Optimising the New Chinese Remainder Theorem 1 for the Moduli Set

$$\{2^{2n+2}+3, 2^{2n+1}+1, 2^{2n}+1,2\}$$

John Bosco Aristotle K. Ansuura
ICT Directorate
University for Development Studies-Ghana

Ismail Rashid Fadulilahi
Department of computer science
University for development Studies-Ghana

ABSTRACT
This paper seeks to improve the performance of the New Chinese Remainder Theorem (CRT) using the new moduli set $\{2^{2n+2}+3, 2^{2n+1}+1, 2^{2n}+1,2\}$. This optimization is very important in order to minimize the cost of hardware implementation and to improve the reverse conversion speed. The major factor responsible for this high hardware cost and high reverse conversion time is the presence of multipliers in the hardware implementation of the reverse converters. This paper proposes the moduli set $\{2^{2n+2}+3, 2^{2n+1}+1, 2^{2n}+1,2\}$, which is applicable for applications requiring larger dynamic range. The moduli set must be relatively prime integers. The computation of multiplicative inverses can be eliminated. We employ the proposed moduli set to optimize the New CRT-I. This scheme can result in less memory and adder based reverse converters, which is shown to be better than known existing similar state of the art scheme.

Keywords
Reverse Conversion, Optimization, Algorithm, Co-prime.

1. INTRODUCTION
Recent times have seen vigorous and continuous research into the improvement of computer performance. Researchers are making progress in improving the efficiency of computers with new ideas and technologies. Computing is the main task of a computer that is dealing with numbers all the time hence the number system. Some examples of number systems are binary number systems, decimal number systems, Residue Number System (RNS) and many more. Research has revealed that binary and decimal number systems intrinsically limit the performance of arithmetic units and processors built based on them. This is a limitation of the Weighted Number System (WNS) therefore making RNS more preferred in computing larger numbers in computers. A number in RNS is represented by the residues of all moduli, and arithmetic operations can be performed independently on each modulus. Thus, RNS offers the properties of parallelism, carry-free addition, borrow free subtraction, which are the major challenges of binary and decimal systems [10]. According to [9], the third-century Chinese scholar Sun Tzu invented the Residue Number System (RNS). Sun Tzu posed a mathematical riddle:

We have things of which we do not know the number
If we count them by threes, we have two left over
If we count them by fives, we have three left over
If we count them by sevens, we have two left over. Tzu [9] gave a rule, how many things are there?

This riddle was later generalized by another Chinese and known as the Chinese remainder theorem which is the bases of RNS [10].

In the 1950s, RNS was rediscovered by some computer scientists who sought to put them to use in the implementation of fast arithmetic and fault-tolerant computing [10]. Their system also offered useful properties for error detection, error correction and fault tolerance in digital systems. These properties increase the efficient in carrying out arithmetic operations. The speed of arithmetic operations relies largely on the size of the numbers involved, smaller numbers result in faster operations. Smaller numbers were considered in their research and therefore known for faster implementation of arithmetic operations. This system is applied in the fields of Digital Signal Processing (DSP), Speech Processing, Image Processing, Computer Engineering and Computer Security.

Sun Tzu [9] also proposed a general structure of a typical RNS processor as shown in Figure 1. Data that is represented in RNS is processed in parallel with no dependence or carry propagation between the processing units. The process of encoding the input data into RNS representation is called Forward Conversion. This data when processed is converted back to the conventional representation. Reverse conversion is the process of converting back the output data from RNS to conventional representation.

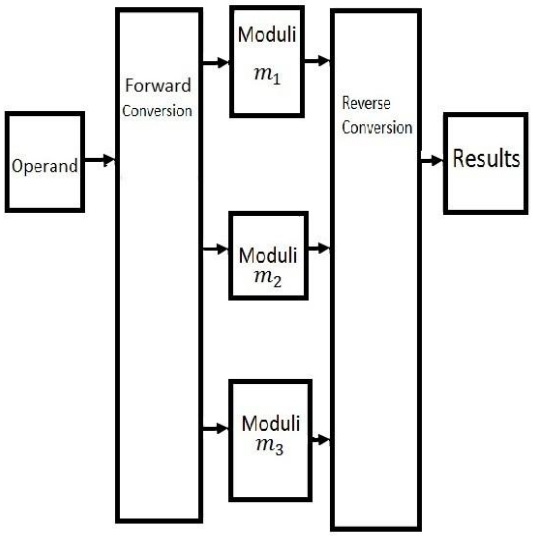

Figure 1: General Structure of an RNS-Based processor

2. SIGNIFICANCE OF RNS REPRESENTATION

Residue Number System is an integer system which is capable of supporting parallel, carry-free and high speed arithmetic operations [12], [3] and [5]. In addition, RNS offers some useful properties for error detection, error correction and fault tolerance in digital systems. It is very efficient in carrying out arithmetic operations like additions, subtractions and multiplications. The speed of the arithmetic operations relies on the size of the numbers involved.

