FPGA Based Physical Unclonable Function (PUF)-A Hardware Security Macro for securing Smart Meter Systems in IoT Environment

Ph.D Thesis

Submitted in partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY

by
DINESH REDDY J
Reg.No.EE16F03

Under the Guidance of
Prof. Dr.K.Panduranga Vittal

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING,
NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA,
SURATHKAL, MANGALORE -575025

January,2024

DECLARATION

I hereby ***declare*** that the Ph.D Thesis entitled **FPGA Based Physical Unclonable Function (PUF)-A Hardware Security Macro for securing Smart Meter Systems in IoT Environment** ***National Institute of Technology Karnataka, Surathkal*** in partial fulfilment of the requirement for the award of the Degree of ***Doctor of Philosophy*** in **Department of Electrical and Electronics Engineering** is a ***bonafide report of the research work carried out by me***. The material contained in this Thesis has not been submitted to any University or Institution for the award of any degree.

.....

Dinesh Reddy J, EE16F03

Department of Electrical and Electronics Engineering.

Place: NITK-Surathkal.

Date: 12/01/2024

ACKNOWLEDGEMENT

"I would like to take this opportunity to express my gratitude to those who helped me with various aspects of research and the writing of this thesis.

I would like to express my sincere gratitude to Dr. K. Panduranga Vittal , Professor, Department of Electrical and Eletronics Engineering, NITK Surathkal , for his guidance , encouragement and for having been my Ph.D supervisor. He has been a constant source of inspiration throughout this journey. I feel proud to have worked under his guidance.

I want to thank my research progress assessment committee (RPAC) members for the input and sense of direction they provided: Dr Shubhanga K N, Professor, Dept of EEE, NITK and Dr.Aparna P, Assistant Professor, Dept of ECE, NITK.

I thank National Institute of Technology Karnatake (NITK), Surathkal for giving me an opportunity for doing my research and Ministry of Human Resource Development (MHRD) , Government of India for awarding research scholarship.

Of course, last but not least, I thank my parents who have supported and encouraged me in every way, and who raised me with skills to success and attitude to handle both success and failure the same way. It was they who instilled in me the love of learning which has come to fruition in this thesis.

Finally , I thank my better half who gave me all the support from family front with a go ahead flag always. "

CERTIFICATE

This is to certify that the Ph.D thesis entitled **FPGA Based Physical Unclonable Function (PUF)-A Hardware Security Macro for securing Smart Meter Systems in IoT Environment** submitted by Dinesh Reddy J, (Register Number: EE16F03) as the record of the research work carried out by him, is accepted as the *Ph.D Thesis submission* in partial fulfillment of the requirements for the award of degree of Doctor of Philosophy.

12/01/2024

Dr. K.Panduranga Vittal
(Research Guide)

12/01/2024

Dr Dattatraya N Gaonkar
(Chairman-DRPC)

PROFESSOR AND HEAD
DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA
SRINIVASNAGAR, SURATHKAL, MANGALORE - 575 025, INDIA

Abstract

Internet of Things (IoT) has transformed the engineering approach towards solving a problem. The key components of an IoT system are smart devices, sensors, gateway, cloud and user interface. The entire process of layered communication among these components of IoT is transparent and hence the intercommunication presents a potential vulnerability to unauthorized access of such an environment. Consequently, it is imperative not only to ensure secure communication channels for data transactions, but also to protect the physical layer, safeguarding the devices against intrusions. Hardware security, particularly device identification, conventionally relies on cryptographic hardware, such as the Secure Hash Algorithm (SHA) or public/private key encryption algorithms.

A formalization of this concept has evolved from physical one-way functions to Physically Unclonable Functions (PUFs). PUFs produce a response for a given challenge by performing a functional operation. The function is built based on physical manufacturing variations of the device to generate a unique secret identification code and is measured by its challenge response pair (CRP). Although Application-Specific Integrated Circuit (ASIC) based PUFs exist with controlled process variations, the present research proposes the implementation of PUFs on field programmable gate arrays (FPGAs) because of their reconfigurable architecture, which facilitates the creation of a dynamic hardware for an optimal balance between time to market and product quality. Configurable Ring Oscillator (CRO) based PUF circuits have proven effective in formulating strong PUF designs by incorporating non-linearity within the model and utilizing configurable inputs. It is optimized with space and resources available on FPGA devices. The present thesis constitutes the following key elements:

- Designing the PUF.
- Analyzing the PUF for steady responses.
- Proposing a unique methodology for combating the environmental

noise factors, such as temperature and timing variations (static and dynamic), affecting the PUF response.

- Validating the proposed methodology on:
  - FPGAs with only programmable logic on the FPGA fabric (Spartan-3 series devices), necessitating an external processor for processing the response in generating the signature.
  - FPGAs possessing both Programming logic (PL) and Processor system (PS) within the FPGA fabric (Artix 7 series devices), allowing the use of PS and PL based architectures to process the PUF response through the PS on the chip.
- Proposing a Multiplexer based PUF with an architectural enhancement to the existing PUF to make it stronger. The aforementioned methodology for extracting the responses is implemented on the proposed structure. The strength of new proposed PUF is analyzed with its performance metrics and efficiency with respect to hardware.
- Evaluating the design efficiency of the proposed PUF for resisting the attacks by modeling an attack with machine learning based logistic regression algorithm. Results showed a significant resistance to attacks in comparison with conventional models of FPGA based PUFs.

The design is validated on the aforementioned hardware and applied to a connected system model of PUF based authentication in an IoT environment, specifically for a smart meter system.

Keywords: Physically Unclonable Functions, FPGA, ASIC, Hard Macros, Machine learning, IoT security.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Security is a paramount feature in the field of electronic devices being operated over the internet. The data transacted in this domain is always vulnerable. Security is an essential feature prioritized by both developers and customers in a connected environment of smart devices. The inevitability of implementing stringent security measures is amplified by the recent emphasis being laid on vulnerability of data.

The inception of the Internet-of-Things (IOT) has facilitated the interconnection, remote operation, and control of electronic devices through the internet networks. This connectivity allows for various applications, including secure access, mobile payment, electronic passports, smart meters, and smart homes. Owing to the nature of these applications processing sensitive user-specific data, unauthorized disclosure could lead to loss of privacy and other unwanted implications. Low-cost pervasive devices, such as Radio Frequency Identification (RFID) devices and wireless sensor nodes, are the foundations for building the next generation of ubiquitous IoT networks Gao et al. (2016a). Such devices typically operate within limited area and energy resources, presenting a significant challenge in providing essential security services, such as device authentication and cryptographic key storage/generation.

In general, cryptography techniques are used to secure and protect the communication between two parties over a network, wherein each party has to store a key privately. The key can be used to encrypt, decrypt, and authenticate the data and it can be used to secure from malicious attacks. In the present modern era, integrated circuits enable electronic devices with several features in different sectors, such as smart-grid,

banking, healthcare, transportation, etc. Smart card applications are widely used in transactions, such as credit card and debit card payments, payment systems used in transportation, Radio Frequency Identification (RFID) tags, and wireless sensor networks. RFID is a passive identification technology that uses radio waves to identify a tagged object. It is employed in a variety of commercial and industrial settings from inventory management in the supply chain to library checkout tracking. It is essential to secure the information during the storage and communication of such confidential data.

In high-end defense equipment manufacturing, stringent measures are implemented to ensure data security through the use of private data. However, when considering low-cost consumer devices, the risk arises that unauthorized individuals gaining access to the private key may lead to potential eavesdropping and impersonation attacks.

1.1 Cryptographic Key Storage Technologies

Cryptography is the study and practice of securing communication from adversarial or third-party interference. Cryptography can ensure the confidentiality, integrity, authenticity, and acknowledgment of the user data. Cryptographic primitives are low-level algorithms which are used to build secure protocols. These include but are not limited to the authentication, digital signatures, one-way functions, encryption, and decryption. Figure 1.1 shows a basic cryptographic algorithm that is an encryption of plaintext XOR operations with a secret key.

The implementation of current security solutions relies on the secret keys stored in the on-chip non-volatile memory (NVM) or battery-backed static random-access memory (SRAM) van der Leest et al. (2014), Delvaux et al. (2015). However, these approaches introduce critical security-related issues. The keys are always available as the secret keys are stored in NVM or battery-backed SRAM. Consequently, they are susceptible to potential extraction or tampering through invasive or semi-invasive attacks through techniques derived from integrated circuit (IC) failure analysis. Resilience against these physical attacks can be improved through a tamper sensing environment, albeit with an increase in the cost of implementation. Moreover, the programming of secret keys often relies on the IC manufacturer or system owner,

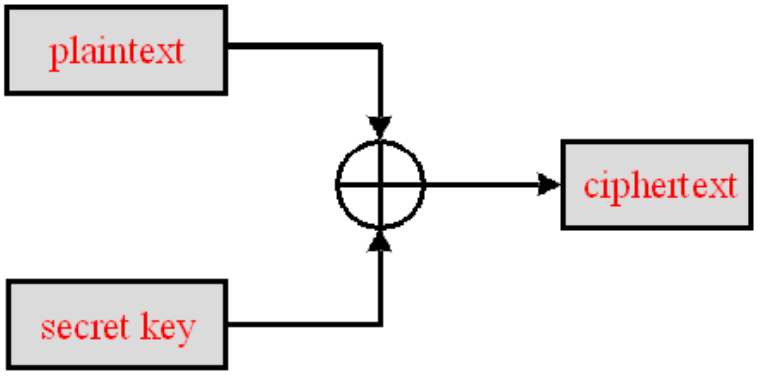


Figure 1.1: Plaintext encryption using a XOR function

potentially leading to a possibility that the secret keys could be compromised by an untrusted third party within the product supply chain.

The relative cost of implementation of the aforementioned key storage solutions is

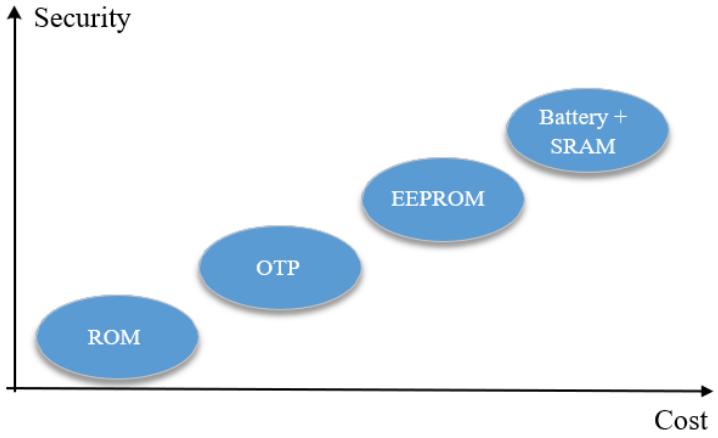


Figure 1.2: Security-cost positioning of permanent key storage technologies van der Leest et al. (2014)

depicted in Figure 1.2. Different technologies, such as Read-Only Memory (ROM), One-Time Programmable (OTP), and Multiple- Time Programmable (MTP), could be deployed for the key storage on NVM. ROM serves as a lightweight approach for storing keys in NVM, suitable for scenarios necessitating large quantities at a low cost. However, this approach offers low security and limited flexibility because the secret keys are mask defined and are shared across devices Delvaux et al. (2015). The

secret keys can be programmed once through the OTP NVM using fuse or anti-fuse technologies. Reprogramming can be emulated via the partitioning of a large OTP NVM, which further increases the cost. Instead, it is more efficient to use MTP NVM, such as electrically erasable programmable read-only memory (EEPROM), which offer more flexibility as the secret keys can be re-programmed multiple times, albeit at a higher cost. Battery-backed SRAM requires a standard CMOS SRAM, but the battery needed to retain the secret keys in volatile memory is considered expensive and space consuming, particularly for resource constrained applications.

Typically, encryption keys are stored in non-volatile memory, such as EEPROMs, which poses a risk of invasive attacks due to the key being stored in a permanent digital format. Attackers can use a combination of optical and chemical methods for conducting reverse engineering attacks. Tamper-sensitive methods can resolve this issue, but these are expensive. Non-invasive side channel attacks rely on implementation weaknesses related to device power and timing and are potent in extracting stored data. These vulnerabilities lie in the cryptographic algorithm analysis involving theory and mathematics.

Integration of a physical layer for security becomes essential to overcome these types of attacks. In a hardware security model, the intruder needs to break the physical layer of the system to get into an application. This approach introduces the concept of the system's physical identity. Physical Unclonable Functions (PUF) are macros derived from manufacturing parameters that define the functionality of the PUF circuit. However, PUFs have limitations due to environmental factors and evolving technological advancements, hindering their ability to generate a stable response. This work aims to address these challenges by presenting a unique methodology for designing PUFs and stabilizing Challenge Response Pairs (CRPs) to mitigate environmental noise factors, such as temperature and timing variations (static and dynamic). The following section provides a brief account of PUFs and Challenge Response Pairs (CRPs).

1.2 Physical Unclonable Functions

A Physical Unclonable Function (PUF) is an emerging technology that offers a cost effective solution to critical security-related issues. PUF operates by mapping a set

of challenges to a set of responses, relying on intrinsic process variations within a silicon chip's transistors and interconnects. The intrinsic process variations are caused by uncontrollable deviations in the chip manufacturing process, which are unique and random across dies and wafers. Consequently, a PUF can be used to generate a unique and random key or identifier. Furthermore, the complex and random nature of the manufacturing process variations makes a PUF practically and physically impossible to clone Gao et al. (2016b).

PUF technology is both secure and cost efficient since it generates the identifier or key during power-on states and wipes it out in the power-off state. The implementation of PUF involves standard CMOS circuit design techniques without the need for special fabrication processes. Moreover, attempts to physically breach PUFs during the power on state using micro-probing techniques are likely to destroy the unique delay characteristics, rendering the identifier or key inaccessible Lee et al. (2004), thereby establishing PUF as a tamper-resistant solution.

The uniqueness, randomness, unclonability, security, low-cost, and tamper resistance provided by PUFs make them ideal for robust hardware-based intrinsic security devices. Leveraging these advantages, PUFs have been proposed for lightweight device identification, authentication, and cryptographic key generation. As a testament to their potential, PUFs are transforming from research to commercial products and are recognized for their substantial industrial value, first commercial PUF was an SRAM PUF in 2007, with the disadvantages of sneaking into the reponse bits in NVM it has evoved as a butterfly PUF ID (2016), NV (2013).

Figure 1.3 shows the security approach towards ICs, which are being recognized with a unique signature in an analogy to a biometric information derived from a fingerprint. The necessity for generating cryptographic keys within these devices stems from the need to safeguard information. These confidential keys are generated using True Random Number Generators (TRNG) and are stored in volatile or non-volatile memories. Non-volatile memories are capable of retaining data without power but they are prone to passive attacks and would be obvious targets to invasive attacks. In contrast, Field Programmable Gate Arrays (FPGAs) employ volatile memories to store the confidential key thereby enabling the erasure of memory contents upon detection

of an attack. This implies that the key would be transmitted by the communication channel after the FPGA configuration, a process vulnerable to corruption and interception of information in communication channels. Consequently, the confidentiality and authenticity of designs would be compromised. A potential solution involves the backing-up of the embedded volatile memory block with a battery. However, it is evidenced that battery-backed RAM content could be accessed after a prolonged period of storage Gao et al. (2016b), even without power. Thus, the need of generating secret keys inside the IC is obvious, resulting in an increasing interest in PUFs.

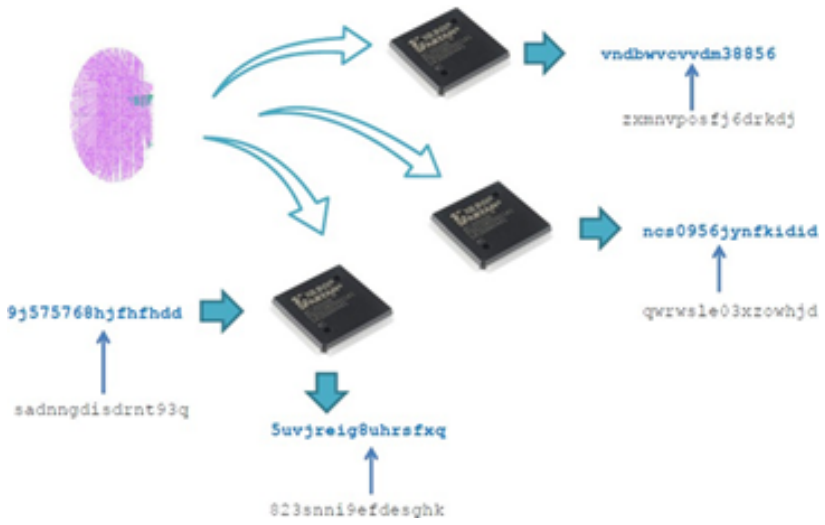


Figure 1.3: Digital Fingerprint of an IC is achieved with PUF Circuits Chongyan Gu (QUB)

PUF can generate unique and secure keys/IDs featuring the following Pappu et al. (2002):

1. Memory less key storage
2. Easy to evaluate
3. Inherently tamper resistant
4. Hard to predict

All of the aforementioned factors make PUFs a versatile solution for security, as PUFs provide cost effective security without additional fabrication cost and inherent

manufacturing variations. PUFs are the novel security primitives used in various applications of hardware devices. Introduction of physical random functions (PRFs) has gradually evolved as PUFs. The evolution of PUF from PRF is explained and some of its properties are listed below for reference (Gassend et al. (2002).

1. The function is based on the physical device characteristics and it generates responses by associating with challenges.
2. The evaluation of challenge response pairs happens within a short period.
3. It is difficult to model the function and re-build the design with the available resources.
4. It is not possible to generate/produce similar devices with identical physical properties, thereby rendering it to be "physically unclonable" or resistant to manufacture.

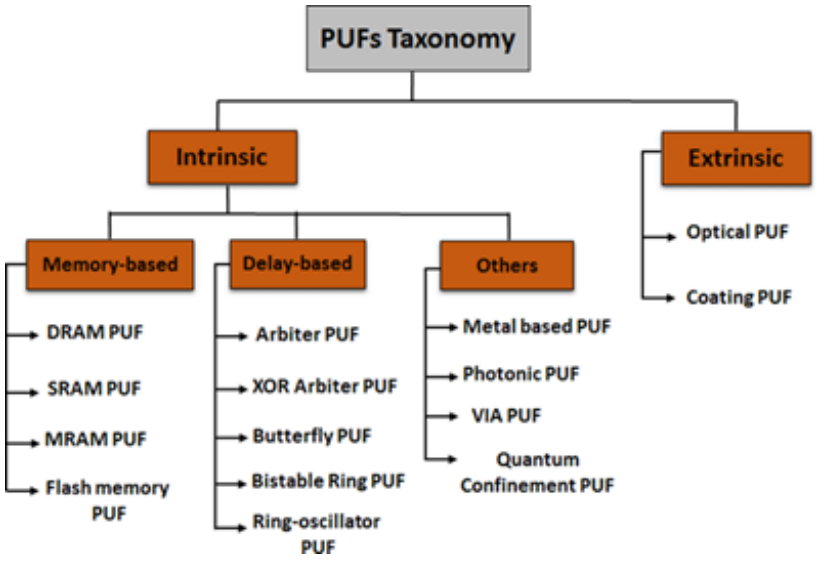


Figure 1.4: Different types of Physical Unclonable Functions

Herder et al. (2014) reported that PUFs serve two primary purposes: low-cost authentication and secure key creation. The categorization of PUFs into "strong" and "weak" types has given rise to these two distinct applications over the past decade. Strong PUFs are typically employed for authentication purposes, while weak PUFs are often utilized for key storage. PUFs essentially operate as black-box challenge–response systems, wherein an input challenge (c) generates a specific response

(r), illustrating the PUF's input/output relationship. This system capitalizes on the internal parameters of the function, which encapsulate the manufacturing variability necessary to construct a unique challenge–response pair (CRP), hence the applicability of the Blackbox model.

As explained in the introduction, the variability of a circuit's internal gate delay significantly contributes to the parameters determining PUF security. The challenge lies in accurately measuring or estimating these parameters, alongside the difficulty of replicating two chips with identical properties. The distinguishing factor between weak and strong PUFs is the domain of the function, signifying the range of unique challenges the PUF can handle. A weak PUF can only withstand a limited number of challenges, sometimes only a single challenge, while a strong PUF can handle a considerable number of challenges, rendering it practically infeasible to determine all challenge–response pairs (CRPs) within a specific timeframe.

1.3 Classification of PUFs

1.3.1 Definition of a PUF

A PUF is defined as a function that maps challenges to responses, the embodiment of which lies within the physical material of the device Gassend et al. (2002). Unlike a deterministic mathematical function that consistently generates the same output for a given input, a PUF, embedded in a physical device, operates non-deterministically and exhibits variations in different instances Maiti (2012). Based on this notion, the present study focused on silicon PUFs, which exploit the intrinsic and random variations in CMOS devices due to the manufacturing process, as described briefly in Section 1.4. Nevertheless, a PUF can be constructed using a non-silicon material, as described in Section 1.5. Specifically for silicon PUFs, the intricate statistical deviations within devices and interconnections allow the mapping of challenges to responses, a mapping that changes stochastically from one instance to another.

Depending on the types of PUFs, the set of CRPs for a PUF can be defined as (Ci,Ri), i = 1, N . Generally, a challenge C can be described as a k-bit input. Challenges are used to control the behavior of a PUF and corresponding responses are

generated based on the challenges applied. As shown in Figure 1.5, when a challenge was applied to two different PUFs (PUF A and PUF B), the respective responses were produced where Response A /= Response B.

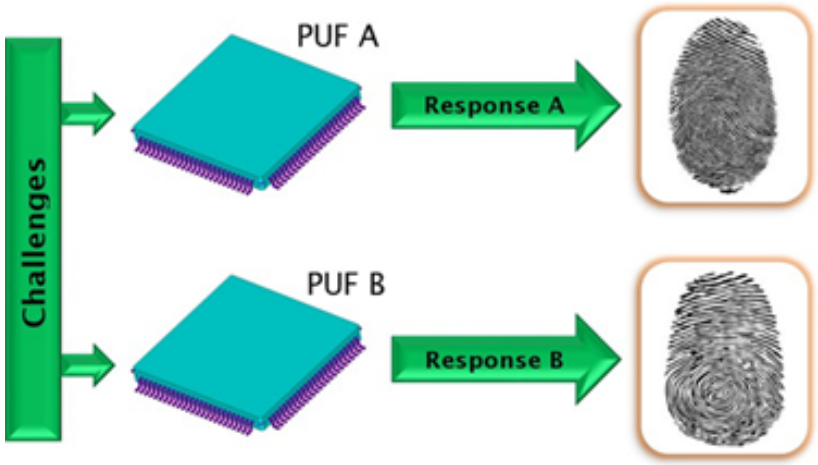


Figure 1.5: Basic functionality of PUF

Each PUF generates a distinct response, akin to an individualized electronic fingerprint, offering unparalleled identification. PUFs negate the necessity for storing a secret key in any memory device, unlike the conventional method of IC security, as discussed in Section 1.1. Instead, a secret key is generated on demand, triggered by the application of a challenge. Thus, PUFs provide unclonable, random, and secure features, rendering them highly promising as a prospective replacement to existing security solutions.

1.4 Variability in Integrated Circuits

Variation in the manufacturing process is a fundamental limitation on controlling the physical features and interconnections of devices during their fabrication Verma et al. (2009).As shown in Figure 1.6, there are two components in process variations,$T_{Nominal}$ and T_{Random}. These process variations can be divided into two categories Siddiqua et al. (2011). The first category is the inter-die variations, where identical devices on different dies might exhibit different characteristics. The second category is the intra-die variations where similar transistors within a single die might display differing

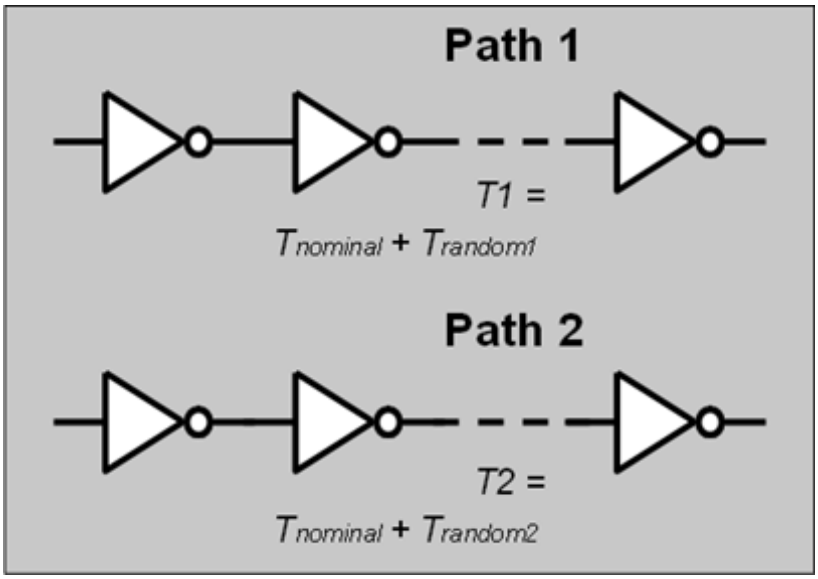


Figure 1.6: Two delay paths with similar nominal delays but different random components of delays

characteristics. The aggressive scaling of Complementary Metal-Oxide-Semiconductor (CMOS) technology has resulted in a drastic increase in process variations, such as oxide thickness and random dopant fluctuations (RDF), which cause a direct impact on the electrical behavior of Metal-Oxide-Semiconductor Field-Effect Transistors (MOSFETs). One of the fundamental challenges for CMOS device performance is RDF, which is caused by the randomness in the amount and position of dopants during the implantation, resulting in fluctuations in the total number of dopants in the transistor channel Ye et al. (2010).

The threshold voltage (V_{th}) of MOSFETs is significantly determined by the dopant density present in the transistor channel. The volume of the channel decreases with the reduction in devices, consequently reducing the overall number of dopants within the channel. As a result, the relative effect of a single change in dopant number increases and the variations in the V_{th} becomes significant Ye et al. (2010). The effect of RDF on device scaling, observed from the 250-nm to 32-nm technology node, is depicted in Figure 1.7. Although process variation is an unwanted element in for CMOS circuitry, it is a desired effect in PUFs.

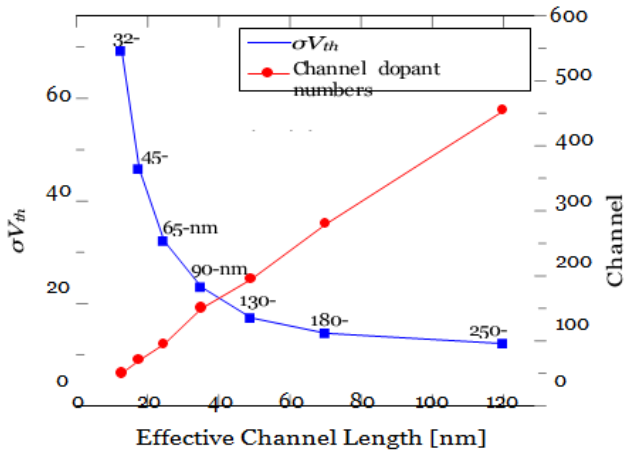


Figure 1.7: Scaling trend of V_{th} variations due to RDF Ye et al. (2010)

1.5 Types of PUFs

1.5.1 Non-Silicon and Silicon PUFs

The idea of a PUF emerged with the introduction of physical one-way functions (POWFs) proposed by Pappu et al., based on an optical operating principle Pappu et al. (2002).

Optical PUFs:

Construction: Optical PUFs are constructed using materials or structures with specific optical properties. These materials may include microscopic patterns,nano-particles, or thin films designed to exhibit unique optical characteristics.

Working Principle:

The working principle relies on the generation of challenge-response pairs based on the interaction of light with the optical material. The challenge involves illuminating the material, and the response is derived from the resulting optical patterns, such as light scattering or absorption. The inherent randomness in optical phenomena provides a source of unpredictability for security applications.The speckle patterns resulting from the application of laser lights at varying angles, distances, and wavelengths on a transparent optical medium, which contained scattering particles, were found to be unique and unpredictable.

Memristive PUFs:

Construction: Memristive PUFs are constructed using memristors, which are two-terminal non-volatile memory devices. These devices have the unique property of exhibiting variable resistance based on the history of applied voltage. Arrays of interconnected memristors form the basis of a Memristive PUF.

Working Principle: The working principle involves exploiting the variability in the resistance states of memristors. Challenge-response pairs are generated by applying specific voltage pulses to the memristor array. The resulting resistance states serve as unique responses, and the history-dependent nature of memristors adds an additional layer of unpredictability.

RFID-based PUFs:

Construction: RFID-based PUFs involve the use of RFID tags and antennas with specific configurations. The construction includes designing RFID tags with distinct physical characteristics that influence radio-frequency communication.

Working Principle: The working principle revolves around creating challenge-response pairs based on the unique characteristics of radio-frequency communication. Variations in signal reflections, phase shifts, or attenuation during RFID communication contribute to the generation of responses. The specific antenna design and tag configuration play a crucial role in ensuring uniqueness.

Biological PUFs:

Construction: Biological PUFs incorporate biological materials such as DNA or proteins. The construction involves leveraging the inherent variability in genetic sequences or protein structures.

Working Principle: Biological PUFs exploit the uniqueness of biological structures. Challenge-response pairs are generated based on the specific characteristics of DNA sequences or protein folding patterns. The inherent variability in biological materials provides a rich source for creating unpredictable responses.

Nanomechanical PUFs

Construction: Nano-mechanical PUFs use nano-scale mechanical devices like nanowires or cantilevers. The construction involves designing these devices to exhibit variations in mechanical properties.

Working Principle: The working principle involves generating challenge-response pairs by measuring the mechanical responses of nano-scale structures to specific stimuli. Variations in length, stiffness, or resonance frequencies contribute to the uniqueness of responses. The nano-scale nature adds an additional layer of complexity and security.

Magnetic PUFs

Construction: Magnetic PUFs incorporate nanoscale magnetic elements, such as nanoparticles or magnetic domains. The construction involves designing materials with specific magnetic properties.

Working Principle: Magnetic PUFs exploit variations in the magnetic properties of these elements. Challenge-response pairs are generated based on the unique magnetic states or behaviors observed under external magnetic fields. The response is influenced by the specific configuration of the magnetic elements, adding a layer of unpredictability.

Chemical PUFs:

Construction: Chemical PUFs involve materials with specific chemical reactions or compositions. The construction includes designing materials that exhibit variations in chemical behavior based on external stimuli.

Working Principle: The working principle revolves around generating challenge-response pairs based on the unique characteristics of chemical reactions. Variations in reaction pathways or responses to specific stimuli contribute to the uniqueness of the responses. The chemical nature of the PUF adds a layer of complexity and diversity.

Quantum PUFs:

Construction: Quantum PUFs leverage quantum bits (qubits) and quantum mechanical principles. The construction involves creating systems that exploit quantum effects such as superposition and entanglement.

Working Principle: The working principle exploits quantum phenomena to generate challenge-response pairs with high levels of unpredictability. Manipulation of qubits creates unique quantum states that serve as responses. Quantum PUFs benefit from the fundamental principles of quantum mechanics, providing a robust foundation for secure applications.

Photonic PUFs:

Construction: Utilizes properties of photons such as phase, polarization, or time-of-flight. Construction involves photonic components like lasers, beam splitters, and detectors.

Working Principle: Relies on the unique interactions of photons with photonic elements. Challenge-response pairs are generated based on specific properties of light, ensuring unpredictability.

Acoustic PUFs: Construction: Uses variations in acoustic wave propagation, reflections, or resonances. Construction involves acoustic transducers and materials with specific acoustic properties.

Working Principle: Involves the generation of challenge-response pairs based on unique characteristics of acoustic waves. Variations in wave propagation, reflections, or resonances contribute to the uniqueness of responses.

Spintronic PUFs:

Construction: Leverages the spin properties of electrons for security. Construction involves spintronic devices such as magnetic tunnel junctions.

Working Principle: Exploits variations in electron spin states. Challenge-response pairs are generated based on the unique spin configurations, and the spin-dependent transport properties contribute to unpredictability.

Plasmonic PUFs:

Construction: Utilizes surface plasmon resonance effects in metallic nanostructures. Construction involves designing nanostructures with specific plasmonic properties.

Working Principle: Generates challenge-response pairs based on interactions of surface plasmons with external stimuli. Variations in plasmon resonance contribute to the uniqueness of responses.

Graphene-based PUFs:

Construction: Uses the electronic properties of graphene. Construction involves the integration of graphene sheets or devices into the PUF architecture.

Working Principle: Exploits electronic properties of graphene, such as conductivity.

Challenge-response pairs are generated based on variations in the electronic structure or conductivity, providing a unique fingerprint.

Superconducting PUFs:

Construction: Utilizes the unique properties of superconducting materials. Construction involves superconducting elements such as Josephson junctions.

Working Principle: Involves generating challenge-response pairs based on quantum properties of superconductors. Variations in superconducting states or flux quantization contribute to unpredictability.

Metamaterial-based PUFs:

Construction: Uses artificially engineered materials with unique electromagnetic properties. Construction involves designing metamaterial structures.

Working Principle: Exploits engineered electromagnetic responses of metamaterials. Challenge-response pairs are generated based on interactions of electromagnetic waves with metamaterial structures, ensuring a high level of security.

Quantum-dot Cellular Automata (QCA) PUFs

Construction: QCA PUFs use quantum-dot cellular automata as the underlying physical system. QCA is a nanoscale computing paradigm.

Working Principle: The working principle involves utilizing the unique configurations and state transitions of quantum dots in a cellular automaton. Challenge-response pairs are generated based on the quantum state configurations, providing a highly secure foundation.

Each of these PUF types offers distinct advantages and challenges, contributing to the diverse landscape of hardware security solutions. The selection of a specific PUF type depends on the application requirements and the desired level of security.Each type of PUF construction involves intricate designs to harness unique physical phenomena. The working principles are tailored to the specific physics of each technology, providing a diverse set of solutions for enhancing hardware security. The inherent variability in these physical systems makes it challenging for adversaries to predict or clone responses, ensuring a high level of security for cryptographic applications.

Following Pappu's initial works, the concept of a silicon PUF was first introduced by Gassend et al. (2002). In this study, Gassend et al. argued that a complex IC could

function as a silicon PUF and described a technique to identify and authenticate individual ICs. Further, the PUF was realized in a real silicon and was called an Arbiter-PUF Lee et al. (2004). The Arbiter-PUF exploits the delay discrepancies between two nominally identical delay paths. Another delay-based PUF called the Ring Oscillator-PUF (RO-PUF) was proposed in Suh and Devadas (2007), where the output response was generated based on the frequency disparities between a pair of ring oscillators (ROs). Moreover, in Guajardo et al. (2007), a memory-based PUF, which is based on the random start-up values (SUVs) of Static Random-Access Memory (SRAM) cells was proposed, wherein the power-up SRAM state was utilized as an identifying fingerprint, as also proposed in Holcomb et al. (2008). Another PUF employing cross-coupled latches, the Butterfly-PUF, was introduced in Kumar et al. (2008). targeting the protection of intellectual property (IP) designs in Field-Programmable Gate Arrays (FPGAs).

Additionally, a memory-based PUF utilizing a known cell structure of data bus keeper or data bus holder, i.e., cross-coupled inverters, was proposed in Vijayakumar and Kundu (2015) as the Buskeeper-PUF.

Strong, Weak and Controlled PUFs

Silicon PUFs can be categorized into three sub-types according to the security properties of their challenge-response behaviors, each with their own preferred applications. The three established types of PUFs are the strong PUFs Guajardo et al. (2007), the weak PUFs Guajardo et al. (2007), and the controlled PUFs Gassend et al. (2008).

1. Strong PUFs: Strong PUFs are with a very large number of CRPs (Ci,Ri), i = 1, ...N Guajardo et al. (2007). The number of CRPs of the considered PUFs grows exponentially as the number of bit challenges increases. The challenge-response interface is directly accessible without a protection mechanism in which the CRPs can be collected using a non-invasive CRPs measurement. Rührmair et al. (2013) refined the Strong PUF definition in which it must also be infeasible to be numerically modeled with a high prediction accuracy based on the observed CRPs (i.e., the PUF response is unpredictable).

2. Controlled PUFs: Controlled PUFs are improved Strong PUFs where the challenge-response interface is not directly accessible but it is protected by a logic processing unit using techniques, such as random hash function, permutation, ob-

fuscation, etc. Gassend et al. (2008) used the random hash function technique for the pre-processing of challenges before being input to the Strong PUF. In a similar way, the responses of the Strong PUF are post processed by the random hash function before being output by the Controlled PUF.
A model-building attack is one of the plausible attacks on strong PUFs Rührmair et al. (2013), Gassend et al. (2008), where the adversary builds a numerical model of the PUF by measuring a number of CRPs. The introduction of additional pre and post-processing steps for the Controlled PUFs increases the level of difficulty to measure and collect the CRPs. Hence, the vulnerability to a model building attack is reduced Gassend et al. (2008).

3. Weak PUFs: Weak PUFs possess a small number of CRPs and fixed challenges. Weak PUFs have only a single challenge in extreme cases Guajardo et al. (2007), Rührmair et al. (2013).

Based on the aforementioned definitions, the main distinction between Strong and Weak PUFs is the number of generated CRPs. As Strong-PUFs can support a large number of CRPs, they can provide authentication capabilities, particularly using a challenge-response protocol without having to store a secret key as a unique identifier. For Weak-PUFs, a limited number of CRPs means that these CRPs must be kept secret. Subsequently, Weak PUFs are well suited for a secret key generation for any cryptographic process, such as encryption/decryption and message authentication code (MAC). The preferred applications based on the aforementioned categorization are further discussed in Section 1.7.

1.6 Silicon PUF Constructions

Since the inception of the silicon PUF concept, as detailed in Gassend et al. (2002) and inspired by Pappu's research Pappu (2001), an enormous number of PUF techniques have been proposed in the past decade. This section categorizes silicon PUFs according to their construction and operating principles.

1. The primary category of construction involves delay-based PUFs, typically created from simple digital circuit structures. These exploit the intrinsic variations in the logic gate and interconnect delays to generate distinctive, device-specific random signatures. Delay based PUFs include Arbiter-PUFs Lee et al. (2004),

Lim et al. (2005), Suh and Devadas (2007), Majzoobi et al. (2008) and RO-PUFs Gassend et al. (2002), Suh and Devadas (2007).

2. The second class of construction is memory-based PUFs, which exploit intrinsic variations in bi-stable memory elements, such as SRAM PUFs Guajardo et al. (2007), Holcomb et al. (2008), D Flip-flop PUF Van der Leest et al. (2010), Butterfly PUF Kumar et al. (2008), SR-NOR latch PUF Su et al. (2008), and Buskeeper PUF Simons et al. (2012).

3. The third class of construction is mixed-signal PUFs in which the PUF behavior is inherently of an analogue nature and require analogue-to-digital converter (ADC) to digitize the measured analogue responses. A mixed-signal PUF typically exploits the variability of minimum size MOSFETs to enhance the threshold voltage mismatches, such as in the Voltage Transfer Characteristic PUF (VTC-PUF) Vijayakumar and Kundu (2015) and Current Mirror-PUF Kumar and Burleson (2014).

4. The final class of construction is PUFs built from emerging nanotechnology devices in order to further achieve secure, robust, and lightweight PUF designs Gao et al. (2016b).

The following sections discuss in detail the construction of the aforementioned PUFs. Though this is not a comprehensive discussion on different PUFs proposed so far, it gives an overview of PUF development in general.

1.6.1 Arbiter-PUF

In Arbiter Physically Unclonable Functions (APUFs), the unpredictable nature of individual responses is intricately tied to the delays in connecting lines, which introduce inherent variations in signal propagation times. This variability contributes to the challenge of predicting specific outputs for given inputs, adding a layer of complexity crucial for the security of APUFs. To comprehensively analyze the security of APUFs, Author in Roy et al. (2022) employed Boolean function mapping, treating the APUF responses as Boolean functions with challenge bits as inputs and corresponding response bits as outputs. This approach facilitates a more nuanced exploration of the statistical properties of the output sequences. Auto-correlation analysis is then ap-

plied to these Boolean functions, aiming to identify potential biases or patterns that might exist in the generated sequences.

While the delays in connecting lines ensure the overall unpredictability of individual APUF responses, Boolean function mapping enables a deeper understanding of the relationship between challenges and responses. we utilize this mapping to identify constraints on the Boolean functions that influence the auto-correlation properties of the response sequences. These constraints may involve specific conditions on input weights and differences in input bits. Applying these constraints based on the Boolean function mapping helps eliminate biases or patterns in auto-correlation, fortifying the overall security of APUFs. In essence, the delays contribute to the inherent randomness of APUF responses, Boolean function mapping provides a systematic means to analyze and understand this unpredictability, and autocorrelation analysis ensures that the collective behavior of the responses remains statistically random. This holistic approach, grounded in Boolean function estimation and mapping, strengthens the security and reliability of APUFs in cryptographic applications, safeguarding against potential vulnerabilities introduced by identifiable patterns or biases in the output sequences.

Lee et al. presented the first PUF which was fabricated on silicon using a TSMC 180-nm CMOS technology called Arbiter-PUF Lee et al. (2004). This PUF circuit exploits the logic gate delays and interconnect variations that are affected by the process variations. The Arbiter-PUF consists of k stages or switching component, where each stage is composed of two 2-to-1 multiplexers, as shown in Figure 1.8 Lim (2004). A rising pulse at an input propagates through two nominally identical delay paths. The paths for the input pulse are controlled by the switching elements, which are set by the bits of the challenge, $C = c_1, c_0..., c_k$. The paths are straight for $c_k = 0$ and are crossed for $c_k = 1$. A delay difference of Δt is caused between the paths because of manufacturing variations. An arbiter at the end generates a random response, '0' or '1', depending on the difference in arrival times.

