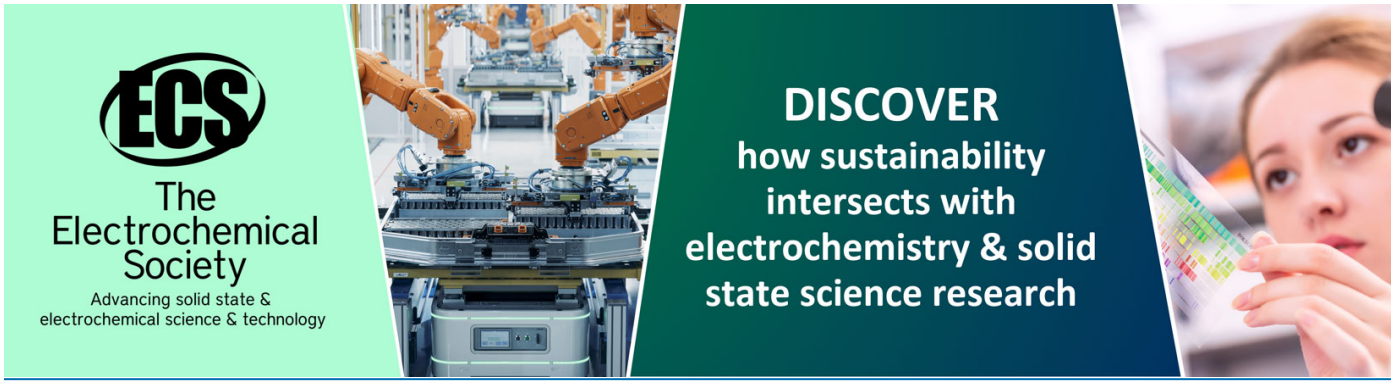
Designing forward converters for data transmission systems in two-level RNS



IOP Conf. Series: Journal of Physics: Conf. Series **1352** (2019) 012005 doi:10.1088/1742-6596/1352/1/012005

Designing forward converters for data transmission systems in two-level RNS

M V Bergerman[1], P A Lyakhov[1], N I Chervyakov[1], D I Kaplun[2] and D V Bogaevskiy[2]

[1]Department of Applied Mathematics and Mathematical Modeling, Institute of mathematics of Natural Sciences, North Caucasus Federal University, Stavropol, Russia
[2]Department of Automation and Control Processes, Saint Petersburg Electrotechnical University "LETI", Saint Petersburg, Russia

E-mail: maxx07051997@inbox.ru

Abstract. We are considering the use of a two-stage Residue number system (RNS) in data transmission systems with a special set of modules $\{2^{l_1}, 2^{l_2}-1, 2^{l_3}-1, ..., 2^{l_p}-1\}$. We propose the forward converter with such type of RNS and demonstrate its advantage in hardware costs and delay. This article discusses the FPGA simulation of a forward conversion circuit to a two-stage RNS using adders and the standard modulo function and their comparison in terms of delay and hardware costs.

1. Introduction

For intensive applications, data often need advanced computer architectures with a large number of cores, a deep memory hierarchy, and super-fast input/output (I/O) [1].

Currently, there is a great practical need to develop efficient data transfer systems that use the parallelism of modern computer architecture.

One of the ways to solve this problem is to use Residue number system (RNS) and its fault tolerance properties [2, 3].

RNS is a number system that splits a number into parts (residues) and performs arithmetic operations in parallel for each residue without the need for transfer, which leads to significant acceleration over the corresponding binary operations [4].

RNS is well suited for applications in which most of the computations are additions, subtractions, and multiplications.

Excessive RNS used to improve the reliability of data transmission systems due to the possibility of recovering corrupted data by introducing additional modules of the system [8].

In this paper, we propose to use a two-level RNS for data transfer, since the introduction of a two-level RNS allows data to be converted to a very low-bit format, which can be very important for systems with low power consumption, for example, MANET networks [9].

In addition, we will show the inexpediency of using a two-level RNS to perform arithmetic operations, set out our vision of the approach to the choice of two-level RNS modules, and proposeforward converters for such a system.

