PROJECT NO. – 2

Developed by: Aashi Srivastava National Institute of Technology, Warangal

AIM : To use universal NAND gate to form other logic gates.

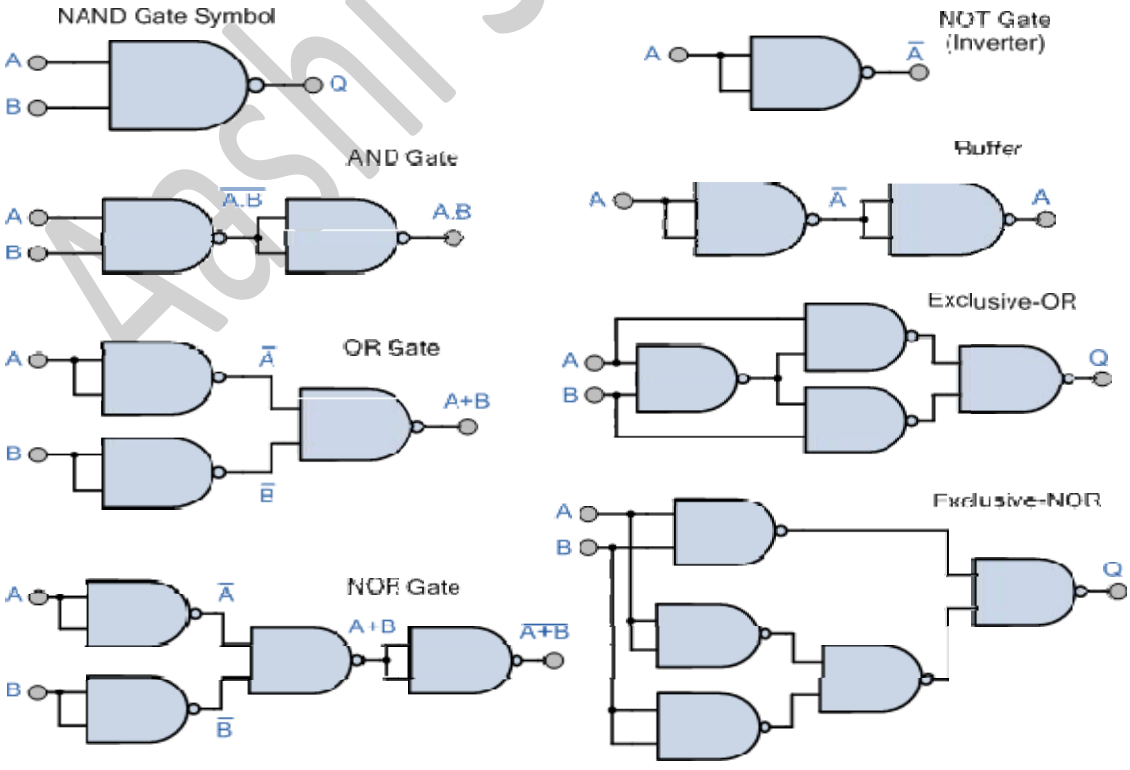
UNIVERSAL LOGIC GATES: The logic gates that are used to produce other logic gates are called universal gates. These gates are: **1. NAND GATE 2. NOR GATE**

One of the main disadvantages of using the complete sets of AND, OR and NOT gates is that to produce any equivalent logic gate or function we require two (or more) different types of logic gate, AND and NOT, or OR and NOT, or all three as shown above. However, we can realize all of the other Boolean functions and gates by using just one single type of universal logic gate, the NAND (NOT AND) or the NOR (NOT OR) gate, thereby reducing the number of different types of logic gates required, and also the cost.

The NAND and NOR gates are the complements of the previous AND and OR functions respectively and are individually a complete set of logic as they can be used to implement any other Boolean function or gate. But as we can construct other logic switching functions using just these gates on their own, they are both called a minimal set of gates. Thus the NAND and the NOR gates are commonly referred to as **Universal Logic Gates**.

