International Journal of Hybrid Information Technology
Vol.8, No.10 (2015), pp.197-204
http://dx.doi.org/10.14257/ijhit.2015.8.10.18

128 bit Unsigned Multiplier Design and Implementation Using an Efficient SQRT-CSLA

M Gopi[1] and GBSR Naidu[2]

[1]GMR Institute of Technology
mogalipuvvugopi@gmail.com
[2]Assistant Professor, GMR Institute of Technology
naidu.gbsr@gmrit.org

Abstract

In Digital systems like digital signal processors, FIR filters and micro processors etc, Multiplier is one of the key hardware blocks. The performance of the overall system is determined by the multiplier performance because the multiplier is generally the slowest element in the whole system and also it is occupying more area. In the multiplier, we use adder circuit repeatedly. So, an efficient adder circuit will be used in multipliers, it gives better performance. In the proposed work, new Carry Select Adders (CSLA) are replaced to enhance the multiplier performance. Carry Select Adder (CSLA) provides better performance with respect to speed and area. Previously, a binary to excess one converter (BEC) based Square Root Carry Select Adder is designed but in that data dependency is very high, it gives some speed penalty. An efficient CSLA design is obtained using improved logic units to eradicate the data dependency and redundant logic operations. In this proposed work, the intended efficient Square Root Carry Select Adder is compared with BEC based CSLA of respective architectures, after having comparison the proposed CSLA is efficient with respective to area and delay is used in Multiplier design. This work gives better results regarding to the performance parameters such as delay and area of designed multiplier using new efficient square root carry select adder compared to BEC based CSLA multiplier.

Keywords: *SQRT based CSLA, BEC, Optimized logic units, Multiplier design*

1. Introduction

In digital computer arithmetic, the basic operations are addition, subtraction, multiplication and division. In those, multiplication operation is very important, because it gives system's performance. Multiplication actual operation is repeated additions followed by shifting results, so we are going to deal with the process of additions implemented to the operation of multiplication. In VLSI designs, the major evaluates are chip area, power and delay for regulating the efficiency and performance of the VLSI architecture. Multiplications and additions are very important arithmetic operations used in all digital signal processing applications like FIR, FFT and IIR. Primary operation is addition for any digital multiplication. In digital system, area efficient, accurate operation and high speed are significantly influenced by the performance of the used adders.

In an arithmetic unit, the major component is adder. A complex digital signal processing (DSP) system needs several adders. An efficient adder design fundamentally improves the performance of a complex DSP system. A ripple carry adder (RCA) is simple design, but the major affect is carry propagation delay (CPD) in this adder. To reduce the CPD of adders, carry select (CS) and Carry look-ahead methods are developed.

A conventional CSLA is having two RCAs configuration which produces a pair of sums and output carries corresponding the expected input carry (c_{in}=0 and 1) and selects one out of each pair for final sum and final output carry. A conventional CSLA is requires less CPD than an RCA, but this adder design is not better since it uses more area because uses a dual RCA. Further implemented BEC based CSLA with one RCA and one binary to excess one convertor (i.e add-one circuit) instead of two RCAs. The BEC-based CSLA gives less logic resources when compare the conventional CSLA, but in this design has slightly higher delay. We examine that improved logic mainly depends on accessibility of redundant operations in the design formulation, whereas data dependence is major affect of adder delay. Based on this analysis, further implemented the new CSLA logic formulation. The main involvement in this design is logic formulation based on data dependence, improved carry generator (CG) and carry selection (CS) design. SQRT-CSLA using the new designed logic formulation gives less delay and less area than previous ones.

By using the new logic formulation design of SQRT-CSLA adder, 128 x128 bit unsigned multiplier is implemented. In this work, we are going to compare the performance of multipliers using with two different adders under concern of time needed and area for estimate. On comparison with the BEC-CSLA based multiplier, the efficient new logic formulation CSLA based multiplier is less area and delay time also reduced. Here, we are dealing with the two 128 bit input (n*n) and 256 bit (2n) resultant output. This efficient multiplier design involves 11% less delay than BEC based multiplier and also 6% area is reduced. In the next sections, explained different adder designs, multiplier design and performance comparisons of different multipliers.

