Area and Delay Efficient Carry Select Adder Using Carry Prediction Approach

Satinder Singh Mohar
Department of Electronics and Communication Engineering,
Punjabi University, Patiala, Punjab, India

Manjit Singh Bhamrah
Department of Electronics and Communication Engineering,
Punjabi University, Patiala, Punjab, India

Abstract: Various approaches and designs of digital arithmetic units have been proposed in the recent years and their merits and demerits have been compared on various performance parameters. In modern day digital systems, Arithmetic and Logic Unit operations on reconfigurable devices are the most important area of research. In this paper, a novel design for Carry Select Adder is proposed. In the design, the half adder of both the ripple carry adders is shared and an initial sum and carry is generated. The carry input module generates the 'carry' values and are passed through multiplexer in first method or xor-ed with the sum in the second method. The novel approach is then compared with the existing approaches in the literature and results show that the proposed approach outperforms all the previous approaches on the basis of delay and area complexity. The proposed approach shows significant improvement in both number of slices used and the combinational path delay.

Keywords: Carry Select Adder, Ripple Carry Adder, Delay, Area.

I. INTRODUCTION

The major block of many processor architectures such as digital signal processors and general purpose microprocessors is Arithmetic and logic unit (ALU). The most commonly functions of DSP such as filtering and convolution etc. are accomplished by ALU. In microprocessors and RISC processors, it is observed that ALU perform addition on 88.4% of total instructions and 71% of the total power consumption is contributed by high active register banks [6].With the rapid increase in mobile industry, the computation speed is not only major concern in design of digital system but the smaller area and low power consumption also becomes the major concern of the system design. In design of digital system reducing the area and power consumption will increase the portability and battery life of the device [1].

Various adders can be used to perform addition. The Ripple Carry Adder (RCA) has smaller area and high propagation delay. Carry Look-Ahead Adder (CLA) has larger area and small propagation delay. The speed of addition in digital adder is limited by time required for carry to propagate through the adder. The sum for each present stage full adder is calculated until the previous stage full adder calculations are performed and carry output of previous stage full adder is given as carry input to the next stage full adder. The delay of Carry select adder is less as compared to CLA and RCA. CSLA consist of two RCA and 2:1 multiplexer. One RCA perform the calculation with assumption $C_{in} = 0$ and other RCA the calculation with assumption $C_{in} = 1$. The correct sum and carry are selected with the help of input carry which is used as select line to 2:1 mux [1].

This paper is structured as follows: In section II, previous work is explained. Section III, describes the proposed CSA. In section IV, Simulation results and comparison of proposed adder against existing architectures are analysed. Lastly, section V briefs the conclusions.

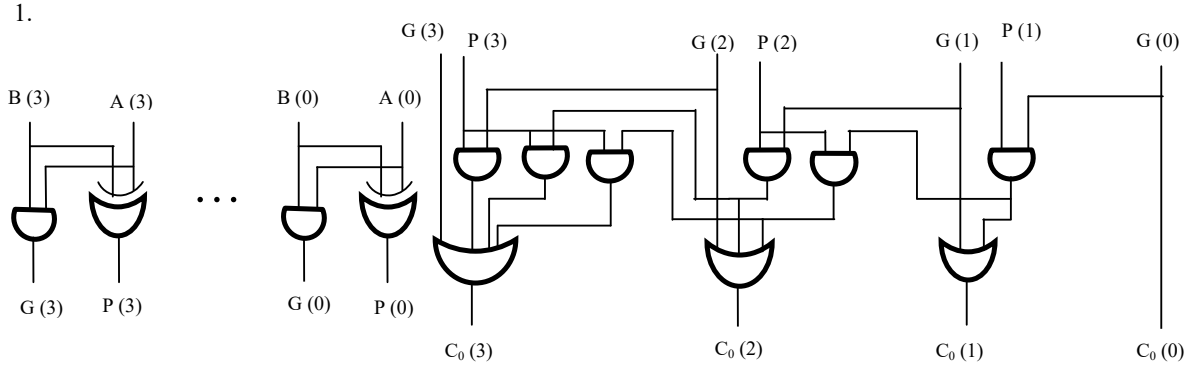
II. PREVIOUS WORK

Carry select adder uses multiple pairs of Ripple carry adders to generate partial sum and carry so Conventional CSLA is not area efficient. In order to reduce the power consumption and area various CSA architectures were designed. Sarabdeep Singh et al. introduced a modified carry select adder in which modification was done at gate-level. The RCA with C_{in}=1 was replaced by Binary to Excees-1 Converter in conventional CSA. The area and power of modified adder was reduced but delay of modified CSA was increased [1]. Samiappa

