A High Speed and Delay Efficient Adaptive Hold Logic Based KS Multiplier

[1]D. Sony, [2]K. Anusha

[1]M.Tech Scholar, Department of Electronics and Communication Engineering, Priyadarshini Institute of Technology and Management, Pulladigunta, Guntur, Andhra Pradesh, India

[1]Assistant Professor, Department of Electronics and Communication Engineering, Priyadarshini Institute of Technology and Management, Pulladigunta, Guntur, Andhra Pradesh, India

ABSTRACT: The multiplier is the fundamental component of many computing modules. As the most important component of a multiplier, the full adder (FA) also has a significant impact on the overall performance. Full adders based on pass transistor logic (PTL) have been a very popular research field in recent years, but the uneven delay makes it difficult to analyze the critical path of multipliers based on PTL full adders. Therefore, a high speed and delay efficient adaptive hold logic based KS (Kogge-Stone) multiplier is introduced. This proposed uses Adaptive Hold Logic (AHL) based Kogge-Stone Multiplier is used in high-performance digital systems to ensure reliable, long-term operation. It pairs the extremely fast computation speed of a Kogge-Stone adder with a specialized circuit that detects and mitigates physical hardware degradation as the chip ages.

KEYWORDS: Multipliers, Full Adder (FA), Pass Transistor Logic (PTL), Adaptive Hold Logic (AHL), KS (Kogge-Stone)

I. INTRODUCTION

Many digital signal processing (DSP) and machine learning applications are heavily dominated by multiplication [1]. the multiplier is an important component in various hardware platforms. The conventional multiplication includes three major phases [2]. (1) Two inputs (multiplier and multiplicand) are multiplied to generate the partial products (PPs). (2) Reducing the PPs' matrix into two rows by partial product reduction schemes (3) The final carry propagated addition of the remaining two rows of PPs. Particularly, the second phase plays a significant role in power consumption, cost, and overall performance [3]. Then, the radix-4 Booth algorithm can improve the performance of multiplication because the radix-4 Booth multiplier can reduce the number of PP rows by half [4].

Among the above-mentioned parallel paradigms, Approximate Computing has emerged as a viable solution to resource-efficient computing in the face of exploding data, increased demands, and complexity of computations [5]. Approximate computing has shown significant efficiency gains with respect to power, chip area, and performance/ throughput by exploiting the inherent error resilience of applications [6]. It works on the principle of relaxed precision, i.e., trading off tolerable accuracy losses (in applications where approximate computations still produce quality results) to enable novel optimizations, keeping a good balance between quality, reliability, and hardware efficiency [7].

The traditional approximate computing approaches [8] such as fail small (small error magnitude, high error rates), fail rare (high error magnitude, small error rates), and fail moderate (moderate error magnitude, moderate error rates), restrict errors based on their magnitudes and rates. These conventional approaches explore only a limited design space, excluding approximations with both high error magnitude and high error rate [9]. Without error cancellation, high error/ high rate designs would fail even in error-tolerant domains. This exclusion limits the potential

efficiency gains for a given quality constraint and undermines the quality–efficiency trade-off [10]. To overcome the constraint of a limited design space, more recent fail balanced techniques, also known as self-healing methodologies, offer an effective quality-efficiency trade-off by providing an opportunity for error cancellation [11].

Approximate computing assumes small approximations in computing while maintaining an acceptable quality of results [12]. Approximate computing has become a popular strategy for arithmetic circuit design, where energy-efficient design is more important than accuracy, and the challenge is to achieve the best trade-off between accuracy and design efficiency [13]. The principal requirement for an approximate arithmetic circuit is to exhibit a controllable error with design efficiency as the central goal. The multipliers are the indispensable components in many systems, e.g., digital signal processors, embedded vision systems and hardware accelerators. At the same time, multipliers represent the most complex arithmetic components in the digital implementation of these systems (e.g., hardware deep neural networks) [14]. Thus, obtaining an efficient multiplier design is of the highest importance. The multiplier design can be remarkably simplified if we exclude the requirement for exact computation, so various versions of approximate multiplier design have been proposed recently.

II. LITERATURE SURVEY

S. V. S. Prasad, N. S. S. Sagar, M. V. Nitya Pushkala, S. Silveri, B. Harshitha and S. Chenna et al. [15] proposes a novel technique towards enhancing VLSI performance as 3-Parallel Polyphase Odd-Length FIR Filtering. Here, the design primarily consists of Kogge-Stone Adder integrated with Booth Multiplier to increase the speed and efficiency of the Kogge-Stone Adder. To address the issues, we make use of the architectural features of Kogge-Stone Adder and the computational efficiency of Booth Multiplier by introducing the 3-Parallel Polyphase composition to overcome the issues by increasing the processing speed while maintaining low-power constraints and a small chip area, both of which are critical factors in today's VLSI applications.

Y. K, G. R, S. S, T. R. T and V. T, et al. [16] analyze the 16x16 Vedic Multiplier architecture using the Kogge-Stone Adder. By increasing the width of the transistor at the current stage, the delay can be reduced. The Vedic multiplier's speed will be increased, and the delay that results from utilizing the half-adder and full adder will be minimized. Therefore, the Kogge-Stone Adder is crucial in this kind of situation for reducing the chip's complexity and delay.Multipliers are significant building blocks in digital systems and are essential to the success of digital designs. Multiplication is the essential calculating process in digital systems.

S. Nikhil and P. V. Lakshmi, et al. [17] propose a design of 8-bit multipliers using fast adders (carry save adder, kogge-stone adder and carry-select adder) to minimize the power delay product of multipliers intended for high-performance applications. Implementation results demonstrate that the proposed Vedic multipliers with fast adders really achieve significant improvement in delay and power-delay product when compared with the conventional multipliers.

