A Novel Delay Efficient Carry Select Adder Using Recursive Logic

Priyanka Agrawal[1], Prof. Vijay Yadav[2], Dr. Rita Jain[3]

[1]Mtech. Scholar, [2]Guide, [3]HOD

Department of Electronics and Communication, LNCT Bhopal

Abstract - 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 - Recursive addition, 32-bit, CSLA Delay.

I. INTRODUCTION

Managing power consumption is an aspect of microprocessor design that is important and growing increasingly more so. In mobile electronics reducing power consumption is a key factor in increasing battery life, and even in servers and desktop computers power dissipation is an important design constraint.

In any form of engineering there are inevitably tradeoffs to be made, and one of the primary tradeoffs facing designers is that of trying to balance performance against power dissipation. Through choices in such factors as voltage levels, micro architecture, logic style, or gate sizing a designer can make tradeoffs between the two, and seek to find the balance that will best fit the designer's goals.

To make these choices designers require information about the consequences of their decisions, and circuit designers use models of varying accuracy and speed to help them forsee the consequences of their choices. An accurate 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 an accurate representation of how a microchip dissipates power is instrumental to good design.

Circuit activity is generally one of the most important factors in energy dissipation. In this thesis I will be looking at adders specifically, and experimentally determine activity factors. From there, I examine how these vary from standard models and how this variance effects design choices.

The simplest form of adder is the ripple-carry adder. An n-bit ripple carry adder consists of n one-bit full adders connected in succession. The carry "ripples" from the least significant bit to the most significant bit and hence the name. Since, the carry in at any stage depends on the carry out from the previous stage, the delay of ripple-carry adders is O (N) and increases linearly as the size of the operands is increased. This becomes inadequate as the size of the operands increases to 64 bits and 128 bits. Thus it is essential to look for other alternates.

Carry select adders [1] consist of two ripple carry adders and a multiplexer. Addition of two n-bit numbers is done with two adders: one time with the assumption that the carry in is zero, and the other assuming that the carry is one. After the two results are calculated, the correct sum and the correct carry is selected with the multiplexer once the correct carry of previous stage is known. The number of bits in each carry select block can be uniform, or variable. In the uniform case, the optimal delay occurs for a block size of VN, where N is the size of the operands. The carry select adders are simple, but rather fast compared to the ripple carry adders having a delay of O (VN).

The higher delay in the case of ripple carry adders is due to the carry chain. In carry look-ahead adders, the carry signals are calculated in advance, based on the input signals [2], [3]. For any bit position i, a carry will be generated if the corresponding input bits are '1' or if the carry-in to that bit was a '1' and at least one of the input bits are '1'. From this, a recurrence relation is derived that expresses carry in to any bit position in terms of the relevant addend and augend digits and some lower-adder carry. This can result in considerable gain in speed.

II. CARRY SELECT ADDER (CSA) DESIGN

Ripple carry multi-bit adder takes more time to finish the computation because of carry propagation through all one bit adder blocks. General carry select adder architecture

consists of two sets of ripple carry adders and a multiplexer. It has two architectures, conventional/linear carry select (LCS) and square root carry select architecture (SQRT CS). LCSA design for multi bit addition use 'p' number of 'm' bit carry select blocks to get n (= p*m) bit addition. SQRT CSA design use 'm' bit carry select block as first block of carry select. In later stages of carry select, the number of bits of carry select addition in each block increases by one bit until it reaches '$m + p - 1$' bits for pth block. The main advantage of using carry select adder is the sum and carry out of any sub block are predicted. In carry select adder, from second sub block to final sub block, the sum outputs and carry outs are selected. So, the design doesn't add any carry propagation delay other than initial sub block set up time (internal ripple carry) and the delay in later stages caused by fan out, selection at multiplexers, for preceding block carry out.