2. Background on Residue Number System

In the RNS the numbers are represented in the basis of mutually prime numbers, called moduli $\beta = \{m_1,...,m_n\}, GCD(m_i, m_j) = 1$, for $i \neq j$. The product of all modules RNS $M = \prod_{i=1}^{n} m_i$ calls the dynamic range of the system. Any integer $0 \leq X < M$ can be uniquely represented in RNS as a vector $\{x_1, x_2, ..., x_n\}$, where $x_i = |X|_{m_i} = X \bmod m_i$.

The operations of addition, subtraction, and multiplication in the RNS determined by the formulas

$$A \pm B = \left(|a_1 \pm b_1|_{m_1}, ..., |a_n \pm b_n|_{m_n} \right), \quad (1)$$

$$A \times B = \left(|a_1 \times b_1|_{m_1}, ..., |a_n \times b_n|_{m_n} \right). \quad (2)$$

Restoration of the number X from residues $\{x_1, x_2, ..., x_n\}$ based on the Chinese remainder theorem (CRT)

$$X = \left| \sum_{i=0}^{n} \left| |M_i^{-1}|_{m_i} x_i \right|_{m_i} M_i \right|_M, \quad (3)$$

where $M_i = \frac{M}{m_i}$. Element $|M_i^{-1}|_{m_i}$ means multiplicative inverse element for M_i, modulo m_i.

3. Practical use of two-level RNS

Let the main RNS consists of moduli $m_1, m_2, ..., m_s$ and this system provides the ability to perform operations in a range $[0; M)$, where $M = m_1 \cdot m_2 \cdot ... \cdot m_s$. Consider the m_s module from the main RNS. The largest number that can be obtained in this digit when multiplying numbers modulo m_s is $(m_s - 1)^2, s \in N, s \in [1, n]$ [3]. We present data modulo m_s from the main base system in the new RNS with moduli $\mu_1, \mu_2, ..., \mu_k$, such that

$$\mu = \mu_1 \cdot \mu_2 \cdot ... \cdot \mu_k \geq (m_s - 1)^2, (\mu_i, \mu_j) = 1, i \neq j. \quad (4)$$

Failure to comply with the condition (4) does not guarantee the correct result of the multiplication operation in a two-level RNS, which is confirmed by the following example.

Example 1. Consider a two-level RNS proposed in [10]. In it, the main base system RNS contains modules 3, 4, and 5, and the range of the system is $M = 3 \cdot 4 \cdot 5 = 60$. The authors of [10] convert the data on module 5 into a new RNS with modules 3, 4. According to the formula (4) we have $(5-1)^2 = 16$ while the range of the new RNS is equal $M = 3 \cdot 4 = 12$. Perform the multiplication of numbers 3 and 4. In the main RNS system, the numbers look like this: $3 = (0, 3, 3), 4 = (1, 0, 4)$. In the new RNS: $3 = [0, 3, (0, 3)_5], 4 = [1, 0, (1, 0)_5]$. Their product is equal: $3 \cdot 4 = [0 \cdot 1, 3 \cdot 0, (0 \cdot 1, 3 \cdot 0)_5] = [0, 0, (0, 0)_5]$, Transferring data from the new RNS to the main base system leads to the result, while the correct result in the first level is: $3 \cdot 4 = (0 \cdot 1, 3 \cdot 0, 3 \cdot 4) = (0, 0, 2) = 12$.

The construction of a two-level RNS by the formula (4) ensures the correctness of the result when performing exactly one multiplication in such a system. However, the following example shows that the use of a two-level RNS built based on a formula (4) is impractical when performing more than one multiplication.

