International Journal of Engineering Research and General Science Volume 3, Issue 6, November-December, 2015
ISSN 2091-2730

AREA DELAY POWER EFFICIENT CARRY SELECT ADDER ON RECONFIGURABLE HARDWARE

Anjaly Sukumaran

MTech , Mahatma Gandhi University,anjalysukumaran2010@gmail.com,9605707726

Abstract— LOW-POWER, area-efficient, and high-performance VLSI systems are increasingly used in portable and mobile devices, multi standard wireless receivers, and bio medical instrumentation. 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 CSLA has less CPD than an RCA, but the design is not attractive since it uses a dual RCA. The main objective of SQRT-CSLA design is to provide a parallel path for carry propagation that helps to reduce the overall adder delay. 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) 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 . However, the CBL-based SQRT CSLA design requires more logic resource and delay than the BEC-based SQRT-CSLA . 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.So in this project work I have designed and implemented an FPGA based Low power and Area Efficient Carry Select adder . The design was coded in VHDL, simulated Xilinx ISE Design Suit 14.1 and implemented in Spartan6 FPGA trainer board.A theoretical estimate shows that the proposed SQRT-CSLA involves nearly 35% less area–delay–product (ADP) than the BEC-based SQRT-CSLA, which is best among the existing SQRT-CSLA designs, on average, for different bit-widths. The application-specified integrated circuit (ASIC) synthesis result shows that the 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.

Keywords— Adder, Ripple Carry Adder, Carry Propagation Delay, Common Boolean Logic, adder delay, low - power design, area-delay -product

INTRODUCTION

A conventional carry select adder (CSLA) is an RCA–RCA configuration that generates a pair of sum words and output carry bits corresponding the anticipated input-carry (cin = 0 and 1) and selects one out of each pair for final-sum and final-output-carry. 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. 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. 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) 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 . However, the CBL-based SQRT CSLA design requires more logic resource and delay than the BEC-based SQRT-CSLA . 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.

Customized algorithm implemented on FPGA generally works out better as compared to the similar algorithm executed on a standard x86 architecture. The algorithm implemented on FPGA functions as a hardware circuit and the complicated computational tasks accomplished by independent modules are formed by logic gates. The parallelism architecture offered by FPGA enables real time processing. Another reason which influences the use of FPGA in this system because it is closed to Application-Specific Integration Circuit (ASIC).Field Programmable Gate Array (FPGA) is an integrated circuit which can be configured by end user to implement functions defined by the designer. FPGAs are gaining popularity in design implementation due to their increasing logic density which

had reduced the cost per logic. FPGAs can be configured to meet custom requirement or functions and as well as performing tasks that can be done in parallel and pipelined whereby hardware outperforms software. Other than that, with existence of internet community designing and publishing IP cores, such as Open Cores, had allowed many custom designs to be made by Plug-and-Play (PnP) of multiple IP cores into single SOC using FPGAs.So in this project work I have designed and implemented an FPGA based Low power and Area Efficient Carry Select adder . The design was coded in VHDL, simulated Xilinx ISE Design Suit 14.1 and implemented in Spartan6 FPGA trainer board.

REMAINING CONTENTS

2. DESIGN OF THE CARRY SELECT ADDER

The CSLA has two units: 1) the sum and carry generator unit (SCG)2) the sum and carry selection unit. The SCG unit consumes most of the logic resources of CSLA and significantly contributes to the critical path. 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 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 dependence.As shown in Figure.3 (a), the SCG unit of the conventional CSLA 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) 4) full-carry generation (FCG)

