Circuits and Systems, 2016, 7, 417-433
Published Online April 2016 in SciRes. http://www.scirp.org/journal/cs
http://dx.doi.org/10.4236/cs.2016.74036

Scientific Research Publishing

Multiplier Design Utilizing Tri Valued Logic for RLNS Based DSP Applications

Shalini Radakirishnan Valliammal, Sampath Palaniswami

Electronics and Communication Engineering, Bannari Amman Institute of Technology, Erode, India
Email: shalugrv@gmail.com

Received 8 March 2016; accepted 25 April 2016; published 28 April 2016

Abstract

Residue Number System (RNS) has proved shaping the Digital Signal Processing (DSP) units into highly parallel, faster and secured entities. The computational complexity of the multiplication process for a RNS based design can be reduced by indulging Logarithmic Number System (LNS). The combination of these unusual number systems forms Residue Logarithmic Number System (RLNS) that provides simple internal architectures. Till date RLNS based processing units are designed for binary logic based circuits. In order to reduce the number of input output signals in a system, the concept of Multiple Valued Logic (MVL) is introduced in literature. In that course of research, this paper uses Tri Valued Logic (TVL) in RLNS technique proposed, to further reduce the chip area and delay value. Thus in this research work three different concepts are proposed, it includes the design of multiplier for RLNS based application for number of bits 8, 16 and 32. Next is the utilization of TVL in the proposed multiplication structure for RLNS based system along with the error correction circuits for the ternary logarithmic and antilogarithmic conversion process. Finally the comparison of the two multiplication schemes with the existing research of multiplier design for RNS based system using booth encoding concepts. It can be found that the proposed technique using TVL saves on an average of about 63% of area occupied and 97% of delay value respectively than the existing technique.

Keywords

Residue Number System (RNS), Residue Logarithmic Number System (RLNS), Tri Valued Logic (TVL), Binary Logic, Error Correction Circuits

1. Introduction

The significant feature of Residue Number System (RNS) has come out as an effective solution in implementing

several DSP applications as it reduces the word length of input operands to smaller length modulo values. This provides high speed processing feature for the system where it is involved. Several research works are found in literature implementing RNS for FIR, IIR filters, and Digital Image Processing applications [1]-[7]. Further an important application of RNS is in cryptography, providing highly secured transmission of data [8]-[11]. The typical process of RNS involves in converting the input binary weighted value to its residues and vice versa with the predefined moduli set. Therefore the intermediate residues obtained cannot be processed further without knowing the exact moduli set. RNS also has the advantage of dividing the operation block into distinct modules that are independent of the others. The total number of modules in RNS depends on the number of moduli set values chosen to perform the corresponding operation. The selection of the moduli set forms an important factor in deciding the efficient operation of this number system [12] [13].

As DSP applications generally deal with consecutive multiplication and addition operations, implementation of these designs with reduced computational complexity is more essential. To fulfill this criterion, the method of introducing LNS into RNS is proposed and it is proved to produce more compressed architectures compared to those designs including RNS features alone [14]-[16]. The combination of this number system is proposed by Arnold [17], represented as Residue Logarithmic Number System (RLNS). The history of literature works depicts the utilization of operands with format 2^p (binary logic) with the value of $p > 1$ for performing RLNS based addition and multiplication operation [18] [19]. This form of input operands in the multiplication operation provides accurate results with simple manipulation process using logarithmic property. This can be explained as follows, for the input operands of the format mentioned before, adding the exponents of operands gives the multiplication result once the antilogarithmic conversion procedure is completed. But if the input operand is not in the specific form given above *i.e*., any integer greater than 1, for example 3, 5, 6, 7, ⋯ etc., then the logarithmic conversion of the operand will not provide a whole number result. Thus to improve the accuracy of the mantissa part obtained, the logarithmic and antilogarithmic error correction circuits become mandatory.

From the literature survey, it has been observed that in two-level logic (binary logic), performance degradation occurs due to large area occupied for interconnection purpose [20]. It is broadly studied that in VLSI chip 70% of the area is consumed for interconnection, 20% to insulation and 10% to devices [21]. Therefore the concept of Multiple Value Logic (MVL) becomes indispensable in reducing the above problem by using larger value of signals over same area. The application of MVL for the design of several circuits and its advantages are proved in previous literature works [22]-[26]. The only limitation of using MVL is that the design techniques are little more complex than the binary logic circuits [27]. However the most efficient radix for implementing switching systems is claimed to be the natural base ($e = 2.71823$). This infers that the best integral radix is 3 rather than 2 [20] [28]. From the previously proposed works the design of basic ternary logic gates and its application in various circuits can be studied [29]-[32]. Also TVL based RNS system is proved to provide increased speed and reduced area consumption values [33] [34].

From all the key features discussed above, the design of multiplier for RLNS based applications using TVL is proposed in this work. This method is more effectual in reducing the computational complexity of the multiplier design as logarithmic property is involved. To further reduce the number of input output terminals, overall area occupied and the delay values of the circuit, TVL is incorporated. The challenging task for the design of proposed idea lie in the conversion of ternary inputs to its corresponding logarithmic values and vice versa. This forms the major contribution of this work, along with the error correction circuit of ternary logarithmic and antilogarithmic conversion process. As far as dealing with logarithmic numbers with base value 2 (binary logic), there are several procedures followed to ensure correction process for the logarithmic and antilogarithmic conversion circuits. They can summarized as Look Up Table (LUT) based approach [35] [36], improving the accuracy of Mitchell's approach [37] using correction term based, Divided approximation [38]-[41], Operand decomposition [42] [43] and so on. But in the case of TVL, Mitchell's approach cannot be applied, as the approaches do not provide even approximate results for all values in the conversion process. Also use of LUT is not appreciated as it involves in increasing the area utilization of the design. Hence different approach is followed for reducing the error value of the ternary logarithmic and antilogarithmic conversion process (discussed in Section 4)

The difficulty encountered in the logarithmic conversion of the ternary values can be explained as follows, in case of base 2 logarithm the value of $\log_2 14$, found by Mitchell's method [37] is explained below. For instance, $(14)_{10}$ is represented in binary values as $(1110)_2$, the corresponding Mitchell's approximated logarithmic value is $(11.110)_2$ and its decimal value is 3.75 against the actual value of 3.8073. Similarly if the same logic is applied

for ternary number $(112)_3$ (ternary value of 14), the ternary logarithmic value becomes $(2.12)_3$ with approximate decimal value $(2.5555)_{10}$ whose actual value is $(2.4021)_{10}$. The ternary logarithmic values seems to be closer, but the actual antilogarithmic value becomes 16.56 ($3^{2.5555}$) for 14. Hence suitable method for logarithmic and antilogarithmic conversion of ternary number has to be designed. The logarithmic conversion of the input ternary operands is then followed with the steps of RNS manipulation discussed in next section. The final result obtained from the reverse conversion process, as explained in the beginning is then converted back to its antilogarithmic format to get the original ternary weighted value. This TVL based design is then compared with the binary logic based multiplication process utilizing the concepts of RLNS procedure and also with the existing research work (binary logic) explained in later sections. This comparison provides the clear idea about the parameter values saved by the TVL based design over the other two utilizing binary logic.

The flow of this paper can be explained as follows, Section 2 deals with the mathematical background of the RNS and RLNS, Section 3 explains the block diagram of the entire multiplication scheme using RLNS, with ternary input values. The explanation on the error correction procedures adopted for logarithmic and antilogarithmic conversion process is given in Section 4. Simulation results obtained and its analysis made on the proposed design using TVL and binary logic along with the existing design for multiplication process using booth encoding techniques are dealt in Section 5. The entire research work and the key results are concluded in section 6.

2. Mathematical Background of RNS and RLNS

The three major operations involved in RNS are given as follows: Initially the conversion of the input operand into its corresponding residues is known as Forward conversion. The integer number representation based on RNS is defined by a set of Q relatively prime integers $\{m_1, m_2, \cdots, m_Q\}$, that can be quoted as the RNS base or moduli set. Relatively prime integers denoted is given by gcd (m_i, m_j) = 1 for $i \neq j$. The weighted input operand can be denoted as, $X = (x_1, x_2, \cdots, x_Q)$, where the value of x_i is given by,

$$x_i = X \bmod m_i = |X|_{m_i}, \quad 0 \le x_i \le m_i \tag{1}$$

This residue computation is exclusive for any integer X, given the range [0, M), whereas M is the dynamic range given as $M = m_1 \times m_2 \times \cdots \times m_Q$. The arithmetic operations such as addition, subtraction, multiplication, division, exponentiation and squaring values are computed by RNS in parallel channels. The carry free propagation across the channels accounts for high speed manipulation of RNS. Let T denotes the required computation to be carried out, then T = X ◦ Y, where ◦ be any of the arithmetic functions mentioned before. Thus the corresponding residues can be represented as

$$(t_1, t_2, \cdots, t_Q) = \left(|x_1 \circ y_1|_{m_1}, |x_2 \circ y_2|_{m_2}, |x_3 \circ y_3|_{m_3}, \cdots, |x_Q \circ y_Q|_{m_Q}\right) \tag{2}$$

where t_i is calculated from x_i and y_i in a modulo channel with the corresponding modulus value given by m_i, $i = 1, 2, \cdots, Q$. The result of the 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) [44]-[47], Mixed Radix Conversion (MRC) [47] and New Chinese Remainder Theorem (New CRT's) [48]. In this paper the reverse conversion operation is done by using CRT method, as smaller range of logarithmic values are dealt in the entire process, for which CRT method is more suitable compared with other procedures.

