DOI: 10.15514/ISPRAS-2026-38(3)-19

High Speed Algorithm for Number Sign Detection in Residue Number System Based on Akushsky Core Function

V.V. Lutsenko, ORCID: 0000-0003-4648-8286 <officialvladlutsenko@gmail.com>
A.E. Geryugova, ORCID: 0009-0005-8389-2204 <ajsanatgerugova@gmail.com>
M.G. Babenko, ORCID: 0000-0001-7066-0061 <mgbabenko@ncfu.ru>

North-Caucasus Federal University, Stavropol,
1, Pushkin st., Stavropol, 355017, Russia.

Abstract. This paper proposes a high-speed algorithm for sign detection in the residue number system based on the Akushsky core function. The method utilizes a set of moduli $\{\mathbf{2^n - 1}, \mathbf{2^{n+1} - 1}, \mathbf{2^{n+a}}\}$ to efficiently determine the sign of a number. Key advantages include reduced operand sizes and replacement of costly modulo operations with efficient bitwise manipulations. Experimental results show that the Akushsky core function-based approach outperforms the traditional method, achieving an average speedup of 25.6%. The algorithm shows consistent performance across all tested bit widths, making it particularly suitable for applications requiring high-speed residue number system arithmetic, such as digital signal processing and cryptography.

Keywords: residue number system; high performance computing; Akushsky core function; sign detection.

For citation: Lutsenko V.V., Geryugova A.E., Babenko M.G. High Speed Algorithm for Number Sign Detection in Residue Number System Based on Akushsky Core Function. Trudy ISP RAN/Proc. ISP RAS, vol. 38, issue 3, part 2, 2026, pp. 15-32. DOI: 10.15514/ISPRAS-2026-38(3)-19.

Acknowledgements. The research was supported by the Russian Science Foundation Grant No. 25-71-30007.

Высокоскоростной алгоритм определения знака числа в системе остаточных классов на основе функции ядра Акушского

В.В. Луценко, ORCID: 0000-0003-4648-8286 <officialvladlutsenko@gmail.com>
А.Э. Герюгова, ORCID: 0009-0005-8389-2204 <ajsanatgerugova@gmail.com>
М.Г. Бабенко, ORCID: 0000-0001-7066-0061 <mgbabenko@ncfu.ru>

Северо-Кавказский федеральный университет,
355017, Россия, г. Ставрополь, ул. Пушкина, д. 1.

Аннотация. В данной статье предлагается высокоскоростной алгоритм определения знака в системе остаточных классов на основе функции ядра Акушского. Метод использует набор модулей $\{\mathbf{2^n - 1}, \mathbf{2^{n+1} - 1}, \mathbf{2^{n+a}}\}$ для эффективного определения знака числа. Ключевые преимущества включают в себя уменьшение размеров операндов и замену вычислительно сложных операций по модулю эффективными побитовыми операциями. Экспериментальные результаты показывают, что алгоритм на основе функции ядра Акушского, превосходит классические методы, достигая среднего ускорения в 25.6%. Алгоритм демонстрирует стабильную производительность во всех протестированных разрядностях, что делает его подходящим для приложений, требующих высокоскоростной арифметики системы остаточных классов, таких как цифровая обработка сигналов и криптография.

Ключевые слова: система остаточных классов; высокопроизводительные вычисления; функция ядра Акушского; обнаружение знака.

Для цитирования: Луценко В.В., Герюгова А.Э., Бабенко М.Г. Высокоскоростной алгоритм определения знака числа в системе остаточных классов на основе функции ядра Акушского. Труды ИСП РАН, том 38, вып. 3, часть 2, 2026 г., стр. 15–32 (на английском языке). DOI: 10.15514/ISPRAS-2026-38(3)-19.

Благодарности. Исследование выполнено за счет гранта Российского научного фонда № 25-71-30007.

1. Introduction

The Residue Number System (RNS) has emerged as a powerful tool for high-performance computing, particularly in cryptographic applications where parallel arithmetic operations are paramount. By decomposing numbers into residues across pairwise co-prime moduli, RNS enables carry-free addition, subtraction, and multiplication [1]. This inherent parallelism makes RNS particularly attractive for modern cryptographic implementations, including homomorphic encryption schemes [2], lattice-based cryptography [3], and side-channel attack resistant designs [4].

Despite these advantages, RNS faces significant challenges in handling non-modular operations, with sign detection representing one of the most critical yet computationally intensive tasks [5]. The inability to efficiently determine number signs directly in RNS has limited its adoption in cryptographic applications requiring signed arithmetic operations, range validation, and error detection [6]. Current approaches typically resort to conversion to Weighted Number Systems (WNS), which not only negates the parallel processing benefits of RNS but also introduces substantial computational overhead [7].

This work proposes a novel algorithm for determining the sign of a number in RNS, based on the Akushsky core function. The proposed algorithm reduces the operand size compared to known sign detection methods in RNS and eliminates the computationally complex operation of finding the division remainder. The key elements of novelty are:

1. The adaptation and application of the Akushsky core function to solve the problem of number sign detection in RNS for the specific moduli set $\{2^n - 1, 2^{n+a}, 2^n + 1\}$.
2. The proof of minimality for the core function with the specified properties for the moduli set $\{2^n - 1, 2^{n+a}, 2^n + 1\}$.

3. The development of a sign detection algorithm based on the proposed core function.

The objective of this research is to develop a high-speed algorithm for determining the sign of a number in RNS for the moduli set $\{2^n-1, 2^{n+a}, 2^n+1\}$, utilizing the Akushsky core function.

To achieve this objective, the following tasks were set:

1. To analyze existing methods for determining the sign of a number in RNS.
2. To derive a formal condition that links the value of the core function to the sign of the number represented in RNS for the given moduli set.
3. To develop a minimal core function with the required properties for number sign detection for the moduli set $\{2^n-1, 2^{n+a}, 2^n+1\}$.
4. To conduct a performance analysis of the proposed method in comparison with known sign detection algorithms.

The content of the paper is organized as follows. Section 2 briefly introduces RNS. Section 3 presents methods for determining the sign of RNS. Section 4 describes the basics of the Akushsky core function. Section 5 presents an algorithm for determining the sign of a number in RNS using the Akushsky core function for a set of moduli $\{2^n-1, 2^{n+a}, 2^n+1\}$. The performance evaluation of the proposed algorithm is described in Section 6. In Section 7, we draw conclusions and discuss future work.

2. Residue Number System

Definition 1. A basis of the residue number system is a set of moduli $\{p_1, p_1, \dots, p_n\}$, where each modulo $p_i \geq 2 (i = 1,2, \dots, n)$ and $gcd(p_i, p_j) = 1$ for $i \neq j$. Here and throughout, gcd denotes the greatest common divisor. By convention, the moduli are ordered in ascending order: $p_1 < p_1 < \dots < p_n$.

Definition 2. The product of moduli $P = \prod_{i=1}^{n} p_i$ is called the dynamic range of RNS.

Definition 3. An integer $X \in [0, P)$ can be represented as an n-dimensional vector of least non-negative residues:

$$X = (x_1, x_2, \dots, x_n),$$

where $x_i \equiv X \; mod \; p_i$, denoted as $x_i = |X|_{p_i}$.

To handle negative numbers, the range is divided into two intervals. A number X then satisfies:

$$\begin{cases} -\dfrac{P-1}{2} \leq X \leq \dfrac{P-1}{2}, & if \; P \equiv 1 \; mod \; 2, \\ -\dfrac{P}{2} \leq X \leq \dfrac{P}{2} - 1, & if \; P \equiv 0 \; mod \; 2. \end{cases}$$

Consider RNS with the basis $\{3, 4\}$. In this basis, numbers from the interval $[-6,6)$ can be uniquely represented, since $P = 3 \times 4 = 12$. If $X = (x_1, x_2, \dots, x_n)$, then the negative number $-X = (\overline{x_1}, \overline{x_2}, \dots, \overline{x_n})$, where $\overline{x_i}$ is the complement of x_i modulo p_i. For the RNS $\{3, 4\}$ and the number $X = (1,1)$, we get $-X = (3-1, 4-1) = (2, 3)$.