Omondi A and Premkumar B [10], proposed an algorithm using parallel distributed arithmetic with no dependence between the arithmetic blocks which simplifies the overall design and reduces the complexity of the individual building blocks. [10] Summarized the advantages of RNS representation as follows:

High Speed: In conventional digital processors, the critical path is associated with the propagation of the carry signal to the last bit of the arithmetic unit. RNS encodes large words into small words minimizing the critical path. Also, the carry-free property of RNS between the arithmetic blocks results in high speed processing.

Reduced Power: RNS processors reduce the switching activities in each channel by using small arithmetic units resulting in a reduction of the dynamic power. This is because the dynamic power is directly proportional to switching activities.

Reduced Complexity: The property of RNS that makes it encode large numbers into smaller residues reduce the complexity of the arithmetic units in each modulo channel. This facilitates and simplifies the overall design.

3. DRAWBACKS OF RNS REPRESENTATION

[11] highlights the advantages of RNS architectures especially in areas of speed and power. These make it suitable to implement RNS in different applications. However, RNS processors are not widely use but remain as an interesting theoretical topic. There are two main reasons behind the limited use of RNS in applications;

1. Although RNS representation simplifies and expedites addition and multiplication compared to the conventional binary system, other operations such as division, square-root, sign detection, and comparison are difficult and costly operations in the residue domain. Thus, this makes it difficult to build an RNS based ALU capable of performing the basic arithmetic operations.
2. Conversion circuitry can be complex and can introduce latency that offsets the speed gained by the RNS processor. Hence, the design of efficient conversion circuits is considered the bottleneck of a successful RNS.

4. APPLICATIONS

RNS is suitable for applications in which addition and multiplication are the predominant arithmetic operations because of its carry-free property. RNS has good potential in applications where speed and/or power consumption is very critical. In addition, RNS facilitates error detection and correction due to the isolation between the modulo channels. Examples of these applications are digital signal processing (DSP), digital image processing, RSA algorithms, communication receivers, and fault tolerance. Intensive multiply-and-accumulate (MAC) operations are required in most of these applications.

Sun Tzu [9] proposed the design of digital filters which is an RNS application in DSP. These digital filters have different uses such as interpolation, decimation, equalization, noise reduction, and band splitting.

Later, [5] identified two basic types of digital filters: Finite Impulse Response (FIR) filters and Infinite Impulse Response (IIR) filters. These filters mostly use multiplication and addition operations. These two arithmetic operations in the residue domain increases system speed and lowers the power consumption.

RNS could also be applied in the field of cryptography to secure information [14]. The research implements an efficient algorithm of RNS for RSA cryptography which enhances security and also decrease delay time complexity for encoding and decoding. The area of hardware requirement for implementation was reduced.

Another possible application of RNS in DSP is the Discrete Fourier Transform (DFT), an engineering based application. In this application, multiplication and addition are equally the main operations of the DFT. Hence, faster operations due to the parallelism in the processing. In addition, the carry-free property of the RNS makes it potentially very useful in fault tolerant applications. Recent integrated circuits are very dense, and therefore full testing will no longer be possible. RNS has no weight information this implies an error in one of the residues does not affect the other modulo channels. Consequently, ordering is not important in RNS representation, therefore, faulty residues can be discarded and corrected separately.

In summary, RNS seems to be good for many applications that are important in modern computing algorithms.

5. CHOICE OF MODULI

The choice of the moduli set is the major consideration in the design of RNS systems this is because the efficient choice of the moduli guarantees the most efficient outcome possible from an RNS system in terms of speed, hardware etc. Consequently, for RNS the moduli $\{m_1, m_2, \dots, m_n\}$ should satisfy the following properties:

1. The moduli should be relatively prime i.e. no two moduli should have a greatest common divisor greater than 1.
2. The moduli should be as small as possible, so that the modulo operations require less computational time.
3. The moduli should be in such a form so as to offer simple forward (weighted to RNS) and reverse (RNS to weighted) conversion with simple residue arithmetic.
4. The product of the moduli should be large enough so as to offer the required dynamic range for the particular system.
5. The moduli should create a balanced decomposition of the dynamic range which means the difference between the number of bits of different moduli should be as small as possible for achieving optimal parallel performance.

