FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Subthreshold SRAM for IoT Systems

Artur Jorge Coutinho Ribeiro

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Ph.D. Cândido Duarte

Co-supervisor: Eng. Francisco Gonçalves

July 3, 2019

© 2019, Artur Ribeiro

Abstract

The Internet of Things (IoT) is an emerging technological paradigm, which is experiencing an exponential growth. It focuses on distributed sensing to acquire knowledge about the real world, and distributed computing for its computational capabilities. As the foreseeable aplications are extremely diverse, the market potential for technologies developed with an intended application in IoT is huge.

An IoT node is composed of a microprocessor, a communication unit (RF radio), sensors and/or actuators and a power management unit. It tries to achieve extremely high battery life (years or decades), mainly through aggressive duty cycling. The aggressive duty cycling used implies that most of the node, apart from some always-on components, stays the majority of its life in a sleep state, only awakening to perform sensing, processing and comunication. The duty cycling comes at a cost: all data must be stored in retentive memories (e.g. flash memory) or in always-on parts of the circuitry, causing either the time to wake up and time to sleep to rise for the former, or the idle power consumption of the node to escalate for the latter.

Static Random Access Memory (SRAM) is a type of non-retentive memory that can hold its data indefinitely (while powered). It is used in small embedded devices and SoC, as well as in processors cache. By designing an SRAM module for the sensors data that focuses on lowering the idle power, the overall consumption of the IoT node can be decreased, while reducing the need for extra processing time to begin and end the sleep cycle.

This work focus on designing a full-custom 180 nm CMOS SRAM for Ultra Low Power. The SRAM has 128 bytes (1024 bits) of storage, and works with a clock frequency of 128 Hz, while powered at 200 mV. This dissertation work resulted in an SRAM memory that functions at the proposed specifications, and which post-layout simulations achieved an idle power consumption of 175 pW (at the tt process corner and 25 ^{o}C) and energy per read access of 10.85 pJ (for 8 bits), while showing proper functionality for the temperature range of 20 ^{o}C to 90 ^{o}C in the tt process corner.

Keywords: IoT, SRAM, Subthreshold, ULP

Resumo

A *Internet of Things* (Internet das Coisas, IoT) é um paradigma tecnológico emergente, que está a sofrer um crescimento exponencial. Foca-se em sensorização distribuída para adquirir conhecimento sobre o mundo real, e computação distribuída para as capacidades computacionais. Como as aplicações previstas são extremamente diversas, o potencial de mercado para tecnologias desenvolvidas com aplicação pretendida em IoT é enorme.

Um nó de IoT é composto por um microprocessador, uma unidade de comunicação (rádio RF), sensores ou atuadores e uma unidade de controlo de energia. O nó tenta obter tempos de vida de bateria extremamente altos (anos ou até décadas), principalmente através de ciclos de trabalho agressivos. Os ciclos de trabalho agressivos implicam que a maioria do nó, à exceção de alguns componentes *always-on* (constantemente ligados), permanece a maior parte da sua vida num estado adormecido, sendo apenas acordados para realizar funções de sensorização, processamento e comunicação. Os ciclos de trabalho trazem uma desvantagem: todos os dados necessitam de ser guardados em memórias retentivas (e.g. memória flash) ou em partes sempre ligadas do circuito, causando ou o aumento do tempo de acordar e de adormecer para o primeiro caso, ou o aumento do consumo de energia em *idle* do nó para o segundo caso.

Memória Estática de Acesso Aleatório (*Static Random Access Memory*, SRAM) é um tipo de memória não retentiva que consegue guardar dados indefinidamente (enquanto alimentada). É normalmente usada em pequenos dispositivos embutidos e em SoC (Sistemas num Chip), assim como em caches de processadores. Ao projetar um módulo SRAM para os dados dos sensores, focando em diminuir a potência inativa, o consumo geral do nó IoT pode ser diminuido, reduzindo simultaneamente a necessidade de tempo extra de processamento para começar e terminar o ciclo de sono.

Este trabalho foca-se em projetar uma SRAM CMOS *full-custom* em 180 nm para *Ultra Low Power*. A SRAM tem 128 bytes (1024 bits) de armazenamento, e trabalha a 128 Hz de frequência de relógio, enquanto alimentada a 200 mV. O trabalho desta dissertação resultou numa memórian SRAM que funciona nas especificações propostas, e cuja simulação pós *layout* alcançou um consumo de energia em *idle* de 175 pW (no *corner* de processp tt e a 25°C), e uma energia por acesso de escrita de 10.85 pJ (para 8 bits), apresentando a funcionalidade correta para um alcance de temperatura de 20°C a 90°C, no *corner* de processo tt.

Palavras-chave: IoT, SRAM, Subthreshold, ULP

Agradecimentos

Em primeiro lugar, tenho que agradecer à minha família, em especial ao meu pai e à minha irmã. Sem o apoio que me deram durante todo o meu percurso, nada disto teria sido possível. Esta tese é o culminar de um percurso iniciado por uma pessoa que já não está cá, mas que me incutiu um enorme gosto pelo saber.

À minha namorada, Mariana, por todo o apoio e força que me deu durante este ano.

Aos meus da terrinha, que tornam os fins de semana mais especiais. Apesar de, neste último ano, os nossos encontros terem sido mais escassos, são sempre preciosos.

Aos colegas do meu ano, que me acompanharam neste percurso de cinco anos. Em especial, ao João, Henrique, Helder e Álvaro, que foram uma companhia mais próxima e constante neste curso, e ao Ricardo e Miguel, que foram uma companhia mais presente neste último semestre.

Ao Rúben, Rui, Nuno e Lisa, que iniciaram esta aventura comigo, há cinco anos atrás.

A todos do CUP, que tornaram a rotina de terça à noite dos momentos mais aguardados da semana.

Por último, uma palavra de agradecimento ao meu orientador, Cândido Duarte, pelo conhecimento e sabedoria partilhados durante a elaboração desta dissertação. Um obrigado também ao meu co-orientador, Francisco, cujo esforço e boa-vontade foram fulcrais em levar esta tese a bom porto, e ao Vasco, pela prontidão em ajudar sempre que um problema aparecia.

Artur Ribeiro

"Our virtues and our failings are inseparable, like force and matter.
When they separate, man is no more"

Nikola Tesla

Contents

1 Introduction **1**
1.1 The Internet of Things - A General View 1
1.2 Applications of IoT 2
1.3 Motivation 3
1.4 Problem Statement 4
1.5 Proposed Solution 4
1.6 Document Outline 5

2 Literature Review **7**
2.1 IoT - Internet of Things 7
2.2 SRAM 9
2.2.1 6T SRAM 10
2.2.2 8T SRAM 10
2.2.3 10T SRAM 11
2.2.4 Main SRAM Designs Comparison 11
2.2.5 SRAM Metrics 12
2.3 Subthreshold Circuits 12
2.3.1 CMOS Power Consumption 13
2.3.2 MOS EKV Model 14
2.3.3 Disadvantages of Subthreshold 16

3 SRAM Architecture **17**
3.1 10T Bitcell 18
3.1.1 Optimization 19
3.1.2 Static Noise Margin 22
3.2 8T Bitcell 25
3.3 10T SRAM Layout 25
3.4 Schematic vs Layout Comparison 28

4 SRAM Control Circuits **31**
4.1 Logic Gates 31
4.2 Logic Gates - Layout 36
4.3 Demultiplexer 41
4.3.1 Demultiplexer Layout 42
4.4 Drive Circuits 43
4.5 Drive Inverters - Layout 43
4.6 Power Gating 44
4.7 Read Circuits 44

4.8 State Machine 45
4.8.1 Flip Flop 47
4.8.2 Write State Machine 47
4.8.3 Read State Machine 48

5 Integration **51**

6 Conclusion **59**
6.1 Future Work 60

A **61**

References **65**

List of Figures

1.1 IoT Application Sectors. 2
1.2 IoT nodes connected to the cloud. 3

2.1 Common elements in an IoT node architecture. 8
2.2 Standard 6T SRAM bitcell schematic. 10
2.3 Standard 8T SRAM bitcell schematic. 11
2.4 Single Ended 10T SRAM bitcell schematic. 11
2.5 Power consumption on CMOS technologies. 13

3.1 SRAM block diagram. 17
3.2 Chosen 10T bitcell architecture - Schematic. 19
3.3 SRAM optimization steps and order. 20
3.4 Variation of the maximum and minimum functioning temperature, while varying the V_{DD} in the tt process corner, for the 10T bitcell in schematic simulation. 20
3.5 Schematic for the determination of the bitcell's SNM. 23
3.6 Butterfly plots for the calculation of the bitcell's SNM. 24
3.7 8T bitcell architecture - Schematic. 24
3.8 10T bitcell layout (the layout is rotated 90° clockwise in this figure). 25
3.9 10T 8 bit layout. 26
3.10 10T 128 byte layout. 27
3.11 Schematic and post layout simulation comparison. 28
3.12 Read bitline for the schematic and post-layout simulations. 29

4.1 Logic gates necessary for the SRAM control circuits. 32
4.2 Optimization steps for the logic gates. 32
4.3 Logic gates transistor level schematic. 33
4.4 Logic gates layout. 36
4.5 Comparison of schematic and post-layout simulation for the inverter. 37
4.6 Comparison of schematic and post-layout simulation for the NAND2. 38
4.7 Comparison of schematic and post-layout simulation for the NAND3. 39
4.8 Comparison of schematic and post-layout simulation for the NOR2. 40
4.9 Comparison of schematic and post-layout simulation for the NOR3. 40
4.10 Demultiplexer schematic. 41
4.11 2 to 4 Demultiplexer - Layout. 42
4.12 3 to 8 Demultiplexer - Layout. 42
4.13 7 to 128 Demultiplexer - Layout (the layout is rotated 90 degrees clockwise in this figure). 43
4.14 Inverter (large) - Layout. 43
4.15 Layout of the power gating transistor. 44

4.16 Layout of the read circuits. 45
4.17 D type Flip Flop - Schematic. 46
4.18 Layout of the D type flip flop. 46
4.19 State machine for the write operation. 47
4.20 State machine implementation with D type flip flops for the write operation. . . . 47
4.21 Write state machine layout. 48
4.22 State machine for the read operation. 49
4.23 State machine implementation with D type flip flops for the read operation. . . . 49
4.24 Read state machine layout. 49

5.1 SRAM conceptual block. 52
5.2 SRAM idle power consumption varying with temperature, in the tt process corner. 52
5.3 Monte Carlo Simulation for the tt corner - Idle power consumption. 53
5.4 Monte Carlo Simulation for the tt corner - output when reading a 0. 54
5.5 Monte Carlo Simulation for the tt corner - output when reading a 1. 55
5.6 Monte Carlo Simulation for the tt corner - energy per read access when reading a 0. 55
5.7 Monte Carlo Simulation for the tt corner - energy per read access when reading a 1. 56
5.8 Variation of SRAM metrics with V_{DD}, for the tt corner (simulated). 56
5.9 SRAM memory layout. 58

A.1 Cadence Virtuoso ADE XL Outputs Setup tab for the 10T SRAM bitcell optimization. 62
A.2 Cadence Virtuoso ADE XL Results tab for the 10T SRAM bitcell optimization. . 62
A.3 Cadence Virtuoso ADE XL Results tab for the 10T SRAM bitcell optimization (2). 63
A.4 10T 128 byte layout - Word Line Detail. 63
A.5 Layout of the final SRAM memory, non annotated. 64

List of Tables

3.1 10T Bitcell Iterations. 20
3.2 10T Bitcell transistor sizing. 22
3.3 10T SRAM PVT. 23
3.4 8T Bitcell transistor sizing. 23

4.1 Transistor sizing for the logic gates. 34
4.2 PVT variation for the logic gates. 34

5.1 SRAM PVT. 51
5.2 Comparison of the SRAM in this dissertation to some works already developed . 57

Abbreviations and Acronyms

BL	Bitline
BLB	Bitline Bar
CMOS	Complementary Metal-Oxide Semiconductor
IoT	Internet of Things
MCU	Microcontroller Unit
MOSFET	Metal-Oxide Semiconductor Field-Effect Transistor
NMOS	N-channel MOSFET
PMOS	P-channel MOSFET
PVT	Process, Voltage and Temperature
RBL	Read Bitline
RFID	Radio Frequency Identification
RWL	Read Word Line
SNM	Static Noise Margin
SoC	System on a Chip
SRAM	Static Random Access Memory
ULP	Ultra Low Power
VLSI	Very Large Scale Integrated
VTC	Voltage Transfer Characteristics
WWL	Write Word Line

Chapter 1

Introduction

The Internet of Things (IoT) is an emerging technological paradigm, that focuses on distributed sensing to acquire knwoledge about the real world, and distributed computing (cloud) for its computational capabilities. However, this paradigm introduces new challenges for the design of electronics devices.

In most cases, the processing speed is relatively low (apart from wireless communications), so the IoT spends most of its lifetime in low power mode, in order to achieve long lifetimes while maintaining a small size. As such, the extended lifetime for an IoT node requires very computation-efficient circuitry and optimized idle mode operation.

Since the SRAM accounts for almost half of a chip's power consumption, a lot of work can be done towards the overall efficiency of the circuity by focusing on the volatile memory.

1.1 The Internet of Things - A General View

The Internet of Things, as a concept, first appeared in 1999 [1], and described a large-scale network of Internet connected RFID devices. Looking at the IoT in a broader perspective, it can be seen as an intersection of the Internet - with the networking and cloud capabilities; and the physical world - using distributed sensing and everyone's activities, gaining an unprecedented access to real-time fine-grain data from goods, resources, tools, environments, among others.

The IoT has been defined many times since then, and the meaning of this phrase can oftentimes include any object that is connected to the Internet - connected cars, smartphones, smart appliances. The IoT can also be defined as pervasive, non obtrusive, systematic and coordinated introduction of sense-, compute-, communication-ability and sense making of physical data in a very large number of objects on earth [2]. Using this perspective, the IoT goes beyond personal devices (like smartphones) and embeds the capabilities in everyday objects environments.

There is a a commonly agreed target of the IoT to expand the number of connected devices per person to around a thousand, reaching a never achieved before scale of trillions of connected devices worldwide [3].

1.2 Applications of IoT

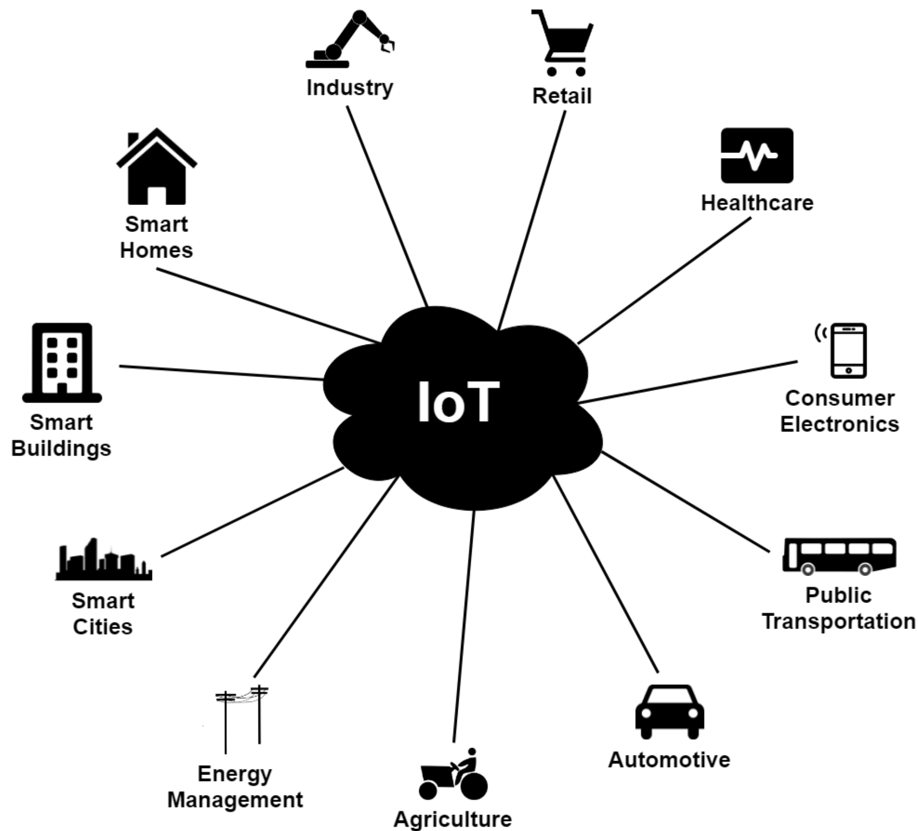


Figure 1.1: IoT Application Sectors.

The IoT has many diverse applications, and has a very fragmented application scenario [4]. It contains a wide range of applications, from agriculture and healthcare to consumer electronics and public transportation, including smart-homes through to smart-cities, as seen in figure 1.1. The applications of IoT will be summarized in the following paragraphs [2].

In agriculture, the IoT can be used to monitor availability, quality and actual usage of resources, for predictive management and storage, avoiding waste, boosting the quality of the final product, all while improving efficiency.

In the automotive sector, the IoT will enable constant monitoring of the vehicle's state, determining the usage, wear and tear of each component. This will allow for predictive maintenance, rather than scheduled or reactive maintenance, which improves the lifetime of the vehicle and its upkeep, all while lowering the maintenance costs. The distributed sensing provided will allow for better traffic control.

In public transportation, the utilization and occupancy of vehicles can be monitored, allowing for better quality of service and real-time response to short term demands.

In consumer electronics, the pervasiveness of IoT introduces new ways for the tracking of smart goods and detection of abnormal situations. It can also improve day-to-day usage, and help track the user's activity.

In healthcare, the very small size and long lifetime of IoT nodes enables an unobtrusive way to constantly monitor vital signs [5] and other health related parameters, making a deeper understanding of a patient's health evolution possible [6]. The availability of big data from a large pool of patients will offer an exceptional opportunity to study diseases and explore correlations.

Industry can also benefit from the IoT, by tracking the operating conditions and status of unfinished products, and detecting events that slow down processes [6]. It can also allow for tracking of storage location and conditions of products in warehouse.

In retail, the IoT can streamline the inventory management, cutting costs, as well as provide a better shopping experience for the customer.

Smart homes can, with IoT, manage utilities efficiently, allow better security, adjust lighting, sound and temperature (among others parameters) [6] based on individual preferences (in a predictive manner), automatically order supplies based on availability and price, and raise the efficiency of waste management while also lowering the cost.

Smart buildings can benefit from the same advantages that smart homes have [6], while also providing a better capability to respond to critical events quickly, minimizing human and material losses. Smart cities can manage resources more efficiently and be much more resilient to disasters. Energy management can be made more efficient with the IoT, optimizing the energy usage across the urban area, leveraging the production of renewable energy.

The applications of IoT across all sectors and fields can be summarized on the ability to collect unprecedented large amounts of data in an unobtrusive manner, using the capabilities of cloud and distributed computing to take conclusions from the data collected, as exemplified in figure 1.2.

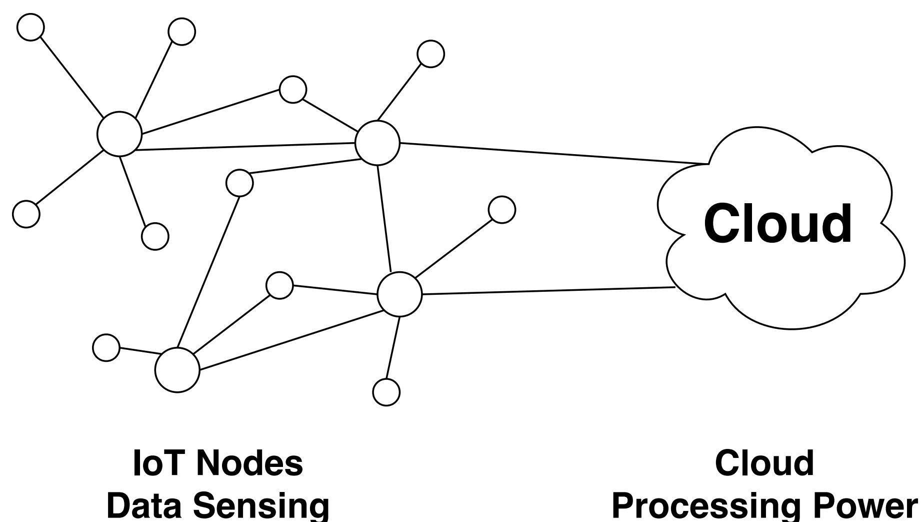


Figure 1.2: IoT nodes connected to the cloud.

1.3 Motivation

This work focus on the development of an SRAM in CMOS technology. The technology chosen is CMOS mainly due to its ubiquity in VLSI and to its power characteristics. The processes ranging

from 180nm to 130nm offer the best leakage currents in CMOS - going to a smaller process would not bring any benefits, since a larger transistor length means lower leakage power.

Currently, the industry focus on lowering the power consumption while increasing the maximum clock frequency of the memory blocks. However, IoT peripherals do not have the need for high performance volatile memories. The sensors/actuators are, in most cases, only activated sporadically, with high inactivity periods between activations. The refresh rate of these peripherals can range, usually, from once every second to once every few minutes.

This use case would benefit from an ultra-low power, low performance volatile memory, which would be optimized for the occasional update of data the IoT peripherals need.

The goal - a 128 byte SRAM working at 128 Hz - perfectly suits the memory needs for an IoT peripheral, while opening the doors to huge improvements on power consumption. This work is expected to provide an SRAM with an even lower power consumption than the current industry.

1.4 Problem Statement

The main objective is the development of an SRAM in CMOS technology, operating in the subthreshold regime. The target scenario is an IoT system designed for ultra-low-power consumption in which the SRAM will serve as the memory device for the IoT peripherals data. The work will consist of a complete CMOS realization, from schematic to layout and post-layout simulation.

The SRAM to develop has the following objectives:

- Size of **128 bytes**
- Clock frequency of **128 Hz**
- Supply voltage ranging from **100 mV to 300 mV**
- Power consumption **under 10 nW** in idle

1.5 Proposed Solution

The proposed solution is to develop a schematic and physical layout of the different components of an SRAM memory, focusing on lowering the power consumption while in idle mode, for use as a memory device for sensing peripherals in IoT, as said in the problem statement. The SRAM memory will use a 10T single ended bitcell to achieve lower power consumption and better robustness, while allowing the SRAM memory to be dual ported (which means that it will be possible to write to and read from the memory at the same time). The steps are as follows:

- Research and study the different SRAM bitcells and implementations;
- Optimize the bitcell, in schematic, for power consumption while not overlooking functionality and robustness;

- Design and optimize all the components needed to control the SRAM
- Layout deployment

1.6 Document Outline

This document is composed by the following chapters, besides the Introduction:

- **Chapter 2 - Literature Review**
  IoT node architecture and requirements are analyzed. SRAM architectures are researched and analyzed, as well as the most common SRAM metrics. The main factors surrounding the subthreshold operation regime are researched, described and related to this work.
- **Chapter 3 - SRAM Architecture**
  The proposed SRAM architecture is described. The bitcell choice and optimization are explained, and the layout process is described.
- **Chapter 4 - Control Circuits**
  The control circuitry needed for the SRAM memory is described and optimized, and the layout is explained.
- **Chapter 5 - Integration**
  The SRAM block and the control circuits are connected, and the layout parasitics components are extracted. The final SRAM memory is simulated and metrics are shown.
- **Chapter 6 - Conclusion**
  Some final considerations are done and the future work is presented.

Chapter 2

Literature Review

The literature review will first focus on the Internet of Things, covering the general architecture of an IoT node and the several requirements that the IoT nodes have. It will be followed by a review on the SRAM designs currently used, with a comparison between the most common (6T, 8T and 10T).

Finally, it will cover the subthreshold region of CMOS, as well as the different ways that CMOS consumes power, and the disadvantages brought by using the subthreshold operating region.

2.1 IoT - Internet of Things

The common IoT node architecture, seen in figure 2.1, includes the following components [2]:

- One or several microprocessors (typically **MCU**s)
- A **communication unit**
- One or several **sensors** and/or **actuators**
- A **battery** and/or **energy harvester**. In the case of a energy harvester:
  - Energy storage (capacitor or battery)
  - A power management unit

The MCU is the heart of the IoT node. It is responsible for the analysis of the data sensed and/or the operation of the actuator(s), and the preparation of the data to be sent over the communication module.

