American Journal of Engineering Research (AJER)
e-ISSN: 2320-0847 p-ISSN : 2320-0936
Volume-13, Issue-6, pp-86-97
www.ajer.org
Research Paper Open Access

Boolean Algebra and Proving Logic Circuits through Logicly

Senad Orhani[1], Xhevdet Spahiu[2*], Yllka Kusari-Radoniqi[3]
[1]*Department of Education, Faculty of Education, University of Prishtina "Hasan Prishtina", Prishtina 10000, Kosovo*
[2]*Department of Computer Science, Faculty of Computer Science, College of AAB, Prishtina 10000, Kosovo*
[3]*Department of Informatics, Faculty of Natural Sciences and Mathematics, University of Tetovo, 1200 Tetovo, Republic of North Macedonia*
Corresponding Author: xhevdet.spahiu@universitetiaab.com

ABSTRACT : *Modern digital computers are built using techniques and symbolism from a field of mathematics called modern algebra. Algebraists have studied for over a hundred years mathematical systems called Boolean Algebra. Nothing could be simpler and more normal for human reasoning than the rules of a Boolean Algebra, because these derive from the studies of how we reason, which reasonings are valid, what constitutes evidence, and other topics. The following design is intended to explain the use of Logicly for validating logic circuits from Boolean Algebra. Logicly is a user interface environment that provides a simple and effective platform for modeling and analyzing these expressions. In this paper, we will explain how we can prove their accuracy. The research results showed that the Logicly program has an excellent performance in validating logic circuits and is an efficient and reliable tool for this purpose. The Logicly program also showed good consistency in the output responses to changes in the inputs of the logic circuits.*
KEYWORDS: *Boolean Algebra, Proving, Logic Circuits, Logicly, Truth table*

Date of Submission: 12-06-2024 Date of acceptance: 24-06-2024

I. INTRODUCTION

Algebra (from the Arabic: al-jabr, meaning "reuniting the broken parts" (Boyer 1991) and "setting the bones" (Claudia and Van Oystaeyen 2017) is one of the most important branches of mathematics, along with number theory, geometry, and analysis. In its form more generally, algebra is the study of mathematical symbols and the rules for manipulating these symbols (Herstein 1965). Algebra is generally considered to be essential to any study of mathematics, science, or engineering, as well as applications such as medicine, economics, computer science, and many other disciplines (Whitesitt 2010).

The name Boolean Algebra honors a fascinating English mathematician, George Boole (1815-1864), who in 1854 published a classic book entitled "An Inquiry into the Laws of Thought, on which the Mathematical Theories of Logic and Probability are Founded" Boole's stated aim was to perform an analysis of mathematical logic. Beginning with his tracing of the laws of logic, Boole constructed a "logical algebra". Boolean Algebra first introduced problems that had arisen in the design of switching relay circuits in 1938 by Claude E. Shannon. He presented a method of representing any circuit consisting of combinations of switches and staffs by a series of mathematical expressions, and an analysis was devised for the manipulation of these expressions. The calculation used was shown to be based on the rules of Boolean Algebra (Schardijn 2016).

Boolean Algebra, like any other axiomatic mathematical structure or algebraic system, can be characterized by specifying a number of basic things (Boole 1854):

- The domain of algebra, that is, the set of elements beyond which algebra is defined
- A set of operations performed on elements
- A set of postulates, or axioms, accepted as premises without proof
- A set of consequences called theorems, laws, or rules, are derived from postulates

Boolean algebra is defined as a set of two elements {0,1}, on which 3 basic operations can be applied: addition, multiplication, and complement. For these three operations, they use the logical operators "or" (OR),

"and" (AND) and "not" (NOT). Instead of the logical operators given above, for the operation of addition and multiplication, in practice, the operators "+" and "·" are also used, while the values of the complement in the literature are mainly marked with a hyphen above the value. The addition and multiplication operations represent binary operations because they are applied to two values, while the complement operation is called a unary operation because only one value participates in it (Dika 2003).

Most of the groups treated in mathematics have an algebraic structure. This is by one or more combined rules which are defined between the elements of the community. More common, are examples of such groups, such as the set of natural numbers, integers, rational, irrational, real, and the set of complex numbers. There are four rules of combination for real numbers: addition, subtraction, multiplication, and division, where the last rule is the exception with division by zero, which is not allowed) (Whitesitt 2010).

Definition 1: A class of elements B together with two binary operations $(+)$ and $(\cdot)$ is a Boolean algebra if and only if the following postulates hold (Whitesitt 2010):

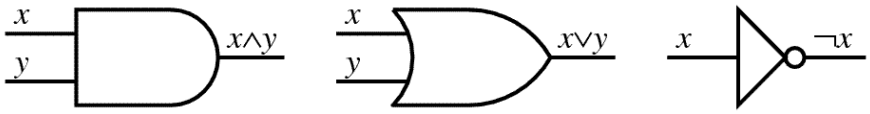
P1. The operations $(+)$ and $(\cdot)$ are commutative.

P2. There is a distinct element of identity B with 0 and 1 compared to the operations $(+)$ and $(\cdot)$.

P3. Each operation is distributive to the other.

P4. For every a in there, B exists an element a' in B such that: $a + a' = 1$ and $aa' = 0$.

While expressions mainly denote numbers in elementary algebra, in Boolean Algebra they denote truth values, false and true. These values are represented by bits (or binary digits), respectively 0 and 1. Digital logic is the application of Boolean Algebra of 0 and 1, to electronic hardware consisting of logic gates connected to form a circuit diagram. Each gate implements a Boolean operation and is schematically described by a shape indicating the operation. The forms associated with the gates for union (AND-gates), division (OR-gates), and complement (inverters) are as follows (Shannon 1949):



Boolean algebra as the calculus of two values is fundamental to computer circuits, computer programming, and mathematical logic, and is also applied to other areas of mathematics, such as set theory and statistics. Today, all modern general-purpose computers perform their functions using two-valued Boolean Algebra logic, that is, their electrical circuits are a physical manifestation of two-valued Boolean logic. By this they achieve this in different ways: as wire voltages in high-speed circuits and storage capacity devices. Most common computer architectures use ordered sequences of Boolean Algebra values, called bits, with 32 or 64 values (Givant and Paul 2009). The structure of a Boolean Algebra is developed from partially ordered sets with lattices and the operations of least upper bound, greatest lower bound, and complement are used to define its properties (Joseph 2013).

