RNS Moduli Based Fast Sign Detection Algorithm for Logic Circuit

Vanshri Kamble[1], Prof. Suresh Gawande[2]
[1]*Research Scholar,* [2]*Research Guide*
Department of Electronics and Communication, Bhabha Bhopal

Abstract - In digital logic system the sign detection is an important operation to make calculations valid and faster processing of logic. The sign detection should be as fast as the logic calculations. The algorithm to make detection of sign has been synthesized on different synthesis tools.we have presented a brief survey about the most recently achievements in the RNS, for improving the system performance. We concern the different proposed moduli sets that provide different dynamic ranges, the common means and structures to perform forward and reverse conversion, universal structures of residue arithmetic units and application where using the RNS is beneficial. The detection algorithm developed for the sign detection purpose is evaluated based on the delay profile of algorithm. The delay of sign detection algorithm is achieved about 15.56ns.

keywords- sign detection, residue number system (RNS), moduli set.

I. INTRODUCTION

The principal aspect that distinguishes the RNS from other number systems is that the standard arithmetic operations; addition, subtraction and multiplication are easily implemented, whereas operations such as division, root, comparison, scaling and overflow and sign detection are much more difficult. Therefore, the RNS is extremely useful in applications that require a large number of addition and multiplication, and a minimum number of comparisons, divisions and scaling. In other words, the RNS is preferable in applications in which additions and multiplications are critical. Such applications are DSP, image processing, speech processing, cryptography and transforms [1], [2].

The main RNS advantage is the absence of carry propagation between digits, which results in high-speed arithmetic needed in embedded processors. Another important feature of RNS is the digits independence, so an error in a digit does not propagate to other digits, which results in no error propagation, hence providing fault-tolerance systems. In addition, the RNS can be very efficient in complex-number arithmetic, because it simplifies and reduces the number of multiplications needed. All these features increase the scientific tendency toward the RNS especially for DSP applications. However, the RNS is still not popular in general- purpose processors, due the aforementioned difficulties.

However the availability of inexpensive 16-bit and 64-bit digital signal processors for extensive work on picture encoding, speech encoding and other digital processing algorithms befogged the need of residue arithmetic based processors. However, special purpose processors have been developed for various applications.

A RNS based system generally operates on integer data. Figure: 1.1 shows a basic signal processor based on RNS.

A. Residue Number System

A residue number system (RNS) is a representation of number in which large integer number can be represented using a set of smaller integers, so that computation may be performed more efficiently. The RNS uses positional bases (called moduli) that are relatively prime to each other, e.g. 2, 3, 5. To convert a conventional weighted number (X) to residue system, simply take the residue of X with respect to each of positional moduli.

B. Special Moduli-Sets

The moduli sets with 2n are special moduli set $\{2^n - 1, 2^n, 2^n + 1\}$ is one of them. The special moduli are usually referred to as low cost moduli, since forward conversion and reverse conversion from their residue can be realized relatively easily and does not require complex operation such as multiplicative inverses, multiplication.

C. Chinese Remainder Theorem

Consider two positive numbers, x (the dividend) and y (the divisor). Then x modulo y (abbreviated as a (mod n)) can be thought of as the remainder, on division of x by y Now two simultaneous congruences $n = n\backslash(mod\, m\backslash)$ and $n = n_2(mod\, m_2)$ are only solvable when $n_1 = n_2\ (mod\ (gcd(m\backslash, m2)))$. The solution is unique when m1 and m2 are co-prime and their gcd is 1. The Chinese Remainder Theorem (CRT) may be stated as one of the most important fundamental results in the theory of RNS.

D. Sign Detection

For sign detection a trite technique was used. In this a sign bit was introduced as 64nd bit.

