НАО «Алматинский университет энергетики и связи имени Гумарбека Даукеева»

УДК 621.396.6:004.056.5+623.36

**ВИТУЛЁВА ЕЛИЗАВЕТА СЕРГЕЕВНА**

# Постиндустриальная парадигма развития инфокоммуникационного сегмента оборонно-промышленного комплекса РК

6D071900 – Радиотехника, электроника и телекоммуникации

Диссертация в форме серии статей
на соискание степени доктора философии (PhD)

Научный консультант:
кандидат технических наук,
профессор Коньшин С.В.

Зарубежный научный консультант:
доктор философских наук,
профессор Габриелян О.А.

Республика Казахстан
Алматы, 2024

scientific reports

Check for updates

OPEN

# Improving the efficiency of using multivalued logic tools

Ibragim E. Suleimenov[1], Yelizaveta S. Vitulyova[2], Sherniyaz B. Kabdushev[1,3] & Akhat S. Bakirov[2]✉

**Multivalued logics are becoming one of the most important tools of information technology. They are in great demand for creation of artificial intelligence systems that are close to human intelligence, since the functioning of the latter cannot be reduced to the operations of binary logic. At the same time, the problem of improving the efficiency of using the results of research in multivalued logics, as well as the problem of interpreting variables of multivalued logic, is acute. These problems create certain interdisciplinary barriers and make it difficult to implement the results of research in the field of multivalued logics in other fields of knowledge. It is shown that the problem of interpreting multivalued logic variables can be removed by establishing correspondence with fuzzy logic variables. Improving the efficiency of using of operations of multivalued logics and their variables can be provided by using their close connection to Galois fields. This connection, among other things, makes it possible to reduce any operations of multivalued logics, the number of variables in which is equal to a prime number, to algebraic functions whose arguments take values in Galois fields. This allows, among other things, to eliminate the very cumbersome constructions used in works on multivalued logic and make its apparatus convenient for use in related scientific disciplines in information technology. Direct verification of the adequacy of algorithms based on the use of Galois fields can be carried out by means of radio-electronic circuits, examples of which are presented in the present paper.**

The emergence of non-Aristotelian logics (in particular, Lukasevich's logic[1] N. Vasiliev's "imaginary logic"[2]) at the beginning of the twentieth century was obviously connected with the transformation of the general situation in the philosophy of mathematics and the discussions concerning the problems of justification of mathematics and logic as such[3]. As noted[2], N. Vasiliev proposed a project of non-Aristotelian logic built without using the law of contradiction, proceeding from the analogy with the non-Euclidean geometry of N. Lobachevsky, which excludes the use of the fifth postulate of Euclid, who also initially called his geometry "imaginary". The construction of logics that partially or completely refuse to use the law of the excluded third ("every statement is either true or false", to use the simplest version of the interpretation) has subsequently led to a great variety of multivalued logics[4,5], including paraconsistent logics[6], paracomplete logics[7], etc.

De facto, there are currently a huge number of varieties of multivalued logics, but the question of how exactly they are applicable to the description of the laws of thought remains open[8].

In this respect, it is worth pointing out that, in accordance with the tradition going back to Aristotle, logic was viewed as a science of how to correctly reason, as a science of the laws of thinking. This is the way J.W. Bull interpreted it. In a famous monograph on the history of mathematics[3] the following passage from one of J. Boole's main works, "Investigation of the laws of thought", is given to illustrate exactly this approach, which prevailed then in the field of logic creation:

> "In the treatise before us we intend to investigate the fundamental laws of those operations of the mind by which thinking is effected, in order to express them in the symbolic language of calculus, and on this basis to construct the science of logic and its method."

Obviously, most modern works on multivalued logics have departed far enough from this tradition, otherwise the problem of interpretation of multivalued logics and their classification would not be so acute.

The problem of applicability of multivalued logics to the reflection of laws of thought is most closely related to the problem of interpreting the variables of multivalued logic, which remains relevant at present[9,10]. Whereas

[1]National Engineering Academy of the Republic of Kazakhstan, Bogenbai Batyr Str. 80, 050010 Almaty, Republic of Kazakhstan. [2]Gumarbek Daukeyev Almaty University of Power Engineering and Telecommunications, Baytursynov Str. 126/1, 050013 Almaty, Republic of Kazakhstan. [3]International Information Technology University, Manas Str, 34/1, 050013 Almaty, Republic of Kazakhstan. ✉email: axatmr@mail.ru

within binary logic, its variables can be uniquely associated with the notion of truth, such uniqueness is lost for multivalued logics, which determines the relevance of research in the philosophy of multivalued logics, which is currently being actively pursued[11,12].

However, we must admit that a full-fledged return to the tradition that considers logic as a reflection of the laws of thinking, obviously, cannot be realized otherwise than on an interdisciplinary basis. This, in turn, requires overcoming pronounced interdisciplinary barriers. The language in which the works on multivalued logics are written remains difficult to comprehend for a large part of specialists in other fields of knowledge, in particular in information technologies.

A definite step towards overcoming the interdisciplinary barriers is knowingly solving the problem of visibility of variables of multivalued logics touched upon in[13,14].

To solve this problem, it is reasonable to use the correspondence between multivalued logics and algebraic structures, such as Galois fields, which are widely used in modern information technologies, especially in cryptography[15–17]. This correspondence can be most easily established when the number of variables of a particular multivalued logic is equal to the degree of the prime number p. In this case, a Galois field element $GF(p^n)$ can be assigned to each value of a variable of multivalued logic in a one-to-one correspondence.

For Galois fields, in turn, the following illustrative interpretation can be proposed. As emphasized[18,19], the standard model of a signal is a function taking values on a set of real numbers. However, in the case when the signal is reduced to a certain set of discrete levels that fit into a finite range of amplitude measurements, this approach is not mandatory. A function taking values in any finite algebraic structure, such as Galois fields, can also be used as a signal model. The simplest kind of Galois fields $GF(p)$ is formed through a homomorphism of a ring of integers to a ring of classes of deductions modulo $p$, where $p$ is a prime number.

In this paper, we show that the problem of interpreting the variables of multivalued logic can be solved, for example, by establishing a correspondence between the variables of multivalued logic and the variables of fuzzy logic. Variables of multivalued logic can also be assigned to the levels of the digitized signal in the case when the signal model is a function that takes values in Galois fields. More broadly, the variables of multivalued logic can be interpreted through the establishment of links between concepts (e.g., philosophical categories). In all these cases, it is important to have a tool that allows you to bring logical relationships to an algebraic form. For the case when the set of variables of multivalued logic can be assigned to the field $GF(p^n)$, this problem is solved through an analogue of the algebraic normal form presented in this paper.

Section 1 shows that the use of multivalued logic variables can be made explicit, including by mapping to multivalued logic variables.

Section "Visualization of the variables of multivalued logic" shows that for the case when the number of variables is equal to a prime number, instead of the truth tables traditionally used in works on multivalued logic, it is also possible to use an analog of the algebraic normal form (the Zhegalkin polynomial).

Section "Reduction of multivalued logic operations to algebraic ones" provides a specific example showing that multivalued logic operations can be performed using electronic devices built on typical binary logic components.

## Visualization of the variables of multivalued logic

Clear illustrations for the practical use of variables of multivalued logics are easiest to offer, focusing on the approaches used in fuzzy logic. As is known, fuzzy logic establishes a certain correspondence between ranges of continuously varying parameters and linguistic variables marking them[20]. Simplifying, the apparatus of linguistic variables allows to "transform into words" the values of parameters, which, under certain conditions, can be quantitatively measured with high accuracy.

It is interesting to note that linguistic variables were introduced in practice long before fuzzy logic was created. For example, in maritime, there has traditionally been a set of commands "full astern, … slow astern, …, slow ahead, …, full ahead." A similar conclusion is also valid in relation to the compass rose (Fig. 1), which is also traditionally used in maritime affairs.

Figure 1 emphasizes that the 8-element compass rose can be used to visually interpret the variables of 9-digit logic.

The variables of such a logic can be put in correspondence with elements of the Galois field $GF(3^2)$, which, in turn, can be constructed as an algebraic extension of the field $GF(3)$.

Recall that the method of algebraic extensions can be viewed as a generalization of the method by which complex numbers are constructed. Let us demonstrate the fact on a simple example of the construction of the field $GF(3^2)$.

The field $GF(3)$ contains three elements. They can be chosen as $(-1, 0, 1)$ by setting the following addition rules.

$$1 + 1 = -1; -1 - 1 = 1 \tag{1}$$

According to the method of algebraic extensions, an additional element $\theta$, which is the root of an equation irreducible (having no solutions) in this field, is attached to this (or any other) field.

$$f(x) = 0 \tag{2}$$

where $f(x)$ is a polynomial of degree n, $x$ is a variable that takes values in $GF(3)$.

In the special case where $n = 2$ such irreducible equation is the equation that allows one to construct complex numbers

$$x^2 + 1 = 0 \tag{3}$$

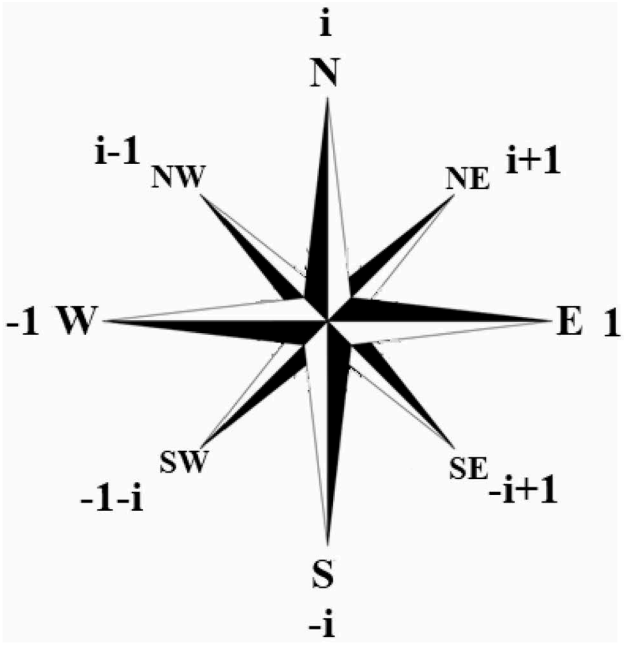


**Figure 1.** 8-wind compass rose (Image was generated in Paint 22H2 https://apps.microsoft.com/store/detail/paint/9PCFS5B6T72H?hl=ru-ru&gl=ru).

Then the element $\theta$ can be treated as a logical imaginary unit, and the elements of the field $GF(3^2)$ can be represented as

$$A = a_0 + ia_1, \tag{4}$$

where variables $a_0, a_1$ belong to the main field.

In this case, we can perform algebraic operations with elements of the form (4) according to formulas (1) and (3). For example,

$$i + i = -i; -i - i = i; -i + i = i - i = 0, \tag{5}$$

The rules of multiplication remain the same as in the classical use of complex numbers, in particular,

$$i^2 = -1; i \cdot (a_1 + ia_2) = ia_1 - a_2, \tag{6}$$

The elements of this field are listed in Table 1.

In general, any element of the field $GF(3^n)$ can be represented as a linear combination of powers of $\theta$.

$$A = \sum_{0}^{n-1} \theta^j a_j \tag{7}$$

where $\theta$ is a primitive element, $a_j$ are coefficients from the main field of $GF(3)$, and $n$ is the degree of the polynomial $f(x)$ generating the element $\theta$.

The field $GF(3^2)$ contains eight non-zero elements (Table 1). Using the notation (4) as a logical coordinate representation, these eight elements can be assigned to the directions of compass roses, which is shown in Fig. 1.

In this example there is a one-to-one correspondence between the elements of multivalued logic, linguistic variables, and elements of the Galois field. More precisely, the elements of the compass rose allow all the above interpretations, which are in a mutually unambiguous correspondence.

Thus, the problem of interpreting multivalued logic variables can be removed if these variables are interpreted through correspondence to fuzzy logic variables. Such an approach, as shown in Section "Reduction of

| $a$ | $a_2 = -1$ | $a_2 = 0$ | $a_2 = 1$ |
|---|---|---|---|
| $a_1 = -1$ | $-1 - i$ | $-1$ | $-1 + i$ |
| $a_1 = 0$ | $-i$ | 0 | $i$ |
| $a_1 = 1$ | $1 - i$ | 1 | $1 + i$ |

**Table 1.** Elements of the Galois field $GF(3^2)$ in the used representation.

multivalued logic operations to algebraic ones", is generalizable. Namely, in this interpretation, rather a wide range of different terms (including philosophical categories) can be used instead of fuzzy logic variables. Obviously, it is not the specific set of sounds or symbols that represent them that gives meaning to natural language words, but the fact that each of these words is built into the overall structure of the language. Therefore, the meaning of terms is actually determined by the connections between them. The "True—False" opposition, which forms the methodological basis of binary logic, is only the simplest form of such a connection.

Let us show that for the case when the number of variables of multivalued logic is equal to a prime number, any operations in such logic can be reduced to operations of addition and multiplication in the Galois field.

## Reduction of multivalued logic operations to algebraic ones

The operations of multivalued logic are usually displayed in the form of truth tables. So, the following Table 2 are reflecting the operations of the logic of paradoxes by G. Priest[21].

In these tables, symbols "0", "1" and "2" are denoting logical variables. The interpretation of the variables of ternary logic as "Truth", "False", "Uncertainly" dates back to the works of Lukasiewicz. The interpretation of such operations (disjunction, conjunction, negation, etc.) as applied to ternary logic can be different, likewise, the use of specific symbols in such tables is nothing more than a matter of agreement.

Such a tabular representation is not always convenient. Operations on logical variables, to which elements of the Galois field are assigned, can be reduced to algebraic ones. For clarity, this can be done, for example, as follows.

To avoid cluttering the notes, we will consider the case of an arbitrary function $f(x,y)$, taking values in the field $GF(p)$, where $x, y$ are elements of the same Galois field. This function corresponds to a truth table given by an ordered enumeration of elements $f(x_i,y_j)$, $i,j = 0,2\ldots p-1$.

Consider the following expression

$$g_i(x) = 1 - (x - x_i)^{p-1} \tag{8}$$

where $x_i$ is a fixed element of the field $GF(p)$.

It is known from Galois field theory that all nonzero elements of the field $GF(p)$ are roots of the equation

$$\theta^{p-1} - 1 = 0 \tag{9}$$

That is, any nonzero element of the field $GF(p)$, if raised to the $p-1$ st power, gives one.

Consequently, the functions $g_i(x)$ have the following property

$$g_i(x) = \begin{cases} 1, x = x_i \\ 0, x \neq x_i \end{cases} \tag{10}$$

This allows us to treat them as a logical analogue of the δ-function.

Let us form the following polynomial

$$F(x,y) = \sum_{i,j=0}^{i,j=p-1} f(x_i,y_j) g_i(x) g_j(y) \tag{11}$$

where the values $f(x_i,y_j)$ form a truth table like the Table 2.

When a particular pair of $x_{i_0}, y_{j_0}$ values of logical variables (or more exactly, their corresponding Galois field elements) is substituted into expression (11), all summands appearing in the sum in the right part of formula (11) turn to zero because of relation (8) except the summand for which $i = i_0, j = j_0$ is satisfied. Hence, it follows that

$$F(x_{i_0}, y_{j_0}) = f(x_{i_0}, y_{j_0}) \tag{12}$$

We see that the polynomial (11) performs the same functions for multivalued logic as the Zhegalkin polynomial for binary logic, i.e., relation (11) indicates a specific algebraic function which realizes a given truth table. It is also seen that relation (11) admits a generalization to the case of an arbitrary number of logical variables.

| $F(x,y) = \vee$ | $y=0$ | $y=1$ | $y=2$ |
|---|---|---|---|
| $x=0$ | 0 | 1 | 2 |
| $x=1$ | 1 | 1 | 2 |
| $x=2$ | 2 | 2 | 2 |
| $F(x,y) = \wedge$ | $y=0$ | $y=1$ | $y=2$ |
| $x=0$ | 0 | 0 | 0 |
| $x=1$ | 0 | 1 | 1 |
| $x=2$ | 0 | 1 | 2 |

**Table 2.** Values of the logical function corresponding to the operations of disjunction and conjunction in the logic of paradoxes by G. Priest.

Note that control methods based on fuzzy logic are currently being actively developed[22,23]. There are known works, in which such methods are proposed to be used for correcting the course of ships[24].

Obviously, if a one-to-one correspondence is established between linguistic variables and Galois field elements, then all "commands" and "data" transformed to such variables can be further processed using algebraic functions, which can be constructed knowingly by the method described above.

Of course, for real problems, the number of variables corresponding to an 8-element compass rose is insufficient, but this is not an obstacle.

For example, starting from the field $GF(7)$, the elements of which can be chosen as $(-3, -2, -1, 0, 1, 2, 3)$, we can construct the field $GF(7^2)$.

The elements of this field are also representable in the "two-coordinate" form (4), where the coefficients $a_0, a_1$ belong to the field $GF(7)$.

The entry (4) in this case, for clarity, can be interpreted, for example, as a discrete representation of the velocity vector (in the plane), which fully corresponds to the traditional complex representation of vectors. The difference is that using the field $GF(7^2)$, the velocity components are discrete, and they can be assigned to seven linguistic variables "full astern, half astern, small astern, stop engine, small ahead, half ahead, full ahead".

The use of such a field also allows us to map the linguistic variables corresponding to the 16-item compass rose, Fig. 2.

Namely, the number of non-zero elements of the field $GF(7^2)$ is 48. Consequently, they are all roots of the equation

$$x^{48} - 1 = \left(x^{16}\right)^3 - 1 = \left(x^{16} - 1\right)\left(x^{32} + x^{16} + 1\right) = 0 \tag{13}$$

Formula (13), among other things, shows that among the elements of the field $GF(7^2)$ there are 16 elements that satisfy the equation

$$x^{16} - 1 = 0 \tag{14}$$

These 16 elements can be viewed as roots of the 16th degree from one, and they form a group by multiplication. Consequently, they can be assigned linguistic variables corresponding to the 16-element compass rose.

Thus, the mutually unique correspondence between multivalued $p^n$-logics, where $p$ is prime number, $n$ is integer and Galois fields $GF(p^n)$ creates all preconditions for making operations on variables of multivalued logic as clear as possible.

It can be argued that visualization in this respect is provided not so much for variables of multivalued logic as for elements of Galois fields. However, the visual representation of operations on the variables of multivalued logic mapped through Galois fields has also a philosophical aspect directly related to the problem of interpretation of the values of the mentioned variables and to the problem of correlation of laws of thinking and multivalued logics touched upon in[8].

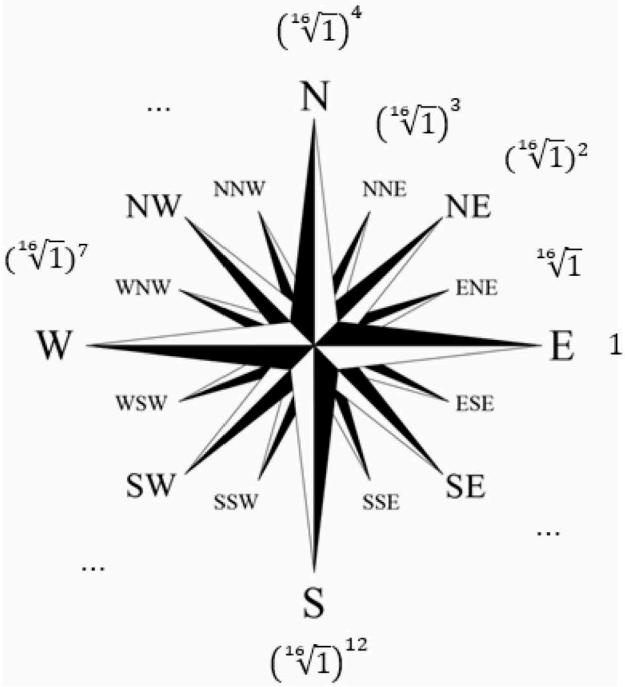


**Figure 2.** 16-element compass rose (Image was generated in Paint 22H2 https://apps.microsoft.com/store/detail/paint/9PCFS5B6T72H?hl=ru-ru&gl=ru).

Namely, the meaning of the variables of binary logic relates to the philosophical category of truth. This category belongs to the number of basic concepts, the question about the nature of which is closely related to the problem of the existence of undefined concepts. Indeed, to "define" means to reveal the meaning of one term through others. Trying to reveal all the terms available in a language in this way leads knowingly to a vicious circle.

Objective dialectics finds a way out by defining the basic categories through the oppositions "quantity–quality," "content–form," etc. Such an approach, in particular, was used[13,25] in order to reveal as correctly as possible, the meaning of the category "information", which it was suggested to consider as a philosophical category paired with the category of matter.

The problem of adequate interpretation of the concept "information" as emphasized in[26,27] becomes more and more relevant in connection with the research in the field of artificial intelligence, but for the purposes of our article the approach of "definition through contraposition" itself is more important.

Namely, it shows that for the definition of basic notions the most important is the structure of relations between them, and contraposition is only one of the forms of such relations, and the one that knowingly corresponds to binary logic and Galois binary fields. Obviously, other forms of connections between basic concepts cannot be reduced to a simple contraposition.

This indicates for example the existence of a pronounced methodological (philosophical) aspect of the development of command languages (even at the level of specific technical systems), which constitute a closed whole at the expense of relations written in algebraic form. Moreover, it is extremely difficult to develop closed "language" systems at the level of abstraction. It is much more convenient (and illustrative) to do this by solving specific problems, for example, those related to control of moving vehicles, in terms of fuzzy logic converted into algebraic form.

This formulation of the question makes it even more urgent to ensure the visibility and usability of multivalued logics. The following section deals with specific computational tools oriented to the use of logics corresponding to the fields $GF(7^n)$.

This example allows you to clearly demonstrate that it is possible to implement various kinds of devices that perform calculations in terms of multivalued logic, but at the same time built on the basis of typical electronic components using binary logic.

## Computational implementation of seven-digit logic operations

Currently, algorithms and schemes of radioelectronic devices that perform calculations modulo are widely represented in the literature. Thus, such algorithms are used in encryption, coding devices, in compression and transmission of information, in automation devices[28–30].

As shown above, any functions whose arguments are variables taking values in the Galois field can be explicitly reduced to algebraic expressions which involve only multiplication and addition operations modulo $p$.

Consequently, multipliers and adders modulo $p$ are the basis for automating any operations on logical (linguistic) variables. Devices of this type can be implemented by rather simple means, as it is proved below.

The block diagram of the multiplier of the considered type is presented in Fig. 3. The scheme includes adders (marking on the scheme is $\Sigma$), which count the number of units on the inputs $a_i$ corresponding to the number representation in binary form. It is supposed, that on the input of the system no signals corresponding to number 7 or number 0 are input. This is acceptable, since when calculating modulo 7, $7 \equiv 0(7)$ takes place, therefore, in this case, the calculated product is equal to zero. In this case $\Sigma a_i$ can take values 1 or 2, as in the binary notation of numbers that vary from 1 to 6, there are at least one and at most two units.

Then

$$6 \cdot a_3 a_2 a_1 =_{(7)} \overline{a}_3 \overline{a}_2 \overline{a}_1 \tag{15}$$

where $a_i$ are characters in the binary notation of the number, a bar over the character means the inversion operation, i.e., 0 changes to 1 and vice versa.

Due to the associativity of multiplication modulo, the product of any two non-zero elements of the field $GF(7)$ can be reduced to the multiplication of two numbers in binary representation, and in both of these numbers only one of the symbols $a_i$ will be non-zero.

Correspondent operation is realized by the inverter block (the standard inverter designation is used in the scheme) controlled by the signal taken from $\Sigma$ elements. If logical zero is formed on the output of these elements, signals $a_i$ and $b_i$ remain unchanged, if logical one, they take inverse values.

The signal sets $\widetilde{a}_i$ and $\widetilde{b}_i$, reduced to a format in which only one of the variables of these sets is non-zero, are fed to the direct multiplier block (schematic designation—$\otimes$).

The signal set $\widetilde{c}_i$ from the output of the direct multiplier is fed to the output inverter block, which operates in the same way as the input inverter block.

The schematic diagram of the direct multiplication block is shown in Fig. 4.

This block works as follows.

The prime number 7 is a special case of prime Mersenne numbers, represented in the form $p_m = 2^n - 1$. Such numbers have the following property. Multiplication of any number by 2 modulo $p_m$ results is a cyclic permutation of symbols. For example,

$$2 \cdot a_2 a_1 a_0 =_{(7)} a_1 a_0 a_2 \tag{16}$$

where $a_i$ are binary characters.

Let us consider the product of two numbers $B \cdot A$ written in binary notation. We have

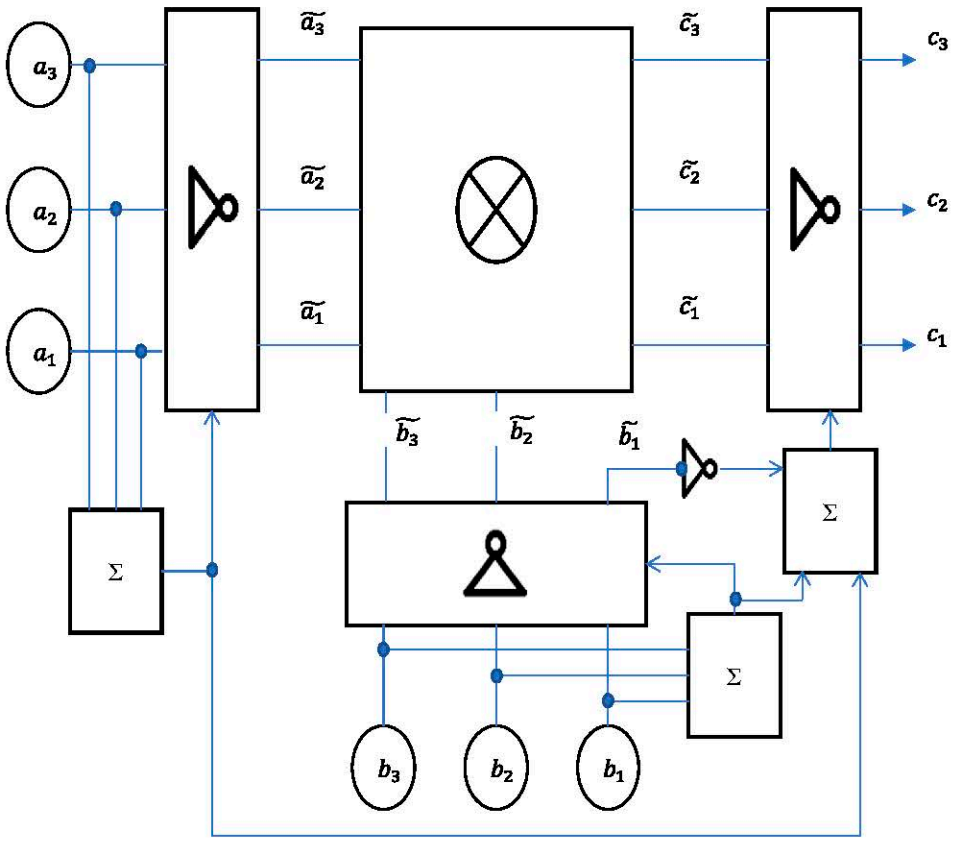


**Figure 3.** Block diagram of the modulo multiplier by seven (Image was generated in PowerPoint Microsoft 365 https://www.microsoft.com/en-ww/microsoft-365/powerpoint).

$$B \cdot A =_{(2)} b_3 \cdot 2^2 \cdot A + b_2 \cdot 2^1 \cdot A \cdot + b_1 \cdot 2^0 \cdot A \tag{17}$$

According to formula (16), products $2^m \cdot A$ may be written through cyclic permutations, i.e. the product calculated modulo 7, is the sum of the following three numbers written in binary form as

$$b_1 \cdot a_3 a_2 a_1 \tag{18}$$

$$b_2 \cdot a_2 a_1 a_3 \tag{19}$$

$$b_3 \cdot a_1 a_3 a_2 \tag{20}$$

where only one of the $b_i$ values is 1, and the rest are 0.

