



# Mathematical Modelling and Implementation of a Low-power Higher-Radix Hybrid Multiplier

**Jithin Kumar M.V.**

Dept. of Electronics & Communication Engg.  
Heera College of Engg. & Technology  
Thiruvananthapuram Dist., Kerala  
[mv.jithin@gmail.com](mailto:mv.jithin@gmail.com)

**Jayanthi K.B.**

Dept. of Electronics & Communication Engg  
K. S. Rangasamy College of Engg. & Technology  
Tiruchengode, Namakkal Dist., Tamil Nadu.  
[jayanthikb@gmail.com](mailto:jayanthikb@gmail.com)

**Abstract**—This paper presents a low-power multiplication algorithm based on Radix-16 Modified Booth Encoder (MBE). Hard multiples are the primary factors for power consumption in higher radix MBE. The paper introduces the design of a Hybrid Multiplication Technique (HMT) to suppress the Hard Multiples that exist in a Radix-16 MBE and its mathematical model. The proposed HMT uses Radix-8 and Radix-4 encoding schemes along with Radix-16 to decrease the number of Hard Multiples. Synthesis results based on Synopsys SDK 90nm, 1.32V standard-cell library show that the proposed HMT reduces the power utilization up to 28% and 22% while comparing with conventional Radix-16, Radix-8 MBE, respectively. HMT's nets capacitance obtained is 37% and 66% less than conventional Radix-16 and Radix-8 MBA. HMT equipped Radix-16 also gives better performance than existing techniques with respect to area and power.

**Keywords**—Higher-radix modified booth multiplier, Hard Multiples, Synthesis

## I. INTRODUCTION

A lot of preceding digital multipliers aimed at transition or switch activity decrement to reduce power dissipation. The power reduction method used in a Modified Booth Multiplier (MBM) is the Radix-4 recoding scheme [1]. This Radix-4 MBE is being widely used in parallel multipliers to diminish the number of partial products by a factor of two. Partial Products are generated by shifting and 2's complement operation [2-4]. It does not generate hard multiples. Better savings in area and dynamic power dissipation are possible for huge word-length multipliers by raising the bit coding which higher than Radix-4 as in [5]. A technique that reduces number of full adders in the compression tree of signed/unsigned MBA is described in [6]. Also it is based on the need to extend the sign bit of the partial products. Based on the extension of 2<sup>nd</sup> order MBA novel concurrent carry

save multiplier accumulator (MAC) architecture is revealed in [6,7] with different computational performance. A combination of wide-bit range Radix-4/Radix-8 architecture is proposed in [8] to compromise between the high speed and the low power dissipation Radix-4 and Radix-8 multiplier architecture, respectively.

A new encoding scheme with multiple-level conditional-sum adder (MLCSMA) hybrid structure is proposed in [9] to improve the performance up to 25% in booth encoded parallel multiplier design. A hybrid spare-tree structure is used in [10] to implement two's complement circuit which further reduces the area to 15.8% and improve the speed up to 11.7% over the standard design. Quite a lot of commercial processors have the radix-8 multiplier design to boost their speed, thereby reducing the number of partial products. The number of partial product in a digit representation is reduced by  $\frac{2}{3}$  in Radix-8 encoding. But its performance bottleneck is the generations of the terms  $\pm 3X$  and  $\pm 4X$  also referred to as 'hard multiples'. This term is usually computed by adding and shifting operations,  $(+4X = +3X + X)$  in a high-speed adder. In a  $3X + X$  addition, full adders share the same input signal. This property consumes much power in hard multiples expression related to  $\pm 3X$  process [11].

The proposed work optimizes the internal architecture of MBM, which controls dynamic multiplier resources to match peripheral data characteristics [12]. The prime objective is power reduction by avoiding the hard multiples in a Radix-16 MBE by the use of Radix-4 and Radix-8 MBE. By using Hybrid architectures, it is possible to achieve both power reduction and delay reduction, which is the strength of high-level optimization. The paper is organized as follows. Section II discusses the mathematical modeling

and design of proposed algorithm. Section III gives the synthesis and performance analysis of Hybrid Multiplier.

