FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

The Ring Oscillator based PUF on FPGAs

by

Filip Kodýtek

A dissertation thesis submitted to
the Faculty of Information Technology, Czech Technical University in Prague,
in partial fulfilment of the requirements for the degree of Doctor.

Dissertation degree study programme: Informatics
Czech Technical University in Prague

Prague, July 2020

Supervisor:
prof. Ing. Róbert Lórencz, CSc.
Department of Information Security
Faculty of Information Technology
Czech Technical University in Prague
Thákurova 9
160 00 Prague 6
Czech Republic

Abstract

This dissertation thesis deals with Physical Unclonable Functions (PUFs). PUFs are an important topic in hardware security as they are being increasingly used in cryptographic architectures. Since PUFs provide a digital fingerprint of the device they are implemented on, they offer an effective solution to security applications such as device identification, authentication, and secure storage of cryptographic keys. In this dissertation thesiswe provide a general description of PUFs and present numerous state-of-the-art PUF constructions focused on PUFs suitable for field-programmable gate arrays (FPGAs). Moreover, we present and compare PUF evaluation metrics and their variants since they are not always uniform across publications. We then introduce our proposal of a PUF design based on ring oscillators (ROs) and we discuss its properties. Furthermore, several measurement techniques that are suitable for the proposed design are presented. The proposed PUF design with different measurement methods is analysed and tested on different FPGA families (Spartan-3E, Spartan-6, Spartan-7) at both a stable and varying temperature and voltage. Moreover, we made several different implementations using mutually asymmetric and symmetric ROs in order to compare their behaviour mainly at a varying temperature and voltage, since there had been indications that the symmetry of ROs may have an influence on the stability of the PUF's responses. We present and discuss the results of our experiments performed on the proposed PUF design.

Keywords:

Physical unclonable function, ring oscillator, hardware security, device identification, key generation, true random number generator.

Acknowledgements

First and foremost, I would like to express my gratitude to my supervisor Prof. Ing. Róbert Lórencz, Csc. for his guidance during my studies, beginning from my bachelor thesis and continuing until this dissertation thesis. It has been many years and during that time he has been encouraging me, he provided invaluable insight and advice during my research, and he helped me with my professional advancements. And in the first place it was him who introduced me to the interesting topic that is the subject of research of this dissertation thesis.

I would also like to thank my colleague Jiří Buček for his help with performing a lot of experiments presented in this work and his helpful comments and advice during my research.

Special thanks go to the staff of the Department of Information Security, who maintained a pleasant and flexible environment for my research. I would like to express special thanks to the department management for providing most of the funding for my research.

Finally, I would like to express my deepest gratitude to my family. They always supported me in all of my efforts, not only during my studies. Most of all, I would like to thank my wife Tereza who gave birth to my amazing son, Richard, just a year before finishing this work. I would like to thank both of them for being a constant source of joy in my life.

My research was partially supported by the Ministry of Education, Youth, and Sport of the Czech Republic under research program CZ.02.1.01/0.0/0.0/16 019/0000765, "Research center for informatics (RCI)", by the Czech Grant Agency, grant No. GA16-05179S and by the Grant Agency of the Czech Technical University in Prague, grants No. SGS15/120/OHK3/1T/18, SGS16/122/OHK3/1T/18, SGS17/214/OHK3/3T/18, and SGS20/212/OHK3/3T/18.

Dedication

To my wife Tereza and my son Richard.

Contents

List of Figures

List of Tables

Abbreviations and Mathematical Notation

Mathematical Functions and Notation

$\{0,1\}$	A set of elements 0 and 1
$\{0,1\}^L$	A string (vector) of elements 0 and 1 of length L (L-bits long string)
$\|\|$	Concatenation of strings (vectors)
$\mathbf{b}$	String (vector) $\mathbf{b}$
b_i	the i-th element of string $\mathbf{b}$
$HW(b)$	Hamming weight of string $\mathbf{b}$
$HD(x,y)$	Hamming Distance of strings $\mathbf{x}$ and $\mathbf{y}$
$\oplus$	XOR operation
μ	Mean value
σ	Standard deviation
$\lfloor x \rfloor$	Rounds a real number x to the greatest integer less than or equal to x
$P(x=k)$	Probability of x being equal to k
$\binom{n}{k}$	Binomial coefficient
$\approx$	Approximately equal
[]	Closed interval
()	Open interval