6. CONVERSIONS AND THEORIES

6.1 Data Conversion

Data conversion is one of the greatest challenges of RNS because the input operands are provided in either standard binary or decimal format and must be converted to RNS before the computation can be performed. Similarly, the final results must be represented in the same way as the input operands, thus RNS to binary/decimal conversion is very essential to a successful RNS design. This implies that RNS based processors make heavy use of data conversions, which are slow processes. For an RNS processor to compete favorable with a conventional processor efficient data converters must be developed so that the RNS speedup will not be nullified by the conversion overhead. Data conversion can be divided into two Categories, namely, forward and reverse conversion. Relatively, the reverse conversion is more complex but the forward conversion is not simple either.

6.2 Forward Conversion

The input operands to the RNS processor are either in the decimal or binary format, and therefore need to be converted into their respective residues before they are used for the computation. This work of converting from decimal to binary to residue is done by the forward conversion.

6.3 Reverse conversion from RNS to binary representation

Reverse conversion algorithms in the literature are all based on either Chinese Remainder Theorem (CRT) or Mixed-Radix Conversion (MRC). The MRC is an inherently sequential approach. On the other hand, the CRT can be implemented in parallel. The main drawback of the CRT based Residue to Binary reverse converter, is the need of a large modulo adder in the last stage. The reverse conversion is one of the most difficult RNS operations and has been a major, if not the major, limiting factor to a wider use of RNS.

6.4 Mixed Radix Conversion (MRC)

According to [9], given a set of pair-wise relatively prime moduli $\{m_1, m_2, \dots, m_n\}$ and a residue representation

$\{x_1, x_2, \dots, x_n\}$ in that system of some number X, where $x_i = |X|_{m_i}$. The number X can be represented uniquely in mixed-radix form as $X = \{z_1, z_2, \dots, z_n\}$ where $X = z_1 + z_2 m_1 + z_3 m_2 m_1 + \cdots + z_n m_{n-1} z_{n-2} \dots m_1$ and $0 \le z_i \le x_i$

The Mixed-Radix Conversion (MRC) establishes an association between the unweighted, non-positional RNS and a weighted, positional mixed-radix system. In order to perform a reverse conversion the z_i values must be obtained. The z_i values are obtained as follows:

$$z_1 = x_1,\ z_2 = \left| |m_1{}^{-1}|_{m_2}(x_2 - z_1) \right|_{m_2}$$

$$z_3 = \left| |m_2{}^{-1}|_{m_3}\left(|m_1{}^{-1}|_{m_3}(x_3 - z_1) - z_2\right) \right|_{m_3}$$

$$\vdots$$

$$z_N = \left| |m_N{}^{-1}|_{m_N}\left(|m_{N-2}{}^{-1}|_{m_N}\left(\dots |m_2{}^{-1}|_{m_N}\left(|m_1{}^{-1}|_{m_N}(x_N - z_1) - z_2\right)\dots\right) - z_{N-1}\right) \right|_{m_N}$$

Example 1: Suppose we wish to find the number, X, whose residue representation is

{1, 0, 4, 0} relative to the moduli set {2, 3, 5, 7}

From the equations above,

$z_1 = 1,$

$z_2 = ||2^{-1}|_3(0-1)|_3 = |2 \times -1|_3$

$= |2 \times 2|_3$ (Additive inverse of -1 w.r.t 3 is 2)

$z_2 = 1$

$z_3 = ||3^{-1}|_5(|2^{-1}|_5(4-1)-1)|_5 = |2 \times (3 \times 3 \times -1)|_5$

$z_3 = 1$

$z_4 = ||5^{-1}|_7(|3^{-1}|_7(|2^{-1}|_7(0-1)-1)|_7 = |27|_7 = 6$

Therefore,

$X \cong (1,1,1,6)$ and for the conventional form, we translate this as $X = 6 \times 2 \times 3 \times 5 + 1 \times 2 \times 3 + 1 = 189$

6.5 Chinese Remainder Theorem (CRT)

The Chinese Remainder Theorem (CRT) may rightly be viewed as one of the most important fundamental results in the theory of residue number systems. CRT assures us that if the moduli of RNS are chosen appropriately then each number in the dynamic range will have a unique representation in RNS and that from such a representation we can determine the number represented. According to [9], [3] and [6], CRT is useful in reverse conversion as well as several other operations. Given a set of pair-wise relatively prime moduli, $m_1, m_2, m_3 \dots, m_n$, and a residue representation $(x_1, x_2, \dots, x_n)$ in that system of some number X.