G. V. Nikhil, B. P. Vaibhav, V. G. Naik and B. S. Premananda, et al. [18] design of these building blocks to dissipate less power is of utmost importance in digital system design. In order

to reduce this power dissipation, there are many low power approaches such as Multi-Vth, reducing the voltage swing, clock gating, use of reversible logic gates etc. Reversible Logic gates are that logic gates which have the same number of inputs and outputs and all the outputs are unique for a given input combination. These gates are used to reduce the power dissipation due to loss of information bits. The proposed reversible Vedic Multiplier consumed 0.621 % less power than the conventional Vedic Multiplier. The power dissipation of reversible Barrel Shifter is found to be 26.49 % less than conventional Barrel Shifter.

S. Malagar, D. P and R. S, et al. [19] digital circuits such as, Carry Select Adder, Carry Conditioned Adder and Kogge-Stone Adder implemented using memristor-based IMPLY logic. Further, a 16-bit multiplier developed using a Kogge-Stone adder with memristor-based IMPLY logic. Finally Xilinx Vivado software is used to implement the work and verify the simulation results. The results demonstrate that memristor-based designs offer significant advantages in terms of area and power efficiency. Further, the integration of memristor-based IMPLY logic with the Kogge-Stone adder in the 16-bit multiplier design leads to notable reductions in area and power consumption.

G. Rohith, P. Jagadeesh and N. Meenakshisundaram, et al. [20] described a novel Dadda multiplier was designed using VHDL and tested for delay and power consumption performance in this work. We used ModelSim for simulation and the BKA and KSA adders for design carrying out. For the purpose of calculating the power usage and delay, the simulations were run with various input sizes and operation frequencies. In order to evaluate the multiplier's performance, the output waveforms were shown and studied. Using the BKA and the KSA, an 8-bit Novel Dadda multiplier was developed and evaluated on 20 sample data sets, 10 samples for each adder. With a beta of 0.20 and an alpha of 0.05, the results were based on a 95% confidence interval. evaluated using G-power software. With a power usage of 4.75 Watts and a delay of 27.45 nanoseconds, the old Kogge Stone Adder (KSA) was less efficient.

K. Sathish and P. Jagadeesh, et al. [21] design and implementation of an 8-bit modified Wallace multiplier using the proposed Kogge-Stone Adder (KSA) and compares its performance with the Ripple carry adder (RCA). Materials and Methods: In this research, a VHDL implementation of an 8-bit modified Wallace multiplier based on the KSA and the RCA has been developed. Modelsim is used to simulate the design, and the results are compared with those of the RCA-based design. The study was able to visualise the waveforms produced by the multiplier and assess its effectiveness with the help of this programme. The simulation results of the proposed 8-bit modified Wallace multiplier using the KSA showed a delay of 28.5 ns, which is significantly lower compared to the delay of 36.1 ns in the traditional RCA based design.

A. Sundhar, S. D. Tharshini, G. Priyanka, S. Ragul and C. Saranya, et al. [22] design approach of 16bit Wallace Tree approximate multiplier with 15-4 compressor is considered to provide more reliability. The 16×16 Wallace tree multiplier is synthesized and simulated using Xilinx ISE 14.5 software. The multiplier occupies about 15% of total coverage area. The dissipated power and delay of the multiplier are 0.042μw, 3.125ns respectively.

A. Raju and S. K. Sa, et al. [23] designing and implementing multipliers using the underlying principle of parallel prefix adders. We are looking for less delay specific multiplier, so among

the prominent PPA's like Kogge Stone Adder(KSA), Sparse Kogge Stone Adder(SKSA), Brent Kung Adder(BKA) and Lander Fischer Adder(LFA), we choose to implement the fastest PPA, i.e., KSA to get a comparative idea about the performance of four different multipliers namely; Binary multiplier, Braun Multiplier, Yedic Multiplier and Baugh Wooley Multiplier. Further the synthesis and simulation results reveal better idea about the proposed multipliers by giving an in depth view of their area, delay and power.

S. Shirahatti, D. Haritha, P. Thejaswini, A. J. Deepika and B. M. Yashaswini, et al. [24] implementing an 8×8 Vedic multiplier utilizing the Kogge Stone Adder and optimizing for power using reversible logic. The design is simulated and synthesized in Cadence tool 180 nm technology. Performance evaluation of the Vedic multiplier is conducted, assessing its area, speed, and power consumption. The performance of Vedic multiplier implemented with conventional gates is compared with the implementation using reversible gates. The comparison shows considerable improvement in power for multiplier with reversible logic. The synthesis results show that power reduction by 7.2% over conventional gate implementation. At the same time, the implementation suffers from area and datapath delay overhead by 16.87% and 27% respectively.

K. K R, S. Vijayalakshmi, B. M. Babu, P. K and N. S, et al. [25] proposes an innovative high-speed Wallace tree multiplier architecture that incorporates the Kogge-Stone adder to optimize performance for VLSI applications. By combining the efficient partial product reduction capabilities of the Wallace tree with the Kogge-Stone adder's fast carry propagation, this design significantly improves multiplier performance. Detailed analysis and simulation results highlight notable enhancements in critical path delay, area efficiency, and power consumption compared to traditional multiplier designs. These substantial improvements make the proposed architecture especially well-suited for applications that demand high-throughput arithmetic operations, such as digital signal processing, image processing, and complex scientific computations.

