
# THE UNIVERSITY OF ZAMBIA

## SCHOOL OF ENGINEERING

## DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

## LAB 1 REPORT

NAME: NTALASHA MWANSA

COMPUTER NUMBER: 2019109646

COURSE: EEE 3131

LECTURER: MR GEORGE ZIBA

DUE DATE: 30TH APRIL, 2021

**1.0 TITLE**

BASIC LOGICAL OPERATTIONS

**2.0 OBJECTIVES**

- To become familiar with the logical operations of AND, OR, NOT, NOR and NAND and the use of truth tables.
- To study the algebraic representation of general logic operations AND, OR, NOT and their realisation with NAND modules.

**3.0 THEORY**

There are various ways of connecting logic gates in order to get an intended output. Several different types of Integrated Circuit Logics are available, OR, AND, NAND, NOT, NOR, but to mention a few. But it is worth noting that the two most available are NOT and NAND integrated logic circuits.

Integrated Circuit Logic NAND is the easiest and cheapest operation to achieve and is therefore more widely used in practice as a logic element than any other.

**4.0 EQUIPMENT REQUIRED**

- Multisim software
- Hp ENVY laptop

**5.0 PROCEDURE**

### Practical 2.1: The NOT gate

Set up the circuit as instructed in the lab manual

### Practical 2.2: The NAND gate

Set up the circuit as instructed in the lab manual

**Practical 2.3**: **The AND gate**

Set up the circuit as instructed in the lab manual

### Practical 2.4: The OR gate

Set up the circuit as instructed in the lab manual

**Practical 2.5: The NOR gate**

Set up the circuit as instructed in the lab manual

**Practical 3.1: BOOLEAN ALGEBRA DISTRIBUTIVE PROPERTY**

Connect up the circuit on multism software and check out the truth table for all combinations of A, B, C.

$$AB + AC = A(B+C)$$

**Practical 3.2: COMBINATIONAL LOGIC**

Set up the result on multism software, using the same input switches for A, B, C as you are already using but a different lamp to indicate the output.

**Practical 3.3: COMBINATIONAL LOGIC**

Set up your circuits for both sides as before on multism software and check that the outputs are equal for all settings of A,B,C. The left hand expression needs no more than one 7400 and one 7404.

