ALU: how mathematical logic creates modern computing

Ahmed Walid
Yousef Abd El-Nabi

Abstract- It is fascinating how humans can find a simple idea and use it to create complex creations. This is the case with mathematical logic and logic gates. This paper goes over how Boolean algebra and logic gates are utilized to create an important part of the CPU. It also discusses how the concepts of logic gates can be simulated in electrical circuits via the use of transistors. It showcases logic gates by showing how a simple Arithmetic Logic Unit would be created using them, showing the different components and describing the binary operations in the process.

Index Terms- Boolean Algebra, Logic gates, Transistors, ALU.

I. INTRODUCTION

Among the history of the humanity, development has become a very basic routine. The development of traffic, writing tools, and medicine are such an example. However, for such developments, achieving the current level of progress, did take tens or even hundreds of decays. On the other hand, while talking about computers, the pattern is totally broken. In just 200 years, from a machine that can hardly compute several sets of numbers, the humanity was able to build the current computers that can make thousands of millions of hard calculations in a few seconds. Also, when taking about computers, the shock is not in just its function development. Its size and weight development are also distinct. For the first computing system, its length has exceeded 50 feet (15 meters), and it was weighting about five tons (this is equivalent to the mass of an adult elephant). Considering the current smart phones that can be put in your pocket and does not exceed 500 grams in mass, this deference is totally chocking. In the coming sections, the technology used in such development has been explained. Considering its huge role for such development.

Identify the constructs of a Journal – Essentially a journal consists of five major sections. The number of pages may vary depending upon the topic of research work but generally comprises up to 5 to 7 pages. These are:

II. BOOLEAN ALGEBRA

Boolean Algebra is algebra that deals with objects having only two types of values, either True or False. It was created by George Boole as he created a systematic system of logic and for this purpose, he developed Boolean Algebra or Symbolic logic. It is a branch of mathematics that can be used for manipulating and describing binary information. Therefore, it has various applications in modern computing.

A Boolean function is a function in Boolean algebra that takes one or more inputs and provides an output result of either True or False. They can be used with Boolean operators for certain constraints on the

output depending on the logical statements. The main Boolean operators are AND, OR, and NOT. Boolean operators can be described using tables, called truth tables.

Inputs		Outputs
x	y	xy
0	0	0
0	1	0
1	0	0
1	1	1

Figure 1 shows truth table of AND operator

The AND truth table shows the relationship between the inputs and outputs of the AND operator as shown in **Figure 1**. AND operator is used with two inputs or more, where both inputs have to be true for the whole expression to be true. For example, if the first statement is "The sky is blue" is true, and the second statement "The earth is round" is also true, then the whole expression is true. That is, since both expressions are true. On the other hand, if one of the two statements is false, that makes the whole expression false. The table shows true and false as 0 and 1. The output is 1 only when both inputs are 1, and it is 0 in every other case. [9], [10], [11]

The OR truth table is shown in **Figure 2**. OR operator is also used with two or more inputs. One of the inputs has to be true in order for the output to be true. For example, if the first statement is "The sky is red" which is false, but the second statement is "Light is so fast" which is true. Therefore, the whole statement is true. [9], [10], [11]

Inputs		Outputs
x	y	$x+y$
0	0	0
0	1	1
1	0	1
1	1	1

Figure 2 shows the OR truth table

The NOT truth table is shown in **Figure 3**. NOT operator can be used with one input and the output is the opposite of the input. It is similar to the negative sign where when you add it to something it negates it. For

Inputs	Outputs
x	$\overline{x}$
0	1
1	0

Figure 3 shows the NOT truth table

example, if we have the statement “The sky is blue” followed with a NOT before it. The output is “the sky is NOT blue”. [9], [10], [11]

Those three Boolean operators were utilized to create logic gates, which were used to create more complicated components. At the end, we ended up with modern computers, which are able to perform complicated operations. All of that, starting with the very simple idea of using logic to represent two values.

