JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Design of 64-bit BCD Adder with Modified Connection Logic using Carry Select Adder

[1]Kuncham Pruthvi Raj, [2]M. Kiran Kumar, [3]Amrita Sajja

[1]Research Scholar, [2,3]Assistant Professor
[1,2,3]Electronics and Communication Engineering,
Anurag University, Hyderabad, India.

Abstract: Binary arithmetic is one of the most primitive and most commonly used applications in microprocessors, digital signal processors etc. But binary arithmetic is unable to fulfill the requirement of fractional terms thus causing inexact results. And in commercial applications fractional terms are common and efficient output is must requirement so we use Binary Coded Decimal (BCD) adders. Because they contain two binary adders and a carry look ahead adder for corrective logic, traditional BCD adders are slow. Thus, new high-speed BCD adders that employ a single binary adder have been devised and built here. The suggested BCD adder lowers the number of binary adders, which lowers the BCD adder's propagation time. Utilizing a Carry Select Adder, a 64-bit BCD adder was also implemented. Vivado 2018.2 version is used to design and implement the suggested BCD adders using Verilog. The results of conventional BCD adders are compared with proposed BCD adders. The experimental results show that the proposed BCD adders outperform the traditional BCD adders by 15.28%.

IndexTerms **- BCD Adder, Carry select adder , Multiplexer, FPGA implementation.**

I. INTRODUCTION

In our day-to-day life electronics play a important role. Back in 90's the technology evolution started and now in 21st century, the technology has been updated far ahead than expected. Once there used to be computer with big size of machines like CPU, Monitor, hard disks, floppy drives, etc. but now laptops, tabs, etc are in use. The approach for latest and simple use electronics is very much needed in present atmosphere. One such improvement is required in adders, amplifiers, transmitters, etc is needed so the design engineers are trying their best to decrease the area of the device and as well as to decrease the delay for the operation of the device. Now, the BCD adder is used by both computers and calculators. The BCD-Adder is used by calculators and computers that perform arithmetic operations directly in the decimal number system. The BCD-Adder accepts binary numbers that are decimal. The Decimal-Adder requires a minimum of nine inputs and five outputs. Nevertheless, a 16-bit binary integer is utilised instead of a 4-bit one in the BCD adder due to its increasing use. Additionally, a correction logic employing CLA is inserted in the adder to reduce delay time.

a) Binary-Coded Decimal (BCD): A type of binary encodings of decimal numbers known as Binary-Coded Decimal uses a defined amount of binary bits to represent each decimal digit. For every decimal digit in the BCD encoding, a 4-bit binary code is usually employed. This makes the binary representation of decimal numbers easier to understand and more readable for humans, which makes it especially helpful in situations where decimal arithmetic is crucial.

b) BCD Addition: BCD addition involves adding two BCD numbers, typically represented as strings of 4-bit nibbles. The addition is performed similarly to binary addition, digit by digit. If the result of adding two BCD digits exceeds 9 (1001 in BCD), a correction factor is added to bring the result back into the valid BCD range.

For example, consider the addition of two BCD digits:

```
  1001   9 in BCD)
+ 1010 (A in BCD, which is 10 in decimal)
  ------
  10011 (Corrected to BCD)
```

Here, the correction factor of 6 (0110 in binary) is added to ensure that the result remains a valid BCD digit.

