SRAM BASED IN MEMORY COMPUTING DESIGN FOR EDGE DEVICES

A Project Report Submitted

in Partial Fulfilment of the Requirements

for the Degree of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS & COMMUNICATION ENGINEERING

by

Adusumilli Laahiri - 2021BEC0015

Indian Institute of
Information Technology
Kottayam

to

DEPARTMENT OF ELECTRONICS &

COMMUNICATION ENGINEERING

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY

KOTTAYAM-686635, INDIA

April 2025

DECLARATION

I, **Adusumilli Laahiri** (**2021BEC0015**) hereby declare that, this report entitled **“SRAM Based In Memory Computing Design for Edge devices”** submitted to Indian Institute of Information Technology Kottayam towards partial requirement of **Bachelor of Technology** in **Electronics and Communication Engineering** is an original work carried out by me under the supervision of **Dr. Lakshmi N. S.** and has not formed the basis for the award of any degree or diploma, in this or any other institution or university. I have sincerely tried to uphold the academic ethics and honesty. Whenever an external information or statement or result is used then, that have been duly acknowledged and cited.

Kottayam-686635 **Adusumilli Laahiri** (**2021BEC0015**)

CERTIFICATE

This is to certify that the work contained in this project report entitled **"SRAM Based In Memory Computing Design for Edge devices"** submitted by **Adusumilli Laahiri** (**2021BEC0015**) to the Indian Institute of Information Technology Kottayam towards partial requirement of **Bachelor of Technology** in **Indian Institute of Information Technology Kottayam** has been carried out by her under my supervision and that it has not been submitted elsewhere for the award of any degree.

Kottayam-686635 Dr. Lakshmi N. S.
April 2025 Project Supervisor

ACKNOWLEDGEMENT

I wish to thank to our project supervisor, Dr. Lakshmi N. S., Assistant Professor in the Department of Electronics and Communication Engineering (ECE), for her invaluable guidance and support throughout the project. Her insightful direction and research-driven approach greatly inspired and motivated me. I am extremely appreciative of her time, knowledge, and dedication.

I also extend our heartfelt thanks to our project evaluation panel members: Dr.Jalaja M.J, Adjunct Professor, Dr.Milind Thomas, Assistant Professor, Dr. Della Thomas, Assistant Professor, Dept. of ECE. Their constructive feedback and encouraging words were instrumental in refining my project and enhancing my understanding. I am grateful for their contributions to my academic growth and professional development.

Kottayam-686635 **Adusumilli Laahiri (2021BEC0015)**

ABSTRACT

This project presents the conceptualization and design of an SRAM-based In-Memory Computing (IMC) architecture for edge computing devices. The architecture consists of upgraded 6T SRAM cells with additional transistors to enable direct XOR logic operation computation in the memory array. A 4×4 array of conventional 6T SRAM cells and on-chip peripheral circuits like a sense amplifier, precharge circuit, write driver, and decoder were designed in the first phase to enable in-memory computation.

Based on this foundation in this phase, the XOR logic was also mapped to the SRAM array, and thorough stability analysis was conducted for both read and hold operations in the 6T cell for assured functionality. The 6T cell physical design was finished, followed by Design Rule Check (DRC) and Layout Versus Schematic (LVS) checks. Additionally, a layout for the implementation of a 2-cell XOR was built and successfully checked via DRC and LVS checks. This demonstrates a full design flow from logic-level mapping to physical verification, proving the viability of logic-in-memory operations in SRAM arrays for edge computing applications.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

The limitations of the traditional Von Neumann computing architecture are becoming increasingly apparent, particularly due to the growing energy and performance costs associated with data movement. In contrast, computation itself remains relatively energy-efficient. This disparity becomes especially significant in data-intensive applications, where frequent memory access leads to excessive energy consumption. To address this challenge, there has been a growing shift toward In-Memory Computing (IMC) — a computing paradigm in which data processing occurs directly within the memory array, thereby minimizing data transfer and its associated inefficiencies.

Static Random-Access Memory (SRAM), a widely used type of memory across various electronic systems, emerges as a strong candidate for implementing IMC. With appropriate integration into existing hardware, SRAM-based IMC can significantly enhance system performance while reducing energy consumption — making it especially attractive for energy-sensitive edge computing applications.

The implementation process begins at the circuit level, where SRAM cells and supporting logic circuits are designed and simulated to validate their functional behavior and performance metrics. Once validated, these schematics are translated into physical layouts using standard VLSI (Very Large Scale Integration) design tools. The layout phase involves precise transistor placement and routing, ensuring adherence to foundry-specific design rules.

To confirm that the layout can be reliably fabricated and matches the intended design, two crucial verification steps are performed:

Design Rule Check (DRC): Confirms that the physical layout abides by the manufacturing constraints defined by the fabrication technology.

Layout Versus Schematic (LVS): Ensures that the layout-derived netlist corresponds exactly to the original schematic netlist, preserving the intended circuit functionality.

In addition to layout design and verification, stability analysis is conducted using Static Noise Margin (SNM). SNM serves as an important metric to assess the reliability of SRAM cells, particularly under read and hold conditions. It measures the maximum tolerable noise voltage that does not alter the cell's stored state. Butterfly curves, derived from the DC transfer characteristics of the cross-coupled inverters within the SRAM cell, are used to compute SNM—the size of the largest square that can fit inside the curve's lobes represents this value.

By combining detailed circuit design, layout implementation, verification, and stability analysis, the project delivers a reliable and efficient in-memory computing system. This holistic design approach supports the development of low-power, high-speed computing solutions well-suited for modern, data-centric applications.

1.2 Problem Statement

Developing an energy-efficient SRAM-based In-Memory Computing (IMC) architecture tailored for edge devices calls for an innovative strategy that overcomes the drawbacks of conventional Von Neumann systems. In standard computing architectures, frequent data transfers between memory and processing units introduce significant latency and energy costs, which severely limit the performance and scalability of resource-constrained edge applications where low power consumption is essential.

To address these inefficiencies, the proposed architecture focuses on executing computations directly within the SRAM memory array, effectively reducing the need for data transfer and thus lowering both latency and power consumption. For successful deployment in edge computing environments, the system must strike a careful balance between energy efficiency, computational performance, and accuracy.

Furthermore, the design must also take into account the unique challenges associated with IMC, such as ensuring memory stability, minimizing the area and power overheads introduced by extra computing circuitry, and preserving the overall reliability of memory operations.

Chapter 2

Literature Survey

In recent years, various studies have explored the integration of logic operations within memory arrays to enhance computational efficiency, especially for applications requiring high energy efficiency and low latency.

[1] presents a low-energy XNOR-SRAM macro that performs ternary XAC operations for binary and ternary neural networks. This design not only maintains the accuracy of high-precision models but also significantly improves energy efficiency, achieving 403 TOPS/W and 88.8% CIFAR-10 accuracy with a 256×64 XNOR-SRAM configuration, which operates in a single cycle.

Similarly, [2] introduces an innovative compute-in-memory (CIM) design using an 8T SRAM structure, which is highly power-efficient for executing neural networks, specifically for XNOR-Net and Ternary-Binary Networks (TBNs). The design incorporates two additional transistors to the traditional 6T SRAM cell, enabling the memory to perform internal XNOR computations for ternary inputs and binary outputs, making it capable of direct vector-matrix multiplication inside the memory array.

This architecture accumulates the analog output voltages from XNOR operations along shared vertical columns, with a provision for digitization via an ADC, improving both energy efficiency and computational speed.

In another approach, [3] compares various SRAM-based in-memory computing (IMC) designs, concluding that while 8T SRAMs offer superior energy efficiency due to their skewed inverters, they occupy more chip area compared to 6T cells.

The paper suggests the exploration of 9T and 10T SRAM cells, as well as non-volatile options like BCAM, TCAM, and MRAM, as potential candidates for future applications in high-performance computing and sectors requiring intensive data processing, such as autonomous vehicles and intelligent transportation systems.

The research presented in [4] focuses on in-memory computing using commercial CMOS bit-cells, including 8T and 8+T SRAMs. These bit-cells facilitate the execution of Boolean functions such as NAND, NOR, and XOR through minimal modifications to peripheral circuits.

The proposed ”Read-Compute-Store” scheme, which exploits the decoupled read and write ports of these cells, reduces memory transactions by up to 75%, resulting in substantial energy savings and performance improvements, particularly for applications such as AES encryption.

Furthermore, [5] explores the potential of 8T SRAM cells for in-memory computing applications, with a proposed sensing scheme that enhances the robustness of NAND and NOR operations. The high read margin and energy efficiency of the 8T SRAM cells make them promising candidates for future computational systems, offering the feasibility of conducting computations directly within the memory array.

[6] presents a PSRAM chip that can perform a full set of Boolean logic operations—including NOR, NAND, XOR, majority, and full adder—within a single memory cycle. Demonstrated through case studies in parallel vector operations, neural networks, and data encryption, this PSRAM design paves the way for high-speed, energy-efficient in-SRAM computing.

Moreover, [7] introduces the P9T SRAM, a read/write margin-enhanced SRAM cell designed for beyond von-Neumann computing applications. This cell not only supports traditional memory functions but also performs Boolean logic operations such as NAND, AND, NOR, OR, and XOR.

The P9T SRAM offers significant improvements, including a 3.5x higher read static noise margin and 51.5% reduction in read energy, making it an ideal candidate for energy-efficient, high-performance computing systems.

A lightweight CIM macro for deep neural networks (DNNs) is proposed in [8], which integrates SRAM-based physical unclonable functions (PUFs) to protect DNN models in edge computing environments. The design features a 10T SRAM cell capable of supporting configurable XOR and MAC operations.

The test chip, fabricated in a 55nm CMOS process, achieves an impressive throughput of 1324.24 GOPS and an energy efficiency of 98.55 TOPS/W, with a MNIST accuracy of 96.20%. Although the encrypted CIM design offers enhanced security, further studies are needed to develop more robust security solutions with minimal accuracy degradation.

Chapter 3

Architecture

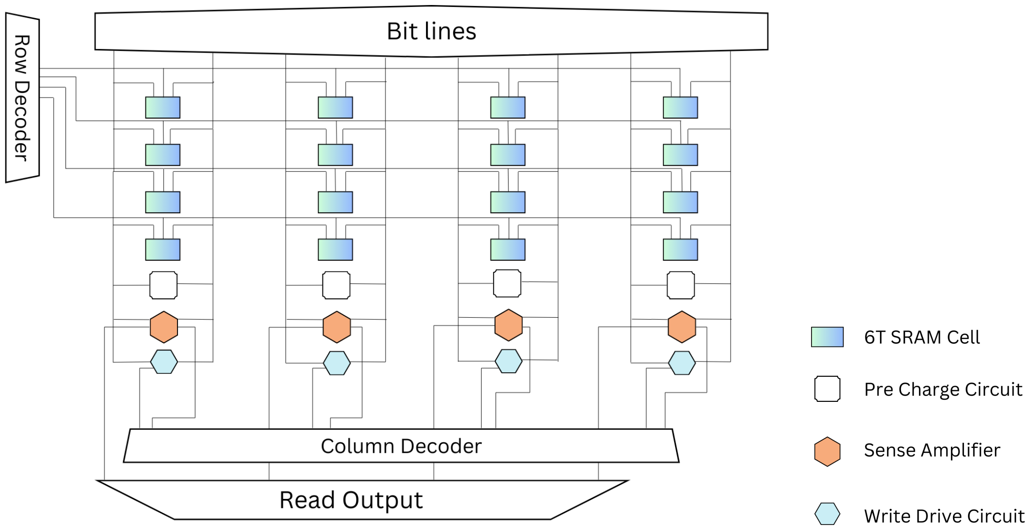


Figure 3.1: 4X4 SRAM Memory Array

3.1 6T SRAM Cell

A 6T SRAM cell is a data storage memory cell that uses a latch made of two cross-coupled inverters and access transistors. ”6T” in 6T SRAM refers to the six transistors (4 NMOS and 2 PMOS) that constitute each cell. It has quick access to data and low power, which is ideal for processor cache memory usage.

3.1.1 Structure of 6T SRAM Cell

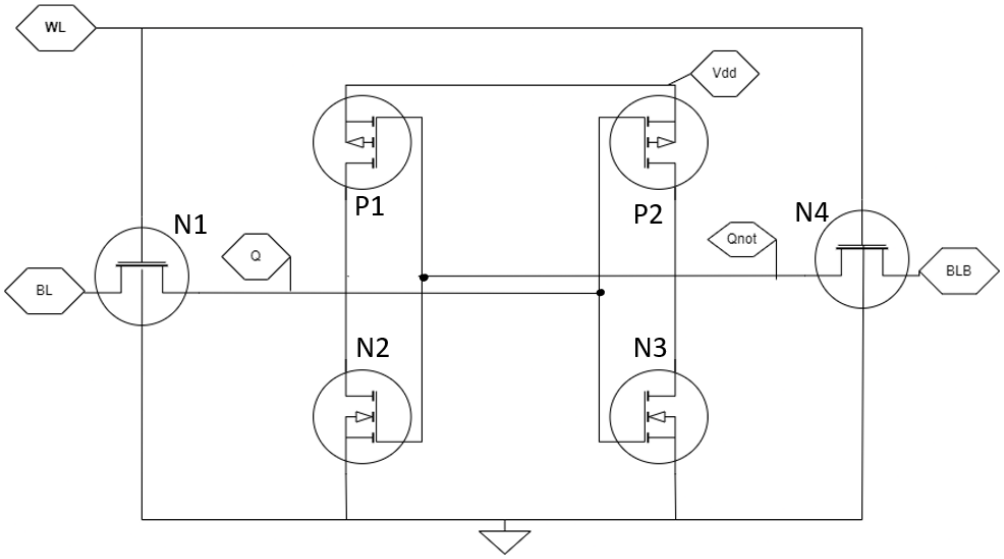


Figure 3.2: Conventional 6T SRAM cell

- **Cross-Coupled Inverters:** The heart of a 6T SRAM cell is two inverters in a feedback loop. This configuration connects each inverter output to the other’s input to create a bistable latch that can hold one bit of data (0 or 1) as long as power is being supplied.

The inverters consist of PMOS and NMOS transistors. Here, PMOS transistors in each inverter pull up the voltage when storing a logic '1' at the respective node.

- **Access Transistors:** Two NMOS transistors (N1 and N4) act as access transistors, connecting the SRAM cell with the bit lines (BL and BLB). The word line (WL) controls the transistors. When the word line is asserted (set to a high voltage), we can access the SRAM cell for read or write operations.

