IJRRECS/June 2016/Volume-4/Issue-6/5511-5515 **ISSN 2321-5461**

INTERNATIONAL JOURNAL OF REVIEWS ON RECENT ELECTRONICS AND COMPUTER SCIENCE

IMPLEMENTATION OF AN EFFICIENT DESIGN OF NOVEL BINARY CODED DECIMAL PARALLEL MULTIPLIER

S.Prasanna Chary[1], P.Murali Krishna[2]

[1]**M.Tech Student, Dept of ECE, Ganapathi Engineering College, Warangal, T.S, India**

[2]**Associate Professor, Dept of ECE, Ganapathi Engineering College, Warangal, T.S, India**

ABSTRACT:

Several redundant decimal formats as well as arithmetic were proposed to get better performance of Binary coded decimal multiplication. We introduce an algorithm in addition to structural design of a Binary coded decimal parallel multiplier that makes use of some properties of two different redundant Binary coded decimal codes to accelerate its computation. We build up new method to decrease latency as well as area of earlier representative high performance implementations. Our system is self-complementing codes, with the intention that negative multiplicand multiple is obtained by means of just inverting bits of corresponding positive one. The accessible redundancy permits a speedy as well as easy production of multiplicand multiples in carry free way.

Keywords: Binary coded decimal multiplication, Latency, Self-complementing codes, Redundant decimal formats, Multiplicand.

1. INTRODUCTION:

The novel IEEE 754-2008 Standard in support of Floating- Point Arithmetic, that includes a format as well as specification in support of decimal floating-point arithmetic has encouraged important quantity of research within decimal hardware. While area as well as power dissipation are important design factors within modern decimal floating-point units, multiplication and division are performed iteratively by means of digit-by-digit algorithms and therefore they present low performance [1].

Moreover, the aggressive cycle time of these processors puts an additional constraint on the use of parallel techniques for reducing the latency of decimal floating-point multiplication within high-performance decimal floating-point multiplications. Binary coded decimal carry-save format distinguishes a radix-10 operand by means of a Binary coded decimal digit as well as carry bit at each of the decimal position. It is projected for carry-free accumulation of Binary coded decimal partial products by means of rows of Binary coded decimal digit adders arranged within linear or tree-like configurations. Hence effective algorithms for speeding up decimal floating-point multiplication must result in regular very-large-scale integration layouts that permit an aggressive pipelining. Hardware implementations usually make use of decimal floating-point Binary coded decimal rather than binary to control decimal fixed-point operands as well as integer significands of decimal floating-point numbers for simple conversion between machine as well as user representations.

2. METHODOLOGY:

Binary coded decimal encodes a number X within decimal format, by each of the decimal digit represented within a 4-bit binary number system. On the other hand, Binary coded decimal is less resourceful for encoding integers than binary, as codes 10 to 15 are not used. It is projected for carry-free accumulation of Binary coded decimal partial products by means of rows of Binary coded decimal digit adders arranged within linear or tree-like configurations [2][3]. Usage of general redundant Binary coded decimal arithmetic to speed up parallel Binary coded decimal multiplication within two ways: Partial product generation. The functioning of binary coded decimal arithmetic has additional complications than binary, which leads to area as well as delay penalties in ensuing arithmetic units. The resultant very-large-scale integration functioning in present technologies of multi-operand adder trees might consequence in additional irregular layouts to binary carry-save adders as well as compressor trees. The overloaded Binary coded decimal representation was projected to get better decimal multi-operand addition, as well as sequential in addition to parallel decimal multiplications. The ODDS present exciting properties for a speedy as well as resourceful hardware functioning of decimal arithmetic. It is redundant decimal

representation with the intention that it permits carry-free generation of simple as well as complex decimal multiples and addition, as digits are represented within binary number system, digit operations are performed by binary arithmetic, and different from Binary coded decimal, there is no requirement to put into practice additional hardware to correct unacceptable 4-bit combinations. In our work we introduce an algorithm in addition to structural design of a Binary coded decimal parallel multiplier that makes use of some properties of two different redundant Binary coded decimal codes to accelerate its computation. Our system is self-complementing codes, with the intention that negative multiplicand multiple is obtained by means of just inverting bits of corresponding positive one. The partial products are recoded to the overloaded Binary coded decimal representation by means of just adding up of a stable factor into partial product reduction tree.