The sensors provide the IoT node with knowledge of the real world, using parameters such as temperature, light, vibration and sound. The actuators allow the IoT node to influence the real world, according to data sensed or received.

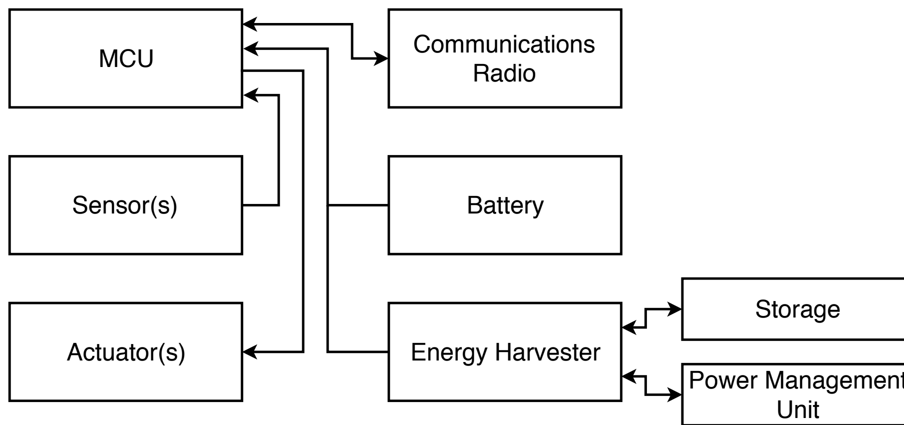


Figure 2.1: Common elements in an IoT node architecture.
The energy harvester and battery can or not coexist, depending on the system.

The communications module/radio allows the IoT node to exchange data with other nodes (e.g. for calibration and distributed sensing) and to send/receive data from the cloud. The battery and/or energy harvester provide the IoT node with the power necessary for its functions. The requirements of IoT nodes can be summarized in four categories: physical constraints, user constraints, interaction with the real world and on-board capabilities.

Physical Constraints

The physical constraints are imposed by size limitations and the need to untether the IoT node and avoid any physical maintenance, which includes the swapping or recharging of batteries. As such, the IoT nodes need to rely either on a battery, a (or several) energy harvester(s) or a combination of both [7]. As for the form factor, the node needs to be small enough to make its deployment non intrusive, with a typical volume of cubic millimeters to hundreds of cubic millimeters.

Considering the target lifetime of IoT nodes in the order of several years to decades, it is certain that improvements need to be made on several fields. Some are the energy storage (mainly, energy density of batteries); energy harvesting (increasing the energy production while decreasing the size); power and energy efficiency of the circuits.

With the slow development of battery technologies when compared to CMOS, the main focus on reducing the size of a node is to reduce the power consumption, allowing for a smaller battery and/or energy harvester, thus reducing the overall size of the node.

User Constraints

The user constraints in IoT nodes are focused on cost and security. Regarding cost, the target for consumer applications is around 1$/€ per node [2]. This limits the the die cost, and puts pressure on the semiconductor industry to provide lower prices, which can be addressed by large sales volume. Achieving such a large sales volume is difficult, due to the IoT space being highly fragmented. That is possible, however, by diluting the cost by using a platform that favors reusing.

Security is an important request from the user, because at such scale of deployment, it can provide lots of backdoors to attackers. Software solutions to counteract cyber-attacks are not applicable, due to the very low computing power and power budget of the nodes.

Interaction with the Real World

The interaction with the real world is done through sensing. However, due to the power constraints on an IoT node, that has to be done infrequently. In order to meet the power budget, there has to be an aggressive duty cycling of the node, where it spends most of its time in sleep mode (minimizing energy consumption), waking up only for a small fraction of the time to perform sensing, computational and communication tasks. An IoT node is organized into an always-on subsystem, that manages the wake-up cycle and the information between tasks (ALWON), and a duty-cycled subsystem (DCYC) that periodically performs the main task(s) [7]. The average power consumption of an IoT node can be written as:

$$P_{Avg} = P_{ALWON} + \frac{E_{DCYC}}{T_{wkup}} \tag{2.1}$$

On-board Capabilities

An IoT node needs to have computational, sensing and communication capabilities. In the active task, a node needs to sense the real world and either communicate the data or have more computational power in order to send less data (send processed data instead of raw data) [2].

One way to reduce the average power consumption of an IoT node through the computational part is to reduce the volatile memory power consumption.

IoT nodes are subject to extremely aggressive duty-cycling, as said above. The memories associated with the IoT peripherals can have a small size (hundreds of bytes) and a really low performance (hundreds to thousands of cycles per second), as most sensors do not require higher parameters. This makes them prime candidates for power reduction.

2.2 SRAM

The volatile memory mostly used in small embedded devices and SoC is SRAM. This happens due to SRAM's lower access times and the lack of need to refresh the bitcells' contents (as happens with DRAM). This gives the SRAM a simpler control circuit, and a nearly limitless retention time (as long as the SRAM is powered).

This section summarizes the different bitcell designs that are currently most used in low power applications.

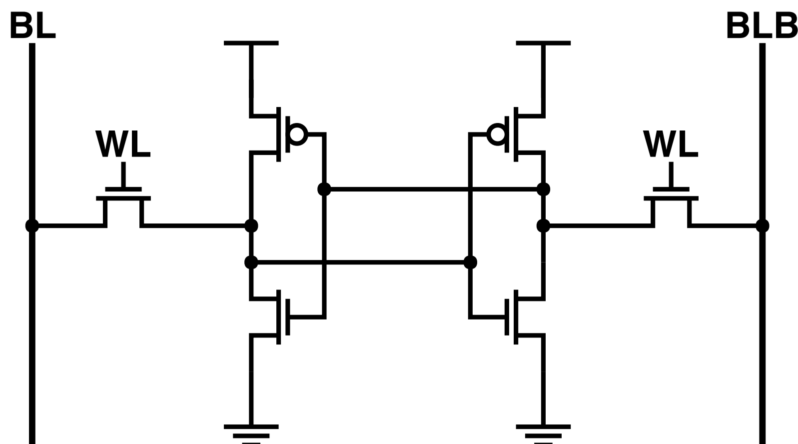


Figure 2.2: Standard 6T SRAM bitcell schematic.

2.2.1 6T SRAM

The standard memory bitcell currently used in SRAM designs consists of a cross-coupled inverter pair with two access transistors, as seen in the schematic in figure 2.2. This is known as a 6T bitcell, due to having a total of six transistors per bitcell [8, 9, 10, 11].

The bitcell is written by setting the bitline (BL) to either V_{DD} or ground, and bitline complement (BR or BLB) to the complement voltage (ground or V_{DD}, respectively); then, the access transistors are closed. The bitcell is read by setting both the bitlines (BL and BR/BLB) to V_{DD}, and asserting the wordline (WL). This causes a differential current on the bitlines, which can be sensed to determine the value of the stored bit.

Due to inter-die and intra-die process variations, supply voltage scaling is limited to a minimum operation voltage (known as V_{MIN}). This limit is set by several memory failures, that can be read failure, write failure, retention failure and access time failure. There are several techniques to minimize these failures.

A read failure is caused by an increase in the access transistor drive strength when compared to the cross-coupled inverter pair. Read assist techniques include lowering the V_{SS} and/or raising the V_{CC} of the cross-coupled inverter pair. These techniques include as well wordline underdrive (WLUD) or raising the supply voltage of the entire bitcell compared to the peripheral circuitry.

On the contrary, a write failure is caused by an increase in the cross-coupled inverter pair drive strength when compared to the access transistor. Write assist techniques are, for example, raising the V_{SS} and/or lowering the V_{CC} of the cross-coupled inverter pair. These techniques also include wordline boosting and negative bitline approaches [11].

2.2.2 8T SRAM

The main objective of the 8T SRAM architecture [12, 13, 14, 15, 16, 17, 18, 19] is to decouple the read and write operations.

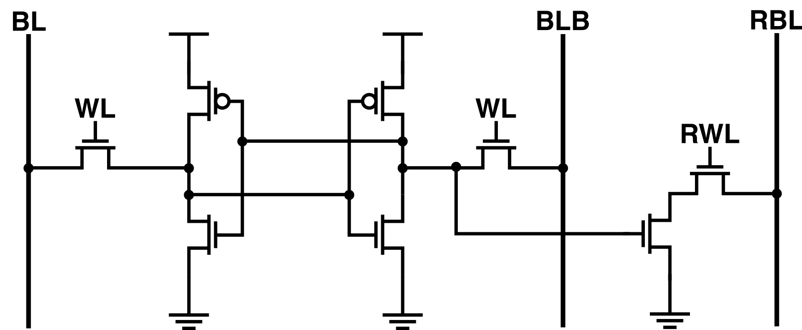


Figure 2.3: Standard 8T SRAM bitcell schematic.

The most common 8T bitcell design consists of a standard 6T bitcell with a dedicated read port (which consists of two transistors), as seen on the schematic of figure 2.3, although there are different designs with its advantages and drawbacks [20]. This separation of the read and write operation means that the bitcell can be optimized for both read and write.

2.2.3 10T SRAM

There are several 10T (10 transistors) bitcell architectures proposed, that can be single ended or differential [8, 21, 22, 23, 24, 25]. Single ended 10T bitcells, which can be seen on the schematic of figure 2.4, are similar to 8T bitcells, with the difference being on the read port configuration. Differential 10T bitcells have only two differential bit lines (BL and BLB/BR), and are more similar to 6T bitcells, though they have more transistors to perform the read/write functions.

Other designs

Other SRAM designs include 7T [26], 9T [27, 28, 18, 29], 11T [30], 12T [31] and latch based [32, 33, 34, 35].

2.2.4 Main SRAM Designs Comparison

The 6T bitcell design offers lower area than the other designs. However, because the design requirements for read stability and write-ability are conflicting, the V_{MIN} is relatively high when

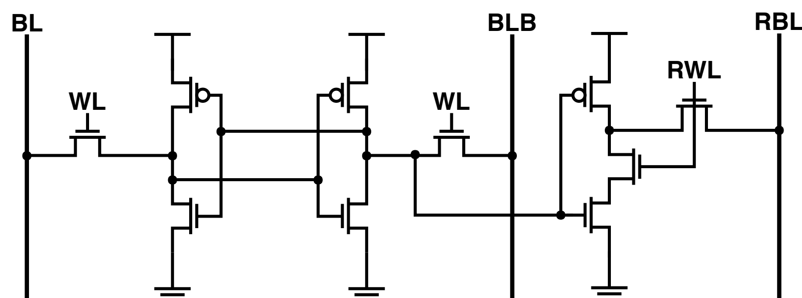


Figure 2.4: Single Ended 10T SRAM bitcell schematic.

compared to the 8T or 10T designs.

The 8T bitcell design adds a dedicated read port to tackle the conflicting design requirements. This offers faster read and write, dual-port capability and a lower V_{MIN} compared to the 6T design.

The 10T single ended bitcell improves on the 8T bitcell, providing a read port with lower leakage current (because of the stack of transistors). This allows for lower V_{MIN} than the 8T design, at the cost of extra area.

2.2.5 SRAM Metrics

The Static Noise Margin (SNM) is an SRAM metric that translates to the maximum DC noise that can be present in a node of the bitcell without the bitcell changing its correct value. The SNM can be measured for both the read and write operations, as well as hold (while the bitcell is not being written to or read from), and it reflects the bitcell's robustness.

Write SNM

The write SNM is calculated by sweeping the input of the bitcell's inverter, while the access transistor is closed. Doing this for both BL = 0 and BL = V_{DD}, the two VTC (Voltage Transfer Characteristics) waves can be obtained. The butterfly plot of the VTC obtained can be used to graphically calculate the SNM, by measuring the size of the largest square that fits within the butterfly plot [36].

Read SNM

In bitcells with a single ended read port, the read SNM can be calculated with the VTC of the read port. When mirroring the VTC, the butterfly plot is obtained, and the SNM can be calculated by measuring the side of the largest square that can be embedded in the butterfly plot [36].

Hold SNM

The hold SNM can be calculated using the butterfly plot resulting from mirroring the VTC of the inverters in the core of the bitcell, obtained by sweeping the input of the inverter [36]. The side of the largest square that can be embedded into the butterfly plot is the hold SNM [36].

2.3 Subthreshold Circuits

CMOS is the main technology used in VLSI, due to its power characteristics. When changing state, it consumes power, but not when it is in a steady state [37]. Traditionally, CMOS operates with a supply voltage V_{DD} well above its threshold voltage V_T. This means that the transistor operates in the region of strong channel inversion [7].

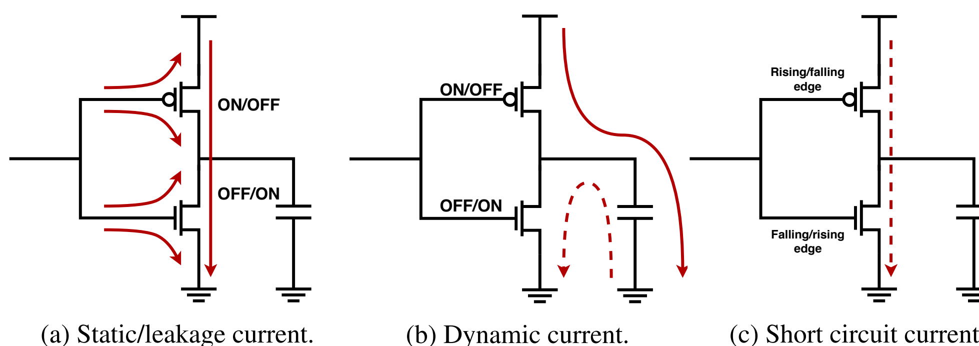
The main path towards reducing power consumption in CMOS is through voltage scaling, which is the reduction of the supply voltage. When the supply voltage is lower than the threshold

voltage, the transistors operate in subthreshold (or weak inversion) region. In this region, the currents are extremely low when compared to the strong inversion region (several orders of magnitude lower). When operating in the strong inversion region, the weak inversion region is considered as leak current, or that the transistor is turned off (since $V_{GS} < V_T$) [7].

Since the weak inversion region provides lower currents and at a lower supply voltage than the strong inversion region, it is a very effective operating region when dealing with ultra low power devices.

2.3.1 CMOS Power Consumption

The power consumption in CMOS technologies can be divided into three categories, seen in figure 2.5: **static**, **dynamic** and **short circuit** [37, 7].



(a) Static/leakage current. (b) Dynamic current. (c) Short circuit current.

Figure 2.5: Power consumption on CMOS technologies.

Power consumption can be given by:

$$P = \frac{1}{T}\int_0^T I \cdot V_{DD} dt \tag{2.2}$$

Static Power

The static power arises from four different factors [37] [7]: gate-oxide leakage current, reverse-biased diode leakage current, gate induced drain leakage current and subthreshold leakage current.

- **Gate-oxide leakage current:** is a current that flows through the oxide in the gate, when a voltage is applied (the transistor is ON). This current is stronger the thinner the oxide layer in the gate. As voltage decreases, this current decreases faster than the subthreshold current.
- **Reverse-biased diode leakage current:** flows from drain and source to the bulk. It is usually negligible, but becomes more pronounced when there is a high potential difference between the drain/source and the bulk (high V_{DD} processes).
- **Gate induced drain leakage current:** flows from the drain interface with the gate oxide to the bulk. It happens due to the high eletrical field near that interface. In the weak inversion region, this current is mostly negligible.

- **Subthreshold leakage current:** flows from the source towards the drain, when $V_{\mathrm{GS}} < V_{\mathrm{T}}$ (transistor is supposed to be OFF). The current varies exponentially with V_{GS}. When the voltage difference between the gate and source is zero, the current (known as I_{OFF}) is in the order of tens of nA.

Dynamic Power

Dynamic power comes from the output changing state. When it changes to a logic level "1", the load capacitance is charged (usually to V_{DD}) through the PMOS transistor(s). When the state is changed to a logic "0" (usually ground), the load capacitance is discharged, with the NMOS transistor(s) dissipating the charge. The dynamic power can be described as [37]:

$$P_{Dyn} = \alpha_{0\rightarrow 1} \cdot f_{clk} \cdot C_L \cdot V_{DD}^2 \tag{2.3}$$

In 2.3, $\alpha_{0\rightarrow 1}$ stands for the activity factor ($0 < \alpha_{0\rightarrow 1} < 1$). C_L is the average load capacitance. As seen in 2.3, the dynamic power consumption is proportional to the clock frequency and the supply voltage squared, while being unaffected by the rise and fall times of the signals.

Short Circuit Power

Short circuit power dissipation occurs when both PMOS and NMOS (or pull-up and pull-down networks) are partially ON during a switch, creating a direct path between V_{DD} and ground. This power dissipation is more significant when the input rise/fall time is much larger than the output rise/fall time, causing the short-circuit path to be active for a larger amount of time. However, short circuit power dissipation is a small part of the total power dissipation in a circuit [38].

2.3.2 MOS EKV Model

An analytical MOS model dedicated to analyzing low-current, low-voltage analog circuits was proposed in [39], called EKV model. The compact equation is described in [40] as:

$$I_D = I_{D0} \cdot e^{\frac{V_G}{nU_T}} \cdot (e^{-\frac{V_S}{U_T}} - e^{-\frac{V_D}{U_T}}) \tag{2.4}$$

where:

$$I_{D0} = I_{spec} \cdot e^{-\frac{V_{T0}}{nU_T}} \tag{2.5}$$

is the residual drain current when $V_G = V_S = 0$, and I_{spec} is the specific current of the transistor, given as:

$$I_{spec} = 2n\beta V_T^2 \tag{2.6}$$

and β is the transfer parameter of the transistor, given as:

$$\beta = \mu C_{ox} \frac{W}{L} \tag{2.7}$$

In these equations (2.4, 2.5, 2.6 and 2.7), the parameters are described as:

- V_T is the threshold voltage
- n is the subthreshold slope factor
- U_T is the thermodynamic voltage, defined as:

$$U_T = \frac{kT}{q} \tag{2.8}$$

  where q is the elementary charge and k is the Boltzmann constant. For a temperature T = 300 K (27°C), U_T is around 25.8 mV.

- C_{ox} is the gate oxide capacitance
- μ is the carrier mobility

Looking at equation 2.4, it can be seen that the drain current grows exponentially with the gate, source and drain voltage, and with the square of the threshold voltage. Therefore, lowering the supply voltage can greatly decrease the power consumption of a device.

Since the subthreshold EKV model shows exponential characteristics, it shows vulnerabilities to Process, Voltage and Temperature (PVT) variations [40].

Process

Process variations usually results from random dopant fluctuations and slight variations in the parameters of physical characteristics, and causes transistors with equal layout structures to not have the same properties [7]. This affects mainly the model parameters V_T, n and β.

Voltage

Voltage differences in a ULP system are caused by variations in the external power supply. In battery powered devices, the voltage supply is relatively constant; for batteryless systems, the voltage supply suffers from much larger deviations, that are handled by voltage regulation circuits [7].

Temperature

As seen in equations (2.4, 2.5 and 2.8), the absolute temperature has an exponential effect on the drain current. It is shown in [41] that subthreshold I_{ON} current increases exponentially with the temperature, while $\frac{I_{ON}}{I_{OFF}}$ ratio decreases due to an increase in leakage currents. As V_{GS} decreases, temperature variation increases drain current more rapidly [42].

2.3.3 Disadvantages of Subthreshold

Although operation of CMOS technologies in the subthreshold region leads to lower power consumption, it has one major drawback: maximum achievable frequency. With the transistors operating in the weak inversion region, the drain currents will be much lower when compared to the strong inversion operation.

In the weak inversion operating region, the propagation delay for an inverter is defined as:

$$t_d = \frac{KC_G V_{DD}}{I_{Dsub}} \tag{2.9}$$

where C_G is the output capacitance of the circuit and K is a delay fitting parameter. As it can be seen in 2.9, as the drain current decreases the propagation delay time increases rapidly. As a consequence of the increased propagation delay, the operational frequency will decrease [7].

Chapter 3

SRAM Architecture

An SRAM memory can be divided into two groups of components: the first group is the SRAM block (group of bitcells); the second group is the SRAM control circuitry. This chapter will focus on the SRAM architecture, detailing the bitcell choice and all the optimization work realized; the bitcell layout is also covered, as well as the layout of the SRAM block, with post-layout simulations being compared to schematic simulations.

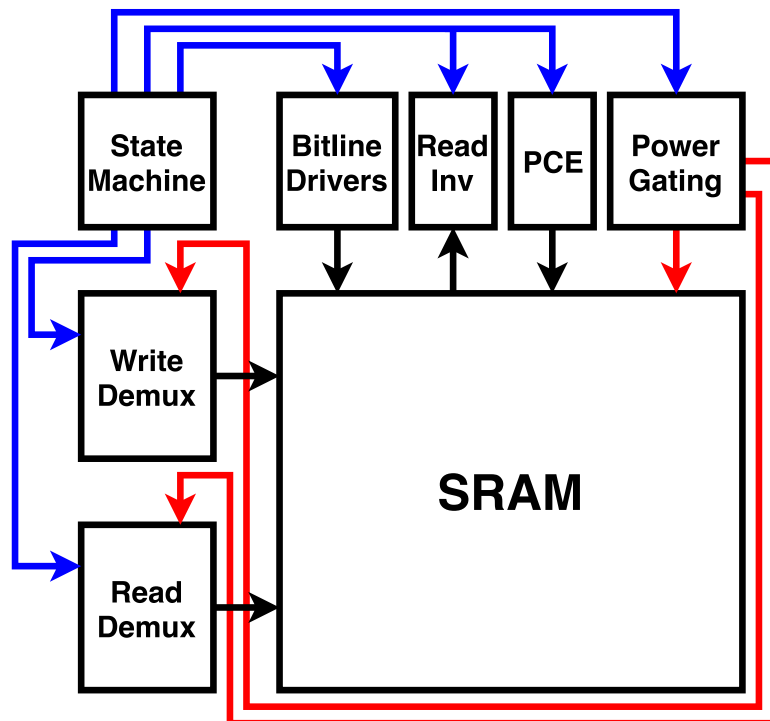


Figure 3.1: SRAM block diagram.

In an SRAM, there are several major components, that can be seen in figure 3.1. These are:

- **SRAM block**, composed of the SRAM bitcells. In this work, it is composed of 128 words of 8 bits each. This means that the block will have 1024 bits in total.

- **Address decoder**, which is responsible of converting the 7 bit address into a line selector, capable of selecting one of the 128 words available in the SRAM block.
- **Bitline Drivers**, that have to drive the bitlines to values of V_{DD} or 0, according to the data value going to be written.
- **Read Circuits**, to aid in the read operation. These are usually composed, in the case of single ended decoupled reading, of a precharger - that drives the bitline to V_{DD} before the read; and a sensor - aiding in the time and accuracy of the read operation.
- **Control block**, that coordinates the steps necessary to write or read a value to/from the desired address.

In figure 3.1, the black arrows represent the data flow paths along the SRAM; the red arrows show the power supply lines that can be power gated; the blue arrows stand for the control signals that are sent from the finite state machine to the other control circuits.

The SRAM designed has the need for other blocks apart from the standardly used [2, 36]. These include:

- **Power gating** circuits, that shut off components that are not needed while not writing nor reading. This is used with the intent of lowering the idle power consumption.
- **Separate Read and Write Address Decoders**
- **Read circuits:** composed of pre charger circuits and read inverters (that aid in reading the stored value)

A dual ported memory can be both written to and read from at the same time. In the use case provided for this SRAM, being dual ported is a major benefit, as it means that the sensor can write the data to the SRAM while the main microcontroller is reading data values from the memory; however, this also generates the need for two address decoders.

As explained above, the bitcell is the atomic component of an SRAM memory. It is the component that stores the logic values, and allows for later retrieval. The following sections will detail the work done on the SRAM, mainly focusing on the bitcell. First, the bitcell choice is explained, and then the optimization work is detailed and justified.

3.1 10T Bitcell

The chosen 10T bitcell architecture is represented in figure 3.2, and it was chosen for a variety of reasons, with the most important being the decoupled read port. The read port consists of 4 transistors (one PMOS and three NMOS), and functions as a tri-state inverter. When the read word line (RWL) is at 0, there are (at least) 2 off transistors , which reduces the leakage current on the read bitline (RBL).

