International Journal of Engineering Research and Development
e-ISSN: 2278-067X, **p**-ISSN : 2278-800X, www.ijerd.com
Volume 5, Issue 1 (November 2012), PP. 38-45

Radix-8 Booth Encoded Modulo $2^n - 1$ Multipliers with Parallel Prefix Adder for High Dynamic Range Residue Number System

K. Bhaskara Rao[1], B.Chinna rao,(Ph.D)[2], P.M.Francis[3],
[1]M.tech(PG student), Dept. of ECE, GITAS
[2]Prof.&Head,Dept.of ECE, GITAS
[3]Asst.Prof.,M.Tech,

Abstract:- A special moduli set Residue Number System (RNS) of high dynamic range (DR) can speed up the execution of very large word-length repetitive multiplications found in applications like public key cryptography. The modulo $2^n - 1$ multiplier is usually the noncritical datapath among all modulo multipliers in such high-DR RNS multiplier. This timing slack can be exploited to reduce the system area and power consumption without compromising the system performance. With this precept, a family of radix-8 Booth encoded modulo $2^n - 1$ multipliers, with delay adaptable to the RNS multiplier delay, is proposed. The modulo $2^n - 1$ multiplier delay is made scalable by controlling the word-length of the ripple carry adder, *k* employed for radix-8 hard multiple generation. Formal criteria for the selection of the adder word-length are established by analyzing the effect of varying *k* on the timing of multiplier components. It is proven that for a given *n*, there exist a number of feasible values of *k* such that the total bias incurred from the partially-redundant partial products can be counteracted by only a single constant binary string. This compensation constant for different valid combinations of *n* and *k* can be precomputed at design time using number theoretic properties of modulo $2^n - 1$ arithmetic and hardwired as a partial product to be accumulated in the carry save adder tree. The adaptive delay of the proposed family of multipliers is corroborated by CMOS implementations. In an RNS multiplier, when the critical modulo multiplier delay is significantly greater than the noncritical modulo $2^n - 1$ multiplier delay, $k = n$ and $k = \frac{n}{3}$ are recommended for *n* not divisible by three and divisible by three, respectively. Conversely, when this difference diminishes, *k* is better selected as $n/4$ and $n/6$ for *n* not divisible by three and divisible by three, respectively. Our synthesis results show that the proposed radix-8 Booth encoded modulo $2^n - 1$ multiplier saves substantial area and power consumption over the radix-4 Booth encoded multiplier in medium to large word-length RNS multiplication.

Index Terms:- Booth algorithm, design space exploration, modulo arithmetic, multiplier, residue number system (RNS).

I. INTRODUCTION

RIVEST, Shamir, and Adleman (RSA) and elliptic curve cryptography (ECC) are two of the most well established and widely used public key cryptographic (PKC) algorithms. The encryption and decryption of these PKC algorithms are performed by repeated modulo multiplications [1]–[3]. These multiplications differ from those encountered in signal processing and general computing applications in their sheer operand size. Key sizes in the range of 512~1024 bits and 160~512 bits are typical in RSA and ECC, respectively [4]–[7]. Hence, the long carry propagation of large integer multiplication is the bottleneck in hardware implementation of PKC. The residue number system (RNS) has emerged as a promising alternative number representation for the design of faster and low power multipliers owing to its merit to distribute a long integer multiplication into several shorter and independent modulo multiplications [8]–[11]. RNS has also been successfully employed to design fault tolerant digital circuits [12], [13].

To transcend the 3n-bit limit, moduli sets, $\{2^n - 1, 2^n, 2^n + 1, 2^{2n} + 1\}$, $\{2^{2n} - 1, 2^n, 2^{2n} + 1\}$ and $\{2^n - 1, 2^n, 2^n + 1, 2^{2n+1} - 1\}$ with DR of 5n bits and $\{2^n - 1, 2^n + 1, 2^{2n}, 2^{2n} + 1\}$ with DR of 6n bits, have been proposed recently [21], [30], [31]. consequently, a RSA cryptosystem with a conservative key-length of 512 bits could be implemented in RNS using either the 5n-bit DR moduli set with *n* = 100 or the 6n -bit DR moduli set with *n* = 85 . For a ECC cryptosystem with a typical key-length of 256 bits, either the 5n-bit DR moduli set with *n* = 50 or the 6n -bit DR moduli set with *n* = 42 could be chosen.

