Volume 3, No. 1, January 2012
Journal of Global Research in Computer Science
ISSN-2229-371X
RESEARCH PAPER
Available Online at www.jgrcs.info

# A NOVEL HIGH DYNAMIC RANGE 5-MODULUS SET $\{2^{2n+1}, 2^{2n}+1, 2^n+1, 2^{n/2}+1, 2^{n/2}-1\}$ WHIT EFFICIENT REVERSE CONVERTER AND REVIEW IMPROVING MODULAR MULTIPLICATION'S DYNAMIC RANGE WITH THIS MODULUS SET

Ramin Aliabadian*[1], Mehdi Hosseinzadeh*[2], Mehdi Golsorkhtabaramiri [2]
[1]Department of Computer Engineering, Islamic Azad University Arak Branch, Arak, Iran
Ramin.ali@gmail.com
[2]Department of Computer Engineering, Science and Research branch, Islamic Azad University, Tehran, Iran
Hosseinzadeh@srbiau.ac.ir, M.golsorkhtabar@iaut.ac.ir

***Abstract:*** In last years, many modulus sets in the Residue Number System (RNS) for increasing Dynamic Range (DR) and parallelism are presented. Hence, for reach these purposes a new 5-Moduli Set $\{2^{2n+1}, 2^{2n}+1, 2^n+1, 2^{n/2}+1, 2^{n/2}-1\}$ for even *n*, and its efficient reverse converter design are introduced in this paper. This modulus set contains pair-wise relatively prime, and it offers the maximum feasible DR. New CRT-I and MRC are used to obtain a high-performance memory-less reverse converter for this modulus set. Also, we review improving the modular multiplication with this modulus set, and its dynamic range compared with other modulus sets.

## INTRODUCTION

The Residue Number System (RNS) is a valid method for the implementation of fast arithmetic and fault tolerant computing. The RNS with its carry-free operations, parallelism and enhanced fault tolerance properties has been used in computer arithmetic since the 1950s [1]. These characteristics make it very useful in some applications consisting of digital signal processing, fault tolerant systems, image processing systems and cryptography. Different modulus sets have been suggested for the RNS that have different properties with regards to the reverse conversion (Residue to Binary or R/B), Dynamic Range (DR) and arithmetic operations. The moduli of the forms $2^n$, $2^n-1$ and $2^n+1$ are very popular according to their easy arithmetic operations [2]. RNS has achieved more attention by researcher in recent years for its ability to do fast arithmetic operation like addition, subtraction and multiplication [3]. Modular multiplication is the most important aspect of public key cryptography algorithms like RSA and elliptic curve cryptography. RNS is the best way to speed up these applications because of its carry free nature. Efficiency of modular multiplication in RNS is depending on choosing of modulus set. The first step for designing an RNS system is the modulus set selection. The modulus set comprises of a set of pair-wise relatively prime integer numbers.

The dynamic range of an RNS system is defined in terms of the product of the module, and it denotes the interval of integers which can be represented in RNS uniquely. The proper selection of a modulus set has an important role in the design of the RNS systems because the speed of RNS arithmetic unit and the complexity of residue to binary converter is affected of the form and number of the modulus set [4]. Up to now, many modulus sets have been suggested for RNS which can be sorted based on their DR [2]. The prominent 3*n*-bit DR modulus sets are $\{2^n-1,\ 2^n, 2^n+1\}$ [5], $\{2^{n-1},\ 2^n-1, 2^n+1\}$ [6], $\{2^{n-1}-1,\ 2^n-1, 2^n\}$ [7] $\{2^n-1,\ 2^n-1, 2^{n+1}-1\}$ [8]. The DR provided by these modulus sets is not practical for applications which require larger DR with more parallelism. Accordingly, 4*n*-bit DR 4-moduli sets such as $\{2^n-1, 2^n, 2^n+1, 2^{n+1}+1\}$, $\{2^n-1, 2^n, 2^n+1, 2^{n+1}-1\}$ [9], $\{2^n-1, 2^n, 2^n+1, 2^{n-1}-1\}$ [10] and $\{2^n-3, 2^n-1, 2^n+1, 2^n+3\}$ [11] have been suggested to raise parallelism in RNS arithmetic unit. Then, 5*n*-bit DR modulus sets such as $\{2^n, 2^{2n}-1, 2^{2n}+1\}$ [12], $\{2^n-1, 2^n, 2^n+1, 2^{2n}+1\}$ [13], $\{2^n-1, 2^n, 2^n+1, 2^n-2^{(n+1)/2}+1, 2^n+2^{(n+1)/2}+1\}$ [14] have been introduced. Some of these modulus sets have defects, for example in modulus set $\{2^n-1, 2^n, 2^n+1, 2^n-2^{(n+1)/2}+1, 2^n+2^{(n+1)/2}+1\}$, Modulo $2^n+2^{(n+1)/2}+1$ is a low-performance modulo [2].

Lately, 6*n*-bit DR modulus sets such as $\{2^n-1, 2^n+1, 2^{2n}-2, 2^{2n+1}-3\}$ [15] and $\{22n, 22n+1-1, 2n+1, 2n-1\}$ [16] have been suggested and researchers tend to do research in these modulus sets. It should be noted that there is no special modular adder for modulo of the form $2^k-3$. Hence, the generic modular adders must be use for performing modulo $2^{2n+1}-3$ addition [2]. Accordingly, the use of modulo $2^k-3$ is cause increasing the delay and the cost of the reverse converter.

In this paper, an efficient memory-less reverse converter for a new RNS modulus set $\{2^{2n+1}, 2^{2n}+1, 2^n+1, 2^{n/2}+1, 2^{n/2}-1\}$ is suggested. The converter for the proposed 6*n*-bit DR modulus set is attained by an adder-based

implementation of the New CRT-I and MRC without the use of ROM or multiplier.

## BACKGROUND

***Residue Number System:*** The Residue Number System is defined in terms of a relatively-prime modulus set $\{P_1, P_2, P_3, \dots, P_n\}$ where $\gcd(P_i, P_j) = 1$ for $i \neq j$, and gcd(*a, b*) denotes the greatest common divisor of $a$ and *b*. A weighted number $X$ can be represented as $X = (x_1, x_2, x_3, \dots, x_n)$, where

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

This representation is unique for any integer $X$ in range $[0, M-1]$, where $M = P_1P_2P_3 \dots P_n$ is the DR of the modulus set $\{P_1, P_2, P_3, \dots, P_n\}$ [17].

