Indonesian Journal of Electrical Engineering and Computer Science
Vol. 13, No. 3, March 2019, pp. 1048~1055
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v13.i3.pp1048-1055

Design of modified booth based multiplier with carry pre-computation

Chaitanya CVS[1], Sundaresan C[2], P R Venkateswaran[3], Keerthana Prasad[4]
[1,2,4]School of Information Sciences, Manipal Academy of Higher Education, India
[3]Bharat Heavy Electricals Limited, India

Article Info

Article history:

Received Oct 11, 2018
Revised Dec 9, 2018
Accepted Dec 22, 2018

Keywords:

Binary Multiplication
Carry Pre-Computation
Modified Booth Multiplier
Multiplier Architecture
Vedic Multiplier

ABSTRACT

Arithmetic unit is the most important component of modern embedded computer systems. Arithmetic unit generally includes floating point and fixed-point arithmetic operations and trigonometric functions. Multipliers units are the most important hardware structures in a complex arithmetic unit. With increase in chip frequency, the designer must be able to find the best set of trade-offs. The ability for faster computation is essential to achieve high performance in many DSP and Graphic processing algorithms and is why there is at least one dedicated Multiplier unit in all of the modern commercial DSP processors. Tremendous advances in VLSI technology over the past several years resulted in an increased need for high speed multipliers and compelled the designers to go for trade-offs among speed, power consumption and area. A novel modified booth multiplier design for high speed VLSI applications using pre-computation logic has been presented in this paper. The proposed architecture modeled using Verilog HDL, simulated using Cadence NCSIM and synthesized using Cadence RTL Compiler with 65nm TSMC library.The proposed multiplier architecture is compared with the existing multipliers and the results show significant improvement in speed and power dissipation.

Corresponding Author:

Chaitanya CVS
School of Information Sciences,
Manipal Academy of Higher Education,
Manipal 576104, Karnataka, India.
Email: chaitanya.cvs@manipal.edu

1. INTRODUCTION

Processors are important part of integrated circuits (IC). Large numbers of functionalities are packed in an IC thanks to tremendous growth in density of integration in recent times. As the number of functions increases, the need for computation also grows. With the advent of new process technologies, shrinking of feature size and availability of modern CAD tools, a development of complex integrated circuits for various applications is possible. Examples of such applications include digital signal processing [1], [2], mobile computations and communications, multimedia applications and processing required for scientific computing and applications etc. The speed and efficiency of processor in such IC is very crucial for meeting the requirements of the applications supported by the IC. The speed of processor and efficiency of processor in-turn depends upon an arithmetic logic unit [3] which is considered as the main computational unit of the processor.

Moreover, the multiplier units [4] are the most important hardware structures in a complex arithmetic unit. The multiplier units are capable of performing operations on operands of various data types such as calculating running sum of products. As multiplication is a crucial arithmetic operation in processors [5] and digital computer systems, multipliers are the core building block for many algorithms in a wide variety of computing applications. Although multipliers are main arithmetic components used for processing scientific data, the excessive power consumption and delay attracts attention from the research community. Usually,

multiple arithmetic cores working in parallel are used so as to process large amounts of data with relatively low power and delay.

However, in the high-speed processors which are operating at higher clock frequencies, the existing multiplier takes more delay for execution of the instructions. The existing multiplier units that consume more power are not suitable to be incorporated in the processors which are used in wireless and portable devices. Thus, power savings is an important area for improvement.

In order to address the low power computation along with high performance, a new approach to multiplier design based on ancient Vedic Mathematics has been explored. The mathematical operations using Vedic mathematics are very fast and require less hardware. This aspect of Vedic mathematics can be utilized to increase the computational speed of multipliers. This paper describes the design and implementation of a Vedic multiplier based on *Urdhva-Tiryagbhyam* Sutra [6]-[9]. The number of steps required to perform a multiplication operation by using *UrdhvaTiryagbhyam* Sutra are considerably less compared to the conventional multiplication techniques. In this paper, we have further explored a novel method to enhance the speed of a Vedic multiplier by pre-computing the carries which are used during summation of partial products. The implementation of pre-computation logic using multiplexer-based carry-look ahead logic and XOR logic resulted in reduction of delay. The proposed carry pre-computation is used along with Modified Booth methodology resulted in further reduction of delay in performing multiplication operation.

