Performance Analysis of Dadda Multiplier Using Kogge Stone Adder

B. Srikanth(✉), Dodda Sai Pranathi, Padmaraju Sai Kumar Raju, and Vemula Sarika

Department of ECE, Vardhaman College of Engineering, Rangareddy, Telangana, India
srikanth.vlsi.2011@gmail.com

Abstract. In many computer systems, the need for effective and high-performance multiplication processes has been increasing quickly. This work gives a thorough performance analysis of a Kogge-Stone adder-based 16-bit Dadda Multiplier. The goal is to assess the multiplier designs computational effectiveness and speed while taking things like power usage and space utilization into account. The performance analysis findings show that the 16-bit Dadda Multiplier, which uses the Kogge-Stone adder architecture, is superior to other multiplier designs. The experimental results show important improvements in multiplication speed due to a reduction in the critical path delay. This work is done with 90 nm technology. By utilizing the parallelism provided by the Kogge-Stone adder, the suggested architecture also demonstrates considerable improvements in power efficiency. A more compact solution is also suggested by the area utilization study, allowing for easier integration into integrated circuits. The findings of this study aid in the selection and optimization of multiplier designs in various computing applications as well as the body of information on effective multiplication methods. The results provided in this paper demonstrate the benefits of combining the 16-bit Dadda Multiplier with the Kogge-Stone adder and show how it can improve computing performance while taking power and space limits into account. In this the total power of the system reduced by 4.8% and the total area also reduced by 8.3%.

Keywords: Dadda Multiplication (DM) · Kogge-Stone Adder (KSA) · Power-Area constraints · Computational performance

1 Introduction

A crucial component of digital systems such arithmetic and logic units, digital signal processors, etc. are multipliers. Indicators of system performance like power, latency, and space utilization are frequently triggered by them [1]. Due to this rising demand, the multipliers performance needs to be enhanced. Three stages make up the multiplier: partial product development, partial product reduction, and adding at the end. More time and energy are used in the multiplier second stage. Different methods to reduce multiplier critical stages were proposed. The compressor is most frequently used during the reduction step of partial product. A simple adder circuit serves as the compressor. It adds a number of bits that are all equally weighted and then generates some sum signals.

X. Cheng (Ed): BROADNETS 2024, LNICST 602, pp. 28–41, 2025.
https://doi.org/10.1007/978-3-031-81171-5_3

A multipliers performance is often assessed using parameters like speed, power consumption, and area utilization. Using a Kogge-Stone adder, addition is completed at its conclusion. The 16 × 16 multipliers structure is depicted in Fig. 1. In comparison to multipliers with compressors employing various adders, such as parallel adders, simulation results demonstrate that the approximate multiplier with Kogge Stone Adder provides great performance [2]. The sections below provide more detail on this essay. In next section, designs for a roughly 16 × 16 Dadda Multiplier are described. The objective is to assess the multiplier design computational effectiveness and speed while taking things like power usage and space utilization into account. The outcomes of this research will offer insightful information on the efficiency of the suggested multiplier design and its potential to improve computing performance.

2 Dadda Multiplier

Dadda Multiplier is efficient hardware-based multiplication algorithm used in digital circuits and computer architecture to perform binary multiplication operations [3]. It is particularly well-suited for applications where high-speed multiplication is crucial, such as in modern microprocessors, digital signal processors (DSPs), and other computational devices. The Dadda Multiplier is known for its ability to minimize both the number of partial products generated during multiplication and the number of adder stages required to obtain the final product. This reduction in partial products and adder stages leads to significant improvements in both speed and power efficiency, making it an attractive choice for hardware designers aiming to optimize the performance of their digital systems.

The Dadda Multiplier is a powerful and efficient multiplication algorithm used in digital circuits to accelerate binary multiplication operations [4]. Its ability to minimize partial products and adder stages makes it a valuable tool for enhancing the speed and power efficiency of various computational devices, from microprocessors to specialized hardware accelerators. The Dadda Multiplier is known for its ability to minimize both the number of partial products generated during multiplication and the number of adder stages required to obtain the final product [5]. This reduction in partial products and adder stages leads to significant improvements in both speed and power efficiency, making it an attractive choice for hardware designers aiming to optimize the performance of their digital systems.