2. Carry Select Adder

The ripple carry adder is generated by cascading each single bit full adder. In the ripple carry adder, each full adder starts its computation till preceding carry out signal is ready. Therefore, carry out propagation path in a ripple carry adder determines the critical path delay. Consider N bit full adder, carry propagation path of N-bit in the full-adders is the critical path. As the bit number N increases, ripple carry adder delay time will increase consequently in linearly. In order to improve the limitation of ripple carry adder to eliminate the linear data dependency between input word length and computation delay time, carry select adder is offered.

The CSLA has hardly two units: 1) the sum and carry generator unit (SCG) and 2) the sum and carry selection unit. More logic resources of CSLA is used by the SCG unit and considerably contributes to the critical path. For efficient SCG unit implementation, Different logic designs were recommended. We completed a revision of the logic designs recommended for the SCG unit of BEC-based CSLAs and conventional CSLAs of [2] by appropriate logic expressions.

The conventional Square Root Carry Select Adder is having dual Ripple Carry Adder followed by 2:1 multiplexer for the selection of sum and carry. Conventional SQRT CSLA major disadvantage is it requires the more area because Ripple Carry Adders are multiple pairs. The Conventional SQRT Carry Select Adder is shown in Figure 1. From the new logic design of conventional SQRT CSLA, delay and area are reduced. In conventional SQRT CSLA, both Sum and Carry bits are calculated for two alternatives that is carry input Cin= 0 and Cin= 1. Formerly Cin is come from preceding stage, mux chose the correct computation (*i.e.,* either Cin is ‘0’ output or Cin is ‘1’ output) for producing the correct output, instead of coming up for Cin to evaluate the sum, as fast as Cin gets there the output sum is properly taken.

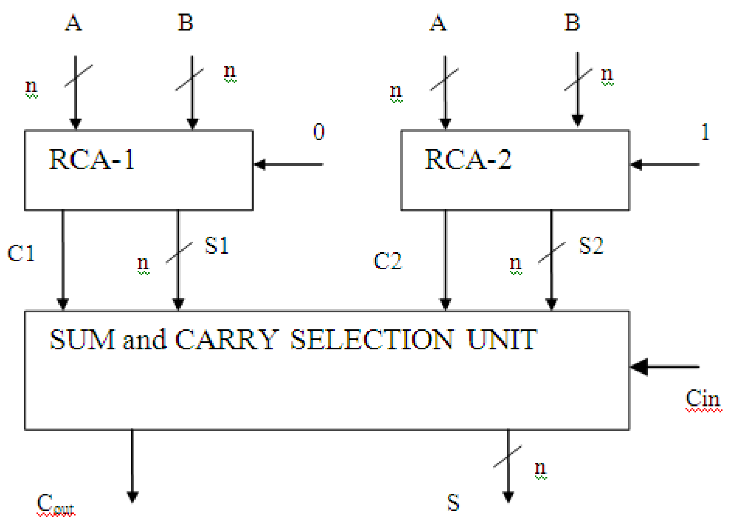


Figure 1. Conventional CSLA

The logic formulations of conventional CSLA as below, from structure of RCA which recommended designs in [1].

$$s_0^0(i) = A(i) \oplus B(i), c_0^0(i) = A(i).B(i) \qquad (1a)$$

$$s_1^0(i) = s_0^0(i) \oplus c_1^0(i-1) \qquad (1b)$$

$$c_1^0(i) = c_0^0(i) + s_0^0(i).c_1^0(i-1), c_{out}^0 = c_1^0(n-1) \qquad (1c)$$

$$s_0^1(i) = A(i) \oplus B(i), c_0^1(i) = A(i).B(i) \qquad (2a)$$

$$s_1^1(i) = s_0^1(i) \oplus c_1^1(i-1) \qquad (2b)$$

$$c_1^1(i) = c_0^1(i) + s_0^1(i).c_1^1(i-1), c_{out}^1 = c_1^1(n-1) \qquad (2c)$$

3. BEC-Based CSLA

By observing the Figure 1 and logic formulation 1(a)-1(c), 2(a)-2(c) and RCA-1 and RCA-2 operations. RCA-2 block performs just like add one bit to the RCA-1 result, Instead of RCA-2, add one circuit is used , *i.e.*, binary to excess one converter (BEC). The basic idea concerning this work is with BEC as a replacement for RCA with Cin =1 for reducing the Regular CSLA area. Common structure and the fundamental function of 4-bit BEC are shown in Figure 2 and the logic operation of BEC unit in [2, 1] as shown in Table 1.

