Impact of SRAM Architectures - Implementing ChaCha20 Encryption Algorithm using In-Memory Compute Design Choices on Sustainability

by

Samridhi Jain

Under the Supervision of Dr. Anuj Grover

Electronics and Communication Engineering

Indraprastha Institute of Information Technology Delhi

New Delhi - 110020

14 May 2025

© Indraprastha Institute of Information Technology Delhi, New Delhi - 110020
May 2025

Impact of SRAM Architectures - Implementing ChaCha20 Encryption Algorithm using In-Memory Compute Design Choices on Sustainability

A Thesis Report

submitted by

Samridhi Jain

in partial fulfilment of the requirements
for the award of the degree of

Master of Technology

to

Electronics and Communication Engineering
Indraprastha Institute of Information Technology Delhi
New Delhi - 110020

14 May 2025

Certificate

This is to certify that the thesis titled "**Impact of SRAM Architectures - Implementing ChaCha20 Encryption Algorithm using In Memory Compute Design Choices on Sustainability**" being submitted by **Samridhi Jain**, to the Indraprastha Institute of Information Technology Delhi, for the award of the degree of **Master of Technology**, is an original research work carried out by her under my supervision. In my opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Anuj Grover
Associate Professor
Department of Electronics and Communication
Indraprastha Institute of Information Technology Delhi
New Delhi 110020

Date: 14 May 2025

Acknowledgements

I would like to express my heartfelt gratitude to my thesis advisor, Dr. Anuj Grover, for his constant support, guidance, and encouragement throughout the course of this work. I am also thankful to my fellow batchmates for their help and cooperation during various stages of the project. I sincerely thank Aamir Sir for always being available to clear my doubts and for his valuable insights. Lastly, I am deeply grateful to my parents and family for their continuous support and motivation.

Samridhi Jain
(MT23219)

Abstract

As our dependency on technology increases, the use of electronic devices also rises, leading to a greater environmental impact that must be monitored. In today's world, where data security is critical, especially in IoT devices, it is essential to implement encryption algorithms that ensure security [1] and support sustainability. This work introduces two key metrics to evaluate sustainability: embodied carbon footprint and operational carbon footprint. Together, these metrics offer a comprehensive way to assess the environmental impact of design decisions. This study analyzes three different SRAM architectures (10T [2], 9T, and 6T) implementing the ChaCha20 encryption algorithm. The implementation based on 10T bitcell [2] is extended for low power and lesser area implementations.This work proposes two new implementations based on 9T and 6T bitcells which is influenced from the 10T implementation. The evaluation includes traditional metrics such as power, performance, area, and newly introduced sustainability metrics. The security of each design against side-channel attacks is assessed using Welch's t-test. Among the three implementations, the 6T bitcell design shows the highest operational footprint due to its longer execution time and the lowest embodied footprint due to its smaller area. While analyzing embodied and operational footprints separately is important, combining both is essential for making informed and sustainable design choices. The design choice should be sustainable and maintain strong protection against side-channel threats.

Keywords: Sustainability, Emobodied Footprint, Operational Footprint, In-Memory Compute, SRAM, Encryption, ChaCha20, Security, Welch's T-test

Contents

Certificate **i**

Acknowledgements **ii**

Abstract **iii**

List of Tables **v**

List of Figures **vi**

Abbreviations **viii**

Notation **ix**

1 Introduction **1**

1.1 Motivation 1

1.2 Sustainability in Semiconductor Industry 1

1.3 In Memory Compute 2

2 Background and Related Work **3**

2.1 ChaCha20 and it's IMC implementation 3

2.1.1 ChaCha20 3

2.1.2 Implementing ChaCha20 using IMC 4

2.2 Welch's T-Test 5

3 Implementation using different SRAM Architectures **7**

3.0.1 Implementation using 10T bitcell 7

3.0.1.1 10T Bitcell Working 7

3.0.1.2 IMC Implementation using 10T 8

3.0.1.3 Layout explanation 9

3.0.1.4 Welch's t-test result 10

3.0.2 Implementation using 9T bitcell 10

3.0.2.1 9T Bitcell 10
3.0.2.2 IMC explanation in bitcell 11
3.0.2.3 Layout explanation 11
3.0.2.4 Welch's t-test result 12
3.0.3 Implementation using 6T bitcell 13
3.0.3.1 Bitcell working 13
3.0.3.2 IMC implementation 13
3.0.3.3 Layout explanation 15
3.0.3.4 Welch's test result 15
3.0.4 Power, Performance, Area and Security Comparison 16

4 Sustainability Metric **17**
4.0.1 Sustainability Analysis 17
4.0.2 Embodied Footprint 17
4.0.2.1 Base Fabrication Energy 17
4.0.2.2 Extra Processing Step Impact 18
4.0.2.3 Porosity Factor 19
4.0.3 Operational Carbon Footprint 20
4.0.3.1 Dynamic Power (DPQ) 20
4.0.3.2 Static Power (SPQ) 21
4.0.3.3 Leakage Power Quantity (LPQ) 21

5 Results **24**
5.0.1 Impact of Introducing Porosity in Design on eCFP 24
5.0.2 Impact of using Different Vt Devices on eCFP 25
5.0.3 Impact of Low Leakage Array and Enabling Retention Mode on oCFP 26
5.0.4 Complete Sustainability Metric 27
5.0.5 Impact of Retention Mode on Arrays of Different Size 28

6 Conclusion **30**

List of Tables

3.1 Logic value obtained for Q i.e. the output for RBL and Qb i.e. the output for RBLB under different possible cases for 10T and 9T implementations . . . 8

3.2 Logic value obtained for Q i.e. the output for BL and Qb i.e. the output for BLB under different possible cases for 6T implementations 14

3.3 Comparison of Different Architectures Based on Power,Performance,Area and security 16

5.1 Embodied Footprint Table for 10T, 9T, and 6T architectures showing comparison for with porosity and without porosity implementations. 24

5.2 Embodied footprint table for 10T, 9T, and 6T architectures for standard Vt and high Vt array implementations. 26

List of Figures

2.1 ChaCha20 encryption algorithm logic 3

2.2 ChaCha20 encryption algorithm : Quarter Round 4

2.3 The above figure shows single column representation. The basic column structure is same for all three implementations (10T, 9T and 6T) 5

3.1 10T SRAM-bitcell [2] 8

3.2 10T IMC implementation [2] 9

3.3 Cross-section of layout for 128 x128 instance for 10T bit cell implementation in 65nm technology. A single column with 4 rows is highlighted. 9

3.4 Welch's test reults for 10T bitcell implementation 10

3.5 9T SRAM-bitcell [14] 11

3.6 Cross-section of layout for 128 x128 instance for 9T bit cell implementation in 65nm technology. A single column with 4 rows is highlighted. 12

3.7 Welch's test results for 9T bitcell implementation 12

3.8 6T SRAM-bitcell [13] 13

3.9 IMC in 6T bitcell [10] 14

3.10 IMC Logic implementation in 6T bitcell [10] 15

3.11 Cross-section of layout for 128 x128 instance for 6T bit cell implementation in 65nm technology. A single column with 4 rows is highlighted. 15

3.12 Welch's test results for 6T bitcell implementation 16

4.1 Diagrammatic explanation of introducing porosity in design. 19

5.1 Graph depicting impact of introducing porosity in design 25

5.2 Graph showing embodied footprint comparison for standard Vt array and high Vt array 26

5.3 Graph for oCFP comparison across original, low leakage array and low leakage with retention mode implementations. 27

5.4 Sustainability metric graph for different implementations under different scenarios. 28

5.5 Impact of retention mode on arrays of different sizes. 29

Abbreviations

IMC	In Memory Compute
SRAM	Static Random Access Memory
RCS	Read Compute Store
SCA	Side Channel Attack
WWL	Write Word Line
RWL	Read Word Line
BL	Bitline
BLB	Bitline Bar
eCFP	Embodied Carbon Footprint
oCFP	Operational Carbon Footprint
DPQ	Dynamic Power Quantity
LPQ	Leakage Power Quantity
SPQ	Static Power Quantity
ATR	Active Time Ratio
STR	Standby Time Ratio
RTR	Retention Time Ratio
PF	Performance Factor
PWM	Pulse Width Modulation

Notation

$\sum$	Summation

CHAPTER 1

Introduction

1.1 Motivation

The use of electronic gadgets in our daily lives is growing rapidly. It is important that we need to keep a check on the impact these electronic systems have on the environment. This impact can be studied at two stages: first, during the fabrication of the device, and second, during its operation. Based on these two stages, there is a need for two combined metrics that can help quantify the overall environmental impact of electronic devices at the designing stage it. In recent years, the demand for IoT devices has increased significantly. IoT devices process a significant amount of sensitive data, making security a major concern [3]. IoT devices have limited resources to implement encryption algorithms that offer protection against side-channel attacks. An algorithm like ChaCha20 requires high bit-width and many operations, which can be challenging for resource-constrained devices. In-Memory Computing (IMC) using SRAM provides an efficient way to implement the ChaCha20 encryption algorithm. The design presented in [2] uses a 10T SRAM bitcell. It is important to analyze this design from a sustainability perspective. The current evaluation of designs is mainly based on power, performance, and area (PPA). A sustainability metric should also be included to make the analysis more complete. For this purpose, ChaCha20 encryption using IMC should be implemented with different types of SRAM bitcells commonly used in the industry to explore implementations that have lower power consumption and area but offer similar performance. This motivates the exploration of other bitcells, such as 9T and 6T, for implementing encryption and performing a comparative sustainability analysis without compromising on security.

1.2 Sustainability in Semiconductor Industry

With rapid technological progress in both fabrication and operation of chips, it has become increasingly important to evaluate its impact on environmental sustainability. As highlighted in [4], TSMC's electricity consumption alone surpassed that of an entire nation—Sri Lanka. To regulate carbon emissions from industrial activities, protocols such as the Greenhouse Gas Protocol have been established. These protocols categorize emissions into three distinct scopes, encompassing everything from direct emissions by company-owned assets to those produced

during manufacturing and even post-fabrication stages like logistics and transportation [5]. Furthermore, comprehensive frameworks have been introduced to track carbon emissions throughout the various stages of the Product Development Life Cycle (PDLC) [6]. Since the fabrication stage contributes the most to total emissions, these frameworks primarily focus on emissions arising from the manufacturing process [7]–[8].

1.3 In Memory Compute

In-memory computing (IMC) helps mitigate the von Neumann bottleneck [9]. Traditional architectures consume significant power when transferring data between memory and the processor, especially when handling large datasets for encryption, making the system highly energy-intensive. Reducing this data movement can significantly lower power consumption, and this is where IMC proves beneficial. IMC enables computation directly within memory, leveraging SRAM to perform fundamental logic operations such as AND/NAND and OR/NOR. By incorporating additional near-memory compute elements, complete encryption can be executed within the memory itself.

In IMC, the read cycle differs from a conventional read operation. During IMC execution, computation co-occurs with the read operation within a single clock cycle. This process, known as the Read-Compute-Store (RCS) cycle [10], involves reading data from the bit cells, performing computations, and writing the results back to the bit cells—all within the same clock cycle. Multiple rows are activated simultaneously, and based on the data stored in bit cell, the bit line and bit line-bar remain charged or discharge. The sense amplifier then detects the voltage difference between BL and BLB, interpreting it as a logical 0 or 1.

CHAPTER 2

Background and Related Work

2.1 ChaCha20 and it’s IMC implementation

2.1.1 ChaCha20

ChaCha20 is a stream cipher encryption algorithm derived from Salsa20 [11]. It processes a 512-bit input by repeatedly applying Addition, XOR, and Rotation operations. This input includes 128 bits of constants, 256 bits of key, 32 bits of block counter, and 96 bits of nonce (a number used only once). The algorithm arranges these 512 bits into 16 words, each 32 bits wide, and generates a keystream from them. It then XORs this keystream with the plaintext to produce the ciphertext. ChaCha20 performs 20 rounds of operations—10 odd and 10 even. In odd rounds, it operates on columns of words, while in even rounds, it operates diagonally. Each round performs four quarter-round operations in parallel, and each quarter-round works on four words. Fig. 2.1 shows the block diagram for logic of chacha20 encryption algorithm. Fig 2.2 shows the logic for single quarter round funtion.

