RRAM-based PUF: Design and Applications in Cryptography

by

Ayush Shrivastava

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved July 2015 by the
Graduate Supervisory Committee:

Chaitali Chakrabarti, Chair
Shimeng Yu
Yu Cao

ARIZONA STATE UNIVERSITY

August 2015

ABSTRACT

The recent flurry of security breaches have raised serious concerns about the security of data communication and storage. A promising way to enhance the security of the system is through physical root of trust, such as, through use of physical unclonable functions (PUF). PUF leverages the inherent randomness in physical systems to provide device specific authentication and encryption.

In this thesis, first the design of a highly reliable resistive random access memory (RRAM) PUF is presented. Compared to existing 1 cell/bit RRAM, here the sum of the read-out currents of multiple RRAM cells are used for generating one response bit. This method statistically minimizes any early-lifetime failure due to RRAM retention degradation at high temperature or under voltage stress. Using a device model that was calibrated using IMEC HfOx RRAM experimental data, it was shown that an 8 cells/bit architecture achieves 99.9999% reliability for a lifetime >10 years at 125℃. Also, the hardware area overhead of the proposed 8 cells/bit RRAM PUF architecture was smaller than 1 cell/bit RRAM PUF that requires error correction coding to achieve the same reliability.

Next, a basic security primitive is presented, where the RRAM PUF is embedded in the cryptographic module, SHA-256. This architecture is referred to as Embedded PUF or EPUF. EPUF has a security advantage over SHA-256 as it never exposes the PUF response to the outside world. Instead, in each round, the PUF response is used to change a few bits of the message word to produce a unique message digest for each IC. The use of EPUF as a key generation module for AES is also shown. The hardware area requirement for SHA-256 and AES-128 is then analyzed using synthesis results based on TSMC 65nm library. It is shown that the area overhead of 8 cells/bit RRAM PUF is only 1.08%

of the SHA-256 module and 0.04% of the AES-128 module. The security analysis of the PUF based systems is also presented. It is shown that the EPUF-based systems are resistant towards standard attacks on PUFs, and that the security of the cryptographic modules is not compromised.

Dedicated to my Parents

ACKNOWLEDGEMENTS

Towards the end of a very fruitful journey as Master of Science graduate from Computer Engineering (Electrical Engineering) department at Arizona State University there are few people to whom I would like to extend my sincere gratitude. First and foremost, I would like to express the deepest appreciation to my advisor and chair Dr. Chaitali Chakrabarti, who guided me through each and every step of my research. I would also like to thank my committee members, Dr. Kevin Cao and Dr. Shimeng Yu, for sharing their ideas and extending their support to make this thesis possible. I would like acknowledge National Science Foundation (NSF) for supporting this work.

In addition, I would like to extend my gratitude to all my colleagues specially Pai-Yu Chen, Rui Liu, Manqing Mao and Zihan Xu for helping me get through some extremely tough hurdles in my research. It was really wonderful experience working with them.

Lastly, I would like to thank my family, friends and colleagues at Arizona State University for believing in my dreams and providing unconditional support and help.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

CHAPTER 1

INTRODUCTION

Recent advances in electronics have enabled enormous successes in mobile devices and wearable medical devices on the one hand, and data storage and computing on the other hand. The unprecedented growth in the number, type and complexity of mobile devices has posed tremendous challenges in ensuring security and privacy during data communication and storage. In spite of all the security measures, the number of security breaches is on the rise. In November 2013, Target system was hacked and information of millions of credit and debit cards was stolen [1]; in August 2014, hundreds of private photographs of Hollywood celebrities were made public [2]; in November 2014, confidential data belonging to Sony Pictures Entertainment employees was leaked [3]. In 2014 alone, several Fortune 500 companies were hacked thereby raising significant concerns on the security of cloud data storage [4]. In 2017, it is expected that even in enterprise programs, half of the employees will bring their own personal devices to work, presenting the most radical change to the culture of mobile computing in decades [5].

To enhance the security of the system techniques based on physical root of trust have been proposed. Physical unclonable function or PUF has emerged as a promising way to provide physical root of trust. PUF is described as “an expression of an inherent and unclonable instance specific feature of a physical object such as a biometric feature like fingerprint of a human being” [6]. Ideally, a PUF is a function whose output cannot be mathematically derived but purely depends upon the device physical properties.

In this thesis we focus on silicon PUF [7] which leverages the inherent randomness in the semiconductor manufacturing process to produce a unique output. The input to the

PUF is referred to as ‘challenge’ and its output is referred to as ‘response’ [7], as shown in Figure 1. To use PUF as a successful security primitive, a high degree of uniqueness among the PUF instances has to be achieved. In other words, the same challenge should produce different responses in different ICs, i.e., Inter-Hamming distance should be high [8]. At the same time, the PUF should be highly reliable, i.e., it should produce the same response for the same challenge (Intra-Hamming Distance should be zero) across various temperature, voltage and aging scenarios [8].

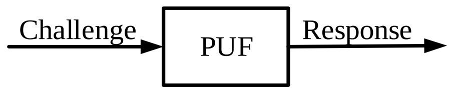


Figure 1: Basic PUF Operation

Based on the number of challenge response pairs (CRPs), PUFs are classified as weak PUFs and strong PUFs. PUFs having a large number of CRPs are termed as strong PUFs, whereas PUFs having a small number of CRPs are defined as weak PUFs [9]. Since weak PUFs have limited number of CRPs, they cannot be used for authentication. Due to their small size, the attacker is assumed to have limited access to tamper the device making them more resistant towards invasive attacks. In contrast, strong PUFs have large number of CRPs and can be used for authentication. But due to their large structure, they are more vulnerable to invasive attacks.

There are several implementations of PUFs, the most common being delay-based PUFs and memory PUFs. The arbiter PUF is a delay-based silicon PUF where the difference in the delay of the two paths is used to determine whether the output is a 0 or a 1 [10-11]. It is typically implemented with multiple switches, where the number of switches is equal to the number of challenge bits. While this PUF is easy to implement, it can be characterized by a linear delay model and the output response can be predicted [12-13]. To add non-linearity, XOR arbiter PUF [14] and lightweight arbiter PUF [15] have been introduced.

Memory-based PUFs are typically based on SRAMs. Here the output response of the circuit depends upon the startup values of the cross coupled inverters of a 6T SRAM cell [9]. Recently, emerging non-volatile memory-based PUFs have been proposed, e.g. the resistive RAM (RRAM) (also referred to as memristor) [16-22]. The RRAM PUF has the advantage of nanoscale dimension, compatibility with CMOS and high retention property.

PUFs are mainly used for authentication and key generation [14] as described below.

1.1 Authentication

As PUFs are based on manufacturing variation and provides a unique identity to an IC, they can be used for authentication. For authentication, the verifier stores the CRPs of the PUF in a database during enrollment phase. To check the authenticity of an IC, he selects a challenge whose response has been previously recorded but has never been used. He also obtains the response from the IC containing PUF. If the response from the chip matches (i.e., is close enough to) the previous recorded response, then the chip is successfully verified. This method is described in Figure 2.

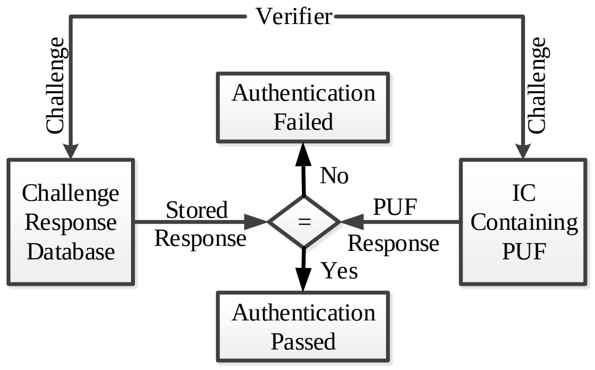


Figure 2: Overview of PUF based authentication

If a strong PUF is used, the system has a larger CRP space but the PUF is more vulnerable to modeling attacks [12-13]. Lightweight arbiter PUF, XOR arbiter PUF etc. alleviate some of these problems but these too are susceptible to machine learning attacks [12-13]. Conversely if a weak PUF [7] is used, then the system is immune to hardware probing attacks [12], but the CRP space is quite small and such a PUF cannot be used for authentication.

Several approaches have been proposed to hide the CRPs from the adversary at the expense of higher hardware cost. These include control PUF [23], where a strong PUF is used in conjunction with a cryptographic hash function, logically reconfigurable PUF [24] which uses an additional nonvolatile memory for reconfiguration, and reverse fuzzy extractor PUF [25] which uses helper data generator on top of the hash function and strong PUF.

1.2 Key Generation

PUFs can also be used for key generation. In this method a challenge is given to the PUF and its response is then passed through a cryptographic hash function. The cryptographic hash function helps in getting a fixed length key with sufficient entropy. Figure 3 describes the process of key generation.

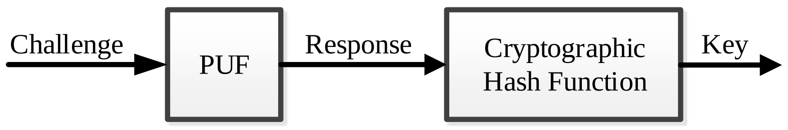


Figure 3: Overview of PUF for Key Generation

For the architecture shown in Figure 3 to work properly, the Intra-Hamming distance of the PUF should be zero. This is because even a small change in its output response will cause a large variation in the output of cryptographic hash function resulting in a wrong key generation. To compensate for the errors in the output response, Error Correction Codes (ECC) have been proposed [14].

For both key generation and authentication, cryptographic hash functions are used along with the PUF and so it is necessary for the PUF to be highly reliable. Designs have been proposed for a highly reliable PUF based on ring oscillator [26] and RRAM [20] but they do not guarantee 100% reliability.

In the ECC based system, part of the response has to be used as helper data. Thus use of ECC not only increases the hardware overhead but also reduces the entropy of the system. Such a system is vulnerable to non-invasive attacks which leverage the information leaked by helper data [27].

1.3 Thesis Contributions

In this thesis we propose the design of a highly reliable RRAM PUF which does not require ECC to guarantee its reliability. For the PUF implementation, we choose RRAM [15-21] because of its great scalability (<10nm), significant variability and superior compatibility with CMOS technology [28]. The initial resistance distribution of the RRAM cells (which leverages the intrinsic variations in a RRAM device) is taken as a randomness source for entropy. Based on comparisons with a reference current, the cells in the RRAM array are self-programmed into a bimodal distribution with sufficiently large resistance ratio. However, the resistance ratio may degrade with time due to operation at elevated temperature or due to read disturbance. To achieve 100% reliability, we propose an architecture where each PUF response bit is represented by multiple RRAM cells physically wired in parallel. This method reduces the probability of an early life failure of a PUF response bit by adding redundancy. We have analyzed this method by representing each PUF bit by 2 cells, 4 cells and 8 cells in parallel. Using RRAM device models calibrated with IMEC HfOx RRAM experimental data, we show that when each bit is represented by 8 parallel RRAM cells, we achieve $< 10^{-6}$ Intra-Hamming distance (or >99.9999% reliability) for a lifetime >10 years at 125°C.

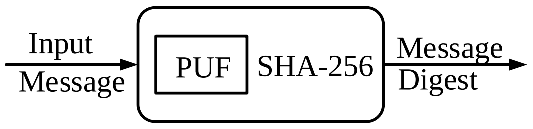


Figure 4: Embedded PUF: PUF embedded in SHA-256 datapath

We then present Embedded PUF (EPUF) which consists of an RRAM PUF embedded inside a SHA-256 [29] module as shown in Figure 4. SHA-256 belongs to the SHA2 family of cryptographic hash function which is widely used in security protocols. It consists of 64 rounds, where in each round (*t*) two parameters change: the 32 bit W_t which is a function of the input message and K_t which is a constant. In EPUF, the RRAM PUF response is used to change 2 bits of W_t in each round. Since the change does not affect the evaluation method, the resistance of SHA-256 to cryptoanalytic attacks is not compromised. The RRAM PUF has to be highly reliable as even a 1 bit change in the PUF characteristic will result in a 50% Hamming distance in the output response. The proposed RRAM PUF based on 8cells/bit meets this requirement.