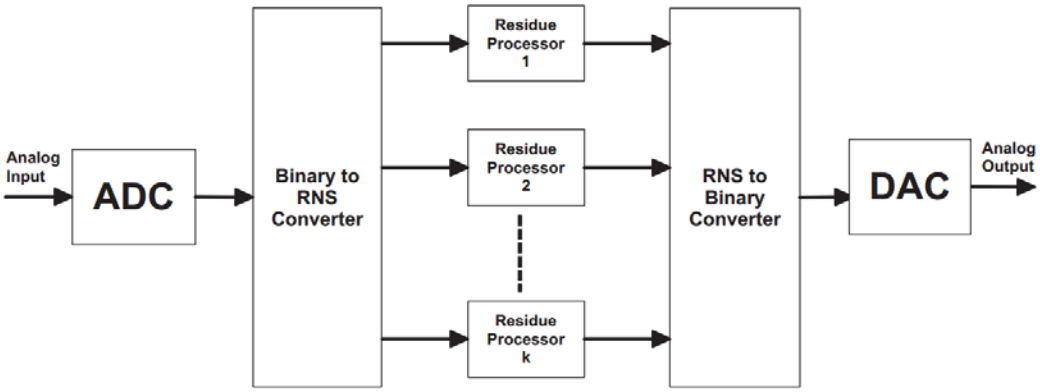


Figure: 1.1 Basic RSN Based Signal Processor.

This results in a signed bit representation of a number. Operating on floating numbers is a difficult task in residue arithmetic. Hence we convert the floating point number to an integer in a range $\pm Z$. The mapping is done such that the operating range $Z < R$. If the floating point number $\pm A$ is in range, and $A < Z$, then

$$P_{sn} = \frac{Z}{A} \qquad \text{……..(1)}$$

$$X = \frac{x}{P_{sn}} \qquad \text{…….(2)}$$

where P_{sn} is precision which decides the incorporation of digits after decimal points.

II. PROBLEM STATEMENT

The 'break and process' construct drives the researchers to work with residue arithmetic. This leads to 'carry-free' arithmetic operations. However operations viz. division, comparison, scaling etc. are very difficult since there is no carry propagation. But applications where these operations are not dominant and can be avoided and applications which are limned by integer arithmetic, RNS can be expended. Hereby, these facts are major stimuli to explore the potential of RNS in various fields of Communication and Signal Processing. Basics of residue arithmetic leads to complexity in magnitude calculation, sign detection and incorporation of fractional numbers. These constraints however made the implementation of ideas challenging and research worthy. Therefore the approach to digital FIR filter design was viewed from different perspective.

III. PROPOSED METHODOLOGY

Sign detection plays an essential role in branching operations, magnitude comparisons, and overflow detection. Because the sign information is concealed in each residue digit in a residue number system (RNS), sign detection in an RNS is more difficult than that in the weighted number system, in which the sign is the most significant bit (MSB). Furthermore, sign detection in an RNS is not as efficient as modular operations, such as addition, subtraction, and multiplication, because of its complexity.

A standard RNS is defined exclusively for positive integers in the range, M). To accommodate negative integers, an implicit signed number system may be considered to be split into a positive half of the range and a negative half of the range. The dynamic range M of the moduli set $\{m_1, m_2, \ldots, m_N - 1, m_N = 2^n - 1, 2 - 1, 2n\}$ is even. After conversion from the residue number to the weighted number, the resulting non integer X in the interval , M/2) carries an implicit representation of the sign of the actual result Y , which can be obtained in its range [$-M/2, M/2 - 1$)].

The two look-up tables are for the truncated part leftovers indicated in the QFS calculation. We put away the halfway remaking turn upward tables with the equipment channels in light of the fact that the qualities and table size depends singularly on the residue channel. As a configuration choice, we altered two of Phatak's suppositions: trading the per-equipment channel snake tree for a solitary viper in every channel; and supporting single-access stockpiling because of the equipment synthesizer's rationale deduction. This expands the execution time many-sided quality of the secluded duplication from O (lg N) to O (N). Thus, this builds the execution time unpredictability of the measured exponentiation from O (N lg N) to O (N2). We kept Phatak's presumption with respect to the decision of RP-RNS base. We settled on these choices to advance FPGA asset usage to oblige more equipment

channels.