The delay of an integer multiplication in RNS domain based on the -bit DR moduli set of [31] for example, is governed by the delay of the modulo 2^{2n}+1 multiplier. As the time complexity of partial product summation by a carry save adder (CSA) tree and a two-operand parallel-prefix adder is a logarithmic function of , the critical path delay can be modeled as O(log2n), but the delays of the modulo 2^n-1 and modulo 2^n+1 multipliers are only O(log2n) . This speedup of around $1/(1 + \log_2 n) \times 100\%$ by modulo 2^n-1 and 2^n+1 modulo multipliers over the critical path delay is of no consequence. As encryption and decryption in PKC involves repeated multiplications, the cumulative difference in the critical and noncritical modulo multiplier delays will increase with the number of multiplications involved. For lightweight cryptographic applications, such as smartcards and radio frequency identification (RFID) tags, the considerations of power, size and cost are of paramount importance [32]. The complexity of implementing reliable cryptographic hardware can be reduced by an ingenious exploitation of this timing headroom in the design of RNS multiplier.

This paper focuses on the design space exploration of arithmetic operation in one of the two special moduli, i.e., the modulo 2^n-1 multiplier design. The Montgomery modulo multiplication, while computing the modular product without trial division, is modulus-independent and incapable of exploiting number theoretic properties of modulo 2^n-1 arithmetic for combinational circuit simplification. The properties of modulo 2^n-1 arithmetic were most effectively exploited for the full adder based implementation of modulo multiplier in [36]–[38]. In [36], the multiplier bits were not encoded, which lead to higher implementation area and longer partial product accumulation time. In [37] and [38], the radix-4 Booth encoding algorithm was employed to reduce the mber of partial products to [n/2]+1 and [n/2] , respectively. The shorthand notations [a] and [a] denote the smallest integer greater than or equal to and the largest integer smaller than or equal to , respectively. With higher radix Booth encoding, the number of partial products is reduced by more than half and consequently, significant reduction in silicon area and power dissipation is feasible [39], [40]. The radix-8 Booth encoding reduces the number of partial products to [n/3] , which is more aggressive than the radix-4 Booth encoding. However, in the radix-8 Booth encoded modulo 2^n-1 multiplication, not all modulo-reduced partial products can be generated using the bitwise circular-left-shift operation and bitwise inversion. Particularly, the hard multiple $+3X_{2^n-1}$ is to be generated by an n-bit end-around-carry addition of X and 2X. The performance overhead due to the end-around-carry addition is by no means trivial and hence, the use of Booth encoding for modulo 2^n-1 multipliers have been restricted to only radix-4 in literature.

The paper is organized as follows. Section II describes the radix-8 Booth encoding algorithm for modulo $\mathbf{2^n-1}$ multiplication. A family of modulo $\mathbf{2^n-1}$ multipliers to adapt to different RNS delay is described in Section III. In Section IV, the criteria for selecting a suitable RCA word-length to achieve the desired performance are highlighted. The performance of the proposed family of modulo $\mathbf{2^n-1}$ multipliers is evaluated and compared against [38] in Section V. The paper is concluded in Section VI. The Appendix provides the derivation of the predetermined compensation constant for different valid combinations of the multiplier and RCA word-lengths.

II. RADIX-8 BOOTH ENCODED MODULO $\mathbf{2^n-1}$ MULTIPLICATION ALGORITHM

Let X = $\sum_{i=0}^{n-1} x_i \cdot 2^i$ and Y = $\sum_{i=0}^{n-1} y_i \cdot 2^i$ represent the multiplicand and the multiplier of the modulo $\mathbf{2^n-1}$ multiplier, respectively. The radix-8 Booth encoding algorithm can be viewed as a digit set conversion of four consecutive overlapping multiplier bits, $y_{3i+2}y_{3i+1}y_{3i}(y_{3i-1})$ to a signed digit. The digit set conversion is formally expressed as

$$d_i = y_{3i-1} + y_{3i} + 2y_{3i+1} - 4y_{3i+2} \qquad (1)$$

$$\text{where } y_{-1} = y_n = y_{n+1} = y_{n+2} = 0 \text{ [40].}$$

TABLE I
MODULO-REDUCED MULTIPLES FOR THE RADIX-8 BOOTH ENCODING

d_i	$\lvert d_i \cdot X\rvert_{2^n-1}$	d_i	$\lvert d_i \cdot X\rvert_{2^n-1}$
+0	$\underbrace{0\cdots0}_{n}$	−0	$\underbrace{1\cdots1}_{n}$
+1	X	−1	$\overline{X}$
+2	$CLS(X, 1)$	−2	$CLS(\overline{X}, 1)$
+3	$\lvert+3X\rvert_{2^n-1}$	−3	$\lvert-3X\rvert_{2^n-1}$
+4	$CLS(X, 2)$	−4	$CLS(\overline{X}, 2)$

Table I summarizes the modulo-reduced multiples of X for all possible values of the radix-8 Booth encoded multiplier digit, d_i , where CLS[X,j] denotes a circular-left-shift of X by j – bit positions.

Three unique properties of modulo 2^n-1 arithmetic that will be used for simplifying the combinatorial logic circuit of the proposed modulo multiplier design are reviewed here.