3.1.2 Modes of Operation

The three main modes of operation of 6T SRAM bitcell : Hold, Read, and Write. Each mode depends on the state of the word line (WL) and the state of the bit lines.

1. Hold Mode

In Hold mode, the word line (WL) is not asserted (WL=0), which turns off the access transistors N1 and N4. This isolates the SRAM cell from the bit lines, preventing any external influence on the stored data. In this state:

The cross-coupled inverters continue reinforcing each other's state, allowing the cell to retain the stored data (either 0 or 1). As long as power is supplied, the data is preserved due to the feedback loop within the inverters.

2. Read Mode

In Read mode, the 6T SRAM cell is accessed to retrieve the stored data. Here, the word line (WL) is asserted (WL=1), enabling access transistors N1 and N4, which connect the cell to the bit lines (BL and BLB) as follows:

The values stored in nodes (Q and Qb) are read at the bit lines via the access transistors. For example, if a '1' is stored at Q, BLB will discharge through NMOS transistors N3 and N4, while BL is pulled up through the PMOS transistor in the corresponding inverter. A sense amplifier is connected to the bit lines to detect the difference of voltage between BL and BLB to determine the stored bit. To ensure "read stability," the design must prevent the stored state from changing during reading. This stability is maintained by balancing the strengths of the NMOS and PMOS transistors in the inverters relative to the access transistors.

3. Write Mode

In Write mode, the data is being written to the SRAM cell by setting the bit lines and asserting the word line. To write a '0' at node Q:

BL is set to 0V (low), and BLB is set to VDD (high). The word line is asserted, which turns on access transistors N1 and N4, which connect the bit lines BL and BLB to Q and Qb. Since N1 is directly connected to Q, it pulls Q down to 0V, overpowering the PMOS transistor in the inverter that was previously holding it at a high state. This forces Q to 0V and Qb to VDD, effectively storing a '0' at Q.

3.1.3 Stability Analysis

Stability analysis is a critical part of SRAM design, ensuring that the memory cell reliably retains and accesses stored data under various operating conditions. The stability of a 6T SRAM cell is typically evaluated using the Static Noise Margin (SNM), which quantifies the cell's resilience to noise or disturbances.

In the hold state, the wordline is deactivated, and the cell is isolated from the bitlines, simply retaining its stored data. Here, SNM reflects the inherent robustness of the cross-coupled inverters without external influence.

In the read state, however, the wordline is enabled, and the access transistors connect the internal nodes to the bitlines. This interaction can cause a voltage disturbance on the internal node storing '0', making it the most vulnerable condition for potential data corruption.

Therefore, read SNM is usually lower than hold SNM, and its analysis is essential to ensure read stability—preventing accidental bit flips during access.

Since write stability is characterized differently (using Write Margin or Write Trip Point), and XOR operations in this work are performed during the read phase, stability analysis is limited to the hold and read conditions. Evaluating SNM in these two states provides meaningful insights into the robustness and reliability of the SRAM cell under real-world operating scenarios.

3.2 Peripheral Circuits

3.2.1 Pre charging Circuit

Pre-Charge Activation: The pre-charge circuit is activated before any read or write operation, setting the bit lines BL and BLB to an intermediate voltage, typically VDD/2 (half the supply voltage). This pre-charging step ensures that both bit lines start from a known, equal voltage level, which reduces power consumption and improves the speed of subsequent operations.

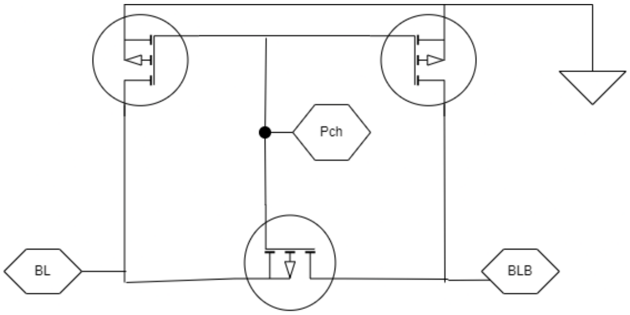


Figure 3.3: Pre charging circuit

Equalization of Bit Lines: During the pre-charge cycle, an equalizer circuit shorts the two bit lines, ensuring that there is no initial voltage difference between them. This equalization is essential because it allows the sense amplifier to detect small voltage differences during the read operation with high accuracy.

Transition to Read or Write: Once the pre-charge cycle completes, the pre-charge circuit is switched off, isolating the bit lines. At this point, the bit lines are ready for read or write operations.

3.2.2 Sense Amplifier

In SRAM cells, the bit stored in memory is determined by the voltage difference across the bit lines (BL and BLB). However, due to the large number of cells connected within each column, obtaining a clear output directly from the bit lines can be challenging. To address this, a differential sense amplifier is employed, which enhances both stability and accuracy.

The differential sense amplifier detects the small voltage difference between the bit lines and amplifies it to provide a clear, stable output.

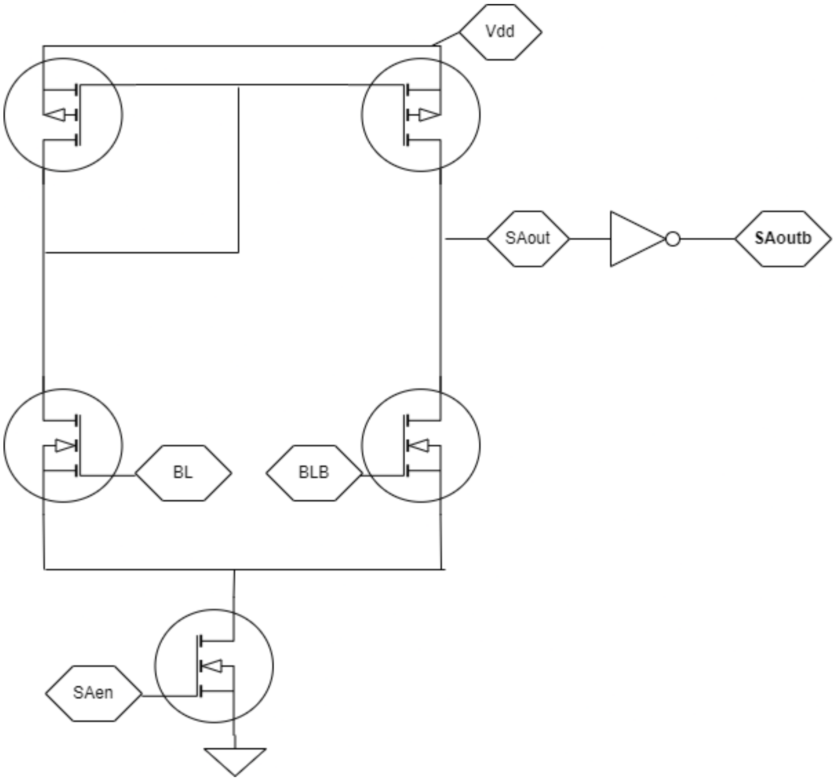


Figure 3.4: Differential Sense Amplifier

Sense amplifier is activated only when the sense input is high, minimizing power consumption by operating only during read cycles.

Cross-Coupled Inverter Pair: When the sense clock is high, it triggers a cross-coupled inverter pair within the differential sense amplifier. This structure utilizes regenerative feedback. The differential sense amplifier improves read speed, data accuracy, and power efficiency.

3.2.3 Write Driver Circuit

The write driver, also known as the driver circuit, is crucial for writing data onto the bit lines. The primary role of the write driver is to pull the bit line (BL) and bit line bar (BLB) to ground potential, which allows for new data to be written to the memory cell.

Initially, both bit lines are pre-charged to the supply voltage (VDD/2) by the pre-charge circuit. The write driver is activated by the Write Enable (Wen) signal; only when Wen is high will the write driver assert data onto the bit lines, ensuring that the data transfer occurs at the right moment.

3.2.4 Row and Column Decoders

In an SRAM memory array, data is organized in rows and columns. Row access is controlled by word lines, while column access is controlled by bit lines.

In a read or write operation, making the word line high grants access to the entire row associated with that line. Both row and column decoders are necessary to select the precise word in the array, based on the row and column addresses provided.

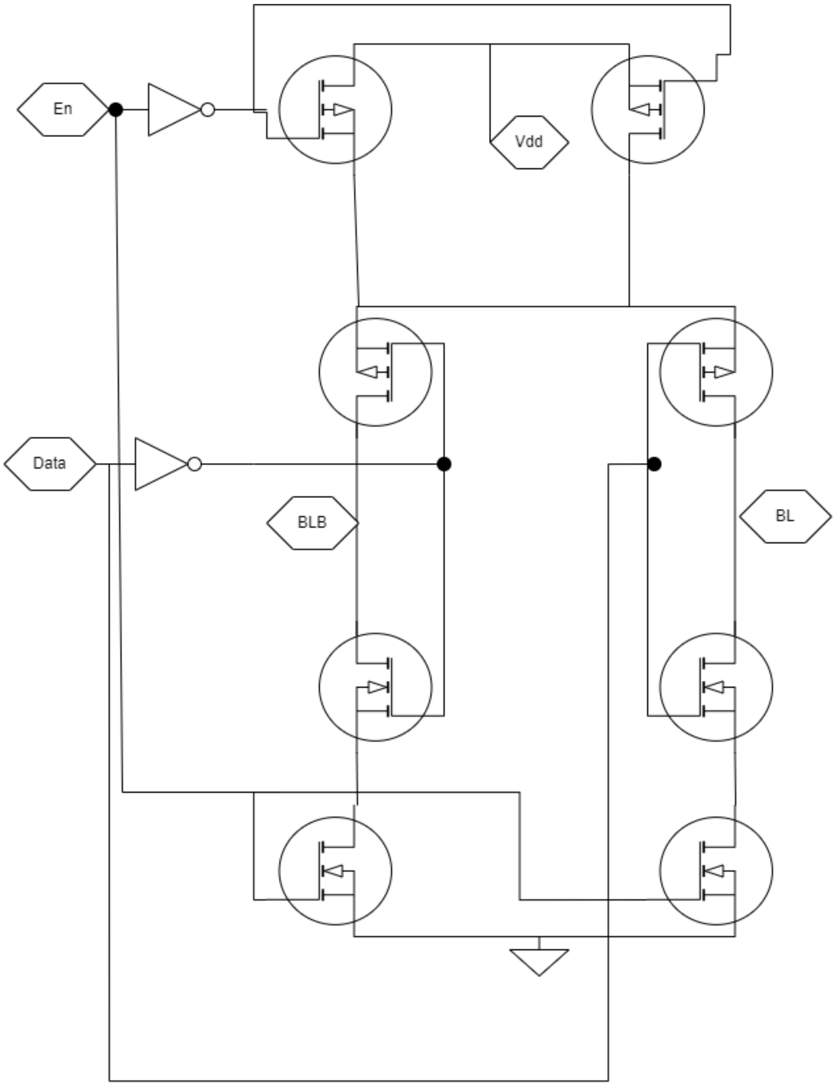


Figure 3.5: Write driver circuit

Chapter 4

Proposed Work

This project aims to implement XOR logic using a 6T SRAM cell, enabling efficient in-memory computing for XOR operations. To achieve this, two specialized circuits inspired by pass transistor logic are proposed. These structures take advantage of the functional capabilities of the SRAM cell structure with very few hardware additions and provide efficient and compact solutions for the XOR function. Bringing the XOR logic operation circuitry inside the memory array serves to cut back on the movement of data for processing which in turn improves energy and speed performance, which are critical for the modern data driven workloads.