III. FRAMEWORK OF A HIGH SPEED AND DELAY EFFICIENT ADAPTIVE HOLD LOGIC BASED KS MULTIPLIER

Multiplier and Multiplicand are the two inputs, where Multiplier operand is used for multiplication process and Multiplicand is used for second operaation. These two inputs are given to the partial product generation. The movement of multiplier bits is controlled by shift register that sequentially shifts the multiplier data and provides control signals for generating partial products that helps in organizing multiplication operations efficiently and reduces hardware complexity. Each block generates a group of partial products from the multiplier and multiplicand bits and these partial products are intermediate multiplication results. AHL monitors the multiplication operation and predicts whether the computation can be completed within the current clock cycle to reduces timing violations and minimizes unnecessary waiting cycles. Pre-Computation Unit is performed for preliminary addition and reduction operations to compresses multiple partial products into fewer rows and reduces the complexity of final addition. Kogge-Stone (KS) Adder is appilied. This adder generates carry signals in parallel and uses a tree structure to propagate carries rapidly. Finally, carry outputs and reduced partial products are combined to produces the final multiplication result.

A Kogge-Stone Multiplier combines the partial product reduction stages of algorithms like Wallace Tree or Vedic mathematics with the Kogge-Stone Adder. This design is used in digital signal processing and VLSI systems to achieve exceptionally high-speed multiplication by eliminating redundant carry logic in the final addition phase.

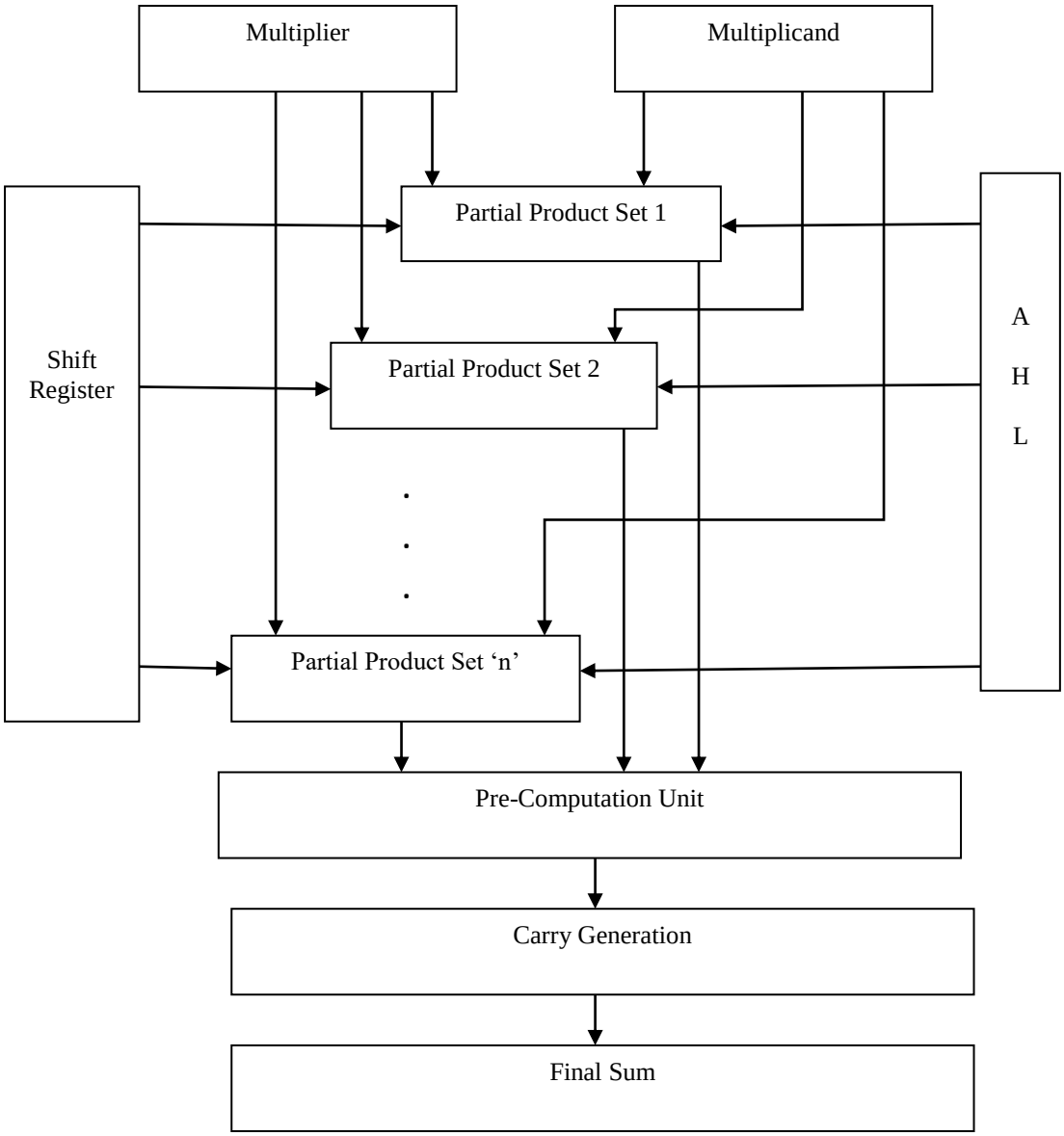

Figure 1. Framework Of A High Speed And Delay Efficient Adaptive Hold Logic Based KS Multiplier

One key component of arithmetic and logic units is the multiplier. In order for numerous real-time signal and image processing applications to operate effectively, faster arithmetic operations are required. The speed and power dissipation are the key factors in a multiplier. The biggest issue with the multiplier is its worst-case delay. This leads to a decrease in both the path delay and the time delay. The lowest switching activity, or the overall number of signal transitions in the system, will cause a delay as a digital signal travels from the input of the logic gate to the output gate. Reducing power consumption can help address challenges associated with other

multiplication methods by decreasing complexity, speeding up execution, and conserving power. The most crucial component of the variable-latency adder design is the AHL. It contributes to the system's increased operating speed. A model called Cyclic Redundancy Check (CRC) is used to identify data errors or corrupted information. A generator polynomial is used on both the sender and recipient sides of a CRC.

