DESIGN AND SIMULATION OF SYSTOLIC ARRAY FOR MATRIX MULTIPLICATION

G. SRIYA, G. SRINATH, K. HARISH, K. MADHU

Dr. J. SUNITHA KUMARI

Associate Professor, Department of Electronics and Communication Engineering

TKR COLLEGE OF ENGINEERING AND TECHNOLOGY AUTONOMOUS
(Affiliated to JNTUH, Accredited By NBA, Accredited by NAAC with "A+" grade)
2025-2026

ABSTRACT

Traditional methods for matrix multiplication often rely on general matrix multiply (GeMM) approaches, where matrices are divided into blocks and processed sequentially or with limited parallelism. These methods face limitations in speed and resource utilization due to the complexity of managing memory and computations, leading to bottlenecks in latency and throughput. Standard implementations typically use multiply-accumulate units arranged in linear or tree structures, which do not fully exploit parallelism and result in slower processing for large matrices. The systolic array architecture overcomes these limitations by arranging processing elements (PEs) in a highly parallel two-dimensional grid, where each element performs multiply-accumulate operations concurrently with data flowing rhythmically between them. This design enables continuous and efficient data movement, minimizing memory access delays and increasing throughput. The modular nature of the array also provides scalability for different matrix sizes, improving performance without significant increases in complexity. Overall, the systolic array-based approach offers a high-speed, efficient, and scalable solution for matrix multiplication, making it suitable for real-time and computation-intensive applications.

1. INTRODUCTION

Matrix multiplication plays a crucial role in many fields such as machine learning, image processing, and scientific computing. As the complexity and size of data increase, the demand for faster and more efficient computation methods also grows. Traditional matrix multiplication techniques are not capable of meeting these performance requirements due to their sequential nature and limited parallelism.

In conventional methods, computations are carried out step-by-step, which leads to increased execution time and higher memory usage. These methods also involve frequent data transfers between memory and processing units, creating bottlenecks that reduce overall system performance.

To overcome these challenges, parallel computing architectures have been introduced. One of the most effective approaches is the systolic array architecture. A systolic array consists of a network of processing elements arranged in a grid-like structure. These elements work simultaneously to perform arithmetic operations while data flows between them in a synchronized manner.

The key advantage of systolic arrays is their ability to perform multiple computations in parallel, thereby significantly reducing execution time. Additionally, the continuous data flow minimizes memory access, improving efficiency and reducing latency.

This paper focuses on the design and simulation of a systolic array for matrix multiplication. The proposed system aims to provide a high-speed and scalable solution that can efficiently handle large matrix operations. By leveraging parallel processing and optimized data flow, the systolic array architecture enhances performance compared to traditional methods.

The simulation results validate the effectiveness of the proposed design, making it suitable for applications requiring high computational power and real-time processing capabilities.

2.LITERATURE SURVEY

Several research works have explored efficient methods for matrix multiplication and hardware acceleration. Traditional approaches such as GeMM have been widely used due to their simplicity and adaptability in software implementations. However, these methods are not efficient for large-scale computations due to limited parallelism and high memory access overhead.

Researchers have proposed various optimization techniques to improve performance, including block matrix multiplication and parallel processing using multi-core processors. While these methods offer some improvements, they still face challenges related to synchronization, memory bandwidth, and scalability.

The concept of systolic arrays was introduced as a solution to overcome these limitations. Early studies demonstrated that

systolic architectures can efficiently perform repetitive computations such as matrix multiplication by using a network of simple processing elements. These elements communicate locally, reducing the need for global memory access.

Recent advancements have focused on implementing systolic arrays using hardware platforms such as Field Programmable Gate Arrays (FPGAs) and Application-Specific Integrated Circuits (ASICs). These implementations have shown significant improvements in speed and energy efficiency compared to traditional CPU-based approaches.

