1. Maciej CZYŻAK[1], 2. Robert SMYK[2]

ANS w Elblagu (1), Politechnika Gdańska (2)
ORCID: 2. 0000-0001-9365-4633

doi:10.15199/48.2025.03.35

New residue-to-binary converter for moduli set $\{2^n - 1, 2^n, 2^n + 1\}$ based on core function

Abstract. *The paper presents a new algorithm for conversion from the residue number system with the base* $\{2^n - 1, 2^n, 2^n + 1\}$ *to the binary system. The converter is based on core function. The use of core function allows for simpler converter formula and thus leads to more effective converter architecture. The conversion algorithm is presented. The architecture of a new converter is shown and its time-hardware complexity is analysed.*

Streszczenie. *W artykule przedstawiono nowy algorytm konwersji z systemu resztowego z bazą* $\{2^n - 1, 2^n, 2^n + 1\}$ *do systemu binarnego. Konwerter opiera się na wykorzystaniu funkcji jądra. Zastosowanie funkcji jądra pozwala na uproszczoną formułę konwertera, a tym samym prowadzi do bardziej efektywnej architektury konwertera. Przedstawiono nowy algorytm konwersji oraz pokazano architekturę nowego konwertera i przeanalizowano jego złożoność czasowo-sprzętową. (**Nowy konwerter resztkowo-binarny dla zestawu modułów* $\{2^n - 1, 2^n, 2^n + 1\}$ ***na podstawie funkcji rdzeniowej**)*

Keywords: core function, residue number system, RNS, residue-to-binary conversion
Słowa kluczowe: funkcja jądra, resztowy system liczbowy, RNS, konwersja z RNS do systemu binarnego

Introduction

Digital signal processing (DSP) is a subarea of electrical engineering as well as the main tool used in this area. The DSP has been applied to improve the design of electric drives [1-5]. For example, FFT can be used to analyze short-circuit currents of transformers [6].

The majority of digital architectures that process signals in real-time require the pipelined high-speed hardware realization. Such solutions call for fine granulation of the architecture due to the delay of the slowest block that determines the minimum pipelining rate. One of the effective tools for these applications where multiply-add operations dominate is the residue number system (RNS) [7]. In this system high speed, low-level pipelined realization of addition subtraction and multiplication is attainable. The RNS replaces operations in the large integer ring by a set of operations in smaller integer rings. After converting an integer number to RNS addition, subtraction and multiplication can be performed independently on the corresponding digits of the arguments without carries between the positions of the number. However, to process numbers in RNS they must be first converted to the RNS representation [8]. This operation is relatively simple. An exemplary solution is given in [8]. RNS is determined by its base $B = \{m_1, m_2, \dots, m_n\}$, where m_i, $i = 1, .. n$ are termed moduli. Two types of bases are usually considered. An useful type of the base is B_1 where the moduli m_i are small 5-, 6-bit numbers. The other type are bases where the moduli are akin to the power of two. The example can be $B_2 = \{2^n, 2^n - 1, 2^n + 1\}$. B_1 type base allows for greater parallelisation and replacing multipliers by constant using small ROMs or even logic functions that permits to avoid the delay introduced by multipliers or greater ROMs. However, each RNS-based architecture must contain a residue-to-binary converter which is a pure overhead. Much work has been done to simplify converter structures and made them hardware efficient and pipelined. In general, the converter formula for the B_1 type bases can be derived using Chinese Remainder Theorem (CRT) [9,10,13], the MRC [11,12] or both [16] as well as New-CRT II [14] or core function [15]. In the case of B_2 base the converter can be implemented without ROMs. Much work as pertaining to $\{2^n, 2^n - 1, 2^n + 1\}$ has been done [17-27].

Recently of special interest is in $\{2^n, 2^n - 1, 2^n + 1\}$ [25,27], due to the fact, that 2^n residue channel is simpler than two other channels and therefore the dynamic range can be increased by increasing k.