That is $x_i = |x|_{m_i}$, that number and its residues are related by the equation $X = |\sum_{i=1}^{n} x_i\ |M^{-1}i|_{m_i} M_i|_M$

The expression above is the Chinese Remainder Theorem

Where $M_i = {}^{M}/_{m_i}$

Example 2: Consider the moduli set {3, 5, 7}, and suppose we wish to find the X whose residue representation is {1, 2, 3}.

Solution

$M = 3 \times 5 \times 7 \quad M_1 = {}^{M}/_{m_1} \quad M_1 = (3 \times 5 \times 7)/3$

$M_1 = 35 \quad M = 3 \times 5 \times 7$

$M_2 = {}^{M}/_{m_2} \quad M_2 = (3 \times 5 \times 7)/5 \quad M_2 = 21 \quad M_3 = {}^{M}/_{m_3}$

$M_3 = (3 \times 5 \times 7)/7 \qquad M_3 = 15$

Where

$\left| M_1\ M_1{}^{-1} \right|_3 = 1 \qquad \left| 35\ M_1{}^{-1} \right|_3 = 1 \qquad M_1{}^{-1} = 2$

$\left| M_2\ M_2{}^{-1} \right|_5 = 1 \qquad \left| 21\ M_2{}^{-1} \right|_5 = 1 \qquad M_2{}^{-1} = 1$

$\left| M_3\ M_3{}^{-1} \right|_7 = 1 \qquad \left| 15\ M_3{}^{-1} \right|_7 = 1 \qquad M_3{}^{-1} = 1$

Then by the CRT, we have (M = 3×5×7) = 105

$$X = |\sum_{i=1}^{3} x_i\ X_i|_{105}$$

$X = |1 \times 35 \times 2 + 2 \times 21 \times 1 + 3 \times 15 \times 1|_{105}$

$X = 52$

6.6 New Chinese Remainder Theorem I (CRT I)

According to [8], the speed of the arithmetic operations is based on the numbers involved in the operations. The size of the numbers is directly proportional to the delay of the

operations, and therefore smaller numbers imply faster operations. However, the Chinese Remainder Theorem requires a slow large modulo operation while the Mixed Radix Conversion requires finding the mixed radix digits which is a slow process. New Chinese Remainder Theorems were designed to make the computations faster and efficient without any overheads. New Chinese Remainder theorem I (CRT I) is a modified version of the traditional Chinese Remainder Theorem. In this conversion process, the weighted number can be retrieved faster because the operations are done in parallel, without depending on other results.

Some propositions proposed are necessary for this new conversion and are as follows,

Proposition 1:

$a = 1 mod(m_1\ m_2)$ implies $a = 1 mod m_1$ and $a = 1 mod\ m_2$

The above proposition is obtained from the corollary,

$a = 1 mod(m_1\ m_2 \ldots m_k)$ implies $a = 1 mod m_1, a = 1 mod m_2, \ldots, a = 1 mod m_k$

Proposition 2:

$[am_1] mod(m_1\ m_2) = [a] mod\ m_2 * m_1$

Proposition 3

For any y belonging to $[0, M-1]$, where $M = m_1 * m_2, \ldots m_{n-1} * m_n$, there is unique mixed radix representation as follows, where y_i satisfies the condition

$0 \le y_i \le m_{i+1}$

$y = y_0 + y_1 m_1 + y_2 m_1 m_2 + y_{n-1} m_1 m_2 \ldots . m_{n-1}$

Given the residue numbers $(x_1, x_2, x_3, \ldots x_n)$, the corresponding weighted number X can be computed using the following equation.

$X = [x_1 + k_1 m_1 (x_2 - x_1) + k_2 m_1 m_2\ (x_3 - x_2) \ldots + k_{n-1} m_1 m_2 \ldots m_{n-1} (x_n - x_{n-1})] mod m_1 m_2 \ldots m_{n-1} m_n$
Where $k_1 = (m_1)^{-1} mod m_2 \ldots m_n$, $k_2 = (m_1 m_2)^{-1}\ mod m_3 m_4 \ldots m_n$ and similarly

$k_{(n-1)} = (m_1 m_2 \ldots m_{n-1})^{-1} mod m_n$. It is to be noted that, the above equation is different from the traditional CRT and MRC equations. [8], Considered a four moduli set with the algorithm shown below: For a four moduli set, the equation is

$X = [x_1 + k_1 m_1 (x_2 - x_1) + k_2 m_1 m_2\ (x_3 - x_2) + k_3 m_1 m_2 m_3 (x_4 - x_3)] mod (m_1 m_2\ m_3 m_4)$