Each of the binary three-digit numbers appearing in formulas (18)–(20) can also be written in powers of two.

Consequently, the result of multiplication in calculations modulo 7 can be written as

$$B \cdot A =_{(2)} c_3 \cdot 2^2 + c_2 \cdot 2^1 \cdot + c_1 \cdot 2^0, \tag{21}$$

where

$$c_3 = b_1 a_3 + b_2 a_2 + b_3 a_1 \tag{22}$$

$$c_2 = b_1 a_2 + b_2 a_1 + b_3 a_3 \tag{23}$$

$$c_1 = b_1 a_1 + b_2 a_3 + b_3 a_2 \tag{24}$$

Since inverters are used in the circuit under consideration, in formula (22)–(24) only one of the values $a_i$ and only one of the values $b_i$ is equal to 1, the rest are equal to 0. Consequently, among all the values $c_i$ only one is also equal to 1, and the rest are 0.

Therefore, the result of the product corresponds to the three outputs of the circuit, on which the logical variables $c_i$ are formed.

Since of all the values $b_i$ only one is equal to 1, then three options are possible.

If $b_1 = 1$, then

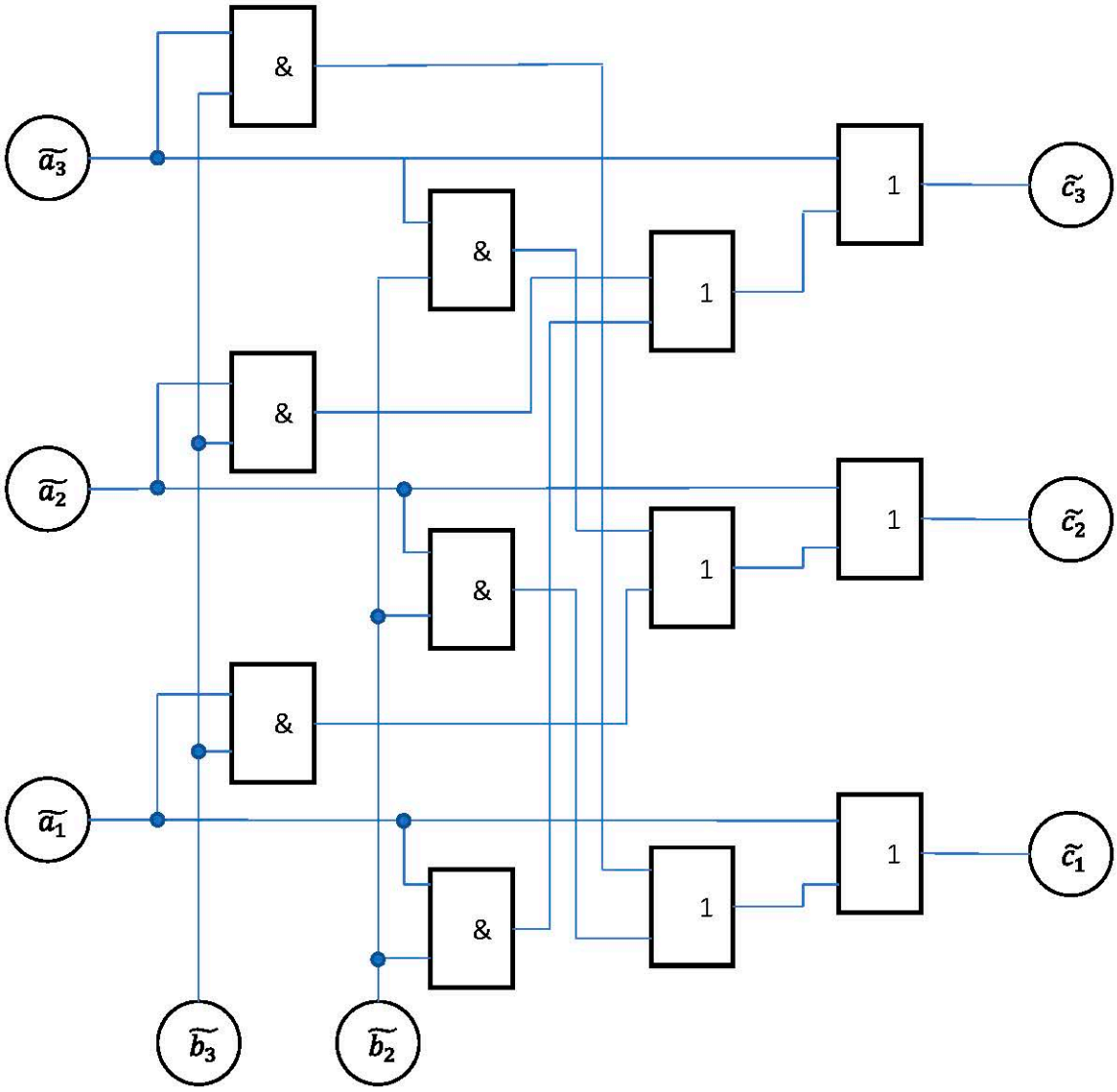


**Figure 4.** Schematic diagram of a direct multiplier (Image was generated in PowerPoint Microsoft 365 https://www.microsoft.com/en-ww/microsoft-365/powerpoint).

$$(c_3, c_2, c_1) = (a_3, a_2, a_1) \tag{25}$$

If $b_2 = 1$, then

$$(c_3, c_2, c_1) = (a_2, a_1, a_3) \tag{26}$$

If $b_3 = 1$, then

$$(c_3, c_2, c_1) = (a_1, a_3, a_2) \tag{27}$$

If $b_1 = 1$, then the state of outputs c_i repeats the state of inputs $a_i$, if $b_2 = 1$, then there is a cyclic permutation one position to the right, and if $b_3 = 1$, then one position to the left.

A scheme that provides such a permutation can be implemented in various ways. One of them is based on a set of operations, which can be represented schematically as follows.

$$NO[(0,1,0)OR(0,0,1)] \rightarrow NO(0,1,1) \rightarrow (1,0,0) \tag{28}$$

$$(0,1,0)OR(0,0,0) \rightarrow (0,1,0) \tag{29}$$

These notations imply that the *NO* and *OR* operations are applied to each of the boolean variables appearing in the sequences. Formulas (28) and (29) show only a particular case; they obviously remain valid for cyclic permutations as well.

From these formulas it follows that the permutation corresponding to formulas (25)–(27) can also be implemented in the way that is implemented by the scheme of Fig. 4.

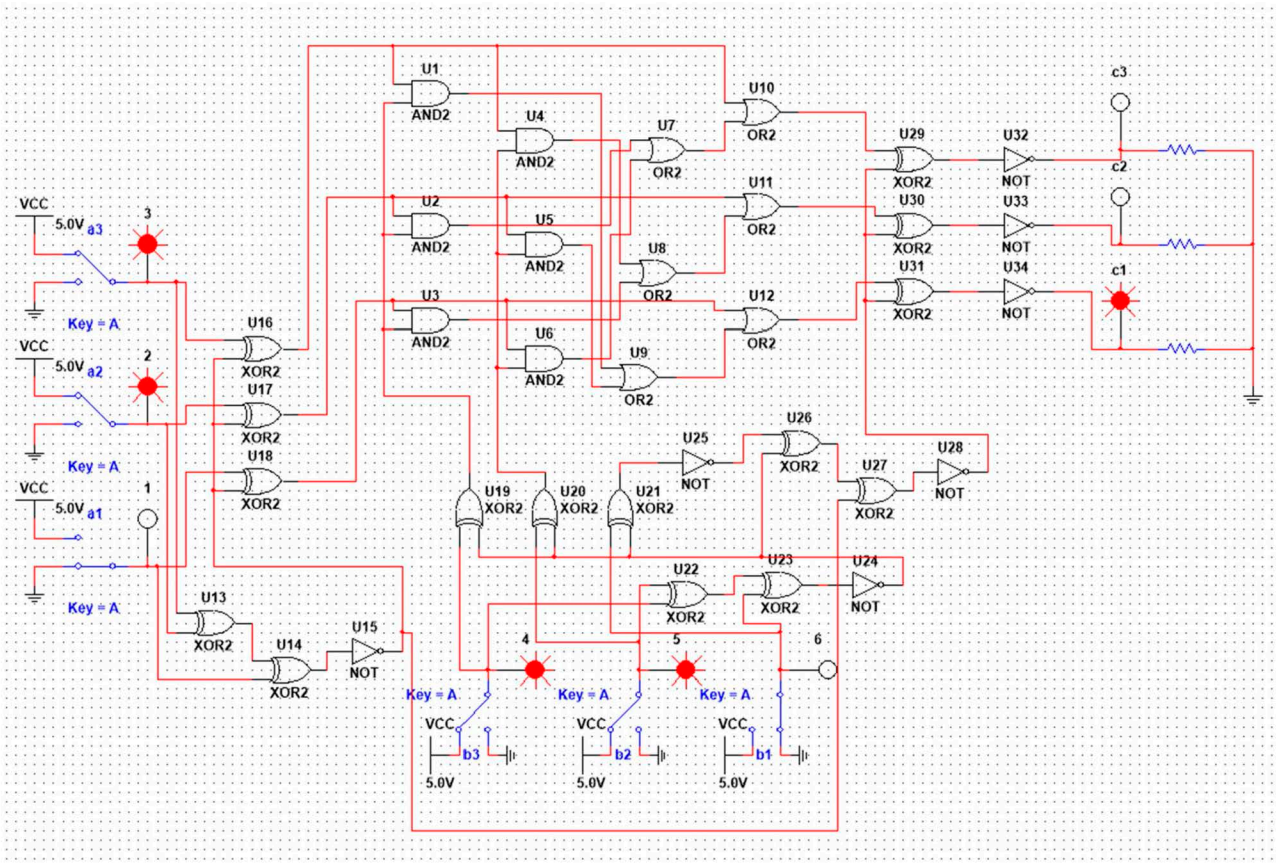


**Figure 5.** Multiplier circuit modulo seven, assembled in the NI Multisim program. (Image was generated in NI Multisim 14.1 https://www.ni.com/ru-ru/support/downloads/software-products/download.multisim.html#306441).

In this case, if $b_1 = 0$, an additional inversion of the signal is performed, which corresponds to the execution of operation (28). According to the diagram in Fig. 4 this operation is performed by the adder, the output of which is connected to the output inverter.

The complete scheme, made in the NI Multisim application[31], is shown in Fig. 5.

As follows from the above description of the multiplier, its scheme takes into account the most important specific features of computational systems carrying out operations in Galois fields, which are directly connected with operations of multivalued logic.

## Conclusion

This paper shows that the problem of interpreting the variables of multivalued logic does not necessarily have to be solved through the involvement of the philosophical category of truth. A possible option is to use a close connection between the variables of multivalued logics, whose number of elements is equal to the degree of a prime number, with Galois fields. In this case it is possible, among other things, to establish a connection between the variables of multivalued logic and the linguistic variables used in fuzzy logic. In addition, this relationship allows to reduce any operations on variables of multivalued logic to the calculation of algebraic functions whose arguments take a value in the Galois field. Otherwise, any operations of multivalued logics of the specified type can be reduced to the operations of addition and multiplication modulo the degree of a prime number.

Such operations, in their turn, can be realized by means of radio electronic circuits assembled on typical elements, performing operations of binary logic. At the same time, as shown in the example of implementation of such circuits, they can be quite simple.

## Data availability

All data generated or analysed during this study are included in this published article.

Received: 4 October 2022; Accepted: 16 January 2023
Published online: 20 January 2023

## References

1. Lukasiewicz, J. On three-valued logic. The Polish Review, 43–44 (1968).

2. Gomes, E. L. Thinking about Contradictions: The Imaginary Logic of Nikolai Aleksandrovich Vasil'ev: V. Raspa, Translated by Peter N. Dale. Heidelberg, New York: Springer International Publishing AG, 2017. xxi+ 160 pp (2019). Hardcover US 109.99,e-bookUS 84.99. Hardcover ISBN 978-3-319-66085-1. eBook ISBN 978-3-319-66086-8. https://doi.org/10.1080/01445340.2019.1591669.
3. Kline, M. *Mathematics: The Loss of Certainty* Vol. 686 (Galaxy Books, 1982).
4. Jo, S. B., Kang, J. & Cho, J. H. Recent advances on multivalued logic gates: a materials perspective. *Adv. Sci.* **8**(8), 2004216. https://doi.org/10.1002/advs.202004216 (2021).
5. Yoo, H. & Kim, C. H. Multi-valued logic system: New opportunities from emerging materials and devices. *J. Mater. Chem. C***9**(12), 4092–4104. https://doi.org/10.1039/D1TC00148E (2021).
6. Zamansky, A. On recent applications of paraconsistent logic: an exploratory literature review. *J. Appl. Non-Classical Logics* **29**(4), 382–391. https://doi.org/10.1080/11663081.2019.1656393 (2019).
7. Hernández-Tello, A., Macías, V. B. & Coniglio, M. E. Paracomplete logics dual to the genuine paraconsistent logics: The three-valued case. *Electron. Notes Theor. Comput. Sci.* **354**, 61–74. https://doi.org/10.1016/j.entcs.2020.10.006 (2020).
8. Gabrielyan, O., Vitulyova, E. & Suleimenov, I. Multi-valued logics as an advanced basis for artificial intelligence (as an example of applied philosophy). *Wisdom* **21**(1), 170–181. https://doi.org/10.24231/wisdom.v21i1.721 (2022).
9. Nakayama, Y., Akama, S. & Murai, T. Deduction system for decision logic based on many-valued logics. *Int. J. Adv. Intell. Syst.* **11**(1/2), 115–126 (2018).
10. Lethen, T. Gödel on many-valued logic. *Rev. Symb. Log.* https://doi.org/10.1017/S1755020321000034 (2021).
11. Rescher, N. *Topics in Philosophical Logic* Vol. 17 (Springer, 2013).
12. Kahane, H., Hausman, A. & Boardman, F. *Logic and Philosophy: A Modern Introduction* (Hackett Publishing, 2020).
13. Suleimenov, I. E., Gabrielyan, O. A., Bakirov, A. S., & Vitulyova, Y. S. Dialectical understanding of information in the context of the artificial intelligence problems. In *IOP Conference Series: Materials Science and Engineering*, Vol. 630, No. 1, p. 012007, 2019, October). IOP Publishing.
14. Suleimenov, I., Bakirov, A., & Moldakhan, I. Formalization of ternary logic for application to digital signal processing. In *Energy Management of Municipal Transportation Facilities and Transport*, 26–35. Springer, Cham, 2019, December. https://doi.org/10.1007/978-3-030-57453-6_3.
15. Dey, S., & Ghosh, R. (2018). 4-bit crypto S-boxes: Generation with irreducible polynomials over Galois field GF (24) and cryptanalysis. Cryptology ePrint Archive.
16. Wang, N. *et al.* Galois fieldbased image encryption for remote transmission of tumor ultrasound images. *IEEE Access* **7**, 49945–49950. https://doi.org/10.1109/ACCESS.2019.2910563 (2019).
17. Khari, M. *et al.* Securing data in Internet of Things (IoT) using cryptography and steganography techniques. *IEEE Trans. Syst. Man Cybern. Syst.* **50**(1), 73–80. https://doi.org/10.1109/TSMC.2019.2903785 (2019).
18. Moldakhan, I., Matrassulova, D. K., Shaltykova, D. B. & Suleimenov, I. E. Some advantages of non-binary Galois fields for digital signal processing. *Indonesian J. Electr. Eng. Comput. Sci.* **23**(2), 871–877. https://doi.org/10.11591/ijeecs.v23.i2.pp871-878 (2021).
19. Vitulyova, E. S., Matrassulova, D. K. & Suleimenov, I. E. New application of non-binary Galois fields Fourier transform: Digital analog of convolution theorem. *Indonesian J. Electr. Eng. Comput. Sci.* **23**(3), 1718–1726. https://doi.org/10.11591/ijeecs.v23.i3.pp1718-1726 (2021).
20. Mittal, K., Jain, A., Vaisla, K. S., Castillo, O. & Kacprzyk, J. A comprehensive review on type 2 fuzzy logic applications: Past, present and future. *Eng. Appl. Artif. Intell.* **95**, 103916. https://doi.org/10.1016/j.engappai.2020.103916 (2020).
21. Marcos, J. On a problem of da Costa. *Essays on the Foundations of Mathematics and Logic* **2**, 39–55 (2005).
22. Rezk, H., Aly, M., Al-Dhaifallah, M. & Shoyama, M. Design and hardware implementation of new adaptive fuzzy logic-based MPPT control method for photovoltaic applications. *IEEE Access* **7**, 106427–106438. https://doi.org/10.1109/ACCESS.2019.2932694 (2019).
23. Sharma, R., Bhasin, S., Gaur, P. & Joshi, D. A switching-based collaborative fractional order fuzzy logic controllers for robotic manipulators. *Appl. Math. Model.* **73**, 228–246. https://doi.org/10.1016/j.apm.2019.03.041 (2019).
24. Sedova, N., Sedov, V., Bazhenov, R. & Bogatenkov, S. Neural network classifier for automatic course-keeping based on fuzzy logic. *J. Intell. Fuzzy Syst.* **40**(3), 4683–4694. https://doi.org/10.3233/JIFS-201495 (2021).
25. Vitulyova, Y. S., Bakirov, A. S., Baipakbayeva, S. T., & Suleimenov, I. E. Interpretation of the category of "complex" in terms of dialectical positivism. In *IOP Conference Series: Materials Science and Engineering* (Vol. 946, No. 1, p. 012004), 2020, October. IOP Publishing. https://doi.org/10.1088/1757-899X/946/1/012004.
26 Kalimoldayev, M. N. *et al.* Methodological basis for the development strategy of artificial intelligence systems in the Republic of Kazakhstan in the message of the president of the Republic of Kazakhstan dated October 5, 2018. News of the National Academy of Sciences of the Republic of the Kazakhstan. *Ser. Geol. Tech. Sci.* **6**, 47–54. https://doi.org/10.32014/2018.2518-170X.34 (2018).
27. Suleimenov, I. E., Vitulyova, Y. S., Bakirov, A. S., & Gabrielyan, O. A. Artificial Intelligence: what is it? In *Proceedings of the 2020 6th International Conference on Computer and Technology Applications*, 22–25, 2020, April. https://doi.org/10.1145/3397125.3397141.
28. Moysis, L., Petavratzis, E., Volos, C., Nistazakis, H. & Stouboulos, I. A chaotic path planning generator based on logistic map and modulo tactics. *Robot. Autonom. Syst.* **124**, 103377. https://doi.org/10.1016/j.robot.2019.103377 (2020).
29. Prasanna, D., Sriram, C. & Murthy, C. R. On the identifiability of sparse vectors from modulo compressed sensing measurements. *IEEE Signal Process. Lett.* **28**, 131–134. https://doi.org/10.1109/LSP.2020.3047584 (2020).
30. Țiplea, F. L., Iftene, S., Teşeleanu, G. & Nica, A. M. On the distribution of quadratic residues and non-residues modulo composite integers and applications to cryptography. *Appl. Math. Comput.* **372**, 124993. https://doi.org/10.1016/j.amc.2019.124993 (2020).
31. NI Multisim 14.1 https://www.ni.com/ru-ru/support/downloads/software-products/download.multisim.html#306441..


## Author contributions

All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Suleimenov I.E., Vitulyova Ye.S. Kabdushev Sh.B., Bakirov A.S. The first draft of the manuscript was written by Suleimenov I.E. and all authors commented on previous versions of the manuscript. Bakirov A.S. made all revisions according to reviewers' comments. All authors read and approved the final manuscript.


## Funding

This research has been/was/is funded by the Science Committee of the Ministry of Higher Education and Science of the Republic of Kazakhstan (Grant No. AP14870281).


## Competing interests

The authors declare no competing interests.

## Additional information

**Correspondence** and requests for materials should be addressed to A.S.B.

PLOS ONE

RESEARCH ARTICLE

# Features of digital signal processing algorithms using Galois fields GF($2^n$+1)

**Ibragim E. Suleimenov**[1], **Yelizaveta S. Vitulyova**[2], **Dinara K. Matrassulova**[2]*

**1** National Engineering Academy of the Republic of Kazakhstan, Almaty, Kazakhstan, **2** Almaty University of Power Engineering and Telecommunications Named After Gumarbek Daukeyev, Almaty, Republic of Kazakhstan

* dinara.kutlimuratovna@gmail.com





**Citation:** Suleimenov IE, Vitulyova YS, Matrassulova DK (2023) Features of digital signal processing algorithms using Galois fields GF ($2^n$+1). PLoS ONE 18(10): e0293294. https://doi.org/10.1371/journal.pone.0293294

**Editor:** Pierluigi Vellucci, Roma Tre University: Universita degli Studi Roma Tre, ITALY

**Received:** March 10, 2023

**Accepted:** October 10, 2023

**Published:** October 25, 2023



**Data Availability Statement:** There is no additional data, all data is available in the article itself.

**Funding:** This research has been/was/is funded by the Science Committee of the Ministry of Higher Education and Science of the Republic of Kazakhstan (Grant No. AP14870281) The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript. All authors receive a salary from the received research grant.

**Competing interests:** The authors have declared that no competing interests exist.

## Abstract

An alternating representation of integers in binary form is proposed, in which the numbers -1 and +1 are used instead of zeros and ones. It is shown that such a representation creates considerable convenience for multiplication numbers modulo p = $2^n$+1. For such numbers, it is possible to implement a multiplication algorithm modulo p, similar to the multiplication algorithm modulo the Mersenne number. It is shown that for such numbers a simple algorithm for digital logarithm calculations may be proposed. This algorithm allows, among other things, to reduce the multiplication operation modulo a prime number p = $2^n$+1 to an addition operation.

## Introduction

Currently, non-binary Galois fields are increasingly used in information technologies [1, 2], in particular, in information security systems [3, 4]. The example is a variety of Galois fields *GF* (*p*), which are residue classes of the ring of integers modulo some prime number *p*. The advantages of using such fields for digital signal processing were clearly demonstrated in [5, 6].

There is also a number of reports devoted to the development of electronic circuits that perform addition and multiplication operations modulo in current literature, for example, [7, 8]. Such reports are closely related to research in the field of practical use of Galois fields, since the operations of addition and multiplication modulo an integer can be considered as operations on the elements of the Galois field. The interest is connected, among other things, with the fact that such operations are of significant interest for cryptography [9, 10].

The development of electronic circuits operating in Galois fields is also of interest from the point of view of improving artificial intelligence (AI) systems. Namely, as it was shown in the reports [11, 12] the further development of AI, assuming its gradual approach to the biological prototype, cannot exclude the transition to multivalued logic, since human thinking is irreducible to binary logic. It is appropriate to emphasize that the problem of creating of AI approaching a biological prototype is of considerable interest, including from the point of view of revealing the essence of intelligence as such [13, 14].

Multivalued logic, dating back to the works of Jan Lukasiewicz [15], has been actively developing recently [16–19], but the level of its practical use obviously does not meet the existing

potential [11]. If the number of values accepted by variables of multivalued logic is equal to an integer power of some prime number, these values can be put in one-to-one correspondence to the elements of the Galois field. Consequently, any operations in this case are reduced to addition and multiplication, i.e., the improvement of electronic circuits that perform addition and multiplication modulo an integer is also of interest from this point of view. The possibilities arising in this case are disclosed, in particular, in [20].

In turn, among the simple Galois fields $GF(p)$, a special place is occupied by fields for which the number $p$ is equal to the Mersenne prime number. Such numbers are representable as

$$p = 2^n - 1 \tag{1}$$

where $n$ are specifically selectable integers, the first of which are, 2, 3, 5, and 7.

Such numbers are used, in particular, to generate pseudorandom numbers [19]. Specifically, a pseudorandom number generator called the Mersenne twister is known, developed in 1997 by Japanese scientists M. Matsumoto and T. Nishimura [21], directly based on the use of Mersenne numbers.

There are works in which the expediency of using such numbers for data transmission is justified [21, 22], etc.

A very remarkable feature of the Mersenne numbers $p_m$ is the fact that Galois fields $GF(p_m)$ allows one to implement fairly simple electronic circuits that perform addition and multiplication operations modulo $p_m$, i.e., those operations to which any other operations performed on the elements of the field $GF(p_m)$ are reduced. Examples of such systems and algorithms of their functioning are reflected in the current literature, for example [23].

In this paper, it is shown that Galois fields, $GF(p)$, for which $p = 2^n+1$., are also of considerable interest for digital signal processing. One of these fields is the $GF(257)$ field, which is advisable to use for digital signal processing with a standard number of levels equal to 256. Thus, it can be argued that for this case there is a very specific Galois field, which, among other things, allows you to bring signals that meet existing standards [24], to logical operations.

## Advantages of Mersenne numbers for computing systems modulo an integer

In relation to digital signal processing, the following property of Mersenne numbers is of interest. Multiplication of a number $a$ written in binary form by 2 modulo the Mersenne number is recused to cyclic permutation of characters takes place. For example, for calculations in the field $GF(7)$, next equality is true

$$2 \cdot a_2a_1a_0 =_{(7)} a_1a_0a_2 \tag{2}$$

where $a_i$ are binary characters.

The convenience of using such a property for digital signal processing in Galois fields is as follows. Let us consider the field $GF(127)$ whose characteristic is a Mersenne number with $n = 7$. Note that this example is also important from a practical point of view, since a scale with 127 levels is often used in digital signal processing too. In binary form, any of the elements of the $GF(127)$ field can be represented as

$$A = 2^6 \cdot a_6 + 2^5 \cdot a_5 + \cdots + 2^1 \cdot a_1 + 2^0 \cdot a_0 \tag{3}$$

We emphasize that the number of elements represented in the form (3) when calculating in terms of ordinary integers is 128, but

$$1111111 \equiv 0000000(127) \tag{4}$$

Consequently, the number of field elements given by Formula (3) is indeed 127.