## II. MATHEMATICAL MODELLING OF HMT

The regular linear array multiplication can be expressed by considering a binary integer Multiplicand (M) and Multiplier (R),

$$\text{If } M = \sum_{i=0}^{m-1} M_i 2^i \quad \text{and} \quad R = \sum_{j=0}^{n-1} R_j 2^j$$

$$\text{The product, } P = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (M_i R_j) 2^{i+j}$$

$$P = \sum_{k=0}^{m+n-1} P_k 2^k \quad (1)$$

Where 'mn' partial products are obtained. Which are produced by a set of AND gates as shown in Figure 1. Scheme [13] is an example for linear array multiplier.

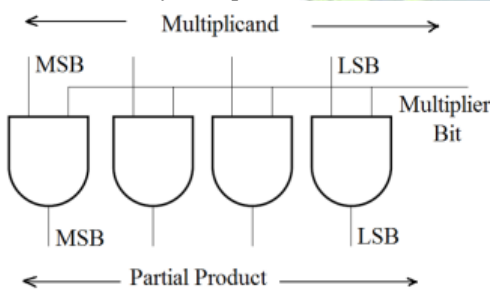


Fig. 1: 4x4 bit linear array multiplier partial product generator

This 'mn' partial products consumes large power. So a booth algorithm and its modification is proposed in [14]. A 2's complement binary number with n-bit Multiplicand (M) and Multiplier (R), if 'n' is even the multiplier bit group can be represented by

$$R = \sum_{i=0}^{(n-2)/2} (r_{2i+1} + r_{2i} - 2r_{2i+1}) 2^{2i} \quad (2)$$

$$M \times R = \sum_{i=0}^{(n-2)/2} S_i \quad (3)$$

$$\text{Where, } S_i = (r_{2i+1} + r_{2i} - 2r_{2i+1}) A 2^{2i} \quad (4)$$

$S_i$  is called choosing of 3 bits. As shown in equation number 4, the multiplier is separated by overlapping groups of 3 bits with an appended zero (represented by Z) at the LSB. The group is represented in another form by  $S_k$ ,

if 'n' is even, and  $k = N-1$ ,

$$S_k = (b_{2k+1} b_{2k} b_{2k-1}) \quad (5)$$

Where,  $(N-2) \leq k \leq 1$

$$S_k = (b_{2k+1} b_{2k} Z), \text{ for } k=0 \quad (6)$$

if 'n' is odd,

$$S_k = (b_{2k} b_{2k} b_{2k-1}), \text{ for } k = N-1 \quad (7)$$

Where, the sign bit is repeated by  $b_{2k}$ .

The combinations of  $S_k$  is generated by a radix - 4 booth encoder as shown in Figure 2. For MBA, the multiples  $S_k \{0, \pm 1, \pm 2\}$  are used to generate the partial products. The final product is given as

$$P = M \times R = M \cdot S_{N-1} 2^{2(N-1)} + M \cdot S_{N-2} 2^{2(N-2)} + \dots + M \cdot S_1 2^2 + M \cdot S_0 2^0 \quad (6)$$

From the above equation it is clear that, each partial product is shifted by 2 bits before compression to produce correct result.

The partial product generation can be further reduced by higher radix encoding technique like Radix-8, Radix-16 etc., In Radix-8 encoding technique; the multiplier is separated by 4-bit group with a zero at LSB.

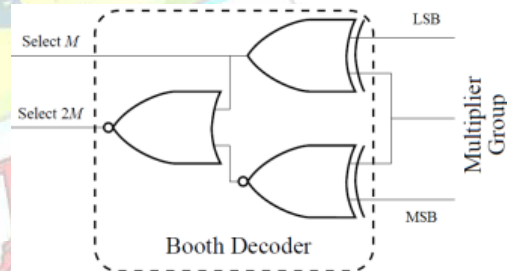


Fig. 2: Radix-4 Modified booth encoder  
if 'n' is even and  $k = N-1$ , the Radix-8 multiplier group  $S_k$  can be represented by,

$$S_k = (b_{3k+2} b_{3k+1} b_{3k} b_{3k-1}) \quad (7)$$

Where,  $(N-2) \leq k \leq 1$

$$S_k = (b_{3k+2} b_{3k+1} b_{3k} Z), \text{ for } k=0 \quad (8)$$

if 'n' is odd,

$$S_k = (b_{3k+1} b_{3k+1} b_{3k} b_{3k-1}), k = N-1 \quad (9)$$

Where, the sign bit is repeated by  $b_{3k+1}$ .

