An Extensive Literature Review on Radix-10 Multiplication

Vikram Singh Yadav[1] & Prof. Suresh Gawande[2]

[1]M-Tech Research, [2]Research Guide and HOD

Department of Electronics and Communication Engineering, Bhabha Research Engineering Institute, Bhopal

Abstract- Multiplication is the basic operation in any signal processing systems and financial applications, all these applications requires multiplication to be performed in a faster and efficient manner on a silicon chip. This review paper describes the algorithm and architecture of a BCD multiplier. In the literature review study the architecture of a BCD multiplier that exploits some properties of two different redundant BCD codes to speed up its computation. In this paper, we also study techniques to reduce the latency and area of previous representative high performance implementations. The Partial products are generated in parallel using a signed-digit radix-10 recoding of BCD multiplier with the digit set [-5, 5], and set of positive multiplicand multiples (0X, 1X, 2X, 3X, 4X, 5X) coded in excess-3 code(XS-3). This encoding may have many advantages. The available redundancy allows a fast and simple generation of multiplicand multiples in a carry free way.

Keywords: Parallel multiplication, BCD representation, redundant arithmetic.

I. INTRODUCTION

Multiplication of decimal numbers plays a vital role in many user-oriented applications like finance and commercial and where rounding and conversion of numbers those inherent to binary representations in floating-point cannot be tolerated4. This is the cause for decimal operations to become more popular in the recent years. The existence of various microprocessors like Fujitsu Sparc X and IBM power microprocessor 6,9 families that are oriented to mainframes and servers include fully IEEE 754-2008 Decimal Floating Point Units (DFPUs) for 16-digit decimal and 32-digit decimal formats. Also, the standard IEEE 754-2008 for Floating-Point Arithmetic which includes specifications and format for decimal multiplication have created a path for decimal hardware 7,10,11 in a significant manner. Again, division and multiplications are performed by using digit by digit algorithm in iterative manner because of which it adds to low performance. The use of aggressive cycle time in Processors will employ additional constraints on parallel techniques in order to reduce the latency employed by parallel design. Thus, an efficient algorithm is required to develop a DFPU to perform multiplication and to have a most regular VLSI layouts which allows pipelining effectively.

II. PARTIAL PRODUCT REDUCTION

The partial product arrays generated by the SD radix-10 encoding each column of p digits is reduced to two digits by means of a decimal digit CSA tree shown in Fig. 1. The decimal carries are passed between adjacent digit columns and decimal coding method used for decimal carry-save addition.

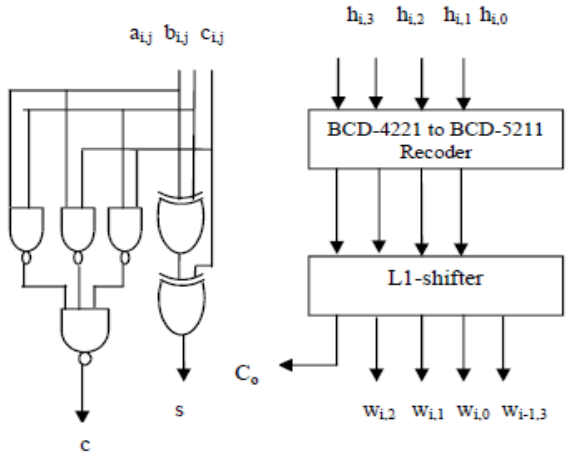

Fig. 1. Scheme of x2 for BCD-4221

The algorithm and architecture of a BCD multiplier that exploits some properties of two different redundant BCD codes to speed up its computations are redundant BCD excess-3 code (XS-3) and the overloaded BCD representation (ODDS). Proposed techniques are developed to reduce significantly the latency and area of previous representative high performance implementations.

Partial Product Generation

Partial products are generated in parallel using a signed-digit radix-10 recoding of the BCD multiplier with the digit set [-5,5] and a set of positive multiplicand multiples (0X, 1X, 2X, 3X, 4X, 5X) coded in XS-3 encoding has several advantages. First, it is a self-complementing code the negative multiplicand multiple can be obtained by just inverting the bits of the corresponding positive one. The available redundancy allows a fast and simple generation of multiplicand multiples in a carry-free way. Finally, the partial products can be recoded to the ODDS representation by just adding a constant factor into the partial product reduction tree.

