ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

# Radix-4 and Radix-8 32 Bit Booth Encoded Multi-Modulus Multipliers

**K.Sai Ram Charan**
**M.Tech Student,**
**Department of ECE,VNR,**
**Vignana Jyothi Institute of Engineering and Technology (Autonomous),**
**Hyderabad, India.**

**K.Kalyan Srinivas**
**Associate Professor,**
**Department of ECE,VNR,**
**Vignana Jyothi Institute of Engineering and Technology (Autonomous),**
**Hyderabad, India.**

## Abstract:

ANovel multi-modulus designs are capable of performing the desired modulo operation for more than one modulus in Residue Number System (RNS) and are explored in this paper to lower the hardware overhead of residue multiplication. Two multi-modulus multipliers that reuses the hardware resources among the modulo and modulo multipliers by virtue of their analogous number theoretic properties are proposed. In the proposed radix- Booth encoded multi-modulus multiplier, the modulo-reduced products for the moduli are computed successively. on the basis of the radixBooth encoded modulo and modulo multiplier architectures the new Booth encoded modulo multipliers are proposed to maximally share the hardware resources in the multi-modulus architectures. Our experimental results based on RNS multiplication has shown that the proposed radix and radix- Booth encoded multi-modulus multipliers has save nearly 60% of area over the corresponding single-modulus multipliers. The proposed radix Booth encoded multi-modulus multipliers increase the delay of the corresponding single-modulus multipliers by 18% and 13%, respectively in the worst case. Compared to the single-modulus multipliers, the proposed multi-modulus multipliers incur a minor power dissipation penalty of 5%.

## Index Terms:

Booth Algorithm, Multi-Modulus Architectures, Multiplier, Residue Number System(RNS).

## Introduction:

Residue number system (RNS) has been shown tobe an advantageous alternative to traditional Two's Complement System (TCS) for accelerating certain computations in high dynamic range (DR) applications such as signal processing, communication, and cryptographic algorithms [1]–[3]. RNS is defined by a set of N pair-wise co-prime moduli{m1,m2,…mN} called the base. For unambiguous representation, the DR of the RNS is given by the product of all moduli comprising the base, i.e., $\prod_{i=1}^{N} m_i$. An integer X within the DR is represented by a unique –tuple{x1,x2…xN},where is a residue defined by modulo $m_i$ or $|X|m_i$ . The operation is implemented as $(z_1, z_2, \cdots, z_N) = (|x_1 \circ y_1|_{m_1}, |x_2 \circ y_2|_{m_2}, \cdots, |x_N \circ y_N|_{m_N})$, where each residue of is independently computed only from the residues,$X_i$ and $Y_i$, of the operands, X and Y, respectively in a modulo channel corresponding to the modulus $m_i$ [4].

The absence of inter-channel carry propagation and the reduced length of intra-channel carry propagation chains lead to faster implementation in RNS when compared to its TCS counterpart. However, the advantages of RNS may not translate well into area and power constrained platforms due to the significant extra hardware required for the conversion between TCS and RNS as well as the concurrent computations in several modulo channels.A factor that directly influences the hardware complexity of RNS is the choice of the moduli that comprise the RNS base. Compared to the general moduli, special moduli of forms,$2^n$ +1,$2^n$,$2^n$-1that posses good number theoretic properties are preferred. By exploiting these properties, full-adder based binary-to-residue and residue-to-binary converters for special moduli sets have been developed [5]–[8]. Hardware efficient modulo2 $^n$-1 and modulo$2^n$ +1 arithmetic circuits such as adders and multipliers have also been designed with these properties as the basis [9]–[23].To lower the RNS area and power dissipation, system level design methodologies like multi-function and multi-modulus designs have also been suggested [24], [25]. Architectures that perform the same modulo operation

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

for more than one modulus are called multi-modulus architectures (MMAs) [24]. MMA is classified into two types: fixed multi-modulus architecture (FMA) and variable multi-modulus architecture (VMA). The modulo operations corresponding to the different moduli are performed concurrently in a FMA butserially in aVMA.In contrast to the single-modulus architecture (SMA), MMA exploits hardware reuse amongst the modulo channels of RNS. Hence, both FMA and VMA result in considerable savings in area as well as power dissipation. MMA is widely used in reconfigurable and fault-tolerant RNSs where the moduli of the base are selectively used. In reconfigurable RNS, the DR of RNS is varied by selecting a subset of moduli from the base to fulfill the DR requirement of the application. The channels corresponding to the unselected moduli are turned off there by reducing the system power dissipation [26], [27]. Infault-tolerant RNS, additional moduli are included in the base such that the DR of RNS far exceeds the DR requirement of the application.

These redundant moduli are used to detect and isolate faults in computations. Once the faulty residue digit is located, the corresponding modulus of the base is discarded[28], [29]. Another specific application of VMA is high DR RNSs based on imbalanced word-length moduli sets [5], [8]. In such RNSs, a VMA can be employed for the non-critical moduli to substantially lower the hardware requirement without compromising the system delay.Consequently, MMAs for the three special moduli,,$2^n+1$,$2^n$ and $2^n-1$have been explored in recent times. In [30], a variable multi-modulus two-operand parallel-prefix adder that computes the sum modulo $2^n-1$ ,modulo$2^n$ ,modulo $2^n+1$ in only[$\log_2 n$]prefix levels was proposed. By extending the number of operands, a variable multi-modulus multi-operand adder for moduli $2^n-1$ and moduli $2^n+1$ and based on regularly structured carry save adder (CSA) tree was described in [31].

By making use of the variable multi-modulus multi-operand adder, a variable multi-modulus multiplier for the three special moduli was suggested in [32]. Recently, a dual-modulus multiplier based onradix-4 Booth encoding for the moduli $2^n-1$ and$2^n+1$ was proposed [33]. For the moduli,$2^n+1$,$2^n$ and $2^n-1$ non-encoded coded fixed and variable multi-modulus squares were proposed in [34].

In [35], the VMA for squarer was further improved by employing the radix-4 Booth encoding algorithm. In this paper, Booth encoding technique is applied to multi modulus multiplication. Two new multi-modulus multipliers for the moduli,$2^n+1$,$2^n$ and $2^n-1$ are proposed. The former uses the radix-$2^2$ Booth encoding scheme while the latter uses the radix- Booth encoding scheme. In the radix-$2^3$ Booth encoded multi-modulus multiplier, the FMA approach is applicable only to the partial product generation stage and not applicable to the bias generation and partial product addition stages. While in the radix-$2^2$ Booth encoded multi-modulus multiplier, the FMA approach is relevant only to the encoding of the multiplier bits. Owing to the restricted applicability of the FMA in encoded multipliers, only VMA is considered in this paper.For the proposed radix-$2^2$ Booth encoded multi-modulus multiplier, the modulo multiplier of [13] is used as the baseline architecture. The preliminary modulo multiplier architecture we proposed in [20] is shown to be advantageous for sharing resources with [13].

