

An Efficient implementation of 16-bit Carry-Select Adder Using Verilog HDL



R.Matri Bai
M.Tech Student,
Malla Reddy Womens College.



G.Sahithi Reddy
Assistant Professor,
Malla Reddy Womens College.

Abstract:

Carry Select Adder (CSLA) is one of the fastest adders used in many data-processing processors to perform fast arithmetic functions. From the structure of the CSLA, it is clear that there is scope for reducing the area and power consumption in the CSLA. This work uses the logic operations involved in conventional carry select adder (CSLA) and Dual RCA's based CSLA, binary to excess converter(BEC)-based CSLA, CLB based CSLA are analyzed to study the data dependence and to identify redundant logic operations. We have eliminated all the redundant logic operations present in the conventional CSLA and proposed a new logic formulation for CSLA. In the proposed scheme, 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 $C_{in}=0$ and 1) and fixed C_{in} bits are used for logic optimization of CS and generation units. An efficient CSLA design is obtained using optimized logic unit. Based on this modification 8-, 16-, 32-, and 64-bit square-root CSLA (SQRT CSLA) architecture have been developed and compared with the regular SQRT CSLA architecture. The proposed design has reduced area and power as compared with the regular SQRT CSLA with only a slight increase in the delay. This work evaluates the performance of the proposed designs in terms of delay, area, power. The results analysis shows that the proposed CSLA structure is better than the regular SQRT CSLA.

Index Terms:

Adder, arithmetic unit, BEC's, CLB's, area efficient, CSLA, low power.

I. INTRODUCTION:

Implementation of 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 to 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]. We observe that logic optimization largely depends on availability of redundant operations in the formulation, whereas adder delay mainly depends on data dependence. In the existing designs, logic is optimized without giving any consideration to the data dependence. In this brief, we made an analysis on logic operations involved in conventional and BEC-based CSLAs to study the data dependence and to identify redundant logic operations. Based on this analysis, we have proposed a logic formulation for the CSLA.

The main contribution in this brief are logic formulation based on data dependence and optimized carry generator (CG) and CS design. Based on the proposed logic formulation, we have derived an efficient logic design for CSLA. Due to optimized logic units, the proposed CSLA involves significantly less ADP than the existing CSLAs. We have shown that the Sqrt-CSLA using the proposed CSLA design involves nearly 32% less ADP and consumes 33% less energy than that of the corresponding Sqrt-CSLA. The rest of this brief is organized as follows. Logic formulation of CSLA is presented in Section II. The proposed CSLA is presented in Section III and the Synthesis and Simulation Results is presented in Section IV. The conclusion is given in Section V.

II. LOGIC FORMULATION:

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.

$$s^0(i) = A(i) \oplus B(i) \quad c^0(i) = A(i) \cdot B(i) \quad (1a)$$

$$s^0(1) = s^0(i) \oplus c^0(1(i-1)) \quad (1b)$$

$$c^0(1) = c^0(i) + s^0(i) \cdot c^0(1(i-1)) \quad c^0_{out} = c^0(1(n-1)) \quad (1c)$$

$$s^1(i) = A(i) \oplus B(i) \quad c^1(i) = A(i) \cdot B(i) \quad (2a)$$

$$s^1(1) = s^1(i) \oplus c^1(1(i-1)) \quad (2b)$$

$$c^1(1) = c^1(i) + s^1(i) \cdot c^1(1(i-1)) \quad c^1_{out} = c^1(1(n-1)) \quad (2c)$$

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 dependent.

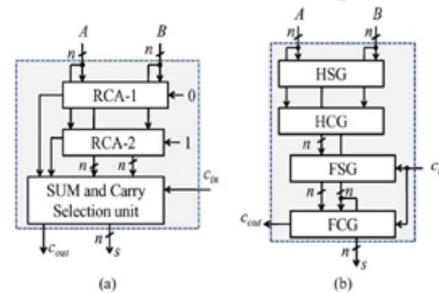


Fig.1. (a) Conventional CSLA; n is the input operand and bit-width. (b) The logic operations of the RCA is shown in split form, where HSG, HCG, FSG, and FCG represent half-sum generation, half-carry generation, full-sum generation, and full-carry generation, respectively.

