Performance analysis of BCD Multipliers with Vedic BCD Multiplier

Arvind Kumar Mehta, Vipin Jain
Department of Computer Science and Engineering
Swami Keshvanand Institute of Technology, Management & Gramothan, Jaipur
Email- mehta.hkc@gmail.com, vipin@skit.ac.in

Abstract: **Decimal data processing applications have grown at a very fast rate in recent years. The IEEE 754-2008 standard for floating point arithmetic has already dictated the importance of decimal arithmetic. In Computer Science where the demand of accurate data processing is highly required, decimal arithmetic plays an important role to support the most accurate data processing at the level of financial and scientific calculations where errors aren't bearable. In most hardware approaches that have been proposed for decimal arithmetic, the implementation is very expensive in terms of occupied resources and path delay. So, in this paper we have proposed Vedic BCD multiplier using the Vedic mathematics. The analyzed synthesized results will clearly explain the performance of Vedic BCD multiplier as compared to previously proposed BCD multipliers.**

Keywords: **Vedic Mathematics, Urdhva-Triyakbhyam, Vedic BCD digit multiplier, Vedic binary Multiplier, Nikhilam.**

1. INTRODUCTION

Decimal multiplication is very frequently used operation in many decimal applications. IEEE–754 standard for floating point arithmetic already incorporates specifications for decimal arithmetic. Thus, it is expected that microprocessor manufacturers include decimal floating–point units in their products oriented to mainframe servers to satisfy the high performance demands of current financial, commercial, banking calculation, currency conversion, insurance, telephone billing, accounting, scientific, tax calculation and user–oriented applications where we cannot tolerate errors. In many databases, numbers are in decimal format. Since many decimal numbers cannot be represented exactly as binary numbers with a finite number of bits, arithmetic operations must be done directly over decimal numbers [1-3].

To avoid errors associated with decimal to binary conversion decimal operations use software algorithms based on binary arithmetic. However, software solutions are very slow. Typically, three or four orders of slower magnitude than binary arithmetic are implemented in hardware [4]. Since decimal applications are increasingly more computationally demanding, it is important to implement decimal operations in hardware. The implementation process of BCD multiplication is more complicated than binary multiplication due to the inherent difficulty to represent decimal numbers using binary number system.

Recently, many hardware approaches have been proposed for parallel decimal multiplier, but typically all of them are very complex, occupying more slices and LUTs (4-input look-up tables) leading to more power and less speed when implemented in hardware. Hence, the major consideration while implementing decimal arithmetic is to enhance its speed and reduce slices as much as possible.

The multiplication process is major consideration to design BCD multiplier. So we are considering this fact, a high speed modified Vedic multiplication process [5] has been proposed in this paper, which is more efficient for hardware design.

In a very high speed area efficient Vedic BCD multiplier has been proposed for VLSI applications by using Nikhilam sutra, but when both operands' digits are far from their base then it will not be useful because it doesn't reduce digits from operands.

Section 2 describes Vedic sutras and Vertical-cross method, an implementation of Vedic BCD multiplier. Section 3 describes binary multiplier and binary to BCD converter. Section 4 describes designing of single-digit and multi-digit BCD multiplier. Section 5 presents our synthesis results and the last, Section 6 is conclusion and future works.

2. VEDIC SUTRAS AND VERTICAL-CROSS METHOD

Vedic mathematics is an ancient mathematics which is based upon 16 Vedic sutras and 14 sub-sutras, which are being applied on various streams of mathematics to make calculations easier, faster, efficient and highly optimized. Out of 16 there are mainly 3 sutras and 2 sub-sutras are given for multiplication, they are shown in table I.

