ESP-IJCEET
ESP International Journal of Communication Engineering & Electronics Technology
ISSN: 2583-9217 / Volume 1 Issue 3 October 2023 / Page No: 7-14
Paper Id: IJCEET-V1I3P102 / Doi: 10.56472/25839217/ IJCEET-V1I3P102

Original Article

Design and Implementation of KARATSUBA Based LMS Filters for Biological Signal Processing

S. Suganthi[1], Dr. S. Durairaj[2]

[1,2]*SRM Madurai College for Engineering and Technology, Tamilnadu, India*

Received Date: 05 September 2023 Revised Date: 18 September 2023 Accepted Date: 03 October 2023

Abstract: *This study introduces a novel approach for implementing a programmable Infinite Impulse Response (IIR) filter by leveraging the Modified Karatsuba formula. The Karatsuba formula is renowned for its capability to accelerate large number multiplication by dividing operands into equal-length parts. Traditionally, multiplying two n-digit numbers demands O(n^2) elementary operations. However, Karatsuba's breakthrough algorithm dramatically reduces this complexity to O(log^2 n) elementary steps.*

In our research, we delve into the historical context of the Karatsuba algorithm, highlighting its origins in disproving a conjecture by Andrey Kolmogorov. We provide a detailed explanation of the algorithm's fundamental steps, which involve cleverly splitting numbers and performing multiplications with additions and digit shifts. To elucidate its practical application, we present a concrete example of multiplying two numbers and outline the pseudocode for Karatsuba's algorithm.

Furthermore, we explore the concept of common subexpression elimination (CSE) in compiler optimization and its relevance to this algorithm. By eliminating redundant expressions, CSE enhances computational efficiency.

Overall, this research sheds light on the Modified Karatsuba formula's significance, offering a fresh perspective on multiplication optimization and its potential applications, particularly in VLSI architectures.

Keywords: *Biological Signal Processing, LMS Filters, Karastuba formula, Algorithm.*

I. INTRODUCTION

In the realm of computational mathematics, the quest for faster and more efficient multiplication algorithms has been a driving force for innovation. Among the myriad methods that have been devised over the years, the Modified Karatsuba formula stands as a remarkable milestone. This formula, rooted in the divide-and-conquer strategy, redefines the landscape of large number multiplication by dramatically reducing the number of required elementary operations. In this context, our research embarks on a journey to explore the Modified Karatsuba formula and its applications, particularly in the implementation of a programmable Infinite Impulse Response (IIR) filter.

Traditional multiplication of two n-digit numbers necessitates a daunting number of elementary operations, roughly proportional to n^2, as expressed in big-O notation. The classical belief, as postulated by Andrey Kolmogorov, was that this algorithm represented an asymptotically optimal solution, implying that any alternative method would inevitably demand n^2 elementary operations.

However, in a twist of fate, a 23-year-old student named Karatsuba shattered this belief in 1960. During a seminar on mathematical problems in cybernetics at the Moscow State University, Kolmogorov introduced his Ω̇n^2 conjecture, among other computational complexity challenges. Within a mere week, Karatsuba formulated an algorithm, often referred to as "divide and conquer," that defied this conjecture. This algorithm could multiply two n-digit numbers in just O(log^2 n) elementary steps, a monumental breakthrough. Kolmogorov was astounded by this discovery and presented it at the seminar, leading to the termination of the event. Subsequently, Karatsuba's algorithm was shared with the global mathematical community.

At its core, the Modified Karatsuba formula employs a strategic approach to multiplication. It divides the operands into two equal-length parts and leverages clever mathematical transformations to achieve faster results. We illustrate this process through concrete examples and provide pseudocode to demonstrate its practicality. Furthermore, our research delves into the concept of common subexpression elimination (CSE) in the context of compiler optimization. We explore how CSE can enhance computational efficiency by identifying and eliminating redundant expressions, making the Modified Karatsuba formula even more potent.

As we delve deeper into this exploration, it becomes evident that the Modified Karatsuba formula holds immense potential for various applications beyond mathematics. One such application is in the realm of VLSI (Very Large-Scale Integration) architectures, where computational efficiency is paramount. By understanding and harnessing the power of this formula, we pave the way for more efficient and streamlined algorithms, ultimately advancing the field of computation. In essence, this research journey takes us from the historical origins of the Karatsuba algorithm to its modern-day applications, shedding light on the enduring quest for computational efficiency and optimization. The implications of this work extend far beyond mathematics, reaching into the heart of technology and innovation.