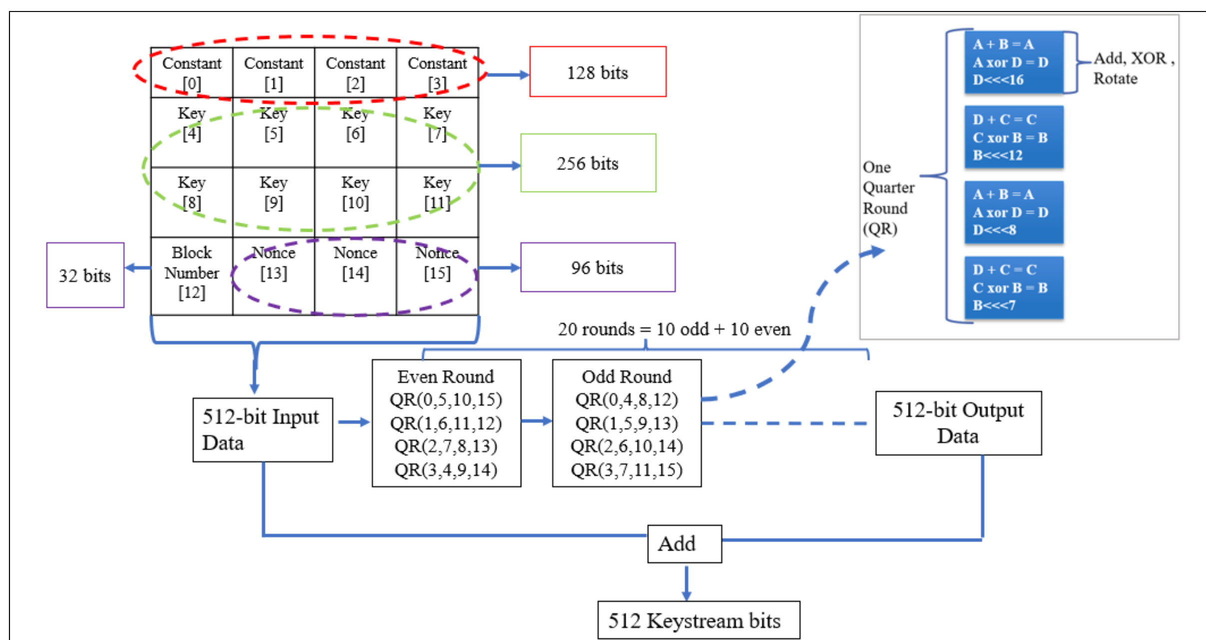
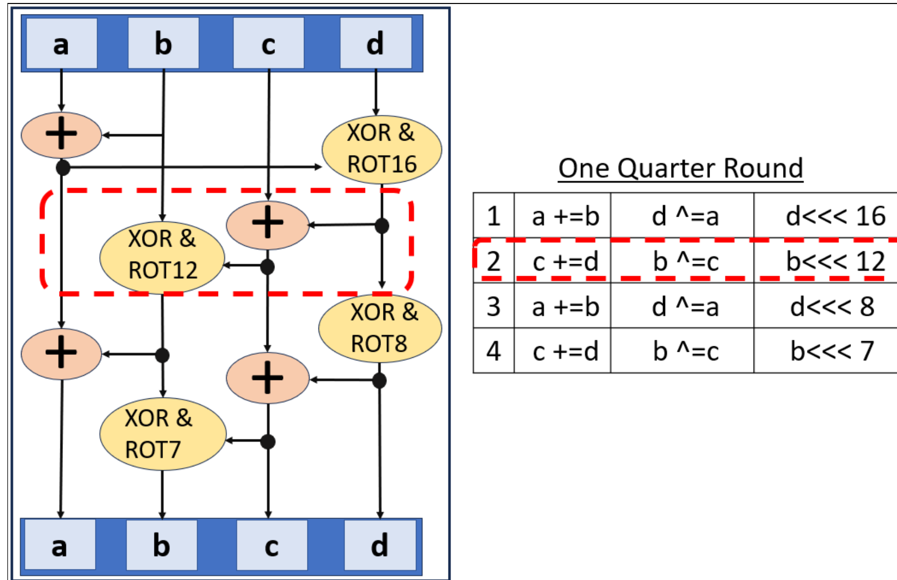


Figure 2.1: ChaCha20 encryption algorithm logic



1	a +=b	d ^=a	d<<< 16
2	c +=d	b ^=c	b<<< 12
3	a +=b	d ^=a	d<<< 8
4	c +=d	b ^=c	b<<< 7

Figure 2.2: ChaCha20 encryption algorithm : Quarter Round

2.1.2 Implementing ChaCha20 using IMC

The ChaCha20 encryption algorithm is implemented using In-Memory Computing (IMC), which allows computation to occur directly within the memory array by leveraging fundamental logic operations [2]. IMC architectures exploit gates such as NAND/AND and OR/NOR through near-memory compute elements to realize complex operations required for ChaCha20, including XOR, addition, and rotation. In this implementation, the 512-bits used in ChaCha20 is arranged into a 4×128 memory array, divided into 16 words of 32 bits each. Each column in the array incorporates additional circuitry, including an adder, a rotation unit (designed using an 8×1 multiplexer), and a logic gate to perform XOR functionality. Two read word lines (RWL1 and RWL2) are activated simultaneously during computation. The values stored in the corresponding bit cells determine whether the bit line (BL) and its complementary bit line (BLB) remain at the charged supply voltage (Vdd) or discharge. Specifically, when both bit cells store the same value (either '00' or '11'), either BL or BLB discharges, when the values differ (i.e., '01' or '10'), both bit lines discharge. A skewed sense amplifier detects this behaviour by interpreting any bit line voltage below 50% of Vdd as logic '0'. This behaviour enables the 10T bit cell to produce an AND function on BL and a NAND function on BLB. These outputs are then passed through a NOR gate to derive the XOR of the two stored values. The resulting XOR output is fed into an adder, which accepts a carry input from the adjacent column, enabling in-memory addition. Through this architecture, the design performs ChaCha20's core operations—XOR, addition, and rotation—within the memory array itself, reducing data movement and enhancing computational efficiency. In a single clock cycle, the data is read from the bitcell, processed, and then written back to the designated bitcell—this complete sequence is referred to as a RCS cycle. Following each quarter round operation, the

resulting bits are rotated and rearranged in such a way that they align within the same column, ensuring they are readily accessible for the next cycle of computation.

The fig.2.3 shows single column structure for implementing ChaCha20 encryption algorithm using IMC. The figure clearly shows the IMC elements and the near-memory compute elements used in the design.

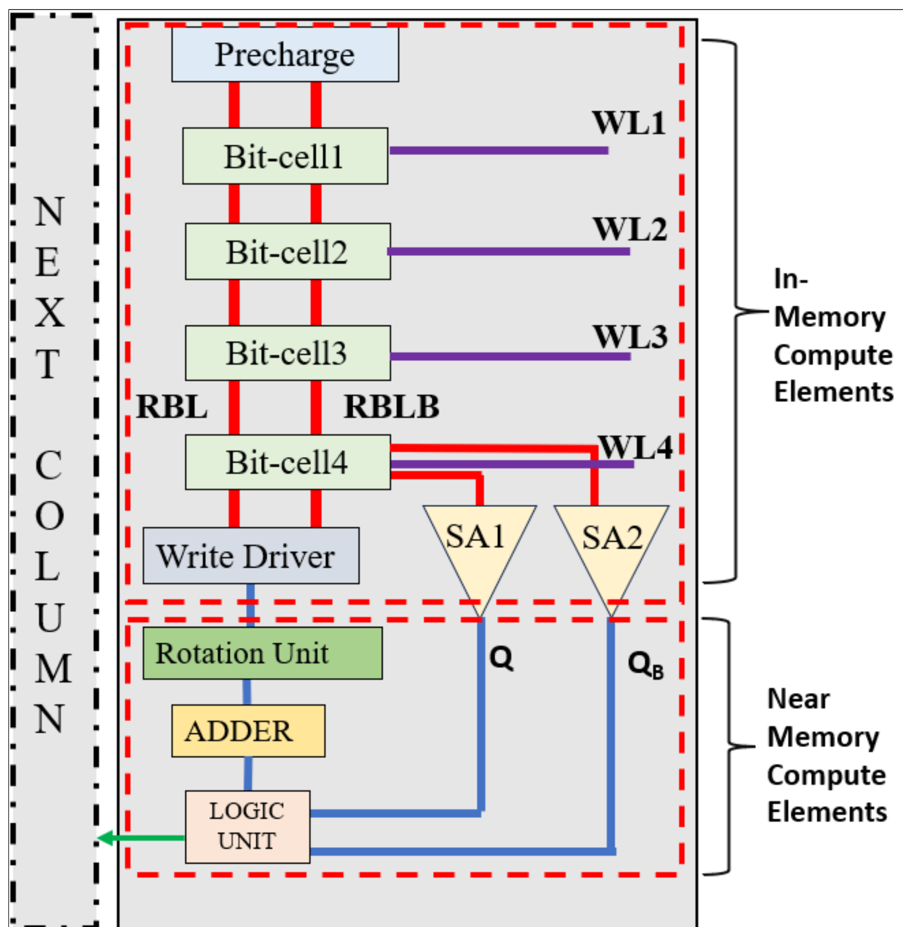


Figure 2.3: The above figure shows single column representation. The basic column structure is same for all three implementations (10T, 9T and 6T)

2.2 Welch's T-Test

When implementing an encryption algorithm, designers must ensure it is secure against Side Channel Attacks (SCA). SCAs exploit physical leakages—such as power consumption, current, or electromagnetic radiation—without requiring modifications to the target device. These attacks allow adversaries to recover secret information by analyzing these leakages. As described in [12], SCAs fall into two main categories: active attacks, where attackers manipulate the device's behavior, and passive attacks, where they observe the system without interfering with its functionality or performance. Passive attacks rely on power consumption, timing information, or electromagnetic emissions to extract sensitive data. In our work, we focus on

securing the implementation against power-based SCAs. We use Welch's t-test to statistically analyze whether the power consumption across different read cycles reveals any information that could compromise the encryption. Eq.2.1 is used to calculate t value for a given architecture. In this equation $\bar{X_1}$ and $\bar{X_2}$ are the mean power values of two different datasets, s_1^2 and s_2^2 are their variances, and n_1 and n_2 are the respective sample sizes.

$$t = \frac{\bar{X_1} - \bar{X_2}}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \tag{2.1}$$

We ensure that the power traces corresponding to various read operations remain indistinguishable so that even if an adversary accesses these traces, they cannot deduce the encrypted data. We perform encryption using multiple input datasets and extract the corresponding power samples. We then categorize these power traces into two groups. We calculate the mean, standard deviation, and variance for each sample. Using these values and the total number of samples in each , we compute the t-value based on Welch's t-test formula. If the resulting t-value remains within ±4.5, we consider the implementation secure against power-based side-channel attacks.

