International Journal of Advance Research in Science and Engineering
Vol. No.6, Issue No. 05, May 2017
www.ijarse.com
IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

A NOVEL DELAY EFFICIENT CARRY-SELECT ADDER USING RECURSIVE LOGIC

Priyanka Agrawal[1], Prof. Vijay Yadav[2], Prof. Rahul Shrivastava
[1]Mtech. Scholar, [2]Guide, [3]Co-Guide
Department of Electronics and Communication, LNCT Bhopal (India)

ABSTRACT
Delay is an important aspect of microprocessor design which is enhancing rapidly. Power consumption and power dissipation are important design constraints that are useful in increasing battery life and maintaining performance. The design of low delay and low-power VLSI architectures require efficient CPU units, which are optimized for the performance parameters, namely, speed and power consumption. Adders often appear in arithmetic logic unit, Intel execution unit and sometimes in the address generation path. Every digital circuit has some logic to perform which is basically an algorithm of frequent operations of addition, subtraction, division and multiplication to achieve a specific task. The repeated addition operation is nothing but addition. The speed of this repeated operation is highly depends on the structure of the adder operation highly depends on the architecture of the adder. This work shows the proposed optimum adder design base on Carry Select logic utilizing the adder architecture with better speed. The improved design is of 32-bit which has a delay of 3.426ns for performing multiplication operation which is 48% less than previous 32-bit adder design. The proposed recursive CSLA design is implemented on Kintex 7 FPGA device.

Keywords- binary adders, carry select adder, Carry look-ahead adder, Ripple carry adder, CMOS, VLSI.

I. INTRODUCTION
Speed is one of the major important design objectives in integrated circuit, after that power dissipation is also other. Multiplier-Accumulator (MAC) unit is main building block of DSP (Digital signal processing). MAC unit is responsible for low power and high speed. This is because speed and output are always concerns of DSP system. Due to dynamic growth of portable electronic devices like laptop, calculator, mobile etc., and the low power devices have become very commercial in today world. A VLSI designer plays the challenging role to achieve a high performance digital signal processing system for real-time signal processing. A common MAC unit consists of multipliers and accumulators that contain the sum of the previous consecutive products. The main agenda of this work is to investigate various multiplier and adder architectures which are suitable for implementing Low power, area efficient and high speed MAC unit.

An Arithmetic unit is made up of adder. A complex digital signal processing (DSP) system involves several adders. To improves the performance of a complex DSP system , an efficient adder design essential. A ripple carry adder (RCA) uses a regular design, but the drawback is carry propagation delay (CPD) in this adder. Various methods have been suggested such as Carry look-ahead and carry select (CS) to reduce the CPD of adders.

A designer can make tradeoffs between the two, through the choices in such factors as voltage levels, micro architecture, logic style or gate sizing and seek to find the balance that will comfort the designer's goals. Designers collect information about the consequences of their decisions of making above choices, and circuit designers use models of varying accuracy and speed to help them for see the consequences of their choices. An perfect understanding of the causes and severity of the power dissipation within a chip might influence a designer to make certain tradeoffs between speed and power, but a less accurate understanding might lead to tradeoffs that are worse for overall performance. Clearly how a microchip dissipates power is instrumental to good design. To lessen the energy dissipation, the circuit activity is looked important. "Arithmetic optimization using carry-save-adders", Carry-save-adder (CSA), well known from multiplier architectures, can be used for

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

the efficient CMOS implementation of a much wider variety of algorithms for high-speed digital signal processing than multiplication. Carry save adders has efficient concepts for implementation of high speed.

Furthermore, the requirement of more optimum adder designs for a modern microprocessors initiated in this research. For the applications such as the RISC processor design, where single cycle execution of instructions is the key measure of performance of the circuits, use of an efficient adder circuit becomes necessary, to realize efficient system performance. Additionally, the area is an essential factor which is to be taken into account in the design of fast adders. Towards this end, high-speed, low power and area efficient addition and multiplication have always been a fundamental requirement of high-performance processors and systems. The major speed limitation of adders arises from the huge carry propagation delay encountered in the conventional adder circuits, such as ripple carry adder and carry save adder.

II. CONVENTIONAL ADDITION

1. Serial Addition

It is common method which we familiar very well. This method is using the binary numbers results in fast algorithm used in mechanical addition. This algorithm was described using Boolean algebra by Shannon [8].

