

School of Computer Science Engineering and Technology

Course- BTech
Course Code: CSET 105
Year- 2022
Name – Ayushman Pranav
Rollno. – E21CSEU0245
Batch – EB10

Type- Core
Course Name- Digital Logic Design Lab
Semester- Even

Lab Assignment 4.1

Perform the following operations:

- Derive the Boolean expression.
 - Write the truth table for the above expression.
 - Write a Verilog code for each Boolean expression and then test using wave form and compare with truth table whether your circuit produced same output or not?
2. Represent the following gates using only NAND gates
- A. NOT

The screenshot shows the EDA Playground interface. On the left, the 'testbench.vv' file contains Verilog code for a testbench. On the right, the 'design.vv' file contains the Verilog code for a NOT gate. The terminal at the bottom shows the simulation results.

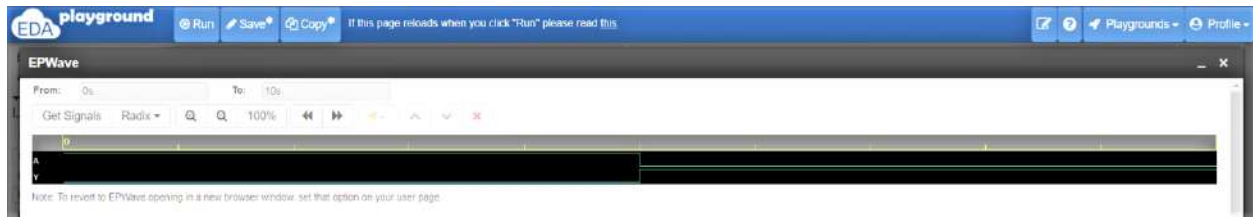
```
testbench.vv
1 module tb_gate;
2   reg A;
3   wire Y;
4   //print
5   //Ayushman Pranav E21CSEU0245
6   gate1 a1(.a(A),.y(Y));
7   initial
8   begin
9     A=1; #5;
10    $display("A Result");
11    $display("%b %b",A,Y);
12    A=0; #5;
13    $display("%b %b",A,Y);
14  end
15
16 initial begin
17   $dumpfile("dump.vcd");
18   $dumpvars(1);
19 end
20
21 endmodule
```

```
design.vv
1 module
2   gate1 (input a, output y);
3   assign y = ~(a&a);
4 endmodule
```

Log

```
[2022-04-15 03:56:08 UTC] iverilog '-wall' design.vv testbench.vv && unbuffer vvp a.out
VCD info: dumpfile dump.vcd opened for output.
A Result
1 0
0 1
Finding VCD file...
./dump.vcd
[2022-04-15 03:56:10 UTC] opening EPWave...
Done
```

School of Computer Science Engineering and Technology



Testbench.sv

```
module tb_gate;
    reg A;
    wire Y;
    //print
    //Ayushman Pranav E21CSEU0245
    gate1 a1(.a(A),.y(Y));
    initial
    begin
        A=1; #5;
        $display("A Result");
        $display("%b %b",A,Y);
        A=0; #5;
        $display("%b %b",A,Y);
    end

    initial begin
        $dumpfile("dump.vcd");
        $dumpvars(1);
    end

endmodule
```

design.sv

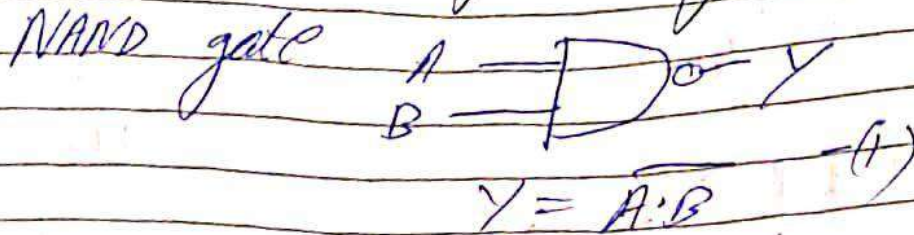
```
module
    gate1 (input a, output y);

    assign y = ~(a&a);

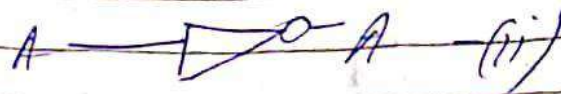
endmodule
```

School of Computer Science Engineering and Technology

1a) Implement Not gate using NAND gate



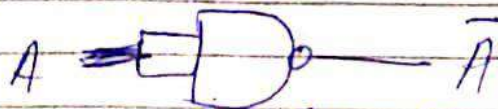
Not gate



Let us $A = B$

put $A = B$ in (i)

$$Y = \overline{A \cdot A} = \overline{A} \quad [A \cdot A = A]$$



A	$Y = \overline{A}$
0	1
1	0

A	B	$B = A$	\overline{AB}	\overline{AA}
1	1	1	0	0
1	0	0	1	0
0	1	1	1	1
0	0	0	1	1

School of Computer Science Engineering and Technology

B. AND

The screenshot shows the EDA Playground interface. On the left, the 'testbench.sv' file contains the following Verilog code:

```
1 module tb_gate;
2   reg A,B;
3   wire Y;
4   //print
5   //Ayushman Pranav E21CSEU0245
6   gate1 a1(.a(A),.b(B),.y(Y));
7   initial
8   begin
9     A=1; B=1; #5;
10    $display("A B Result");
11    $display("%b %b %b",A,B,Y);
12    A=1; B=0; #5;
13    $display("%b %b %b",A,B,Y);
14    A=0; B=1; #5;
15    $display("%b %b %b",A,B,Y);
16    A=0; B=0; #5;
17    $display("%b %b %b",A,B,Y);
18  end
19
20 initial begin
21   $dumpfile("dump.vcd");
22   $dumpvars(1);
23 end
24
25 endmodule
```

On the right, the 'design.sv' file contains the following Verilog code:

```
1 module
2   gate1 (input a,b, output y);
3   assign y = ~(~a&b);
4 endmodule
```

Below the code editors, the simulation output is displayed:

```
[2022-04-15 04:46:11 UTC] iverilog '-wall' design.sv testbench.sv && unbuffer vvp a.out
VCD info: dumpfile dump.vcd opened for output.
A B Result
1 1 1
1 0 0
0 1 0
0 0 0
Finding VCD file...
./dump.vcd
[2022-04-15 04:46:13 UTC] Opening EPWave...
Done
```

The screenshot shows the EPWave window displaying the timing diagram for the simulation. The signals A, B, and Y are shown. A is high from 0 to 5 ns, then low. B is high from 0 to 5 ns, then low. Y is high from 0 to 5 ns, then low. The timing diagram shows the output Y following the AND operation of inputs A and B.