4.1 XOR implementation using 4 additional transistors

4.1.1 2 6T SRAM cells used for XOR

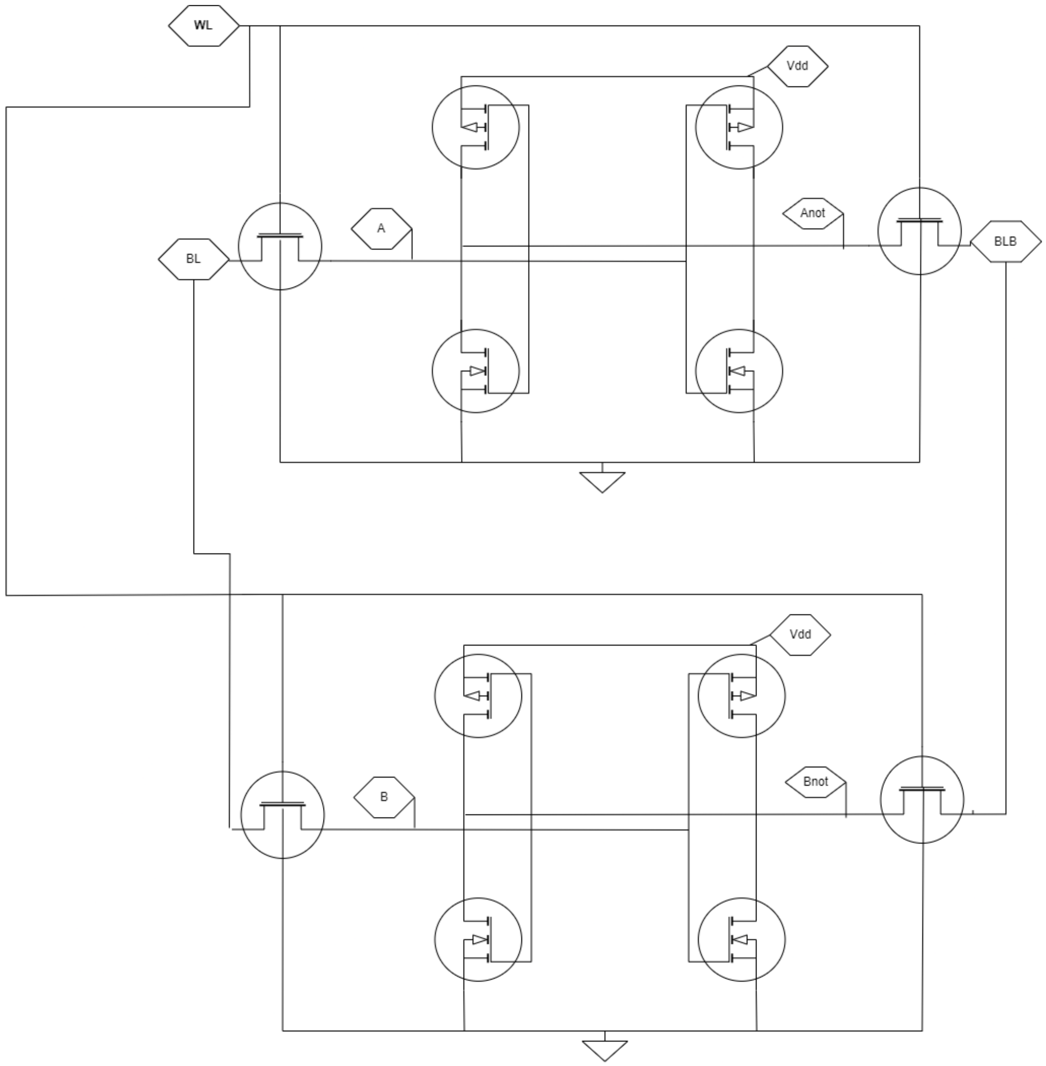


Figure 4.1: 2 6T cells

4.1.2 XOR

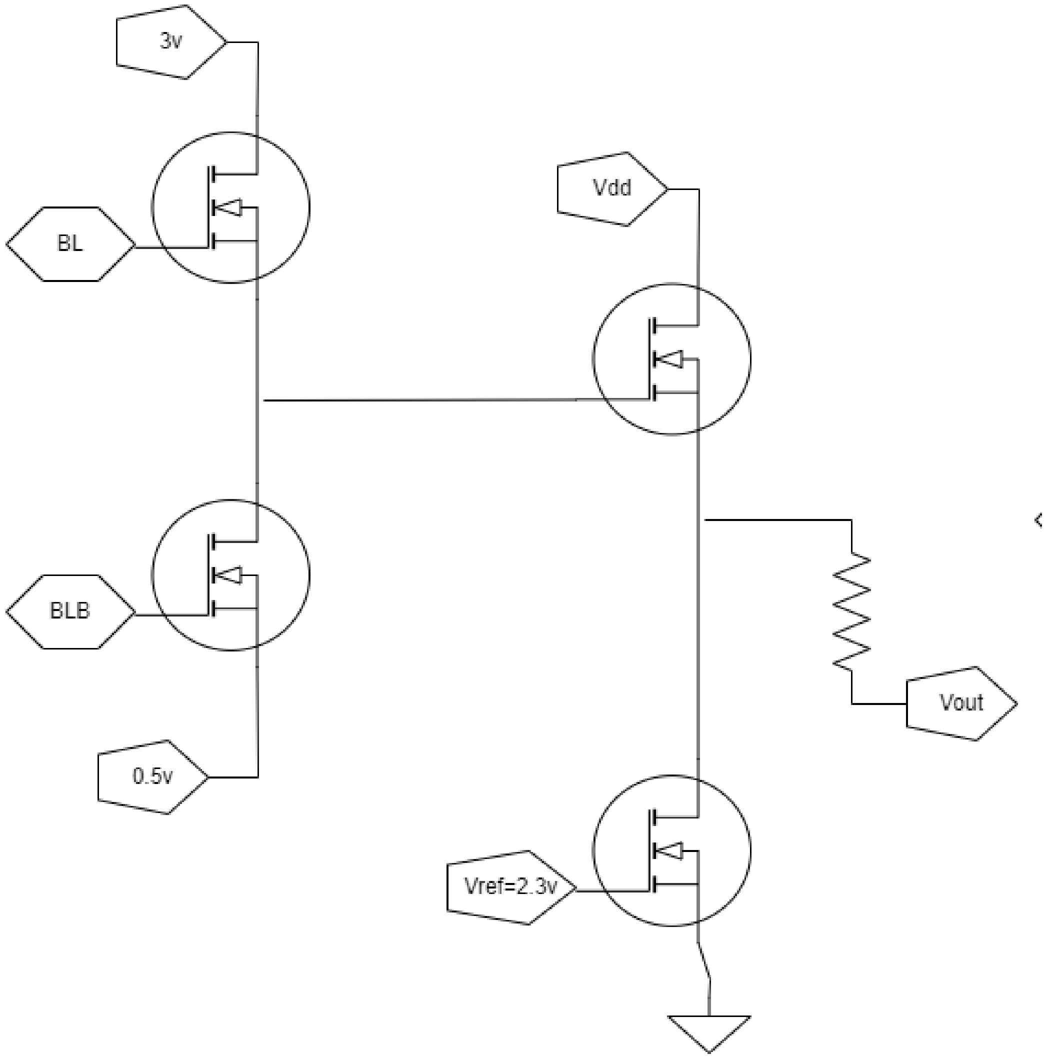


Figure 4.2: Proposed Circuit 1

4.1.3 4X4 SRAM Memory Array with XOR

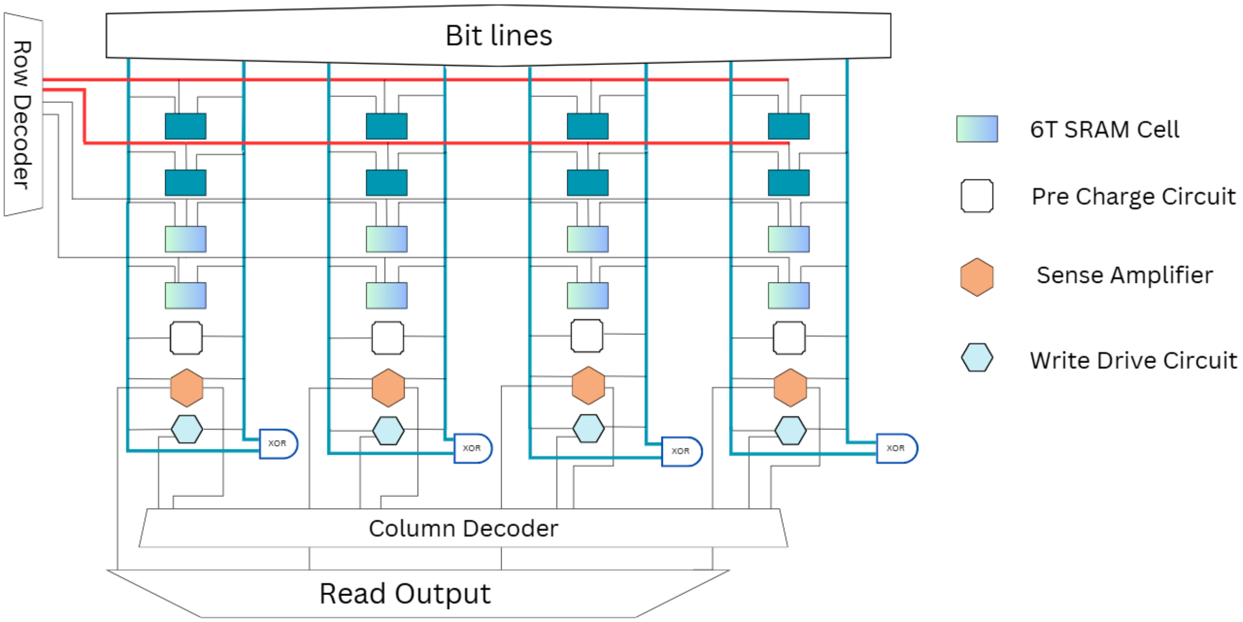


Figure 4.3: XOR circuit for 2 cells in 4X4 array

4.1.4 XOR for array

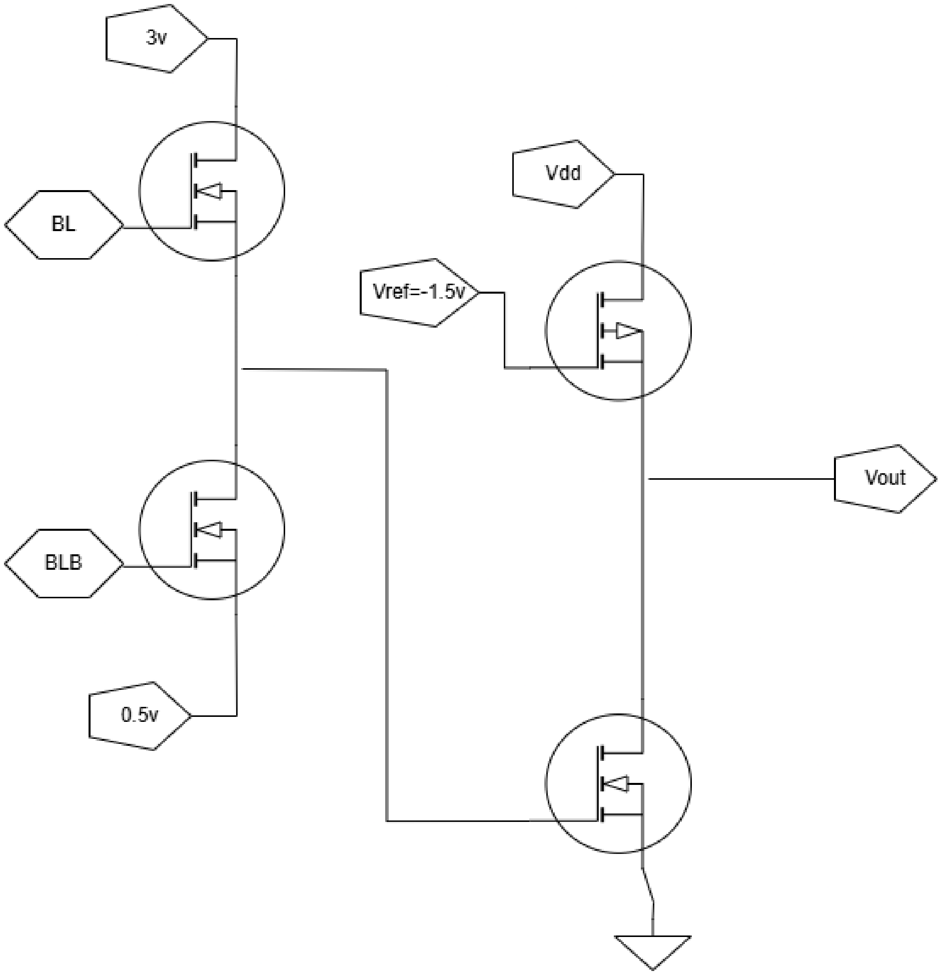


Figure 4.4: Proposed Circuit 1 modified for array

4.2 XOR implementation using 2 additional transistors

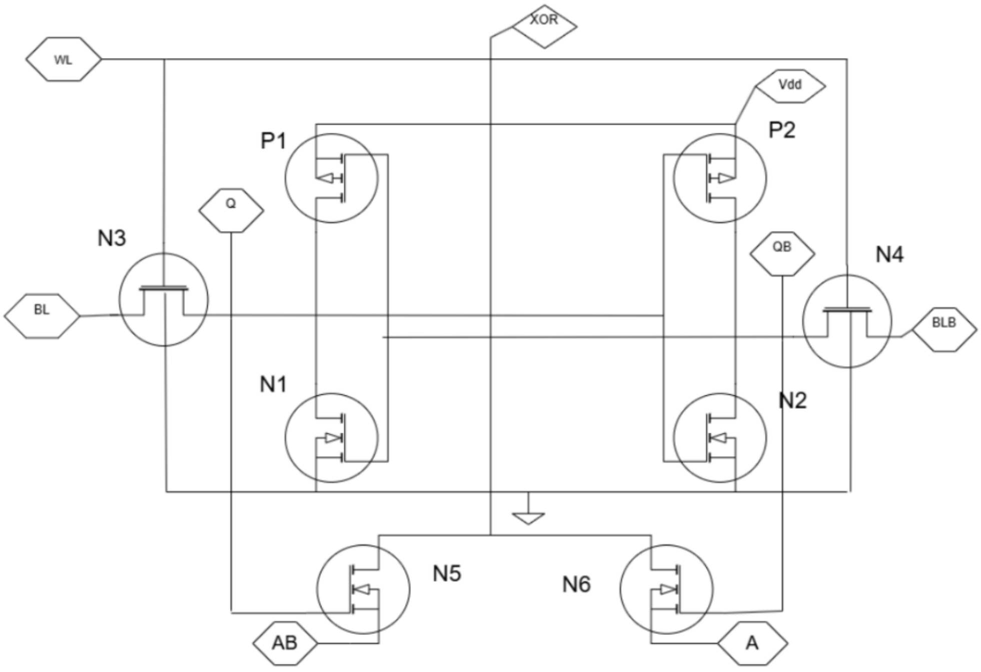


Figure 4.5: Proposed Circuit 2

Chapter 5

Simulation Results

5.1 Schematic

5.1.1 Peripheral Circuits

In a 6T SRAM cell, peripheral circuits are essential for the operation, control, and interfacing of the memory array. These circuits are not part of the 6T cell itself but are critical for making the memory functional in a real system

