Multiplier Design Incorporating Logarithmic Number System for Residue Number System in Binary Logic

Shalini R. V[#1], Dr. P. Sampath[*2]
[#]*Research Scholar & ECE & Bannari Amman Institute of Technology, Sathyamangalam Tamil Nadu, India*
**Professor & ECE & Bannari Amman Institute of Technology, Sathyamangalam Tamil Nadu, India*

Abstract

Residue Number System (RNS) incorporates several significant features that are indispensible in Digital Signal Processing (DSP) applications. It includes higher operational speed, secured processing of data, carry free operations that reduces propagation of error among modules and so on. Multiplication process is the vital part of several DSP functions and hence design of such process using RNS system is gaining potential. For further improving the processing speed and security level of RNS, Multilevel-Residue Number System (MRNS) is introduced. This paper deals about the implementation of Logarithmic Number System (LNS) in RNS to propose the multiplication design based on Residue Logarithmic Number System (RLNS). Multilevel-Residue Number System (M-RNS) is incorporated in this research work introducing Multilevel-Residue Logarithmic Number System (M-RLNS) based multiplier design. Use of logarithmic numbers are restricted on accuracy constraints, hence improvement in accuracy is realized by employing error correction circuits. Area, Total Power Dissipation (TPD), delay and Power Delay Product (PDP) of the multiplication design proposed are tabulated for number of bits, N=8, 16 and 32 and the same is compared with the existing design.

Keywords - *Residue Number System (RNS), Logarithmic Number System (LNS), Multilevel-Residue Number System (M-RNS), Multilevel-Residue Logarithmic Number System (M-RLNS), Error correction circuits.*

I. INTRODUCTION

RNS involves in reducing the longer length input operands to shorter length modulo values. This helps in producing high speed processing aspect in the system where it is involved. Thus the scope of RNS is widened for filter design [1-4], cryptography [5-7] and several Digital Image Processing applications [8-10]. RNS involves in converting the binary weighted value into its residues and vice versa with the predefined moduli set. Therefore the intermediate residue values obtained cannot be processed further without knowing the exact moduli set values.

DSP applications generally deal with consecutive multiplication and addition operations, therefore designing with reduced computational complexity is essential. Introducing LNS into RNS proves to produce more compressed architectures compared to those designs including RNS features alone [11-13]. The combination of these unusual number systems is proposed by Arnold [14] represented as Residue Logarithmic Number System (RLNS). As addition and subtraction in LNS are difficult compared to multiplication and division operation, more research works are published for the efficient design of the former [15].

The literature survey depicts the use of RLNS technique with the operands of format b^q [16, 17], where b and q are integers >1. But for the operands which are not in exact power of logarithmic base (b), the multiplier design becomes complex and produces inaccurate results. It is due to logarithmic and antilogarithmic approximations made during the corresponding conversion process [18]. LNS is avoided due to this accuracy constraint, though it can provide promising results in terms of hardware utilization and power dissipation values [11]. In this paper the multiplication process with Residue Logarithmic Number System (RLNS) technique is implemented for all numbers with no consideration about the number format mentioned above.

II. MATHEMATICAL OPERATION OF RNS AND LNS

The three major processes involved in RNS are forward conversion, residue arithmetic unit and reverse conversion. In forward conversion process the input operand is converted into its corresponding residues. The integer number representation based on RNS is defined by a set of 'Q' relatively prime integers or moduli set given by $\{m_{1L}, m_{2L} \dots \dots \dots, m_{QL}\}$. The suffix variable '$L$' denotes the type of logic used in the RNS processing which is represented as 'b' or 't' for binary or ternary logic based circuits. Relatively prime integers taken as moduli set values is given by, gcd (m_{iL}, m_{jL}) = 1 for $i \neq j$. The weighted input operand is denoted as,

$X_L = (x_{1L}, x_{2L}, \ldots, x_{QL})$ where the value of x_{iL} is calculated by the expression,

$$x_{iL} = X_L \bmod m_{iL} = |X|_{m_{iL}} \quad 0 \le x_{iL} \le m_{iL} \tag{1}$$

The residue computation is limited for any integer X_L, given the range $(0, M_L]$, where M_L is the dynamic range given as $M_L = m_{1L} \times m_{2L} \times \ldots\ldots \times m_{QL}$. The arithmetic operations such as addition, subtraction, multiplication, division, exponentiation and squaring values can be computed by RNS in parallel channels [1]. The carry free propagation across the channels accounts for high speed computation in RNS. Let T_L denotes the required computation to be carried out, then $T_L = X_L \circ Y_L$, where $\circ$ may be any of the operations mentioned above. Thus the corresponding residues of the final result can be represented as

$$(t_{1L}, t_{2L}, \ldots., t_{QL}) = \left(|x_{1L} \circ y_{1L}|_{m_{1L}}, |x_{2L} \circ y_{2L}|_{m_{2L}}, |x_{3L} \circ y_{3L}|_{m_{3L}}, \ldots, |x_{QL} \circ y_{QL}|_{m_{QL}}\right) \tag{2}$$

The value t_{iL} is calculated from x_{iL} and y_{iL} in a modulo channel with the corresponding modulus value given by m_{iL} , i = 1,2,…..,Q. The residue values of a specific operation has to be converted back to its corresponding weighted number. This process is done by the reverse conversion method. The algorithm for the reverse conversion process is primarily based on the Chinese Remainder Theorem (CRT) [19-21], Mixed Radix Conversion (MRC) [21] and New Chinese Remainder Theorem (New CRT) [22]. In this research work smaller moduli set values are chosen, hence CRT is more suitable compared with other reverse conversion processes.

The logarithmic value of an integer ‘a’ is given as,

$$a \xrightarrow{LNS} \{s, \log_L |a|\} \tag{3}$$

Where ‘s’ denotes the sign of ‘a’ and ‘L’ represents the logarithmic base value that can be ‘2’ (binary logic) or ‘3’(ternary logic) based on the logic used.

$$s = \begin{cases} 0 \text{ if } a > 0 \\ 1 \text{ if } a < 0 \end{cases} \tag{4}$$

In this work, analysis of only positive input operands is considered and the value of ‘s’ is always ‘0’. The multiplication and division operations using LNS on the operands say A and B is given by the following logarithmic rules,

$$\log_L (A \times B) = \log_L A + \log_L B \tag{5}$$

$$\log_L (A / B) = \log_L A - \log_L B \tag{6}$$

III. THE PROPOSED MULTIPLIER DESIGN FOR RLNS BASED SYSTEM

The algorithm of RLNS based multiplication is as follows. Let the weighted input operands be represented as A_b and B_b . The multiplication proposed for RLNS based design using binary logic for input operands with number of bits, N_b = 8, 16 and 32 produces the output with number of bits, 16, 32 and 64 respectively. The block diagram of the proposed N_b bit multiplication process for RLNS based is shown in Fig. 1.