Some studies have also integrated systolic arrays with machine learning accelerators, highlighting their effectiveness in deep learning applications. These designs utilize large-scale arrays to perform high-dimensional matrix operations efficiently.

Despite these advancements, challenges such as hardware complexity and resource utilization still exist. This paper aims to address these challenges by designing an optimized systolic array architecture and validating its performance through simulation.

3. SYSTEM ANALYSIS

3.1 Existing System

The existing system for matrix multiplication is primarily based on traditional methods such as GeMM. These approaches rely on sequential execution and limited parallel processing, which results in slower computation speed. The major drawbacks include high latency, inefficient memory usage, and limited scalability.

3.2 Proposed System

The proposed system utilizes a systolic array architecture for efficient matrix multiplication. It consists of multiple processing elements arranged in a two-dimensional grid. Each element performs multiplication and accumulation operations while passing data to neighboring elements.

This architecture enables parallel execution of operations, reducing computation time significantly. The continuous flow of data minimizes memory access, improving overall efficiency. The modular design allows scalability, making it suitable for different matrix sizes.

The proposed system aims to achieve high performance, low latency, and efficient

resource utilization compared to traditional approaches.

5.PROPOSED METHODOLOGY

The proposed methodology involves designing a systolic array architecture and simulating its performance for matrix multiplication.

Initially, the architecture of the systolic array is defined, including the number of processing elements and their interconnections. Each processing element is designed to perform multiply-accumulate operations.

Input matrices are fed into the array in a synchronized manner. Data flows horizontally and vertically across the processing elements, allowing simultaneous computation of partial results. These partial results are accumulated to produce the final output matrix.

The design is implemented using a hardware description language, and simulation is carried out using appropriate tools. Test cases are created to validate the correctness of the output.

Performance metrics such as latency, throughput, and resource utilization are analyzed. The results are compared with traditional methods to demonstrate improvements.

This methodology ensures efficient computation and validates the effectiveness of the systolic array architecture.

6. IMPLEMENTATION

The implementation of the systolic array for matrix multiplication is carried out using a hardware description language such as Verilog. The design consists of multiple processing elements arranged in a grid structure.

Each processing element is responsible for performing multiplication and accumulation operations. These elements are interconnected to allow data transfer between them. Input matrices are provided to the array through input ports, and the output matrix is collected from the output ports.

The design is simulated using tools like ModelSim or Xilinx Vivado. A testbench is created to provide input data and verify the output results. Simulation helps in analyzing the functionality and performance of the system.

The results show that the systolic array significantly improves computation speed compared to traditional methods. The

parallel processing capability and efficient data flow contribute to reduced latency and increased throughput.

7. RESULTS AND DISCUSSION

The simulation results demonstrate the effectiveness of the systolic array architecture for matrix multiplication. The system successfully computes the output matrix with high accuracy and efficiency.

Compared to traditional methods, the systolic array shows a significant reduction in computation time. The parallel processing of multiple elements allows faster execution of operations. Additionally, the continuous data flow reduces memory access, improving overall performance.

Performance metrics such as latency and throughput are analyzed. The results indicate that the systolic array achieves lower latency and higher throughput compared to conventional approaches.

The scalability of the architecture is also evaluated by increasing the size of the matrix. The results show that the system maintains efficiency even for larger matrices.