Table 1: Vedic Multiplication Sutras Along With Their Brief Meaning

	Sutras	
1	Urdhva-tiryakbhyam	–Vertically and crosswise.
2	NikhilamNavatashcaramamDashatah	–All from 9 and last from 10
3	Anurupyena	–Proportionality
4	EkanyunenaPurvena	–By one less than the previous one.
5	Antyayordasake'pi	–Last totaling to ten

A. Various Vedic multiplication sutras

Nikhilam is not a universal method for decimal numbers because at least one operand has to be in the near power of 10. Anurupyena, which gives solution to this problem, is again not a good choice as it requires multiplication or division Ekanyunena Purvena and Antyayordasake'pi are astronomic

application specific sutras. But, Urdhva-Tiryagbhyam is universally adopted method to all the cases of multiplications, so it is chosen for BCD multiplication.

B. Multiplication process using Vertical-Cross Method

Fig. 1 is illustrating 2x2 digit multiplication using vertical-cross method.

Step 1: Multiply the numbers in the one's place and put the product directly under the one's.(4 *9)

Step 2: Cross multiply, we would form fractions by taking the top number's tens digit multiplied by the bottom number's ones place. Then take the top number's tens place multiplied by the bottom number's tens place. Add both products. (4*2+1*9)

Step 3: Multiply again, the numbers in the tens place and place the answer to the left of the previous step's answer.(2*1)

Step 4: Add all the partial products with previous step's carry if it is.

Multiplication of 4x4 BCD numbers using Urdhva-tiryakbhyam by applying divides and conquers approach; shown in fig. 2.

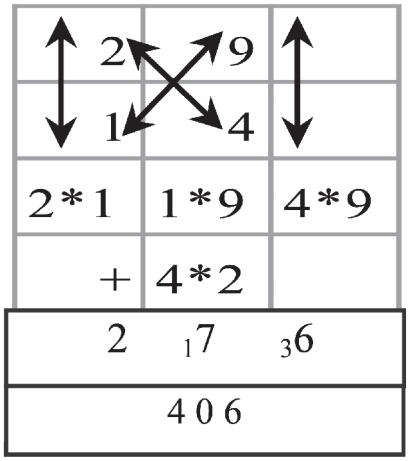

Fig 1: Vertical and cross product

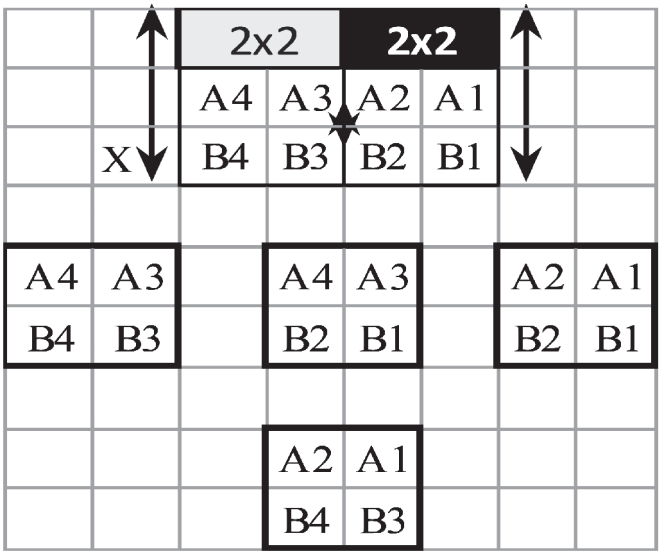

Fig 2: Urdhva-Triyakbhyam and divide and conquers approach

Here A1-A4 and B1-B4 are 4 digit BCD number and multiplication took place using 2x2 digit multiplier blocks. Only four port map statements are required for partial multiplication process.

C. Vedic Binary Multiplier

Each single decimal digit can be represented by a unique 4-bit pattern; they are as follows in table II. If we want to implement single digit multiplier then we have to implement 4x4 binary bit multiplier and it gives us only 7 bit output to express BCD multiplication. A little modification is required in [6]'s architecture between adders of partial products to implement binary multiplier for decimal multiplier as shown in fig. 3

3. BINARY TO BCD CONVERTER

Decimal multiplication in particular has been manipulated in many ways. For example, one way for decimal multiplication is to perform the multiplication directly in decimal. Another approach is to convert the operands to binary, perform the multiplication in binary, and then convert the result back to decimal.

