applied sciences

MDPI

Article

Prospects for the Use of Quasi-Mersen Numbers in the Design of Parallel-Serial Processors

Aruzhan Kadyrzhan [1], Kaisarali Kadyrzhan [2], Akhat Bakirov [1,2,*] and Ibragim Suleimenov [2]

[1] Department of Telecommunication Engineering, Institute of Communications and Space Engineering, Gumarbek Daukeev Almaty University of Power Engineering and Communications, Almaty 050040, Kazakhstan; aru.kadyrzhan@gmail.com
[2] National Engineering Academy of the Republic of Kazakhstan, Almaty 050010, Kazakhstan; kaisarali1997ss@gmail.com (K.K.); esenych@yandex.ru (I.S.)
* Correspondence: axatmr@mail.ru

Abstract: It is shown that a serial-parallel processor, comparable in bit capacity to a 16-bit binary processor, can be implemented based on an algorithm built on the residue number system, a distinctive feature of which is the use of the first four quasi-Mersenne numbers, i.e., prime numbers representable as $p_k = 2^k + 1,\ k = 1, 2, 3, 4$. Such a set of prime numbers satisfies the criterion $2p_1p_2p_3p_4 + 1 = P$, where P is also a prime number. Fulfillment of this criterion ensures the possibility of convenient use of the considered RNS for calculating partial convolutions developed for the convenience of using convolutional neural networks. It is shown that the processor of the proposed type can be based on the use of a set of adders modulo a quasi-Mersenne number, each of which operates independently. A circuit of a modulo $2^k + 1$ adder is proposed, which can be called a trigger circuit, since its peculiarity is the existence (at certain values of the summed quantities) of two stable states. The advantage of such a circuit, compared to known analogs, is the simplicity of the design. Possibilities for further development of the proposed approach related to the use of the digital logarithm operation, which allows reducing the operations of multiplication modulo $2^k + 1$ to addition operations, are discussed.

Keywords: quasi-Mersen numbers; trigger adder; effective bit depth of adder; computing performance; parallel-serial computing; residue number system (RNS)

check for updates

Academic Editor: Alessandro Lo Schiavo

Received: 4 September 2024
Revised: 7 January 2025
Accepted: 7 January 2025
Published: 13 January 2025

Citation: Kadyrzhan, A.; Kadyrzhan, K.; Bakirov, A.; Suleimenov, I. Prospects for the Use of Quasi-Mersen Numbers in the Design of Parallel-Serial Processors. *Appl. Sci.* **2025**, *15*, 741. https://doi.org/10.3390/app15020741

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

At present, there is considerable interest in increasing the productivity of computing equipment [1–4]. One approach to solving this problem is the use of parallel-sequential calculations [5,6], which are implemented, among other methods, using the residue number system (RNS) [7,8]. In this case, each integer x is represented as $(x_1, x_2, \ldots, x_n)$, where $x_i = x \bmod m_i$, $m_i; i = 1, 2, \ldots n$ are integers that form the RNS [9]. The moduli m_i are generally assumed to be pairwise coprime.

The advantage of RNS is its ability to independently operate with individual remainders x_i, in particular, $z_i = x_i + y_i$, if $z = x + y$ (similarly for multiplication). Such properties allow the move from sequential to parallel-sequential computations, since, when $x < M = m_1m_2 \ldots m_n$, each set of residues $(x_1, x_2, \ldots, x_n)$ represents exactly one integer x. Consequently, as long as this inequality is satisfied, computations using integers can be reduced to computations in RNS. From the perspective of computer technology, this limitation is not significant. In any case, the range within which standard processors operate is also limited (e.g., 16-bit adders).

The question of choosing a concrete set of moduli m_i from the perspective of computer technology is not trivial. In particular, in the case that all numbers m_i are prime, i.e., $m_i = p_i$, the computations corresponding to each component x_i of the number x are de facto performed in the Galois field $GF(p_i)$. Currently, numerous different types of electronic circuits are known that are designed to perform addition and multiplication operations in such fields, i.e., adders and multipliers modulo a prime number [10–13]. They differ significantly from one another in their design, which, moreover, significantly depends on the choice of the modulus, as is proven, in particular, by report [14]. Moreover, it was noted in [15] that different Galois fields have different specifics.

Consequently, there is every reason to assert that the choice of a set of moduli m_i should be determined, first, by the specifics of the problems being solved. The technical implementation of this approach is facilitated by the widespread use of microcircuits with a reconfigurable logical structure [16–18]. Accordingly, it is permissible to raise the question of developing calculators oriented towards solving a certain range of problems.

In [19], it was shown that for calculating digital convolutions, it is convenient to use an RNS corresponding to a set of prime numbers p_i, such that $p_1 p_2 \dots p_n + 1 = P$, where P is also a prime number. In this case, it becomes possible to reduce the discrete scale corresponding to the result of calculating the convolution to a discrete scale corresponding to the original function, without using fractional values.

This paper shows that a serial-parallel processor, comparable in bit depth to a 16-bit binary processor, can be implemented based on an algorithm built on the residue number system. The distinctive feature of this processor is the use of the first four quasi-Mersenne numbers, i.e., prime numbers representable as $p = 2^k + 1$.

The basis of such a processor is an adder modulo a quasi-Mersenne prime number. A specific electronic circuit of such an adder is presented, which can be called a trigger, since it can be in two different states. Its advantage over known analogs is the simplicity of the design.

It should be noted that Mersenne primes are currently widely used in information technology. In particular, these numbers are used to generate pseudorandom numbers [20]. Specifically, a pseudorandom number generator called the Mersenne Twister, developed in 1997 by Japanese scientists M. Matsumoto and T. Nishimura [21], is directly based on the use of Mersenne numbers.

Some reports demonstrating the expediency of using such numbers for data transmission are presented in the current literature [22,23], etc. Mersenne codes 3, 7, and 11 provide greater noise immunity and an increased probability of correct detection in radar channels, offering a potential alternative to Barker codes for noise-resistant data transmission in radio channels [24]. Modified RSA algorithm, using fixed Mersenne prime numbers, improves encryption in medical ultrasound imaging while maintaining adequate elapsed time for image transfer [25].

From the perspective of performing calculations modulo an integer, these numbers are particularly convenient because the multiplication of a number a written in binary form by two modulo the Mersenne number is reduced to cyclic permutation of characters. For example, for calculations in the field $GF(7)$, the next equality is true

$$2 \cdot a_2 a_1 a_0 =_{(7)} a_1 a_0 a_2 \quad (1)$$

Relation (1) reflects the uniqueness of Mersenne primes. Quasi-Mersenne numbers are also unique, which is demonstrated in report [26]. Their uniqueness is due to the fact that the operation of digital logarithm, the study of which has recently received very serious attention [27–29], has a pronounced specificity in the case under consideration. Namely, the operation of digital logarithm allows us to reduce the operation of multiplication in a

Galois field to the operation of addition. However, this operation is applicable only to non-zero elements of Galois fields. In Galois fields $GF(p)$, corresponding to quasi-Mersenne numbers $p = 2^k + 1$, the number of non-zero elements is obviously equal to 2^k, i.e., the multiplication operation (after using the digital logarithm operation) is de facto reduced not simply to an addition operation, but to an addition operation in terms of ordinary binary logic, easily implemented on the existing element base (addition modulo 2^k is operationally reduced to discarding the most significant digits in the binary representation of a number).

