# Fast Overflow Detection in Moduli Set $\{2^n - 1, 2^n, 2^n + 1\}$

**Mehrin Rouhifar[1,*], Mehdi Hosseinzadeh[2], Saeid Bahanfar[3] and Mohammad Teshnehlab[4]**

**[1] Dept. of Computer engineering , Islamic Azad University, Tabriz branch
Tabriz, Iran**

**[2] Islamic Azad University, Science and research branch
Tehran, Iran**

**[3] Dept. of Computer engineering , Islamic Azad University, Tabriz branch
Tabriz, Iran**

**[4] Dept. of Control engineering, K. N. Toosi University of Technology
Tehran, Iran**

**Abstract**

The Residue Number System (RNS) is a non weighted system. It supports parallel, high speed, low power and secure arithmetic. Detecting overflow in RNS systems is very important, because if overflow is not detected properly, an incorrect result may be considered as a correct answer. The previously proposed methods or algorithms for detecting overflow need to residue comparison or complete convert of numbers from RNS to binary. We propose a new and fast overflow detection approach for moduli set $\{2^n\text{-}1, 2^n, 2^n\text{+}1\}$, which it is different from previous methods. Our technique implements RNS overflow detection much faster applying a few more hardware than previous methods.

***Keywords:*** *Residue number system, overflow detection, moduli set $\{2^n\text{-}1, 2^n, 2^n\text{+}1\}$, group number.*

## 1. Introduction

Residue number systems (RNS) have been for a long time a topic of intensive research. Their usefulness has been demonstrated, especially for computations where additions, subtractions and multiplications dominate, because such operations can be done independently for each residue digit without carry propagation [1]. Other operations such as overflow detection, sign detection, magnitude comparison and division in RNS are very difficult and time consuming [2, 3]. However, above mentioned operations are essential in certain applications, e.g. in exact arithmetic or computational geometry, where residue arithmetic is applied [4].

The RNS is determined by the set m of n positive coprime integers $m_i > 1$, which forms the base of the system. The dynamic range $M$ of that system is given as a product of the moduli $m_i$ where

$$M = \prod_{i=1}^{n} m_i . \tag{1}$$

Any integer $X \in [0, M)$ has a unique representation $(x_1, x_2, ..., x_n)$ in RNS $(m_1, m_2, ..., m_n)$. The residues $x_i = |X|_{m_i}$, also called residue digits, are defined as

$$x_i = X \bmod m_i , \quad 0 \le x_i < m_i . \tag{2}$$

To convert a residue number $(x_1, x_2, ..., x_n)$ into its binary representation $X$, the Chinese Reminder Theory (CRT) is widely used. In CRT, the binary $X$ is computed by:

$$X = \left\langle \sum_{i=1}^{n} (x_i N_i)_{m_i} \times M_i \right\rangle_M \tag{3}$$

where $M_i = M / m_i$ and $N_i = \left\langle M_i^{-1} \right\rangle_{m_i}$ is the multiplicative inverse $M_i$ modulo $m_i$ [5].

RNS has numerous applications in Digital Signal Processing (DSP) for filtering, convolutions, correlations, FFT computation [6, 7], fault tolerant computer systems

* Corresponding Author

[8], communication [9], cryptography and image processing [10, 11].

Overflow detection is one of the fundamental issues in efficient design of RNS systems. In a generic approach, overflow occurs in the addition of two numbers $X$ and $Y$, whenever $Z = (X+Y) \bmod M$ be less than $X$. Thus, the problem of overflow detection in RNS arithmetic is equivalent to the magnitude of the problem of comparison [12, 13]. Another algorithm which proposed for overflow detection in odd dynamic range $M$ is a ROM-based algorithm and called the parity checking technique. In this method, parity indicates whether an integer number is even or odd. Let operands of $X$ and $Y$ have the same parity and $Z = X + Y$ . So, the addition process is with overflow, if $Z$ be an odd number [14, 15]. For signed RNS, overflow occurs when the sign of the sum is different from the operands [16].

In this paper, we will propose an algorithm to detect overflow in moduli set $\{2^n\text{-}1, 2^n, 2^n\text{+}1\}$. This moduli set is one of the most popular three-module set, and can also be extended to improve the RNS dynamic range [17]. In proposed method, numbers [0, $M$ - 1] are distributed among several groups. Then, by using their group numbers, is diagnosed in the process of addition of two numbers, whether overflow has occurred or no.

## 2. Proposed Method

To detect overflow in moduli set $\{2^n\text{-}1, 2^n, 2^n\text{+}1\}$, we distribute the numbers in dynamic representation range $M$ into several groups. Since, residue representation of $X$ in mentioned moduli set is corresponding with $(x_1, x_2, x_3)$, so its group number obtained according to Fig.1.

