



Article

# Binomial Number System

Oleksiy Borysenko <sup>1</sup>, Svitlana Matsenko <sup>2,\*</sup>  and Vjaceslavs Bobrovs <sup>3</sup> 

<sup>1</sup> Department of Electronics and Computer Technology, Sumy State University, 40007 Sumy, Ukraine; O.Borysenko@ekt.sumdu.edu.ua

<sup>2</sup> Communication Technologies Research Center, Riga Technical University, 1048 Riga, Latvia

<sup>3</sup> Institute of Telecommunications, Riga Technical University, 1048 Riga, Latvia; Vjaceslavs.Bobrovs@rtu.lv

\* Correspondence: Svitlana.Matsenko@rtu.lv

**Abstract:** This paper presents and first scientifically substantiates the generalized theory of binomial number systems (BNS) and the method of their formation for reliable digital signal processing (DSP), transmission, and data storage. The method is obtained based on the general theory of positional number systems (PNS) with conditions and number functions for converting BNS with a binary alphabet, also allowing to generate matrix BNS, linear-cyclic, and multivalued number systems. Generated by BNS, binomial numbers possess the error detection property. A characteristic property of binomial numbers is the ability, on their basis, to form various combinatorial configurations based on the binomial coefficients, e.g., compositions or constant-weight (CW) codes. The theory of positional binary BNS construction and generation of binary binomial numbers are proposed. The basic properties and possible areas of application of BNS researched, particularly for the formation and numbering of combinatorial objects, are indicated. The CW binomial code is designed based on binary binomial numbers with variable code lengths. BNS is efficiently used to develop error detection digital devices and has the property of compressing information.

**Keywords:** binomial number systems (BNS); generalized positional number systems (GPNS); binomial code; constant-weight (CW) binomial code



**Citation:** Borysenko, O.; Matsenko, S.; Bobrovs, V. Binomial Number System. *Appl. Sci.* **2021**, *11*, 11110. <https://doi.org/10.3390/app112311110>

Academic Editors: Aleksandr Cariou and Oleg Finko

Received: 17 October 2021

Accepted: 20 November 2021

Published: 23 November 2021

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2021 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

Positional number systems (PNS) solve two main tasks, namely determining the amount and arranging information about different objects. The PNS uses numeric numbering according to specific rules and provides arithmetic operations' performance on them. In addition, these systems number numbers according to particular rules and perform arithmetic operations on them [1]. The most simple and effective PNS used remain natural number systems, namely decimal, binary, and octal. There are more complex PNS, e.g., polyadic, allowing to solve particular tasks, such as transforming numbers from the system of residual classes into PNS, or using such an important variable as factorial number systems to generate permutations for combinatorial optimization problems.

Many researchers have investigated the non-integer PNS based on the golden ratio [2], Fibonacci code [3–5]. The development of new number systems research is available in [6,7]. Scientific works in the field of complex numbers are considered in [8–10]. A new method for representing images using a non-standard PNS is developed and discussed in [11]. These papers have shown that the research and development of PNS are of scientific importance and potential for applications. The effective use of PNS in design technologies and new applications in cryptography, machine learning and deep convolutional neural networks, the internet of things (IoT) devices, post-quantum algorithms and circuits, embedded processor design, and other related emerging topics based on the algorithms of the residue number system (RNS) are addressed in [12–17]. A comparison of various techniques and architectures for efficiently performing arithmetic operations in the logarithmic number system (LNS) is presented in [18].

PNS comprises the alphabet, the set of numbers representable in it, restrictions, and a numerical (numbering) function. Conditions of the PNS form the numbers used by number systems, determine the rules for performing arithmetic operations on them, and, if adding redundancy, make it possible to detect errors. However, the primary function for the number system is the numerical function, which uses the digits of numbers as arguments. Such a function solves the task of numbering, transforms it from the original number system into the numbers of the natural system, and, vice versa, organizes the transition from numbers of the natural number system to the numbers of the original design. When the digits numerate, the data compression function performs, and when denumerated, it restores. Data compression is possible only for numbers with rather complex restrictions, leading to redundant information and, therefore, the error detection property for error detection codes. Natural number systems have the most specific limits in the number of digits in the alphabet and representable numbers in its set. It, therefore, does not have redundant information in the numbers and thus cannot detect errors.

The most famous system among redundant number systems is the Fibonacci number system [19]. At present, several classes of BNS are being developed. These are binary, multivalued, linear-cyclic, matrix systems [20,21]. Each of these classes has its specific features, united by binomial coefficients as its weights. Fibonacci and BNS can detect errors in numbers for different types of errors. However, the increased complexity of BNS makes them less used for building universal devices, although its error detection property is not lower than that of Fibonacci. BNS can generate and number based on its digits various combinatorial configurations, which makes its use preferable to other number systems when solving several information tasks and, especially, in the circuit implementation of the corresponding digital devices [20,21]. The BNS determines the weight of the digits of binomial numbers and depends not only on their positions in the number, but also on the digits preceding it. Such connections complicate the algorithms for constructing binomial numbers, although they lead to new positive qualities. The most researched theoretical and practical applications are binary BNS, are considered in sufficient detail in [20]. This paper aims to present the most significant theoretical and empirical results in the theory and practice of binary BNS.