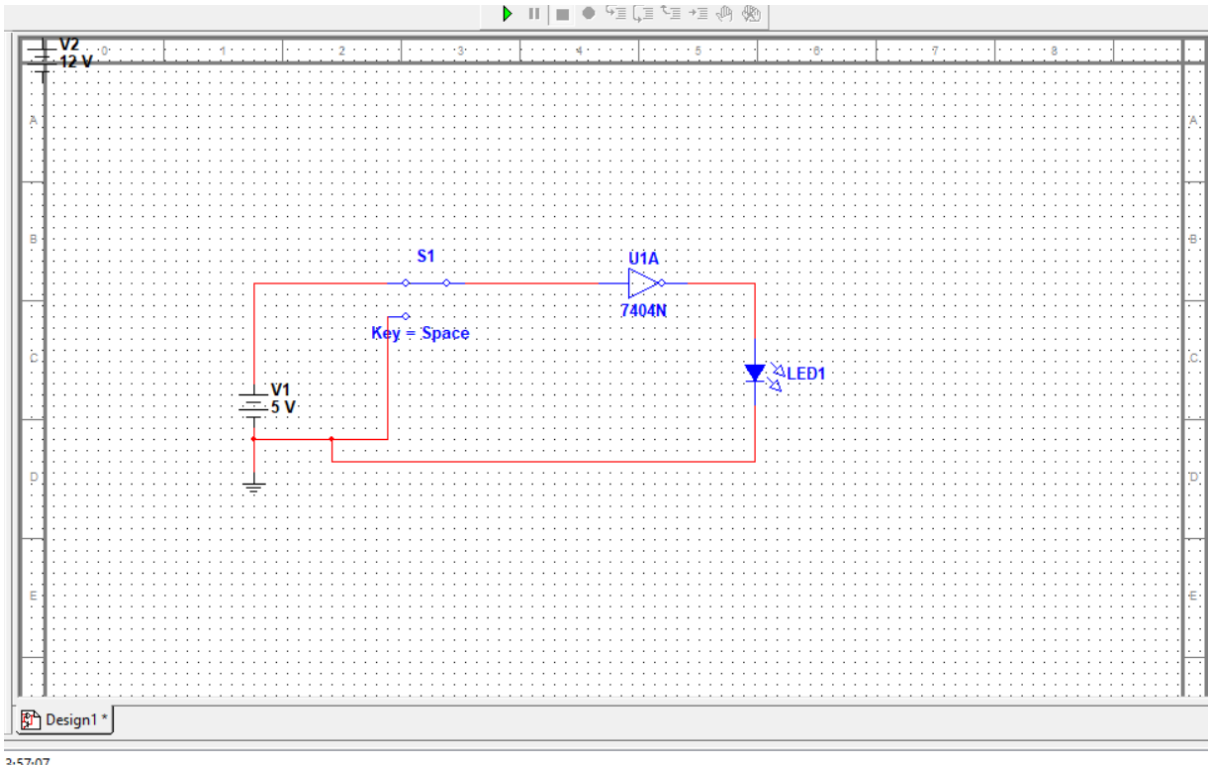
$$(A + B)\,(A + C) = A + BC$$

**Practical 3.4: COMBINATIONAL LOGIC**

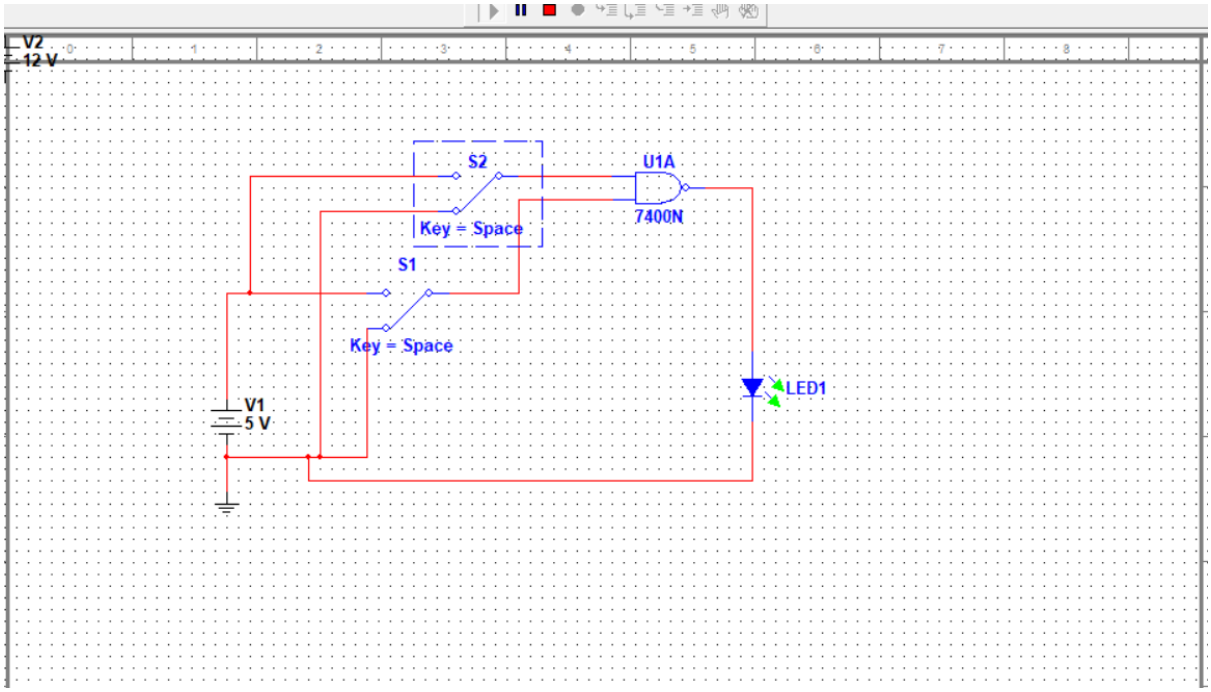
Set up logic circuits on multism software and simulate for (A + B) C and for A B + C and display their outputs on adjacent lamps.

