And gate:

And gate.v:

module and\_gate(

input a,b,

output y);

//Above style of declaring ports is ANSI style.Verilog2001 Feature

assign y = a & b;

endmodule

Test bench and gate:

module tb\_and\_gate;

reg A,B;//inputs

wire Y;// connection to output

and\_gate a1 (.a(A) ,.b(B),.y(Y));// Unit Under Test

//Above style is connecting by names

initial begin

A =1'b0; //Meaning is , input is single bit and value is zero

B= 1'b0;

#45 $finish;

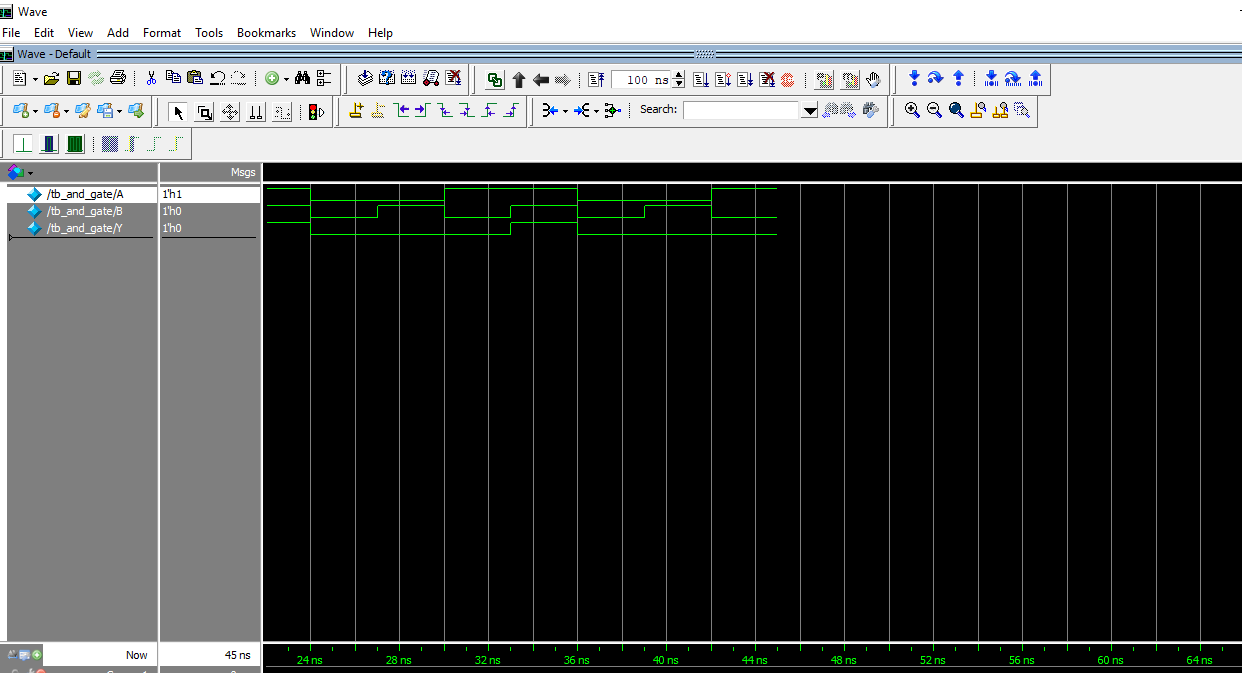
end

always #6 A =~A; // Delay to change the input values

always #3 B =~B;

endmodule

Screenshot: (And gate)



Or gate:

Orgate.v

module or\_gate(

input a,b,

output reg y);

always @(a,b)

y = a |b;

endmodule

Test bench or gate:

module tb\_or\_gate;

reg A,B;

wire Y;

or\_gate a1 (.a(A) ,.b(B),.y(Y));

initial begin

A =1'b0;

B= 1'b0;