Figure 1 shows the operation is taking place between the two three bit operands. The operands are 3 (011) added to 2 (010) to yield 5 (101). The operands are in large bold number. The carry values are represented below the dotted line. The dotted line is used to mark that the carries are generated separately from the original operands, but that they are still added in as though they form a third operand. The solid line separates the sum from the carries. The addition start from the least significant bit to the most significant bit. The serial addition contains a parallel component since both sums and carries are generated simultaneously in each column.

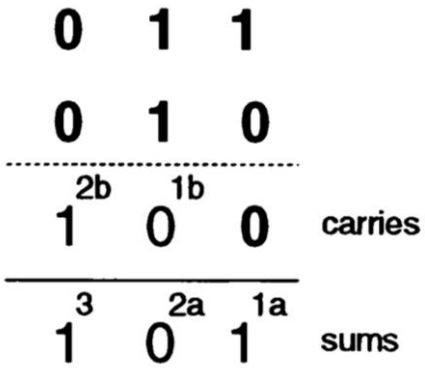

Figure 1 Serial Additions Algorithm.

Although, the algorithm is serial in the sense that the columns are processed sequentially starting from the right. The serial steps in figure 1 are numbered consecutively, while letters are used to distinguish among the steps that may occur in parallel or in any other order.

2. Ripple Carry Addition

The main difference between the ripple carry addition and serial addition is that ripple carry addition is asynchronous while serial addition is synchronous. In ripple carry addition all of the rectangular style carry chains start simultaneously, as shown for the example case in figure Signals in the ripple carry adder can transition multiple times1 due to the overlapping calculations. The wavering is denoted in the example with crossed out bits.

Figure 2 shows the prime events for an example add: the generation of two carries, the carries propagating across the adder, and then later one path running over the other in the upper four bits. More clearly, assuming that when adder starts with all zero carries, the first step in the algorithm comes with the sums in columns 0 and 3, and the carries for columns 1 and 4.

International Journal of Advance Research in Science and Engineering
Vol. No.6, Issue No. 05, May 2017
www.ijarse.com
IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

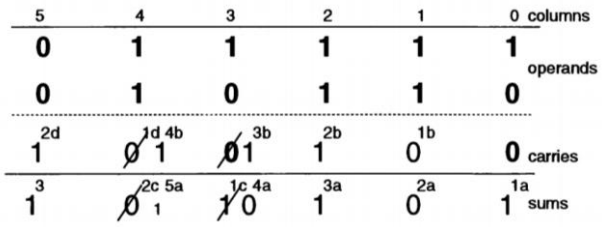

Figure 2 Ripple carry Addition Algorithm.

3. Carry Select Addition

This method is based on the divide and conquer phenomenon, which protects itself against making the wrong choices on the outcome of intermediate carries. To improve the speed of addition, the length of ripple propagate chains must be reduced. Carry would always be zero by the assumptions of any particular point, then the adder could be broken at that point into separate parallel processes, and the sum could be completed early. However, the assumption would be wrong 50% of the time can hedge bets on the outcome of such a carry calculation by starting two upper half adds, each with a different assumption about the value of the particular carry-out. Figure 3 shows an example of adding 01011111 to 01010110. The upper four bits of the add is replicated, the top version has the carry input bit set, while the bottom version does not have the carry bit set. Six items are calculated simultaneously in the first serial step. These contain the first column sum bits in each of the three additions, and the first column carry-out bits. Independently, these three additions continue on as in ripple carry addition till the carry proceed from the least significant four bits lead to the selection of the correct upper four bits, 1011 in the last step. This carry select addition requires only five serially dependent steps to add 8 bit operands in contrast to ripple carry addition of the same length operands would require 8 serially dependent steps.

The carry select algorithm is as follows:

(A).Conduct the three ripple carry adds simultaneously:

i. One adds with the lower half bits.
ii. Two adds with the upper half bits.
- One with the carry-in set to one.
- One with the carry-in set to zero.

(B). Take in the correct upper half.