The structure of the paper is divided as follows: The methodology and the architecture of the proposed multipliers are given in section 2. Results are presented in section 3. Finally, conclusion is given in section 4.

2. RESEARCH METHOD

2.1. Carry Pre-Computation Based Binary Multiplier

An 8-bit Binary Vedic Multiplier has been proposed with A and B as inputs and P as the final 16-bit product. The block diagram for 8-bit multiplication is shown in Figure 1. In the proposed multiplier the operands A and B are divided into Higher and Lower parts with 4-bits each.

$$A = \{AH, AL\} \quad (1)$$

$$B = \{BH, BL\} \quad (2)$$

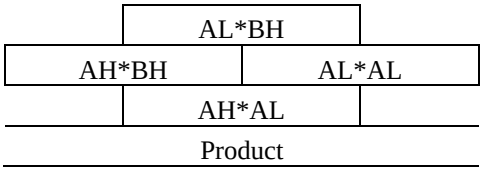

Figure 1. Block Diagram of 8-bit Multiplication

In this type of multiplier an 8-bit Binary multiplication is realized using 4-bit binary vedic multiplication using carry pre-computation logic shown in Figure 2, where A_3, A_2, A_1, A_0 & B_3, B_2, B_1, B_0 are 4-bit binary inputs and P_7, P_6, P_5, P_4, P_3, P_2, P_1, P_0 are the binary output bits.

The partial product generator is the first block of the multiplier to which the 4-bit multiplicand and multiplier are given as inputs. At this juncture, the multiplication technique used is Urdhva-Tiryagbhyam. The 4-bit multiplication results in a total of 16 partial products (pp_1-pp_{16}). The result of multiplying any one binary bit with another is either a zero or a one which is simply the logic of ANDing of the two bits.

The second stage in the block diagram is the carry generation circuit. Here, we have integrated pre-computation logic along with the Urdhva-Tiryagbhyam multiplication technique. The carry equations are generated separately for each column of partial products and the inputs for these equations are taken from the previous column. The equations for pre-computed carries are given.

$$c_2 = pp_5 \,\&\, pp_2; \quad (3)$$

$$c_3t_1 = (pp_6 \,\&\, pp_3) \mid (pp_9 \,\&\, (pp_3 \mid pp_6)); \quad (4)$$

$$c_3t_2 = (pp_9 \,\&\, \sim pp_6) \mid (pp_3 \,\&\, \sim pp_9) \mid (\sim pp_3 \,\&\, pp_6); \quad (5)$$

$$c_{31} = c_2 ? c_3t_2 : c_3t_1; \quad (6)$$

$$c_{32} = pp_2 \,\&\, pp_5 \,\&\, pp_3 \,\&\, pp_6 \,\&\, pp_9; \quad (7)$$

$$c_{41}t_1 = pp_{13} ? ((pp_{10} \,\&\, \sim pp_7) \mid (pp_4 \,\&\, \sim pp_{10}) \mid (\sim pp_4 \,\&\, pp_7)) : ((pp_7 \,\&\, pp_4) \mid (pp_{10} \,\&\, (pp_4 \mid pp_7))); \quad (8)$$

$$c_{41}t_2 = pp_{13} ? ((\sim pp_7 \,\&\, \sim pp_4) \mid (\sim pp_{10} \,\&\, (\sim pp_4 \mid \sim pp_7))) : ((\sim pp_7 \,\&\, pp_4) \mid (pp_{10} \,\&\, \sim pp_4) \mid (\sim pp_{10} \,\&\, pp_7)); \quad (9)$$

$$c_{41} = c_{31} ? c_{41}t_2 : c_{41}t_1; \quad (10)$$

$$c_{42} = ((c_{31} \,\&\, pp_{13}) \,\&\, ((pp_{10} \,\&\, (pp_7 \mid pp_4)) \mid (pp_7 \,\&\, pp_4))) \mid ((pp_{10} \,\&\, pp_7 \,\&\, pp_4) \,\&\, (c_{31} \mid pp_{13})); \quad (11)$$

