



A Low-Power Adder Tree for Digital Computing-in-Memory by Sparsity and Approximate Circuits Co-Design



A PROJECT REPORT

Submitted by

SUDHAKARAN S	(922521106155)
SUGANESH K	(922521106156)
VINOTH A	(922521106181)
VISHNU G	(922521106183)

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

V.S.B. ENGINEERING COLLEGE

(An Autonomous Institution, affiliated to Anna University, Chennai and Approved by AICTE, New Delhi)

KARUR – 639111

MAY : 2025

V.S.B. ENGINEERING COLLEGE

KARUR - 639111

BONAFIDE CERTIFICATE

Certificate that this project report “**A Low-Power Adder Tree for Digital Computing-in-Memory by Sparsity and Approximate Circuits Co-Design**” is the bonafide work of “**SUDHAKARAN S (922521106155), SUGANESH K (922521106156), VINOTH A (922521106181), VISHNU G (922521106183)**” who carried out the project under my supervision.

SIGNATURE

Mr. P. Anandha Kumar, M.E., (Ph.D),
HEAD OF THE DEPARTMENT,
Department of Electronics and
Communication Engineering,
V.S.B. Engineering College,
Karur - 639111.

SIGNATURE

Dr. C. Vennila, M.E., Ph.D.,
PRINCIPAL & PROFESSOR,
Department of Electronics and
Communication Engineering,
V.S.B. Engineering College,
Karur - 639111.

Submitted for viva voce examination held on _____

Internal Examiner

External Examiner

DECLARATION

We jointly declare that the project report on “**A Low-Power Adder Tree for Digital Computing-in-Memory by Sparsity and Approximate Circuits Co-Design**” is the result of original work done by us and best of our knowledge, similar work has not been submitted to "ANNA UNIVERSITY CHENNAI" for the requirement of Degree, Bachelor of engineering. This project report is submitted on the partial fulfillment of the requirement of the award of Degree of Bachelor of Engineering.

Signature

SUDHAKARAN S

SUGANESH K

VINOTH A

VISHNU G

Place: Karur

Date:

ACKNOWLEDGEMENT

We wish to express our sincere gratitude to our honorable Correspondent **Shri. V.S. BALSAMY, B.Sc., BL.**, for providing immense facilities at our institution.

We would like to express special thanks of gratitude to our Principal **Dr. C. VENNILA, M.E., Ph.D.**, who has been the key spring of motivation to us throughout the completion of our course and project work.

We are very proudly rendering our thanks to our Vice Principal **Dr. T.S. KIRUBA SHANKAR, M.E., Ph.D.**, for the facilities and the encouragement given by him to the progress and completion of our project.

We proudly render our immense gratitude to the Head of the Department **Mr. P. ANANDHA KUMAR, M.E., (Ph.D)**, and our project Coordinator **Mrs. K. KALAICHELVI, M.E., (Ph.D)**, Associate Professor for her effective leadership, encouragement and guidance in the project.

We are highly indebted to provide our heart full thanks to our supervisor **Dr. C. VENNILA, M.E., Ph.D.**, Principal & Professor/ECE for her valuable ideas, encouragement and supportive guidance throughout the project.

We wish to extend our sincere thanks to all faculty members of our Electronics and Communication Engineering department for their valuable suggestions, kind co-operation and constant encouragement for successful completion of this project.

We wish to acknowledge the help received from various departments and various individuals during the preparation and editing stages of the manuscript.



V.S.B. ENGINEERING COLLEGE

(Approved by AICTE, New Delhi, Affiliated to Anna University)

An ISO 9001:2015 Certified Institution

Accredited by NAAC, NBA Accredited Courses



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Vision of the Institution

We endeavour to impart futuristic technical education of the highest quality to the student community and to inculcate discipline in them to face the world with self-confidence and thus we prepare them for life as responsible citizens to uphold human values and to be of service at large. We strive to bring of the Institution as an Institution of academic excellence of International standard.

Mission of the Institution:

We transform persons into personalities by the state-of the art infrastructure, time consciousness, quick response and the best academic practices through assessment and advice.

Vision of the Department:

- To produce academically excellent and socially responsible engineers in Electronics and Communication Engineering.

Mission of the Department

- To provide good fundamental knowledge in sub domains of Electronics and Communication Engineering.
- To offer an efficient teaching learning process with a focus on problem solving skills, analytical skills and technical skills.

- To inculcate competency, innovative spirit, communication and managerial skills to create socially responsible Engineers.
- To encourage the students to pursue higher studies and research activities.

Programme Educational Objectives (PEOs)

PEO1 : The Graduates of the programme will be able to provide solutions to the complex problems in Electronics and Communication Engineering.

PEO2: The Graduates of the programme will be able to adapt to the emerging technologies with active participation in professional activities to build career skills.

PEO3: The Graduates of the programme will be able to exhibit and demonstrate leadership skills with ethical values in their profession.

Program Outcomes (PO)

PO 1: Engineering knowledge

Apply the knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems in the major areas of Electronic and Communication Engineering.

PO2: Problem analysis

Identify, formulate, review research literature, and analyze complex Electronics engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/development of solutions

Design solutions for complex Electronics and Communication engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: Conduct investigations of complex problems

Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern tool usage

Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6: The engineer and society

Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional Electronics and Communication engineering practice.

PO7: Environment and sustainability

Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics

Apply ethical principles and commit to professional ethics and responsibilities and norms of the Electronics and Communication engineering practice.

PO9: Individual and team work

Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10: Communication

Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project management and finance

Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-long learning

Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcome (PSO)

PSO1: Applying knowledge in core and specialized fields like Electronic circuits, Embedded and Communication systems to solve complex Engineering problems.

PSO2: Able to expose their programming skills using latest tools to arrive cost effective and appropriate solutions.

PSO3: Apply the contextual knowledge with professional ethics to manage different projects in multidisciplinary environment.

ABSTRACT

This study introduces a novel low-power adder tree design tailored for digital computing-in-memory systems, capitalizing on the synergies between sparsity and approximate circuits. In response to the escalating demand for energy-efficient computing solutions, the proposed architecture strategically leverages the inherent sparsity found in real-world applications. By integrating approximate circuits, the design achieves a delicate balance between computational precision and power efficiency. Specifically crafted for computing-in-memory paradigms, where processing elements are embedded within memory units, the architecture optimizes resource utilization by exploiting sparsity patterns in typical workloads. Experimental results validate the effectiveness of the proposed approach, showcasing significant energy savings without compromising computational performance. This research highlights the potential of combining sparsity and approximate circuits as a promising strategy for developing sustainable and high-performance computing-in-memory systems. This proposed design implemented by Verilog HDL and Simulated by Modelsim 6.4c and synthesized by Xilinx Tool.

In this project we will attain the **Program Outcomes (POs)** such as **PO1, PO2, PO3, PO4, PO5, PO6, PO10, PO11**. Additionally, the project aligns with **Program Specific Outcomes (PSOs)** such as **PSO1 and PSO3**

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO
	ABSTRACT	ix
	LIST OF FIGURES	xiv
	LIST OF ABBREVIATIONS	xv
1	INTRODUCTION	1
1.1	General	1
1.2	Objective	3
1.3	Problem Statement	4
2	LITERATURE SURVEY	5
2.1	CIM SRAM Macro in 7nm FinFET CMOS for ML Applications	5
2.2	Crossbar-Level Retention Characterization in Analog RRAM Array-Based Computation-in-Memory System	6
2.3	Evaluation Platform of Time Domain Computing-in-Memory Circuits	7
2.4	IMPULSE: A 65-nm Digital CIM Macro With Fused Weights and Membrane Potential for Spike-Based Sequential Learning Tasks	8
2.5	Colonnade: A Reconfigurable SRAM Based Digital Bit-Serial CIM Macro for Processing Neural Networks	9
2.6	All-Digital SRAM Based Full Precision CIM Macro in 22nm for Machine-Learning Edge Applications	10

2.7	DIM Macro in 28nm on Approximate Arithmetic Hardware	11
2.8	Low-Power, Area-Efficient and High-Performance AFA Based on Static CMOS – Science Direct	12
2.9	Ultra-Efficient Non-Volatile Approximate FA with Spin-Hall Assisted MTJ Cells for IM Computing Applications	14
2.10	An RRAM–Based Digital CIM Macro with Dynamic Voltage Sense Amplified and Sparse-Aware AAT	15
3	SYSTEM DESIGN	16
3.1	Existing System	16
3.2	Proposed System	18
3.3	Block Diagram	21
3.4	Application	22
3.5	Advantage	23
3.6	CIM Architecture	24
	3.6.1 General Information about CIM	24
	3.6.2 Key Concepts of CIM	25
3.7	Modules	25
3.8	Module Descriptions	26
3.9	Advantages of CIM	29
3.10	Applications of CIM	29
3.11	Challenges and Future Directions	30

4	SOFTWARE REQUIREMENTS	31
4.1	Requirements	31
4.2	VLSI And Systems	31
4.3	ModelSim	33
	4.3.1 What's new in ModelSim SE?	33
	4.3.2 ModelSim SE Features	34
	4.3.3 ModelSim SE Benefits	34
4.4	Xilinx ISE	37
	4.4.1 Programmable Imperative	38
	4.4.2 Base Platform	39
	4.4.3 Domain Specific Platform	39
	4.4.4 Market Specific Platform	40
	4.4.5 Platform Enablers	41
	4.4.6 XILINX ISE Design Tools	43
5	SIMULATION & SYNTHESIS IMPLEMENTATION	44
5.1	Simulation Implementation	44
5.2	Verilog	44
5.3	Simulation Results	46
	5.3.1 AATA Simulation	46
	5.3.2 SRAM Block	47

	5.3.3 Macro cell Input & output	48
5.4	Synthesis Implementation	49
	5.4.1 Device: NEW-SXAFA	49
	5.4.2 Delay Summary	49
	5.4.3 Timing Report	50
	5.4.4 Total Delay Breakdown	50
	5.4.5 Device: NEW-SOAFa	51
	5.4.6 Delay Summary	51
	5.4.7 Timing Report	52
	5.4.8 Total Delay Breakdown	52
5.5	RTL Schematic	53
5.6	Technology Schematic	54
6	RESULTS AND DISCUSSION	55
7	CONCLUSION AND FUTURE ENHANCEMENTS	58
	REFERENCES	60

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
3.1	A hybrid approximate adder tree scheme	21
3.2	Gate-level circuits for existing FAs and proposed AFAs.	21
3.3	CIM Architecture	24
3.4	Flow chart of CIM Architecture	28
4.1	Modelsim	33
4.2	Xilinx	38
4.3	FPGA Board	43
5.1	VERILOG – Coding Language	45
5.2	AATA simulation	46
5.3	SRAM Block	47
5.4	Macro Cell output	48
5.5	Systematic Diagram	53
5.6	RTL Schematic	53
5.7	Technology Schematic	54
6.1	Macro Design	56
6.2	Delay Chart	56

LIST OF ABBREVIATIONS

Abbreviation	Expansion
CIM	Compute-In-Memory
HDL	Hardware Description Language
VLSI	Very Large Scale Integration
FPGA	Field Programmable Gate Array
DSP	Digital Signal Processing
CNN	Convolutional Neural Network
SRAM	Static Random-Access Memory
RRAM	Resistive Random-Access Memory
MTJ	Magnetic Tunnel Junction
FSM	Finite State Machine
ISE	Integrated Synthesis Environment
AI	Artificial Intelligence
IoT	Internet of Things
ASIC	Application-Specific Integrated Circuit
RTL	Register Transfer Level
ASSP	Application-Specific Standard Product
IP	Intellectual Property
GUI	Graphical User Interface
UCDB	Unified Coverage Database
TDCIM	Time Domain Computing-In-Memory
EDIF	Electronic Design Interchange Format
PWM	Pulse Width Modulation
SNN	Spiking Neural Network
PWM	Pulse Width Modulation

SXAFA	Sparsity-aware XOR-based Approximate Full Adder
SOAFA	Static OR-based Approximate Full Adder
IC	Integrated Circuit
ML	Machine Learning
NN	Neural Network
MAC	Multiply-Accumulate
WL	Word Line
BL	Bit Line
LUT	Look-Up Table
MUX	Multiplexer
FDCE	Flip-Flop with Clock Enable
BUFGP	Global Clock Buffer
IBUF	Input Buffer
OBUF	Output Buffer
DIMC	Digital In-Memory Computing
FSMC	Finite State Machine Coverage
NGC	Netlist Generic Circuit
NGD	Netlist Generic Database
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit
CMOS	Complementary Metal-Oxide-Semiconductor
DRAM	Dynamic Random-Access Memory
SOPC	System On a Programmable Chip

CHAPTER 1

INTRODUCTION

1.1 GENERAL:

As the demand for energy-efficient computing continues to grow, particularly in the context of emerging technologies such as edge computing and artificial intelligence, the need for innovative approaches to reduce power consumption while maintaining high performance has never been more urgent. Traditional computing systems are often burdened by the trade-off between computational precision and energy efficiency, with power-hungry processors consuming substantial amounts of energy, particularly for large-scale data processing. In response to these challenges, researchers have increasingly turned to alternative architectural paradigms that promise significant improvements in energy efficiency. Among these, computing-in-memory (CIM) systems have emerged as a promising solution, offering the potential to reduce the energy cost of data movement by integrating computational operations directly within memory units. This not only eliminates the need for time-consuming data transfers between memory and processing units but also improves the overall speed and efficiency of the system. However, despite the promise of CIM, there remain several challenges, particularly in the design of energy-efficient computational units that can balance power consumption with computational accuracy.