The combination of LNS and RNS by Arnold simplifies RNS based applications by enabling manipulation of smaller values of data due to logarithmic conversion of the operands. The algorithm of RLNS which inherits the logarithmic property in RNS is explained below. For instance, the logarithmic value of integer a is given by,

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

where "s" denotes the sign of "a" and the value of "b" is 3 or 2, based on the logic chosen.

As this paper includes the analysis of only positive input operands, the value of sign bit or trit (ternary digit) is always "0" and is not considered further. The multiplication and division operation that is performed in LNS on the operands say A and B is given by the following logarithmic rules,

$$\log_b(A \times B) = \log_b A + \log_b B \tag{4}$$

$$\log_b(A/B) = \log_b A - \log_b B \tag{5}$$

The RLNS scheme for the multiplication process using TVL proposed in this paper is explained below. Let the input operands in the ternary weighted form can be represented as X and Y. As the initial step of RLNS multiplication process, these values are converted into its logarithmic form providing its characteristic values Xc and Yc and mantissa values Xmn and Ymn respectively. The overall block explaining the proposed RLNS based multiplication process is shown in **Figure 1**. The mantissa values chosen are corrected by the error correction procedures proposed, discussed in Section 4 and are denoted as Xcmn and Ycmn. The moduli set for which the corresponding residues are obtained are given as $\{m_1, m_2\}$. After the logarithmic conversion process, the steps of RNS processing is continued as discussed before. During the forward conversion process, the characteristic values Xc and Yc are converted into its corresponding residues given by,

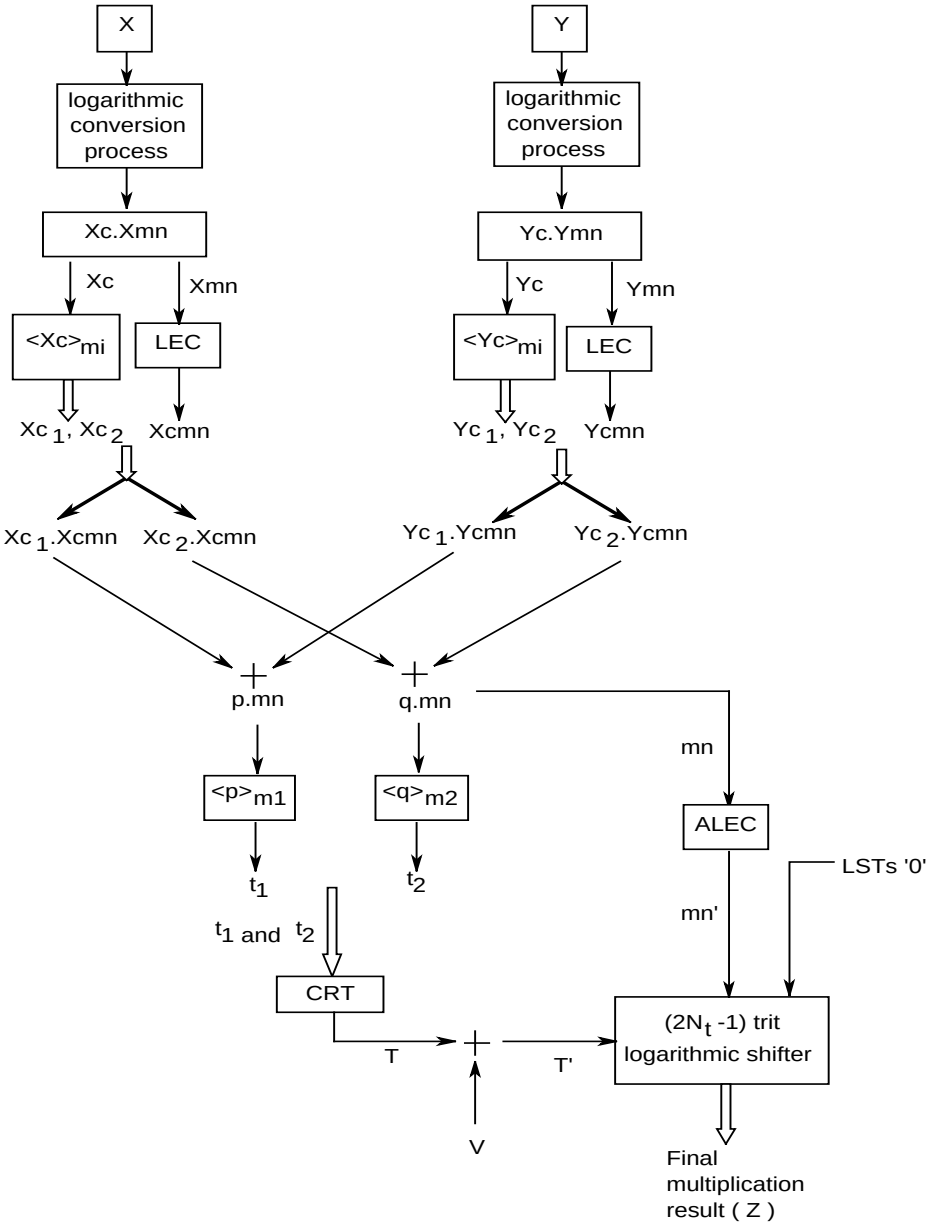


Figure 1. TVL based multiplication process for RLNS based system.

$$Xc_i = Xc \bmod m_i \tag{6}$$

$$Yc_i = Yc \bmod m_i \tag{7}$$

where i takes the value of 1 and 2 that represents the two moduli values chosen. The multiplication operations on the residues are done based on the logarithmic rule given in Equation (4) that form the second step of RNS data processing. The corresponding added operands are denoted as p.mn and q.mn in **Figure 1**, where p, q denotes the characteristic values and mn, the final added mantissa value. The value of t_i utilized for next step of RNS is given by the following equations,

$$t_1 = \langle p \rangle_{m_1} \tag{8}$$

$$t_2 = \langle q \rangle_{m_2} \tag{9}$$

Then the reverse conversion process is followed to convert the residues to the corresponding ternary weighted value explained in equations given below. The reverse conversion method using Chinese Remainder Theorem (CRT) can be explained as,

$$T = \left\langle \sum_{i=1}^{Q} \langle t_i \times N_i \rangle_{m_i} \times M_i \right\rangle_M \tag{10}$$

$$M = \prod_{i=1}^{Q} m_i \tag{11}$$

$$M_i = \frac{M}{m_i} \tag{12}$$

$$N_i = \langle M_i^{-1} \rangle_{m_i} \tag{13}$$

where $i = 1, 2, \cdots, Q$. As the major manipulation processes of RLNS is carried out using smaller range of logarithmic values (characteristic values), CRT method is more suitable for performing reverse conversion operation compared to other techniques discussed before. The value of the added mantissa part (mn) is used in the antilogarithmic conversion process undergoing error correction process (discussed in later sections) to get the final multiplication result. The corrected value is represented as mn' in **Figure 1** that is given as input to the $(2N_t - 1)$ logarithmic shifter, along with the value of T' (explained in section 3) forming the control input to the shifter used. The final multiplication result, "Z" is thus obtained from the antilogarithmic conversion process.

3. Multiplication Process for RLNS Based Applications Utilizing TVL Concept

This paper proposes the initial idea in designing the complete RLNS based multiplication structure for ternary input values. It includes the schematic for the logarithmic conversion of the input ternary numbers, its manipulation process based on RLNS technique and its antilogarithmic conversion process to get the final multiplication result. The logarithmic conversion of the ternary values are designed by gaining the basic idea of the same for binary values [41] [49], with the whole part redesigned for ternary inputs. The rest of this work includes the design of multiplication process based on RLNS concept utilizing both TVL and binary logic, hence the number of trits is denoted as N_t and number of bits as N_b in the following sections.