$$c_{51}t_1 = c_{32} ? ((pp_{14} \,\&\, \sim pp_{11}) \mid (pp_8 \,\&\, \sim pp_{14}) \mid (\sim pp_8 \,\&\, pp_{11})) : ((pp_{11} \,\&\, pp_8) \mid (pp_{14} \,\&\, (pp_8 \mid pp_{11}))); \quad (12)$$

$$c_{51}t_2 = c_{32} ? ((\sim pp_{11} \,\&\, \sim pp_8) \mid (\sim pp_{14} \,\&\, (\sim pp_8 \mid \sim pp_{11}))) : ((\sim pp_{11} \,\&\, pp_8) \mid (pp_{14} \,\&\, \sim pp_8) \mid (\sim pp_{14} \,\&\, pp_{11})); \quad (13)$$

$$c_{51} = c_{41} ? c_{51}t_2 : c_{51}t_1; \quad (14)$$

$$c_{52} = ((c_{41} \,\&\, c_{32}) \,\&\, ((pp_{14} \,\&\, (pp_{11} \mid pp_8)) \mid (pp_{11} \,\&\, pp_8))) \mid ((pp_{14} \,\&\, pp_{11} \,\&\, pp_8) \,\&\, (c_{41} \mid c_{32})); \quad (15)$$

$$c_6t_1 = (pp_{12} \,\&\, pp_{15}) \mid (c_{42} \,\&\, (pp_{12} \mid pp_{15})); \quad (16)$$

$$c_6t_2 = (c_{42} \,\&\, \sim pp_{15}) \mid (pp_{12} \,\&\, \sim c_{42}) \mid (pp_{15} \,\&\, \sim pp_{12}); \quad (17)$$

$$c_{61} = c_{51} ? c_6t_2 : c_6t_1; \quad (18)$$

$$c_{62} = c_{51} \,\&\, c_{42} \,\&\, pp_{12} \,\&\, pp_{15}; \quad (19)$$

$$c_{71} = (c_{52} \,\&\, pp_{16}) \mid (c_{61} \,\&\, (c_{52} \mid pp_{16})); \quad (20)$$

		A_3	A_2	A_1	A_0		
	X	B_3	B_2	B_1	B_0		
				pp_4	pp_3	pp_2	pp_1
			pp_8	pp_7	pp_6	pp_5	
		pp_{12}	pp_{11}	pp_{10}	pp_9		
	pp_{16}	pp_{15}	pp_{14}	pp_{13}			
			c_{32}	c_{31}	c_2		
		c_{42}	c_{41}				
	c_{52}	c_{51}					
c_{62}	c_{61}						
c_{71}							
P_8	P_7	P_6	P_5	P_4	P_3	P_2	P_1

Figure 2. Carry Pre-Computation Based Multiplier

The architecture of the 4-bit multiplier can be understood from the block diagram as shown in Figure 3.

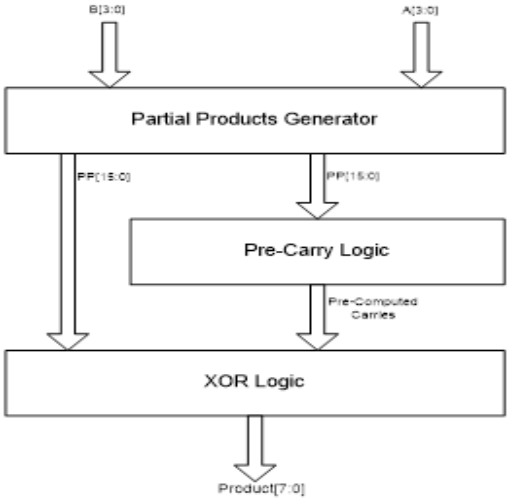

Figure 3. Architecture of Carry Pre-Computation based Multiplier

The third stage in the block diagram involves the use of XOR logic for the partial products and carry generated in each column. The output of this stage gives the final 16-bit product which is obtained in a parallel mechanism instead of sequential mechanism. The products of AL*BL, AH*BL, AL*BH, AH*BH are determined using above 4-bit carry pre-computation-based multiplier and the results of all sub multipliers are added to determine the final product. The block of the 8-bit multiplier as shown in Figure 4.