***New Chinese Remainder Theorem 1:*** by New CRT-I [18] with the 4-moduli set $\{P_1, P_2, P_3, P_4\}$ the number $X$ can be computed from its corresponding residues $(x_1, x_2, x_3, x_4)$ using the following equations

$$X = x_1 + P_1 \left| \begin{matrix} k_1(x_2 - x_1) + k_2P_2(x_3 - x_2) \\ +k_3P_2P_3(x_4 - x_3) \end{matrix} \right|_{P_2P_3P_4} \quad (2)$$

Where

$$|k_1P_1|_{P_2P_3P_4} = 1 \quad (3)$$
$$|k_2P_1P_2|_{P_3P_4} = 1 \quad (4)$$
$$|k_3P_1P_2P_3|_{P_4} = 1 \quad (5)$$

The $k_1, k_2$ and $k_3$ in above equations are multiplicative inverses.

***Mixed-Radix Conversion:*** by MRC [19] the number $L$ can be computed as following

$$L = h_1 + h_2P_1 + h_3P_2P_1 + \cdots + h_n \prod_{i=1}^{n-1} P_i \quad (6)$$

The coefficients $h_i$s for tow module can be calculated by

$$h_1 = x_1 \quad (7)$$
$$h_2 = \left|(x_2 - x_1)\right| P_1^{-1}|_{P_2}\Big|_{P_2} \quad (8)$$

## REVERSE CONVERTER

The proposed reverse converter contains of two converters named part A and part B. In part A, New CRT-I is used to extract efficient reverse conversion algorithm for the modulus set $\{2^{2n+1}, 2^{2n}+1, 2^n+1, 2^n-1\}$. In part B, MRC is used for the modulus set $\{2^{n/2}+1, 2^{n/2}-1\}$ that is the subset of module $2^n - 1$. Also, adder-based hardware implementation of the conversion algorithms is illustrated.

*Designing part A:* The following theorems, propositions and properties are needed for extracting conversion algorithms. We ought to show that the module are pair-wise relatively prime for any natural number *n.*

In this paper we assumed $m_1 = 2^{2n+1}, m_2 = 2^{2n}+1, m_3 = 2^n+1, m_4 = 2^{n/2}+1, m_5 = 2^{n/2}-1$ and $m_6 = 2^n - 1$.

**Theorem 1.** The modulus set $\{2^{2n+1}, 2^{2n}+1, 2^n+1, 2^n-1\}$ contains pair-wise relatively prime numbers.

**Proof.** We must demonstrate that the numbers $2^n - 1$, $2^n + 1$, $2^{2n+1}$ and $2^{2n} + 1$ are relatively-prime. Hence, based on Euclid's Theorem, we have

$$\gcd(a, b) = \gcd(b, a \bmod b). \quad (9)$$

Therefore, we have

$$\gcd(2^{2n}+1, 2^n-1) = \gcd(2^n-1, 2) = 1 \quad (10)$$
$$\gcd(2^{2n}+1, 2^n+1) = \gcd(2^n+1, 2) = 1 \quad (11)$$
$$\gcd(2^{2n+1}, 2^{2n}+1) = \gcd(2^{2n}+1, -2) = 1 \quad (12)$$

So, our proposed modulus set can be used in the RNS, and we can now deal with its reverse converter.

**Lemma 1.** The multiplicative inverse of $2^{2n+1}$ modulo $(2^{4n}-1)$ is $k_1 = 2^{2n-1}$.

**Proof.** By letting the values $k_1 = 2^{2n-1}$, $P_1 = m_1 = 2^{2n+1}$, $P_2 = m_2 = 2^{2n}+1$, $P_3 = m_3 = 2^n + 1$ and $P_4 = m_6 = 2^n - 1$ in (3), we have

$$\begin{aligned} &|k_1 \times 2^{2n+1}|_{2^{4n}-1} \\ &= |2^{2n-1} \times 2^{2n+1}|_{2^{4n}-1} \\ &= |2^{4n}|_{2^{4n}-1} = 1. \end{aligned} \quad (13)$$

**Lemma 2.** The multiplicative inverse of $2^{2n+1} \times (2^{2n}+1)$ modulo $(2^{2n}-1)$ is $k_2 = 2^{2n-2}$.

**Proof.** By considering (4) it is obvious that

$$\begin{aligned} &|k_2 \times 2^{2n+1} \times (2^{2n}+1)|_{2^{2n}-1} \\ &= |2^{2n-2} \times 2^{2n+1} \times (2^{2n}+1)|_{2^{2n}-1} \\ &= |2^{-2} \times 2 \times 2|_{2^{2n}-1} = 1 \\ &= |1|_{2^{2n}-1} = 1. \end{aligned} \quad (14)$$

**Lemma 3.** The multiplicative inverse of $2^{2n+1} \times (2^{2n}+1) \times (2^n+1)$ modulo $(2^n-1)$ is $k_3 = 2^{n-3}$.

**Proof.** By letting $k_3 = 2^{n-3}$, $P_1 = m_1$, $P_2 = m_2$, $P_3 = m_3$ and $P_4 = m_6$ in (5), then we have

$$\begin{aligned} &|k_3 \times 2^{2n+1} \times (2^{2n}+1) \times (2^n+1)|_{2^n-1} \\ &= |2^{n-3} \times 2^{2n+1} \times (2^{2n}+1) \times (2^n+1)|_{2^n-1} \\ &= |2^{-3} \times 2 \times 2 \times 2|_{2^n-1} = 1 \\ &= |1|_{2^n-1} = 1. \end{aligned} \quad (15)$$

**Theorem 2.** In the RNS relying on the 4-moduli set $\{m_1, m_2, m_3, m_6\} = \{2^{2n+1}, 2^{2n}+1, 2^n+1, 2^n-1\}$, the number $X$ can be computed from its corresponding residues $(x_1, x_2, x_3, x_6)$ by

$$X = x_1 + 2^{2n+1} \left| \begin{matrix} 2^{2n-1} \times (x_2 - x_1) \\ +2^{2n-2} \times (2^{2n}+1) \times (x_3 - x_2) \\ +2^{n-3} \times (2^{2n}+1) \times (2^n+1) \\ \times (x_6 - x_3) \end{matrix} \right|_{2^{4n}-1} \quad (16)$$

**Proof.** By letting $P_1 = m_1$, $P_2 = m_2$, $P_3 = m_3$ and $P_4 = m_6$, and the values of $k_1, k_2, k_3$ from Lemmas 1, 2 and 3 into (2), equations (16) is procured.