The fundamental operators of Boolean Algebra are non-linear. Furthermore, it should be noted that Boolean Algebra is related to linear algebra as it allows its implementation within digital circuits. Thus, Boolean Algebra can be seen as a union between abstract algebra and computational science. Boolean Algebra allows the concise description and manipulation of binary variables (variables), although it is in no way limited to base 2 systems. Variables in Boolean Algebra have a unique characteristic, where they can take only one of two possible values. Variables that can take only two logical values, such as 0 and 1, are known as Boolean variables, or logical variables. Therefore, if $x \neq 0$, then $x = 1$ and if $x \neq 1$, then $x = 0$ (Dika 2003).

Table 1. Variables

Value (Bits)	Alternative name
0	F, False, No, OFF, LOW
1	T, True, Yes, ON, HIGH

On the other hand, Logicly is a user interface environment used for modeling and analyzing logic circuits. It is a computer program developed for use in Boolean Algebra and computer science classes to visualize and experiment with logical expressions. With the help of Logicly, you can create logic circuits using various components built into the software. This program provides a wide range of logic circuits such as AND Gate, OR Gate, NOT Gate, NAND Gate "not and"), "NOR Gate" (circuit "not or"), "XOR Gate" (circuit "exclusive or"), "XNOR Gate" (circuit "exclusive not or") and many others. Through a simple and intuitive interface, Logicly allows you to connect the inputs of logic circuits and see how the output values change when

the input values change. This program is a useful tool for understanding circuit logic and expanding knowledge of Boolean Algebra (Tynjala, 2020).

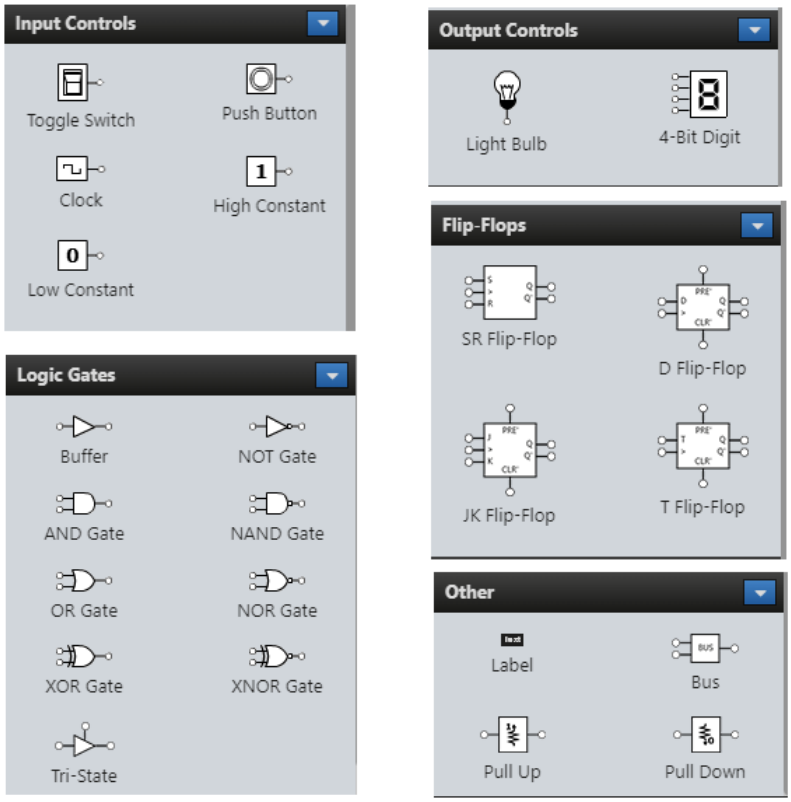


Fig. 1. Elements of the Logically software

Therefore, the main problem of this research is to give an insight into Boolean Algebra from a mathematical perspective. Also, the paper identifies a problem of significant importance, which begins with an expertise of the Logicly program for validating logic circuits from Boolean Algebra, which addresses some challenges that are related to the understanding and modeling of logical expressions.

II. SCIENTIFIC RESEARCH METHODS

Boolean Algebra with the modeling of logic circuits in Logicly, using experiments and analysis to use this knowledge efficiently. First, an in-depth study of Boolean Algebra and formal logic will be conducted. This includes knowledge of logical operations such as "AND", "OR", and "NOT", as well as truth tables and rules associated with these operations. Logicly 's user interface will then be used to create various logic circuits and connect their inputs and outputs. Here we will experiment with changing the input values to observe how the output values change. In the next step, we will analyze the outputs modeled in Logicly and compare them with the previously prepared truth tables. If the modeled outputs match the truth table values, the logic circuits will be validated as correct. If any discrepancies are encountered, the researchers will look for the reasons for the errors and correct their models. Finally, we will perform additional experiments with Logicly, analyzing other logic circuits and using different input changes to see the effect on the output. They will create concrete examples and explain their results by referring to lectures and knowledge gained from the study of Boolean Algebra.

III. PURPOSE OF THE RESEARCH

The purpose of the research in this scientific paper is to use the Logicly program to analyze and validate different logical circuits from Boolean Algebra. The use of Logicly is intended to expand the understanding of formal logic and Boolean Algebra concepts, exploring the modeling of logic circuits visually and interactively. However, the specific goal is to analyze and prove the basic concepts of Boolean Algebra, such as the logical operations "AND", and "OR" and the relationships between them by applying them to the binary, octal, and hexadecimal number systems.

