American Journal of Engineering Research (AJER)
e-ISSN : 2320-0847 p-ISSN : 2320-0936
Volume-02, Issue-07, pp-176-184
www.ajer.org

Research Paper Open Access

Optimized Modulo Multiplier Based On R.N.S

Manjula.S.Doddamane, G.Parameshappa
Dept of Electronics and Communication JSSATE Bengaluru, India

Abstract: - To implement long and repetitive multiplications of cryptographic and signal processing algorithm we often adopt residue number system. In this paper a new low power and low modulo multiplier foe well established $\{2^n-1,2^n,2^n+1\}$ based is proposed .Radix-8 Booth encoding technique is used in the proposed modulo 2^n-1 and modulo 2^n+1 multipliers. In the proposed modulo 2^n-1 multiplier, the number of partial products is lowered to [n/3]+1. For modulo 2^n+1 multiplication ,the aggregate bias due to the hard multiple and the modulo reduced partial product generation is composed of multiplier dependent dynamic bias and multiplier-independent static bias .In the proposed modulo 2^n+1 multiplier , the number of partial products is lowered to n/3+6 .For different modulo 2^n-1 and modulo 2^n+1 multiplier our proposed modulo multiplier consumes less area and has minimum power dissipation over radix-4 Booth encoded and non-encoded modulo multiplier.

Keywords: - *Booth algorithm, multiplication, residue number system (RNS).*

I. INTRODUCTION

Residue Number System (RNS) is adopted in high speed digital signal processors and cryptographic application. RNS is defined by a set of co-prime moduli and is represented as non-positional number representation .The residue of modulo is represented as P=XoY is (p1,p2,--pn)=(|x1oy1|L1,|x2oy2|L2----------|xnoyn|Ln) ,where pi is computed from xi and yi in the modulo channel corresponding to Li for i=1,2…,N. In any operation carry is generated and carry chain slows down the VLWL(Very Long Word Length).The addition and multiplication is effectively broken in RNS ,into smaller digits so that carry chain can be avoided. The main barrier in implementing VLWL arithmetic is with limited space and constrained battery specification, so that it can used for smart cards and Radio Frequency Identification (RFID) tags.

RNS are especially useful in algorithms where multiplication and addition dominate .It distributes a long integer multiplication into several shorter and independent modulo .So arithmetic circuits required for digits are smaller and faster. As RNS is most suited for applications involving repetitive computations like repeated modulo multiplications in cryptographic algorithm and multiply-add operations in signal processing algorithm, the research emphasis has shifted markedly in recent years to the area-power efficient implementation of concurrent modulo arithmetic operations in RNS.

In this paper, we propose a new radix-8 Booth encoded modulo $\mathbf{2^n-1}$ multiplier and radix-8 Booth encoded modulo $\mathbf{2^n+1}$ multiplier. Our objective is to minimize the area and power consumption of VLWL multiplication in RNS based on moduli $\mathbf{2^n-1}$ and $\mathbf{2^n+1}$. Both the proposed modulo hard multiple generators (HMGs) employ only prefix levels and are by far the most area-delay-power efficient customized adders for this application. In the proposed modulo multiplier, one partial product per radix-8 Booth encoded multiplier digit is generated. As the hard multiple is generated in an unbiased form, no further correction term is incurred. Thus, the number of modulo-reduced partial products in modulo multiplier is lowered , which is the best achievable partial product count using radix-8 Booth encoding scheme. So the hardware's required for implementing the proposed multiplier modulo $\mathbf{2^n-1}$ and modulo $\mathbf{2^n+1}$ is less due to which we can minimize size of the multiplier block.

II. MODULO-REDUCED PARTIAL PRODUCTS FOR RADIX-8 BOOTH ENCODED MULTIPLICATION

This section presents the preliminaries of radix-8 Booth encoded modulo 2^n-1 multiplication using binary representation with dual zeros for modulo 2^n+1 arithmetic and diminished-1 representation for modulo arithmetic. Mathematically, the modulo 2^n-1 and modulo 2n+1 multiplications can be expressed.

