IMPLEMENTATION OF THE ARITHMETIC ADDITION OPERATION IN THE SYSTEM OF RESIDUAL CLASSES

Abstract. The subject of the article is the development of a method for implementing the arithmetic operation of addition numbers that are represented in the system of residual classes (SRC). This method is based on the use of the principle of circular shift (PCS). The purpose of the article is to reduce the time it takes to implement the arithmetic operation of addition numbers that are represented in the SRC. Tasks: to analyze and identify the shortcomings of existing number systems that are used in the construction of computer systems and components, to explore possible ways to eliminate identified deficiencies, to evaluate the implementation time of the arithmetic addition operation based on the use of the developed method. Research methods: methods of analysis and synthesis of computer systems, number theory, coding theory in SRC. The following results are obtained. It is shown that the basic principle of the implementation of arithmetic operations in positional number systems is the use of binary adders. Moreover, the use of such number systems determines the presence of inter-bit connections, which in turn increases the time it takes to implement arithmetic operations, and also cause errors to occur. It is established that the use of a non-positional number system in residual classes allows us to eliminate these shortcomings, through the use of the basic properties of SRC. The peculiarity of the developed method is the result of the operation of adding numbers can be found by successive cyclic shifts of the bits of the contents of the data blocks by the corresponding SRC modules. The use of PCS allows eliminating the influence of inter-bit bonds between the residues of the processed numbers, which allows to reduce the implementation time of the operation of adding two numbers in the SRC. Conclusions. The results of the studies showed that the use of SRC allows to eliminate the existing shortcomings of positional number systems, and also improves the speed of implementation of arithmetic operations.

Keywords: number system; residual number system; circular shift register; speed of implementation of arithmetic operations; reliability of calculations; computer system; computer component.

Introduction

In positional binary number system (PBNS), the execution of the arithmetical addition operation presumes the sequential processing of digit positions of numbers by the rules determined by the content of this operation [1-3]. The processing continues until the values of all intermediate results (from low-order digit positions of a number to its high-order digit positions) are sequentially determined with allowance for all logical connections between binary places of summands. Thus, the PBNS in which data are represented and processed in modern computer systems and components (CSCs) possesses an essential drawback, namely, the presence of interbit connections between binary places of summands, which affects the speed and reliability of CSC calculations. Therefore, it is natural to search for the possibilities of creating and applying the machine arithmetic to be used in CSCs in which bitwise logical connections between binary places of the summands being processed would be weakened or would be absent at all. In this aspect, the nonpositional number system in residual classes (SRC) attracts attention [4-6].

Problem statement

The method of representing integer numbers in SRC possesses the following three basic properties of nonpositional code structures (NCSs): independence, equal rights, and a small bit capacity of residues whose totality determines NCSs in SRC. Using these properties opens ample opportunities in constructing not only a new machine arithmetic and methods for implementing modular arithmetic operations but also fundamentally new circuit implementation of CSCs in SRC [7-9].

The arithmetic operation of adding two numbers

\[ A = (a_1, a_2, ..., a_i, ..., a_n) \]

and

\[ B = (b_1, b_2, ..., b_i, ..., b_n) \]

in SRC is implemented by adding the corresponding residues \( a_i \) and \( b_i \) modulo \( m_i \) \((i = 1, n)\) independently of each of \( n \) bases. A small bit capacity of residues \( a_i \) and \( b_i \), in the representation of the numbers \( A = (a_1, a_2, ..., a_i, ..., a_n) \) and \( B = (b_1, b_2, ..., b_i, ..., b_n) \) being added allows to implement the modular addition operation \((a_i + b_i) \mod m_i\) using modulo adders of small bit capacity. In this case, the execution time of the operation of adding the numbers \( A = (a_1, a_2, ..., a_i, ..., a_n) \) and \( B = (b_1, b_2, ..., b_i, ..., b_n) \) is determined by the time necessary for obtaining the result of the operation \((a_i + b_i) \mod m_i\) based on the greatest base \( m_i \) of ordered SRC \((a_i < a_{i+1}, \text{ where } i = 1, n-1)\). Note the main drawbacks of the summation variant of implementing the modular arithmetic operation \((a_i + b_i) \mod m_i\) of addition in SRC.

