DIO: 30.0485/ijearst.06.01.3478 https://ijearst.co.in/

ENGINEERING IN ADVANCED RESEARCH SCIENCE AND TECHNOLOGY

ISSN 2386-7876
Vol.01, Issue.03
Mar-2026
Pages: 762-772

VLSI IMPLEMENTATION OF FAST ADDITION USING QUATERNARY SIGNED DIGIT NUMBER SYSTEM

[1]P RENUKA, [2]GOGULAPATI SUDHEER [3]NANGEDLA KALYANI, [4]BADUGU JUHI SRAVANTHI, [5]KOMMARA SRAVANI

[1]Guide, Dept of ECE, ABR College of Engineering and Technology, Kanigiri, A.P.
[2,3,4,5]B. Tech, Dept of ECE, ABR College of Engineering and Technology, Kanigiri, A.P.

ABSTRACT: High performance adders are essential since the speed of the digital processor depends heavily on the speed of the adders used is the system. Also, it serves as a building block for synthesis of all other arithmetic operations. In present study, QSD number system eliminates carry propagation chain which reduces the computation time substantially, thus enhancing the speed of the machine. QSD Adder circuits are logic circuits designed to perform high-speed arithmetic operations. As the number of digits is large a carry free addition is desirable.by the exploitation of QSD number and QSD addition achieve the carry free addition. As an extension of this concept, Residue Number System is proposed for low power addition strategies. The proposed RNS components are not only results in fast arithmetic operation and it also highly reduced the hardware complexity since it requires fewer amount of logic elements. In this work, the proposed components are implemented in different moduli sets reverse converter designs and the performances are compared for different values of n.

Keywords: Quaternary Signed Digit Number, residue number system, Low power, Density, Latency.

INTRODUCTION: The Residue Number System plays a significant role in the battery based and portable devices because of its low power features and its competitive delay. The Residue number system reverse converter is designed with parallel prefix addition by using new components methodology for higher speed operation[1].The RNS consists of two main components forward and the reverse converter that are integrated with the existing digital system. The forward converter performs the operation of converting the binary number to the modulo number whereas the reverse converter performs the operation of reverse converting the modulo number to the binary number which is the hard and time consuming process compared with the forward converter. The fundamental RNS concepts such as 1)RNS definition with properties and their applications,2) consideration of modulo set selection,3) design of forward converter,4) modulo arithmetic units,4) design of reverse converter are discussed[2]. The voltage over scaling(VOS)

technique is applied to the residue number system to achieve high energy efficiency. The VOS technique introduces soft errors which degrades the performance of the system. To overcome these soft errors a new technique is implemented called joint RNS-RPR(JRR) which is the combination of RNS and the reduced precision redundancy. This method provides the advantage of satisfying the basic properties of RNS includes shorter critical path, reduced complexity and low power[3].New architectures are presented for the moduli sets(2n1,2n,2n+1) for the conversion from the residue to the binary equivalents[4].Here the speed and the cost are major concern. Distributed arithmetic principles are used to perform the inner product computation in[5].The input data which are in the residue domain which are encoded using the Thermometer code format and the ouputs are encoded using the One hot code format. Compared to the conventional method which used Binary code format, the proposed system which achieves higher operating speed. The residue number system which provides carry free addition and fully arithmetic operation[6],for several applications such as digital signal processing and cryptography[7]-[11]. This paper presents an investigation into a combination of two number representations; the residue number system (RNS) and the signed-digit (SD) number system. In RNS, an integer is decomposed into a set of residues with shorter binary representations, which can be processed in parallel. Carry propagation within RNS arithmetic circuits can be eliminated by using the SD number system to represent the residues. The SD number system provides a redundant number representation that facilitates carry-free addition. The use of SD numbers also implies efficient modulo arithmetic, which helps to simplify crucial RNS operations. The basis of a residue number system is a set of pairwise prime integers, called the moduli set. The performance of RNS processing depends on the choice of this set and on the implementation of forward and reverse RNS conversion. Many moduli sets and conversion techniques have been suggested for RNS systems with residues on 2's complement form. This work investigates moduli sets and converters for use with signeddigit residue number systems (RNS/SD). The aim is to investigate how the choice of the moduli set affects the performance of RNS/SD arithmetic operations. A number of moduli sets were selected for evaluation and forward and reverse RNS/ SD converters were implemented for each of these sets. In order to compare the performance of RNS/SD processing, RNS/SD finite impulse response (FIR) filters were implemented using Synopsys Design Compiler and a In the world of battery-based and portable devices, the residue number system (RNS) can play a significant role due to its low-power features and competitive delay.

