Area-Delay-Power analysis of Carry Look-Ahead Adder Architecture

Ch.Rajendra Prasad¹,Y.Srikanth², P.Ramchandar Rao³,S.Sanjay Kumar⁴
¹,²,⁴Department of Electronics & Communication Engineering, SR University, Warangal, Telangana, India- 506371
³Center for Embedded & IoT, Department of Electronics & Communication Engineering, S R Engineering College, Warangal, Telangana, India- 506371
¹rajendra_prasad_ch@srecwarangal.ac.in, ²srikanth.y@sru.edu.in, ³ramchandar_rao_p@srecwarangal.ac.in,⁴sanjaykumar.s@sru.edu.in

Abstract. The addition is common in hardware for the microprocessor and digital signal processor (DSP), and an adder is used to execute the addition. The Adder should feature high speed and low power for real-time applications. An effective adder architecture principally advances the performance of microprocessors and DSP systems. The carry propagation delay (CPD) is the main apprehension in the design of adder architecture. To address CPD, a new Carry Look-Ahead architecture is proposed, in which the carry propagation is scheduled before the calculation of the final sum using carry look–ahead (CLA) method. A quantitative estimate shows that the Area Delay Product (ADP) of proposed adder architecture is minimized by 10% as compared with the existing adders’ architectures.

Keywords: Arithmetic Unit, Low-power, Carry Propagation Delay, Adders, Area Delay Product.

1. Introduction
The basic functionality of an arithmetic unit is addition. Which is commonly used operation and has been ranked among a collection of real-time signal processing to general-purpose processors [1, 2]. The CSLA architecture provides less CPD in contrast to RCA, but this architecture employs dual RCA, hence it is not an impressive design. Some of the modifications made in the CSLA architecture to sidestep the dual use of RCA. In [3] the authors replaced one RCA with a one-add-one circuit and the multiplexer is used to design one-add-one circuit in the architecture. In [4] the authors presented a square-root (SQR)-CSLA to design less delay architecture with large bit-widths. In this architecture, a cascading structure is employed to provide large bit widths and the overall delay is minimized by the propagation of carry through a parallel path. In [5], the authors proposed a Binary to Excess-1 Converters (BEC) - based CSLA, which includes fewer logic gates in contrast to traditional CSLA, but it is suffer with higher CPD. The Common Boolean Logic (CBL) - based CSLA of [6] comprises less logic gates than the traditional CSLA, but the CPD is approximately equivalent to the RCA.

On an extensive study on carry propagation and generation within the Carry select adder. We observed that CSLA designs use more number of logic gates required to carry selection and carry propagation units. To address, CPD the following are the possible solutions for adder architectures.

1. Perceive the end of propagation
2. Speed-up propagation via
   - Carry select
   - Look-ahead
3. Restrict CP for a fewer bits
4. Remove CP via the redundant bits

For the minimization of CPD, we are suggesting a Carry Look-Ahead (CLA) scheme for the design of the adder. In this paper, an area-delay-power enriched carry look-ahead adder is proposed to reduce CPD.

The remainder of the paper presented as follows; section 2 presents a review on adder architectures. Section 3 explains the proposed CLA scheme. Section 4 employs a comparison of CSLA adder with
the proposed scheme. Finally, conclusions are provided in Section 5.

2. Review on adder architectures

This section introduces the different adder architectures. In this paper, the following adder architectures are taken for comparison of the proposed adder design.

- Ripple Carry Adder (RCA)
- Carry Select Adder (CSLA)
- Carry Save Adder (CSaA)

2.1 RCA: The RCA architecture consists of a series of full-adder (FA) is illustrated in Figure 1. Every FA is accountable for ripple carry at any stage because it gives the sum of 2 binary bits at that stage. Out-carry of one FA is directly given to input carry for the consecutive FA. This adder architecture is simple and it can add large bit size numbers, but which is not perform well when the input bit size is large. The major disadvantage is that the CPD of RCA architecture rises proportionally with the bit size. In RCA, the delay under worst-case is defined as a carry shift ripples via every stage of adder sequence from the LSB to the MSB, which is expressed approximately by:

$$t = t_s + t_c (n - 1)$$

where $t_s$ is the total computational delay and $t_c$ is the FA carry stage delay. In RCA this delay is linearly relative to the input bits $n$, hence the speed of the RCA is restricted as the number of input bits is increased. Excluding delay, the RAC architecture simpler and consume less power and smaller chip area due to the compact design.