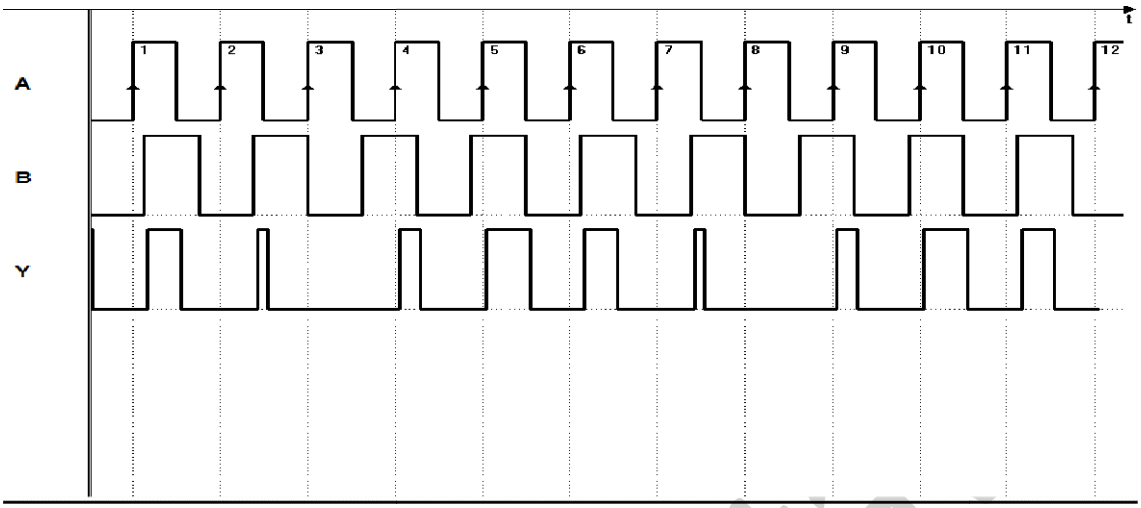
A. Implementation of Logic Functions Using Only NAND Gates.

The 7400 (or the 74LS00 or 74HC00) quad 2-input NAND TTL chip has four indiv dual NAND gates within a single IC package. Thus we can use a single 7400 TTL chip to produce all the Boolean functions from a NOT gate to a NOR gate as shown.



DEED SIMULATED OUTPUT :

1. **NAND to AND gate**



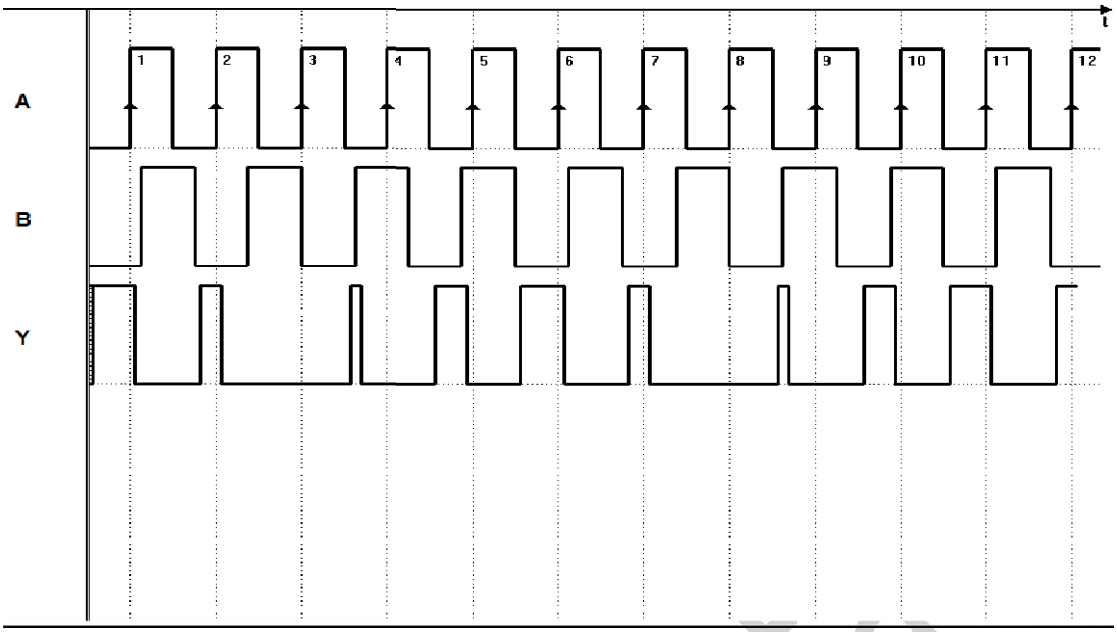
VERILOG CODE:

```
// Developed by: Aashi Srivastava
// TITLE: AND FROM NAND
// Date: 08.10.23, 10:31 IST

module andfromnand (
    out, in1, in2,
);
    input in1, in2;
    output out;
    wire wire1;

    nand n1(wire1, in1, in2);
    nand n2(out, wire1, wire1);

endmodule
```