In the present work a novel approach to reverse converter design is proposed based on the use of core function. The use of core function simplifies the converter formula. The direct application of the core function is not effective due to the necessity to perform division. However, it can be avoided by the appropriate use of weights of core function as in [15]. In this paper the derivation of the reverse converter based on core function using the Chinese Remainder Theorem for Core Functions (CRTCF) [26] is presented. In the following sections an introduction to RNS is provided, the properties of modulo $2^n - 1$ are discussed, next properties of the core function are analyzed and finally, the derivation of the converter formula is given, along with the converter architecture and its evaluation and comparison.

RNS basics

RNS [7] is defined by its base $B = \{m_1, m_2, \dots, m_n\}$, i.e. being a set of usually mutually nonnegative numbers called moduli. The RNS number range M is expressed as $M = \prod_{i=1}^{n} m_i$. M denotes the number of integers that can be represented using the given base. If the moduli are mutually prime, i.e. $gcd\left(m_j, m_k\right) = 1$ for $\neq k, j, k = 1,2, \dots, n$ then any integer X from $[0, M - 1]$ can be represented by a vector of n integers $(x_1, x_2, \dots, x_n)$, where $x_i = |X|_{m_j}$ are residues that uniquely represent X in RNS. The arithmetic operations of addition, subtraction and multiplication in RNS are defined as

$$(x_1, x_2, \dots, x_n) \oplus (y_1, y_2, \dots, y_n) = (z_1, z_2, \dots, z_n) \tag{1}$$

where $z_i = |x_i \oplus y_i|_{m_i}$, $i = 1,2, \dots, n$ and $\oplus$ denotes an operation of addition, subtraction or multiplication performed in small rings $R(m_i)$, $i = 1,2, \dots, n$. The mutual primality condition of moduli ensures that the mapping between the ring modulo M and the sum of the rings is an isomorphism. Such a mapping can be implemented using the Chinese remainder theorem (CRT) or Mixed Radix Conversion (MRC). CRT has the following form

$$X = \left| \sum_{i=1}^{n} M_i \left| M_i^{-1} \right|_{m_i} x_i \right|_M \tag{2}$$

where $M_i = \frac{M}{m_i}$, $\left| M_i^{-1} \right|_{m_i}$ is the multiplicative inverse of M_i modulo m_i.

Properties of modulo $2^n - 1$ arithmetic

We consider here two operations that will be needed for realization of the converter: calculation of $|-Y|_{2^n-1}$ and reduction modulo $2^n - 1$. Let Y be represented in the binary form as $Y \leftrightarrow (y_{n-1}, \dots, y_0)$ where $y_i, i = 0,1, \dots, n-1$ are digits of its binary representation. Y after sign change can be obtained as

$$|-Y|_{2^n-1} = |2^n - 1 - Y|_{2^n-1} \tag{3}$$

The value on the right side of (3) is one's complement that corresponds to the negation of all bits of Y

$$2^n - 1 - Y \leftrightarrow (\bar{y}_{n-1}, \dots, \bar{y}_0) \tag{4}$$

The reduction modulo $2^n - 1$ of an *n*-bit number Y, where $m > n$ can be performed by segmentation of an *n*-bit number into $k = \left\lfloor \frac{m}{n} \right\rfloor + |m|_n$ *n*-bit blocks. Finally, the reduction can be performed as

$$|Y|_{2^n-1} = \sum_{j=1}^{k} \left|Y_j\right|_{2^n-1} \tag{5}$$

Properties of core function

For the base $B = \{m_1, m_2, \dots, m_n\}$ the core function is defined as

$$C(X) = \sum_{i=1}^{n} w_i \left\lfloor \frac{X}{m_i} \right\rfloor \tag{6}$$

For the given base B the weights w_i, $i = 1,2, \dots, n$ are constants and have to be properly selected to obtain the desired properties of the core function. $C(X)$ is a step function and if all weights are nonnegative, then it is a nondecreasing step function and it contains the direct positional characteristic of the number. However, in such a case its number range may be too large. In fact, the choice of weights w_i, should be such that the minimum C_{min} should be close to $C(0)$ and maximum core C_{max} to $C(M)$ on $[0, M)$, where

$$C(M) = \sum_{i=1}^{n} w_i \cdot M_i \tag{7}$$

In practice, the local selectivity and the number range must be balanced. The formula (6) can be expressed in the more convenient form

$$C(X) = \frac{C(M)}{M} \cdot X - \sum_{i=1}^{n} \frac{w_i}{m_i} \cdot |X|_{m_i} \tag{8}$$