A single binary digit (1 or 0) can be stored in flip flops. Nevertheless, a variety of flip-flops are needed in order to store numerous bits of data. To hold n bits of data, you need to link N flip flops together. A register is a tool used to store this type of data. It is a series of flip-flops connected to store multiple data points. Using shift registers helps to transfer the data stored within them. A shift register, which is a group of flip flops, is utilized to hold multiple data elements. Clock pulses have the ability to shift the bits stored in registers, moving them internally and externally. Linking n bistables, each of which holds a lone bit of data, can form an n-bit shift register. "Left shift registers" are registers that will shift the bits to the left. "Right shift registers" are registers that shift bits to the right. The most crucial component of the variable-latency multiplier design is the Adaptive Hold Logic (AHL). Two judging blocks, an aging indicator, a D flip-flop, and a 2:1 multiplexer make up the AHL circuit. The AHL block's aging indicator is mainly used to indicate if the design has seen a significant decrease in performance from aging effects. The aging indicator is built with a basic counter that tallies the errors occurring during a set period of operation. The counter resets to zero after reaching the predetermined amount. If the cycle period is too brief, the Vedic multiplier may not function effectively, leading to timing violations.

IV. RESULT ANALYSIS

In this section, result analysis of a high speed and delay efficient adaptive hold logic based KS multiplier is observed.

The Register Transfer Level (RTL) schematic, which is depicted in Figure 2, illustrates the data flow through the implementation logic of a circuit.RTL is a high-level method of describing a circuit's architecture, and is often used to create VHDL code and schematics

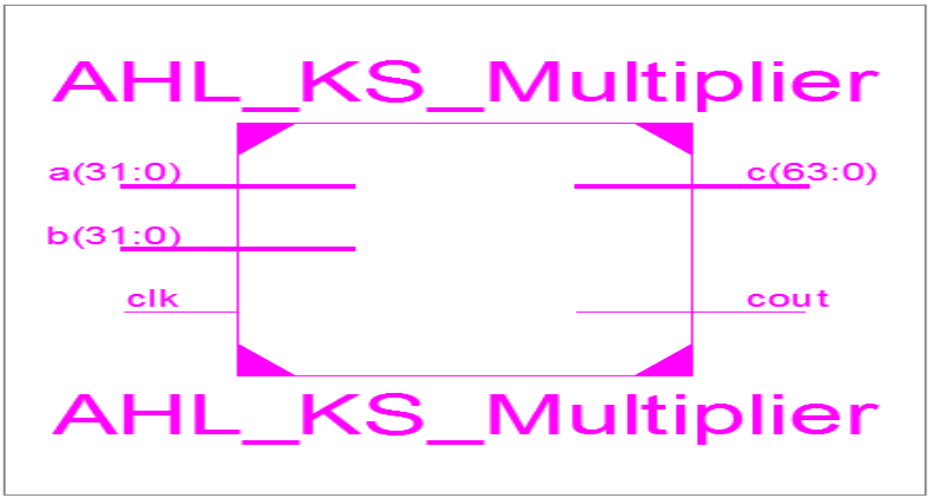

Figure 2. RTL SCHEMATIC

In Figure 3, a technology schematic is a first-level design that includes a technology schematic and an RTL schematic. It is a representation of a system's elements using abstract symbols and graphic representations instead of realistic pictures. Schematics are used to represent technical systems in a simplified way.

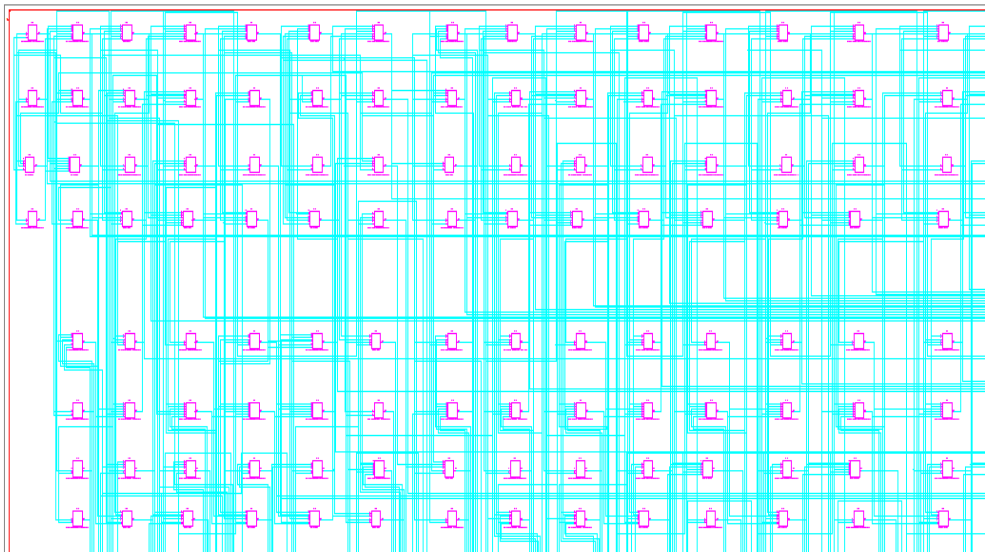

Figure 3. Technology schematic

In Figure 4, A digital circuit known as a lookup table (LUT) can be used in FPGA design to implement any Boolean function. A number of input variables and an output value that is determined by the input variables make up this system.