Test bench:

```
// Developed by Aashi Srivastava
//TITLE: TEST BENCH FOR AND GATE
// Date: 06.10.23, 11:21 IST

module andGate_tb();
    reg in1,in2; //input is declared as reg type
    wire out, wire1 ; //output is declared as net type

  andfromnand s(out,in1,in2, wire1); //calling primtive
  /* primitives are directly instantiated in the test-bench*/

    initial begin
    in1=0;
    in2=0;
    #1 in2=1;
    end

    initial begin //input signal generation
        repeat(20)
        #5 in1=~in1;
        #5 in2=~in2;
    end

    initial begin
        $dumpfile("andfromnand.vcd"); //for gtkwave opening
        $dumpvars(0,andGate_tb); // specifiies to dump all the variables
        $monitor($time,"out=%b in1=%b in2=%b",out,in1,in2); //to monitor
real-time change in variables
        #30 $finish;
    end

endmodule
```


2. NAND to NOR gate



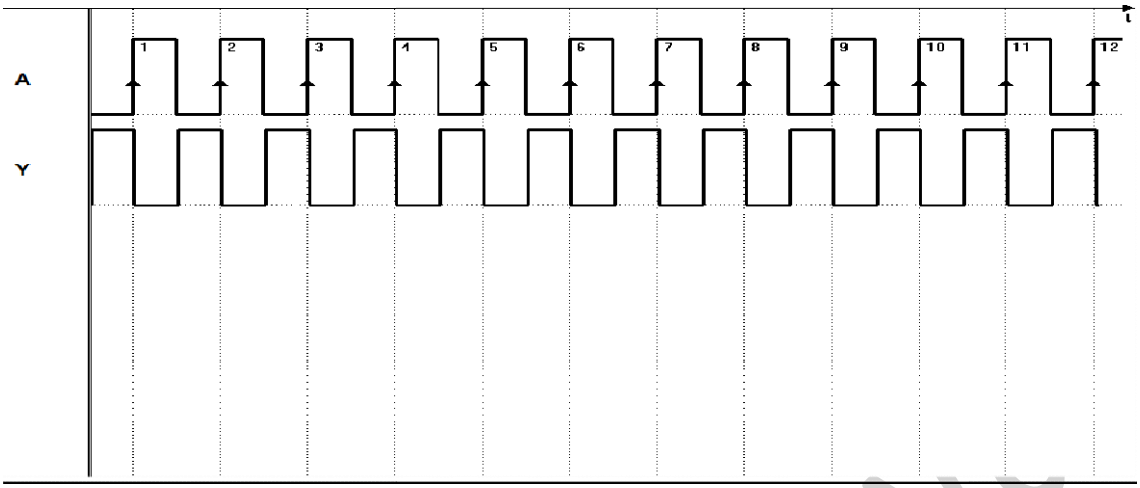
VERILOG CODE:

```
// Developed by: Aashi Srivastava
// TITLE: NOR FROM NAND
// Date: 08.10.23, 10:31 IST

module norfromnand (
    out, in1, in2
);
    input in1, in2;
    output out;
    wire wire1, wire2, wire3;

    nand n1(wire1, in1, in1);
    nand n2(wire2, in2, in2);
    nand n3(wire3, wire1, wire2);
    nand n4(out, wire3, wire3);


endmodule
```

Test bench:

```
// Developed by Aashi Srivastava
//TITLE: TEST BENCH FOR NOR GATE
// Date: 06.10.23, 11:21 IST

module norGate_tb();
    reg in1,in2; //input is declared as reg type
    wire out; //output is declared as net type

 norfromnand s(out,in1,in2); //calling primtive
  /* primitives are directly instantiated in the test-bench*/

    initial begin
    in1=0;
    in2=0;
    #1 in2=1;
    end

    initial begin //input signal generation
        repeat(20)
        #5 in1=~in1;
        #5 in2=~in2;
    end

    initial begin
        $dumpfile("norfromnand.vcd"); //for gtkwave opening
        $dumpvars(0,norGate_tb); // specifiies to dump all the variables
        $monitor($time,"out=%b in1=%b in2=%b",out,in1,in2); //to monitor
real-time change in variables
        #30 $finish;
    end

endmodule
```