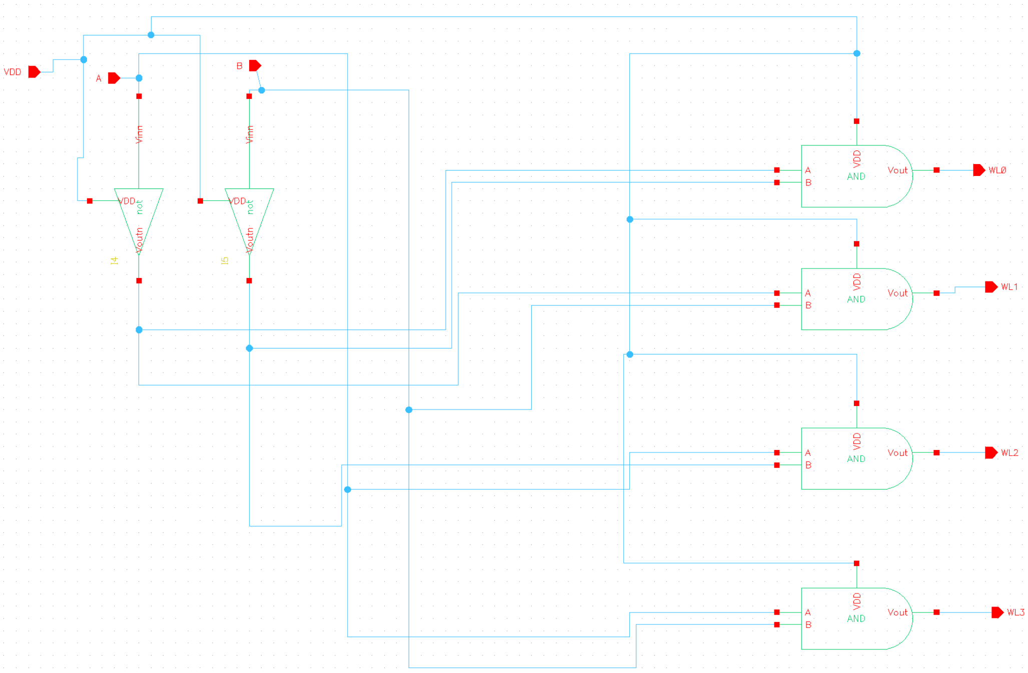


Figure 5.1: 2-to-4 Decoder Circuit

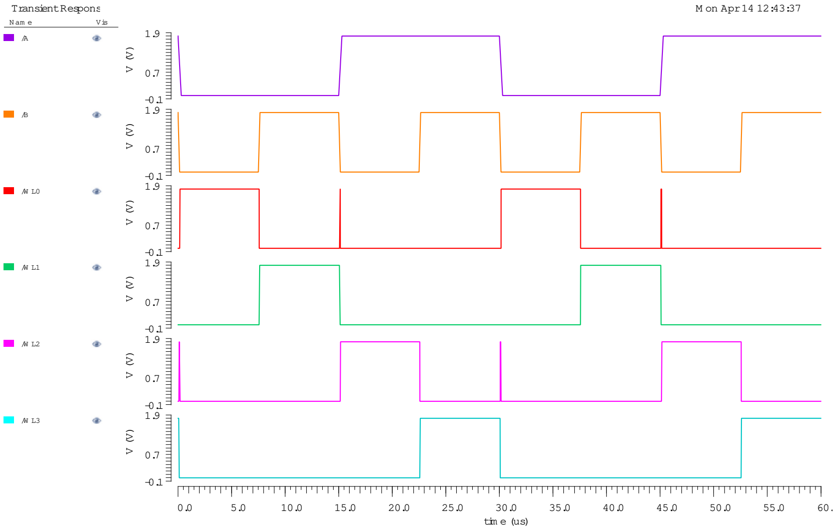


Figure 5.2: 2-to-4 Decoder Circuit transient analysis

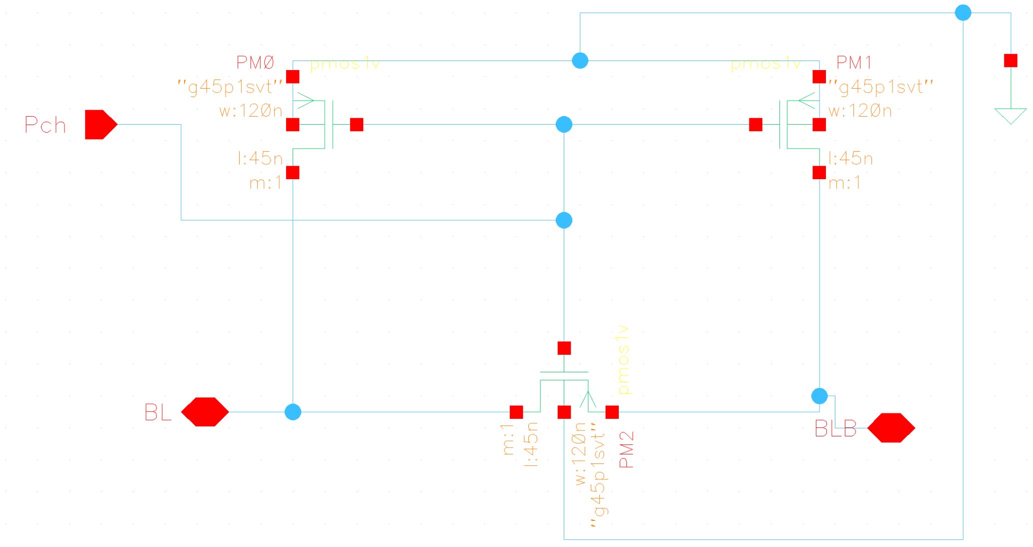


Figure 5.3: Precharge Circuit

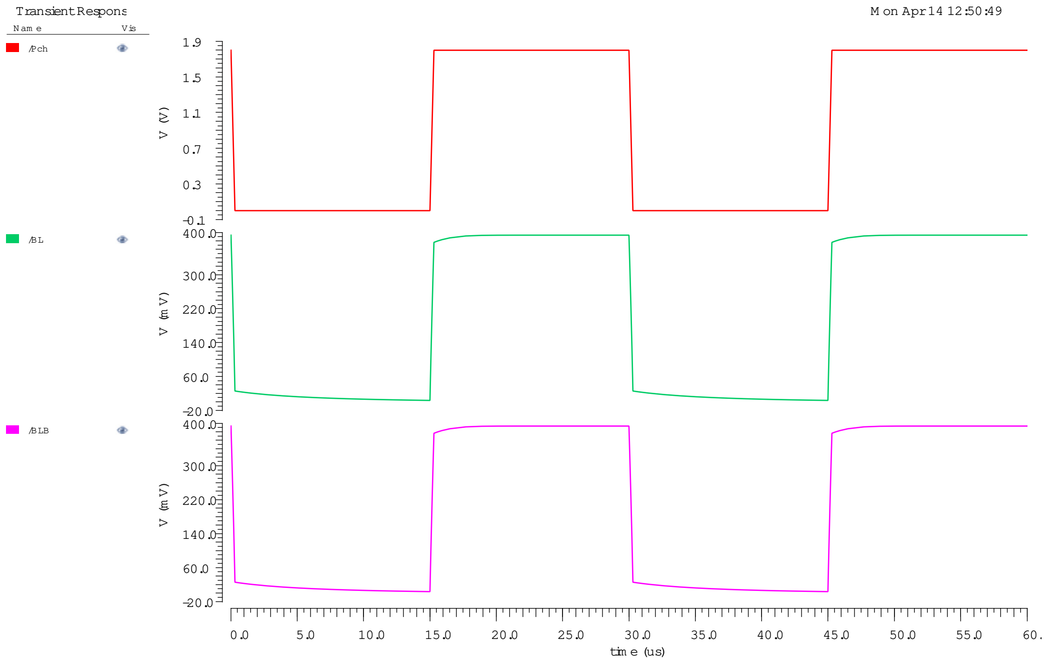


Figure 5.4: Precharge Circuit transient analysis

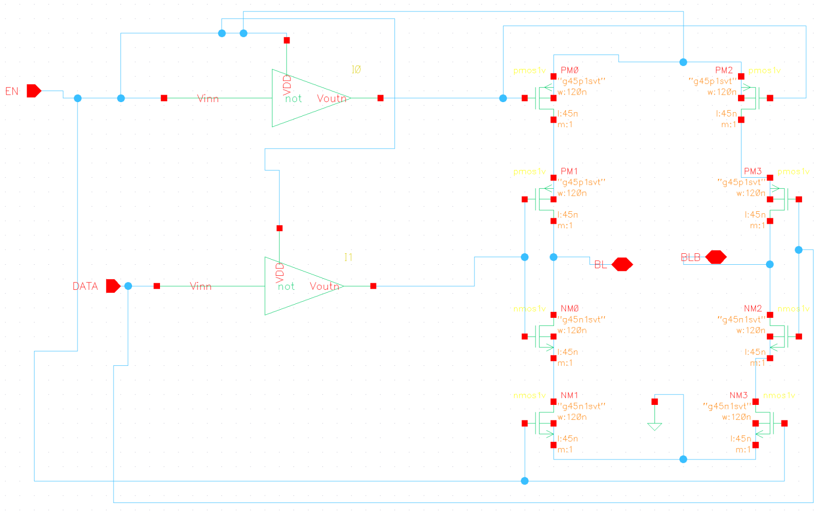


Figure 5.5: Write Driver Circuit

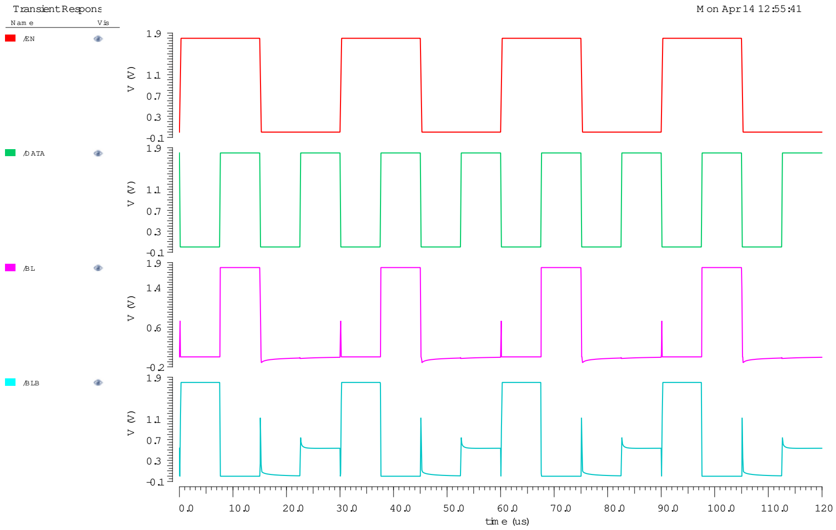


Figure 5.6: Write Driver Circuit transient analysis

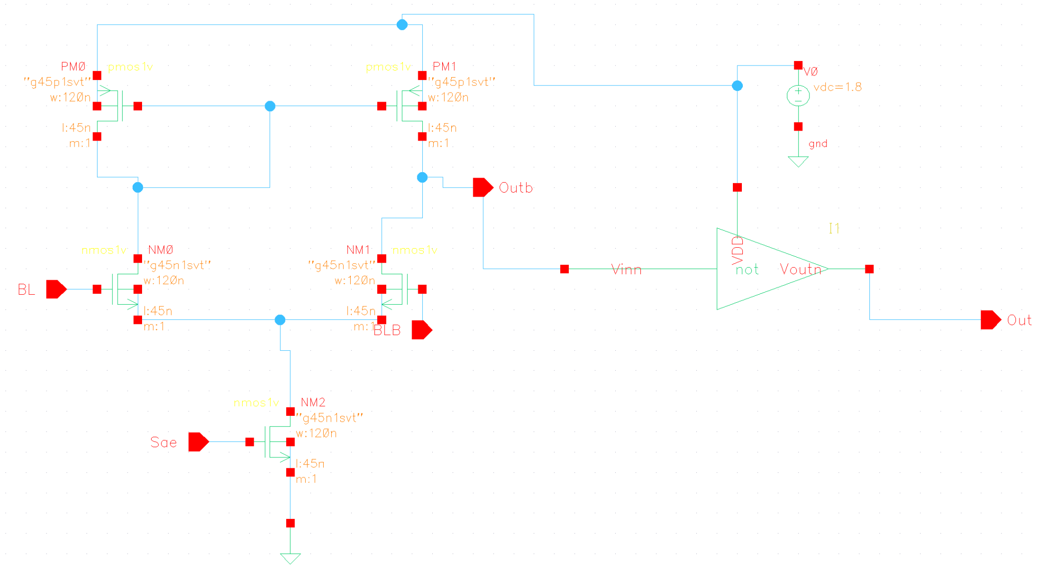


Figure 5.7: Sense Amplifier Circuit

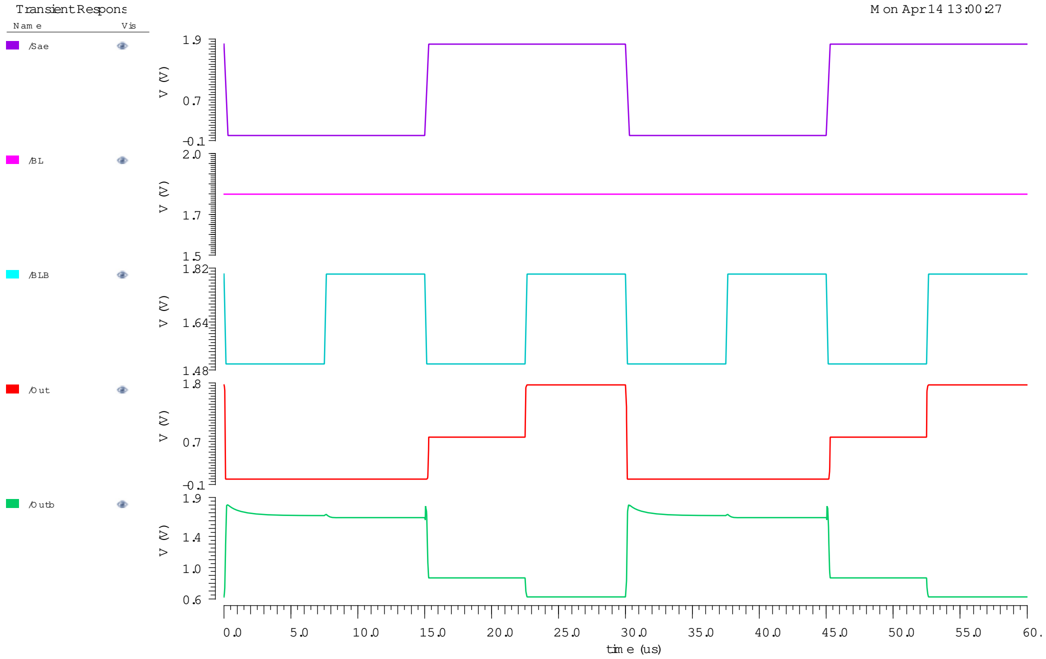


Figure 5.8: Sense Amplifier Circuit transient analysis

5.1.2 Read and Write operations in 6T SRAM cell

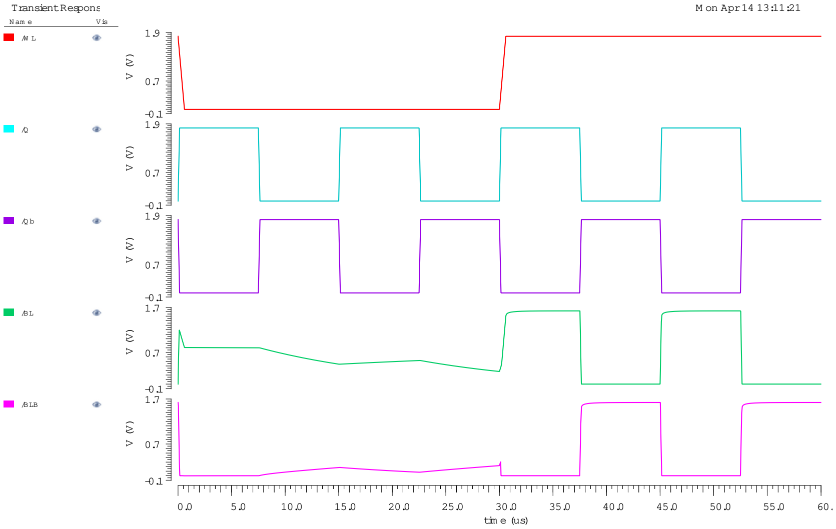


Figure 5.9: Read operation of 6T SRAM cell

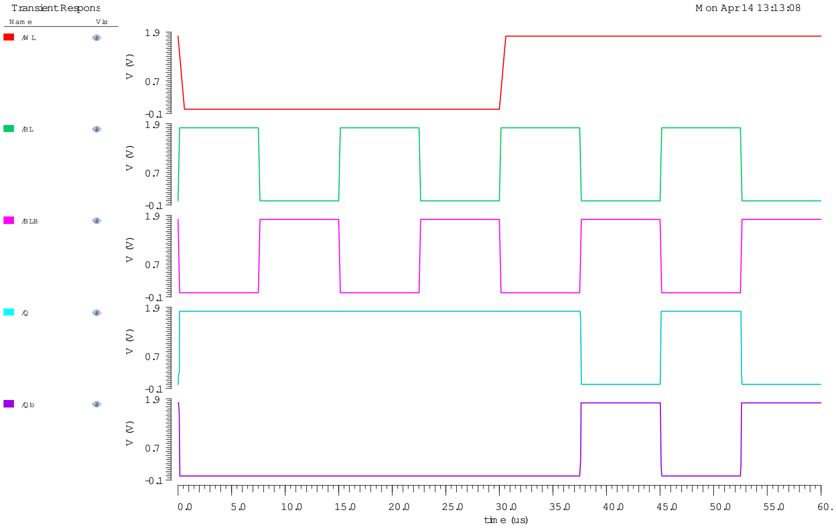


Figure 5.10: Write operation of 6T SRAM cell

5.1.3 Stability Analysis of 6T SRAM cell

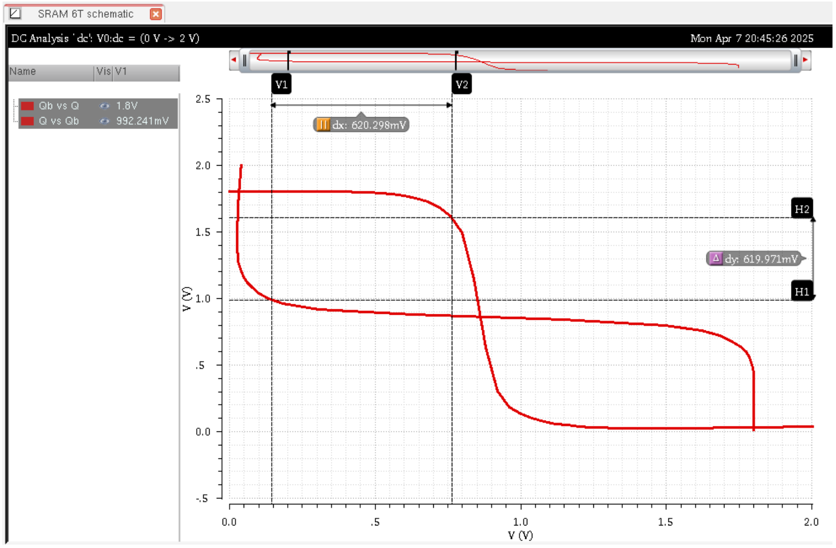


Figure 5.11: Hold SNM of 6T SRAM cell

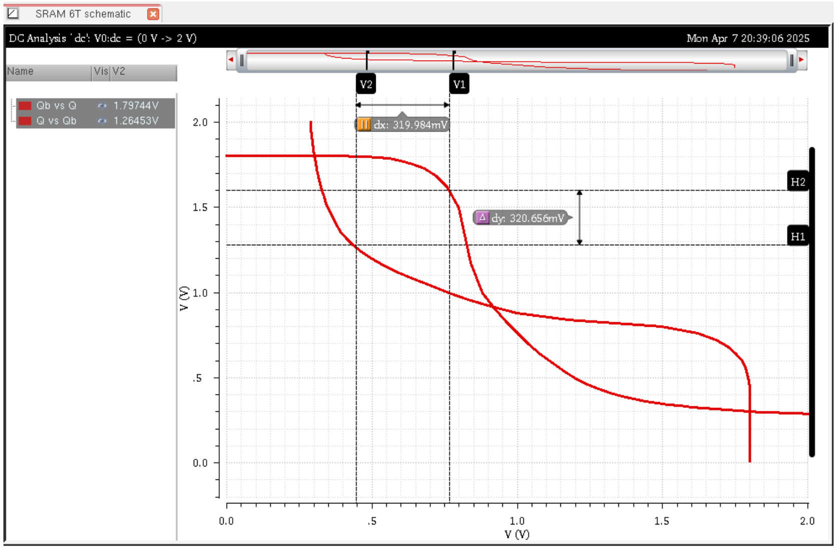


Figure 5.12: Read SNM of 6T SRAM cell

Hold SNM is 620 mV and Read SNM is 320 mV