A third approach for decimal multiplication involves performing decimal digit-by-digit multiplication in binary and then converting the resulting binary partial product to decimal [7]. So the binary to BCD conversion is most important part of the decimal multiplication.

Table 2: Decimal Digits Are In 4-bit Binary Pattern

Decimal Digit	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

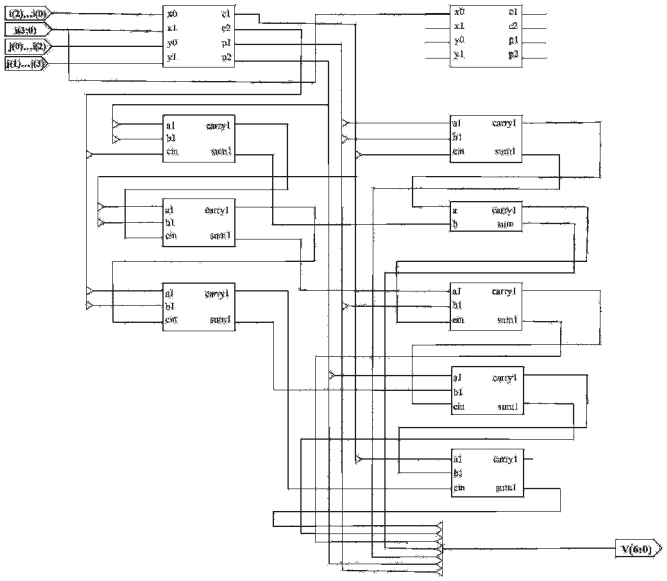

Fig 3: VHDL Implementation of 4-Bit Vedic Binary Multiplier for BCD-8421

A. Literature Survey

Several algorithms had been already proposed by [7-9] for binary to decimal conversion, which improves speed, delay, power consumption and area of chip. [8] Has proposed a highly efficient algorithms, but it gives wrong outputs to some decimal values as shown in table 3 [7] has also proposed an optimized version of [8] architecture by using logical equations and optimized DL block, but it is also giving wrong outputs of the following values (50, 56) as shown in table 3. These errors are occurring because the proposed algorithm isn't correct for some special cases and there are some architecture level faults too. In the next section we are giving explanation of the above errors and give proper improvement in proposed algorithm and architecture of [7-8].

B. Errors And Improvements

There is an error in [8] proposed algorithm as shown in fig. 4—when we give 56 as an input, step-2 gives carry2=1 and final LSB is became 0000 so no correction took place but actual number through second addition was "10000" (16) and it is greater than 9. So a correction has to be implemented in this algorithm—if (no>9 or carry2==1) then add 6. There is also an architecture level fault in [6] shown in fig. 5. when we are giving 46 as an input then carry c1=1 and HSB t3t2t1t0 is 0011 and according to this architecture 2-bit one adder from HSB side can't give correct output due to the carry generated by this adder isn't added into next t3 bit. A logical OR gate is require to resolve this problem. In [7] proposed DL optimizer and logical equations in four-three algorithm are giving wrong output, when we give (56, 50) as an input. Reason is same carry2=1 and final LSB is "0000", as show in fig. 4. So, these errors from carry generation equations and in the resulting block named "optimized correction block" are due to incorrect conversion algorithm. So these little modifications are required to eliminate above errors and it will give proper output in all cases of BCD number system. Implementation of this modified algorithm is shown in fig. 6

Table 3: The Illustration Of Some Wrong Outputs Of [7,8]'s Algorithms And Architectures

