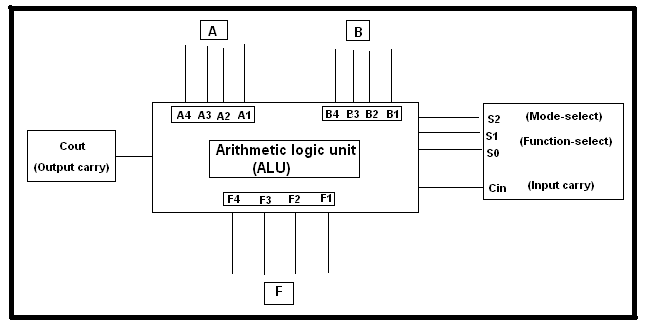
**PROBLEM 1.** Design an Arithmetic and Logic Unit (ALU)

**CIRCUIT DIAGRAM:**



**TRUTH TABLE:**

**Control Instruction Operation**

000 Add Output <= A + B

Cout contains the carry

001 Sub Output <= A – B ;

Cout contains the borrow

010 Or Output <= A or B

011 And Output <= A and B

100 Shl Output <= A[2:0] & ‘0’

101 Shr Output <= ‘0’ & A[3:1]

110 Rol Output <= A[2:0] & A[3]

111 Ror Output <= A[0] & A [3:1]

**(RANDOM VALUE A&B)**

**Sel A B COUT RESULT**

000 1000 1000 1 0000

001 1001 1000 0 0001

010 0010 0101 0 0111

011 0010 0101 0 0000

100 0010 0101 0 0100

101 0010 0101 0 0001

110 0010 0101 0 0100

111 0010 0101 0 0001

**VERILOG CODE:**

module ALU( input [3:0]A,

input [3:0]B,

input [2:0]sel,

output [4:0]R);

wire [3:0]Reg1;

wire [3:0]Reg2;

reg [4:0]Reg3;

assign Reg1 = A;

assign Reg2 = B;

assign R = Reg3;

always @(sel,Reg1,Reg2,Reg3)

begin

case (sel)

0 : Reg3 = Reg1 + Reg2;

1 : Reg3 = Reg1 - Reg2;

2 : Reg3 = Reg1 | Reg2;

3 : Reg3 = Reg1 & Reg2;

4 : Reg3 = Reg1<<1;

5 : Reg3 = Reg1>>1;

6 : Reg3 = {Reg1[2:0],Reg1[3]};

7 : Reg3 = {Reg1[0], Reg1[3:1]};

endcase

end

endmodule

**TEST BENCH:**

module ALU\_TB;

reg [3:0]a;

reg [3:0]b;

reg [2:0] SEL;

wire [4:0] OUT ;

ALU uut(.A(a),.B(b),.sel(SEL),.R(OUT));

initial

begin

a=4'd8;b=4'd8;SEL=000;

#100

a=3'd2;b=4'd5;SEL=001;

#100

a=3'd2;b=4'd5;SEL=010;

#100

a=3'd2;b=4'd5;SEL=011;

#100

a=3'd2;b=4'd5;SEL=100;

#100

a=3'd2;b=4'd5;SEL=101;

#100

a=3'd2;b=4'd5;SEL=110;

#100

a=3'd2;b=4'd5;SEL=111;

#100

a=3'd6;b=4'd9;SEL=000;

#100

a=3'd7;b=4'd10;SEL=001;

#100

a=3'd5;b=4'd5;SEL=010;

#100

a=3'd4;b=4'd5;SEL=011;

#100

a=3'd2;b=4'd12;SEL=100;

#100

a=3'd1;b=4'd10;SEL=101;

#100

a=3'd4;b=4'd11;SEL=110;