Table 1 shows the correspondence between numbers in positional notation and residue number system.

Table 1. Number representation for RNS with basis $\{3, 4\}$.

$-6 \overset{RNS}{\longrightarrow} (0,2)$	$-5 \overset{RNS}{\longrightarrow} (1,3)$	$-4 \overset{RNS}{\longrightarrow} (2,0)$
$-3 \overset{RNS}{\longrightarrow} (0,1)$	$-2 \overset{RNS}{\longrightarrow} (1,2)$	$-1 \overset{RNS}{\longrightarrow} (2,3)$
$0 \overset{RNS}{\longrightarrow} (0,0)$	$1 \overset{RNS}{\longrightarrow} (1,1)$	$2 \overset{RNS}{\longrightarrow} (2,2)$
$3 \overset{RNS}{\longrightarrow} (0,3)$	$4 \overset{RNS}{\longrightarrow} (1,0)$	$5 \overset{RNS}{\longrightarrow} (2,1)$

RNS has the distinctive feature of performing addition, subtraction, and multiplication operations in parallel and independently for each modulo. For numbers $X = (x_1, x_2, \dots, x_n)$ and $Y = (y_1, y_2, \dots, y_n)$, arithmetic operations can be expressed as:

$$X * Y = \left(|x_1 * y_1|_{p_1}, |x_2 * y_2|_{p_2}, \dots, |x_n * y_n|_{p_n}\right), \quad (1)$$

where $* \in \{+, -, \times\}$.

The process of performing modular operations in RNS can be described by Algorithm 1.

Algorithm 1: Modular operations in RNS.

Input: $\{p_1, p_2, \dots, p_n\}, (x_1, x_2, \dots, x_n), (y_1, y_2, \dots, y_n), \; \circ \in \{+, -, \times\}$
Output: $R = (r_1, r_2, \dots, r_n)$
1. **for** $i = 1, i \leq n, i++$ **do**:
1.2 $r_i = |x_i \circ y_i|_{p_i}$

Example 1 demonstrates the addition of two numbers in RNS.

Example 1. Let's add two numbers $X = -2$ and $Y = 3$ in the basis $\{3, 4\}$. Their representation in this basis can be found in Table 1. Using equation 1 for addition:

$$X + Y = (|1+0|_3, |2+3|_4) = (1,1).$$

This example clearly demonstrates the advantages of modular arithmetic. The considered range allows working with numbers up to 4 bits. However, the moduli 3 and 4 themselves require only 3 bits for encoding, enabling parallel computations of smaller bit-width for each modulo. This property is particularly valuable in cryptography: instead of processing 2048-bit numbers as in modern encryption algorithms, one can use 33 parallel RNS channels with 32-bit moduli [8-9].

3. Methods of Sign Detection

Given that $\forall X \in N$ and $X \leq \frac{P}{2}: P - X \equiv -X \; mod \; P$, in the complete system of least non-negative residues modulo P, the sign of a number in RNS for even dynamic range is determined by:

$$Sign(X) = \begin{cases} 0, & if \; 0 \leq X < \dfrac{P}{2}, \\ 1, & if \; \dfrac{P}{2} \leq X < P. \end{cases} \quad (2)$$

For odd dynamic range, the sign is determined by:

$$Sign(X) = \begin{cases} 0, & if \; 0 \leq X < \dfrac{P+1}{2}, \\ 1, & if \; \dfrac{P+1}{2} \leq X < P. \end{cases} \quad (3)$$

The sign of a number can be determined using reverse conversion methods.

To recover the positional representation of a number in RNS, one can use the Chinese Remainder Theorem (CRT) [10].

Theorem 1: Let $\{p_1, p_2, \dots, p_n\}$ be pairwise coprime natural numbers and $P = \prod_{i=1}^{n} p_i$. Any number X such that $0 \leq X < P$ can be uniquely represented as a sequence $(x_1, x_2, \dots, x_n)$, where $x_i = X \; mod \; p_i$, and

$$X = \left| \sum_{i=1}^{n} P_i \cdot x_i \cdot |P_i^{-1}|_{p_i} \right|_P, \quad (4)$$

where $P_i = \frac{P}{p_i}$ and $|P_i^{-1}|_{p_i}$ is the multiplicative inverse of P_i modulo p_i.

Definition 4. The values $B_i = |P_i^{-1}|_{p_i} \cdot P_i$ are called the orthogonal bases of the RNS.

Using the CRT, the sign detection can be expressed as Algorithm 2.

Algorithm 2: Sign detection using CRT.

Input: $\{p_1, p_2, \dots, p_n\}, (x_1, x_2, \dots, x_n)$
Output: $sign$
1. $P = \prod_{i=1}^{n} p_i$
2. $K = \begin{cases} \frac{P}{2}, if\ P\ even, \\ \frac{P+1}{2}, if\ P\ odd \end{cases}$ # Threshold
3. $P_i = \frac{P}{p_i}$
4. $B_i = |P_i^{-1}|_{p_i} \cdot P_i, \quad i = 1,2, \dots, n$
5. $sum = 0$
6. **for** $i = 1$ **to** n
6.1. $sum = sum + x_i \cdot B_i$
7. $X = sum\ mod\ P$
8. **if** $X < K$ **then**
8.1. $sign = 0$ # Non-negative
9. **else**
9.1. $sign = 1$ # Negative
10. **return** $sign$

Example 2 demonstrates sign detection using CRT.

Example 2. Consider an RNS with moduli $p_1 = 3, p_2 = 7, p_3 = 8$. The dynamic range is $P = 3 \cdot 7 \cdot 8 = 168$. The range midpoint is $K = \frac{P}{2} = 84$. Let us determine the sign of the number $X = (2,0,1)$ using CRT.

The values of P_i are:

$$P_1 = \frac{P}{p_1} = 56, \quad P_2 = 24, \quad P_3 = 21.$$

The multiplicative inverses are:

$$|P_1^{-1}|_{p_1} = 2, \quad |P_2^{-1}|_{p_2} = 5, \quad |P_3^{-1}|_{p_3} = 5.$$

Using these values, we can compute X via (4):

$$X = |56 \cdot 2 \cdot 2 + 24 \cdot 0 \cdot 5 + 21 \cdot 1 \cdot 5|_{168} = 161.$$

Since $161 > 84$, the number $(2,0,1)$ is negative.

Approximate Chinese Remainder Theorem (ACRT) [5] are also used to determine the sign of a number. This method is analogous to Algorithm 2.

The sign of a number can be determined using a Diagonal Function (DF) without performing reverse conversion to a positional number system [11]**.** The sign determination algorithm using the DF is presented in Algorithm 3.

Consider an example of sign determination using the diagonal function.

Example 3. Consider an RNS with moduli $p_1 = 3, p_2 = 7, p_3 = 8$. The dynamic range is $P = 168, K = \frac{P}{2} = 84, D(K) = 50$. Let us determine the sign of number $X = (2,0,1)$ using the diagonal function.

The partial products are:

$$P_1 = \frac{P}{p_1} = 56, \quad P_2 = 24, \quad P_3 = 21,$$
$$SQ = P_1 + P_2 + P_3 = 101.$$

The diagonal coefficients are:

$$\widetilde{k_1} = |-3^{-1}|_{101} = 67, \quad \widetilde{k_2} = 72, \quad \widetilde{k_3} = 63.$$

Computing $D(X)$ yields:

$$D(X) = |67 \cdot 2 + 72 \cdot 0 + 63 \cdot 1|_{101} = 96.$$

Since $96 > 50$, the number $(2,0,1)$ is negative.

Algorithm 3: Sign detection using DF.