The AND operator can be written as the multiplication sign. I

III. TRANSISTORS

Transistor, the fire of the 20th century, is a very small devise that is used as an amplifier or a switch in electric circuit. It is made of materials called semiconductors (materials that has the ability of both conducting and insulating the electric current). The reason behind that is to give it the ability to either conduct the electric current or not. In the language of Boolean Algebra, this ability will be responsible for the 0 and 1 values. However, to clearly understand how transistors are used in any electrical circuits, analysis of the structure of the transistor is needed.[1]

While working with Boolean algebra, transistors utility as fast switches have made them the best choices in creating the logic gates. However, transistors vary in type, and each type have a distinct structure. However, while working with the logic gates, the type that is commonly used, is the (BJT) Bipolar Junction Transistors. [2]

As said before, the building material of the transistor is a semiconductor material (mainly silicon and germanium). This is due to its ability of whether conducting the electric current or not. However, silicon in commonly used due to its availability. In the BJTs, there is always two distinct regions integrated together. Both is made of a semiconducting material, but the reason for the positive and negative sign is the material added to the semiconductor.

In the n-type region, the silicon is mixed with a small ratio of arsenic or phosphorus. Such materials have five electrons in their outermost shell. As shown in **Figure 4**, when such a material makes covalent bonds with silicon, only four electrons are bonded. Thus, the fifth electron will have the ability to move freely, making the n-type region have a negative charge.[3]

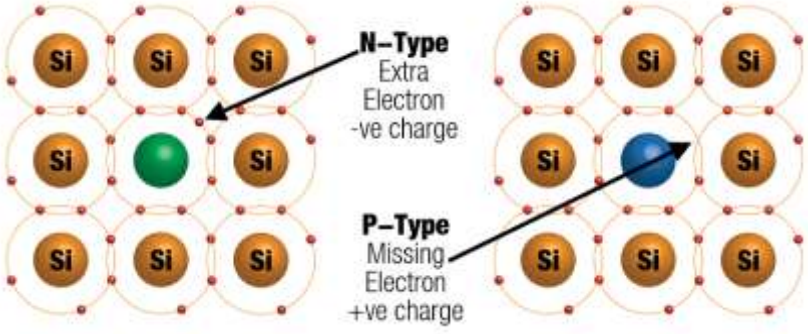

Figure 4 shows the two types of semiconductors

On the other hand, when talking about the p-side, this time the silicon is mixed with boron or gallium. Such materials have only three electrons in their outermost shell. So, while interacting with the silicon atoms, the boron makes three single covalent bonds with the silicon atoms. For the fourth bond, the boron atom will steal one of the electrons of the silicon atoms. Thus, due to the absence of the electrons, a hole is formed. However, in the p-side material, the number of holes is greater than the number of electrons. Thus, the p-side is positively charged.[4]

To well understand how transistors work, first, a simpler version called the diode is discussed. In a diode, a n-type and p-type materials are connected as shown in **Figure 5.** At first, trying to reach an equilibrium state, the electrons from the n-type materials and the holes from the p-type materials are trying to switch places. Fortunately, this phenomenon does not happen. The ions formed due to the formation of the free electrons and holes form a barrier between the n-type and p-type materials. Such barrier is called the depletion region. This region is responsible for the low permeability between the two materials. When the holes try to go to the n-type materials, they are prevented by the positive ions of the depletion region, and when the electrons try to go to p-type, it is prevented be the negative ions of the depletion region. However, its applications in the logic gates come to real when connected to a battery.[5]

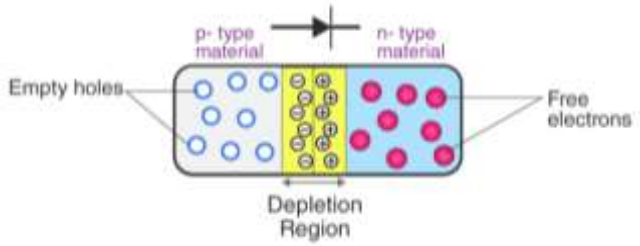

Figure 5 shows the p and n semiconductors connected together forming what is called a diode