It can be seen from (8) that $C(X)$ has the linear and nonlinear part. Equation (8) is derived form (6) and (7). Regarding that $M_i = M/m_j$, (8) cannot be used directly since the residues of X are available and not X itself. Moreover, computations of the core function by (8) would require division that is noneffective. The more suitable way is the use of Chinese Remainder Theorem for Core Function (CRTCF) in the following form

$$C(X) = \sum_{i=1}^{n} x_i \cdot C_i - r(X) \cdot C(M) \tag{9}$$

where $r(X)$ is the magnitude index coefficient and

$$C_i = \frac{C(M) \cdot \left|M_i^{-1}\right|_{m_i} - w_i}{m_i}, \; i = 1,2, \dots, n \tag{10}$$

The CRTCF can be also written as

$$C(X) = \left|\sum_{i=1}^{n} x_i \cdot C_i\right|_{C(M)} \tag{11}$$

The correct value of $C(X)$ cannot be obtained for these X for which $C(X) < 0$ or $C(X) \geq C(M)$. Such values of $C(X)$ are called critical cores. X using the CRTCF can be expressed as

$$X = \frac{M \cdot C(X) + \sum_{i=1}^{n} w_i \cdot M_i \cdot |X|_{m_i}}{C(M)} \tag{12}$$

New conversion algorithm

In this section we shall derive the conversion formula using (12) for $B = \{2^n - 1, 2^n, 2^n + 1\}$. We select $C(M)$ as

$$C(M) = (2^n - 1) \cdot (2^n + 1) = 2^{2n} - 1 \tag{13}$$

Next we shall compute the coefficients M_i

$$M_1 = 2^n(2^n + 1) \tag{14a}$$
$$M_2 = (2^n - 1)(2^n + 1) \tag{14b}$$
$$M_3 = (2^n - 1)2^n \tag{14c}$$

In the next step we should choose the weights w_i, $i = 1,2,3$ so that there will be no division in (12) when computing (10). The choice of $W = (0,1,0)$ will make it possible.

The multiplicative inverses required in (10) have the following values

$$|M_1^{-1}|_{m_1} = 2^{n-1} \tag{15a}$$
$$|M_2^{-1}|_{m_1} = 2^n - 1 \tag{15b}$$
$$|M_3^{-1}|_{m_1} = 2^{n-1} + 1 \tag{15c}$$

Using (15) we can compute coefficients C_i, $i = 1,2,3$ as

$$C_1 = \frac{(2^n-1)(2^n+1) \cdot 2^{n-1} - 0}{2^n - 1} = (2^n + 1)2^{n-1} \tag{16a}$$

$$C_2 = \frac{(2^n-1)(2^n+1)(2^n-1) - 1}{2^n} = 2^{2n} - 2^n + 1 \tag{16b}$$

$$C_3 = \frac{(2^n-1)(2^n+1)(2^{n-1}+1) - 0}{2^n + 1} = (2^n - 1)(2^{n-1} + 1) \tag{16c}$$

Using coefficients C_1, C_2, C_3 from (16) we receive

$$\begin{aligned} C(X) &= \left|\sum_{i=1}^{3} C_i\right|_{C(M)} \\ &= \left| \begin{matrix} x_1(2^n + 1)2^{n-1} + x_2(2^{2n} - 2^n - 1) \\ + x_3(2^{2n-1} + 2^n - 2^{n-1} - 1) \end{matrix} \right|_{2^{2n}-1} \end{aligned} \tag{17}$$

After reordering terms for x_i, $i = 1,2,3$ we have the first form of $C(X)$

$$\begin{aligned} C(X) = |&(x_1 + 2x_2 + x_3)2^{2n-1} \\ &+ (x_1 - 2x_2 + x_3)2^{n-1} \\ &+ (-x_2 - x_3)|_{2^{2n}-1} \end{aligned} \tag{18}$$