However, the number of series steps for conducting this type of addition is approximately half the number for ripple carry addition, constant factors do not change the order of growth, and the performance is still linear:

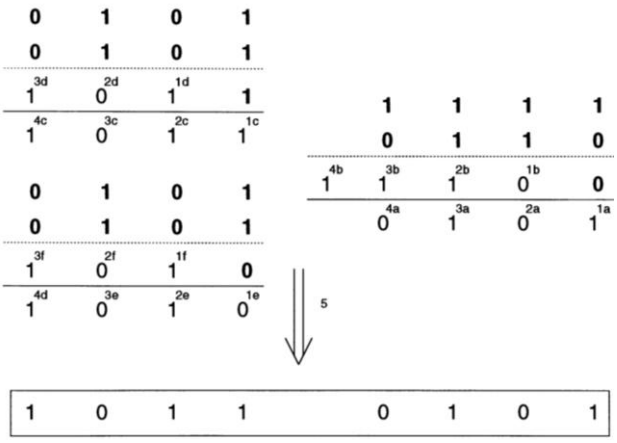

Figure 3 Carry Select Addition Algorithm.

4. Ripple Carry Select.

Carry select method divide the number into two segment but ripple carry addition is made by breaking into a large number of segment.

International Journal of Advance Research in Science and Engineering
Vol. No.6, Issue No. 05, May 2017
www.ijarse.com
IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

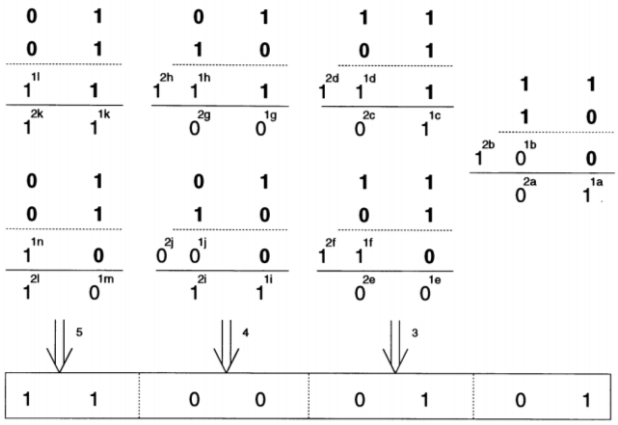

Figure 4 A ripple Carry Addition algorithm.

For instance, From the previous example, the 8 bit addition can be broken into four 2 bit segment, as shown in figure 4. Each of the 2 bit segment include duplicated 2 bit adders: one copy adds with an assumed carry-in of one, and the other with an assumed carry-in of zero. The right segment are selected in ripple fashion: the bottom 2 bit add selects among the right second section add, and the right second section add then provides the right carry for the third section, which provides the right carry for the fourth section.

III. PRIOR WORK

B. K. Mohanty and S. K. Patel,[1] In this brief, the logic operations involved in conventional carry select adder (CSLA) and binary to excess-1 converter (BEC)-based CSLA are analyzed to study the data dependence and to identify redundant logic operations. We have eliminated all the redundant logic operations present in the conventional CSLA and proposed a new logic formulation for CSLA. In the proposed scheme, the carry select (CS) operation is scheduled before the calculation of-final-sum, which is different from the conventional approach. Bit patterns of two anticipating carry words (corresponding to cin = 0 and 1) and fixed cin bits are used for logic optimization of CS and generation units. An efficient CSLA design is obtained using optimized logic units. 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 square-root (SQRT) CSLA. 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.

L. Mugilvannan and S. Ramasamy,[2] 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. This work uses a simple and efficient transistorlevel modification in BEC-1 converter to significantly reduce the area and power of the CSLA. Based on this modification 16-b square-root CSLA (SQRT CSLA) architecture have been developed and compared with the SQRT CSLA architecture using ordinary BEC-1 converter. The proposed design has reduced area and power as compared with the SQRT CSLA using ordinary BEC-1 converter with only a slight increase in the delay. This work evaluates the performance of the proposed designs in terms of delay, area, and power by hand with logical effort and through Cadence Virtuoso. The results analysis shows that the proposed CSLA structure is better than the SQRT CSLA with ordinary BEC-1 converter.

L. Suri, D. Lamba, K. Kritarth and G. Sharma,[3] Highly-increasing requirement for mobile and several electronic devices want the use of VLSI circuits which are highly power efficient. The most primitive arithmetic operation in processors is addition and the adder is the most highly used arithmetic component of the processor. Carry Select Adder (CSA) is one of the fastest adders and the structure of the CSA shows that there is a possibility for increasing its efficiency by reducing the power dissipation and area in the CSA. This research work presents power and delay analysis of various adders and proposed a 32-bit CSA that is implemented using Hybrid PTL/CMOS logic style. This work evaluates and analyses the performance of the proposed designs in

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

terms of area, delay, power, and their products in 90nm CMOS process technology. The results analysis is showing that the proposed CSA structure shows better result in terms of area, power and PDP (Power Delay Product) than the others.

