ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Design and Implementation of Efficient Binary to Residue Converter Using Moduli Method

Vidhyalakshmi.M[1], Prof.Satyabama[2]

PG Scholar, Dept., of ECE, Government College of Technology, Coimbatore, Tamilnadu, India[1]

Professor & Head, Dept., of ECE, Government College of Technology, Coimbatore, Tamilnadu, India[2]

ABSTRACT: Recent analyses demonstrate that operations in some bases of Residue Number System (RNS) exhibit higher elasticity to process variations than in normal binary number system. Under this premise, moduli method offer greater flexibility in forming high cardinality balanced RNS. Limited in number theoretic property, converting an integer into residue for an arbitrary modulus is as difficult as complex arithmetic operation. This paper presents a new design of efficient residue generator and the design approach is demonstrated with large input word length of 32 bits for moduli of up to 5 bits. The proposed scheme is aimed to reduce the unwanted zero–zero additions which reduce number of computations thereby reducing power and enhance the performance. Our experimental results on moduli of different periodicities show that the proposed design is faster than the state-of-the-art residue generator.

KEYWORDS: Binary-to-residue conversion, residue number system, moduli, and periodicity

I. INTRODUCTION

RESIDUE NUMBER SYSTEM (RNS) emerges as an attractive solution for the implementation of low-power digital systems. Besides the frequently cited advantages over the conventional binary system, recent studies indicate that RNS computations also offer significant delay tolerance against process-induced parameter variations with properly selected bases [1]. It is found that larger modulo operation dominates circuit behavior and exhibits increased delay variations. From variation-tolerant perspective, RNS constructed by a large number of small and balanced moduli is preferable. This finding is timely and has paradigmatic impact on the digital integrated circuit design and yield for the continued scaling of transistor dimensions. Although moduli of the forms 2^n, 2^n+1 and 2^n-1 are modulo arithmetic friendly, it is very difficult to obtain more than five coprime integers of comparable wordlength in these forms. For the best of our knowledge, the highest reported cardinality (i.e.,number of modulus channels) of such balanced moduli set is five [3]. As the dynamic range increases, the wordlengths of some or all of these moduli will have to increase. Except for the simplicity of forward and reverse conversions in RNS, it may not pay off to have a large modulo operation of special modulus than multiple small generic modulo operations. The study in [4] shows that the area, delay and power costs contributed from the reverse conversion are cardinality insensitive once the cardinality exceeds certain threshold (usually between five to eight). Hence, it is better to increase the cardinality rather than enlarging the sizes of one or more moduli to extend the dynamic range of a RNS. Fortunately, a valid RNS can be formed with relative ease by selecting as many moduli as desired from plentiful small integers to fulfill the relative primality criterion and meet the dynamic range requirement. The significance of general moduli sets is also reflected in the continuous research into their efficient reverse conversion problem [5]–[7]. Hardware implementation of forward converter, also known as the residue generators, is not trivial for general moduli set. Unlike the reverse converter, as many residue generators as the cardinality of the moduli set are needed for each integer operand, which can become a performance bottleneck. In this paper, we investigate the existing formal design approaches to the binary-to-residue conversion problem for general moduli and identified their carry propagation addition and modular adder tree as the two main performance bottlenecks. To eliminate these problems, we have proposed a new approach to the design of highly efficient residue generators for any arbitrary moduli of up to five bits wide.

The rest of this paper is organized as follows. Section II introduces the preliminaries of RNS and Current art residue generators for an moduli are reviewed and analyzed. Our proposed architecture and its design procedures are

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

presented in Section III. Synthesis results are compared and analyzed in Section IV, and the conclusion is given in Section V.

II. REVIEW OF PREVIOUS WORKS

An RNS is defined by a base consisting of a set of co-prime integers$\{m_1,m_2,...,m_n\}$, where m_i is called a modulus and the greatest common divisor between any two moduli is one. The dynamic range of a RNS is given by M = $_i$. An L-bit integer X = $_j 2^j$ in weighted binary number system, where $x_j = \{0,1\}$ and $0 < X < M$, can be represented uniquely by an n–tuple in RNS, where the operation X mod m is widely known as the residue generation, binary-to-residue conversion or forward conversion.