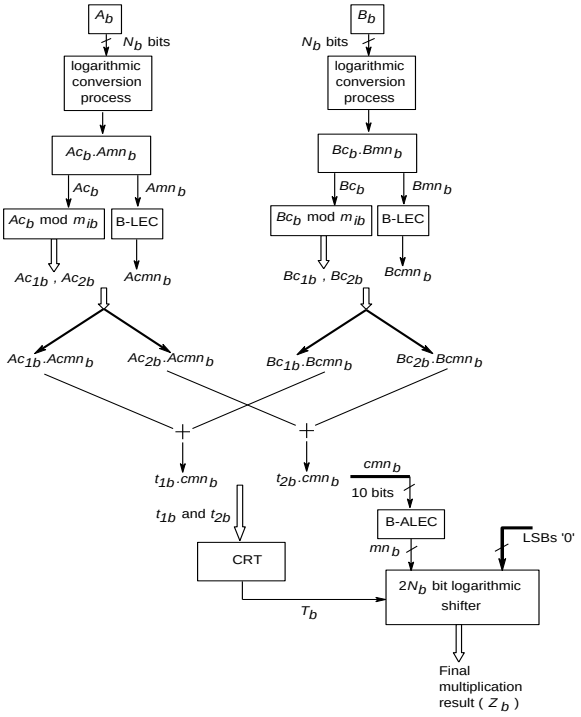

Fig 1: Binary logic based multiplication process for RLNS based system

A. *First step of RLNS*

The first step of the proposed technique involves in the logarithmic conversion of the input operands A_b and B_b producing the corresponding characteristics (Ac_b and Bc_b) and mantissa values (Amn_b and Bmn_b) respectively. Logarithmic conversion process is accomplished by the Leading

One Detector circuit (LOD), $N_b \times \log_2 N_b$ bit MOS ROM structure and N_b bit logarithmic shifter [23]. The logarithmic conversion of the input operand is shown in Fig. 2.

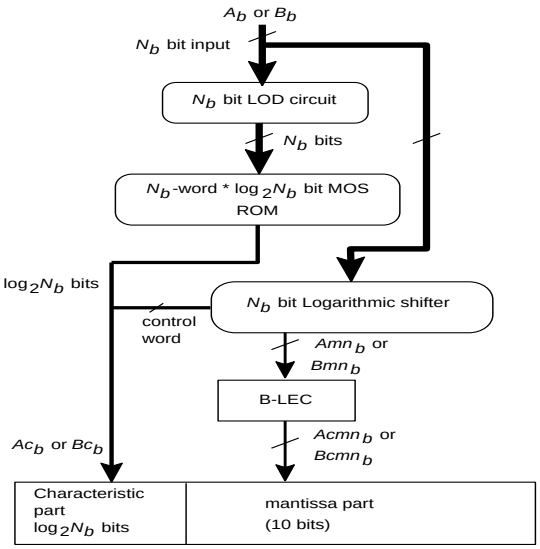

Fig 2 : Logarithmic conversion process

The mantissa value is obtained from the N_b bit logarithmic shifter. The mantissa values can be approximated using several error correction techniques. The existing research works for reducing the error value can be summarized as LUT based approach [24, 25], improving the accuracy of Mitchell's approach using correction term based [18], Linear Approximation [26-29] etc. LUT based error correction method involves in storage of data which expands with the increase in number of bits of input operands [30-32]. As linear approximation provides reduced hardware implementation compared with LUT based approach, this method is used in this research work to reduce error value. Dividing the mantissa interval, m ($0 \le m < 1$) to 2, 4 or more improves the accuracy of the logarithmic value [26, 29]. In this research work, the mantissa region is divided into eight equal intervals where each interval is estimated by a straight-line equation $y=a_j x + b_j$, where 'j'= 1 to 8, a and b are constant values chosen randomly based on several trial and error method.

The procedure explaining the logarithmic error correction for the mantissa values (Amn_b and Bmn_b) is explained as follows.

1. Binary – logarithmic error correction (B-LEC) circuit

The procedure of logarithmic conversion is initially proposed by Mitchell Jr 1962. Let B be the binary number in the interval $2^{k+1} > B \ge 2^j$, where j = 0, ±1, ±2……; k = 0, ±1, ±2… and $k \ge j$. B can be represented as,

$$B = \sum_{i=1}^{k} 2^i z_{ib} \tag{7}$$

z_{ib} is either '0' or '1' as the design is based on binary logic. Let z_{kb} denote the Most Significant Bit (MSB) and is assumed as z_{kb} = 1. If 2^k is factored out as per Mitchells approximation, the value of B becomes,

$$B = 2^k \left(1 + \sum_{i=1}^{k-1} 2^{i-k} z_{ib}\right) \tag{8}$$

Let the term $B = \sum_{i=1}^{k} 2^{i-k} z_{ib}$ is < 1 be the mantissa part represented as 'm', then the equation (8) becomes,

$$B = 2^k (1 + m) \tag{9}$$

The actual value of binary logarithm is given as,

$$\log_2 B = k + \log_2 (1 + m) \tag{10}$$

Mitchell approximation for the logarithmic value of B is represented as $(\log_2 B)'$ and is given by,

$$(\log_2 B)' \approx k + m \tag{11}$$

The error value (E) in the approximation procedure followed by Mitchell is calculated from the equation given below,

$$E = \log_2 B - (\log_2 B)' \tag{12}$$

$$= \log_2 (1 + m) - (m) \tag{13}$$

The proposed error correction procedure following linear approximation technique is explained as follows.