5.1.4 Read and Write operations of 1 bitcell in 4X4 array

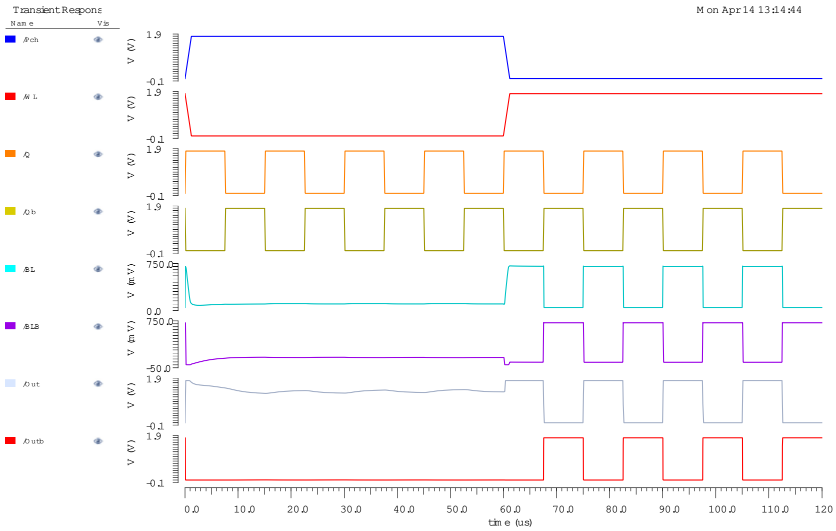


Figure 5.13: Read operation of 1 bitcell in 1st row and column in 4X4 array

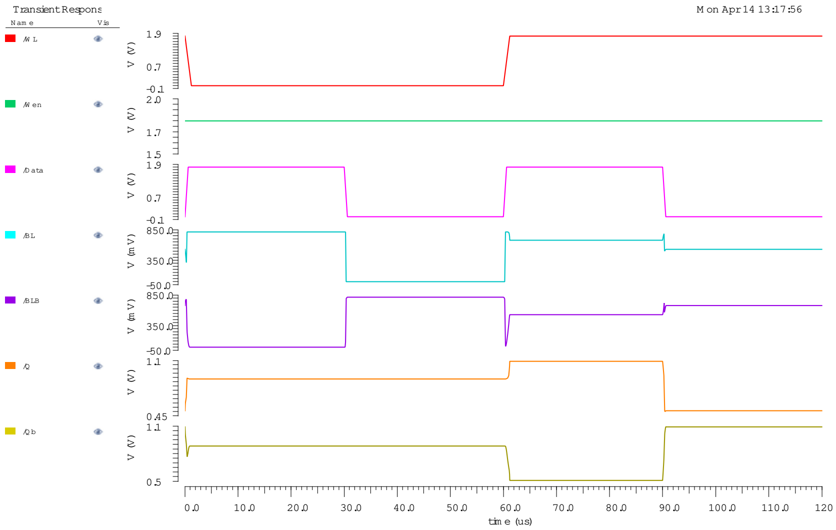


Figure 5.14: Write operation of 1 bitcell in 1st row and column in 4X4 array

5.1.5 Proposed Circuit 1

2 6T SRAM cells

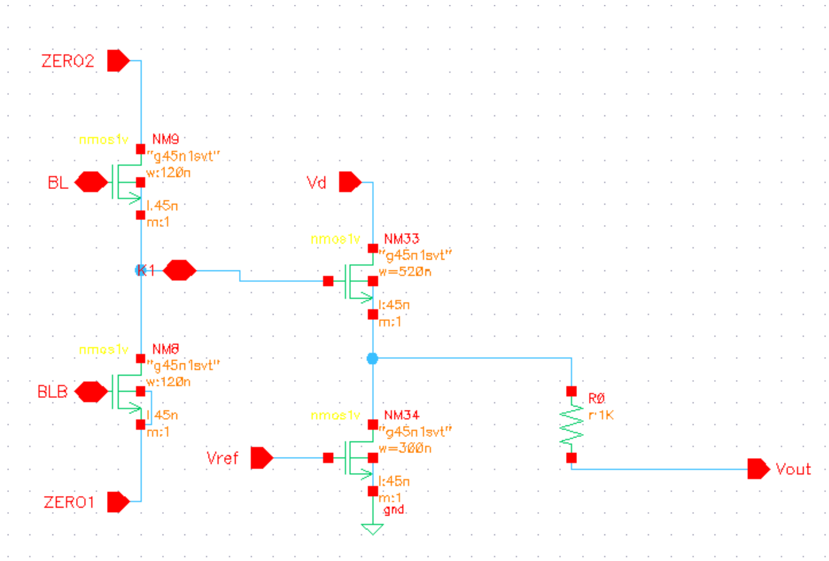


Figure 5.15: Proposed circuit 1

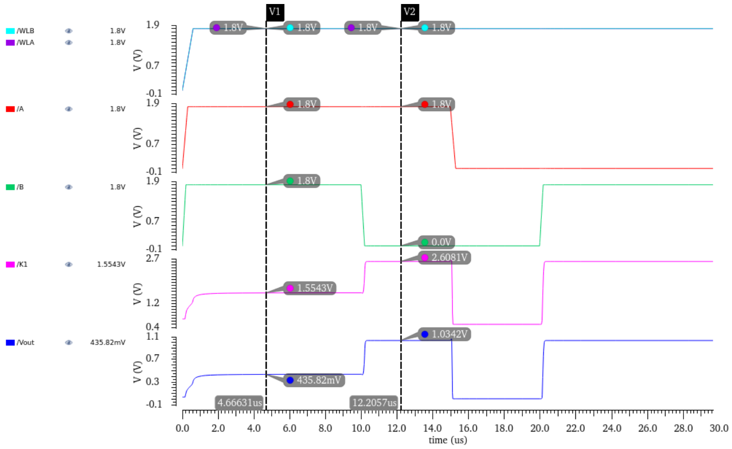


Figure 5.16: XOR operation of proposed circuit 1

A	**B**	**V_{out} (XOR)**
0	0	0 ($<V_{DD}/2 = 0$)
0	V_{DD}	1 ($>V_{DD}/2 = 1$)
V_{DD}	0	1 ($>V_{DD}/2 = 1$)
V_{DD}	V_{DD}	0.4 ($<V_{DD}/2 = 0$)

Table 5.1: Truth table for proposed circuit 1

4X4 Array

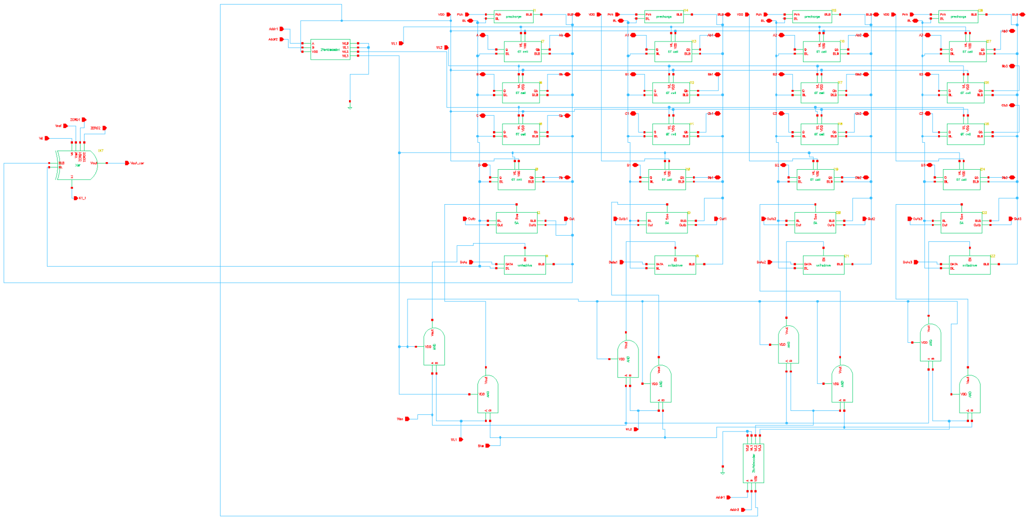


Figure 5.17: Array with modified proposed circuit 1

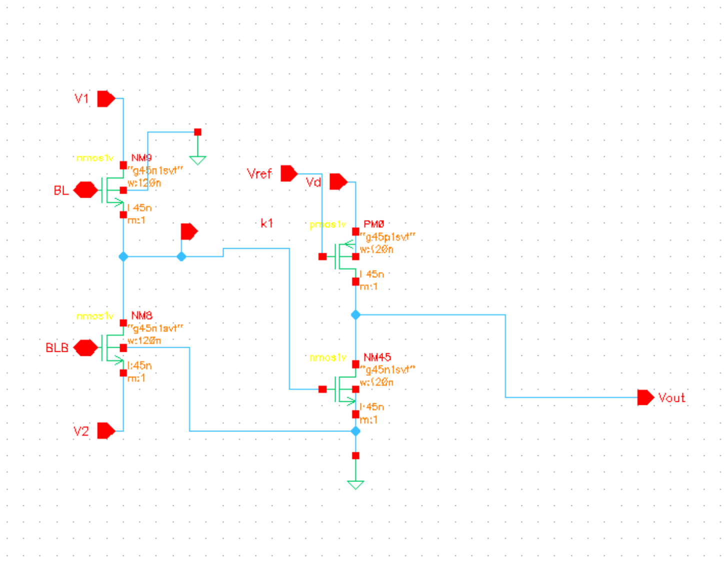


Figure 5.18: Modified proposed circuit 1 for array

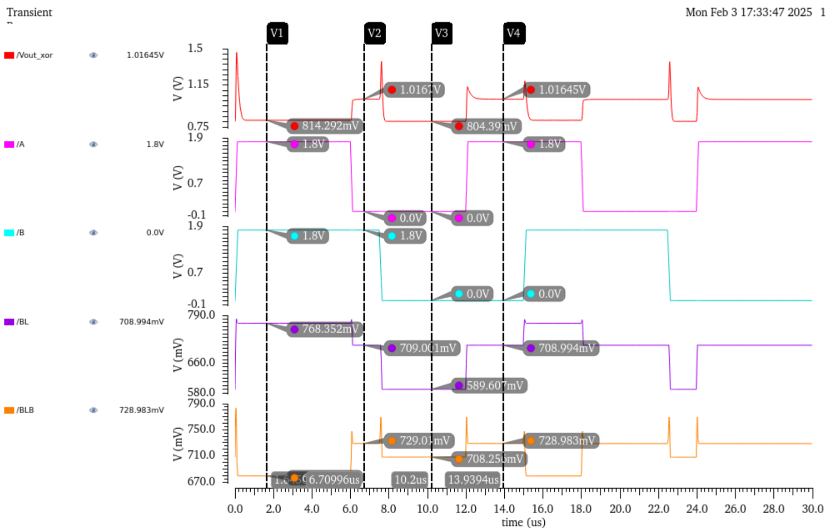


Figure 5.19: XOR operation of array with modified proposed circuit 1

A	**B**	**V_{out} (XOR)**
0	0	0 (0.8 <$V_{DD}/2$)
0	V_{DD}	1 (1.01 >$V_{DD}/2$)
V_{DD}	0	1 (1.01 >$V_{DD}/2$)
V_{DD}	V_{DD}	0 (0.8 <$V_{DD}/2$)