While connected to a battery, if the positive side is connected to the p-type material and the negative side is connected to the n-type, the repulsion force causes the electrons and holes to overcome the force of the depletion region and the electrons and holes will flow in opposite directions, causing the current to flow normally in the circuit, which is called a forward biasing. On the other hand, if the positive side of the battery is connected to the n-type material, and the negative side of the battery is connected to the p-type material, this will attract the electrons and holes away from the depletion region causing it to have bigger effect. In this case, the current will cut, and the diode will work as an insulator, which is called a reverse biasing.[6]

As simple as that, a transistor is composed of three adjacent materials. Each one is different from its neighbors, which results in two combinations. Either a P-N-P type, which consists of two p-type materials with one n-type material in between, or a N-P-N type, which consists of two n-type materials with a p-type material in between. The way the three parts are connected, how they conduct or insulate electric current, and how this is applied in creating logic gates are all explained in the next sections.[7]

IV. THE ARITHMETIC LOGIC UNITS

Arithmetic Unit:

The first part of the CPU is called the Arithmetic Logic Unit (ALU). This is the unit responsible for doing arithmetic operations in the CPU. It is where addition, subtraction, multiplication and division happen. Before creating the ALU, however, it is necessary to understand how binary addition works.

The binary system is used as it could be naturally represented with low voltage (< 0.8V) and high voltage (> 2.0V). As there are only two values in a binary system, each place is worth two times the place before it. That is shown in **Figure 6**. [13]

A	B	Carry	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Figure 6 shows binary addition

To add values in binary, all of the values that add up to that value have to be true. For example, for representing the number nine, the values set to true have to add up to number nine. A problem arises however, when two numbers are added together. This is because when two numbers are set to true and the output is true, there has to be a carrier part to carry the extra one. [10]

The first operation of the ALU is adding values together, which is done by the Adder. For creating the adder, however, a NAND gate is needed. A NAND gate is an AND gate that takes two inputs with a NOT gate. We can then use the NAND gate to create another gate called the Exclusive OR gate – XOR for short – as shown in **Figure 7**. It has the truth table of a normal OR gate, with the exception of outputting False when both inputs are True. [10], [11]

A	B	Out
0	0	1
0	1	1
1	0	1
1	1	0

Figure 7 shows an XOR gate

At this point, it is possible to start making the adder. Therefore, we will create two outputs and three inputs. The three inputs are A, B, and the Carry in, while the first output is the sum of the inputs, and the second

is the carry part. The reason we have a carry in is that at some points when adding we have a third part to be added as a result from the two added ones before such as shown in **Figure 8**.

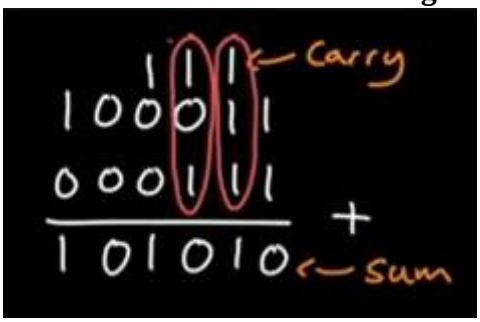

Figure 8 shows addition in binary and the carry bit

At this point, it is possible to add two numbers together using what was created in **Figure 9**. However, there has to be a third input to add the carry part after having the sum of the first two inputs. This is shown in **Figure 10**. Therefore, it will output one if only one input is true, it will output a zero if two numbers are true and outputs a one in the carry node. It will also output a one in the sum node. [12]