1. The complexity of synthesizing binary \( k_i \)-digit adders \((k_i = \lfloor \log_2 (m_i - 1) \rfloor + 1) \mod m_i\).

2. Considerable time expenditures required when adding \((a_n + b_n) \mod m_n\) for large formats of CSC machine words determined in SRC by the value of \( m_n \), the largest of the bases.
3. A low reliability of calculating \((a_i + b_i) \mod m_i\), in view of possible errors arising in the process of determining values
\[
S_i = (a_i \oplus b_i) \mod 2 \lor c_{i-1}
\]
of sums in 1-bit binary positional adders as a result of possible errors arising in the process of formations of values
\[
C_{i+1} = a_{i+1} \land b_{i+1} \lor (a_{i+1} \lor b_{i+1}) \land c_i
\]
of carries between 1-bit binary adders from a low-order bit of a number to its high-order bit whose totality forms a \(k_i\)-bit binary adder modulo \(m_i\)
\[
(k_i = \lceil \log_2(m_i - 1) \rceil + 1)
\]
and also as a result of possible distortions of values \(C_{i+1}\) of carries when implementing the process of carrying intermediate values of the bit-by-bit summation \((a_i + b_i) \mod m_i\) within a given residue modulo \(m_i\) of SRC [2, 9].

The purpose of the article is to reduce the time of implementation of the arithmetic operation of addition numbers that are represented in the SRC.

Method for implementing the arithmetic addition operation in residue number system

In [10], the principle of technical implementation of integer-valued arithmetic operations modulo \(m_i\), namely, the principle of circular shift (PCS) is formulated. A peculiarity of methods for implementing arithmetic operations using PCS is that the result of, for example, the modular operation of adding numbers can be found by successive cyclic shifts of bits of the information content of a data block.

As has been noted, PCS can be used when implementing the arithmetic operation of adding numbers
\[
A = (a_1, a_2, ..., a_i, ..., a_n) \quad \text{and} \quad B = (b_1, b_2, ..., b_i, ..., b_n)
\]
in SRC. In this case, the result of the arithmetic addition operation in SRC is determined by a collection of residues \((a_i + b_i) \mod m_i\) modulo an arbitrary \(m_i\) of SRC; this collection consists of \(\{m_i\}\) bases.

The result of the operation in the residue modulo \(m_i\) of SRC in the adding circuit \((a_i + b_i) \mod m_i\) is determined only due to cyclic shifts of the information content of the circular shift register (CSR). In this case, there is no need (which is present in PBNS) for determining the values of
\[
S_i = (a_i \oplus b_i) \mod 2 \lor c_{i-1}
\]
and
\[
C_{i+1} = a_{i+1} \land b_{i+1} \lor (a_{i+1} \lor b_{i+1}) \land c_i.
\]

This allows to increase the speed of the implemented arithmetic operation of adding the numbers
\[
A = (a_1, a_2, ..., a_i, ..., a_n) \quad \text{and} \quad B = (b_1, b_2, ..., b_i, ..., b_n)
\]
in SRC and also to increase the reliability of implementing this operation.

Let \(G_1 = (Z, +)\) be the additive Abelian group of integers, and let \(G_2 = (Z_{m_i}, \oplus)\) be the additive Abelian group of residues of integers modulo \(m_i\). We specify a mapping \(F\) of the additive Abelian group of integers \(G_1\) onto the additive Abelian group of residues of integers \(G_2\) modulo \(m_i\). For any integer \(A\), its image \(F(A)\) is equal to the residue \(a_i = A \mod m_i\) of dividing the number \(A\) by the modulus \(m_i\) of SRC. It is easily verified that, for any integers (summands) \(A\) and \(B\), we have the equality
\[
F(A + B) = F(A) + F(B),
\]
i.e., for integers, the residue \((A + B) \mod m_i\), of dividing the sum of numbers \(A + B\) by die modulus \(m_i\), is equal to die sum modulo \(m_i\) of residues \(a_i = A \mod m_i\) and \(b_i = B \mod m_i\) of dividing each summand \(A\) and \(B\) by the modulus \(m_i\). Hence, the mapping \(F\) of the additive Abelian group of integers \(G_1\) onto the additive Abelian group of residues of integers modulo \(m_i\) is a homomorphism.

