|  |  |  |
| --- | --- | --- |
| G:\nsu-logo.png  **North South University**  Department of Electrical & Computer Engineering    **LAB REPORT**  Course Name:CSE231L  Section: 01  Experiment Number: 2     |  | | --- | | Experiment Name: Universal Gates |     Experiment Date: 7/03/2021  Report Submission Date:14/03/2021  Group Number: | |
| Student Name:Ishrat Jahan | Score |
| Student ID:1921909042 |  |
| Remarks: |

**Universal Gates**

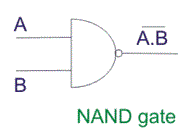
**Objectives:**

* Understanding the concept of Universal Gates (NAND&NOR).
* Implementing the basic logic gates using universal gates in Logisim.
* Learning about Boolean functions using universal gates and converting given circuits to a NAND and NOR gate equivalent circuit.

**Theory:**

**Universal Gate:** A **universal gate** is a logic gate which can implement any Boolean function without the need to use any other type of logic gate. The [NOR gate](https://www.electrical4u.com/nor-gate/) and [NAND gate](https://www.electrical4u.com/nand-gate/) are universal gates. This means that we can create any logical Boolean expression using only NOR gates or only NAND gates.

**NAND Gate as Universal Gate:** The below diagram is of a two-input NAND gate. The first part is an [AND gate](https://www.electrical4u.com/logical-and-gate/) and the second part is a dot after it represents a NOT gate.



During the operation of the NAND gate, the inputs first go through AND gate and after that the output gets reversed and we get the final output.

* A NAND gate is a NOT of an AND gate. Therefore, complementing a AND gate with same inputs creates the NAND equivalent circuit of a NOT gate.
* Complementing the AND gate with two different inputs results in a NAND gate. Thus replacing the NOT part in the gate with its NAND equivalent will result in NAND equivalent AND circuit. This process is reversible. If we complement a NAND gate we will get an AND gate.
* We can use De Morgan’s Law to split a complemented AND gate and draw the OR circuit and replace NOT part in the circuit with its equivalent NAND circuit.
* If we complement the same input of two AND gates and combine them with an OR gate and draw its NAND equivalent circuit we get XOR in NAND gate.
* If we complement the one AND gate and among two and combine them with an OR gate and draw its NAND equivalent circuit we get XNOR in NAND gate.

## NOR Gate as a Universal Gate: A NOR gate is logically an inverted OR gate, which we can find by complementing an OR gate.

|  |
| --- |
| [NOT from NOR.svg](https://en.wikipedia.org/wiki/File:NOT_from_NOR.svg) |
| NOR gate |  |

## As a NOR gate is equivalent to an OR gate leading to NOT gate, joining the inputs makes the output of the "OR" part of the NOR gate the same as the input, eliminating it from consideration and leaving only the NOT part resulting in a NOT gate’s NOR equivalent circuit.

## An OR gate is made by inverting the output of a NOR gate.We already know that a NOT gate is equivalent to a NOR gate with its inputs joined.

