Volume 6, Issue 6, June 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

VHDL Architecture for Delay Efficient SQRT Carry Select Adder

[1]Varsha Viswam*, [2]Suchithra S Nair
[1] PG Student, [2] Assistant Professor
[1, 2] Department of Electronics and Communication, Rajadhani Institute of Engineering and Technology, Trivandrum, Kerala University, Kerala, India

Abstract— Carry Select Adders (CSLA) are faster than any other adders. It is used in multipliers to perform addition operation. The important aspects while designing the adder is data dependency and redundant logic operations. The logic operations of Binary to Excess Carry -1 and conventional CSLA are studied and arrived at a logic formulation by studying the above aspects. In this proposed adder, the Carry Select (CS) is generated before the sum calculation. The input carry bits (c_{in}=0 and cin = 1) decides the carry and the sum. Due to small output carry delay, an efficient adder with less delay can be implemented. CSLAs have great importance when it has less delay. In this paper, the regular Square Root (SQRT) CSLA using VHDL architecture having less delay is proposed.

Keywords— Adder, Delay, SQRT CSLA, Binary to Excess Carry – 1, Carry Select,

I. INTRODUCTION

ADDERS have great importance in portable electronics devices, wireless receivers, mobiles, calculators, computers, other processors and similar applications. An electronics device with less power dissipation, less area and delay, grows faster in modern days. CSLAs are used in complex signal processing unit to perform faster addition. Ripple Carry Adder (RCA, the simplest but the slowest CSLA used for addition operation. RCA is made of full adders connected in series. The Carry Propagation Delay (CPD) is the main problem in RCA. Here carry is calculated only after the scheduling of sum.

The conventional CSLA uses two RCAs, one performs the addition operation with c_{in} = 0 and the other with c_{in} = 1. Conventional CSLA has less delay than RCA. It generates sum and carry bits according to input carry bits (c_{in}) and the anticipating carry bits. Then it selects each bit from the pair produced. The carry bits are selected by using Multiplexer. Since it used dual RCA and has large CPD, other methods having less CPD are designed. To avoid the dual RCA, Kim and Kim proposed new CSLA in which one RCA is replaced with add one circuit. Multiplexer is used to implement add-one circuit. Then he et.al proposed SQTR CSLA which reduces delay for large bit addition operation. The SQRT CSLA is one with increasing sizes are cascaded. This is done to provide parallel addition operation which makes the adder faster.

Ram Kumar and Kittur suggested a new CSLA which uses BEC. This BEC based CSLA uses less resources than the previous adder, but has a large delay. In BEC based CSLA, one CSLA with c_{in}= 1 is replaced with BEC circuit. Here the addition operation is done with c_{in} = 0. The BEC-1 circuit operates when the anticipated carry is 1. Basant Kumar Mohanty and Sujith Kumar Patel proposed a new formulation. They analyzed that the existing system does not consider logic optimization, which largely depends on data dependency. That is, logic resources and data dependency largely depends on adder characteristics such as delay, power dissipation and area. They formulate new formulation based on carry select units and data dependence.

In proposed adder the logic resource is analyzed and get into a new logic expression by avoiding redundant 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.

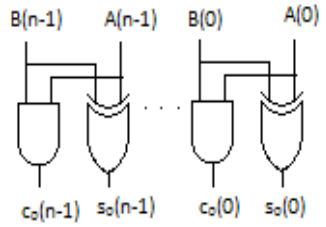

Fig. 1. Gate level design of HSCG

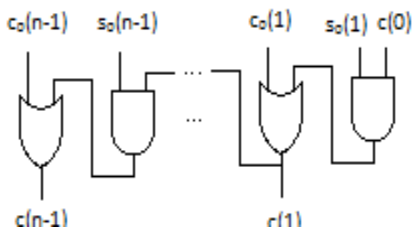

Fig. 2. Gate level design of FSCG (carry generation unit)

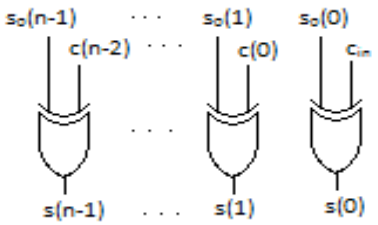

Fig. 3. Gate level design of FSCG (sum generation unit)

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

$s^0_0(i) = A(i) \oplus B(i)$; $c^0_0(i) = A(i).B(i)$ (1.1)
$s^0_1(i) = s^0_0(i) \oplus c^0_1(i-1)$; (1.2)
$c^0_1(i) = c^0_0(i) + s^0_0(i).c^1_0(i-1)$; $c^0_{out} = c^0_1(n-1)$ (1.3)
$s^1_0(i) = A(i) \oplus B(i)$; $c^1_0(i) = A(i).B(i)$ (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^0out) 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_1on conventional CSLA. Therefore, the BEC-1 based CSLA increases the data dependence in the conventional CSLA.