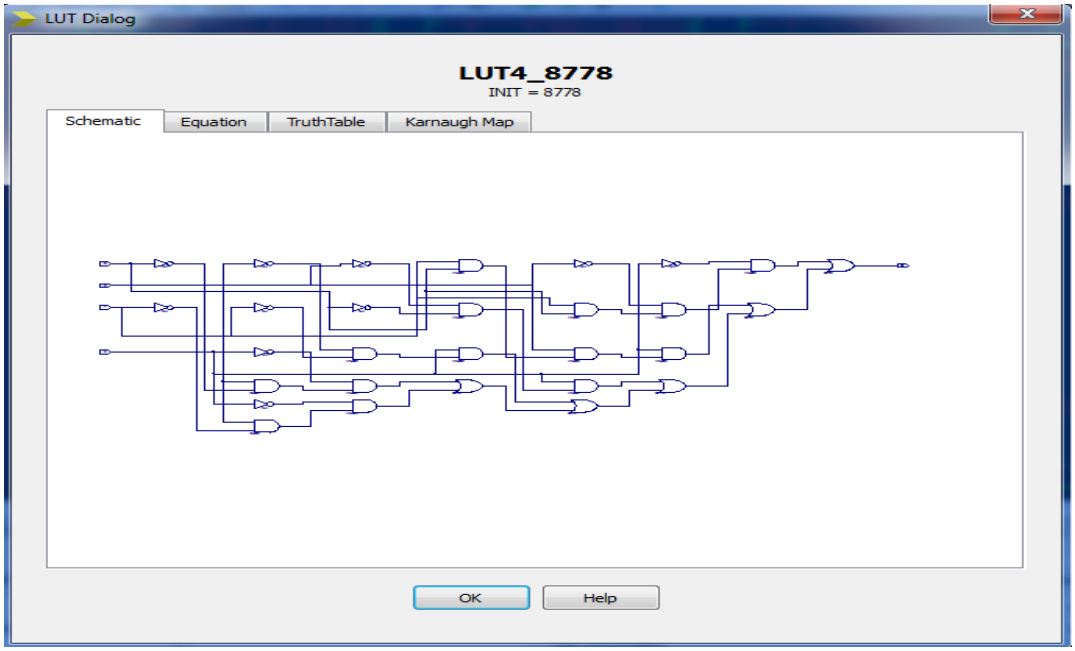

Figure 4. LUT

In Figure 5, VLSI, the waveform window displays simulation output that can be cross-probed to a layout or schematic.

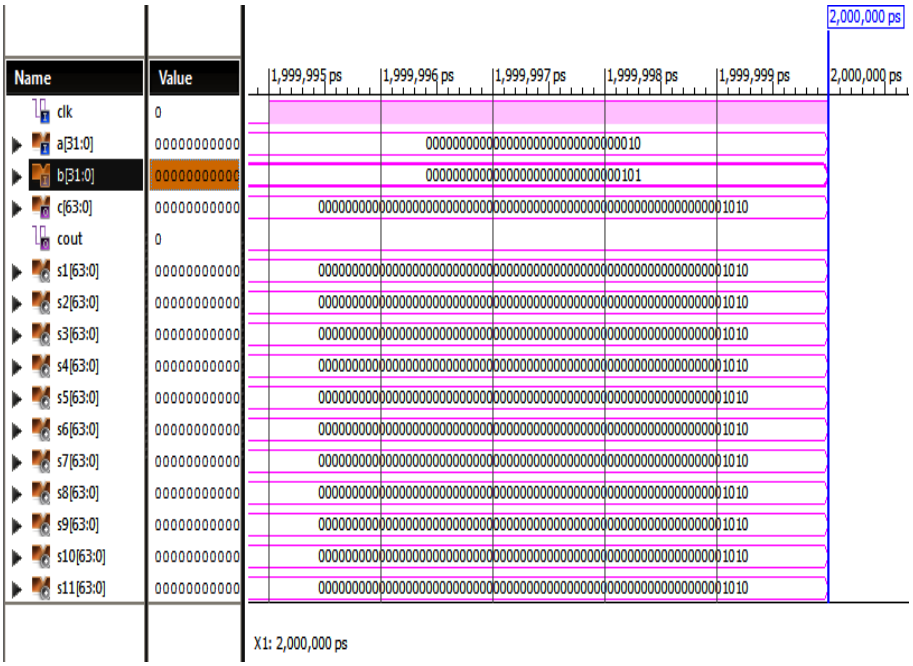

Figure 5. Output

AHL_KS_Multiplier Project Status			
Project File:	Project.xise	**Parser Errors:**	No Errors
Module Name:	AHL_KS_Multiplier	**Implementation State:**	Synthesized
Target Device:	xc6vcx75t-2ff484	**• Errors:**	No Errors
Product Version:	ISE 14.7	**• Warnings:**	1 Warning (1 new)
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 Slice LUTs	2017	46560	4%
Number of fully used LUT-FF pairs	0	2017	0%
Number of bonded IOBs	129	240	53%

Figure 6. AHL Report

Table 1. Performance Comparison

Parameters	Existing System	Proposed System
Total Delay	64.248ns	40.400ns
Logic Delay	16.861ns	3.849ns
Route Delay	47.387ns	36.551ns
Memory Used	647852kb	289592kb

In Figure 7, total delay comparison graph is represented between existing system and proposed system is observed. The proposed system shows low delay when compared with existing system.