The paper is structured as follows. Section 2 describes the theory of BNS. The quantitative characteristics of BNS are present in Section 3. The generation of CW binomial code is shown in Section 4. Finally, we conclude this paper in Section 5.

## 2. Theory of BNS

**Definition 1.** A binary  $k$ —BNS is a system that includes a numeric function:

$$F = x_{r-1}C_{n-1}^{k-q_{r-1}} + \dots + x_i C_{n-r+i}^{k-q_i} + \dots + x_1 C_{n-r+1}^{k-q_1} + x_0 C_{n-r}^{k-q_0}, \tag{1}$$

set of binomial numbers  $F = x_{r-1} \dots x_i + \dots + x_1 x_0$ , binary alphabet  $x_i = 0, 1$ , and systems of conditions, generating binomial code [21]:

$$\begin{cases} q \leq r \leq n - 1 \\ x_0 = 1, q = k \end{cases} \tag{2}$$

and

$$\begin{cases} n - k = r - q \\ x_0 = 0, 0 \leq q \leq k - 1, \end{cases} \tag{4}$$

where  $r$  denotes the number of binary digits in binomial numbers (code length);  $r \in 1, 2, \dots$ ,

$i$  is a position index  $i = 0, 1, \dots, r - 1$ ;

$q$  is a number of 1's in a binomial code;

$n, k$  is an integer parameter of the BNS— $1, 2, \dots$ ;

$q_i$  is a sum of 1's bits of the  $x_i$  from  $(r - 1)$ -th to the  $(i + 1)$ -th bits,

$$q_i = \sum_{j=i+1}^r x_j \tag{6}$$

$i = 0, 1, \dots, r - 1; x_r = 0.$

**Definition 2.** Any finite sequence of binary digits  $x_{r-1}, \dots, x_1, x_0$ , with conditions (2) and (3) or (4) and (5) is the binary binomial number.

Thus, the codewords shown in Table 1 below are binary binomial numbers with the above conditions. Based on the binomial numbers, binomial codes have two essential features that distinguish them from ordinary numbers in natural number systems. The first feature of binomial codes is that their code lengths  $r$  are different, and in total, they form a prefix code, that is, a code in which no combination can be the beginning of another. It follows from the systems of equalities (2) and (3), (4), and (5), where  $r$  is a variable depending on the number of  $q$  ones in codes and their parameters  $n$  and  $k$ . Such codes are called non-uniform, in contrast to binomial codes in natural number systems, such as the binary system, which have the same code length. Therefore, such codes are called uniform. The second feature of binomial codes is the presence of the dependence of the values of the weights of their digits not only on their position in the number, as is the case in natural number systems, but also on the importance of the preceding most significant bits (MSB), which equation described (6). Due to this property, the lengths of the binomial codes are different. These properties significantly complicate the binomial codes, give them the property of error detection, and enable them to form various combinatorial configurations, e.g., combinations, combinations with repetitions, compositions, etc.

**Table 1.** Binomial non-uniform codewords with parameters  $n = 6, k = 4$  and quantitative equivalents.

No	Binomial Codeword	Quantitative Equivalent
0	00	$0C_5^4 + 0C_4^4$
1	010	$0C_5^4 + 1C_4^4 + 0C_3^3$
2	0110	$0C_5^4 + 1C_4^4 + 1C_3^3 + 0C_2^2$
3	01110	$0C_5^4 + 1C_4^4 + 1C_3^3 + 1C_2^2 + 0C_1^1$
4	01111	$0C_5^4 + 1C_4^4 + 1C_3^3 + 1C_2^2 + 1C_1^1$
5	100	$1C_5^4 + 0C_4^4 + 0C_3^3$
6	1010	$1C_5^4 + 0C_4^4 + 1C_3^3 + 0C_2^2$
7	10110	$1C_5^4 + 0C_4^4 + 1C_3^3 + 1C_2^2 + 0C_1^1$
8	10111	$1C_5^4 + 0C_4^4 + 1C_3^3 + 1C_2^2 + 1C_1^1$
9	1100	$1C_5^4 + 1C_4^3 + 0C_3^2 + 0C_2^2$
10	11010	$1C_5^4 + 0C_4^3 + 0C_3^2 + 1C_2^2 + 0C_1^1$
11	11011	$1C_5^4 + 1C_4^3 + 0C_3^2 + 1C_2^2 + 1C_1^1$
12	11100	$1C_5^4 + 1C_4^3 + 1C_3^2 + 0C_2^1 + 0C_1^1$
13	11101	$1C_5^4 + 1C_4^3 + 1C_3^2 + 0 + 1C_1^1$
14	1111	$1C_5^4 + 1C_4^3 + 1C_3^2 + 1C_2^1$

As a weighting coefficient of the  $i$ -th digit of a numerical function, the binomial coefficient is determined  $C_{n-r+i}^{k-q_i}$ . The binomial coefficient depends on the position index  $i = 0, 1, \dots, r - 1$  and the amount,  $q_i$  preceding this index of bits  $x_i$ . There can be as many binary BNS for code length  $n$  as there can be for  $n$  values of the parameter  $k$ . Therefore, for the BNS, it is necessary to indicate the parameters  $n$  and  $k$ , or at least the parameter  $k$ , since it does not exceed the value of  $n$ .

