OPEN ACCESS
ISSN: 2249-6645

International Journal Of Modern Engineering Research (IJMER)

Carry Select Adder with High Speed and Power Efficiency

V P C Reddy[1], Chenchela V K Reddy[2], V Ravindra Reddy[3]

[1]*(ECE Dept, VVIT-Guntur, India, Email: vpcreddy@gmail.com)*

[2]*(ECE Dept., VVIT-Guntur, India, Email:vijayreddy235@gmail.com)*

[3]*(ECE Dept., VVIT-Guntur, India, Email:vssr.reddy@gmail.com)*

ABSTRACT*: -Design of efficient and high-speed data path logic systems are one of the most substantial areas of research in VLSI system design. In digital adders, the speed of addition is limited by the time required to propagate a carry through the adder. The sum for each bit position in an elementary adder is generated sequentially only after the previous bit position has been summed and a carry propagated into the next position. The CSLA is used in many computational systems to alleviate the problem of carry propagation delay by independently generating multiple carries and then select a carry to generate the sum. However, the CSLA is not time efficient because it uses multiple pairs of Ripple Carry Adders (RCA) to generate partial sum and carry by considering carry input , then the final sum and carry are selected by the multiplexers (mux). The logic operations of the RCA is done in split form, where HSG, HCG, FSG, and FCG represent half-sum generation, half-carry generation, full-sum generation, and full-carry generation, respectively. The basic idea of this work is to use Binary to Excess-1 Converter (BEC) instead of RCA in the regular CSLA. In the proposed scheme, CSLA is implemented using D-latch in which, the carry select (CS) operation is scheduled before the calculation of final sum, which is different from the conventional approach. Bit patterns of two anticipating carry words (corresponding to cin = 0 and 1) and fixed cin bits are used for logic optimization of Carry selection. A delay efficient CSLA design is obtained using D-latch logic unit. The proposed CSLA design involves significantly less **Delay** than the recently proposed BEC-based CSLA.*

I. INTRODUCTION

LOW-POWER, area-efficient, and high-performance VLSI systems are increasingly used in portable and mobile devices, multi standard wireless receivers, and biomedical instrumentation [1], [2]. An adder is the main component of an arithmetic unit. A complex digital signal processing (DSP) system involves several adders. An efficient adder design essentially improves the performance of a complex DSP system. A ripple carry adder (RCA) uses a simple design, but carry propagation delay (CPD) is the main concern in this adder. Carry look-ahead and carry select (CS) methods have been suggested to reduce the CPD of adders.

A conventional carry select adder (CSLA) is an RCA–RCA configuration that generates a pair of *sum* words and *output- carry* bits corresponding the anticipated input-carry (c_{in} = 0 and 1) and selects one out of each pair for *final-sum* and *final-output-carry* [3]. A conventional CSLA has less CPD than an RCA, but the design is not attractive since it uses a dual RCA. Few attempts have been made to avoid dual use of RCA in CSLA design. Kim and Kim [4] used one RCA and one add-one circuit instead of two RCAs, where the add-one circuit is implemented using a multiplexer (MUX). He *et al.* [5] proposed a square-root (SQRT)-CSLA to implement large bit-width adders with less delay. In a SQRT CSLA, CSLAs with increasing size are connected in a cascading structure. The main objective of SQRT-CSLA design is to provide a parallel path for carry propagation that helps to reduce the overall adder delay. Ramkumar and Kittur [6] suggested a binary to BEC-based CSLA. The BEC-based CSLA involves less logic resources than the conventional CSLA, but it has marginally higher delay. A CSLA based on common Boolean logic (CBL) is also proposed in [7] and [8]. The CBL-based CSLA of [7] involves significantly less logic resource than the conventional CSLA but it has longer CPD, which is almost equal to that of the RCA. To overcome this problem, a SQRT-CSLA based on CBL was proposed in [8]. However, the CBL-based SQRT- CSLA design of [8] requires more logic resource and delay than the BEC-based SQRT-CSLA of [6].

