

Area-Delay-Power Efficient Carry-Select Adder

Shruthi Nataraj¹, Karthik .L²

¹M-Tech Student, Karavali Institute of Technology, Neermarga, Mangalore, Karnataka

²Assistant professor , Karavali Institute of Technology, Neermarga, Mangalore, Karnataka

Abstract: This paper proposes on the logic operations in conventional carry select adder (CSLA) and binary to excess-1 converter (BEC)-based CSLA to study the data dependency and to identify redundant logic operations. The new logic formulations have been proposed by eliminating all the redundant logic operations present in conventional CSLA. In the proposed scheme, the carry-select operation is scheduled before calculation of the final-sum. Anticipating carry-words (corresponding to $C_{in}=0$ and 1) and fixed C_{in} bits are the two Bit-patterns used for logic optimization of carry-select and general units. An efficient CSLA design is obtained using optimized logic units. The proposed CSLA design involves significantly less area and delay than the proposed BEC-based CSLA. Due to small carry-output delay, the proposed CSLA design is good for square root (SQRT)-CSLA. As per the theoretical estimation the proposed SQRT-CSLA involves nearly 35% less area-delay-product (ADP) than the BEC-based SQRT-CSLA which is the best amongst the existing SQRT-CSLA designs on average for different bit-widths. FPGA synthesis result shows that, the BEC-based SQRT-CSLA design involves more Area-Delay Product and consumes more energy than the proposed SQRT-CSLA on average for different bit-widths.

Keywords: Adder, BEC, Arithmetic unit, Area-Delay Product, MSB, Low power design.

1. Introduction

High-speed data path logic systems are one of the most substantial areas of research in VLSI system design. High performance, low power and area-efficient systems are increasingly used in portable devices. A complex digital signal processing (DSP) system involves several adders. An adder is the main component of an arithmetic unit. A ripple carry adder 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 used to reduce the CPD of adders.

A conventional carry select adder (CLSA) is an RCA-RCA configuration that generates a pair of sum words and output-carry bits corresponding anticipated input-carry ($C_{in}=0$ and 1) and selects one out of each pair for final-sum and final-output-carry.

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.

The CSLA is not area efficient because it uses multiple pairs of Ripple Carry Adders (RCA) to generate partial sum and carry.

The basic idea of the proposed work is using n-bit binary to excess-1code converters (BEC) to improve the speed of addition. The proposed 16, 32 and 64-bit adders are compared with the conventional fast adders such as (CSA) and (CLA). The performance of the CSA can be improved by using BEC logic through customizing design and layout. The decrease in carry propagation delay will result in major enhancement of the speed of the adder and multiplier. CSL is implemented with dual ripple-carry adder (RCA) with the carry-in of 0 and 1, respectively.

The final carry propagation adder (CPA) structure of many adders constitutes high carry propagation delay and this delay reduces the overall performance of the DSP processor.

Based on this approach a 16, 32 and 64-bit adder architecture has been developed and compared with conventional fast adder architectures. This work identifies the performance of proposed designs in terms of delay-area-power through custom design and layout in process technology. The result analysis shows that the proposed architectures have better performance in reduction of carry propagation delay than contemporary architectures.

2. Block Diagram of Conventional CSLA and BEC-based CSLA

The main objective is to identify redundant logic operations and data dependence.

The CLSA has two units: 1) the sum and carry generator unit (SCG) and 2) the sum and carry selection unit. Various logic designs have been suggested for efficient implementation.

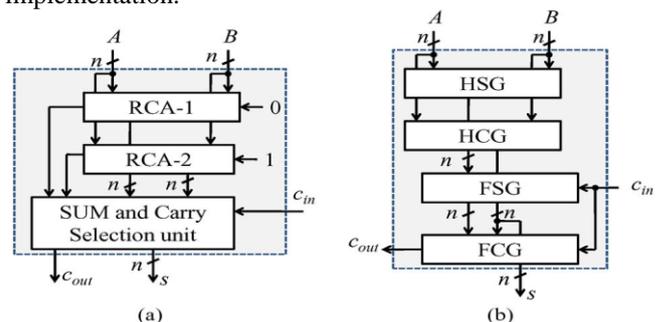


Figure 1: (a) Conventional CSLA (b) Logical operations of RCA

Block diagram of BEC-Based CSLA

BEC-based CSLA provides the best area-delay-power efficiency among various CSLA's