Ideally, an Arbiter-PUF response is equally likely to be '0' or '1' in which it is solely dependent on the randomness in process variations. However, the response could be biased to be either '0' or '1'. A large bias reduces the uniqueness and randomness of Arbiter-PUF responses. The following two important criteria need to be fulfilled at the design stage for avoiding the bias:

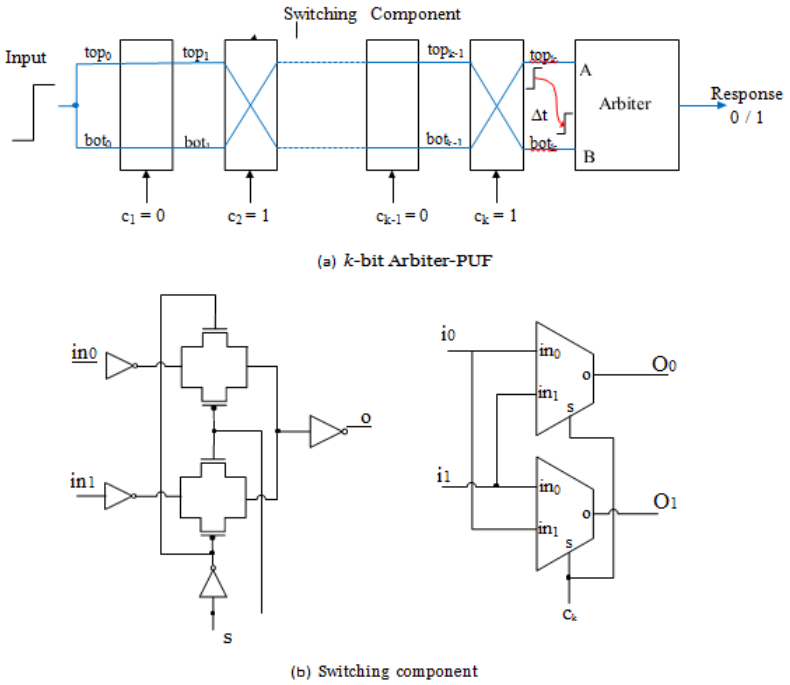


Figure 1.8: The construction of Arbiter-PUF as proposed in Lim (2004)

1. The degree of symmetry needs to be the highest in the layout to ensure an unbiased response of the Arbiter-PUF. The symmetry routing includes the routing before the first switching element, inside the switching element, in between stages, before and inside the arbiter circuit. Although it is non-trivial, it is possible to achieve a symmetrical routing in application-specific integrated circuits (ASICs) Lee et al. (2004). The implementations on FPGAs seem to be difficult due to the placement and routing constraints of the general FPGA hardware architecture Roel and Verbauwhede (2010).

2. An unbiased arbiter circuit must be used for digitization of the delay difference. According to Lin et al. (2012), fair arbitration can be achieved by using an SR-latch that has a symmetric circuit topology.

This follows the inference from Siddhanti et al. (2019). It introduces Arbiter-based Physically Unclonable Functions (Arbiter PUFs) as a method to generate cryptographically secure secret keys during runtime, addressing the vulnerability of storing keys in Non-Volatile Memory (NVM). However, the Arbiter PUF construction is susceptible to statistical and modeling attacks, particularly leaking information if certain

challenge-response pairs are known. The paper identifies a statistical weakness in the form of bias in response, specifically towards the effect of flipping certain bits, known as the Strict Avalanche Criterion (SAC). To address this, the authors propose a generalized framework for analyzing any Arbiter PUF variant against SAC and introduce a new variant that is highly resistant to SAC while maintaining good reliability.

The Strict Avalanche Criterion (SAC) is a property that a cryptographic function or primitive should possess. It states that if a single bit in the input of the function is flipped, then each output bit should change with a probability of 1/2. In other words, a small change in the input should cause a drastic and unpredictable change in the output. In the experiment section, the proposed S-PUF is implemented in Verilog with placement constraints and tested on a Nexys-4 DDR Artix-7 FPGA board. The responses to challenges are recorded, and the performance metrics are discussed. The focus is on the PUF's ability to generate different combinations of responses for a given challenge, with the Intra Hamming distance used to assess the uniqueness of responses. The results indicate that the proposed S-PUF exhibits a Gaussian-like plot of Hamming distance, demonstrating its capability to generate diverse responses over its length.
The conclusion of the paper emphasizes theoretical estimations of observed experimental biases and their verification through hardware and software experiments. The authors claim that biases, determined theoretically, cannot be computationally obtained for several PUFs with large inputs. Additionally, a new construction of a PUF is introduced, presented as secure against existing attack techniques. The implemented S-PUF on an Artix-7 FPGA shows good uniqueness, bit-aliasing, and uniformity. The design achieves a reliability of 92%, with errors recoverable using error correction mechanisms.In summary, the paper contributes to the literature by addressing statistical weaknesses in Arbiter PUFs, proposing a generalized analysis framework, and introducing a new variant with enhanced resistance to the Strict Avalanche Criterion. The experimental results on the implemented S-PUF further validate its uniqueness, reliability, and resistance to biases observed in other PUF constructions.

For an Arbiter-PUF, although the effect of process variations on logic gates and interconnect delays tends to be random, a low probability of encountering meta-stability could be possible. This likelihood arises when minimal timing discrepancies occur between the rising edges of inputs A and B within the arbiter circuit. Thus, to mitigate

the influence of arbiter bias and the associated meta-stability effects, the preference leans towards a fair arbiter circuit, such as the symmetric SR-latch, which has a symmetric circuit topology Maes et al. (2012).

The construction of an Arbiter-PUF, as in Figure 1.8, is based on additive delays

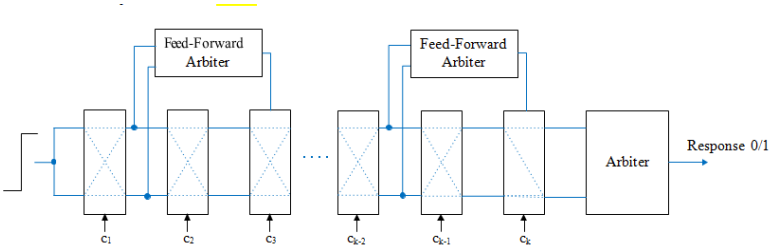


Figure 1.9: The construction of Feed-Forward Arbiter-PUF as proposed in Lim et al. (2005)

caused by individual switching components with linear characteristics Lim (2004). Consequently, the complexity of the CRPs mapping is minimal, facilitating susceptibility to model based attacks using ML techniques. A few derivatives of the Arbiter-PUF have been proposed to introduce non linearity into its structure. Lim et al. (2005) presented the Feed-Forward Arbiter-PUF that uses the outputs of intermediate arbiters to configure subsequent switching components, as depicted in Figure 1.9. However, the intermediate arbiters increase the probability of metastability states, potentially resulting in increased errors in the PUF response.

Study, Majzoobi et al. (2008) proposed a Lightweight-PUF consisting of l parallel Arbiter-PUFs. The outputs of these Arbiter-PUFs are XORed to generate an overall PUF response O1, O2, and O3, as shown in Figure 1.10. The inclusion of an input XOR network in the Lightweight-PUF is expected to increase non linearity in the CRPs' mapping. Earlier, l-XOR Arbiter-PUF was proposed in Suh and Devadas (2007), consisting of l parallel Arbiter-PUFs, as illustrated in Figure 1.11.

In another study from Singh et al. (2022) ,talks about the Priority Arbiter PUF(P-APUF). This paper introduces a novel physically unclonable function (PUF) design centered around a 3-input priority arbiter. The priority arbiter, constructed with a simple arbiter, two 2:1 multiplexers, and an XOR logic gate, exhibits excellent uniformity (49.45%) by yielding an equal probability of 0's and 1's at the output. Building upon this, a Priority Arbiter-based PUF (PA-PUF) is presented, allowing configura-

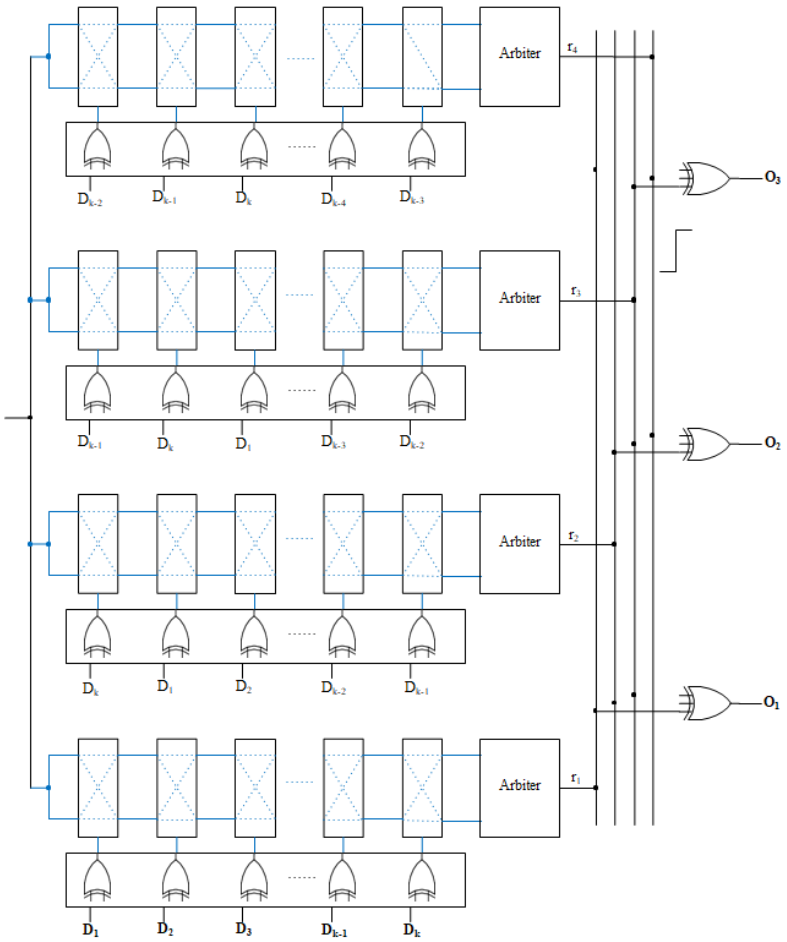


Figure 1.10: : The construction of Lightweight-PUF as proposed in Majzoobi et al. (2008) for l=4

bility in challenge-response pairs through the integration of an arbitrary number of feed-forward priority arbiters. The PA-PUF design is rigorously evaluated for uniqueness, non-linearity, and uniformity, demonstrating a reliability of 100% post error correction techniques and a uniqueness of 49.63%. Comparative analysis with existing literature affirms the superior implementation efficiency of the proposed design. The study concludes by outlining plans for a more detailed exploration of attacks on PA-PUF, both practically and theoretically, in future research. The experimental results, conducted on a Nexys Video Artix-7 FPGA board, underscore the reliability, uniformity, and uniqueness enhancements achieved by the proposed PA-PUF design.

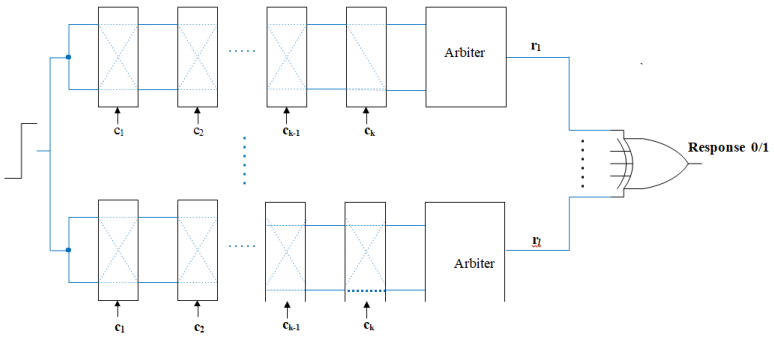


Figure 1.11: The construction of the l-XOR Arbiter-PUF as proposed in Suh and Devadas (2007)

In another study from Sahoo et al. (2015) In programmable Delay Line(PDL) PUFs , Hardmacro feature plays a very important role. Importance of Hard Macro in FPGA for PUF Implementation.In the realm of Physical Unclonable Functions (PUFs) implemented on Field-Programmable Gate Arrays (FPGAs), the use of Hard Macro features is of paramount significance. A Hard Macro is a predefined and pre-characterized block of digital logic that can be instantiated within an FPGA design. In PUF implementation, this feature plays a crucial role in achieving bias-free and high-quality results. Here's why:

- 1. Design Consistency: - Hard Macros provide a level of design consistency by offering a predefined and characterized block of logic. This consistency is essential for ensuring that the PUF implementation across multiple FPGAs maintains the desired characteristics without unintended variations.

- 2. Elimination of Bias: - PUFs, and particularly Arbiter PUFs (APUFs), are sensitive to variations in delay paths. Hard Macros enable the creation of symmetric delay paths that are bias-free, reducing the risk of unintentional biases that could compromise the security and uniqueness of the PUF-generated keys.

- 3. Hardware Efficiency: - Leveraging Hard Macros contributes to hardware efficiency by streamlining the implementation process. By utilizing predefined logic blocks, the design methodology becomes more straightforward and resource-efficient, enabling the creation of PUFs with reduced complexity and improved performance.

- 4. Consistent Performance Across FPGAs: - The use of Hard Macros ensures

consistent performance characteristics across different FPGAs. This is particularly crucial in applications where uniformity, reliability, and uniqueness of PUF responses are paramount. It facilitates reliable and reproducible outcomes, critical for cryptographic applications.

From the observation of Sahoo et al. (2015), authors address the challenge of achieving high-quality, bias-free implementations of Arbiter Physical Unclonable Functions (APUF) on FPGAs, with a focus on the Programmable Delay Line (PDL)-based APUF design. They propose a scalable design methodology for implementing close-to-ideal APUFs on Xilinx FPGAs, utilizing the Hard Macro feature of the Xilinx CAD tool flow to design bias-free symmetric delay paths. The study demonstrates the effectiveness and superiority of their design over previously proposed PDL-based PUFs through implementation and characterization results.

The experimental evaluation was conducted on Spartan-3 (XC3S400) FPGAs under normal operating conditions. The authors implemented a PDL-APUF with 30 PDL cores on each PDL chain, evaluated it with 5000 random challenges, and reported results for average uniformity, reliability, and uniqueness. The proposed design, utilizing only four tuning elements on both upper and lower paths, achieved superior APUF implementations compared to previously proposed approaches. The average uniformity outperformed previous designs, with the proposed scheme being highlighted as hardware-efficient.

In conclusion, the paper presents a design methodology that systematically eliminates sources of bias in PDL-APUF, leveraging the "hard macro" feature of Xilinx CAD tools. The implementation results indicate substantial improvements over existing PDL-APUF designs in terms of uniformity, reliability, and uniqueness.

1.6.2 Ring Oscillator PUF

The ring oscillator PUF (RO-PUF) is a type of delay-based PUF. The working principle of the RO-PUF relies on measuring the frequencies of digital oscillating circuits that randomly vary due to the uncontrollable effects of silicon process variations on logic gates and interconnect delays. The notion of a PUF that uses an oscillating circuit was first proposed by Gassend et al. (2002). Further, Suh and Devadas (2007)

proposed an RO-PUF architecture that consists of an array of n nominally identical ROs. The architecture also contains two frequency counters to count the number of rising edges at the selected RO pairs. Two n-to-1 multiplexers control the pair of ROs being applied to both counters. The multiplexers' selection signals act as the PUF's challenge. The construction of an RO-PUF is depicted in Figure 1.12. Both frequency counters are activated for a fixed duration and the resulting counter values are compared to generate a response, '0' or '1', based on which oscillator from the selected RO pair is faster. Since the process variations affect the frequencies of the n RO arrays, the resulting comparison bit becomes random and specific to the device.

Each comparison of a pair of oscillators generates a bit. Thus, given an array of n oscillators, a total of $\frac{n(n-1)}{2}$ pairs can be compared. However, the number of independent bits produced is less than $\frac{n(n-1)}{2}$. For example, if oscillator A is faster than B and if oscillator B is faster than C, then it is evident that A will also be faster than C, resulting in a correlated pair-wise comparison. Suh and Devadas (2007) concluded that the maximum number of uncorrelated comparisons is limited by the possible orderings in which oscillators can be ordered. Given n oscillator frequencies, which are independent and equally distributed, there exist n! equally likely possible orderings. Hence, the maximum entropy (representing uncorrelated pair-wise comparisons) of an RO-PUF is log2 n!. It is also possible to avoid any correlation by simply comparing a pair of ROs once, resulting in only n response bits.

In the study fromHesselbarth et al. (2018), the research team employed multiple t-Welch tests to draw important inferences regarding the comparison of Ring Oscillators (ROs) in a specific context. The findings strongly advocate for comparing ROs exclusively within the same type to prevent bias in responses. For instance, ROs implemented in SLICEM were discovered to be approximately 5 MHz slower on average compared to those in SLICEL. The study also highlighted the significance of considering the location of RO implementation. ROs situated in proximity to clock routing resources exhibited an impact of approximately 20 MHz, while those near other fast-switching circuitry showed an impact of around 10 MHz. To ensure accurate comparisons, it is advisable to avoid such configurations.

The research team identified a critical concern related to large time-to-response in RO-PUFs and provided insights into minimizing it. They explained the optimization of the trade-off between evaluation time and measurement precision. Remarkably, for

the specific architecture and design under consideration, the optimal trade-off was determined to be 70.71 µs.

In summary, the study underscores the importance of careful RO comparison within the same type, taking into account the impact of implementation location, and addresses the critical issue of time-to-response by proposing a specific optimal trade-off for the given architecture and design.

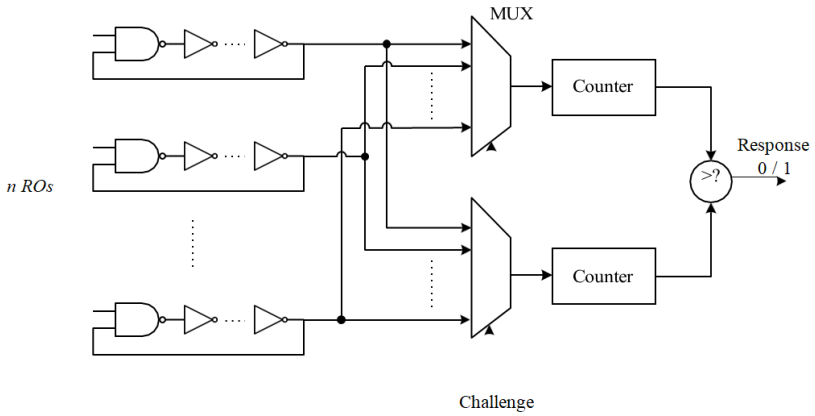


Figure 1.12: The construction of an RO-PUF as proposed in Suh and Devadas (2007)

1.6.3 SRAM-PUF

The previously discussed PUFs are all delay-based PUFs that require additional circuitry to use them as an on-chip hardware security. Another suggestion is to use on-chip resources as a PUF, particularly using SRAM, which is available in any computing system Guajardo et al. (2007), IntrinsicID et al. (2015), Holcomb et al. (2008). Guajardo et al. (2007) exploited SRAM memory as an intrinsic PUF for using it as a secret key generator for FPGA bitstream encryption/decryption to support an IP protection. Holcomb et al. (2008) targeted wider applications of SRAM as PUFs to support resource-constrained security-critical applications, such as contactless credit cards, etc. Generally, SRAM memory is constructed based on rows and columns of bit cells. The number of available bit cells in an SRAM represents its storage size. Each bit cell of SRAM is typically a six-transistor CMOS circuit that consists of two cross-coupled inverters (MP1, MP2, MN1, and MN2) and two access transistors (MN3 and MN4), as illustrated in Figure 1.13.

The transistors form the cross-coupled inverters and are also known as a bi stable element, in which each node Q and QB can be in either of the two states, '0' or '1'. The power-up or SUVs of bi-stable elements depend on the device mismatches, which result from process variations. During the power-up of an SRAM cell, the current flowing through MN1 and MN2 will slowly pull up the voltage at nodes Q and QB as the supply voltage increases. Due to the random process variations, transistor MP2 has a slightly higher threshold voltage compared to that of MP1. Consequently, the current that flows through MN1 is slightly higher than through MN2, thus turning the MN2 ON and pulling down node QB to GND. Simultaneously, when node QB is discharging, MP1 is turned ON and node Q is pulled up to Vdd. The nodes Q and QB settle at '0' and '1', respectively. When powering-up the SRAM, the SUVs across different memory blocks within an SRAM and across multiple SRAMs show device-specific and random patterns, which are the desired qualities to be used as a PUF Guajardo et al. (2007), Holcomb et al. (2008), as shown in Figure 1.14.

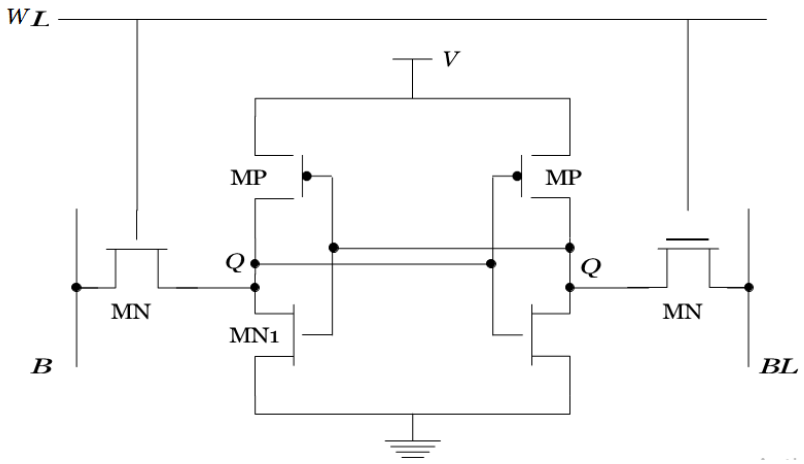


Figure 1.13: 6-T SRAM cell circuit

1.6.4 Flip-flop, Latch and Buskeeper PUFs

PUFs based on flip-flops and latches are derived from the same working principles and physical effects as SRAM-PUFs. Kumar et al. (2008) proposed a Butterfly PUF that consists of a pair of cross-coupled latches, forming a bi stable circuit, as illustrated in

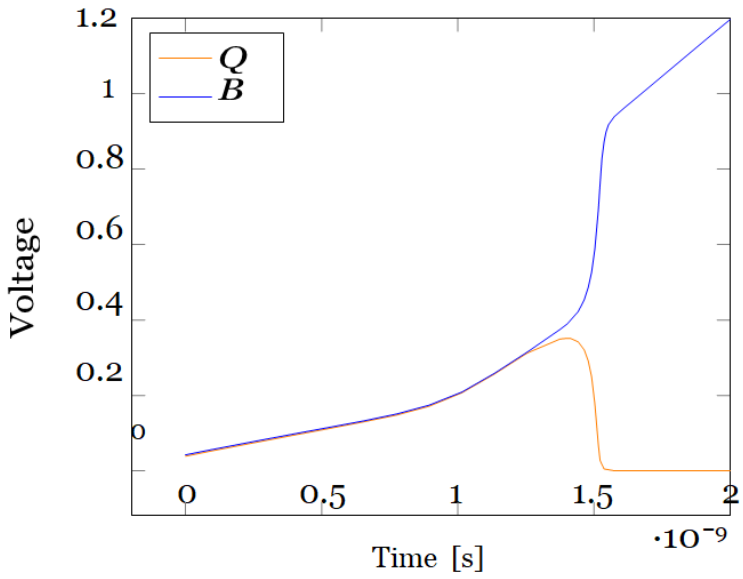


Figure 1.14: Bi-stable SRAM internal nodes, Q and QB resolving to '1' and '0' during power-up process

Figure 1.15(a). When the preset (PRE) and clear (CLR) signals are simultaneously set to high, the Butterfly PUF can be forced into an unstable state and converged into a stable state when both signals are set low after a few clock cycles. The Butterfly PUF was proposed to overcome the disadvantages of the SRAM-PUF on FPGA platforms, whereby the SRAM is cleared after power-up on most of the commercial FPGAs. However, the Butterfly PUF requires symmetrical routing to minimize the impact of design mismatch that is not trivial to achieve due to the routing constraints on FPGAs Roel and Verbauwhede (2010). In another study, a PUF based on the power-up behavior (similar to the SRAM PUF) of clocked D flip-flops was proposed by Van der Leest et al. (2010). Most D flip-flops are based on two latches, as depicted in Figure 1.15(b). The advantage of D Flip-flop PUF is that the location of the individual flip-flop can be randomly spread across a design and their signal lines connecting them to the read-out circuitry could be obfuscated, thereby increasing the defense against invasive attacks, such as probing attacks.

Su et al. (2008) proposed a device identification technique based on two cross-coupled NOR-gates that constitute a simple SR-NOR latch, as shown in Figure 1.15(c). Upon the activation of a high reset signal, the SR-NOR latch enters an undefined state,

eventually stabilizing into a stable state determined by the internal mismatch among the NOR gates when the reset signal becomes low. The SR-NOR latch uses the same working principle as SRAM-PUF; however it does not rely on a power-up state that depends on supply voltage. Subsequently, it provides increased flexibility because it can generate unpredictable and reproducible responses when the reset signal is (re)asserted at any instance when a device requires the secret key. Another variant of latch based PUF is the Buskeeper PUF, proposed in Simons et al. (2012), which is constructed using bus keeper cells. The basic structure of a bus keeper cell is a cross coupled inverter with a low drive-strength, interfaced with a bus line, as shown in Figure 1.15(d). A bus keeper cell is used to maintain the last driven state on the bus line and prevents the bus line from floating. Similar to SRAM cells, the power-up state of bus keeper cells is determined by the device mismatches that result from process variations. An advantage of the Buskeeper PUF is the low area overhead in comparison to other types of memory based-PUFs, such as D Flip-flop PUFs.

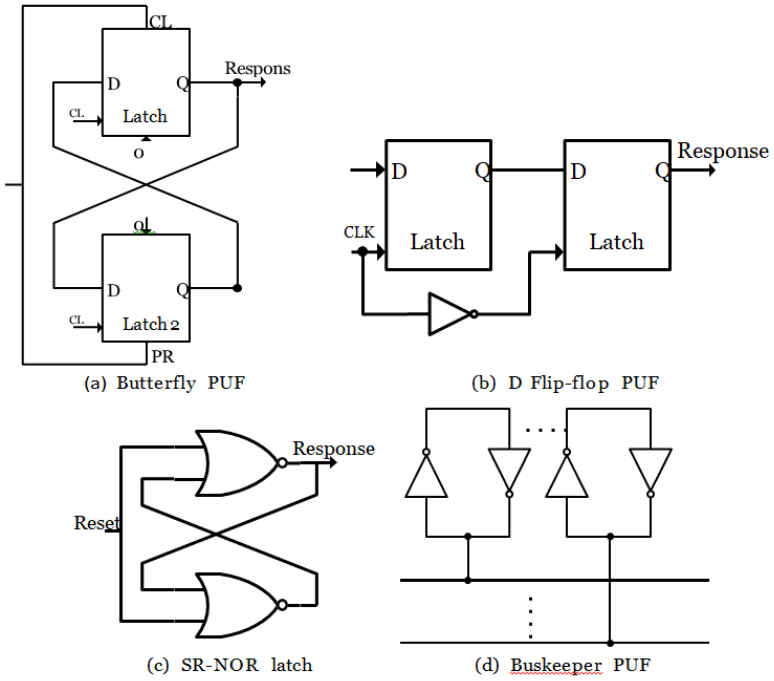


Figure 1.15: The construction of Butterfly PUF Kumar et al. (2008), D Flip-flop PUF Van der Leest et al. (2010), SR-NOR latch PUF Su et al. (2008) and Buskeeper PUF Simons et al. (2012)

1.6.5 Mixed-Signal PUFs

Vijayakumar and Kundu (2015) introduced the VTC-PUF, a mixed-signal PUF by exploiting the variability in a voltage transfer characteristic (VTC) circuit, as shown in Figure 1.16(a). The VTC circuit has a non-linear relationship between input and output voltages in which the non-linearity is controlled by the feedback transistor, M3. Adapted from the architecture of Arbiter-PUF, the VTC circuit is used as a basic building block in which it is coupled with a switching component in each stage, creating a cascaded circuit, as depicted in Figure 1.16(b). The switching components, created from a simple transmission gate-based circuit, interact with the input node connected to Vdd . A voltage sense amplifier is used to detect the difference of the propagated input voltages and generate a response, '1' or '0', based on the sign of the differential output at the final stage.

Kumar and Burleson (2014) proposed a Current Mirror-PUF, which is also adapted

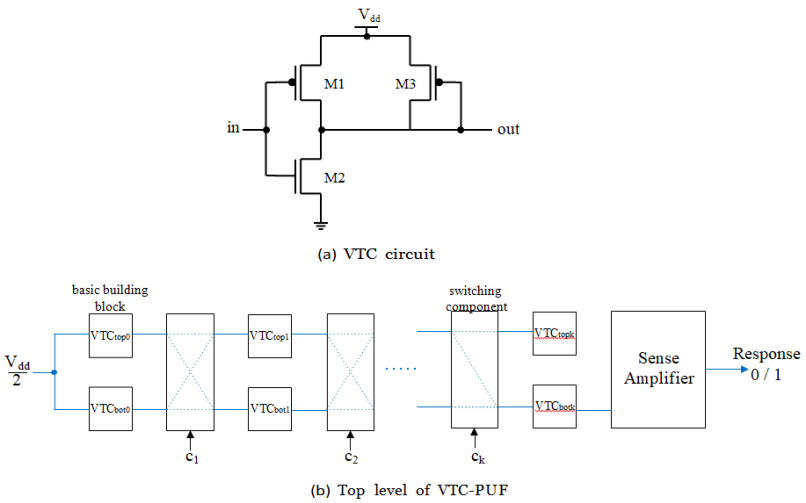


Figure 1.16: The construction of VTC-PUF Vijayakumar and Kundu (2015)

from the Arbiter-PUF architecture, as shown in Figure 1.16. However, the basic building block is the current mirror circuit, as illustrated in Figure 1.17. Iout is the mirrored current of Iref which varies slightly due to the process variations experienced by the transistors, M1-M4. Similar to the VTC-PUF, the switching components are created by a simple transmission gate based circuit, but the input is connected to a constant current source. A current sense amplifier is used to detect the difference of the propagated input currents and generate a response, '1' or '0', based on the sign of

the differential output at the final stage.

In another study, Cao et al. (2015) proposed a mixed-signal PUF based on a CMOS

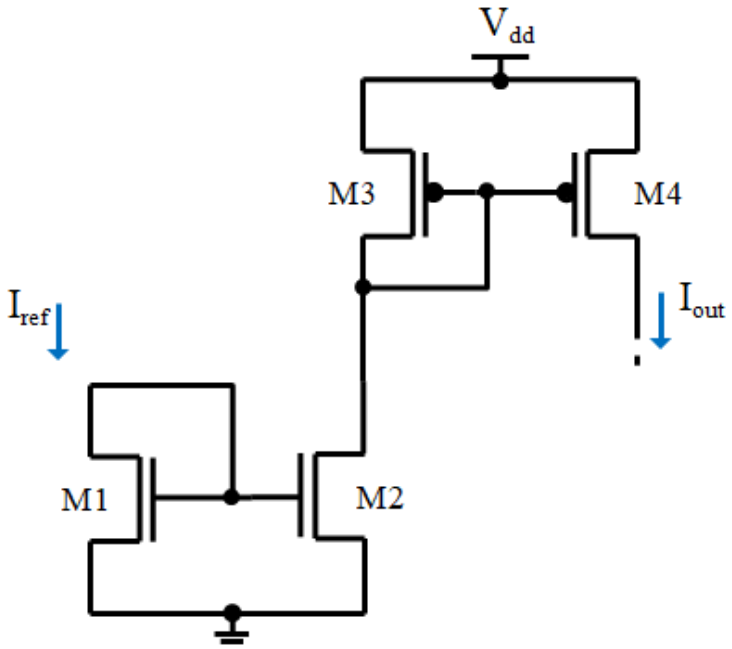


Figure 1.17: The construction of the current mirror circuit in the Current Mirror-PUF Zhang et al. (2014)

image sensor circuit. The image sensor function was not affected when operated in a PUF mode. The core of a CMOS image sensor is a pixel array and its typical structure is depicted in Figure 1.18. This PUF exploits the variations of the pixel output voltage during the reset phase, which varies due to the variations in the threshold voltages of transistors MRS and MSF. The two reset voltages of two pixels are compared to generate a response, '0' or '1', based on the larger reset voltage. A predefined threshold is introduced during the comparison of reset voltages that effectively increases the reliability of the response by about 10.8%.

Sehwag and Saha (2016) proposed a fast and lightweight mixed-signal PUF that exploits the susceptibility of the threshold voltage of MOSFETs to process variations; hence, the name "Threshold Voltage PUF" or TV-PUF. The core circuit in the TV-PUF is a block consisting of n cascaded transistors, as depicted in Figure 1.19. When $V_{IN} = V_{dd}$, the output voltage at the source terminal of the cascaded series of transis-

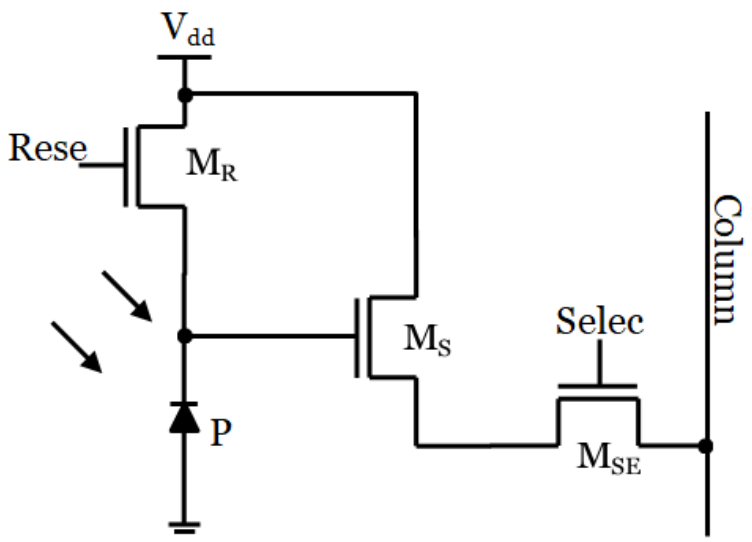


Figure 1.18: 3-T pixel circuit

tors becomes $V_{\text{out}} = V_{dd} - V_{th1} - V_{th2} - \ldots - V_{thn}$The output voltage depends on the threshold voltages of n cascaded transistors that are susceptible to random process variations. Due to this, the output voltages of two nominally identical blocks are different and are compared using the sense amplifier to generate a 1-bit response. A decoder circuit is used to set the level of V_{IN} to HIGH (V_{dd}). In order to address a unique challenge (i.e., an input decoder), only one decoder output is set to a HIGH state, which is connected to two blocks of n cascaded transistors.

1.6.6 Emerging Nanotechnology-based PUFs

All of the aforementioned PUF designs focus on exploiting process variations intrinsic to the CMOS technology. Recently, nanotechnology-based PUFs have made progress since the scaling down to the nano region has resulted in an increased variability in nano electronic devices. PUFs that have been proposed include the carbon-nanotube field-effect transistors (CNFET) based PUF Konigsmark et al. (2014), the phase change memory (PCM) based PUF Zhang et al. (2014), and the Memristor-based PUF Gao et al. (2015). Nanotechnology-based PUFs offer a few advantages over CMOS based PUFs, such as substantial process variations, small footprints, and lower energy consumption Gao et al. (2016b). Since this study focuses on CMOS silicon-based PUFs, one may refer to Gao et al. (2016b) for a comprehensive discussion on PUFs

based on emerging nanotechnology devices.

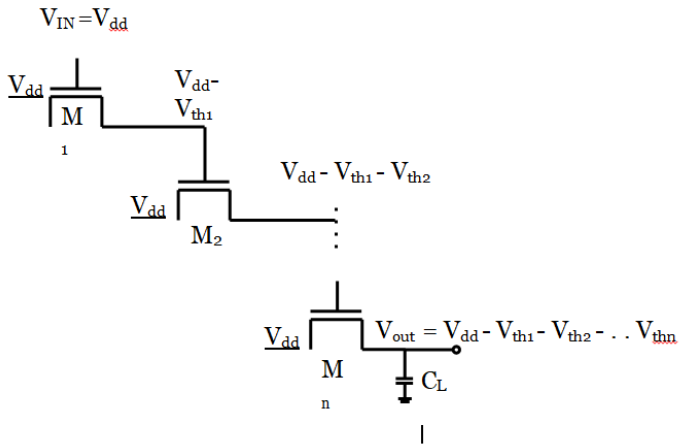


Figure 1.19: The construction of n cascaded transistors of one block in TV-PUF Sehwag and Saha (2016)

Table 1.1: Comparison results in percentage - Arbiter PUF with Ring Oscillator Based PUF(Maiti et al. (2013))

	APUF	**RO PUF**	**ideal value**
Uniformity	55.69%	50.56%	50%
Bit-aliasing	19.57%	50.56%	50%
Uniqueness	7.2%	47.24%	100%
Reliability	99.76%	99.14%	100%

The metrics shown in the table would measure the performance of a PUF. A PUF's capacity to distinguish one chip from another of the same type is referred to as its uniqueness. A pair of PUF identifiers are compared using Hamming distance (HD) to determine uniqueness Maiti et al. (2013). APUF with its linear characteristics will always be biased towards a common path taken for various challenges, resulting in lower uniformity. Conversely, the non linear characteristics of Ring Oscillator PUF tend to exhibit a better performance in this metric. Reliability measures a PUF's efficacy in reproducing response bits, gauged through intra-chip HD among multiple samples of PUF response bits. Table 1.1 shows that both RO PUF and APUF perform well in reproducing response bits.

Uniformity assesses the balance of '0's and '1's in PUF response bits, ideally maintaining a 50% fraction for fully random responses. Linear characteristics, as in Arbiter PUF, cause bias, while the non-linear nature of Ring Oscillator PUF showcases superior performance in this metric. Bit-aliasing, the phenomenon where multiple chips produce almost identical PUF responses, influences uniqueness, as observed in the results presented in the table. Bit-aliasing of the lth bit in the PUF identifier is estimated as the percentage Hamming Weight (HW) of the lth bit of the identifier across k devices. Bit Aliasing is one of the reasons for affecting the uniqueness of the chip.

These metrics are used for evaluating PUFs and this research assesses the performance of designed PUFs with these references.

Despite the existence of other delay-based PUF circuits, such as Bistable Ring PUF, Loop PUF, and Glitch-based PUF circuits, their implementation on FPGA devices often presents more disadvantages than advantages. Most of the literature focuses on arbiter based PUFs and RO PUF in the implementation specific to FPGAs. Additionally, Memory Based PUFs, such as SRAM-PUF, Butterfly PUF, Latch PUF, Flip-Flop PUFs, etc., face challenges when implemented on SRAM based FPGAs duct to initialization logic clearing unused configuration memory cells to predefined values, which is not desirable for SRAM PUF on FPGAs. Furthermore, Guajardo et al. (2007) observed that the amount of randomness in the power-up values of D-Flip-flops is limited due to their non-uniform distribution, skewed towards '0'. This necessitates further post-processing to render these values suitable for use as random keys.

1.7 Applications of Physical Unclonable Functions

- **Device Identification:** The designed PUF circuit helps in the identification of device, as the generated data would be unique for every device.
- **Key generation and storage:** As shown in Figure 1.20 (Bernard et al., 2012), PUF helps in generating the key/data from the device itself, instead of storing

in external or on chip memory. Thus, it provides more security in comparison to the traditional way. It generates the data only when the PUF device is on and robust post processing is needed on the response for the key to be reliable and random.

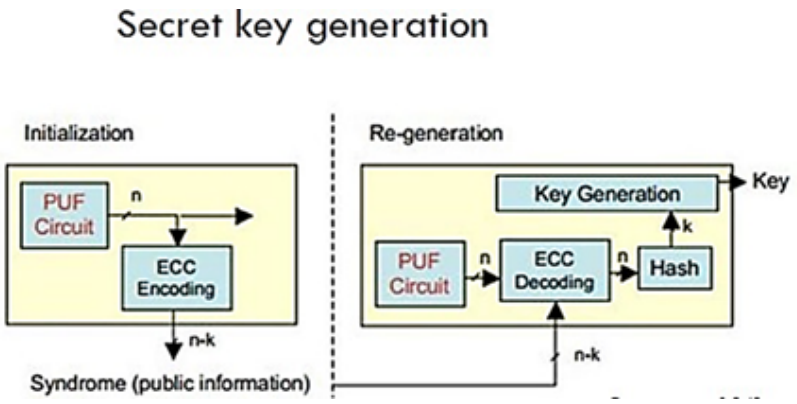


Figure 1.20: Application of PUF as key generation Kodỳtek and Lórencz (2015)

- **IP Protection/authentication:** It helps in providing security to device protocols with the help of PUF challenge response pair (CRP) dataset. It also helps in authenticating the device when a user has a CRP dataset. The device is authenticated whenever a user sends a authentication challenge and if the response is matched.

- **Public key encryption:** PUF finds practical application in public key encryption by generating data encrypted as a secret key. Subsequently, only the person possessing the corresponding public key has the capability to decrypt this data.

- **Timed authentication:** It serves as a timed authentication method for devices, differentiating genuine device response times from those generated by model building attacks.

- **Software licensing:** The PUF acts as an identification (ID) for the chip, playing a vital role in licensing the software to provide security for both the processor and the memory. This program is copy protected, ensuring that it exclusively operates on the designated chip and not on any other chip.

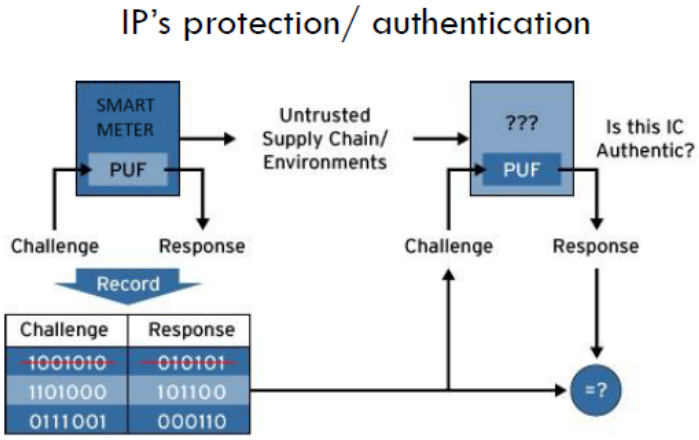


Figure 1.21: Application of PUF as device authentication Herder et al. (2014)

The application of low-cost authentication is generally associated with robust PUF architectures, as shown in Figures 1.9 and 1.10. An Arbiter PUF circuit generates two delay pathways of identical layout length for each input X, yielding an output Y based on the faster pathway. In this situation, a powerful PUF will replace a secure memory and cryptographic hardware in an embedded device, facilitating secure device identification to a server. A PUF-based solution requires less area, power, and mask layers than a typical approach to secure authentication because it does not require secure nonvolatile memory, anti tamper circuitry, or additional supporting crypto acceleration hardware. The requirements of a strong PUF stipulate that an adversary presented with polynomial CRPs should not be able to predict the response to a fresh challenge. While being desirable, this quality poses a challenge in utilization. As the PUF functions as a "black box", even the authentication server can solely access previously observed CRPs and hence cannot predict responses to new challenges. Consequently, the protocol for employing PUFs differs greatly from that of typical public/private key cryptography systems.