A. 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 (s0 and s1) and output-carry (c0 and c1) corresponding to input-carry (Cin = 0 and Cin = 1), respectively. Logic expressions of RCA-1 and RCA-2 of the SCG unit of the n-bit CSLA are given as

where $c_{01}(-1) = 0$, $c_{11}(-1) = 1$, and $0 \leq i \leq n - 1$. As shown in (1a)–(1c) and (2a)–(2c), the logic expression of $\{s_{00}(i), c_{00}(i)\}$ is identical to that of $\{s_{10}(i), c_{10}(i)\}$. 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. Logic Expression of the SCG Unit of the BEC Based CSLA

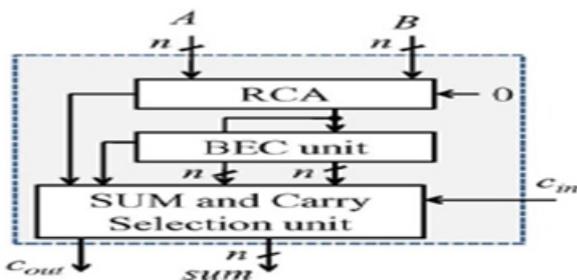


Fig.2. Structure of the BEC-based CSLA; n is the input operand bit-width.

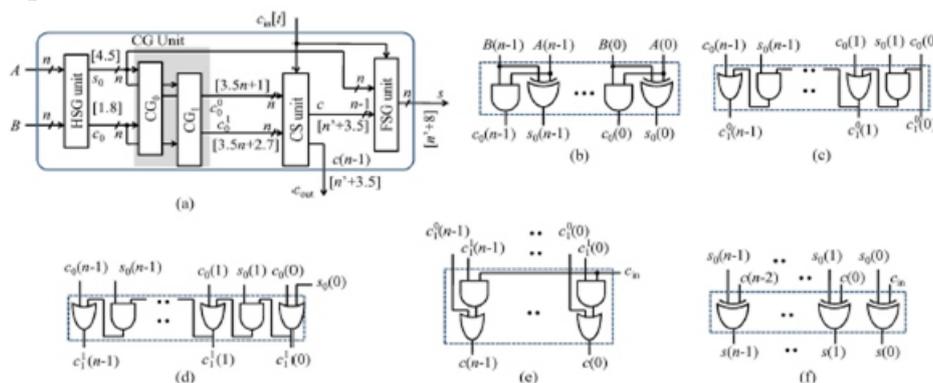


Fig. 3.(a) Proposed CS adder design, where n is the input operand bit-width, and [] represents delay (in the unit of inverter delay), $n = \max(t, 3.5n + 2.7)$. (b) Gate-level design of the HSG. (c) Gate-level optimized design of (CG) for input-carry = 0. (d) Gate-level optimized design of (CG1) for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

We can find from (1a)–(1c) and (3a)–(3d) that, in the case of the BEC-based CSLA, c_{11} depends on s_{01} , which otherwise has no dependence on s_{01} in the case of the conventional CSLA. The terms c_{01} and c_{11} since $(s_{00} = s_{10} = s_0)$. In addition, we find that c_{01} and c_{11} depend on $\{s_0, c_0, C_{in}\}$, where $c_0 = c_{00} = c_{10}$.

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_{01} and s_{11} are identical except as shown in Fig. 2, the RCA calculates n-bit sum s_{01} and c_{0out} corresponding to $C_{in} = 0$.

The BEC unit receives s_{01} and c_{0out} from the RCA and generates (n + 1)-bit excess-1 code. The most significant bit (MSB) of BEC represents c_{1out} , in which n least significant bits (LSBs) represent s_{11} . 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

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

$$s^1_1(i) = s^0_1(i) \oplus c^1_1(i - 1) \quad (3b)$$

$$c^1_1(i) = s^0_1(i) \cdot c^1_1(i - 1) \quad (3c)$$

$$c^1_{out} = c^0_1(n - 1) \oplus c^1_1(n - 1) \quad (3d)$$

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

Since c_{01} and c_{11} have no dependence on s_{01} and s_{11} , the logic operation of c_{01} and c_{11} can be scheduled before s_{01} and s_{11} , and the select unit can select one from the set (s_{01}, s_{11}) for the final-sum of the CSLA. We find that a significant amount of logic resource is spent for calculating $\{s_{01}, s_{11}\}$, 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 $\{c_0 \text{ and } c_1\}$ 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 is avoided in the SCG unit;