The moduli set chosen for the design of RLNS is $\{3^{N_t} - 1, 3^{N_t}\}$, given the value of N_t = 2 forming {8, 9} which satisfies the condition of the values to be relatively prime. Thus the dynamic range provided by RLNS based multiplication scheme proposed is $[0, 3^{72})$, once the antilogarithmic conversion process is completed. This is the major advantage of using LNS, as the range of the moduli set though small, covers an abundant range of possible output values. The limit of the output value to be obtained can be set by designing the specific antilogarithmic converter with desired number of output trits. As $\log_3 255 = 5.0438$ and its ternary representation is 100110, in order to compare the proposed TVL based circuit with 8 bit binary logic circuits, the ternary logarithmic conversion circuit for 6 trit input is designed. Similar manipulation is followed for comparing 16 and 32 bit binary logic circuits with 11 and 21 trit length ternary logic circuits.

3.1. Ternary Logarithmic Conversion Process

The logarithmic conversion of the ternary input operand is shown in **Figure 2**. The initial operation involves in the identification of the position of the leading trit value of either "1" or "2", to determine the characteristic part of the input operand. This process is accomplished by the Leading Trit Detecting (LTD) circuit given in **Figure 3(a)**, whose operation is similar to that of Leading One Detector (LOD) circuit used for logarithmic conversion of binary input operand [50] [51]. Only the leading trit of logic value either "1" or "2" is passed as the output value with other trit values maintained as "0". The Ternary multiplexer (T-MUX) gives the output as "2" only when the Most Leading Trit (MST) is "0" else provides the result as "0", this output of T-MUX is then passed to consecutive multiplexers to identify the position of the leading trit of "1"or "2" in the input operand. The Standard Ternary AND gate (ST-AND) gate is used to give the leading trit value as output. For N_t greater than 6, *i.e.*, for 11 and 21 trit LTD circuit, the 5 and 6 trit LTD circuit forms the basic block, as shown in **Figure 3(b)** and the operation of block M is given in **Figure 3(c)**. The input variable d_i represents the output from the N_t trit LID circuit and the value a_i is the control value to the T-multiplexers (T-MUXs) used in the block M, which decides the flow of leading trit value to the output. This a_i in each block keeps the rest of the output values to be "0". Similarly for N_t = 11, one 6-trit and 5-trit LTD circuits are used with 2-trit LID to produce the control variable m_i.

The outputs from Leading Trit Detecting (LTD) circuit form input to the Characteristic Value Identifying (CVI) circuit similar to N_b-bit × $\log_2 N_b$ MOS ROM structure [52]. This circuit provides the characteristic value of the input operand with trit length of $\log_3 N_t + 1$, when the corresponding input line is evoked. The design of CVI circuit is shown in **Figure 4**. As the input operand taken is always > 1, value of the Least Significant Trit (LST) output from the LTD circuit is "0", keeping the initial PMOS of CVI in ON condition. The controlling gate at the input node (ST-AND and Negative Ternary-OR (NT-OR)) is required to maintain the correct characteristic value corresponding to the input line that drives the leading trit value.

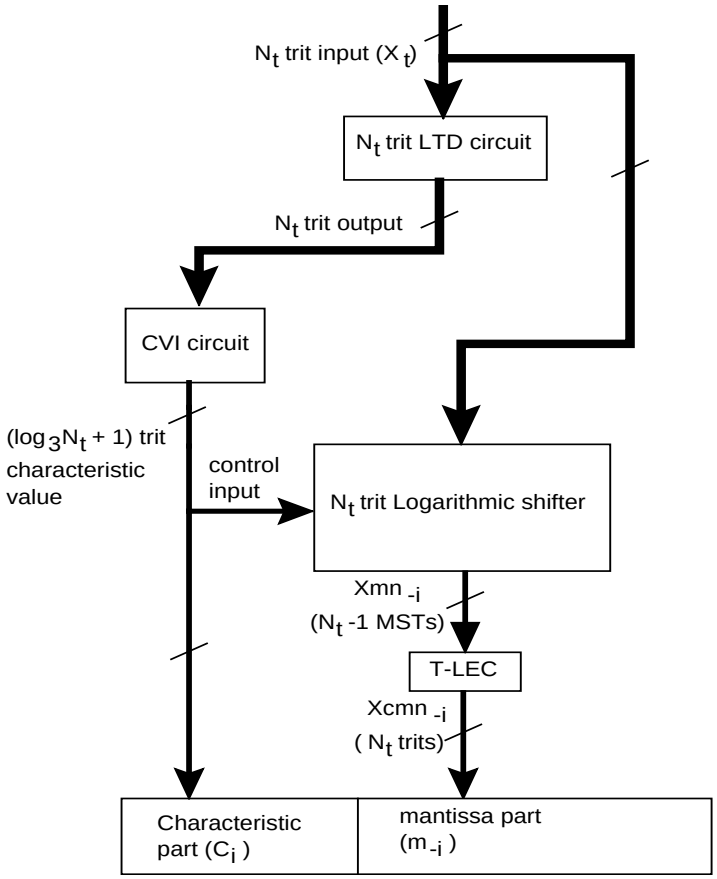
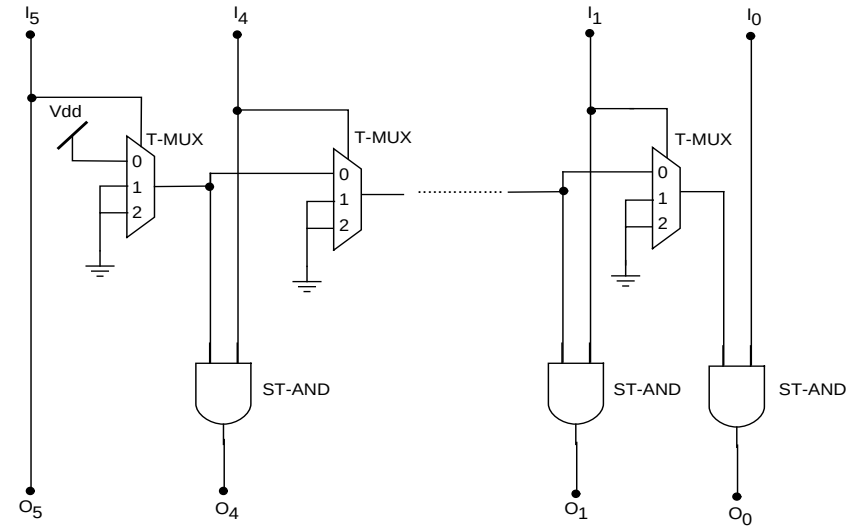
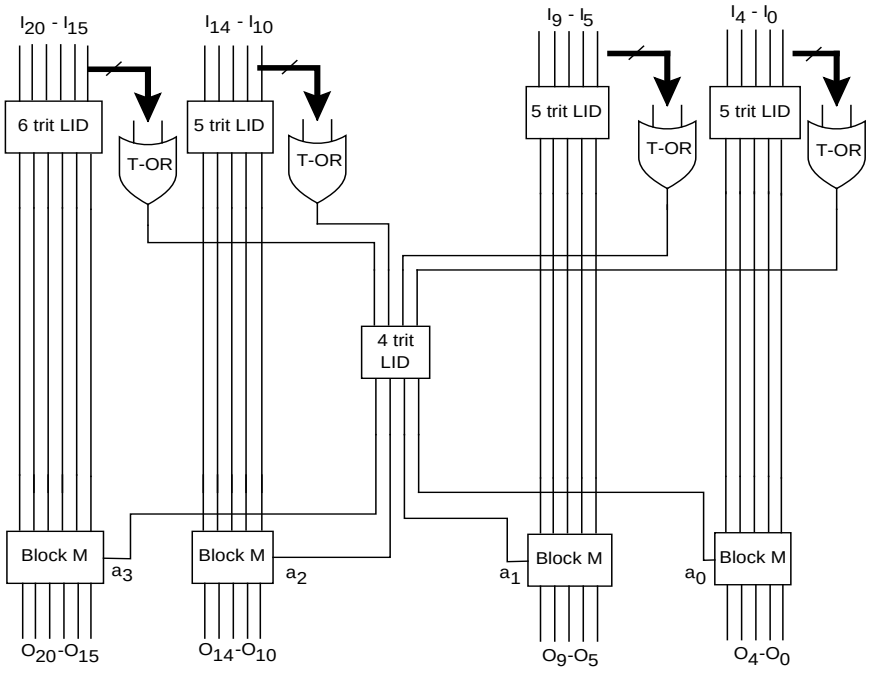


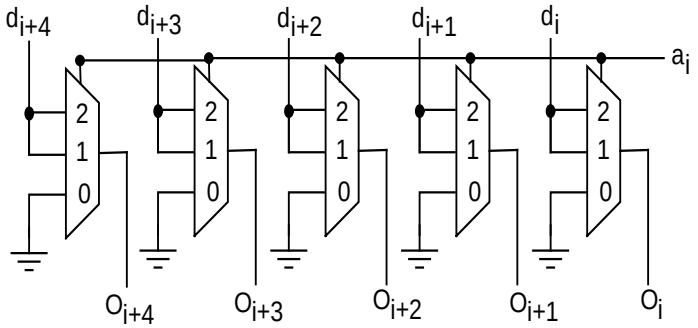
Figure 2. Logarithmic conversion process for N_t trit input operand.



