International Journal of advancement in electronics and computer engineering (IJAECE)
Volume 4, Issue 10, Feb. 2016, pp.614-618, ISSN 2278 -1412

A Decimal / Binary Multi-operand Adder using a Fast Binary to Decimal Converter-A Review

Ruchi Bhatt, Divyanshu Rao, Ravi Mohan

[1] *M. Tech Scholar, Department of Electronics & Communication Engineering, SRIT, Jabalpur (M.P.) ruchi.bhatt02@gmail.com, India;*

[2] *Assistant Professor, Department of Electronics & Communication Engineering, SRIT, Jabalpur (M.P.) divyanshu3@gmail.com, India;*

[3] *PG course In charge, Department of Electronics & Communication Engineering, SRIT, Jabalpur(M.P.) Ravimohan7677@yahoo.co.in,India;*

Abstract – Decimal arithmetic has obtained considerable attention presently due to its availability for many financial and commercial applications, where precision is very important. Binary digits have a disadvantage of not being able to represent digits like 0.1 or 0.7, needs an infinitely re-occurring binary number. The availability of multi-operand decimal adders can ease of financial and commercial implementations depend on existing huge databases. The contemporaneous addition of many decimal numbers is the general operation in multiplication and division algorithms. Multi-operand addition is an important operation as it is a core element of arithmetic operations, such as division and multiplication. In case of decimal multiplication Multi-operand decimal addition comes in handy for swiftly summing large amounts of decimal data. This project introduces a multi-operand decimal addition algorithm by employing high speed binary to BCD converter circuit, which fastness the procedure of decimal sum when numerous BCD operands are added together. A Novel design for 7-bit binary to BCD converter circuit is proposed.

Keywords*: BCD, Decimal / Binary Multi-operand Adder, Multi Operand Adder, FPGA*

I. Introduction

For years, the use of dedicated decimal hardware has been limited to mainframe business computers and handheld calculators at the low end. Very recently a renewed interest in providing hardware advancement for decimal arithmetic has transpired. It has been boosted by the numerically intensive computing requirements of new commercial, financial and Internet applications, such as e-commerce and e-banking. Because of the perspectives of a additional worldwide utilization of decimal processing, the revised IEEE 754-2008 Standard for Floating-Point incorporates a specification for decimal arithmetic.

The design of decimal units for FPGAs faces several challenges: first, the inherent inefficiency of decimal representations in systems depend on two-state logic and a complex mapping of the decimal arithmetic regulations into Boolean logic. For example, the binary numbers cannot be used for the representations of some fractions, e.g., 0.310=0.01001.….2, which will needs to be infinite bits for representation. This thing is not applicable for proper decimal fractions, since the incorrect output for the appropriate representation of inputs will lead to subsequent precision errors and thus will degrade the accuracy for the overall computations. To clear the drawback, the binary coded decimal (BCD) numbers is utilized as a general representation of decimal numbers, as BCD can recode each digit of decimal numbers from 0 to 9 using four bits 00002 to 10012, respectively. In the above example of the representation of 0.310, the BCD numbers can only be recorded as 0.0011 (BCD) in finite and exact representations. On the another hand, the special built-in features of FPGA architectures make it complicated to utilize several well-known procedures to fastness computations (for example, carry-save and signed-digit arithmetic's). Therefore, it may be preferable to develop specific decimal algorithms additional applicable for FPGAs comparatively than adapting existing ones targeted for ASIC platforms. In this context, we present the algorithm, architecture and FPGA execution of a handwork unit to operate fast sum of a large amount of decimal (BCD) fixed-point or integer operands. This operator is also of key importance for other arithmetic operations such as decimal multiplication and division. This project proposes a multi-operand decimal addition algorithm by implementing high speed binary to BCD converter circuit, which quick up the procedure of decimal addition when numerous BCD operands are added together. A Novel representation for 7-bit binary to BCD converter circuit is to be introduced. Later on, review is completed accompanied by respect to the existing binary to BCD converter architectures. The introduced algorithm is basically varies from multi-operand BCD addition

algorithms since in-between BCD corrections are not completed comparatively correction is completed at the resultant step to obtain up to mark BCD results. As the decimal corrections are obtained separately from the computation of the binary addition, such that the structure of the binary carry-save adder does not need to be any further re-arrangement, the representation can be operated as consolidate Binary/ BCD multi-operand adder.

