International Journal of Wireless and Ad Hoc Communication

♦ ISSN: 2692-4048 Vol: 09, No. 02, 2025 ♦ PP. 33–37

Article DOI: https://doi.org/10.54216/IJWAC.090206

FULL LENGTH ARTICLE

FPGA-Based Arithmetic Operator Implementation for FIR Filter Design Using FLUT Architecture

A. Arun[1,*] **M. Thangavel**[2] **V. Kunavathi**[3]

[1] Assistant Professor, Department of ECE, Knowledge Institute of Technology, Salem, India
[2] Professor, Department of ECE, Knowledge Institute of Technology, Salem, India
[3] PG Scholar, Department of ECE, Knowledge Institute of Technology, Salem, India

Emails: aaece@kiot.ac.in · mtece@kiot.ac.in · kunavathi.venmani@gmail.com

Received: February 04, 2025 Revised: February 28, 2025 Accepted: March 30, 2025 ⋆ Corresponding author

ABSTRACT

A hardened adder with carry logic is commonly used in commercial field-programming gate arrays (FPGAs) to enhance arithmetic performance. The rapid expansion of portable multimedia players and communication systems has boosted the demand for high-speed, energy-efficient Digital Signal Processing (DSP) systems. The Finite Impulse Response (FIR) filter is an essential component when developing an effective digital signal processing system. Digital multipliers and adders are the crucial arithmetic units used in FIR filters, determining the entire system's performance. As a result, the low-power design of systems has become a primary performance target. This paper explores the influence of fLUTs and their interactions with hardened arithmetic. FLUTs (Fracturable-LUT) reduce the area by 15%, complementing the latency reductions offered by hardened arithmetic. An FIR filter based on the Carry-Look-ahead Adder (CLA) and multipliers is proposed. The tentative results show that the FIR filter using the proposed multiplier method achieves less delay and power reduction compared to conventional methods.

Keywords: Carry Look-ahead Adder (CLA) ▪ Field Programmable Gate Array (FPGA) ▪ Fracturable Look-up Table (FLUT)

1. INTRODUCTION

A field programmable logic block connected by a grid may be changed in the field to interact with other logical blocks to perform different digital operations [1]. An ASIC design for a 64th-order adjustable FIR filter architecture with low power and area, based on the enhanced multiplier, has been reported. The Basys-3 FPGA board uses the programmable Vedic FIR filtering design, defined in VHDL, for quick prototyping. The delay in propagation is estimated based on the longest path between the input and output. The power consumption evaluation is performed using a clock frequency of 100 kHz [2]. The Carry Look-ahead Adder (CLA) is evaluated based on its arithmetic performance. The purpose of the CLA adder is to reduce the amount of space occupied by the circuit with increased delay while also reducing the delay with increased area consumption. The proposed adder structure achieves optimal performance, which means that the delay is minimized while consuming the same amount of area as in a conventional adder design. Typically, the size of the look-ahead logic is limited to three carries. AND gates with five to six inputs are required for the following two carry signals, making their implementation in CMOS extremely slow due to the stacked transistors in the pull-up or pull-down routes. The carry look-ahead block provides the carry calculation; hence, the one-bit adder equations for a CLA are simplified because carry calculations are no longer needed [3]. Digital filters are obtained in the Digital Signal Processor (DSP), although DSP-based solutions cannot match the high-speed demands

of applications due to their sequential structure [4]. Integer linear programming (ILP) can be used for minimizing the number of adders necessary to create a direct or transposed FIR filter complying with a certain frequency parameter [5]. The adder is constructed using a two-level netlist to interleave corrections throughout the design. The Boolean equations associated with every netlist are generated and implemented in CMOS technology at 45 nm. The proposed BCD adders perform with better speed and power than other designs [6].

Fixed-point arithmetic operations suffer from significant data losses, as do single-precision floating-point operations. Most double-precision floating-point arithmetic operations use dual rail coding to execute complete detections and require the system to receive acknowledgment upon complete execution, resulting in a worst-case delay regardless of the actual completion time [7]. The approximate recoding adder reduces energy consumption, area, and the critical path. An FIR adaptive filter for reducing partial products (PP) and accumulating circuits is designed using distributed arithmetic, signed 32-bit and sixteen-bit radix-8 Booth algorithms, and approximate computing under an insignificant adder. High-productivity, space-effective AES encryption can be designed and implemented using an FPGA for safety reasons [8].

