International Journal of Pure and Applied Mathematics
Volume 120 No. 6 2018, 1377-1391
ISSN: 1314-3395 (on-line version)
url: http://www.acadpubl.eu/hub/
Special Issue

http://www.acadpubl.eu/hub/

HYBRID SIGNED DIGIT PARALLEL AND MULTI OPERAND BCD ADDERS

G.Sreelakshmi[1],Dr. Kaleem Fatima[2],Dr.B.K.Madhavi[3],
Geethanjali College of Engineering and Technology[1],
Muffakamjah College of Engineering and Technology[2],
Sridevi Womens Engineering College,Hyderabad[3]

June 25, 2018

Abstract

Decimal Arithmetic is having its own significance in many fields like commercial, financial, industrial and scientific applications. It plays a vital role in Floating point and Fixed point Decimal Processors. Adders and Multipliers are basic building blocks of any arithmetic unit. This paper presents a new method for the decimal signed digit addition based on the vinculum digit set {-5, 5} where the delay associated with carry generation and propagation is significantly reduced. The proposed Hybrid signed digit adder, adds two N-digit operands using binary fast adders in parallel. The correction logic is parallel applied along with one previous stage hybrid carry. This reduced the critical path delay very significantly. Multi operand BCD addition up to 8 operands is successfully implemented using the above mentioned parallelism in binary tree method. The proposed multi-operand BCD adder is 3 times faster compared to the method proposed in Signed Digit Adder multi operand adder of [17]. All the designs are implemented in Verilog HDL and tested exhaustively on FPGA and cadence digital encounter tools 0.18m technology and the results show that the proposed

parallel multi-operand BCD adder provides very less delay and is suitable for High speed Applications

***Keywords*:**Decimal Arithmetic, Signed Digit, Multi operand, Vedic Vinculum (VBCD)

1 INTRODUCTION

Decimal Number systems became more attractive from last decade because of its importance in currency conversions, tax calculations, billing and banking systems. Owing to the importance of decimal number system in our daily life few standards were passed by IEEE in 2008 to perform Decimal fixed and floating point arithmetic operations popularly known as IEEE 754-2008 format [18]. From zhang. Applications which consume more time for decimal calculation stand to benefit from hardware based decimal operations. Therefore research into decimal arithmetic has gained much interest in researchers. Improvement of adders performance can benefit directly other operations like subtraction and multiplication where many methods were proposed to enhance the performance of adders. In literature we had various adders like binary Carry save [1,2] Reduced Delay BCD Adders[4], carry look ahead adders[4][5], dynamic decimal CLA [5], decimal carry free adder [10], Redundant BCD adder[11] and svoboda adder[8]. The main motivation behind this paper is to introduce a concept of signed digit number system. It uses both positive and negative numbers in the process of addition. It has the benefit of carry free addition. The first signed digit adder was proposed by svoboda in the year 1969 with the digit set of -9, 9. Several papers have been presented in last decade.

This paper proposes signed parallel adder and multi operand BCD adders along with simulation and synthesis results.

Existing Signed Digit Decimal Adders

Decimal Carry Free Adder: In this paper author discussed the importance of complex functions like multiplication and digit recurrence division where partial products must be calculated and accumulated as partial products. In this he took an advantage of

signed digit (SD) systems which limit the carry propagation to few digit positions and can be processed all digits in parallel. In this authors [10] provided a decimal signed digit adder in the digit set of [-9, 9], and 5 bits are used to represent digit set.

Svoboda Adder: It is a very early method proposed by Svoboda. He used the concept of Signed Digit set where number set is a combination of +ve and ve numbers. It uses digit set from [-6, 6] and each digit is represented using 5 bit binary code. It requires conversion from BCD to signed digit form and vice versa [8].