Saktikumaran et al proposed a fast CSA in which second RCA block was replaced with basic unit. The basic unit consists of incrementer and input to basic unit was 4 bit binary data. The output carry was generated in parallel using parallel chain of AND gates that reduced the propagation delay as compared to CCSA [2]. I. Chyn et al presented an area efficient CSA by sharing the CBL. The duplicated adder cells in the conventional uniform CSA were replaced by Common Boolean logic. The common Boolean logic had reduced the transistor count and power consumption and power delay product but delay of CSA_CBL was almost equivalent to that of RCA [3]. In order to reduce this problem, S. Manju et al introduced SQRT CSA_CBL. This adder required less number of gates which reduced delay, area and power of modified adder [4]. B. Ramkumar et al replaced RCA with C_{in}=1 with Binary to Excess-1 Converter in SQRT CSA. The modified CSA had larger delay as compared to regular SQRT CSA but area and power consumption was reduced [5]. Kore Sagar Dattatraya et al used one RCA and n-bit binary adder which was used as BEC-1. The OR structure along with n-bit binary adder was result in generating BEC-1 operation. The number of logic gates was reduced in modified CSA and it consumed less power but delay was increased [6]. G.Karthik Reddy et al used one RCA and D-Latch. To replace n bit RCA n D-Latches with enable pin as a clock were required and latches were used to store the information. The performance of modified adder was better than SQRT CSA since it consumed less power [7]. V. Kokilavani et al had analysed the performance of seven different types of 32 bit and 64 bit adders: CCSA, CSA-BEC, CSA-CBL, CSA_CLA, CSA-BEC_CLA, CSA_SCBCLA, CSA-BEC_SCBCLA in terms of area and delay [8]. R.P.P.Singh et al proposed hybrid carry skip adder in which RCA was replaced by CLA for the generation of sum and carry for the next block. The power and area of modified carry skip adder was less as compared with other basic adders but it required more number of CLB's [9]. Shivani Parmar et al replaced RCA by BEC in conventional carry select adder. The area and power consumption of modified architecture was less as compared to CSA but delay was increased in modified CSA [10]. Basant Kumar Mohanty et al presented novel logic formulation for CSLA. In the proposed adder calculation of sum for C_{in}=1 was avoided in the sum and carry generator unit. The proposed adder used less logic devices and delay of output carry was small [11]. Vinod Kumar Naik et al presented modified adder in which all unnecessary operations was removed. In the purposed adder final carry was selected before the final sum was computed. The area utilization and power consumption of proposed adder was less as compared to CSA [12].

III. PROPOSED METHODOLOGY

In the proposed approach, the carry select adder is designed using the carry generation technique and further implemented using two different methods of carry and sum selection phase. The carry select adder is divided into different sub blocks. In the first unit, simple half adder logic is implemented. This block is the common block for both the carry '1' and '0' calculation. Both techniques along with their different blocks is shown in figure 1 and 2. In figure 1 the first method (method 1) is shown, in this method firstly after the half adder block carry generation block generates the carry and then multiplexer is used to select the carry. After the selection of carry, sum is xor-ed with the carry to generate the final sum. In second method (method 2) after the carry generation part firstly the sum is xor-ed with the carry and then it is selected using multiplexer with carry input as the select line. Figure 2 shows the variation in method of sum generation after common logic used in method 1.