EPUF design has the following advantages:

1. High Security: EPUF has a large CRP space even though it is based on an RRAM-based weak PUF. Since the PUF responses are not exposed to the communication channel, it is immune to modeling attacks. Use of the RRAM PUF makes it more secure against semi-invasive attacks compared to an SRAM PUF. In addition, EPUF has a security level of 2^{8x64} for random guessing attacks.
2. Low Area: The hardware overhead is due to the small RRAM array and other simple circuitry for integrating the RRAM-based PUF inside the hash module. Also as EPUF is based on an RRAM-based PUF which is very robust, there is no need for any error correction circuits. We show that the proposed multiple cell per bit RRAM PUF has lower area overhead compared to the one cell per bit RRAM PUF that requires Hamming code based ECC.

We also explain the use of EPUF as a key generation unit for AES-128 [30] which is used for data encryption. We synthesized EPUF and the AES module using TSMC 65nm library and calculated their area after doing place and route in Cadence Encounter 10.1. We see that the RRAM PUF area overhead is very small. For instance, it is only 1.08% of SHA-256 area and 0.04% of AES-128 area.

1.4 Thesis Organization

The rest of the thesis is organized as follows. Background information of different types of PUF and their properties are discussed in Chapter 2. Information about the architecture of SHA-256 and AES-128 is explained in Chapter 3. Architecture, simulation results and hardware overhead of a highly reliable RRAM PUF are presented in Chapter 4. The synthesis results for EPUF and AES-128, along with the security analysis of the complete system, are presented in Chapter 5. The conclusion and future work are included in Chapter 6.

CHAPTER 2

Background: Physical Unclonable Function

2.1 Basics of PUF and its Properties

Physical Unclonable Function or PUF is a security primitive that leverages the inherent randomness in the physical systems (e.g. the semiconductor manufacturing process) to produce unique responses (outputs) corresponding to challenges (inputs) [8]. It is also known as a physical one-way function as it is easy to compute but hard to invert.

Based upon the number of challenge response pairs (CRPs), the PUF are classified as strong PUFs or weak PUFs. Strong PUFs have large CRPs and are used for authentication; examples include arbiter PUF, ring oscillator PUF, etc. Weak PUFs have limited CRPs and are mainly used for key generation along with cryptographic modules. SRAM PUF and RRAM PUF are examples of weak PUF.

A PUF has the following properties:

1. Reproducibility: - Reproducibility means that response resulting from evaluating the same challenge on the same PUF instance should be similar i.e. the Intra-Hamming distance between the responses of the PUF should be negligible [8]. This is a very important property especially when PUF is used with cryptographic functions since even a small change in the response of the PUF will result in a large Hamming distance in the final output. This property should be checked across various voltage and temperature corners, as PUF response depends on the variability of the device and that variability can change across different process corners.

2. Uniqueness: - Uniqueness means responses resulting from evaluating the same challenge on different PUF instances should not be similar i.e. the Inter-Hamming distance

between different instances of PUF for the same challenge should be high [8]. This property is important in terms of security since if an attacker is able to predict the characteristics of PUF on one device, he will not be able to do that for other devices.

3. Identifiabilty: - If the PUF satisfies both reproducibility and uniqueness it makes it identifiable [8].

4. Unpredictability: - A PUF is said to be unpredictable if even after the knowledge of a large set of CRPs, the adversary is not able to predict the response for any random challenge [8]. This property is advantageous while performing authentication because if the PUF is not unpredictable, then an adversary can predict the outcome of a challenge based on a previously recorded response.

5. Mathematical Unclonability: -A PUF should be mathematically unclonable i.e. based on its previously recorded CRPs, the adversary should not be able to create a mathematical model for it [8]. This can be interpreted as the resistance of the PUF to machine learning attacks [12-13].

2.2 Types of PUFs

There are three major categories of PUF: Non electronic PUF, electronic PUF and silicon PUF. The non-electronic PUFs are built using non-electronic technologies or materials, for example optical PUF [31-32] which is based on random reflection of scattering characteristics of the optical medium. Electronic PUFs are built using the electronic characteristics of material such as resistance, capacitance and inductance values; an LC PUF [33] is an example of an electronic PUF. The third category is silicon PUF [7] which are integrated electronic circuits exhibiting PUF behavior embedded in a silicon chip. In this work we focus on silicon based PUF. The two major type of silicon PUFs are delay-based PUF and memory-based PUF.

2.2.1 Delay-Based PUF

In a delay-based PUF, the difference in the delay of the two paths is used to determine whether the response is low or high. It is considered as a strong PUF due to large number of CRPs and is mainly used for authentication of ICs. Next we describe several delay-based PUFs.

2.2.1.1 Arbiter PUF

The arbiter PUF [10-11] is a delay-based silicon PUF. An arbiter PUF consists of series of switches, where the number of switches is equal to the number of challenge bits. The arbiter decides whether the output is a 1 or 0 depending upon which path is faster. Figure 5 describes a simple arbiter PUF.

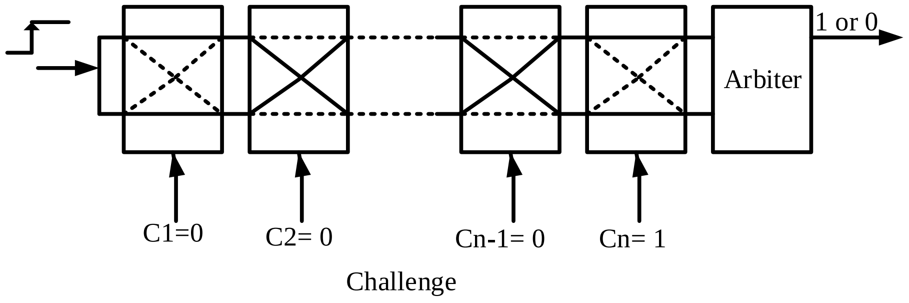


Figure 5: Arbiter PUF

The main disadvantage of the arbiter PUF is the simplicity of its model. The attacker can characterize the arbiter as a linear delay model and therefore can predict the output response of the PUF [12-13]. To add non linearity to the arbiter PUF, variants such as XOR arbiter PUF and lightweight arbiter PUF have been introduced.

2.2.1.2 XOR Arbiter PUF

The XOR arbiter PUF [14] has non linearity and so cannot be characterized using a linear delay model. The *n*-XOR arbiter PUF consists of *n* different arbiter PUFs in parallel; the outputs are XORed to get a one bit response [34]. Unfortunately, machine learning can be used to model this type of PUF.

2.2.1.3 Lightweight Arbiter PUF

Lightweight arbiter PUF was introduced in [15]. It resembles the XOR arbiter PUF as it consists of *n* parallel chain of switches whose outputs are XORed to get one bit response. While in a XOR arbiter PUF, the same challenge is given to all the parallel stages, in a lightweight arbiter PUF, a different challenge is given to each parallel stage; these challenges are derived from one fixed challenge. In addition, arbiters are used in an intermediate stage to generate a challenge bit for some of the intermediate switches as shown in Figure 6. In spite of significant hardware overhead, this PUF can also be modelled by machine learning with the help of side channel information [35-36]

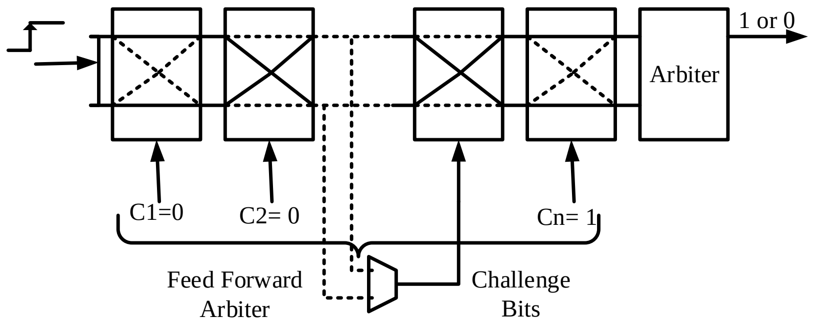


Figure 6: Lightweight Arbiter PUF

2.2.1.4 Ring Oscillator PUF

Ring oscillator PUF [37] is made up of multiple ring oscillators as shown in Figure 7. Each ring oscillator is a simple circuit that oscillates with a particular frequency. Due to manufacturing variations, each ring oscillator oscillates with a slightly different frequency. In order to generate a fixed number of bits, a fixed sequence of oscillator pairs is selected, and their frequencies are compared to generate an output bit. The output bits generated by the same sequence of oscillator pair, vary from chip to chip. Given that oscillators are identically laid out, the frequency differences are determined by manufacturing variations and an output bit is equally likely to be one or zero if random variations dominate. Ring Oscillator PUFs are also vulnerable to machine learning attacks because of their linear structure [12][27].

Another issue with delay-based PUF is reliability. Architectures have been proposed to make them highly reliable [26] but none of them guarantee 100% reliability.

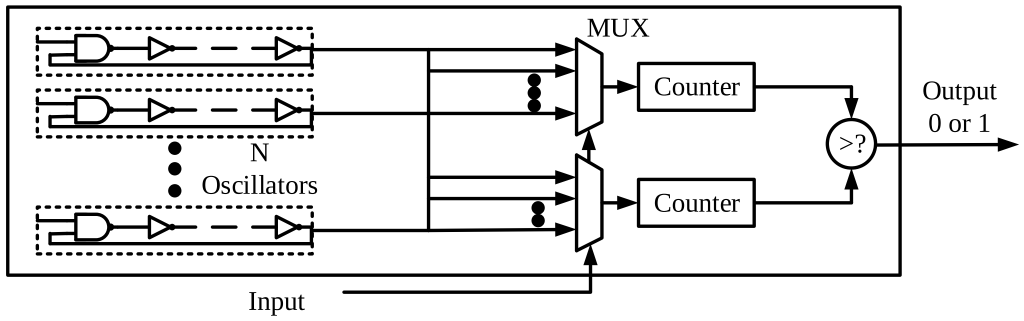


Figure 7: Ring Oscillator PUF

2.2.2 Memory-Based PUF

To increase the reliability of the system, memory-based PUFs have been introduced. They are considered as weak PUFs due to the number limited CRPs. The memory-based PUF is mainly used for key generation along with cryptographic hash functions as discussed in [14]. Error Correction Code (ECC) mechanism is typically required in order to generate the same key, as no errors can be tolerated at the input of the hash function.

2.2.2.1 SRAM PUF

SRAM-based PUF is one of the most commonly used memory-based PUFs [9]. The output response of the SRAM PUF depends upon the startup values of the cross coupled inverters, as shown in Figure 8. The startup values differ across different ICs due to the manufacturing variations. Methods have been proposed to increase the reliability of SRAM PUF by creating a mismatch between the length and threshold voltage of NMOS in the cross coupled inverters. Unfortunately the system still does not achieve Intra-Hamming distance of 0% [38].

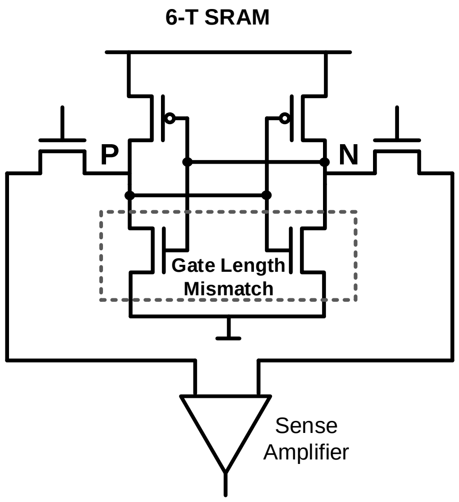


Figure 8: SRAM-based PUF

2.2.2.2 RRAM PUF

Resistive random access memory (RRAM/ReRAM) is a non-volatile memory whose structure is a metal/oxide/metal stack as shown in Figure 9. It can be integrated at the interconnect levels on top of CMOS circuits and are generally used as a cross point array structure [39] or as a 1T1R (1 transistor 1 resistor) structure [40]. The RRAM differentiates between a high resistance state (HRS, or “0”, or off-state) and a low resistance state (LRS, or “1”, or on-state) based upon the tunneling gap between the electrode and tip of the residual filaments that are made of oxygen vacancies. Despite being more reliable than the other PUFs, RRAM PUFs cannot guarantee 100% reliability, as shown in section 4.3.