Furthermore, a new radix-$2^2$Booth encoded modulo multiplier that maximally reuses the hard ware blocks of [13] and [20] is proposed. The equivalences in the radix-$2^3$ Booth encoded modulo$2^n+1$ and modulo$2^n-1$multiplier architectures initially presented in [23] areexploited to design the proposed radix-$2^3$ Booth encoded multi-modulus multiplier. The structure of the modulo hard multiple generator (HMG) is modified to employ dual prefix operators for handling the complemented and non-com-plemented modified-generate and modified-propagate signalpair. A new radix-$2^3$ Booth encoded modulo multiplier that is analogous to [23] is also introduced. In both the proposed Radix- $2^n$ and radix- $2^2$ Booth encoded multi-modulus multipliers, an efficient technique to include the bias associated with modulo $2^n$negation is also proposed.

The suggested modulo $2^n$ bias generation involves only hardwiring of the Booth encoder outputs and does not incur any additional hardware overhead.This paper is organized as follows. The properties of modulo$2^n-1$, modulo $2^n$ and modulo $2^n-1$arithmetic are discussedin Section II. In Sections III and IV, the proposed radix- and radix- Booth encoded multi-modulus multipliers aredescribed with emphasis on multi-modulus partial

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

product,hard multiple and bias generations as well as multi-moduluspartial product accumulation. In Section V, the performancesof the proposed multi-modulus multipliers are evaluated andcompared againstbased RNS multipliersthat employ the same encoding scheme. Conclusions are drawnin Section VI.

## II. Preliminaries

In this section, the fundamental properties of modul $2^n$ -1,modulo2 $^n$ and modulo$2^n$ +1 arithmetic that will be employed in the design of radix 2 $^2$ and radix-2 $^3$ Booth encoded multi modulus multipliers are reviewed [4], [11], [15], [18], [19], [23].All equations are assumed to be modulo-reduced by,m E{$2^n$ +1,$2^n$,2 $^n$-1}unless otherwise specified.

let $X = \sum_{i=0}^{n-1} 2^i x_i$, $Y = \sum_{i=0}^{n-1} 2^i y_i$ and $Z = \sum_{i=0}^{n-1} 2^i z_i$ be the -bit multiplicand, multiplier and product, respectively of each modulo multiplier of the RNS. Specifically, the residues X,Y and Z are in binary representation with dual zeros,i.e.,0..0,and 1..1 and for the modulo$2^n$ -1 channel and in diminished-1 representation for the modulo$2^n$ +1channel. Conversions to and from these representations will be handled by the RNS forward and reverse converters, respectively.

Therefore, for the modulo2 $^n$-1 and modulo 2 $^n$ multiplications, the binary product Z=X.Y is to be computed from the binary represented inputs, X and Y , whereas for the modulo 2 $^n$+1 multiplication, the diminished-1 product Z is to be computed directly from the diminished-1inputs X and Y. Hence, Z=(X+1).(Y+1)-1=X.Y+X+Yfor m=2 $^m$.+1the multi-modulus multiplication can be expressed as follows.

$$|Z|_m = \begin{cases} |X \cdot Y|_m & \text{if } m = 2^n - 1 \text{ or } m = 2^n \\ |X \cdot Y + X + Y|_m & \text{if } m = 2^n + 1 \end{cases} = \begin{cases} \left| \sum_{i=0}^{n-1} 2^i y_i \cdot X \right|_m & \text{if } m = 2^n - 1 \text{ or } m = 2^n \\ \left| \sum_{i=0}^{n-1} 2^i y_i \cdot X + X + Y \right|_m & \text{if } m = 2^n + 1 \end{cases} \quad (1)$$

In the radix-2 $^k$ Booth encoding algorithm, adjacent multiplier bits $y_{ki+k-1} y_{ki+k-2} \cdots y_{ki+1} y_{ki}$ and an overlap multiplier bit $y_{ki-1}$ are scanned and converted to a signed digit $d_i$, $d_i$ E[-2 $^{k-1}$,2 $^{k-1}$].Thus, the multiplier bits are encoded as[n/k]+1 multiplier digits as follows.

$$d_i = -2^{k-1} y_{ki+k-1} + \sum_{j=0}^{k-2} 2^j y_{ki+j} + y_{ki-1} \quad (2)$$

The translation of $y_i$ to d $_i$ as well as the generation and the additionof the modulo-reduced partial products can be simplifiedfor efficient hardware implementation by the following numbertheoretic properties of modulo arithmetic.

**Property 1***:* Let x denote the bit complement of .The modulo reduction of the binary weight of x in excess of the modulus is given by

$$|2^{n+i} x|_m = \begin{cases} 2^i \bar{x} & \text{if } m = 2^n - 1 \\ 0 & \text{if } m = 2^n \\ |-2^i \bar{x}|_{2^n+1} = 2^i \bar{x} - 2^i & \text{if } m = 2^n + 1 \end{cases} \quad (3)$$

**Property 2:** Let X denote the one's complement of X.by the definition of additive inverse, the modulo negation of X is given by

$$|-X|_m = \begin{cases} \bar{X} & \text{if } m = 2^n - 1 \\ \bar{X} + 1 & \text{if } m = 2^n \\ \bar{X} + 2 & \text{if } m = 2^n + 1 \end{cases} \quad (4)$$

**Property 3:** Let LS(x ,j) denote the left shift of by bit positions where the resultant j least significant bits (lsbs) are zeros. Furthermore, let the circular-left-shift CLS(X, j)and complementary-circular-left-shift CCLS(x,j) operations denote the circular left shift of X by j bit positions, where the resultant j lsbs are not inverted and inverted, respectively. As a corollary of *Property 1*,the modulo multiplication of X by a positive power-of-two term 2 $^j$ is given by

$$|2^j X|_m = \begin{cases} CLS(X, j) & \text{if } m = 2^n - 1 \\ LS(X, j) & \text{if } m = 2^n \\ CCLS(X, j) - 2^j + 1 & \text{if } m = 2^n + 1 \end{cases} \quad (5)$$

**Property 4:**By *Properties 2* and *3*, the modulo multiplication of X by -2 $^j$ is given by

$$|-2^j X|_m = \begin{cases} CLS(\bar{X}, j) & \text{if } m = 2^n - 1 \\ LS(\bar{X}, j) + 2^j & \text{if } m = 2^n \\ CCLS(\bar{X}, j) + 2^j + 1 & \text{if } m = 2^n + 1 \end{cases} \quad (6)$$

**Property 5:** The modulo addition of two operands, X and Y is given by

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

$$|S|_m = \begin{cases} |X+Y|_{2^n-1} = |X+Y+c_{out}|_{2^n} & \text{if } m = 2^n-1 \\ |X+Y|_{2^n} & \text{if } m = 2^n \\ |X+Y+1|_{2^n+1} = |X+Y+\bar{c}_{out}|_{2^n} & \text{if } m = 2^n+1 \end{cases} \quad (7)$$

Where $c_{out}$ is the carry-out bit from the -bit addition of and the binary weight of is modulo-reduced using *Property 1*. It must be highlighted that in modulo $2^n+1$ addition, the diminished-1 result S is the sum of not only the diminished-1addends, X and Y , but also a constant one. Further modulo $2^n$ -1, modulo $2^n$ and modulo $2^n+1$ additions are equivalent to n-bit end-around-carry (EAC), carry-ignore and complementary-end-around-carry (CEAC) additions, respectively.