Input: $\{p_1, p_2, \dots, p_n\}, (x_1, x_2, \dots, x_n)$
Output: $sign$
1. $P = \prod_{i=1}^{n} p_i$
2. $K = \begin{cases} \frac{P}{2}, if\ P\ even, \\ \frac{P+1}{2}, if\ P\ odd \end{cases}$ # Threshold value
3. $D(K) = \sum_{i=1}^{n} \left\lfloor \frac{K}{p_i} \right\rfloor$ # Diagonal function at threshold
4. $P_i = \frac{P}{p_i}$ # Partial products
5. $SQ = \sum_{i=1}^{n} P_i$ # Sum of partial products
6. $\widetilde{k_i} = |-p_i^{-1}|_{SQ}, \quad i = 1,2, \dots, n$ # Diagonal coefficients
7. $sum = 0$
8. **for** $i = 1$ **to** n
8.1. $sum = sum + x_i \cdot \widetilde{k_i}$
9. $D(X) = sum\ mod\ SQ$
10. **if** $D(X) < D(K)$
10.1. $sign = 0$ # Non-negative
11. **else**
11.1. $sign = 1$ # Negative
12. **return** $sign$

4. Akushsky Core Function

Research conducted by Akushsky I.Ya., Burtsev V.M. and Pak I.T. [12] focused on developing positional characteristics of numbers in RNS. These studies resulted in a novel mathematical construct known as the Akushsky Core Function (ACF), defined by the following equation:

$$C(X) = \sum_{i=1}^{n} w_i \cdot \left\lfloor \frac{X}{p_i} \right\rfloor = \sum_{i=1}^{n} (Xx_i) \cdot \frac{w_i}{p_i}, \tag{5}$$

where integer coefficients w_i serve as constant values determined by the choice of interpolation points. These weighting constants w_i determine the contribution of each partial term $\left\lfloor \frac{X}{p_i} \right\rfloor$ in formula (5), thereby defining the core function's properties and behavior.

Substituting $X = P$ into (5) yields:

$$C(P) = C_P = \sum_{i=1}^{n} w_i \cdot \left\lfloor \frac{P}{p_i} \right\rfloor = \sum_{i=1}^{n} w_i \cdot P_i. \tag{6}$$

Dividing $C(P)$ by P gives:

$$\frac{C(P)}{P} = \sum_{i=1}^{n} \frac{w_i}{p_i}. \tag{7}$$

This implies that to obtain small values of $C(P)$, some coefficients w_i must be negative. Substituting (7) into (5) produces:

$$C(X) = X \cdot \sum_{i=1}^{n} \frac{w_i}{p_i} - \sum_{i=1}^{n} \frac{x_i \cdot w_i}{p_i} = X \cdot \frac{C(P)}{P} - \sum_{i=1}^{n} \frac{x_i \cdot w_i}{p_i}. \quad (8)$$

The core function enables extraction of positional number characteristics as illustrated in Figure 1. It provides a mapping from RNS numbers to a coordinate line, where C_{min} and C_{max} represent the minimum and maximum values of the core function for a given set of weights.

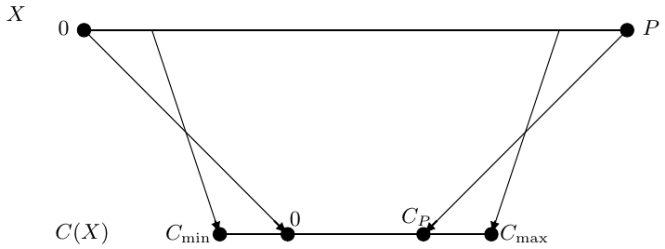

Fig. 1. Mapping $0 \le X < P$ *to* $C_{min} \le C(X) \le C_{max}$.

Equation (8) indicates that the plot of $C(X)$ versus X should form a straight line with slope $C(P)/P$, exhibiting some nonlinearity. The degree of nonlinearity is determined by the weight values, which are in turn related to the specific value of $C(P)$ for the given RNS moduli set.

Considering that $P_j \equiv 0 \; mod \; p_i$ for any $i \neq j$, the weights w_i of the function $C(X)$ can be determined by the relation:

$$w_i \equiv |C(P) \cdot P_i^{-1}|_{p_i}. \quad (9)$$

Thus, the weights can be determined after selecting $C(P)$, but must satisfy condition (6).

The value of the core function $C(X)$, with $w_1, w_2, \dots, w_n$, under the condition $0 \le C(X) < C_P, X \in [0, P)$, can be computed using the formula

$$C(X) \equiv \left| \sum_{i=1}^{n} k_i \cdot x_i \right|_{C_P}, \quad (10)$$

where $k_i = C(B_i)$.

Using (2) and (3), one can determine the sign of a number in RNS based on the core function.

$$Sing(X) = \begin{cases} 0, & if \;\; C(X) = \left| \sum_{i=1}^{n} k_i \cdot x_i \right|_{C(P)} < C(K), \\ 1, & if \;\; C(X) = \left| \sum_{i=1}^{n} k_i \cdot x_i \right|_{C(P)} \ge C(K), \end{cases} \quad (11)$$

where $K = \frac{P}{2}$ when P is even, or $K = \frac{P+1}{2}$ when P is odd.

5. Algorithm for Number Sign Detection Using ACF for a set of moduli $\{2^n - 1, 2^{n+1} - 1, 2^{n+a}\}$

Consider the minimal Akushsky core function for a set of moduli $\{2^n - 1, 2^{n+1} - 1, 2^{n+a}\}$, where $0 \le a < n$. We define the core function as:

$$C(X) = w_1 \left\lfloor \frac{X}{2^n - 1} \right\rfloor + w_2 \left\lfloor \frac{X}{2^{n+1} - 1} \right\rfloor + w_3 \left\lfloor \frac{X}{2^{n+a}} \right\rfloor,$$

with $C_P = C(P) = w_1 P_1 + w_2 P_2 + w_3 P_3$, where $P = (2^n - 1)(2^{n+1} - 1)2^{n+a}$,

$P_1 = (2^{n+1} - 1)2^{n+a}, P_2 = (2^n - 1)2^{n+a}, P_3 = (2^n - 1)(2^{n+1} - 1)$, and $w_1, w_2, w_3 \in Z$.

We investigate the constraints on the coefficients w_1, w_2, w_3 of the core function $C(X)$ that satisfy the condition $\forall X \in [0, P): 0 \le C(X) \le C_P$. To establish this, we prove Lemmas 1 and 2.

Lemma 1: If $\forall X \in [0, P): 0 \le C(X) \le C_P$, then

$$\begin{cases} 0 \le w_1 \le C_P, \\ 0 \le w_1 + w_2 + w_3 \le C_P \end{cases}$$

Proof: Let us compute $C(2^n - 1)$ and $C(P - 1)$:

$$C(2^n - 1) = w_1,$$

$$C(P - 1) = w_1(P_1 - 1) + w_2(P_2 - 1) + w_3(P_3 - 1) = C_P - w_1 - w_2 - w_3.$$

Since $0 \le C(2^n - 1) \le C_P$ and $0 \le C(P - 1) \le C_P$, it follows that: $0 \le w_1 \le C_P$ and $0 \le C_P - w_1 - w_2 - w_3 \le C_P$, which implies: $0 \le w_1 \le C_P$ and $0 \le w_1 + w_2 + w_3 \le C_P$.

The lemma is proven.

Lemma 2: If $\forall X \in [0, P): 0 \le C(X) \le C_P$ in an RNS $\{2^n - 1, 2^{n+1} - 1, 2^{n+a}\}$, the following holds:

For $a = 0$: $0 \le w_1 + w_3 \le C_P$ and $0 \le 2w_1 + w_2 + w_3 \le C_P$.

For $a \ge 1$: $0 \le 2w_1 + w_2 \le C_P$ and $0 \le 2^a w_1 + 2^{a-1} w_2 + w_3 \le C_P$.