2.2 CSaA: In CSaA, carry-save element composed of n FAs, each FA gives a one-bit sum and carry on the equivalent bits of the 3 inputs. The CSaA minimizes the addition of 3 numbers to the 2 numbers. The transition delay is only through 3 gates irrespective of the n. The entire sum is computed by shifting 1-bit position by the carry sequence to left and append a zero to the MSB of the sum sequence and summing up the result with RCA generates corresponding $(n + 1)$-bit number. This operation repeated indefinitely and summing input for every stage of FAs, by neglecting any transitional carry propagation. Each stage might be organized in a tree configuration, with collective delay is logarithmic number as inputs bits are added, and it is in-variant on input bit number. The key implementation of the carry-save architecture is the multiplier structure and is employed for effective CMOS application of much extensive diversity of designs for high-speed DSP systems in real-time applications. In CSA, the CP speed up by an array of multipliers is applied as the partial product for the array. The architecture of CSaA is illustrated in Figure 2.
2.3 CSLA: A CSLA is split into sections, each section will perform addition in parallel, except for the LSB. One assumes input carry as ‘0’ and the second one assumes input carry as ‘1’ is illustrated in Figure 3. A 4-bit CSLA involves two RCAs and a multiplexer. This architecture is very fast and it is very simple and has an area of \( \sqrt{n} \) at the gate-level depth. For \( 2^n \)-bit numbers, the CSLA has two adders for performing the computation twofold, the first time it assumes carry as 0 and for the next time it assumes as 1. Once the precise carry is known, the precise sum, and also the precise carry, is then identified using multiplexer after the double computation.
3. Problem statement
The FA circuit indicating carry generation $G_i$ and CP $P_i$ is shown in Figure 4.

![Figure 4. FA indicating carry generation Gi and CP Pi](image)

3.1. Proposed Adder Design
The CLA adder has been configured to defeat the latency caused by the carry bits rippling effect. In parallel adders, the CP delay occurred can be reduced by CLA architecture. This architecture is built on the concept of looking at the augends and lower order bits of the addend while producing a higher-order carry. In addition, it decreases the carry delay by which the number of gates a carrying signal has to transit through. CLA based on two parameters: computing for each digit position, if the position will transit a carry when one generates from the right and combining those determined signals to have been able to effectively deduce if a group will propagate a carry coming into it from the correct for each group of digits. The overall result is that the carry begins by gradually transit via each 4-bit unit, and like in a ripple-carry scheme, but it transfers four times faster, jumping from one look-ahead to take the unit to another. Eventually, the carry propagates gradually within the digits in that group within each group that receives a carry, as illustrated in Figure 5.

The CLA adder is composed of three blocks: 1. Generate or propagate block, 2. Sum generator and 3. carry generator. The expression (2) is employed for the realization of generating block

$$G_i = (x_i) \times (y_i)$$

(2)

The expression (3) is employed for the realization of the propagating block.
\[ P_i = (x_i) \oplus (y_i) \]  \hspace{1cm} (3)

The \((i-1)\)\textsuperscript{th} stage Carry-out is attained from expression (4)

\[ C_i(C_o) = P_iC_i + G_i \]  \hspace{1cm} (4)

The output sum is obtained using the following expression

\[ S_i = (x_i) \oplus (y_i \times C_i) \]  \hspace{1cm} (5)

The gate-level diagram of CLA generator for 4-bit CLA adder is illustrated in Figure 6.

**Figure 5.** 4-bit CLA adder

**Figure 6.** Gate level diagram of CLA generator
An 8-bit CLA adder employing two 4-bit CLA units is illustrated in Figure 7. The output-carry of the first 4-bit CLA is fed to the input-carry of the second 4-bit CLA.

4. Comparison of adders

4.1 Area–Delay Estimation Method: For estimation of power dissipation, delay, and area, basic gates are considered such as inverter, 2-input OR, and 2-input AND (IOA). A 2-input XOR is consists of two NOT gates, single OR, and two AND gates. Table 1 shows delay and area required for NOT gate, 2-input OR gate, and 2-input AND gate. These values are taken from standard datasheet cell library for qualitative validation of adder architectures. The delay and area of a proposed adder design are estimated by employing the following expressions:

\[ T = n_i.T_i + n_o.T_o \]  \hspace{1cm} (6)

\[ A = i.N_i + a.N_a + r.N_o \]  \hspace{1cm} (7)

where \((N_i, N_o, N_a)\) and \((n_i, n_o, n_a)\), respectively, denote the \(\text{NOT, OR, AND}\) gate counts of the proposed adder architecture and its critical path. \((T_i, T_o, T_a)\), respectively, denote the delay of a single gate \(\text{NOT, OR, AND}\). In the proposed design, \(\text{IOA}\) gate counts considered estimation delay and area of the adder. Using (6) and (7), the delay and area of the proposed architecture are computed from the \(\text{IOA}\) gate counts \((n_i, n_o, n_a)\), \((N_i, N_o, N_a)\), and the cell details of Table 1.

