2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems

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

Ch. Santosh Varma, Syed Ershad Ahmed and M.B.Srinivas
Department of Electrical Engineering, Birla Institute of Technology and Science –Pilani, Hyderabad Campus
Hyderabad -500078, India

Abstract— This paper presents a new architecture for a 7-bit Binary to BCD (BD) converter which forms the core of our proposed high speed decimal Multi-operand Adder. Our proposed design contains various improvements over existing architectures. These include an improved 7-bit BD Converter that helps in reducing the delay of the Multi-operand decimal Adder. Simulation results indicate that with a marginal increase in area, the proposed BD converter exhibits an improvement of 55% in delay and up to 27% reduction of power-delay product over earlier designs. Further the decimal Multi-operand Adder achieves up to 15% faster design and power-delay product falls to 13% when compared to previously published results.

Keywords-Multiplier, Decimal Arithmetic, Binary to BCD Converter, Multi-operand Adder;

I. INTRODUCTION

The use of decimal arithmetic has been increasing over binary due to increase in the applications of internet banking and there are many others places where precision is very important. Binary digits have a disadvantage of not being able to represent digits like 0.1 or 0.7, requires an infinitely recurring binary number. The availability of multi-operand decimal adders can facilitate financial and commercial applications based on existing huge databases.

The simultaneous addition of several decimal numbers is the common operation in multiplication and division algorithms. Multi-operand addition is a vital operation as it is a core component 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 paper introduces a multi-operand decimal addition algorithm by employing high speed binary to BCD converter circuit, which speeds up the process of decimal addition when multiple BCD operands are added together. A Novel design for 7-bit binary to BCD converter circuit is proposed. Further, analysis is done with respect to the existing binary to BCD converter architectures. The proposed algorithm is fundamentally different from multi-operand BCD addition algorithms [3, 5] since intermediate BCD corrections are not done rather correction is done at the final stage to get proper BCD results. As the decimal corrections are achieved separately from the computation of the binary sum, such that the layout of the binary carry-save adder does not require any further rearrangement, the design can perform as unified Binary/ BCD multi-operand adder.

In the next section, we present preliminary information about the previous work on multi-operand adders and discuss the binary to BCD converter. The proposed design is presented in section III. Experimental results related to the performance of existing and proposed multi-operand adders are compared in section IV and conclusions are drawn in section V.

II. RELATED WORK

A. Multi-operand Addition

The multi-operand decimal addition is dealt exhaustively by kenney [3] which is carried in serial fashion. The addition is realized by employing CSA's and depending on the carries at the intermediate stages from the CSA's, the design finds decimal correction logic in two ways that are categorized as speculative and non-speculative multi-operand addition [3]. The speculative algorithm adds the correction i.e six in the intermediate stages while non- speculative does this correction at the later stages of the design based the intermediate CSA carries. In case of Non-speculative, carries are generated as a result of addition of BCD input operands which are passed to the higher significant digit. The final decimal sum and carry is obtained by feeding the sums and carry-outs from the carry-save adder tree into combinational logic network. The architecture [4] utilizes tree of binary CSA`s to compress operands and makes use of a CPA to obtain non-redundant binary value followed by binary to decimal converter cell. This architecture can perform both BCD and binary multi-operand addition.

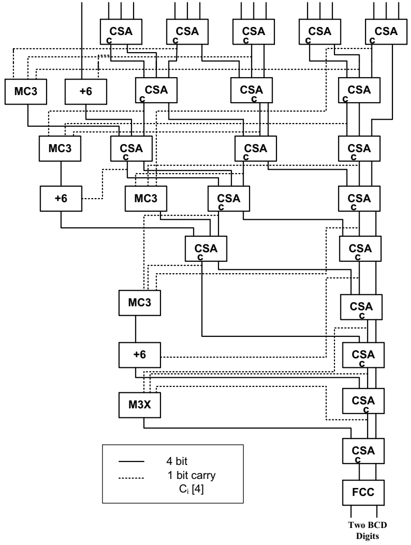

Figure 1. Decimal multi-operand Adder [5]

1063-9667/14 $31.00 © 2014 IEEE
DOI 10.1109/VLSID.2014.69

The paper [5] discusses a design specifically for a 16 multi-operand which does exclusively 16 operand BCD addition, whereas [3-4] designs can be configured to variable number of BCD operands. The architecture [5] proposes an algorithm on similar lines to that of speculative algorithm proposed by [3], however performs the computations in parallel [5]. The design introduces new blocks called as Merging three circuit MC3, MC4 and M3X, these blocks are used for the correction process in the intermediate stages of the architecture followed by Final Correction Circuit (FCC), which produces the BCD output as shown in figure 1.