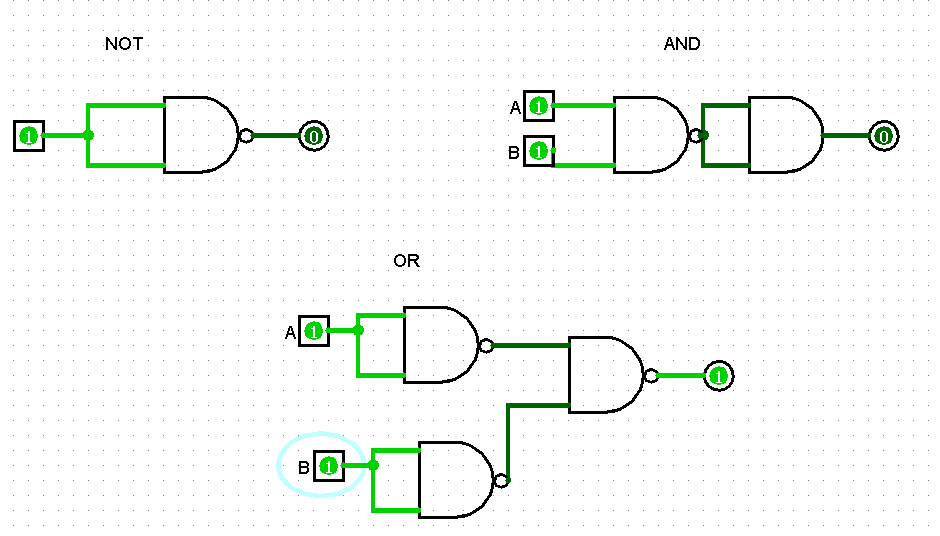
## An AND gate gives a 1 output when both inputs are 1. Therefore, an AND gate is made by inverting the inputs of a NOR gate.

## An XOR gate is made by considering de Morgan’s Law that a NOR gate is an inverted input AND gate. Therefore, complementing one input of an AND gate and the other input of another AND gate and connecting both of the gives a NOR equivalent circuit.

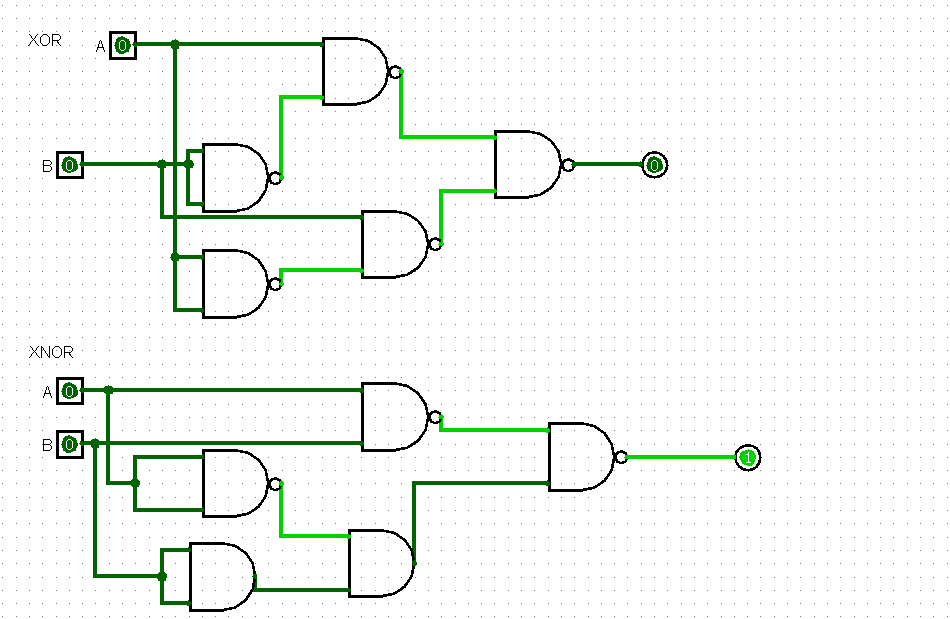
## 

* An XOR gate is made by connecting two AND gates by an OR gate and complementing one of the AND gate. Thus, drawing each of the gates NOR equivalent circuit produces XOR. ( A + B ) ⋅ ( A ¯ + B ¯ ) {\displaystyle (A+B)\cdot ({\overline {A}}+{\overline {B}})}

