THE UNIVERSITY OF ZAMBIA

DEPARTMENT OF MECHANICAL ENGINEERING

NAME : TWAAMBO CHOOPA

COMPUTER NUMBER : 2018207369

SCHOOL : MINES

COURSE : MEC 3102

LAB NUMBER : 02 LOGIC UNITS

LECTURER : MR. BOYD MUNKOMBWE

AIM

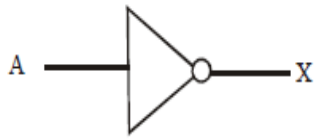
Getting a full understanding of the logic operations of the AND, OR, NOT, NOR and NAND gates and familiarizing with creating, use and interpretation of truth tables.

INTRODUCTION

A logic gate is an idealized model of computation or physical electronic device implementing a Boolean function, a logical operation performed on one or more binary inputs that produces a single binary output. Its output would be HIGH (1) or Low (0) depending on the digital level (s) at the input terminal (s).
Digital or logic circuits operate in the binary mode where each input and output voltage is either a 0 or 1. The 0 and 1 designations represent predefined voltage ranges. The characteristic of logic circuits allows us to use Boolean algebra as a tool for the analysis and design digital circuits. Boolean algebra is a relatively simple mathematical tool that allows us to describe the relationship between logic circuit output (s) and its inputs as an algebraic equation. This algebraic equation is known as Boolean expression or simply Boolean equation.

There three fundamental gates that are used in logic circuit namely the AND, NOT and the OR gate.

The INVERTER is also known as a NOT gate. It is a logic gate that performs inversion (also called complementation) operation. It only allows one input. It changes one logic level to the opposite logic level, i.e., from HIGH to LOW level or from LOW to HIGH level. In terms of bits, it changes a 1 to 0 and a 0 to 1.



(*a*) logic symbol

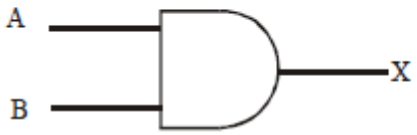
Input *A*	*Output* *X*
0	1
1	0

(*b*) truth table

Fig. 3-1. *An INVERTER.*

The AND gate is one of the basic gates from which all logic functions are constructed. It is composed of 2 or more inputs and a single output. Fig 3-9(a) shows a logic symbol for a 2-input AND gate. Note that inputs are shown on the left (labelled as A and B) and the output is on the

right (labelled as X).



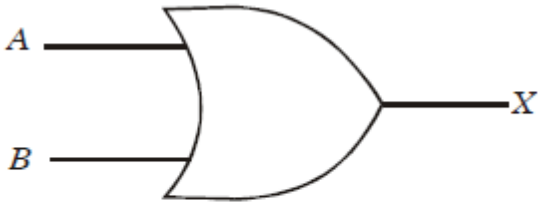
(a) logic symbol

Inputs		Output
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

(b) truth table

Fig. 3-9. *A 2-input AND gate.*

Like AND gate, the OR gate, is another basic logic gate from which all logic functions are constructed. It is composed of two or more inputs and a single output. Fig-3-21(a) shows a logic symbol for a 2-input OR gate. Notice that the inputs are shown on the left labelled as A and B and the output is on the right (labelled as X).



(a) Logic symbol

Inputs		Output
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

(b) Truth table

Fig. 3-21. *A 2-input OR gate*

Other gates are as a result of a combination of these three fundamental gates. For example, a NOR gates are a combination of a NOT and an OR. A NAND gates is a combination of an AND followed by a NOT GATE.

APPARATUS

Indikit, wires.

PROCEDURE

As indicated in the laboratory manual.