II. Literature Survey

The attentive analysis of the associated work and published literature, it is found that several researches from tars have the sum of two n-digit BCD numbers that follows the similar technique. Therefore, to enhance the BCD adders speed, designers have to introduce various enhancements to the fundamental BCD addition algorithm .Direct decimal addition, decimal speculative addition and conditional speculative decimal addition, are examples of such types of refinements.

Rekha k. james et.al.[1] in this author proposed work on the Decimal Multiplication using compact BCD Multiplier. In IEEE International Conference on Electronic Design. Decimal multiplication is an integral form of financial, commercial and internet-depended computations. The key building block of a decimal multiplier is a single digit multiplier. It receive two Binary Coded Decimal (BCD) inputs and provides a product in the range [0, 81] represented by two BCD digits. A hardback representation for single digit decimal multiplication that decreases the critical path delay and area is introduced in this research. Out of the 256 possible combinations for the 8-bit input, only hundred combinations are reasonable BCD inputs. In the hundred reasonable combinations only four combinations require 4 x 4 multiplication 64, combinations need 3 x3 multiplication, and the remaining 32 combinations utilize either3 x4 or4 x3 multiplication. The introduced representation makes utilize of this property. This representation leads to additional regular VLSI execution, and does not need to be special registers for saving simple multiples.

Under this visualizes the novel design for unique digit decimal multiplication to decrease the critical path delay and area, which permits for a quick multiplier design. The assembly of partial products prepared by utilizing unique digit multipliers is completed by an array of multi-operand BCD adders for an (n-digit n-digit) multiplication. This is an entirely parallel multiplier by utilizing only combinational logic, and can be enlarged for floating point multiplication of decimal digits. Advantage is represented that this design gains a 7% savings in the area and 16% savings in delay as comparison to the occurring representation of this design leads to additional regular VLSI execution, and does not need any remarkable registers for saving simple multiples of these.

Alvaro Vazquez, et. al. [2] in this proposed work on the Improved Design of High-Performance Parallel Decimal Multipliers in the ieee transactions on computers, vol. 59, no. 5. The recent generation of high-performance decimal floating-point units (DFUs) is demanding efficient executions of parallel decimal multipliers, He explains the structures of two parallel decimal have demonstrated various procedures to execute decimal parallel multiplication in hardware. He introduce two different SD encodings for the multiplier that lead to quick parallel and normal generation of partial products He have developed a decimal carry-save algorithm depend on unconventional (4221) and (5211) decimal encodings for partial product reduction. It makes possible the construction of p:2 decimal CSA trees that outperform the area and detain figures of existing options. He have introduced architectures for decimal SD radix-10 and SD radix-5 parallel multiplication. The area and detain figures from a comparative study containing conventional binary parallel multipliers and other representative decimal offers display that our decimal SD radix-10 multiplier is an interesting choice for high performance with medium area.

Álvaro Vázquez et. al.[3] in this author work on Multi-operand Decimal Adder Trees for FPGAs. The research and development of hardware representations for decimal arithmetic is presently going within an intense action. For most part, the techniques introduced to execute fixed and floating point operations are calculated for ASIC designs. Thus, a direct mapping or adaptation of these procedures into a FPGA could be far from an optimal solution. To improve the efficiency of Virtex-5/6 executions, we have developed a recent algorithm for BCD carry-propagate sum. Combinational and pipelined versions of the BCD multioperand adder were synthesized in a Virtex-6 speed grade-3 device and the results compared with a binary carry-ripple adder tree and a BCD multi-operand adder tree build of BCD carry chain adders .He demonstrate that the introduced representation is a very competitive choice for high-performance low-latency executions of BCD multiplication on Virtex-5/6 FPGAs at a medium hardware cost.

L. Dadda et.al.[4] work on A Parallel-Serial Decimal Multiplier Architecture is published in 2012 IEEE 15th International Conference on Computational Science and Engineering Derived from a parallel multiplier, a parallel-serial decimal multiplier is introduced in which the multiplicand is imagine in parallel while the multiplier is in digit-serial arrangement. A pattern for a parallel-serial decimal multiplier is presented, by utilizing BCD digits. The multiplicand is imagine in parallel, the multiplier in digit-serial format. The values of the Digit Products in the successive columns of the product array are added in binary and converted in decimal. Their decimal alignment produces a combination of three or four serial decimal numbers whose addition is the product. The product digits are obtained via a serial multi-operand adder. The

assessment of the Columns requires the greatest portion of the total area. Its arrangement is naturally pipelined, permitting a elevated processing speed. The condition of multipliers in which two successive multiplications are overlapped in time has also been employed. On trying to examine recent efficient and less resource demanding results, He is also investigating a latest version of the multiplier, which will try to take advantages by the resources suitable on the FPGAs.