| Adder | AND-Gate | OR-Gate | NOT-Gate |
|-------|----------|---------|----------|
| Area(\(\mu\text{m}^2\)) | 7.37 | 7.37 | 6.45 |
| Delay(ps) | 180 | 170 | 100 |

4.2 Results and Discussion: We analyzed and compared the performance of our proposed CLA with recently proposed CSLA competitive ones. To obtain the gate count and analyze the worst-case delay, and average power consumption of the proposed architecture, we reproduce the contending circuits as
stated in the literature. In the process the schematic is found to be given[1] requires 32 gates for CP. So we are reduced the number of gates from 32 to 28 in Carry look-ahead adder with considerable delay, area, and power dissipation. This work has been developed with Verilog HDL. It is simulated and synthesized using Xilinx 14.1. Table 2 illustrates a comparison of area, delay, power consumption, and gate count for CP of different 8-bit adders. Figure 8 illustrates the ADP of the proposed adder with traditional adders. The proposed algorithm gives less ADP as compared with the CSLA[1], CSLA[6], CSLA[7], and CSLA[8].

| Adder    | Power (µW) | Area (µm²) | Delay (nS) | Gate Count(CP) | ADP (µW.µm²) |
|----------|------------|------------|------------|----------------|--------------|
| CSLA[6]  | --         | 1282.45    | 4.08       | --             | 5.23         |
| CSLA[7]  | 12.897     | 906.56     | 3.78       | --             | 3.42         |
| CSLA[8]  | 6.433      | 1654.65    | 7.30       | --             | 12.08        |
| CSLA[1]  | 9.8326     | 951.09     | 3.42       | 32             | 3.68         |
| Proposed | 8.6035     | 832.0      | 2.99       | 28             | 3.22         |

Figure 8. ADP of proposed adder with traditional adders

5. Conclusion
An area delay power-efficient CLA adder is presented in this paper. By using the CLA generator we can reduce logic operations. In this way, the gate count for carry look-ahead adder can be significantly minimized from 32 to 28. However, the proposed CLA design involves less area, delay, and power dissipation than the recently proposed CSLA [1]. Due to the small output carry delay, this adder is best suitable for the arithmetic unit. The synthesis results reveal that the proposed adder provides approximately 10% ADP than existing adder designs of 8-bit widths.
References

[1] Mohanty B K, and Patel S K 2014, Area–delay–power-efficient carry-select adder *IEEE transactions on circuits and systems II: express briefs*, **61**(6), 418-422

[2] PravalikaK, and Prasad C R 2018, Fault-Tolerant Parallel Filters Based on ECC Codes *Advances in Computational Sciences and Technology* **11**(7), 597-605

[3] Parhi K 2007 VLSI digital signal processing systems: design and implementation *John Wiley & Sons.*

[4] Chandrakasan A, et.al. 2008, Ultralow-power electronics for biomedical applications. *Annual review of biomedical engineering* **10**(1), 247-274

[5] Kavitha M, Prasad C R, and Ahmed S M Vedic multiplier for RC6 Encryption Standards Using FPGA.

[6] Bedrij O J 1962, Carry-select adder. *IRE Transactions on Electronic Computers* (3), 340-346

[7] B. Parhami, 2010, Computer Arithmetic: Algorithms and Hardware Designs, 2nd ed. New York NY USA Oxford Univ. Press

[8] Kareemoddin M, Kumar A A and Ahmed S M 2013, Design of low power comparator for SAR ADC in biomedical applications. *IJIRCCE* **1**(5), 1161-1167

[9] Masood S, Shamshi M A H, and Ahmed S M 2017, An Efficient, Cost-Effective, All-Optical Implementation of Reversible Fast Adder Using Mach-Zehnder Interferometer Based Switches. *In 2017 IEEEICRTEECT* 23-28

[10] Reddy S K, Ahmed S M, Improving Perfom Mance & Power in Multi-Core Processors.

[11] Chakradhar A, MD Hameed Pasha and G Shravan Kumar 2016, Low Power Area Efficient Counter Using Pulse Triggered Flip-Flop *International Journal of Innovations in engineering and Technology NCRTEFOSS 2016* 85-89