It is appropriate to emphasize that the uniqueness of quasi-Mersenne numbers has not previously received the attention that they deserve. Evidence for this is provided in our recent report [26], where it was shown that it is possible to implement a simple and convenient method of digital logarithm specifically for such fields.

We also note that the approach using a specific Galois field (or varieties of such fields) is not fundamentally new. Modern binary computing technology is de facto built on the use of a specific field, GF(2). Consequently, the question of developing computing technology that uses the specifics of a particular type of Galois fields in a certain sense corresponds to the existing tradition.

2. Results

2.1. RNS Used

The RNS used is formed by the following quasi-Mersenne numbers (Table 1) and the prime number 2.

Table 1. Quasi-Mersenne numbers used.

k	1	2	4	8
$P = 2^k + 1$	3	5	17	257

Equality holds

$$2 \cdot 3 \cdot 5 \cdot 17 \cdot 257 + 1 = 131,071 \quad (2)$$

It is significant that the number 131,071 is prime. This means that this set of numbers satisfies the criterion formulated in [19], i.e.,

$$p_1 p_2 \ldots p_n + 1 = P \quad (3)$$

where P is also a prime number.

This criterion is very important, in particular, from the perspective of performing operations that are used by convolutional neural networks. As noted in [20], the following problem arises when digital convolutions are being used. The discretization scale of the convolution result may differ from the discretization scale of the original signal. To make the discretization step of these scales the same, one can use, for example, the operation of rounding to an integer. From a computational point of view, however, this is not convenient, since it is more convenient for any electronic devices to perform calculations in integers. As shown in [19], if criterion (1) is satisfied, then there is a simple and convenient way to reduce the above scales to a single format in which only calculations in integers are used. Thus, the fulfillment of criterion (1) for the first quasi-Mersenne numbers is additional evidence of their uniqueness.

In addition, $\log_2 131,071 \approx 16.999978$. This means that the range of integers with which a processor built on the RNS under consideration can work with exceeds a typical 16-bit processor in terms of effective bit depth.

This fact also testifies in favor of the uniqueness of the set of prime numbers presented in Table 1. Calculations using the RNS corresponding to such a system correspond in their capabilities to one of the existing standards, but, as will become clear, there is an opportunity to move on to serial-parallel calculations. In this regard, it is appropriate to note that any existing technical standard (including those in the field of computer technology) is deliberately oriented towards specific quantitative indicators. The simplest example is the existing television standards, which fix the number of lines and the number of pixels in a line [30,31]. It is important that these standards can have a fairly large variety, as demonstrated by the standards corresponding to information display systems. Similarly, based on considerations of convenience and efficiency, a set of standards corresponding to computer technology can be implemented. In this regard, it is appropriate to note once again that the existing digital technology is de facto oriented towards a very specific standard associated with the choice of a specific Galois field $GF(2)$. It is precisely in this field that existing binary adders and other elements of binary logic are built.

This field has well-defined specifics, as well as well-defined advantages. This, however, does not exclude the possibility of using other Galois fields. Based on the analogy with existing processors, it seems advisable to begin by developing an adder modulo quasi-Mersenne numbers. For the processor under consideration, such an adder plays the same role as the classical binary adder in the most common processors. It is also important that the circuit for the developed adder be oriented toward the use of components that correspond to binary logic, since these are the ones that are produced industrially. Thus, this paper substantiates the possibility of using another standard of computer technology, one which has the same computing capabilities as existing 16-bit processors but allows for serial-parallel computations using a well-defined RNS. This approach is particularly convenient for computations corresponding to convolutional neural networks due to the fulfillment of criterion (2). In addition, as will become clear, it is the use of quasi-Mersenne numbers that allows us to propose fairly simple computing devices based on standard elements developed for binary logic.

2.2. Basic Scheme of Operation of the Calculators of the Proposed Type

The operational scheme of the proposed type of calculators is based on the following properties of algebraic rings, of which RNS is a special case, as emphasized in the report [19]. Such rings R decompose into a direct sum of ideals r_i

$$R = r_1 + r_2 + \cdots + r_n \tag{4}$$

Each of the ideals r_i is generated by its complementary idempotent element e_i

$$r_i = Re_i, \tag{5}$$

The elements e_i cancel each other out

$$e_i e_j = 0,\ i \neq j;\ e_i e_i = e_i \tag{6}$$

The sum of these elements is equal to the unit of the ring R

$$\sum_i e_i = 1 \tag{7}$$

Examples of such rings are given by rings of residue classes modulo a number p, representing the product of prime numbers $p = p_1 p_2 \ldots p_n$. In this case, idempotent elements can be formed according to the rule used, in particular, in [20]

$$e_i = \alpha_i \prod_{j \neq i} p_j \tag{8}$$

where α_i is an integer that is not a multiple of p_i. The choice of these numbers is based on the condition

$$e_i e_i = 1 \tag{9}$$

It is obvious that by construction we have

$$e_i p_i \equiv 0(P) \tag{10}$$

since any product of the form (10) contains the factor $p = p_1 p_2 \dots p_N$.

With the choice of integers α_i made, it also holds

$$e_1 + e_2 + \dots + e_N \equiv 1(P) \tag{11}$$

In the case under consideration, an arbitrary element of the ring R can be represented as

$$u = e_1 u_1 + e_2 u_2 + \dots + e_N u_N \tag{12}$$

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

The convenience of representing (12) in calculations is that the elements u_i are multiplied independently, and the multiplication is performed modulo p_i. Indeed, since e_i are mutually annihilating idempotent elements, the product of two numbers $u^{(1)}$ and $u^{(2)}$ is expressed as

$$u^{(1)} u^{(2)} = e_1 u_1^{(1)} u_1^{(2)} + e_2 u_2^{(1)} u_2^{(2)} + \dots + e_N u_N^{(1)} u_N^{(2)} \tag{13}$$

A similar formula is valid for the addition operation, which was also used in report [19]

$$u^{(1)} + u^{(2)} = e_1 \left(u_1^{(1)} + u_1^{(2)} \right) + e_2 \left(u_2^{(1)} + u_2^{(2)} \right) + \dots + e_N \left(u_N^{(1)} + u_N^{(2)} \right) \tag{14}$$

Formulas (13) and (14) allow us to implement the following circuits of the multiplier and adder modulo the number $p = p_1 p_2 \dots p_N$, where p_i are prime numbers (Figures 1 and 2, respectively). Common to these circuits are elements (1_i) and (2_i), which ensure the reduction of the original numbers to values modulo each of the prime numbers p_i. From the algebraic-theoretical perspective, such elements correspond to the transition to the use of calculations in Galois fields complementary to the ideals r_i. Blocks (3_i) perform multiplication or addition modulo prime numbers p_i, i.e., multiplication or addition in terms of Galois fields $GF(p_i)$. As follows from Formulas (13) and (14), the multiplication and addition of individual "components" of the original number can indeed be performed independently. Block (4) returns the calculation result to the number in the original representation in accordance with Formula (12).

Let us emphasize that formally the calculations using the presented schemes are performed modulo the integer p. However, if the result of addition and multiplication does not exceed p, it coincides with the result of calculations using ordinary arithmetic operations. For any physically implemented processor, the range in which the result of calculations fits is finite. This is particularly true for existing 16-bit processors. As follows from Table 1, the product $2 \cdot 3 \cdot 5 \cdot 17 \cdot 257 = 131,070$ corresponds to approximately the same range. It is this fact that allows calculations to be performed in accordance with the proposed schemes without changing the result in the traditional "arithmetic" sense.