B. Binary to BCD conversion

The algorithms proposed in [6-8] converts a 7-bit binary number to 2-digit BCD number to support high performance decimal multiplication. The proposed algorithm in [6] calculates the contributions for lower digit and the higher digit of the BCD number from each of the input binary bits which are later summed in a binary fashion. The contributions are generated as shown in figure 2 below. The drawback of this algorithm the computation is performed in a binary manner rather than a BCD approach, this results in erroneous BCD conversion.

The architecture in [6] is corrected in [7], by adding the contributions in a BCD fashion. The algorithm in [7] partitions or splits the binary input into two sub-parts, three MSB's and four LSB's. The design then calculates the contributions for the two BCD digits. These contributions are added in BCD fashion to get the final result.

80	40	20	10	8	4	2	1
0	P_6	P_5	P_4	P_3	P_2	P_1	P_0
0	0	P_6	P_5	0	P_4	P_4	0
				0	P_6	P_5	0
b_3	b_2	b_1	b_0	c_3	c_2	c_1	c_0

Figure 2. Binary to BCD Conversion [6]

Work in [8] proposes two architectures, the Three-Four Split and Four-Three Split binary to BCD converters. The Three-Four Split algorithms is similar to the one proposed in [7] however [8] designs a clever optimized D_L and D_H generator blocks, resulting in better performance in terms of area, delay and power.

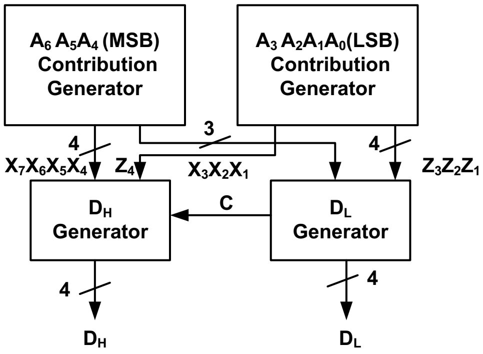

Figure 3. Three-Four Split 7-bit BD Converter [8]

The diagram of the three-four split is depicted in figure 3. The Four-Three Split design, partitions the 7-bit binary input into four MSB and three LSB bits. Since the LSB bits don't contribute to the higher BCD digit so the LSB contribution generator is removed resulting in area savings at the cost of increase in the complexity of the MSB contribution generator as shown in figure 4. The Three-Four split is faster than the four- three split whereas the Four-Three Split algorithm results in a more area efficient architecture. In the case of multi-operand addition the overall delay of adder play a pivotal role as adders will be in the critical path.

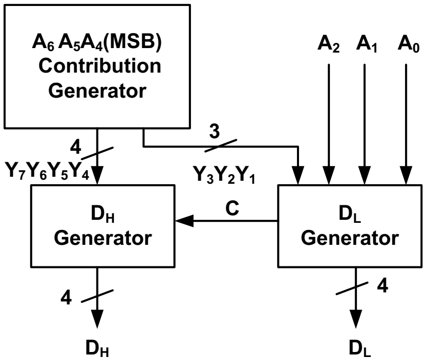

Figure 4. Four - Three Split 7-bit BD Converter [8]

III. PROPOSED DESIGN:

A. Multi-operand Decimal Adder:

The binary parallel multi-operand addition is realized using a CSA tree for compressing the input operands. Efficient multi-operand binary adder circuits can be realized using carry-save adders. The absence of carry propagation until the last stage makes the CSA adders very fast [9]. An added advantage is their simple structure. The proposed algorithm is as depicted in the figure 5. It comprises of a binary tree structure formed by 3:2 CSA followed by a high speed Binary to Decimal convertor as depicted in figure 5. The proposed BD converter will be discussed in detail section III B.

The proposed multi-operand adder architecture can perform both decimal and binary multi-operand operation. Figure 6 illustrates the multi-operand addition of 8 input operands and a carryin (C_{in}) shown in bold (9+9+9+9+9+9+9+9+**7**) using the proposed algorithm in figure 5. We have considered the extreme case of each input operand being 9. Further the carry in from the previous multi-operand column can at most be **7**.The algorithm computes the binary sum S_{binary} by summing up the input operands in a parallel fashion. Unlike the existing BD converters [7-8] we have removed the contribution blocks resulting in a very fast design at the cost of increase in the complexity of the D_H and D_L generators as shown in figure. The binary output S_{binary} is fed to the BD converter which produces 2 digit BCD number, $S_{decimal}$. The design proposed is different from [5] in the sense that the design doesn't do the correction at the intermediate stages, rather it does the decimal conversion after the binary sum has been produced, this improves the performance in terms of speed.