Redundant BCD Adder: In this author uses digit set between [-7, 7] and is represented in 4 bit two's complement encoding called as Redundant binary coded decimal (RBCD) to represent signed digit set. In this author proposed carry free operations. The addition delay is constant for n-digit RBCD adder. Conversions can be carried out in a constant time and for conversion simple Ripple carry operation and Carry Look Ahead circuit were accomplished. [8]

Signed Digit Decimal Adder: In this author uses the digit set [-9, 9] and is represented in 5 bit two's complement vector. Intermediate sum is of 6 bit wide with carry propagate adder and correction unit which uses the concepts of positive and negative magnitude components to represent +ve and ve carries. In this author used Carry Propagate Adder, carry generation and correction vector in his hardware [17].

Speculative SD Adder: In this author used 5 bit two's complement form to represent the digit set between -9 to 9. The input operands may be +ve or ve with two comparison constants +1 and -1. In this paper authors used compressor circuit for addition, correction circuit, conversion circuit for converting SD number into unsigned BCD result [9].

Reduced Delay BCD Adder: In this paper author tried to reduce the delay of conventional BCD adder. In conventional adder input carry ripples upto the most significant digit. In this

paper he used the concept of CLA and generated Digit Carry DP and Digit Propagate signals for computing. All are generated in parallel added independently. In his hardware he used Adder circuit Analyzer and carry network for computation. This work is extended to Modified Reduced Delay BCD Adder in which they rearranged the hardware architecture for further reduction in delay [9].

Fast Signed Digit Decimal Adders: In this author proposed Fast signed digit adders and also multi operand decimal adders. He used the concept of signed digit with number set [-9, 9]. He represented in 5 bit conventional binary in two's complement notation. Intermediate sum is of 6 bits wide and carry propagate adder is used for propagation of carry bits. For multi operand adder he took the advantage of parallel prefix adder. In this method any digit position carry depends only previous carry digit. There is no ripple carry effect since carry generation does not depend on C_{in}. Correction factor is required for which the sum vector is out of range. Depending on Carry bits correction vectors are generated for invalid sum bits [15].

Proposed Hybrid Parallel Decimal VBCD Adder:

In the proposed architecture, the digit set used is -5, 5 inclusive and is represented using a conventional 4 bit, two's complement vector[19]. For the digit at position i, A_i and B_i are added to get 4 bit wide intermediate sum Z_i. Further by inspecting Z_i if the value is outside the digit set it is passed through correction unit. This corrected intermediate sum generate carry component which must be propagated to the next stage.

$$\begin{cases} -1 \text{ if } Z_i \leq 5 \\ -1 \text{ if } Z_i \geq 5 \text{ .....(1)} \\ 0 \text{ otherwise} \end{cases}$$

By observing the intermediate sum digits zi we can determine +ve and ve carry signals. It uses simple CLA to calculate carry generations and propagations. All sum bits are generated in parallel and carries only propagate to the next digit. There is no ripple effect because carry generations does not depend on carry

in .

3. Decimal Signed-Digit Theory

The decimal signed-digit VBCD number system used here hasall the properties and limitations set forth in [12]. A decimal signed-digit set D is a special case of general signed-digit sets when r = 10 (r is the radix or base).

$D = \beta, ..., 1, 0, ...\alpha$.....(2)

where α, β and r are related by.....(3). $\alpha + \beta + 1 \geq r$.....(4)

Usually, symmetric signed-digit sets are used ($\alpha = \beta$). The signed-digit set is said to be redundant (any number has multiple representations) if (3) holds.

$\alpha > \frac{r-1}{2}$.....(5)

The decimal signed-digit set used in this work is {5,5} or 5 to 5 inclusive (r = 10, α = 5). With this decimal signed-digit set, to add two decimal signed-digits A_i and B_i, three quantities must be added together: the intermediate sum Z_i(i.e., interim sum), the carry C_i (i.e., transfer) which can take values in 1,0,1, and the correction 6 * C_i . The intermediate sum Z_i is A_i + B_i and ranges from 11 to 11inclusive carry bit along with the digit set {5,5}. The carry C_i is generated using a rule set such that $8 \leq U_i 6 * C_i 8$ holds. Determining C_i is done by comparing U_i to comparison constants. For the decimal signed-digit set used, one comparison constant is in {5,1}, the other in {1,5} and are usually chosen to minimize hardware complexity [15]. If 1 and 1 were the comparison constants, then C_i is 1 when U_i is less than 1; C_i is 1 when U_i is greater than 1; C_i is zero when U_i is 1, 0 or 1. The correction is and adds to the intermediate sum Z_i along with the previous digit's carry to obtain the sum. Note that i is between 5 and 5 inclusive, so an input carry of 1 or 1 will never cause a carry to propagate to the next decimal digit. The above is expressed in 6 to 8