In [10] by Basant Kumar Mohanty and et.all proposed a new idea to made the performance of CSLA is fast instead with the use of BEC-1. They have designed a new logic formulation for CSLA. The carry select operation is scheduled before calculation of final sum which is different from conventional approach. They have used one RCA and one add one circuit instead of two RCA circuit. This method is known as sort of analysis because it divides main logic of operation into two parts first part gives half sum and carry calculation as it has done by HSC generator means half sum and carry generator and rest of the calculation part will be done by FSC (Full Carry Sum) generator . Therefore original sum and carry will be obtained by performing both this operation. There is consumption of quite large time to produced result sum.

This paper presents the choice of selecting the adder topologies. The adder topology used in designing carry select adder work are ripple carry adder, binary to excess one converter and D-latch. Addition is an indispensable operation for any digital system, DSP or control system. Adders are also very significant component in digital systems because of their widespread use in other basic digital operations such as subtraction, multiplication and division. Hence, for improving the performance of the digital adder would extensively advance the execution of binary operations inside a circuit compromised of such blocks. Ripple carry adder is the simplest but slowest adders with n operand size in bits. The carry-ripple adder is composed of many cascaded single-bit full-adders.

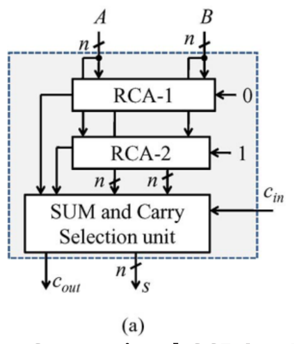
In the proposed scheme, CSLA is implemented using D-latch in which, the carry select (CS) operation is scheduled before the calculation of final sum, which is different from the conventional approach. Bit patterns of two anticipating carry words (corresponding to cin = 0 and 1) and fixed cin bits are used for logic optimization of Carry selection. A delay efficient CSLA design is obtained using D-latch logic unit. The proposed CSLA design involves significantly less **Delay** than the recently proposed BEC-based CSLA.

The rest of this brief is organized as follows. Logic formulation of CSLA is presented in Section II. The proposed D-Latch CSLA is presented in Section III and the performance comparison is presented in Section IV. The conclusion is given in Section V.

II. LOGIC FORMULATION

A.Conventional CSLA:

The CSLA has two units: 1) the *sum* and *carry* generator unit (SCG) and 2) the *sum* and *carry* selection unit [9]. The SCG unit consumes most of the logic resources of CSLA and significantly contributes to the critical path. Different logic designs have been suggested for efficient implementation of the SCG unit. We made a study of the logic designs suggested for the SCG unit of conventional and BEC-based CSLAs of [6] by suitable logic expressions. The main objective of this study is to identify redundant logic operations and data dependence. Accordingly, we remove all redundant logic operations and sequence logic operations based on their data dependence.



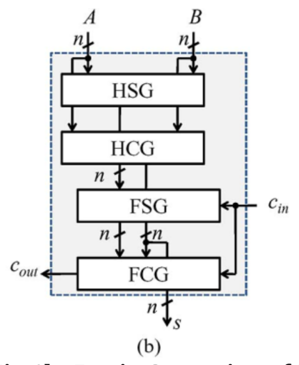


Fig 1a: Conventional CSLA   Fig 1b: Logic Operation of RCA

Logic Expressions of the SCG Unit of the Conventional CSLA:

As shown in Fig. 1(a), the SCG unit of the conventional CSLA [3] is composed of two *n*-bit RCAs, where *n* is the adder bit-width. The logic operation of the *n*-bit RCA is performed in four stages: 1) *half-sum* generation (HSG); 2) *half-carry* generation (HCG); 3) *full-sum* generation (FSG); and 4) *full-carry* generation (FCG). Suppose two *n*-bit operands are added in the conventional CSLA, then RCA-1 and RCA-2 generate *n*-bit *sum* (s^0 and s^1) and output-carry (c^0_{out} and c^1_{out}) cor-responding to input-carry ($c_{in} = 0$ and $c_{in} = 1$), respectively.