Moreover, using (12) and substituting $M = (2^n - 1)(2^n + 1) \cdot 2^n$, $M_i, i = 1,2,3$, $C(M) = (2^n - 1) \cdot (2^n + 1)$ and $w = \{0,1,0\}$, after reduction, we obtain

$$X = 2^n C(X) + |X|_{2^n} \tag{19}$$

It is evident, that first term is the right-shift of the $C(X)$ by *n*-bits and addition of $|X|_{2^n}$ is a simple concatenation.

Numerical example

In this section we shall present calculations of (15) for two bases. The first one is $B_3 = \{2^5 - 1, 2^5, 2^5 + 1\}$ and the second $B_4 = \{2^{16} - 1, 2^{16}, 2^{16} + 1\}$. The dynamic ranges are: $M = 32736$ for B_3 and $M = 281474976645120$.

For B_3 and $X = 10253$ we obtain

$$x_1 = |X|_{2^5-1} = 31$$

$$x_2 = |X|_{2^5} = 32$$
$$x_3 = |X|_{2^5+1} = 33$$

After inserting it in (15) we have

$$C(X) = |(31 + 2 \cdot 32 + 33)2^9 + (31 - 2 \cdot 32 + 33)2^4 + (-32 - 33)|_{2^{10}-1} = |36864 + 320 - 36|_{1023} = 320$$

Finally, X for B_3 is

$$X = 32 \cdot 320 + |10253|_{32} = 10253$$

For B_4 and $X = 67998$ we obtain

$$x_1 = |X|_{2^{16}-1} = 2463$$
$$x_2 = |X|_{2^{16}} = 2462$$
$$x_3 = |X|_{2^{16}+1} = 2461$$

After substituting it in (15) we get

$$C(X) = |(2463 + 2 \cdot 2462 + 2461)2^{31} + (2463 - 2 \cdot 2462 + 2461)2^{15} + (-2462 - 2461)|_{2^{32}-1} = |21148418965504 + 0 - 4923|_{4294967295} = 1$$

Finally, X for B_4 is equal to

$$X = 2^{16} \cdot 1 + |67998|_{2^{16}} = 65536 + 2462 = 67998$$

Converter design

In order to determine the formula for implementing the converter, we need to group the terms from (18) for x_1, x_2, x_3. Ordering by individual residues we get

$$C(X) = \left| \begin{matrix} x_1(2^{2n-1} + 2^{n-1}) + \\ x_2(2 \cdot 2^{2n-1} - 2 \cdot 2^{n-1} - 1) + \\ x_3(2^{2n-1} + 2^{n-1} - 1) \end{matrix} \right|_{2^{2n}-1} \quad (20)$$

The terms are sums of the shifted versions of operands. Simplifying the second term in (20) to $|x_2(2^{2n} - 2^n - 1)|_{2^{2n}-1}$ we can remark that $|2^{2n} - 1|_{2^{2n}-1} = 0$ and $-2 \cdot 2^{2n-1} = -2^{2n}$, thus we receive $|x_2(-2^{2n})|_{2^{2n}-1}$. For the third term $x_3(2^{2n-1} + 2^{n-1} - 1)$ we transform 2^{2n-1} to $2^{2n} - 2^{2n-1}$. Due to the modulo reduction in $|2^{2n} - 1 - 2^{2n-1} + 2^{n-1}|_{2^{2n}-1}$ we get $|-2^{2n-1} + 2^{n-1}|_{2^{2n}-1}$. Ultimately, we obtain a relationship that will be used to design the converter

$$C(X) = \left| \begin{matrix} x_1(2^{2n-1} + 2^{n-1}) + \\ x_2(-2^n) + \\ x_3(-2^{2n-1} + 2^{n-1}) \end{matrix} \right|_{2^{2n}-1} \quad (21)$$

The first term A: $x_1(2^{2n-1} + 2^{n-1})$ can be realised by shifted concatenation of $x_1$bits. It can be remarked that there is no overlapping of $x_1 2^{2n-1}$ and $x_1 2^{n-1}$. Hence, it can be used as a single operand.