II. LITERATURE SURVEY

"Parallel BCD adders with new majority gate architectures for quantum-dot cellular automata". This is a new definition for the output carry computation of a BCD adder using majority gates, which may be used to compute all of the multi-digit BCD adder's carries simultaneously [1-2]. To calculate carries in the BCD adder, we have implemented decimal group generate and decimal group propagate signals. We have thereby decreased the multi digit BCD adder's latency. To implement the suggested multi-digit BCD adder, we have used various binary adder types, including RCA, CFA, and parallel binary adder (PBA) [4]. Our PBA-based n-digit BCD adder decreases the area-delay product (ADP) and delay, theoretically. The 4-bit binary adder (ADD1), correction logic (CL), and the 1-digit BCD adder are the components of the 4-bit binary adder (ADD2). To obtain the binary total bS3:0 and the output carry bCout[3] , the binary adder (ADD1) adds the decimal numbers dA3:0, dB3:0, and dCin. To convert the binary sum bS3:0 to decimal sum dS3:0, the CL circuit generates the carry signals cL3:0 and dCout. Otherwise dCout = 1, then cL3:0 = $(0110)_2$, otherwise not, cL3:0 = $(0000)_2$[5]. The binary adder (ADD2) adds bS3:0 and cL3:0 to generate the decimal digit

dS3:0.The multi-digit BCD adder produces parallel decimal input carry, but the decimal group generate and propagate signals are independent of it. As a result, the multi-digit BCD adder's generation and propagation signals are shared by all decimal groups, and they share the same latency.

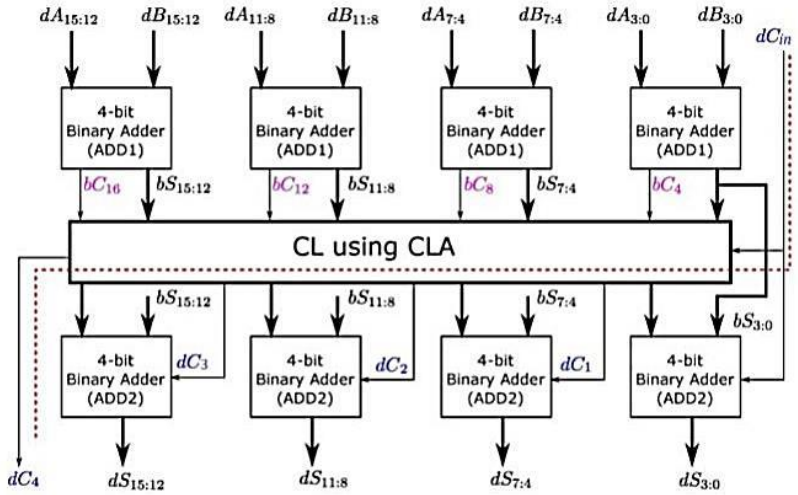

Fig.1 BCD adder correction using CLA

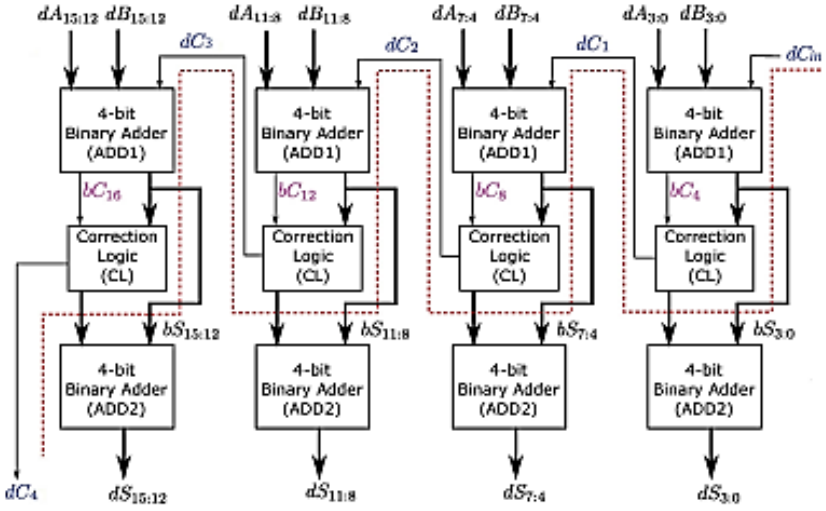

Fig.2 BCD adder correction logic

III. EXISTING METHOD

The design in [12] used the same block diagram for the implementation of BCD adder but they have used AND OR gate based output carry as shown in Fig.1.

