M.Tech. Thesis

In Memory Computation Based Boolean and Logic Circuit

RAM LAKHAN

May 2025

In Memory Computation Based Boolean and Logic Circuit Design

A Thesis Submitted

In Partial Fulfillment of the Requirements for the

Degree of

MASTER OF TECHNOLOGY

in

VLSI DESIGN AND EMBEDDED SYSTEM

Submitted by

RAM LAKHAN

(2K23/VLS/16)

Under the supervisor of

Assistant Prof. AKSHAY MANN

EPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi college of Engineering)

Shahbad Daulatpur, Main Bawana Road, Delhi-110042

May, 2025

ELECTRONICS AND COMMUNICATION ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi college of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-110042

ACKNOWLEDGEMENT

To begin with, I am incredibly grateful to God Almighty for enabling me to surpass my expectations and finish this project on schedule.

I express my sincere gratitude to my mentors at Delhi Technological University for their tremendous support, motivation, and direction during the project work. They have inspired me to work hard and accomplish the task at hand with utmost attention. Without their constant support and guidance, I would not have been able to attempt this project.

Place: Delhi **RAM LAKHAN**
Date: 31st may 2025 **(2K23/VLS/16)**

ELECTRONICS AND COMMUNICATION ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi college of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

I **RAM LAKHAN** Roll no: **2K23/VLS/16** hereby certify that the work which is being presented in the thesis entitled **In Memory Computation Based Boolean and Logic Circuit Design** in partial fulfillment of the requirements for the award of the Degree of **Master of Technology**, submitted in the Department of **Electronics and Communication,** Delhi Technological University is an authentic record of my own work carried out during the period from June 2024 to May 2025 under the supervision of **Assistant Prof AKSHAY MAAN** The matter presented in the thesis has not been submitted by me for the award of any other degree of this or any other Institute.

Place: Delhi — **RAM LAKHAN**

Date: 31st may 2025 — **(2K23/VLS/16)**

ELECTRONICS AND COMMUNICATION ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi college of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-110042

CERTIFICATE BY THE SUPERVISOR(s)

Certified that **RAM LAKHAN** Roll no : **2K23/VLS/16** has carried out their search work presented in this thesis entitled **In Memory Computation Based Boolean and Logic Circuit Design** the degree of **Master of Technology** from Department of Electronics and Communication, Delhi Technological University, Delhi, under my supervision. The thesis embodies results of original work, and studies are carried out by the student himself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Place: Delhi — **Assistant Prof. AKSHAY MAAN**

Date: 31st may2025 — **SUPERVISOR**

ABSTRACT

Data organisation is a very important task today. As we already know data stores digitally so there comes issue of accessing time. Today electronics system and gadgets are increasing very fast so data also increasing. Because of more data there becomes accessing of data rate slow. Today we are working with AI, machine learning, live project, live location, satellites, antenna, radio system many things are totally based on data speed. We cannot compromise with latency for a critical work. This is possible that we can make our memory system fast as well implementation of some basic operation inside memory only. In-Memory Computing is a technique, in which we store our data in RAM rather than slow server or disk. As our all system based on Von Neumann architecture and this architecture having both power and memory wall. This cases a bottleneck on performance of system. In memory computing is a good way to break this bottleneck and make our system to free from memory and power wall. This project intends to implementing Boolean and logic circuit design using In-Memory computing technique. This project is combination of two things, first one to select a proper SRAM (Static Random-Access Memory) and second is, with help of in-memory computing scheme implementation of Boolean and logic circuit. In this work we are presented what is SRAM and what is issue with 6T SRAM, 8T SRAM and 9T SRAM. Here also presented in-memory computation technology. The Boolean logic operation are demonstrated with 9T SRAM cell in 250 nm, 90nm and 22nm CMOS technology with help of Tanner and Cadence tool. The NAND, AND, NOR, OR, EXOR, EXNOR Boolean logics as well some combinational circuits are demonstrate using 9T SRAM cells with the proposed sensing scheme to verifying the In-Memory computations ability of 9T SRAM.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATION

Abbreviation	Full Form
SRAM	Static Random Access Memory
9T	9-Transistor
6T	6-Transistor
8T	8-Transistor
VDD	Supply Voltage
EXOR	Exclusive OR (XOR)
RBL	Read Bit Line

CHAPTER-1
INTRODUCTION

The swift growth of the Internet of Things, along with the advent of advanced computing systems, has paved the way for its broad adoption. It is commonly understood that the primary hurdle to achieving the transformative goal of universal access to computing resources is the issue of energy consumption, which limits the extent, scope, and durability of many computing systems. Over the past decade, there has been a sharp increase in the demand for ultra-efficient computing systems that require robust, low-power processing cores. This demand becomes more urgent due to the growing need to scale computation to handle key tasks such as deep learning and artificial intelligence, which require efficient large-scale data processing. A crucial factor in realizing this vision is the availability of computing power that can process vast amounts of data quickly, reliably, and with minimal energy consumption. Conversely, the research community speculates that CMOS scaling might reach its limits around 2024 [1], making it unlikely that further improvements in performance, size, and power will be primarily based on fabrication technology. There is a general consensus among researchers that computing based on traditional architectures is nearing its limits in terms of scalability and energy efficiency [1, 2]. Due to these challenges, research has begun to explore new computing systems.

1.1 Historical background

The Internet of Things (IoT) is anticipated to connect billions of devices, leading to the generation of an enormous volume of raw data. The number of internet-connected devices and the data they produce is increasing every day. In 2016, this data reached 16.1 zettabytes (ZB), and it is forecasted to grow tenfold by 2025 [3]. In recent years, the number of smart electronic devices has surpassed the global human population [4], as depicted in Figure 1.1. Various algorithms have been developed to preprocess and compress large datasets [5]. Traditionally, these algorithms have been executed on standard general-purpose processors. However, conventional processing architectures face challenges with large datasets due to the significant data transfer between the main memory and processing cores [6].

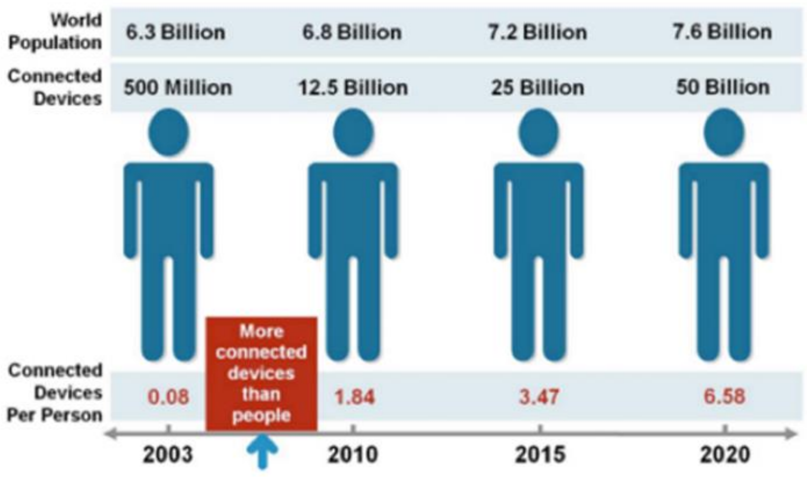
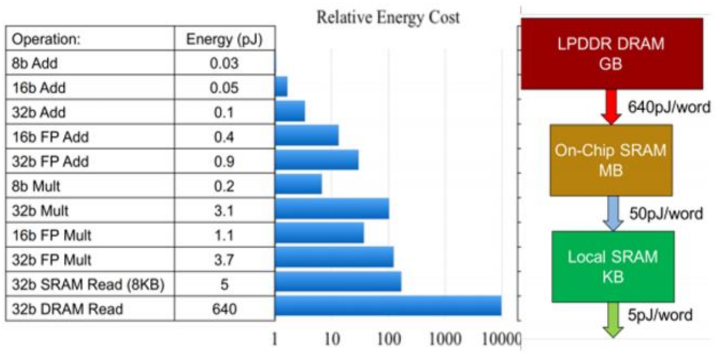


Figure 1.1: Increase in number of connected devices over the years [4]

Figure 1.2 shows that the DRAM read operation is the most energy-consuming task in a device. A 32-bit read operation consumes 170 times more energy than a 32-bit floating-point multiplication [7]. This inefficiency arises from the limited on-chip cache memories in conventional cores and the restricted memory bandwidth of the main memory. For instance, executing the k-nearest algorithm requires calculating the distance of each data point with all other points in the dataset [8]. To process 2 billion candidate points, a single query demands 150 GFLOPs of computation and 500G of data transfer [7]. This results in significant performance overhead when the data cannot fit into memory.



Operation:	Energy (pJ)
8b Add	0.03
16b Add	0.05
32b Add	0.1
16b FP Add	0.4
32b FP Add	0.9
8b Mult	0.2
32b Mult	3.1
16b FP Mult	1.1
32b FP Mult	3.7
32b SRAM Read (8KB)	5
32b DRAM Read	640

Figure 1.2: The cost of various operations in a typical computer organization. Reading data from DRAM is the most energy consuming operation [9]

1.2 Basics of In-Memory Computing

In-memory computing (IMC) and near-data computing (NDC) are two effective strategies designed to reduce the costs associated with data movement [10]. NDC positions the computing system close to the main memory to eliminate the expenses of data transfer during processing [11]. Conversely, IMC leverages the analog properties of emerging memory technologies to perform computations directly within the memory itself. The primary aim of both approaches is to execute computations on data where it is stored or nearby. By avoiding the transfer of all data from memory to processing units, they can significantly reduce data communication and its associated high costs. However, these methods come with their own challenges. The application of NDC might be constrained by the complex integration of memory and logic and the cost of large computing cores, while processor-in-memory could face limitations due to slow device latency, low endurance, and limited computing capabilities [12].

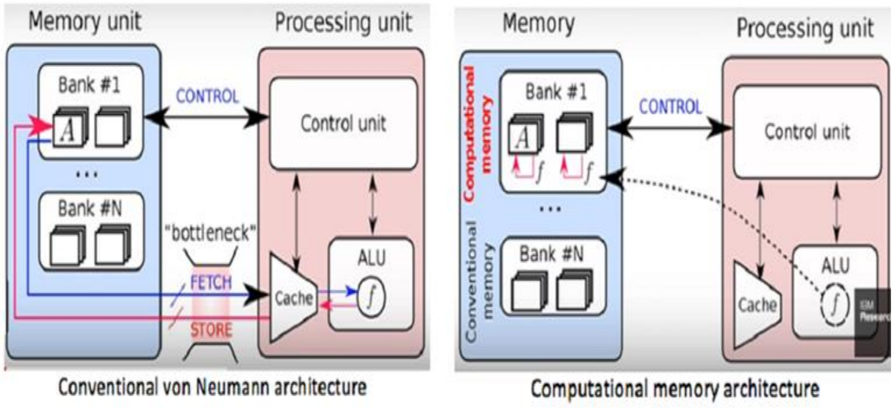


Figure 1.1: Conventional Memory and In-Memory Computing [11]

1.3 Overview of Memory

Our current computer architecture is fundamentally built on the von Neumann framework. This architecture includes a central processing unit (CPU) responsible for executing arithmetic, logical, and control operations, a main memory that stores both instructions and data, and an external memory that offers additional storage capacity and supports the main memory. In modern computer systems, the memory hierarchy typically identifies the hard disk as secondary storage, while Random-Access Memory (RAM) is recognized as primary storage, often simply called memory.

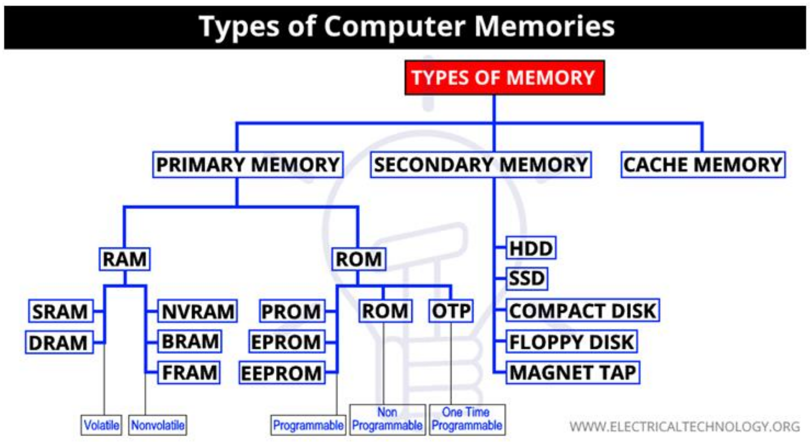


Figure 1.2: Types of Memory [13]

Because RAM is expensive, it mostly determines a system's cost. RAM requires less than 20 times as much access time as secondary storage. Because the CPU is becoming faster every day but the storage is not, this delay led to problems with memory casing and speed synchronization. This delay makes it possible to employ RAM for data storage in a favorable way.

1.4 Need of In-Memory computing

Transfers between the processor and memory units need a lot of power and restrict throughput for quick statistics [14]. By reducing the number of pointless statistics transfers, computing in the memory array enhances the reminiscence capability. It is anticipated that this will result in significant improvements in both energy efficiency and throughput [15].