**Property 6:** The modulo addition of k operands is given by

$$|S|_m = \begin{cases} \left|\sum_{i=0}^{k-1} X_i\right|_m & \text{if } m = 2^n-1 \text{ or } m = 2^n \\ \left|\sum_{i=0}^{k-1} X_i + k - 1\right|_{2^n+1} & \text{if } m = 2^n+1 \end{cases} \quad (8)$$

## III. Radix- Booth Encoded Multi-Modulus Multiplier:

### A. Multi-Modulus Partial Product Generation:

By substituting k=2 in (2), the radix-$2^2$ Booth encoded multiplier is given by

$$\sum_{i=0}^{n-1} 2^i y_i = y_0 - 2y_1 + \sum_{i=1}^{n/2-1} 2^{2i}(y_{2i-1} + y_{2i} - 2y_{2i+1}) + 2^n y_{n-1}$$
$$= 2^n y_{n-1} + y_0 - 2y_1 + \sum_{i=1}^{n/2-1} 2^{2i} d_i \quad (9)$$

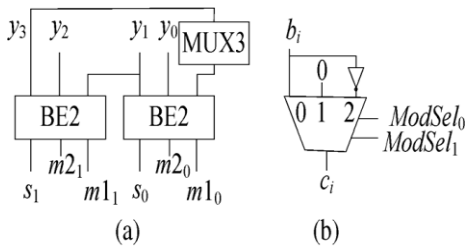


**Fig. 1. (a) Proposed multi-modulus radix- Booth encoder. (b) Proposed 3:1Multiplexer (MUX3).**

where is even. The term $2^n y_{n-1}$ can be modulo-reduced using *Property 1* as follows.

$$|2^n y_{n-1}|_m = \begin{cases} y_{n-1} & \text{if } m = 2^n-1 \\ 0 & \text{if } m = 2^n \\ \bar{y}_{n-1} - 1 & \text{if } m = 2^n+1 \end{cases} \quad (10)$$

Hence, (9) can be simplified to

$$\sum_{i=0}^{n-1} 2^i y_i = \begin{cases} (y_{-1} + y_0 - 2y_1) + \sum_{i=1}^{n/2-1} 2^{2i} d_i & \text{if } m = 2^n-1 \text{ or } m = 2^n \\ -1 + (y_{-1} + y_0 - 2y_1) + \sum_{i=1}^{n/2-1} 2^{2i} d_i & \text{if } m = 2^n+1 \end{cases}$$
$$= \begin{cases} \sum_{i=0}^{n/2-1} 2^{2i} d_i & \text{if } m = 2^n-1 \text{ or } m = 2^n \\ -1 + \sum_{i=0}^{n/2-1} 2^{2i} d_i & \text{if } m = 2^n+1 \end{cases} \quad (11)$$

where

$$y_{-1} = \begin{cases} y_{n-1} & \text{if } m = 2^n-1 \\ 0 & \text{if } m = 2^n \\ \bar{y}_{n-1} & \text{if } m = 2^n+1 \end{cases} \quad (12)$$

encoded digit The radix-$2^2$ Booth encoded digit is formatted using three bits a sign bit $s_i$ and one-hot encoded magnitude bits,$m1_i$ and $m2_i$ .The proposed multi-modulus radix- $2^2$ Booth encoder using n/2 radix-$2^2$ Booth Encoder (BE2) slices and one MUX3 block is shown in Fig. 1(a) for n=4. To select the appropriate $y_{-1}$bit corresponding to the modulus $2^n+1$, $2^n$ or $2^n$ -1,a custom 3:1 multiplexer (MUX3) with 2-bit select input (ModSel $_1$ ModSel $_0$)depicted in Fig. 1(b) is used. The input corresponding to the modulus $2^n+1$, $2^n$ or $2^n$ -1, is selected (ModSel $_1$ ModSel $_0$ )is "00," "01," or "10," respectively.From Fig. 1(b), it can be easily shown that $c_i$ equals and hence, MUX3 can be simplified to an XOR-AND implementation. In what follows, all illustrations of MUX3 block are assumed to include ModSel inputs. Using the radix-$2^2$ Booth encoded multiplier form of (11), the multi-modulus multiplication of (1) becomes

$$|Z|_m = \begin{cases} \left|\sum_{i=0}^{n/2-1} 2^{2i} d_i \cdot X\right|_m & \text{if } m = 2^n-1 \text{ or } m = 2^n \\ \left|\sum_{i=0}^{n/2-1} 2^{2i} d_i \cdot X + Y\right|_m & \text{if } m = 2^n+1 \end{cases} \quad (13)$$

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

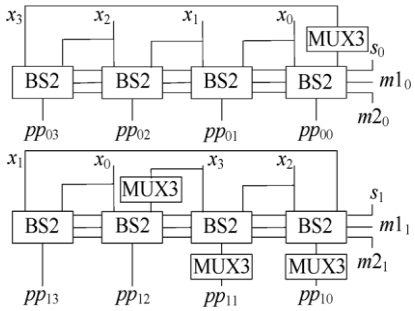


**Fig. 2. Proposed Multi-Modulus Partial Product Generation For Radix- Booth Encoding.**

The radix-$2^2$ Booth encoded multi-modulus multiplication of (13) can then be reformulated as

$$|Z|_m = \begin{cases} \left|\sum_{i=0}^{n/2-1} PP_i\right|_m & \text{if } m = 2^n - 1 \\ \left|\sum_{i=0}^{n/2-1} PP_i + \sum_{i=0}^{n/2-1} K_i\right|_m & \text{if } m = 2^n \\ \left|\sum_{i=0}^{n/2-1} PP_i + \sum_{i=0}^{n/2-1} K_i + Y\right|_m & \text{if } m = 2^n + 1 \end{cases} \quad (17)$$

In the following, the generation of the n/2 $PP_i$s for the three moduli,$2^n+1$, $2^n$ and $2^n-1$,, and is described. The standard circuit implementation of a bit slice of the radix-$2^2$ Booth Selector (BS2) generates a single bit of $PP_i$,pp i.e., , by selecting a bit of either the multiplicand or one-bit shifted multiplicand and conditionally inverting it. Hence, MUX3 of Fig. 1(b) can be used at the output of the BS2 blocks in the least significant 2i bit positions to select from $pp_{i,j}$ 0,or $pp_{i,j}$ bar.Thus this input to the BS2 block is also selected using a MUX3. The proposed multi-modulus generation of the n/2 $PP_i$S using BS2 andMUX3blocksisshowninFig.2for n=4. The number of BS2 and MUX3 blocks required in the $PP_i$ generation is n^2/2 and n^4/2 respectively.