$$\begin{aligned}
 s_0(i) &= A(i) \oplus B(i) & c_0(i) &= A(i) \cdot B(i) \quad (4a) \\
 c^0_1(i) &= c^0_1(i-1) \cdot s_0(i) + c_0(i) & \text{for } c^0_1(0) &= 0 \quad (4b) \\
 c^1_1(i) &= c^1_1(i-1) \cdot s_0(i) + c_0(i) \text{ for} & c^1_1(0) &= 1 \quad (4c) \\
 c(i) &= c^0_1(i) & \text{if } (C_{in} = 0) & \quad (4d) \\
 c(i) &= c^1_1(i) & \text{if } (C_{in} = 1) & \quad (4e) \\
 c_{out} &= c(n-1) & & \quad (4f) \\
 s(0) &= s_0(0) \oplus c_{in}s(i) = s_0(i) \oplus c(i-1). & & \quad (4g)
 \end{aligned}$$

III. PROPOSED ADDER DESIGN:

The proposed CSLA is based on the logic formulation given in (4a)–(4g), 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_0 and c_1 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_0 when $C_{in} = 0$; otherwise, it selects c_1 . 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_0 and c_1 follow a specific bit pattern. If $c_0(i) = '1'$, then $c_1(i) = 1$, 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).

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

The LSB of s_0 is XORed with C_{in} to obtain the LSB of s . The multipath carry propagation feature of the CSLA is fully exploited in the SQR-CSLA [5], which is composed of a chain of CSLAs. CSLAs of increasing size are used in the SQR-CSLA to extract the maximum concurrence in the carry propagation path. Using the SQR-CSLA design, large-size adders are implemented with significantly less delay than a single-stage CSLA of same size. However, carry propagation delay between the CSLA stages of SQR-CSLA is critical for the overall adder delay. Due to early generation of output-carry with multipath carry propagation feature, the proposed CSLA design is more favorable than the existing CSLA designs for area-delay efficient implementation of SQR-CSLA. A 16-bit SQR-CSLA design using the proposed CSLA is shown in Fig. 4, where the 2-bit RCA, 2-bit CSLA, 3-bit CSLA, 4-bit CSLA, and 5-bit CSLA are used. We have considered the cascaded configuration of (2-bit RCA and 2-, 3-, 4-, 6-, 7-, and 8-bit CSLAs) and (2-bit RCA and 2-, 3-, 4-, 6-, 7-, 8-, 9-, 11-, and 12-bit CSLAs), respectively, for the 32-bit CSLA and the 64-bit SQR-CSLA to optimize adder delay. To demonstrate the advantage of the proposed CSLA design in SQR-CSLA, we have estimated the area and delay of SQR-CSLA using the proposed CSLA design and the BEC-based CSLA of [6] and the CBL-based CSLA of [7] for bit-widths 16, 32, and 64. The estimated delay of the CBL-based SQR-CSLA [7] is significantly higher for large bit-widths than the proposed SQR-CSLA and BEC-based SQR-CSLA designs. Compared with SQR-CSLA designs of [6] and [7], the proposed SQR-CSLA design, respectively, involves 35% and 72% less ADP, on average, for different bit-widths.

