A Physical-Unclonable-Function Circuit based on Cells Composed of Diode-Connected Transistors

A Thesis

submitted to the designated by the Assembly
of the Department of Computer Science and Engineering
Examination Committee

by

Fani Moka

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN DATA AND COMPUTER SYSTEMS ENGINEERING

WITH SPECIALIZATION
IN ADVANCED COMPUTER SYSTEMS

University of Ioannina
School of Engineering
Ioannina 2023

Examining Committee:

- **Georgios Tsiatouchas,** Professor, Computer Science and Engineering Department, University of Ioannina (Advisor)
- **Aristides Efthymiou,** Assistant Professor, Computer Science and Engineering Department, University of Ioannina
- **Georgia Tsirimokou,** Assistant Professor, Computer Science and Engineering Department, University of Ioannina

TABLE OF CONTENTS

List of Figures

List of Tables

ABSTRACT

Fani Moka, M.Sc. in Data and Computer Systems Engineering, Department of Computer Science and Engineering, School of Engineering, University of Ioannina, Greece, October 2023

Physical-Unclonable-Function Circuit based on Cells Composed of Diode-Connected Transistors

Advisor: Georgios Tsiatouchas, Professor

In this thesis a circuit to be used as a Physically Unclonable Function (PUF) is presented. Physically Unclonable Functions are hardware based cryptographic primitives, which can be used for device authentication, secure key generation, and other security related operations.

The proposed circuit is based on an n×m array design (n rows and m columns), where each cell of the array consists of a PMOS block and an NMOS block. Furthermore, each block comprises of two serially connected transistors, a regular one whose role is to act as a switch and a diode-connected one whose role is to act as a non-linear resistor. The PMOS block is connected between the power supply V_{DD} and the corresponding column bitline where it belongs in the array, while the NMOS block is connected between the ground and the corresponding column bitline. By activating the PMOS block of a cell concurrently with the NMOS block of another cell in the same bitline (turning on the corresponding switch transistors), a voltage divider is composed, and a voltage is developed on the pertinent bitline. This can be extended to the activation of more than one PMOS and NMOS blocks in the same bitline, providing the ability to compose a strong PUF. Due to process variations, the voltage of any two bitlines will not be identical, either in the same chip or among

different chips. Thus, comparing the voltage levels of pairs of bitlines in the array a unique PUF response is generated.

The PUF circuit was designed in a commercial 90nm CMOS technology using the CADENCE platform. SPICE level Monte Carlo simulations (10.000 runs in each case) were conducted (exploiting the statistical models of the used technology) to analyze its behavior and validate its performance characteristics, also considering temperature and supply voltage variations. The proposed circuit presents a uniformity of 49.730%, a uniqueness of 50.003%, and a worst-case reliability of 97.925%. Comparisons of the new PUF design with state-of-the art array-based PUF designs in the literature, accentuate its efficiency.

Εκτεταμενη Περιληψη

Φανή Μόκα, Δ.Μ.Σ. στην Μηχανική Δεδομένων και Υπολογιστικών Συστημάτων, Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πολυτεχνική Σχολή, Πανεπιστήμιο Ιωαννίνων, Ελλάδα, Οκτώβριος 2023
Κύκλωμα Φυσικά-Μη Κλωνοποιήσιμης-Συνάρτησης βασισμένο σε Κελιά Αποτελούμενα από Διοδικά-Συνδεδεμένα Τρανζίστορ
Ερευνητικός σύμβουλος: Γεώργιος Τσιατούχας, Καθηγητής

Το επίκεντρο της παρούσας μεταπτυχιακής διπλωματικής εργασίας είναι η πρόταση ενός κυκλώματος, το οποίο δίνει την δυνατότητα να χρησιμοποιηθεί ως Φυσικώς-Μη Κλωνοποιήσιμη-Συνάρτηση (PUF). Οι συναρτήσεις αυτές αποτελούν φυσικές οντότητες, εναλλακτικές των συνηθισμένων αλγορίθμων που υλοποιούνται για την επίτευξη λειτουργιών σχετιζόμενες με την ασφάλεια των συστημάτων, όπως η ταυτοποίηση συσκευών, η δημιουργία ασφαλών κλειδιών, κ.ά.

Το κύκλωμα που προτείνεται είναι ένας πίνακας nxm (n γραμμές και m στήλες), όπου κάθε κελί του πίνακα αποτελείται από ένα PMOS μπλοκ και ένα NMOS μπλοκ. Κάθε μπλοκ αποτελείται από δύο τρανζίστορ συνδεδεμένα σε σειρά, ένα απλό τρανζίστορ που λειτουργεί ως διακόπτης, και ένα δεύτερο, διοδικά συνδεδεμένο, που λειτουργεί ως μη γραμμική αντίσταση. Το PMOS μπλοκ συνδέεται μεταξύ της τροφοδοσίας V_{DD} και της αντίστοιχης bitline που αντιστοιχεί στην στήλη στην οποία ανήκει. Το NMOS μπλοκ συνδέεται μεταξύ της bitline και της γείωσης. Ενεργοποιώντας το PMOS μπλοκ ενός κελιού ταυτόχρονα με το NMOS block ενός άλλου κελιού στην ίδια bitline (ενεργοποιώντας τα αντίστοιχα τρανζίστορ που λειτουργούν ως διακόπτες), σχηματίζεται ένας διαιρέτης τάσης, και αναπτύσσεται μια τάση στη σχετική bitline. Αυτό μπορεί να επεκταθεί με την ενεργοποίηση περισσοτέρων του ενός PMOS και NMOS μπλοκ στην ίδια bitline, παρέχοντας την

δυνατότητα επίτευξης ενός ισχυρού κυκλώματος-PUF. Εξαιτίας των κατασκευαστικών διακυμάνσεων κατά τα στάδια κατασκευής, η τάση δύο bitlines δεν αναμένεται να είναι η ίδια, ούτε όταν αυτές βρίσκονται στο ίδιο ολοκληρωμένο κύκλωμα, ούτε όταν είναι σε διαφορετικά ολοκληρωμένα κυκλώματα. Συνεπώς, συγκρίνοντας τις τιμές της τάσης μεταξύ ζευγαριών από bitlines στον πίνακα δημιουργείται μία μοναδική απόκριση του κυκλώματος.

Το κύκλωμα PUF σχεδιάστηκε σε εμπορική 90nm τεχνολογία CMOS χρησιμοποιώντας την πλατφόρμα της CADENCE. Εκτελέστηκαν SPICE Monte Carlo προσομοιώσεις (με 10.000 επαναλήψεις σε κάθε περίπτωση, και χρησιμοποιώντας τα στατιστικά μοντέλα της τεχνολογίας) για την ανάλυση της συμπεριφοράς του και την επαλήθευση των χαρακτηριστικών της απόδοσής του, λαμβάνοντας υπόψη διακυμάνσεις στην θερμοκρασία και την τάση τροφοδοσίας. Το κύκλωμα αναδεικνύει ομοιομορφία ίση με 49.730%, μοναδικότητα ίση με 50.003%, ενώ η χειρότερη αξιοπιστία του είναι ίση με 97.925%. Οι συγκρίσεις του προτεινόμενου κυκλώματος με αντίστοιχα υψηλής στάθμης κυκλώματα της βιβλιογραφίας αναδεικνύουν τις εξαιρετικές του επιδόσεις.

CHAPTER 1

INTRODUCTION

1.1 Goals

The focus of the present thesis is to investigate the possibility of diode-connected transistors, with the architecture put forth in the third chapter, as a viable physically unclonable function, or, in short, PUF. A detailed introduction to PUFs will be given in the following chapter, but, in essence, these electronic circuits hold the capacity to function as robust, hardware-based solutions for security-related issues.

In the present thesis it will be attempted to develop a circuit design and simulate its behaviour under different temperatures, and supply voltage variations. For the purpose of the simulations, SPICE simulations were used. The thesis finally aims to provide a comprehensive comparison between the proposed circuit design and other state-of-the-art array-based designs in the domains of power consumption, area demands, and the metrics that are commonly used for assessing the behaviour of the PUF, namely uniqueness, uniformity, and reliability.

1.2 Structure of the Thesis

The thesis consists of five chapters. In the second chapter an introductory exploration into the domain of Physically Unclonable Functions will be given. The third chapter is dedicated to presenting the proposed circuit design and describing the working principle behind it. The fourth chapter will hold the results from the simulations and the comparison between other similar designs. Lastly, in the fifth and final chapter the key findings will be summarized, and insights into avenues for potential future research will be offered.

CHAPTER 2

PHYSICALLY UNCLONABLE FUNCTIONS

In this chapter, the primary focus is to provide a comprehensive overview of physically unclonable functions (PUFs) and their significance in the realm of hardware-based security. The chapter begins by delving into a thorough explanation of what PUFs are and how the necessity for their existence came about. The motivation behind developing PUFs lies in addressing the escalating concerns regarding hardware-related security threats, such as counterfeiting, unauthorized duplication, tampering, and other potential attacks that pose risks to sensitive information and intellectual property.

To thoroughly assess the effectiveness of PUFs, the chapter addresses the metrics used to evaluate their performance and security. These metrics are crucial in understanding the strengths and weaknesses of different PUF implementations. Some of the metrics commonly used include uniqueness, reliability, and uniformity. By defining these metrics, we provide basic tools to critically analyze and compare different PUF solutions.

Another focus of this chapter will be to present state-of-the-art in array-based Physically Unclonable Functions (PUFs), which represent the cutting-edge and most advanced PUF implementations currently available. These PUFs will later be used for comparison against the novel PUF proposed in this thesis.

2.1 Introduction to Physically Unclonable Functions

A physically unclonable function (PUF) can be defined as a physical entity whose behaviour is a function of its structure and the intrinsic variation of its manufacturing process [1]. They are a class of cryptographic primitives[1] and security devices that are used to provide unique and unclonable identifiers based on physical characteristics of the underlying hardware. The concept of PUFs emerged as a response to the increasing concern about hardware-based security threats, such as counterfeiting, tampering, and unauthorized duplication of integrated circuits.

The key idea behind PUFs is to exploit the inherent randomness and uniqueness found in the physical properties of individual hardware components during manufacturing. No two chips or devices are exactly alike due to inherent variations in the manufacturing process, even when produced in the same batch or using the same design. These variations can be exploited to create a unique and unpredictable identifier for each individual chip.

There are different types of PUFs, but one of the most common implementations is the "silicon PUF" that exploits the intrinsic variations during the manufacturing process of devices on the silicon substrate. Silicon PUFs typically rely on small analog circuit structures, such as ring oscillators or delay lines, which are extremely sensitive to manufacturing variations. For each input signal that is provided to the circuit, which is referred to as a "challenge", an output is generated, which is termed a "response". So, by applying certain challenges to these circuits and observing their responses, a unique response or "fingerprint" is generated, which serves as the chip's unclonable identifier.

[1] A cryptographic primitive can be defined as a low-level cryptographic algorithm which has a specific security-related functionality [1]

PUFs have found applications in various areas, such as device authentication, secure key generation and storage, secure bootstrapping, anti-counterfeiting, and anti-tamper systems. However, PUFs are not without challenges, such as reliability issues due to environmental variations and aging effects, noise-induced variations, and potential vulnerabilities to advanced attacks. As technology advances, research and development continue to improve the effectiveness and security of PUFs for real-world applications.

2.1.1 Security attacks

It is no secret that the dependency on portable hardware devices for performing security-sensitive tasks has been steadily rising. Among these devices, smartphones stand out as the most prominent examples, boasting a plethora of sophisticated applications that have become indispensable in modern life. These include, but are not limited to, online shopping, using platforms for movie streaming, conducting secure bank transactions, and facilitating business operations.

Another domain which highlights the need for secure hardware systems is the Internet of Things (IoT), which is essentially a network of physical objects connected to the Internet infrastructure to perform tasks without human interaction [1]. The role of these devices is to collect and exchange data, and sometimes act upon the assessment of said data. This is the reason why they are equipped with sensors, and of course the necessary software and hardware to facilitate their functionality.

Whatever the domain may be, creating secure systems is a challenging endeavor, compounded by the rapid escalation in the intricacy of computing devices. The designers need to ensure that the device remains resilient against a wide array of security attacks, which can be broadly categorized into communication attacks, software attacks, and hardware attacks. Each type of attack targets different vulnerabilities in the system and aims to compromise its integrity, confidentiality, or availability.

1. Communication attacks: they focus on intercepting, manipulating, or disrupting the data transmitted between electronic devices or over networks. Some common communication attacks include:

a. Man-in-the-Middle (MITM) Attack: An attacker secretly intercepts and possibly alters the communication between two parties, making them believe they are directly communicating with each other.
b. Eavesdropping: Attackers passively monitor and capture data exchanged between devices or transmitted over a network, aiming to extract sensitive information.
c. Denial-of-Service (DoS) and Distributed Denial-of-Service (DDoS) Attacks: These attacks overload a system or network with excessive traffic or requests, rendering it unable to respond to legitimate users or services.
d. Replay Attacks: Attackers capture and retransmit valid data packets to deceive a system into accepting outdated or repeated commands.

2. Software Attacks: they exploit vulnerabilities in software applications, operating systems, or protocols to gain unauthorized access, execute malicious code, or compromise data. Common types of software attacks include:
    a. Malware: Malicious software, such as viruses, worms, trojans, and ransomware, infects systems and disrupts their normal operations or steals sensitive data.
    b. Buffer Overflow: Attackers input more data than a program's buffer can hold, causing it to overwrite adjacent memory, potentially leading to code execution or system crashes.
    c. Injection Attacks: Malicious data is inserted into user inputs, exploiting vulnerabilities in the software to execute unauthorized commands or access sensitive data.
    d. Zero-Day Exploits: Attackers target previously unknown software vulnerabilities before developers can release patches, gaining a significant advantage.
3. Hardware Attacks: they target the physical components of electronic systems to compromise security or extract sensitive information. These attacks are often sophisticated and require physical access to the device. Some common hardware attacks include:

a. Side-Channel Attacks: Attackers monitor the power consumption, electromagnetic radiation, or other physical characteristics of the device to infer sensitive information, such as cryptographic keys.
b. Fault Injection: Attackers deliberately introduce faults, such as voltage spikes or radiation, to disrupt the normal behavior of the hardware and potentially bypass security measures.
c. Hardware Trojans: Malicious circuitry is inserted into the hardware during manufacturing, acting as a backdoor or causing malfunctions when triggered by specific conditions.
d. Reverse Engineering: Attackers attempt to understand and manipulate the hardware design to extract proprietary information or discover vulnerabilities.