**Property 1.** The residue of a negative residue number $(-v)$ in modulo $(2^n - 1)$ is the one's complement of $v$, where $0 \leq v < 2^n - 1$ [20].

**Property 2.** The multiplication of a residue number $v$ by $2^P$ in modulo $(2^n - 1)$ is preformed by *P* bit circular left shift, where *P* is a natural number [20].

Regard the 4-moduli set $\{2^{2n+1}, 2^{2n}+1, 2^n+1, 2^n-1\}$ and let the corresponding residues of the integer number $X$ be $(x_1, x_2, x_3, x_6)$. The residues have bit-level representation as:

$$x_1 = (\underbrace{x_{1,2n}\, x_{1,2n-1} \ \dots\ x_{1,1}\, x_{1,0}}_{2n+1 \text{ bits}})_2 \quad (17)$$

$$x_2 = (\underbrace{x_{2,2n}\, x_{2,2n-1} \ \dots\ x_{2,1}\, x_{2,0}}_{2n+1 \text{ bits}})_2 \quad (18)$$

$$x_3 = (\underbrace{x_{3,n}\, x_{3,n-1} \ \dots\ x_{3,1}\, x_{3,0}}_{n+1 \text{ bits}})_2 \quad (19)$$

$$x_6 = (\underbrace{x_{6,n-1}\, x_{6,n-2} \ \dots\ x_{6,1}\, x_{6,0}}_{n \text{ bits}})_2 \quad (20)$$

We simplify equation (16) as follows:

$$x_1 + 2^{2n+1}Z \quad (21)$$

Where

$$Z = |v_1 + v_2 + v_3 + v_4|_{2^{4n}-1} \quad (22)$$

$$v_1 = |-2^{2n-1}x_1|_{2^{4n}-1}$$

$$= \left| -2^{2n-1}\left(\underbrace{0 \dots 00}_{2n-1 \text{ bits}}\ \underbrace{x_{1,2n} \ \dots\ x_{1,1}\, x_{1,0}}_{2n+1 \text{ bits}}\right)\right|_{2^{4n}-1}$$

$$= \underbrace{\bar{x}_{1,2n} \dots \bar{x}_{1,1}\, \bar{x}_{1,0}}_{2n+1 \text{ bits}}\ \underbrace{11 \dots 1}_{2n-1 \text{ bits}}. \quad (23)$$

$$v_2 = |(2^{2n-1} - 2^{2n-2} - 2^{4n-2})x_2|_{2^{4n}-1}$$

$$= |(2^{2n-2} - 2^{4n-2})x_2|_{2^{4n}-1}. \quad (24)$$

We can divide equation (24) in two parts $v_{21}$ and $v_{22}$ as follows:

$$v_2 = |v_{21} + v_{22}|_{2^{4n}-1}. \quad (25)$$

Where

$$v_{21} = |2^{2n-2}x_2|_{2^{4n}-1}$$

$$= \left| 2^{2n-2}\left(\underbrace{0 \dots 00}_{2n-1 \text{ bits}}\ \underbrace{x_{2,2n} \ \dots\ x_{2,1}\, x_{2,0}}_{2n+1 \text{ bits}}\right)\right|_{2^{4n}-1}$$

$$= 0\ \underbrace{x_{2,2n} \ \dots\ x_{2,1}\, x_{2,0}}_{2n+1 \text{ bits}}\ \underbrace{0 \dots 00}_{2n-2 \text{ bits}}. \quad (26)$$

$$v_{22} = |-2^{4n-2}x_2|_{2^{4n}-1}$$

$$= \left| -2^{4n-2}\left(\underbrace{0 \dots 00}_{2n-1 \text{ bits}}\ \underbrace{x_{2,2n} \ \dots\ x_{2,1}\, x_{2,0}}_{2n+1 \text{ bits}}\right)\right|_{2^{4n}-1}$$

$$= \bar{x}_{2,1}\, \bar{x}_{2,0}\ \underbrace{1 \dots 11}_{2n-1 \text{ bits}}\ \underbrace{\bar{x}_{2,2n} \ \dots\ \bar{x}_{2,3}\, \bar{x}_{2,2}}_{2n-1 \text{ bits}}. \quad (27)$$

Then, $v_3$ can be written as

$$v_3 = |(2^{2n-3}(2^{2n}+1) - 2^{n-3}(2^{2n}+1))x_3|_{2^{4n}-1} \quad (28)$$

Now, equation (28) can be computed by

$$v_3 = |v_{31} + v_{32}|_{2^{4n}-1}. \quad (29)$$

Where

$$v_{31} = |2^{2n-3}(2^{2n}+1)x_3|_{2^{4n}-1}$$

$$= \left| 2^{2n-3}(2^{2n}+1)\left(\underbrace{0 \dots 00}_{n-1 \text{ bits}}\ \underbrace{x_{3,n} \ \dots\ x_{3,1}\, x_{3,0}}_{n+1 \text{ bits}}\right)\right|_{2^{4n}-1}$$

$$= \left| \begin{matrix} 2^{2n-3}\left(\underbrace{0 \dots 00}_{n-1 \text{ bits}}\ \underbrace{x_{3,n} \ \dots\ x_{3,1}\, x_{3,0}}_{n+1 \text{ bits}}\right) \\ \underbrace{0 \dots 00}_{n-1 \text{ bits}}\ \underbrace{x_{3,n} \ \dots\ x_{3,1}\, x_{3,0}}_{n+1 \text{ bits}} \end{matrix}\right|_{2^{4n}-1}$$

$$= (x_{3,2}\, x_{3,1}\, x_{3,0}\ \underbrace{0 \dots 00}_{n-1 \text{ bits}}\ \underbrace{x_{3,n} \ \dots\ x_{3,1}\, x_{3,0}}_{n+1 \text{ bits}}$$
$$\underbrace{0 \dots 00}_{n-1 \text{ bits}}\ \underbrace{x_{3,n} \ \dots\ x_{3,4}\, x_{3,3}}_{n-2 \text{ bits}}). \quad (30)$$

$$v_{32} = |-2^{n-3}(2^{2n}+1)x_3|_{2^{4n}-1}$$

$$= \left| -2^{n-3}(2^{2n}+1)\left(\underbrace{0 \dots 00}_{n-1 \text{ bits}}\ \underbrace{x_{3,n} \ \dots\ x_{3,1}\, x_{3,0}}_{n+1 \text{ bits}}\right)\right|_{2^{4n}-1}$$