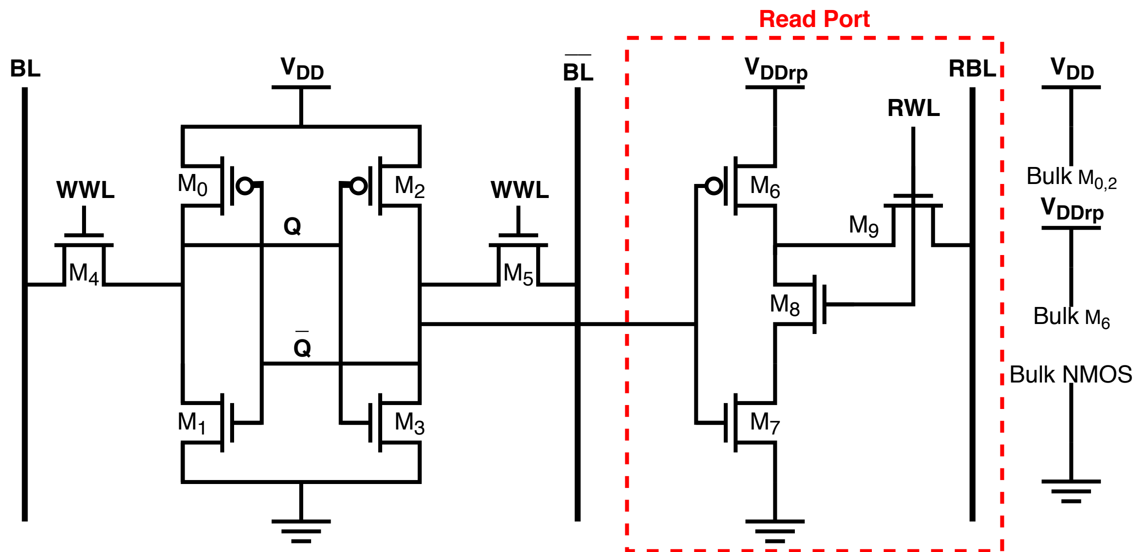


Figure 3.2: Chosen 10T bitcell architecture - Schematic.

The other possible configuration of a 10T bitcell differs only in the read port: it is an inverter followed by a transmission gate. The latter was not chosen due to the presence of transmission gates, which proved challenging to implement at subthreshold voltage supply levels, in work realized previous to this dissertation.

When analyzing the chosen 10T bitcell, it presents two main advantages: the first is that the 10T bitcell has lower leakage current form the RBL through the read port than the 8T bitcell [43]; the second advantage is the ability to power gate the read port of the 10T bitcell, which is not present in the 8T [43]. Power gating is a technique in which the V_{DD} supply is cut to parts of the circuit that are not in use at the moment, and can significantly reduce the idle power consumption of the bitcell.

The bitcell architecture has a huge impact on its power consumption and performance, but it alone does not guarantee the best possible power consumption while retaining the performance needed. Prior to the optimization (i.e. all the transistor have minimum length and width) the idle power consumption of a single bitcell is 4,925 pW. This means that an SRAM block that has 128 words of 8 bits each word, the final block would have an idle power consumption of around 5 nW. This leaves a narrow margin of available power for all of the control circuitry in order to comply with the 10 nW idle power given in this thesis objectives, and proves the necessity for the optimization work.

3.1.1 Optimization

All optimization was done using Cadence Virtuoso ADE XL, mainly utilizing the **Corners** tab and the auto step function to perform parametric simulations. In the **Outputs Setup** tab, the **Specs** were used together with Cadence Virtuoso Calculator expressions to facilitate the optimization, by automatizing the evaluation of the results to a certain degree. The calculator expressions were

Table 3.1: 10T Bitcell Iterations.

Attempt	Transistor Type	Write Pass Transistor	Reverse Body Bias	VDD
1	LVT	NMOS	No	300 mV
2	HVT	NMOS	No	300 mV
3	HVT	PMOS	No	300 mV
4	HVT	PMOS	Yes (M0 and M2)	300 mV
5	LVT	NMOS	Yes	200 mV
6	LVT	NMOS	No	200 mV

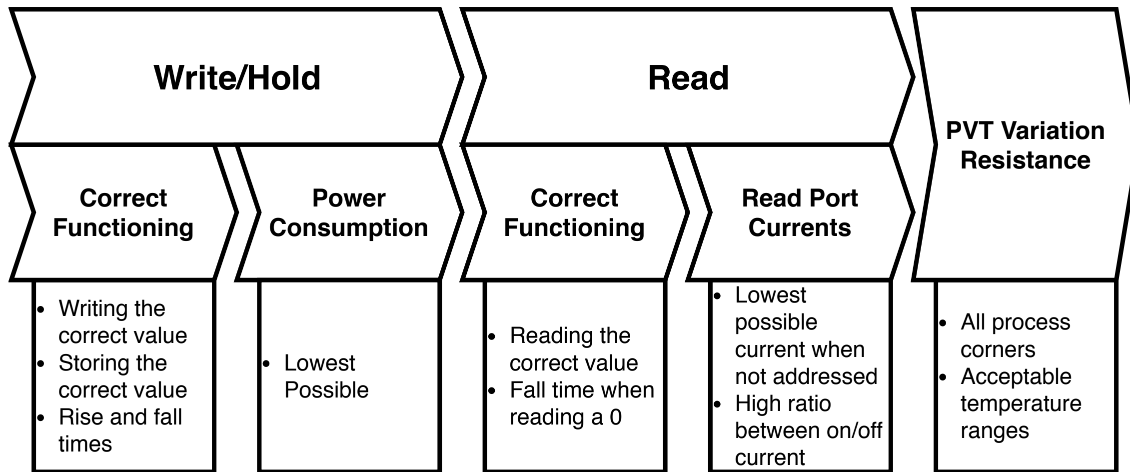


Figure 3.3: SRAM optimization steps and order.

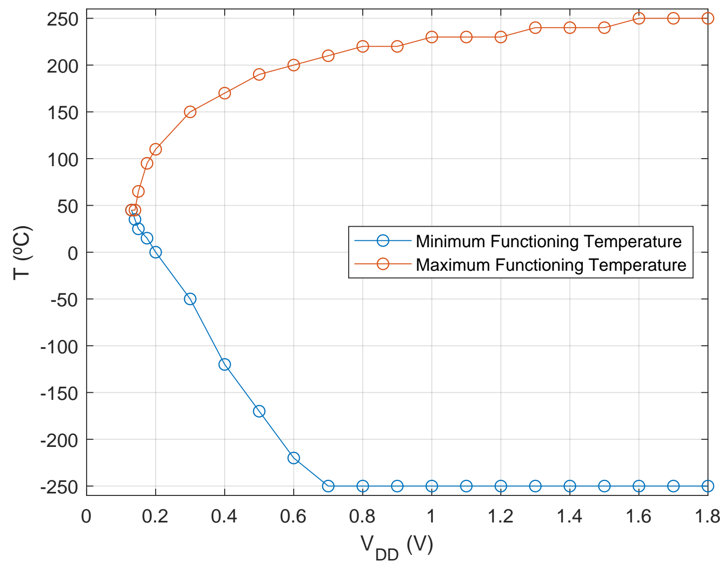


Figure 3.4: Variation of the maximum and minimum functioning temperature, while varying the V_{DD} in the tt process corner, for the 10T bitcell in schematic simulation.

used to determine average and final voltage levels stored in the bitcell; voltage levels of the read value; idle power consumption and read port currents - the Cadence Virtuoso ADE XL Outputs Setup tab can be seen in appendix.

Thanks to the **Specs** definition, the evaluation of the results of each parametric simulation is made easier. As it can be seen in appendix, the **Results** tab color codes the results as green when the result is within the specified; and as red when it is not. Additionally, when the results is within 10% of the spec, it is color coded as yellow.

The optimization was done in two parts: first for the write/retention part (corresponds to transistors M0-M5), and then for the read port (transistors M6-M9). The optimization steps are summarized in figure 3.3.

As the core of the bitcell consists in two inverters connected in a positive feedback loop and two access transistors (M4 and M5), the transistors in the two inverters and the two access transistors were assumed to be equal. Due to computational constraints, subsequent parametric analysis were run on the bitcell changing the parameters: L0, W0, L1, W1, L4 and W4 (length and width of the transistors M0, M1 and M4). The aim of this part of the optimization was to get the lowest possible idle power consumption, while guaranteeing correct functioning of the bitcell (retention of the correct Q/QB value and acceptable rise and fall times) and resistance to PVT variation. Each parametric analysis would provide a shorter range for evaluation for each parameter. The final value would require 6-10 sweeps.

After the core of the bitcell was optimized, the focus shifted to the read port: the values for width and length of each of the transistors were swept, with a decreasing range after each iteration. The main objectives when optimizing the read port were guaranteeing correct functioning (the value read was appropriate and the fall time was within the target); and minimizing the current through the read port when not reading, while also aiming for a high ratio between on and off current. As with the core of the bitcell, each parametric analysis would provide a shorter range of evaluation for each parameter, with the final value requiring between 8 to 12 sweeps.

Before reaching the final version, there were several attempts at reaching the lowest possible power consumption, which are summarized in table 3.1:

In the **first attempt**, the bitcell used all LVT transistors at a VDD of 300 mV.

The **second attempt** used all HVT transistors, with a supply voltage of 300 mV. Although this lead to lower idle power consumption, the performance and the PVT variation were worse.

The **third attempt**, the pass transistors (M4 and M5) were replaced with PMOS transistors, in order to increase the write performance. In this attempt, all transistors were HVT as well. The HVT PMOS offers greater ON current than the HVT NMOS.

The **fourth attempt** used reverse body bias in the PMOS transistors. In this attempt, all transistors were HVT, once again. Reverse body bias is a technique in which the body of the transistor is connected to a higher voltage node than the source (in the case of PMOS). This leads to lower leakage currents when the transistors switched off. However, this attempt was unsuccessful, due the voltage values stored in Q and QB going past the supply voltage.

Table 3.2: 10T Bitcell transistor sizing.

Transistor	0	1	2	3	4	5	6	7	8	9
Width	1.5 um	380 nm	1.5 um	380 nm	450 nm	450 nm	240 nm	240 nm	400 nm	240 nm
Length	50 um	50 um	50 um	50 um	450 nm	450 nm	180 nm	180 nm	500 nm	500 nm

In the **fifth attempt**, all transistors used were LVT. In this attempt, the same solution proposed in the fourth attempt was used, and switching from HVT to LVT lead to the same problems.

The **sixth and final attempt** used LVT transistors and power gating on the read port , while giving up on reverse body biasing. At the same time, the V_{DD} was lowered, which was possible due to reverting back from HVT to LVT transistors. (Power gating is a technique in which parts of the circuit are switched off from the power supply when they are not needed). With a supply voltage of 200 mV and power gating on the read port, the best power consumption values were reached, while maintaining good resistance to PVT variation.

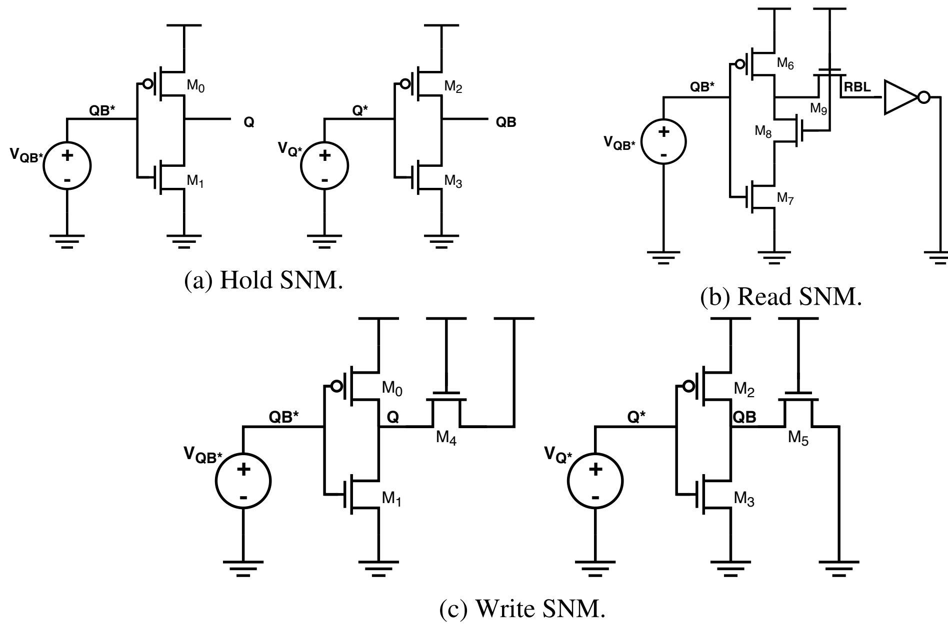
After the optimization process, the resulting 10T SRAM bitcell transistor sizing can be seen in table 3.2.

In the bitcell part that is always on, the transistors that make the two inverters have the maximum length of the process - 50 µm. The bigger the length of the transistor, the lower the subthreshold current. On the other hand, the transistors on the read port (M6 - M9) are power gated in idle. This means that its subthreshold leakage current is already negligible, and the transistors can be optimized almost only for read performance. This explains the huge difference in size among the transistors in this bitcell.

The idle power consumption for the 10T bitcell is 141.15 fW. This representes an improvement of almost 35 times when compared to the non optimized bitcell. This value is measured when the bitcell is in idle: not writing and the read port is power gated (its power supply is at 0 V). The power consumption can be lowered even more by the control circuits, mainly the bitlines drivers, by leaving the bitlines (BL and BLB) floating during idle. The operational temperature of the bitcell in the different process variations can be seen in table 3.3, while the maximum and minimum temperature variation for the tt process corner can be seen in figure 3.4.

3.1.2 Static Noise Margin

The Static Noise Margin (SNM) is an important metric for SRAM. It reflects the maximum DC noise that can be present before the value that the cell is holding, or that is being written to or read from the bitcell "flips" (goes from a 0 to a 1, or from a 1 to a 0). The SNM shows the bitcells robustness to noise. For the 10T bitcell that was optimized, the hold SNM is 52.8 mV, and can be seen in figure 3.6a; the write SNM is 100.9 mV, and is seen in figure 3.6b; finally, the read SNM is 58.7mV, and can be seen in figure 3.6c.



(a) Hold SNM.

(b) Read SNM.

(c) Write SNM.

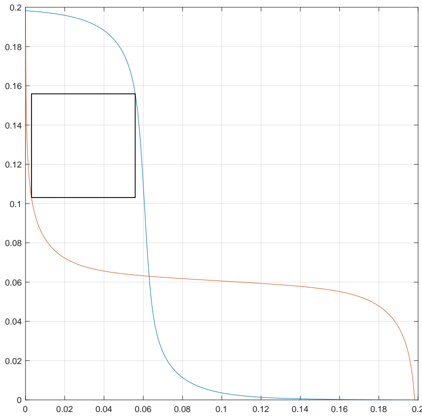
Figure 3.5: Schematic for the determination of the bitcell's SNM.

Table 3.3: 10T SRAM PVT.

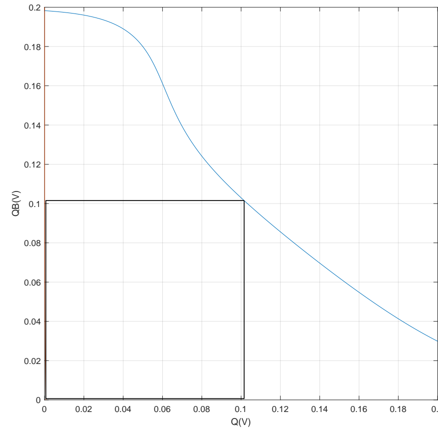
Process Variation	tt	ff	ss	snfp	fnsp
Minimum Temperature (^{o}C)	0	-30	-10	-10	0
Maximum Temperature (^{o}C)	110	110	110	130	80

Table 3.4: 8T Bitcell transistor sizing.

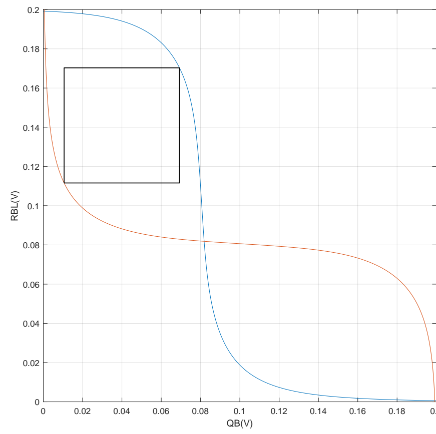
Transistor	0	1	2	3	4	5	6	7
Width	1.5 um	400 nm	1.5 um	400 nm	450 nm	450 nm	2 um	1.2 um
Length	50 um	50 um	50 um	50 um	450 nm	450 nm	50 um	50 um



(a) Hold SNM for the 10T bitcell.



(b) Write SNM for the 10T bitcell.



(c) Read SNM for the 10T bitcell.

Figure 3.6: Butterfly plots for the calculation of the bitcell's SNM.

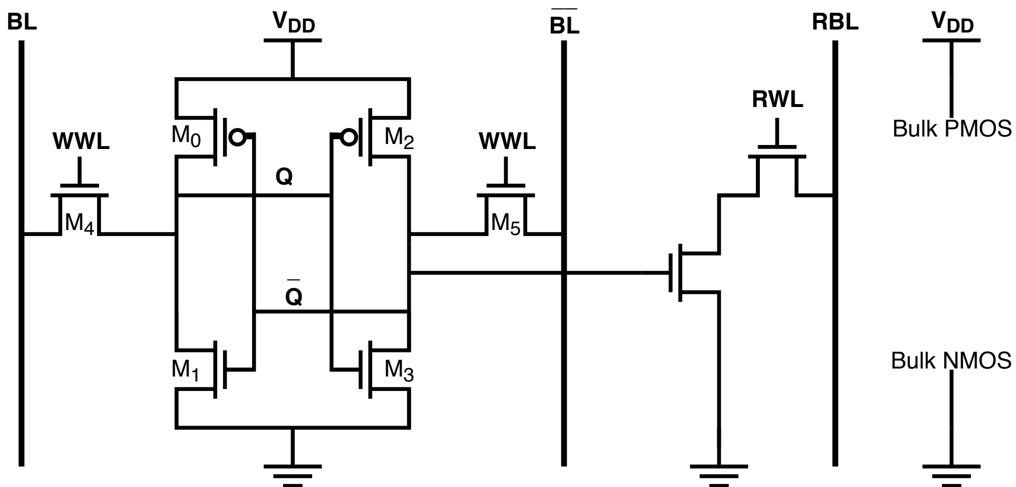


Figure 3.7: 8T bitcell architecture - Schematic.

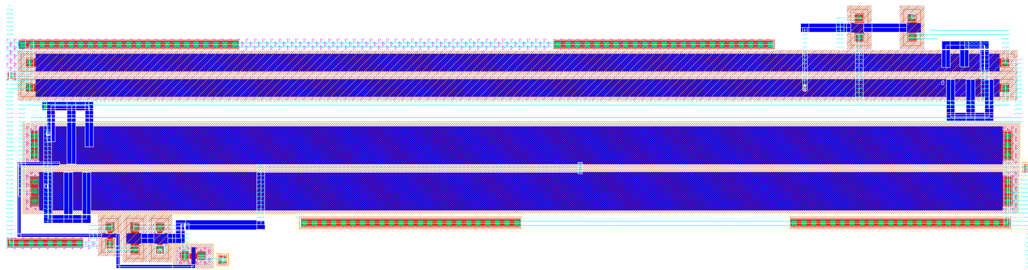


Figure 3.8: 10T bitcell layout (the layout is rotated 90º clockwise in this figure).

3.2 8T Bitcell

An 8T bitcell (which architecture can be seen in figure 3.7) was also experimented with the hope of reaching a bitcell with similar leakage currents (and, subsequently, idle power), while occupying lower area.

The 8T bitcell was optimized with the same criteria as the 10T bitcell, seen in figure 3.3. After the optimization was finished, the 8T bitcell transistors had the sizing that can be seen in table 3.4. Has it can be seen in table 3.4, the two transistors of the read port have the maximum length possible in this technology. The idle power consumption for the 8T bitcell is 288 fW, when simulated in schematic at 25ºC in a tt corner. This value was measured under the same circumstances as the 10T bitcell, namely the switching of the bitlines right after the access transistors opened.

The 10T has smaller transistors, and provides a lower idle power consumption. Additionally, the read port of the 10T has lower leakage current from the read bit line, and the bitcell provides an overall better resistance to PVT variation. When compared to the 10T bitcell, this bitcell presented higher power consumption, higher overall gate area and a less robust read port, while being less resistant to PVT variation. For these reasons, it was not implemented in the final SRAM.

3.3 10T SRAM Layout

In the physical layout, a modular approach was followed: first, a single bitcell was implemented; then, 8 bitcells were put together to form a single byte (or word, in this SRAM); finally, 128 bytes were joined to form the final SRAM block.

In order to facilitate the layout process, a few measures were taken while doing the layout for the 10T bitcell, that can be seen in figure 3.8: the power supply line (V_{DD}) was placed to the top and rigth corner; the ground line was placed to the bottom and left corner. The power gated supply line for the read port (V_{DDrp}) was placed to the right and on the lower third of the SRAM, so the wire in the final SRAM block could be layed out horizontally.

Additionally, the pins for the bitlines (BL, BLB and RBL) were setup so that the bitline wires in the final SRAM block could be vertical and not interfere with each other; and the pins for the word lines (WWL and RWL) were setup so that the word line wires in the final SRAM block could

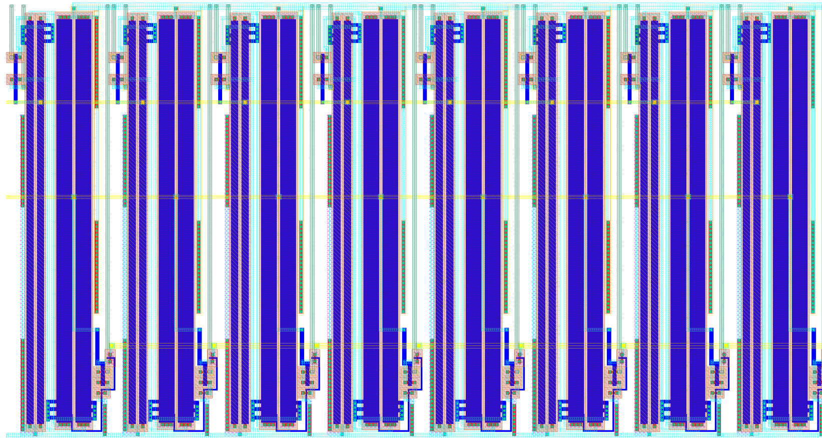


Figure 3.9: 10T 8 bit layout.

be horizontal, without interfering with the others. This can be seen in the detailed zoom in figure A.4 in appendix.

In the bitcell, all the metal routed was in the Metal1 layer. This was done in order to avoid possible conflicts with upper layers when routing the signals in the final SRAM block. The bitcell has a layout size of 13.54 µm by 53.56 µm, which means an area of 725.2 μm^2.

Eight bitcells were joined horizontally, in order to make the layout for a byte, and the physical layout can be seen in figure 3.9. The bitline pins for the 8 bits of this cell were put along the top of the byte cell, and the word line pins (just two: WWL and RWL) were put on the left of the byte cell. Again, the power supply pin was put on the top right corner, and the ground pin on the bottom left corner. The power gated read supply pin was put on the right, on the bottom third of the cell.

For the byte cell and the SRAM block, Metal2 and Metal4 were used when placing horizontal wires; Metal3 was used for vertically placed wires. The layout of a byte has a size of 102.5 µm by 54.2 µm, which means an area of 5555.5 μm^2.

When taking into account the size of the byte layout, and that a chip in the 0.18 µm process is 1.2 mm by 1.2 mm, the only possible arrangement where the full SRAM block (that has 128 bytes) can fit into a chip is by placing a grid of 8 bytes horizontally by 16 bytes vertically. As such, the layout for 128 bytes of the SRAM block can be seen in figure 3.10.

All pins for the bitlines were put along the top of the SRAM block, with the bitlines running vertically in Metal3, and connected horizontally with Metal2. The pins for the 128 write word lines and 128 read word lines were put along the left side of the SRAM block, in groups of eight (corresponding to the 8 byte in each row). The layout of the final SRAM block has dimensions of 819.1 µm by 870.8 µm, reaching an area of 731272.3 um^2 (0.7313 mm^2).