To defend against these attacks, electronic systems must employ a combination of robust software design, secure communication protocols, encryption mechanisms, and hardware protections. This is why it is considered essential to provide an overview of the simplest cryptographic primitives which are being used in developing defense mechanisms, and as a result, to highlight the apparent benefits that PUFs offer in comparison.

2.1.2 Cryptographic primitives

Cryptographic primitives are fundamental building blocks of modern cryptography, serving as the foundational elements upon which secure communication, data protection, and authentication systems are constructed. These primitives are essential mathematical functions and algorithms that provide various security properties, such as confidentiality, integrity, authenticity, and non-repudiation.

Cryptographic algorithms can be broadly divided into three categories [4]:

1. Keyless: they do not use any keys during the necessary cryptographic transformations. These algorithms are valuable tools in cryptography, especially for tasks where secret keys are not required, such as data integrity verification, checksumming, and generating unique identifiers. Two of the most prominent keyless algorithms are hash functions and pseudo random number generators:
    a. Hash functions are one of the most common types of keyless algorithms. They take an input (message or data of any length) and

produce a fixed-size output, known as a hash value or message digest. The same input will always produce the same hash value, and even a slight change in the input results in a completely different hash value. Hash functions are widely used for data integrity verification, digital signatures, and generating secure identifiers.

b. A pseudorandom number generator (PRNG) generates a predictable sequence of numbers or bits that resembles a truly random sequence. Despite its appearance of lacking any discernible pattern, this sequence will eventually repeat after a certain length. However, for certain cryptographic applications, this seemingly random sequence is adequate and serves its purpose.

2. Single-key: apart from the input data, the transformation needs only one single secret key. Encryption algorithms that use a single key are referred to as symmetric encryption algorithms. During the encryption process, an encryption algorithm utilizes a single key so as to transform the input data. For the decryption process, a corresponding decryption algorithm uses the same key on the transformed data in order to recover the original data. Single-key algorithms can be categorized into block ciphers and stream ciphers based on their encryption methods:

   a. Block ciphers operate on fixed-size blocks of data, typically dividing the plaintext into blocks of equal length before encrypting them. The algorithm takes a fixed-size block of plaintext and a secret key as input and produces a corresponding block of ciphertext of the same size. Block ciphers are well-suited for encrypting large chunks of data efficiently, making them ideal for securing files, disks, and data transmission over secure channels. A well-known example of a block cipher is the Advanced Encryption Standard (AES), widely used for various security applications.

   b. Stream ciphers, unlike block ciphers, encrypt data bit by bit or byte by byte, operating on continuous streams of data. They generate a pseudorandom keystream based on a secret key, which is then combined with the plaintext using the XOR operation to produce the ciphertext.

Stream ciphers are often faster than block ciphers for encrypting data in real-time or streaming scenarios, making them suitable for applications like secure communication and real-time encryption of multimedia data. However, stream ciphers generally do not provide the same level of error propagation and diffusion as block ciphers, making them potentially vulnerable to certain types of attacks if the keystream is compromised.

3. Two-key: there are two keys which are used throughout various stages of the process, which are called public and private. These keys work together to enable secure communication, data protection, digital signatures, and key exchange. The fact that different keys are employed for encryption and decryption provides additional security and versatility compared to symmetric encryption. There are several widely used asymmetric encryption algorithms, including RSA (Rivest-Shamir-Adleman), Diffie-Hellman Key Exchange, Elliptic Curve Cryptography (ECC), and Digital Signature Algorithm (DSA). For example:
    a. Asymmetric encryption is instrumental in secure key exchange protocols. Two parties can use each other's public keys to establish a shared secret key securely, which can then be used for subsequent symmetric encryption of data.
    b. Asymmetric encryption is commonly used to generate digital signatures, which provide authentication, data integrity, and non-repudiation. The sender uses their private key to sign the message, and the recipient uses the sender's public key to verify the signature's authenticity and ensure the message's integrity.

Problems arise when classic cryptographic algorithms are implemented in IoT devices, as these devices are characterized by constraints in the availability of resources. There have been attempts at characterizing the implementations of some standard block ciphers in [13] and [14], as the research for energy and performance efficiency is still ongoing. And it is usually the case, that such implementations are prohibitively expensive for IoT devices [1], with a power consumption in the order

of mW [13]. This is one of the reasons why PUFs seem like a viable alternative, as they promise a much lower power consumption, usually at the order of nW.

2.1.3 Variability in the manufacturing process

The present optimal approach for ensuring a secure memory or authentication mechanism within a mobile system involves embedding a secret key within a nonvolatile EEPROM or battery-backed SRAM. Subsequently, hardware cryptographic operations like digital signatures or encryption are employed. This approach incurs significant costs, both in terms of design space and power consumption. Additionally, these nonvolatile memory methods are frequently susceptible to invasive attack methods. Countering such threats necessitates implementing active tamper detection and prevention circuitry, demanding a continuous power supply [3].

The reason why PUFs are such a promising solution is because they can be used for authentication and secret key storage without the need of hardware like EEPROMs or SRAMs. They produce a secret key, instead of storing it, by utilizing the physical characteristics of the integrated circuit (IC). The variability which occurs during the manufacturing process ensures that no two ICs are the same.

It is this variability that cannot be replicated, which results in differences in the performance of the circuits that are manufactured. And there are several sources for this variability, as the ongoing scaling of semiconductor technologies has significantly complicated the fabrication of accurately sized devices. Mainly [1]:

1. Devices' geometry: there are variations in the gate oxide thickness and the lateral dimensions (e.g., the channel length or width) of the MOSFETs that comprise the circuit.
2. Devices' material: deviations arise in the material parameters of the MOSFETs, such as doping variations, and variations in silicide formation, or the grain structure of poly or metal lines.
3. Interconnects' geometry: there are fluctuations in the width and the spacing of the interconnects, as well as in the metal thickness, and in the dielectric height.
4. Interconnects' material: the metal resistivity, the dielectric constant, and the contact and via resistance are all sensitive to the different kind of processes that might be required for the devices' fabrication.

Such processes include, and are not limited to lithography and masking, etching and annealing, and chemical mechanical polishing (CMP). Variations are also introduced not because of the manufacturing processes themselves, but due to differences in the equipment used, or variations inherent to the random nature of physical processes at the atomic level. No matter the cause, process variations are unavoidable when dealing with VLSI circuits, which is why PUFs seem like a viable solution.

2.2 Metrics Used for Evaluation

Before presenting some common architectures for PUFs and analyzing the PUFs that will be used for the comparison with the proposed one in this work, the metrics that are going to be used throughout this work will be defined. These are uniqueness, uniformity, reliability, and tamper resistance, with the latter one being used scarcely. With these metrics, the quality of a design and its suitability for an application can be evaluated. At first, the *Hamming distance* and *Hamming weight* need to be defined, as the metrics are based on them [1]:

> **Definition 2.1** *The Hamming Distance HD(a, b) between two words a = (a_i) and b = (b_i) of length n is defined to be the number of positions where they differ, that is, the number of (i)s such that $a_i \neq b_i$.*
>
> **Definition 2.2** *Let 0 denotes the zero vectors: 00 . . . 0. The Hamming Weight HW(a) of a word a = a_1 is defined to be d(a, 0), the number of symbols a_i != 0 in a.*

As has been stated in the beginning of the chapter, when a PUF is fed by a challenge as an input, it produces some sort of response. Starting with the *uniqueness*, it is basically the average inter-chip Hamming distance among k chips. That means, the average Hamming distance of the responses between k different chips, when they are given the same input as a challenge. If for a challenge C two chips i and j ($i \neq j$) produce n-bit responses $R_i(n)$ and $R_j(n)$, then uniqueness is defined as [1]:

$$HD_{INTER} = \frac{2}{k(k-1)} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \frac{HD\left(R_i(n), R_j(n)\right)}{n} * 100\% \qquad (2.1)$$

Ideally, uniqueness should be as close to 50% as possible.

When it comes to *reliability,* it is used to quantify how consistent the response R for a certain challenge C is, no matter the variations in the conditions of the environment. Reliability is usually measured by varying either the temperature, or the voltage supply, while keeping the other parameter constant. Let i be a single chip, which for the same challenge C has an n-bit response $R_i(n)$ at normal operating conditions, and $R_i'(n)$ at different conditions. Then, the average intra-chip Hamming distance for k chips is defined as [1]:

$$HD_{INTRA} = \frac{1}{k}\sum_{i=1}^{k}\frac{HD\left(R_i(n), R_i'(n)\right)}{n} * 100\% \tag{2.2}$$

Then, the reliability of a PUF is defined as follows, and its desired value is as close to 100% as possible [1]:

$$Reliability = 100\% - HD_{INTRA} \tag{2.3}$$

The *uniformity* is the metric needed to quantify the unpredictability of a PUF. In a truly random response, the amount of 0's is equal to the amount of 1's, and the uniformity is 50%. If k is the total number of responses, and r_i is the Hamming weight of the *i*th response, then uniformity is defined as follows [1]:

$$Uniformity = \frac{1}{k}\sum_{i=1}^{k} r_i * 100\% \tag{2.4}$$

Lastly, *tamper resistance* signifies how difficult it is to tamper with a PUF in any way. It should be expected that the response of a modified PUF will be distinctly different than the original one, i.e., the percentage of the metric should be close to 50%. If CRP is the total number of challenge/response pairs, and $R_i(n,l)$ and $R_j(n,l)$ are the responses of the authentic (i) and the modified (j) chip for a specific challenge l, the tamper resistance is defined as [1]:

$$HD_{AVE} = \frac{1}{CRP}\sum_{l=1}^{CRP}\frac{HD\left(R_i(n,l), R_j(n,l)\right)}{n} * 100\% \tag{2.5}$$

2.3 Common PUF Architectures

A PUF can generally be conceptualized as a challenge-response mechanism contained within a black box. The response r produced by the PUF when being given a challenge c as an input is a function of its internal parameters, i.e., r = f(c). These internal parameters represent the process variations that are unique to each PUF and are being exploited for various applications [3].

By taking a closer look, one can develop a generic architecture for silicon PUFs, as there are two requirements these circuits should follow. It is not enough for a PUF to be unique because of the process variations it underwent during manufacturing. It needs to be able to transform these variations into some sort of measurable

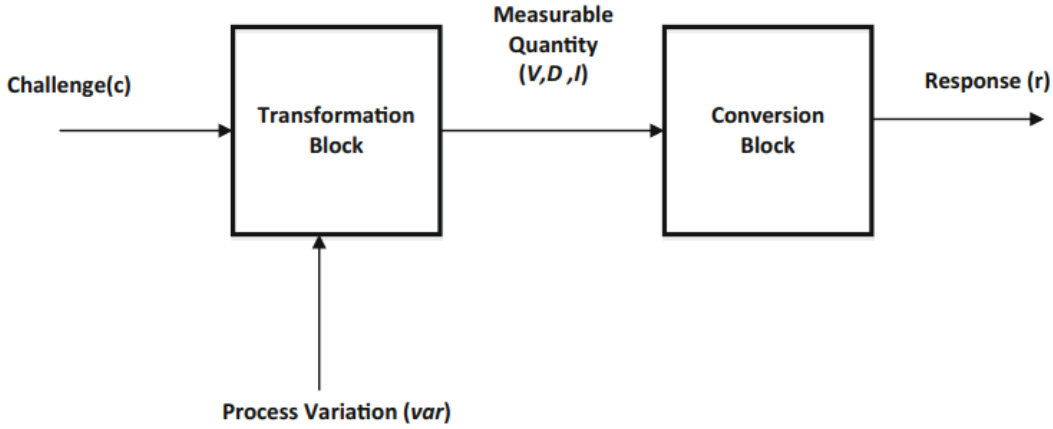

Figure 2.1: A generic architecture for silicon PUFs [1].

quantity, e.g., voltage, current or delay. Furthermore, the response of the PUF should be digitized for the vast majority of applications. By combining the requirements above, one can derive the architecture shown in Fig. 2.1. There, the transformation block is the part of the circuit which is given the challenge and is responsible for turning the process variations into a measurable quantity. The conversion block is necessary in order to convert the measurement into a binary value [1].

In general, a PUF can be characterized as being strong or weak, with the former being used typically for low-cost authentication, and the latter for secure key generation. Whether a PUF is considered strong or weak depends on the number of unique challenges c the PUF can process. When it comes to weak PUFs, only a small number of challenges can be supported, sometimes even just a single challenge. The number of challenges a strong PUF can support, on the other hand, is so large, that

it is impossible to determine all challenge-response pairs within a limited timeframe [3].

PUFs can also be categorized depending on the measurable quantity they are based on. Meaning, they can be delay-based, current-based or voltage-based.

2.3.1 Delay-based PUFs

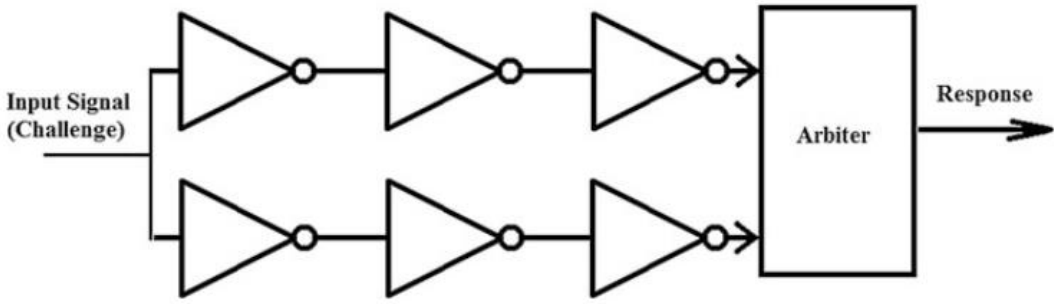

Figure 2.2: A simple arbiter-based PUF [1].