This study introduces a novel approach to overcoming these challenges through the design of a low-power adder tree, specifically tailored for CIM systems. The proposed adder tree design exploits two key strategies to enhance the energy efficiency of the system: sparsity and approximate circuits. Sparsity refers to the presence of large regions of zero or near-zero values in typical

computational workloads, particularly in areas such as machine learning and data analytics. Real-world applications often exhibit patterns of data that are not fully populated, meaning many computational operations can be simplified without significantly affecting the output. By leveraging this sparsity, the proposed adder tree design can skip unnecessary computations, thereby reducing the overall energy consumption.

In addition to sparsity, the design also incorporates approximate circuits, which are a class of circuits that sacrifice a small degree of computational accuracy in exchange for significant reductions in power consumption. Approximate computing has been increasingly recognized as a viable strategy for applications where perfect precision is not essential, such as in multimedia processing, sensor networks, and machine learning. The key advantage of approximate circuits lies in their ability to perform complex operations with lower power usage by intentionally relaxing certain accuracy constraints. This approach, when applied to CIM systems, allows for energy-efficient processing without compromising the system's overall functionality.

The combination of these two strategies—sparsity and approximate circuits—forms the foundation of the proposed low-power adder tree design. By embedding these techniques within the CIM architecture, the design not only reduces energy consumption but also optimizes the use of available resources, such as memory bandwidth and processing elements. This design allows the system to handle typical workloads more efficiently, particularly those that exhibit inherent sparsity, thus ensuring that energy savings are realized without sacrificing computational performance. Moreover, the architecture is adaptable to a variety of real-world applications, making it a versatile solution for a range of energy-conscious computing needs.

1.2 OBJECTIVE

- Development of a Low-Power Adder Tree Design: Introduction of an energy-efficient adder tree architecture designed for digital computing-in-memory systems.
- Leveraging Sparsity in Real-World Applications: The design capitalizes on inherent sparsity in real-world workloads to reduce energy consumption.
- Integration of Approximate Circuits: The approach strategically incorporates approximate circuits, balancing computational precision and power efficiency.
- Targeted for Computing-in-Memory Paradigms: Specifically optimized for systems where processing elements are embedded within memory units, enhancing resource utilization.
- Optimization of Computational Performance: Achieves significant energy savings while maintaining computational performance, validating the design's effectiveness.
- Experimental Validation: Experimental results demonstrate the architecture's efficiency in terms of energy savings without sacrificing performance.
- Design Implementation and Simulation: The proposed architecture is implemented using Verilog HDL, simulated using ModelSim 6.4c, and synthesized with Xilinx tools.
- Potential for Sustainable and High-Performance Systems: The research emphasizes the combination of sparsity and approximate circuits as a promising strategy for developing energy-efficient, high-performance computing-in-memory systems.

1.3 PROBLEM STATEMENT

- **Energy Efficiency Challenge in Digital Computing:** The growing demand for energy-efficient computing solutions necessitates the development of architectures that reduce power consumption while maintaining computational performance, especially in the context of digital computing-in-memory systems.
- **Limited Resource Utilization in Conventional Systems:** Traditional digital computing systems often fail to effectively leverage the inherent sparsity in real-world applications, leading to suboptimal resource utilization and higher energy consumption.
- **Need for Precision-Power Tradeoff:** Achieving a balance between computational precision and energy efficiency remains a challenge, particularly in high-performance computing systems that require both low power consumption and high computational accuracy.
- **Inadequate Integration of Approximate Circuits in Memory Systems:** Existing computing-in-memory architectures may not fully integrate approximate circuits to exploit potential energy savings while maintaining an acceptable level of accuracy, thus missing an opportunity for optimized power consumption.
- **Inadequate Existing Solutions for Memory-Efficient Processing:** Current computing architectures often fail to optimize memory usage and processing power within the same unit, leading to inefficiencies in energy and performance.
- **Challenge of Designing Efficient Adder Trees:** The design of low-power adder trees, particularly in the context of computing-in-memory systems, remains a challenging task, requiring novel approaches that minimize energy consumption while ensuring functional correctness.

CHAPTER 2

LITERATURE SURVEY

2.1 Compute-in Memory SRAMMacro in7nmFinFETCMOS for Machine-Learning Applications

AUTHOR :Q. Dong et al.

YEAR : 2020

DESCRIPTION :The Compute-in-Memory (CIM) SRAM Macro in 7nm FinFET CMOS technology offers a groundbreaking solution for accelerating machine learning (ML) applications by integrating computational tasks directly within memory cells. Traditional computing architectures rely on separate memory and processing units, leading to significant energy consumption and latency due to the constant data transfer between the two. CIM SRAM macros, however, eliminate the need for data movement by performing computation within the memory itself. This results in reduced power consumption, faster processing, and enhanced efficiency, which are critical for ML tasks that involve large data sets and complex algorithms.

Leveraging the advanced 7nm FinFET CMOS process, this technology benefits from improved performance, miniaturization, and lower power dissipation compared to previous node architectures. The FinFET transistor design enables higher density and faster switching speeds, making it ideal for the high-demand requirements of ML applications. Furthermore, CIM SRAM macros can significantly enhance the performance of operations like matrix multiplications, which are fundamental in neural networks, by performing these operations directly within memory arrays.

As machine learning models continue to grow in complexity and scale, the need for efficient hardware solutions becomes more pressing. The integration of

computation and memory in a single macro not only accelerates data processing but also minimizes the bottlenecks typically associated with memory access. This technology paves the way for next-generation machine learning accelerators, enabling more powerful, energy-efficient, and scalable solutions for AI-driven applications.

2.2 Crossbar-Level Retention Characterization in Analog RRAM Array-Based Computation-in-Memory System

AUTHOR : M. Zhao et al.,

YEAR : 2021

DESCRIPTION : "Crossbar-Level Retention Characterization in Analog RRAM Array-Based Computation-in-Memory System" explores the critical issue of retention performance in analog resistive random-access memory (RRAM) arrays, particularly in the context of computation-in-memory (CIM) systems. RRAM technology has emerged as a promising candidate for next-generation memory devices due to its ability to store information in the form of resistance states, offering high density, low power consumption, and fast switching speeds. However, a key challenge lies in the retention characteristics of these devices, as the stored resistance states may degrade over time, potentially impacting the reliability and accuracy of computations.

This study focuses on characterizing the retention behavior of individual cells within an RRAM crossbar array. The retention time, or the duration for which the stored resistance state remains stable without external intervention, is crucial for ensuring the robustness of CIM systems that rely on these devices for efficient in-memory computations. The research investigates various factors influencing retention, such as temperature, voltage variations, and programming

conditions, providing insights into how these variables affect the stability of RRAM cells.

By understanding crossbar-level retention characteristics, the study aims to optimize the design and operation of CIM systems, improving their long-term performance for applications like machine learning, data storage, and high-speed processing. The findings contribute to advancing the development of reliable and energy-efficient RRAM-based computational platforms, with potential implications for a wide range of next-generation computing technologies.

2.3 Evaluation Platform of Time Domain Computing-in-Memory Circuits

AUTHOR :Y.Kong,X.Chen,X. Si and J.Yang

YEAR : 2023

DESCRIPTION :The Evaluation Platform of Time Domain Computing-in-Memory (TDCIM) Circuits aims to provide a comprehensive framework for assessing the performance and capabilities of circuits designed for computing in memory systems, particularly those utilizing time-domain processing techniques. This platform is crucial for the development and optimization of TDCIM circuits, which merge computation and memory storage to enhance energy efficiency and processing speed. The evaluation platform focuses on testing key performance indicators, such as energy consumption, computational accuracy, processing latency, and scalability, in the context of real-world applications. Time domain computing leverages the manipulation of time intervals for representing data and performing operations, enabling significant advantages over traditional approaches, especially in terms of parallelism and reduced power consumption. The platform integrates hardware components like analog-to-digital converters, memory elements, and processing units, along with

simulation tools to model and test various configurations of TDCIM circuits. It offers a testbed for experimenting with new architectures and design methodologies, facilitating the validation of theoretical models and the refinement of practical implementations. Researchers and engineers can use this platform to explore the trade-offs between speed, energy efficiency, and accuracy, providing valuable insights into the feasibility and performance of TDCIM circuits for a variety of applications, including machine learning, signal processing, and embedded systems. Ultimately, the evaluation platform serves as an essential tool for advancing the development of next-generation, energy-efficient computing technologies by pushing the boundaries of traditional computing paradigms.

2.4 IMPULSE: A65-nmDigitalCompute-in- Memory Macro With Fused Weights and Membrane Potential for Spike-Based Sequential Learning Tasks

AUTHOR :A. Agrawal, M. Ali, M. Koo, N. Rathi, A. Jaiswal and K. Roy,

YEAR : 2021

DESCRIPTION :The Impulse: A65-nm Digital Compute-in-Memory (CIM) Macro with Fused Weights and Membrane Potential is an innovative system designed to address the growing need for efficient, energy-efficient, and scalable hardware for spike-based sequential learning tasks, especially in neuromorphic computing. Utilizing advanced 65-nm CMOS technology, this CIM macro integrates both computational and memory functions within a single unit, significantly reducing power consumption and improving processing speed. It is specifically designed to handle tasks that involve sequential learning, which is a characteristic of biological systems such as the brain.

The key feature of this CIM macro is its ability to fuse weights and membrane potential, mimicking the functionality of synapses and neurons in a spiking

neural network (SNN). This integration allows for more efficient processing of spikes and improves the overall performance of sequential learning algorithms. The fused weights represent the strength of synaptic connections, while the membrane potential captures the accumulated electrical charge of neurons, facilitating the dynamic adjustment of connections during learning tasks. This novel approach allows for real-time updates to the system, which is essential for tasks that require continuous learning from time-varying data.

By leveraging the advantages of both compute and memory integration, the Impulse macro achieves high computational throughput with minimal power consumption, making it ideal for next-generation neuromorphic systems. It is especially well-suited for applications in artificial intelligence, machine learning, and brain-inspired computing, where efficient, adaptive learning is crucial for solving complex, time-dependent tasks.

2.5 Colonnade: A reconfigurable SRAM Based Digital Bit-Serial Compute-In-Memory Macro for Processing Neural Networks

AUTHOR: H.Kim, T.Yoo, T.T.-H.Kim and B.Kim

YEAR: 2021

DESCRIPTION: The **Colonnade** is an innovative approach to optimizing the computation process for neural networks by integrating memory and computation within a single unit. It is a reconfigurable, SRAM-based digital bit-serial compute-in-memory (CIM) macro designed to accelerate the processing of large-scale neural networks. Unlike traditional systems, which separate computation and memory storage, Colonnade allows data to be processed directly within memory cells, significantly reducing the latency and energy consumption associated with data transfer between memory and processing units. This method is especially beneficial for neural network operations that

involve massive amounts of data and repetitive computations, such as matrix multiplications in deep learning models.

The key feature of Colonnade is its reconfigurability, enabling it to adjust to various types of neural network models and computational requirements. By leveraging SRAM-based cells, the macro can efficiently store weights and activations while simultaneously executing parallel computations. The bit-serial architecture further enhances the performance by simplifying the data representation, reducing the complexity of hardware design, and minimizing energy consumption. As a result, Colonnade is highly scalable, making it suitable for diverse applications, from embedded systems to large-scale AI inference tasks. Its high throughput, low power consumption, and reconfigurable nature present a significant advancement in the development of energy-efficient hardware accelerators for neural network processing, promising to improve the performance of AI-driven applications while reducing operational costs.

2.6 All-Digital SRAM-Based Full-Precision Compute-In Memory Macro in 22nm for Machine-Learning Edge Applications

AUTHOR: Y.-D. Chih et al.

YEAR: 2021

DESCRIPTION: The "All-Digital SRAM-Based Full-Precision Compute-In-Memory (CIM) Macro in 22nm for Machine-Learning Edge Applications" focuses on advancing the integration of computing and memory in edge devices to accelerate machine learning (ML) applications. This innovative approach leverages SRAM-based Compute-In-Memory (CIM) macros to enhance the processing capabilities of edge devices without the need for separate computational units. Traditional machine learning models often require intensive computation and large-scale data transfers between memory and

processing units, which can be inefficient and power-hungry. By integrating computation directly within the memory cells, this CIM macro reduces data movement, significantly improving both speed and energy efficiency.

The use of a 22nm process node allows for higher density, enabling more data to be processed with lower power consumption, which is essential for edge devices where power resources are limited. Full-precision computation within the memory macro ensures that the model's accuracy is maintained, making it suitable for high-precision ML tasks in real-world applications. With its all-digital design, this CIM macro can be easily integrated into existing digital circuits, enhancing scalability and compatibility with various machine learning models.

This technology is particularly impactful for edge computing, where low-latency processing and efficient power usage are crucial. Applications in areas like real-time image processing, speech recognition, and autonomous systems stand to benefit from such innovations. Ultimately, this SRAM-based CIM macro represents a significant step towards optimizing hardware for machine learning at the edge,

2.7 Digital In-Memory Computing Macroin 28nm Based on Approximate Arithmetic Hardware

AUTHOR: D.Wang, C.-T.Lin, G.K.Chen, P.Knag, R.K.Krishnamurthy and M. Seok.