For modulo 2^n-1

$$|P|_m=X.Y= \sum_{i=0}^{n-1} X.yi\ 2^i \qquad \text{if } m= 2^n-1 \tag{1}$$

For modulo 2^n+1

$$|P|_m= X.Y+X+Y$$
$$= \sum_{i=0}^{n-1} X.yi.2^i+X+Y \qquad \text{if } m=2^n+1$$

where X,Y and P represent multiplicand , multiplier and product respectively By adopting Radix-8 Booth encoding ,the product can be computed from only n/3+1 modulo reduced partial products
and (1) can be expressed as

for modulo 2^n-1

$$|P|_m=X.Y= \sum_{i=0}^{n/3} X.di\ 2^{3i} \qquad \text{if } m= 2^n-1 \tag{2}$$

for modulo 2^n+1

$$|P|_m = \sum_{i=0}^{n/3} X.di.2^{3i}+X+Y \qquad \text{if } m=2^n+1$$

can be further simplified to

for modulo 2^n-1

$$|P|_m= \sum_{i=0}^{n/3}.PP_i \qquad \text{if } m= 2^n-1 \tag{3}$$

for modulo 2^n+1

$$|P|_m = \sum_{i=0}^{n/3}(PPi+ki)+X+Y \qquad \text{if } m=2^n+1$$

III. PROPOSED MODULO AND MODULO HARD MULTIPLE GENERATORS (HMGs)

A . Modulus 2^n-1

As is congruent to zero modulo , 2^n-1 zero can be represented by an -bit binary string of all zeros or all ones in modulo 2^n-1 arithmetic. Thus, a modulo addition of two operands, A and B is equivalent to an -bit addition of ,A,B and cout i.e.,

$$|G+H|_{2n-1}= \begin{cases} |G+H|_{2n} & \text{if } G+H<2^n \\ |G+H+1|_{2n} & \text{if } G+H>2^n \\ |G+H+C_{out}|_{2n} & \end{cases} \tag{4}$$

where C_{out} is the carry output resulting from the addition of G and H . As is added to the sum of G and H at the position, modulo 2^n-1 addition is commonly referred to as end-around-carry (EAC) addition.This modulo 2^n-1 adder can be implemented by a parallel-prefix structure with three operator stages, namely pre-processing, prefix-computation and post-processing .The pre-processing stage computes the generate (gi), propagate(pi) and half-sum (hi)bits for i=0 to n-1.

$$g_i=a_i\ .b_i$$
$$p_i=a_i+b_i$$
$$h_i=a_i \oplus b_i \tag{5}$$

The prefix-computation stage uses the prefix operators (•) on (gi, pi) to calculate the carry bits c_i for the EAC addition. For i=0 to n-1.

$$c_i=(g_i,p_i)\bullet\text{-------}(g_0,p_0)\bullet(g_{n-1},p_{n-1})\bullet\text{-----}\bullet(g_{i+1},p_{i+1}) \quad (6)$$

where $(g_i,p_i)\bullet(g_j,p_j)=(g_i+p_i.g_j,pi,p_j)$

The modified n/2 generate and propagate bits pairs (g_i,p_i) are required in the prefix computation stage of modulo 2^n-1 is given as

$$\begin{aligned} g_i&=x_{n-1}(x_0+x_{n-2}) && \text{for i=0 to n-1}\\ p_i&=x_{n-1}+x_0.x_{n-2} && \text{for i=0 to n-1}\\ h_i&=x_i \oplus x_{i-1} && \text{for i=0 to n-1} \end{aligned} \quad (7)$$