Table 5.2: Truth table of XOR operation for array

5.1.6 Proposed Circuit 2

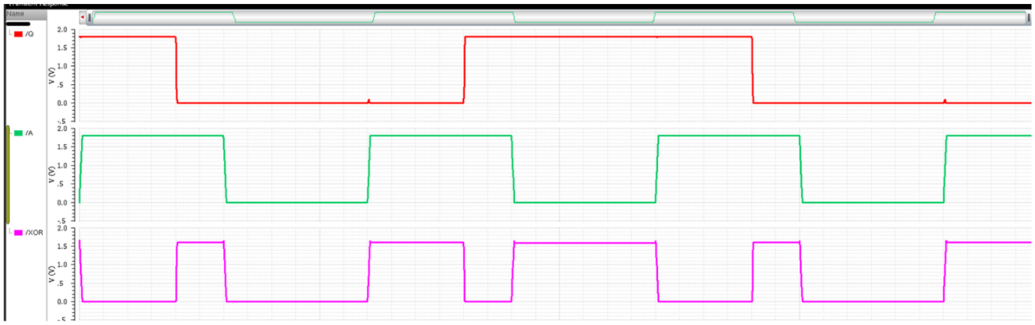


Figure 5.20: XOR operation of proposed circuit 2

Q (Stored Value in SRAM)	**A** (Input Value)	**XOR**
0	0	0
0	1	1
1	0	1
1	1	0

Table 5.3: Truth table for proposed circuit 2

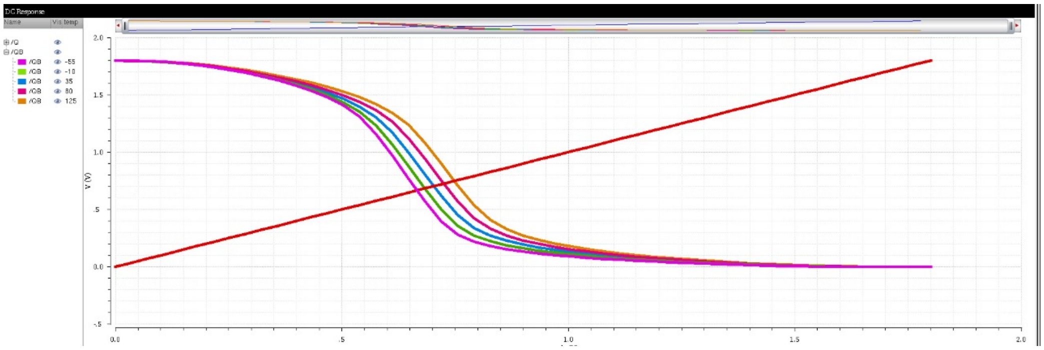


Figure 5.21: DC response analysis of proposed circuit 2 with variation of temperature

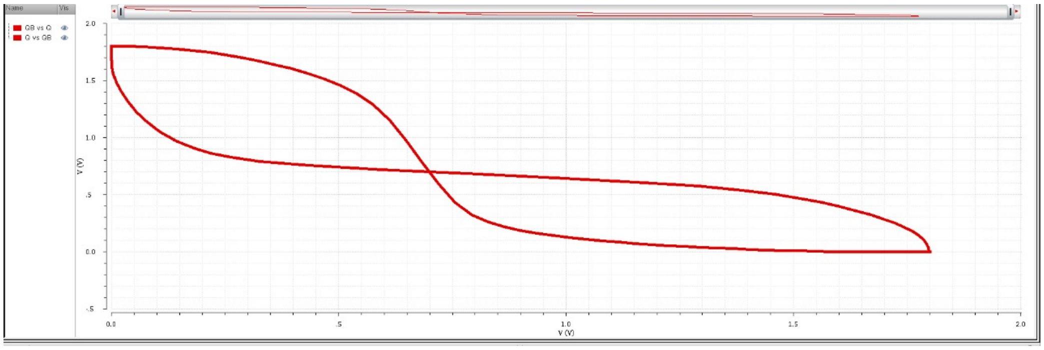


Figure 5.22: Butterfly curve for proposed circuit 2

5.2 Layout

5.2.1 6T SRAM cell

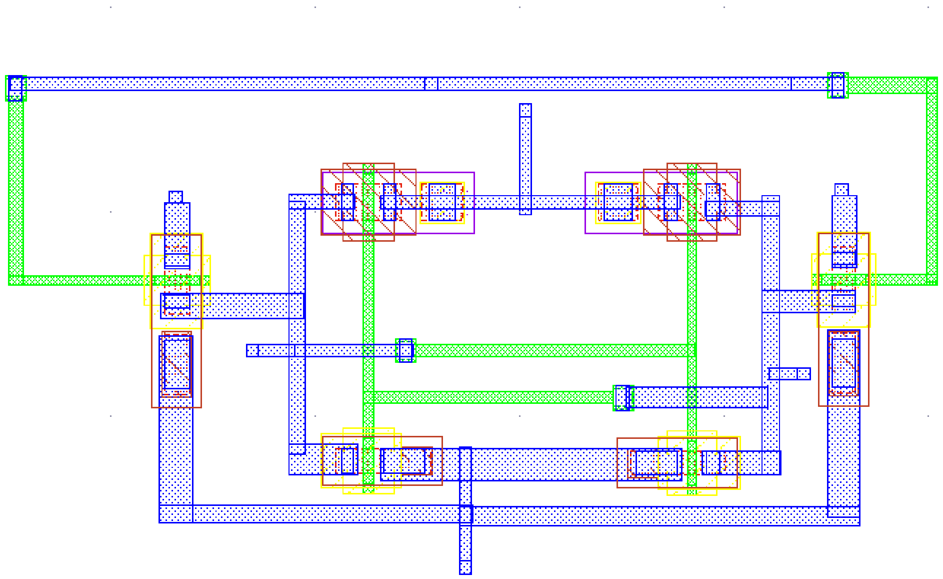
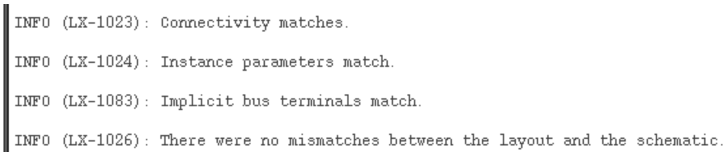


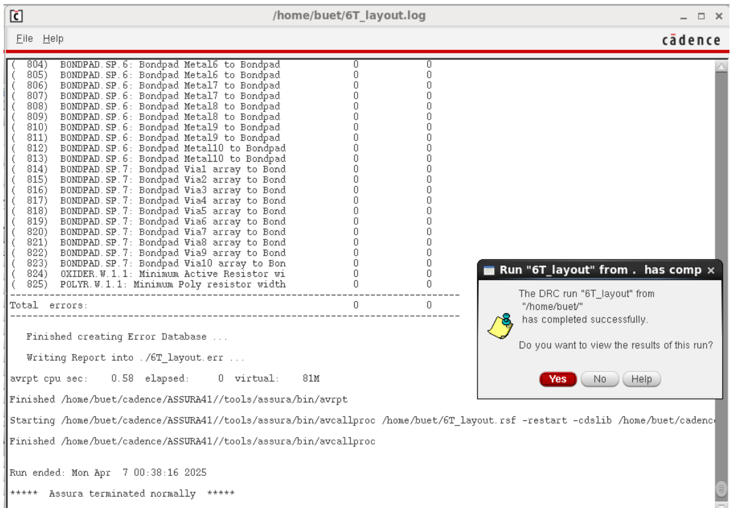
Figure 5.23: Layout of 6T SRAM cell



```
INFO (LX-1023): Connectivity matches.
INFO (LX-1024): Instance parameters match.
INFO (LX-1083): Implicit bus terminals match.
INFO (LX-1026): There were no mismatches between the layout and the schematic.
```

Figure 5.24: Check layout against the source schematic

DRC



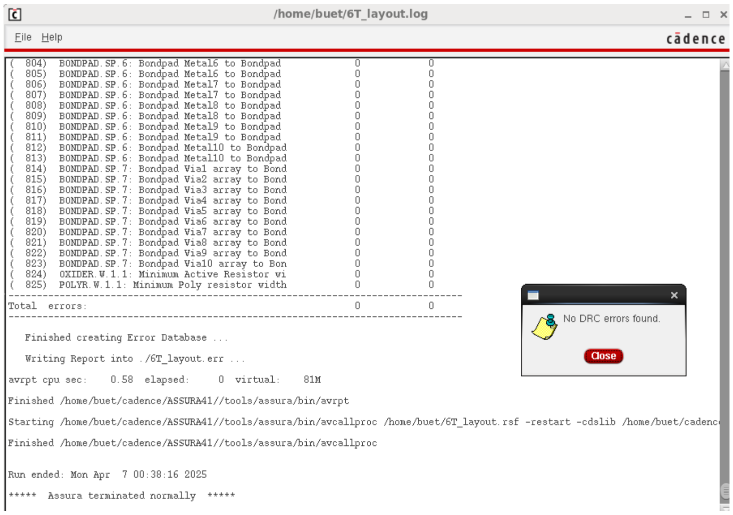
```
(  804)  BONDPAD.SP.6: Bondpad Metal6 to Bondpad           0          0
(  805)  BONDPAD.SP.6: Bondpad Metal6 to Bondpad           0          0
(  806)  BONDPAD.SP.6: Bondpad Metal7 to Bondpad           0          0
(  807)  BONDPAD.SP.6: Bondpad Metal7 to Bondpad           0          0
(  808)  BONDPAD.SP.6: Bondpad Metal8 to Bondpad           0          0
(  809)  BONDPAD.SP.6: Bondpad Metal8 to Bondpad           0          0
(  810)  BONDPAD.SP.6: Bondpad Metal9 to Bondpad           0          0
(  811)  BONDPAD.SP.6: Bondpad Metal9 to Bondpad           0          0
(  812)  BONDPAD.SP.6: Bondpad Metal10 to Bondpad          0          0
(  813)  BONDPAD.SP.6: Bondpad Metal10 to Bondpad          0          0
(  814)  BONDPAD.SP.7: Bondpad Via1 array to Bond          0          0
(  815)  BONDPAD.SP.7: Bondpad Via2 array to Bond          0          0
(  816)  BONDPAD.SP.7: Bondpad Via3 array to Bond          0          0
(  817)  BONDPAD.SP.7: Bondpad Via4 array to Bond          0          0
(  818)  BONDPAD.SP.7: Bondpad Via5 array to Bond          0          0
(  819)  BONDPAD.SP.7: Bondpad Via6 array to Bond          0          0
(  820)  BONDPAD.SP.7: Bondpad Via7 array to Bond          0          0
(  821)  BONDPAD.SP.7: Bondpad Via8 array to Bond          0          0
(  822)  BONDPAD.SP.7: Bondpad Via9 array to Bond          0          0
(  823)  BONDPAD.SP.7: Bondpad Via10 array to Bon          0          0
(  824)  OXIDER.W.1.1: Minimum Active Resistor wi          0          0
(  825)  POLYR.W.1.1: Minimum Poly resistor width          0          0
-----------------------------------------------------------------------
Total  errors:                                             0          0
-----------------------------------------------------------------------

   Finished creating Error Database ...

   Writing Report into ./6T_layout.err ...

avrpt cpu sec:    0.58  elapsed:     0  virtual:    81M

Finished /home/buet/cadence/ASSURA41//tools/assura/bin/avrpt

Starting /home/buet/cadence/ASSURA41//tools/assura/bin/avcallproc /home/buet/6T_layout.rsf -restart -cdslib /home/buet/cadenc

Finished /home/buet/cadence/ASSURA41//tools/assura/bin/avcallproc


Run ended: Mon Apr  7 00:38:16 2025

*****  Assura terminated normally  *****
```

Run "6T_layout" from . has comp

The DRC run "6T_layout" from
"/home/buet/"
has completed successfully.

Do you want to view the results of this run?