(a)



(b)



(c)

Figure 3. (a) Leading trit detecting (LTD) circuit; (b) 21 trit LID circuit; (c) Block M.

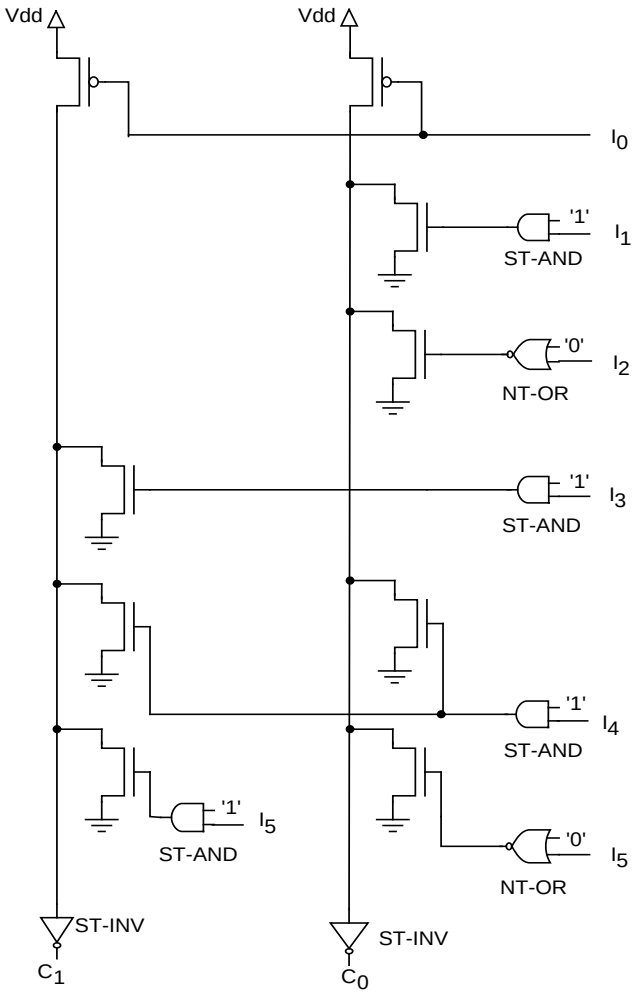


Figure 4. Characteristic value Identifying (CVI) circuit.

For instance if the input I_2 to CVI circuit is the leading trit, then its original value can be either "1" or "2", but the corresponding output lines should give the result as "02"(C_1C_0), which is the correct characteristic value. Thus including NT-OR gate in the input node I_2, fixes its output to be either "2" or "0" for the possible inputs of "1" and "2" or "0" respectively. Similar features are taken for consideration in placing the suitable control gate at the input nodes. **Figure 4** shows the circuit diagram for number of trits, $N_t = 6$, which can be expanded for $N_t > 6$, with the corresponding control gates at the input.

The value of mantissa part is obtained from N_t trit logarithmic shifter and is explained in **Figure 5**. The characteristic value forms the controlling input to the logarithmic shifter, which decides the shifting operation for a particular trit position is required or not. The number of shifts done in each stage i is either 0, 3^i or 2×3^i trit positions where the value of i starts from 0, 1, 2, ⋯ and so on, based on the value of control trit (characteristic value). The number of trits from the least significant positions is shifted for 0, 3^i or 2×3^i positions if the value of control trit, C_i is "0", "1" or "2" respectively as shown in **Figure 5**. So a ternary multiplexer is used before selecting a particular input bit for doing the corresponding shifting operation. The characteristic input is denoted as C_i, which controls the shifting operation of the circuit. The circuit shown in **Figure 5** is used to get the mantissa value for 6 trit input operand, where the LST is eliminated forming 5 trit result.

The value A_i shown in **Figure 5**, is the modified value of the corresponding characteristic trit C_i, which is "2" for C_i = "1" or "2" and "0" for C_i = "0". From the N_t trit logarithmic shifter, ($N_t - 1$) trit mantissa value is taken, denoted as Xmn and Ymn in **Figure 1**. The Least Significant Trit (LST) value of the mantissa portion is omitted to get accurate logarithmic values for input operands of format 3^n where n = 1, 2, ⋯ and so on. Once the logarithmic value of the input ternary operand is obtained, the mantissa value is corrected to get the near accurate

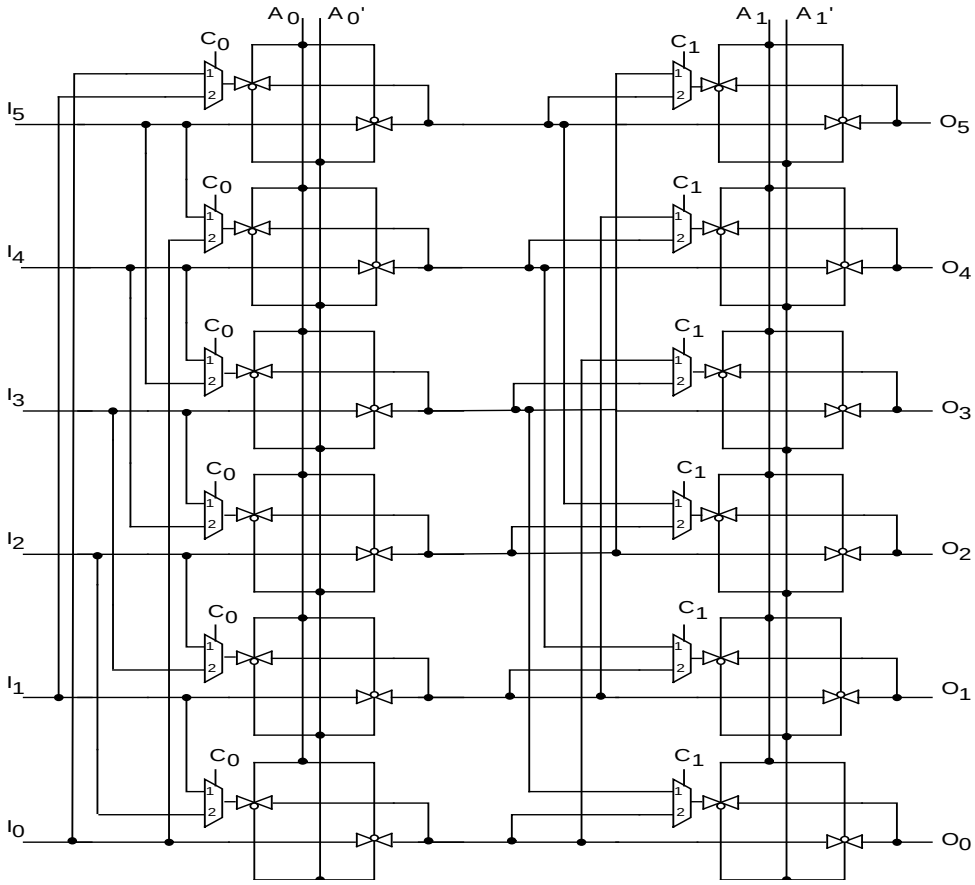


Figure 5. 6-trit logarithmic shifter.

result. The correction procedure followed is explained in later section (Section 4). Thus from the above mentioned procedures, the logarithmic value of the input ternary operand is obtained.

3.2. RLNS Processing of the Input Operands

As per the steps followed for RLNS technique proposed explained in Section 2, the residue values of the characteristic part (Xc_1, Xc_2 and Yc_1, Yc_2) based on the modulo values m_1 and m_2 are manipulated. The mantissa values obtained from the N_t trit logarithmic shifter discussed before (Xmn and Ymn) are corrected by the LEC process explained in Section 4, forming Xcmn and Ycmn. Now the Logarithmic addition on the residue values is done, which is the second step of RNS computing (arithmetic operation process).