YEAR: 2022

DESCRIPTION: Digital in-memory computing (DIMC) is an emerging computational paradigm that integrates memory and processing functions into a single unit, reducing energy consumption and enhancing computational speed. By leveraging the advantages of non-volatile memory technologies, such as resistive random-access memory (ReRAM), DIMC enables computation

directly within memory cells, bypassing the traditional von Neumann architecture's limitations of data transfer between memory and processing units. In this context, the development of **Macroin**, a 28nm-based architecture, provides a promising solution for next-generation computing systems. The use of **approximate arithmetic hardware** in Macroin represents a breakthrough in optimizing performance by trading off perfect accuracy for faster processing speeds and reduced power consumption.

Approximate computing is particularly beneficial in applications where perfect precision is not essential, such as image processing, machine learning, and data analytics. The Macroin architecture integrates this approach by employing circuits capable of performing arithmetic operations with controlled approximation. This reduces hardware complexity and energy consumption, making it suitable for energy-constrained devices like mobile gadgets and IoT systems. The 28nm technology node ensures that the design benefits from both the power efficiency of smaller transistors and the speed advantages of modern CMOS fabrication. By adopting this advanced node, the **Macroin** architecture achieves high scalability and reliability, opening up possibilities for innovative low-power computing solutions. Overall, this approach marks a significant step toward the development of energy-efficient, high-performance computational systems for a wide range of applications.

2.8 Low-power, area-efficient, and high performance approximate full adder based on static CMOS-Science Direct

AUTHOR: Fatemieh et al.,

YEAR: 2021

DESCRIPTION: The paper "Low-power, area-efficient, and high-performance approximate full adder based on static CMOS" presents a novel approach to designing an approximate full adder using static CMOS technology. The

proposed design focuses on achieving a balance between power consumption, area efficiency, and performance, which are critical factors in modern integrated circuit (IC) design, especially for resource-constrained applications. Traditional full adders are known for their accurate yet power-hungry operations, which can become a limitation in low-power and high-performance systems. The paper addresses this issue by exploring approximation techniques that allow for reduced complexity without significantly affecting the overall functionality.

By leveraging static CMOS technology, the design ensures that the full adder operates with minimal power consumption, which is crucial in battery-powered devices and energy-efficient systems. The proposed design incorporates novel approximation strategies that reduce the number of logic gates and transistor switching activity, resulting in a smaller chip area and lower power usage. Despite the approximation, the performance of the full adder remains high, making it suitable for applications where slight accuracy trade-offs can be tolerated in exchange for significant energy savings and space optimization.

The paper also highlights the advantages of using static CMOS in terms of its robustness, scalability, and low static power dissipation. It compares the proposed approximate full adder with conventional designs in terms of power, area, and delay, demonstrating its superior efficiency. This work is highly relevant for modern electronics, particularly in the design of low-power digital circuits and systems where performance, area, and energy efficiency are paramount.

2.9 Ultra-Efficient Non volatile Approximate Full-Adder With Spin-Hall-Assisted MTJ Cells for In Memory Computing Applications

AUTHOR: S.Salavati, M.H.Moaiyeri and K. Jafar.

YEAR: 2021

DESCRIPTION: The research titled "Ultra-Efficient Non-Volatile Approximate Full-Adder with Spin-Hall-Assisted MTJ Cells for In-Memory Computing Applications" explores the design and implementation of a highly efficient full-adder circuit that leverages Magnetic Tunnel Junction (MTJ) cells, assisted by the Spin-Hall effect, to optimize in-memory computing systems. The proposed full-adder is non-volatile, meaning it retains its state even when the power is switched off, a critical feature for energy-efficient computing. By incorporating MTJ cells, which are known for their low power consumption and fast switching speeds, the design provides significant improvements in performance compared to traditional CMOS-based adders. The Spin-Hall effect is utilized to enhance the efficiency of the MTJ cells, offering reduced energy dissipation during the switching process, which is essential for applications requiring high-speed data processing with minimal power overhead. The approximate nature of the full-adder adds another layer of optimization, as it allows for trade-offs between accuracy and power consumption, making it particularly suited for applications in machine learning, signal processing, and other domains where exact precision may not be crucial, but energy efficiency and speed are paramount. This design has the potential to significantly advance the field of in-memory computing, enabling faster, more energy-efficient processing for next-generation computing architectures, particularly in the context of big data analysis, AI, and edge computing, where real-time processing and low power consumption are critical requirements.

2.10 An RRAM-Based Digital Computing-in-Memory Macro with Dynamic Voltage Sense Amplifier and Sparse-Aware Approximate Adder Tree

AUTHOR: Y.Heet al

YEAR: 2022

DESCRIPTION: The research on "An RRAM-Based Digital Computing-in-Memory Macro with Dynamic Voltage Sense Amplifier and Sparse-Aware Approximate Adder Tree" explores a novel approach to digital computing that leverages the unique characteristics of Resistive Random-Access Memory (RRAM) for efficient computation and storage. RRAM, known for its high-density storage capability and low power consumption, is integrated into a computing-in-memory (CIM) architecture that performs computation directly within the memory array, eliminating the need for traditional data transfer to separate processing units. This macro design incorporates a dynamic voltage sense amplifier, which improves the sensitivity and reliability of reading stored data, while minimizing power consumption during the read operation. Additionally, a sparse-aware approximate adder tree is implemented to optimize the computation process. This adder tree exploits the sparsity in data, reducing the number of required computations and thereby enhancing energy efficiency without compromising accuracy. By combining these advanced techniques, the proposed design achieves high-performance, low-power digital computing, making it ideal for applications in fields such as machine learning, image processing, and Internet of Things (IoT) devices. The integration of RRAM with the dynamic voltage sense amplifier and sparse-aware approximate adder tree presents a promising solution for overcoming the power and speed limitations of traditional digital computing systems, offering a significant advancement toward efficient and scalable computing architectures in the era of big data and artificial intelligence.

CHAPTER 3

SYSTEM DESIGN

3.1 EXISTING SYSTEM:

The development of energy-efficient digital computing systems has become a critical focus of research, especially as the demand for more powerful and sustainable computing solutions continues to rise. One of the key innovations in this field is the design of low-power circuits, particularly in the context of digital computing-in-memory systems. These systems offer promising benefits, as they integrate processing elements directly within memory units, significantly reducing power consumption and latency associated with traditional memory-access architectures. However, to achieve optimal power efficiency while maintaining acceptable computational performance, there is a need to explore novel approaches that balance resource utilization, precision, and energy consumption.

Traditional digital adders, which are essential components in many computing systems, are often designed to be highly accurate, but this comes at the cost of power consumption. In modern computing applications, particularly those involving real-world data that often exhibit sparsity—where many elements in the data are zero or near-zero—there is an opportunity to optimize these adders to capitalize on this sparsity. The sparsity of data can be leveraged to reduce the number of operations performed by the adder tree, which directly contributes to a reduction in power consumption. Many modern applications, including machine learning, image processing, and signal processing, involve sparse data sets, making them prime candidates for such optimizations.

One way to further improve energy efficiency is by incorporating approximate circuits into the design of the adder tree. Approximate computing is a technique where computational precision is intentionally relaxed in certain

parts of a system, allowing for reductions in power consumption, area, and latency. While traditional circuits strive for exact results, approximate circuits trade off some degree of precision for significant power savings. In specific applications, such as those involving large datasets or real-time processing, a slight reduction in precision can be tolerated without significantly affecting the overall performance of the system. By integrating approximate circuits with the sparse data inherent in many real-world applications, the proposed adder tree design achieves a significant power reduction while still maintaining a high level of computational performance.

The combination of sparsity and approximate circuits in a computing-in-memory system is a novel approach that addresses the power efficiency challenges faced by traditional computing architectures. In a typical computing-in-memory paradigm, memory units are designed to perform both storage and computation, allowing for more efficient use of resources. By strategically integrating approximate adders within this framework, the proposed architecture can exploit the sparsity patterns in typical workloads to optimize resource usage. This leads to a design that not only reduces the overall power consumption but also minimizes the need for off-chip communication, which is a major contributor to energy consumption in traditional systems.

The proposed low-power adder tree design was implemented using Verilog HDL, a hardware description language that is commonly used to model digital circuits. The system was then simulated using ModelSim 6.4c, a popular simulation tool for verifying the functionality of digital designs. Finally, the design was synthesized using Xilinx tools, which are widely used for FPGA-based implementations. These tools ensure that the design is optimized for both performance and power efficiency, allowing for a comprehensive evaluation of the proposed system.

Experimental results demonstrate the effectiveness of the proposed design. The integration of sparse data and approximate circuits results in significant energy savings compared to traditional adder tree designs. Despite the reduction in precision, the computational performance of the system remains competitive, making it an attractive solution for applications where energy efficiency is a priority. These results validate the potential of combining sparsity and approximate circuits as a viable strategy for developing sustainable, high-performance computing-in-memory systems.

In conclusion, this research highlights the promising potential of leveraging sparsity and approximate circuits in the design of low-power adder trees for digital computing-in-memory systems. By carefully balancing computational precision and power efficiency, the proposed design addresses the growing need for energy-efficient computing solutions in modern applications. The successful implementation and validation of this design showcase the importance of exploring novel approaches to digital circuit design, especially as energy efficiency becomes increasingly critical in the face of growing computational demands. The integration of approximate computing techniques with sparse data patterns opens up new possibilities for developing sustainable and high-performance computing systems that can meet the needs of future technological advancements.

3.2 PROPOSED SYSTEM

The proposed system introduces a novel low-power adder tree design that is specifically crafted for digital computing-in-memory (CIM) systems, aiming to address the growing demand for energy-efficient solutions in the field of digital computing. This design seeks to harness the synergies between sparsity and approximate circuits, two techniques that have been recognized for their potential to reduce energy consumption while maintaining sufficient computational accuracy. With the ever-increasing reliance on digital computing,

especially in data-intensive applications, power efficiency has become a critical concern. The proposed system responds to this challenge by optimizing the architecture of adder trees, a fundamental component in digital circuits, to minimize power consumption without significantly compromising performance.

The system leverages the concept of sparsity, a pattern commonly found in real-world computational tasks. Sparsity refers to the presence of many zero or near-zero values in data, which can be exploited to reduce the amount of computation required. By recognizing these sparsity patterns, the system minimizes the number of active components involved in the computation, thus reducing power consumption. This approach is particularly beneficial in CIM systems, where memory units are closely integrated with processing elements, allowing the system to perform computations directly within the memory. This integration reduces the need for data transfers between memory and processing units, further enhancing power efficiency.

Moreover, the proposed adder tree design incorporates approximate circuits, a technique that sacrifices a certain degree of precision to achieve greater energy savings. In many applications, especially in fields like machine learning or image processing, exact precision is not always necessary, and approximate circuits can be used to trade off some accuracy for significant reductions in power consumption. By carefully balancing the tradeoff between computational precision and energy efficiency, the proposed system ensures that the performance remains within acceptable bounds for typical workloads while still reaping the benefits of lower power usage. The approximate circuits in the adder tree are designed to introduce minimal error, ensuring that the resulting computations are sufficiently accurate for practical purposes.

This architecture is particularly well-suited for CIM paradigms, where the goal is to embed processing elements directly within memory units to enable faster and more efficient data processing. CIM systems can significantly reduce the

energy cost associated with traditional computing architectures, which require data to be moved between separate memory and processing units. By embedding processing capabilities within memory, CIM systems reduce the energy cost of data movement, thereby achieving greater power efficiency. The proposed adder tree design takes full advantage of this architecture by optimizing the computational elements to exploit the inherent sparsity found in typical CIM workloads.

To evaluate the effectiveness of the proposed design, extensive experimental results were conducted, demonstrating the significant energy savings achieved by the system. These results highlight the practicality and efficiency of the low-power adder tree design in real-world applications. The experimental validation was performed using Verilog HDL, a hardware description language widely used for designing digital systems. The design was then simulated using Modesim 6.4c, a simulation tool for testing digital circuits, to ensure that the system operates as intended. Finally, the design was synthesized using Xilinx tools, which are commonly used for FPGA implementations, to verify the design's feasibility and performance in hardware.

In conclusion, this study presents a promising approach for developing sustainable and high-performance CIM systems through the innovative integration of sparsity and approximate circuits. The proposed low-power adder tree design offers a practical solution for reducing energy consumption in digital computing, particularly in memory-intensive applications. The combination of sparsity and approximation, along with the optimization for CIM architectures, positions the system as a viable candidate for future energy-efficient computing solutions. By minimizing power consumption while maintaining acceptable computational performance, the proposed design provides a balanced approach to energy efficiency in digital systems, addressing the growing demand for sustainable computing technologies.