**6.0 RESULTS**

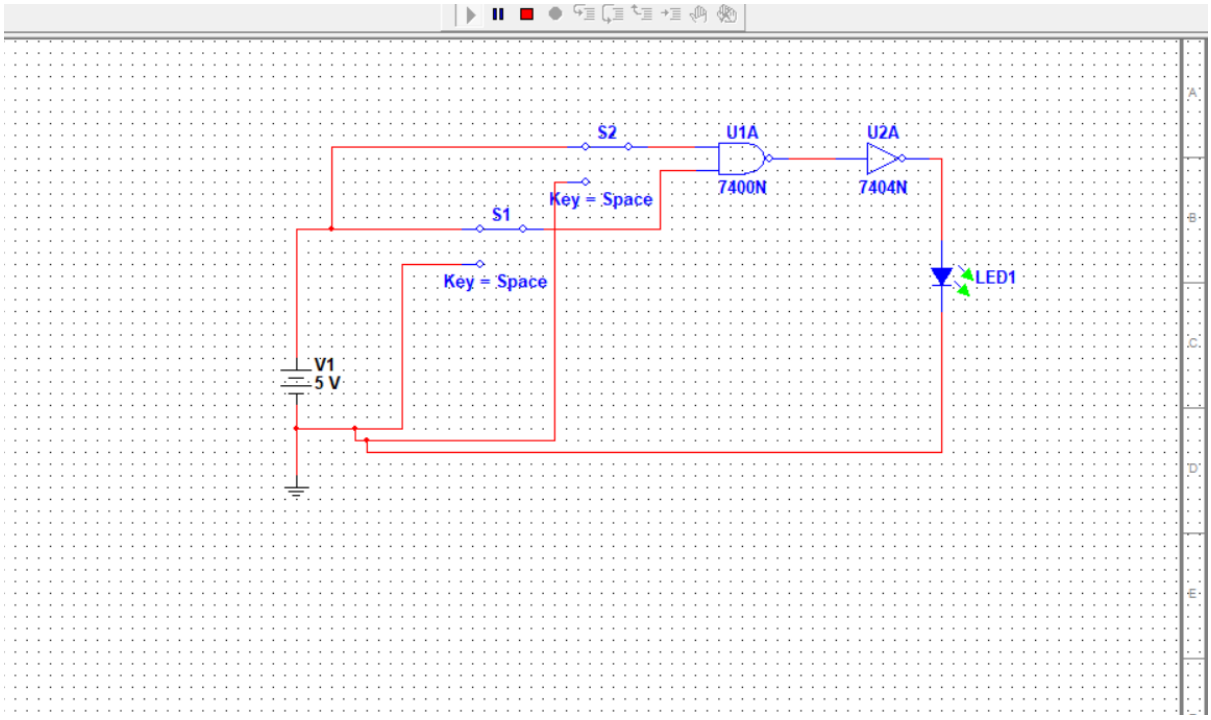
Practical 2.1



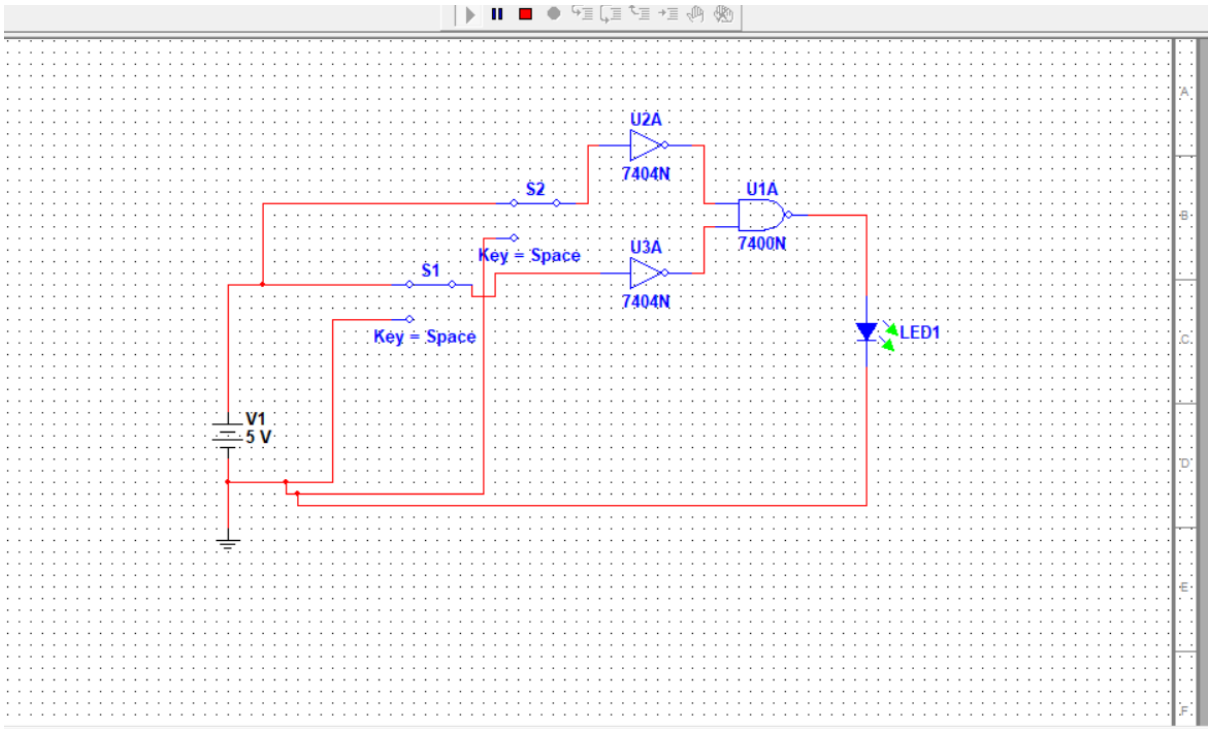

Practical 2.2



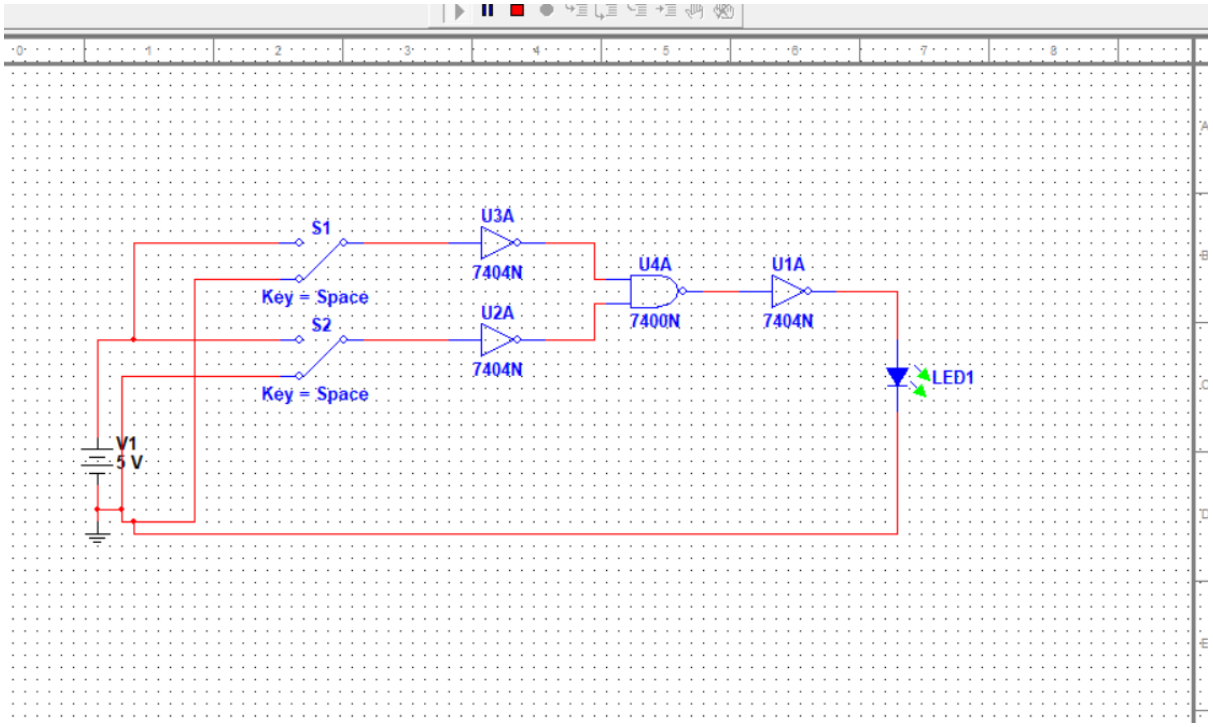
Practical 2.3



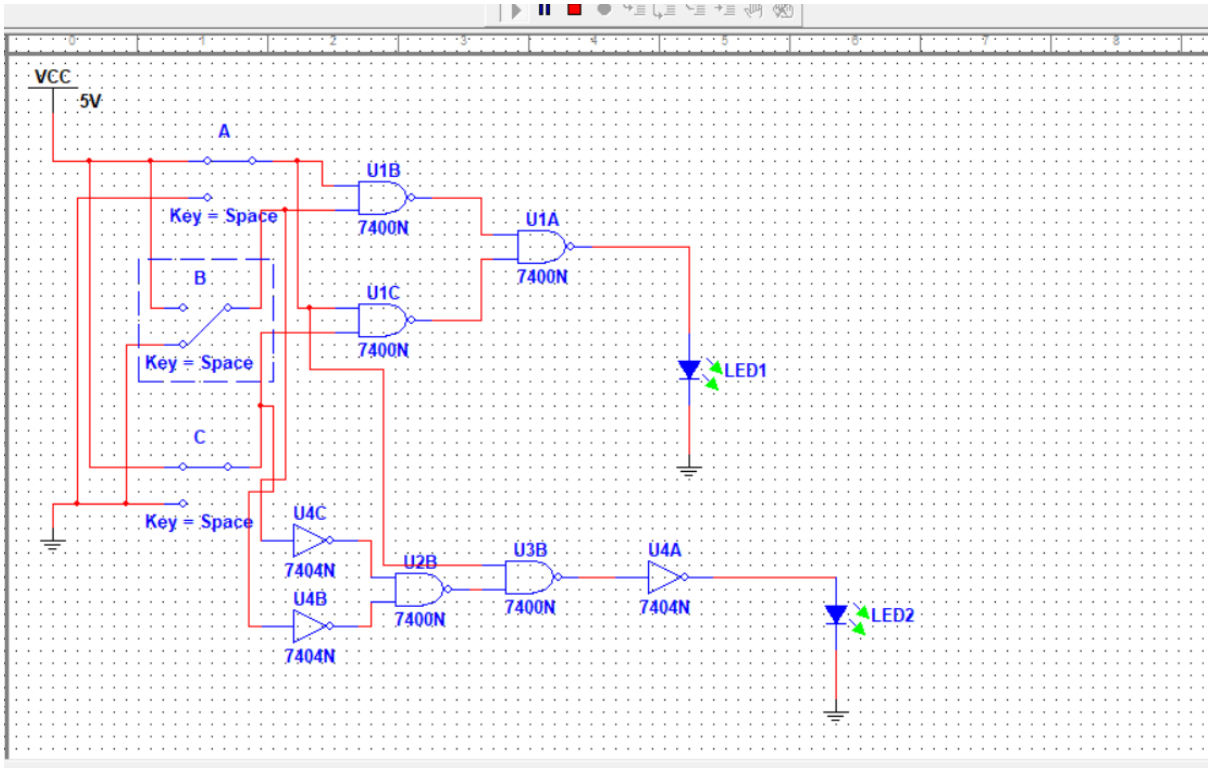

Practical 2.4



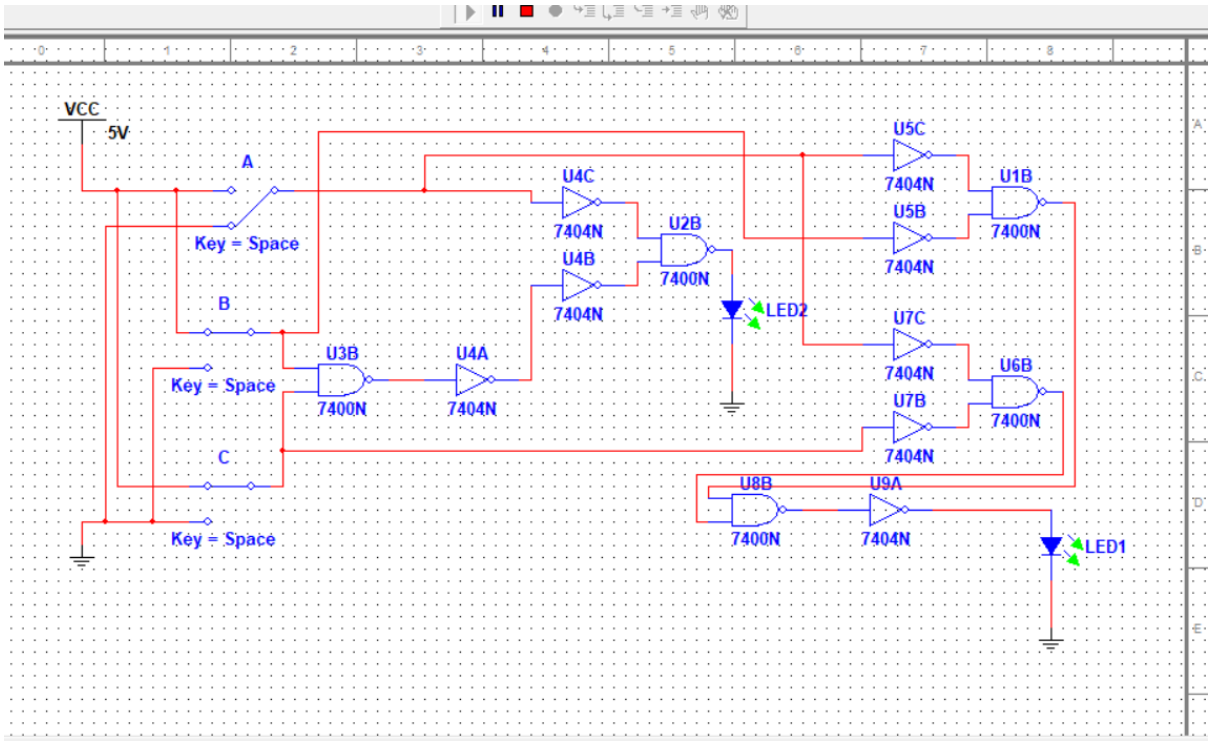
Practical 2.5