The combinations of  $b_{3k+1}$  is produced by Radix - 8 encoder as shown in Figure 3. The multiples  $S_k \{0, \pm 1, \pm 2, \pm 3, \pm 4\}$  are used to generate the partial products and the final product is

$$P = M \times R = M.S_{N-1}2^{3(N-1)} + M.S_{N-2}2^{3(N-2)} + \dots + M.S_12^3 + M.S_02^0 \quad (10)$$

Hence each partial product is shifted by 3 bits before compression. In Radix-16 encoding technique, the multiplier is separated by 5-bit group and an appended zero is added at LSB. If 'n' is even and  $k=N-1$ , the Radix-16 multiplier group  $S_k$  can be represented by

$$S_k = (b_{4k+3}b_{4k+2}b_{4k+1}b_{4k}b_{4k-1}) \quad (11)$$

Where  $(N-2) \leq k \leq 1$

$$S_k = (b_{4k+2}b_{4k+1}b_{4k}b_{4k-1}Z), \text{ for } k=0 \quad (12)$$

if 'n' is odd,

$$S_k = (b_{4k+2}b_{4k+1}b_{4k}b_{4k-1}), k = N-1 \quad (13)$$

The multiples  $S_k/0, \pm 1, \pm 2, \pm 3, \pm 4, \pm 5, \pm 6, \pm 7, \pm 8$  are used to generate the partial products.

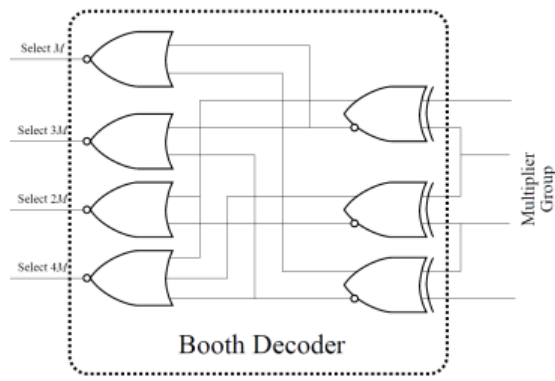


Fig. 3: Radix-8 Modified booth encoder

The final product can be written as

$$P = M \times R = M.S_{N-1}2^{4(N-1)} + M.S_{N-2}2^{4(N-2)} + \dots + M.S_12^4 + M.S_02^0 \quad (14)$$

The equation number 11 to 13 often generates hard multiples. In Radix-16 Booth encoding, the number of partial products is reduced by 1/4 [15]. However these reductions in the number of partial products come at the expense of increased complexity in their generation. Specifically, the generation of the Hard Multiples  $\pm 3A, \pm 5A, \pm 6A, \pm 7A, \pm 8A$  results increase in delay and carry - propagation chain.

To avoid this Hard Multiples, a HMT is proposed as shown in Figure 4. The block diagram has 3 sections, namely

HMT equipped booth encoder and partial product compressor. Section 1 is planned with R4, R8 and R16 MBA. Section 2 uses 4:2 counter based Carry Save Adder to compress the four main partial products from HMT block. The result from the 4:2 CSA stages is fed to section 3. It is Carry Look Ahead based fast adder to produce the final product [16]. The partial products with two's complement format creates sign extension crisis. Especially instead of hard multiples, adding the partial products with different bit length in R4 and R8 cause sign extension problem. If the sign bit is negative; it increases the number of overhead bits in partial products. So before partial product compression few modifications [15], [17] have to be done in 2's complement signed MBM.

The HMT scheme can be mathematically modelled by considering the Table 1. The multiplier groups which generates hard multiples in R16 Encoding scheme is shifted to R8 and R4 Encoding scheme.