On this basis, it is possible to organize the process of determining the result of the modular arithmetic addition operation \((a_i + b_i) \mod m_i\) using PCS. In particular, a number in SRC is represented by a collection of \(n\) residues \(\{a_j\}\) obtained as a result of sequential division of the initial number \(A\) by \(n\) pairwise prime numbers (moduli) \(\{m_i\}\) for \(i = 1, n\) [1, 3]. In particular, among \(n\) given different natural numbers \(m_1, m_2, ..., m_n\) (SRC moduli), pairwise prime numbers are those of them for which the greatest common divisor (GCD) of any two different numbers is equal to 1, i.e., GCD \((m_i, m_j) = 1\) when \(i \neq j\).

We note some important data properties in a Cayley table. It follows from the existence of the neutral element in the field \(GF(m_i)\) that the Cayley table for implementing the addition \((a_i + b_i) \mod m_i\) contains a row in which elements of this field are in ascending order. The fact that these elements are different in the residue field \(GF(m_i)\) (the order of the group is equal to \(m_i\)) implies that each row (column) contains all elements of the field only once. According to the Cayley table, the necessary row of the table of \((a_i + b_i) \mod m_i\) of modular addition can be obtained by successive cyclic shift of elements of the first row. Circular shift registers that are often used in PBNS can be the systems engineering basis for synthesizing means (devices) to implement the method of arithmetic operation in SRC based on PCS.

The above-mentioned properties of a Cayley table allow to implement the operation of modular addition of numbers.
\[ A = (a_1, a_2, \ldots, a_n) \quad \text{and} \quad B = (b_1, b_2, \ldots, b_n) \]
in SRC using PCS in the adding circuit \((a_i + b_i) \mod m_i\), for each of \(n\) residues \(a_i\) of the number \(A = (a_1, a_2, \ldots, a_n)\). For each residue \(a_j\) from a number \(n\), \(m_j\) digit positions are formed (from the zeroth digit position to the \((m_j - 1)\)th digit position) of CSR. The number of bit positions in each CSR digit position is equal to
\[
\log_2 (m_j - 1) + 1.
\]
The presented method for implementing the arithmetic operation of adding numbers
\[ A = (a_1, a_2, \ldots, a_n) \quad \text{and} \quad B = (b_1, b_2, \ldots, b_n) \]
in SRC based on PCS is as follows.
1. Based on the number \(n\) of residues of the numbers
\[ A = (a_1, a_2, \ldots, a_n) \quad \text{and} \quad B = (b_1, b_2, \ldots, b_n) \]
in \(n\) adding circuits \((a_i + b_i) \mod m_i\) \((i = 1, n)\), the initial state of each CSR bit is set. To this end the values of the first row of Cayley table of modular addition \((a_i + b_i) \mod m_i\), are assigned in binary code to the corresponding CSR bits (from the zeroth to the \((m_j - 1)\)-th binary places (positions)).
2. Based on the value of \(a_i\), i.e., the first summand of the sum \((a_i + b_i) \mod m_i\) of the number \(A = (a_1, a_2, \ldots, a_n)\), the number of the digit position of the initial CSR information content is determined, and it is precisely this number that will represent the result of the modular operation \((a_i + b_i) \mod m_i\) in the \(i\)-th residue of SRC.
3. Based on the value of \(b_j\), i.e., the second summand of the sum \((a_i + b_i) \mod m_i\) of the number \(B = (b_1, b_2, \ldots, b_n)\), by \(1\) shifts of the initial content of each CSR digit position in the positive (counterclockwise) direction, the final CSR information content is obtained.
4. The obtained information contents of bits (from the zeroth to the \((m_i - 1)\)-th binary places) of all \(n\) CSRs determine the result of the operation of adding two numbers
\[ A = (a_1, a_2, \ldots, a_n) \quad \text{and} \quad B = (b_1, b_2, \ldots, b_n) \]
in SRC. The result of adding the two numbers
\[ A = (a_1, a_2, \ldots, a_n) \quad \text{and} \quad B = (b_1, b_2, \ldots, b_n) \]
is determined as the totality of \(n\) values of sums of residues \((a_i + b_i) \mod m_i\) \((i = 1, n)\) that are obtained in the corresponding CSR digit positions (see item 2).