Where

$k_1 = (m_1)^{-1} mod (m_2 m_3 m_4)$

$k_2 = (m_1 m_2)^{-1} mod (m_3 m_4)$

$k_3 = (m_1 m_2 m_3)^{-1} mod (m_4)$

[1], in a paper stated that Amanda Mohan suggested the New Chinese Remainder Theorem introduced by Wang et al [13] can be derived from the constructive proof of the well-known Chinese Remainder Theorem (CRT)

According to, Wang et al [13] the New Chinese Remainder Theorem 1 (CRT 1) is a fast conversion algorithm substantially different from the CRT approach. According to [2], a new RNS output converters based on the CRT require the computation of a sum of products modulo a large number. The new converter presented in this paper uses the fractional representation for the output and eliminates the requirement for multiplications, thereby reducing area and delay. Further area improvements are possible by exploiting the period of terms to be added. An algorithmic approach is used to obtain full adder-based architectures that are optimized for area and delay.

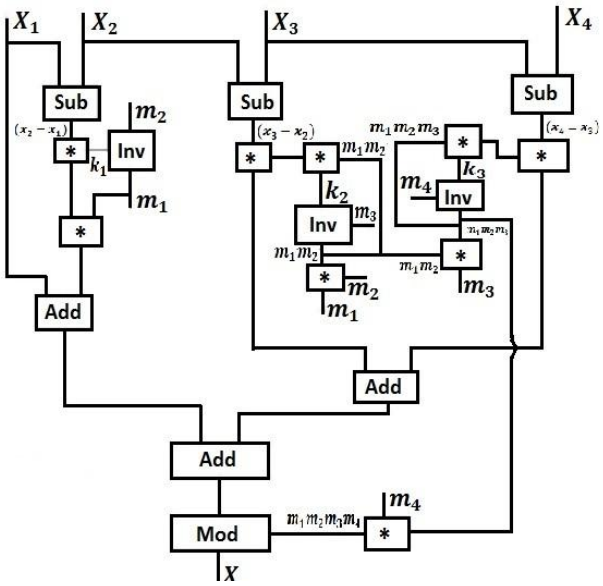

Figure 2: Hardware assembly for the calculation of X using CRT

7. ADDER

An adder or summer is a digital circuit that performs addition of numbers. In many computers and other kinds of processors, adders are used not only in the arithmetic logic unit(s), but also in other parts of the processor, where they are used to calculate addresses, table indices, and similar operations.

The figure 3 below is the hardware assembly of X without optimization.

8. OPTIMIZED HARDWARE IMPLEMENTATION OF NEW CHINESE REMAINDER THEOREM I

The new Chinese Remainder theorem one (CRT1) was optimized by removing inverse modulo operations which reduced the overhead of using big size summation terms in the equations. But, the equation carried multiplication terms which requires additional full and half adders, and this is a barrier for hardware implementation. This section further attempt to remove those multiplication terms which means the usage of additional full and half adders can be eliminated. The optimization is carried out by using carry save adders, which reduces the time delay in carrying out the multiplication operations.

Optimized Hardware Implementation of the Base Theorem for New Chinese Remainder Theorem one (CRT1)

Using the moduli set proposed, $M = \{2^{2n+2} + 3, 2^{2n+1} + 1, 2^{2n} + 1, 2\}$ where $n = 1, 2, 3, \ldots$ and the optimized equation from the previous section, we get

$X = [x_1 + {m_1}^2 (x_2 - x_1) + m_1 m_2\ (x_3 - x_2) + m_1 m_2 m_3 (x_4 - x_3)] mod (m_1 m_2\ m_3 m_4)$

Putting the above moduli set values in the equation, we get,
$X = x_1 + |Q_1 + Q_2 + Q_3|] mod (m_1 m_2\ m_3 m_4) \ldots \ldots . . (1)$

Where

$$Q_1 = (2^{4n+4} + 2^{2n+4} + 2^{2n+3} + 2^3 + 1) \times (x_2 - x_1)$$

$$Q_2 = (2^{4n+3} + 2^{2n+3} + 2^{2n+1} + 2^1 + 1) \times (x_3 - x_2)$$

$$Q_3 = [(2^{4n+3} + 2^{2n+3} + 2^{2n+1} + 2^1 + 1) \times (2^{2n} + 1)] \times (x_4 - x_3)$$

$$Q_3 = (2^{2n}a + a)(x_4 - x_3)$$

Where

$$a = 2^{4n+3} + 2^{2n+3} + 2^{2n+1} + 2^1 + 1$$

The hardware optimization using carry save adders reduces the number of steps needed for the implementation. This intends helps to reduce the area and delay for implementation.