C_i= -6*ci

$Z_i = A_i + B_i$ and C_{out}.....(6)

$S_i = Z_i + 6 * C_i + Ci - 1$.....(7)

$$C_i = \begin{cases} +1 \text{ if } 6 \leq Z < 10 \text{ and } C_{out} = 0 \\ -1 \text{ if } 6 \leq Z < 11 \text{ and } C_{out} = 1 \\ 0 \text{ otherwise} \end{cases} \quad \text{.....(8)}$$

An example demonstrates that the signed-digit technique still produces carries, but never propagates them. In other words, a digit's carry out is not a function of its carry in. In this example, the operands are absent of negative digits only to facilitate demonstration. The comparison constants are 1 and 1. For the least significant digit column, the intermediate sum is 5 + 5 = 10. Since 10 > 1, c0 = 1 is added to the next column and 6 is the correction for the least significant column.

Augend (Ai)	2 $\bar{3}$ 4 $\bar{1}$ $\bar{3}$ 1 3 $\bar{3}$
Addend (Bi)	2 3 3 $\bar{4}$ $\bar{4}$ 3 2 $\bar{4}$
Intermediate Sum (Zi)	4 0 7 5 $\bar{7}$ 4 5 $\bar{7}$
corrected Sum (Yi)	4 0 $\bar{3}$ 5 3 4 5 3
carry bits (Ni or Pi)	0 1 $\bar{1}$ $\bar{1}$ 0 0 $\bar{1}$ 0
Final sum (Si)	4 1 $\bar{4}$ 4 3 4 4 3

Special cases: It was observed that there are two extreme cases where one more correction is required.

Case1: corrected sum (Yi) 5 and carry bit is +1 final sum bit is +6 which is not vinculum. So one more correction stage is required.

Case 2: Similarly if corrected sum (Yi) is $\bar{5} + \bar{1} = \bar{6}$ which is an invalid vinculum digit. In this case also one more correction stage is required.

Example: *Addend*	$\bar{3}$	3	3
Augend	$\bar{2}$	2	4
Intermediate sum	$\bar{5}$	5	7
Corrected sum	$\bar{5}$	5	$\bar{3}$
Carry bits(Ni &Pi)		1	
Second stage Sum	$\bar{5}$	6	$\bar{3}$
Correction stage	1	$\bar{4}$	$\bar{3}$
Final Sum		$\bar{4}$	$\bar{4}$ $\bar{3}$

The general algorithm is as follows:

Algorithm:

Step1:(Z_i, C_i) = A_i+B_i // where Z_i=binary sum of the i^{th} digit and C_i is the carry out which may take the value C_i =0 or C_i =1. Where i is the i^{th} digit of the signed decimal digit.

Step2: Set correction.
Y_i= Z_i+6*C_i
If C_i=0 and $6 \leq Z_i \leq 10$, then (Y_i,P_i)= Z_i+6 where P_i= 1 termed as +ve carry C_i=1 and $6 \leq Z_i \leq 11$, then (Y_i, N_i)=Z_i 6 where N_i= 1 termed as ve carry

Else

Y_i=Z_i

Step3: (S_i, P_i, N_i) =Y_i+P_i -1 $N_i - 1$

Where S_i, is the final sum

1. For each digit, add the two operands to get the intermediate sum, U_i. The range for U_i is 11 to 11 inclusive of carry bit.

2. If Ui $>$ 1, set correction i = 6 and C_i =1. If $U_i < 1$, set correction i = 6 and $C_i = 1$. In the example, the c vector is shifted one digit position to the left so that addition occurs within each column.

3. Find the sum of the three vectors, U, correction and C. This operation, as those before, will not propagate carries because the method guarantees: $5 \leq S_i \leq 5$.