$$= \left| \begin{matrix} -2^{n-3}\left(\underbrace{0 \dots 00}_{n-1 \text{ bits}}\ \underbrace{x_{3,n} \ \dots\ x_{3,1}\, x_{3,0}}_{n+1 \text{ bits}}\right) \\ \underbrace{0 \dots 00}_{n-1 \text{ bits}}\ \underbrace{x_{3,n} \ \dots\ x_{3,1}\, x_{3,0}}_{n+1 \text{ bits}} \end{matrix}\right|_{2^{4n}-1}$$

$$= 11\ \underbrace{\bar{x}_{3,n} \ \dots\ \bar{x}_{3,1}\, \bar{x}_{3,0}}_{n+1 \text{ bits}}\ \underbrace{1 \dots 11}_{n-1 \text{ bits}}\ \underbrace{\bar{x}_{3,n} \ \dots\ \bar{x}_{3,1}\, \bar{x}_{3,0}}_{n+1 \text{ bits}}\ \underbrace{1 \dots 11}_{n-3 \text{ bits}}. \quad (31)$$

Finally, $v_4$ can be calculated as follows:

$$v_4 = |2^{n-3}(2^{3n} + 2^{2n} + 2^n + 1)x_6|_{2^{4n}-1}$$

$$= \left| \begin{matrix} 2^{n-3}(2^{3n} + 2^{2n} + 2^n + 1) \\ \times\ (\underbrace{x_{6,n-1} \ \dots\ x_{6,1}\, x_{6,0}}_{n \text{ bits}}) \end{matrix}\right|_{2^{4n}-1}$$

$$= \left| \begin{matrix} 2^{n-3}\left(\underbrace{x_{6,n-1} \ \dots\ x_{6,1}\, x_{6,0}}_{n \text{ bits}}\ \underbrace{x_{6,n-1} \ \dots\ x_{6,1}\, x_{6,0}}_{n \text{ bits}}\right. \\ \left.\underbrace{x_{6,n-1} \ \dots\ x_{6,1}\, x_{6,0}}_{n \text{ bits}}\ \underbrace{x_{6,n-1} \ \dots\ x_{6,1}\, x_{6,0}}_{n \text{ bits}}\right) \end{matrix}\right|_{2^{4n}-1}$$

$$= (x_{6,2}\, x_{6,1}\, x_{6,0}\, \underbrace{x_{6,n-1}\, x_{6,n-2} \ \dots\ x_{6,1}\, x_{6,0}}_{n \text{ bits}}$$
$$\underbrace{x_{6,n-1}\, x_{6,n-2} \ \dots\ x_{6,1}\, x_{6,0}}_{n \text{ bits}}\ \underbrace{x_{6,n-1}\, x_{6,n-2} \ \dots\ x_{6,1}\, x_{6,0}}_{n \text{ bits}}$$
$$\underbrace{x_{6,n-1}\, x_{6,n-2} \ \dots\ x_{6,4}\, x_{6,3}}_{n-3 \text{ bits}}). \quad (32)$$

Because, (20), (27) and (31) have some constant bits, thus we can use the following vectors in place of $v_1$, $v_{22}$, and $v_{32}$.

$$v_1' = \underbrace{\bar{x}_{1,2n} \dots \bar{x}_{1,1}\, \bar{x}_{1,0}}_{2n+1 \text{ bits}}\ \underbrace{\bar{x}_{2,2n} \ \dots\ \bar{x}_{2,3}\, \bar{x}_{2,2}}_{2n-1 \text{ bits}} \quad (33)$$

$$v_{22}' = 11\ \underbrace{1 \dots 11}_{2n-1 \text{ bits}}\ \underbrace{1 \dots 11}_{2n-1 \text{ bits}}. \quad (34)$$

$$v_{32}' = (\bar{x}_{2,1}\, \bar{x}_{2,0}\ \underbrace{\bar{x}_{3,n} \ \dots\ \bar{x}_{3,1}\, \bar{x}_{3,0}}_{n+1 \text{ bits}}\ \underbrace{1 \dots 11}_{n-1 \text{ bits}}$$

$$\underbrace{\bar{x}_{3,n} \dots \bar{x}_{3,1}\bar{x}_{3,0}}_{n+1 \text{ bits}}\underbrace{1 \dots 11}_{n-3 \text{ bits}}) \quad (35)$$

We know that

$$v'_{22} = |\underbrace{1 \dots 11}_{4n \text{ bits}}|_{2^{4n}-1} = |2^{4n}-1|_{2^{4n}-1} = 0 \quad (36)$$

Therefore, Equation (19) can be computed as follows:

$$Z = |v'_1 + v_{21} + v_{31} + v'_{32} + v_4|_{2^{4n}-1}. \quad (37)$$

***Designing part B:*** Regard the subset $\{2^{n/2}+1, 2^{n/2}-1\}$ and let the corresponding residues of the integer number $L$ be $(\alpha_4, \alpha_5)$. For $\alpha_4 = 2^{n/2}+1$ and $\alpha_5 = 2^{n/2}-1$ the residues have bit-level representation as:

$$\alpha_4 = (\underbrace{\alpha_{4,n/2}\ \alpha_{4,(n-2)/2} \dots \alpha_{4,1}\ \alpha_{4,0}}_{(n/2)+1 \text{ bit}})_2 \quad (38)$$

$$\alpha_5 = (\underbrace{\alpha_{5,(n-2)/2}\ \alpha_{5,(n-4)/2} \dots \alpha_{5,1}\ \alpha_{5,0}}_{(n/2) \text{ bit}})_2 \quad (39)$$

Now, implementation of part B can be done with MRC algorithm. At first we must obtain the multiplicative inverse $|\alpha_4^{-1}|_{\alpha_5}$ with equations as follow:

$$\left|\,|\alpha_4^{-1}|_{\alpha_5} \times 2^{(n/2)} + 1\right|_{2^{(n/2)}-1} = 1 \quad (40)$$

Then

$$|\alpha_4^{-1}|_{\alpha_5} = 2^{(n-2)/2}. \quad (41)$$

We have

$$h_1 = \alpha_4. \quad (42)$$

$$h_2 = |(\alpha_5 - \alpha_4) \times 2^{(n-2)/2}|_{2^{(n/2)}-1}. \quad (43)$$

So

$$X = h_1 + h_2\alpha_4 \quad (44)$$

$$X = \alpha_4 + (2^{(n/2)}+1)\left|(\alpha_5 - \alpha_4) \times 2^{(n-2)/2}\right|_{2^{(n/2)}-1}. \quad (45)$$