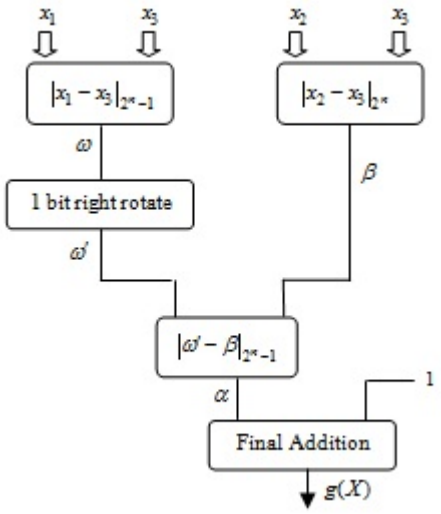


Fig. 1 Group Number Detection.

The number of groups required for this distribution is equal to γ and can be expressed as

$$\gamma = \left| |x_1 - x_3|_{2^n-1} - |x_2 - x_3|_{2^n} \right|_{2^n-1} = 2^n - 1. \qquad (4)$$

So, we can concluded that length of any group namely $l$ is given as

$$l = \frac{M}{\gamma} = \frac{(2^n-1).2^n.(2^n+1)}{2^n-1} = 2^n.(2^n+1). \qquad (5)$$

In any of these groups there are $2^n$ subgroups, because

$$\beta = |x_2 - x_3|_{2^n}, \quad 0 \le \beta \le 2^n - 1. \qquad (6)$$

For example, the value of $\beta$ for numbers in first group with range [0, $2^{2n} + 2^n$) is shown in the following:

$$\beta = |x_2 - x_3|_{2^n} \Rightarrow \begin{cases} 0 \le X < 2^n+1, & \beta = 0 \\ 2^n+1 \le X < 2\,(2^n+1), & \beta = 1 \\ \vdots & \\ (2^n-1)(2^n+1) \le X < 2^n\,(2^n+1), & \beta = 2^n-1. \end{cases} \qquad (7)$$

For determination of group number of any residue number, first should be get the value of $\omega$. For clarity, we have exhibited it in range [0, $2^{2n} + 2^n$) as follows:

$$\omega = |x_1 - x_3|_{2^n-1} \Rightarrow \begin{cases} 0 \le X < 2^n+1, & \omega = 0 \\ 2^n+1 \le X < 2\,(2^n+1), & \omega = 2 \\ 2(2^n+1) \le X < 3\,(2^n+1), & \omega = 4 \\ \vdots & \\ (2^{n-1}-1)(2^n+1) \le X < 2^{n-1}\,(2^n+1), & \omega = 2^n-2 \\ 2^{n-1}(2^n+1) \le X < (2^{n-1}+1)(2^n+1), & \omega = 1 \\ \vdots & \\ (2^n-2)(2^n+1) \le X < (2^n-1)(2^n+1), & \omega = 2^n-3 \\ (2^n-1)(2^n+1) \le X < 2^n\,(2^n+1), & \omega = 0. \end{cases} \qquad (8)$$

According to (8) and with regard to the product result from moduli subtraction in each group be appeared first, odd values and afterward even respectively. Since, in order to accomplishment of arithmetic operations should be arranged the $\omega$ values increasingly, so it is achievable through one bit right rotate. Therefore, if assume $\omega$ = 0, 2, 4, 6, …, $2^n$ - 2, 1, 3, .., $2^n$ - 3, after 1-bit right rotate, we get $\omega'$ = 0, 1, 2, …, $2^n$ - 3, $2^n$ - 2.

Now by having the values of $\beta$ and $\omega'$, the group number of any residue number in RNS (counting from 0) is defined as

$$\alpha = \left|\omega' - \beta\right|_{2^n-1}, \quad 0 \le \alpha \le 2^n - 2. \tag{9}$$

For facility in implementation of proposed algorithm, we add the obtained group number from (9) with one. In this case, if $X$ be an integer, its group number $g(X)$ is

$$g(X) = \alpha + 1, \quad 1 \le g(X) \le 2^n - 1. \tag{10}$$

Table 1 shows the distribution of numbers in dynamic range $[0, 2^{3n} - 2^n)$ which is given as a product of the $m_i$'s in moduli set $\{2^n\text{-}1, 2^n, 2^n\text{+}1\}$.