Osama D. Al-Khaleeli et. al. [5] in this proposed method FPGA implementation of Binary Coded Decimal Digit Adders and Multipliers published a paper in 2012 IEEE. Decimal arithmetic has achieved elevated impact on the entire performance of today's financial and commercial applications. Decimal additions and multiplication are the main decimal operations utilized in any decimal arithmetic algorithm Decimal digit adders and decimal digit multipliers are generally the building blocks for higher order of decimal adders and multipliers.

FPGAs offer an efficient hardware platform that can be applicable for increasing decimal algorithms. In this paper, various representations for two decimal digit adders and one decimal digit multiplier are introduced. In it two new BCD digit adders and one new BCD digit multiplier are created for the intention of speeding up decimal arithmetic applications over FPGAs. Each design is explained, verified and tested and optimized for a correct functionality by utilizing VHDL coding and simulation. The various designs are executed by utilizing Xilinx ISE10.1 Up to these all worked up to simulations we are postpone it up to the behavior on the board.

III. Method

Generic techniques for binary-to-decimal conversion had been developed long ago. When used in the context of decimal multiplication, these techniques miss the opportunity of optimizations that utilize the particularities of decimal addition.

Numerous research has been already done in the area of decimal addition.. Fully combinational (parallel) decimal addition has been explored in various ways to support applications that require high speed multiplication. For example, an algorithm by which BCD operands are first converted to binary. The operands are then added in binary and then the final result is converted to BCD. The advantage of using such a scheme is that it utilizes the binary adders already available in configurable hardware. In literature, a novel algorithm is proposed to convert a 7-bit binary partial product to 2-digit BCD partial product (denoted as D_H and D_L for most significant and least significant BCD digits respectively). The algorithm splits the 7-bit binary partial product into two parts: the three most significant bits and the four least significant bits. Table I lists some examples to show the contribution of the three most significant bits to D_H and D_L. The algorithm then proceeds as follows. The four least significant bits are first BCD-corrected if needed (by adding (0110)2). This process results into a BCD digit and a potential carry. If there is a carry, it will be added to the sum that produces (D_H). The resulting BCD digit is added to the contribution of the most significant bits to D_L and the result is corrected again. The outcome of this operation is the least significant BCD digit (D_L) and a potential carry which is added again to the circuit that computes D_H. Therefore, D_H is computed as the contribution of the three most significant bits to D_H plus the two potential carries resulting from the two BCD correction operations performed to compute D_L.

Later work improves this algorithm by computing not only the contribution of the three most significant bits to D_H and D_L, but also the contribution of the four least significant bits to D_H and D_L. For both, D_H and D_L, the contributions of the two bit groups are added. In addition, proper optimizations are applied to increase the performance of our proposed architecture. Furthermore, another algorithm is also proposed in which we split the 7-bit binary partial product into the four most significant bits and the three least significant bits. Using the same principle, we show that this algorithm performs better due to the fact that the three least significant bits make no contribution to D_H. Furthermore, the contribution of the three least significant bits to D_L is the three bits themselves. Therefore, no circuit is needed to compute this contribution. The Three-Four split algorithm is illustrated in Figure 1 where the 7-bit binary number is split into two groups of bits: the three most significant bits and the four least significant bits. We compute the contribution of the three most significant bits to each of the two BCD digits (D_L and D_H) similar to what has been done in [2] as discussed in Section II. In addition, the contribution of the four least significant bits to each of the two BCD digits (D_L and D_H) is computed. Some examples of the Contribution of the four least significant bits to D_L and D_H.

TABLE I Four Least Significant bits to D_L and D_H.

Most three bits	**Contribution to D_H**	**Contribution to D_L**
000	0000	0000
010	0011	0010
101	1000	0000

We observe that the contribution of the four least significant bits to D_H is only a one bit carry (the other three bits are always zero) and $D_L[0] = A_0$. An optimized logic for the $A_3A_2A_1A_0$ Contribution Generator block of Figure 1 is derived based on all possible combinations of the four least significant bits and their contribution to D_H and D_L. Furthermore, the contribution of $A_6A_5A_4$ to D_L has the least significant bit always zero. Therefore, the contribution of these bits to D_L is only three bits with weights of 2, 4, and 8. The optimized logic of the $A_6A_5A_4$ Contribution Generator block of Figure 1 is also

derived based on all possible combinations of the three most significant bits and their contribution to D_H and D_L. To compute D_L, we add the contributions of the two bit groups to form D_L as shown in the D_L Generator Circuit in Figure 2. Similarly, computing D_H is done by adding the contributions of the two bit groups to D_H plus any carry (C in Figures 1 and 2) generated from the D_L Generator circuit. The D_H Generator circuit is shown in Figure 3.