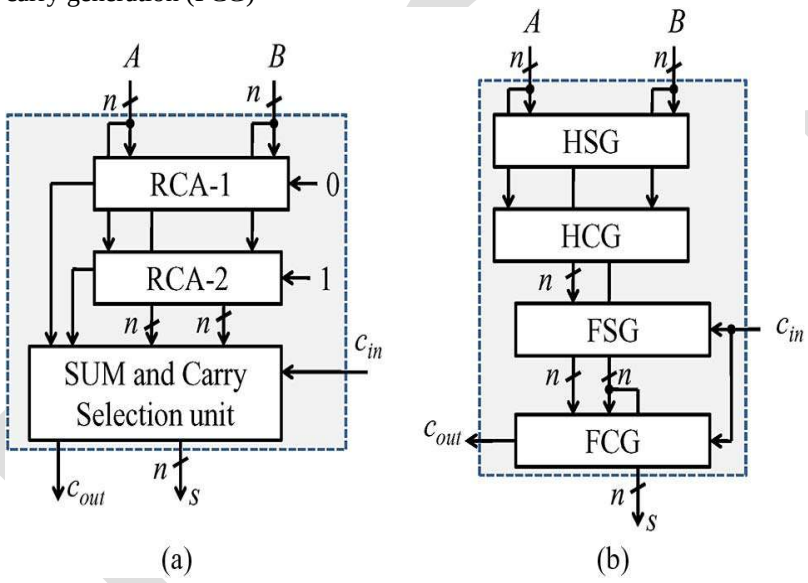


Figure.1. Various stages of proposed CSLA

3. IMPLEMENTATION OF THE LOW POWER AND AREA EFFICIENT CARRY SELECT ADDER

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. 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. However, the CSLA is not area efficient because it uses multiple pairs of Ripple Carry Adders (RCA) to generate partial sum and carry by considering carry input and, then the final sum and carry are selected by the multiplexers (mux). The main advantage of BEC logic comes from the lesser number of logic gates than the Full Adder (FA) structure. Based on this modification 8-, 16-, 32-, and 64-b square-root CSLA (SQRT CSLA) architecture have been developed and compared with the regular SQRT CSLA architecture.

3.1 DELAY AND AREA EVALUATION METHODOLOGY OF THE BASIC ADDER BLOCKS

The AND, OR, and Inverter (AOI) implementation of an XOR gate is shown in Figure.1. The gates between the dotted lines are performing the operations in parallel and the numeric representation of each gate indicates the delay contributed by that gate. 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 required for each logic block. Based on this approach, the CSLA adder blocks of 2:1 mux, Half Adder (HA), and FA are evaluated and listed in Table I.

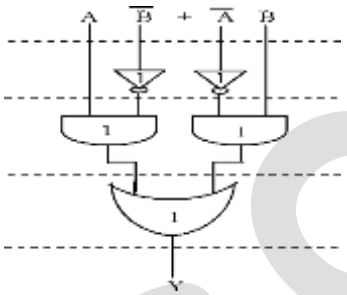


Figure.2. Delay and Area evaluation of an XOR gate.

Adder blocks	Delay	Area
XOR	3	5
2:1 Mux	3	4
Half adder	3	6
Full adder	6	13

Table .1. Delay and area count of the basic blocks of CSLA

As stated above the main idea is to use BEC instead of the RCA in order to reduce the area and power consumption of the regular CSLA. A structure and the function table of a 4-b BEC are shown in Figure. 2 and Table 2, respectively.

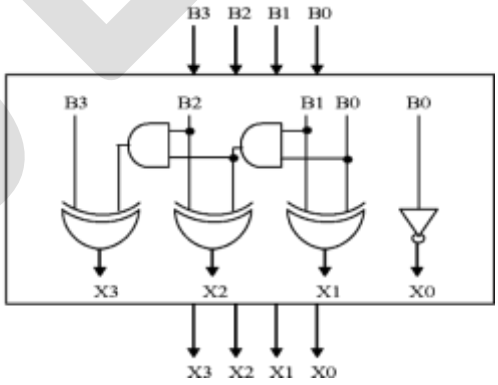


Figure.3. 4-b BEC

B[3:0]	X[3:0]
0000	0001
0001	0010
⋮	⋮
1110	1111
1111	0000

Table 2. Function Table of the 4-b BEC

Figure. 3 illustrates how the basic function of the CSLA is obtained by using the 4-bit BEC together with the mux. One input of the 8:4 mux gets as it input (B3, B2, B1, and B0) and another input of the mux is the BEC output. This produces the two possible partial results in parallel and the mux is used to select either the BEC output or the direct inputs according to the control signal Cin. The importance of the BEC logic stems from the large silicon area reduction when the CSLA with large number of bits are designed.

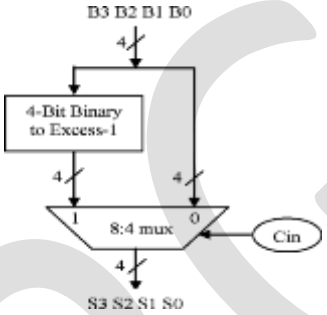


Figure. 4. 4-b BEC with 8:4 mux.

3.2. DELAY AND AREA EVALUATION METHODOLOGY OF REGULAR 16-B SQRT CSLA

The structure of the 16-b regular SQRT CSLA is shown in Figure.4. It has five groups of different size RCA. The delay and area evaluation of each group are shown in Figure.5, in which the numerals within [] specify the delay values, e.g., sum2 requires 10 gate delays. The group2 [see Figure. 5(a)] has two sets of 2-b RCA. Based on the consideration of delay values of Table I, the arrival time of selection input of 6:3 mux is earlier. Except for group2, the arrival time of mux selection input is always greater than the arrival time of data outputs from the RCA's. The one set of 2-b RCA in group2 has 2 FA and the other set has 1 FA and 1 HA . Similarly, the estimated maximum delay and area of the other groups in the regular SQRT CSLA are evaluated and listed in Table 3.