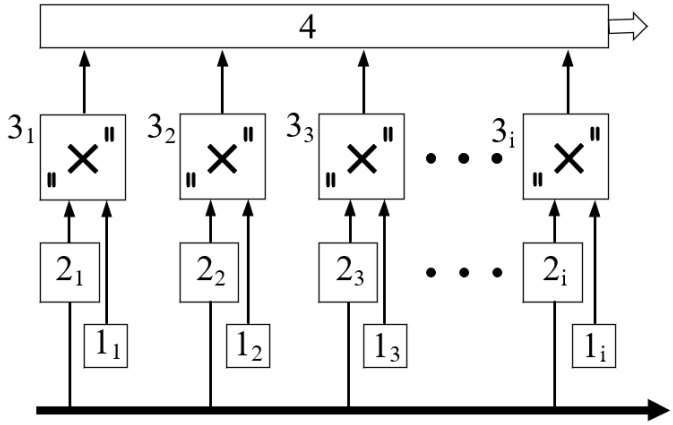


Figure 1. General diagram of a multiplier built based on Formula (13).

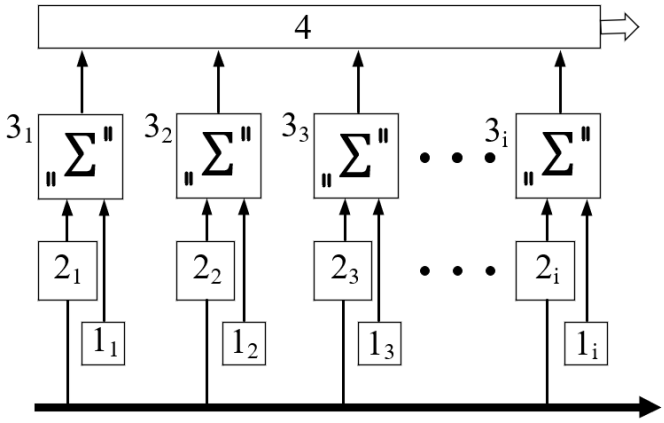


Figure 2. General diagram of an adder built based on Formula (14).

The circuits in Figures 1 and 2 are applicable to any numbers $p = p_1 p_2 \dots p_N$, where p_i are prime numbers. However, there is a significant nuance. As will become clear, the degree of complexity of electronic circuits that perform calculations in different Galois fields varies. One of the simplest types of such calculators corresponds to the case where all the prime numbers in the above product are representable as $p_i = 2^{k_i} + 1$, i.e., quasi-Mersenne numbers. For this reason, we focus on the use of such numbers in advanced processor technology, especially for that which ensures the implementation of convolutional neural networks.

2.3. Scheme of a Trigger Adder Modulo a Quasi-Mersenne Number

This section considers a specific design of adders, which have numbers (3_i) in the diagram in Figure 2. The proposed design is common to all adders modulo the quasi-Mersenne number.

The proposed adder circuit differs significantly from the modulo integer adder circuits reported in the current literature [32,33], including patents [34,35]. The adder circuits known in the literature are built entirely on direct logical operations, excluding the use of more than one stable state of the circuit. The paper [32] presents a modulo 2^n adder using the carry-save method. The circuit contains 48 logical elements and provides high performance for residual class systems. The paper [33] describes a unified modulo adder/subtractor designed for arbitrary modules. Its architecture uses 58 elements and a universal algorithm to improve hardware efficiency. Patent [34] describes a modular binary adder circuit that uses field-effect transistors to minimize computational delays. This patent includes 48 elements to ensure high-speed operation. Patent [35] describes a circuit that

uses iterative calculations to obtain a residual value through multi-step addition, which significantly improves efficiency in systems that handle large numbers. This circuit uses 56 elements, including multiple operation units to improve accuracy and speed.

The fundamental difference between the proposed circuit and the known ones is that it is essentially an analog of a trigger, i.e., it is capable of being in two different stable states. This allows us to significantly reduce the number of logical elements used, as shown below. It is also significant that such circuits, which can be called trigger circuits, are quite simple to implement, especially when the addition is performed modulo a quasi-Mersenne number.

The circuit diagram of the proposed adder is shown in Figure 3. It includes two sets of conventional single-digit adders (1_i) and (2_i), where $i = 1, 2, \ldots, k+1$, logical AND elements (3) and (4), and a rectangular pulse generator (5).

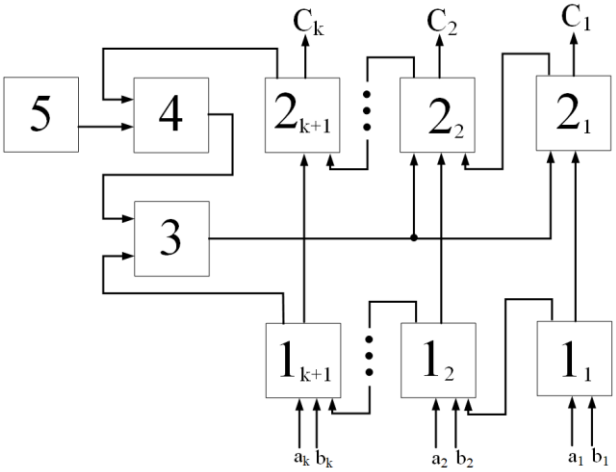


Figure 3. Circuit diagram of a modulo $2^k + 1$ trigger adder.

The circuit is designed for summation modulo $2^k + 1$ of numbers represented in binary form and lying in the range

$$0 \leq a \leq 2^k \tag{15}$$

It is thus assumed that, before the summation operation, the original numbers have already been reduced to a value modulo the number over which the summation is performed. Thus, this circuit performs summation in the field $GF\left(2^k + 1\right)$. In particular, this means that the largest number that this circuit operates, when represented in binary form, is written as

$$a_{max} = \underbrace{100\ldots0}_{k} \tag{16}$$

In particular, this means that when summation is performed in the Galois field $GF(17)$, it is assumed that the inputs of the adder are fed signals corresponding to numbers in the range $0 \leq a \leq 16$.

The circuit operates as follows. Binary signals corresponding to the representation of integers a and b in binary format are fed to the inputs a_i *and* b_i. The first line of adders (1_i) ensures the summation of these numbers in the usual binary representation. The second line of adders (2_i) is designed to reduce the sum to the sum modulo $2^k + 1$.

Looking ahead, we note that the part of the circuit involving the second line of adders can also be used to calculate the components of the original numbers according to the rule $x_i = x \ mod \ m_i$. As noted above, it is assumed that before performing the operations implemented using the circuit in Figure 3, the numbers being summed have already been reduced to the format $x_i = x \ mod \ m_i$.

Let us consider the operation of the circuit in Figure 3 in more detail, taking into account that the nature of its functioning depends on the specific value of the summation result, which corresponds to the signals generated at the output of adders (1_i). In particular, the case when two numbers corresponding to the binary representation of the form (4) are summed is exceptional. The logical AND element (3) takes this exceptional case into account. When two numbers of the form (16) are summed, a logical one is formed at the output of the adder carry bit (1_{k+1}). Accordingly, a logical one is also formed at the output of element (3), which is transferred to the second gatherings of adders in the second line with numbers from 1 to k. The adder with number $k+2$ is absent in the second line. Consequently, transferring a logical one to the inputs of the adders indicated above effectively means that the number 2^{k+1} is subtracted from the number formed at the output of the first line of adders, and a number that contains exactly k ones in binary representation is added.