Logic expressions of RCA-1 and RCA-2 of the SCG unit of the *n*-bit CSLA are given as

$$s_0^0(i) = A(i) \oplus B(i) \quad c_0^0(i) = A(i) \cdot B(i) \qquad \text{(1a)}$$

$$s_1^0(i) = s_0^0(i) \oplus c_1^0(i-1) \qquad \text{(1b)}$$

$$c_1^0(i) = c_0^0(i) + s_0^0(i) \cdot c_1^0(i-1) \quad c_{out}^0 = c_1^0(n-1) \qquad \text{(1c)}$$

$$s_0^1(i) = A(i) \oplus B(i) \quad c_0^1(i) = A(i) \cdot B(i) \qquad \text{(2a)}$$

$$s_1^1(i) = s_0^1(i) \oplus c_1^1(i-1) \qquad (2b)$$

$$c_1^1(i) = c_0^1(i) + s_0^1(i) \cdot c_1^1(i-1) \quad c_{out}^1 = c_1^1(n-1) \qquad (2c)$$

where $c^0_1(-1) = 0$, $c^1_1(-1) = 1$, and $0 \leq i \leq n - 1$.

As shown in (1a)–(1c) and (2a)–(2c), the logic expression of $\{s^0_0(i), c^0_0(i)\}$ is identical to that of $\{s^1_0(i), c^1_0(i)\}$. These redundant logic operations can be removed to have an optimized design for RCA-2, in which the HSG and HCG of RCA-1 is shared to construct RCA-2. Based on this, [4] and [5] have used an add-one circuit instead of RCA-2 in the CSLA, in which a BEC circuit is used in [6] for the same purpose. Since the BEC-based CSLA offers the best area–delay–power efficiency among the existing CSLAs, we discuss here the logic expressions of the SCG unit of the BEC-based CSLA as well.

B. BEC based CSLA:

As shown in Fig. 2, the RCA calculates n-bit *sum* s^0_1 and c^0_{out} corresponding to $c_{in} = 0$. The BEC unit receives s^0_1 and c^0_{out} from the RCA and generates $(n + 1)$-bit excess-1 code. The most significant bit (MSB) of BEC represents c^1_{out}, in which n least significant bits (LSBs) represent s^1_1.

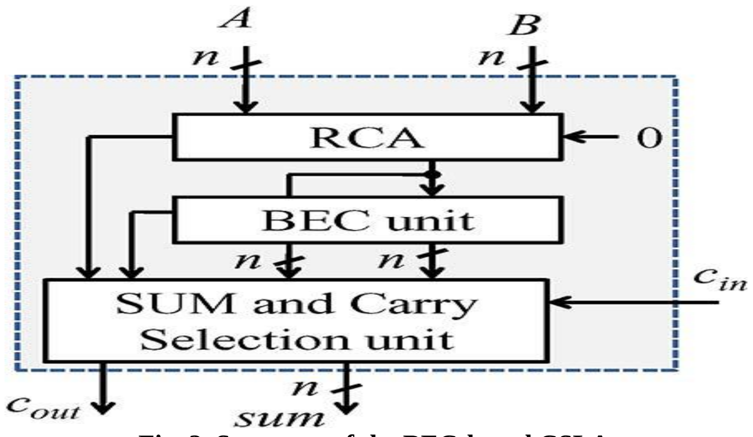


Fig. 2. Structure of the BEC-based CSLA;

The logic expressions of the RCA are the same as those given in (1a)–(1c). The logic expressions of the BEC unit of the n-bit BEC-based CSLA are given as

$$s^1_1(0) = s^0_1(0) \quad c^1_1(0) = s^0_1(0) \qquad (3a)$$

$$s_1^1(i) = s_1^0(i) \oplus c_1^1(i-1) \qquad (3b)$$

$$c_1^1(i) = s_1^0(i) \cdot c_1^1(i-1) \qquad (3c)$$

$$c_{out}^1 = c_1^0(n-1) \oplus c_1^1(n-1) \qquad (3d)$$

for $1 \leq i \leq n - 1$.

We can find from (1a)–(1c) and (3a)–(3d) that, in the case of the BEC-based CSLA, c^1_1 depends on s^0_1, which otherwise has no dependence on s^0_1 in the case of the conventional CSLA. The BEC method therefore increases data dependence in the CSLA. We have considered logic expressions of the conventional CSLA and made a further study on the data dependence to find an optimized logic expression for the CSLA.