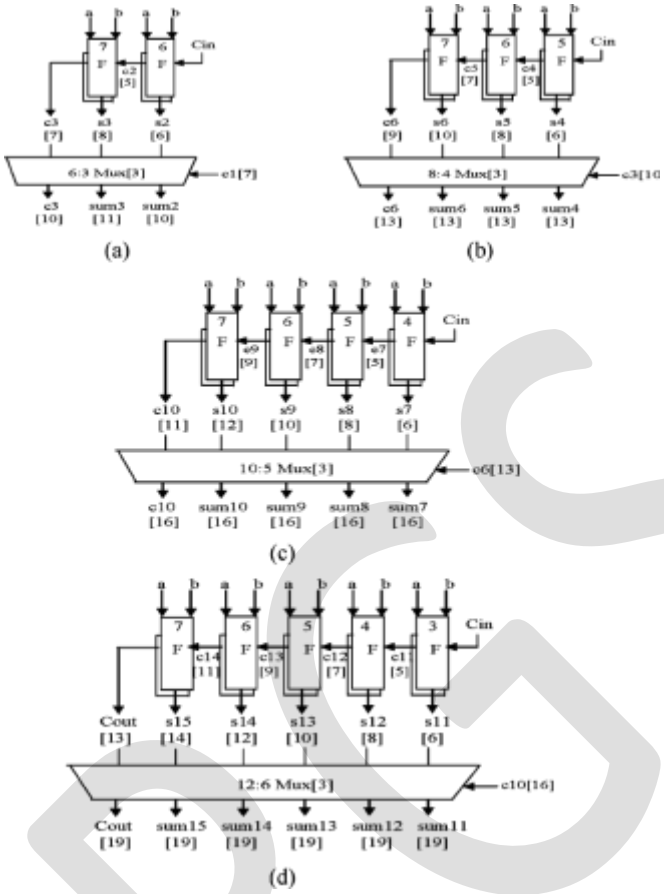


***Figure. 6. Delay and area evaluation of regular SQRT CSLA: (a) group2, (b) group3, (c) group4, and (d) group5. F is a Full* Adder.**

Group	Delay	Area
Group2	11	57
Group3	13	87
Group4	16	117
Group5	19	147

Table .3. Delay and area count of regular SQRT CSLA groups

3.3 SQRT CSLA USING COMMON BOOLEAN LOGIC

In the design of Integrated Circuits, area occupancy plays a vital role because of increasing the necessity of portable systems. Carry Select Adder (CSLA) is one of the fastest adders used in many data-processing processors to perform fast arithmetic functions. After logic simplification and sharing partial circuit, only one XOR gate and one inverter gate in each summation operation as well as one AND gate and one inverter gate in each carry-out operation are needed. Through the multiplexer, the correct output is selected according to the logic states of the carry in signal. Based on this modification a new architecture has been developed and compared

with the regular and modified Square-root CSLA (SQRT CSLA) architecture. The modified architecture has been developed using Binary to Excess-1 converter (BEC). The proposed architecture has reduced area and delay as compared with the regular SQRT CSLA architecture.

An area-efficient carry select adder by sharing the common Boolean logic term to remove the duplicated adder cells in the conventional carry select adder is shown and it saves many transistor counts and achieves a low power. Through analyzing the truth table of a single bit full adder, to find out the output of summation signal as carry-in signal is logic '0' is the inverse signal of itself as carry-in signal is logic '1'. By sharing the common Boolean logic term in summation generation, a proposed carry select adder design is generated. To share the common Boolean logic term, it only needs to implement one OR gate with one INV gate to generate the carry signal and summation signal pair. Once the carry-in signal is ready, then select the correct carry-out output according to the logic state of carry-in signal. This method replaces the BEC add one circuit by Common Boolean Logic. The summation and carry signal for full adder which has Cin=1, generate by INV and OR gate. Through the multiplexer, the correct output result is selected according to the logic state of carry-in signal.. One input to the mux goes from ripple carry adder block with Cin=0 and other input from the Common Boolean logic.

While analyzing the truth table of single bit full adder, results show that the output of summation signal as carry-in signal is logic "0" is inverse signal of itself as carry-in signal is logic "1". It is illustrated by circles in Table 6. To share the Common Boolean Logic term, we only need to implement a XOR gate and one INV gate to generate the summation pair. And to generate the carry pair, we need to implement one OR gate and one AND gate. In this way, the summation and carry circuits can be kept parallel.

C_{in}	A	B	S0	C0
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Table .4. Truth table of single bit full adder , where the upper half part is the case of cin=0 and the lower half part is the case of cin=1

This method replaces the Binary to Excess-1 converter add one circuit by common Boolean logic. As compared with modified SQRT CSLA, the proposed structure is little bit faster. Internal structure of proposed CSLA is shown in Figure. 8.In the proposed SQRT CSLA, the transistor count is trade-off with the speed in order to achieve lower power delay product. Thus the proposed SQRT CSLA using CBL is better than all the other designed adders. Figure.9 shows the block diagram of Proposed SQRT CSLA.