Miscellaneous abbreviations

API	Application Programming Interface
ASIC	Application-Specific Integrated Circuit
BCH	Bose–Chaudhuri–Hocquenghem (code)
CLB	Configurable Logic Block
CRP	Challenge-Response Pair
DEMUX	Demultiplexer
DEPP	Digilent Adept Asynchronous Parallel Port Interface
DRAM	Dynamic Random Access Memory
ECC	Error Correcting Code
FAR	False-Acceptance Rate
FF	Flip-Flop
FRR	False-Rejection Rate
FPGA	Field-Programmable Gate Array
HD	Hamming Distance
$\mathbf{HD_{inter}}$	Inter-Hamming Distance
$\mathbf{HD_{intra}}$	Intra-Hamming Distance
HW	Hamming Weight
ID	Identifier
IP	Intellectual Property
ISE	Integrated Software Environment
LCD	Liquid Crystal Display
LISA	Longest Increasing Subsequence-Based Grouping Algorithm
LSB	Least Significant Bit
LUT	Look-Up Table
MSB	Most Significant Bit
MUX	Multiplexer
NIST	National Institute of Standards and Technology
POWF	Physical One-Way Function
PRF	Physical Random Function
PUF	Physical Unclonable Function
RFID	Radio Frequency Identification
RNG	Random Number Generator
RO	Ring Oscillator
ROPUF	Ring Oscillator Physical Unclonable Function
SDK	Software Development Kit
SRAM	Static Random Access Memory
STS	Statistical Test Suite
TERO	Transient Effect Ring Oscillator
TRNG	True Random Number Generator
USB	Universal Serial Bus
VHDL	(VHSIC-HDL) Very High Speed Integrated Circuit Hardware Description Language

Chapter 1

Introduction

In this chapter we present the main motivation and background of our work, followed by goals of this dissertation thesis and description of its structure.

1.1 Motivation and background

Electronic devices are currently an integral part of our everyday life. Such devices (for example mobile phones, smart cards, RFIDs) can be used to authenticate their owners and provide them with access to private areas or their bank accounts, to store personal data, and have many other applications. Since these devices are widespread and commonly used, they are a target for adversaries. This fact implies a problem with security. Most of the devices contain some secret information or keys that are used to authenticate their owners. Therefore this secret has to be stored in a secure manner so that the potential adversary will not be able to extract it from the device.

When designing the architecture of the device it is necessary to consider various countermeasures against possible attacks to prevent an adversary from obtaining the secret from the device. However, designing such secure architecture is not a trivial task. There are numerous possible attacks on the devices that the adversary can perform. From the perspective of hardware security, the possible threats are side channel attacks such as power analysis (simple power analysis, differential power analysis etc.), timing analysis, and also fault injection attacks. Of course, the adversary can perform other attacks than physical attacks. These include mathematical attacks (linear cryptanalysis, differential cryptanalysis etc.) on the cipher that is used, the cryptographic protocol itself, or exploit wrong implementation of the cryptographic system.

From what has been said it is obvious that secure storage and usage of the secret key is a complex task. However, for secure storage of keys we can use Physical Unclonable Functions that are able to hide the secret in a secure manner. Usually the secret keys are stored in a non-volatile memory, but is difficult to secure and therefore expensive. Non-volatile memory also tends to be vulnerable to invasive attacks, because the key is stored in a digital form. For a high level of security, the electronic devices have to be protected by

expensive circuits that are able to detect manipulation with the device and moreover, they need to be continually supplied with power. An additional disadvantage can be the cost of even basic cryptographic operations for resource-constrained platforms such as RFID chips.