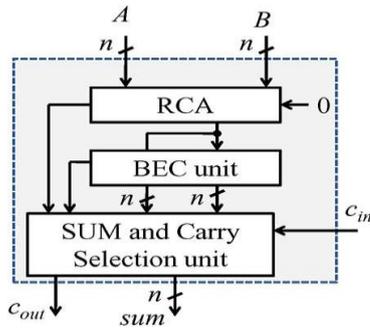


Figure 2: BEC-Based CSLA

3. Module's

- Proposed CS Adder Design
- HSG Unit
- CG0 & CG1 Unit
- CS Unit
- FSG Unit

4. Module Description

4.1) Proposed CS Adder Design.

The proposed CSLA is based on the logic formulation and its structure is shown in Fig. 3. It consists of one HSG unit, one FSG unit, one CG unit, and one CS unit. The CG unit is composed of two CGs (CG0 and CG1) corresponding to input-carry '0' and '1'. The HSG receives two n-bit operands (A and B) and generate half-sum word s0 and half-carry word c0 of width n bits each. Both CG0 and CG1 receive s0 and c0 from the HSG unit and generate two n-bit full-carry words c01 and c11 corresponding to input-carry '0' and '1', respectively.

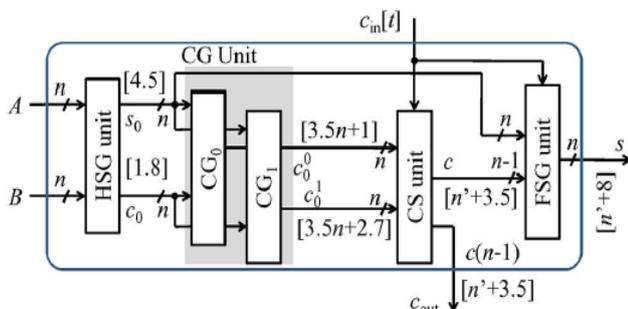


Figure 3: Proposed CSLA Design.

4.2) HSG Unit

The logic diagram of the HSG unit is shown in figure.4. It consist of Combinational Logical AND Gate and OR Gate.

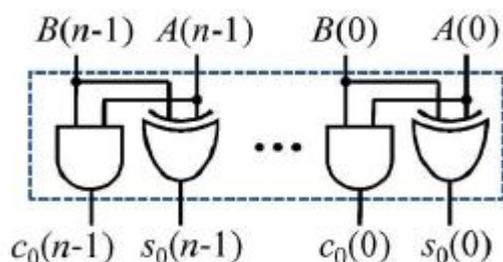


Figure 4: HSG Unit

4.3) CG0 & CG1 Unit

The logic circuits of CG0 and CG1 are optimized to take advantage of the fixed input-carry bits. The optimized designs of CG0 and CG1 are shown in Figures. It consist of logical or and logical and gates.

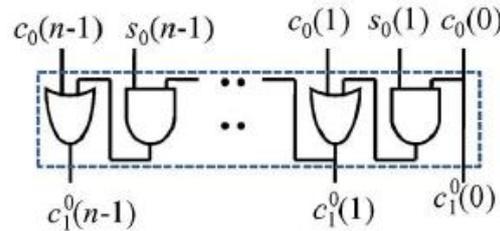


Figure 5(a): CG0 & CG1 Unit

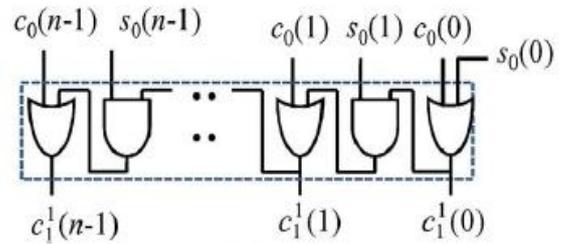


Figure 5(b): CG0 & CG1 Unit

4.4) CS Unit

CS Unit is referred as Carry Select unit and it consist of combination of logical OR and Logical AND gates. The CS unit selects one final carry word from the two carry words available at its input line using the control signal cin. It selects c01 when cin = 0; otherwise, it selects c11. The LSB of s0 is XORed with cin to obtain the LSB of s.