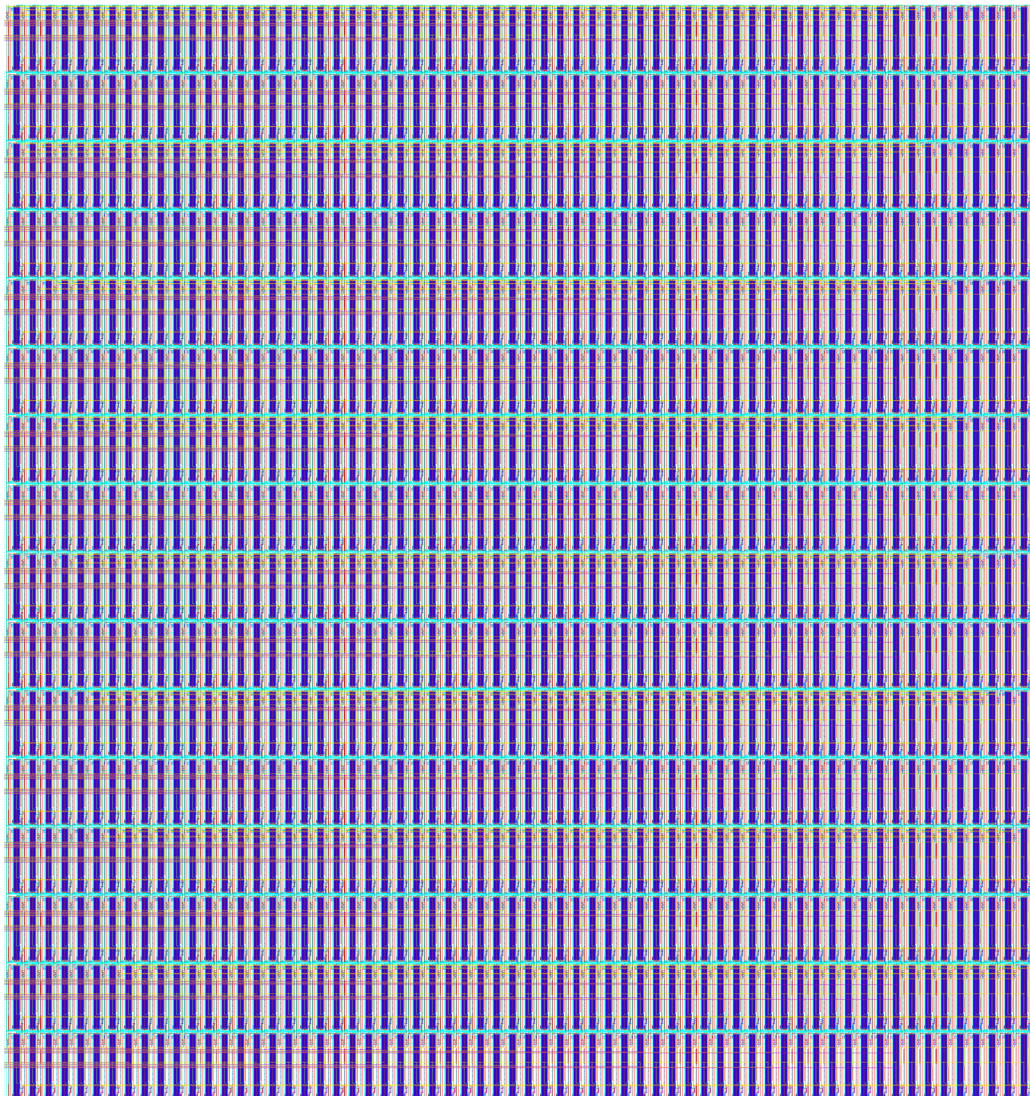
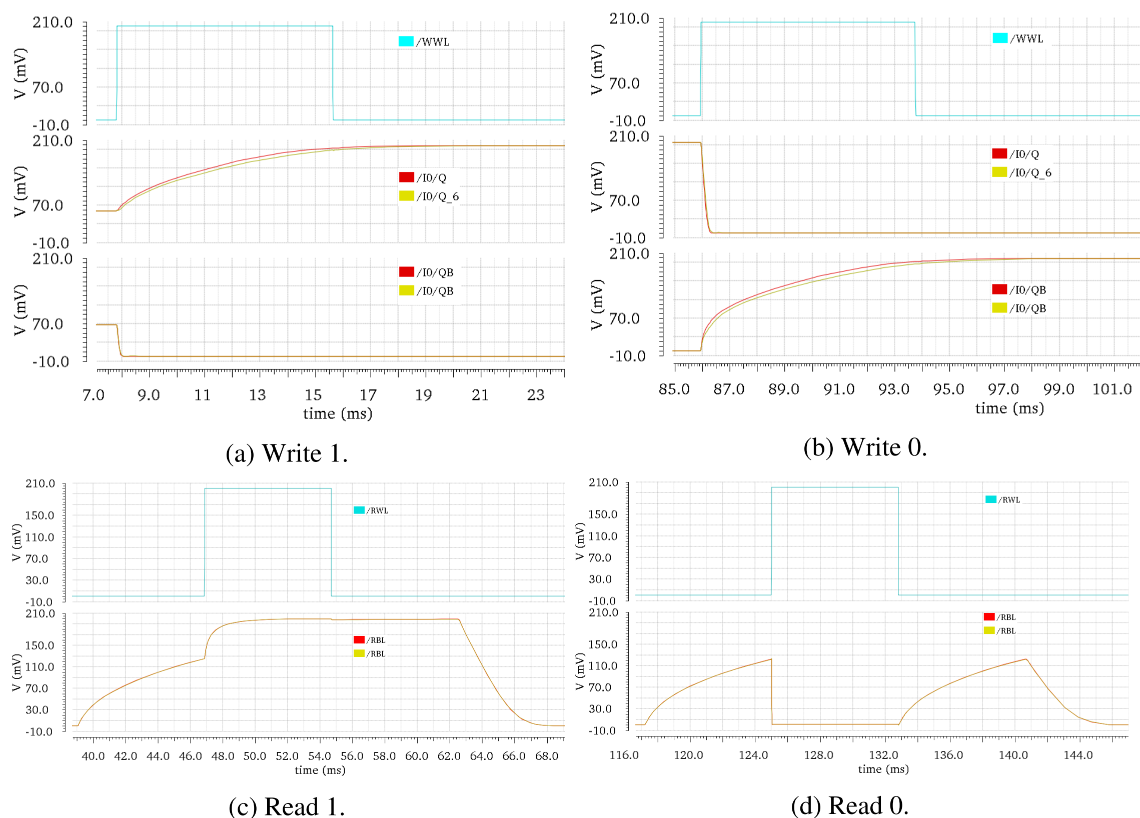


Figure 3.10: 10T 128 byte layout.



(a) Write 1.

(b) Write 0.

(c) Read 1.

(d) Read 0.

Figure 3.11: Schematic and post layout simulation comparison.

3.4 Schematic vs Layout Comparison

This section will cover the tests comparing the signals in schematic simulation against post layout simulation, for each of the steps in the modular layout design (i.e. bit, byte, final block).

Bitcell

To test the bitcell, a simple cycle of **write 1** (figure 3.11a), **read 1** (figure 3.11c), **write 0** (figure 3.11b), **read 0** (figure 3.11d) was ran.

When writing, the waveforms of **Q** and **QB** are almost identical, with the major difference being the rise time of Q (when writing a "1") or QB (when writing a "0"), which is 14% larger. This, however, does not pose any limitation on the bitcell functionality, since the value is already set by QB (when writing a "1") or Q (when writing a "'0"), seeing that the fall time is much shorter.

When reading either a "0" or a "1", the waveform of the read bitline (RBL) is identical whether in schematic or in post layout simulation.

Full block

The full block was also simulated in both schematic and calibre (post-layout), and the main difference in the two simulations was in the read bitline (RBL).

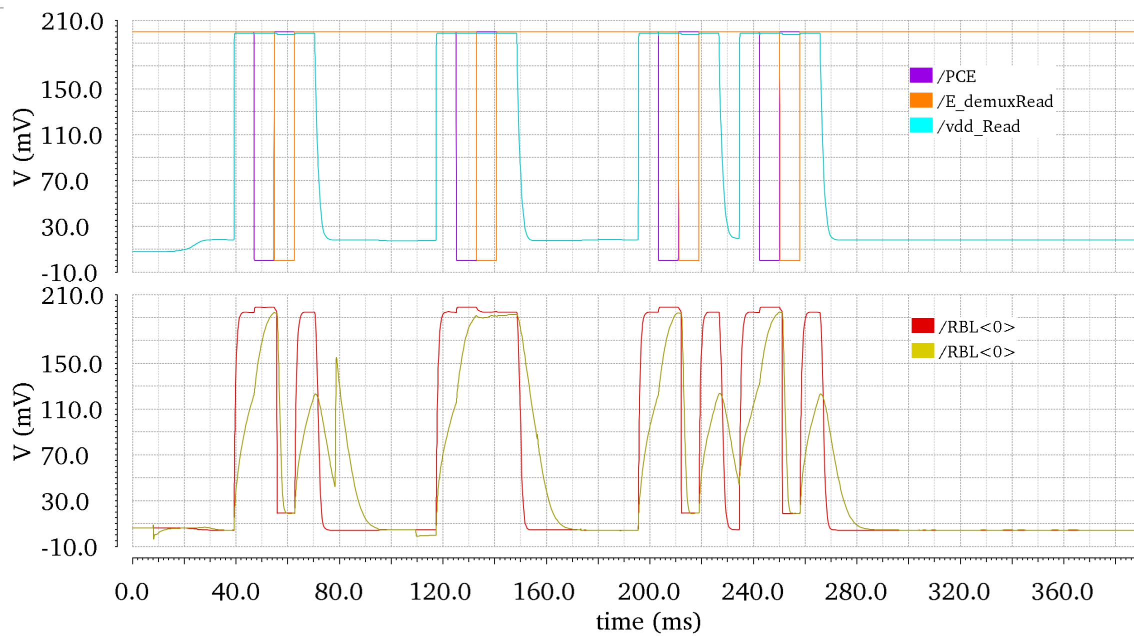


Figure 3.12: Read bitline for the schematic and post-layout simulations.

Looking at figure 3.12, it can be seen that the rise and fall times in the RBL of the post-layout simulation are much larger than the ones in the schematic simulation. This barely affects the final voltage of the RBL, as the final value when reading a 0 is at 18.945 mV in the schematic simulation, and is at 18.886 mV in the post-layout; when reading a 1, the final voltage in RBL is at 194.826 mV in schematic, and at 192.545 mV in post-layout.

This increase in the rise and fall times of the waveform in RBL can be explained by high resistance and/or high capacitance in the RBL wire in the layout.

Chapter 4

SRAM Control Circuits

The control circuits allow the SRAM to function: they coordinate the several steps required for each write or read operation, and allow the data to be saved to/read from the correct word in memory.

The control circuits that are going to be used in this SRAM are:

- **7 to 128 demultiplexer**, that is going to function as an address decoder, allowing to convert a 7 bit address into 128 lines that control each of the words individually
- **Bitline drivers** that allow for the differential writing that the SRAM bitcell needs
- **Power gating** circuits, used to lower the idle power of the SRAM memory by completely shutting off parts of the circuit that are only needed during a write or read operation
- **Read circuits** to aid in the read operation
- **State machine** that generates the control signals needed for the several steps of writing or reading a byte

As the control circuits need to comply with the dual gated nature of the 10T bitcell, there is the necessity for two address decoders: one for writing and one for reading.

Also, as the read port of the 10T bitcell is single ended, the reading circuits cannot consist of the traditional differential sense amplifier seen in SRAMs based on the 6T bitcell [2].

4.1 Logic Gates

The logic gates needed for the control circuit (mostly the demultiplexer) can be seen in figure 4.1, and were each optimized with the target of the lowest possible power consumption. These logic gates are: **inverter**; two and three input **NAND**; two and three input **NOR**. During the optimization, the correct functioning of the logic gates, the drive strength and the rise and fall times were taken into account. The different optimization criteria can be seen in figure 4.2, where the steps are followed from left to right.

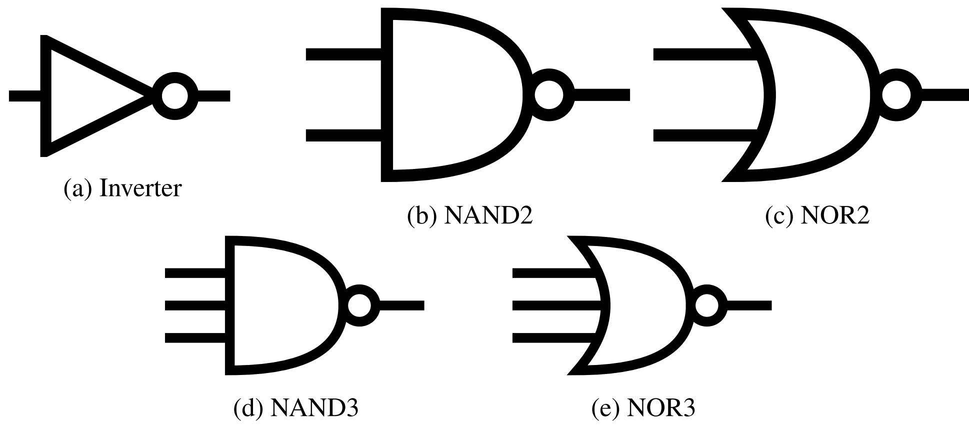


Figure 4.1: Logic gates necessary for the SRAM control circuits.

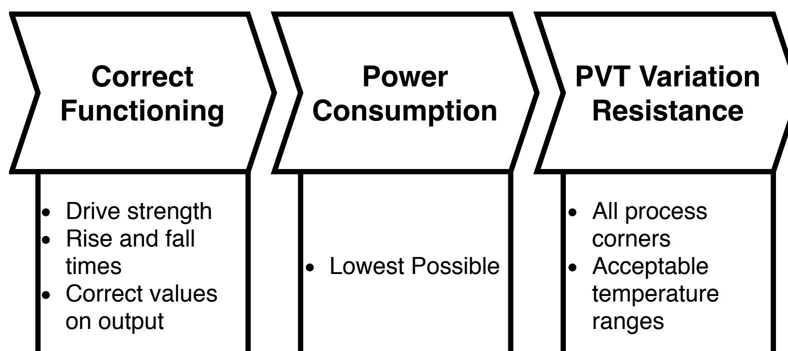
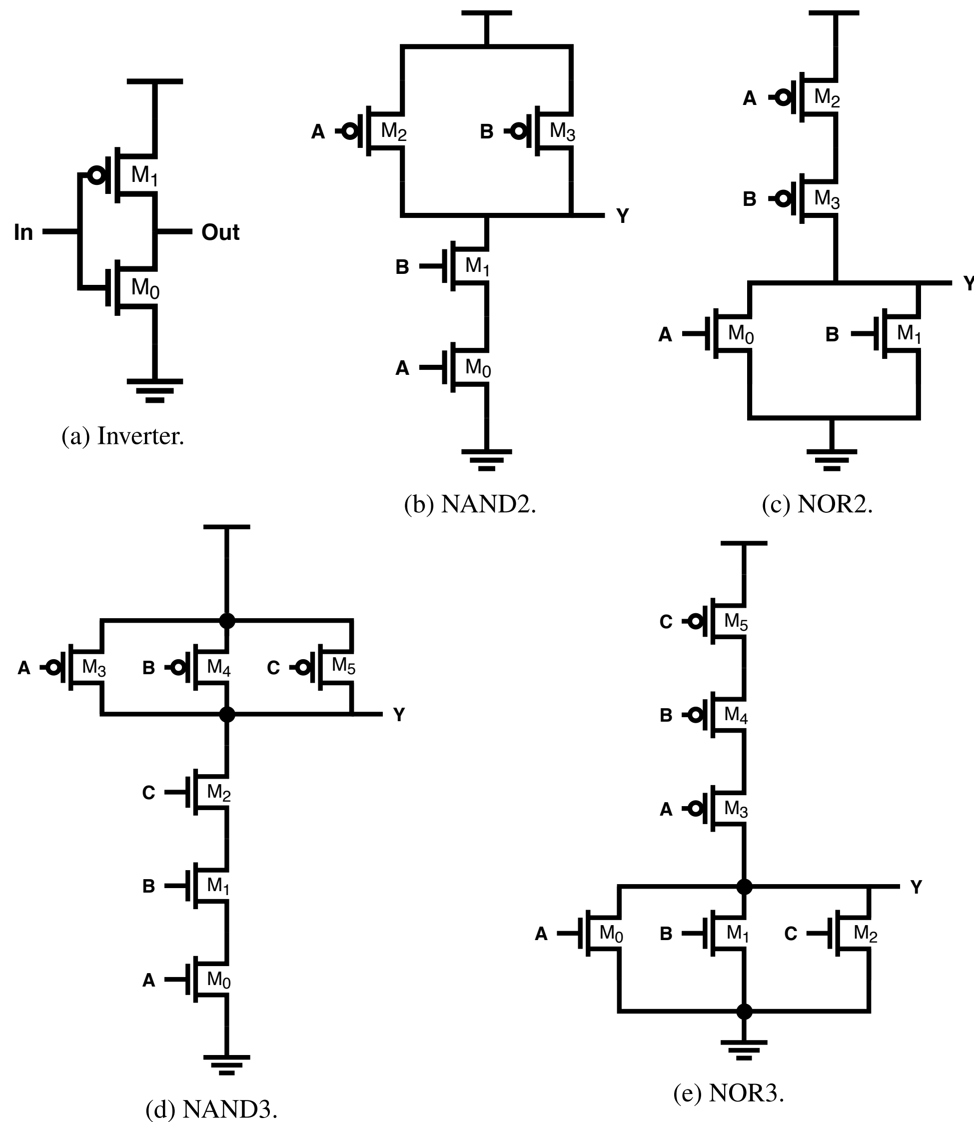


Figure 4.2: Optimization steps for the logic gates.



(a) Inverter.

(b) NAND2.

(c) NOR2.

(d) NAND3.

(e) NOR3.

Figure 4.3: Logic gates transistor level schematic.

For all logic gates, the parametric simulations were run using Cadence Virtuoso ADE XL. In this tool, the parameters were set using the **Corners** tab. The values for the length and width of the transistors were swept using "Auto:Initial value:Number of steps:Final value".

Using the Cadence Virtuoso Calculator, expressions were set up to determine: output voltage for all combination of inputs; rise and fall times; and power consumption. In the ADE XL "Outputs Setup" tab, each expression result regarding voltage levels or rise/fall times was assigned a spec: a value or expression to determine if the corner passed the test.

After each run, each expression was plotted across corners, allowing for a reduction of the range of the inputs values (length and width of the transistors). After 6 to 10 runs, a logic gate was finally optimized. The inputs for the tests were generated using VerilogA scripts, for ease of generating the testbenches.

For most logic gates, all PMOS transistors were assumed to be equal, as well as all the NMOS

Table 4.1: Transistor sizing for the logic gates.

Logic Gate		M0	M1	M2	M3	M4	M5
Inverter	**Width**	500 nm	1 µm	-	-	-	-
	Length	400 nm	600 nm	-	-	-	-
NAND2	**Width**	400 nm	400 nm	400 nm	400 nm	-	-
	Length	470 nm	470 nm	1 µm	1 µm	-	-
NAND3	**Width**	700 nm	700 nm	700 nm	600 nm	600 nm	600 nm
	Length	450 nm	450 nm	450 nm	1 µm	1 µm	1 µm
NOR2	**Width**	400 nm	400 nm	240 nm	300 nm	-	-
	Length	470 nm	470 nm	1 µm	900 nm	-	-
NOR3	**Width**	450 nm	450 nm	450 nm	1.4 µm	1.4 µm	1.4 µm
	Length	450 nm	450 nm	450 nm	600 nm	600 nm	600 nm

transistors. Only the NOR2 bitcell was optimized using different transistor sizing for the different transistors of the same type, but this resulted in a power consumption reduction of 1-2% only. Thus, the other logic gates were not optimized using this criteria.

Inverter

The inverter, which transistor level schematic can be seen in figure 4.3a, was optimized in schematic, using another instance of the same inverter as load. It was the first logic gate to be optimized, because of the intention to use it as both drive and load when optimizing the other logic gates.

The final sizing of the transistors can be seen in table 4.1. It was tested for its functioning under process variations and temperature change, and the functioning range of the inverter can be seen in table 4.2.

NAND2

The NAND2 logic gate, which schematic can be seen in figure 4.3b, was optimized in schematic, following the general criteria defined for the logic gates (power consumption and correct function-

Table 4.2: PVT variation for the logic gates.

Logic Gate	Process Variation	tt	ff	ss	snfp	fnsp
Inverter	**Min. Temp. (ºC)**	-40	-40	-20	-20	-20
	Max. Temp. (ºC)	180	200	180	200	180
NAND2	**Min. Temp. (ºC)**	-20	-40	0	-20	0
	Max. Temp. (ºC)	140	140	120	140	120
NAND3	**Min. Temp. (ºC)**	-20	0	0	-20	0
	Max. Temp. (ºC)	120	140	120	140	120
NOR2	**Min. Temp. (ºC)**	0	-40	0	-20	0
	Max. Temp. (ºC)	120	120	140	140	80
NOR3	**Min. Temp. (ºC)**	-20	-40	0	0	0
	Max. Temp. (ºC)	140	160	80	100	140

ing). The load was an inverter, an each of the two inputs was driven by an inverter. The two NMOS transistors (M0 and M1) were assumed to be equal to each other, and the same was assumed for the two PMOS (M2 and M3). This assumption was followed in order to reduce the optimization time for the logic gate.

The final sizing of the transistors can be seen in table 4.1. The NAND2 was tested for its functioning under process variations and temperature change, and the functioning range of the logic gate can be seen in table 4.2.

NAND3

The NAND3 logic gate, which schematic can be seen in figure 4.3d, was optimized in the same way as the NAND2. The load was an inverter, and each of the three inputs was driven by an inverter. The three NMOS transistors (M0 - M2) were assumed to be equal to each other, and the same was assumed for the three PMOS transistors. This assumption was once again followed in order to reduce the optimization time for the logic gate.

The final sizing of the transistors can be seen in table 4.1. The NAND3 was tested for its functioning under process variations and temperature change, and the functioning range of the logic gate can be seen in table 4.2.

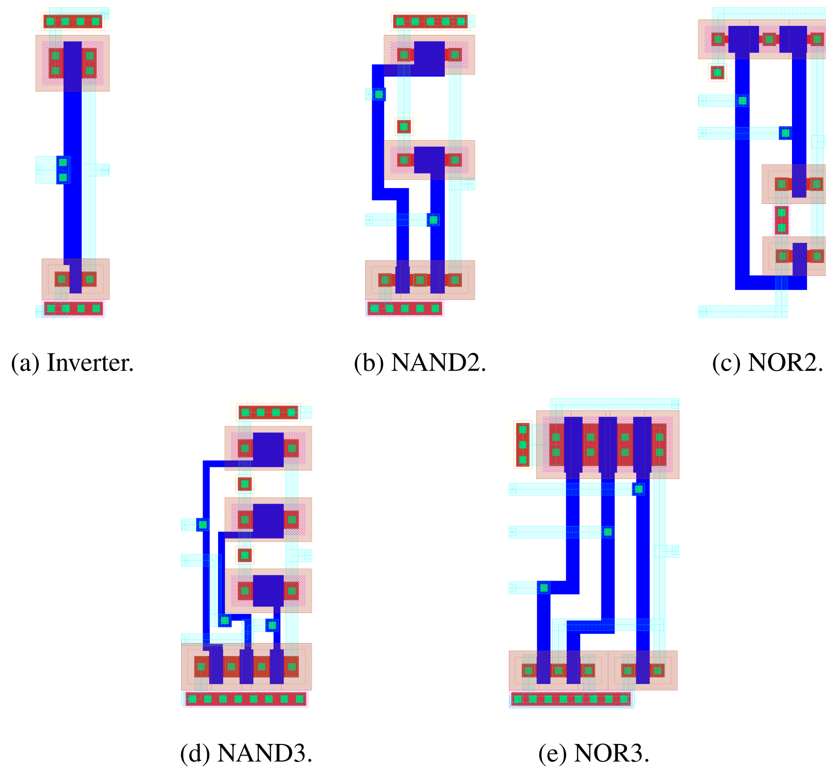
NOR2

The NOR2 logic gate, which schematic can be seen in figure 4.3c, was optimized in schematic two distinct times, using the criteria defined above. Both times, the load was an inverter, an each of the two inputs was driven by an inverter. In the first optimization, the two NMOS transistor (M0 and M1) were assumed to be equal to each other, and the same was assumed for the two PMOS (M2 and M3). This assumption was followed in order to reduce the optimization time for the logic gate.