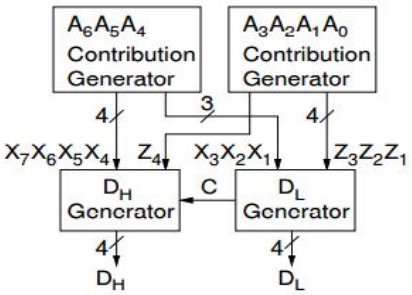

Fig. 1 Three-Four Split 7-bit Binary to 2-digit BCD Converter

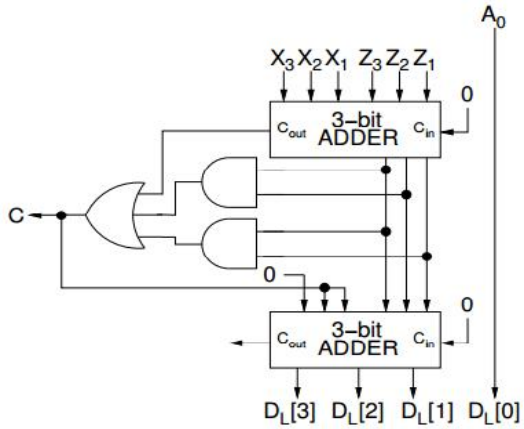

Fig. 2. DL Generator

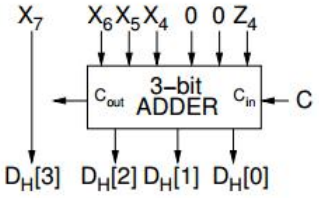

Fig. 3 D_H Generator

To speed up the operation of the DL Generator circuit in Figure 2, we observe the following issues. First, the C_{in} input in the top 3-bit adder is always zero. Furthermore, only a subset of combinations can appear on the inputs of the 3- bit adder. Therefore, we replace this adder with a customized version in which we remove C_{in} and we consider only the possible combinations of $Z_3Z_2Z_1$ and $X_3X_2X_1$. The resulting customized circuit is named "optimized addition stage I" in Figure 4 and it is described by the following logic equations:

$$S_3 = X_3\bar{Z}_3 + X_2Z_1 + X_2Z_2 + \bar{X}_3Z_3$$
$$S_2 = X_2\bar{Z}_2\bar{Z}_1 + \bar{X}_2Z_2 + \bar{X}_2X_1Z_1$$
$$S_1 = \bar{X}_1Z_1 + X_1\bar{Z}_1 \qquad \text{.....(1)}$$

The circuit that computes this contribution to both D_H and D_L is described with the following logic equations:

$$Y_7 = A_6A_4$$
$$Y_6 = A_5A_4\bar{A}_3 + A_6\bar{A}_4\bar{A}_3 + A_5A_3 + A_6A_3$$
$$Y_5 = \bar{A}_5A_4A_3 + A_6\bar{A}_4\bar{A}_3 + A_5\bar{A}_4\bar{A}_3 + A_6A_3$$
$$Y = \bar{A}_6\bar{A}_5A_4\bar{A}_3 + A_5\bar{A}_4\bar{A}_3 + A_5A_4A_3 + A_6A_3$$
$$Y = \bar{A}_6\bar{A}_5\bar{A}_4A_3 + A_5A_4\bar{A}_3$$
$$Y = \bar{A}_6\bar{A}_5A_4\bar{A}_3 + \bar{A}_5A_4A_3 + A_6\bar{A}_4\bar{A}_3 + A_5A_4A_3$$
$$Y = \bar{A}_6\bar{A}_5A_4\bar{A}_3 + A_5\bar{A}_4\bar{A}_3 + A_5A_4A_3 + A_6A_3 \qquad \text{.....(2)}$$

