Contemporary Engineering Sciences, Vol. 1, 2008, no. 2, 71 - 80

A Multiplier-Free Residue to Weighted Converter for the Moduli Set $\{3^n - 2, 3^n - 1, 3^n\}$

Amir Sabbagh Molahosseini and Mehdi Hosseinzadeh

Islamic Azad University, Science and Research Branch, Tehran, Iran
amir.sabbagh@sr.iau.ac.ir , hosseinzadeh@sr.iau.ac.ir

Keivan Navi

Faculty of Electrical and Computer Engineering
Shahid Beheshti University, Tehran, Iran
navi@sbu.ac.ir

Abstract

The residue number system (RNS) is a carry-free number system which can support high-speed and parallel arithmetic. One of the major issues in efficient design of RNS systems is the residue to weighted conversion. In this paper, we present an efficient design of residue to weighted converter for the newly introduced moduli set $\{3^n–2, 3^n–1, 3^n\}$, based on mixed-radix conversion (MRC) algorithm. The proposed residue to weighted converter is adder-based and memory-less which can results in a high-performance hardware. The proposed residue to weighted converter has better performance and also eliminates the use of multiplier, compared to the last work.

Keywords: Computer arithmetic, Residue number system (RNS), Multiple-valued logic (MVL), Residue to weighted converter

1 Introduction

The residue number system (RNS) is a non-weighted number system which speed up arithmetic operations by dividing them into smaller parallel operations. Since the arithmetic operations in each moduli are independent of the others, there is no carry propagation among them and so RNS leads to carry-free addition, multiplication and borrow-free subtraction [1]. One of the major issues in efficient design of RNS systems is the residue to weighted conversion. The algorithms of

residue to weighted conversion are mainly based on chinese remainder theorem (CRT), mixed-radix conversion (MRC) [1] and new chinese remainder theorems (New CRTs) [2]. In addition to these, novel conversion algorithms [3] which are designed for some special moduli sets have been proposed.

Multiple-valued logic (MVL) has been proposed as a means for reducing the power, improving the speed, and increasing the packing density of VLSI circuits [4]. In MVL, the number of discrete signal values or logic states extends beyond two. Arithmetic units implemented with MVL achieve more efficient use of silicon resource and circuit interconnections [5]. There is a clear mathematical attraction of using multiple-valued number representation in RNS. The modular arithmetic that is inherent in MVL can be match with modular arithmetic needed in RNS.

The first MVL-RNS system was introduced by Soderstrand et al. [6] to design a high speed FIR digital filter. The residue to weighted converter proposed in [6] is based on chinese remainder theorem (CRT) and implemented with read-only memories (ROM's). this converter is practical to implement small and medium RNS dynamic ranges and it is not appropriate for large dynamic ranges. In [7], new RNS systems based on the moduli of forms r^a, r^b-1 and r^c+1 are presented. Abdallah et al. in [7] developed a systematic framework utilizing high-radix arithmetic for efficient MVL-RNS implementations and proposed many radix-r moduli sets. This moduli sets are not include pairwise relatively prime moduli, and this resulting in reduced dynamic ranges and unbalanced moduli. The residue to weighted converter presented in [7] is based on CRT and because of the scale-down factors which are used for making moduli pairwise relatively prime, conversion delay and cost are increased. Recently, In [8] a new residue number system based on the moduli set $\{3^n-2, 3^n-1, 3^n\}$ was introduced. This moduli set includes pairwise relatively prime and balanced moduli that offers large dynamic range and simple realization of related circuits. The residue to weighted converter presented in [8] is based on CRT and requires multipliers and large modulo adder, so its area and delay complexities have been increased.

In this paper, we developed a two-level MRC algorithm for designing an efficient residue to weighted converter for the moduli set $\{3^n - 2, 3^n - 1, 3^n\}$. The proposed hardware architecture for residue to weighted converter is multiplier-free and memoryless. In comparison with the residue to weighted converter proposed in [8], our converter has better performance in terms of area and delay.