1.7.1 Securing Smart meters

Smart meters are devices that measure the energy consumption of consumers and communicate with the utility company or the system operator through a smart grid network. Smart meters can also receive information, such as pricing, demand response,

or outage notifications, from the utility company or the system operator. Smart meters are part of the Advanced Metering Infrastructure (AMI), which is a key component of the smart grid.

Installed in various consumer locations like homes, offices, and public buildings, smart meters play a pivotal role in enabling users to engage in demand response programs, energy efficiency initiatives, and distributed generation schemes. Smart meters can replace the traditional electro-mechanical or electronic meters that require manual reading or one-way communication. Smart meters can provide real-time or near-real-time data on energy usage, power quality, and other parameters to the utility company, system operator, and consumers. Smart meters can enable consumers to participate in demand response programs, energy efficiency measures, or distributed generation schemes.

The connection layout between the PUF and the smart meter depends on the type and design of the PUF and the smart meter. Generally, the PUF circuit can be integrated into the smart meter chip or board or it can be attached as a separate module. The PUF circuit can be connected to the smart meter processor or controller through a bus or a serial interface. The PUF circuit can also be connected to other components of the smart meter, such as sensors, memory, display, or communication modules. The PUF circuit can provide authentication, identification, encryption, or key generation functions for the smart meter.

In Figure 4.10, the PUF nodes represent smart meters with PUF circuits integrated into the smart meter chip. The PUF circuit receives a challenge from the smart meter processor through a bus interface, generates a response based on its physical characteristics, and sends it back to the processor for verification against a reference value stored in a secure memory. Once authenticated, the smart meter communicates with the utility company or system operator via a communication module. It is used in the process of securing the smart meter.

The use of PUFs involves a series of steps for the security of the smart meter. The server authenticates the client, (in this case a smart meter at the customer's premises) by engaging in challenges and responses with the PUF-based IC (Figure 1.21) ((Herder

et al., 2014). The server maintains a secure table of CRPs, validating responses and marking used CRPs to avoid repetition. Strategies to address scalability issues related to CRP tables involve innovative protocols based on compact PUF models. The following are the steps for using PUFs for the security of the smart meter.

1. PUF is produced.
2. The server gains access to the PUF and creates a table of CRPs. These pairs are kept in a secure location.
3. The PUF is supplied to the client.
4. The client sends an authentication request to the server.
5. The server selects a known CRP and issues the challenge to the client.
6. The client executes the challenge on the PUF and sends the result to the server.
7. The server validates the answer and marks the CRP as being utilized. Because the server cannot foresee PUF behavior, it must store CRPs internally for eventual use. Each CRP can only be used once. As a result, the server must either keep enough CRPs to avoid running out, or it must "recharge" the table on a regular basis by establishing secure communication with an authenticated client and requesting responses to new challenges. To overcome the scalability issue inherent in CRP tables, innovative protocols based on the storage of a compact model for PUF have emerged.

Since PUFs were introduced in Pappu et al. (2002), Gassend et al. (2002), Lee et al. (2004), Guajardo et al. (2007), Holcomb et al. (2008), they have been receiving a lot of attention because of their potential for being robust and cost-efficient hardware-based security devices. Based on the sub-types of silicon PUFs as discussed in Section 1.5.1, PUFs have been proposed for use in two primary applications Suh and Devadas (2007), which are: 1) low cost identification and authentication; and 2) cryptographic key generation. The details of these two applications are explained in the subsequent sections.

Low-Cost Identification and Authentication

Identification and authentication are two essential processes in any security field. Identification occurs when a subject claims an identity, such as a username, a process ID,

or a smart card, which can uniquely identify a subject. Authentication is the process of proving an identity by providing appropriate credentials. It is critical for an IC to be identified and authenticated securely for using an IC application to process sensitive and user specific data. As the PUF output is device-specific and unpredictable, it is suitable for use by a challenge-and-response protocol to identify and authenticate an IC. As explained in Section 1.5.1, a strong PUF has an exponentially large number of CRPs. Hence, it is the right type of PUF for a challenge-and-response protocol for IC identification and authentication.

Figure 1.22 illustrates the PUF-based authentication process. An enrolment phase takes place prior to the deployment of the authentic device in the field. During the enrolment phase, a trusted party utilizes a verifier to apply randomly chosen challenges to an authentic device. Subsequently, this process obtains unpredictable responses , known as CRPs, which are then stored in a designated database $j(d_j)$ for future identification and authentication. In the field, when device j, which contains a PUF, is requested for authentication, a verifier selects a challenge from d_j and obtains the PUF response from the device j. To avoid an exhaustive comparison with all CRP profiles in the database during the authentication process, the verifier performs a profile match to retrieve a d_j with which the response given by the device j is compared. The device j passes the authentication process if the response matches or is less than the HD threshold, ϵ,as compared to the stored value in the database. As the challenge-response protocol does not require an ECC by forgiving bit errors that are less than ϵ, it is extremely lightweight and suitable for resource-constrained devices, such as RFID tags. These extremely lightweight PUF-based RFID tags can be promoted as a secure alternative to memory-based RFID tags and have been proposed as an anti-counterfeit solution for medical drugs, supply chain control, and secure access Becker (2015).

Two types of errors could possibly occur in the PUF-based identification and authentication scheme, shown in Figure 1.22.

1. False negative - This occurs when the response of a valid PUF deviates significantly from its initial state stored in the database. It might be deemed to be a different PUF and might be rejected during an authentication process. The probability of this rejection is given as preject and can be computed using Lim

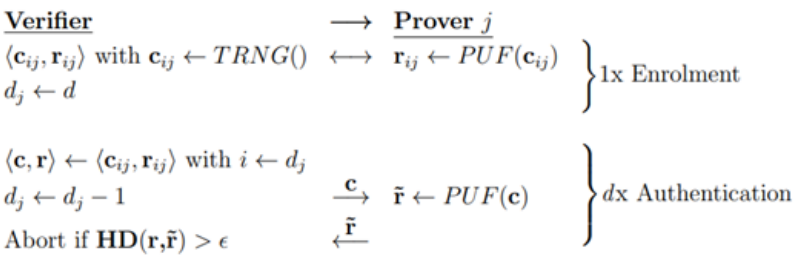


Figure 1.22: Low-cost PUF-based identification and authentication Suh and Devadas (2007), Delvaux et al. (2015)

et al. (2005), Suh and Devadas (2007):

$$Preject = 1 - \sum_{i=0}^{\epsilon} \binom{n}{i} p_{intra}^{i} (1 - p_{intra})^{n-i} \tag{1.1}$$

where Pintra denotes the bit error probability which is computed using Eq. (2.2), $\frac{Intra-HD}{100}$ n is the total number of bits in the response, and is the HD threshold

2. False positive - This occurs when a wrong PUF is issued to a server, and the server authenticates it by mistake. The probability of this misidentification is given as pmis and can be computed using Lim et al. (2005), Suh and Devadas (2007):

$$Pmis = \sum_{i=0}^{\epsilon} \binom{n}{i} p_{inter}^{i} (1 - p_{inter})^{n-i} \tag{1.2}$$

where Pinter denotes the bit unique probability which is computed using Eq. (2.1), $\frac{Intra-HD}{100}$ n is the total number of bits in the response, and e is the HD threshold.

Su et al. (2008) further described that the unreliable bits may have an impact on the probability of misidentification. Given two chips, k and l, a misidentification will occur when $HD(rl, \tilde{rk}) \leq HD(rk, \tilde{rk})$ due to the effect of the unreliable bits. Assume that all unreliable bits will always evaluate to the worst possible case such that:

$$HD(rl, \tilde{r}k) = HD(rk, rl) - HD(rk, \tilde{r}k) \tag{1.3}$$

The condition of misidentification can be re-written as:

$$HD(rl, rk) - HD(rk, \tilde{r}k) \leq HD(rk, \tilde{r}k) \tag{1.4}$$

$$HD(rl, rk) \leq 2 \times HD(rk, \tilde{r}k) \tag{1.5}$$

Therefore, the probability of misidentification is determined by the product between the probability of a particular HD and the misidentification occurring at that particular HD.

$$p'_{mis} = \sum_{i=0}^{2-\epsilon} \binom{n}{i} p_{inter}^{i} (1 - p_{inter})^{n-i} \left[1 - \sum_{i=0}^{\epsilon} \binom{2\epsilon}{i} p_{intra}^{i} (1 - p_{intra})^{2\epsilon - i} \right] \tag{1.6}$$

Based on Equations (1.1) and (1.6), Figure 1.23 shows the probability of rejection and misidentification at different bit error rates for a 128-bit identifier, considering an ideal uniqueness level of 50%, under variations of ϵ. As can be seen from Figure 1.23, if ϵ is set too low, then the probability of authentic PUFs being rejected increases, while setting ϵ too high increases the probability of misidentification. The setting of ϵ further impacts the vulnerability to an ML-attack Hospodar et al. (2012).An adversary only needs to achieve at least $1 - \left(\frac{e}{n}\right) \times 100\%$ prediction accuracy. It is always desirable for a PUF to achieve low bit error rates to reduce the probability of rejection or misidentification and improve security.

The presented challenge-response protocol is targeted for cost effective applications.

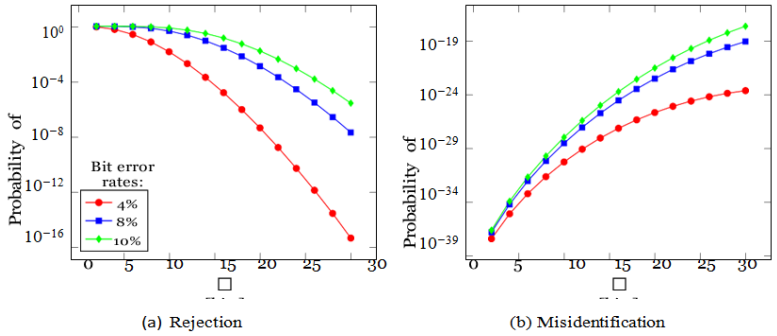


Figure 1.23: Probability of rejection and misidentification at different bit error rates and ϵ for n=128-bit and $p_{inter} = 0.5$

Consequently, when the verifier communicates with device j, the CRPs are sent without added protection, potentially allowing a person-in-the-middle to intercept the challenge and obtain the response from the PUF. Subsequently, these acquired CRPs could be utilized to spoof device j. Therefore, it is suggested in Suh and Devadas (2007) to avoid reusing the CRPs to prevent replay or person-in-the-middle attacks. In an extreme case, it is also possible that an attacker can get possession of device j and perform a non-invasive measurement of CRPs. A low- cost challenge-and-response protocol can only be realized with a strong assumption that strong PUFs are resilient against ML-attacks.

As mentioned earlier, a database is required to store the CRPs for a given PUF, which might pose a limitation in terms of the physical space overhead in the verifier. The idea of using a compact or mathematical model for the verifier to save space has been suggested but not described in detail Becker (2015), Gao et al. (2016a), Rostami et al. (2014). Nevertheless, as argued in Becker (2015), several key features of PUFs are lost if a compacted model is needed. For example, Strong PUFs are no longer resilient against model building and the verifier needs to be a trusted entity to maintain the integrity.

The study from Harishma et al. (2022), introduces a novel and secure key-exchange scheme designed for communication between Smart meters and Utility servers. The proposed protocol accommodates the resource constraints of the meter, incorporating an embedded Physical Unclonable Function (PUF) instance. Unlike traditional methods, the meter generates credentials dynamically during the protocol run, eliminating the need for credential storage. The protocol combines elements of identity-based cryptography and symmetric key cryptography to distribute computational overhead effectively between the server and the meter. The protocol is proven to be forward-privacy preserving and resilient against various attacks, including impersonation, man-in-the-middle, and replay attacks. To validate the effectiveness of the scheme, the researchers implemented an end-to-end Smart metering testbed using commercial-off-the-shelf products. Experimental results demonstrate that, despite significant resource disparities, the inherent asymmetric properties of the protocol ensure secure communication between the meter and the server.Furthermore, the paper addresses potential physics-based attacks, specifically load modification attacks, by presenting

a mitigation technique. The authors successfully integrated this countermeasure with the authenticated key-exchange protocol, emphasizing its effectiveness. The paper concludes with an experimental validation of the proposed techniques, providing a comprehensive demonstration of the protocol's security and practical utility in real-world scenarios.

Cryptographic Key Generation

Cryptographic key generation is another class of application for PUFs. Memory-based PUFs, such as SRAM-PUF, Butterfly-PUF and Buskeeper-PUF, are considered to be weak PUFs and are typically used for generating cryptographic keys Rührmair et al. (2010), Handschuh (2012), Zeitouni et al. (2015). However, outputs from the PUF circuit are known to be noisy due to environmental variations and aging Maiti and Schaumont (2013), rendering direct usage of the secret key for cryptographic primitives unfeasible. Security applications require every bit of a key to stay constant, which might not be possible at every instance Suh and Devadas (2007). Figure 1.24 shows a practical example of cryptographic key generation based on SRAM-PUF. An ECC with the helper data, h, is used to generate error free cryptographic keys. However, a partial information on the PUF response, k, could be recovered by the attacker because of the information obtained from the helper data. Consequently, privacy amplification is used to maximize the uncertainty of the generated keys Simons et al. (2012). Guajardo et al. (2007) suggested the use of a hash function as a privacy amplification module in such scenarios.

The procedure of cryptographic key generation is divided into two phases: Enrolment and Reconstruction. Enrolment is a one time event wherein a new key is generated or stored. During enrolment, the SRAM-PUF confronts a challenge in the form of an SRAM memory address range denoted as w, resulting in the extraction of stable unique values (SUVs) termed as y. During this phase, an ECC receives k, which is a subset of the SUVs y, and generates a code word n; k is then fed into the privacy amplification module to generate a key and the helper data, h, is computed as $n \oplus y$ At the end of this phase, the generated helper data, h, and the memory address range, w, are stored in the memory. In the Reconstruction phase, the same PUF is measured again (i.e., the noisy response y^J and a noisy code word n^J is computed as $y^J \oplus h$.

An ECC is used to correct n^J and the key programmed during the Enrolment phase is then recovered after privacy amplification.

Unfortunately, the implementation of ECC requires significant area overheads, which

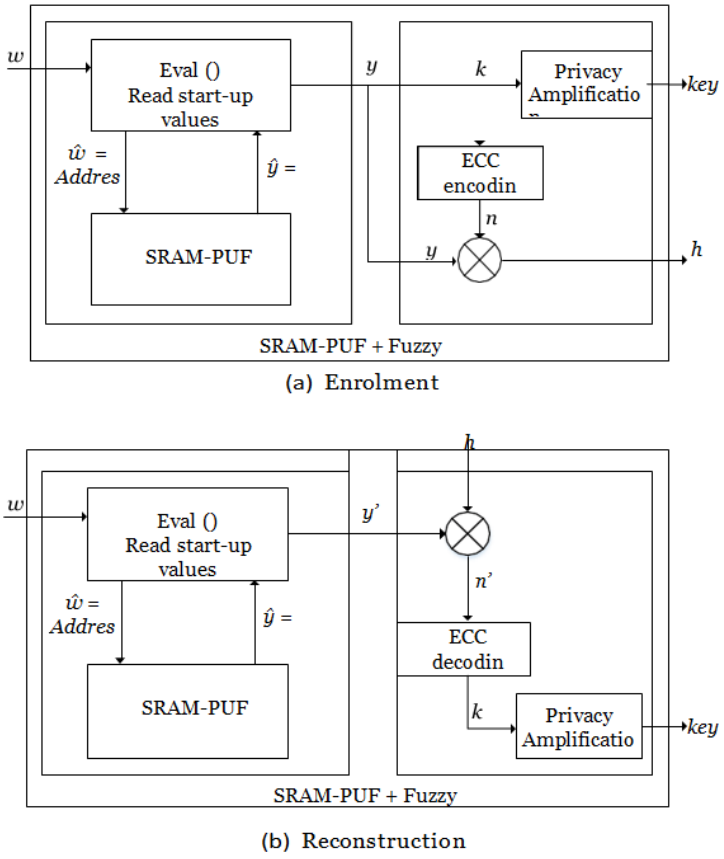


Figure 1.24: The procedure of cryptographic key generation based on SRAM- PUF Simons et al. (2012)

increase with an increase in the bit error rate Bhargava and Mai (2014), Lao et al. (2016). The increase in the area overhead can hinder the widespread adoption of PUFs as a lightweight security entity. Notably, while bit errors can be induced by temperature and supply voltage fluctuations, errors arising from aging are more severe and prolonged.

The study shown in IntrinsicID et al. (2015) demonstrates the feasibility of generating secure keys by employing the code-offset method and biased Physical Unclonable Functions (PUFs). Importantly, this approach ensures the integrity of the secret key without compromising its security, irrespective of the degree of bias in the PUF. The

conventional methods, on the other hand, are prone to revealing or losing the secret key when there is a significant reduction in entropy within the PUF.

Based on the proposed usage of PUFs, other potential applications of PUFs, such as IP protection in FPGAs Guajardo et al. (2007), Kumar et al. (2008), remote activation of ICs Alkabani et al. (2007), hardware- software binding Schaller et al. (2014), Kohnhäuser et al. (2015), and hardware obfuscation Wendt and Potkonjak (2014), have been suggested. Presently, PUF based hardware security has transitioned from academic research labs into the field of next-generation security products. For example, NXP uses PUF as a secure key storage to improve the security level in their upcoming products Semiconductors (2016). IntrinsicID's Apollo system ID (2016) integrates a butterfly Physical Unclonable Function (PUF) with Intrinsic ID's helper data algorithms. This involves configuring butterfly-shaped circuits on the fabric of an FPGA (Field-Programmable Gate Array) to inherently generate the necessary entropy for a robust hardware root of trust. The keys derived from Apollo are volatile, generated only when needed, providing a high level of security assurance.Apollo is considered a "soft PUF" implementation because it is part of the FPGA configuration file. This design allows for the retrofitting of security functionality on deployed devices, enabling remote installation of a hardware root of trust even in existing systems ("brownfield" installations). This flexibility in implementation contributes to the adaptability and security enhancement of deployed devices.

1.8 Known Attacks to PUFs

As suggested by the name, a "Physical Unclonable Function", in which the most important feature is "Unclonable", is considered as a promising solution to deal with insecure key storage in NVM, hardware fingerprinting, and several other security issues. However, PUFs are threatened because many different successful attacks have revealed vulnerabilities in certain silicon PUFs. These attack techniques include invasive, semi invasive, and non-invasive. Invasive attacks alter the physical properties of the chip irreversibly. Common techniques for invasive attacks include micro-probing. Scanning Electron Microscope (SEM), and Focus Ion Beam (FIB) require a sample preparation, such as decapsulation and depassivation. In contrast, non-invasive attacks require no sample preparation, exploiting external data, such as input and

output values, running time, power consumption, etc. Semi-invasive attacks mandate partial or complete removal of the device packaging. Unlike invasive attacks, destructive modifications are not required in semi-invasive attacks. Recent attacks on PUFs are discussed next and categorized based on the attack technique.

1.8.1 Invasive Attacks

Helfmeier et al. (2013) successfully cloned an SRAM-PUF using an FIB circuit edit technique. The cloning process was performed on SRAM memory within the ATmega328P micro controller, with a feature size of approximately 600-nm. Following the removal of the device's package and excess bulk silicon from the backside, Photonic Emission Analysis (PEA) was deployed to capture extremely weak photo emissions from switching transistors during the power-up process. Based on the captured emission image, an FIB circuit edit was conducted to alter the transistor characteristics according to the fingerprint of the target device. Although this shows that an SRAM-PUF could be cloned, producing a physical clone with these techniques remains economically infeasible for devices with limited financial value. Besides, modern ICs with small feature sizes require complex and expensive PEA techniques Schlösser et al. (2012). In another study, an invasive attack on an SRAM-PUF through device aging was analyzed and evaluated Roelke and Stan (2016). The fingerprint generated from an SRAM-PUF could be erased through a device aging process, hence making it susceptible to a denial of service (DOS) attack.

1.8.1.1 Semi-invasive Attacks

A semi-invasive attack was proposed in Nedospasov et al. (2013), wherein a laser stimulation (LS) technique was used to retrieve the SUVs of SRAM-PUFs. The experiment was conducted on SRAM memory in AtMega328P and ATXMEga128A1 micro-controllers, with feature sizes of around 600-nm and 300-nm, respectively. The package and excess bulk silicon of the device backside were removed prior to the LS process. The cloning of the PUF was continued further as in Helfmeier et al. (2013) using an FIB circuit edit method. However, only the LS technique, capable of functioning down to a 180-nm technology node, was primarily explored in this context Nedospasov et al. (2013).

Tajik et al. (2014) used a PEA technique to physically characterize the Arbiter-PUF and extract its delay parameters. The 8-bit Arbiter-PUF was implemented on a Complex Programmable Logic Device (CPLD) manufactured in a 180-nm technology. Prior to the PEA process, the device was decapsulated and the bulk silicon material of the device was thinned down. The delay characterization of the Arbiter-PUF was based on the time difference measured between enabling the PUF and photon emission at the outputs of the last stage (i.e., before the arbiter circuit) in which the challenge 00000000 was chosen as a reference measurement. Subsequently, each challenge bit was flipped (HD=1) and the procedure was repeated. Finally, the delay for each stage was revealed by subtracting the delay of the reference measurement and the delay of each challenge with HD=1. As the overall delay at the outputs of the last stage was the sum of the delay in each stage, the measured delay parameters could be further used to compute the responses for arbitrary challenges.

1.8.2 Non-invasive Attacks

Non invasive attacks is another attack model on a PUF. This particular approach is considered highly feasible due to its cost-effectiveness, requiring the attacker only to access the device's interface Gassend et al. (2002). Consequently, an attacker is restricted to non-invasive CRPs measurement and can apply a polynomial number of challenges to the device to collect the corresponding responses. Armed with the measured CRPs of a particular PUF, the adversary tries to build a numerical model of the PUF. Hence, a non-invasive attack is also known as a model building attack. Several techniques have been proposed in the literature to perform model-building attacks, such as ML Rührmair and Sölter (2014), side-channel Delvaux and Verbauwhede (2014), Zeitouni et al. (2015), and hybrid (combination of side-channel and ML) Rührmair et al. (2014). The work shown in Fraunhofer institute Merli et al. (2011) infers that EM emissions will lead to information leakage , irrespective of the type of PUF construction. The research in Fraunhofer institute Merli et al. (2011) demonstrates that common Physical Unclonable Function (PUF) architectures are vulnerable to side-channel attacks. Arbiter PUFs, by their nature, leak information through electromagnetic (EM) emissions, while Ring Oscillator (RO) PUFs not only exhibit EM emission leakage but also provide attackers with insights into oscillator frequencies and running counters. Although optical systems are not directly susceptible to side-channel analysis, potential vulnerabilities in their sensors should be considered

during both the design and implementation of PUF architectures. Furthermore, the study identifies two successful attacks against fuzzy extractors: one targeting Toeplitz Hashing (TH-SCA) and another against the code-offset construction (CO-SCA), a foundation for many proposed fuzzy extractors. The implementation of the TH-SCA attack on an FPGA fuzzy extractor prototype underscores the tangible risk of side-channel analysis for systems generating cryptographic keys using fuzzy extractors. This threat extends beyond FPGAs to ASIC and likely software implementations of fuzzy extractors. Consequently, both Weak PUFs and Strong PUFs, particularly when utilized with a fuzzy extractor, are not inherently resistant to side-channel analysis. This highlights the importance of considering and addressing potential side-channel leakage during the design and implementation phases of PUF-based systems.

An ML based attack is the most applicable to strong PUFs Rührmair and Sölter (2014) and it was first demonstrated in Lim (2004) to predict the response of an Arbiter-PUF by using a Support Vector Machine (SVM). As mentioned in Section 1.6.1, the delay parameters of switching components in an Arbiter-PUF can be modeled with an additive linear model. Hence, an Arbiter-PUF can be predicted with high accuracy using an SVM, as discussed in Lim (2004). As counter measures, the Feed-forward Arbiter-PUF Lim et al. (2005), XOR Arbiter-PUF Suh and Devadas (2007), and Lightweight-PUFs Majzoobi et al. (2008) were proposed to introduce non-linearity into the Arbiter-PUF. Further, Rührmair and Sölter (2014) performed a comprehensive ML based attack using SVM, Logistic Regression (LR), and Evolution Strategies (ES) on the aforementioned PUFs. Their initial analysis was based on simulated data, later validated by measurements from FPGA and ASIC, confirming the high accuracy of ML techniques in modeling the Feed-Forward Arbiter-PUF, XOR Arbiter-PUF, and Lightweight PUFs Rührmair and Sölter (2014).

In a recent study, Becker (2015), shows that an attacker having access to the primary inputs of a PUF-based RFID tag could collect CRPs through a non-invasive measurement and could perform an ML-based attack. The parameters derived by the ML-based attack was built into the software on a programmable RFID smart-card emulator and used to clone a PUF tag successfully. However, prior to the ML-based attack, software reverse engineering was performed to discover the configuration of the PUF model in the reader, such as the exact linear feedback shift register (LFSR) and

XORing functions. As the internal configurations were known, the ML-based attack was performed based on the mapping of the internal CRPs by using LR.

ML based techniques, such as SVM, LR and ES, were used to perform an ML-attack on an Arbiter-PUF and its derivative. In another study, Hospodar et al. (2012) compared an ML-based attack performance between ANN and SVM on Arbiter-PUF and XOR Arbiter-PUF and it was found that ANN outperformed SVM. Vijayakumar et al. (2016) explored the learnability of ML techniques, such as SVM and LR, concerning the increasing non linearity in PUFs. Bagging and Boosting techniques were used for the ML-based attack analysis Vijayakumar et al. (2016), since these techniques have the potential to generate a strong classifier by combining predictions from several classifiers. Based on previous ML-based attack analyses Lim et al. (2005), Becker (2015), Hospodar et al. (2012), Rührmair and Sölter (2014), Vijayakumar et al. (2016), ANN, LR, ES, and Boosting are found to be the most favorable to solve non-linearity problems.

Study fromGanji et al. (2016) investigates the susceptibility of Physical Unclonable Functions (PUFs) to non-invasive Machine Learning (ML) attacks, particularly addressing the challenge when the underlying mathematical model is unknown. Introducing a provable framework based on Probably Approximately Correct (PAC) learning, the study focuses on the Bistable Ring PUF (BR-PUF) family, renowned for its undisclosed mathematical model. Through extensive experiments on BR-PUFs implemented on Field-Programmable Gate Arrays (FPGAs), the paper empirically validates the effectiveness of the proposed ML algorithm. Notably, the research demonstrates that PUFs, specifically the BR-PUF family, exhibit challenge bit positions with more significant influences on responses than other bits, challenging the assumption of equal determination by each bit. The boosting algorithm results reveal that the accuracy of the ML algorithm improves with iterations, reinforcing the robustness of the approach. The paper underscores the importance of a precise mathematical description of PUF characteristics, aiming to bridge the gap between the mathematical design of cryptographic primitives and the practical design of PUFs. Additionally, the authors suggest that insights from well-known theorems and analyses in modern cryptography, such as the Siegenthaler Theorem and Fourier analysis, could inform the physical design of secure PUFs in the future.

Delvaux and Verbauwhede (2014) exploited repeatability imperfections in Arbiter-PUF responses, stemming from the CMOS device noise, as a side-channel information to perform a model-building attack. The key insight is that repeatability measurements provided direct timing information. If the delay difference, Δt, for a given challenge is very large, then it is unlikely that the noise changes the sign of Δt. In contrast, if Δt is close to zero (entering the metastability state, as discussed in Section 1.6.1), the response of the Arbiter-PUF would be influenced by the noise, resulting in changes of the Δt sign. Although the timing information is relative, Delvaux and Verbauwhede (2014) successfully built the model of Arbiter-PUF with a prediction accuracy of over 85%.

Zeitouni et al. (2015) exploited remanence decay in volatile memory as a side-channel technique to attack SRAM-PUFs, which were integrated into 65-nm CMOS technology. The approach in this attack was to recover the PUF response in a device after overwriting the SRAM-PUF with data known to the adversary. In this study Zeitouni et al. (2015), the adversary's capability was assumed to initialize memory with known values and their awareness that the targeted PUF-based system utilized a Message Authentication Code (MAC). Leveraging these assumptions, the adversary could recover the PUF state by examining the encryption in a series of data remanence experiments. Although the secret key was successfully recovered, this attack is likely to be impractical in terms of the timescale, as it takes approximately two CPU-months. Besides, this attack is limited by the precision of the equipment to control the remanence decay in the SRAM.

Furthermore, a hybrid technique was proposed wherein side-channel information and ML techniques were combined Rührmair et al. (2014). As the CRP mapping complexity of strong PUFs increases, the ML technique reaches its limit (i.e., the training time increases exponentially as the number of XORs increase) when applied to Lightweight PUFs or XOR Arbiter-PUFs with a bit-length of 256 or more and with 6 or more XORs Rührmair and Sölter (2014). Ru¨hrmair et al. exploited the power and timing traces of an XOR Arbiter-PUF and a Lightweight-PUF as side-channel information to overcome the limitations in ML techniques Rührmair et al. (2014). Based on the power and timing side-channel information, the cumulative number of zeros and ones in the outputs of the XOR Arbiter-PUF and Lightweight-PUF before the XOR gate

was estimated. Subsequently, the adapted ML technique was used to exploit this information and successfully attack the XOR Arbiter-PUFs and Lightweight PUFs for up to 16 XORs and for a bit-length of up to 512 (timing side-channel) and 128 (power side-channel) with a high accuracy and polynomial training time.

In another studySahoo et al. (2017). It introduces a novel solution to enhance the security and reliability of Arbiter Physically Unclonable Functions (APUFs). Research paper Sahoo et al. (2017)proposes a Multiplexer-based APUF Composition (MPUF) and introduces two variants, cMPUF and rMPUF, addressing challenges such as vulnerability to modeling attacks and reliability issues. The rMPUF, in particular, demonstrates improved resilience against reliability-based modeling attacks compared to XOR APUFs. The key advantage of MPUF compositions is their ability to achieve higher reliability than practical XOR APUFs. Experimental results validate these findings, showcasing the potential of the proposed rMPUF as a secure and reliable alternative to XOR APUFs in practical applications.

1.8.3 Challenges

Thus far, the recent attacks on PUF including invasive, semi-invasive, and non-invasive attacks were discussed, as summarized in Table 1.2. One of the PUF core properties is its resistance to precise cloning due to inherent random process variations that cannot be replicated atom-by-atom using current fabrication technologies Rührmair et al. (2014). Consequently, generating a perfect clone is still infeasible. Nevertheless, as discussed in Section 1.6.3, an SRAM-PUF can be physically cloned by editing the circuit using a FIB method. To the best of our knowledge, none other PUFs architectures have been physically cloned using the same method.

As detailed in Table 1.2, invasive and semi-invasive attacks require complex techniques to perform an attack, such as PEA, FIB, LS, and burn-in. These techniques have a relatively higher cost compared to ML techniques used to perform non-invasive attacks. Nevertheless, if sufficient financial and technical investments are offered, an obvious threat to the PUF system is that its function is clonable by using one of the attack types mentioned in Table 1.2. An intense competition between code-makers and code-breakers in the area of strong PUFs and weak PUFs is expected in the near future Rührmair and Sölter (2014). Nevertheless, the proposal of various attack

methods serves to validate the quality of a PUF and motivates people to look for countermeasures in overcoming the weakness of that particular PUF. It is expected that at some point, this competition will converge to solutions that are resilient against the known attacks.

Table 1.2: Summary of known attacks to PUFs

Attack Type	PUF Type	Platform	Technology	Technique
Invasive	SRAM -PUF Helfmeier et al. (2013)	ATmega328P	600-nm	PEA and FIB
	SRAM-PUF Roelke and Stan (2016)	AS6C6264 SRAM IC	NR	Burn-in
Semi-invasive	SRAM-PUF Nedospasov et al. (2013)	AtMega328P ATXMEga 128A1	600-nm 300-nm	LS
	Arbiter-PUF Tajik et al. (2014)	Altera Max V CPLD	180-nm	PEA
Non-invasive	Arbiter-PUF Lim (2004)	ASIC	180-nm	SVM
	5-XOR Arbiter- PUF Rührmair and Sölter (2014)	ASIC	45-nm	LR
	Lightweight -PUF (5 XORs) Rührmair and Sölter (2014)	Simulation	NR	LR
	Feed-Forward Arbiter-PUF Rührmair and Sölter (2014)	Simulation	NR	ES
	Arbiter-PUF Becker (2015)	PUF-based RFID tag	NR	LR
	Arbiter-PUF Delvaux and Verbauwhede (2014)	ASIC	65-nm	Side-channel
	SRAM-PUF Zeitouni et al. (2015)	ASIC	65-nm	Side-channel
	16-XOR Arbiter -PUF Rührmair et al. (2014)	Xilinx Spartan-6 FPGA	NR	Side-channel and LR
	Lightweight- PUF (16 XORs) Rührmair et al. (2014)	Xilinx Spartan-6 FPGA	NR	Side-channel and LR

1.9 Patents on PUF by notable semiconductor organizations

- **Taiwan Semiconductor Manufacturing Co.:**
  - Physically unclonable function (PUF) generation: US10965475B2. This patent describes a method and apparatus for generating a PUF value based on the variations of the threshold voltage of a transistor. The PUF value can be used for device identification, authentication, or encryption.
  - I/O circuit design for SRAM-based PUF: US10972292B2. This patent describes a circuit design for an SRAM-based PUF that can reduce the power consumption and improve the reliability of the PUF value. The circuit design includes a pre-charge circuit, a sense amplifier, and a latch circuit.
  - PUF method and structure: US20210409233A1. This patent describes a PUF method and structure that can generate a PUF value based on the variations of the resistance of a resistor. The PUF method and structure can be implemented in a memory device, such as a resistive random-access memory (RRAM) device.
- **Intel Corporation:**
  - System and methods for PUF-based authentication in resourced constrained environments: US10354280B2. This patent describes a system and methods for PUF-based authentication in resource-constrained environments, such as the Internet of Things (IoT) devices. The system and methods use a PUF circuit that can generate a PUF value based on the variations of the capacitance of a capacitor. The PUF value can be used for device authentication, key generation, or key exchange.
  - PUF-based device authentication and key generation: US20190343128A1. This patent describes a PUF-based device authentication and key generation method that can generate a PUF value based on the variations of the delay of a ring oscillator. The PUF value can be used for device authentication, key generation, or key exchange.
  - PUF-based secure key generation and storage: US20190343129A1. This patent describes a PUF-based secure key generation and storage method

that can generate a PUF value based on the variations of the current of a current mirror. The PUF value can be used for key generation, key storage, or key retrieval.

- **NXP Semiconductors:**
  - Physical Unclonable Function (PUF) security key generation: US10965476B2. This patent describes a PUF security key generation method that can generate a PUF value based on the variations of the voltage of a voltage divider. The PUF value can be used for key generation, key storage, or key retrieval.
  - PUF-based device authentication and key exchange: US20190343127A1. This patent describes a PUF-based device authentication and key exchange method that can generate a PUF value based on the variations of the frequency of a voltage-controlled oscillator. The PUF value can be used for device authentication, key generation, or key exchange.
  - PUF-based secure key storage and retrieval: US20190343130A1. This patent describes a PUF-based secure key storage and retrieval method that can generate a PUF value based on the variations of the phase of a phase-locked loop. The PUF value can be used for key storage, key retrieval, or key encryption.
- **Samsung Electronics:**
  - PUF circuit and method of generating PUF value: US10965477B2. This patent describes a PUF circuit and method of generating a PUF value based on the variations of the resistance of a metal-oxide-semiconductor field-effect transistor (MOSFET). The PUF circuit and method can reduce the influence of environmental factors, such as temperature and voltage, on the PUF value.
  - PUF circuit and method of operating the same: US10965478B2. This patent describes a PUF circuit and method of operating the same that can generate a PUF value based on the variations of the resistance of a MOSFET. The PUF circuit and method can improve the stability and reliability of the PUF value by using a feedback loop and a calibration circuit.

- PUF circuit and method of generating PUF value using the same: US10965479B2. This patent describes a PUF circuit and method of generating a PUF value using the same that can generate a PUF value based on the variations of the resistance of a MOSFET. The PUF circuit and method can enhance the security and robustness of the PUF value by using a masking circuit and a scrambling circuit.

- **Micron Technology:**

  - PUF-based memory device authentication: US10965480B2. This patent describes a PUF-based memory device authentication method that can generate a PUF value based on the variations of the data retention time of a memory cell. The PUF value can be used for memory device authentication, key generation, or key exchange.
  - PUF-based memory device authentication and key generation: US10965481B2. This patent describes a PUF-based memory device authentication and key generation method that can generate a PUF value based on the variations of the data retention time of a memory cell. The PUF value can be used for memory device authentication, key generation, or key encryption.
  - PUF-based memory device authentication and key exchange: US10965482B2. This patent describes a PUF-based memory device authentication and key exchange method that can generate a PUF value based on the variations of the data retention time of a memory cell. The PUF value can be used for memory device authentication, key exchange, or key decryption.

- **SK Hynix:**

  - PUF circuit and method of generating PUF value: US10965483B2. This patent describes a PUF circuit and method of generating a PUF value based on the variations of the resistance of a resistor. The PUF circuit and method can reduce the power consumption and improve the reliability of the PUF value.
  - PUF circuit and method of operating the same: US10965484B2. This patent describes a PUF circuit and method of operating the same that can generate a PUF value based on the variations of the resistance of a resistor.

The PUF circuit and method can improve the stability and robustness of the PUF value by using a feedback loop and a calibration circuit.

- PUF circuit and method of generating PUF value using the same: US10965485B2. This patent describes a PUF circuit and method of generating a PUF value using the same that can generate a PUF value based on the variations of the resistance of a resistor. The PUF circuit and method can enhance the security and efficiency of the PUF value by using a masking circuit and a compression circuit.

- **Qualcomm Incorporated:**

  - PUF-based device authentication and key generation: US10965486B2. This patent describes a PUF-based device authentication and key generation method that can generate a PUF value based on the variations of the charge of a capacitor. The PUF value can be used for device authentication, key generation, or key exchange.
  - PUF-based secure key storage and retrieval: US10965487B2. This patent describes a PUF-based secure key storage and retrieval method that can generate a PUF value based on the variations of the charge of a capacitor. The PUF value can be used for key storage, key retrieval, or key encryption.
  - PUF-based device identification and authentication: US10965488B2. This patent describes a PUF-based device identification and authentication method that can generate a PUF value based on the variations of the charge of a capacitor. The PUF value can be used for device identification, device authentication, or device registration.

- **Broadcom Corporation:**

  - PUF circuit and method of generating PUF value: US10965489B2. This patent describes a PUF circuit and method of generating a PUF value based on the variations of the resistance of a resistor. The PUF circuit and method can reduce the power consumption and improve the reliability of the PUF value.
  - PUF circuit and method of operating the same: US10965490B2. This patent describes a PUF circuit and method of operating the same that can

generate a PUF value based on the variations of the resistance of a resistor. The PUF circuit and method can improve the stability and robustness of the PUF value by using a feedback loop and a calibration circuit.

- PUF circuit and method of generating PUF value using the same: US10965491B2. This patent describes a PUF circuit and method of generating a PUF value using the same that can generate a PUF value based on the variations of the resistance of a resistor. The PUF circuit and method can enhance the security and efficiency of the PUF value by using a masking circuit and a compression circuit.

- **Texas Instruments Incorporated:**

  - PUF-based device authentication and key generation: US10965492B2. This patent describes a PUF-based device authentication and key generation method that can generate a PUF value based on the variations of the delay of a delay line. The PUF value can be used for device authentication, key generation, or key exchange.
  - PUF-based secure key storage and retrieval: US10965493B2. This patent describes a PUF-based secure key storage and retrieval method that can generate a PUF value based on the variations of the delay of a delay line. The PUF value can be used for key storage, key retrieval, or key encryption.
  - PUF-based device identification and authentication: US10965494B2. This patent describes a PUF-based device identification and authentication method that can generate a PUF value based on the variations of the delay of a delay line. The PUF value can be used for device identification, device authentication, or device registration.

1.10 Scope of the Research

The discussion in the previous section on the applications of Physical Unclonable Function (PUF) circuits highlights their versatile features, expanding possibilities for research in identifying the distinctive characteristics of these circuits.

- To select a PUF,

- Design a PUF and
- Study the aspects related to factors effecting its functionality
- Arriving at the optimal solution to bring them to a use-case.

Various areas worth considering are discussed in the following sections as a scope of research.

1.10.1 Selection of PUF

The unclonable nature of a circuit or device, often defined by factors, such as process, voltage, and temperature, plays a crucial role in establishing the unique and reliable signature or identification (ID) of the device Maiti (2012). Despite the availability of various PUF types, identifying a robust PUF has remained a challenge since its inception Maiti and Schaumont (2011). While current literature does not explicitly guide the creation of a strong PUF, it does elucidate the factors that can be controlled to enhance the efficiency of PUFs in generating responses Suh and Devadas (2007), Bernard et al. (2012), Maiti and Schaumont (2011), Xin et al. (2011) The versatility of FPGA based delay PUF circuits is well justified in literature Bernard et al. (2012). FPGA's reconfigurable architecture helps in implementing a dynamic hardware, specifically catering to state of the art IoT applications Stanciu et al. (2016). It is worth noting that most PUF-related work thus far has been extensively proven and practiced on FPGA technologies Kodỳtek and Lórencz (2015), Suh and Devadas (2007).