3.4 AREA DELAY POWER EFFICIENT CARRY SELECT ADDER

LOW-POWER, area-efficient, and high-performance VLSI systems are increasingly used in portable and mobile devices, multi standard wireless receivers, and biomedical instrumentation. 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 the anticipated

input-carry (cin = 0 and 1) and selects one out of each pair for final-sum and final-output-carry. 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. 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. The BEC-based CSLA involves less logic resources than the conventional CSLA, but it has marginally higher delay. The CBL-based CSLA involves significantly less logic resource than the conventional CSLA but it has longer CPD, which is almost equal to that of the RCA. The CBL-based SQRTCSLA design requires more logic resource and delay than the BEC-based SQRT-CSLA. 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.

- **LOGIC FORMULATION**

The CSLA has two units:
1) the sum and carry generator unit (SCG)
2) the sum and carry selection unit.

The SCG unit consumes most of the logic resources of CSLA and significantly contributes to the critical path. 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 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 dependence.

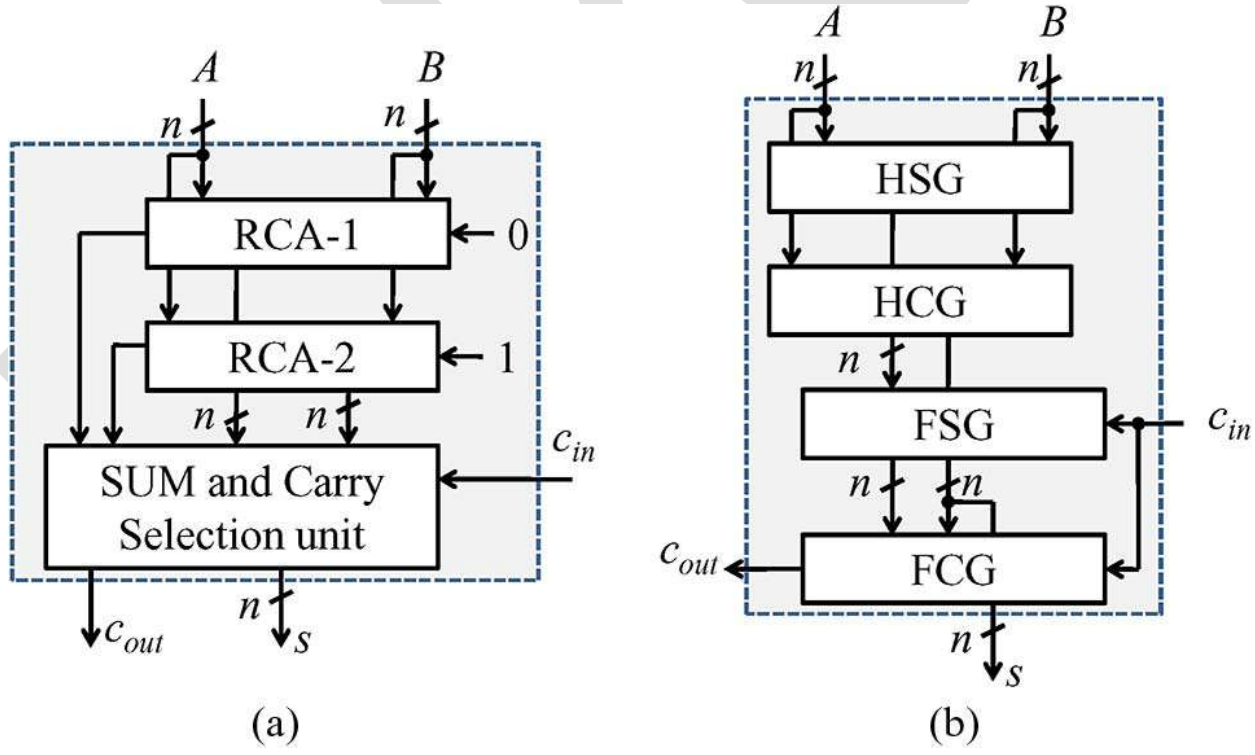


Figure. 7. (a) Conventional CSLA; n is the input operand bit-width. (b) The logic operations of the RCA is shown in split form, where HSG, HCG, FSG, and FCG represent half-sum generation, half-carry generation, full-sum generation, and full-carry generation, respectively.

- **Logic Expressions of the SCG Unit of the Conventional CSLA**

As shown in Figure.10 (a), the SCG unit of the conventional CSLA 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)
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 out and c1 out) 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

$$s_0^0(i) = A(i) \oplus B(i) \quad c_0^0(i) = A(i) \cdot B(i) \tag{1a}$$
$$s_1^0(i) = s_0^0(i) \oplus c_1^0(i-1) \tag{1b}$$
$$c_1^0(i) = c_0^0(i) + s_0^0(i) \cdot c_1^0(i-1) \quad c_{\text{out}}^0 = c_1^0(n-1) \tag{1c}$$
$$s_0^1(i) = A(i) \oplus B(i) \quad c_0^1(i) = A(i) \cdot B(i) \tag{2a}$$
$$s_1^1(i) = s_0^1(i) \oplus c_1^1(i-1) \tag{2b}$$
$$c_1^1(i) = c_0^1(i) + s_0^1(i) \cdot c_1^1(i-1) \quad c_{\text{out}}^1 = c_1^1(n-1) \tag{2c}$$

where $c_1^0(-1) = 0$, $c_1^1(-1) = 1$, and $0 \le i \le n-1$.