2 Background

A residue number system is defined in terms of a relatively-prime moduli set $\{P_1,P_2, \ldots,P_n\}$ that is $\gcd(P_i,P_j)=1$ for $i \neq j$ and $i, j = 1,2,\ldots,n$. A weighted number X can be represented as $X=(x_1,x_2, \ldots ,x_n)$, where

$$x_i = X \bmod P_i = \left|X\right|_{P_i}, 0 \leq x_i < P_i \quad (1)$$

Such a representation is unique for any integer X in the range $[0,M-1]$, where

$M=P_1P_2\ldots P_n$ is the dynamic range of the moduli set $\{P_1,P_2, \ldots,P_n\}$ [9].

Addition, subtraction and multiplication on residues can be performed in parallel without any carry propagation among the residue digits. Hence, by converting the arithmetic of large numbers to a set of the parallel arithmetic of smaller numbers, the RNS representation yields significant speed up.

The algorithms of residue to weighted conversion are based mainly on chinese remainder theorem (CRT) and mixed-radix conversion (MRC).

Chinese Remainder Theorem: by CRT, the number X is calculated from residues by

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

where $M_i = M/P_i$ and $N_i = | M_i^{-1} |_{P_i}$ is the multiplicative inverse of M_i modulo P_i.

Mixed-Radix Conversion [10]: the weighted number X can be computed by

$$X = a_n \prod_{i=1}^{n} P_i + \ldots + a_3 P_2 P_1 + a_2 P_1 + a_1 \tag{3}$$

where a_is are called the mixed-radix coefficients and they can be obtained from the residues by

$$a_n = \left| (((x_n - a_1)\left| P_1^{-1} \right|_{P_n} - a_2)\left| P_2^{-1} \right|_{P_n} - \cdots - a_{n-1})\left| P_{n-1}^{-1} \right|_{P_n} \right|_{P_n} \tag{4}$$

where $n>1$ and $a_1=x_1$.

For a simple 2-moduli set $\{P_1,P_2\}$, the number X can bconverted from its residue representation (x_1,x_2) by

$$X = a_1 + a_2 P_1 = x_1 + P_1 \cdot \left| (x_2 - x_1)\left| P_1^{-1} \right|_{P_2} \right|_{P_2} \tag{5}$$

where $| P_1^{-1} |_{P_2}$ is the multiplicative inverse of P_1 modulo P_2.

The RNS with Moduli Set $\{3^n - 2, 3^n - 1, 3^n\}$: In [8], a new moduli set $\{3^n -2, 3^n -1, 3^n\}$ was introduced for RNS. This moduli set contains pairwise relatively prime and balanced moduli which can offer large dynamic range and fast internal RNS processing. Because of using of high radix (r=3), this RNS can be simply realized in ternary-valued logic (TVL). Addition circuits for moduli set $\{3^n -2, 3^n -1, 3^n\}$ can be obtained by using the method of [8] as follow.

If we consider three numbers A, B and C as the residues in respect of the modulo m, then addition of these numbers in modulo m, can be performed as

$$\begin{cases} A + B + C < m & \Rightarrow A + B + C \\ A + B + C \geq m & \Rightarrow A + B + C - m \end{cases} \tag{6}$$

In other words, if the result is greater than or equal to the moduli, we add it to the complement of the moduli and ignore the carry out. For performing the addition operation in the modulo 3^n, we add up two numbers and as the carry out is a multiple of 3^n, we simply ignore the carry out. The corresponding circuit is illustrated in Fig. 1. It should be noted that the full adder (FA) basic cell in ternary is a 4-input adder cell [7].

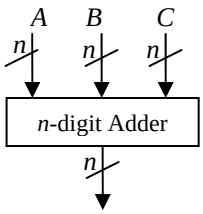

Fig. 1. Modulo 3^n adder

In modulo 3^n–1 if the result is greater than or equal to the 3^n–1 then the result will be added to the complement of the modulo, i.e. $3^n - (3^n-1)=1$. By using parallelism, the result and the same result plus one are generated simultaneously and by using a multiplexer, the correct value will be directed to the output. The corresponding circuit is presented in Fig. 2.