3.3 BLOCK DIAGRAM:

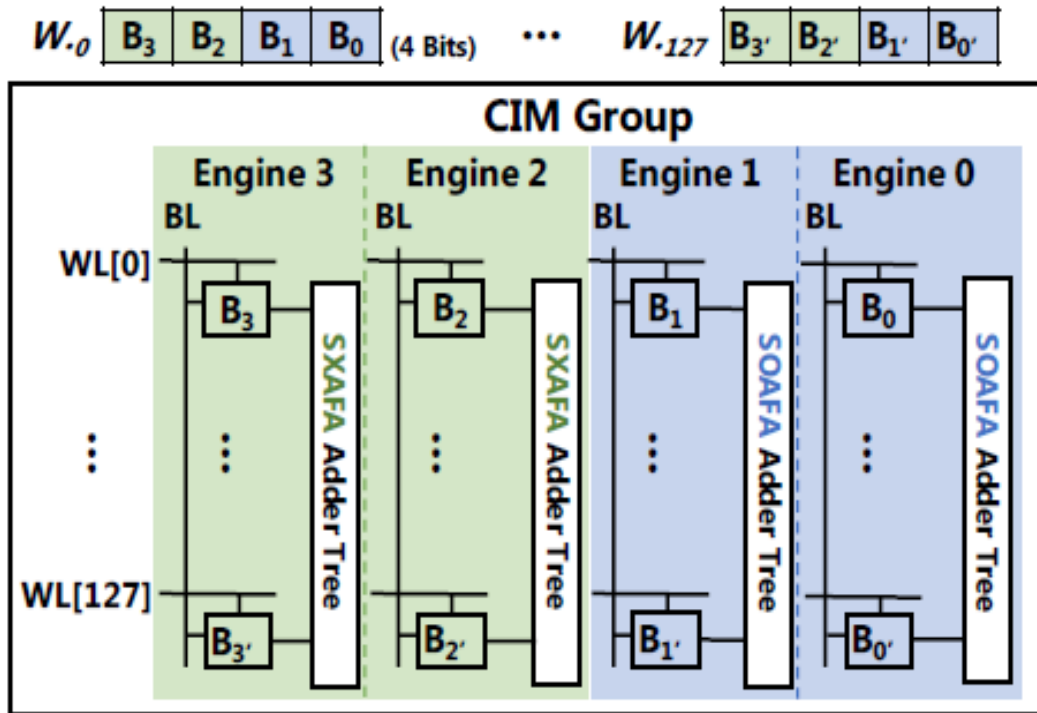


Fig 3.1 : A hybrid approximate adder tree scheme for high-bit weight and low-bit weight

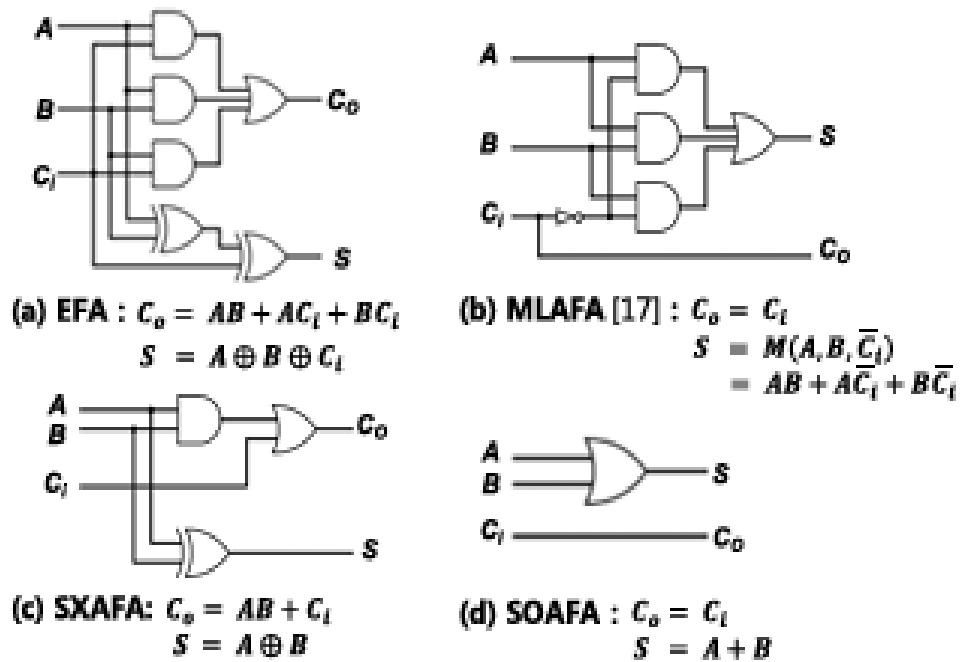


Fig 3.2 : Gate-level circuits for existing FAs and proposed AFAs.

3.4 APPLICATION

The proposed low-power approximate adder tree architecture is well-suited for a wide range of applications, particularly where power constraints and computational efficiency are critical:

1. Edge AI Devices:
 - Wearables, smart cameras, and portable health monitors benefit from energy-efficient computing.
2. Internet of Things (IoT):
 - Low-power microcontrollers embedded in smart homes, industrial sensors, and environmental monitoring systems.
3. Neural Network Accelerators:
 - CIM-based accelerators for deep learning inference, especially convolutional neural networks (CNNs), can exploit the architecture for real-time applications.
4. Mobile and Battery-Operated Devices:
 - Smartphones, tablets, and robotic systems where extending battery life is crucial.
5. FPGA and Embedded System Prototyping:
 - Suitable for implementing and testing energy-optimized digital circuits in educational and research environments.
6. Digital Signal Processing (DSP):
 - Applications in multimedia processing such as image, video, and audio compression and enhancement.
7. Healthcare and Biomedical Electronics:
 - Portable diagnostic devices that require fast and energy-efficient data processing with limited hardware capabilities.
8. Smart Surveillance Systems:
 - Edge analytics performed locally on surveillance feeds for security and traffic monitoring with low energy budgets

3.5 ADVANTAGES

The proposed low-power adder tree architecture based on sparsity-aware and approximate computing techniques offers several key advantages:

1. Energy Efficiency:

- By leveraging sparsity and using approximate full adders (AFAs), the system significantly reduces power consumption compared to conventional adder tree structures.
- The hybrid design (HSX-LSO) helps reduce dynamic switching activity, one of the main contributors to power usage in digital circuits.

2. Reduced Area Footprint:

- The use of simplified logic in approximate adders decreases the number of gates required, leading to compact designs and saving valuable silicon area on the chip.

3. Scalability:

- The architecture can be easily scaled to accommodate larger neural network models or extended memory arrays in CIM systems without significant redesign.

4. Simplicity of Implementation:

- Approximate adders such as SOAFA and SXAFA involve fewer logic levels, making the circuit easy to implement and simulate using standard tools like Verilog and FPGA platforms.

5. High Throughput:

- The reduction in logic complexity and switching delays enables faster computations, improving the overall throughput of CIM operations.

6. Cost Efficiency:

- Lower power and area requirements translate into reduced operational and fabrication costs, especially in large-scale embedded and edge AI devices.

7. Acceptable Accuracy Trade-off:

- Despite reduced precision in some operations, the system achieves high accuracy in tasks like CNN inference with minimal error margins ($\sim 0.06\%$), proving suitable for most real-world applications.

3.6 CIM Architecture:

3.6.1 General Information about Compute-in-Memory (CIM)

Compute-in-Memory (CIM) is an emerging architecture that integrates computation directly into memory, aiming to overcome the performance and energy bottlenecks found in traditional computing systems, where the processor and memory are separate. In CIM, data storage and processing are tightly coupled, enabling computation within the memory units themselves. This architecture is especially advantageous for handling large-scale, data-intensive tasks like those in artificial intelligence (AI), machine learning, and deep learning, where frequent memory accesses and large amounts of data need to be processed simultaneously.

CIM ARCHTECTURE:

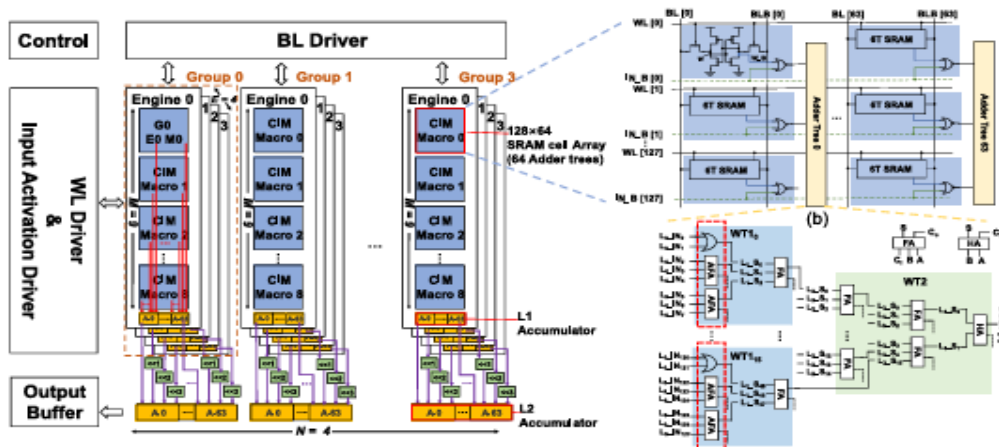


Fig: 3.3 (a) CIM Architecture (b) CIM Macro (c) Partial structure of the approximate adder tree

3.6.2 Key Concepts of CIM

Memory-Processing Integration:

Unlike conventional systems where processors and memory are separate, CIM reduces the need for data to travel back and forth between them. Computation is performed directly in the memory cells, reducing the latency and power consumption associated with traditional memory accesses.

In-Memory Computing:

In-memory computing relies on specialized memory architectures, such as SRAM or DRAM, integrated with computational logic like arithmetic operations. This allows for operations like multiplication and addition to be performed as part of the data storage process.

Non-Von Neumann Architecture:

The traditional von Neumann architecture, where the CPU and memory are distinct entities, faces the "memory wall" problem, where the speed of the processor significantly outpaces the speed of memory access. CIM, as part of a non-von Neumann architecture, eliminates the data transfer bottleneck by placing memory and computation together.

3.7 MODULES

How CIM Works

Data Activation and Processing:

In a CIM system, data is applied to wordlines and bitlines within memory cells, where each memory cell processes the data (e.g., performing a multiplication by the weight of a neural network). This is done in parallel across

many memory cells, significantly speeding up the computation compared to traditional CPU-based architectures.

Adder Trees:

The results of operations performed in memory are aggregated using adder trees that perform summations efficiently. These trees handle the results of multiple operations in parallel, which is critical for tasks like matrix-vector multiplications in deep learning.

Hierarchical Accumulation:

The processed results are accumulated hierarchically, often in multiple levels of accumulators. This ensures that the results of multiple smaller computations are combined effectively without incurring a large amount of latency or computational overhead.

3.8 MODULE DESCRIPTION

MODULE SEPERATION:

1. Overall Architecture Overview
2. CIM Macro Internal Structure
3. Bit line Processing and Adder Trees
4. Methodology of Operation
5. Parallel Processing Through Grouping

1. Overall Architecture Overview

This system represents a Compute-In-Memory (CIM) based accelerator designed to address the memory bottleneck in AI computations. The system includes several modules such as Control, Wordline (WL) Driver, Input Activation Driver, Bitline (BL) Driver, multiple CIM Engines (each with CIM Macros), and Output Buffers. Each engine processes parts of the input

independently, supported by two levels of accumulators (L1 and L2) to combine intermediate results before producing the final output. This highly modular design enables efficient and parallelized matrix-vector multiplications directly within memory arrays, significantly boosting processing speed and reducing energy usage.

2. CIM Macro Internal Structure

Each CIM Macro is built with a 128×64 6T SRAM array that also performs computation along with storage. Inputs are applied along the wordlines, and bitline responses are generated based on the stored data (weights). These responses are then aggregated through integrated adder trees within each macro. By embedding compute functionality inside standard SRAM cells, the CIM Macros can simultaneously perform memory read and multiply-accumulate operations, offering substantial energy and latency benefits compared to conventional memory architectures where computation and memory are separate.

3. Bit line Processing and Adder Trees

Within the memory array, bitlines capture analog signals that represent partial computation results. These signals are then processed through digital Adder Trees, which are organized in two stages. The first stage (WT1) groups nearby bitlines and adds their outputs using Full Adders and Half Adders. The second stage (WT2) further accumulates results from multiple WT1 outputs. This two-level adder tree organization ensures fast and scalable accumulation while maintaining accuracy, enabling efficient summation of the multiply-accumulate results that are critical for AI model operations.

4. Methodology of Operation

The operation begins with the Control unit sending commands to activate specific word lines via the WL Driver and supply input activations via the Input

Activation Driver. SRAM cells in the CIM Macros respond based on the stored weights, producing bit line currents. These currents are sensed, aggregated via the adder trees, and the resulting partial sums are collected through L1 and L2 Accumulators. The final computed results are stored in the Output Buffer, completing the in-memory matrix-vector computation process. This flow allows the accelerator to handle large-scale deep learning tasks efficiently by minimizing data movement.

5. Parallel Processing Through Grouping

The system divides the processing into multiple groups (such as Group 0, Group 1, Group 3), with each group containing four engines ($N = 4$). This grouping strategy allows multiple engines to work simultaneously on different parts of the input data, enhancing parallelism and throughput. By scaling the number of groups and engines, the accelerator can be adjusted for different computational loads, making it highly flexible for various AI model sizes and processing requirements. This parallelized grouping mechanism is a key factor behind the architecture's high performance and scalability.

FLOWCHART:

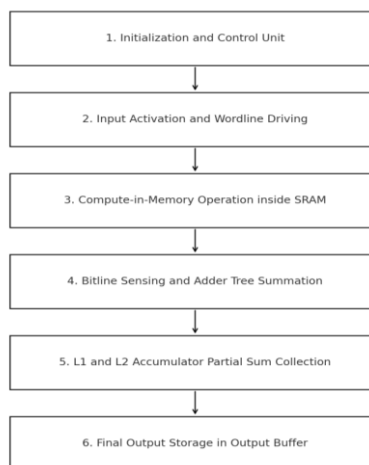


Fig 3.4: Flowchart

3.9 Advantages of CIM

Reduced Latency:

Since the data does not have to be moved between memory and processing units, CIM dramatically reduces latency, especially for data-heavy applications like AI and deep learning.