These issues present one of the motivations that contributed to the deeper interest and extended development in research of Physical Unclonable Functions. Physical Unclonable Functions (abbreviated as PUFs) are being increasingly used in proposals of cryptographic protocols and security architectures. PUF is a function based on physical properties that are unique for each device. It exploits local mismatches and differences between physical components of a device arising during the manufacturing process to generate unpredictable outputs. Its concept is based on these random variations that cannot be controlled during the manufacturing process because they result from the effects of random and uncontrollable influences. Therefore it is impossible or extremely difficult to produce two identical devices with same physical properties that are used in the PUF present on these devices. It is primarily these random variations arising during the manufacturing process that play the main role in how the PUFs are used and what source of randomness they benefit from.

There is a strong similarity to human biometrics such as fingerprints, retina and others. For example we are able to identify any person by their fingerprints. Using physical properties as a fingerprint of a device, we can similarly identify the electronic devices because the physical properties are unique for each device and also are random (or unpredictable) among various devices.

PUFs have a wide spectrum of applications. For example they can be used for device identification, authentication, anti-counterfeiting, binding software to hardware platforms, cryptographic key generation, and they can also be integrated into cryptographic algorithms. Nowadays, security products based on PUFs are already being announced to the market, focusing on intellectual property protection, anti-counterfeiting and RFID applications (Verayo, Intrinsic-ID, Microsemi, QuantumTrace, Invia, ...).

1.2 Goals of the dissertation thesis

The goals of this dissertation thesis are presented in the following list:

1. Analyse existing PUF designs suitable for FPGAs with focus on ring oscillator based PUFs.

2. Improve the proposed ROPUF design of [A.3].

3. Analyse the properties of the proposed PUF design.

4. Examine and evaluate the behaviour of the proposed PUF design at both stable and varying temperature and supply voltage.

1.3 Structure of the dissertation thesis

The thesis is organized into eight chapters as follows:

1. *Introduction*: Describes the motivation behind our efforts together with our goals.
2. *Physical Unclonable Functions*: Contains the literature research on the topic of PUFs. We present the reader with a general description of PUFs, their applications, and classification.
3. *PUF Constructions*: Presents numerous PUF proposals with focus on PUFs suitable for FPGAs because that is our target platform. We mainly present PUFs based on ring oscillators as our proposal also uses them.
4. *PUF Evaluation Parameters*: Compares several approaches of PUF evaluation and provides a set of evaluation parameters used in this dissertation thesis.
5. *The Proposed Ring Oscillator Based PUF*: Explains our PUF design based on ring oscillators that was originally proposed to be used solely as a PUF, but can also be used for TRNG.
6. *Implementation*: Describes the implementation of the proposed PUF design on several implementation platforms that are used for our experiments.
7. *Experimental Results*: Presents the results of our experiments. The experiments are related to both PUF and TRNG.
8. *Conclusion*: Summarizes the results of our research, suggests our future work, and concludes the dissertation thesis.

Chapter 2

Physical Unclonable Functions

Physical Unclonable Functions are now a very popular research topic especially in the area of hardware security. In this chapter we provide a description of PUFs in general and the properties we may require of them. Finally, we will present possible PUF classification and applications.

2.1 Description of PUFs

Nowadays, we can find many scientific papers focusing on the topic of PUFs. That is why we can encounter multiple definitions of a PUF. In general, it can be said that PUF is a function that is realised within some physical system and expresses its inherent and instance-specific features [35], thus it is strongly similar to human biometric features.

The first description of the general PUF concept can be found in Pappu's dissertation thesis written in 2001 [47]. Pappu used a term POWF (Physical One-Way Function) and defined it as a function that is easy to compute but hard to invert and the underlying physical system is difficult to clone. Therefore simulating the physical interaction is computationally demanding. The next used term denoting a new PUF construction was PRF (Physical Random Function) proposed by Gassend et al. [18]. To avoid confusion with the term Pseudo Random Function (also abbreviated as PRF) they used the term PUF.

PUF is now a widespread term and various constructions and concepts that share a number of properties are called PUFs. Some of these constructions or concepts were proposed earlier than the term PUF first was used and therefore they were not denoted as PUFs from the beginning. In other cases, the proposal of some construction that could be qualified as PUF was made in other research areas than hardware security and there this term is unknown.

As the term PUF indicates, PUF is a function that is unclonable. The concept of PUFs is based on random variations arising during the manufacturing process which causes each device to possess unique physical properties. These physical properties are, for example, circuit delay, or bias of memory cells after power-up to some certain value (0 or 1). The

variations of physical properties arising during the manufacturing process are random since they arise from the influence of random and uncontrollable effects.