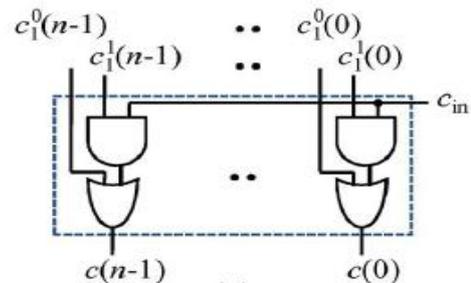


Figure 6: CS unit

4.5) FSG Unit

FSG Unit Referred as final-sum generation unit. It consists of Combination of logical XOR gates. It performs Sum operation.

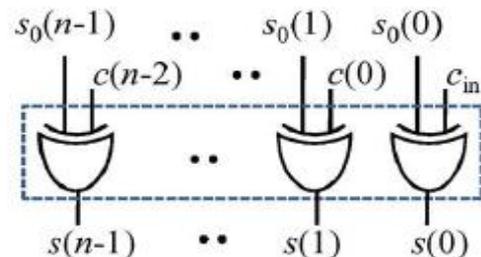


Figure 7: FSG Unit

5. Relevant Terminology

5.1 Field-Programmable Device (FPD)

A general term that refers to any type of integrated circuit used for implementing digital hardware, where the chip can be configured by the end user to realize different designs. Programming of such a device often involves placing the chip into a special programming unit, but some chips can also be configured “in-system”. Another name for FPDs is programmable logic devices (PLDs)

5.2 Programmable Logic Array (PLA)

A Programmable Logic Array (PLA) is a relatively small FPD that contains two levels of logic, an AND-plane and an OR-plane, where both levels are programmable.

5.3 Programmable Array Logic (PAL)

A Programmable Array Logic (PAL) is a relatively small FPD that has a programmable AND-plane followed by a fixed OR-plane.

5.4 Field-Programmable Gate Array (FPGA)

A Field-Programmable Gate Array is an FPD featuring a general structure that allows very high logic capacity. Whereas CPLDs feature logic resources with a wide number of inputs (AND planes), FPGAs offer more narrow logic resources. FPGAs also offer a higher ratio of flip-flops to logic resources than do CPLDs.

5.5 High-Capacity PLDs (HCPLD)

High-capacity PLDs: a single acronym that refers to both CPLDs and FPGAs. This term has been coined in trade literature for providing an easy way to refer to both types of devices. PAL is a trademark of Advanced Micro Devices.

5.6 Logic Block

A relatively small circuit block that is replicated in an array in an FPD. When a circuit is implemented in an FPD, it is first decomposed into smaller sub-circuits that can each be mapped into a logic block.

5.7 Logic Capacity

The amount of digital logic that can be mapped into a single FPD. This is usually measured in units of “equivalent number of gates in a traditional gate array”. In other words, the capacity of an FPD is measured by the size of gate array that it is comparable to. In simpler terms, logic capacity can be thought of as “number of 2-input NAND gates”.

5.8 Proposed Adder Design

The proposed CSLA is based on the logic formulations and its structures. It consists of one HSG unit, one FSG unit and one CS unit. The CG unit is composed of two CGs (CG₀ and CG₁) corresponding to input-carry ‘0’ and ‘1’. The HSG receives two n-bit operands (A and B) and generate half-sum

word s₀ and half carry word c₀ of width n bits each. Both CG₀ and CG₁ receive s₀ and c₀ from the HSG unit and generate two n-bit full carry words c₁⁰ and c₁¹ corresponding to input-carry ‘0’ and ‘1’, respectively.

The CS unit selects one final carry word from the two carry. Words available at its input line using the control signal c_{in}. It selects c₁⁰ when c_{in} = 0; otherwise, it selects c₁¹. The CS unit can be implemented using an n-bit 2-to-1 MUX. However, we find from the truth table of the CS unit that carry words c₁⁰ and c₁¹ follow a specific bit pattern. If c₁⁰(i) = ‘1’, then c₁¹(i) = 1, irrespective of s₀(i) and c(i), for 0 ≤ i ≤ n - 1. This feature is used for logic optimization of the CS unit. The optimized design of the CS unit is shown which is composed of n AND-OR gates. The final carry word c is obtained from the CS unit. The MSB of c is sent to output as c_{out}, and (n - 1) LSBs are XOR ed with (n - 1) MSBs of half-sum (s₀) in the FSG [to obtain (n - 1) MSBs of final-sum (s). The LSB of s₀ is XORED with c_{in} to obtain the LSB of s.