## B. Multi-Modulus Bias Generation:

Analogous to the generation of $PP_i$ using BS2 blocks, the bias $K_i$ can be generated by decoding appropriately. Thistechnique while being straightforward has an obvious limitation.By considering $K_i$ as an -bit word, the n/2 KiS double the number of partial products to be added. Hence, the reduction in the number of partial products achieved by the radixBooth encoding technique is essentially nullified. An efficient technique to include $K_i$ with minimal hardware circuitry based on the properties of modulo $2^n$ and modulo $2^n+1$ arithmeticis proposed.

From the three-variable($s_i$,$m1_i$,$m2_i$) Karnaugh map withdon't care minterms "011" and "111," it can be shown that a=$s_i$ . Hence, the aggregate bias is given by the simplified expression $K_1$ and can be generated by merely hardwiring the $s_i$ output of the BE2 blocks to appropriate bit positions.

$$\left|\sum_{i=0}^{n/2-1} K_i\right|_{2^n} = \sum_{i=0}^{n/2-1} 2^{2i} \cdot a = \sum_{i=0}^{n/2-1} 2^{2i} \cdot s_i = K1 \quad (18)$$

The dynamic bias can be minimized by the three-variable($s_i$,$m1_i$,$m2_j$)Karnaugh map, which results in, a=$m1_I$and b=$s_i$ ,and c=$s_i$ where the operator " ," denotes Boolean AND if its operands are Boolean variables.

$$KD_i = -2^{2i}(m2_i) + 2^{2i+1}(s_i) + 2^{2i+1}(s_i \cdot m2_i) \quad (19)$$

By applying *Property 1* to the negatively weighted term, (19)becomes

$$KD_i = 2^{2i}(\overline{m2_i}) - 2^{2i} + 2^{2i+1}(s_i) + 2^{2i+1}(s_i \cdot m2_i) \quad (20)$$

The next step entails modulo-reduced addition of n/2+3 partial products. From *Property 6*, the modulo $2^n+1$ addition of n/2+3 operands inherently increments the result by n/2+2.To negate this effect, the constant n/2+2 is subtracted from the aggregate bias. The result is

$$\begin{aligned} \left|\sum_{i=0}^{n/2-1} K_i\right|_{2^n+1} &= \sum_{i=0}^{n/2-1} KS_i + \sum_{i=0}^{n/2-1} KD_i - n/2 - 2 \\ &= -n/2 - 2 + \sum_{i=0}^{n/2-1} (-2^{2i} + 1) \\ &\quad + \sum_{i=0}^{n/2-1} (2^{2i} \cdot \overline{m2_i} - 2^{2i} + 2^{2i+1} \\ &\quad \cdot s_i + 2^{2i+1}(s_i \cdot m2_i)) \\ &= \sum_{i=0}^{n/2-1} (2^{2i} \cdot \overline{m2_i} + 2^{2i+1} \\ &\quad \cdot s_i + 2^{2i+1}(s_i \cdot m2_i) + 2^{2i}) \\ &= K1 + K2 \end{aligned} \quad (21)$$

where

$$\begin{aligned} K1 &= \sum_{i=0}^{n/2-1} 2^{2i} \cdot \overline{m2_i} + \sum_{i=0}^{n/2-1} 2^{2i+1} \cdot s_i \\ K2 &= \sum_{i=0}^{n/2-1} 2^{2i} + \sum_{i=0}^{n/2-1} 2^{2i+1}(s_i \cdot m2_i) \end{aligned} \quad (22)$$

Both $K_1$ and $K_2$ can be implemented by hardwiring the BE2 outputs,$m2_i$ and $s_i$ as well as logic one to appropriate bit positionsvia at most one level of AND gates.For m=$2^n+1$, the aggregate bias is composed of two –bit words,k1 and k2 given by (21). For ,m=$2^n$ the aggregate bias is composed of one-bit word given by

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

(18).These three aggregate biases are generalized in (23) as a sum of k1 and k2 given by (24) and (25), respectively, such that k1 and k2 for m=$2^n$ -1as well as for are zero.

$$\left|\sum_{i=0}^{n/2-1} K_i\right|_m = K1 + K2 \qquad (23)$$

$$K1 = \begin{cases} \sum_{i=0}^{n/2-1} 2^{2i}\cdot 0 \\ +\sum_{i=0}^{n/2-1} 2^{2i+1}\cdot 0 & \text{if } m = 2^n - 1 \\ \sum_{i=0}^{n/2-1} 2^{2i}\cdot s_i \\ +\sum_{i=0}^{n/2-1} 2^{2i+1}\cdot 0 & \text{if } m = 2^n \\ \sum_{i=0}^{n/2-1} 2^{2i}\cdot \overline{m2_i} \\ +\sum_{i=0}^{n/2-1} 2^{2i+1}\cdot s_i & \text{if } m = 2^n + 1 \end{cases} \qquad (24)$$

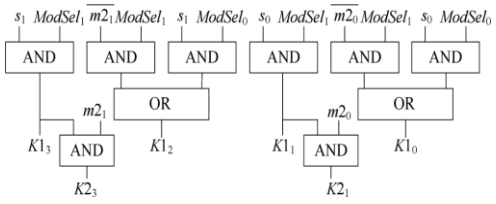


**Fig. 3. Proposed multi-modulus bias generation for radix- Booth encoding.**

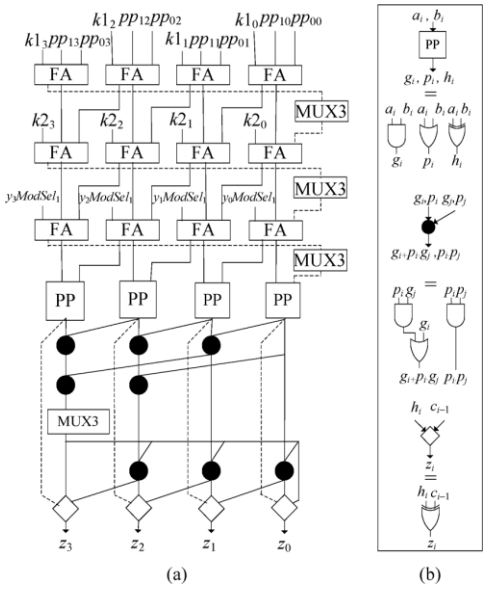


**Fig. 4. (a) Proposed multi-modulus partial product addition for radix- Booth encoding.**

**(b) Implementation of parallel-prefix adder components.**

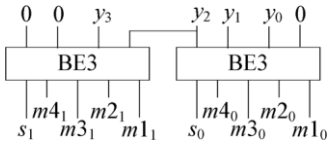


**Fig. 5. Proposed multi-modulus radix- Booth encoder.**

$$K2 = \begin{cases} \sum_{i=0}^{n/2-1} 2^{2i}\cdot 0 \\ +\sum_{i=0}^{n/2-1} 2^{2i+1}\cdot 0 & \text{if } m = 2^n - 1 \\ & \text{or } m = 2^n \\ \sum_{i=0}^{n/2-1} 2^{2i}\cdot 1 \\ +\sum_{i=0}^{n/2-1} 2^{2i+1}(s_i \cdot m2_i) & \text{if } m = 2^n + 1 \end{cases} \qquad (25)$$

Equations (23)–(25) are implemented as shown in Fig. 3 for n=4,k2 $_0$and k2 $_2$ are simply hardwired to Modsel $_1$ andhence, are not illustrated in Fig. 3.