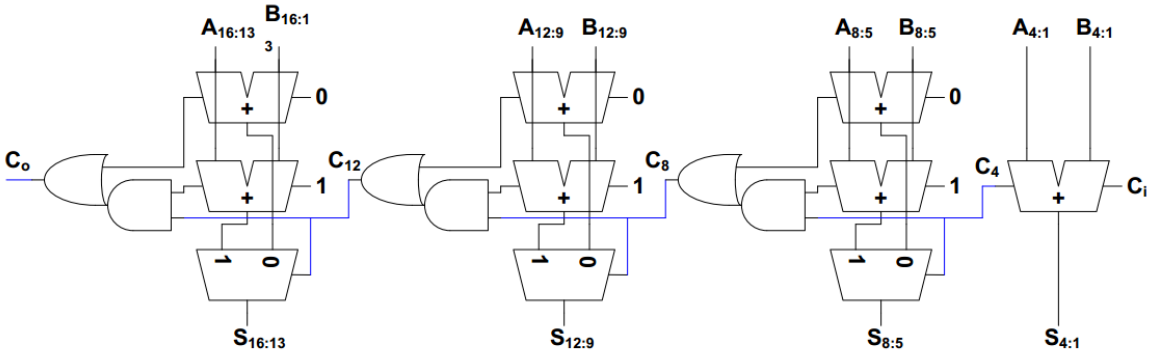

Figure 1.1 16-bit Carry Select Adder

III. PROPOSED ARCHITECTURE DESIGN

In the CSLA architecture the redundant and the repeating logic blocks are replaced with the optimized gate level blocks. These blocks reduce a large logic from the conventional CSLA design. The logic block diagram of the CSLA that is implemented in this work is shown in Fig 3.1 RTL Schematic of Proposed Recursive CSLA and in Figure 3.2 RTL Schematic of 4-Bit Adding Operation. This design has following internal blocks: (1) half sum generator (HSG), (2) carry generator with „0" input (CG0), (3) carry generator with „1" input (CG1), (4) carry select unit (CSU), and (5) full sum generator (FSG). The HSG block performs the half-adder based arithmetic addition of the corresponding bits of the two inputs. It involves logic AND gate and logic XOR gate for its realization. Computational time in the CSLA design is demonstrated in table 1. The CSU unit selects the carry as per the carry input of the CSLA. The MSB output of CSU block is the carry output of the CSLA operation. The FSG block generates the sum bits of the CSLA operation by adding the sum output from the HSG block and the carry outputs of the CSU block. The gate level logic diagram of the sub-blocks of CSLA Design is shown in Fig 3.2. This diagram describes the interconnection of logic gates to implement „4"-bit adder. In this architecture, the carry is generated for two fixed initial carry values, i.e., logic-„0" and logic-„1" carry value. The output of these blocks is used to select the final carry on the basis of the carry Cin input. The carry bits generated from CSU are used in the FSG to generate the sum output.