Equation (43) can be simplified as below

$$h_2 = |\alpha_1 + \alpha_2|_{2^{(n/2)}-1}. \quad (46)$$

$$\alpha_1 = \left|2^{(n-2)/2}(\underbrace{\alpha_{5,(n-2)/2}\ \alpha_{5,(n-4)/2} \dots \alpha_{5,1}\ \alpha_{5,0}}_{(n/2)\ bits})\right|_{2^{(n/2)}-1}$$

$$= \underbrace{\alpha_{5,0}\alpha_{5,(n-2)/2}\ \alpha_{5,(n-4)/2} \dots \alpha_{5,1}}_{(n/2)\ bits}. \quad (47)$$

$$\alpha_2 = \left|-2^{(n-2)/2}(\underbrace{\alpha_{4,n/2}\ \alpha_{4,(n-2)/2} \dots \alpha_{4,1}\ \alpha_{4,0}}_{(n/2)+1\ bits})\right|_{2^{(n/2)}-1}$$

$$= \left|-2^{(n-2)/2}\left(\begin{matrix}\alpha_{4,n/2} \times 2^{n/2} \\ + \underbrace{\alpha_{4,(n-2)/2} \dots \alpha_{4,1}\ \alpha_{4,0}}_{(n/2)\ bits}\end{matrix}\right)\right|_{2^{(n/2)}-1}. \quad (48)$$

We know that $\alpha_4$ is a number that is smaller than $2^{n/2}+1$, so we can consider two cases for $\alpha_4$. First, $\alpha_4$ is smaller than $2^{n/2}$, and the second, when $\alpha_4$ is equal to $2^{n/2}$. Therefore, if $\alpha_{4,n/2} = 0$, we have

$$\alpha_{21} = \left|-2^{(n-2)/2}(\underbrace{\alpha_{4,(n-2)/2} \dots \alpha_{4,1}\ \alpha_{4,0}}_{(n/2)\ bits})\right|_{2^{(n/2)}-1}$$

$$= \underbrace{\bar{\alpha}_{4,0}\bar{\alpha}_{4,(n-2)/2} \dots \bar{\alpha}_{4,1}}_{(n/2)\ bits}. \quad (49)$$

Else if $\alpha_{4,n/2} = 1$, the following binary vector can be achieved as

$$\alpha_{22} = \left|-2^{(n-2)/2} \times 2^{n/2}(\underbrace{0 \dots 00}_{(n/2)-1\ bits}\ \alpha_{4,n/2})\right|_{2^{(n/2)}-1}$$

$$= 0\ \underbrace{1 \dots 11}_{(n/2)-1\ bits}. \quad (50)$$

Therefore, $\alpha_2$ can be calculated as below

$$\alpha_2 = \begin{cases} \alpha_{21} & \text{if } \alpha_{4,n/2} = 0 \\ \alpha_{22} & \text{if } \alpha_{4,n/2} = 1 \end{cases}. \quad (51)$$

Now, we can rewrite (45) as follow

$$X = \alpha_4 + (2^{(n/2)}+1)(\underbrace{h_{2,(n-2)/2} \dots h_{2,1}h_{2,0}}_{(n/2)\ bits}). \quad (52)$$

$$X = \alpha_4 + Y \quad (53)$$

$$Y = \underbrace{h_{2,(n-2)/2} \dots h_{2,1}h_{2,0}}_{(n/2)\ bits}\underbrace{h_{2,(n-2)/2} \dots h_{2,1}h_{2,0}}_{(n/2)\ bits}. \quad (54)$$

$$X = \underbrace{\alpha_{4,n/2}\ \alpha_{4,(n-2)/2} \dots \alpha_{4,1}\ \alpha_{4,0}}_{(n/2)+1\ bit} + \underbrace{\alpha_{n-1} \dots \alpha_1\alpha_0}_{n\ bits}. \quad (55)$$

Since, we can use the $L$ as input for module $\alpha_6 = 2^n - 1$ instead of $\alpha_6$. Thus we can rewrite equation (32) as follow:

$$\alpha_4 = \alpha_2\ \alpha_1\ \alpha_0\underbrace{\alpha_{n-1}\ \alpha_{n-2} \dots \alpha_1\ \alpha_0}_{n\ bits}\underbrace{\alpha_{n-1}\ \alpha_{n-2} \dots \alpha_1\ \alpha_0}_{n\ bits}$$

$$\underbrace{\alpha_{n-1}\ \alpha_{n-2} \dots \alpha_1\ \alpha_0}_{n\ bits}\underbrace{\alpha_{n-1}\ \alpha_{n-2} \dots \alpha_4\ \alpha_3}_{n-3\ bits}. \quad (56)$$

At last, the number X in (21) is achieved by below equation

$$X = \alpha_1 + 2^{2n+1}Z = \underbrace{\alpha_{4n-1} \dots \alpha_1\ \alpha_0}_{4n\ bits}\underbrace{\alpha_{1,2n} \dots \alpha_{1,1}\ \alpha_{1,0}}_{2n+1\ bits}. \quad (57)$$

**Example 1**: Examine the modulus set $\{2^{2n+1}, 2^{2n}+1, 2^n+1, 2^{n/2}+1, 2^{n/2}-1\}$ where $n = 4$. The weighted number $X$ can be computed from its RNS representation (97, 87, 5, 2, 2) as follow For $n = 4$ the modulus set is {512, 257, 17, 5, 3} and also residues have binary representation as below

$$\alpha_1 = 97 = (001100001)_2$$
$$\alpha_2 = 87 = (001010111)_2$$
$$\alpha_3 = 5 = (00101)_2$$
$$\alpha_4 = 2 = (010)_2$$
$$\alpha_5 = 2 = (10)_2.$$

By supposing the values of residues and $n = 4$ in (33), (26), (32), (30) and (35) we have

$$\alpha'_1 = (1100111101101010)_2 = 53098$$
$$\alpha_{21} = (0001010111000000)_2 = 5568$$
$$\alpha_{31} = (1010000010100000)_2 = 41120$$
$$\alpha'_{32} = (0011010111110101)_2 = 13813$$
$$\alpha_4 = (0100010001000100)_2 = 17476$$

Next, the required values must be substituted in (37). So

$$Z = |53098 + 5568 + 41120 + 13813 + 17476|_{65535}$$
$$= 5 = (0000000000000101)_2$$

Finally, by letting values of $Z$ and $\alpha_1$ in (57) the weighted number $X$ can be calculated as follows:

$$X = (\underbrace{0000000000000101}_{16}\underbrace{001100001}_{9})_2 = 2657.$$

To show the accuracy of these equations, we have:

$$\alpha_1 = |2657|_{512} = 97$$

$$\square_2 = |2657|_{257} = 87$$
$$\square_3 = |2657|_{17} = 5$$
$$\square_4 = |2657|_5 = 2$$
$$\square_5 = |2657|_3 = 2.$$

So, the weighted number *2657* in the RNS based on the 5-moduli set {512, 257, 17, 5, 3}has representation as (97, 87, 5, 2, 2).