The second term B $x_2(-2^n)$ can be implemented as $x_2(2^{2n} - 1 - 2^n)$ in order to avoid subtraction.

The third term $x_3(-2^{2n-1} + 2^{n-1})$ consists of two parts C: $x_3(-2^{2n-1})$ and D: $x_3 2^{n-1}$. Table 1 shows schema of operands x_1, x_2, x_3 representations for n=3. In order to avoid negative numbers when computing $C(X)$ in (21) we have to replace negative terms by $2^{2n} - 1 - a$ or if $a \geq 2^{2n+2} - 1$ then $2^{2n+2} - 1 - a$.

$$C(X) = \left| \begin{matrix} x_1(2^{2n-1} + 2^{n-1}) + \\ 2^{2n} - 1 - x_2 2^n + \\ x_3 2^{n-1} + 2^{3n-1} - 1 - x_3 2^{2n-1} \\ + (2^{2n} - 1) - |2^{3n-1} - 1|_{2^{2n}-1} \end{matrix} \right|_{2^{2n}-1} \quad (22)$$

For $n = 3$ because $(2^{2n} - 1) - |2^{2n+2} - 1|_{2^{2n}-1} = 3$ we should subtract -3 from the final sum. But it can be replaced by adding 60 to the final sum.

Table 1. Example of binary representation of residues $x_i, i = 1, 2, 3$ from (21) for n=3

	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
A	0	x_1^2	x_1^1	x_1^0	x_1^2	x_1^1	x_1^0	0	0
B	0	0	0	$\bar{x}_2^2$	$\bar{x}_2^1$	$\bar{x}_2^0$	1	1	1
C	0	$\bar{x}_3^2$	$\bar{x}_3^1$	$\bar{x}_3^0$	1	1	1	1	1
D	0	0	0	x_3^3	x_3^2	x_3^1	x_3^0	0	0
E	0	0	0	1	1	1	1	0	0

We can see that there is no bit $\bar{x}_3^3$ because the 4-bit representation has the residue equal to 2^8. Thus, the negation of this bit always gives 0, so $\bar{x}_3^3$ can be omitted. In the Table 2 we show the binary representations of individual operands from (22) for $x = 6$.

Table 2. Example of binary representation of residue values ($x_1 = 6, x_2 = 6, x_3 = 6$) for $n = 3$ and $X = 6$

	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
A	0	1	1	0	1	1	0	0	0	216
B	0	0	0	0	0	1	1	1	1	15
C	0	0	0	1	1	1	1	1	1	63
D	0	0	0	0	1	1	0	0	0	24
E	0	0	0	1	1	1	1	0	0	60

Hence, we obtain for the Table 2 $C(x) = C(6) = |378|_{63} = 0$. Finally using (19) we obtain $x = 6$ at the output, that is the correct result.

Converter architecture

The converter architecture is built of the CSA tree adder with end-around-carry (EAC) and modulo $2^{2n} - 1$ carry propagate adder (CPA). The CSA consists of three layers that sum up the operands consisting of shifted individual bits of the residues and inverses of selected bits as shown in Table 1. First two layers sum up the individual terms from (22) and the third layer is used for adding EAC bits resulting from bits exceeding $2^{2n} - 1$. The inversion of bits is the result of the replacing subtraction modulo $2^{2n} - 1$ by the negation of the operands as in (22). In the third layer EAC bits resulting from the reduction powers of 2 modulo $2^{2n} - 1$ are added. Finally, the CSA carry and sum vectors are added by the CPA that performs addition of operands and recurrent carry bit obtained as a result of the reduction modulo $2^{2n} - 1$ MSB in the CPA. In Fig. 1 discussed converter architecture for $n = 3$ is given.