$$dC_{out} = bC_{out} + (bS3:0 >= 10) + (bS3:0 == 9)dC_{in}$$

$$dCout = bCout + (bS3:0 >= 10) + (bS3:0 >= 9)$$

$$dCin = bCout + (bS3:0 >= 10) + [bCout + (bS3:0 >= 9)]dCin$$

The logic signals bCout + (bS3:0 >= 10) and bCout + (bS3:0 >= 9) can be rewritten as [bCout + (bS3:0 >= 10)] · [bCout + (bS3:0 >= 9)] and [bCout + (bS3:0 >= 10)] + [bCout + (bS3:0 >= 9)], respectively. By substituting these values in dCout, we can rewrite the equation of dCout as follows: dCout = [bCout + (bS3:0 >= 10)] · [bCout + (bS3:0 >= 9)] + [bCout + (bS3:0 >= 10) + bCout + (bS3:0 >= 9)]dCin (3) The dCout in (3) is clearly in 3-input majority gate form with inputs bCout + (bS3:0 >= 10), bCout + (bS3:0 >= 9) and dCin. dCout = M(bCout + (bS3:0 >= 10), bCout + (bS3:0 >= 9), dCin). The terms (bS3:0 >= 10) and (bS3:0 >= 9) are binary signals and we are calling these signals as decimal group generate and decimal group propagate signals. These two signals are represented as dG3:0 and dP3:0. dG3:0 = bCout + (bS3:0 >= 10). dP3:0 = bCout + (bS3:0 >= 9). The proposed majority gate form of dCout using dG3:0 and dP3:0 signals is given as follows:. dCout = M(dG3:0, dP3:0, dCin). The dCout uses decimal group generate and decimal group propagate signals for calculation. This is similar to CLA method for the calculation of carry. Because of this, we are calling CL stage as CL-CLA. The cL3:0 signal is calculated using the dCout. cL3:0 = {0, dCout, dCout, 0} The proposed dCout requires only 1 majority gate after calculating the dG3:0 and dP3:0 signals. The Proposed majority gate circuit for calculating dCout used the majority gate results presented for calculation of dG3:0. dG3:0 = bCout + bS3 · bS2 + bS3 · bS1 = M(bCout, M(bCout, bS3, 1), M(bS3, bS2, bS1)) To save the area, we have calculated dP3:0 as follows: dP3:0 = bCout + (bS3:0 >= 9) = bCout + (bS3:0 >= 10) + (bS3:0 == 9) = dG3:0 + bS3 · bS0. We can observe that the decimal group generate and decimal group propagate signals are independent of decimal input carry, which are produced parallelly in the multi-digit BCD adder. Consequently, all decimal group generate and decimal group propagate signals of the multi-digit BCD adder share the same delay. The majority gate circuit for calculating the carries dC1, dC2, dC3 and dC4 using decimal group generate and decimal group propagate signals. The delay required for calculating the dC4 is only the delay of four majority gates, which can be achieved from the proposed definition of dCout.

TABLE I: Theoretical Area, Delay and ADP Comparisons for Different Types of n-digit BCD Adders

Type	Area				Delay				ADP
	$I_{a1}(n)$	$I_{a2}(n)$	$I_{cl}(n)$	$I(n)$	d_{a1}	d_{a2}	$d_{cl}(n)$	$d(n)$	$I(n) \times d(n)$
Prop. RCA-BCD	$12n$	$12n$	$6n$	$30n$	7	7	$3+n$	$17+n$	$30n^2+510n$
Prop. CFA-BCD	$12n$	$12n$	$6n$	$30n$	7	7	$3+n$	$17+n$	$30n^2+510n$
Prop. PBA-BCD	$14n$	$14n$	$6n$	$34n$	5	5	$3+n$	$13+n$	$34n^2+442n$
PBA-BCD [10]	$16n$	$10n$	$3n$	$29n$	5	4	$7n-5$	$7n+4$	$203n^2+116n$
BCD [11]	$12n$	$10n$	$3n$	$25n$	5	2	$7n-5$	$7n+2$	$175n^2+50n$
CFA-BCD [12]	$16n$	$12n$	$6n$	$34n$	5	4	$2+2n$	$2n+11$	$68n^2+374n$