IV. RESEARCH HYPOTHESES

Main hypothesis:

H0: Using the Logicly program will allow correct validation of logic circuits other than Boolean Algebra.

H1: Using the Logicly program will not allow correct validation of logic circuits other than Boolean Algebra.

Input impact hypothesis:

H0: Changing the input values in the logic circuits will not affect the output given by the Logicly program.

H1: Changing the input values in the logic circuits will affect the output given by the Logicly program.

The reliability hypothesis of the Logicly program:

H0: The Logicly program will produce reliable and accurate output for all logic circuits modeled.

H1: The Logicly program will produce inaccurate and unreliable output for some modeled logic circuits.

V. MEASURING TECHNIQUES OR INSTRUMENTS

With the help of measuring techniques or instruments in this research, the collection of complex information will be done for the illumination and study of problems related to Boolean Algebra. The research techniques or instruments in this research represent the dynamic elements in the realization of this study, which include the methods and tools that will be used to collect data and evaluate the performance of the Logicly program in validating logic circuits. These techniques and instruments are important to achieve the objectives of the research and to test the hypotheses presented in the study. Here are some of the techniques and measuring instruments used. The main technique in this study is the modeling of logic circuits with the Logicly program. Using this software will allow us to create and analyze logic circuits using different logic components. Through modeling in Logicly, we will evaluate the program's ability to produce the correct output for each configuration of inputs. Preparing truth tables for selected logic circuits is another important measurement technique. These tables will help us to compare the outputs produced by the Logicly program with the appropriate output values for each combination of inputs. This rating will be used to evaluate the accuracy of the program. Also, we will use the results comparison technique to evaluate how well the outputs produced by the Logicly program match the correct values of the truth tables. This comparison will evaluate the efficiency of the program and reveal potential cases of errors. Thus, they will analyze the differences in the outputs of the Logicly program and the values of the truth tables. This analysis will help to identify possible cases of errors and discover the reasons for output changes.

VI. RESEARCH RESULTS

The interpretation of the results from this research gives us an approach to the processes through which these data are reviewed to reach an informed conclusion about the topic studied. The interpretation of the results gives an understanding of the information analyzed by Boolean Algebra and the performance of the Logicly program is evaluated in comparison with the presented truth tables. Below we present the obtained results:

Table 2. Truth table - Binary logic operation "AND" Gate

A	B	Light Bulb	4-Bit Digit
0	0	0	0
0	1	0	1
1	0	0	2
1	1	1	3

Table 2 presents a truth table for the logical operation "AND" (and) in a binary logic frame entry. In this table, "A" and "B" are the two logic inputs which can be either 0 (false) or 1 (true). The result of this logical operation is displayed in the last two columns: " Light Bulb" and "4-Bit Digit ".

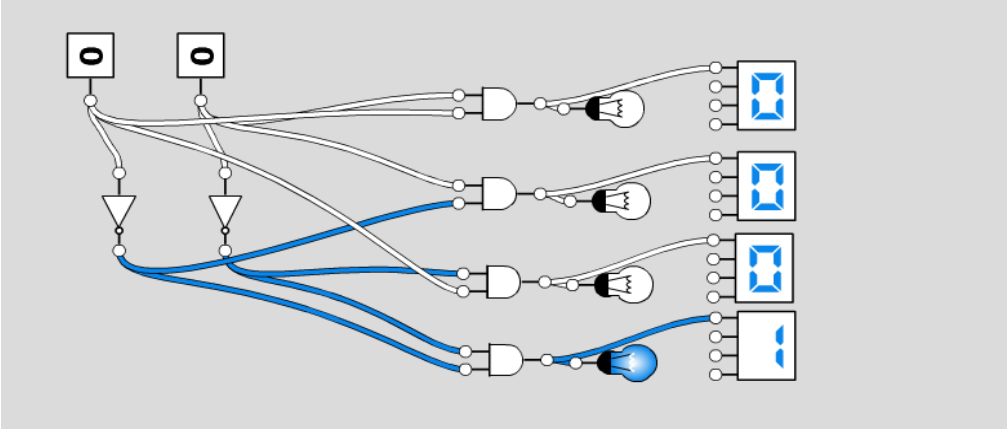


Fig. 2. The digital circuit realized with the logical "AND" operation for the binary system

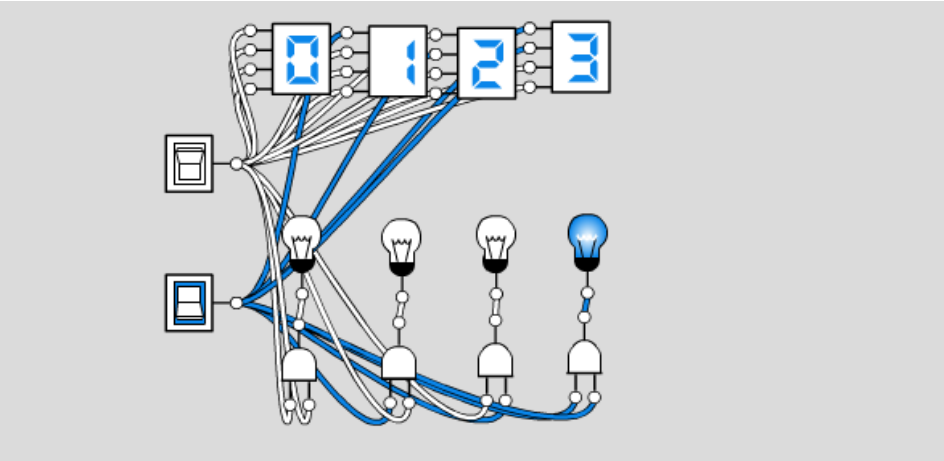


Fig. 3. Digital circuit with results in numerical indicators for the binary system

The first figure shows the realization of the digital circuit using the logic circuit "AND" with two inputs named with the letters A and B. Each output port of the logic circuit "AND" is connected to an electrical pot and a numerical indicator.