The equations of the resulting piecewise linear approximations for $\log_2 (1 + m)$ are given below, where 'cm' denotes the corrected value and 'm' represents the actual mantissa input.

$$cm = m + \frac{13}{32} m, \quad for \ 0 \le m < \frac{1}{8} \tag{14}$$

$$cm = m + \frac{23}{64}m, \quad for \quad \frac{1}{8} \le m < \frac{1}{4} \tag{15}$$

$$cm = m + \frac{73}{1024}, \quad for \quad \frac{1}{4} \le m < \frac{3}{8} \tag{16}$$

$$cm = m + \frac{43}{512}, \quad for \quad \frac{3}{8} \le m < \frac{1}{2} \tag{17}$$

$$cm = m + \frac{1}{8}\overline{m}_{7MSB} + \frac{3}{128}, \quad for \quad \frac{1}{2} \le m < \frac{5}{8} \tag{18}$$

$$cm = m + \frac{1}{8}\overline{m}_{7MSB} + \frac{15}{512}, \quad for \quad \frac{5}{8} \le m < \frac{6}{8} \tag{19}$$

$$cm = m + \frac{17}{64}\overline{m}_{5MSB}, \quad for \frac{6}{8} \le m < \frac{7}{8} \tag{20}$$

$$cm = m + \frac{5}{32}\overline{m}_{5MSB}, \quad for \frac{7}{8} \le m < 1 \tag{21}$$

The expansion of the coefficient values are given below,

$$\frac{13}{32} = 2^{-2} + 2^{-3} + 2^{-5} \quad ; \quad \frac{23}{64} = 2^{-2} + 2^{-4} + 2^{-5} + 2^{-6} \quad ;$$

$$\frac{73}{1024} = 2^{-4} + 2^{-7} + 2^{-10} \; ; \; \frac{43}{512} = 2^{-4} + 2^{-6} + 2^{-8} + 2^{-9} \; ;$$

$$\frac{3}{128} = 2^{-6} + 2^{-8} \qquad \frac{15}{512} = 2^{-6} + 2^{-7} + 2^{-8} + 2^{-9} \quad ;$$

$$\frac{17}{64} = 2^{-2} + 2^{-6} \; ; \frac{5}{32} = 2^{-3} + 2^{-5} \; ;$$

paragraphs The value of $\overline{m}_{7MSB}$ and $\overline{m}_{5MSB}$ denote the inversion of first 7 and 5 Most Significant Bits (MSBs) of the mantissa part respectively. The error correction circuit proposed uses 10 MSBs of the mantissa part and produces 10 bit result. The proposed Binary- Logarithmic Error Correction (B-LEC) is shown in Fig. (3 – 6). Sequence of Full Adder (FA) and Half Adder (HA) circuits are used in the error correction circuit. The carry value generated at each adder is propagated to the consecutive adder from the LSB of the mantissa value (m_{-10}). The carry value from the most significant mantissa bit value (m_{-1}) is neglected, as it is always '0'.

The input mantissa bit is taken as m_{-j} with the corresponding output value as cm_{-j}, where the value of 'j' ranges from 1 to 10 representing the 10 bits of result of the correction circuit (B-LEC). The corrected logarithmic value obtained is used for the proposed multiplication.

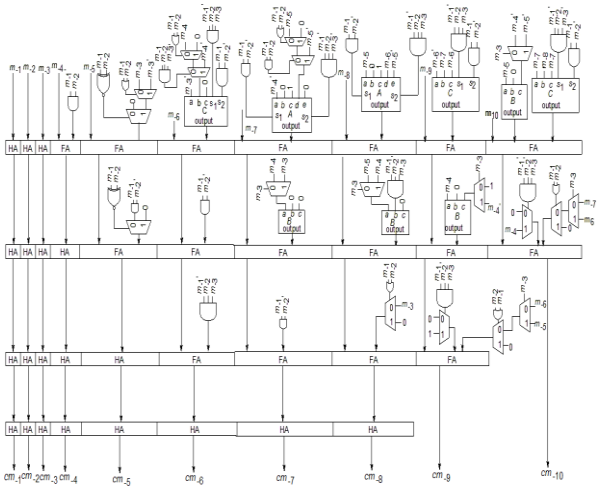

Fig 3: Binary-Logarithmic Error Correction (B-LEC) circuit

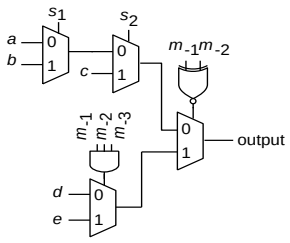

Fig 4 : Block A

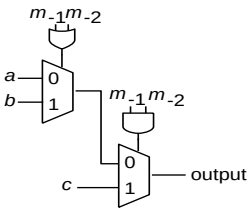

Fig 5 : Block B

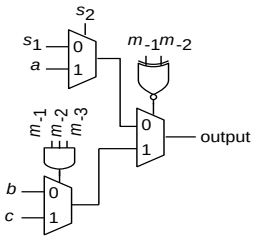

Fig 6 : Block C

The corrected mantissa values evaluated from the proposed B-LEC circuit (*cm*) are denoted as $Acmn_b$ and $Bcmn_b$ respectively in Fig. 1.

After the logarithmic conversion, the steps of typical RNS processing are followed. Based on the application of the multiplier design for a RLNS based system the moduli set values are selected. The dynamic range provided by the moduli set values chosen must cover the possible output values of the multiplication operation. The moduli set values chosen for the proposed binary logic based RLNS design is $\{2^{N_b}-1, 2^{N_b}+1\}$. Substituting $N_b = 3$ the values of the moduli set is given by, $\{m_{1b}, m_{2b}\} = \{7, 9\}$. The dynamic range value provided by these values is given by $[0, 2^{M_b}-1]$, where $M_b = 7 \times 9 = 63$. This moduli set values cover the multiplication result for input operands with bit length, $N_b = 8$, 16 and 32.

In forward conversion process, the characteristic values Ac_b and Bc_b are converted into its corresponding residues with respect to the moduli set {7, 9} by direct conversion method [33, 34] and is given by,

$$Ac_{ib} = Ac_b \bmod m_{ib} \tag{22}$$

$$Bc_{ib} = Bc_b \bmod m_{ib} \tag{23}$$

Two moduli values are taken for the process therefore '*i*' takes the value of 1 and 2 respectively.

B. Second step of RLNS

As multiplication operation is the idea of research, the residues are calculated based on the logarithmic rule in equation (5). The residue arithmetic unit is the second step of RLNS processing. The addition operation on the residues along with the corrected mantissa part is performed by ripple carry addition, where '+' in Fig. 1 denotes the addition operation. As area and power efficient design is the main aim of this research work, ripple carry addition method is used. The modulo operation on the added result is given by the equations (24) and (25).

$$t_{1b}.cmn_b = \left\langle Ac_{1b}.Acmn_b + Bc_{1b}.Bcmn_b \right\rangle_{m_{1b}} \tag{24}$$

$$t_{2b}.cmn_b = \left\langle Ac_{2b}.Acmn_b + Bc_{2b}.Bcmn_b \right\rangle_{m_{2b}} \tag{25}$$

The bracket $\langle\ \rangle$ represents the modulo operation performed with the corresponding modulo value

The added operands are represented as $t_{1b}.cmn_b$ and $t_{2b}.cmn_b$ in equations (24) and (25), where t_{ib} is the characteristic part and cmn_b the added mantissa value.