1) ***Property 1:*** The modulo 2^n-1 reduction of –X can be implemented as the n-bit one's complementation of the binary word X as follows:

$$-X_{2^n-1} = 2^n - 1 - X = \overline{X}. \qquad (2)$$

2) ***Property 2:*** For any nonnegative integer,s, the periodicity of an integer power of two over modulus 2^n-1 can be stated as follows [41]:

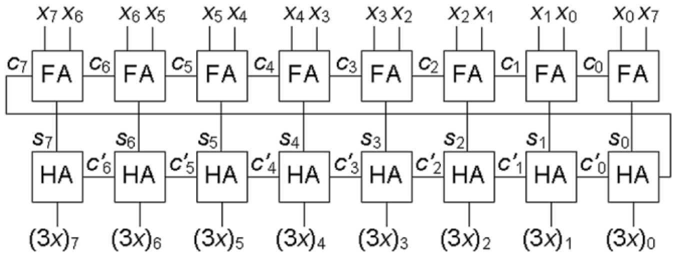
$$\left|2^{n\cdot s+i}\right|_{2^n-1} = \left|\left|2^{n.s}\right|_{2^n-1} \cdot 2^i\right|_{2^n-1\; 2^n-1} = 2^i{}_{2^n-1}. \qquad (3)$$

Property 2 ensures that the modulo 2^n-1 reduction of binary exponents can be implemented with no logic cost. As a corollary, the modulo 2^n-1 reduction of the product of a binary word X and an integer power of two,2^j , is equivalent to CLS[X,j] [14]. This property can be formally expressed as *Property 3*.

3) Property 3: For j < n

$$|2^j X|_{2^n-1} = \sum_{i=0}^{n-j-1} x_i \cdot 2^{i+j} + \sum_{i=n-j}^{n-1} x_i \cdot 2^{i+j-n} = CLS(X,j). \tag{4}$$

In Table I, the modulo 2^n-1 reduction for $d_i$are replaced by simple bitwise inversion and bitwise circular-left-shift of X using *Properties 1* and *3,* respectively.



$x_7\ x_6$
$x_6\ x_5$
$x_5\ x_4$
$x_4\ x_3$
$x_3\ x_2$
$x_2\ x_1$
$x_1\ x_0$
$x_0\ x_7$
FA
c_7 c_6 c_5 c_4 c_3 c_2 c_1 c_0
s_7 s_6 s_5 s_4 s_3 s_2 s_1 s_0
HA
c'_6 c'_5 c'_4 c'_3 c'_2 c'_1 c'_0
$(3x)_7$ $(3x)_6$ $(3x)_5$ $(3x)_4$ $(3x)_3$ $(3x)_2$ $(3x)_1$ $(3x)_0$

Fig. 1. Generation of $|+3X|_{2^n-1}$ using two _-bit RCAs.

The above technique for $+3X_{2n-1}$computation involves two -bit carry-propagate additions in series such that the carry propagation length is twice the operand length, n . In the worst case, the late arrival of the $+3X_{2n-1}$ may considerably delay all subsequent stages of the modulo 2^n-1 multiplier. Hence, this approach for hard multiple generation can no longer categorically ensure that the multiplication in the modulo 2^n-1 channel still falls in the noncritical path of a RNS multiplier.

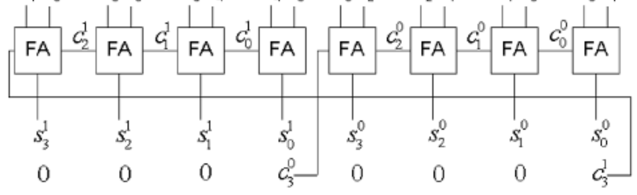
III. PROPOSED RADIX-8 BOOTH ENCODED MODULO MULTIPLIER DESIGN

To ensure that the radix-8 Booth encoded modulo 2^n-1 multiplier does not constitute the system critical path of a high-DR moduli set based RNS multiplier, the carry propagation length in the hard multiple generation should not exceed n-bits. To this end, the carry propagation through the HAs in Fig. 1 can be eliminated by making the end-around-carry bit a partial product bit to be accumulated in the CSA tree. This technique reduces the carry propagation length to n- bits by representing the hard multiple as a sum and a redundant end-around-carry bit pair. The resultant [n/3]+1 end-around-carry bits C_7 in the partial product matrix may lead to a marginal increase in the CSA tree depth and consequently, may aggravate the delay of the CSA tree. In which case, it is not sufficient to reduce the carry propagation length to merely n-bits using the above technique.

Since the absolute difference between the noncritical modulo 2^n-1 multiplier delay and the system critical path delay depends on the degree of imbalance in the moduli word-length of a RNS, the delays cannot be equalized by arbitrarily fixing the carry propagation length to n-bits. Instead, we propose to accomplish the adaptive delay equalization by representing the hard multiple in a partially-redundant form [48].

