**LAB EXPERIMENT 1**

**Aim:** To design the 7 logic gates in Xilinx software using Verilog Programming.

**Theory:** There are 7 basic logic gates in Electronics.

1. AND Gate: The AND gate is an electronic circuit that gives a **high** output (1) only if **all** its inputs are high.  A dot (.) is used to show the AND operation i.e. A.B.  Bear in mind that this dot is sometimes omitted i.e. AB.

Circuit Diagram:

Shape

Description automatically generated

Truth Table:

A picture containing text, clock

Description automatically generated

1. OR Gate: The OR gate is an electronic circuit that gives a high output (1) if **one or more** of its inputs are high.  A plus (+) is used to show the OR operation.

Circuit Diagram:

Shape

Description automatically generated

Truth Table:

A picture containing text, crossword puzzle

Description automatically generated

1. NOT Gate: The NOT gate is an electronic circuit that produces an inverted version of the input at its output.  It is also known as an *inverter*.  If the input variable is A, the inverted output is known as NOT A.  This is also shown as A', or A with a bar over the top, as shown at the outputs.

Circuit Diagram:

A picture containing shape

Description automatically generated

Truth Table:

![Table

Description automatically generated](data:image/gif;base64,R0lGODdhSwBFAPcAAAAAAAAAQAAAgAAA/wAgAAAgQAAggAAg/wBAAABAQABAgABA/wBgAABgQABggABg/wCAAACAQACAgACA/wCgAACgQACggACg/wDAAADAQADAgADA/wD/AAD/QAD/gAD//yAAACAAQCAAgCAA/yAgACAgQCAggCAg/yBAACBAQCBAgCBA/yBgACBgQCBggCBg/yCAACCAQCCAgCCA/yCgACCgQCCggCCg/yDAACDAQCDAgCDA/yD/ACD/QCD/gCD//0AAAEAAQEAAgEAA/0AgAEAgQEAggEAg/0BAAEBAQEBAgEBA/0BgAEBgQEBggEBg/0CAAECAQECAgECA/0CgAECgQECggECg/0DAAEDAQEDAgEDA/0D/AED/QED/gED//2AAAGAAQGAAgGAA/2AgAGAgQGAggGAg/2BAAGBAQGBAgGBA/2BgAGBgQGBggGBg/2CAAGCAQGCAgGCA/2CgAGCgQGCggGCg/2DAAGDAQGDAgGDA/2D/AGD/QGD/gGD//4AAAIAAQIAAgIAA/4AgAIAgQIAggIAg/4BAAIBAQIBAgIBA/4BgAIBgQIBggIBg/4CAAICAQICAgICA/4CgAICgQICggICg/4DAAIDAQIDAgIDA/4D/AID/QID/gID//6AAAKAAQKAAgKAA/6AgAKAgQKAggKAg/6BAAKBAQKBAgKBA/6BgAKBgQKBggKBg/6CAAKCAQKCAgKCA/6CgAKCgQKCggKCg/6DAAKDAQKDAgKDA/6D/AKD/QKD/gKD//8AAAMAAQMAAgMAA/8AgAMAgQMAggMAg/8BAAMBAQMBAgMBA/8BgAMBgQMBggMBg/8CAAMCAQMCAgMCA/8CgAMCgQMCggMCg/8DAAMDAQMDAgMDA/8D/AMD/QMD/gMD///8AAP8AQP8AgP8A//8gAP8gQP8ggP8g//9AAP9AQP9AgP9A//9gAP9gQP9ggP9g//+AAP+AQP+AgP+A//+gAP+gQP+ggP+g///AAP/AQP/AgP/A////AP//QP//gP///yH5BAAAAAAALAAAAABLAEUAAAj/AAEIHEiwoMGDCBMqXKjwn8OHECNKnEixosWLEwVi3Mixo8eHGj+KHCky5D+Bthya1CZJoCRtKg+SnAkSQE2bJ3FqA/CSJQCYNWk6ZDnT5MCUIVum/LdTEkSTRXGONDowZ8ynUq1SVBrSp0CYBB22BLAUI1VbZENCXZsVotKxYmHuxJmULNqyFqnmVJuVLUW+D7XZgqsVsFmpGtEavho0I12caF8aNrhR716xZB1Gxvr3MePJbQ83Vjz061ygjCW+5SvYsNK5lRHLDjz2ZUSoEblCRhly87/aqPOGFloRN3HOx/+mbJrccfOJc3kGf76XofXr2LOHpS5xOHXj3L13/0cYm3t387fFF/+o/i/79ufhP2V/0iP4vPU7yp+/PzVHm/3lp5+A5Q2okoH/gaRfgIghWJmC//Un04MRFhShgwTil+B8BZqVEYUenicaiBxetN9w8t33XIAkqdgci1PBWBJ6WGln4404ykgfjf6hp6N9P2JonovjEXnbjEA6KOF7TG54oJMPsmgkhE+SCGWHIVaZpZAmLknllleOSKKX9jU5JpcmmrklmWgWx2BVba6XJI8ZDhlkmOHdaaWdOfbpZ0J01plnoHqKSWOhXSIZZ5FvqrmnnFomOqCUSi6oKJ7ufSlpmZyeeeGOi5YoaGeXPppppMKVCqaGqLqpappQShYII6ULTtkYj4imSmeuru5K6J/ABhsQADs=)