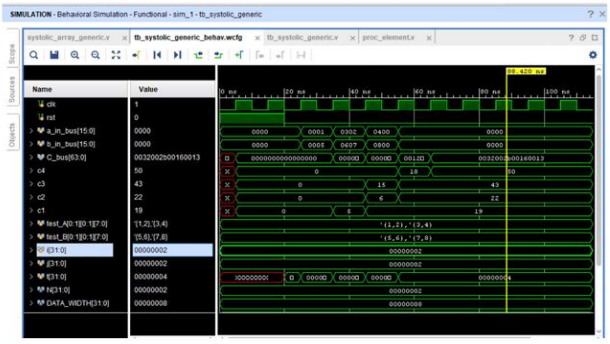

Fig 1: Waveform of Matrix Multiplication

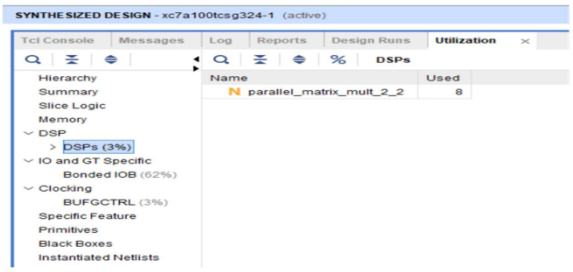

Fig 2: DSP Slices Utilization in conventional method

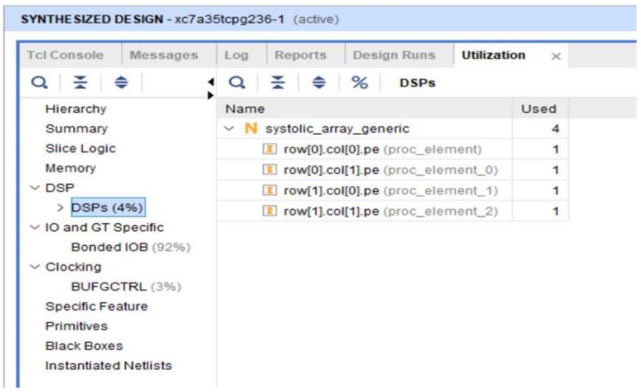

Fig 3: DSP Slices Utilization in Systolic Array Architecture

CONCLUSION

This project presents the design, implementation, and comprehensive

analysis of a systolic array architecture aimed at achieving efficient and high-speed matrix multiplication. The study begins with an understanding of conventional matrix multiplication techniques, which typically rely on sequential execution or limited parallelism. While such methods are straightforward and easy to implement, they suffer from significant drawbacks when handling large datasets. These limitations include increased computational time, higher latency, inefficient data reuse, and excessive dependency on memory access, all of which make conventional approaches less suitable for modern high performance applications. To address these challenges, the project proposes a systolic array-based architecture, which introduces a highly structured and parallel computing model. In this approach, multiple processing elements (PEs) are arranged in a regular two-dimensional grid, where each element performs a simple yet essential operation—multiply and accumulate (MAC). The key strength of this architecture lies in its coordinated data movement. Input data from matrix A flows horizontally across the array, while data from matrix B moves vertically. Partial results are accumulated locally within each processing element and propagated in a synchronized manner. This rhythmic flow of data between neighboring elements eliminates the need for repeated access to external memory, thereby improving efficiency. One of the most significant advantages demonstrated in this project is the ability of the systolic array to exploit parallelism and pipelining effectively. Unlike conventional architectures, where operations are executed step-by-step, the systolic array allows multiple computations to occur simultaneously. Once the pipeline is fully initialized, the system is capable of producing output results at every clock cycle. This leads to a substantial increase in throughput and a reduction in overall computation time. The pipelined execution ensures that hardware resources are continuously utilized, minimizing idle cycles and improving performance. The modular design of processing elements is another important contribution of this work. Each PE is designed with a simple structure consisting of input registers, a multiplier, an adder, and an accumulator. This simplicity enables easy replication of PEs across the array, making the architecture highly scalable. As a result, the system can be extended to handle larger matrix sizes without requiring significant changes in the core design. This scalability is particularly important in applications where computational requirements continue to grow. In addition to performance improvements, the project also emphasizes