It is interesting to note from (1a)–(1c) and (2a)–(2c) that logic expressions of s^0_1 and s^1_1 are identical except the terms c^0_1 and c^1_1 since ($s^0_0 = s^1_0 = s_0$). In addition, we find that c^0_1 and c^1_1 depend on $\{s_0, c_0, c_{in}\}$, where $c_0 = c^0_0 = c^1_0$. Since c^0_1 and c^1_1 have no dependence on s^0_1 and s^1_1, the logic operation of c^0_1 and c^1_1 can be scheduled before s^0_1 and s^1_1, and the select unit can select one from the set (s^0_1, s^1_1) for the *final-sum* of the CSLA. We find that a significant amount of logic resource is spent for calculating $\{s^0_1, s^1_1\}$, and it is not an efficient approach to reject one sum-word after the calculation. Instead, one can select the required carry word from the anticipated carry words $\{c^0$ and $c^1\}$ to calculate the *final-sum*. The selected carry word is added with the *half-sum* (s_0) to generate the *final-sum* (s). Using this method, one can have three design advantages: 1) Calculation of s^0_1 is avoided in the SCG unit; 2) the n-bit select unit is required instead of the (n + 1) bit; and 3) small output-carry delay. All these features result in an area–delay and energy-efficient design for the CSLA. We have removed all the redundant logic operations of (1a)–(1c) and (2a)–(2c) and rearranged logic expressions of (1a)–(1c) and (2a)–(2c) based on their dependence. The proposed logic formulation for the CSLA is given as

$$s_0(i) = A(i) \oplus B(i) \quad c_0(i) = A(i) \cdot B(i) \tag{4a}$$

$$c^0_1(i) = c^0_1(i-1) \cdot s_0(i) + c_0(i) \quad \text{for } c^0_1(0) = 0 \tag{4b}$$

$$c^1_1(i) = c^1_1(i-1) \cdot s_0(i) + c_0(i) \quad \text{for } c^1_1(0) = 1 \tag{4c}$$

$$c(i) = c_1^0(i) \quad \text{if } (c_{in} = 0) \tag{4d}$$

$$c(i) = c_1^1(i) \quad \text{if } (c_{in} = 1) \tag{4e}$$

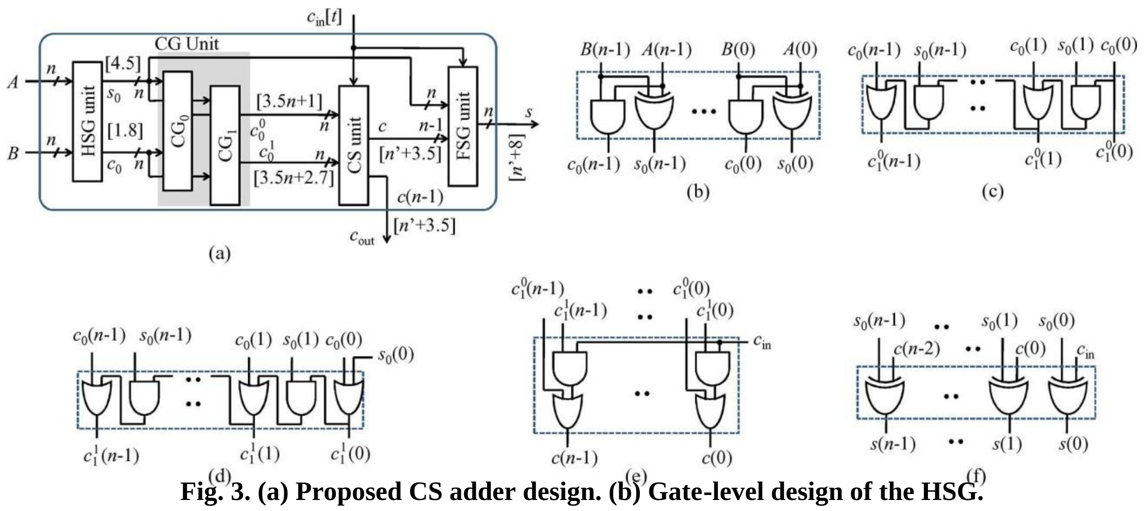


Fig. 3. (a) Proposed CS adder design. (b) Gate-level design of the HSG.

(c) Gate-level optimized design of (CG$_0$) for input-carry = 0.
(d) Gate-level optimized design of (CG$_1$) for input-carry =1.
(e) Gate-level design of the CS unit. (f) Gate-level design of the final-sum generation (FSG) unit.