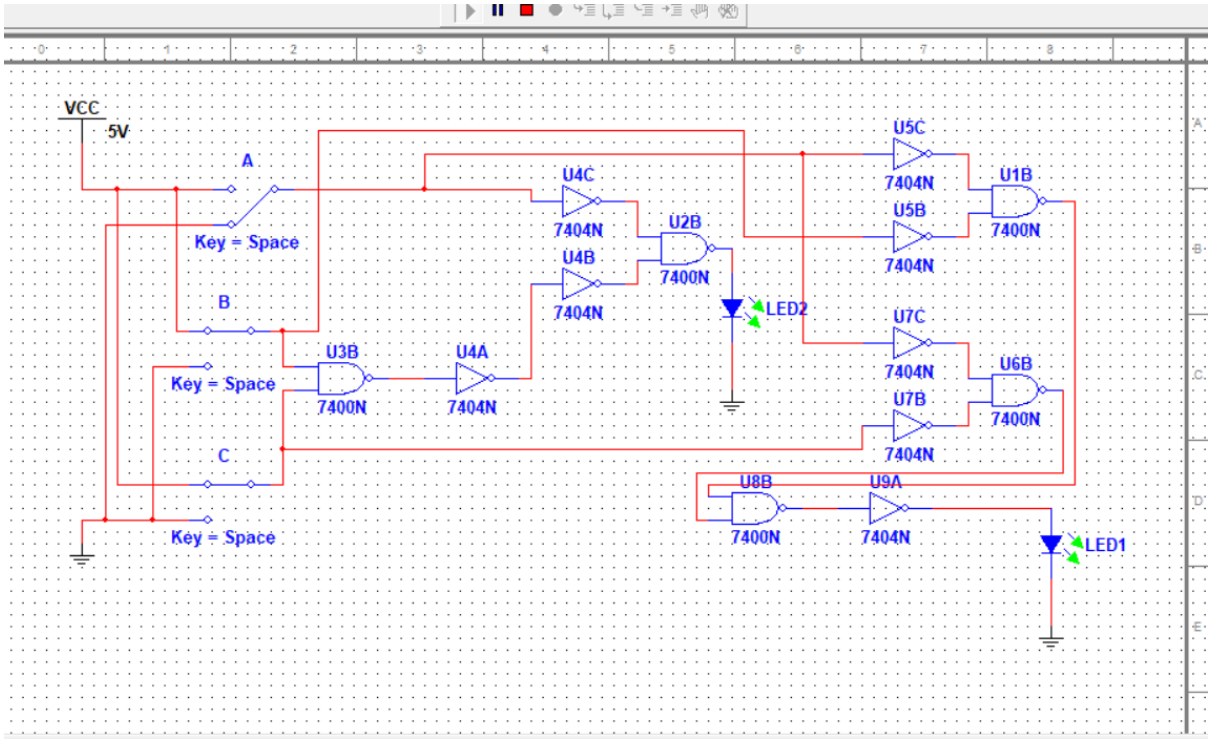
Practical 3.1



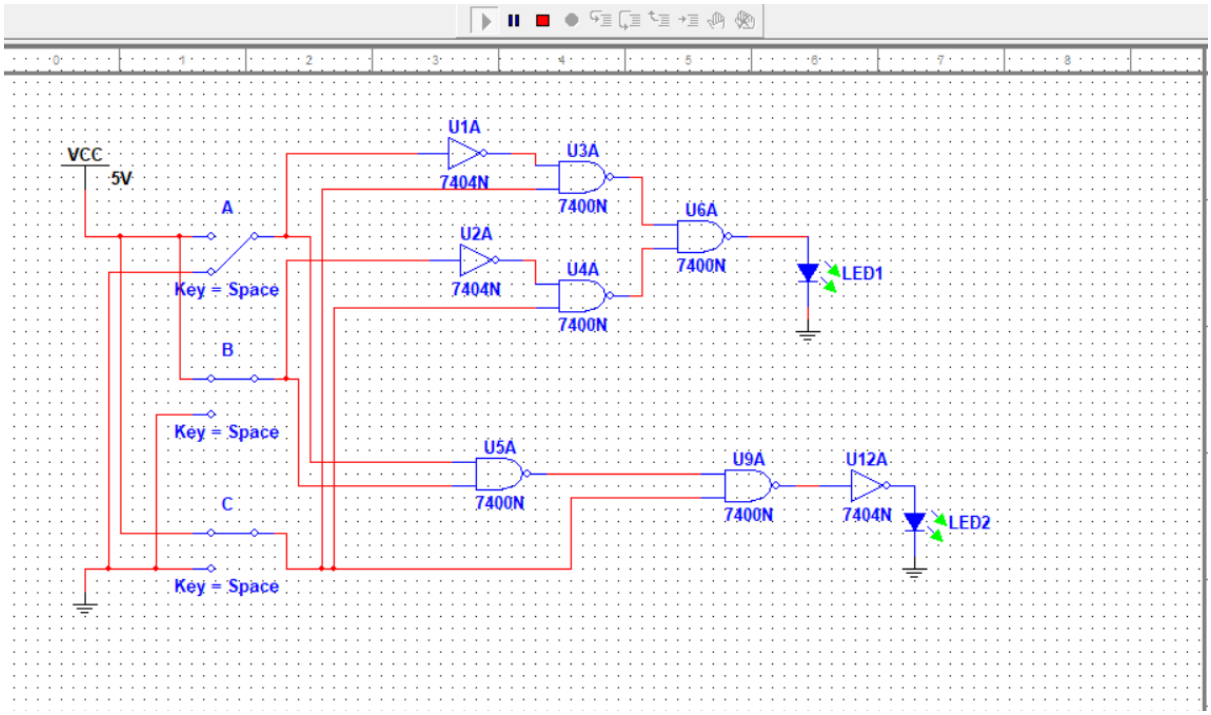

Practical 3.2



Practical 3.3




Practical 3.4



**7.0 DISCUSSION**

Using the 7400N and 7404N TTL gates, different logic operations were conducted using the multism simulation software. To achieve these operations, Single Pole Double Throw Switches (SPDT) were used to switch between 0 and 1 inputs.

The outputs of the circuits were indicated by an LED, which lit up when the output was '1' or in other terms 5V (HIGH) and was off when the output was '0' or in other terms 0V (LOW)

For instance 1 and 0 can represent any of the pairs of opposites shown in the following table.

1 '0'

TRUE FALSE

IN OUT

YES NO

WET DRY

Each operation was accompanied with a truth table that was used as a guide to check whether the circuit was correctly connected, by checking the output Z, with respect to all its inputs.