As the name suggests, these PUFs turn the impact that process variations have into a measurable delay figure, which is then digitized. As an example, an arbiter-based and a ring oscillator design will be discussed below [1].

In its simplest form, an arbiter-based PUF looks like the one in Fig 2.2. The two paths leading to the arbiter circuit have an identical nominal delay, but due to the process variations, in reality these delays will be a bit different. So, when a challenge is being given as an input, it will traverse the two paths and reach the arbiter's inputs at slightly different moments. The arbiter circuit is usually based on something like an S-R latch, so depending on which path wins the race, the output will be different.

On the other hand, ring oscillators (ROs) have been proposed as suitable for PUFs. The circuit looks similar to that of the arbiter as shown in Fig. 2.3. It comprises of N ring oscillators, two multiplexers, two counters, and a comparator. Similar to the arbiter PUF, all the ring oscillators are designed to have identical nominal delays. It is the process variations which will alter the delays slightly, making every oscillator oscillate at a different frequency. With the input driven to the multiplexers, two ROs will be selected for comparison. The counter will count the oscillations of each RO in a fixed time interval, and once this interval is over, the comparator will

compare the results of the counter. Depending on which value is the highest, the response of the PUF will be either 0 or 1.

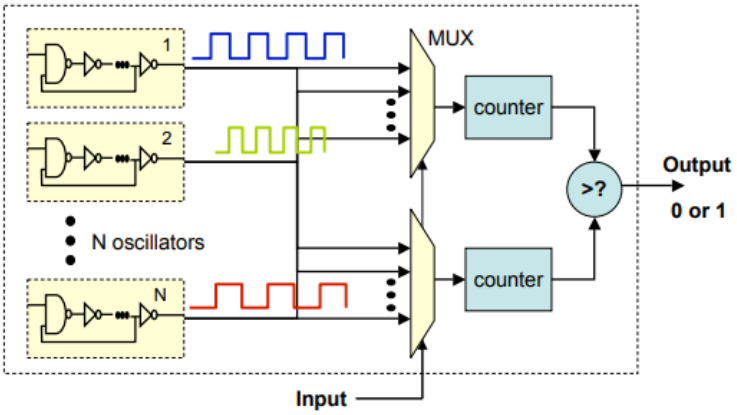

Figure 2.3: A ring oscillator PUF circuit [5].

2.3.2 Current-based PUFs

Such PUFs are counting on measurable current figures, in order to produce a binary response. One of the following examples is utilizing the sub-threshold currents of a transistors' array [6], and the other one needs the leakage current of a DRAM cell [8].

In [6], the authors are exploiting the way the current of the sub-threshold region exponentially depends on the threshold voltage V_{th} and the gate-to-source voltage V_{GS}. There are two identical two-dimensional transistor arrays, which are being driven by the same input signal, as shown in Fig. 2.4. In the absence of process variations, the output voltages of the two arrays will be identical. But the differences in the threshold voltages of the transistors after manufacturing leads to a difference

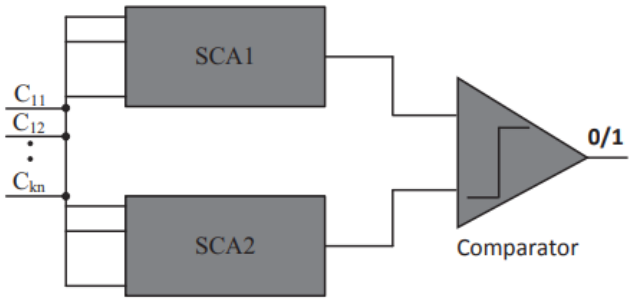

Figure 2.4: The proposed PUF architecture in [6].

in the output voltage each array will have. The comparator, then, will produce a binary response based on this difference.

The structure of the array proposed in [6] is shown in Fig. 2.5. The transistors $M_{ij,}$ which have their gates grounded, operate always in the subthreshold region when they start conducting, and are referred to as stochastic transistors by the authors. By that, they mean that their threshold voltage is highly sensitive to process

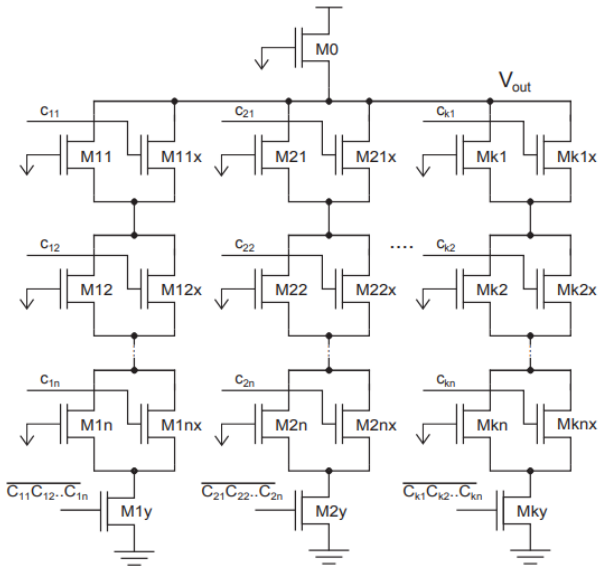

Figure 2.5: The structure of the subthreshold arrays proposed in [6], consisting of k columns and n rows.

variations, as they are of minimum size. These transistors are connected in parallel to the transistors M_{ijx}, which act as switches, need to have small on-state resistance, and a subthreshold current that is negligible compared to the subthreshold current of the stochastic transistors. The result is, the input mandates whether a stochastic transistor will be part of the pull-down network, and thus contribute its subthreshold current to the total branch current, or if it its contribution can effectively be ignored, when a switch transistor is on.

DRAM cells have also been proposed for adaptation and usage as a PUF. A DRAM cell is typically composed of a single capacitor and a single access transistor. The capacitor stores the charge that represents the binary value, while the access transistor controls the read and write operations to the cell. But unlike other memory technologies (like SRAM for example), DRAM cells suffer from a phenomenon called "data decay". Over time, the charge in the capacitor leaks away due to electrical properties of the materials involved. To combat this, DRAM cells need to be periodically refreshed by reading their contents and then rewriting them. This decay

depends on the leakage current, which varies in each cell due to the process variations. In [8], the authors are exploiting this decay behaviour of DRAM cells, in order to turn them into a PUF. Basically, a region of the memory is chosen to function as a PUF for a short period of time, and its refreshing function is disabled. The response of the PUF is then dependent only on the decay rate each cell is going to have, which in turn is affected by how the process variations impact the leakage current [1].

2.3.3 Voltage-based PUFs

In the two examples that are going to follow, the PUF designs use a measurable voltage figure for their response. One of the designs is based on SR latches, and the other on SRAM cells [1].

In general, most SRAM cells exhibit a distinctive and consistent initial state upon power-up. This property enables their utilization in PUF implementations. For example, in [9] the authors are utilizing PUFs based on SRAM in order to securely store a device-specific encryption key for FPGAs, by scrambling the bit streams of the key beforehand [1].

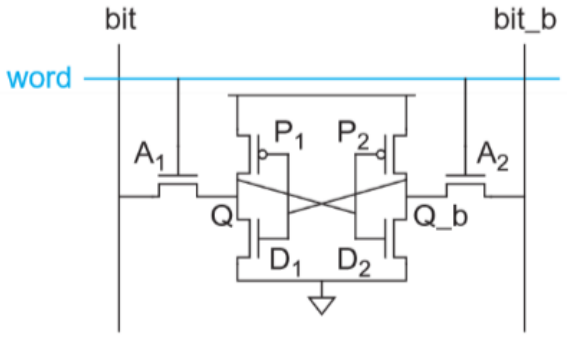

Figure 2.6: The layout of a standard 6-transistor SRAM cell [15].

The structure of a standard 6-transistor SRAM cell is shown in Fig. 2.6. The two cross-coupled inverters are in essence holding the state Q and Q_b of the cell, and the two access transistors A_1 and A_2 are used when attempting to read or write the state through the two bitlines named bit and bit_b. The cell needs to have *read stability*, and *writability*, meaning that the inverters holding the state have to be strong enough in order to not be disturbed when a reading operation is executed, but they have to be weak enough, so that they can be overwritten during a writing operation [15].

During power-up, an SRAM cell undergoes a transient phase called the "power-up" or "power-on" phase, where its internal nodes and transistors transition from an undefined state to a stable state. The power struggle in an SRAM cell refers to the dynamic interplay between various components within the cell as they settle into their desired states. Depending on the driving strength of the two cross-coupled inverters, the cell will eventually reach a certain state. Each cell usually assumes a specific initial state consistently after the power up, because process variations ensure that the driving strength of the inverters isn't identical. It is exactly this characteristic which renders SRAM memories suitable for the creation of PUF circuits [1].

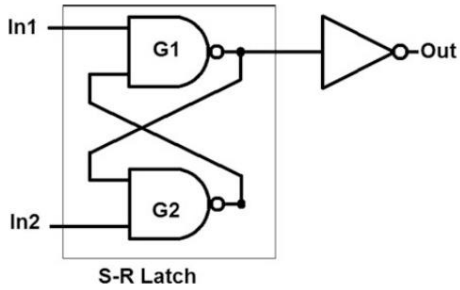

Figure 2.7: An arbiter based on an SR latch [1].

Going back to the arbiter PUFs, an arbiter PUF based on an SR latch is shown in Fig. 2.7. When both In1 and In2 are low, the output of the inverter will be low as well. With both inputs low at first, if In1 goes high, the output of G1 will be 0, forcing the output of the inverter to go high. On the other hand, if with both inputs low at first, In2 goes high, the output of G2 will be 0, and the output of the inverter will remain low as well. Depending on which signal changes its value first, the output of the inverter will assume a corresponding value.

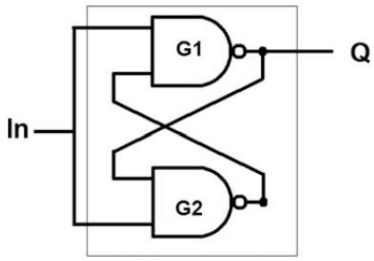

Figure 2.8: A modified PUF based on an SR latch [1].

However, it is an issue when both inputs change within a short time span, or at the same time, as the latch may enter a metastable state. That means, its output can

hover in an indeterminate level between logical high and low for a certain period. The authors in [10] and [11] tried to force such metastability events and utilize the differences in the driving strength of the gates comprising the latches due to the process variations, in order for the latch to eventually assume a value. An example of how a single bit challenge/response PUF based on an SR latch might look like can be seen in Fig. 2.8. There, two cross-coupled NAND gates are used, and the inputs S and R which are usually separated, now comprise the single input In. There are reliability issues to be considered, however, as the duration of metastability can be hard to predict, and the effect of the process variations isn't the only parameter influencing the outcome. Power supply variations, noise, and other environmental conditions can also impact the output of the latches [1].

2.4 SRAM Based PUFs

It was considered essential to provide a more detailed look on PUFs based on SRAMs, since they are in abundance in literature. In [17], the authors propose the embedment of a true random number generator (i.e., TRNG) and a PUF within the SRAM array of a device, for the purpose of secure key generation. Instead of exploiting the power-up state of the SRAM cells, as was explained in subsection 2.3.3, the bitline charge rate is being used in both the TRNG and the PUF.

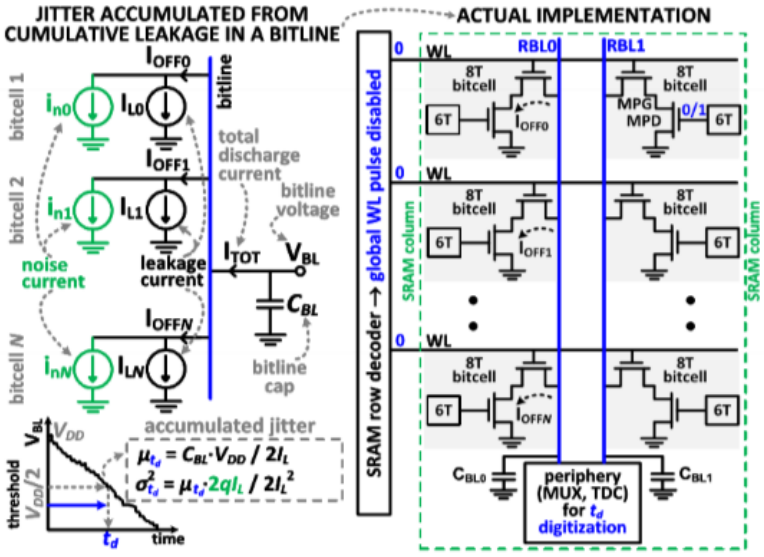

Figure 2.9: The logic behind the usage of a TRNG [17].

The logic the authors propose for the TRNG is shown in Fig. 2.9. To facilitate the operation of the TRNG, the prerequisites are the disabling of all wordlines, and the precharge of the capacitance C_{BL} at a voltage of V_{dd}. Then, the cumulative bitline leakage current from all bitcells in a column initiates the discharge process of the capacitance C_{BL}. The total discharge current, denoted as I_{TDT}, is solely influenced by white Gaussian noise, thereby inducing the accumulation of temporal variations in the discharge process. Eventually, a pulsewidth will be started when the bitline voltage crosses 60% of V_{dd}, and it will be stopped when it crosses 40% of V_{dd}. This pulsewidth t_W will then be driven into a ring oscillator (i.e., RO), resulting in its digitization.