CHAPTER 3

Implementation using different SRAM Architectures

3.0.1 Implementation using 10T bitcell

3.0.1.1 10T Bitcell Working

The fig. 3.1 below illustrates the structure of a 10T bitcell. This cell builds upon the traditional 6T bitcell [13] used for write operations, with four additional NMOS transistors added to create dedicated read ports. Specifically, the read port includes two NMOS transistors for each bit line: RBL (read bit line) and RBLB (read bit line complement), separating read and write paths. Write Operation: The write operation in the 10T bitcell closely resembles the conventional 6T bitcell. Initially, both write bitlines (WBL and WBLB) are precharged to Vdd. The write driver then sets the appropriate data on WBL and WBLB. Activating the write word line (WWL), active high turns on transistors N1 and N2. These transistors pass the write data to the internal nodes BLTI and BLFI, effectively storing the value in the cell. Read Operation: During a read operation, both RBL and RBLB are precharged to Vdd. The read behavior depends on the stored data at the internal nodes BLTI and BLFI. For example, if the bitcell stores a logic '1' (i.e., BLTI is high and BLFI is low), transistor N6 turns on. When the read word line (RWL) is activated, it enables both N5 and N6, allowing RBL to discharge. On the other hand, since N8 remains off, RBLB stays at Vdd. The differential voltage between RBL and RBLB is then sensed using an inverter-based sense amplifier connected at the end of both bit lines, producing the final output.

Figure 3.1: 10T SRAM-bitcell [2]

3.0.1.2 IMC Implementation using 10T

Both RBL and RBLB are initially pre-charged to Vdd. In a single cycle, the read word lines (RWL) of two rows involved in the in-memory computation (IMC) are activated simultaneously. The resulting discharge behavior of RBL and RBLB depends on the data stored in the corresponding bit cells. If both bit cells have a logic '1' at their BLTI nodes, RBL will discharge, while if both store a logic '0', RBL will remain at Vdd. When the stored values are different—either '01' or '10'—both RBL and RBLB discharge partially, dropping to approximately 50% of Vdd. The inverting sense amplifier is designed with a skew to interpret a 50% discharge as logic '0'. As a result, the RBL line provides the logical AND of the two bits, and RBLB gives the NAND output. The outcomes are summarized in table 3.1.

Bitcell 1	**Bitcell 2**	**Q**	**Qb**
0	0	0	1
0	1	0	0
1	0	0	0
1	1	1	0

Table 3.1: Logic value obtained for Q i.e. the output for RBL and Qb i.e. the output for RBLB under different possible cases for 10T and 9T implementations

The waveform in fig 3.2 shows the IMC implementation in 10T bitcell. It can be seen that RBL shows AND logic implementation and RBLB shows NAND.

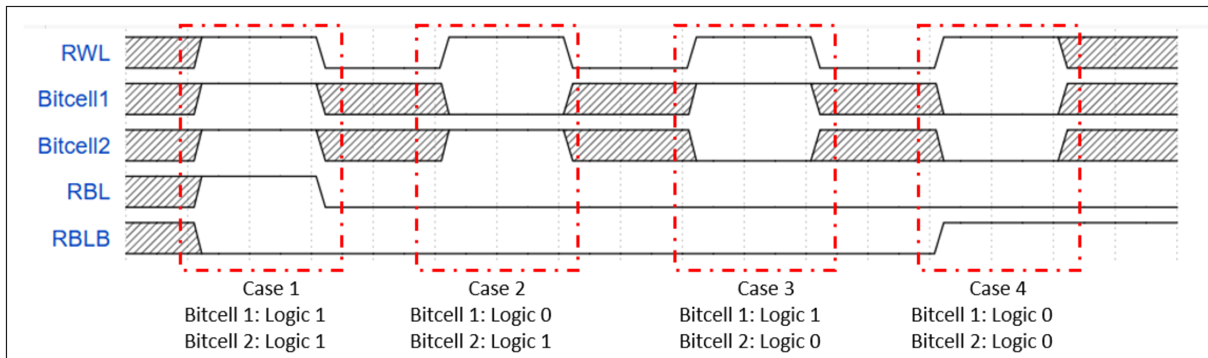


Figure 3.2: 10T IMC implementation [2]

3.0.1.3 Layout explanation

The layout for the proposed implementation is designed using 65nm CMOS technology. The layout is designed keeping in mind the practical implementations for a memory size of 128X128 array. The bit cell layout is first created, with bit lines placed on the lowest possible metal layer. Metal layer M2 is used for the bit lines, as this layer is more cost-effective during fabrication compared to higher metal layers, especially given the large number of bit lines. The column-level periphery components—such as the pre-charge circuit, write driver, sense amplifier, and additional near-memory compute units, including the XOR logic unit, adder, and rotation unit—are all designed to match the width of the bit cell. This uniform width enables vertical stacking of all these elements within a single column. On adjacent columns, devices are placed strategically to allow flipping and overlapping, which helps achieve a denser layout. The read and write word lines are routed using metal layer M3, while power rails are placed on M4. The complete layout for the 10T bit cell-based implementation is illustrated in Fig.3.3.

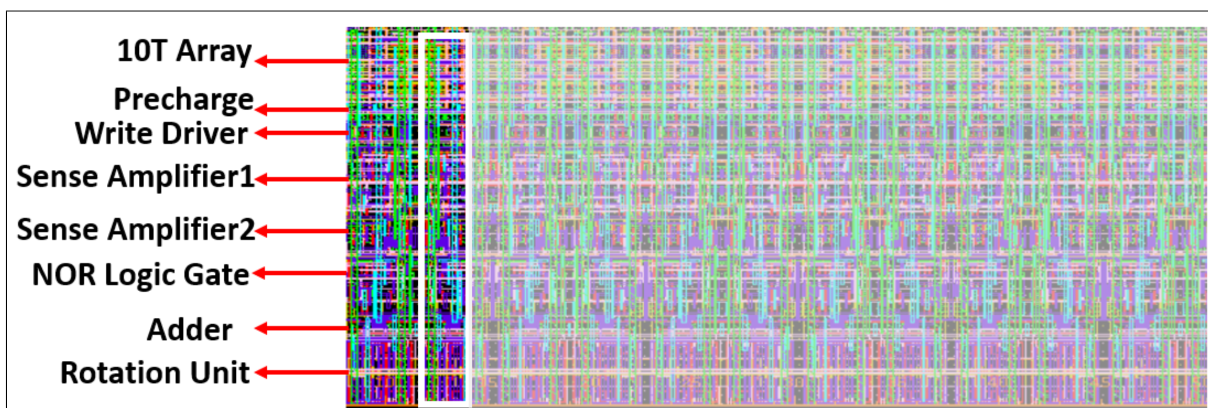


Figure 3.3: Cross-section of layout for 128 x128 instance for 10T bit cell implementation in 65nm technology. A single column with 4 rows is highlighted.

3.0.1.4 Welch's t-test result

Using the previously described Welch's t-test methodology, we performed the side-channel analysis for the 10T implementation. The results are presented in Fig. 3.4. The graph illustrates that, across all input samples, the calculated t-values consistently lie within the acceptable threshold range of -4.5 to 4.5. This confirms that the implementation is secure against power-based side-channel attacks.

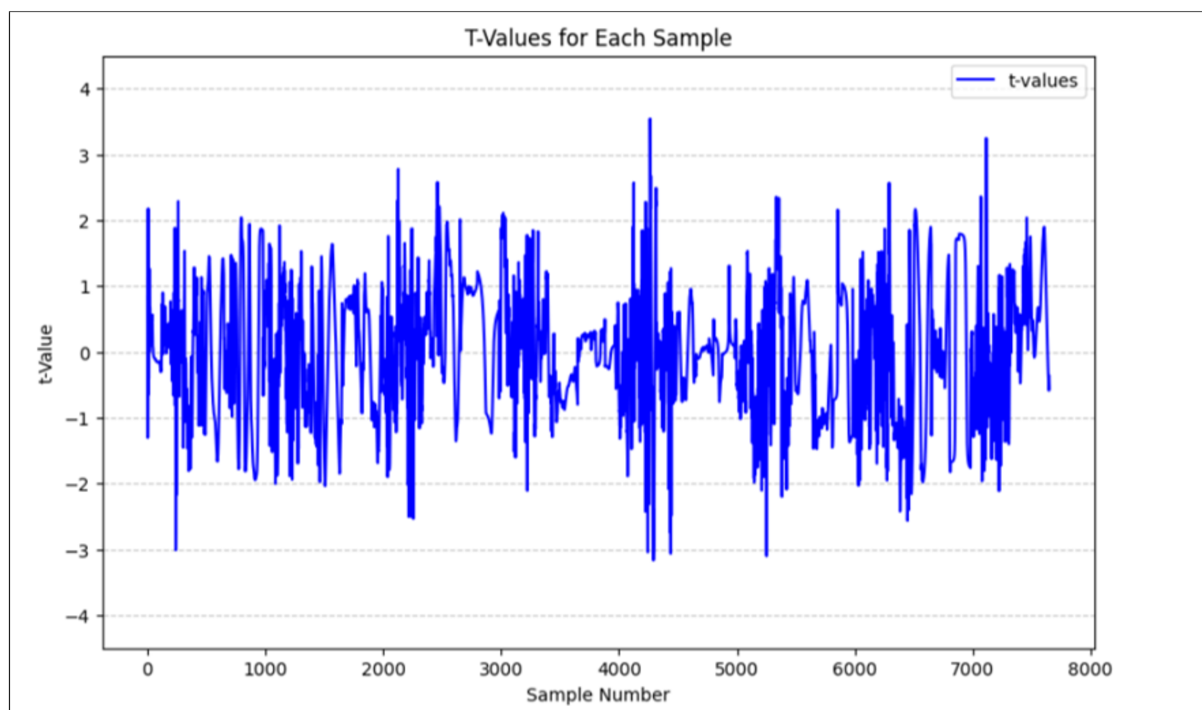


Figure 3.4: Welch's test reults for 10T bitcell implementation

3.0.2 Implementation using 9T bitcell

3.0.2.1 9T Bitcell

Figure 3.5 illustrates the 9T [14] bitcell. The write operation in this design follows the conventional 6T approach, similar to what was previously described for the 10T bitcell. The key difference lies in the read port configuration—here, the NMOS transistor (N7) responsible for controlling the read operation is shared between both the RBL and RBLB paths. During the write operation, activating the write word line (WWL) enables the transfer of data from the bit lines to the internal storage nodes BLTI and BLFI via transistors N1 and N2. For the read operation, the bit lines are initially precharged, WWL remains low, and the read word line (RWL) is activated. Based on the values stored at BLTI and BLFI, either N5 or N6 is turned on. For instance, if BLTI holds a '1' and BLFI holds a '0', transistor N5 conducts, allowing RBL to discharge through N5 and the shared transistor N7. The sense amplifier, implemented

as a skewed inverter, detects the bit line discharge and outputs the corresponding logic level.

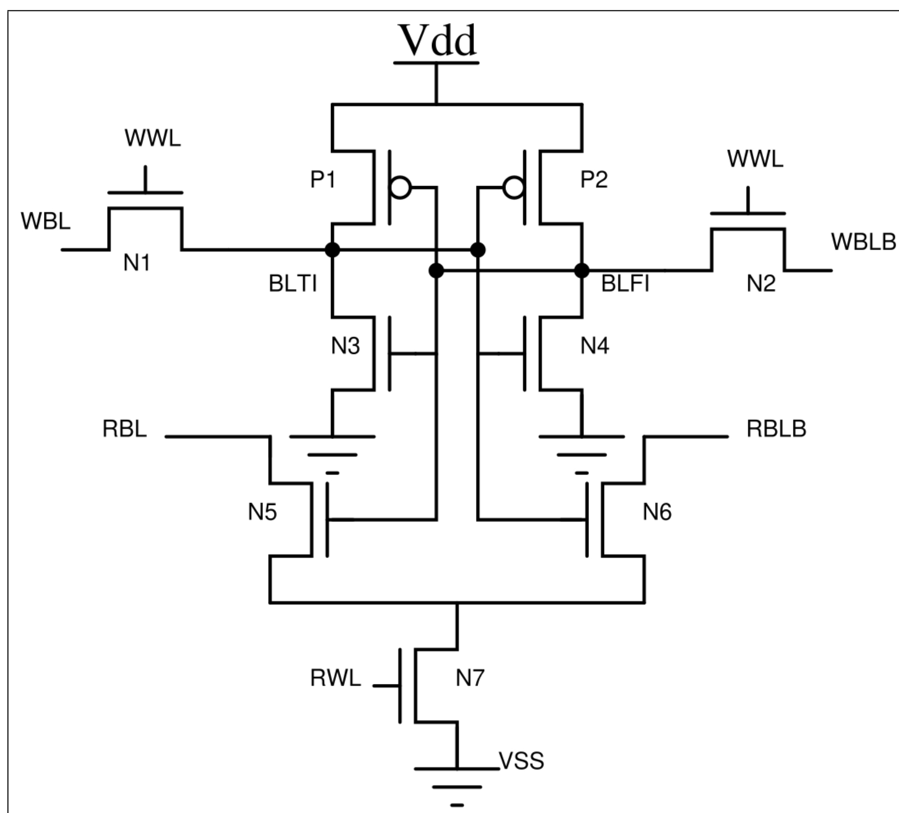


Figure 3.5: 9T SRAM-bitcell [14]

3.0.2.2 IMC explanation in bitcell

The IMC operation in 9T bitcell is similar to 10T bitcell. The BL exhibits AND functionality and BLB exhibits NAND functionality. The output table is same as 10T implementation shown in table 3.1.