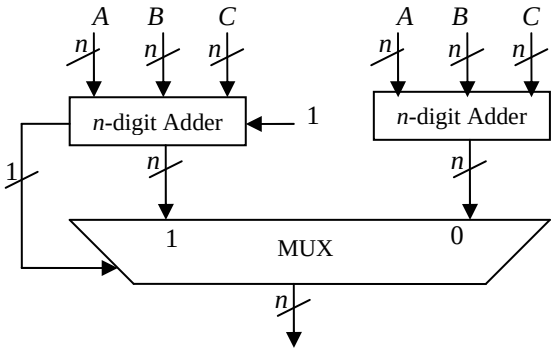

Fig. 2. Modulo 3^n –1 adder

In modulo 3^n–2, if the result of addition is greater than or equal to 3^n–2 then it will be added to the complement of the modulo which is 3^n–(3^n–2)=2. An interesting property of TVL is the possibility of having a carry in generated equal to two (in an adder in the base 3, carry in can be between zero and 2). Fig. 3 shows the circuit of this modulo 3^n–2 adder.

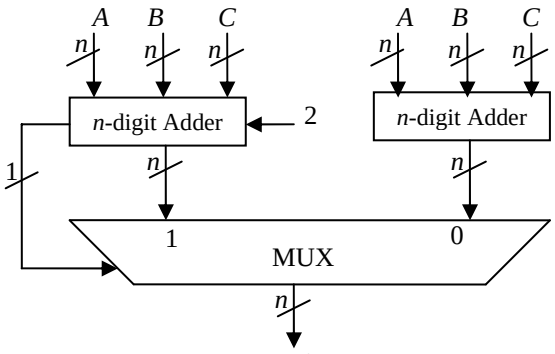

Fig. 3. Modulo 3^n –2 adder

3 Residue to Weighted Conversion Algorithm

We propose a two-level conversion algorithm for the residue to weighted conversion of the moduli set $\{3^n - 2, 3^n - 1, 3^n\}$. In the first level we use a MRC block for combining the two residues. The second level consists of another MRC block combining the result of the first level with the third residue. Fig. 4 shows the block diagram of the proposed residue to weighted converter.

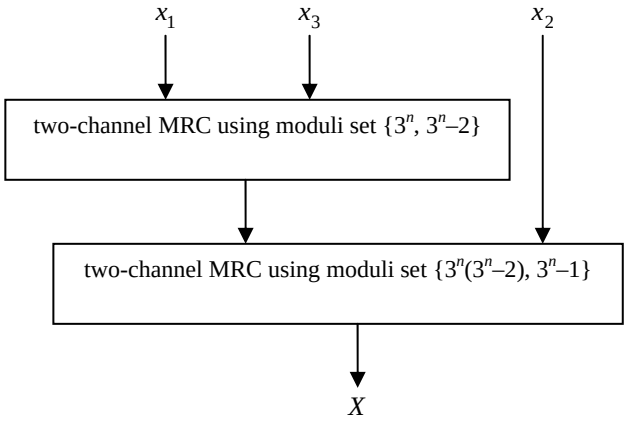

Fig. 4. Block diagram of the proposed converter

The following propositions are needed for the derivation of our algorithm.

Proposition 1: the multiplicative inverse of 3^n-2 modulo 3^n is $k_0=(3^n-1)/2$.

Proof: it is clear that $|3^n-2|_{3^n} = -2$, so

$$\left|k_0 \times (3^n - 2)\right|_{3^n} = \left|\frac{(3^n - 1)}{2} \times (3^n - 2)\right|_{3^n} = \left|-\frac{1}{2} \times -2\right|_{3^n} = 1 \tag{7}$$

Proposition 2: the multiplicative inverse of $3^n(3^n-2)$ modulo 3^n-1 is $k_1= -1$.

Proof: Since $|3^n|_{3^n-1} = 1$ and $|3^n-2|_{3^n-1} = -1$, we have

$$\begin{aligned}\left|k_1 \times 3^n \times (3^n - 2)\right|_{3^n-1} &= \left|-1 \times 3^n \times (3^n - 2)\right|_{3^n-1} \\ &= \left|-1 \times 1 \times -1\right|_{3^n-1} = 1\end{aligned} \tag{8}$$

Consider the three-moduli set $\{3^n - 2, 3^n - 1, 3^n\}$ and let the corresponding residues of the integer X be (x_1,x_2,x_3). Consider the moduli set $\{3^n - 2, 3^n\}$and $Z=(x_1,x_3)$. Using the MRC conversion algorithm (5), Z can be calculated by

$$Z = x_1 + (3^n - 2)\left|k_0 (x_3 - x_1)\right|_{3^n} \tag{9}$$

where

$$\left|k_0 (3^n - 2)\right|_{3^n} = 1 \tag{10}$$

Substituting the value of k_0 from proposition 1 into (9) gives

$$Z = x_1 + (3^n - 2)\left|\frac{(3^n - 1)}{2} \times (x_3 - x_1)\right|_{3^n} \tag{11}$$