The principle behind the utilization of a part of the SRAM array as a PUF is the same as the aforementioned one. Meaning, it involves harnessing the bitline discharge rate, but in this instance, process variations introduce an element of randomness to this rate, rather than noise. In the case of the PUF, all the bitlines have to be precharged, and all wordlines but one are deactivated. The activation of a specific wordline will result in a I_{read} current, which will then discharge C_{BL}. However, it is mandatory for all bitcells and array rows that will be used for this process to hold the same value, so as to ensure uniformity in the polarity of the discharge. If the time that is necessary for two adjacent bitlines A and B to discharge can be denoted

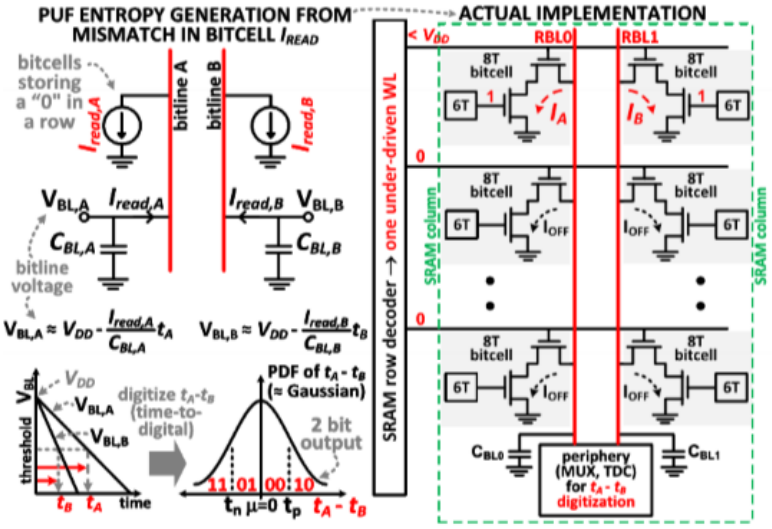

Figure 2.10: The mechanism behind the usage of the bitline discharge rate for a PUF design [17].

as t_A and t_B respectively, then the time difference t_A-t_B will be used as an input to a time-to-digital converter that employs delay and D-latches as time arbiters.

In [18], the authors propose the utilization of not only stable SRAM bits, but also unstable ones, for the realization of a conventional SRAM PUF. They first explore how the SRAM bits are influenced by different temperatures and voltages, as well as the *data remanence* effect. This effect describes how some SRAM cells will retain their previously stored data after powering on the chip, if it was turned off only for a short period of time, as the contents of the SRAM will not be immediately lost when powering it off. After outlining the power-on behaviour of the SRAM cells, they can categorize them into stable and unstable, and they put forward two methods for their utilization, the "16-Power-On Method" and the "Remanence-Based Method", with both methods being split into two phases, the enrollment and the reconstruction phase.

For the first method, during the enrollment phase, the SRAM is powered on 16 times under various temperatures and voltages, and the power-on values the cells attain are read. Whenever a cell has a consistent value with every power-on, whether that is a logic 0 or a logic 1, it is characterized as a stable bit. On the other hand, if a cell reaches both a logic 0 and a logic 1 at least three times during the power-ups, it is denoted as a "reliable" unstable bit. The power-ups have to take place under various circumstances, as the number of stable and "reliable" unstable bits isn't constant, but is highly impacted by temperature and voltage variations. After discarding any unreliable bits (either stable, or unstable), the foundation for the reconstruction is laid. Then, each cell will be read 16 times. Whenever the values read from a cell are all logic 1's or 0's, the response of the bit will be set to 1. If the read values of a cell aren't constant, but the bit is a "reliable" unstable one, its response is set to 0.

The second method exploits the data remanence effect in order to make the selection of the cells that will be used for the PUF. Again, the SRAM cells will undergo the enrollment phase, so as to designate only the desired bits as PUF bits, which will then be read during the reconstruction phase. The SRAM is powered up, with half of its bits being reset to 1, and the other half to 0. It is then powered off only for a short period of time, so as to let the remanence effect impact the results. Upon powering up the SRAM again, the values retained by the cells are recorded.

When the value read from a cell is identical to the reset value it was assigned during the first power up of the SRAM, the cell is denoted as Reset Dominated (RD). The rest of the bits, which display a resilience against the data remanence effect and demonstrate a strong bias towards one direction, are called Strong stable bits. These bits will hold a value different from their initial reset value. The authors go through the enrollment phase under different temperatures and voltage variations, so as to select only "reliable" RD and "reliable" SS bits as their PUF bits, where reliable means that the bits exhibit the same behaviour under all circumstances. The reconstruction phase is of the same logic with the enrollment phase, only now the authors know which bits to read for their responses after the second power up of the SRAM.

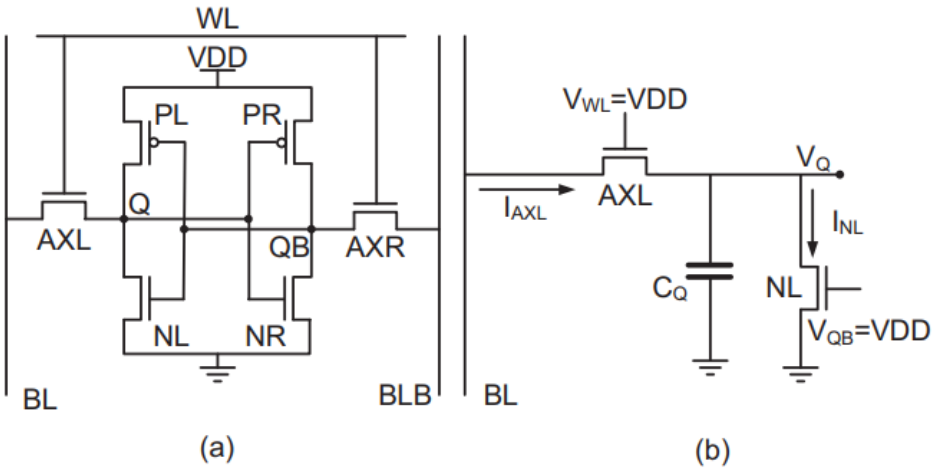

Figure 2.11: (a) A conventional 6T SRAM cell. (b) The flow of current during a read operation when Q=0. [19]

The SRAM continues to be explored for its possibility as a PUF circuit, as in [19] the authors count on the variability of the read current of each cell. This variability stems mainly from fluctuations of $V_{th,}$ as the authors conclude that the variations in parameters like L, W, T_{ox} can be included in the variations of $V_{th.}$ When the value stored in a cell is a logic 0, i.e., Q=0, the read current flowing through the access transistors AXL and NL will be directly impacted by the variations of their V_{th}. Accordingly, when the stored value is 1, the read current which will discharge BLB and flow through AXR and NR will be influenced by their variations in V_{th}. The authors then measure the average read current $\overline{I_{DDT}}$ over a time span Δt for a specific word pair, and the average read current $\overline{I'_{DDT}}$ when the complement of the first word is given as an input. They form 384 128-bit such complementary word pairs for an

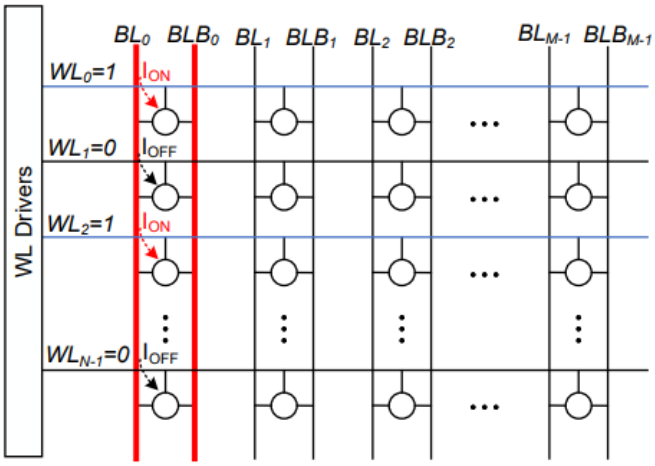

Figure 2.12: An overview of the bitline in-memory computing architecture, where multiple wordlines are enabled. The circles denote a 6T SRAM cell [20].

input, and measure these two average read currents. The digitization, finally, is the result of the difference between the two currents.

Finally, the authors in [20] come to the conclusion that the bitflips that occur during in-memory computations aren't stochastic, but consistent, and can be used for an in-memory PUF design. The authors are in favour of the *bitline architecture* instead of the *wordline architecture*, as the number of columns N is usually higher than the row size M, leading to a strong PUF realization. The bitline architecture for in-memory computing can be seen in Fig. 2.12. Within this architectural framework, multiple wordlines are simultaneously activated. Consequently, the cells corresponding to the rows activated in the previous step engage in a concerted effort to drive both the BL and BLB signal lines. The collision of two adjacent SRAM cells when one of them has stored a logic 0 value, and the other a value of 1, can be seen in Fig. 2.13. Notably, the effective resistances of the transistors situated along the path of the two short-circuit currents will directly impact the value of said currents. This collision will result in the two cells being in a metastable state, which lays the foundation for utilizing the systematic bitflips for a PUF design.

For the PUF operation, when considering two distinct row addresses as A and B, and singling out a specific column among the M columns, designated as m, the challenge will be denoted as C=A_B_m, where the symbol "_" signifies the concatenation operator. The row corresponding to the first address is necessitated to retain

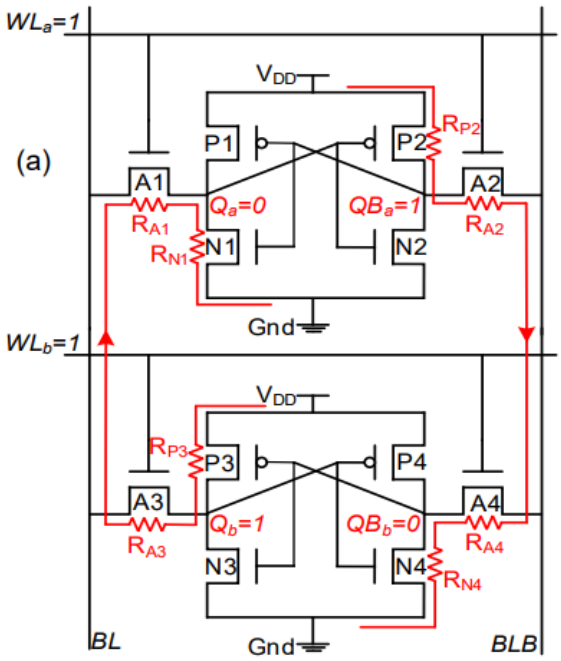

Figure 2.13: The short circuit currents from a read operation when Qa=0 and Qb=1 [20].

a logic 0, while the row corresponding to the second address will have to be initialized to a logic 1, hence the order of the two addresses is important. Once the two cells have been chosen, the bitlines are precharged, and the wordlines are activated. After a specified duration, the worldlines will be deactivated, and the values held by the two cells subsequent to their collision event are ascertained. Ultimately, the response of the PUF will be the AND (&) operation applied to the two post-collision cell values c'_a and c'_b: $R = c'_a \& c'_b$

2.5 Diode Connected Transistor Based PUFs

In [12], the proposal of the authors is a subthreshold current array (SCA) PUF, with the motivation behind it being that other strong PUFs which have been proposed are vulnerable to machine learning attacks, due to the CRPs not being completely unpredictable. Their architecture is the one in Fig. 2.14.

The PUF comprises of two two-dimensional transistor arrays, a common mode feedback circuit (CMFB), and a comparator. Each array consists of n rows and k

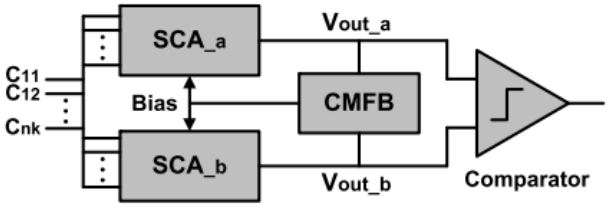

Figure 2.14: The basic architecture of the PUF proposed in [12].

columns of unit cells, as is shown in Fig. 2.15. In every cell, with a row index of i, and a column index of j, two transistors are needed. The diode-connected one (M_{ij}) is called stochastic by the authors, meaning it is sensitive to process variations, as it is of minimum size. The other transistor functions as a switch (M_{ijx}), which is connected in parallel with the stochastic transistor, and is sized so as to keep the process variations from having an impact on it.

The authors are exploiting the exponential dependency the FETs' subthreshold current has with regards to V_{th} as shown in eq. 2.6.

$$I_{ds} = \mu C_{ox} \frac{W}{L}(m-1)\left(\frac{kT}{q}\right)^2 e^{\frac{q(V_{gs}-V_{th})}{mkT}}\left(1-e^{-\frac{qV_{ds}}{kT}}\right) \tag{2.6}$$

The transistors M_{c1} and M_{c2} from Fig. 2.16 act like current sources, with equal and constant currents through them. Their role is to select the bias of each array so as to keep the diode-connected transistors in the subthreshold region, no matter what the inputs C_{ij} are.

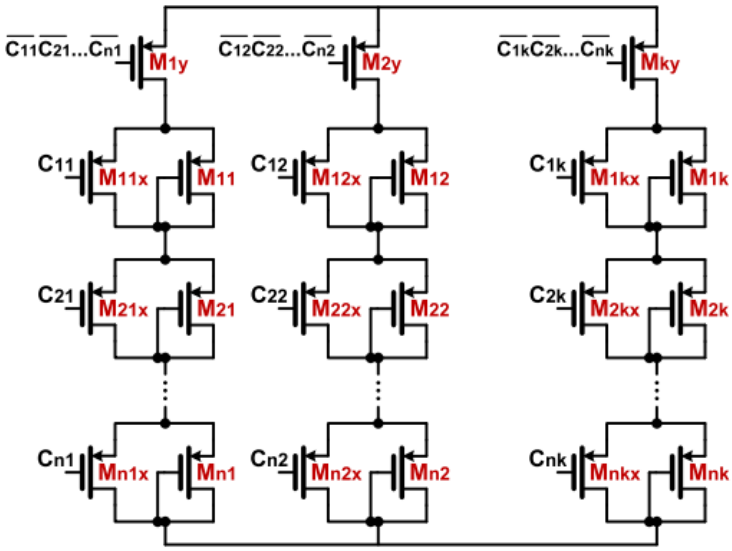

Figure 2.15: The array of n rows and k columns in [12].