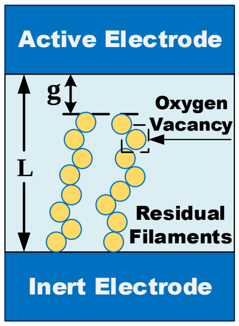


Figure 9: Basic structure of an RRAM cell with two electrodes and oxide in between [41]

2.2.3 Summary of PUFs

Table I summarizes the properties of the different PUFs. Even though the PUF constructions satisfies the properties of reproducibility and uniqueness up to a certain extent, they do not exhibit 0% Intra-Hamming distance or 50% Inter-Hamming distance. Also all the delay-based PUFs are prone to machine learning attacks.

	Arbiter PUF	XOR Arbiter PUF	Lightweight Arbiter PUF	Ring Oscillator PUF	SRAM PUF	RRAM PUF
Reproducibility	Yes	Yes	Yes	Yes	Yes	Yes
Uniqueness	Yes	Yes	Yes	Yes	Yes	Yes
Identifiabilty	Yes	Yes	Yes	Yes	Yes	Yes
Unpredictability	Yes	Yes	Yes	Yes	Yes	Yes
Mathematical Unclonability	No	No	No	No	NA	NA

Table I: Summary of the PUF properties

2.3 Attacks on PUFs

Some of the common non-invasive security attacks on PUFs are as follows:

1. Replay Attack: This attack is a cause of concern when the PUFs are used for authentication [6]. In this attack, the adversary can use a previously used CRP for false authentication. To prevent this attack, the PUF should have a large CRP space so that the verifier has an option to never use a previously used CRP.

2. Chosen Challenge Attack: In this attack, the correlation between different CRPs is exploited [6]. The attacker tries to model the PUF by changing one bit of the challenge at a time and track the variations in the response of the PUF. To prevent this attack, the PUF should satisfy the unpredictability property.

3. Machine Learning Attack: In this attack, the adversary tries to build a mathematical model of the PUF based on previously recorded CRPs [12-13]. To increase the resistance against such attacks, nonlinearities are added into the PUF architecture but it affects the reproducibility property of the PUF. Also machine learning attacks have been shown to be effective against highly nonlinear PUFs as well. Machine learning attacks along with the side channel information can be used to model the PUF at a much faster rate.

In Chapter 4 we describe a RRAM-based PUF which is shown to be immune against these standard attacks.

CHAPTER 3

Background: SHA-256 and AES-128

3.1 Secure Hash Algorithm (SHA) 256

Cryptographic hash functions [42] has been traditionally used for digital signature, public key encryption etc. They are also used to hash the PUF response so that it is in not exposed to the outside world [23]. A cryptographic hash function has the following properties [42].

1. Pre-image Resistant: - Pre-image resistant property defines the one way nature of the hash function. It states that if there is an output message digest h of a hash function, then it should be difficult to find any message m such that h = hash(m).
2. Second pre-image Resistant: - A cryptographic hash function is said to be second pre-image resistant if for a given input message m_1 and its output message digest (hash(m_1)) it is difficult to find a different input message m_2 such that hash(m_1) = hash(m_2).
3. Collision Resistant:- A cryptographic hash function is said to be collision resistant if an adversary is not able to find two different input messages m_1 and m_2 such that hash(m_1) = hash(m_2).

A cryptographic hash function should be immune towards attacks on its basic properties and should have minimum hardware and computational overhead. Earlier cryptographic hash functions such as MD5 and SHA1 have been proven to be vulnerable against crypto-analytic attack on their collision resistance property and therefore we have chosen SHA-256 [29] which belongs to the SHA2 family for our implementation. SHA-256 has an output message digest (output message length) of 256 bits and operates on a word size of

32 bits. The hardware architecture for SHA-256 is based on Merkle- Damgard construction [43] as shown in Figure 10.

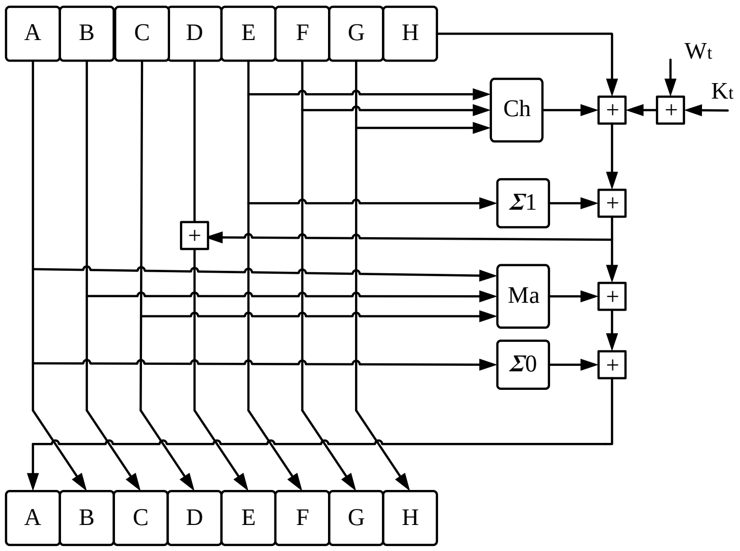


Figure 10: Merkel-Damgard Architecture of SHA-256

There are 64 rounds in SHA-256. Registers A,B,C,D,E,F,G,H are loaded with predetermined 32 bit constants H_0 to H_7 as shown in Table II; their values get updated in every round during the computation. The computations are defined by the following equations

$$Ch(E,F,G) = (E \wedge F) \oplus (\neg E \wedge G) \quad (3.1)$$

$$Ma(A,B,C) = (A \wedge B) \oplus (A \wedge C) \oplus (B \wedge C) \quad (3.2)$$

$$\textstyle\sum_0(A) = (A \ggg 2) \oplus (A \wedge C) \oplus (B \wedge C) \quad (3.3)$$

$$\textstyle\sum_1(E) = (E \ggg 6) \oplus (E \ggg 11) \oplus (E \ggg 25) \quad (3.4)$$

The addition performed here is modulo 2^{32} and is represented by $\boxed{+}$ in Figure 10, i.e., the final carry is dropped after performing the addition.

Initial Constants	Values (Hexadecimal)
H_0	0x6a09e667
H_1	0xbb67ae85
H_2	0x3c6ef372
H_3	0xa54ff53a
H_4	0x510e527f
H_5	0x9b05688c
H_6	0x1f83d9ab
H_7	0x5be0cd19

Table II: The initial constants values used in SHA-256 computation

The W_t register shown in Figure 10 is of 32 bits and is computed in every round *t*. To compute W_t, first the input message is converted to a length of 512 bits as follows. If length of the input message is *L*, a 1 is appended after the message and *k* zeroes are appended after that such that *L*+1+*k*=448. To make the total width 512 bits, the remaining 512- 448= 64 bits are obtained by representing the value of *L* in binary. These 512 bits are used as W_t for the first 16 rounds. To compute W_t for round number 17 to 64 the following equation is used

$$\mathrm{W_t}=\sigma_1\ (\mathrm{W_{t-2}})+\mathrm{W_{t-7}}+\sigma_0(\mathrm{W_{t-15}})+\mathrm{W_{t-16}} \qquad (3.5)$$

where σ_0 and σ_1 are constants for SHA-256. Previously computed W_t values are multiplied by these constants to get the current value of W_t as shown in equation 3.5.

The round constant K_t is also of 32 bits and have the predefined values provided by NIST given in [29].

All the values of K_t satisfies the following properties:

- The value of round constants should be pairwise distinct
- The round constant values should not be too symmetric.

In this thesis we have kept the original architecture of SHA-256 and introduced an embedded PUF in its data path as discussed in Chapter 5.

3.2 Security Analysis of SHA-256

As the basic structure of SHA-256 is kept unchanged, its immunity against crypto-analytic attacks on its properties namely, pre-image resistance, second pre-image resistance and collision resistance, remains the same [44]. So far, no crypto-analytic attack or side channel attack has been able to break these properties when all 64 rounds are computed.

3.3 Advance Encryption Standard (AES) 128

The Advance Encryption Standard (AES) also known as Rijndael, is a specification for encryption given by NIST [30]. It is a symmetric block cipher as it maps plaintext blocks to cipher text blocks and the same key is used for both encryption and decryption. The size of both the plaintext and cipher text blocks is 128 bits in AES. In this thesis we focus on AES-128 which requires the cipherkey also to be of 128 bits.

A data block of 16 bytes in AES is referred to as a state. Operations in AES are performed on basic units of 8 bits, one byte. All bytes are interpreted as elements of the finite field GF (2^8). This ensures that the results of all multiplications and additions also are elements of the same finite field.

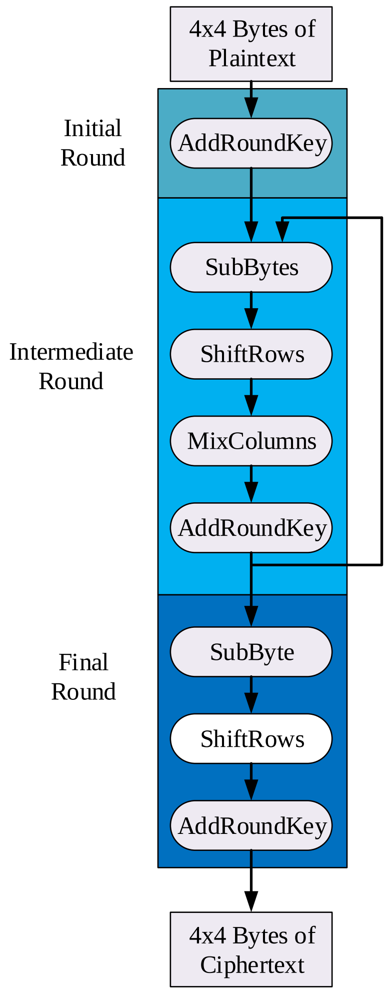


Figure 11: Block diagram representing complete flow of AES-128 [30]

AES-128 consists of 10 rounds of operation. There are four basic operations used in AES:

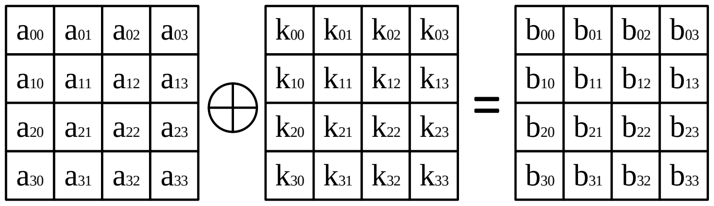
1. AddRoundKey
2. ShiftRows
3. MixColumns
4. Sub Bytes

The first round only consists of AddRoundKey step; the intermediate rounds from 2 to 9 consist of all the four steps and the final round consists of all the steps except the

MixColumns step. Figure 11 gives an overview of how encryption is performed. The original cipherkey of 128 bits is expanded so that each round of operation can operate on 128 bits of expanded key as described in section 3.3.5 .The details of all the rounds are given from section 3.3.1 to section 3.3.4.

3.3.1 AddRoundKey

In AddRoundKey step, each byte in the state is XOR'ed with a corresponding byte in the expanded key as shown in Figure 12. The expanded key is derived from the cipherkey according to the key schedule algorithm described in section 3.3.5.

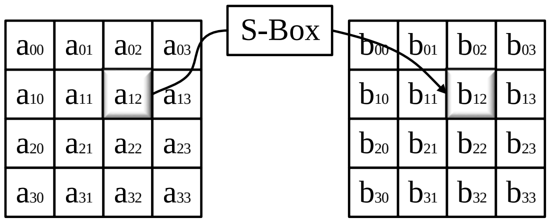


a_{00} a_{01} a_{02} a_{03}
a_{10} a_{11} a_{12} a_{13}
a_{20} a_{21} a_{22} a_{23}
a_{30} a_{31} a_{32} a_{33}
$\oplus$
k_{00} k_{01} k_{02} k_{03}
k_{10} k_{11} k_{12} k_{13}
k_{20} k_{21} k_{22} k_{23}
k_{30} k_{31} k_{32} k_{33}
=
b_{00} b_{01} b_{02} b_{03}
b_{10} b_{11} b_{12} b_{13}
b_{20} b_{21} b_{22} b_{23}
b_{30} b_{31} b_{32} b_{33}

Figure 12: AddRoundKey Step in AES-128 [45]

3.3.2 SubBytes

The SubBytes operation substitutes the state, one byte at a time, using a substitution box known as the Rijndael S-box. Figure 13 illustrates the SubBytes operation. The operation provides the non-linear property of the cipher which is crucial for protection against differential and linear cryptanalysis [46].