#45 $finish;//Check in ModelSim Reference

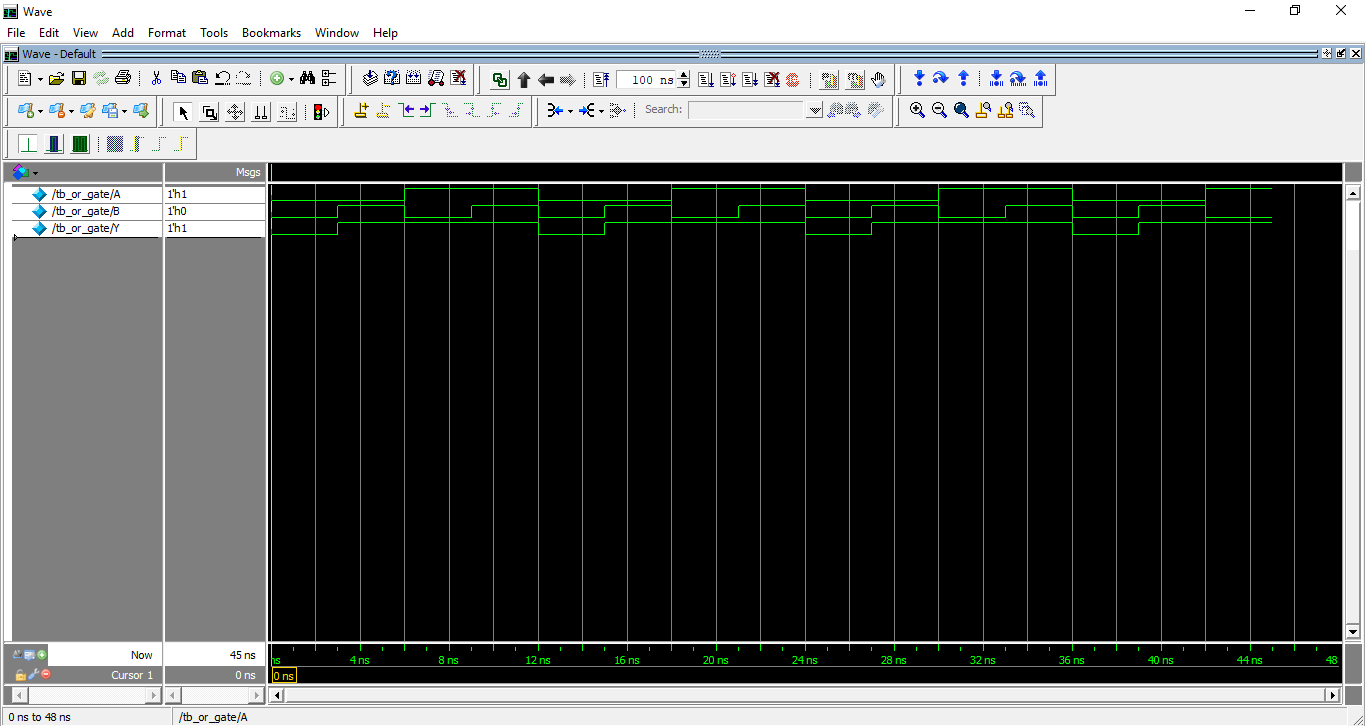
end

always #6 A =~A;

always #3 B =~B;

endmodule

Screenshot: (Or gate)



Xor gate:

Xor gate.v

module xor\_gate (

input a,b,

output y);

xor x1(y,a, b); //xor is a built in primitive. While using these primitives //you should follow the connection rules. First signal should be output and //then inputs.

Endmodule

Test bench for xor gate:

module tb\_xor\_gate;

reg A,B;

wire Y;

xor\_gate a1 (.a(A) ,.b(B),.y(Y)); //Unit under Test

initial begin

A =1'b0;

B= 1'b0;

#45 $finish;