Due to the cyclic periodicity of p, 2^i mod m , the integers 2^i and 2^{i+kp} for any integer k have the same residue modulo [8]. The residue corresponding to the jth bit of an n bit binary number with respect to modulus mi is generated and serially stored in a register.Two such registers storing residues of adjacent bits are combined using an processing element(PE). A total of n/2 PEs are used to generate the residues corresponding to each and every bit in the n-bit binary word. For a given binary number, depending on the value of bit bj either the register contents or zeros will be output. In general, the state-of-the-art high-speed implementation of residue generators for an moduli involves two stages of multi-operand additions. The first stage reduces theL -bit input to a p-bit word based on the cyclic periodicity of the modulus. The second stage reduces this p-bit word to the final residue by modular exponentiation. A binary CSA tree and one or more p-bit binary CPAs are needed in the first stage depending on the number of overflow carries generated from the CSAs. The second stage is usually implemented with a large number of logic gates and multiplexers, and a tree of modulo adders. One main problem of this approach is the speed, area and power consumption of the residue generator is strongly dependent on the periodicity, which varies irregularly from modulus to modulus.

III. PROPOSED RESIDUE GENERATORS

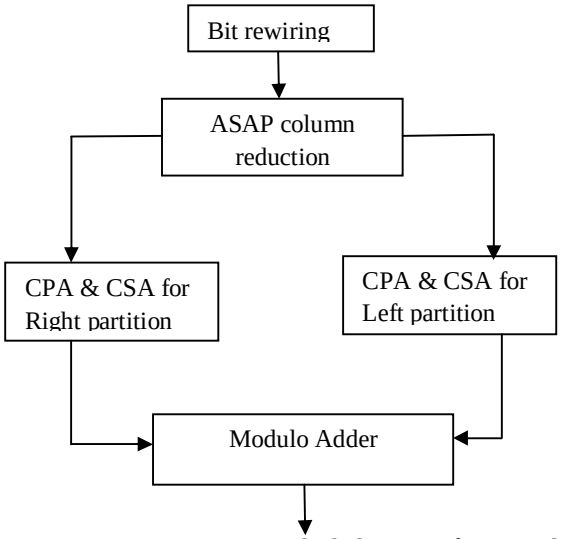

Figure1: Block diagram of proposed method

Our approach to the design of residue generators for arbitrary moduli is a great departure from all existing solutions. The main ideas that distinguish our proposed architecture are: 1) depth-bounded carry-save addition; 2) carry-free wordlength reduction of the sum and carry vectors; and 3) a single modified modulo adder. An overview of the essential building blocks is shown in Fig. 1.

To avoid the great delay disparity of CSA due to the differences in periodicity of different moduli, the residue is calculated using the distributive property in modular arithmetic [18] as follows:

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

$$|x|_m = \left| \sum_j 2^j \right|_m |_m$$

Where x_j is either 0 or 1.

As opposed to the input partitioning of conventional designs, the number of partial sums is L instead of L/p , which is independent of the modulus m. For convenience, we call each partial sum $|2^{j*}b_j|$mod m,a partial residue, where i=0,1,2,.....L-1 .Each partial residue is r-bit wide. In conventional approach, the array of partial sums is reduced to two binary vectors by a CSA tree followed by a CPA to produce a p-bit or longer word as input operand to the next stage.Thus, subsequent modular additions also become unwieldy due to its input dependency on p. To minimize the hardware complexity of the modulo reduction process, the wordlengths of the sum and carry vectors resulted from the CSA tree need to be minimized so as to reduce the wordlength of the CPA and hence the number of modulo adders needed later.A compact dot matrix diagram, where each dot denotes a binary variable, is formed by vacating all the "0" bits in the matrix of partial residue bits. The height of the dot matrix can be reduced by adding every three (two) dots in the same column by a FA (HA) in a carry save manner iteratively until it is one.