6. Tables - 1

Area-Delay of standard Cell Library Datasheet

	AND-gate	OR-gate
Area (um ²)	7.37	7.37
Delay (ps)	180	170

The area and delay of each design are calculated from the AOI gate counts (N_a, N_o, N_i), (n_a, n_o, n_i), and the cell details of Table I.

7. Result Analysis

The simulation results are carried out for Area-delay-power carry select adder to find out the area-delay. The simulation is carried out by Modelsim 6.4c as a simulator tool. The simulation result is shown as a snapshots.

7.1 Various Snapshots

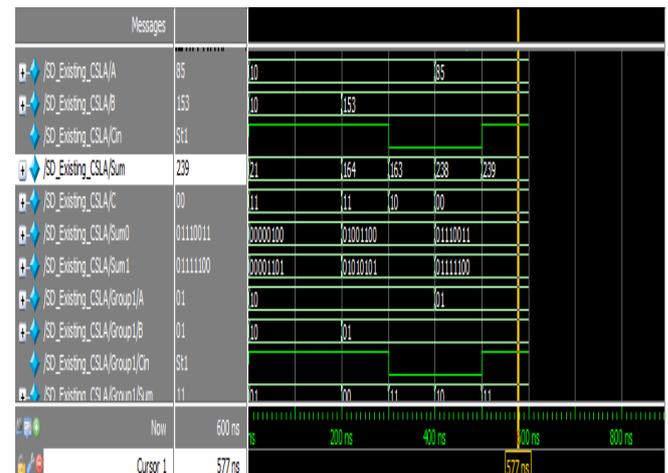


Figure 8(a): BEC-Based CSLA n=8

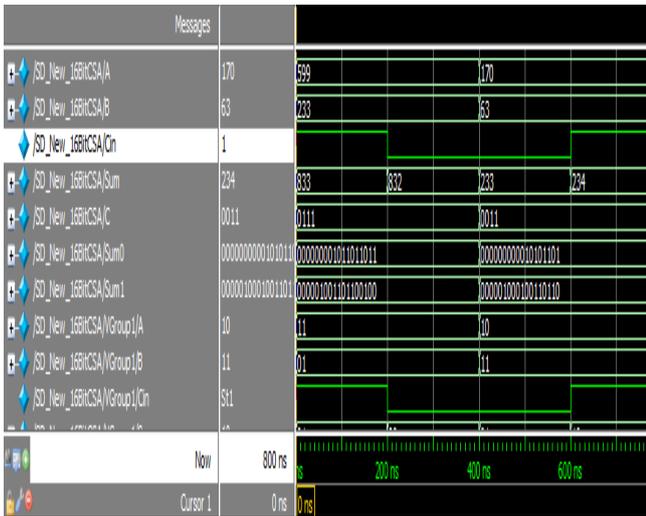


Figure 8(b): BEC-Based CSLA n=16

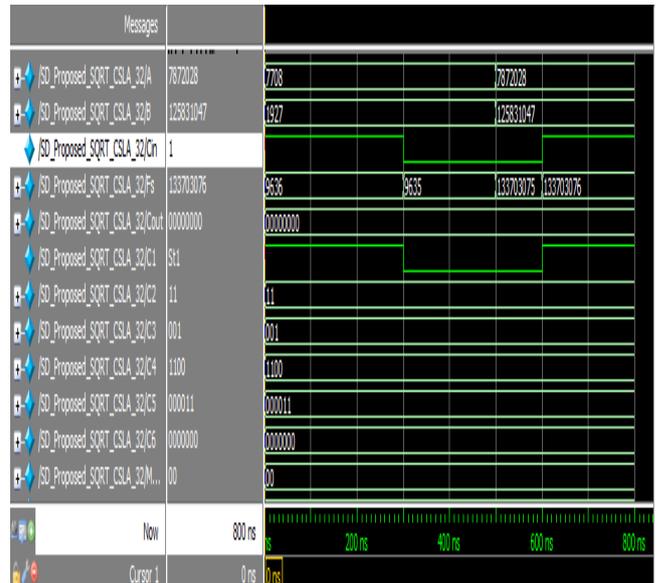


Figure 8(e): Proposed SQRT CSLA n=64

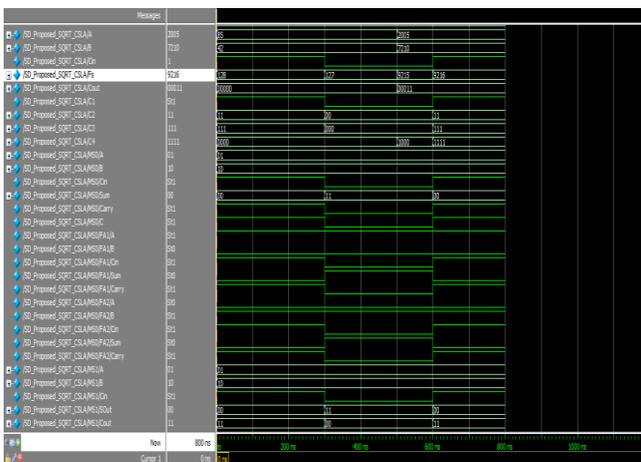


Figure 8(c): Proposed SQRT CSLA n=16

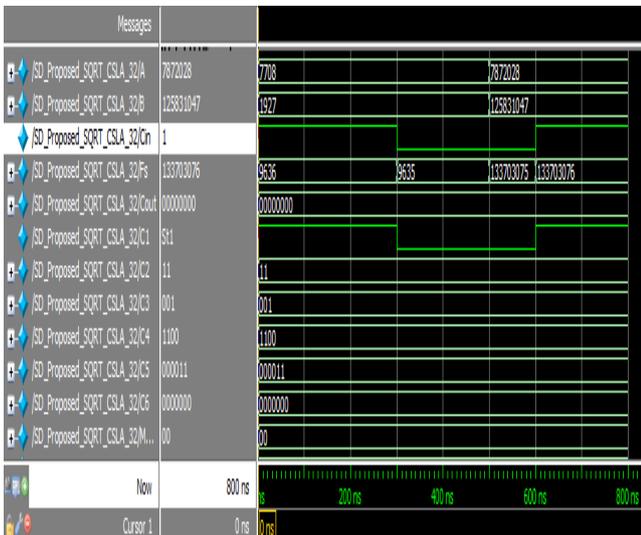


Figure 8(d): Proposed SQRT CSLA n=32

8. Device Utilization Summary

PROPBASEDCSLA Partition Summary				
No partition information was found.				
Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of 4 input LUTs	47	7,168	1%	
Logic Distribution				
Number of occupied Slices	28	3,584	1%	
Number of Slices containing only related logic	28	28	100%	
Number of Slices containing unrelated logic	0	28	0%	
Total Number of 4 input LUTs	47	7,168	1%	
Number of bonded IOBs	50	141	35%	
Total equivalent gate count for design	288			
Additional JTAG gate count for IOBs	2,400			

Figure 9(a): Area utilization for Existing

PROPBASEDCSLA Partition Summary				
No partition information was found.				
Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of 4 input LUTs	42	7,168	1%	
Logic Distribution				
Number of occupied Slices	25	3,584	1%	
Number of Slices containing only related logic	25	25	100%	
Number of Slices containing unrelated logic	0	25	0%	
Total Number of 4 input LUTs	42	7,168	1%	
Number of bonded IOBs	54	141	38%	
Total equivalent gate count for design	264			
Additional JTAG gate count for IOBs	2,592			

Figure 9(b): Area utilization for Proposed

9. Conclusion

The logic operations involved in the conventional and BEC-based CSLAs to study the data dependence and to identify redundant logic operations. The proposed CSLA design involves significantly less area and delay than the recently proposed BEC-based CSLA. Due to the small carry output delay, the proposed CSLA design is a good candidate for the SQR T adder. The FPGA synthesis result shows that the existing BEC-based SQR T-CSLA design involves more Area Delay Product and consumes energy than the proposed SQR T 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 electronics 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 SQR T 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] B. Parhami, *Computer Arithmetic: Algorithms and Hardware Designs*, 2nd ed. New York, NY, USA: Oxford Univ. Press, 2010.