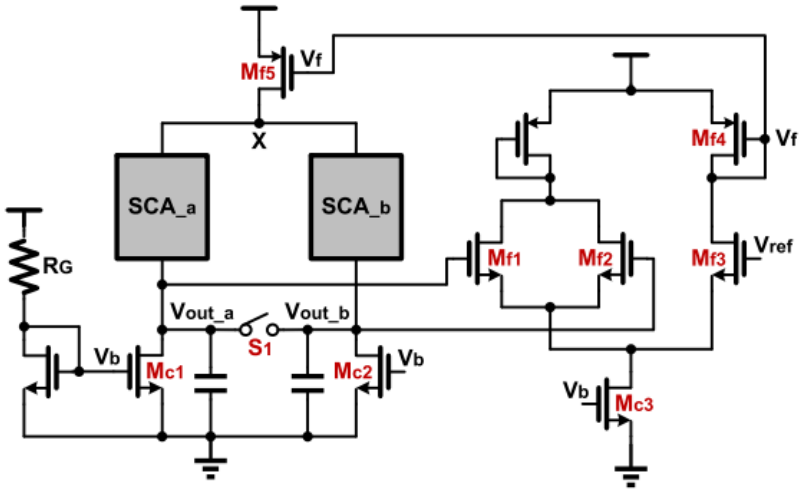

Figure 2.16: The PUF circuit with the CMFB [12].

There are two cases in each cell depending on which value C_{ij} is going to have. When it is equal to 0, the switch transistor is turned on, so the output voltage won't be affected by the stochastic transistor. When C_{ij}=1, on the other hand, the switch transistor is turned off, and the stochastic transistor will take part in making up the output voltage. The input bits C_{ij} are driving both arrays, so the exact same switches will be either on or off on both arrays. In absence of process variations, where V_{th} wouldn't differ across the stochastic transistors, the output voltage of the two arrays would be identical. But due to V_{th} being highly sensitive to process variations, and the subthreshold current depending on it exponentially, in real conditions the two voltages will be different. Based on this difference, the comparator will output a binary value as a response. The authors are also using a common mode feedback circuit, in order to keep V_{out_CM} as mandated by eq. 2.7 from deviating from the desired common mode voltage V_{ref}, as V_{out_CM} is sensitive to the inputs C_{ij}. Finally, they have proposed a comparator based on a strong-arm latch, so as to minimize its offset and its noise.

$$V_{out\_CM} = \frac{V_{out\_a} + V_{out\_b}}{2} \tag{2.7}$$

In [16], the authors exploit diode-connected transistors operating in the sub-threshold region too, as is shown in Fig. 2.17. The PUF consists of 16 columns, with each column containing 16 bitcells, making the PUF a 16 x 16 array. The array also requires a decoder for the selection between the 256 bitcells, a 16-to-1 dual-ended MUX, and a 1-bit comparator, in order to digitize the final output.

The cornerstone of the PUF is the bitcell, as shown in Fig. 2.17 (b). The header is shared for all bitcells in the column. Each bitcell comprises of six transistors. Four of them are arranged as access transistors, in order to get the reading of V_{outL} and V_{outR}. MBR and MBL, which are diode-connected, will form a pair of PTAT generators with MTL and MTR.

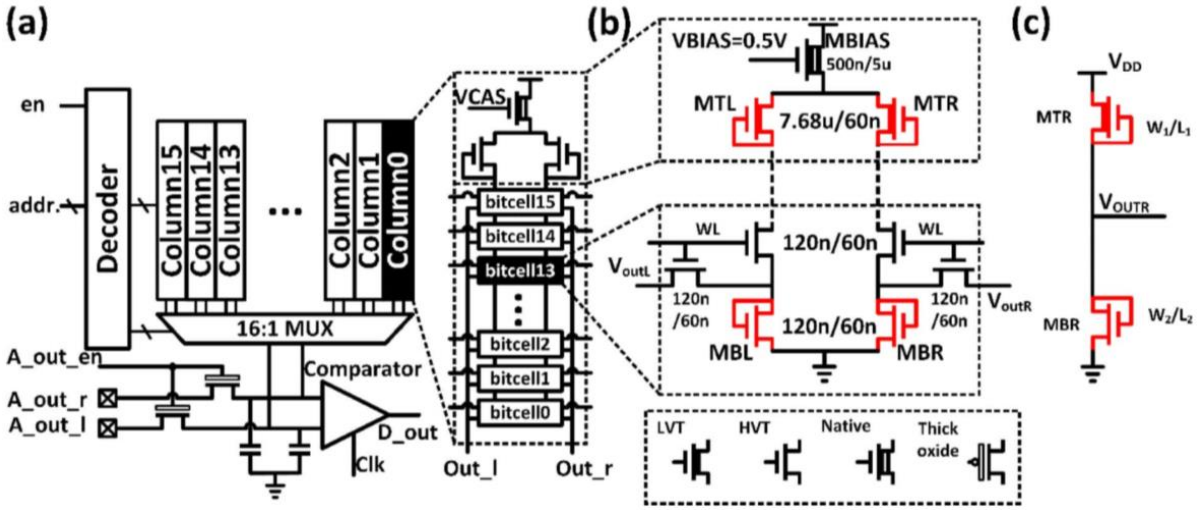

Figure 2.17: (a) The architecture of the 256-bit PUF proposed in [16]. (b) The bitcells and the shared header which comprise a column. (c) What the authors call a PTAT generator. A couple of these is needed, in order for a single bitcell to be formed.

The basic idea is that the top devices MTL and MTR are biased with $V_{gs}=0$, therefore they will operate in the subthreshold region, and so will the bottom devices MBL and MBR. They are diode-connected, and the current is determined mainly by the top devices. So, by taking a look at a single PTAT generator, as is shown in Fig 2.17 (c), and equating the currents flowing through both MTR and MBR, both of which follow eq. 2.6, one can solve for V_{OUTR}, as is shown in eq. 2.8.

$$V_{OUTR} = V_{th2} - \frac{m_2}{m_1} V_{th1} - K_{V_{th}}(T_0) + K_{V_{th}} T + \frac{m_2 kT}{q} \ln\left(\frac{\mu_1}{\mu_2} \frac{C_{ox1}}{C_{ox2} \frac{W_1 L_1}{W_2 L_2} \frac{m_1 - 1}{m_2 - 1}} \right) \quad (2.8)$$

In eq. 2.8 MTR is represented by 1, MBR by 2, and it has been assumed that V_{ds} is sufficiently larger than V_t, so one can eliminate the second exponential term in eq. 2.6. K_{Vth} holds the combined temperature dependencies of V_{th} of the devices MTR and MBR. It also becomes evident why the authors chose LVT devices for the header, and HVT for the bitcells, as there needs to be a sufficient difference between V_{th1}

and V_{th2}. The first three terms of eq. 2.8 are determined by V_{th}, and only the two last terms are dependent on the temperature.

Eventually, by selecting a single bitcell, the difference between V_{OUTR} and V_{OUTL} will determine the output of the comparator. This difference, due to the random process variations that will occur, will be unique in every case.

.

CHAPTER 3

THE PROPOSED ARRAY-BASED PUF

In the current chapter, an in-depth examination of the proposed circuit will be undertaken. This analysis will commence with describing the PMOS and NMOS blocks that constitute the fundamental cells of the array. An overview of their behavior, and the reason as to why they can be utilized for a PUF design will be showcased.

Next, the operation of the overall structure of the entire array will be clarified. The organization of the PMOS and NMOS blocks into cells will be showed, and hence, the modularity the circuit has to offer will be demonstrated.

Finally, the operational principles of the comparator used for the digitization of the output will be outlined, in order to explain how each bit of the response gets calculated.

3.1 Building Cells

The proposed PUF design is based on a memory-like array topology of building cells. The cell structure is illustrated in Fig. 3.1. It is a low silicon area cost design, consisting of only two PMOS and two NMOS transistors. We will first take a comprehensive look on the operational mechanism governing the two blocks. Each PMOS block (or, pBlock) is composed of two serially connected PMOS transistors one of them in the diode-connected topology, and the same stands for every NMOS block (nBlock).

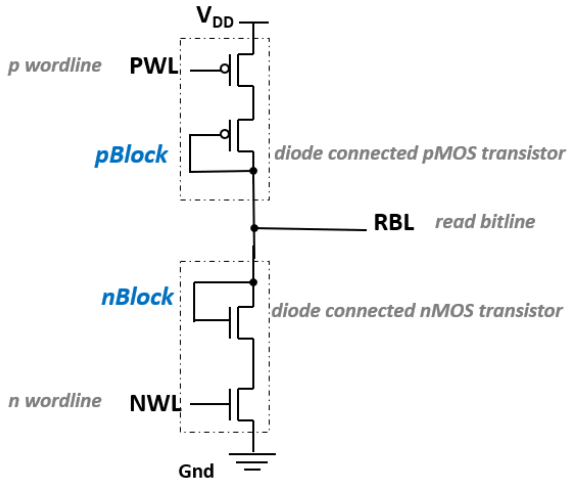

Figure 3.1: The building cells of the proposed array PUF.

The PMOS transistor that is not in the diode-connected topology acts as a switch and has its gate driven with the signal PWL (PMOS wordline), while the gate of the corresponding NMOS transistor is driven by the signal NWL (NMOS wordline). The diode-connected transistors serve as non-linear resistances, by providing a somewhat constant voltage drop V_{ds} across their terminals, higher than V_{th}. Since V_g=V_d, these transistors always operate in the saturation region if they start conducting, with the current flowing through them being mandated by equation 3.1.

$$I_{ds} = \frac{\mu C_{ox}}{2}\frac{W}{L}\left(V_{gs} - V_{th}\right)^2 \tag{3.1}$$

However, due to the process variations, transistor parameters like W, L, C_{ox}, V_{th} will vary in each device. Recall from subsection 2.1.3, the variations during the manufacturing process directly impact the geometry and the properties of the materials that form the devices and the interconnects on the final circuit. Therefore, the voltage

drop across each diode-connected transistor will not be identical, as I_{ds} is directly impacted.

If the array is considered to consist of n rows, as a challenge we denote any activated pair of lines PWL_i and NWL_j, with $i,j \in [0, n-1]$. PWL_i should have the value 0, and NWL_i the value 1 if they are going to be used as an activation pair, so as to concurrently activate a pBlock and an nBlock, and provide a short-circuit current mandated by eq. 3.1. The different indexing of PWL and NWL also means that the indexes don't always have to go in pairs; e.g. PWL_0 doesn't exclusively have to be activated together with NWL_0, but it can be activated with any of the NWL_j. This kind of modularity is what gives the PUF a large CRP space, as each row of nBlocks can be activated with every row of pBlocks. Thus, in a cell either the pBlock or the nBlock or both blocks can be activated given the applied challenge.

The outcome of the simultaneous activation of a pBlock and an nBlock will be the development of a certain voltage on the RBL. It is this voltage that serves as the foundation for the derivation of the PUF's response bits, as it will be showcased in the next subsections.

3.2 PUF Array Design

A basic architecture for the whole PUF array is the one in Fig. 3.2. The array consists of n rows corresponding to nBlocks, and n rows corresponding to pBlocks, i.e., 2n rows in total, and m columns of read bitlines (RBLs). There are m-1 comparators in

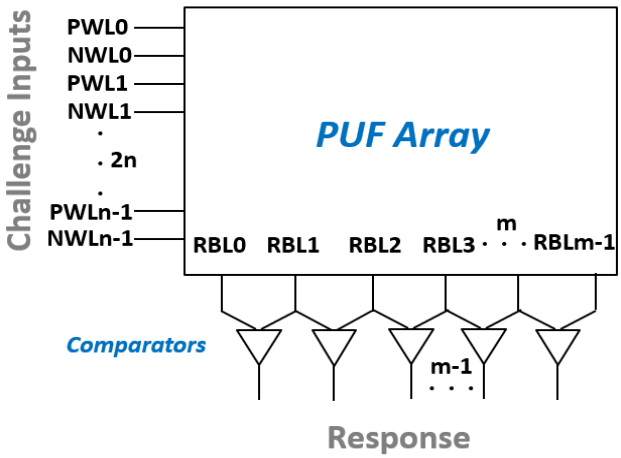

Figure 3.2: The basic architecture of the proposed PUF array.

total, which are driven by the RBL signals of the array. The comparator will be explained in subsection 3.3, however a variety of comparators could be used instead.

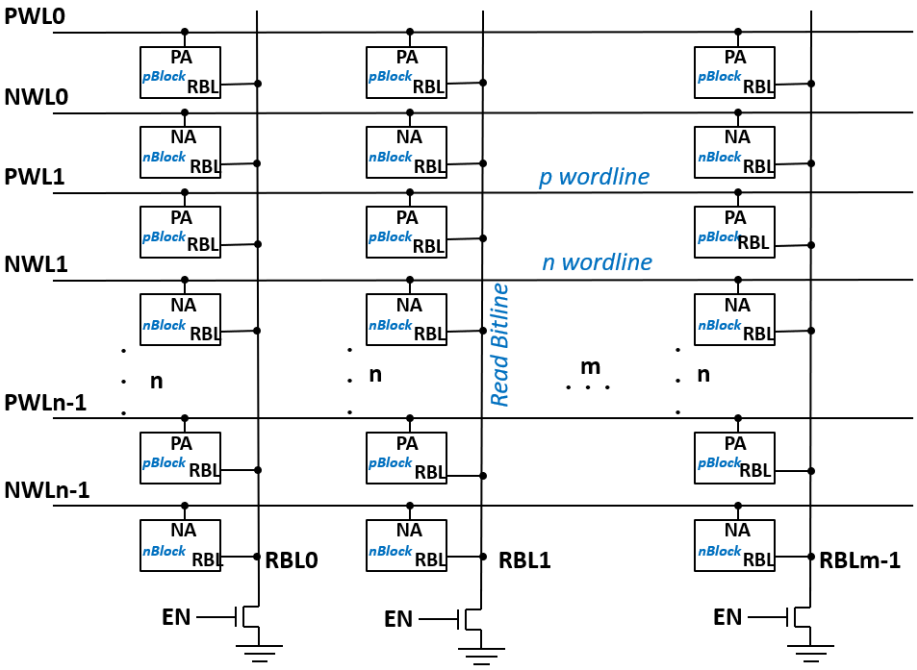

Figure 3.3: A more detailed look on the PUF array. Each PWL_i and NWL_i controls one specific row of blocks.