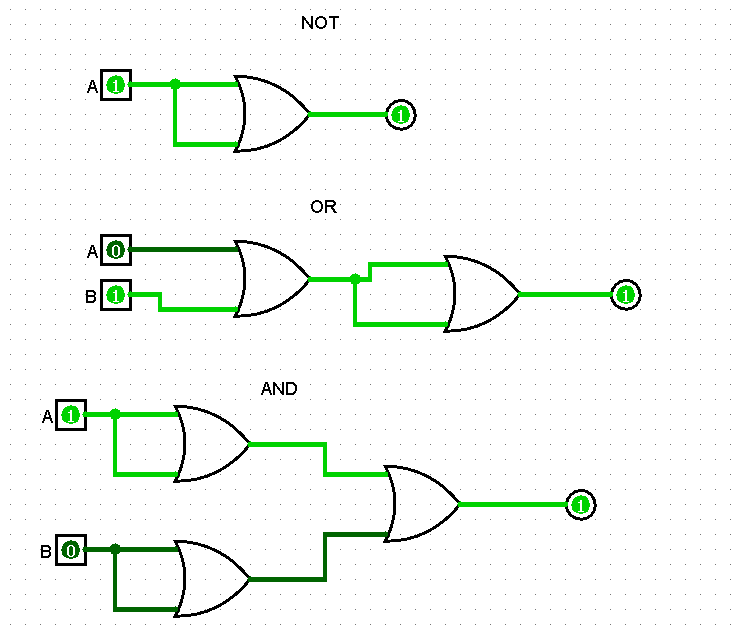
**Circuit Diagrams:**

****

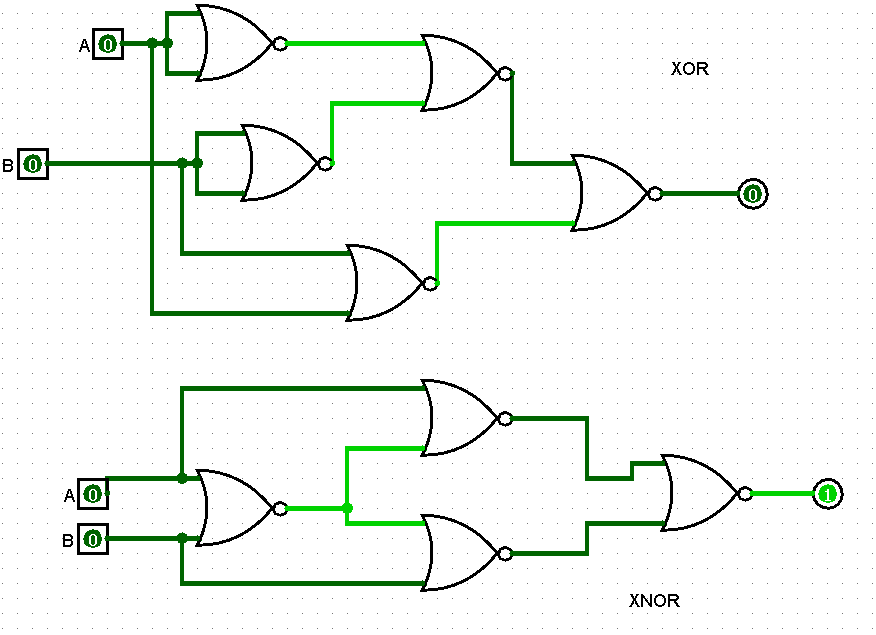
**Figure C1: Implementation of NOT, AND & OR gates using only NAND gates.**



