CODING OF INFORMATION IN THE BINARY NUMBER SYSTEM

Zaripov Nozimbek Nayimovich
Bukhara State Pedagogical Institute Doctoral student (DSc), Doctor of Philosophy in Pedagogical Sciences (PhD), Associate professor

Abstract: *This article describes methods of encoding information using binary. In addition, this article provides examples of reverse and forward encoding methods. Also, the article explains in detail the concepts of addition and subtraction according to the sign of numbers.*

Key words: *uneven encoding, number system, complementary code, code length.*

One of the important tasks is to improve the methodology of developing practical and professional training of students in the teaching of informatics and digital technologies, to teach the basics of logical computer operation. Encoding, decoding information, performing actions on them, as well as explaining the significance and essence of the Fano method to students, explaining the topic through practical and real-life examples will prepare the ground for the development of mature personnel for the field of digital technologies in the future.

When a computer performs arithmetic operations, the results will vary with respect to the number entered. For example, we all know that -5+(-3) =-5-3=-8. But it is important to determine how this -8 result is produced and to convey it to future teachers of informatics and digital technologies through modern methods. Therefore, we will consider the sequence of execution of this arithmetic expression.

(-5) direct code reverse code complementary code

1	0	0	0	0	1	0	1

1	1	1	1	1	0	1	0

1	1	1	1	1	0	1	1

We use the inversion (negation) operation when moving from the correct code to the reverse code. When converting to a complementary code, the number 1 is always added to the reverse code. Then, as above, we generate the correct, inverted and complementary codes for the number -3.

(-3) direct code reverse code complementary code

1	0	0	0	0	0	1	1

1	1	1	1	1	1	0	0

1	1	1	1	1	1	0	1

To reverse code -5 and -3 above, the binary numbers are logically added, then 1 is added to the resulting result, and the correct answer is obtained by negating the result. To create a complement code, the binary representation of two numbers is first added, then the result is negated and the number 1 is added.

direct code reverse code complementary code

-5	1	0	0	0	0	1	0	1
-3	1	0	0	0	0	0	1	1

	1	1	1	1	1	0	1	0
	1	1	1	1	1	1	0	0
1	1	1	1	1	0	1	1	0
+								1
	1	1	1	1	0	1	1	1
	1	0	0	0	1	0	0	0

	1	1	1	1	1	0	1	1
	1	1	1	1	1	1	0	1
1	1	1	1	1	1	0	0	0
	1	0	0	0	0	1	1	1
+								1
	1	0	0	0	1	0	0	0

It can be seen that when the values in the last line of the table above are converted to the decimal number system, 8 numbers are formed.

When generating forward, reverse, and complement codes for negative numbers, the number 1 is always written in the first space of binary numbers. In positive numbers, the number 0 is written [4; p. 54]. Let's

see how the sum is calculated when the numbers are positive and negative. For example: we choose the expression 5+(-3)=2. First of all, we describe the numbers 5 and 3 in the binary number system.

$$5(10) = 1 * 2^2 + 1 * 2^0 = 101(2)$$

$$3(10) = 1 * 2^1 + 1 * 2^0 = 11(2)$$

Using the above result, we generate forward, reverse and complement codes. 1 is written in the first digit of negative -3. In the reverse code, the negation operation is performed, in the complement code, the number 1 is added. The result is shown in the table below.

direct code

0	0	0	0	0	1	0	1

reverse code

0	0	0	0	0	1	0	1

complementary code

0	0	0	0	0	1	0	1

direct code

1	0	0	0	0	0	1	1

reverse code

1	1	1	1	1	1	0	0

complementary code

1	1	1	1	1	1	0	1

direct code

0	0	0	0	0	1	0	1
1	0	0	0	0	0	1	1

reverse code

0	0	0	0	0	1	0	1
1	1	1	1	1	1	0	0
1	0	0	0	0	0	0	1
+							1
1	0	0	0	0	0	1	0

complementary code

0	0	0	0	0	1	0	1
1	1	1	1	1	1	0	1
1	0	0	0	0	0	1	0

You can check the result by converting the number 0000010 from the binary system to the decimal system.