A. Generation of Partially-Redundant Hard Multiple

Let $1X_{2n-1}$ and $2X_{2n-1}$ be added by a group of M(n/k)k-bit RCAs such that there is no carry propagation between the adders. Fig. 2 shows this addition for n= 8 and k = 4 , where the sum and carry-out bits from the RCA block are represented as S_i^j and C_i^j respectively. In Fig. 2, the carry-out of RCA 0,C_3^0 , is not propagated to the carry input C_3^1 of RCA 1 but preserved as one of the partial product bits to be accumulated in the CSA tree. The binary weight of the carry-out of RCA 1 has, however, exceeded the maximum range of the modulus and has to be modulo reduced before it can be accumulated by the CSA tree.



$x_7\ x_6$
$x_6\ x_5$
$x_5\ x_4$
$x_4\ x_3$
$x_3\ x_2$
$x_2\ x_1$
$x_1\ x_0$
$x_0\ x_7$
FA
c_2^1 c_1^1 c_0^1 c_2^0 c_1^0 c_0^0
s_3^1 s_2^1 s_1^1 s_0^1 s_3^0 s_2^0 s_1^0 s_0^0
0 0 0 c_3^0 0 0 0 c_3^1

Fig. 2. Generation of partially-redundant $|+3X|_{2^n-1}$ using k-bit RCAs.

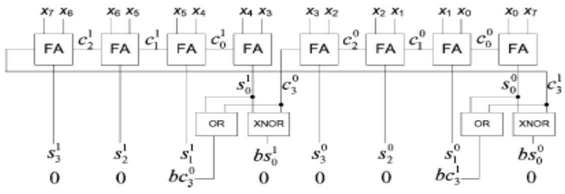


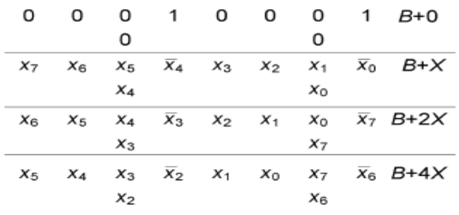
Fig. 3. Generation of partially-redundant $|B+3X|_{2n-1}$.

By *Property 2*, the binary weight of C_3^1 can be reduced from 2^8 to 2^0 . Thus, C_3^1 is inserted at the least significant bit (lsb) position in Fig. 2. It should be stressed that the carry-out C_3^1 is a partial carry propagated through only most significant FAs and hence, is different from the end-around-carry bit in the modulo 2^n-1 addition of X and 2X *i.e.,*C_7 of Fig. 1
From Fig. 2, the partially-redundant form of $1+3X_{2n-1}$ is given by the partial-sum and partial-carry pair (S,C)where

$$S = s_{k-1}^{M-1} s_{k-2}^{M-1} \cdots s_0^{M-1} \cdots s_{k-1}^{0} s_{k-2}^{0} \cdots s_0^{0}$$
$$C = \underbrace{0\cdots0}_{k-1} c_{k-1}^{M-2} \cdots \underbrace{0\cdots0}_{k-1} c_{k-1}^{0} \underbrace{0\cdots0}_{k-1} c_{k-1}^{M-1}. \qquad (5)$$

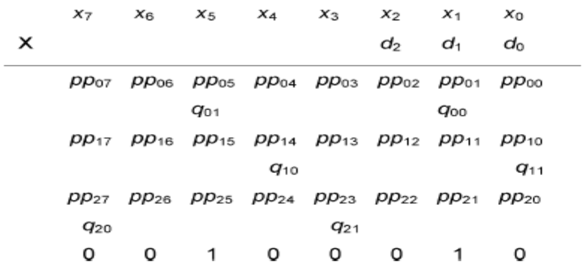
Since modulo 2^n-1 negation is equivalent to bitwise complementation by *Property 1*, the negative hard multiple in a partially- redundant form, $1-3X1_{2n-1}$ = (S,C) , is computed as follows:

$$\bar{S} = \bar{s}_{k-1}^{M-1} \bar{s}_{k-2}^{M-1} \cdots \bar{s}_0^{M-1} \cdots \bar{s}_{k-1}^{0} \bar{s}_{k-2}^{0} \cdots \bar{s}_0^{0}$$
$$\bar{C} = \underbrace{1\cdots1}_{k-1} \bar{c}_{k-1}^{M-2} \cdots \underbrace{1\cdots1}_{k-1} \bar{c}_{k-1}^{0} \underbrace{1\cdots1}_{k-1} \bar{c}_{k-1}^{M-1}. \qquad (6)$$