The post-processing stage computes the sum bit for i=0 to n-1 i.e.,

$$s_i=h_i \oplus c_{i-1} \quad (8)$$

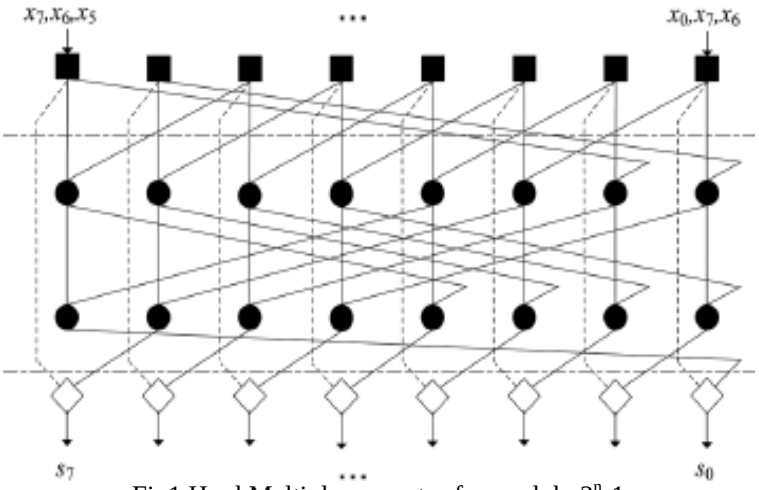


Fig1.Hard Multiple generator for modulo 2^n-1

B. Modulus 2^n+1

A modulo **2^n+1** addition of two diminished-1 represented operands, and , is equivalent to an -bit addition of G and H with

$$\begin{aligned} |S+1|_{2n+1}&=|G+H+1|_{2n+1}\\ &\quad |G+H+\overline{Cout}|_{2n} \end{aligned} \quad (8)$$

where $C_{ou}t$ is the carry output from the addition of G and H . As $\overline{Cout}$ is added to the sum of and at the LSB position, modulo addition is commonly referred to as complementary-end around- carry (CEAC) addition .The pre-processing and post-processing stages of the parallel prefix modulo **2^n+1** adder are identical to those of the modulo**2^n-1** adder but the carry equation is implemented differently in the prefix-computation stage due to CEAC addition

$$c_i=(g_i,p_i)\bullet\text{--------}\bullet(g_0,p_0)\bullet\overline{(g_{n-1},p_{n-1})\bullet\text{------}\bullet(g_{i+1},p_{i+1})} \quad (9)$$

where $(g_i,p_i)\bullet(g_j,p_j)=(g_i+p_i.g_j,pi,p_j)$

$(\overline{p_i},\overline{g_i})\bullet(\overline{p_j},\overline{g_j})=(\overline{p_i}+\overline{g_i}.\overline{p_i},\overline{g_i}.\overline{g_i})$

$(g_i,p_i)\ \circledcirc\ (g_j,p_j)=(g_i+p_i.g_j,pi,p_j)(p_i+g_i\ p_j,g_i.g_j)$

The modified prefix operator for modulo 2^n+1 $(g_i,p_i)$is given as

$$g_i=\overline{x}_{n-1}.(x_0+\overline{x}_{n-2}) \quad \text{for i=0 to n-1}$$

$$p_i=\overline{x}_{n-1}+x_0.\overline{x}_{n-2} \quad \text{for i=0 to n-1}$$

$$h_i=x_i \oplus x_{i-1} \quad \text{for i=0 to n-1} \qquad (10)$$

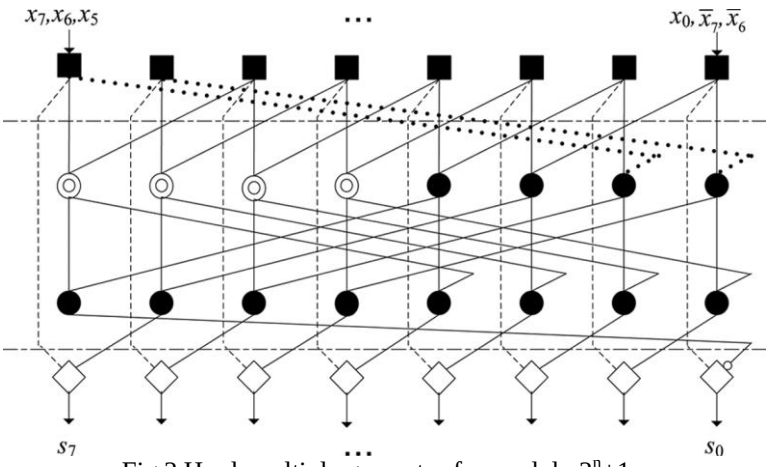
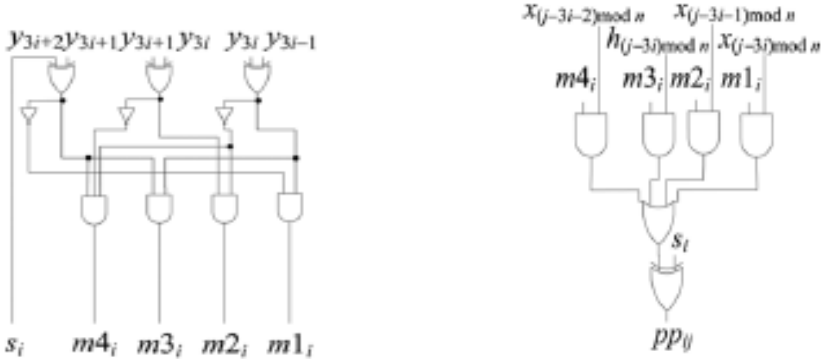


