



# AREA-DELAY EFFICIENT IMPLEMENTATION OF SQRT-CSLA

G. SWETHA  
M. Tech (VLSI DESIGN)  
[Gopuswetha42@gmail.com](mailto:Gopuswetha42@gmail.com)  
JITS, KARIMNAGAR

G. SHYAM KISHORE  
Assoc. Prof in ECE Dept  
[urs\\_shyamg@yahoo.com](mailto:urs_shyamg@yahoo.com)  
JITS, KARIMNAGAR

*Abstract— Design of area, high speed and power-efficient data path logic systems forms the largest areas of research in VLSI system design. The addition speed is limited by the time necessary to transmit a carry through the adder. Carry Select Adder (CSLA) is one of the fastest adders used in several data processing processors to perform fast arithmetic purpose. From the configuration of the CSLA, it is clear that there is scope for decreasing the area and delay in the CSLA. This work uses a simple and an efficient gate-level modification which drastically reduces the area and delay of the CSLA. Based on this modification 16, 32, 64 and 128-bit square-root Carry Select Adder (SQRT CSLA) architectures have been improved and compared with the regular SQRT CSLA architecture. The proposed design has compact area and delay to a great extent when compared with the regular SQRT CSLA. This work estimates the performance of the planned designs with the regular designs in terms of delay, area and synthesis are implemented in Xilinx FPGA. The results analysis shows that the proposed CSLA structure is better than the regular SQRT CSLA.*

*Keywords— (ASIC), area-efficient, CSLA, low delay*

## I. INTRODUCTION

Reduced area and high speed data path logic systems are the major areas of research in VLSI system design. High-speed addition and multiplication has always been a fundamental necessity of high-performance processors and systems. In digital adders, the speed of addition is partial by the time necessary to propagate a carry through the adder. The sum for each bit position in an elementary adder is generated serially only after the previous bit position has been summed and a carry propagated into the next position. There are several types of adder designs available (RCA, CLAA, CSA, CSA) which have its own advantages and disadvantages. The main speed limitation in any adder is in the production of carries and many authors considered the addition problem. To solve the carry propagation delay CSLA is developed which drastically decreases the area and delay to a great extent. The CSLA is used in many computational systems design to moderate the problem of carry propagation delay by separately generating multiple carries and then select a carry to generate the sum. It uses independent ripple carry adders (for  $C_{in}=0$  and  $C_{in}=1$ ) to generate the resultant sum. However, the Regular CSLA is not area and speed efficient because it uses multiple pairs of Ripple Carry Adders (RCA) to generate partial sum and carry by considering carry input. The final sum and carry are selected by the multiplexers (mux). Due to the use of two independent RCA the area will enlarge which leads an increase in delay. To overcome the above problem, the basic idea of the planned work is to use n-bit binary to excess-1 code converters (BEC) to improve the speed of addition. This logic can be replaced in

RCA for  $C_{in}=1$  to further improves the speed and thus decreases the delay. Using Binary to Excess-1 Converter (BEC) instead of RCA in the regular CSLA will achieve lower area, delay which speeds up the addition operation. The major advantage of this BEC logic comes from the lesser number of logic gates than the Full Adder (FA) structure because the number of gates used will be decreased.

## II. BASIC ADDER BLOCK

The adder block with a Ripple carry adder, BEC and Mux is explained in this section. In this we calculate and explain the delay & area using the theoretical approach and explain how the delay and area effect the total implementation. The AND, OR, and Inverter (AOI) implementation of an XOR gate is shown in Fig. 1. The delay and area evaluation methodology considers all gates to be made up of AND, OR, and Inverter, each having delay equal to 1 unit and area equal to 1 unit. We then add up the number of gates in the longest path of a logic block that contributes to the maximum delay. The area evaluation is done by counting the total number of AOI gates necessary for each logic block. Based on this approach, the blocks of 2:1 mux, Half Adder (HA), and FA are evaluated in Table I.