At its core, the Dadda Multiplier employs a tree-based structure to perform multiplication. The algorithm takes advantage of the fact that each bit in the binary representation of a number can be thought of as a coefficient in a polynomial. By using a clever grouping and addition strategy, the Dadda Multiplier [6] efficiently calculates the product of two binary numbers by minimizing the number of additions required.

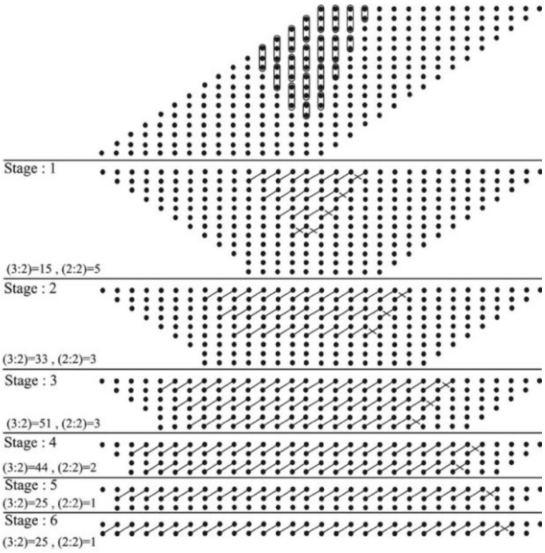

Fig. 1. Structure of 16 × 16 Dadda Multiplier

3 Kogge Stone Adder

The Kogge-Stone Adder, often referred to as the Kogge- Stone Parallel Prefix Adder, is a high-speed and highly parallelized hardware circuit used for binary addition operations in digital systems [7]. It named after its inventors, Peter Kogge and Harold Stone, who introduced the concept in the early 1970s. This adder architecture is known for its efficient and scalable design, making it popular in applications where fast arithmetic operations are essential, such as in microprocessors, digital signal processors (DSPs), and custom integrated circuits (ICs). The key features and principles behind the Kogge-Stone Adder are Parallel Prefix Structure, Tree-Based Architecture, Scalability, Efficient Carry Propagation, Trade-Offs.

The Kogge-Stone Adder employs a parallel prefix structure, which means that it breaks down the addition process into multiple stages that can be performed simultaneously [8]. This parallelism is achieved by computing intermediate carry values at each stage in parallel, significantly reducing the propagation delay and improving overall speed. The adder structure resembles a binary tree, where each node in the tree represents a stage of the addition process. At each node, partial carry values are calculated based on the input bits and the carry from the previous stage. The carry propagation occurs simultaneously at multiple levels of the tree. One of the notable advantages of the Kogge-Stone Adder is its scalability [9]. You can easily expand the size of the adder to accommodate larger binary numbers by adding more stages to the tree structure. This scalability allows for efficient implementation of multi-bit adders with minimal increase in delay.

In traditional ripple-carry adders, carry bits ripple through each bit position, leading to a linear delay. The Kogge-Stone Adder, on the other hand, significantly reduces the

carry propagation delay by allowing carry values to propagate in parallel through multiple paths in the tree structure. While the Kogge-Stone Adder is known for its speed and parallelism, it may require more hardware resources compared to simpler adder designs like the ripple-carry adder. Therefore, designers often choose it when optimizing for speed and are willing to allocate additional hardware resources.

Essentially, it is a parallel prefix adder. Based on design time, this type of adder excels at providing the quickest addition [10]. It is renowned for its unique addition, which is the fastest in terms of design time. In Figs. 2 and 3 (from base paper), the functional block diagram and RTL view of a 4-bit Kogge-Stone Adder is shown. The propagate signal "Pi" and the generate signal "Gi" are calculated using the ith bit of the supplied input. Similar to how they generate signals, they also produce and carry signals [11]. Therefore, Prefix Adders are primarily divided into three kinds by minimizing the computation delay.