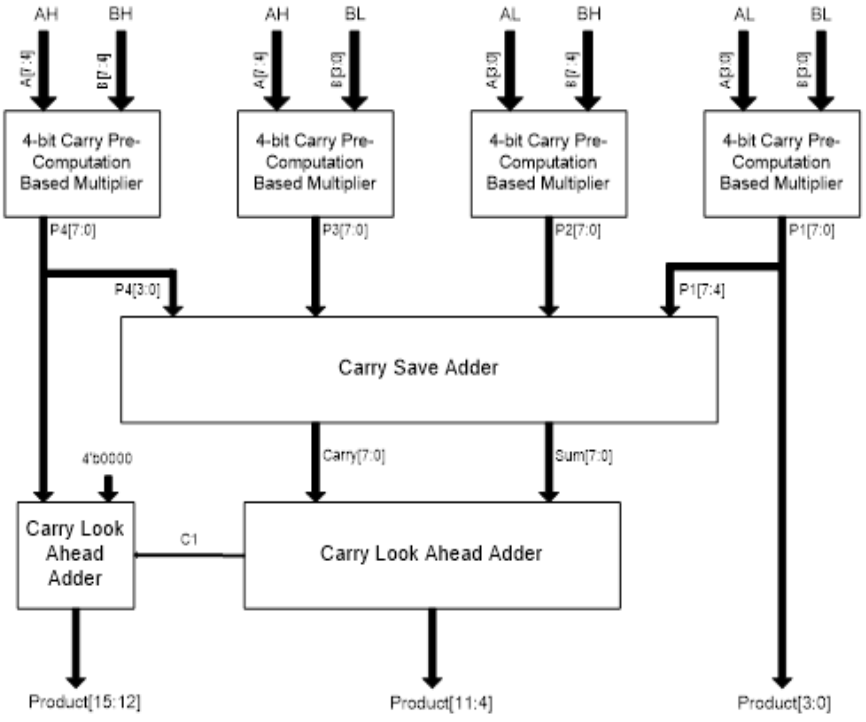

Figure 4. Block Diagram of 8-bit Multiplier Using 4-bit Carry Pre-Computation Based Multiplier

2.2. Modified Booth Based Binary Multiplier with Carry Pre-Computation

An 8-bit Binary Modified Booth based multiplier with carry pre-computation has been proposed. The architecture of the 8-bit multiplier can be understood from the block diagram as shown in Figure 5.

