A New Architecture Designed for Implementing Area Efficient Carry-Select Adder

D. Durgaprasad *1
Assistant Professor, Dept of ECE
Vishnu Institute of Technology
A.P, India

A. M. V.Pathi *2
Assistant Professor, Dept of ECE
Vishnu Institute of Technology
A.P, India

Ch. Rama Krishna *3
Associate Professor, Dept of ECE
Vishnu Institute of Technology
A. P, India

C V Pradeep Kumar Reddy *4
Assistant Professor, Dept of ECE
Vishnu Institute of Technology
A.P, India

Abstract: In this work, We have analyze the Conventional Carry Select Adder (CSLA) and Binary to Excess-1 converter (BEC) based CSLA contained logic operations and recognize the superfluous logic operations. The conventional CSLA contained superfluous logic operations are removed and introduce new logic operations for carry select adder. The carry select (CS) operation is performed before the calculation of final sum in newly introduced logic operations. We modify the design of carry select adder using newly introduced logic operations based logic units. From the synthesis results, the modified structure of CSLA yields significantly less area than the conventional CSLA and BEC-based CSLA. Due to the small carry-output delay, the modified CSLA design is best suitable for square-root (SQRT) CSLA.

Keywords: Adder, VLSI design, Arithmetic Unit.

1. INTRODUCTION

Low-power, area-efficient and high-speed VLSI systems are increasingly used in portable and mobile devices, biomedical instrumentation [2], [3] and multi standard wireless receivers. Addition is an obligatory operation that is crucial to processing the fundamental arithmetic operations. It is used extensively in many VLSI design hypothesis and is by far the most frequently used operation in a general purpose systems and in application specific processors, because the operations of subtraction, multiplication, division and address calculation usually relay on the operation of addition. Addition is often seen as an indispensable part of the arithmetic unit. It is dubbed the heart of any microprocessor, DSP architecture and data processing system.

A complex digital signal processing (DSP) system performance is basically improved using an efficient adder design. The ripple carry adder (RCA) design occupies small area, but carry propagation delay (CPD) is large for this adder. The carry propagation delay of the adders is reduced using Carry look-ahead and carry select (CS) methods.

The conventional carry select adder (CSLA) structure contains two ripple carry adders (RCA) that generates a pair of sum words and output carry bits corresponding to the anticipated input-carry (Cin=0 and 1) and selects one out of each pair of final-sum and final-output-carry [4]. A conventional CSLA has less CPD than an RCA, but the design is not attractive since it uses a dual RCA. Several designs are proposed to avoid dual use of RCA in CSLA design. Kim and Kim [5] introduce a new structure that contains 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.[6] proposed a square-root (SQRT)-CSLA for implementation of large bit-width adders with less delay. In a SQRT CSLA, CSLAs are connected in a cascading structure with increasing size. The main intention of SQRT-CSLA design is to provide a parallel path for carry propagation that helps to decrease the overall adder delay. Ramkumar and Kittur [7] introduce BEC-based CSLA. The BEC-based CSLA involves less logic resources than the conventional CSLA, but it has slightly higher delay. Common Boolean logic (CBL) based CSLA is also proposed in [8] and [9]. The CBL-based CSLA of [8] involves less logic resource than the conventional CSLA but it has higher CPD, which is equal to the RCA. To overcome this problem, a SQRT-CSLA based on CBL was proposed in [9]. The CBL-based SQRT-CSLA design of [9] involves more logic resource and delay than the BEC-based SQRT-CSLA of [7]. We examine that the logic optimization largely depends on availability of redundant operations in the logic formulation, whereas adder delay largely depends on data dependence. In the existing designs, logic is optimized without considering the data dependence. In this work, We analyze the Conventional Carry Select Adder (CSLA) and Binary to Excess-1 converter (BEC) based CSLA contained logic operations and recognize the superfluous logic operations. The conventional CSLA contained superfluous logic operations are removed and introduce new logic operations for carry select adder. In this work, carry generator (CG) and CS units are optimized using newly introduced logic operations. We modify the design of carry select adder using optimized logic units. Due to optimized logic units, the new CSLA design involves significantly less area than the existing CSLAs.
II. LOGIC FORMULATION
The Carry Select Adder (CSLA) has two units:
1) The sum and carry generator unit (SCG)
2) The sum and carry selection unit

The most of the logic operations of the CSLA are present in SCG unit. Different structures have been proposed for SCG unit implementation. We analyze the conventional CSLA and BEC based CSLA contained SCG unit logic designs. The major aim of this analysis is to recognize the superfluous logic operations and data dependence. In view of that, we eliminate all superfluous logic operations.