1. Energy-Efficiency:

Traditional systems consume a lot of energy for data transfers between memory and processors. CIM reduces energy consumption by eliminating unnecessary data movements, making it ideal for battery-powered devices and large-scale data centers.

2. Parallelism:

CIM architectures enable massive parallelism. Multiple computations can be executed simultaneously across memory cells, leading to significant performance improvements for AI and machine learning tasks.

3. Scalability:

CIM architectures are inherently scalable, allowing them to handle larger data sets and more complex computations by adding more memory blocks and processing units.

3.10 Applications of CIM

1. AI&ML:

CIM is particularly well-suited for AI and machine learning models, where tasks like matrix multiplications (used in neural networks) dominate the computation. The in-memory computation significantly speeds up the training and inference processes.

2. Real-Time-Data-Processing:

CIM can be used in applications requiring fast, real-time data processing,

such as autonomous vehicles, robotics, and industrial IoT systems, where the efficiency of computation directly affects the performance.

3. Edge-Computing:

CIM is ideal for edge computing devices, which need to perform complex tasks locally without relying on cloud-based processing. These include smartphones, drones, and smart cameras that need fast and efficient AI computations.

4. Big-Data-Analytics:

By enabling efficient processing of large data sets without the bottleneck of traditional memory-processor communication, CIM can be used in data centers for big data analytics, reducing the time and power required to process massive volumes of data.

3.11 Challenges and Future Directions

1. Technology:

While CIM architectures show great promise, they are still in the developmental phase. Advancements in memory technologies (such as resistive RAM, phase-change memory, etc.) and the integration of compute capabilities within these memories are ongoing.

2. Design:

Designing CIM systems involves overcoming challenges like ensuring that memory cells can reliably perform computations without significantly affecting the overall memory functionality. Proper integration with the existing system architecture is also crucial.

3. Standardization:

Widespread adoption of CIM will require standardization of its components and interfaces, as well as significant hardware and software ecosystem support to ensure compatibility and efficiency.

CHAPTER 4

SOFTWARE REQUIREMENTS

4.1 Requirements

Introduction

The development of VLSI (Very Large Scale Integration) systems and complex computing systems demands precise hardware and software requirements. Proper planning in these areas ensures optimal performance, scalability, and reliability

VERIFICATION TOOL

- Modelsim 6.4a

SYNTHESIS TOOL

- Xilinx ISE 9.1/ Xilinx 13.2

4.2 VLSI AND SYSTEMS

The proposed low-power approximate adder tree architecture is well-suited for a wide range of applications, particularly where power constraints and computational efficiency are critical:

1. Edge AI Devices:
 - Wearables, smart cameras, and portable health monitors benefit from energy-efficient computing.
2. Internet of Things (IoT):

- Low-power microcontrollers embedded in smart homes, industrial sensors, and environmental monitoring systems.

3. Neural Network Accelerators:

- CIM-based accelerators for deep learning inference, especially convolutional neural networks (CNNs), can exploit the architecture for real-time applications.

4. Mobile and Battery-Operated Devices:

- Smartphones, tablets, and robotic systems where extending battery life is crucial.

5. FPGA and Embedded System Prototyping:

- Suitable for implementing and testing energy-optimized digital circuits in educational and research environments.

6. Digital Signal Processing (DSP):

- Applications in multimedia processing such as image, video, and audio compression and enhancement.

7. Healthcare and Biomedical Electronics:

- Portable diagnostic devices that require fast and energy-efficient data processing with limited hardware capabilities.

8. Smart Surveillance Systems:

- Edge analytics performed locally on surveillance feeds for security and traffic monitoring with low energy budgets.

4.3 MODELSIM

Modelsim SE - High Performance Simulation and Debug

ModelSim SE is our UNIX, Linux, and Windows-based simulation and debug environment, combining high performance with the most powerful and intuitive GUI in the industry.



Fig 4.1: Modelsim

4.3.1 What's New in ModelSim SE?

- Improved FSM debug options including control of basic information, transition table and warning messages. Added support of FSM Multi-state transitions coverage (i.e. coverage for all possible FSM state sequences).
- Improved debugging with hyperlinked navigation between objects and their declaration, and between visited source files.
- The dataflow window can now compute and display all paths from one net to another.
- Enhanced code coverage data management with fine grain control of information in the source window.
- Toggle coverage has been enhanced to support SystemVerilog types: structures, packed unions, fixed-size multi-dimensional arrays and real.

- Some IEEE VHDL 2008 features are supported including source code encryption. Added support of new VPI types, including packed arrays of struct nets and variables.

4.3.2 ModelSim SE Features:

- Multi-language, high performance simulation engine
- Verilog, VHDL, System Verilog Design
- Code Coverage
- System Verilog for Design
- Integrated debug
- Job Spy Regression Monitor
- Mixed HDL simulation option
- System C Option
- TCL / tk
- Solaris and Linux 32 & 64-bit
- Windows 32-bit

4.3.3 ModelSim SE Benefits:

- High performance HDL simulation solution for FPGA & ASIC design teams
- The best mixed-language environment and performance in the industry
- Intuitive GUI for efficient interactive or post-simulation debug of RTL and gate-level designs
- Merging, ranking and reporting of code coverage for tracking verification progress
- Sign-off support for popular ASIC libraries

- All ModelSim products are 100% standards based. This means your investment is protected, risk is lowered, reuse is enabled, and productivity is enhanced
- Award-winning technical support

High-Performance, Scalable Simulation Environment:

ModelSim provides seamless, scalable performance and capabilities. Through the use of a single compiler and library system for all ModelSim configurations, employing the right ModelSim configuration for project needs is as simple as pointing your environment to the appropriate installation directory.

ModelSim also supports very fast time-to-simulation turnarounds while maintaining high performance with its new black box use model, known as box. With box, non-changing elements can be compiled and optimized once and reused when running a modified version of the test bench. bbox delivers dramatic throughput improvements of up to 3X when running a large suite of test cases.

Easy-to-Use Simulation Environment:

An intelligently engineered graphical user interface (GUI) efficiently displays design data for analysis and debug. The default configuration of windows and information is designed to meet the needs of most users. However, the flexibility of the ModelSim SE GUI allows users to easily customize it to their preferences. The result is a feature-rich GUI that is easy to use and quickly mastered.

A message viewer enables simulation messages to be logged to the ModelSim results file in addition to the standard transcript file. The GUI's organizational and filtering capabilities allow design and simulation information

to be quickly reduced to focus on areas of interest, such as possible causes of design bugs.

ModelSim SE allows many debug and analysis capabilities to be employed post-simulation on saved results, as well as during live simulation runs. For example, the coverage viewer analyzes and annotates source code with code coverage results, including FSM state and transition, statement, expression, branch, and toggle coverage. Signal values can be annotated in the source window and viewed in the waveform viewer. Race conditions, delta, and event activity can be analyzed in the list and wave windows. User-defined enumeration values can be easily defined for quicker understanding of simulation results. For improved debug productivity, ModelSim also has graphical and textual dataflow capabilities. The memory window identifies memories in the design and accommodates flexible viewing and modification of the memory contents. Powerful search, fill, load, and save functionalities are supported. The memory window allows memories to be pre-loaded with specific or randomly generated values, saving the time-consuming step of initializing sections of the simulation merely to load memories. All functions are available via the command line, so they can be used in scripting.

Advanced Code Coverage

The ModelSim advanced code coverage capabilities deliver high performance with ease of use. Most simulation optimizations remain enabled with code coverage. Code coverage metrics can be reported by-instance or by-design unit, providing flexibility in managing coverage data. All coverage information is now stored in the Unified Coverage Database (UCDB), which is used to collect and manage all coverage information in one highly efficient database. Coverage utilities that analyze code coverage data, such as merging and test ranking, are available.

The coverage types supported include:

- Statement coverage: number of statements executed during a run
- Branch coverage: expressions and case statements that affect the control flow of the HDL execution
- Condition coverage: breaks down the condition on a branch into elements that make the result true or false
- Expression coverage: the same as condition coverage, but covers concurrent signal assignments instead of branch decisions
- Focused expression coverage: presents expression coverage data in a manner that accounts for each independent input to the expression in determining coverage results
- Enhanced toggle coverage: in default mode, counts low-to-high and high-to-low transitions; in extended mode, counts transitions to and from X
- Finite State Machine coverage: state and state transition coverage

4.4 XILINX ISE

INTRODUCTION

For two-and-a-half decades, Xilinx has been at the forefront of the programmable logic revolution, with the invention and continued migration of FPGA platform technology. During that time, the role of the FPGA has evolved from a vehicle for prototyping and glue-logic to a highly flexible alternative to ASICs and ASSPs for a host of applications and markets. Today, Xilinx® FPGAs have become strategically essential to world-class system companies that are hoping to survive and compete in these times of extreme global economic instability, turning what was once the programmable revolution into the “programmable imperative” for both Xilinx and our customers.

4.4.1 Programmable Imperative

When viewed from the customer's perspective, the programmable imperative is the necessity to do more with less, to remove risk wherever possible, and to differentiate in order to survive. In essence, it is the quest to simultaneously satisfy the conflicting demands created by ever-evolving product requirements (i.e., cost, power, performance, and density) and mounting business challenges (i.e., shrinking market windows, fickle market demands, capped engineering budgets, escalating ASIC and ASSP non-recurring engineering costs, spiralling complexity, and increased risk). To Xilinx, the programmable imperative represents a two-fold commitment. The first is to continue developing programmable silicon innovations at every process node that deliver industry-leading value for every key figure of merit against which FPGAs are measured: price, power, performance, density, features, and programmability. The second commitment is to provide customers with simpler, smarter, and more strategically viable design platforms for the creation of world-class FPGA-based solutions in a wide variety of industries—what Xilinx calls targeted design platforms.



Fig 4.2 Xilinx

4.4.2 Base Platform

The base platform is both the delivery vehicle for all new silicon offerings from Xilinx and the foundation upon which all Xilinx targeted design platforms are built. As such, it is the most fundamental platform used to develop and run customer-specific software applications and hardware designs as production system solutions. Released at launch, the base platform comprises a robust set of well-integrated, tested, and targeted elements that enable customers to immediately start a design. These elements include:

- FPGA silicon
- ISE® Design Suite design environment
- Third-party synthesis, simulation, and signal integrity tools
- Reference designs common to many applications, such as memory interface and configuration designs.
- Development boards that run the reference designs
- A host of widely used IP, such as GigE, Ethernet, memory controllers, and PCIe.

4.4.3 Domain-Specific Platform

The next layer in the targeted design platform hierarchy is the domain-specific platform. Released from three to six months after the base platform, each domain specific platform targets one of the three primary Xilinx FPGA user profiles (domains): the embedded processing developer, the digital signal processing (DSP) developer, or the logic/connectivity developer. This is where the real power and intent of the targeted design platform begins to emerge. Domain-specific platforms augment the base platform with a predictable, reliable, and intelligently targeted set of integrated technologies, including:

- Higher-level design methodologies and tools
- Domain-specific embedded, DSP, and connectivity IP

- Domain-specific development hardware and daughter cards
- Reference designs optimized for embedded processing, connectivity, and DSP
- Operating systems (required for embedded processing) and software

Every element in these platforms is tested, targeted, and supported by Xilinx and/or our ecosystem partners. Starting a design with the appropriate domain-specific platform can cut weeks, if not months, off of the user's development time.

4.4.4 Market-Specific Platform

A market-specific platform is an integrated combination of technologies that enables software or hardware developers to quickly build and then run their specific application or solution. Built for use in specific markets such as Automotive, Consumer, Mil/Aero, Communications, AVB, or ISM, market-specific platforms integrate both the base and domain-specific platforms and provide higher level elements that can be leveraged by customer-specific software and hardware designs. The market-specific platform can rely more heavily on third-party targeted IP than the base or domain-specific platforms. The market-specific platform includes: the base and domain-specific platforms, reference designs, and boards (or daughter cards) to run reference designs that are optimized for a particular market (e.g., lane departure early-warning systems, analytics, and display processing). Xilinx will begin releasing market-specific platforms three to six months after the domain-specific platforms, augmenting the domain-specific platforms with reference designs, IP, and software aimed at key growth markets. Initially, Xilinx will target markets such as Communications, Automotive, Video, and Displays with platform elements that abstract away the more mundane portions of the design, thereby further reducing the customer's development effort so they can focus their attention on creating differentiated value in their end solution. This systematic platform

development and release strategy provides the framework for the consistent and efficient fulfillment of the programmable imperative—both by Xilinx and by its customers.

4.4.5 Platform Enablers

Xilinx has instituted a number of changes and enhancements that have contributed substantially to the feasibility and viability of the targeted design platform. These platform-enabling changes cover six primary areas:

- Design environment enhancements
- Socket able IP creation
- New targeted reference designs
- Scalable unified board and kit strategy
- Ecosystem expansion
- Design services supporting the targeted design platform approach

Design Environment Enhancements

With the breadth of advances and capabilities that the Virtex®-6 and Spartan®-6 programmable devices deliver coupled with the access provided by the associated targeted design platforms, it is no longer feasible for one design flow or environment to fit every designer's needs. System designers, algorithm designers, SW coders, and logic designers each represent a different user-profile, with unique requirements for a design methodology and associated design environment. Instead of addressing the problem in terms of individual fixed tools, Xilinx targets the required or preferred methodology for each user, to address their specific needs with the appropriate design flow. At this level, the design language changes from HDL (VHDL/Verilog) to C, C++, MATLAB® software, and other higher level languages which are more widely used by these designers, and the design abstraction moves up from the block or component to the system level. The result is a methodology and complete design flow tailored

to each user profile that provides design creation, design implementation, and design verification. Indicative of the complexity of the problem, to fully understand the user profile of a “logic designer,” one must consider the various levels of expertise represented by this demographic. The most basic category in this profile is the “push-button user” who wants to complete a design with minimum work or knowledge.