C. THIRD STEP OF RLNS

The third step includes the reverse conversion and antilogarithmic conversion. The result of reverse conversion is given as the control input to the antilogarithmic conversion.

The entire process of RLNS is carried out using small range of logarithmic values and hence the reverse conversion operation is done by Chinese Remainder Theorem (CRT). CRT method of reverse conversion process is explained using the following equations.

$$T_b = \left\langle \sum_{i=1}^{Q} \left\langle t_{ib} \times N_{ib} \right\rangle_{mib} \times M_{ib} \right\rangle_{M_b} \tag{26}$$

$$M_{ib} = \frac{M_b}{m_{ib}} \tag{27}$$

$$N_{ib} = \left\langle M_{ib}^{-1} \right\rangle_{m_{ib}} \tag{28}$$

T_b is the reverse conversion result calculated from the residue values t_{1b} and t_{2b} obtained from equations (18) and (19). N_{ib} is the multiplicative inverse of M_{ib} modulus m_{ib}. The dynamic range M_b is calculated from the equation given below,

$$M_b = \prod_{i=1}^{Q} m_{ib} \tag{29}$$

Where $i = 1,2,...,Q$ denotes the number of moduli set values chosen. From the moduli set values chosen, the constant values M_b, N_{1b}, N_{2b}, M_{1b} and M_{2b} are calculated as per the equations (20) – (23) and are 63_{10}, 4_{10}, 4_{10}, 9_{10} and 7_{10} respectively.

Antilogarithmic conversion includes the antilogarithmic error correction circuit and the logarithmic shifter. Several works are reported in literature to design antilogarithmic converters with or without using ROM, LUT etc., [34 and 35]. Limiting the deviations of the result from the actual antilogarithmic curve by linear approximations, the proposed method using 12 MSBs of the mantissa part shows considerable increase in the accuracy of the final antilogarithmic value when compared with the existing work [34]. In the proposed error correction method, improvement is made in the piecewise linear approximation procedure by designing 8-region correction circuit using 10 MSBs of the mantissa part.

The added mantissa value (cmn_b) is corrected by the error correction procedures proposed and is denoted as mn_b in Fig. 1.

1.Binary-antilogarithmic error correction (B-ALEC) circuit

Let the characteristic part of the logarithmic value be '*k*' and the mantissa part be '*m*'. The improvement of antilogarithmic approximation is as follows [18],

$$\text{Let } A = (\log_2 B)' = k + m \tag{30}$$

The antilogarithm of *A* is given as,

$$anti\log_2 A = 2^A \tag{31}$$

$$2^A = 2^{k+m} = 2^k.2^m \tag{32}$$

The antilogarithmic approximation is given as,

$$(anti\log_2 A)' = 2^k(1+m) \tag{33}$$

The error value (E) due to the above approximation is,

$$E = 2^k\left[2^m - (1+m)\right] \tag{34}$$

The proposed 8-region antilogarithmic approximation equations with the approximated value denoted as $(2^m)'$ are given below,

$$(2^m)' = m + \frac{7}{8}(c1 + c2 + c3) + \frac{51}{512}(c1) + \frac{1}{128}(c2) + \frac{1}{64}(c3) + \frac{7}{1024}(c4),\ for\ 0 \le m < \frac{1}{8} \tag{35}$$

$$(2^m)' = m + \frac{15}{16} + 2^{-6} + 2^{-8} + 2^{-9} + 2^{-10},\ for\ \frac{1}{8} \le m < \frac{1}{4} \tag{36}$$

$$(2^m)' = m + \frac{29}{32} + 2^{-6} + 2^{-7},\ for\ \frac{1}{4} \le m < \frac{3}{8} \tag{37}$$

$$(2^m)' = m + \frac{29}{32} + 2^{-7} + 2^{-9},\ for\ \frac{3}{8} \le m < \frac{4}{8} \tag{38}$$

$$(2^m)' = m + \frac{29}{32} + 2^{-7} + 2^{-8} + 2^{-10},\ for\ \frac{4}{8} \le m < \frac{5}{8} \tag{39}$$

$$(2^m)' = m + \frac{29}{32} + 2^{-6} + 2^{-9} + 2^{-10},\ for\ \frac{5}{8} \le m < \frac{6}{8} \tag{40}$$

$$(2^m)' = m + \frac{15}{16} + 2^{-8} + 2^{-10},\ for\ \frac{6}{8} \le m < \frac{7}{8} \tag{41}$$

$$(2^m)' = m + \frac{15}{16} + 2^{-6} + 2^{-7} + 2^{-8} + 2^{-9} + 2^{-10},\ for\ \frac{7}{8} \le m < 1 \tag{42}$$

The expansion of the coefficient values used are given as,

$$\frac{7}{8} = 2^{-1} + 2^{-2} + 2^{-3};\ \frac{15}{16} = 2^{-1} + 2^{-2} + 2^{-3} + 2^{-4};$$

$$\frac{29}{32} = 2^{-1} + 2^{-2} + 2^{-3} + 2^{-5};$$

$$\frac{51}{512} = 2^{-4} + 2^{-5} + 2^{-8} + 2^{-9};\ \frac{7}{1024} = 2^{-8} + 2^{-9} + 2^{-10}$$

The block diagram of B-ALEC proposed is shown in Fig. 7, 8. 10 MSBs denoted by m_{-i}, are input to B-ALEC where *i* ranges from 1 to 10. B-LEC produces 11 bit approximated mantissa value $(2^m)'$ as output denoted by 'cm_{-j}', where '*j*' ranges from 0 to 10. 11 bit output is thus obtained by keeping the MSB of the output as '1' as shown in Fig. 7 and the circuit diagram of block D used in B-ALEC circuit is shown in Fig. 8.