We emphasize that the circuit in Figure 3 includes not k, but $k+1$ adders. This is necessary because there is an exceptional case when the output of the adder (2_{k+1}) differs from one. This case corresponds to the maximum number that can arise in calculations within the Galois field corresponding to a quasi-Mersenne number, i.e., a number that in binary representation has the form (16). Accordingly, when two numbers of the form (16) are summed modulo 2^k+1, the number 2^k+1 should be subtracted from the result of direct summation. This corresponds to subtracting the number 2^{k+1}, which is achieved by the absence of an adder numbered $k+2$ in the circuit under consideration, followed by the addition of the number $2^{k+1}-2^k-1=2^k-1$. In binary representation, the last number is displayed as follows: The number 2^k in binary form is written similarly to Formula (16)

The difference $2^{k+1}-2^k=2^k$ is written as

$$2^k = 1\underbrace{00\ldots0}_{k} \tag{17}$$

Therefore, the number $q=2^{k+1}-2^k-1$ is written as

$$q = 2^{k+1}-2^k-1 = 0\underbrace{11\ldots1}_{k} \tag{18}$$

This formula shows that technically, bringing the summation result to a value modulo 2^k+1 in the case under consideration essentially involves adding ones to all digits from the first to the digit with number k. This is the operation performed by the circuit in Figure 3, or more precisely, by the adders in the second line (2_i). It is appropriate to emphasize that in this circuit, additional ones are fed to the inputs of adders with numbers 1 through k, but are not fed to the adder numbered $k+1$.

As an example, let us consider the case when summation is performed modulo $2^2+1=5$. In this case, both lines (1_i) and (2_i) of the adder in Figure 3 contain three single-digit adders. The maximum value of the numbers being summed is 4, or 100 in binary representation. Note again that in this case, the most significant digit with the number $k+1$ (in this case, with the number 3), is used only to represent this special case. When calculating in the Galois field $GF(5)$, the two least significant digits are sufficient to represent the remaining elements. The sum of two numbers, 4, is equal to 8, or 1000 in binary representation. By discarding the most significant digit as a result of summation and adding the number 3 (written as 11 in binary representation), we exactly subtract the number 5, which gives the correct answer when summing modulo 5, i.e., 3.

Thus, element (3) allows us to obtain an adequate answer in the special case considered above. All other summation results can be classified as follows.

$$z : \begin{cases} 0 \le z < 2^k \\ z = 2^k \\ 2^k < z \le 2^{k+1} - 1 \end{cases} \tag{19}$$

In relation to the modulo 5 adder, this classification looks like this:

$$z : \begin{cases} 0 \le z < 4 \\ z = 4 \\ 4 < z \le 7 \end{cases} \tag{20}$$

In the cases corresponding to the first line of Formula (19), the second line of adders does not perform any actions. Integers lying in the range $0 \le z < 2^k$ do not require reduction to a value modulo 2^{k+1}. In this case, the variable formed at the output of the adder (2_{k+1}) is zero. It is non-zero in the cases corresponding to the second and third lines of Formula (5). The difference between these cases is that the number $z = 2^k$ does not require reduction modulo 2^{k+1}, and the numbers corresponding to the range $2^k < z \le 2^{k+1} - 1$—vice versa.

Reducing a number in the above range $2^k < z \le 2^{k+1} - 1$ to a number modulo 2^{k+1} means that the number 2^{k+1} should be subtracted from the given number and, as in the case considered above, adding a number that in binary representation contains exactly k ones, i.e., adding the number $q = 2^{k+1} - 2^k - 1$.

Let us emphasize once again that for the binary representation of the number 2^{k+1}, $k + 2$ binary digits are required. For example, the number 8 in binary representation is 1000. The second line of adders in the circuit in Figure 3 contains $k + 1$ single-digit adders. This means that the number 2^{k+1} will be automatically subtracted when it is used, even when a logical one is formed at the output of the carry bit of the adder with the number $k + 1$.

The addition of the number $q = 2^{k+1} - 2^k - 1$, composed in binary representation from k ones, according to the circuit in Figure 3 is carried out in a manner similar to that discussed above.

The carry-out output of the single-digit adder (2_{k+1}) is connected by feedback to adders (2_i), $i = 1, 2.., k$ via the "logical AND" element (4). Consequently, the number q, given by Formula (16), will be added to the result of the primary summation (the first line of adders) if a logical one is formed at the output of the adder (2_{k+1}) or at the output of the generator (5).

The above feedback forms an analog of a trigger. Namely, if the result of the summation lies in the range $2^k < z \le 2^{k+1} - 1$, then the circuit can be in two stable states. Indeed, adding the number q results in the formation of a logical one at the carry-out output of the adder (2_{k+1}), which supports summation with the above number q. This represents one of the two stable states. The second stable state occurs when a logical zero is formed at the output of the adder carry bit (2_{k+1}). In this case, the number q is not added to the result of the primary summation.

The generator (5) ensures switching between the two specified stable states. It transfers the circuit to the state "add the number q to the primary summation result". This reduces the result of the primary summation to the sum modulo $2^k + 1$.

An exception arises when the result of the primary summation corresponds to Formula (16) or the second line in Formula (19). In these cases, it should also remain unchanged (for example, the number 4 does not require reduction modulo 5). This is also ensured by the circuit in Figure 3. To prove this, consider the following summation result

$$1\underbrace{00\ldots0}_{k} + 0\underbrace{11\ldots1}_{k} = 1\underbrace{11\ldots1}_{k} \tag{21}$$

This result does not lead to the formation of a logical unit at the input of the adder bit carry (2_{k+1}). Therefore, after switching off the generator (5), a return to the initial state corresponding to the number $1\underbrace{00\ldots0}_{k}$ will be performed.

Thus, in all three cases corresponding to Formula (19), the proposed circuit actually provides summation modulo $2^k + 1$. Let us consider the results of testing the proposed circuit.

It should also be noted that the circuit in Figure 3 clearly demonstrates the advantages of using quasi-Mersenne numbers, as well as the possibility of generalizing the proposed approach to summation modulo other numbers. Namely, if we use the algebraic scheme for reducing the direct summation result to the summation result modulo, then depending on the value of the result, it is necessary to ensure that a certain value is subtracted from it, with a coefficient equal to 1 or 0. Taking into account the discarding of the most significant digit, this means that a number equal to the corresponding difference should be added to the summation result. With regard to quasi-Mersenne numbers, this is the number specified by Formula (18). With the same success, we can use other numbers whose binary representation contains k digits. The transition to using other numbers is reduced to changing the nature of the connections between the output of element (3) and the inputs of the adders in the second line in the circuit in Figure 3. However, when quasi-Mersenne numbers are used, this method corresponds to the possibility of feeding the same signal corresponding to a logical unit to the second inputs of the adders in the second line (Figure 3). While this may not be fundamentally important from the perspective of implementing electronic circuits, this does not diminish the convenience of using prime numbers of the type under consideration for building calculators designed to perform digital convolution operations and similar tasks. There is another significant argument in favor of using quasi-Mersenne numbers, considered in the Section 3. As shown in [26], for such numbers, it is quite simple to implement the digital logarithm operation, which allows us to reduce multiplier circuits in Galois fields to adder circuits in terms of ordinary binary logic.