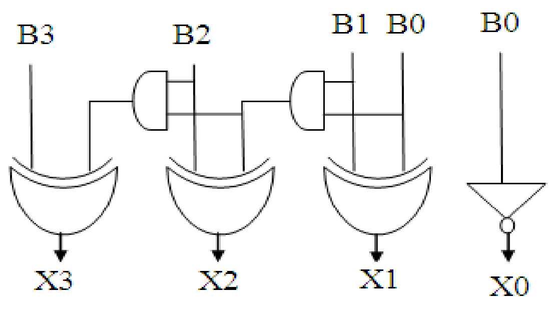


Figure 2. 4-bit Binary to Excess One Converter

Table 1. Function Table of 4 bit BEC

B[3:0]	X[3:0]
0000	0001
0001	0010
0010 ---- ----	0011 ---- ----
1110	1111
1111	0000

By using above BEC structure, n-bit design structure of BEC based CSLA is shown in below Figure.

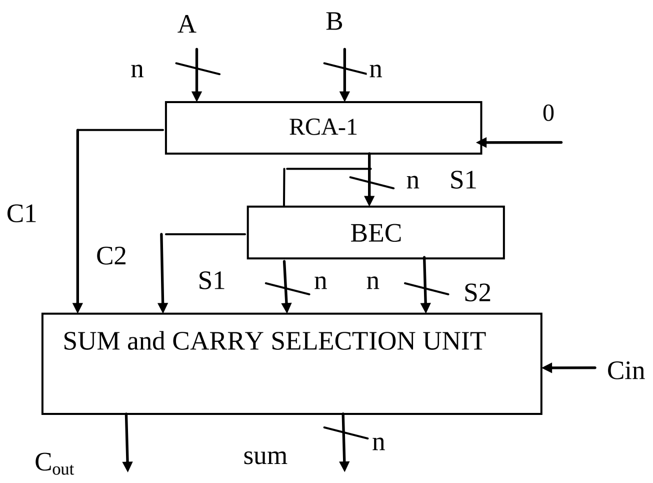


Figure 3. BEC Based CSLA

As shown in Figure 3, the RCA-1 calculates n-bit sum S1 and carry C1 corresponding to C_{in}=0.The BEC unit receives S1 and C1 from the RCA-1 and produces excess-1 code. From the operation of BEC based CSLA in [1], the logic expressions of the RCA are the same as those given in (1a)–(1c). In the n-bit BEC-based CSLA, the BEC unit logic expressions are given as

$$s_1^1(0) = \overline{s_1^0(0)}, c_1^1(0) = s_1^0(0) \qquad (3a)$$

$$s_1^1(i) = s_1^0(i) \oplus c_1^1(i-1) \qquad (3b)$$

$$c_1^1(i) = s_1^0(i).c_1^1(i-1) \qquad (3c)$$

$$c_{out}^1 = c_1^0(n-1) \oplus c_1^1(n-1) \qquad (3d)$$

By observing the fig3 and logic expressions, the BEC block must wait for the result of RCA-1 block result, so the data dependency of design is increased, delay also more. So, in the next section, the new logic formulation of CSLA is designed.

4. New Logic Formulation of SQRT-CSLA

In this new logic formulation of SQRT-CSLA, for decreasing data dependency based on the improved logic formulations which are given in (4a)–(4g), and its structure is shown in Figure 4. It incorporates of one HSG unit, one FSG unit, one CG unit, and one CS unit. The CG unit is composed of two CGs (CG_0 and CG_1) with input carry ‘0’ and ‘1’.