Decimal Number	Binary	from [8] algo		Correct		Remark
		D_H	D_L	D_H	D_L	
56	"0111000"	"0101"	"0000"	"0101"	"0110"	Carry problem
57	"0111001"	"0101"	"0001"	"0101"	"0111"	
Decimal Number	Binary	from [8]'s Architecture		Correct		Remark
		D_H	D_L	D_H	D_L	2-bit Adder problem
46	"0101110"	"0000"	"0110"	"0100"	"0110"	
Decimal Number	Binary	from [7]'s 4-3algo's logical eq.		Correct		Remark
		D_H	D_L	D_H	D_L	logical equations and D_L optimizer problem
56	"0111000"	"0101"	"0000"	"0101"	"0110"	
50	"0110010"	"0100"	"1010"	"0101"	"0000"	

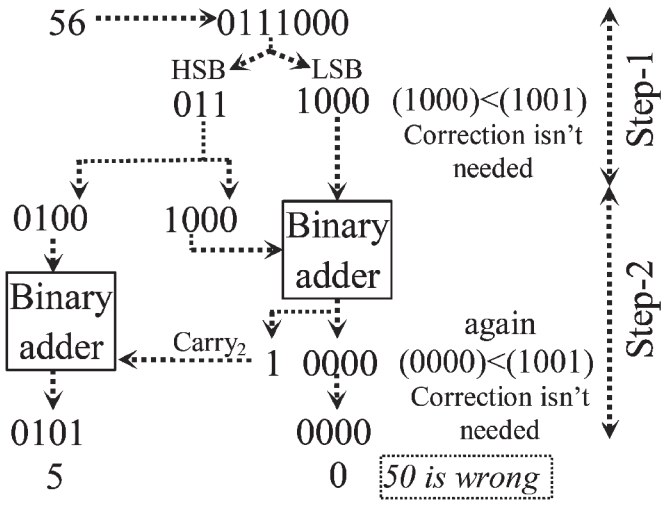

Fig 4: Wrong output from [7,8]'s proposed architecture

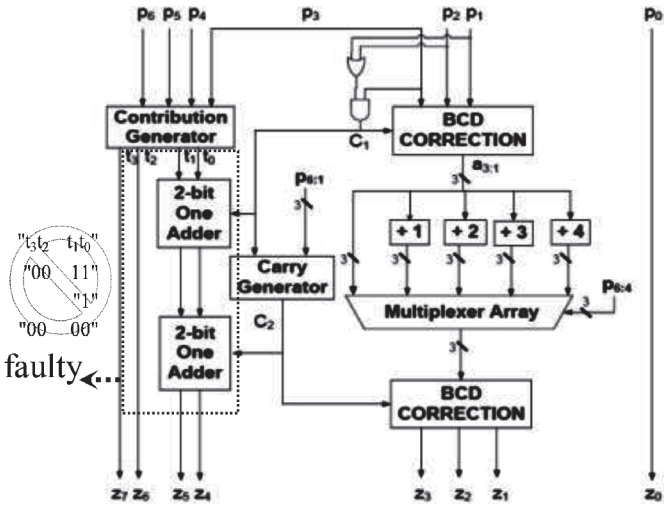

Fig 5: Wrong output from [8]'s proposed architecture

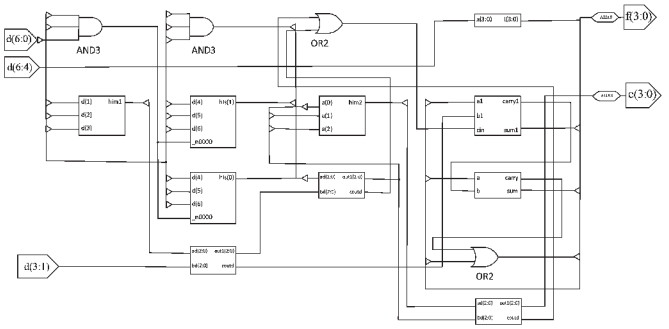

Fig 6: Corrected Implementation (VHDL) of 7-bit binary to BCD converter

4. PROPOSED VEDIC BCD MULTIPLIER

There are several ways to implement decimal multiplier: one way is to perform the multiplication directly in decimal, another approach is to convert the operands into binary; then perform the binary multiplication and lastly convert the result back to