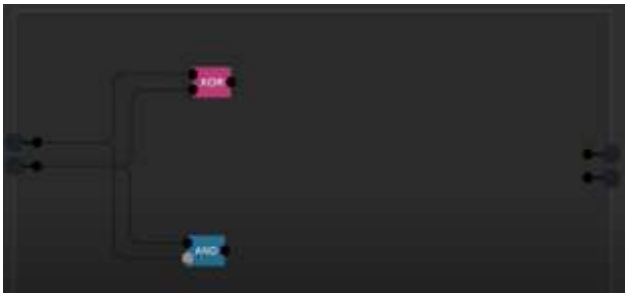

Figure 9 shows the adder with two inputs

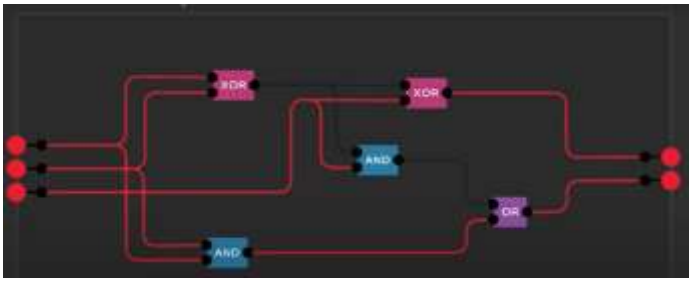

Figure 10 shows the adder with the carry in node

After creating the adder, various adders could be added together to create more-bit adders. For example, four adders connected together will be a 4-bit adder which can calculate numbers up to the sum of 16 as shown in **Figure 11.**

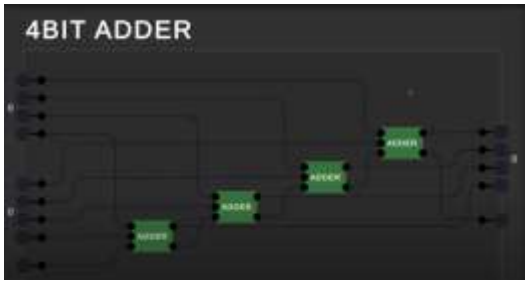

Figure 11 shows a 4-bit adder

In modern days, a slightly different approach is used called the Carry-look-ahead adder which has a few built-in operations that are used by the unit such as subtraction, addition, incrementing, decrementing…etc.

Processes such as multiplication or division are not going to be described as they are more complicated, however, in simple ALUs they can be done by repeating addition, for example. A problem that arises is representing negative numbers. This can be done in a lot of ways. For simplicity, one way will be discussed in this article which is called the two's compliment system. The way this system works is y replacing the last index of a x-bit adder with a $-n$, where n is the maximum number that could be represented by the x-bit adder. That negative number is added to other positive numbers to create the negative numbers. [13]

Logic Unit:

We use the logic unit to determine a few operations and check for a few things. The first circuit in the logic unit checks if the sum is a zero. This could be useful for checking if two numbers are equal, by subtracting them and checking if they are equal to zero. The way this circuit looks is as shown in **Figure 12**. The input(s) goes through one OR gate or a series of them until reaching the NOT gate, where if at least one of the inputs is one, the output is zero/False, which means that there are no zeroes as the output of the process. However, if all of the inputs are zero, the output is then turned to True, which means that the process gives a zero output.

Another problem with the adder, however. The fact that the adder is only 4-bit. Therefore, having numbers more than 15 will cause an overflow and give a value of zero. This problem could be solved simply by extending the number of adders to 8-bit or 16-bit adders, which makes it less likely for an overflow to happen. This, however, has a downside as the adder takes more time. [12], [13]

Figure 12 shows a zero checker which is made of an OR gate and a NOT gate

V. TRANSISTORS IN LOGIC GATES

After explaining the principle of the logic gates, the use of transistor to create them is quite simple. As mentioned before, the transistor is composed of three parts, which are called the emitter, base, and the collector. In N-P-N transistor, the current will flow from the emitter through the base and to the collector. Therefore, the creation of logic gates using an N-P-N type transistor is very simple. **Figure 13** shows how and gate is created using transistors. In such arrangement, only if the two transistors have an input the output will exist. If, for any reason, any of the inputs or the two of them is gone, this will result in an open circuit and there will be no output.[2]

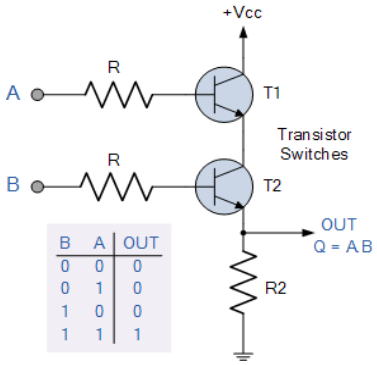

Figure 13 shown an AND logic gate created with transistors

The OR gate arrangement is shown in **figure 14**, in this arrangement, as shown in the figure, if any of the two input or both of them are on then, the output will be on. For the not gate, its circuit is shown in **figure 15**. As said before, such gate takes only one input and gives the opposite output. Using these three simple gates, more complicated one are made, which are essential parts of the current computers. [8]