C. 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)

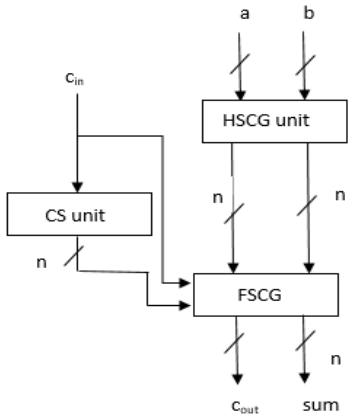

Fig. 4. Proposed SQRT-CSLA architecture for n- bit

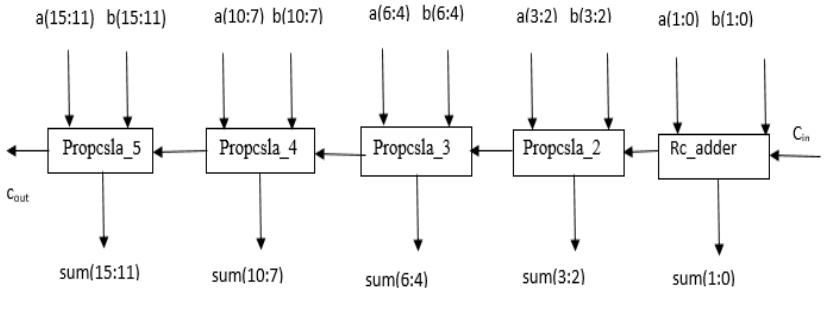

Fig. 5. Proposed SQRT CSLA block diagram for 16-bit

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_1even 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 gives 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 SQRT-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 SQRT-CSLA architecture.

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. 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 device utilization summary (area) is shown in Fig. 6. This gives the details about Look Up Table (LUT) and Input Output Blocks (IOB). The power report is shown in Fig. 7. The delay is shown in Fig. 8

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice LUTs	24	2400	1%
Number of fully used LUT-FF pairs	0	24	0%
Number of bonded IOBs	50	132	37%

Fig. 6. Device area summary of 16-bit proposed SQRT CSLA

```
2.3.  Power Supply Summary
---------------------------------------------------------
|                 Power Supply Summary                  |
---------------------------------------------------------
|                     | Total | Dynamic | Static Power |
---------------------------------------------------------
| Supply Power (mW)   | 13.66 | 0.00    | 13.66        |
---------------------------------------------------------
```

Fig. 7. Power report of proposed 16-bit SQRT-CSLA

```
Log
Speed Grade: -2

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 4.599ns

==========================================================================
```

Fig. 8. Delay report of proposed 16-bit SQRT-CSLA

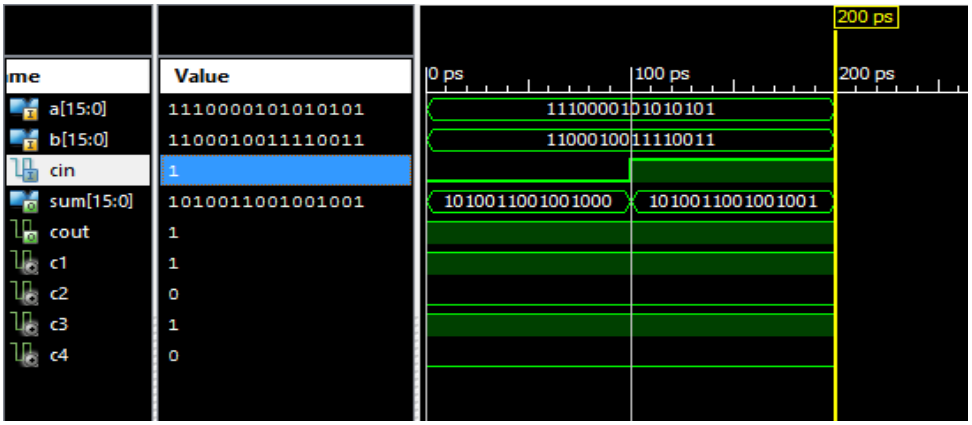

Fig. 9. Synthesis result of 16-bit proposed SQRT-CSLA adder

The proposed SQRT-CSLA is synthesized using ISim Simulator. Fig. 9. shows the synthesis result.

V. 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.

ACKNOWLEDGMENT

The heading of the Acknowledgment section and the References section must not be numbered.

Causal Productions wishes to acknowledge Michael Shell and other contributors for developing and maintaining the IEEE LaTeX style files which have been used in the preparation of this template. To see the list of contributors, please refer to the top of file IEEETran.cls in the IEEE LaTeX distribution.

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.Mariya Priyadarshini, N.V.N. Ravi Kiran, N. Tejasri, T.C. Venkat Anish. "Design of Area and Speed Efficient Square Root Carry 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 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.MECS, 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] 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.

[10] Padma Devi, Ashima Girdher, Balwinder Singh, "Improved Carry Select Adder with Reduced Area and Low Power Consumption", International Journal of Computer