0	0	0	1	0	0	0	1	$B+0$
		0				0		
x_7	x_6	x_5	$\bar{x}_4$	x_3	x_2	x_1	$\bar{x}_0$	$B+X$
		x_4				x_0		
x_6	x_5	x_4	$\bar{x}_3$	x_2	x_1	x_0	$\bar{x}_7$	$B+2X$
		x_3				x_7		
x_5	x_4	x_3	$\bar{x}_2$	x_1	x_0	x_7	$\bar{x}_6$	$B+4X$
		x_2				x_6		

Fig. 4. Generation of partially-redundant simple multiples.



	x_7	x_6	x_5	x_4	x_3	x_2	x_1	x_0
×						d_2	d_1	d_0
	pp_{07}	pp_{06}	pp_{05}	pp_{04}	pp_{03}	pp_{02}	pp_{01}	pp_{00}
			q_{01}				q_{00}	
	pp_{17}	pp_{16}	pp_{15}	pp_{14}	pp_{13}	pp_{12}	pp_{11}	pp_{10}
				q_{10}				q_{11}
	pp_{27}	pp_{26}	pp_{25}	pp_{24}	pp_{23}	pp_{22}	pp_{21}	pp_{20}
	q_{20}				q_{21}			
	0	0	1	0	0	0	1	0

Fig. 5. Modulo-reduced partial products and CC for $|X \cdot Y|_{2^8-1}$.

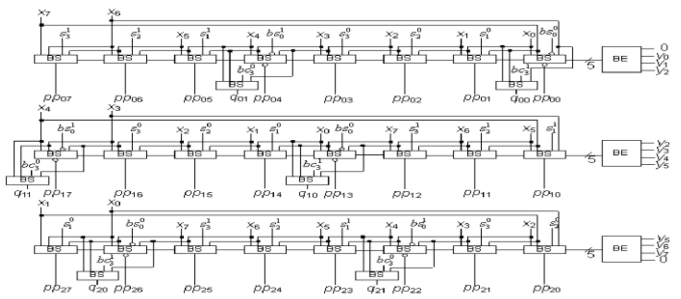


Fig. 6. Modulo-reduced partial product generation.

B. Generation of Partially-Redundant Simple Multiples

The proposed technique represents the hard multiple in a biased partially-redundant form. Since the occurrences of the hard multiple cannot be predicted at design time, all multiples must be uniformly represented. Similar to the hard multiple, all other Booth encoded multiples listed in Table I must also be biased and generated in a partially-redundant form. Fig. 4 shows the biased simple multiples, represented in a partially-redundant form for . From Fig. 4, it can be seen that the generation of these biased multiples involves only shift and selective complementation of the multiplicand bits without additional hardware overhead.

C. Radix-8 Booth Encoded Modulo 2^n-1 Multiplication With Partially-Redundant Partial Products

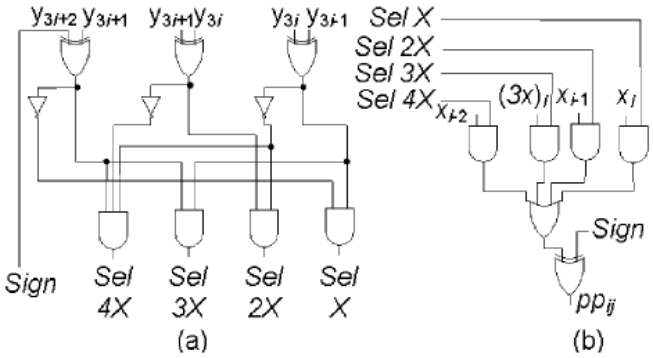
The i-th partial product of a radix-8 Booth encoded modulo **2^n-1** multiplier is given by

$$PP_i = \left. 2^{3i} \cdot d_i \cdot X \right|_{2^n-1}. \quad (12)$$

To include the bias B necessary for partially-redundant representation of PPi, (12) is modified to

$$PP_i = \left| 2^{3i} \left(B + d_i \cdot X \right) \right|_{2^n-1}. \quad (13)$$

Using *Property 3*, the modulo 2^n-1 multiplication by in (13) is efficiently implemented as bitwise circular-left-shift of the iased multiple,(B+di.X) . For n=8 and k= 4, Fig. 5 illustrates the partial product matrix of X.Y$2^8 - 1$ with ([n/3]+1) partial products in partially-redundant representation. Each consists of an -bit vector, and a vector of redundant carry bits, and . Since and are the carry-out bits of the RCAs, they are displaced by -bit positions for a given . The bits, is displaced circularly to the left of by 3 bits, *i.e.*, and are displaced circularly to the left of and by 3 bits, respectively and and are in turn displaced to the left of and by 3 bits, respectively. The last partial product in Fig. 5 is the Compensation Constant (*CC*) for the bias introduced in the partially- redundant representation. The derivation of this constant is detailed in Section IV and the Appendix.