The push-button user just needs “good-enough” results. Contrastingly, more advanced users want some level of interactive capabilities to squeeze more value into their design, and the “power user” (the expert) wants full control over a vast array of variables. Add the traditional ASIC designers, tasked with migrating their designs to an FPGA (a growing trend, given the intolerable costs and risks posed by ASIC development these days), and clearly the imperative facing Xilinx is to offer targeted flows and tools that support each user's requirements and capabilities, on their terms. The most recent release of the ISE Design Suite includes numerous changes that fulfil requirements specifically pertinent to the targeted design platform. The new release features a complete tool chain for each top-level user profile (the domain-specific personas: the embedded, DSP, and logic/connectivity designers), including specific accommodations for everyone from the push-button user to the ASIC designer.

The tighter integration of embedded and DSP flows enables more seamless integration of designs that contain embedded, DSP, IP, and user blocks in one system. To further enhance productivity and help customers better manage the complexity of their designs, the new ISE Design Suite enables designers to target area, performance, or power by simply selecting a design goal in the setup. The tools then apply specific optimizations to help meet the design goal. In addition, the ISE Design Suite boasts substantially faster place-and-route and simulation run times, providing users with 2X faster compile

times. Finally, Xilinx has adopted the FLEX net Licensing strategy that provides a floating license to track and monitor usage.

4.4.6 XILINX ISE Design Tools:

Xilinx ISE is the design tool provided by Xilinx. Xilinx would be virtually identical for our purposes.

There are four fundamental steps in all digital logic design. These consist of:

1. Design – The schematic or code that describes the circuit.
2. Synthesis – The intermediate conversion of human readable circuit description to FPGA code (EDIF) format. It involves syntax checking and combining of all these separate design files into a single file.
3. Place & Route– Where the layout of the circuit is finalized. This is the translation of the EDIF into logic gates on the FPGA.
4. Program – The FPGA is updated to reflect the design through the use of programming (.bit) files.

Test bench simulation is in the second step. As its name implies, it is used for testing the design by simulating the result of driving the inputs and observing the outputs to verify your design.

ISE has the capability to do a variety of different design methodologies including: Schematic Capture, Finite State Machine and Hardware Descriptive Language (VHD Lor Verilog).



Fig 4.3 FPGA Board

CHAPTER 5

SIMULATION & SYNTHESIS IMPLEMENTATION

5.1 SIMULATION IMPLEMENTAION:

Verilog HDL is a Hardware Description Language (HDL). A Hardware Description Language is a language used to describe a digital system, for example, a computer or a component of a computer. One may describe a digital system at several levels. For example, an HDL might describe the layout of the wires, resistors and transistors on an Integrated Circuit (IC) chip, i. e., the switch level. Or, it might describe the logical gates and flip flops in a digital system, i. e., the gate level. An even higher level describes the registers and the transfers of vectors of information between registers. This is called the Register Transfer Level (RTL). Verilog supports all of these levels. However, this handout focuses on only the portions of Verilog which support the RTL level.

5.2 VERILOG:

Verilog is one of the two major Hardware Description Languages (HDL) used by hardware designers in industry and academia. VHDL is the other one. The industry is currently split on which is better. Many feel that Verilog is easier to learn and use than VHDL. As one hardware designer puts it, “I hope the competition uses VHDL.” VHDL was made an IEEE Standard in 1987 and Verilog in 1995. Verilog is very C-like and liked by electrical and computer engineers as most learn the C language in college. VHDL is very most engineers have no experience. Verilog was introduced in 1985 by Gateway Design System Corporation, now a part of Cadence Design Systems, Inc.’s Systems Division. Until May, 1990, with the formation of Open Verilog International (OVI),

Verilog HDL was a proprietary language of Cadence. Cadence was motivated to open the language to the Public Domain with the expectation that the market for Verilog HDL-related software products would grow more rapidly with broader acceptance of the language. Cadence realized that Verilog HDL users wanted other software and service companies to embrace the language and develop Verilog-supported design tools.



Fig 5.1 VERILOG – Coding language

5.3 SIMULATION RESULT:

5.3.1 AATA Simulation:

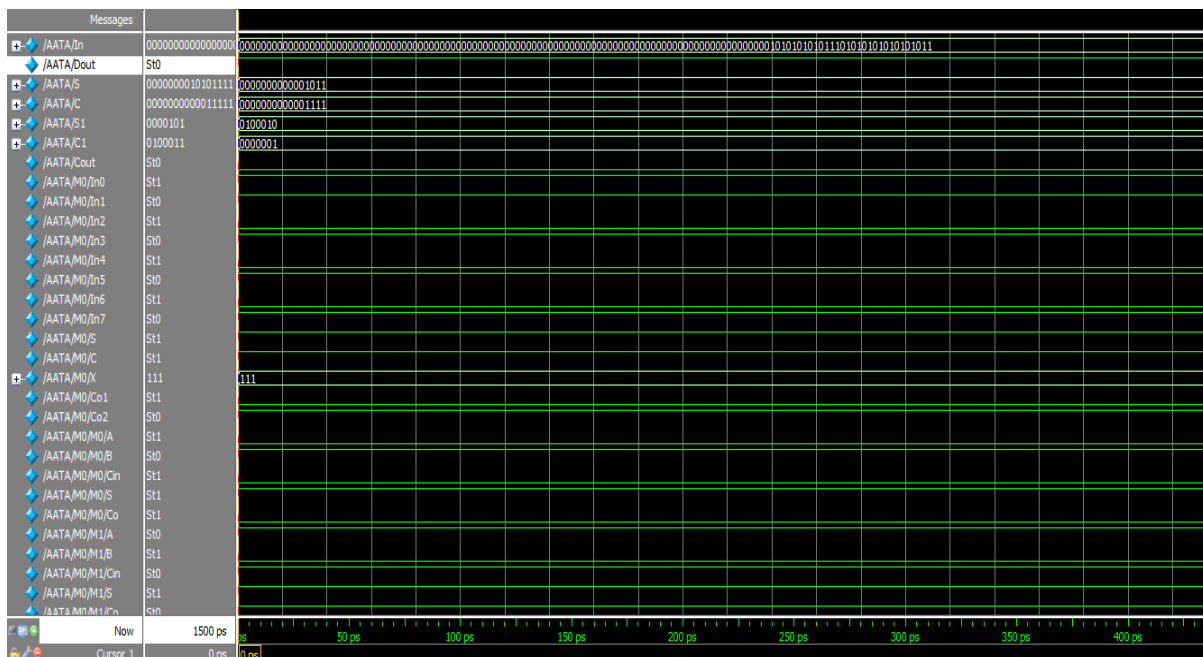


Fig 5.2 a) AATA Simulation Input

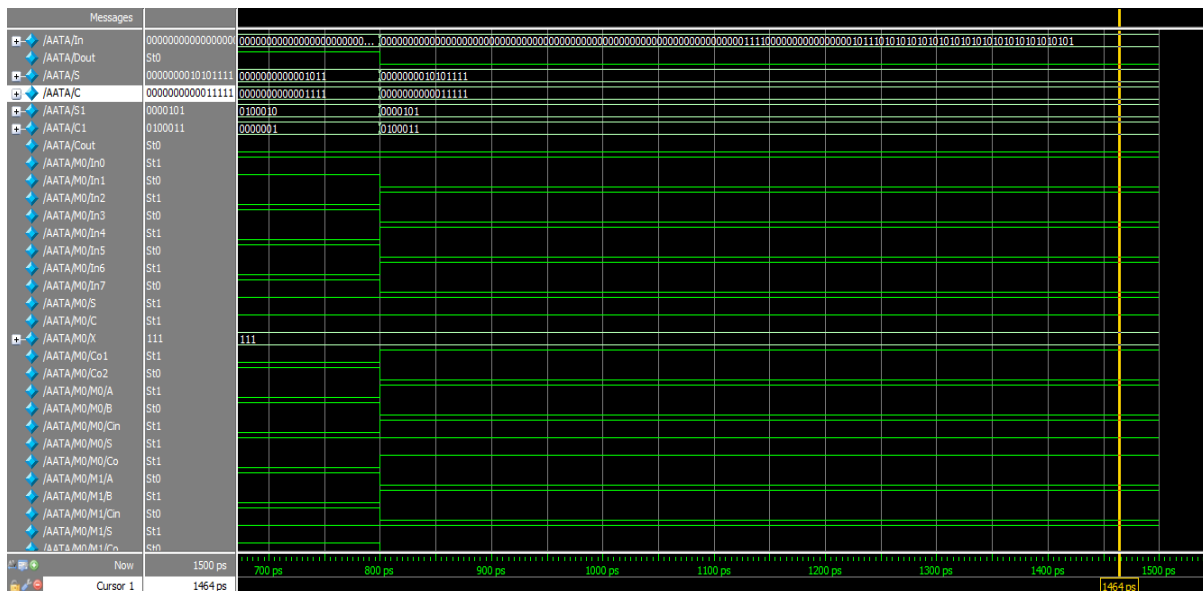


Fig 5.2 b) AATA Simulation Output

5.3.2 SRAM Block:

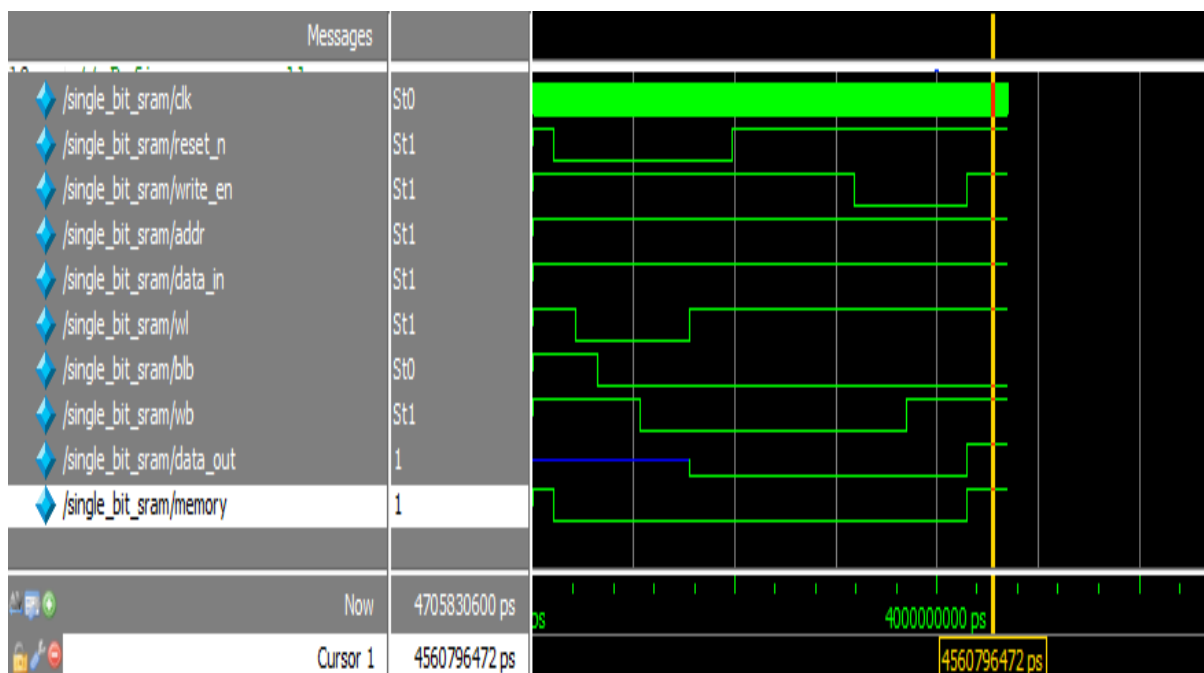


Fig: 5.3 a) SRAM Block Memory

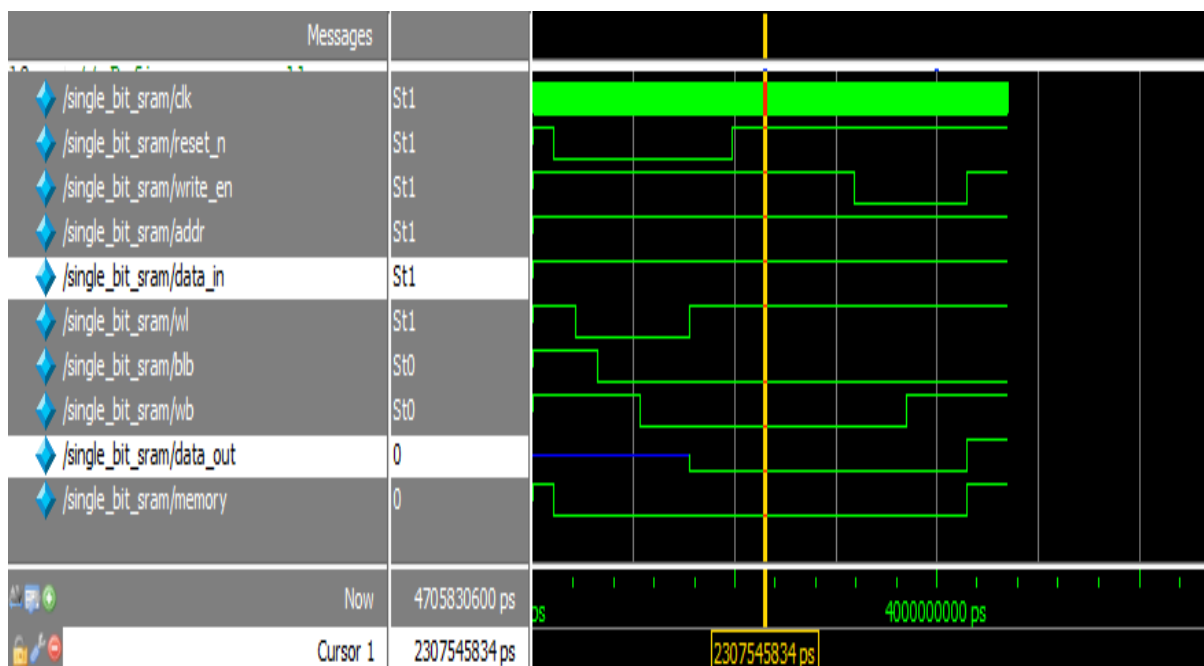


Fig: 5.3 b) SRAM Block Input & Output

[illegible]