As it has already been mentioned in subsection 3.1, as a challenge we consider any activation pair PWL_i and NWL_j, with $i, j \in [0, n-1]$. Once a specific pair is activated, the pertinent PMOS and NMOS blocks of the corresponding rows form a voltage divider and the RBL_k of each column k will eventually reach a certain voltage. Each RBL_k will be compared to the voltages of the columns directly next to it, and each comparator will produce a logic output (0 or 1) depending on which signal is larger. In order to have an activation pair, PWL_i will have to be a logic 0, and NWL_j will have to be a logic 1. Since each of the n in total PWL_i signals can be uniquely combined with any of the n in total NWL_j signals, the total number of CRPs is n^2.

When the PUF is idle, all PWL_i and NWL_j are inactive while all footer NMOS transistors are in conducting state (by setting the EN signal to high) to fully discharge all RBLs in the array. For the following, RBL describes the behavior of a specific column k, but all columns have the same behavior. Before the application of a challenge all footer transistors are deactivated. The application of the challenge and the effect this has on RBL can be summed up in Fig. 3.4. In Fig. 3.4, PWL and NWL

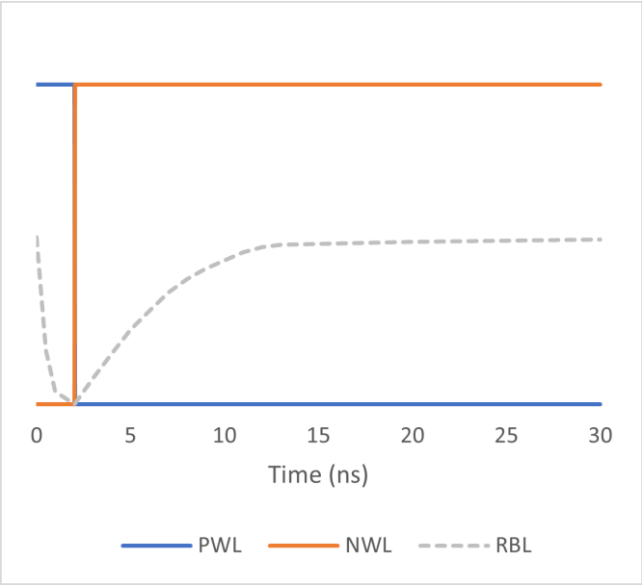

Figure 3.4: The application of a challenge.

correspond to the input signals of a specific PMOS row i, and a specific NMOS row j. The rest of the rows in the array remain inactive. Due to the formation of the expected voltage divider, the RBL will eventually reach a certain voltage. This voltage level depends on the process variations on the transistors involved in the current path and especially these on the NMOS and PMOS diode-connected transistors, as has been explained in subsection 3.1. Finally, the comparators will digitize the difference between the column voltages of adjacent RBLs.

3.3 The Comparator

The voltage comparator to be used in the PUF design can be selected among a large number of comparators proposed in the literature. In this work, for the demonstration of the proposed PUF the comparator in Fig. 3.5 is considered. When it is inactive (SA_SE=0), its outputs OUT_L and OUT_R are equalized to be ready for the read phase that provides the PUF response. The transmission gate is active while the comparator is inactive, in order to impose this equalization of OUT_L and OUT_R. Once the comparator gets enabled (SA_SE=1), it is driven by the two input signals RBL_L and RBL_R, the left and right RBLs that feed the comparator, and depending on the difference on their voltage levels, the final output signals BIT_L and BIT_R will turn

to high or low, being complementary between each other. Notice that the comparator will have to be activated once the difference between RBL_L and RBL_R has built up.

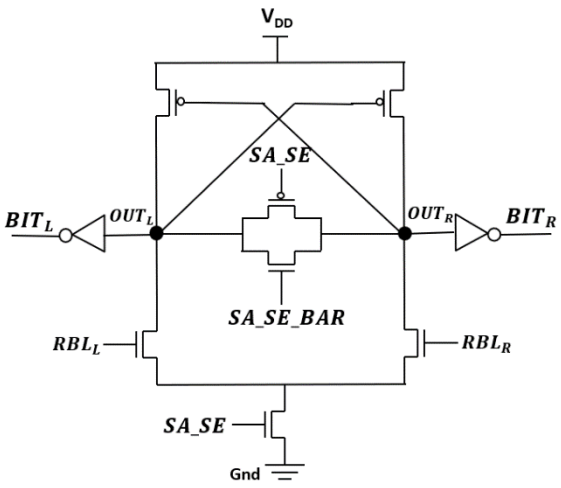

Figure 3.5: The topology of the comparator under consideration.

The operation of the comparator can be summed up in Fig. 3.6. There, the moment of activation is the one of interest, as it displays the effect the difference between RBL_L and RBL_R will have on the outputs BIT_L and BIT_R. Only BIT_L is displayed here, for the sake of simplicity, as BIT_R is complementary to BIT_L when the comparator provides its final decision.

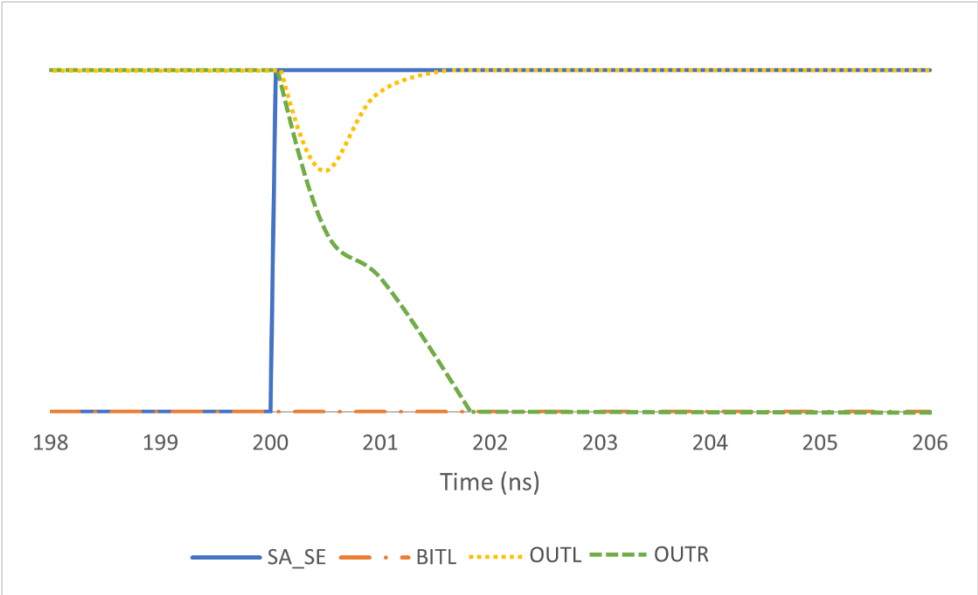

Figure 3.6: An example for the computation of BITL, when RBL_R is larger than RBL_L.

In more details, while SA_SE = 0, the transmission gate makes sure OUT_L and OUT_R will be equal between each other, and kept high, as there is no path to the ground. At the same time, both BIT_L and BIT_R are kept low, since both RBL_L and RBL_R are low. Furthermore, while the comparator is inactive, both NMOS transistors that are being driven with the RBLs have the same V_{ds}, as OUT_L is equal to OUT_R.

When the comparator gets activated with SA_SE = 1, the transmission gate will be deactivated, and the difference that will be developed between RBL_L and RBL_R will take effect on the NMOS transistors they drive. Let us assume that $RBL_R > RBL_L$. In other words, $V_{gsR} > V_{gsL}$. Hence, the current I_{ds} which will be imposed by the right NMOS transistor will be larger than the one imposed by the left. This means, there will be a bigger voltage drop across the PMOS transistor on the right than on the left, and therefore, OUT_L will be larger than OUT_R. Thus, the PMOS transistor to the left, which is driven by OUT_R, turns to be more conductive than the right PMOS transistor. Consequently, OUT_L will start to rise, and go towards V_{dd}. This will make sure that the PMOS transistor on the right will be turned off, locking OUT_R to 0, and hence, OUT_L will also lock to 1. With OUT_L being high, BIT_L will remain low, and BIT_R will be high.

This sums up how the difference between two column voltages, RBL_L and RBL_R is digitized, once a row i of PMOS blocks and a row j of NMOS blocks get activated, with $PWL_i = 0$, and $NWL_j = 1$.

CHAPTER 4

SIMULATION RESULTS

In the present chapter, details on the PUF circuit design and the corresponding simulations will be given. At first, the circuit that was simulated will be discussed, with the technology and the precise dimensions of the devices being given. Details will be also given regarding the Monte Carlo simulations.

The results from the calculations of uniqueness, uniformity, and reliability with regards to temperature and voltage will be discussed. Comparisons with respect to silicon area and the power consumption under nominal circumstances will also be presented. These performance metrics will be compared to these in [12], [16], and [17]-[20].

4.1 PUF Circuit Design

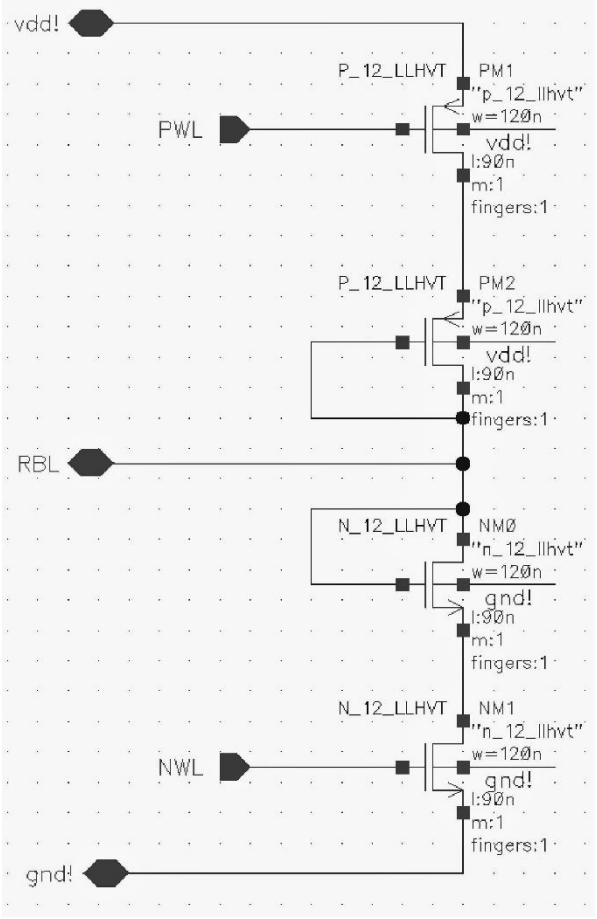

Figure 4.1: The design of the proposed PUF cell.

For the PUF circuit design, the LLHVT (low leak, high V_{th}) devices of the UMC 90nm CMOS technology were used, with the supply voltage being V_{dd}=1.2V. The design of the basic cell is presented in Fig. 4.1. With the use of the basic cell two columns of 256 basic cells were designed. Minimum size transistors (W/L=120nm/90nm) are used for the basic cell. In typical conditions, by activating a pair of a PMOS block and an NMOS block a voltage level of 0.627V is developed on the RBL.

The footer transistor at the bottom of a column (NM2 transistor in Fig. 4.2) that is used for the initialization of the RBL by discharging it to the ground, has a W/L equal to 480nm/90nm.

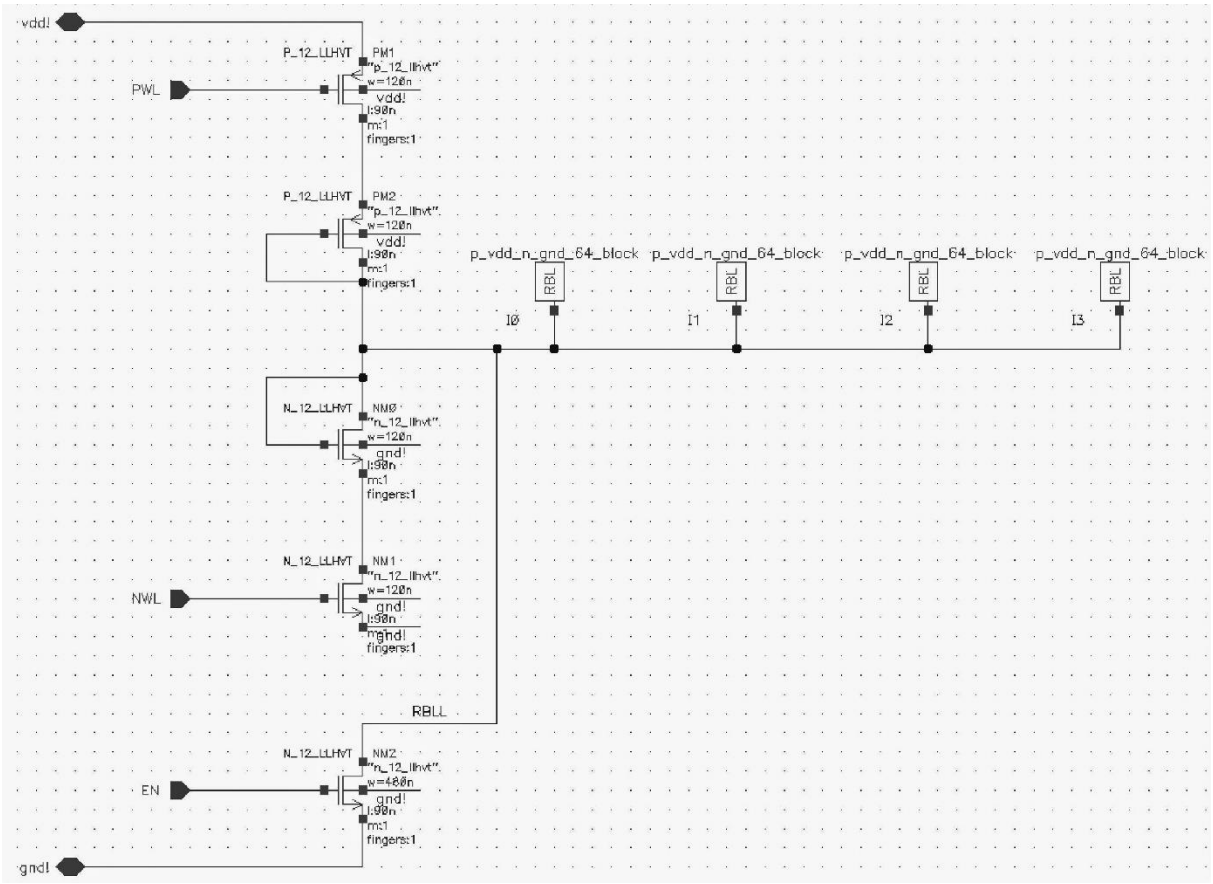

Figure 4.2: A column of 256 NMOS blocks and 256 PMOS blocks, i. e., 512 rows in total.