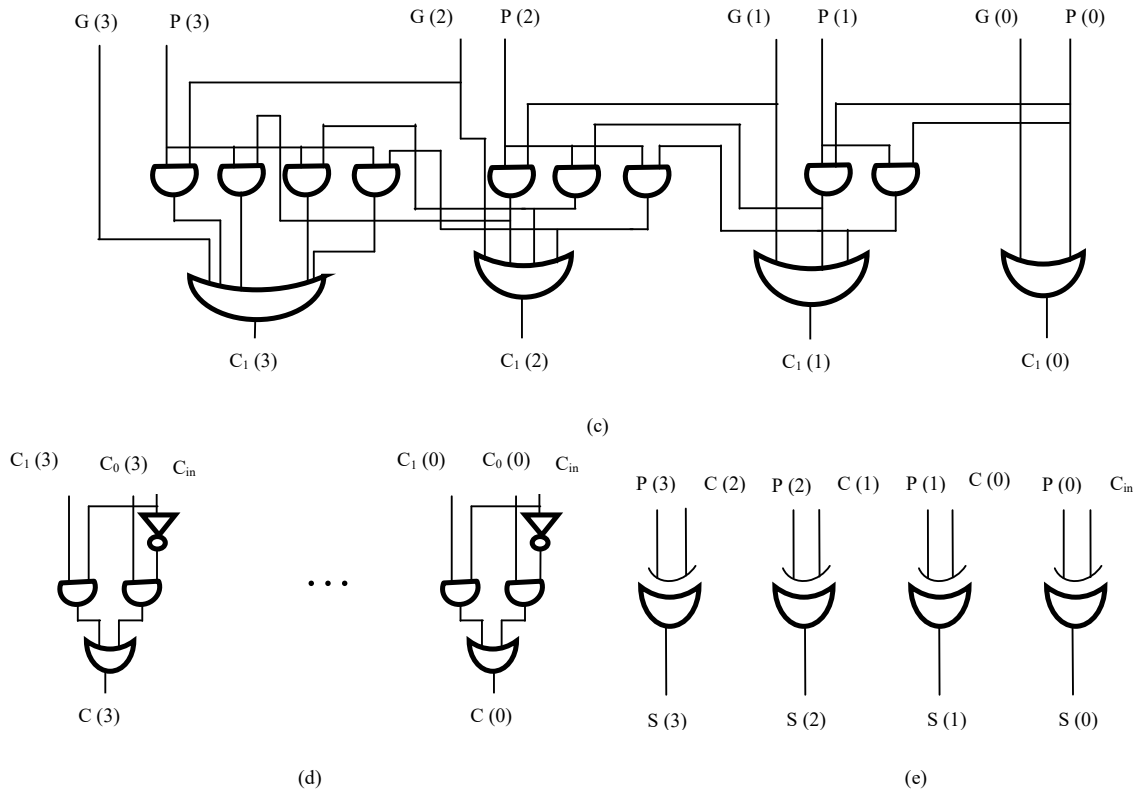


Figure 1: Method 1 (a) Half Adder Unit (b) Carry Generation Unit for $C_{in} = 0$ (c) Carry Generation Unit for $C_{in} = 1$ (d) Multiplexer Unit (e) Sum Generation Unit

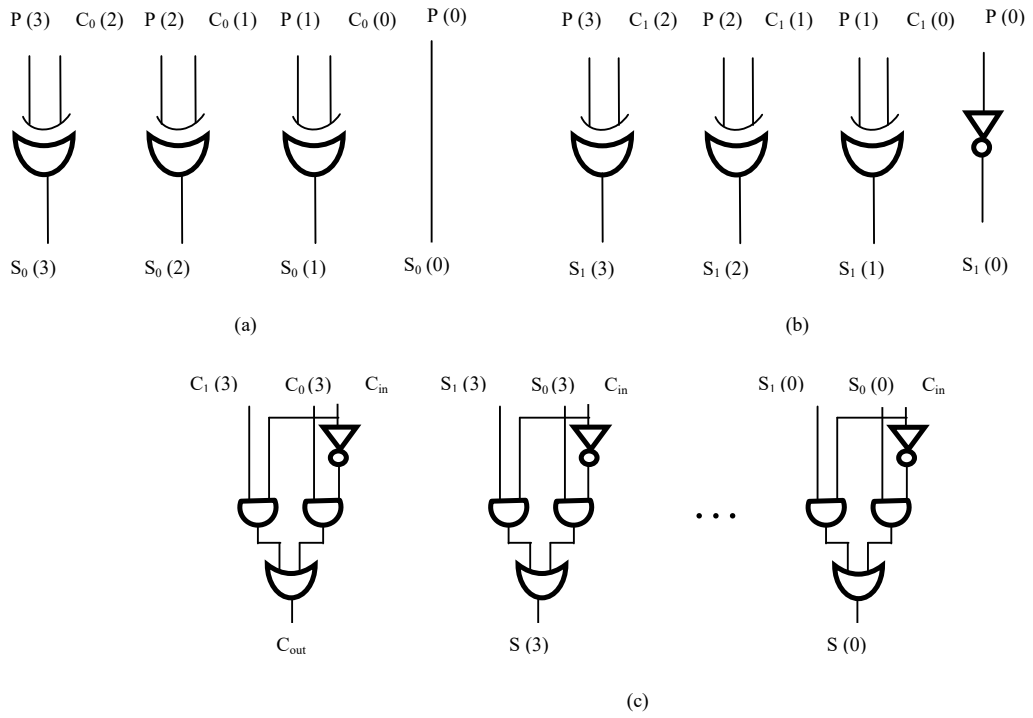


Figure2: Method 2 (a) Sum Generation Unit for Carry = 0 (b) Sum Generation Unit for Carry = 1 (c) Multiplexer Unit.