S-Box
a_{00} a_{01} a_{02} a_{03}
a_{10} a_{11} a_{12} a_{13}
a_{20} a_{21} a_{22} a_{23}
a_{30} a_{31} a_{32} a_{33}
b_{00} b_{01} b_{02} b_{03}
b_{10} b_{11} b_{12} b_{13}
b_{20} b_{21} b_{22} b_{23}
b_{30} b_{31} b_{32} b_{33}

Figure 13: SubBytes step in AES-128 [30]

3.3.3 ShiftRows

In ShiftRows operation, each row of the state is subjected to a different order of rotational left shift. Figure 14 illustrates the ShiftRows operation in detail where the row 0 is not subjected to any shift; row 1, row 2 and row 3 and shifted by a rotational left shift of 1, 2 and 3, respectively. This step introduces inter column diffusion to the algorithm which provides resistance against differential and linear cryptanalysis [46].

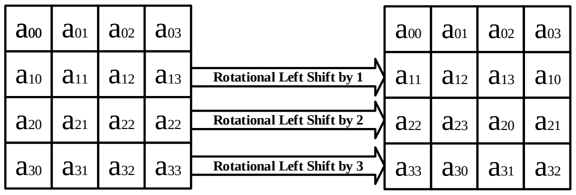
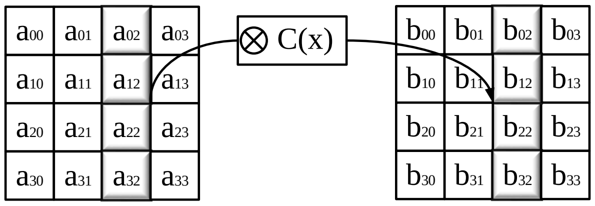


Figure 14: ShiftRows step in AES-128 [30]

3.3.4 MixColumns

MixColumns performs a transformation of the state, column by column. Each column is interpreted as a polynomial with coefficients in GF (2^8) and is then multiplied by modulo $x^4 + 1$ with a fixed polynomial c(x) = {03}x^3 + {01}x^2 + {01}x + {02}. This operation can be written as the matrix multiplication given in Equation 3.6. Figure 15 illustrates the MixColumn operation.

$$\begin{bmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} a_{0,j} \\ a_{1,j} \\ a_{2,j} \\ a_{3,j} \end{bmatrix} \tag{3.6}$$



a_{00} a_{01} a_{02} a_{03}
a_{10} a_{11} a_{12} a_{13}
a_{20} a_{21} a_{22} a_{23}
a_{30} a_{31} a_{32} a_{33}
$\otimes$ C(x)
b_{00} b_{01} b_{02} b_{03}
b_{10} b_{11} b_{12} b_{13}
b_{20} b_{21} b_{22} b_{23}
b_{30} b_{31} b_{32} b_{33}

Figure 15: MixColumn step in AES-128 [45]

3.3.5 Key Expansion Algorithm

The key expansion algorithm [30] is used to derive the expanded key from the original cipher key *K*. In this algorithm, the number of columns in a state is denoted by N_b, the number of rounds is denoted by N_r and the number of columns in the cipher key is denoted by N_k. For AES-128, the value of N_b, N_r and N_k is 4, 10 and 4, respectively. The key expansion generates a total of N_b (N_r + 1) words: the algorithm requires an initial set of N_b words, and each of the N_r rounds requires $N_b$words of key data. The resulting key schedule consists of a linear array of 4-byte words, denoted w[i], with i in the range $0 \leq i < N_b(N_r + 1)$.The expansion of the input key into the key schedule proceeds according to the pseudo code shown in Figure 16.

In Figure 16, SubWord() is a function that takes a four-byte input word and applies the S-box to each of the four bytes to produce an output word. The function RotWord () takes a word $[a_0, a_1, a_2, a_3]$ as input, performs a cyclic permutation, and returns the word $[a_1, a_2, a_{3\,,}a_0]$. The round constant word array, Rcon[i], contains the values given by

```
KeyExpansion (byte key [4*$N_k$], word w [$N_b$*(Nr+1)], $N_k$)
begin
  word temp
  i = 0
  while (i < $N_k$)
    w[i] = word(key[4*i], key[4*i+1], key[4*i+2], key[4*i+3])
    i = i+1
  end while
  i = $N_k$
 while (i < $N_b$ * ($N_r$ +1)]
   temp = w[i-1]
   if (i mod $N_k$ = 0)
     temp = SubWord(RotWord(temp)) xor Rcon[i/$N_k$]
   else if ($N_k$ > 6 and i mod $N_k$ = 4)
     temp = SubWord(temp)
   end if
   w[i] = w[i-$N_k$] xor temp
   i = i + 1
 end while
end
```

Figure 16: Key Expansion Algorithm [30]

[x^{i-1}, {00}, {00}, {00}], with x^{i-1} being power of x (x is denoted as {02}) in the field GF (2^8). From Figure 16, it can be seen that the first N_k words of the expanded key are filled with the Cipher Key. Every following word, w[i], is equal to the XOR of the previ-ous word, w [i-1], and the word N_k positions earlier, w [i-N_k]. For words in positions that are a multiple of N_k, a transformation is applied to w [i-1] prior to the XOR, followed by an XOR with a round constant, Rcon[i]. This transformation consists of a cyclic shift of the bytes in a word (RotWord ()), followed by the application of a table lookup to all four bytes of the word (SubWord ()).

3.4 Security Analysis of AES-128

AES-128 has been shown to be immune against non-invasive linear and differential cryptanalysis attacks [47]. However side channel attacks, which are based on the side channel information extracted from the physical implementation, are still of concern. The side channel information may consist of output timing, power consumption and electro-magnetic radiation.

There are two main types of side channel attacks (SCA) [48]: Simple SCA and Differential SCA. Simple SCA depends on the encryption process and individual bits of the secret key. This kind of attack requires detailed information about the encryption device and can be prevented easily. Timing attack is an example of Simple SCA. Differential side channel attacks are more complex and effective than Simple SCA [49]. These attacks try to extract a correlation between the data and the instantaneous side channel leakage of the cryptographic device. As this correlation is very small, statistical methods such as Pearson correlation coefficient are used. In a differential SCA, the adversary builds a speculative model of the device and breaks the target (intermediate result related to the secret key) into small parts, usually of size 8 bits to perform the attack. The value of each part is then guessed and the results are statistically analyzed.

Differential power analysis (DPA) attack [49] is a popular differential SCA. In AES-128, DPA is done on either the initial round of computation or the last round of computation [50-52]. The attack on the initial round is more efficient, as the initial round consists of an XOR operation of the plaintext and the original key. If an adversary can extract the key from the first operation itself, he can predict the key for the rest of the rounds as well [51]. Generating keys from the PUF does not add any complexity to this round; its vulnerability against DPA remains the same.

The last round is also targeted for DPA because it does not have the Mixcolumn vertical diffusion step [51]. If an adversary is able to predict the final round key, he can predict the initial cipher key by reversing the key expansion algorithm. Adding PUF in the shift row operation can possibly increase the complexity of the final round and its resistance towards DPA attacks. This work is not presented in this thesis and is open to further research.

Additionally DPA attacks are done on the S-box substitution used in the SubBytes step. As the S-Box substitution is performed one byte at a time, the adversary can target each byte separately [53]. This attack can be repeated 16 times using the same power traces to get the 128 bit key. If successful, this attack reduces the random guess key search space from 2^{128} to 16x28 combinations. To prevent this, hardware architectures have been proposed for S-box which show minimal power correlation with the operations performed [45]. Discussion of those architectures is beyond the scope of this thesis.

Other possible attacks on AES-128 are semi-invasive and invasive attacks on the key storage. Photoemission attacks on SRAM memory can be used to detect the stored values. Passive probing attacks can be used to leak the key values in non-volatile memories. Generating keys from PUFs can possibly help in reducing these type of attacks.

CHAPTER 4

Reliable RRAM PUF

Since PUFs are used with cryptographic hash function for doing authentication as well as key generation, there is a need to guarantee a highly stable PUF output. Studies have been made to make PUFs highly reliable but none of them guarantee 100% reliability. Currently the only approach that has a 0% Intra-Hamming distance involves using ECC on PUF output. Such a scheme not only reduces the entropy of the PUF but compromises the security level of the system. In this chapter we present the architecture of a highly reliable RRAM based PUF which can guarantee a stable output without the use of ECC.

4.1 Basics of RRAM PUF

RRAM is an emerging NVM candidate due to its simple structure, low programming voltage (<3 V), fast switching speed (<10 ns), high on/off ratio (>10×), excellent scalability (<10 nm), good programming endurance (10^6-10^{12} cycles) and great compatibility with silicon CMOS technology [18]. It has large variability, making it challenging for use in some memory applications. Here we leverage this variability in designing an RRAM-based PUF. Compared to other NVM candidates such as phase change memory (PCM), or spin-transfer-torque RAM (STT-RAM), RRAM has more stable resistance states and larger on/off resistance ratio (therefore, larger noise margin for better reliability). For example, RRAM's on/off resistance ratio can be as large as 10× ~ 100× (while STT-RAM's on/off ratio is only ~ 2×).

The typical RRAM device structure is a metal/oxide/metal stack as shown in Figure 9, which can be integrated at the interconnect levels on top of the CMOS circuits. The RRAM differentiates between a high resistance state (HRS, or "0", or off-state) and a low resistance state (LRS, or "1", or on-state) based upon the tunneling gap between the electrode and tip of the residual filaments that are made of oxygen vacancies. When the tunneling gap is large, the RRAM is in HRS and when the tunneling gap is very small, the RRAM is in LRS. Due to the stochastic nature of the atomistic processes, e.g., generation and annihilation of oxygen vacancies, the shape of the filaments varies from device to device, and even from programming cycle to cycle within one device. Owing to the tunneling mechanism, any atomistic change of the filaments results in significant variations in the resistances of the device, which provides sufficient entropy for the PUF application.

4.2 Read and Retention Property

To evaluate the RRAM's reliability when it is used as PUF, we need to understand its failure mechanism, e.g. data retention at high temperature and read disturbance under voltage stress. To analyze its retention property, the RRAM cell is subjected to high temperature baking. Over a period of time, oxygen vacancies in the oxide tend to diffuse thereby increasing the tunneling gap. As a result, the LRS cells shifts towards HRS. In addition, when the RRAM cell is frequently read, it suffers from read voltage disturbance.

To evaluate the disturbance property of RRAM, a constant read voltage is applied across the cell, which causes the oxygen vacancies to migrate thereby causing a variation in its resistance. When a negative voltage is applied across the RRAM, the tunneling gap tends to be widened, leading to an increase in the resistance of the cell. A reverse phenomenon happens when a positive voltage is applied across the RRAM.

4.3 Prior Work on RRAM PUF

Most of the prior work on RRAM PUF has focused on attaining a high level of uniqueness. The method in [54] leverages a weak write mechanism on the RRAM PUF that results in an unpredictable logic state, thereby improving the uniqueness metric. Memristor-based PUF, which depend upon the write time variability of the memristor device, was proposed in [55] [17]. The actual SET time was chosen to be close to the minimum SET time so that the probability of the output logic being high or low is 0.5.

Recently the focus has shifted towards designing a highly reliable RRAM PUF. The method in [20] uses the bimodal distribution of RRAM and programs the RRAM cells such that 50% of the cells are in LRS and 50% in HRS. To split the cells into LRS and HRS state equally, an on chip voltage to digital converter and a median finding algorithm is used in [20]. Such a method has high overhead but still does not guarantee 100% reliability. In this thesis we embed the PUF in cryptographic module to enhance the uniqueness metric and propose a multi bit per cell architecture with minimal area overhead to achieve high reliability.