Table 1: Distribution of Numbers

| *Number* | *Group* |
|---|---|
| $0 \to 2^n(2^n+1) - 1$ | 1 |
| $2^n(2^n+1) \to 2[2^n(2^n+1)] - 1$ | 2 |
| ⋮ | |
| $(2^n-2)[2^n(2^n+1)] \to (2^n-1)[2^n(2^n+1)] - 1$ | γ |

Let $X$ and $Y$ are two operands in the process of addition $Z = X + Y$ and also $g(X)$ and $g(Y)$ be the group number of operands, respectively. It can be shown from Table 1 that:

i) if $g(X) + g(Y) < 2^n$, no overflow will occur.

ii) if $g(X) + g(Y) > 2^n$, overflow must occur.

iii) if $g(X) + g(Y) = 2^n$, overflow may or may not occur. So, is required it be checked more.

**Proof:** in case iii, range of the sum $X + Y$ in binary system is

$$(2^n - 2)[2^n(2^n + 1)] \le Z \le 2^n[2^n(2^n + 1)] - 2. \tag{11}$$

Since, $M$ is exactly located in middle of obtained range from (11), so it can be rewritten as

$$(2^n - 2)[2^n(2^n + 1)] \le M \le 2^n[2^n(2^n + 1)] - 2. \tag{12}$$

In order to proof of $g(X) + g(Y) = 2^n$, we replace the values of $(2^n - 2)$ and $2^n$ in terms of $g(X) + g(Y)$. Therefore, the final form of (12) is

$$\begin{aligned} &((g(X) - 1) + (g(Y) - 1))[2^n(2^n + 1)] \\ &< (2^n - 1)2^n(2^n + 1) < \\ &(g(X) + g(Y))[2^n(2^n + 1)] \end{aligned} \tag{13}$$

As seen, value of $2^n(2^n + 1)$ is common in the sides of inequality (13), thus it can be eliminated as follows:

$$g(X) + g(Y) - 2 < 2^n - 1 < g(X) + g(Y) \tag{14}$$

After adding one whit the sides of (14), the resulting inequality be defined as

$$g(X) + g(Y) - 1 < 2^n < g(X) + g(Y) + 1. \tag{15}$$

Finally (15) can be divided by two parts, that is

$$\begin{cases} g(X) + g(Y) < 2^n + 1 \\ g(X) + g(Y) > 2^n - 1 \end{cases} \Rightarrow g(X) + g(Y) = 2^n. \tag{16}$$

Therefore, overflow can be detected by comparing the sum of the groups of operands with $2^n$. If the sum exceeds $2^n$, overflow must occur. Notice that, overflow probability should be again checked in third mode. For this purpose, $g(X) + g(Y) = 2^n$ is given 1-bit shift to right as $2^n / 2 = 2^{n-1}$. Subsequently, it be compared with group number of sum of operands $g(Z)$. In this case, if $g(Z) > 2^{n-1}$, then overflow does not exist and otherwise $g(Z) < 2^{n-1}$ overflow has occurred. Fig. 2 shows the overflow detection circuit in moduli set $\{2^n\text{-}1, 2^n, 2^n\text{+}1\}$.

Table2: Group Number Calculations for RNS {15,16,17}

| $X$ | $X_{RNS}$ | $\beta$ | $\omega'$ | $\alpha = \lvert\omega' - \beta\rvert_{15}$ | $g(X)$ |
|---|---|---|---|---|---|
| 62 | (2, 14, 11) | 3 | 3 | 0 | 1 |
| 1045 | (10, 5, 8) | 13 | 1 | 3 | 4 |
| 1111 | (1, 7, 6) | 1 | 5 | 4 | 5 |
| 2040 | (0, 8, 0) | 8 | 0 | 7 | 8 |
| 2048 | (8, 0, 8) | 8 | 0 | 7 | 8 |
| 3097 | (7, 9, 3) | 6 | 2 | 11 | 12 |
| 4079 | (14, 15, 16) | 15 | 14 | 14 | 15 |

As an example, consider moduli set {15,16,17}. Therefore $M = 15 \times 16 \times 17 = 4080$ and the number of groups $\gamma = 15$. The example calculation for the distribution of a few values of numbers are shown in Table 2. If $X$ = 1111 and $Y$ = 2048, then $Z = X + Y = 3159 < 4079$. In RNS, according to Table 2, groups of operands are equal to 5 and 8 respectively. Based on proposed method, because sum of the group of operands 13 is less than 16, thus no overflow exits. Another instance of overflow consists of:

$X$ =1045 = (10, 5, 8) → $g(X) = 4$

$Y$ = 3097 = (7, 9, 3) → $g(Y) = 12$

Since, $g(X) + g(Y) = 16 = 2^n$ therefore, is required $g(Z)$ be compared with $2^{n-1} = 8$

$Z = |X + Y|_M = (2, 14, 11) \to g(Z) = 1$

According to our algorithm 1<8 and it denotes that an overflow has occurred. In the other words, we have: $Z = X + Y = 4142 > 4080$.