The adder architecture is divided into various sub blocks which are shown in figure 1(a) – (e) for method 1 and from figure 1(a) – 1(c) and 2(a) – 2(c) for method 2. In the first block the simple half adder is used which computes the sum and carry value (shown in figure 1(a) P_i is used for sum and G_i is used for carry output and i in the approach is taken from 0 *to* $n-1$). The Half adder sum and carry output generation equations are

$$P(i) = A(i) \; xor \; B(i) \quad (1)$$
$$G(i) = A(i) \; and \; B(i) \quad (2)$$

In the second block, Carry and Sum generation are implemented which takes the output from the first unit and generates the sum and carry values (C_0,S_0 and C_1,S_1 used for carry and sum respectively) for input carry '0' and '1'. The carry is generated in the second block and used in method 1 and method 2 both while the sum generated is used in second method only. The output of the sum and carry are generated using

$$C_0(0) = G(0) \quad (3)$$
$$C_0(1) = G(1) \; or \; (P(1) \; and \; G(0)) \quad (4)$$
$$C_0(2) = G(2) \; or \; (P(2) \; and \; G(1)) \; or \; (P(2) \; and \; P(1) \; and \; G(0)) \quad (5)$$
$$C_0(3) = G(3) \; or \; (P(3) \; and \; G(2)) \; or \; (P(3) \; and \; P(2) \; and \; G(1)) \; or \; (P(3) \; and \; P(2) \; and \; P(1) \; and \; G(0)) \quad (6)$$

$$C_1(0) = G(0) \; or \; P(0) \quad (7)$$
$$C_1(1) = G(1) \; or \; (P(1) \; and \; P(0)) \; or \; (P(1) \; and \; G(0)) \quad (8)$$
$$C_1(2) = G(2) \; or \; (P(2) \; and \; G(1)) \; or \; (P(2) \; and \; P(1) \; and \; G(0)) \; or \; (P(2) \; and \; P(1) \; and \; P(0)) \quad (9)$$
$$C_1(3) = G(3) \; or \; (P(3) \; and \; G(2)) \; or \; (P(3) \; and \; P(2) \; and \; G(1)) \; or \; (P(3) \; and \; P(2) \; and \; P(1) \; and \; G(0)) \; or \; (P(3) \; and \; P(2) \; and \; P(1) \; and \; P(0)) \quad (10)$$

$$S_0(0) = P(0) \quad (11)$$
$$S_0(1) = P(1) \; xor \; C_0(0) \quad (12)$$
$$S_0(2) = P(2) \; xor \; C_0(1) \quad (13)$$
$$S_0(3) = P(3) \; xor \; C_0(2) \quad (14)$$

$$S_1(0) = not \; P(0) \quad (15)$$
$$S_1(1) = P(1) \; xor \; C_1(0) \quad (16)$$
$$S_1(2) = P(2) \; xor \; C_1(1) \quad (17)$$
$$S_1(3) = P(3) \; xor \; C_1(2) \quad (18)$$

Finally in the block shown in figure 1(d) for method 1 the mux architecture is implemented which selects one of the carry outputs from the second and third block. The final carry in method 1 is generated using the equation 19 and final sum is generated using equation 20 – 23 and C_{out} is calculated using equation 24. In equation 25 and 26 the final sum and final carry is generated using method 2.

$$C(i) = (C_0(i) \; and \; not \; C_{in}) \; or \; (C_1(i) \; and \; C_{in}) \quad (19)$$
$$S(0) = P(0) \; xor \; C_{in} \quad (20)$$
$$S(1) = P(1) \; xor \; C(0) \quad (21)$$
$$S(2) = P(2) \; xor \; C(1) \quad (22)$$
$$S(3) = P(3) \; xor \; C(2) \quad (23)$$

$$C_{out} = C(3) \quad (24)$$

$$S(i) = (S_0(i) \text{ and not } C_{in}) \text{ or } (S_1(i) \text{ and } C_{in}) \quad (25)$$

$$C_{out} = (C_0(3) \text{ and not } C_{in}) \text{ or } (C_1(3) \text{ and } C_{in}) \quad (26)$$