Testbench.sv

```
module tb_gate;

  reg A,B;

  wire Y;

  //print

  //Ayushman Pranav E21CSEU0245

  gate1 a1(.a(A),.b(B),.y(Y));

  initial
```

School of Computer Science Engineering and Technology

begin

A=1; B=1; #5;

\$display("A B Result");

\$display("%b %b %b",A,B,Y);

A=1; B=0; #5;

\$display("%b %b %b",A,B,Y);

A=0; B=1; #5;

\$display("%b %b %b",A,B,Y);

A=0; B=0; #5;

\$display("%b %b %b",A,B,Y);

end

initial begin

\$dumpfile("dump.vcd");

\$dumpvars(1);

end

endmodule

Design.sv

module

gate1 (input a,b, output y);

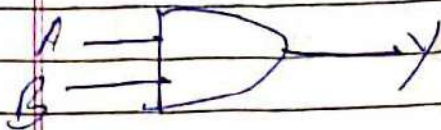
assign y = ~(~(a&b));

endmodule

2) AND gate using NAND gate

And gate

And gate using
NAND gate



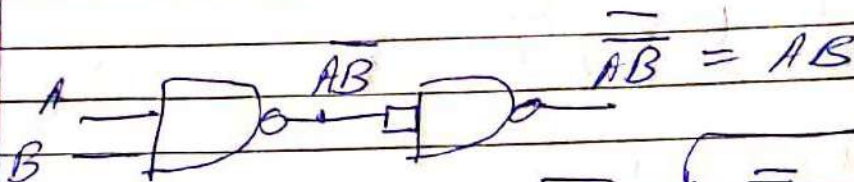
$$Y = AB$$

Expression $Y = AB$ using double inversion principle

A	B	AB
1	1	1
1	0	0
0	1	0
0	0	0

$$\overline{\overline{Y}} = \overline{\overline{AB}} \left[\because \overline{\overline{A}} = A \right]$$

$$Y = \overline{\overline{AB}}$$



A	B	\overline{AB}
1	1	0
1	0	1
0	1	1
0	0	1

$\overline{\overline{AB}}$	AB
1	1
0	0
0	0
0	0

↳ Logically Equivalent

School of Computer Science Engineering and Technology

$$A + B$$
$$\overline{AB} = A + B$$

C. OR

EDA playground

Run Save* Copy* "Run" please read this. Playgrounds Profile

testbench.v

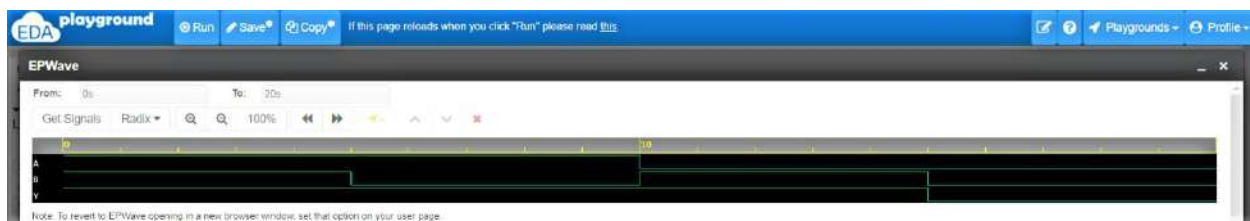
```
1 module tb_gate:
2   reg A,B;
3   wire Y;
4   //print
5   //Ayushman Pranav E21CSEU0245
6   gate1 a1(.a(A)..b(B)..y(Y));
7   initial
8     begin
9       A=1; B=1; #5;
10      $display("A B Result");
11      $display("%b %b %b",A,B,Y);
12      A=1; B=0; #5;
13      $display("%b %b %b",A,B,Y);
14      A=0; B=1; #5;
15      $display("%b %b %b",A,B,Y);
16      A=0; B=0; #5;
17      $display("%b %b %b",A,B,Y);
18    end
19
20  initial begin
21    $dumpfile("dump.vcd");
22    $dumpvars(1);
23  end
24
25 endmodule
```

design.v

```
1 module
2   gate1 (input a,b, output y);
3   assign y = ~(a&b);
4 endmodule
```

Log Share

[2022-04-15 05:05:56 UTC] iverilog '-Wall' design.v testbench.v && unbuffer vvp a.out
VCD info: dumpfile dump.vcd opened for output.
A B Result
1 1 1
1 0 1
0 1 1
0 0 0
Finding VCD file...
./dump.vcd
[2022-04-15 05:05:58 UTC] opening EPWave...
Done



School of Computer Science Engineering and Technology

Testbench.sv

```
module tb_gate;

    reg A,B;

    wire Y;

    //print
    //Ayushman Pranav E21CSEU0245

    gate1 a1(.a(A),.b(B),.y(Y));

    initial
    begin
        A=1; B=1; #5;
        $display("A B   Result");
        $display("%b %b   %b",A,B,Y);
        A=1; B=0; #5;
        $display("%b %b   %b",A,B,Y);
        A=0; B=1; #5;
        $display("%b %b   %b",A,B,Y);
        A=0; B=0; #5;
        $display("%b %b   %b",A,B,Y);
    end

    initial begin
        $dumpfile("dump.vcd");
        $dumpvars(1);
    end

endmodule
```