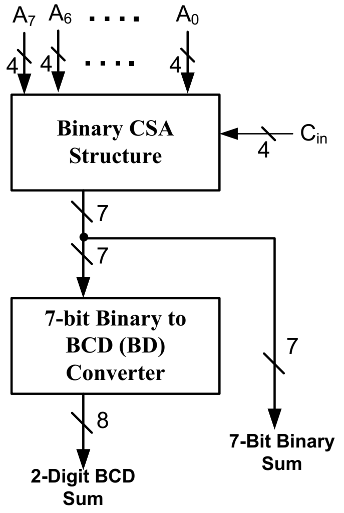

Figure 5. Proposed Multi-operand Decimal/Binary Adder

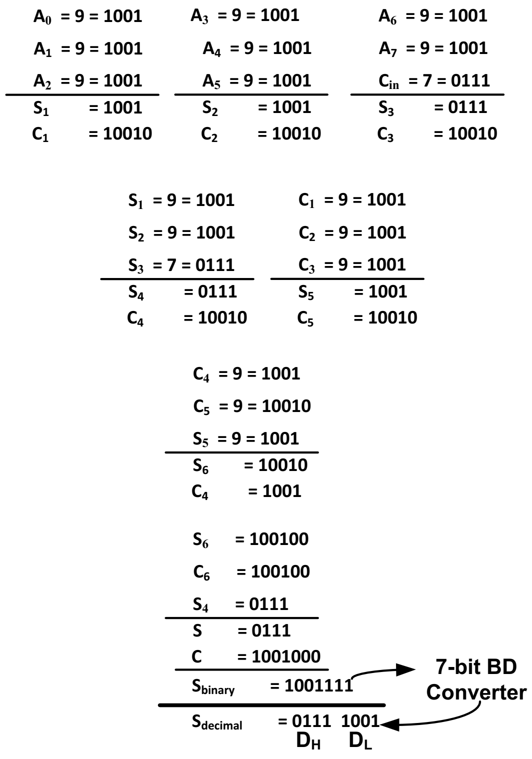

Figure 6. Example of proposed eight operand decimal addition

B. Proposed 7- bit Binary to BCD Converter:

The proposed BD converter was designed with the intention of speeding up of the multi-operand decimal adder. The 7-bit binary number is converted to the two BCD digits i.e. four LSB bits and four MSB bits of $S_{decimal}$ form D_L and D_H respectively. In [8] the contribution block adds to the critical path delay of the converter, the proposed design aims at removing these contribution blocks and thereby drastically improving the overall delay of the converter as depicted in figure 7.

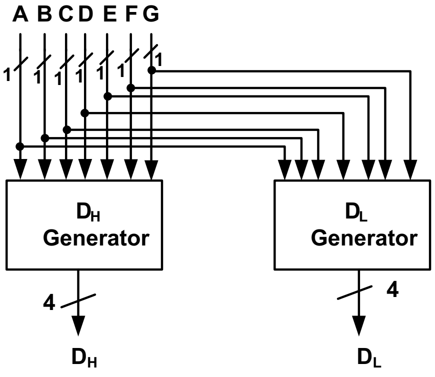

Figure 7. Proposed 7-bit BD Converter

Instead of splitting and finding the contributions for D_H and D_L a method is devised to directly find the contribution to each bit of BCD digits by using equations (1) and (2) respectively.

The maximum BCD number is nine ($9_{10} = 1001_2$) so eight operand decimal addition will result in a maximum of seventy-two ($72_{10} = 1001000_2$), hence a carry in to higher position can be maximum of seven. So, the proposed BD convertor block has been optimized to convert a maximum of seventy-nine ($79_{10} = 1001111_2$), implying in D_H the MSB D_{H3} can never be `1`.

Further, it can be observed that the first two MSB bits of S_{binary} cannot be of value `1` simultaneously. Based on this observation the equations (1-2) were devised for the convertor block.

$$D_{Hj} = \sum_{i=0}^{2} P_{ij} \quad (1)$$

TABLE I. PROPOSED D_H GENERATOR OF 7-BIT BD CONVERTER

j i	0	1	2	3
	D_{H0}	D_{H1}	D_{H2}	D_{H3}
0	A'B'[DEF+C(D+F)'+C'D(E+F)]	B'C[D+E]	0	0
1	B[C'D'+C[DE'+D'(E+F)]]	B[C'D'+CDE]	B[C+D]	0
2	A[D+EF]	AC'	AC'	0

where P_{ij} corresponds to (i , j) element in the Table II

The four MSB bits D_{H0} , D_{H1} , D_{H2} and D_{H3} are formed by replacing j =0 ,1,2,3 in (1). From the Table I and (1) we can obtain the value of D_H as follows.