In the general case, for an arbitrary base \(m_i\) of SRC, the initial digital, structure of CSR digit positions is determined as the content of the first row of the table of the modular addition \((a_i + b_i) \mod m_i\) and can be represented in the following form:

\[
P_{\text{ini}}^{(m_i)} = [P_0(a_0) \parallel P_1(a_1) \parallel \ldots \parallel P_{m_i-1}(a_{m_i-1})],
\]

where \(\parallel\) is the concatenation operation; \(P_0(a_i)\) – is the \(v\)-th \((v = 0, m_j - 1)\) CSR digit position; \(a_v\) is the information content of the \(v\)-th CSR digit position that is represented by an \(k\)-bit binary code \((k = \log_2 (m_j - 1) + 1)\) that corresponds to a possible value of the \(a_e\)-th residue \((a_e = 0, m_j - 1)\) in the binary code of the corresponding number modulo \(m_i\).

Thus, based on circular shift registers that are widely used in PBNS (in particular, in cryptography [11-13]), it is easy to implement the arithmetic operation of adding numbers in SRC. In this case, the degree \(Z\) of cyclic permutations (shifts) is defined based on structure (1) by the following expression:

\[
[P_0(a_0) \parallel P_1(a_1) \parallel \ldots \parallel P_{m_j-1}(a_{m_j-1})]^Z = [P_0(a_0) \parallel P_{v+1}(a_{z+1}) \ldots \parallel P_{m_j-1}(a_{m_j-1})] \quad (2)
\]

When technically implementing the proposed method, the first summand out of \(a_i\) determines the number of the CSR digit position whose content is the result of the operation \((a_i + b_i) \mod m_i\). The second summand out of \(b_i\) determines the number \(Z\) of CSR digit positions by which initial CSR content (1) should be shifted according to expression (2).

To comparatively analyze the time of executing the integer-valued arithmetic addition operation in PBNS and in SRC, we will calculate the execution time of the operation of adding two numbers \(A = (a_1, a_2, \ldots, a_n)\) and \(B = (b_1, b_2, \ldots, b_n)\) in SRC using PCS. For PCS, in the circuit of determining the value of \((a_i + b_i) \mod m_i\) \((i = 1, n)\), the time \(t\) of modular addition of two residues \(a_i\) and \(b_i\) is mainly determined by the time \(t_s\) of shifting the initial contents of CSR digit positions (in what follows, we assume that \(t = t_s\)).

The time of shifting the digital contents of CSR digit position is described by the expression

\[
t_s = Z \cdot k_i \cdot \tau,
\]

where \(Z\) is the number of the CSR digits being shifted; \(k_i = \log_2 (m_i - 1) + 1\) is the number of binary places in one CSR digit place (digit place) of die circuit for determining the residue of \((a_i + b_i) \mod m_i\); \(\tau = 3 \cdot \tau_g\) is the time of shifting one bit (the firing time of one trigger); \(\tau_g\) is the firing time of one logical gate (an element AND, NOT, OR).