Design.sv

School of Computer Science Engineering and Technology

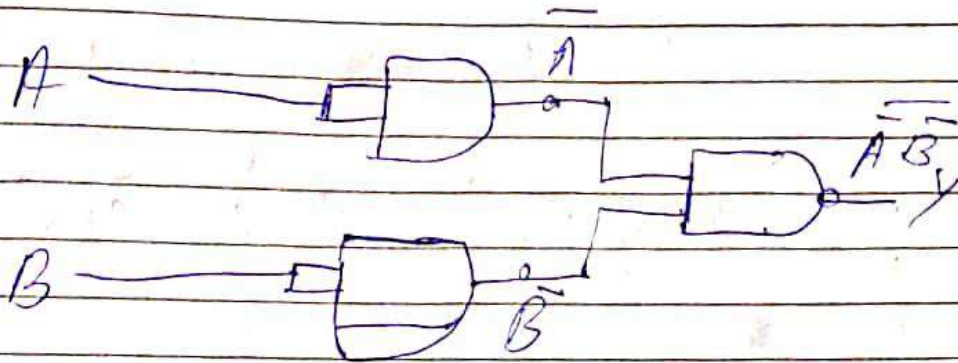
module

gate1 (input a,b, output y);

assign y = ~(~a&~b);

endmodule

3) or gate using NAND
or gate



$$Y = \overline{\overline{A} \overline{B}} \\ = A + B \text{ (DeMorgan)}$$

A	B	\overline{A}	\overline{B}	$\overline{A} \overline{B}$	$\overline{\overline{A} \overline{B}}$	$A + B$
1	1	0	0	0	1	1
1	0	0	1	0	1	1
0	1	1	0	0	1	1
0	0	1	1	1	0	0

School of Computer Science Engineering and Technology

D. NOR

EDA playground

Run Save* Copy* ? Playgrounds Profile

testbench.sv

SV/Verilog Testbench

```
1 module tb_gate;
2   reg A,B;
3   wire Y;
4   //print
5   //Ayushman Pranav E21CSEU0245
6   gate1 a1(.a(A),.b(B),.y(Y));
7   initial
8   begin
9     A=1; B=1; #5;
10    $display("A B Result");
11    $display("%b %b %b",A,B,Y);
12    A=1; B=0; #5;
13    $display("%b %b %b",A,B,Y);
14    A=0; B=1; #5;
15    $display("%b %b %b",A,B,Y);
16    A=0; B=0; #5;
17    $display("%b %b %b",A,B,Y);
18  end
19
20  initial begin
21    $dumpfile("dump.vcd");
22    $dumpvars(1);
23  end
24
25 endmodule
```

design.sv

SV/Verilog Design

```
1 module
2   gate1 (input a,b, output y);
3   assign y = ~(~a&~b);
4 endmodule
```

Log Share

[2022-04-15 05:24:27 UTC] iverilog '-wall' design.sv testbench.sv && unbuffer vvp a.out
VCD info: dumpfile dump.vcd opened for output.
A B Result
1 1 0
1 0 0
0 1 0
0 0 1
Finding VCD file...
./dump.vcd
[2022-04-15 05:24:28 UTC] Opening EPWave...
Done

EDA playground


Run Save* Copy* If this page reloads when you click "Run" please read this

Playgrounds Profile

EPWave

From: 0s To: 20s

Get Signals Radix 100% 100%



Note: To revert to EPWave opening in a new browser window, set that option on your user page.

School of Computer Science Engineering and Technology

Testbench.sv

```
module tb_gate;

    reg A,B;

    wire Y;

//print
//Ayushman Pranav E21CSEU0245

    gate1 a1(.a(A),.b(B),.y(Y));

    initial

    begin

        A=1; B=1; #5;

        $display("A B   Result");

        $display("%b %b   %b",A,B,Y);

        A=1; B=0; #5;

        $display("%b %b   %b",A,B,Y);

        A=0; B=1; #5;

        $display("%b %b   %b",A,B,Y);

        A=0; B=0; #5;

        $display("%b %b   %b",A,B,Y);

    end

    initial begin

        $dumpfile("dump.vcd");

        $dumpvars(1);

    end

endmodule
```

Design.sv

School of Computer Science Engineering and Technology

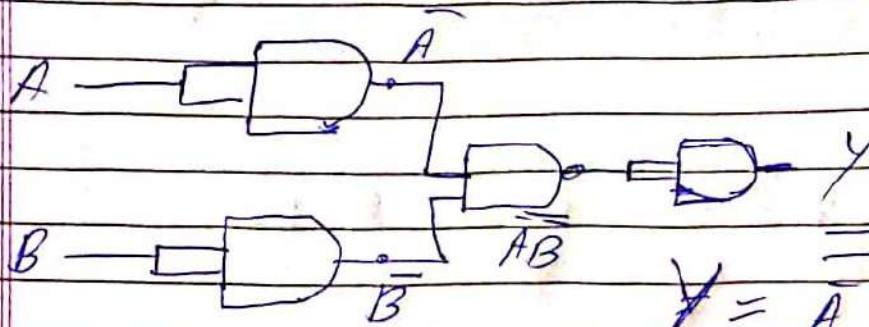
module

gate1 (input a,b, output y);

assign y = ~(~(~a&~b));

endmodule

4) Nor gate using ~~NAND~~ NAND



$$Y = \overline{\overline{A} \overline{B}}$$
$$= \overline{A \cdot B}$$

$$= \overline{A + B}$$

(DeMorgan's Law)

A	B	\overline{A}	\overline{B}	$\overline{A} \cdot \overline{B}$	$\overline{A + B}$
1	1	0	0	0	0
1	0	0	1	0	0
0	1	1	0	0	0
0	0	1	1	1	1

School of Computer Science Engineering and Technology

E. XOR

The screenshot shows the EDA Playground interface with two files: `testbench.sv` and `design.sv`.

testbench.sv (SV/Verilog Testbench):

```
1 module tb_gate;
2   reg A,B;
3   wire Y;
4   //print
5   //Ayushman Praniav E21CSEU0245
6   gate1 a1(.a(A)..b(B)..y(Y));
7   initial
8   begin
9     A=1; B=1; #5;
10    $display("A B Result");
11    $display("%b %b %b",A,B,Y);
12    A=1; B=0; #5;
13    $display("%b %b %b",A,B,Y);
14    A=0; B=1; #5;
15    $display("%b %b %b",A,B,Y);
16    A=0; B=0; #5;
17    $display("%b %b %b",A,B,Y);
18  end
19
20 initial begin
21   $dumpfile("dump.vcd");
22   $dumpvars(1);
23 end
24
25 endmodule
```

design.sv (SV/Verilog Design):

```
1 module
2   gate1 (input a,b, output y):
3     assign y = ~(a&(~(a&b)))&~(b&~(a&b));
4 endmodule
```