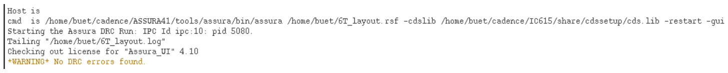
Figure 5.25: DRC of 6T SRAM cell layout I



```
(  804)  BONDPAD.SP.6: Bondpad Metal6 to Bondpad           0          0
(  805)  BONDPAD.SP.6: Bondpad Metal6 to Bondpad           0          0
(  806)  BONDPAD.SP.6: Bondpad Metal7 to Bondpad           0          0
(  807)  BONDPAD.SP.6: Bondpad Metal7 to Bondpad           0          0
(  808)  BONDPAD.SP.6: Bondpad Metal8 to Bondpad           0          0
(  809)  BONDPAD.SP.6: Bondpad Metal8 to Bondpad           0          0
(  810)  BONDPAD.SP.6: Bondpad Metal9 to Bondpad           0          0
(  811)  BONDPAD.SP.6: Bondpad Metal9 to Bondpad           0          0
(  812)  BONDPAD.SP.6: Bondpad Metal10 to Bondpad          0          0
(  813)  BONDPAD.SP.6: Bondpad Metal10 to Bondpad          0          0
(  814)  BONDPAD.SP.7: Bondpad Via1 array to Bond          0          0
(  815)  BONDPAD.SP.7: Bondpad Via2 array to Bond          0          0
(  816)  BONDPAD.SP.7: Bondpad Via3 array to Bond          0          0
(  817)  BONDPAD.SP.7: Bondpad Via4 array to Bond          0          0
(  818)  BONDPAD.SP.7: Bondpad Via5 array to Bond          0          0
(  819)  BONDPAD.SP.7: Bondpad Via6 array to Bond          0          0
(  820)  BONDPAD.SP.7: Bondpad Via7 array to Bond          0          0
(  821)  BONDPAD.SP.7: Bondpad Via8 array to Bond          0          0
(  822)  BONDPAD.SP.7: Bondpad Via9 array to Bond          0          0
(  823)  BONDPAD.SP.7: Bondpad Via10 array to Bon          0          0
(  824)  OXIDER.W.1.1: Minimum Active Resistor wi          0          0
(  825)  POLYR.W.1.1: Minimum Poly resistor width          0          0
-----------------------------------------------------------------------
Total  errors:                                             0          0
-----------------------------------------------------------------------

   Finished creating Error Database ...

   Writing Report into ./6T_layout.err ...

avrpt cpu sec:    0.58  elapsed:     0  virtual:    81M

Finished /home/buet/cadence/ASSURA41//tools/assura/bin/avrpt

Starting /home/buet/cadence/ASSURA41//tools/assura/bin/avcallproc /home/buet/6T_layout.rsf -restart -cdslib /home/buet/cadenc

Finished /home/buet/cadence/ASSURA41//tools/assura/bin/avcallproc


Run ended: Mon Apr  7 00:38:16 2025

*****  Assura terminated normally  *****
```

No DRC errors found.

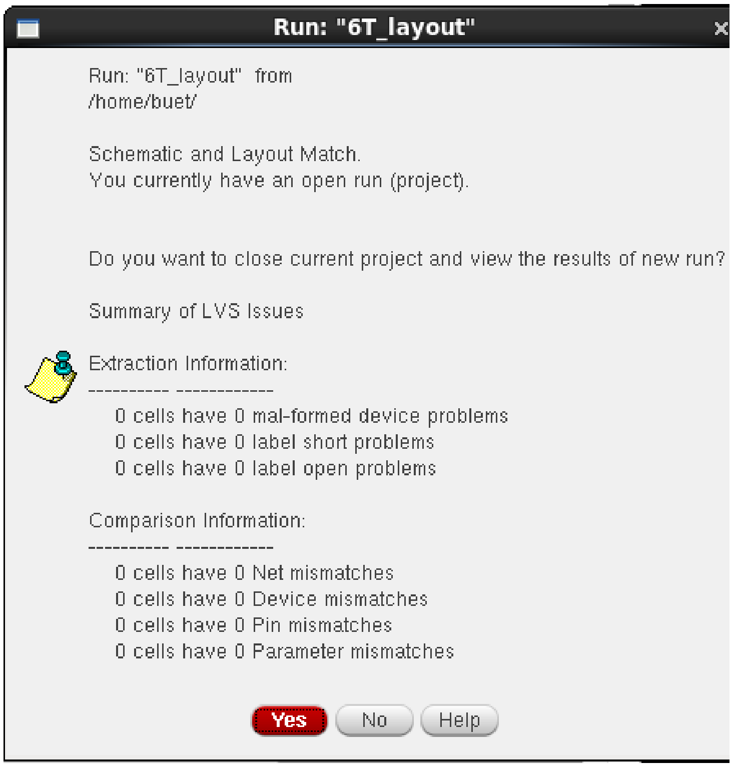
Figure 5.26: DRC of 6T SRAM cell layout II



```
Host is
cmd  is /home/buet/cadence/ASSURA41/tools/assura/bin/assura /home/buet/6T_layout.rsf -cdslib /home/buet/cadence/IC615/share/cdssetup/cds.lib -restart -gui
Starting the Assura DRC Run: IPC Id ipc:10: pid 5080.
Tailing "/home/buet/6T_layout.log"
Checking out license for "Assura_UI" 4.10
*WARNING* No DRC errors found.
```

Figure 5.27: DRC of 6T SRAM cell layout III

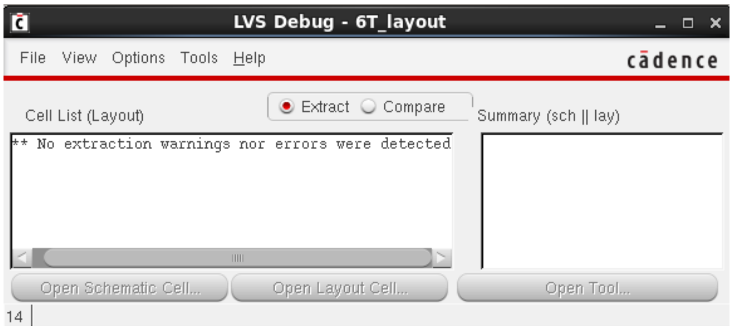
LVS



Run: "6T_layout"

```
Run: "6T_layout"  from
/home/buet/

Schematic and Layout Match.
You currently have an open run (project).


Do you want to close current project and view the results of new run?

Summary of LVS Issues

Extraction Information:
---------- ------------
    0 cells have 0 mal-formed device problems
    0 cells have 0 label short problems
    0 cells have 0 label open problems

Comparison Information:
---------- ------------
    0 cells have 0 Net mismatches
    0 cells have 0 Device mismatches
    0 cells have 0 Pin mismatches
    0 cells have 0 Parameter mismatches
```

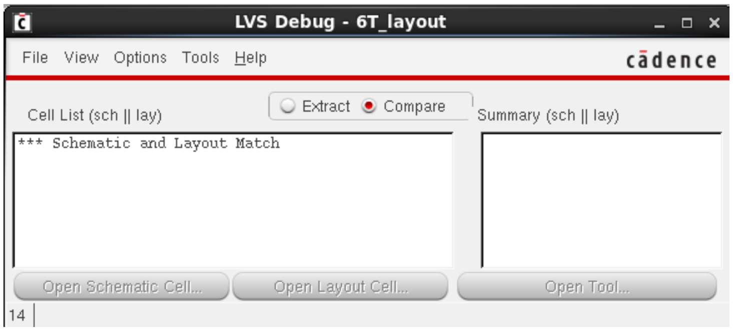
Figure 5.28: LVS of 6T SRAM cell layout I



● Extract ○ Compare

```
** No extraction warnings nor errors were detected
```

Figure 5.29: LVS of 6T SRAM cell layout II



○ Extract ● Compare

```
*** Schematic and Layout Match
```

Figure 5.30: LVS of 6T SRAM cell layout III

5.2.2 XOR for 2 6T cells

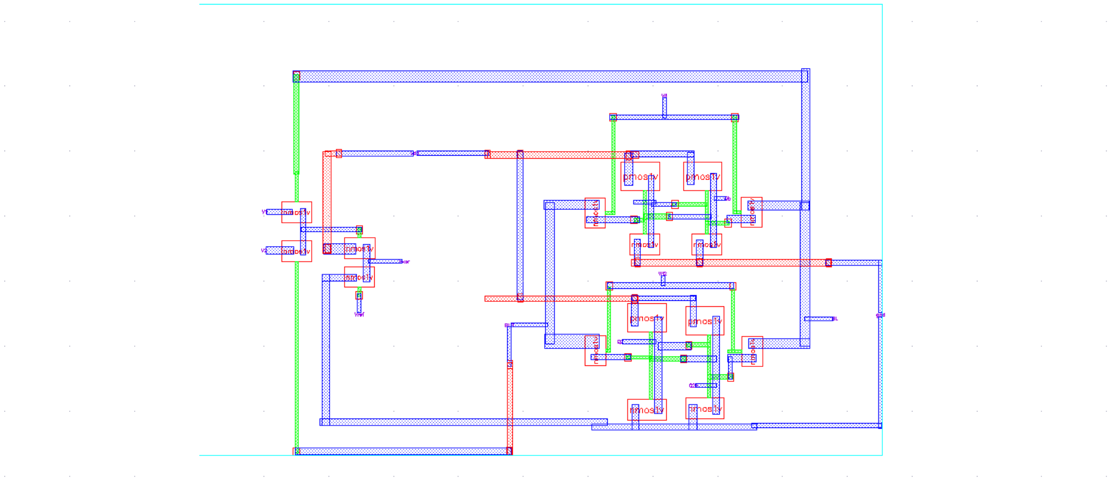
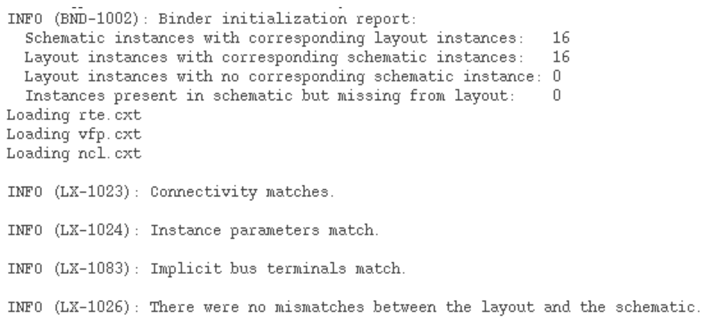


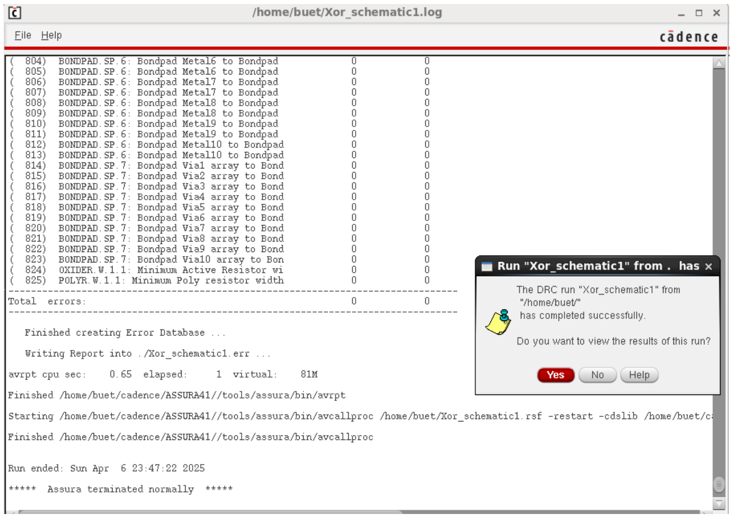
Figure 5.31: Layout of XOR circuit for 2 6T cells



```
INFO (BND-1002): Binder initialization report:
  Schematic instances with corresponding layout instances:   16
  Layout instances with corresponding schematic instances:   16
  Layout instances with no corresponding schematic instance: 0
  Instances present in schematic but missing from layout:    0
Loading rte.cxt
Loading vfp.cxt
Loading ncl.cxt

INFO (LX-1023): Connectivity matches.

INFO (LX-1024): Instance parameters match.

INFO (LX-1083): Implicit bus terminals match.

INFO (LX-1026): There were no mismatches between the layout and the schematic.
```

Figure 5.32: Check layout against the source schematic

DRC



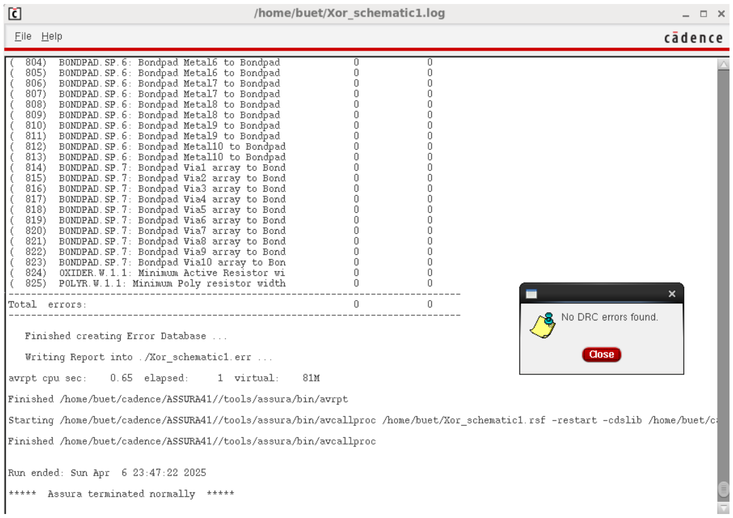
```
(  804)  BONDPAD.SP.6: Bondpad Metal6 to Bondpad          0          0
(  805)  BONDPAD.SP.6: Bondpad Metal6 to Bondpad          0          0
(  806)  BONDPAD.SP.6: Bondpad Metal7 to Bondpad          0          0
(  807)  BONDPAD.SP.6: Bondpad Metal7 to Bondpad          0          0
(  808)  BONDPAD.SP.6: Bondpad Metal8 to Bondpad          0          0
(  809)  BONDPAD.SP.6: Bondpad Metal8 to Bondpad          0          0
(  810)  BONDPAD.SP.6: Bondpad Metal9 to Bondpad          0          0
(  811)  BONDPAD.SP.6: Bondpad Metal9 to Bondpad          0          0
(  812)  BONDPAD.SP.6: Bondpad Metal10 to Bondpad         0          0
(  813)  BONDPAD.SP.6: Bondpad Metal10 to Bondpad         0          0
(  814)  BONDPAD.SP.7: Bondpad Via1 array to Bond         0          0
(  815)  BONDPAD.SP.7: Bondpad Via2 array to Bond         0          0
(  816)  BONDPAD.SP.7: Bondpad Via3 array to Bond         0          0
(  817)  BONDPAD.SP.7: Bondpad Via4 array to Bond         0          0
(  818)  BONDPAD.SP.7: Bondpad Via5 array to Bond         0          0
(  819)  BONDPAD.SP.7: Bondpad Via6 array to Bond         0          0
(  820)  BONDPAD.SP.7: Bondpad Via7 array to Bond         0          0
(  821)  BONDPAD.SP.7: Bondpad Via8 array to Bond         0          0
(  822)  BONDPAD.SP.7: Bondpad Via9 array to Bond         0          0
(  823)  BONDPAD.SP.7: Bondpad Via10 array to Bon         0          0
(  824)  OXIDER.W.1.1: Minimum Active Resistor wi         0          0
(  825)  POLYR.W.1.1: Minimum Poly resistor width         0          0
-----------------------------------------------------------------------------
Total  errors:                                            0          0
-----------------------------------------------------------------------------

   Finished creating Error Database ...

   Writing Report into ./Xor_schematic1.err ...

avrpt cpu sec:    0.65  elapsed:     1  virtual:    81M

Finished /home/buet/cadence/ASSURA41//tools/assura/bin/avrpt

Starting /home/buet/cadence/ASSURA41//tools/assura/bin/avcallproc /home/buet/Xor_schematic1.rsf -restart -cdslib /home/buet/c

Finished /home/buet/cadence/ASSURA41//tools/assura/bin/avcallproc


Run ended: Sun Apr  6 23:47:22 2025

*****  Assura terminated normally  *****
```

Run "Xor_schematic1" from . has

The DRC run "Xor_schematic1" from "/home/buet/" has completed successfully.