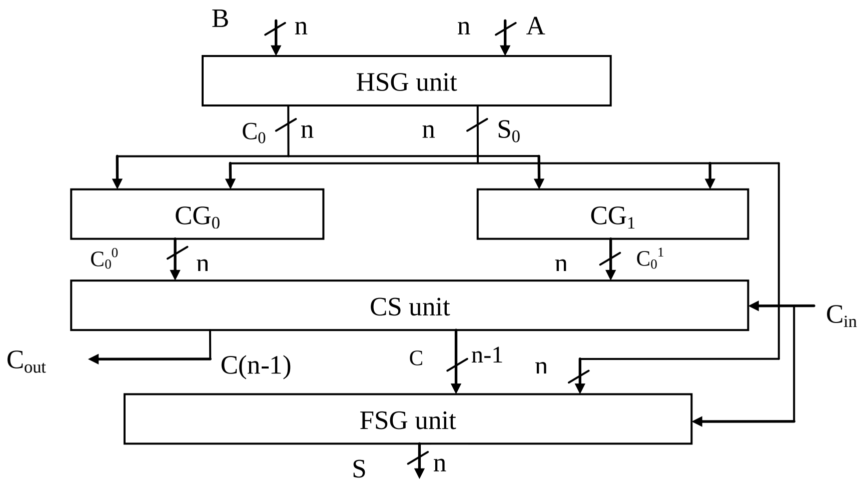


Figure 4. The New Logic Formulation CSLA

$$
\begin{aligned}
& s_0(i) = A(i) \oplus B(i), c_0(i) = A(i).B(i) && (4a) \\
& c_1^0(i) = c_1^0(i-1).s_0(i) + c_0(i) \quad for \quad (c_1^0(0) = 0) && (4b) \\
& c_1^1(i) = c_1^1(i-1).s_0(i) + c_0(i) \quad for \quad (c_1^1(0) = 1) && (4c) \\
& c(i) = c_1^0(i) \quad if \quad (c_{in} = 0) && (4d) \\
& c(i) = c_1^1(i) \quad if \quad (c_{in} = 1) && (4e) \\
& c_{out} = c(n-1) && (4f) \\
& s(0) = s_0(0) \oplus c_{in}, s(i) = s_0(i) \oplus c(i-1) && (4g)
\end{aligned}
$$

The HSG unit produces half-sum word S_0 and half-carry word C_0 of width n bits each from inputs of two n-bit operands A and B. Both CG_0 and CG_1 receive S_0 and C_0 from the HSG unit and they produce two n-bit full-carry words C_1^0 and C_1^1 corresponding to input-carry '0' and '1', respectively. The logic circuits of CG_0 and CG_1 are improved to get improvement of the fixed input-carry bits. The CS unit selects one final carry from the two carries at its input line using the control signal C_{in}. It selects C_1^0 when C_{in}= 0; otherwise, it selects C_1^1. The CS unit is implemented using an n-bit 2-to-l MUX. This aspect is used for improve the logic optimization of the CS unit. The improved designs of internal block are presented in [1]. As shown in figure 4, before full sum generation (FSG) carry selection is done, here logic optimization is improved.

In BEC based CSLA, generation of half sum and full sum when Cin =0 in RCA1 after that BEC unit is waiting for result of RCA1. In the new logic formulation half sum is independently generated and the carry is generated parallely when Cin=0 and 1. After that Carry selection is presented with selection line Cin and follows full sum is generated using this selection carry and half sum result. This optimization of logic design is reduced the area and also delay when compared with BEC based CLSA because decreases redundant logic operations and data dependency. The new logic formulation of CSLA is less delay and less area than BEC based CSLA.

5. MULTIPLIER DESIGN

Multiplication operation requires the partial products generation, each digit of the multiplier is having one partial product generation. After that partial products are summed to produce the final product. The multiplication of two n bit unsigned integers results in a product of 2n bits in length. The multiplication operation implementation for unsigned data is given in following algorithm.