$$0000010\ (2) = 1 * 2^1 + 0 * 2^0 = 2$$

One of the main devices of the computer is the processor. The main task of the processor is to read commands from the program located in the RAM device, perform actions, write down the results and read the next command. The processor reads and processes data from the RAM in chunks. The number of bits in a chunk that it can get for processing at a time is called the processor's throughput. While the first computers had 4-bit processors, most computers have 16-bit, 32-bit processors, and now 64-bit, 128-bit processors. We usually come across entries like "i74809, 16/32" on most CPUs. We often do not pay attention to what this writing means. The "i" in the notation given in this manual stands for the name of this intel compass. Currently, two major companies are producing processors. These are: Intel and AMD. 80386 is the processor number, that is, the name of the version. 16/32 is considered one of the most basic concepts for the user. The entry 16 means that this processor has a 16-bit data bus, and the entry 32 means a 32-bit address bus. In other words, 16 records can process 16 bits of information at the same time. 32 indicates the ability to work with 2^32 address space.

All information in computer memory is stored in binary number system. Computer programs written in the hexadecimal number system are converted to the binary number system using special compiler programs and sent to the central processor for execution. Special coding methods are used to transfer numbers from one number system to another number system [5, p. 13].

There are floating-point and fixed-point representations of real numbers on a computer. Whole numbers are represented in the computer memory by the method of fixed-point numbers [5, p. 19]. Let's consider the following example of how numbers given in the binary number system are expressed in 32-digit form. For example: 18,125 is given in the decimal number system, let's see how this number is expressed in 32-digit form on a computer. In this case, we divide the number 18,125(10) into two parts and switch to the binary number system. In this case, the whole part of the number is transferred to the binary number system by dividing it by 2 or spreading it, and by multiplying the fractional part by 2.

$$18 = 1 * 2^4 + 1 * 2^1 = 10010(2)$$

The integer part of the given number takes the form 10010(2) in binary notation.

0,	125
	* 2
0	250
	* 2
0	500
	* 2
1	000

As a result, the number given in the decimal number system 18,125 will look like this in the binary number system.

$$10010{,}001(2)$$

To describe the resulting result in 32-digit form, we do the following. First of all, we move the comma in the number 10010,001(2) until the whole part is 0.

That is, it takes the form $0.10010001*10^5$. We also express the number 5 on the level by binary (Fig. 1). $0.10010001*10^{101}$

Figure 1. Number representation table in 32-digit form.

It can be seen from Figure 1 above that the result of the given problem will be as follows.

0	0	0	0	0	0	1	0	1	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The number, text, graphics, audio, video information displayed in the computer memory is stored in the form of a file. Files occupy a certain amount of space from the computer memory. Computer memory consists of cells intended for storing information [5, p. 21]. Each state can be either a 0 or a 1 and occupies one cell in the computer's memory and is called a bit.

The most involved characters are coded with the fewest bits, and the less involved characters are coded with the most bits. The most basic condition of coding is that no code word can be the beginning of another code word. The last number in the branch of the tree is not used to encode other characters.

In a computer, 8 digits are equal to 1 byte, and this represents one character code. When using right, reverse, and complement codes, the binary representation of the number is written with 0 for positive numbers and 1 for negative numbers. Inversion and complement codes are also used. Arithmetic operations on positive and negative numbers can be performed through the sample examples and problems given above. Through this, it serves to increase the effectiveness of the lesson in teaching students the subject of correct, inverse, and complementary codes and to develop their knowledge, skills, and abilities on the subject of coding.

REFERENCES

1. Monica Borda «Fundamentals in Information Theory and Coding» Springer-Verlag Berlin Heidelberg, 2011. – 504 p.
2. Mark Kelbert, Yuri Suhov «Information Theory and Coding by Example» Cambridge University Press, 2014. – 528 p.
3. Sebesta, Robert W. «Concepts of programming languages» Includes bibliographical references and index. Pearson, 2012. – 816 p.
4. Zaripov N. N. Using methods of foreign experiences in teaching informatics and information technologies in school. Theoretical & Applied Science, (3), 2020. – 111-114 p.
5. Zaripov N. N. Methods of Teaching Programming Languages in General Secondary Schools. In “Online-conferences” platform, 2021. – 38-40 p.