Case I: j = 0

$$D_{H0} = A'B'\left[DEF + C(D+F)' + C'D(E+F)\right] + B\left[C'D' + C\left[DE' + D'(E+F)\right]\right] + A\left[D+EF\right] \quad (2)$$

Similarly we can obtain the values of D_{H1} , D_{H2} , D_{H3} by replacing the value of j = 1 ,2, 3 in (1)

$$D_{Lj} = \sum_{i=0}^{2} Q_{ij} \quad (3)$$

TABLE II. PROPOSED D_L GENERATOR OF 7-BIT BD CONVERTER

i \ j	0	1	2	3
	D_{L0}	D_{L1}	D_{L2}	D_{L3}
0	G	A’B’[C’[D’F+DEF’]+ C[DE’F+D’[E⊕ F]’]]	A’B’[CE’[D+F’] + C’E[F+D’]]	A’B’[C’DE’F’ + C[DEF’+D’E’F]]
1	G	B[D’F’[E+C’] + D[CE’F’+F[E+C’]]]	B[C[E’F’D+EFD’]+ C’[E⊕ [DF’]]]	B[CE’[D⊕ F]’+ C’D’EF]
2	G	A[DF’+C’E’F]	A[E’F+EF’D]	AE[F⊕D]’

where Q_{ij} corresponds to (i , j) element in the Table II
The four LSB bits D_{L0} , D_{L1} , D_{L2} , D_{L3} are produced by substituting j =0,1,2,3 in (3). From the Table II and (4) we can achieve the value of four bits of D_L as follows.

Case I: j = 0

$$D_{L0} = G \quad (4)$$

On similar lines we can obtain the values of D_{L1} , D_{L2} , D_{L3} By substituting the value of j = 1 ,2, 3 in (3)

IV. SIMULATION RESULTS

All the 7-bit Binary to BCD converters and Multi-operand structures were described using Verilog data flow modeling and simulated using Cadence Incisive Unified Simulator (IUS) v6.1. The Binary to BCD converters and Multi-operand designs were mapped on TSMC 180nm Technology Slow-Normal library (operating conditions 0.9V, 125°C) using Cadence RTL Compiler v7.1. All the inputs were set to have a toggle rate of 50%.

Multi-operand structures based on the proposed algorithm were designed and the Binary to BCD converter in the proposed algorithm was replaced with that of architecture [8] for fair comparisons. Table III shows the comparison of Binary to BCD converter with existing design [8]. Synthesis results show that there is a reduction in delay by 55 % with a tradeoff in power by 18 %. This in turn reduces power delay product by 27 %.

TABLE III. COMPARISON OF PROPOSED BINARY TO BCD DESIGN WITH FOUR –THREE SPLIT AND THREE– FOUR SPLIT [8] BINARY TO BCD DESIGNS

Metric	Proposed Design	Four –Three Split [8]	Three –Four Split [8]
Area (µm2)	201.802 (100%)	137.592 (68%)	150.293 (74.5%)
Delay (pS)	590F (100%)	914 (155%)	845 (143%)
Power (nW)	3282.645 (100%)	2703.367 (82.3%)	2693.709 (82%)
Power Delay Product (fJ)	1.936734 (100%)	2.470907 (127.6%)	2.276176 (117%)

Comparison between proposed multi-operand adder and modified multi-operand adder designed using Four –Three Split and Three– Four Split [8] is presented in Table IV. From the Table, it is evident that there is improvement in delay by 15% and in turn reduces power delay product by 13%.

From these three designs, the proposed multi-operand adder in conjunction with modified multi-operand adder using Four –Three Split and Three– Four Split [8] gives better performance in terms speed as well as power –delay product. Further it is evident from Table IV that the proposed design performs better compared to [4] with respect to delay as well as power delay product.

TABLE IV. COMPARISON OF PROPOSED MULTI-OPERAND ADDER WITH MODIFIED THREE SPLIT AND THREE– FOUR SPLIT [8] ADDERS

Metric	Area (µm2)	Delay (pS)	Power (nW)	Power Delay Product (fJ)
Proposed Design	749.3 (100%)	2146 (100%)	34937.9 (100%)	74.976 (100%)
Design A [8]	685.1 (91.4%)	2470 (115%)	34358.7 (98.3%)	84.86 (113.2%)
Design B [8]	697.8 (93.1%)	2401 (112%)	34349 (98.3%)	82.472 (110%)
Dadda [4]	655.5 (87.5%)	2429 (113 %)	33382.2 (95.5%)	81.08 (108 %)

Design A [8]: Four –Three split converter
Design B [8]: Three– Four split converter

V. CONCLUSIONS

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. Simulation results demonstrate the efficiency of our proposed BD converter as well as multi-operand decimal adder with respect to exiting designs.

REFERENCES

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

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

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

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

[5] 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.

[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]- 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.

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