## 3. Hardware Implementation

The group detection function is determined by Eq.(10) as a sum of $\alpha$ and 1. The value of $\alpha$ is given by $\alpha = |\omega' - \beta|_{2^n-1}$. Since $\alpha$ is computed as a residue modulo $2^n$-1 then, instead of subtracting $|\beta|_{2^n-1}$ we can add its additive inverse modulo $2^n$-1. An additive inverse modulo $2^n$-1 is simply a negation of binary representation. For simplification reasons the additive inverse of $|\beta|_{2^n-1}$ is denoted as

$$\hat{\beta} = \left| -|\beta|_{2^n-1} \right|_{2^n-1}. \quad (17)$$

So that, the binary form of (17) is $\hat{\beta} = \overline{\beta}_{n-1},...,\overline{\beta}_1, \overline{\beta}_0$. Thus (9) can be rewritten as the sum

$$\alpha = \left| \omega' + \hat{\beta} \right|_{2^n-1}. \quad (18)$$

From [18], an addition modulo ($2^n$ - 1) with redundant zero elimination can be expressed as

$$|a+b|_{2^n-1} = |a+b+c_{out}+p|_{2^n} \quad (19)$$

where $c_{out}$ is a carry bit of $a + b$ addition and $p = 1$ for $a + b = 11\ldots1_2$. The sum $c_{out} + p$ is 0 for $a + b < 2^n - 1$ and 1 for $a + b \geq 2^n - 1$ [1]. By assuming that $C_{in} = c_{out} + p$, the final form of (18) is then

$$\alpha = \left| \omega' + \hat{\beta} + C_{in} \right|_{2^n}. \quad (20)$$

Also, the values of $\omega$ and $\beta$ is given using this way. Notice that, in computing of $\omega = |x_1 - x_3|_{2^n-1}$, because $x_3$ is a residue number modulo $2^n$ +1 and $x_3 \leq 2^n$ then $|x_3|_{2^n-1}$ is given by OR-ing the least and the most significant bits of $x_3$. Therefore, binary form of $\hat{x}_3$ is $\overline{x_{3,0} + x_{3,n}}, \overline{x}_{3,n-1},..., \overline{x}_{3,1}$.

To overflow detection, should be compared $g(X)+g(Y)$ with $2^n$. We know the number comparison in RNS is one of the difficult and time consuming operations, therefore attempted to do this operation whit another way. In this paper, in order to $g(X)+g(Y)$ addition, we designed a new circuit that just generates the required valves. The outputs of this unit are the most significant bit (MSB) of the sum as ($M$), a carry bit of the sum namely $C$ and $P'_1$ where if be equal to one, denotes the all the bits of the sum, expect $M$, are zero. Hence, mentioned unit is called MCP′$_1$G. Consequently, comparison operation performs as following:

$$g(X)+g(Y) \begin{cases} < 2^n, & if \quad C = 0 \\ = 2^n, & if \quad C = 1, M = 0, P'_1 = 1 \\ > 2^n, & otherwise. \end{cases} \quad (21)$$

As mentioned above, whenever $g(X)+g(Y) = 2^n$, is required to overflow probability be checked again. For this propose, $g(Z)$ be compared with $2^{n-1}$. In this case, by having the MSB of $g(Z)$ as $W = S'_{n-1}$ and its $P'_2 = P'_{0:n-2}$, can be said:

$$g(Z) \begin{cases} > 2^{n-1}, & if \quad W = 1, P'_2 = 0 \\ < 2^{n-1}, & otherwise. \end{cases} \quad (22)$$

The proposed method to overflow detection is implemented as shown in Fig. 2. The circuit consists of five main blocks: three group detection units, a unit for generation of (MSB), output carry and $P'_1$ of $g(X)+g(Y)$ addition and the final post-processing unit to detecting overflow.

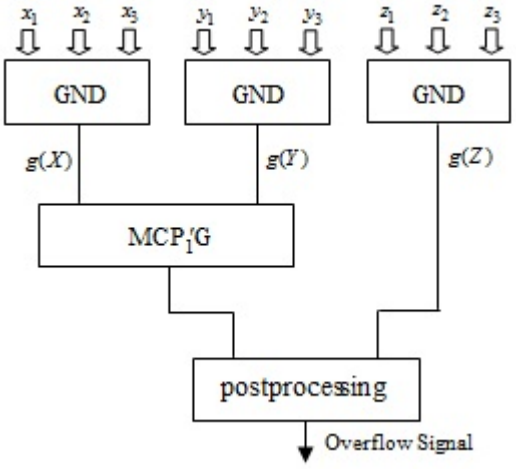


Fig. 2 Overflow detection unit.