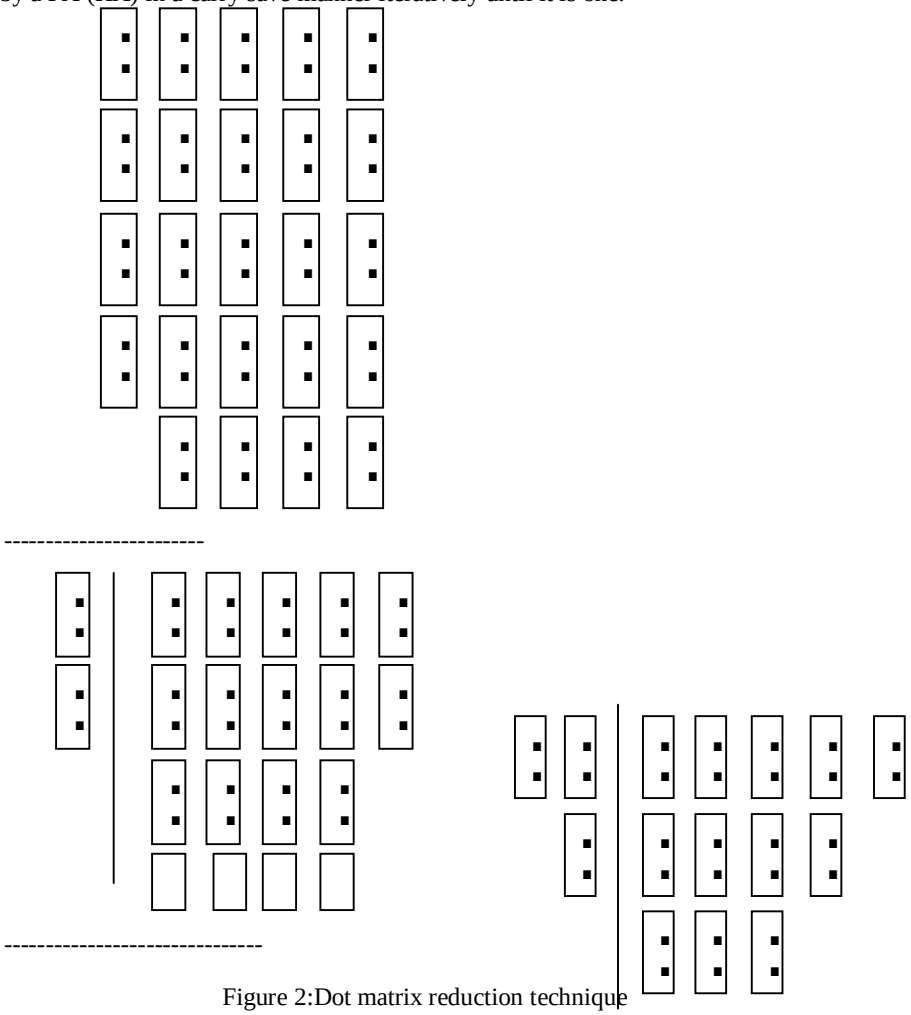

Figure 2:Dot matrix reduction technique

Modulo m Adder

The final result of is obtained by adding the two r-bit operands A and B modulo m.

The simple modulo addition be

$$|A + B|m = A + B \quad \text{if } A + B < m$$

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

A + B - m otherwise

This can be obtained using logic gates and multiplexers. Modulo 2^n addition of any two numbers X and Y, each of n bits, is done by adding the two numbers using a conventional adder. The result is an n+1 bit output, where the most significant bit is the carry-out. The residue is the first n lowest significant bits, and the final carry-out is neglected. Therefore, modulo addition 2^n is the most efficient modulo addition operation in the residue domain. 2^n-1 is a commonly used modulus in most special moduli-sets . Some architectures to implement the 2^n-1 modulo addition are available in the literature. Here, present the basic idea behind these algorithms and architectures. To understand the operation of modulo 2^n-1 addition of any two X and Y, where

0<= X,Y < m, we need to distinguish between three cases:

$0 <= X+Y < 2^n-1$

$X+Y = 2^n-1$

$2^n-1 < X+Y < 2^{n+1} -2$

In the first case, the result of the conventional addition is less than the upper limit 1 and no carry-out (Cout) is generated at the most significant bit. In this case, the modulo addition of X and Y is equivalent to the conventional addition. In the second case, the result is equal to 2^n- 1 (i.e. all 1's in binary representation). However, from RNS definition, the result has to be less than 2^n- 1 . In this case, the result should be zero. This case can be detected when all bits of the resulting number are ones . Correction is done simply in this case by adding a one and neglecting the carry-out. In the third case, the result of the conventional addition exceeds 2^n- 1 and a carry-out is generated at the most significant bit. This case is easily detected by the carry-out. Correction is done by ignoring the carry-out (equivalent to subtracting 2^n) and adding 1 to produce the correct result.

IV. SIMULATION RESULTS AND COMPARISION

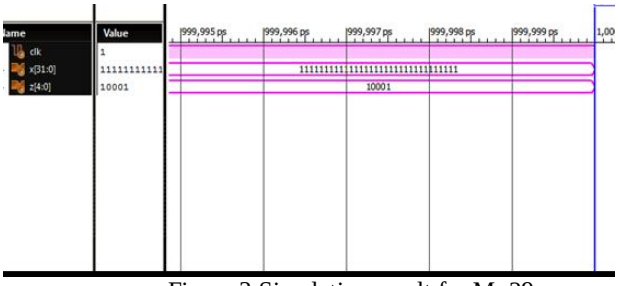

Figure 3:Simulation result for M=29

Parameter	[12]	Proposed
No of adders	256	125
Path Delay	17.7 ns	17.4 ns
Device Utilization	17%	LUT-1% Flipflop-42% IO-17%

V. CONCLUSION

The proposed scheme reduces the unwanted zero-zero addition ,thereby reducing the number of fulladders . The combinational path delay gets reduced and hence reduces power. This inturn enhances the efficiency .The proposed scheme is implemented for 32 bit binary input data.The technique is extended for 64 bit,128 bit and so on.