II. LITERATURE SURVEY

In their study, Khan et al introduced a novel architecture tailored for distributed arithmetic (DA)-based Least Mean Square (LMS) adaptive filters. This innovative design prioritizes reduced hardware complexity and a shorter critical path. It's widely acknowledged that the throughput of a DA-based adaptive filter hinges on both the critical path and the number of clock cycles required to produce the output.

Meanwhile, He et al ventured into the implementation of a 14-bit, 200-MS/s Successive Approximation Register (SAR) analog-to-digital converter (ADC) devoid of the traditional Sample-and-Hold Amplifier (SHA). They utilized a 0.13 μm CMOS process and incorporated a blind Least Mean Square (BLMS) calibration technique. This approach effectively corrected errors in the pipelined ADC, leveraging fast, low-gain, and imprecise operational amplifiers. By employing a skip and fill technique and integrating an interpolation filter along with a front-end Digital-to-Analog Converter (DAC), the pipelined ADC was rendered self-calibratable in real-time.

In contrast, Siva et al proposed an efficient Very Large Scale Integration (VLSI) architecture for LMS adaptive filters using distributed arithmetic. Distributed arithmetic involves serial bit computation. In their design, two distinct lookup tables were employed to store potential filter partial product combinations of input samples and filter coefficients. This was followed by accessing and summing up these entries, marking a significant departure from conventional approaches.

Bujjibabu et al introduced an innovative shift-add tree structure that effectively minimized the critical path and silicon area without increasing the number of adaptation delays. Their approach to designing an Infinite Impulse Response (IIR) adaptive filter was centered on two key blocks: the IIR block and the new coefficients block (weights block). The weights block consisted of a series of partial product generators and a shift/add tree.

Mula et al's research was centered on the implementation of Pt-NLMS algorithms in hardware, a pioneering effort. They proposed various reformulations to simplify the original Pt-NLMS algorithms, making them suitable for real-time VLSI implementations. These reformulated algorithms, referred to as the delayed μ-law proportionate LMS (DMPLMS) for white input and delayed wavelet MPLMS (DWMPLMS) for colored input, were successfully implemented in hardware.

Lu et al tackled the implementation of a 1T synapse circuit capable of performing multiply/divide/sum operations, crucial for realizing an adaptive neural network. They demonstrated the feasibility of employing this circuit in adaptive neural networks through a 4-bit analog-to-digital converter circuit based on the Hopfield modified neural network model with analog LMS adaptive feedback.

Chin et al put forth a digit-serial VLSI architecture tailored for the realization of an adaptive Finite Impulse Response (FIR) filter, equipped with the delayed Least Mean Square (LMS) algorithm. Their architecture is particularly well-suited for applications where bit-serial arithmetic is too sluggish, and bit-parallel arithmetic demands excessive hardware or falls short in achieving the desired convergence performance.

Ramanathan et al developed a systolic architecture characterized by minimal adaptation delay and input/output latency, substantially enhancing convergence behavior, akin to the original LMS algorithm. This architecture was achieved through a series of function-preserving transformations applied to the signal flow graph representation of the delayed LMS algorithm. Meanwhile, Song et al embarked on the implementation of an algorithm for echo cancellation in VLSI, with a specific focus on minimizing the word length of coefficients for cost-efficiency. They acknowledged that an insufficient word length could degrade cancellation performance but mitigated this issue through the use of a non-uniform number representation system.

In another vein, Govil et al introduced an estimation framework for constant delay, actively identifying the active regions. They integrated this framework with the fast LMS/Newton algorithm to efficiently realize long adaptive filters. Their assumption was rooted in modeling the input sequence to the adaptive filter as an autoregressive (AR) process, allowing for a significantly lower order compared to the adaptive filter's length. Lastly, Tiu et al presented an innovative VLSI design for adaptive Infinite Impulse Response (IIR) filters, tailored for high-speed channel echo cancellation. Their design leveraged the delayed LMS adaptation algorithm to exploit computational concurrency between the adaptation and filtering sections. Furthermore, they employed a non-recursive equation error criterion to expedite computation within the adaptation section.