Let the element $A$, represented in the form (3), be multiplied modulo 127 by the element $B$, represented in the same form.

$$B = 2^6 \cdot b_6 + 2^5 \cdot b_5 + \cdots + 2^1 \cdot b_1 + 2^0 \cdot b_0 \tag{5}$$

Then the result of the product is the sum of the following terms

$$\begin{gathered} A \cdot b_0 = (2^6 \cdot a_6 + 2^5 \cdot a_5 + \cdots + 2^1 \cdot a_1 + 2^0 \cdot a_0)b_0 \\ 2^1 \cdot A \cdot b_1 = (2^6 \cdot a_5 + \cdots + 2^2 \cdot a_1 + 2^1 \cdot a_0 + 2^0 \cdot a_6)b_1 \\ \cdots \\ 2^6 \cdot A \cdot b_6 = (2^6 \cdot a_0 + 2^5 \cdot a_6 + \cdots + 2^1 \cdot a_2 + 2^0 \cdot a_1)b_6 \end{gathered} \tag{6}$$

Grouping the terms at the same powers of two, we get

$$\begin{gathered} c_6 = a_6 b_0 + a_5 b_1 + \cdots + a_0 b_6 \\ c_5 = a_5 b_0 + a_4 b_1 + \cdots + a_6 b_6 \\ \cdots \\ c_0 = a_0 b_0 + a_6 b_1 + \cdots + a_1 b_6 \end{gathered} \tag{7}$$

We emphasize that although Formula (7) include only quantities taking the values 0 or 1, no such restrictions are imposed on the summation result itself, i.e., the sum is calculated in the sense of the original field $GF(127)$.

The algorithm based on Formula (7) allows for a fairly simple circuit implementation, therefore, it makes sense to consider whether there is no way to implement its analogue for fields $GF(2^n+1)$, in particular, for the field $GF(257)$. This is due to an obvious consideration: the signal digitization scale, which provides for the use of 256 levels, is one of the most common [24].

## Algorithms of calculations modulo in the special case of the field *GF*(17)

Let's start from the consideration of one of the simplest fields of type $GF(2^n+1)$, specifically from the field $GF(17)$.

This field, as well as other Galois fields, is a ring of residue classes of the ring of integers, in this case modulo 17. Traditionally, positive integers are used when representing field elements, however, this is not mandatory at all. In particular, as emphasized in [25], it is advisable to use a set of elements $\{-1,0,1\}$ to represent the field $GF(3)$, where the use of curly brackets emphasizes that the set is being considered.

Similarly, the field $GF(17)$ can be considered as a set of elements

$$GF(17) = \{-8, -7, \ldots, 0, \ldots 7, 8\} \tag{8}$$

The selection of the representing elements is arbitrary up to the modulo comparison operation, for example,

$$-8 \equiv 9(17) \tag{9}$$

The advantages of such a choice for the purposes of this work are demonstrated by Tables 1 and 2. In these tables, the degrees of the elements of the field under consideration are counted in the usual representation through positive integers and in the representation (8), respectively.

Both tables demonstrate the fact that all nonzero elements of the field under consideration, as follows from the general theory of Galois fields, obey the equation

$$x^{16} - 1 = 0 \tag{10}$$

From this relation, in particular, it follows that any element of the field under consideration can be represented as

$$x = (\sqrt[2]{1})_4^{s_4} (\sqrt[4]{1})_3^{s_3} (\sqrt[8]{1})_2^{s_2} (\sqrt[16]{1})_1^{s_1} \tag{11}$$

where $s_i = 0; 1$.

We show this by first revealing the meaning of the formal notation $(\sqrt[k_2]{1})_{k_1}$ using Tables 1 and 2.

The ratio (10) can be rewritten in the form

$$(x^8)^2 - 1 = 0 \tag{12}$$

Formula (12) emphasizes that there are only two different values of the element $z = x^8$. This corresponds to the fact that in the fourth column of Tables 1 and 2 there are only two different elements.

**Table 1. The degrees of the elements of the *GF*(17) field in terms of positive integers.**

| $n = 0$ | $n = 2$ | $n = 4$ | $n = 8$ | $n = 16$ |
|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 |
| 2 | 4 | 16 | 1 | 1 |
| 3 | 9 | 13 | 16 | 1 |
| 4 | 16 | 1 | 1 | 1 |
| 5 | 8 | 13 | 16 | 1 |
| 6 | 2 | 4 | 16 | 1 |
| 7 | 15 | 4 | 16 | 1 |
| 8 | 13 | 16 | 1 | 1 |
| 9 | 13 | 16 | 1 | 1 |
| 10 | 15 | 4 | 16 | 1 |
| 11 | 2 | 4 | 16 | 1 |
| 12 | 8 | 13 | 16 | 1 |
| 13 | 16 | 1 | 1 | 1 |
| 14 | 9 | 13 | 16 | 1 |
| 15 | 4 | 16 | 1 | 1 |
| 16 | 1 | 1 | 1 | 1 |

https://doi.org/10.1371/journal.pone.0293294.t001

**Table 2. Degrees of elements of the field *GF*(17) in the representation (8).**

| $n = 0$ | $n = 2$ | $n = 4$ | $n = 8$ | $n = 16$ |
|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 |
| 2 | 4 | -1 | 1 | 1 |
| 3 | -8 | -4 | -1 | 1 |
| 4 | -1 | 1 | 1 | 1 |
| 5 | 8 | -4 | -1 | 1 |
| 6 | 2 | 4 | -1 | 1 |
| 7 | -2 | 4 | -1 | 1 |
| 8 | -4 | -1 | 1 | 1 |
| -8 | -4 | -1 | 1 | 1 |
| -7 | -2 | 4 | -1 | 1 |
| -6 | 2 | 4 | -1 | 1 |
| -5 | 8 | -4 | -1 | 1 |
| -4 | -1 | 1 | 1 | 1 |
| -3 | -8 | -4 | -1 | 1 |
| -2 | 4 | -1 | 1 | 1 |
| -1 | 1 | 1 | 1 | 1 |

https://doi.org/10.1371/journal.pone.0293294.t002

There are two other similar forms of representation of the relation (10).

$$(x^4)^4 - 1 = 0 \tag{13}$$

$$(x^2)^8 - 1 = 0 \tag{14}$$

Formula (13) emphasizes that there are four different values of the element $z = x^4$, which are found in the third columns of Tables 1 and 2. Similarly, as the ratio (14) shows, there are eight different elements $z = x^2$, which are found in the second columns of these tables.

Accordingly, formally we can write

$$(\sqrt[16]{1})_1 = 16 \tag{15}$$

$$(\sqrt[16]{1})_2 = 4; 13 \tag{16}$$

$$(\sqrt[16]{1})_3 = 2; 8; 9; 15 \tag{17}$$

The eight remaining y elements highlighted in Tables 1 and 2 in color represent primitive elements. They have the property that $y^m = 1$ if and only if $m = 16$. We have

$$(\sqrt[16]{1})_4 = 3; 5; 6; 7; 10; 11; 12; 14 \tag{18}$$

The proved relation (11) follows directly from Formulas (15)–(18), and the choice of elements $(\sqrt[k2]{1})_{k_1}$ is determined by the following considerations. Any nonzero element of the field under consideration represents some degree of one of the primitive elements $y$ listed in the right part of Formula (18). This follows from the general theory of Galois fields, and for clarity it can be demonstrated as follows.

All the degrees of the primitive element y of the field under consideration from 0 to 15 are different. At the same time, all these degrees are the roots of Eq (10), i.e., they exhaust the elements of the field.

Consider the power of $y^m$ and represent the number m, where $0 \leq m \leq 15$ in binary form

$$m = m_4 m_3 m_2 m_1 \tag{19}$$

where $m_i$ are binary characters.

Such an entry means that

$$m = 2^3 \cdot m_4 + 2^2 \cdot m_3 + 2^1 \cdot m_2 + 2^0 \cdot m_1 \tag{20}$$

Therefore, the degree $y^m$ is representable as

$$y^m = (y^8)^{m_4} (y^4)^{m_3} (y^2)^{m_2} (y^1)^{m_1} \tag{21}$$

This expression coincides with (11) if we put the corresponding root of unity equal to one of the powers of the primitive element appearing in (21).

For further it is essential that when moving to the representation of the elements of the field under consideration in the form (8), the roots of unity (15)–(18) acquire a symmetrical form that and shows Table 2.

$$(\sqrt[16]{1})_1 = -1 \tag{22}$$

$$(\sqrt[16]{1})_2 = 4; -4 \tag{23}$$

$$(\sqrt[16]{1})_3 = 2; 8; -8; -2 \tag{24}$$

$$(\sqrt[16]{1})_4 = 3; 5; 6; 7; -7; -6; -5; -3 \tag{25}$$

This determines the convenience of representation in the form (8): all elements of the field under consideration, with the exception of primitive ones, are representable as powers of two, which is also emphasized by Table 2.

This fact generates another efficient algorithm for multiplying field elements, de facto based on the digital logarithm method. Digital logarithm has been considered in many reports, in particular in [26, 27], but this problem, if put in a general form, remains unresolved.

However, for practical needs, it can be limited to solving it for specific Galois fields. In particular, within the framework of this work, it is solved in relation to Galois fields of the form $GF(2^p+1)$, which, as noted above, also includes the field $GF(257)$, which corresponds to the number of digital signal levels that is often used in practice.

In relation to the $GF(17)$ field, the digital logarithm algorithm can be constructed as follows.

1. The field element is identified as belonging to a set of primitive elements.
2. If this element belongs to the specified set, then the value of $m_1$ in Formula (21) is chosen to be 1, if not, then zero.
3. If $m_1 = 1$, then the element in question is multiplied by 3 modulo 17. As a result of the representation in the form (8), the element is reduced to the power of two. When using binary notation, this means that the logical unit will stand only on one of the positions, which identifies the exponents $m_2$, $m_3$ and $m_4$.
4. The set of exponents of degrees $m_i$ completes the procedure of digital logarithm.

Solving the digital logarithm problem, in turn, makes it possible to significantly simplify the algorithm for multiplying field elements on each other. Indeed, in this case, the operation is

reduced to the addition of numbers (20) modulo 16. Carrying out such an operation by circuit means does not cause difficulties, since it boils down to the usual addition of binary numbers with the rejection of the highest digit.

The described algorithm really makes it possible to significantly simplify the multiplication operation in the *GF*(17) field, but, firstly, the question of the circuit identification of primitive elements remains open, and secondly, this algorithm is very specific. Indeed, it is built on the fact that any of the primitive elements is reduced to the power of two (with a positive or negative sign) by multiplying by a fixed element (for example, by 3). This situation is not realized for other Galois fields of the type under consideration, in particular, for the field *GF*(257), which is of primary interest from the point of view of practical applications.

## Alternating binary encoding

We show that the indicated problem is solved using alternating encoding of elements of Galois fields of the type under consideration.

Again, let's start from the example of the field *GF*(17). Consider an expression of the form (3), but only now we will understand by $a_i$ the numbers taking the values +1 and -1.

$$A = 2^3 \cdot a_3 + 2^2 \cdot a_2 + 2^1 \cdot a_1 + 2^0 \cdot a_0 \tag{26}$$

It can be easily shown that the result of calculations by Formula (3) in this case will certainly give an odd number.

In total, there are $2^4$ such combinations of the form (3), and the maximum number is

$$A_{max} = 2^3 + 2^2 + 2^1 + 2^0 = 15, \tag{27}$$

and, accordingly, $A_{min} = -15$.

Otherwise, the set of numbers given by Formula (26) coincides with the set of odd numbers in the range from -16 to +16. We exclude zero from consideration, which is quite justified, since the multiplication operation is being considered. Then the number of combinations of the form (26) in the case under consideration coincides with the number of non-zero elements of the field *GF*(17).

All possible combinations are listed in Tables 3 and 4. Table 3 refers to the elements of the field that are not primitive, Table 4 –vice versa. The first four columns of these tables display the numbers $a_i$ appearing in expressions (26), the fifth column displays the result of summation in terms of ordinary integers $A_1$, the sixth–displays the result of reduction modulo 17 to the form (8) A.

**Table 3. Alternating binary representation of elements of the *GF*(17) field that are not primitive.**

| $a_3$ | $a_2$ | $a_1$ | $a_0$ | $A_1$ | $A$ |
|---|---|---|---|---|---|
| 1 | -1 | -1 | -1 | 1 | 1 |
| -1 | -1 | -1 | -1 | -15 | 2 |
| -1 | -1 | -1 | 1 | -13 | 4 |
| -1 | -1 | 1 | 1 | -9 | 8 |
| -1 | 1 | 1 | 1 | -1 | 1 |
| 1 | 1 | 1 | 1 | 15 | -2 |
| 1 | 1 | 1 | -1 | 13 | -4 |
| 1 | 1 | -1 | -1 | 9 | -8 |
| 1 | -1 | -1 | -1 | 1 | 1 |

https://doi.org/10.1371/journal.pone.0293294.t003

**Table 4. Alternating binary representation of primitive elements of the *GF*(17) field.**

| $a_3$ | $a_2$ | $a_1$ | $a_0$ | $A_1$ | $A$ |
|---|---|---|---|---|---|
| 1 | -1 | -1 | 1 | 3 | 3 |
| -1 | -1 | 1 | -1 | -11 | 6 |
| -1 | 1 | -1 | 1 | -5 | -5 |
| 1 | -1 | 1 | 1 | 7 | 7 |
| -1 | 1 | 1 | -1 | -3 | -3 |
| 1 | 1 | -1 | 1 | 11 | -6 |
| 1 | -1 | 1 | -1 | 5 | 5 |
| -1 | 1 | -1 | -1 | -7 | -7 |
| 1 | -1 | -1 | 1 | 3 | 3 |

https://doi.org/10.1371/journal.pone.0293294.t004

It can be seen that the modulo reduction field of the numbers represented in the form (26) exhaust the set of nonzero elements of the field *GF*(17). Consequently, such a representation can be used along with any other, especially if we take into account that representatives of the residue classes of the ring of integers modulo a prime number can be chosen arbitrarily.

As applied to the field under consideration, a representation of the form (26) in which $a_i = \pm 1$ has a property similar to the property possessed by Mersenne primes. Namely, the multiplication of the number written in the representation (26), which hereafter we will call alternating, by two can be displayed as the following operation

$$2 \cdot A = 2^3 \cdot a_2 + 2^2 \cdot a_1 + 2^1 \cdot a_0 - 2^0 \cdot a_3 \tag{28}$$

This follows from the fact that the element -1 is the root of Eq (12) considered at $z = x^8$ or from the fact that in the field under consideration $2^4 \equiv -1(17)$.

Consequently, the operation of multiplication by two when using an alternating binary representation of a number is reduced to a cyclic permutation of binary elements in the entry resulting from (18) with a change in the sign of the element being rearranged. We have

$$2 \cdot A = 2 \cdot a_2 a_1 a_0 a_3 = a_2 a_1 a_0 (-a_3) \tag{29}$$

It can be seen that this property is indeed analogous to the property possessed by the operation of multiplication by two in fields formed using Mersenne numbers (2).

Property (29), in particular, allows you to implement an algorithm for multiplying two elements of the field *GF*(17), represented in alternating binary form, similar to the algorithm given by Formulas (6), (7).

Indeed, the result of multiplying the number *A*, represented in alternating form (26), by the number *B*, represented in the same form

$$B = 2^3 \cdot b_3 + 2^2 \cdot b_1 + 2^1 \cdot b_1 + 2^0 \cdot b_0 \tag{30}$$

it is the sum of the following terms

$$A \cdot b_0 = (2^3 \cdot a_3 + 2^2 \cdot a_2 + 2^1 \cdot a_1 + 2^0 \cdot a_0) b_0 \tag{31}$$

$$2^1 \cdot A \cdot b_1 = (2^3 \cdot a_2 + 2^2 \cdot a_1 + 2^1 \cdot a_0 - 2^0 \cdot a_3) b_1 \tag{32}$$

$$2^2 \cdot A \cdot b_2 = (2^3 \cdot a_1 + 2^2 \cdot a_0 - 2^1 \cdot a_3 - 2^0 \cdot a_2) b_2 \tag{33}$$

$$2^3 \cdot A \cdot b_3 = (2^3 \cdot a_0 - 2^2 \cdot a_3 - 2^1 \cdot a_2 - 2^0 \cdot a_1)b_3 \quad (34)$$

Grouping elements at powers of two, we obtain the following expressions for the coefficients $c_i$, that arise when multiplying two elements of the field

$$c_1 = a_3b_0 + a_2b_1 + a_1b_2 + a_0b_3 \quad (35)$$

$$c_2 = a_2b_0 + a_1b_1 + a_0b_2 - a_3b_3 \quad (36)$$

$$c_3 = a_1b_0 + a_0b_1 - a_3b_2 - a_2b_3 \quad (37)$$

$$c_4 = a_0b_0 - a_3b_1 - a_2b_2 - a_1b_3 \quad (38)$$

It can be seen that in Formulas (35)–(38) the coefficients $a_i$ are rearranged cyclically with a sign change, which corresponds to the specifics of the field under consideration. We also emphasize that the coefficients $c_i$ are not necessarily equal to ±1, i.e., they are not coefficients in the representation of the form (26). These are the weights with which the powers of two are added when calculating the result of the product.

The algorithm based on Formulas (35)–(38) can be implemented quite simply schematically, moreover, it admits an obvious generalization to any fields of $GF(2^n+1)$. However, the fact that the coefficients $c_i$ are not coefficients in binary alternating representation makes us consider further simplifying the circuit implementation of multipliers in fields of the type under consideration.

## Algorithmic basis of digital logarithm in the $GF(2^n+1)$ field

A significant simplification of the circuit implementation of the multiplier in the field under consideration can be achieved by using the digital logarithm operation, as noted above.

In relation to the field $GF(17)$ under consideration, the operation of digital logarithm is reduced to the operation of circuit identification of primitive elements. As can be seen from Table 2, multiplication by a primitive element leads to the fact that the number becomes equal to the power of two, which is easily identified schematically when using binary representation.

To identify primitive elements, you can use the following technique, which follows from the property of a cyclic permutation with a change in the sign of the element being rearranged. In the Table 5 presents examples of continuation of the sequence of characters $a_i$, appearing in the alternating binary representation of the elements of the field $GF(17)$. Each of these sequences may include one additional element $-a_3$, as illustrated in Table 5.

**Table 5. Example of continuation of a sequence of binary characters with alternating representation of elements of the $GF(17)$ field.**

| $a_3$ | $a_2$ | $a_1$ | $a_0$ | $-a_3$ |
|---|---|---|---|---|
| 1 | -1 | -1 | -1 | -1 |
| -1 | -1 | -1 | -1 | 1 |
| -1 | -1 | -1 | 1 | 1 |

https://doi.org/10.1371/journal.pone.0293294.t005

**Table 6. Values of $q_i$ values and their sums for elements of the *GF*(17) field that are not primitive.**

| $q_3 = Q_{3,2}$ | $q_2 = Q_{2,1}$ | $q_1 = Q_{1,0}$ | $q_0 = Q_{0,3}$ | $\Sigma q_i$ |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |

https://doi.org/10.1371/journal.pone.0293294.t006

Let's form the function $Q_{j,j-1}$, which provides counting the number of sign variables in sequences similar to those presented in Table 5. This function is defined as follows

$$q_i = Q_{j,j\ 1} = \begin{cases} 1, a_j a_{j\ 1} = -1 \\ 0, a_j a_{j\ 1} = +1 \end{cases} \quad (39)$$

The values of $q_i$ values for non–primitive field elements are shown in Table 6, and for primitive elements—in Table 7.

The table data show that the sums $\Sigma q_i$ are invariants for the set of primitive elements of the field *GF*(17) (all of them, when using the alternating binary representation of the elements of this field, are formed by a cyclic permutation with a sign change), as well as for the set of elements that are not primitive. In one case, the invariant is 1, in the other– 3.

This approach, based on the calculation of invariants, can easily be generalized to other fields of $GF(2^n+1)$, which directly follows from the fact that multiplication by two reduces to a cyclic permutation with a sign change. In particular, the nonzero elements of the field *GF*(257) decompose into 16 subsets, each of which is formed by cyclic permutations of the type under consideration. This, in turn, follows from the fact that in the field *GF*(257) there is

$$2^8 \equiv -1(257) \quad (40)$$

## Conclusions

Thus, along with a direct algorithm for multiplying numbers in alternating binary representation, given by Formulas (35)–(38), we can propose the following algorithm for performing the multiplication operation for fields $GF(2^n+1)$, in particular, for the field *GF*(17).

**Table 7. Values of $q_i$ values and their sums for primitive elements of the *GF*(17) field.**

| $q_3 = Q_{3,2}$ | $q_2 = Q_{2,1}$ | $q_1 = Q_{1,0}$ | $q_0 = Q_{0,3}$ | $\Sigma q_i$ |
|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 3 |
| 0 | 1 | 1 | 1 | 3 |
| 1 | 1 | 1 | 0 | 3 |
| 1 | 1 | 0 | 1 | 3 |
| 1 | 0 | 1 | 1 | 3 |
| 0 | 1 | 1 | 1 | 3 |
| 1 | 1 | 1 | 0 | 3 |
| 1 | 1 | 0 | 1 | 3 |

https://doi.org/10.1371/journal.pone.0293294.t007

1. The multiplied numbers are translated into binary alternating representation. This is provided by direct recalculation using the formula $A = 2A_0+1$, where $A_0$ is the original number, followed by the representation of the resulting odd number in the form (26).
2. Using binary alternating coefficients of the number $A = 2A_0+1$, the invariants $\sum q_i$ are calculated, which determine the division of the field into subsets, each of which corresponds to multiplication by a power of two.
3. Multiplication is performed by the element corresponding to the invariant $\sum q_i$. As a result, an element is formed that represents the power of two (taking into account the sign). This degree sets the digital logarithm of the field element in question.
4. The exponents are added modulo the power of two, which schematically corresponds to the usual operation of adding binary numbers with dropping the highest digit.
5. The reverse transition from the exponent to the non-zero element of the field $GF(2^n+1)$ is carried out.

In general, the paper shows that the use of alternating binary representation for elements of fields $GF(2^n+1)$ allows them to realize all the same advantages that occur when working with Galois fields corresponding to Mersenne primes.

The most important type of fields of this type is the $GF(257)$ field, since it corresponds to the number of levels of the digitized signal, which is often used in practice. For example, the most commonly used analog-to-digital converters assume the use of 256 levels.

## Author Contributions

**Formal analysis:** Yelizaveta S. Vitulyova.

**Methodology:** Ibragim E. Suleimenov.

**Supervision:** Ibragim E. Suleimenov.

**Visualization:** Dinara K. Matrassulova.

**Writing – original draft:** Ibragim E. Suleimenov.

**Writing – review & editing:** Yelizaveta S. Vitulyova.

## References

1. Lehnigk-Emden T., When N. Complexity evaluation of non-binary Galois field LDPC code decoders. *2010 6th International Symposium on Turbo Codes & Iterative Information Processing, IEEE*, 53–57 (2010).
2. Isla H., Prakash O. New Quantum and LCD Codes over Finite Fields of Even Characteristic. *Defence Science Journal*, 71(5) (2021).
3. Kuo Y. M., Garcia-Herrero F., Ruano O., Maestro J.A. RISC-V Galois Field ISA Extension for Non-Binary Error-Correction Codes and Classical and Post-Quantum Cryptography, *IEEE Transactions on Computers* (2022).
4. Matsumine T., Ochiai H. A design of non-binary turbo codes over finite fields based on Gaussian approximation and union bounds, in *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, 1–5 (2021).
5. Moldakhan I., Matrassulova D. K., Shaltykova D. B., Suleimenov I. E. Some advantages of non-binary Galois fields for digital signal processing, *Indonesian Journal of Electrical Engineering and Computer Science*, 23(2), 871–877 (2021).
6. Vitulyova E. S., Matrassulova D. K., Suleimenov I. E. New application of non-binary Galois fields Fourier transform: Digital analog of convolution theorem, *Indonesian Journal of Electrical Engineering and Computer Science*, 23(3), 1718–1726 (2021).

7. Kalimoldayev M., Tynymbayev S., Gnatyuk S., Ibraimov M., Magzom M. The device for multiplying polynomials modulo an irreducible polynomial, *News of the National Academy of Sciences of the Republic of Kazakhstan Series of Geology and Technical Sciences*, 2(434), 199–205 (2019).
8. Dey S., Ghosh R. 4-bit crypto S-boxes: Generation with irreducible polynomials over Galois field GF (24) and cryptanalysis, *Cryptology ePrint Archive* (2018).
9. Kuppuswamy P., Al-Khalidi S. Q. Implementation of security through simple symmetric key algorithm based on modulo 37, *International Journal of Computers & Technology*, 3(2), 335–338 (2012).
10. Verkhovsky B. S. Enhanced Euclid Algorithm for Modular Multiplicative Inverse and Its Application in Cryptographic Protocols, *Int. J. Commun. Netw. Syst. Sci.*, 3(12), 901–906 (2010).
11. Gabrielyan O.A., Vitulyova Ye. Suleimenov S., I. E. Multi-valued logics as an advanced basis for artificial intelligence, *Wisdom*, 1(21), 170–181 (2022).
12. Vitulyova Y.S., Bakirov A.S., Baipakbayeva S.T., Suleimenov I.E. Interpretation of the category of complex in terms of dialectical positivism, *IOP Conference Series: Materials Science and Engineering*, 946 (1), 012004. https://doi.org/10.1088/1757-899X/946/1/012004 (2020).
13. Gabrielyan O.A., Vitulyova Ye. S., Suleimenov I. E. Multi-valued logics as an advanced basis for artificial intelligence, *Wisdom*, 1(21), 170–181 (2022).
14. Suleimenov I. E., Matrassulova D. K., Moldakhan I., Vitulyova Y. S., Kabdushev S. B., Bakirov A. S. Distributed memory of neural networks and the problem of the intelligence's essence, *Bulletin of Electrical Engineering and Informatics*, 11(1), 510–520 (2022).
15. Lukasiewicz J. "On Three-Valued Logic," *Jan Lukasiewicz. Selected Works / Ed. by* Borkowski L., Amsterdam: North-Holland, 87–88 (1970).
16. Zamansky A. On recent applications of paraconsistent logic: an exploratory literature review, *Journal of Applied Non-Classical Logics*, 29(4), 382–391 (2019).
17. Hernández-Tello A., Macías V. B., Coniglio M. E. Paracomplete Logics Dual to the Genuine Paraconsistent Logics: The Three-valued Case, *Electronic Notes in Theoretical Computer Science*, 354, 61–74 (2019).
18. Nakayama Y., Akama S., Murai T. Deduction System for Decision Logic Based on Many-valued Logics, *International Journal on Advances in Intelligent Systems*, 11(½), 115–126 (2018).
19. Suleimenov I. E., Vitulyova Y. S., Kabdushev S. B., Bakirov A. S. Improving the efficiency of using multi-valued logic tools, *Scientific Reports*, 13(1), 1108 (2023). https://doi.org/10.1038/s41598-023-28272-1 PMID: 36670172
20. Shahov V.V. Review and comparative analysis of pseudo-random number generator libraries, *Problems of computer science*, 2, 66–74.
21. Matsumoto M., Nishimura T. Mersenne Twister: A 623-dimensionally equidistributed uniform pseudo-random number generator, *ACM Trans. Model. Comput. Simulat.*, 8(1), 3–30 (1998).
22. Smirnov A. A., Bondar V. V., Rozhenko O. D., Mirzoyan M. V., Darjania A. D. Mersenne Numbers in the Bases of Systems of Residual Classes when Transmitting Data in Serial Communication Channels, *Journal of Mathematical Sciences*, 260, 241–248 (2022).
23. Kumar R., Jaiswal R. K., Mishra R. A. "Perspective and Opportunities of Modulo 2n−1 Multipliers in Residue Number System: A Review," *Journal of Circuits, Systems and Computers*, 29(11), 2030008 (2020).
24. Elaskary, R. M., Mehana, A. H., Fahmy, Y., El-Ghoneimy, M. Performance of digital predistorter in system with analogue to digital converter, *IET Communications*. https://doi.org/10.1049/cmu2.12443 (2022)
25. Suleimenov I., Bakirov A., Moldakhan I. Formalization of Ternary Logic for Application to Digital Signal Processing, *In Energy Management of Municipal Transportation Facilities and Transport Springer, Cham*, 26–35 (2019).
26. Moldovyan D. N. New form of the hidden logarithm problem and its algebraic support, *Bul. Acad., Stiin¸te Repub. Mold. Mat.* (2020).
27. Cheng T., Masuda Y., Chen J., Yu J., Hashimoto M. Logarithm-approximate floating-point multiplier is applicable to power-efficient neural network training, *Integration, Elsevier*, 74, 19–31 (2020).

scientific reports

Check for updates

OPEN

# The specifics of the Galois field *GF*(257) and its use for digital signal processing

Akhat Bakirov[1], Dinara Matrassulova[2], Yelizaveta Vitulyova[1]✉, Dina Shaltykova[2] & Ibragim Suleimenov[2]

**An algorithm of digital logarithm calculation for the Galois field *GF*(257) is proposed. It is shown that this field is coupled with one of the most important existing standards that uses a digital representation of the signal through 256 levels. It is shown that for this case it is advisable to use the specifics of quasi-Mersenne prime numbers, representable in the form $p = 2^n + 1$, which includes the number 257. For fields $GF(2^n + 1)$, an alternating encoding can be used, in which non-zero elements of the field are displayed through binary characters corresponding to the numbers +1 and −1. In such an encoding, multiplying a field element by 2 is reduced to a quasi-cyclic permutation of binary symbols (the permuted symbol changes sign). Proposed approach makes it possible to significantly simplify the design of computing devices for calculation of digital logarithm and multiplication of numbers modulo 257. A concrete scheme of a device for digital logarithm calculation in this field is presented. It is also shown that this circuit can be equipped with a universal adder modulo an arbitrary number, which makes it possible to implement any operations in the field under consideration. It is shown that proposed digital algorithm can also be used to reduce 256-valued logic operations to algebraic form. It is shown that the proposed approach is of significant interest for the development of UAV on-board computers operating as part of a group.**

**Keywords** Galois fields, Mersenne numbers, Quasi-cyclic permutations, Digital logarithm, Modulo multiplication algorithm, UAV flight computers

Binary and non-binary Galois fields are increasingly used in information technology. Mostly such algebraic structures are used to develop information security algorithms[1–3], i.e. in the area that obviously operates with alphanumeric symbols that can be put in correspondence with the elements of one or another discrete set studied by abstract algebra. Very significant results have been obtained in this area, using, among other things, Fourier-Galois transformations[4–6]. Analysis of current literature in this area, in particular, such reports as[7–9], allows us to assert that coding theory (including the theory of error-correction codes[10]) has already in many ways become a part of applied abstract algebra. For developing information security algorithms, other nontrivial algebraic structures are used, in particular, finite algebraic rings[11–13].