As shown in (1a)–(1c) and (2a)–(2c), the logic expression of {s00(i), c00(i)} is identical to that of {s10(i), c10(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, we have used an add-one circuit instead of RCA-2 in the CSLA, in which a BEC circuit is used 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.

Logic Expression of the SCG Unit of the BEC-Based CSLA

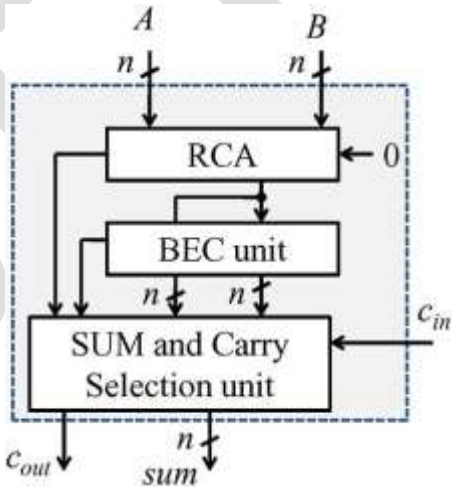


Figure. 8. Structure of the BEC-based CSLA; n is the input operand bit-width.

As shown in Figure.11, the RCA calculates n-bit sum s01 and c0 out corresponding to cin = 0. The BEC unit receives s01 and c0 out from the RCA and generates (n + 1)-bit excess-1 code. The most significant bit (MSB) of BEC represents c1out, in which n least significant bits (LSBs) represent s11 . 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

$$s_1^1(0) = \overline{s_1^0(0)} \quad c_1^1(0) = s_1^0(0) \tag{3a}$$
$$s_1^1(i) = s_1^0(i) \oplus c_1^1(i-1) \tag{3b}$$
$$c_1^1(i) = s_1^0(i) \cdot c_1^1(i-1) \tag{3c}$$
$$c_{\text{out}}^1 = c_1^0(n-1) \oplus c_1^1(n-1) \tag{3d}$$

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

We can find from (1a)–(1c) and (3a)–(3d) that, in the case of the BEC-based CSLA, c11 depends on s01, which otherwise has no dependence on s01 in the case of the conventional CSLA. 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 s01 and s11 are identical except the terms c01 and c11since (s00 = s10 = s0). In addition, we find that c01 and c11depend on {s0, c0, cin}, where c0 = c00 = c10 . Since c01 and c11 have no dependence on s01 and s11 , the logic operation of c01 and c11 can be scheduled before s01 and s11 , and the select unit can select one from the set (s01 , s11) for the final-sum of the CSLA. We find that a significant amount of logic resource is spent for calculating {s01 , s11 }, 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 {c0 and c1} to calculate the final-sum. The selected carry word is added with the half-sum (s0) to generate the final-sum (s). Using this method, one can have three design advantages:

1) Calculation of s01 is avoided in the SCG unit;
2) the n-bit select unit is required instead of the (n + 1) bit;
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

$$s_0(i) = A(i) \oplus B(i) \quad c_0(i) = A(i) \cdot B(i) \tag{4a}$$
$$c_1^0(i) = c_1^0(i-1) \cdot s_0(i) + c_0(i) \quad \text{for } (c_1^0(0) = 0) \tag{4b}$$
$$c_1^1(i) = c_1^1(i-1) \cdot s_0(i) + c_0(i) \quad \text{for } (c_1^1(0) = 1) \tag{4c}$$
$$c(i) = c_1^0(i) \quad \text{if } (c_{\text{in}} = 0) \tag{4d}$$
$$c(i) = c_1^1(i) \quad \text{if } (c_{\text{in}} = 1) \tag{4e}$$

- **PROPOSED ADDER DESIGN**

The proposed CSLA is based on the logic formulation given in (4a)–(4e), and its structure is shown in Figure. 12(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 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.

The logic diagram of the HSG unit is shown in Figure. 12(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 Figure. 12(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 cin. It selects c01 when cin = 0; otherwise, it selects c11. The CS unit can be implemented using an n-bit 2-to-l MUX. However, we find from the truth table of the CS unit that carry words c01 and c11 follow a specific bit pattern. If c01 (i) = '1', then c11 (i) = 1, irrespective of s0(i) and c0(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 in Figure. 12(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 cout,

and (n − 1) LSBs are XORed with (n − 1) MSBs of half-sum (s0) in the FSG [shown in Figure. 12(f). to obtain (n − 1) MSBs of final-sum (s). The LSB of s0 is XORed with cin to obtain the LSB of s.