During the second optimization, this assumption was not followed, in order to try to achieve a lower power consumption. As this only resulted in an improvement of around 1% to 2%, the assumption was taken on the other logic gates. The final sizing of the transistors can be seen in table 4.1. The NOR2 was tested for its functioning under process variations and temperature change, and the functioning range of the logic gate can be seen in table 4.2.

NOR3

The NOR3 logic gate, which schematic can be seen in 4.3c, was optimized in schematic for the lowest possible power consumption, while guaranteeing correct functioning. As with the other logic gates, the load was an inverter and each of the inputs was driven by an inverter. The three NMOS transistors (M0 - M2) were assumed to be equal to each other, and the three PMOS (M3 - M5) were also assumed to be equal to each other. This assumption was made in order to reduce the optimization time for the logic gate.



(a) Inverter. (b) NAND2. (c) NOR2.

(d) NAND3. (e) NOR3.

Figure 4.4: Logic gates layout.

The final sizing of the transistors can be seen in table 4.1. The NOR3 was tested for its functioning under process variations and temperature change, and the functioning range of the logic gate can be seen in table 4.2.

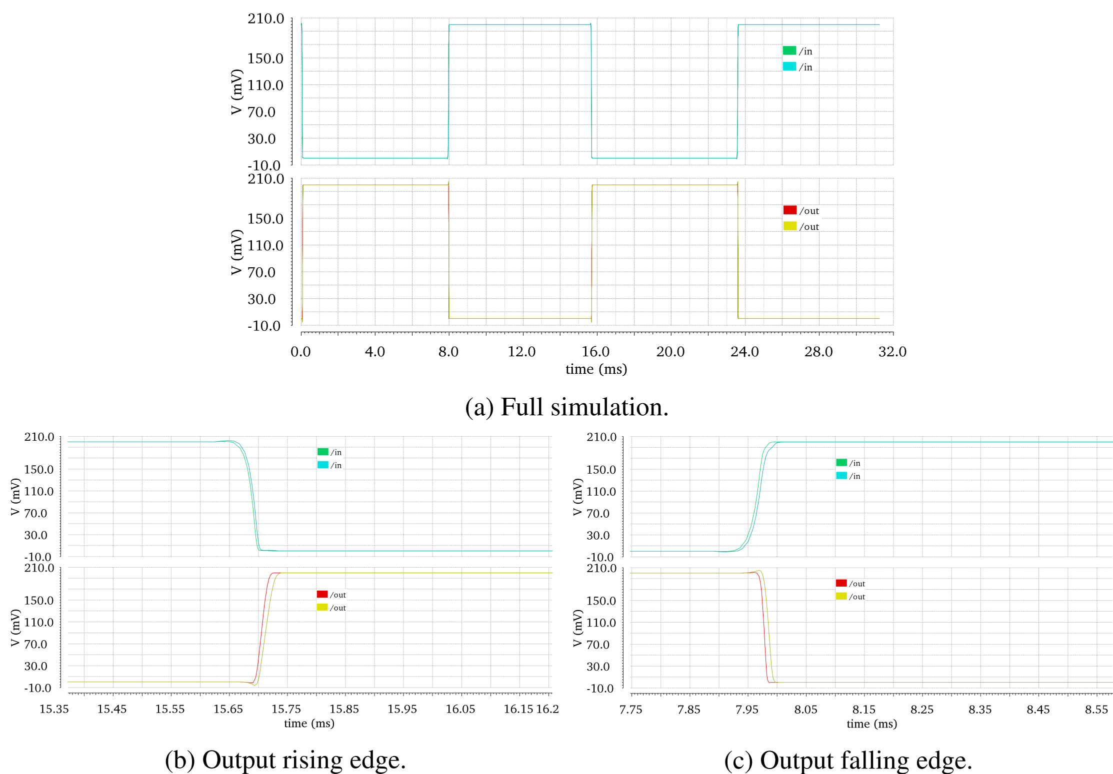
4.2 Logic Gates - Layout

The logic gates are used mostly in the demultiplexer and the flip flops, and inverters are used to accept the inputs and reduce the fan-out of some logic gates.

In order to facilitate the layout, some guidelines were setup:

- All logic gates should have the **same height**
- The power supply line (V_{DD}) should be on the top of the layout, and to the right
- The ground wire should be on the bottom of the layout, and to the left
- All inputs should be put along the left of the logic gate
- All outputs should be put along the right of the logic gate

All logic gates have a layout height of 10.25 μm.



(a) Full simulation.

(b) Output rising edge.

(c) Output falling edge.

Figure 4.5: Comparison of schematic and post-layout simulation for the inverter.

Inverter

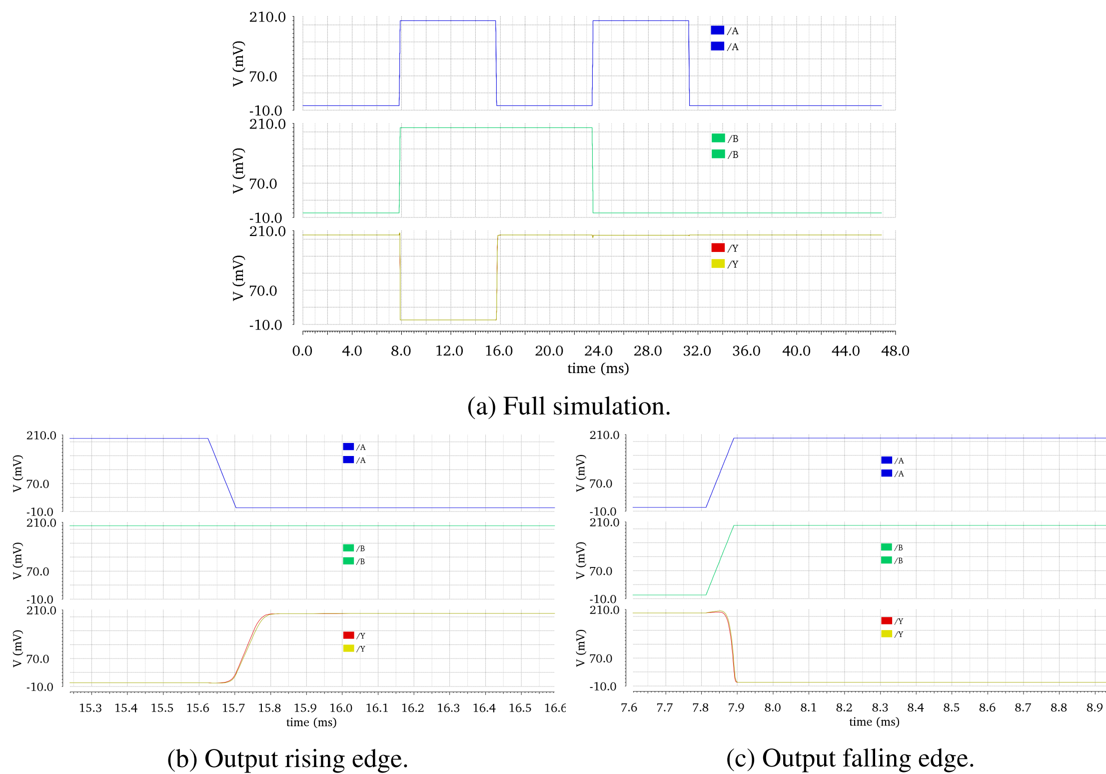
The layout of the inverter can be seen in figure 4.4a. It has a width of 2.44 μm, and an area of 25.01 μm^2.

The simulation that compares the schematic and layout of the inverter can be seen in figure 4.5. In the top half, the waveforms for the input of the inverter can be seen: in green, the waveform of the schematic simulation; in blue, the waveform for the post-layout simulation. It can be observed that the behaviour is identical whether in schematic or post-layout simulation, and the only difference is in the rise and fall times. This difference can be seen in more detail in figure 4.5b for the rise time and figure 4.5c for the fall time, with the rise time increasing from 18.58 μs in the schematic simulation to 23.47 μs in the post layout simulation, and the fall time increasing from 10.24 μs in the former to 13.11 μs in the latter.

NAND2

The NAND2 layout can be seen in figure 4.4b. It has a width of 3.6 μm, which gives a layout area of 36.9 μm^2.

The simulation comparing the schematic and the layout of the NAND2 logic gate can be seen in figure 4.6. In figure 4.6a, the top wave and middle wave, colored blue and greeen, respectively, are the two inputs of the NAND; the bottom waveforms correspond to the output of the NAND,



(a) Full simulation.

(b) Output rising edge.

(c) Output falling edge.

Figure 4.6: Comparison of schematic and post-layout simulation for the NAND2.

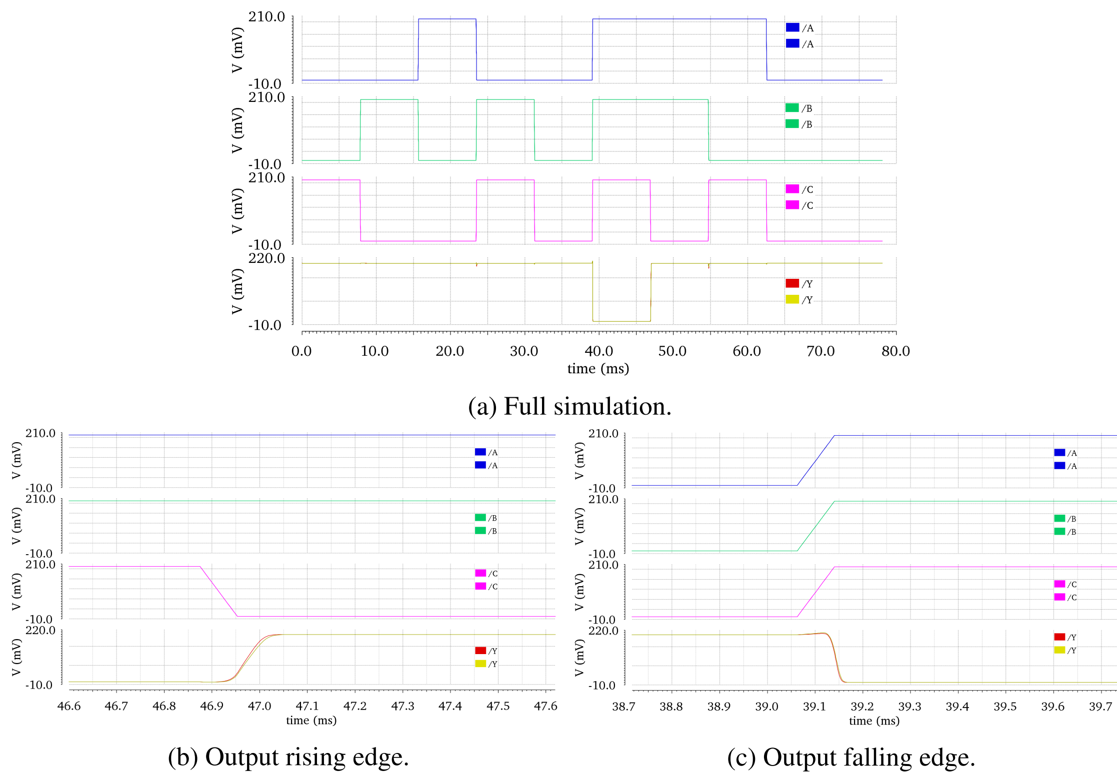
with the red wave belonging to the schematic simulation and the yellow wave to the post-layout simulation.

In figure 4.6b, the rise of the output is represented with more detail, and there is only a slight difference in the rise time of the output, which goes from 20.04 µs in the schematic simulation to 19.08 µs in the post-layout simulation; in figure 4.6c, there is a more detailed representation of the fall of the output, with the two waveforms differing only in the fall time, which raises from 69.41 µs in the schematic simulation to 73.68 µs in the post-layout simulation.

NAND3

The NAND3 layout can be seen in figure 4.4d. Its width is 4.27 µm, which means an area of approximately 43.78 μm^2.

In figure 4.7, the results of the simulations comparing the schematic and the layout can be seen. In figure 4.7a, the first three waveforms from the top are the three inputs of the NAND; in figure 4.7b, there is a more detailed waveform of the output when it is rising. The only difference is the rise time, which is slightly larger in the post-layout simulation. The rise time goes from 59.37 µs in the schematic simulation to 64.1 µs in the post-layout simulation; in figure 4.7c, the fall of the output can be seen in more detail, and the difference in the waveforms is only in the fall time, which increases from 20.04 µs in the schematic simulation to 20.96 µs in the post-layout simulation.



(a) Full simulation.

(b) Output rising edge.

(c) Output falling edge.

Figure 4.7: Comparison of schematic and post-layout simulation for the NAND3.

NOR2

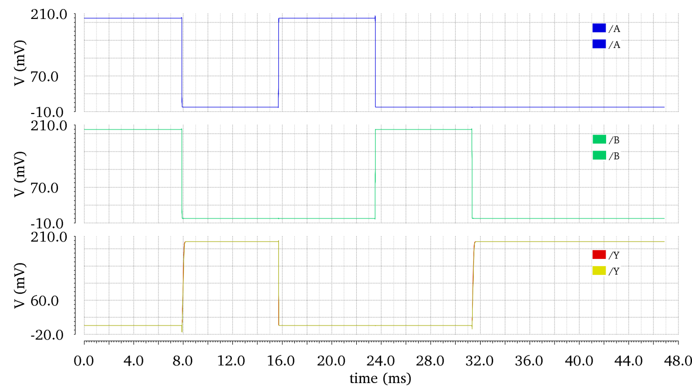
The NOR2 layout can be seen in figure 4.4c and has a width of 4.56 µm. This gives a layout area of 46.74 μm^2.

The waveforms of the simulation comparing the schematic and the layout can be seen in figure 4.8. In figure 4.8a, the waveforms on the top, colored blue and green, are the inputs of the NOR2; the waveforms on the bottom correspond to the output of the NOR2, colored red for the schematic simulation and yellow for the post-layout simulation. In figure 4.8b, the rise of the output can be seen in more detail, and the difference between the waveforms lies in the rise time, which increases from 131.1 µs in the schematic simulation to 147.7 µs in the post-layout simulation. In figure 4.8c, the fall of the output can be seen, with the fall time going from 7.6211 µs in the schematic simulation to 9.373 µs in the post-layout simulation.

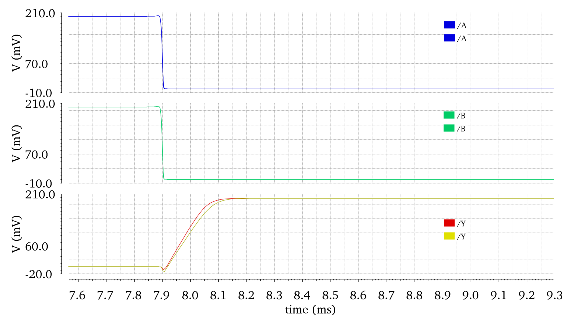
NOR3

The NOR3 layout can be seen in figure 4.4e. It has a width of 5.63 µm, and an area of approximately 57.71 μm^2.

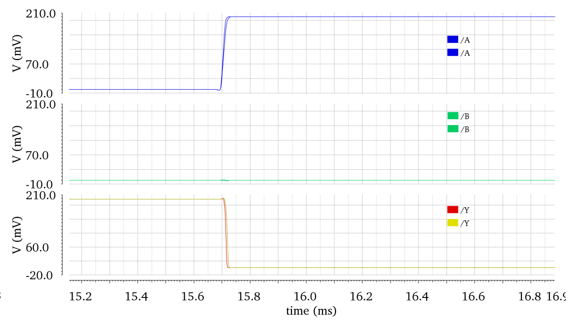
In figure 4.9, the waveforms resulting from the simulation comparing schematic and layout are presented. In figure 4.9a, the top three waveforms, colored blue, green and pink, are the inputs of the NOR3; the bottom waveforms correspond to the output of the NOR3: colored red for the schematic simulation and yellow for the post-layout simulation. In figure 4.9b, the rise of the



(a) Full simulation.

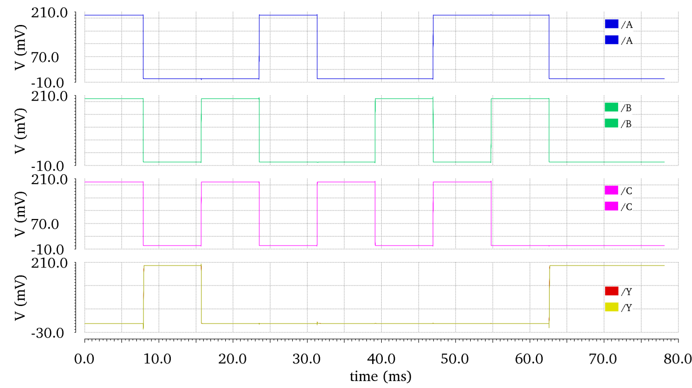


(b) Output rising edge.

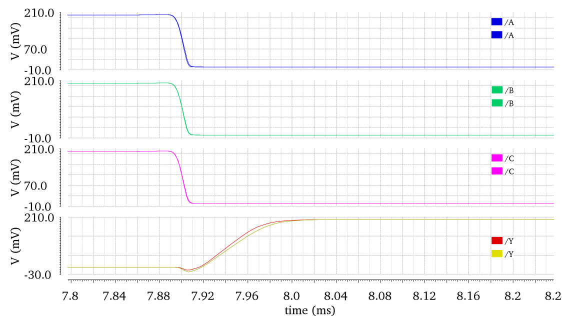


(c) Output falling edge.

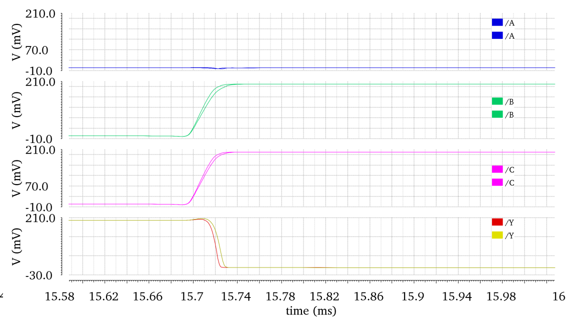
Figure 4.8: Comparison of schematic and post-layout simulation for the NOR2.



(a) Full simulation.

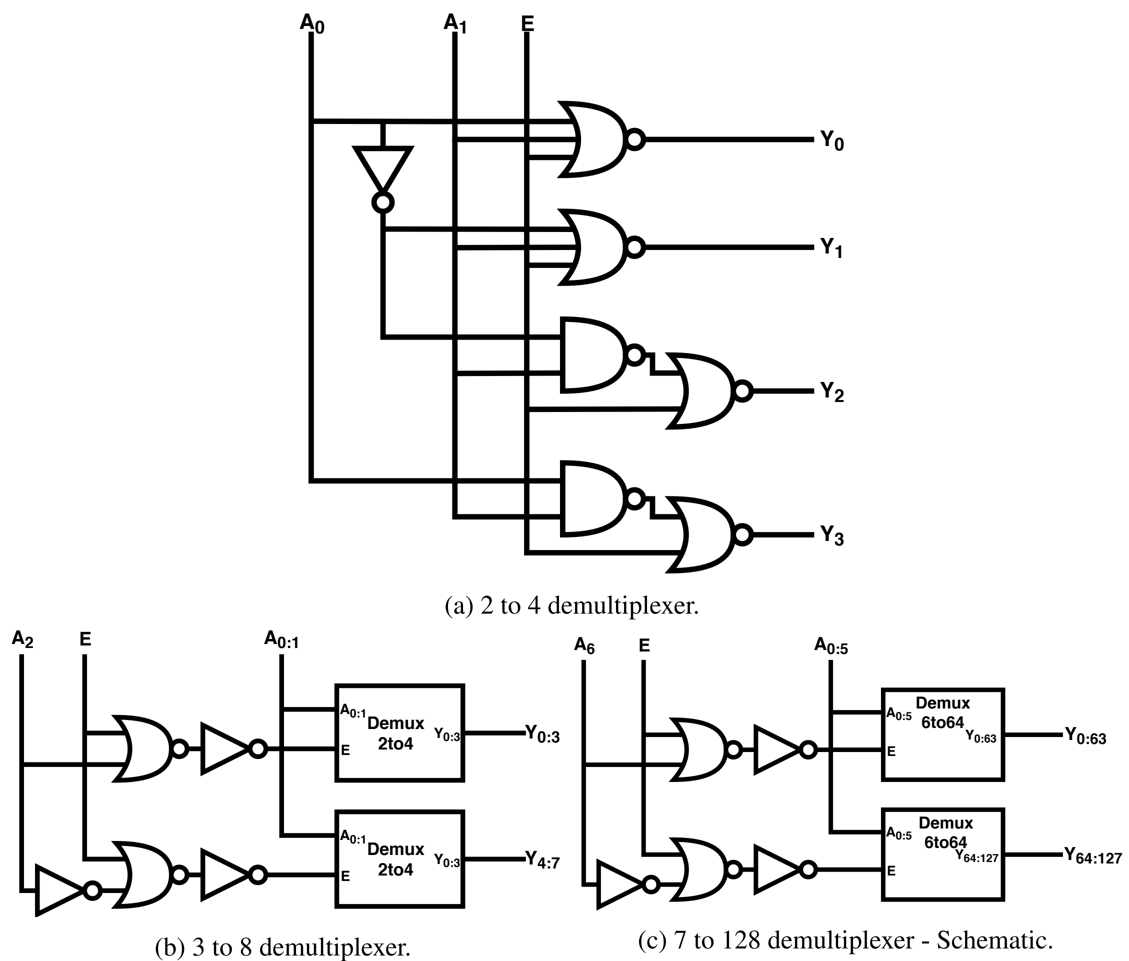


(b) Output rising edge.



(c) Output falling edge.

Figure 4.9: Comparison of schematic and post-layout simulation for the NOR3.



(a) 2 to 4 demultiplexer.

(b) 3 to 8 demultiplexer.

(c) 7 to 128 demultiplexer - Schematic.

Figure 4.10: Demultiplexer schematic.

output can be seen in more detail: the two waveforms differ only in the rise time, with the rise time going from 52.08 µs for the schematic simulation to 57.07 µs for the post-layout simulation. Looking at figure 4.9c, the fall of the output can be seen with more detail: the fall time is the major difference between the waves, with it increasing from 8.324 µs for the schematic simulation to 9.473 µs for the post-layout simulation.

In summary, the major differences when comparing the layout and schematic lie in the rise and fall times, which increase 12.7% on average.

4.3 Demultiplexer

The 7 to 128 demultiplexer that the control circuit needs was implemented using a recurrent aproach.

First, a 2 to 4 demultiplexer, seen in figure 4.10a, was designed and optimized for the lowest power consumption, while satisfying the need for rail to rail drive of the outputs and acceptable rise and fall times. Then, a 3 to 8 demultiplexer, seen in figure 4.10b, was designed, using two 2 to

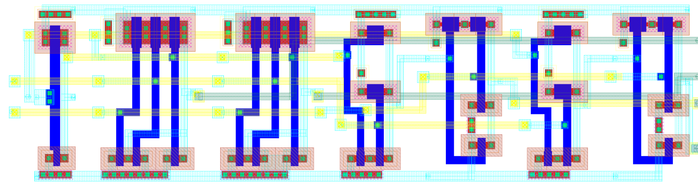


Figure 4.11: 2 to 4 Demultiplexer - Layout.

4 demultiplexers. This was repetead until the 7 to 128 demultiplexer was obtained. The schematic for the final demultiplexer can be seen in figure 4.10c.

4.3.1 Demultiplexer Layout

In similarity to the schematic, the layout for the 7 to 128 demultiplexer was done in a modular way, in order to ease the layout process. Using the previously layed out logic gates (Inverter, NAND2, NOR2 and NOR3), the 2 to 4 demultiplexer was layed out in a single row, as it can be seen in figure 4.11. This smallest demultiplexer was layed out with similar cares as the ones taken when laying out the logic gates: V_{DD} was routed on top; ground was routed across the bottom; all inputs were placed along the left of the demultiplexer; all outputs were placed on the right. The layout for this demultiplexer has a width of 41.41 µm and a height of 10.45 µm.