4.4 Modeling RRAM

To assess the RRAM reliability, a physical device model [56-57] calibrated with IMEC HfOx RRAM experimental data [58-60], is used. In the model, the RRAM I-V relationship is described by a tunneling mechanism with gap distance (g) as an internal state variable,

$$I = I_0 \exp\left(-\frac{g}{g_0}\right) \sinh\left(\frac{V}{V_0}\right) \qquad (4.1)$$

where I_0, g_0 and V_0 are fitting parameters. The RRAM switching dynamics are represented via the following equations,

$$\frac{dg}{dt} = -v_0 \left[\exp\left(-\frac{qE_{ag}}{kT}\right)\exp\left(\frac{\gamma a_0}{L}\frac{qV}{kT}\right) - \exp\left(-\frac{qE_{ar}}{kT}\right)\exp\left(-\frac{\gamma a_0}{L}\frac{qV}{kT}\right)\right] \quad (4.2)$$

$$\gamma = \gamma_0 - \beta\left(\frac{g}{g_1}\right)^3 \quad (4.3)$$

$$\frac{dT}{dt} + \frac{T-T_0}{\tau_{th}} = \frac{|V\times I|}{C_{th}} \quad (4.4)$$

where dg/dt is the gap growth/dissolution velocity. L is the oxide thickness, a_0 is the atomic hopping distance and γ is the g-dependent local field enhancement factor. v_0, γ_0, β and g_1 are fitting parameters. Equation (4.4) describes the heat conduction process where T is the local temperature of the conductive filaments, T_0 is the ambient temperature, and τ_{th} and C_{th} are the effective thermal time constant and thermal capacitance, respectively.

4.5 Initial Programming and Architecture for a Highly Reliable RRAM PUF

Our primary objective is to design a highly reliable PUF with large entropy. The resistance distribution shown in Figure 17(a) illustrates the randomness that exists in the initial resistance of the as-fabricated RRAM cells in an array. This distribution represents the entropy source for the RRAM PUF.

A read voltage is applied across each RRAM cell and the current through it is measured. A reference current is chosen within the distribution. The cells which have current below the reference level are programmed into HRS and the others are programmed into LRS. Figure 17(b) shows the histogram after each cell is programmed into the LRS and HRS states. Allowing the reference current level to have some flexibility (instead of exactly in the middle of the distribution as in [19]) saves a lot of additional hardware and calibration steps. However, this flexibility also causes the probability of each bit being 0 and 1 to be no longer 50%, which may reduce the inter Hamming distance among different PUFs. In applications where the PUF is used for key generation, this is acceptable since the PUF output is hashed using a cryptographic hash function which provides an inter Hamming distance of 50% to the whole system.

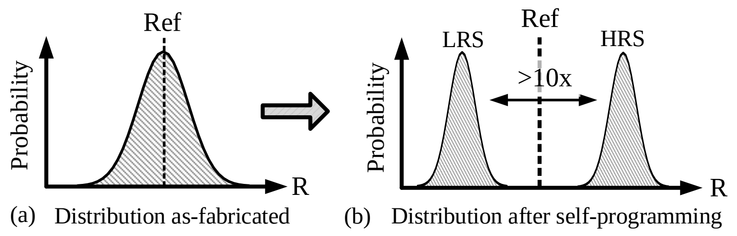

Ref
Probability
R
(a) Distribution as-fabricated
Ref
LRS
HRS
>10x
Probability
R
(b) Distribution after self-programming

Figure 17 : (a) Initial distribution of resistances in an RRAM array, (b) Distribution after the cells are programmed into the LRS and HRS according to the reference.

After each cell is programmed into the LRS and HRS states, the window between the two states is sufficiently large. However, under high temperature conditions over a long period of operation, the resistance distribution of LRS tends to shift towards that of HRS. If there is overlap between LRS tail bits and HRS tail bits, then errors occur in the PUF output response. The criterion for reliability assessment in this work is that the PUFs output should be reproducible for > 10 years (3×10^8 seconds) under military temperature condition of 125°C.

Figure 18(a) shows the PUF architecture when a single response bit is represented by one RRAM cell (baseline in this paper). To enhance the reliability of the PUF, we propose an architecture where a single response bit is represented by multiple RRAM cells connected in parallel, as shown in Figure 18(b). The multiple RRAM cells are physically wired together to form one column output. This architecture statistically reduces the probability of an early life failure for a single PUF response bit. We further analyze this architecture for structures where a single bit is represented by 2 cells, 4 cells and 8 cells.

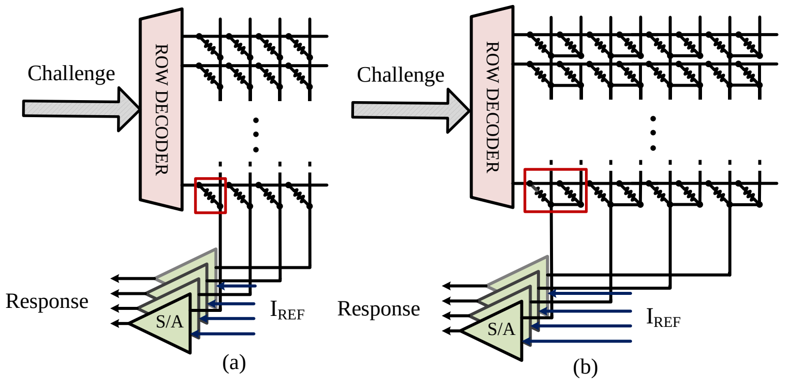


Figure 18 : (a) Design of RRAM crossbar array where each PUF bit is represented by one RRAM cell, (b) Design of RRAM crossbar array where each PUF bit is represented by multiple (e.g. 2) RRAM cells.

4.6 Reliability Analysis of RRAM PUF

For statistical analysis, we run 1 million Monte-Carlo (MC) simulations. We considered variations in the fitting parameters in the I-V characteristics (Equation 4.1), gap growth/dissolution velocity of the RRAM cell (Equation 4.2) and the local field enhancement factor (Equation 4.3). The variations have been chosen such that the resistance values remain in accordance with IMEC HfOx RRAM data. Table III lists the parameters that were varied along with their mean and sigma values. All simulations were performed using MATLAB R2013a on a workstation (Intel i-7 2.4 GHz CPU with 8 cores and 256 GB RAM). The simulations took almost 300 hours to complete.

	Mean: μ	Variance: σ/μ
gap0_L	0.403 nm	3.5%
gap0_H	1.367 nm	5%
I_0	61.4 μA	5%
V_0	0.43 V	5%
g_0	0.275 nm	5%
v_0	300 m/s	10%
γ_0	16.5	2%

Table III : Variations in the fitting parameters of I-V characteristics, local field enhancement factor and the gap dynamics of the RRAM model for Monte Carlo simulations

4.6.1 Analysis of 1 cell architecture

Figure 18 (a) shows the architecture when one PUF bit is represented by one RRAM cell. In this architecture, we define failure point as the read-out current value at which the sense amplifier cannot distinguish between the LRS and the HRS states. Specifically, for a single cell configuration, the failure point corresponds to the case when the LRS read-out current drops to 3μA.

4.6.1.1 Retention property

When the RRAM PUF is not being used, the RRAM states should retain the values even under extreme temperature conditions. Figure 19 shows the simulated RRAM retention degradation without voltage bias at 125°C for 1000 MC runs. The LRS current decreases over time as the LRS resistance shifts towards the HRS. No failure is observed for 1000 MC runs. However, as we increase the MC runs, we see that 3 cells among 1 million cells fail before the 10-year lifetime, which gives an error percentage of 0.003% as shown in Table IV.

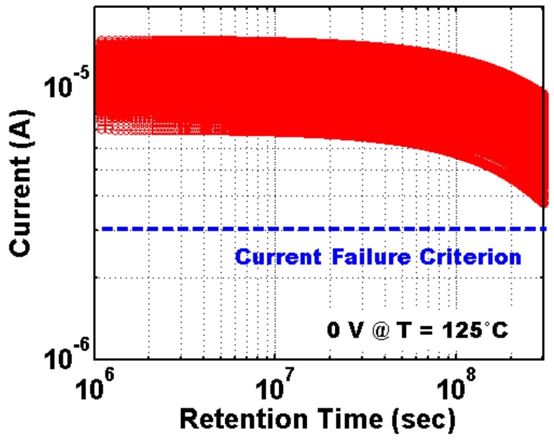


Figure 19 : LRS read-out current drift of a single cell RRAM in retention mode for a period of 10 years (3×10^{8} seconds) at 125°C. There are no failures in 1000 MC runs.

4.6.1.2 Read disturbance property

Every time the response from the RRAM PUF is read, the RRAM cells undergo read disturbance. We perform simulations corresponding to a constant read stress of 10^{4} seconds which is equivalent to reading the RRAM cell 10^{12} times assuming the read pulse is 10 ns. If the RRAM cell is read once every 0.3 ms, a constant stress of 10^{4} seconds can be translated to a 10-year lifetime. Figure 20 shows the simulated RRAM read disturbance with -0.3V read voltage at 125°C for 1000 MC runs. For 1 million MC runs, 17979 cells fail before the 10-year lifetime, which gives an error percentage of 1.7979% as shown in Table III. An error rate of 1.7979% is unacceptable for key generation. To enhance the reliability, in the next section, we propose an architecture in which a single bit is represented by multiple RRAM cells in parallel.

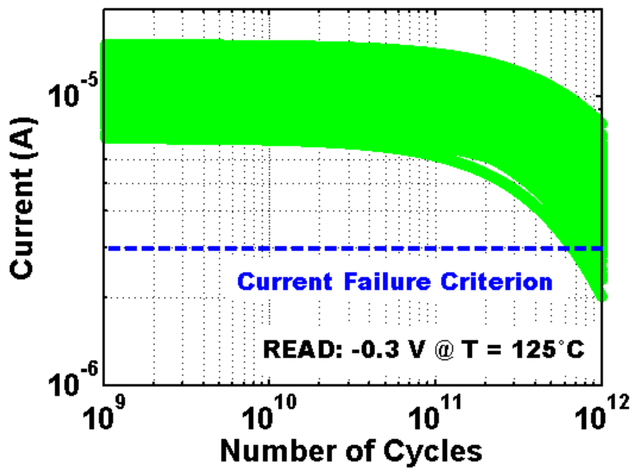


Figure 20: LRS read-out current drift of a single RRAM cell when a constant read stress voltage of -0.3V is applied for a period of 10^4 seconds (which represents 10^{12} read cycles). There are 17 failures in 1000 MC runs.

4.6.2 One PUF bit represented by multiple RRAM cells

Here we analyze the retention and read disturbance properties in structures where a PUF bit is represented by 2 parallel RRAM cells, 4 parallel RRAM cells and 8 parallel RRAM cells. The concept behind this approach is that if multiple RRAM cells are wired in parallel, the read-out current will be added up. Due to inherent variations, some cells may see their resistance increasing faster than the others. However, even if the current through that cell decreases earlier, the net sum current will still be above the reference current of the sense amplifier. The reference current value for each of the configurations is proportional to the number of cells per bit. For a configuration with 2 parallel RRAM cells per bit, failure point is defined as the time when the net sum current drops to 6 µA. Similarly, for 4 parallel RRAM cells per bit, the threshold current value is 12 µA and for 8 parallel RRAM cells per bit, the threshold current is 24 µA. The simulation conditions such as 125 °C and -0.3 V read voltage that are used for analyzing the single RRAM cell per bit, are applied here as well.

4.6.2.1 Retention property

Table IV shows that the architecture with two RRAM cells in parallel has 0 error out of 1 million MC runs for a duration of 10 years. Further increasing the number of RRAM cells in parallel to 4 and 8 also guarantees 10-year lifetime.

Number of cells per bit	Ref. Current for Failure	Total Number of Failures	Error
1	3 µA	5	0.005%
2	6 µA	0	0%
4	12 µA	0	0%
8	24 µA	0	0%

Table IV : Total number of failures in 1 million MC runs under retention condition (125°C) for a 10-year lifetime.

4.6.2.2 Read disturbance property

Table V shows how the failure rates decrease as the number of RRAM cells per bit increases. While the failure rate is 1.7% for the architecture with 1 RRAM cell per bit, the failure rate decreases to 0.1346% for the architecture with 2 RRAM cells per bit. The failure rate decreases further to 0.0014% for 4 RRAM cells per bit and becomes 0 for 8 RRAM cells per bit. Thus there is no need for an ECC unit if each bit is represented by 8 RRAM cells for a system with 10-year lifetime.

Total Number of cells per bit	Reference for Failure	Total Number of Failures	Error
1	3 µA	17979	1.7979%
2	6 µA	1346	0.1346%
4	12 µA	14	0.0014%
8	24 µA	0	0%

Table V : Total number of failures in 1 million MC runs under read condition (-0.3V and 125°C) for a 10-year lifetime.