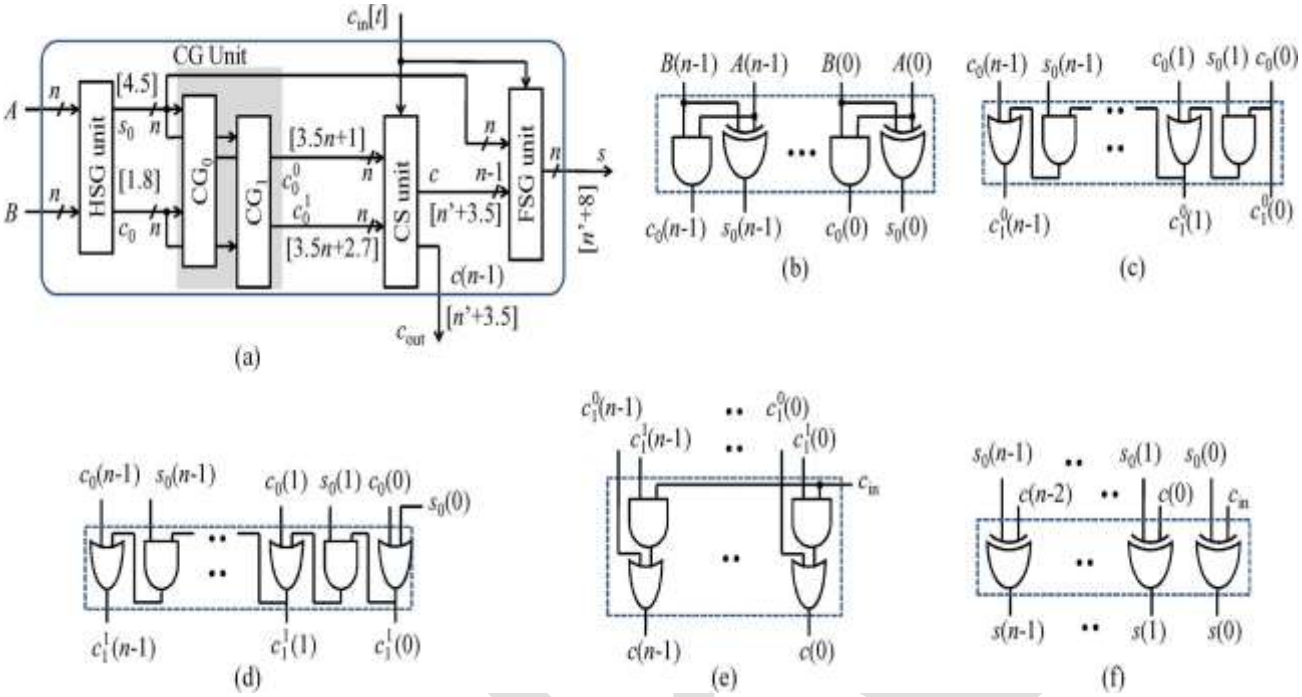


Figure. 9. (a) Proposed CS adder design, where n is the input operand bit-width, and [∗] represents delay (in the unit of inverter delay), n = max(t, 3.5n + 2.7).
(b) Gate-level design of the HSG.
(c) Gate-level optimized design of (CG0) for input-carry = 0.
(d) Gate-level optimized design of (CG1) for input-carry = 1.
(e) Gate-level design of the CS unit.
(f) Gate-level design of the final-sum generation (FSG) unit.

- **PERFORMANCE COMPARISON**

- ***Area–Delay Estimation Method***

We have considered 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 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:

$$A = a \cdot N_a + r \cdot N_o + i \cdot N_i \qquad (5a)$$

$$T = n_a \cdot T_a + n_o \cdot T_o + n_i \cdot T_i \qquad (5b)$$

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. We have calculated the (AOI) gate counts of each design for area and delay estimation. Using (5a) and (5b), the area and delay of each design are calculated from the AOI gate counts (Na,No,Ni), (na, no, ni), and the cell details of Table B.