y_{3i+2} y_{3i+1} y_{3i+1} y_{3i} y_{3i} y_{3i-1}
Sel X
Sel 2X
Sel 3X
Sel 4X
x_{i-2} $(3x)_i$ x_{i-1} x_i
Sign
Sign
Sel 4X
Sel 3X
Sel 2X
Sel X
pp_{ij}
(a)
(b)

Fig. 6. (a) Bit-slice of Booth Encoder (BE). (b) Bit-slice of Booth Selector (BS)

The generation of the modulo-reduced partial products,PP0 , PP1 and PP2, in a partially-redundant representation using Booth Encoder (BE) and Booth Selector (BS) blocks are illustrated in Fig. 5. The BE block produces a signed one-hot encoded digit from adjacent overlapping multiplier bits as illustrated in Fig. 6(a). The signed one-hot encoded digit is then used to select the correct multiple to generate PPi . A bit-slice of the radix-8 BS for the partial product bit, is shown in Fig. 6(b).

As the bit positions of do not overlap, as shown in Fig. 4, they can be merged into a single partial product for accumulation. The merged partial products, and the constant *CC* are accumulated using a CSA tree with end-around-carry addition at each CSA level and a final two-operand modulo 2^n-1 adder as shown in Fig. 7.

IV. SELECTION OF *K*

The guidelines for choosing the RCA word-length, *k*, to achieve the desired performance are presented in this section.

Firstly, irrespective of the targeted delay, the choice of must satisfy the following two criteria.

1) Criterion 1: As the residues of modulus 2^n-1 are represented using only bits, it is imperative that divides is a trivial case and is excluded from this consideration. This criterion is expressed.

2) Criterion 2: Since each partial product in radix-8 Booth encoding is shifted by three bits relative to the previous partial product, must not be a multiple of three to ensure that the bits are nonoverlapping.

In the proposed modulo 2^n-1 multiplier, each partial product is incremented by a bias of as expressed in (13). To negate the effect of the bias, a constant *CC* is added and the value of *CC* is given by

$$CC = -\sum_{i=0}^{\lfloor n/3 \rfloor} B \cdot 2^{3i} \Bigg|_{2^n-1} \quad (14)$$

where B is an -bit binary word consisting of logic one at bit position 2kj , and logic zero at all other positions as defined in (7).

It is evident that the value of *CC* depends only on and . As *CC* is considered as one or more partial products to be summed in the CSA tree, the choice of indirectly determines the regularity of the multiplier design and consequently its efficiency in VLSI implementation. A detailed analysis on the computation of *CC* for various combinations of n and k is presented in the Appendix. For any that satisfies *Criteria 1* and *2*, it is shown that *CC* can be simplified by the properties of modulo 2^n-1 arithmetic and precomputed at design time. The resultant *CC* is shown to be a single binary word with a specific repetitive pattern of logic ones and zeros. As the generation of *CC* involves merely the assignment of logic constants to appropriate bit positions, it can be directly hardwired into the CSA tree as a constant partial product without any logic circuitry.

The effect of *k* on the delay of the constituent components of a radix-8 Booth encoded modulo 2^n-1 multiplier is analyzed qualitatively and summarized in Table II. As indicated in Table II, the partial products and *CC* can be generated in constant time. Similarly the delay of the final two-operand parallel- prefix modulo 2^n-1 adder is independent of . From Table II, by reducing , the delay of the RCA reduces linearly but the delay of the CSA tree stage increases only logarithmically. Hence, the delay of the modulo 2^n-1 multiplier is logarithmically dependent on *n* and almost linearly dependent on . For a given , the modulo 2^n-1 multiplier delay can be manipulated by varying the word-length of the RCA, . In the following section, we show by means of synthesis results how the modulo multiplier delay can be matched to the RNS delay to save silicon area and reduce power dissipation.

V. PERFORMANCE COMPARISON

In this section, we evaluate the performance of the proposed family of partially-redundant modulo 2^n-1 multipliers with different suitably chosen RCA word-length, *k* . The proposed multipliers are also compared against the recent radix-4 Booth encoded modulo 2^n-1 multiplier [38].

For experimental analysis, *k* is selected as *n* , *n/2* and *n/4* to satisfy *Criteria 1* and *2* when *n* is not divisible by three. When *n* is divisible by three but not by higher powers of three, *k* is selected as *n/3* and *n/6*. As M partial-carry bits are introduced by each partially-redundant partial product, the number of additional partial products resulting from merging the nonoverlapping bits is given by Q .

$$Q = \left\lceil \frac{\left(\lfloor \frac{n}{3} \rfloor + 1\right) \cdot M}{n} \right\rceil = \left\lceil \frac{\left(\lfloor \frac{n}{3} \rfloor + 1\right)}{k} \right\rceil . \quad (15)$$