It is appropriate to emphasize that the needs of practice, in particular those related to the calculation of Fourier-Galois transforms[14], force one to turn to the use of non-trivial code systems[15] too.

However, the use of finite algebraic structures is of interest not only from the point of view of cryptography (more widely—information security systems). In particular, the creation of groups of unmanned aerial vehicles (UAVs) operating as a group and controlled by a single operator is currently attracting more and more attention of researchers[16,17]. Creating algorithms for controlling such groups is a non-trivial task[18–20]. Any such algorithms are based on the fact that the onboard computer of an individual UAV processes not only the commands received from the operator, but also the information received from the other UAVs in the group. All this information must be converted into executable commands to be fed to the actuating units of a particular UAV. It is significant that fuzzy logic is nowadays increasingly used to solve such a problem[21,22]. The number of variables corresponding to such logic is known to be finite, which was clearly demonstrated in[23] using the example of the rhumb rose. Moreover, as shown in the cited work, the values of fuzzy logic variables can be put in correspondence with the values of variables of multivalued logic, and such a comparison for practical purposes does not necessarily have to be mutually unambiguous. In particular, "empty" commands can be introduced into consideration. This

[1]Al-Farabi Kazakh National University, Almaty, Kazakhstan. [2]National Engineering Academy of the Republic of Kazakhstan, Almaty, Kazakhstan. ✉email: lizavita@list.ru

allows us to use the most convenient variants of multivalued logics, in particular, those of them, the set of values of variables of which can be mutually unambiguously matched to the set of elements of the Galois field $GF(p^n)$.

Moreover, any information exchanged between the elements of the UAV group, both among themselves and with the operator, is usually represented in a digital form, and one that meets existing standards. One of the most common is the 256-signal level standard. Therefore, even if we do not consider the application of fuzzy logic to the control of UAV groups, this issue can be solved in terms of multi-valued logic, and the relevant issue is the choice of logic that meets the existing standards.

This example allows us to take a somewhat broader view at the applied use of abstract algebra tools[23]. Namely, in the natural science tradition, physical phenomena are, as a rule, described by functions that take real or complex values. However, as emphasized in[24,25], this is nothing more than a matter of agreement. A function that takes values in a particular set is nothing more than a model of a real process, for example, a signal[24,25]. From a general methodological point of view, there is a mapping of a real physical process (for example, signal generation) onto a certain mathematical object, the choice of which is ultimately determined by issues of convenience and efficiency of use.

In particular, for a digital signal varying in a finite range of amplitudes, a function that takes values in Galois fields[24,25] or algebraic rings[26,27] can be used as a signal model. The number of discrete levels that fit into a finite range of amplitudes is also finite. Consequently, a function that takes values in any algebraic structure with a finite number of elements can be used for description of processes of this kind.

This fully correlates with the above statement: for a number of applications (including the development of algorithms for controlling groups of UAVs) it is acceptable to operate with elements of finite algebraic structures.

We emphasize that the elements of the Galois field can be assigned to the values of a multivalued logic variable. Moreover, as shown in[23,28], it is possible to reduce any logical operations to algebraic form. We emphasize that traditionally functions that depend on a logical variable are presented in tabular form[29,30]. The results obtained in[23,28] allow one to convert such tables to explicit algebraic expressions. Such expressions can be used, for example, as a basis for on-board UAV calculators acting as part of a group due to the finite number of executable commands and the possibility of representing the information on the basis of which they are generated, in terms of fuzzy logic.

One of the most commonly used standards divided the amplitude range into 256 levels[31]. This standard corresponds to the Galois field $GF(2^8)$, i.e. each of the signal levels can be associated with an element of this field.

Consequently, it is permissible to consider the problem of reducing calculations corresponding to such a number of discrete levels to calculations in Galois field. It is appropriate to note that there are reports devoted to the development of electronic circuits that perform modulo addition and multiplication operations in current literature, for example,[32,33]. Such operations correspond to operations in the simplest Galois fields $GF(p)$, where $p$ is a prime number. The proposed formulation of the problem fully meets this trend.

From the point of view of applied purposes, different Galois fields have different specifics[34,35], i.e. it is not always justified to consider the question of the maximum generalization of the results obtained using a specific Galois field. In particular, the field $GF(2^4+1)$, studied in[36], is conjugate to the field GF($2^4$), which makes it possible to bring into explicit form any operations for the important special case of 16-valued logic. The fact that the number 17 is one of the quasi-Mersen primes, representable in the form $p = 2^n + 1$, is used in[36].

Moreover, from the point of view of solving the problem of controlling groups of UAVs (and similar ones), it is acceptable to raise the question of creating universal calculators oriented to the use of a sufficiently large number of commands. The indicator corresponding to 256 levels, obviously overlaps the existing needs (especially if we take into account that the UAV course correction can correspond to a discrete rhumb rose[23]). Consequently, if we focus on this existing standard, it is possible in the future to raise the question of creating universal onboard calculators, allowing the use of different variants of fuzzy logic.

The number of levels equal to 256 is associated with one of the quasi-Mersenne numbers (257), which makes it possible to significantly simplify any computational procedures associated with carrying out calculations in the field $GF(2^8)$ due to transition to calculations in the conjugate Galois field $GF(257)$.

This paper presents specific algorithms that can significantly simplify any calculations in the field $GF(257)$, and also presents specific electronic circuits that prove the effectiveness of the proposed algorithms. It is important that these algorithms, among other things, make it possible to implement calculations in the field under consideration based on a standard element base corresponding to binary logic.

The basis of these algorithms is, in particular, the operation of digital logarithm. This issue is also quite actively discussed in the literature[37–39], and there are examples when this problem is considered in relation to a specific Galois field[40].

In this work, it is proved that the specificity of the $GF(257)$ field makes it possible to implement a digital logarithm algorithm, which can be used to create electronic circuits, including those that perform operations in 256-valued logic, i.e. one of the most important technical standards.

## Methods: comparison of Mersen and quasi-Mersen numbers using for digital signal processing

The method used in this report is based on the use of prime numbers, which can be called quasi-Mersen numbers.

To make the comparison adequate, let us briefly consider the basic properties of Mersenne numbers, which currently find concrete practical applications[41,42].

Such numbers can be represented in the form

$$p = 2^n - 1 \tag{1}$$

where $n$—are specifically chosen integers, the first of which are 2, 3, 5, and 7.

In relation to digital signal processing, the following property of Mersenne numbers is of interest, which is convenient to consider using the example of the field $GF(127)$ whose characteristic is the Mersenne number with $n = 7$.

In binary form, any of the elements of the $GF(127)$ field can be represented as

$$A = a_6 a_5 \ldots a_1 a_0. \tag{2}$$

where $a_i = 0,1$

Multiplying a given number by 2 modulo 127 is reduced to a cyclic permutation of symbols

$$2a_6 a_5 \ldots a_1 a_0 = a_5 \ldots a_1 a_0 a_6 \tag{3}$$

Formula (3) is a consequence of the next relation

$$1111111 \equiv 0000000(127) \tag{4}$$

Property (3), in particular, makes it possible to significantly simplify the algorithm for multiplying numbers modulo 127 by each other.

A similar algorithm was proposed for particular case of quasi-Mersen numbers in[29].

Such numbers can be represented in the form

$$p = 2^n + 1 \tag{5}$$

The algorithm[36] is based on the following properties of quasi-Mersen numbers, which can be conveniently considered using the field $GF(17)$ as an example. This field can be considered as a set of elements

$$GF(17) = \{-8, -7, \ldots, 0, \ldots 7,8\} \tag{6}$$

since the choice of representing elements is arbitrary up to the modulo comparison operation, for example, $-8 \equiv 9(17)$.

All non-zero elements of the field under consideration satisfy the relation

$$x^{16} - 1 = 0 \tag{7}$$

From this relation, in particular, it follows that any element of the field under consideration can be represented in the form

$$x = \left(\sqrt[2]{1}\right)^{s_4} \left(\sqrt[4]{1}\right)^{s_3} \left(\sqrt[8]{1}\right)^{s_2} \left(\sqrt[16]{1}\right)^{s_1} \tag{8}$$

where $s_i = 0; 1$, and the notation $\sqrt[2^k]{1}$ denotes an element having next properties:

$$\left[\sqrt[2^k]{1}\right]^{2^k} = 1; \left[\sqrt[2^k]{1}\right]^{2^{k-1}} \neq 1 \tag{9}$$

Specifically, for the field $GF(17)$ such elements are equal

$$\left(\sqrt[16]{1}\right)_1 = -1 \tag{10}$$

$$\left(\sqrt[16]{1}\right)_2 = 4; -4 \tag{11}$$

$$\left(\sqrt[16]{1}\right)_3 = 2; 8; -8; -2 \tag{12}$$

$$\left(\sqrt[16]{1}\right)_4 = 3; 5; 6; 7; -7; -6; -5; -3 \tag{13}$$

The possibility of using representation (8) follows from the general theory of Galois fields. Indeed, all powers of any primitive element $y$ of the field under consideration from 0 to 15 are different. At the same time, all these powers are roots of Eq. (5), i.e. they exhaust the elements of the field.

Let us consider the degree $y^m$, and represent the number m, where $0 \leq m \leq 15$ in binary form

$$m = m_4 m_3 m_2 m_1 \tag{14}$$

where $m_i$ are binary characters, i.e.

$$m = 2^3 \cdot m_4 + 2^2 \cdot m_3 + 2^1 \cdot m_2 + 2^0 \cdot m_1 \tag{15}$$

Therefore, the degree $y^m$ can be represented in the form

$$y^m = \left(y^8\right)^{m_4} \left(y^4\right)^{m_3} \left(y^2\right)^{m_2} \left(y^1\right)^{m_1} \tag{16}$$

This expression may be reduced to the form (8) by using relations (10)–(13).

Starting from an expression of the form (8), one can propose an alternating encoding[36], which is also convenient to consider using the example of the $GF(17)$ field.

Let us consider an expression for non-zero elements of $GF(17)$ field

$$A = 2^3 \cdot a_3 + 2^2 \cdot a_2 + 2^1 \cdot a_1 + 2^0 \cdot a_0 \tag{17}$$

where $a_i = \pm 1$

The result of calculations using formula (17) will certainly give an odd number. There are $2^4$ combinations of the form (17), with the maximum number being $A_{max} = 15$ and, accordingly, $A_{min} = -15$.

One can see, that the number of combinations of the form (17) in the case under consideration coincides with the number of non-zero elements of the field $GF(17)$. Thus, after modulo reduction, the numbers represented in the form (17) exhaust the set of non-zero elements of the field $GF(17)$. Consequently, this representation can be used along with any other, especially if we take into account that "representatives" of the residue classes of the ring of integers modulo a prime number can be chosen in an arbitrary way.

A representation of the form (17), in which $a_i = \pm 1$ has a property similar to the property possessed by Mersenne prime numbers (3). Namely, multiplying the number written in representation (15) by 2 can be represented as the following operation

$$2 \cdot A = 2^3 \cdot a_2 + 2^2 \cdot a_1 + 2^1 \cdot a_0 - 2^0 \cdot a_3 \tag{18}$$

This follows from the fact that in the field under consideration $2^4 \equiv -1(17)$.

Consequently, the operation of multiplication by 2 in alternating binary representation of a number is reduced to a cyclic rearrangement of binary elements with a change in the sign of the last element. We have

$$2 \cdot A = 2 \cdot a_2 a_1 a_0 a_3 = a_2 a_1 a_0 (-a_3) \tag{19}$$

It can be seen that this property is indeed similar to the property of the fields formed using Mersenne numbers (3).

This property gives possibility to propose simple algorithm of digital logarithm operation for very important particular case $GF(257)$ too.

## Results and discussion

### Computational invariants for elements of field GF(257)

Calculation of elements $s_j$ in representation (6) and similar ones corresponds to digital logarithm operation.

With the help of algorithms (and/or digital devices) that perform such an operation, the multiplication operation can obviously be reduced to an addition operation.

In this section it is proved that the set of non-zero elements of the field GF(257) can be divided into subsets, and this division allows one to significantly simplify digital logarithm operation. Looking ahead somewhat, we note that proposed algorithms also make it possible to significantly simplify the electronic circuits that perform this operation.

The proposed algorithm is based on quantities that can be called computational invariants. The rationale for their use is given in this section.

Let us start from the identity

$$x^{256} - 1 = \left(x^{16}\right)^{16} - 1, \tag{20}$$

The right side of relation (20) emphasizes the following fact. To represent an arbitrary non-zero element of the field GF(257) using relation (17), 8 bits are needed. Taking into account the change of sign of the last element when multiplying by 2 (16), there are 16 elements that differ from each other by the factor $2^k$, k = 0,1,…,15 in this field.

Therefore, an arbitrary non-zero element of a given field can be represented in the form

$$x = \left(\sqrt[16]{2}\right)^{\sigma} 2^k \tag{21}$$

where $\sigma = 0,1,\ldots,15$, $k = 0,1,\ldots,15$.

We emphasize that the root $\sqrt[16]{2}$ should be chosen as equal to a primitive element, i.e. the different degrees of the root must give all elements of the field $GF(257)$.

Similarly, all non-zero elements of the field $GF(17)$ can be expressed by a formula similar to (24), obtained in[36]:

$$x = \left(\sqrt{2}\right)^{\sigma} 2^k \tag{22}$$

The adequacy of representations (21) can also be proven as follows. Let us consider an arbitrary degree of a primitive element $\sqrt[16]{2}$

$$z = \left(\sqrt[16]{2}\right)^{w} \tag{23}$$

where $w = 0,1,\ldots,256$.

Let's represent the number $w$ in standard binary encoding

$$w = 2^7 w_7 + 2^6 w_6 + \cdots + 2^0 w_0 \tag{24}$$

Let's substitute expression (24) into formula (23). We have

$$z = \left(2^4\right)^{w_7} \left(2^3\right)^{w_6} \ldots \left(\sqrt[8]{2}\right)^{w_1} \left(\sqrt[16]{2}\right)^{w_0} \tag{25}$$

Note that formula (25) is equally valid for representing the field elements $GF(257)$ both in the form $-128$, $-127$ ,…,0,…, 127, 128 and in the form 0 ,…, 255, 256. These representations differ only by specific "representatives" of the corresponding classes of residues are used as $\sqrt[i]{2}$ (Tables 1 and 2).

There are exactly 16 primitive elements $\sqrt[16]{2}$ in the field $GF(257)$. They are listed in Table 1. In Table 2 only 9 elements are presented, since in the case of alternating representation the elements of $\sqrt[16]{2}$ fall into two groups that differ in sign. This is emphasized by the 9th line in Table 2.

Thus, formula (25) shows that, up to a permutation of the indicated type, the alternating binary representation allows us to reduce all elements of the field $GF(257)$ to sixteen elements of the form

$$y = \left(\sqrt[2]{2}\right)^{w_3} \left(\sqrt[4]{2}\right)^{w_2} \left(\sqrt[8]{2}\right)^{w_1} \left(\sqrt[16]{2}\right)^{w_0} \tag{26}$$

The remaining elements of the $GF(257)$ field can be obtained from these sixteen elements by cyclic permutation with a change in the sign of the symbols in the alternating binary representation.

Further, representation the elements of the field $GF(257)$ in an alternating binary encoding allows one to form function $Q_{j,j-1}$.

$$q_i = Q_{j,j-1} = \begin{cases} 1, a_j a_{j-1} = -1 \\ 0, a_j a_{j-1} = +1 \end{cases} \tag{27}$$

| $\sqrt[16]{2}$ | $\sqrt[8]{2}$ | $\sqrt[4]{2}$ | $\sqrt[2]{2}$ | $\left(\sqrt[16]{2}\right)^{16}$ |
|---|---|---|---|---|
| 27 | 215 | 222 | 197 | 2 |
| 41 | 139 | 46 | 60 | 2 |
| 54 | 89 | 211 | 60 | 2 |
| 71 | 158 | 35 | 197 | 2 |
| 82 | 42 | 222 | 197 | 2 |
| 93 | 168 | 211 | 60 | 2 |
| 108 | 99 | 35 | 197 | 2 |
| 115 | 118 | 46 | 60 | 2 |
| 142 | 118 | 46 | 60 | 2 |
| 149 | 99 | 35 | 197 | 2 |
| 164 | 168 | 211 | 60 | 2 |
| 175 | 42 | 222 | 197 | 2 |
| 186 | 158 | 35 | 197 | 2 |
| 203 | 89 | 211 | 60 | 2 |
| 216 | 139 | 46 | 60 | 2 |
| 230 | 215 | 222 | 197 | 2 |

**Table 1.** Elements $\sqrt[i]{2}$ of the field $GF(257)$ in terms of positive numbers.

| $\sqrt[16]{2}$ | $\sqrt[8]{2}$ | $\sqrt[4]{2}$ | $\sqrt[2]{2}$ | $\left(\sqrt[16]{2}\right)^{16}$ |
|---|---|---|---|---|
| 27 | −42 | −35 | −60 | 2 |
| 41 | −118 | 46 | 60 | 2 |
| 54 | 89 | −46 | 60 | 2 |
| 71 | −99 | 35 | −60 | 2 |
| 82 | 42 | −35 | −60 | 2 |
| 93 | −89 | −46 | 60 | 2 |
| 108 | 99 | 35 | −60 | 2 |
| 115 | 118 | 46 | 60 | 2 |
| −115 | 118 | 46 | 60 | 2 |

**Table 2.** Elements $\sqrt[i]{2}$ of the $GF(257)$ field in representation using both positive and negative numbers.

This function provides a count of the number of sign changes in code sequences corresponding to the alternating representation of an element of the field. In the case under consideration, this function contains 8 clock cycles.

Otherwise, we can assume that this function "takes values" at the vertices of the octagon, as shown in Fig. 1. Multiplication by powers of two in this geometric representation corresponds to the rotation of the octagon around the axis of symmetry by an angle multiple of $\frac{\pi}{4}$.

Consequently, values (27) actually correspond to well-defined geometric constructions (Fig. 1). Due to the fact that cyclic permutations with a change in sign are used, the number of non-zero values $q_i$ of function (27) can only be odd.

This geometric interpretation is illustrated by Table 3. It shows that for each specific number presented in an alternating encoding, there are certain invariants that correspond to the situations presented in Fig. 1.

This table presents examples of $q_i$ values for different elements of the field $GF(257)$ in alternating binary encoding, as well as invariants corresponding to the number of sign changes in the sequence (last column of Table 3).

It can be seen that the examples presented in this table clearly correlate with the geometric construction of Fig. 1. Namely, the number of sign changes in the sequences under consideration is odd, and, therefore, can be equal to 1, 3, 5 and 7. The last two cases (the invariants are 5 and 7) are reduced to the first two of those indicated by the inversion $0 \leftrightarrow 1$.

Consequently, only the cases $\sum q_i = 1$ and $\sum q_i = 3$ can be kept in consideration. They correspond to possible placements of one and three elements on the vertices of the octagon, as shown in Fig. 1, and such placement is specified up to rotation by an angle $\frac{\pi}{4}$, i.e. placements that differ in rotation by such an angle are considered identical.

The proposed invariants make it possible to significantly simplify the operation of digital logarithm and propose a specific electronic circuit that implements this operation based on standard logic elements. It is discussed in the next section.

## The operation of digital logarithm and the electronic circuit that implements it

As follows from the materials in the previous section, the problem of calculation of digital logarithm in the field $GF(257)$ is divided into two ones.

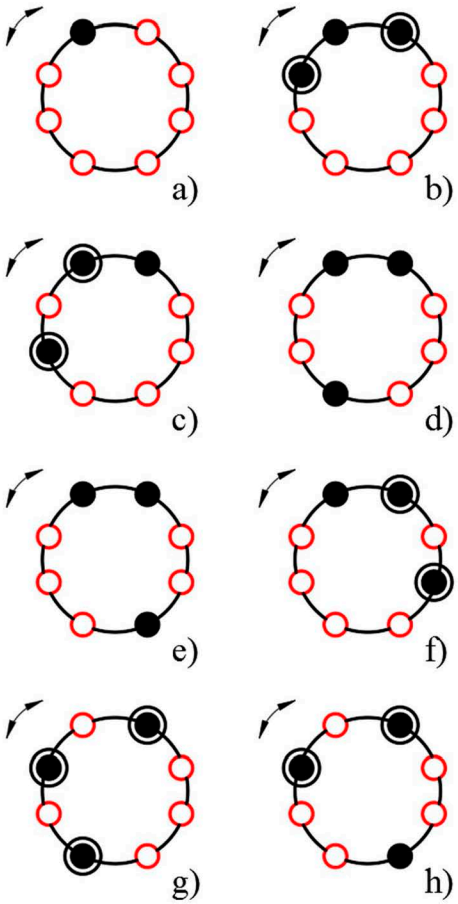


**Figure 1.** Geometric interpretation of the elements given by formula (27); black circles are vertices corresponding to values equal to 1 in formula (27), black circles with an additional black circle correspond to the case when vertices corresponding to such values are separated by one vertex.

| n | $q_7$ | $q_6$ | $q_5$ | $q_4$ | $q_3$ | $q_2$ | $q_1$ | $q_0$ | $\Sigma q_i$ |
|---|---|---|---|---|---|---|---|---|---|
| − 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| − 253 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| − 251 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 3 |
| − 249 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| − 247 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 3 |
| − 245 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 3 |
| − 243 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 3 |
| − 241 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| − 239 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 3 |
| − 237 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 3 |
| − 235 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 5 |
| − 233 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 3 |
| − 231 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 3 |
| − 229 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 3 |
| − 227 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 3 |
| − 225 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| − 223 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 3 |
| − 221 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 3 |
| − 219 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 5 |
| − 217 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 3 |
| − 215 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 5 |
| − 213 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 5 |
| − 211 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 5 |
| − 209 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 3 |
| − 207 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 3 |

**Table 3.** Invariants of sequences corresponding to alternating encoding (examples).

The first one is equivalent to determination of the rotation angle of the octagon shown in Fig. 1, which actually corresponds to the definition of a power of two (when multiplied modulo 257) in representation (25).

The second one corresponds to the identification of one of the configurations presented in Fig. 1. Having determined such a circuit configuration, one can automatically determine one of the elements represented by formula (26).

Let's consider the block diagram of a device for calculation of digital logarithm in accordance with the algorithm, which follows from the above. Let's consider second-order invariants, reflecting the relative position of units at the vertices of the octagon (Fig. 1). Such invariants are, in particular, the sums $U_{S_j}$

$$U_{S_j} = \sum q_i^{(s_j)} \tag{28}$$

of quantities $q_i^{(s_{1,2})}$ given by the formulas

$$q_i^{(s_1)} = q_i q_{i-1} \tag{29}$$

$$q_i^{(s_2)} = q_i q_{i-2} \tag{30}$$

The identification of seven cases, except for the trivial one (Fig. 1a) is of interest. In this trivial case, the digit number directly corresponds to the non-zero value $q_j$ (27).

It can be seen that the invariant $U_{S_1}$ is equal to 2 for the configuration shown in Fig. 1b only. Consequently, calculation of this invariant automatically gives identification of this case. In this case, the angle of rotation of the hexagon is fixed by function (29), which in this case takes only one non-zero value.

This sum $U_{S_1}$ also takes a non-zero value equal to 1 for the cases of Fig. 1c–f. This corresponds to the fact that there are only two nodes in the sequence located in close proximity to each other. This invariant is equal to zero for the cases of Fig. 1g and h. Consequently, the calculation of the indicated invariant allows one to identify the case corresponded to Fig. 1b and divide the remaining options into two subsets.

Calculation of the invariant $U_{S_2}$ automatically selects two cases from the considered set (Fig. 1d and e). In these diagrams, there are no nodes separated from each other by only one empty vertex. Additionally, the calculation of this invariant uniquely identifies the case of Fig. 1g, for which this invariant is equal to 2. For clarity, filled vertices, separated from each other by two turns at an angle of π/4, are highlighted with additional circles.

Thus, to complete the digital logarithm operation for the field under consideration, all that remains is to ensure the difference between the cases of Fig. 1c and f, as well as between the cases of Fig. 1d and e. This

difference is identified by the phase shift between functions (29) and (30), which does not depend on the presence of a factor equal to a power of 2 (which is equivalent to rotations of the octagons under consideration).

