IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
National Conference on Emerging Trends in Engineering and Technology (NCETET'16)
Lourdes Matha College of Science & Technology, Thiruvananthapuram
Vol. 3, Special Issue 3, August 2016

# SQRT Carry Select Adder with Efficient Area Delay Product Using VHDL Architecture

**Varsha Viswam[1], Suchithra S Nair[2]**

P G Scholar, ECE, Rajadhani Institute of Engineering and Technology, Thiruvananthapuram, Kerala, India[1]

Faculty Member, ECE, Rajadhani Institute of Engineering and Technology, Thiruvananthapuram, Kerala, India[2]

**Abstract**: Reduced area, delay and power dissipation are the main factors that play an important role in the increasing demand of electronic devices. Adders have a great role in computing arithmetic unit. In the proposed adder design, new logic formulation is proposed by analysing the existing methods thereby reducing the redundant logic operations and data dependency. The probable carry-in (either $c_{in}$ = 0 or $c_{in}$ =1) from the previous Carry Select Adder (CSLA) decides the original sum and carry out. Here, the carry is scheduled before the sum generation. Thus the proposed system experiences small carry output delay. Low area efficient and delay efficient design is implemented. Thus the proposed SQRT CSLA can replace the existing CSLA designs.

**Keywords:** Adder, Arithmetic Unit, Carry Select Adder, Delay Efficient

## I. INTRODUCTION

Adders have great importance in electronic industry. The demand for high speed equipment with small area is increasing day by day. CSLAs are used for high speed calculations in calculators, mobiles, computers etc. Ripple Carry Adder (RCA) is the simplest CSLA for addition calculation. But it is the slowest CSLA. RCA is built with cascaded full adders (FA).

The existing conventional CSLA is made up of two RCAs for each bit group. One RCA for addition with carry-in ($c_{in}$)=0and other with $c_{in}$= 1. The two addition operations are done separately and the real sum and carry out are selected with the use of Multiplexer (MUX). The Carry Propagation Delay (CPD) is the problem while using RCA. So here the delay is more. The CSLAs are arranged in increasing order of bits. Since this existing SQRT CSLA has more CPD, the design is not attractive since it uses two RCAs. Later Kim and Kim proposed a new SQRT CSLA with one RCA and an add one circuit. The MUX selects the real carry out and sum since it uses the previous carry bit as the select line. It has less CPD than the conventional CSLA. Then Ramkumar and Kittur proposed Binary to Excess One (BEC) converter based CSLA. Here one RCA for addition operation with $c_{in}$= 0 and one BEC for addition with $c_{in}$= 1. It has less resources than the conventional CSLA. So the area is reduced to some extent. Basant Kumar Mohanty and Sujith Kumar Patel proposed a new logic formulation by analysing the logic expressions of conventional CSLA and the BEC based CSLA. They formulate logic expressions based on data dependency and redundant logic operations. In the proposed adder the logic resources are analyzed and new logic expressions are made by avoiding redundant logic operations and data dependency. In the proposed system, the logic resources depend on the anticipated carry ($c_{in}$ = 0 and 1). Based on the proposed formulation, CSLA with less delay is designed by considering the data dependency and logic resources.

## II. PROPOSED LOGIC FORMULATION

The proposed CSLA has mainly two units: 1) Sum Generation and 2) Carry Selection and Generation. Anticipated carry decides the carry and thus the sum. Here the logic formulation is done in such a way that carry is generated before sum is scheduled. The logic operation to select the anticipated carry is an important task while designing the CSLA. In the proposed CSLA, the new logic formulation is proposed by analyzing the BEC based CSLA and RCA-RCA based CSLA and avoiding the redundant bits based on the data dependency. The data dependency is analyzed and reached into a new logic formulation by avoiding the redundant formulations.