In Radix-16 encoding scheme the partial product collection and summation are expressed by the equation

$$S_{N-1}2^{4(N-1)} + S_{N-2}2^{4(N-2)} + \dots + S_12^4 + S_02^0 \quad (15)$$

In the equation (15), if  $S_12^4$  has hard multiples this can be represent by lower radix encoders. Consider  $S_12^4$  is the combination of any hard multiples  $\pm 3A, \pm 4A, \pm 5A$ , then the new partial product can be written as

$$S_{N-1}2^{4(N-1)} + S_{N-2}2^{4(N-2)} + \dots + \{S_{N-1}2^{2(N-1)} + S_{N-2}2^{2(N-2)} + \dots + S_12^2 + S_02^0\} + S_02^0 \quad (16)$$

TABLE 1. HYBRID MULTIPLICATION ENCODER SELECTION

Radix 16	$B_k$ (Hard Multiples)	Encoder selector
00000	+0	Radix-16
00001, 00010	+1	$S_{N-1}2^{4(N-1)} + S_{N-2}2^{4(N-2)} + \dots$
00011, 00100	+2	$+ S_12^4 + S_02^0$
00101, 00110	+3	Radix-4
00111, 01000	+4	$S_{N-1}2^{2(N-1)} + S_{N-2}2^{2(N-2)} + \dots$
01001, 01010	+5	$+ S_12^2 + S_02^0$
01011, 01100	+6	Radix-8
01101, 01110	+7	$S_{N-1}2^{3(N-1)} + S_{N-2}2^{3(N-2)} + \dots$
01111	+8	$+ S_12^3 + S_02^0$
10000	-8	Radix-4
10001, 10010	-7	$S_{N-1}2^{2(N-1)} + S_{N-2}2^{2(N-2)} + \dots$
10011, 10100	-6	$+ S_12^2 + S_02^0$
10101, 10110	-5	Radix-16
10111, 11000	-4	$S_{N-1}2^{4(N-1)} + S_{N-2}2^{4(N-2)} + \dots$
11001, 11010	-3	$+ S_12^4 + S_02^0$
11011, 11100	-2	Radix-8
11101, 11110	-1	$S_{N-1}2^{3(N-1)} + S_{N-2}2^{3(N-2)} + \dots$
11111	-0	$+ S_12^3 + S_02^0$

If Radix-8 and Radix-4 partial products are used instead of Radix-16 encoder; at  $S_{N-2}2^{4(N-2)}$  and  $S_02^0$  partial product, respectively. The equation can be represented by

$$S_{N-1}2^{4(N-1)} + \{S_{N-1}2^{3(N-1)} + S_{N-2}2^{3(N-2)} + \dots + S_12^3 + S_02^0\} + \dots + S_12^4 + \{S_{N-1}2^{2(N-1)} + S_{N-2}2^{2(N-2)} + \dots + S_12^2 + S_02^0\} \quad (17)$$

Thus each combination of HMT comes with Radix-8, Radix-4 or both in Radix-16. Depend on the arithmetic

weight; the shifting operation should be performed before partial product compression. Unlike normal Radix-4 and Radix-8 recoding scheme, there is no virtual zero at the Least Significant Bit of the multiplier in lower order Radix-8 and Radix-4 HMT recoding scheme.