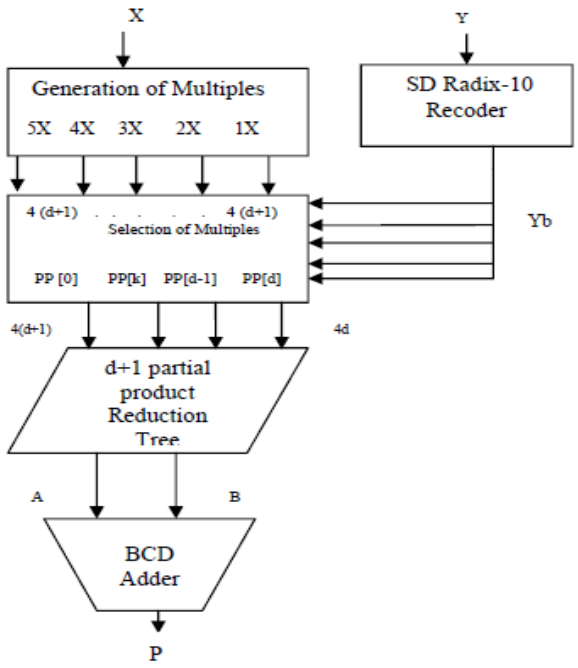

Fig. 2 Combinational SD Radix-10 Architecture

The ODDS uses a similar 4-bit binary encoding as non-redundant BCD techniques explains binary carry-save adders and compressor trees, can be adapted efficiently to perform decimal operations. A variety of redundant decimal formats and arithmetic have been proposed to improve the performance of BCD multiplication. The BCD carry-save format represents a radix-10 operand using a BCD digit and a carry bit at each decimal multiplication area and power dissipation are critical design factors in DFPU. Multiplication and division are performed iteratively by means of digit-by-digit algorithms for reducing the latency of DFP multiplication in high performance DFPUs.

III. LITERATURE REVIEW

Vazquez, A.; Antelo, E.; Bruguera, J.D.,[1] presented the algorithm and architecture of a BCD multiplier that exploits some properties of two different redundant BCD codes to speed up its computation: the redundant BCD excess-3 code (XS-3), and the overloaded BCD representation (ODDS). In addition, new techniques have been developed to reduce significantly the latency and area of previous representative high-performance implementations. Partial products are generated in parallel using a signed-digit radix-10 recoding of the BCD multiplier with the digit set [-5, 5], and a set of positive multiplicand multiples (0X, 1X, 2X, 3X, 4X, 5X) coded in XS-3. This encoding has several advantages. First, it is a self-complementing code, so that a negative multiplicand multiple can be obtained by just inverting the bits of the corresponding positive one. Also, the available redundancy allows a fast and simple generation of multiplicand multiples in a carry-free way. Finally, the partial products can be recoded to the ODDS representation by just adding a constant factor into the partial product reduction tree. They show that the proposed decimal multiplier has an area improvement roughly in the range 20-35 percent for similar target delays with respect to the fastest implementation.

Wang, Zhuo; Han, Liu; Seok-Bum Ko, [2] The study of presents a Radix-100 divider based on decimal non-restoring and selection by truncation method. Two decimal quotient digits can be selected in each iteration, which can reduce half of the iteration cycles. Initialization is required to scale the divisor into a pre-calculated range, and also used for generating some multiples of the scaled divisor. Implemented with STM 90-nm standard cells library, the proposed architecture takes 14 clock cycles, which is 373 FO4 to reach the desired accuracy. The latency is much shorter than Radix-10 dividers.

Minchola Guardia, C.E.,[3] proposed to the Decimal multiplication is one of the most frequently used operations in financial, scientific, commercial and internet-based applications. The study of presents an efficient implementation of a fully pipelined decimal multiplier designed with Carry Save Addition and coded into a reduced group of BCD. This design is based on multiplier operands recoded in Signed-Digit radix-10, a simplified partial products generator, and decimal adders. A variety of multipliers architectures are processed on a Virtex-6 FPGA device. Several assessments are carried out in various N by M multiplications and their respective synthesis results show

slightly optimistic figures in terms of area and delay in regard to some previously published works.