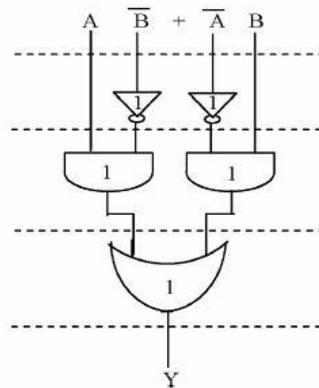


Fig 1: Delay and area evaluation of xor

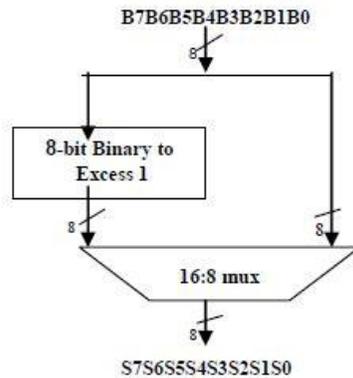


Fig 2: 8-bit BEC with 16:8 mux

Fig 2 shows the basic 8-bit addition procedure which includes 8-bit data, a 8-bit BEC logic and 16:8 mux. The addition operation is performed for  $C_{in}=0$  and for  $C_{in}=1$ . For  $C_{in}=0$  the addition is performed using ripple carry adder and for  $C_{in}=1$  the operation is performed using 8-bit BEC (replacing the RCA for  $C_{in}=1$ ). The resultant is selected based on Carry in signal from the earlier group. The total delay depends on mux delay and  $C_{in}$  signal from earlier group.

### III. PROPOSED CSLA

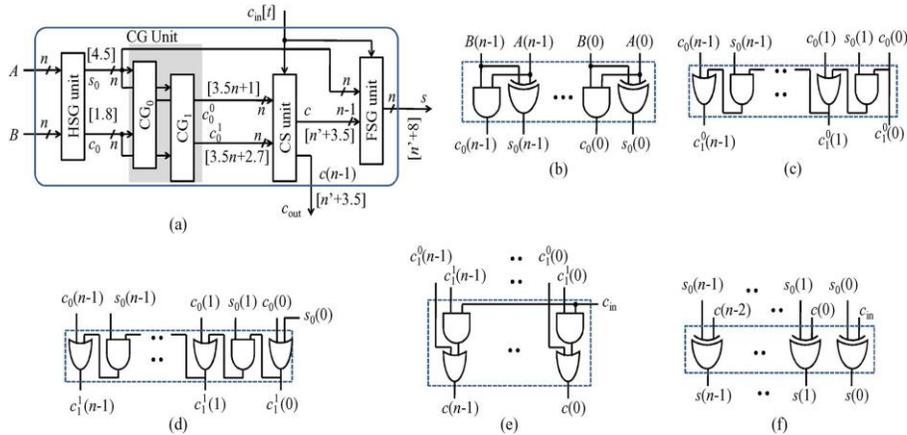


Fig. 3. (a) Proposed CS adder design, where  $n$  is the input operand bit-width, and  $[*]$  represents delay (in the unit of inverter delay),  $n = \max(t, 3.5n + 2.7)$ . (b) Gate-level design of the HSG. (c) Gate-level optimized design of (CG0) for input-carry = 0. (d) Gate-level optimized design of (CG1) for input-carry = 1. (e) Gate-level design of the CS unit. (f) Gate-level design of the final-sum generation (FSG) unit

The proposed CSLA is based on the logic formulation given in [1], and its structure is shown in Fig. 3(a). 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  $s_0$  and *half-carry* word  $c_0$  of width  $n$  bits each. Both CG0 and CG1 receive  $s_0$  and  $c_0$  from the HSG unit and generate two  $n$ -bit full-carry words  $c_{01}$  and  $c_{11}$  corresponding to input-carry ‘0’ and ‘1’, respectively. The logic diagram of the HSG unit is shown in Fig. 3(b). 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 Fig. 3(c) and (d), 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  $c_{01}$  when  $c_{in} = 0$ ; otherwise, it selects  $c_{11}$ . 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  $c_{01}$  and  $c_{11}$  follow a specific bit pattern. If  $c_{01}(i) = ‘1’$ , then  $c_{11}(i) = 1$ , irrespective of  $s_0(i)$  and  $c_0(i)$ , for  $0 \leq i \leq n - 1$ . This feature is used for logic optimization of the CS unit. The optimized design of the CS unit is shown in Fig. 3(e), 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 XORed with  $(n - 1)$  MSBs of *half-sum* ( $s_0$ ) in the FSG [shown in Fig. 3(f)] to obtain  $(n - 1)$  MSBs of *final-sum* ( $s$ ). The LSB of  $s_0$  is XORed with  $c_{in}$  to obtain the LSB of  $s$ .