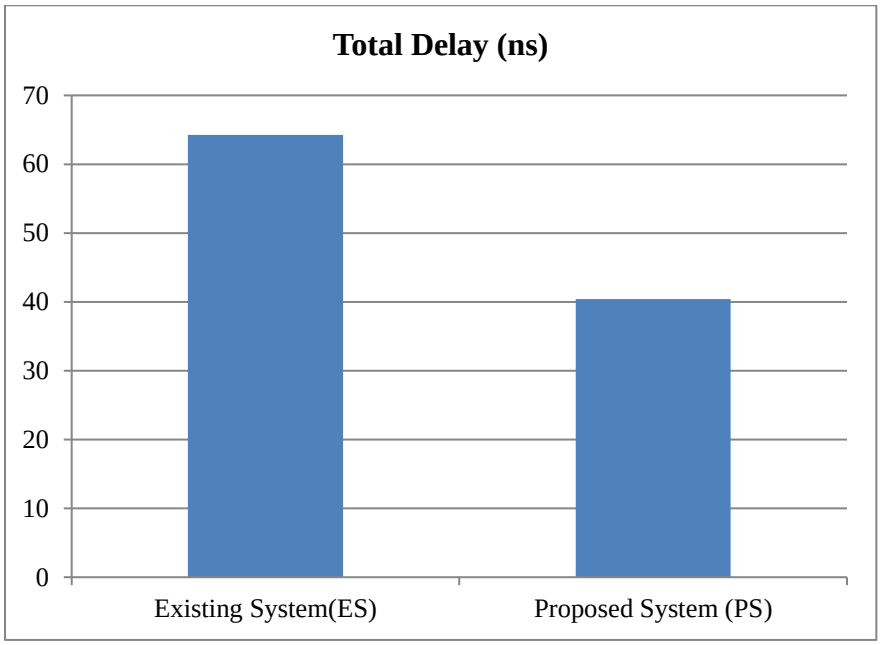

Figure 7. Total Delay Graph

In Figure 8, logic delay comparison graph is represented between existing system and proposed system is observed. The proposed system shows low logic delay when compared with existing system.

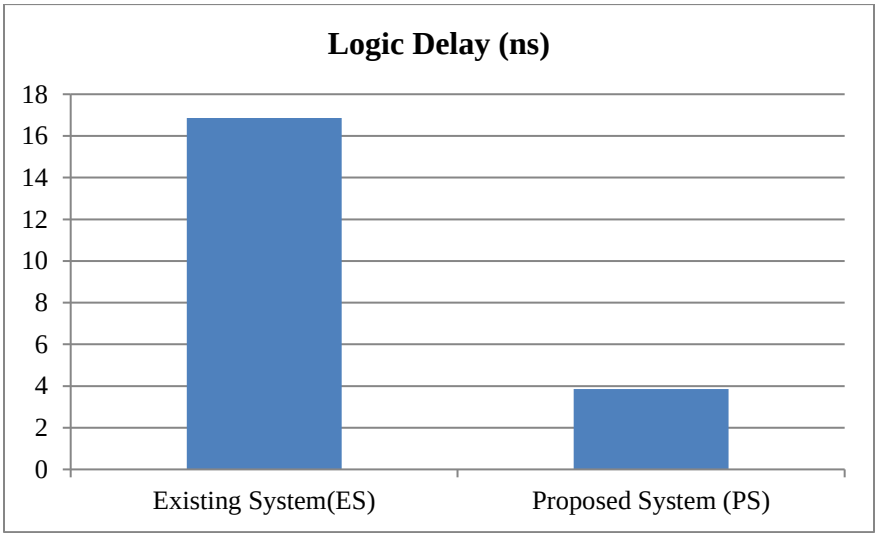

Figure 8. Logic Delay Graph

In Figure 9, route delay comparison graph is represented between existing system and proposed system is observed. The proposed system shows low route delay when compared with existing system.

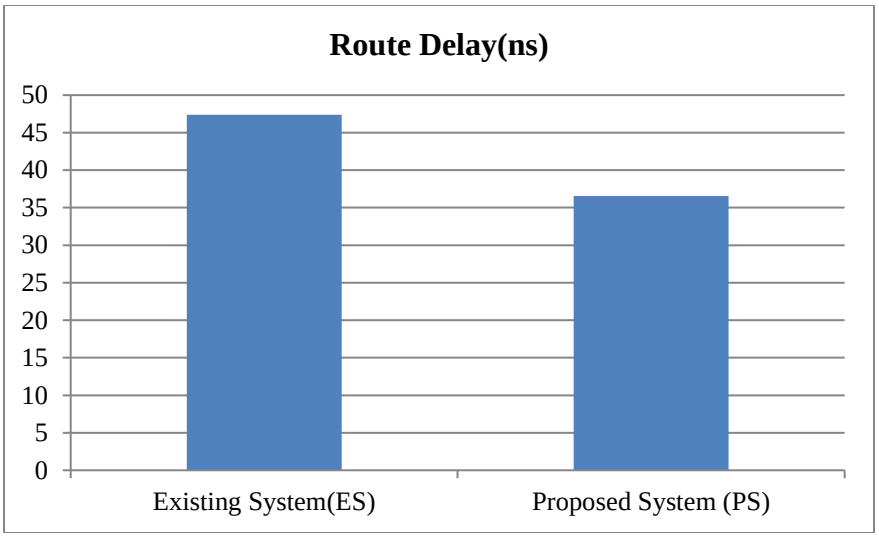

Figure 9. Route Delay Graph

In Figure 10, memory usage comparison graph is represented between existing system and proposed system is observed. The proposed system shows memory usage when compared with existing system.