Log:

```
[2022-04-15 06:01:42 UTC] iverilog '-wall' design.sv testbench.sv && unbuffer vvp a.out
VCD info: dumpfile dump.vcd opened for output.
A B Result
1 1 0
1 0 1
0 1 1
0 0 0
Finding VCD file...
./dump.vcd
[2022-04-15 06:01:43 UTC] Opening EPWave...
Done
```



Testbench.sv

```
module tb_gate;
```

```
  reg A,B;
```

```
  wire Y;
```

```
//print
```

School of Computer Science Engineering and Technology

//Ayushman Pranav E21CSEU0245

```
gate1 a1(.a(A),.b(B),.y(Y));
```

```
initial
```

```
begin
```

```
    A=1; B=1; #5;
```

```
    $display("A B   Result");
```

```
    $display("%b %b   %b",A,B,Y);
```

```
    A=1; B=0; #5;
```

```
    $display("%b %b   %b",A,B,Y);
```

```
    A=0; B=1; #5;
```

```
    $display("%b %b   %b",A,B,Y);
```

```
    A=0; B=0; #5;
```

```
    $display("%b %b   %b",A,B,Y);
```

```
end
```

```
initial begin
```

```
    $dumpfile("dump.vcd");
```

```
    $dumpvars(1);
```

```
end
```

```
endmodule
```

Design.sv

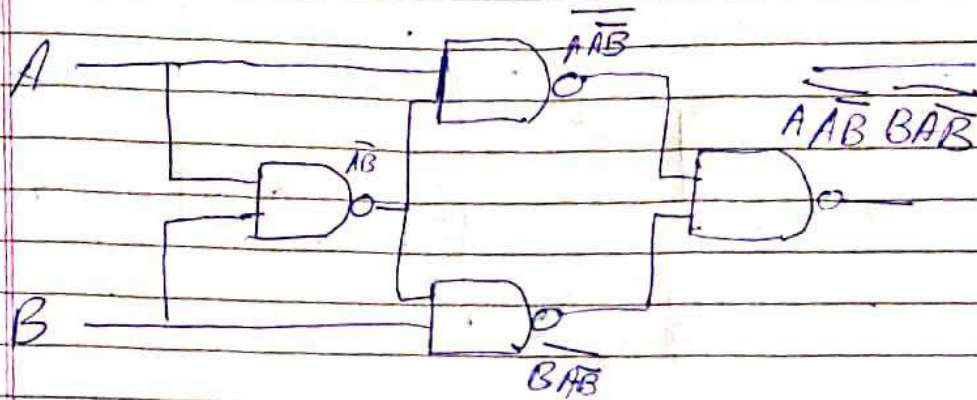
```
module
```

```
    gate1 (input a,b, output y);
```

```
    assign y = ~(~(a&(~(a&b)))&~(b&~(a&b)));
```

```
endmodule
```

5) XOR gate using NAND



$$\begin{aligned}
 F(ABC) &= \overline{A \cdot \overline{A} \cdot B} \cdot \overline{B \cdot \overline{A} \cdot A} \\
 &= A \cdot \overline{A} \cdot B + B \cdot \overline{A} \cdot A \\
 &= A(\overline{A} + \overline{B}) + B(\overline{A} + \overline{B}) \\
 &= \overline{A}A + \overline{A}B + \overline{B}A + \overline{B}B \\
 &= \overline{A}B + A\overline{B} \\
 &= A \oplus B
 \end{aligned}$$

$\overline{A}B + B\overline{A}$	A	B	\overline{A}	\overline{B}	$\overline{A}B$	$A\overline{B}$	$B\overline{A}$	$\overline{A}B + B\overline{A}$	F(ABC)
0	1	1	0	0	0	0	0	1	1
1	1	0	0	1	1	1	0	0	0
1	0	1	1	0	1	0	1	0	0
0	0	0	1	1	1	0	0	1	1

→ logically equivalent

School of Computer Science Engineering and Technology

F. XNOR

The screenshot shows the EDA Playground interface with two code editors. The left editor, titled 'testbench.sv', contains the following Verilog code:

```
1 module tb_gate;
2   reg A,B;
3   wire Y;
4   //print
5   //Ayushman Pranav E21CSEU0245
6   gate1 a1(.a(A),.b(B),.y(Y));
7   initial
8   begin
9     A=1; B=1; #5;
10    $display("A: %b B: %b Result: %b", A, B, Y);
11    $display("A: %b B: %b Result: %b", A, B, Y);
12    A=1; B=0; #5;
13    $display("A: %b B: %b Result: %b", A, B, Y);
14    A=0; B=1; #5;
15    $display("A: %b B: %b Result: %b", A, B, Y);
16    A=0; B=0; #5;
17    $display("A: %b B: %b Result: %b", A, B, Y);
18  end
19
20 initial begin
21   $dumpfile("dump.vcd");
22   $dumpvars(1);
23 end
24
25 endmodule
```

The right editor, titled 'design.sv', contains the following Verilog code:

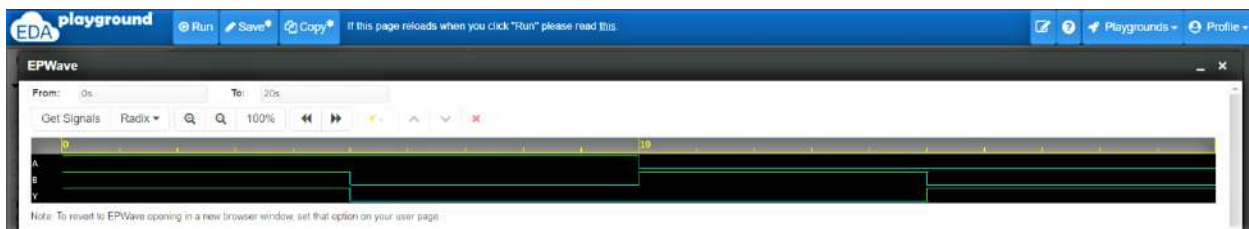
```
1 module
2   gate1 (input a,b, output y);
3   assign y = ~(~(a&(a&b)))&~(b&~(a&b));
4 endmodule
```