1.10.2 Modelling of PUF

PUFs do not address the narrow space between advancing device technology and the PUFs' ability to generate a strong Challenge and Response Pairs. There is a crucial need to understand how PUFs can be modeled as efficient security macros, considering the factors influencing a PUF's ability to generate a stable response. The scope of research presented in this thesis involves only FPGA friendly PUFs with controllable delay parameters Bernard et al. (2012) and aims to identify feasible implementation platforms and more cost effective methods with highly optimized hardware. The literature Maiti and Schaumont (2011) highlights Ring Oscillator based PUF circuits as well established to achieve a strong PUF Herder et al. (2014) by introducing the

notion of Configurable Ring Oscillators Kodỳtek and Lórencz (2015). The configurable inputs has added strength to PUFs while optimizing the space and resources available within the FPGA design. However, the approach towards exploring efficient ways to use the FPGA resources on chip and to model a PUF circuit is yet to be explored.

1.10.3 Generating and enrolling of PUF challenge response pairs

The FPGA fabric comprises two key aspects that govern the generation and stabilization of a reliable response for a PUF: process parameter variation and the interconnect matrix. Process variation predominantly influences the creation of the unique CRP for the device, aiming to reduce the impact of randomness caused by interconnects within the FPGA fabric Gassend et al. (2002). This helps in generating a unique ID rather than an unpredictable random response bit out of the PUF device.

A study by Stanciu et al. (2016) shows an approach towards generating and enrolling the PUF challenges while discussing the influence of other regular circuit activity on the PUF response. This underscores the importance of ensuring that the generation and enrollment logic on the PUF circuit do not bias its response.

An experiment conducted by Majzoobi et al. (2010) identified the optimal number of flip-flops required to design a frequency meter, determining the optimum unit time pulse. This work helps to minimize biasing of the PUF response by employing minimal hardware on the PUF, while highlighting significant frequency differences among a set of ROs within a single CLB.

The scope of this research involves extracting process variations while neglecting interconnect influences to create consistent challenge and response pairs. The primary objective of the PUF circuit is to extract device-level process variations within the entire FPGA fabric, with respect to process variations within the interconnect matrix and the CLB network. In FPGA designs, the placement of logic into specific slices in CLBs is controllable, but manual routing through the interconnect matrix is not feasible using FPGA editor tools. Hence, the only controllable aspect in the extraction of device process parameter variations is within the CLB network, not the switch ma-

trix network. Consequently, the logic incorporated to explore these variations should effectively examine the device variations within the CLB network, as failure to do so could render the generated device signature susceptible to process randomness caused by the interconnect, resulting in inconsistent challenge and response pairs. However, constructing a hard macro of the logic allows for the locking of internal routing. Various methodologies and compensation techniques in the open literature contribute to generating a robust PUF with consistent CRPs, partially mitigating these issues. However, the literature does not address the potential bias in response bits generated by the remaining logic interacting with the PUF as a macro. Hence, a crucial aspect of this research is to ensure that the enrolling logic does not influence the response bit and is instead shifted from the programmable logic to an on-chip or off-chip processor.

Interconnect delays significantly influence process parameter variation in devices, resulting in the generation of a random number rather than a consistent response to a specific challenge in PUF circuits. Consequently, it is crucial to attribute process variation as the device's response, rather than interconnect wire delays within the switch matrix for enhancing performance metrics for a PUF circuit Suh and Devadas (2007). The orientation of the hard macro and the location of the CROs to be compared would have a major impact on generating the unique challenge response pair. The following section explains this process.

1.10.4 Evaluation of PUF with literature specified metrics

When it comes to performance of the PUF, the metrics are often confined to the field of its application, lacking standardization until Maiti et al. (2013) analyzed and defined them. Standardization of the metrics gave a scope of research towards working on particular metrics and quantizing the parameters that affect those metrics.

- Uniqueness
- Reliability
- Uniformity
- Bit aliasing
- Attack resistance (through modeling an attack)

Maiti and Schaumont (2011) contributed significantly by conducting an analysis to derive ideal metric values. This analysis was based on the metrics defined by Maiti et al. (2013). While the first four metrics are commonly examined in existing literature, analyzing attack resistance necessitated modeling an attack on the PUF and determining the prediction rate, constituting a primary focus of research. Authors, such as Sedcole and Cheung (2006), Rührmair et al. (2010), Guo et al. (2016)have provided insights into this attack modeling approach, guiding the current research in analyzing the performance of the proposed PUF. The study focuses on the analysis of two types of PUFs:

- The first is a ring oscillator PUF, employing a specific methodology to obtain a stable response for the first four metrics.

- The second is a proposed Arbiter PUF specifically examined for Attack resistance, given that Arbiter PUFs are susceptible to such attack modeling.

1.10.5 Application of PUF based authentication

The seamless operation of a Physical Unclonable Function (PUF) in server-client environments to authenticate transactions requires a deep understanding of the underlying physics within PUF circuits and the engineering involved to meet these demands. One of the main reasons behind portraying this work on a lower architectural standards in initial phases is that it ensures that if a designed PUF is capable to propagate unique response on a resource constrained environment like Xilinx's Spartan 3e, it is quite evident that it would be easier to generate a stabler response with higher end FPGA architectures with processor system and programmable logic based environment without any dearth of resources.

The design is validated through test environments involving hardware-in-loop simulations using an application programming interface (API). This implementation encompasses Hardware Description Language (HDL) hard-macros, coupled with a comprehensive understanding of the functional and architectural aspects of the FPGA to ensure the accuracy and reliability of the test results.

1.11 Organization of the thesis

The research is structured in the form of a thesis comprising five chapters to explore PUF concepts and applications comprehensively. The concept of Physical Unclonable Function (PUF) primarily signifies an FPGA-based delay PUF throughout the thesis unless otherwise specified. It commences with an abstract delineating the necessity for hardware security and the cutting-edge features required for IoT security. This overview provides a concise insight into the identified problems within the domain and introduces keywords relevant to the research.

Chapter-1 introduces the Physical Unclonable Function and the associated concepts. It provides an overview about various types of PUFs and specifically talks about the FPGA based delay PUFs. This chapter identifies the research problems based on the research gap analysis.

Chapter-2 discusses the challenges involved in PUF design and validation. The chapter includes a literature review, gap analysis, problem identification, and a streamlined method for arriving solutions with specified metrics of evaluation.

Chapter-3 discusses PUF modeling for stable responses. It details the design, placement, and packaging of PUF macros to mitigate random noise from external factors, enrollment processes without the influence of additional logic, communication interface between PUF and CPU to acquire the responses, post-processing of the response for generating the signature, performance metric analysis of obtained responses, and validation of the proposed methodology to generate and enroll the responses.

Chapter-4 proposes an arbiter PUF to optimize area utilization while sustaining PUF capabilities. The chapter covers details about PUF implementation on FPGA, attack modeling on the Arbiter PUF, and assessment of the predictability factor in terms of attack resistance. It examines the application of designed PUFs in securing IoT enabled smart meters, which includes a connected model for the designed PUF in authenticating the transactions in an IoT environment.

Chapter-5 concludes with the discussion on the research scope and implications.

This is followed by an Appendix containing information about the hardware used and the associated data sheets.

Chapter 2

Assessment of PUF Design Challenges

2.1 Literature Review

2.1.1 Conventional RO PUF design from the reference (Xin et al. (2011))

Physically Unclonable Function is a circuit that uses manufacturing process variations to generate a unique digital fingerprint. Literature offers a scope of available state of the art and efficient methods to design a PUF circuitry with both silicon and non silicon based approaches. It has been observed from Herder et al. (2014) that silicon based PUF circuits are gaining attention towards securing ASIC or FPGA devices, which are primarily used in IoT applications at the physical layer. Various types of PUF circuits are explored Xin et al. (2011) to ensure an efficient and secure function that can provide strong challenge and response pairs, considering the state of the art machine learning algorithms Herder et al. (2014). All linear function based PUF circuits are prone to attacks Herder et al. (2014) and could easily learn the challenge and response pattern, thereby compromising the device's security protocols.

A literature survey Herder et al. (2014) rendered a large spectrum of PUFs and categorized them based on the randomness and the underlying behind their functionality. However, this research is restricted to silicon based PUFs Maiti and Schaumont (2011)

. In silicon based PUFs, FPGA-based PUF circuits were chosen due to their proven efficiency in implementation, despite the strength of ASIC PUFs. The time-to-market factor significantly influences commercial product releases (Bernard et al. (2012). A study on various delay based PUF circuits (Herder et al. (2014) has identified that a configurable Ring Oscillator with customizability is promising for an efficient implementation, thereby making the PUF circuit strong Maiti and Schaumont (2011). Despite its difficulty in achieving symmetry over the available resources on FPGA, a Configurable Ring Oscillator Kodỳtek and Lórencz (2015) is still an efficient implementation. The symmetry supposed to achieve on FPGA devices is often influenced by the design engineer's skills and the expertise to achieve this symmetry is thus patented by many research agencies. The knowledge on achieving symmetry is never exposed. The primary objective is to design a configurable ring oscillator and the subsequent survey addresses the various factors and challenges in implementing this design. Based on the literature survey Kodỳtek and Lórencz (2015), a hard macro implementation of the available evaluation logic, combined with the PUF circuit, appears to create a stronger challenge response pair. Maiti and Schaumont (2011) introduced an improved version of RO PUF by Suh and Devadas (2007). It was developed with an objective to reduce overall area of its implementation while improving the utilization of Look up Tables (LUT), which are function generator blocks inside programmable logic on FPGAs.

Figure 2.2 shows the logic structure of a configurable ring oscillator. It is an improvement over conventional RO PUF, shown in Figure 2.1, accomplished by incorporating multiplexer logic between two inverters and by utilizing LUT G and LUT F of a SLICE.

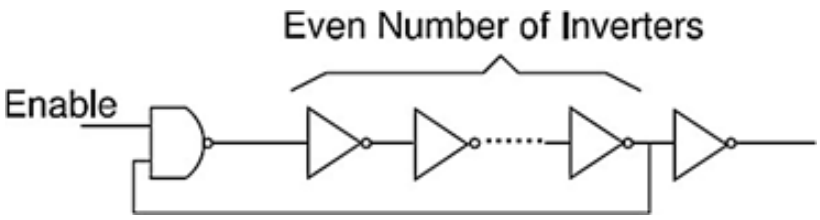


Figure 2.1: Basic Ring Oscillator

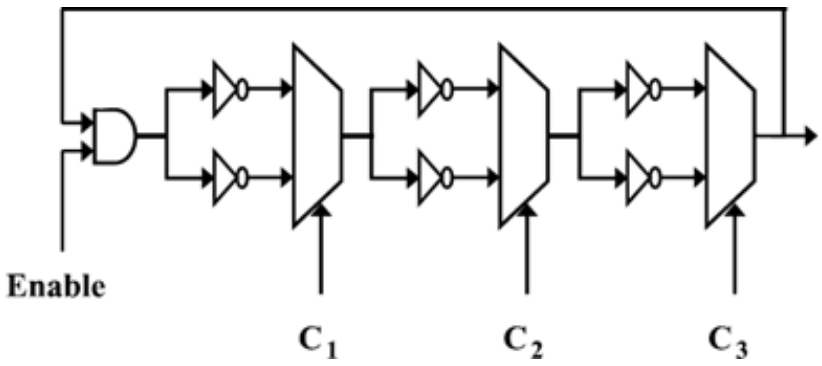


Figure 2.2: Configurable ring oscillator (Xin et al. (2011))

2.1.2 PUF enrollment phase

Prior to deployment of PUF over the target application, a PUF-based circuit's signature needs to be generated and stored in a secure server before it can be used. In a PUF based authentication system, this procedure is critical and is known as "enrollment". In order to create and store a relevant signature, enrollment challenges and responses should be fully controllable and observable by the participant. In order to protect the PUF's secret information, the CRPs must be stored in a secure database and secured access methods must be used to ensure that unauthorized persons cannot reuse this mechanism to build the same database as the one that contains CRPs. The PUF must meet the security criteria to ensure the development of reliable security services while in mission mode. Consequently, PUF-based security schemes must be developed using advanced and methodical techniques to deal with the following issues:

- How can the PUF's secret information (i.e., the set of CRPs) be safely extracted after fabrication ?
- What is the best way to store CRPs in the server's database?
- To what extent can PUF data extraction (in terms of time required to extract CRPs and storage space) be made to fit within the constraints of a budget?

An attacker could potentially use the device's mission mode to steal its physical properties, which could be used to create a database backup.

- Nevertheless, how can this be prevented? If CRPs naturally fluctuate over time, how can the database be securely updated in mission mode to account for this?

- How can the PUF's quality be checked after it has been manufactured?
- How can the PUF primitive's security properties be assessed in mission mode? These issues are remarkably similar to those faced by the testing community when evaluating traditional electronic gadgets. Methodical and well structured techniques have helped the testing community solve both offline and online test challenges, such as optimizing test time and coverage, and dealing with security issues during testing.

EDA tools or design techniques have been proposed to address these issues. Manufacturing and online testing for PUFs and security assessments of their properties are fundamentally different from conventional devices because the responses of PUFs are not monastically known during the designing stage. PUF enrollment and life-cycle management are addressed Suh and Devadas (2007). Literature survey enabled the identification of the problem about how the PUF data for enrollment is communicated and how an environment can be setup to secure the process of enrolment and understand the post processing of the response for signature generation.

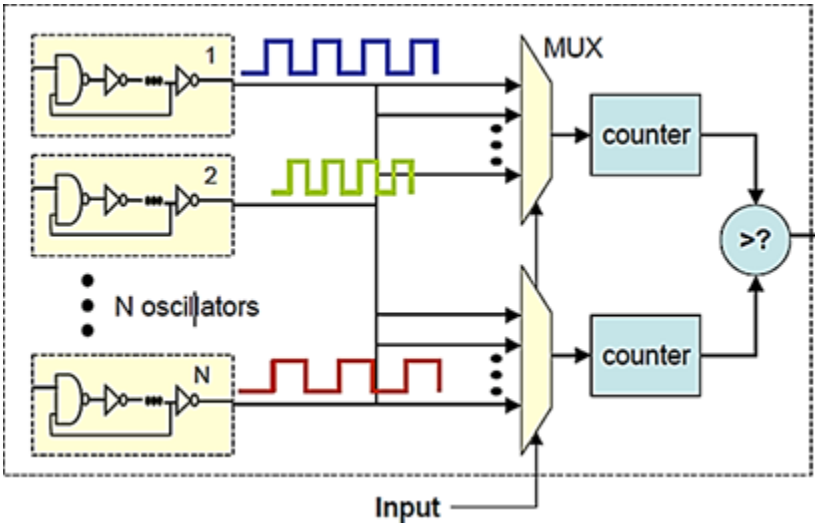


Figure 2.3: Ring oscillator based PUF - Evaluation logic circuit (Suh and Devadas (2007)

2.1.3 Post-processing response bits generated with PUFs

Suh and Devadas (2007) explains about how a stable response can be guaranteed by

propagating the variations in the parameters of PUFs' manufacturing process while separating systematic variations, such as static and dynamic timing characteristics. Figure 2.1 shows the basic architecture of a Ring Oscillator based PUF and Figure 2.3 explains the way response is generated from the PUF. Each comparison of a pair of oscillators generates a bit. There are N(N-1)/2 distinct pairs given N ROs. However, the entropy of this circuit, which corresponds to the number of independent bits it can generate, is clearly less than N(N-1)/2 because the bits obtained from pair-wise comparisons are correlated. For example, if RO1 is faster than RO2, the comparisons will yield 1. If RO2 is in turn faster than RO3, the comparison will yield a 1. It is clear that when RO1 is compared with RO3, it will yield a 1, as these bits are correlated. Fortunately, it is possible to derive the maximum entropy of this circuit by assuming pair wise comparisons. This method involves determining the number of independent bits generated by the circuit in relation to the variable N, which signifies the quantity of oscillators present. There are N! different orderings of ring oscillators based on their frequencies. If the orderings are equally likely, the entropy will be log2(N!) bits. For example, 35 oscillators can produce 133 bits, 128 oscillators can produce 716 bits, and 1024 oscillators can produce 8769 bits. Each oscillator can be used only once to avoid any correlation in the generated response. For example, 128 pairs of oscillators (total of 256) could be used to generate 128 bits. In addition to exploring this correlation, Herder et al. (2014) discussed environmental influences, such as voltage and temperature, examining their impact on compensation schemes in generating reliable bits.

Figure 2.4 shows the relationship between the frequency and location of Ring Oscillator on the fabric and the temperature in giving the probability of PUF output flip. Figure 2.4a shows the bit plot for frequencies extracted from nearby ring oscillators, while Figure 2.4b is from two distant oscillators. When comparing the frequencies of blue and green light at room temperature, it is anticipated that the blue frequency will be higher than that of the green light. It is observed that as temperature increases, both blue and green slow down; however, blue slows down faster than green. Suh and Devadas (2007) contributed the occurrence of the phenomenon of bit reversals to temperature effects. Thus, it is specified that by comparing ROs from distant frequencies, it becomes plausible to compensate for temperature effects and bit correlation effects. One such example is 1-out-of-k masking scheme proposed by the author.

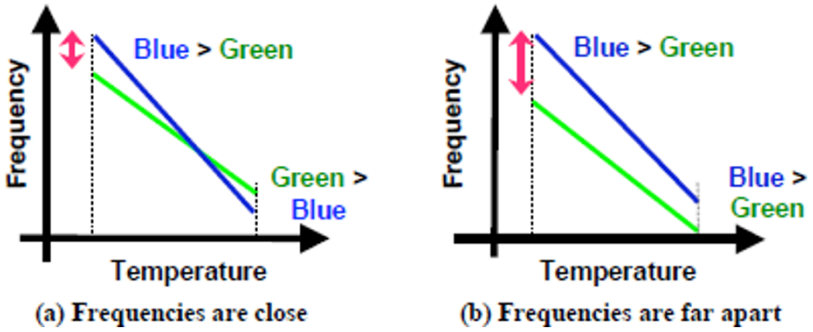


Figure 2.4: Temperature changes with respect to location of ROs Kodỳtek and Lórencz (2015)

With an expected N bit response, It generates the sequence of ROs k times larger than N, which gives (k x N) no of ROs and there is need to identify pair of ROs among k which has maximum frequency difference and use this as response generation pair(CRP).This method however has an over head of number of ROs to be deployed in generating a significant number of response bits. As far as FPGA specific implementations are concerned its quite obvious that FPGA resource utilisation plays a very important role, so this part of the literature survey gives a picture of PUF's reponse generation, factors effecting the reliability of signature generated out of response bits. The process of detecting and correcting or compensating the response from the PUF to make it more reliable,unique and uniform is called post processing of PUF response(Herder et al. (2014)).

2.1.4 PUF's Performance Characteristics (Metrics)

These Metrics decide the unclonable nature of PUF circuit and analyse how various environmental noise factors and other the adjacent digital circuit activity can influence the PUF responses. The PUF response Thad's generated has three dimensions on it.

- **Device dimension**
  If there are k number of chips generating the responses. that 'k' is the device dimension of the response it generates

- **Space dimension**
  If the response generated by 'k' no of chip is of 'n' bits wide. that 'n' is the space dimension of the response it generates.

- **Time dimension**

  If the response is generated on 'k' no of chips as 'n' bit wide responses for 'm' no of times. then m is the time dimension of the response it generates.

Thus response bits that are generated on k chip , with n bits wide and m no of times will have three different dimensions to measure its metrics. If we analyse some random one bit on that n-bit response it can be expressed as

$$Response_{i,l,t} = t_{th} \text{ sample of } l_{th} \text{ response bit of } i_{th} \text{ chip} \tag{2.1}$$

where $1 \leq i \leq k; 1 \leq l \leq n; 1 \leq t \leq n$

Maiti et al. (2013) summarized the standard performance metrics with that used by other authors and came up with standardized metrics. Since the original of PUFs, many authors have defined their own parameters to measure the PUFs' security. Average hamming distance is a quality factor that decides the PUFs' response evaluation. Table 2.1 shows the performance metrics established by various authors for determining the performance characteristics of a PUF.

Table 2.1: Different PUF parameters Maiti et al. (2013)

Author	**Performance Metrics**
Hori et al.	Randomness
	Steadiness
	Correctness
	Diffuseness
	Uniqueness
Maiti et al.	Uniformity
	Bit-aliasing
	Uniqueness
	Reliabiltity
Su et al.	Probability of misidentification
Majzoobi et al	Single bit probability
	Conditional Probability
Yamamoto et al.	Variety

Maiti et al. (2013) analyzed these parameters in a three dimensional space, as shown in Figure 2.5, constituting device, space and time. Maiti's analysis on these

parameters by different authors finally converges to the following metrics space, as shown in Figure 2.6.

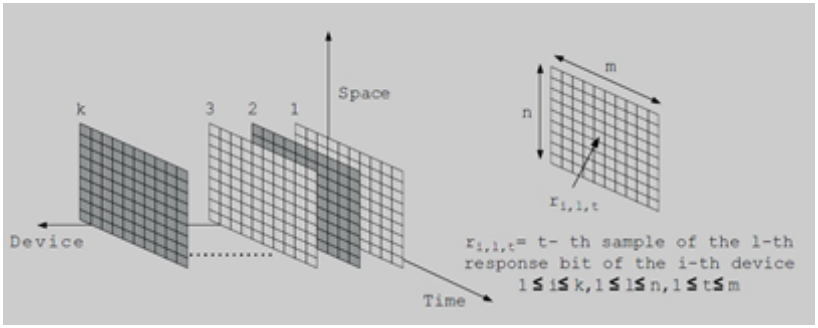


Figure 2.5: Dimensions of PUF Measurement Herder et al. (2014)

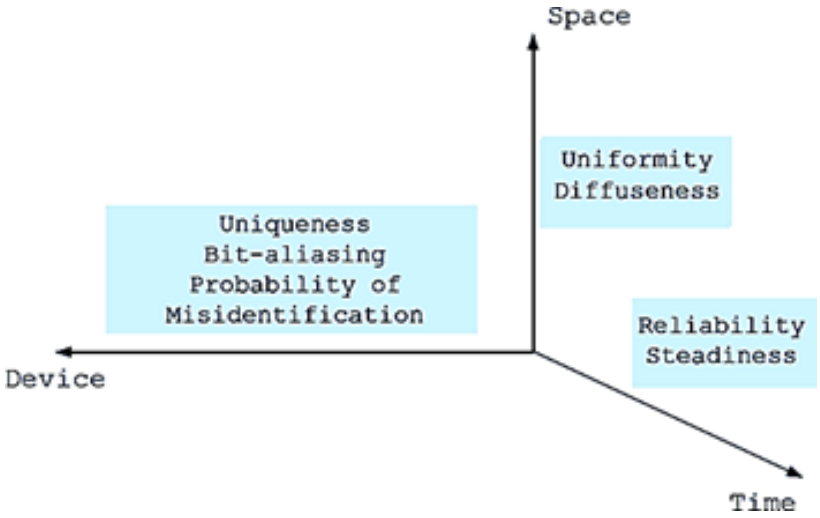


Figure 2.6: Final parameters mapped on the PUF measurement dimension Maiti et al. (2013)

Table 2.2 shows a summary of comparison between an Arbiter PUF and Ring Oscillator PUF – as an outcome for a given test case. It was concluded by Maiti et al. (2013) that the RO PUF performed better than the APUF in terms of uniqueness, bit-aliasing, and probability of misidentification (PMSID), while other parameters yielded comparable results from both PUFs.

2.1.5 Machine Learning technique to predict the model

In addition to the aforementioned metrics, the analysis performed on the response for the three dimensions of performance metrics, attack resistance falls under the category of device space, which is directly proportional to uniqueness. The methodology applicable to improve uniqueness also contributes towards improvisation of the attack resistance metric. Arbiter PUFs in comparison to Ring oscillator PUFs are regarded as stronger PUFs because of the greater number of challenge and response pairs (Herder et al. (2014). However, in delay PUFs, their delay is generated through a random path on the FPGA fabric and it is easy to model and predict its behavior through simple machine learning algorithm.

This literature survey focuses on modeling attacks on PUFs through machine learning algorithms. Attackers aim to construct a model using computational algorithms from a subset of challenge-response pairs of the designed PUF, attempting to predict the data correctly. These attackers have achieved significant success in accurately predicting the data, ultimately compromising the design's security Rührmair et al. (2010). Furthermore, different types of attacks can be applied, including invasive Roelke and Stan (2016), side-channel Rührmair et al. (2010) and virus attacks Guo et al. (2016). Linear based designs can be attacked with mathematical linear models Ye et al. (2018).

Table 2.2: Comparison results in percentage - Arbiter PUF with Ring Oscillator Based PUF(Maiti et al. (2013))

	APUF	**RO PUF**	**ideal value**
Uniformity	55.69%	50.56%	50%
Bit-aliasing	19.57%	50.56%	50%
Uniqueness	7.2%	47.24%	100%
Reliability	99.76%	99.14%	100%

Machine learning (ML) algorithms serve as natural and powerful tools for modeling attacks. Among various attacking methods, they exhibit significant promise in compromising robust PUF designs. One of the techniques, as proposed by Ye et al. (2016), involves dividing the collected dataset of challenge-response pairs for a specific PUF design into two sets. One set is utilized to train the model, while the other is employed to test the model's performance, aiming to regenerate the designed PUF for

predicting the response dataset.

2.1.6 Inferences from Literature Survey

- The comparison between Arbiter PUF and Ring Oscillator PUF has yielded a conclusion: the Ring Oscillator PUF, with its non-linear effects impacting security parameters, emerges as a superior option when contrasted with the Arbiter PUF. Additionally, within an FPGA environment, the Ring Oscillator PUF demonstrates greater efficiency compared to its performance in a resource-constrained setting. These findings, outlined in Table 2.2, exhibit notably higher values for the Ring Oscillator PUF in contrast to the Arbiter PUF.

- An effectively designed configurable Ring Oscillator on FPGA fabric necessitates precise symmetry to fulfill its intended purpose of facilitating PUF for extracting process variations.

- The manner in which the PUF is managed during the process of enrolling responses significantly influences the reliability, consistency, and distinctiveness of the PUF. The logic operations surrounding the PUF have a consequential impact on the aforementioned metrics. It is evident that implementing the evaluation logic alongside the PUF circuit as a hard macro will result in stronger challenge-response pairs.

- Several compensation techniques are available to mitigate the impact of environmental noise factors on the performance of PUF circuits. Adhering to these methodologies can significantly enhance the outcomes related to uniqueness, uniformity, and bit aliasing within PUF circuits.

- Arbiter PUF, except for its linear characteristics, is a strong PUF with a large CRP database. By attending to its structure and employing optimized area implementation, it is possible to improve the CRP space, thereby reducing the prediction rate on the PUF. The utilization of compensation techniques and methodologies aimed at improving uniqueness could directly affect the resistance on an APUF against potential attacks.

2.2 Formulation of Research Problem

The research problem was formulated according to the inferences drawn from literature survey.

- Selection of PUF circuit that suffices to silicon based delay model.
- Generating and enrolling the Stable response from the PUF circuit: How best a PUF module can extract the process parameters noise from the FPGA device by separating the static and dynamic noise parameters of the circuit?
- Evaluation of the PUF Circuit: Unless a proper study is conducted on the efficiency of the designed circuit and response it generates, a strong PUF with reliable and unique CRPs cannot be guaranteed. Quantification of the PUF's ability is necessary to produce uniform, reliable, and unique response consistently.
- Addressing Attack Resistance: Arbiter PUFs are prone to modeling attacks Sedcole and Cheung (2006), necessitating the development of a strong PUF design in terms of its attack resistance. An enhancement to the architecture of conventional Arbiter PUF should be the area to work on improving this metric of the PUF.
- Target Architecture Dependency of the PUF circuits: FPGA devices change technologies drastically with advanced architectural features. PUF circuits designed to fit in a specific FPGA device need to guarantee its portability on various upgraded devices.

The designed PUFs can be validated in an IoT environment. A PUF in an IoT environment will have a conventional gateway communication between the node and the host. It can secure a transaction happening between a mote (in this case it is a smart meter with PUF node) communicating with a host (server) through a gateway device (probably a Zync board with Ethernet port). The strength of the PUF is most evident during transactions that involve layered communication among peers. A specific methodology is identified from the literature survey to address these problems. This research would focus on incorporating and validating this methodology to scrutinize the scope and application of PUF in such transactions.

2.3 Proposed Methodology for Modelling and Evaluation of PUF

Performance metrics identified from Maiti et al. (2013) are uniqueness, uniformity, reliability and bit-aliasing. Process variations are the factors that directly affect these metrics. The compensation technique to isolate these variations and propagate an efficient response is described in form of a methodology. It also addresses environmental noise factors, such as voltage and temperature, as a part of its analysis. This methodology is organized from the literature survey as a part of the research problem solution. It starts from modeling of PUF to evaluation of PUF.

2.3.1 Identify FPGA based PUF Circuit

Maiti et al. (2013) compared Arbiter PUF with a Ring Oscillator PUF and concluded that RO PUF with non-linear effects on its security parameters was a better option than Arbiter PUF. Further, RO PUF works more efficiently in an environment of FPGA than in a resource constrained environment. The results in Table 2.2 show a significant number with RO PUF than Arbiter PUF. Going into the details of how an RO PUF is studied for its performance and efficiency, ROs are obtained by constraining the delay of FPGA component resources. Ring oscillator PUF is built with a prerequisite that nominal delay $T_{Nominal}$ should be maintained, irrespective of where the PUF circuit is placed and routed on FPGA. This is applicable to all delay based PUF circuits discussed so far, as well as the response generation circuit. This is done by applying proper placement and routing constraints to the design through static and dynamic timing analysis. Subsequently, the aforementioned RO PUF and Arbiter PUF work with a concept of symmetry in its implementation on FPGA devices. Unless this symmetry is guaranteed or $T_{Nominal}$ (shown in Figure 1.6) is maintained the same, process parameter noise cannot be separated from other static and dynamic components associated with the circuit. If this is not assured, then it is a failure of PUF's reliability with inter chip implementations. Thus, actual research problem lies in how best the process parameter variation is extracted. Literally, this is digging a step after GDSII in VLSI design flow. Ideally, any VLSI circuit would consider this process parameter noise to be negligible, because, it normally does not go beyond 200 ps. With all these considerations, a PUF design poses to be a challenge.

Figure 2.3 shows the evaluation logic used for RO PUF circuits. Given the oscillators to be symmetrically laid out, the frequency differences are determined by manufacturing variation and an output bit is equally likely to be one or zero, if random variations dominate. This is a generic understanding of how a PUF response bit is generated. Most of these implementations are IP protected and the knowledge is not discussed in open literature.

2.3.2 Quantize the random variation of circuit parameters that affect the delay of the PUF under test (PUT)

This concept involves the study of Static Timing Analysis of an FPGA logic circuit. It involves applying constraints to ensure the design meets the required timing without violating the resource timing specifications. The aim is to constrain the logic for packing the design and fitting it in specified logic resources of the FPGA. These steps would ensure that every time the logic runs on FPGA, it attributes to the delay due to process parameter variations, rather than influencing static and dynamic timing characteristics, which could include supply noise.

Maiti and Schaumont (2011) introduced a novel approach of quantifying the effect of systematic variations on bit aliasing in a PUF response, A RO-PUF circuit has n identically laid out ROs, RO_1 to RO_n, with frequencies, f_1 to f_n. A pair of frequencies, f_1 and f_2 tend to differ from each other. A response bit R_{12} is produced by the quantization of two real-valued quantities, f_1 and f_2 using a simple comparison method as follows:

$$\begin{aligned} R_{12} &= 1 \qquad iff\, f_1 > f_2 \\ &= 0 \qquad otherwise \end{aligned} \tag{2.2}$$

Various methods of quantization can convert pair of real-valued frequencies into binary bits. For example, Maiti and Schaumont (2011) indicated the use of index-based syndrome coding to convert Ring Oscillator frequencies into binary strings. If a comparison method is considered, a response bit is generated, as shown in equation (2.2). An RO-PUF is characterized to extract all possible challenge-response pairs (CRPs) under normal environmental conditions, wherein these pairs are stored in a secure database. Stanciu et al. (2016), Maiti and Schaumont (2011), Kodỳtek and Lórencz

(2015), showed this PUF to be FPGA friendly, because of the implementation issues on FPGAs. However, most of the FPGA manufactures reset the startup state of SRAM cells to a known value, rendering SRAM-PUFs difficult to use Sedcole and Cheung (2006).

Contrastingly, the silicon random function described by Gassend et al. (2002), the APUF by Suh and Devadas (2007), and the BPUF by Anderson (2010) necessitate structurally symmetric logic and routing to harvest Process and Voltage (PV)-induced mismatch. These assumptions may not be valid for the interconnect, as identical components within a logic block could be connected to the routing matrix using routes of different lengths based on individual placements. In addition, the routing tracks near the corner of an FPGA may be different from that in the middle part.

The existing FPGA design tools can minimize the delay-skew between a pair of routes, but they do not guarantee the structural symmetry. Even the fabric-level manipulation tools (e.g., Xilinx FPGA Editor) allow a designer to modify the placement and the routing of a design with only a higher level of abstraction. Therefore, the implemented PUF circuits often have delay-skew introduced by the design. In this case, process variations will not be sufficient to offset, resulting in a biased PUF output. It is observed that even a strict static timing analysis and constrained placement of logics produces a highly skewed output. However, RO PUF can be implemented using the hard macro technique provided by design tools Roelke and Stan (2016).
The quality factors as defined in Maiti and Schaumont (2011) for a PUF are a) Uniqueness, b) Reliability, and c) Security/Attack resiliency Moving forward in the quantization process concerning the aforementioned quality factors, it is imperative to estimate the uniqueness of a PUF by calculating the average inter-die Hamming distance (HD) across a set of chips. With a pair of chips, n and m ($n \neq m$), , both having n-bit response, Rn and Rm respectively, the average inter-die HD among a group of k chips is defined as:

$$HD = \frac{2}{k(k-1)} \sum_{n=1}^{k-1} \sum_{m=n+1}^{k} \frac{HD(R_n, R_m)}{n} X100\% \tag{2.3}$$

The above equation is an estimate of the inter-die variation in terms of PUF responses

and it is not the actual probability of the inter-die process variation Herder et al. (2014). If the PUF responses are truly random in nature, i.e., if each binary response bit (r) of a PUF has an equal probability of producing a '0' or a '1', the uniqueness should converge to 50%. Random response bits also produce a uniformly distributed response string R. This can be estimated by the Hamming weight (HW) of R. For uniformly distributed R, the value of HW is given by the following equation. It should converge to 50% of the length n of R.

$$Hammingweight = HW(R) = \sum_{t=1}^{n} r_t$$

Random response bits exist only if random process variation exists. In real time, the systematic or correlated process variation could be identified beside the random variation. The systematic variation becomes more significant by the continuous scaling down of silicon device feature size. If this is significant, then it would lead to ID collisions, as cited by Maiti and Schaumont (2011). The analysis of the systematic process variation on an RO-PUF was done by Maiti and Schaumont (2011) to mitigate the effect of systematic variations and maximize the uniqueness.

The total delay in a ring oscillator loop can be modeled as follows:

$$d_{LOOP} = d_{AVG} + d_{RAND} + d_{SYST} \tag{2.4}$$

where d_{AVG} is normal delay thats common to all inverters functional delay In the comparison method, two oscillator delays are compared directly or the differences between two oscillator delays are assessed, with an assumption and constraint that d_{AVG} would be a nominal delay, which would be the same across all oscillators due to the identical layout

$$\Delta d_{LOOP} = \Delta d_{RAND} + \Delta d_{SYST} \tag{2.5}$$

It is evident from the above equation (2.4) that Δd_{LOOP} is clearly a biased function with Δd_{SYST}. If Δd_{LOOP} is assumed to generate an unbiased result it should purely be due to Δd_{RAND} with out the effect of Δd_{SYST}. Redefining the Response bit R_{ab} in

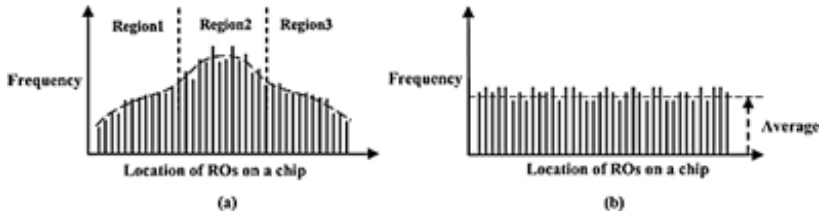


Figure 2.7: (a) Case with the systematic variation. (b) Case without the systematic variation(Herder et al. (2014))

equation (2.1) as below

$$\begin{aligned} R_{ab} &= 1 \qquad if \Delta d_{LOOP} < 0 \\ &= 0 \qquad otherwise \end{aligned} \tag{2.6}$$

R_{ab} could be modelled as a binomial trial with a probability of getting 1 or 0 , and the probability is biased on the systematic variations .

$$\begin{aligned} P &= 0.5 \qquad if \Delta d_{SYST} = 0 \\ &= 0.5 \pm \Delta_p \qquad if \Delta d_{SYST} \neq 0 \end{aligned} \tag{2.7}$$

It is seen that Δ_p is dependent on the location of ROs on FPGA from Figure 2.7. This is based on a conceptual idea only but not on real data. Xilix (2018) presumed that the distribution of RO frequencies is dependent on the region of their location. Maiti and Schaumont (2011) showed the distribution of frequency patterns in relationship with the location of ROs on the FPGA device. The bit aliasing effect is observed during comparisons between region 1 and region 2, and between region 2 and region 3. The value of P is important in the evaluation of inter-chip HD, i.e., the uniqueness of the PUF. The Hamiming distance between two n-bit responses R_i and R_j from a pair of chips, i and j, is calculated as follows:

$$HD(R_i, R_j) = \sum_{k=1}^{n} r_{ik} \oplus r_{jk} \tag{2.8}$$

Where r_{ik} is the kth response in n bit response string R_i of $chip_i$. If bit wise xor is assumed to be binomial trial with bit biasing as 1 and no bit biasing as 0 and

considering p_{xor} as probability of no bit biasing and q_{xor} as bit biasing , then no bit biasing probability could be derived as

$$p_{xor} = p_i + p_j – 2p_ip_j \tag{2.9}$$

without any systematic variation, i.e., when $p_i = p_j = 0.5$, (2.9) comes out to be 0.5. Presence of systematic variation would lead to biasing of the bit.

$$p_i = 0.5 \pm \Delta p_i \qquad similarly \qquad p_j = 0.5 \pm \Delta p_j$$

Assuming the case of opposite bias and substituting p_i and p_j in (2.9)

$$p_{xor} = 0.5 + 2\Delta p_i \Delta p_j \tag{2.10}$$

in case of same sign ,

$$p_{xor} = 0.5 - 2\Delta p_i \Delta p_j \tag{2.11}$$

It is evident from the equations (2.10) and (2.11) that when two ROs under comparison exhibit opposing biasing effects, it prevents bit aliasing in the generated response. Conversely, similar biasing effects between the two oscillators induce bit aliasing in the resulting response. Maiti and Schaumont (2011) elaborate on the boundaries of uniqueness, detailing the maximum and minimum limits in Table 2.3.

Table 2.3: Limits of PUF uniqueness

U_{MIN}	0	when l=k or m=k
U_{MAX}	$\frac{k}{2(k-1)}$	when l=m

Uniqueness with a system is measured by the average Hamming distance with respect to a set of k chips. As shown in Equation (2.1), the scenario involves k such responses out of k chips with n bit width each. Within this context, an arbitrary t_{th} bit response from the n bits across all k chips is considered for pairwise comparisons. It is observed that the computation requires $\frac{k(k-1)}{2}$ XOR operations for the t_{th} response bits. In order to derive limits on PUF uniqueness, arbitrary composition of different biases are assumed.

- Let l = number of response bits with $p = 0.5 + \Delta p (0 \leq l \leq k)$

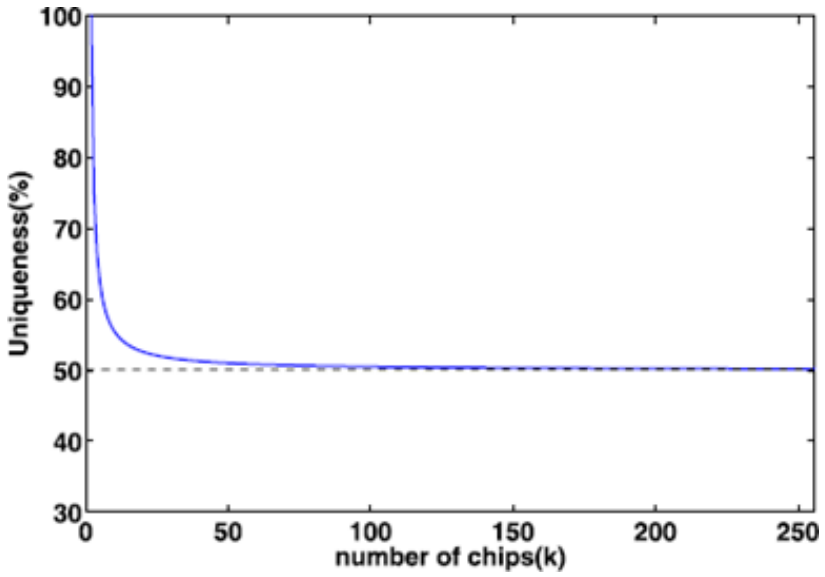


Figure 2.8: Maximum PUF uniqueness for k chips Herder et al. (2014)

m = number of response bits with $p = 0.5$–$\Delta p(0 \leq m \leq k)$, k-l-m = number of response bits which are unbiased

- Assuming that there are same number of 1s and 0s on unbiased bits, the number of 1s in response bits $N1 = l + \frac{k-l-m}{2} = \frac{k}{2} + \frac{l-m}{2}$

- Similarly number of 0s in response bits $N0 = \frac{k}{2} - \frac{l-m}{2}$ and the total number of mismatches or number of ones as output to xor operation for t_{th} bit is $N1 * N0 = \frac{k^2}{4} - \frac{(l-m)^2}{4}$ and uniqueness as defined by average hamming distance would be

$$U = \frac{N_1 N_0}{\frac{k(k-l)}{2}} \tag{2.12}$$

$$U = \frac{k^2 - (l-m)^2}{2k(k-1)} \tag{2.13}$$

$$U_{l=m} = U_{max} = \frac{k^2}{2k(k-1)} = \frac{k}{2(k-1)} \tag{2.14}$$

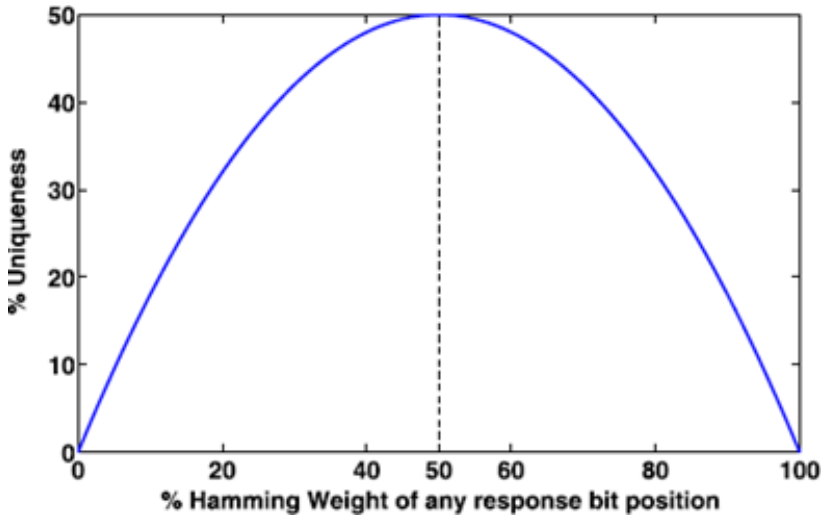


Figure 2.9: PUF uniqueness vs. HW of any bit position across sample chips Herder et al. (2014)

Table 2.3 ,could now be correlated with the maximum and minimum values of uniqueness(U), assuming for large values of k , 1/k tends to be 0 and U_{max} comes to 0.5. Eventually proportion of 1s and 0s determine uniqueness between 0 and 0.5. Figure 2.8 shows maximum PUF uniqueness for k-chips.