### IV. ARCHITECTURE OF REGULAR 128-BIT SQRT CSLA

A 16-bit carry select adder can be designed in two different sizes namely uniform block size and variable block size. Similarly a 32, 64 and 128-bit can also be developed in two modes of different block sizes. Ripple-carry adders are the simplest and most compact full adders, but their performance is limited by a carry that must spread from the least-significant bit to the most significant bit. The various 16, 32, 64 and 128-bit CSLA can also be developed by using ripple carry adders. The speed of a carry-select adder can be enhanced upto 40% to 90%, by performing the additions in parallel, and reducing the maximum carry delay.

CSLA. It includes many ripple carry adders of variable sizes which are separated into groups. Group 0 contains 2-bit RCA which contains only one ripple carry adder which adds the input bits and the input carry and results to sum [1:0] and the carry out. The carry out of the Group 0 which acts as the selection input to mux which is in group 1, selects the result from the consequent RCA ( $C_{in}=0$ ) or RCA ( $C_{in}=1$ ). Similarly the remain groups will be selected depending on the  $C_{out}$  from the previous groups.

In Regular CSLA, there is only one RCA to achieve the addition of the least significant bits [1:0]. The remaining bits (other than LSBs), the addition is performed by using two RCAs corresponding to the one assuming a carry-in of 0, the other a carry-in of 1 within a group. In a group, there are two RCAs that receive the same data inputs but different  $C_{in}$ .

The upper adder has a carry-in of 0, the lower adder a carry-in of 1. The actual Cin from the previous sector selects one of the two RCAs. That is, as shown in the Fig.4, if the carry-in is 0, the sum and carry-out of the upper RCA is selected, and if the carry-in is 1, the sum and carry-out of the lower RCA is selected. For this Regular CSLA architecture, the execution code, for the Full Adders and Multiplexers of different sizes (6:3, 8:4, 10:5 up to 24:11) were designed initially. The regular 64-bit, 128-bit CSLA were implemented by calling the ripple carry adders and all multiplexers.

### V. ARCHITECTURE OF MODIFIED 128-BIT SQRT CSLA

This architecture is alike to regular 128-bit Sqrt CSLA, the only change is that, we change RCA with Cin=1 among the two available RCAs in a group with a BEC. This BEC has a feature that it can perform the alike operation as that of the replaced RCA with Cin=1. Fig 5 shows the Modified block diagram of 128-bit Sqrt CSLA. The number of bits necessary for BEC logic is 1 bit more than the RCA bits. The modified block diagram is also separated into various groups of variable sizes of bits with each group having the ripple carry adders, BEC and corresponding mux. As shown in the Fig.5, Group 0 include one RCA only which is having input of lower significant bit and carry in bit and produces result of sum[1:0] and carry out which is acting as mux selection line for the next group, also the procedure continues for higher groups but they include BEC logic instead of RCA with Cin=1. Based on the consideration of delay values, the arrival time of selection input C1 of 8:3 mux is before than the sum of RCA and BEC. For remain groups the selection input arrival is later than the RCA and BEC. Thus, the sum1 and c1 (output from mux) are depending on mux and results computed by RCA and BEC respectively. The sum 2 depends on c1 and mux. For the remain parts the arrival time of mux selection input is always larger than the arrival time of data inputs from the BEC's. Thus, the delay of the remaining MUX depends on the arrival time of mux selection input and the mux delay. In this Modified CSLA architecture, the implementation code for Full Adder and Multiplexers of 6:3, 8:4, and 10:5 up to 24:11 were designed. The design code for the BEC was designed by using NOT, XOR and AND gates. Then 2, 3, 4, 5 up to 11-bit ripple carry adder was designed