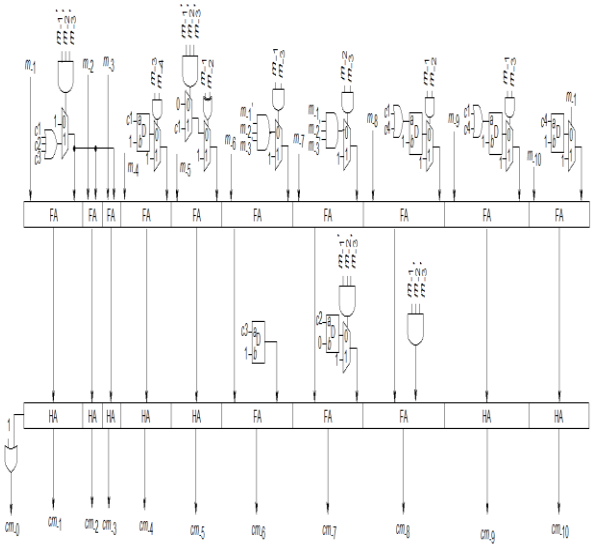

Fig 7 : Binary-Antilogarithmic Error Correction (B-ALEC) circuit

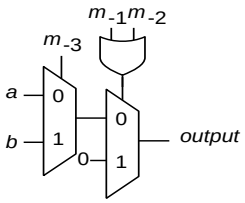

Fig 8 : Block D of B-ALEC circuit

Additional condition variables c_1, c_2, c_3, c_4 and c_5 are used for the error correction given by equation (29), for the mantissa part in the interval $0 \leq m <0.125$. At this region of mantissa part, the value of 1's that occur in the positions from m_{-4} to m_{-10} are taken for the error correction process as shown in equation (29). For the linear approximation in the region $0 \leq m <0.125$, the value of '1' present from the positions m_{-4} to m_{-10} are detected using LOD circuit. Based on the leading position the corresponding output am_{-j} , where 'j' ranges from 4 to 10 are obtained as shown in Fig. 9.

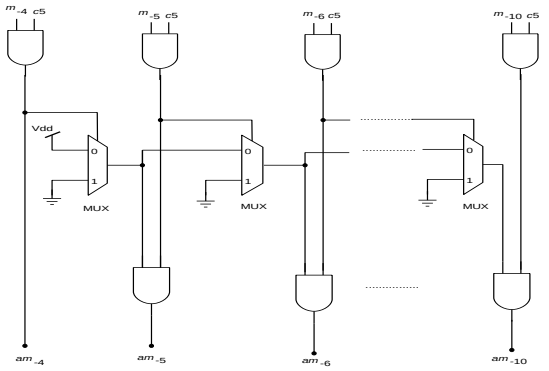

Fig 9 : Altered mantissa value (am_{-j})

Based on several trial and error method for the input operands in this interval, the conditional variables $c1$, $c2$, $c3$ and $c4$ are realized using equations (37) - (41). These equations are used to add the corresponding constant coefficients with the approximated mantissa value given in equation (36) to get corrected output in this region.

$$c1 = (am_{-4}) + (am_{-5}) + (am_{-6}) + (am_{-7}) \quad (43)$$

$$c2 = (am_{-5}) + (am_{-7}) \quad (44)$$

$$c3 = (am_{-6}) + (am_{-7}) \quad (45)$$

$$c4 = (am_{-8}) + (am_{-9}) + (am_{-10}) \quad (46)$$

$$c5 = \overline{m_{-1}} \times \overline{m_{-2}} \times \overline{m_{-3}} \quad (47)$$

Thus the corrected mantissa part obtained from the proposed B-ALEC circuit, denoted as mn_b in Fig. 1 is given as input to the $2N_b$ bit logarithmic shifter [34] of the antilogarithmic conversion. The antilogarithmic conversion process is shown in Fig. 10.

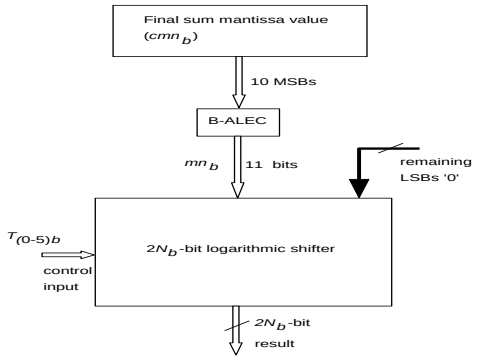

Fig 10 : Antilogarithmic conversion process

The logarithmic shifter provides the shifted result based on the value of the control word, ' T_b ' calculated by the equation (20). As the reverse conversion result is the output modulo 63, the maximum bit length is 6 as shown in Fig. 10. The control input decides the number of shifts required to get the final multiplication result denoted as ' Z_b ', in Fig. 1.

IV. PROPOSED BINARY LOGIC MULTIPLICATION PROCESS USING MRNS CONCEPT IN RLNS DESIGN

To improve the data encryption feature of the proposed RLNS based multiplication structure, MRNS concept is incorporated proposing a new idea of Multi-level Residue Logarithmic Number System (MRLNS). The feature of MRNS include choosing moduli set values at different levels of RNS until the residue values becomes simple. The condition in MRNS technique is to check the dynamic range of the moduli set values. The dynamic range in 1st level should be greater or equal to that of the next level [36,37], and the process is continued further until lower values of moduli set is achieved. Reducing the moduli values further reduces the complexity of the conversion circuits and its operations [36, 38].

In MRLNS based multiplier design, two level RNS is considered. As the characteristic values of smaller range are involved in RLNS, simple values of residues are obtained in the second level of MRLNS technique. In the proposed design of MRLNS, the moduli set chosen for the first level is given as $\{m_{1Ab}, m_{2Ab}\}$ = {7, 9}, substituting N_b = 3 in the set $\{2^{N_b}-1, 2^{N_b}+1\}$. This moduli set provides the dynamic range, [0, 2^{63}-1], enabling the comparison of

MRLNS design with the proposed single level RLNS scheme in terms of dynamic range. The second level of the moduli set chosen is $\{m_{1Bb}, m_{2Bb}\} = \{2^{N_b}, 2^{N_b}+1\}$, given $N_b = 2$ the values are {4, 5}, which is lesser than the previous moduli set values. The reverse conversion is carried out in the reversed order of moduli set values chosen i.e., first and the second level reverse conversion process use second and the first level moduli set values respectively. Therefore the first level moduli set values decides the dynamic range of the multiplier design. The multiplication process using MRLNS is shown in Fig. 11.

The binary input operands X_b and Y_b with N_b number of bits are initially converted into its logarithmic format with its characteristic part Xc_b and Yc_b and its mantissa $mn1_b$ and $mn2_b$ respectively. The error value of the mantissa $mn1_b$ and $mn2_b$ is corrected by the error correction circuit (B-LEC), that gives the results $cmn1_b$ and $cmn2_b$ respectively. The corrected mantissa values are added to give cmn_b which along with the residue values are taken for the antilogarithmic conversion process.