For example, Table 1 shows the binomial codewords with parameters  $n = 6$  and  $k = 4$  and quantitative equivalents.

The binomial codewords with different code lengths  $r$  are shown in Table 1, which for numbers corresponding to the system of equalities (2) and (3) varies from  $q = k = 4$  (3) to  $n - 1 = 6 - 1 = 5$  (2). Moreover, all these codewords have 1 in the least significant bit

(LSB). The codewords in the considered example, containing  $k = 4$  ones and 1 in the LSB, are determined by the binomial coefficient  $C_5^4 = 5$ . These are the codewords—01111, 10111, 11011, 11101, 1111. The codeword 1111 =  $1C_5^4 + 1C_4^3 + 1C_3^2 + 1C_2^1 = 14$  in this example is the largest.

For the remaining codewords, based on the system of Equations (4) and (5), with the number of units  $q = 0$ , code length  $r = n - k + q = 6 - 4 + 0 = 2$ , for  $q = k - 1 = 4 - 1 = 3$ ,  $r = n - k + q = n - k + k - 1 = n - 1 = 6 - 1 = 5$ . Moreover, all these codewords must contain 0 in the LSB. Consequently, the code length  $r$  of the binomial code for equalities (4) and (5) varies from  $n - k$  to  $n - 1$  value. For the number of ones  $q = 0$ , as follows from equality (4), codeword obtained encoding 0 and consisting only of  $n - k = 6 - 4 = 2$  zeros. The presence of such a zero-code combination for positional numbers of any class is required. The codewords with 0 in the LSB, for the example under consideration in Table 1, are determined by the binomial coefficient  $C_5^2 = 10$ . These are the codewords—00, 010, 0110, 01110, 100, 1010, 10110, 1100, 11010, 11100. The code combination  $00 = 0C_5^4 + 0C_4^4 = 0$  in this example is the smallest. Summing the numbers of codewords, the LSB containing 0 and 1, we obtain the range of binomial codes for the parameters  $n = 6$  and  $k = 4 - C_5^2 = C_5^3 + C_5^4 = C_6^4 = 15$ .

In Table 1, the  $r$  of binomial codes in classes containing 0 and 1 in the LSB does not exceed the value  $n - 1 = 5$ . If the codewords in the LSB contain 1, then it must contain  $q = k = 4$  ones, if the LSB contains 0, then the number of  $q$  units changes from 0 to  $k - 1 = 4 - 1 = 3$ , wherein the number of zeros constant and equals  $n - k = 6 - 4 = 2$ . Violation of these conditions is a sign of errors in binomial codes. For example, if binary code combinations represented by binomial codes have the form 11111, 000, 0001, they contain errors since they belong to the code combinations prohibited for binomial codes. In the first case, the sequence has more than five ones, and in the second and third, there are more than two zeros in the code combination. The above conditions for binomial codes forbid both cases.

However, the indicated errors in binomial codes detect if markers of the MSB and LSB of the binomial code combinations. These markers indicate using zeros, which contain the LSB that transforms non-uniform binomial code into uniform code, with length  $r = n - 1$ , shown in Table 2.

**Table 2.** Non-uniform and uniform binomial codewords with parameters  $n = 6, k = 4$ .

No	Binomial Non-Uniform Codewords	Binomial Uniform Codewords	No	Binomial Non-Uniform Codewords	Binomial Uniform Codewords
0	00	00000	8	10111	10111
1	010	01000	9	1100	11000
2	0110	01100	10	11010	11010
3	01110	01110	11	11011	11011
4	01111	01111	12	11100	11100
5	100	10000	13	11101	11101
6	1010	10100	14	1111	11110
7	10110	10110			

Thus, non-uniform binomial codewords are binomial codewords. Binomial uniform codewords are converted binomial code combinations with added redundancy. In the binary code combinations 01100 00000 11101 11010 by their number equal to 5, it is possible to identify binomial uniform codewords and, accordingly, errors in them. For example, if, after counting five positions, the code combination 01101 appears instead of the binomial codeword 01100, then it is not a binomial code. This error is detected because in non-uniform binomial codewords having 1 in the LSB, with parameters  $n = 6, k = 4$ , there can be no zeros until the first 1 on the right, but in this case, there are two of them.

### 3. Quantitative Characteristics of BNS

#### 3.1. BNS Property

Each PNS, including BNS, must meet several requirements for such number systems. First of all, this is the requirement to represent the digits of the number system in bits. After this, the implementation of transitions from these codes to the corresponding codes of the natural number system. The transition for PNS with redundant information leads to its elimination and, accordingly, information compression. Moreover, the requirement for PNS is the ability to implement the reverse transition from the obtained natural number to the digit of the original redundant number system. At the same time, in the transition from one natural number system to another, there is no compression of information since natural number systems do not contain redundant information.

As shown above, binomial numbers effectively represent using the system conditions (2) and (3), (4) and (5), and from the numerical function for the BNS (1), it follows that it is calculating entirely and, therefore, is capable of transforming binomial numbers to their natural digits. The inverse problem of converting the digits of the natural number system into binomial numbers can also be solved efficiently using this numerical function [21]. Thus, the requirement for efficient representation of binomial numbers using BNS is performed. Algorithms for constructing binomial numbers obtain if the conditions meet the specified restrictions. Further theoretical constructions aim to prove the theoretical foundations for creating all binomial numbers from a given range.