LITERATURE SURVEY: Maskell (2007) formulated an algorithm for reducing the hardware complexity of linear phase finite impulse response digital filters that minimised the adder depth in the multiplier block adders (MBAs). The algorithm starts by aggressively reducing both the coefficient word length and the number of non-zero bits in the filter coefficients. It reduces the number of adders (the adder depth) needed

to construct the coefficient multiplier and results in an increased operating frequency. The results showed that this technique achieved a 67% reduction in the number of MBAs and 70% reduction in the number of multiplier block FAs, respectively. Vinod et al (2007) presented minimal-difference differential coefficients method for low power and high speed realisation of differential coefficients based finite impulse response filters. The minimal difference differential coefficients can be coded using fewer bits, which in turn reduce the number of full additions required for coefficient multiplication. By employing a differential coefficient partitioning algorithm and a pseudo floating point representation, the number of full adders and the net memory needed to implement the coefficient multipliers can be significantly reduced. Park et al (2000) developed an FIR filter that can be expressed as multiplication of a vector by scalars. The high speed implementations were presented for adaptive and non-adaptive filters based on a computation sharing multiplier which specifically targets computation reuse in vector scalar products. The performance of the implementation was compared with implementations based on carry save and Wallace tree multipliers in 0.6ȝ technology. The result showed that the sharing multiplier scheme gives improvement in speed by approximately 30% and 21% with respect to the Wallace tree multiplier based implementation for non-adaptive and adaptive filters, respectively. Badave & Bhalchandra (2012) pointed out the area complexity in the algorithm of finite impulse response (FIR) filter mainly caused by multipliers. Among the multiplier less techniques of FIR filter, Distributed Arithmetic was the most preferred area efficient technique. In this technique, precomputed values of inner product were stored in LUT, which are further added and shifted with a number of iterations equal to the precision of input samples. Design implementation and synthesis result showed the improvement in speed of operation as well as saving in area. Mirzaei et al (2010) mooted a method for implementation of high speed finite impulse response (FIR) filters on field programmable gate arrays (FPGAs).

EXISTING METHOD:

QUATERNARY SIGNED DIGIT ADDITION:

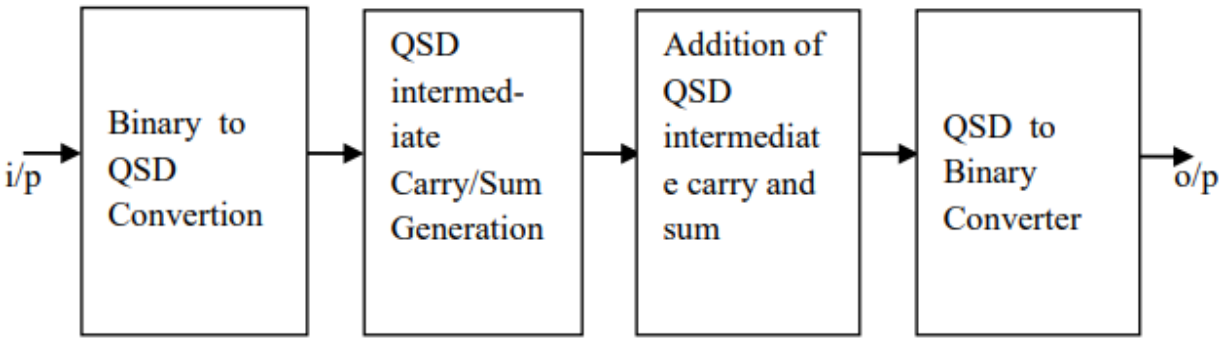


Fig1: Existing QSD addition

The speed of the digital processors depends mostly on the speed of the adders used in the system. Most of the arithmetic operations suffers from problems like limited number of bits, circuits complexity and delay. Carry look ahead adder produces a less propagation delay, but it is limited to small number of bits due to the circuit complexity. In this paper, a high speed QSD adder is proposed which is capable of performing carry free addition and borrow free subtraction using QSD numbers. For any operand size the QSD addition/subtraction operation employs a fixed number of min terms. In QSD number System carry propagation chains are eliminated which reduces the computation time. QSD is also advantages in case of parallelism and gate complexity. In QSD number system the carry propagation chains are eliminated which in tern reduces the circuit complexity and interconnections, thus the speed of the machine increases. As the QSD number system ranges from -3 to +3,the results produced by the addition if two QSD numbers varies from -6 to +6.Table 1 represents the outputs for all possible combinations of two QSD numbers. For the representation of decimal numbers in the range -3 to +3,one QSD digit is required. As the range of decimal number exceeds this range, more the one QSD digit is required. For the representation of addition result, which is in the range of -6 to +6,two QSD digits are required. Among the two digits in the QSD result, the MSB digit represents the carry bit and the LSB digit represents the sum bit. In order to prevent this carry bit to propagate from lower digit position to higher digit position QSD number representation is used. In the number representation, QSD numbers allow redundancy. In QSD representation the same decimal number can be represented in more than one form. To prevent further rippling of carry one of the QSD number is chosen among the available QSD numbers.

Sum	QSD represented number	QSD ceded number
-6	$2^{\bar{1}}2,1^{\bar{1}}2^{\bar{1}}$	$1^{\bar{1}}2^{\bar{1}}$
-5	$2^{\bar{1}}3,1^{\bar{1}}1^{\bar{1}}$	$1^{\bar{1}}1^{\bar{1}}$
-4	$1^{\bar{1}}0$	$1^{\bar{1}}0$
-3	$1^{\bar{1}}1,03^{\bar{1}}$	$1^{\bar{1}}1$
-2	$1^{\bar{1}}2,02^{\bar{1}}$	$02^{\bar{1}}$
-1	$1^{\bar{1}}3,01^{\bar{1}}$	$01^{\bar{1}}$
0	00	00
1	$01,13^{\bar{1}}$	01
2	$02,12^{\bar{1}}$	02
3	$03,11^{\bar{1}}$	$11^{\bar{1}}$
4	10	10
5	$11,23^{\bar{1}}$	11
6	$12,22^{\bar{1}}$	12

Table 1: Intermediate sum and carry between -6 to +6

By following the above mentioned rules the intermediate sum and carry from the step-1 have the range from -6 to +6.By utilizing the redundancy feature of QSD numbers we can choose the QSD represented numbers from the table-1 which satisfies the two rules mentioned above. The step-2 adds the intermediate sum of the current digit with the intermediate carry of the lower digit and produces the result which cannot be greater than 3 i.e, it will be in the range of -3 to +3.A single digit QSD number can be used to represent the result in this range i.e, -3 to +3.hence no further carry is required. The step-1 produces the output in the range of -6 to +6 which can be represented in intermediate sum and carry format as shown in the table-1.In the table-1,some numbers are having multiple QSD representations in column-2,but only those that meet the above mentioned two rules are chosen and are listed in the column-2 of the table-1 and are named as QSD coded numbers.

Quaternary Signed-Digit (QSD) number representation is a redundant number system with base 4. Unlike conventional quaternary numbers (0–3), QSD uses signed digits, which helps in reducing carry propagation during arithmetic operations. This makes QSD very useful in high-speed arithmetic circuits, VLSI design, and DSP architectures.

A redundant number system allows more than one representation for a given number. In QSD, redundancy arises because the digit set includes both positive and negative values. This redundancy enables local correction of sums without waiting for long carry chains.