The above equation can be rewritten as

$$Z = x_1 + (3^n - 2)T \tag{12}$$

where

$$T = \left| \frac{(3^n - 1)}{2} \times (x_3 - x_1) \right|_{3^n} \tag{13}$$

we know that

$$\frac{(3^n - 1)}{2} = 3^0 + 3^1 + ... + 3^{(n-1)} \tag{14}$$

Therefore, (13) can be rewritten as

$$\begin{aligned} T &= \left| (3^0 + 3^1 + ... + 3^{(n-1)}) \times (x_3 - x_1) \right|_{3^n} \\ &= \left| (3^0 + 3^1 + ... + 3^{(n-1)}) \times V \right|_{3^n} = \left| V^0 + V^1 + ... + V^{(n-1)} \right|_{3^n} \end{aligned} \tag{15}$$

where

$$V = \left| x_3 - x_1 \right|_{3^n} \tag{16}$$

V^is in (15) can be obtained by the i digit left shifting of V. Since the final result of the addition of V^i temrs must be reduced in modulo 3^n, we only need to consider the least significant n digits of V^i terms, and the other digits are ignored as they are multiplies of 3^n. The equation (12) can be rewritten as

$$Z = x_1 + 3^n T - 2T \tag{17}$$

Now, consider the moduli set $\{3^n(3^n–2),\ 3^n–1\}$and $X=(Z,x_2)$. Using the derivation like before, X can be calculated by

$$X = Z + 3^n(3^n - 2)\left| k_1(x_2 - Z) \right|_{3^n-1} \tag{18}$$

where

$$\left| k_1 \times 3^n(3^n - 2) \right|_{3^n-1} = 1 \tag{19}$$

By substituting the value of k_1 from proposition 2, we have

$$X = Z + 3^n(3^n - 2)\left| Z - x_2 \right|_{3^n-1} \tag{20}$$

So, (20) can be rewritten as

$$X = Z + 3^{2n}D - 3^n 2D = (Z + 3^{2n}D) + 3^n(-D - D) \tag{21}$$

where

$$D = \left| Z - x_2 \right|_{3^n-1} \tag{22}$$

Since Z is a $2n$-digit number, we can write

$$D = \left| Z_1 3^n + Z_0 - x_2 \right|_{3^n-1} = \left| Z_1 + Z_0 - x_2 \right|_{3^n-1} \tag{23}$$

where Z_1 and Z_0 have digit level representation as

$$Z_1 = (z_{2n-1} \cdots z_{n+1} z_n) \tag{24}$$

$$Z_0 = (z_{n-1} \cdots z_1 z_0) \tag{25}$$

Example: Given the moduli set $\{3^n– 2,\ 3^n - 1,\ 3^n\}$ where n=2. The residue number (1,4,2) converted into its equivalent weighted number as follow

For n=2 the moduli set is {7,8,9}. So, by substituting values in (11) and (20) we have

$$Z = 1 + 7\left|4 \times 1\right|_9 = 29$$

$$X = 29 + 7 \times 9\left|29 - 4\right|_8 = 92$$

To verify the result, we have

$$x_1 = \left|92\right|_7 = 1$$

$$x_2 = \left|92\right|_8 = 4$$

$$x_3 = \left|92\right|_9 = 2$$

Therefore, the weighted number 92 has RNS representation as (1,4,2) in the RNS with moduli set {7,8,9}.

4 Hardware Implementation

The MRC block of the first level are represented by equations (15)–(17) whereas equations (21) and (23) represent the MRC block of the second level. Details on the first-level and second-level are as follow.

The First Level: Equation (16) can be calculated by a regular *n*-digit ternary adder. Then, (15) is implemented by an *n*-digit ternary multioperand adder which is consists of a *n*-digit ternary carry save adder (CSA) tree followed by a regular *n*-digit ternary adder. Finally, (17) can be calculated by a 2*n*-digit regular ternary adder. It should be noted that since x_1 is an *n*-digit number, no extra hardware is needed for computation of x_1+3^nT. The desired result can be obtained by concatenating x_1 with *T*. Fig. 4(a) shows the hardware implementation of the first level of the residue to weighted converter.