4.7 Hardware Overhead

The area overhead of the proposed RRAM PUF and its peripheral circuitry is calculated using the NVSim simulator [61]. NVSim adopts analytical modules for NVM cells for circuit-level assessment and estimates the physical footprint of the array. Synthesis for Hamming (7,4) ECC and 6:64 decoder is done using design compiler and their area are calculated after doing place and route using Cadence Encounter 10.1. All estimations are done in the 65nm node. The area of Hamming (7, 4) ECC came to be 144 um^2 and its latency was 1ns.

We compare the two architectures: the proposed multiple-cell per bit where each PUF bit is represented by 8 parallel RRAM cells versus the one cell per bit architecture that employs Hamming (7, 4) ECC. We choose Hamming (7, 4) ECC even though the error is only 1.7979%, because there could be an error in 1 of the 4 bits that are read out in each round, and those error need to be corrected. With Hamming (7, 4) ECC data array of size 64x4 increases to 64x7. Table VI shows the area estimations for the baseline RRAM array (64x4) along with its peripheral circuitry (as shown in Figure 18(a)), 64x32 RRAM array corresponding to 8 cells/bit architecture, 64x7 RRAM array corresponding to 1 cell/bit architecture along with Hamming (7,4) ECC unit.

Components	Number of Components Required	Total Area (μm^2)
64x4 RRAM Array	1	4.327
6:64 Decoder	1	232.254
Sense Amplifier	4	4.815
64x32 RRAM Array	1	34.956
64x7 RRAM Array	1	7.571
Hamming (7, 4) ECC	1	144.000

Table VI : Area of different modules in the RRAM PUF based architecture

The hardware overhead of the proposed architecture (8 cells per bit) and the conventional architecture (1 cell per bit with ECC) have been shown with respect to the baseline (1 cell per bit) in Table VII . With 8 cells per bit, the RRAM array has smaller overhead because the RRAM cell size is much smaller than that of the peripheral circuits.

	Total Area (μm^2)	Area Overhead (%)
RRAM (1 cell/bit)	241.396	Baseline
RRAM (1 cell/bit) + Hamming (7,4) ECC	392.251	62.49
RRAM (8 cells/bit)	272.025	12.68

Table VII: Hardware overhead of different RRAM PUF implementation

CHAPTER 5

Embedding RRAM PUF in Cryptographic Modules

Embedded PUF (EPUF) is defined as embedding a PUF in the datapath of SHA-256. The advantage of such a scheme is the large CRP space which makes it suitable for authentication and key generation. EPUF satisfies the uniqueness property since even if two devices have a small difference in their PUF characteristics, the corresponding Hamming distance of the output messages will be large. In our proposed EPUF architecture, the PUF has an additional requirement of reproducibility of responses across different voltage and temperature conditions. If the response of the PUF varies by even one bit, the output will have a Hamming distance of almost 50% as it is embedded in a hash function. In such a case, the device cannot be authenticated or an incorrect key will be produced. Thus the PUF in an EPUF has to be implemented using a robust memory array such as the 8 cells/bit RRAM array as described in Chapter 4.

5.1 Architecture of Embedded PUF

As discussed in Section 3.1, the SHA-256 architecture is based on Merkel Damgard construction where 64 rounds of computation are needed to produce the output message digest. In this work, we propose to embed the highly reliable RRAM PUF, described earlier, inside the SHA-256 hardware unit as shown in Figure 21.

In each round *t*, the PUF response can be used to change the message word W_t and/or the round constant K_t. Since the change does not affect the evaluation method for W_t or the properties of K_t, the immunity of SHA-256 against the crypto-analytic attacks is not compromised.

To decide on the number of bits that needs to be flipped, we have considered the impact on the security level of the system as well as the hardware cost. If only one bit of K_t or W_t is changed, the number of brute force attacks that are required to crack this system is 2^{5x64}.This number changes to 2^{8x64} if two bits of K_t or W_t are changed. Since the hardware overhead of changing 2 bits is not very high, we consider changing 2 bits in this work.

Furthermore, we chose to modify W_t over K_t. Modification of K_t or W_t would have resulted in the same security level theoretically, in practice, modifying W_t leads to higher security. This is because in a W_t based system, even if the attacker is able to predict the message digest for one specific input message, he will not be able to successfully predict for another message without resorting to another round of attacks.

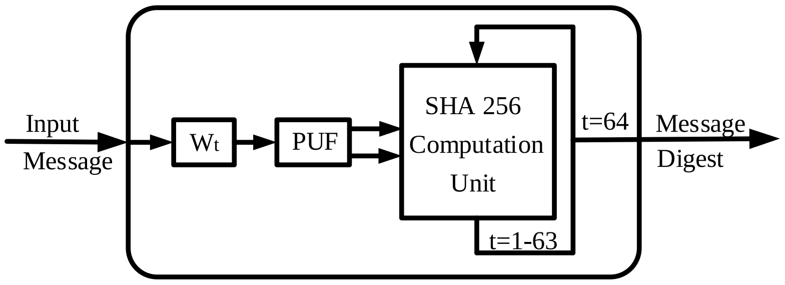


Figure 21: Proposed architecture for EPUF. The value of W_t changes in every round based on the PUF response

Figure 22 describes the proposed method where 2 bits of W_t are inverted. W_t is divided into two equal halves and 1 bit from the lower half and 1 bit from the upper half are chosen to be inverted. This is implemented using two PUF arrays along with a decode circuitry. Since there are 64 rounds, the size of each of the PUF arrays is 4x64. In every round, one row of the PUF array is selected based on the round number. A 6:64 decoder is used to select a row in both the PUF arrays. Each of the PUF arrays gives a 4 bit output which is then fed to a corresponding 4:16 decoder. The 2x16=32 bits are then XOR with 32 bit W_t. The synthesis results for SHA-256 along with the PUF circuitry in the datapath is explained in section 5.2 and the security analysis of the system is presented in section 5.3.

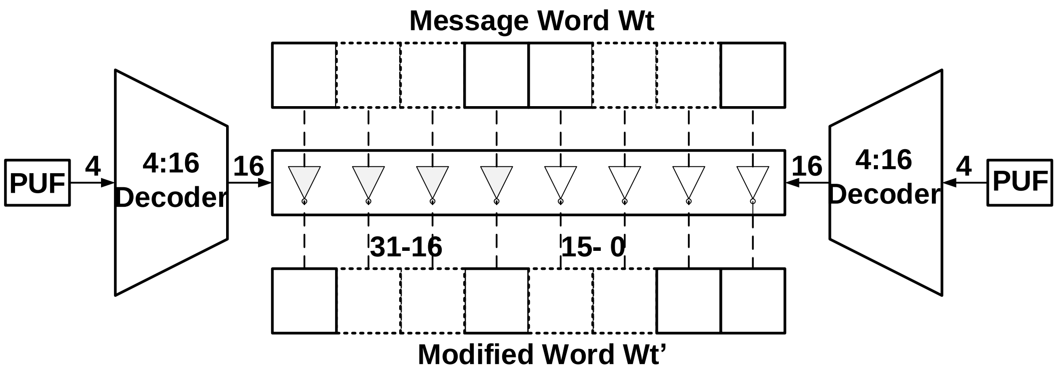


Figure 22: Implementation details for proposed scheme in which 2 bits of W_t are inverted where the position of inversion is determined by the EPUF response.

5.2 Synthesis result for Embedded PUF

The HDL code for SHA-256 was obtained from [62]. Synthesis was done using Design Compiler and place and route was performed using Cadence Encounter 10.1. TSMC 65nm library was used to generate the final hardware. The output of the synthesized hardware was compared against the results obtained from [63]. The outputs are identical, thereby verifying the correctness of the hardware implementation.

Timing: The clock period for the computation of one round was set at 5ns and the total latency to produce one message digest was found to be 320ns.

Area: Figure 23 gives the image of the synthesized hardware after place and route. The total area of the SHA-256 module is 230.4μm*239.7μm =55226.88μm^2. The 8 cells/bit RRAM PUF has an area of 601.60μm^2. Thus the hardware overhead of the RRAM PUF is only 1.08% of the existing SHA-256 module.

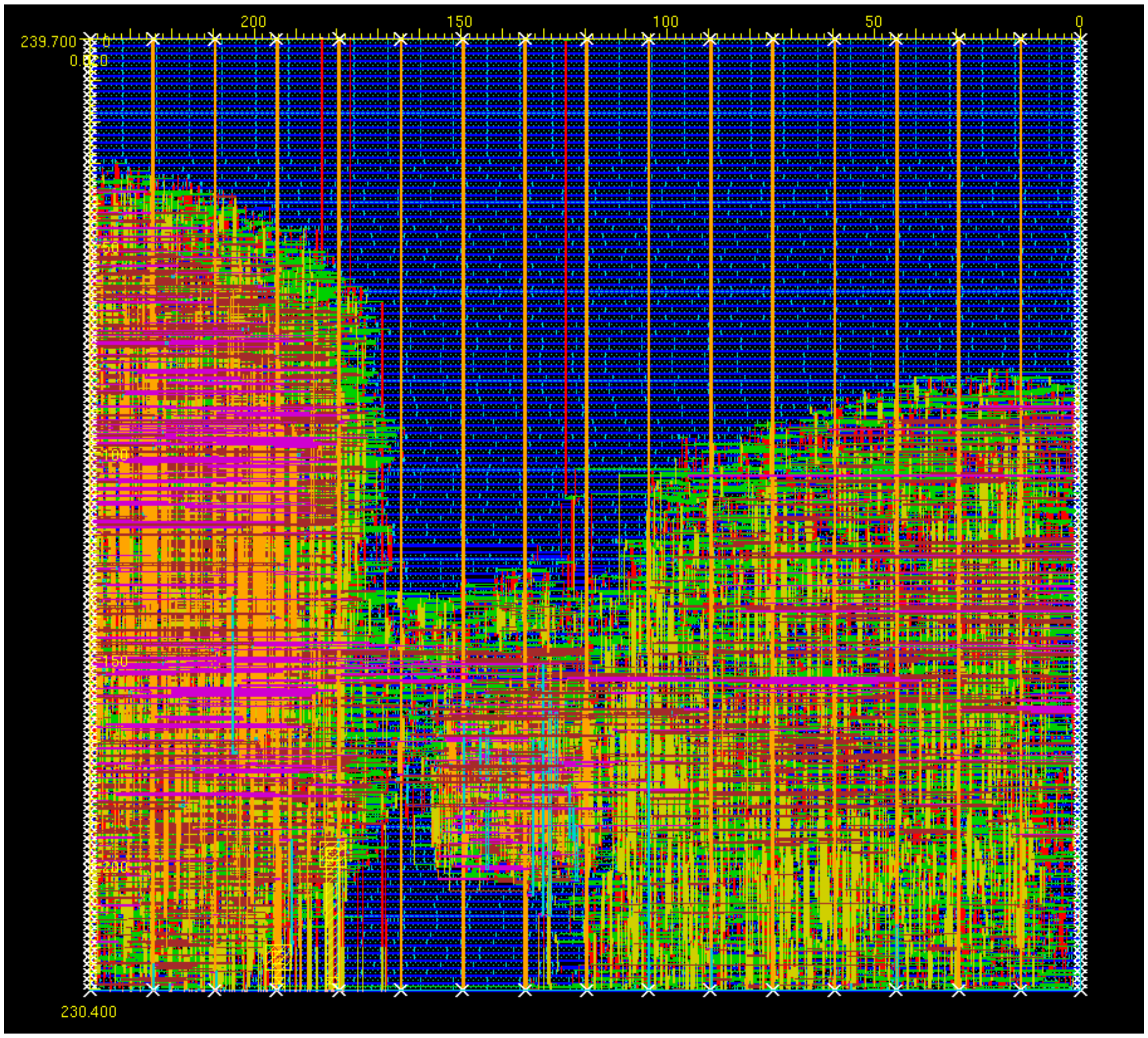


Figure 23: Hardware Implementation of SHA-256

5.3 Security Analysis of EPUF

As the basic properties and architecture of SHA-256 are not changed, the resistance of EPUF against crypto-analytic attack remains the same. We further analyze the immunity of the EPUF architecture towards the following standard non-invasive attacks on PUFs.

1. Replay attack: In a replay attack, the adversary can reuse previously used CRPs. To prevent this attack, the verifier never uses a CRP more than once and so to make this

scheme practical, the PUF needs to have a large CRP space. In our scheme, since the input is fed to the hash function and the output is the message digest, the CRP is very large.