Do you want to view the results of this run?

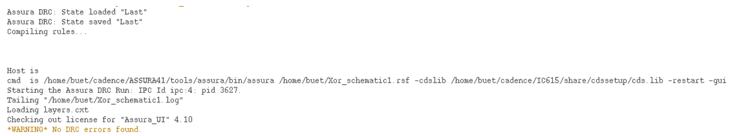
Figure 5.33: DRC of XOR circuit for 2 6T cells layout I



```
(  804)  BONDPAD.SP.6: Bondpad Metal6 to Bondpad          0          0
(  805)  BONDPAD.SP.6: Bondpad Metal6 to Bondpad          0          0
(  806)  BONDPAD.SP.6: Bondpad Metal7 to Bondpad          0          0
(  807)  BONDPAD.SP.6: Bondpad Metal7 to Bondpad          0          0
(  808)  BONDPAD.SP.6: Bondpad Metal8 to Bondpad          0          0
(  809)  BONDPAD.SP.6: Bondpad Metal8 to Bondpad          0          0
(  810)  BONDPAD.SP.6: Bondpad Metal9 to Bondpad          0          0
(  811)  BONDPAD.SP.6: Bondpad Metal9 to Bondpad          0          0
(  812)  BONDPAD.SP.6: Bondpad Metal10 to Bondpad         0          0
(  813)  BONDPAD.SP.6: Bondpad Metal10 to Bondpad         0          0
(  814)  BONDPAD.SP.7: Bondpad Via1 array to Bond         0          0
(  815)  BONDPAD.SP.7: Bondpad Via2 array to Bond         0          0
(  816)  BONDPAD.SP.7: Bondpad Via3 array to Bond         0          0
(  817)  BONDPAD.SP.7: Bondpad Via4 array to Bond         0          0
(  818)  BONDPAD.SP.7: Bondpad Via5 array to Bond         0          0
(  819)  BONDPAD.SP.7: Bondpad Via6 array to Bond         0          0
(  820)  BONDPAD.SP.7: Bondpad Via7 array to Bond         0          0
(  821)  BONDPAD.SP.7: Bondpad Via8 array to Bond         0          0
(  822)  BONDPAD.SP.7: Bondpad Via9 array to Bond         0          0
(  823)  BONDPAD.SP.7: Bondpad Via10 array to Bon         0          0
(  824)  OXIDER.W.1.1: Minimum Active Resistor wi         0          0
(  825)  POLYR.W.1.1: Minimum Poly resistor width         0          0
-----------------------------------------------------------------------------
Total  errors:                                            0          0
-----------------------------------------------------------------------------

   Finished creating Error Database ...

   Writing Report into ./Xor_schematic1.err ...

avrpt cpu sec:    0.65  elapsed:     1  virtual:    81M

Finished /home/buet/cadence/ASSURA41//tools/assura/bin/avrpt

Starting /home/buet/cadence/ASSURA41//tools/assura/bin/avcallproc /home/buet/Xor_schematic1.rsf -restart -cdslib /home/buet/c

Finished /home/buet/cadence/ASSURA41//tools/assura/bin/avcallproc


Run ended: Sun Apr  6 23:47:22 2025

*****  Assura terminated normally  *****
```

No DRC errors found.

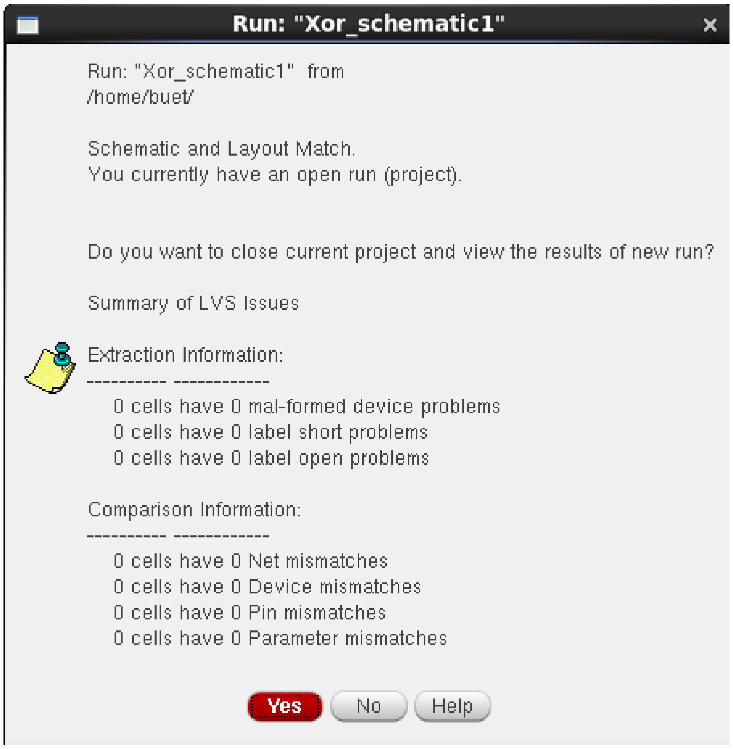
Figure 5.34: DRC of XOR circuit for 2 6T cells layout II



```
Assura DRC: State loaded "Last"
Assura DRC: State saved "Last"
Compiling rules...


Host is
cmd  is /home/buet/cadence/ASSURA41/tools/assura/bin/assura /home/buet/Xor_schematic1.rsf -cdslib /home/buet/cadence/IC615/share/cdssetup/cds.lib -restart -gui
Starting the Assura DRC Run: IPC Id ipc:4: pid 3627.
Tailing "/home/buet/Xor_schematic1.log"
Loading layers.cxt
Checking out license for "Assura_UI" 4.10
*WARNING* No DRC errors found.
```

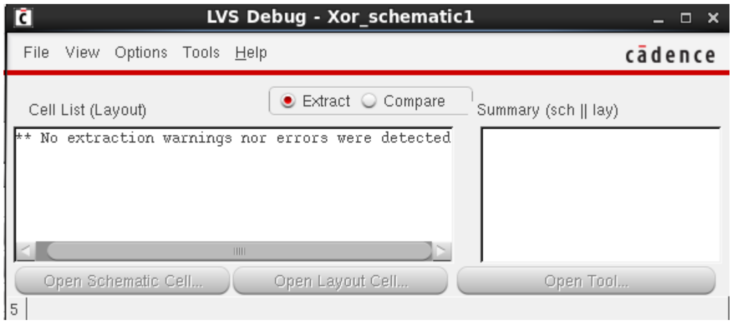
Figure 5.35: DRC of XOR circuit for 2 6T cells layout II

LVS



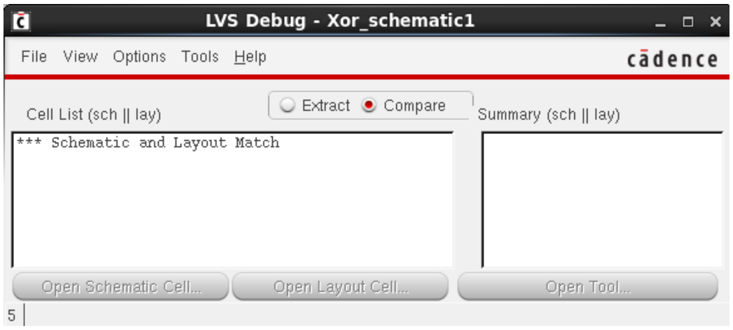
```
Run: "Xor_schematic1"  from
/home/buet/

Schematic and Layout Match.
You currently have an open run (project).


Do you want to close current project and view the results of new run?

Summary of LVS Issues

Extraction Information:
---------- ------------
    0 cells have 0 mal-formed device problems
    0 cells have 0 label short problems
    0 cells have 0 label open problems

Comparison Information:
---------- ------------
    0 cells have 0 Net mismatches
    0 cells have 0 Device mismatches
    0 cells have 0 Pin mismatches
    0 cells have 0 Parameter mismatches
```

Figure 5.36: LVS of XOR circuit for 2 6T cells layout I



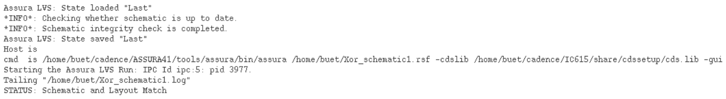
```
** No extraction warnings nor errors were detected
```

Figure 5.37: LVS of XOR circuit for 2 6T cells layout II



```
*** Schematic and Layout Match
```

Figure 5.38: LVS of XOR circuit for 2 6T cells layout III



```
Assura LVS: State loaded "Last"
*INFO*: Checking whether schematic is up to date.
*INFO*: Schematic integrity check is completed.
Assura LVS: State saved "Last"
Host is
cmd  is /home/buet/cadence/ASSURA41/tools/assura/bin/assura /home/buet/Xor_schematic1.rsf -cdslib /home/buet/cadence/IC615/share/cdssetup/cds.lib -gui
Starting the Assura LVS Run: IPC Id ipc:5: pid 3977.
Tailing "/home/buet/Xor_schematic1.log"
STATUS: Schematic and Layout Match
```

Figure 5.39: LVS of XOR circuit for 2 6T cells layout IV

Chapter 6

Conclusion

6.1 Comparison

Proposed Circuit 1 (6T XOR)	**Proposed Circuit 2 (8T XOR)**
Less area consumption on the chip	More area consumption on the chip
Less SNM (Static Noise Margin), less stability	More SNM, more stability
Useful for applications which require XOR operation between data stored in two bitcells in memory	Useful for applications which require XOR operation between data stored in a bitcell and external data
Applications: Hamming code, Parity Check	Applications: Cryptography, Content Addressable Memory (CAM)

Table 6.1: Comparison between 6T XOR and 8T XOR SRAM cells

6.2 Conclusion

In this project, a 6T SRAM bitcell was successfully implemented and simulated for both read and write operations. To enable full SRAM functionality, key peripheral circuits were designed and implemented, including row and column decoders, sense amplifiers, precharge circuits, and write drivers. A 4x4 SRAM array was constructed and simulated, demonstrating successful read and write operations for 1-bit data.

To enhance in-memory computing capabilities, two novel circuits were proposed to realize XOR logic using conventional 6T SRAM cells. One of the proposed circuits was further modified to compute the XOR of two bitcells within the 4x4 array, enabling data-level logic operations directly within the memory array.

Stability analysis was carried out for the hold and read states of the 6T SRAM cell, as XOR operations were only performed during the read phase. Post-layout verification was conducted for the 6T SRAM cell to ensure accuracy after physical design implementation. Additionally, the layout of the XOR circuit using two 6T SRAM cells was constructed and verified through post-layout simulations, confirming the correctness and feasibility of integrating logic-in-memory within standard SRAM architectures.

These designs pave the way for efficient in-memory computing, reducing data movement and enhancing energy efficiency. This work serves as a foundational step toward more advanced and integrated memory-based logic operations in SRAM architectures.

Bibliography

[1] Amogh Agrawal, Akhilesh Jaiswal, Chankyu Lee, and Kaushik Roy. X-sram: Enabling in-memory boolean computations in cmos static random access memories. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 65(12):4219–4232, 2018.

[2] Nader Alnatsheh, Youngbae Kim, Jaeik Cho, and Kyuwon Ken Choi. A novel 8t xnor-sram: Computing-in-memory design for binary/ternary deep neural networks. *Electronics*, 12(4), 2023.

[3] Zhuojun Chen, Ming Wu, Yifeng Zhou, Renlong Li, Jinzhe Tan, and Ding Ding. Puf-cim: Sram-based compute-in-memory with zero bit-error-rate physical unclonable function for lightweight secure edge computing. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 31(8):1234–1247, 2023.

[4] Anil Kumar Rajput and Manisha Pattanaik. Energy efficient 9t sram with r/w margin enhanced for beyond von-neumann computation. In *2020 24th International Symposium on VLSI Design and Test (VDAT)*, pages 1–4. IEEE, 2020.

[5] Anil Kumar Rajput and Manisha Pattanaik. Implementation of boolean and arithmetic functions with 8t sram cell for in-memory computation. In *2020 International Conference for Emerging Technology (INCET)*, pages 1–5. IEEE, 2020.

[6] Amitesh Sridharan, Shaahin Angizi, Sai Kiran Cherupally, Fan Zhang, Jae-Sun Seo, and Deliang Fan. A 1.23-ghz 16-kb programmable and generic processing-in-sram accelerator in 65nm. In *ESSCIRC 2022-IEEE 48th European Solid State Circuits Conference (ESSCIRC)*, pages 153–156. IEEE, 2022.

[7] Priyanka Tiwari and Prasanna Kumar Misra. Fast and energy efficient (0.01-2.78 aj) logic in memory module using sram cells. In *2024 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, pages 1–6. IEEE, 2024.

[8] Shihui Yin, Zhewei Jiang, Jae-Sun Seo, and Mingoo Seok. Xnor-sram: In-memory computing sram macro for binary/ternary deep neural networks. *IEEE Journal of Solid-State Circuits*, 55(6):1733–1743, 2020.