## C. Multi-Modulus Partial Product Addition:

By replacing $|\sum_{i=0}^{n/2-1} K_i|_m$ in (17) with (23), the radix-$2^2$Boothencodedmulti-modulus multiplication is given by (26). In (26), an n-bit all-zeros partial product is included for the moduli $2^n$ -1 and $2^n$ so that the number of partial products to be added is n/2+3 for all three moduli.

$$|Z|_m = \begin{cases} \left|\sum_{i=0}^{n/2-1} PP_i + K1 + K2 + 0\right|_m & \text{if } m = 2^n - 1 \\ & \text{or } m = 2^n \\ \left|\sum_{i=0}^{n/2-1} PP_i + K1 + K2 + Y\right|_m & \text{if } m = 2^n + 1 \end{cases} \qquad (26)$$

From *Property 5*, a multi-modulus addition can be implemented using a MUX3 in the carry feedback path that selects from $c_{out}$,0 or $cbar_{out}$ . The multi-modulus addition of n/2+3 partial products in a CSA tree and a parallel-prefix two-operand adder is illustratedinFig. 4(a) for n=4 .InFig.4(a),theparallel-prefix adder is constructed from the pre-processing (PP),prefix and post-processing blocks and the implementation of these blocks is shown in Fig. 4(b). The number of MUX3 blocks needed for multi-modulus partial product addition is n/2+2.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

## IV. RADIX- BOOTH ENCODED MULTI-MODULUS MULTIPLIER

### A. Multi-Modulus Partial Product Generation:

By substituting k=3 in (2), the radix-2 $^3$ Booth encoded multiplier is given by

$$\sum_{i=0}^{n-1} 2^i y_i = \sum_{i=0}^{\lfloor n/3 \rfloor} 2^{3i}(y_{3i-1}+y_{3i}+2y_{3i+1}-4y_{3i+2}) = \sum_{i=0}^{\lfloor n/3 \rfloor} 2^{3i} d_i \quad (27)$$

Where $y_{-1} = y_n = y_{n+1} = y_{n+2} = 0$.

The radix-2 $^3$ Booth encoded multiplier digit is formatted using five bits: a sign bit $s_I$ and four one-hot encoded magnitude bits,$m1_i$,$m2_i$,$m3_i$and $m4_i$ . n/2+3 radix-2 $^3$booth Encoder (BE3) blocks are used to encode the n-bit multiplier As shown in Fig.5 for n-4.Using the radix-2$^3$Booth encoded multiplier form of (27),the multi-modulus multiplication of (1) becomes

$$|Z|_m = \begin{cases} \left|\sum_{i=0}^{\lfloor n/3 \rfloor} 2^{3i} d_i \cdot X\right|_m & \text{if } m = 2^n - 1 \text{ or } m = 2^n \\ \left|\sum_{i=0}^{\lfloor n/3 \rfloor} 2^{3i} d_i \cdot X + X + Y\right|_m & \text{if } m = 2^n + 1 \end{cases} \quad (28)$$

The radix-2 $^3$ Booth encoded multi-modulus multiplication of (28) can then be reformulated as

$$|Z|_m = \begin{cases} \left|\sum_{i=0}^{\lfloor n/3 \rfloor} PP_i\right|_m & \text{if } m = 2^n - 1 \\ \left|\sum_{i=0}^{\lfloor n/3 \rfloor} PP_i + \sum_{i=0}^{\lfloor n/3 \rfloor} K_i\right|_m & \text{if } m = 2^n \\ \left|\sum_{i=0}^{\lfloor n/3 \rfloor} PP_i + \sum_{i=0}^{\lfloor n/3 \rfloor} K_i + X + Y\right|_m & \text{if } m = 2^n + 1 \end{cases} \quad (29)$$

Fig. 6 shows the generation of n/3+1 PPisusing n([n/3]+1)radix-2$^3$ Booth Selector (BS3) blocks for n=4,where each bit slice of BS3 selects a bit from the multiplicand,the one-bit shifted multiplicand, the hard multiple or the two-bit

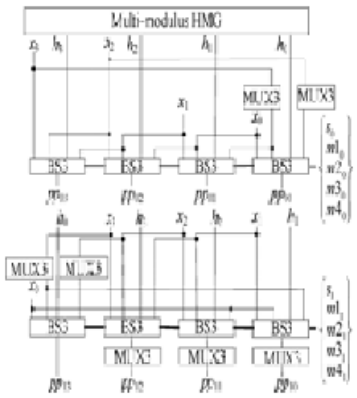


**Fig.6. Proposed multi-modulus partial product generation for radix- Booth encoding.**

### B. Multi-Modulus Hard Multiple Generation:

Application-specific adders known as HMGs that computeonly the sum of Xand $|2X|_m$to generate the hard multiple $|+3X|_m$were proposed in [23] for the moduli $2^n$ -1and $2^n$ +1 .The HMGs were designed by reformulating the carry equationsof modulo $2^n$ -1 and modulo$2^n$ +1 additions using the bitcorrelation between the addends,X and $|2X|_m$. The carry equationsfor modulo $2^n$ -1and modulo $2^n$ +1 HMGs are givenby (30) and (31) for even and odd cases ofi, respectively.

$$c_i = \begin{cases} (g_i^*, p_i^*) \bullet (g_{i-2}^*, p_{i-2}^*) \bullet \cdots \bullet (g_0^*, p_0^*) \\ \bullet (g_{n-2}^*, p_{n-2}^*) \bullet \cdots \bullet (g_{i+2}^*, p_{i+2}^*) & \text{if } m = 2^n - 1 \\ (g_i^*, p_i^*) \bullet (g_{i-2}^*, p_{i-2}^*) \bullet \cdots \bullet (g_0^*, p_0^*) \\ \bullet (\bar{p}_{n-2}^*, \bar{g}_{n-2}^*) \bullet \cdots \bullet (\bar{p}_{i+2}^*, \bar{g}_{i+2}^*) & \text{if } m = 2^n + 1 \end{cases} \quad (30)$$

$$c_i = \begin{cases} (g_i^*, p_i^*) \bullet (g_{i-2}^*, p_{i-2}^*) \bullet \cdots \bullet (g_1^*, p_1^*) \\ \bullet (g_{n-1}^*, p_{n-1}^*) \bullet \cdots \bullet (g_{i+2}^*, p_{i+2}^*) & \text{if } m = 2^n - 1 \\ (g_i^*, p_i^*) \bullet (g_{i-2}^*, p_{i-2}^*) \bullet \cdots \bullet (g_1^*, p_1^*) \\ \bullet (\bar{p}_{n-1}^*, \bar{g}_{n-1}^*) \bullet \cdots \bullet (\bar{p}_{i+2}^*, \bar{g}_{i+2}^*) & \text{if } m = 2^n + 1 \end{cases} \quad (31)$$