REFERENCES

[1] I. Kouretas and V. Paliouras, "Residue arithmetic for designing multiply-add units in the presence of non-gaussian variation," in Proc. IEEE Int. Symp. Circuits. Syst., Seoul, Korea, May 2012, pp.1231–1234.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

[2] I. Kouretas and V. Paliouras, "Residue arithmetic bases for reducing delay variation," in Proc. IEEE Int. Symp. Circuits Syst., Paris, France, Jun. 2010, pp. 3885–3888.

[3] B.Cao, C. H. Chang, andT. Srikanthan, "Aresidue-to-binary converter for a new 5-moduli set," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 54, no. 5, pp. 1041–1049, May 2007.

[4] M. Bhardwaj, T. Srikanthan, and C. T. Clark, "VLSI costs of arithmetic parallelism: A residue reverse conversion perspective," in Proc. 14th IEEE Symp. Computer Arithmetic, Adelaide, Australia, Apr. 1999, pp. 176–184.

[5] O.Maslennikow, N. Maslennikowa,M. Rajewska, D. Gretkowski, and J. Lienou, "Design of FPGA-based Residue Number Systemconverters for digital signal processing systems," in Proc. 9th Int. Conf. Experience Designing Appl. CAD Syst. Microelectron., Lviv, Ukraine, Feb. 2007, pp. 194–201.

[6] S. Bi, W. Wang, and A. Al-Khalili, "New modulo decomposed residue-to-binary algorithm for general moduli sets," in Proc. IEEE Int. Conf. Acoustics, Speech, Signal Process., Quebec City, QC, Canada, May 2004, pp. 141–144.

[7] G. Dimauro, S. Impedovo, R. Modugno, G. Pirlo, and R. Stefanelli, "Residue-to-binary conversion by the "quotient function"," IEEE Trans. Circuits Syst. II, Analog Digital Signal Process., vol. 50, no. 8, pp. 488–493, Aug. 2003.

[8] P. V. A. Mohan, "Novel design for binary to RNS converters," in Proc. IEEE Int. Symp. Circuits Syst., Bangalore, India, Jun. 1994, pp. 357–360.

[9] P. V. A. Mohan, "Efficient design of binary to RNS converters," J.Circuits, Syst. Comput., vol. 9, no. 3–4, pp. 145–154, Jun./Aug. 1999.

[10] G. Alia and E. Martinelli, "VLSI algorithm for direct and reverse conversion from weighted binary number to residue number system," IEEE Trans. Circuits Syst., vol. 31, no. 12, pp. 1033–1039, Dec. 1984.

[11] R. M. Capocelli and R. Giancarlo, "Efficient VLSI networks for converting an integer from binary system to Residue Number System and vice versa," IEEE Trans. Circuits Syst., vol. 35, no. 11, pp. 1425–1430,Nov. 1988.

[12] S. J. Piestrak, "Design of residue generators and multioperand modular adders using carry-save adders," IEEE Trans. Comput., vol. 43, pp. 68–77, Jan. 1994.

[13] T. Stouraitis, "Analogue- and binary-to-residue conversion schemes,"in IEE Proc.—Cir., Devices and Syst., Apr. 1994, pp. 135–139.

[14] A. B. Premkumar, "A formal framework for conversion from binary to residue numbers," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process., vol. 49, no. 2, pp. 135–144, Feb. 2002.

[15] A. B. Premkumar, E. L. Ang, and E.M.-K. Lai, "Improved memoryless RNS forward converter based on the periodicity of residues," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 53, no. 2, pp. 133–137, Feb.2006.

[16] W. J. Townsend, E. E. Swartzlander, Jr., and J. A. Abraham, "A comparison of dadda and wallace multiplier delays," in Proc. SPIE, Advanced Signal Process. Algor., Architectures, Implementations XIII, F. T. Luk, Ed., Dec. 2003, vol. 5205, pp. 552–560.

[17] A. A. Hiasat, "High-speed and reduced-area modular adder structures for RNS," IEEE Trans. Computers, vol. 51, no. 1, pp. 84–89, Jan. 2002.

[18] M. Dasygenis, K. Mitroglou, D. Soudris, and A. Thanailakis, "A full-adder-based methodology for the design of scaling operation in Residue Number System," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 55, no. 2, pp. 546–558, Mar. 2008.

[19] M. Bayoumi and G. Jullien, "A VLSI implementation of residue adders," IEEE Trans. Circuits Syst., vol. 34, no. 3, pp. 284–288, Mar.1987.

[20] M. Dugdale, "VLSI implementation of residue adders based on binary adders," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process., vol. 39, no. 5, pp. 325–329, May 1992.