1.1. Pre- processing
1.2. Generation of Carry
1.3. Final processing

3.1 Pre-processing

This is the first stage; in this two signals are generated. They are generate signal and propagate signals. The expressions for generate signal and propagate signals are 1 and 2.

$$P_i = A_i \oplus B_i \tag{1}$$

$$G_i = A_i . B_i \tag{2}$$

3.2 Generation of Carry

This is the second stage. Carriers are now calculated using the appropriate bits, and this process is carried out in parallel. As intermediate signals, carry propagation and generation are employed. Carry, propagate, and generate signals have the expressions 3 and 4.

$$G_i = \left(P_i . G_{i\,prev}\right) + G_i \tag{3}$$

$$P_i = \left(P_i . P_{i\,prev}\right) \tag{4}$$

3.3 Final Processing

These is the final processing stage, in this the sum and carry outputs bits are calculated based on the input bits that are provided, and the logic equation for the last processing step is provided by 5 and 6.

$$C_i = G_i \tag{5}$$

$$S_i = P_i \oplus C_{i-1} \tag{6}$$

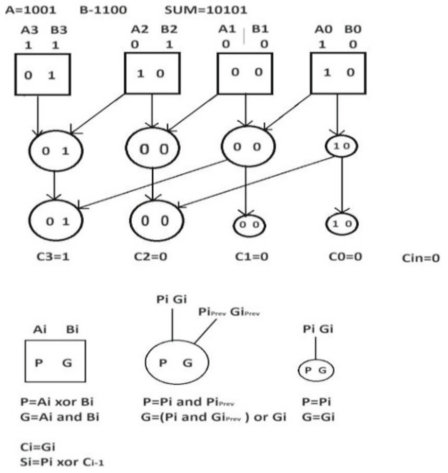

Fig. 2. Block diagram of Kogge Stone Adder

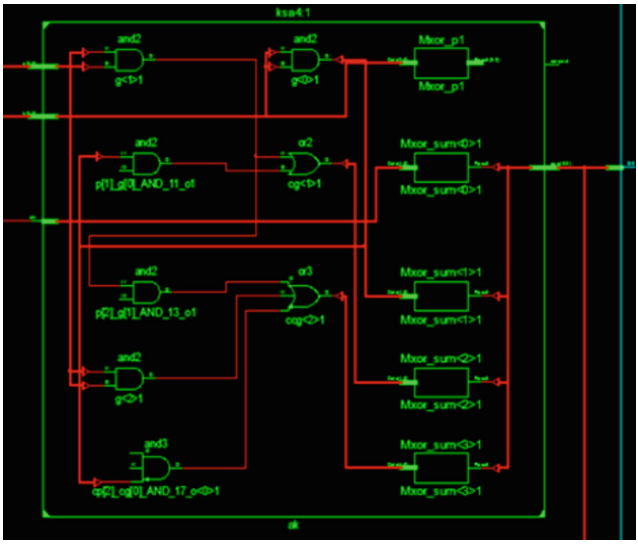

Fig. 3. RTL view of Kogge-Stone Adder

4 Proposed Work

The working of a Dadda Multiplier using Kogge Stone Adders can be summarized as follows:

The operands are represented as binary numbers. The partial products are generated by multiplying the corresponding bits of the operands. The partial products are added together using Kogge Stone Adders. The final product is generated by adding together the outputs of the Kogge Stone Adders. The speed of a Dadda Multiplier using Kogge

Stone Adders is determined by the number of gates required and the type of adder used. Kogge Stone Adders are typically faster than other types of adders, such as carry-save adders, because they can perform addition in a single clock cycle. As a result, Dadda Multipliers using Kogge Stone Adders are typically faster than other types of multipliers [12]. The area of a Dadda Multiplier using Kogge Stone Adders is determined by the number of gates required and the size of the adders.

Kogge Stone Adders are typically larger than other types of adders, but they can be implemented in a variety of ways to reduce area. As a result, the area of a Dadda Multiplier using Kogge Stone Adders can be comparable to or even smaller than the area of other types of multipliers. The power consumption of a Dadda Multiplier using Kogge Stone Adders is determined by the number of gates required and the type of adder used. Kogge Stone Adders are typically more power-efficient than other types of adders, such as carry-save adders, because they do not require as many gates. As a result, Dadda Multipliers using Kogge Stone Adders are typically more power efficient than other types of multipliers.