The Second Level: Equation (23) can be performed by an *n*-digit modulo(3^n–1) ternary adder which is shown in Fig. 2. calculation of (21) rely on an *n*-digit ternary adder followed by a 3*n*-digit regular ternary adder. Like before, since *Z* is a 2*n*-digit number, no extra hardware is needed for computation of $Z+3^{2n}D$. Fig. 4(b) shows the hardware implementation of the second level of the residue to weighted converter.

As shown in figures 1 and 2, the proposed residue to weighted converter for the moduli set {3^n – 2, 3^n – 1, 3^n} is multiplier-free and consists of ternary adders. The residue to weighted converter for the moduli set {3^n – 2, 3^n – 1, 3^n} which is presented in [8], is based on direct implementation of the CRT algorithm and requires *n*-digit ternary multipliers and a modulo (3^n – 2)(3^n – 1)(3^n) ternary adder for final reduction. So, as a result, the converter of [8] achieve long conversion delay and high hardware cost. But the larger modulo adder used in our converter is a modulo (3^n–1) adder and also the proposed design eliminates the use of multiplier. Therefore, our proposed residue to weighted converter has better performance than the residue to weighted converter of [8].

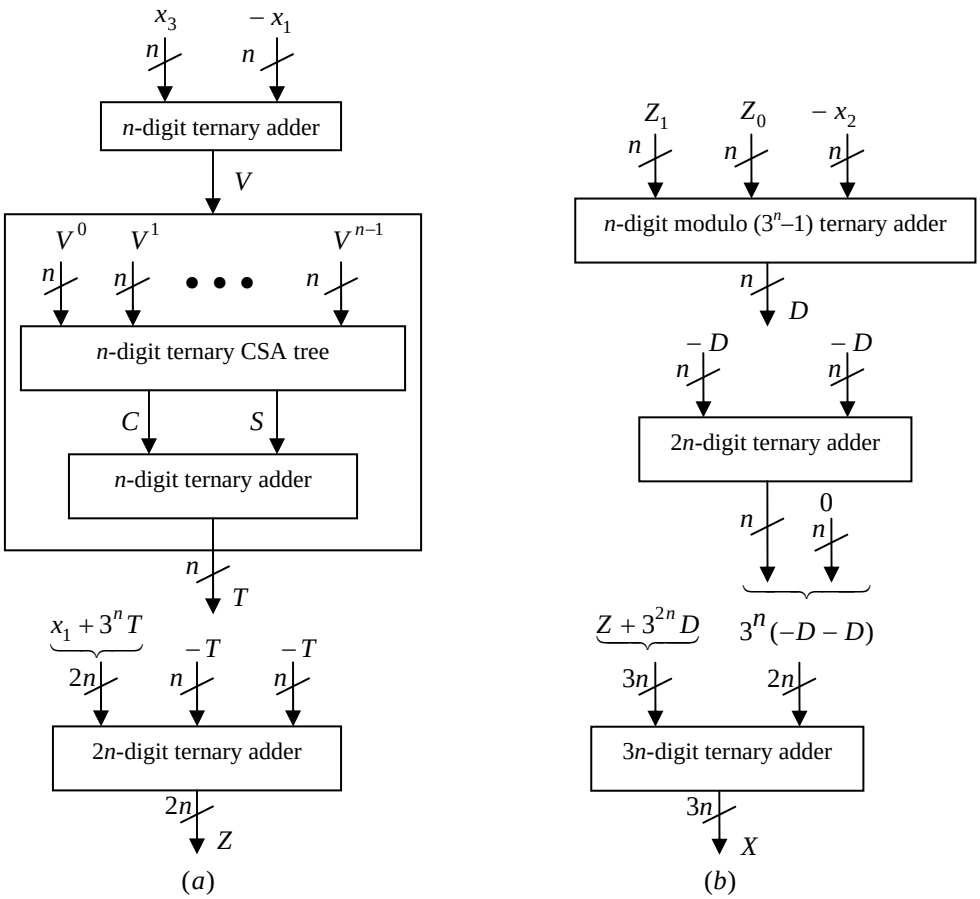

Fig. 4 Hardware architecture of the first level (a) and the second level (b) of the converter

5 Conclusion

In this paper an efficient design of the residue to weighted converter for the moduli set $\{3^n–2, 3^n–1, 3^n\}$ is presented. The proposed hardware implementation of the residue to weighted converter is multiplier-free and memoryless, which can be efficiently implemented in VLSI. In comparison with the last residue to weighted converter for the moduli set $\{3^n – 2, 3^n – 1, 3^n\}$, the proposed design has better performance.