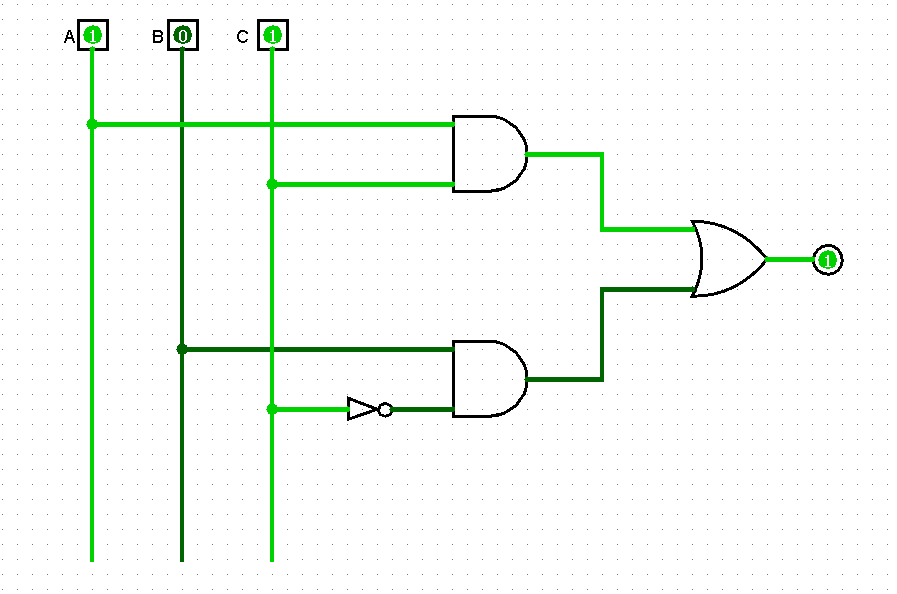
**FigureF1: Implementation of XOR and XNOR using NAND gates**

****

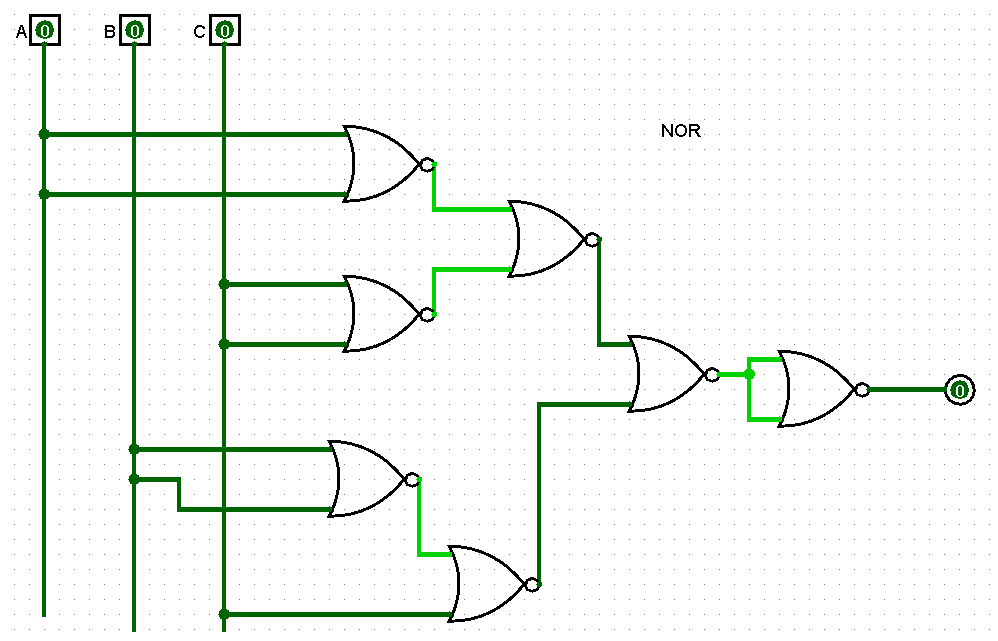
**Figure F2: Implementation of NOT, AND, OR using NOR gates**