Since PUF is a function, it should have some characteristics of functions; given an input we should obtain the corresponding output (in this text, we will often use term challenge instead of input and response instead of output). It maps any input (challenge) to its corresponding output (response), forming challenge-response pairs (CRPs). However, it is not strictly a mathematical function because a PUF can produce multiple different outputs for one input and it can even produce one output from several inputs. This behaviour may be caused by random variations that can be caused by various physical conditions [24]. A more fitting mathematical description of a PUF is a probabilistic function, where part of the input is an uncontrollable random variable [35].

In summary, PUF is a function that gives us, for a given challenge, a corresponding response. These responses may change in time in dependence on physical conditions; however, the responses should be similar enough so that we can recognize that the response we obtained belongs to the given challenge. At the same time, we require the PUF responses to be unique among different devices. This means that for the same challenge we obtain a different response from each device. The difference between these responses should be sufficiently large because based on these responses we can identify or authenticate the devices. The uniqueness applies also to responses from one device, but produced by a PUF for various challenges.

Both of these requirements (similarity of responses from one device and uniqueness of responses from various devices or different challenges) imply that we need the PUF responses to be both stable and unique. This is the main difference compared to TRNG (True Random Number Generator). The purpose of TRNG is to produce a sequence of bits that are random and even if we know a large sequence of these bits, we should not be able to predict the following bits. In the case of PUF, we need the PUF responses to behave like random sequences of bits from the perspective of population (devices) so that if we know a large number of responses from a large population of devices, we should not be able to predict a response for a given challenge from some unknown device. This condition also holds for various challenge-response pairs of one device. Therefore, responses should be random from the viewpoint of device population and also for different challenges, but not for the same challenge applied to one device repeatedly. In summary, even though PUF and TRNG may have a common basis because they both exploit physical properties of some given device, the source of randomness for PUFs and TRNGs is very different. TRNGs exploit continuous real-time random behaviour of the hardware they are implemented on, while PUFs benefit from the randomness that occurs only once during the manufacturing process.

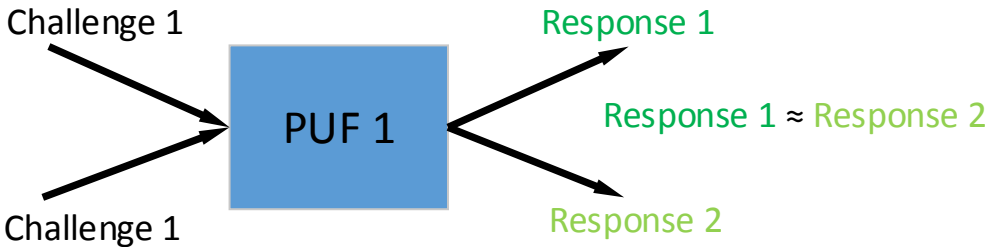


Figure 2.1: PUF has to produce the same or at least similar responses for the same challenge on the same device.

2.2 Properties of PUFs

This section provides an overview of the properties that are sensible for PUFs. Some of the presented properties are necessary and they define PUFs while others are only considered "nice to have" properties and are not guaranteed for PUF construction [35].

Constructibility

A necessary condition for a PUF and all of its properties is constructibility. We can hardly discuss the remaining properties of PUFs were they not practically feasible. Constructibility requires the PUF proposal to be at least feasible within the laws of physics. However, from a more practical viewpoint, constructibility is related to the cost of producing the PUF. It also makes a considerable difference if we require for the produced PUF to have some particular challenge-response behaviour. Producing a *random* PUF without any specific requirement on its challenge-response behaviour should be easy. Conversely, if we want to construct a specific PUF with defined challenge-response behaviour, it can be infeasible to construct. This implies that this property is strongly related to physical unclonability.

Evaluability

A PUF is considered to be evaluable if for any random challenge it is "easy" to evaluate a corresponding response. Since PUF exhibits a challenge-response behaviour, this property is necessary for a PUF to achieve because it would be difficult to discuss any properties of a PUF that is not evaluable. However, the "easiness" is context dependent. From a theoretical perspective this refers to polynomial time and effort. In practice it means that it is evaluable in terms of timing, area, power, energy, and cost.