The total delay required for the n-digit BCD adder is the sum of delay required for ADD1 (da1), ADD2 (da2) and CLCLA (dcl(n)) circuits as shown in Fig. 2. All ADD1 blocks in n-digit BCD adder can calculate in parallel. The da1, da2 and dcl(n) represent the delay of 1-digit ADD1, ADD2 and n-digit CL-CLA blocks, respectively. The delay da1 and da2 depend upon the selection of 4-bit binary adder. In case of proposed PBA-BCD design, both of the da1 and da2 values are 5 majority gates. The delay dcl(n) of n-digit BCD adder is the sum of delay required for calculation of dGi+3:i , dPi+3:i and all dCouts, as shown in Fig. 2. The delay required for dGi+3:i and dPi+3:i terms is 3 majority gates, as shown in Fig. 1. An n-digit BCD adder requires delay of n majority gates for calculation of all dCouts after calculation of dGi+3:is and dPi+3:is, as shown in Fig. 1. The delay term dcl(n) is given as follow: dcl(n) = 3 + n (11) The generalized expression for calculating the delay complexity of an n-digit BCD adder (in terms of majority gates) is given as follow: d(n) = da1 + da2 + 3 + n.
The delay complexity for an n-digit RCA-BCD, CFA-BCD and PBA-BCD designs are shown below.
d(n) = 17 + n (13)
d(n) = 17 + n (14)
d(n) = 13 + n (15)

IV. PROPOSED METHOD:

We propose a new definition for BCD adder output carry computation in terms of majority gates and use it for computing all the carries of the multi-digit BCD adder in parallel. We have introduced decimal group generate and decimal group propagate signals to calculate carries in the BCD adder. As a result, we have reduced delay in the multi digit BCD adder. We have used different types of binary adders, such as RCA, CFA and parallel binary adder (PBA) for realizing the proposed multi-digit BCD adder. Theoretically, our PBA based n-digit BCD adder reduces the delay and area-delay product (ADP) by 50% compared with the existing designs. Carry flow adder (CFA) based and carry look ahead adder (CLA) based BCD adders, which show good performance. Moreover, exploits novel binary adder to propose the efficient 1-digit BCD adder, reducing comprehensive consumption. In order to fully utilize the majority gates, and rewrite the correction function for less delay by using Carry Select Adder. The BCD adder uses the 4-bit binary adder for generation of decimal digits. The performance of BCD adder also depends upon the selection of 4-bit binary adder. In this section, we are going to change the Correction logic by modified it with Carry Select Adder for area and delay complexity of n-digit BCD adder.By decoding bigger groups of adder bits, higher-order carry select adder decoding may minimise the amount of partial product rows by a wider margin. This procedure of adding requires 3operations.

- First, the 64 bits are grouped, and then we instantiate 16 instances of a 4-bit BCD adder (bcd_adder_4bit) within a generate loop.
- The 64-bit inputs a and b are split into 16 4-bit parts, each part being processed by one instance of the 4-bit BCD adder.
- In the second stage, the outputs of each 4-bit BCD addition (sum and carry_out) are assigned to the corresponding part of the sum and carry_out signals of the 64-bit BCD adder.
- The carry_out of the last 4-bit BCD addition is considered as the carry_out of the entire 64- bit BCD addition.
- In the second stage, for each 4-bit part we instantiate a carry select adder (carry_select_adder) to perform the BCD addition.
- The sum and carry out of each 4-bit addition are assigned to the corresponding part of the sum and carry_out signals of the 64-bit BCD adder.
- The carry_out of the last stage is considered as the carry_out of the entire 64-bit BCD addition. This design allows the 64-bit BCD addition to be performed in parallel using carry select adders for efficient carry handling.