- ***Single-Stage CSLA***

The general expression to calculate the AOI gate counts of the n-bit proposed CSLA and the BEC-based CSLA and CBL-based CSLA are given in Table II of single stage design. We have calculated the AOI gate counts on the critical path of the proposed n-bit CSLA and CSLAs of and used those AOI gate counts in (5b) to find an expression for delay of final-sum and output-carry in the unit of Ti (NOTgate delay). The delay of the n-bit single-stage CSLA is shown in Table II for comparison. For further analysis of the critical path of the proposed CSLA, the delay of each intermediate and output signals of the proposed n-bit CSLA design of Fig. 3 is shown in the square bracket against each signal. We can find from Table II that the proposed n-bit single-stage CSLA adder involves 6n less number of AOI gates than the CSLA of [6] and takes 2.7 and 6.6 units less delay to calculate final-sum and output-carry. Compared with the CBL-based CSLA of [7], the proposed CSLA design involves n more AOI gates, and it takes (n − 4.7) unit less delay to calculate the output-carry. Using the expressions of Table II and AOI gate details of Table I, we have estimated the area and delay complexities of the proposed CSLA and the existing CSLA of [6]–[8], including the conventional one for input bit-widths 8 and 16. For the single-stage CSLA, the input-carry delay is assumed to be t = 0 and the delay of final-sum (fs) represents the adder delay. The estimated values are listed in Table III for comparison. We can find from Table III that the proposed CSLA involves nearly 29% less area and 5% less output delay than that of [6]. Consequently, the CSLA of [6] involves 40% higher ADP than the proposed CSLA, on average, for different bit-widths. Compared with the CBL-based CSLA of [7], the proposed CSLA design has marginally less ADP. However, in the CBL-based CSLA, delay increases at a much higher rate than the proposed CSLA design for higher bitwidths. Compared with the conventional CSLA, the proposed CSLA involves 0.42 ns more delay, but it involves nearly 28% less ADP due to less area complexity. Interestingly, the proposed CSLA design offers multipath parallel carry propagation, whereas the CBL-based CSLA of [7] offers a single carry propagation path identical to the RCA design. Moreover, the proposed CSLA design has 0.45 ns less output-carry delay than the output-sum delay. This is mainly due to the CS unit that produces output-carry before the FSG calculates the final-sum.

3.5 PROPOSED DESIGN OF SQRT CSLA

The proposed SQRT CSLA is developed with the help of modified half adder (HAM), modified full adder (FAM) and modified XOR gate (XORM). In place of BEC, combinational logic block (CLB) is used. XORM has 1 gate less than the conventional XOR gate of 5 gates (AND-OR-NOT implementation) [8] as in Fig.3. HAM has 2 gates less than the conventional half adder as shown in Fig.4. The full adder is constructed with two HAMs and an AND gate [8] shown in Fig.5 has only 9 gates, 4 gates less than conventional full adder. As the number of gates reduce in the basic building blocks of the proposed SQRT CSLA area is also reduced. A part from the above modifications, one more small change is made in the design which allows us to get the carryout of the group without using the mux [7]. The proposed design SQRT CSLA with CLB is as shown in Fig.6. A 16-bit model is presented which is divided into 5 groups. Each group consists of different size of RCAs, CLBs and multiplexers. The CLB is used instead of RCA for Cin = 1. The structure of a 4-bit CLB is given in Fig.7. The sizes of RCA differ from 2-bit in first two groups and consequently increase to 5-bit in group 5. Similarly the size of CLB also increases from 3-bit in group 2 to 6-bit in group 5. The groups in the proposed adder are shown in Fig.8.

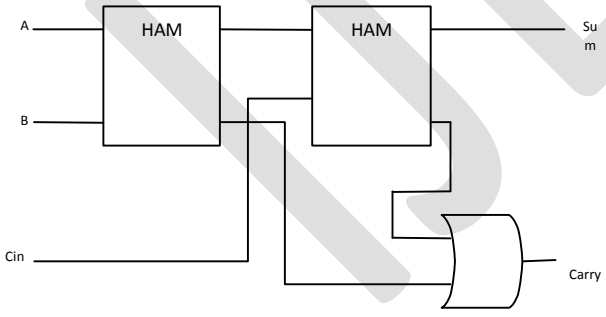


Figure.10. Modified full adder (FAM) using two HAMs

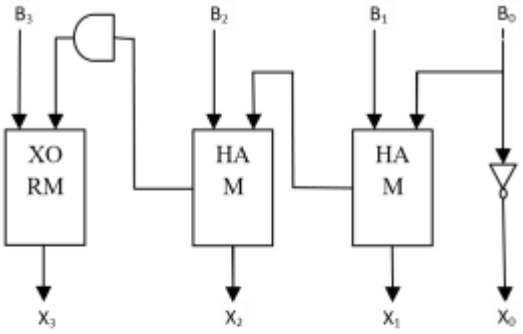
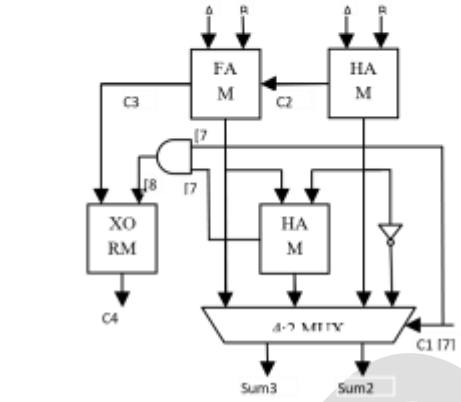
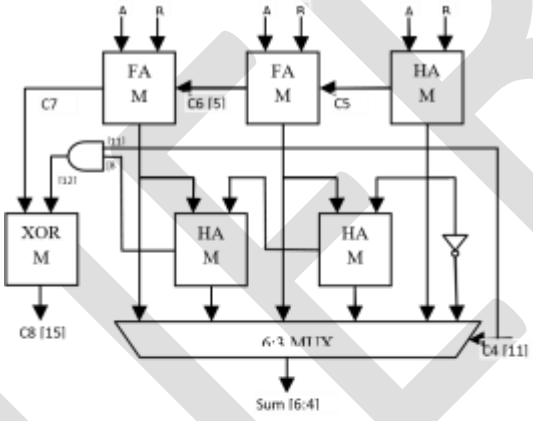