A. Grover and N. Grover, [4] This article proposed an area-efficient carry select adder by sharing the common Boolean logic term. Representation of the circuit in summation operation need one XOR gate and one Inverter, while carry out can be represented using one AND gate and an inverter. Using the multiplexer, we are able to select the correct output results according to the carry input signal. In this way, the transistor count in a 32-bit carry select adder can be greatly reduced from 1947 to 960 and the carry select adder performs with a faster speed as compare to the carry ripple adder. Two different design styles using one multiplexer and two multiplexers has also been considered in terms of number of transistors, average power consumption, propagation delay at sum and at carry output.

A. Ramakrishna Reddy and M. Parvathi,[5] Most of the VLSI applications, such as DSP, image and video processing, and microprocessors use carry select adder (CSLA) for arithmetic functions. From the structure of regular SQRT CSLA, still there is possibility to obtain better design in which optimization of area, power are to be major concentrations along with high speed performance. One of the existing solutions used in SQRT CSLA is replacement of second level RCA by BEC. Though increases the performance, very less percentage of improvement in reduction of area and power dissipation. And also the existing adder with BEC technique is not suitable for low power applications. Hence this work proposes Special Hardware using Multiplexers (SHM) design in place of second level RCA. It is observed from the results that the area and power dissipation are reduced at comparable percentages with respect to the RCA and BEC techniques. When SHM is used at the second level of second block in 16-bit SQRT CSLA, observed that area is reduced by 13.5% and power dissipation is reduced by 6.4%. This proposed logic is designed in transistor level using 0.12µm technology in the Micro wind tool.

M. A. Akbar and J. A. Lee, [6] The common design problem in various approaches for self-checking adders is the fault propagation due to carry. Such a fault can misguide the system to detect the particular faulty module. In this work, we proposed a self-checking Carry Select Adder (CSA) with fault localization ability. Our scheme can provide minimum area overhead for self-recovery process because instead of replacing the whole system we can now replace the particular faulty modules. The proposed self-checking CSA consumes 12% less area with equal performance as compared to the previously proposed self-checking CSA approach.

S. Parmar and K. P. Singh,[7] The work describes the power and area efficient carry select adder (CSA). Firstly, CSA is one of the fastest adders used in many data-processing systems to perform fast arithmetic operations. Secondly, CSA is intermediate between small areas but longer delay Ripple Carry Adder (RCA) and a larger area with shorter delay carry look-ahead adder. Third, there is still scope to reduce area in CSA by introduction of some add-one scheme. In Modified Carry Select Adder (MCSA) design, single RCA and BEC are used instead of dual RCAs to reduce area and power consumption with small speed penalty. The reason for area reduction is that, the number of logic gates used to design a BEC is less than the number of logic gates used for a RCA design. Thus, importance of BEC logic comes from the large silicon area reduction when designing MCSA for large number of bits. MCSA architectures are designed for 8-bit, 16-bit, 32-bit and 64-bit respectively. The design has been synthesized at 90nm process technology targeting using Xilinx Spartan-3 device. Comparison results of modified CSA with conventional CSA show better results and improvements.

IV. PROBLEM STATEMENT

In a microprocessor or a digital signal processor (DSP), data path plays a prominent role since performance metrics like the die-area, speed of operation, power dissipation etc., depend directly on the efficiency of data-path. As is known, core of the data path involves complex computations like addition, subtraction, multiplication and division, etc. Thus, realizing efficient hardware units for these computations, which directly affect the performance of data path, is of prime importance. The most executed operation in the data path is addition, which requires a binary adder that adds two given numbers. Adders also play a vital role in more complex computations like multiplication, division and decimal operations. Hence, an efficient implementation of binary adder is crucial to an efficient data path. Relatively significant work has been done in proposing and realizing efficient adder circuits for binary addition.