Fig 2.Hard multiple generator for modulo 2^n+1

IV. PROPOSED MODULO 2^n-1 AND MODULO 2^n+1 MULTIPLIERS

The Booth Encoder (BE) produces a signed digit represented by a sign bit and one-hot encoded magnitude bits denoted by m1i,m2i,m3i,m4i and from four consecutive multiplier bit. The generation PPi of in BE, BS and HMG blocks for the moduli $\mathbf{2^n+1}$ and $\mathbf{2^n-1}$. A bank of identical BS blocks is required to generate a single PPi. Specifically for modulo $\mathbf{2^n+1}$ multiplier, the outputs of the BS blocks at the least-significant 3i bit positions are inverted to implement the CCLS operation

The modulo-reduced partial product accumulation (MPPA) differs significantly for modulo $\mathbf{2^n-1}$ and modulo $\mathbf{2^n+1}$ multipliers. For modulo $\mathbf{2^n-1}$ multiplier, the number of modulo-reduced partial products to be accumulated is n/3+1. Since modulo addition can be efficiently implemented by EAC addition, the CSA for its MPPA is shown in Fig. 6 for .n=8



(a) (b)

Fig 3 (a)Booth encoder and (b)Booth selector

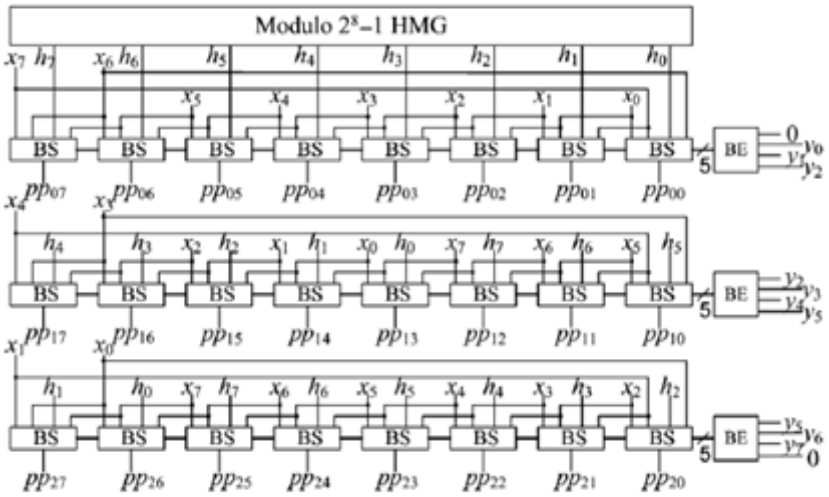


Fig 4.Modulo 2^n-1PP_i generation

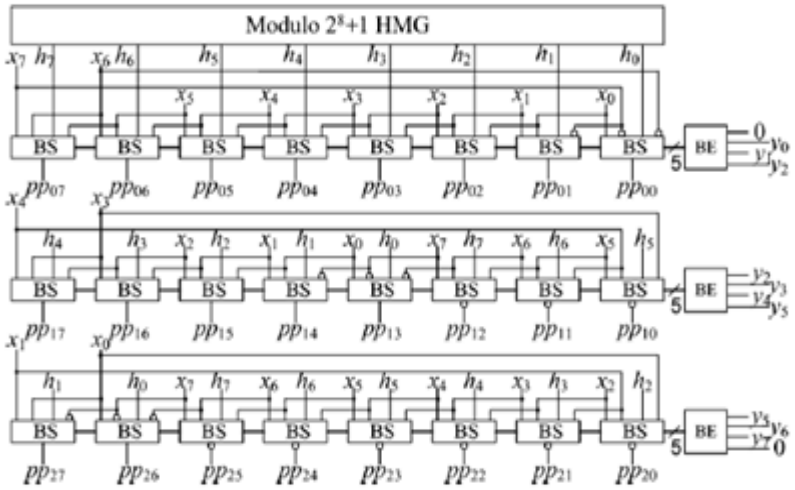


Fig 5.Modulo 2^n+1PP_i generation

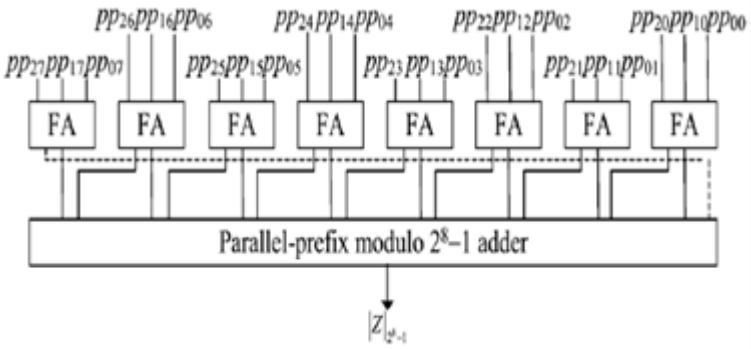


Fig 6.Modulo 2^n-1 reduced partial product accumulation

For modulo $\mathbf{2^n-1}$ multiplier, the number of modulo-reduced partial products and biasing constant to be accumulated is n/3+6. Since modulo addition can be efficiently implemented by CEAC addition, the CSA for its MPPA is shown in Fig. 7 for .n=8

we need to compute the bias constant that is

$$K1=\sum_{i=0}^{n/3} \overline{m2_i \vee m3_i} \cdot 2^{3i} + \overline{m3_i \vee m4_i} \cdot 2^{3i+1}$$
$$K2=\sum_{i=0}^{n/3} ((m2_i \vee m4_i) \wedge s_i) \cdot 2^{3i+1} + \sum_{i=0}^{\frac{n}{3}-1} ((m3_i \vee m4_i) \wedge s_i) \cdot 2^{3i+2}$$
$$K3=\sum_{i=1}^{n/3} 2^{3i} + \sum_{i=0}^{n/3} \overline{s_i} \; 2^{3i+1} + \sum_{i=0}^{n/3} s_i \; 2^{3i+1} \quad (10)$$