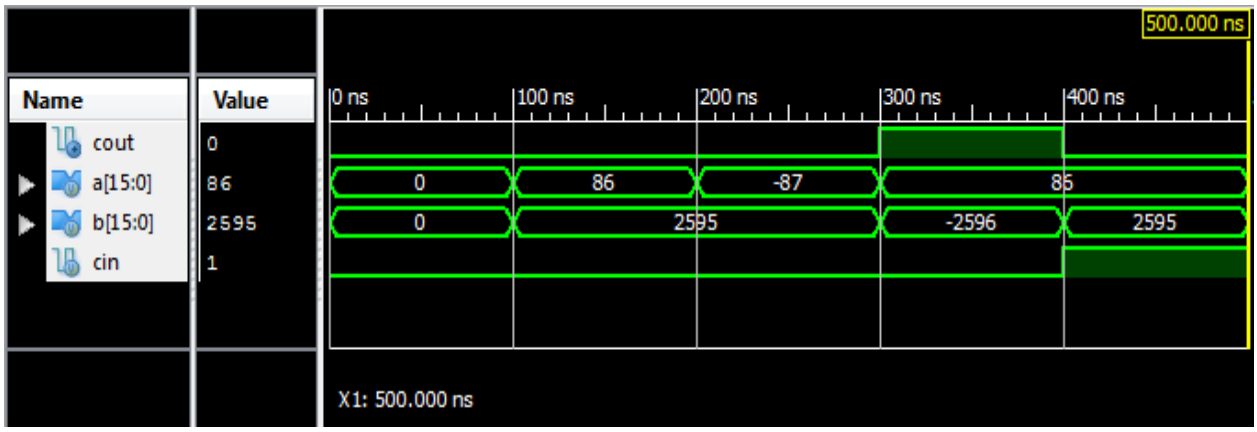


Figure 3.1 Timing waveform of carry generator.

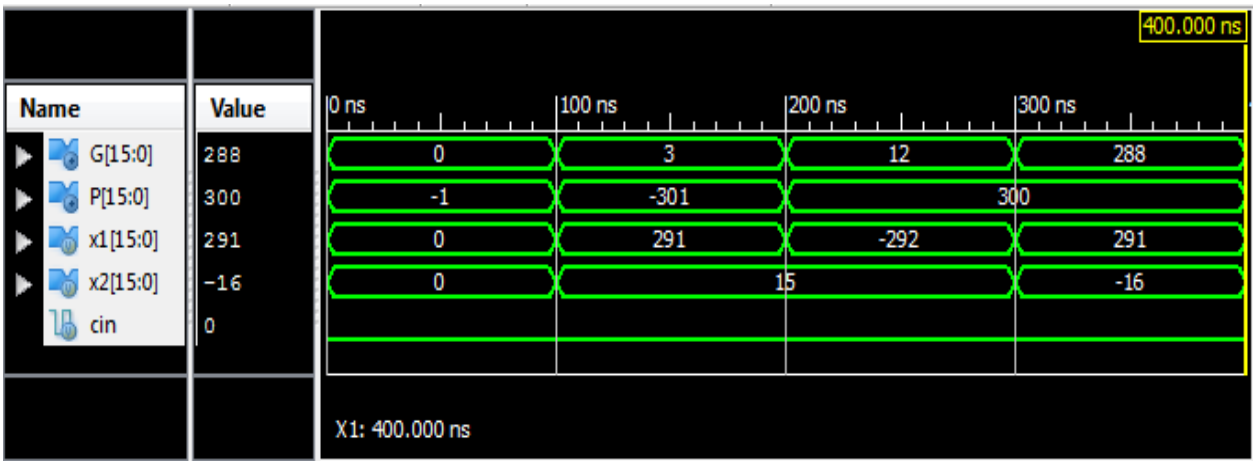


Figure 3.2 Timing waveform comparator.

The Test Bench Waveform Editor View is the graphical altering environment in which you can show and alter your Test Bench Waveform (TBW). You can make a test seat that incorporates data boost, and test seat length. The qualities for your data boost can be seen and altered as waveforms. The default Waveform Editor View foundation is dark. This is to recognize it from the reproduction results in the Simulation View, which has a default dark foundation. You can likewise change the Test Bench Waveform Editor View shading plan. A Test Bench Waveform (TBW) that you made in the Test Bench Waveform Editor can be revived for review and altering. When you make your TBW in the ISim, the record is naturally added to your outline venture.