Step 1: Examine the least significant bit of multiplier. If it is a 1, copy the multiplicand and call it first partial product. If the least significant bit is zero, then enter zero as first partial product and preserve the partial product.
Step 2: Examine the bit left of the bit examined last. If it is a 1, do step 3. Else do step 4.
Step 3: Add the multiplicand to the previously stored partial product after shifting the partial product one bit to the right. This sum becomes the new partial product. Go to step 5.
Step 4: Get new partial product by shifting the previous partial product one bit to the right.
Step 5: Repeat steps 2 to 4 till all bits in the multiplier have been considered. The final value obtained for the partial product is the product of the multiplicand and the multiplier.

In the above multiplication process, as shown in the Figure 5, controller block controls the multiplicand using LSB of multiplier every time and also controls shifting operations in entire block. In the 2n bit register, the product of each bit of multiplier X with multiplicand Y is placed and shift left each n bit binary value and the remaining bits are assigned with zeros. This 2n bit result is added with previous product Z and it continues until all bits in multiplier X.

In the adder block, I used different techniques like BEC based CSLA and the new logic formulation of CSLA. Because, the adder is the main block in multiplier. And compare different parameters like delay and area of multipliers, because speed is the main consideration in multipliers in digital design systems.

The comparisons of both multipliers using two parameters like delay and area (number of look up tables used). Performance comparisons of both multipliers are presented using design summary/ report of Xilinx 14.2 tool synthesis based on the family of Spartan3E.

This multiplication process is shown in Figure 5.

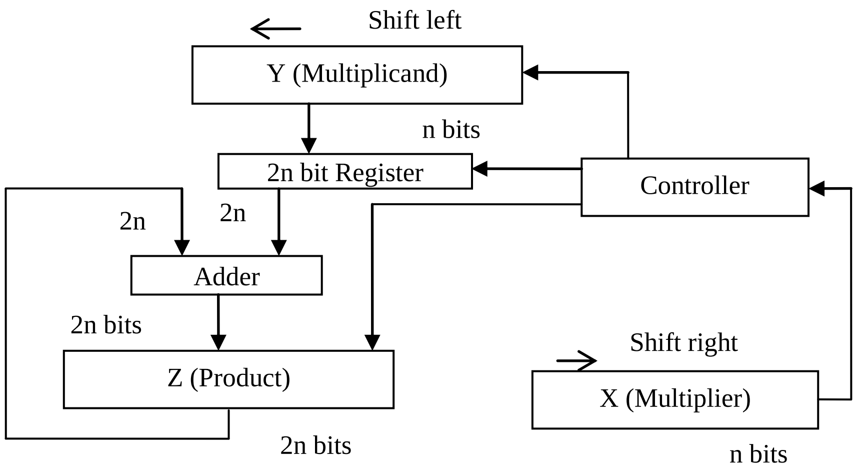


Figure 5. Multiplication Process Schematic

6. Performance Comparison of Multipliers using Synthesis Results

The proposed new logic formulation of CSLA based multiplier and BEC-CSLA based multiplier in this paper have been developed using Verilog HDL and synthesized in Xilinx ISE 14.2 for 128x128 unsigned multiplier. The similar design flow is followed for both the BEC and new logic CSLA based multipliers. The simulation results of both multipliers in terms of delay and area.

The performance analysis for the delay of both multipliers in nanoseconds and the performance analysis for the area of both multipliers in terms of number of Look up Table (LUT) required are represented in the form of table shown in Table 2.