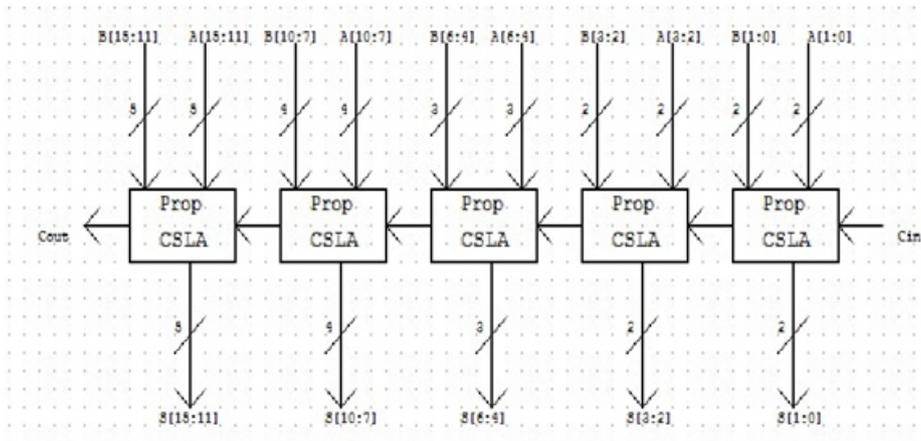


Fig. 4. Proposed 16-bit Sqrt-CSLA

IV. SYNTHESIS AND SIMULATION RESULTS:

We have coded the Sqrt-CSLA in Verilog HDL using the proposed CSLA design and the existing CSLA designs of [6] and [7] for bit-widths 8, 16, 32, and 64. All the designs are synthesized in the Xilinx Synthesis Tool and Simulated using Xilinx ISE simulator. After placement and route on FPGA, the area, delay and power values are compared with conventional Sqrt-CSLA's. This synthesis result confirms that the proposed Sqrt-CSLA involves significantly less area and less delay and consumes less power than the existing designs. We can find from Fig. 5 that the proposed Sqrt-CSLA design offers a saving of 39% ADP and 37% energy than the RCA-based conventional Sqrt-

CSLA; 32% ADP and 33% energy than the BEC-based Sqrt-CSLA of [6]; and 55% ADP and 30% energy than the CBL-based Sqrt-CSLA of [7], on average, for different bit-widths. The schematics and simulated outputs are found from fig6 and fig7.

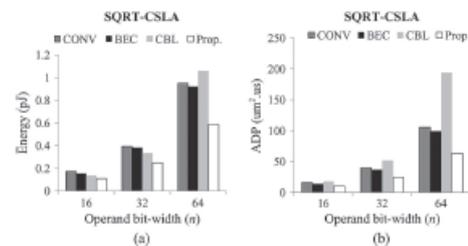


Fig. 5. (a) Comparison of energy consumption. (b) Comparison of ADP.