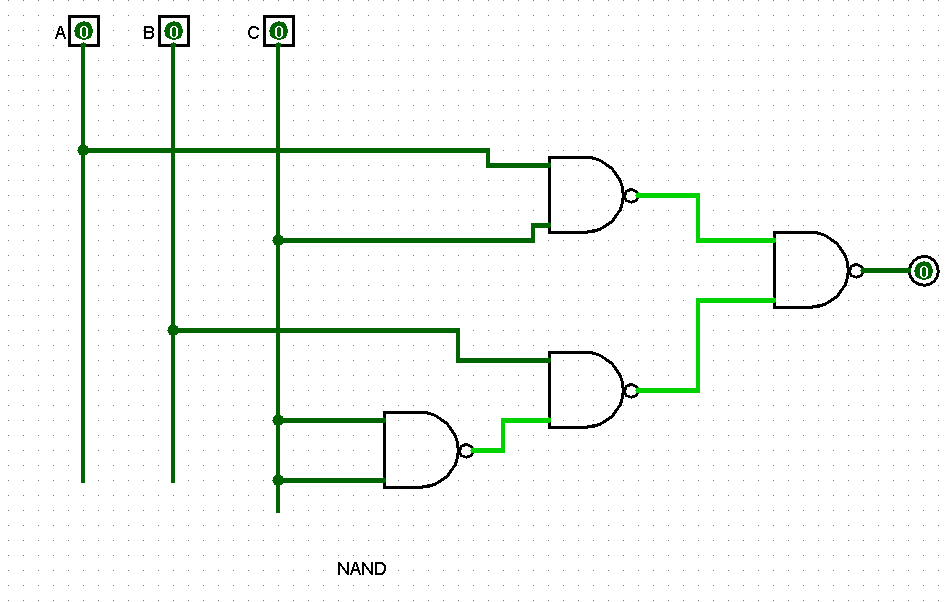
**Figure F2(II):Implementation of XOR and XNOR using NOR gates**



**Figure D2: A combinational circuit**

****

**Universal (NOR) gate implementation of Figure D2**



**Universal (NAND) gate implementation of Figure D2**

**Truth Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **A B C** | **I1=AC** | **I2=BC’** | **F=I1+I2** |
| **000** | **0** | **0** | **0** |
| **001** | **0** | **0** | **0** |
| **010** | **0** | **1** | **1** |
| **011** | **0** | **0** | **0** |
| **100** | **0** | **0** | **0** |
| **101** | **1** | **0** | **1** |
| **110** | **0** | **1** | **1** |
| **111** | **1** | **1** | **1** |

**Truth table of combinational circuit in Figure D2**

**Discussion:** The concept of the lecture was based on Universal Gates. At

first we learned which gates are called Universal Gates and the most

common Universal Gates NAND and NOR and their respective circuit

diagrams. Then we proceeded to learn how to draw all other basic gates OR,

NOT and AND using both NAND and NOR.We used Logisim to verify all

the basic gates by varying their input between 1 and 0 and determining the

output it gives. We carried and converted XNOR and XOR gates to their

respective NAND and NOR gate equivalent. For conversion to a NAND gate

we used the following formulas for the following gates:-

**NAND as NOT**: A NAND gate is a NOT of an AND gate. Therefore, (A.A)’=A’; Inputs are same.

**NAND as AND**: We get NAND gate by complementing the AND gate.

(A.B)-> (A.B)’; two different inputs. We can then draw the Y=(A.B)’ circuit and replace the NOT part with its equivalent NAND gate.

This process is reversible. If we complement a two different input NAND gate we will get an AND gate.

**XOR as NAND:** A.B’+A.B’

**XNOR as NAND:** A.B+A’.B’

**NAND as OR:** If we have two different input A and B NAND gate we can use De Morgan’s Law to split it and draw a OR circuit. Then we can replace NOT part in the circuit with its equivalent NAND circuit.

For conversion to a NOR gate we used the following formulas for the

following gates:-

**NOR as NOT:** We use the formula (A+A)’=A; with same inputs to create

the NOT gate.

**NOR as AND:** We use the formula (A’+B’)’= ((A)’) =AB;

**NOR as OR:** We use the formula (A+B) = A’.B’=A’+B’ = (AB)’

**XOR as NOR:** (A+B’).(A’.B)

**XNOR:** (A+B).(A’+B’)

Lastly, we converted a given combinational circuit into its NAND and NOR

equivalent circuit and validated the truth table of the universal gate circuit.