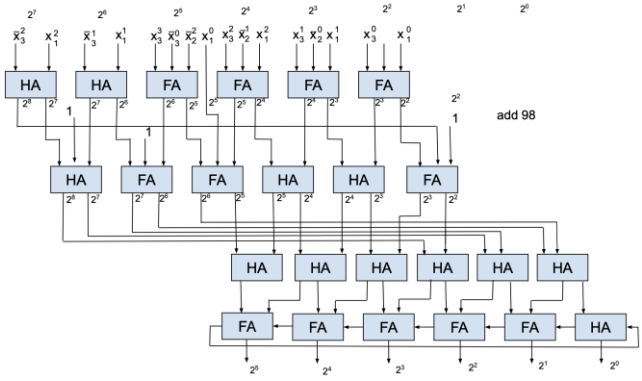

Fig. 1. New converter architecture for $n = 3$

Evaluation and comparison

Several converter architectures have been presented

[17-31]. We shall compare the delay and area with the probably most effective converter given in [28].

For the new converter the approximate area can be written as

(23) $$A_{new} = (6n+1)A_{FA}$$

where additional A_{FA} represents equivalent of $2n+2$ inverters. The delay can be expressed as

(24) $$t_{new} = t_{inv} + 3t_{FA} + 2t_{CPA}$$

We do not include the area and delay of $2^{2n}-1$ modulo adder as in other works. We can remark that the delay generally does not depend on n.

In Table 3 we have compared our converter architecture with [28] and [30] because they are probably the best known converters with respect to hardware and time complexity. Converter from [30] does not explicitly contains the CSA that performs the EAC that would need an additional layer. In fact the CE converter requires additionally an equivalent of n FAs due to AND/OR gates. The additional delay caused by one FA is not important for pipelined implementations because in new technologies t_{FA} may be less than $50\ ps$.

Table 3. Comparison of hardware and time complexity of the converters

Converter	FA	AND/OR	XOR	Delay
[28] CE	$4n+1$	$2n-1$	$2n$	$2t_{FA}+2t_{CPA}$
HS	$6n+1$	$2n-1$	$2n$	$2t_{FA}+2t_{CPA}+t_{MUX}$
[30]	$4n$	0	0	$2t_{FA}+2t_{CPA}$
New	$6n+1$	0	0	$t_{inv}+3t_{FA}+2t_{CPA}$

Summary

In this paper we presented a new algorithm of residue-to-binary conversion based on core function and converter architecture. The use of core function leads to more simple conversion formulas, that also facilitate the converter design. The converter has the comparable time-hardware complexity with the best known converters.

***Authors**: dr hab. inż. Maciej Czyżak, Akademia Nauk Stosowanych w Elblągu, Instytut Informatyki Stosowanej im. K. Brzeskiego, ul. Wojska Polskiego 1, Elbląg, E-mail: m.czyzak@ans-elblag.pl; dr inż. Robert Smyk, Wydział Elektrotechniki i Automatyki, Politechnika Gdańska, ul. Narutowicza 11/12, Gdańsk, E-mail: robert.smyk@pg.edu.pl .*