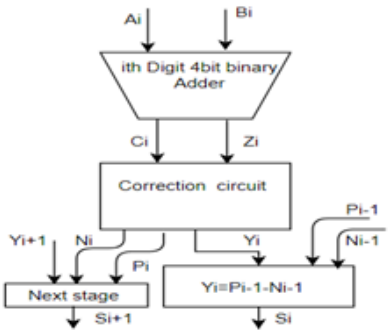

Fig 1: 4bit i^{th} digit parallel Adder

Figure 1 shows i^{th} digit adder in an N-bit number. It uses a simple binary CLA adder to add two operands of 4 bits each and gives intermediate sum Z_i and Carry C_i. If that output is not with in the range it is passed through correction circuit to give correct sum. The output of the correction circuit is $SumY_i$ and carry in the form of P_i , N_i. This Y_i is added with previous stage P_{i-1}, N_{i-1}. The output of it is final Sum S_i . Readers must remember that in parallel adder all digits can be added in parallel and present sum depends only on one stage(previous) carry delay.

N-digit Parallel Adder: Figure 2 shows an N-digit each of 4 bit wide is shown. The main advantage in this circuit is that all digits can be added in parallel for which correction is done on component Z_i. The ouput of correction circuit is corrected sum with carry ouputs P_i or N_i. These carries are forwarded to next sum digit Y_{i+1} where it is passed through simple binary CLA adder circuit where the output of it is final sum S_i.

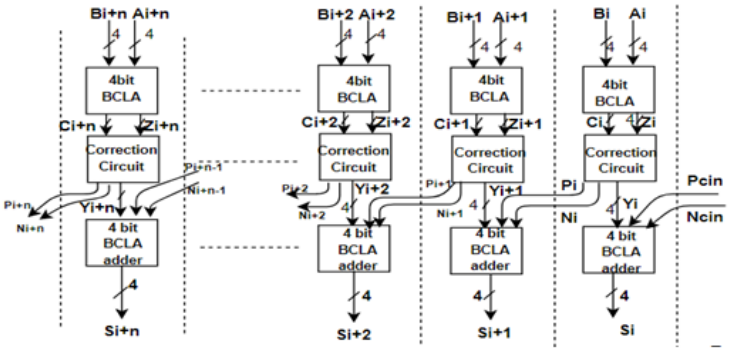

Fig 2: N-digit Parallel Adder

Multi-operand Addition: Two operand Signed digit Decimal adders can be used to add multiple numbers in binary tree method. Corrections are made after every addition. However it is apparent that immediate correction is not required. Correction can be done at later stage also. The operation for n-operand addition for a digit in the i^{th} position can be summarized in equations (6) - (8). ($a_{i,n}$ represents the n^{th} input for the i^{th} digit position). These shows that an intermediate sum can be calculated from multiple operands and a correction procedure must be used to get final sum.

For three operand addition, -5 > =u_i -6*c_i >= 5, for all possible u_i in {-15, 15} a C_i value can be found that makes -5< = U_i -6*C_i < =5 true. Since we are representing multi operand using two operand adders intermediate sum at any stage will be in the range . In our paper we proposed 8 operands with 8 digit each and went up to 32 bits.