Taking into account the aforesaid and also that \(t = t_s\), the time \(t\) of modular addition \((a_i + b_i) \mod m_i\) of two residues \(a_i\) and \(b_i\) is found as follows:

\[
t = 3 \cdot b_i \cdot \{\log_2 (m_i - 1) + 1\} \cdot \tau_g.
\]
Since \( b_i = 0, m_i - 1 \), the greatest possible value of \( t \) for an arbitrary modulus \( m_i \) of SRC (\( b_i = m_i - 1 = \max \)) is determined as follows:

\[
t = 3 \cdot (m_i - 1) \cdot \left[ \left\lceil \log_2 (m_i - 1) \right\rceil + 1 \right] \cdot \tau_g, \tag{5}
\]

and the minimal value of \( t \) equals zero (\( b_i = 0 \)). It is obvious that the time \( t \) of modular addition \((a_i + b_i) \mod m_i\) of two residues \( a_i \) and \( b_i \) in using PCS depends on the value of \( b_i \).

It is obvious that the time \( T_{\text{SRC}}^{(+)} \) of adding two numbers

\[
A = (a_1, a_2, ..., a_n) \quad \text{and} \quad B = (b_1, b_2, ..., b_n)
\]

in SRC based on the use of PCS is determined by the time \( t \) of executing the modular operation \((a_i + b_i) \mod m_i\) for which the following condition is satisfied:

\[
b_i \cdot k_i = \max, \tag{6}
\]

out of all possible values of paired products \( b_j \cdot k_j \) (\( j = 1, n; i \neq j \)). Condition (6) underlies the determination of the time \( T_{\text{SRC}}^{(+)} \) of executing the operation of adding two numbers

\[
A = (a_1, a_2, ..., a_n) \quad \text{and} \quad B = (b_1, b_2, ..., b_n)
\]

in SRC.

### Examples of executing the operation of addition of two numbers in SRC

We will consider examples of concrete execution of the operation of adding two numbers

\[
A = (a_1, a_2, ..., a_n) \quad \text{and} \quad B = (b_1, b_2, ..., b_n)
\]

in SRC for a one-byte \((l=1)\) adder. Let the value of \( p = 8 \cdot l \) be a quantity presented in binary places being processed in a positional adder of \( l \)-byte machine words (format of adder words). In PBNS, for \( l = 1 \) \((p = 8 \cdot l = 8 = 1 \) binary places), the SRC bases can be as follows: \( m_1 = 3, m_2 = 4, m_3 = 5 \) and \( m_4 = 7 \).

#### Example 1

Let the second summand be as follows: \( B = (10 \| 10 \| 100 \| 001) \). Then

- for \( m_1 = 3 \), we have \( b_1 = 2(10) \) and \( k_1 = \left\lceil \log_2 (m_1 - 1) \right\rceil + 1 = 2 \); \( b_1 \cdot k_1 = 2 \cdot 2 = 4 \);
- for \( m_2 = 4 \), we have \( b_2 = 2(10) \) and \( k_2 = \left\lceil \log_2 (m_2 - 1) \right\rceil + 1 = 2 \); \( b_2 \cdot k_2 = 2 \cdot 2 = 4 \);
- for \( m_3 = 5 \), we have \( b_3 = 4(100) \) and \( k_3 = \left\lceil \log_2 (m_3 - 1) \right\rceil + 1 = 3 \); \( b_3 \cdot k_3 = 4 \cdot 3 = 12 \);
- for \( m_4 = 7 \), we have \( b_4 = 1(001) \) and \( k_4 = \left\lceil \log_2 (m_4 - 1) \right\rceil + 1 = 3 \); \( b_4 \cdot k_4 = 1 \cdot 3 = 3 \).

As is obvious, the greatest number (i.e., 12) of the bits being shifted is obtained in the adding circuit \((a_3 + b_3) \mod m_3\) for adding modulo \( m_3 = 5 \).