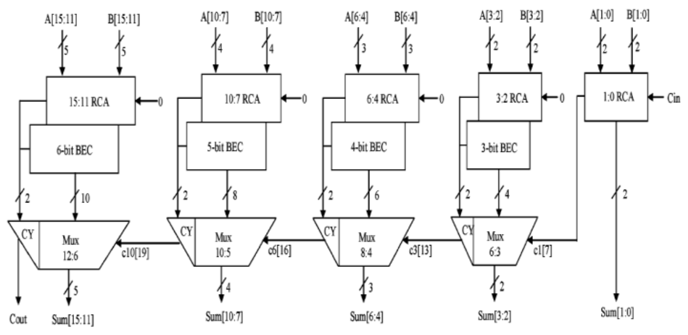


Fig 4: 16-BIT CSLA WITH BEC ARCHITECTURE

III. PROPOSED D-LATCH BASED CSLA

To compensate power consumption in previous technology we are proposing a new technique with D latch. D latches are used for load and store operation because it will give output as one when clock is equal to one.

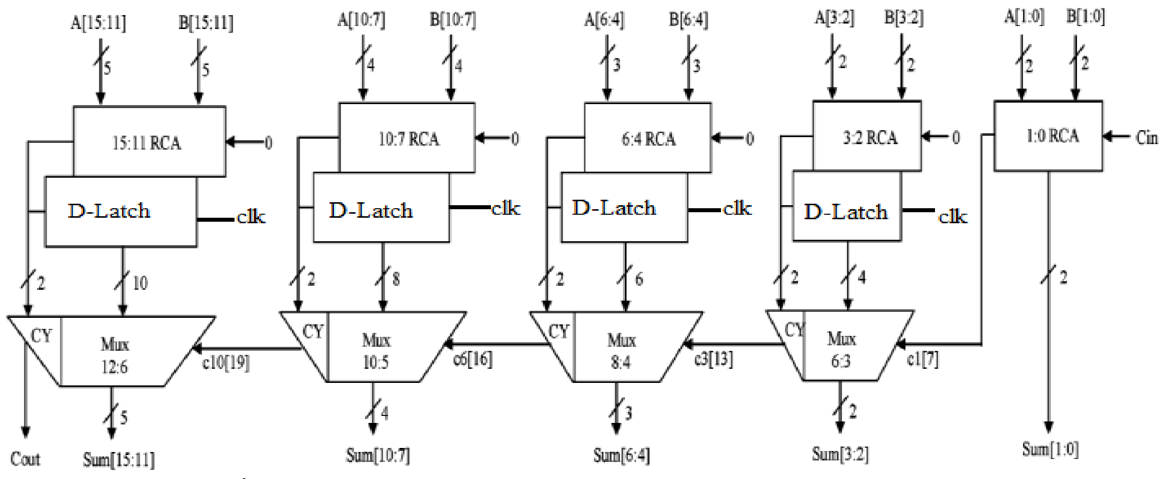


Fig 5: 16-BIT CSLA WITH D-LATCH ARCHITECTURE

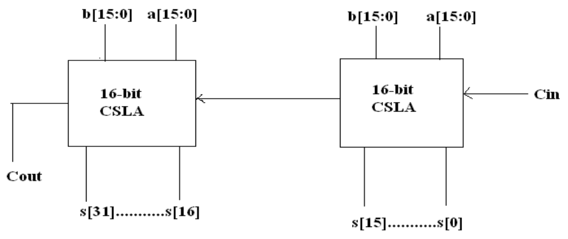


Fig 6: 32-BIT CSLA WITH D-LATCH ARCHITECTURE

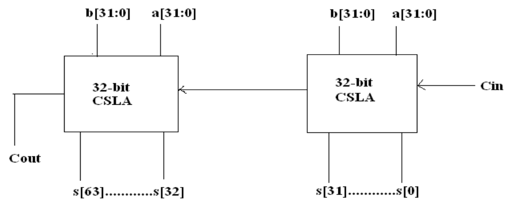
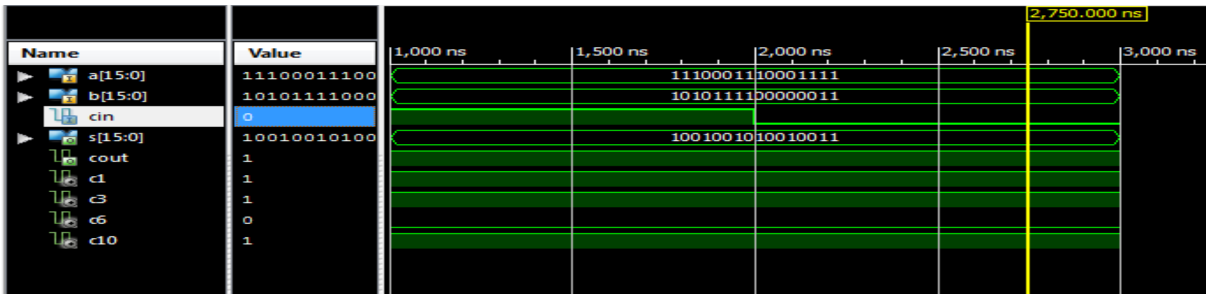


Fig 7: 64-BIT CSLA WITH D-LATCH ARCHITECTURE

This proposed architecture of 16 bit CSLA using D latch consists of five clusters of bit word size starting from n bit RCA and D latch later on 3b, 4b, 5b, 6b word size simultaneously .Instead of using two separate adders in regular CSLA, in this method only one adder is used to reduce area, power consumption and delay and each of two addition is performed in one clock cycle. In this 16-b adder the LSB is ripple carry adder which is 2 bit. The upper half of adder i.e. MSB which is 14-b wide works on the principal of clock. Whenever clock goes high the addition for carry input zero is performed and when clock goes low carry is assumed to be zero and addition is stored in adder itself. Carry out from the previous stage i.e., least significant bit adder is used as control signal for multiplexer to select final output carry and sum of the 16- bit adder. If the actual carry input is one, then computed sum and carry latch is accessed and for carry input zero MSB adder is accessed. Cout is the output carry.

IV. SIMULATION AND RESULTS



Name	Value
a[15:0]	11100011100
b[15:0]	10101111000
cin	0
s[15:0]	10010010100
cout	1
c1	1
c3	1
c6	0
c10	1