PROPOSED TECHNIQUE:

RESIDUE NUMBER ADDITION:

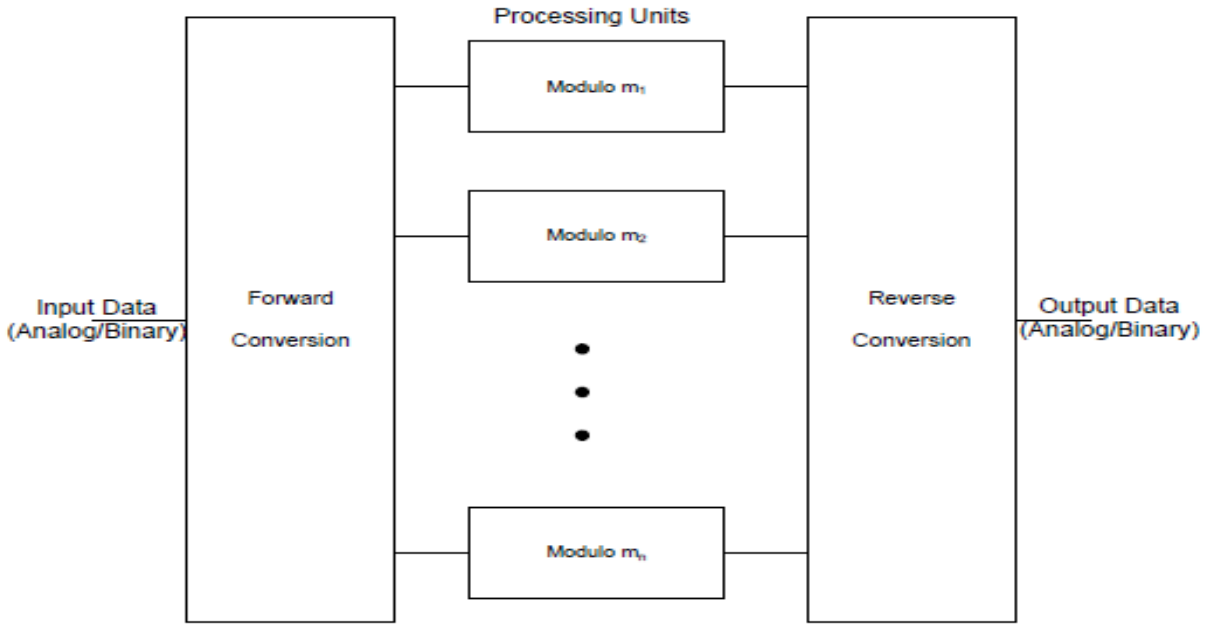


Fig:2 Proposed RNS addition block

RESIDUE NUMBER SYSTEM: RNS refers to the Residue Number System. It always deals with the remainders of the numbers. The remainders in this terminology are known as the "residues" and this type numbering system does not have any particular base. The base in this terminology is known as "modulo". As it has no particular weight it is known as "weight-free number system". It is based on "remainder theorem" of modular arithmetic.

(i)NEED FOR RNS:

It performs all the basic operations like addition, subtraction, multiplication and division. Despite of regular binary, octal, decimal and hexadecimal the operation of RNS is very simple.

(ii)WHY RNS?

Rather than other systems RNS is used for the high security purposes.

It provides an attractive feature called "carry-free arithmetic". It takes minimum time to perform any arithmetic operations.

(iii)IMPORTANT TERMS

MODULO SET: The collection of all these bases is known as "modulo set".

DYNAMIC RANGE: The largest integer in the modulo set until where the residues repeats again is defined as "dynamic range."