## HARDWARE ARCHITECTURE

Hardware implementation of the suggested reverse converter for the 5-moduli set $\{2^{2\square+1}, 2^{2\square}+1, 2^{\square}+1, 2^{\square/2}+1, 2^{\square/2}-1\}$ with input residues $(\square_1, \square_2, \square_3, \square_4, \square_5)$ is shown in Figure 1. Implementation is based on (37) and (57). In the first step, the operand preparation unit 1 (OPU 1) prepares the needed operands (33), (26), (30), (35), (47) and (51) and these preparations depend on simply manipulating the routing of the bits of the residues. We require $6.5\square + 4$ NOT gates for doing the inversions in (33), (35) and (49). Also, a (*n/2*)-bit $2\times1$ multiplexer (MUX) is used for obtaining (51). Realization of (37) depends on a 5-operand modulo ($2^{4\square}-1$) adder [21], and it can be implemented by three $4\square$-bit carry save adders (CSAs) with EAC followed by one modulo ($2^{4\square}-1$) adder. Because (26) and (30) has ($4\square-3$) bits of 0's ($4\square-3$) of the FA's in CSA1 and CSA2 are reduced to the pairs of XOR/AND gates. For achieved (46) we used modulo ($2^{\square/2}-1$) adder. In addition, OPU2 and OPU3 prepare the needed operands (54) and (56) respectively, and these preparations depend on simply manipulating the routing of the bits of the residues. For done (55) we used carry propagate adder1 (CPA1). Also, since (35) has ($2\square-4$) bits of 1's, ($2\square-4$) of the FA's in CSA3 are reduced to the pairs of XNOR/OR gates. It is important to know that realization of (57) depend on simple concatenation without the use of any calculative hardware.
It can be seen from Table I comparison of conversion delay and hardware requirements of the different reverse converters.

## MODULAR MULTIPLICATION

To calculate the modular multiplication of two integers *A* and *B*, we can consider the binary product of the input operands followed by a reduction of the product, modulo *m* [22]. Considering $P = A \times B$, and $\square_{[\square-1:0]}$, the *n* least significantly bits of product, the subsequent equations depicts the arithmetic operations included in the calculation of the multiplications modulo $\{2^{\square/2}-1\}$, $\{2^{\square/2}+1\}$, $\{2^{\square}+1\}$, $\{2^{2\square}+1\}$ and $\{2^{2\square+1}\}$.

***Modulo $\{2^{n/2}-1\}$:*** By Considering *P* as the binary product of *A* and *B*, we can describe multiplication modulo $\{2^{\square/2}-1\}$ by:

$$|\square\times\square|_{2^{\square/2}-1} = |2^{\square/2}.\square_{[\square-1:n/2]} + 2^0.\square_{[(\square/2)-1:0]}|_{2^{\square/2}-1}$$
$$= |2^{\square/2}.\square_1 + 2^0.\square_0|_{2^{\square/2}-1}$$
$$= |\square_1+\square_0|_{2^{\square/2}-1}. \quad (58)$$

***Modulo $\{2^{n/2}+1\}$:*** The multiplication modulo$\{2^{n/2}+1\}$ of *A* and *B*, can be calculated by:

$$|\square\times\square|_{2^{\square/2}+1} = |2^{\square}.\square_{[\square+1:n]} + 2^{\square/2}.\square_{[\square-1:n/2]} + 2^0.\square_{[(\square/2)-1:0]}|_{2^{\square/2}+1}$$
$$= |\square_2-\square_1+\square_0|_{2^{\square/2}+1}. \quad (59)$$

***Modulo $\{2^n+1\}$:*** For this modulo, the modular multiplication can be illustrated by:

$$|\square\times\square|_{2^{\square}+1} = |2^{2\square}.\square_{[2\square+1:2n]} + 2^{\square}.\square_{[2\square-1:n]} + 2^0.\square_{[\square-1:0]}|_{2^{\square}+1}$$
$$= |\square_2-\square_1+\square_0|_{2^{\square}+1}. \quad (60)$$

***Modulo $\{2^{2n}+1\}$:*** The multiplication modulo $\{2^{2n}+1\}$ of *A* and *B*, can be computed by:

$$|\square\times\square|_{2^{2\square}+1} = |2^{4\square}.\square_{[4\square+1:4n]} + 2^{3\square}.\square_{[4\square-1:3n]} + 2^{2\square}.\square_{[3\square-1:2n]} + 2^{\square}.\square_{[2\square-1:n]} + 2^0.\square_{[\square-1:0]}|_{2^{2\square}+1}$$
$$= |\square_4 - 2^{\square}.\square_3 - \square_2 + 2^{\square}.\square_1 + \square_0|_{2^{2\square}+1}. \quad (61)$$

***Modulo $\{2^{2n+1}\}$:*** Considering *P* as the binary product of *A* and *B*, multiplication modulo $\{2^{2n+1}\}$ can be described as follows:

$$|\square\times\square|_{2^{2\square+1}} = |2^{4\square}.\square_{[4\square+1:4n]} + 2^{3\square}.\square_{[4\square-1:3n]} + 2^{2\square}.\square_{[3\square-1:2n]} + 2^{\square}.\square_{[2\square-1:n]} + 2^0.\square_{[\square-1:0]}|_{2^{2\square}+1}$$
$$= |0+0+2^{2\square}.\square_2 + 2^{\square}.\square_1 + \square_0|_{2^{2\square+1}}. \quad (62)$$

As shown in Table I, the proposed modulus set $\{2^{2\square+1}, 2^{2\square}+1, 2^{\square}+1, 2^{\square/2}+1, 2^{\square/2}-1\}$ provides the dynamic range greater than other modulus sets. The dynamic range of this modulus set is $6\square$-bit. Therefore more efficient modular multiplication is obtained by the suggested modulus set. Since the modular multiplication is the important part of public key cryptography algorithms specifically RSA and elliptic curve cryptography, thus this proposed modulus set with high dynamic range will be very useful to improving these applications.