Thus, for Example 1, the time of addition of two numbers \( A \) and \( B \) in SRC based on PCS is determined by the value of

\[
T_{\text{SRC}}^{(+)} = b_3 \cdot k_3 \cdot 3 \cdot \tau_g = 4 \cdot 3 \cdot 3 \cdot \tau_g = 36 \cdot \tau_g.
\]

#### Example 2

Let the second summand be as follows: \( B = (10, 11, 001, 001) \). Then

- for \( m_1 = 3 \), we have \( b_1 = 2(10) \) and \( k_1 = \left\lceil \log_2 (m_1 - 1) \right\rceil + 1 = 2 \); \( b_1 \cdot k_1 = 2 \cdot 2 = 4 \);
- for \( m_2 = 4 \), we have \( b_2 = 3(11) \) and \( k_2 = \left\lceil \log_2 (m_2 - 1) \right\rceil + 1 = 2 \); \( b_2 \cdot k_2 = 3 \cdot 2 = 6 \);
- for \( m_3 = 5 \), we have \( b_3 = 1(001) \) and \( k_3 = \left\lceil \log_2 (m_3 - 1) \right\rceil + 1 = 3 \); \( b_3 \cdot k_3 = 1 \cdot 3 = 3 \);
- for \( m_4 = 7 \), we have \( b_4 = 1(001) \) and \( k_4 = \left\lceil \log_2 (m_4 - 1) \right\rceil + 1 = 3 \); \( b_4 \cdot k_4 = 1 \cdot 3 = 3 \).

As is obvious, the greatest number (i.e., \( 6 \)) of the bits being shifted is obtained in the adding circuit \((a_2 + b_2) \mod m_2\) for adding modulo \( m_2 = 4 \). The time of addition of two numbers \( A \) and \( B \) in SRC based on PCS is determined by the value of

\[
T_{\text{SRC}}^{(+)} = b_2 \cdot k_2 \cdot 3 \cdot \tau_g = 2 \cdot 3 \cdot 3 \cdot \tau_g = 18 \cdot \tau_g.
\]

A peculiarity of using PCS in SRC is that the execution time of the addition operation depends on the value of the second summand \( b_i \), of the \((a_i + b_i) \mod m_i\), and can be different. The time \( T_{\text{SRC}}^{(+)} \) of adding two numbers

\[
A = (a_1, a_2, ..., a_n) \quad \text{and} \quad B = (b_1, b_2, ..., b_n)
\]

in SRC can be characterized by both the maximal \( T_{\text{SRC max}}^{(+)} \) time and a possible average addition time \( T_{\text{SRC avg}}^{(+)} \). Hereafter, the time of addition of two numbers

\[
A = (a_1, a_2, ..., a_n) \quad \text{and} \quad B = (b_1, b_2, ..., b_n)
\]

in SRC will be described by the expression \( T_{\text{SRC max}}^{(+)} \). Proceeding from expression (5), the maximal time of adding two numbers

\[
A = (a_1, a_2, ..., a_n) \quad \text{and} \quad B = (b_1, b_2, ..., b_n)
\]

in SRC is determined as

\[
T_{\text{SRC max}}^{(+)} = k_m \cdot \left\lceil \log_2 (m_i - 1) \right\rceil + 1 \cdot \tau_g. \tag{7}
\]

Note that there are algorithms that allowed to synthesize a number of technical solutions (devices) for implementing the integer modular arithmetic addition operation based on the use of PCS. In this case, the maximal time \( T_{\text{SRC max}}^{(+)} \) of executing the addition operation guaranteedly decreases at least twice [14]. In this case, the time \( T_{\text{SRC}}^{(+)} \) of adding two numbers

\[
A = (a_1, a_2, ..., a_n) \quad \text{and} \quad B = (b_1, b_2, ..., b_n)
\]

in SRC is described by the expression

\[
T_{\text{SRC}}^{(+)} = T_{\text{SRC max}}^{(+)} / 2.
\]
Let us perform an evaluative calculation and a comparative analysis of the execution time of the arithmetic operation of adding two numbers in PBNS and in SRC for one-byte \((l=1)\) and two-byte \((l=2)\) machine words. For \(l=1\) \((\rho=8\) binary places\), SRC is represented by the collection of the following bases: \(m_1 = 3\), \(m_2 = 4\), \(m_3 = 5\) and \(m_4 = 7\) and, for \(l=2\) \((\rho=16\) binary places\), SRC is represented by the following collection of bases: \(m_1 = 2\), \(m_2 = 3\), \(m_3 = 5\), \(m_4 = 7\), \(m_5 = 11\) and \(m_6 = 13\). When calculating, we will use formulas (7) - (9).