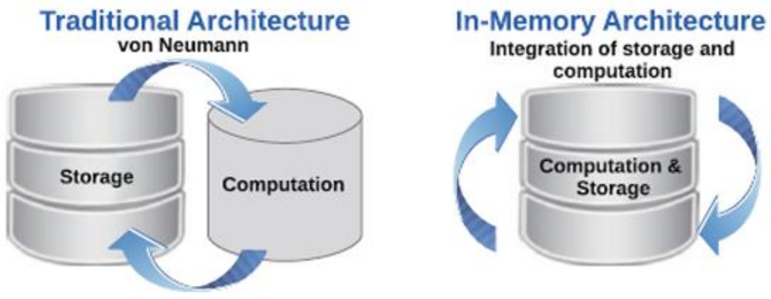


Figure 1.3: Von Neumann V/S In-Memory Architecture [11]

1.5 Motivation

In-Memory computing presents a number of difficulties. This is a developing topic since we want to obtain data quickly and efficiently. Instead of employing another type of memory storage, we are using RAM to store data. The processing that takes place within memory is sophisticated. However, IMC is an excellent way to get speed and quick access. IMC is a challenging operation that requires us to compute inside the memory while also taking into account other parameters like area, power, speed, size, cell functioning, current, voltage technology, and the creation of computational logic inside memory. Thus, this serves as a strong incentive to concentrate on SRAM design constraints, transistor scaling, logic implementation, computation, memory's advantages and disadvantages, and other future-related topics.

1.6 Objective and Project Outline

The aim of this project is to develop Boolean and Logical circuits utilizing the In-Memory Computing Technique. The focus is on creating fundamental logic gates with 9T SRAM through this approach and evaluating the circuit's robustness by adjusting process corners, temperature, and voltage within a defined range. The project aspires to enhance the In-Memory Computing technique by employing SRAM beyond the traditional 6T and 8T models, as well as to construct Logical circuits and analyze the impact of worst-case process variations on circuit and transistor parameters. We are applying basic methods of logic gates and digital logic theory for In-Memory Computing. The circuit implementation will be based on 9T SRAM. Our primary approach involves using basic digital logic theory and a sensing scheme to build the Boolean circuit. By leveraging Boolean logic gates and a universal gate, we will create some combinational circuits.

1.7 Organization of Report

- Chapter 2 provides a brief overview of the literature related to SRAM specifications, the concept of In-Memory computing, and the techniques used for its implementation.
- Chapter 3 focuses on the design of cells and the basic principles underlying SRAM functionality.
- Chapter 4 explores the idea of In-Memory Computing and presents a strategy for implementing logic circuits through this method.
- Chapter 5 explains the reasons for selecting 9T SRAM and details the project's implementation, including the demonstration of results. This chapter covers the creation of Boolean and logic circuits using technologies of 250nm, 90nm, and 22nm.
- Chapter 6 investigates how variations in process corners, temperature, and voltage affect the circuit, with output waveforms provided for illustration.
- Chapter 7 concludes the work and discusses future possibilities, summarizing the outcomes of our research and considering the future potential of In-Memory computation.

CHAPTER-2
LITERATURE REVIEW

2.1 Overview

This chapter presents a concise overview of several papers and documents concerning memory, SRAM, and In-Memory computing, with a primary emphasis on computing and the implementation of Boolean circuits.

2.2 Literature Survey

• The current computing infrastructure is built on the Von-Neumann architecture, which encounters difficulties related to the memory wall and power wall. These challenges are particularly significant in today's computing landscape, where data-intensive applications are prevalent. In-Memory Computing (IMC) emerges as a novel architecture aimed at overcoming these power and memory limitations [6].

• The idea of bringing logic and memory closer together has been explored at different levels to alleviate the inherent bottlenecks between processors and memory. Various research projects have delved into the possibilities of near and in-memory computing using conventional CMOS memory technologies like SRAM and DRAM, as well as new memory technologies such as ReRAM [16], PCM [17], and Spintronic. These initiatives are often referred to in the literature as logic-in-memory and computing-in-memory [7].

• A fundamental low-power SRAM cell is made up of a latch (two back-to-back inverters) and access transistors. In lower technology nodes, power dissipation becomes a major issue. This power dissipation impacts gate delay, thereby reducing operational frequency. The two back-to-back inverters are symmetric, allowing them to store logic 0 and logic 1 without any degradation. However, due to leakage from the NMOS in the SRAM cell, it is effective but not highly efficient for low-power applications [18, 19].

• The 6T SRAM cell is mainly used in embedded memory because of its high speed and relatively compact size. The 6T SRAM involves trade-offs between area, error immunity, low leakage through off transistors, and high cell on current. Nonetheless, it provides good stability, NML, NMH, full voltage swing, and high speed [20].

• SRAM semiconductor memory employs two interconnected latching circuits to retain each data bit. An SRAM cell comprises two cross-coupled CMOS inverters that can store a single bit of data, along with two NMOS pass transistors that act as access transistors to facilitate storage. The cell functions in three modes: read, write, and hold. During the read mode, both bit lines and the word line are driven to a high logic level. Two complementary bit lines are connected to a sense amplifier's input, which determines whether a logic '1' or '0' is stored in the SRAM cell. In the write mode, one bit line is driven to a high logic level (VDD), while the other is grounded, enabling the write operation. In hold mode, the word line is grounded to activate the access transistors, and the feedback path preserves the stored data [20].

• The read and hold margins are vital for maintaining data in a memory cell, while the write margin is key for storing data in a memory cell. Traditionally, the yield of SRAM is evaluated based on either the read margin or the write margin. To ensure data stability and performance in a standard 6T SRAM cell, strict control over transistor dimensions is required. For stability during write operations, the current capacity of the access transistor must be greater than that of the NMOS in the back-to-back inverter. The read margin is more important than the hold margin because, during read mode, Q and QB are linked to the precharged bit lines. To achieve high read stability, the current capacity of the PMOS transistors must exceed that of the access transistors. However, this compromises stability during write operations, resulting in a trade-off between read and write performance [21].

• By analyzing the 6T SRAM circuit structure and operation, we find that it consists of two back-to-back inverters (for storing one bit of data) and two access transistors that enable read and write operations. In 6T SRAM, the read and write paths are the same, causing conflicts and affecting the proper execution of these operations. Therefore, 8T SRAM is preferred to eliminate this trade-off between read and write operations [22] [23].

• The shared read and write data path in 6T SRAM leads to conflicts and affects the read and write noise margins. Due to the read disturbance issue in 6T SRAM, a new SRAM circuit with separate paths for read and write operations is needed. This is where 8T SRAM comes into play. The 8T SRAM includes two additional transistors. While the write operation in 8T SRAM remains the same as in the 6T SRAM circuit, the read operation involves these two extra transistors. When reading, the RWL (Read Word Enable) signal is set high, and the output is obtained at RBL. A pre-charge circuit is required in RBL to charge it to VDD [24].

• Compared to the 6T SRAM cell, the 8T-SRAM offers unique variability tolerance. It eliminates the need for a dynamic power supply, which is necessary for 6T SRAM, making it more suitable for low-power operations. While some designs require array-level implementation, the 8T SRAM provides a robust solution for both write ability and stability [22].

• When sizing an 8T SRAM, it is crucial to consider the dimensions of the two additional transistors and the pre-charge condition for reading data from the cell. During the pre-charge condition, attention must also be given to the capacitor's charging and discharging mechanism to ensure accurate data storage in the memory cell.

• While 8T SRAM reduces some of the limitations found in 6T SRAM, the increase in transistor count results in greater power usage. In 8T SRAM, reading and writing the cell are straightforward, eliminating the need for extra circuits required for read operations in 6T. Overall, 8T SRAM provides good speed and avoids read-write complications. The 8T-SRAM cell notably improves RSNM while keeping the write margin, access time, and read margin similar. Generally, read and write operations in 8T SRAM occur without problems, yet there remains a need to enhance the read and write margins and energy efficiency within the bit-cell. To achieve better margins, a 9T SRAM circuit can be employed [25, 22].

• Nevertheless, in the 8T-SRAM cell, data can be written without impacting read performance. The 8T SRAM cell features distinct paths for read and write operations, which enhances the read margin, but further improvements in read and write margins and energy efficiency in the SRAM cell are still necessary [?].

• Researchers have proposed the 9T SRAM to improve both read and write margins along with energy efficiency. In 9T SRAM, except for one additional transistor, the others are the same as in 8T SRAM and function similarly. The two extra transistors provide a decoupled path that aids in improving the read margin. To achieve a higher read noise margin in 9T SRAM, an extra transistor and a VGND (virtual ground) signal are included. This additional transistor provides a virtual ground for the hold operation, reducing hold power and isolating the transistor from the ground during write operations, thereby improving the write noise margin and reducing Ion/Ioff. Compared to the 8T SRAM path, there is only one extra EN signal [?].

• Once we have chosen the 9T SRAM for use, we will proceed to select an In-Memory computation-based Boolean and logic circuit technique. There are primarily two methods for implementing logical circuits. The first is the Inverter method, which involves using an inverter and performing Boolean operations with the aid of an appropriate read enable signal [26]. The second method is the sense amplifier technique, which employs a sense amplifier for reading operations from SRAM. This technique utilizes a suitable reference voltage [25].

2.3 Problem Statement

• Currently, we utilize the Von Neumann architecture in our computer systems, but it faces challenges related to power and memory limitations. Addressing these issues requires developing a new architecture that overcomes the memory and power barriers.

• Numerous studies and research have been conducted on the 6T conventional SRAM circuit, which encounters problems with read disturbance due to the shared read and write path. Some research has explored 8T SRAM to eliminate read slack, but to enhance the write margin, we need to explore new or modified SRAM circuits to achieve optimal read and write margins.

• Most research and scholarly work focus on 90nm and 180nm technology. As technology scales every two years, it is essential to implement circuits and techniques on smaller technology nodes.

• Our literature review indicates that robustness analysis of circuits has been overlooked. We must conduct robustness analysis using Monte Carlo simulation and process corner variation.

2.4 Objective

The primary aim of our work is to address and resolve our problem statements.
• To tackle the challenges associated with the Von Neumann architecture, we are employing an In-Memory computation technique, which assists in overcoming the memory wall and power wall issues.
• To improve the read and write margins, we will incorporate a modified 9T SRAM into the circuit, enhancing these margins effectively.
• For evaluating low technology nodes and making comparisons, our circuit will be implemented in 250nm, 90nm, and 22nm technology nodes.
• To analyze the circuit's robustness, we will conduct Monte Carlo simulations and process corner variations, observing the resulting output waveform.

2.5 Summary

In summary, it can be noted that only a few techniques have been suggested and applied in the realm of In-Memory Computing. Although some progress has been made in the IMC domain, it primarily involved the use of traditional 6T SRAM. In the context of IMC, data is stored in RAM instead of other types of memory, and processing along with minor logical operations are conducted within the memory itself. Two main methods have been proposed for In-Memory Computing: one involves the inverter method, and the other utilizes the sense amplifier sensing scheme. In terms of data processing and management, In-Memory Computing surpasses Nera Data Computing (NDC) and the Von-Neumann architecture.

CHAPTER-3
CELL DESIGN AND WORKING OF SRAM

In recent times, innovations like cloud storage, the Internet of Things (IoT), data mining, and artificial intelligence have been enabled by progress in data storage technology. A typical system relies on both volatile and non-volatile storage to operate effectively. Currently, these requirements are fulfilled by CMOS SRAM, dynamic random-access memory (DRAM), and Flash memory [12]. Each type of storage device offers its own set of benefits and drawbacks. SRAM is extremely fast with very low access time, making it ideal for rapid processor interactions, but it is volatile. Flash memory, on the other hand, is used for large storage needs at high speed due to its lower cost compared to SRAM, although it is slower [13]. SRAM is composed of two inverters connected in a loop, enabling data storage at two nodes. DRAM, which uses a capacitor to hold data, needs a refreshing circuit because the capacitor can lose charge. We plan to use SRAM for in-memory computing. This approach is a promising solution to speed challenges and represents a growing field. In-memory computing (IMC) involves storing data in RAM rather than in disk-hosted databases.

3.1 Working and Design Criteria for 6T-SRAM

In the current context, SRAM is predominantly utilized instead of microprocessors, cache memory, computers, laptops, and engineering workstations. It is also employed in high-speed devices due to its low power consumption. The 6T SRAM, which consists of six transistors, includes four transistors (M3, M4, M5, M6) that function as a latch (back-to-back inverter) for data storage, while the remaining two transistors (M2, M1) are used for accessing or writing data into the latch. The two back-to-back inverters store data in both its normal and complementary forms, thanks to the inverter (regenerative circuit) that provides a full logic swing of either 0 or 1. The 6T SRAM has a very low Read Noise Margin.

To improve this, the width of the M3 and M4 transistors needs to be increased, but this leads to a larger area, which in turn increases leakage currents, thus presenting a challenge in terms of area and power consumption. When SRAM is used for operations requiring low power consumption, voltage scaling techniques are often employed. However, the reliability of SRAM becomes somewhat questionable during scaling operations, making it crucial to analyze SRAM performance during these operations.