The process can be explained from the equation given below,

$$p.mn = Xc_1.Xcmn + Yc_1.Ycmn \tag{14}$$

$$q.mn = Xc_2.Xcmn + Yc_2.Ycmn \tag{15}$$

Further modulo operation on the values p, q has to be done to get the final residue values t_1 and t_2 explained in Equations (8) (9) that are given as final residue inputs for the reverse conversion process based on the moduli set values $\{m_1, m_2\}$. The reverse conversion process is represented as CRT in **Figure 1**, whose operation is explained in Equations (10)-(13) are carried out to get the result T. Here the value of M = 72, as the moduli values m_1 and m_2 are taken as 8 and 9 respectively. Antilogarithmic conversion process is to be carried out on the final added mantissa value (mn) to obtain the final multiplication result. The mantissa part evaluated before, mn is corrected after undergoing ALEC process forming mn' (explained in Section 4) as shown in **Figure 1**, and is given as input to the $(2N_t - 1)$ trit logarithmic shifter. For binary logic based logarithmic conversion circuits with N_b bit input, the corresponding $2N_b$ bit antilogarithmic converter should be used [52], but in the case of TVL based design proposed $(2N_t - 1)$ trit antilogarithmic converter is appropriate. In order to compare 8 bit

RLNS based multiplication design with TVL, N_t has to be 6, as the value of $(255)_{10}$ is represented in ternary logic as $(100110)_3$ and the value of $\log_3 255 = 5.0438$. Similarly in order to easily compare the simulation results obtained, 16 and 32 bit RLNS based multiplication scheme is compared with 11 and 21 trit designs. The corrected mantissa value is of N_t trit length, hence the remaining LST values to $(2N_t - 1)$ trit logarithmic shifter are given as "0" mentioned in **Figure 1**.

The output obtained from the reverse conversion process, T along with some modification (T') is given as the control input to this $(2N_t - 1)$ trit logarithmic shifter. The value T is altered by adding an adjusting variable (V), this value is required to reduce the unwanted circuit portion of the shifter. As we compare 8 bit multiplication RLNS design with 6 trit TVL based scheme, the corresponding 11 trit logarithmic shifter is used for the antilogarithmic conversion process. But the multiplication of two 6 trit input operand can produce the maximum output value of $(728)_{10} \times (728)_{10} = (529984)_{10}$ *i.e.*, multiplication of $(222222)_3 \times (222222)_3$. Hence the circuit operation can be optimized to handle the maximum output value of a equivalent 16 bit value *i.e.*, $(255)_{10} \times (255)_{10} = (65025)_{10}$. Then the ternary logarithmic value of the maximum possible product value can be given as $\log_3 65025 = 10.0877$. In order to prevent the value of the control input T' to not exceed $(101)_3$, which is the required maximum possible characteristic value, the adjusting variable (V) of $(121)_3$ is added with T and inverted to avoid unwanted shifting of the logarithmic shifter, *i.e.*, $T' = (T + V)'$. Similarly for 11 and 21 trit multiplication designs, the adjusting variables of $(020)_3$ and $(1111)_3$ are added with the respective T values to get the corresponding values of T'. The final multiplication value "Z", as shown in **Figure 1** is obtained from the $(2N_t - 1)$ trit logarithmic shifter. The circuit operation of $(2N_t - 1)$ trit logarithmic shifter is same as that of the logarithmic shifter shown in **Figure 5**.

4. Logarithmic and Antilogarithmic Error Correction Process (LEC and ALEC)

In order to approximate the mantissa values obtained during the logarithmic conversion process, some additional parameters are required. The three MSTs of the input operands are given as the additional inputs in the logarithmic error correction process, orderly denoted as M_3, M_2 and M_1. The error correction process is not done based on the different error correction procedures that are followed during the binary logarithmic conversion process [41] [49]. This is due to the fact that, following Mitchell's approximation in ternary values is not suitable as it produces more inaccurate results. Therefore the methods that are proposed earlier for the improvisation of the Mitchell's approximation also become not suitable for designing ternary logarithmic circuits.

4.1. Logarithmic Error Correction (LEC) Process

The correction process proposed for ternary logarithmic values is shown in **Figures 6(a)-(b)**. This ternary logarithmic error correction circuit is common for 6, 11 and 21 trit multiplication circuit. The correction procedure that is carried out can be explained as follows, the variable m_{-i} in **Figure 6(a)** represents the mantissa input to the correction circuit. For instance, the ternary logarithmic value for a 6 trit input operand can be represented as $C_1C_0 \cdot m_{-1}m_{-2}m_{-3}m_{-4}m_{-5}$, where C_i and m_{-i} denotes the characteristic part and mantissa part respectively. The corresponding corrected mantissa part is denoted as Om_{-i}. The variables m_{-i} and Om_{-i} indirectly represents the logarithmic mantissa values of the input operands denoted as Xmn, Ymn and Xcmn, Ycmn respectively. Initially the three MSTs of the input operand M_3, M_2 and M_1 are checked for the different conditions that are dealt below.

Condition 1: If logic "2" value is present in any one positions of the two MSTs, M_3 or M_2, the MST (M_3) is taken out as the initial mantissa bit Om_{-1}, with the actual input mantissa values following Om_{-1} denoted as Om_{-2}-Om_{-6}, totally forming 6 trit output.

Condition 2: If both M_3 and M_2 is "1" with the value of M_1 "1" or "2", *i.e.*, 112 or 111, then the MST (M_3) is taken out as Om_{-1}, with the input mantissa values (m_{-1}-m_{-5}) inverted if and only if the logic state "2" is present or else passed as output values without any change. This selected inversion operation is denoted as m'_{-i} and this process is shown in **Figure 6(b)**. One exception is followed for this condition when Ms_1 value of the input operand is "1", in this case the 2nd corrected mantissa trit value Om_{-2} is taken out as "0" to get more accurate logarithmic value.

Condition 3: If M_3 and M_2 values as "1" and M_1 "0", the selected inversion operation alone is done on the input mantissa values, forming m'_{-i}, which is the required mantissa output, Om_{-i}.

Condition 4: If M_3 and M_2 are "1" and "0" respectively, then the input mantissa values are passed without

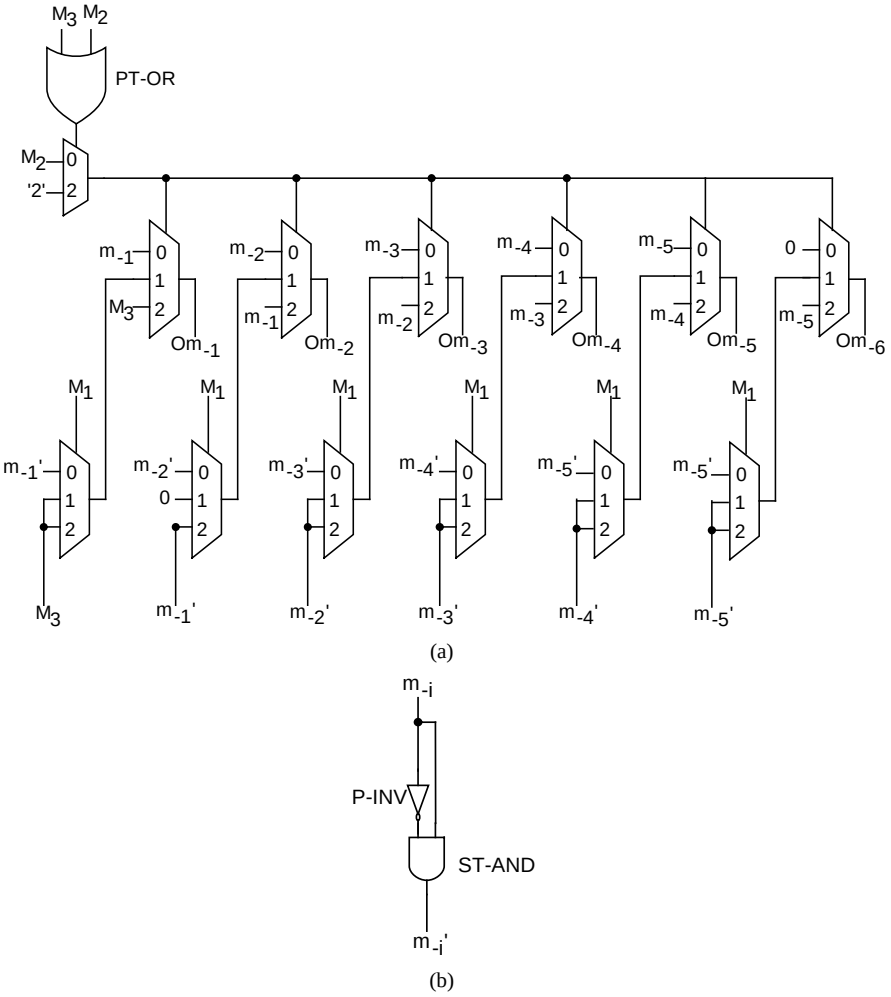


Figure 6. (a) Logarithmic error correction circuit for $N_t = 6$. (b) m_{-i}' value.

any corrections made.

The correction circuit performs the above mentioned correction procedures to get near accurate ternary logarithmic values (mantissa part). The working of the proposed correction circuit can be explained from the example given as follows, $(192)_{10} = (021010)_3$, the actual ternary logarithmic value of $\log_3(192) = 4.7855$. By the proposed technique the value of the three MSTs are $M_3 = 2$, $M_2 = 1$ and $M_1 = 0$, this falls under the condition 1, which states that the mantissa value can be passed without any corrections as output from Om_{-2} - Om_{-6}, following the MST (M_3) as Om_{-1} *i.e.*, (0.21010), thus the logarithmic value after identifying the characteristic part becomes, 4.7901. This correction procedure for 5 MSTs of the mantissa part remains same for the multiplication structure with $N_t > 6$. But the circuit gets expanded for proper selection of trit values to make the correction procedure suitable for the conditions 1, 2 and 3 explained above.