2.4. Test Result of the Proposed Trigger Adder Circuit

Initial testing of the proposed schemes was carried out in the Proteus program. Figure 4 shows the results of testing the proposed adder circuit in Proteus for the case of the quasi-Mersenne number $2^2 + 1 = 5$. Figure 5 shows a similar result for a processor that provides summation modulo $2^4 + 1 = 17$. It is evident that the circuit operates in full accordance with the algorithm described above.

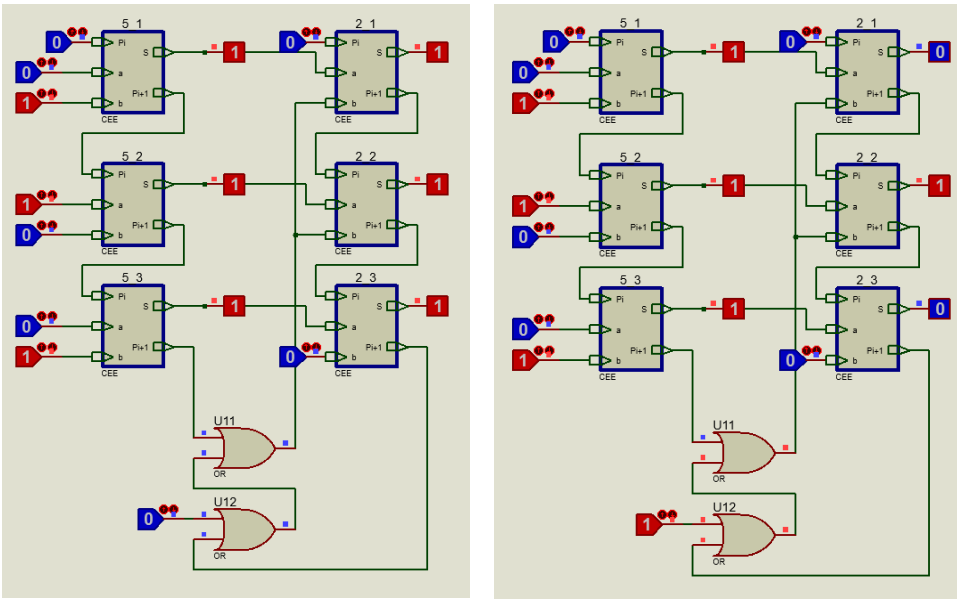
The circuits in Figures 4 and 5 fully correspond to the circuit in Figure 3, or rather, are its specifications. These figures use standard notations for adders and logical AND elements. Figures 4a and 5a correspond to the case when a logical zero is formed at the output of the rectangular pulse generator, which ensures the transition of the circuit from one stable state to another, and Figures 4b and 5b correspond to the case when a logical one is formed at its output. The circuit shows that, when a logical one is formed at the output of the specified generator, the primary result of summation is reduced to the value of the sum modulo the number 5 (Figure 4) or 17 (Figure 5).

To test the functionality and correctness of the project written in VHDL, Testbench is used, which is a simulation environment.

To test the logic circuit, the input data consists of 5-bit binary numbers for the modulo 17 adder and 3-bit binary numbers for the modulo 5 adder. These adders were designed using Verilog. The program listing is attached in the Supplementary Materials.

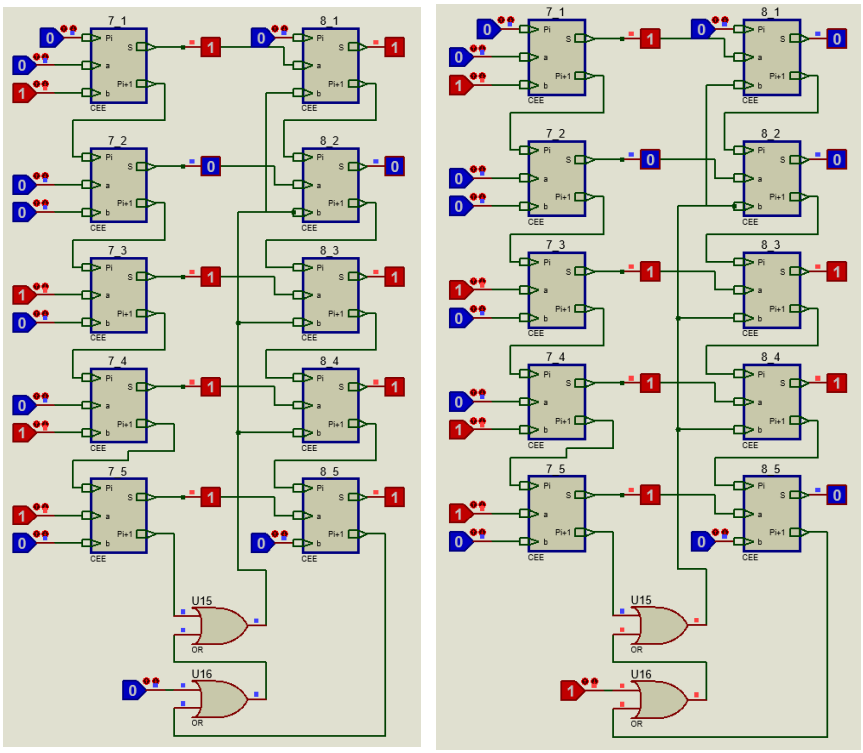
The program for conducting testbench iterates over input binary 5-bit numbers from 0 to 31 (i, j) inside the program cycle. If the sum of the input binary numbers is greater than or equal to 17 (if sum_ij $\geq$ 17), the U14_Control input of the generator is activated.

The results of the circuit's operation are visible on the timing diagram of the ModelSim simulator (Figure 6).



(**a**) (**b**)

Figure 4. Illustration of the existence of two stable states in the circuit of a trigger adder of the proposed type; adder modulo 5, example of adding two numbers 5 and 2; (**a**)—state before reducing the result to a sum modulo 5; (**b**)—after.



(**a**) (**b**)

Figure 5. Illustration of the existence of two stable states in the circuit of a trigger adder of the proposed type; adder modulo 17, example of adding two numbers 20 and 9; (**a**)—state before the result is transferred to the sum modulo 5; (**b**)—after.