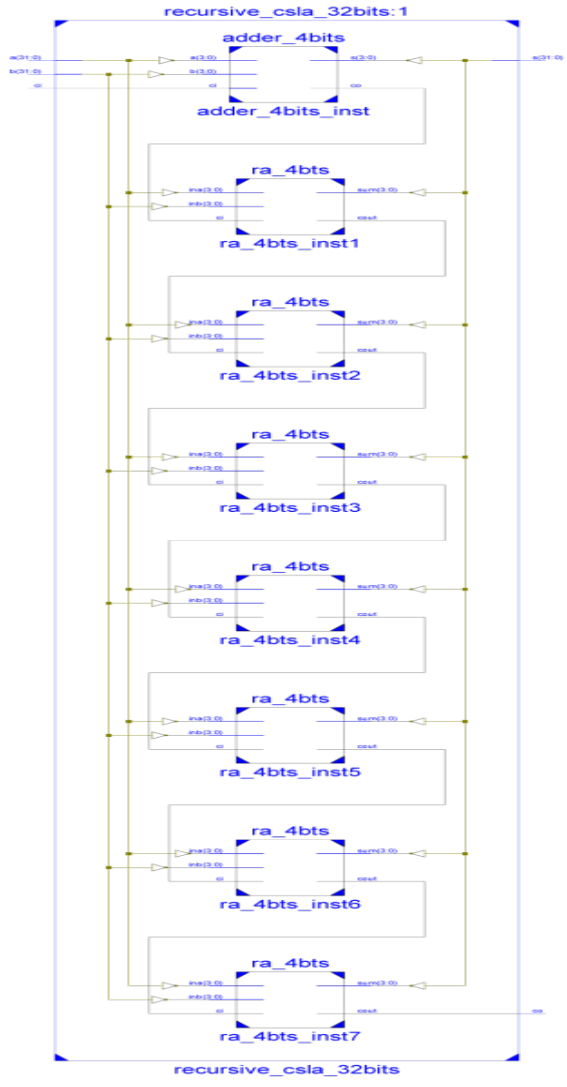

Fig. 3.1 RTL Schematic of Proposed Recursive CSLA

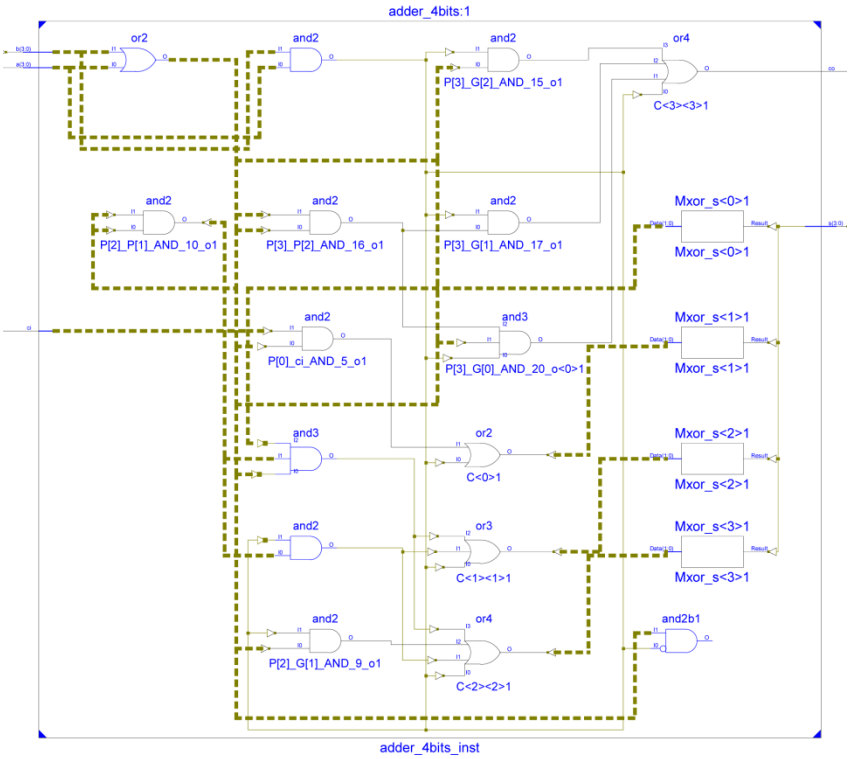

Fig. 3.2. RTL Schematic of 4-Bit Adding Operation

IV. SYNTHESIS OUTCOMES

The design is simulated for functional verification and dynamic power consumption. Xilinx ISE Tool is used for performing functional and power simulation. Figure 4.3 shows the resource utilization summary of the carry select adder design for FPGA.

Fig 4.1 shows Project Windows of the Proposed Recursive Carry Select Adder (CSLA) Architecture Design. The proposed implementation is a combinational realization of CSLA. Fig 4.2 shows Simulation waveform of the proposed CSLA Design. The simulation of the proposed design in performed on different input values to ensure the authentication of the operational performance of the various blocks of the proposed CSLA design.