Table 2: BCD Adder Truth Table

Binary Sum K Z8 Z4 Z2 Z1	BCD Sum C S8 S4 S2 S1	Decimal
0 0 0 0 0	0 0 0 0 0	0
0 0 0 0 1	0 0 0 0 1	1
0 0 0 1 0	0 0 0 1 0	2
0 0 0 1 1	0 0 0 1 1	3
0 0 1 0 0	0 0 1 0 0	4
0 0 1 0 1	0 0 1 0 1	5
0 0 1 1 0	0 0 1 1 0	6
0 0 1 1 1	0 0 1 1 1	7
0 1 0 0 0	0 1 0 0 0	8
0 1 0 0 1	0 1 0 0 1	9
0 1 0 1 0	1 0 0 0 0	10
0 1 0 1 1	1 0 0 0 1	11
0 1 1 0 0	1 0 0 1 0	12
0 1 1 0 1	1 0 0 1 1	13
0 1 1 1 0	1 0 1 0 0	14
0 1 1 1 1	1 0 1 0 1	15
1 0 0 0 0	1 0 1 1 0	16
1 0 0 0 1	1 0 1 1 1	17
1 0 0 1 0	1 1 0 0 0	18
1 0 0 1 1	1 1 0 0 1	19

Table 3 Carry Select Adder Truth Table.

Inputs			Outputs	
A	B	C_{in}	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

V. RESULTS

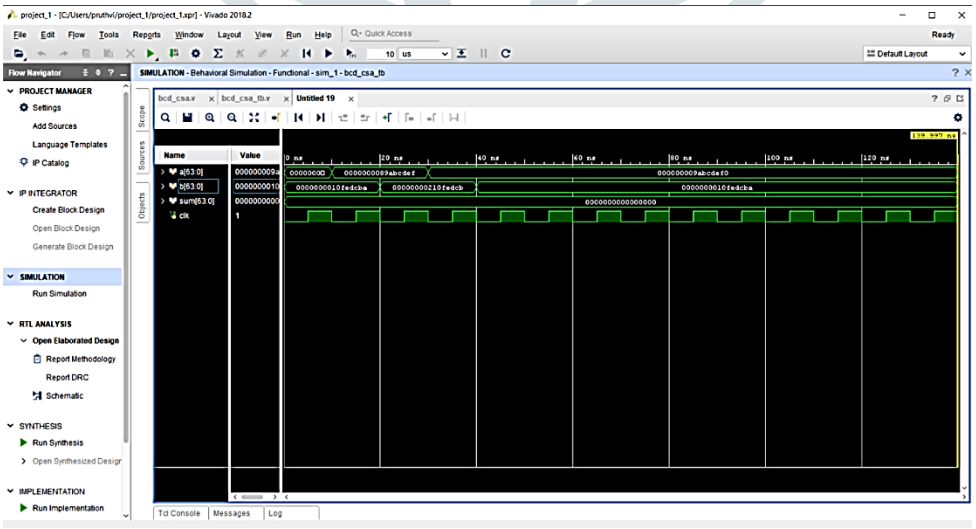

Fig.3 BCD adder results

.

Table 4 Comparison Table

PARAMETER	EXISTING	PROPOSED
Methodology	**CLA**	**CSA**
Delay(ns)	**7.75ns**	**4.835ns**

CONCLUSION

In this, BCD adder engineering is adjusted by supplanting the adjustment rationale of carry see ahead adder with carry select adder. The proposed plan of adder is reenacted and synthesized for different input bit sizes and after that assessed in terms of delay (ns) with the existing adders and region involved. The modern proposed strategy is would be managing Carry select adder, carry select adder for the most part a sort of parallel adder that points to diminish the engendering delay related with parallel expansion. It is planned to make strides the speed of expansion by employing a dual-bank structure, permitting for the parallel calculation of carries in both banks. By this delay is been diminished. In future, the BCD adders can be worked barcode functions which is point of sale, high automated testing machines, etc.