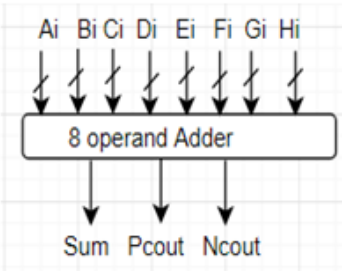

Fig:3 a) Multi-operand 4bit Adder

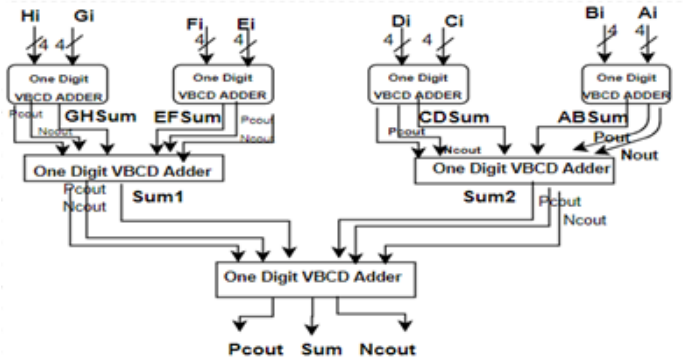

Fig3b): Multi_operand Adder using 2-operand Adders

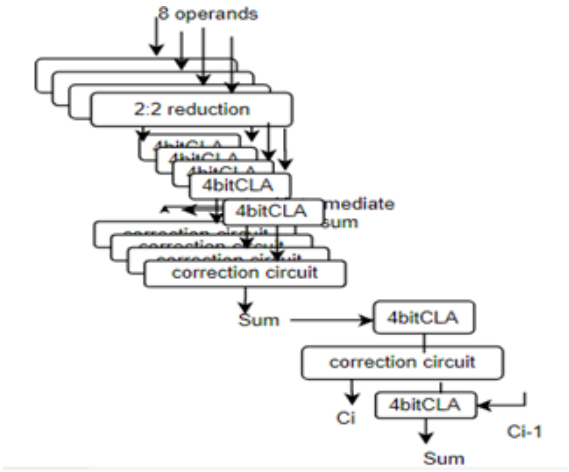

Fig: 3c) Multi operand Adder

2 RESULTS

All designs were written in Verilog HDL. Synthesis results for area, power and delay were obtained from TSMC 180nm standard cell technology from Cadence Digital Encounter Tools. Each design was compiled as an 8 bit adder or multiples of 8 bits.

Table 1:Synthesis Results for Parallel Adders

Sl.No.	No. of bits	LUT's	IOB's	Delay(ns)
1	4	15	14	3.037
2	8	32	28	3.037
3	16	64	52	3.037
4	32	132	112	3.037
5	64	268	224	3.037

Table 2:Synthesis Results for Multi Operand Adder (4bits)

Sl.No.	No. of operands	LUT's	IOB's	Delay(ns)
1	2	15	14	3.041
2	3	27	18	4.201
3	4	43	24	4.688
4	8	106	42	6.769
5	16	239	76	9.319

Table 3:Synthesis Report using TSMC 180nm technology (Parallel Adders)

Sl.No.	No. of bits	Area (μm^2)	Power(μw)	Delay (ns)
1	8	293	17.290	0.394
2	16	958	45.512305	0.422
3	32	2129	51.186626	0.422
4	64	4551	222.802032	0.422

Table 4: Synthesis Report using TSMC 180nm technology (Multi Operand Adders)

Sl.No.	No. of bits	Area (μm^2)	Power (μw)	Delay (ps)
1	8	878	43451	394
2	16	2578	132319.14	441
3	32	-----	-------	------

Table 5: Comparison Table for two digit Adders

Sl.No.	Type of Adder	Area (mm^2)	Delay(ns)	Area_Delay ($ns*mm^2$)
1	RBCD	0.03840	1.66	0.063744
2	Svoboda	0.0898	2.18	1.95700
3	Speculative	0.0852	1.40	0.11920
4	DFCA	0.0498	1.48	0.073704
5.	SDA	0.0373	1.36	0.050728
6.	Proposed	0.0263	0.82	0.021566

3 CONCLUSION

Hence we conclude that in this paper fast signed digit or hybrid digit two operand and multioperand decimal adders are proposed. Performance of adders have been investigated and compared with other adders. The proposed adder had very less delay and hence can be used for High speed applications.

References

[1] R. Kenney and M. Schulte, "High-Speed Multioperand Decimal Adders," IEEE Transactions on Computers, Vol. 54, No. 8, 2005, pp. 953-963. doi:10.1109/TC.2005.129

[2] I. D. Castellanos and J. E. Stine, "Compressor Trees for Decimal Partial Product Reduction," GLSVLSI'08:

Proceedings of the 18th ACM Great Lakes Symposium on VLSI, ACM, Orlando, 4-6 May 2008, pp. 107-110.