Proposed CSA –

In Proposed CSA architecture both method 1 and method 2 have two variations considered as case1 and case 2. The variation is in the form of first block which is either proposed 4 bit CSA or Ripple Carry Adder (RCA). Figure 3 shows the architecture of 16 bit CSA for case 1 using proposed 4 bit CSA for both method 1 and method 2. Figure 4 shows the architecture of 16 bit CSA for case 2 using both Proposed 4 bit CSA and 4 bit RCA.

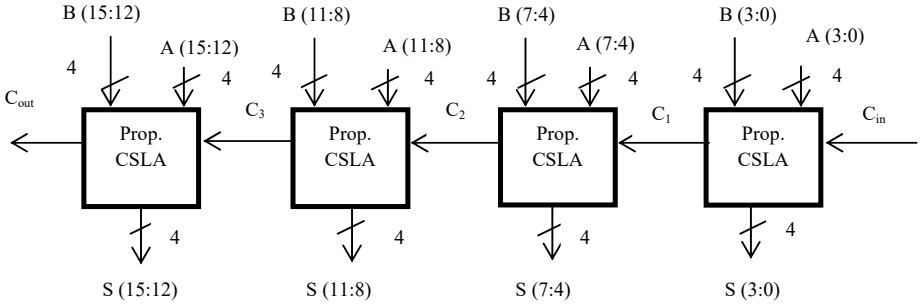


Figure 3: Proposed 16 bit CSA using case 1 approach

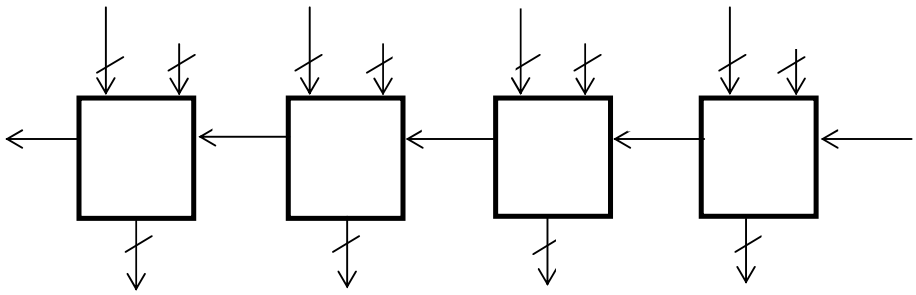


Figure 4: Proposed 16 bit CSA using case 2 approach

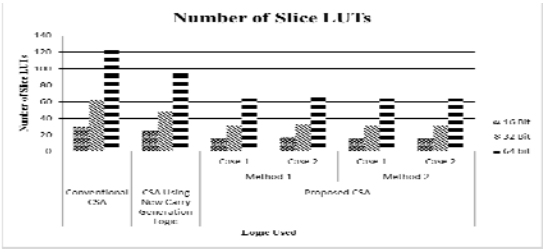
IV. RESULTS AND DISCUSSIONS

The proposed Carry Select Adder is designed using the Xilinx ISE 14.2 system design environment using the VHDL language. The FPGA device used for the synthesis and simulation is Virtex 5. The design strategy used for method 1 is Area Optimization and method 2 is Power Optimization with physical synthesis. Table 1 shown below compares various architecture and methodologies of the design of Carry Select Adder for different number of bits. Various approaches like Conventional CSA using Ripple Carry Adder, CSA design using the new Carry generation logic and Proposed CSA are compared on the basis of performance parameters like Number of Slices, Number of Slice LUT's, Delay, Area delay Product and Maximum Attainable Frequency for 16 bit, 32 bit and 64 bits.