**Theorem 1.** *The sum of zeros contained in the binomial code combinations formed by conditions (2) and (3):*

$$l = 0, 1, \dots, n - k - 1. \quad (7)$$

**Proof of Theorem 1.** The sum of zeros  $l$  in binomial code combinations is equal to  $l = r - q$ . Condition (3) shows that  $q = k$ . Therefore,  $l = r - k$ . Since it follows from condition (2) of the system of conditions (2) and (3) that the minimum value is  $r = r_{min} = k$ , and the maximum value is  $r = r_{max} = n - 1$ . Then, substituting the minimum value  $r_{min}$  and maximum value into equation  $l = r - k$ , as a result  $l = l_{min} = 0$  and  $l = l_{max} = n - k - 1$ . Other possible values are between  $l_{min}$  and  $l_{max}$ , respectively  $l = 0, 1, \dots, n - k - 1$ .  $\square$

**Theorem 2.** *The sum of zeros in binomial codes, which comply with the conditions (4) and (5):*

$$l = n - k. \quad (8)$$

**Proof of Theorem 2.** The sum of zeros  $l$  in binomial codes, including specified by the conditions (4) and (5), equals  $l = r - q$ . It follows from Equation (4) that the number of ones in the corresponding binomial code combinations is  $q = r - (n - k)$ . This is substituting the last equality in the previous:  $l = n - k$ .  $\square$

**Theorem 3.** *The number of units  $q$ , which correspond to conditions (2) and (3), equals  $k$ , and conditions (4) and (5)— $0, 1, \dots, k - 1$ .*

**Proof of Theorem 3.** The first condition of the theorem follows from Equation (3) of the system of conditions (2) and (3). From Equation (5) of the system of conditions (4) and (5) follows that  $q = q_{min} = 0$  and  $q = q_{max} = k - 1$ . Other possible values of  $q$  are in the range between  $q_{min}$  and  $q_{max}$ . Therefore, the number of ones in binomial codes with zero in the LSB is  $q = 0, 1, \dots, k - 1$ .  $\square$

**Theorem 4.** *The maximum code length of binomial codes specified by conditions (4) and (5) is  $r_{max} = n - 1$ , the minimum is  $r_{min} = n - k$ .*

**Proof of Theorem 4.** From Equation (4) of the system of conditions (4) and (5), it follows that  $r = n - k + q$ , and from Equation (5) of the same system, it follows that the maximum value  $q = q_{max} = k - 1$ . Then,  $r = r_{max} = n - k + q_{max} = n - k + k - 1 = n - 1$ . It also follows from (5) that the minimum value  $q = q_{min} = 0$  and, accordingly,  $r = r_{min} = n - k + q_{min} = n - k$ .  $\square$

**Theorem 5.** *The number of different code lengths of the binomial codes with conditions (2) and (3),  $d' = n - k$ , and the number of different code lengths given by the conditions (4) and (5),  $d'' = k$ .*

**Proof of Theorem 5.** To find the number of code lengths of binomial codes, it is necessary to subtract the minimum code length  $r_{max}$  from its maximum code length  $r_{min}$ , adding 1 to this difference. From Equation (2), it follows that  $r_{max} = n - 1$  and  $r_{min} = k$ , the number of different code lengths of binomial code combinations determined by conditions (2) and (3) is found from the expression:  $d' = r_{max} - r_{min} + 1 = (n - 1) - k + 1 = n - k$ . For binomial codes corresponding to conditions (4) and (5), it follows from Theorem 4 that  $r_{max} = n - 1$  and  $r_{min} = n - k$ . Then, the number of different code lengths of binomial codes by conditions (4) and (5) determines from the expression  $d'' = r_{max} - r_{min} + 1 = (n - 1) - (n - k) + 1 = k$ .  $\square$

**Theorem 6.** *The parameter  $k$  (the maximum number of units) in binomial codes is equal to the minimum value  $k_{min} = 1$ , and the maximum value is equal to  $k_{max} = n - 1$ .*

**Proof of Theorem 6.** From Equation (5):  $q \leq k - 1$  and  $q \geq 0$ . These inequalities follow that for  $q = 0 = k - 1$ , and parameter  $k$  reaches its minimum value  $k_{min}$ , equal to 1. Since the code length of binomial codes is  $r = q + 1$ , for the system of conditions (2) and (3),  $q = k$ , then  $r = k + 1$ . Substituting  $k + 1$  into Equation (2):

$$k \leq k + 1 \leq n - 1.$$

In this equation,  $k$  reaches its maximum value  $k_{max}$ , when  $l$  comes the minimum value  $l_{min}$ . As follows from Theorem 1,  $l = l_{min} = 0$ . Substituting this value into the inequality obtained above, we obtain that, for  $l = 0$ , it is transformed into the form  $k \leq n - 1$ . Thus, the parameter  $k$  reaches the maximum value if  $k = n - 1$ , for  $k = k_{max} = n - 1$ .  $\square$