2. RELATED WORK

FIR filters have become common in many fields of digital signal processing due to their ability to provide linear phase as well as system stability. A 70-tap low-frequency FIR filter is employed. The sampling frequency was set to 40 MHz, the bandwidth was 2 MHz, and the precisions of the input and filter coefficients were 13 and 12, respectively. To achieve optimal filtering performance and resource efficiency, 4-input LUT units with additional multiplexing devices and complete adders are used. The symmetrical design of the 70-tap FIR filter allows for a reduction to 35 taps; then the 35-tap filter is separated into seven small filters, each with five DA-LUT units. A 4-input LUT combined with a 2×1 multiplexer and a full adder can be used to build the 5-tap DA-LUT [9]. Reconfigurable FIR filters are implemented and manufactured utilizing 40 nm CMOS semiconductor technology. The suggested design outperforms existing approaches in area-delay products and power performance, making it appropriate for low-power filtering applications [10].

The subsequent processing unit renders the spintronic adders immune to input scheduling. The proposed adder requires at least 29% less space. It also provides a 40% shorter carry propagation latency and 37% lower power-delay products than efficient ones [11]. Based on cache conflict data, an integer linear programming approach was proposed to produce the best constant function for system performance. Furthermore, to enhance the scalability of the suggested allocation technique for an enormous task set, providing an interference ratio allows the interference impact to be determined quantitatively [12]. The FPGA architecture used in this paper is heterogeneous, with soft logic blocks, simple I/Os, adjustable memory, fracturable multipliers, and a 50% depopulated crossbar connecting block inputs and basic logic element (BLE) outputs to BLE inputs. Hardening adders and carry chains dramatically increases the precision of arithmetic operations. Delay reduction was increased by 69–79%, with an average circuit delay improvement of 13–16%. While more complicated architectures that harden a CLA enhance standalone adder speed, simple hardened ripple-carry adders perform equally well on entire application circuits. The greater adaptability of fLUT-based architectures allows them to be area-efficient by up to 15% [13].

3. PROPOSED METHODOLOGY

3.1 Baseline Architecture

Fundamental FPGA architecture is built in 22-nm CMOS processing and is heterogeneous, with soft logical blocks, simple I/Os, customizable memory, and fracturable multipliers [13]. The carry look-ahead adder is implemented to demonstrate the adder structures, as shown in Figure 1. All 1-bit addition algorithms calculate their output. The look-ahead units perform their calculations at the same time. Table 1 demonstrates that if a carry occurs in a specific group, it will appear at the left-hand end of that group within at most five gate delays and begin propagation throughout the group to its left [3].

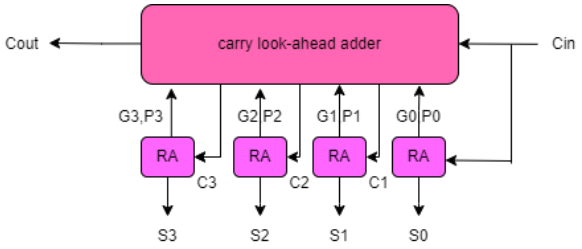

Figure 1. Schematic diagram of carry look-ahead adder.

The look-ahead carry logic grows more sophisticated as the number of bits in a group increases. The following equations compute each position's generate and propagate signals:

$$G_i = A_i \cdot B_i, \tag{1}$$

$$P_i = A_i + B_i. \tag{2}$$

The equations for the carries in a CLA are given by:

$$C_1 = G_0 + P_0 C_{in}, \tag{3}$$

$$C_2 = G_1 + G_0 P_1 + P_0 P_1 C_{in}, \tag{4}$$

$$C_3 = G_2 + G_1 P_2 + G_0 P_1 P_2 + P_0 P_1 P_2 C_{in}. \tag{5}$$

In Equations (3)–(5), the carries are computed further and further in advance, so larger gates are required. For example, computing C_3 requires the use of a 4-input AND gate and a 4-input OR gate. Hence, the size of the look-ahead logic is usually limited to three carries [3]. The CLA employs intermediate information to predict whether a particular bit position will be carried out. Table 1 depicts the truth table for a full adder, including the extra carry information.

3.2 Digital Filter Signal Flow Graph

A digital filter modifies signal characteristics in the time and/or frequency domains. A digital signal is created by sampling a continuous signal in multiple periods and then representing the signal as a sequence of discrete values, as opposed to an analog signal, which is continuous and represented as a function of time. The signal flow graph is shown in Figure 2.

The difference equation is a mathematical description of digi-

Table 1. Generate and propagate information for a CLA.