The final product is given by

$$|P|_{2n+1}=\sum_{i=0}^{n/3} PPi + X + Y + K1 + K2 + K3 \quad (11)$$

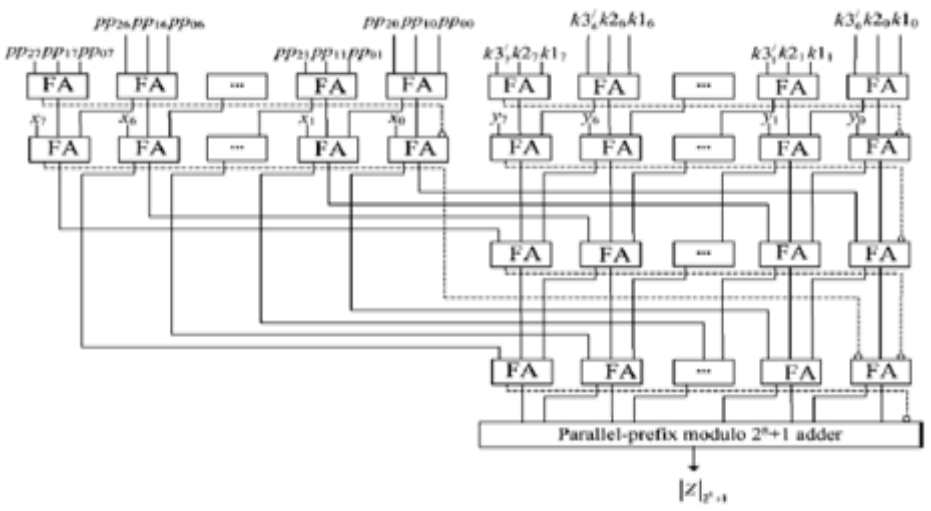


Fig 7. Modulo 2^n+1 reduced partial product accumulation

V.SIMULATION RESULT

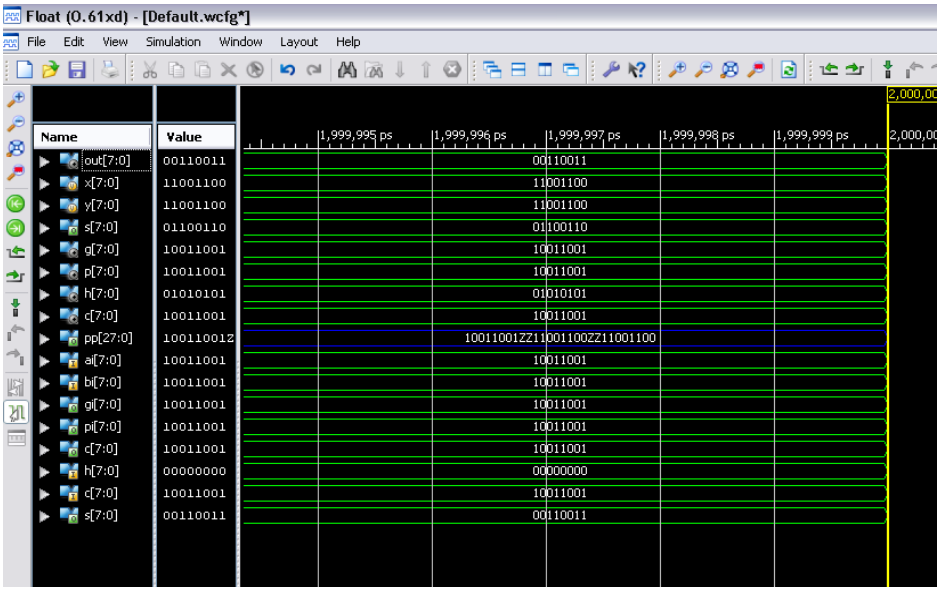


Fig 8 For Modulo2^n**-1** multiplier

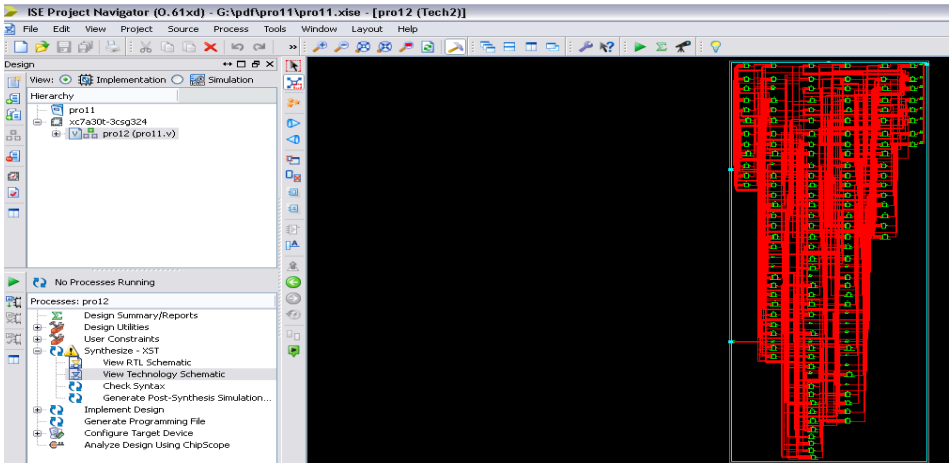


Fig 9.Technology schematic for modulo 2^n-1

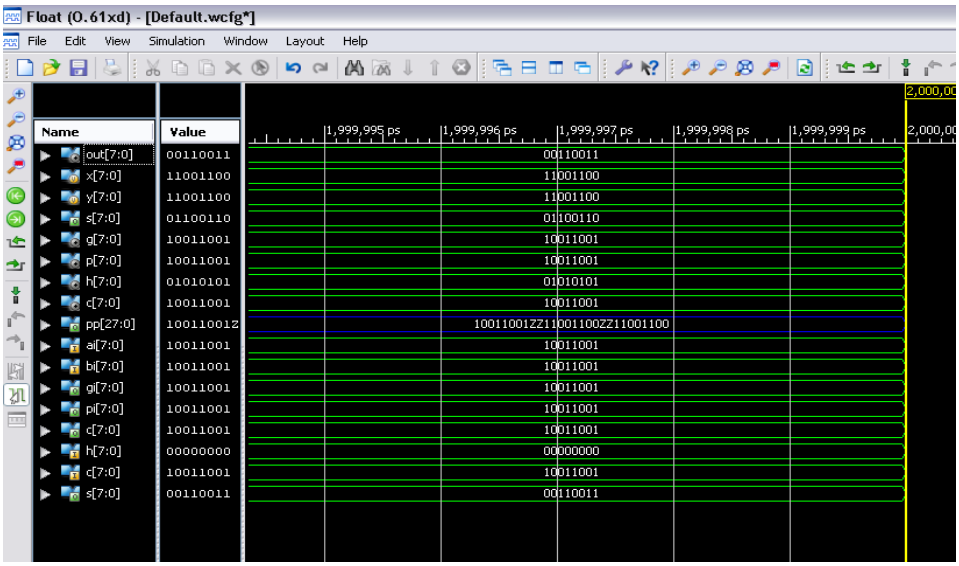


Fig 10 .For Modulo2^n+1 multiplier