**Corollary 1.** *There are no binomial codes with parameter  $k = 0$ .*

**Corollary 2.** *In the case when the parameter  $k = k_{min} = k_{max} = 1$  in binomial codes, the value of the parameter  $n$  becomes minimal— $n = n_{min} = 2$ . Thus,  $n = 2, 3, \dots$*

Obviously, for  $n = n_{min} = 2$  only two non-uniform binomial code combinations with  $k = 1$ —0, 1 are possible; three for  $n = 3$ —00, 01, 1, four for  $n = 4$ —000, 001, 01, 1, etc. The corresponding uniform binomial codewords: 0, 1, 00, 01, 10; 000, 001, 010, 100. In the case when  $k = n - 1$ , for  $n = 2$  we have 0.1; for  $n = 3$ —0, 10, 11; for  $n = 4$ —0, 10, 110, 111. The corresponding uniform binomial codewords are 0, 1, 00, 10, 11; 000, 100, 110, 111.

### 3.2. BNS Classes

The BNS comprises two classes. The first contains numbers with a constant amount of  $k$  units and variables of  $i = 0, 1, \dots, n - k - 1$  number of zeros. The second class of the BNS consists of the constant number  $n - k$  number of zeros and a variable amount of units  $q = 0, 1, \dots, k - 1$ .

**Theorem 7.** *Conditions (2)–(5) divide binomial codes into two disjoint classes, the first of which contains  $k$  ones and  $0 \leq l \leq n - k - 1$  zeros, and the second includes  $l = n - k$  zeros and  $0 \leq q \leq k - 1$  units.*

**Proof of Theorem 7.** From Equation (3), it follows that conditions (2) and (3) generate binomial codes containing  $k$  ones, and from Theorem 2, those conditions (4) and (5) generate

numbers that constantly have  $n - k$  zeros. It also follows from Theorem 1 that the number of zeros in binomial codes specified by conditions (2) and (3),  $l = 0, 1, \dots, n - k - 1$ , and the number of ones in numbers specified by conditions (4) and (5),  $q = 0, 1, \dots, k - 1$ .

Moreover, binomial codes corresponding to conditions (2) and (3) do not correspond to requirements (4) and (5) and vice versa. By definition, there are no other binomial codes, except for the numbers generated by conditions (2)–(5).

Therefore, each binomial code can belong only to one of two classes of binomial numbers, corresponding to conditions (2) and (3) and consisting of  $k$  ones and  $0 \leq l \leq n - k - 1$  zeros or resembling conditions (4) and (5) and consisting of  $n - k$  zeros and  $0 \leq q \leq k - 1$  units. □

**Corollary 3.** *Binomial codes of the first-class contain 1 in the LSB, and the second class include 0 in the LSB since when the  $k$ -th unit or  $(n - k)$ -th zero appears, is the critical sign of the formation of the binomial codes.*

**Lemma 1.** *The sum of binomial codes of the first-class contains 1 in the LSB, with a fixed number of zeros  $l$ ,  $0 \leq l \leq n - k - 1$ :*

$$N_l = C_{k+l-1}^l. \tag{9}$$

**Proof of Lemma 1.** Since in the LSB of each binomial code combinations of the first class, by Corollary 1 of Theorem 7, there is 1, then the total number of binomial code of code length  $r$  for a given value  $l$   $N_l = C_{r-1}^l$ . Since the code lengths of binomial code from this class are  $r = k + l$ , then  $r - 1 = k + l - 1$ , and accordingly  $N_l = C_{k+l-1}^l$ . □

**Corollary 4.** *For  $l = l_{max} = n - k - 1$ :*

$$N_l = N_{l_{max}} = C_{k+l_{max}-1}^{l_{max}} = C_{n-2}^{n-k-1} = C_{n-2}^{k-1}. \tag{10}$$

**Corollary 5.** *For  $l = l_{min} = 0$ :*

$$N_l = N_{l_{min}} = C_{k-1}^0 = 1. \tag{11}$$

For parameters  $n = 6, k = 4$ , the number of binomial codewords, which contain 1 in the LSB and do not contain zeros, by Corollary 2, equal to 1, and with one 0— $N_{l=1} = C_{r-1}^1 = C_{5-1}^1 = 4$ . These are the codewords: 1111 01111, 10111, 11011, 11101. No other codewords contain more than one zero and have 1 in the LSB in this class of binomial codes, since with two zeros, the binomial codes, by Theorem 7, transform to the type of numbers containing 0 in the LSB.

**Theorem 8.** *Sum of all binomial code combinations of the first-class contain 1 in the LSB:*

$$N_k = C_{n-1}^k. \tag{12}$$

**Proof of Theorem 8.** Summing the  $N_l$  values obtained in Lemma 1 over all possible  $l = 0, 1, \dots, n - k - 1$ , provides the sum of all binomial code combinations containing  $k$  ones:

$$N_k = \sum_{l=0}^{n-k-1} N_l = \sum_{l=0}^{n-k-1} C_{k+l-1}^l = C_{n-2}^{n-k+1} + \dots + C_k^1 + C_{k-1}^0 = \sum_{j=0}^{n-k-1} C_{n-2-j}^{n-k-1-j} = C_{n-1}^{n-k-1} = C_{n-1}^k.$$