In general, Dadda Multipliers using Kogge Stone Adders offer a good trade-off between speed, area, and power consumption. They are typically faster than other types of multipliers, but they can also be implemented in a variety of ways to reduce area and power consumption. Some additional considerations for the working of Dadda Multipliers using Kogge Stone Adders are the size of the operands, implementation of the multiplier, the size of the operands affects the number of gates required and the speed of the multiplier. Larger operands require more gates and are slower than smaller operands. There are different types of Kogge Stone Adders, and the type of adder used can affect the speed, area, and power consumption of the multiplier. The multiplier can be implemented in a variety of ways, and the implementation can affect the speed, area, and power consumption of the multiplier [13]. Overall, the working of a Dadda Multiplier using Kogge Stone Adders can be optimized by considering the size of the operands, the type of Kogge Stone Adder, and the implementation of the multiplier (Fig. 4).

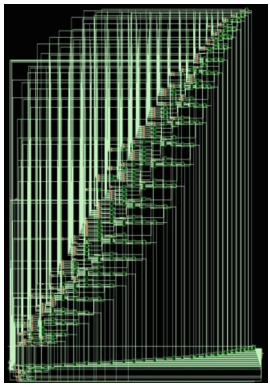

Fig. 4. Schematic View of Dadda Multiplier using Kogge Stone Adder

5 Result Analysis

While both Full Adders and Kogge-Stone adders serve the purpose of adding binary numbers, their design, speed, efficiency, and applications differ significantly. Full Adders are simple and suitable for small-scale applications, whereas Kogge-Stone adders are complex but excel in high-speed, large-scale addition operations. A Full Adder and a Kogge-Stone adder are both digital circuits used in digital logic design and computer architecture for performing addition operations. However, they differ in terms of their structure, performance characteristics, and applications. The choice between them depends on the specific requirements of the digital system being designed. They are Structure, Speed and Efficiency, Area and Complexity, Scalability, A Full Adder is a basic building block for addition operations. It takes three input bits: A, B, and a carry input (Cin), and produces two outputs, Sum (S) and carry output (Cout). The Full Adder can be used to add two binary numbers bit by bit. The Kogge-Stone Adder is a more complex structure compared to the Full Adder. It is a parallel prefix adder that uses a tree-like structure to compute the sum of binary numbers more efficiently. It consists of multiple Full Adders interconnected in a specific way.

Full Adders are used for small-scale addition operations, and they are relatively slower compared to parallel adders like the Kogge-Stone adder. For large binary numbers, multiple Full Adders need to be cascaded, which can result in slower operation. The Kogge-Stone Adder is designed for high- speed addition of large binary numbers. It takes advantage of parallelism by computing multiple bits of the sum simultaneously, making it much faster for large inputs compared to cascaded Full Adders [14]. Full Adders have a relatively simple and compact structure, making them suitable for low-complexity applications. Kogge-Stone Adders are more complex and require more hardware resources due to their parallel nature. They are typically used in high-performance computing applications where speed is a priority. Full Adders can be easily cascaded to create multi-bit adders, but this approach becomes less efficient as the number of bits increases. Kogge-Stone Adders are designed for efficient scalability, making them well-suited for adding large binary numbers.

Full Adders are commonly used in small-scale applications, such as arithmetic units of microcontrollers and simple digital systems. Kogge-Stone Adders are used in high-performance computing systems, such as CPUs and GPUs, where the speed of addition operations is crucial for overall system performance. The four instances of Full Adder are replaced with a single Kogge Stone Adder instance. In this process the area is reduced

Table 1. Power analysis of Full-Adder Vs Kogge Stone Adder

Instances	Leakage Power (uW)	Dynamic Power (uW)	Total Power (uW)	Percentage improvement
Full Adder	1.180	112.069	113.249	-
Kogge Stone Adder	0.753	82.889	82.959	26.74%

by 24.6% and the total power is reduced by 26.74% when compared between Full Adder and Kogge Stone Adder (Figs. 5 and 6 and Tables 1 and 2).