Proof: When $a = 0, 2^{n+a} = 2^n$. Computing $C(2^n)$ and $C(2^{n+1} - 1)$ yields:

$$C(2^n) = w_1 + w_3,$$

$$C(2^{n+1} - 1) = 2w_1 + w_2 + w_3.$$

For $a \ge 1$, computing $C(2^{n+1} - 1)$ and $C(2^{n+a})$ gives:

$$C(2^{n+1} - 1) = 2w_1 + w_2,$$

$$C(2^{n+a}) = w_1 \left\lfloor \frac{2^{n+a}}{2^n - 1} \right\rfloor + w_2 \left\lfloor \frac{2^{n+a}}{2^{n+1} - 1} \right\rfloor + w_3$$

$$= 2^a w_1 + 2^{a-1} w_2 + w_3.$$

Since $\forall X \in [0, P): 0 \le C(X) \le C_P$, it follows that $0 \le w_1 + w_3 \le C_P, 0 \le 2w_1 + w_2 + w_3 \le C_P$ when $a = 0$, and $0 \le 2w_1 + w_2 \le C_P, 0 \le 2^a w_1 + 2^{a-1} w_2 + w_3 \le C_P$ when $a \ge 1$.

The lemma is proven.

Lemma 3: If $\forall X \in [0, P): 0 \le C(X) \le C_P$ and $C_P = 2^b$, then $w_1 \ge 1$.

Proof: Since $C_P = 2^b$, we have:

$$w_1 P_1 + w_2 P_2 + w_3 P_3 = 2^b.$$

Assuming $w_1 = 0$ would imply $(2^n - 1) | 2^b$, which is false for all $n \ge 2$. Therefore, the assumption is false and $w_1 \ne 0$. From Lemma 1, we have $0 \le w_1 \le C_P$ and $w_1 \ne 0$, hence $w_1 \ge 1$.

The lemma is proven.

Theorem 2: If for the function

$$C(X) = w_1 \left\lfloor \frac{X}{2^n - 1} \right\rfloor + w_2 \left\lfloor \frac{X}{2^{n+1} - 1} \right\rfloor + w_3 \left\lfloor \frac{X}{2^{n+a}} \right\rfloor,$$

the following conditions hold: $\forall X \in [0, P): 0 \le C(X) \le C_P < P$ and $C_P = 2^b$, then $b \ge 2n + a$.

Proof: Consider two cases.

Case 1. If a=0, then C_P can be expressed as:

$$C_P = 2^n(2^n - 1)(w_1 + w_2 + w_3) + (2^n - 1)^2(w_1 + w_3) + (2^{n+1} - 1)w_1. \quad (12)$$

From Lemmas 1, 2 and 3, it follows that: $w_1 + w_2 + w_3 \ge 0, w_1 + w_3 \ge 0$ and $w_1 \ge 1$, hence: $C_P \ge 2^{n+1} - 1$ and $b \ge n + 1$.

Considering that

$$\begin{cases} w_1 \equiv 2^b \ (mod 2^n - 1), \\ w_2 \equiv -2^{b+2} (mod 2^{n+1} - 1), \\ w_3 \equiv 0 (mod 2^n). \end{cases}$$

then $w_1 = q_1(2^n - 1) + 2^{|b|_n}, w_2 = q_2(2^{n+1} - 1) - 2^{|b+2|_{n+1}}$ and $w_3 = q_3 2^n$.

Note that for $b = 2n$ and $w_1 = 1, w_2 = -1, w_3 = 0$, the conditions of Lemmas 1, 2 and 3 are satisfied.

Suppose there exist w_1, w_2 and w_3 satisfying the conditions of Lemmas 1, 2 and 3 with $n + 1 \leq b < 2n$. Then $b = n + |b|_n$ and $|b + 2|_{n+1} = |b|_n$,hence: $w_1 = q_1(2^n - 1) + 2^{|b|_n}, w_2 = q_2(2^{n+1} - 1) - 2^{|b|_n}$ and $w_3 = q_3 2^n$.

From (12) it follows that $(2^{n+1} - 1)w_1 \leq 2^{2n}$, hence:

$$0 \leq (2^{n+1} - 1)\left(q_1(2^n - 1) + 2^{|b|_n}\right) \leq 2^{2n},$$

$$-2^{|b|_n} \leq q_1(2^n - 1) \leq \frac{2^{2n}}{2^{n+1}-1} - 2^{|b|_n} < 2^{n-1}.$$

Therefore, $q_1 = 0$ and $w_1 = 2^{|b|_n}$.

From (12), if $w_1 + w_3 \geq 1$ then $C_P \geq 2^{2n}$, which contradicts the assumption $n + 1 \leq b < 2n$. Hence $w_1 + w_3 = 0$ and $2^{|b|_n} + q_3 2^n = 0$. Since $2^{|b|_n} \neq 0$ and $1 + q_3 2^{n-|b|_n} \neq 0$, we have $2^{|b|_n}\left(1 + q_3 2^{n-|b|_n}\right) = 2^{|b|_n} + q_3 2^n \neq 0$, leading to a contradiction. Therefore, there exist no w_1, w_2 and w_3 satisfying the conditions of Lemmas 1, 2, 3 with $n + 1 \leq b < 2n$. Thus, if $\forall X \in [0, P)$: $0 \leq C(X) \leq C_P < P$ and $C_P = 2^b$, then $b \geq 2n$.

Case 2. If $a \geq 1$, then C_P can be expressed as:

$$\begin{aligned} C_P = {} & (2^n - 1)(2^{n+1} - 1)(w_1 + w_2 + w_3) \\ & + (2^n - 1)(2^{n+a} - 2^{n+1} + 1)(2w_1 + w_2) \\ & + (2^{2n+1} + (2^a - 3)2^n + 1)w_1 \end{aligned} \quad (13)$$

From Lemmas 1, 2 and 3, it follows that: $w_1 + w_2 + w_3 \geq 0, 2w_1 + w_2 \geq 0$ and $w_1 \geq 1$, hence:

$$C_P \geq 2^{2n+1} + (2^a - 3)2^n + 1 \geq 2^{2n+1} - 2^n + 1 > 2^{2n}.$$

Therefore, $b > 2n$.

$$\begin{cases} w_1 \equiv 2^{b-a} \ (mod 2^n - 1), \\ w_2 \equiv -2^{b-a+2} (mod 2^{n+1} - 1), \\ w_3 \equiv 0 (mod 2^{n+a}). \end{cases}$$

then $w_1 = q_1(2^n - 1) + 2^{|b-a|_n}, w_2 = q_2(2^{n+1} - 1) - 2^{|b-a+2|_{n+1}}$ and $w_3 = q_3 2^{n+a}$.

Note that for $b = 2n + a, w_1 = 1, w_2 = -1$ and $w_3 = 0$, the conditions of Lemmas 1, 2 and 3 are satisfied. Suppose there exist w_1, w_2 and w_3 satisfying the conditions of Lemmas 1, 2, 3 with $2n + 1 \leq b < 2n + a$. Then b can be expressed as:$b = n + a + |b - a|_n$ and $|b - a + 2|_{n+1} = |b - a|_n, w_2 = q_2(2^{n+1} - 1) - 2^{|b-a|_n}$.

From (13), it follows that:

$$0 \leq (2^{2n+1} + (2^a - 3)2^n + 1)w_1 < 2^{2n+a}.$$

Therefore, $q_1 = 0$ and $w_1 = 2^{|b-a|_n}$. Hence

$$\begin{aligned} 0 \leq {} & (2^n - 1)(2^{n+a} - 2^{n+1} + 1)(2w_1 + w_2) \\ & + (2^{2n+1} + (2^a - 3)2^n + 1)w_1 \leq C_P \end{aligned}$$

Since $2w_1 + w_2 = q_2(2^{n+1} - 1) + 2^{|b-a+2|_{n+1}}$ and $w_1 = 2^{|b-a|_n}$, then

$$\begin{aligned} & (2^n - 1)(2^{n+a} - 2^{n+1} + 1)(2w_1 + w_2) \\ & + (2^{2n+1} + (2^a - 3)2^n + 1)w_1 = q_2 P + 2^{|b-a|_n + 2n + a}, \\ & 0 \leq q_2 P + 2^{|b-a|_n + 2n + a} < 2^{2n+a}, \end{aligned}$$

and $-1 < q_2 < 0$ has no integer solutions.

The theorem is proven.