3. AN OVERVIEW OF PROPOSED SYSTEM:

In our work we spotlight on enhancement of parallel decimal multiplication by means of exploiting redundancy of two decimal representations such as ODDS and redundant Binary coded decimal excess-3 representation, a self-complementing code by means of digit set [4]. We build up new techniques to reduce latency as well as area of earlier representative high performance implementations. We make use of a minimally redundant digit set for recoding of Binary coded decimal multiplier digits, signed-digit radix-10 recoding, specifically recoded signed digits are in set {-5,-4,-3,-2,-1, 0, 1, 2, 3, 4, 5}. For this set, the most important issue is to carry out x3 multiple devoid of long carry-propagation. We propose usage of general redundant Binary coded decimal arithmetic to speed up parallel Binary coded decimal multiplication within two ways: Partial product generation. By means of producing positive multiplicand multiples coded within XS-3 within a carry free form. An advantage of this representation on non-redundant decimal codes is that the entire motivating multiples in support of decimal partial product generation, includes 3X multiple, and is implemented in stable time by an equal delay of concerning three XOR gate levels [5]. Our system is self-complementing codes, with the intention that negative multiplicand multiple is obtained by means of just inverting bits of corresponding

positive one. The obtainable redundancy permits a speedy as well as easy production of multiplicand multiples in carry free way. By means of performing reduction of partial products are coded in ODDS by means of binary carry-save arithmetic. Partial products are recoded from XS-3 representation to ODDS representation by means of adding of a stable factor into partial product reduction tree. The resulting partial product reduction tree is put into practice by means of regular structures of binary carry-save adders or else compressors. The 4-bit binary encoding of ODDS operands permits resourceful mapping of decimal algorithms to binary techniques [6]. By contrast, signed-digit radix-10 as well as BCD carry-save redundant models requires particular radix-10 digit adders.

4. RESULTS

We provide a more detailed assessment of fastest Binary coded decimal BCD 16x16-digit combinational multipliers regarding latency as well as area.

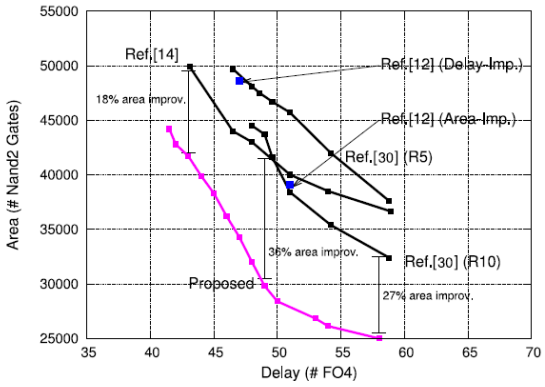

Fig1: Area-delay space for fastest 16x16-digit mults.

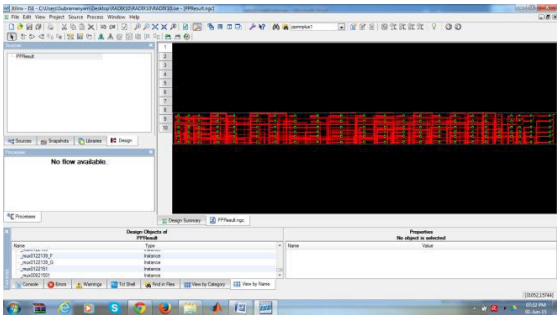

Fig2: Technology Schematic

5. CONCLUSION:

Decimal fixed-point as well as floating-point formats are significant in financial, commercial, as well as user-oriented computing, in which conversion as well as rounding errors that are intrinsic towards floating-point binary representations cannot be accepted. We introduce an algorithm in addition to structural design of a Binary coded decimal parallel multiplier that makes use of some properties of two different redundant Binary coded decimal codes to accelerate its computation. We propose

usage of general redundant Binary coded decimal arithmetic to speed up parallel Binary coded decimal multiplication within two ways: Partial product generation. We make use of a minimally redundant digit set for recoding of Binary coded decimal multiplier digits, signed-digit radix-10 recoding. We build up new techniques to reduce latency as well as area of earlier representative high performance implementations. By producing positive multiplicand multiples coded within XS-3 within a carry free form. An advantage of this representation on non-redundant decimal codes is that the entire motivating multiples in support of decimal partial product generation, includes 3X multiple, and is implemented in stable time by an equal delay of concerning three XOR gate levels. Our system is self-complementing codes, with the intention that negative multiplicand multiple is obtained by means of just inverting bits of corresponding positive one. The obtainable redundancy permits a speedy as well as easy production of multiplicand multiples in carry free way.

REFERENCES

[1] M. F. Cowlishaw, ―Decimal floating-point: Algorism for computers,‖ in Proc. 16th IEEE Symp. Comput. Arithmetic, Jul. 2003,pp. 104–111.

[2] S. Carlough and E. Schwarz, ―Power6 decimal divide, in Proc. 18th IEEE Symp. Appl.-Specific Syst., Arch., Process., Jul. 2007, pp. 128–133.

[3] L. Dadda and A. Nannarelli, ―A variant of a Radix-10 combinational multiplier,‖ in Proc. IEEE Int. Symp. Circuits Syst., May 2008, pp. 3370–3373.

[4] L. Eisen, J. W. Ward, H.-W. Tast, N. Mading, J. Leenstra, S. M. Mueller, C. Jacobi, J. Preiss, E. M. Schwarz, and S. R. Carlough, ―IBM POWER6 accelerators: VMX and DFU,‖ IBM J. Res. Dev., vol. 51, no. 6, pp. 663–684, Nov. 2007.

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

[6] 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.