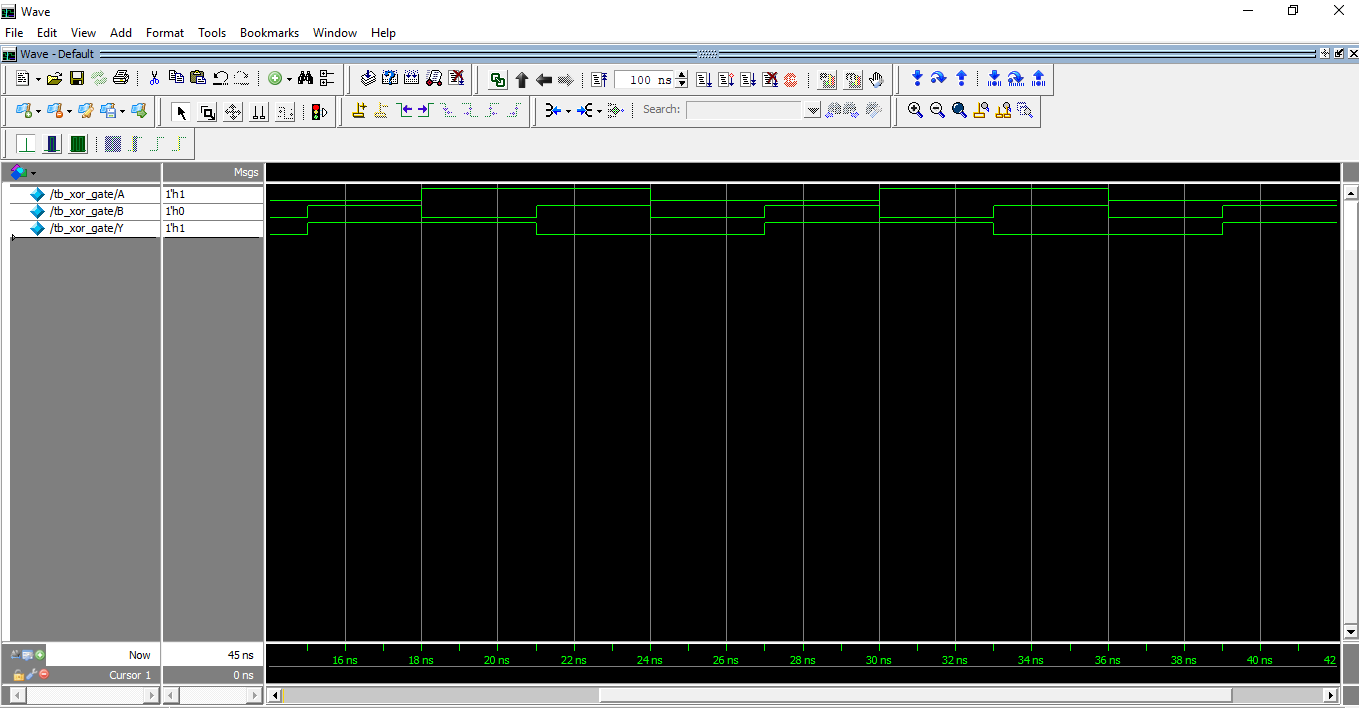
end

always #6 A =~A;

always #3 B =~B;

endmodule

Screenshot: (XOR gate)



Half adder:

Halfadder.v:

module half\_adder(

input a,b,

output sum,carry);

assign sum = a^b;

assign carry = a & b;

endmodule

Test bench for Half adder:

module half\_adder(

input a,b,

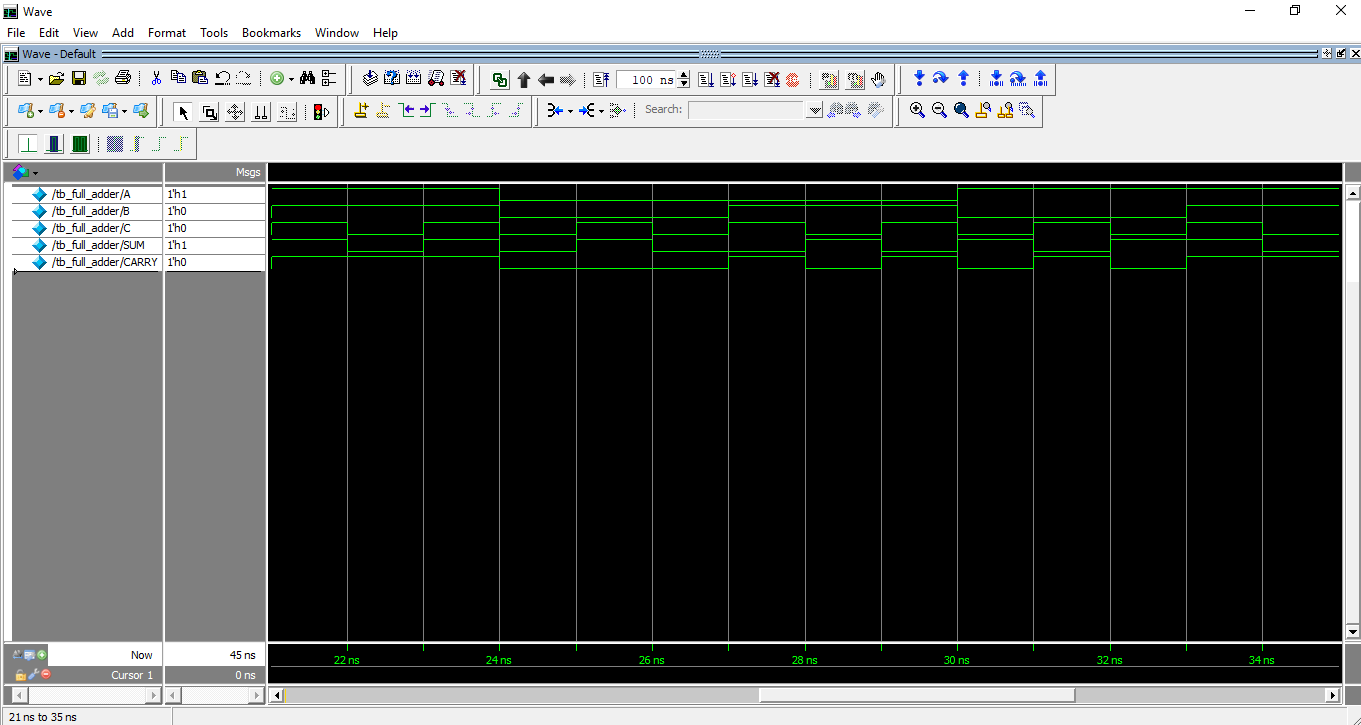
output sum,carry);

assign sum = a^b;

assign carry = a & b;

endmodule

Screenshot:(Half adder)



Full Adder:

Full Adder.v:

module full\_adder(

input a,b,c,

output sum,carry);

assign sum = a^b^c;

assign carry = a & b | b & c | c & a;

endmodule

Test bench for Full adder:

Tb\_full\_adder.v:

module tb\_full\_adder;

reg A,B,C;

wire SUM,CARRY;

full\_adder FA (.a(A) ,.b(B),.c(C),.sum(SUM),.carry(CARRY));

initial begin

A =1'b0;

B= 1'b0;

C= 1'b0;

#45 $finish;

end

always #6 A =~A;

always #3 B =~B;

always #1 C = ~C;

endmodule

ScreenShot: (Full adder):