**ANSWERS TO EXERCISES**

**EXERCISE 2.2**

| A | B | A' | B' | A'.B' | (A'.B')' | A+B |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 |

**EXERCISE 2.5**

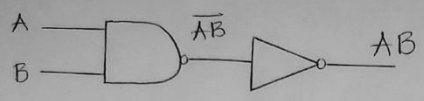
0+0=0

0+1=1

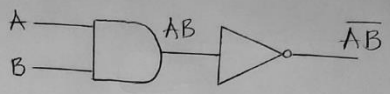
1+1=1

1+0=1

## EXERCISE 2.1




AND gate




NAND gate

## EXERCISE 2.3

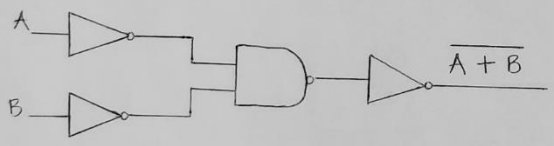
a) $\overline{A\,\overline{B}} = \overline{A} + B$

b) $\overline{\overline{A} + \overline{B}} = \overline{\overline{A}} \cdot \overline{\overline{B}} = AB$

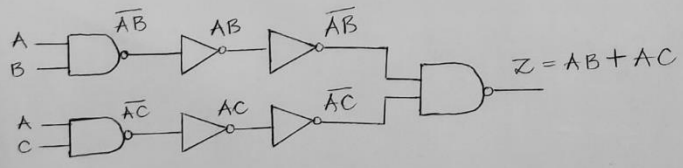
c) $A + \overline{B} = \overline{\overline{A} \cdot B}$

d) $\overline{A + B} = \overline{A} \cdot \overline{B}$

## EXERCISE 2.4

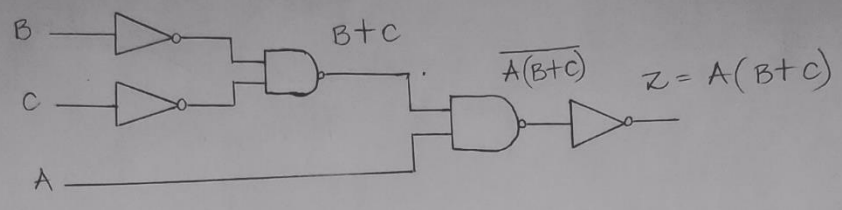



## EXERCISE 3.2

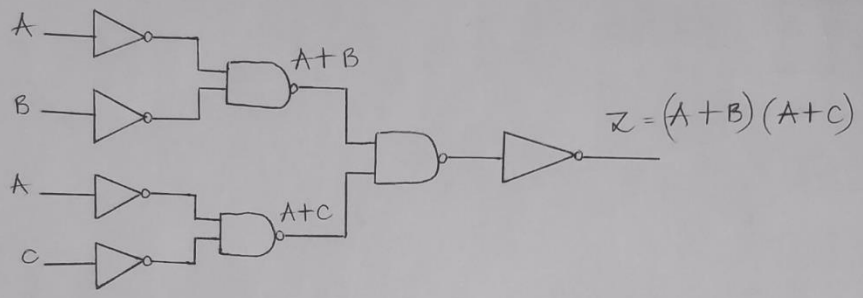


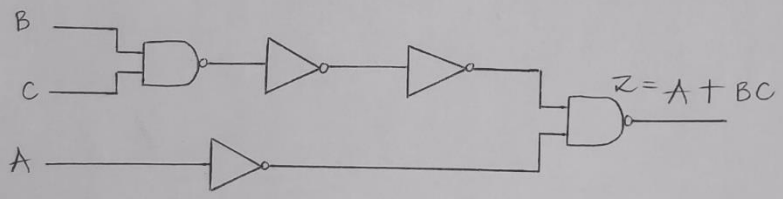

The two inverters cancel each other out

EXERCISE 3.3




EXERCISE 3.4




EXERCISE 3.5

For fig a): $X = \overline{AB}$

$Y = \overline{BC}$

$Z = \overline{\overline{AB} \cdot \overline{BC}}$

fig. b): $Y = \overline{AB}$

$Z = \overline{\overline{AB} \cdot C}$

EXERCISE 3.6

a) $(A+B)(C+D)(B+C)$

$= \overline{AB} + \overline{CD} + \overline{BC}$

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

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

$= \overline{AB + C} + \overline{\bar{A}B + \bar{C}}$ $\Rightarrow$ $AB + \overline{\bar{A}B} + \{C + \bar{C}\}$, $\{C + \bar{C}\} = 1$