Based on the results obtained for the logical operator "AND", the output of this circuit shows the digit "1" when and only when both inputs are in the "1" state, from the circuit realized in the Logicly program we see that in the application of two inputs "0", one "0" and one "1", as well as the opposite "1" and "0", the output in the three numerical indicators is "0" and the electric pots are not turned on. In the fourth case where two "1" signals are applied to the input, the numerical indicator shows the number "1" at the output, and the electric pot is also switched on.

In the second picture, you can see the circuit made to verify the use of the Logicly program in the circuits with output to the 4-bit code, which according to the truth table, based on certain inputs, the circuit must output the numbers 0, 1, 2 and 3. From testing the circuit in the program we see that the outputs are identical to the truth table showing the numbers 0, 1, 2, and 3.

Table 3. Truth table - Binary logic "OR" Gate operation

A	B	Light Bulb	4-Bit Digit
0	0	0	0
0	1	1	1
1	0	1	2
1	1	1	3

Table 3 presents a truth table for the logical operation "OR" (or) on an input of the binary logic frame. In this table, "A" and "B" are the two logic inputs which can be either 0 (false) or 1 (true). The result of this logical operation is displayed in the last two columns: " Light Bulb" and "4-Bit Digit".

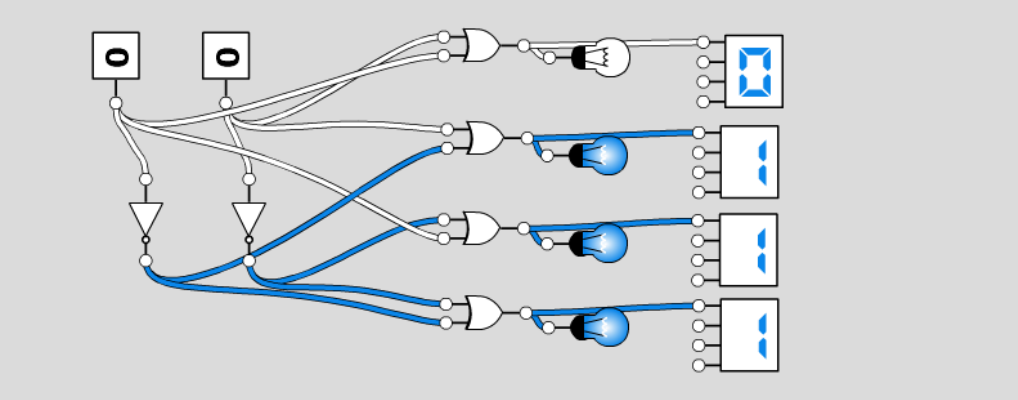


Fig. 4. The digital circuit realized with the logical "OR" operation for the binary system

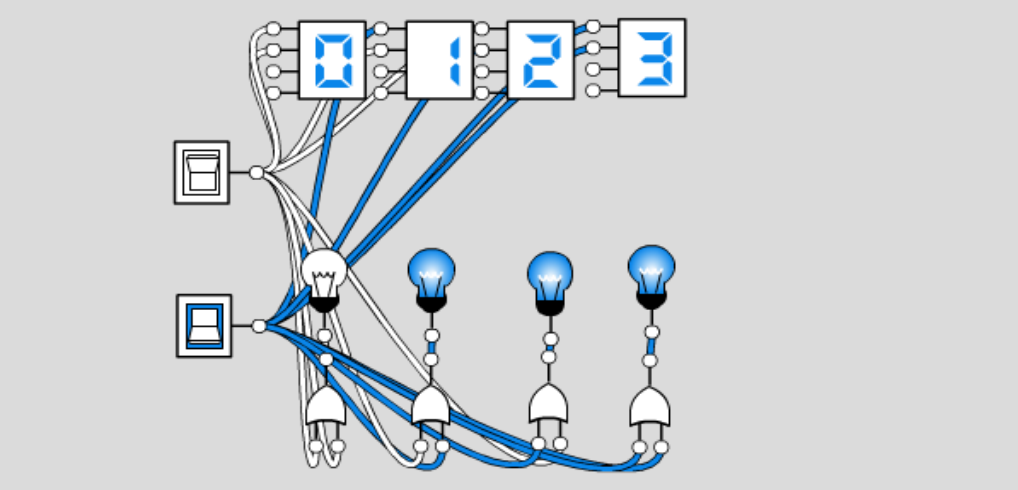


Fig. 5. Figure 6 Digital circuit with results in numerical indicators for the binary system

In the truth table, we have noted the results obtained by using the logical operation "OR". In the same way as in the previous table, two inputs named the letters A and B are used, while the output signal is marked with "1" in the case of ignition of the pot and "0" when the pot is stopped. In the last column of the table, the results obtained for the presentation of the number in the numerical indicator are noted.

The circuits in both figures are implemented like the circuits from the previous simulation, but now the logical operator "OR" is used. Each output gate of the "OR" logic circuit is connected to an electrical pot and a numerical indicator. The Logicly program is quite practical by providing several shapes that can be used to represent the input signal. Such a thing can be seen in the two figures where the first circuit is connected to two inputs with a preliminary value of "0" and then to obtain the "1" values of the inputs, the logic circuits "NO" are used - the inverter or otherwise called inversion. While in the second circuit, toggle switches were used Switch.

From the circuits realized and tested in the Logicly program, it is confirmed that the obtained results are in agreement with the results in the table as in the case of lighting the pots in all cases except when both input signals are "0", also in the numbers presented in numerical indicators.

Table 4. Truth table - Logical octal "AND" Gate operation

A	B	C	Light Bulb	4-Bit Digit
0	0	0	0	0
0	0	1	0	1
0	1	0	0	2

0	1	1	0	3
1	0	0	0	4
1	0	1	0	5
1	1	0	0	6
1	1	1	1	7

Table 4 shows a truth table for the logical AND operation on an octal logic frame entry. In this table, "A", "B", and "C" are the three logic inputs, which can be either 0 (false) or 1 (true). The result of this logical operation is displayed in the last two columns: " Light Bulb" and "4-Bit Digit ".