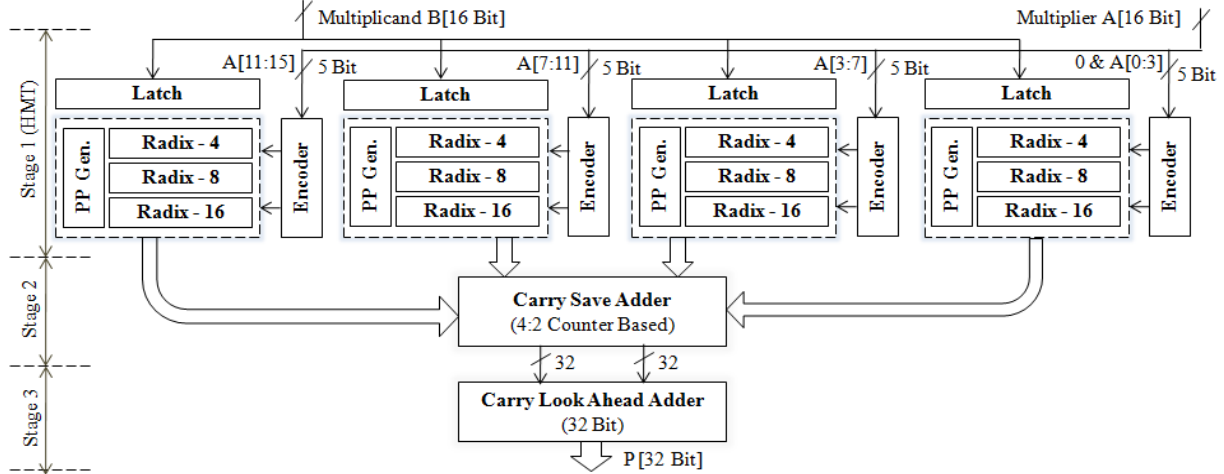


Figure 4: Proposed block diagram of HMT

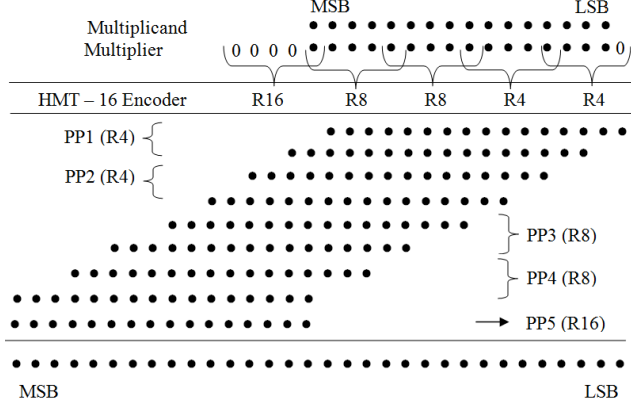


Fig. 5: Illustration of HMT

The partial product generation method is modelled by the use of a dot diagram [20] as given in Figure 5. Fig. 5 shows that the HMT encoder uses Radix-4(R4), Radix-8(R8) encoder along with Radix-16(R16). Each partial product is generated with the corresponding encoding scheme and differentiates by horizontal dotted rows. The partial products are shifted according to the arithmetic weight of the bits in the multiplier for different encoder. The final product is represented by 32 bit length dotted rows.

### III. SYNTHESIS AND ANALYSIS OF HMT

The proposed HMT-16, Conventional Radix 16, Radix-8 and Radix-4 multipliers are described in Verilog HDL and its RTL functionality verification is done in Synopsys *Verilog Compiler Simulator (VCS)* in the RH Linux AS v5.7 platform. The individual designs for Radix-16, Radix-8, Radix-4 & HMT-16 are optimized by Synopsys *Design Compiler (DC)* for minimum attainable area and delay independently under the same synthesis design environment.

The synthesized-constrained schematic diagram of Radix-4, Radix-8, Radix-16 and HMT-16 multipliers are shown in figure 6, 7, 8, 9. Total number of cells (excluding interconnects) used in Radix-4, Radix-8, Radix-16 & HMT-16 at 450MHz are 804, 1440, 1503 & 1662 respectively. From Table 2 it is clear that the HMT technique is much better than Radix-8 and Radix-16 encoding scheme. Synopsys *SDK* standard-cell library with operating voltage 1.32V and temperature -40 to +55 as min/max show that the proposed 16x16 bit Hybrid multiplier reduces the power consumption by 12%, 22% compared to conventional R16, R8 booth encoders, respectively. When compared to Radix-4, HMT-16 put away little extra Power. But it's negligible; when higher radix encoding with different frequency is taken. Also the data arrival time of the HMT-16 multiplier is better than any other schemes.

The performance assessment of the proposed multiplier with some existing designs like Li [4], Muralidharan [5],

Hsu [7], Wang [12] and Chen [18] gives more complete statistics as shown in Table (3). From this it is evident that the power optimization of this HMT-16 multiplier is better than any other design algorithm. The proposed idea can also be extended to Radix-32, as well as high Radix multipliers. In general power savings in multiplier is possible without compromising the system speed, offered that HMT is also a noncritical feed. On the other hand, due to the hard multiple terms connected with the radix-16 Booth encoding, addition and negation in modulo arithmetic, HMT based multiplication is a fine alternate.