Theorem 8 that the sum of binomial code combinations for  $n = 6, k = 4$  containing 1 in the LSB is equal to  $C_{n-1}^k = C_{6-1}^4 = 5$ . The sum of the numbers given in the previous example, the LSB of which includes 1, also using Table 1, their total number is 5, which confirms this theorem. □

**Lemma 2.** *The sum of binomial code combinations of the second class containing 0 in the LSB and having a fixed number of ones  $q$ ,  $0 \leq q \leq k - 1$ :*



$$N_q = C_{n-k+q-1}^q \tag{13}$$

**Proof of Lemma 2.** Since, by Corollary 1 of Theorem 7, the LSB of the binomial codes contain 0, the total number of binomial codes, for a given value of  $q$ , is determined by the number of combinations of  $q$  units of  $r - 1$  element  $N_q = C_{r-1}^q$ . For  $r = n - k + q$ ,  $N_q = C_{n-k+q-1}^q$ . □

**Corollary 6.** For the number of units in binomial code combinations,  $q = q_{max} = k - 1$ :

$$N_q = N_{q_{max}} = C_{n-k+q_{max}-1}^{q_{max}} = C_{n-2}^{k-1} \tag{14}$$

Suppose it is necessary to determine the sum of binomial code combinations with two units among the set of numbers with parameters  $n = 6, k = 4$ . To define, apply Equation (14) from Lemma 2,  $N_q = C_{n-k+q-1}^q = C_{6-4+2-1}^2 = C_3^2 = 3$ . Table 1 also shows that there are three such binomial code combinations, 0110, 1010, and 1100.

**Theorem 9.** Sum of binomial code combinations of the second class that contain 0 in the LSB:

$$N_{n-k} = C_{n-1}^{k-1} \tag{15}$$

**Proof of Theorem 9.** Summing the values obtained in Lemma 2  $N_q$  for all possible  $q = 0, 1, \dots, k - 1$ , the sum of all binomial code combinations containing  $n - k$  zeros:

$$N_{n-k} = \sum_{q=0}^{k-1} N_q = \sum_{q=0}^{k-1} C_{n-k+q-1}^q = C_{n-2}^{k-1} + \dots + C_{n-k}^1 + C_{n-k-1}^0 = \sum_{j=0}^{k-1} C_{n-2-j}^{k-1-j} = C_{n-1}^{k-1}$$

The sum of such binomial codewords with parameters  $n = 6, k = 4$  are  $C_{n-1}^{k-1} = C_{6-1}^{4-1} = 10$ . □

**Theorem 10.** Sum of binomial code combinations, with conditions (2)–(5):

$$N = C_n^k \tag{16}$$

**Proof of Theorem 10.** Since the value  $N = N_k + N_{n-k}$ , then instead of  $N_k$  and  $N_{n-k}$ , values from Equations (13) and (17) are used:

$$N = N_k + N_{n-1} = C_{n-1}^k + C_{n-1}^{k-1} = C_n^k$$

□

**Corollary 7.** Since the sum of all binomial code combinations specified by conditions (2) and (3), (4) and (5) is equal to  $C_n^k$ , then the range  $P$  is equal to the same number:

$$P = C_n^k \tag{17}$$

## 4. Generation of Constant-Weight (CW) Binomial Code

### 4.1. Binomial Counting Algorithm

BNS, which generate binomial numbers and codes, can potentially be helpful for algorithms that solve practical tasks in information coding and the construction of reliable and high-speed digital technology. In conceptual terms, some of the possible applications of binomial numbers are implemented either in programs or schematic form for specialized digital devices and systems, data compression, and error detection codes. However, the practical application of binomial numbers does not end with this list. Almost everything practically implemented based on binary numbers can be implemented on binomial numbers, but in some exceptional cases with greater efficiency. These numbers' efficiency use on binomials is the same as the structure of combinatorial objects built on combinations. This makes it easy to transform from these objects to binomials and vice versa. As



a result, it is possible to propose a unified method for constructing combinatorial objects based on binomial numbers. In this case, there is no need for each type of combinatorial object to develop its process of creating them.

The binomial counting algorithm consists of two steps, the transition from several combinatorial objects to the binomial number. Then, a change is made, respectively, either to a combinatorial object or to a number. In the first case, the task of constructing combinatorial objects based on code combinations is solved, and in the second, their numbering. A universal approach is used for solving tasks of creating and numbering combinatorial objects, which makes it possible to simplify them and improve the quality of solutions.

We developed the binomial counting algorithm based on binomial numbers to solve many practical coding tasks independently and as part of binary digital devices and systems. The binomial counting algorithm consists of the following steps:

- (1) An initial code combination consisting of  $n - k$  zeros is formed.
- (2) The LSB of the code combination is set to 1, which is assigned a zero to the right.
- (3) Step 2 is repeated until the number of ones in the code combination becomes equal to  $k - 1$ . After that, one is set into the LSB, but 0 is not assigned.
- (4) The LSB containing zero is set 1, and zeros are assigned to the right so that their number in the number becomes  $n - k$ .
- (5) The number of units in the code combination is determined. If it is equal to  $k$  and the ones not contained in the  $k$  MSB, the algorithm returns to step 2.
- (6) If  $k$  ones are contained in the code combination of  $k$  MSB, the algorithm stops.
- (7) If the number of ones in the code combination is not equal to  $k$ , then to the right of the LSB containing 1, zeros are set until their total number becomes  $n - k$ .
- (8) Return to step 2.