Kornerup, P.; Lefevre, V.; Louvet, N.; Muller, J.-M., [4] The study of presents a study of some basic blocks needed in the design of floating-point summation algorithms. In particular, in radix-2 floating-point arithmetic, they show that among the set of the algorithms with no comparisons performing only floating-point additions/subtractions, the 2Sum algorithm introduced by Knuth is minimal, both in terms of number of operations and depth of the dependency graph. They investigate the possible use of another algorithm, Dekker's Fast2Sum algorithm, in radix-10 arithmetic. They give methods for computing, in radix 10, the floating-point number nearest the average value of two floating-point numbers. They also prove that under reasonable conditions, an algorithm performing only round-to-nearest additions/subtractions cannot compute the round-to-nearest sum of at least three floating-point numbers. Starting from an algorithm due to Boldo and Melquiond, they also present new results about the computation of the correctly-rounded sum of three floating-point numbers. For a few of their algorithms, they assume new operations defined by the recent IEEE 754-2008 Standard are available.

Lang, T.; Nannarelli, A.,[5] presented original article see Kaivani, et al., ibid, vol. 5, pp. 393-404 (2011). Lang and Nannarelli comment on the paper of Kaivani, et al., which reported a proposed unit 46% faster than the unit from their study. Lang and Nannarelli show in this comment that the evaluation done by Kaivani, et al. is based on wrong assumptions and the results of the comparison are erroneous.

Ercegovac, M.D.; McIlhenny, R.,[6] They present a shared implementation of a radix-10 and radix-16 fixed-point digit-recurrence algorithm for the square root operation using limited-precision multipliers, adders, and table-lookups. They discuss the proposed algorithm, its design, and its ASIC implementation using a standard cell library. They present the cost and delay characteristics for precisions of 8 (single-precision), 16 (double-precision) for both radix-10 and radix-16 digits. The proposed scheme uses short (2-4 digit-wide) operators which leads to compact modules, reduced interconnections, reduced area and increased delay with respect to non-shared implementations.

Emami, S.; Dorrigiv, M.; Jaberipur, G.,[7] in presented the support for decimal computer arithmetic is growing to meet the increasing user demands in many computer applications such that in the past decade some commercialized processors have been equipped with decimal hardware units and the latest IEEE standard for floating point arithmetic. Particular, the 10-bit densely packed encoding for compact storage of three decimal digits has been defined, which require pre and post conversions to make arithmetic operations and proper storage possible. In the study of, they offer the 10-bit radix-1000 encoding of three decimal digits that can be directly processed by decimal arithmetic operators. The 16-digit and 34-digit BCD operands are converted to 54-bit and 114-bit chiliad operands, respectively. Following the practice of using word-wide binary adders for decimal operands with some off-the-critical-path correction logic, they device an adder architecture for intermediate chiliad operands. The same adders can be shared by the binary floating point units with the IEEE-754-2008 53-bit and 113-bit significant. The synthesis results show that the proposed scheme is more area and potheyr efficient than the best previous method.

Ercegovac, M.D.; McIlhenny, R.,[8] implemented of a radix-10 and radix-16 fixed-point digit-recurrence algorithm for division operation using limited-precision multipliers, adders, and table-lookups. Authors discuss the proposed algorithm, its design, and its ASIC implementation using a standard cell library. They present the cost and delay characteristics for precisions of 7 (single-precision), 14 (double-precision) decimal digits, and single and double precision for radix-16. The proposed scheme uses short (2-3 digit-wide) operators which leads to compact modules, reduced interconnections and has an advantage at the layout level optimization.

IV. PROBLEM IDENTIFICATION

The performance of algorithm of a BCD multiplier that exploits some properties of two different redundant BCD codes to speed up its computation: the redundant BCD excess-3 code (XS-3), and the overloaded BCD representation (ODDS). In addition, new techniques may be developed to reduce significantly the latency and area of previous representative high performance implementations. Partial products are generated in parallel using a signed-digit radix-10 recoding of the BCD multiplier with the digit set [-5, 5], and a set of positive multiplicand multiples (0X, 1X, 2X, 3X, 4X, 5X) coded in XS-3. Also, the available redundancy allows a fast and simple generation of multiplicand multiples in a carry free way.