decimal. A third approach to decimal multiplication involves performing decimal digit-by-digit multiplication in binary and then converting the resulting binary partial product to decimal. These partial products are added as appropriate to form the final decimal product [9]. To reduce the circuitry overhead and easier implementation; we are using modified third approach, in which N decimal digits (N is even) are divided into equal parts; then apply Vedic urdhva-triyakbhyam multiplication. After this, decimal adders are used to get the final output digits; as shown in fig. 7, 8. List of BCD adders for implementation purpose are 4by4, 4by2, 4by3 and 2by2 etc. which reduce slices and speed up overall multiplication process. Architecture of 2x2 and 4x4 BCD digits multiplication is shown in fig. 9, 10. These architectures are depicting that if we want to design 2x2 digits multiplier we can use 1x1 multipliers and the Vedic multiplication process is same as mentioned in the section 1. Its implementation is shown in fig. 11. The 4x4 digits multiplier can be implemented easily by using 2x2 digits multipliers, in this way NxN can also be implemented. In the next section results will depict that how these proposed Vedic BCD multiplier are much better than previously implemented multiplier in term of speed and hardware resources.

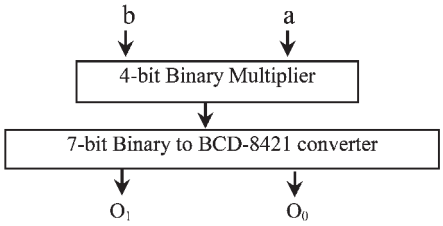

Fig 7: Block diagram of 1x1 BCD Digit Multiplier

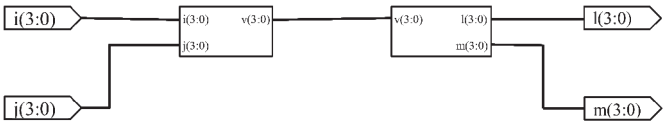

Fig 8: VHDL implementation of 1x1 BCD Digit Multiplier

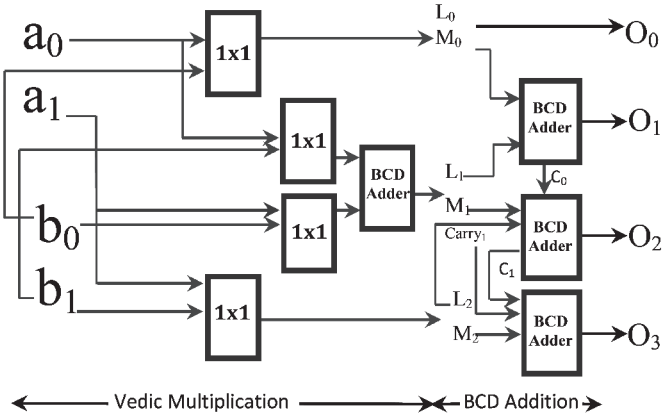

Fig 9: Proposed Architecture of 2x2 BCD Digits Multiplier

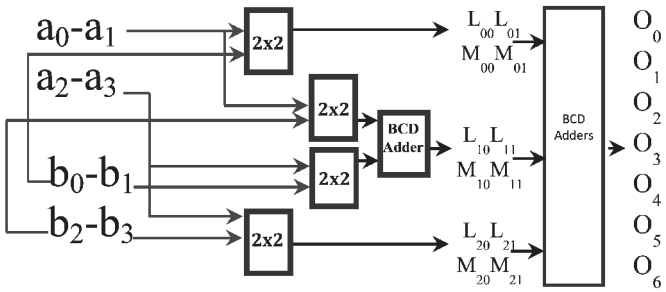