2. Chosen challenge attack: In this attack, the adversary can just change one bit of the challenge at a time and can track the variations in the response to model the PUF. To prevent this, the PUF should have a strict avalanche property, that is, even if one bit of the input message is changed, the probability of flipping each response bit should be 0.5. In the proposed design, changing one bit of the input message results in Hamming distance of 50%. So even if the chosen challenge attack is performed on the system, the effort the adversary has to make to extract the EPUF characteristic does not change.

3. Machine learning attack: These attacks aim to predict the response of PUF for a randomly chosen challenge based on previously recorded CRPs. Strong PUFs, whose CRPs can easily be extracted from the communication channel, have been proven to be vulnerable against this type of attack [12-13]. In the proposed design, the CRPs of the PUF are never exposed to the communication channel; instead they are well embedded inside the hash module and thus the machine learning algorithms will not be able to model the EPUF.

4. Random guessing attack: To analyze the probability of success of such an attack we assume that the attacker has the input message to the SHA as well as the output message digest. Also the attacker knows that the method is based on changing the bits in Wt. Since one bit out of lower 16 bits of W_t and one bit from the upper 16 bits are inverted, there are $\binom{16}{1}$ x$\binom{16}{1}$ different combinations for each round, and so the total number of attacks that needs to be performed in the worst case is 2^{8x64}.

While doing the initial programming of the RRAM, we have not split the cells into low resistance and high resistance states with 50% probability unlike the approach in [10]. To analyze the effect of skewed distribution, we consider an extreme case in which only 25% of the bits are 1 and the rest are 0. Thus one out of every 4 bits (instead of 16 bits) in a 64×4 array will be 1. As a result, the decoder will choose 1 out of 4 bits (instead of 1 out of 16 bits) to perform the inversion. Thus, in each round, the number of different combinations is $\binom{4}{1}\text{x}\binom{4}{1}$, and in 64 rounds, the total number of different combinations will be 2^{64x4} which is still a very large number.

5.4 EPUF used for Authentication

In the proposed authentication scheme if the verifier wants to verify an IC with EPUF he needs to have access to the exact characteristics of the PUF that is embedded inside SHA. The verifier sends a message to the device to be authenticated. The device produces the hashed response of the message. As PUF characteristics vary in every device, each device has its unique response for a given message. The verifier then calculates the hashed response based on the PUF characteristics stored in the database. If the calculated response matches the received response the device is successfully authenticated. This method is described in Figure 24.

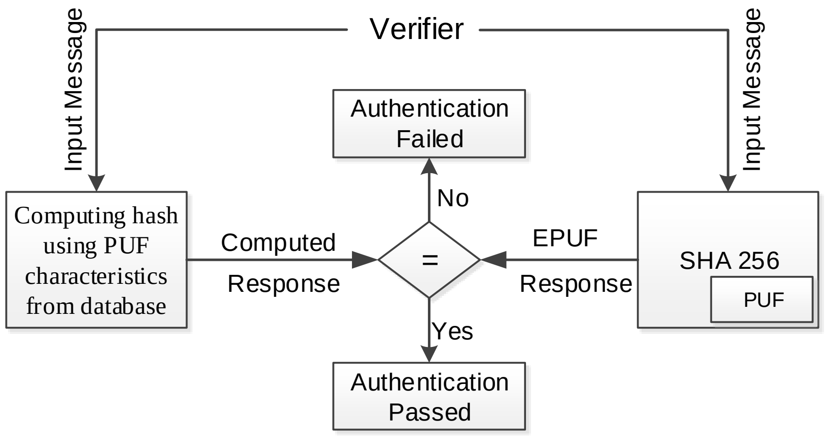


Figure 24: EPUF used for authentication of IC

5.5 EPUF used for key generation in AES-128

In this work we use EPUF as a key generation unit for AES-128.

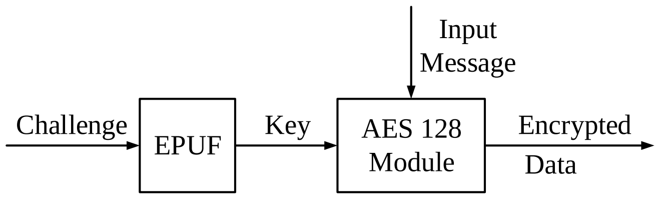


Figure 25: EPUF used for key generation is AES-128 module

Figure 25 shows the use of EPUF as a key generation unit for AES. In this architecture, challenge is given to EPUF for key generation. Since EPUF is based on SHA-256, it produces a 256 bit key output. As AES-128 only requires a 128 bit key, the remaining bits are ignored. The 128 bits are used as a cipher key for encryption, and the key generation algorithm discussed in section 3.5 can be used to produce expanded keys for all the rounds in AES-128.

5.6 Synthesis result for AES-128

The HDL code for AES-128 was obtained from [64]. The AES-128 design was synthesized using Design Compiler, and place and route was performed using Cadence Encounter 10.1. TSMC 65nm library was used to generate the hardware implementation. To verify the correctness of the hardware implementation, the output of the synthesized hardware was compared against the output from [65]. The two outputs were found to be the same, thereby confirming the hardware implementation was functionally correct.

Timing: The clock period for the computation was set at 1ns. The total latency to produce 128 bit encrypted data was 20ns.

Area: Figure 26 gives the image of the hardware implementation after place and route in Cadence Encounter. The total area of the design is $1179\mu m$*$1179.2\mu m$ =$1390276.8\mu m^2$. Thus the area of the 8 cells/bit RRAM PUF is only 0.04% compared to the AES-128 module.

5.7 Security Analysis of AES-128 having EPUF for Key Generation

As the EPUF module is used for key generation, the security analysis presented in section 5.3 for EPUF remains the same. Also the architecture of AES-128 is unchanged so its immunity towards cryptoanalytic attack is sustained. The advantage of generating keys from EPUF is that even if the adversary know the input challenge to EPUF, he will not be able to predict the key for the given challenge or a similar challenge with a minimum Hamming distance.

Several invasive attacks have been shown on keys stored in a non-volatile memory. EPUF with an embedded RRAM PUF is immune towards invasive attacks due to its nanoscale dimension. The scalability of RRAM has been proved to be less than 10 nm, thus it is extremely difficult to directly probe such small cell using invasive techniques. Even if a single RRAM cell is probed, the atomistic changes in the defect density in the oxide are invisible even under high-resolution transmission electron microscopy. The only weak point is that RRAM's characteristics may leak out if the sense amplifier's outputs are invasively probed. Also compared to an SRAM PUF, RRAM PUF is more resistant towards semi invasive attacks. This is because RRAM is not expected to emit photons under photon emission analysis unlike the hot carriers in transistors of an SRAM cell [66].

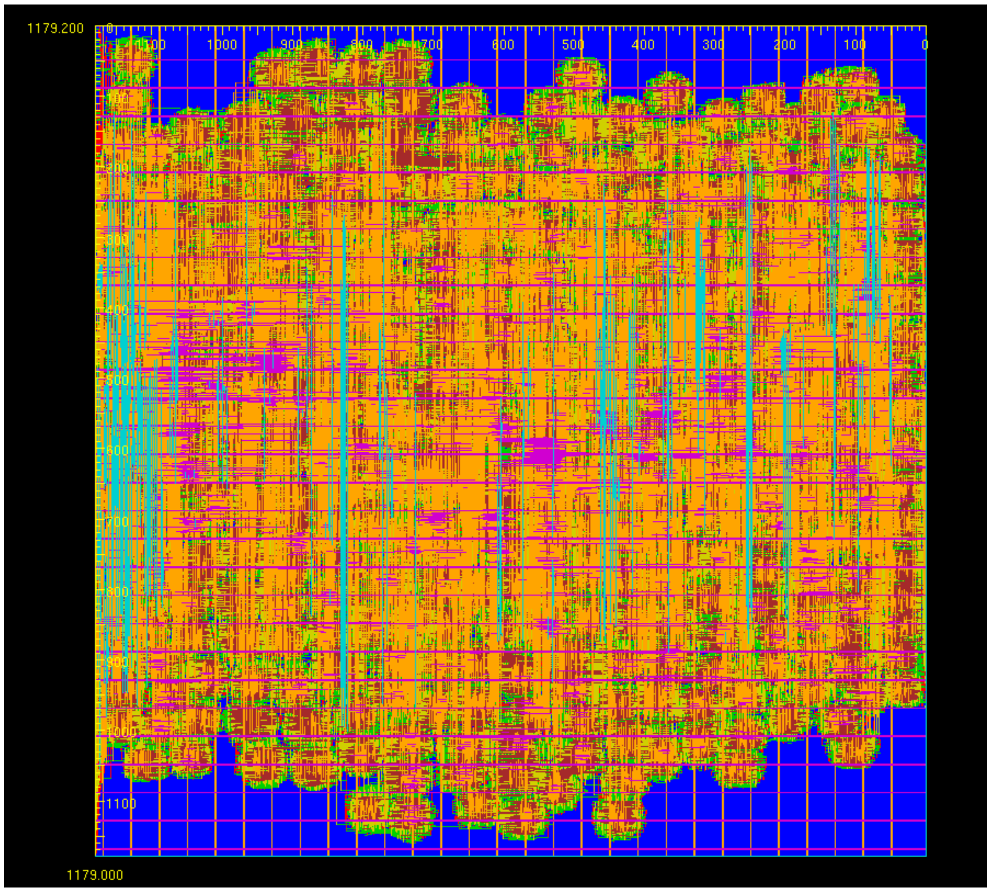


Figure 26: Hardware Implementation of AES-128

CHAPTER 6 Conclusion and Future Work

In this thesis we have described the use of PUF to enhance system security. We first present the design of a highly reliable RRAM PUF based on a multiple cell per bit architecture. Such an architecture can be embedded in cryptographic modules and do not require ECC to generate a stable output. We studied RRAM PUF architectures with 2 cells/bit, 4 cells/bit and 8 cells/bit. Using a device model that was calibrated with IMEC HfOx RRAM experimental data [58-60], we showed that the architecture with 8 cells/bit has an endurance of 10 years under military conditions of 125°C and can sustain read disturbance for 10^{12}cycles.

Next we presented EPUF architecture in which the highly reliable RRAM PUF is embedded in the SHA-256 datapath. The EPUF was also used in key generation module for AES-128. SHA-256 and AES-128 modules were synthesized using the TSMC 65nm library. It was found that the area overhead of 8 cell/bit RRAM PUF is only 1.08% and 0.04%, of SHA-256 and AES-128, respectively. Finally, the security analysis of the complete system was presented and it was shown that EPUF is resistant towards standard attacks on PUFs and does not affect the security level of cryptographic modules against cryptoanalytic attacks.

6.1 Future Work

To enhance the security of the system, the next step can be device specific encryption. PUF can be embedded in AES datapath such that it does not affect the security criteria for encryption as defined by NIST [30]. One way is to change the rotation offset in each round in ShiftRows based on the RRAM PUF response. In the ShiftRows step, row 0 is shifted by C_0 bytes, row 1 is shifted by C_1 bytes, row 2 is shifted by C_2 bytes and row 3 is shifted by C_3 bytes. For diffusion optimality, C_0, C_1, C_2, C_3 have to be different [30]. When the block length is 128 bits, C_0, C_1, C_2 and C_3 are chosen to be 0, 1, 2 and 3 for all the rounds. In the proposed method, different values of C_0, C_1, C_2 and C_3 can be used in different rounds. For instance in round k, C_0, C_1, C_2 and C_3 can be set to 2, 3, 1, 0, while in round k+1, they can be set to some other set of values such as 3, 2, 0, 1. In the decryption module, the InvShiftRows function shifts the rows so that the effect of the shift in ShiftRows is annulled. The effect of the proposed variable offset based method can also be evaluated on different types of attacks. These include differential power analysis (DPA) attacks and fault injection attacks.

Further the proposed multi cell per bit architecture can also be applied to the existing RRAM PUF architectures which have a high degree of uniqueness but reduced reliability with increasing time. In [67], two RRAM PUF arrays are programmed into HRS state to utilize the variability of RRAM PUF. For every challenge, two values are read out from the two arrays and compared. The comparison information is then used to derive the response bit. This method achieves Inter-Hamming distance of 50% with minimal hardware overhead but its Intra-Hamming Distance degrades over time. The proposed multi cell/bit architecture can be used to compensate for the degradation.