Reproducibility

For a given PUF and challenge on one chosen device we should obtain the same response with a high probability when the challenge is evaluated repeatedly. Reproducibility is one of the properties that puts constraints on a PUF's challenge-response behaviour. The responses of the PUF are influenced by varying physical conditions, therefore some errors

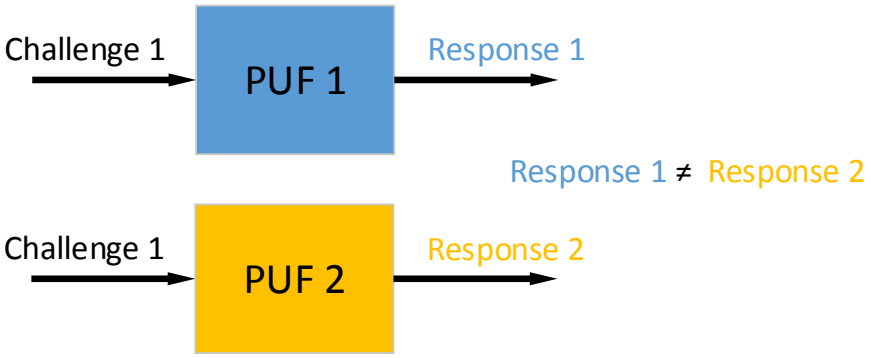


Figure 2.2: PUF has to produce different responses for the same challenge on different devices.

may occur in them when they are obtained repeatedly for the same challenge. For this reason we consider a response with sufficiently small number of errors as the same response. Similarity of the PUF responses is evaluated based on the considered distance metric (usually Hamming distance of the bit strings that represent the outputs). The concept is depicted in Fig. 2.1.

Uniqueness

As in the case of reproducibility we consider one given PUF and challenge but we observe the responses obtained from different devices, not only from one device (see Fig. 2.2). The responses resulting from evaluating the same challenge on different devices should be different enough (dissimilar) with a high probability. Again, the similarity of the PUF responses is evaluated according to the used distance metric.

Physical unclonability

This property is crucial for PUFs. Since a PUF is based on random variations arising during the manufacturing process due to the influence of random and uncontrollable influences, it is infeasible to manufacture two identical devices containing PUF that would exhibit the same challenge-response behaviour. The infeasibility is related to the physical and technical difficulties in manufacturing such pair of identical devices.

The property of physical unclonability has the security advantage that even the manufacturer who may influence the manufacturing process cannot break the uniqueness property since there are uncontrollable influences that interfere with the manufacturing process. Therefore, it is not necessary to trust the manufacturer to be sure that every device containing a PUF is unique with a high probability because this is implied by the physical unclonability of PUFs.

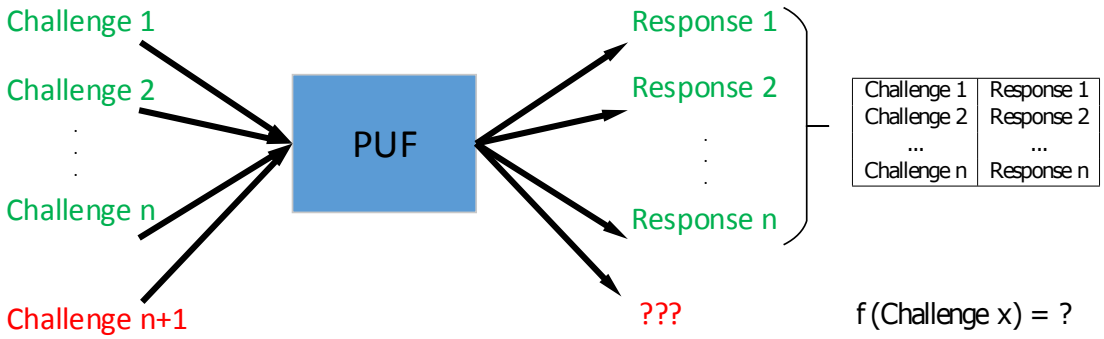


Figure 2.3: PUF's responses for new challenges should be unpredictable even when some challenge-response pairs are observed.