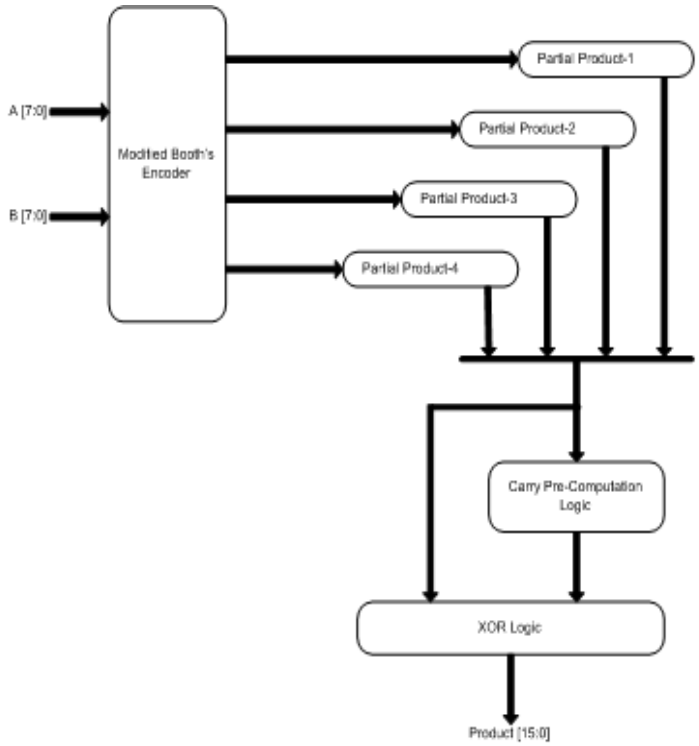

Figure 5. Modified Booth Multiplier with Carry Pre-Computation Logic

The modified booth encoding is the first block of the multiplier to which the 8-bit multiplicand and multiplier are given as inputs. The modified booth encoder generates 4 partial products based on the following Table 1.

Considering that both inputs A and B are of 8 bits. As inputs are 8 bits, four partial products will be generated as shown in Figure 6.

Table 1. Modified booth encoding

Xi+1	Xi	Xi-1	Action
0	0	0	0 × Y
0	0	1	1 × Y
0	1	0	1 × Y
0	1	1	2 × Y
1	0	0	-2 × Y
1	0	1	-1 × Y
1	1	0	-1 × Y
1	1	1	0 × Y

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
s1	s1	s1	s1	s1	s1	s1	p18	p17	p16	p15	p14	p13	p12	p11	p10
s2	s2	s2	s2	s2	p28	p27	p26	p25	p24	p23	p22	p21	p20		
s3	s3	s3	p38	p37	p36	p35	p34	p33	p32	p31	p30				
s4	p48	p47	p46	p45	p44	p43	p42	p41	p40						
c132	c122	c112	c102	c92	c82	c72	c62	c52	c42	c41	c3	c2			
c141	c131	c121	c111	c101	c91	c81	c71	c61	c51						

Figure 6. Partial Products of Modified Booth Multiplier

Partial product – 1: p10 to p18 and s1 is the sign extension
Partial product – 2: p20 to p28 and s2 is the sign extension
Partial product – 3: p30 to p38 and s3 is the sign extension
Partial product – 4: p40 to p48 and s4 is the sign extension

In Figure,
c2 is carry generated from column 2
c3 is carry generated from column 3
c41 and c42 is carries generated from column 4
c51 and c52 is carries generated from column 5
c61 and c62 is carries generated from column 6
c71 and c72 is carries generated from column 7
c81 and c82 is carries generated from column 8
c91 and c92 is carries generated from column 9
c101 and c102 is carries generated from column 10
c111 and c112 is carries generated from column 11
c121 and c122 is carries generated from column 12
c131 and c132 is carries generated from column 13
c141 is carry generated from column 14

The second stage in the block diagram is the carry generation circuit. Here, we have integrated pre-computation logic along with the modified booth multiplication technique. The carry values are generated separately for each column of partial products and the inputs for these equations are taken from the previous column. The carry pre-computation logic discussed in section 2.1 is used to generate the values. The third stage in the block diagram involves the use of XOR logic for the partial products and carry generated in each column. The output of this stage gives the final 16-bit product which is obtained in a parallel mechanism instead of sequential mechanism. In proposed multiplier, as Carry Pre-Computation unit computes all the carries in parallel using carry look ahead logic and remove dependencies between columns, the total time required to generate the product will be reduced.

3. RESULTS AND ANALYSIS

The proposed architecture modeled using Verilog HDL, simulated using Cadence NCSIM and synthesized using Cadence RTL Compiler with 65nm TSMC library. Different implementation methodology has been taken and implemented in same technological environment and then compared the performance parameters. For the comparison point of view the ideas have been considered from the references and simulated and performance parameters were computed using the same MOSFET technology file. Input data was taken in a regular fashion for experimental purpose. The delay and the power measured using the worst-case pattern and from the output where the delay is maximum.

It is observed that the proposed carry pre-computation-based multiplier and modified booth multiplier with carry pre-computation based offered substantial reduction of propagation delay and total power consumptions. From Table 1 and Table 2, it can be observed that the proposed carry pre-computation based multiplier design offered ~23%, ~64%, ~57%, ~83% when compared with array multiplier, wallace multiplier, column based multiplier, and Nikhilam based based multipliers respectively, and modified booth multiplier with carry pre-computation offered ~83%, ~92%, ~91%, ~97% when compared with array multiplier, wallace, column based, and Nikhilam based multipliers respectively.