To layout the 3 to 8 demultiplexer, two of the 2 to 4 demultiplexers were placed, one above the other, with the remaining logic gates being put on the left of the two smaller demultiplexers. Again, all outputs were placed along the right of the demultiplexers. However, as this demultimplexer was not the final one and there will be a need to expand to the left when laying out the logic gates necessary for the larger demultiplexers, the two ground lines were not connected; the V_{DD} lines were connected, as the final 7 to 128 demultiplexer will not expand to the right. The inputs were placed near the left of the 3 to 8 demultiplexer, in order to facilitate the vertical stacking of these devices in order to layout the 7 to 128 demultiplexer. The layout for the 3 to 8 demultiplexer can be seen in figure 4.12, and has a width of 54.69 µm and a height of 21.14 µm.

When laying out the final 7 to 128 demultiplexer, an additional rule was set: as seen in the figure A.4, the word lines are arranged in groups of eight, interleaving a group of eight WWL with a group of eight RWL, for the same row of bytes, due to the final bitcell block byte arrangement

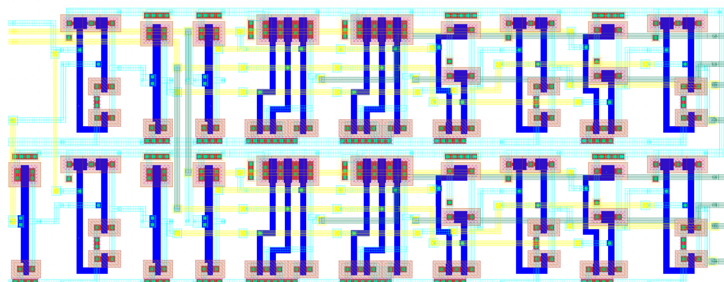


Figure 4.12: 3 to 8 Demultiplexer - Layout.

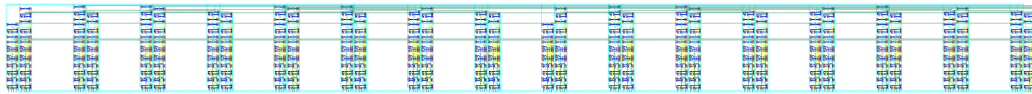


Figure 4.13: 7 to 128 Demultiplexer - Layout (the layout is rotated 90 degrees clockwise in this figure).

of 8 columns by 16 rows, resulting in a spacing of 54.44 µm between groups of word lines. As the 3 to 8 demultiplexer has a height of 21.14 µm, there is enough spacing to interleave the two 7 to 128 demultiplexers, with one being used for the write operation and the other for the read operation. As such, the 16 needed 3 to 8 demultiplexers were stacked vertically, with a spacing of 54.44 µm between the top of a demultiplexer and the top of the next. All of the needed logic gates to implement the 7 to 8 demultiplexer from the 3 to 8 demultiplexers were added to the left of the latter, being distributed among the 16 in order to minimize the width of the final demultiplexer. As such, the final width of the demultiplexer is 70.47 µm, the final height is 837.74 µm, giving a final area of 59035.5 µm^2.

4.4 Drive Circuits

The circuits consist mainly of high current output inverters. Naturally, these need to have larger transistors (in width) than the regular inverters, so as to comply with acceptable rise and fall times of the signal in greater loads (such as the bitlines, for example).

These strong inverters were originally optimized for driving the inputs of the demultiplexer, as there are inputs that need to drive around 130 gates.

4.5 Drive Inverters - Layout

The layout for the larger inverter used to drive the bitlines, as well as most of the inputs with a larger fan-out, can be seen in figure 4.14. Its width is 16.03 µm, and its height is 10.25 µm, which gives a layout area of 164.31 µm^2.

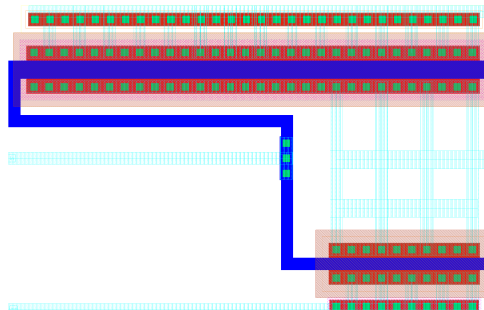


Figure 4.14: Inverter (large) - Layout.

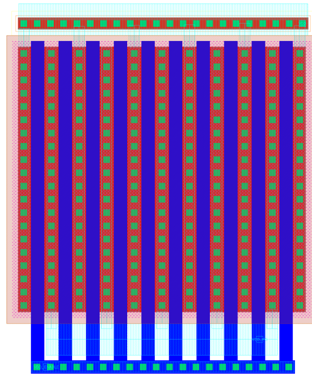


Figure 4.15: Layout of the power gating transistor.

4.6 Power Gating

In this memory, there will be three power supply lines, all with the same voltage level. Besides the main V_{DD} line, there will be two power gated lines: one for the reading operation and the other for the writing operation. Each power gated supply line has to power one 7 to 128 demultiplexer, as well as the bitline drivers (for the power gating circuit associated with the write operation) or the read circuits (for the power gating circuit associated with the read operation).

The power gating is done by a single PMOS transistor, which was optimized to provide adequate current when closed and to have the lowest possible shut off current. This resulted in a transistor with a length of 500 nm and a width of 100 µm. In order to facilitate the layout of the power gating transistor, the finger number was increased to 10, making the layout more compact.

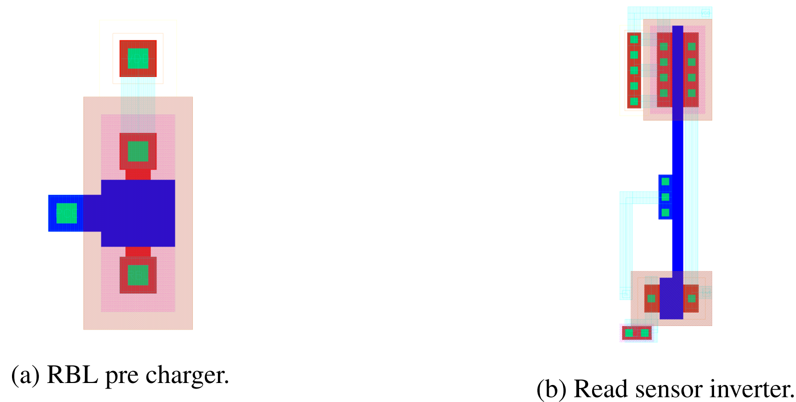
The layout for this component can be seen in figure 4.15. This single PMOS transistor (and its connections) have a layout width of 11.7 µm and a height of 13.95 µm, resulting in an area of 163.22 µm^2.

4.7 Read Circuits

The read circuits that aid in the read operation consist of a read bitline **pre charger**, that charges the bitline to V_{DD} before the reading process; and an **inverter** that serves as load for the read bitline, and is optimized to have a V_M of half of V_{DD}. This allows it to distinguish the voltage level for reading a zero from the voltage level from reading a one, even in the most extreme corners of process and temperature.

Pre Charger

The pre charger consists in a single PMOS transistor. It is used only to guarantee that the RBL is at V_{DD} when the read port is closed, assuring that the read port only has to function has a pull-down circuit. Since it does not need to provide high current, its dimensions are 800 nm for the length and 300 nm for the width.



(a) RBL pre charger.

(b) Read sensor inverter.

Figure 4.16: Layout of the read circuits.

Its layout can be seen in figure 4.16a. Due to its purpose, the PMOS dimensions (width and length) are much smaller than those of other transistors in this SRAM, giving its layout a width of 1.72 µm and a height of 3.7 µm, resulting in a layout area of 6.36 μm^2.

Read Inverter

This inverter was optimized with a different goal than the two other already optimized: instead of aiming for the lowest possible power consumption, or the ability to drive a large number of transistor gates, this inverter was optimized with the main purpose of having its V_M at half of V_{DD}. This was done to improve the functioning region of the SRAM: with the V_M at half of V_{DD}, it is easier to distinguish when the RBL voltage level is the result of a stored 0 or 1.

The inverter was layed out, and the result can be seen in figure 4.16b. It has a width of 2.95 µm and a height of 10.81 µm, giving an area of 31.89 μm^2. The height is larger than the 10.25 µm that all the other logic gates have; however, this will not pose any problem to the final layout, as this inverter is only used to sense the RBL, and it does not have to be layed out alongside other logic gates.

4.8 State Machine

Both the write and read operation consist of several steps: write has to power up the demultiplexer and the bitline drivers, decode the write address and power down; read has to power up the read ports and the demultiplexer, precharge the RBL, decode the read address and power down. There is the need for a state machine to coordinate these steps and operations. The state machine will be implemented using D type flip flops, chained together so as to create a counter. This will be used to send the control signals for the write and read operations.

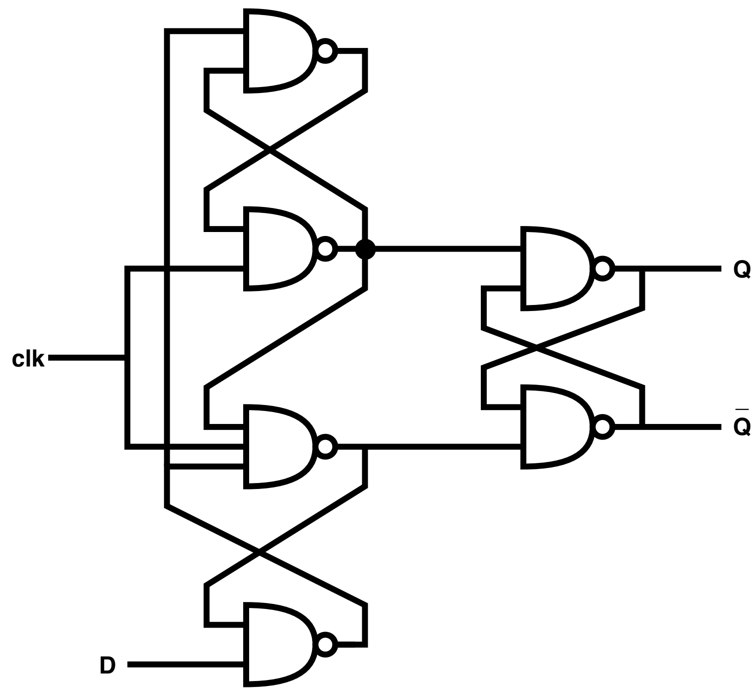


Figure 4.17: D type Flip Flop - Schematic.

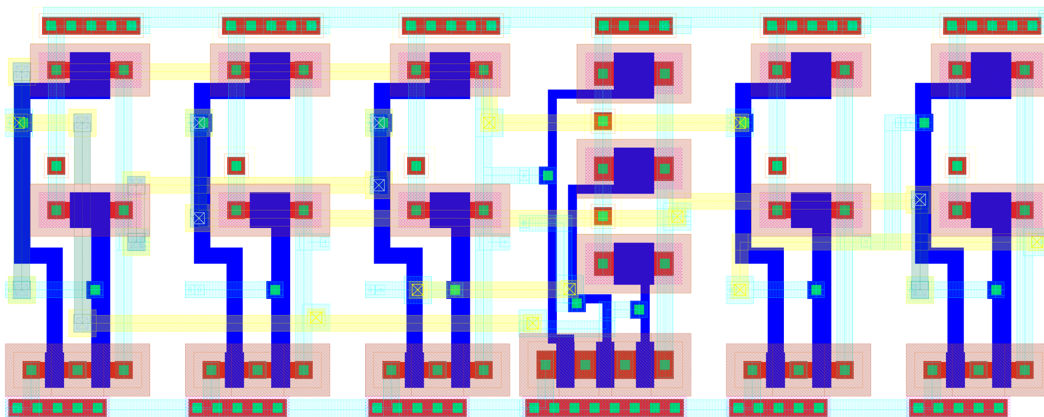


Figure 4.18: Layout of the D type flip flop.

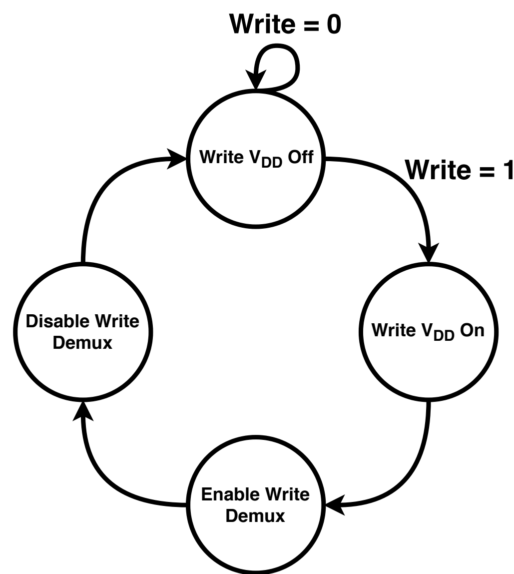


Figure 4.19: State machine for the write operation.

4.8.1 Flip Flop

The D type flip flop, which schematic can be seen in figure 4.17, was implemented using two SR latches, using the previously optimized logic gates (NAND2 and NAND3). The flip flop was layed out using the logic gates previously described, laying them out in a single row. Its final layout can be seen in figure 4.18, and its dimensions are: 26.03 µm for the width and 10.25 µm for the height, resulting in an area of 266.8 μm^2. The inputs (D and clk) were put on the left side of the flip flop, and the outputs were put along the right side; once again, the V_{DD} line runs across the top of the layout, and the ground line across the bottom.

4.8.2 Write State Machine

The write operation has the following steps, that can also be seen in figure 4.19:

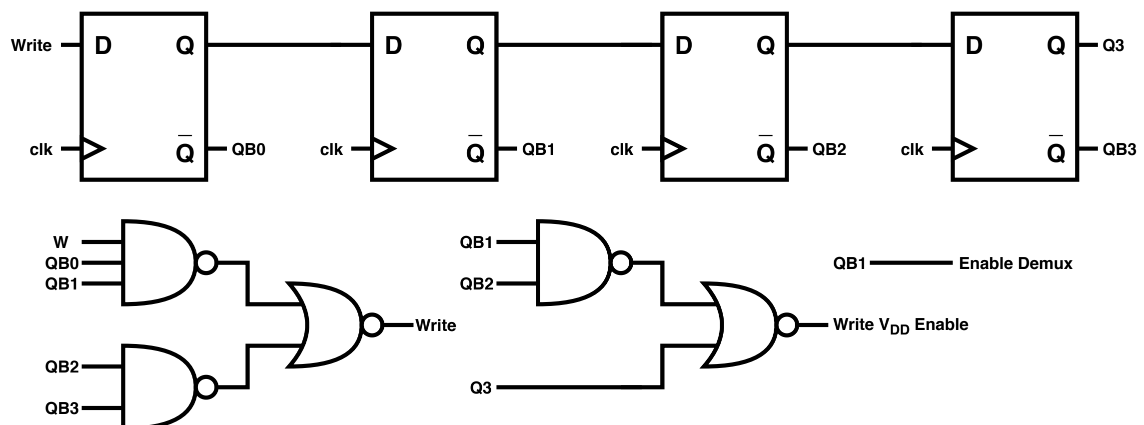


Figure 4.20: State machine implementation with D type flip flops for the write operation.

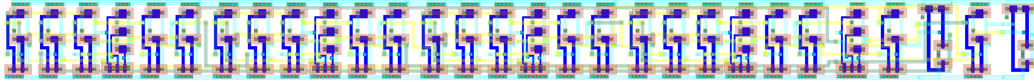


Figure 4.21: Write state machine layout.

1. Turn on the write power supply
2. Enable the write demux
3. Disable the write demux
4. Turn off the write power supply

Taking into account the steps for the write operation, the state machine can be implemented with four D type flip flops, as seen in figure 4.20. The state machine accepts as input the clock signal and a "W" signal, which signals a write operation when at logic level "1"; and outputs the two control signals needed for the write operation: **Write V_{DD} Enable** and **Write Demux Enable**. To prevent the activation of more than one step at the same time, the input of the first flip flop can only be at logic level "1" when the outputs of all the flip flops are at logic level "0".

The layout for the write state machine can be seen in figure 4.21. The four flip flops and the other logic gates (NAND and NOR) were layed out in a single row, with the chained flip flops on the left. The inputs and outputs can be accessed across the width of the state machine, which eases the final integration. The layout of this state machine has a width of 136.71 µm and a height of 10.45 µm, resulting in an area of 1428.6 μm^2.

4.8.3 Read State Machine

The read operation has the following steps, that can also be seen in figure 4.22:

1. Turn on the read power supply
2. Pre charge the read bitlines
3. Enable the read demux
4. Disable the read demux
5. Turn off the read power supply

When taking into account the steps necessary, the read state machine can be implemented with five chained D type flip flops, as seen in figure 4.23. The state machine inputs are the clock signal and a "R" signal, which means the start of a read operation when at logic level "1". Its outputs are the three control signals needed for the read operation: **Read V_{DD} Enable**, **Pre Charge Enable** and **Read Demux Enable**. As in the write state machine, the input of the first flip flop is unable to go to logic level "1" when a read operation is occurring.

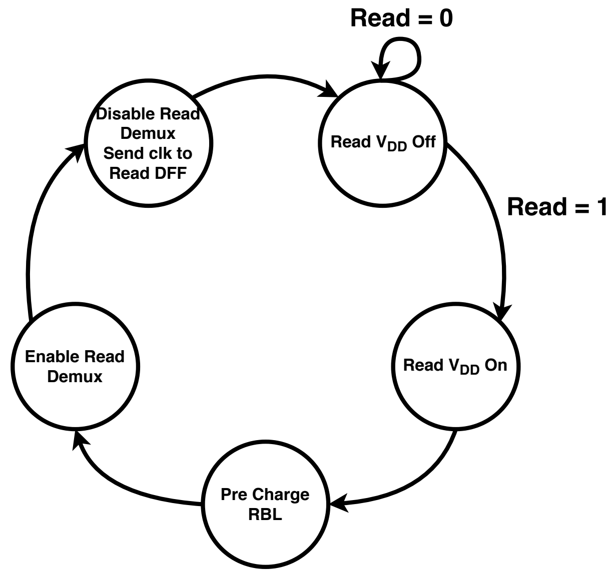


Figure 4.22: State machine for the read operation.

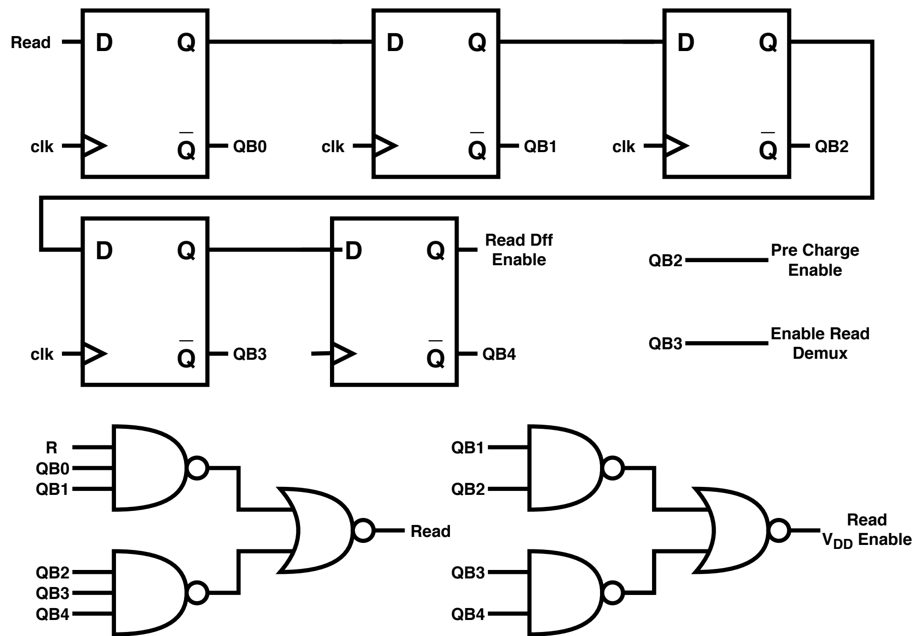


Figure 4.23: State machine implementation with D type flip flops for the read operation.

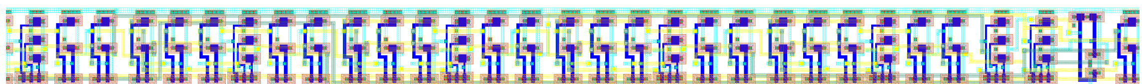


Figure 4.24: Read state machine layout.

The layout for the state machine designed to control the read operation can be seen in figure 4.24. Similarly to the layout of the write state machine, the five chained flip flops were layed out starting on the left, with the remaining logic gates (NAND and NOR) being layed out on the right of the flip flops. As was done with the write state machine, the inputs and outputs can be accessed throughout the layout of the state machine, with the objective of easing the final integration. This state machine layout has a width of 169.99 μm and a height of 10.43 μm, giving an area of 1773 μm^2.

Chapter 5

Integration

The SRAM block and all control circuits are connected in order to create a functioning SRAM. The final layout is made, and the performance of the circuit is analyzed, after the parasitic extraction. The SRAM conceptual block can be seen in 5.1.

To ensure correct functionality, all the components were connected in schematic, and the SRAM assembly was tested using a March C algorithm implemented in a VerilogA script. This verifies the correct functionality of the SRAM, by following the algorithm:

$$\text{March C} : \Updownarrow w0 \Uparrow (r0, w1) \Uparrow (r1, w0) \Updownarrow r0 \Downarrow (ro, w1) \Downarrow (r1, w0) \Updownarrow r0 \tag{5.1}$$

as it checks for errors such as: stuck at 0 and 1, addressing errors, transitions faults and some coupling faults.

The layout for the final assembly of the SRAM memory can be seen in 5.9 (and in figure A.5), and it has 976.67 µm in width and 901.44 µm in height. This results in a total layout area of 0.88041 mm^2. The layout seen in figure 5.9 is color coded for the different components.

After parasitic extraction, the circuit was tested for its resistance to temperature variation. However, not many tests were performed with post-layout simulation, due to the limited time available and the extremely long simulation times in post-layout. It was verified, in post-layout simulation, that the SRAM functions properly in a temperature range from 20°C to 90°C, and the variation of the idle power consumption with temperature can be seen in figure 5.8a. The idle power consumption at 25 ^{o}C is 175 pW. Due to the previously mentioned time constraints, the remaining process corners were only simulated in schematic, which results can be seen in table 5.1.

Monte Carlo simulation is performed to achieve full overview of the most likely behavior of

Table 5.1: SRAM PVT.

Process Variation	tt	ff	ss	snfp	fnsp
Minimum Temperature (^{o}C)	0	-30	30	0	10
Maximum Temperature (^{o}C)	90	110	70	80	80

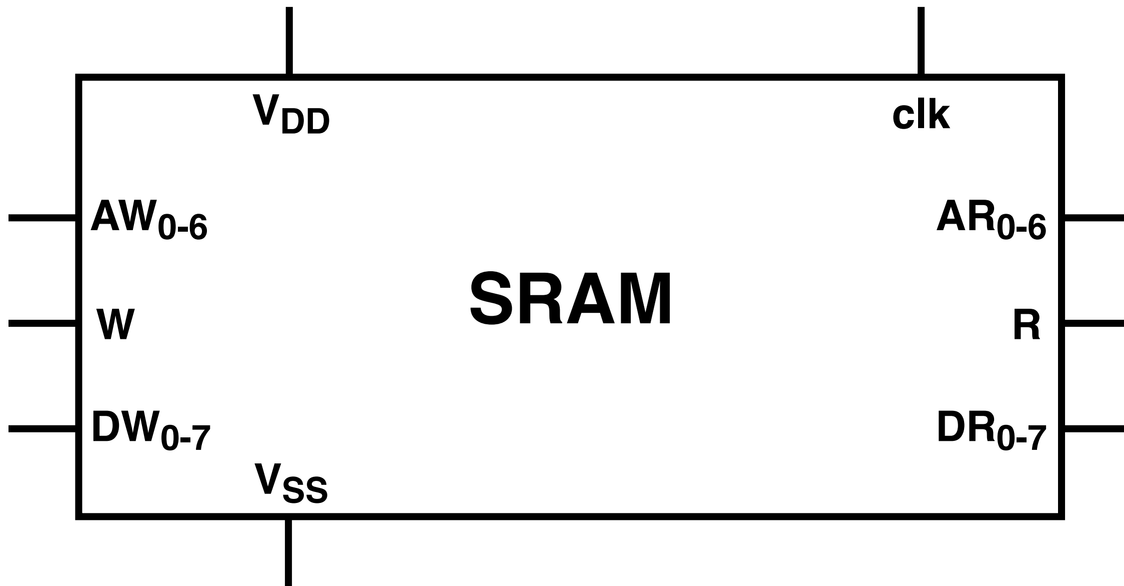


Figure 5.1: SRAM conceptual block.