When we combine this property with constructibility, we can say that it is easy to create a PUF with arbitrary and random challenge-response behaviour, but it is infeasible to create a PUF with a specific challenge-response behaviour.

Unpredictability

Numerous PUF applications rely on their challenge-response functionality, which is to send some challenge and to obtain random (but corresponding to given challenge) response. In this sense, neither uniqueness nor physical unclonability sufficiently guarantee security. It is necessary to achieve unpredictability of the PUF responses to ensure the randomness of PUF responses for an adversary even if the adversary has already observed a number of challenges and their responses. Unpredictability means that the adversary should not be able to build a model based on observed challenge-response pairs that would predict responses for a new challenge (see Fig. 2.3).

Mathematical unclonability

In case of unpredictability, it was assumed that an adversary learned a limited number of challenge-response pairs that he used to predict other responses. This is usually the case when the adversary eavesdrops on a communication when a challenge-response based protocol is used. However, a situation may occur where the adversary has unlimited physical access to a device containing a PUF and therefore can obtain as many challenge-response pairs as they are able to store. If the responses remain unpredictable, one can say that the mathematical unclonability property is achieved. A necessary assumption for mathematical unclonability is the fact that the set of possible challenge-response pairs is so large (preferably exponential) that it is beyond the capacity of any adversary to store the whole challenge-response set.

Mathematical unclonability takes into account a stronger adversarial model, where the adversary has unlimited physical access to a PUF; hence it is the extension of unpredictability. It is obvious that mathematical unclonability implies unpredictability.

True unclonability

Two different notions of unclonability were already defined in this section. They are physical and mathematical unclonability. Both of them have the same goal, which is to ensure that a PUF cannot be cloned. But they have different perspectives. Physical unclonability deals with the infeasibility of creating a clone of a specific device containing a PUF with the same challenge-response behaviour. Mathematical unclonability addresses only the cloning of the challenge-response behaviour of a chosen PUF, but does not clone the physical device itself. The true unclonability property is achieved by both physical and mathematical unclonability at the same time.

One-wayness

The one-wayness property is defined similarly to the definition of physical one-way functions proposed by Pappu [47]. A PUF exhibits one-wayness if it is evaluable and there exists no efficient inversion algorithm that finds a challenge based on a given response that produces similar response to the given one. This definition resembles to the definition of one-way functions but it takes into account the unreliability and uniqueness of PUFs.

Tamper-evidence

Tampering is the alteration of the physical integrity of some given circuit, in this case a PUF. The intent is to modify the circuit's operation in an unauthorized and harmful manner. It is usually used to remove or bypass protection mechanisms to obtain confidential data, and is therefore a powerful attack against security implementations. Hence it is essential to detect tampering and to provide an appropriate reaction, such as clearing confidential data or blocking all functionality.

In order to detect any tampering attempt, a security system needs to have some tamper-evidence. This means that tampering will have an unavoidable and observable impact on the system. In the perspective of PUFs, tamper-evidence means that it is very hard to physically alter a PUF without any noticeable effect on its challenge-response behaviour. Ideally, the alteration would cause the PUF to become a completely different one.

2.3 PUF classification

The PUF constructions were proposed for a large variety of technologies, material and platforms. Therefore, we can classify PUFs based on the nature of their features (electronic components, glass, silicon integrated circuits) or even on their sources of random-

ness [35, 37]. Another classification can be the division of PUFs into *intrinsic* and *non-intrinsic* PUFs in dependence on the source of measurement and the origin of their random features [35, 37]. Ultimately, PUFs can be classified based on the security properties of their challenge-response behaviour, i.e. *weak* and *strong* PUFs [20].

2.3.1 Electronic and non-electronic PUFs

The terms electronic and non-electronic PUFs corresponds to the nature of the components that are used for a PUF and contribute to its randomness and uniqueness. These terms are not related to the processing methods or measurements that can be performed with the help of electronic equipment.

The first class of PUFs are non-electronic PUFs. Their properties are based on non-electronic technologies or materials such as light scattering characteristics of an optical medium. The term non-eletronic in this case reflects the non-electronic physical basis of the PUF and not the way PUF responses are handled.