REFERENCES

[1] Zhang, Tingting, Vikramkumar Pudi, and Weiqiang Liu. "New majority gate-based parallel BCD adder designs for quantum-dot cellular automata." IEEE Transactions on Circuits and Systems II: Express Briefs 66, no. 7 (2018): 1232-1236.
[2] S. Shankland, “IBMs POWER6 gets help with math, multimedia,” in ZDNet News, 2006.
[3] X. Cui, W. Dong, W. Liu, E. E. Swartzlander Jr, and F. Lombardi, “High performance parallel decimal multipliers using hybrid BCD codes,” IEEE Transactions on Computers, vol. 66, no. 12, pp. 1994-2004, 2017.
[4] K. Walus and G. A. Jullien, “Design tools for an emerging SoC technology: quantum-dot cellular automata,” in Proceedings of the IEEE, vol. 94, no. 6, pp. 1225-1244, 2006.
[5] M. Vacca, M. Graziano, J. Wang, F. Cairo, and G. Causapruno, NanoMagnet Logic: An Architectural Level Overview. Springer Berlin Heidelberg, 2014.
[6] A. Khitun and K. L. Wang, “Nano scale computational architectures with spin wave bus,” Superlattices & Microstructures, vol. 38, no. 3, pp. 184- 200, 2005.
[7] M. Taghizadeh, M. Askari, and K. Fardad, “BCD computing structures in quantum-dot cellular automata,” in Proc. International Conference on Computer and Communication Engineering, 2008, pp. 1042-1045.
[8] F. Kharbash and G. M. Chaudhry, “The design of quantum-dot cellular automata decimal adder,” in Proc. IEEE International Conference on Multitopic, 2008, pp. 71-75.
[9] W. Liu, L. Lu, M. O’Neil, and E. E. Swartzlander Jr, “Cost-efficient decimal adder design in quantum-dot cellular automata,” in Proc. IEEE International Symposium on Circuits and Systems, 2011, pp. 1347-1350.
[10] G. Cocorullo, P. Corsonello, F. Frustaci, and S. Perri, “Design of efficient BCD adders in quantum-dot cellular automata,” IEEE Transactions on Circuits & Systems II Express Briefs, vol. 64, no. 5, pp. 575-579, 2017.
[11] D. Abedi and G. Jaberipur, “Decimal full adders specially designed for quantum-dot cellular automata,” IEEE Transactions on Circuits & Systems II Express Briefs, vol. 65, no. 1, pp. 106-110, 2017.
[12] Cocorullo, Giuseppe, Pasquale Corsonello, Fabio Frustaci, and Stefania Perri. "Design of efficient BCD adders in quantum-dot cellular automata." IEEE Transactions on Circuits and Systems II: Express Briefs 64, no. 5 (2016): 575-579.
[13] Haque, Mubin Ul, Zarrin Tasnim Sworna, Hafiz Md Hasan Babu, and Ashis Kumer Biswas. "A fast fpga-based bcd adder." Circuits, Systems, and Signal Processing 37, no. 10 (2018): 4384-4408.
[14] D. Ajitha, K. Ramanaiah, and V. Sumalatha, “An enhanced high-speed multi-digit BCD adder using quantum-dot cellular automata,” Journal of Semiconductors, vol. 38, no. 2, pp. 38-46, 2017.
[15] A. Roohi, R. Zand, S. Angizi, and R. F. Demara, “A paritypreserving reversible QCA gate with self-checking cascadable resiliency,” IEEE Transactions on Emerging Topics in Computing, 2018,
[16] W. Liu, L. Lu, M. O’Neill, and E. E. Swartzlander, “A first step toward cost functions for quantum-dot cellular automata designs,” IEEE Transactions on Nanotechnology, vol. 13, no. 3, pp. 476-487, 2014.