4.2. Antilogarithmic Error Correction (ALEC) Process

The correction procedures proposed for antilogarithmic conversion process is shown in **Figure 7**. This error correction circuit shown suits for 6 trit RLNS based multiplication design. As per the entire multiplication process explained in **Figure 1**, the added mantissa value (mn) is given as input to the antilogarithmic correction process. For 6 trit length multiplication process, the length of the added mantissa part is 5. But the antilogarithmic error correction circuit for $N_t = 6$, delivers 6 MST values, with the rest of the LST (Least Significant Trit) values as "0", which form the input to the 11 trit logarithmic shifter. Similarly for the 11 and 21 trit multiplication process, the corresponding antilogarithmic error correction circuit produces 11 and 21 trit output that form

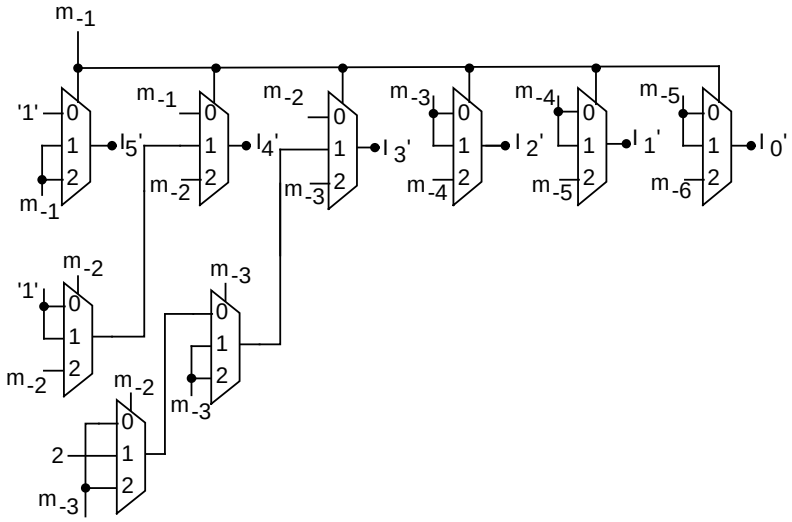


Figure 7. Antilogarithmic error correction circuit (ALEC) for $N_t = 6$.

the MST inputs for the logarithmic shifter.

The output from the correction circuit is denoted as I'_i in **Figure 7**, this value gets shifted in the logarithmic shifter based on the control input obtained from the reverse conversion process, T'. The values m_{-i} and I'_i indirectly represents the variables mn and mn', shown in **Figure 1** in its generalized form. The working principle of the ALEC process proposed can be stated as follows. The major correction operation is preformed on the 3 most significant mantissa value, is explained as follows.

Condition 1: If the value of m_{-1} is "2", then the input value from m_{-1} to m_{-6} is passed as outputs without any changes denoted as I'_5 to I'_0 in **Figure 7**.

Condition 2: If the value of m_{-1} is "1", the value of m_{-2} and m_{-3} are checked for providing appropriate corrected results. For this condition correction is made by reversing the operation followed for the 3 MSTs of the input operand, "112", "111" and "110" in LEC process. The effective process of counteracting the selected inversion operation is done on first 3 MSTs of the output, with the rest of the input mantissa trits cannot be assumed producing minimum error value.

Condition 3: If the value of m_{-1} is "0", then value of "1" is concatenated as the MST output I'_5, with the rest of the mantissa values m_{-i} shifted one step right forming output with 6 trit length.

This error correction circuit gets expanded for 11 and 21 trit input designs based on the condition 1 and 3, and finally the corrected output is given as input to the corresponding $(2N_t - 1)$ trit logarithmic shifter.

5. Analysis and Comparison of the Simulation Results Obtained

Simulation of the circuits designed are done using Cadence tool, Virtuoso with 45 nm CMOS technology, with the supply voltage of 0 V and 0.5 V for logic state "0" and "1" respectively in binary logic based design. Similarly for ternary logic circuits 1 V, 0.5 V and 0 V power supplies are used for logic states "2", "1" and "0" respectively. As RLNS based multiplication process is designed for TVL, the study on area, delay and total power dissipation values obtained are done by comparing the same with the design that uses binary logic. In binary logic based multiplication process for RLNS based system, the moduli set chosen is {8, 9}, with the value of N_b taken 3 in the moduli set $\left\{2^{N_b}, 2^{N_b}+1\right\}$. Thus the value of M becomes 72 as per the Equation (11), thus the dynamic range provided by the binary logic based multiplication scheme will be $[0, 2^{72})$. It should be noted that the actual dynamic range provided by the binary logic based design is very much lesser than the value provided by ternary logic based design. The working procedure of the multiplication process does not change irrespective of different logics followed, as the logarithmic concepts remain same. The modifications in the circuit are made in type of circuits based on logic chosen, the corresponding modulo conversion process or forward conversion process and in the number of bits or trits taken for the logarithmic and antilogarithmic conversion process.

The forward conversion process which includes the process of converting the characteristic part of the logarithmic operands to its corresponding residues, with respect to the moduli set {8, 9} in binary logic is done by

direct conversion method [46] [53]. The total number of input bits manipulated by the binary logic circuits (N_b) varies from that of the required trits (N_t) used in TVL based design. The binary logarithmic and antilogarithmic conversion process is done based on the circuits proposed by [41] [49]. The error correction process followed is based on divided approximation technique. The binary logic based multiplication scheme proposed is then compared for its area utilized, total power dissipation and delay values with the proposed TVL based design.

In addition to the comparison of the proposed concept with binary logic based structure, analysis is also made by comparing it with the design of modulo multipliers for $\left\{2^{N_b}-1, 2^{N_b}, 2^{N_b}+1\right\}$ based RNS system using booth encoding technique [54]. Here the multiplication process is done by Radix-8 booth encoding technique. It should be noted that this particular paper involves only in the design of modulo multipliers for the moduli set chosen. To compare the performance parameters of the technique [54] with the proposed RLNS based multiplication designs (both binary and ternary logic) the entire process along with reverse conversion process is designed using Cadence tool and the simulation results are taken. This existing technique is taken for analysis purpose to get the detailed idea of how RLNS based processing units show better area, power and delay parameters than the typical procedures of multiplication process followed.

It should be noted that the area utilization (μm^2) and delay values (ns) of the TVL based multiplication structure for RLNS based system prove to be more efficient compared to binary logic based RLNS design and the existing technique using Radix-8 booth encoding technique. Similarly the binary logic based RLNS design show low total power dissipation values than the other two techniques. The simulation results are tabulated in **Table 1** and the percentage of the parameter values saved by the proposed techniques are shown in **Table 2**. The total power dissipation values of the multiplication process using TVL is greater than the other two designs compared. This is due to the utilization of multiple power supplies to represent different states "2", "1" and "0" respectively.

Table 1. Comparison of Area, Total Power Dissipation, Delay and Power Delay Product (PDP) values obtained.

Multiplication structure		Area (μm^2)	Total Power Dissipation (μW/mW)	Delay (ns)	PDP ($\times 10^{-15}$ Joules)
Technique used	Number of trits/bits (N_t/N_b)				
TVL based RLNS scheme	6	12,536	3.96*	2.5	9.9
	11	19,910	15.7#	9.2	144,440
	21	31,734	36.5#	15.5	492,750
Binary logic based RLNS scheme	8	27,208	1.86*	123	228.78
	16	31,761	3.29*	188	618.52
	32	45,184	4.53*	232	1050.96
Based on Radix-8 booth encoding technique [54]	8	117,676	15.97*	443	7074.71
	16	164,997	32.17*	697	22,422.4
	32	215,781	53.25*	885	47,126.2

Note: * and # denotes total power dissipation values in μW and in mW units respectively.

Table 2. Percentage of delay and area values saved by the proposed technique over other two techniques.

Proposed Multiplication structure	Designs taken for comparison and analysis purpose	Number of bits (N_b)	Percentage of parameter values saved (%)	
			Delay	Area
TVL based RLNS scheme	Existing technique [54]	8	99.4	89
		16	98.6	80
		32	98.2	85
	Binary logic based RLNS design	8	97.9	54
		16	95.1	37
		32	93.3	30

The disadvantage of the proposed idea is encountered in terms of accuracy of the final result due to the approximation procedures followed for logarithmic and antilogarithmic conversion process. For a random selection of 500 set of input operands the accuracy of the final result obtained are analyzed for each N_t category and studied. For the proposed RLNS based multiplication design using TVL with the corresponding error correction method, final value is obtained with the average error percent of about ≤7. Also for the same scheme using binary logic and with the existing error correction procedures [41] [49], error value of ≤2% is obtained. And it should be noted that the RNS based multiplication process using Radix-8 booth encoding techniques provide accurate results as no approximation procedure is required in this process.