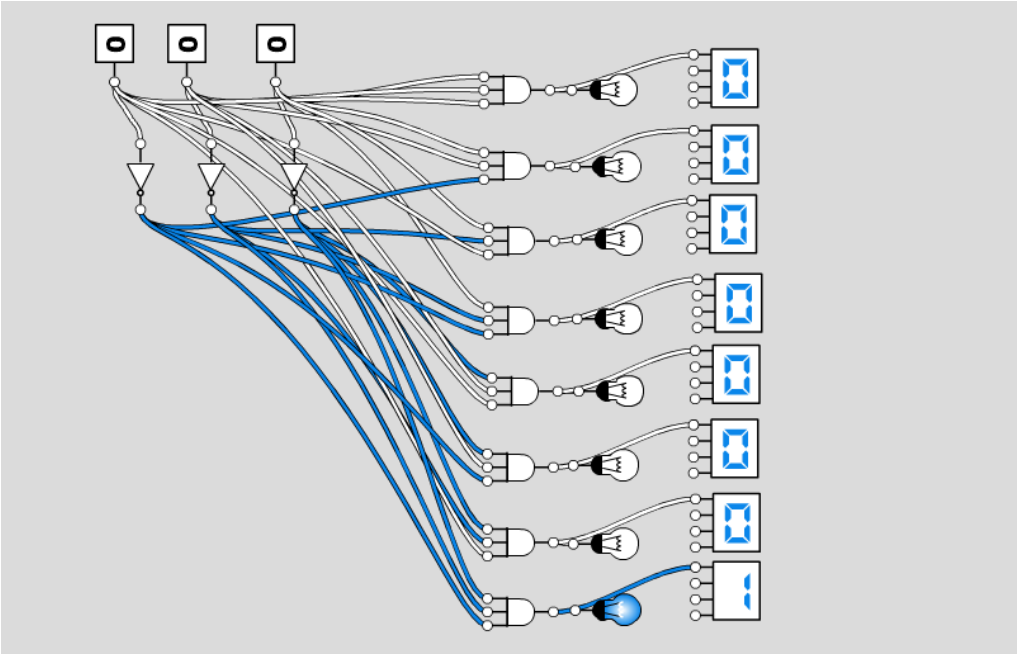


Fig. 7. The digital circuit realized with the logical "AND" operation for the octal system

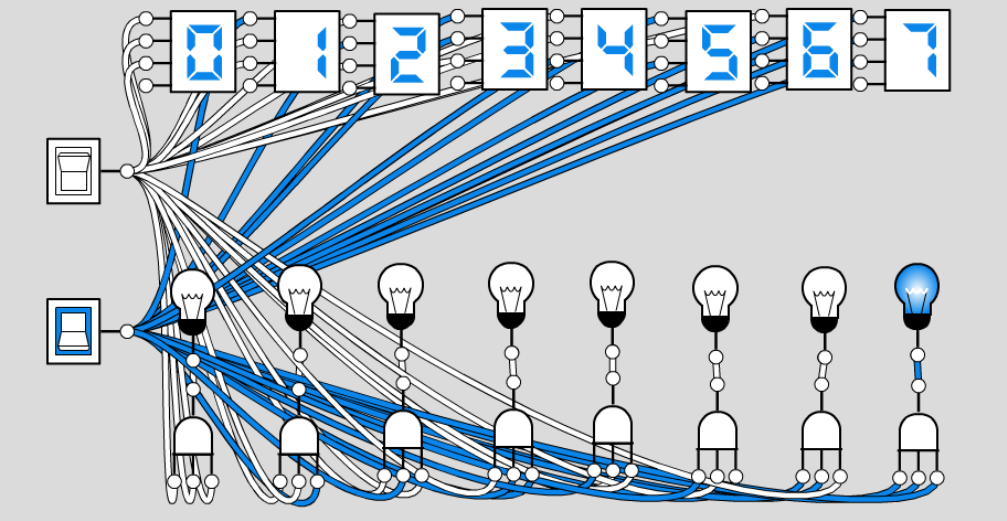


Fig. 8. Digital circuit with results in numerical indicators for the octal system

To obtain output in the octal numerical system, three input signals were used, named with the letters A, B, and C. By using 2n =23=8 combinations of bits of the input signals, the outputs were obtained so that only in the case when the three input bits are "1" bits, the output result of the logical operator "AND" will be "1".

In the Logicly program, we have implemented the two circuits as shown in the figure, to each logic circuit "AND" we have connected three input signals with values as in the table. The results from testing in

these circuits give the same values as in the table, being demonstrated in the first circuit with the ignition of only one pot, the last one which is connected to the logic operation "AND" with three inputs with values of "1". All other outputs have a value of "0" and the condition for the logical operator "AND" is fulfilled that only in the case of all inputs with a value of "1" the result will be "1".

In the second circuit, it can also be seen that in the numerical indicators in which the input data are given as in our table, the output shows the numbers from 0 to 7, showing that up to this point, the results given by our research are inconsistent and the same as those tested in the Logicy program.

Table 5. Truth table - Logical octal "OR" Gate operation

A	B	C	Light Bulb	4-Bit Digit
0	0	0	0	0
0	0	1	1	1
0	1	0	1	2
0	1	1	1	3
1	0	0	1	4
1	0	1	1	5
1	1	0	1	6
1	1	1	1	7

Table 5 presents a truth table for the logical OR operation on an octal logic frame input. In this table, "A", "B", and "C" are the three logic inputs, which can be either 0 (false) or 1 (true). The result of this logical operation is displayed in the last two columns: "Light Bulb" and "4-Bit Digit ".