III. PROPOSED SYSTEM

In this research endeavor, we propose the implementation of a programmable Infinite Impulse Response (IIR) filter that leverages the Modified Karatsuba formula, a powerful mathematical technique originally designed to expedite large number multiplication. The Karatsuba formula achieves this acceleration by splitting the operands into two equal-length parts. The conventional method for multiplying two n-digit numbers typically necessitates a number of elementary operations proportional to n^2, often expressed as O(n^2) in big-O notation. Notably, Andrey Kolmogorov initially conjectured that this classical algorithm was asymptotically optimal, implying that any algorithm for this task would invariably require n^2 elementary operations.

In a pivotal turn of events, in 1960, during a seminar on mathematical problems in cybernetics at the Moscow State University, Kolmogorov introduced his Ὼn^2 conjecture, among other computational complexity challenges. Astonishingly, within a week, a 23-year-old student named Karatsuba formulated an algorithm (later referred to as "divide and conquer") that could multiply two n-digit numbers in O(log^2 n) elementary steps, thereby refuting the conjecture. Kolmogorov was greatly impressed by this discovery, presenting it at the following seminar meeting, which was subsequently terminated due to the breakthrough. Kolmogorov lectured on the Karatsuba algorithm at various international conferences and published it in 1962, attributing authorship to "A. Karatsuba and Yu. Ofman." Remarkably, Karatsuba himself only became aware of this paper when he received reprints from the publisher.

The core concept underpinning Karatsuba's algorithm involves a formula that enables the computation of the product of two large numbers, x and y, using merely three multiplications of smaller numbers, each possessing approximately half the digits of x or y. This foundational step is, in essence, a generalization of Gauss's complex multiplication algorithm, albeit with the replacement of the imaginary unit "i" with a power of the base.

In practice, let x and y be represented as n-digit strings in some base B. For any positive integer m less than n, we can express the two numbers as follows:

$x = x_1 * B^m + x_0$

$y = y_1 * B^m + y_0$

These expressions involve four multiplications and were actually known to Charles Babbage. Karatsuba ingeniously observed that the product xy could be computed with just three multiplications, albeit with some additional additions and digit shifts. Calculating z0 and z2 as previously described, we can determine:

$z_1 = (x_1 + x_0) * (y_1 + y_0) - z_2 - z_0$

An interesting challenge arises when computing z1, as the addition of (x1 + x0) and (y1 + y0) may lead to overflow, necessitating an additional bit. However, this issue can be circumvented by considering:

x0 - x1) * (y1 - yo)

This computatioresults in a range of values within -bm < result < bm, where "b" is the base. While this method may yield negative numbers, an extra bit for signedness and an additional bit for the multiplier are still required.

To illustrate this approach, let's consider the product of 12345 and 6789 in base 10 (B = 10), choosing m = 3 for decomposition. The operands are represented as:

12345 = 12 * 1000 + 345

6789 = 6 * 1000 + 789

With only three multiplications involving smaller integers, we obtain three partial results:

z2 = 12 * 6 = 72

z0 = 345 * 789 = 272205

z1 = (12 + 345) * (6 + 789) - z2 - z0 = 357 * 795 - 72 - 272205 = 283815 - 72 - 272205 = 11538

Finally, the result is obtained by adding these three partial results, each shifted accordingly, and accounting for carries, following decomposition similar to that used for the input operands:

Result = z2 * (B^m*2) + z1 * (B^m*1) + z0 * (B^m*0)

Result = 72 * 1000^2 + 11538 * 1000 + 272205 = 83810205.

It's important to note that the third multiplication operates on an input domain slightly less than twice the size of the first two multiplications, with an output domain smaller than four times the size. Additionally, when performing base-1000 carries, they must be considered when processing these three inputs. In terms of pseudocode for the Karatsuba algorithm, the procedure involves splitting the input numbers and recursively computing the products using three multiplications and several additions, with proper shifting and base considerations.

Furthermore, the implementation includes a concept called "common subexpression elimination (CSE)," which is a compiler optimization technique that identifies identical expressions and replaces them with a single variable to minimize redundancy. This optimization relies on available expression analysis and can be performed locally within a single basic block or globally across an entire procedure.