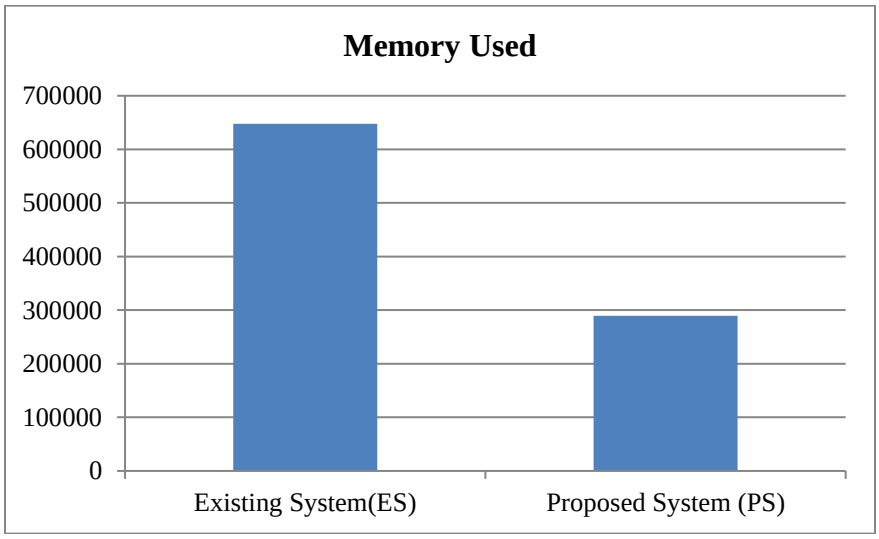

Figure 10. Memory Used Graph

V. CONCLUSION

As technology has advanced, numerous researchers have attempted—and continue to attempt—to create multipliers that provide either of the following design goals: high speed, low power consumption, or regularity of layout. They are therefore appropriate for a variety of high speed, low power, and compact VLSI implementations because they require less space or even a combination of them in a single multiplier. The fundamental building block of many DSP processors, image processing, and many others is multiplication. This study proposes a high speed and delay efficient adaptive hold logic based KS multiplier. This Kogge-Stone Multiplier uses the Kogge-Stone adder (a highly efficient parallel prefix adder) to rapidly calculate the sum of partial products. Its primary use is minimizing computation delay in high-performance digital

circuits where both speed and large-width operand processing are critical. The multiplier's intermediate operands employ Adaptive Hold Logic (AHL) to expedite the multiplication process. Through variable latency, the multiplier can increase throughput and modify the AHL circuit to counteract performance degradation brought on by the aging effect.Hence, delay and memory usage are reduced by using adaptive hold logic.

VI. REFERENCES

[1] D. J. M. Moss, D. Boland and P. H. W. Leong, "A Two-Speed, Radix-4, Serial–Parallel Multiplier," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 27, no. 4, pp. 769-777, April 2019, doi: 10.1109/TVLSI.2018.2883645.

[2] X. Cui, W. Liu, X. Chen, E. E. Swartzlander and F. Lombardi, "A Modified Partial Product Generator for Redundant Binary Multipliers," in IEEE Transactions on Computers, vol. 65, no. 4, pp. 1165-1171, 1 April 2016, doi: 10.1109/TC.2015.2441711.

[3] Aliparast, Peiman & daie koozehkanani, Ziaddin & Nazari, Farhad. (2013). An Ultra High Speed Digital 4-2 Compressor in 65-nm CMOS. International Journal of Computer Theory and Engineering. 593-597. 10.7763/IJCTE.2013.V5.756.

[4] Jongsu Park, San Kim and Yong-Surk Lee, "A low-power Booth multiplier using novel data partition method," Proceedings of 2004 IEEE Asia-Pacific Conference on Advanced System Integrated Circuits, Fukuoka, Japan, 2004, pp. 54-57, doi: 10.1109/APASIC.2004.1349403.

[5] A. Dalloo, A. Najafi and A. Garcia-Ortiz, "Systematic Design of an Approximate Adder: The Optimized Lower Part Constant-OR Adder," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 26, no. 8, pp. 1595-1599, Aug. 2018, doi: 10.1109/TVLSI.2018.2822278.

[6] G. A. Gillani, M. A. Hanif, B. Verstoep, S. H. Gerez, M. Shafique and A. B. J. Kokkeler, "MACISH: Designing Approximate MAC Accelerators With Internal-Self-Healing," in IEEE Access, vol. 7, pp. 77142-77160, 2019, doi: 10.1109/ACCESS.2019.2920335.

[7] Q. Xu, T. Mytkowicz and N. S. Kim, "Approximate Computing: A Survey," in IEEE Design & Test, vol. 33, no. 1, pp. 8-22, Feb. 2016, doi: 10.1109/MDAT.2015.2505723

[8] E. Nogues, D. Menard and M. Pelcat, "Algorithmic-Level Approximate Computing Applied to Energy Efficient HEVC Decoding," in IEEE Transactions on Emerging Topics in Computing, vol. 7, no. 1, pp. 5-17, 1 Jan.-March 2019, doi: 10.1109/TETC.2016.2593644

[9] G. A. Gillani, M. A. Hanif, M. Krone, S. H. Gerez, M. Shafique and A. B. J. Kokkeler, "SquASH: Approximate Square-Accumulate With Self-Healing," in IEEE Access, vol. 6, pp. 49112-49128, 2018, doi: 10.1109/ACCESS.2018.2868036.