Specifically, this phase shift can be determined, for example, through the calculation of invariants built on the functions

$$q_i^{(s_3)} = q_{i+2}q_iq_{i-1} \tag{31}$$

$$q_i^{(s_4)} = q_{i+3}q_iq_{i-1} \tag{32}$$

To use such functions, it is already important to take into account the direction; specifically, it is assumed that the direction in Fig. 1 is counted clockwise.

It can be seen that function (31) can take a non-zero value only for the case of Fig. 1c, and function (32) is only for the case of Fig. 1d. Therefore, to identify the above cases, it is enough to count the invariants $\sum q_i^{(s_3)}$ and $\sum q_i^{(s_4)}$.

As a result, we can propose the following block diagram of device for digital logarithm (Fig. 2).

A device built on this circuit works as follows.

The original binary signals (their number is 16) are sent to the converter (1), which converts them to the familiar encoding. Due to this, in particular, the set of signals generated at the output (1) can be considered cyclic.

From all 16 outputs of the converter, signals are fed to the adder (2), which counts the number of units. A logical one is formed at the output $A_1$ of the adder (2) if the number of ones is 1, i.e. the case corresponding to Fig. 1a is realized. In this case, the digital logarithm of the number is exactly equal to the number of the converter output (1) on which a signal other than 0 is generated.

A logical one is formed at the output $A_2$ of the adder (2) if the number of ones is 3, i.e. the cases corresponding to Fig. 1c-h are realized. If the number of ones is 5, then a logical one is formed at output $A_3$, and if it is 7, at output $A_4$.

Signals from outputs $A_3$ and $A_4$ are fed to the OR element (4). This signal is used to control the inverters ($5_i$), which perform the inversion operation $0 \leftrightarrow 1$, provided that the number of ones is 5 or 7. The inverters ($5_i$) perform the addition operation modulo 2, i.e. they are EXCLUSIVE OR elements. The second inputs of the inverters ($5_i$) are supplied with signals from the outputs of the converter (1).

As a result, a set of signals is formed at the outputs of the inverters ($5_i$), which contain either one non-zero signal or three such signals.

This set of signals is supplied to the first stage of the first digital logarithm block, consisting of elements ($6_i$) and ($7_i$), as well as an OR logic element (8).

Elements ($6_i$) perform a logical OR operation. One of the inputs of each of these elements is supplied with a signal from the output of the element (8), and the second is supplied with a signal from the output of the inverter ($5_{i-1}$). Consequently, the signal at the output of each of the elements ($6_i$) will be equal to 1 if a logical unit is formed at the output $A_1$ or output $A_4$ of the adder (2), and equal to the value of the $(i-1)$ st signal in the opposite case.

The signal from the output of each of the elements ($6_i$)) is fed to one of the inputs of the element ($7_i$), which performs a logical AND operation. The second input of each of these elements is fed directly from the output of the inverter ($5_i$).

As a result, the block under consideration calculates correspondent invariant if the number of units in the set of signals at the output of the converter (1) was equal to 3 or 5 and leaves the signals generated by the set of inverters ($5_i$) unchanged if the specified number was 1 or 7.

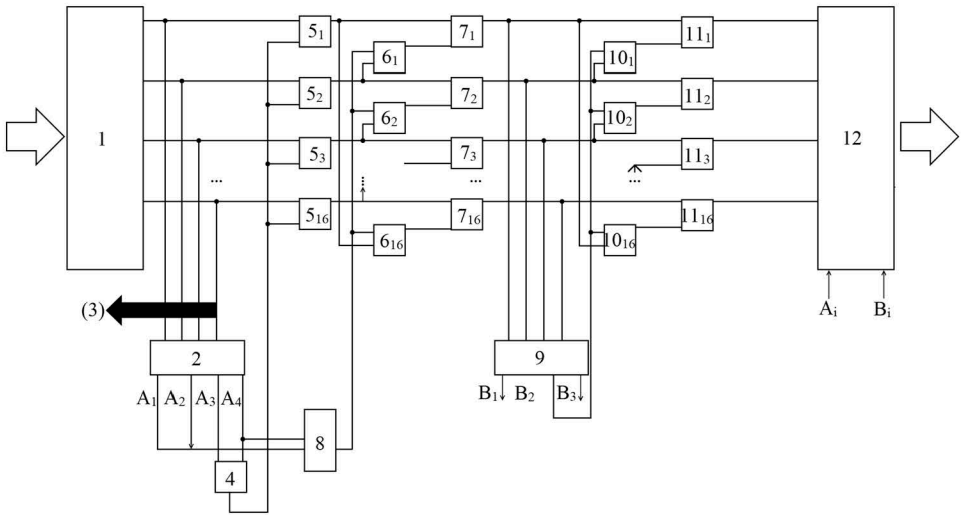


**Figure 2.** Block diagram of a device for digital logarithm in the field $GF(257)$.

As follows from Fig. 1, this cascade already identifies sequences corresponding to Fig. 1a, as well as Fig. 1b–f in the sense that for these cases, in the set of signals generated at the outputs of elements ($7_i$), only one logical unit is formed, which uniquely identifies the digital logarithm of the corresponding element of the Galois field.

To identify the remaining cases, an adder (9) is used, which counts the number of ones at the outputs of the elements ($7_i$). If this number is 1, then the digital logarithm operation should be considered complete. If this number is 0, then the situations corresponding to Fig. 1g or Fig. 1h. In this case, the result of digital logarithm is generated by the second digital logarithm block, which is discussed below. Identification signals are generated at the outputs $B_1$, $B_2$ and $B_3$ of the adder (9). A logical unit is formed at these outputs if the sum of logical units at the outputs of elements ($7_i$) is equal to 0, 1 and 2, respectively.

To take into account the situation corresponding to Fig. 1b, the second stage of the first digital logarithm block is used, which is designed similarly to the first. It consists of elements ($10_i$) and ($11_i$). Elements ($10_i$) perform a logical OR operation, and elements ($11_i$) perform an AND operation. The first inputs of each of the elements ($10_i$) are supplied with signals from the outputs of elements ($7_{i-1}$), and their second inputs are supplied with a signal from output $B_2$, i.e. in the case when there is a logical unit at this output, then elements ($10_i$) do not affect the operation of the system.

If a logical zero is formed at the specified input, then the second stage de facto cyclically performs the operation

$$q_i^{(5)} = q_{i+1}q_iq_{i-1} \tag{33}$$

This operation in the case corresponding to Fig. 1b, leads to the appearance of a logical unit at only one of the outputs of the elements ($11_i$), the outputs of which are connected to the inputs of the decision device (12).

Thus, the considered part of the circuit provides unambiguous identification of the digital logarithm for cases corresponding to Fig. 1a–f.

Cases Fig. 1g,h correspond to a logical unit generated at the output $B_1$ of the adder 9. This signal blocks the operation of the first digital differentiation block and transmits it to the second, arranged in a similar way, with the difference that identification is carried out through the use of functions (29) and (30).

Thus, we have shown that the use of an alternating binary representation for elements of the $GF(257)$ field allow one to realize all the same advantages that occur when Galois fields corresponding to prime Mersenne numbers are used.

The most important type of field of this type is the $GF(257)$, since it corresponds to the number of levels of the digitized signal, which is often used in practice. For example, the most commonly used analog-to-digital converters involve the use of exactly 256 levels.

The proposed algorithm allows one to reduce the multiplication operation to the addition operation.

As shown, in particular, in[23], any operations in the Galois field, given, for example, by a truth table, can be reduced to the operations of multiplication and addition. The electronic circuit providing the multiplication operation has been presented above. The scheme of a universal adder on the modulus of an arbitrary integer was presented in our work[43] on the basis of modernization of the adder scheme, for which we received a patent of Kazakhstan[44]. Thus, the scheme presented above allows to realize, for example, on-board calculators for UAVs operating as part of a group, as well as to solve similar problems.

## Probable generalization and applications of proposed approach

Let us consider the possibilities for generalizing the proposed approach, although at this stage of research this issue is rather of academic interest. In particular, the next quasi-Mersenne number after 257 is the number $2^{16}+1=65{,}537$.

For any Galois field $GF(2^n+1)$ we have

$$2^{n-1} \equiv -1\left(2^n+1\right) \tag{34}$$

This follows from the fact that

$$2^n+1 \equiv 0\left(2^n+1\right) \tag{35}$$

Further, all non-zero elements of the field $GF(2^n+1)$ are roots of the equation

$$x^{2^n}-1=0, \tag{36}$$

which directly follows from the theory of finite algebraic fields: the number of non-zero elements of the field is one less than their total number, therefore all elements of the field satisfy an equation of degree $2^n$.

In accordance with the method used above, it is convenient to represent this equation as follows

$$x^{2^n}-1=\left(x^{\frac{2^n}{2n}}\right)^{2n}-1=0 \tag{37}$$

where $N$ is the number of bits in the binary alternating representation of the field $GF(2^n+1)$.

Formula (37) takes into account the following fact. $n$ bits are required for representation of non-zero elements of the field under consideration in alternating encoding,

$$N=\log_2\left(2^n\right)=n \tag{38}$$

Therefore, there are $2n$ field elements, differing from each other by multiplication by a power of 2 in representation under consideration: in order for a quasi-cyclic permutation to lead to the original result, the field element must be multiplied by $2^{2n}$ modulo $2^n + 1$. Consequently, the set of non-zero elements of the field under consideration is divided into 2n subsets, each of which contains $\frac{2^n}{2n}$ elements.

The values of these quantities for the first 4 quasi-Mersenne numbers are presented in Table 4.

It can be seen that for Galois fields corresponding to the first three quasi-Mersenne numbers, it certainly makes sense to identify subsets whose elements differ by multiplication by a power of 2. Already for the fourth quasi-Mersenne number, the advisability of using this approach is, at a minimum, not obvious. Elements similar to those shown in Fig. 1 becomes not 8, but 1024.

In general, Table 4 shows that the proposed approach is indeed appropriate to apply to specific Galois fields $GF(17)$ and $GF(257)$, which are of practical interest. Taking into account the results obtained in[36], the field $GF(5)$ may also be of interest as conjugate (in the sense of digital logarithm) with the field $GF\left(2^2\right)$ to simplify the operations of four-valued logic.

Let us show that the results obtained are indeed of interest from the point of view of combining methods of digital signal processing and multi-valued logic for the case when the signal is reduced to 256 discrete levels.

As shown in[23,28], to reduce the operations of multivalued logic to algebraic ones, it is advisable to use algebraic analogue of the δ-function

$$\delta_i(x) = 1 - (x - x_i)^{p^n - 1} \tag{39}$$

where $x$ is the current variable that takes values in the Galois field $GF\left(p^n\right)$, $x_i$ is the $i$-th element of the field in question.

This function has the following property

$$\delta_i(x) = \begin{cases} 1, & x = x_i \\ 0, & x \neq x_i \end{cases} \tag{40}$$

This property follows from the general theory of Galois fields, according to which, for an arbitrary element of a Galois field containing $p^n$ elements $x^{p^n - 1} = 1$.

The use of algebraic analogue of the $\delta$-function allows, in particular, to reduce any operations of $p^n$-logic (logic, the number of values of a variable is equal to $p^n$) to an explicit algebraic form[23,28]. In relation to the logical function of two variables and the Galois fields $GF(p)$, the corresponding expression has the form

$$F(x,y) = \sum_{i,j=0}^{i,j=p-1} f\left(x_i, y_j\right) \delta_i(x) \delta_j(y) \tag{41}$$

where the values $f(x_i, y_j)$ form a truth table (recall that the operations of multi-valued logic are currently specified through truth tables[29]).

Relation (41) clearly expresses the well-known fact: if the number of values of variables of multivalued logic is equal to an integer power of a prime number, then logical operations can be reduced to calculations in the conjugate Galois field.

If this condition is not met, then it is advisable to modify the algebraic analogue of the δ-function using the digital logarithm operation[36]. Moreover, it is also advisable to use this approach for the case of the field $GF\left(p^n + 1\right)$.

Specifically, we can compose an algebraic analogue of the δ-function in the following form

$$\delta_{n_i}(n) = 1 - \left(\theta^n - \theta^{n_i}\right)^{p^n} \tag{42}$$

In this formula, $n$ and $n_i$ denote integers that correspond to the elements of the field $GF\left(p^n\right)$. It is assumed that the values of the function itself $\delta_{n_i}(n)$ belong to the field $GF\left(p^n + 1\right)$, on which the mapping is performed; $\theta$ —is a primitive element of the field $GF\left(p^n + 1\right)$, i.e. such an element which degrees are exhausting all non-zero elements of a given field.

This formula is convenient in that it allows you to get an expression for any operation carried out in terms of 256-valued logic to algebraic ones.

Indeed, using (42), we have

$$Q(n,m) = \sum_{i,j=0}^{i,j=p-2} Q_{i,j} \delta_{n_i}\left(\theta^n\right) \delta_{m_j}\left(\theta^m\right) \tag{43}$$

| $n$ | 2 | 4 | 8 | 16 |
|---|---|---|---|---|
| $2^n + 1$ | 5 | 17 | 257 | 65,537 |
| $2n$ | 4 | 8 | 16 | 32 |
| $2^n/2n$ | 1 | 2 | 16 | 2048 |

**Table 4**. Values of quantities for the first 4 quasi-Mersenne numbers.

where the quantities $Q_{i,j}$ re associated with the elements of the truth table $p^n$-valued logic in the following way

$$Q_{i,j} = \theta^{n_{i,j}} \tag{44}$$

where $n_{i,j}$ is the number corresponding to the element of the truth table with numbers $i, j$.

Formula (43) can be considered as a function of a pair of elements of the field $GF\left(p^n\right)$, corresponding to the numbers $n$ and $m$, and taking a value in the field $GF\left(p^n+1\right)$. Formula (43) admits an obvious generalization to a function of an arbitrary number of elements from the field $GF\left(p^n\right)$.

When substituting two arbitrary $n_0$ and $m_0$ into formula (43), due to (42), we have

$$Q(n_0, m_0) = Q_{n_0,m_0} \tag{45}$$

Only one term in the formula (43) is non-zero.

It can be seen that in order to go back to the elements of the field $GF\left(p^n\right)$ when using formula (43), namely the digital logarithm operation is required, which is proposed in this work for the important special case of $GF\left(2^8+1\right)$.

Therefore, in the future it is possible to develop systems that directly operate in 256-valued logic.

## Algorithms for controlling groups of unmanned vehicles as an area of application of the obtained results

It was noted above that one of the practical applications of calculators operating with Galois fields of relatively small size is the development of algorithms for controlling groups of UAVs, which, we emphasize again, are currently attracting increasing interest of researchers[16,17]. In this section, we will try to demonstrate that for this purpose, the digital differentiation operation realized thanks to the developed approach is essential.

The problem of group control of robots for various purposes has been considered in the literature for a very long time[16,45,46]. This includes vehicles moving in a 3-D environment[17,47]. Various methods are used for its solution, in particular, those based on self-organization (Self-adaptive collective motion) of UAV groups[48], on machine learning[49], on the use of graph theory[50]. There are known works that consider a modernized Olfati-Saber algorithm using a virtual leader who is tracked by all UAVs forming a group[51]. In[52], algorithms built using artificial intelligence combined with IoS have been proposed to control a swarm of UAVs. On this basis, a self-organizing ZigBee network is simulated in the cited work.

However, the decentralized robot control algorithm, which only takes into account information about the positions of other system elements but not about the directions of their movement, have significant limitations[52]. This difficulty is partially overcome in[20].

The control algorithms for a system of multiple UAVs considered in[53] also focus on distributed control centered on the so-called leader–follower consensus, which ensures that the entire swarm as a system whole moves according to a predetermined trajectory. In[54], where the drone swarm is considered from the perspective of Networked Control Systems (NCS), the role of on-board computing systems for controlling the UAV swarm as a systemic whole is emphasized. In[55], the problem of interfacing an artificial neural network with a UAV swarm was solved, which, among other things, provides for maintaining a given distance between the elements of the swarm, as well as to maintain the formation of the group.

Thus, the solution of the problem of controlling the UAV swarm as an integral system, as follows from the above, is closely related to the solution of the problem of information processing by onboard computing systems.

It can also be seen that this problem can be solved by a variety of means. However, there is an essential nuance, which, in particular, is demonstrated by the results of[54,55]. On the one hand, the amount of information received by the individual elements of the UAV swarm should not be excessively large. On the other hand, it should be sufficient, for example, to allow a particular element of the group to take an adequate position in the swarm (especially when the swarm is ordered).

This returns to the issue of using fuzzy logic to control groups of UAVs, which was considered in particular in[21,22].

In[21], an algorithm based on fuzzy logic is proposed that can control a swarm of robots in order to maintain a leader–follower formation without collisions with other agents in the swarm. Simulations have shown that the swarm moves as a unit following the leader. In[22], algorithms based on fuzzy logic were used to solve the problem of fault tolerance of a group of several autonomous UAVs when they form a formation in the shape of a certain geometric figure. The proposed approach based on fuzzy logic allows on-board control units of each UAV to make their own decision in a decentralized manner. Such decisions, including the possibility of changing the configuration of the whole group.

Consequently, it is reasonable to raise the question of creating computational means for use in on-board computing units of UAVs, which will be purposefully designed to perform operations of odd logic.

As noted above, such operations can be reduced to multi-valued logic operations as demonstrated, for example, in[23]. Further, it is possible to lead these operations to computations in Galois fields or finite algebraic rings[28]. It is this fact that determines the significance of using the operation of digital logarithmization, which can be realized by quite simple means using the proposed approach.

We will show that when we pass to the operations of multivalued or fuzzy logic, the use of Galois fields has very significant advantages compared to the situation when the operations of multivalued or fuzzy logic are represented in tabulated form.

As emphasized, in particular, in[56], the development of methods for controlling groups of UAVs is inextricably linked to the problems of providing secure communication channels. The main methods of such protection are

based on the use of cryptography, but it is also relevant to provide protection at the physical level, for which various approaches can be used, a review of which is given in the cited work. These include, for example,[57,58].

Among them, one of the methods of physical protection of UAV onboard computers from third-party information influences is the implementation of appropriate algorithms not at the level of programs executed by the onboard computer, but at the level of electronic circuits. Any program remains unguaranteed from third-party interference. On the contrary, if an algorithm is realized at the level of electronic circuitry, it is much more difficult to transform it due to third-party informational influences.

It is this circumstance that determines the advantages of the approach based on the use of explicit algebraic expressions expressing the operations of multivalued logic over their representation in tabular form. Indeed, the use of the tabular form obviously implies the installation of this or that program on the on-board computer. On the contrary, as it was shown in[23,28] on concrete examples, the representation of operations of multivalued logic in algebraic form allows to realize electronic circuits performing the corresponding computations without using software.

Further, among multivalued logics, as it has been clearly shown, in particular, in[23,28], a special place is occupied by logics complementary to Galois fields $GF(p^n)$. In this case, the reduction of operations of multivalued logics to algebraic form turns out to be the simplest. In particular, the algebraic expression to which any operation of such a multivalued logic is reduced contains only the operations of multiplication and addition. Even in the case when the number of values of a multivalued logic variable is only one less than a prime number (or its degree), we have to introduce additional algebraic operations into consideration (the operation of digital logarithmization and its inverse[28]). However, for the purposes of building algorithms for controlling groups of UAVs, this difficulty is not fundamental, since it is always possible to introduce "empty" commands, supplementing the number of commands to a convenient value $p^n$.

Consequently, the effectiveness of any UAV group control algorithms based on the use of fuzzy or multivalued logic can be evaluated on the basis of their compliance with formulas similar to formula (43), which leads the operations of multivalued logic of the considered type to an algebraic expression (and further—to the realization in the form of a specific electronic circuit).

We emphasize that any algorithm of the considered type can be regarded as a special case of the above formulas, since any operation of multivalued logic of the considered type is reduced to an expression of this type.

One cannot but see that from the point of view of realization in the form of an electronic circuit the most resource-intensive operation is the operation of raising in degree, on which the algebraic delta-function (42) is built.

The corresponding calculations can be simplified by using the digital logarithm operation, which is performed by the circuit shown in Fig. 2. In this case, the operation of increasing in degree is reduced to a multiplication operation. Moreover, due to the specificity of the considered field $GF(257)$ the digital logarithm operation maps non-zero elements of this field to elements of the field $GF(2^8)$, computations in which are realizable on the basis of standard elements corresponding to binary logic.

There is every reason to believe that control algorithms of UAV groups in the foreseeable future will be oriented to some standards similar to those currently developed in the field of digital signal processing, television, etc. Proceeding from the fact that such algorithms, in the end, conveniently lead to calculations in Galois fields (and their realization through electronic circuits), it is acceptable to assume that the expected standard will be associated with a specific Galois field.

Taking into account that the most resource-intensive is the operation of raising to degree, it seems reasonable to focus on those Galois fields which, on the one hand, have enough elements to cover the needs of practice, and, on the other hand, allow to realize the operation of digital logarithmization by the simplest means possible.

It is this criterion that the field $GF(257)$ considered in this paper satisfies, which is complementary (from the point of view of performing the operation of digital logarithmization) to the field $GF(2^8)$, the computations in which can be realized on the basis of standard elements.

It is also appropriate to note that the circuit providing digital logarithmization is also realizable on the basis of type elements corresponding to binary logic. This, among other things, means that to realize the proposed approach in practice it is possible to use microcircuits with programmable logic structure, which are being actively developed at present[59]. Let us also note that there is a possibility to realize an adder on the modulus of an integer with adjustable modulus value, which is also built on typical logic elements[43]. Thus, there is a possibility for realization of on-board UAV calculators, completely based on calculations in Galois fields.

Note also that electronic circuits providing computations modulo integer (adders and multipliers) have been actively developed recently[60,61]. Such devices, obviously, can be used also for computations in Galois fields. Among others, there are known constructions of modulo $2^n + 1$ calculators that satisfy fields of the considered type[62,63]. Investigations in the field of modular adders and multipliers are also reflected in the patent literature[64,65]. The disadvantage of existing modulo adder schemes, however, remains their complexity. For example, the adder scheme presented in cited reports can be replaced by a substantially simpler one[44]. A similar conclusion is true for the schemes proposed in[62–66]. The operation of digital logarithmization, based on the algorithm proposed in this paper and realized in the form of the scheme of Fig. 2, allows to pass from the multiplication operation to the addition operation, and the latter is performed on the basis of schemes corresponding to the usual binary logic, as it was shown above.

## Conclusion

Thus, the use of finite algebraic structures is of interest not only for the purposes of cryptography, where the use of algebraic fields or algebraic rings of large size is required. Of no less interest are problems in which the number of elements of algebraic structures remains relatively small, which, in particular, is demonstrated by the example of algorithms of UAV flight computers operating as part of a group.

The fact, as well as results obtained once again show that for applied using it is extremely important to take into account the specifics of concrete Galois fields. In particular, this applies to the field $GF(257)$, which corresponds to one of the quasi-Mersenne primes, i.e. numbers that can be represented in the form $p = 2^n + 1$.

This field is associated with the number 256, which corresponds to one of the most important standards used in modern digital technologies.

For numbers corresponding to fields $GF(2^n + 1)$, it is convenient to use an alternating encoding, in which multiplication by the number 2 modulo $p = 2^n + 1$ corresponds to a quasi-cyclic permutation of binary symbols, i.e. cyclic permutation with a change in the sign of the permuted element.

This encoding allows one to implement a simple digital logarithm algorithm, which allows one to reduce the multiplication operation to the addition operation, etc.

Specifically, for the $GF(257)$ field, the digital logarithm operation is simplified due to the fact that the set of non-zero elements of this field can be divided into 16 subsets, the elements of which differ from each other by a quasi-cyclic permutation of binary symbols. As a result, the operation of digital logarithm for a given field leads to the identification of an element by belonging to one of these subsets.

It is important that the operation of digital logarithm in the field under consideration, which leads the multiplication operation to the addition operation, can also be implemented using relatively simple electronic circuits. A corresponding example is presented in this work. This scheme, along with the scheme of adder modulo integer with adjustable modulus value, proposed in[43], allows to realize any operations in the field GF(257), for example, set through the truth table. In the future, this approach can be the basis, for example, for on-board UAV calculators acting as part of a group. Operations in Galois fields of relatively small size are also of interest in the future for the development of new artificial intelligence systems, approaching the biological prototype by the principle of operation, the functioning of which cannot be reduced to binary logic.

## Data availability

All data generated or analyzed during this study are included in this published article.

Received: 13 November 2023; Accepted: 1 July 2024
Published online: 04 July 2024

## References

1. Lehnigk-Emden, T. & Wehn, N. Complexity evaluation of non-binary Galois field LDPC code DECODERS. In *2010 6th International Symposium on Turbo Codes & Iterative Information Processing* 53–57. https://doi.org/10.1109/ISTC.2010.5613874 (2010).
2. Pruss, T., Kalla, P. & Enescu, F. Equivalence verification of large Galois field arithmetic circuits using word-level abstraction via Gröbner bases. In *Proceedings of the 51st Annual Design Automation Conference* 1–6. https://doi.org/10.1145/2593069.2593134 (2014).
3. Jagadeesh, H., Joshi, R. & Rao, M. Group Secret-key generation using algebraic rings in wireless networks. *IEEE Trans. Veh. Technol.* **70**(2), 1538–1553. https://doi.org/10.1109/TVT.2021.3054031 (2021).
4. Liu, P., Pan, Z. & Lei, J. Parameter identification of reed-Solomon codes based on probability statistics and Galois field Fourier transform. *IEEE Access* **7**, 33619–33630. https://doi.org/10.1109/ACCESS.2019.2904718 (2019).
5. Girisankar, S. B., Nasseri, M., Priscilla, J., Lin, S. & Akella, V. Multiplier-free implementation of Galois field Fourier transform on a FPGA. *IEEE Trans. Circuits Syst. II Express Briefs* **66**(11), 1815–1819. https://doi.org/10.1109/TCSII.2019.2894361 (2019).
6. Huang, Q., Tang, L., He, S., Xiong, Z. & Wang, Z. Low-complexity encoding of quasi-cyclic codes based on Galois Fourier transform. *IEEE Trans. Commun.* **62**(6), 1757–1767. https://doi.org/10.1109/TCOMM.2014.2316174 (2014).
7. Zhang, A. & Feng, K. A unified approach to construct MDS self-dual codes via Reed-Solomon codes. *IEEE Trans Inf. Theory* **66**(6), 3650–3656. https://doi.org/10.1109/TIT.2020.2963975 (2020).
8. Nardo, L. G. *et al.* A reliable chaos-based cryptography using Galois field. *Chaos Interdiscip. J. Nonlinear Sci.* **31**(9), 091101. https://doi.org/10.1063/5.0061639 (2021).
9. Hazzazi, M. M., Attuluri, S., Bassfar, Z. & Joshi, K. A novel Cipher-Based data encryption with Galois field theory. *Sensors* **23**(6), 3287. https://doi.org/10.3390/s23063287 (2023).
10. Kuo, Y.-M., Garcia-Herrero, F., Ruano, O. & Maestro, J. A. RISC-V Galois field ISA extension for non-binary error-correction codes and classical and post-quantum cryptography. *IEEE Trans. Comput.* **72**(3), 682–692. https://doi.org/10.1109/TC.2022.3174587 (2022).
11. Bagheri, K. & Sadeghi, M. R. A new non-associative cryptosystem based on NTOW public key cryptosystem and octonions algebra. *ACM Commun. Comput. Algebra* **49**(1), 13 (2015).
12. Markov, V. T., Mikhalev, A. V. & Nechaev, A. A. Nonassociative algebraic structures in cryptography and coding. *J. Math. Sci.* **245**(2), 178–196. https://doi.org/10.1007/s10958-020-04685-5 (2020).
13. Markov, V. T., Mikhalev, A. V. & Kislitsyn, E. S. Non-associative structures in homomorphic encryption. *J. Math. Sci.* **262**(5), 735–739. https://doi.org/10.1007/s10958-022-05850-8 (2022).
14. Deshmukh, T. P. & Dewalkar, V. P. The design approach for fast computation of Fourier transform over a finite field. In *2014 International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE)* 1–4. https://doi.org/10.1109/ICGCCEE.2014.6922465 (2014).
15. Chernov, V. M. Calculation of Fourier-Galois transforms in reduced binary number systems. *Comput. Opt.* **42**(3), 495–500. https://doi.org/10.18287/2412-6179-2018-42-3-495-500 (2018).
16. Dorigo, M., Theraulaz, G. & Trianni, V. Swarm robotics: Past, present, and future [Point of View]. *Proc. IEEE* **109**(7), 1152–1165. https://doi.org/10.1109/JPROC.2021.3072740 (2021).
17. Chung, S.-J., Paranjape, A. A., Dames, P., Shen, S. & Kumar, V. A survey on aerial swarm robotics. *IEEE Trans. Robot.* **34**(4), 837–855. https://doi.org/10.1109/TRO.2018.2857475 (2018).
18. Liu, W. & Gao, Z. A distributed flocking control strategy for UAV groups. *Comput. Commun.* **153**, 95–101. https://doi.org/10.1016/j.comcom.2020.01.076 (2020).