When determining the sizing of a 6T SRAM cell, several factors must be considered to ensure it performs read, write, and hold operations correctly. If not, the data written may not be accurately retrieved during reading.

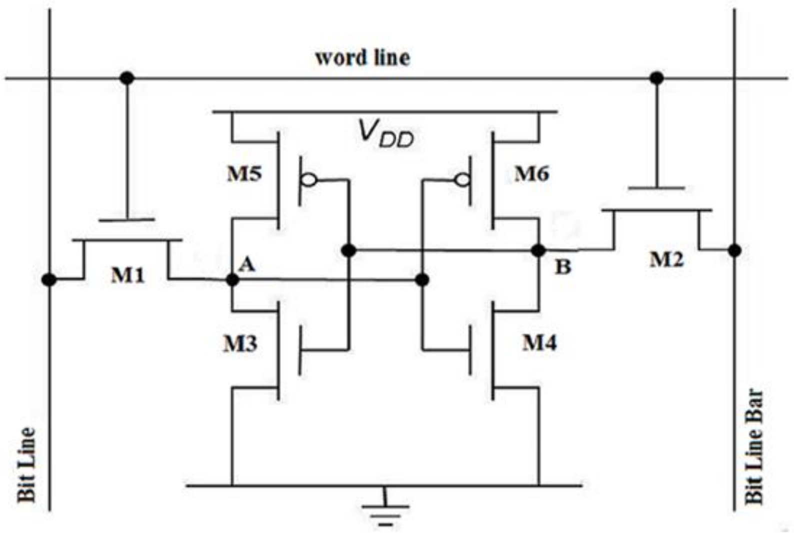


Figure 3.1: Schematics of 6T SRAM circuit [27]

Therefore, designing the appropriate (W/L) ratio of transistors is crucial for achieving effective read and write operations. Increasing VDD can enhance the cell's stability and speed, but it also leads to higher power consumption, creating a trade-off between speed and power. Thus, it is essential to design SRAM to optimize power efficiency while maintaining good speed. SRAM memory operates in three modes: standby, write, and read.

3.1.1 Standby mode

In standby mode, the word line is grounded, which disconnects both access transistors M1 and M2 from the latch [27]. During this state, the data remains unchanged as long as the power supply is active, ensuring that the information stored in the latch is preserved.

3.1.2 Write Mode

In write mode, the write word line must be set to a logic high or connected to VDD. This allows data to be overwritten in memory using the bit line and bit line bar. Writing data is feasible when the internal driver is weaker than the external driver, ensuring that the external driver transistors are stronger, which facilitates easier data writing. During the read operation, it is crucial to provide inverted logic on the bit and bit-bar lines; for instance, to store a 0, the bit line should be 0 and the bit-line-bar should be 1 (see figure 3.2). Providing the same logic to both the bit and bit-bar lines will result in incorrect logic storage. For proper write operations, transistor sizing must adhere to equation A.

$$\frac{\left(\frac{W}{L}\right)_5}{\left(\frac{W}{L}\right)_1} < \frac{\mu_n}{\mu_p} \frac{2(V_{DD}-1.5V_{T,n})V_{T,n}}{(V_{DD}+V_{T,p})^2} \text{-------------------------------------} \quad (A)$$

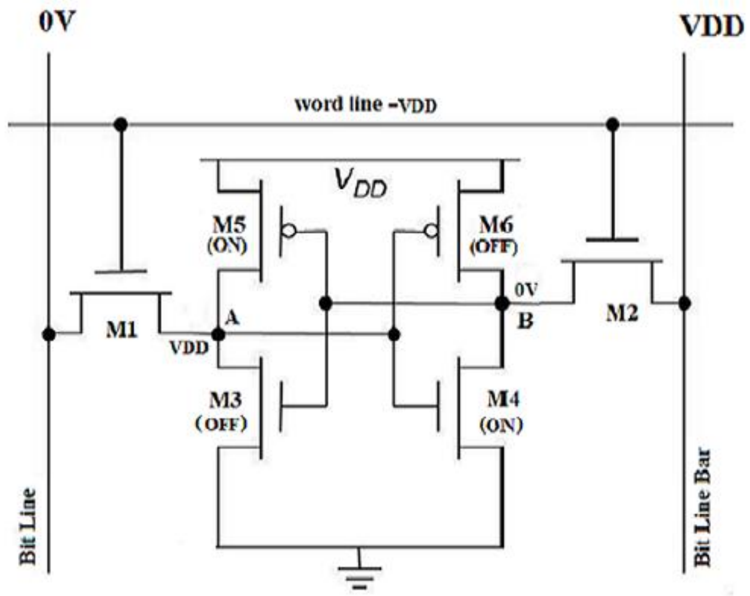


Figure 3.2: Write 0 Operation in 6T SRAM Cell [27]

3.1.3 Read Mode

In order to read data from 6T SRAM, we must create both bit lines at logic 1 or VDD. Let's say we would want to read whatever we write in figure 3.2. Due to the stored zero, M4 and M5 transistors will operate in the cut-off zone whereas M3 and M6 transistors will operate in the linear region. We charged both bit lines at VDD.M1 and M2 must continue to use Word Line. Due to its source and drain having the same voltage, M2 will not function. However, bit lines begin to discharge to ground voltage and M1 and M3 will display a current circuit. The sensing amplifier now receives both bits and will detect and supply the contents of the memory cell. The necessary condition equation B is perform read operation without failure:

$$\frac{\left(\frac{W}{L}\right)_1}{\left(\frac{W}{L}\right)_3} < \frac{2(V_{DD}-1.5V_{T,n})V_{T,n}}{(V_{DD}-2V_{T,n})^2} \text{ - - - - - - - - - - - - - - - -}(B)$$

It is possible to build SRAM without any read or write failures by employing mathematical criteria (A) and (B). If not, the condition equation C below must be met in order to carry out the proper action, which is either a read or write operation.

$$\left(\frac{W}{L}\right)_{pull-up} < \left(\frac{W}{L}\right)_{access} \ll \left(\frac{W}{L}\right)_{pull-down} \text{------------------------}(C)$$

3.1.4 Problem with 6T SRAM

In conventional 6T SRAMs, read and write operations and stabilities clash. It will thus impact the read and write margin. In order to solve the read/write issue, we switch to 8T SRAM [28].

3.2 Working and Design Criteria for 8T-SRAM

In 6T SRAM, the read and write data paths are the same, which can lead to conflicts affecting the noise margin during these operations. To address the read disturbances encountered with 6T SRAM, we transitioned to a new SRAM design featuring separate read and write paths. This is where 8T SRAM becomes relevant. As illustrated in figure 3.3, 8T SRAM incorporates two additional transistors. While the write process in the 8T SRAM circuit mirrors that of the 6T SRAM, the read process necessitates the use of the two extra transistors depicted in the diagram. When we intend to read, we elevate the RWL (Read Word Enable) signal as much as possible, resulting in an output at RBL. To elevate RBL to VDD, a pre-charge circuit in RBL is required [29].

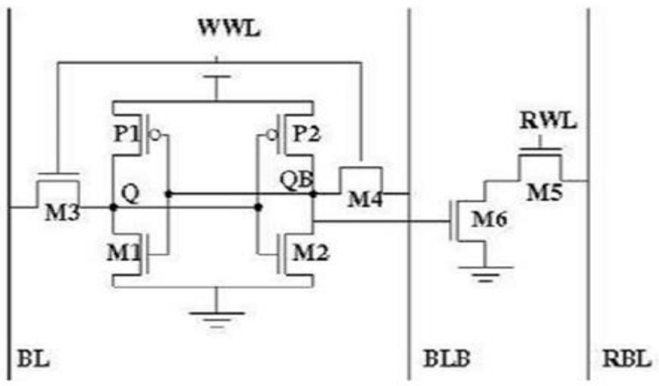


Figure 3.3: Schematics of 8-T SRAM [26]

The 8T SRAM cell performs the same write operation as the 6T SRAM cell. We may use the following discussion to write 0 or 1 in the SRAM cell:

3.2.1 Write '0' Operation

We set bit line to logic 0 and bit-line-bar to logic VDD in order to write logic 0 in an SRAM cell. We also set WWL (Word Write Line) to logic 1.

3.2.2 Write '1' Operation

To write logic 0 in an SRAM cell, we set bit line to logic 0and bit-line-bar to logic VDD. We also set WWL (Word Write Line) to logic. This write operation is the same for both 6T and 8T SRAM operations. The RBL line must be pre-charged to logic VDD before we can proceed with the read operation in the 8T SRAM cell. As in the traditional one, the read operation is started by pre-charging the read bit line to VDD.

3.2.3 Read ‘0’ Operation

We must create an active RWL line for the read operation by providing logic VDD, which puts the M5 transistor in an active state. When the value at Q is "0," the M6 transistor is turned on, and the RBL receives a discharge route from VDD to ground via the M5 and M6 transistors, resulting in a discharge to logic 0. According to this method, logic 0 is the value stored in SRAM.

3.2.4 Read ‘1’ Operation

RBL will be at logic 1 and there won't be a discharge path if transistor M6 is off and stores logic 1. This indicates that store logic 1 is present in SRAM.

3.2.5 Design Constraints

The size of two additional transistors and the pre-charge condition for accessing the data from the cell are the most important factors in 8T SRAM sizing. Pay attention to the capacitor's charging and discharging mechanism during the pre-charge phase to ensure that the memory cell stores accurate data.

3.2.6 Problem with 8T SRAM

Due to the area and power consumption drawbacks of 8T SRAM, the challenges associated with 6T SRAM are minimized. With 8T, reading and writing to the cell are straightforward, eliminating the need for additional circuits like 6T for read operations. Consequently, 8T SRAM delivers satisfactory speed without any significant read/write issues. The 8T-SRAM cell provides a notably improved RSNM while maintaining similar write margin, access time, and write time. Typically, 8T SRAM does not encounter difficulties in writing and reading. Nonetheless, there is still a need to enhance the bit-cell's read and write margins and energy efficiency. To address this, a 9T SRAM circuit might be employed to boost read and write margins.

3.3 Working of 9T-SRAM

In order to improve SRAM's read and write margins as well as its energy efficiency, researchers suggested 9T SRAM. The proposed 9T SRAM's schematics are shown in figure 3.4 below. Other than the T7 transistor, the other components of the 9T SRAM are identical to those of the 8T SRAM and function similarly. While T5 and T6 function as access transistors and T9 and T8 as read circuitry components, T1, T2, T3, and T4 offer latches. The decoupling route offered by T8 and T9 contributes to an increase in read margin.
The 9T SRAM uses the VGND signal to further increase the read noise margin. The design additionally has an additional transistor (T7) that isolates the cell core from GND and provides a virtual ground condition to the core during the hold state. This improves the bit-cell's static write noise margin, lowers hold power, and increases the Ion/I off ratio. Table 3.1 displays the 9T SRAM's read and write operation table. Compared to 8T SRAM, there is just one more EN signal.

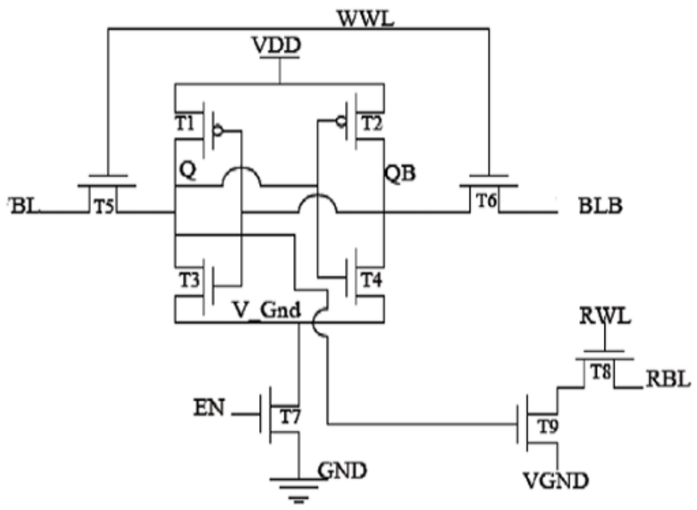


Figure 3.4: Schematics of 9-T SRAM [30]

3.3.1 Read/Write Operation of 9T SRAM Bit-Cell

Table 3.1: Operation of 9T SRAM

Operation	WWL	BL	BLB	EN	RWL	VGND	RBL
Write 0	VDD	GND	VDD	GND	GND	VDD	Precharge
Write 1	VDD	VDD	GND	GND	GND	VDD	Precharge
Read	GND	Floating	Floating	VDD	VDD	VDD	Precharge
Hold	GND	Floating	Floating	GND	GND	VDD	Precharge

3.4 Conclusion

This chapter covered the operation of SRAM as well as the benefits and drawbacks of 6T, 8T, and 9T SRAM. Lastly, we decided to use 9T SRAM for our dissertation.