(iv)METHODS OF CONVERSION

There are two important methods to convert a binary number to residue number. They are:

(a) Chinese Remainder Theorem (CRT)

(b) Mixed Radix Conversion (MRC)

CONVERTING A NUMBER FROM DECIMAL TO RNS:

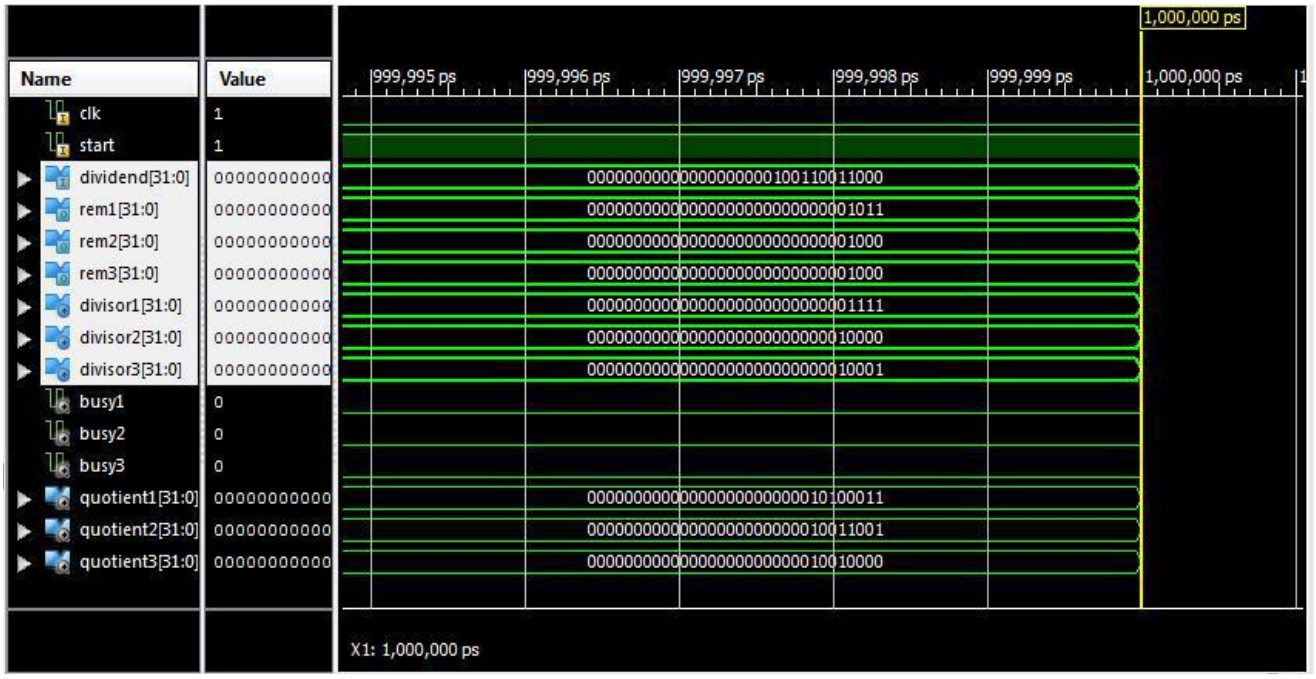
Let us consider a decimal 11 to be converted into the RNS with the modulo set {2, 3, 5}.The conversion is done in 3 steps.

Divide the decimal number 11 with the first modulo (or) base that is '2' such that we get a remainder as'1'

Now divide the same decimal number 11 with the second modulo that is '3' such that we get a remainder as '2'

Divide the decimal number 11 with the third modulo that is '5' such that we get a remainder as '1' Totally, the RNS representation for the decimal number 11 is given by {1, 2, 1}. In this way a decimal number can be converted into RNS.

RESULT:



Figa: Proposed simulation results

CONCLUSION: This project presents a method that can be applied to most of the current reverse converter architectures to enhance their performance and adjust the cost/performance to the application specifications. The use of modular and regular parallel-prefix adders proposed in this brief in reverse converters highly decrease the delay at the expense of significantly more power and circuit area, whereas the proposed prefix-based adder components allows one to achieve suitable tradeoffs between speed and cost by choosing the right adders for the parts of the circuits that can benefit from them the most. Performance and accuracy got increased by using RNS adder implementation.