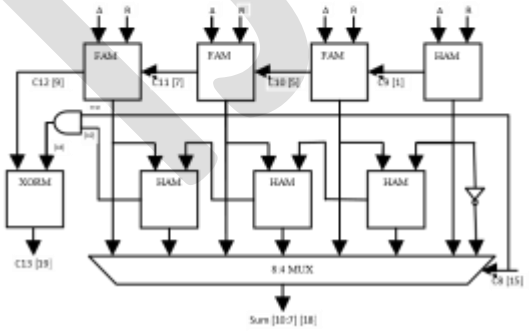
Fig7. 4-bit CLB



(a)



(b)



(c)

Fig.11. (a) group 2, (b) group3, (c) group4 of SQRT CSLA with CLB

Group 2 of the proposed adder consists of 2-bit RCA, 3-bit CLB and 4:2 mux. RCA is constructed using one FAM and one HAM. The CLB has one NOT gate, one HAM, one AND gate and a XORM. For the remaining groups the size of RCA and CLB is listed in Table I.

Groups	RCA		Combinational logic block (CLB)			Total no. of gates
	FAM	HAM	NOT gate +AND gate	HAM	XORM	
Group1	2(*9)	0	0	0	0	18
Group 2	1(*9)	1(*4)	1+1	1(*4)	1(*4)	31
Group 3	2(*9)	1(*4)	1+1	2(*4)	1(*4)	48
Group 4	3(*9)	1(*4)	1+1	3(*4)	1(*4)	59
Group 5	4(*9)	1(*4)	1+1	4(*4)	1(*4)	82

Table.9. Gate Count of SQRT CSLA with CLB

Word size	Adders	Delay (ηs)	Area (μm^2)	Total Power (μW)	Powerdelay product (10^{-15}) (J)	Areadelay product (10_{-15})
8-bit	SQRT CSLA	1.68	1035	76.687	129.2954	1745.01
	SQRT CSLA with BEC	1.82	888	66.735	121.4579	1616.16
	SQRT CSLA with CLB	1.85	845	53.186	98.394	1563.25
16-bit	SQRT CSLA	2.903	2259	171.797	498.729	6557.87
	SQRT CSLA with BEC	2.858	1873	139.209	397.859	5353.03
	SQRT CSLA with CLB	3.524	1866	116.850	411.779	6609.37
32-bit	SQRT CSLA	4.310	4787	390.199	1681.758	20631.97
	SQRT CSLA with BEC	4.405	3922	307.931	1356.438	17276.41
	SQRT CSLA with CLB	5.206	3895	253.098	1317.628	20277.37
64-bit	SQRT CSLA	7.119	9883	829.125	5902.543	70357.08
	SQRT CSLA with BEC	6.970	8007	628.192	4378.499	55808.79
	SQRT CSLA with CLB	7.978	7973	545.083	4348.621	63608.59

Table.5. Comparison of SQRT CSLA, SQRT CSLA with BEC and SQRT CSLA with CLB

ACKNOWLEDGMENT

First of all I would like to thank GOD, the Almighty for His divine grace and blessings throughout this Thesis work.

I am greatly indebted to my academic mentors, whose support has given me the confidence to make a study on the topic and present the Thesis work. I hereby express my heart full gratitude to Prof. Sunny Joseph, Head of the Department of Electronics and communication, for being a great source of inspiration.

I will remain indebted to my guide, Mr. Thomas George, Associate Professor in Electronics and communication Department, for his valuable guidance and help extended to me. His guidance enabled me to complete study of the topic in time and present it well. My deep sense of gratitude goes to all teachers of Electronics and communication department who gave valuable support and guidance.

I express my gratitude to the college management and our principal Prof. Geetha B for providing us good library facilities and internet facilities, that helped me a lot for the study of topic in detail.

I would also like to extend my sincere thanks to my family and friends for their whole-hearted support and encouragement.

CONCLUSION

We have analyzed the logic operations involved in the conventional and BEC-based CSLAs to study the data dependence and to identify redundant logic operations. We have 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.

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 carryselect 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] Akhilesh Tyagi, "A Reduced Area Scheme for Carry-Select Adders", IEEE International Conference on Computer design, pp.255-258, Sept 1990.

[11] O. J. Bedrij, "Carry-Select Adder", IRE transactions on Electronics Computers, vol.EC-11, pp. 340-346, June1962.

[12] Neil H.E.Weste and K.Eshraghian, "Principle Of CMOS VLSI designs: a system perspective", (Addison-Wesley, 1998), 2nd edn, 1998.

[13] Richard P. Brent and H. T. Kung, "A Regular Layout for Parallel Adders", IEEE transactions on Computers, vol.c-31, pp.260-264, March 1982.