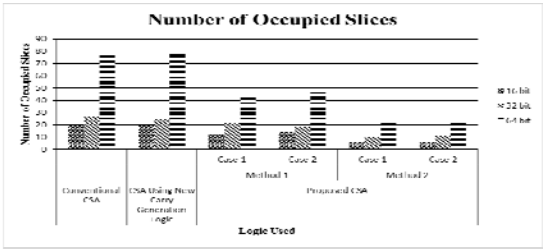
Table 1: Performance Comparison of Various Approaches

Word-Size	Adder			No. of Slice LUTs	No. of Occupied Slices	Delay(ns)	Area Delay Product (ns)	Max. Attainable Frequency(MHz)
16-bit	Conventional CSA			30	20	8.364	250.92	119.56
	CSA Using New Carry Generation Logic			25	20	8.464	211.6	118.14
	Proposed CSA	Method 1	Case 1	16	12	8.106	129.69	123.36
			Case 2	17	14	7.024	119.40	142.36
		Method 2	Case 1	16	6	8.242	131.87	121.32
			Case 2	15	6	7.386	110.79	135.39
32-bit	Conventional CSA			62	27	13.699	849.33	72.99
	CSA Using New Carry Generation Logic			49	25	13.120	642.88	76.21
	Proposed CSA	Method 1	Case 1	32	22	12.762	408.38	78.35
			Case 2	33	19	11.680	385.44	85.61
		Method 2	Case 1	32	10	11.915	381.28	83.92
			Case 2	32	11	11.059	353.88	90.42
64-bit	Conventional CSA			126	77	24.370	3070.62	41.03
	CSA Using New Carry Generation Logic			97	78	22.431	2175.80	44.58
	Proposed CSA	Method 1	Case 1	64	43	22.073	1412.67	45.30
			Case 2	65	48	20.991	1364.41	47.63
		Method 2	Case 1	64	24	19.261	1232.70	51.91
			Case 2	64	22	18.405	1177.92	54.33

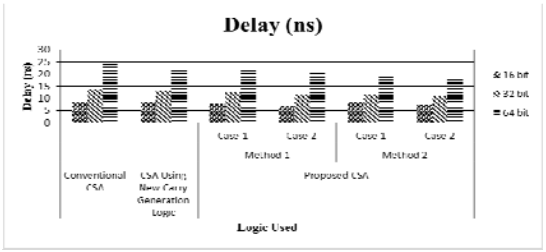
From the table it is evident that the proposed CSA is both area and delay efficient for various numbers of bits. The comparison results shown in the above table are plotted for 16, 32 and 64 bits and are shown in figure 5(a) – (e). In figure 5(a) and 5(b) the comparison of the number of Slice LUTs and number of occupied Slices for different architectures is plotted. In figure 5(c), 5(d) and 5(e) the combinational path delay and area delay product, maximum attainable frequency is plotted respectively. The graphs show a quantitative analysis of the proposed architecture with the approaches available in the literature.



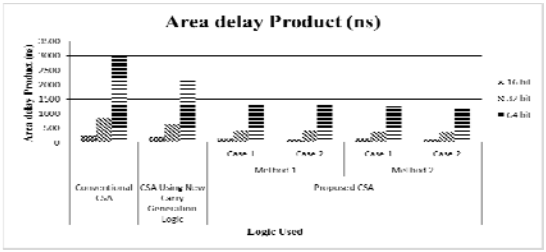
(a)



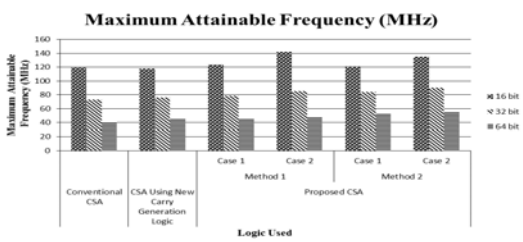
(b)



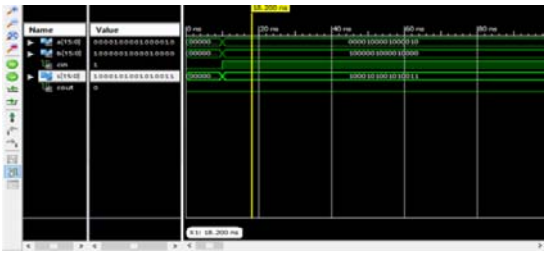
(c)



(d)



(e)



(f)

Figure 5. (a) Number of Slice LUTs vs CSA design Logic Used (b) Number of Occupied Slices vs CSA design Logic Used (c) Combinational Path Delay vs CSA Design Logic Used (d) Area Delay Product vs CSA Design Logic Used (e) Max. Attainable Frequency vs CSA Design Logic Used (f) Simulation Waveform of the Proposed Approach