CHAPTER-4
BASIC'S OF IN-MEMORY COMPUTING

4.1 In-Memory Computing (IMC) of SRAM

In-memory computing (IMC) is the practice of storing data in dedicated servers' random-access memory (RAM) rather than in intricate relational databases that operate on comparatively slow disk drives [23]. Compared to old disk-based structures, In-Memory Computing uses high-performance, integrated, allocated memory structures to compute and transact on large-scale information units in real time—orders of magnitude faster [31]. Enterprise clients, including banks, utilities, and merchants, may quickly identify trends, analyze massive amounts of data on the go, and complete tasks quickly thanks to in-memory computing [32].

4.2 In-Memory Computing Using Sense-Amplifier Method

The term "in-memory computation" (IMC) refers to information processing processes that are carried out internally by the memory array, including computation, movement, and statistics storage. Enabling several examination phrase traces at the same time is the main motivation for 9T SRAM cell-based completely IMC. A sense amplifier receives input from a 9T SRAM array, which is made up of many RBL SRAM cells. 9T SRAM, which helps with read margin, has two extra transistors for read operations and a 6T cell for data writing (Figure 2.13(a)). Because of this setup, we can use a sense amplifier and SRAM cell to create a Boolean circuit [33, 34].

As seen in figure 4.2, we are utilizing two 9T SRAM cells and integrating the RBL of both SRAMs to achieve the Boolean equation. Since we previously established that there is a channel for RBL to discharge but no path for RBL to charge (without precharge), we need a precharge circuit at the RBL location. Therefore, we may utilize one capacitor for a smooth response and PMOS as a precharge circuit here.

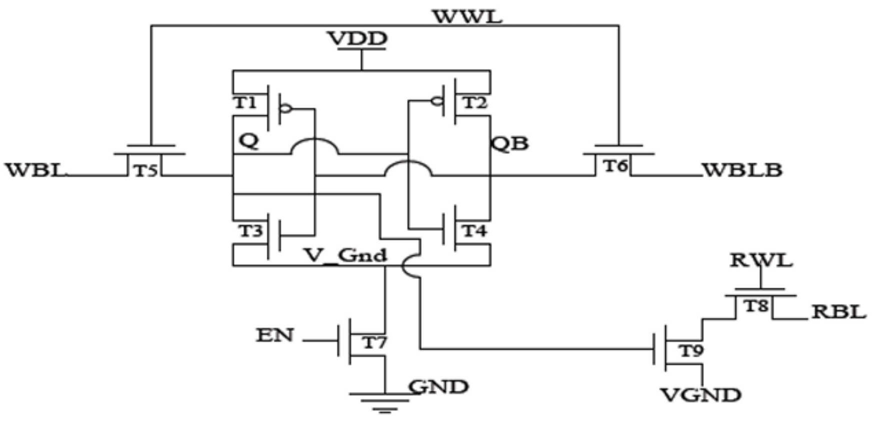


Figure 4.1: The schematic of 9T SRAM cell

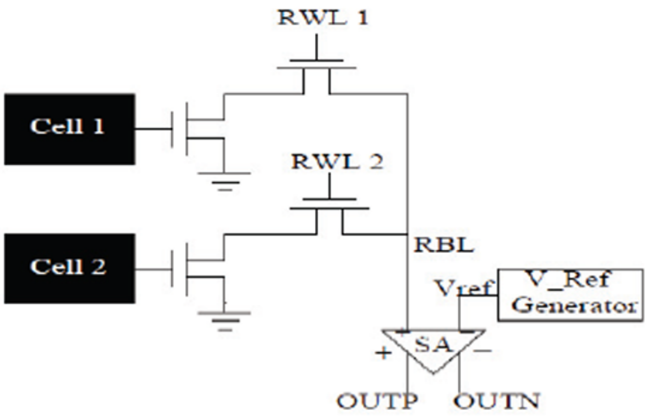


Figure 4.2: The circuit schematic of IMC operation with 9T SRAM Cell and Sense amplifier [19]

This primarily introduces two new names: sensing amplifier and voltage-reference generator. The voltage reference generator produces voltage in accordance with operation, and the sense amplifier detects the RBL voltage and compares inputs.

4.2.1 Voltage Reference Generator

The voltage is produced in accordance with operation by the Voltage Reference Generator. According to this computation technique, reference voltage is needed while conducting NAND/NOR operations. Figure 4.3 may be used to compute the reference voltage for NAND and NOR operations.

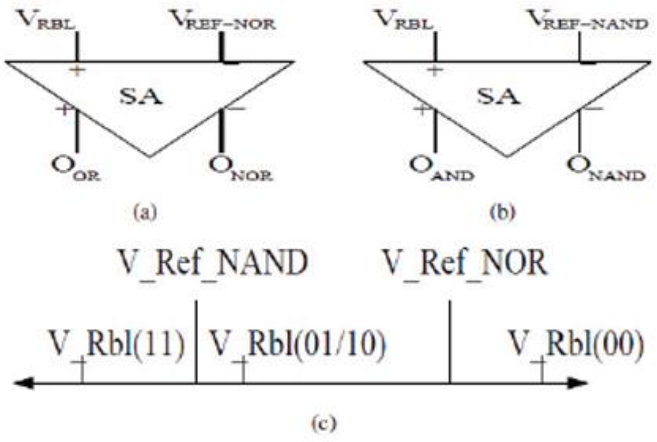


Figure 4.3 : Sensing configuration for IMC functionality. a) Configuration for OR/NOR functionality. b) Configuration for AND/NAND functionality. c) The range of reference voltage in the sensing configuration [25]

Figure 4.3 illustrates that our reference voltage needs to fall between V-RBL(00) and V-RBL(10/01) in order for NAND operation to function. The reference voltage for NOR functioning has to fall between V-RBL(01/10) and V-RBL(11).

4.2.2 Sense Amplifier

A sense amplifier is an integral part of the memory's read circuitry, essential for retrieving data from a memory cell. It serves two main functions: detecting the data from the bit cell and boosting the small voltage swing to logic levels that can be accurately interpreted by external logic systems. In this context, a latch-type sense amplifier, which consists of back-to-back inverters, is employed.

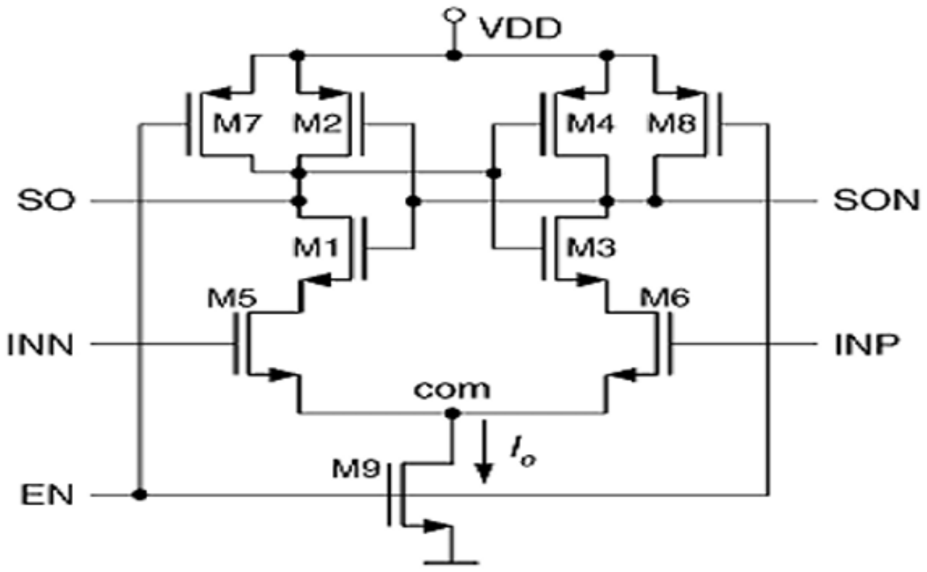


Figure 4.4: Latch type Sense Amplifier [35]

To execute Boolean logic gate operations, two separate cells shown in Figure 4.2(b) are utilized to hold the operands, and the logic gate output is obtained through a sense amplifier that detects variations in the RBL voltage. Figure 4.2(b) demonstrates the IMC operation using a 9T SRAM by activating two read word lines simultaneously and linking VRBL and VRef voltages to the sense amplifier. In this configuration, a latch-type sense amplifier with a 300 fF RBL capacitor is used. The sense amplifier acts as a comparator circuit, evaluating two inputs to ascertain which one is greater or lesser. Initially, the RBL voltage is pre-charged to VDD and either discharges or stays in its pre-charged state depending on the data stored in the bit-cell. We suggest a sensing scheme to identify the different VRBL values, thereby enabling logic operations on the operand values stored in the bit-cell.

4.3 Bit-wise NOR/OR operation

To perform a NOR operation, the sensing scheme shown in Figure 4.2 must be utilized. The RBL of both SRAM cells should be connected in a wired AND configuration and attached to the positive terminal of the sense amplifier. The negative terminal should be supplied with a reference voltage, which can be adjusted to be near VDD as per the sensing scheme detailed in Figure 4.3. The reference voltage, VRBL, will change depending on the input combinations. As demonstrated in Figure 4.5, for the 00 input combinations, VRBL (00) is high, whereas for the 11 combination, VRBL-(11) is low.

For the 01/10 combinations, VRBL-(01/10) lies between VRBL-(00) and VRBL-(11). It is understood that the sense amplifier acts as a comparator circuit. We have selected VREF-NOR to be positioned between VRBL-(00) and VRBL-(01/10). Assuming a VDD of 5 Volts, VRBL-(00) is approximately 4.6 Volts, VRBL-(01/10) is about 2.5 Volts, VRBL-(11) is roughly 1.8 Volts, and V-REF is set at 4 Volts.

For Logic 00, the comparison between 4.6 Volts and 4 Volts results in an output of 1.
For Logic 01/10, the comparison between 2.5 Volts and 4 Volts yields an output of 0.
For Logic 11, the comparison between 1.8 Volts and 4 Volts also results in an output of 0.

Thus, we can confirm the Logic NOR operation. Due to the latch-type sense amplifier, the complement of NOR is also present, allowing for the implementation of an OR gate as well.

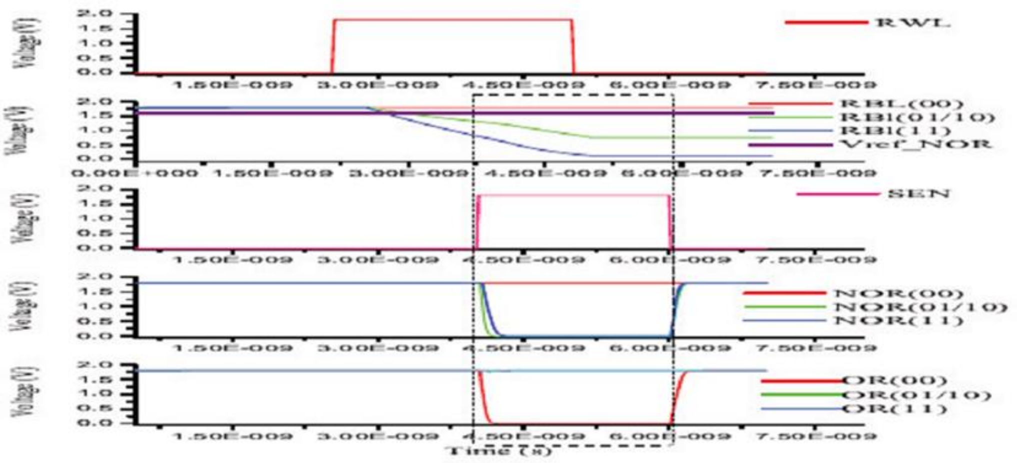


Figure 4.5: The output for all possible input (00,01,10,11) for performing NOR/OR operation [35]

4.5 Conclusion

This chapter delves into the idea of In-Memory computing methods and demonstrates how to apply these techniques to execute fundamental Boolean logic.

CHAPTER-5

BOOLEAN AND LOGIC CIRCUIT DESIGN USING IN-MEMORY COMPUTING

In this chapter, we will verify the reasons behind the functionality of 9T SRAM, explore how it operates, and perform appropriate sizing calculations to ensure successful read and write operations. Following this, we will apply 9T SRAM to an in-memory computing application. We will implement some Boolean and logic circuits and confirm that we are executing in-memory computation-based Boolean and logic circuits.

5.1 Why 9T SRAM

After implementing circuits using 6T, 8T, and the proposed 9T SRAM, we have obtained some results. The waveform and implementation details were already covered in chapters 5 and 6. All our calculations are conducted in Cadence using GPDK 90nm technology at room temperature. We will compare the results of the 6T, 8T, and 9T SRAM.

5.1.1 Noise Margin

As we scale our device, the size of transistors and the required voltage supply decrease. However, this also introduces the problem of increased sensitivity to noise in our circuits and devices. This is a crucial issue that must be addressed before utilizing these devices. We are evaluating the read static noise margin, write static noise margin, and dynamic noise margin. The dynamic noise margin is defined as the minimum duration of the write-word line necessary to perform a write operation without failure.