Fig 10: Proposed Architecture of 4x4 BCD Digits Multiplier

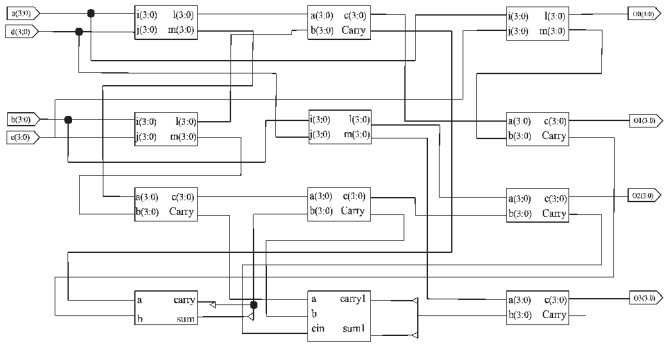

Fig 11: VHDL implementation of 2x2 BCD digits Multiplier

5. SYNTHESIZED RESULTS

All the proposed optimized architectures are successfully implemented in VHDL as shown in fig. 3, 6, 8, 11 and synthesized using Xillinx-ISE8.1 for virtex-4 xc4vfx12 device. Simulation is done using modelSim5.7. Hence the analyzed results are compared with previously implemented architectures for Virtex-4 family, as shown in table IV.

A. Discussion

Fig. 12, 13 are clearly depicting that Vedic 2x2 BCD digits multiplier is at least 2.37 times faster (less delay) and occupies 4 times less hardware slices over previously proposed architectures in [10-12].

Only 7-bit binary to decimal converter is used for 2x2 digits multiplication. After removing all the previously proposed architecture level corrections, we further improve its performance by using proposed three-three split algorithm.

We have also implemented a modified version of [6] for BCD multiplication so as to get an output of 8 bits rather than 7 and it requires 2 adders less.

Finally, we can conclude that our proposed Vedic BCD multiplier is better in terms of area in slices and time of delay over the most efficient in this field of architecture.

1In this paper 1x1-digit is equitant to 4x4 BCD as used in [10-12].

Table 4: Comparison Between Vedic Bcd Multiplier And Previously Proposed Multipliers In Terms Of Delay

Types of Multiplier	Proposed	Delay
1x1-digit	Vedic BCD multiplier	15.621ns
	[11] sutter	34ns
2x2-digits	Vedic BCD multiplier	29.412
	[12] Vestias	58
	[11] Sutter	68
	[10]Erld	77