The group number detection unit shown in Fig.1 is used for determination of group number of operands and their sum. These values are represented as three vectors $g(X)$, $g(Y)$ and $g(Z)$ respectively. The produced vectors are connected to the inputs of the MCP′$_1$G unit.

The goal of the MCP′$_1$G unit is to determination the $C = c_n$, $M = P_{n-1:n-1} \oplus G_{0:n-2}$ where $G_{0:n-2}$ is the carry of the $(n-1)$-bit of $g(X)+g(Y)$ from the position 0 to $n-2$ and also generation of $P'_1 = P'_{0:n-2}$ which detects a result in the form of $X00...0_2$. These signals can be

computed in a simplified and new prefix structure proposed in [18]. Hence, we no need to use a full *n*-bit adder.

A parallel prefix adder and also parallel prefix adder with end-around-carry are built from elements shown in Fig. 3.

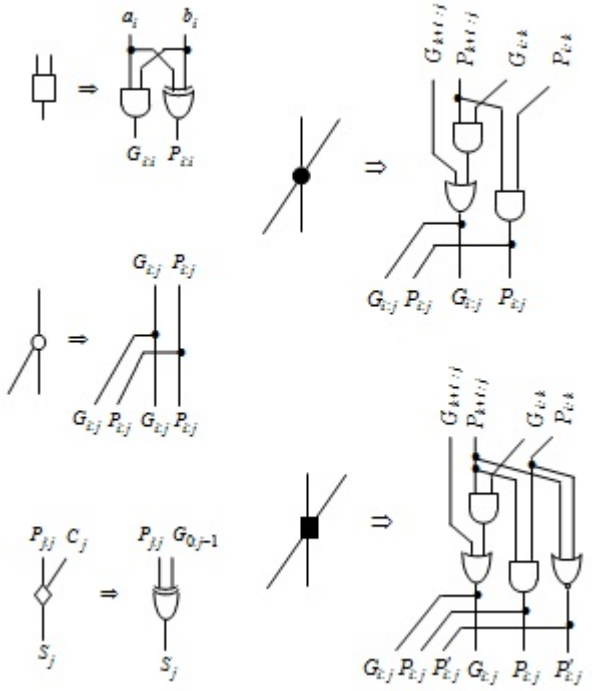

Fig. 3 Blocks of prefix adder.

The signals $G_{i:j}$ and $P_{i:j}$ are the carry generation and propagation functions from the position $i$ to $j$. The $P'_{i:j}$ signal is a function where indicates whether all the bits from the position $i$ to $j$ are equal zero or no. For an addition of two binary vectors $a_{n-1}...a_0$ and $b_{n-1}...b_0$ and for $i < k < j$, these functions can be expressed by logic equations

$$\begin{aligned} G_{i:i} &= a_i \cdot b_i \\ P_{i:i} &= a_i \oplus b_i \\ G_{i:j} &= G_{i:k} \cdot P_{k+1:j} + G_{k+1:j} \\ P_{i:j} &= P_{i:k} \cdot P_{k+1:j} \\ P'_{i:j} &= \overline{P_{i:k} + P_{k+1:j}}. \end{aligned} \tag{23}$$

The carry signals $c_j$ are equal to $G_{0:j-1}$ and the bits $s_j$ of a final sum are $s_j = P_{j:j} \oplus c_j$. An addition advantage of prefix structures is that the end-around carry can be added in the last stage with a delay cost of two logic levels [1]. The detailed description of this idea is presented in [18].

Fig.4. depicts the structure of parallel prefix adder with end-around-carry (PPA with EAC). We applied it for doing addition operations in order to obtain the values of $\alpha$ and $\omega$.

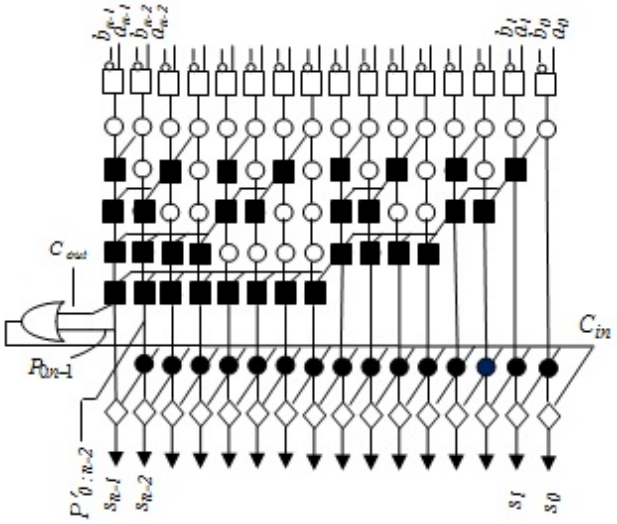

Fig. 4 Parallel prefix adder structure with End-around-carry.