International Journal of Advance Research in Science and Engineering
Vol. No.6, Issue No. 05, May 2017
www.ijarse.com
IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

V. PROPOSED METHODOLOGY

RTL schematic of proposed CSLA design has demonstrated in figure and schematic of Schematic of 4-Bit Adding Operation is demonstrated in figure

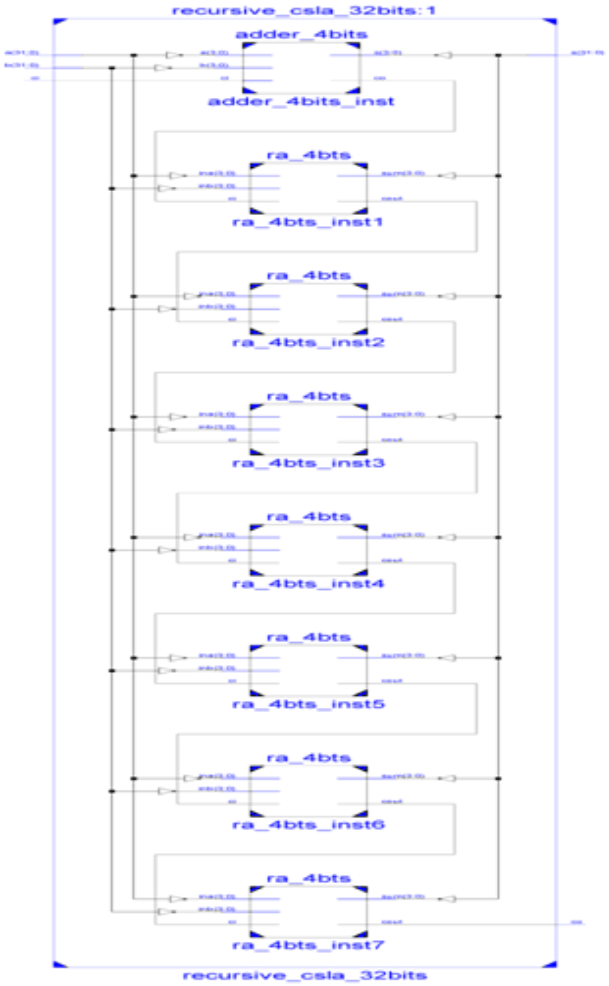

Figure 5. RTL Schematic of Proposed Recursive CSLA

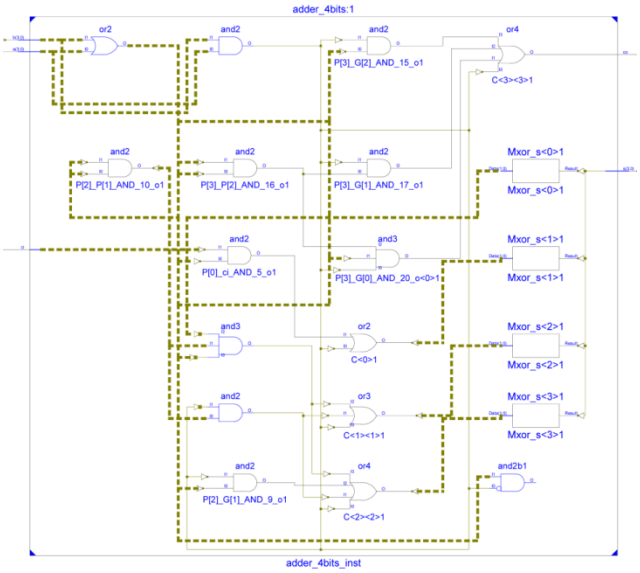

Figure 6. RTL Schematic of 4-Bit Adding Operation

International Journal of Advance Research in Science and Engineering
Vol. No.6, Issue No. 05, May 2017
www.ijarse.com
IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

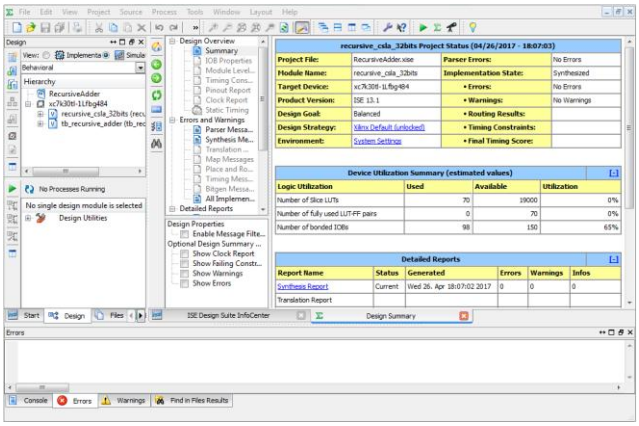

Figure 7. Project Windows of the Proposed Recursive Carry Select Adder (CSLA) Architecture Design