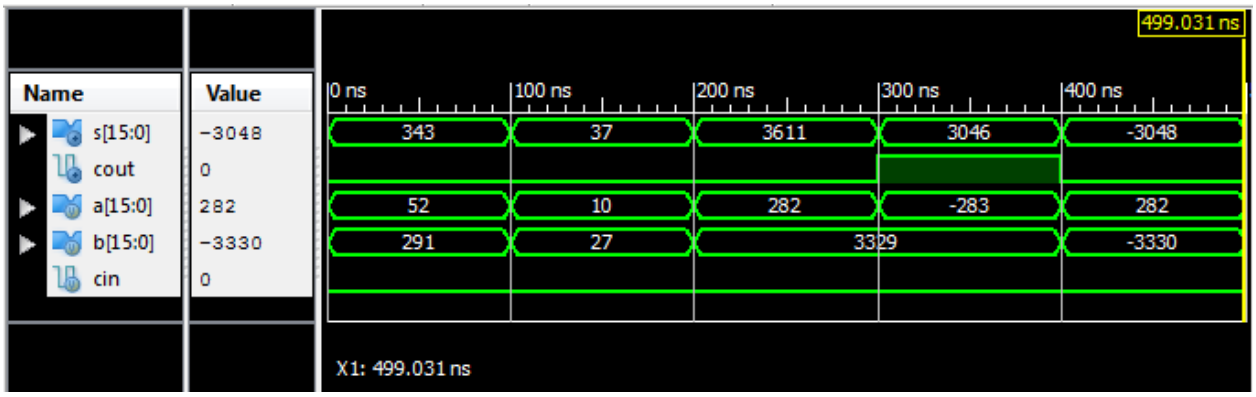


Figure 3.3 Timing Wawe Form of CSA.

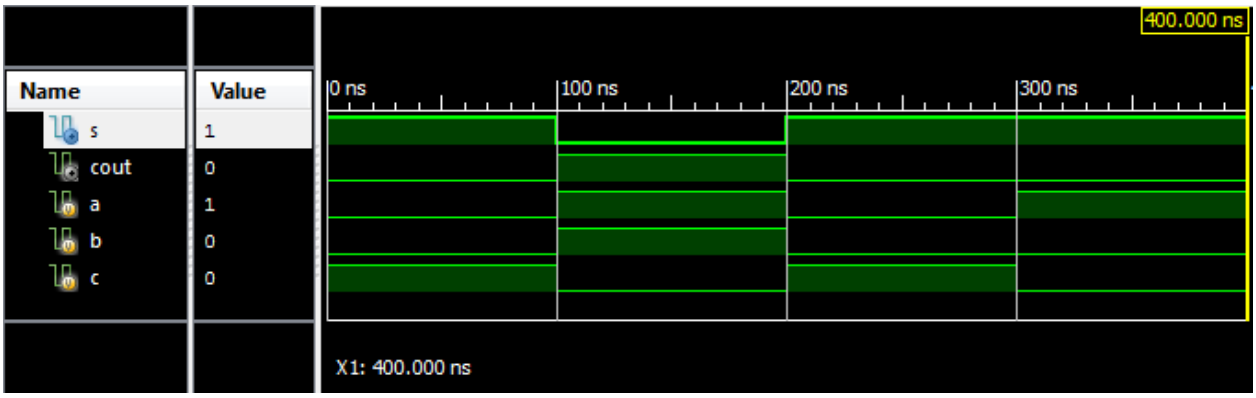


Figure: 3.4 Timing wawe Form of Full Adder.