The performed comparative analysis of the results of calculations has shown the efficiency of using PCS in implementing the arithmetic operation of addition in SRC.

Conclusions

The modularity of structure of the computational process in SRC allows to use PCS to implement main modular integer arithmetic operations. Based on this, a method is developed in this article for implementing the arithmetic operation of addition in system of residual classes on the basis of using the principle of circular shift. Using PCS in SRC allows to eliminate the main drawback of existing CSCs operating in PBNS, namely, to get rid of the influence of interbit connections between binary places of numbers \(a_i\) and \(b_i\) in executing the operation of addition \((a_i+b_i)\mod m_i\), i.e., to eliminate the effect of interbit connections between \(l\)-bit positional adders on the result of the operation of addition \(A+B\) of two numbers. This article shows that the use of PCS allows to increase the operation speed of executing the operation \(A+B\) of addition of two numbers.

Using PCS can increase the reliability of calculations (i.e., the execution of the operation of modular addition in SRC) owing to eliminating possible errors from the process of executing the arithmetic operation of addition \((a_i+b_i)\mod m_i\); such errors can occur when determining the values of sums \(S_i\) and the values \(C_i\) of carry signals. The reliability of calculations can also be increased due to eliminating the possible effect of a distorted value of a carry signal \(C_i\) in the process of carrying this signal from the \(i\)th \(l\)-bit positional binary adder to the \((i+1)\)th \(l\)-bit positional binary adder. However, the statement about a possible increase in the reliability of executing the addition operation in SRC due to using PCS requires additional investigations. Note that the systems engineering basis for synthesizing means (devices) that implement the method of the arithmetic addition operation in SRC based on PCS can be shift registers that are widely used in PBNS.

References

1. Kuznetsov, O., Lutsenko, M. and Ivanenko, D. (2016), “Strukov stream cipher: Specification and basic properties”, Proc. 3rd Intern. Sci.-Pract. Conf. Problems of Informacommunications, Science and Technology (PICS&T), Kharkiv, pp. 59-62.
2. Krasnobayev, V. A. (2007), “Method for Realization of Transformations in Public-Key Cryptography”, Telecommunications and Radio Engineering, vol. 66, no. 17, pp. 1559–1572.
3. Tariq Jamil (2013), Complex Binary Number System. Algorithms and Circuits, India: Springer, 83 p.
4. Ananda Mohan (2016), Residue Number Systems, Birkhäuser Basel, 351 p.
5. Cheryyakov, N. I. (2017), “Residue-to-binary conversion for general moduli sets based on approximate Chinese remainder theorem”, International Journal of Computer Mathematics, vol. 94, No. 9, pp. 1833-1849.
6. Kasianchuk, M., Yakymenko, I., Pazdriy, I. and Zastavnyy, O. (2015), “Algorithms of findings of perfect shape modules of remaining classes system”, The Experience of Designing and Application of CAD Systems in Microelectronics, Lviv, pp. 316-318, DOI: https://doi.org/10.1109/CADSM.2015.720866.
7. Amir Sabbagh Molahosseini, Leonel Seabra de Sousa and Chip-Hong Chang (2017), Embedded Systems Design with Special Arithmetic and Number Systems, Springer International Publishing, 389 p.
8. Patterson, D.A. and Hennessy, J.L. (2016), Computer Organization and Design: The Hardware Software Interface: ARM Edition [The Morgan Kaufmann Series in Computer Architecture and Design], 1st Edition: Morgan Kaufmann, 720 p.
9. Yadin, A. (2016), Computer Systems Architecture, CRC Press, 526 p.
10. Valvano, J. (2017), Embedded Systems: Real-Time Operating Systems for Arm Cortex M Microcontrollers [2nd ed. edition], CreateSpace Independent Publishing Platform, 486 p.
11. Hsu, J. Y. (2017), Computer Architecture: Software Aspects, Coding, and Hardware, CRC Press, 456 p.
12. Zimbauer, M. R. (2018), “Symmetry classes”; Oxford Handbooks Online, DOI: https://doi.org/10.1093/oxfordhb/9780198744191.013.3
13. Krasnobayev, V.A., Koshman, S.A. and Mavrina, M.A. (2014), “A method for increasing the reliability of verification of data represented in a residue number system”, Cybernetics and Systems Analysis, vol. 50, Issue 6, pp. 969-976.
14. Krasnobayev, V.A., Yanko, A.S. and Koshman S.A. (2016), “A Method for arithmetic comparison of data represented in a residue number system”, Cybernetics and Systems Analysis, vol. 52, Issue 1, pp. 145-150.
Красnobаев Віктор Анатолійович – доктор технічних наук, професор, професор кафедри електроніки і управляючих систем, Харківський національний університет імені В. Н. Каразіна, Харків, Україна;
Victor Krasnobaev – Doctor of Technical Sciences, Professor, Professor of Electronics and Control Systems Department, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine;
e-mail: v.a.krasnobaev@gmail.com; ORCID ID: http://orcid.org/0000-0001-5192-9918.