Example 2. Consider the main base system RNS with moduli $m_1 = 7, m_2 = 31, m_3 = 32.$ The range of such a system is equal $M = 7 \cdot 31 \cdot 32 = 6944.$ We convert the data modulo $m_3 = 32$ into a new RNS with moduli $\mu_1 = 7,\ \mu_2 = 11,\ \mu_3 = 13,$ built by the formula (4), as well $(32-1)^2 = 961$ and $\mu = 7 \cdot 11 \cdot 13 = 1001.$ Consider the product of three numbers: 15, 17 and 20 in such a two-level RNS. In the main base system, these numbers will look like this: $15 = (1,15,15), 17 = (3,17,17), 20 = (6,20,20).$ In the new RNS, these numbers will look like $15 = [1,15,(1,4,2)_{32}], 17 = [3,17,(3,6,4)_{32}], 20 = [6,20,(6,9,7)_{32}].$ Performing multiplication in a two-level RNS leads to the result $15 \cdot 17 \cdot 20 = [1 \cdot 3 \cdot 6, 15 \cdot 17 \cdot 20, (1 \cdot 3 \cdot 6, 4 \cdot 6 \cdot 9, 2 \cdot 4 \cdot 7)_{32}] = [4,16,(4,7,4)_{32}]$. Translation of data from the new RNS into the main base system leads to the result $[4,16,(4,7,4)_{32}] = (4,16,31) = 543,$ while the correct result in the first level is equal $15 \cdot 17 \cdot 20 = (1 \cdot 3 \cdot 6, 15 \cdot 17 \cdot 20, 15 \cdot 17 \cdot 20) = (4,16,12) = 5100$.

The analyzed examples show that the construction of systems that require a large number of arithmetic operations using multi-levels RNS is impractical. In particular, it is not very promising to design digital signal processing systems and cryptographic systems using multi-layered and nested RNS, since these applications require intensive multiplication and addition operations.

The only promising, in our opinion, the use of multi-level RNS is the development of data transmission systems. In the absence of arithmetic operations in a two-level RNS, condition (4) can be replaced by

$$\mu = \mu_1 \cdot \mu_2 \cdot \ldots \cdot \mu_k \geq m_s - 1, (\mu_i, \mu_j) = 1, i \neq j. \tag{5}$$

Figure 1 shows the scheme of data transmission, presented in the traditional binary number system, using two-levels RNS. The information presented in the binary number system is converted to the format of the first-level RNS by modules $m_1, m_2, \ldots, m_s$.