But the power dissipation and area utilization values saved by the proposed techniques should be taken seriously, as both the parameters play an important role in every DSP applications. Also the computational complexity of the multiplication process can be reduced to a maximum level when logarithmic property is involved. The Radix-N booth encoding technique incorporates more complex features into multiplication process when the value of N is increased further, though the partial product count is reduced considerably. Hence all the above mentioned criteria is considered for the design of effective RNS based processors using logarithmic concept. As multiplication operations are performed repetitively in almost all DSP based applications, the proposed RLNS based multiplication process for those that can tolerate the minimum error value produced can be considered to be more appropriate. The circuits where area and delay parameters play an important role, TVL based multiplication scheme can be used. For overall reduction of area, power and delay parameters that deserve near accurate results, RLNS based multiplier design using binary logic can be chosen.

6. Conclusion

Among the simplified and effective procedures followed for performing multiplication operation, utilizing logarithmic number system produce significant results in terms of area, power and delay values. This number system reduces the multiplication operation to mere addition process, thus removing the complexity of undergoing typical procedures like partial product generation and its accumulation. In this paper, the initial process of utilizing ternary values in its logarithmic format is undertaken and the circuit design is made accordingly. The RLNS based multiplication procedure is also designed for the binary logic based input values to compare its area, power and delay parameters with the proposed design. Additionally the existing research work on modulo multiplier design for RNS based system using Radix-8 booth encoding technique is taken for analysis purpose to compare this procedure with the proposed techniques. Though different percentage of error values are produced for designs using different logics (binary and ternary), the optimum design suitable for a specific application can be chosen. From the simulation results obtained, the following conclusions can be made, the highly efficient design in terms of area and delay can be obtained by employing TVL. The overall resource efficient multiplier design delivering near accurate results using the proposed scheme is possible by utilizing binary logic. Thus the ternary and binary logic based multiplication process for RLNS based system can be used in DSP applications where area and power efficient designs become mandatory.

References

[1] Wei, W., Swamy, M.N.S. and Ahmad, M.O. (2004) RNS Application for Digital Image Processing. *Proceedings. 4th IEEE International Workshop on System-on-Chip for Real-Time Applications*, 19-21 July 2004, 77-80.

[2] Cardarilli, G.C., Nannarelli, A. and Re, M. (2007) Residue Number System for Low-Power DSP Applications. *Conference Record of the Forty-First Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, 4-7 November 2007, 1412-1416. http://dx.doi.org/10.1109/acssc.2007.4487461

[3] Jenkins, W. K. and Leon, B. J. (1977) The Use of Residue Number Systems in the Design of Finite Impulse Response Digital Filters. *IEEE Transactions on Circuits and Systems*, **24**, 191-201. http://dx.doi.org/10.1109/TCS.1977.1084321

[4] Conway, R. and Nelson, J. (2004) Improved RNS FIR Filter Architectures. *IEEE Transactions on Circuits and Systems II*: *Express Briefs*, **51**, 26-28.

[5] Ramirez, J., Meyer-Base, U. and Garcia, A. (2005) Efficient RNS-Based Design of Programmable FIR Filters Targeting FPL Technology. *Journal of Circuits, Systems and Computers*, **14**, 165. http://dx.doi.org/10.1142/S0218126605002131

[6] Ammar, A., Al Kabbany, A., Youssef, M. and Amam, A. (2001) A Secure Image Coding Using Residue Number System. *Proceedings of the* 18*th National Radio Science Conference*, **2**, 399-405.

http://dx.doi.org/10.1109/nrsc.2001.929397

[7] Younes, D. and Steffan, P. (2013) Efficient Image Processing Application Using Residue Number System. *Proceedings of the* 20*th International Conference on Mixed Design of Integrated Circuits and Systems* (*MIXDES*), Gdynia, 20-22 June 2013, 468-472.

[8] Navin, A.H., *et al.* (2011) A Novel Approach Cryptography by Using Residue Number System. *Proceedings of* 6*th International Conference on Computer Sciences and Convergence Information Technology* (*ICCIT*), Seogwipo, 29 November-1 December 2011, 636-639.

[9] Posch, K.C. and Posch, R. (1992) Residue Number System: A Key to Parallelism in Public Key Cryptography. *Proceedings of the Fourth IEEE Symposium on Parallel and Distributed Processing*, Arlington, TX, 1-4 December 1992, 432-435.

[10] Schinianakis, D.M., Kakarountas, A.P. and Stouraitis, T. (2006) A Novel Approach to Elliptic Curve Cryptography: An RNS Architecture. *Proceedings of IEEE Mediterranean Electrotechnical Conference* (*MELECON*), Malaga, 16-19 May 2006, 1241-1245.

[11] Kong, Y. and Lai, Y.F. (2013) Low Latency Modular Multiplication for Public-Key Cryptosystems Using a Scalable Array of Parallel Processing Elements. *Proceedings of IEEE* 56*th International Midwest Symposium on Circuits and Systems* (*MWSCAS*), Columbus, 4-7 August 2013, 1039-1042. http://dx.doi.org/10.1109/mwscas.2013.6674830

[12] Abdallah, M. and Skavantzos, A. (1995) A Systematic Approach for Selecting Practical Moduli Sets for Residue Number Systems. *Southeastern Symposium on System Theory*, Starkville, 12-14 March 1995, 445-449. http://dx.doi.org/10.1109/ssst.1995.390542

[13] Wang, W., Swamy, M.N.S. and Ahmad, M.O. (2003) Moduli Selection in RNS for Efficient VLSI Implementation. *Proceedings of the* 2003 *International Symposium on Circuits and Systems*, **4**, 512-515. http://dx.doi.org/10.1109/iscas.2003.1205945

[14] Paliouras, V. and Stouraitis, T. (2001) Low-Power Properties of the Logarithmic Number System. *Proceedings of the* 15*th IEEE Symposium on Computer Arithmetic*, Vail, 11-13 June 2001, 229-236. http://dx.doi.org/10.1109/arith.2001.930124

[15] Agrawal, R.K. and Kittur, H.M. (2013) ASIC Based Logarithmic Multiplier Using Iterative Pipelined Architecture. *Proceedings of the IEEE Conference on Information and Communication Technologies* (*ICT*), JeJu Island, 11-12 April 2013, 362-366. http://dx.doi.org/10.1109/cict.2013.6558121

[16] Lewis, D.M. (1995) 114 MFLOPS Logarithmic Number System Arithmetic Unit for DSP Applications. *IEEE Journal of Solid-State Circuits*, **30**, 1547-1553. http://dx.doi.org/10.1109/4.482205

[17] Arnold, M.G. (2005) The Residue Logarithmic Number System: Theory and Implementation. *Proceedings of the* 17*th IEEE Symposium on Computer Arithmetic*, 27-29 June 2005, 196-205. http://dx.doi.org/10.1109/ARITH.2005.44

[18] Lee, B. and Burgess, N. (2003) A Dual-Path Logarithmic Number System Addition/Subtraction Scheme for FPGA. In: Cheung, P.Y.K. and Constantinides, G.A., Eds., *Field Programmable Logic and Application*, Springer-Verlag Berlin Heidelberg, 808-817. http://dx.doi.org/10.1007/978-3-540-45234-8_78

[19] Mousavi, A. and Taleshmekaeil, D.K. (2010) Pipelined Residue Logarithmic Number System for General Modules Set $\{2^N - 1, 2^N, 2^N + 1\}$. *Proceedings of the IEEE* 5*th International Conference on Computer Sciences and Convergence Information Technology* (*ICCIT*), Seoul, 30 November-2 December 2010, 699-703. http://dx.doi.org/10.1109/ICCIT.2010.5711144

[20] Srivastava, A. and Venkatapathy, K. (1996) Design and Implementation of Low Power Ternary Full Adder. *VLSI Design*, **4**, 75-81. http://dx.doi.org/10.1155/1996/94696

[21] Butler, J.T. (1991) Multiple-Valued Logic in VLSI. IEEE Computer Society Press Technology Series, Los Alamitos.