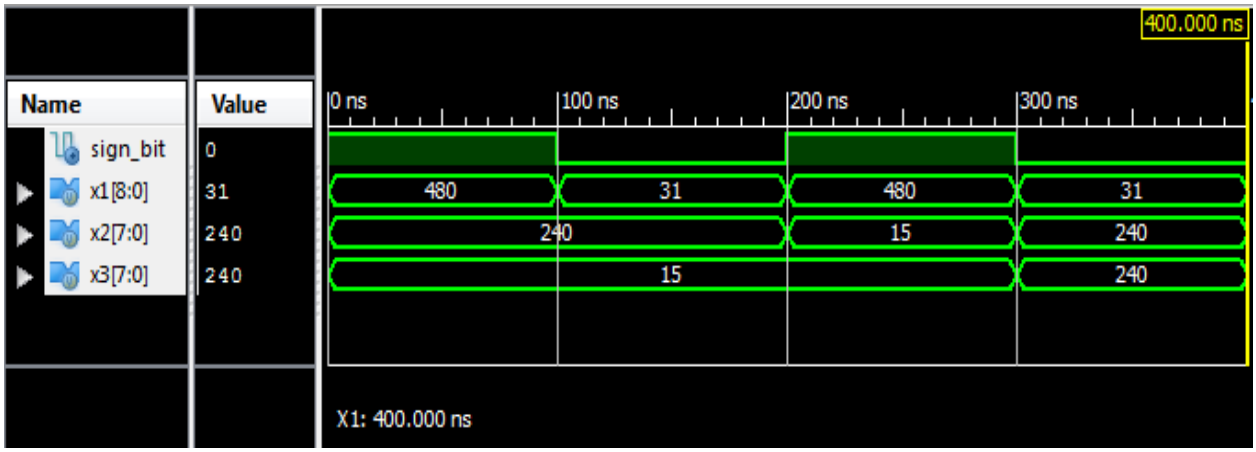
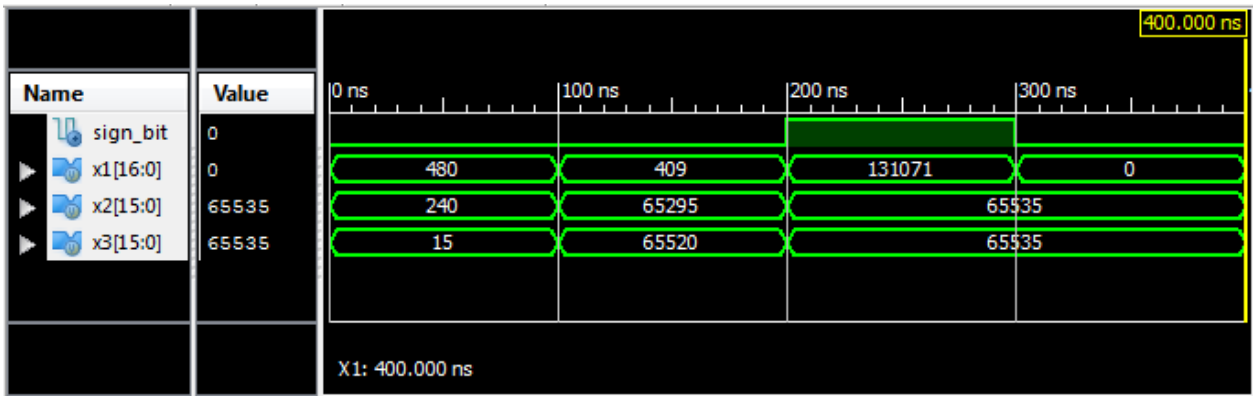


Figure 3.5 Sign Detection waweform of 8 bit.

The Test Bench Waveform Editor View is the graphical altering environment in which you can show and alter your Test Bench Waveform (TBW). You can make a test seat that incorporates data boost, and test seat length. The qualities for your data boost can be seen and altered as waveforms. The default Waveform Editor View foundation is dark. This is to recognize it from the reproduction results in the Simulation View, which has a default dark foundation. You can likewise change the Test Bench Waveform Editor View shading plan. A Test Bench Waveform (TBW) that you made in the Test Bench Waveform Editor can be revived for review and altering. When you make your TBW in the ISim, the record is naturally added to your outline venture.



Figurep:3.6 Sign Detection waweform of 16 bit.