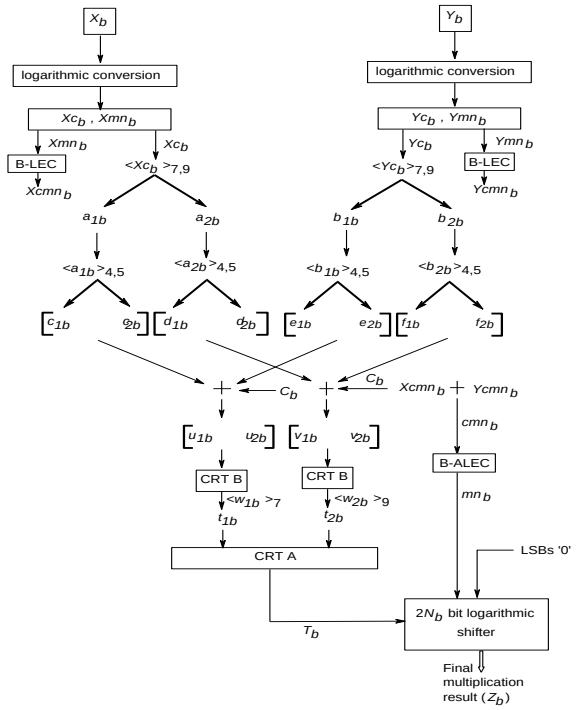

Fig 11 :Multiplication process using Multi-level Residue Logarithmic Number System (MRLNS) technique

During the first level of MRLNS the characteristic part of the operands are converted into its corresponding residues with respect to the moduli set {7, 9}. This residue values obtained are denoted as a_{ib} and b_{ib} in Fig. 11, where '*i*' takes the value of 1 and 2, as the number of moduli chosen is 2. The variables c_{ib}, d_{ib}, e_{ib} and f_{ib} represent the second four set of residue values obtained for the moduli set {4, 5} from a_{ib} and b_{ib} respectively. As the number of second set moduli values are also 2, '*i*' takes the value of 1 and 2 in the second level also. As the required arithmetic process is multiplication, the logarithmic values are added using the equations below, to give final two set of residues.

$$u_{ib} = \left\langle c_{ib} + e_{ib} + C_b \right\rangle_{m_{iBb}} \tag{48}$$

$$v_{ib} = \left\langle d_{ib} + f_{ib} + C_b \right\rangle_{m_{iBb}} \tag{49}$$

C_b is the carry value obtained from the addition of the mantissa values $cmn1_b$ and $cmn2_b$ respectively. The set of residue values u_{ib} and v_{ib} are given as input to the first level of reverse conversion. The reverse conversion process is done using CRT method, to differentiate for each level it is taken as CRT B in first level and CRT A for the second level, as shown in Fig. 6. The corresponding dynamic range, moduli values and the variables N_1, N_2, M_1 and M_2 that are required according to the CRT method are calculated. At each level of reverse conversion the corresponding variables calculated are represented with suffix *B* and *A* respectively based on the moduli set values utilized at each level. The use of lower case suffix '*b*' denotes the binary logic. The operation to calculate the values w_{1b} and w_{2b} denoted as CRT B blocks in Fig. 11 is given below,

$$w_{1b} = \left\langle \begin{matrix} (\left\langle u_{1b} \times N_{1Bb} \right\rangle_{m_{1Bb}} \times M_{1Bb}) + \\ (\left\langle u_{2b} \times N_{2Bb} \right\rangle_{m_{2Bb}} \times M_{2Bb}) \end{matrix} \right\rangle_{M_{Bb}} \tag{50}$$

$$w_{2b} = \left\langle \begin{matrix} (\left\langle v_{1b} \times N_{1Bb} \right\rangle_{m_{1Bb}} \times M_{1Bb}) + \\ (\left\langle v_{2b} \times N_{2Bb} \right\rangle_{m_{2Bb}} \times M_{2Bb}) \end{matrix} \right\rangle_{M_{Bb}} \tag{51}$$

M_{Bb} value used in the equations (51 and 52) denote the dynamic range provided by the second level moduli set, {4, 5}. The value of M_{Bb}, is calculated from the product of 4 and 5, $M_{Bb} = 20$. The values N_{iBb} and M_{iBb} are calculated from {4, 5} using equations (12 -14) and the calculated values are $N_{1Bb} = 1$, $N_{2Bb} = 4$, $M_{1Bb} = 5$ and $M_{2Bb} = 4$. From the values of w_{ib}, the second set of residues, t_{ib} is

calculated from the modulo operation done by the values { m_{1Ab}, m_{2Ab} } = {7, 9} using the equations (52) and (53).

$$t_{1b} = \langle w_{1b} \rangle_{m_{1Ab}} \tag{52}$$

$$t_{2b} = \langle w_{2b} \rangle_{m_{2Ab}} \tag{53}$$

The calculated values of t_{1b} and t_{2b} are given as input to the second level of reverse conversion (CRT A). The reverse conversion result, T_b is taken as output modulo of M_{Ab} = 63 in this level (CRT A block) as shown in Fig. 11. The corresponding reverse conversion operation is as follows,

$$T_b = \left\langle \begin{matrix} (\langle t_{1b} \times N_{1Ab} \rangle_{m_{1Ab}} \times M_{1Ab}) + \\ (\langle t_{2b} \times N_{2Ab} \rangle_{m_{2Ab}} \times M_{2Ab}) \end{matrix} \right\rangle_{M_{Ab}} \tag{54}$$

This ' T_b ' value along with the added mantissa part ' cmn_b ' is input for the antilogarithmic conversion process. The mantissa part ' cmn_b ' is corrected by the proposed Binary- Antilogarithmic Error Correction Circuits (B-ALEC) and is denoted as ' mn_b ' in Fig. 11. 'mn_b' is given as input to the $2N_b$ bit logarithmic shifter.' T_b ' controls the shifting operation of the given mantissa input in the shifter providing $2N_b$ bit final multiplication result represented as ' Z_b '.

V. SIMULATION RESULTS OBTAINED FOR THE PROPOSED DESIGN AND ITS COMPARISON WITH THE EXISTING TECHNIQUES

Simulation of the circuits are made using Cadence tool, Virtuoso 6.1.5 with 45nm TSMC CMOS technology and supply voltage of 0.5 V. For logic states of '1' and '0', 0.5 V and 0 V are used respectively. The design of the proposed and existing research works are made for N_b = 8, 16 and 32 and the simulation results are compared. The area occupied, Total Power Dissipation (TDP) value, delay and PDP values of the proposed multiplication structure (RLNS and MRLNS) are compared with the existing work [40] in Table 1. To prove the efficiency of the proposed design for a RLNS based system, the existing work of modulo multiplier design using Radix-8 booth encoding technique for a RNS based system is designed with the same TSMC 45nm technology file using Cadence tool.

The simulation results including area, TPD, delay and PDP of the existing work are given in Table 1.