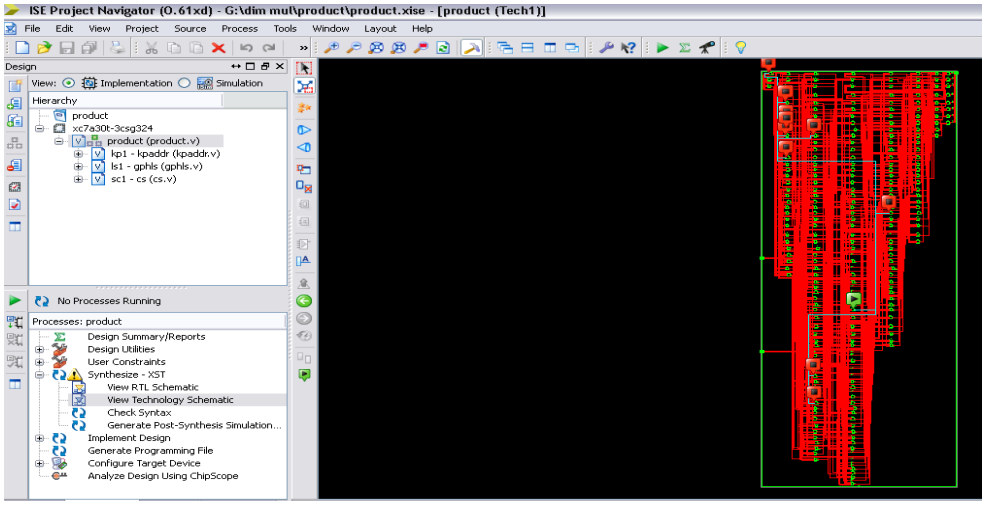


Fig 11.Technology schematic for modulo 2^n+1

VI.PERFORMANCE COMPARISION

The proposed multiplier is synthesized in Xilinx ISE simulator cadence tool to find optimized area and power dissipation for modulo 2^n-1 and modulo 2^n+1 .The modulo multipliers is compared with different modulo multiplier that can be seen below

TABLE 1.Area of modulo 2^n-1

n	PROPOSED MULTIPLIER	[2]	[3]
8	1189	1147	988
16	4270	4249	3921
32	14961	15966	15301

TABLE 2 .Power dissipation of modulo 2^n-1

n	PROPOSED MULTIPLIER (n W)	[2] (n W)	[3](n W)
8	47910.969	61227.246	48260.121
16	29334.151	337299.160	316111.870
32	1457361.477	1799592.653	1746144.796

TABLE 3. Area of modulo 2^n+1

n	PROPOSED MULTIPLIER	[5]	[6]
8	2358	1926	1710
16	6750	6813	6233
32	20745	24810	23099

TABLE 4. Power dissipation of modulo 2^n+1

n	PROPOSED MULTIPLIER (n W)	[5] (n W)	[6](n W)
8	103190.856	104933.202	78377.144
16	478557.460	525754.151	446820.961
32	2023247.140	2307567.659	2173617.139