SRAM	**RSNM(milli Volt)**	**WSNM(milli Volt)**	**DNM(Pico Sec.)**
6T SRAM	212	672	32
8T SRAM	608	703	39
9T SRAM	848	848	18

5.1.2 Leakage Current

We determined the leakage current and the Ion/Ioff ratio. The table below allows us to make a comparison.

SRAM	**Leakage Current (Pico Amp)**	**Ion/Ioff**
6T SRAM	5.51	26.257
8T SRAM	5.48	26.551
9T SRAM	3.8	53.261

5.1.3 Performance Analysis

We implemented an in-memory computation-based Boolean logic design utilizing 9T SRAM. The performance of the SRAM circuit can be evaluated by considering factors such as read access time, write access time, read-energy [36], and write-energy [36].

SRAM	Read Access (pSec)	Write Access(pSec)	Read Energy (zJ)	Write Energy (fJ)
6T SRAM	7.5	55	3	0.089
8T SRAM	10	63	4	0.101
9T SRAM	14	61	1.7	0.095

5.1.4 Conclusion

In this analysis, we evaluate the performance of our SRAM in terms of working, read and write noise margins, leakage current, and read and write access times. The simulation results and analysis indicate that the 9T SRAM outperforms the 8T and 6T versions. As shown in the table, the 9T SRAM exhibits lower leakage current and longer read access time, while avoiding the read and write issues present in the 6T SRAM. Additionally, the 9T SRAM has a higher Ion/Ioff ratio compared to the 6T and 8T SRAMs. Due to these advantages, we plan to utilize the 9T SRAM in our future projects.

5.2 Sizing of 9T SRAM Transistor for Operation’s

As we are aware, the proper functioning of SRAM heavily relies on the correct sizing of transistors. If not sized correctly, it can lead to issues with read or write failures during operation. In this context, we are utilizing a 9T SRAM circuit and have compiled a reference table.

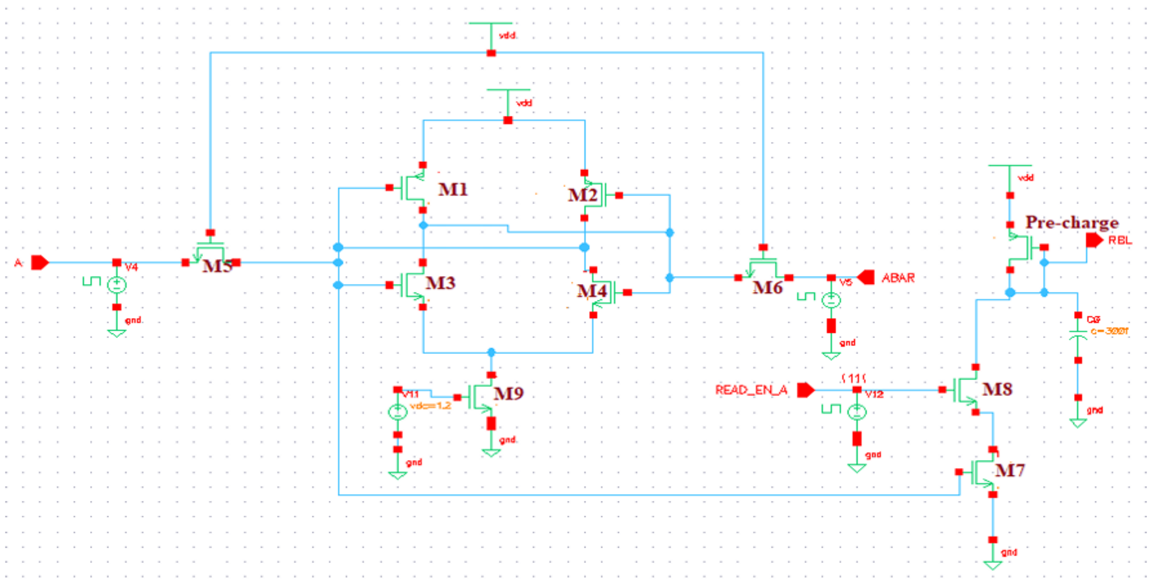


Figure 5.1: Circuit Schematics of 9T SRAM

We determined the sizing for all transistors in the 9T SRAM, ensuring that it effectively performs read, hold, and write operations under all conditions.

Table 5.1: Operation of 9T SRAM

Transistor	Sizing for operation(um)
M1	0.50
M2	0.50
M3	6.50
M4	6.50
M5	1.50
M6	1.50
M7	2.50
M8	6.50
M9	2.50
cap	300fF
Pre-Charge Transistor	7.50

5.3 In-Memory Computing based Boolean and Logical Circuit using 250nm Technology

We are developing a logic circuit based on in-memory computation using Generic 250nm technology, utilizing the Tanner Tool. Our plan includes implementing fundamental Boolean logic as well as some combinational circuits.

5.3.1 NAND/NOR Logic Implementation

In this work, we are implementing a 2-input NAND logic using a 9T SRAM and conducting IMC. The waveforms for the NAND logic are shown in Figure 4.2. We utilize Generic 250nm technology (5 Volt) to implement both NAND and NOR logic. Initially, we perform a write operation followed by a read operation. We use two SRAM cells, A and B, and then combine the RBL of both cells, directing it to the Sense Amplifier. It is crucial to provide an enable logic for the sense amplifier.

Therefore, we must ensure that both the read enable and the sense amplifier enable are activated simultaneously to achieve the correct NAND and NOR logic output. As discussed earlier, we need to consider the reference voltage as shown in Figure 4.3(c). For the NAND operation, the reference voltage should be between VRBL01/10 and VRBL11. For the NAND logic, Vref is set at 1.6 Volts for the sense amplifier.

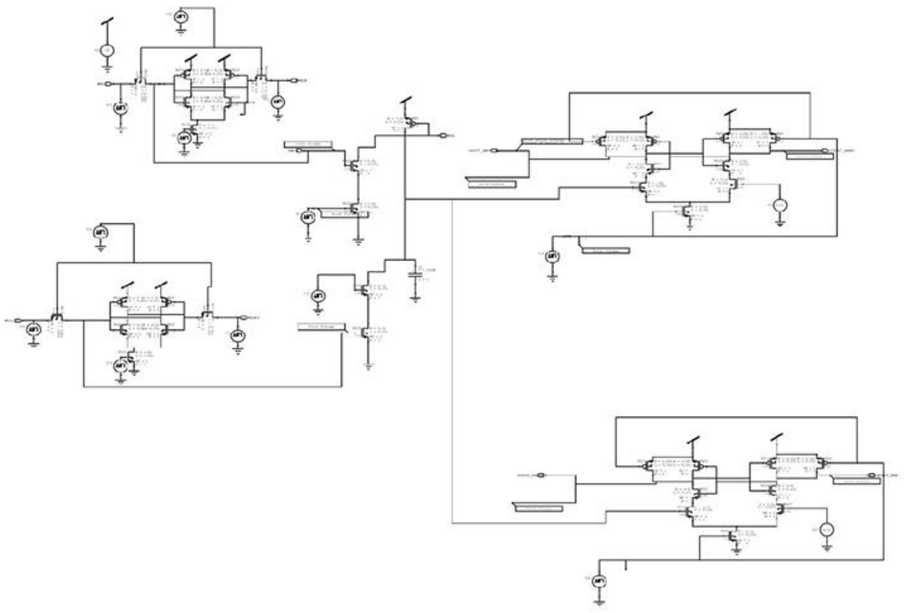


Figure 5.2: Circuit Schematics of IMC using 9T SRAM

The sense amplifier functions as a comparator, and we are using a latch-type sense amplifier here. When both the READ and the sense amplifier are enabled, we will obtain the correct result. We are implementing the NAND logic, and the waveforms are provided below. The results are obtained using the Tanner tool.

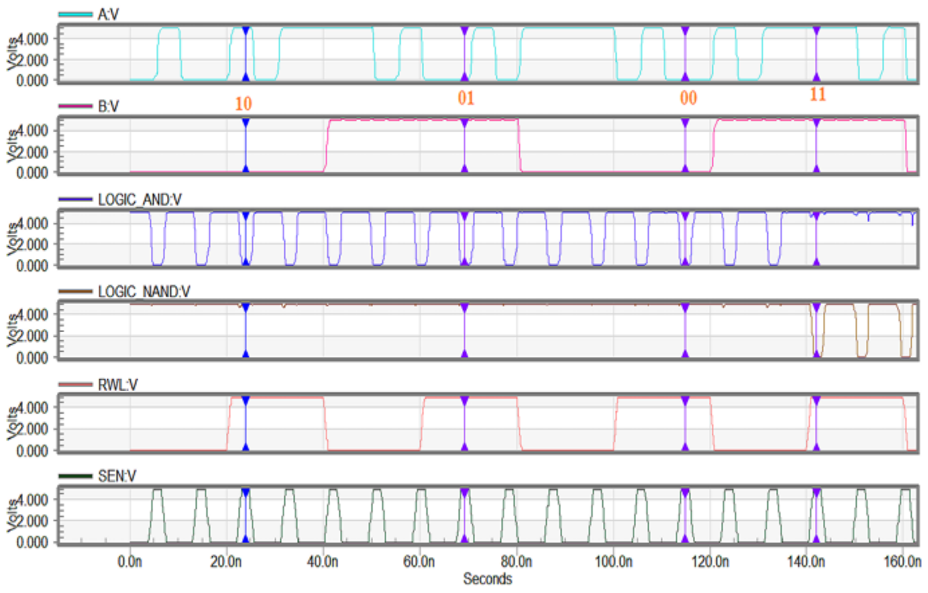


Figure 5.3: Output Waveform of NAND/AND Logic using 9T SRAM

Following our previous discussion, it is crucial to focus on the reference voltage as depicted in Figure 4.3(c). For the NOR operation, the reference voltage should be set between VRBL01/10 and VRBL00. Specifically, for NOR logic, the reference voltage (Vref) for the sense amplifier is established at 4.3 Volts. The sense amplifier acts as a comparator, and we are employing a latch-type sense amplifier in this scenario.

When the READ function is activated and the sense amplifier is enabled simultaneously, we will achieve accurate results. We are implementing NOR logic, and the corresponding waveforms are illustrated below. The results have been derived using the Tanner tool, and we have also measured the delay time during the execution of the Boolean circuit.

Operation	**Delay(nsec)**
NAND	0.20
NOR	0.15
AND	0.17
OR	0.8

The table below indicates that the NAND gate exhibits the longest delay, whereas the OR gate has a shorter delay. From the table above, it is clear that NAND logic experiences more delay, so when constructing our circuit using NAND gates with in-memory computing techniques, we must be cautious of the delay, as it could affect our operation.

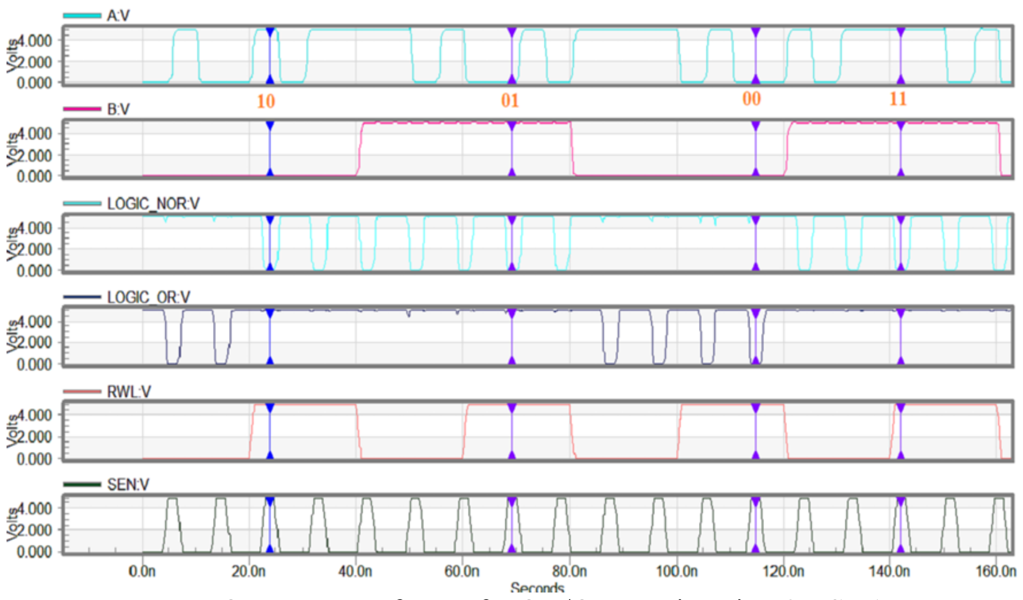


Figure 5.4: Output Waveform of NOR/OR Logic using 9T SRAM

The waveforms above verify that our circuit delivers the correct output for all input combinations: 00, 01, 10, and 11. It is well-known that NAND and NOR are universal gates, enabling us to build any digital circuit using them. Consequently, we will design our circuits using in-memory computing techniques, prioritizing minimal hardware requirements. Although it is feasible to create a circuit in various ways with more hardware, this approach is neither efficient nor reliable for operation. In our work, we will develop other combinational circuits using fewer hardware components and in an efficient manner.