A	B	C	Sum	Carry Out	Condition
0	0	0	0	0	Delete
0	0	1	1	0	Delete
0	1	0	1	0	Propagate
0	1	1	0	1	Propagate
1	0	0	1	0	Propagate
1	0	1	0	1	Propagate
1	1	0	0	1	Generate/propagate
1	1	1	1	1	Generate/propagate

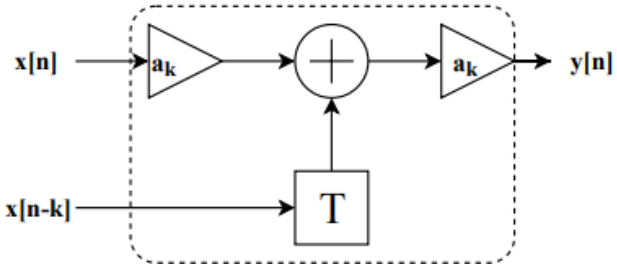

Figure 2. Signal flow graph structure.

tal FIR filter techniques. A linear time-invariant (LTI) filter is one of the most often used digital filters. The impulse response of an LTI filter is denoted as $h[n]$. The impulse response $h[n]$ is the output achieved by using an impulse as an input sequence [14]. The convolution method convolves the impulse response and input sequence in the time domain to produce the output $y[n]$, which may be mathematically expressed as:

$$y[n] = x[n] * h[n]. \tag{6}$$

3.3 Structure of 4-bit Carry Look-ahead Adder with fLUT

Presenting a method for carrying adaptive structure generation, technology mapping, partitioning, and positioning for any SRAM-based FPGA can be characterized using generic models. During each design process, knowledge of the logical structure is efficiently applied. As previously stated, the duration of a BCLA unit is not limited to a certain value. As a result, equations can be developed for block-propagate, generate, and carries.

The CLA employs additional circuitry to generate carry bits in parallel, eliminating the time required to compute the bigger value bit result. Figure 3 depicts the architecture of a four-bit CLA. CLA can be divided into two parts: the Partial Full Adder (PFA) and the carry look-ahead logic. The PFA generates propagate signals p_i, generate signals g_i, sum output s_i, and carry-out bits C_{i+1} via the look-ahead logic circuit. In a partial full-adder, propagate P and generate G rely solely on the input bits, while the carry generator is not affected by previous carry-ins. After computing C_0, C_4 can reach a stable state without waiting for C_3 to propagate.

3.4 FIR Filter Using CLA Adder Design

The FIR filter has three primary components: a D-FF to create a basic delay [15], a multiplier used to carry out the filter coefficients, and an adder that sums all nodes at the completion of each tap [16]. Functional verification of all adders and multipliers is shown in Figure 4.

In this case, $X(n)$ represents the input filter coefficients; C_0,

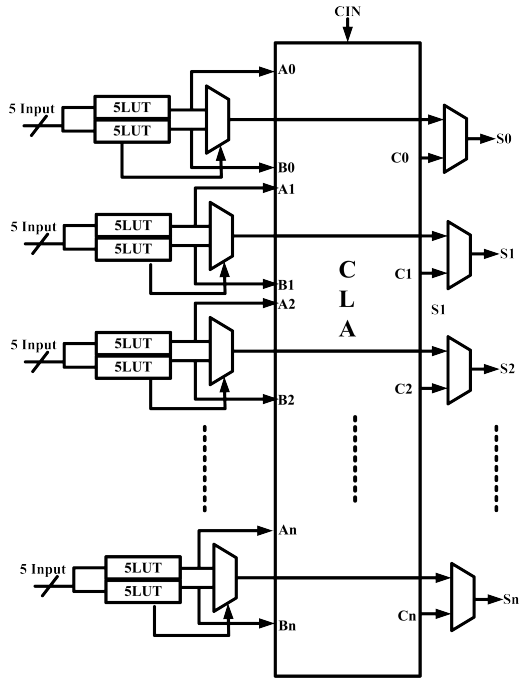

Figure 3. Carry look-ahead adder with LUT.

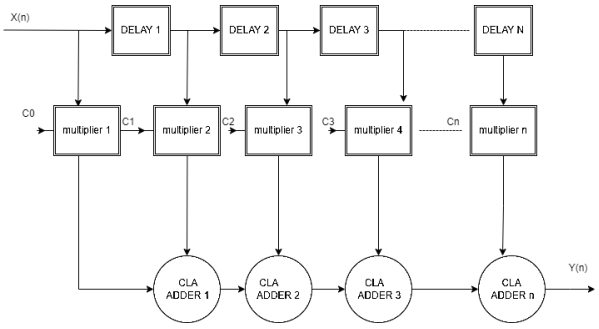

Figure 4. Direct method FIR filter with CLA.