REFERENCES

[1] Sozański K., Digital signal processing in power electronics control circuits, 2 ed., Springer Verlag London Ltd, 2017.
[2] Rebizant W., Szafran J., Wiszniewski A., Digital signal processing in power systems protection and control (Signals and Communication Technology), Springer Verlag London Ltd, 2011.
[3] Barzegaran M.R., Youssef T. A, Berzoy A., Mohammed O.A., Electric machine drive design improvements through control and digital signal processing techniques, IEEE Transactions on Energy Conversion, 30, (3), 1255-1264, 2015.
[4] Moynihan F., Fundamentals of DSP-based control for ac machines, Analog Devices, vol. 34, 2000.
[5] Toliyat, H.A., Campbell S.G., DSP-Based electromechanical motion control, CRC Press, 20003.
[6] Czyżak, M., Digital structures for high-speed digital signal processing, Wyd. Pol. Gdańskiej, 2013.
[7] Mohan, P.A., Residue Number Systems: Algorithms and Architectures (2016). Springer Science & Business Media.
[8] Smyk, R., Czyżak, M., High-speed binary-to-residue converter design Using 2-bit segmentation of the input word (2022). Scientific Journal of Gdynia Maritime University, 42-56.
[9] Piestrak, S. J., Design of high-speed residue-to-binary number system converter based on Chinese remainder theorem (1994). In Proceedings 1994 IEEE International Conference on Computer Design: VLSI in Computers and Processors, 508-511.
[10] Czyżak, M., An improved high-speed residue-to-binary converter based on the Chinese Remainder Theorem (2007). Pomiary Automatyka Kontrola, 53(4), 72-73.
[11] Chakraborti, Soundararajan, and Reddy. An implementation of mixed-radix conversion for residue number applications. IEEE Transactions on Computers (1986): 762-764.
[12] Miller, D. F., McCormick W. S., An arithmetic free parallel mixed-radix conversion algorithm, IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing (45), (1998), 158-162.
[13] Van Vu, Efficient implementations of the Chinese remainder theorem for sign detection and residue decoding. IEEE Transactions on Computers (1985): 646-651.
[14] Wang, Y., Residue-to-binary converters based on new Chinese remainder theorems (2000). IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, 47(3), 197-205.
[15] Czyżak, M., RNS reverse converter based on core function, SPETO (2008): 141-142.
[16] Huang, C., A fully parallel mixed-radix conversion algorithm for residue number applications. IEEE Transactions on computers100.4 (1983): 398-402.
[17] Bernardson, P., Fast memoryless, over 64 bits, residue-to-binary convertor (1985). IEEE Transactions on Circuits and Systems, 32(3), 298-300.
[18] Ibrahim, K.M., Saloum, S.N., An efficient residue to binary converter design (1988). IEEE Transactions on Circuits and Systems, 35(9), 1156–1158.
[19] Sweidan, A., Hiasat, A., A new efficient memoryless residue to binary converter (1988). IEEE Transactions on Circuits and Systems, 35(11), 1441–1444.
[20] Dhurkadas, A., Comments on" An efficient residue-to-binary converter design" by KM Ibrahim and SN Saloum, (1990). IEEE Transactions on Circuits and Systems, 37(6), 849-850.
[21] Piestrak, S. J., A high-speed realization of a residue to binary number system converter (1995). IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, 42(10), 661-663.
[22] Dhurkadas, A., Comments on "A high speed realization of a residue to binary number system converter" (1998). IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, 45(3), 446-447.
[23] Wang, Y., Residue-to-binary converters based on new Chinese remainder theorems (2000). IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, 47(3), 197-205.
[24] Wang, Z., Jullien, G.A., Miller, W.C., An improved residue-to-binary converter (2000). IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, 47(9), 1437–1440.
[25] Patronik, P., Piestrak, S. J., Design of Reverse Converters for General RNS Moduli Sets, (2014). IEEE Transactions on Circuits and Systems I: Regular Papers, 61(6), 1687-1700.
[26] Miller, D.D., Altschul, R.E., King J.R., Polky, J.N., Analysis of the residue class core function of Akushskii, Burcev, Pak, in Residue Number System Arithmetic: Modern Applications in Digital Signal Processing, Aug. 1986, 390-401.
[27] Patronik, P., Piestrak, S.J., Design of reverse converters for the general 3-moduli set $\{2^k, 2^n-1, 2^n+1\}$, EURASIP Journal on Advances in Signal Processing, (2023), 2023-92.
[28] Piestrak, S. J., A high-speed realization of a residue to binary number system converter (1995). IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, 42(10), 661-663.
[29] Wang, Y., Song, X., Aboulhamid, M., Shen, H., Adder based residue to binary number converters for (2n − 1, 2n , 2n + 1). (2002). IEEE Transactions on Signal Processing, 50(7), 1772–1779.
[30] Wang, Z., Jullien, G.A., Miller, W.C., An improved residue-to-binary converter (2000). IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, 47(9), 1437–1440.
[31] Ibrahim, K.M., Saloum, S.N., An efficient residue to binary converter design (1988). IEEE Transactions on Circuits and Systems, 35(9), 1156–1158.