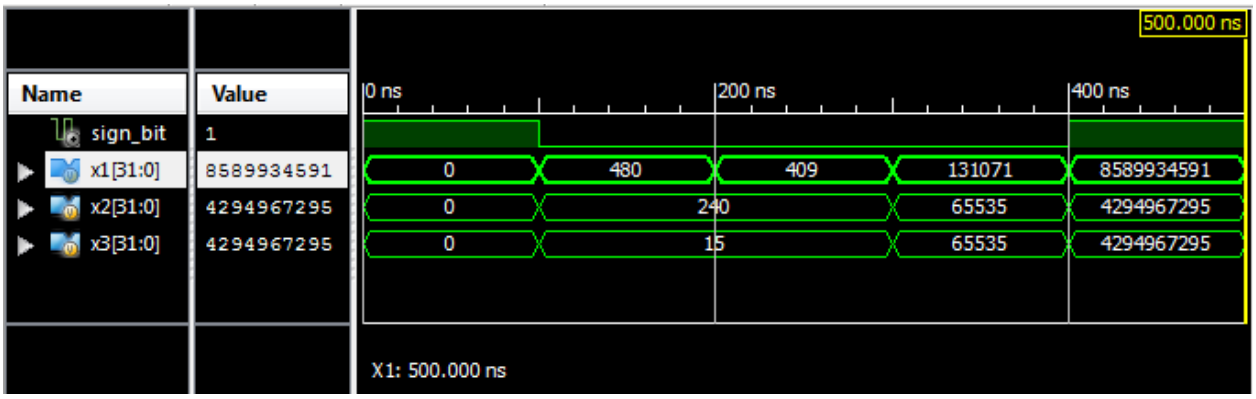


Figure 3.7 Sign Detection waweform of 32 bit.

A full programming recreation strategy is useful in two viewpoints. It offers full deceivability into the framework and grants the test seat or plan to be changed and re-affirmed in a quick manner. The challenges, then again, are getting an (a precise) entertainment model of the external memory module, and finishing a sensible reproduction speed.

The ISim gives a Test Bench Waveform Editor in which you can graphically characterize your test seats or test apparatuses. In the Test Bench Waveform (TBW), you can determine boost, and test seat lengths to check your outline with no learning of HDL or dialect scripting. The TBW you characterize can be added to your ISE venture. You can then utilize this TBW to drive your configuration re-enactment, in the same way you would utilize a HDL test bench.

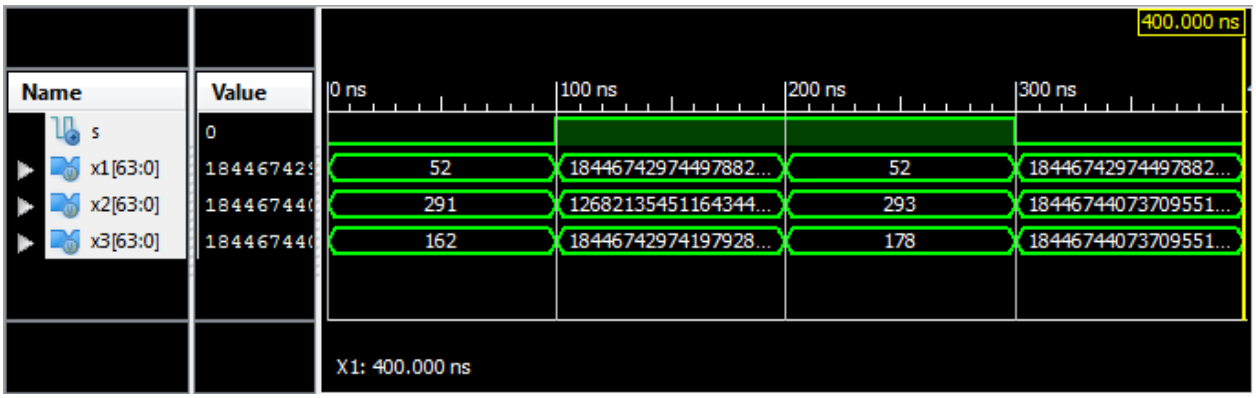


Figure 3.8 Sign ditection waweform of 64 bit.