Consider the minimal core function satisfying Theorem 2 and show that it can be used for efficient implementation of the sign function in RNS. The core function takes the form:

$$C(X) = \left\lfloor \frac{X}{2^n - 1} \right\rfloor - \left\lfloor \frac{X}{2^{n+1} - 1} \right\rfloor.$$

With the following parameters:

$$P = (2^n - 1)(2^{n+1} - 1)2^{n+a},$$

$$C(P) = 2^{2n+a},$$

$$\frac{P}{2} = (2^n - 1)(2^{n+1} - 1)2^{n+a-1}.$$

The statement is proven**.**

Statement 1. $\forall X \in [0, P)$: $C(X) \geq 0$.

Proof: Since for all n: $2^n - 1 < 2^{n+1} - 1$, it follows that

$$\frac{X}{2^n - 1} > \frac{X}{2^{n+1} - 1},$$

and consequently

$$\left\lfloor \frac{X}{2^n - 1} \right\rfloor \geq \left\lfloor \frac{X}{2^{n+1} - 1} \right\rfloor,$$

which means that for all $X \in [0, P)$: $C(X) \geq 0$.

The statement has been proven.

Statement 2. $\forall X \in [0, P)$: $C(X) \leq 2^{2n+a}$.

Proof: Consider the function $G(P - X) = 2^{2n+a} - C(X)$, where we have:

$$G(P - X) = \left\lfloor \frac{P - X}{2^n - 1} \right\rfloor - \left\lfloor \frac{P - X}{2^{n+1} - 1} \right\rfloor.$$

Let $P - X = Y$, then $Y \in (0, P]$ and the function $G(Y)$ takes the form:

$$G(Y) = \left\lfloor \frac{Y}{2^n - 1} \right\rfloor - \left\lfloor \frac{Y}{2^{n+1} - 1} \right\rfloor.$$

Since for all n: $2^n - 1 < 2^{n+1} - 1$, it follows that

$$\frac{Y}{2^n - 1} > \frac{Y}{2^{n+1} - 1},$$

and consequently

$$\left\lfloor \frac{Y}{2^n - 1} \right\rfloor \geq \left\lfloor \frac{Y}{2^{n+1} - 1} \right\rfloor,$$

which means that for all $Y \in (0, P]$: $C(X) \leq 2^{2n+a}$.

The statement is proven.

Theorem 3. $\forall X \in [0, P)$ the following conditions hold:

If $C(X) > 2^{2n+a-1}$, then $X > (2^n - 1)(2^{n+1} - 1)2^{n+a-1}$.

If $C(X) < 2^{2n+a-1}$, then $X < (2^n - 1)(2^{n+1} - 1)2^{n+a-1}$.

Proof: *Case 1*. If $X \geq \frac{P}{2} + 2^n - 1$, then $C(X) > 2^{2n+a-1}$). Consider numbers of the form: $Y = (2^n - 1)(2^{n+1} - 1)2^{n+a-1} + 2^n - 1 + X, where 0 \leq X < \frac{P}{2} - 2^n + 1$, we obtain:

$$C(Y) = \left\lfloor \frac{(2^n - 1)(2^{n+1} - 1)2^{n+a-1} + 2^n - 1 + X}{2^n - 1} \right\rfloor$$

$$-\left\lfloor\frac{(2^n-1)(2^{n+1}-1)2^{n+a-1}+2^n-1+X}{2^{n+1}-1}\right\rfloor$$

$$=2^{2n+a-1}+1+\left\lfloor\frac{X}{2^n-1}\right\rfloor-\left\lfloor\frac{X}{2^{n+1}-1}\right\rfloor$$

$$+\left\lfloor\frac{X}{2^{n+1}-1}\right\rfloor-\left\lfloor\frac{X+2^n-1}{2^{n+1}-1}\right\rfloor$$

$$=2^{2n+a-1}+1+C(X)+\left\lfloor\frac{X}{2^{n+1}-1}\right\rfloor-\left\lfloor\frac{X+2^n-1}{2^{n+1}-1}\right\rfloor.$$

If $\left\lfloor\frac{X}{2^{n+1}-1}\right\rfloor=\left\lfloor\frac{X+2^n-1}{2^{n+1}-1}\right\rfloor$, then: $C(Y)=2^{2n+a-1}+1+C(X)$. From Statement 2, $C(X)\geq 0$, hence: $C(Y)>2^{2n+a-1}$.

If $\left\lfloor\frac{X}{2^{n+1}-1}\right\rfloor=\left\lfloor\frac{X+2^n-1}{2^{n+1}-1}\right\rfloor-1$, then: $2^n\leq|X|_{2^{n+1}-1}<2^{n+1}-1$. We show that if $2^n\leq|X|_{2^{n+1}-1}<2^{n+1}-1$ then $C(X)\geq 1$. Since $2^n\leq|X|_{2^{n+1}-1}<2^{n+1}-1$, then $X=\xi(2^{n+1}-1)+\delta$, where $\xi\leq 0$ and $2^n\leq\delta<2^{n+1}-1$. Computing $C(X)$, we have:

$$C(X)=\left\lfloor\frac{\xi(2^{n+1}-1)+\delta}{2^n-1}\right\rfloor-\left\lfloor\frac{\xi(2^{n+1}-1)+\delta}{2^{n+1}-1}\right\rfloor.$$

Since $\left\lfloor\frac{\xi(2^{n+1}-1)+\delta}{2^n-1}\right\rfloor=2\xi+\left\lfloor\frac{\xi+\delta}{2^n-1}\right\rfloor$ and $\left\lfloor\frac{\xi(2^{n+1}-1)+\delta}{2^{n+1}-1}\right\rfloor=\xi$, then:

$$C(X)=\xi+\left\lfloor\frac{\xi+\delta}{2^n-1}\right\rfloor.$$

Given that $\xi+\delta\geq 2^n$, then $\left\lfloor\frac{\xi+\delta}{2^n-1}\right\rfloor\geq 1$, consequently, $C(X)\geq 1$, hence $C(Y)>2^{2n+a-1}$.

Case 2. If $X\leq\frac{P}{2}-2^n+1$ then $C(X)<2^{2n+a-1}$. Consider numbers of the form: $G=(2^n-1)(2^{n+1}-1)2^{n+a-1}-2^n+1-X$, where $0\leq X\leq\frac{P}{2}-2^n+1$, we obtain:

$$C(G)=\left\lfloor\frac{(2^n-1)(2^{n+1}-1)2^{n+a-1}-2^n+1-X}{2^n-1}\right\rfloor$$

$$-\left\lfloor\frac{(2^n-1)(2^{n+1}-1)2^{n+a-1}-2^n+1-X}{2^{n+1}-1}\right\rfloor$$

$$=2^{2n+a-1}-1-\left\lfloor\frac{X}{2^n-1}\right\rfloor+\left\lfloor\frac{X+2^n-1}{2^{n+1}-1}\right\rfloor.$$

Since $\forall X\geq 2^n-1$ the inequality holds:

$$\frac{X}{2^n-1}>\frac{X+2^n-1}{2^{n+1}-1},$$

then:

$$\left\lceil\frac{X}{2^n-1}\right\rceil\geq\left\lceil\frac{X+2^n-1}{2^{n+1}-1}\right\rceil,$$

consequently, $C(G)<2^{2n+a-1}$.

If $0\leq X<2^n-1$, then $\left\lceil\frac{X}{2^n-1}\right\rceil=0$ and: $\left\lceil\frac{X+2^n-1}{2^{n+1}-1}\right\rceil=0$, consequently, $C(G)=2^{2n+a-1}-1<2^{2n+a-1}$.