Note that the contribution of the four most significant bits to D_L has the least significant bit always zero. Therefore, these four bits contribute only with three bits with weights of 2, 4, and 8. The circuit that computes the contribution of the D_L Generator circuit is similar to the circuit shown in Figure 2 with inputs $Z_3Z_2Z_1$ replaced with ${}_0A_2A_1$ and $X_3X_2X_1$ replaced with the three bits coming out of the $A_6A_5A_4A_3$ contribution generator ($Y_3Y_2Y_1$). Observations similar to those made in Figure 4 can also be made. For example, now the top 3-bit adder has C_{in} always zero and one of its inputs is always zero (since we replaced $Z_3Z_2Z_1$ with $0A_2A_1$). The optimized D_L generator circuit consists of three blocks similar to the optimized D_L generator of Figure 2. The implementation of these blocks has output logic equations as follows:
Optimized addition stage I:

$$S_3 = Y_3 + Y_2Y_1A_1 + Y_2A_2$$
$$S_2 = Y_2\bar{A}_2A_1 + \bar{Y}_2A_2 + Y_2\bar{Y}_1\bar{A}_2 + \bar{Y}_2Y_1A_1 + Y_1A_2A_1$$
$$S_1 = Y_1\bar{A}_1 + \bar{Y}_1A_1 \qquad \text{.....(3)}$$

Optimized correction**:** same as the optimized correction of the Three-Four split architecture of Figure 1.
Carry generator:

$$C = Y_2Y_1A_2 + Y_3A_1 + Y_3A_2 + Y_2A_2A_1 \qquad \text{.....(4)}$$

The DH Generator block is also similar to that shown in the circuit of Figure 3 with inputs $00Z_4$ replaced with all zeros. Similarly, a customized circuit can be made similar to that. The output logic equations for this circuit are as follows:

$$D_H[0] = Y_4\bar{C} + \bar{Y}_4C$$
$$D_H[1] = CY_4 + Y_5$$
$$D_H[2] = Y_6$$
$$D_H[3] = Y_7 \qquad \text{.....(5)}$$

IV. Conclusion

In this paper review 7-bit Binary to BCD converters and Multi-operand structures. A Novel Unified BCD/ Binary multi-operand addition algorithm has been proposed. The binary parallel multi-operand addition is realized using a CSA tree for compressing the input operands. The proposed BD converter forms the core of the multi-operand decimal adder. In This paper demonstrate the method to optimized efficiency of our proposed BD converter as well as multi-operand decimal adder.

V. References

[1] REKHA K. JAMES, SHAHANA T. K, and K. POULOSE JACOB give International Conference on Electronic Design in 2008

[2] Alvaro Vazquez, Elisardo Antelo is worked on the Improved Design of High-Performance Parallel Decimal Multipliers in the ieee transactions on computers, vol. 59, no. 5, may 2010

[3] Álvaro Vázquez, Florent de Dinechin give Efficient implementation of Parallel BCD Multiplication in LUT-6 FPGAs in 2010

[4] Luigi. Dadda, "Multi-operand parallel decimal adder: A mixed binary and bcd approach." Computers, IEEE Transactions on 56.10 (2007): 1320-1328.

[5] Al-Khaleel, Osama, et al. "Fast and compact binary-to-BCD conversion circuits for decimal multiplication." Computer Design (ICCD), 2011 IEEE 29th International Conference on. IEEE, 2011.

[6]- Jaberipur, Ghassem, and Amir Kaivani. "Improving the speed of parallel decimal multiplication." Computers, IEEE Transactions on 58.11 (2009): 1539-1552.

[7]- Bhattacharya, Jairaj, Aman Gupta, and Anshul Singh. "A high performance binary to BCD converter for decimal multiplication." VLSI Design Automation and Test (VLSI-DAT), 2010 International Symposium on. IEEE, 2010.

[8]- Lin, Kuan Jen, et al. "A parallel decimal adder with carry correction during binary accumulation." New Circuits and Systems Conference (NEWCAS), 2012 IEEE 10th International. IEEE, 2012.

[9] S. Knowles, ‘‘A family of adders,’’ in: Proceedings of the 14th IEEE Symposium on Computer Arithmetic, pp. 30–34, 1999.

[10] M. F. Cowlishaw. Decimal floating-point: Algorism for computers. In Proc. IEEE 16th Symposium on Computer Arithmetic, pages 104–111, July 2003.

[11] M. D. Ercegovac and T. Lang, Digital Computer Arithmetic. Elsevier/Morgan Kaufmann Publishers,2004.

[12] R. D. Kenney and M. J. Schulte. High-speed multi-operand decimal adders. IEEE Trans. on Computers, 54(8):953–963,Aug. 2005.