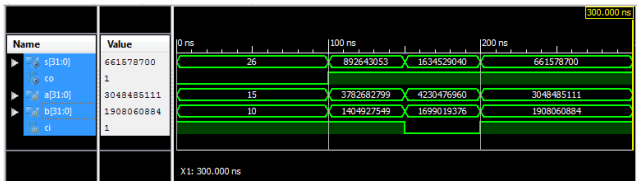

Figure 8. Testbench Waveforms of Proposed Recursive Carry Select Adder (CSLA)

```
Timing Details:
---------------
All values displayed in nanoseconds (ns)
=========================================================================
==========
Timing constraint: Default path analysis
  Total number of paths / destination ports: 396 / 33
-------------------------------------------------------------------------
----------
Delay:               3.426ns (Levels of Logic = 7)
  Source:            a<25> (PAD)
  Destination:       s<31> (PAD)

  Data Path: a<25> to s<31>
                                   Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net
Name)
    ----------------------------------------  ------------
     IBUF:I->O             4   0.003   0.710  a_25_IBUF
(a_25_IBUF)
     LUT4:I0->O            1   0.069   0.419
ra_4bts_inst6/co0_SW0 (N24)
     LUT5:I4->O            4   0.069   0.439  ra_4bts_inst6/co0
(ra_4bts_inst6/co0)
     LUT3:I2->O            1   0.069   0.690
ra_4bts_inst7/nst1/Mmux_s41_SW0 (N26)
     LUT5:I1->O            1   0.069   0.419
ra_4bts_inst7/t1/Mmux_s41)
     LUT3:I2->O            1   0.069   0.399
ra_4bts_inst7/mux_s42 (s_31_OBUF)
     OBUF:I->O                 0.002          s_31_OBUF (s<31>)
    ----------------------------------------
    Total                      3.426ns (0.350ns logic, 3.076ns
route)
                                       (10.2% logic, 89.8%
route)
```

Figure 9: Timing Summary of the Proposed Recursive Carry Select Adder (CSLA).

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

Table 1: Comparison of Delay and Area Utilization

Technique	Delay	Area
Previous Work (Base Paper) SQRT-CSLA	6.59 ns	3735.66 μm^2
Proposed Work (Our) Recursive - CSLA	3.426 ns	70 lice LUTs

VI. CONCLUSION

The proposed design is implemented using Verilog for 32-bit recursive adder. Verilog was used to model and synthesis our architecture. Using recursive logic improves the overall performance of the multiplier. Thus a 48% less delay with the use of the recursive CSLA. The future extension could the use of same adder architecture to implement higher bit sized architectures for example 64-bit or 128-Bit which will significantly improved in terms of area as well as delay.

REFERENCES

[1] B. K. Mohanty and S. K. Patel, "Area–Delay–Power Efficient Carry-Select Adder," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 61, no. 6, pp. 418-422, June 2014.
[2] L. Mugilvannan and S. Ramasamy, "Low-power and area-efficient carry select adder using modified BEC-1 converter," 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), Tiruchengode, 2013, pp. 1-5.
[3] L. Suri, D. Lamba, K. Kritarth and G. Sharma, "High performance and power efficient 32-bit carry select adder using hybrid PTL/CMOS logic style," 2013 International Mutli-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), Kottayam, 2013, pp. 765-768.
[4] A. Grover and N. Grover, "Comparative Analysis: Area-Efficient Carry Select Adders 180 Nm Technology," 2013 7th Asia Modelling Symposium, Hong Kong, 2013, pp. 99-102.
[5] A. Ramakrishna Reddy and M. Parvathi, "Efficient carry select adder using 0.12µm technology for low power applications," 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Mysore, 2013, pp. 550-553.
[6] M. A. Akbar and J. A. Lee, "Self-Checking Carry Select Adder with Fault Localization," 2013 Euromicro Conference on Digital System Design, Los Alamitos, CA, 2013, pp. 863-869.
[7] S. Parmar and K. P. Singh, "Design of high speed hybrid carry select adder," 2013 3rd IEEE International Advance Computing Conference (IACC), Ghaziabad, 2013, pp. 1656-1663.
[8] N. Sloane and A. D. Wyner, eds., Claude Elwood Shannon Collected Works. IEEE Press, 1993. [1] 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.
[9] 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.
[10] 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.
[11] 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.
[12] B. Parhami, Computer Arithmetic: Algorithms and Hardware Designs, 2nd ed. New York, NY, USA: Oxford Univ. Press, 2010.