Where $(g_i, p_i) \bullet (g_j, p_j) = (g_i + p_i).(g_i, p_i)(g_j, p_j)$and the modified-generate and modified-propagate pair $(g^*_i, p^*_i)$defined asfollows.

$$(g_0^*, p_0^*) = \begin{cases} (x_{n-1} \cdot (x_0 + x_{n-2}), \\ \quad x_{n-1} + (x_0 \cdot x_{n-2})) & \text{if } m = 2^n - 1 \\ (\bar{x}_{n-1} \cdot (x_0 + \bar{x}_{n-2}), \\ \quad \bar{x}_{n-1} + (x_0 \cdot \bar{x}_{n-2})) & \text{if } m = 2^n + 1 \end{cases}$$

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

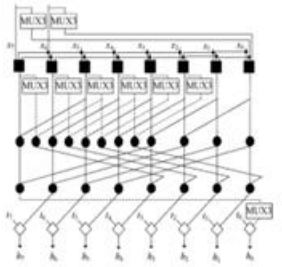

Fig. 7. Proposed multi-modulus HMG.

$$(g_1^*, p_1^*) = \begin{cases} (x_0 \cdot (x_1 + x_{n-1}), \\ \quad x_0 + (x_1 \cdot x_{n-1})) & \text{if } m = 2^n - 1 \\ (x_0 \cdot (x_1 + \bar{x}_{n-1}), \\ \quad x_0 + (x_1 \cdot \bar{x}_{n-1})) & \text{if } m = 2^n + 1 \end{cases}$$
$$(g_i^*, p_i^*) = (x_{i-1} \cdot (x_i + x_{i-2}), x_{i-1} + (x_i \cdot x_{i-2})) \quad (32)$$

As modulo $2^n$ addition is equivalent to carry-ignore addition,the carry equation for modulo $2^n$ HMG becomes

$$c_i = (g_i^*, p_i^*) \bullet (g_{i-2}^*, p_{i-2}^*) \bullet \cdots \bullet (g_0^*, p_0^*) \bullet (0,0) \bullet \cdots \bullet (0,0) \quad (33)$$

Where $(g^*_i, p^*_i)$ is defined as

$$(g_0^*, p_0^*) = (0 \cdot (x_0 + 0), 0 + (x_0 \cdot 0)) = (0,0)$$
$$(g_1^*, p_1^*) = (x_0 \cdot (x_1 + 0), x_0 + (x_1 \cdot 0)) = (x_0 \cdot x_1, x_0)$$
$$(g_i^*, p_i^*) = (x_{i-1} \cdot (x_i + x_{i-2}), x_{i-1} + (x_i \cdot x_{i-2})) \quad (34)$$

From (30), (31), and (33), the parallel-prefix implementation of multi-modulus HMG is illustrated for n=8 in Fig. 7. It consists of three stages: pre-processing, prefix computation and post-processing. In the preprocessing stage, each operator "■" computes using AND-OR and OR-AND gates. The $t_I$ bit is also computed by the XOR operation on the i-th bits of the two addends. Two MUX3 blocks are required at the input of the pre-processing operator to select from $x_{n-1}$,0,$x_{n-1}$or and From $x_{n-2}$ ,0,$x_{n-2}$ for computing$(g_0^*,p_0^*)$ and $(g_1^*,p_1^*)$ the prefix-computation stage,$c_i$is computed using the"●" prefix operators in only $[\log_2 n]$-1 levels.

To implement the multi-modulus carry generation,dual prefix operators are used in the first level such that the input to one prefix operator is$(g_1^*,p_1^*)$while the input to the other to other prefix operator is selected from $(g_i^*,p_i^*)$,(0,0)or $(p_i^*,g_i^*)$ using MUX3blocks.The number of MUX3 blocks required is n-2 and the number of dual prefix operators in each prefix level l is $n-2^{l+1}$ ,where $1 \le l \le [\log_2 n]$-1,In the post-processing stage,each operator "◊ " computes a bit of the hard multiple $h_i$ by $c_{i-1} \oplus t_i$ when i=0,a MUX3 is employed at the $c_{i-1}$ ,0 or $c_{n-1}$ to implement EAC,carry-ignore,and CEAC additions,respectively.

### C. Multi-Modulus Bias Generation:

Using a five-variable($s_i$,$m1_i$,$m2_i$,$m3_i$,$m4_i$) Karnaugh map, the aggregate of the static bias, the dynamic bias and the constant[n/3]-5 that negates the constant incurred from the modulo $2^n$ +1 addition [n/3]+6 of partial products, is reduced to

$$\left| \sum_{i=0}^{\lfloor n/3 \rfloor} K_i \right|_{2^n+1} = \sum_{i=0}^{\lfloor n/3 \rfloor} KS_i + \sum_{i=0}^{\lfloor n/3 \rfloor} KD_i - \lfloor n/3 \rfloor - 5 = K1 + K2 + K3 \quad (36)$$

where

$$K1 = \sum_{i=0}^{\lfloor n/3 \rfloor} \left(2^{3i}(\overline{m2_i + m4_i}) + 2^{3i+1}(\overline{m3_i + m4_i})\right)$$
$$K2 = \sum_{i=0}^{\lfloor n/3 \rfloor} 2^{3i+1}((m2_i + m4_i) \cdot s_i) + \sum_{i=0}^{\lfloor n/3 \rfloor - 1} 2^{3i+2}((m3_i + m4_i) \cdot s_i)$$
$$K3 = \sum_{i=1}^{\lfloor n/3 \rfloor} 2^{3i} \cdot 1 + \sum_{i=0}^{\lfloor n/3 \rfloor} 2^{3i+1} \cdot \bar{s}_i + \sum_{i=0}^{\lfloor n/3 \rfloor - 1} 2^{3i+2} \cdot s_i \quad (37)$$

The operators "+" and "." Denote Boolean OR and A respectively when their operands are Boolean variables.$m=2^n+1$,the aggregate bias is composed of three n-bit words,K1,K2,and K3 given by (36).for $m=2^n$ ,the aggregate bias is composed of one n-bit word K1 given by(35).They can be generalized in(38) as a sum of K1,K2, and K3 given by (39),(40),and (41),respectively,such that K1,K2, and K3 for $m=2^n-1$ as well as K2 and K3 for $m=2^n$ are zero.(see equation at bottom of page.)

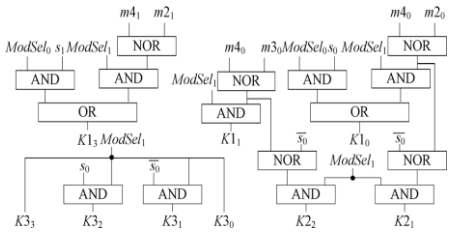


**Fig. 8. Proposed multi-modulus bias generation for radix- Booth encoding.**

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Equations (38)–(41) are implemented as shown in Fig. 8 for n=4.$K1_2$,$K2_0$ and $K2_3$ are hardwired to logic "0" andhence, are not illustrated in Fig. 8.