If x is considered to be the proportion of no of 1s , then

$N1 = x \times k$ for k chips$(0 \leq x \leq 1)$

$N0 = (1 - x) \times k$

Thus , eq (2.11) could be expressed as a function of proportion of 1's and 0's as follows

$$U = \frac{N1 \times N0}{\frac{k \times k-1}{2}} = \frac{k \times 2x(1-x)}{k-1} \tag{2.15}$$

Table shows that when l=m then the U_{max} value as shown in Equation 2.13; hence, Equation 2.14 finally comes out to be

$U = U_{max}.4x(1-x)$

U_{max} tends to 0.5 for large values of k, so Equation (2.14) would converge to

$$U = 2x(1-x) \tag{2.16}$$

Thus , Equation (2.15) explains Figure 2.9 as uniqueness reaching to maximum when Hamming Weight is 50% i.e when x = 0.5 in Equation (2.15) uniqueness reaches to maximum of 0.5.

The quantitative analysis conducted on the uniqueness concerning HW and HD reveals that an even proportion of 0s and 1s would achieve maximum uniqueness. Conversely, the uniqueness would be minimumal when the values of 'l' or 'm' approach 'k'

The following are the concluding remarks to achieve maximum uniqueness:

1. Keeping no of unbiased bits high by making l+m as low as possible
2. Keeping l = m

The discussion for t_th bit could be interpolated to all the n bits of the response. The effect of l and m with systematic variations is yet another point to be observed. Systematic variations are defined as an unwanted correlated variation due to spatial location of a ring oscillator on a chip. This may happen for many reasons, one being die pattern with its layout dependency, along with wafer thickness. However, though the variations are systematic inside a die, this could be random over a group of chips, which helps in achieving an even number of l and m, thereby increasing uniqueness. However, according to the discussion on bit aliasing and the biasing effects, if a group of chips are being biased with a similar trend of systematic variations, it leads to bit aliasing on the response, thereby increasing either l or m, which is not desirable. Thus, a compensation method was proposed by Maiti and Schaumont (2011).

The spatial nature of systematic variations signifies that adjacent interconnects and logic experience similar variations. By comparing physically neighboring Ring Oscillators (ROs), the impact of systematic variations Δd_{SYST} is minimized, leading to an offset effect Δd_{RAND}). This counteracts the impact of systematic variations across a group of chips experiencing similar biases, thereby mitigating bit aliasing. Consequently, this emphasizes that the placement of RO pairs significantly influences uniqueness. To summarize, the subsequent steps outline the procedural design of a RO PUF and their positioning on an FPGA, considering their evaluation for PUF response.

1. Ring oscillators to be placed as close as possible in adjacent logic blocks of FPGA.

2. Response bit is generated based on comparison of adjacent bits.

This procedural way of design works for environmental effects, such as temperature, with an assumption that closely packed ROs would tend to have a similar effect of environmental conditions. This arrives at a stable and unbiased unique response. This methodology could be streamlined further by evaluating the generated response bit for independent comparisons and stable response bits for reliability as well. A similar method was described by Bernard et al. (2012).

Reliability is another quality factor that introduces average intra Hamming distance among responses for the same challenge driven into PUF circuit under the influence of various noise parameters, such as voltage fluctuation, temperature, etc. These factors tend to introduce errors in the response bits for the same challenges generated multiple times. This affects the reliability of a PUF circuit and similar quantitative analysis could be extended to reliability to measure intra chip variations. Figure 2.10 shows a detailed distribution of RO frequencies within a chip. Maiti and Schaumont (2011) observed that the peaks indicated the randomness that could be extracted to build a reliable PUF response and systematic variations are seen to vary from the left side region to the middle and from the middle to the right regions. Suh and Devadas (2007) reported a similar systematic variation existing in 180 nm and 90 nm FPGA devices. This discussion overlooks the interconnect delay noise related to

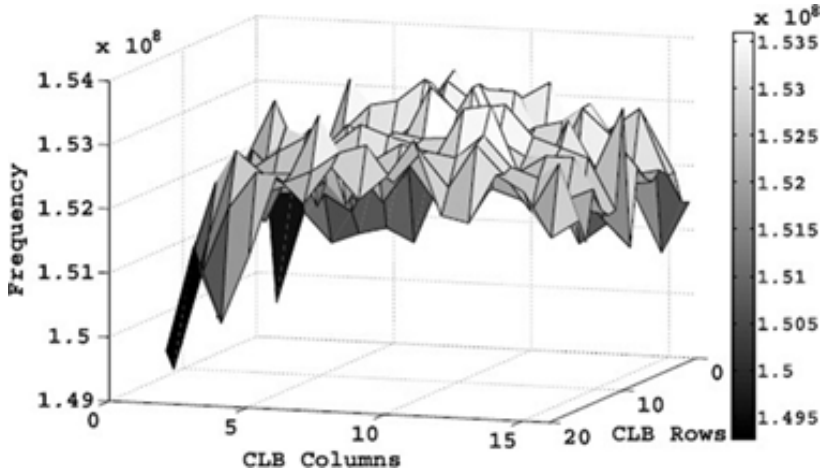


Figure 2.10: Systematic intra die variations of RO frequencies Herder et al. (2014)

FPGA devices. In essence, the inference suggests that optimizing control over inter-

connect delay, combined with systematic and random processes, will yield an optimal solution for achieving uniformity, reliability, and mitigating bit aliasing issues.

2.3.3 A connected System Model

This section discusses about the application environment of the PUF that is being designed. Figure 2.11 shows the networked architecture of the manner in which PUF node at the user end communicates with the host through the gateway. The task of the gateways is to translate the communication between the host and the PUF nodes from Transmission Control Protocol (TCP) (host side) to Universal Serial Bus (USB) (PUF node side) and to configure the PUF nodes on the host's request. The gateways are implemented on the Zync boards (ZYBO), running the GNU/Linux/petalinux operating system. The host is implemented on a server machine running the GNU/Linux Debian operating system. It is inferred that the point of interest lies in developing the security module, which solely speaks of its performance metrics and efficiency associated with its implementation on the platform. Details of the smart meter's communication content and other backend details of the data being carried by it are not discussed in this thesis as it is beyond the scope of this research. Two steps are

Figure 2.11: Overview of connected environment Chongyan Gu (QUB)

involved in the achieving the above objective. First, it involves hardware/software co-simulation environment where PUF nodes generating the CRPs are directed to come from the real FPGA hardware. Post processing the PUF response within the PUF node and the communication interface to the host through the gateway is simulated as a Bus Functioning Model (BFM). It is verified through a System verilog based TESTBED. Later it is validated in the final step by prototyping it on Zync Board with a host server. The above problem can be handled in system verilog by modeling a TLM packet with PUF data as a part of it. It requires CAD tools that support the flow of FPGA based system designs. A hardware in loop simulation model can be developed on the Xilinx ISE design suite. It is an integrated environment that has ISIM for behavioral simulation, post placement and routing simulation models, and

Xilinx synthesis tool for synthesis and optimization. It also has Xilinx FPGA editor for mapping, placement, and routing plan ahead for PIN planning and placement constraints. As far as PUF circuit design is concerned, it needs careful placement and routing of the circuit, because symmetry is the main concern. It needs tweaking into the process of designing an FPGA development environment. However, such an environment requires an open source FPGA development tools. For example, if interconnect lengths are under control, then it is obvious that systematic variations are controlled. Most of the open literatures show their work at improving the obtained PUF response, rather than ensuring if the circuit could be controlled for its systematic variations. Ye et al. (2016) and Herder et al. (2014) conducted a quantitative analysis on how to identify the effect of systematic variations on response bits generated from a PUF, and also how many comparisons could be made and how many of these could turn out to be independent. Most of these details are already discussed in Chapter 2.

If an algorithm is designed to control the placement and routing of logic specific to PUF designs, then it would ease its portability on to various architectural upgrades in FPGA devices. The following chapter would discuss about the PUF design involving the method ology discussed in the chapter. It addresses the modeling of PUF and the logic circuit to propagate the response and acquisition of the response from FPGA and process it to generate the signature.

2.4 Inferences from Literature survey for methodology

- Ring oscillators are to be placed as close as possible in adjacent logic blocks of FPGA. Response bit is generated based on comparison of adjacent bits.
- Bits affected by noise parameters are to be identified and discarded for PUF ID generation and the effective response bits that are stable need to be used.
- Highly stable bit positions need to be identified while generating the PUF response. The pattern can be taken as reference in overcoming the effect of systematic variations on FPGA devices.
- The responses need to be collected from the PUF circuit so that the minimized

effect of evaluation circuit skew in generating a stable response is observed.

- Entropy is considered as the main aspect producing an efficient PUF circuit. This is taken as reference in design for PUF evaluation logic. If the PUF circuit is stitched following the method specified in the previous section to compensate the voltage and temperature effect on the PUF and follow the placement technique, then uniformity and reliability can also have good numbers.
- Another important aspect that needs to be taken care is to utilize the resources in generating the response efficiently with respect to area and power. This is because the ratio of number of ROs to the number of bits it generates is one of the important factors for reliability and uniformity. This quantization would help in identifying the resource utilization and area efficiency of the design.
- Increasing the challenge space on existing APUF will make it more unique, which in turn would help in improving the attack resistance factor.

Work plan based on outcome of survey

- Design of a Configurable Ring Oscillator (CRO) PUF in verilog for its implementation on the FPGA device.
- Development of a system level test-bench for hardware-in-loop verification.
- Implementation of the PUF under test with the methodology discussed in the literature survey, section 2.3.
- Acquisition of the responses using the system level test bench
- Analysis of responses received for performance metrics surveyed from literature and inference of observations in comparison with a conventional CRO PUF design and methodology propose in this research for the CRO PUF design.
- Designing of an Arbiter PUF (APUF) in similar lines, with proposed architectural enhancement and verification of the PUF with a developed system level test bench.
- Utilization of logistic regression and sigmoid as activation function and development of the model for APUF.

  - Analysis of the prediction rate with the model on the designed APUF with proposed architectural enhancement and comparison with the work done by authors on the same PUF, followed by consolidation of results and observations.

- Creation of the observed PUF node on a smart meter that is assumed to be on an IoT Environment and validation of the PUF based authenticated transaction from the host to the device.

Chapter 3

Design and Validation of PUF

3.1 Modelling of PUF

3.1.1 Modelling a delay based PUF circuit

Modeling of a Physically Unclonable Function, as discussed in Chapter 1 (Figure 1.6), commences with the extraction of signatures from random delays associated with process variations. These variations are of two types, namely, Systematic process variation and Random process variation. Systematic process variations are correlated throughout the silicon die, can be predicted in the fabrication process, and are compensable by adjusting offsets in data-sheets. Random process variations are due to small corner mismatches during fabrication and are accommodated by the fab, with only a percentage offset. Figure 3.1 shows the process induced variations in transistor fabrication with their deviations and variance in the respective data-sheets of the device, except for random process variations.

Referring to Figure 1.6 in Chapter 1, the nominal delay is supposed to be the systematic variation, while $T_{random1}$ and $T_{random2}$ are the random variation. Thus, Delay PUFs on silicon are about extracting this Trandom to generate the keys or for authentication. The survey in Chapter 2 indicates that Ring Oscillator based FPGA delay PUFs are efficient for implementation on FPGA devices.

The first step in this modeling approach is to design a logic circuit. Figure 2.1 in Chapter 2 shows a basic ring oscillator producing oscillations whose frequency is

determined by the delay of a number of inverters. This frequency can be controlled with an enabler to start or stop the oscillations. Analysis of the delay of this combined loop is crucial and it can be analyzed from Equation 2.3 in Chapter 2.

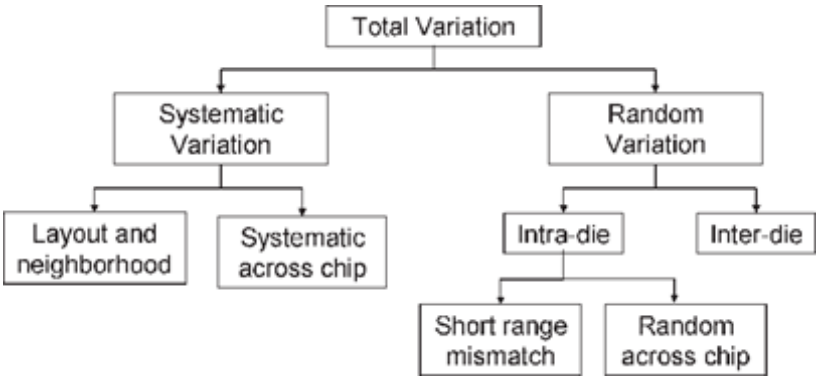


Figure 3.1: Classification of the different types of variation in transistor characteristics

The total delay in a ring oscillator loop can be modeled as:

$$d_{LOOP} = d_{AVG} + d_{RAND} + d_{SYST} \tag{3.1}$$

where d_{AVG} is the normal delay that is common to all inverters' functional delay, which is not the concern. The point of interest lies in the method used to extract the random delay while decreasing the effect of systematic delay.

If n-number of inverters are accommodated on the CLB structure of FPGAs, then each inverter will extract the delay associated with that region of slice and come up with the oscillations controlled by variations, as shown in the equation above. Literature has several proposals on this; however, the work is confined at identifying the best use case scenario of Ring oscillator PUF as Configurable Ring Oscillator (CRO) Maiti and Schaumont (2011). Figure 3.2 shows the initial model of a ring oscillator placed on the fabric. However, to manage the delays mentioned in the above equation, a quantitative analysis is conducted in Chapter 2, section 2.3.2. This analysis aims to extract only random delay, offsetting the systematic delay. It involves comparing the frequencies of adjacent oscillators within the conventional RO. In this scenario, the RO within one CLB is compared with the adjacent RO in the next CLB. Consequently, two CLBs are utilized to generate a single response bit, contributing to a heavier construction of the PUF. The outcomes of a literature survey incorporate

an analysis by Maiti and Schaumont (2011), proposing a configurable ring oscillator (Figure 2.2 from Chapter 2) to address this issue. As a result, the model's circuit structure is modified and the logic is implemented on the fabric, as shown in Figure 3.4. The implementation of configurable ring oscillator PUF on Spartan 3E FPGA is

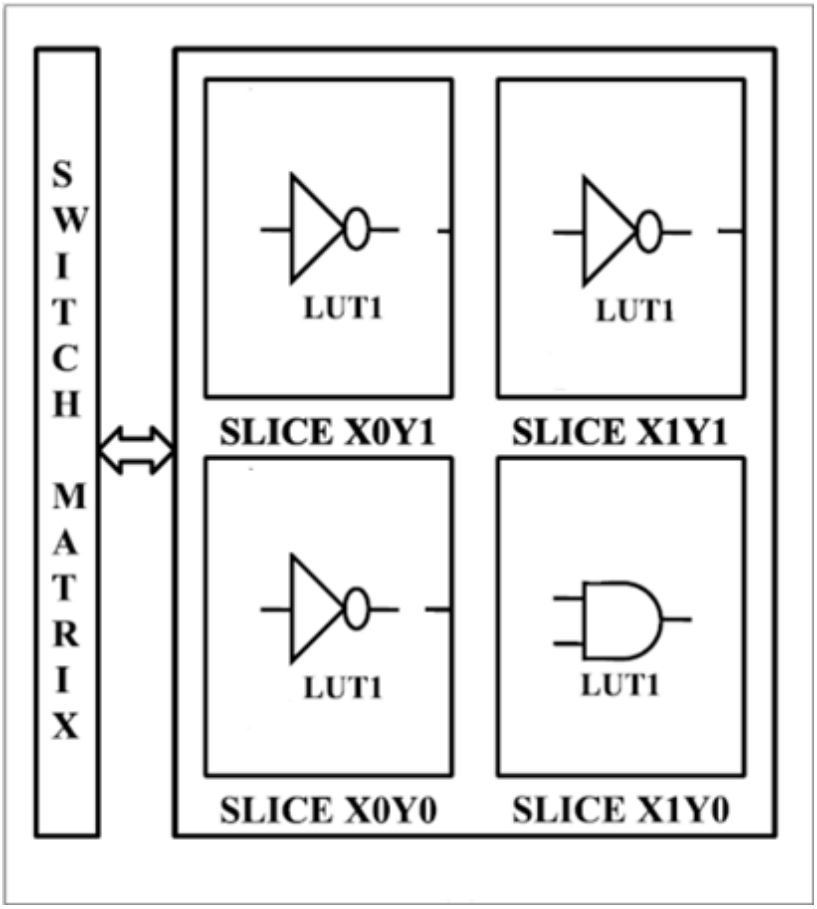


Figure 3.2: Placement of Ring Oscillator on FPGA fabric

shown in Figure 3.4. It is observed that single CLB can generate the eight possible RO outputs with three select lines from 000 to 111, resulting in 8 RO outputs. A pair of ROs with maximum frequency difference could be fixed as a reliable Challenge and Response Pairs, as shown in Figure 3.3.

In the study conducted by Kodytek and Lórencz (2015), it was found that the implementation of the same structure would require 8 CLBs for 1 out of k×N masking scheme (k=8). Thus, Maiti's CRO as proposed in the work by Maiti and Schaumont (2011) has brought down the area by 8 times in implementing the same scheme. Maiti's

analysis in the same paper on uniqueness and reliability on RO PUF has provided a valuable understanding of the estimating of uniqueness, specifically addressing the maximum uniqueness for a set number of chips and elucidating how the Hamming distance (HD) and Hamming weight (HW) can serve as metrics for assessing PUF device performance. This analysis comprised a detailed mathematical analysis Maiti and Schaumont (2011) of the aforementioned concepts, delivering both qualitative and quantitative assessments of ROPUF circuits and could also be extended to any PUF circuits. Quantitative analyses of these parameters are discussed in subsequent studies Bernard et al. (2012), Suh and Devadas (2007). Figure 3.4 shows a CLB with four slices, each containing a LUT and a switch matrix. The applied methodology

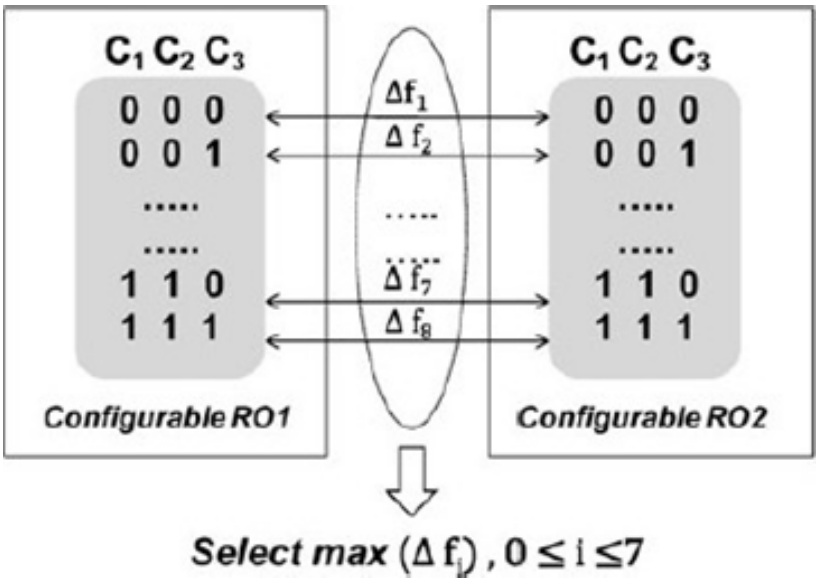


Figure 3.3: Comparison of frequencies Maiti and Schaumont (2011)

is primarily focused on analyzing the delays arising out of CRO PUFs so that the device is authenticated in the form of unique, uniform and reliable challenge and response pairs with reduced bit aliasing effects. The principal objective is to mitigate bit aliasing effects while strengthening the CRO PUF, which has been traditionally regarded as a weaker form of PUF in existing literature. The methodology proposed in the following sections in enrolling the PUF response makes it function like a strong PUF. The following text delineates the steps undertaken to achieve this objective.

- Step 1: Create an array of 128 CROs with 4 inputs, one to enable the PUF and the remaining 3 to select the configuration on the PUF. Enable high on the PUF

would run the oscillator and the frequency of oscillations could be measured by using a frequency meter, a time to digital (TDC). This segment involves the design specifications of a frequency counter.

- Step 2: Design a 128X1 Multiplexer where select lines are so chosen that they select the required Configurable Ring Oscillator's frequency to come out. This phase involves the methodological intricacies of designing an efficient frequency counter.

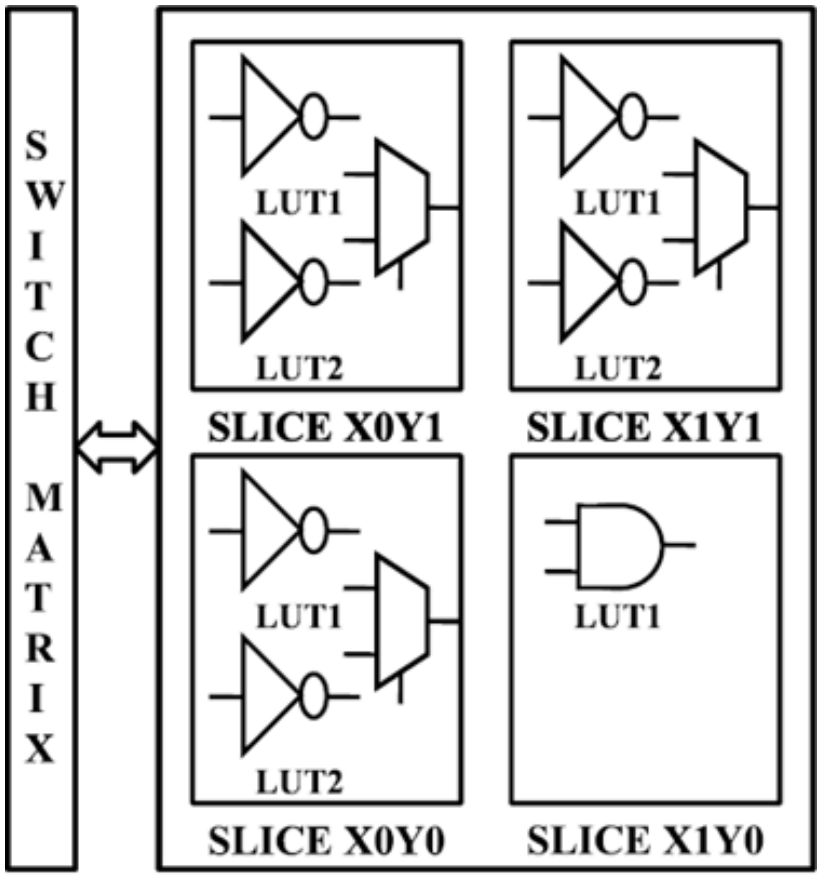


Figure 3.4: Enhanced Mapping of CRO in a single CLB in Spartan 3E FPGA Kodỳtek and Lórencz (2015)

- Step 3: A controller need to be designed to control the TDC block and the CRO ARRAY block, along with reset generation logic to control the counter that generates select lines for the design block.

- Step 4: According to the inference drawn from the previous chapter in the

literature survey in section 2.3.3 and shown in Figure 3.5, the study identifies the high stable bit positions in generating the PUF response. In order to determine this, the period on the enable pulse of PUF circuit needs to be found which would generate high stable bits on the frequency captured on n-bit register. In this work, optimum n value is analyzed, which gives a stable PUF response. Identify the optimum time pulse required in registering the frequency.

Figure 3.5: Behavior of positions' stability in a 16 bit response Herder et al. (2014)

- Step 5: Design a FPGA communication interface to stream the calculated frequency to a local machine, because FPGA does not have the required resources to capture this heavy data. The FPGA's Enhanced Parallel Port (EPP) interface must be used to capture the FPGA device's frequency response for the PUF device.

- Step 6: In continuation to the inferences from section 2.3.3, structure the hard macro of the CROs so that process variation would dominate the randomness due to interconnect. After capturing the frequency data generating from all configurations of the PUF through the communication interface, post processing of the responses can be done to generate a signature or challenge and response pairs.

Conventional PUFs discussed in the literature lack a comprehensive setup to capture all frequencies and enroll the PUF responses. These PUFs generate a single bit response with an additional comparator logic. However, the additional comparator logic would only generate the response and may definitely bias the response generated by the PUF Stanciu et al. (2016). Hence, this work has the job of extracting delays on device; but after extracting the delays, the extra circuit load to calculate

the PUF response is done using a processor on-chip/off-chip. Subsequently, the work constitutes the implementation of the controller logic in the application interface of FPGA. This strategy ensures that the only hardware overhead for the PUF comprises the multiplexer logic and the register logic essential for frequency capture. These two components, if designed as hard-macros, can potentially be integrated into the PUF to generate the signature, as inferred from the study of Stanciu et al. (2016). The following sections provide a detailed and systematic discussion of the procedures involved.

3.2 Designing an array of 128 CROs

Design of a configurable ring oscillator as discussed in the literature, involves constrained placement of logic blocks on the FPGA fabric and making of an array of the same, so that it can generate a configurable ring oscillator output as a PUF design. It is shown in Figure 3.4 that a single Configurable Logic Block (CLB) of the FPGA is used to generate the 8 possible RO outputs with 3 select lines, wherein 128 such CROs are designed as an array. As shown in Figure 3.6, the output of each CRO is driven to eight 16×1 multiplexers. The eight outputs from eight 16×1 multiplexers are driven to one 8×1 multiplexer. It is important to observe from the inference of the study of Stanciu et al. (2016) that these multiplexer blocks also need to be hard macro. Figure 3.7 illustrates a model employing four select lines from a 16×1 multiplexer and three lines from a 8×1 multiplexer. Totally, seven select lines along with CRO's three configurable inputs generate a 10 bit challenge, resulting in 1024 response bits. Concerning FPGA fabric utilization, 128 CLBs were utilized in placing the 128 CROs.

3.3 Delay analysis on a Configurable Ring Oscillator PUF with frequency meter

This section describes the methodology employed for delay analysis on the CRO PUF. This analysis influences the performance metrics of the proposed PUF while analyzing the potential influence of adjacent digital circuit activity on PUF responses. The characteristics of a PUF are supposed to be robust, unique, easy to evaluate, difficult to replicate, and very difficult or impossible to predict. The present study focuses

on the implementation of a frequency counter logic for evaluating PUF circuits in generating stable challenge and response pairs. Stanciu et al. (2016) identified that PUF circuit and its associated adjacent logic be designed as a hard macro to ensure the propagation of response bits. Considering this, the following experiment was conducted for identifying an optimized hardware with specific timing constraints for designing a frequency counter.

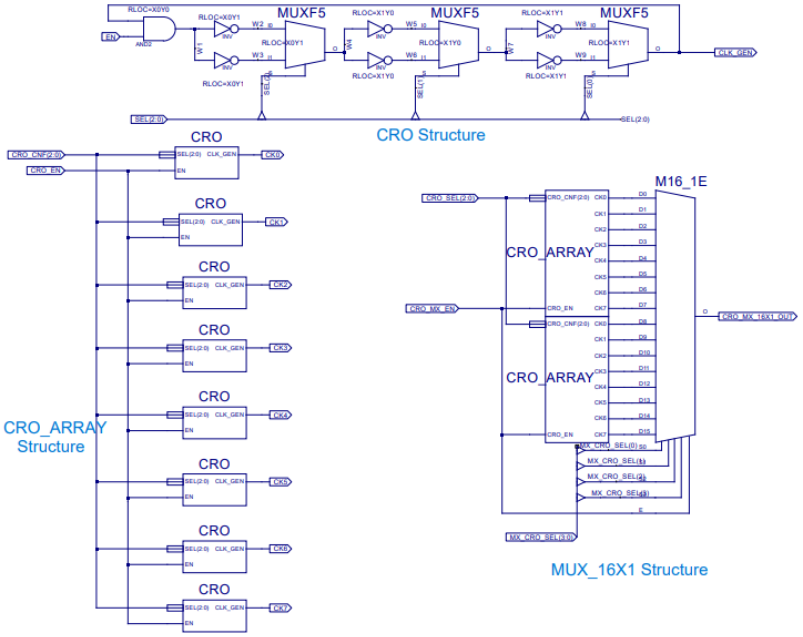


Figure 3.6: Design of Array of 16 CRO PUFs

3.3.1 Frequency Meter

The basic principle of calculating the frequency of a random oscillating signal was explained by Anderson (2010), wherein a digital gate of a specific frequency is designed as a single pulse. The number of oscillations is measured inside this pulse to calculate the frequency of the signal according to the pulse designed. For example, if the pulse is one second wide, then the number of oscillations measured inside this pulse would give the frequency in Hertz. Similarly, if the pulse width is one micro second, then the number of oscillations measured in this pulse would be given in Mega Hertz. This method involves the usage of time base as a reference clock from the board to generate a unit of time pulse. The enabling and disabling of the counter counting of edges of the internal signal (for which the frequency needs to be determined) is controlled by

the unit pulse generated. This is called a normal direct frequency counting. The problem with this method is that the number of digits displayed is dependent on the input frequency 'f' given in Equation (3.2).

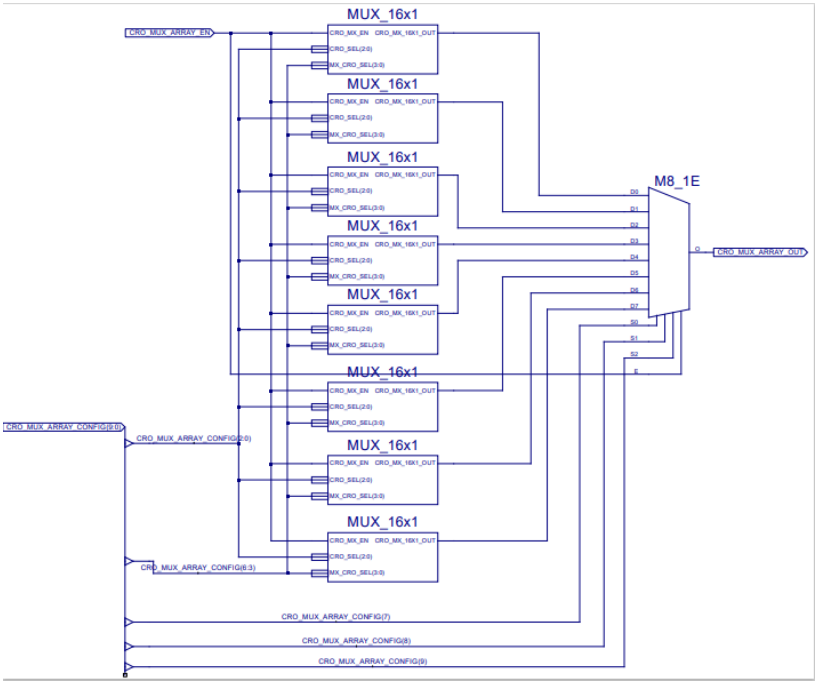


Figure 3.7: Design of Array of 16X8 (128) CRO PUFs

$$f = \frac{events}{time} = \frac{\text{counter value}}{one - second} = \text{counter value in Hz} \tag{3.2}$$

If one-second unit time pulse is considered, then the displayed frequency will be in hertz and the need for representing the frequency would require the corresponding number of digits. An alternative method involves reciprocal counting wherein reference clock periods are counted instead of counting the input signal. Rather than counting the input signal, the starting and stopping of the counter are synchronized with the input signal for which the frequency needs to be determined. For example, the counter is started at the rising edge of the input signal and stopped at the next rising edge. Consequently, the counter reading is actually the period of the input signal in multiples of the reference clock. This approach is termed as reciprocal counting Anderson (2010) since Frequency 'f'=1.

3.4 Design of a Frequency Counter for Configurable Ring Oscillator

Configurable ring oscillator has been designed to generate a random frequency Kodytek and Lórencz (2015), Bernard et al. (2012). It involves manual placement with constraints Majzoobi et al. (2010). The aforementioned designs are evaluated to identify their feasibility for an unbiased response bit during inter and intra chip analyses. Additionally, these designs are assessed for their capacity to produce an optimal unit time pulse using minimal hardware. This time pulse is used to control the enable function on the CRO. The optimal pulse would determine the stable bit on frequency output **?**. The mechanism involves a mod N counter design to design a specific clock period. For a 50 MHz clock on board, it requires a MOD 50 counter to generate a pulse of one microsecond. Similarly, a MOD (5010^3) counter is required to generate a one millisecond pulse and a MOD (5010^6) counter is required to generate one second pulse width. In this study, all three unit time pulses were generated to analyze the response of PUF macro. The implemented block diagram is shown in Figure 3.8, wherein customized counters involved in the design are placed using relative location constraints, thereby optimizing the utilization of resources available in the slice. This approach ensures a more densely packed design, thereby enabling better extraction of process variations with respect to the design. All the resources available in the design are well utilized so that a symmetry is achieved in the design and a constrained placement would ensure the propagation of unique response with respect to the device. The forthcoming sections would analyze the frequency observed in the design shown in Figure 3.8.

3.5 Identify the optimum Pulse window (Unit time pulse) to capture the stable frequency from the PUF

This is a detailed analysis on Step 4 in section 3.1.1. A direct frequency counting method Anderson (2010) has been used to find the frequency of the oscillating signal originating from the CRO. The unit time pulse block shown in Figure 3.9 is actually a parameterized Verilog RTL, engineered to generate a configurable MOD N counter

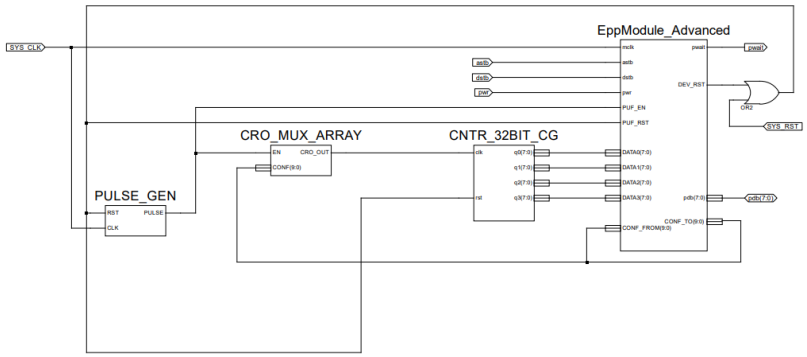


Figure 3.8: Block Diagram of CRO PUF with EPP interface as implemented on Spartan 3E FPGA

for the required time pulse using the board clock frequency of 50 MHz. Figure 3.8 shows the implementation of the designed frequency counter with a Configurable Ring Oscillator. The pulse generated from Figure 3.9 is used as the enable signal for the Configurable Ring Oscillator. The oscillations observed during the pulse period are actually due to the combinational loop with the delayed cascaded inverters.

The counter, denoted as CNTR 32BIT CG, will have the CRO MUX ARRAY output as its clock source, thereby enabling the counter to count the number of oscillations of the selected CRO. The design comprises a 128 CROs array, constructed with eight 16×1 multiplexers and one 8×1 multiplexer. The select lines will be 4 lines from 16×1 and 3 lines from 8×1, culminating in a total of 7 lines. When combined with CRO's configurable inputs, this accumulates to a 10 bit select input, facilitating the measurement of one frequency out of 1023 possible frequency outputs. The most important part of the entire process is to acquire and interpret the data available from the counter.

Reset logic is required to reset the PULSE GEN and CNTR 32BIT CG blocks as and when the frequency is registered. Subsequently, an edge detection logic is used in finding the negative edge on the pulse generated. When the pulse goes low, the edge detector would assert a RESET OUT signal, which would control the reset going into the required blocks. Thus, a MOD counter would control the required RESET OUT pulses to observe the specific number of frequencies out of logic. In this case, the MOD 1023 counter would facilitate the generation of RESET OUT signal 1024 times. The

negative edge on the pulse can be used in capturing the data on the counters. The captured data can be displayed on seven segment displays. This is the conventional way followed for observing the frequency from any oscillating logic. However, in this case, in order to observe and analyze 1024 frequency outputs with the aforementioned method would be a tedious job and it would run short of resources on FPGA to register so many frequencies. Hence, an efficient model has been devised to interact with the FPGA device and extract the frequency information out of logic.

The following section shows how an enhanced parallel port interface is used in addressing the aforementioned problem. The plots shown below depict data acquired from a single CRO at a particular CLB of a specific FPGA device. It is quite evident from these results that am increase in the Gate Pulse to measure the frequency would render more stable results. However, increasing it more than a specific value will result in more time being consumed for generating the device ID and it has to sort a greater number of CROs placed on the FPGA device. The observation conveys that any pulse width between one millisecond and one second is optimum for generating a stable result with significant improvements on CRO based PUF circuit Maes and Maes (2013).

3.5.1 Observations made on achieving optimum pulse window

Upon establishing the environment to record the frequency values from CROs, the subsequent step is to start the process of streaming the frequencies. In order to accommodate the environmental effects on the device's response, an additional dimension is added along with configuration and frequency output. The third step is the the random sampling times of the device across all configurations to understand the stable bits obtained in the response (section 2.3.3, Figure 2.11) Maiti and Schaumont (2011).

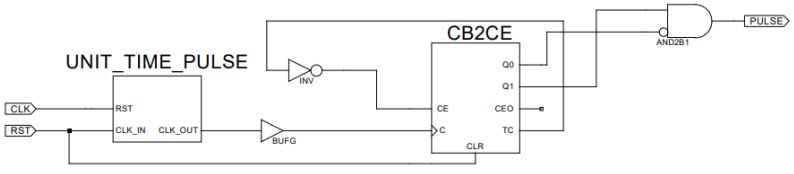


Figure 3.9: Block diagram of Unit Time Pulse Generator

- X-axis - time (at different intervals)

- Y-axis - Frequency

- Z-axis - configuration of specific CRO at random locations on the fabric

Figure 3.10 shows the plot displaying a 1 microsecond pulse window enabling the PUF circuit across eight configurations, yielding a random distribution of frequencies, which means that the response bit generated from this pulse window would not be stable and could therefore be discarded. Contrarily, Figures 3.11 and 3.12 reveal an uniform distribution of frequencies across random time intervals, signifying that the response bit would be stable during this period of 1ms to 1s. Although Figure 3.10 indicates minor disturbances during two random time instances, it can be deduced that the 1 millisecond to 1-second interval serves as an efficient pulse enable window for the PUF device to capture stable response bits. In section 3.7, some more important inferences are drawn from these plots combining with the methodology described in section 3.6.2 to enroll a stable response.