[10] Y. Rasheed et al., "AMPEREISH: Approximate Multipliers for Power Efficiency in FPGA Designs Using Internal-Self-Healing," in IEEE Access, vol. 14, pp. 60252-60267, 2026, doi: 10.1109/ACCESS.2026.3684720.

[11] S. Mazahir, O. Hasan and M. Shafique, "Adaptive Approximate Computing in Arithmetic Datapaths," in IEEE Design & Test, vol. 35, no. 4, pp. 65-74, Aug. 2018, doi: 10.1109/MDAT.2017.2772874.

[12] A. Agrawal et al., "Approximate computing: Challenges and opportunities," 2016 IEEE International Conference on Rebooting Computing (ICRC), San Diego, CA, USA, 2016, pp. 1-8, doi: 10.1109/ICRC.2016.7738674.

[13] L. Eeckhout, "Approximate Computing, Intelligent Computing," in IEEE Micro, vol. 38, no. 4, pp. 6-7, Jul./Aug. 2018, doi: 10.1109/MM.2018.043191119.

[14] M. Horowitz, "1.1 Computing's energy problem (and what we can do about it)," 2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC), San Francisco, CA, USA, 2014, pp. 10-14, doi: 10.1109/ISSCC.2014.6757323.
[15] S. V. S. Prasad, N. S. S. Sagar, M. V. Nitya Pushkala, S. Silveri, B. Harshitha and S. Chenna, "Enhancing VLSI Performance: An Innovative Approach to 3-Parallel Polyphase Odd-Length FIR Filtering with Kogge-Stone Adder and Booth Multiplier," 2024 International Conference on Electrical Electronics and Computing Technologies (ICEECT), Greater Noida, India, 2024, pp. 1-5, doi: 10.1109/ICEECT61758.2024.10739232.
[16] Y. K, G. R, S. S, T. R. T and V. T, "Design and Simulation of 16x16 Vedic Multiplier using Kogge-Stone Adder," 2023 7th International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2023, pp. 452-457, doi: 10.1109/ICCMC56507.2023.10083594.
[17] S. Nikhil and P. V. Lakshmi, "Implementation of a high speed multiplier desired for high-performance applications using kogge stone adder," 2016 International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 2016, pp. 1-4, doi: 10.1109/INVENTIVE.2016.7823244.
[18] G. V. Nikhil, B. P. Vaibhav, V. G. Naik and B. S. Premananda, "Design of low power barrel shifter and vedic multiplier with kogge-stone adder using reversible logic gates," 2017 International Conference on Communication and Signal Processing (ICCSP), Chennai, India, 2017, pp. 1690-1694, doi: 10.1109/ICCSP.2017.8286680.
[19] S. Malagar, D. P and R. S, "Implementation of a 16-bit Multiplier Leveraging the Kogge-Stone Adder With Memristor IMPLY Logic," 2024 International Conference on Distributed Systems, Computer Networks and Cybersecurity (ICDSCNC), Bengaluru, India, 2024, pp. 1-5, doi: 10.1109/ICDSCNC62492.2024.10939928.
[20] G. Rohith, P. Jagadeesh and N. Meenakshisundaram, "Implementation of 8-bit Dadda Multiplier by Using 4:2 Compressors and Brentkung Adder for Reducing Propagation Delay in Comparison with Kogge Stone Adder," 2024 Ninth International Conference on Science Technology Engineering and Mathematics (ICONSTEM), Chennai, India, 2024, pp. 1-6, doi: 10.1109/ICONSTEM60960.2024.10568807.
[21] K. Sathish and P. Jagadeesh, "An Efficient Design and Performance Analysis of Novel 8 Bit Modified Wallace Multiplier Using Kogge-Stone Adder (KSA) in Comparison with Ripple carry adder (RCA)," 2023 Eighth International Conference on Science Technology Engineering and Mathematics (ICONSTEM), Chennai, India, 2023, pp. 1-7, doi: 10.1109/ICONSTEM56934.2023.10142406.
[22] A. Sundhar, S. D. Tharshini, G. Priyanka, S. Ragul and C. Saranya, "Performance Analysis of Wallace Tree Multiplier with Kogge Stone Adder using 15-4 Compressor," 2019 International Conference on Communication and Signal Processing (ICCSP), Chennai, India, 2019, pp. 0903-0907, doi: 10.1109/ICCSP.2019.8697981.
[23] A. Raju and S. K. Sa, "Design and performance analysis of multipliers using Kogge Stone Adder," 2017 3rd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), Tumkur, India, 2017, pp. 94-99, doi: 10.1109/ICATCCT.2017.8389113.
[24] S. Shirahatti, D. Haritha, P. Thejaswini, A. J. Deepika and B. M. Yashaswini, "Performance Analysis of Vedic Multiplier using Kogge Stone Adder and Reversible Logic," 2024 International Conference on Recent Advances in Science and Engineering Technology

(ICRASET), B G Nagara,Mandya, India, 2024, pp. 1-5, doi: 10.1109/ICRASET63057.2024.10895585.
[25] K. K R, S. Vijayalakshmi, B. M. Babu, P. K and N. S, "High-Performance Wallace Tree Multiplier with Kogge-Stone Adder," 2024 International Conference on Sustainable Communication Networks and Application (ICSCNA), Theni, India, 2024, pp. 1200-1205, doi: 10.1109/ICSCNA63714.2024.10864344.