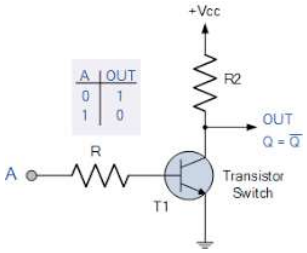

Figure 15 shows a NOT gate created using Transistors

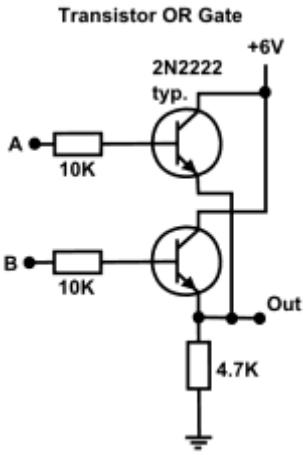

Figure 14 shows an OR gate created using transistors

VI. CONCLUSION

In this article, the reason for the rabid development of computers was discussed. Boolean algebra which is a language of 1 and 0 that is the base of creating computers was explained. Followed by the transistors, the fire of 20th century, that was the main reason for the rabid development of these brilliant devises. Finally, the way that these components are used to make mathematical operations was explained, showing principle behind the main part of any computer.

REFERENCES

[1] "Transistors 101." http://apps.usd.edu/coglab/psyc770/transistors101.html (accessed Oct. 14, 2022).
[2] "Transistor Gates." http://hyperphysics.phy-astr.gsu.edu/hbase/Electronic/trangate.html (accessed Oct. 14, 2022).
[3] P. Zheng, "Material properties of n-type silicon and n-type UMG solar cells," p. 196.
[4] "(PDF) Electrical Optical and Structural Properties of p-type Silicon." https://www.researchgate.net/publication/276318630_Electrical_Optical_and_Structural_Properties_of_p-type_Silicon (accessed Oct. 14, 2022).
[5] "DiodeTransistorNotes.pdf." Accessed: Oct. 14, 2022. [Online]. Available: https://inst.eecs.berkeley.edu/~ee100/su07/handouts/DiodeTransistorNotes.pdf
[6] "Tutorial on BJT - Part (2).pdf." Accessed: Oct. 14, 2022. [Online]. Available: https://feng.stafpu.bu.edu.eg/Electrical%20Engineering/833/crs-14354/Files/Tutorial%20on%20BJT%20-%20Part%20(2).pdf
[7] N. Fernando, "Types of transistor", Accessed: Oct. 14, 2022. [Online]. Available: https://www.academia.edu/29533520/Types_of_transistor
[8] "From transistors to gates!" Available: https://www.cs.bu.edu/~best/courses/modules/Transistors2Gates

[9] S. Sarma, R. Bhuyan, "Boolean Algebra and Logic Gates", *International Journal of Mathematics Trends and Technology*. 65. p. 147-151. Available: https://www.researchgate.net/publication/334054812_Boolean_Algebra_and_Logic_Gates

[10] C. Maxfield, W. Pete, "Boolean Algebra". Available: https://www.researchgate.net/publication/345041400_Chapter_9_Boolean_Algebra

[11] R. Lianly, "Boolean Algebra for Xor-Gates". Journal of Computer Sciences and Applications. Available: https://www.researchgate.net/publication/274526470_Boolean_Algebra_for_Xor-Gates

[12] S. John, "Digital Logic for Computing" p. 11-24 Available: https://www.researchgate.net/publication/318132575_Boolean_Algebra

[13] N. Shanthala, C. Chandrashekar. Y. Siva, "Basic operations performed on arithmetic logic unit for 32-bit floating point numbers" International Journal of Applied Engineering Research. Available: https://www.researchgate.net/publication/318771835_Basic_operation_performed_on_arithmetic_logic_unit_ALU_for_32-bit_floating_point_numbers_Initial_results