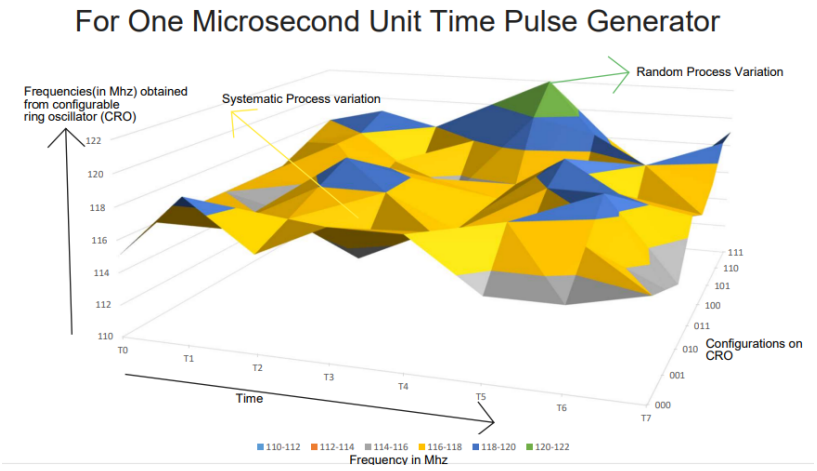


Figure 3.10: : Plot 1- Surface Plot of frequency with respect to configurations (000 to 111) on CRO while sampling at random times (T0-T7): With One Micro Second pulse generation

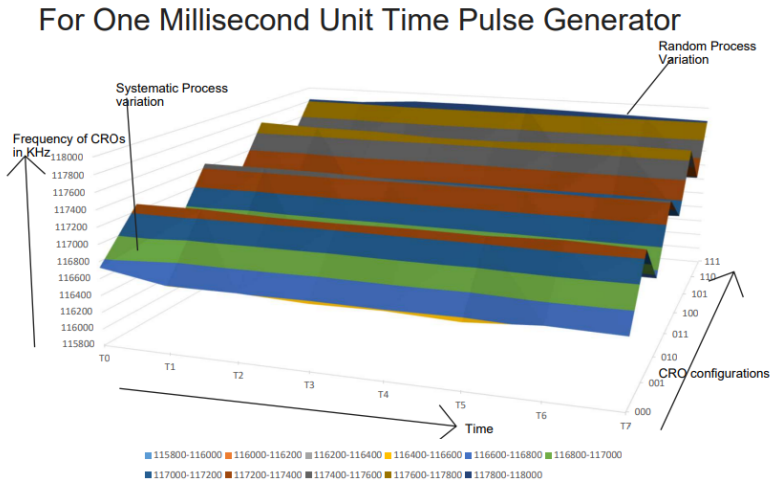


Figure 3.11: Plot 2 - Surface Plot of frequency with respect to configurations (000 to 111) on CRO while sampling at random times (T0-T7): with one Millisecond pulse generation

3.6 Hardware in Loop (HIL) Verification environment

In order to sort the frequencies originating from various CLBs within different CROs, it would be difficult for an Integrated Logic Analyzer to stream and register the signals stemming from the FPGA device. Thus, a communication interface was established between FPGA and the HOST CPU. Enhanced Parallel Port (EPP) interface was introduced to solve this problem and it is available on diligent Basys2 Board. Leveraging SDK libraries and integrated functions within Visual Studio via an Integrated Development Environment (IDE) serves as a means to facilitate the development of an Application Programming Interface (API) to enable seamless communication between the device and the host CPU. The objective remains centered on continuously streaming frequency count data generated through the logic of a ring oscillator and subsequently processing this data to produce a signature for the PUF. In order to do this on these 3-series devices, an on chip debugger is required. However, the substantial spatial demands it incurs within the fabric prompts the exploration of alternative solutions. Fortunately, the board from Digilent provided an interface (EPP), with a support of 256 8 bit registers, alleviating the need for excessive space utilization. This interface enabled a hardware in loop verification environment, enabling the streaming

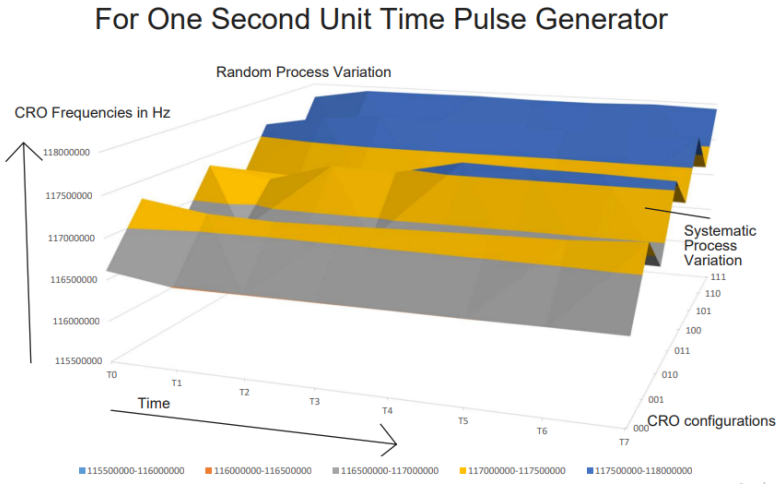


Figure 3.12: Plot 3-Surface Plot of frequency with respect to configurations (000 to 111) on CRO while sampling at random times (T0-T7): with one-Second pulse generation

of data from the FPGA to the host machine and reciprocally transmitting inputs from the host machine to the FPGA via the EPP interface on the Basys2 board.

The EPP interface on the Basy2 board has 256 8-bit registers, establishing communication between the logic on device and the host machine through a bidirectional port. In this case, if device's logic needs to communicate frequency measured data to the host, then it writes the data into the available registers on EPP interface by specifying the data and address, respectively (Host Read), as detailed by George (2004). Conversely, when the host needs to transmit control data to the device, it executes a Host Write cycle. The control signals—Data Strobe (DSTB) and Address Strobe (ASTB)—govern the categorization of the cycle as a write or read cycle. The timing specifications for the wait signal prior to the subsequent transaction are explicitly articulated by Roel and Verbauwhede (2010).

Visual Studio has been used to setup the environment for calling the GET function and PUT function to transact data from and to the device. This environment eases the process of determining the maximum frequency difference among a set of 8 configurations on single CRO, as shown in Figure 3.3 Kodỳtek and Lórencz (2015). Table 3.10 shows the resultant file that gives out the challenge and response pairs. The challenge column designates the Most Significant Bits (MSB) consisting of 7

bits indicating the position of CRO, while the Least Significant Bits (LSB) of 3 bits indicates the configuration on each CRO. The block diagram shown in Figure 3.13

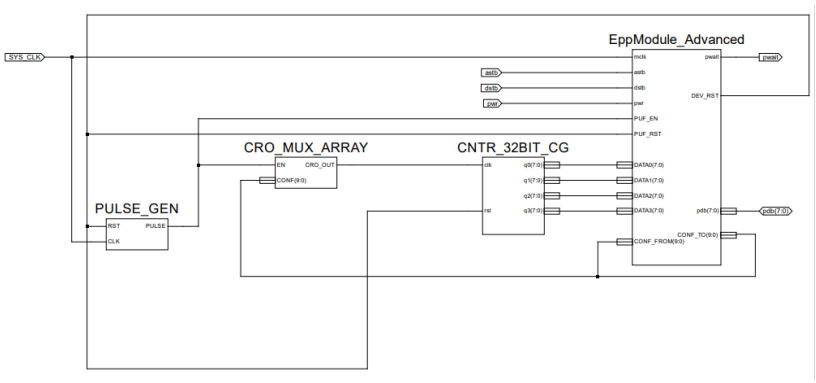


Figure 3.13: Block Diagram of CRO PUF with EPP interface as implemented on Spartan 3E FPGA

explains the communication process between the logic system and the host. Within this system, four data read registers, addressed as 0×00 to 0×03, were used to store data received from a 32 bit counter integrated in RTL. The arrangement designates the segmentation of the 32 bit counter into four distinct 8 bit registers. Initiating the configurable ring oscillator involves the activation of the gate pulse generator (PULSE GEN), which receives a reset signal from the write data register addressed at 0×06, originating from the host. The most significant bit (MSB) within this control register instigates the reset signal. Additionally, the register at 0×07 handles another control function, sending the SELECT LINES and CONFIGURATION BITS to the CRO MUX ARRAY.

When a reset pulse is transmitted to the gate pulse generator (PULSE GEN), it generates an enabling pulse of the required period. This enabling pulse, when connected to the CRO, triggers the oscillation signal of the CRO for the specified period. Consequently, the 32 bit counter commences recording the count value until the enabling signal diminishes. Upon the cessation of the enabling signal, the last latched 32 bit counter value signifies the frequency of the CRO. Awareness of the enabling signal's deactivation within the logic device is required for the host machine to access data through the designated registers. A data read register, specifically addressed at 0×04, is utilized to detect the transition of the enabling signal to a low state. The least significant bit (LSB) from this register signifies the status of the enabling signal,

providing feedback to the host machine when the data is retrieved. Upon detection of the enabling signal's deactivation, the host retrieves the data available in the registers addressed from 0×00 to 0×03. Subsequently, the gate pulse generator is reset once again before it starts to record the second response. The data acquired from the device is transmitted to an Excel file to determine the maximum difference in frequencies between two adjacent locations of configurable ring oscillators. The following section describes the methodology used for comparing the frequencies to generate the desired response.

3.7 Stable Response with proposed methodology

The present study has focused on the establishment of a testing environment built around the modeled PUF circuit and the subsequent analysis of the generated responses. All test configurations were derived from comprehensive literature surveys, forming the basis for recording the responses within the established environment. The primary focus now shifts to the methodology employed to extract random process variations while effectively mitigating the influence of systematic variations. However, the incorporation of an optimal time pulse window has partly addressed systematic variations, and further reduction strategies are to be delineated in the following section.

3.7.1 Extract Unique challenge Response Pairs (CRPs) from the specified FPGA device

As understood from the previous section, the unit time pulse generated to capture a stable frequency from CROs is between one millisecond and one second. Therefore, generating one millisecond pulse facilitates the extraction of a stable response. It is inferred that it is imperative to enroll this response to fix the challenge and response pairs. The next step is the evaluation of frequencies obtained with the frequency meter, a process referred to as enrolling. This intricate procedure involves propagating the variations in their manufacturing process parameters while separating the systematic variations, such as static and dynamic timing characteristics. This section discusses the methodology employed to extract stable responses by evaluating the PUF concerning its placement within the FPGA fabric and the specific techniques for comparing frequencies to yield enhanced performance metrics.

In the FPGA editor's layout (Figure 3.14), a configuration is depicted wherein 128 CROs are situated. These CROs are organized into groups of seven, forming hard macros. Each macro is positioned from SLICE X12Y8 to SLICE X30Y34, as highlighted in red in Figure 3.14, totaling 18 macros and two separate CROs. The positioning of these two separate CROs is at the top middle section. The orientation of the hard macro, represented in black color, signifies a fixed vertical orientation, unalterable upon placement. The rationale behind the selection of seven CROs as hard macros is explained in the subsequent sections, and this grouping approach is derived from a spatial analysis of systematic process variations, detailed further in the following section.

3.7.2 Methodology involved in selecting the CROs to propagate the unique response

The existing literature review has emphasized the significance of considering the geographical placement of CROs concerning the challenge and response pair during evaluation Guajardo et al. (2007). The orientation of the CROs as hard macros needs to be considered. All single CRO hard macros need to connect to the following hard macros with the interconnect matrix. However, an increase in interconnects between hard macros significantly impacts the intra HD results, as these interconnects may potentially dominate logic process delays. Consolidating all 128 CROs into a single hard macro can affect the Inter HD results, potentially resulting in heightened bit aliasing across multiple devices. Therefore, it becomes crucial to identify an optimal grouping of CROs to strike a balanced compromise in the number of CROs grouped together to attain favorable INTER and INTRA HD results.

Moreover, the specific placement of CROs and their comparison location play a pivotal role. Literature Kodỳtek and Lórencz (2015) suggests that comparing adjacent CROs aids in generating a unique response. It was observed that along with adjacent comparisons, it is also important to compare the CROs from two different hard macros but not from the same hard macro. If two CROs within the same hard macro are compared, then hardly any change is expected in their bit responses. The literature analysis infers that the least adjacent CRO is susceptible to common system-

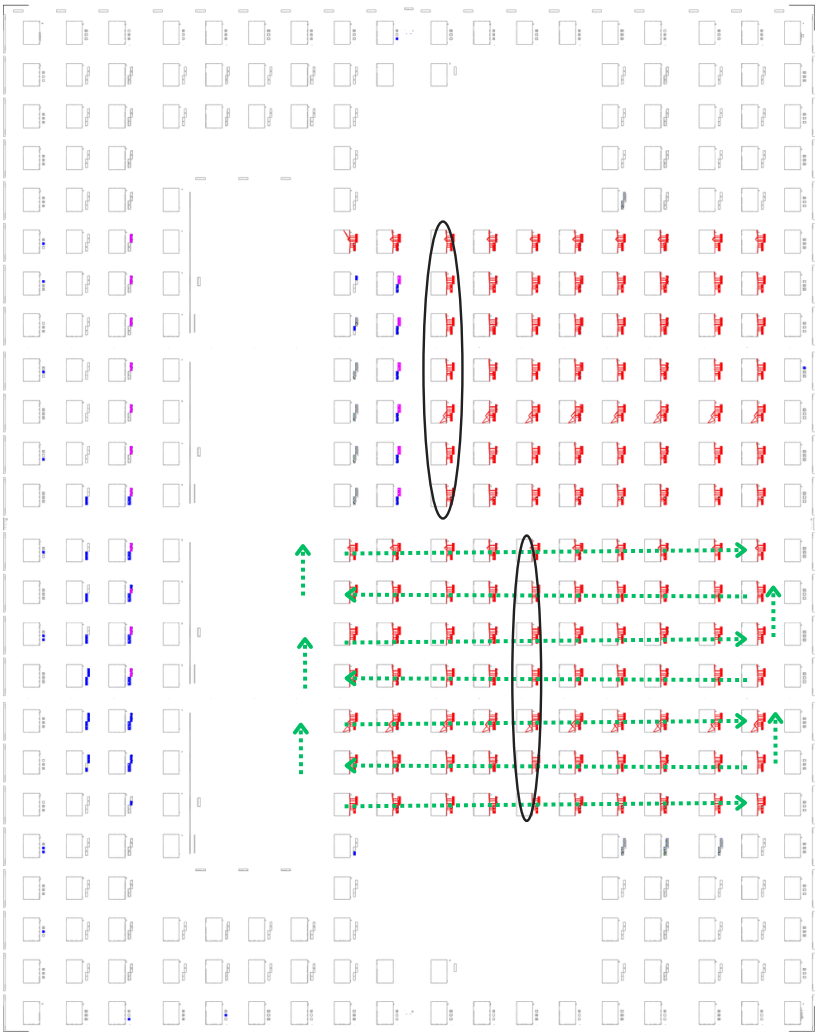


Figure 3.14: 128 CROs and MUX LOGIC Placed as Hard macros

atic processes, resulting in a more stable response. However, this analysis overlooks the grouping of CROs as a hard macro and the variations in the interconnect process, which can significantly affect intra chip evaluations positively but might detrimentally affect inter chip HD evaluations. In order to compensate for both inter and intra HD of the chip, a recommended strategy involves pairing CROs from adjacent locations, but not from the same hard macro. Figure 3.3 illustrates the comparison of frequencies with the same configuration between two CROs Kodỳtek and Lórencz (2015). Frequency with a specific configuration on SLICEX12Y8 is compared with same configuration on SLICEX14Y8, as shown in Figure 3.14. The black arrow indicates the direction of comparison along the horizontal path with the same configuration. A

green arrow to top shows how the flow of this comparison. The alignment of the hard macro in a vertical orientation and the horizontal comparison method align with the compensation strategy proposed by Kodỳtek and Lórencz (2015).

The frequency outputs originating from CROs are correspondingly oriented with respect to the application program for generating the challenge inputs to the PUF evaluation logic to generate the response. Application program generates the challenge bits to the logic through the configuration register. The details of the same are explained in the previous section. A total of 127 bit responses are recorded with 127 pairs of challenges to the PUF circuits. Figure 3.15 shows the experimental setup conducted on four different Basys2 boards. The logic on the device side is implemented in Verilog using Xilinx ISE 14.5, involving manual placement and specific constraints for designing a hard macro to achieve symmetry, detailed in the appendix. An External Perception Processor (EPP) is utilized with a C language-based API in Microsoft Visual Studio 2010. When the logic's bit stream is programmed onto the board, the

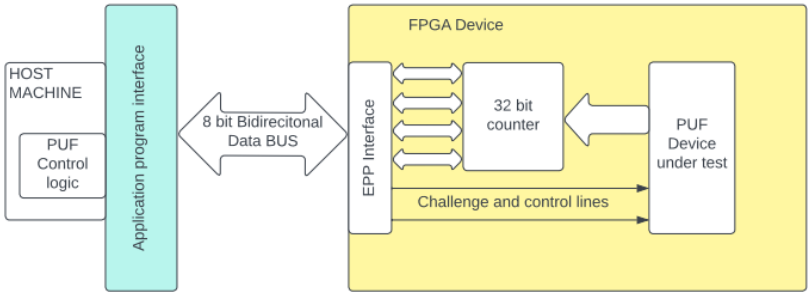


Figure 3.15: Visual Studio API, reading the data from FPGA device through the HIL verification environment

binary generated by the visual studio starts reading data from the device. Figure 3.15 shows the block diagram of the visual studio application programming interface, which reads all 8 frequencies (setting all possible configuration bits on CRO) from a specific CRO placed at a constrained CLB. The data read from the device is in hexadecimal format, as explained in the console, showing the way in which API reads the data through registers (DATA READ from FPGA DEVICE is a 32 bit data from registers addressed at 0×00 to 0×03). While writing this data to a file, it is converted into decimal format. FPGA STATUS indicates the reset signal and configuration bit status, while CONF displays the configuration running on the CRO at that time. The

following section presents the results and analysis of the collected data, focusing on frequencies read using this method.

3.8 Observations and results from the previous sections

3.8.1 Selecting the optimum pulse window to capture the frequencies

The graphical representations in section 3.5 vividly illustrate the methodology of propagating a stable challenge response pairs. Observing Plot 1 in Figure 3.10, it is apparent that employing a one-microsecond unit time pulse window for the ring oscillator significantly increases susceptibility to noise from the interconnect matrix, resulting in a more random response. This behavior of the PUF would tend to only extract the delay of the PUF from the fabric, but it would not be sufficient to determine the process variation pattern on the fabric and hence it fails to generate a stable and unique result out of the PUF.

Conversely, Plot 2 in Figure 3.11 and Plot 3 in Figure 3.12, utilizing pulse windows of one millisecond and one second respectively, yield a consistent frequency output. This substantiates that the resultant response is a consequence of process variations within the fabric. Furthermore, the plots indicate that nearby composite ring oscillators (CROs) exhibit similar noise effects Sedcole and Cheung (2006). Consequently, extracting data from adjacent CROs mitigates the impact of temperature on the response.

Section 2.2.3 details a proposed methodology aimed at obtaining a stable response by amalgamating an optimal number of CROs as hard macros, allowing control over process variations specific to the device's fabric, as outlined in the work by Jeeru et al. (2019). To validate this approach, a comparison is made between two methods: one involves generating the PUF response using a single CRO as a hard macro in a PUF device, while the other entails grouping CROs as a hard macro (in this case, a group of 7).

3.8.2 Results from two methods incorporated for enrolling the PUF response

The literature review (Section 2.1.5) Maiti et al. (2013) finalizes the parameters to be measured, delineating four metrics showcased in Sections 3.8.3 to 3.8.6 along with their respective definitions. The quantification of these four parameters is presented in equations 3.3 to 3.7. The results and analysis are subsequently discussed. Notably, attack resistance, another metric, is discussed in the following chapter, given the proposed architectural enhancements to the existing design to evaluate the augmented outcomes of this specific metric. Hence, the methodologies employed in eliciting the PUF response are evaluated against the four parameters, compared with the work presented in the literature by Maiti et al. (2013). These methodologies represent two approaches utilized in enrolling the PUF response.

- PUF response extracted from Macro with a Group of CROs (Methodology adopted from this research)

- PUF response extracted from Macro with a single CRO (Method followed conventionally in all literature survey papers)

The analysis of response bits generated by 'k' chips, each 'n' bits wide and produced 'm' times, involves the assessment of three distinct dimensions to evaluate their metrics. When a random 'lth' bit is scrutinized in the 'n'-bit response, it can be expressed as follows: $Response_{i,l,t} = t_{th}$ sample of l_{th} response bit of i_{th} chip

where $1 \leq i \leq k; 1 \leq l \leq n; 1 \leq t \leq n$

3.8.2.1 Uniqueness

If there are k chips and if $chip_i$ generates response R_i and if $chip_j$ generates response R_j then, Inter Chip variation as defined by Hamming distance between a pair of PUF identifiers, is given by following equation , as discussed already in previous chapter Hamming distance is bit wise exclusive or of all n bit response from $chip_i$ and $chip_j$. Ideally this should be 0.5(section 2.3.2, Figure 2.9) the quantitative parameters that calculates the uniqueness is shown in eq 3.3

Uniqueness in terms of Hamming distance between two chips

$$\textbf{Uniqueness} = \frac{2}{k(k-1)} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \frac{HD(R_i, R_j)}{i} X100\% \tag{3.3}$$

3.8.2.2 Reliability

PUF reliability denotes the efficiency with which a PUF reproduces response bits. Reliability is assessed through the Hamming distance between various samples of PUF response bits. Ideally, this value should be 0%. Equations 3.4 and 3.5 outline the quantitative parameters that calculate the reliability.

Reliability in terms of Hamming distance several samples on the same chip

$$HD_{INTRA} = \frac{1}{m} \sum_{t=1}^{m} \frac{HD(R_i, R'_{i,t})}{n} X100\% \tag{3.4}$$

$$\textbf{Reliability} = 100\% - HD_{INTRA} \tag{3.5}$$

3.8.2.3 Uniformity

The uniformity of a PUF characterizes the proportion of '0s' and '1s' within its response bits. It quantifies the uniformity of '0s' and '1s' within the PUF response. Ideally, this value should be 50%. Equation 3.6 presents the quantitative parameters for calculating uniformity, which is an intra-chip measure.

$$\textbf{(Uniformity)}_{\textbf{i}} = \frac{1}{n} \sum_{l=1}^{n} r_{i,l} X100\% \tag{3.6}$$

where $r_{i,l}$ is the l_{th} binary bit of an n-bit response from chip i.

3.8.2.4 Bit aliasing

In cases where bit-aliasing occurs, different chips may generate nearly identical PUF responses, an undesirable outcome. Equation 3.7 provides quantitative parameters for

assessing and quantifying bit aliasing.

$$\textbf{(Bit-aliasing)}_{\mathbf{i}} = \frac{1}{k}\sum_{i=1}^{k} r_{i,l} X 100\% \tag{3.7}$$

where $r_{i,l}$ is the l_{th} binary bit of an n-bit response from chip i.

Table 3.1: Signature of 127 bits for specific challenge bits

Macro with Group of CROs	Responses
DEV-1	E15AA474AE1555B8593955F54D7468
DEV-2	E165A59F565764B0BA38A24C75AA562A
DEV-3	C11BA1255A5554BB39724C6354697A
DEV-4	E175AD0A5755B0B129224D2B157732

The experimental setup utilizes an Enhanced Parallel Port interface as hardware within a loop verification environment to analyze the PUF circuit, particularly concerning its challenge and response pairs. It is observed that the utilization of a hard macro for the designed CRO array reduces noise within the interconnect logic, allowing the process variation factor to dominate. Plots 1, 2, and 3 in Figures 3.10, 3.11, and 3.12 demonstrate that peak points represent random variations in parameters, while slopes signify systematic process variations.

Using an array of CROs configured as a hard macro yields improvements in both inter and intra Hamming distance when compared to a single CRO configured as a hard macro. This observation is a key finding in this study, suggesting that Hamming Distance serves as an indicator of the uniqueness of the generated responses. Equations 3.3 to 3.7 outline the methodology for calculating both inter and intra Hamming distances for the responses. The FPGA captures the responses, which are then generated into signatures by the host machine. This approach is an integral part of the methodology for propagating the bits without introducing bias.

Hamming Distance results when a group of CROs are made as hard macro

Table 3.1 depicts distinct responses specific to individual devices, with a subset of exemplary challenge and response pairs showcased in Table 3.8. It comprises an extensive database housing the challenge and response pairs for all devices, while this work primarily depicts only 64 samples of device-1. The responses in Table 1 are presented in hexadecimal form, with the challenges duly recorded. Table 3.1 selectively exhibits the response bits in focus.

Table 3.2: Hamming Distance between the responses generated at random times T0, T1, and T2

Reliability	**INTRA(T1,T2)**	**INTRA(T0,T1)**	**INTRA(T0,T2)**
DEV-1	0	0	0
DEV-2	2	3	1
DEV-3	2	0	2
DEV-4	2	0	2

Table 3.3: Hamming Distance between 4 devices, ((N*N-1)/2 = 6)

INTER HD(Uniqueness and Bit aliasing						
(D1,D2)	**(D1,D3)**	**(D1,D4)**	**(D2,D3)**	**(D2,D4)**	**(D3,D4)**	**AVG**
48	42	52	42	46	40	45

Tables 3.2 and 3.3 present the INTRA HD and INTER HD of responses obtained from the PUF circuit utilizing the Enhanced Parallel Port Interface. This approach aims to cluster an optimal number of CROs while comparing the adjacent CROs beyond the hard macro. Table 3.3 distinctly reveals the uniqueness concerning devices - 1, 2, 3, and 4. The values in Table 3.3 indicate the bits differing from other devices under comparison, illustrating proportional uniqueness and bit aliasing metrics of the PUF. Table 3.4 demonstrates the entropy corresponding to the generated responses, ideally targeting a 50 percent value. The idle value for a 128-bit response is 64. On average, the Hamming distance approaches 62, a near-ideal outcome. The values presented in the table align with the uniformity metrics proposed by Stanciu et al. (2016) for the PUF.

Hamming Distance results when a single CRO is made as hard macro

In summary, this chapter explains the Enhanced Parallel Port Interface, which simplifies the experimental configuration for PUF assessment. The rationale behind

Table 3.4: number of 1s in the 127 bit response that's generated

Uniformity in terms of Hamming Distance				
DEV-1	**DEV-2**	**DEV-3**	**DEV-4**	**AVG**
60	64	58	66	62

the grouping of CROs as hard macros has been validated with the setup explained in the previous sections. Previous sections explain the results achieved through CRO grouping, whereas this section summarizes the results in comparison with the single CRO hard macro. Table 3.5 shows the response generated for specific challenges and sample challenge and response pairs are shown in Table 3.8. It is evident from Tables 3.4 and 3.6 that INTER HD has been improved when grouping of CROs as hard macro is considered and the adjacent hard macros are compared. While consistent values reflect response uniformity, this work primarily emphasizes uniqueness and bit aliasing. The method followed in this work enhances the uniqueness with respect to multiple devices. Moreover, expanding the number of CROs to 256 and beyond is projected to yield substantial improvements in INTRA and INTER HD results, aligning with the technique of CRO grouping Stanciu et al. (2016).

Table 3.5: Response of 127 bits for specific challenge bits

Single CRO as Hard macro	**Responses**
DEV-1	6CA9B364C781122D35532A9CB5126A56
DEV-2	26E935726693322AAD576A26D335BA45
DEV-3	6EF917A26F913212E9532A2A9132E253
DEV-4	2D4C53666F919A2AD9532E6AD534A655

Table 3.6: Hamming Distance between 4 devices, ((N*N-1)/2 = 6)

INTER HD(Uniqueness)-Single CRO as Hard macro						
(D1,D2)	**(D1,D3)**	**(D1,D4)**	**(D2,D3)**	**(D2,D4)**	**(D3,D4)**	**AVG**
43	38	43	31	36	31	37

It is also evident that efficiently structured PUF Circuit would enhance the effect on the Inter and Intra HD results of the configurable ring oscillator. Table 3.9 summarizes the improvement with respect to uniqueness of the CRO PUF, when analyzed for an optimum unit time pulse in generating a consistent challenge and response pairs.

A significant improvement is observed in uniqueness when optimum grouping of CROs is considered in comparison with conventional PUFs with a single CRO as hard macro. The performance metrics were quantitatively analyzed for the parameters defined by Stanciu et al. (2016). It talks about various ways in which different authors had quantified the metrics. The date presented in Table 3.9 is evaluated from the method established by Roel and Verbauwhede (2010).

Table 3.7: Summary of the performance metrics with respect to two different methodologies mentioned here comparison results in percentage

		PUF with Individual CRO as a Hardmacro	PUF with Group of CROs as a Hardmacro	Ideal Value
Average Uniqueness with respect to 4 devices (%)	D1 to D2 D1 to D3 D1 to D4 D2 to D3 D2 to D4 D3 to D4	28.9589	35.7830	50
Reliability (%)	Device 1	98.4252	100	100
	Device 2	98.9501	99.2126	100
	Device 3	98.9501	98.9501	100
	Device 4	99.4751	98.9501	100
Uniformity (%)	Device 1	46.7192	47.2441	50
	Device 2	50.3937	49.8688	50
	Device 3	48.0315	46.1942	50
	Device 4	50.6562	51.9685	50

This study elucidates the diverse impacts of efficiently structured PUF Circuits on the performance of configurable ring oscillators, displaying the critical improvements in uniqueness and the comprehensive analysis of performance metrics adopted from relevant literature.

3.9 Summary

The proposed methodology for designing and validating a PUF circuit involves modelling a delay-based PUF circuit, designing an array of 128 CROs, and performing delay analysis on a Configurable Ring Oscillator PUF with a frequency meter. The methodology also involves designing a frequency counter for Configurable Ring Oscillator and identifying the optimum pulse window to capture the stable frequency from the PUF.The Experimental setup for hardware in Loop (HIL) Verification environment is developed as part of the research and is used to verify the stable response with

the proposed methodology. The methodology involves extracting unique challenge response pairs (CRPs) from the specified FPGA device (3-series device) and selecting the CROs to propagate the unique response.

The observations and results from the previous sections include selecting the optimum pulse window to capture the frequencies, results from two methods incorporated for enrolling the PUF response, and the uniqueness, reliability, uniformity, and bit aliasing of the results.

The scientific contribution of this research lies in the comprehensive exploration and optimization of the Ring Oscillator Physically Unclonable Function (RO PUF) design within the context of FPGA architecture. While the concept of RO PUFs is established, the novelty in this research is manifested in the intricate understanding and manipulation of factors influencing PUF behaviour. The study delves into the placement of RO PUFs on the FPGA fabric, considering the nuances of FPGA architecture and the impact of adjacent logic. Furthermore, the investigation into process variations, encompassing both random and systematic variations, represents a significant scientific contribution. The research not only characterizes the distribution of these variations across the FPGA silicon but also explores their implications on the reliability and uniqueness of PUF responses. By addressing these multifaceted aspects, the research not only advances the understanding of RO PUFs in the specific context of FPGA but also contributes valuable insights to the broader field of hardware security and digital design. The scientific merit of the work lies in its holistic approach, bridging the gap between theoretical PUF concepts and their practical implementation within the intricacies of FPGA technology.

Scientific Contribution in terms of the domain areas of research:

- Optimized RO PUF Design:
  The research introduces an optimized design of Ring Oscillator Physically Unclonable Function (RO PUF) within the FPGA context, showcasing innovation in the implementation of this well-established cryptographic primitive.

- FPGA Architecture Considerations:
  Scientific merit is evident in the meticulous examination of FPGA architecture,

considering factors such as configurable logic blocks (CLBs), interconnect resources, and routing structures. The research optimizes the placement of RO PUFs to enhance signal integrity and minimize delays.

- Impact of Adjacent Logic:
  The study recognizes the influence of adjacent logic on RO PUF performance. Scientifically, it explores and quantifies how neighbouring components within the FPGA affect the behaviour of the PUF, providing insights into reliability and security considerations.

- In-Depth Analysis of Process Variations:
  A significant scientific contribution lies in the detailed analysis of both random and systematic process variations. The research characterizes the distribution of these variations across the FPGA silicon, advancing our understanding of how they impact the uniqueness and reliability of PUF responses:

- Holistic Approach to Hardware Security The research takes a holistic approach by seamlessly integrating theoretical PUF concepts with practical considerations in FPGA technology. This contributes not only to the specific domain of RO PUFs but also enriches the broader field of hardware security and digital design.

- Advancement of Practical Implementation

By bridging the gap between theoretical concepts and practical implementation, the research advances the state-of-the-art in RO PUFs, making a tangible contribution to the scientific and practical aspects of hardware security.

The methodology proposed is the outcome of the above mentioned points. This method ensures the response to be stable even without error-correction mechanisms. The conventional approach on delay based PUFs as discussed in literature survey from previous chapter, will need a error correction techniques to have a stable response. The methodology shown in this chapter enables the PUF to generate consistent performance metrics as discussed in section 3.8.

Following chapter starts with strong PUF(Arbiter PUF) design and construction. A new Arbiter PUF is proposed utilising the non-linear characteristics from ring os-

cillator, to strengthen the PUF for having a good attack resilience. It combats the modelling attacks that come up with arbiter PUF, for its linear characteristics.

Table 3.8: : Challenge and Response Pair for Device -1 (only 64 samples are shown in the table)

Challenge					Challenge				
SEL[9:3]	SEL[2:0]	SEL[9:3]	SEL[2:0]	DEV1	SEL[9:3]	SEL[2:0]	SEL[9:3]	SEL[2:0]	DEV1
0	5	1	5	1	32	7	33	7	1
1	6	2	6	1	33	7	34	7	0
2	0	3	0	1	34	3	35	3	1
3	0	4	0	0	35	3	36	3	0
4	5	5	5	1	36	0	37	0	0
5	6	6	6	0	37	2	38	2	1
6	7	7	7	0	38	7	39	7	1
7	5	8	5	1	39	6	40	6	0
8	5	9	5	0	40	1	41	1	0
9	4	10	4	0	41	0	42	0	0
10	3	11	3	0	42	7	43	7	0
11	3	12	3	1	43	7	44	7	1
12	2	13	2	1	44	6	45	6	1
13	3	14	3	0	45	7	46	7	0
14	7	15	7	1	46	5	47	5	1
15	7	16	7	0	47	5	48	5	0
16	7	17	7	1	48	3	49	3	0
17	6	18	6	0	49	7	50	7	1
18	6	19	6	1	50	2	51	2	0
19	4	20	4	0	51	2	52	2	1
20	2	21	2	0	52	1	53	1	0
21	7	22	7	1	53	3	54	3	0
22	1	23	1	1	54	4	55	4	0
23	5	24	5	0	55	4	56	4	1
24	2	25	2	0	56	7	57	7	1
25	3	26	3	1	57	6	58	6	0
26	1	27	1	0	58	6	59	6	1
27	5	28	5	1	59	4	60	4	1
28	6	29	6	0	60	5	61	5	0
29	7	30	7	1	61	7	62	7	1
30	7	31	7	0	62	7	63	7	0
31	3	32	3	0	63	0	64	0	0

Chapter 4

Proposed Arbiter PUF and Attack Modeling

Thus far, the developed PUF is regarded as an application for generating a signature and has validated methods for its enrollment. This chapter explores PUF as a low cost authentication device and its application for securing smart meter systems in an IoT environment. In order to attain this objective, a strong PUF (as defined in literature and explained in Chapter 1, section 1.1) is needed. An Arbiter PUF (APUF) is the chosen solution for generating an adequate number of challenge and response pairs for authentication.

4.1 Arbiter based Physical Unclonable Function(APUF)

The APUF operates by accepting n-input challenges (C) and furnishing a singular output as the response (R). APUF was the first design proposed within silicon-based PUFs. The core principle of the PUF design revolves around creating a race condition between two signal paths, where the input challenge triggers the generation of a single-bit response output. The functionality of this design relies on the delay variances across the chip. During chip fabrication, manufacturing variations significantly impact the physical parameters, consequently influencing the precise delay in each path and introducing slight random deviations between the two delays.

The PUF circuit contains 'n' consecutive multiplexer blocks and an arbiter element, typically a flip flop. Each multiplexer block contains two 2-input multiplexers with a

common select bit for both. The inputs are provided to these selected bits. Based on the logic value at the select line, the signal will travel through the respective path. This traversal involves delays based upon process variations along that path. As seen from

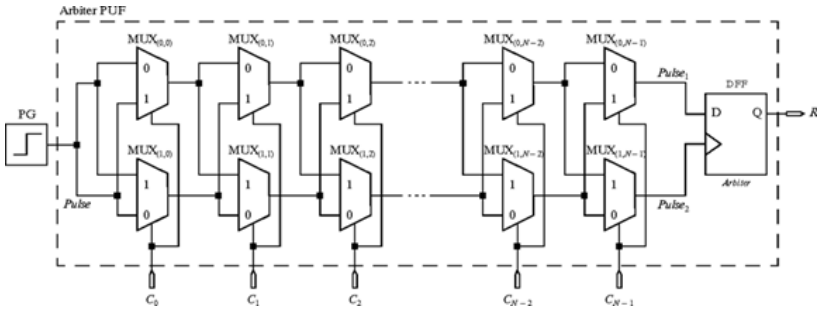


Figure 4.1: Arbiter PUF circuit

Figure 4.1 Ye et al. (2016), there are four multiplexer switch blocks, where each block's outputs act as inputs to the successive block in a cascaded manner. These outputs act as inputs to the arbiter element during the final stage. In the initial stage block, the inputs are interconnected and activated by an enable signal. Consequently, a pulse signal travels through the two paths. Typically, the two paths should be similar, but subtle differences are introduced due to the fabrication variations, resulting in a delay in reaching the final element. Ultimately, the D flip flop determines the output bit as 0 or 1 based on the arrival of path arrivals at the input.

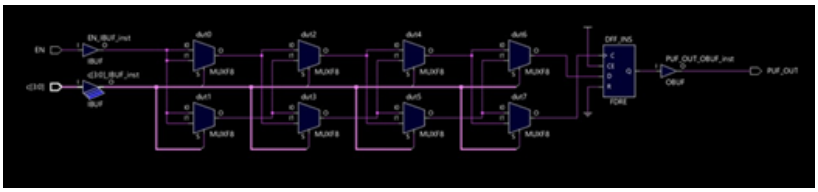


Figure 4.2: Schematic of 4-bit Arbiter PUF in a CLB with single response per CLB

4.1.1 Components of PUF Circuit

The components of arbiter PUF are multiplexer instances and D-type Flip Flop (Figure 4.2). The multiplexer instances have two inputs, outputs, and common control pin. When logic "0" is applied as input to the control pin, a direct connection path is established between the input and the output, indicating that the signal is on the same path. Conversely, when logic "1" is applied as input to the control pin, the connection is cross coupled, causing the input signals to interchange their paths. Figure

4.3 illustrates these operations within the multiplexer block component Kumar and Niamat (2018). An arbiter element is a positive edge triggered D-type flip flop and it

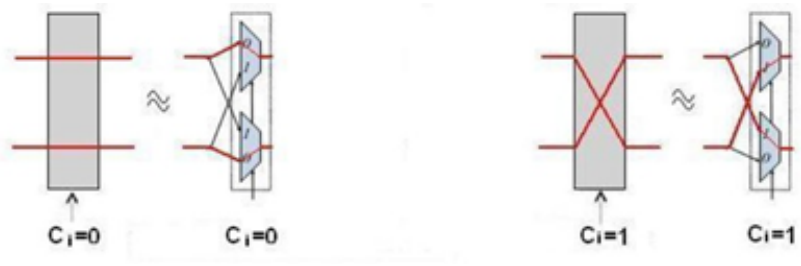


Figure 4.3: Multiplexer block components operation

has two input and one output pins. When the rising edge appears on the clock, the logic value of output Q is set based on the input D. The resultant logic states persist at the output until the next transition in the clock input. The arbiter element has to

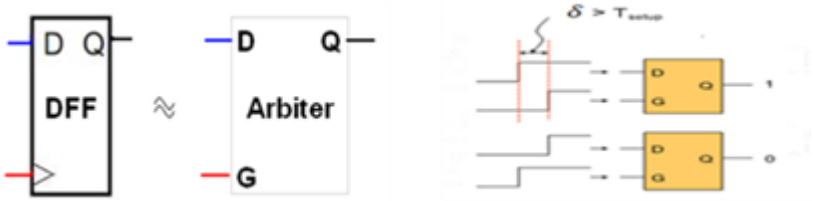


Figure 4.4: Operation of arbiter element

satisfy the setup and hold time. As shown in Figure 4.4, the flip flop will work when the input D arrives before the setup time and stabilizes after the hold time. In a worst case scenario, the signal will delay due to the setup and the "clock to Q" delay of the flip flop, as shown in Figure 4.5.

In a PUF circuit, the output Q will be logic '1' when the delay difference between input signals is more than the setup time (T_setup); otherwise, the response will be logic "0". In general, the circuit's design and implementation provides challenge response pairs in an exponential manner, i.e., for a given 'n' number of challenges, for a case like n=64-bit challenges will generate 264 CRPs. The possibility of exponential number of CRPs makes the design difficult to predict, model, and duplicate.

4.2 Modified APUF for stronger resistance:Inference from literature

In the domain of enhanced security for Physically Unclonable Functions (PUFs), the literature introduces innovative modifications to the conventional Arbiter Physically Unclonable Functions (APUFs) to bolster their resistance against potential vulnerabilities. One notable advancement is the Priority Arbiter-based PUF (PA-PUF), as outlined in Singh et al. (2022). The PA-PUF revolves around the utilization of a 3-input priority arbiter, skillfully constructed with a simple arbiter, two 2:1 multiplexers, and an XOR logic gate. This design not only ensures excellent uniformity, with an equal probability of 0s and 1s at the output, but also allows for configurability in challenge-response pairs through the integration of multiple feed-forward priority arbiters. The reliability post-error correction techniques reaches an impressive 100%, emphasizing the efficacy of this modified APUF. In a separate endeavor documented in Sahoo et al. (2015), researchers focus on Programmable Delay Line (PDL)-based APUF design, with a specific emphasis on leveraging the Hard Macro feature. This feature proves to be of paramount significance, providing predefined and characterized logic blocks that contribute to design consistency, elimination of bias, and overall hardware efficiency. The proposed design methodology systematically eliminates bias sources in PDL-APUF, resulting in superior implementations compared to previous approaches. Experimental evaluations on Spartan-3 FPGAs showcase improved uniformity, reliability, and uniqueness. Collectively, these modified APUFs, namely PA-PUF and PDL-APUF with Hard Macro Features, represent substantial advancements in design methodologies, addressing bias-related vulnerabilities and ensuring robust and reliable PUF implementations in cryptographic applications. In the context of enhancing the security of Physically Unclonable Functions (PUFs), one promising area for further research and proposal development is the refinement and expansion of the challenge space. The challenge space in PUFs refers to the set of possible input challenges applied to the PUF to generate corresponding responses. A more robust and diversified challenge space can contribute to increased security and resistance against various attacks, including modeling attacks and machine learning-based attacks.

Based on the literature discussed, particularly in the context of modified Arbiter Physically Unclonable Functions (APUFs), here are potential areas to work on and which lead to the proposal of new Arbiter PUF

Dynamic Challenge Space Exploration:
Propose methods to dynamically vary the challenges applied to the PUF during operation. Dynamic challenge spaces can add an additional layer of complexity, making it more challenging for attackers to predict and model PUF responses accurately.

Multi-Dimensional Challenge Spaces:
Investigate the use of multi-dimensional challenges, where challenges are not limited to a single dimension but involve combinations or sequences of inputs. This approach could enhance the entropy of the challenge space, making it more difficult for attackers to exploit patterns.

Temporal Variations in Challenges:
Explore the introduction of temporal variations in the challenge space, where challenges change over time during PUF operation. This temporal evolution can introduce unpredictability and prevent attackers from relying on static models.

Machine Learning-Resistant Challenge Designs:
Devise challenges that are specifically designed to resist machine learning-based attacks. This could involve exploring challenge patterns that are inherently challenging for machine learning algorithms to capture or predict.

Nonlinear Challenge Relationships: Investigate the introduction of nonlinear relationships between challenges and responses. By incorporating nonlinearities, the challenge space becomes more resistant to linear modeling attacks commonly employed by adversaries.

Adaptive Challenge Strategies:
Propose adaptive challenge generation strategies that respond to the evolving threat landscape. This could involve integrating feedback mechanisms that adjust the challenge space based on detected attack patterns or vulnerabilities.

Hardware-Informed Challenge Designs:
Leverage insights from hardware characteristics, such as delay paths and logic gate structures, to inform the design of challenges. This hardware-aware approach can lead

to challenge spaces that exploit the unique features of the underlying PUF architecture.