Table 2. Summary of synthesis results of 8-bit multiplier architectures

S. No	Architecture (8-bit)	Delay (ns)	Dynamic Power (uW)	Static Power (uW)	Total Power (uW)	Power-Delay Product (pJ)
1	Array Based Multiplier [9]	1.5	15.09	6	21.09	31.63
2	Wallace Based Multiplier [2]	1.2	6.27	49.913	56.184	67.42
3	Column Based Multiplier [6]	1.95	26.74	2.8	29.54	57.6
4	Nikhilam Based Multiplier [7]	3.2	42.56	4.3	46.86	149.95
5	Pre-Computation Based Multiplier	0.75	25.77	7.45	33.23	24.23
6	Modified Booth's Multiplier with Carry Pre-Computation	0.45	9.4	3.99	13.39	6.02

From the Table 2 and Table 3, it can be observed that Modified Booth based multiplier with carry pre-computation consumes less delay when compared to carry pre-computation-based multiplier with the delay tradeoff. Proposed Modified Booth based multiplier with carry pre-computation gave the better power-delay product when compared to proposed carry pre-computation-based multiplier and existing multiplier from literature.

Table 3. Summary of synthesis results of 16-bit multiplier architectures

S. No	Architecture (16-bit)	Delay (ns)	Dynamic Power (uW)	Static Power (uW)	Total Power (uW)	Power-Delay Product
1	Array Based Multiplier [9]	2.89	30.18	12	42.18	121.90
2	Wallace Based Multiplier [2]	2.46	12.54	99.826	112.366	276.42
3	Column Based Multiplier [6]	3.82	52.48	5.4	57.88	221.10
4	Nikhilam Based Multiplier [7]	5.96	80.65	8.1	88.75	528.95
5	Pre-Computation Based Multiplier	1.4	51.54	14.9	66.44	93.01
6	Modified Booth's Multiplier with Carry Pre-Computation	0.84	17.23	8.3	25.53	21.44

4. CONCLUSION

In this paper, a Vedic mathematics-based multiplier has been proposed which uses Carry pre-computation and operand decomposition methodology. The proposed architectures combine the benefits of Vedic method and parallel pre-computation of carries thereby resulting in reduction of power-delay product. The propagation delay of carry pre-computation-based multiplier for calculation of 8 bit and 16-bit multiplication was 0.75ns and 1.4ns while power consumption was 33.23 uW and 66.44 uW. The propagation delay of modified booth multiplier with carry pre-computation for calculation of 8 bit and 16-bit multiplication was 0.45ns and 0.84ns while power consumption was 13.39 uW and 25.53 uW. The delay of multiplication was decreased by ~85% and power consumption were reduced by ~88% when compared to Nikhilam based Vedic multiplier.