### *A.* Logic Formulation of Conventional CSLA

The SCG unit of conventional Sum Carry Generation (SCG) unit has four stages: i) Half-Sum Generation(HSG); (ii)Half Carry Generation (HCG); (iii) Full Sum Generation (FSG); and (iv) Full Carry Generation(FCG). Here the SCG unit consists of two n-bit width RCAs. Suppose two n-bit operands are added. Here RCA-1 generates the n-bit sum ($s^0$) and carry ($c^0$) bits and RCA-2 generates the n-bit sum($s^1$) and carry($c^0$) bits. The corresponding output carry bits are $c^0_{out}$and $c^1_{out}$respectivelycorresponding to input carry bits ($c_{in}$ = 0 and $c_{in}$ = 1). The logic formulations of RCA-RCA CSLA corresponding to the SCG unit are given as

$$s^0_0(i) = A(i) \oplus B(i); \qquad c^0_0(i) = A(i).B(i) \tag{1.1}$$

$$s^0_1(i) = s^0_0(i) \oplus c^0_1(i-1); \tag{1.2}$$

$$c^0_1(i) = c^0_0(i) + s^0_0(i).c^1_0(i-1); \qquad c^0_{out} = c^0_1(n-1) \tag{1.3}$$

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

IARJSET

International Advanced Research Journal in Science, Engineering and Technology

National Conference on Emerging Trends in Engineering and Technology (NCETET'16)

Lourdes Matha College of Science & Technology, Thiruvananthapuram

Vol. 3, Special Issue 3, August 2016

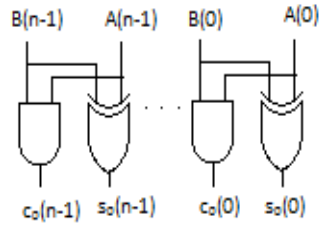


Fig. 1. Gate level design of HSCG

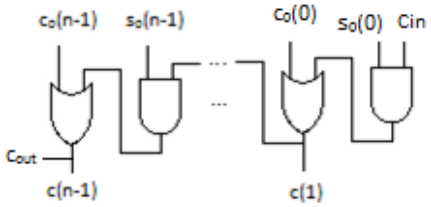


Fig.2 Gate level design of CSU (carry selectionunit)

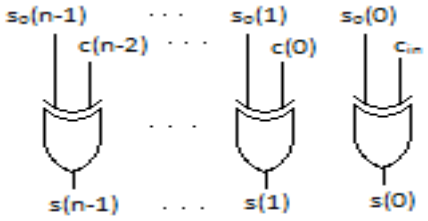


Fig. 3. Gate level design of FSCG (sum generation unit)

$s^1_0(i) = A(i) \oplus B(i)$; $c^1_0(i) = A(i).B(i)$ (2.1)

$s^0_1(i) = s^1_0(i) \oplus c^1_1(i-1)$ (2.2)

$c^1_1(i) = c^1_0(i) + s^1_0(i) . c^1_1(i-1)$; $c^1_{out} = c^1_1(n-1)$ (2.3)

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

From the above expressions redundant operations are observed, that is identical expressions are noticed {(1.1), (2.1) and (1.2), (2.2)}. These redundant operations are removed and insert an add-one circuit (later BEC) is used.

*B.* Logic Formulation of BEC based CSLA

In BEC based CSLA, RCA generates sum ($s^0_1$) and carry ($c^0$out) bits corresponding to input carry ($c_{in}$ = 0). That is the logic expressions is same as (1.1) – (1.3). From the RCA, the BEC-1 circuit receives $s^0_1$ and $c^0_{out}$ bits and generates (n+1) bits excess-1 code if the anticipated input carry is 1. The Most Significant Bit (MSB) denotes the carry out ($c^1_{out}$) and the n-Least Significant Bit (LSB) represents the sum ($s^1_1$). The BEC based logic expressions are given as

$s^1_1(0) = s^0_1(0)$; $c^1_1(0) = s^0_1(0)$ (3.1)

$s^1_1(i) = s^0_1(i) \oplus c^1_1(i-1)$ (3.2)

$c^1_1(i) = s^0_1(i) . c^1_1(i-1)$ (3.3)

$c^1_{out} = c^0_1(n-1) \oplus c^1_1(n-1)$ (3.4)

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

From (1.1) - (1.3) and (3.1) – (3.4), $c^1_1$depends on $s^0_1$, but has no dependence on $s^0_1$on conventional CSLA. Therefore, the BEC-1 based CSLA increases the data dependence in the conventional CSLA.