19. Wang, X., Chen, G., Gong, H. & Jiang, J. UAV swarm autonomous control based on internet of things and artificial intelligence algorithms. *IFS* **40**(4), 7121–7133. https://doi.org/10.3233/JIFS-189541 (2021).
20. Zheng, Y., Huepe, C. & Han, Z. Experimental capabilities and limitations of a position-based control algorithm for swarm robotics. *Adapt. Behav.* **30**(1), 19–35. https://doi.org/10.1177/1059712320930418 (2022).
21. Quesada, W. O., Rodriguez, J. I., Murillo, J. C., Cardona, G. A., Yanguas-Rojas, D., Jaimes, L. G. & Calderón, J. M. Leader-follower formation for UAV robot swarm based on fuzzy logic theory. In *Artificial Intelligence and Soft Computing*, Vol. 10842, 740–751. https://doi.org/10.1007/978-3-319-91262-2_65 (2018).
22. Hafez, A. T. & Kamel, M. A. Fault-Tolerant control for cooperative unmanned aerial vehicles formation via fuzzy logic. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)* 1261–1266. https://doi.org/10.1109/ICUAS.2016.7502660 (2016).
23. Suleimenov, I. E., Vitulyova, Y. S., Kabdushev, S. B. & Bakirov, A. S. Improving the efficiency of using multivalued logic tools. *Sci. Rep.* **13**(1), 1108. https://doi.org/10.1038/s41598-023-28272-1 (2023).
24. Moldakhan, I., Matrassulova, D., Shaltykova, D. & Suleimenov, I. Some advantages of non-binary Galois fields for digital signal processing. *Indones. J. Electr. Eng. Comput. Sci.* **23**, 871–878. https://doi.org/10.11591/ijeecs.v23.i2.pp871-878 (2021).
25. Vitulyova, E. S., Matrassulova, D. K. & Suleimenov, I. E. New application of non-binary galois fields fourier transform: Digital analog of convolution theorem. *IJEECS* **23**(3), 1718. https://doi.org/10.11591/ijeecs.v23.i3.pp1718-1726 (2021).
26. Matrassulova, D. K., Vitulyova, Y. S., Konshin, S. V. & Suleimenov, I. E. Algebraic fields and rings as a digital signal processing tool. *IJEECS* **29**(1), 206–216. https://doi.org/10.11591/ijeecs.v29.i1.pp206-216 (2022).
27. Suleimenov, I. E., Bakirov, A. S. & Vitulyova, Y. S. Prospects for the use of algebraic rings to describe the operation of convolutional neural networks. In *2022 The 6th International Conference on Advances in Artificial Intelligence* 1–7. https://doi.org/10.1145/3571560.3571561 (2022).
28. Suleimenov, I. E., Vitulyova, Y. S., Kabdushev, S. B. & Bakirov, A. S. Improving the efficiency of using multivalued logic tools: Application of algebraic rings. *Sci. Reports* **13**(1), 22021. https://doi.org/10.1038/s41598-023-49593-1 (2023).
29. Marcos, J. On a Problem of da Costa. In *Essays on the Foundations of Mathematics and Logic* Vol. 2 (ed. Sica, G.) 53–69 (Polimetrica, Monza, 2005).
30. Ciuciura, J. A note on Fernández–Coniglio's hierarchy of paraconsistent systems. *Axioms* **9**(2), 35. https://doi.org/10.3390/axioms9020035 (2020).
31. Díaz De Aguilar, J. *et al.* Characterization of an Analog-to-digital converter frequency response by a Josephson arbitrary waveform synthesizer. *Meas. Sci. Technol.* **30**(3), 035006. https://doi.org/10.1088/1361-6501/aafb27 (2019).
32. Kalimoldayev, M., Tynymbayev, S., Gnatyuk, S., Ibraimov, M. & Magzom, M. The device for multiplying polynomials modulo an irreducible polynomial. *News Natl. Acad. Sci. Repub. Kazakhstan Ser. Geol. Tech. Sci.* **2**(434), 199–205. https://doi.org/10.32014/2019.2518-170X.55 (2019).
33. Dey, S. & Ghosh, R. *A Review of Cryptographic Properties of S-Boxes with Generation and Analysis of Crypto Secure S-Boxes.*; preprint; PeerJ Preprints. https://doi.org/10.7287/peerj.preprints.26452v1 (2018).
34. Vitulyova, Y. S., Bakirov, A. S. & Suleimenov, I. E. Galois fields for digital image and signal processing: Evidence for the importance of field specificity. In *2022 5th International Conference on Pattern Recognition and Artificial Intelligence (PRAI)* 637–642. https://doi.org/10.1109/PRAI55851.2022.9904074 (2022).
35. Matrassulova, D. K., Kabdushev, S. B., Bakirov, A. S. & Suleimenov, I. E. Algorithm for analyzing rotating images based on the Fourier–Galois transform. In *2023 15th International Conference on Computer Research and Development (ICCRD)* 204–209. https://doi.org/10.1109/ICCRD56364.2023.10080084 (2023).
36. Suleimenov, I. E., Vitulyova, Y. S. & Matrassulova, D. K. Features of digital signal processing algorithms using Galois fields GF($2^{n+1}$). *PLoS ONE* **18**(10), e0293294. https://doi.org/10.1371/journal.pone.0293294 (2023).
37. Joux, A., Odlyzko, A. & Pierrot, C. The past, evolving present, and future of the discrete logarithm. In *Open Problems in Mathematics and Computational Science* 5–36. https://doi.org/10.1007/978-3-319-10683-0_2 (2014).
38. Vishnoi, S. & Shrivastava, V. A new digital signature algorithm based on factorization and discrete logarithm problem. *Int. J. Comput. Trends Technol.* **3**(4), 653–657 (2012).
39. Fried, J., Gaudry, P., Heninger, N. & Thomé, E. A kilobit hidden SNFS discrete logarithm computation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques* 202–231 (2017).
40. Barbulescu, R., Bouvier, C., Detrey, J., Gaudry, P., Jeljeli, H., Thomé, E. *et al.* Discrete logarithm in GF ($2^{809}$) with FFS. In *Public-Key Cryptography–PKC 2014: 17th International Conference on Practice and Theory in Public-Key Cryptography*, Vol. 17 221–238. https://doi.org/10.1007/978-3-642-54631-0_13. (2014).
41. Matsumoto, M. & Nishimura, T. Mersenne Twister: A 623-dimensionally equidistributed uniform pseu-dorandom number generator. *ACM Trans. Model. Comput. Simulat.* **8**(1), 3–30 (1998).
42. Smirnov, A. A., Bondar, V. V., Rozhenko, O. D., Mirzoyan, M. V. & Darjania, A. D. Mersenne numbers in the bases of systems of residual classes when transmitting data in serial communication channels. *J. Math. Sci.* **260**(2), 241–248. https://doi.org/10.1007/s10958-022-05688-0 (2022).
43. Suleimenov, I. E., Vitulyova, Y. S., Shaltykova, D. B., Matrassulova, D. K. & Bakirov, A. S. Pattern recognition methods as a base of development of new instruments for investigations in physical chemistry. In *2022 The 3rd European Symposium on Software Engineering* 127–132. https://doi.org/10.1145/3571697.3573941 (2022).
44. Patent 36236. Adder by module $2^2$–1/Mun G. A., Baipakbaeva S. T., Kadyrzhan K. N., Kabdushev Sh. B., Vitulyova E. S., Konshin S. V. & Suleimenov I. E.; publ. 26.05.2023.
45. Cheah, C. C., Hou, S. P. & Slotine, J. J. E. Region-based shape control for a swarm of robots. *Automatica* **45**(10), 2406–2411 (2009).
46. Bayındır, L. A review of swarm robotics tasks. *Neurocomputing* **172**, 292–321 (2016).
47. Li, C., Wang, J., Liu, J. & Shan, J. Cooperative visual–range–inertial navigation for multiple unmanned aerial vehicles. *IEEE Trans. Aerosp. Electron. Syst.* **59**(6), 7851–7865 (2023).
48. Zhao, H., Liu, H., Leung, Y. W. & Chu, X. Self-adaptive collective motion of swarm robots. *IEEE Trans. Autom. Sci. Eng.* **15**(4), 1533–1545 (2018).
49. Yahao, D. *et al.* Distributed machine learning for UAV swarms: Computing Sensing, and Semantics. *IEEE Internet Things J.* **1**(15), 7447–7473 (2024).
50. Li, C., Guo, G., Yi, P. & Hong, Y. Distributed pose-graph optimization with multi-level partitioning for multi-robot SLAM. *IEEE Robot. Autom. Lett.* **9**(6), 4926–4933 (2024).
51. Wei, L. & Zhijun, G. A distributed flocking control strategy for UAV groups. *Comput. Commun.* **153**, 95–101. https://doi.org/10.1016/j.comcom.2020.01.076 (2020).
52. Xinhua, W., Guanyu, C., Huajun, G. & Jiang, Ju. UAV swarm autonomous control based on Internet of Things and artificial intelligence algorithms. *J. Intell. Fuzzy Syst.* **40**(4), 7121–7133. https://doi.org/10.3233/JIFS-189541 (2021).
53. Carli, R., Cavone, G., Epicoco, N., Ferdinando, M., Scarabaggio, P. & Dotoli, M. Consensus-Based Algorithms for Controlling Swarms of Unmanned Aerial Vehicles. In *Ad-Hoc, Mobile, and Wireless Networks*, Vol. 12338 (2020).
54. Asaamoning, G., Mendes, P., Rosário, D. & Cerqueira, E. Drone swarms as networked control systems by integration of networking and computing. *Sensors* **21**, 2642. https://doi.org/10.3390/s21082642 (2021).
55. Elkilany, B. G. *et al.* A proposed decentralized formation control algorithm for robot swarm based on an optimized potential field method. *Neural Comput. Appl.* **33**, 487–499. https://doi.org/10.1007/s00521-020-05032-0 (2021).

56. Ermukhambetova, B. *et al.* New approaches to the development of information security systems for unmanned vehicles. *Indones J. Electr. Eng. Comput. Sci.* **31**, 810 (2023).
57. Hamamreh, J. M., Furqan, H. M. & Arslan, H. Classifications and applications of physical layer security techniques for confidentiality: A comprehensive survey. *IEEE Commun. Surv. Tutor.* **21**(2), 1773–1828 (2018).
58. Wang, D., Bai, B., Zhao, W. & Han, Z. A survey of optimization approaches for wireless physical layer security. *IEEE Commun. Surv. Tutor.* **21**(2), 1878–1911 (2018).
59. Kundu, S., Hossain, M. & Mandal, S. Modeling of silicon microring resonator-based programmable logic device for various arithmetic and logic operation in Z-domain. *Opt. Quantum Electron.* 55(2), 175 (2023).
60. Beuchat, J.-L. Some modular adders and multipliers for field programmable gate arrays. In *Proceedings International Parallel and Distributed Processing Symposium*, Vol. 8 https://doi.org/10.1109/IPDPS.2003.1213353 (2003).
61. Abd-Elkader, A. A. H., Rashdan, M. & Hasaneen, E. S. A. M. HFA Hamed, Efficient implementation of Montgomery modular multiplier on FPGA. *Comput. Electr. Eng.* **97**, 107585. https://doi.org/10.1016/j.compeleceng.2021.107585 (2022).
62. Kuo, C.-T. & Wu, Y.-C. FPGA implementation of a novel multifunction modulo ($2^n \pm 1$) multiplier using radix-4 booth encoding scheme. *Appl. Sci.* **13**(18), 10407. https://doi.org/10.3390/app131810407 (2023).
63. Patel, B. K. & Kanungo J. Efficient Tree Multiplier Design by using Modulo $2^n + 1$ Adder. In *Emerging Trends in Industry 4.0 (ETI 4.0)* 1–6. https://doi.org/10.1109/ETI4.051663.2021.9619220 (2021).
64. Patent EA030205B1. Modulo four adder. Valeriy Pavlovich Suprun. 2018-07-31
65. Patent RU2724597C1. Russian Federation, G06F 7/72. Multi-digit parallel adder modulo with serial transfer: 2019144521, 27.12.2019: published 25.06.2020. Petrenko Viacheslav Ivanovich, Stepanian Nerses Ernestovich, Nelidin Iurii Romanovich.
66. Gabrielyan, O., Vitulyova, E. & Suleimenov, I. Multi-valued logics as an advanced basis for artificial intelligence (as an example of applied philosophy). *Wisdom* **1**(21), 170–181 (2022).


## Acknowledgements

This research has been funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. AP15473354).

## Author contributions

Conceptualization, I.S. and A.B.; methodology, Y.V. and D.M.; validation, D.S. and D.M.; formal analysis, A.B and Y.V.; investigation, D.S.; resources, Y.V.; data curation, Y.V.; writing—original draft preparation, I.S.; writing—review and editing, Y.V.; visualization, D.M.; supervision, A.B and Y.V.; project administration, Y.V.; funding acquisition, A.B. All authors have read and agreed to the published version of the manuscript.

## Competing interests

The authors declare no competing interests.

## Additional information

**Correspondence** and requests for materials should be addressed to Y.V.

**Reprints and permissions information** is available at www.nature.com/reprints.

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

applied sciences

MDPI

*Article*

# Peculiarities of Applying Partial Convolutions to the Computation of Reduced Numerical Convolutions

**Ibragim Suleimenov [1], Aruzhan Kadyrzhan [2], Dinara Matrassulova [1] and Yelizaveta Vitulyova [1,3,*]**

[1] National Engineering Academy of the Republic of Kazakhstan, Almaty 050010, Kazakhstan; esenych@yandex.kz (I.S.); dinara.kutlimuratovna@gmail.com (D.M.)
[2] Institute of Communication and Space Engineering, Almaty University of Power Engineering and Telecommunication Named Gumarbek Daukeev, Almaty 050013, Kazakhstan; aru.kadyrzhan@gmail.com
[3] National Scientific Laboratory for the Collective Use of Information and Space Technologies (NSLC IST), Satbayev University, Almaty 050013, Kazakhstan
* Correspondence: lizavita@list.ru; Tel.: +7-(705)-6660266

**Abstract:** A method is proposed that reduces the computation of the reduced digital convolution operation to a set of independent convolutions computed in Galois fields. The reduced digital convolution is understood as a modified convolution operation whose result is a function taking discrete values in the same discrete scale as the original functions. The method is based on the use of partial convolutions, reduced to computing a modulo integer $q_0$, which is the product of several prime numbers: $q_0 = p_1 p_2 \dots p_n$. It is shown that it is appropriate to use the expansion of the number $q_0$, to $q = p_0 p_1 p_2 \dots p_n$, where $p_0$ is an additional prime number, to compute the reduced digital convolution. This corresponds to the use of additional digits in the number system used to convert to partial convolutions. The inverse procedure, i.e., reducing the result of calculations modulo $q$ to the result corresponding to calculations modulo $q_0$, is provided by the formula that used only integers proved in this paper. The possibilities of practical application of the obtained results are discussed.

**Keywords:** partial convolutions; algebraic rings; Galois fields; moving average method; convolution theorem; computation modulo a prime number; idempotent elements



**Citation:** Suleimenov, I.; Kadyrzhan, A.; Matrassulova, D.; Vitulyova, Y. Peculiarities of Applying Partial Convolutions to the Computation of Reduced Numerical Convolutions. *Appl. Sci.* **2024**, *14*, 6388. https://doi.org/10.3390/app14146388

Academic Editor: Jose María Alvarez Rodríguez

Received: 31 May 2024
Revised: 16 July 2024
Accepted: 19 July 2024
Published: 22 July 2024



## 1. Introduction

The convolution operation is widely used in various sciences (optics [1,2], electronics [3,4], etc.).

Digital convolution is the basis of convolutional neural networks (CNN), [5,6]. By digital convolution, hereinafter, we will understand convolution, under the sign of which are functions whose values correspond to a certain finite set of discrete quantities.

It is appropriate to emphasize that CNN [5,6] represent one of the most common varieties of artificial neural networks (ANNs). The last wave of ANN development, demonstrating the ability of CNN to show extraordinary results in the field of pattern classification, is exactly associated with such networks [7]. An important area of application of convolutional neural networks is pattern recognition and computer vision [8–10], natural language text processing [11,12], speech recognition in noisy environments [13], etc.

As it was noted in [14,15], for digital signals, there is a wide enough space in the choice of the function serving as a signal model. In particular, in these reports, it was emphasized that functions taking values on the set of real or complex numbers actually represent some mathematical object, allowing for the creation of a signal model. The choice of such an object allowing for the creation of a signal model is ultimately a matter of agreement and convenience [14,15].

In particular, if a signal corresponding to discrete levels lying within a finite range of amplitudes is considered, then any function that can serve as a model of this signal must actually represent a mapping of the current variable (time variable) onto some finite set.

Both Galois fields [16,17] and finite algebraic rings [18] can be chosen as such a set. It should be emphasized that such algebraic structures are nowadays more and more widely used in information technologies [19–21], and this also applies to non-associative algebras [22].

In this paper we show that it is possible to bring the computation of digital convolutions to computations in Galois fields, in which, in particular, the numerical analog of the classical convolution theorem [15] is valid.

This approach is of interest, in particular, because from the functions taking the values within Galois fields, it is admissible to pass to functions whose values correspond to variables of multivalued (as well as fuzzy) logic [23,24].

Specific computations that are performed in Galois fields or finite algebraic rings can in many cases be reduced to computations modulo integers. At present, circuits of various types allowing for the realization of such an operation are being developed quite actively [25–28], with cryptography remaining the main field of application here. This question is also reflected in the patent literature [29–32], where various variants of circuits of multipliers and adders modulo some numbers have been proposed.

It is well known that computations modulo integers allows for operating independently on the components of an integer representation [33–36].

It can also be used to represent a convolution operation as a set of partial convolutions, each of which corresponds to a particular Galois field.

However, when using digital convolutions, there is a significant nuance. The sampling scales of the original function and the function resulting from the convolution operator are different.

To bring them into a mutually unambiguous correspondence, it is necessary to use an operation that can be interpreted as a transition to the reduced digital convolution.

By the term "reduced digital convolution", we mean the result of the convolution operation computation reduced to the same sampling scale as the original functions (functions under the convolution sign).

Such operation can be implemented using the usual rounding operation for fractional numbers; however, it is of interest to ensure that the calculation reduced digital convolution in terms of operations modulo integers.

This approach provides a significant reduction in the amount of computation required to bring the result of the convolution operation to the same discretization scale as the discretization scale of the functions under its sign. When using conventional rounding, decimal fractions have to be calculated. The proposed approach allows for operating only with integers. Moreover, this approach allows for the realization of computations directly in the form of electronic circuits, since nowadays, methods allowing for the realization of computations in Galois fields directly in the form of electronic circuits are well enough developed.

In addition, this approach creates prerequisites for the subsequent transition to the use of functions taking values in variables of multivalued logic, etc.

The specific formulation of the problem solved in this paper is presented in Section 3.

The novelty of the work is as follows. We propose an approach based entirely on modulo integer computations, which allows for the result of convolution operation to be brought to the same discretization scale as the discretization scale of original functions.

The proposed approach, among other things, allows for bringing the digital convolution to calculations in Galois fields. Separately, this work for the first time substantiates the method of choosing simple numbers that allow for carrying out this operation in the simplest and most convenient way.

## 2. Related Works

This work lies in the general trend of modern research aimed at improving the efficiency of computational systems, including those focused on the use of convolutional neural networks. References to specific works are given in Table 1.

**Table 1.** Classification of related works.

| Problem Being Discussed | References |
|---|---|
| Fast arithmetic, Systems of Residual Classes | [37–40] |
| Use of non-trivial algebraic structures in information technology | [18,41,42] |
| Convolutional neural networks | [5–13] |
| Use of Galois fields in information technology | [43,44] |
| Using Galois fields in the aspect of multivalued logics | [23,24] |
| Chips with reconfigurable logic | [45–48] |
| Applications of computing based on chips with reconfigurable logic structure | [49,50] |

One of the important directions here is related to the use of RNS (modular arithmetic), which can be used for neural networks too [37]. Such systems are closely related to fast arithmetic [38,39], i.e., an area of research that has received increasing attention over the last decades [40]. This work solves the problem of bringing the result of convolution operation to the original discretization scale, while preserving the possibility of computation in terms of RNS, i.e., it responds to the above trend.

It is of importance that the latter approach allows for generalization, since RNS can be considered as a special case of finite algebraic rings [24]. A wide variety of rings can be used for representation any set containing a finite number of elements, including non-associative ones [41,42]. This, in particular, justifies the use of the term "partial convolutions", which implies the possibility of a subsequent transition to the use of finite algebraic rings of various types. A concrete example of using non-trivial algebraic rings for digital signal processing was given in [18]. The generalization of the RNS approach obviously creates additional opportunities for digital signal processing.

Further, some subsets of an algebraic ring used to represent sets containing a finite number of elements in many cases (including the case considered in this paper) can be put in correspondence with Galois fields. Such fields are also increasingly used in information technology, in particular, in cryptography to improve the reliability of encoding [43,44].

Computations in such fields can be realized directly in the form of electronic circuits, and research in this direction is also actively carried out at present [45,46]. More precisely, computations of this kind can be realized on the basis of chips with tunable logic structure, which are also actively used nowadays [47,48]. Reduced convolution operation is therefore of interest, also from the point of view of implementing the corresponding computations (including the implementation of convolutional neural networks) directly by specially programmed chips.

An example in this respect is a very definite problem arising in the processing of satellite video information intended for remote sensing of the Earth. Communication channels have limited bandwidth, so it is important to exclude images in those situations when the Earth surface is covered by clouds [49,50]. To solve this problem, it is expedient to use on-board means of information pre-processing, which can use convolutional neural networks. This particular example demonstrates the importance of the problem solved in this paper. A sufficiently coarse sampling scale can be used to exclude frames with cloud cover. Applying the convolution operation without reduction to the original sampling scale will not only correspond to the excess of accuracy, but will also require additional computational resources. It becomes especially important to overcome this difficulty when the corresponding operations are performed directly by the onboard computer.

The transition to computations in Galois fields has one more aspect. Namely, in this case it is possible to operate with functions taking values on variables of multivalued logic [23,24]. This aspect is important in those cases when, for example, we analyze the relationships between the preceding and following values of time series data, which are also used in different sciences. Provided the system is linear, it can often be assumed that such relationships are given by a convolution operation. The transition to the use of multivalued logics in the future allows us to raise the question of identifying true logical relationships in prediction, but the convolution operation itself should be reduced to logical

variables. In this case, the preservation of the discretization scale is essential too, since this corresponds to the use of the same set of logical variable values for both the input and output functions.

Thus, this work, which provides the possibility of reducing the convolution operation to computations in Galois fields of relatively small size (which is ensured, among other things, by bringing the discretization scale of the result to the discretization scale of the original functions), contributes, among other things, to the interdisciplinary integration between the above-mentioned fields of research.

## 3. Background Section

Consider the usual expression for the convolution operation applied to functions taking discrete values

$$U_{out}(i) = \sum_j K(j)\ U_{in}(i-j) \tag{1}$$

where $K(j)$—convolution operator core, $U_{out}(i)$ and $U_{in}(i)$—functions, which we will treat as "output" and "input", respectively.

Let us assume that the functions $U_{in}(i)$ and $K(j)$ are digital, i.e., each of them at each of the index values takes one of $N$ discrete values corresponding to the sampling scale arising from the nature of the problem to be solved.

Besides, we will also assume that the function $K(j)$ is different from zero only at $M$ clock cycles

Then the number of values $N_1$, which can be taken by the function $U_{out}(i)$, generally speaking, lies within the limits defined by the inequality

$$0 \le N_1 \le MN^2 \tag{2}$$

One can see that this numerical range differs significantly from the range of variation of the original function $U_{in}(i)$.

The reduction of these two ranges to each other can be performed by the operation of division by a suitable number $M_1$ followed by rounding to an integer.

Operations of this kind, however, cannot be performed using operations only on integers, for example, they cannot be reduced to modulo operations.

Consequently, the problem is to find algebraic operations in terms of integers, allowing for the reduction of the scale of the function $U_{out}(i)$ to the same scale as the function $U_{in}(i)$.

This problem is solved in this paper using the method of partial convolutions.

## 4. Methods

### *4.1. Representation of Integer Computations in Terms of Finite Algebraic Rings*

The basic method used in this paper is the theory of algebraic rings. The existence of rings $R$ which are decomposed into a direct sum of ideals $r_i$ is proved in this theory

$$R = r_1 + r_2 + \cdots + r_n \tag{3}$$

Each of these ideals is generated by idempotent elements $e_i$

$$r_i = Re_i \tag{4}$$

Such elements cancel each other out

$$e_i e_j = 0,\ i \ne j;\ e_i e_i = e_i \tag{5}$$

Their sum is equal to the unit of the ring $R$

$$\sum_i^n e_i = 1 \tag{6}$$

The simplest example of rings of this type is generated by a homomorphism of the ring of integers to the ring of residue classes modulo $p$ for the case where the number $p$ is the product of several prime numbers $p_i$.

$$P = p_1 p_2 \dots p_N \tag{7}$$

In this case one can represent an arbitrary element of the ring $R$ in the form

$$u = e_1 u_1 + e_2 u_2 + \dots + e_N u_N \tag{8}$$

where $e_i$—idempotent mutually cancelling elements, and $u_i = 0, 1, 2, \dots, p_i$.