A) Logic Operations Involved in Conventional CSLA
The conventional structure of the CSLA [4] is shown in Fig.1(a). The conventional CSLA structure contains two n-bit RCAs, where n is the adder bit-width. The n-bit RCA contained logic operations are performed in four stages that can be shown in Fig.1(b): They are

1) Half-sum generation (HSG)
2) Half-carry generation (HCG)
3) Full-sum generation (FSG)
4) Full-carry generation (FCG)

For example the conventional CSLA perform addition operation on two n-bit operands. The RCA-1 generates n-bit sum \( s^0 \) and output-carry \( c^0_{\text{out}} \) for input carry \( c_{\text{in}} = 0 \). The RCA-2 generates the n-bit sum \( s^1 \) and output carry \( c^1_{\text{out}} \) for input carry \( c_{\text{in}} = 1 \). The logic expressions of the n-bit CSLA contained SCG unit are given as

\[
\begin{align*}
    s^0_0(j) &= A(j) \oplus B(j) & c^0_0(j) &= A(j) \cdot B(j) & (1a) \\
    s^0_1(j) &= s^0_0(j) \oplus c^1_1(j - 1) & (1b) \\
    c^0_1(j) &= c^0_0(j) + s^0_0(j) \cdot c^0_0(j - 1) & c^0_{\text{out}} &= c^0_0(n - 1) & (1c) \\
    s^1_0(j) &= A(j) \oplus B(j) & c^1_0(j) &= A(j) \cdot B(j) & (2a) \\
    s^1_1(j) &= s^1_0(j) \oplus c^1_1(j - 1) & (2b) \\
    c^1_1(j) &= c^1_0(j) + s^1_0(j) \cdot c^1_0(j - 1) & c^1_{\text{out}} &= c^1_1(n - 1) & (2c)
\end{align*}
\]

Where \( c^1_1(-1) = 0, c^1_1(-1) = 1, 0 \leq j \leq n - 1 \).

From the equations (1a)–(1c) and (2a)–(2c), the logic operations of \( \{s^0_0(j), c^0_0(j)\} \) is similar to that of \( \{s^1_0(j), c^1_0(j)\} \). The design of the RCA-2 is optimized by removing redundant logic operations and constructs the RCA-2 by sharing the HSG &HCG of RCA-1. In [4], [5] the CSLA contains an add-one circuit instead of RCA-2. In [6] the CSLA contains BEC circuit in place of RCA-2. The Area-Delay-Power efficiency of the BEC based CSLA is better than the existing CSLAs, So the logic expressions of the BEC based CSLA contained SCG unit is discussed.

B) Logic operations involved in BEC based CSLA
The structure of the BEC based CSLA is shown in Fig. 2. It contains the n-bit RCA and BEC unit. The RCA calculates the n-bit sum \( s^0 \) and output carry \( c^0_{\text{out}} \) for given input carry \( c_{\text{in}} = 0 \). The sum \( s^0 \) and carry \( c^0_{\text{out}} \) are given to the BEC unit and obtains \( (n + 1) \)-bit excess-1 code. The most significant bit (MSB) of the BEC represent the carry \( c^0_{\text{out}} \) and n-least significant bits (LSBs) of the BEC represent the sum \( s^1 \).
The logic operations of the RCA of the BEC based CSLA structure are the same as those given in (1a)–(1c). The logic operations of the CSLA contained BEC unit are given as
\[ s_1^0(0) = s_0^0(0) \]
\[ c_1^0(j) = s_0^0(j) \oplus c_1^0(j-1) \]
For \( 1 \leq j \leq n-1 \)
\[
\begin{align*}
\text{We can observe from equations (1a)–(1c) and equations (3a)–(3d), the carry } c_1^0 \text{ of the BEC based CSLA depends on } s_0^0, \text{ but in conventional CSLA carry } c_1^0 \text{ is not depends on } s_1^0. \text{ The data dependence in CSLA increases with the BEC method. We can perform the analysis on logic operations of the CSLA and identify the superfluous logic operations. The logic units of the CSLA are optimized by removing the superfluous logic operations.}
\end{align*}
\]