5.3.2 EXOR Circuit Implementation

There are primarily four straightforward methods to implement the EXOR logic.

A. Using Only NAND Gates:

An XOR gate can be constructed by connecting four NAND gates, as illustrated below. This setup results in a propagation delay that is three times longer than that of a single NAND gate.

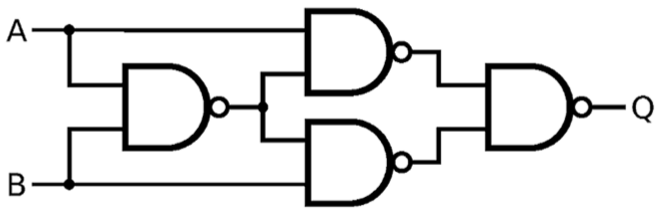


Figure 5.5: EXOR Logic using NAND Gate Only

B. Using Only NOR Gates:

An XOR gate can also be created by connecting four NAND gates, as depicted below. This arrangement leads to a propagation delay three times that of a single NAND gate.

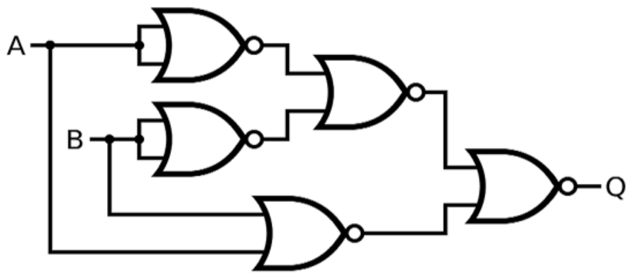


Figure 5.6: EXOR Logic using NOR Gate Only

C. Using Only NOR and AND Gates:

We have previously implemented NOR and AND logic using a sense amplifier in pre-dissertation work. Therefore, this method is also a straightforward way to implement EXOR logic, as shown in the connection below.

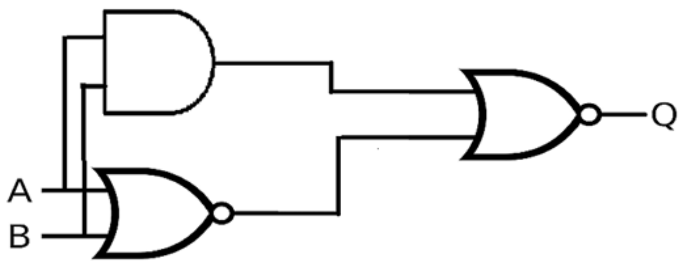


Figure 5.7: EXOR Logic using NOR and AND Gate Only

D. Using Only OR and NAND Gates:

We have already implemented OR and NAND logic using a sense amplifier in pre-dissertation work. Thus, this method is also a simple way to implement EXOR logic, as demonstrated in the connection below.

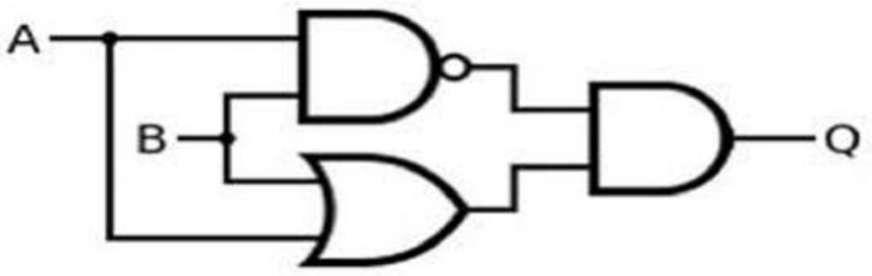


Figure 5.8: EXOR Logic using OR and NAND Gate Only

Implementing the EXOR gate using NOR logic here helps reduce the number of transistors required. Other methods can also be used to create an EXOR circuit.

Truth Table of EXOR Logic: According to the implementation of EXOR logic using in-memory computing techniques, our circuit can be constructed in various ways with the help of NAND and NOR gates. However, our objective is to create a logical circuit with minimal hardware in an efficient manner. We have obtained output and verified it against standard results. Through waveform and truth table analysis, we confirmed the successful implementation of EXOR logic.

Table 5.2: Truth Table of EXOR Logic

Input A	Input B	Output (EXOR)
0	0	0
0	1	1
1	0	1
1	1	0

Our EXOR logic results are displayed in Figure 5.9. The figure below allows us to verify that our results are accurate, and we can confirm our results for input combinations 00, 01, 10, and 11. The overall performance of the EXOR logic implementation is both good and accurate.

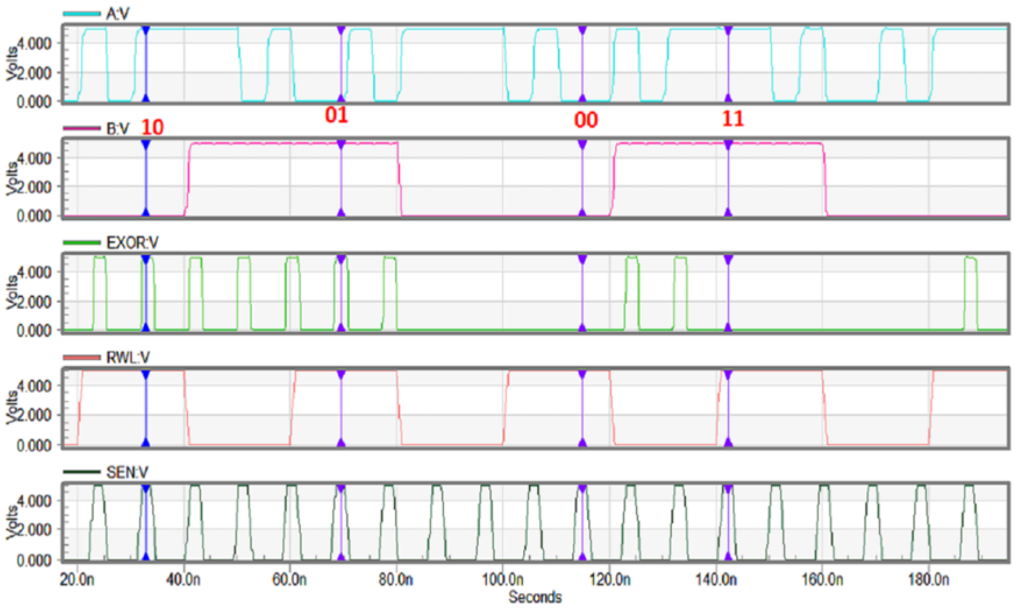


Figure 5.9: Output Wave-forms of EXOR Gate

5.3.3 EXNOR Circuit Implementation

In our previous research, we successfully implemented NAND, NOR, AND, and OR logic using in-memory computing techniques. While there are multiple methods to create an EXNOR circuit, we have already established NAND, NOR, AND, and OR logic. There are four straightforward approaches to constructing EXNOR logic:

A. Using Only NOR Gates:

An EXNOR gate can be constructed by connecting four NOR gates, as illustrated below. This setup results in a propagation delay three times that of a single NOR gate.

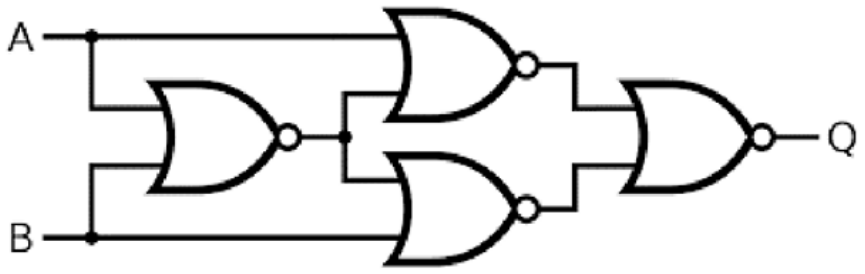


Figure 5.10: EXNOR Logic using NOR Gate Only

B. Using Only NAND Gates:

An EXNOR gate can be formed by linking five NAND gates, as depicted below. This configuration also results in a propagation delay three times that of a single NAND gate.

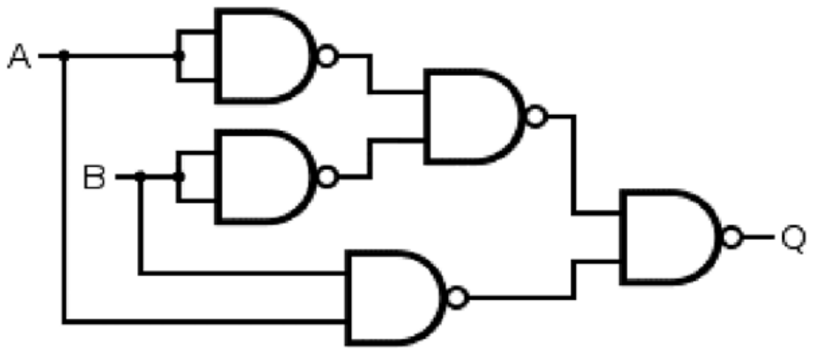


Figure 5.11: EXNOR Logic using NOR Gate Only

C. Using Only NOR and AND Gates:

Since we have previously implemented NOR and AND logic using a sense amplifier in our earlier work, this method offers a simple way to create EXNOR logic using the connections shown below.

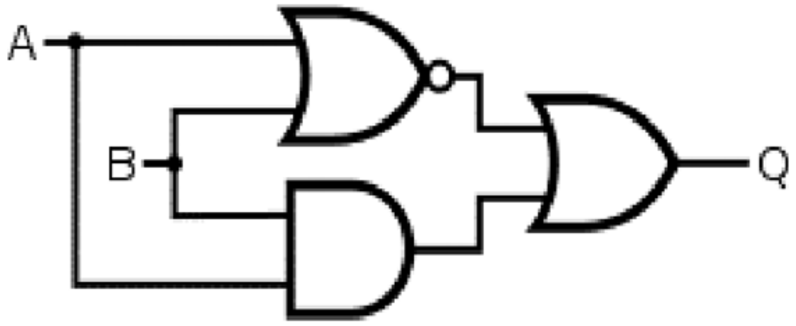


Figure 5.12: EXNOR Logic using NOR and AND Gate Only

D. Using Only OR and NAND Gates:

Similarly, having already implemented OR and NAND logic with a sense amplifier in our prior work, this method provides an easy way to construct EXNOR logic using the connections provided below.

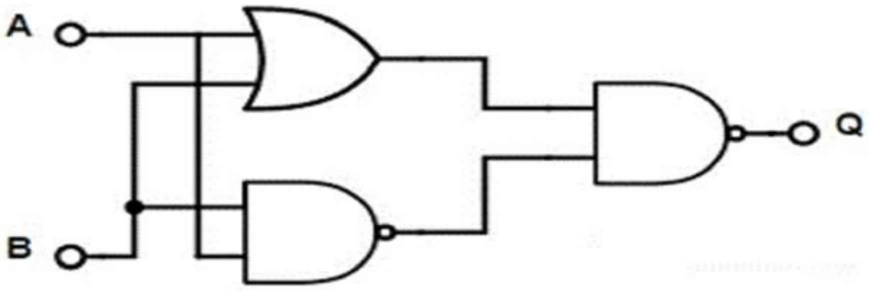


Figure 5.13: EXNOR Logic using OR and NAND Gate Only

We are creating an EXNOR gate using NOR logic, which reduces the number of transistors needed. There are other methods to construct an EXNOR circuit as well. Upon implementing the EXNOR, we obtain results such as:

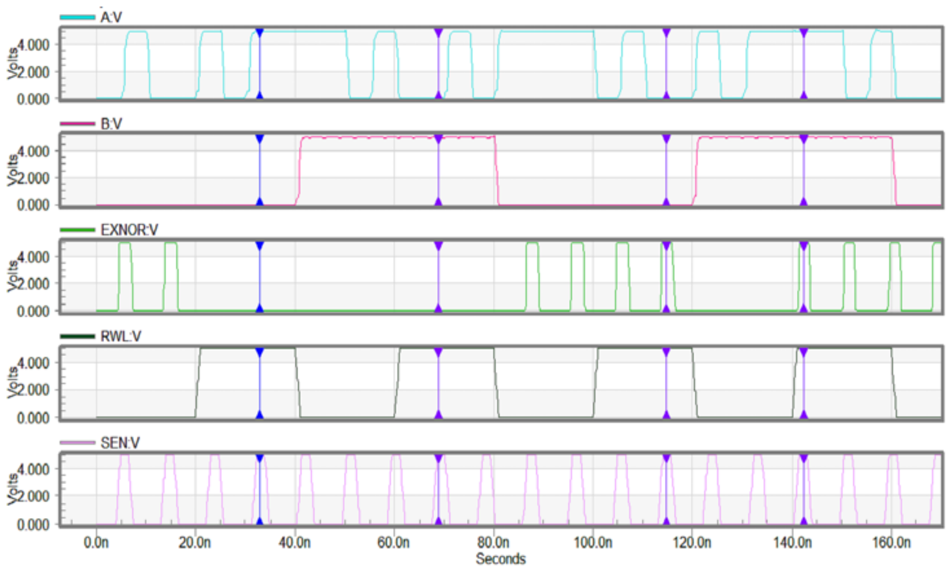


Figure 5.14: Output Wave-forms of EXNOR Gate

Truth table of EXNOR Logic

By implementing the EXNOR gate using NOR logic, we can reduce the number of transistors required. Although other methods can also be used to create an EXNOR circuit, after implementation, we obtained results such as the EXNOR logic truth table. Using the in-memory computing technique for EXNOR logic, we achieved outputs that matched standard results. Through waveform analysis and the truth table, we confirmed the successful implementation of EXNOR logic.