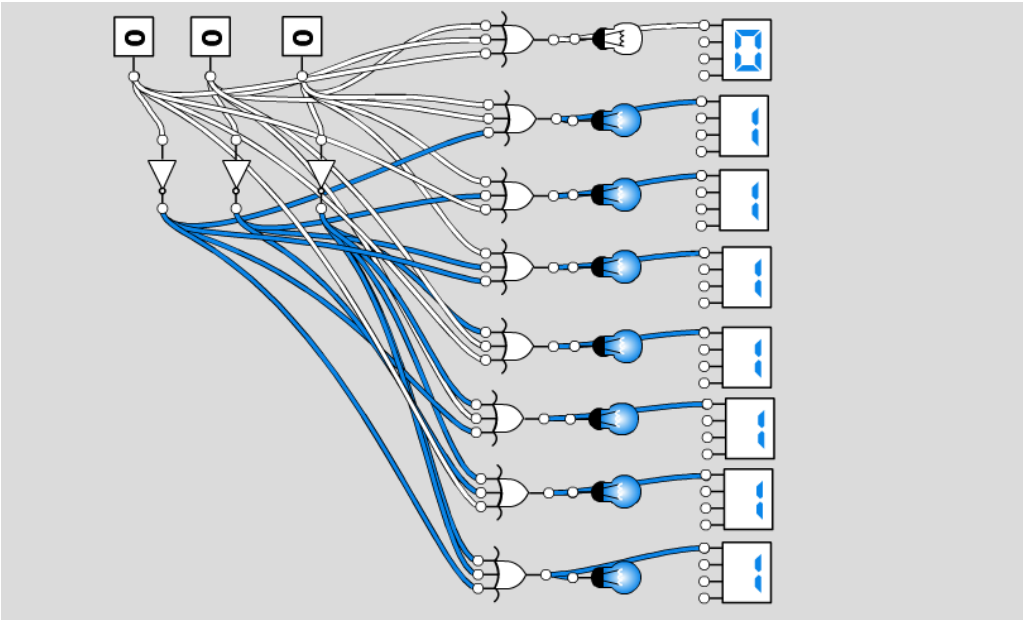


Fig. 9. The digital circuit realized with the logical “OR” operation for the octal system

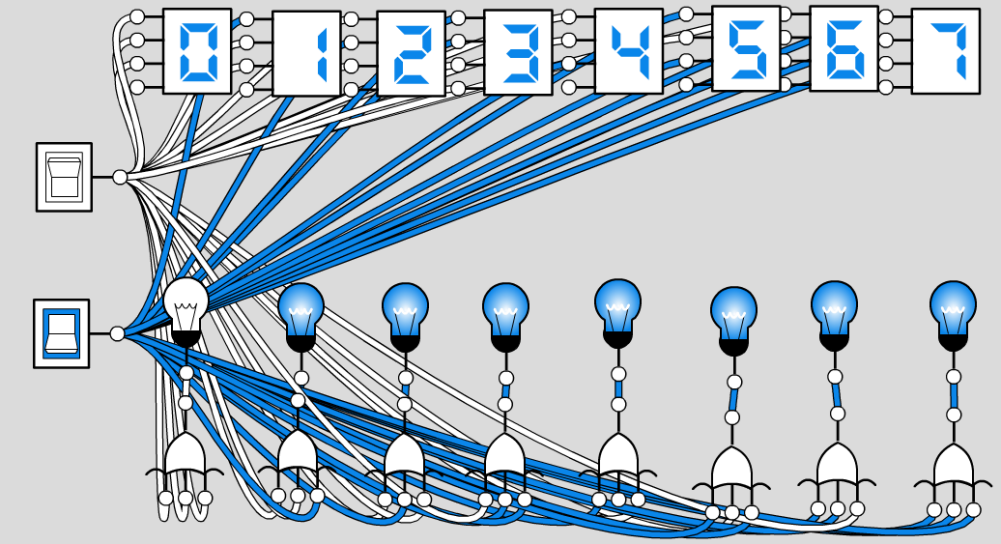


Fig. 10. Digital circuit with results in numerical indicators for the octal system

In the implementation of the logical operator "OR" based on our research, we have concluded that this circuit will have a value of "1" at the output if one of the inputs will have a value of "1". We will prove this through the circuit made with three inputs A, B, and C in the Logicly program.

In the first circuit realized by applying the binary values of the three signals at the input of each logical operation "OR" as in the given table we see that the output results are coherent and in line with those in the table, while in the second circuit the numerical indicators using received as input signals "0" and "1" according to the order in the table, show the numbers of the octal system 0 to 7.

Table 6. Truth table - Hexadecimal logic operation "AND" Gate

A	B	C	D	Light Bulb	4-Bit Digit
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	0	2
0	0	1	1	0	3
0	1	0	0	0	4
0	1	0	1	0	5
0	1	1	0	0	6
0	1	1	1	0	7
1	0	0	0	0	8
1	0	0	1	0	9
1	0	1	0	0	10 (A)
1	0	1	1	0	11 (B)
1	1	0	0	0	12 (C)
1	1	0	1	0	13 (D)
1	1	1	0	0	14 (E)
1	1	1	1	1	15 (F)

Table 6 presents a truth table for the logical AND operation on a hexadecimal logic frame entry. In this table, "A", "B", "C", and "D" are the four logic inputs, which can be either 0 (false) or 1 (true). The result of this logical operation is displayed in the last two columns: "Light Bulb" and "4-Bit Digit".