We can observe from equations (1a)–(1c) and (2a)–(2c), in that logic expressions of \( s_0^0 \) and \( s_1^0 \) are equal except the carry \( c_0^0 \) and carry\( c_1^0 \), so that \( s_0^0 = s_0 = s_0 \). We also observe, the \( c_0^0 \) and \( c_1^0 \) depend on \( s_0 c_0 c_{in} \), where \( c_0 = c_0^0 = c_0 \). The carriers \( c_0^0 \) and \( c_1^0 \) are not depend on \( s_0^0 \) [and\( s_1^0 \)], so the logic operations of \( c_0^0 \) and \( c_1^0 \) can be performed before \( s_0^0 \) and \( s_1^0 \). The select unit of the CSLA selects final sum from the set \( (s_0^0, s_1^0) \). We find that, in order to calculate the \( (s_0^0, s_1^0) \) spent significant amount of logic resource and after calculation the rejection of one sum word is not an efficient method. Instead, one can perform the calculation of the final sum by selecting carry word from the anticipated carry words and \( c_1 \). The half-sum \( (s_0) \) is added with selected carry word to obtain the final-sum \( (s) \). This method has three design advantages: 1) The \( s_0^0 \) calculation is not performed in the SCG unit 2) The n-bit select unit is needed in place of the \( (n + 1) \) bit select unit 3) The delay of the output-carry is minimum. All these features result in an area–delay and energy-efficient design for the CSLA. The redundant logic operations of equations (1a)–(1c) and (2a)–(2c) are eliminated and rearranged the logic operations of equations (1a)–(1c) and (2a)–(2c) based on their data dependence. The newly introduced logic operations for the CSLA is given as
\[
\begin{align*}
s_0^0(j) &= A(j) \oplus B(j) \\
c_0^0(j) &= A(j) \cdot B(j) \\
c_1^0(j) &= c_1^0(j - 1) \cdot s_0(j) + c_0(j) \text{ for } (c_1^0(0) = 0) \\
c_1^0(j) &= c_1^0(j - 1) \cdot s_0(j) + c_0(j) \text{ for } (c_1^0(0) = 1) \\
c(j) &= c_1^0(j) \text{ if } (c_{in} = 0) \\
c(j) &= c_1^0(j) \text{ if } (c_{in}) \\
c_{out} &= c(n - 1) \\
s(0) &= s_0(0) \oplus c_{in} \text{ for } (i = s_0(i) \oplus c(i-1) \text{ for } (i = 1)
\end{align*}
\]
Depending on control signal $c_{in}$, one final carry word is selected using CS unit from the two carry words available at its input. If $c_{in}$=0, it selects $c^0$ otherwise, it selects $c^1$. An n-bit 2-to-1 MUX is used to implement the CS unit. However, we observe from the CS unit truth table, carry words $c^1(j)$ and $c^1(j)$ follow a specific bit pattern. If $c^1(j) = '1'$, then $c^1(j) = 1$, irrespective of $s_0(j)$ and $c_0(j)$, for $0 \leq j \leq n - 1$. The CS unit logic optimization uses this feature. The optimized gate level design of the CS unit is shown in Fig. 7, which is composed of n AND–OR gates. The CS unit generates the final carry word $c$. The output carry $c_{out}$ is the MSB of $c$ and (n - 1) LSBs of $c$ are XORed with (n - 1) MSBs of half-sum ($s_0$) in the FSG and obtains (n - 1) MSBs of final-sum(s). The LSB of half sum ($s_0$) is XORed with $c_{in}$ to acquire the LSB of $s$. The structure of the Final Sum Generator (FSG) is shown in Fig. 8.