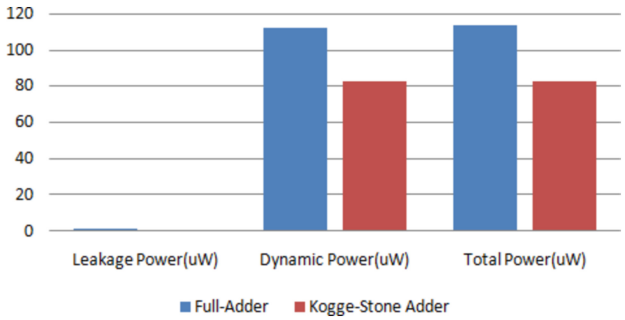

Fig. 5. Power dissipation analysis of Kogge Stone Adder Vs Full Adder

Table 2. Area analysis of Full-Adder Vs Kogge Stone Adder

Instances	Cell Area	Total Area	Percentage improvement
Full Adder	205.876	205.876	-
Kogge Stone Adder	155.165	155.165	24.6%

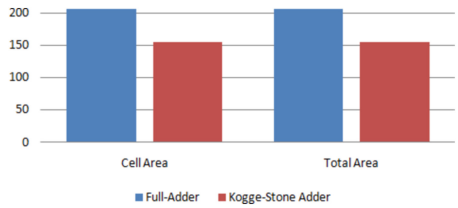

Fig. 6. Area analysis of Kogge Stone Adder Vs Full Adder

The different reports generated are the timing report, power report, area report. The tool used to generate all these timing report, area report, power report is the Genus Tool.

Genus is a popular RTL synthesis tool used in digital circuit design to convert RTL (Register Transfer Level) code into gate-level net-lists. When you run the synthesis process using Genus, it generates a timing report that provides essential information about the performance of the synthesized design. The timing report helps you understand the timing characteristics of our design and identify potential issues. The key components of a timing report in Genus:

In terms of critical path, the timing report highlights the critical path in your design. The critical path is the longest path through the combinational logic that determines the maximum delay for the design. The timing analyzer in Genus identifies the gates and

signals involved in this critical path, and their cumulative delay is reported as the critical path delay. In terms of critical path slack, Slack represents the timing margin available on the critical path. It is calculated as the difference between the required time (usually based on the target clock frequency) and the actual time taken by the critical path. When there is positive slack, the design complies with timing specifications; when there is negative slack, there is a timing infraction.

In terms of setup and hold time violations, the timing report also indicates any setup and hold time violations. Hold time is the minimum amount of time that data must remain stable after the clock edge, and setup time is the minimum amount of time that data must remain stable before the clock edge. Violations of these constraints can lead to data integrity issues. In terms of clock frequency, the timing report provides information about the achieved clock frequency after synthesis. This frequency is based on the critical path delay and can give you an idea of the design performance in terms of speed. In terms of timing paths with violations, it will highlight the specific reasons for the violation, such as excessive delay or setup/hold violations. In terms of input and output delay, the timing report may include information about the input and output delays for various cells and interfaces in the design. Input delay is the time taken for the input signal to propagate through the logic to the output, while output delay is the time taken for the output signal to stabilize after changes in inputs.

In terms of Constraint Information, the timing report may also include details about the timing constraints applied during synthesis, such as clock frequency targets, setup/hold constraints, and input/output delays (Fig. 7 and Table 3).

Table 3. Timing Report of Dadda Multiplier without Kogge-Stone Adder Vs Dadda Multiplier with Kogge-Stone Adder

Instances	Required Time (ps)	Input Delay (ps)	Data Path (ps)	Slack (ps)
DM without KSA	1200	800	134	266
DM with KSA	1805	800	937	68

