**AMINA QADEER**

**359607**

**CE-42-A**

**LABFINAL**

**4X4 signed multiplier using Wallace reduction.**

**Verilog code:**

module half\_adder(A, B, S, C);

input A, B;

output S, C;

assign {S, C} = A + B;

endmodule

module full\_adder(A, B, Cin, S, Cout);

input A, B, Cin;

output S, Cout;

assign {S, Cout} = A + B + Cin;

endmodule

module wallace(A, B, prod);

input [3:0] A, B;

output [7:0] prod;

wire s11, s12, s13, s14, s15, s22, s23, s24, s25, s26, s32, s33, s34, s35, s36, s37;

wire c11, c12, c13, c14, c15, c22, c23, c24, c25, c26, c32, c33, c34, c35, c36, c37;

wire [6:0] p0, p1, p2, p3;

half\_adder ha11(p0[1], p1[0], s11, c11);

full\_adder fa12(p0[2], p1[1], p2[0], s12, c12);

full\_adder fa13(p0[3], p1[2], p2[1], s13, c13);

full\_adder fa14(p1[3], p2[2], p3[1], s14, c14);

half\_adder ha15(p2[3], p3[2], s15, c15);

half\_adder ha22(c11, s12, s22, c22);

full\_adder fa23(p3[0], c12, s13, s23, c23);

full\_adder fa24(c13, c32, s14, s24, c24);

full\_adder fa25(c14, c24, s15, s25, c25);

full\_adder fa26(c15, c25, p3[3], s26, c26);

half\_adder ha32(c22, s23, s32, c32);

half\_adder ha34(c23, s24, s34, c34);

half\_adder ha35(c34, s25, s35, c35);

half\_adder ha36(c35, s26, s36, c36);

half\_adder ha37(c36, c26, s37, c37);

assign p0 = A & {4{B[0]}};

assign p1 = A & {4{B[1]}};

assign p2 = A & {4{B[2]}};

assign p3 = A & {4{B[3]}};

assign prod[0] = p0[0];

assign prod[1] = s11;

assign prod[2] = s22;

assign prod[3] = s32;

assign prod[4] = s34;

assign prod[5] = s35;

assign prod[6] = s36;

assign prod[7] = s37;

endmodule

**Testbench:**

`timescale 1ns / 1ps

module tb;

// Inputs

reg [3:0] A;

reg [3:0] B;

// Outputs

wire [7:0] prod;

integer i,j,error;

// Instantiate the Unit Under Test (UUT)

wallace uut (

.A(A),

.B(B),

.prod(prod)

);

initial begin

// Apply inputs for the whole range of A and B.

// 16\*16 = 256 inputs.

error = 0;

for(i=0;i <=15;i = i+1)

for(j=0;j <=15;j = j+1)

begin

A <= i;

B <= j;

#1;

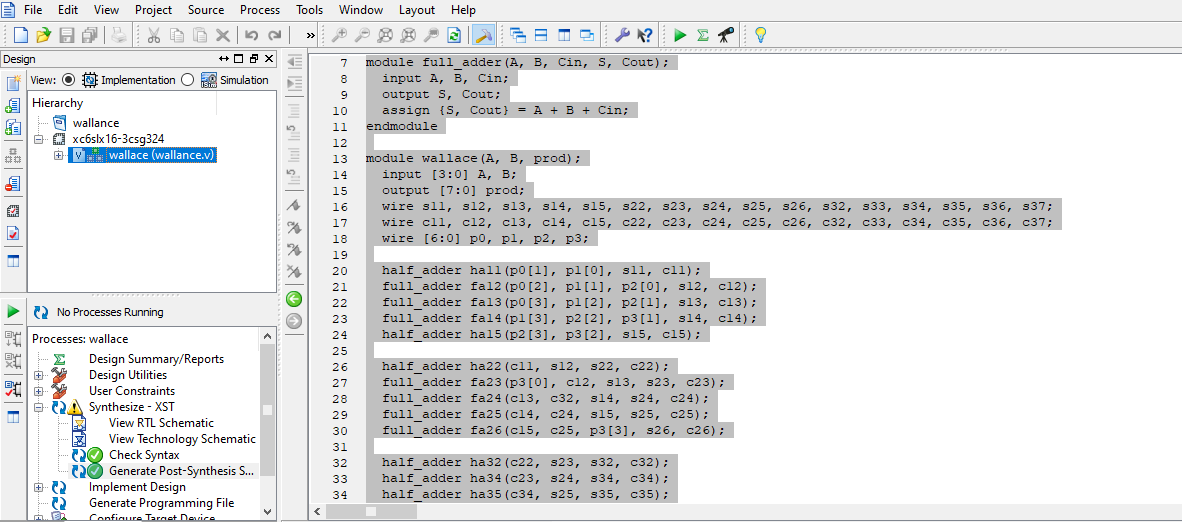
if(prod != A\*B) //if the result isnt correct increment "error".

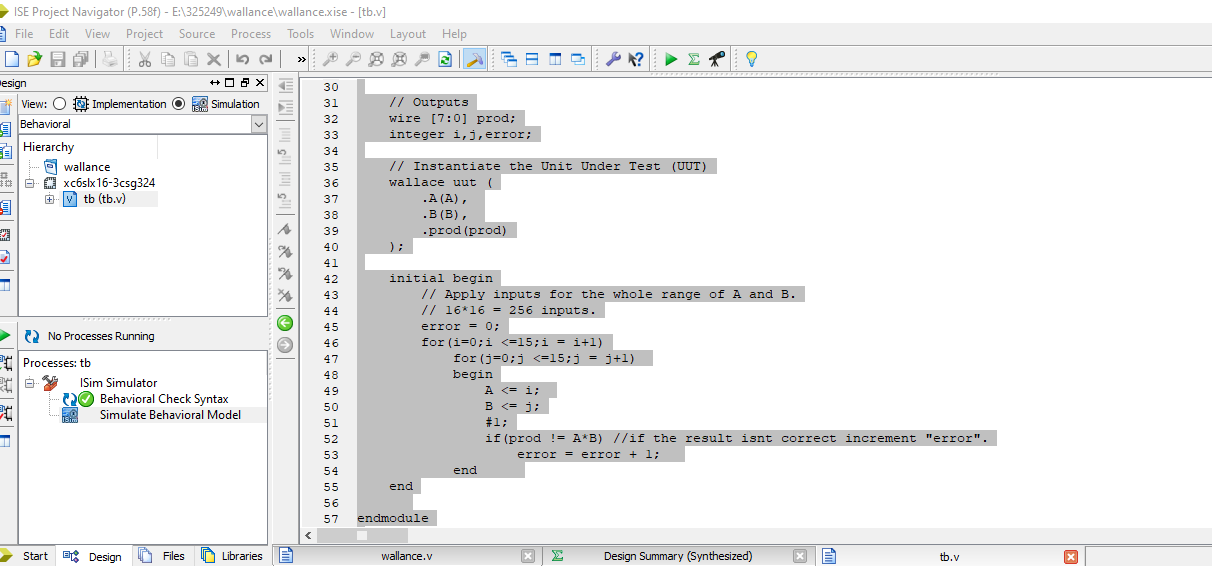
error = error + 1;

end

end

endmodule



**result:**