Fig 8:Regular SQRT CSLA Simulation

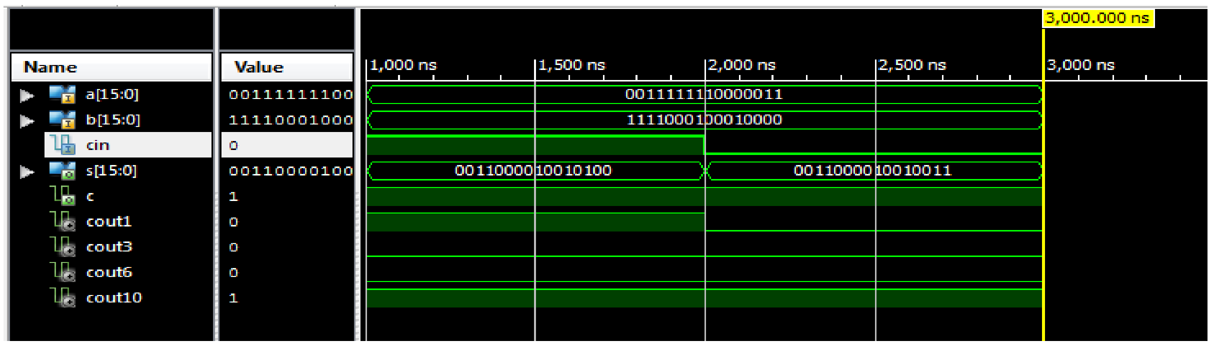


Fig 9: CSLA using BEC

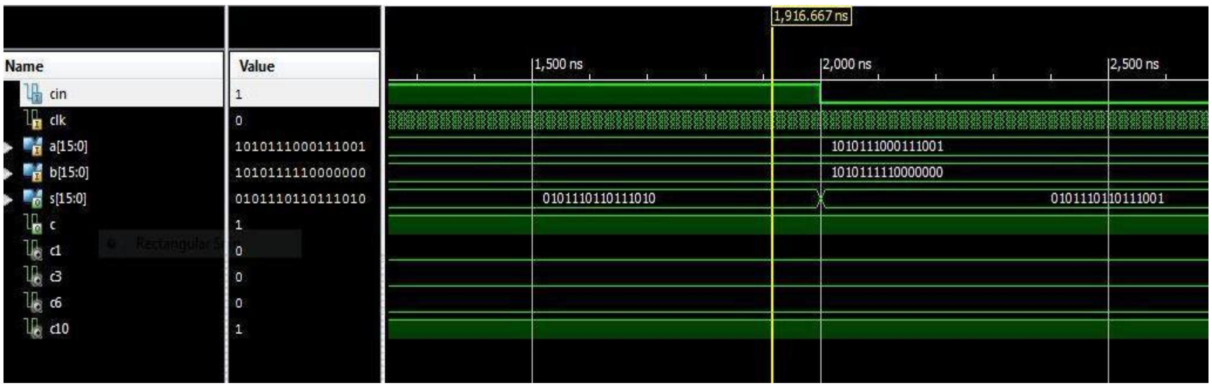


Fig 10:Modified CSLA using D-Latch

Comparison table:

	16 bit RCA	16 bit BEC	16 bit D LATCH	32 bit RCA	32 bit BEC	32 bit D LATCH	64bit RCA	64 bit BEC	64 bit D LATCH
Number of 4 input LUTs used	43	48	32	102	101	134	204	201	285
Available LUTs	9312	9312	9312	9312	9312	9312	9312	9312	9312
DELAY ns	14.362	14.73	12.211	16.117	21.756	16.506	27.539	37.772	18.873
Total supply power mw	76	76	76	76	76	76	76	76	76
Power Utilized mw	0.351	0.392	0.039	0.832	0.823	1.092	1.663	1.638	2.3235

V. CONCLUSION

In the proposed scheme, CSLA is implemented using D-latch in which, the carry select (CS) operation is scheduled before the calculation of final sum, which is different from the conventional approach. Bit patterns of two anticipating carry words (corresponding to cin = 0 and 1) and fixed cin bits are used for logic optimization of Carry selection. A delay efficient CSLA design is obtained using D-latch logic unit. The proposed CSLA design involves significantly less **Delay** than the recently proposed BEC-based CSLA

REFERENCES

[1] K. K. Parhi, *VLSI Digital Signal Processing*. New York, NY, USA: Wiley, 1998.

[2] A. P. Chandrakasan, N. Verma, and D. C. Daly, “Ultralow-power electron-ics for biomedical applications,” *Annu. Rev. Biomed. Eng.*, vol. 10, pp. 247– 274, Aug. 2008.

[3] O. J. Bedrij, “Carry-select adder,” *IRE Trans. Electron. Comput.*, vol. EC-11, no. 3, pp. 340–344, Jun. 1962.

[4] Y. Kim and L.-S. Kim, “64-bit carry-select adder with reduced area,” *Electron. Lett.*, vol. 37, no. 10, pp. 614–615, May 2001.

[5] Y. He, C. H. Chang, and J. Gu, “An area-efficient 64-bit square root carry-select adder for low power application,” in *Proc. IEEE Int. Symp. Circuits Syst.*, 2005, vol. 4, pp. 4082–4085.

[6] B. Ramkumar and H. M. Kittur, “Low-power and area-efficient carry-select adder,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 2,

pp. 371–375, Feb. 2012.

[7] I.-C. Wey, C.-C. Ho, Y.-S. Lin, and C. C. Peng, "An area-efficient carry select adder design by sharing the common Boolean logic term," in *Proc. IMECS*, 2012, pp. 1–4.

[8] S. Manju and V. Sornagopal, "An efficient SQRT architecture of carry select adder design by common Boolean logic," in *Proc. VLSI ICEVENT*, 2013,
pp. 1–5.

[9] B. Parhami, *Computer Arithmetic: Algorithms and Hardware Designs*, 2nd ed. New York, NY, USA: Oxford Univ. Press, 2010.

[10] Basant Kumar Mohanty and Sujit Kumar Patel, "Area–Delay–Power Efficient Carry-Select Adder",IEEE Transaction On CircuitAnd System-*i:*Express Briefs, Vol 61, No 6, June 2014.