Below the code editors, the simulation output is displayed. It shows the command used to run the simulation and the resulting VCD file information.

```
[2022-04-15 06:10:04 UTC] iverilog '-Wall' design.sv testbench.sv && unbuffer vvp a.out
VCD info: dumpfile dump.vcd opened for output.
```

A	B	Result
1	1	1
1	0	0
0	1	0
0	0	1

The output table shows the results of the XNOR gate for all possible combinations of inputs A and B. The results are 1 for (1,1) and (0,0), and 0 for (1,0) and (0,1).



Testbench.sv

```
module tb_gate;

  reg A,B;

  wire Y;

  //print

  //Ayushman Pranav E21CSEU0245

  gate1 a1(.a(A),.b(B),.y(Y));
```

School of Computer Science Engineering and Technology

initial

begin

A=1; B=1; #5;

\$display("A B Result");

\$display("%b %b %b",A,B,Y);

A=1; B=0; #5;

\$display("%b %b %b",A,B,Y);

A=0; B=1; #5;

\$display("%b %b %b",A,B,Y);

A=0; B=0; #5;

\$display("%b %b %b",A,B,Y);

end

initial begin

\$dumpfile("dump.vcd");

\$dumpvars(1);

end

endmodule

Design.sv

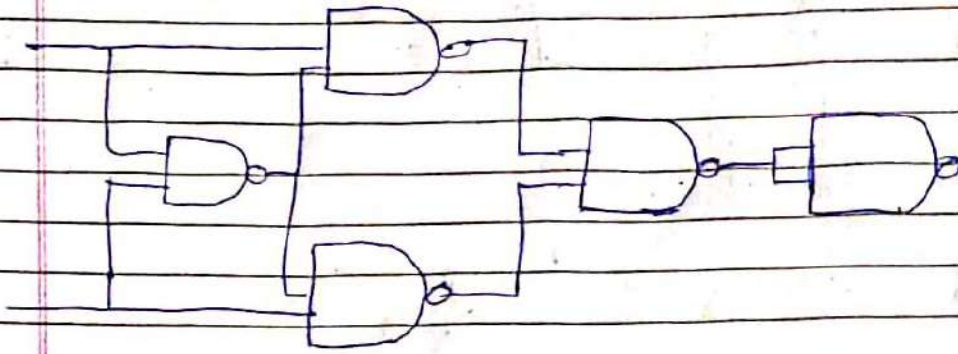
module

gate1 (input a,b, output y);

assign y = ~(~(a&(~(a&b)))&~(b&~(a&b)));

endmodule

6) XNOR using NAND



$$\begin{aligned}
 &= \overline{\overline{A} \overline{B} \overline{B} \overline{A}} \\
 &= \overline{A \cdot \overline{B} + B \overline{A}} \\
 &= \overline{A(\overline{A} + \overline{B}) + B(\overline{A} + \overline{B})} \\
 &= \overline{A\overline{A} + A\overline{B} + \overline{A}B + B\overline{B}} \\
 &= \overline{A\overline{B} + \overline{A}B} = A \oplus B \\
 &= \overline{A\overline{B} \cdot \overline{A}B} \\
 &= \overline{(A + \overline{B})(\overline{A} + B)} \\
 &= \overline{A\overline{A} + A\overline{B} + \overline{A}B + \overline{B}B} \\
 &= \overline{A\overline{B} + \overline{A}B} \\
 &= A \odot B
 \end{aligned}$$

						$A \odot B$	
A	B	\overline{A}	\overline{B}	$A\overline{B}$	$\overline{A}B$	$A\overline{B} + \overline{A}B$	$A \oplus B$
1	1	0	0	1	0	1	1
1	0	0	1	0	0	0	0
0	1	1	0	0	0	0	0
0	0	1	1	0	1	1	1

School of Computer Science Engineering and Technology

3. Write a Verilog code to verify Absorption Law and then test using wave form and compare with truth table whether your circuit produced same output or not?

The screenshot displays the EDA Playground interface. The left pane shows the testbench code (`testbench.v`) and the right pane shows the design code (`design.v`).

testbench.v (SV/Verilog Testbench):

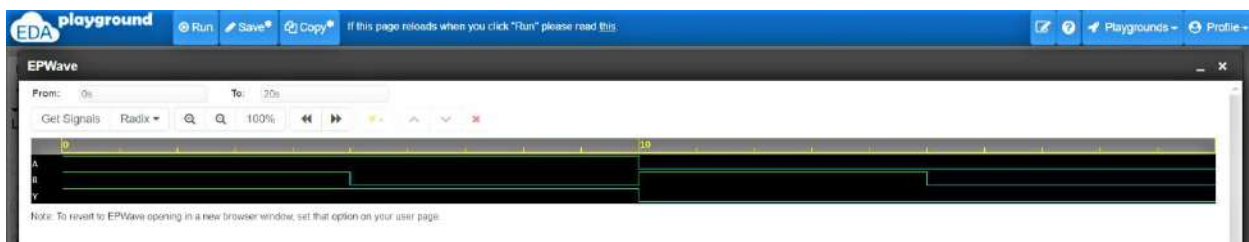
```
1 module tb_gate;
2   reg A,B;
3   wire Y;
4   //print
5   //Ayushman Pranav E21CSEU0245
6   gate1 a1(.a(A),.b(B),.y(Y));
7   initial
8   begin
9     A=1; B=1; #5;
10    $display("A B Result");
11    $display("%b %b %b",A,B,Y);
12    A=1; B=0; #5;
13    $display("%b %b %b",A,B,Y);
14    A=0; B=1; #5;
15    $display("%b %b %b",A,B,Y);
16    A=0; B=0; #5;
17    $display("%b %b %b",A,B,Y);
18  end
19
20  initial begin
21    $dumpfile("dump.vcd");
22    $dumpvars(1);
23  end
24
25 endmodule
```

design.v (SV/Verilog Design):

```
1 module
2   gate1 (input a,b, output y);
3   assign y = a|(a&b);
4 endmodule
```