V. CONCLUSIONS AND FUTURE SCOPE

We have analyzed the algorithm and architecture of a new BCD multiplier. The further improvements of the architecture rely may be designed on the use of certain

redundant BCD codes. Partial products would be generated very fast in the XS-3 representation using the SD radix-10 PPG scheme: positive multiplicand multiples (0X, 1X, 2X, 3X, 4X, 5X) are pre computed in a carry-free way, while negative multiples are studied by bit inversion of the positive ones. The high speed and area efficient decimal multiplier may be designed for improving the system performance.

REFERENCES

[1] Vazquez, A.; Antelo, E.; Bruguera, J.D., "Fast Radix-10 Multiplication Using Redundant BCD Codes," in *Computers, IEEE Transactions on* , vol.63, no.8, pp.1902-1914, Aug. 1 2014.

[2] Wang, Zhuo; Han, Liu; Seok-Bum Ko, "Design and implementation of a Radix-100 division unit," in *Circuits and Systems (ISCAS), 2012 IEEE International Symposium on* , vol., no., pp.1239-1242, 20-23 May 2012.

[3] Minchola Guardia, C.E., "Implementation of a fully pipelined BCD multiplier in FPGA," in *Programmable Logic (SPL), 2012 VIII Southern Conference on* , vol., no., pp.1-6, 20-23 March 2012.

[4] Kornerup, P.; Lefevre, V.; Louvet, N.; Muller, J.-M., "On the Computation of Correctly Rounded Sums," in *Computers, IEEE Transactions on* , vol.61, no.3, pp.289-298, March 2012.

[5] Lang, T.; Nannarelli, A., "Comments on 'improving the speed of decimal division'," in *Computers & Digital Techniques, IET* , vol.6, no.6, pp.370-371, November 2012.

[6] Ercegovac, M.D.; McIlhenny, R., "Shared implementation of radix-10 and radix-16 square root algorithm with limited precision primitives," in *Signals, Systems and Computers (ASILOMAR), 2012 Conference Record of the Forty Sixth Asilomar Conference on* , vol., no., pp.345-349, 4-7 Nov. 2012.

[7] Emami, S.; Dorrigiv, M.; Jaberipur, G., "Radix-10 addition with radix-1000 encoding of decimal operands," in *Computer Architecture and Digital Systems (CADS), 2012 16th CSI International Symposium on* , vol., no., pp.139-144, 2-3 May 2012.

[8] Ercegovac, M.D.; McIlhenny, R., "Shared implementation of radix-10 and radix-16 division algorithm with limited precision primitives," in *Signals, Systems and Computers (ASILOMAR), 2011 Conference Record of the Forty Fifth Asilomar Conference on* , vol., no., pp.1828-1832, 6-9 Nov. 2011.

[9] Khan, K.M.N.H.; Ali, M.L.; Islam, S., "A new technique for high speed decimal logarithm computation of decimal floating-point number," in *Computer and Information Technology (ICCIT), 2011 14th International Conference on* , vol., no., pp.208-212, 22-24 Dec. 2011.

[10] Baesler, M.; Voigt, S.; Teufel, T., "FPGA Implementations of Radix-10 Digit Recurrence Fixed-Point and Floating-Point Dividers," in *Reconfigurable Computing and FPGAs (ReConFig), 2011 International Conference on* , vol., no., pp.13-19, Nov. 30 2011-Dec. 2 2011.

[11] Faraday Tech. Corp. (2004). 90nm UMC L90 standard performance low-K library (RVT). [Online]. Available: *http://freelibrary.faraday-* tech.com/

[12] S. Gorgin and G. Jaberipur, "A fully redundant decimal adder and its application in parallel decimal multipliers," Microelectron. J., vol. 40, no. 10, pp. 1471–1481, Oct. 2009.

[13] S. Gorgin and G. Jaberipur. (2013, May). "High speed parallel decimal multiplication with redundant internal encodings," IEEE Trans. Comput. vol. 62, no. 5, [Online]. Available: *http://doi.ieeecomputersociety.* org/10.1109/TC.2013.160

[14] L. Han and S. Ko, "High speed parallel decimal multiplication with redundant internal encodings," IEEE Trans. Comput., vol. 62, no. 5, pp. 956–968, May 2013.

[15] IEEE Standard for Floating-Point Arithmetic, IEEE Std 754(TM)-2008 IEEE Comput. Soc., Aug. 2008.