[22] Kameyama, M. (1990) Toward the Age of Beyond-Binary Electronics and Systems. *Proceedings of the* 20*th International Symposium on Multiple-Valued Logic*, Charlotte, 23-25 May 1990, 162-166. http://dx.doi.org/10.1109/ISMVL.1990.122613

[23] Yoeli, M. and Rosenfeld, G. (1965) Logical Design of Ternary Switching Circuits. *IEEE Transactions on Electronic Computers*, **EC-14**, 19-29. http://dx.doi.org/10.1109/PGEC.1965.264050

[24] Balla, P.C. and Antoniou, A. (1984) Low Power Dissipation MOS Ternary Logic Family. *IEEE Journal of Solid-State Circuits*, **19**, 739-749. http://dx.doi.org/10.1109/JSSC.1984.1052216

[25] Das, S., Dasgupta, P. and Sensarma, S. (2012) Arithmetic Algorithms for Ternary Number System. In: Rahaman, H., Chattopadhyay, S. and Chattopadhyay, S., Eds., *Progress in VLSI Design and Test*, Springer, Berlin Heidelberg, 111-120. http://dx.doi.org/10.1007/978-3-642-31494-0_13

[26] Smith, K.C. (1981) The Prospects for Multi Valued Logic: A Technology and Application View. *IEEE Transactions on Computers*, **C-30**, 619-634. http://dx.doi.org/10.1109/TC.1981.1675860

[27] Hurst, S.L. (1988) Two Decades of Multiple Valued Logic—An Invited Tutorial. *Proceedings of the* 18*th International Symposium on Multiple-Valued Logic*, Palma de Mallorca, 24-26 May 1988, 164-175. http://dx.doi.org/10.1109/ismvl.1988.5170

[28] Vranesic, Z.G. and Hamacher, V.C. (1971) Ternary Logic in Parallel Multipliers. *The Computer Journal*, **15**, 254-258. http://dx.doi.org/10.1093/comjnl/15.3.254

[29] Mouftah, H.T. and Jordan, I.B. (1977) Design of Ternary COS/MOS Memory and Sequential Circuits. *IEEE Transactions on Computers*, **C-26**, 281-288. http://dx.doi.org/10.1109/TC.1977.1674821

[30] Mouftah, H.T. (1976) A Study on the Implementation of Three Valued Logic. *Proceedings of the* 6*th International Symposium on Multiple-Valued Logic*, 123-126.

[31] Mouftah, H.T. and Smith, K.C. (1980) Design and Implementation of Three-Valued Logic Systems with M.O.S. Integrated Circuits. *IEE Proceedings G-Electronic Circuits and Systems*, **127**, 165-168. http://dx.doi.org/10.1049/ip-g-1.1980.0028

[32] Gaikwad, V.T. and Deshmukh, P.R. (2015) Design of CMOS Ternary Logic Family based on Single Supply Voltage. *Proceedings of IEEE International Conference on Pervasive Computing* (*ICPC*), Pune, 8-10 January 2015, 8-10. http://dx.doi.org/10.1109/pervasive.2015.7087114

[33] Hosseinzadeh, M. and Navi, K. (2007) A New Moduli Set for Residue Number System in Ternary Valued Logic. *Journal of Applied Sciences*, **7**, 3729-3735. http://dx.doi.org/10.3923/jas.2007.3729.3735

[34] Hosseinzadeh, M., Jassbi, S.J. and Navi, K. (2008) A New Moduli Set $\{3^N - 1, 3^N + 1, 3^N + 2, 3^N - 2\}$ in Residue Number System. 10*th International Conference on Advanced Communication Technology*, Gangwon-Do, 17-20 February 2008, 1601-1603.

[35] Arnold, M., Bailey, T. and Cowles, J. (2003) Error Analysis of the Kmetz/Maenner Algorithm. *Journal of VLSI Signal Processing Systems for Signal*, *Image and Video Technology*, **33**, 37-53. http://dx.doi.org/10.1023/A:1021189701352

[36] Brubaker, T.A. and Becker, J.C. (1975) Multiplication Using Logarithms Implemented with Read-Only-Memory. *IEEE Transactions on Computers*, **24**, 761-766. http://dx.doi.org/10.1109/T-C.1975.224307

[37] Mitchell Jr., J.N. (1962) Computer Multiplication and Division Using Binary Logarithms. *IEEE Transactions on Electronic Computers*, **11**, 512-517. http://dx.doi.org/10.1109/TEC.1962.5219391

[38] Combet, M., Van Zonneveld, H. and Verbeek, L. (1965) Computation of the Base Two Logarithm of Binary Numbers. *IEEE Transactions on Electronic Computers*, **14**, 863-867. http://dx.doi.org/10.1109/PGEC.1965.264080

[39] Hall, E.L., Lynch, D.D. and Dwyer, S.J. (1970) Generation of Products and Quotients Using Approximate Binary Logarithms for Digital Filtering Applications. *IEEE Transactions on Electronic Computers*, **19**, 97-105. http://dx.doi.org/10.1109/T-C.1970.222874

[40] SanGregory, S.L., Brothers, C., Gallagher, D. and Siferd, R. (1999) A Fast, Low-Power Logarithm Approximation with CMOS VLSI Implementation. 42*nd Midwest Symposium on Circuits and Systems*, **1**, 388-391. http://dx.doi.org/10.1109/MWSCAS.1999.867287

[41] Abed, K.H. and Siferd, R.E. (2003) CMOS VLSI Implementation of a Low-Power Logarithmic Converter. *IEEE Transactions on Computers*, **52**, 1421-1433. http://dx.doi.org/10.1109/TC.2003.1244940

[42] Mahalingam, V. and Ranganathan, N. (2006) Improving Accuracy in Mitchell's Logarithmic Multiplication Using Operand Decomposition. *IEEE Transactions on Computers*, **55**, 1523-1535. http://dx.doi.org/10.1109/TC.2006.198

[43] Mahalingam, V. and Ranganathan, N. (2006) An Efficient and Accurate Logarithmic Multiplier Based on Operand Decomposition. 19*th International Conference on VLSI Design held jointly with* 5*th International Conference on Embedded Systems Design* (*VLSID*'06), 3-7 January 2006. http://dx.doi.org/10.1109/vlsid.2006.42

[44] Parhami, B. (2000) Computer Arithmetic: Algorithms and Hardware Design. Oxford University Press, Oxford.

[45] Mohan, P.V.A. (2007) RNS-to-Binary Converter for a New Three-Moduli Set $\{2^{n+1} - 1, 2^n, 2^n - 1\}$. *IEEE Transactions on Circuits and Systems II*: *Express Briefs*, **54**, 775-779. http://dx.doi.org/10.1109/TCSII.2007.900844

[46] Hariri, A., Navi, K. and Rastegar, R. (2008) A New High Dynamic Range Moduli Set with Efficient Reverse Converter. *Computers & Mathematics with Applications*, **55**, 660-668. http://dx.doi.org/10.1016/j.camwa.2007.04.028

[47] Cao, B., Chang, C.H. and Srikanthan, T. (2003) Adder Based Residue to Binary Converters for a New Balanced 4-Moduli Set. *Proceedings of the* 3*rd International Symposium on Image and Signal Processing and Analysis*, **2**, 820-825.

[48] Cao, B., Chang, C.H. and Srikanthan, T. (2003) 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 Transactions on Circuits and Systems I*: *Fundamental Theory and Applications*, **50**, 1296-1303. http://dx.doi.org/10.1109/TCSI.2003.817789

[49] Abed, K.H. and Siferd, R.E. (2003) VLSI Implementation of a Low-Power Antilogarithmic Converter. *IEEE Transactions on Computers*, **52**, 1221-1228. http://dx.doi.org/10.1109/TC.2003.1228517

[50] Oklobdzija, V. (1992) An Implementation Algorithm and Design of a Novel Leading Zero Detector Circuit. *Signals, Conference Record of the* 26*th Asilomar Conference on Systems and Computers*, Pacific Grove, 26-28 October 1992, 391-395. http://dx.doi.org/10.1109/acssc.1992.269243

[51] Oklobdzija, V. (1994) An Algorithmic and Novel Design of a Leading Zero Detector Circuit: Comparison with Logic Synthesis. *IEEE Transactions on Very Large Scale Integration* (*VLSI*) *Systems*, **2**, 124-128. http://dx.doi.org/10.1109/92.273153

[52] Abed, K.H. and Siferd, R.E. (2000) CMOS VLSI Implementation of 16-Bit Logarithm and Anti-Logarithm Converter. *Proceedings of the* 43*rd IEEE Midwest Symposium on Circuits and Systems*, **2**, 776-779. http://dx.doi.org/10.1109/MWSCAS.2000.952871

[53] Samhitha, N.R., Cherian, N.A., Jacob, P.M. and Jayakrishnan, P. (2013) Implementation of 16-bit floating point multiplier using Residue Number System. 2013 *International Conference on Green Computing, Communication and Conservation of Energy* (*ICGC*E), Chennai, 12-14 December 2013, 195-198. http://dx.doi.org/10.1109/icgce.2013.6823427

[54] Muralidharan, R. and Chang, C.-H. (2012) 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*, **59**, 2263-2273. http://dx.doi.org/10.1109/TCSI.2012.2185334