DATA COLLECTION

Practical 2.1

Truth Table

A	$\bar{A}$
1	0
0	1

Practical 2.2

Truth Table

A	B	Z
0	0	1
0	1	1
1	0	1
1	1	0

Practical 2.3

Truth Table

A	B	Z
0	0	0
0	1	0
1	1	1
1	0	0

Practical 2.4

Truth Table

A	B	Z
0	0	0
0	1	1
1	0	1
1	1	1

Practical 2.5

Truth table

A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0

Practical 3.1

Truth Table

A	B	C	Z
0	0	0	0
0	0	1	0
0	1	1	0
1	1	1	1
0	1	1	1
0	0	1	1
1	0	1	0
0	1	0	0

Practical 3.2

Truth Table

A	B	C	Z
0	0	0	0
0	0	1	0
0	1	1	0
1	1	1	1
0	1	0	0
1	0	1	1
1	1	0	1
1	0	0	0

Practical 3.3

Truth Table

A	B	C	Z
0	0	0	0
0	0	1	0
0	1	1	1
1	1	1	1
0	1	0	1
1	0	1	0
1	1	0	0
1	0	0	0

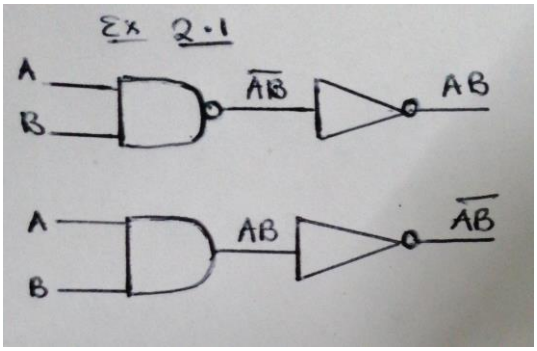
Practical 3.4

Truth Table

A	B	C	Z
0	0	0	0
0	0	1	0
0	1	1	0
1	1	1	1
0	1	0	0
1	0	1	0
1	1	0	1
1	0	0	1

Exercise 2.1

(a)



Truth table for AND

A	B	Z
0	0	0
0	1	0
1	0	0
1	1	1

(b)

The Truth table note the revaluations with correspondence to the executions from the operation above in the practical

A	B	Z
0	0	1
0	1	1
1	0	1
1	1	0

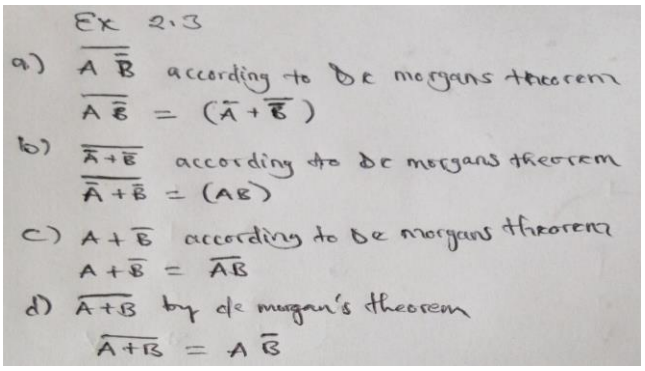
Q2.2 How does the OR operation differ from ordinary addition?

OR gates does not perform addition or any other mathematical function but rather makes logical decision on true or false on two [more] inputs both input are false then the output will be false "0" that is the only premise for an OR gate.

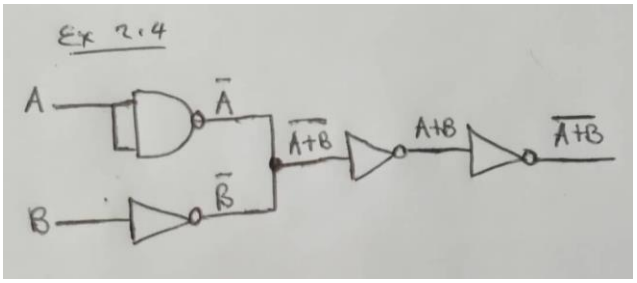
Exercise 2.2

A	**B**	$\overline{\mathbf{A}}$	**B**	$\overline{\mathbf{A}}.\overline{\mathbf{B}}$	$\overline{\overline{\mathbf{A}}.\overline{\mathbf{B}}}$	**A+B**
0	0	1	0	1	0	0
0	1	1	1	0	1	1
1	1	0	1	0	1	1
1	0	0	0	0	1	1

Exercise 2.3



Exercise 2.4



Exercise 2.5

The exclusive probable four identities for the OR gate are as highlighted below;

- 0 + 0 = 0
- 1 + 0 = 1
- 0 + 1 = 1
- 1 + 1 = 1

Exercise 3.1

Basing on truth table method will prove the groups of theorems

(a) 1 + A = 1
1-1=A=1-1
A=0

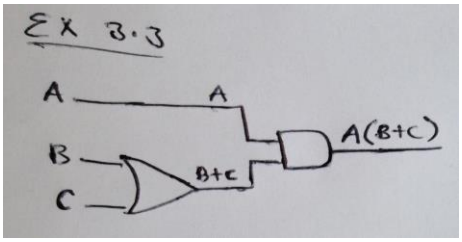
(b) A+A=A
A (1+1) = A
As per logical gate principle perspective, when 1 is ANDED WITH 1, 1+1=1 hence,
A (1) = A, A=A Hence proven.
Therefore, A.A=A is also logically proven since initially A WAS found to be 1,when 1 is ANDED with 1, 1.1 results to 1. Such that A ANDED A gives A.

(C) A+A'=? A.A'=0 Going across this will divide A' into 0 to make a the subject
A=0/A=0 Hence A=0. To replace A & A' we get 0+0=0 also 0.0=0 hence proven.