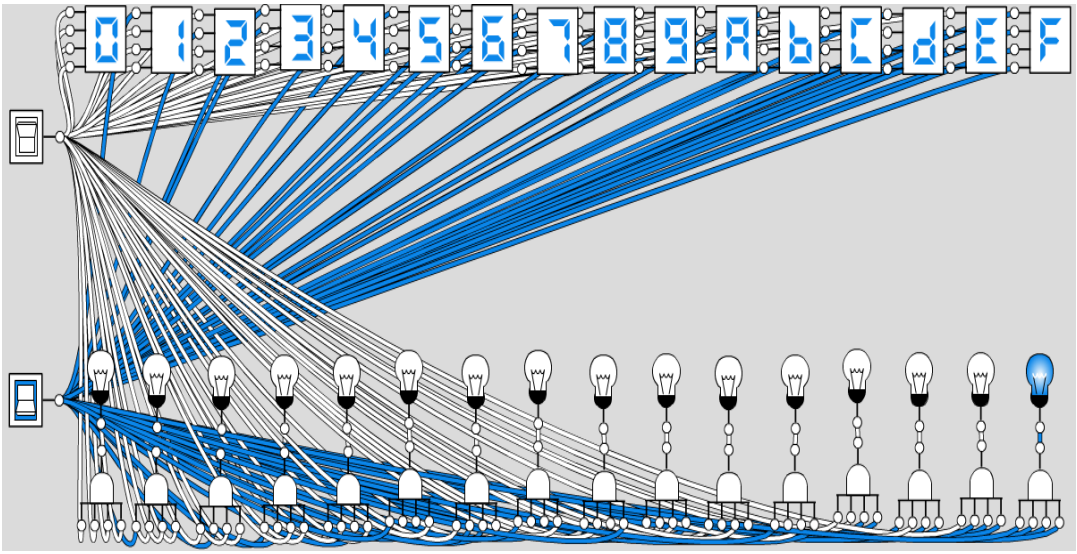


Fig. 11. The digital circuit realized with the logical "AND" operation and in numerical indicators for the hexadecimal system

To interpret the results from the truth table given above for the AND circuit, we say that four input signals are used to obtain the combination of bits for the hexadecimal number system. According to our results in the truth table, they are such that only in the case of all inputs with a signal of value "1" the result will be "1". In other cases, the results are "0" for all modes of input bit combinations.

The circuit realized with the logical operation "AND" in the program Logicly gives us the same results by not turning on the pots in the fourteen cases of incoming signals and turning on only the pot - whose logical operation has four input signals "1". On the other hand, the numerical indicators show the numbers of the hexadecimal system 0 – F, thus confirming the results from the table.

A characteristic of the presentation of results in numerical indicators for hexadecimal numbers which are written in the letters A, B, C, D, E, and F is observed in the numbers "B" and "D" which are presented in lowercase letters "b" and " d". Numeric indicators in Logicly are 7-segment indicators, so the presentation with lowercase letters is to distinguish the number "B" from "8" and the number "D" from "0".

Table 7. Truth table - Hexadecimal logic operation "OR" Gate

A	B	C	D	Light Bulb	4-Bit Digit
0	0	0	0	0	0
0	0	0	1	1	1
0	0	1	0	1	2
0	0	1	1	1	3
0	1	0	0	1	4
0	1	0	1	1	5
0	1	1	0	1	6
0	1	1	1	1	7
1	0	0	0	1	8
1	0	0	1	1	9
1	0	1	0	1	10 (A)
1	0	1	1	1	11 (B)
1	1	0	0	1	12 (C)
1	1	0	1	1	13 (D)
1	1	1	0	1	14 (E)
1	1	1	1	1	15 (F)

Table 7 presents a truth table for the logical "OR" operation on a hexadecimal logic frame input. In this table, "A", "B", "C", and "D" are the four logic inputs, which can be either 0 (false) or 1 (true). The result of this logical operation is displayed in the last two columns: "Light Bulb" and "4-Bit Digit".

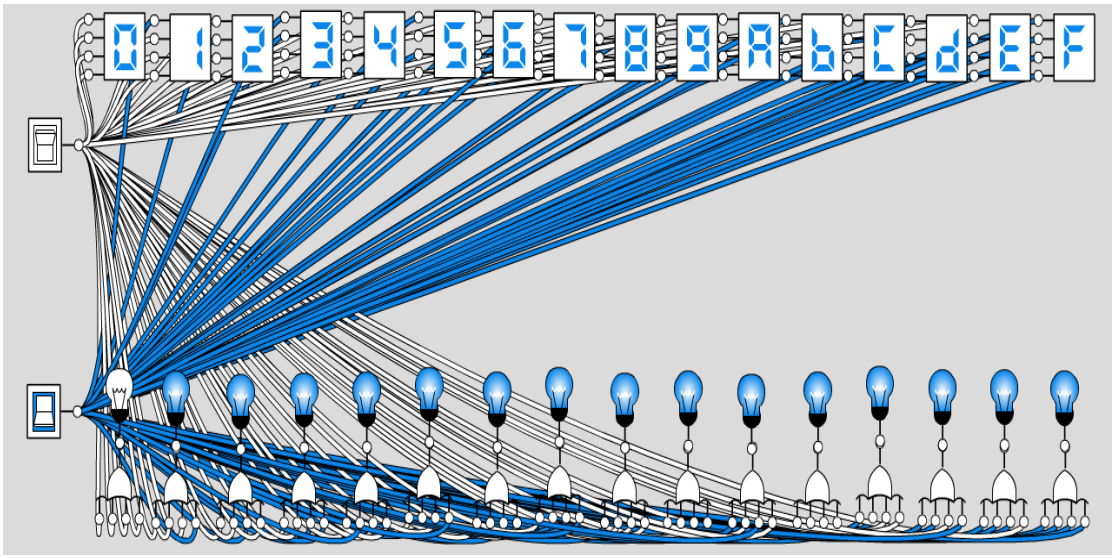


Fig. 12. The digital circuit realized with the logical "OR" operation and in numerical indicators for the hexadecimal system

In all the earliest cases it can be seen that the results obtained from our research are very well realized and have been proven correct in their testing through the Logicly program. Let's look at the circuit implemented using the logical OR operation for the hexadecimal number system.

There are used fifteen logical "OR" operations configured with four inputs each in order to obtain output in the hexadecimal number system. From the realization of the circuit, we confirm that our results in the table match the results from the circuit testing by turning on all the pots, except for one in the case when all four input signals are "0". Also, the numerical indicators, as in the previous circuit, from the combinations of four bits at the input of each indicator, at its output show the numbers of the hexadecimal system 0 to F, as in our results from the table.