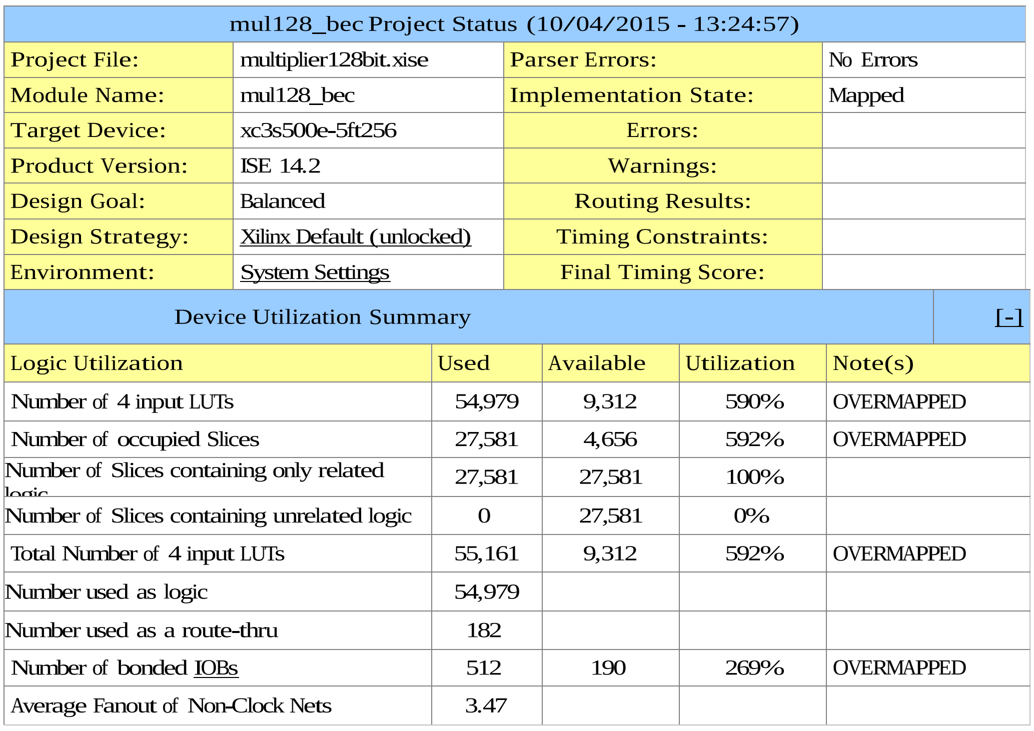
The Table 3 and Table 4 are the design summary/ report of BEC-CSLA based multiplier and new logic CSLA based multiplier, from area is represented using number of lookup tables used.

The delay of design of new logic formulation of CSLA based multiplier is 11% less than the BEC-CSLA based multiplier and the area of the design is 6% less than the BEC-CSLA based multiplier. The delays of multipliers are calculated using the execution time of design program in Xilinx ISE 14.2 tool.

Table 2. Delay and Area Comparisons of Both Multipliers

Multipliers	Delay(ns)	Area (LUT used)
BEC-CSLA based multiplier	105.840	55161
The new logic formulation CSLA based multiplier	93.438	51432

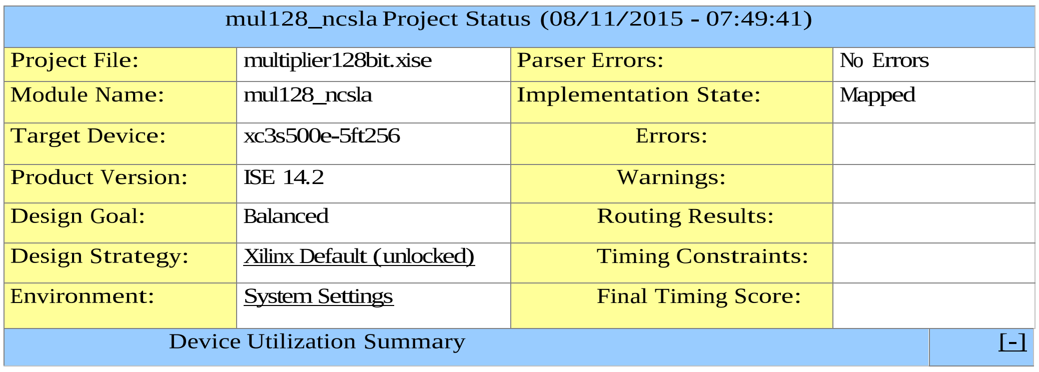
Table 3. Design Summary/Report of BEC-CSLA Based Multiplier



mul128_bec Project Status (10/04/2015 - 13:24:57)			
Project File:	multiplier128bit.xise	Parser Errors:	No Errors
Module Name:	mul128_bec	Implementation State:	Mapped
Target Device:	xc3s500e-5ft256	Errors:	
Product Version:	ISE 14.2	Warnings:	
Design Goal:	Balanced	Routing Results:	
Design Strategy:	Xilinx Default (unlocked)	Timing Constraints:	
Environment:	System Settings	Final Timing Score:	