3. NAND to NOT gate



VERILOG CODE:

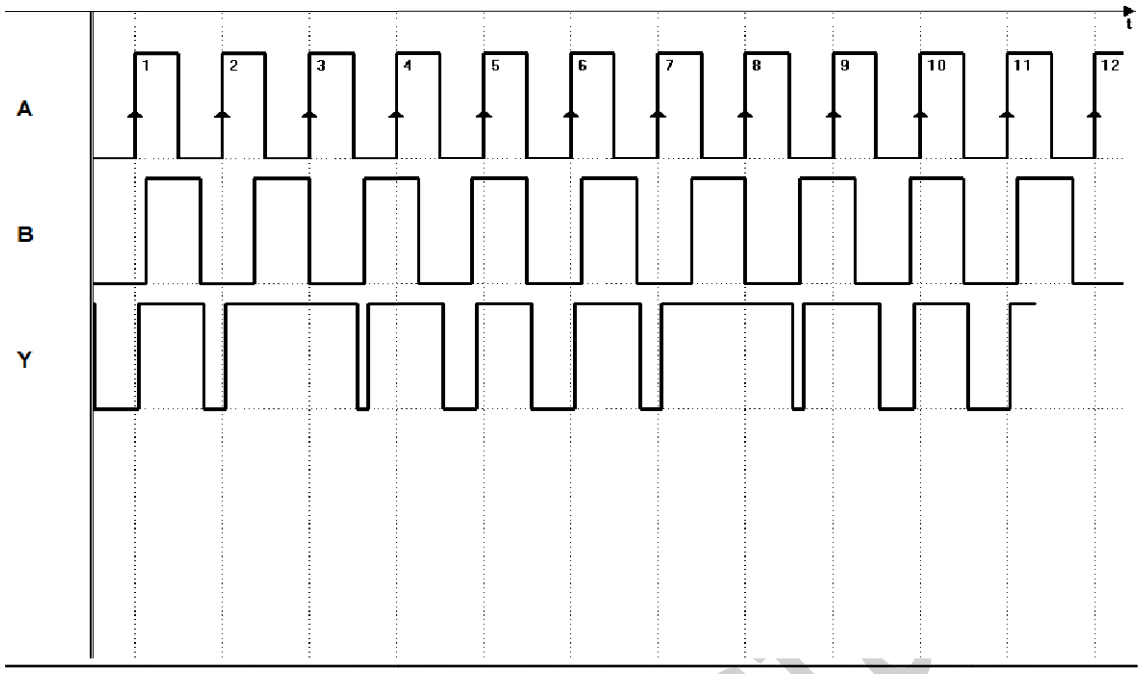
```
// Developed by: Aashi Srivastava
// TITLE: NOT FROM NAND
// Date: 08.10.23, 10:31 IST

module notfromnand (
    out, in1
);
    input in1;
    output out;

    nand n1(out, in1, in1);

endmodule
```

-

Test bench:

```
// Developed by Aashi Srivastava
//TITLE: TEST BENCH FOR NOT GATE
// Date: 06.10.23, 11:21 IST
module notGate_tb (

);
    reg in; //input is declared as reg type
    wire out; //output is declared as net type

    notfromnand N(out,in); // notgate module instantiation

    initial begin
        in=0;
    end

    initial begin //input signal generation
    repeat(20)
     #5 in=~in;

    end

    initial begin
        $dumpfile("notfromnand.vcd"); //for gtkwave opening
        $dumpvars(0,notGate_tb); // specifiies to dump all the variables
        $monitor($time,"out=%b in=%b", out, in); //to monitor real-time
change in variables
        #30 $finish;
    end

endmodule
```

4. **NAND to OR gate**



VERILOG CODE:

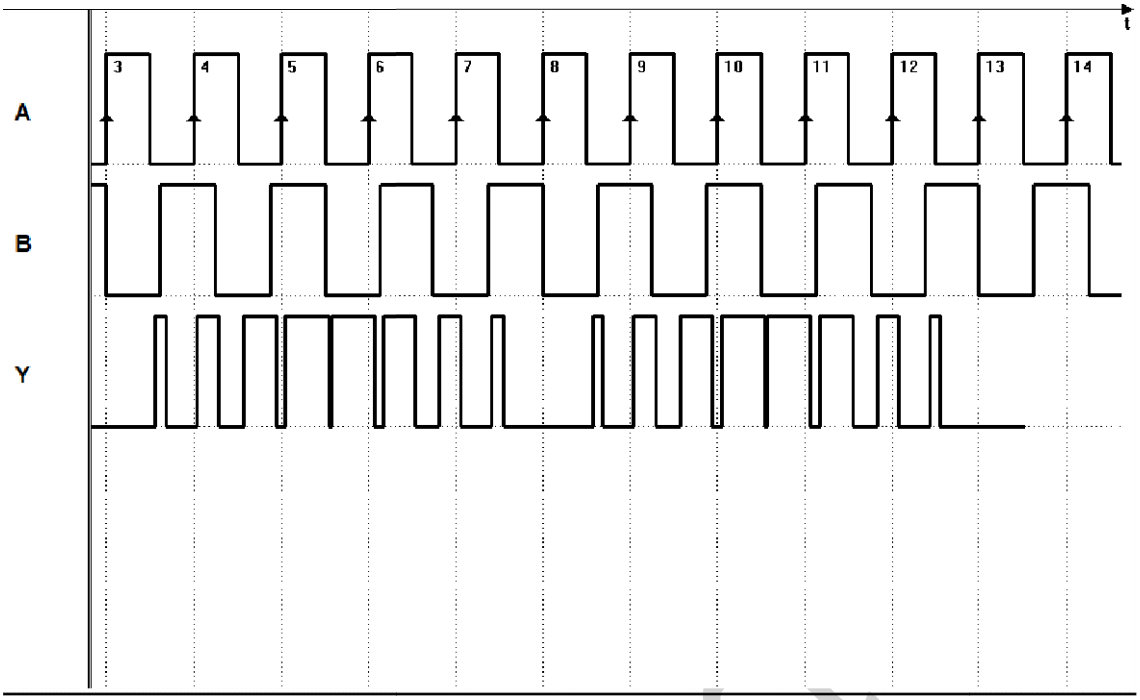
```
// Developed by: Aashi Srivastava
// TITLE: OR FROM NAND
// Date: 08.10.23, 10:31 IST

module orfromnand (
    out, in1, in2
);
    input in1, in2;
    output out;
    wire wire1 , wire2;

    nand n1(wire1, in1, in1);
    nand n2(wire2, in2, in2);
    nand n3(out, wire1, wire2);

endmodule
```

Test bench:

```
// Developed by Aashi Srivastava
//TITLE: TEST BENCH FOR 0R GATE
// Date: 06.10.23, 11:21 IST

module orGate_tb();
    reg in1,in2; //input is declared as reg type
    wire out; //output is declared as net type

  orfromnand s(out,in1,in2); //calling primtive
  /* primitives are directly instantiated in the test-bench*/

    initial begin
    in1=0;
    in2=0;
    #1 in2=1;
    end

    initial begin //input signal generation
        repeat(20)
        #5 in1=~in1;
        #5 in2=~in2;
    end

    initial begin
```

```
        $dumpfile("orfromnand.vcd"); //for gtkwave opening
        $dumpvars(0,orGate_tb); // specifiies to dump all the variables
        $monitor($time,"out=%b in1=%b in2=%b",out,in1,in2); //to monitor
real-time change in variables
        #30 $finish;
    end

endmodule
```