REFERENCES

[1] “Insulin pump hack delivers fatal dosage over the air,” The Register, 2011. (Available at http://www.theregister.co.uk/2011/10/27/fatal_insulin_pump_attack/)

[2] “2014 celebrity photo leaks,” Wikipedia, (Available at http://en.wikipedia.org/wiki/2014_celebrity_photo_leaks)

[3] “Sony Pictures Entertainment hack,” Wikipedia, (Available at http://en.wikipedia.org/wiki/Sony_Pictures_Entertainment_hack)

[4] “Russian Hackers Amass Over a Billion Internet Passwords,” The New York Times, 2014 (Available at http://www.nytimes.com/2014/08/06/technology/russian-gang-said-to-amass-more-than-a-billion-stolen-internet-credentials.html)

[5] “Bring your own device program best practices (BYOD),” Gartner Webinar, 2013 (Available at http://www.gartner.com/newsroom/id/2466615)

[6] R. Maes. “Physically unclonable functions: Constructions, properties and applications”. Dissertation, University of KU Leuven, 2012

[7] B. Gassend, D. Clarke, M. Van Dijk, and S. Devadas, “Silicon Physical Random Functions,” ACM Conference on Computer and Communications Security (CCS) pp. 148–160, 2002.

[8] C. Herder, et al, “Physical unclonable functions and applications: a tutorial,” *Proc. IEEE*, vol. 102, no. 8, pp. 1126-1141, 2014

[9] J. Guajardo, S. S. Kumar, G. J. Schrijen, and P. Tuyls, “FPGA intrinsic PUFs and their use for IP protection,” *Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pp. 63-80, 2007.

[10] B. Gassend, D. Lim, D. Clarke, M. Van Dijk, and S. Devadas, “Identification and Authentication of Integrated Circuits”, Concurrency and Computation: Practice and Experience, vol. 16, issue 11, pp.1077–1098, 2004.

[11] J. W. Lee, D. Lim, B. Gassend, G. E. Suh, M. Van Dijk, and S. Devadas, “A Technique to Build a Secret Key in Integrated Circuits for Identification and Authentication Application”, *Symposium on VLSI Circuits*, pp. 159-176, 2004.

[12] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, and S. Devadas, “Modeling attacks on physical unclonable functions,” *ACM Conference on Computer and Communications Security (CCS)*, pp. 237-249, 2010.

[13] U. Rührmair, J. Solter, F. Sehnke, X. Xu, A. Mahmoud, V. Stoyanova, G. Dror, J. Schmidhuber, W. Burleson, and S. Devadas, “PUF modeling attacks on simulated and

silicon data," *IEEE Transaction on Information Forensic and Security,* vol. 8, no. 11, pp. 1876-1891, 2013.

[14] G. E. Suh, and S. Devadas, "Physical Unclonable Functions for Device Authentication and Secret Key Generation," *ACM Design Automation Conference (DAC)*, pp. 9-14, June 2007.

[15] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Lightweight secure PUFs," *IEEE/ACM International Conference on Computer-Aided Design (ICCAD),* pp. 670-673, 2008.

[16] P. Koeberl, et al, "Memristor PUFs: A New Generation of Memory-based Physically Unclonable Functions," *IEEE Design and Automation Test in Europe (DATE)*, pp. 428-431, 2013.

[17] G. S. Rose, et al, "A write-time based memristive PUF for hardware security applications," *IEEE International Conference on Computer Aided Design (ICCAD),* pp. 830-833, 2013.

[18] U. Rührmair, et al, "Applications of High-Capacity Crossbar Memories in Cryptography," *IEEE Trans. Nanotechnology*, vol. 10, no. 3, pp. 489-498, 2011.

[19] J. Rajendran, et al, "Nano-PPUF: A memristor-based security primitive," *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 84-87, 2012.

[20] W. Che, et al, "A Non-Volatile Memory Based Physically Unclonable Function without Helper Data," *IEEE International Conference on Computer Aided Design (ICCAD),* pp. 148-153, 2014.

[21] A. Chen, "Utilizing the variability of resistive random access memory to implement reconfigurable physical unclonable functions," *IEEE Electron Device Letters,* vol. 36, no. 2, pp. 138-140, 2015.

[22] P. Y. Chen, et al. "Exploiting Resistive Cross-point Array for Compact Design of Physical Unclonable Function," *IEEE Hardware Oriented Security and Trust (HOST),* pp. 26-31, 2015.

[23] B. Gassend, D, Clarke, M. V. Dijk, and S. Devadas, "Controlled physical random functions," *18th Annual IEEE Computer Security Application Conference,* pp. 149-160, 2002.

[24] S. Katzenbeisser et al. "Recyclable PUFs: Logically Reconfigurable PUFs," *Cryptographic Hardware and Embedded Systems, LNCS vol*. 6917, pp. 374-389, Sep. 2011.

[25] A. Van Herrewege, et al. “Reverse fuzzy extractors: Enabling lightweight mutual authentication for PUF-enabled RFIDs,” *Financial Cryptography and Data Security FC, LNCS vol.* 7397, pp.374-389, 2012.

[26] T. Rahman, D. Forte, J. Fahrny, J., and M. Tehranipoor, “ ARO-PUF: An aging-resistant ring oscillator PUF design,” *IEEE Design, Automation & Test in Europe* (DATE), pp. 69-74, 2014.

[27] J. Delvaux, I. Verbauwhede, “Key-recovery attacks on various RO PUF constructions via helper data manipulation,” IEEE *Design, Automation and Test in Europe (DATE)*, pp. 72-77, 2014.

[28] H.-S. P. Wong, et al., “Metal–oxide ReRAM,” *Proc. IEEE*, vol. 100, no. 6, pp. 1951–1970, 2012.

[29] National Institute of Standards and Technology (NIST). Descriptions of SHA-256, SHA-384 and SHA-512, 2001, http://csrc.nist.gov/encryptionishs/sha2S6-3X4-SI2.pdf.

[30] National Institute of Standards and Technology (NIST). Advanced Encryption Standard (AES), 2001. FIPS-197. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[31] R. S. Pappu,“Physical One-Way Functions”, *Ph.D. thesis, Massachusetts Institute of Technology (MIT)*, pp. 24, 29, 58, 66, 74, 76, 78, 208, 2001.

[32] R. S. Pappu, et al., “Physical One-Way Functions”, *Science* vol. 297, no. 2026, 2002.

[33] J. Guajardo, et al., “Anti-Counterfeiting, Key Distribution, and Key Storage in an Ambient World via Physical Unclonable Functions”, *Information Systems Frontiers* vol. 11, issue 1, pp. 19–41, 2009.

[34] D. Merli, D. Schuster, F. Stumpf, G. Sigl, “Side-channel analysis of PUFs and fuzzy extractors,” *Trust and Trustworthy Computing, vol. 6740*, pp. 33-47, 2011.

[35] J. Delvaux, and I. Verbauwhede, “Side channel modeling attacks on 65nm arbiter PUFs exploiting CMOS device noise,” *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pp. 137-142, June 2013.

[36] A. Mahmoud, U. Rührmair, M. Majzoobi, M., and F. Koushanfar, “Combined Modeling and Side Channel Attacks on Strong PUFs”, *IACR Cryptology*, 2013.

[37] S. Sharif Mansouri, and E. Dubrova, “Ring oscillator physical unclonable function with multi level supply voltages,” *IEEE International Conference on Computer Design (ICCD)*, pp.520-521, 2012.

[38] M. Cortez, A. Dargar, S. Hamdioui, and G. J. Schrijen, “ Modeling SRAM start-up behavior for Physical Unclonable Functions”, *IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT),*pp. 1-6, 2012

[39] T. Yi Liu et al., “A 130.7mm2 2-layer 32gb reram memory device in 24nm technology”, *IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC),* pp. 210–211, Feb 2013.

[40] L. Zhang, X. Fong, C.-H. Chang, Z. H. Kong, and K. Roy, “Feasibility Study of Emerging Non-Volatile Memory Based Physical Unclonable Functions,” *IEEE International Memory Workshop (IMW),* pp. 135-138, 2014.

[41] S. Yu, X. Guan, and H.-S. P. Wong, “Conduction mechanism of TiN/HfOx/Pt resistive switching memory: A trap-assisted-tunneling model,” Appl. Phys. Lett., vol. 99, no. 063507, 2011.

[42] A. J. Menezes, S. A. Vanstone, and P. C. V. Oorschot, “Handbook of Applied Cryptography”, *CRC Press,*1996.

[43] M. Kim, et al, "Efficient hardware architecture of SHA-256 algorithm for trusted mobile computing," *Information Security and Cryptology, Springer Berlin Heidelberg,* pp. 240-252, 2009.

[44] H. Handschub and H. Gilbert, “Evaluation Report Security Level of Cryptography – SHA-256”, Technical Report, Issy-les-Moulineaux, 2002.

[45] S. R. Patel,“Improving resilience against differential power analysis with low area and power overhead using threshold logic”, *M.S. Thesis, Arizona State University,* 2010.

[46] H. Dobbertin, L. Knudsen, and M. Robshaw, “The Cryptanalysis of the AES - A Brief Survey”, Springer Berlin / Heidelberg, 2005.

[47] J. Deamen and V. Rijmen, “The Design of Rijndael”, Springer, 2002.

[48] T. Popp, “An introduction to implementation attacks and countermeasures” *7th IEEE/ACM International Conference on Formal Methods and Models for Codesign,* pp. 108-115, 2009.

[49] Y. Zhou, and D. Feng, “Side-Channel Attacks: Ten Years After Its Publication and the Impacts on Cryptographic Module Security Testing”, *IACR Cryptology ePrint Archive,* 2005.

[50] P. Kocher, J. Jaffe, and B. Jun, “Differential power analysis”, *Advances in Cryptology, CRYPTO,* pp. 388-397, 1999.

[51] K. Smith Jr., "Methodologies for power analysis attacks on hardware implementations of AES," *Master's thesis, Rochester Institute of Technology*, 2009.

[52] S. B. Ors, F. Gurkaynak, M. E. Oswald, and B. Preneel, "Power Analysis Attack on an "ASIC AES Implementation," *International Conference on Information Technology,* pp. 546 – 552, 2004.

[53] F. Regazzoni, T. Eisenbarth, J. Grossschadl, J., and L. Breveglieri, "Power attacks resistance of cryptographic s-boxes with added error detection circuits." *22nd IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems*, pp. 508-516, 2007.

[54] P. Koeberl, et al, "Memristor PUFs: A New Generation of Memory-based Physically Unclonable Functions", *IEEE Design, Automation & Test in Europe* (DATE), pp. 428-431, 2013.

[55] G. S. Rose, et al, "Foundations of Memristor Based PUF Architectures," *IEEE/ACM International Symposium on Nanoscale Architecture (NANOARCH),* pp. 52-57, 2013.

[56] X. Guan, et al, "A SPICE compact model of metal oxide resistive switching memory with variations," *IEEE Electron Device Leter.*, vol. 33, no. 10, pp. 1405-1407,2012.

[57] ASU RRAM model, http://faculty.engineering.asu.edu/shimengyu/modeldownloads

[58] Y. Y. Chen, et al, "Balancing SET/RESET pulse for endurance in 1T1R bipolar RRAM," *IEEE Trans. Electron Devices,* vol. 59, no. 12, pp. 3243-3249, 2012.

[59] Y. Y. Chen, et al, "Improvement of data retention in HfO2/Hf 1T1R RRAM cell under low operating current," *IEEE International Electron Devices Meeting (IEDM),* pp. 252-255, 2013.

[60] A. Fantini, et al, "Intrinsic switching variability in HfO2 RRAM," *IEEE International Memory Workshop (IMW),* pp. 30-33, 2013.

[61] X. Dong, et al, "NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Non-Volatile Memory," *IEEE Transaction on Computer Aided Design (CAD),* vol.31, no.7, pp. 994 - 1007, 2012.

[62] J. Strombergson, https://github.com/secworks/sha256.

[63] http://hash.online-convert.com/sha256-generator.

[64] H. Hsing , http://opencores.org/project,tiny_aes.

[65] http://aes.online-domain-tools.com/.

[66] C. Helfmeier, et al, "Cloning physically unclonable functions," *IEEE Hardware Oriented Security and Trust (HOST)*, pp. 1-6, 2013.

[67] A. Chen, "Utilizing the Variability of Resistive Random Access Memory to Implement Reconfigurable Physical Unclonable Functions", IEEE *Electron Device Letters,* pp. 138-140, 2015.