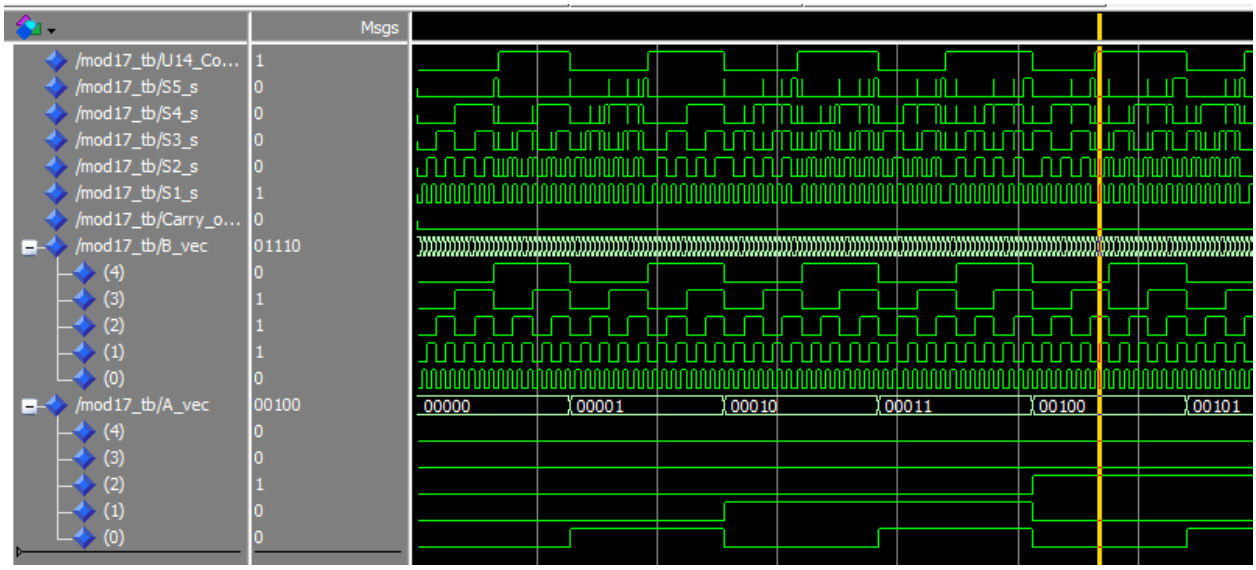


Figure 6. ModelSim testbench result (mod17).

3. Discussion

The proposed scheme for the adder modulo a quasi-Mersen number allows for the future to implementation of a processor that performs serial-parallel calculations due to the built in algorithm, built on a very specific RNS. As outlined in Section 2, the fact that this RNS is built on quasi-Mersen numbers allows to implement a relatively simple (compared with existing analogs) adder scheme. The basis of this processor is a set of adders in the fields $GF\left(2^k+1\right)$, each of which operates independently.

The simplest architecture of this processor can be implemented by analogy with existing processors, in which the basic operation is addition. The difference is that all numbers inputted into the calculator are initially converted into a format corresponding to the RNS of the proposed type. Consequently, further operations with each "component" of each number are carried out independently. Specifically, this means that the performance of such a processor is minimally affected by the units that provide the reduction of a number to a set of numbers modulo the first quasi-Mersenne numbers (the same applies to the inverse operation). Performance is determined solely by the adder itself, and the limiting factor (in terms of calculation speed, etc.) is associated with the adder containing the maximum number of elements (i.e., the adder modulo 257). This number corresponds to an 8-bit adder, while the adder as a whole corresponds to a 16-bit adder in terms of performance.

The prospects for the widespread use of a processor of this type may still be debatable; however, it is possible to indicate a very specific range of problems for which this processor is either optimal or close to optimal. For example, this applies to the calculation of partial convolutions, considered in [19].

There is no need to emphasize that the development of devices that calculate digital convolutions is becoming increasingly relevant. In particular, this is due to the increasingly widespread use of convolutional neural networks [36–38], which are used, among other things, for primary data processing by onboard computing systems installed on devices for various purposes. In particular, the development of onboard computing systems that perform primary image processing during remote sensing of the Earth is relevant. Specifically, it is necessary to exclude the transmission of images in cases where the surface is covered by cloud cover [39,40].

To solve such problems, it is advisable to use a processor of the proposed type precisely because it operates independently with different components of an integer. Consequently, it can be used both for standard calculations and for calculations oriented towards the use of partial convolutions, as described in [19], which is specially designed for the implementation of convolutional neural network algorithms.

Examples from other fields can also be provided, in which the proposed approach may be in demand. This applies, in particular, to the development of new measuring devices using the Fourier-Galois transform. An example of such a device (a viscometer), using the GF(31) field, was considered in [41]. Other devices of this type, in which the electronic unit is significantly modernized [42], could be oriented towards other fields. However, the most important thing is that, in this case, the range of numbers in which the calculations are performed remains limited. Another non-trivial field of application for the developed approach in the future may be electronics implemented on a non-standard element base. The search for such an element base is currently being pursued very actively (for example, this concerns the development of neuromorphic materials [43–45], optical computers [46,47], etc.). Promising neuromorphic materials are very diverse. In particular, it has been proposed to use organic electronics for this purpose [48,49], including those that use organic transistor [50] and optoelectronic elements [51]. A distinctive feature of such elements is ion-electron conductivity [52], which raises the question of developing an algorithmic basis complementary to the presence of charge carriers of several types. It should also be considered that developments in this area are focused on the use of non-trivial memory cells [53]. Moreover, the results reflected in [54,55] suggest that it is possible for materials representing a "hybrid" of neural networks and classical logical electronics to be implemented on the basis of polymeric materials that form memory cells due to their physical and chemical properties. This is due to the specificity of these environments, in which the manifestations of the phase transition [56] are accompanied by the so-called "multidimensional" hysteresis [57]. As emphasized in [54], the non-trivial element base of computing technology inevitably requires significant modernization of the algorithmic basis for performing calculations.

Furthermore, the use of quasi-Mersenne numbers as the basis for the RNS used in the processor algorithm offers another advantage. Namely, as shown in [26], for the Galois fields $GF(17)$ and $GF(257)$, a relatively simple algorithm for digital logarithm can be proposed, which allows the multiplication operation to be reduced to the addition operation.

This algorithm implements the mapping of any non-zero element of the Galois field, represented as

$$x = \theta^n, \tag{22}$$

where θ is a primitive element of the field, on n: $x = \theta^n \rightarrow n$. Recall that a primitive element exists in any Galois field $GF(p)$, and its powers from 1 to $p-1$ exhaust all elements of this field.

The number of non-zero elements of the Galois field $GF(p)$ is $p-1$, and the digital logarithm operation provides a mapping of the fields $GF(17)$ and $GF(257)$ onto the Galois fields $GF(2^4)$ and $GF(2^8)$, respectively [26]. This means that the multiplication operation, when using the digital logarithm operation, is reduced to standard addition operations, which can be performed using standard processors.

Thus, the combination of a processor of the proposed type, built on adders in the fields $GF\left(2^k+1\right)$, creates additional prospects for increasing the performance of computing equipment by transitioning to parallel-sequential calculations.

4. Conclusions

Thus, a purely mathematical fact—the existence of a unique prime number 131,071, representable as $P = p_1p_2p_3p_4p_5 + 1$, where $p_1 = 2$ and the remaining p_i represent the first four quasi-Mersenne numbers, paves the way for the future implementation of a serial-parallel processor comparable in equivalent binary capacity to a classical 16-bit binary processor. This processor assumes independent operation with each component of an integer, representable in RNS, formed by the first four prime quasi-Mersenne numbers (3, 5, 17, and 257) and the number two. The architecture of such a processor, such as its classical analog, are adders, but for the proposed type of processor, these are adders modulo quasi-Mersenne numbers. The specificity of these numbers enables the implementation of a relatively simple circuit of a modulo adder. The simplicity of its design (compared to existing analogs) is ensured by the transition to a trigger circuit, where for certain values of the summed numbers, the circuit can be in two different stable states. Further development of the proposed approach can be based on the use of the digital logarithm operation. In this case, the multiplication operations in individual Galois fields corresponding to individual RNS components are reduced to addition operations, which are performed using conventional binary adders. The developed approach is applicable in cases where calculations are obviously carried out within a limited range of numerical values. In particular, for a processor built on the use of the first quasi-Mersenne numbers, the results obtained during calculations should not exceed 131,070. However, this is sufficient for many practical applications, particularly for the implementation of measuring devices built on the analysis of Fourier-Galois spectra.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/app15020741/s1. The program listing written in VHDL is attached in the Supplementary Materials.