This algorithm makes it possible to obtain all binomial codewords similar to the codewords shown in Table 1, but with a considerable code length, containing tens or more digits, which is a determining factor for many practical applications.

The considered algorithm can be used for a software implementation or microprocessor system (MPS). However, due to the uneven length of binomial codes, it is practically difficult to implement it in the form of a digital circuit with rigid logic, for example, on a field-programmable gate array (FPGA). Therefore, for particular tasks, there are more complex binomial calculating algorithms than the one presented above, using uniform binomial numbers, with the help of which several specific binomial digital devices and systems are constructed [20,21]. Uniform binomial codes, as indicated above, can also be obtained by replacing non-uniform binomial codes with uniform ones by equalizing the  $r$  of the first zeros, as shown in Table 2. As a result, they acquire the property of error detection, which can be practically used to detect errors due to the natural redundancy introduced into the structure of these devices at the time of their construction. This allows for an additional effect of error detection without introducing artificial redundancy while adding to these codes for higher noise immunity and artificial redundancy.

#### 4.2. CW Binomial Code

The main property of binomial numbers, both uniform and non-uniform, is forming various combinatorial configurations. In this paper, we consider CW binomial codes contained in each of their code combinations the same number of zeros and ones. Such codes widely used in practice for a long time, such as error-control codes. However, binomial numbers give them new properties. For example, they can effectively compress information [21].

The algorithm for generating CW binomial codes based on binomial numbers differs in its practical implementation in its simplicity. This is implemented by assigning ones to the binomial code if it contains  $(n - k)$  zeros, or if it contains  $k$  ones, zeros until its length equals  $n$  bits. The algorithm for obtaining CW codes consists of two steps—acquiring binomial numbers from natural number systems and constructing CW code combinations on their basis. The first step, which requires the conversion of binary numbers into binomial numbers for its implementation, is described, for example, in [21]. The second step, which transforms binomial numbers into CW binomial code combinations, is considered above. Based on binomial numbers, using the formation of CW binomial code combinations according to the above algorithm, it is also easy to organize their enumeration. It can convert binary codes into CW binomial codes and build the corresponding encoders [21].

Using the considered algorithm, Table 3 shows an example of transitions from binomial code-words to CW binomial codewords, for  $n = 6, k = 4$ .

**Table 3.** CW binomial codewords with parameters  $n = 6, k = 4$ .

No	Binomial Codewords	CW Binomial Codewords	No	Binomial Codewords	CW Binomial Codewords
0	00	001111	8	10111	101110
1	010	010111	9	1100	110011
2	0110	011011	10	11010	110101
3	01110	011101	11	11011	110110
4	01111	011110	12	11100	111001
5	100	100111	13	11101	111010
6	1010	101011	14	1111	111100
7	10110	101101			

The advantage of this algorithm, in addition to simplicity, is that on its basis, especially when using specialized devices, it is possible to quickly obtain combinations of CW binomial codes of considerable code length. CW binomial codes are indivisible error-detecting codes since they do not have an information and control part. As is the case with ordinary block codes, e.g., Hamming or low-density parity-check (LDPC) forward error correction (FEC) codes. These codes are asymmetric and adapt to the nature of the interference, which is one of their advantages over cycling block codes. With 100% asymmetry of the communication channel, these codes can detect all errors. The second advantage of these codes is the ease of detecting errors in their code combinations. It is enough to count the number of units in the code combination to find an error. If the number of ones is equal to  $k$ , then it is considered that there is no error in the code combinations, and if it is not identical, then its presence is assumed. This property is beneficial for hardware implementation based on CW binomial codes of encoding devices. The disadvantage of these codes is their inseparability, making it necessary to develop special algorithms for converting binary codes into CW binomial codes and vice versa with a considerable length of code combinations [21].

CW binomial codes, especially with a considerable code length, can also compress information, including digital images. It is enough to count the number in the compressible binary array of data to convert it into the CW binomial code combination. After that, we apply the standard procedure for moving from equilibrium codewords which consists of deleting the ones to the right to the first zero or zeros to the first unit (please see Table 3). The binomial number is obtained and converted to the corresponding binary numbers using a numeric function. In this case, that information compresses, and the degree of information compression depends on the number of ones and the code length of the original binary code combination.

Without the digit  $k$ , the number is not easy to recover, protecting information from unauthorized access. In this case, the parameter  $k$  is the key. If necessary, two keys can be entered,  $k$  and  $n$ . However, in this case, transmission reliability decreases since the natural redundancy removes from the messages. However, applying FEC codes with artificial redundancy, the reliability of the systems is increased. All considered coding operations solve without CW binomial codes based on binomial codes only. Still, the coding efficiency will be somewhat lower, although the equipment and algorithms will be more straightforward and reliable.

