**COA LAB EVALUATION -1**

**Name: Yaswanth ram Ponnada**

**Roll no: CB.EN.U4CSE18243**

**Q1** ) Using NAND gate, construct AND, OR, NOT & XOR gates

**CODE** :

module universalnand(a,b,notgate,orgate,andgate,xorgate);

input a,b;

output notgate,orgate,andgate,xorgate;

assign notgate = !(a&a);

assign orgate = !((!(a&a))& (!(b&b)));

assign andgate = !((!(a&b)) & (!(a&b)));

assign xorgate = !((a| (!b)) & ((!a) | b));

endmodule

module universalnand\_tb;

wire t\_andgate,t\_orgate,t\_notgate,t\_xorgate;

reg t\_a,t\_b;

universalnand my\_gate(.a(t\_a),.b(t\_b),.andgate(t\_andgate),.orgate(t\_orgate),.notgate(t\_notgate), .xorgate(t\_xorgate));

initial

begin

$monitor("time= %d: Input a=%b b=%b and = %b or = %b not(a) = %b xor = %b\n",$time,t\_a,t\_b,t\_andgate,t\_orgate,t\_notgate , t\_xorgate);

t\_a = 1'b0;

t\_b = 1'b0;

#5

t\_a = 1'b0;

t\_b = 1'b1;

#5

t\_a = 1'b1;

t\_b = 1'b0;

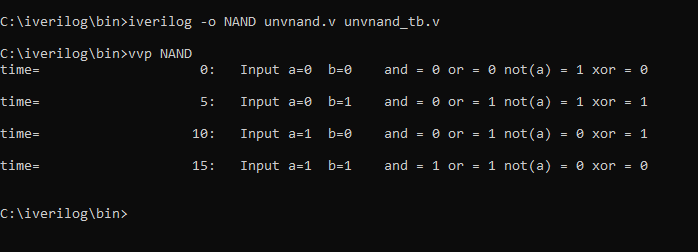
#5

t\_a = 1'b1;

t\_b = 1'b1;

end

Endmodule



**Q2 )** Using NOR gate, construct AND, OR, NOT & XOR gates

**CODE** :

module universalnor(a,b,andgate,orgate,notgate,xorgate);

input a,b;

output andgate,orgate,notgate,xorgate;

assign andgate=!((!(a|a))|(!(b|b)));

assign orgate=!((!(a|b))|(!(a|b)));

assign notgate=!(a|a);

assign xorgate = ((!a) & b) | (a & (!b));

endmodule

module unviversalnor\_tb;

wire t\_andgate,t\_orgate,t\_notgate,t\_xorgate;

reg t\_a,t\_b;

universalnor my\_gate(.a(t\_a),.b(t\_b),.andgate(t\_andgate),.orgate(t\_orgate),.notgate(t\_notgate),.xorgate(t\_xorgate));

initial

begin

$monitor("time= %d: Input a=%b b=%b and = %b or = %b not = %b xor = %b \n",$time,t\_a,t\_b,t\_andgate,t\_orgate,t\_notgate,t\_xorgate);

t\_a = 1'b0;

t\_b = 1'b0;

#5

t\_a = 1'b0;

t\_b = 1'b1;

#5

t\_a = 1'b1;

t\_b = 1'b0;

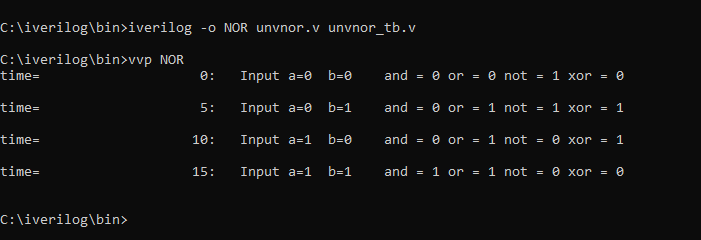
#5

t\_a = 1'b1;

t\_b = 1'b1;

end

endmodule



**Q3 )** Half Adder

**CODE** :

module halfadder(x,y,sum,carry);

input x,y;

output sum,carry;

xor(sum,x,y);

and(carry,x,y);

initial begin

$display("\nThis is Half Adder output\n");

end

endmodule

module halfadder\_tb;

reg t\_x,t\_y;

halfadder ha(t\_x,t\_y,t\_sum,t\_carry);

initial

begin

$monitor("x= %b y= %b sum=%b carry=%b",t\_x,t\_y,t\_sum,t\_carry);

t\_x=1'b0;

t\_y=1'b0;

#10

t\_x=1'b0;

t\_y=1'b1;

#10

t\_x=1'b1;

t\_y=1'b0;

#10

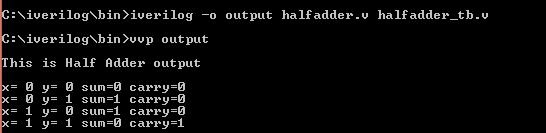
t\_x=1'b1;

t\_y=1'b1;

end

endmodule

**OUTPUT** :



**Q4 )** Full subtractor

**CODE** :

module fullsub(a,b,c,difference,borrow);

input a ,b , c;

output difference, borrow;

assign difference =(a^(b^c));

assign borrow=((!a & (b|c))|(b&c));

endmodule

module fullsub\_tb;

reg a, b, c;

wire difference, borrow;

fullsub sub(a,b,c,difference,borrow);

initial

begin

$monitor("time :%d Input a: %b b: %b c: %b diff: %b borr: %b ",$time, a,b,c,difference, borrow);

a=0;b=0;c=0;

#5

c=1;

#5

b=1;

#5

a=1;

#5

b=0;

#5

c=0;

#5

b=1;

#5

a=0;

end

endmodule

**OUTPUT** :