Idempotent elements are formed by the rule

$$e_i = \alpha_i \prod_{i \neq j}^{N} p_j \tag{9}$$

where $\alpha_i$—integer. The choice of these numbers is based on the condition

$$e_i e_i = 1 \tag{10}$$

One can see,

$$e_i p_i \equiv 0 \bmod P \tag{11}$$

since any product of the form (11) contains a multiplier $P = p_1 p_2 \dots p_N$.

Indeed, the idempotent element $e_i$ contains (9) the product of all prime factors of the number $P$ except $p_i$, and $p_i$ enters formula (11) directly.

With the choice of integers made, $\alpha_i$ also holds.

$$e_1 + e_2 + \dots + e_N \equiv 1 \bmod P \tag{12}$$

Note that a special case of the representation (3) is the case when we consider the ring of residual classes modulo an integer.

In this case, the representation in the form (3) corresponds to RNS computations.

To illustrate, consider an example corresponding to the case of the product of three prime numbers 2, 3 and 7. The product of these numbers is 42. We will consider the ring of residue classes modulo 42.

Idempotent mutually cancelling elements can be easily found by the scheme corresponding to formula (9). Let us compose the products $3 \cdot 7 = 21$, $3 \cdot 2 = 6$, $2 \cdot 7 = 14$. All these elements will (when calculating modulo 42) cancel each other, since the result of their multiplication by each other will contain the factor $2 \cdot 3 \cdot 7$, i.e., this result will be a multiple of 42.

By direct verification it is also possible to verify that for the case under consideration the following equations holds.

$$36 \cdot 36 \equiv 36 \bmod 42,\ 28 \cdot 28 \equiv 28 \bmod 42,\ 21 \cdot 21 \equiv 21 \bmod 42 \tag{13}$$

Accordingly, when carrying out calculations modulo 42, the record (8) takes the form

$$u = 21 \cdot u_1 + 28 \cdot u_2 + 36 \cdot u_3 \tag{14}$$

where

$$u_1 = 0, 1,\ u_2 = 0, 1, 2,\ u_3 = 0, 1, \dots, 6 \tag{15}$$

The relation (12) is also fulfilled

$$21 + 28 + 36 \equiv 1 \bmod 42 \tag{16}$$

Note also that the representation (8) and its special case (14) can be considered as a representation of numbers not exceeding 42 in a number system with hybrid base.

Indeed, one can see an analogy [51] between the notation (14) and the expression that defines the positional notation of a number, say, in a number system with base 10.

$$a = a_0 + 10^1 \cdot a_1 + 10^2 \cdot a_2 + \cdots \tag{17}$$

It can be seen that in both cases there is a certain selected set of numbers, which form a representation of any other number as a sequence of symbols from a finite set (digits).

In the case of the notation (17), such a set is formed by the powers of the base number 10. But, such a choice, in general, is not obligatory.

The standard decimal notation $a = \cdots a_2 a_1 a_0$ A similar notation can be used from expression (8) or its special case (14). For example, for the representation of (14) one can write down

$$u = u_1 u_2 u_3 \tag{18}$$

where the positions occupied by specific values $u_i$ can be interpreted as analogues of digits in the traditional number system.

An essential advantage of the considered number system is the possibility of independent operation with analogues of digits of a number.

The operation of addition in decimal (binary, etc.) representation is obviously connected with the fact that the result of adding the lower digits, generally speaking, will affect the result obtained by adding the higher digits.

When using the representation of the form (14), such a problem disappears. The "high" and "low" digits can be handled in a completely independent way.

Consider the product of two numbers written in the form (8)

$$u^{(1)}u^{(2)} = \left(e_1 u_1^{(1)} + e_2 u_2^{(1)} + \cdots + e_N u_N^{(1)}\right)\left(e_1 u_1^{(2)} + e_2 u_2^{(2)} + \cdots + e_N u_N^{(2)}\right) \tag{19}$$

Because $e_i$ are mutually cancelling idempotent elements, we have

$$u^{(1)}u^{(2)} = e_1 u_1^{(1)} u_1^{(2)} + e_2 u_2^{(1)} u_2^{(2)} + \cdots + e_N u_N^{(1)} u_N^{(2)} \tag{20}$$

In this case, the computation of products $u_i^{(1)} u_i^{(2)}$ is actually performed in the sense of the multiplication operation in the Galois field $GF(p_i)$, since the multiplication operation is performed modulo $p_i$.

This conclusion is also true for the addition operation.

Expression (20) unambiguously shows that the result of calculating the product of "higher" digits is indeed completely independent of the result of calculating the product of "lower" digits. Moreover, such digits can be interchanged.

Formulas (19) and (20) allow for the passing to partial convolutions.

### 4.2. Partial Convolutions

Let us choose the above prime numbers $p_i$ in accordance with next formula

$$U_{out}(i) < p_1 p_2 \ldots p_N \tag{21}$$

where $N$—number of digits in hybrid coding.

Then, one can consider that the convolution formula applied to discrete functions (1) is written in terms of elements of rings admitting the representation (8).

With the assumption made, we can substitute into formula (1) the expansion through idempotent elements, and for both $K(j)$, and $U_{in}(i)$. We have:

$$U_{out}(i) = \sum_j \left(\sum_m e_m K_m(j)\right)\left(\sum_m e_m U_{m,in}(i-j)\right) \tag{22}$$

By reversing the signs of summation, we obtain

$$U_{out}(i) = \sum_m e_m \left( \sum_j K_m(j) U_{m,in}(i-j) \right) \quad (23)$$

when deriving formula (23), it is taken into account that idempotent elements $e_m$ cancel each other at multiplication (5).

Accordingly, the multiplier at $e_m$ contains only values with the same index $m$.

In the last formula, the summation at index $j$ is taken in separate brackets to emphasise the following circumstance.

When the functions corresponding to discrete signals are represented in a hybrid number system, the convolution operation is factorised. Each component of the used functions can be operated with in a completely independent way.

Accordingly, in this case we can speak about partial convolutions. We have:

$$U_{out,m}(i) = \sum_j K_m(j) U_{m,in}(i-j) \quad (24)$$

Note that the operations of addition and multiplication in formula (24), as follows from the above, are performed modulo $p_m$, i.e., operation (24) is performed in the Galois field $GF(p_m)$.

In particular, the description of any "digitized" system (a system reduced to discrete values) possessing the property of invariance with respect to the shift operation on the current discrete variable is exhausted by operations of this kind.

It is essential that the use of partial convolutions in terms of Galois fields is admissible only when an inequality of the form (2) is satisfied. In this case, the maximum value that can be given by the convolution operation does not exceed the integer modulo which the computation is performed.

Consequently, in this case, the result of operations modulo coincides with the results of operations in the sense of ordinary addition and multiplication of integers.

Here, however, a nuance arises. Indeed, if the original function varied in the range corresponding to integers not exceeding $N_1$, the convolution result can vary within a much wider range, formula (2).

Consequently, if the original function was mapped to a hybrid number system containing $N$ digits, the number of digits must be increased to adequately represent the convolution result of the form (1).

The solution to this problem serves as the foundation for the proposed approach.

## 5. Results

### *5.1. Increasing the Number of Digits in the Hybrid Number System*

Let us find a rule according to which a number represented in a hybrid number system with a certain number of digits can be written in a similar system with a larger number of digits.

The number 1 in an arbitrary encoding of the type under consideration is represented by the sum of idempotent elements (an example is formula (16)).

$$1 \equiv e_1 + e_2 + \cdots + e_N \bmod P \quad (25)$$

where the comparison is modulo the integer $P$.

Accordingly, when adding a unit to any number, a unit is added to each of the digits. But for each of the digits the addition is performed modulo the prime number $p_i$, which corresponds to the given digit. Hence,

$$u_i(n+1) \equiv u_i(n) + 1 \ (p_i) \quad (26)$$

Formula (26) allows, among other things, for an explicit switch from writing a number in a decimal number system to writing in a number system with a hybrid base. Namely,

$$u_i \equiv U \bmod p_i \tag{27}$$

i.e., the value of each of the digits is directly the initial number $U$ taken modulo $p_i$. Thus, formula (27) allows for the reduction of any original function given in the usual numerical form to the form corresponding to the hybrid number system.

The value of the additional digit can also be found using formula (27).

Supplement the hybrid number system formed by the prime numbers $p_i$, $i = 1, 2, .., N$ with one more digit corresponding to the prime number $p_{N+1}$.

The number of idempotent elements appearing in the formula (8) will increase by one, thus

$$e_i = \widetilde{\alpha}_i p_{N+1} \prod_{i \neq j}^{N} p_j;\ i = 1, \ldots, N;\ e_{N+1} = \widetilde{\alpha}_{N+1} \prod_{1}^{N} p_j \tag{28}$$

In particular, it means that idempotent elements change, but in the formula of the form (8) the values of the components $u_i;\ i = 1, .., N$ remain unchanged.

The fact follows from formula (26). This formula is also applicable to calculate the value of the additional component $u_{N+1}$.

Consequently, the operations of calculating partial convolutions in the fields $GF(p_i)$; $i = 1, .., N$ remain unchanged when one more digit is added. Its value, as follows from formula (26) is

$$u_{N+1} \equiv U \bmod p_{N+1} \tag{29}$$

This formula solves the problem of number representation when increasing the number of digits; however, the next question arises. It is necessary to reduce the result of calculations in the system with an increased number of digits to the original sampling scale, in which the considered functions take integer values in the range from 0 to $N_1$.

The most obvious way of reduction to the original scale is given by the formula

$$U_{out}(i) = \left\lfloor \frac{1}{p_{N+1}} \sum\nolimits_j K(j)\ U_{in}(i-j) \right\rfloor \tag{30}$$

where $\lfloor a \rfloor$ is floor function symbol.

The formula is obtained from the following considerations.

According to the above assumptions, the functions under the convolution sign change in the range from 0 to $P$. The result of convolution calculation changes in the range from 0 to $Pp_{N+1}$. Therefore, to bring the sampling scale to the original one, the result of convolution calculation should be divided by $p_{N+1}$.

However, as noted in Section 2, it is an urgent task to find a method that would allow for the obtainment of the same result as formula (30), but only by algebraic means, i.e., without using fractional numbers.

### 5.2. *Formula for Converting the Result of Calculations Using Partial Convolutions to the Original Sampling Scale*

The following lemma is valid.

For two natural numbers $w_1$ and $w_2$, $w_1 > w_2$ and any natural n the following holds.

The value $q$

$$q \equiv (w_1 - w_2) \left\lfloor \frac{n}{w_1} \right\rfloor \bmod w_2 \tag{31}$$

where $\lfloor a \rfloor$ is floor function symbol, can also be calculated as

$$q \equiv (n - g) \bmod w_2,\ g \equiv n \bmod w_1 \tag{32}$$

The nature of the change in the value of $q$ with increasing $n$ for the special case $p_1 = 19$, $p_2 = 17$ is illustrated by curve 1, Figure 1. The figure emphasizes that the value of $q$ remains

constant at each of the intervals corresponding to the number $p_2 = 17$, i.e., it changes by a jump when the abscissa value becomes a multiple of 17.

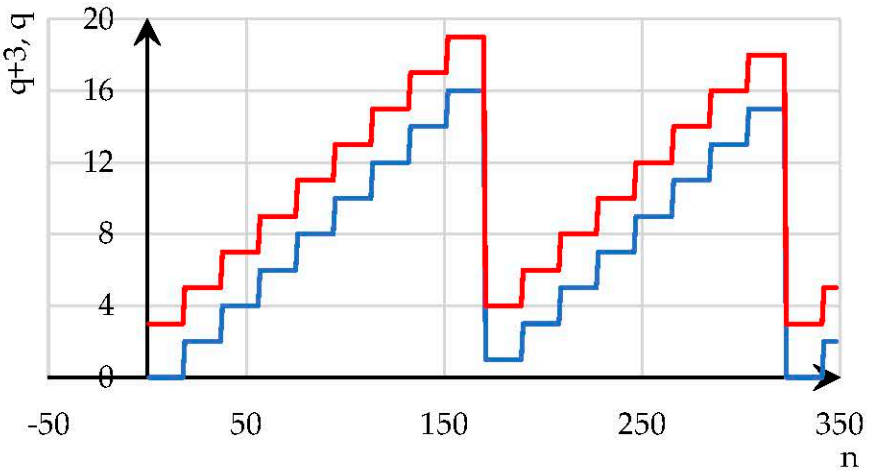


**Figure 1.** Illustration of the lemma proof; $w_1 = 19$, $w_2 = 17$, curve 1 (blue)—calculations of $q$ by formula (31), i.e., through the operation of calculating the integer part of the number, curve 2 (red)—values of the sum $q + 3$, where $q$ is calculated by formula (32), i.e., using only modulo operations.

This property allows us to prove the lemma under consideration.

Figure 1 also shows the dependence of $q + 3$, where $q$ is calculated by formula (32). The dependence of $q$ is presented with a shift by 3 so that the curves do not overlap.

It can be seen that if the artificial shift by 3 is not taken into account, the curves coincide.

Similar curves, but for the case $p_1 = 11$, $p_2 = 8$ are presented in Figure 2. This figure also emphasizes that the value of $q$ changes by a jump when the value of abscissa becomes a multiple of $p_1 = 11$. This figure also clearly shows that the specific value of $q$ also depends on the value of $p_2$, in particular, at the chosen values of $p_1$ and $p_2$ the considered function does not have a pronounced periodic character. In the future, this fact will be used to justify the choice of the relationship between the numbers $p_1$ and $p_2$.

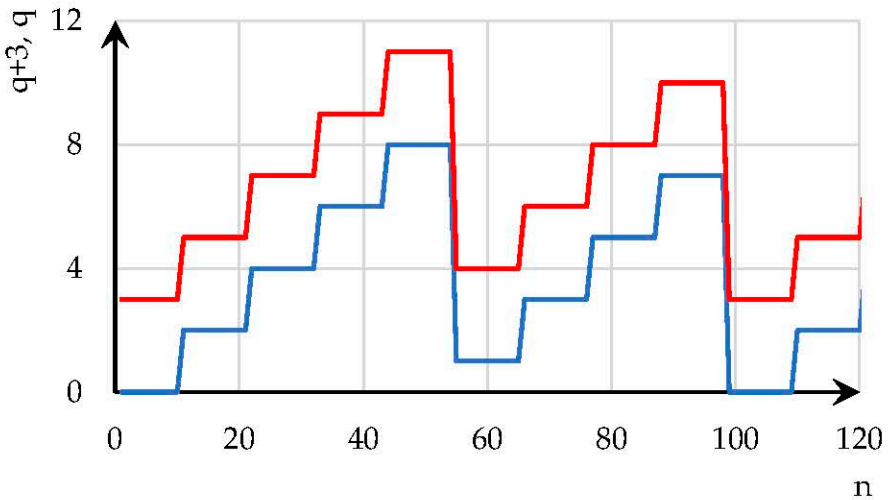


**Figure 2.** Illustration of the lemma proof; $w_1 = 11$, $w_2 = 9$, curve 1 (blue)—calculations of $q$ by formula (31), i.e., through the operation of calculating the integer part of the number, curve 2 (red)—values of the sum $q + 3$, where $q$ is calculated by formula (32), i.e., using only modulo operations.

Let us return to the proof of the lemma formulated above.

There is an identity

$$n - n(\text{mod } w_1) = \left\lfloor \frac{n}{w_1} \right\rfloor w_1 \tag{33}$$

where the notation mod $w_1$ is written out explicitly, because the difference on the left side of (33) is calculated in the sense of the usual operations with integers.

The identity (33) follows from the fact that an arbitrary number $n$ can be represented as the sum of the residue of division by an integer $w_1$ of the integer part of the result of such division

$$n = n(\text{mod } w_1) + \left\lfloor \frac{n}{w_1} \right\rfloor w_1 \tag{34}$$

Provided that $w_1 > w_2$, the following equality is also true

$$\left( \left\lfloor \frac{n}{w_1} \right\rfloor w_1 \right) \equiv \left( (w_1 - w_2) \left\lfloor \frac{n}{w_1} \right\rfloor \right) \text{mod } w_2 \tag{35}$$

Since the value $\left\lfloor \frac{n}{w_1} \right\rfloor w_2$ is known to be divisible by $w_2$.

Hence, applying the operation of taking modulo $w_2$ to both parts of equality (33), we obtain

$$\left( (w_1 - w_2) \left\lfloor \frac{n}{w_1} \right\rfloor \right) \equiv (n - n(\text{mod } w_1)) \text{ mod } w_2 \tag{36}$$

The lemma is proved.

The most interesting case is when $w_1 - w_2 = 1$. Then the left part of equality (36) becomes equal to $\left\lfloor \frac{n}{w_1} \right\rfloor (\text{mod } w_2)$.

Note that if we consider such a range of variation of $n$ that $0 < n < w_1 w_2$, then $0 < \left\lfloor \frac{n}{w_1} \right\rfloor < w_2$ takes place. Hence,

$$\left\lfloor \frac{n}{w_1} \right\rfloor (\text{mod } w_2) = \left\lfloor \frac{n}{w_1} \right\rfloor \tag{37}$$

Hence, in the considered case, the remainder from dividing $n$ by the integer $w_1$ is expressed as

$$\left\lfloor \frac{n}{w_1} \right\rfloor = (n - n(\text{mod } w_1))(\text{mod } w_2) \tag{38}$$

It can be seen that, in the considered particular case, the proved lemma allows for the reduction of the calculation of the value $\left\lfloor \frac{n}{w_1} \right\rfloor$ to the operations of modulo taking.

This allows us to preserve the character of sampling of the scales of the "input" and "output" functions.

Let us obtain specific formulas that allow us to perform this operation.

The initial range of variation of the functions under consideration, whose values are represented in the form (8), is limited by the number $w_2 = \prod_1^N p_j$.

To fulfill the condition ensuring the fulfilment of formula (38), the prime number $p_{N+1}$ corresponding to the additional digit must be given by the formula

$$p_{N+1} = \prod_1^N p_j + 1 \tag{39}$$

Examples of prime numbers $p_{N+1}$ represented in the form (39) are given in Table 2.

**Table 2.** Multipliers $p_i$ giving examples of numbers of the form (39). (The symbol "-" means that this multiplier is not included in M).

| $p_4$ | $p_3$ | $p_2$ | $p_1$ | $M = p_1 p_2 \dots p_s$ | $p_{N+1}$ | $Mp_{N+1}$ |
|---|---|---|---|---|---|---|
| - | - | 3 | 2 | 6 | 7 | 42 |
| - | - | 5 | 2 | 10 | 11 | 110 |
| - | - | 11 | 2 | 22 | 23 | 506 |
| - | 5 | 3 | 2 | 30 | 31 | 930 |
| - | 7 | 3 | 2 | 42 | 43 | 1806 |
| - | 11 | 3 | 2 | 66 | 67 | 4422 |
| - | 7 | 5 | 2 | 70 | 71 | 4970 |
| - | 17 | 3 | 2 | 102 | 103 | 10,506 |
| - | 13 | 5 | 2 | 130 | 131 | 17,030 |
| - | 19 | 5 | 2 | 190 | 191 | 36,290 |
| 7 | 5 | 3 | 2 | 210 | 211 | 44,310 |
| - | 31 | 5 | 2 | 310 | 311 | 96,410 |
| 11 | 5 | 3 | 2 | 330 | 331 | 109,230 |
| - | 19 | 11 | 2 | 418 | 419 | 175,142 |

As Table 2 emphasizes, the using even relatively small numbers $p_i$ gives possibility to ensure the computation of partial convolutions of practical interest. Indeed, the range of variation of the numbers represented in the form under consideration is quite large (it is bounded by the product $Mp_{N+1}$).

Proceeding from the above, we will consider the case when

$$w_1 = p_{N+1} = \prod_1^N p_j + 1;\ w_2 = \prod_1^N p_j \tag{40}$$

To return to the original sampling scale, operation (38) must be applied to the result of the computation using partial convolutions, which is represented in the form.

$$\widetilde{U} = e_1 u_1 + e_2 u_2 + \dots + e_N u_N + e_{N+1} u_{N+1},\ \mathrm{mod}(w_1 w_2) \tag{41}$$

This notation, in particular, emphasizes that the integer values of $\widetilde{U}$ vary in the range $0 < \widetilde{U} < w_1 w_2$.

According to formula (33), for an arbitrary integer $s$ it is true that

$$s(\mathrm{mod}\ w_1 w_2) = s - \left\lfloor \frac{s}{w_1 w_2} \right\rfloor w_1 w_2 \tag{42}$$

From where

$$(s(\mathrm{mod}\ w_1 w_2))\mathrm{mod}\ w_1 \equiv s\ \mathrm{mod}\ w_1 \tag{43}$$

since $\left\lfloor \frac{s}{w_1 w_2} \right\rfloor w_1 w_2$ is divisible by $w_1$.

Similarly,

$$[s(\mathrm{mod}\ w_1 w_2)](\mathrm{mod}\ w_2) = s(\mathrm{mod}\ w_2) \tag{44}$$

There are only the values calculated by mod $w_2$ or mod $w_1$ in the right part of the formula (38).

Consequently, as follows from formulas (43) and (44), instead of the value $\widetilde{U}$ given by formula (41), i.e., providing for taking mod $w_1 w_2$, we can use the value $U$ calculated according to the usual rules of addition and multiplication

$$U = p_{N+1} \sum_i u_i \widetilde{\alpha}_i \prod_{j \neq 1}^N p_j + \widetilde{\alpha}_{N+1} \prod_1^N p_j u_{N+1} \tag{45}$$

Or, taking into account (40)

$$U = w_1\left(\sum_i u_i\widetilde{\alpha}_i\prod_{j\neq 1}^{N} p_j\right) + e_{N+1}u_{N+1} \tag{46}$$

Let us apply the mod $w_1$ operation to expression (32). Then

$$U(\text{mod } w_1) \equiv (e_{N+1}u_{N+1})(\text{mod } w_1) = u_{N+1} \tag{47}$$

where it is taken into account that

$$e_i \equiv 1 \text{ mod } p_i \tag{48}$$

The identity (48) follows directly from (25).

As a result, we obtain the following calculation formula, which solves the problem.

$$\left\lfloor \frac{U}{w_1} \right\rfloor \equiv \left(\sum_i u_i\widetilde{\alpha}_i\prod_{j\neq 1}^{N} p_j - u_{N+1}\right) \text{ mod } w_2 \tag{49}$$

Let us consider how exactly the obtained formula can be applied to calculate digital convolutions.

## 6. Discussion

### *6.1. Justification of the Constructiveness of the Proposed Approach*

The advantages of formula (49) over operations of the form (30) are as follows.

The calculations used in formula (49) use only integers. Consequently, the speed of execution of this kind of operation is obviously higher than the speed of execution of operations in which the division operation is used.

Moreover, all of these operations, in principle, can be realized by means of specific adders and multipliers modulo integers. Circuits of this kind of adders and multipliers are known [29–32] and they continue to be improved. It should also be taken into account that integrated circuits with configurable logic are now known too [52,53].

However, even disregarding the above considerations, the computation by formula (51) is known to have an advantage over operations such as (30). Indeed, the range of variation of each partial convolution in the computation in Galois fields $GF(p_i)$ is organised by the integer $p_i$ and, hence, all summands appearing in formula (51) will not exceed $w_1 = 1 + \prod_1^N p_j$. Let us underline, that actual range of the convolution computation (before reduction to the original scale) is $w_1w_2 = \left(1 + \prod_1^N p_j\right)\prod_1^N p_j$.

This is what preserves the original sampling scale when computing the convolution.

Let us consider a concrete example illustrating this advantage.

We will use the ring of modulo 110 residue classes. The elements of such a ring are represented in the form

$$u \equiv 55{\cdot}u_1 + 66{\cdot}u_2 + 100{\cdot}u_3 \text{ mod } 110 \tag{50}$$

where

$$u_1 = 0, 1,\ u_2 = 0, 1, \ldots, 5,\ u_3 = 0, 1, \ldots, 10 \tag{51}$$

The relation (12) is also fulfilled

$$55 + 66 + 100 \equiv 1 \text{ mod } 110 \tag{52}$$

The prime number 11 is representable in the form (39): $11 = 1 + 2{\cdot}5$. Consequently, formula (50) can be considered as the result of increasing the number of digits in the number system corresponding to the ring of subtraction classes modulo 10. In this ring

$$u \equiv 5{\cdot}u_1 + 6{\cdot}u_2 \text{ mod } 10 \tag{53}$$

where

$$u_1 = 0, 1,\ u_2 = 0, 1, \ldots, 5, \tag{54}$$

It can also be written as

$$u \equiv 11 \cdot (5 \cdot u_1 + 6 \cdot u_2) + 100 \cdot u_3 \bmod 110 \tag{55}$$

Accordingly, the calculation formula (49) for the considered particular case takes the following form

$$\left\lfloor \frac{U}{w_1} \right\rfloor = (5 \cdot u_1 + 6 \cdot u_2 - u_3) \bmod 10 \tag{56}$$

where $u_i$ are the values of the digits of the numbers resulting from the calculation of partial convolutions in the number system corresponding to formula (50).

Let us apply the formula to calculate the convolution of model functions.

As the first of such functions, we will use the function $f_1$, shown in Figure 3, curve 1. As the second model function $f_2$ we will use the rectangle function given by the relation

$$f_2(k) = \begin{cases} 0, & |k| > 5 \\ 1, & |k| \leq 5 \end{cases} \tag{57}$$

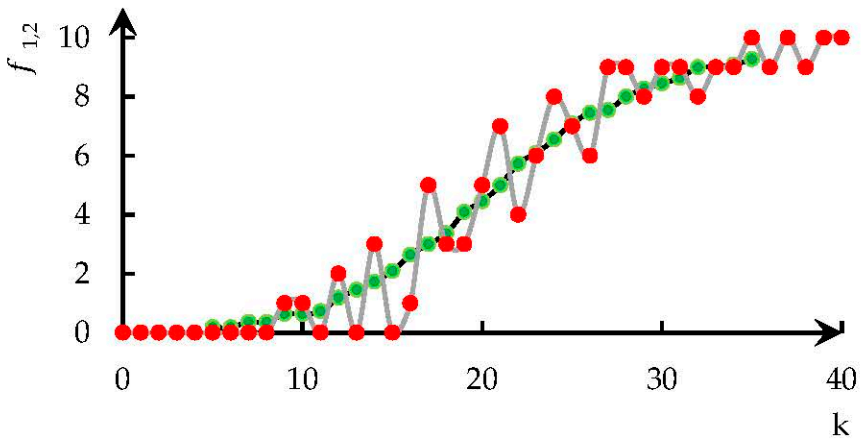


**Figure 3.** Model function $f_1$ (curve 1, red dots) and the result of applying the moving average calculation operation to it (curve 2, green dots).

It can be seen that $f_1$ models a function describing some noisy transient process. Its convolution with the function $f_2$ corresponds to the moving average calculation, which provides noise filtering. If fractional values are allowed, the convolution of functions $f_1$ and $f_2$ is expressed by the usual moving average formula

$$\langle f_1(k) \rangle = \frac{1}{11} \sum_{i=k-5}^{i=k+5} f_1(k) \tag{58}$$

Formula (60) takes into account that function (59) is different from 0 at 11 clock cycles, including the clock with index 0. Accordingly, the summation starts at clock $k - 5$ and ends at clock $k + 5$.