The figures 3,4,5 and 6 below show the optimized implementation of equation using carry save adders and carry propagate adders.

Hardware implementation of Q_1, Q_2, Q_3 *and* X as shown in Figures 3, 4, 5 and 6 respectively.

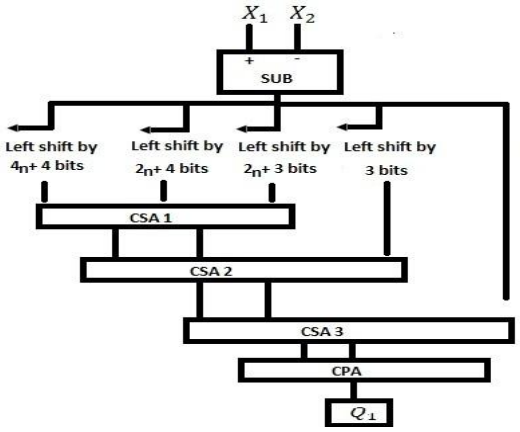

Figure 3: Hardware implementation of Q_1

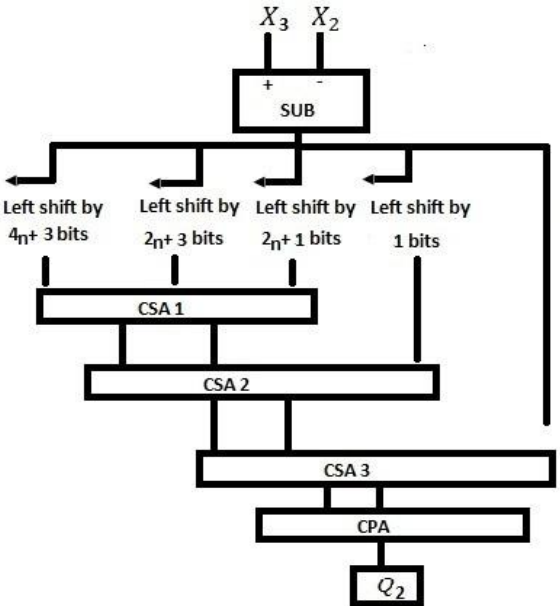

Figure 4: Hardware implementation of Q_2

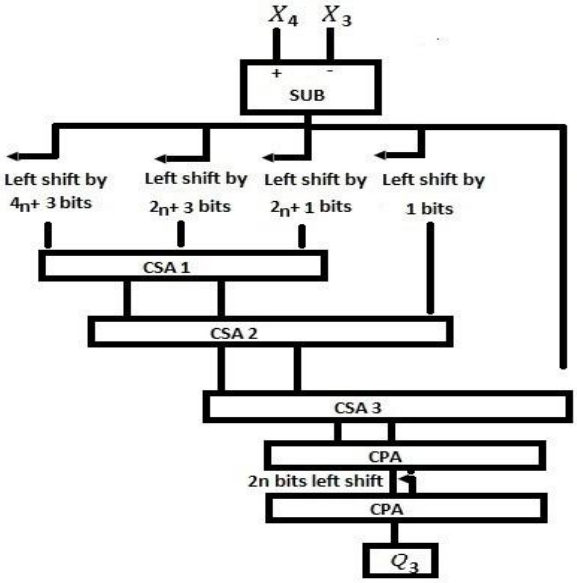

Figure 5: Hardware implementation of Q_3

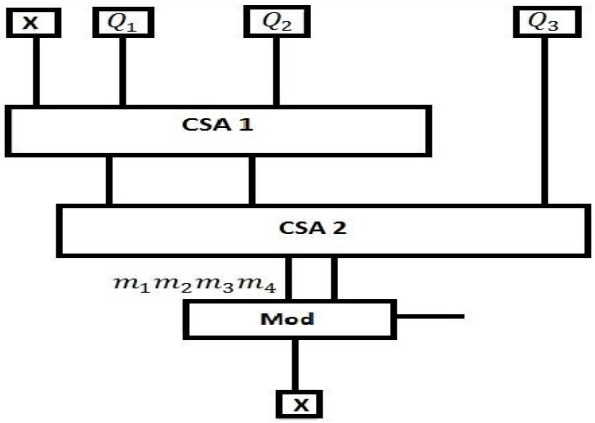

Figure 6: Hardware implementation of X

9. ANALYSES OF RESULTS

Area and Delay of Implementation

The tables below shows the area and delay of implementation of the optimized equation with the proposed moduli set $\{2^{2n+2} + 3, 2^{2n+1} + 1, 2^{2n} + 1, 2\}$.