3.0.2.3 Layout explanation

The layout for 9T implementation is designed similarly as for 10T implementation. The layout is shown below in fig. 3.6. The layout is designed for 128X128 array implementation.

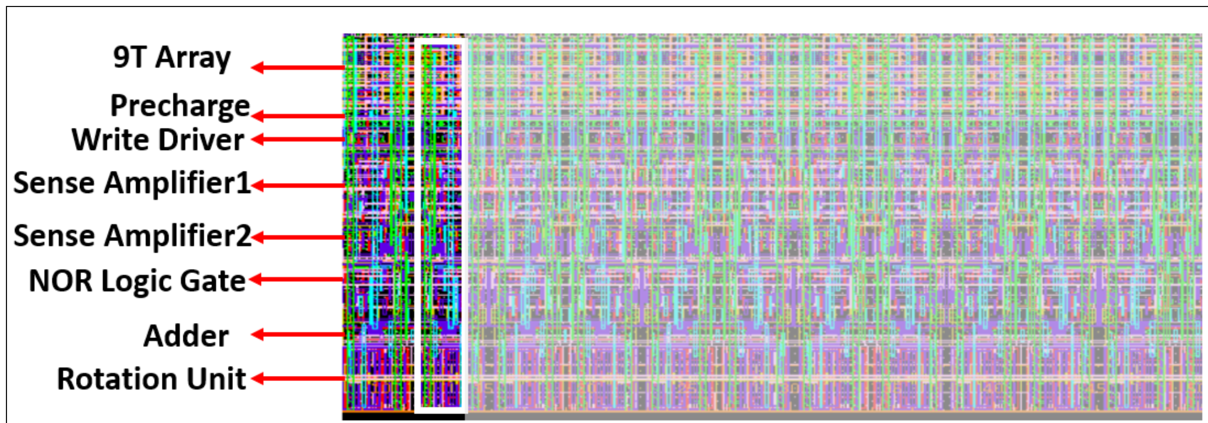


Figure 3.6: Cross-section of layout for 128 x128 instance for 9T bit cell implementation in 65nm technology. A single column with 4 rows is highlighted.

3.0.2.4 Welch's t-test result

Based on the earlier description of welch's test the analysis is done for 10T implementation. The result for this is shown in fig.3.7. The graph shows that for all the given input samples the t-value remains well with the given range of -4.5 to 4.5. The implementation is secure against power-based side channel attack.

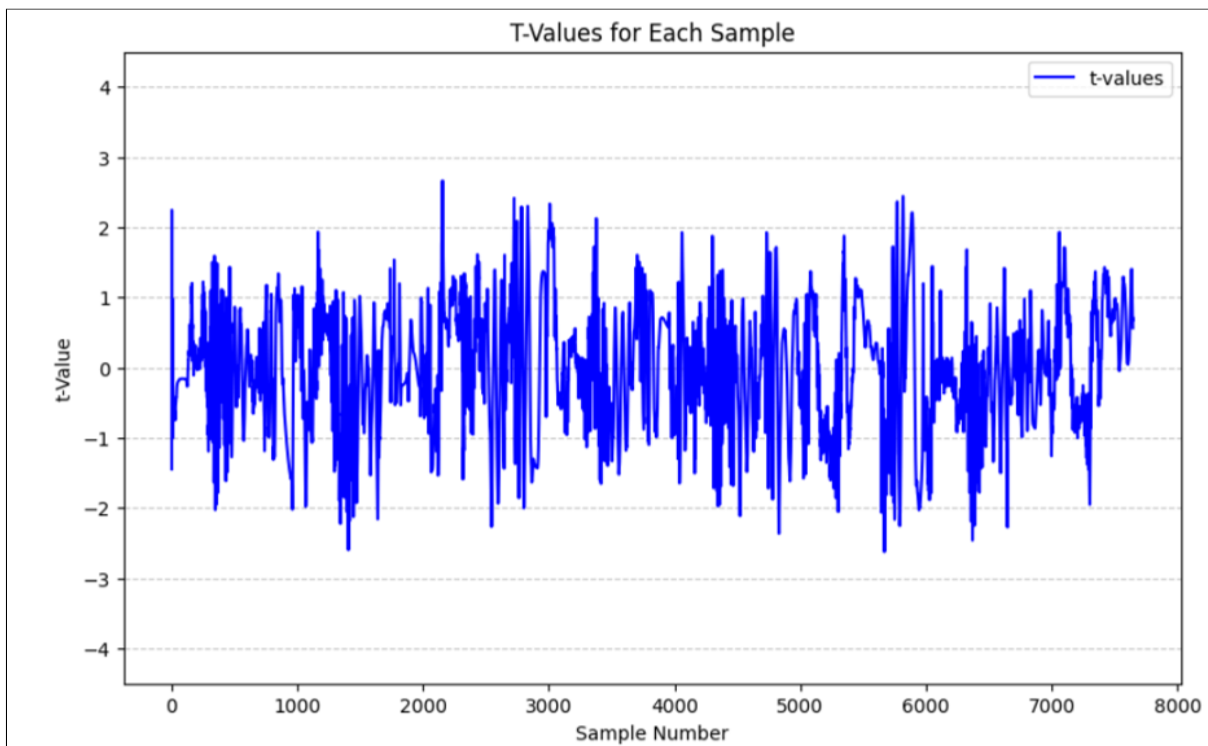


Figure 3.7: Welch's test results for 9T bitcell implementation

3.0.3 Implementation using 6T bitcell

3.0.3.1 Bitcell working

The 6T [13] bitcell consists of six transistors, as illustrated in the fig. 3.8. It features a single port that serves both read and write operations, with the word line (WWL) activated during both processes. In a write operation, the data is first driven onto the bit lines BL and BLB using the right driver. Once WWL is enabled, this data is transferred to the internal storage nodes BLTI and BLFI via access transistors. For instance, if a '1' needs to be written to BLTI (assuming it initially held a '0') and a '0' to BLFI, activating WWL allows transistor N1 to discharge BL and charge BLTI accordingly. During a read operation, both BL and BLB are initially precharged to Vdd. Activating WWL connects the internal nodes to the bit lines. Depending on the stored data, one of the bit lines discharges while the other remains at Vdd. A sense amplifier connected at the end of the bit lines detects this differential voltage, amplifies it, and forwards the output to a latch for storage.

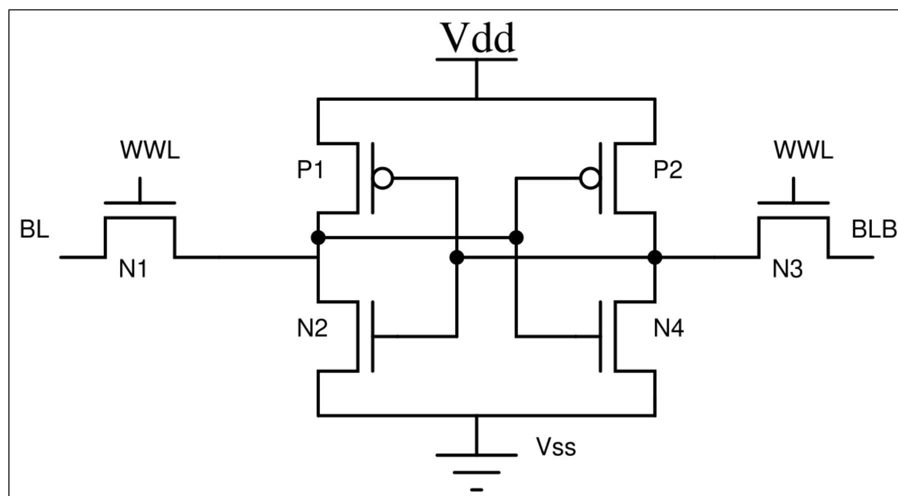


Figure 3.8: 6T SRAM-bitcell [13]

3.0.3.2 IMC implementation

Implementing In-Memory Computing (IMC) using a 6T bitcell is more challenging compared to 10T or 9T bit cells because the read and write paths are shared. Read failures can occur if two-word lines are activated simultaneously and the corresponding bit cells hold opposite data values (i.e., 01 or 10). This is because both bit lines are initially precharged to Vdd, and simultaneous activation can cause both bit lines to discharge. If one bitcell discharges the bit line while the other is still active, it can lead to data corruption in the second bitcell. Instead of activating both word lines simultaneously, they are enabled sequentially to address this issue [10]. The first-word line is turned on just long enough to allow the bit line to discharge by 50%. Then, the second word line is activated. If the second bitcell holds the same value as the first,

the bit line continues to discharge; if it holds the opposite value, the other bit line begins to discharge. Since only one bitcell is accessed at any given moment, data corruption is avoided. A sense amplifier is connected to both bit lines to detect the voltage levels and amplify the resulting output accordingly. Fig.3.9 shows the waveform to illustrate the sequential triggering of word lines to enable IMC using 6T bitcell.

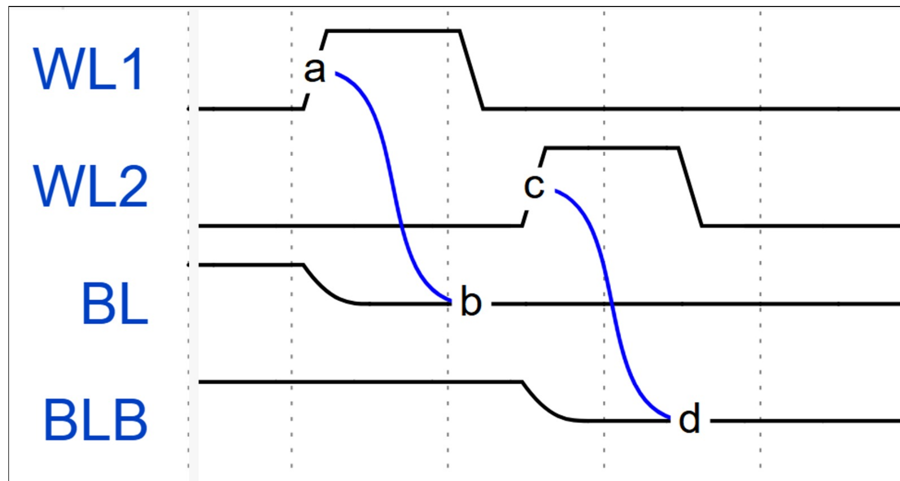


Figure 3.9: IMC in 6T bitcell [10]

The table 3.2 shows the logic outputs on Q and Qb terminals from sense amplifier for BL and BLB respectively. It can be seen that BL shows OR functionality and BLB shows NOR functionality. The IMC functionality on BL and BLB is shown in fig. 3.10.