Case 3. If $\frac{P}{2}-2^n+1<X<\frac{P}{2}+2^n-1$ then $C(X)=2^{2n+a-1}$. Consider numbers of the form: $U=(2^n-1)(2^{n+1}-1)2^{n+a-1}+X$, we obtain:

$$C(U)=\left\lfloor\frac{(2^n-1)(2^{n+1}-1)2^{n+a-1}+X}{2^n-1}\right\rfloor$$

$$-\left\lfloor\frac{(2^n-1)(2^{n+1}-1)2^{n+a-1}+X}{2^{n+1}-1}\right\rfloor$$

$$=2^{2n+a-1}+\left\lfloor\frac{X}{2^n-1}\right\rfloor-\left\lfloor\frac{X}{2^{n+1}-1}\right\rfloor.$$

If $0\leq X<2^n-1$, then $\left\lfloor\frac{X}{2^n-1}\right\rfloor=0$ and $\left\lfloor\frac{X}{2^{n+1}-1}\right\rfloor=0$, consequently, $C(U)=2^{2n+a-1}$.

If $-2^n+1<X<0$, then $\left\lfloor\frac{X}{2^n-1}\right\rfloor=-1$ and $\left\lfloor\frac{X}{2^{n+1}-1}\right\rfloor=-1$, consequently, $C(U)=2^{2n+a-1}$.

The statement is proven.

The residues of a number with zero core function value must satisfy the condition $x_1\geq x_2$.

Let us compute the constants for RNS with the moduli set $\{2^n-1,2^{n+1}-1,2^{n+a}\}$.

$$C_P=C(P)=(2^{n+1}-1)2^{n+a}-(2^n-1)2^{n+a}=2^{2n+a},$$

$$P_1=(2^{n+1}-1)2^{n+a},\quad P_2=(2^n-1)2^{n+a},$$

$$P_3=(2^n-1)(2^{n+1}-1),$$

$$|P_1^{-1}|_{p_1}=\left|\frac{1}{(2^{n+1}-1)2^{n+a}}\right|_{2^n-1}=\left|\frac{1}{2^a}\right|_{2^n-1}=|2^{n-a}|_{2^n-1}$$

$$\begin{cases}1, & if\ a=0\\ 2^{n-a}, & if\ 1\leq a\leq n\end{cases}$$

$$|P_2^{-1}|_{p_2}=\left|\frac{1}{(2^n-1)2^{n+a}}\right|_{2^{n+1}-1}=\left|\frac{-1}{2^{n+a-1}}\right|_{2^{n+1}-1}=\left|\frac{-2^{2n+2}}{2^{n+a-1}}\right|_{2^{n+1}-1}$$

$$=|2^{n+1}-2^{n-a+3}-1|_{2^{n+1}-1}$$

$$\begin{cases}2^{n+1}-2^2-1, & if\ a=0\\ 2^{n+1}-2-1, & if\ a=1\\ 2^{n+1}-2, & if\ a=2\\ 2^{n+1}-2^{n-a+3}-1, & if\ 3\leq a\leq n.\end{cases}$$

$$|P_3^{-1}|_{p_3}=\left|\frac{1}{(2^n-1)(2^{n+1}-1)}\right|_{2^{n+a}}=|2^{n+1}+2^n+1|_{2^{n+a}}$$

$$\begin{cases}1, & if\ a=0\\ 2^n+1, & if\ a=0\\ 2^{n+1}-2^n+!, & if\ 2\leq a\leq \text{n.}\end{cases}$$

For $a>2$: $|P_1^{-1}|_{p_1}=2^{n-a}$, $|P_2^{-1}|_{p_2}=2^{n+1}-2^{n-a+3}-1$, $|P_3^{-1}|_{p_3}=2^{n+1}+2^n+1$, therefore:

$$k_1=C\left(|P_1^{-1}|_{p_1}\cdot P_1\right)$$

$$=\left\lfloor\frac{(2^{n+1}-1)2^{n+a}2^{n-a}}{2^n-1}\right\rfloor-\left\lfloor\frac{(2^{n+1}-1)2^{n+a}2^{n-a}}{2^{n+1}-1}\right\rfloor$$

$$=\left\lfloor\frac{(2^{n+1}-1)2^{2n}}{2^n-1}\right\rfloor-2^{2n}=2^{2n+1}+2^n+1-2^{2n}$$

$$=(2^n-1)(2^{2n+1}+2^n+1)+1,$$

$$k_2=C\left(|P_2^{-1}|_{p_2}\cdot P_2\right)$$

$$=\left\lfloor\frac{(2^{n+1}-2^{n-a+3}-1)(2^n-1)2^{n+a}}{2^n-1}\right\rfloor$$

$$-\left\lfloor\frac{(2^{n+1}-2^{n-a+3}-1)(2^n-1)2^{n+a}}{2^{n+1}-1}\right\rfloor$$

$$= (2^{n+1} - 2^{n-a+3} - 1)2^{n+a} - (2^n - 1)2^{n+a} - (2^{n+1} + 1 - 2^{2n+2})$$
$$= 2^{2n+a+1} - 2^{2n+3} - 2^{n+a} - 2^{2n+a} + 2^{n+a} - 2^{n+1} - 1 + 2^{2n+2}$$
$$= 2^{2n+a} - 2^{2n+2} - 2^{n+1} - 1,$$
$$k_3 = C\left(|P_3^{-1}|_{p_3} \cdot P_3\right)$$
$$= \left\lfloor \frac{(2^n - 1)(2^{n+1} - 1)(2^{n+1} + 2^n + 1)}{2^n - 1} \right\rfloor$$
$$- \left\lfloor \frac{(2^n - 1)(2^{n+1} - 1)(2^{n+1} + 2^n + 1)}{2^{n+1} - 1} \right\rfloor$$
$$= (2^{n+1} - 1)(2^{n+1} + 2^n + 1) - (2^n - 1)(2^{n+1} + 2^n + 1)$$
$$= 2^{2n+1} + 2^{2n} + 2^n.$$

For $a = 2$: $|P_1^{-1}|_{p_1} = 2^{n-2}$, $|P_2^{-1}|_{p_2} = 2^{n+1} - 2$, $|P_3^{-1}|_{p_3} = 2^{n+1} + 2^n + 1$, therefore:

$$k_1 = C\left(|P_1^{-1}|_{p_1} \cdot P_1\right) = \left\lfloor \frac{(2^{n+1} - 1)2^{2n}}{2^n - 1} \right\rfloor - \left\lfloor \frac{(2^{n+1} - 1)2^{2n}}{2^{n+1} - 1} \right\rfloor$$
$$= 2^{2n} + 2^n + 1,$$
$$k_2 = C\left(|P_2^{-1}|_{p_2} \cdot P_2\right) = \left\lfloor \frac{(2^{n+1} - 2)(2^n - 1)2^{n+2}}{2^n - 1} \right\rfloor$$
$$- \left\lfloor \frac{(2^{n+1} - 2)(2^n - 1)2^{n+2}}{2^{n+1} - 1} \right\rfloor$$
$$= (2^{n+1} - 2)2^{n+2} - (2^n - 1)2^{n+2} + 2^{n+1} - 1$$
$$= 2^{2n+2} - 2^{n+1} - 1,$$
$$k_3 = C\left(|P_3^{-1}|_{p_3} \cdot P_3\right) = \left\lfloor \frac{(2^n - 1)(2^{n+1} - 1)(2^{n+1} + 2^n + 1)}{2^n - 1} \right\rfloor$$
$$- \left\lfloor \frac{(2^n - 1)(2^{n+1} - 1)(2^{n+1} + 2^n + 1)}{2^{n+1} - 1} \right\rfloor$$
$$= (2^{n+1} - 1)(2^{n+1} + 2^n + 1) - (2^n - 1)(2^{n+1} + 2^n + 1)$$
$$= 2^{2n+1} + 2^{2n} + 2^n.$$