## CONCLUSION

This paper introduced high dynamic range 5-moduli set $\{2^{2\square+1}, 2^{2\square}+1, 2^{\square}+1, 2^{\square/2}+1, 2^{\square/2}-1\}$. Also, efficient reverse converter for this modulus set based on New CRT-I and MRC is presented. The proposed converter architecture is memory-less and adder-based and it can be efficiently pipelined. Furthermore, with the new proposed modulus set, the internal RNS arithmetic circuits as well as the reverse converter can be implemented efficiently.

Table: 1 Conversion Delay and Hardware Complexity Comparison

| *Modulus set* | *DR* *bit* | *Area* | *Delay* |
|---|---|---|---|
| $\{2^n-1,\ 2^n, 2^n+1, 2^{n+1}-1, 2^{n-1}-1\}$ [23] | $5n-1$ | $((5n^2+43n+m^*)/6+16n-1)A_{FA}$ $+(6n+1)A_{NOT}$ | $(18n+L^*+7)t_{FA}$ |
| $\{2^n, 2^{2n+1}-1, 2^{n/2}-1, 2^{n/2}+1, 2^n+1\}$ [24] | $5n+1$ | $(12.5n+6)A_{FA}+(4.5n-1)A_{XNOR}$ $+(4.5n-1)A_{OR}+(1.5n-1)A_{XOR}$ $+(1.5-1n)A_{AND}+(7n+1)A_{NOT}$ | $(12n+6)t_{FA}+3t_{NOT}$ |
| Proposed $\{2^{2n+1}, 2^{2n}+1, 2^n+1, 2^{n/2}+1, 2^{n/2}-1\}$ | $6n+1$ | $(11.5n+7)A_{FA}+(2n-4)A_{XNOR}$ $+(2n-4)A_{OR}+(4n-3)A_{XOR}$ $+(4n-3)A_{AND}+(6.5n+4)A_{NOT}$ $+(n/2)A_{MUX2\times1}$ | $(10n+1)t_{FA}+t_{NOT}+t_{MUX}$ |

* m=n-4, 9n-12 and 5n-8 for n=6k-2, 6k and 6k+2, respectively, and L is the number of the levels of a CSA tree with ((n/2)+1) inputs.

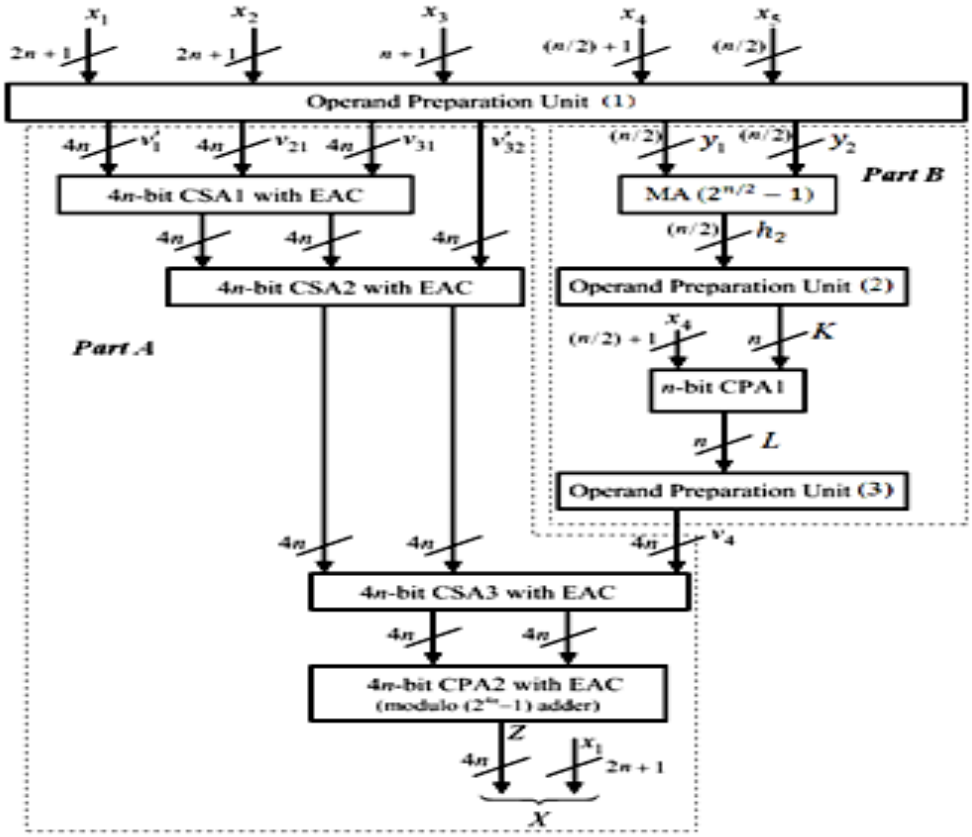


Figure: 1 Converter for modulus set $\{2^{2n+1}, 2^{2n}+1, 2^n+1, 2^{n/2}+1, 2^{n/2}-1\}$.

## REFERENCES


**[1].** B. Parhami, Computer Arithmetic, Oxford University Press, 2000.

**[2].** A. S. Molahosseini, K. Navi, C. Dadkhah, O. Kavehei, and S. Timarchi "Efficient Reverse Converter Designs for the New 4-Moduli Sets $\{2^n+1, 2^n, 2^n+1, 2^{2n+1}-1\}$ and $\{2^n-1, 2^n+1, 2^{2n}, 2^{2n}+1\}$Based on New CRTs," IEEE Trans. Circuits Syst.I: Reg. Papers, vol 57, no. 4, pp.823–834 April 2010.

**[3].** K. Navi, A. S. Molahosseini, M. Esmaeildoust, "How to Teach Residue Number System to Computer Scientists and Engineers," IEEE Transactions on Education, vol. 53, no. 3, 2010.

**[4].** W. Wang, M. N. S. Swamy, and M. O. Ahmad, "Moduli selection in RNS for efficient VLSI implementation," in Proc. IEEE Int. Symp. Circuits Syst., pp. 25–28, 2003.