*C.* Logic Formulation of proposed CSLA

Here, the expressions of conventional CSLA and BEC based CSLA are identified and removed the redundant logic expressions. From (1.1) – (1.3) and (2.1 – 2.3), the logical expressions for sum ($s^0_1$ and $s^1_1$) are similar except the carry terms $c^0_1$ and $c^1_1$. But $s^0_0 = s^1_0 = s_0$ and $c^0_0 = c^1_0 = c_0$. So $c^0_1$ and $c^1_1$ have no dependence on $s^0_1$ and $s^1_1$ and the output carry is generated before the final sum calculation. Thus the data dependency can be conserved. The logic formulations of the modified CSLA are given as

$s_0(i) = A(i) \oplus B(i)$ $c_0(i) = A(i) . B(i)$ (4.1)

$c^0_1(i) = c^0_1(i-1) . s_0(i) + c_0(i)$ for $(c^0_1(0) = 0)$ (4.2)

$c^1_1(i) = c^1_1(i-1) . s_0(i) + c_0(i)$ for $(c^1_1(0) = 1)$ (4.3)

$c(i) = c^0_1(i)$ if $(c_{in} = 0)$ (4.4)

$c(i) = c^1_1(i)$ if $(c_{in} = 1)$ (4.5)

$c(i) = c^0_1(i) + c^1_1(i).c_{in}$ (4.6)

$c_{out} = c(n-1)$ (4.7)

$s(0) = s_0(0) \oplus c_{in}$ $s(i) = s_0(i) \oplus c(i-1)$ (4.8)

By analysing the expressions from (4.1) – (4.8.), the expression for calculating the final carry bit (c(i)) depends on both $c^0_1$ and $c^1_1$. The main problem concern here is if the anticipated carry is 1, the adder has to calculate $c^0_1$ and $c^1_1$even if $c^1_1$ is necessary. This leads to redundant logic operations. The equations (4.1) - (4.8) are further analysed and new formulation is proposed. The carry generation corresponding to anticipated carry bits ($c_{in}$= 0 and 1) have similar logic formula (4.2) and (4.3) and the only difference is the carry bit $c^0_1$(i-1) and $c^1_1$(i-1). By analysing (4.2), (4.3) and (4.6), a new expression can be find out and thus can eliminate the redundant logic operations. Thus delay can be reduced to some extent. The logic formulations of proposed CSLA are given as

$s_0(i) = A(i) \oplus B(i)$; $c_0(i) = A(i) . B(i)$ (5.1)

$c(i) = c(i-1) . s_0(i) + c(i-1)$ (5.2)

$c(i) = c_0(0)$ if $(c_{in} = 0)$ (5.3)

$c(i) = c_0(0) + s_0(0)$ if $(c_{in} = 1)$ (5.4)

$c_{out} = c(n-1)$ (5.5)

$s(0) = s_0(0) \oplus c_{in}$ $s(i) = s_0(i) \oplus c(i-1)$ (5.6)

## III. PROPOSED ADDER BLOCK DESIGN

The structure of the proposed CSLA has three units: 1) Half Sum and Carry Generation (HSCG) Unit, 2) Carry Selection Unit (CSU) and 3) Full Carry and Sum Generation (FSCG) Unit. Gate level design of HSCG and FSCG (sum and carry generation) are shown in Fig. (1) - (3). In HSCG unit, the RCA receives two n-bit operands and generate n-bit half sum and half carry bits. RCA consists of full adders. The second part is CSU unit. 2:1 multiplexer (MUX) is used as CSU unit. Here the anticipated carry bit is selected and gives the result to FCSG unit. The FSCG unit receives the anticipated carry bit and n-bit full carry generation is done. The MSB bit is the carry out ($c_{out}$). Then this n-bit carry bits are used for the full sum generation. Thus the carry is generated before the sum calculation and hence the delay can be reduced. The proposed CSLA architecture is shown in Fig. 4. The FSCG unit receives the anticipated carry bit and n-bit full carry generation is done. The MSB bit is the carry out ($c_{out}$). Then this n-bit carry bits are used for n-bit full sum generation. Thus the carry is generated before the final sum calculation and hence the delay is reduced. Fig.4 shows the proposed CSLA architecture.