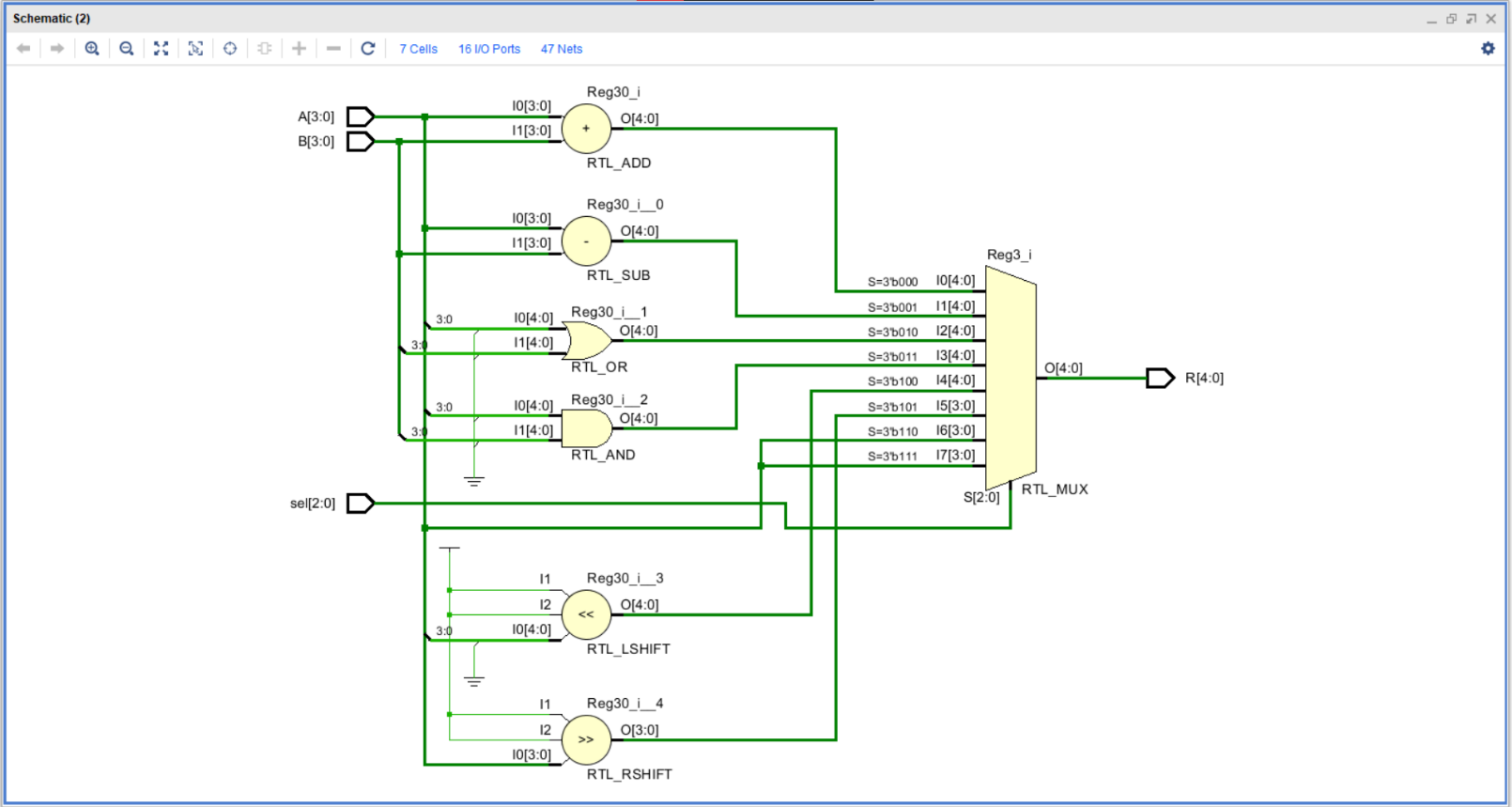
#100

a=3'd1;b=4'd5;SEL=111;

