**source code for 64 bit adder with 4\*16**

module not\_gate(abar,a);

input a;

output abar;

not x1(abar,a);

endmodule

module and\_gate(c,a,b);

input a,b;

output c;

and r1(c,a,b);

endmodule

module or\_gate(c,a,b);

input a,b;

output c;

or r1(c,a,b);

endmodule

module xor\_gate(out,a,b);

input a,b;

output out;

wire abar,bbar,y1,y2;

not\_gate xx1(abar,a);

not\_gate xx2(bbar,b);

and\_gate xx3(y1,a,bbar);

and\_gate xx4(y2,abar,b);

or\_gate xx5(out,y1,y2);

endmodule

module half\_adder(a,b,sum,carry);

input a,b;

output sum;

output carry;

xor\_gate xx1(sum,a,b);

and\_gate xx2 (carry,a,b);

endmodule

module full\_adder(a,b,cin,fsum,fcarry);

input a,b,cin;

output fsum,fcarry;

wire s\_1,c\_1;

half\_adder xx1 (a,b,s\_1,c\_1);

half\_adder xx2(s\_1,cin,fsum,c\_2);

or\_gate xx3 (fcarry,c\_1,c\_2);

endmodule

module full\_adder\_withoutcarry(a,b,cin,s);

input a,b,cin;

output s;

wire s\_1;

xor\_gate x1 (s\_1,a,b);

xor\_gate x2 (s,s\_1,cin);

endmodule

module four\_bit\_adder(a,b,cin,sum,carry);

input [3:0]a,b;

input cin;

output carry;

output [3:0] sum;

wire c1,c2,c3;

full\_adder x1(a[0],b[0],cin,sum[0],c1);

full\_adder x2(a[1],b[1],c1,sum[1],c2);

full\_adder x3(a[2],b[2],c2,sum[2],c3);

full\_adder\_withoutcarry x4(a[3],b[3],c3,sum[3]);

four\_bit\_carry x5(a,b,cin,carry);

endmodule

module four\_bit\_carry(a,b,cin,carry);

input [3:0]a,b;

input cin;

output carry;

wire x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,x11,x12,x13,x14,x15,x16,x17,x18,x19,x20,x21,x22,x23;

and\_gate a1(x1,a[3],b[3]);

and\_gate a2 (x2,a[0],b[0]);

and\_gate a3 (x3,a[1],b[1]);

and\_gate a4 (x4,a[2],b[2]);

or\_gate a5 (x5,a[3],b[3]);

or\_gate a6 (x6,a[2],b[2]);

or\_gate a7 (x7,a[1],b[1]);

or\_gate a8 (x8,a[0],b[0]);

and\_gate a9 (x9,x4,x5);

and\_gate a10 (x10,x5,x6);

and\_gate a11 (x11,x7,x8);

and\_gate a12(x12,x10,x11);

and\_gate a13(x13,x2,x12);

and\_gate a14(x14,x3,x5);

and\_gate a15(x15,x2,x12);

and\_gate a16(x16,x3,x12);

and\_gate a17(x17,x14,x6);

or\_gate a18(x18,x13,x17);

or\_gate a19(x19,x15,x16);

or\_gate a20(x20,x19,x9);

or\_gate a21(x21,x18,x20);

or\_gate a22(x22,x1,x21);

and\_gate a23(x23,x12,cin);

or\_gate a24(carry,x22,x23);

endmodule

module sixtyfour\_bit\_adder (a,b,cin,sum,cout);

input [63:0]a,b;

input cin;

output [63:0]sum;

output cout;

wire c4,c8,c12,c16,c20,c24,c28,c32,c36,c40,c44,c48,c52,c56,c60;

four\_bit\_adder x1 (a[3:0],b[3:0],cin,sum[3:0],c4);

four\_bit\_adder x2 (a[7:4],b[7:4],c4,sum[7:4],c8);

four\_bit\_adder x3 (a[11:8],b[11:8],c8,sum[11:8],c12);

four\_bit\_adder x4 (a[15:12],b[15:12],c12,sum[15:12],c16);

four\_bit\_adder x5 (a[19:16],b[19:16],c16,sum[19:16],c20);

four\_bit\_adder x6 (a[23:20],b[23:20],c20,sum[23:20],c24);

four\_bit\_adder x7 (a[27:24],b[27:24],c24,sum[27:24],c28);

four\_bit\_adder x8 (a[31:28],b[31:28],c28,sum[31:28],c32);

four\_bit\_adder x9 (a[35:32],b[35:32],c32,sum[35:32],c36);

four\_bit\_adder x10 (a[39:36],b[39:36],c36,sum[39:36],c40);

four\_bit\_adder x11 (a[43:40],b[43:40],c40,sum[43:40],c44);

four\_bit\_adder x12 (a[47:44],b[47:44],c44,sum[47:44],c48);

four\_bit\_adder x13 (a[51:48],b[51:48],c48,sum[51:48],c52);

four\_bit\_adder x14 (a[55:49],b[55:49],c52,sum[55:52],c56);

four\_bit\_adder x15 (a[59:56],b[55:56],c56,sum[55:59],c60);

four\_bit\_adder x16 (a[63:60],b[63:60],c60,sum[63:60],cout);

endmodule

module sixtyfour\_bit\_adder\_tb;

reg [63:0]a,b;

reg cin;

wire sum;

wire carry;

reg clk;

integer i,j,k;

sixtyfour\_bit\_adder s1(a,b,cin,sum,carry);

always

#10 clk=~clk;

initial

begin

clk=0;

a=0;

b=0;

cin=0;

for(i=0;i<100;i=i+1)

for(j=0;j<100;j=j+1)

for(k=0;k<2;k=k+1)

begin

a=i;

b=j;

cin=k;

#20;

$monitor("a=%h, b=%h, cin=%b, carry=%b, sum=%b",a,b,cin,carry,sum);

end

end

endmodule