efficient hardware resource utilization. The systolic array design demonstrates reduced reliance on DSP slices compared to conventional implementations. This is primarily due to effective data reuse and distributed computation across multiple processing elements. By minimizing redundant operations and optimizing data flow, the architecture achieves better utilization of available hardware resources. This makes it highly suitable for FPGA-based implementations and other resource-constrained environments. The simulation results obtained from the designed system validate both the functional correctness and stability of the architecture. The outputs generated by the systolic array match the expected results of matrix multiplication, confirming the accuracy of the design. Furthermore, waveform analysis shows proper synchronization between input data, intermediate computations, and final outputs. The system operates without timing issues, glitches, or data hazards, indicating reliable performance under simulated conditions. The structured data flow also ensures reduced latency and eliminates bottlenecks commonly observed in traditional designs. Another key observation from this project is the balanced trade-off between complexity and performance. While the systolic array introduces additional structural organization compared to conventional methods, the benefits in terms of speed, throughput, and efficiency far outweigh the increase in design complexity. The regular interconnection pattern simplifies routing and control logic, further supporting practical implementation. In conclusion, this project clearly demonstrates that systolic array architecture is a powerful and efficient solution for matrix multiplication in high-performance computing systems. The combination of parallel processing, pipelined execution, and optimized data movement significantly enhances computational efficiency while reducing hardware resource usage. The proposed design is not only accurate and reliable but also scalable and adaptable to various application requirements. Therefore, the systolic array-based matrix multiplication architecture developed in this project is highly suitable for applications such as digital signal processing, image and video processing, machine learning, scientific computing, and real-time systems. It provides a strong foundation for building advanced hardware accelerators and contributes to the development of efficient computing solutions in modern digital systems.

REFERENCES

[1] H. T. Kung, "Why Systolic Architectures?", IEEE Computer, vol. 15, no. 1, pp. 37–46, Jan. 1982.

[2] H. T. Kung and C. E. Leiserson, "Systolic Arrays for VLSI," Sparse Matrix Proceedings, Society for Industrial and Applied Mathematics, 1979.

[3] D. A. Patterson and J. L. Hennessy, "Computer Organization and Design: The Hardware/Software Interface", 5th ed., Morgan Kaufmann, 2014.

[4] J. L. Hennessy and D. A. Patterson, "Computer Architecture:" A Quantitative Approach, 6th ed., Morgan Kaufmann, 2019.

[5] S. V. Adve and K. Gharachorloo, "Shared Memory Consistency Models: A Tutorial," IEEE Computer, vol. 29, no. 12, pp. 66–76, 1996.

[6] M. Flynn, "Very High-Speed Computing Systems," Proceedings of the IEEE, vol. 54, no. 12, pp. 1901–1909, 1966.

[7] T. G. Mattson, B. A. Sanders, and B. L. Massingill, "Patterns for Parallel Programming", Addison Wesley, 2004.

[8] K. K. Parhi," VLSI Digital Signal Processing Systems: Design and Implementation", Wiley, 1999.

[9] R. P. Brent and H. T. Kung, "A Regular Layout for Parallel Adders," IEEE Transactions on Computers, vol. C-31, no. 3, pp. 260–264, 1982.

[10] S. Hauck and A. DeHon, "Reconfigurable Computing: The Theory and Practice of FPGA-Based Computation", Morgan Kaufmann, 2007.

[11] M. J. Flynn and W. Luk, "Computer System Design: System-on-Chip", Wiley, 2011.

[12] Xilinx Inc., Vivado Design Suite User Guide, 2024.

[13] Intel Corporation, FPGA Architecture and Design Handbook, 2023.

[14] N. P. Jouppi et al., "In-Datacenter Performance Analysis of a Tensor Processing Unit," Proceedings of ISCA, 2017.

[15] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," Nature, vol. 521, pp. 436–444, 2015.

[16] S. Williams et al., "Roofline: An Insightful Visual Performance Model for Multicore Architectures," Communications of the ACM, vol. 52, no. 4, pp. 65–76, 2009.

[17] A. Pedram and J. Rabaey, Power Aware Design Methodologies, Springer, 2002.

[18] J. Cong and B. Xiao, "Minimizing Computation in Convolutional Neural Networks," Proceedings of ICANN, 2014.