**Experiment 1. Realization of Logic Gates and Familiarization of Verilog**

Date: 11/05/21

**AIM**

a.Write a verilog program for OR gate in gate level,data flow level and behavioral level modelling  
  
b.Write a verilog program for the boolean expression y=ab+bc+ac in gate level,data flow level and behavioral level modelling

**a. OR GATE**

**VERILOG CODE**

**Gate level**

module or\_gate(output y,input a,b);

or(y,a,b);

endmodule

**Data flow level**

module or\_gate

(input wire a,b,

output wire y);

assign y = a||b;

endmodule

**Behavioural level**

module or\_gate(output reg y,input a,b);

always@(a or b)

begin

if(a==1'b0 & b==1'b0)

begin

y = 1'b0;

end

else

y = 1'b1;

end

endmodule

**Testbench**

`timescale 1ns/1ps

module test\_or;

reg t\_a,t\_b;

wire t\_y;

or\_gate uut(.a(t\_a),.b(t\_b),.y(t\_y));

initial

begin

$dumpvars(1,test\_or);

t\_a = 1'b0;

t\_b = 1'b0;

#10;

t\_a = 1'b0;

t\_b = 1'b1;

#10;

t\_a = 1'b1;

t\_b = 1'b0;

#10;

t\_a = 1'b1;

t\_b = 1'b1;

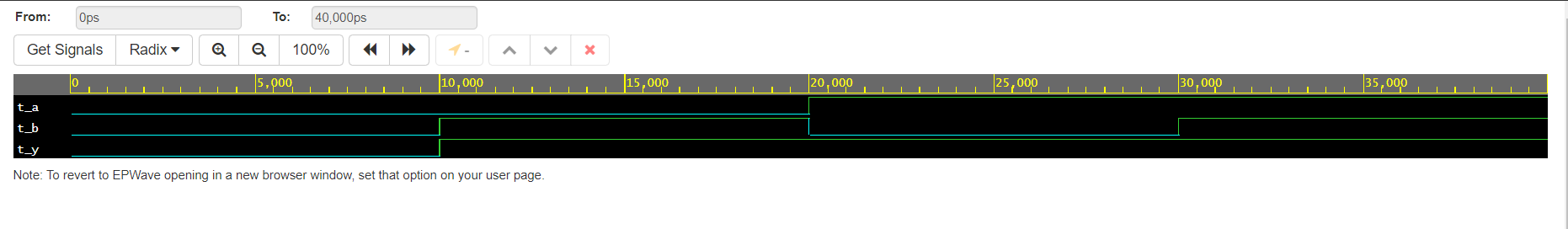
#10;

$stop;

end

endmodule

**Output**



**b. y = ab+ac+bc**

**VERILOG CODE**

**Gate level**

module boolean\_exp(output y,input a,b,c);

or(y,(a&b),(b&c),(a&c));

endmodule

**Data flow level**

module boolean\_exp

(input wire a,b,c,

output wire y);

assign y = (a&b)|(b&c)|(a&c);

endmodule

**Behavioural level**

module boolean\_exp(output reg y,input a,b,c);

always@(a or b or c)

begin

if((a==1'b1 & b==1'b1)||(a==1'b1 & c==1'b1)||(b==1'b1 & c==1'b1))

begin

y = 1'b1;

end

else

begin

y = 1'b0;

end

end

endmodule

**Testbench**

`timescale 1ns/1ps

module test\_boolean\_exp;

reg t\_a,t\_b,t\_c;

wire t\_y;

boolean\_exp uut(.a(t\_a),.b(t\_b),.c(t\_c),.y(t\_y));

initial

begin

$dumpvars(1,test\_boolean\_exp);

t\_a = 1'b0;

t\_b = 1'b0;

t\_c = 1'b0;

#10;

t\_a = 1'b0;

t\_b = 1'b0;

t\_c = 1'b1;

#10;

t\_a = 1'b0;

t\_b = 1'b1;

t\_c = 1'b0;

#10;

t\_a = 1'b0;

t\_b = 1'b1;

t\_c = 1'b1;

#10;

t\_a = 1'b1;

t\_b = 1'b0;

t\_c = 1'b0;

#10;

t\_a = 1'b1;

t\_b = 1'b0;

t\_c = 1'b1;

#10;

t\_a = 1'b1;

t\_b = 1'b1;

t\_c = 1'b0;

#10;

t\_a = 1'b1;

t\_b = 1'b1;

t\_c = 1'b1;

#10;

$stop;

end

endmodule

**Output**