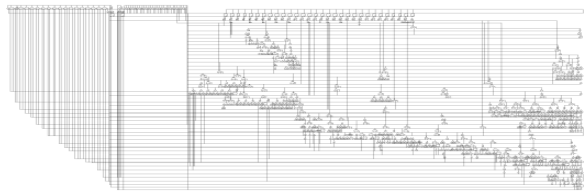


Figure 6: Radix-4 16x16 bit RTL at 450 MHz

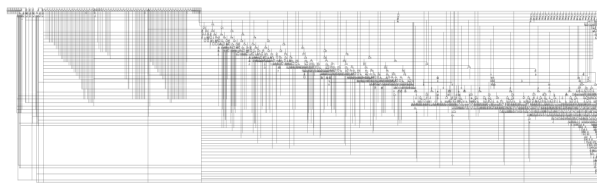


Figure 7: Radix-8 16x16 bit RTL at 450 MHz

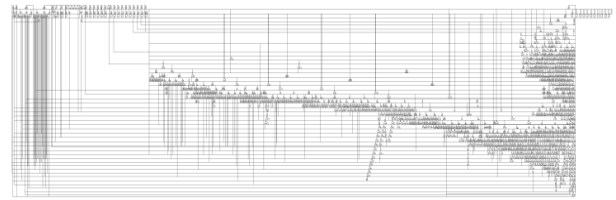


Figure 8: Radix-16 16x16 bit RTL at 450 MHz

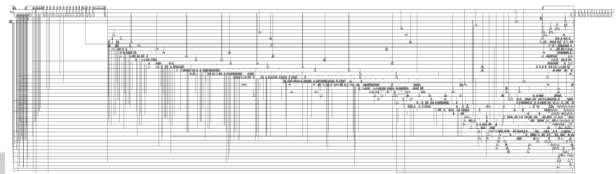


Figure 9: HMT-16 16x16 bit RTL at 450 MHz

TABLE 2: POWER REPORT OF DIFFERENT MBA

Parameters	Radix-4	Radix-8	Radix-16	HMT-16
Total area	10951.5	17182.3	17353.9	18638.4
Power (mW)	2.5921	3.4865	3.807	2.7267
Leakage Power (pW)	55.270	78.961	78.275	91.278
Date Arrival Time	-2.46	-2.17	-0.18	-0.17
Slack (+)	0.00	0.28	2.28	2.291
No. of buf/inv:	65	124	169	167

TABLE 3: 16x16 BIT HMT COMPARISONS WITH EXISTING TECHNIQUE

Design	Li [4]	Muralidharan [5]	Hsu [7]	Wang [12]	Chen [18]	Proposed HMT-16
Feature	32 × 32 Bit	64 × 64 Bit	16 × 16 Bit	8 × 8 Bit	16 × 16 Bit	16 × 16 Bit
Technology	0.13 μm	0.18 μm	90 nm	0.13 μm	0.25 μm	90 nm
Frequency	100 MHz	NA	1 GHz	125 MHz	NA	450 MHz
Voltage (V)	NA	1.8	1.3	1.2	NA	1.32
Power (μW)	12.1	NA	9.00	1.35	17.3	2.7267
Area (μm <sup>2</sup> )	85983	29086	30000	6683	33700	18638.4
Delay	3 ns	0.76	1	5.76	8.3	1.54

Further more detailed analysis of the HMT-16 static time analysis considered. It is characterized by data arrival time and data required time in an algorithm. The data arrival time is the time required for the signal to travel from path start point to a path end point. The data required time is the maximum time a signal has taking for traveling that path. The difference of data required time and data arrival time is called slack or timing margin of the path. If slack is negative, there is a timing violation on that path.