Log:

```
[2022-04-15 06:28:25 UTC] iverilog '-wall' design.v testbench.v && unbuffer vvp a.out
VCD info: dumpfile dump.vcd opened for output.
A B Result
1 1 1
1 0 1
0 1 0
0 0 0
Finding VCD file...
./dump.vcd
[2022-04-15 06:28:28 UTC] Opening EPWave...
Done
```



Testbench.v

```
module tb_gate;
```

School of Computer Science Engineering and Technology

```
reg A,B;

wire Y;

//print
//Ayushman Pranav E21CSEU0245

gate1 a1(.a(A),.b(B),.y(Y));

initial

begin

    A=1; B=1; #5;
    $display("A B   Result");
    $display("%b %b   %b",A,B,Y);
    A=1; B=0; #5;
    $display("%b %b   %b",A,B,Y);
    A=0; B=1; #5;
    $display("%b %b   %b",A,B,Y);
    A=0; B=0; #5;
    $display("%b %b   %b",A,B,Y);

end

initial begin

    $dumpfile("dump.vcd");
    $dumpvars(1);

end

endmodule
```

School of Computer Science Engineering and Technology

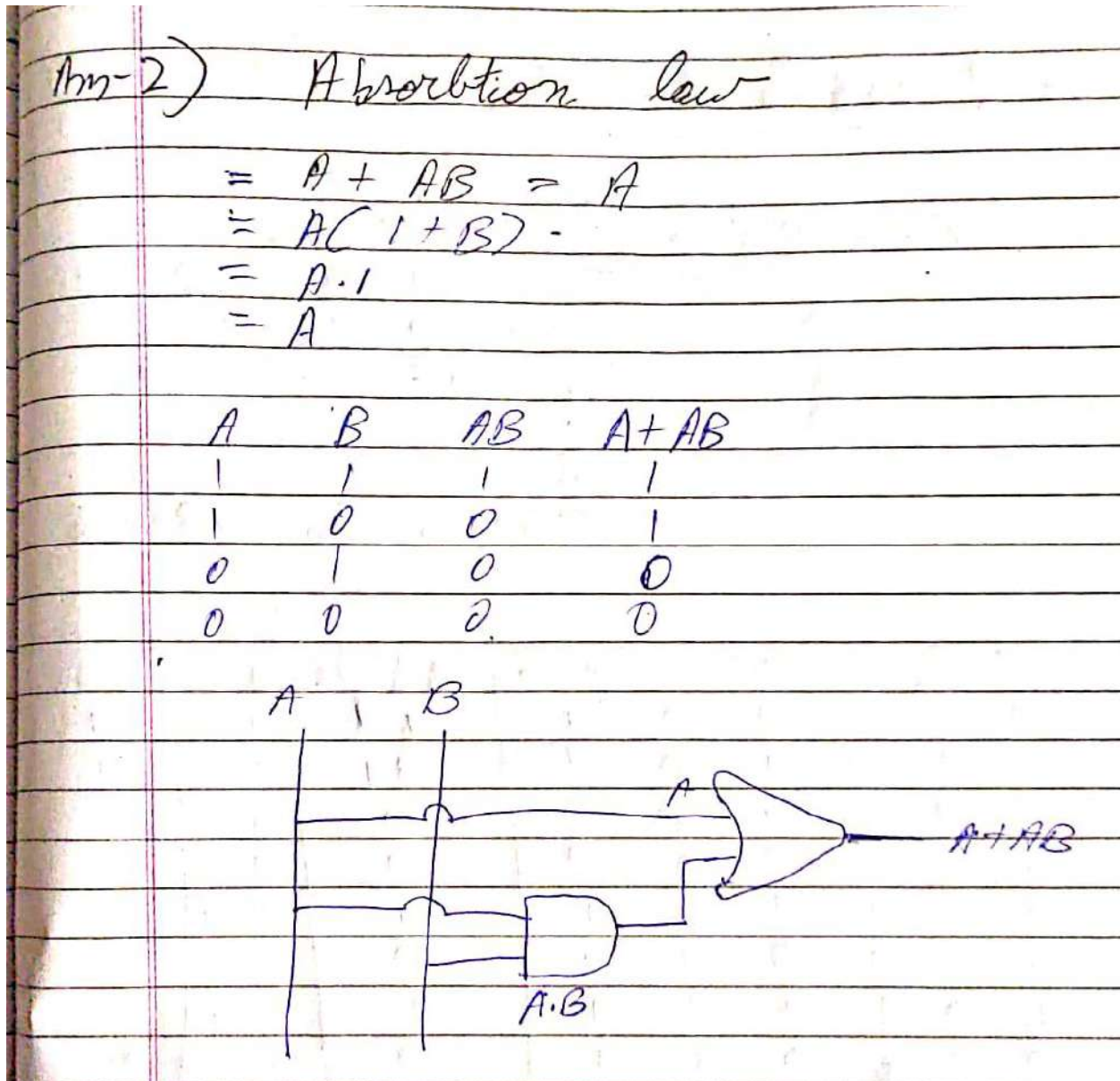
Design.sv

module

gate1 (input a,b, output y);


assign y = a|(a&b);

endmodule





4. Write a Verilog code to verify Transposition Law and then test using wave form and compare with truth table whether your circuit produced same output or not?