5. NAND to X-NOR gate



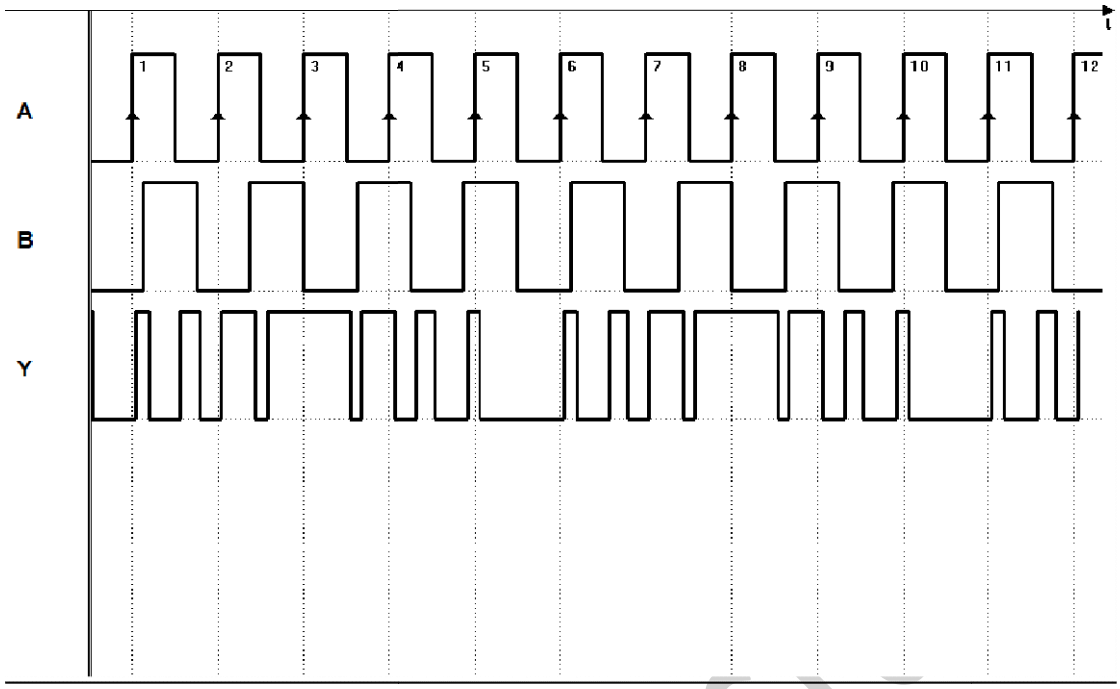
VERILOG CODE:

```
// Developed by: Aashi Srivastava
// TITLE: XNOR FROM NAND
// Date: 08.10.23, 10:31 IST

module xnorfromnand (
    out, in1, in2
);
    input in1, in2;
    output out;
    wire wire1, wire2, wire3, wire4;

    nand n1(wire1, in1, in2);
    nand n2(wire2, wire1, in1);
    nand n3(wire3, wire1, in2);
    nand n4(wire4, wire3, wire2);
    nand n5(out, wire4, wire4);


endmodule
```

Test bench:

```
// Developed by Aashi Srivastava
//TITLE: TEST BENCH FOR XN0R GATE
// Date: 06.10.23, 11:21 IST

module xnorGate_tb();
    reg in1,in2; //input is declared as reg type
    wire out; //output is declared as net type
xnorfromnand s(out,in1,in2); //calling primtive
  /* primitives are directly instantiated in the test-bench*/

    initial begin
    in1=0;
    in2=0;
    #1 in2=1;
    end

    initial begin //input signal generation
        repeat(20)
        #5 in1=~in1;
        #5 in2=~in2;
    end

    initial begin
        $dumpfile("xnorfromnand.vcd"); //for gtkwave opening
        $dumpvars(0,xnorGate_tb); // specifiies to dump all the variables
        $monitor($time,"out=%b in1=%b in2=%b",out,in1,in2); //to monitor
real-time change in variables
        #30 $finish;
    end

endmodule
```

6. NAND to X-OR gate



VERILOG CODE:

```
// Developed by: Aashi Srivastava
// TITLE: XOR FROM NAND
// Date: 08.10.23, 10:31 IST

module xorfromnand (
    out, in1, in2
);
    input in1, in2;
    output out;
    wire wire1, wire2, wire3;

    nand n1(wire1, in1, in2);
    nand n2(wire2, wire1, in1);
    nand n3(wire3, wire1, in2);
    nand n4(out, wire3, wire2);


endmodule
```

Test bench:

```
// Developed by Aashi Srivastava
//TITLE: TEST BENCH FOR X0R GATE
// Date: 06.10.23, 11:21 IST

module xorGate_tb();
    reg in1,in2; //input is declared as reg type
    wire out; //output is declared as net type

  xorfromnand s(out,in1,in2); //calling primtive
  /* primitives are directly instantiated in the test-bench*/

    initial begin
    in1=0;
    in2=0;
    #1 in2=1;
    end

    initial begin //input signal generation
        repeat(20)
        #5 in1=~in1;
        #5 in2=~in2;
    end

    initial begin
        $dumpfile("xorfromnand.vcd"); //for gtkwave opening
        $dumpvars(0,xorGate_tb); // specifiies to dump all the variables
        $monitor($time,"out=%b in1=%b in2=%b",out,in1,in2); //to monitor
real-time change in variables
        #30 $finish;
    end

endmodule
```