TABLE III
SYNTHESIS RESULTS WHEN n IS NOT DIVISIBLE BY THREE

n	k					
	n		$n/2$		$n/4$	
	Area (μm^2)	Delay (ns)	Area (μm^2)	Delay (ns)	Area (μm^2)	Delay (ns)
16	16026	6.99	16602	5.63	18208	5.18
20	23477	8.34	24136	6.20	26192	6.13
28	44703	10.50	45624	7.77	48555	6.72
32	56691	11.59	57909	8.38	60730	7.69
40	87730	13.62	89134	10.05	92656	8.23
44	103391	14.81	104851	10.45	109232	8.56
52	144422	16.76	146018	12.29	151148	9.13
56	168768	21.53	172051	13.43	178621	10.63
64	221082	20.15	224821	17.30	231550	10.99

TABLE IV
SYNTHESIS RESULTS WHEN n IS DIVISIBLE BY THREE

n	k			
	$n/3$		$n/6$	
	Area (μm^2)	Delay (ns)	Area (μm^2)	Delay (ns)
12	11459	4.90	12883	4.87
24	37302	6.45	40209	6.33
48	130800	9.78	137410	8.62
60	207194	12.57	212277	9.24

The proposed modulo 2^n-1 multiplier designs for various feasible combinations of and were specified in VHDL, synthesized using Synopsys Design Compiler (V2004.06-SP2) and mapped to TSMC 0.18 1.8 V CMOS standard-cell library. The designs were synthesized under nominal synthesis design environment, i.,e., 25 C and 1.8 V initially before the timing constraint from the RNS is imposed. The area and delay synthesis results are shown in Table III for that is not divisible by three and in Table IV for *n* that is divisible by three.

As the dynamic power dissipation of a combinational circuit is dependent on the input pattern, a Monte Carlo simulation method [49], [50] using a finite number of randomly generated test patterns is adopted to estimate the average power dissipation with 99.9% confidence that the error is bounded below 3% for a data rate of 20 Msamples/s. The average dynamic power and the leakage power are listed in Tables V and VI for *n* not divisible and divisible by three, respectively.

VI. CONCLUSION

A family of low-area and low-power modulo 2^n-1 multipliers with variable delay to achieve delay balance amongst individual modulo channels in a high-DR RNS multiplier was proposed. The delay of the proposed multiplier is controlled by the word-length of the small parallel RCAs that are used to compute the requisite hard multiple of the radix-8 Booth encoded multiplication in a partially-redundant form. The trade-offs between the RCA word-length and the VLSI performance metrics, .i.e, area, delay and power dissipation of the modulo 2^n-1 multiplier were analyzed by means of CMOS implementations. For maximal area and power savings, *n* when *n* is not divisible by three and *n/3* when *n* is divisible by three, were recommended for the RCA word-length when the RNS multiplier delay exceeded the noncritical modulo 2^n-1 multiplier delay substantially. Conversely, when the RNS multiplier and the modulo 2^n-1 multiplier delays were nearly balanced, RCA word-lengths of *n/4* and *n/6* were recommended when is not divisible and divisible by three, respectively. From synthesis results constrained by the critical channel delay of the RNS, it was shown that the proposed multiplier simultaneously reduces the

area as well as the power dissipation of the radix-4 Booth encoded multiplier for $n \geq 28$, which is the useful dynamic range of RNS multiplication to meet the minimum key-size requirements of ECC and RSA algorithms.

REFERENCES

[1]. R. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public key cryptosystems,” *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.
[2]. [2] V. Miller, “Use of elliptic curves in cryptography,” in *Proc. Advances in Cryptology-CRYPTO’85, Lecture Notes in Computer Science*, 1986, vol. 218, pp. 417–426.
[3]. N. Koblitz, “Elliptic curve cryptosystems,” *Mathemat. of Comput.*, vol. 48, no. 177, pp. 203–209, Jan. 1987.
[4]. National Institute of Standards and Technology [Online]. Available: http://csrc.nist.gov/publications/PubsSPs.html
[5]. A. K. Lenstra and E. R. Verheul, “Selecting cryptographic key sizes,” *J. Cryptol.*, vol. 14, no. 4, pp 255–293, Aug. 2001.
[6]. C. McIvor, M. McLoone, and J. V. McCanny, “Modified Montgomery modular multiplication and RSA exponentiation techniques,” *IEE Proc. Comput. and Dig. Techniq.*, vol. 151, no. 6, pp. 402–408, Nov. 2004.
[7]. C. McIvor,M.McLoone, and J. V. McCanny, “Hardware elliptic curve cryptographic processors over □_□__,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 53, no. 9, pp. 1946–1957, Sep. 2006.
[8]. D. M. Schinianakis, A. P. Fournaris, H. E. Michail, A. P. Kakarountas, and T. Stouraitis, “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.
[9]. J. C. Bajard and L. Imbert, “A full RNS implementation of RSA,” *IEEE Trans. Comput. – Brief Contributions*, vol. 53, no. 6, pp. 769–774, Jun. 2004.
[10]. H. Nozaki, M. Motoyama, A. Shimbo, and S. Kawamura, “Implementation of RSA algorithm based on RNS Montgomery multiplication,” in *Proc. Workshop on Cryptographic Hardware and Embedded Systems*, Paris, France, May 2001, pp. 364–376.
[11]. T. Stouraitis and V. Paliouras, “Considering the alternatives in lowpower design,” *IEEE Circuits Devices Mag.*, vol. 17, no. 4, pp. 22–29, Jul. 2001.
[12]. S. Pontarelli, G. C. Cardarilli, M. Re, and A. Salsano, “Totally fault tolerant RNS based FIR filters,” in *Proc. 14th IEEE Int. On-Line Testing Symp.*, Rhodes, Greece, Jul. 2008, pp. 192–194.
[13]. . Steiner *et al.*, “A fault-tolerant modulus replication complex FIR filter,” in *Proc. 16th IEEE Int. Conf. Application-Specific Systems, Architecture and Processors*, Samos, Greece, Jul. 2005, pp. 387–392.