The transistor which has the role of enabling the comparator has a W/L equal to 2.8um/90nm. The dimensions of the rest of the transistors in the comparator have W/L=120nm/90nm. Lastly, the NMOS transistors used in the inverters that provide the final outputs OUT_L and OUT_R have a W/L equal to 280nm/90nm.

In Fig. 4.4 and Fig. 4.5 a simulation at typical conditions can be seen. During the first 2ns the discharging of RBL_L and RBL_R towards the ground is happening, as the footer NMOS in both columns is activated with EN, and every other cell is deactivated. At 2ns, the footer will be deactivated when EN=0, the NMOS in Fig. 4.2 driven with NWL will be activated, and the PMOS driven with PWL in Fig. 4.2 will also be activated.

The RBLs from the two columns reached a stable voltage at around 50ns for nominal temperature and supply voltage, but in order to include as many outliers as possible, BIT_L was eventually measured at 250ns. For this purpose, there was a delay of 200ns at the function of the sources driving the transmission gate and the

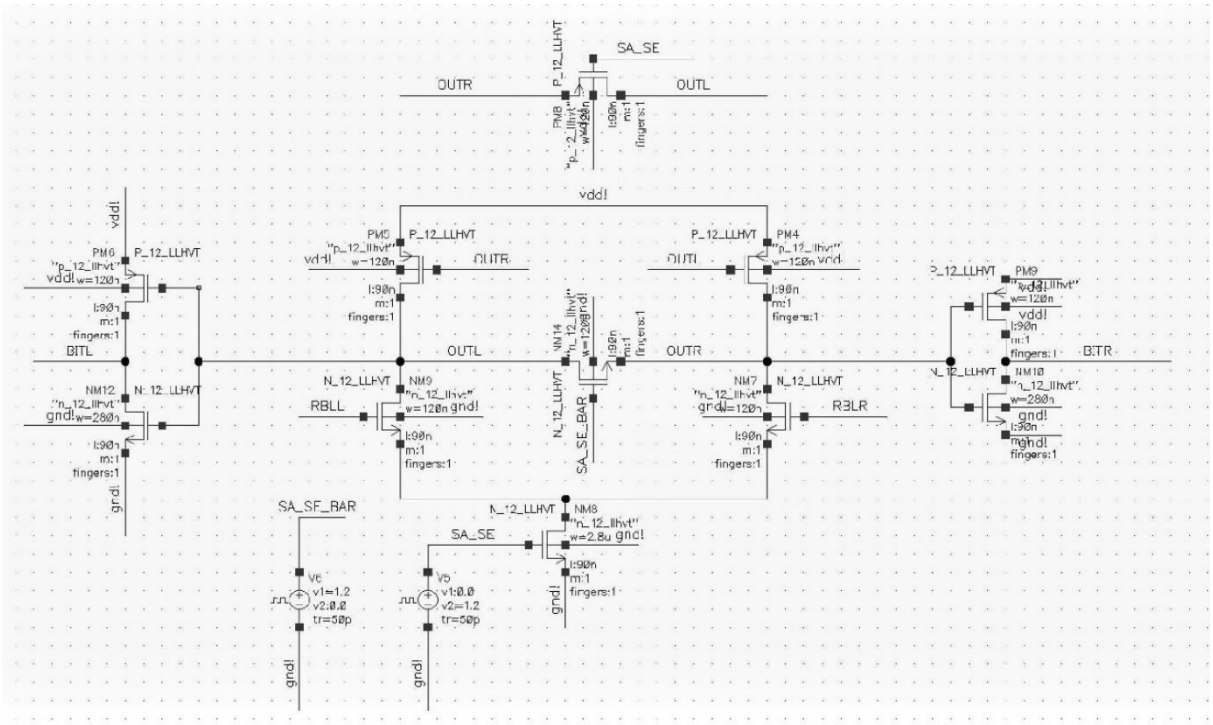

Figure 4.3: The comparator's design.

footer of the comparator. In detail, SA_SE, which activates the footer transistor, and hence the whole comparator, is kept low from the beginning of the simulation, and will rise high at 200ns, as can be seen in Fig. 4.5. SA_SE also drives the PMOS transistor of the transmission gate, so as to keep it active while the comparator is deactivated. The SA_SE_BAR signal drives only the NMOS from the transmission

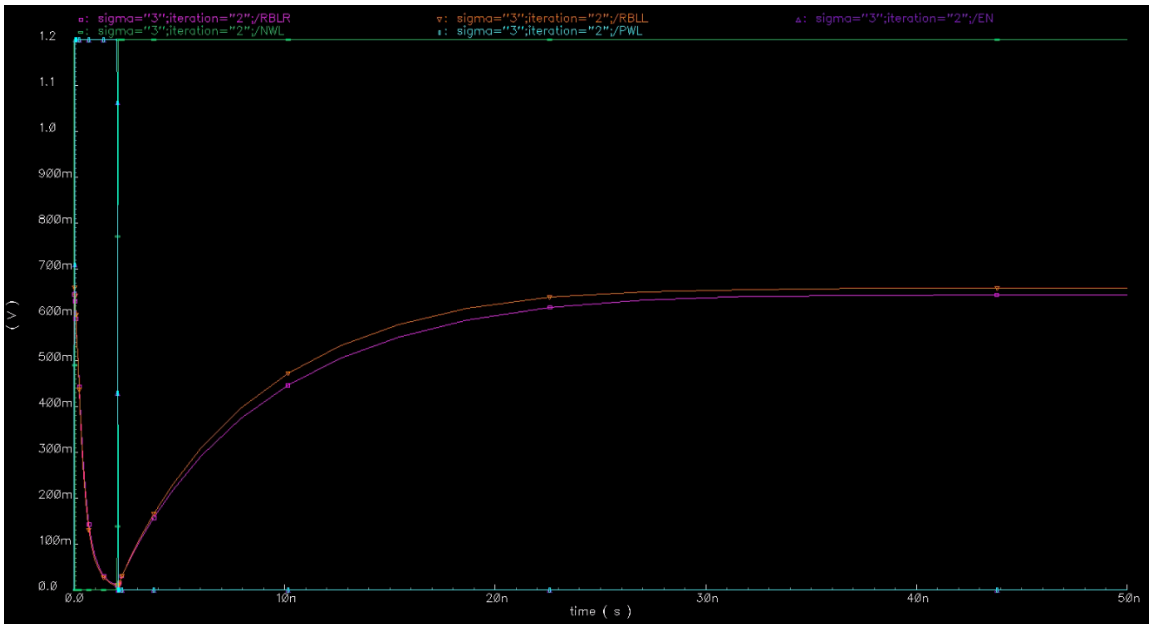

Figure 4.4: Simulated signal waveforms for PWL, NWL, EN, RBLL and RBLR at typical conditions.

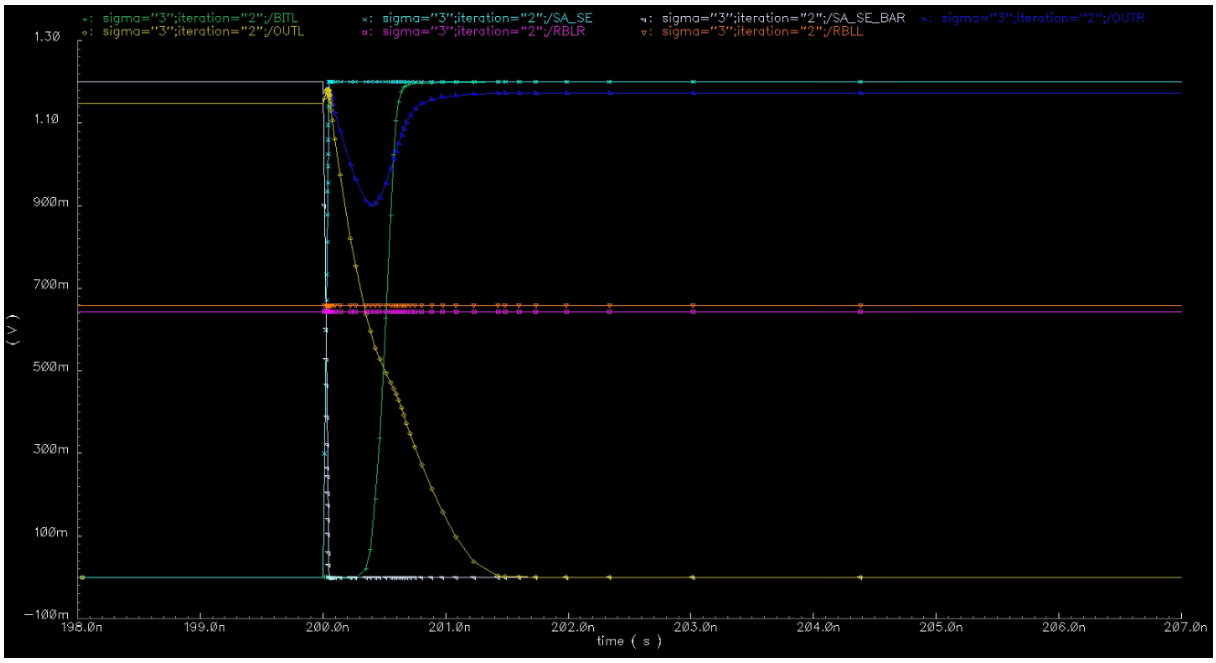

Figure 4.5: Simulated signal waveforms for SA_SE, SA_SE_BAR, RBLR, RBLL, OUTR, OUTL, and BITL at typical conditions as in Fig. 4.4.

gate, and for that purpose, it Is the complementary signal of the SA_SE. Meaning, it is high in the beginning of the simulation, and at 200ns it will go low, so as to deactivate the NMOS of the transmissions gate.

As it is evident in Fig. 4.5, when the comparator gets activated, OUT_L will drop low if RBL_L is higher than RBL_R, and also OUT_R will rise high. This sets BIT_L to logic 1, and BIT_R to logic 0.

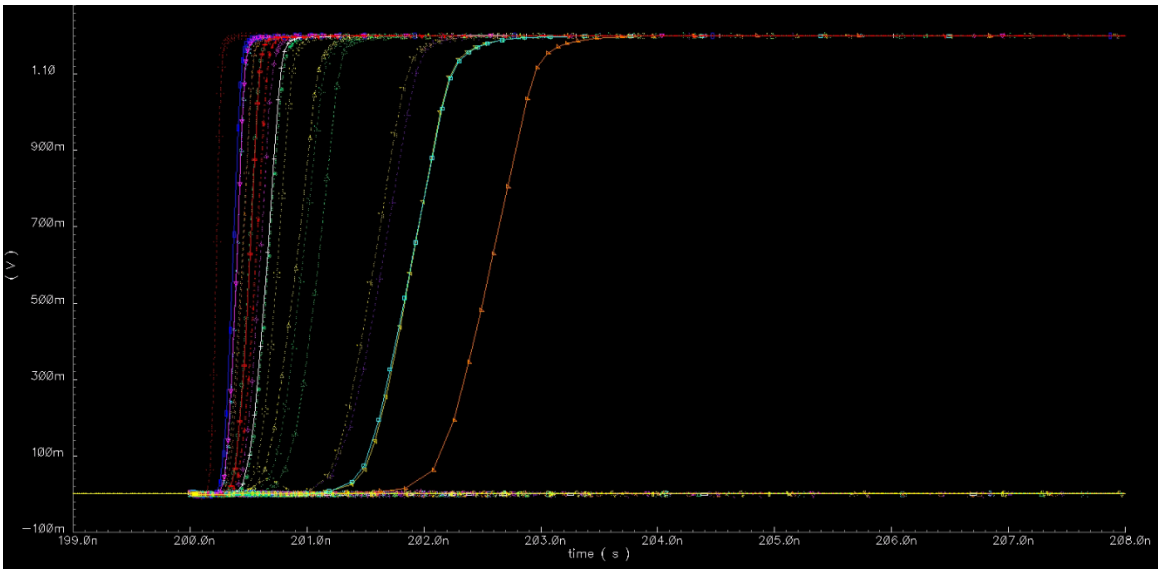

Figure 4.6: An indicative Monte Carlo simulation run.

For the evaluation of the proposed PUF design, Monte Carlo simulations were considered, performing 10,000 runs, considering the statistical models of the used technology. The typical power supply voltage was 1.2V and the typical temperature was 27°C. In the simulations for the reliability estimation, the power supply voltage variations were ±10% of the nominal value (from 1.08V to 1.32V) while the temperature varies among [0, 27, 85 and 125] °C. For the simulations, process and mismatch variations were selected, so as to impose both global and local variations. The sigma parameter was equal to 3. For the calculations of uniqueness, uniformity, and reliability with regards to temperature and voltage, the framework described in [21] was used.