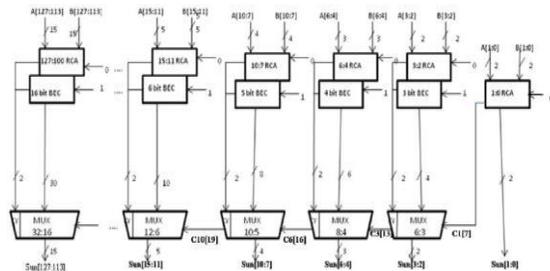


Figure 5: Architecture of Modified 128-bit Sqrt CSLA

### VI. RESULTS

The implemented design in this work has been simulated using Verilog-HDL. The adders (of various size 16, 32, 64 and 128) are designed and simulated with Modelsim. All the V files (Regular and Modified) are also simulated in Modelsim and corresponding results are compared. After simulation the different size codes are synthesized using Xilinx ISE 9.1i. The simulated V files are imported into the synthesized tool and resultant values of delay and area are noted. The synthesized reports include area and delay values for different sized adders. The similar design flow is followed for both the regular and modified Sqrt CSLA of different sizes.

The comparison of regular and modified CSLA of various bits which includes Delay and area comparisons. From the table it is clear that the delay decreases for 16-bit modified method when compared with regular method. Similarly the table also shows the comparison for the various 32, 64, and 128 bits. The relative values of areas shows that the number of LUT will be more for modified the 16, 32 and 64. This value reduces gradually for 128 bits. For 256 bits the value almost equal to regular method which will decreases more for still higher order bits. Thus the modified method reduces the delay and also area to a great extent.

## VII. CONCLUSION

An efficient approach is planned in this paper to decrease the area and delay of SQR CSLA architecture. The reduction in the number of gates is obtained by simply replacing the RCA with BEC in the structure. The compared results shows that the modified SQR CSLA has a slightly bigger area for lower order bits which further reduces for higher order bits. The delay is reduced to a great extent with the modified SQR CSLA. Thus the results shows that using modified method the area and delay will reduce thus leads to good alternative for adder implementation for many processors. The modified CSLA architecture is therefore low are and high speed approaches for VLSI hardware implementation.

## REFERENCES

- [1] Bedrij, O. J., (1962), "Carry-select adder," IRE Trans. Electron. Comput. Pp.340–344.
- [2] Ramkumar,B. , Kittur, H.M. and Kannan ,P. M.,(2010 ),"ASIC implementation of modified faster carry save adder," Eur. J. Sci. Res., vol. 42, no. 1,pp.53–58.
- [3] Kim ,Y. and Kim ,L.-S.,(May2001), "64 bit carry-select adder with reduced area, "Electron Lett., vol. 37, no. 10, pp. 614–615.
- [5] Ceiang, T. Y. and Hsiao. J., (Oct 1998), "Carry-select adder using single ripple carry adder," Electron. Lett., vol. 34, no. 22, pp. 2101– 2103
- [6] He, Y., Chang, C. H. and Gu, J., (2005), "An A rea efficient 64-bit square root carry-select adder for low power application," in Proc. IEEE Int. Symp.Circuits Syst. vol. 4, pp. 4082–4085.
- [7] E. Abu-Shama and M. Bayoumi, "A New cell for low power adders," in Proc.Int.Midwest Symp. Circuits and Systems, 1995, pp. 1014– 1017
- [8] Samir Palnitkar, "Verilog Hdl, A guide to Digital Design and Synthesis



G.Shyam Kishore received his B.Tech degree in Electronics and communication Engineering. and received his M.Tech degree in Electronics and communication Engineering with specialization in VLSI System Design. Currently, he is working as Assoc.Prof in ECE Dept, JITS, Karimnagar with 12 years of teaching experience.



G.swetha received B.Tech degree in Electronics and communication Engineering. Currently, I am studying M.Tech (vlsi design)in jits, karimnagar