**[5].** 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 Process., vol. 50, no. 7, pp. 1772–1779, Jul. 2002.

**[6].** A. Hiasat and A. Sweidan, "Residue number system to binary converter for the moduli set $(2^n-1,\ 2^n-1, 2^n+1)$," Elsevier J. Syst.Architect., vol. 49, pp. 53–58, 2003.

**[7].** W.Wang, M. N. S. Swamy,M.O. Ahmad, and Y.Wang, "A high-speed residue-to-binary converter and a scheme of its VLSI implementation," IEEE Trans. Circuits Syst. II, Analog. Digit. Signal Process., vol. 47, no. 12, pp. 1576–1581, Dec. 2000.

**[8].** P. V. A. Mohan, "RNS-to-binary converter for a new three-moduli set $\{2^{n+1}-1,\ 2^n, 2^n-1\}$," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 54, no. 9, pp. 775–779, Sep. 2007.

**[9].** P. V. A. Mohan and A. B. Premkumar, "RNS-to-binary converters for two four-moduli set $\{2^n-1,\ 2^n, 2^n+1, 2^{n+1}-1\}$ and $\{2^n-1,\ 2^n, 2^n+1, 2^{n+1}+1\}$," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 54, no. 6, pp. 1245–1254, Jun. 2007.

**[10].** B. Cao, T. Srikanthan, and C. H. Chang, "Efficient reverse converters for the our-moduli sets $\{2^n-1,\ 2^n, 2^n+1, 2^{n+1}-1\}$ and $\{2^n-1,\ 2^n, 2^n+1, 2^{n-1}-1\}$," Proc. IEE Comput. Digit. Tech., vol. 152, pp. 687–696, 2005.

**[11].** P. V. A. Mohan, "New reverse converters for the moduli set $\{2^n-3,\ 2^n-1, 2^n+1, 2^n+3\}$," Elsevier J. Electron. Commun. (AEU), vol. 62, no. 9, pp. 643–658, 2008.

**[12].** A. Hariri, K. Navi, and R. Rastegar, "A newhigh dynamic range moduli set with efficient reverse converter," Elsevier J. Comput. Math. With Appl., vol. 55, no. 4, pp. 660–668, 2008.

**[13].** B. Cao, C. H. Chang, and T. Srikanthan, "An efficient reverse converter for the 4-moduli set$\{2^n-1,\ 2^n, 2^n+1, 2^{2n}+1\}$ based on the new Chinese remainder theorem," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 50, no. 10, pp. 1296–1303, 2003.

**[14].** A. A. Hiasat, "VLSI implementation of new arithmetic residue to binary decoders," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 13, no. 1, pp. 153–158, Jan. 2005.

**[15].** W. Zhang and P. Siy, "An efficient design of residue to binary converter for four moduli set $\{2^n-1, 2^n+1, 2^{2n}-2, 2^{2n+1}-3\}$ based on new CRT II," J. Inf. Sci., vol. 178, no. 1, pp. 264–279, 2008.

**[16].** A.S. Molahosseini, K. Navi, "A Reverse Converter for the Enhanced Moduli Set $\{2^n-1, 2^n+1, 2^{2n}, 2^{2n+1}-1\}$ Using CRT and MRC," IEEE Annual Symposium on VLSI. Pp 456–457, 2010.

**[17].** F. J. Taylor, "Residue arithmetic: A tutorial with examples," IEEE Computer, vol. 17, pp. 50–62, 1984.

**[18].** Y. Wang, "Residue-to-binary converters based on new Chinese remainder theorems," IEEE Trans. Circuits Syst. II, Analog. Digit. Signal Process., vol. 47, no. 3, pp. 197–205, Mar. 2000.

**[19].** A. Omondi and B. Premkumar, "Rrsidue Number Systems: Theory and Implementations," Imperial College Press, Londen 2007.

**[20].** S. J. Piestrak, "A high speed realization of a residue to binary converter," IEEE Trans. Circuits Syst. II, Analog. Digit. Signal Process., vol. 42, no. 10, pp. 661–663, Oct. 1995.

**[21].** S. J. Piestrak, "Design of residue generators and multioperand modular adders using carry-save adders," IEEE Trans. Comput., vol. 423, no. 1, pp. 68–77, Jan. 1994.

**[22].** P. M. Matutino, R. Chaves, L. Sousa, "Arithmetic units for RNS moduli $\{2^n-3\}$ and $\{2^n+3\}$ operations," 13th Euromicro Conference on Digital System Design: Architectures, Methods and Tools, pp. 243-246, 2010.

**[23].** B. Cao, C. H. Chang, and T. Srikanthan, "A residue-to-binary converter for a new five-moduli set," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 54, no. 5, pp. 1041–1049, May 2007.

**[24].** M. Esmaeildoust, K. Navi, M.R. Taheri, "High speed reverse converter for new five-moduli set $\{2^n, 2^{2n+1}-1, 2^{n/2}-1, 2^{n/2}+1, 2^n+1\}$," IEICE Electronics. Express, vol.7, no.3, pp. 118-125,2010.

### Short Biodata for the Author

**Ramin Aliabadian** was born in Babol, a city of North of Iran, in 1983. Received B.Sc. in Computer Hardware Engineering from the Shomal University, Amol, Iran, in 2007. He is currently M.Sc. student in Computer Architecture at Islamic Azad University, Arak branch, Iran. His research interests are Computer Arithmetic and RFID

**Mehdi Hosseinzadeh** Was born in Dezful, a city in the southwestern of Iran, in 1981. Received B.Sc. in Computer Hardware Engineering from Islamic Azad University, Dezful branch, Iran in 2003. He also received the M.Sc. and Ph.D. degree in Computer System Architecture from the Science and Research Branch, Islamic Azad University, Tehran, Iran in 2005 and 2008, respectively. He is currently Assistant Professor in Department of Computer Engineering of Science and Research Branch of Islamic Azad University, Tehran, Iran. His research interests are Computer Arithmetic with emphasis on Residue Number System, Cryptography, Network Security and E-Commerce.

**Mehdi Golsorkhtabaramiri** received the B.E. degree in 2007 in Computer Hardware Engineering, and the M. Eng. Degree in 2010 in Computer systems Architecture Engineering, from Islamic Azad University, Tabriz branch, Iran. He is currently working toward a Ph.D. degree at the Department of Computer Engineering, Science and Research branch, Islamic Azad University Tehran, Iran. His current research interests lie in the areas of wireless communications and networking and Residue Number System. He is editorial board member of Journal of Global Research in Electronics & Communications.