### D. Multi-Modulus Partial Product Addition:

By replacing $\left|\sum_{i=0}^{\lfloor n/3 \rfloor} K_i\right|_{rm}$ in (29) with (38), the radix-$2^3$Boothencodedmulti-modulus multiplication is given by (42).Two n-bit all-zeros partial products are included for the moduli $2^n$-1 and $2^n$ so that the number of partial products to be added is [n/3]+6 for all three moduli. (See equation at bottom of page.) The multi-modulus addition of [n/3]+6 partial products in a CSA tree and a two-operand adder with MUX3 blocks in

$$\left|\sum_{i=0}^{\lfloor n/3 \rfloor} K_i\right|_m = K1 + K2 + K3 \tag{38}$$

$$K1 = \begin{cases} \sum_{i=0}^{\lfloor n/3 \rfloor} (2^{3i}\cdot 0 + 2^{3i+1}\cdot 0) + \sum_{i=0}^{\lfloor n/3 \rfloor - 1} 2^{3i+2}\cdot 0 & \text{if } m = 2^n - 1 \\ \sum_{i=0}^{\lfloor n/3 \rfloor} (2^{3i}\cdot s_i + 2^{3i+1}\cdot 0) + \sum_{i=0}^{\lfloor n/3 \rfloor - 1} 2^{3i+2}\cdot 0 & \text{if } m = 2^n \\ \sum_{i=0}^{\lfloor n/3 \rfloor} (2^{3i}\overline{(m2_i + m4_i)} + 2^{3i+1}\overline{(m3_i + m4_i)}) + \sum_{i=0}^{\lfloor n/3 \rfloor - 1} 2^{3i+2}\cdot 0 & \text{if } m = 2^n + 1 \end{cases} \tag{39}$$

$$K2 = \begin{cases} \sum_{i=0}^{\lfloor n/3 \rfloor} (2^{3i}\cdot 0 + 2^{3i+1}\cdot 0) + \sum_{i=0}^{\lfloor n/3 \rfloor - 1} 2^{3i+2}\cdot 0 & \text{if } m = 2^n - 1 \text{ or } m = 2^n \\ \sum_{i=0}^{\lfloor n/3 \rfloor} (2^{3i}\cdot 0 + 2^{3i+1}((m2_i + m4_i)\cdot s_i)) + \sum_{i=0}^{\lfloor n/3 \rfloor - 1} 2^{3i+2}((m3_i + m4_i)\cdot s_i) & \text{if } m = 2^n + 1 \end{cases} \tag{40}$$

$$K3 = \begin{cases} \sum_{i=0}^{\lfloor n/3 \rfloor} (2^{3i}\cdot 0 + 2^{3i+1}\cdot 0) + \sum_{i=0}^{\lfloor n/3 \rfloor - 1} 2^{3i+2}\cdot 0 & \text{if } m = 2^n - 1 \text{ or } m = 2^n \\ \sum_{i=1}^{\lfloor n/3 \rfloor} 2^{3i}\cdot 1 + \sum_{i=0}^{\lfloor n/3 \rfloor} 2^{3i+1}\cdot s_i + \sum_{i=0}^{\lfloor n/3 \rfloor - 1} 2^{3i+2}\cdot s_i & \text{if } m = 2^n + 1 \end{cases} \tag{41}$$

$$|Z|_m = \begin{cases} \left|\sum_{i=0}^{\lfloor n/3 \rfloor} PP_i + K1 + K2 + K3 + 0 + 0\right|_m & \text{if } m = 2^n - 1 \text{ or } m = 2^n \\ \left|\sum_{i=0}^{\lfloor n/3 \rfloor} PP_i + K1 + K2 + K3 + X + Y\right|_m & \text{if } m = 2^n + 1 \end{cases} \tag{42}$$

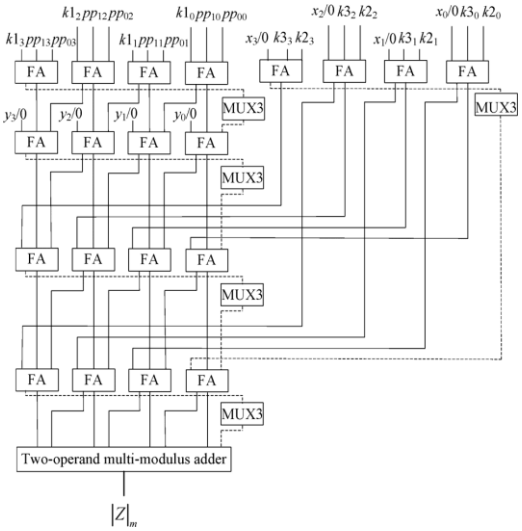


**Fig. 9. Proposed multi-modulus partial product addition for radix- Booth encoding.**

the carry feedback paths is illustrated in Fig. 9 for n-4.The number of MUX3 blocks needed is[n/3]+5.