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

IARJSET

International Advanced Research Journal in Science, Engineering and Technology

National Conference on Emerging Trends in Engineering and Technology (NCETET'16)

Lourdes Matha College of Science & Technology, Thiruvananthapuram

Vol. 3, Special Issue 3, August 2016

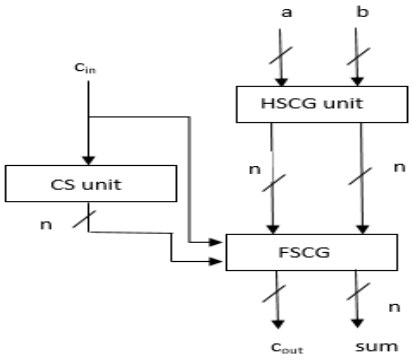


Fig. 4. Proposed CSLA architecture for n- bit

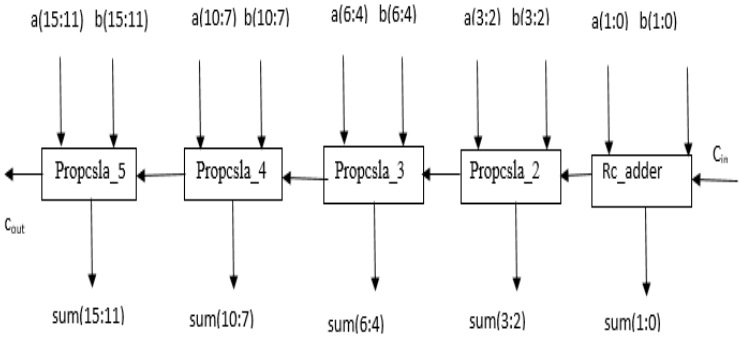


Fig. 5. Proposed SQRT CSLA block diagram for 16-bit

The proposed SQRT-CSLA of 16- bit is shown in Fig.5. The first block is a Ripple Carry Adder (RCA). RCA performs 2-bit addition using two full adders in series connection. The second block represents the proposed SQRT-CSLA for 2-bit using proposed SQRT-CSLA architecture. The third, fourth and fifth block also represents the proposed SQRT-CSLA for 3-bit, 4-bit and 5-bit respectively. The output carry, $c_{out}$ is taken from the last CSLA.

## IV. PERFORMANCE COMPARISON

Area–Delay Estimation Method

Consider all the gates to be made of 2-input AND, 2-input OR, and inverter (AOI). A 2-input XOR is composed of 2 AND, 1 OR, and 2 NOT gates. The area and delay of the 2-input AND, 2-input OR, and NOT gates (shown in Table I) are taken from the Synopsys Armenia Educational Department (SAED) 90-nm standard cell library datasheet for theoretical estimation. The area and delay of a design are calculated using the following relations:

$$A = a \cdot Na + r \cdot No + i \cdot Ni \quad (6.1)$$

$$T = na \cdot Ta + no \cdot To + ni \cdot Ti \quad (6.2)$$

where (Na,No,Ni) and (na, no, ni), respectively, represent the (AND, OR, NOT) gate counts of the total design and its critical path. (a, r, i) and (Ta, To, Ti), respectively, represent the area and delay of one (AND, OR, NOT) gate. The (AOI) gate counts of each design for area and delay estimation are made. Using (8.1) and (8.2), the area and delay of each design are calculated from the AOI gate counts (Na,No,Ni), (na, no, ni), and the cell details in Table I.

Table I: Area and Delay of AND, OR, and NOT gates given in the SAED90-nm standard cell library datasheet

| | AND gate | OR gate | NOT gate |
|---|---|---|---|
| Area ($um^2$) | 7.37 | 7.37 | 6.45 |
| Delay (ps) | 180 | 170 | 100 |

Using the expressions in (6.1) - (6.2), the area, delay, Area-Delay Product (ADP) and the Excess Area Delay Product (EADP) is found. The calculated values using the expressions (6.1) - (6.2) are given in Table II.

Table II: Estimate of Area and Delay Complexities of the proposed and Existing CSLAs

| Design | width(n) | Area $um^2$ | Delay (ns) | ADP ($um^2$.ns) | EADP (%) |
|---|---|---|---|---|---|
| Conventional CSLA | 16 | 2562.1 | 13.469 | 34508.5 | 10 |
| Modified BEC based CSLA | 16 | 1350.6 | 13.032 | 17600.9 | 6.4 |
| Proposed CSLA | 16 | 657.8 | 12.244 | 8053.99 | - |

## V. RESULT

The proposed SQRT CSLA and the existing design is coded in VHDL language. The program is implemented using Xilinx ISE Project Navigator 14.7. The proposed SQRT-CSLA is synthesized using ISim Simulator. The synthesis report of proposed adder is shown in fig. 6.