$= \overline{AB + \bar{A}B + 1}$

$= \overline{\bar{A}B} + AB + 0$

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

EXERCISE 3.7

a) $A + AB \Rightarrow A(1 + B) = A$

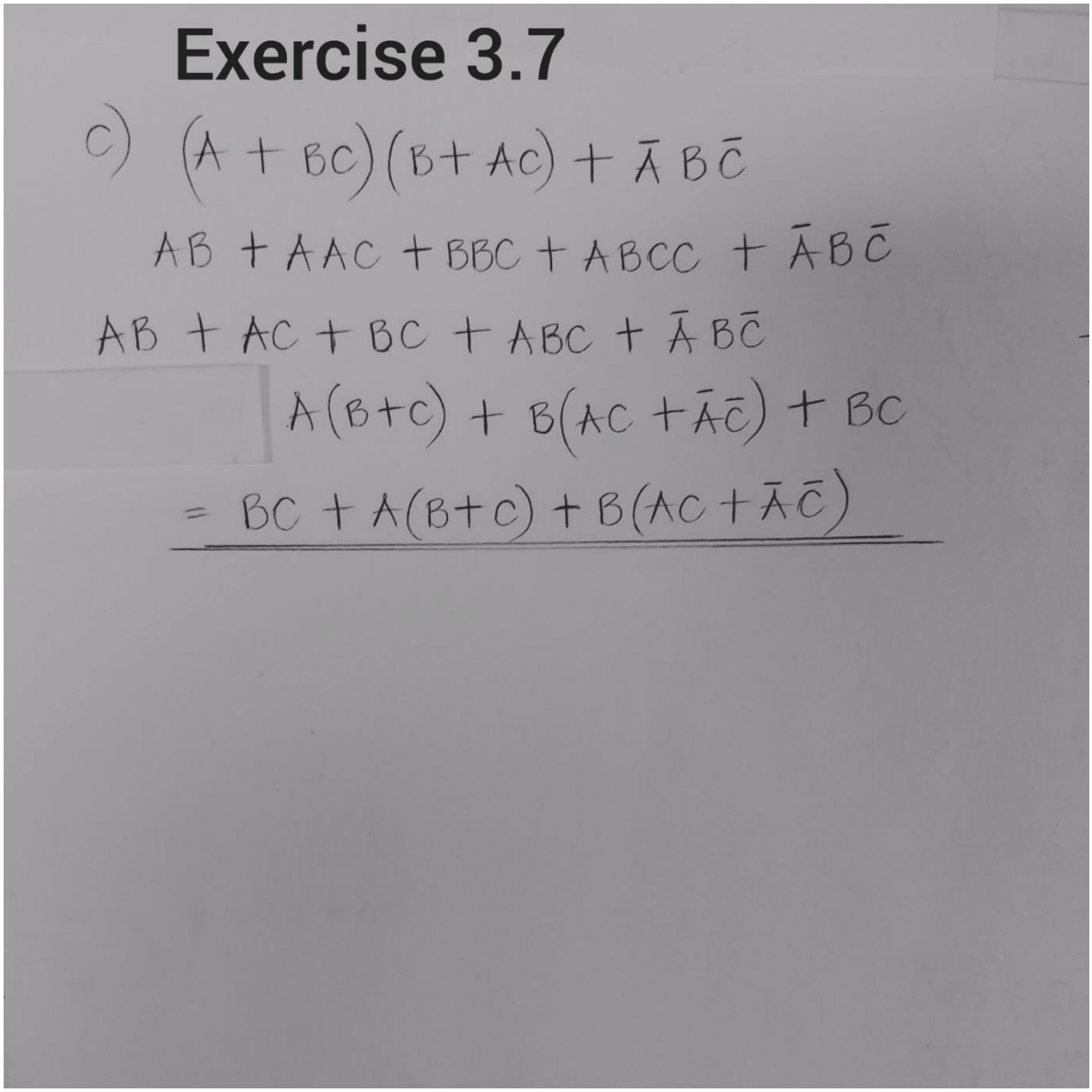
b) $(A + B + C)(A + \bar{B})$

$= AA + BA + CA + A\bar{B} + B\bar{B} + C\bar{B}$

$= A(1 + B + C + \bar{B}) + \bar{B}(B + C)$

$= A(1) + 0 + C\bar{B}$

$= A + C\bar{B}$




EXERCISE 3.1

| A | A’ | 1+A | 0.A | 1.A | A+A | A.A | A+A’ | A.A’ |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |

**8.0 CONCLUSION**

The experiment was successfully done as the objectives were met. All connected circuits responded in accordance with their respective truth tables.

Furthermore, experiments that required a comparison of two sides of an equation were successfully proven to have the same outputs for every logic operation conducted.

A better understanding of logic circuits was attained simply by observing the outputs produced by several different logic circuit connections. The output was observed on an LED which lit up upon a “HIGH” output and was off when output was “LOW” once the software was simulated.

## 9.0 APPLICATIONS

The NAND gate is almost universally used in integrated circuit logic although all types of operation can be obtained, usually at greater cost. However, in designing logic systems, the design tools and techniques are mostly better adapted to the use of the three fundamental operations of AND, OR, NOT so methods of converting the resulting expressions to the exclusive use of NAND have to be used.

Some formal methods do exist but in most practical cases these are heavy handed and cumbersome and have little advantage over simple experience.

The fact that multi-input gates can be used for operations requiring fewer inputs than are available by leaving inputs disconnected means that when designing a large logic system it is easier to ensure that the number of modules (or packages) can be kept to a minimum. This is usually the most important economic factor nowadays.

## 10.0 REFERENCES

[1] William Kleitz, 2006, Digital Electronics with VHDL, Prentice Hall ISBN-100131714902 [Practical 1]

[2] Maini Anil K., Digital Electronics: Principles, Devices and Applications, 2007, John Wiley and Sons Ltd, ISBN 978-0-470-03214-5. [Theory 1]

[3] Thomas L. Floyd, 2006, Digital Fundamentals with PLD Programming, Prentice Hall ISBN-10: 0131701886 [Practical 2]

[4] Sedha R.S, A textbook of Digital Electronics, S. Chand, 2010 [Theory 2]

[5] Alan C. Diixon, JamesL. Antonakos, 2000, A Practical Approach To Digital Electronics, Prentice Hall ISBN-10: 0137275 951.