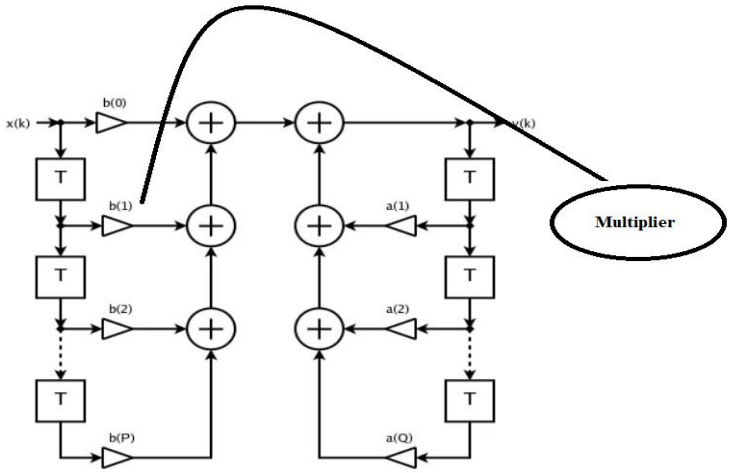

Figure 1: IIR Hardware Architecture

In summary, the Modified Karatsuba formula revolutionizes multiplication by reducing the number of required elementary operations, with applications extending from mathematical problem-solving to VLSI architectures. This algorithm's elegance lays in its ability to achieve more efficient computations through strategic decomposition and common subexpression elimination, ultimately benefiting various computational and hardware domains SIMULATION RESULTS