1. NAND Gate: This is a NOT-AND gate which is equal to an AND gate followed by a NOT gate.  The outputs of all NAND gates are high if **any** of the inputs are low. The symbol is an AND gate with a small circle on the output. The small circle represents inversion.

Circuit Diagram:

Shape

Description automatically generated

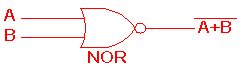
Truth Table:

A picture containing text, crossword puzzle

Description automatically generated

1. NOR Gate: This is a NOT-OR gate which is equal to an OR gate followed by a NOT gate.  The outputs of all NOR gates are low if **any** of the inputs are high. The symbol is an OR gate with a small circle on the output. The small circle represents inversion.

Circuit Diagram:



Truth Table:

A picture containing text, crossword puzzle, clock

Description automatically generated

1. EXOR Gate: The '**Exclusive-OR**' gate is a circuit which will give a high output if **either, but not both**, of its two inputs are high.  An encircled plus sign () is used to show the EOR operation.

Circuit Diagram:

Shape

Description automatically generated

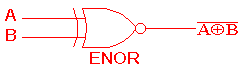
Truth Table:

A picture containing text, crossword puzzle

Description automatically generated

1. EXNOR Gate: The '**Exclusive-NOR'**gate circuit does the opposite to the EOR gate. It will give a low output if **either, but not both**, of its two inputs are high. The symbol is an EXOR gate with a small circle on the output. The small circle represents inversion.

Circuit Diagram:



Truth Table:

A picture containing text, crossword puzzle, clock

Description automatically generated

**Verilog Codes and Screenshots of the program:**

1. **AND Gate:**
   1. **Verilog Code:**

module LAB1\_1\_062(

    input a,

    input b,

    output c

    );

and(c,a,b);

endmodule

* 1. **Screenshots:**

Graphical user interface, text, application

Description automatically generated

A picture containing text, indoor, monitor, screenshot

Description automatically generated

1. **OR Gate:**
2. **Verilog Code:**

module LAB1\_1\_062(

    input a,

    input b,

    output c

    );

or(c,a,b);

endmodule

1. **Screenshot of the Program:**

A picture containing text, indoor, screenshot, electronics

Description automatically generated

Graphical user interface, text, application

Description automatically generated

1. **NOR Gate:**
2. **Verilog Code:**

module LAB1\_1\_062(

    input a,

    input b,

    output c

    );

nor(c,a,b);

endmodule

1. **Screenshot of the Program:**

Graphical user interface, text, application

Description automatically generated

A picture containing text, screenshot, electronics, computer

Description automatically generated

1. **XOR Gate:**
2. **Verilog Code of the Program:**

module LAB1\_1\_062(

    input a,

    input b,

    output c

    );

xor(c,a,b);

endmodule

1. **Screenshots of the Program:**

Graphical user interface, text, application, email

Description automatically generated

A picture containing text, indoor, monitor, screenshot

Description automatically generated

1. **XNOR Gate:**
2. **Verilog Code of the Program:**

module LAB1\_1\_062(

    input a,

    input b,

    output c

    );

xnor(c,a,b);

endmodule

1. **Screenshots of the Program:**

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

1. **NAND Gate:**
2. **Verilog Code of the Program:**

module LAB1\_1\_062(

    input a,

    input b,

    output c

    );

nand(c,a,b);

endmodule

1. **Screenshots of the Program:**

A picture containing text, screenshot, electronics, computer

Description automatically generated

Graphical user interface, text, application

Description automatically generated

1. **NOT Gate:**
2. **Verilog Code of the Program:**

module LAB1\_1\_062(

    input a,

    output c

    );

not(c,b);

endmodule

1. **Screenshots of the Program:**

Graphical user interface

Description automatically generated

Graphical user interface, text, application

Description automatically generated

**Conclusion:** From this experiment we learnt how to use logic gates in Verilog programming. We used the gate level modelling technique to design the logic gates. We also have learnt how to check our output on the ISIM Simulator by applying force clock and force value parameters.