Bitcell 1	**Bitcell 2**	**Q**	**Qb**
0	0	0	1
0	1	1	1
1	0	1	1
1	1	1	0

Table 3.2: Logic value obtained for Q i.e. the output for BL and Qb i.e. the output for BLB under different possible cases for 6T implementations

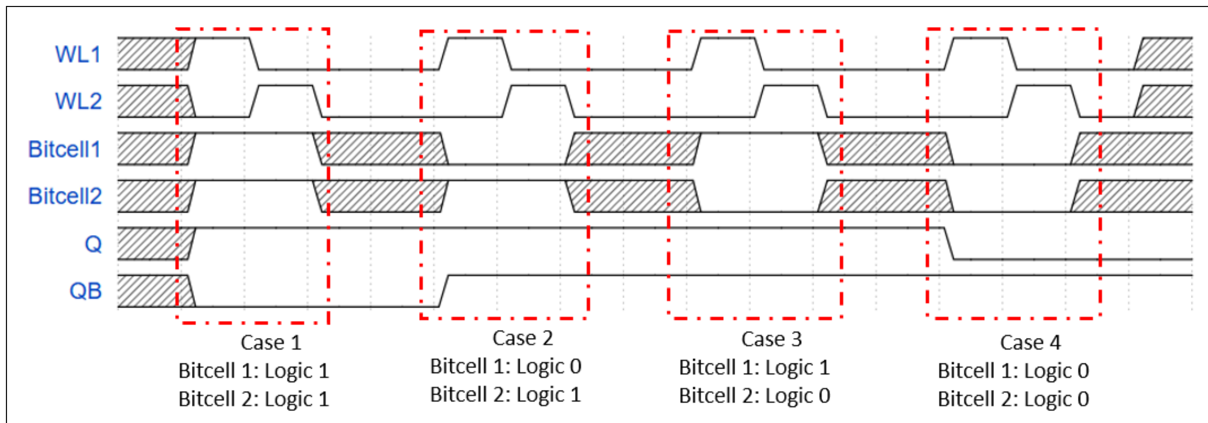


Figure 3.10: IMC Logic implementation in 6T bitcell [10]

3.0.3.3 Layout explanation

The fig. 3.11 shows the layout for 6T implementation. The layout is designed for 128X128 array implementation using the similar approach as for 10T bitcell.

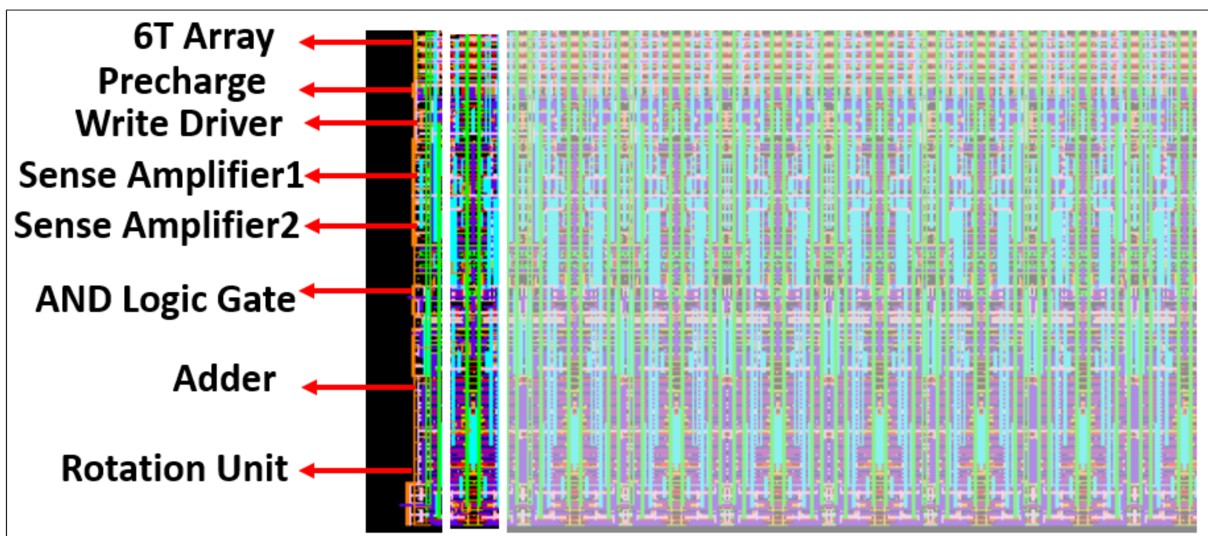


Figure 3.11: Cross-section of layout for 128 x128 instance for 6T bit cell implementation in 65nm technology. A single column with 4 rows is highlighted.

3.0.3.4 Welch's test result

Based on the earlier description of welch's test the analysis is done for 6T implementation. The result for this is shown in fig.3.12. The graph shows that for all the given input samples the t-value remains well with the given range of -4.5 to 4.5. The implementation is secure against power-based side channel attack

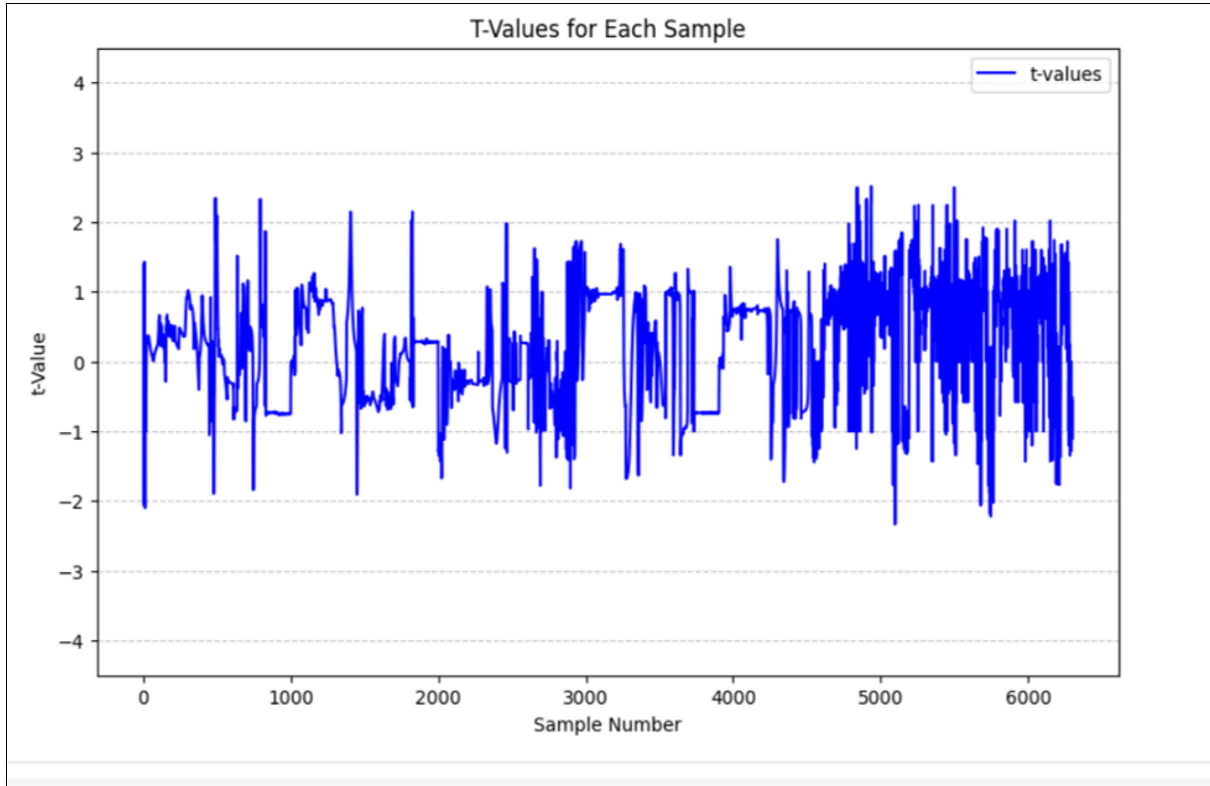


Figure 3.12: Welch's test results for 6T bitcell implementation

3.0.4 Power, Performance, Area and Security Comparison

Table 3.3 presents a comparative summary of power, performance, area, and security metrics across the different implementations. The reported power represents the total energy consumed during the encryption of 512 bits. Performance is evaluated based on the duration required to complete one RCS cycle for each design. Among the three, the 6T implementation exhibits the poorest performance, primarily due to sequential word line activation rather than simultaneous switching. Additionally, since the read and write paths are shared, the bitlines must be precharged twice within a single RCS cycle, once before computation and again before writing back, further degrading its performance. The area values provided in the table correspond to a 128 × 128 array, offering insight into the physical area of each design. Lastly, the Welch's t-test scores indicate that all three implementations maintain security against side-channel attacks (SCA), as the results fall comfortably within the acceptable range.

Architecture	Power (mW)	Performance (ns)	Area (μm^2)	Welch's T-test Score
10T	1.43	3.8	375608	-3.1 to 3.5
9T	1.62	4.1	360417	-2.6 to 2.6
6T	1.87	10.8	134883	-2.3 to 2.5

Table 3.3: Comparison of Different Architectures Based on Power,Performance,Area and security

CHAPTER 4

Sustainability Metric

4.0.1 Sustainability Analysis

Integrated circuit (IC) fabrication carries environmental costs that arise during the manufacturing and operational phases of the chip's lifecycle. These environmental impacts are broadly categorized into two levels: embodied footprint and operational footprint. The embodied footprint refers to the environmental burden associated with the chip's production, including factors such as the materials used, energy consumed during fabrication, and other resources involved in manufacturing. Once the chip is fabricated and deployed for use, it consumes energy during its operation, contributing to its operational footprint [15]. Together, these two components determine the overall sustainability impact of an IC. In this work, we present formulas to quantify the embodied and operational footprints for any SRAM architecture. These formulas are then applied to evaluate SRAM-based architectures' sustainability metrics for implementing ChaCha20 encryption through IMC.