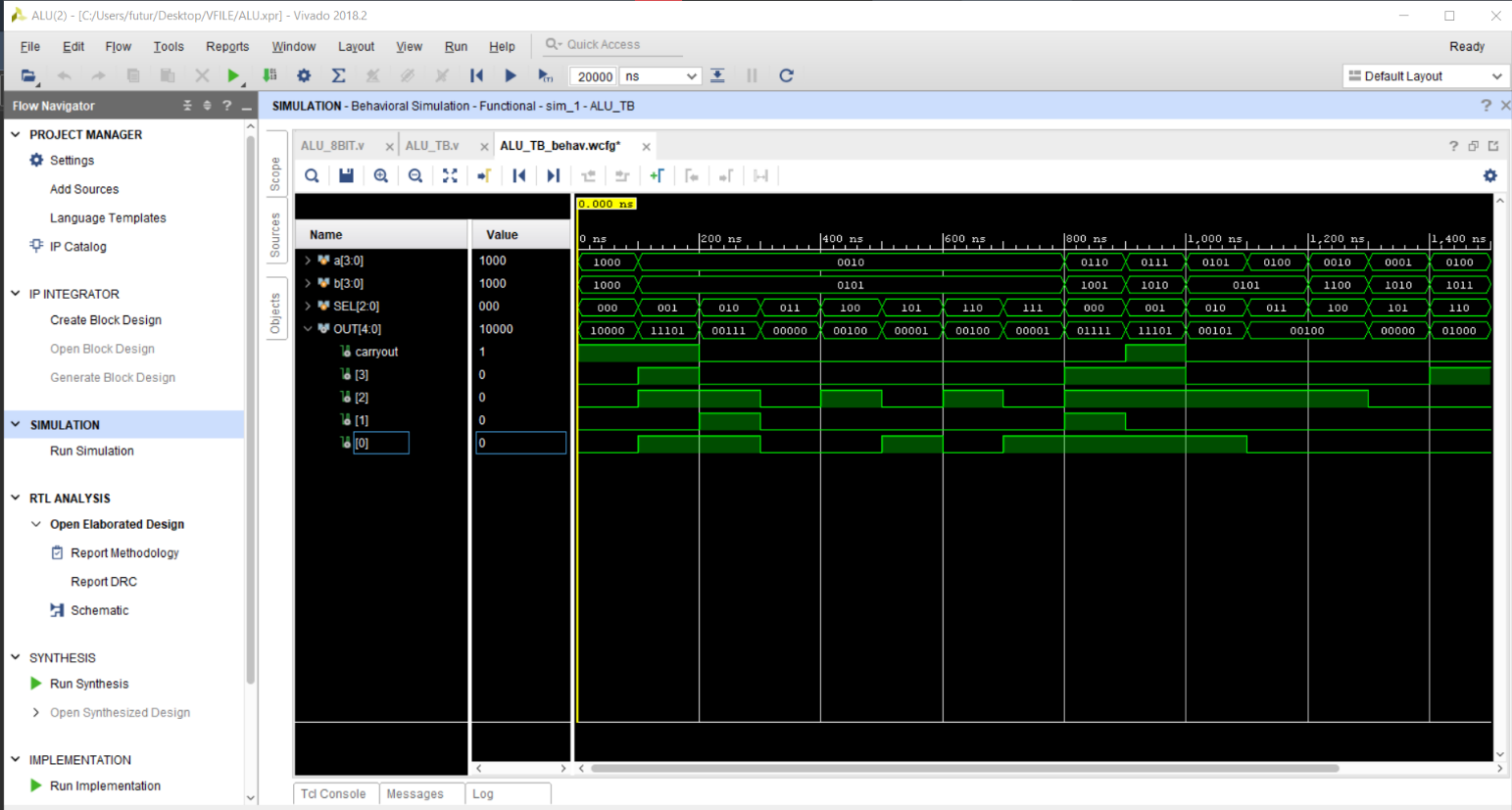
end

endmodule

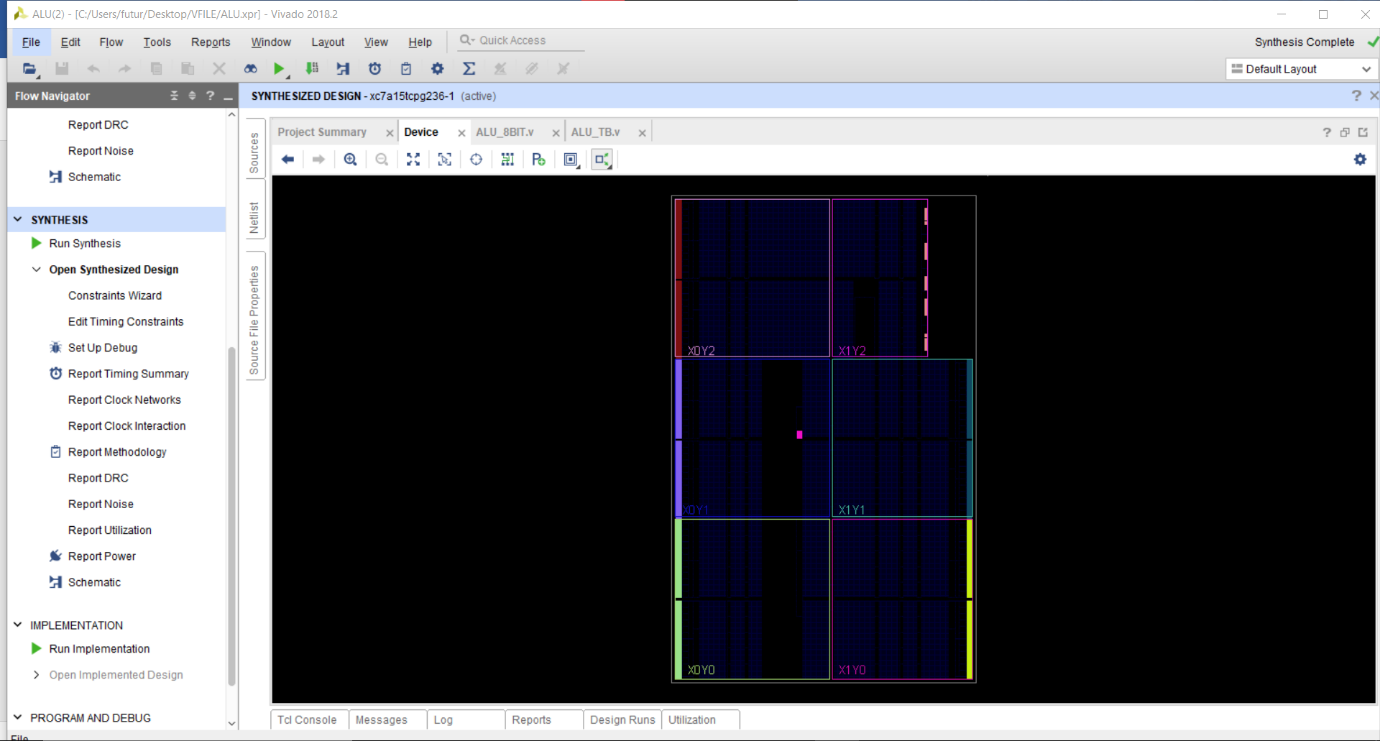
**SCHEMATIC DIAGRAM:**



**OUTPUT:**



**REPORT UTILIZATION:**



**CONCLUSION :** we have learn design ALU using Verilog.