Histogram views can be used to analyze the distribution of timing slack values in the design. Figure (10) shows the positive end-point path slack histogram of the entire algorithm at 450MHz. It shows that the HMT equipped Radix-16 multiplier's positive endpoint slack is better than any other conventional design. Figure 10 shows that the positive end-point slack path of HMT-16 MBA after the slack unit +2.30 is 13 out of 32. That means HMT has good tolerance in constraints. Table 4 gives the best and worst nets capacitances and positive path slack of all MBA. HMT-16 encoder

consumes 23%, 66% and 37% less capacitance in worst case, while compare with Radix-4, Radix-8 and Radix-16, respectively. HMT-16 positive path slack is also better than other design. These are the strong reasons why proposed HMT-16 equipped modified booth multiplier consumes less power than conventional methods. From this it is distinct that this HMT-16 multiplier gives better power reduction in the different bit ranges and frequency.

Table 4: Nets Capacitance of Different MBA

MBA	Case	Nets Capacitance (pf)	+ve Path Slack
Radix-4	Best	0.00773	0.2449
	Worst	2.27169	0.0009
Radix-8	Best	0.00773	0.6232
	Worst	5.21393	0.2811
Radix-16	Best	0.00773	2.3118
	Worst	2.77697	2.2759
HMT-16	Best	0.00411	2.3233

	Worst	1.73553	2.2897
--	-------	---------	--------

## CONCLUSION

In this paper, a well-configured HMT is proposed for Radix-16 MBA. The design is fast and consumes less power because this HMT-16 MBA uses 3-bit and 4-bit modified Booth encoder. Rearranging and reducing partial products and reduction of higher radix hard multiple is possible, which need speed and accuracy. According to the experimental results, the design can obtain 22%, 26% reduction on the power savings over the Radix-8 MBA, Radix-4 MBA design. Proposed method can be used for Multimedia applications like texture coding of H.264, DSP application and MAC unit design. The future work can be extended by using Radix-32, Radix-64 and higher radix MBA with higher bit range for a better power reduction and speed improvement.