4.0.2 Embodied Footprint

During the fabrication of IC, multiple factors influence environmental sustainability. Fabrication of a chip involves steps such as photolithography, doping, and etching. All these steps are both energy and material-intensive. These factors contribute to the embodied footprint of a chip, which measures the environmental costs before the chip is put into operation. This work presents a metric that can be utilized to calculate the embodied footprint of a given SRAM architecture and quantify the environmental impact during fabrication. The embodied footprint metric can be expressed using Eq.4.1.

$$\begin{aligned}\text{Embodied Footprint} = {} & \text{Base Fabrication Energy} \times \text{Area of Macro} \times \\ & \text{Extra Processing Step Impact} \times \text{Porosity Factor}\end{aligned} \tag{4.1}$$

4.0.2.1 Base Fabrication Energy

Chip area plays a crucial role in determining the embodied carbon impact. As the die size increases, fewer dies can be produced from a single silicon wafer. However, the fabrication

consumes nearly the same amount of electricity, photoresist, and other chemicals regardless of how many dies are produced per wafer. Consequently, each larger die inherits a proportionally greater share of these process-related emissions. Moreover, larger dies exhibit lower manufacturing yield because they capture more defects, necessitating additional wafers to achieve the required number of functional parts—further amplifying resource use and emissions. Therefore, every additional square millimetre in the chip layout contributes directly to higher material consumption and a greater embodied carbon footprint before the chip leaves the fabrication facility. This work references the energy consumption for fabricating a 300mm wafer at the 65nm technology node—up to metal layer 6—[16]. Based on this, a parameter is derived to estimate the energy consumption per unit area for a given chip design.

$$\text{Energy per IP} = \left(\frac{\text{Standard wafer energy consumption}}{\text{Total wafer surface area}}\right) \times \text{Area of Macro} \tag{4.2}$$

4.0.2.2 Extra Processing Step Impact

The number of masks used during fabrication has a direct impact on the embodied footprint of a chip. As the number of masks increases, fabrication becomes more time-consuming and resource-intensive, leading to higher energy consumption. A standard number of reference masks is typically used to fabricate a chip for each technology node. Any additional masks beyond this reference set contribute to extra energy usage. According to reference [16], the energy consumption for fabricating a chip with up to six metal layers is 4204 kWh, while it rises to 4895 kWh for eight metal layers. Each additional metal layer generally requires two extra masks—one for the metal layer itself and one for the corresponding vias. In this analysis, it is assumed that each additional mask adds a similar amount of energy overhead. Based on the increase in energy consumption from six to eight metal layers (an increment of four masks), we can estimate the energy cost per mask. This allows us to define a mask factor, representing the energy impact of adding each extra mask. This factor can then be incorporated into the overall energy estimation to account for the embodied energy overhead due to additional mask usage during fabrication. The energy increase from using six metal layers (M6) to eight metal layers (M8) is calculated as:

$$\frac{4895 - 4204}{4204} = 0.164$$

This 16.4% rise in energy corresponds to the use of 4 additional masks (2 for metal layers and 2 for vias). Thus, the energy increase per additional mask is:

$$\frac{0.164}{4} = 0.041 \approx 4.1\%$$

This means that each extra mask contributes approximately a 4.1% increment in energy

consumption, assuming uniform energy usage per mask. To capture this impact, the following parameter is defined to estimate the effect of additional masks on the embodied footprint:

$$\text{Mask Overhead} = ((\text{Total Mask} - \text{Ref. Mask}) \times \text{MF} + 1) \tag{4.3}$$

Where:

- MF is the Mask Factor, representing the energy increment per additional mask i.e. 0.041.
- A constant value of 1 is added to ensure that when the total number of masks equals the reference count, the mask overhead remains 1, indicating no additional energy impact.

4.0.2.3 Porosity Factor

In SRAM architectures, a key design parameter called porosity can affect the chip's area and the resources used during fabrication, impacting the embodied footprint. Porosity refers to the availability of routing space within the design and indicates how many extra metal tracks can pass through a particular region. This analysis evaluates the porosity of the M2 and M4 metal layers using 65nm technology. As illustrated in the diagrams, the spacing between the SRAM array and the decoder can be arranged in two ways. In Figure 4.1.A, the decoder is placed right next to the SRAM array with minimum spacing, leaving no room for additional routing tracks. In contrast, Figure 4.1.B shows a layout where increased spacing allows multiple M2 and M4 tracks to pass through.

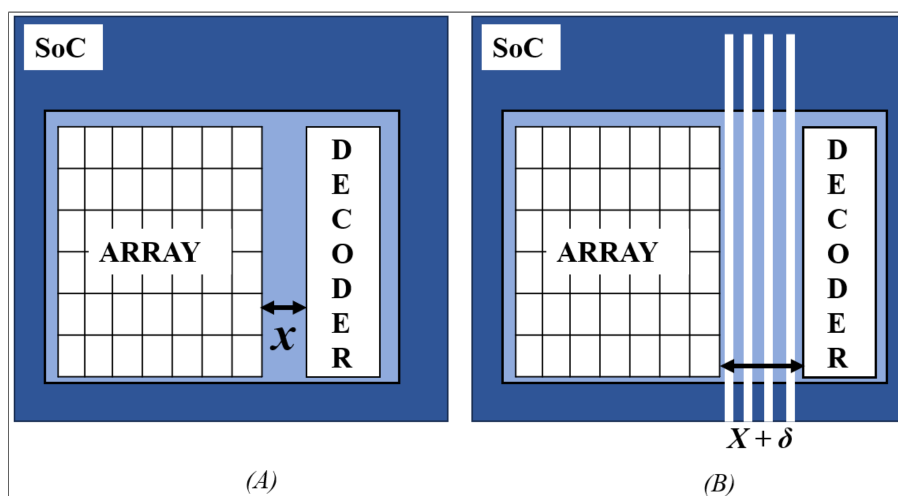


Figure 4.1: Diagrammatic explanation of introducing porosity in design.

By enabling more routing within lower metal layers, porosity helps reduce the need for routing in higher metal layers, which typically require more masks and thicker metal lines. These higher layers consume more energy and materials, increasing the embodied footprint. Although increasing porosity slightly expands the chip area, it improves routing flexibility,

reduces SoC congestion, and can lower the overall environmental impact of the chip. The porosity factor is calculated as shown below in eq 4.4:

$$\text{Porosity Factor} = \left(1 - \frac{\text{Porosity offered}}{\text{Total Number of Tracks}}\right) \tag{4.4}$$

Porosity offered is the number of tracks that can be routed through the given channel spacing and total number of tracks is the number of tracks possible in a given metal layer as per the width of the design.

4.0.3 Operational Carbon Footprint

After fabrication, when the customer uses the product, it consumes a certain amount of energy, which also impacts environmental sustainability. An additional metric called the operational carbon footprint is introduced to account for this alongside the embodied carbon footprint. The operational carbon footprint measures the device's energy, typically expressed in milliwatt-hours (mWh). This energy consumption is calculated by considering power usage during the dynamic, static, and leakage phases over the device's total operating time. In this work, the operating time for an IoT device is considered to be about 20,000 hours over a period of five years. However, the actual operating time will vary depending on the specific application, ensuring that the operational carbon footprint is tailored to real-world usage scenarios. The operational carbon footprint can be expressed using the eq.4.5.

$$\begin{aligned}\text{Operational Footprint} = (&\text{Dynamic Power Quantity} + \text{Static Power Quantity} \\ &+ \text{Leakage Power Quantity}) \times \text{Operating Time}\end{aligned} \tag{4.5}$$

4.0.3.1 Dynamic Power (DPQ)

When there is a switching in the signals from one logic state to another, the internal capacitances within the circuit charge and discharge. This charging and discharging of capacitors consume power. This power consumption is accounted as dynamic power quantity. It is measured as shown in eq.4.6:

$$\text{DPQ} = \sum \alpha V_{\text{op}} f_{\text{op}} Q_{\text{dyn}} \tag{4.6}$$

The activity factor is assumed to be 0.01 for the SRAM architectures considered. This low value reflects the nature of IoT devices, where memory units typically remain in standby or retention mode most of the time, resulting in minimal switching activity. The operating frequency (f_{op}) is the external clock frequency of the device, which is assumed to be 1000 MHz

for typical IoT applications. The operating voltage (V_{op}) is taken as 1.2 V, representing the supply voltage of the overall system. Dynamic power is consumed when signals toggle, causing internal capacitors to charge and discharge. The dynamic charge (Q_{dyn}) refers to the charge involved in this toggling process. In the case of SRAM architectures that implement encryption algorithms using In-Memory Computing (IMC), this dynamic charge consumption is evaluated per RCS (Read-Compute-Store) cycle. These assumptions form the basis for estimating the dynamic power consumption in the operational carbon footprint analysis of SRAM-based IMC designs.

4.0.3.2 Static Power (SPQ)

This parameter represents the power a circuit consumes in a steady (non-switching) state. While it is not required to estimate the operational carbon footprint of SRAM architectures specifically, it is included to make the formula applicable to a broader range of analog circuits, such as charge pumps.

$$\text{SPQ} = \sum I_{\text{Bias}} \times V_{\text{Bias}} \tag{4.7}$$

In eq.4.7 I_{bias} is the bias current and V_{bias} is the corresponding bias voltage. This term captures the static power consumed due to constant biasing in analog components.

4.0.3.3 Leakage Power Quantity (LPQ)

The power consumed by a circuit when it is not switching and remains in a static state is known as leakage power. This power results from leakage currents that persist even when the memory is not actively reading or writing. In SRAM architectures, leakage can occur in three different operational modes:

- Standby mode – when the memory is idle but powered
- Active mode – during normal read/write operations
- Retention mode – when the memory retains data with minimal power

The total leakage power is calculated by considering the contribution of each mode over the total operation time. It can be expressed as:

$$\text{LPQ} = P_{\text{standby}} + P_{\text{active}} + P_{\text{retention}} \tag{4.8}$$

This approach provides a weighted average of leakage across all operating modes.

Where individual leakage components are calculated as shown below:

$$P_{\text{active}} = I_{\text{active}} V_{\text{op}} \left(ATR \cdot PF^{PWM} \right) \tag{4.9}$$

$$P_{\text{standby}} = I_{\text{standby}} V_{\text{standby}} \left(\text{STR} + (PWM \cdot ART \cdot (1 - PF)) \cdot k1 \right) \tag{4.10}$$

$$P_{\text{retention}} = I_{\text{ret}} V_{\text{ret}} \left(RTR + (PWM \cdot ART \cdot (1 - PF)) \cdot k2 \right) \tag{4.11}$$

In all three modes of memory operation—active, standby, and retention—specific terms like ATR, PF, PWM, k1,k2 are introduced to account for how energy consumption varies with area and time. This is because, throughout the operational lifetime of a device, not all parts of the IC consume the same amount of energy at the same time. The proportion of chip area in a particular mode at any given time influences the overall energy usage.

- ATR (Active Time Ratio) quantifies the fraction of the total chip area operating in active mode out of the total operating time of device.
- STR (Standby Time Ratio) quantifies the fraction of the total chip area operating in standby mode out of the total operating time of device.
- RTR (Retention Time Ratio) quantifies the fraction of the total chip area operating in retention mode out of the total operating time of device.
- PF (Performance Factor) is the ratio of the execution time of the design to the worst-case execution time of a chosen reference design.
- PWM (Pulse Width Modulation) indicates whether power-saving techniques are enabled. In case these techniques are enabled then the design is expected to complete its operation faster than the reference design and then transition into standby or retention mode for rest of the duration of operation. This allows the device to save leakage power.
- k1 and k2 are scaling factors used for giving the time proportion for standby and retention respectively mode when PWM is enabled as given in eq.4.12 and eq.4.13.

$$k1 = \frac{SRT}{SRT + RTR} \tag{4.12}$$

$$k2 = \frac{RTR}{SRT + RTR} \tag{4.13}$$

For designs where PWM is used (i.e. when PWM is active), PF becomes particularly important. For instance, if the worst-case execution time for a reference design is T_{on}, and the current design finishes its operation in less than T_{on}, the remaining time can be in standby or retention mode. This transition reduces energy consumption since leakage power is significantly lower in these modes. The total operation time of the SRAM architecture is divided into different modes based on both area usage and time, especially for IoT applications. The typical distribution is as follows:

- 1% of the chip area is in active mode
- 9% in standby mode
- 20% in retention mode
- 70% in switched-off mode

The parameters ATR (Active Area Time Ratio), STR (Standby Area Time Ratio), and RTR (Retention Area Time Ratio) are incorporated into the leakage power estimation to account for energy use over time and area. Switched-off mode is excluded from this calculation since leakage current is negligible when the circuit is powered down.

Mode-wise Power Description:

- Active Leakage: This occurs when the memory is actively functioning, such as when data encryption is performed or when read/write operations are occurring. The current drawn from the supply at VDD = 1.2 V is measured to get the active leakage current based on which active leakage power is calculated using the eq.4.9.
- Standby Leakage: When the memory is connected to supply, and the clock is high, but there is no data activity (i.e., no signal switching), the circuit is in standby mode. The voltage remains at 1.2 V, and the power measured during this idle period is defined as standby leakage as shown in eq.4.10
- Retention Leakage: In retention mode, the memory array remains powered at a reduced voltage to retain the stored data while the peripheral circuits are shut down. The array voltage is lowered to 0.8 V, and the periphery voltage is reduced to 0 V. The leakage current during this mode is referred to as retention leakage.This is calculated as given in eq. 4.11

This area and operating mode-dependent breakdown allows for a more accurate estimate of leakage power consumption, contributing to a comprehensive assessment of the operational carbon footprint.