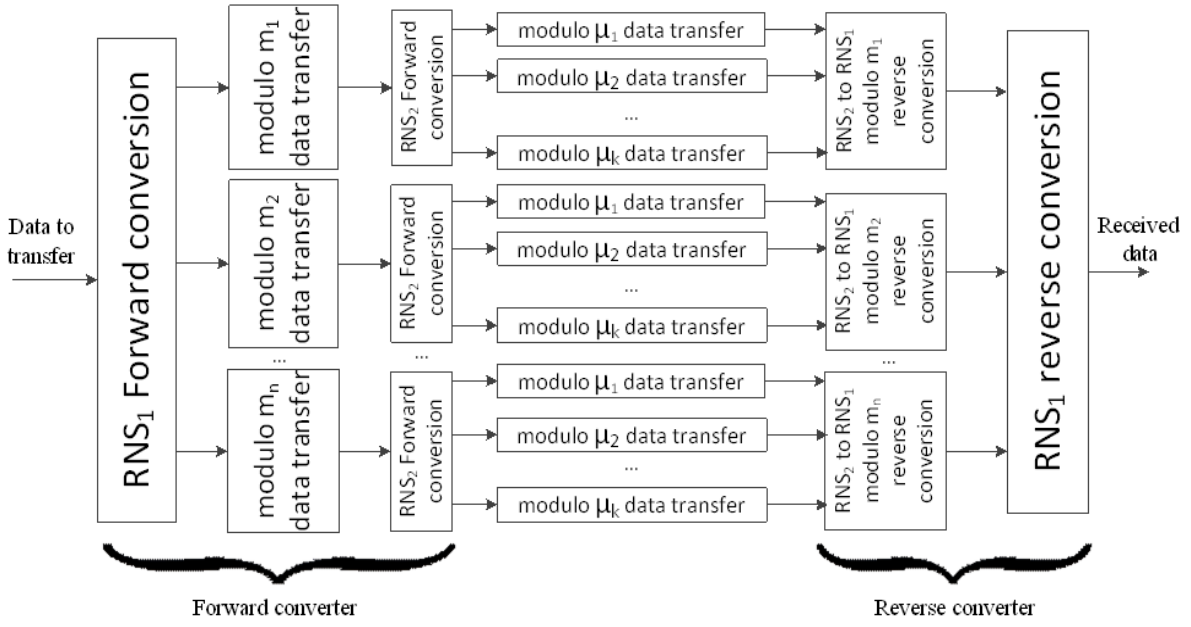


Figure 1. Transformation of information using a two-level RNS.

After that, the data of each digit of the first-level RNS is converted into the second-level of the RNS by the moduli $\mu_1, \mu_2, ..., \mu_k$, that are selected in accordance with formula (5), and transmitted over the communication channel. The received data for the moduli $\mu_1, \mu_2, ..., \mu_k$, is converted into bits of the firstlevel of the RNS by modules $m_1, m_2, ..., m_s$, after which the final inverse conversion from the firstlevel of the RNS to the traditional binary number system is performed.

In the remainder of this article, we will consider the aspects of choosing a set of bases for the system presented in figure 1 and propose a forward conversion device for such a two-level RNS.

4. Device description

Consider a special set of modules of the form $\{2^{l_1}, 2^{l_2}-1, 2^{l_3}-1, ..., 2^{l_p}-1\}$. To translate the number in the RNS it is necessary to calculate the residuals from the division for each of the modules [11]. The operation of calculating the remainder of the modulo 2^{l_1} division is performed by simply trimming the l_1 low bits of the source number. For modules $2^{l_2}-1, 2^{l_3}-1, ..., 2^{l_p}-1$, the calculation of the remainder of the division is more complicated. Consider the process of finding the remainder of dividing a number $X = \overline{X_{l-1}X_{l-2}\ldots X_0}$ by a number 2^n-1. Divide X into $m = \lceil l/n \rceil$ numbers of n bits. To do this, we add to X the left zeros, to the bit $l' = m \cdot n$, now $X' = \overline{X_{l'-1}X_{l'-2}\ldots X_0}$. Then $Y_0 = \overline{X_{n-1},..,X_1,X_0}$, $Y_1 = \overline{X_{2n-1},..,X_{n+1},X_n}$, ..., $Y_m = \overline{X_{l'-1},..,X_{(m-1)n+1},X_{(m-1)n}}$. Imagine the number X' as $X' = Y_0 + Y_1 \cdot 2^n + Y_2 \cdot 2^{2n} + ... + Y_m \cdot 2^{mn}$. In this way $|X'|_{2^n-1} = |Y_0 + Y_1 + Y_2 + ... + Y_m|_{2^n-1}$ whence it follows that there is a calculation of the remainder of dividing modulo $2^{l_p}-1$ reduces to adding n-bit numbers modulo 2^n-1.

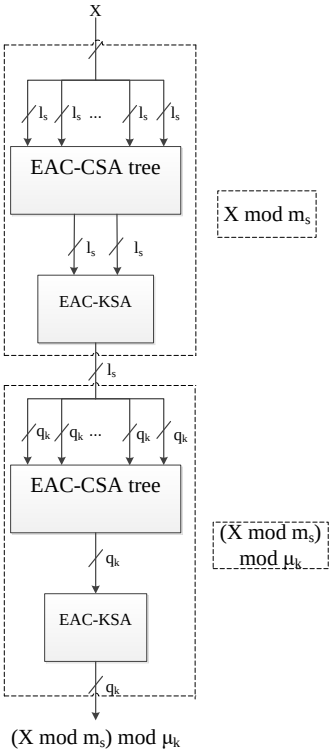


Figure 2. The scheme of calculating the remainder of the division in a two-level RNS modulo the main RNS and then perform the operation of taking modulo the new RNS.

To find the residuals in the modules of the main and new RNS, we will use adders CSA, KSA [12], EAC-CSA and EAC-KSA [13] (figure 2).