Area and Delay of Q_1

In the implementation of Q_1 as shown in Fig 3, we have 3 carry save adders, 1 carry propagate adder and 1 sub-tractor. The proposed moduli set is a $6n + 5$ moduli set. The delay of a $CSA = 1$ and the area is $6n + 5$. The table below shows the calculations of the area and delay.

Table 1 Area and Delay of Q_1

	Area	Delay
$CSA = 3$	$18n + 15$	3
$CPA = 1$	$6n + 5$	$12n + 10$
Sub-tractor	$2n + 3$	$2n + 3$
Total	$26n + 23$	$14n + 16$

Area and Delay of Q_2

The implementation of Fig 4 shows that, we have 3 carry save adders, 1 carry propagate adder and 1 sub-tractor. This is shown below.

Table 2 Area and Delay of Q_2

	Area	Delay
$CSA = 3$	$18n + 15$	3
$CPA = 1$	$6n + 5$	$12n + 10$
Sub-tractor	$2n + 1$	$2n + 1$
Total	$26n + 21$	$14n + 14$

Area and Delay of Q_3

The implementation of Fig 5 shows that, we have 3 carry save adders, 2 carry propagate adder and 1 sub-tractor. This is shown below.

Table 3 Area and Delay of Q_3

	Area	Delay
$CSA = 3$	$18n + 15$	3
$CPA = 2$	$12n + 10$	$24n + 20$
Sub-tractor	$2n$	$2n$
Total	$32n + 25$	$26n + 23$

Area and Delay of X

The implementation of Fig 6 shows that, we have 2 carry save adders and 1 carry propagate adder. This is shown below.

Table 4 Area and Delay of X

	Area	**Delay**
$CSA = 2$	$12n + 10$	2
$CPA = 1$	$6n + 5$	$12n + 10$
Total	$18n + 15$	$12n + 12$

$$The\ total\ Area = \text{Area } Q_1 + \text{Area } Q_2 + \text{Area } Q_3 + \text{Area X}$$
$$= 102n + 84$$

For the total delay involved in the implementation, we add the delay of Q_3 to the delay of X. We take the delay of Q_3 because it has the largest delay.

$$Total\ Area = Q_3 + \text{Delay X} = 38n + 35$$

Comparison between the proposed moduli set $\{2^{2n+2} + 3, 2^{2n+1} + 1, 2^{2n} + 1,2\}$ and the previous one $\{2^{2n+2} + 3, 2^{2n+1} + 1, 2^{n} + 1,2\}$

Table 4.5 Comparison of Area and Delay

Moduli set	Area	Delay
$\{2^{2n+2} + 3, 2^{2n+1} + 1, 2^{n} + 1,2\}$ (Narayanaswamy,2010)	$54n + 91$	$43n + 71$
$\{2^{2n+2} + 3, 2^{2n+1} + 1, 2^{2n} + 1,2\}$ (proposed)	$102n + 84$	$38n + 35$

The table above shows the comparison between the two moduli set. The proposed one is a $6n + 5$ bits moduli set whiles the one being compared with is a $3n + 5$ bits moduli set. In terms of area, the proposed moduli set have a larger area than the previous one.

In terms of time delay, the proposed moduli set has a smaller delay. We can therefore conclude that, the proposed moduli set is more efficient in terms of time.

Illustration

Consider a weighted number $X = 100$ and the moduli set of the form $\{2^{2n+2} + 3, 2^{2n+1} + 1, 2^{2n} + 1,2\}$, and taking $n = 2$, we get the moduli set $m = \{m_1, m_2, m_3, m_4\} = \{67,33,17,2\}$

The dynamic range is $M = 75174$ RNS representation of X is shown below:

$X = (x_1, x_2, x_3, x_4) = (Xmodm_1, Xmodm_2, Xmodm_3, Xmodm_4)$

$= (33,11,15,0)$

$X_1 = 33, X_2 = 1, X_3 = 15, X_4 = 0$

From the above implementation,

$Q_1 = (2^{4n+2} + 2^{2n+4} + 2^{2n+3} + 2^3 + 1) * (x_2 - x_1)$

When $n = 2$

$Q_1 = (2^{12} + 2^8 + 2^7 + 2^3 + 1) * (1 - 33)$

$Q_1 = (2^{12} + 2^8 + 2^7 + 2^3 + 1) * (-33)$

$Q_1 = 4489 * 75142$

$Q_1 = 337312438$

$Q_2 = (2^{4n+3} + 2^{2n+3} + 2^{2n+1} + 2^1 + 1) * (x_3 - x_2)$

When $n = 2$

$Q_2 = (2^{11} + 2^7 + 2^5 + 2^1 + 1) * (15 - 1)$

$Q_2 = (2^{11} + 2^7 + 2^5 + 2^1 + 1) * (14)$

$Q_2 = 2211 * 14$

$Q_2 = 30954$

$Q_3 = (2^{4n+3} + 2^{2n+3} + 2^{2n+1} + 2^1 + 1) * (2^{2n} + 1) * (x_4 - x_3)$ When $n = 2$