For $a = 1$: $|P_1^{-1}|_{p_1} = 2^{n-1}$, $|P_2^{-1}|_{p_2} = 2^{n+1} - 2 - 1$, $|P_3^{-1}|_{p_3} = 2^n + 1$, therefore:

$$k_1 = C\left(|P_1^{-1}|_{p_1} \cdot P_1\right) = \left\lfloor \frac{(2^{n+1} - 1)2^{2n}}{2^n - 1} \right\rfloor - \left\lfloor \frac{(2^{n+1} - 1)2^{2n}}{2^{n+1} - 1} \right\rfloor$$
$$= 2^{2n} + 2^n + 1,$$
$$k_2 = C\left(|P_2^{-1}|_{p_2} \cdot P_2\right) = \left\lfloor \frac{(2^{n+1} - 2 - 1)(2^n - 1)2^{n+1}}{2^n - 1} \right\rfloor$$
$$- \left\lfloor \frac{(2^{n+1} - 2 - 1)(2^n - 1)2^{n+1}}{2^{n+1} - 1} \right\rfloor$$
$$= (2^{n+1} - 2 - 1)2^{n+1} - (2^n - 1)2^{n+1} + 2^{n+1} - 1$$
$$= 2^{2n+1} - 2^{n+1} - 1,$$
$$k_3 = C\left(|P_3^{-1}|_{p_3} \cdot P_3\right) = \left\lfloor \frac{(2^n - 1)(2^{n+1} - 1)(2^n + 1)}{2^n - 1} \right\rfloor$$

$$- \left\lfloor \frac{(2^n - 1)(2^{n+1} - 1)(2^n + 1)}{2^{n+1} - 1} \right\rfloor$$
$$= (2^{n+1} - 1)(2^n + 1) - (2^n - 1)(2^n + 1) = (2^n + 1)2^n = 2^{2n} + 2^n.$$

For $a = 0$: $|P_1^{-1}|_{p_1} = 1, |P_2^{-1}|_{p_2} = 2^{n+1} - 2^2 - 1, |P_3^{-1}|_{p_3} = 1$, therefore:

$$k_1 = C\left(|P_1^{-1}|_{p_1} \cdot P_1\right) = \left\lfloor \frac{(2^{n+1} - 1)2^n}{2^n - 1} \right\rfloor - \left\lfloor \frac{(2^{n+1} - 1)2^n}{2^{n+1} - 1} \right\rfloor$$
$$= (2^{n+1} - 2 + 1)2^n = (2^n - 1)(2^{n+1} + 1) + 1,$$
$$k_2 = C\left(|P_2^{-1}|_{p_2} \cdot P_2\right) = \left\lfloor \frac{(2^{n+1} - 2^2 - 1)(2^n - 1)2^n}{2^n - 1} \right\rfloor$$
$$- \left\lfloor \frac{(2^{n+1} - 2^2 - 1)(2^n - 1)2^n}{2^{n+1} - 1} \right\rfloor$$
$$= (2^{n+1} - 2^2 - 1)2^n - (2^n - 1)2^n - \left\lfloor -\frac{(2^{n+1} - 2)2^{n+1}}{2^{n+1} - 1} \right\rfloor$$
$$= (2^{n+1} - 2^2 - 1)2^n - (2^n - 1)2^n + 2^{n+1} - 1$$
$$= 2^{2n} - 2^{n+1} - 1,$$
$$k_3 = C\left(|P_3^{-1}|_{p_3} \cdot P_3\right) = \left\lfloor \frac{(2^n - 1)(2^{n+1} - 1)}{2^n - 1} \right\rfloor$$
$$- \left\lfloor \frac{(2^n - 1)(2^{n+1} - 1)}{2^{n+1} - 1} \right\rfloor$$
$$= 2^{n+1} - 1 - 2^n + 1 = 2^n.$$

For determining the sign of a number using the minimal core function for the moduli set $\{2^n - 1, 2^{n+1} - 1, 2^{n+a}\}$, Algorithm 4 is proposed.

Algorithm 4: Sign detection using core function for moduli set $\{2^n - 1, 2^{n+1} - 1, 2^{n+a}\}$

Input: $(x_1, x_2, x_3), \{2^n - 1, 2^{n+1} - 1, 2^{n+a}\}, P = (2^n - 1)2^{n+a}(2^n + 1), N = 2n + a, C_P = 2^N, C(P/2) = 2^{2n+a-1}, \{k_1, k_2, k_3\}$

Output: $sign$ # 0 for positive, 1 for negative

1. $C(X) = (k_1x_1 + k_2x_2 + k_3x_3) \wedge (2^N - 1)$
2. **if** $C(X) = 0$ **and** $x_1 < x_2$
2.1. $C(X) = C_P$
3. **else**
3.1. $C(X) = 0$
4. **if** $C(X) < C(P/2)$
4.1. $sign = 0$
5. **else**
5.1. $sign = 1$
6. **return** $sign$

Consider Example 4 demonstrating sign detection using the proposed core function.

Example 4. For $n = 3$ and $a = 0$, we have moduli $p_1 = 7, p_2 = 15, p_3 = 8$. The dynamic range is $P = 7 \cdot 15 \cdot 8 = 840$. The core function of the dynamic range is

$$C_P = 2^{2n+a} = 64 = 2^6.$$

The core function of half-range is

$$C\left(\frac{P}{2}\right) = 2^{2n+a-1} = 32.$$

Next, we find the coefficients k_i:

$$k_1 = 9, k_2 = 47, k_3 = 8.$$

Let's determine the sign of number $(5, 8, 3)$. Compute $C(X)$:

$$C(X) = |9 \cdot 5 + 47 \cdot 8 + 8 \cdot 3|_{2^6} = |61|_{2^6} = 61.$$

Since $61 > 32$, the number $(5, 8, 3)$ is negative.

Thus, the proposed approach allows reducing operand sizes, and the computationally expensive modulo operation can be replaced by either taking the N highest bits or a bitwise AND operation.

6. Performance Analysis

To evaluate the performance of the proposed sign detection algorithm based on the ACF, we conducted comprehensive computational experiments comparing it with traditional methods: CRT, its approximate version, and the DF approach discussed in Section 3. The experiments were carried out for different bit depths in the dynamic range from 16 to 64 bits.

The evaluation was conducted on a modern computing platform with the following specifications:

- Processor: Intel Core i7-7700HQ @ 2.80 GHz
- Memory: 8 GB DDR4
- Operating System: Windows 10
- Implementation: All algorithms were implemented in C++

We used the moduli set $\{2^n - 1, 2^{n+1} - 1, 2^{n+a}\}$ with varying parameters n and a to cover different bit widths. For each method and bit width, we measured the average execution time over 10^6 random test cases to ensure statistical significance. The simulation results are presented in Tables 2, 3. The best timing results for each dimension are shown in Table 4.

The core function we proposed demonstrates the best execution time in all tests, especially for large bit widths (~22.7% faster than CRT at 64 bits). DF is also efficient but falls short of ACF. CRT performs stably but is slower, while ACRT is not always justified, as it loses to CRT at small bit widths. Thus, the proposed algorithm is on average 25.6% faster than classical methods of sign detection in RNS.

7. Conclusion

This paper presented a high-speed algorithm for sign detection in the RNS based on the ACF. The proposed method leverages the properties of a specially selected moduli set $\{2^n - 1, 2^{n+1} - 1, 2^{n+a}\}$ to efficiently compute the sign of a number.

The key advantages of the proposed approach include reduced operand sizes, which simplify arithmetic operations, and an efficient modulo replacement where costly division is substituted by bitwise operations or truncation of the highest bits, significantly accelerating computations. The experimental results demonstrate that the ACF-based algorithm outperforms traditional methods (CRT, ACRT, and DF) across all tested bit widths, achieving an average 25.6% speedup.

The comparative analysis (Table 4) confirms that ACF consistently delivers the best timing results, while DF remains a viable alternative. Although CRT is stable, its performance is inferior, and ACRT proves inefficient for smaller bit widths.

Future research directions may include extending the method to larger moduli sets and non-uniform RNS bases, optimizing hardware implementations for real-time applications.

Table 2. Simulation results of sign detection in RNS using CRT, ACRT and DF.