4.2 Simulation Results and Comparisons

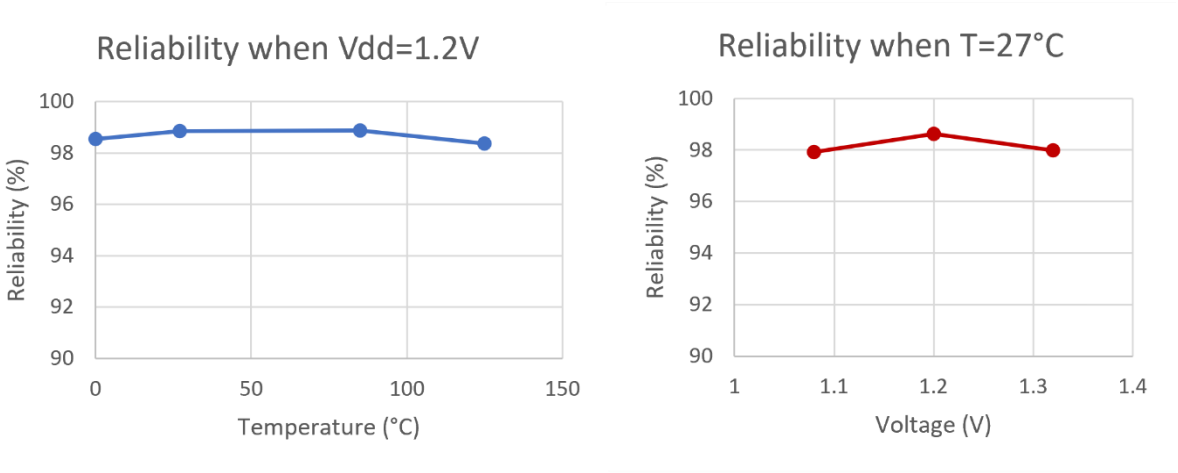

Figure 4.7: PUF reliability results under different voltage and temperature values.

The simulation results with respect to the reliability at different voltage and temperature values can be seen in Fig. 4.7 as well as in Table 4.1. From these results we see that the reliability is always higher than 97.925%. Uniqueness and uniformity

Table 4.1: The measurements for reliability under different voltage and temperature variations.

V=1.2V		T=27(°C)	
T(°C)	Reliability(%)	V(V)	Reliability(%)
0	98.54	1.08	97.925
27	98.86	1.2	98.635
85	98.8667	1.32	97.98
125	98.36		

were measured at nominal conditions (i.e., V=1.2V and T=27°C). Uniqueness was calculated to be equal to 50.0034%, and uniformity equal to 49.73%.

Regarding the power consumption, this was equal to 4.98μW at typical conditions. Comparisons among this work and these in [12], and [16]-[20] can be seen in Table 4.2. Regarding the silicon area requirements estimation, in all cases only the W*L area of the transistors comprising the pertinent basic cell was taken into account, without considering any additional circuitry that is necessary for applying the challenge or reading the response. The silicon area values correspond to the number of minimum sized transistors of the technology used. Hence, for this work, besides the four transistors that constitute a basic cell, seen in Fig. 4.1, the footer transistor from Fig. 4.2 was also taken into account. It is equal to four minimum sized transistors, so this cost is shared among the 256 cell of a row (4/256 per basic cell). Similarly, for [16], from Fig. 2.17(b) the three transistors from the left column of the bitcell were taken into account, with MTL divided by 16, and the header divided by 32. Finally, for all PUFs that are based on SRAMs, it was assumed that the basic cell consists of six transistors, four of them minimum sized, and two of them 10% larger than the minimum size.

Regarding reliability, in all cases measurements without considering aging or error correction methods are presented. Furthermore, there were cases where only

Table 4.2: Comparisons among this work and [12], [16]-[20].

	This work	SCA-ML [12]	PTAT [16]	TRNG-PUF [17]	Power-on / Remanence [18]	Read current [19]	SiCBit-PUF [20]
Technology (nm)	90	130	65	28	-	45	32
Type	Diode-connected	Diode-connected	Diode-connected	SRAM	SRAM	SRAM	SRAM
Vdd (V)	1.08 - 1.32	1.08 - 1.32	0.6 - 1.2	0.75 - 1.05	4.5 - 5	0.9 - 1.1	0.9 - 1.1
Temperature range (°C)	0 - 125	-20 - 80	0 - 80	-25 - 100	0 - 100 / 10 - 100	10 - 85	0 - 100
Reliability (T=c) (%)	97.925	96†	99.55	-	-	-	92.1†
Reliability (V=c) (%)	98.36	88†	97.55	-	-	94.93†°	94.6†
Average reliability (%)	-	94.2	-	96.6†	98.06 / 99.79	95.39×	-
Uniqueness (%)	50.0034	49.9	50.01	50.3	49.33 / 49.5	0.4997°/0.4365×	49.99
Uniformity (%)	49.73	52.8	49.3	-	-	-	49.74
Area/basic cell (min. dev.)	4.02	-	17.85	6.2	6.2	6.2	6.2

†: worst case, °: simulation, ×: fabrication.

the range of the reliability or the intra-HD reported, that's why the lower limit was chosen, referred to as worst case. Lastly, there is also a distinction whenever there were measurements reported from both the simulations and the actual chips that underwent testing.

Overall, the proposed PUF seems to be resilient against temperature and voltage variations, while simultaneously achieving high uniqueness and uniformity levels. Moreover, it requires the least area per basic cell when compared with the rest of the state-of-the-art PUF solutions. Only [12] may have comparable area requirements, but since no details were given regarding the dimensions of the devices used, no calculation was feasible.

CHAPTER 5

CONCLUSIONS

The proposed array PUF circuit exploits diode-connected transistors for the formation of the basic cell while it presents a uniformity of 49.73%, a uniqueness of 50.0034%, and a worst-case reliability of 97.925%. The PUF is robust with better performance metrics with respect to state-of-the-art PUF designs. The estimated silicon area of each basic is equal to 4.02 minimum sized transistors with respect to the used technology, while the power consumption by the activation of a pair of columns was calculated equal to 4.98μW. The response time of the PUF In typical conditions is 250ns.

In case that only one row of PMOS blocks and one row of NMOS blocks are simultaneously activated the possible CRPs is equal to n^2 (where n is the number of rows in the array).

However, for the PUF operation, a higher number of rows with nBlocks and pBlocks can be concurrently activated by proper challenge stimulus. Given that k NMOS rows and s PMOS rows are simultaneously activated, the number of possible CRPs is provided by Eq. 5.1 (where k and s > 0).

$$CRP_{k,s} = \left(\frac{n!}{k!\,(n-k)!}\right)\left(\frac{n!}{s!\,(n-s)!}\right) \tag{5.1}$$

According to Eq. 5.1 the proposed PUF can be considered as a strong PUF. However, the aforementioned feature would potentially require additional circuitry, so as to apply the new challenge.

Note that the readout circuitry doesn't necessarily have to comprise of m-1 comparators. Instead, an m-to-1 multiplexer could be used (as in [16]), which drives a

single comparator. In that case, m challenges are applied, one after the other, in order to compose the m-bit response. This approach will effectively reduce the silicon area of the PUF.

As a future work, we may consider the calibration of the comparator aiming to face offset related issues. The difference $|\Delta V|$ between the two RBLs should be statistically analyzed, so as to select a suitable comparator with regards to its sensitivity. In addition, the influence of temporal noise as well as this of transistor aging on the PUF operation should be analyzed. Furthermore, error correction schemes based on majority voting topologies, commonly used in literature, may be explored.

Finally, the statistical analysis of the PUF response time can be conducted aiming to define the best operating conditions for the PUF. The measurements were eventually taken at 250ns, as it is an extreme case scenario to cover outliers due to process variations. It is expected that the PUF response is a lot faster, since the difference between the RBLs seems to be stable enough earlier than 200ns. In essence, however, the proposed topology provides a good starting point for the development of a low cost and strong PUF circuit.

REFERENCES

[1] B. Halak, *Physically Unclonable Functions, From Basic Design Principles to Advanced Hardware Security Applications*, Springer, 2018. available: https://doi.org/10.1007/978-3-319-76804-5

[2] F. Zerrouki, S. Ouchani and H. Bouarfa, "A survey on silicon PUFs", *Journal of Systems Architecture*, vol. 127, June 2022, available: https://doi.org/10.1016/j.sysarc.2022.102514

[3] C. Herder, M. -D. Yu, F. Koushanfar and S. Devadas, "Physical Unclonable Functions and Applications: A Tutorial," in *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1126-1141, Aug. 2014, doi: 10.1109/JPROC.2014.2320516

[4] W. Stallings, *Cryptography and Network Security, Principles and Practice,* 8th ed., UK: Pearson, 2022

[5] G.E. Suh, S. Devadas, "Physical unclonable functions for device authentication and secret key generation", in *2007 44th ACM/IEEE Design Automation Conference*, pp. 9–14

[6] M. Kalyanaraman, M. Orshansky, "Novel strong PUF based on nonlinearity of MOSFET subthreshold operation", in *2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pp. 13–18

[7] M.S. Mispan, B. Halak, Z. Chen, M. Zwolinski, "TCO-PUF: a subthreshold physical unclonable function", in Ph.D. Research in Microelectronics and Electronics (PRIME), 2015 11th Conference on (2015), pp. 105–108

[8] W. Xiong, A. Schaller, N.A. Anagnostopoulos, M.U. Saleem, S. Gabmeyer, S. Katzenbeisser, J. Szefer, "Run-time accessible dram pufs in commodity devices", presented at the *Cryptographic Hardware and Embedded Systems International Conference*, Santa Barbara, CA, USA, 2016

[9] S.S.K.J. Guajardo, G.-J. Schrijen, P. Tuyls, "FPGA intrinsic PUFs and their use for IP protection", in *International Conference on Cryptographic Hardware and Embedded Systems*, pp. 63–80, 2007

[10]N. Torii, D. Yamamoto, T. Matsumoto, "Evaluation of latch-based PUFs implemented on 40 nm ASICs", in *2016 Fourth International Symposium on Computing and Networking (CANDAR)*, pp. 642–648

[11]A. Stanciu, M.N. Cirstea, F.D. Moldoveanu, "Analysis and evaluation of PUF-based SoC designs for security applications", *IEEE Trans. Industr. Electron.* 63, 5699–5708, 2016

[12]H. Zhuang, X. Xi, N. Sun, M. Orshansky, "A strong subthreshold current array PUF resilient to machine learning attacks", *IEEE Transactions on Circuits and Systems-I: Regular papers,* Vol. 67, No. 1, Jan. 2020

[13]S. Kerckhof, F. Durvaux, C. Hocquet, D. Bol, F.-X. Standaert, "Towards Green Cryptography: a Comparison of Lightweight Ciphers from the Energy Viewpoint", in *Proceedings of the 14th international conference on Cryptographic Hardware and Embedded Systems*, Sep. 2012, available: https://link.springer.com/chapter/10.1007/978-3-642-33027-8_23

[14]T. Eisenbarth, Z. Gong, T. Güneysu, S. Heyse, S. Indesteege, S. Kerckhof, F. Koeune, T. Nad, T. Plos, F. Regazzoni, F.-X. Standaert, L. v. O. t. Oldenzeel, "Compact Implementation and Performance Evaluation of Block Ciphers in ATtiny Devices", in *Progress in Cryptology - AFRICACRYPT 2012*, Vol. LNCS 7374, pp 172–187, July 2012, available: https://link.springer.com/chapter/10.1007/978-3-642-31410-0_11

[15]N. H. L. Weste, D. M. Harris, *CMOS VLSI Design, A Circuits and Systems Perspective,* Fourth edition, Pearson, 2011

[16]J. Li, M. Seok, "Ultra-Compact and Robust Physically Unclonable Function Based on Voltage-Compensated Proportional-to-Absolute-Temperature Voltage Generators", in *IEEE Journal of Solid-State Circuits,* Vol. 51, No. 9, Sep. 2016

[17]S. Taneja, V. K. Rajanna, M. Alioto, "In-Memory Unified TRNG and Multi-Bit PUF for Ubiquitous Hardware Security", in *IEEE Journal of Solid-State Circuits,* Vol. 57, No. 1, Jan. 2022

[18]Z.-W. Lai, P.-H. Huang, K.-J. Lee, "Using both Stable and Unstable SRAM Bits for the Physical Unclonable Function", in *Journal of Electronic Testing*, Vol. 38, pp 511-525, Oct. 2022, available: https://link.springer.com/article/10.1007/s10836-022-06025-8

[19]F. Zhang, S. Yang, J. Plusquellic, S. Bhunia, "Current based PUF Exploiting Random Variations in SRAM Cells", in *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Dresden, Germany, 2016, pp. 277-280.

[20]A. Xynos, V. Tenentes, Y. Tsiatouhas, "SiCBit-PUF: Strong in-Cache Bitflip PUF Computations for Trusted SoCs", in *2023 IEEE European Test Symposium (ETS)*, Venezia, Italy, 2023, pp. 1-6, doi: 10.1109/ETS56758.2023.10173941.

[21] A. Xynos and V. Tenentes, "MetaSPICE: Metaprogramming SPICE Framework for the Design Space Exploration of PUF Circuits," *2023 12th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, Athens, Greece, 2023, pp. 1-4, doi: 10.1109/MOCAST57943.2023.10176643.

Short Biography

Fani Moka was born in 1997 in Köln, Germany, and completed her undergraduate studies in Physics at the University of Thessaloniki, in 2021. She established a path towards computer science with her undergraduate thesis "Canny edge detector in VHDL and Matlab". She pursued post-graduate studies in the Department of Computer Science and Engineering, at the University of Ioannina, so as to nurture the interest she had taken in that direction. She gravitated towards any hardware-related course the university had to offer, and especially analog and digital circuit design, and computer architecture. Simultaneously, she also found the software-related courses appealing, which led her to start pursuing a career in software development.