VII. DISCUSSION AND CONCLUSION

Algebra is used by scientists and engineers everywhere. As a result, the concepts of algebra are very important to scientists. Boolean Algebra can be equally useful if applied to the field of computer science. When George Boole first developed symbolic logic (Boolean logic), he had the idea that his system could be used by mathematicians to help put complicated arguments into a more convenient position. Little did he know that his system of "and," "or" and "not" operations would one day transform the world, ushering in the digital revolution and the modern-day computer. Instead of true and false truth values, digital computers rely on two states, either (1) or (0). This is because a computer consists of many circuits, which are electrical paths that can be closed to allow current to flow, or opened to break the connection. A "1" will mean a closed loop, while a "0" will represent an open loop. Therefore, this research supports the hypotheses presented, with which we argue the importance identified for the common rules related to the logic of Boolean Algebra, which can be applied in the studied field.

The purpose of the research in this scientific paper is to use the Logicly program to analyze and validate logic circuits different from Boolean Algebra. In particular, the purpose of this research is to analyze and prove the basic concepts of Boolean Algebra, such as the logical operations "AND", "OR", and "NOT", as well as relations among them. Also, the specific goal is to analyze and prove the basic concepts of Boolean Algebra in the context of binary, octal, and hexadecimal number systems. Based on the results presented in the previous results chapter, we examine their interpretation and give a deeper meaning to our conclusions.

The results showed that the Logicly program has an excellent performance in validating logic circuits other than Boolean Algebra. Through modeling in this software, a good fit was achieved between the produced outputs and the presented truth tables. This helped to accept the main hypothesis, which shows that the Logicly program is an efficient and reliable tool for validating logic circuits.

Analysis of the impact of input changes on the outputs produced by the Logicly program showed that the program was consistent in its responses. Changing the input values in the logic circuits changed the output values and this accepted the corresponding alternative hypothesis. This result marks a high quality of the program, proving that it can handle input changes correctly.

Through analyzing the differences in the Logicly output and the truth tables, we identified several cases where the differences had appeared. This analysis provided an assessment of the program's ability to use logic circuits and help researchers understand the reasons for the changes. According to our results, we can also prove the null hypothesis that the Logicly program produces reliable and accurate outputs for all modeled logic circuits.

Using this software allows us researchers to model, analyze, and validate logic circuits with ease and precision. In-depth knowledge of Boolean Algebra influenced users' ability to use the program successfully. With good interactivity, the Logicly program eased the work of validating logic circuits. This discussion concludes that the Logicly program is a valuable and recommended tool for the analysis and validation of logic circuits and is useful in the field of mathematical logic and electrical engineering.

In conclusion, this scientific work has achieved its goals of using the Logicly program for validating logic circuits from Boolean Algebra. It is understood that the search for formal logic and Boolean Algebra concepts can be extended and deepened by relying on available technology. The use of the Logicly program and its positive evaluation contribute to the development of knowledge in this field, and the dissemination of knowledge about formal logic in the scientific and academic community.

Through this scientific paper, it was proved that using the Logicly program provides an easy and efficient way to model and analyze logic circuits and its accurate validation makes it known as a useful tool for academics, engineers, and researchers. The results showed that the Logicly program has an excellent performance in validating logic circuits and is an efficient and reliable tool for this purpose. The Logicly program also showed good consistency in the output responses to changes in the inputs of the logic circuits. This research has evaluated several important concepts of Boolean Algebra, including logical operations "AND", "OR", and "NOT" and the relationships between them. This has helped me to recognize and understand the applications of these concepts in the binary, octal, and hexadecimal number systems.

Based on the conclusions of this research, we recommend the use of the Logicly program in the context of learning and developing knowledge about formal logic and Boolean Algebra. This software provides a powerful tool to explore and understand logic circuits visually and interactively. Also, this scholarly work can serve as a reference source for all those interested in formal logic, electrical engineering, and computer science.

One of the limitations of this research was the number of logic circuits modeled in the Logicly program. Due to time and resource constraints, it was not possible to analyze all types of logic circuits. This limited the application of the research to some aspects of Boolean Algebra. Future research can be extended by analyzing more complex logic circuits and using other programs for their modeling and validation. The impact of using the Logicly program in teaching Boolean Algebra and formal logic can be examined.

REFERENCES

[1]. Boole, GB (1854). An Investigation of the Laws of Thought in. Dover Publications, Inc.
[2]. Boyer, C. B. (1991). A History of Mathematics. John Wiley & Sons, Inc.
[3]. Claudia, M., & Van Oystaeyen, F. (2017). Abstract Algebra: A Comprehensive Treatment. CRC Press.
[4]. Dika, A. H. (2003). Qarqet Kompjuterike Kombinuese 1. Universieti i Prishtinës.
[5]. Givant, S., & Paul, H. (2009). Introduction to Boolean Algebras. Undergraduate Texts in Mathematics. Springer.
[6]. Herstein, S. (1965). An algebraic system can be described as a set of objects together with some operations for combining them. Ginn and Company.
[7]. Joseph, G. A. (2013). Contemporary abstract algebra. Paperback.
[8]. Schardijn, A. (2016). An Introduction to Boolean Algebras. Semantic Scholar.
[9]. Shannon, C. (1949). The Synthesis of Two-Terminal Switching Circuits. System Technical Journal.
[10]. Tynjala, J. (2020). Logic.ly 1.12: Disable strict mode, edit IC names, limit propagation mode, edit IC names and limit propagation. Gjetur në Logic.ly: https://logic.ly/blog/2020/07/logic-ly-1-12-disable-strict-mode-edit-ic-names-and-limit-propagation/
[11]. Whitesitt, E. (2010). Boolean Algebra and its Applications. Dover Publications.