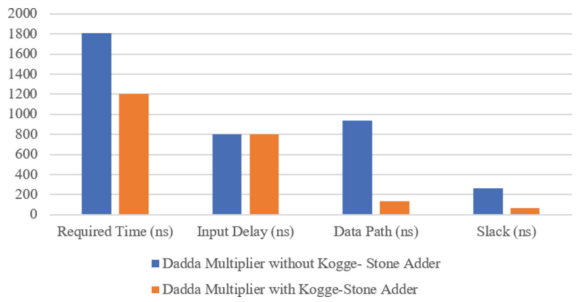

Fig. 7. Timing analysis of DM without KSA Vs DM with KSA

In Genus, the power report provides essential information about the power consumption of the synthesized digital circuit. Power analysis is crucial in modern integrated circuits design as power consumption impacts the overall performance, battery life (in case of portable devices), and heat dissipation of the chip. The power report generated by Genus helps designers understand the power characteristics of the design and identify areas where power optimization may be needed. The key components typically found in a power report from Genus:

Using power report, the total power can be calculated. The power report provides the total power consumption of the design in watts or milli watts. This is the sum of all the power consumed by the various components in the chip, including combinational logic, registers, memories, and other elements.

The power report, the dynamic power is calculated. Dynamic power is the power dissipated due to the charging and discharging of capacitive loads as a result of switching activities in the circuit. It is the dominant component of power consumption in most digital designs. The power report shows the dynamic power along with a breakdown of its components, such as switching power and short- circuits power. The power report provides the leakage power. Leakage power is the power consumed by a transistor even when it is not actively switching. As transistor sizes decrease and leakage currents become more significant, leakage power becomes a significant portion of total power consumption. The power report provides the leakage power along with a breakdown of its components.

Using the power report the internal power can be calculated. Internal power represents the power consumed within the core logic of the chip, excluding the I/O pads and interfaces. It includes power consumption in registers, combinational logic, and other internal elements (Fig. 8 and Table 4).

Table 4. Power Report of DM without KSA Vs DM with KSA

Instances	Leakage Power (uW)	Dynamic Power (uW)	Total Power (uW)	Percentage improvement
DM without KSA	75.224	8791.656	8866.88	-
DM with KSA	68.904	8364.326	8433.23	4.8%

In Genus, the area report provides information about the physical size or area of the synthesized digital circuit. Area analysis is crucial in integrated circuit design as it directly impacts the chip manufacturing cost, performance, and overall feasibility. The area report generated by Genus helps designers understand the physical characteristics of the design and identify areas where area optimization may be needed.

The key components typically found in an area report from Genus:

The area report provides the total area of the design in terms of square micrometers or square millimeters (mm^2). This is the cumulative size of all the components in the chip, including standard cells, memories, I/O pads, and other elements. The area report may provide a breakdown of the total area into different categories, such as core area, I/O area, and any other user-defined areas. The core area typically refers to the area

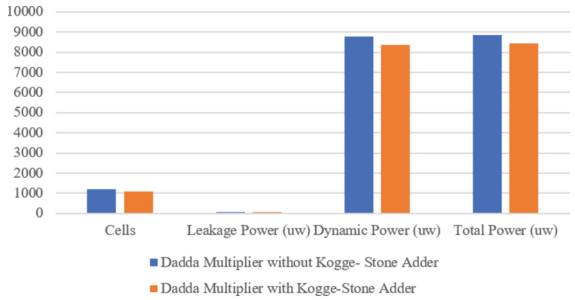

Fig. 8. Power Dissipation analysis of DM without KSA Vs DM with KSA

occupied by the main functional logic of the chip, while the I/O area includes the space taken by input/output pads and interfaces. The area report provides the overall cell area. Genus can provide details about the individual area of each standard cell used in the design. This information helps designers identify cells with large footprints and focus on optimizing them.

Using the area report, the hierarchical area analysis is performed. Genus can perform hierarchical area analysis, breaking down the area consumption into different blocks or modules in the design. This breakdown helps identify area- intensive blocks and facilitates area optimization at the block level. Using the area report, the utilization is generated. The area report may include utilization metrics, such as the percentage of core area used and the percentage of unused or "white space" in the chip. Low utilization may indicate inefficient use of resources, while high utilization could lead to potential congestion issues. The aspect ratio can also be calculated using the area report. The aspect ratio is the ratio of the chip height to its width. A balanced aspect ratio is often desirable to ensure efficient chip manufacturing and layout.