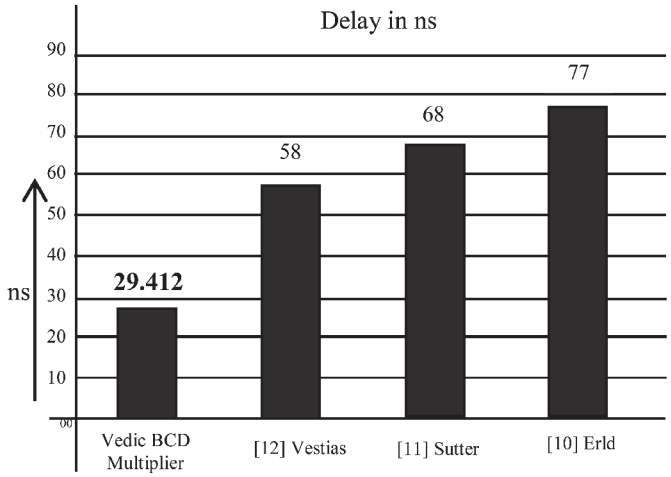

Fig 12: Comparison of 2x2 BCD digits multipliers in term of delay

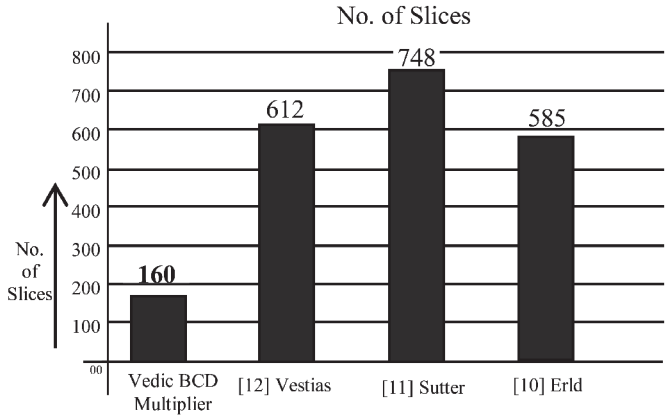

Fig 13: Comparison of 2x2 BCD digits multipliers in term of slices

6. ACKNOWLEDGMENT

We would like to thank all the faculty members of Computer Science department for their invaluable guidance during this research.

7. CONCLUSION AND FUTURE WORK

In this paper we have compared conventional BCD multiplier to proposed Vedic BCD multiplication and the synthesized results of Vedic 2x2 BCD multiplier depicts its efficiency in terms of slices and time of delay

Now, this paper will boost the on-going research of the scholars in the field of decimal arithmetic. A highly efficient binary to BCD conversion algorithm is required for further improvement in this Vedic BCD Multiplier.

REFERENCES

[1] Alvaro Vázquez and Paolo Montuschi, "A New Family of High–Performance Parallel Decimal Multipliers," 18th IEEE Symp. Computer Arithmetic, pp. 195 – 204, June 2007.

[2] Rekha K. James and Sreelasasi, "Decimal Multiplication using compact BCD Multiplier," International Conf. Electronic Design, pp. 1-3, Dec. 2008.

[3] Mario Vestias and HoracioNeto, "Parallel Decimal Multipliers and Squarers using Karatsuba-Ofman's Algorithm," 15th Euromicro Conf. Digital System Design, pp. 782 – 788, Sept. 2012.

[4] M. F. Cowlishaw, "Decimal floating-point: Algorism for computers," Proc. 6th IEEE International Symp. Computer Arithmetic, pp. 104–111, Jun. 2003.

[5] Prabir Saha, Arindam Banerjee, Anup Dandapat, and Partha Bhattacharyya "Design of High Speed Vedic Multiplier for Decimal Number System," Springer-Verlag Berlin Heidelberg, pp. 79–88, 2012.

[6] V Jayaprakasan, S Vijayakumar and V S KanchanaBhaaskaran, "Evaluation of the Conventional vs. Ancient Computation methodology for Energy Efficient Arithmetic Architecture," International Conf. Automation, Control and Computing, pp. 20-22 July 2011.

[7] Osama Al-Khaleel, Zakaria Al-Qudah and Mohammad Al-Khaleel, "Fast and compact binary-to-BCD conversion circuits for decimal multiplication," 29th International Conf. on Computer Design, pp. 226 – 231, Oct. 2011.

[8] Jairaj Bhattacharya, Aman Gupta and Anshul Singh, "A High Performance Binary to BCD Converter for Decimal Multiplication," International Symp. VLSI Design Automation and Test, pp. 315-318, April 2010.

[9] Liu Han and Seok-Bum, "High-Speed Parallel Decimal Multiplication with Redundant Internal Encodings," IEEE Trans. Computers, Vol. 62, no. 5, pp. 956 – 968, May 2013.

[10] M. A. Erle and M. J. Schulte, "Decimal multiplication via carry-save addition," Proc. 14th IEEE International Conf. Application Specific Systems, pp. 348–358, June 2003.

[11] G. Sutter, E. Todoro vich, G. Bioul, M. Vazquez, and J.-P. Deschamps, "FPGA implementations of BCD multipliers," Proc. IEEE International Conf. Reconfigurable Computing and FPGAs, pp. 36–41, Dec. 2009.

[12] Vestias, M.P, Neto and H.C. "Iterative Decimal Multiplication Using Binary Arithmetic," 7th Southern Conference Programmable Logic, pp. 257 – 262, April 2011.

◆ ◆ ◆