## 5. Conclusions

The BNS with the binary alphabet uses binomial coefficients as bit weights. The binomial numbers generated by the BNS divide into the first and second classes, characterized by 0 or 1 in the LSB of the binomial code. Each class is divided into groups with binomial codes of the same code length within the groups and different code lengths between the groups, differing by one bit. Compared with natural PNS, the advantage of binomial codes is the property of error detection and practical uses in information systems for data transmission and processing. Based on the BNS, digital devices, such as binomial decoders and counters, are effectively developed, which have the property of detecting errors. The binomial code has the data compressing property and allows the generation of CW binomial codes. Binary binomial codes have a novelty compared to known PNS and known binomial codes of different classes. The scientific novelty of this paper lies in the generalization of scientific results in the field of BNS and the method of obtaining them.

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

**Funding:** This work has been supported by the European Regional Development Fund within the Activity 1.1.1.2 “Post-doctoral Research Aid” of the Specific Aid Objective 1.1.1 “To increase the research and innovative capacity of scientific institutions of Latvia and the ability to attract external financing, investing in human resources and infrastructure” of the Operational Programme “Growth and Employment” (No. 1.1.1.2/VIAA/3/19/421).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

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

## References

1. Parhami, B. *Computer Arithmetic: Algorithms and Hardware Designs*; Oxford University Press: Oxford, UK, 2010; p. 641.
2. Bergman, G. A number system with an irrational base. *Math. Mag.* **1957**, *31*, 98–110. [[CrossRef](#)]
3. Sergeev, I. On the Complexity of Fibonacci Coding. *Probl. Inf. Transm.* **2018**, *54*, 343–350. [[CrossRef](#)]
4. Stakhov, A. Numeral Systems with Irrational Bases for Mission-Critical Applications. *World Sci.* **2017**, *61*, 284.
5. Stakhov, A.; Aranson, S.; Olsen, S. The “Golden” Non-Euclidean Geometry. *World Sci.* **2016**, 308.
6. Dimitrov, S.; Jullien, G.A. Loading the bases: A new number representation with applications. *IEEE Circuits Syst. Mag.* **2003**, *3*, 6–23. [[CrossRef](#)]
7. Elmasry, A.; Jensen, C.; Katajainen, J. Two skew-binary numeral systems and one application. *Theory Comput. Syst.* **2012**, *50*, 185–211. [[CrossRef](#)]
8. Sarkar, S.; Gomony, M.D. Quater-imaginary base for complex number arithmetic circuits. In Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, Germany, 19–23 May 2018; pp. 1481–1483.
9. Gilbert, J. Complex numbers with three radix expansions. *Can. J. Math.* **1982**, *34*, 1335–1348. [[CrossRef](#)]
10. Sun, W.; Qi, D.X. Complex number system-based algorithm for engineering graph digital watermarking. *J. Image Graph.* **2003**, *8*, 626–630.
11. Yang, W.; Tian, Y.; Zhou, F.; Liao, Q.; Chenglin, Z.; Zheng, C. Consistent coding scheme for single-image super-resolution via independent dictionaries. *IEEE Trans. Multimed.* **2016**, *18*, 313–325. [[CrossRef](#)]
12. Lan, T.; Cai, Z.C. A Novel Image Representation Method under a Non-Standard Positional Numeral System. *IEEE Trans. Multimed.* **2021**, *23*, 1301–1315. [[CrossRef](#)]
13. Sousa, L. Nonconventional Computer Arithmetic Circuits, Systems and Applications. *IEEE Circuits Syst. Mag.* **2021**, *21*, 6–40. [[CrossRef](#)]
14. Kenneth Jenkins, W.; Soderstrand, M.; Radhakrishnan, C. Historical Patterns of Emerging Residue Number System Technologies During the Evolution of Computer Engineering and Digital Signal Processing. In Proceedings of the 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, Italy, 27–30 May 2018; pp. 1–5.
15. Olsen, E. RNS Hardware Matrix Multiplier for High Precision Neural Network Acceleration. In Proceedings of the 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, Italy, 27–30 May 2018; pp. 1–5.
16. Nakahara, H.; Sasao, T. A High-Speed Low-Power Deep Neural Network on an FPGA Based on the Nested RNS: Applied to an Object Detector. In Proceedings of the 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, Italy, 27–30 May 2018; pp. 1–5.
17. Molahosseini, A.S.; Asadpoor, A.; Zarandi, A.; Sousa, L. Towards Efficient Modular Adders Based on Reversible Circuits. In Proceedings of the 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, Italy, 27–30 May 2018; pp. 1–5.
18. Parhami, B. Computing with Logarithmic Number System Arithmetic: Implementation Methods and Performance Benefits. *Comput. Electr. Eng.* **2020**, *87*, 1–15. [[CrossRef](#)]
19. Matsenko, S.; Borysenko, O.; Spolitis, S.; Bobrovs, V. Noise Immunity of the Fibonacci Counter with the Fractal Decoder Device for Telecommunication Systems. *Latv. J. Phys. Tech. Sci.* **2019**, *56*, 12–21. [[CrossRef](#)]
20. Borysenko, O.; Matsenko, S.; Kulik, I.; Berezhna, O.; Matsenko, O. Optimal synthesis of digital counters in the Fibonacci codes with the minimal form of representation. *East. Eur. J. Enterp. Technol.* **2016**, *4*, 4–10. [[CrossRef](#)]
21. Borysenko, A. *Introduction to the Theory of Binomial Counting, Monograph*; Sumy ITD University Book: Sumy, Ukraine, 2004; p. 88.