Device Utilization Summary				[-]
Logic Utilization	Used	Available	Utilization	Note(s)
Number of 4 input LUTs	54,979	9,312	590%	OVERMAPPED
Number of occupied Slices	27,581	4,656	592%	OVERMAPPED
Number of Slices containing only related logic	27,581	27,581	100%	
Number of Slices containing unrelated logic	0	27,581	0%	
Total Number of 4 input LUTs	55,161	9,312	592%	OVERMAPPED
Number used as logic	54,979			
Number used as a route-thru	182			
Number of bonded IOBs	512	190	269%	OVERMAPPED
Average Fanout of Non-Clock Nets	3.47			

Table 4. Design Summary/Report of New Logic CSLA Based Multiplier



mul128_ncsla Project Status (08/11/2015 - 07:49:41)			
Project File:	multiplier128bit.xise	Parser Errors:	No Errors
Module Name:	mul128_ncsla	Implementation State:	Mapped
Target Device:	xc3s500e-5ft256	Errors:	
Product Version:	ISE 14.2	Warnings:	
Design Goal:	Balanced	Routing Results:	
Design Strategy:	Xilinx Default (unlocked)	Timing Constraints:	
Environment:	System Settings	Final Timing Score:	
Device Utilization Summary			[-]

Logic Utilization	Used	Available	Utilization	Note(s)
Number of 4 input LUTs	51,235	9,312	550%	OVERMAPPED
Number of occupied Slices	25,716	4,656	552%	OVERMAPPED
Number of Slices containing only related logic	25,716	25,716	100%	
Number of Slices containing unrelated logic	0	25,716	0%	
Total Number of 4 input LUTs	51,432	9,312	552%	OVERMAPPED
Number used as logic	51,235			
Number used as a route-thru	197			
Number of bonded IOBs	512	190	269%	OVERMAPPED
Average Fanout of Non-Clock Nets	3.57			

7. Conclusion

The design and implementation of 128-bit unsigned multiplier with BEC based CSLA and the new logic formulation CSLA using Verilog HDL was presented. The Xilinx ISE 14.2 tool is used for the synthesis and simulation of both multipliers. The simulation result shows that the new logic formulation of CSLA based multiplier design involves 11% less delay and occupies 6% less area than the BEC-CSLA based multiplier.

If the proposed multipliers are used in the digital systems, the performance of the digital system will be increased. Multipliers are mainly used in arithmetic and logic units (ALU) and DSP systems.

References

[1] B. K. Mohanty and S. K. Patel, “Area Delay–Power Efficient Carry-Select Adder”, IEEE Transactions on Circuits and Systems-II: express briefs, vol. 61, no. 6, **(2014)** June.
[2] B. Ramkumar and H. M. Kittur, “Low-power and area-efficient carry select adder”, IEEE Trans. Very Large Scale Integrated. (VLSI) Syst., vol. 20, no. 2, **(2012)** February, pp. 371–375.
[3] 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.
[4] P. Saxena, U. Purohit and P. Joshi, “Analysis of Low Power, Area- Efficient and High Speed Fast Adder”, in IJARCCE, vol. 2, Issue 9, **(2013)** September.
[5] V. Rajaraman and T. Radhakrishnan, “Digital Logic and Computer Organization”.
[6] V. K. Vadladi and D. S. Y. Raju, “A High Speed Design of 32 Bit Multiplier Using Modified CSLA”, in IJETEE, volume 10, issue 9, **(2014)** October.
[7] R. K. Maddheshiya, B. N. S. Munda and A. A. Singh, “Design of high speed and area efficient unsigned multiplier using CBL technique”, in IJVES, vol. 5, **(2014)** July.