Messages			
▶ /CIM_Macro_Cell/Ck	St1		
▶ /CIM_Macro_Cell/Rst	St1		
▶ /CIM_Macro_Cell/WE	St1		
▶ /CIM_Macro_Cell/BL	St1		
▶ /CIM_Macro_Cell/BLB	S10		
▶ /CIM_Macro_Cell/Addr	St1		
▶ /CIM_Macro_Cell/ML	1111111111111111		
▶ /CIM_Macro_Cell/in_B	0000000000000000		
▶ /CIM_Macro_Cell/inb	St1		
▶ /CIM_Macro_Cell/Dout	St1		
▶ /CIM_Macro_Cell/D	0000000000000000		
▶ /CIM_Macro_Cell/M0/ck	St1		
▶ /CIM_Macro_Cell/M0/reset_n	St1		
▶ /CIM_Macro_Cell/M0/write_en	St1		
▶ /CIM_Macro_Cell/M0/addr	St1		
▶ /CIM_Macro_Cell/M0/data_in	St1		
▶ /CIM_Macro_Cell/M0/inl	St1		
▶ /CIM_Macro_Cell/M0/bb	S10		
▶ /CIM_Macro_Cell/M0/inb	St1		
▶ /CIM_Macro_Cell/M0/data_out	1		
▶ /CIM_Macro_Cell/M0/memory	1		
▶ /CIM_Macro_Cell/M1/ck	St1		
▶ /CIM_Macro_Cell/M1/reset_n	St1		
▶ /CIM_Macro_Cell/M1/write_en	St1		
▶ /CIM_Macro_Cell/M1/addr	St1		
▶ /CIM_Macro_Cell/M1/data_in	St1		
▶ /CIM_Macro_Cell/M1/inl	St1		
▶ /CIM_Macro_Cell/M1/bb	S10		
▶ /CIM_Macro_Cell/M1/inb	St1		
▶ /CIM_Macro_Cell/M1/data_out	1		
Now	13491700 ps		
		13490400 ps	13491600 ps

48

5.4 SYNTHESIS IMPLEMENTATION

5.4.1 DEVICE: NEW - SXAFA


Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of 4 input LUTs	3,366	66,560	5%	
Number of occupied Slices	1,724	33,280	5%	
Number of Slices containing only related logic	1,724	1,724	100%	
Number of Slices containing unrelated logic	0	1,724	0%	
Total Number of 4 input LUTs	3,366	66,560	5%	
Number of bonded IOBs	317	633	50%	
IOB Flip Flops	96			
Number of BUFGMUXs	1	8	12%	
Average Fanout of Non-Clock Nets	3.74			

Table No 5.1 NEW- SXAFA – Device Utilization Summary

From the table no 5.1, The design utilizes 5% of the available 4-input LUTs (3,366 out of 66,560) and 5% of the available slices (1,724 out of 33,280). All occupied slices contain only related logic, with no unrelated logic present. The number of bonded IOBs used is 317, accounting for 50% of the available 633. Additionally, 96 IOB flip-flops and 1 BUFGMUX (12% of 8 available) are used. The average fanout of non-clock nets is 3.74.

5.4.2 DELAY SUMMARY:

Category	Details
RTL Top Level Output File Name	Macro.ngr
Top Level Output File Name	Macro
Output Format	NGC
Optimization Goal	Speed
Keep Hierarchy	No

Table No 5.2 Delay Summary

From the table no 5.2, The RTL top-level output file is named Macro.ngr, and the top-level output file is Macro, generated in NGC format. The synthesis was optimized for speed, and hierarchy preservation was not enabled.

5.4.3 TIMING REPORT

Parameter	Value
Speed Grade	-4
Minimum period	No path found
Minimum input arrival time before clock	5.334 ns
Maximum output required time after clock	19.323 ns
Maximum combinational path delay	19.849 ns

Table No 5.3 Timing Summary

From the table no 5.3, The design was synthesized with a speed grade of -4. No minimum period path was identified. The minimum input arrival time before the clock is 5.334 ns, while the maximum output required time after the clock is 19.323 ns. The maximum combinational path delay observed is 19.849 ns.

5.4.4 TOTAL DELAY BREAKDOWN

Parameter	Value (ns)	Description
Total	5.334	(1.974 ns logic, 3.360 ns route)
Logic Contribution	1.974	37.0% logic
Route Contribution	3.360	63.0% route

Table No 5.4 Total Delay Breakdown

From the table no 5.4, The total delay is 5.334 ns, comprising 1.974 ns from logic (37.0%) and 3.360 ns from routing (63.0%). This indicates that routing contributes the majority of the delay in the current design implementation.

5.4.5 DEVICE: NEW - SOAFA

Device Utilization Summary				[-]
Logic Utilization	Used	Available	Utilization	Note(s)
Number of 4 input LUTs	5,268	66,560	7%	
Number of occupied Slices	2,736	33,280	8%	
Number of Slices containing only related logic	2,736	2,736	100%	
Number of Slices containing unrelated logic	0	2,736	0%	
Total Number of 4 input LUTs	5,268	66,560	7%	
Number of bonded IOBs	317	633	50%	
IOB Flip Flops	96			
Number of BUFGMUXs	1	8	12%	
Average Fanout of Non-Clock Nets	3.93			

Table No 5.5 NEW-SOAFA – Device Utilization Summary

From the table no 5.5, The design utilizes 7% of the available 4-input LUTs (5,268 out of 66,560) and 8% of the slices (2,736 out of 33,280). All occupied slices contain only related logic (100%) with no unrelated logic usage. 317 bonded IOBs are used, which is 50% of the available 633. Additionally, the design includes 96 IOB flip-flops, 1 BUFGMUX (12% of 8 available), and has an average fanout of non-clock nets of 3.93.

5.4.6 DELAY SUMMARY:

Description	Details
RTL Top Level Output File Name	Macro.ngr
Top Level Output File Name	Macro
Output Format	NGC
Optimization Goal	Speed
Keep Hierarchy	No

Table No 5.6 Delay Summary

From the table no 5.6, The RTL top-level output file is named Macro.ngr, and the corresponding top-level output file is Macro, generated in the NGC format. The design was synthesized with an optimization goal of speed, and hierarchy preservation was disabled during the process.

5.4.7 TIMING REPORT

Parameter	Value
Speed Grade	-4
Minimum Period	No path found
Minimum Input Arrival Time Before Clock	5.334 ns
Maximum Output Required Time After Clock	19.323 ns
Maximum Combinational Path Delay	19.849 ns

Table No 5.7 Timing Summary

From the table no 5.7, The design targets a speed grade of -4. No valid path was identified for the minimum period. The minimum input arrival time before the clock is 5.334 ns, the maximum output required time after the clock is 19.323 ns, and the maximum combinational path delay is 19.849 ns.

5.4.8 Total Delay Breakdown

Total Delay	1.974ns (Logic)	3.360ns (Route)
Total Delay	5.334ns	-
Percentage Logic	37.0%	-
Percentage Route	63.0%	-

Table No 5.8 Total Delay Breakdown

From the table no 5.8, The total delay in the design is 5.334 ns, composed of 1.974 ns from logic (accounting for 37.0%) and 3.360 ns from routing (accounting for 63.0%). This indicates that the majority of the delay is contributed by routing resources.

5.5 RTL SCHEMATIC:

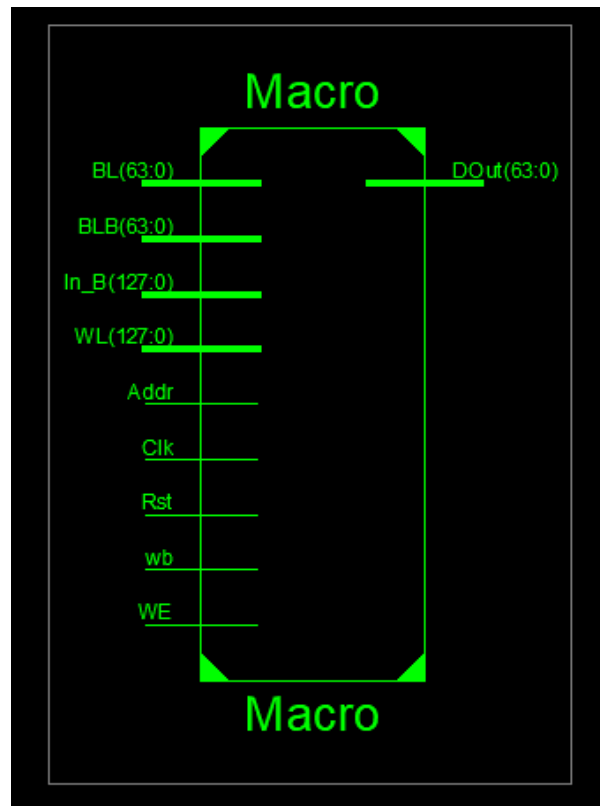


Fig 5.5 Systematic diagram

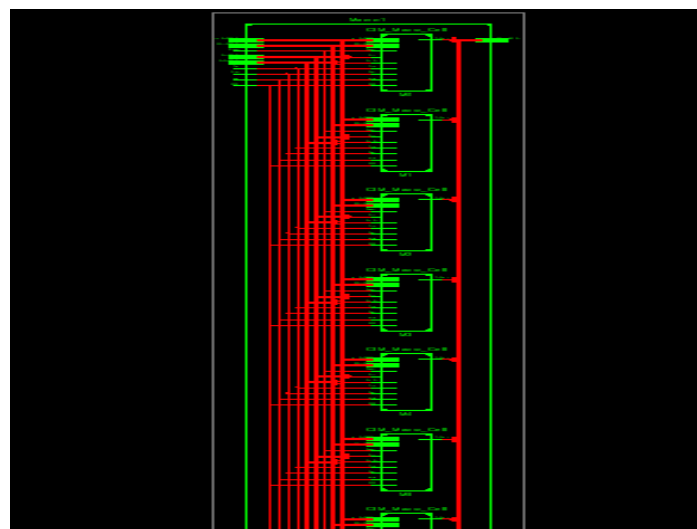


Fig 5.6 RTL Schematic

5.6 TECHNOLOGY SCHEMATIC:

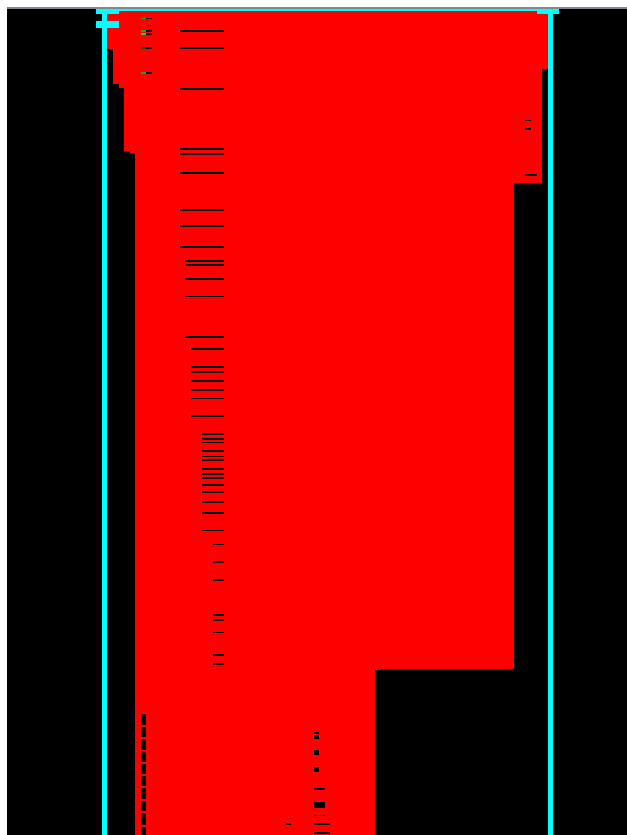


Fig 5.7 Technology Schematic

Parameter	Value
Total REAL time to Xst completion	27.00 secs
Total CPU time to Xst completion	27.01 secs
Total memory usage	5,127,856 kilobytes
Number of errors	0
Number of warnings	2097
Number of infos	128

Table No 5.9 Synthesis Report

From the table no 5.9, The synthesis process completed successfully with no errors but had 2,097 warnings. It took 27 seconds of real and CPU time, utilizing around 5 GB of memory.