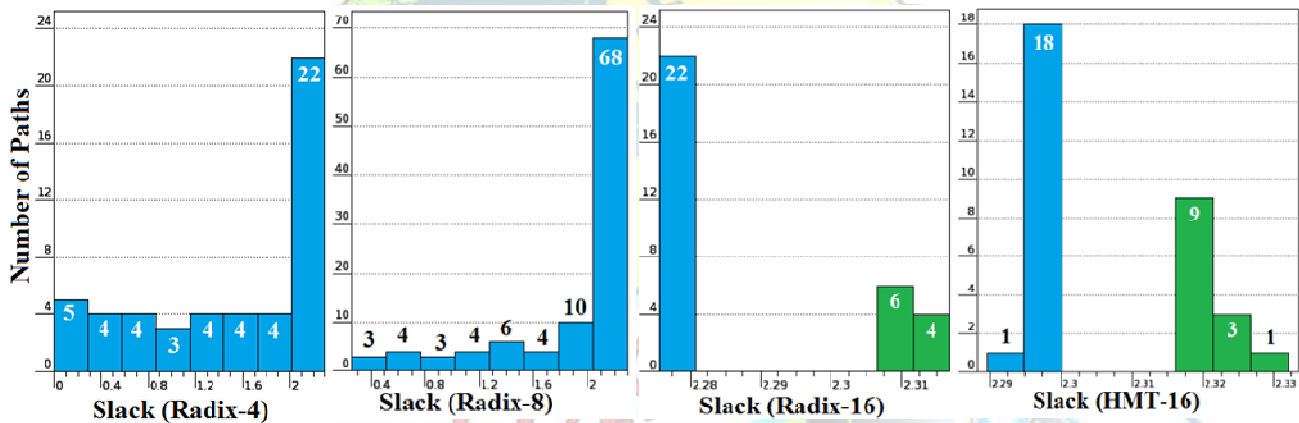


Fig. 10: Positive Endpoint slack histogram of different MBA

## REFERENCES

- [1] L.P. Rubinfeld, "A Proof of the Modified Booth's Algorithm for Multiplication," IEEE Transactions on Computers, vol. C-24, pp.1014-1015, Oct. 1975.
- [2] Villegier, V.G. Oklobdzija, "Analysis of Booth Encoding Efficiency in Parallel Multipliers Using Compressors for Reduction of Partial Products," Proc. IEEE 27th Asilomar Conf. Comp. Systems and Signals, vol. 1, pp. 781-784, 1993.
- [3] Villegier, V.G. Oklobdzija, "Evaluation of Booth Encoding Techniques for Parallel Multiplier Implementation," Electronics Letters, vol.29, pp. 2016-2017, Nov. 1993.
- [4] L.R. Wang, S.J. Jou, C.L. Lee, "A well-structured modified Booth multiplier design," IEEE International Symposium on VLSI Design Automation and Test, pp.85-88, April 2008.
- [5] R. Muralidharan, C.H. Chang, "Fast hard multiple generators for radix-8 Booth encoded modulo  $2^n-1$  and modulo  $2^n+1$  multipliers," Proceedings of 2010 IEEE International Symposium on Circuits and Systems, pp.717-720, May 30 - June 2 2010.
- [6] H. Sangjin, K. Suhwan, M.C. Papaefthymiou, W.E. Stark, "Low power parallel multiplier design for DSP applications through coefficient optimization," Proceedings of Twelfth Annual IEEE International ASIC/SOC Conference, pp.286-290, 1999.
- [7] S.K. Hsu, S.K. Mathew, M.A. Anders, B.R. Zeydel, V.G. Oklobdzija, R.K. Krishnamurthy, et. al, "A 110 GOPS/W 16-bit multiplier and reconfigurable PLA loop in 90-nm CMOS," IEEE Journal of Solid-State Circuits, vol.41, pp.256-264, Jan. 2006.
- [8] B.S. Cherkauer, E.G. Friedman, "A hybrid radix-4/radix-8 low power, high speed multiplier architecture for wide bit widths," IEEE International Symposium on Circuits and Systems, pp.53-56, May 1996.
- [9] W.C. Yeh, C.W. Jen, "High-speed Booth encoded parallel multiplier design," IEEE Transactions on Computers, vol.49, pp. 692-701, Jul 2000.
- [10] L. R. Wang; S.J. Jou, C.L. Lee, "A well-structured modified Booth multiplier design," , IEEE International Symposium on VLSI Design Automation and Test, pp.85-88, April 2008
- [11] R. Muralidharan, C.H. Chip, "Hard multiple generator for higher radix modulo  $2^n-1$  multiplication," Proceedings of the 2009 12th



ISSN 2394-3777 (Print)

ISSN 2394-3785 (Online)

Available online at [www.ijartet.com](http://www.ijartet.com)

**International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)**  
**Vol. 1, Special Issue 11, November 2014**

- International Symposium on Integrated Circuits, pp.546-549, December 2009.
- [12] J.P. Wang, S.R. Kuang, S.C. Liang, "High-Accuracy Fixed-Width Modified Booth Multipliers for Lossy Applications," IEEE T. Very Large Scale Integration (VLSI) Systems, vol.19, pp.52-60, Jan. 2011.
- [13] S.D. Pezaris, A 40-ns 17-bit by 17-bit array multiplier. IEEE Transaction on Computers; Vol. C-20, pp. 442-447, 1971.
- [14] A.S. Min, D.V. Lan, C.H. Ting, Y.K.Sy, "A generalized methodology for low-error and area-time efficient fixed-width Booth multipliers," 47th IEEE International Midwest Symposium on Circuits and Systems; pp. 9-12, July 2004.
- [15] W.B. Gary, "Fast multiplication: Algorithms and implementation," PhD Thesis, Stanford University, California, United States, 1994.
- [16] Z. Huang, M.D. Ercegovic, "High-performance low-power left-to-right array multiplier design," IEEE Transactions on Computers, vol.54, pp.272-283, March 2005.
- [17] M. Roorda, "Method to reduce the sign bit extension in a multiplier that uses the modified booth algorithm," 1986 electronic lett. vol.22, pp. 1061-62, August 1986.
- [18] Chen, T.C. Oskal, W. Sandy, Y.W. Wu, "Minimization of switching activities of partial products for designing low-power multipliers," IEEE T. Very Large Scale Integration (VLSI) Systems, vol.11, pp.418-433, June 2003.