By focusing on these aspects of challenge space improvement, the proposed research can contribute to advancing the security of PUFs, making them more resilient to a wide range of attacks, and ensuring their effectiveness in cryptographic applications.

4.3 Proposed Design

As previously discussed in Chapter 1, the conventional arbiter PUF design is secure and the possibility of generation of exponential number of challenge response pairs makes the prediction of the design challenges difficult. There are lot of modification on Arbiter PUF proposed by different authors. They are discussed in detail in chapter-1 literature survey, in section 1.6.1. However, due to the advancement of several computational algorithms, the design could be possibly breached with a good prediction rate. Studies by Ye et al. (2016) and Kumar and Niamat (2018) have illustrated that with the support of machine learning algorithms, such as logistic regression, support vector machine (SVM), and specific artificial neural network algorithms Design (2018), the practical responses could be reproduced with a high percentage of prediction rate for existing PUF designs, including the conventional Arbiter based PUF design circuit. Consequently, proposed modifications aim to fortify the existing design, enhancing its resilience against attackers aiming to model the design for predictive purposes. Before going ahead to discuss the proposed structure, a comprehensive grasp of the methodology outlined in Chapter 2, section 2.3.1 is necessary. The more area the PUF device occupies on the fabric, the more stable and enhanced results would be assured. This is because it gets an opportunity to explore the process variations widely. This quantitative analysis, as detailed in Chapter 2, underscores the pivotal role of spatial allocation. Thus, for a CRO shown in Chapter 2 or an Arbiter PUF shown in Figure 4.1, both of these occupy on a single CLB structure. Chapter 3 explains how the incorporation of more CROS into a single hard-macro affects the resulting responses. In a similar way, the process variation results could be enhanced if more cores can be accommodated in a single CLB, thereby enriching the functionality of process variation derived from the device fabric.

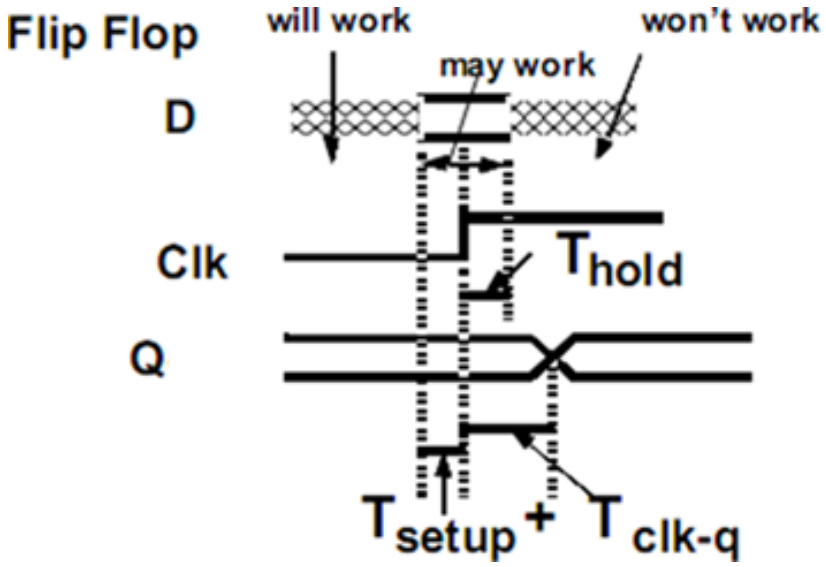


Figure 4.5: Flip-Flop timings

The concept of architectural enhancement pertains to the augmentation of the number of UNIT PUF circuits integrated within a singular CLB. Specifically, leveraging the capabilities of Artix-7 or any 7-series device streamlines this endeavor, given their fabrication with advanced technology. Consequently, these FPGAs can accommodate an increased number of functional building blocks or components on a single CLB, thus facilitating the inclusion of more diverse functionalities derived from process variations within the device fabric.

The proposed modification aims to fortify the security of the arbiter PUF by enhancing its area utilization capabilities. One of the primary modifications involves the generation of an equal number of CRPs, a departure from the conventional setup where only one response is generated for a given challenge in a configurable ring oscillator, as discussed in the previous chapter. This enhancement is aimed at producing an N-bit response for an N-bit challenge, thereby increasing the resilience of the design.

In order to contextualize this design in the Artix-7 CLB structure (Figure 3 in the Appendix), it is essential to note that each CLB comprises two logic slices, with each slice housing four 6-input LUTs accompanied by 8 flip-flops (as illustrated in Figure 5 in Appendix A.3.1). Constructing a block of the Arbiter PUF (4-bit challenge) necessitates four LUTs per block. This configuration implies that a single CLB

can accommodate four such Arbiter PUFs. In contrast, the Spartan-3e architecture contains four slices within one CLB, each with 2 LUTs. Consequently, in Spartan-3e, only one PUF can be housed within a CLB. Therefore, this architectural advancement augments the Arbiter PUF's strength by increasing the number of CRPs. The heightened count of CRPs extends the time required for model prediction significantly.

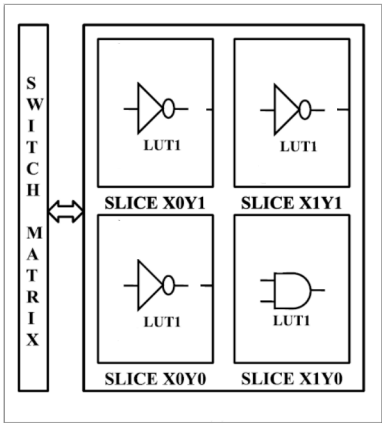
In order to illustrate the proposed work, Reference Figure 2 demonstrates the manner in which the LUT is utilized for generating responses with more number of challenges compared to the one in Reference Figure 3. This exploration delves deeper into the fabric, improving the configurability of the design and making it stronger. The same explanation is extended to Figure 4.2 and Figure 4.6. Figure 4.2 shows a conventional arbiter PUF that generates single response from four challenges from a single CLB. However, the proposed PUF in Figure 4.6 generates four challenges and four responses from the same CLB, thereby improving the PUF's efficiency by four times. This is because when the conventional model is modeled through a machine learning algorithm, it would require lesser time to obtain its model; where as in this, it needed more time to reach a considerable percentage of prediction. This is observed by incorporating a logistic regression algorithm on the CRP set of designed arbiter PUF.

By allowing some percentage of CRPs to train the model and the remaining CRPs to test the model, the results showed a significant a marked decrease in predictability for the proposed arbiter PUF. The implementation details of how it is mapped to the targets (Spartan 3e and Artix-7 devices) is given in Appendix A.1 for Spartan-3e implementation and A.2 for Artix 7 implementation.

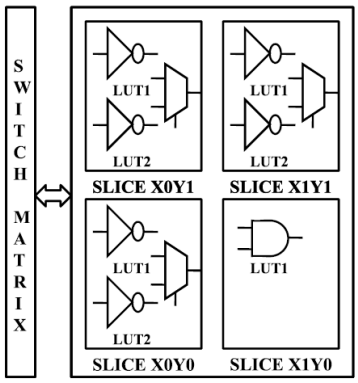
The following section analyses the proposed Arbiter PUF, which is designed and implemented for its resistance against attacks. The figures presented below are reproduced from Chapter 3.

4.4 Attack Studies and Resistance

From section 1.8.2 , the literature inferences on various non-invasive attacks are drawn as below



Reference Figure-1:Inefficient utilisation of CLB



Reference Figure-2:Efficient utilisation of CLB

Non-Invasive Attacks and Model Building Attacks:

Non-invasive attacks, particularly model-building attacks, are considered highly feasible and cost-effective. These attacks involve accessing the device's interface to measure Challenge-Response Pairs (CRPs) and building a numerical model of the PUF. Techniques for non-invasive attacks include Machine Learning (ML), side-channel, and hybrid approaches combining side-channel and ML methods. Electromagnetic (EM) emissions are highlighted as a source of information leakage in common PUF architectures, making them vulnerable to side-channel attacks.

Machine Learning-Based Attacks:

ML-based attacks, especially using techniques like Support Vector Machines (SVM), Logistic Regression (LR), and Evolution Strategies (ES), are found to be applica-

ble and effective against strong PUFs. ML techniques are used to predict responses of Arbiter PUFs and their derivatives, with countermeasures proposed to introduce non-linearity. These attacks are demonstrated to be accurate, even when validated with measurements from Field-Programmable Gate Arrays (FPGA) and Application-Specific Integrated Circuits (ASIC).

ML-Based Attacks on RFID Tags:
ML-based attacks on PUF-based RFID tags are discussed, where attackers, with access to primary inputs, perform non-invasive measurements to collect CRPs. These attacks are successful, leading to the cloning of PUF tags. Software reverse engineering is utilized to discover internal configurations, enabling ML-based attacks using techniques like Logistic Regression.

Machine Learning Attacks on Bistable Ring PUFs:
A study investigates ML attacks on Bistable Ring PUFs, focusing on the challenge when the underlying mathematical model is unknown. The study introduces a Probably Approximately Correct (PAC) learning framework, demonstrating the effectiveness of ML algorithms, particularly boosting, on Bistable Ring PUFs implemented on FPGAs. The study challenges the assumption of equal determination by each bit in PUF responses.

Side-Channel Attacks:
Side-channel attacks exploit imperfections in PUF responses, such as repeatability imperfections in Arbiter PUFs and remanence decay in SRAM-PUFs. Repeatability measurements are used to perform model-building attacks on Arbiter PUFs with high prediction accuracy. Remanence decay in volatile memory is exploited to recover the PUF response in SRAM-PUFs, although this attack is limited by timescale and precision constraints.

Hybrid Attacks Combining Side-Channel and ML Techniques:
Hybrid attacks combining side-channel information and ML techniques are proposed to overcome limitations in ML techniques as the complexity of PUFs increases. Power and timing traces are used as side-channel information to successfully attack XOR Arbiter PUFs and Lightweight PUFs, demonstrating high accuracy and polynomial training time.

Multiplexer-Based APUF Composition (MPUF): A novel solution, Multiplexer-based APUF Composition (MPUF), is introduced to enhance the security and reliability of Arbiter PUFs. Two variants, cMPUF and rMPUF, are proposed to address modeling attack vulnerability and reliability issues. The rMPUF shows improved resilience against reliability-based modeling attacks compared to XOR APUFs, offering a secure and reliable alternative in practical applications.

These summaries provide insights into various attack models, countermeasures, and vulnerabilities associated with PUFs, showcasing the diverse approaches and challenges in securing PUF-based systems.

Attack Resistance Maiti et al. (2013) discussed attack resistance measurement methods in their work. The understanding of the measurement of attack resistance initiates with a discussion about the assessment of PUF. A PUF is considered, characterized by challenge-response pairs (CRPs) of considerable strength, generating N-bit challenges and equally robust N-bit responses. Authentication via this robust PUF involves the exchange of these CRPs to validate the identity of an authorized user engaging in a transaction. An authorized member has the freedom to access 'm' challenges from a set of 'n', where 'n' represents a substantially large number and 'm' is significantly smaller than 'n'. Over time, the authorized access expires, limiting the member's service access. Subsequently, with a known set of 'm' CRPs, an unauthorized member might predict the transaction response, termed as a PUF attack. This vulnerability is exploited using state-of-the-art Machine Learning (ML) algorithms, particularly neural network-based ML algorithms with various activation functions to forecast behavior and responses—a process known as attack modeling. This model development can occur even in a black-box scenario. This section is about the resistance to this parameter of PUF. Rührmair et al. (2010) and Ye et al. (2016) explained the methodologies involved in these aspects. The response of a PUF is confined to a binary data space, typically returning 0 or 1. Understanding the functionality of a black box, with unknown inner workings, typically involves observing output patterns concerning the provided inputs. The efficacy of prediction improves with a larger number of unique input trials. In order to predict a black box's behavior, two crucial elements come into play: the training space and the testing space within the system

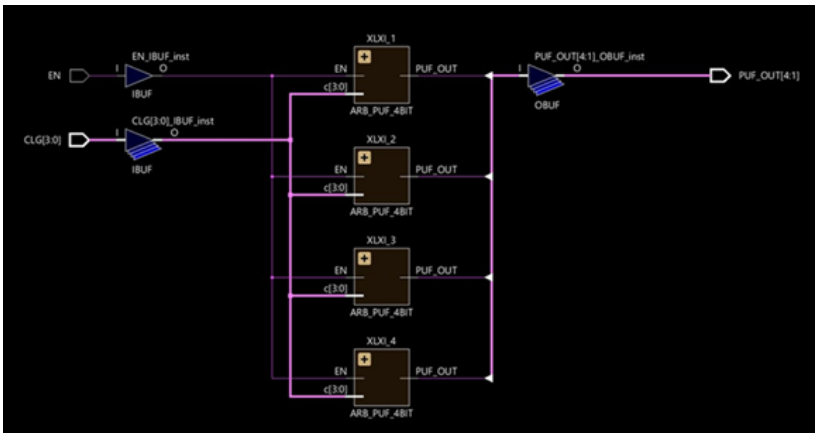


Figure 4.6: Schematic of 4-bit proposed design-with four responses per CLB

(black box). In this context, if an attacker has gained access to a hundred CRPs, it becomes feasible to observe the functionality represented by f(x)=y, where 'x' signifies the challenge and 'y' indicates the response. If, for instance, 80 out of the 100 challenges are employed to predict the behavior of f(x) and model its performance, this constitutes training the model with 80% of the CRPs. Upon assuming that the model accuracy is substantial with the available 80% of the challenges, the model can then be tested with the remaining CRPs to ascertain the correctness of the predicted responses. However, the critical concern here pertains to the data space. In the domain of binary data space, Logistic Regression (LR) stands out as an algorithm capable of modeling behavior through specific activation functions. This influential Machine Learning-based classification tool operates on the principles of probability, utilizing an activation function, such as the sigmoid function (illustrated in Figure 4.7), to interpolate results within the range of 0 to 1. The modeling algorithm used in this research is only LR. In this technique, the independent variables are given as input values and it produces the dependent variable, which is output value/predicted value. In this study, the CRPs are collected through an environment developed for enrolling the PUF responses, as discussed in Chapter 3 (section 3.6). EPP interface allows seamless integration of the PUF Under Test (PUT) and has been replaced with the proposed PUF model. The environment, previously discussed in section 3.6 and depicted in Figure 3.12, witnesses the replacement of CRO MUX ARRAY with APUF ARRAY. Upon initiating the enable pulse train, all responses are gathered through the EPP interface and directly recorded into an Excel spreadsheet, constituting the CRP dataset. Unlike CRO PUFs, post-processing is required to generate a signature; the

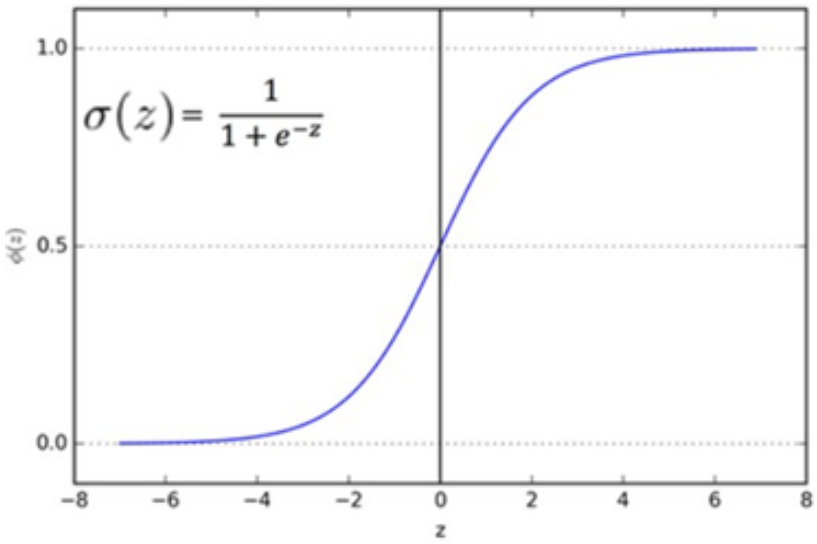


Figure 4.7: Sigmoid function

data can be acquired directly into the Excel sheet for modeling purposes. Table 4.1 is generated from this Excel sheet, displaying a small sample of the extensive CRP space. Specifically, the initial 20 CRPs are displayed in Table 4.1. The available CRPs are leveraged within the LR algorithm to evaluate the resistance against potential attacks.

The collected data set of challenge response pairs is divided into two separate sets for a particular PUF instance 'P'. Utilizing a machine learning-based algorithm, a model is generated from one of these sets. Another dataset is employed to assess the predictive accuracy of this model for the 'P' instance and the LR algorithm is used to make, predictions, as illustrated in Figure 4.8.

Logistic regression is employed to model an attack and verify the resistance metric for the proposed APUF. The design functionality is thoroughly validated within the environment elaborated upon in Chapter 3, which captures diverse sets of random challenge-response pairs (CRPs) presented in Table 4.2.

4.5 Observation from Logistic Regression

CRPs acquired through the environment are shown in Figure 3.12 and these are posted into a Notepad or an Excel application. A CRP space consisting of 5000 entries is

considered for the construction of the model and a subset of these is displayed in Table 4.1. Observations were conducted on three subsets of the 5000 CRPs based on the

Table 4.1: Challenge response pairs (CRPs) data set

Challenges	**Responses**
64'h9283c630815977c	FF00FF0000FF00FF
64'h824e3d711516856b	FFFF0000FFFFFF00
64'h92304516c4bb0240	FF00FFFF0000FF00
64'h200fbac6d9bb7303	0000FFFFFF00FF00
64'h6844dcc6a582ac22	000000FFFFFFFFFF
64'h686d3ec2141a7dfb	00FFFFFF00FF0000
64'h6494978f8293cf35	FF000000FF0000FF
64'hef911feddf105f4e	00FF00FF000000FF
64'h7b2869d1d09564d2	0000FFFF00FFFFFF
64'hf0d856b216b4c3a3	FF00FFFFFF000000
64'hb7c4e36a630d8a4c	000000FF0000FFFF
64'hbcb61ee2e15a1210	FFFFFFFFFFFFFF00
64'h663191b8194b8886	FFFFFF00FF0000FF
64'h7ef15bcce0bca68a	000000FF00FFFFFF
64'h86adf529941ee585	FF00FFFF00FF00FF
64'hcbcd8a2d11808228	0000FF00FF00FF00
64'h99d8e3dfdd44e6d6	FF000000FF00FFFF
64'h60e2deebf11c28de	00FFFF0000FF00FF
64'h8ec4b310818b10b6	FF000000FF0000FF

recorded dataset in Table 4.2.

- 15% of CRPs to test and 85
- 25% of CRPs to test and 75
- 35% of CRPs to test and 65

The responses are plotted using the sigmoid as the activation function. Sigmoid function is detailed in Equation (4.1).

$$Y = \frac{1}{1 + e^{-z}} \tag{4.1}$$

The modeling algorithm is shown in Figure 4.8, involving the registration and subsequent analysis of the CRP space. The outcomes obtained from this process are

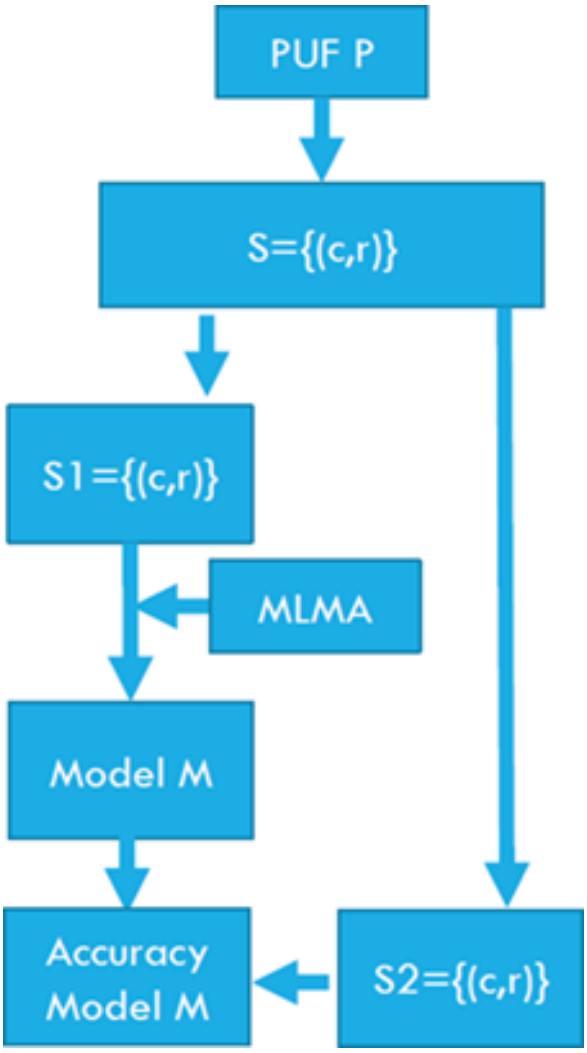


Figure 4.8: Flow chart for designing a machine learning based model

presented in Table 4.2.

Table 4.3 presents the prediction rates derived from the values recorded in Table 4.2, utilizing Equation (4.2). Using varied percentages of testing and training the model with 5000 CRPs, the prediction rate of the collected data, employing the sigmoid activation function, is executed in a Jupyter notebook. The Jupyter notebook, an application within Linux facilitating the execution of Python scripts for logistic regression, was employed for this study. If the test rate is 0.25 for dataset, the remaining set is used for training the model. This process is used to calculate the prediction rate (Pr) accuracy of the model. The prediction rate, defined as the ratio of the correctly

Table 4.2: Results from the model using Signmoid as activation function

No of stages of Arbiter PUF Proposed	15% of CRPs are used to test the model and 85% to train the model					
	Total CRPs	No of Correct responses Predicted	Undetermined responses	No of Wrong Responses Predicted	Time Taken	Prediction Rate
	750	400	5	345	0.1396	0.533333333
	1650	860	12	778	0.1714	0.521212121
	2850	1419	6	1425	0.2972	0.497894737
	2460	1222	14	1224	0.3782	0.496747967
	25% of CRPs are used to test the model and 75% to train the model					
	Total CRPs	No of Correct responses Predicted	Undetermined responses	No of Wrong Responses Predicted	Time Taken	Prediction Rate
	750	390	10	350	0.1276	0.52
	1650	835	12	803	0.1727	0.506060606
	2850	1400	18	1432	0.184	0.49122807
	2460	1220	10	1230	0.2948	0.495934959
	35% of CRPs are used to test the model and 65% to train the model					
	Total CRPs	No of Correct responses Predicted	Undetermined responses	No of Wrong Responses Predicted	Time Taken	Prediction Rate
	750	400	15	335	0.1351	0.533333333
	1650	860	12	778	0.2198	0.521212121
	2850	1419	5	1426	0.2383	0.497894737
64	2460	1220	9	1231	0.3798	0.495934959

predicted responses to the predefined test set, is determined using this equation and is displayed in Table 4.2.

$$Predictionrate(Pr) = \frac{Truepredictedresponse}{TestSet} \tag{4.2}$$

Inference on Proposed APUF for attack resistance

Table 4.3: Performance of logistic regression to predict the model (Pr)

CRP / TEST		0.15	0.25	0.35
		Prediction rate(Pr)		
750		53.3	52	53.33
1650		52.12	50.6	52.12
2850		49.78	49.12	49.8
4920		49.67	49.59	49.59

The investigation into the proposed Arbiter Physical Unclonable Function (APUF)

for its resistance to attacks was conducted via an extensive literature review. The results, as detailed in Table 4.4, indicate a remarkably high predictability ranging from 95% to 99% Ye et al. (2016), Kumar and Niamat (2018) when considering the same APUF. However, it is essential to note that this analysis was performed on a basic APUF using fundamental FPGA architectures, limiting the incorporation of additional functional blocks on the fabric. Nevertheless, the evaluation of the proposed APUF demonstrates a significant improvement of almost 50% in its resistance to predictive attacks Ye et al. (2016). This affirms the efficacy of the methodology employed, particularly in organizing Configurable Reconfigurable Objects (CROs) as Hard-macros. Consequently, this approach indicates the potential to mitigate systematic process variations, thereby accommodating technological advancements and reduced gate lengths. It is apparent that random process variations can be leveraged to generate signatures or Challenge Response Pairs (CRPs). The efficiency of this process is contingent upon the design's packaging and the optimization of the PUF enrollment process on the FPGA device.

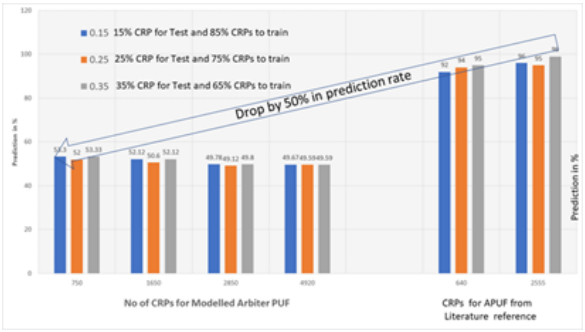


Figure 4.9: Plot for the CRPs Vs. Prediction rate (with reference to Ye et al. (2016), Kumar and Niamat (2018)

Table 4.4: Observation from Literature survey Ye et al. (2016) for a conventional Arbiter PUF

LR on AR PUF				
ML Method	**No of Stages**	**Prediction Rate**	**CRPs**	**Training Time**
LR	64	95	640	0.01 sec
LR	64	99	2555	0.13 sec

The subsequent section shows the manner in which the APUF model can function within a connected environment by assuming the role of a node responsible for authenticating various data streams passing through it.

4.6 A Connected System Model for Securing Smart Meters

As discussed in section 2.3.4, the connected model with the PUF node is implemented on the Xilinx Vivado, an IDE that helps in designing a FPGA based System on Chips (SoCs), by utilizing the on chip processor based FPGA targets from specific vendors. The model relies exclusively on IPs provided by Xilinx and the architecture is outlined in Figure 2.13, serving as a reference in Figure 4.10.

The host shown in the figure is actually a processor based system located on the server that serves the application data to the user communicating through the PUF nodes, which are located on the smart meters. Smart meters host these PUFs and these are controlled by the on-chip CPU. Smart Meter Gateway (SMG) would be specific to the environment with which it is connected. For instance, in a smart meter application, the gateways would constitute the distribution servers, commonly referred to as SMG.

Figure 4.11 shows the block diagram implementation for the connected model shown in Figure 4.10. The PUF node at the extreme right in Figure 4.10 is already designed and validated. In the same Vivado environment, the PUF is integrated with the on chip microblaze CPU and the interface part is addressed.

The environment of PUF node is built in Vivado. Figure 4.12 shows the manner in which the designed Arbiter PUF/Ring Oscillator communicates with the processor to process the CRPs. The work done thus far is only at the PUF Node stage. Consequently, the processing work of packet data communicated from the HOST is still in progress.

Assuming the packet data challenge is relayed through the communication interface of the PUF node, the PUF node has been structured to generate a response using the AXI UARTLITE, representing the UART port on the Artix-7 Device's microblaze CPU. Subsequently, the microblaze CPU verifies the packet data received through the UART port using an application interface.

The current ongoing communication model involves the extraction of data from the AXI layer, transmitting it to the Physical Unclonable Function (PUF), receiving the response from the PUF, relaying it back to the microblaze, and conducting the necessary handshaking and validation processes through the gateway. This comprehensive application stack remains a work in progress.

Figure 4.12 demonstrates the establishment of an AXI wrapper termed "axi_cro_puf_0," created specifically for interfacing the RTL block with the AXI master. This wrapper manages the address mapping and facilitates communication between the AXI slave address and the PUF block through an AXI slave port. Compliance with the AXI bus protocol is essential for seamless data communication.

It is observed that the PUF node and smart meter coexist on the same device. In the connected model shown in Figure 4.10, the extreme right PUF nodes are actually on the smart meters. Thus, this setup is meant to use designed PUFs as security macro for safeguarding Smart Meter Systems in an IoT Environment.

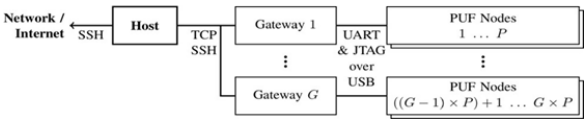


Figure 4.10: Overview of connected environment Chongyan Gu (QUB)

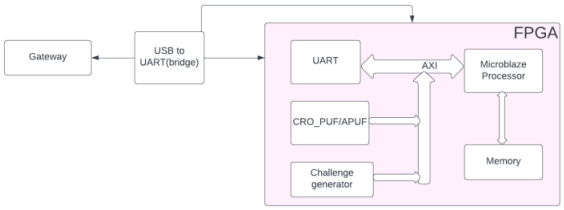


Figure 4.11: Block diagram of PUF Node with the Processor on-chip

4.6.1 Verification environment to verify the Transaction level model(TLM) packet of PUF CRP

Memory¬ mapped direct addressing of slave peripherals is used to communicate between the MicroBlaze CPU and the AXI-Slave peripherals. Each peripheral includes

a software API that describes how to access these registers; nonetheless, an overview of the register functions is presented herein for reference.

The packet format of the command originating from the HOST is shown in Figure 4.13. The command's first byte indicates the nature of the command, followed by a two byte command. Subsequently, the payload, typically comprising the challenge and indicating the source as either from the host or the peripheral, is communicated.

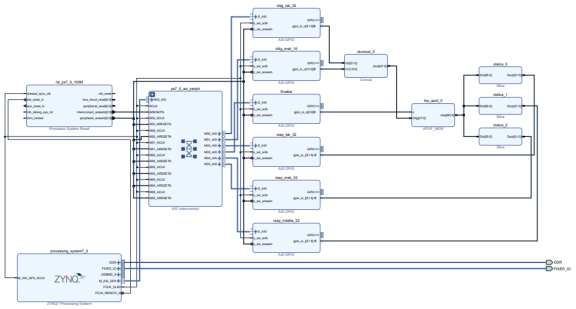


Figure 4.12: Schematic of PUF Node with the Processor on-chip

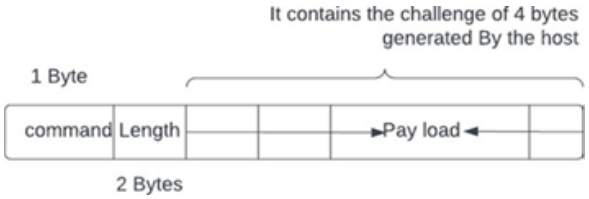


Figure 4.13: Packet format from the HOST to PUF peripheral

The AXI registers of the PUF peripheral can be accessed directly using a command given by the system. Debugging the PUF design is a snap with this feature, which makes it simple to get at the PUF's internal values. The format of the replies returned to the host is shown in Figure 4.14.

The payload initiates with the "LEN" (length in bytes) as the initial two-byte field, followed by 'length' bytes of payload. Finally, a 2-byte checksum confirms the integrity of the current data block.

Figure 4.14: Packet format for PUF peripheral

4.6.2 Results Recorded Through TLM packet transaction over AXI interface

Given the ARM-based hardware system, communication between the host and the peripheral occurs via an AXI stream interface. A C-program operating on the microblaze as the application layer enables user communication through this interface, as illustrated in Figure 4.15. The entire programming environment for initiating transactions, including challenges and recording response databases, is detailed.

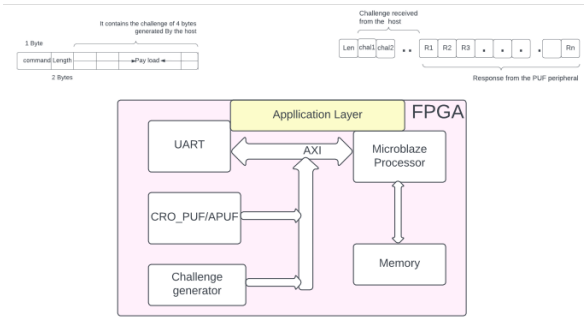


Figure 4.15: Verification environment enabling TLM packet: System on chip Environment for Smart-meter with PUF node and on-chip CPU

The algorithm governing the transactions from the application layer is shown in Figure 4.16. The challenge and response pairs are recorded through these transactions, acquired on a FIFO system, and documented in an Excel file. The recorded data is presented in a table for reference.

4.7 Summary

In section 4.1, the physical unclonable function (PUF) circuit is discussed, which is a hardware security module that generates a unique digital fingerprint for each chip. The

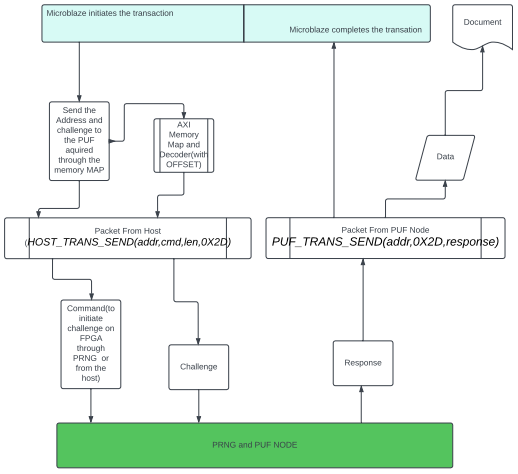


Figure 4.16: Algorithm for TLM PACKET between CPU AND PUF node

PUF circuit is composed of several components, including an arbiter, which is responsible for generating the digital fingerprint. Section 4.2 presents a proposed design for the PUF circuit. The design is based on a ring oscillator, which is a type of oscillator that generates a periodic signal. The proposed design uses a modified ring oscillator to generate the digital fingerprint. Section 4.3 discusses the attack resistance of the PUF circuit. The PUF circuit is designed to be resistant to various types of attacks, including modelling attacks, side-channel attacks, and invasive attacks. Section 4.4 presents the results of a logistic regression analysis of the PUF circuit. The analysis shows that the PUF circuit is highly resistant to modelling attacks and side-channel attacks. Section 4.5 presents a connected system model for securing smart meters. The model is based on the PUF circuit and includes a verification environment to verify the transaction level model (TLM) packet of PUF CRP. The results recorded through TLM packet transaction over AXI interface are also discussed. Subsection 4.5.1 discusses the verification environment used to verify the TLM packet of PUF CRP. The verification environment includes a testbench, which is used to simulate the behaviour of the PUF circuit. Subsection 4.5.2 presents the results of TLM packet transaction over AXI interface. The responses obtained using the test environment are listed down in table 4.5 and 4.6. The responses are actually a large database which is recorded in an excel sheet. The tables only show a few of them. It is processed and interpreted in tabular 4.2 and 4.3. The results show that the PUF circuit is highly resistant to modelling attacks and side-channel attacks.

Table 4.5: Challenge and Response pairs of PUF recorded from the environment implemented for smart-meters

CHALLENGE AND RESPONSE PAIRS				CHALLENGE AND RESPONSE PAIRS			
Location	CHALLENGE		Response	Location	CHALLENGE		Response
X12Y8	000000100	000001100	0	X16Y8	010000111	010001111	0
X12Y10	000001010	000010010	0	X16Y10	010001000	010010000	1
X12Y12	000010100	000011100	0	X16Y12	010010011	010011011	1
X12Y14	000011100	000100100	1	X16Y14	010011000	010100000	0
X12Y16	000100110	000101110	0	X16Y16	010100000	010101000	1
X12Y18	000101011	000110011	0	X16Y18	010101000	010110000	0
X12Y20	000110110	000111110	1	X16Y20	010110100	010111100	1
X12Y22	000111010	001000010	1	X16Y22	010111001	011000001	1
X14Y22	001000000	001001000	0	X18Y22	011000011	011001011	0
X14Y20	001001001	001010001	1	X18Y20	011001010	011010010	0
X14Y18	001010101	001011101	0	X18Y18	011010100	011011100	0
X14Y16	001011101	001100101	1	X18Y16	011011011	011100011	0
X14Y14	001100000	001101000	0	X18Y14	011100110	011101110	0
X14Y12	001101111	001110111	0	X18Y12	011101001	011110001	1
X14Y10	001110011	001111011	1	X18Y10	011110001	011111001	1
X14Y8	001111100	010000100	1	X18Y8	011111110	100000110	1

In conclusion, this chapter presents various components of Arbiter PUF circuit with its construction and presents a proposed design for the Arbiter PUF circuit based on a modified ring oscillator. It extends the discussion on the attack resistance of the PUF circuit and presents the results of a logistic regression analysis of the PUF circuit. Finally, presents a connected system model for securing smart meters based on the PUF circuit and includes a verification environment to verify the transaction level model (TLM) packet of PUF CRP. The results of TLM packet transaction over AXI interface show that the PUF circuit is significantly resistant to modelling attacks and side-channel attacks.

Table 4.6: Continued Challenge and Response pairs of PUF recorded from the environment implemented for smart-meters

CHALLENGE AND RESPONSE PAIRS				CHALLENGE AND RESPONSE PAIRS			
Location	CHALLENGE		Response	Location	CHALLENGE		Response
X20Y8	100000110	100001110	0	X24Y8	110000011	110001011	1
X20Y10	100001011	100010011	0	X24Y10	110001011	110010011	0
X20Y12	100010111	100011111	1	X24Y12	110010111	110011111	1
X20Y14	100011110	100100110	0	X24Y14	110011000	110100000	0
X20Y16	100100000	100101000	1	X24Y16	110100111	110101111	0
X20Y18	100101101	100110101	0	X24Y18	110101110	110110110	1
X20Y20	100110111	100111111	1	X24Y20	110110011	110111011	1
X20Y22	100111011	101000011	1	X24Y22	110111111	111000111	0
X22Y22	101000010	101001010	0	X26Y22	111000000	111001000	0
X22Y20	101001100	101010100	1	X26Y20	111001001	111010001	1
X22Y18	101010100	101011100	0	X26Y18	111010001	111011001	0
X22Y16	101011111	101100111	1	X26Y16	111011110	111100110	0
X22Y14	101100100	101101100	1	X26Y14	111100010	111101010	1
X22Y12	101101010	101110010	0	X26Y12	111101100	111110100	1
X22Y10	101110111	101111111	1	X26Y10	111110011	111111011	1
X22Y8	101111111	110000111	0	X26Y8			

Chapter 5

Conclusion and Future Work

5.1 Conclusive Remarks

- The primary aim of this research is to secure a smart meter with a delay based PUF circuit based on FPGA. A significant emphasis was placed on device physics, because securing a device starts from the physical layer.

- The study focused on the understanding of PUF circuit functionalities and strategies to enhance resource utilization efficiency within FPGA frameworks.

- The parameters that determine the strength of FPGA were studied and bench marked performance metrics were analyzed for the proposed PUF circuits.

- The study was conducted on generation of a stable PUF response, while mitigating the influence of circuit activity around PUF. This included addressing issues, such as minimal bit aliasing effects and mitigating environmental noise variables, such as voltage and temperature fluctuations.

- The performance metrics of Ring Oscillator PUFs and Arbiter PUFs were meticulously evaluated, comprising aspects of uniformity, uniqueness, reliability, and resistance against potential attacks.

- The proposed methodology demonstrated a significant improvement, rendering the system 7% more unique than conventional methods. The proposed method specifically focused on achieving uniqueness, ensuring that each chip within a group generates a signature distinct from others.

- An enhanced architectural design for an Arbiter PUF was proposed. It was found that increased FPGA fabric utilization directly correlated with heightened PUF strength. This conclusion was drawn based on the outcomes derived from an attack modeled using Logistic regression, exhibiting that the proposed PUF was 50% more resilient against attacks compared to conventional PUFs based on a literature survey.

- Off-loading the FPGA from enrolment activities was found to be crucial, because any circuit activity around the PUF would bias the response.

- The study analyzed the characterization of RO frequencies and the impact of systematic variations and environmental noise parameters on a wafer and across a group of dice. The distinction between random noise within individual chips and systematic process variations across a group of chips manufactured on a single wafer was explained qualitatively and quantitatively.

- Conventional PUF design methodologies from existing literature have overlooked the aforementioned critical aspects. While they could derive efficient responses using post-processing techniques, the comprehensive analysis of response generation, including error correction codes, was explored.

- PUF could consistently produce the same output despite significant environmental changes, such as voltage and temperature fluctuations.

- As a part of this study, one of the important concerns with PUF circuits is identified as the evolving target architectures in FPGA devices along with shrinking of device's technology. The designed hard macro of the PUF circuit needs to be portable on to changing architectures of FPGA devices.

- The primary focus revolved around ensuring the security of the smart meter, prompting the development of its interconnected model. However, the current progress only covers the PUF node component, necessitating the involvement of a team of engineers for the further development of additional layers. The endeavor constitutes a comprehensive hardware-software co-design environment. Presently, the setup pertains specifically to the PUF node, validated for authenticating smart meter data. Nevertheless, pending tasks include the establish-

ment of an application layer to facilitate packet retrieval via the TCP protocol through the gateway.

The future scope of work involving PUFs is explained in the subsequent section, outlining several pivotal areas for exploration.

5.2 Scope of future work

1. Modeling parameters on PUF to investigate the effect of aging, voltage, and temperature
2. Quantitative analysis on how response correlates with FPGA fabric utilization
3. Quantification of the level of symmetry necessary in the PUF circuit during circuit activities
4. Validation of the designed PUF for a complete transaction from the host device to gateway to USB to JTAG to AXI/MICROBLAZE to PUF and vice versa. Presently, the AXI/MICROBLAZE to PUF layer is done. However, it is desirable to validate a smart meter communication model in an IoT environment for its security by enabling transactions across all layers.
5. Portability of a PUF across architectures needs to be addressed.
6. The delay associated with the PUF response need to be considered

List of publications

Conference proceedings

1. **D. Reddy Jeeru, K. Panduranga Vittal, Anikethan. H V U and Anjana.Anjana. S. Kumar, "Implementation of Enhanced Parallel port interface for Frequency analysis in a configurable Ring Oscillator PUF circuits on Xilinx Spartan 3E architecture," 2019 IEEE International Con- ferenceConference on Electronics, Computing and Communication Technologies (CONECCT), 2019, pp. 1-7, doi: 10.1109/CONECCT47791.2019.9012874**

Refereed journals

1. **Abhijith Manchikanti Venkata, Dinesh Reddy Jeeru, Vittal K.P. "Design And Modelling An Attack on Multiplexer Based Physical Unclonable Function" International Journal of Engineering Trends and Technology 68.6(2020):63-67.**

Bibliography

Alkabani, Y., Koushanfar, F., and Potkonjak, M. (2007). Remote activation of ics for piracy prevention and digital right management. In *2007 IEEE/ACM International Conference on Computer-Aided Design*, pages 674–677. IEEE.