CHAPTER 6

RESULTS AND DISCUSSION

6.1 COMPARISON CHART:

	Area			Delay		
	LUT	Slices	IOB	Overall Delay	Gate Delay	Path Delay
MACRO Design based on SXAFA	3366	1724	317	19.849ns	9.771ns	10.078ns
MACRO Design based on SOAFA	5268	2736	317	20.287ns	10.322ns	9.965ns

Table No 6.1 Area and Delay comparison of SXAFA & SOAFA

From the table no 6.1, provides a comparative analysis of two MACRO designs implemented using SXAFA and SOAFA techniques, evaluated across area utilization and timing performance.

1. Area Comparison

- **LUT-Usage:**
SXAFA-based design uses 3366 LUTs, which is significantly lower than SOAFA's 5268 LUTs, suggesting that SXAFA is more area-efficient in logic element consumption.
- **Slices:**
SXAFA requires 1724 slices, while SOAFA uses 2736, confirming a smaller silicon footprint for SXAFA.
- **Input/Output-Blocks:**
Both designs use 317 IOBs, indicating no difference in external pin requirements.

2. Delay (Performance) Comparison

- Overall-Delay:
SXAFa has a slightly better delay at 19.849ns versus 20.287ns for SOAFa.
- Gate-Delay:
Gate delay is lower in SXAFa (9.771ns) compared to SOAFa (10.322ns), which reflects more efficient logic-level processing.
- Path-Delay:
SOAFa shows a slightly better path delay (9.965ns) compared to SXAFa (10.078ns), but this gain is minor and does not outweigh the other advantages.

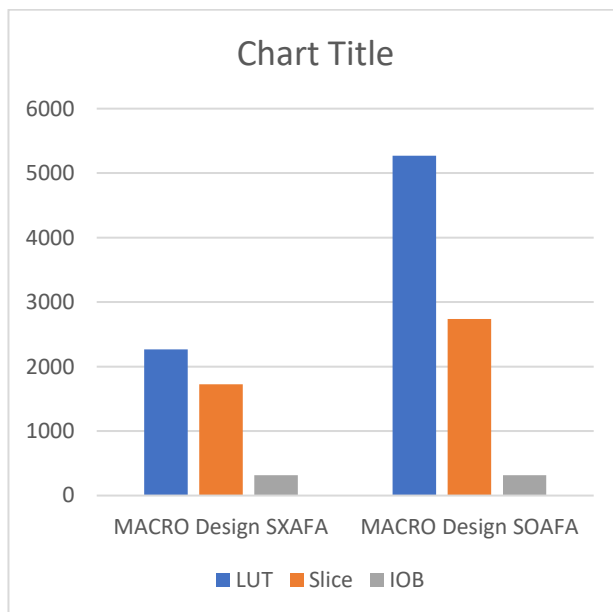


Fig 6.1 Macro Design

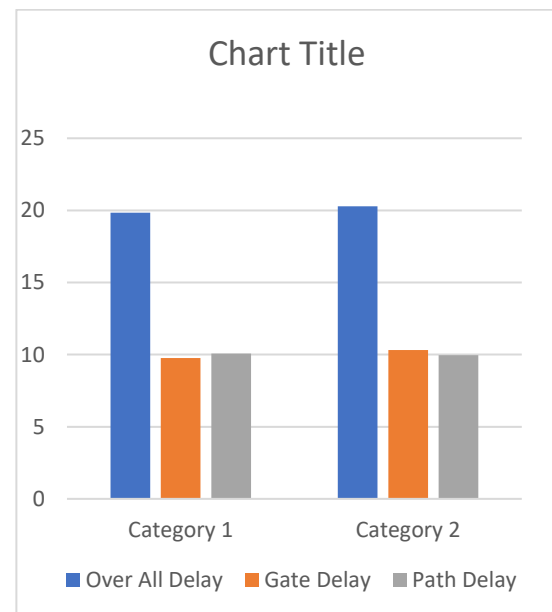


Fig 6.2 Delay Chart

From the Figure 6.1 and 6.2, The comparison shows that MACRO Design SXAFa uses significantly fewer hardware resources (LUTs and Slices) compared to SOAFa, indicating better resource efficiency. However, both designs exhibit similar timing performance, with nearly equal overall, gate, and path delays, suggesting that SXAFa achieves comparable speed with lower resource usage.

POWER RESULT:

NEW SXAFA:

Power summary:	I(mA)	P(mW)
Total estimated power consumption:		21097
Vccint 1.20V:	4768	5722
Vccaux 2.50V:	70	175
Vcco25 2.50V:	6080	15200
Clocks:	42	50
Inputs:	914	1097
Logic:	1171	1405
Outputs:		
Vcco25	6080	15200
Signals:	2521	3026
Quiescent Vccint 1.20V:	120	144
Quiescent Vccaux 2.50V:	70	175

Table No 6.2 Power Result of NEW SXAFA

NEW SOAFA:

Power summary:	I(mA)	P(mW)
Total estimated power consumption:		36480
Vccint 1.20V:	4921	5905
Vccaux 2.50V:	70	175
Vcco25 2.50V:	12160	30400
Clocks:	42	50
Inputs:	914	1097
Logic:	1194	1433
Outputs:		
Vcco25	12160	30400
Signals:	2650	3180
Quiescent Vccint 1.20V:	120	144
Quiescent Vccaux 2.50V:	70	175

Table No 6.3 Power Result of NEW SOAFA

From the table no 6.2 and 6.3, The power analysis results reveal that the NEW SXAFA architecture consumes significantly less power compared to the NEW SOAFA. Specifically, the total estimated power consumption for NEW SXAFA is 21,097 mW, while NEW SOAFA consumes 36,480 mW. The major contributor to this difference is the Vcco25 supply, which accounts for 15,200 mW in SXAFA but rises sharply to 30,400 mW in SOAFA. Other components such as logic, inputs, and signals also show slightly higher consumption in SOAFA, though the gap is less pronounced. Overall, NEW SXAFA demonstrates a clear advantage in terms of power efficiency, consuming approximately 42% less total power than NEW SOAFA, making it a more suitable choice for low-power applications.

CHAPTER 7

CONCLUSION & FUTURE ENHANCEMENTS

7.1 CONCLUSION:

In this project, a comparative analysis between two MACRO designs — one based on the SXAFA (Skewed XOR AND Full Adder) approach and the other based on the SOAFA (Skewed OR AND Full Adder) approach — was carried out with respect to key performance parameters: **Area**, **Delay**, and **Power**. The results demonstrate that the MACRO design based on **SXAFA** outperforms the SOAFA-based MACRO design across all critical metrics. Specifically, the SXAFA-based design achieves a smaller silicon area footprint, lower delay, and reduced power consumption. These improvements make the SXAFA-based MACRO design a more efficient and optimized solution for high-performance and low-power Compute-in-Memory (CIM) accelerators, particularly suited for AI, machine learning, and other data-intensive applications. Hence, adopting SXAFA structures in MACRO design contributes significantly to achieving better overall system efficiency and scalability.

In this project, we have attained key **Program Outcomes (POs)** such as **PO1, PO2, PO3, PO4, PO5, PO7, PO8, PO9, PO10 and PO12**. Additionally, the project aligns with **Program Specific Outcomes (PSOs)** such as **PSO1, PSO3**.

7.2 FUTURE ENHANCEMENTS

Although the current Low Power Adder Tree design meets the intended goals, there are several areas where it can be improved to suit more complex and high-performance systems. Incorporating advanced power reduction techniques like clock gating, multi-voltage operation, and power gating can help further reduce dynamic and static power consumption. Introducing pipelining stages will help increase the overall throughput and make the design more suitable for high-speed applications.

Scalability can be enhanced by allowing the design to adapt dynamically to varying data widths and operational modes, making it more flexible for integration into larger systems. Post-synthesis improvements such as better floor planning and timing-driven optimizations during place-and-route can help reduce delays and improve timing performance.

For ASIC implementation, using low-power standard cells and adopting advanced CMOS scaling methods will help minimize power, area, and delay. Optimizing the critical path to shorten the longest delay path in the circuit will directly improve the clock frequency.

Further, enhancing verification through formal verification and advanced simulation environments can help ensure better functional correctness. Support for multiple clock domains and asynchronous interfacing will allow the design to be integrated into more diverse systems. Lastly, with the growing demand for efficient arithmetic units in AI and ML applications, the Adder Tree can be adapted to function as a key component in MAC units, enhancing its usefulness in neural network accelerators and signal processing hardware.

REFERENCES

- [1] A. Biswas et al., “CONV-SRAM: An energy-efficient SRAM with in memory dot-product computation for low-power convolutional neural networks,” *IEEE J. Solid-State Circuits*, vol. 54, no. 1, pp. 217–230, Jan. 2019.
- [2] Q. Dong et al., “15.3 A 351TOPS/W and 372.4GOPS compute-in memory SRAM macro in 7nm FinFET CMOS for machine-learning applications,” in *Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC)*, 2020, pp. 242–244.
- [3] M. Zhao et al., “Crossbar-level retention characterization in analog RRAM array-based computation-in-memory system,” *IEEE Trans. Electron Devices*, vol. 68, no. 8, pp. 3813–3818, Aug. 2021.
- [4] H. Valavi, P. J. Ramadge, E. Nestler, and N. Verma, “A 64-tile 2.4-Mb in-memory-computing CNN accelerator employing charge-domain compute,” *IEEE J. Solid-State Circuits*, vol. 54, no. 6, pp. 1789–1799, Jun. 2019.
- [5] Y. Kong, X. Chen, X. Si, and J. Yang, “Evaluation platform of time domain computing-in-memory circuits,” *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 70, no. 3, pp. 1174–1178, Mar. 2023.
- [6] A. Agrawal, M. Ali, M. Koo, N. Rathi, A. Jaiswal, and K. Roy, “IMPULSE: A 65-nm digital compute-in-memory macro with fused weights and membrane potential for spike-based sequential learning tasks,” *IEEE solid-State Circuits Lett.*, vol. 4, pp. 137–140, 2021.
- [7] H. Kim, T. Yoo, T. T.-H. Kim, and B. Kim, “Colonnade: A reconfigurable SRAM-based digital bit-serial compute-in-memory macro for processing neural networks,” *IEEE J. Solid-State Circuits*, vol. 56, no. 7, pp. 2221–2233, Jul. 2021.
- [8] Y.-D. Chih et al., “16.4 an 89TOPS/W and 16.3TOPS/mm² all-digital SRAM-based full-precision compute-in memory macro in 22nm for machine-learning edge applications,” in *Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC)*, 2021, pp. 252–254.
- [9] D. Wang, C.-T. Lin, G. K. Chen, P. Knag, R. K. Krishnamurthy, and M. Seok, “DIMC: 2219TOPS/W 2569F2/b digital in-memory computing macro in 28nm based on approximate arithmetic hardware,” in *Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC)*, 2022, pp. 266–268.

- [10] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, “Low-power digital signal processing using approximate adders,” *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 32, no. 1, pp. 124–137, Jan. 2013.
- [11] S. E. Fatemieh, S. S. Farahani, and M. R. Reshadinezhad, “LAHAF: Low-power, area-efficient, and high-performance approximate full adder based on static CMOS,” *Sustain. Comput. Informat. Syst.*, vol. 30, pp. 1–12, Jun. 2021.
- [12] S. Salavati, M. H. Moaiyeri, and K. Jafari, “Ultra-efficient nonvolatile approximate full-adder with spin-hall-assisted MTJ cells for in-memory computing applications,” *IEEE Trans. Magn.*, vol. 57, no. 5, pp. 1–11, May 2021.
- [13] Y. He et al., “An RRAM-based digital computing-in-memory macro with dynamic voltage sense amplifier and sparse-aware approximate adder tree,” *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 70, no. 2, pp. 416–420, Feb. 2023.
- [14] F. Tu et al., “SDP: Co-designing algorithm, dataflow, and architecture for in-SRAM sparse NN acceleration,” *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 42, no. 1, pp. 109–121, Jan. 2023.
- [15] J. Yue et al., “STICKER-IM: A 65 nm computing-in-memory NN processor using block-wise sparsity optimization and inter/intra-macro data reuse,” *IEEE J. Solid-State Circuits*, vol. 57, no. 8, pp. 2560–2573, Aug. 2022.
- [16] T.-H. Yang et al., “Sparse ReRAM engine: Joint exploration of activation and weight sparsity in compressed neural networks,” in *Proc. ACM/IEEE 46th Annu. Int. Symp. Comput. Architect. (ISCA)*, 2019, pp. 236–249.
- [17] W. Liu, T. Zhang, E. McLarnon, M. O’Neill, P. Montuschi, and F. Lombardi, “Design and analysis of majority logic-based approximate adders and multipliers,” *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 3, pp. 1609–1624, Jul.–Sep. 2021.
- [18] F. Tu et al., “ReDCIM: Reconfigurable digital computing-in-memory processor with unified FP/INT pipeline for cloud AI acceleration,” *IEEE J. Solid-State Circuits*, vol. 58, no. 1, pp. 243–255, Jan. 2023.