5. Modeling

For the simulation, we chose direct conversion to a two-level RNS, having a bit depth of 16, 32, 64 and 128 bits. In the main RNS selected from 3 to 5 modules. In the new RNS, the range must be greater than the maximum modulus in the main RNS, in accordance with formula (5). Table 1 shows the selected modules for the main and new RNS.

Table 1. The values of the modules of two-level RNS for the selected ranges.

Range, bits	Number of modules	Modules m_i	Modules μ_i
16	3	$(2^6, 2^6\text{-}1, 2^5\text{-}1)$	$(2^3\text{-}1, 2^2, 2^2\text{-}1)$
	4	$(2^5, 2^5\text{-}1, 2^4\text{-}1, 2^3\text{-}1)$	$(2^3\text{-}1, 2^2\text{-}1, 2^1)$
32	3	$(2^{12}, 2^{11}\text{-}1, 2^{10}\text{-}1)$	$(2^5\text{-}1, 2^3, 2^3\text{-}1, 2^2\text{-}1)$
	4	$(2^9, 2^9\text{-}1, 2^8\text{-}1, 2^7\text{-}1)$	$(2^4\text{-}1, 2^3, 2^3\text{-}1)$
	5	$(2^{10}, 2^8\text{-}1, 2^7\text{-}1, 2^5\text{-}1, 2^3\text{-}1)$	$(2^5\text{-}1, 2^3, 2^3\text{-}1)$
64	3	$(2^{22}, 2^{22}\text{-}1, 2^{21}\text{-}1)$	$(2^7\text{-}1, 2^5\text{-}1, 2^4, 2^4\text{-}1, 2^3\text{-}1)$
	4	$(2^{17}, 2^{17}\text{-}1, 2^{16}\text{-}1, 2^{15}\text{-}1)$	$(2^6, 2^5\text{-}1, 2^4\text{-}1, 2^3\text{-}1)$
	5	$(2^{16}, 2^{16}\text{-}1, 2^{13}\text{-}1, 2^{11}\text{-}1, 2^9\text{-}1)$	$(2^5, 2^5\text{-}1, 2^4\text{-}1, 2^3\text{-}1)$
128	3	$(2^{44}, 2^{43}\text{-}1, 2^{42}\text{-}1)$	$(2^{11}\text{-}1, 2^{10}, 2^9\text{-}1, 2^8\text{-}1, 2^7\text{-}1)$
	4	$(2^{33}, 2^{33}\text{-}1, 2^{32}\text{-}1, 2^{31}\text{-}1)$	$(2^9\text{-}1, 2^8\text{-}1, 2^7\text{-}1, 2^5, 2^5\text{-}1)$
	5	$(2^{28}, 2^{27}\text{-}1, 2^{26}\text{-}1, 2^{25}\text{-}1, 2^{23}\text{-}1)$	$(2^8\text{-}1, 2^7\text{-}1, 2^6, 2^5\text{-}1, 2^3\text{-}1)$

Hardware modeling was carried out in the Xilinx ISE Design Suite 14.7 environment using the VHDL language. Devices with 16 and 32-bit ranges were implemented on the Xilinx xc6slx45-3fgg676 board. Devices with 64 and 128-bit ranges were implemented on the vx690t-3ffg1157 board. The simulation results are presented in table 2.

Table 2. The results of hardware modeling of devices for direct data conversion from the binary number system to two-level RNS.

Range, bits	Number of modules	Built-in modulo function			Proposed method		
		Delay, ns	Slices	LUT Slices	Delay, ns	Slices	LUT Slices
16	3	63.481	286	854	12.932	28	66
	4	63.542	335	968	12.735	26	64
32	3	153.991	1511	4370	20.351	91	219
	4	165.139	1487	4854	16.488	68	177
	5	141.343	1944	5614	17.844	76	202
64	3	211.598	6906	20721	26.813	278	740
	4	219.732	7944	21982	23.590	205	566
	5	221.270	8808	24595	24.112	201	563
128	3	594.979	26830	85594	35.620	603	1584
	4	514.119	31285	94338	35.471	490	1487
	5	469.069	31007	105027	33.214	641	1703