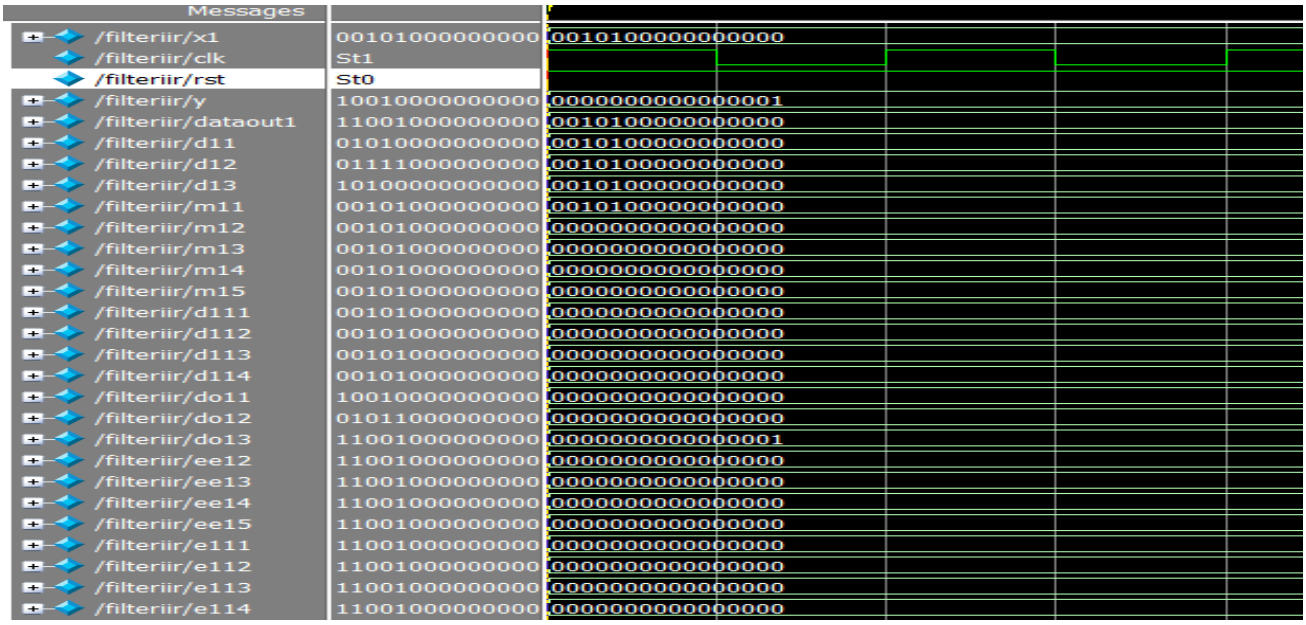

Figure 2: Filter Output Reset=1

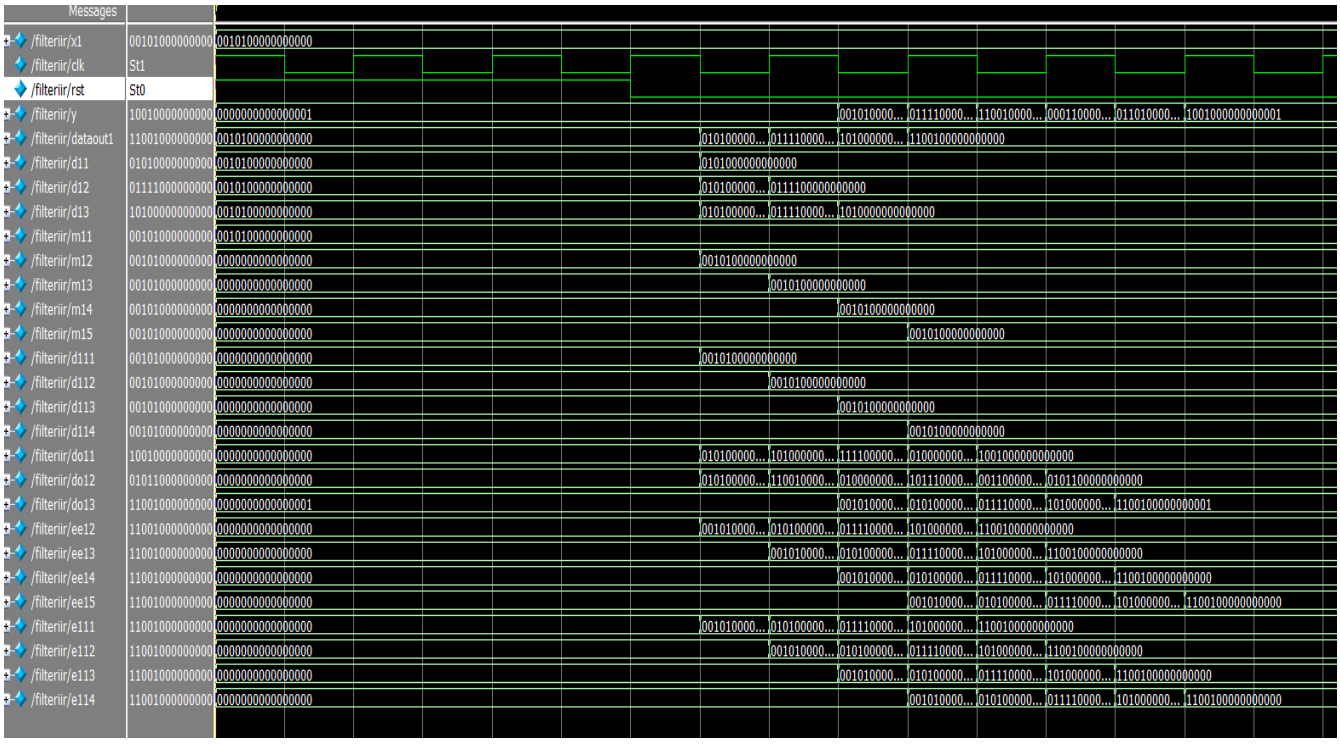

Figure 3: Filter Output Reset=0

IV. XILINX SYNTHESIS REPORT

A. Existing System:

Design Goal:	Balanced	• Routing Results:
Design Strategy:	Xilinx Default (unlocked)	• Timing Constraints:
Environment:	System Settings	• Final Timing Score:

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	612	960	
Number of 4 input LUTs	1106	1920	
Number of bonded IOBs	33	83	

Detailed Reports				
Report Name	Status	Generated	Errors	Warnings
Synthesis Report	Current	Thu Oct 4 16:10:56 2018	0	24 Warnings (21 new)
Translation Report				

Figure 4: Existing System Report

B. Proposed System:

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	503	960	
Number of 4 input LUTs	919	1920	
Number of bonded IOBs	33	66	

Detailed Reports					
Report Name	Status	Generated	Errors	Warnings	Infos
Synthesis Report	Current	Sun Mar 28 06:00:26 2021	0	25 Warnings (0 new)	0
Translation Report					

Figure 5: Proposed System Report

The proposed has been simulated and the synthesis report can be obtained by using Xilinx ISE 12.1i. The various parameters used for computing existing and proposed systems with Spartan-3 processor are given in the table 7.1.