The timing summary of the synthesis model is given in the below table 1 with unit gates and overall delay logic delay as well as route delay.

The proposed fast sign detection algorithm shown in this dissertation is synthesis on the XILINX FPGA board. The synthesis outcome in terms of delay of the proposed design is quite better than the existing work. The comparison is shown in the Table 5.2 below.

Table 3.1: Comparison Overall Delay and Unit Gate Delay.

Architecture	Overall Delay	Delay
Previous Design	19 ns	1 ns
Our Proposed Design	15.56 ns	0.196

		ns

from figure 3.1 to figure 3.10 timing waveform for carry generator, comparator , CSA, Full Adder, and sign detection waveform for 8 bit , 16 bit , 32 bit ,64 bit are demonstrated correspondingly and overall performance of gate delay are shown in comparison table 3.1

IV. CONCLUSION

The proposed algorithm for detection of sign is required in every digital logic circuit and must be detected as fast as possible simultaneously with the digital calculations. To make it faster in this research an logic algorithm is proposed and synthesized for delay calculations which is based on the residue number system with 3-factor moduli

set, such digital logic is the key to improve the speed detection. The synthesis outcome of the proposed architecture design is improved than the previous work. Future logic designs to make detection faster will be equipped with lower nano meter chip designs to speed up calculations; few changes in comparator logic design will also help to speed up the detection.

REFERENCES

[1] M. Xu, Z. Bian and R. Yao, "Fast Sign Detection Algorithm for the RNS Moduli Set $\{\{2^{n+1} - 1, 2^n - 1, 2^n\}\}$," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 23, no. 2, pp. 379-383, Feb. 2015.

[2] T. Tomczak, "Fast Sign Detection for RNS (2^{n}-1,2^{n},2^{n}+1) ," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 55, no. 6, pp. 1502-1511, July 2008.

[3] S. Bi and W. J. Gross, "The Mixed-Radix Chinese Remainder Theorem and Its Applications to Residue Comparison," in IEEE Transactions on Computers, vol. 57, no. 12, pp. 1624-1632, Dec. 2008.

[4] P. V. A. Mohan and A. B. Premkumar, "RNS-to-Binary Converters for Two Four-Moduli Sets $\{2^n-1,2^n,2^n+1,2^n+1-1\}$ and $\{2^n-1,2^n,2^n+1,2^n+1+1\}$," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 54, no. 6, pp. 1245-1254, June 2007.

[5] Thu Van Vu, "Efficient Implementations of the Chinese Remainder Theorem for Sign Detection and Residue Decoding," in IEEE Transactions on Computers, vol. C-34, no. 7, pp. 646-651, July 1985.

[6] Z. D. Ulman, "Sign Detection and Implicit-Explicit Conversion of Numbers in Residue Arithmetic," in IEEE Transactions on Computers, vol. C-32, no.

[7] N. Szabo, “Sign detection in nonredundant residue systems,” IRE Trans. Electro1”. Comput., vol. EC-11, no. 4, pp. 494-500, Aug. 1962.

[8] E. Al-Radadi and p. Siy, “RNS sign detector based on Chinese remainder theorem II (CRT II),” Comput. Math. Appl., vol. 46, nos. 10-11, pp. 1559-1570, 2003.

[9] M. Akkal and P. Siy, “Optimum RNS sign detection algorithm using MRC-II with special moduli set,” J. Syst. Arch., vol. 54, no. 10, pp. 911-918, Oct. 2008.

[10] S. Piestrak, “Design of residue generators and multioperand modular adders using carry-save adders,” IEEE Trans. Comput., vol. 43, no. 1, pp. 68-77, Jan. 1994.

[11] R. Zimmermann, “Efficient VLSI implementation of modulo (2n ± 1) addition and multiplication,” in Proc. 14th IEEE Symp. Comput. Arith¬metic, 1999, pp. 158-167.

[12] K. Furuya, “Design methodologies of comparators based on parallel hardware algorithms,” in Proc. 10th ISCIT, Oct. 2010, pp. 591-596.