Input A	Input B	Output(EXNOR)
0	0	1
0	1	0
1	0	0
1	1	1

We also measured the delay time during the operation of EXOR and EXNOR circuits, finding that EXNOR has a longer delay than the EXOR circuit.

Operation	Delay(nsec)
EXOR	0.18
EXNOR	0.28

The table above verifies that EXNOR experiences more delay because it was implemented using NAND gates, and as discussed earlier, NAND gates have more delay due to the sense amplifier mechanism of discharging and comparison with lower voltage.

5.4. In-Memory Computing-based Boolean Circuit using 90nm Technology

For further work, we are employing GPDK 90nm technology with a VDD of 1.8V. According to Moore's law, the number of transistors on a chip doubles approximately every 24 months. Therefore, we are implementing NAND, NOR, AND, OR, EXOR, and EXNOR circuits using Cadence Virtuoso. As previously implemented in section 5.2, we are applying the same concept to lower technology nodes.

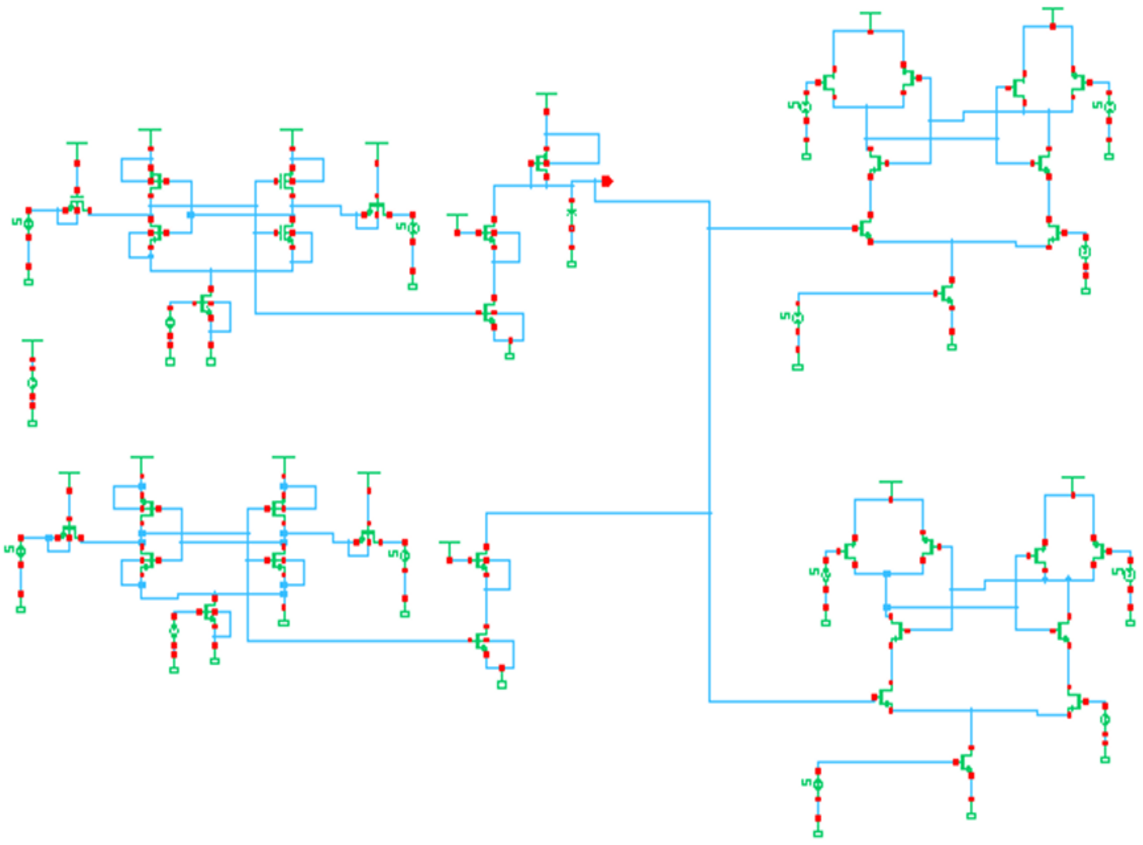


Figure 5.15: Circuit schematics of IMC using 9T SRAM (90nm Technology)

We are implementing NAND and NOR logic. We use two SRAM cells, A and B, as inputs to the RBL through wired AND logic. We set Vref-NOR and Vref-NAND to 1.1 Volt and 0.653 Volt, respectively, for operation. During the operation stage, we activate the sense amplifier, which provides the correct NAND and NOR logic outputs and their complements.

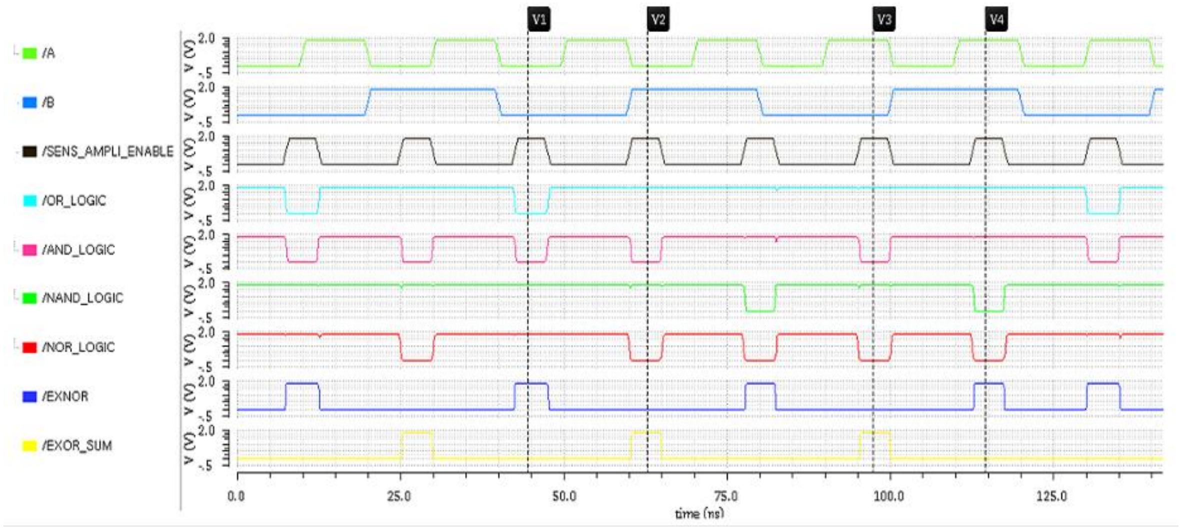


Figure 5.16: Output Wave-form of NAND/NOR/AND/OR/EXOR Logic(90nm Technology)

Based on the circuit implementation and output waveform, we create a table of inputs and outputs.

Table 5.7: Truth Table of Boolean Logic

Input A	Input B	AND	NAND	OR	NOR	EXOR	EXNOR
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
0	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

We have successfully implemented NAND, NOR, AND, OR, EXOR, and EXNOR logic, as verified by the table. We also calculated delay, average energy, EDP, and PDP using Cadence. The table below displays all these parameters.

	Delay(nsec)	Avg Power(Watt)	PDP(E-11)	EDP(E-21)
NAND	1.1665	0.133	22.144	29.45
NOR	1.352	0.031	4.1912	1.2992
AND	1.370	0.0468	6.411	3.0006
OR	1.1660	0.0448	7.4368	3.3168

5.5 Conclusion

In this chapter, we have completed most of the work related to our problem statement discussed in the literature review chapter, including circuit implementation, comparison, combinational logic implementation, and more.

• Our problem statement involves the use of appropriate SRAM. In this chapter, we analyze 6T SRAM, 8T SRAM, and 9T SRAM. We calculated delay, Ion/Ioff, leakage current, read margin, write margin, energy, and many parameters, solving the problem of using 9T SRAM for in-memory computing techniques.

• For read and write operations with SRAM, we calculated transistor sizing to perform operations correctly. After that, we implemented basic NAND, NOR, AND, and OR logic using 250nm technology and focused on in-memory computing-based Boolean circuit design. We also implemented other digital circuits using this scheme, including a decoder, EXOR, EXNOR, binary to gray converter, and 2-bit comparator circuit, and verified all successfully.

• Considering technology scaling, we also implemented the same Boolean circuits using Cadence with 90nm and 22nm technology models, and conducted a thorough discussion of working, delay, average energy, EDP, and PDP parameters. Through graph, table comparison, and waveform, we conclude that this technique can be implemented in lower technology without significant issues.

CHAPTER-6
CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

This dissertation explores the potential of utilizing SRAM for in-memory computing, which is considered a promising option for future on-chip memory solutions. The project aims to develop Boolean and logic circuits based on in-memory computing principles. We examine the main characteristic of the Von Neumann architecture and emphasize the problem of memory read and write delays that are inherent in this system. In-memory computing enables basic operations to be executed directly within the memory, thereby minimizing time delays and improving memory speed. We have implemented fundamental logic gates and combinational circuits using this approach and evaluated the circuits' robustness, concluding with the performance parameters of SRAM.

• Chapters 3 and 4 detail the operation of 9T SRAM and the fundamentals of in-memory computing. • In Chapter 5, we initially calculated parameters such as read and write noise margins, leakage current, Ion/Ioff, read access time, write access time, and read energy. Based on these parameters, we determined that 9T SRAM outperforms 6T and 8T SRAM (using 250nm, VDD 5 Volt, and Tanner Tool).

• In subsequent work, we focused on optimizing the sizing of 9T SRAM for read and write operations in 250nm technology, facilitating easy implementation for other researchers or scholars. • We then implemented basic NAND and NOR logic gates using 9T SRAM with the in-memory computing scheme, successfully demonstrating Boolean logic circuit implementation.

• After implementing NAND, NOR, AND, and OR circuits, we compared the delay of all Boolean logic gates, finding that NAND had the highest delay and OR the lowest.

• We further implemented EXOR and EXNOR logic using NAND, NOR, AND, and OR gates, minimizing hardware usage. We calculated the delay for EXOR and EXNOR, noting that EXNOR had a higher delay due to its implementation with NAND gates.

• To further explore our work on in-memory computation-based combinational circuit design, we implemented a comparator, a 4-bit binary to gray converter, and a two-bit decoder, verifying the sensing scheme and concluding that the in-memory computing technique functions correctly for these digital circuits.

• For future work, we scaled the technology from 250nm to 90nm using the Cadence Virtuoso tool. With 90nm technology (VDD 1.8 Volt, gpdk 90nm), we implemented the same in-memory computation-based Boolean circuits and verified their functionality. We also calculated the delay, average power, PDP, and EDP of NAND, NOR, AND, and OR gates, comparing them in a table.

• For further advancements, we scaled our technology to 22nm using the PTM model library and Cadence Virtuoso software. We implemented NAND, NOR, AND, and OR logic with VDD at 1.2 volts, calculated delay, average power, EDP, and PDP parameters, and compared them in a table.

• In our subsequent work, we conducted simulations to evaluate the robustness of the circuit, implementing it on hardware for both optimal and suboptimal scenarios. We carried out PVT (Process, Voltage, and Temperature) variations for 250nm technology and found that our circuit functioned correctly for this technology. We then transitioned to 90nm technology, performing PVT variations and Monte Carlo simulations. Through PVT analysis, we assessed the circuit's robustness and concluded that it did not fail under any PVT parameter variations. The Monte Carlo simulation allowed us to determine the statistical range of RBL voltage, ensuring no overlap between VRBL(01/10) and VRBL(11), enabling the implementation of a NAND circuit without failure. We also determined that implementing a NOR gate posed no issues due to a larger voltage gap between VREF(01/10) and VREF(00). Additionally,

6.2 Future Work

In this project, we implemented in-memory computation-based Boolean and logic circuits. Numerous industries are working on real-time ad platforms, medical imaging processing, banking, and machine learning using in-memory computing techniques. Many companies, including major players like IBM and SAP, predict that in-memory computing will completely replace disk space dependence in the coming years. We can also explore IMC for resistive RAM (RRAM). We also authored and published a paper on this topic.

6.3 Social Impact

This project is about using a special type of memory called SRAM to do **In-Memory Computing (IMC)**, which means performing calculations directly inside the memory. This idea is very important today because the world is moving towards faster and more energy-efficient technology. With more people using smart devices, artificial intelligence, and working with huge amounts of data, our current computer systems are starting to struggle. Traditional systems, like the **Von Neumann architecture**, are slow because they keep moving data back and forth between memory and the processor — this causes delays. Our project helps solve this problem by letting the memory itself handle some of the computing, making things faster and more efficient.