FUTURE SCOPE: Based on our experience this far we see the task of reducing the number of multiplications for the base extension in the following perspective: (i) for a new modulus the new residue must be a function of all k residues in the initial base (and of course, of the new modulus); (ii) in the best case, a linear relation might exist and then at least k multiplications are required; (iii) for k new moduli at least k 2 multiplications and thus further reduction of the number of multiplications is unlikely. Special cases may exist for which one encounters a situation similar to fast transforms. Then more realistic approach would be to find such bases for which the multiplications could be shorter due to special form of the moduli (polynoms of powers of 2). - The background of the proposed work lies in the CRT, which enables breaking down the multiplication in a field of a very large number in multiplications into smaller fields. The main issue in this context is that division and other operations are

not simpler but more cumbersome. Then if one could find a way to construct another representation of numbers in finite fields with a better behavior when writing them as a product of fields, more efficient algorithms might be developed.

REFERENCES:

[1] N. Szabo and R. Tanaka, Residue Arithmetic and Its Applications to Computer Technology.

[2] G. Cardarilli, A. Nannarelli, and M. Re, "Residue number system for low-power DSP applications"

[3] J. Bajard and L. Imbert, "A full RNS implementation of RSA"

[4] S. Antão, J.-C. Bajard, and L. Sousa, "RNS based elliptic curve point multiplication for massive parallel architectures"

[5] F. E. P. D. Gallaher and P. Srinivasan, "The digit paralell method for fast RNS to weighted number system conversion for specific moduli $\{2n-1, 2n, 2n+1\}$"

[6] P. A. Mohan, "Reverse converters for the moduli sets $\{22N - 1, 2N, 22N + 1\}$ and $\{2N - 3, 2N + 1, 2N - 1, 2N + 3\}$"

[7] M.-H. Sheu, S.-H. Lin, C. Chen, and S.-W. Yang, "An efficient VLSI design for a residue to binary converter for general balance moduli $\{2n - 3, 2n + 1, 2n - 1, 2n + 3\}$" [8] R. Chaves and L. Sousa, "$\{2n + 1, 2n+k, 2n - 1\}$: A new RNS moduli set extension" [9] A. Premkumar and A. P. Vinod, "A memoryless reverse converter for the 4-moduli superset $\{2n - 1, 2n, 2n + 1, 2n+1 - 1\}$"

[10] B. Cao, C.-H. Chang, and T. Srikanthan, "An efficient reverse converter for the 4-moduli set $\{2n-1, 2n, 2n+1, 22n+1\}$ based on the new Chinese remainder theorem"

[11] H. Pettenghi, R. Chaves, and L. Sousa, "RNS reverse converters for moduli sets with dynamic ranges up to (8n+1)-bit"

[12] R. Zimmermann, "Efficient VLSI implementation of modulo $\{2n \pm 1\}$ addition and multiplication"

[13] P. Matutino, H. Pettenghi, R. Chaves, and L. Sousa, "RNS arithmetic units for modulo $\{2n \pm k\}$"

[14] S. Piestrak, "Design of residue generators and multi operand modular adders using carry-save adders"

[15] A. Premkumar, "A formal framework for conversion from binary to residue numbers"

[16] A. Premkumar, E. Ang, and E.-K. Lai, "Improved memoryless RNS forward converter based on the periodicity of residues"

[17] J. Low and C.-H. Chang, "A new approach to the design of efficient residue generators for arbitrary moduli"

[18] H. Pettenghi, L. Sousa, and J. Ambrose, "Efficient mplementation of multi-moduli architectures for binary-2-RNS conversion"

[19] D. Soudris, M. Dasigenis, S. Vasilopoulou, and A. Thanailakis, "A CAD tool for architecture level exploration and automatic generation of RNS converters"

[20] D. Soudris, M. Dasygenis, and A. Thanailakis, "VLSI methodology for the design of RNS and QRNS full adder based converters"

[21] P. M. Matutino, H. Pettenghi, R. Chaves, and L. Sousa, "Multiplierbased binary-2-RNS converter modulo $\{2n \pm k\}$

[22] P. M. Matutino, R. Chaves, and L. Sousa, "Theoretical analysis of modulo $\{2n \pm k\}$ units"

[23]"High performance standard cell library (UMC 0.13 μm)," Virtual Silicon Technology Inc., Palo Alto, CA, USA, Tech. Rep., 2004.