REFERENCES

[1] Xiangui Kang, AnjiePeng, XianyuXu, Xiaochun Cao, "Performing Scalable Lossy Compression on Pixel Encrypted Images," *EURASIP Journal on Image and Video Processing*, pp. 1-6, 2013.
[2] Nikolay Ponomarenko, Sergey Krivenko, Vladimir Lukin, Karen Egiazarian, Jaakko T., Astola, "Lossy Compression of Noisy Images Based on Visual Quality: A Comprehensive Study, " *EURASIP Journal on Advances in Signal Processing*, pp. 1-13, 2010.
[3] L.-K. Wang, M. A. Erle, C. Tsen, E. M. Schwarz, and M. J. Schulte, "A survey of hardware designs for decimal arithmetic," *IBM Journal of Research and Development*, vol. 54(2), pp. 8:1-8:15, 2010.
[4] M. Jeevitha, R. Muthaiah, P. Swaminathan, "Efficient Multiplier Architecture in VLSI Design," *Journal of Theoretical and Applied Information Technology*, vol. 38(2), pp. 196-201.2, 2012.
[5] J. R. Boddie, G. T. Daryanani, I. I. Eldumiati, R. N, Gadenz, J. S. Thompson, S. M. Walters, "Digital Signal Processor: Architecture and Performance," *Bell System Technical Journal*, vol. 60(7), pp. 1449-1462, 1981.
[6] BharatiKrsnaTirthaji, V. S Agrawala, "Vedic Mathematics", *13th Edition, Motilal Banarsidass*, 2010.
[7] P. Saha, A. Banerjee, A. Dandapat, and P. Bhattacharyya, "ASIC design of a high-speed low power circuit for factorial calculation using ancient Vedic mathematics, " *ELSEVIER Microelectronics Journal*, vol. 42(12), pp. 1343-1352, Dec 2011.
[8] MD. Belal Rashid, Balaji B.S and Prof. M.B. Anandaraju, "VLSI Design and Implementation of Binary Multiplier based on UrdhvaTiryagbhyam Sutra with reduced Delay and Area," *International Journal of Engineering Research and Technology*, vol. 6(2), pp. 269-278, Mar 2013.
[9] Ko-Chi Kuo, Chi-Wen Chou, "Low Power and High-Speed Multiplier Design with Row Bypassing and Parallel Architecture," *Microelectronics Journal*, vol. 41, pp. 639-650, 2010.
[10] Constantinos Efstathiou, N. Moshopolous, N. Axelos, K. Pekmestzi, Efficient Modulo 2n+1 Multiply and Multiply-Add Units Based on Modified Booth Encoding, Integration, the VLSI Journal, 47 (2014), pp. 140-147.
[11] Manas Ranjan Meher, Ching Chuen Jong, and Chip-Hong Chang, "A High Bit Rate Serial-Serial Multiplier with On-the-Fly Accumulation by Asynchronous Counters", IEEE trans. On VLSI systems, Vol. 19, No. 10, pp. 1733-1745, October, 2011.

BIOGRAPHIES OF AUTHORS

	Chaitanya CVS received his Bachelor Degree in Electronics and Communication Engineering in 2006 from JNTU, Hyderabad and his MS degree in VLSI-CAD from Manipal University in 20 007. In 2010, he started his career as Assistant Professor in School of Information Sciences, Manipal. Currently, he is doing Ph. D at Manipal University. His research interest includes High Performance Computer Arithmetic, Advanced Computer Architecture, Low-power VLSI Design, Electronic Design Automation, and Parallel Algorithms/Architectures.
	Dr. C Sundaresan completed Bachelor degree in Electronics and Communication in 2000 from Madurai Kamaraj University and MS degree in VLSI CAD in 2003 from Manipal University and PhD in 2018 from Manipal Academy of Higher Education. He started his career as R & D engineer at Aplab Ltd. Currently he is working as Assistant Professor in School Of Information Sciences. His research interests include Computer Arithmetic, Low-Power VLSI Design, Logic Synthesis, Static Timing Analysis.
	Dr. P. R. Venkateswaran obtained his bachelor's degree in Electronics and Instrumentation Engineering from National Engineering College, Kovilpatti in 1998 and Masters in Instrumentation and Control Engineering from Technical Teachers' Training Institute, Chandigarh in 2002. He completed his doctoral research in 2008 from Manipal University, Manipal. He started his career as teaching faculty at Sethu Institute of Technology, Madurai and continued his teaching career with Technical Teachers' Training Institute, Chandigarh and later at Manipal Institute of Technology, Manipal. Presently, he is working as Senior Engineer (Control and Instrumentation) at Welding Research Institute, BHEL, Tiruchirappalli and is associated in the areas of Welding Automation and Welding Power Sources. His areas of interest are linear Control theory, Electronic Instrumentation and Soft Computing Techniques. He has been a reviewer for journals like IEEE SMC, Elsevier, AMSE etc. He is a member of professional bodies of ISTE, IWS and IE.
	Dr. Keerthana Prasad is working as Professor in School of Information Sciences, a constituent institution of Manipal University. Her research interests are image analysis and its applications in medicine and high-performance computing approach for image processing.