Anderson, J. H. (2010). A puf design for secure fpga-based embedded systems. In *2010 15th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 1–6. IEEE.

Becker, G. T. (2015). The gap between promise and reality: On the insecurity of xor arbiter pufs. In *Cryptographic Hardware and Embedded Systems–CHES 2015: 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings 17*, pages 535–555. Springer.

Bernard, F., Fischer, V., Costea, C., and Fouquet, R. (2012). Implementation of ring-oscillators-based physical unclonable functions with independent bits in the response. *International Journal of Reconfigurable Computing*, 2012:13–13.

Bhargava, M. and Mai, K. (2014). An efficient reliable puf-based cryptographic key generator in 65nm cmos. In *2014 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–6. IEEE.

Cao, Y., Zhang, L., Zalivaka, S. S., Chang, C.-H., and Chen, S. (2015). Cmos image sensor based physical unclonable function for coherent sensor-level authentication. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 62(11):2629–2640.

Chongyan Gu (QUB), Neil Hanley (QUB), R. H. A. M. H. A. G. M. Puf enhanced smart meter hardware architecture and an authentication/key management deployment architecture. In *SPARKS, Smart Grid Protection Against CybeR Attacks*.

Delvaux, J., Peeters, R., Gu, D., and Verbauwhede, I. (2015). A survey on lightweight entity authentication with strong pufs. *ACM Computing Surveys (CSUR)*, 48(2):1–42.

Delvaux, J. and Verbauwhede, I. (2014). Fault injection modeling attacks on 65 nm arbiter and ro sum pufs via environmental changes. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 61(6):1701–1713.

Design, M.-B. D. (2018). Vivado design suite user guide.

Fraunhofer institute Merli, D., Schuster, D., Stumpf, F., and Sigl, G. (2011). Side-channel analysis of pufs and fuzzy extractors. In McCune, J. M., Balacheff, B., Perrig, A., Sadeghi, A.-R., Sasse, A., and Beres, Y., editors, *Trust and Trustworthy Computing*, pages 33–47, Berlin, Heidelberg. Springer Berlin Heidelberg.

Ganji, F., Tajik, S., Fäßler, F., and Seifert, J.-P. (2016). Strong machine learning attack against pufs with no mathematical model. In Gierlichs, B. and Poschmann, A. Y., editors, *Cryptographic Hardware and Embedded Systems – CHES 2016*, pages 391–411, Berlin, Heidelberg. Springer Berlin Heidelberg.

Gao, Y., Li, G., Ma, H., Al-Sarawi, S. F., Kavehei, O., Abbott, D., and Ranasinghe, D. C. (2016a). Obfuscated challenge-response: A secure lightweight authentication mechanism for puf-based pervasive devices. In *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, pages 1–6. IEEE.

Gao, Y., Ranasinghe, D. C., Al-Sarawi, S. F., Kavehei, O., and Abbott, D. (2015). mrpuf: A novel memristive device based physical unclonable function. In *Applied Cryptography and Network Security: 13th International Conference, ACNS 2015, New York, NY, USA, June 2-5, 2015, Revised Selected Papers 13*, pages 595–615. Springer.

Gao, Y., Ranasinghe, D. C., Al-Sarawi, S. F., Kavehei, O., and Abbott, D. (2016b). Emerging physical unclonable functions with nanotechnology. *IEEE access*, 4:61–80.

Gassend, B., Clarke, D., Van Dijk, M., and Devadas, S. (2002). Silicon physical random functions. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 148–160.

Gassend, B., Dijk, M. V., Clarke, D., Torlak, E., Devadas, S., and Tuyls, P. (2008). Controlled physical random functions and applications. *ACM Transactions on Information and System Security (TISSEC)*, 10(4):1–22.

Guajardo, J., Kumar, S. S., Schrijen, G.-J., and Tuyls, P. (2007). Fpga intrinsic pufs and their use for ip protection. In *Cryptographic Hardware and Embedded Systems-CHES 2007: 9th International Workshop, Vienna, Austria, September 10-13, 2007. Proceedings 9*, pages 63–80. Springer.

Guo, Q., Ye, J., Gong, Y., Hu, Y., and Li, X. (2016). Efficient attack on non-linear current mirror puf with genetic algorithm. In *2016 IEEE 25th Asian Test Symposium (ATS)*, pages 49–54. IEEE.

Handschuh, H. (2012). Hardware-anchored security based on sram pufs, part 1. *IEEE Security & Privacy*, 10(3):80–83.

Harishma, B., Mathew, P., Patranabis, S., Chatterjee, U., Agarwal, U., Maheshwari, M., Dey, S., and Mukhopadhyay, D. (2022). Safe is the new smart: Puf-based authentication for load modification-resistant smart meters. *IEEE Transactions on Dependable and Secure Computing*, 19(1):663–680.

Helfmeier, C., Boit, C., Nedospasov, D., and Seifert, J.-P. (2013). Cloning physically unclonable functions. In *2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 1–6. IEEE.

Herder, C., Yu, M.-D., Koushanfar, F., and Devadas, S. (2014). Physical unclonable functions and applications: A tutorial. *Proceedings of the IEEE*, 102(8):1126–1141.

Hesselbarth, R., Wilde, F., Gu, C., and Hanley, N. (2018). Large scale ro puf analysis over slice type, evaluation time and temperature on 28nm xilinx fpgas. In *2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 126–133.

Holcomb, D. E., Burleson, W. P., and Fu, K. (2008). Power-up sram state as an identifying fingerprint and source of true random numbers. *IEEE Transactions on Computers*, 58(9):1198–1210.

Hospodar, G., Maes, R., and Verbauwhede, I. (2012). Machine learning attacks on 65nm arbiter pufs: Accurate modeling poses strict bounds on usability. In *2012*

IEEE international workshop on Information forensics and security (WIFS), pages 37–42. IEEE.

ID, I. (2016). Products.

IntrinsicID, R. M., van der Leest, V., van der Sluis, E., and Willems, F. (2015). Secure key generation from biased pufs. Cryptology ePrint Archive, Paper 2015/583. https://eprint.iacr.org/2015/583.

Jeeru, D. R., Vittal, K. P., Anikethan, H., and Kumar, A. S. (2019). Implementation of enhanced parallel port interface for frequency analysis in a configurable ring oscillator puf circuits on xilinx spartan 3e architecture. In *2019 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, pages 1–7. IEEE.

Kodỳtek, F. and Lórencz, R. (2015). A design of ring oscillator based puf on fpga. In *2015 IEEE 18th International Symposium on Design and Diagnostics of Electronic Circuits & Systems*, pages 37–42. IEEE.

Kohnhäuser, F., Schaller, A., and Katzenbeisser, S. (2015). Puf-based software protection for low-end embedded devices. In *Trust and Trustworthy Computing: 8th International Conference, TRUST 2015, Heraklion, Greece, August 24-26, 2015, Proceedings 8*, pages 3–21. Springer.

Konigsmark, S. C., Hwang, L. K., Chen, D., and Wong, M. D. (2014). Cnpuf: A carbon nanotube-based physically unclonable function for secure low-energy hardware design. In *2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 73–78. IEEE.

Kumar, R. and Burleson, W. (2014). On design of a highly secure puf based on nonlinear current mirrors. In *2014 IEEE international symposium on hardware-oriented security and trust (HOST)*, pages 38–43. IEEE.

Kumar, S. and Niamat, M. (2018). Machine learning based modeling attacks on a configurable puf. In *NAECON 2018-IEEE National Aerospace and Electronics Conference*, pages 169–173. IEEE.

Kumar, S. S., Guajardo, J., Maes, R., Schrijen, G.-J., and Tuyls, P. (2008). The butterfly puf protecting ip on every fpga. In *2008 IEEE International Workshop on Hardware-Oriented Security and Trust*, pages 67–70. IEEE.

Lao, Y., Yuan, B., Kim, C. H., and Parhi, K. K. (2016). Reliable puf-based local authentication with self-correction. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 36(2):201–213.

Lee, J. W., Lim, D., Gassend, B., Suh, G. E., Van Dijk, M., and Devadas, S. (2004). A technique to build a secret key in integrated circuits for identification and authentication applications. In *2004 Symposium on VLSI Circuits. Digest of Technical Papers (IEEE Cat. No. 04CH37525)*, pages 176–179. IEEE.

Lim, D. (2004). Extracting secret keys from integrated circuits [msc. thesis].

Lim, D., Lee, J. W., Gassend, B., Suh, G. E., Van Dijk, M., and Devadas, S. (2005). Extracting secret keys from integrated circuits. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 13(10):1200–1205.

Lin, L., Srivathsa, S., Krishnappa, D. K., Shabadi, P., and Burleson, W. (2012). Design and validation of arbiter-based pufs for sub-45-nm low-power security applications. *IEEE Transactions on Information Forensics and Security*, 7(4):1394–1403.

Maes, R. and Maes, R. (2013). *Physically unclonable functions: Concept and constructions*. Springer.

Maes, R., Rozic, V., Verbauwhede, I., Koeberl, P., Van der Sluis, E., and van der Leest, V. (2012). Experimental evaluation of physically unclonable functions in 65 nm cmos. In *2012 Proceedings of the ESSCIRC (ESSCIRC)*, pages 486–489. IEEE.

Maiti, A. (2012). *A systematic approach to design an efficient physical unclonable function*. PhD thesis, Virginia Tech.

Maiti, A., Gunreddy, V., and Schaumont, P. (2013). A systematic method to evaluate and compare the performance of physical unclonable functions. *Embedded systems design with FPGAs*, pages 245–267.

Maiti, A. and Schaumont, P. (2011). Improved ring oscillator puf: An fpga-friendly secure primitive. *Journal of cryptology*, 24:375–397.

Maiti, A. and Schaumont, P. (2013). The impact of aging on a physical unclonable function. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 22(9):1854–1864.

Majzoobi, M., Koushanfar, F., and Devadas, S. (2010). Fpga puf using programmable delay lines. In *2010 IEEE international workshop on information forensics and security*, pages 1–6. IEEE.

Majzoobi, M., Koushanfar, F., and Potkonjak, M. (2008). Lightweight secure pufs. In *2008 IEEE/ACM International Conference on Computer-Aided Design*, pages 670–673. IEEE.

Nedospasov, D., Seifert, J.-P., Helfmeier, C., and Boit, C. (2013). Invasive puf analysis. In *2013 Workshop on Fault Diagnosis and Tolerance in Cryptography*, pages 30–38. IEEE.

NV, N. S. (2013). Puf-physical unclonable functions–protecting next-generation smart card ics with sram based pufs.

Pappu, R. (2001). Physical one-way functions [ph. d. thesis]. *Massachusetts Institute of Technology, Cambridge, Mass, USA*.

Pappu, R., Recht, B., Taylor, J., and Gershenfeld, N. (2002). Physical one-way functions. *Science*, 297(5589):2026–2030.

Roel, M. and Verbauwhede, I. (2010). Physically unclonable functions: A study on the state of the art and future research directions. *Towards Hardware-Intrinsic Security: Foundations and Practice*, pages 3–37.

Roelke, A. and Stan, M. R. (2016). Attacking an sram-based puf through wearout. In *2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 206–211. IEEE.

Rostami, M., Majzoobi, M., Koushanfar, F., Wallach, D. S., and Devadas, S. (2014). Robust and reverse-engineering resilient puf authentication and key-exchange by substring matching. *IEEE Transactions on Emerging Topics in Computing*, 2(1):37–49.

Roy, A., Roy, D., and Maitra, S. (2022). *How Do the Arbiter PUFs Sample the Boolean Function Class?*, pages 111–130.

Rührmair, U., Sehnke, F., Sölter, J., Dror, G., Devadas, S., and Schmidhuber, J. (2010). Modeling attacks on physical unclonable functions. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 237–249.

Rührmair, U. and Sölter, J. (2014). Puf modeling attacks: An introduction and overview. In *2014 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–6. IEEE.

Rührmair, U., Sölter, J., Sehnke, F., Xu, X., Mahmoud, A., Stoyanova, V., Dror, G., Schmidhuber, J., Burleson, W., and Devadas, S. (2013). Puf modeling attacks on simulated and silicon data. *IEEE transactions on information forensics and security*, 8(11):1876–1891.

Rührmair, U., Xu, X., Sölter, J., Mahmoud, A., Majzoobi, M., Koushanfar, F., and Burleson, W. (2014). Efficient power and timing side channels for physical unclonable functions. In *Cryptographic Hardware and Embedded Systems–CHES 2014: 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings 16*, pages 476–492. Springer.

Sahoo, D., Chakraborty, R., and Mukhopadhyay, D. (2015). Towards ideal arbiter puf design on xilinx fpga: A practitioner's perspective.

Sahoo, D., Mukhopadhyay, D., Chakraborty, R., and Nguyen, P. (2017). A multiplexer-based arbiter puf composition with enhanced reliability and security. *IEEE Transactions on Computers*, PP:1–1.

Schaller, A., Arul, T., van der Leest, V., and Katzenbeisser, S. (2014). Lightweight anti-counterfeiting solution for low-end commodity hardware using inherent pufs. In *International Conference on Trust and Trustworthy Computing*, pages 83–100. Springer.

Schlösser, A., Nedospasov, D., Krämer, J., Orlic, S., and Seifert, J.-P. (2012). Simple photonic emission analysis of aes. In Prouff, E. and Schaumont, P., editors, *Cryptographic Hardware and Embedded Systems - CHES 2012*, pages 41–57, Berlin, Heidelberg. Springer Berlin Heidelberg.

Sedcole, P. and Cheung, P. Y. (2006). Within-die delay variability in 90nm fpgas and beyond. In *2006 IEEE International Conference on Field Programmable Technology*, pages 97–104. IEEE.

Sehwag, V. and Saha, T. (2016). Tv-puf: A fast lightweight analog physical unclonable function. In *2016 IEEE International Symposium on Nanoelectronic and Information Systems (iNIS)*, pages 182–186. IEEE.

Semiconductors, N. (2016). Step up security and innovation with next generation smartmx2 products.

Siddhanti, A., Bodapati, S., Chattopadhyay, A., Maitra, S., Roy, D., and Stănică, P. (2019). *Analysis of the Strict Avalanche Criterion in Variants of Arbiter-Based Physically Unclonable Functions*, pages 556–577.

Siddiqua, T., Gurumurthi, S., and Stan, M. R. (2011). Modeling and analyzing nbti in the presence of process variation. In *2011 12th International Symposium on Quality Electronic Design*, pages 1–8. IEEE.

Simons, P., van der Sluis, E., and van der Leest, V. (2012). Buskeeper pufs, a promising alternative to d flip-flop pufs. In *2012 IEEE International Symposium on Hardware-Oriented Security and Trust*, pages 7–12. IEEE.

Singh, S., Bodapati, S., Patkar, S., Leupers, R., Chattopadhyay, A., and Merchant, F. (2022). Pa-puf: A novel priority arbiter puf.

Stanciu, A., Cirstea, M. N., and Moldoveanu, F. D. (2016). Analysis and evaluation of puf-based soc designs for security applications. *IEEE Transactions on Industrial Electronics*, 63(9):5699–5708.

Su, Y., Holleman, J., and Otis, B. P. (2008). A digital 1.6 pj/bit chip identification circuit using process variations. *IEEE Journal of Solid-State Circuits*, 43(1):69–77.

Suh, G. E. and Devadas, S. (2007). Physical unclonable functions for device authentication and secret key generation. In *Proceedings of the 44th annual design automation conference*, pages 9–14.

Tajik, S., Dietz, E., Frohmann, S., Seifert, J.-P., Nedospasov, D., Helfmeier, C., Boit, C., and Dittrich, H. (2014). Physical characterization of arbiter pufs. In *Cryptographic Hardware and Embedded Systems–CHES 2014: 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings 16*, pages 493–509. Springer.

van der Leest, V., Maes, R., Schrijen, G.-J., and Tuyls, P. (2014). Hardware intrinsic security to protect value in the mobile market. In *ISSE 2014 Securing Electronic Business Processes: Highlights of the Information Security Solutions Europe 2014 Conference*, pages 188–198. Springer.

Van der Leest, V., Schrijen, G.-J., Handschuh, H., and Tuyls, P. (2010). Hardware intrinsic security from d flip-flops. In *Proceedings of the fifth ACM workshop on Scalable trusted computing*, pages 53–62.

Verma, K., Kaushik, B., and Singh, R. (2009). Effects of process variation in vlsi interconnects–a technical review. *Microelectronics International*, 26(3):49–55.

Vijayakumar, A. and Kundu, S. (2015). A novel modeling attack resistant puf design based on non-linear voltage transfer characteristics. In *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 653–658. IEEE.

Vijayakumar, A., Patil, V. C., Prado, C. B., and Kundu, S. (2016). Machine learning resistant strong puf: Possible or a pipe dream? In *2016 IEEE international symposium on hardware oriented security and trust (HOST)*, pages 19–24. IEEE.

Wendt, J. B. and Potkonjak, M. (2014). Hardware obfuscation using puf-based logic. In *2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 270–271. IEEE.

Xilix, I. (2018). Xilinx spartan-3e fpga family: Data sheet.

Xin, X., Kaps, J.-P., and Gaj, K. (2011). A configurable ring-oscillator-based puf for xilinx fpgas. In *2011 14th Euromicro conference on digital system design*, pages 651–657. IEEE.

Ye, J., Guo, Q., Hu, Y., Li, H., and Li, X. (2018). Modeling attacks on strong physical unclonable functions strengthened by random number and weak puf. In *2018 IEEE 36th VLSI Test Symposium (VTS)*, pages 1–6. IEEE.

Ye, J., Hu, Y., and Li, X. (2016). Rpuf: Physical unclonable function with randomized challenge to resist modeling attack. In *2016 IEEE Asian Hardware-Oriented Security and Trust (AsianHOST)*, pages 1–6. IEEE.

Ye, Y., Liu, F., Chen, M., Nassif, S., and Cao, Y. (2010). Statistical modeling and simulation of threshold variation under random dopant fluctuations and line-edge roughness. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 19(6):987–996.

Zeitouni, S., Oren, Y., Wachsmann, C., Koeberl, P., and Sadeghi, A.-R. (2015). Remanence decay side-channel: The puf case. *IEEE Transactions on Information Forensics and Security*, 11(6):1106–1116.

Zhang, L., Kong, Z. H., Chang, C.-H., Cabrini, A., and Torelli, G. (2014). Exploiting process variations and programming sensitivity of phase change memory for reconfigurable physical unclonable functions. *IEEE Transactions on Information Forensics and Security*, 9(6):921–932.

Appendix

Implementation over Xilinx FPGA : Spartan XC3S100E

The devices targeted here are from Xilinx and all of them come under three series categories, which has only a programmable logic inside the FPGA fabric (Xilix, 2018), soft core IPs can still take their place on the fabric, but a dedicated footprint of a hardcore processor is not available..

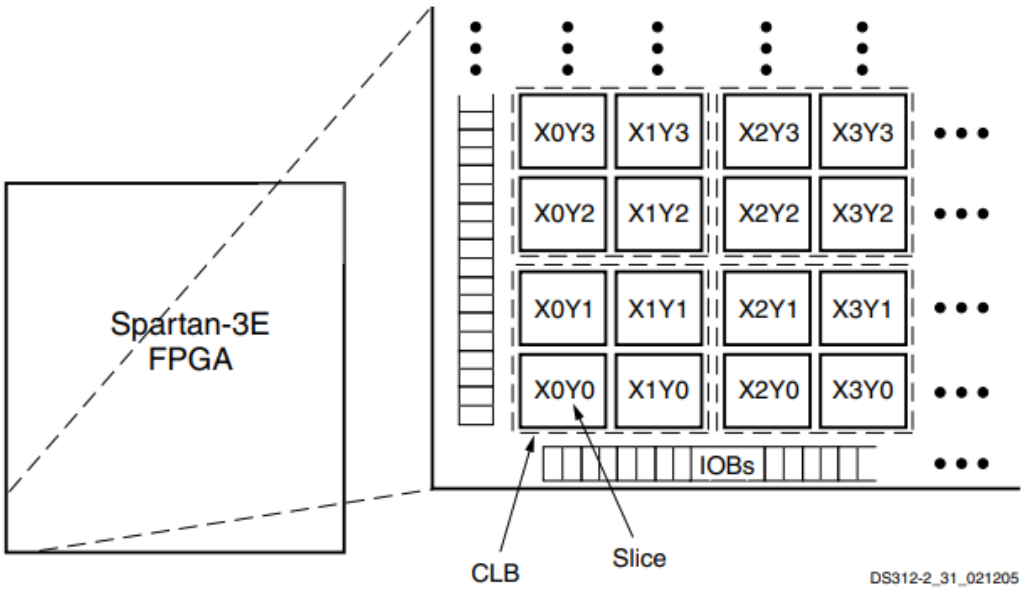


Figure A1: CLB Locations

Figure A1 explains the way the slices are organized on the fabric. Figure A2 also shows how each slice is identified by its location along the X-axis and Y-axis. To implement a PUF circuit on the FPGA fabric, the design entry should be in either

hardware descriptive language or schematic entry. In either of these cases the design need to be addressed with placement constraints as shown in Figure A9, because the PUF circuit should not be reprogrammed on a different location when the design is reprogrammed on FPGA. The constraints written will haves specific format of Xilinx design constraints and need to follow the syntax in executing them on the Integrate Design Environment (Xilinx Vivado IDE). Chapter 3 explains about how the logic is accommodated inside the slice and also shows the methodology involved in propagating the response from the circuit. The implementation steps involved are

- Behavioral entry through a HDL or schematic entry
- Apply location constraints
- Create a hard-macro
- Register the PUF challenge and response pair
- Develop the application layer over the designed hardware to acquire the responses
- Reuse this environment to plug and play any PUF circuit of interest and evaluate the PUF for its performance metrics

Artix-7 as Target Device

In comparison to 3-series devices, 7-series devices will have both Processor System (PS) and Programmable Logic (PL) on the fabric. This eases the process of registering the PUF challenge and response pairs. With the enhanced architectural features, PUF implementation is more efficient. FPGA devices are preferred to ASICs because of advantages, such as re configurability, lesser time to marker. The device contains programmable logic components called configurable Logic Blocks (CLBs), and it has hierarchical re-configurable interconnects that allows to make blocks "wired together" like a bread board which is on-chip programmable. The CLBs are configured to perform simple operations of logic gates or complex combinational function or any sequential logic. In most FPGA devices, the CLBs also contain memory elements,

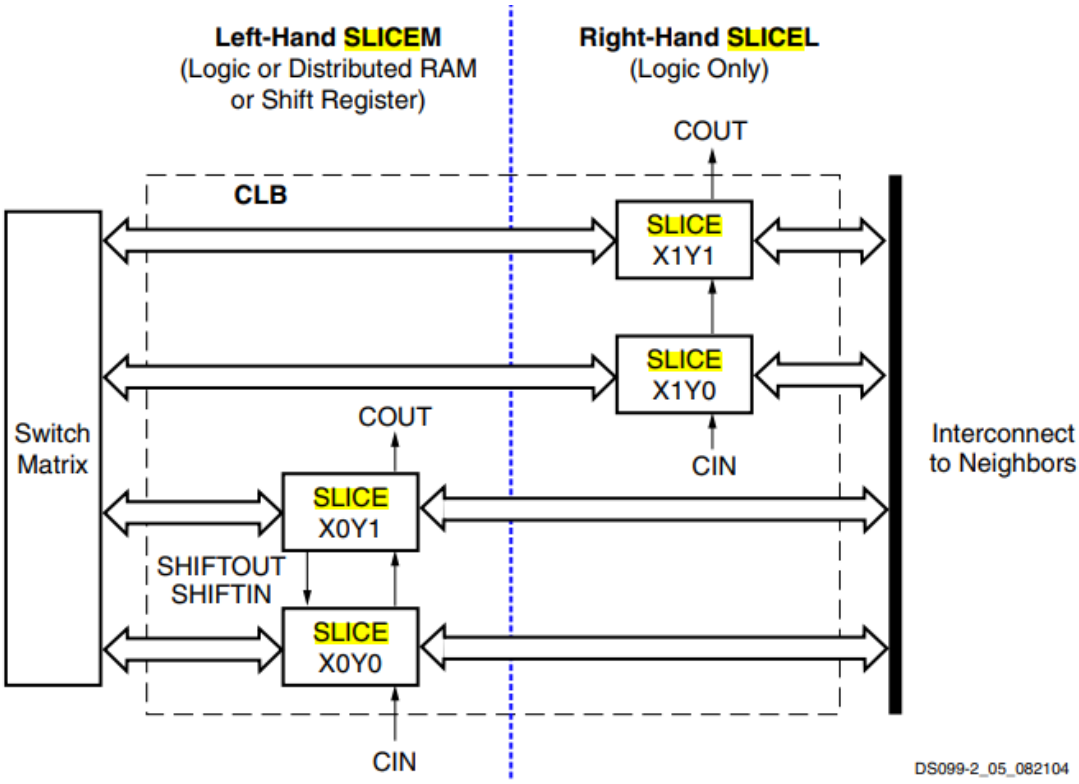


Figure A2: Arrangement of slices with in CLB of Spartan 3e Device

which may be simple registers or complete memory blocks.

The proposed Arbiter design explained in the chapter-4 is implemented on this Basys-3 board, 7-series (Artix-7) based FPGA board as shown in Figure A4 and the slice representation is also shown in Figure A5, having part number (xc7a35tcpg236-1). The Basys-3 is an entry level FPGA development board, which is a complete, ready to use digital circuit development platform based on the latest Artix-7 Field Programmable Gate Array (FPGA) from Xilinx. With its advanced capabilities, this FPGA comes with lesser cost with additional communication ports like USB, VGA, and other ports. With these features, Basys-3 can host designs ranging from introductory combinational circuits to complex sequential circuits like embedded processors with controllers. It allows a large number of designs to be completed without any support and need of external hardware, because it has enough number of General Purpose I/Os and other required I/O devices to allow a large number of designs to be completed without the need for any additional hardware.

As Basys-3 is 7-series based board, it consists of CLBs, Memory elements, each one of the logic blocks consists of two slices. Based on their functionality slices are named as slice L, which can be used for the design implementation purposes only, while other one as slice M, which can be used for implementing the design modules and storage that is for memory purposes. Table A1 shows the components available on the Basys-3 board.

CLB resources, as shown in Figure A3, include two slices that can be two slice L or slice L and slice M. Each slice of the logic block contains:

- Four logic function generators
- Eight storage elements
- Wide function multiplexers
- Fast look ahead carry logic

The function generators (FG) are configurable with 6-input look-up tables (LUTs) ideally, which can also be used as 5 input LUTs. Storage elements like 32¬ bit shift registers, distributed RAM are available only in slice M. Moreover as mentioned earlier, each slice contains eight storage elements. Four of eight configured as either edge triggered D type flip flops or level-sensitive latches. Other available four storage elements can be configured as flip-flops only. Wide multiplexers are available for efficient utilization and for arithmetic functions and it has dedicated high speed carry logic. For accessing the general routing resources, each CLB connected to a switch matrix as shown in Figure A3 below.

Table A1: Component description of Basys-3 board

1	Power good LED	9	FPGA configuration reset button
2	Pmod connector(s)	10	Programming mode jumper
3	Analog signal Pmod connector (XADC)	11	USB host connector
4	Four digit 7-segment display	12	VGA connector
5	Slide switches (16)	13	Shared UART/ JTAG USB port
6	LEDs (16)	14	External power connector
7	Pushbuttons (5)	15	Power Switch
8	FPGA programming done LED	16	Power Select Jumper

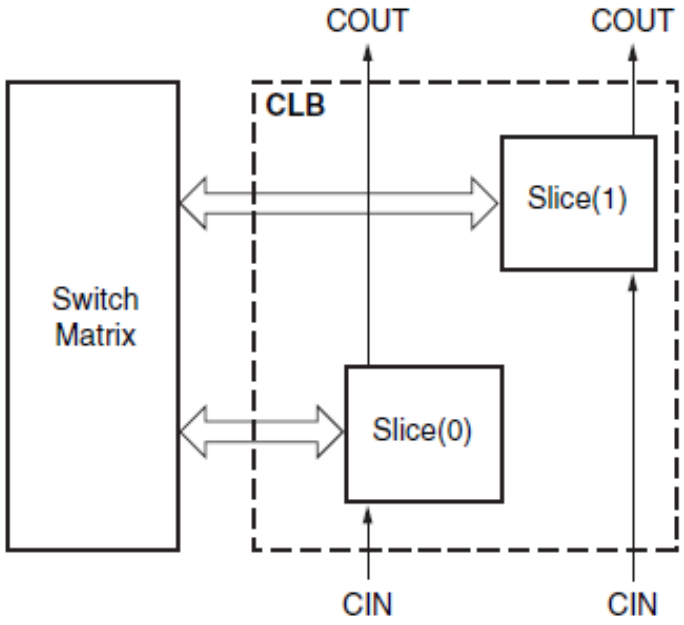


Figure A3: Arrangement of Slices within the logic block

Implementation Details

As discussed in the previous section, about the steps involved in implementation, HDL entry or schematic entry needs a tool from the vendor. Vivado design suite is an Integrated development environment, from Xilinx for the purpose of synthesis and analysis of different HDL designs, which helps the designer to synthesize designs, perform different timing analysis, examine RTL diagrams, and to verify the functionality of a design, it provides simulator to check for different stimulus, and configure the target device with the programmer and it supports high level synthesis for designs

It supports more features than the previous software versions and other development environments from Xilinx like ISE, it supports the 7-series and above device packages.

In order to synthesize a design and to implement it on FPGA fabric there is a need to know hardware description language design flow, because of the need to behaviorally code this design using this for further processes. It includes simulation, synthesis, and implementation. With the help of function generators, any primitive logic in FPGA can be implemented with this tool. The role of synthesizer is to synthe-

size the design by converting the RTL design in HDL to Net lists. While synthesizing the design functionality, the synthesizer tool tries to optimize the design logic. Thus, there is a need to set the hierarchy in properties. After synthesis, there is a provision to drag and drop the design instances, which are present in net list or apply constraints for the placement of instances using constraints. All the required constraints like placement and timing have to be mentioned in the separate file with an extension of '.xdc' (Xilinx design constraints) for better routing and to maintain the symmetry.

The required some placement constraints and their syntaxes are discussed below

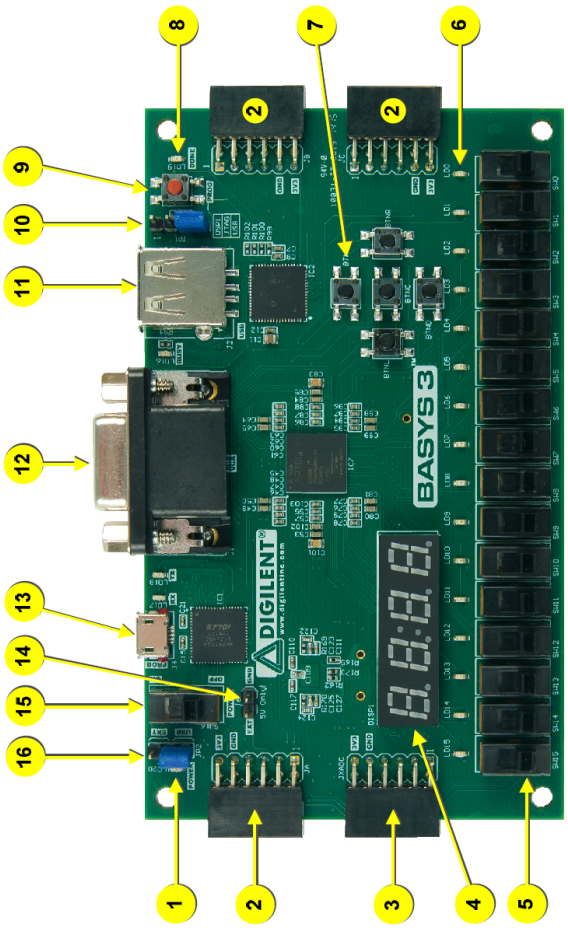


Figure A4: Basys-3 FPGA board

RLOC (Relative Location)

Relative location (RLOC) is a basic mapping, placement, and synthesis constraint. With the help of this constraint, one can group different logic elements into discrete sets. Within the set, one can fix the any element location relative to other elements in the set, without considering the eventual placement in the overall design. This constraint is specified using the slice-based XY coordinate system. As shown in Figure A6, the logic blocks can be placed relative to each other to increase speed and use die resources efficiently. These can provide an order and structure to relate design elements without need to specify their absolute placement on the FPGA die. Moreover, this can replace any existing hard macro with an equivalent that can be directly simulated. This constraint can be added to logic design in HDL design code itself or these constraints could also be applied directly for the instances of design in design constraint file but the RLOC set can be defined in the process of defining macro for logic design.

RLOC ORIGIN (Relative Location Origin) helps to fix the exact location in the die. This constraint helps to change the RLOC values into absolute LOC constraints that represent the structure of the set.

LOC (location)

On the FPGA fabric, design instance/element can be placed in a specific slice by using this constraint. It provides the absolute placement on the FPGA device for design components. This can be range of locations or single location. One can define this constraint for element from the design file or else placement through a GUI interface. At slice level, the FPGA has Cartesian based XY designator.

'LOC' constraint example: Figure A6 and Figure A7 shows the manner in which rows and columns are connected in slices.

BEL

Whenever the basic net list objects need to be placed or mapped on to target device like flip flops, look up tables and carry logic, BEL constraints are applied. As mentioned earlier, generally a BEL or Basic Element corresponds to leaf-cell in the net list view of the design. These can be grouped together on the device in site objects such as

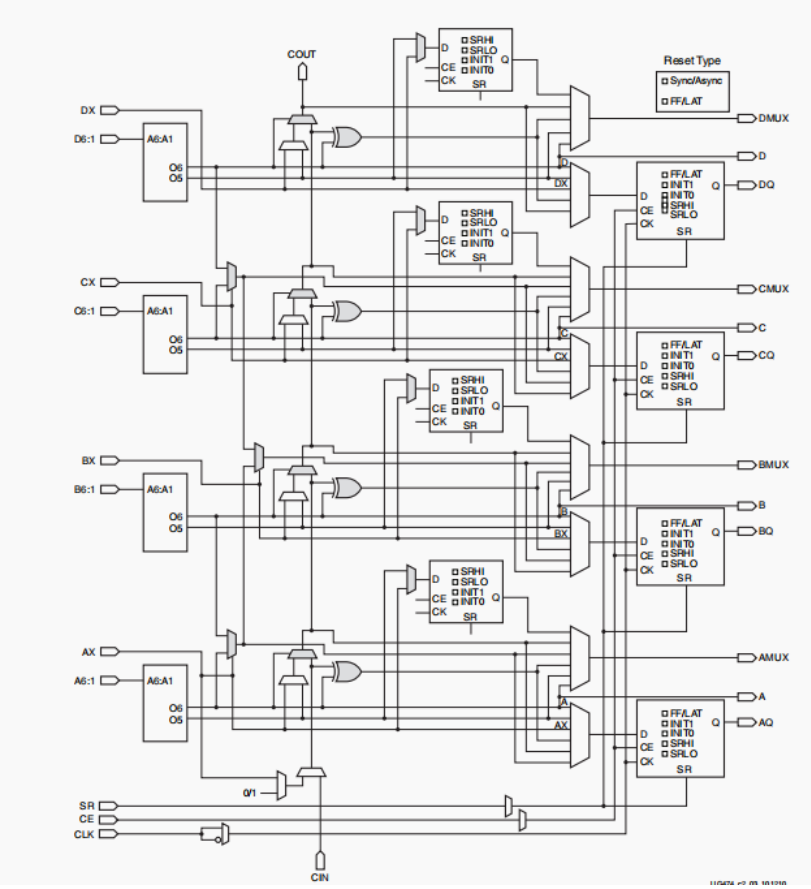


Figure A5: Slice representation of 7-series FPGA device

SLICEs and IO Blocks (IOBs).One or more BEL constraints can be located in a single site as shown in Figure A9 and can use this constraint to assign logic from the design net list into specific locations or device resources on the target device.

Hard Macro

Whenever there is a need to utilize the design repeatedly while maintaining the same placement of design instances and routing resources, then creating and defining of a macro helps to achieve that re-usability with hard-coded macro and that will maintain the design in a fixed logic with fixed placement and routing attributes. For a FPGA based design, the hard macro is an essential part, because there is a need to maintain symmetry of components in the design and for place and route the design components on FPGA has its own routing property.

```
(*RLOC_ORIGIN = "X0Y1"*) ARB_PUF_64BIT  XLXI_1 (.c(CLG[31:0]),
                        .EN(EN),
                        .PUF_OUT(PUF_OUT[1]));
(*RLOC_ORIGIN = "X1Y1"*) ARB_PUF_64BIT  XLXI_2 (.c(CLG[31:0]),
                        .EN(EN),
                        .PUF_OUT(PUF_OUT[2]));
(*RLOC_ORIGIN = "X3Y1"*)ARB_PUF_64BIT  XLXI_3 (.c(CLG[31:0]),
                        .EN(EN),
                        .PUF_OUT(PUF_OUT[3]));
```

Figure A6: Verilog representation for RLOC Constraint

```
(* LOC = "X0Y4", BEL = "MUXF8"  *)MUXF8 dut4(.I0(y1[2]),.I1(y1[3]),.S(c[29]),.O(y1[4]));
(* LOC = "X0Y5", BEL = "MUXF8"  *)MUXF8 dut5(.I0(y1[3]),.I1(y1[2]),.S(c[29]),.O(y1[5]));
(* LOC = "X0Y6" , BEL = "MUXF8"  *)MUXF8 dut6(.I0(y1[4]),.I1(y1[5]),.S(c[28]),.O(y1[6]));
(* LOC = "X0Y7" , BEL = "MUXF8" *)MUXF8 dut7(.I0(y1[5]),.I1(y1[4]),.S(c[28]),.O(y1[7]));
```

Figure A7: Representation of different constraints

The designs in which operations are completely dependent on the symmetry of wire length/delay, such designs cannot be implemented in FPGA devices. One such design is Physical Unclonable Function. Thus, for implementing this on FPGA, the hard macro helps in creating multiple instances with same routing length or wire length. The steps to create a macro for design are discussed below.

- open vivado Tcl console for executing below commands.
  createmacro m1

- now execute below command by adding list of locations
  set rlocs [list d0 X1Y3 d1 X1Y3 d2 X1Y3 ..... d6 X0Y3 d7 X0Y3 '
  here d0, d1, d2 are the instance of particular design
  x0Y0,x0y1 ...is the slice locations where to place the particular instance

- update the macro by executing below command
  updatemacro m1 rlocs

The above steps would allow us to create a hard macro and after successful placement of the macro.
The proposed design macro placement and implementation on FPGA are shown in following Figure 10. .

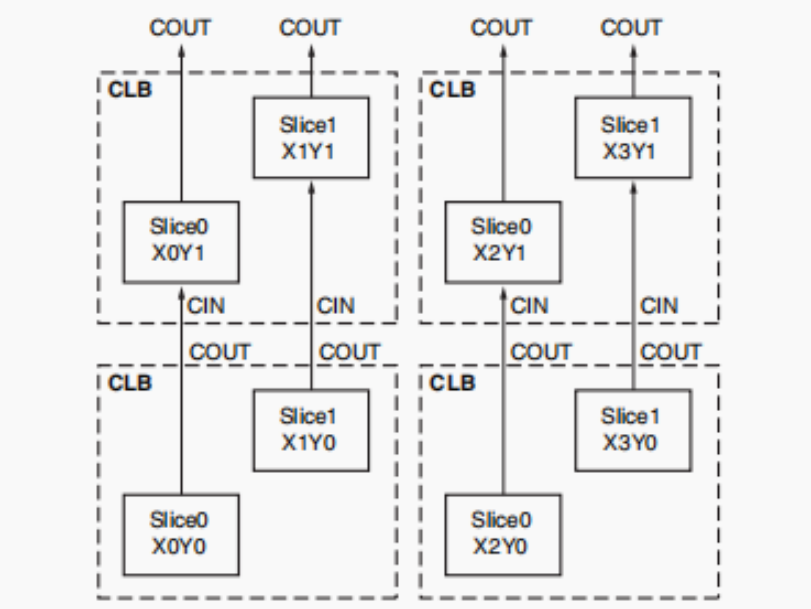


Figure A8: Relation between Rows and Columns in slices

```
set_property BEL F7BMUX [get_cells ins_2/i_0/d0]
set_property LOC SLICE_X0Y32 [get_cells ins_2/i_0/d0]
set_property BEL F7BMUX [get_cells ins_3/i_0/d0]
set_property LOC SLICE_X0Y48 [get_cells ins_3/i_0/d0]
set_property BEL F7BMUX [get_cells ins_4/i_0/d0]
set_property LOC SLICE_X36Y0 [get_cells ins_4/i_0/d0]
```

Figure A9: Design constraints example

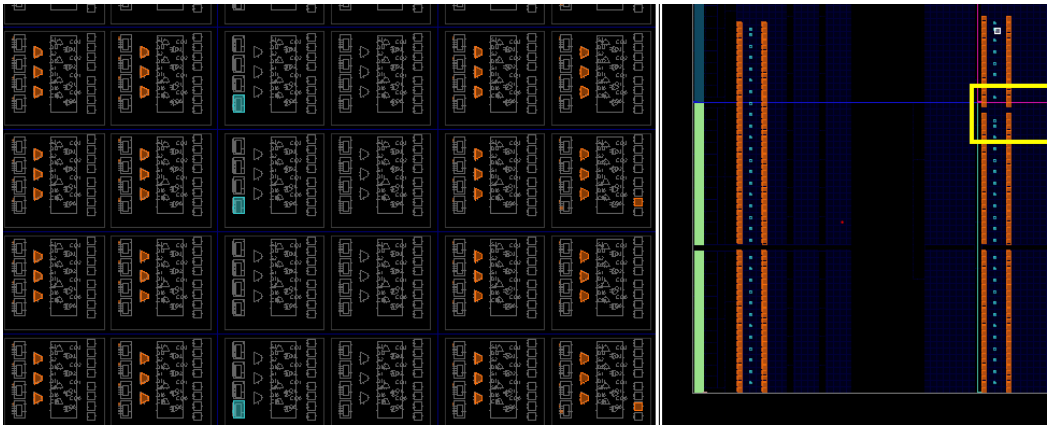


Figure A10: 8-bit Macro placement on FPGA fabric