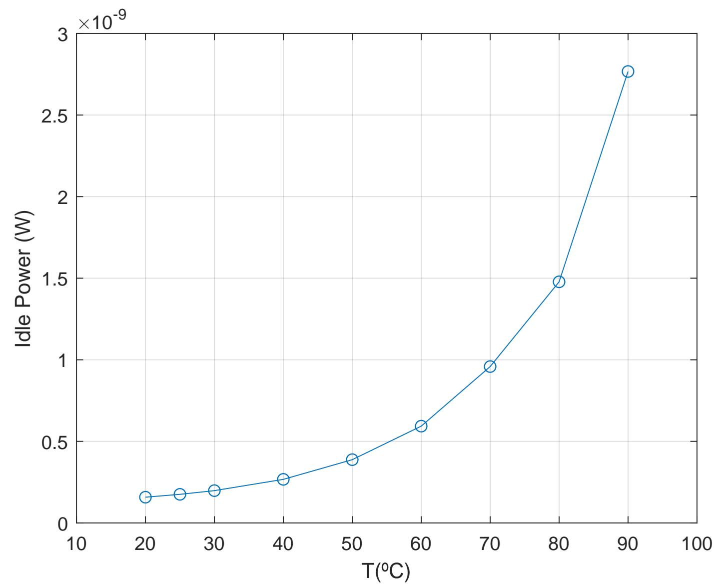


Figure 5.2: SRAM idle power consumption varying with temperature, in the tt process corner.

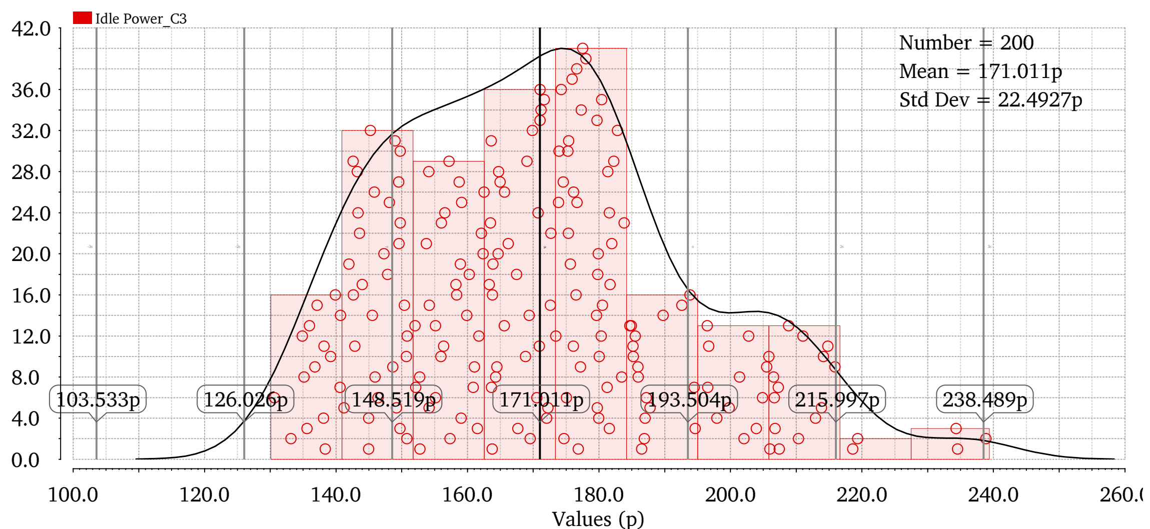


Figure 5.3: Monte Carlo Simulation for the tt corner - Idle power consumption.

the memory. The Monte Carlo simulation was ran 200 times, at 25°C, for the analysis of five parameters: idle power consumption (figure 5.3), output value (figure 5.4) and energy per read access (figure 5.6) when reading a 0, and output value (figure 5.5) and energy per read access (figure 5.7) when reading a 1 .

Figure 5.4 shows the histogram for the voltage output when reading a 0, combined across the 8 bits. The mean voltage output is 204.267 µV, with a standard deviation of 115.364 µV. Looking at figure 5.5, the histogram for the voltage output when reading a 1, combined across the 8 bits can be seen. When reading a 1, the mean voltage output is 199.300 mV, with a standard deviation of 310,296 µV. Analyzing the two histograms shows that, despite the small variations in process and mismatch of the transistors, the memory can function correctly, when powered at 200 mV and with a clock signal with a periodicity of 128 Hz.

The mean value for the idle power consumption of the SRAM memory is 171.011 pW, with a standard deviation of 22.4927 pW. This shows that small variations of the process and mismatch of the transistors in the circuit have a moderate impact on the power consumption of the circuit, while not impairing the ability of the memory to function properly.

Figure 5.8b shows the variation of the idle power consumption with the power supply voltage, which as expected shows exponential growth. In order to comply with the goal of the power consumption, which has to be under 10 nW in idle, V_{DD} can be raised up to 1 V.

Regarding energy per read access, its values differ whether reading a 0 or a 1. When reading a 0, the mean value is 10.4535 pJ with a standard deviation of 1.83486 pJ, as it can be seen in figure 5.6; when reading a 1, the mean value is 9.87884 pJ with a standard deviation of 1.65150 pJ, as seen in figure 5.7. In post-layout simulation, with the tt process corner and 25°C, the energy per read access is: 11.17 pJ when reading a 0; and 10.45 pJ when reading a 1.

In the Monte Carlo simulations, the parameters idle power and energy per read access are within the 3σ range. The read voltage outputs go past the 3σ range, but nevertheless comply with

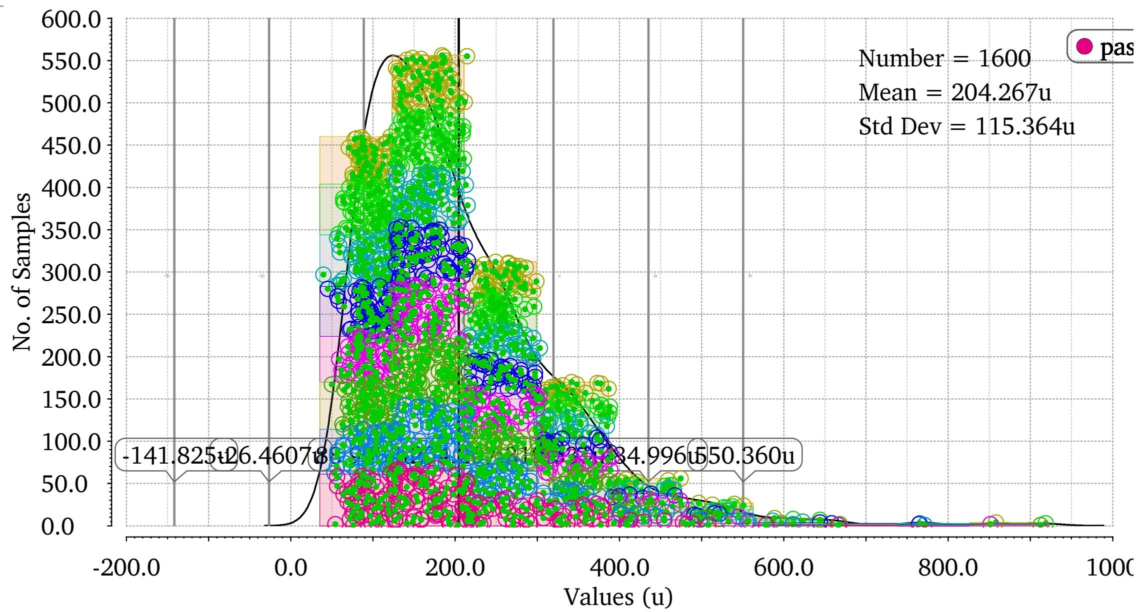


Figure 5.4: Monte Carlo Simulation for the tt corner - output when reading a 0.

the specification given, and do not compromise the SRAM's ability to function properly.

The table seen in 5.2 summarizes the comparison between the SRAM designed in this dissertation with work seen in the literature. This SRAM has lower idle power consumption than all previous works, with the exception of the work presented in [43], to the author's best knowledge. This SRAM perfectly illustrates the trade-off between area and power, has it shows one of the best idle power consumptions but at the expense of a greater area.

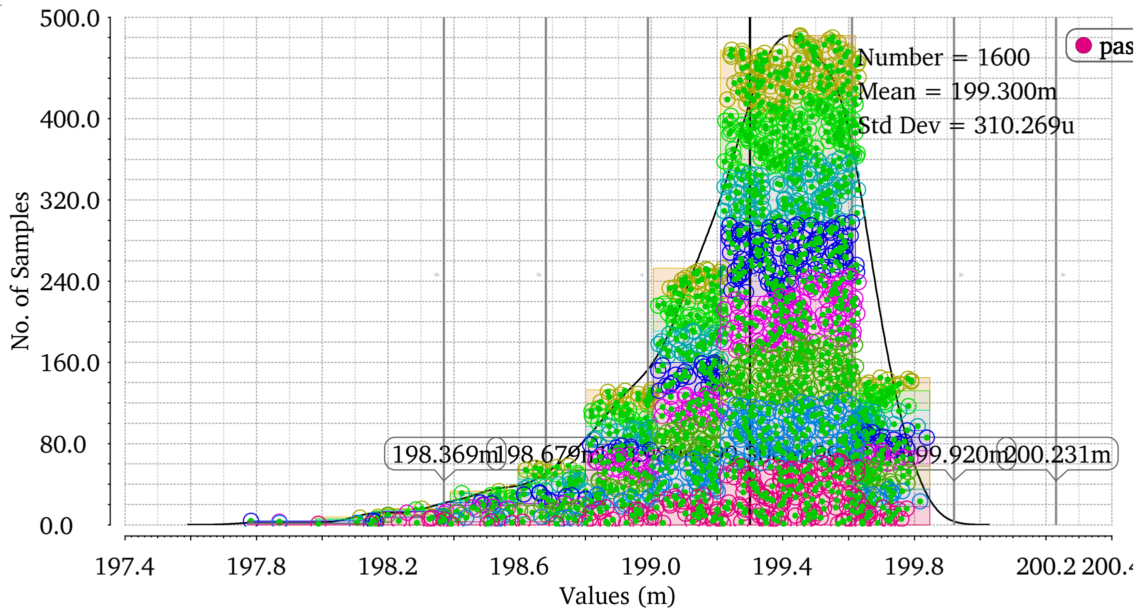


Figure 5.5: Monte Carlo Simulation for the tt corner - output when reading a 1.

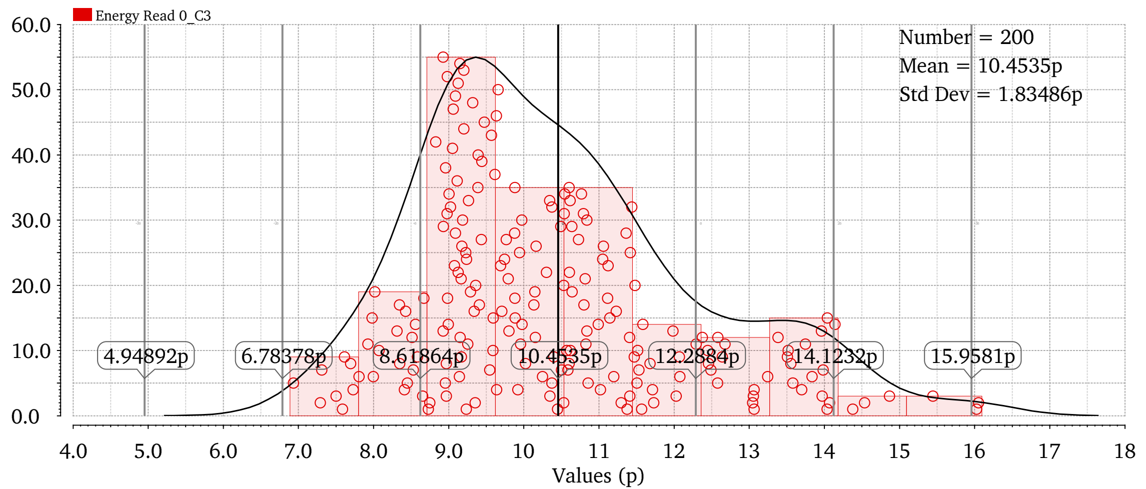


Figure 5.6: Monte Carlo Simulation for the tt corner - energy per read access when reading a 0.

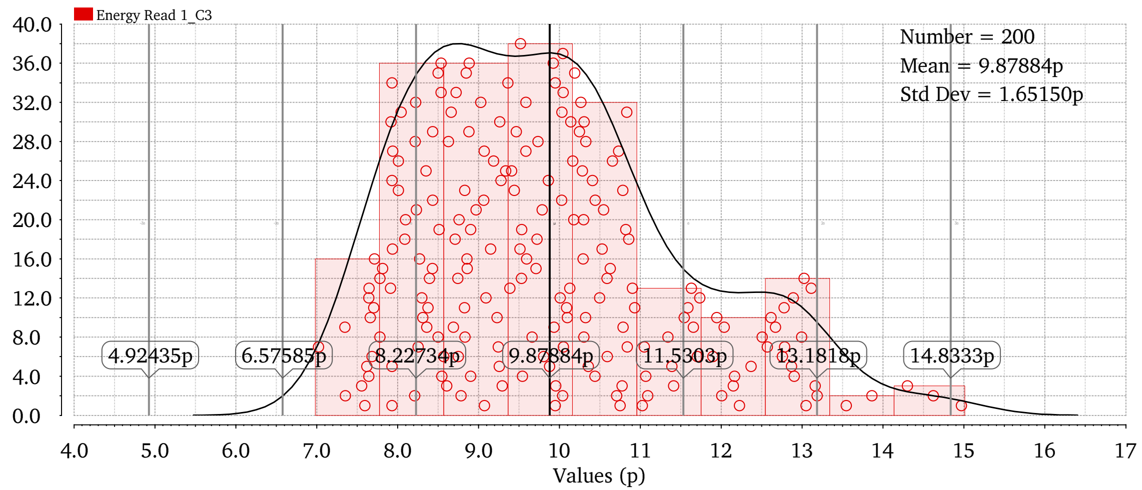
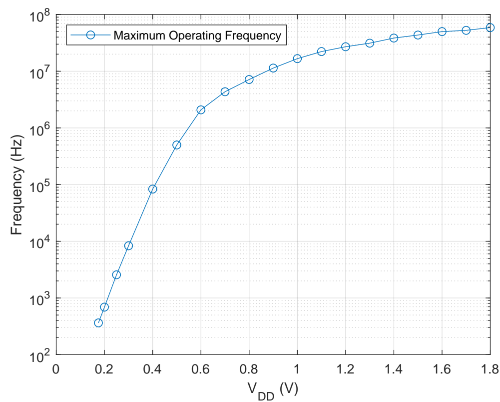
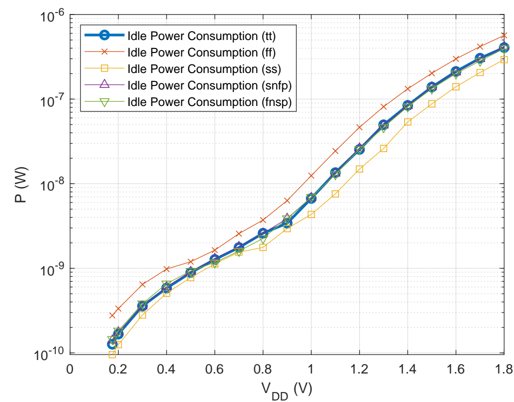


Figure 5.7: Monte Carlo Simulation for the tt corner - energy per read access when reading a 1.



(a) Maximum functioning frequency.



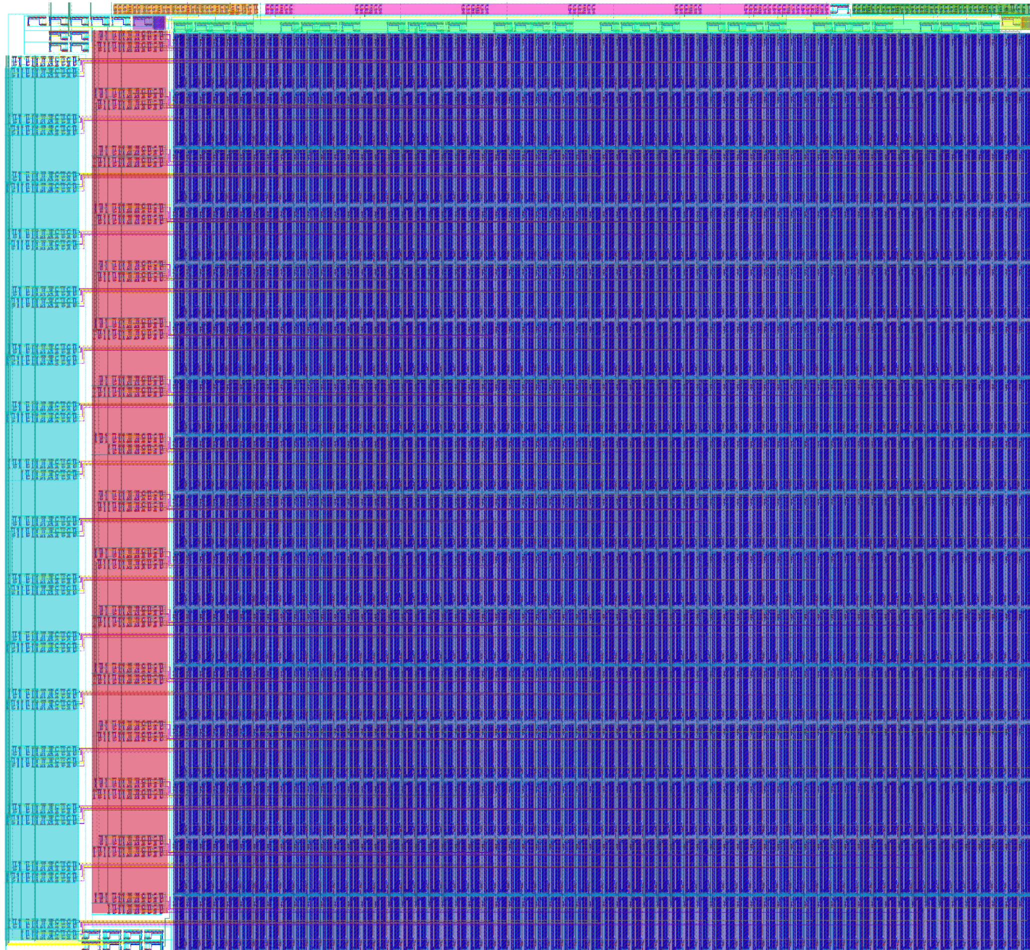
(b) Idle power consumption.

Figure 5.8: Variation of SRAM metrics with V_{DD}, for the tt corner (simulated).

Table 5.2: Comparison of the SRAM in this dissertation to some works already developed

Work	Bitcell	Technology (nm)	Area (bitcell (μm^2)/ /total (mm^2))	Memory Size (kbits)	V_{DD} (mV)	Frequency (Hz)	Idle Power (pW/bit)	E/R* (pJ)	Hold SNM (mV)	Write SNM (mV)	Read SNM (mV)
This	10T	180	725.5 / 0.88	1	200	128	0.1709	10.85	52.8	100.9	58.7
[44]	9T	40	1.052 / NA	9	400	$1 \cdot 10^9$	4.34 - 6.51	0.210	NA	NA	NA
[43]	10T	180	17.48 / NA	24	350	3500	0.00185	NA	NA	NA	NA
[12]	8T	90	3.9 / NA	1	300	NA	6670	NA	227.2	70.33	89.57
[8]	10T	65	NA / NA	256	400	475000	12.51	NA	NA	NA	NA
[31]	12T	40	NA / 0.17688	4	350	$11.5 \cdot 10^9$	5371	1,8	NA	156	119

* Energy per read access



Caption:

- Write State Machine;
- Read output flip flops;
- Read State Machine;
- Write Power Gate;
- Bitline drivers and read circuitry (pre charger and sensor inverter), interleaved;
- Read Power Gater;
- Read Demultiplexer;
- Write Demultiplexer;
- SRAM blocks;

All other devices are input buffers.

Figure 5.9: SRAM memory layout.

Chapter 6

Conclusion

Following the latest trends in consumer electronics, the IoT segment is always pushing for new ways to extend the battery life of its interconnected nodes. Thus, after studying the base structure of the IoT node, the SRAM was identified as one of the crucial elements to be optimized in the core of a node. This element is responsible for the data keeping while operating in an IDLE mode, as such, maintaining its power consumption to the lowest possible level is a key factor when trying to extend the battery life of the IoT peripherals.

Due to the challenges of the subthreshold operation region, consisting mainly of higher propagation delay of the signals and lower drain currents, the traditional 6 transistor SRAM cannot be used effectively in the weak inversion region [2, 8]. For this reason, a number of different SRAM bitcell designs have been proposed, ranging from 7T designs to 12T or even 14T, and also latch or standard cell based [26, 12, 27, 8, 30, 31, 35]. After an analysis of the available bitcell architectures, the 10T [8] and an 8T [19] were chosen.

Resorting that, in this thesis, the intrinsic study of an SRAM was done with focus on the bitcell optimization for the 180 nm CMOS process used. First a bitcell of 10 transistors, that can be seen in figure 3.2, was developed and optimized, aiming for the lowest possible power consumption while guaranteeing correct functioning of the bitcell (in terms of stored and read value). The bitcell suffered six design iterations, where the supply voltage and transistor type (LVT and HVT) were changed, and even reverse body-bias was tried. However the final bitcell architecture, that can be seen in figure 3.2, used all LVT transistors with power gating on the read port and a supply voltage of 200 mV, resulting in a bitcell idle power consumption of 141 fW, with a hold SNM 52.8mV; write SNM of 100.9 mV; and read SNM of 58.7 mV. The 8T architecture, seen in figure 3.7, was optimized resulting in a bitcell with a power consumption of 288 fW and a larger gate area than the 10T bitcell, reasons which caused the 8T bitcell to not be implemented in the final SRAM. The 10T bitcell allows the SRAM to be dual ported, meaning that it can be both written to and read from at the same time, and the control circuitry needs to be designed to allow this functionality.

The memory block was layed out in a modular aproach: first, the single bitcell was layed out, achieving a size of 13.54 μm by 53.56 μm; eight bitcells were joined horizontally, to form a byte;

the 128 bytes needed were layed out in a grid composed of 8 bytes (in a row) by 16 bytes (in a column), as this was the only arrangement that gave a rectangular layout and fit in a chip of 1.2 mm by 1.2 mm.

The SRAM needs control circuitry, that allow it to coordinate the several steps required for each write and read operation, and allow the data to be saved to/read from the correct word in the memory. In this SRAM, the control circuits required are: two **address decoders**, implemented with 7 to 128 demultiplexer, one for the write and the other for the read operation; **bitline drivers**, allowing for the differential writing that the SRAM needs; **read circuits**, composed by bitline prechargers and sense inverters, that aid in the read operation, increasing its reliability and robustness; **power gating** circuits, used to lower the idle power of the SRAM by completely shutting off parts of the circuit that are not needed; and **state machines**, controlling the write and read operations. The demultiplexers and flip flops needed are built with logic gates, which were optimized for power consumption and reliability. The other components needed (bitline drivers, pre charger and power gater) were also optimized, always with the lowest power consumption possible in mind.

The layout of the logic gates used a standard cell aproach: all logic gates have the same height, with V_{DD} across the top and ground across the bottom, the inputs placed on the left and the outputs on the right. These were used to layout the demultiplexers, with the spacing requirements that the SRAM block imposed determining the final layout this component. After all components were layed out, the final assembly of the SRAM was performed, resulting in the layout that can be seen in figures 5.9 and A.5.

The layout parasitics were extracted and post-layout simulations were performed, to verify the operation of the SRAM. Due to time constraints, and taking into consideration that a single post-layout simulation lasted 6 hours, most metrics were calculated using schematic simulations. The Monte Carlo simulations were ran at 25 ^{o}C, and proved that, although the idle power consumption and energy per read access vary with process and transistor mismatch, the proper operation of the SRAM is not affected.