$Q_3 = (2^{11} + 2^7 + 2^5 + 2^1 + 1) * (2^4 + 1) * (0 - 15)$

$Q_3 = (2^{11} + 2^7 + 2^5 + 2^1 + 1) * (2^4 + 1) * (-15)$

$Q_3 = 2211 * 17 * 75159$

$Q_3 = 2825001333$

$X_1 = 33$

$X = X_1 + Q_1 + Q_2 + Q_3$

$X = (33 + 337312438 + 30954 + 2825001333) mod 75174$

$X = 3162344758\ mod 75174 = 100$

10. CONCLUSION

In this paper, we address the problems identified with the New CRT 1 which includes the presence of an inverse modulo and multipliers which makes implementation difficult and expensive in terms of speed, area and cost. Our scheme

optimizes the New Chinese Remainder Theorem one (CRT 1) by eliminating the inverse modulo operators and also reduce the number of multipliers using four moduli set to implement the hardware optimization using carry save adders therefore reduces the delay. Our proposed scheme's implementation shows clearly that the proposed moduli set is better than the other ones stated in the work with regards to time.

The future direction of this work is to extend the number of moduli set to 5 or more to increase the dynamic range. Further efforts could be made to completely reduce the final modulo and this could improve the operations. Exploring other implementation methods could also be looked at.

11. REFERENCES

[1] Cao B., Chang C., Sirkanthan T (1998) A residue-to-binary converter for a new five-moduli set, IEEE Transactions on Circuits and Systems, 35 (11).

[2] Conway R. and Nelson J. (2004) Improved RNS FIR Filter Architectures, IEEE Transactions On Circuits and Systems II, Vol. 51, No. 1, pp. 26-28.

[3] Daabo M.I and Gbolagade K.A.Overflow Detection Scheme in RNS Multiplication Before Forward Conversion Journal of computing, Volume 4, Issue 12, pp. 13-16, December 2012 ISSN (Online) 2151-9617

[4] Daabo M.I and Gbolagade K.A. RNS Overflow Detection Scheme for the Moduli Set {M-1, M}.Journal of computing, Vol. 4, Issue 8 pp.39-44, August 2012 ISSN (Online) 2151-9617.

[5] Gbolagade K.A and Cotofana S.D. MRC Technique for RNS to Decimal Conversion for the moduli set {2n+2, 2n+1, 2n}. 16th Annual Workshop on Circuits, Systems and Signal Processing, pp.318-321, Veldhoven, The Netherlands, November 2008.

[6] Gbolagade K.A and Cotofana S.D.Residue-to-decimal converters for moduli set with common factors. 52nd IEEE International Midwest Symposium on Circuits and Systems (MINSCAS, 2009), PP.624-627, 2009.

[7] Narayanaswamy N. (2010). Optimization of New Chinese Remainder Theorems using special moduli sets.

[8] Narayanaswamy N., Skavantzos A., Stouraitis T. (2010). Optimal Modulus Sets for Efficient Residue-to-Binary Conversion Using the New Chinese Remainder Theorems. Accepted in 17th IEEE International Conference on Electronics, Circuits, and Systems, Athens.

[9] Number systems, IEEE Trans. on CAS -I, Vol. 41, No. 12, pp. 927-

[10] Omondi A and Premkumar B (2007).Residue Number System- Theory and Implementation (Volume 2)

[11] Szabo, N.S and Tanaka, R.I. (1967). Residue Arithmetic and Its Applications to Computer Technology. McGraw-Hill, New York.

[12] Umar A. F (2011) Data conversion in Residue Number System, Department of Electrical and Computer Engineering Mc Gill University Montreal.

[13] Wang, Y., Song, X., Aboulhamid, M., Shen, H. (2002). Adder based residue to binary number converters for (2n-1, 2n, 2n+1), IEEE Transactions on Signal Processing, vol.50, no.7, pp.1772-1779.

[14] I. R. Fadulilahi, E. K. Bankas, J. B. A. K. Ansuura (2015). Efficient Algorithm for RNS Implementation of RSA. International Journal of Computer Applications 11/2015; Vol.127, No.5, pp.14-19.