References

[1] B. Parhami, Computer arithmetic: algorithms and hardware designs, Oxford University Press, 2000.

[2] Y. Wang, Residue-to-Binary Converters Based on New Chinese remainder theorems, IEEE Trans. Circuits Syst.-II, 47 (2000), 197-205.

[3] M. Hosseinzadeh, A. S. Molahosseini, K. Navi, A Fully Parallel Reverse Converter, International Journal of Electrical, Computer, and Systems Engineering, 1 (2007), 183-187.

[4] K.W. Current, V.G. Oklobdzija, D. Maksimovic, Low-energy logic circuit techniques for multiple valued logic, Proceedings of 26th International Symposium on Multiple-Valued Logic, 1996, 86-90.

[5] E. Dubrova, Multiple-Valued logic in VLSI: Challenges and opportunities, Proceedings of NORCHIP'99, Norway, 1999, 340-350.

[6] M. A. Soderstrand and R. A. Escott, VLSI implementation in multiple-valued logic of an FIR digital filter using residue number system arithmetic, IEEE Trans. Circuits Syst., 33 (1986), 5–25.

[7] M. Abdallah and A. Skavantzos, On MultiModuli Residue Number Systems With Moduli of Forms r^a, r^b-1, r^c+1, IEEE Transactions Circuits System I: Regular Paper, 52 (2005), 1253-1266.

[8] M. Hosseinzadeh and K. Navi, A New Moduli Set for Residue Number System in Ternary Valued Logic, Journal of Applied Sciences, 7 (2007), 3729-3735.

[9] W. K. Jenkins and B. J. Leon, The use of residue number systems in the design of finite impulse response digital filters, IEEE Transactions on Circuits and Systems, 24 (1977) 191–201.

[10] B. Cao, C. H. Chang and T. Srikanthan, Adder Based Residue to Binary Converters for a New Balanced 4-Moduli Set, Proceedings of 3rd International Symposium on Image and Signal Processing and Analysis, 2003, 820-825.

[11] M. Hosseinzadeh, K. Navi, S. Gorgin, A New Moduli Set for Residue Number System: $\{r^n\text{-}2, r^n\text{-}1, r^n\}$, IEEE International Conference on Electrical Engineering, 2007, 1-6.

[12] E. Kinvi-Boh, M. Aline, O. Sentieys, and E. D. Olson, MVL circuit design and characterization at the transistor level using SUS-LOC, in Proc. 33rd Int. Symp. Multiple-Valued Logic, 2003, 105–110.

[13] A. Hiasat and H. S. Abdel-Aty-Zohdy, Residue-to-binary arithmetic converter for the moduli set (2^k, 2^k-1, 2^{k-1}-1), IEEE Trans. Circuits Syst., 45 (1998), 204–208.

[14] Y. Wang, X. Song, M. Aboulhamid and H. Shen, Adder based residue to binary numbers converters for (2^n-1, 2^n, 2^n+1), IEEE Trans. Signal Processing, 50 (2002), 1772-1779.

[15] A. K. Jain, R. J. Bolton, and M. H. Abd-El-Barr, CMOS multiplevalued logic design—Part I: Circuit implementation, IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., 40 (1993) 503–514.

[16] S. Timarchi, K. Navi and M. Hosseinzadeh, New Design of RNS Subtractor for modulo $(2^n + 1)$, Proc. 2th IEEE International Conference on Information & Communication Technologies: From Theory To Applications, 2006.

[17] A. A. Hiasat, VLSI implementation of New Arithmetic Residue to Binary Decoders, IEEE Trans. VLSI Systems, 13 (2005), 153-158.

Received: December 3, 2007