School of Computer Science Engineering and Technology

 **playground**

Run Save Copy

  Playgrounds Profile

testbench.sv

design.sv

```
1 module tb_gate():
2   reg A,B,C;
3   wire Y;
4   //print
5   //Ayushman Pranav E21cseu0245
6   gate1 a1(.a(A)..b(B)..c(C)..y(Y));
7   initial begin
8     A=1;B=1;C=1;#5;
9     $display("A B C Result");
10    $display("%b %b %b %b",A,B,C,Y);
11    A=1;B=1;C=0;#5;
12    $display("%b %b %b %b",A,B,C,Y);
13    A=1;B=0;C=1;#5;
14    $display("%b %b %b %b",A,B,C,Y);
15    A=1;B=0;C=0;#5;
16    $display("%b %b %b %b",A,B,C,Y);
17    A=0;B=1;C=1;#5;
18    $display("%b %b %b %b",A,B,C,Y);
19    A=0;B=1;C=0;#5;
20    $display("%b %b %b %b",A,B,C,Y);
21    A=0;B=0;C=1;#5;
22    $display("%b %b %b %b",A,B,C,Y);
23    A=0;B=0;C=0;#5;
24    $display("%b %b %b %b",A,B,C,Y);
25
26  end
27  // EP wave
28  initial begin
29    $dumpfile("dump.vcd");
30    $dumpvars(1);
31  end
32 end
33 endmodule
27
```

```
1 module
2   gate1 (input a,b,c, output y);
3   assign y = (a&b)|(~a&c);
4 endmodule
```

Log

Share

[2022-04-15 13:04:18 UTC] iverilog '-wall' design.sv testbench.sv && unbuffer vvp a.out

VCD info: dumpfile dump.vcd opened for output.

A B C Result

1 1 1 1

1 1 0 1

1 0 1 0

1 0 0 0

0 1 1 1

0 1 0 0

0 0 1 1

0 0 0 0

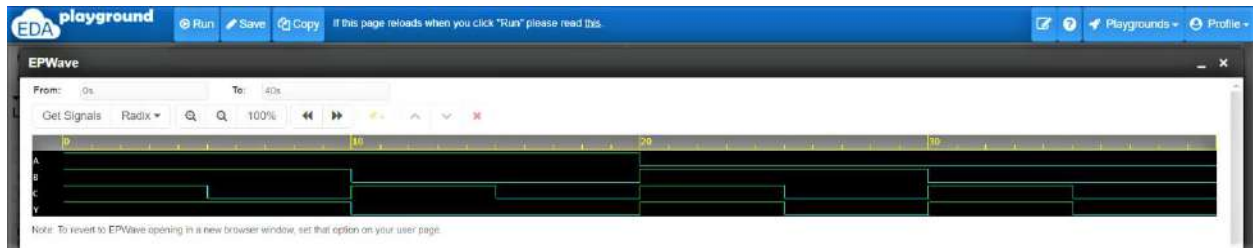
Finding VCD file...

./dump.vcd

[2022-04-15 13:04:20 UTC] Opening EPWave...

Done

School of Computer Science Engineering and Technology



Testbench.sv

```
module tb_gate();

    reg A,B,C;

    wire Y;

    //print

    //Ayushman Pranav E21cseu0245

    gate1 a1(.a(A),.b(B),.c(C),.y(Y));

    initial begin

        A=1;B=1;C=1;#5;

        $display("A B C Result");

        $display("%b %b %b %b",A,B,C,Y);

        A=1;B=1;C=0;#5;

        $display("%b %b %b %b",A,B,C,Y);

        A=1;B=0;C=1;#5;

        $display("%b %b %b %b",A,B,C,Y);

        A=1;B=0;C=0;#5;

        $display("%b %b %b %b",A,B,C,Y);

        A=0;B=1;C=1;#5;

        $display("%b %b %b %b",A,B,C,Y);

        A=0;B=1;C=0;#5;

        $display("%b %b %b %b",A,B,C,Y);

        A=0;B=0;C=1;#5;

        $display("%b %b %b %b",A,B,C,Y);
```

School of Computer Science Engineering and Technology

```
A=0;B=0;C=0;#5;
```

```
$display("%b %b %b %b",A,B,C,Y);
```

```
end
```

```
// EP wave
```

```
initial begin
```

```
    $dumpfile("dump.vcd");
```

```
    $dumpvars(1);
```

```
end
```

```
endmodule
```

Design.sv

```
module
```

```
    gate1 (input a,b,c, output y);
```

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

```
endmodule
```

Ans-3) Transposition Law

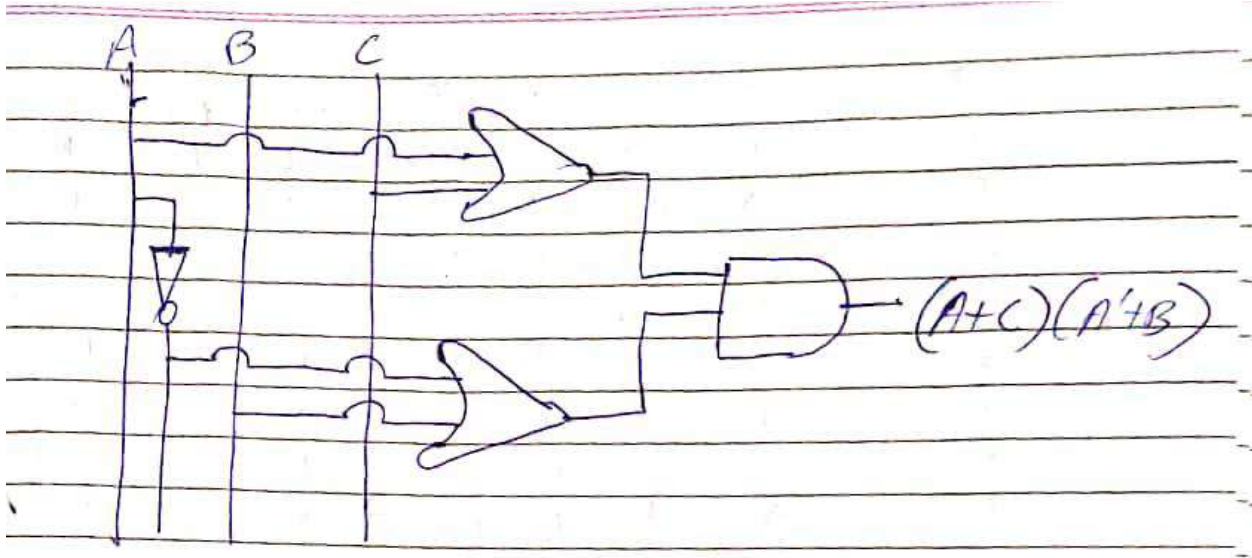
$$\begin{aligned}
 AB + A'C &= (A+C)(A'+B) \\
 &= AA' + AB + CA' + CB \\
 &= AB + CA' + CB \\
 &= AB + BC + A'C
 \end{aligned}$$