Кошман Сергій Олександрович – доктор технічних наук, доцент, доцент кафедри безпеки інформаційних систем і технологій, Харківський національний університет імені В. Н. Каразіна, Харків, Україна;
Sergey Kosman – Doctor of Technical Sciences, Associate Professor, Associate Professor of Information Systems and Technologies Security Department, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine;
e-mail: s.kosman@karazin.ua; ORCID ID: http://orcid.org/0000-0001-8934-2274.

Курчанов Валерій Микитович – кандидат технічних наук, доцент, доцент кафедри побудови телекомуникаційних систем, Військовий коледж сержантського складу, Полтава, Україна;
Valerii Kurchanov – Candidate of Technical Sciences, Associate Professor, Associate Professor of Construction of Telecommunication Systems Department, NCO Military College, Poltava, Ukraine;
e-mail: kvnkvn@meta.ua; ORCID ID: http://orcid.org/0000-0002-4736-6769.

Реалізація арифметичної операції додавання у системі залишкових класів

Цікавося слова: система числення; система залишкових класів; кільцевий реєстр зсуву; швидкодія виконання арифметичних операцій; достовірність обчислень; комп’ютерна система; комп’ютерний компонент.

Реалізація арифметичної операції додавання в системе остаточных класов

В. А. Красnobаев, С. О. Кошман, В. М. Курчанов

Анотація. Предметом статті є розробка методу реалізації арифметичної операції додавання числ, які представлені у системі залишкових класів (СЗК). Даний метод працює на основі використання принципу кільцевого зсуву (ПКЗ). Метою статті є зменшення часу реалізації арифметичної операції додавання числ, які представлені у СЗК.

Заключні результати показано, що основним принципом реалізації арифметичних операцій у позиційних системах числення, є використання двійкових суматорів. При цьому використання таких систем числення обумовлює наявність міжрозрядних зв’язків, що з усього часу реалізації арифметичної операції додавання аніміт, а також зумовлює виникнення помилок. Встановлено, що використання непозиційної системи числення у залишкових класах дозволяє зменшити час реалізації операції додавання двох чисел у СЗК. Висновки. Результати проведених досліджень показали, що використання СЗК дозволяє зменшити відносні недоліки позиційних систем числення, а також підвищує швидкість реалізації арифметичних операцій.

Ключові слова: система числення; система залишкових класів; кільцевий реєстр зсуву; швидкодія виконання арифметичних операцій; достовірність обчислень; комп’ютерна система; комп’ютерний компонент.