C_1, C_2, and C_n are the transfer function coefficients; and $Y(n)$ is the output filter coefficient. Multipliers can be replaced by a shift-and-add operation, known as Multiple Constant Multiplication (MCM), in which a set of constants, here h_0 and h_1, is multiplied by a variable, here $x(n)$.

4. RESULTS AND DISCUSSION

The comparison concentrates on the various hard adder versions using full application evaluations. These allow assessment of the general quality of the various implementation techniques, considering total delay and area. Architectures that reduce the area-delay product are the most efficient.

The benchmarks are comprehensive application circuits that can perform a wide range of arithmetic operations, such as multiplication, addition, and subtraction, using CLA adders with an FIR filter. Vivado 2019.1 is used for design, synthesis, and implementation.

4.1 Schematic Diagram of 4-bit CLA Adder

Figure 5 shows the representation of the 4-bit carry look-ahead adder design.

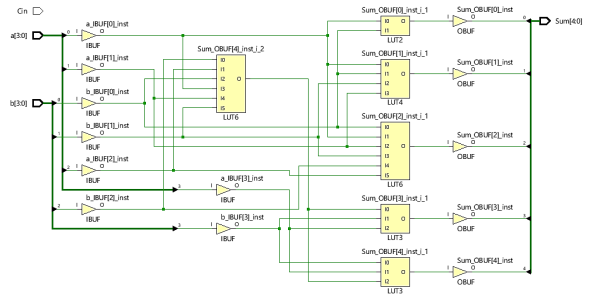

Figure 5. Schematic diagram of CLA 4-bit.

4.2 Output Wave for FIR Filter Design

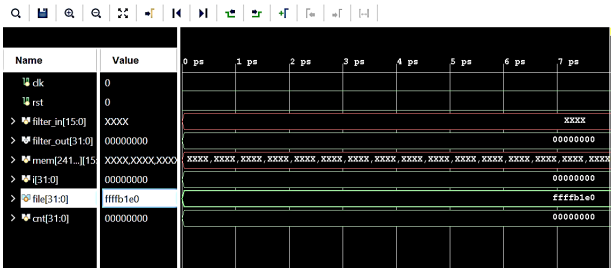

Figure 6. FIR filter behavioral simulation.

Figure 6 shows the behavioral design of the proposed FIR filter design with inputs and output.

4.3 FIR Filter Implemented Design

The FIR filter implemented design simulated output is shown in Figure 7.

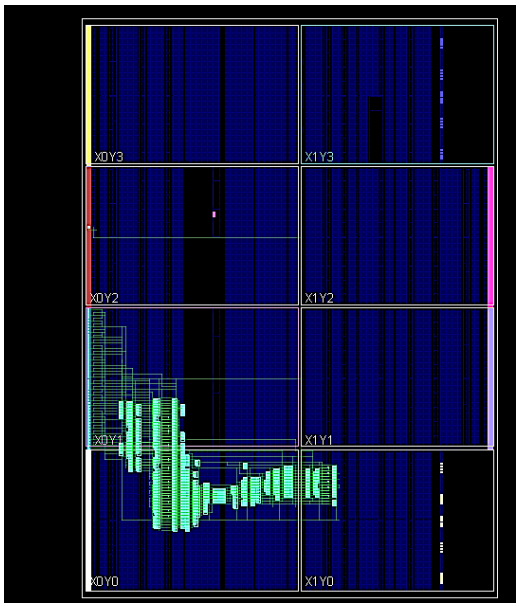

Figure 7. FIR filter implemented design.

4.4 FIR Filter LUT Design Runs

The synthesis and implementation completion report, with usage of LUT, Flip-flop, and DSP, is shown in Figure 8.

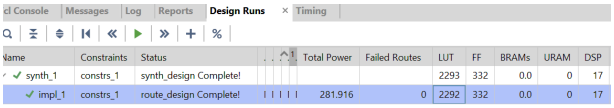

Figure 8. FIR filter synthesis and implementation design runs.

4.5 Design ON-CHIP Power Summary

Figure 9 shows the on-chip power analysis implemented in the netlist. The dynamic on-chip power is 281.916 W.