CHAPTER 5

Results

Application of the above derived formula in different cases to measure sustainability of the architectures. The array size for all the there implementations is considered to be of 128X128. The embodied footprint is estimated as per size of the memory. The operational footprint on the other hand is estimated based on performance as discussed earlier. The operational footprint in this work is estimated on the basis of encrypting 512 bits from the 128 X128 bits.

5.0.1 Impact of Introducing Porosity in Design on eCFP

A channel is introduced between the array and decoder to include porosity in the given design, as illustrated in the figure above. This increases the overall area, but the number of tracks that can be routed through the IP increases by a factor of two as porosity impacts the M2 and M4 layers (the layers where tracks run vertically in this analysis). As a result, the overall embodied footprint is reduced since the total number of tracks that can be accommodated within the IP increases. The results shown below in table 5.1 indicate that with porosity, the design's embodied footprint is lower than when there is no porosity. 10T bitcell implementation has the highest overall embodied footprint where as 6T has the lowest. This is mainly because 10T implementation has the highest area out of the given three implementations, as discussed above. For all the three implementations the embodied footprint reduced but even after including porosity 10T has the highest embodied footprint. As shown in table 5.1 porosity factor for all the three architectures is introduced which depends on the channel width spacing.

Parameter	**10T Implementation**		**9T Implementation**		**6T Implementation**	
	Without Porosity	**With Porosity**	**Without Porosity**	**With Porosity**	**Without Porosity**	**With Porosity**
Energy Consumption per IP (KWh)	0.022	0.023	0.021	0.022	0.008	0.0081
Mask overhead	1	1	1	1	1	1
Porosity Factor	1	0.9810	1	0.9804	1	0.97002
Total embodied footprint (mWh)	22333	22122	21429	21220	8016	7895

Table 5.1: Embodied Footprint Table for 10T, 9T, and 6T architectures showing comparison for with porosity and without porosity implementations.

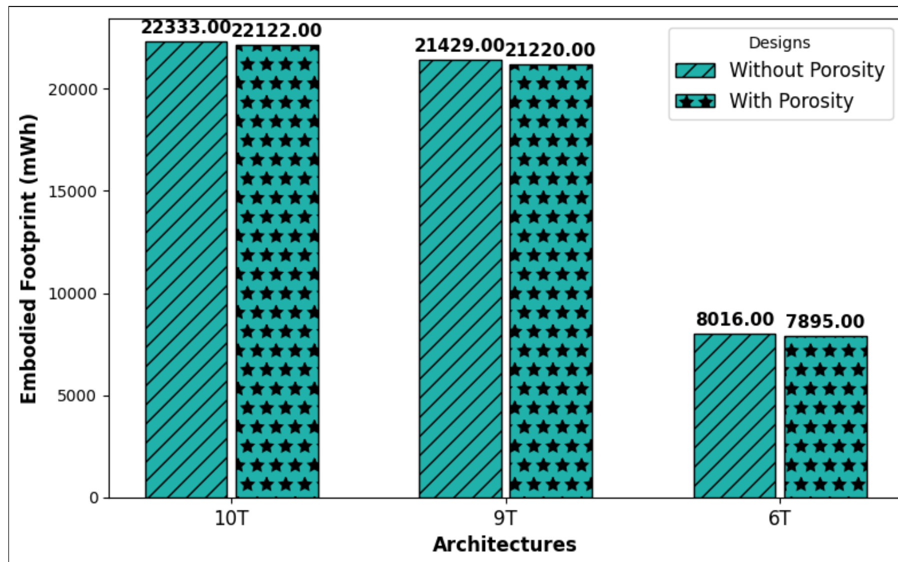


Figure 5.1: Graph depicting impact of introducing porosity in design

5.0.2 Impact of using Different Vt Devices on eCFP

The memory array can be implemented using low-leakage, high-threshold voltage (High-Vt) devices, while the periphery remains on standard-threshold voltage (Standard-Vt) devices. This helps reduce array leakage, lowering the operational energy footprint. However, using different Vt devices for the array and periphery increases fabrication complexity. The mask count matches the reference process when the array and periphery use the same Vt devices. In contrast, introducing High-voltage devices in the array adds two extra masks. Based on the previously derived formula, each additional mask contributes a 4% increase in energy consumption. Therefore, using High-Vt for the array and Standard-Vt for the periphery results in an 8% higher embodied energy footprint than a uniform Standard-Vt design. The graph in fig 5.2 below shows the results obtained. The table 5.2 shows that when low leakage array is used i.e. high Vt bitcell is used mask overhead factor increases by around 8 percent as 2 additional masks are used when moving from standard Vt to high Vt.

Parameter	10T Implementation		9T Implementation		6T Implementation	
	Standard Vt	High Vt	Standard Vt	High Vt	Standard Vt	High Vt
Energy Consumption per IP (KWh)	0.022	0.022	0.021	0.021	0.008	0.008
Mask overhead	1	1.09	1	1.09	1	1.09
Porosity Factor	1	1	1	1	1	1
Total embodied footprint (mWh)	22333	24343	21429	23358	8016	8737

Table 5.2: Embodied footprint table for 10T, 9T, and 6T architectures for standard Vt and high Vt array implementations.

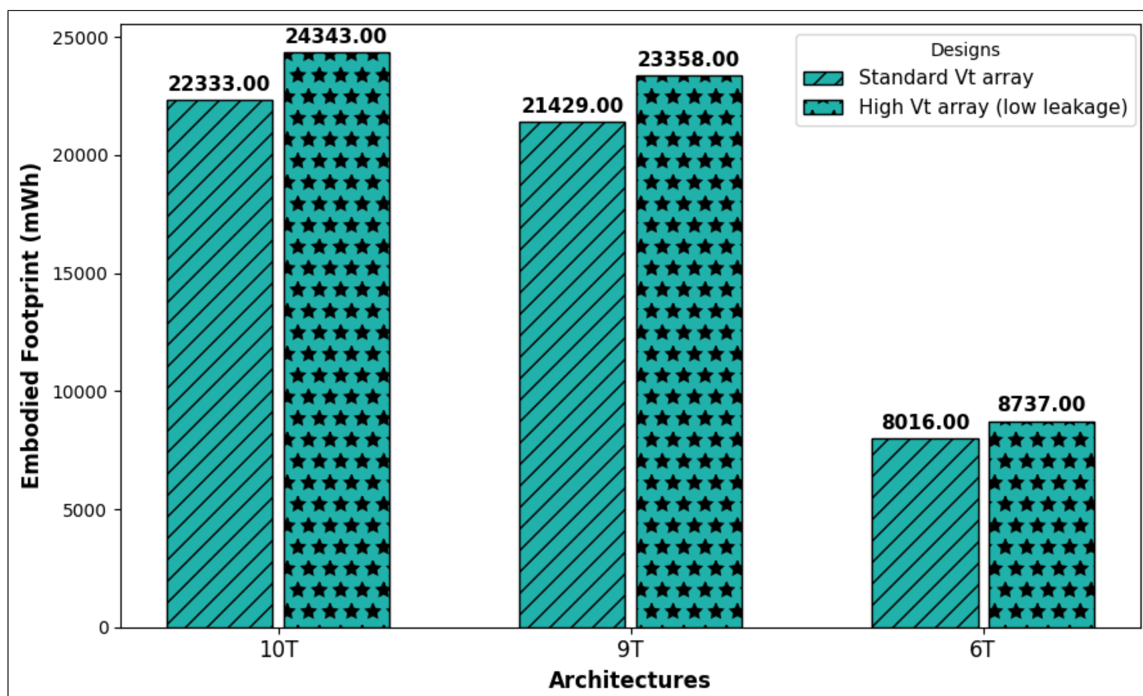


Figure 5.2: Graph showing embodied footprint comparison for standard Vt array and high Vt array

5.0.3 Impact of Low Leakage Array and Enabling Retention Mode on oCFP

When the memory array is designed using low-leakage, high-threshold voltage (High-Vt) devices, the embodied footprint increases slightly due to additional processing steps, but the operational footprint decreases. As previously derived, the operational footprint is primarily influenced by leakage power across different modes, such as active, standby, and retention. Using High-voltage devices in the array reduces active-mode leakage, lowering the operational footprint. Enabling retention mode further reduces approximately 10% compared to the base-

line case. In retention mode, the array operates at a reduced voltage of 0.8V (retention voltage) while the periphery is switched off. This significantly reduces leakage, resulting in a lower operational footprint. In contrast, if retention mode is not enabled, the memory enters standby mode, where both the array and periphery remain connected to a 1.2V supply—leading to higher leakage. The graph below shows that the operational footprint decreases when a High-Vt array is used and is further reduced when retention mode is enabled. It is observed that 6T implementation has the highest operational footprint out of the three given implementations as the time taken to complete single RCS cycle is highest for 6T implementation. The reason for higher time is explained above. 10T and 9T implementations have very closely related operational footprint values.The graph in fig 5.3 shows that by using a lower leakage array, the operational footprint reduces as the leakage power consumed is lower. Further if the retention mode is enabled then the oCFP further reduces.

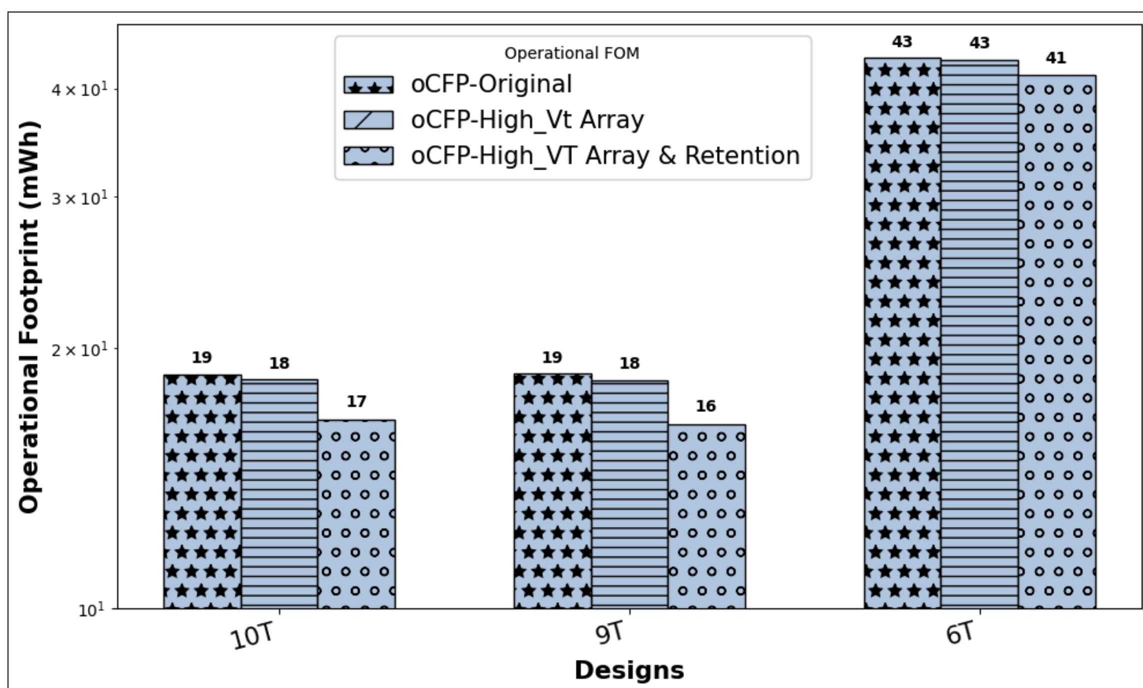


Figure 5.3: Graph for oCFP comparison across original, low leakage array and low leakage with retention mode implementations.