[14]. N. S. Szabo and R. I. Tanaka, *Residue Arithmetic and its Applications to Computer Technology*. New York: McGraw-Hill, 1967.

[15]. [15] M. A. Soderstrand, W. K. Jenkins, G. A. Jullien, and F. J. Taylor, *Residue Number System Arithmetic: Modern Applications in Digital signal Processing*. New York: IEEE Press, 1986.

[16]. P. V. A. Mohan, *Residue Number Systems: Algorithms and Architectures*. Norwell, MA: Kluwer, 2002.

[17]. I. Kouretas and V. Paliouras, “A low-complexity high-radix RNS multiplier,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 11, pp. 2449–2462, Nov. 2009.

[18]. G. Dimitrakopoulos and V. Paliouras, “A novel architecture and a systematic graph-based optimization methodology for modulo multiplication,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 2, pp. 354–370, Feb. 2004.

[19]. A. A. Hiasat, “New efficient structure for a modular multiplier for RNS,” *IEEE Trans. Comput.*, vol. 49, no. 2, pp. 170–174, Feb. 2000.

[20]. B. Cao, C. H. Chang, and T. Srikanthan, “An efficient reverse converter for the 4-moduli based on the New Chinese Remainder Theorem,” *IEEE Trans. Circuits Syst. I, Fundam.*

[21]. *Theory Appl.*, vol. 50, no. 10, pp. 1296–1303, Oct. 2003.

[22]. B. Cao, T. Srikanthan, and C. H. Chang, “Efficient reverse converters for four-moduli sets *IEE Proc. Comput., Dig. Techniq.*, vol. 152, no. 5, pp. 687–696, Sep. 2005.

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

[24]. P. V. A. Mohan and A. B. Premkumar, “RNS-to-binary converters for two four-moduli sets *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, no. 6, pp. 1245–1254, Jun. 2007.

[25]. P. V. A. Mohan, “RNS-to-binary converter for a new three moduli set *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 54, no. 9, pp. 775–779, Sep. 2007.

[26]. T. Tomczak, “Fast sign detection for RNS *Trans. Circuits Syst. I, Reg. Papers*, vol. 55, no. 6, pp. 1502–1511, Jul. 2008.

[27]. T. Stouraitis, S. W. Kim, and A. Skavantzos, “Full adder-based arithmetic units for finite integer rings,” *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 40, no. 11, pp. 740–745, Nov. 1993.

[28]. D. J. Soudris, V. Paliouras, T. Stouraitis, and C. E. Goutis, “A VLSI design methodology for RNS full adder-based inner product architectures,” *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 44, no. 4, pp. 315–318, Apr. 1997.

[29]. B. Cao, Y. S. Low, C. H. Chang, and T. Srikanthan, “Performance analysis of different special moduli sets for RNS-based inner product step processor,” in *Proc. 2010 Int. Conf. Green Circuits and Systems*, Shanghai, China, Jun. 2010, pp. 236–241.

[30]. P. V. A. Mohan, “Reverse converters for a new moduli set *Circuits, Syst. Signal Process.*, vol. 26, no. 2, pp. 215–227, Apr. 2007.

[31]. A. S. Molahosseini, K. Navi, C. Dadkhah, O. Kavehei, and S. Timarchi, “Efficient reverse converter designs for the new 4-moduli sets based on New CRTs,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 57, no. 4, pp. 823–835, Apr. 2010.

[32]. S. Ravi, A. Raghunathan, P. Kocher, and S. Hattangady, “Security in embedded systems: Design challenges,” *ACM Trans. Embedded Comput. Syst.*, vol. 3, no. 3, pp. 461–491, Aug. 2004.