RNS moduli set	Time, μs		
	CRT	ACRT	DF
16 bits			
$\{19, 29, 128\}$	289.7	326.9	268.1
$\{29, 67, 64\}$	298.4	336.7	275.0
$\{7, 11, 19, 64\}$	309.5	330.5	290.7
24 bits			
$\{131, 257, 1024\}$	331.30	366.6	320.2
$\{29, 63, 67, 256\}$	316.1	360.2	310.5
$\{5, 7, 11, 19, 29, 128\}$	333.5	378.5	330.3
32 bits			
$\{1031, 2039, 4096\}$	363.6	401.4	360.5
$\{131, 257, 511, 512\}$	375.9	396.1	350.1
$\{29, 63, 65, 131, 512\}$	381.7	388.9	345.4
40 bits			
$\{8171, 16383, 16411\}$	411.6	412.6	390.2
$\{511, 1023, 2047, 2048\}$	393.4	407.9	380.8
$\{127, 255, 263, 509, 512\}$	390.5	401.6	375.6
$\{19, 31, 65, 129, 509, 512\}$	417.33	412.3	395.7
48 bits			
$\{65535, 65537, 131072\}$	434.4	420.6	420.0
$\{2047, 4095, 8191, 8192\}$	446.7	422.5	430.8
$\{263, 511, 1023, 1025, 2048\}$	423.4	417.3	425.4
$\{61, 127, 129, 263, 509, 2048\}$	415.1	425.9	415.2
$\{17, 31, 61, 127, 129, 509, 1024\}$	440.5	428.2	435.2
56 bits			
$\{8191, 16383, 32767, 32768\}$	459.9	443.4	440.1
$\{127, 257, 511, 513, 2047, 8192\}$	460.9	438.9	450.9
$\{17, 31, 65, 127, 129, 257, 511, 1024\}$	439.4	441.9	445.4
64 bits			
$\{32767, 65535, 131071, 131072\}$	464.2	453.5	455.8
$\{257, 511, 1025, 2049, 8191, 8192\}$	463.2	450.6	452.6
$\{65, 127, 257, 511, 1023, 2047, 8192\}$	465.2	455.8	457.3

Table 3. Simulation results of sign detection in RNS using ACF for moduli sets $\{2^n - 1, 2^{n+1} - 1, 2^{n+a}\}$.

RNS moduli set	Time, μs
16 bits	
$\{31, 63, 32\}$	227.9
24 bits	
$\{127, 255, 512\}$	251.5
32 bits	
$\{1023, 2047, 2048\}$	285.5
40 bits	
$\{8191, 16383, 8192\}$	321.1
48 bits	
$\{32767, 65535, 131072\}$	340.3
56 bits	
$\{262143, 524287, 524288\}$	351.6
64 bits	
$\{2097151, 4194303, 2097152\}$	371.2

Table 4. Execution time of sign detection in RNS.

Time, μs							
Bit width	16	24	32	40	48	56	64
CRT	289.7	316.1	363.6	390.5	415.1	439.4	463.2
ACRT	326.9	360.2	388.9	401.6	417.3	438.9	450.6
DF	268.1	310.5	345.4	375.6	415.2	440.1	452.6
ACF	227.9	251.5	285.5	321.1	340.3	351.6	371.2
Reduction, %							
$\frac{CRT-ACF}{ACF}$	27.1	25.7	27.4	21.6	22.0	25.0	24.8
$\frac{ACRT-ACF}{ACF}$	43.4	43.2	36.2	25.1	22.6	24.8	21.4
$\frac{DF-ACF}{ACF}$	17.6	23.5	21.0	17.0	22.0	25.2	21.9

In summary, the proposed ACF-based sign detection algorithm offers a significant performance improvement over classical approaches, making it a promising solution for high-speed RNS arithmetic in digital signal processing, cryptography, and parallel computing.

References

[1]. Schoinianakis, D. Residue arithmetic systems in cryptography: a survey on modern security applications. Journal of Cryptographic Engineering 2020, 10, 249-267.

[2]. Cheon, J.H.; Han, K.; Kim, A.; Kim, M.; Song, Y. A full RNS variant of approximate homomorphic encryption. In Proceedings of the International Conference on Selected Areas in Cryptography. Springer, 2018, pp. 347-368.

[3]. Bajard, J.C.; Eynard, J. RNS Approach in Lattice-Based Cryptography. Embedded Systems Design with Special Arithmetic and Number Systems 2017, pp. 345-368.

[4]. Selvam, R.; Tyagi, A. Power side channel resistance of RNS secure logic. In Proceedings of the 2018 31st International Conference on VLSI Design and 2018 17th International Conference on Embedded Systems (VLSID). IEEE, 2018, pp. 143-148.

[5]. Kawamura, S.; Komano, Y.; Shimizu, H.; Osuka, S.; Fujimoto, D.; Hayashi, Y.; Imafuku, K. Efficient algorithms for sign detection in RNS using approximate reciprocals. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences 2021, 104, 121-134.

[6]. Lutsenko, V.; Zgonnikov, M. Investigation of Neural Network Methods for Error Detection and Correction in the Residue Number System. In Proceedings of the International Workshop on Advanced Information Security Management and Applications. Springer, 2024, pp. 194-206.

[7]. Lutsenko, V.V.; Babenko, M.G.; Khamidov, M.M. High speed method of conversion numbers from residue number system to positional notation. Proceedings of the Institute for System Programming of the RAS 2024, 36, 117-132.

[8]. Bajard, J.C.; Didier, L.S.; Kornerup, P. An RNS Montgomery modular multiplication algorithm. IEEE Transactions on Computers 339 1998, 47, 766-776.

[9]. Nozaki, H.; Motoyama, M.; Shimbo, A.; Kawamura, S. Implementation of RSA algorithm based on RNS Montgomery multiplication. In Proceedings of the InternationalWorkshop on Cryptographic Hardware and Embedded Systems. Springer, 2001, pp. 364-376.

[10]. Cardarilli, G.; Re, M.; Lojacono, R. RNS-to-binary conversion for efficient VLSI implementation. IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications 2002, 45, 667-669.

[11]. Babenko, M.; Deryabin, M.; Piestrak, S.J.; Patronik, P.; Chervyakov, N.; Tchernykh, A.; Avetisyan, A. RNS number comparator based on a modified diagonal function. Electronics 2020, 9, 1784.

[12]. Akushsky, I.Y.; Burtsev, V.M.; Pak, I.T. Calculation of the positional characteristic (core) of the non-positional code; Nauka, 1977; pp. 17-25.

Информация об авторах / Information about authors

Владислав Вячеславович ЛУЦЕНКО – ассистент кафедры вычислительной математики и кибернетики факультета математики и компьютерных наук имени профессора Н.И. Червякова ФГАОУ ВПО «Северо-Кавказский федеральный университет». Сфера научных интересов: высокопроизводительные вычисления, система остаточных классов, умный город, нейронные сети, интернет вещей.

Vladislav Vyacheslavovich LUTSENKO – assistant professor, Department of computational mathematics and cybernetics, Faculty of mathematics and computer science named after Professor N.I. Chervyakov, North Caucasus Federal University. Research interests: high-performance computing, residue number system, smart city, neural networks, Internet of Things.

Айсанат Эдуардновна ГЕРЮГОВА – студент кафедры математического анализа, алгебры и геометрии факультета математики и компьютерных наук имени профессора Н.И. Червякова ФГАОУ ВПО «Северо-Кавказский федеральный университет». Сфера научных интересов: система остаточных классов, математика и компьютерные науки.

Aisanat Eduardnovna GERYUGOVA – student in the Department of mathematical analysis, algebra and geometry, Faculty of mathematics and computer science named after Professor N.I. Chervyakov, North Caucasus Federal University. Her research interests include residue number systems, mathematics, and computer science.

Михаил Григорьевич БАБЕНКО – доктор физико-математических наук, заведующий кафедры вычислительной математики и кибернетики факультета математики и компьютерных наук имени профессора Н.И. Червякова ФГАОУ ВПО «Северо-Кавказский федеральный университет». Сфера научных интересов: облачные вычисления, высокопроизводительные вычисления, система остаточных классов, нейронные сети, криптография.

Mikhail Grigoryevich BABENKO – Dr. Sci. (Phys.-Math.), Head of the Department of computational mathematics and cybernetics, Faculty of mathematics and computer science named after Professor N.I. Chervyakov, North Caucasus Federal University. His research interests include cloud computing, high-performance computing, residue number systems, neural networks, cryptography.