The calculations according to formula (58) are also presented in Figure 3, curve 2. It can be seen that the result of convolution calculation really gives a function describing the model transient process. The function obtained by applying the convolution operation in the form (58) is indeed quite smooth, i.e., this operation suppresses the noise present in the model function, as one would expect.

The presented simple example, among other things, clearly shows that the operation of reducing the convolution result to the original scale is indeed justified.

In many cases, this is also dictated by physical considerations.

Figure 4 also presents the smoothed model curve obtained from the original function using the moving average method (curve 1).

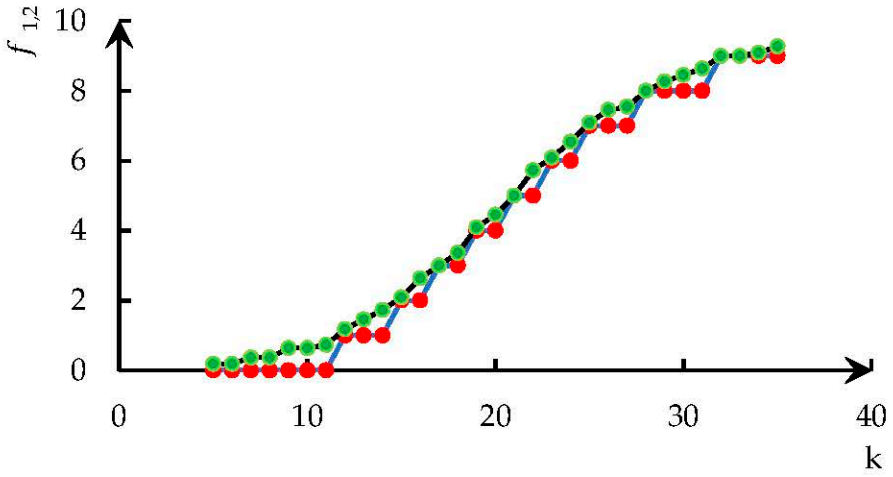


**Figure 4.** Model function $f_1$ smoothed using the moving average method (curve 1, green dots) and the result of calculating the analogue using formula (51), curve 2, red dots.

The same figure shows the result of calculations by formula (51) using partial convolutions. Specifically, the formula was used

$$\langle f_1 \rangle_0 = \left(5 \cdot \left\langle f_1^1 \right\rangle + 6 \cdot \left\langle f_1^2 \right\rangle - \left\langle f_1^3 \right\rangle\right) \bmod 10 \tag{59}$$

where

$$\left\langle f_1^1 \right\rangle = \left(\sum_{i=k-5}^{i=k+5} f_1(k), \mathrm{mod}2\right), \mathrm{mod}2 \tag{60}$$

$$\left\langle f_1^2 \right\rangle = \left(\sum_{i=k-5}^{i=k+5} f_1(k), \mathrm{mod}5\right), \mathrm{mod}5 \tag{61}$$

$$\left\langle f_1^3 \right\rangle = \left(\sum_{i=k-5}^{i=k+5} f_1(k), \mathrm{mod}11\right), \mathrm{mod}11 \tag{62}$$

Formulas (60)–(62) correspond to calculations in Galois fields. The values of the original function taken modulo the corresponding prime number $p_i$ are summed up, and then the operation of taking modulo $p_i$ is applied to the summation result again.

It can be seen that the presented curves coincide with the accuracy up to rounding down, i.e., formula (49) really provides the desired result.

It can also be seen that curve 2, Figure 4, which approximates the smoothed curve with downward rounding accuracy, can indeed be derived from calculations performed in Galois fields.

We emphasize that Figure 4 shows that the result of calculations performed only in Galois fields (in particular, without using fractional values) coincides (with rounding accuracy) with the result obtained by formula (58).

Thus, the used model example serves as a clear demonstration of the adequacy of the proposed methodology of bringing the convolution operation to calculations in Galois fields.

The peculiarities of the methodology used are also vividly illustrated by Figure 5, which shows plots the results of intermediate calculations using formulas (60)–(62).

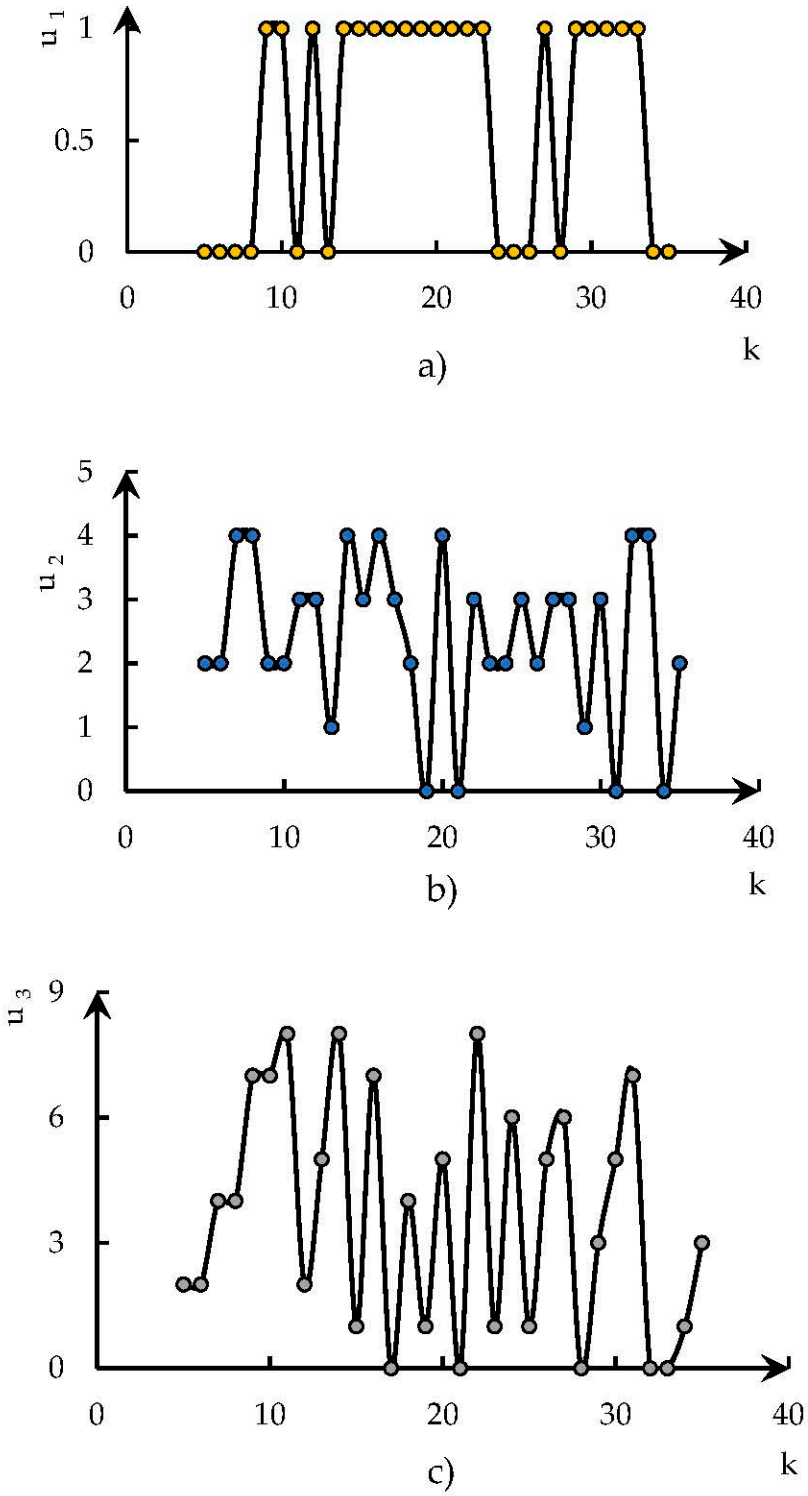


**Figure 5.** Plots of partial convolutions for three digits of number representation modulo 110; (**a**–**c**) correspond to partial convolutions in Galois fields GF(2), GF(5), and GF(11), respectively.

As can be seen in Figure 5, all the functions presented in this figure, vary within the limits corresponding to a particular prime number $p_i$.

Besides, the behavior of obtained functions is not ordered, yet the utilization of formula (51) enables the desired outcome to be achieved.

Accordingly, this figure serves as another visual illustration of the nature of the technique used. The result of the above convolution is computed using a set of functions, each of which varies within finite limits corresponding to a certain Galois field.

Further, all operations for computing partial convolutions in accordance with the above are de facto performed in Galois fields.

Consequently, the numerical analogue of the convolution theorem [15] applies to each of them.

These functions are direct analogues of the transfer functions used in classical linear circuit theory.

Formally, we can write at once

$$F[U_{out}(i)] = \left(\sum_m [K_m(i)]F[U_{m,in}(i)]\widetilde{\alpha}_m\prod_{j\neq m}^{N} p_j - [K_{m+1}(i)]F[U_{m+1,in}(i)]\right)(\mathrm{mod}\ w_2) \quad (63)$$

where $F[f(i)]$ is a notation for the Galois Field Fourier Transform.

Here, of course, there arises a nontrivial problem of finding an adequate basis representing an analogue of harmonic functions; however, the relation (63) already shows that the proposed approach in the future allows for finding transfer functions for systems of different nature described by digital convolution operations.

### *6.2. Some Perspectives on the Use of the Proposed Approach*

Let us consider the most illustrative example of using the operations proposed in this paper. It is related to the analysis of time series of data, which arise in various scientific disciplines, for example, in meteorology [54], as well as in econometrics [55], etc.

Neural networks as well as various intelligent systems on this base are often used for analyzing such data series (including for forecasting purposes) [56–58]. In this respect, convolutional neural networks are obviously of particular interest, since the considered systems, as a rule, possess the property of invariance with respect to the time shift operation.

The proposed approach allows one to reduce any convolution operations to operations in terms of logical variables, which, among other things, is of interest from the point of view of revealing the true causal relations inherent in a particular system.

This example, however, is mainly for illustrative purposes.

Much more important seems to be the next step related to the application of the numerical convolution theorem [18].

Namely, if all operations corresponding to the convolution to the original discretization scale are reduced to convolutions in the sense of Galois fields, then it is acceptable to pass from the convolution to the analog of the transfer function (the Fourier–Galois image of the convolution is the product of the Fourier–Galois images of the functions under its sign).

In the future, this creates prerequisites for the formation of convolutional neural networks with predetermined properties (i.e., neural networks possessing a given analog of the transfer function).

This step, however, requires the formation of appropriate sets of orthogonal basis functions.

Besides, it is appropriate to emphasize that the interest to the practical use of multivalued logics at present is connected not only with the improvement of digital signal processing, but also with the improvement of AI. In particular, as shown in [59], the biological prototype of AI: human intelligence—is not reducible to binary logic. At the same time, there is no doubt that the processing of signals by the human brain also corresponds to the above-mentioned symmetry property. This opens additional perspectives for using the results obtained in this work.

## 7. Conclusions

Thus, a method that allows us to reduce convolution operation to calculations modulo integer is proposed. In other words, the proposed approach allows us to bring the operation of reduced convolution computation to computations in Galois fields.

In this case, the number of discrete levels of functions under the convolution sign coincides with the number of discrete levels corresponding to the result of the convolution operation.

It makes sense to treat the convolution operation, in which the discretization character of the initial and resulting functions is preserved, as a reduced convolution.

The constructiveness of this approach is demonstrated on a concrete model example.

In constructing this method, RNS-based computations are considered as a special case of computations in finite algebraic rings, which creates prerequisites for its use in the transition to algebraic rings of different varieties.

Further development of this approach implies, among other things, the use of functions taking values corresponding to variables of multivalued logic.

The results are also of interest in terms of creating convolutional neural networks with predetermined properties.

**Author Contributions:** Conceptualization, Y.V. and I.S.; methodology, I.S.; software, A.K.; validation, Y.V.; formal analysis, D.M.; investigation, A.K.; resources, Y.V. and D.M.; data curation, Y.V.; writing—original draft preparation, I.S.; writing—review and editing, I.S. and Y.V.; visualization A.K. and D.M.; supervision, Y.V.; project administration, Y.V.; funding acquisition, Y.V., I.S. and D.M. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research has been funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. AP14870281).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author/s.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Goodman, J.W. *Introduction to Fourier Optics*, 3rd ed.; Roberts & Co: Englewood, CO, USA, 2005; p. 491.
2. Tyson, R.K. *Principles and Applications of Fourier Optics*; IOP Expanding Physics; IOP Publishing: Bristol, UK, 2014; p. 117.
3. Darlington, S. A History of Network Synthesis and Filter Theory for Circuits Composed of Resistors, Inductors, and Capacitors. *IEEE Trans. Circuits Syst.* **1984**, *31*, 3–13. [CrossRef]
4. Izadian, A. *Fundamentals of Modern Electric Circuit Analysis and Filter Synthesis: A Transfer Function Approach*; Springer International Publishing: Cham, Switzerland, 2019; p. 115. [CrossRef]
5. Gu, J.; Wang, Z.; Kuen, J.; Ma, L.; Shahroudy, A.; Shuai, B.; Liu, T.; Wang, X.; Wang, G.; Cai, J.; et al. Recent Advances in Convolutional Neural Networks. *Pattern Recognit.* **2018**, *77*, 354–377. [CrossRef]
6. Aghdam, H.H.; Heravi, J.E. *Guide to Convolutional Neural Networks*; Springer International Publishing: Cham, Switzerland, 2017; p. 282. [CrossRef]
7. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]
8. Monteiro, A.; Oliveira, M.; Oliveira, R.; Silva, T. Embedded Application of Convolutional Neural Networks on Raspberry Pi for SHM. *Electron. Lett.* **2018**, *54*, 680–682. [CrossRef]
9. Shichijo, S.; Nomura, S.; Aoyama, K.; Nishikawa, Y.; Miura, M.; Shinagawa, T.; Takiyama, H.; Tanimoto, T.; Ishihara, S.; Matsuo, K.; et al. Application of Convolutional Neural Networks in the Diagnosis of Helicobacter Pylori Infection Based on Endoscopic Images. *EBioMedicine* **2017**, *25*, 106–111. [CrossRef] [PubMed]
10. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861. [CrossRef]
11. Ketkar, N.; Moolayil, J. Convolutional Neural Networks. In *Deep Learning with Python*; Apress: Berkeley, CA, USA, 2021; pp. 197–242. [CrossRef]
12. Zhang, C.; Zhao, H.; Cao, M. Research on General Text Classification Model Integrating Character-Level Attention and Multi-Scale Features. In Proceedings of the 2021 10th International Conference on Computing and Pattern Recognition, Shanghai, China, 15–17 October 2021; pp. 183–187. [CrossRef]
13. McLaren, M.; Lei, Y.; Scheffer, N.; Ferrer, L. Application of Convolutional Neural Networks to Speaker Recognition in Noisy Conditions. In Proceedings of the Fifteenth Annual Conference of the International Speech Communication Association, Interspeech 2014, Singapore, 14–18 September 2014; pp. 686–690. [CrossRef]

14. Moldakhan, I.; Matrassulova, D.K.; Shaltykova, D.B.; Suleimenov, I.E. Some Advantages of Non-Binary Galois Fields for Digital Signal Processing. *Indones. J. Electr. Eng. Comput. Sci.* **2021**, *23*, 871. [CrossRef]
15. Vitulyova, E.S.; Matrassulova, D.K.; Suleimenov, I.E. New Application of Non-Binary Galois Fields Fourier Transform: Digital Analog of Convolution Theorem. *Indones. J. Electr. Eng. Comput. Sci.* **2021**, *23*, 1718. [CrossRef]
16. Suleimenov, I.E.; Vitulyova, Y.S.; Matrassulova, D.K. Features of Digital Signal Processing Algorithms Using Galois Fields GF(2n+1). *PLoS ONE.* **2023**, *18*, e0293294. [CrossRef] [PubMed]
17. Vitulyova, E.S.; Matrassulova, D.K.; Suleimenov, I.E. Construction of generalized Rademacher functions in terms of ternary logic: Solving the problem of visibility of using Galois fields for digital signal processing. *Int. J. Electron. Telecommun.* **2022**, 237–244. [CrossRef]
18. Matrassulova, D.K.; Vitulyova, Y.S.; Konshin, S.V.; Suleimenov, I.E. Algebraic Fields and Rings as a Digital Signal Processing Tool. *Indones. J. Electr. Eng. Comput. Sci.* **2022**, *29*, 206. [CrossRef]
19. Kuo, Y.-M.; Garcia-Herrero, F.; Ruano, O.; Maestro, J.A. RISC-V Galois Field ISA Extension for Non-Binary Error-Correction Codes and Classical and Post-Quantum Cryptography. *IEEE Trans. Comput.* **2022**, *72*, 682–692. [CrossRef]
20. Nazarov, L.E. Investigation of Noise Immunity of Optimal Symbol Reception of Frequency-Efficient Signals with Correction Coding in Non-Binary Galois Fields. *J. Commun. Technol. Electron.* **2023**, *68*, 960–965. [CrossRef]
21. Jagadeesh, H.; Joshi, R.; Rao, M. Group Secret-Key Generation Using Algebraic Rings in Wireless Networks. *IEEE Trans. Veh. Technol.* **2021**, *70*, 1538–1553. [CrossRef]
22. Alcayde, A.; Ventura, J.; Montoya, F.G. Hypercomplex Techniques in Signal and Image Processing Using Network Graph Theory: Identifying Core Research Directions [Hypercomplex Signal and Image Processing]. *IEEE Signal Process. Mag.* **2024**, *41*, 14–28. [CrossRef]
23. Suleimenov, I.E.; Vitulyova, Y.S.; Kabdushev, S.B.; Bakirov, A.S. Improving the Efficiency of Using Multivalued Logic Tools. *Sci Rep.* **2023**, *13*, 1108. [CrossRef]
24. Suleimenov, I.E.; Vitulyova, Y.S.; Kabdushev, S.B.; Bakirov, A.S. Improving the Efficiency of Using Multivalued Logic Tools: Application of Algebraic Rings. *Sci Rep.* **2023**, *13*, 22021. [CrossRef] [PubMed]
25. Tynymbayev, S.; Berdibayev, R.S.; Omar, T.; Gnatyuk, S.A.; Namazbayev, T.A.; Adilbekkyzy, S. Devices for Multiplying modulo Numbers with Analysis of the Lower Bits of the Multiplier. *Bull. Natl. Acad. Sci. Repub. Kazakhstan* **2019**, *4*, 38–45. [CrossRef]
26. Sayed-Ahmed, A.; Große, D.; Kühne, U.; Soeken, M.; Drechsler, R. Formal verification of integer multipliers by combining Gröbner basis with logic reduction. In Proceedings of the 2016 Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, Germany, 14–18 March 2016; pp. 1048–1053.
27. Munoz-Coreas, E.; Thapliyal, H. Quantum Circuit Design of a T-Count Optimized Integer Multiplier. *IEEE Trans. Comput.* **2019**, *68*, 729–739. [CrossRef]
28. Sousa, L.; Antao, S.; Martins, P. Combining Residue Arithmetic to Design Efficient Cryptographic Circuits and Systems. *IEEE Circuits Syst. Mag.* **2016**, *16*, 6–32. [CrossRef]
29. Hidenori, E.; Kiyoto, K.; Denki, C. Circuit for Modulo Multiplication and Exponentiation Arithmetic. Patent EP0801345B1, 16 October 2002.
30. Lablans, P. Multi-Value Digital Calculating Circuits, Including Multipliers. Patent US20060031278A1, 9 February 2006.
31. Irkhin, V.P.; Obukhov, A.N.; Gul'bin, S.S. G06F 7/49. Device for Modulo Addition and Subtraction of Numbers. Patent RU2145112C1, 27 January 2000.
32. Pisek, E.; Henige, T.M. Method and Apparatus for Efficient Modulo Multiplication. U.S. Patent 2009/0144353 A1, 4 June 2009.
33. Nakahara, H.; Sasao, T. A Deep Convolutional Neural Network Based on Nested Residue Number System. In Proceedings of the 2015 25th International Conference on Field Programmable Logic and Applications (FPL), London, UK, 2–4 September 2015; pp. 1–6. [CrossRef]
34. Valueva, M.V.; Nagornov, N.N.; Lyakhov, P.A.; Valuev, G.V.; Chervyakov, N.I. Application of the Residue Number System to Reduce Hardware Costs of the Convolutional Neural Network Implementation. *Math. Comput. Simul.* **2020**, *177*, 232–243. [CrossRef]
35. Jenkins, W.; Leon, B. The Use of Residue Number Systems in the Design of Finite Impulse Response Digital Filters. *IEEE Trans. Circuits Syst.* **1977**, *24*, 191–201. [CrossRef]
36. Aithal, G.; Bhat, K.N.H.; Sripathi, U. Implementation of Stream Cipher System Based on Representation of Integers in Residue Number System. In Proceedings of the 2010 IEEE 2nd International Advance Computing Conference (IACC), Patiala, India, 19–20 February 2010; pp. 210–217. [CrossRef]
37. Dubey, A.; Ahmad, A.; Pasha, M.A.; Cammarota, R.; Aysu, A. ModuloNET: Neural Networks Meet Modular Arithmetic for Efficient Hardware Masking. *ACR Trans. Cryptogr. Hardw. Embed. Syst.* **2021**, 506–556. [CrossRef]
38. Yassine, H.M. Fast Arithmetic Based on Residue Number System Architectures. In Proceedings of the 1991 IEEE International Symposium on Circuits and Systems (ISCAS), Singapore, 11–14 June 1991; Volume 5, pp. 2947–2950. [CrossRef]
39. Bos, J.W.; Friedberger, S. Fast Arithmetic Modulo 2^x p^y ± 1. In Proceedings of the 2017 IEEE 24th Symposium on Computer Arithmetic (ARITH), London, UK, 24–26 July 2017; pp. 148–155.
40. Torabi, Z.; Jaberipur, G.; Belghadr, A. Fast Division in the Residue Number System $\{2^n+1, 2^n, 2^n-1\}$ Based on Shortcut Mixed Radix Conversion. *Comput. Electr. Eng.* **2020**, *83*, 106571. [CrossRef]

41. Markov, V.T.; Mikhalev, A.V.; Nechaev, A.A. Nonassociative Algebraic Structures in Cryptography and Coding. *J. Math. Sci.* **2020**, *245*, 178–196. [CrossRef]
42. Sanam, N.; Ali, A.; Shah, T.; Farooq, G. Non-Associative Algebra Redesigning Block Cipher with Color Image Encryption. *Comput. Mater. Contin.* **2021**, *67*, 1–21. [CrossRef]
43. Liu, P.; Pan, Z.; Lei, J. Parameter Identification of Reed-Solomon Codes Based on Probability Statistics and Galois Field Fourier Transform. *IEEE Access* **2019**, *7*, 33619–33630. [CrossRef]
44. Huang, Q.; Tang, L.; He, S.; Xiong, Z.; Wang, Z. Low-Complexity Encoding of Quasi-Cyclic Codes Based on Galois Fourier Transform. *IEEE Trans. Commun.* **2014**, *62*, 1757–1767. [CrossRef]
45. Hazzazi, M.M.; Attuluri, S.; Bassfar, Z.; Joshi, K. A Novel Cipher-Based Data Encryption with Galois Field Theory. *Sensors* **2023**, *23*, 3287. [CrossRef]
46. Asif, M.; Asamoah, J.K.K.; Hazzazi, M.M.; Alharbi, A.R.; Ashraf, M.U.; Alghamdi, A.M. A Novel Image Encryption Technique Based on Cyclic Codes over Galois Field. *Comput. Intell. Neurosci.* **2022**, *2022*, 1912603. [CrossRef]
47. Hiasat, A.; Sousa, L. On the Design of RNS Inter-Modulo Processing Units for the Arithmetic-Friendly Moduli Sets $\{2^{n+k}, 2^n - 1, 2^{n+1} - 1\}$. *Comput. J.* **2019**, *62*, 292–300. [CrossRef]
48. Ha, J.; Kim, S.; Choi, W.; Lee, J.; Moon, D.; Yoon, H.; Cho, J. Masta: An HE-Friendly Cipher Using Modular Arithmetic. *IEEE Access* **2020**, *8*, 194741–194751. [CrossRef]
49. Hu, K.; Zhang, D.; Xia, M. CDUNet: Cloud Detection UNet for Remote Sensing Imagery. *Remote Sens.* **2021**, *13*, 4533. [CrossRef]
50. Li, Z.; Shen, H.; Weng, Q.; Zhang, Y.; Dou, P.; Zhang, L. Cloud and Cloud Shadow Detection for Optical Satellite Imagery: Features, Algorithms, Validation, and Prospects. *ISPRS J. Photogramm. Remote Sens.* **2022**, *188*, 89–108. [CrossRef]
51. Suleimenov, I.; Vitulyova, Y.; Bakirov, A. Hybrid Number Systems: Application for Calculations in Galois Fields. In Proceedings of the 2022 3rd Asia Conference on Computers and Communications (ACCC), Shanghai, China, 16–18 December 2022; pp. 126–130. [CrossRef]
52. Sadeghi, A.; Shiri, N.; Rafiee, M.; Rahimi, P. A Low-Power Pseudo-Dynamic Full Adder Cell for Image Addition. *Comput. Electr. Eng.* **2020**, *87*, 106787. [CrossRef]
53. Özkilbaç, B. Implementation and Design of 32 Bit Floating-Point ALU on a Hybrid FPGA-ARM Platform. *J. Brill. Eng.* **2019**, *1*, 26–32. [CrossRef]
54. Yan, B.; Chan, P.W.; Li, Q.; He, Y.; Shu, Z. Dynamic Analysis of Meteorological Time Series in Hong Kong: A Nonlinear Perspective. *Intl J. Climatol.* **2021**, *41*, 4920–4932. [CrossRef]
55. Nonejad, N. An overview of dynamic model averaging techniques in time-series econometrics. *J. Econ. Surv.* **2021**, *35*, 566–614. [CrossRef]
56. Kumar, G.; Singh, U.P.; Jain, S. Hybrid Evolutionary Intelligent System and Hybrid Time Series Econometric Model for Stock Price Forecasting. *Int. J. Intell. Syst.* **2021**, *36*, 4902–4935. [CrossRef]
57. Chatterjee, A.; Bhowmick, H.; Sen, J. Stock Price Prediction Using Time Series, Econometric, Machine Learning, and Deep Learning Models. In Proceedings of the 2021 IEEE Mysore Sub Section International Conference (MysuruCon), Mysuru, India, 16–17 October 2021; pp. 289–296. [CrossRef]
58. Rahman, M.; Shakeri, M.; Khatun, F.; Tiong, S.K.; Alkahtani, A.A.; Samsudin, N.A.; Amin, N.; Pasupuleti, J.; Hasan, M.K. A Comprehensive Study and Performance Analysis of Deep Neural Network-Based Approaches in Wind Time-Series Forecasting. *J. Reliab. Intell. Environ.* **2023**, *9*, 183–200. [CrossRef]
59. Gabrielyan, O.A.; Vitulyova, E.; Suleimenov, I.E. Multi-valued logics as an advanced basis for artificial intelligence. *Wisdom* **2022**, *1*, 170–181. [CrossRef]