Consensus Law = $AB + A'C$

A	B	C	AC	A'	A'B	(AC)(A'B)	AB	A'C	AB+A'C
1	1	1	1	0	1	1	1	0	1
1	1	0	1	0	1	1	1	0	1
1	0	1	1	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0
0	1	1	1	1	1	1	0	1	1
0	1	0	0	1	1	0	0	0	0
0	0	1	1	1	1	1	0	1	1
0	0	0	0	1	1	0	0	0	0

→ logically equivalent

School of Computer Science Engineering and Technology



5. Write a Verilog code to verify Consensus Law and then test using wave form and compare with truth table whether your circuit produced same output or not?

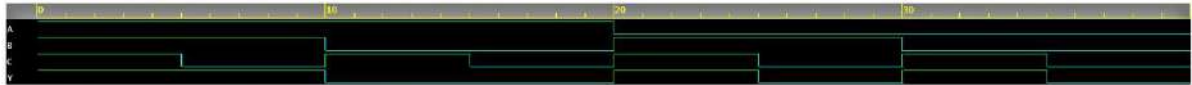
```
EDA playground Run Save Copy Playgrounds Profile
testbench.sv design.sv
1 module tb_gate():
2   reg A,B,C;
3   wire Y;
4   //print
5   //Ayushman Pranav E21cseu0245
6   gate1 al(.a(A),.b(B),.c(C),.y(Y));
7   initial begin
8     A=1;B=1;C=1;#5;
9     $display("A B C Result");
10    $display("%b %b %b %b",A,B,C,Y);
11    A=1;B=1;C=0;#5;
12    $display("%b %b %b %b",A,B,C,Y);
13    A=1;B=0;C=1;#5;
14    $display("%b %b %b %b",A,B,C,Y);
15    A=1;B=0;C=0;#5;
16    $display("%b %b %b %b",A,B,C,Y);
17    A=0;B=1;C=1;#5;
18    $display("%b %b %b %b",A,B,C,Y);
19    A=0;B=1;C=0;#5;
20    $display("%b %b %b %b",A,B,C,Y);
21    A=0;B=0;C=1;#5;
22    $display("%b %b %b %b",A,B,C,Y);
23    A=0;B=0;C=0;#5;
24    $display("%b %b %b %b",A,B,C,Y);
25
26  end
27  // EP wave
28  initial begin
29    $dumpfile("dump.vcd");
30    $dumpvars(1);
31  end
32 end
33 endmodule
1 module
2   gate1 (input a,b,c, output y);
3   assign y = (a&b)|(~a&c);
4 endmodule
```

School of Computer Science Engineering and Technology

[Log](#) [Share](#)

```
[2022-04-15 13:04:18 UTC] iverilog '-wall' design.sv testbench.sv && unbuffer vvp a.out
VCD info: dumpfile dump.vcd opened for output.
A B C Result
1 1 1 1
1 1 0 1
1 0 1 0
1 0 0 0
0 1 1 1
0 1 0 0
0 0 1 1
0 0 0 0
Finding VCD file...
./dump.vcd
[2022-04-15 13:04:20 UTC] Opening EPWave...
Done
```

EDA playground [Run](#) [Save](#) [Copy](#) If this page reloads when you click "Run" please read this. [Playgrounds](#) [Profile](#)

EPWave
From: 0s To: 40s
Get Signals Radix 100%

Note: To revert to EPWave opening in a new browser window, set that option on your user page.

Testbench.sv

```
module tb_gate();

  reg A,B,C;

  wire Y;

  //print

  //Ayushman Pranav E21cseu0245

  gate1 a1(.a(A),.b(B),.c(C),.y(Y));

  initial begin

    A=1;B=1;C=1;#5;

    $display("A B C Result");
```


School of Computer Science Engineering and Technology

```
$display("% b % b % b  % b",A,B,C,Y);  
A=1;B=1;C=0;#5;  
$display("% b % b % b  % b",A,B,C,Y);  
A=1;B=0;C=1;#5;  
$display("% b % b % b  % b",A,B,C,Y);  
A=1;B=0;C=0;#5;  
$display("% b % b % b  % b",A,B,C,Y);  
A=0;B=1;C=1;#5;  
$display("% b % b % b  % b",A,B,C,Y);  
A=0;B=1;C=0;#5;  
$display("% b % b % b  % b",A,B,C,Y);  
A=0;B=0;C=1;#5;  
$display("% b % b % b  % b",A,B,C,Y);  
A=0;B=0;C=0;#5;  
$display("% b % b % b  % b",A,B,C,Y);
```

```
end  
// EP wave  
initial begin  
    $dumpfile("dump.vcd");  
    $dumpvars(1);  
end  
endmodule
```

Design.sv

```
module  
    gate1 (input a,b,c, output y);
```

School of Computer Science Engineering and Technology

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

endmodule

Ans-4) Consensus law

$$\begin{aligned} &= AB + A'C + BC = AB + A'C \\ &= AB + A'C + BC(A + A') \\ &= AB + A'C + BCA + BCA' \\ &= ABC(1 + C) + A'C(1 + B) \\ &= AB + A'C \end{aligned}$$

1 ←
Negation Law

A B C

$AB + A'C$

School of Computer Science Engineering and Technology

A	B	C	\bar{A}	\bar{B}	\bar{C}	$AB \cdot A'C$	BC	$AB+A'C$	$AB+A'C+BC$
1	1	1	0	0	0	1	0	1	1
1	1	0	0	0	1	1	0	1	1
1	0	1	0	1	0	0	0	0	0
1	0	0	0	1	1	0	0	0	0
0	1	1	1	0	0	0	1	1	1
0	1	0	1	0	1	0	0	0	0
0	0	1	1	1	0	0	1	1	1
0	0	0	1	1	1	0	0	0	0

Logically Equivalent