Table 1 Comparison of Area, Power and Delay values

Multiplication structure		Area (μm^2)	Total Power Dissipation (TPD) (μW)	Delay (*ns*)	Power Delay Product (PDP) (*Joules*)
Technique used	Number of bits, N				
RLNS	8	41094	2.918	78	227.6
	16	49545	4.033	142	572.68
	32	60364	5.169	281	1452.48
MRLNS	8	47702	3.228	89	287.2
	16	52121	4.271	194	828.57
	32	66597	5.817	310	1803.27
Radix-8 booth encoding technique [40]	8	117676	15.97	130	2076.1
	16	164997	32.15	340	10931
	32	215781	53.25	567	30192.75

It is inferred from the values, the proposed multiplication design provide efficient results in terms of area, TPD, delay and PDP values. The area occupied, TPD and delay values of multiplication structure using MRLNS technique is 15% more compared to that of RLNS technique due to the additional forward and reverse conversion operations carried out in two levels. The percentage of

parameter values saved by the proposed techniques compared to the existing work is given in Table 2. From the percentage values it is inferred that 67.3% of area and 45% delay are saved whereas TPD and PDP are 86% and 92% less compared to that of existing work.

Table 2 Percentage of TPD, area, delay and PDP saved by the proposed techniques over existing method

Multiplication structure using	Number of bits, N_b	Percentage of parameter values saved over existing technique (%)			
		TPD	Area	Delay	PDP
RLNS	8	82	65	40	89.03
	16	87.4	70	58.2	95
	32	90.2	72	50.4	95
MRLNS	8	80	59	31.5	86
	16	87	68	43	92
	32	89.07	69	45.3	94

The advantage of using LNS eliminates the partial product generation and its accumulation process of the existing modulo multiplier designs, thereby reducing the TPD, area and delay values. As LNS involves only the addition of input operands the error reduction of logarithmic and antilogarithmic values, is done by the proposed B-LEC and B-ALEC circuits. The EP values are calculated for the random selection 250 set of input values for each N_b category chosen. The formula for calculating the EP includes the True Value (*TV*) of the multiplication result and the Experimental Value (*EV*) obtained from the proposed technique as given below,

$$EP = \frac{TV - EV}{TV} \times 100\% \quad (49)$$

The Average Error Percent (AEP) of the final multiplication values obtained with the existing works [29, 34] and proposed (B-LEC and B-ALEC) error correction circuits respectively for N_b = 8, 16 and 32 are given in Table 3. The AEP value is calculated using the formula given below,

$$AEP = \frac{\sum_{n=1}^{N} EP}{N} \quad (50)$$

N denotes the number of input set of operands considered and its value is 250. The AEP value obtained with the proposed B-LEC and B-ALEC circuits is 0.36 and for existing error correction circuit, it is 0.77. The Error Percentage (EP) calculated for the input operands includes both positive and negative error value. The value of EP is as obtained without considering the positive and negative range of the error value produced.

Table 3 Comparison of Average Error Percent (AEP) (%) obtained for the proposed multiplication design with existing error correction circuits

Average Error Percentage (AEP) of the results obtained with the proposed designs (RLNS and MRLNS)		
Number of bits (N_b)	With proposed B-LEC and B-ALEC	With existing error correction circuits [29, 34]
8	0.39	0.54
16	0.40	0.69
32	0.30	1.08

From the multiplication results it is observed that the proposed error correction circuits (B-LEC and B-ALEC) reduces the AEP to 48% when compared with the existing work [29, 34] for the proposed multiplier design for RLNS based system. The AEP values obtained for each N_b category is same for the proposed RLNS and MRLNS based multiplication designs as the correction is done in the mantissa part in the logarithmic and antilogarithmic conversion process. The difference in the RLNS and MRLNS structures is in processing the corresponding characteristic values to get two stages of residues. The residues transmitted with two levels of encryption prevents the misuse of data, as the two levels of moduli set values are known only to user [37, 39]. Thus MRLNS based design may be considered for secured applications.

VI. CONCLUSION

The multiplier design for RLNS based system is proposed in this research reducing the area, TPD and delay values when compared with the existing research. Using LNS in the design of multipliers simplifies the operation by avoiding the partial product reduction and accumulation as only addition of the input operands is performed. In order to reduce the error produced due to the approximation process using LNS, error correction circuits are proposed in this research to produce the final multiplication result with AEP = 0.36. MRLNS technique is proposed to include the secured features of MRNS that includes multilevel forward and reverse conversion processes.

Thus for the RLNS based Digital Signal Processing (DSP) application where the minimum error value produced is acceptable the proposed multiplier design may be considered.

REFERENCES

[1] Keivan Navi, Amir Sabbagh Molahosseini, and Mohammad Esmaeildoust, "How to Teach Residue Number System to Computer Scientists and Engineers", IEEE Transaction on Education, vol. 54, pp. 156-163, November 2011.

[2] W. K. Jenkins, and B. J. Leon, "The use of residue number systems in the design of finite impulse response digital filters", IEEE Trans. Circuits Syst., vol. 24, pp. 191–201, April 1977.

[3] R. Conway, and J. Nelson, "Improved RNS FIR filter architectures", IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 51, pp. 26–28, January 2004.

[4] J. Ramirez, U. Meyer-Base, and A. Garcia, "Efficient RNS-based design of programmable FIR filters targeting FPL technology", J. Circuits,Syst. Comput., vol. 14, pp. 165–17, Februrary 2005.

[5] K. C. Posch, and R. Posch, "Residue Number System: a key to parallelism in public key cryptography", in Proc. of the Fourth IEEE Symposium on Parallel and Distributed Processing, 1992, p. 432.

[6] D. M. Schinianakis , A. P. Kakarountas, and T. Stouraitis, "A novel approach to elliptic curve cryptography: an RNS architecture", in Proc. of IEEE Mediterranean Electrotechnical Conference (MELECON), 2006, p. 1241.

[7] Yinan Kong, and Yufeng Lai, "Low latency modular multiplication for public-key cryptosystems using a scalable array of parallel processing elements", in Proc. of IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS), 2013, p. 1039.

[8] Wang Wei, M. N. S. Swamy, and M. O. Ahmad, "RNS application for digital image processing", in Proc. 4th IEEE Int. Workshop System-on-Chip Real-Time, 2004, p. 77.

[9] G. C. Cardarilli, A. Nannarelli, and M. Re, "Residue number system for low-power DSP applications", in Proc. 41st IEEE Asilomar Conf. Signals, Syst., Comput., 2007, p. 1412.

[10] A. Ammar, et al, "A secure image coding using residue number system", in Proc. 18th Nat. Radio Sci. Conf. 2, 2001, p. 399.

[11] V. Paliouras, and T. Stouraitis, "Low-power properties of the logarithmic number system", in Proc. 17th IEEE Symposium on Computer Arithmetic, 2001, p. 229.