By analyzing the area report, designers can gain insights into the physical size of the design, identify potential area bottlenecks, and implement area optimization strategies to reduce the chip footprint and manufacturing costs. Properly optimizing area and power requirements is crucial for designing efficient and cost-effective integrated circuits (Fig. 9 and Table 5).

Table 5. Area Report of DM without KSA Vs DM with KSA

Instances	Cell Count	Cell Area	Total Area	Percentage improvement
DM without KSA	1104	14247.885	1424.885	-
DM with KSA	1193	13052.740	1305.740	8.3%

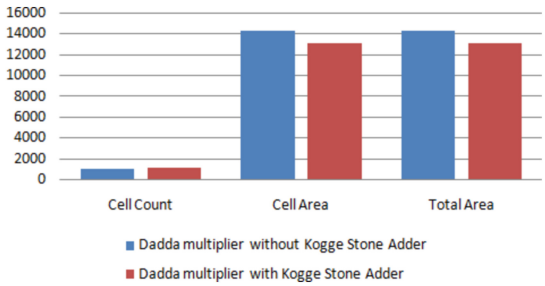

Fig. 9. Area Analysis of DM without KSA Vs DM with KSA

6 Simulation Results

In this output of the Dadda Multiplier we get the clock latency. For one particular test case we get the 10 clock latency. So, due to this clock latency the output is delayed for 10 clock cycles of the CLK input.

Clock latency, also known as clock cycle time or clock period, refers to the time duration between successive rising edges or falling edges of a clock signal in a digital system. It is a crucial parameter in digital electronics and computer architecture, as it directly affects the performance and speed of the system. In synchronous digital systems, such as CPUs, GPUs, and other digital logic circuits, operations are synchronized and controlled by a clock signal.

The clock signal is used to regulate the timing of various operations and to synchronize the transfer of data between different components within the system. The clock latency is determined by the inverse of the clock frequency. If the clock frequency is denoted as f (measured in Hertz, Hz). A lower clock latency means that the clock signal rising edges occur more frequently, allowing the digital system to perform operations faster. However, decreasing the clock latency often requires sophisticated and more expensive circuitry, as it poses challenges related to signal integrity, power consumption, and thermal management (Fig. 10).

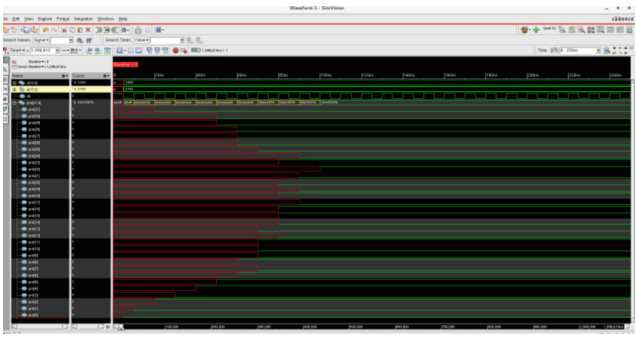

Fig. 10. Simulation Result

7 Conclusion

The efficiency of a Dadda Multiplier using Kogge Stone Adders can be analyzed in terms of speed, area, and power usage. Dadda Multipliers using Kogge Stone Adders are typically faster than Dadda Multipliers without using Kogge Stone Adder. The area of a Dadda Multiplier using Kogge Stone Adders is even smaller. Dadda Multipliers using Kogge Stone Adders are typically more power- efficient. In this work the clear view of the comparisons between Full Adder and the Kogge Stone Adder is demonstrated in terms of their usage in Dadda Multiplier.

Based on the results, the area is reduced by 24.6% and the power is less by 26.74%. The type of Kogge Stone Adder used can also affect the performance of the multiplier. Overall, the performance of a Dadda Multiplier using Kogge Stone Adders can be optimized in terms of area by 8.3% and in terms of power by 4.8%.