VII. CONCLUSION

The Optimized modulo 2^n-1 and modulo2^n+1 multipliers employing radix-8 Booth encoding scheme were proposed. The hardware cost of hard multiple generation was minimized by customizing the parallel-prefix modulo 2^n-1 and modulo 2^n+1 adders for the efficient computation of hard multiples . The area-delay-power metrics of the proposed modulo multipliers were evaluated and compared against radix-4 Booth encoded and non-encoded modulo multipliers. the simulation of multiplier is done in Xilinx and synthesis in cadence In RNS multiplier based on moduli 2^n-1 and , 2^n+1 the proposed multipliers lower both the implementation area and the total power dissipation.

REFERENCES

[1]. Z.Wang.G.A.Jullien and W.C.Miller ."An algorithm for Multiplication modulo 2^n-1 ."in Proc 39th IEEE Midwest Symp.circuit Syst ,Ames ,IA Aug 1996pp 1301-1304

[2]. R. Zimmermann ,"Efficient VLSI implementation of modulo addition and multiplication" ,in proc 14th IEEE Symp Computer Arithmetic ,Adelaide ,Australia apr 1999,pp 158-167.

[3]. C. Efstathiou, H. T. Vergos, and D. Nikolos, "Modified Booth modulo2^n-1 multipliers," IEEE Trans. Comput., vol. 53, no. 3, pp. 370–374, Mar. 2004.

[4]. Z.Wang, G. A. Jullien, andW. C.Miller, "An efficient tree architecture for modulo 2^n+1 multiplication," J. VLSI Signal Process., vol. 14, no. 3, pp. 241–248, Dec. 1996.

[5]. C. Efstathiou, H. T. Vergos, G. Dimitrakopoulos, and D. Nikolos, "Efficient diminshed-1 modulo 2^n+1 multipliers," IEEE Trans. Comput., vol. 54, no. 4, pp. 491–496, Apr. 2005.

[6]. J.W.Chen R.H.Yao " Efficient modulo 2^n+1 multipliers for diminished -1 representation."

[7]. Y. Ma, "A simplified architecture for modulo 2^n+1 multiplication," IEEE Trans. Comput., vol. 47, no. 3, pp. 333–337, Mar. 1998.

[8]. L. Sousa and R. Chaves, "A universal architecture for designing efficient modulo 2^n+1 multipliers," IEEE Trans. Circuits Syst. I, Reg.Papers, vol. 52, no. 6, pp. 1166–1178, Jun. 2005.

[9]. J. W. Chen and R. H. Yao, "Efficient modulo 2^n+1 multipliers fordiminished-1 representation," IET Circuits, Devices Syst., vol. 4, no. 4,pp. 291–300, Jul. 2010.

[10]. L. Leibowitz, "A simplified binary arithmetic for the Fermat number transform," IEEE Trans. Acoustics, Speech Signal Process., vol. 24,no. 5, pp. 356–359, Oct. 1976.

[11]. H. T. Vergos and C. Efstathiou, "Design of efficient modulo 2^n+1 multipliers," IET Comput. Digital Tech., vol. 1, no. 1, pp. 49–57, Jan.2007.

[12]. G.W. Bewick, "Fast Multiplication: Algorithms 2^n-1 and2^n+1 Implementation,"Ph.D. dissertation, Stanford Univ, Stanford, CA, 1994.

[13]. B. S. Cherkauer and E. G. Friedman, "A hybrid radix-4/radix-8 low power signed multiplier architecture," IEEE Trans. Circuits Syst. II, Analog.Digit. Signal Process., vol. 44, no. 8, pp. 656–659, Aug. 1997.

[14]. M. J. Flynn and S. F. Oberman, Advanced Computer Arithmetic Design. New York: Wiley, 2001.

[15]. R. Muralidharan and C. H. Chang, "Radix-8 Booth encoded modulo 2^n-1 multipliers with adaptive delay for high dynamic range residue number system," IEEE Trans. Circuits Syst.I, Reg. Papers, vol. 58, no. 5, pp. 982–993, Jun.2011