Exercise 3.2

From the two figures, it is observed that when the output of one inverter(inverter A) is passed through a second inverter(inverter B) and taken as an input, the output will be similar with the original input for inverter A

Exercise 3.3



Exercise 3.4 – Distributive Theorem

Consider the expression:

Z = A + BC

Which can be expressed as (A + B)(A + C). Therefore:

(A + B)(A + C) = A + BC

The logic circuits for both sides are as shown below:

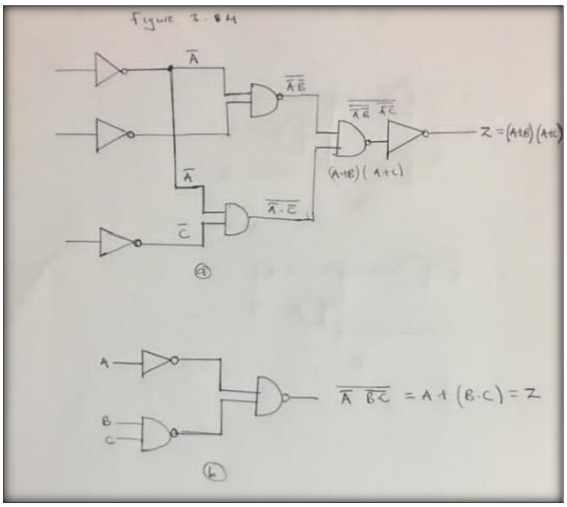
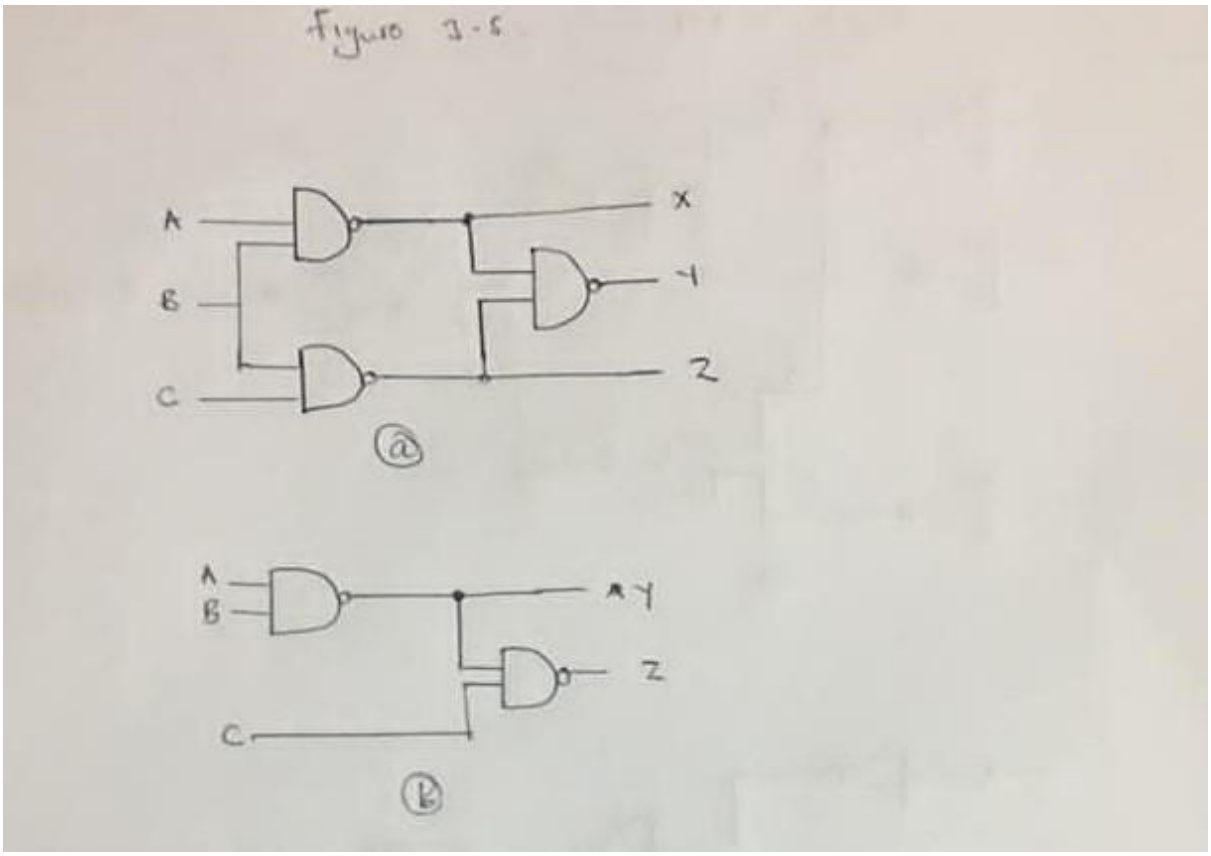


Figure 1 - Logic Circuit for Z = (A+B)(A+C

Exercise 3.5 – Writing expressions from logic circuit



Expressions for X, Y, and Z were written from Figure 17 (a) and (b) below.