The possibility of adding one bit with the delay of two logic levels enables computation of $M$ and $g(X) = \alpha + 1$. Since $s_j = P_{j:j} \oplus c_j$, then $M$ is given in the additional stage of new parallel-prefix adder. The value of $M$ from $M = P_{n-1:n-1} \oplus G_{0:n-2}$ is computed by EX-ORing of $P_{n-1:n-1}$ and $G_{0:n-2}$ of the MCP′$_1$G unit. The full circuit to evaluate $M$ for $n$ = 16 is shown in Fig. 5.

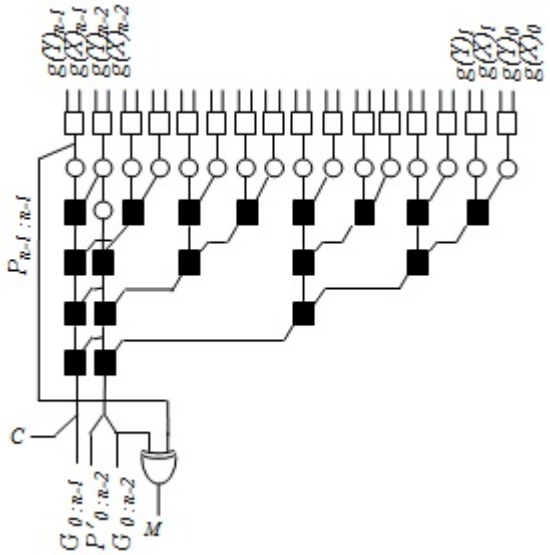

Fig. 5 MCP′$_1$G unit for $n$ =16.

The post-processing unit block diagram is shown in Fig. 6. It comprises a limited number of components, which can detect the overflow in process of addition of two numbers.

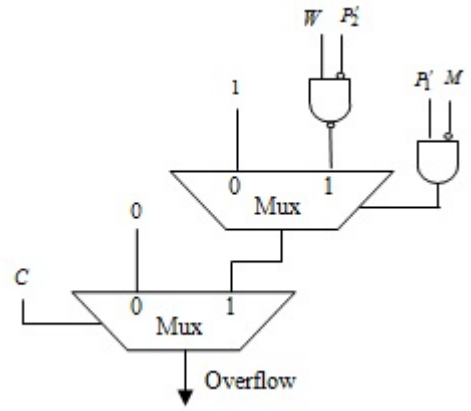


Fig. 6 post-processing unit.

The area and time (AT) characteristics of proposed circuit in order to overflow detection for RNS by moduli set $\{2^n-1, 2^n, 2^n+1\}$ are estimated using the standard unit-gate model used [18]. In this model, each gate of two-input such as AND, OR, NAND, NOR has area $A = 1$ and delay $T = 1$. Also, for each 2-input gate XOR / XNOR there are $A = T = 2$.

The group number detection unit of shown in Fig. 1. comprises three main adders: one modulo $(2^n)$ adder and two adder mod $(2^n-1)$. For calculation of $\beta$ modulo $2^n$, we used the parallel adder structure from [18] by $A = 5n + (3/2)n\log_2 n$ and $T = 2\log_2 n + 4$. As we said previously, for determination of values $\omega$ and $\alpha$, applied the PPA with EAC (Fig. 4) which it uses $(n-1)$ black nodes, $n$ input nodes (as square) and $n/2$ black square in each level where number of levels is equal be $\log_2 n$. Thus, AT parameters of any adder modulo $(2^n-1)$ are

$$\begin{aligned} A_{adder \bmod 2^n-1} &= 8n + 2n\log_2 n - 3 \\ T_{adder \bmod 2^n-1} &= 2\log_2 n + 6. \end{aligned} \quad (24)$$

After the value determination of $\alpha$ for group detection of each number, should be add $\alpha$ with 1. Therefore, $g(X)$ also obtains in the additional stage of PPA with EAC from Fig. 4. which it requires the hardware of $2n$ and delay of 2 logic levels (see Fig. 7). Consequently, area and delay of GND unit are

$$\begin{aligned} A_1 &= 23n + \frac{11}{2} n\log_2 n - 6 \\ T_1 &= 4\log_2 n + 14. \end{aligned} \quad (25)$$