References

[1] H. A. Du Nguyen, J. Yu, L. Xie, M. Taouil, S. Hamdioui, and D. Fey, “Memristive devices for computing: Beyond cmos and beyond vonneumann,” in*2017IFIP/IEEE International Conference on Very Large Scale Integration(VLSI-SoC)*. IEEE,2017, pp.1–10.

[2] M. M. Waldrop, “The chips are down for moore’s law,” *Nature News*, vol. 530, no. 7589, p. 144, 2016.

[3] J. G. D. Reinsel and J. Rydning, “Data age 2025:The evolution of data to life-critical dont focus on big data; focus on data that’s big,” in *2008 Device Research Conference*.IDC, Seagate, 2017, pp. 47–48.

[4] S. Poslad, “Ubiquitous computing: smart device , environments and interactions,” 2017.

[5] M.A.H.I.H.Witten,E.FrankandC.J.Pal,“Datamining: Practical machine learning tools and techniques,” *Acm Sigmod Record*, vol. 31, no. 1, pp. 76–77, 2016.

[6] J.Ahn,S.Hong,S.Yoo,O.Mutlu,andK.Choi,“Ascalableprocessing-in-memory accelerator for parallel graph processing,” in *Proceedings of the 42nd Annual Inter- national Symposium on Computer Architecture*, 2015, pp. 105–117.

[7] W.Dally,“High-performance hardware for machine learning,”*NIPSTutorial*,vol.2, 2015.

[8] K.Q.WeinbergerandL.K.Saul,“Distancemetriclearningforlargemarginnearest neighbor classification.”*Journal of machine learning research*, vol.10,no.2,2009.

[9] Y.DengandB.Manjunath,“Content-basedsearchofvideousingcolor,texture,and motion,” in *Proceedings of International Conference on Image Processing*, vol. 2. IEEE, 1997, pp. 534–537.

[10] X. Yin, A. Aziz, J. Nahas, S. Datta, S. Gupta, M. Niemier, and X. S. Hu, “Exploit- ing ferroelectric fets for low-power non-volatile logic-in-memory circuits,” in *2016 IEEE/ACM International Conference on Computer-Aided Design(ICCAD)*.IEEE, 2016, pp. 1–8. workshop,”*IEEEMi- cro*, vol. 34, no. 4, pp. 36–42, 2014.

[11] R.Balasubramonian,J.Chang,T.Manning,J. H.Moreno,R.Murphy,R. Nair,and S.Swanson,“Near-data processing: Insights from a micro-46

[12] T. Jiang, Q. Zhang, R. Hou, L. Chai, S. A. Mckee, Z. Jia, and N. Sun, “Understand- ing the behavior of in-memory computing workloads,” in *2014 IEEE International Symposium on Workload Characterization (IISWC)*.IEEE, 2014, pp. 22–30.

[13] http://electricaltechnology.org.

[14] G. Singh,L. Chelini,S. Corda,A. J. Awan,S. Stuijk,R. Jordans,H. Corporaal, andA.-J.Boonstra,“Are view of near-memory computing architectures: Opportuni- ties and challenges,” in *2018 21st Euromicro Conference on Digital System Design (DSD)*.IEEE, 2018, pp. 608–617.

[15] https://semiengineering.com/in-memory-vs-near-memory-computing/.

[16] “http://ee380.stanford.edu/http://web.stanford.edu/class/ee380/Abstracts/180307. html.”

[17] C.Li, D.Belkin, Y.Li,P.Yan, M.Hu,N.Ge,H.Jiang, E.Montgomery, P.Lin, Z.Wang*etal.*,“In-memory computing with memristorarrays,”*Inter-national Memory Workshop (IMW)*.IEEE, 2018, pp. 1–4.

[18] G.De Micheli, *Synthesis and optimization of digital circuits*.McGrawHill,1994, no. BOOK.

[19] R.L.RudellandA.Sangiovanni-Vincentelli,“Multiple-valuedminimizationforpla optimization,”*IEEE Transactionson Computer- Aided Design of Integrated Circuits and Systems*, vol. 6, no. 5, pp. 727–750, 1987.

[20] J.Singh,D.K.Pradhan,S.Hollis,S.P.Mohanty,andJ.Mathew,“Singleended 6t sram with isolated read-port for low-power embedded systems,” in *2009 Design, Automation & Test in Europe Conference & Exhibition*.IEEE,2009,pp.917–922.

[21] D. Mukherjee, H. K. Mondal, and B. Reddy, “Static noise margin analysis of sram cell for high speed application,” *International Journal of Computer Science Issues (IJCSI)*, vol. 7, no. 5, p. 175, 2010.

[22] L. Rajesh and J. Santosh, “Design and power analysis of 8t sram cell using charge sharing technique,”*International Journal of Innovative Research in Electronics and Communications*, vol. 3, no. 1, pp. 20–26, 2016.

[23] A.K.Rajputand M.Pattanaik,“Implementation of boolean and arithmetic functions with 8t sram cell for in-memory computation,”in *2020 International Conference for Emerging Technology (INCET)*.IEEE, 2020, pp. 1–5.

[24] A.Agrawal,A.Jaiswal,C.Lee,andK.Roy,"X-sram: Enabling in-memory boolean computations in cmos static random access memories," *IEEE Transactions on Cir- cuits and Systems I: Regular Papers*, vol. 65, no. 12, pp. 4219–4232, 2018.

[25] S.-M.KangandY.Leblebici,*CMOS digital integrated circuits*.Tata McGraw-Hill Education, 2003.

[26] J.P.Kulkarni,A.Goel,P.Ndai,andK.Roy,"Aread-disturb-free,differentialsensing1r/1wport,8tbitcellarray,"*IEEEtransactionsonverylargescaleintegration(VLSI) systems*, vol. 19, no. 9, pp. 1727–1730, 2010.

[27] J. M. Rabaey, A. P. Chandrakasan, and B. Nikolić,*Digital integrated circuits:a design perspective*.Pearson education Upper Saddle River, NJ, 2003, vol. 7.

[28] M.PattanaikandA.K.Rajput, "Energy efficient 9t sram with r/w margin enhanced for beyond von-neumann computation," IEEE, pp. 1–4, 2020.

[29] A.Agrawal,A.Jaiswal,C.Lee,andK.Roy,"X-sram: Enabling in-memory boolean computations in cmos static random access memories," *IEEE Transactions on Cir- cuits and Systems I: Regular Papers*, vol. 65, no. 12, pp. 4219–4232, 2018.

[30] A. Jaiswal, I. Chakraborty, A. Agrawal, and K. Roy, "8t sram cell as a multi bit dot- product engine for beyond von neumann computing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 11, pp. 2556–2567, 2019.

[31] A.K.GuptaandA.Acharya,"Exploration of 9t sramcell for in memory computing application," in *2021 Devices for Integrated Circuit (DevIC)*.IEEE, 2021.

[32] B. Chen, F. Cai, J. Zhou, W. Ma, P. Sheridan, and W. D. Lu, "Efficient in-memory computing architecture based on cross bararrays,"in*2015IEEEInternationalElec- tron Devices Meeting (IEDM)*.IEEE, 2015, pp. 17–5.

[33] B. Wicht, T. Nirschl, and D. Schmitt-Landsiedel, "Yield and speed optimization of a latch-type voltage sense amplifier," *IEEE Journal of Solid-State Circuits*, vol. 39, no. 7, pp. 1148–1158, 2004.

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

PLAGIARISM VERIFICATION

Title of the Thesis In **Memory Computation based Boolean and Logic Circuit Design**

Total Pages **34,** Name of the Scholar **RAM LAKHAN**

Supervisor(s)

(1) Assistant Prof. AKSHAY MANN

Department **Electronics and Communication**

This is to report that the above thesis was scanned for similarity detection. Process and outcome is given below:

Software used: **turnitin** Similarity Index: **13 %**

Total word count: **9004**

Date: **28 may 2025**

Candidate's Signature **Signature of Supervisor(s)**

Ram

final_thesis_pdf (1)_______25 (1) (1) (3)_removed.pdf

Delhi Technological University

Document Details

Submission ID

trn:oid:::27535:98221302

Submission Date

May 28, 2025, 11:43 PM GMT+5:30

Download Date

May 28, 2025, 11:45 PM GMT+5:30

File Name

final_thesis_pdf (1)...........25 (1) (1) (3)_removed.pdf

File Size

1.7 MB

34 Pages

9,004 Words

45,894 Characters

13% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 8 words)

Match Groups

- **107** Not Cited or Quoted 13%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

4%	Internet sources
8%	Publications
7%	Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **107** Not Cited or Quoted 13%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

4%	Internet sources
8%	Publications
7%	Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Internet	varys.ucsd.edu	2%
2	Submitted works	Sardar Vallabhbhai National Inst. of Tech.Surat on 2022-02-06	1%
3	Publication	Aman Kumar Gupta, Prashant Joshi, Shobhit Srivastava, Sourabh Panwar et al. "9...	1%
4	Publication	Anil Kumar Rajput, Manisha Pattanaik. "Implementation of Boolean and Arithmet...	<1%
5	Publication	Anil Kumar Rajput, Manisha Pattanaik. "Energy Efficient 9T SRAM With R/W Margi...	<1%
6	Internet	www.coursehero.com	<1%
7	Publication	Aman Kumar Gupta, Abhishek Acharya. "Exploration of 9T SRAM Cell for In Memo...	<1%
8	Submitted works	CSU, Long Beach on 2021-02-12	<1%
9	Publication	Shilpi Birla, Shashi Kant Dargar, Neha Singh, P. Sivakumar. "Low Power Designs i...	<1%
10	Internet	tikansari.wordpress.com	<1%

11	Submitted works	
	B.V. B College of Engineering and Technology, Hubli on 2024-11-25	<1%
12	Submitted works	
	Higher Education Commission Pakistan on 2014-11-20	<1%
13	Internet	
	research.ijcaonline.org	<1%
14	Submitted works	
	87988 on 2014-09-17	<1%
15	Submitted works	
	V.B.S Purvanchal University, Jaunpur on 2024-11-30	<1%
16	Publication	
	"Design and Power Analysis of 8T SRAM Cell Using Charge Sharing Technique", In...	<1%
17	Submitted works	
	ABV-Indian Institute of Information Technology and Management Gwalior on 201...	<1%
18	Submitted works	
	M S Ramaiah University of Applied Sciences on 2025-02-12	<1%
19	Submitted works	
	CCS Haryana Agricultural University Hisar on 2022-07-22	<1%
20	Submitted works	
	University of Nottingham on 2023-04-17	<1%
21	Submitted works	
	University of Florida on 2017-12-07	<1%
22	Internet	
	ia902309.us.archive.org	<1%
23	Publication	
	Elena I. Vătăjelu, Álvaro Gómez-Pau, Michel Renovell, Joan Figueras. "Sram cell st...	<1%
24	Internet	
	coek.info	<1%

25	Internet	
	www.ijert.org	<1%
26	Internet	
	www.ir.juit.ac.in:8080	<1%
27	Submitted works	
	Cranfield University on 2013-01-13	<1%
28	Submitted works	
	University of Colorado, Denver on 2021-11-29	<1%
29	Submitted works	
	Birla Institute of Technology on 2014-08-19	<1%
30	Submitted works	
	Imperial College of Science, Technology and Medicine on 2012-12-03	<1%
31	Publication	
	Nurul Ezaila Alias, Afiq Hamzah, Michael Loong Peng Tan, Usman Ullah Sheikh, M...	<1%
32	Publication	
	Sushmita Jaiswal, Santosh Kumar Gupta. "Proposal and Analysis of a High Read a...	<1%
33	Submitted works	
	University of North Texas on 2010-08-24	<1%
34	Publication	
	"Proceedings of the 2nd International Conference on Data Engineering and Com...	<1%
35	Publication	
	"VLSI Design and Test", Springer Science and Business Media LLC, 2019	<1%
36	Submitted works	
	California State University, Fresno on 2019-03-12	<1%
37	Publication	
	Changhwan Shin. "Variation-Aware Advanced CMOS Devices and SRAM", Springer...	<1%
38	Publication	
	Lin, Sheng. "Analysis and Design of Robust Storage Elements in Nanometric Circu...	<1%

39 Publication

M. Izumikawa, M. Yamashina. "A current direction sense technique for multiport ... <1%

40 Publication

M. YOSHIMOTO. "Area Comparison between 6T and 8T SRAM Cells in Dual-Vdd Sc... <1%

41 Publication

N.M. Sivamangai, K. Gunavathi. "A Low Power SRAM Cell with High Read Stability"... <1%

42 Submitted works

V.B.S Purvanchal University, Jaunpur on 2024-12-02 <1%

43 Submitted works

VIT University on 2011-04-11 <1%

44 Submitted works

VIT University on 2016-04-07 <1%

45 Publication

Vivek Kumar, V. K. Tomar. "A Comparative Performance Analysis of 6T, 7T and 8T ... <1%