5.0.4 Complete Sustainability Metric

The sustainability metric, as defined above, is the sum of the embodied footprint and the operational footprint. It is important to analyse the cumulative impact of both these components across all architectures under different scenarios. The graph is shown in fig.5.4.

$$\text{Sustainability Metric} = \text{eCFP} + \text{oCFP} \tag{5.1}$$

The graph below illustrates three cases for combined footprint analysis:

- Combined metric for the original case (i.e., without retention mode).
- Combined metric for the High-Vt case, reflecting the cumulative impact of using High-Vt devices in the memory array on both operational and embodied footprints
- Combined metric for the case where retention mode is enabled alongside a High-Vt memory array

In the second case, using a High-Vt memory array without retention mode reduces the operational footprint but increases the embodied footprint by 8%. As a result, the overall combined footprint is higher than in the original case. In the third case, enabling retention mode significantly reduces the operational footprint, as shown above. However, the combined footprint remains slightly higher than the original case due to the additional circuitry required for retention mode, though it is still lower than the High-Vt-only case.

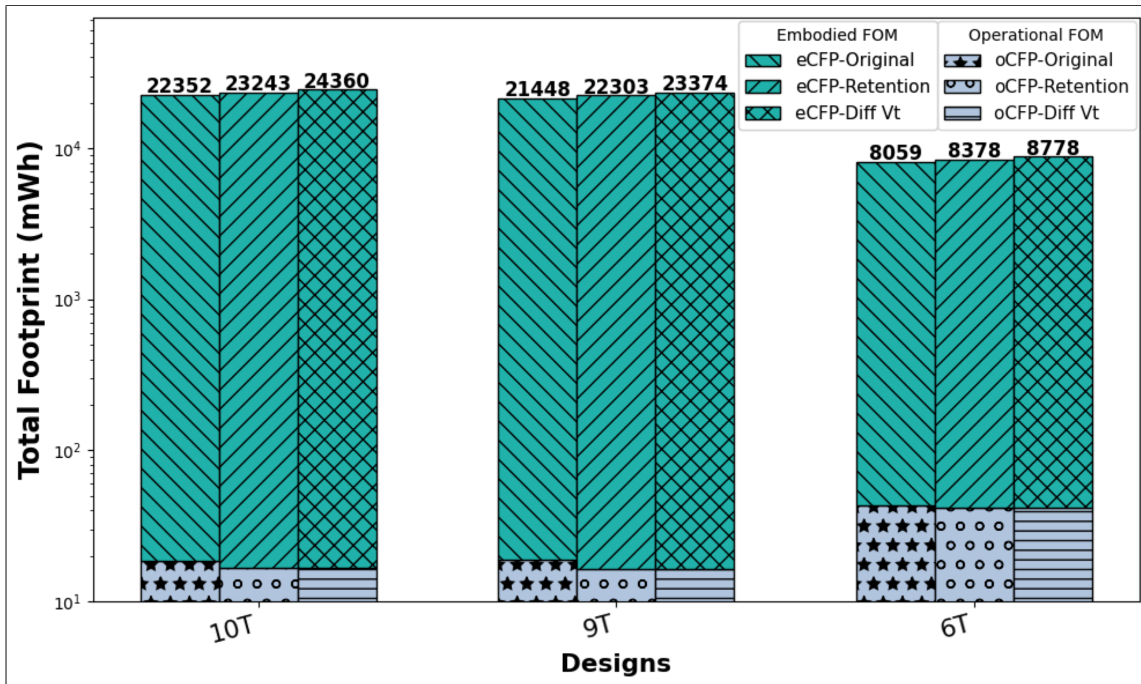


Figure 5.4: Sustainability metric graph for different implementations under different scenarios.

5.0.5 Impact of Retention Mode on Arrays of Different Size

Fig. 5.5 shows the impact of retention mode on arrays of different sizes. It can be seen that the impact of retention mode on operational footprint is more evident and the gain is much higher for larger sized array of 1024 x 1024 than smaller size array of 128 X 128. The analysis is done for 6T bitcell implementation.

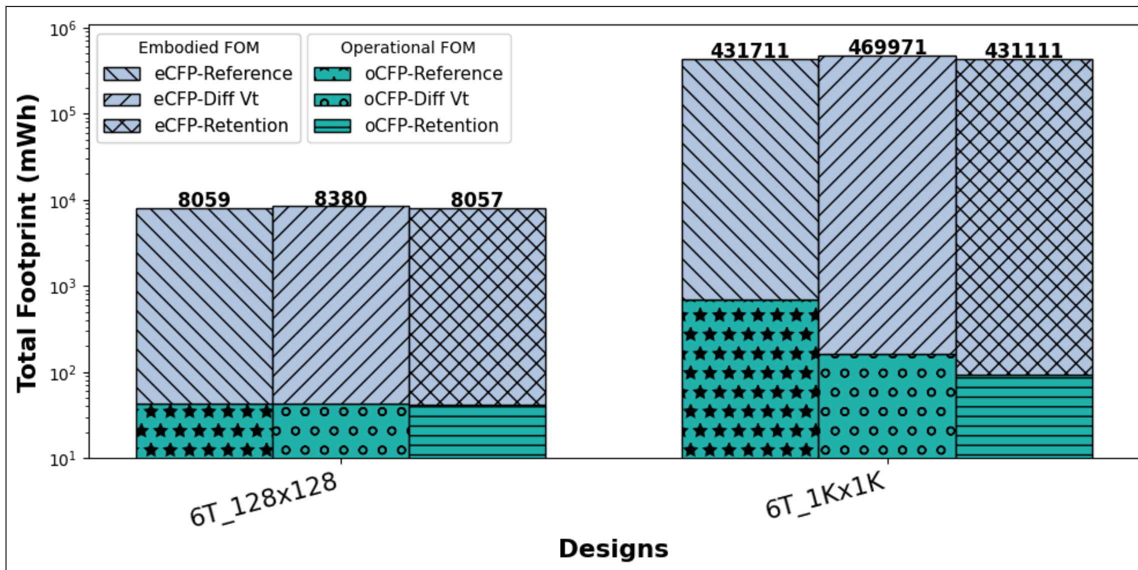


Figure 5.5: Impact of retention mode on arrays of different sizes.

CHAPTER 6

Conclusion

In this work, we proposed a set of sustainability metrics to evaluate the environmental impact of SRAM-based architecture designs, particularly in implementing the ChaCha20 encryption algorithm. The developed metrics are categorized into two major components: the embodied carbon footprint (eCFP), which reflects the environmental cost of fabrication and material use, and the operational carbon footprint (oCFP), which accounts for energy consumption during runtime. Our analysis across three SRAM implementations—10T, 9T, and 6T—revealed key trade-offs. The 10T architecture demonstrated the highest performance and lowest power consumption but also incurred the most significant area and, consequently, the highest eCFP. In contrast, the 6T implementation offered significant area savings (64% less than 10T), resulting in the lowest eCFP, but with 32% higher power consumption, leading to a higher oCFP. The 9T design emerged as a balanced alternative, achieving similar performance to 10T while reducing area by 4% and power consumption by approximately 13%. The security analysis using Welch's t-test confirmed that all three implementations maintained resistance against side-channel attacks, with t-scores well within acceptable bounds. This indicates that optimizing for power and area does not compromise the system's overall security. This work demonstrates that sustainable hardware design can be achieved without sacrificing performance or security. The proposed metrics provide a structured framework for evaluating architectural trade-offs, enabling designers to make informed choices based on their application's specific sustainability, performance, and security requirements.

In reference to the thesis-

Our research paper titled **"Framework to Estimate and Benchmark Sustainability of Circuit Design"** has been accepted in IEEE Region 10 Symposium 2025 (TENSYMP) which will be presented in New Zealand, July 2025.

Our research paper titled **"Sustainably Secure: ChaCha20 Encryption Based on In-Memory Compute"** is currently under review in ISLPED 2025 (IEEE/ACM International Symposium on Low Power Electronics and Design).

References

[1] S. Rizvi, J. Pfeffer III, A. Kurtz, and M. Rizvi, “Securing the Internet of Things (IoT): A Security Taxonomy for IoT,” in 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications / 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), pp. 163–168, IEEE, 2018.

[2] M. Aamir, S. Sharma, and A. Grover, "ChaCha20-in-memory for side-channel resistance in IoT edge-node devices," IEEE Open Journal of Circuits and Systems, vol. 2, pp. 833-842, 2021, doi:

[3] "State of IoT 2024: Number of connected IoT devices growing 13% to 18.8 billion globally," IoT-Analytics, 2024. Available: https://iot-analytics.com/wp/wp-content/uploads/2024/09/INSIGHTS-RELEASE-Number-of-connected-IoT-devices-vf.pdf 10.1109/OJCAS.2021.3127273.

[4] ”Chipmaking’s next big thing guzzles as much power as entire countries,” https://www.tbsnews.net/bloomberg-special/chipmakingsnext-big-thing-guzzles-much-power-entire-countries-484338, (accessed March 1, 2025).

[5] U. Gupta et al., "Chasing Carbon: The Elusive Environmental Footprint of Computing," in IEEE Micro, vol. 42, no. 4, pp. 37-47, 1 July-Aug. 2022, doi: 10.1109/MM.2022.3163226.

[6] M. Czerniak, D. Hacker and H. Schneider, ”Reducing the environmental impact of semiconductor manufacture,” 2016 China Semiconductor Technology International Conference (CSTIC), Shanghai, China, 2016,pp. 1-3, doi: 10.1109/CSTIC.2016.7464026.

[7] A. Meddendorf et al., “EE-Toolbox-a modular assessment system for the environmental optimization of electronics,” Jan. 2000, doi: https://doi.org/10.1109/isee.2000.857643.

[8] A. Middendorf, N. F. Nissen, L. Stobbe, O. Wittler and K. -D. Lang, ”Eco-reliability as a new approach of multi-criteria optimisation,” 2012 Electronics Goes Green 2012+, Berlin, Germany, 2012, pp. 1-6.

[9] B. Iqbal, A. Grover, and H. Rawat, "A process and data variations tolerant capacitive coupled 10T1C SRAM for in-memory compute (IMC) in deep neural network

accelerators," 2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS), Incheon, South Korea, 2022, pp. 459-462, doi: 10.1109/AICAS54282.2022.9870026.

[10] A. Agrawal, A. Jaiswal, C. Lee, and K. Roy, "X-SRAM: Enabling in-memory Boolean computations in CMOS static random access memories," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 65, no. 12, pp. 4219-4232, Dec. 2018, doi: 10.1109/TCSI.2018.2848999.

[11] D. J. Bernstein, "ChaCha, a variant of Salsa20," in Workshop Record of SASC 2008, vol. 8, pp. 3-5, 2008.

[12] S. Bhunia and M. Tehranipoor,Hardware Security. Elsevier, 2019. Available: https://www.sciencedirect.com/book/9780128124772/hardware-security.

[13] A. Mishra and A. Grover, "A 0.9V 64Mb 6T SRAM cell with read and write assist schemes in 65nm LSTP technology," 2020 24th International Symposium on VLSI Design and Test (VDAT), Bhubaneswar, India, 2020, pp. 1-4, doi: 10.1109/VDAT50263.2020.9190373.

[14] Z. Liu and V. Kursun, "High read stability and low leakage cache memory cell," in IEEE International Symposium on Circuits and Systems (ISCAS), 2007, pp. 2774-2777, doi: 10.1109/ISCAS.2007.378628.

[15] S. B. Boyd, Life-Cycle Assessment of Semiconductors, 1st ed. New York, NY, USA: Springer, 2012. DOI:https://doi.org/10.1007/978-1-4419-9988-7.

[16] D. Kline et al., ”Sustainable IC design and fabrication,” 2017 Eighth International Green and Sustainable Computing Conference (IGSC), Orlando, FL, USA, 2017, pp. 1-8, doi: 10.1109/IGCC.2017.8323572.