After performing the synthesize process, the RTL schematic has been created automatically based on the functionality. The routing between the different cells can be viewed clearly by this schematic.

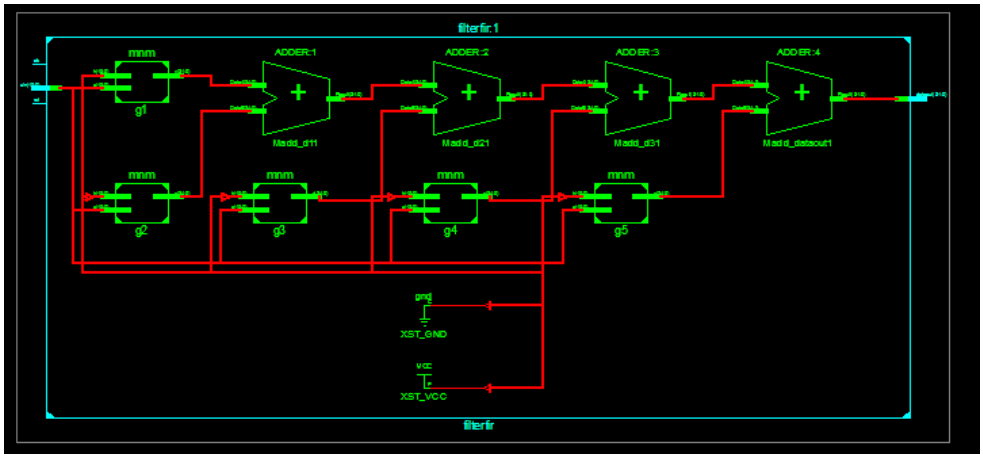

Figure 6: Gate Level Netlist

V. PERFORMANCE ANALYSIS

The Figure given below is shown that there is a considerable reduction in time and area based on the implementation results which have been done by using Spartan-3 processor. The proposed algorithm significantly reduces area consumption when compared to the existing system.

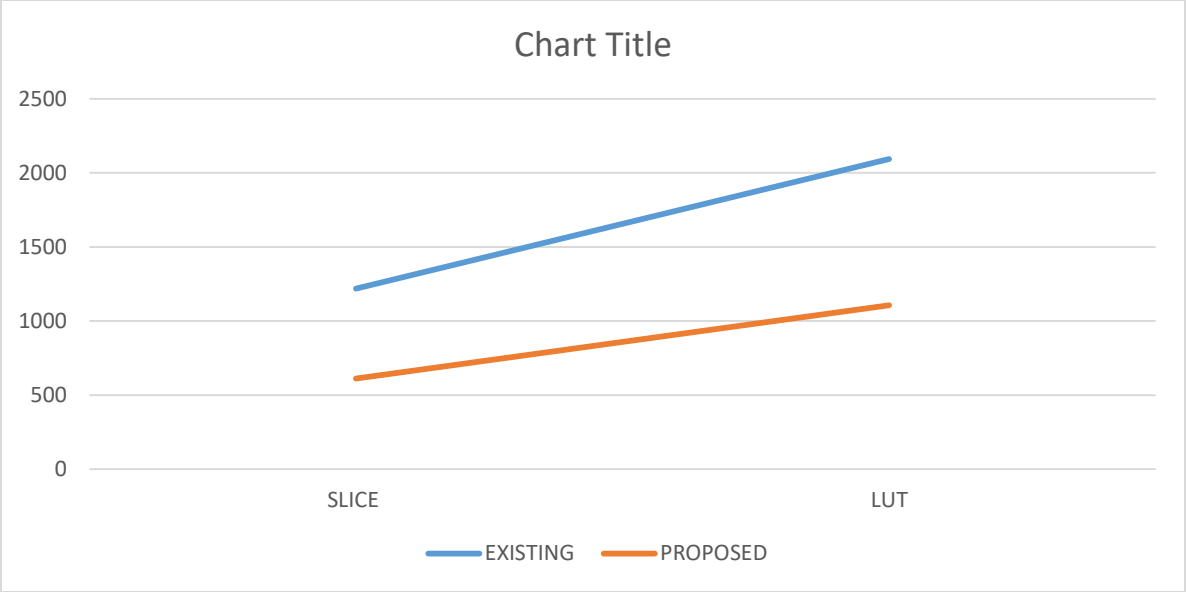

Figure 7: Performance Analysis Chart

VI. CONCLUSION

In conclusion, this research presents an innovative approach to implementing a programmable Infinite Impulse Response (IIR) filter using the Modified Karatsuba formula. The historical context of Karatsuba's groundbreaking algorithm, which revolutionized large number multiplication, has been elucidated. By dissecting the algorithm's core steps, we have demonstrated its ability to significantly reduce computational complexity, making it a valuable tool for various applications.