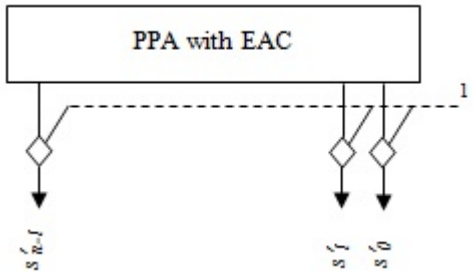


Fig.7 Final addition unit

The requirements for MCP′$_1$G unit (Fig.5) are as follows: $n$ input nodes, $(n+2)$ black square, and an additional gate. For determination of delay should be noticed the maximum number of black square that is required to working in parallel is $\log_2 n$. So, the MCP′$_1$G area and delay can expressed as

$$\begin{aligned} A_2 &= 7n + 10 \\ T_2 &= 2\log_2 n + 4. \end{aligned} \quad (26)$$

The post-processing unit contains a limited number of the gates and multiplexers. Notice that, a Mux$_{2:1}$ has $A = 3$ and $T = 2$. So, the AT parameters of mentioned unit are

$$\begin{aligned} A_3 &= 8 \\ T_3 &= 5. \end{aligned} \quad (27)$$

Total delay of the circuit is determined by a path consisting one unit of group detection, MCP′$_1$G unit and post-processing unit. The total area and delay of the designed overflow detection circuit are

$$\begin{aligned} A_{tot} &= 3A_1 + A_2 + A_3 = 76n + \frac{33}{2} n\log_2 n \\ T_{tot} &= T_1 + T_2 + T_3 = 6\log_2 n + 23. \end{aligned} \quad (28)$$

## 4. Comparison

One of the fastest and most efficient RNS comparator for the moduli set $\{2^n-1, 2^n, 2^n+1\}$ are introduced in references [17] and [19] respectively. In a generic approach, after a residue to binary convert, comparison operation can be done by using $n$ or $(n+1)$ bits comparator which has a delay of residue to binary converter plus delay of a $(n+1)$ bit Binary Comparator (BC). In Table 3 proposed technique is compared with other methods.

Table 3: Comparison Area and Delay of proposed method with other methods using unit-gate model

| ***Design*** | ***Area*** | ***Delay*** |
|---|---|---|
| [17] | $115n + 186$ | $4n + \log_2 n + 36$ |
| [19] | $96n + n\log_2 n + 16$ | $8n + \log_2 n + 12$ |
| [20] - CI | $56n + 22 + A_{BC}$ | $16n + 4 + \tau_{BC}$ |
| [20] - CII | $96n + 24 + A_{BC}$ | $4n + 4 + \tau_{BC}$ |
| [20] - CIII | $80n + 18 + A_{BC}$ | $4n + 4 + \tau_{BC}$ |
| Proposed method | $76n + (33/2)n \log_2 n$ | $6\log_2 n + 23$ |

The most effective overflow detection circuit based on reverse converters can be built on the base on Converter I from [20]. In Converter I and also Converters II and III from [20], the minimum delay is $O(n)$ whereas, delay of proposed method is factor of $O(\log_2 n)$.

As seen from Table 3, the proposed approach for overflow detection in moduli set $\{2^n\text{-}1, 2^n, 2^n\text{+}1\}$ is faster than previous works. However, the hardware cost of the presented method is more. It is essential to remark that, although the proposed design consumes more hardware but it demonstrates significant improvement in terms of delay, especially for large $n$. Furthermore, our proposed method detects overflow without applying a complete comparator or reverse converter.

## 5. Conclusions

Detecting overflow is one of the most important and complex operations in residue number system. In this paper, a novel and different method has been presented for detecting overflow in moduli set $\{2^n\text{-}1, 2^n, 2^n\text{+}1\}$. Our proposed technique is based on group of numbers which leads to the correct result without doing a complete comparison or need to use the residue to binary converter. The presented approach has significant reduction in delay, compared to other methods.

## References

[1] T. Tomczak, "Fast Sign Detection for RNS $\{2^n\text{-}1, 2^n, 2^n\text{+}1\}$", IEEE Transactions on Circuits and Systems I: Regular Papers, Vol. 55, Iss. 6, 2008, pp. 1502-1511.

[2] N. S. Szabo and R. I. Tanaka, Residue Arithmetic and Its Application to Computer Technology, New York: McGraw-Hill, 1967.

[3] W. A. Chren, Jr. "A new residue number system division algorithm", Comput, Math. Appl., Vol. 19, No. 7, 1990, pp. 13-29.