Summarizing the obtained results, we can conclude that in all respects the method proposed by us for fast calculation and the lowest hardware costs wins. In numbers, then with 16 bits for the delay, the proposed method is 5 times faster than the standard modulo function, and for hardware costs, more than 10 times. At 32 bits, our method is about 10 times faster than the built-in function, and in terms of

hardware costs, it is more than 20 times. At 64 bits - 10 times faster, in terms of hardware costs more than 40 times. At 128 bits - about 15 times faster, in terms of hardware costs more than 50 times.

6. Conclusion

The article demonstrated the unsuitability of a two-level RNS to perform a large number of arithmetic operations and proposed a method for using two-level RNS in data transmission systems. The results of the hardware simulation showed that the use of view modules $m_s = \{2^{m_1}, 2^{m_2}-1, 2^{m_3}-1, ..., 2^{m_p}-1\}$ for the main RNS and the modules of the form $q_k = \{2^{q_1}, 2^{q_2}-1, 2^{q_3}-1, ..., 2^{q_t}-1\}$ for the new RNS allows obtaining a significant gain in computation speed and hardware costs compared with the built-in modulo operation. More specifically, the simulation results on 16, 32, 64 and 128 bits showed that the use of the method proposed by us yields a gain in speed of 10–15 times, on average, depending on the bit depth, and hardware costs of 10–50 times depending on bit depth compared to the built-in modulo function.

The approach developed in the article makes it possible to reduce the transfer of data of high bit depth to the transfer of small blocks with a bit width not exceeding 8 bits, which opens up new opportunities for energy efficient communication in systems for which low power consumption is a key factor, for example, in wireless sensor networks.

References

[1] Li T, Ren Y, Yu D and Jin S 2017 RAMSYS: Resourse-Aware Asynchronus Data Transfer with Multicore Systems *IEEE Trans. Parallel Distrib. Syst.* Vol **28 5** 1430–1444
[2] Omondi A and Premkumar B 2007 Residue Number Systems: Theory and Implementation *Imperial College Press*
[3] Parhami B 2009 Computer arithmetic 2nd ed p cm *Oxford University Press Inc*
[4] Akusskij I J and Judickij D I 1968 Machine Arithmetic in Residual Classes (Moscow: Sovetskoje Radio)
[5] Rossi D 2013 Application Space Exploration of a Heterogeneous Run-Time Configurable Digital Signal Processor *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* Vol **21 2** 193–205
[6] Ramires J et al. 2002 RNS-Enabled Digital Signal Processor Design *Electronics Letters* Vol **38 6** 266–268
[7] Zhu S, Zhu C, Cui H and Wang W 2019 A Class of Quadratic Polynomial Chaotic Maps and its Application in Cryptography *IEEE Access* Vol **7** 34141–34152
[8] Mu L, Liu X and Yang L 2018 Redundant Residue Number System Coded Diffusive Molecular Communications *IEEE Conferences 10th International Conference on Wireless Communication and Signal Processing (WCSP)* 1–6
[9] Doss S, Nayyar A, Suseendran G et al. 2018 APD-JFAD: Accurate Prevention and Detection of Jelly Fish Attack in MANET *IEEE Access* Vol **6** 56954–56965
[10] Nakahara H and Sasao T 2018 A High-speed Low-power Deep Neural Network on an FPGA based on the Nested RNS: Applied to an Object Detector *IEEE International Symposium on Circuits and Systems (ISCAS)* 1–5
[11] Tomczak T 2011 Hierarchical residue number systems with small moduli and simple converters (Wrocław Poland: Institute of Computer Engineering, Control and Robotics Wrocław University of Technology) 173–192
[12] Kogge P M 1973 A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations *IEEE Transaction on computers* ed P M Kogge and H S Stone Vol **C-22 8** 786–793
[13] Vergos H T 2012 On Modulo Adder Design *IEEE Transactions on computers* ed Vergos G Dimitrakopoulos Vol **61 2** 173–186