The discussion of common subexpression elimination (CSE) as a compiler optimization technique has further highlighted the algorithm's relevance in computational efficiency and code optimization.

Overall, this study underscores the enduring significance of the Karatsuba algorithm and its potential impact on diverse fields, particularly in the realm of VLSI architectures. As computation continues to play a pivotal role in technology, the optimization techniques explored in this research hold promise for enhancing the efficiency of various computational processes.

VII. REFERENCES

[1] J. Govil, "Enhanced Residual Echo Cancellation using Estimation of Delay and Fast LMS/Newton Algorithm based on Autoregressive Model," 2016 40th Annual Conference on Information Sciences and Systems, Princeton, NJ, 2006, pp. 1356-1356, doi: 10.1109/CISS.2006.286675.

[2] S. Ramanathan and V. Visvanathan, "A systolic architecture for LMS adaptive filtering with minimal adaptation delay," Proceedings of 9th International Conference on VLSI Design, Bangalore, India, 1996, pp. 286-289, doi: 10.1109/ICVD.1996.489612.

[3] Chin-Liang Wang, Ching-Chia Chen and Che-Fu Chang, "A digit-serial VLSI architecture for delayed LMS adaptive FIR filtering," Proceedings of ISCAS'95 - International Symposium on Circuits and Systems, Seattle, WA, USA, 1995, pp. 545-548 vol.1, doi: 10.1109/ISCAS.1995.521571.

[4] T. C. Lu, M. L. Chiang and J. B. Kuo, "A one-transistor synapse circuit with an analog LMS adaptive feedback for neural network VLSI," 1991., IEEE International Sympoisum on Circuits and Systems, Singapore, 1991, pp. 1303-1306 vol.3, doi: 10.1109/ISCAS.1991.176610.

[5] S. Mula, V. C. Gogineni and A. S. Dhar, "Algorithm and VLSI Architecture Design of Proportionate-Type LMS Adaptive Filters for Sparse System Identification," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 26, no. 9, pp. 1750-1762, Sept. 2018, doi: 10.1109/TVLSI.2018.2828165.

[6] P. Bujjibabu and K. Sirisha, "Design and implementation of efficient IIR LMS adaptive filter with improved performance," 2017 International Conference on Big Data Analytics and Computational Intelligence (ICBDAC), Chirala, 2017, pp. 240-245, doi: 10.1109/ICBDACI.2017.8070841.

[7] S. R. B, G. S. L and N. C K, "FPGA based Optimized LMS Adaptive Filter using Distributed Arithmetic," 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India, 2018, pp. 1863-1867, doi: 10.1109/RTEICT42901.2018.9012288.

[8] Y. He, B. Chen and Q. Li, "Blind-LMS based digital background calibration for a 14-Bit 200-MS/s pipelined ADC," 2013 IFIP/IEEE 21st International Conference on Very Large Scale Integration (VLSI-SoC), Istanbul, 2013, pp. 348-351, doi: 10.1109/VLSI-SoC.2013.6673307.

[9] M. T. Khan, S. R. Ahamed and F. Brewer, "Low Complexity and Critical Path Based VLSI Architecture for LMS Adaptive Filter Using Distributed Arithmetic," 2017 30th International Conference on VLSI Design and 2017 16th International Conference on Embedded Systems (VLSID), Hyderabad, 2017, pp. 127-132, doi: 10.1109/VLSID.2017.16.

[10] Yin-Tsung Hwang and Chun Shang Lin, "VLSI design of DLMS adaptive IIR filters for high speed echo cancellation," 1997 IEEE Workshop on Signal Processing Systems. SiPS 97 Design and Implementation formerly VLSI Signal Processing, Leicester, UK, 1997, pp. 341-350, doi: 10.1109/SIPS.1997.626258.