**Student Name: Federico Watkins**

**CSIS 1800: Introduction to Computer Science and Information Systems**

**Chapter number: 4 Gates and Circuits**

**Assignment number: 4**

1. Give the three representations of a NOT gate and say in words what NOT means.

Boolean Expression:

X = 

Logical Diagram Symbol:



Truth Table:

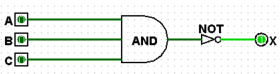
|  |  |
| --- | --- |
| A | B |
| 0 | 1 |
| 1 | 0 |

What it means:

NOT gate is an inverter gate because it inverts the input value. If the input value for a NOT gate is 0, the output value is 1, and if the input value is 1, the output is 0.

(Dale and Lewis, Computer Science Illuminated, p. 90-92)

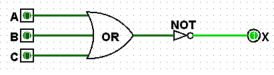
2. Draw and label the symbol for a three input NAND gate, then show its behavior with a truth table.



|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C | X |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

(Dale and Lewis, Computer Science Illuminated, p. 94-95)

3. Draw and label the symbol for a three-input NOR gate, then show its behavior with a truth table



|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C | X |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

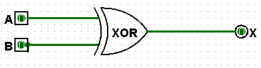
(Dale and Lewis, Computer Science Illuminated, p. 94-95)

4. Give the three representations of an XOR gate and say in words what XOR means.

Boolean Expression:

A ^ B

Logical Diagram Symbol:



Truth Table:

|  |  |  |
| --- | --- | --- |
| A | B | X |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

What it means:

The exclusive OR (XOR) produces 0 if its two inputs are the same, and a 1 otherwise. So when both input signals are 1 the XOR produces a 0, unlike the OR which produces a 1. That is why the OR is sometimes called the inclusive OR.

(Dale and Lewis, Computer Science Illuminated, p. 93-94)

5. Differentiate between a half adder and a full adder using truth tables and explain what each means.

Half Adder:

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | Sum | Carry |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

A half adder is a circuit that computes the sum of two bits and produces the correct carry bit. It is important to note that the half adder is fine for adding two single digits, but it cannot be used as is to computes the sum of two binary values with multiple digits each. That’s because the half adder does not take into account a possible carry-in value into the calculation.

Full Adder:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | Carry-in | Sum | Carry-out |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

A full adder is essentially two half adders combined to handle the situation in which two binary values with multiple digits each are added. The input to the sum is the carry-in and the sum from adding the two input values. That is, the sum from the half adder is added to the carry-in. Both additions have a carry out.

(Dale and Lewis, Computer Science Illuminated, p. 102-104)

6. Provide mathematical expressions for six properties of Boolean algebra and explain what each means.

(Dale and Lewis, Computer Science Illuminated, p. 95-102)

***Commutative property:***

* The commutative property is the property which produces the same results when adding or multiplying the two variables and its reverse order. It is represented in the binary operations “AND” and “OR” gate.

AND operation:

A∙B = B∙A 🡪 A is 1 and B is 0, the commutative property for AND operation 🡪 1∙0 = 0∙1 🡪 0 = 0 🡪 both produce the same result.

OR operation:

A+B = B+A 🡪 A is 1 and B is 0, the commutative property for OR operation 🡪 1+0 =0+1 🡪 1 = 1 🡪 both produce the same result.

***Associative property:***

* The associative property is the property which produces the same results when the group of variables is added or multiplied together within the parentheses and its reverse order. It is represented in the binary operations “AND” and “OR” gate.

AND operation:

 (A∙B)∙C = A∙(B∙C) 🡪 For example: 🡪 A is 1, B is 0, and C is 1, the associative property for AND operation 🡪 (1∙0)∙1 = 1∙(0∙1) 🡪 0∙1 = 1∙0 🡪 0 = 0 🡪 both produce the same result.

 OR operation:

 (A+B)+C = A+(B+C) 🡪 For example: 🡪 A is 1, B is 0 and C is 1, associative property for OR operation (1+0)+1 = 1+(0+1) 🡪 1+1 = 1+1 🡪 1 = 1 🡪 both produce the same result.

***Distributive property:***

* The distributive property is the property when the variable multiplied by a group of variable added together produces the result which is same as that of the variable multiplied separately and then added together. The distributive property is represented in the binary operations “AND” and “OR” gate.