[3] I. S. Hwang, "High Speed Binary and Decimal Arithmetic Unit," USA Patent No. 4,866,656.

[4] A. Bayrakci and A. Akkas, "Reduced Delay BCD Adder," IEEE International Conference on Application Specific Systems, Architectures and Processors, Montreal, 9-11 July 2007, pp. 266-271.

[5] Y. You, Y. D. Kim and J. H. Choi, "Dynamic Decimal Adder Circuit Design by Using the CarryLookahead,"Copyright2011SciRe CS J. REBACZ ET AL. Copyright 2011 SciRes.CS 236 IEEE Design and Diagnostics of Electronic Circuits and Systems, Prague, 18-21 April 2006, pp. 242-244. doi:10.1109/DDECS.2006.1649627

[6] A. Kaivani and G. Jaberipur, "Fully Redundant Decimal Addition and Subtraction Using Stored-Unibit Encoding," Integration, the VLSI journal, Vol. 43, No. 1, 2010, pp. 34-41.

[7] L. Dadda, "Multioperand Parallel Decimal Adder: A Mixed Binary and BCD Approach," IEEE Transactions on Computers, Vol. 56, No. 10, 2007, pp. 1320-1328. doi:10.1109/TC.2007.1067

[8] A. Svoboda, "Decimal Adder with Signed Digit Arithmetic," IEEE Transactions on Computers, Vol. C-18, No. 3, 1969, pp. 212-215. doi:10.1109/T-C.1969.222633

[9] J. Moskal, E. Oruklu and J. Saniie, "Design and Synthesis of a Carry-Free Signed-Digit Decimal Adder," IEEE International Symposium on Circuits and Systems, New Orleans, 27-30 May 2007, pp. 1089-1092.

[10] H. Nikmehr, B. Phillips and C. Lim, "A Decimal Carry-Free Adder," Smart Structures, Devices, and Systems-II, Sydney, 13 December 2005, pp. 786-797.

[11] B. Shirazi, D. Yun and C. Zhang, "RBCD: Redundant Binary Coded Decimal Adder," IEE Proceedings on Computers and Digital Techniques, Vol. 136, No. 2, 1989, pp. 156-160. doi:10.1049/ip-e.1989.0021

[12] A. Avizienis, "Signed Digit Number Representations for Fast Parallel Arithmetic," IRE Transactions on Electronic Computers, Vol. EC-10, No. 3, 1961, pp. 389-400. doi:10.1109/TEC.1961.5219227

[13] B. Parhami, "Generalized Signed-Digit Number Systems: A Unifying Framework for Redundant Number Representations," IEEE Transactions on Computers, Vol. 39, No. 1, 1990, pp. 89-98. doi:10.1109/12.46283

[14] J. Rebacz, E. Oruklu and J. Saniie, "High Performance Signed-Digit Decimal Adders," IEEE International Conference on Electro/Information Technology, Windsor, 7-9 June 2009, pp. 251-255. doi:10.1109/EIT.2009.5189621

[15] D. Phatak and I. Koren, "Hybrid Signed-Digit Number Systems: A United Framework for Redundant Number Representations with Bounded Carry Propagation Chains," IEEE Transactions on Computers, Vol. 43, No. 8, 1994, pp. 880-891. doi:10.1109/12.295850

[16] J. Rebacz, E. Oruklu and J. Saniie, "Performance Evaluation of Multi-Operand Fast Decimal Adders," 52nd IEEE International Midwest Symposium on Circuits and Systems, Cancun, 2-5 August 2009, pp. 535-538. doi:10.1109/MWSCAS.2009.5236036

[17] Jeff Rebacz, ErdalOruklu, JafarSaniie "Fast Signed-Digit Multi-operand Decimal Adders" Circuits and Systems, 2011, 2, 225-236 doi:10.4236/cs.2011.23032.

[18] M. Erle, M. Schulte and B. Hickmann, "Decimal Float-ing-Point Multiplication via Carry-Save Addition," 18^{th} IEEE Symposium on Computer Arithmetic, Montpellier, 25-27 June 2007, pp. 46-55.