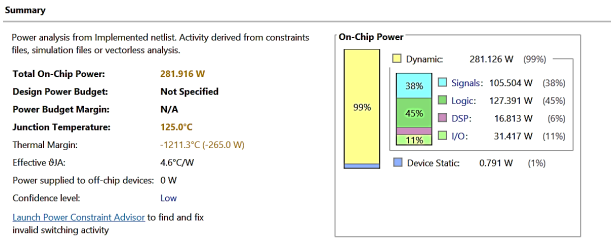

Figure 9. Summary of design ON-CHIP power.

5. CONCLUSION

A basic analysis was conducted in terms of area, power, and delay. This multiplier based on the CLA adder proved to be considerably more efficient in terms of speed of operation than traditional multipliers. The synthesis findings demonstrate that the suggested FIR filter using modified multipliers that depend on CLA adders provides high speed while reducing hardware costs and power consumption when compared to standard FIR filters using other multipliers. In future work, the pipeline concept will be extended to include the adder unit used in digital FIR filters to achieve greater power and area savings. Furthermore, the design of a 16-tap FIR filter can be expanded to include *n*-taps for usage in real-time applications.

REFERENCES

[1] Z. Lin *et al.*, "An incremental placement flow for advanced fpgas with timing awareness," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 9, pp. 3092–3103, 2022.

[2] S. Janwadkar and R. Dhavse, "Asic design of power and area efficient programmable fir filter using optimized urdhva-tiryagbhyam multiplier for impedance cardiography," *Microprocessors and Microsystems*, vol. 107, p. 105048, 2024.

[3] A. Khan, A. Chakraborty, U. B. Joy, S. Wairya, and M. Hasan, "Carry look-ahead and ripple carry method based 4-bit carry generator circuit for implementing wide-word length adder," *Microelectronics Journal*, vol. 140, p. 105949, 2023.

[4] M. Keerthi, V. Midasala, and S. N. Bhavanam, "Fpga implementation of distributed arithmetic for fir filter," *International Journal of Engineering Research*, vol. 1, no. 9, 2012.

[5] M. Kumm, A. Volkova, and S.-I. Filip, "Design of optimal multiplierless fir filters with minimal number of adders," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 2, pp. 658–671, 2023.

[6] A. Share *et al.*, "Design of high-speed bcd adder using cmos technology," *IEEE Access*, vol. 11, pp. 141 628–141 639, 2023.

[7] P. Sivaprasad, V. Anandi, and S. Murthy, "Design of high throughput and low latency double precision floating point arithmetic unit for space signal applications," *International Journal of Computer Science and Network Security*, vol. 22, no. 9, pp. 808–815, 2022.

[8] S. S. S. Priya, P. Karthigaikumar, and N. R. Teja, "Fpga implementation of aes algorithm for high-speed applications," *Analog Integrated Circuits and Signal Processing*, vol. 112, no. 1, pp. 115–125, 2022.

[9] S. C. Prasanna, B. P. James, and V. Dhandapani, "High speed and area efficient coded input bcsm shared lut-based fir filter architecture," *International Journal of System Assurance Engineering and Management*, 2024.

[10] L. Zhang, C. Rao, and X. Lou, "Low-power reconfigurable fir filter design based on common operation sharing," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 70, no. 8, pp. 3169–3173, 2023.

[11] M. Raouf and S. Timarchi, "Non-volatile and high-performance cascadable spintronic full-adder with no sensitivity to input scheduling," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 70, no. 6, pp. 2236–2240, 2023.

[12] Z. Sun, Z. Zhou, and F.-W. Fu, "Optimizing code allocation for hybrid on-chip memory in iot systems," *Integration*, vol. 97, p. 102195, 2024.

[13] K. E. Murray *et al.*, "Optimizing fpga logic block architectures for arithmetic," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 6, pp. 1378–1391, 2020.

[14] R. Kumar, R. K. Gupta, and M. K. Gupta, "Design of a high-speed and low-power 8-bit multiplier using vedic mathematics," *Microelectronics Journal*, vol. 118, pp. 105–112, 2022.

[15] D. S. S. Raja *et al.*, "A compact dual feed wide band slotted antenna for future wireless applications," *Analog Integrated Circuits and Signal Processing*, vol. 118, pp. 291–305, 2024.

[16] S. Kumarganesh *et al.*, "A novel analytical framework is developed for wireless heterogeneous networks for video streaming applications," *Journal of Mathematics*, vol. 2022, no. 1, pp. 1–7, 2022.