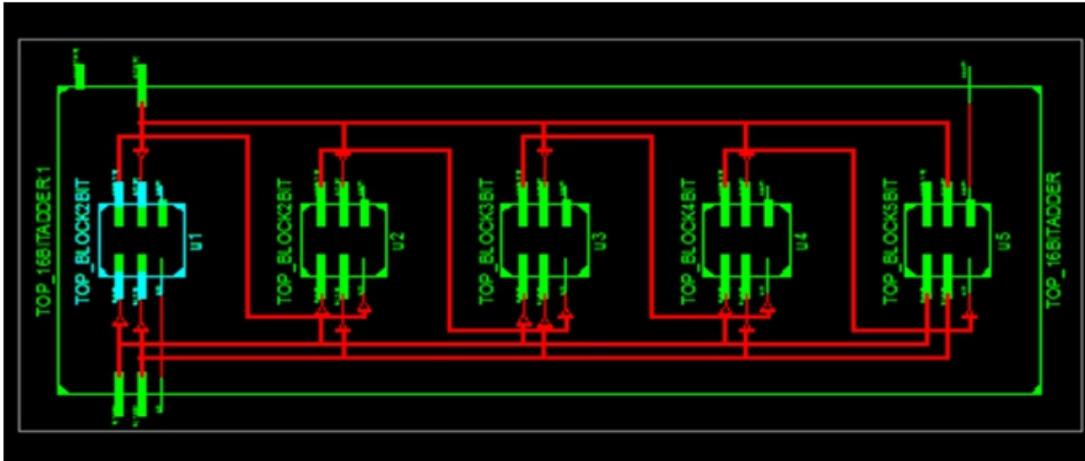


Fig. 6. (a) RTL schematic of Proposed 16 bit SQRT CSLA

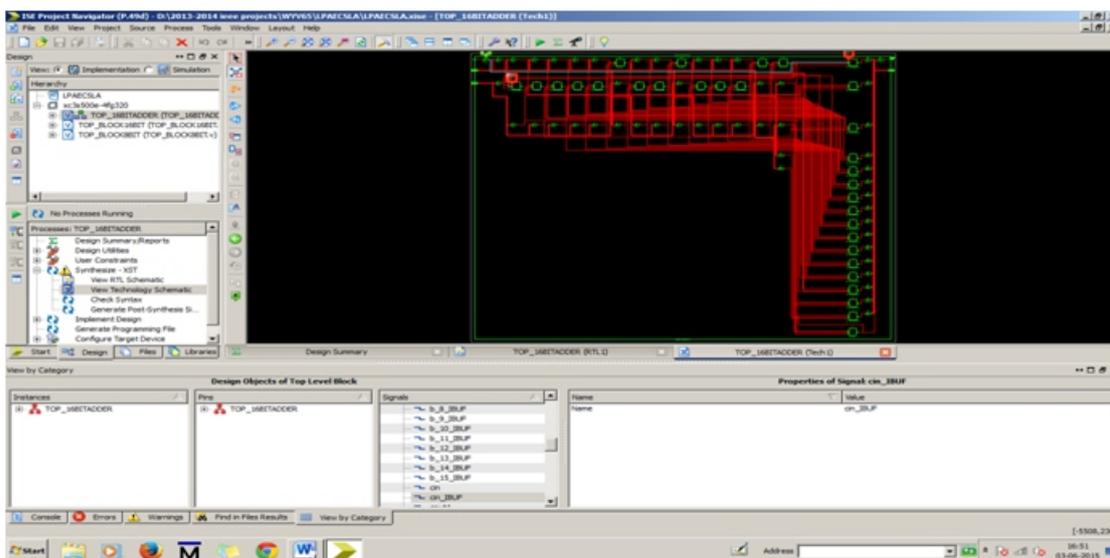


Fig. 6. (b) Technology schematic of Proposed 16 bit SQRT CSLA

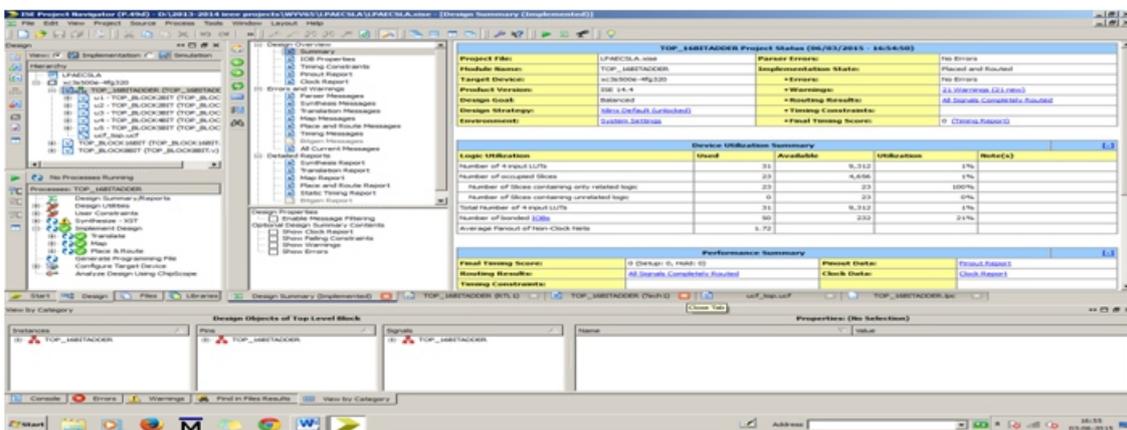


Fig. 6. (c) Synthesis Summary of Proposed 16 bit SQRT CSLA



Fig. 7. (a) Simulation output of Proposed 16 bit Sqrt CSLA

V. CONCLUSION:

A simple approach is proposed in this paper to reduce the area and power of Sqrt CSLA architecture. The logic operations eliminated all the redundant logic operations of the conventional CSLA and proposed a new logic formulation for the CSLA. In the proposed scheme, the CS operation is scheduled before the calculation of final-sum, which is different from the conventional approach. Carry words corresponding to input-carry '0' and '1' generated by the CSLA based on the proposed scheme follow a specific bit pattern, which is used for logic optimization of the CS unit. Fixed input bits of the CG unit are also used for logic optimization. Based on this, an optimized design for CS and CG units are obtained. Using these optimized logic units, an efficient design is obtained for the CSLA. 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 Sqrt adder. The synthesis result shows that the existing BEC-based Sqrt-CSLA design involves 48% more ADP and consumes 50% more energy than the proposed Sqrt-CSLA, on average, for different bit-widths.

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 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, Senior Member, IEEE, and Sujit Kumar Patel: *IEEE Transactions On Circuits And Systems—II: Express Briefs*, VOL. 61, NO. 6, JUNE 2014