In conclusion, a fully functioning SRAM with lower than proposed idle power consumption was achieved. The SRAM memory operates at 200 mV of supply voltage, 128 Hz of clock period, in a temperature range from 20 ^{o}C to 90 ^{o}C, while consuming 175 pW of power at idle, and an average energy per read access of 10.85 pJ (per byte).

6.1 Future Work

Future work passes by adding a serializer and deserializer, endowing the SRAM with serial communication capabilities and allowing it to exchange data with a microcontroller and a sensor. IC fabrication and testing should also be done.

The layout of the RBL should be tweaked to allow functioning at lower temperature and slower process corners. Tests in order to decrease the number of cycles to write or read should be done, in order to reduce energy per read access and overall power consumption.

Appendix A

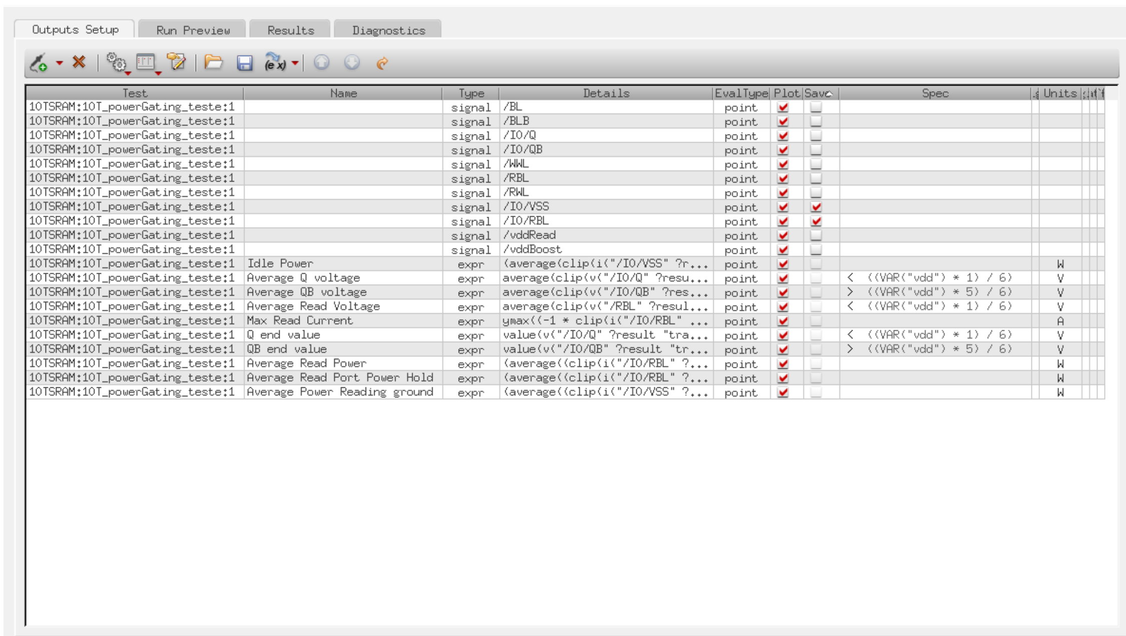


Figure A.1: Cadence Virtuoso ADE XL Outputs Setup tab for the 10T SRAM bitcell optimization.

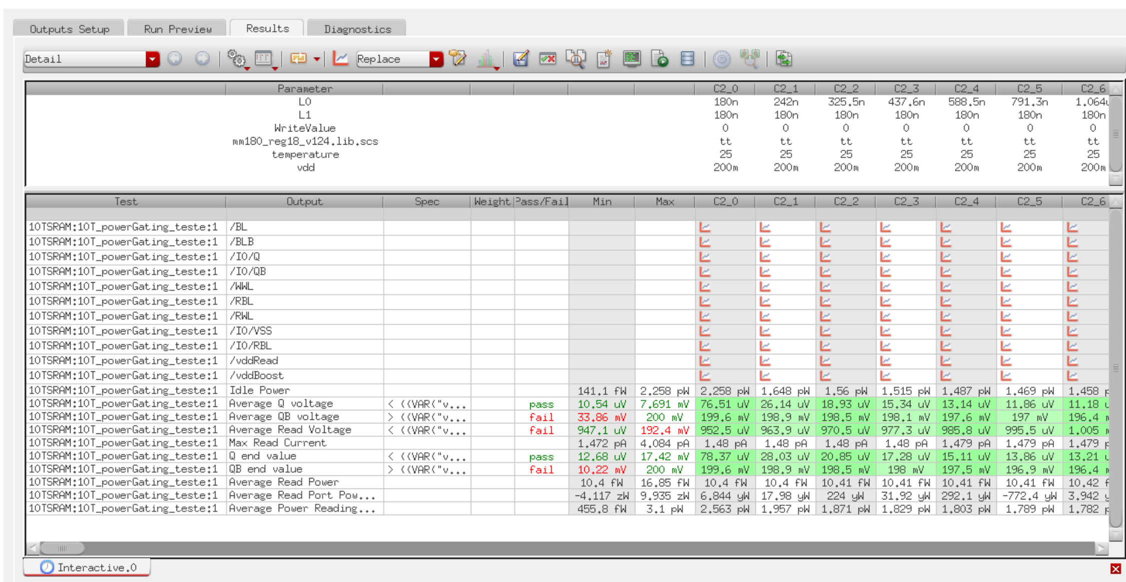


Figure A.2: Cadence Virtuoso ADE XL Results tab for the 10T SRAM bitcell optimization.

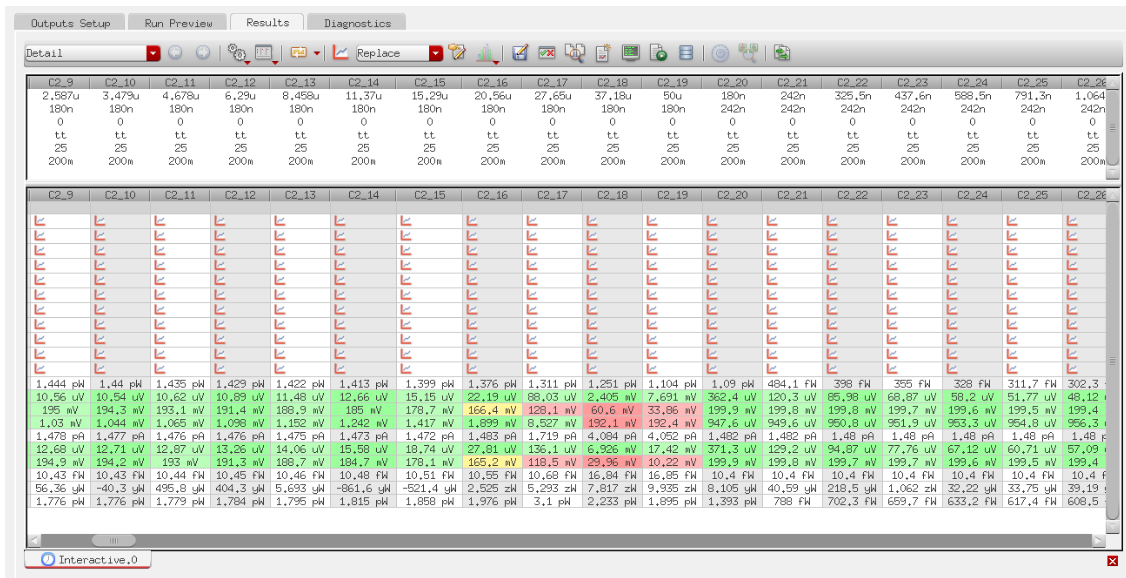


Figure A.3: Cadence Virtuoso ADE XL Results tab for the 10T SRAM bitcell optimization (2).

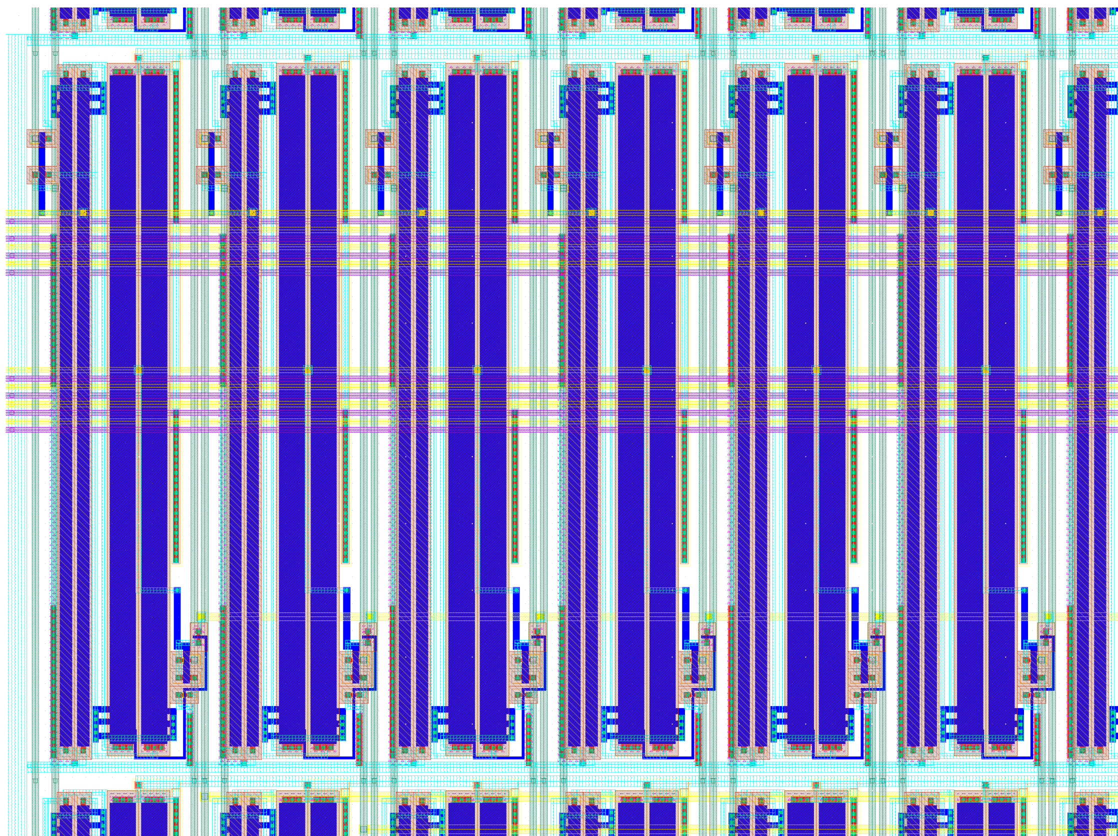


Figure A.4: 10T 128 byte layout - Word Line Detail.

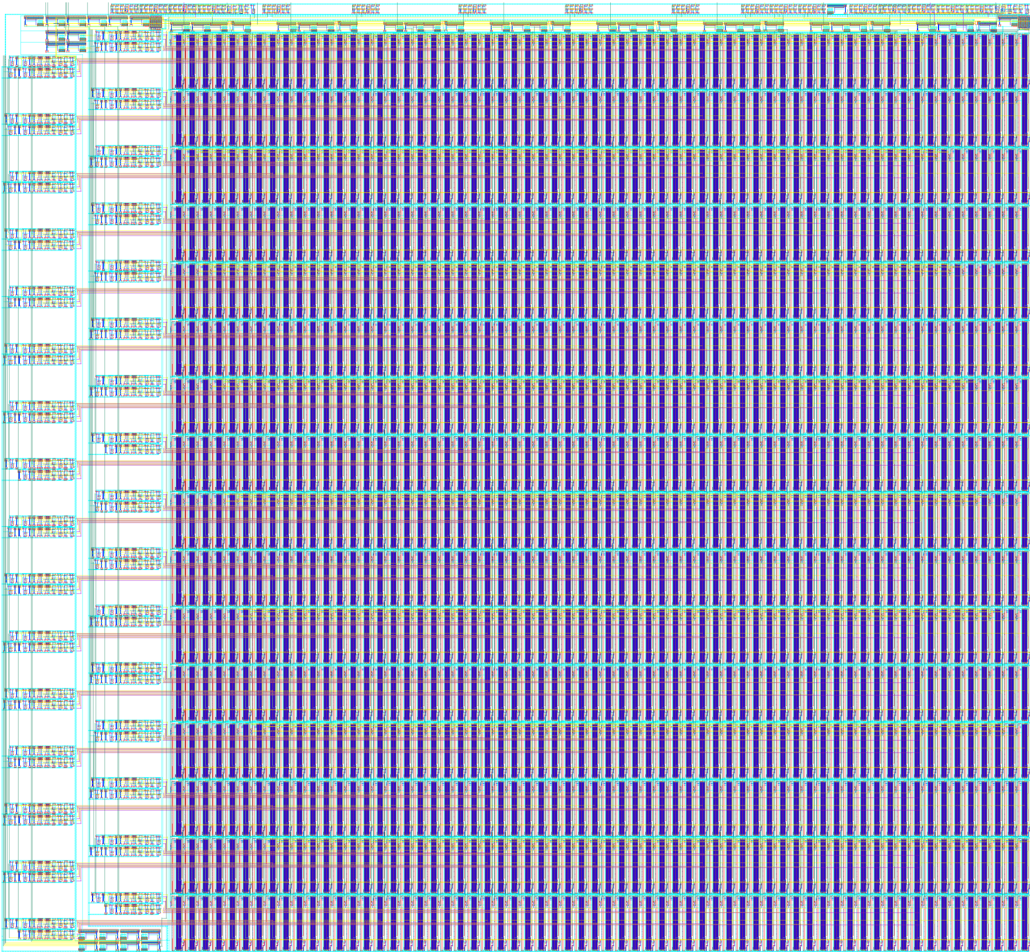


Figure A.5: Layout of the final SRAM memory, non annotated.

References

[1] Ashton K. That 'Internet of Things' Thing. *RFID Journal*, page 4986, 2009.

[2] Massimo Alioto. *Enabling the Internet of Things - From Integrated Circuits to Integrated Systems*. 2017.

[3] Sharon Gaudin. Get ready to live in a trillion-device world (with video), Sep 2015.

[4] Ovidiu Vermesan and Peter Friess. *Internet of Things – From Research and Innovation to Market Deployment*. 2014.

[5] Yen-po Chen, Student Member, Dongsuk Jeon, Student Member, Yoonmyung Lee, Yejoong Kim, Student Member, Zhiyoong Foo, Inhee Lee, Student Member, Nicholas B Langhals, Grant Kruger, Hakan Oral, Omer Berenfeld, Zhengya Zhang, David Blaauw, and Dennis Sylvester. An Injectable 64 nW ECG Mixed-Signal SoC in 65 nm for Arrhythmia Monitoring. 50(1):375–390, 2015.

[6] Farah Yahya, Christopher Lukas, and Benton Calhoun. A Top-Down Approach to Building Battery-Less Self-Powered Systems for the Internet-of-Things. *Journal of Low Power Electronics and Applications*, 8(2):21, 2018.

[7] Massimo Alioto. Ultra-low power VLSI circuit design demystified and explained: A tutorial. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 59(1):3–29, 2012.

[8] Benton Highsmith Calhoun and Anantha P. Chandrakasan. A 256-kb 65-nm Sub-threshold SRAM Design for Ultra-Low-Voltage Operation. *IEEE Journal of Solid-State Circuits*, 42(3):680–688, 2007.

[9] Hassan Afzali-Kusha, Alireza Shafaei, and Massoud Pedram. A 125mV 2ns-access-time 16Kb SRAM design based on a 6T hybrid TFET-FinFET cell. *Proceedings - International Symposium on Quality Electronic Design, ISQED*, 2018-March:280–285, 2018.

[10] Shang Lin Wu, Kuang Yu Li, Po Tsang Huang, Wei Hwang, Ming Hsien Tu, Sheng Chi Lung, Wei Sheng Peng, Huan Shun Huang, Kuen DI Lee, Yung Shin Kao, and Ching Te Chuang. A 0.5-V 28-nm 256-kb Mini-Array Based 6T SRAM with Vtrip-Tracking Write-Assist. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 64(7):1791–1802, 2017.

[11] Harsh N. Patel, Farah B. Yahya, and Benton H. Calhoun. Subthreshold SRAM: Challenges, Design Decisions, and Solutions. pages 321–324, 2017.

[12] C. B. Kushwah and S. K. Vishvakarma. A Single-Ended with Dynamic Feedback Control 8T Subthreshold SRAM Cell. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 24(1):373–377, 2016.

[13] Roghayeh Saeidi, Student Member, Mohammad Sharifkhani, and Khosrow Hajsadeghi. A Subthreshold Symmetric SRAM Cell With High Read Stability. *Circuits and Systems II: ...*, 61(1):26–30, 2014.

[14] Robert Giterman, Maoz Vicentowski, Itamar Levi, Yoav Weizman, Osnat Keren, and Alexander Fish. Leakage Power Attack-Resilient Symmetrical 8T SRAM Cell. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pages 1–5, 2018.

[15] Amit Kazimirsky, Adam Teman, Noa Edri, and Alexander Fish. A 0.65-V, 500-MHz Integrated Dynamic and Static RAM for Error Tolerant Applications. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 25(9):2411–2418, 2017.

[16] Tony Tae Hyoung Kim and Ngoc Le Ba. Design of a Temperature-Aware Low-Voltage SRAM With Self-Adjustable Sensing Margin Enhancement for High-Temperature Applications up to 300 °C. *IEEE Journal of Solid-State Circuits*, 49(11):2534–2546, 2014.

[17] Mahmood Khayatzadeh, Student Member, and Yong Lian. Average-8T Differential-Sensing Subthreshold SRAM With Bit Interleaving and 1k Bits Per Bitline. 22(5):971–982, 2014.

[18] Farah Yahya, Harsh Patel, James Boley, Arijit Banerjee, and Benton Calhoun. A Sub-Threshold 8T SRAM Macro with 12.29 nW/KB Standby Power and 6.24 pJ/access for Battery-Less IoT SoCs. *Journal of Low Power Electronics and Applications*, 6(2):8, 2016.

[19] Pooran Singh and Santosh Kumar Vishvakarma. Ultra-Low Power High Stability 8T SRAM for Application in Object Tracking System. *IEEE Access*, 6:2279–2290, 2017.

[20] Harsh N. Patel, Farah B. Yahya, and Benton H. Calhoun. Optimizing SRAM Bitcell Reliability and Energy for IoT Applications. *Proceedings - International Symposium on Quality Electronic Design, ISQED*, 2016-May:12–17, 2016.

[21] James Boley, Jiajing Wang, and Benton H. Calhoun. Analyzing Sub-Threshold Bitcell Topologies and the Effects of Assist Methods on SRAM VMIN. *Journal of Low Power Electronics and*, pages 143–154, 2012.

[22] Sylvain Clerc, Fady Abouzeid, Gilles Gasiot, David Gauthier, and Philippe Roche. A 65nm SRAM achieving 250mV retention and 350mV, 1MHz, 55fJ/bit access energy, with bit-interleaved radiation Soft Error tolerance. *European Solid-State Circuits Conference*, pages 313–316, 2012.

[23] P. Upadhyay, Prasanta Kundu, R. Kar, D. Mandal, and S. P. Ghoshal. A Novel 10T SRAM Cell with Low Power Dissipation in Active and Sleep Mode for Write Operation. *2014 11th Int. Joint Conf. on Computer Science and Software Engineering: "Human Factors in Computer Science and Software Engineering" - e-Science and High Performance Computing: eHPC, JCSSE 2014*, 66:206–211, 2014.

[24] P. Shiny Grace and N. M. Sivamangai. Design of 10T SRAM Cell for High SNM and Low Power. *Proceedings of the 3rd International Conference on Devices, Circuits and Systems, ICDCS 2016*, pages 281–285, 2016.

[25] Sylvain Clerc, Fady Abouzeid, Gilles Gasiot, David Gauthier, Dimitri Soussan, Philippe Roche, and J Monnet. A 0.32V, 55fJ per bit access energy, CMOS 65nm bit-interleaved SRAM with radiation Soft Error tolerance. pages 12–15, 2012.

[26] Yuan Lin Yeoh, Bo Wang, Xiangyao Yu, and Tony T. Kim. A 0.4V 7T SRAM with Write Through Virtual Ground and Ultra-fine Grain Power Gating Switches. *Proceedings - IEEE International Symposium on Circuits and Systems*, pages 3030–3033, 2013.

[27] Soumitra Pal and Aminul Islam. 9-T SRAM Cell for Reliable Ultralow-Power Applications and Solving Multibit Soft-Error Issue. *IEEE Transactions on Device and Materials Reliability*, 16(2):172–182, 2016.

[28] S. Karthika and N.M. Sivamangai. Power Analysis of Bit interleaving 9T SRAM array. *Proceedings of the 3rd International Conference on Devices, Circuits and Systems, ICDCS 2016*, pages 275–280, 2016.

[29] Subthreshold Sram. A 0.325 V, 600-kHz, 40-nm 72-kb 9T Subthreshold SRAM with Aligned Boosted Write Wordline and Negative Write Bitline Write-Assist. 23(5):958–962, 2015.

[30] Sayeed Ahmad, Belal Iqbal, Naushad Alam, and Mohd Hasan. Low Leakage Fully Half-Select-Free Robust SRAM Cells With BTI Reliability Analysis. *IEEE Transactions on Device and Materials Reliability*, 18(3):337–349, 2018.

[31] Yi-wei Chiu, Yu-hao Hu, Ming-hsien Tu, Jun-kai Zhao, Yuan-hua Chu, Shyh-jye Jou, and Senior Member. 40 nm Bit-Interleaving 12T Subthreshold SRAM With Data-Aware Write-Assist. *Ieee Transactions on Circuits and Systems—I*, 61(9):2578–2585, 2014.

[32] Babak Mohammadi, Oskar Andersson, Pascal Meinerzhagen, Yasser Sherazi, Andreas Burg, and Joachim Neves Rodrigues. A 0.28-0.8V 320fW D-latch for Sub-VT Memories in 65nm CMOS Babak. *2014 IEEE Faible Tension Faible Consommation, FTFC 2014*, pages 1–4, 2014.

[33] Oskar Andersson, Babak Mohammadi, and Joachim Neves Rodrigues. A Wide-Operating Range Standard-Cell Based Memory in 28nm FD-SOI. pages 28–29, 2016.

[34] Information Technology. A 35 fJ/bit-access Sub-VT Memory Using a Dual-Bit Area-Optimized Standard-cell in 65 nm CMOS Oskar. *Essderc Esscirc*, pages 243–246, 2014.

[35] Kai Robert Liknes. Ultra Low Leakage Memory. (June), 2016.

[36] Jawar Singh, Saraju P. Mohanty, and Dhiraj K. Pradhan. *Robust SRAM Designs and Analysis*. Number Mm. 2013.

[37] Neil H E Weste and David Money Harris. *CMOS VLSI Design : A Circuit and Systems Perspective*, volume 53. 2011.

[38] A P Chandrakasan and R W Brodersen. Minimizing power consumption in CMOS circuits. *Proc of the IEEE*, 83(4):498–523, 1995.

[39] Christian C Enz and Eric A Vittoz. 1995 Enz An analytical MOS transistor model valid in all regions of operation and dedicated to low-voltage and low-current applications. 14:1–32, 2001.

[40] Benton Highsmith Calhoun and Anantha P Chandrakasan. *Sub-threshold Design for Ultra Low-Power Systems*. 2006.

[41] Basab Datta and Wayne Burleson. Temperature effects on energy optimization in sub-threshold circuit design. *Proceedings of the 10th International Symposium on Quality Electronic Design, ISQED 2009*, pages 680–685, 2009.

[42] Armin Tajalli and Yusuf Leblebici. *Extreme Low-Power Mixed Signal IC Design*. 2010.

[43] Daeyeon Kim, Gregory Chen, Matthew Fojtik, Mingoo Seok, David Blaauw, and Dennis Sylvester. A 1.85fW/bit ultra low leakage 10T SRAM with speed compensation scheme. *Proceedings - IEEE International Symposium on Circuits and Systems*, pages 69–72, 2011.

[44] Adam Teman, Lidor Pergament, Omer Cohen, and Alexander Fish. A 250 mV 8 kb 40 nm Ultra-Low Power 9T Supply Feedback SRAM (SF-SRAM). *IEEE Journal of Solid-State Circuits*, 46(11):2713–2726, 2011.