References

1. Pathak, K.C., Darji, A.D., Sarvaiya, J.N.: Low power Dadda Multiplier using approximate almost full adder and Majority logic based adder compressors. IEEE Region 10 Symposium 2022 (TENSYMP), pp. 1–6. Mumbai, India (2022). https://doi.org/10.1109/TENSYMP54529.2022.9864428
2. Naresh, K., Sai, Y.P., Majumdar, S.: Design of 8-bit Dadda multiplier using gate level approximate 4:2 compressor. 35th International Conference on VLSI Design and 2022 21st International Conference on Embedded Systems (VLSID), pp. 269–274. Bangalore, India (2022). https://doi.org/10.1109/VLSID2022.2022.00059
3. Chanda, S., et al.: An energy efficient 32 bit approximate dadda multiplier. IEEE Calcutta Conference (CALCON), pp. 162–165. Kolkata, India (2020). https://doi.org/10.1109/CALCON49167.2020.9106548
4. Sundhar, A., Deva Tharshini, S., Priyanka, G., Ragul, S., Saranya, C.: Performance analysis of wallace tree multiplier with kogge stone adder using 15–4 compressor. International Conference on Communication and Signal Processing (ICCSP), pp. 0903–0907 (2019). https://doi.org/10.1109/ICCSP.2019.8697981
5. Riaz, M.H., Ahmed, S.A., Javaid, Q., Kamal, T: Low power 4×4 bit multiplier design using Dadda algorithm and optimized Full Adder. 15th International Bhurban Conference on Applied Sciences and Technology (IBCAST), pp. 392–396 (2018). https://doi.org/10.1109/IBCAST.2018.8312254
6. Ye, J.-H., Shieh, M.-D.: High-performance NTT architecture for large integer multiplication. 2018 International Symposium on VLSI Design, Automation and Test (VLSI-DAT), IEEE
7. Sinthura, S.S., Begum, A., Amala, B., Vimala, A., Vidhya Aparna, V.: Implementation and analysis of different 32-bit multipliers of aspects of power. Speed and Area. 2nd International Conference on Trends in Electronics and Informatics (ICOEI), pp. 312–317 (2018). https://doi.org/10.1109/ICOEI.2018.8553859
8. Madhavi, S., Rasagna, K., Kavya, N., Sindhu, M., Kaveri, V.: Implementation of programmable FIR filter using dadda multiplier and parallel prefix adder. IEEE 2018 International Conference on Inventive Research in Computing Applications (ICIRCA). https://doi.org/10.1109/ICIRCA.2018.8597249
9. James, R.M., Ravindran, A.: Review of full adder performance analysis using kogge stone adder and magnetic tunnel junction. 4th International Conference on Devices, Circuits and Systems (ICDCS), pp. 84–90. Coimbatore, India (2018). https://doi.org/10.1109/ICDCSyst.2018.8605064

10. Sebastian, A., Jose, F., Gopakumar, K., Thiyagarajan, P.: Design and implementation of an efficient dadda multiplier using novel compressors and fast adder. IEEE 2020 International Symposium on Devices, Circuits and Systems (ISDCS). https://doi.org/10.1109/ISDCS49393.2020.9263014
11. Penchalaiah, U., Vg, S.K.: Design of high-speed and energy-efficient parallel prefix kogge stone adder. IEEE International Conference on System, Computation, Automation and Networking (ICSCA), pp. 1–7. Pondicherry, India (2018). https://doi.org/10.1109/ICSCAN.2018.8541143
12. Raju, A., Sa, S.K.: Design and performance analysis of multipliers using Kogge Stone Adder. 3rd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), pp. 94–99. Tumkur, India (2017). https://doi.org/10.1109/ICATCCT.2017.8389113
13. Ganesh, K., Pushpalatha, P.: Implementation of a high-speed pipelined FFT processor using dadda multipliers to process two independent data streams. IEEE 2017 6th International Conference on Reliability, Infocom Technologies and Optimization. https://doi.org/10.1109/ICRITO.2017.8342479
14. Abraham, S., Kaur, S., Singh, S.: Study of various high-speed multipliers. International Conference on Computer Communication and Informatics (ICCCI), pp. 1–5. Coimbatore, India (2015). https://doi.org/10.1109/ICCCI.2015.7218139