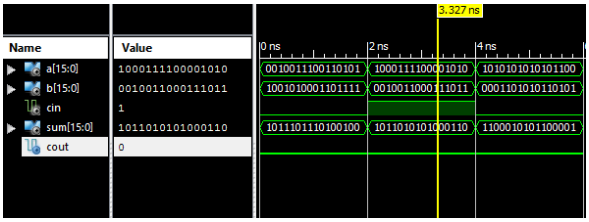


Fig. 6. Synthesis result of 16-bit proposed SQRT-CSLA adder

The delay report of proposed adder and existing methods are shown in fig (7) - (9) respectively.

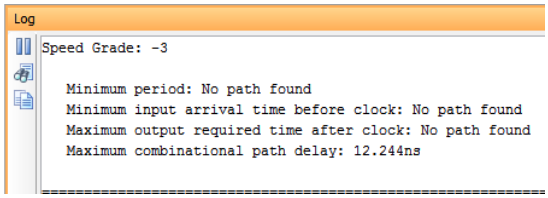


FIG.7. DELAY REPORT OF PROPOSED 16 BIT SQRT CSLA

IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
National Conference on Emerging Trends in Engineering and Technology (NCETET'16)
Lourdes Matha College of Science & Technology, Thiruvananthapuram
Vol. 3, Special Issue 3, August 2016

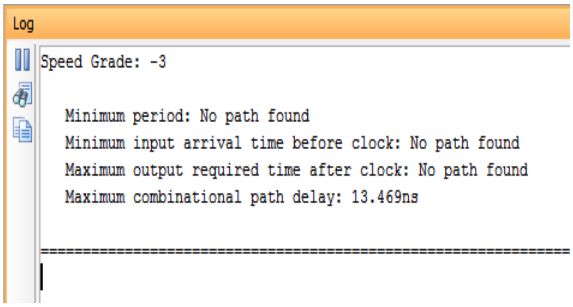


FIG. DELAY REPORT OF 16 BIT MODIFIED BEC BASED SQRT CSLA

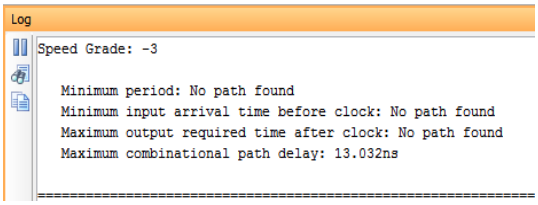


Fig.8. Delay report of 16 bit regular SQRT CSLA

## VI. CONCLUSION

The proposed SQRT CSLA uses less logic resources by avoiding redundant data bits. This is done by studying about the adder's logic resources and data dependency. The anticipated carry bits decide the output carry and the sum. Here carry is calculated after the calculation of sum operation. So the delay due to carry generation is reduced. Therefore, the adder delay can be reduced and by using this the faster devices can be made.

## REFERENCES

[1] Basant Kumar Mohanty, and Sujit Kumar Patel. "Area–Delay–Power Efficient Carry-Select Adder", IEEE transactions on circuits and systems—II: Express briefs, Vol. 61, No. 6, June 2014
[2] Ramkumar B and Harish M Kittur, (2012) "Low-Power and Area-Efficient Carry Select Adder", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 20, no. 2.
[3] K.MariyaPriyadarshini, N.V.N. Ravi Kiran, N. Tejasri, T.C. Venkat Anish. "Design of Areaand Speed Efficient Square RootCarry Select Adder Using Fast Adders", International Journal of Scientific & Technology research volume 3, issue 6, June 2014
[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 carryselectadder for low power application," in Proc. IEEE Int. Symp. CircuitsSyst., 2005, vol. 4, pp. 4082–4085.
[6] B. Ramkumar and H.M. Kittur, "Low-power and area-efficient carry-selectadder," 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 carryselect adder design by sharing the common Boolean logic term," in Proc.MECS, 2012, pp. 1–4.
[8] S.Manju and V. Sornagopal, "An efficient SQRT architecture of carry selectadder design by common Boolean logic," in Proc. VLSI ICEVENT, 2013,pp. 1–5.
[9] Kala Priya.K1, KSN Raju, "Carry select adder using BEC and RCA", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 3, Issue 10, October 2014.