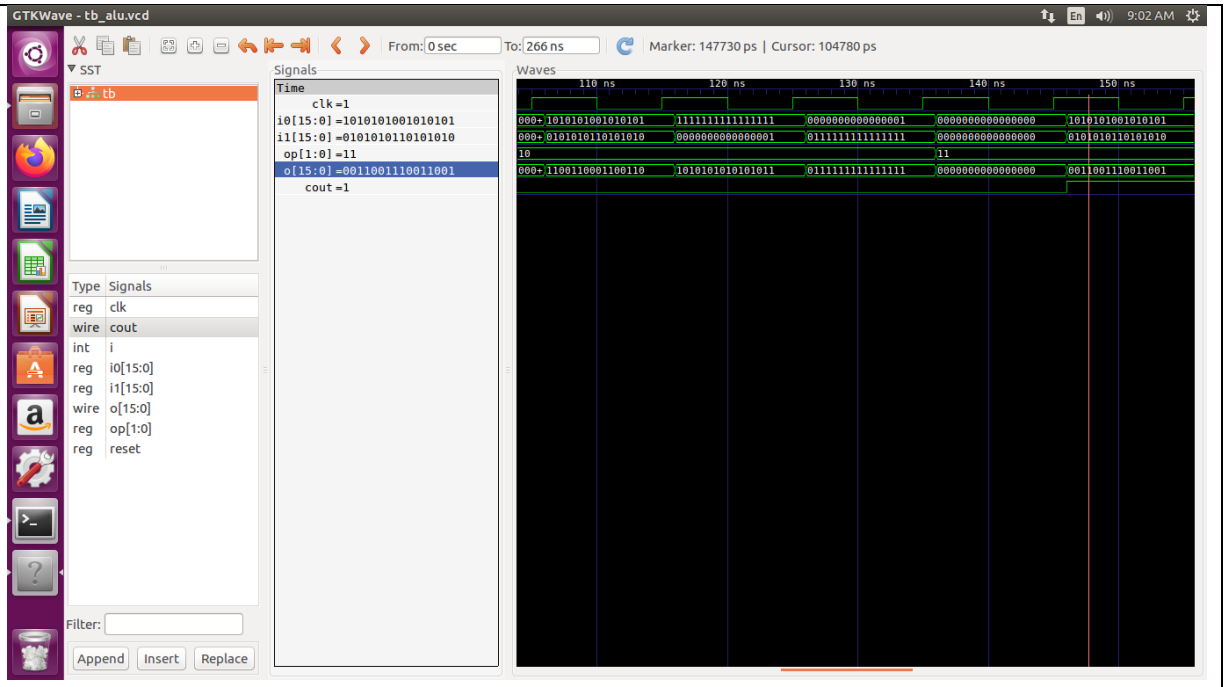
```
`timescale 1 ns / 100 ps
`define TESTVECS 16

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
```

Output Screenshot:




GTKWave - tb_alu.vcd
9:02 AM
From: 0 sec
To: 266 ns
Marker: 147730 ps | Cursor: 104780 ps
SST
tb
Signals
Waves
Time
clk =1
i0[15:0] =1010101001010101
i1[15:0] =0101010110101010
op[1:0] =11
o[15:0] =0011001110011001
cout =1
110 ns
120 ns
130 ns
140 ns
150 ns
Type Signals
reg clk
wire cout
int i
reg i0[15:0]
reg i1[15:0]
wire o[15:0]
reg op[1:0]
reg reset
Filter:
Append
Insert
Replace