Author Contributions: Conceptualization, K.K. and I.S.; methodology, I.S., A.B. and A.K.; formal analysis, A.K., K.K., A.B. and I.S.; writing—original draft preparation, A.K., K.K., A.B. and I.S.; writing—review and editing, A.K., K.K., A.B. and I.S.; visualization, K.K. and I.S.; supervision, I.S. All authors have read and agreed to the published version of the 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. AP23490107).

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/Supplementary Materials, further inquiries can be directed to the corresponding authors.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Brynjolfsson, E.; Hitt, L.M. Computing productivity: Firm-level evidence. *Rev. Econ. Stat.* **2003**, *85*, 793–808. [CrossRef]
2. Leiserson, C.E.; Thompson, N.C.; Emer, J.S.; Kuszmaul, B.C.; Lampson, B.W.; Sanchez, D.; Schardl, T.B. There's plenty of room at the Top: What will drive computer performance after Moore's law? *Science* **2020**, *368*, eaam9744. [CrossRef] [PubMed]
3. Qiu, T.; Chi, J.; Zhou, X.; Ning, Z.; Atiquzzaman, M.; Wu, D.O. Edge computing in industrial internet of things: Architecture, advances and challenges. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 2462–2488. [CrossRef]
4. Schweikl, S.; Obermaier, R. Lessons from three decades of IT productivity research: Towards a better understanding of IT-induced productivity effects. *Manag. Rev. Q.* **2020**, *70*, 461–507. [CrossRef]
5. Umuroglu, Y.; Conficconi, D.; Rasnayake, L.; Preusser, T.B.; Själander, M. Optimizing bit-serial matrix multiplication for reconfigurable computing. *ACM Trans. Reconfig. Technol. Syst. (TRETS)* **2019**, *12*, 1–24. [CrossRef]

6. Moss, D.J.; Boland, D.; Leong, P.H. A two-speed, radix-4, serial–parallel multiplier. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2018**, *27*, 769–777. [CrossRef]
7. Isupov, K. Using floating-point intervals for non-modular computations in residue number system. *IEEE Access* **2020**, *8*, 58603–58619. [CrossRef]
8. Peng, J.; Sun, S.; Narayana, V.K.; Sorger, V.J.; El-Ghazawi, T. Residue number system arithmetic based on integrated nanophotonics. *Opt. Lett.* **2018**, *43*, 2026–2029. [CrossRef]
9. Mohan, P.A.; Mohan, P.A. *Residue Number Systems*; Birkhäuser: Cham, Switzerland, 2016; pp. 16–24.
10. Efstathiou, C.; Kouretas, I.; Kitsos, P. On the modulo 2n+1 addition and subtraction for weighted operands. *Microprocess. Microsyst.* **2023**, *101*, 104897. [CrossRef]
11. Patel, B.K.; Kanungo, J. Efficient Tree Multiplier Design by using Modulo 2 n+ 1 Adder. In Proceedings of the 2021 Emerging Trends in Industry 4.0 (ETI 4.0), Raigarh, India, 19–21 May 2021; pp. 1–6. [CrossRef]
12. Irkhin, V.P.; Glazkov, E.B.; Lukyanov, M.A.; Dolgachev, A.A.; Kryukov, Y.G. Device for Addition and Subtraction of Numbers Modulo. 2020. Available online: https://patenton.ru/patent/SU1599857A1 (accessed on 12 August 2024).
13. Ahmadifar, H.; Torabi, Z. Adder-Only Reverse Converters for 5-Moduli Set {2 q, 2 q− 1, 2 q+ 1±1, 2 q+ 2− 1}. *IETE J. Res.* **2024**, *70*, 7346–7353. [CrossRef]
14. 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]
15. 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 Proceedings of the 2022 5th International Conference on Pattern Recognition and Artificial Intelligence (PRAI), Chengdu, China, 19–21 August 2022; pp. 637–642.
16. Pan, C.; Wang, C.Y.; Liang, S.J.; Wang, Y.; Cao, T.; Wang, P.; Miao, F. Reconfigurable logic and neuromorphic circuits based on electrically tunable two-dimensional homojunctions. *Nat. Electron.* **2020**, *3*, 383–390. [CrossRef]
17. Wei, S. Reconfigurable computing: A promising microchip architecture for artificial intelligence. *J. Semicond.* **2020**, *41*, 020301. [CrossRef]
18. Luo, S.; Song, M.; Li, X.; Zhang, Y.; Hong, J.; Yang, X.; You, L. Reconfigurable skyrmion logic gates. *Nano Lett.* **2018**, *18*, 1180–1184. [CrossRef]
19. 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. [CrossRef]
20. Bhattacharjee, K.; Das, S. A search for good pseudo-random number generators: Survey and empirical studies. *Comput. Sci. Rev.* **2022**, *45*, 100471. [CrossRef]
21. Matsumoto, M.; Nishimura, T. Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Trans. Model. Comput. Simulat.* **1998**, *8*, 3–30. [CrossRef]
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. *J. Math. Sci.* **2022**, *260*, 241–248. [CrossRef]
23. Ali, S.; Cenk, M. Faster residue multiplication modulo 521-bit Mersenne prime and an application to ECC. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2018**, *65*, 2477–2490. [CrossRef]
24. Nenashev, V.A.; Sergeev, A.M.; Kapranova, E.A. Research and analysis of autocorrelation functions of code sequences formed on the basis of monocyclic quasi-orthogonal matrices. *Inf. Control Syst.* **2018**, *4*, 9–14. [CrossRef]
25. Shin, S.H.; Yoo, W.S.; Choi, H. Development of modified RSA algorithm using fixed mersenne prime numbers for medical ultrasound imaging instrumentation. *Comput. Assist. Surg.* **2019**, *24* (Suppl. S2), 73–78. [CrossRef] [PubMed]
26. Bakirov, A.; Matrassulova, D.; Vitulyova, Y.; Shaltykova, D.; Suleimenov, I. The specifics of the Galois field GF (257) and its use for digital signal processing. *Sci. Rep.* **2024**, *14*, 15376. [CrossRef] [PubMed]
27. Yin, W.; Wen, Q.; Li, W.; Zhang, H.; Jin, Z. An anti-quantum transaction authentication approach in blockchain. *IEEE Access* **2018**, *6*, 5393–5401. [CrossRef]
28. Turner, C.S. A fast binary logarithm algorithm [DSP tips & tricks]. *IEEE Signal Process. Mag.* **2010**, *27*, 124–140. [CrossRef]
29. Sinha, S.K.; Kumari, S.; Kataria, A.; Thangarasu, N.; Sahoo, G.S. Blockchain empowerment: Investigating integration with software-defined networks and its impact on IoT privacy. *Multidiscip. Rev.* **2023**, *6*, e2023ss073. [CrossRef]
30. Chen, X.; Zhang, Z.; Ren, J.S.; Tian, L.; Qiao, Y.; Dong, C. A new journey from SDRTV to HDRTV. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Virtual, 11–17 October 2021; pp. 4500–4509.
31. Arnold, J.; Frater, M.; Pickering, M. *Digital Television: Technology and Standards*; John Wiley & Sons: Hoboken, NJ, USA, 2007.
32. Hiasat, A.A. High-speed and reduced-area modular adder structures for RNS. *IEEE Trans. Comput.* **2002**, *51*, 84–89. [CrossRef]
33. Tay, T.F.; Chang, C.H. A new unified modular adder/subtractor for arbitrary moduli. In Proceedings of the 2015 IEEE International Symposium on Circuits and Systems (ISCAS), Lisbon, Portugal, 24–27 May 2015; pp. 53–56. [CrossRef]
34. Maitland, D.S.; Chumbley, S.L.; Bradley, H.E. Modular Binary Half-Adder. U.S. Patent No. 4,054,788, 18 October 1977. Available online: https://patents.google.com/patent/US4054788A/en (accessed on 12 August 2024).