The multipath carry propagation feature of the CSLA is fully oppressed in the SQRT-CSLA [6], which is collected of a chain of CSLAs. CSLAs with increasing size are used in the SQRT-CSLA design to remove the most consensus in the carry propagation path. The large-size adders are realized using SQRT CSLA with less delay than the same size single-stage CSLA. However, the carry propagation delay between the CSLA stages of SQRT-CSLA is crucial for the overall adder delay. The modified CSLA design is more favourable for area delay efficient implementation of SQRT CSLA than the existing CSA designs, since the output carry is generated premature with multipath carry propagation feature. A 16-bit SQRT-CSLA design with modified CSLA structure is shown in Fig. 9. The 16-bit SQRT CSLA uses the 2-bit RCA, 2-bit CSLA, 3-bit CSLA, 4-bit CSLA and 5-bit CSLA. To exhibit the advantage of the modified CSLA design in SQRT-CSLA, we calculate the area and delay of the modified CSLA design based SQRT-CSLA and the conventional CSLA based SQRT-CSLA for bit-widths 16. The modified CSLA design based SQRT-CSLA design has less area than the existing SQRT-CSLA.
V. RESULTS AND PERFORMANCE ANALYSIS

We have verified the modified structure and existing structures of carry select adder by writing VHDL code, Simulated and Synthesized.

The simulation result of the carry select adder (CSLA) is shown in Fig.10.

![Fig.10. Simulation Result of the CSLA](image)

Table I and Fig.11 shows the comparison between the modified CSLA design, Conventional CSLA design and BEC based CSLA design in terms of Delay, Number of LUTs and Number of Slices.

<table>
<thead>
<tr>
<th>Design</th>
<th>Delay (ns)</th>
<th>No. of LUTs</th>
<th>No. of Slices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conventional CSLA</td>
<td>10.382</td>
<td>27</td>
<td>15</td>
</tr>
<tr>
<td>BEC Based CSLA</td>
<td>13.767</td>
<td>27</td>
<td>15</td>
</tr>
<tr>
<td>Modified CSLA</td>
<td>12.738</td>
<td>25</td>
<td>14</td>
</tr>
</tbody>
</table>

![Fig.11. Comparison Graph of CSLA structures](image)

VI. CONCLUSION

We have also verified the modified CSLA based 16-bit SQRT CSLA design and conventional CSLA based 16-bit SQRT CSLA by writing VHDL code, Simulated and Synthesized.

The simulation result of the SQRT CSLA is shown in Fig.12.

![Fig.12. Simulation Result of the SQRT CSLA](image)

Table II and Fig.13 shows the comparison between the modified CSLA based 16-bit SQRT CSLA design, Conventional CSLA based 16-bit SQRT CSLA design in terms of Delay, Number of LUTs and Number of Slices.

<table>
<thead>
<tr>
<th>Design</th>
<th>Delay (ns)</th>
<th>No. of LUTs</th>
<th>No. of Slices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conventional CSLA Based SQRT CSLA</td>
<td>15.434</td>
<td>37</td>
<td>22</td>
</tr>
<tr>
<td>Modified CSLA Based SQRT CSLA</td>
<td>16.33</td>
<td>34</td>
<td>21</td>
</tr>
</tbody>
</table>

![Fig.13. Comparison Graph of SQRT CSLA structures](image)
We analyzed the Conventional Carry Select Adder (CSLA) and Binary to Excess-1 converter (BEC) based CSLA contained logic operations and recognize the superfluous logic operations. The conventional CSLA contained superfluous logic operations are removed and introduce new logic operations for carry select adder. The carry select (CS) operation is performed before the calculation of final sum in newly introduced logic operations. We modify the design of carry select adder using newly introduced logic operations based logic units. From the synthesis results, the modified structure of CSLA yields significantly less area than the conventional CSLA and BEC-based CSLA. Due to the small carry-output delay, the modified CSLA design is most suitable for square-root (SQRT) CSLA. The modified CSLA based SQRT CSLA design yields less area than the existing SQRT CSLA designs.

REFERENCES