## V.SIMULATION RESULTS:

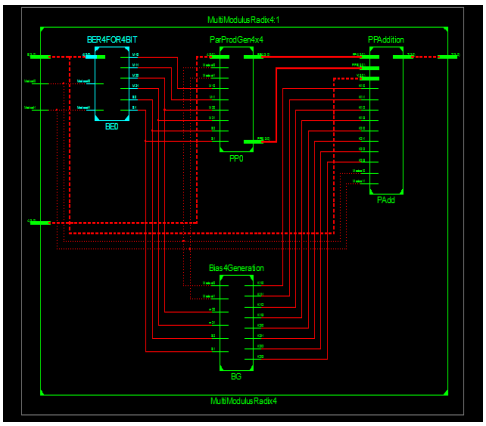

**Fig:RTL of 4-4bit**

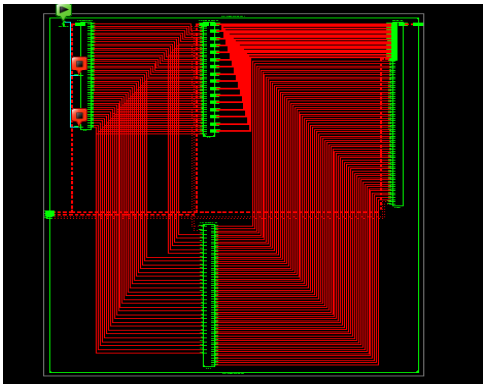

**Fig:RTL of 4-32 bit**

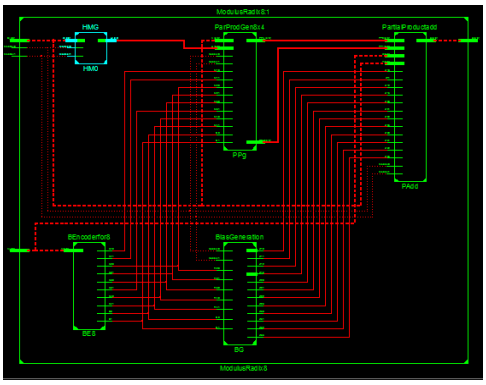

**Fig:RTL of 8-4bit**

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

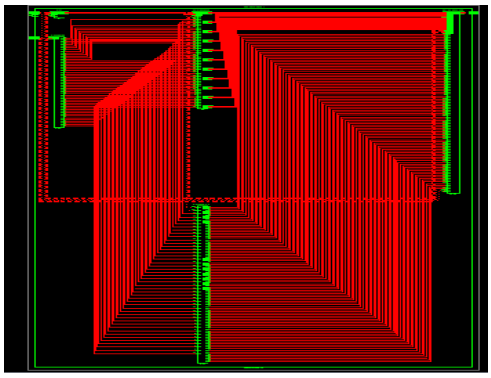

Fig:RTL of 8-32 bit

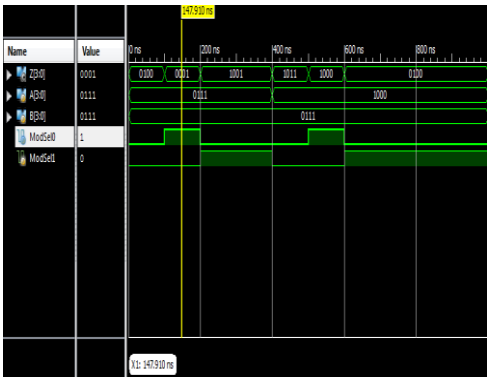

Fig:waveform of 8-4bit

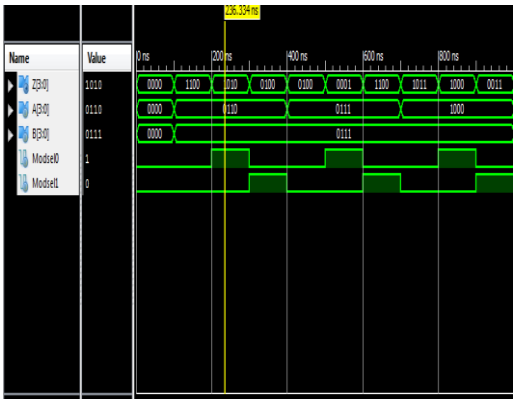

Fig:waveform of 4-4bit

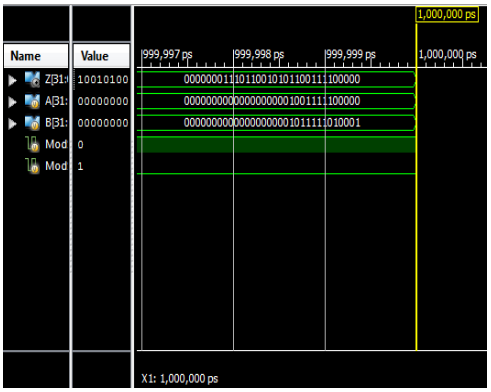

Fig:waveform of 8-32 bit

Fig:waveform of 4-32bit

## VI. Conclusion:

The equivalences in operations central to modulo multiplication i.e. modulo negation, modulo reduction of binary weight and modulo multiplication by powers-of-two and two-operand modulo addition for the three special moduli,$2^n$ -1,$2^n$and$2^n$+1 were demonstrated.New radix-$2^2$ and radix-$2^3$ Boothencoded modulo multipliers with architectures comparable to those of the corresponding modulo $2^n$ -1 and modulo $2^n$ +1 multipliers were introduced. With the correlation among modulo $2^n$ -1, modulo $2^n$ and modulo $2^n$ +1 operations as the basis, radix-$2^2$ and radix- $2^3$ Booth encoded multi-modulus multipliers that perform modulo multiplication for the three special moduli successively were developed. The proposed multi-modulus multipliers were compared against RNS multipliers employing Booth encoding of the same radix.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

## REFERENCES:

[1] R. Conway and J. Nelson, "Improved RNS FIR filter architectures,"*IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 51, no. 1, pp. 26–28,Jan. 2004.

[2] A. S. Madhukumar and F. Chin, "Enhanced architecture for residue number system-based CDMA for high rate data transmission," *IEEETrans. Wireless Commun.*, vol. 3, no. 5, pp. 1363–1368, Sep. 2004.

[3] D.M.Schinianakis*et al.*, "An RNS implementation of an Elliptic curve point multiplier," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 6, pp. 1202–1213, Jun. 2009.

[4] N.S.SzaboandR.I. Tanaka, *Residue Arithmetic and its Application to Computer Technology*. New York: McGraw-Hill, 1967.

[5] B.Cao,C.H.Chang,andT.Srikanthan,"An efficient reverse converter for the 4-moduli set{$2^n$,$2^n$-1,$2^n$+1,$2^{2n}$+1} based on the NewChinese Remainder Theorem," *IEEE Trans. Circuits Syst. I, Fundam.Theory Appl.*, vol. 50, no. 10, pp. 1296–1303, Oct. 2003.

[6] V. Paliouras and T. Stouraitis, "Multifunction architectures for RNS processors," *IEEE Trans. Circuits Syst. II, Analog Digit. SignalProcess.*, vol. 46, no. 8, pp. 1041–1054, Aug. 1999.

[7] D. Adamidis and H. T. Vergos, "RNS multiplication/sum-of-squares units," *IET Comput. Digit. Tech.*,vol.1,no.1,pp.38–48,Jan.2007.

[8] G. C. Cardarilli, A. Del Re, A. Nannarelli, and M. Re, "Residue Number System reconfigurable datapath," in *Proc. IEEE Int. Symp.Circuits Syst.*, Scottsdale, AZ, USA, May 2002, vol. 2, pp. 756–759.

[9] W. K. Jenkins and A. J. Mansen, "Variable word length DSP using se-rial-by-modulus residue arithmetic," in *Proc. IEEE Int. Conf. Acoust.,Speech, Signal Process.*, Minneapolis, MN, USA, Apr. 1993, vol. 3,pp. 89–92.

[10] W. K. Jenkins and B. A. Schnaufer, "Fault Tolerant architectures for efficient realization of common DSP kernels," in *Proc. 35th MidwestSymp. Circuits Syst.*, Washington, DC, USA, Aug. 1992, vol. 2, pp. 1320–1323.

[11] W. K. Jenkins, B. A. Schnaufer, and A. J. Mansen, "Combined systemlevel redundancy and modular arithmetic for fault tolerant digital signal processing," in *Proc. 11th Symp. Comput. Arithmetic*, Windsor,Canada, Jun. 1993, pp. 28–35.

[12] H. T. Vergos and D. Bakalis, "Area-time efficient multi-modulus adders and their applications," *Microprocessors Microsyst.*, vol. 36, no. 5, pp. 409–419, Jul. 2012.

[13] C. H. Chang, S. Menon, B. Cao, and T. Srikanthan, "A configurable dual-moduli multi-operand modulo adder," in *Proc. IEEE Int. Symp.Circuits Syst.*, Kobe, Japan, May 2005, vol. 2, pp. 1630–1633.