[12] R. K. Agrawal, and H. M. Kittur, "ASIC based logarithmic multiplier using iterative pipelined architecture", in Proc. IEEE conference on Information and Communication Technologies (ICT), 2013, p. 362.

[13] D. M. Lewis, "114 MFLOPS Logarithmic Number System Arithmetic Unit for DSP Applications", IEEE Journal of Solid-State Circuits, vol. 30, pp. 1547 – 1553, February 1995.

[14] M. G. Arnold, "The residue logarithmic number system: theory and implementation", in Proc 17th IEEE Symposium on Computer Arithmetic, 2005, p. 196.

[15] B. Lee, and N. Burgess, "A Dual-Path Logarithmic Number System Addition/Subtraction Scheme for FPGA", in Proc International Conference on Field Prog. Logic App., 2003, p. 808.

[16] A. Mousavi, and D. K. Taleshmekaeil, "Pipelined Residue Logarithmic Number System for general modules set {2^n-1, 2^n, 2^n+1}", in Proc. 5th Internaional conference on Computer Sciences and Convergence Information Technology, 2010, p.700.

[17] Davar Kheirandish Taleshmekaeil, and Alireza Mousavi, "Circuit design Residue Logarithmic Number System (RLNS) using the One-Hot system", Research Journal of Applied Sciences, Engineering and Technology, vol. 5, pp. 286-291, January 2013.

[18] J. N. Mitchell Jr, "Computer Multiplication and Division using Binary Logarithms", IEEE Transactions on Electronic Computers, vol. EC-11, pp. 512-517, August 1962.

[19] B. Parhami, Computer Arithmetic: Algorithms and Hardware Design, Oxford University Press, United Kingdom, 2000

[20] Arash Hariri , Keivan Navi & Reza Rastegar, "A new high dynamic range moduli set with efficient reverse converter", J. Comput. Math. Appl., vol. 55, pp. 660-668, Februrary 2008.

[21] 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}", IEEE transactions on Circuits and Systems I: Fundamental Theory and Applications, vol. 50, pp. 1296-1303, October 2003.

[22] B. Cao, C.H. Chang, and T. Srikanthan, "Adder based residue to binary converters for a new balanced 4-moduli set", in Proc. of the 3rd International Symposium on Image and Signal Processing and Analysis, 2003, p. 820.

[23] Khalid H Abed, and Raymond E Siferd, "CMOS VLSI implementation of 16-bit logarithm and anti-logarithm converter", in Proc. of the 43rd IEEE Midwest Symposium on Circuits and Systems, 2000, pp. 776.

[24] Davide De Caro, Nicola Petra, and Antonio GM Strollo, "Efficient logarithmic converters for digital signal processing applications", IEEE Transactions on Circuits and systems-II: express briefs, vol. 58, pp. 667-671, 2011.

[25] M. Arnold, T. Bailey, and J. Cowles, "Error analysis of the Krnetz/Maenner algorithm", Journal of VLSI signal processing, vol. 33, pp. 37-53, month 2003.

[26] M. Combet, H. Van Zonneveld, L. Verbeek, "Computation of the Base Two Logarithm of Binary Numbers", IEEE Transaction on Electronic Computers, vol. EC-14 , pp. 863-867, December 1965.

[27] E.L Hall, D. D. Lynch, and S. J. Dwyer, "Generation of Products and Quotients Using Approximate Binary Logarithms for Digital Filtering Applications", IEEE Transaction on Electronic Computers, vol. C-19 , pp. 97-105, February 1970.

[28] S. L. SanGregory, C. Brothers, D. Gallagher, and R. Siferd, "A Fast, Low-Power Logarithm Approximation with CMOS VLSI Implementation", in Proc. IEEE Midwest Symp. Circuits and Systems, 1999, pp. 388.

[29] Khalid H Abed, and Raymond E Siferd, "CMOS VLSI Implementation of a Low-Power Logarithmic Converter", IEEE Transactions on Computers, vol. 52 , pp. 1421-1433, 2003.

[30] T. A. Brubaker, J. C. Becker, "Multiplication using logarithms implemented with Read-only-memory", IEEE Transactions on computers, vol. 24, pp. 761-766, August 1975

[31] D. M. Lewis, "Interleaved memory function interpolators with application to accurate LNS arithmetic unit", IEEE Transactions on computers, vol. 43, pp. 974 – 982, August 1994

[32] F. S. Lai, C. F. E. Wu, "A Hybrid number system processor with geometric and complex arithmetic capabilities", IEEE Transactions on Computers, vol. 40, pp. 652-662, August 1991.

[33] Naamatheertham R Samhitha, Neethu Acha Cherian, Pretty Mariam Jacob, and P. Jayakrishnan, "Implementation of 16-bit floating point multiplier using Residue Number System", in Proc. International Conference on Green Computing, Communication and Conservation of Energy (ICGCE), 2013, p. 195.

[34] Khalid H Abed, and Raymond E Siferd, "VLSI Implementation of a Low-Power Antilogarithmic Converter", IEEE Transactions on Computers, vol. 52, pp. 1221-1228, September 2003.

[35] E. L. Hall, D. D. Lynch, and S. J. Dwyer, "Generation of Products and Quotients Using Approximate Binary Logarithms for Digital Filtering Applications", IEEE Transaction on Electronic Computers, vol. C-19 , pp. 97-105, February 1970.

[36] Mehdi Hosseinzadeh, Somayyeh Jafarali Jassbi, and Keivan Navi, "A New Moduli Set {3^n-1, 3^n+1, 3^n +2, 3^n - 2} in Residue Number System", in Proc. 10th International Conference on Advanced Communication Technology (ICACT), 2008, p. 1601.

[37] H. M. Yassine, "Hierarchical Residue Numbering System suitable for VLSI Arithmetic Architectures", in Proc. IEEE international symposium on Circuits and Systems, 1992, p. 811.

[38] M. Abdallah, and A. Skavantzos, "A systematic approach for selecting practical moduli sets for residue number systems", in Procs of the Twenty-Seventh Southeastern Symposium on System Theory, 1995, p. 445.

[39] Pazhar Ali , Mizanian Kambiz, Safi Seyyed Mohammad, Taghipour Eivazi Shiva, and Rezael Mehdi, "Fault-Tolerant and Information security in Networks using Multi-level Redundant Residue Number System", Research Journal of Recent Sciences, vol. 3, pp. 89-92, March 2014.

[40] Ramya Muralidharan, Chip-Hong Chang, "Area-Power Efficient Modulo 2^n-1 and Modulo 2^n+1 Multipliers for {2^n-1, 2^n, 2^n+1}", IEEE Transactions on Circuits and Systems I Regular papers, vol. 59, pp. 2263-2273, October 2012.