[4] H. Bronnimann, I. Z. Emiris, V. Y. Pan, and S. Pion, "Computing exact geometric predicates using modular arithmetic with single precision", in Proc. 13th Annu. Symp. Comput. Geom., ACM press , 1997.

[5] R. C. Debnath and D. A. Pucknell, "On Multiplicative Overflow detection in Residue Number System", Electronics Letters, Vol. 14, No. 5, 1978

[6] R. Conway and J. Nelson, "Improved RNS FIR Filter Architectures", IEEE Trans. On Circuits and Systems-II: Express Briefs, Vol. 51, No.1, 2004.

[7] P. G. Fernandez and et al., "A RNS-Based Matrix-Vector-Multiply FCT Architecture for DCT Computation", Proc, 43th IEEE Midwest Symposium on circuits and Systems 2000, pp. 350-353.

[8] L. Yang and L. Hanzo, "Redundant Residue number System Based ERROR Correction Codes", IEEE VTS 54th on Vehicular Technology Conference, 2001, Vol. 3, pp. 1472-1467.

[9] J. Ramirez, et al., "Fast RNS FPL-Based Communication", Proc. 12th Int'l Conf. Field Programmable Logic, 2002, pp. 472-481.

[10] R. Rivest, A. Shamir, and L.Adleman, "A Method for obtaining Digital Signatures and Public Key Cryptosystems", Comm. ACM, Vol. 21, No. 2, 1948, pp. 120-126.

[11] J. Bajard, and L. Imbert, "A Full RNS Implementation of RSA", IEEE Transactions on computers, Vol. 53, No. 6, 2004, pp. 769-774.

[12] B. Parhami, "Computer arithmetic: algorithms and hardware designs", New York : Oxford University Press, 2000.

[13] M. Askarzadeh, M. Hosseinzadeh and K. Navi, "A New approach to overflow detection in moduli set $\{2^n\text{-}3, 2^n\text{-}1, 2^n\text{+}1, 2^n\text{+}3\}$", Second International Conference on Computer and Electrical Engineering, 2009, pp. 439-442.

[14] M. shang, H. JianHao, Z. Lin and L. Xiang, "An efficient RNS parity checker for moduli set $\{2^n\text{-}1, 2^n\text{+}1, 2^{2n}\text{+}1\}$ and its applications", Springer Journal of Science in China Series F: Information Sciences", Vol. 51, No. 10, 2008, pp. 1563-1571.

[15] A. Omondi and B. Premkumar, "Residue Number Systems: Theory and Implementation", Imperial College Press, 2007.

[16] M. Rouhifar, M. Hosseinzadeh and M. Teshnehlab, "A new approach to Overflow detection in moduli set $\{2^n-1, 2^n, 2^n+1\}$", International Journal of Computational Intelligence and Information Security, Vol. 2, No.3, 2011, pp. 35-43.
[17] E. Gholami, R. Farshidi, M. Hosseinzadeh and K. Navi, "High speed residue number system comparison for the moduli set $\{2^n-1, 2^n, 2^n+1\}$", Journal of communication and computer, Vol. 6, No. 3, 2009, pp. 40-46.
[18] R. Zimmerman, "Efficient VLSI implementation of modulo $(2^n\pm1)$ addition and multiplication", in Proc. 14th IEEE Symp. Comput. Arithm. , 1999, pp. 158-167.
[19] BI. Shao-quiang and W. J. Groos, "Efficient residue comparison algorithm for general Moduli sets", IEEE International Circuits and Systems, 2005, pp. 1601-1604.
[20] Y. Wang, X. Song, M. Aboulhamid and H. Shen, "Adder based residue to binary converters for $\{2^n-1, 2^n, 2^n+1\}$", 2002, pp. 1772-1779.

**Mehrin Rouhifar** received her B.Sc. in Computer Software Engineering from Islamic Azad University, Shabestar branch, Iran in 2008. Recently, she is received the M.Sc. degree in Computer System Architecture from Islamic Azad University, Tabriz branch, Iran in 2011. Her main research interests include Computer Arithmetic, Residue Number System, VLSI Design and Network reliability.

**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. degrees 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.

**Saeid Bahanfar** received the B.Sc. degree in Computer Software Engineering from Payam Noor University (PNU), Tabriz branch, Iran in 2008. Currently, he is a M.Sc. student of Computer System Architecture in Islamic Azad University, Tabriz branch, Iran. His research interests include Residue Number System and VLSI Design.

**Mohammad Teshnehlab** is professor at Department of Control Engineering, Faculty of Electrical Engineering, K. N. Toosi University, Tehran, Iran. His current research interests include Fuzzy, Neural Network, Soft Computing, Evolutionary Filtering and Simultaneous Localization and Mapping.