AND operation:

 A∙(B+C) = (A∙B)+(A∙C) 🡪 A is 1, B is 0, and C is 1, distributive property for AND operation 🡪 1∙(0+1) = (1∙0)+(1∙1) 🡪 1∙1 = 0+1 🡪 1 = 1 🡪 both produce the same result.

OR operation:

 A+(B∙C) = (A+B) ∙(A+C) 🡪 A is 1, B is 0 and C is 1, distributive property for OR operation: 🡪 1+(0∙1) = (1+0) ∙(1+1) 🡪 1+0 = 1∙1 🡪 1 = 1 🡪 both produce the same result.

***Identity property:***

* The identity property is the property which produces the same results when sum of 0 and one variable produces the variable itself or product of 1 with one variable produces the variable itself. It is represented in the binary operations such as “AND” and “OR” gate.

AND operation**:**

A∙1 = A 🡪 A is 1, identity property for AND operation: 🡪 1∙1 = 1 🡪 1 = 1

**OR operation:**

A+0 = A 🡪 A is 1, identity property for OR operation: 🡪 1+0 = 1 🡪 1 = 1

***Complement property*:**

* The product of variable with its complement produces 0 and the Sum of variable with its complement produces the 1.

AND operation**:**

A∙ Ᾱ = 0 🡪 A is 1, complement property for AND operation: 🡪 1 = 0 🡪 1∙0 = 0 🡪 0 = 0

OR operation**:**

A+Ᾱ = 1 🡪 A is 1, complement property for OR operation: 🡪 1+ = 1 🡪 1+0 = 1 🡪 1 = 1

***DeMorgan’s law property*:**

* The DeMorgan’s law states that the complement of results produced in AND gate is equivalent to the complement of the individual inputs and then passed into an OR gate. The DeMorgan’s law also states that the complement of result produced in OR gate is equivalent to the complement of the individual inputs and then passed into an AND gate.

AND operation**:**

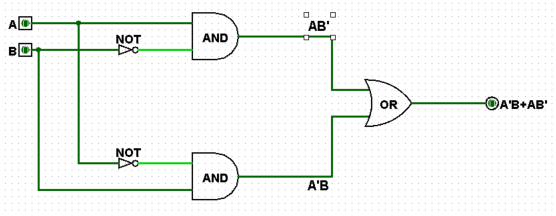
(Ᾱ) = Ᾱ + 🡪 A is 1 and B is 0, DeMorgan’s law property for AND operation: 🡪 () = + 🡪 = 0+1 🡪 1 = 1

OR operation**:**

(Ᾱ+) = Ᾱ 🡪 A is 1 and B is 0, DeMorgan’s law property for OR operation: 🡪 🡪 🡪 0 = 0

7. Draw a circuit diagram corresponding to the following Boolean expression:

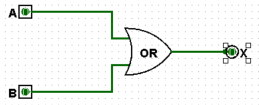
A’B + AB’. What gate is it?



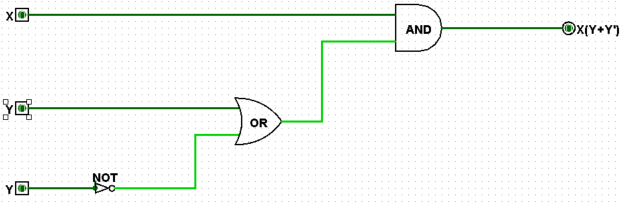
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | A’B | AB’ | A’B+AB’ |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 |

(Dale and Lewis, Computer Science Illuminated, p. 90-106)

8. Minimize the following Boolean expression first and then draw a circuit diagram:  
 [ ( A+B )’ ( A’+B’) ]’ 🡪 Applying properties of Boolean algebra discussed in question 6 this expression can be minimized to: A + B

  
(Dale and Lewis, Computer Science Illuminated, p. 90-106)

9. Minimize the following Boolean expression first and then draw a circuit diagram:

[ X’Y + XY’ ]’ 🡪 Applying properties of Boolean algebra discussed in question 6 this expression can be minimized to: X(Y+Y’) 🡪   
(Dale and Lewis, Computer Science Illuminated, p. 90-106)  
  
10.Design a table showing XOR calculations for 3 input variables (What parity is it: odd or even?)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A | B | C | XOR | Odd | Even |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 |

The table shows that a three input XOR gate and the Odd Parity gate behaves identically so it is safe to say that the XOR gate has Odd parity.

(http://www.cburch.com/logisim/docs/2.1.0/libs/gates/xor.html)