(a) From Figure 17 (a)

$\boldsymbol{X = \overline{A.B}}$ $\boldsymbol{Y = \overline{B.C}}$ $\boldsymbol{Z = A.B + B.C}$

(b) From Figure 17 (B)

$\boldsymbol{Y = \overline{A.B}}$ $\boldsymbol{Z = A.B + \overline{C}}$

Exercise 3.6 – Writing sum of products expressions for complements

(a) $Y = (A + B)(C + D)(B + C)$ Then the complement is

$\bar{Y} = \overline{(A + B)(C + D)(B + C)}$
$\boldsymbol{\bar{Y} = \bar{A}.\bar{B} + \bar{C}.\bar{D} + \bar{B}.\bar{C}}$

(b) $Z = \overline{(A + B)C} + \overline{(\bar{A} + \bar{B})\bar{C}}$

$Z = \overline{(AC + BC)} + \overline{(\bar{A}\bar{C} + \bar{B}\bar{C})}$ By Distributive Laws
$\bar{Z} = \overline{\overline{(AC + BC)} + \overline{(\bar{A}\bar{C} + \bar{B}\bar{C})}}$ Complementing Z
$= \overline{\overline{(AC + BC)}}.\overline{\overline{(\bar{A}\bar{C} + \bar{B}\bar{C})}}$ By De Morgan's theorem
$= (AC + BC).(\bar{A}\bar{C} + \bar{B}\bar{C})$
$= AC(\bar{A}\bar{C} + \bar{B}\bar{C}) + BC(\bar{A}\bar{C} + \bar{B}\bar{C})$ By Distributive Laws

Exercise 3.7

(a) $A + AB$

$= A.(1 + B)$ But 1 + B = 1
$\boldsymbol{Therefore\ A + AB = A}$

(b) $(A + B + C)(A + \bar{B})$

$= A(A + B + C) + \bar{B}(A + B + C)$ By Distributive laws
$= AA + AB + AC + A\bar{B} + B\bar{B} + \bar{B}C$ But $AA = A$ and $B\bar{B} = 0$
$= A + AB + AC + A\bar{B} + \bar{B}C$
$= A(1 + B + \bar{B} + C) + \bar{B}C$ But 1 + X = 1
$A.1 + \bar{B}C = A + \bar{B}C$
$\boldsymbol{Therefore\ (A + B + C)(A + \bar{B}) = A + \bar{B}C}$
(c) $(A + BC)(B + AC) + \bar{A}B\bar{C}$
(d) $= A(B + AC) + BC(B + AC) + \bar{A}B\bar{C}$ By Distributive laws
(e) $= (AB + AAC) + (BBC + ABCC) + \bar{A}B\bar{C}$
(f) $= AB + AC + BC + ABC + \bar{A}B\bar{C}$
$\boldsymbol{(g)\ Therefore, (A + BC)(B + AC) + \bar{A}B\bar{C} = AB + AC + BC + ABC + \bar{A}B\bar{C}}$

DISCUSSION

The modules given had NAND and NOT gate which were combined to come up with other gates. For example, the OR operation in practical 2.4 was achieved using two NOT gates and one NAND gate as opposed to use the actual OR gate, the modules were used. Thus, in the absence of an AND, OR gate, a NAND gate can be very handy in achieving the same operations. However, this means that we would use more modules. This in circuit design is costly. Simplifying Boolean expressions of a given logic circuit can also help simplify the logic circuit to be constructed thereby reducing the number of modules or gates used. The experiment was also used to verify some of the rules of Boolean Algebra. The NAND gate is almost universally used in integrated circuit logic although all types of operations can be obtained, usually at a greater cost. The design tools are mostly better adapted to the use of the three fundamental operations of AND, OR, NOT was connecting a NOT at the output of AND gives a NAND and adding a NOT to output of OR gives a NOR. The sum of products for Boolean functions is the most useful for NAND logic but if the number of variables is large or if the number of product terms is large, either could exceed the number of inputs available on standard integrated circuit module. Up to eight inputs are available on some types and in either input expanders can be connected to increase the number of available inputs.

CONCLUSION

The experiments the assignments were successful as the desired objectives were met. The experiments enhanced my understanding of basic logic operations, combinational logic equivalence, non-equivalence and other circuits.

REFERENCES

1. EEE 3131 – **INTIKIT I.C. Logic Tutor Student Assignments** Lab Manual.
2. Hatilima J, **EEE 3131 – Digital Electronics Notes**, (2015), the University of Zambia, Lusaka.