In figure 5(f) the simulation waveforms for the proposed methodology is shown and it verifies the working of the proposed logic architecture. Results also show that the number of occupied slices for 64 bit proposed CSA are decreased by 44% and 68% for method 1 and method 2 (case 1) respectively and 37% and 71% for method 1 and method 2 (case 2) respectively as compared to conventional CSA. The maximum combinational path delay for the proposed approach for 64 bit CSA is decreased by 2.297 and 5.109 ns for method 1 and method 2 (case1) and by 3.379 and 5.965 ns for method 1 and method 2 (case2) respectively as compared to the conventional CSA and the area delay product for the 64 bit proposed CSA is decreased by 53%, 59%, 55% and 61% for method 1 and method 2 (case1 and 2) respectively than that of regular CSA.

V. CONCLUSION

Arithmetic operations in the digital systems are one of the most important issues of research in recent years. The fast and area efficient operations on FPGA are the necessity of each and every digital system implementing ALU or Signal Processing unit operations. In this work, a novel approach for the calculation of sum is implemented for different sizes of the input operands. The approach uses resource sharing among various building blocks of the adder. The results also justifies the proposed approach and the delay is reduced by 5.965 ns compared to the conventional CSA, while the area utilization in terms of LUT's and slices is also reduced by 71% than that of regular CSA.

REFERENCES

[1] Sarabdeep Singh and Dilip Kumar, "Design of Area and Power Efficient Modified Carry Select Adder," International Journal of Computer Applications, vol. 33, no. 3, pp. 14-18, November 2011.

[2] Samiappa Saktikumaran, S. Salivahanan, V.S.Kanchana Bhaaskaran, V.Kavinilavu, B.Brinda and C.Vinoth, "A Very Fast and Low Power Carry Select Adder Circuit," in Proc. of 3rd IEEE International Conference on Electronics Computer Technology (ICECT), vol. 1, pp. 273-276, 2011.

[3] I-Chyn Wey, Cheng-Chen Ho, Yi-Sheng Lin and Chien-Chang Peng, "An Area Efficient Carry Select Adder Design by Sharing the Common Boolean Logic," in Proceedings of the International MultiConference of Engineers and Computer Scientists 2012, IMECS 2012, pp. 1-4, March 2012.

[4] S.Manju and V.Sonagopal, "An Efficient SQRT Architecture of Carry Select Adder Design by Common Boolean Logic," in Proceeding of IEEE International Conference on Emerging Trends in VLSI, Embedded System, Nano Electronics and Telecommunication System (ICEVENT), 2013, pp. 1-5.

[5] B. Ramkumar and Harish M Kittur, "Low-Power and Area-Efficient Carry Select Adder," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 20, no. 2, pp. 371-375, February 2012.

[6] Kore Sagar Dattatraya and V.S.Bhaaskaran, "Modified Carry Select Adder using Binary Adder as BEC-1," European Journal of Scientific Research, vol. 103, no. 1, pp. 156-164, 2013.

[7] G Karthik REDDY, D.Sharat Babu Rao, "A Comparative study on Low-Power and High Speed Carry Select Adder," in Proceeding of 9th IEEE International Conference on Intelligent Systems and Control (ISCO), pp. 1-7, 2015.

[8] V. Kokilavani, P. Balasubramanian, H. R. Arabnia,"FPGA realization of Hybrid Carry Select-Cum-Section-Carry Based Carry Lookahead Adders," in Proceeding 12th International Conference on Embedded Systems and Applications, pp. 81-85, Oct. 2014.

[9] R. P. P. Singh, Parveen Kumar, Balwinder Singh, "Performance Analysis of Fast Adders Using VHDL," in Proceeding of IEEE International Conference on Advances in Recent Technologies in Communication and Computing, pp. 189-193, 2009.

[10] Shivani Parmar, Kirat Pal Singh, "Design of high speed hybrid carry select adder," in Proceeding of IEEE 3rd International Conference on Advance Computing (IACC), pp. 1656-1663, 2013.

[11] Basant Kumar Mohanty, and Sujit Kumar Patel, "Area-Delay- Power Efficient Carry-Select Adder," IEEE transaction on circuits and systems,VOL.6I, NO.6,JUNE 2014.

[12] Vinod Kumar Naik, Mohammed Aneesh.Y, "Design of carry select adder for low-power and high speed VLSI applications," in Proceeding of IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), pp. 1-4, 2015.