35. Lin, W.C. Modular Operation Circuit Adopting Iterative Calculations. U.S. Patent No. 11,662,978, 30 May 2023. Available online: https://patents.google.com/patent/US11662978B2/en (accessed on 12 August 2024).
36. Gu, J.; Wang, Z.; Kuen, J.; Ma, L.; Shahroudy, A.; Shuai, B.; Chen, T. Recent advances in convolutional neural networks. *Pattern Recognit.* **2018**, *77*, 354–377. [CrossRef]
37. Afridi, M.J.; Ross, A.; Shapiro, E.M. On automated source selection for transfer learning in convolutional neural networks. *Pattern Recognit.* **2018**, *73*, 65–75. [CrossRef]
38. Li, Z.; Liu, F.; Yang, W.; Peng, S.; Zhou, J. A survey of convolutional neural networks: Analysis, applications, and prospects. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 6999–7019. [CrossRef]
39. Sun, L.; Zhang, Y.; Chang, X.; Wang, Y.; Xu, J. Cloud-aware generative network: Removing cloud from optical remote sensing images. *IEEE Geosci. Remote Sens. Lett.* **2019**, *17*, 691–695. [CrossRef]
40. Li, X.; Wang, L.; Cheng, Q.; Wu, P.; Gan, W.; Fang, L. Cloud removal in remote sensing images using nonnegative matrix factorization and error correction. *ISPRS J. Photogramm. Remote Sens.* **2019**, *148*, 103–113. [CrossRef]
41. Kadyrzhan, K.; Kaldybekov, D.; Baipakbaeva, S.; Vitulyova, Y.; Matrassulova, D.; Suleimenov, I. Electronic Fourier–Galois Spectrum Analyzer for the Field GF (31). *Appl. Sci.* **2024**, *14*, 7770. [CrossRef]
42. Suleimenov, I.E.; Mun, G.A.; Kabdushev, S.B.; Alikulov, A.; Shaltykova, D.B.; Moldakhan, I. The design of viscometer with smartphone controlling. *Indones. J. Electr. Eng. Comput. Sci.* **2022**, *27*, 366–374. [CrossRef]
43. Torres, F.; Basaran, A.C.; Schuller, I.K. Thermal management in neuromorphic materials, devices, and networks. *Adv. Mater.* **2023**, *35*, 2205098. [CrossRef] [PubMed]
44. Sangwan, V.K.; Hersam, M.C. Neuromorphic nanoelectronic materials. *Nat. Nanotechnol.* **2020**, *15*, 517–528. [CrossRef]
45. Oh, S.; Hwang, H.; Yoo, I.K. Ferroelectric materials for neuromorphic computing. *APL Mater.* **2019**, *7*, 091109. [CrossRef]
46. Zhu, H.H.; Zou, J.; Zhang, H.; Shi, Y.Z.; Luo, S.B.; Wang, N.; Liu, A.Q. Space-efficient optical computing with an integrated chip diffractive neural network. *Nat. Commun.* **2022**, *13*, 1044. [CrossRef]
47. Kazanskiy, N.L.; Butt, M.A.; Khonina, S.N. Optical computing: Status and perspectives. *Nanomaterials* **2022**, *12*, 2171. [CrossRef]
48. Krauhausen, I.; Koutsouras, D.A.; Melianas, A.; Keene, S.T.; Lieberth, K.; Ledanseur, H.; Sheelamanthula, R.; Giovannitti, A.; Torricelli, F.; Mcculloch, I.; et al. Organic Neuromorphic Electronics for Sensorimotor Integration and Learning in Robotics. *Sci. Adv.* **2021**, *7*, eabl5068. [CrossRef]
49. Krauhausen, I.; Coen, C.; Spolaor, S.; Gkoupidenis, P.; Van De Burgt, Y. Brain-Inspired Organic Electronics: Merging Neuromorphic Computing and Bioelectronics Using Conductive Polymers. *Adv. Funct. Mater.* **2024**, *34*, 2307729. [CrossRef]
50. Giovannitti, A.; Sbircea, D.-T.; Inal, S.; Nielsen, C.B.; Bandiello, E.; Hanifi, D.A.; Sessolo, M.; Malliaras, G.G.; McCulloch, I.; Rivnay, J. Controlling the Mode of Operation of Organic Transistors through Side-Chain Engineering. *Proc. Natl. Acad. Sci. USA* **2016**, *113*, 12017–12022. [CrossRef] [PubMed]
51. Wu, X.; Wang, S.; Huang, W.; Dong, Y.; Wang, Z.; Huang, W. Wearable In-Sensor Reservoir Computing Using Optoelectronic Polymers with through-Space Charge-Transport Characteristics for Multi-Task Learning. *Nat. Commun.* **2023**, *14*, 468. [CrossRef] [PubMed]
52. Zhang, Y.; Van Doremaele, E.R.W.; Ye, G.; Stevens, T.; Song, J.; Chiechi, R.C.; Van De Burgt, Y. Adaptive Biosensing and Neuromorphic Classification Based on an Ambipolar Organic Mixed Ionic–Electronic Conductor. *Adv. Mater.* **2022**, *34*, 2200393. [CrossRef] [PubMed]
53. Zhang, B.; Chen, W.; Zeng, J.; Fan, F.; Gu, J.; Chen, X.; Yan, L.; Xie, G.; Liu, S.; Yan, Q.; et al. 90% Yield Production of Polymer Nano-Memristor for in-Memory Computing. *Nat. Commun.* **2021**, *12*, 1984. [CrossRef]
54. Suleimenov, I.; Gabrielyan, O.; Kopishev, E.; Kadyrzhan, A.; Bakirov, A.; Vitulyova, Y. Advanced Applications of Polymer Hydrogels in Electronics and Signal Processing. *Gels* **2024**, *10*, 715. [CrossRef]
55. 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 International Publishing: Cham, Switzerland, 2019; pp. 26–35. [CrossRef]
56. Shaikhutdinov, R.; Mun, G.; Kopishev, E.; Bakirov, A.; Kabdushev, S.; Baipakbaeva, S.; Suleimenov, I. Effect of the Formation of Hydrophilic and Hydrophobic–Hydrophilic Associates on the Behavior of Copolymers of N-Vinylpyrrolidone with Methyl Acrylate in Aqueous Solutions. *Polymers* **2024**, *16*, 584. [CrossRef]
57. Suleimenov, I.E.; Gabrielyan, O.A.; Bakirov, A.S. Initial study of general theory of complex systems: Physical basis and philosophical understanding. *Bull. Electr. Eng. Inform.* **2025**, *14*, 774–789. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.