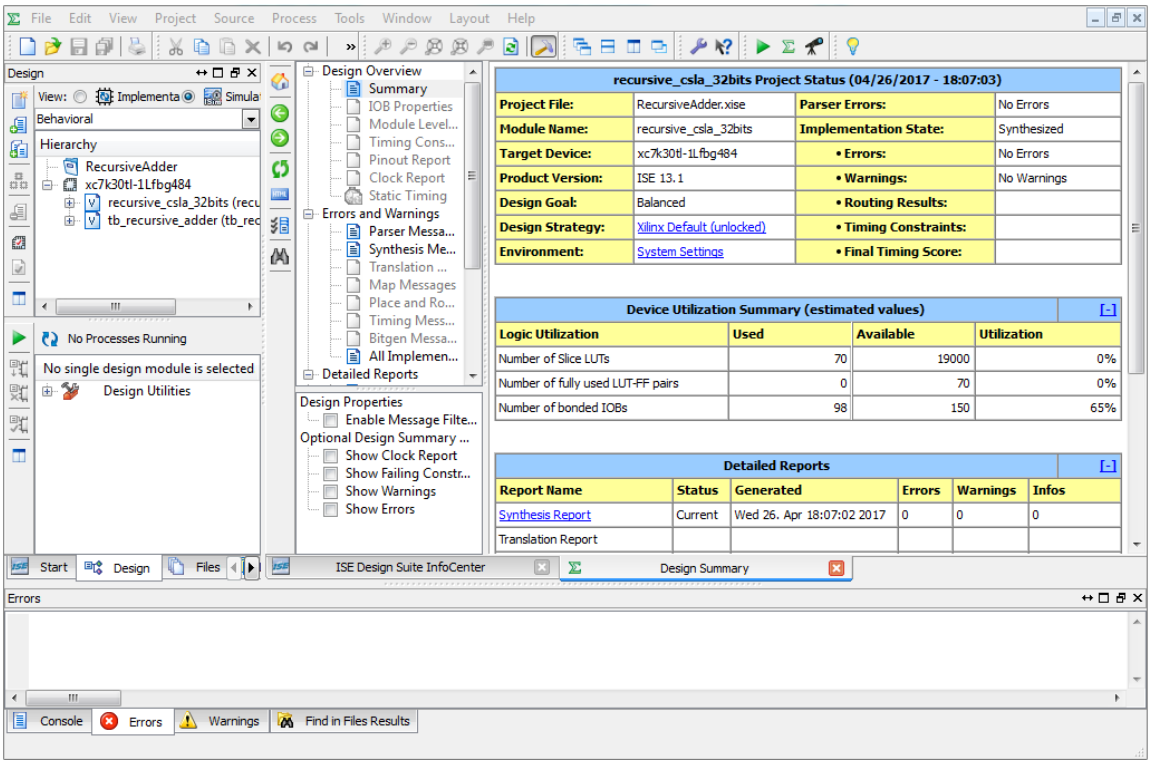

Figure 4.1 Project Windows of the Proposed Recursive Carry Select Adder (CSLA) Architecture Design.

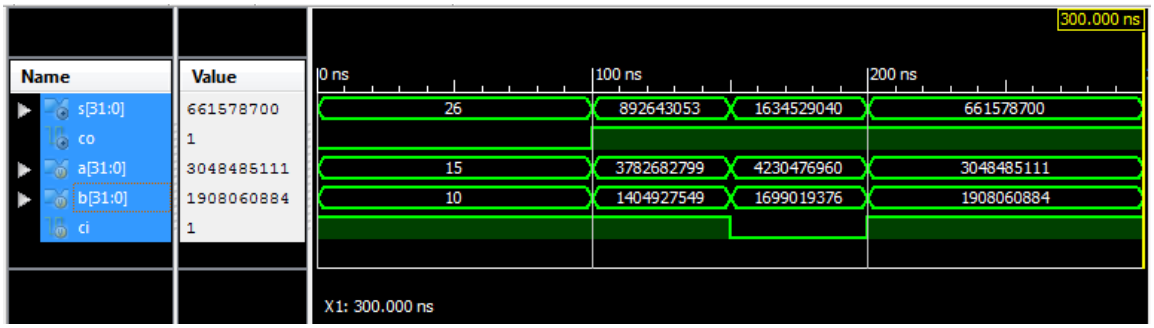

Figure 4.2 Testbench Waveforms of Proposed Recursive Carry Select Adder (CSLA).

```
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 4.3 Timing Summary of the Proposed Recursive Carry Select Adder (CSLA).

Table 1: Comparison of Delay and Area Utilization

Technique	*Delay*	*Area*
Previous Work (Base Paper) SQRT-CSLA	T6.59 ns	3735.66 um^2
Proposed Work (Our) Recursive - CSLA	3.426 ns	70 Slice LUTs

V. 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.