The opposite of non-electronic PUFs are electronic PUFs that exploit random variations in the electronic characteristics of electronic components or circuits. These characteristics are, for example, resistance, capacitance, delay etc. Furthermore, some of the electronic PUFs have their basic operations consisting of an analog measurement of some electric or electronic features [37] while other PUF proposals perform the measurements digitally. Therefore, electronic PUFs may also be distinguished from this perspective.

A large subclass of electronic PUFs are silicon PUFs that are the most popular in security solutions since they can be used in cryptographic implementations directly on an integrated circuit. Chapter 3 is focused on this type of PUFs.

2.3.2 Intrinsic and non-intrinsic PUFs

Another possible classification of PUFs is based on the construction properties of PUFs. PUFs are divided into intrinsic and non-intrinsic PUFs. According to [35, 37], two following conditions need to be met for a PUF to be classified as intrinsic PUF:

1. The PUF together with the measurement equipment should be embedded in the device and its evaluations should be performed internally by the embedded measurement equipment.

2. Its random features are implicitly introduced during the manufacturing process.

Since there are some practical and security advantages to intrinsic PUFs, both of these conditions are discussed in the following text. Regarding the first condition, we can distinguish between two forms of PUF evaluation, i.e. external and internal evaluation. In case of external evaluation, the measurements are performed using external instruments and the measured features have to be externally observable. Internal evaluation assumes that the necessary equipment used for the measurement of random features of a device is embedded in the device together with the PUF. However, this implies a possible disadvantage for

an internal evaluation because the embedded measurement equipment needs to be trusted since it might be impossible to verify the measurements externally.

One advantage of performing the evaluations internally is its practicality. Internal evaluations can be more accurate since they avoid external influences that possibly cause measurement errors. More importantly, the device containing a PUF can evaluate itself without any restrictions since the necessary measurement equipment is embedded in the device.

The second advantage of internal evaluations is associated with security. When all evaluations are performed internally, the PUF response remains in the device and can be considered as internal secret that can be used for example as a key (in case of a PUF used for key generation). This is useful when the PUF responses are used immediately for embedded cryptographic applications.

The next discussed condition is related to the source of randomness that is measured during the PUF evaluation. The randomness used by the PUF can be introduced implicitly to the device during the manufacturing process and form an integral and inseparable part of the PUF. The measured random features are caused by uncontrollable effects during the manufacturing process. The randomness may also be introduced by an explicit procedure during the manufacturing process with the sole purpose of introducing random features that will be used by PUFs. This condition implies that in case of an intrinsic PUF, no extra manufacturing steps are required during the manufacturing process [37].

The advantage of implicit random variations is that there is no extra overhead and additional cost since they are an inherent part of the manufacturing process. Introducing the randomness explicitly usually comes with extra cost. However, the main advantage of implicit random variations lies in the perspective of security. The implicit randomness is caused by random variations arising during the manufacturing process and they are considered undesirable because they may have a negative impact on the manufactured device. Therefore manufacturers apply countermeasures against these process variations to reduce the effect of various random influences. However, it is technically impossible for the manufacturers to completely eliminate all random effects in the manufacturing process. This implies an interesting security advantage of PUF construction based on implicit process variations: Even though the manufacturer has control over the manufacturing process, they are not able to control or eliminate the random features present in their manufactured devices that are later used by PUFs.

Intrinsic PUFs can be divided into two classes based on their basic operations. The two major classes of intrinsic PUFs that are also suitable for field-programmable gate arrays (FPGAs) according to their sources of randomness are delay-based and memory-based PUFs. Since many electronic devices have embedded SRAM [22, 50], a very common PUF design is based on SRAM (static random-access memory) that it uses as a source of randomness. This PUF is based on the content of SRAM after power-up. However, some FPGAs initialise their memory after power-up, so all randomness is lost. That led to proposals of other memory-based PUFs such as the Butterfly PUF [28], the Latch PUF [57] and the Flip-flop PUF [36].

Delay-based PUFs exploit random variations in delays of logic gates and interconnects.

One of the first delay-based PUFs was the Arbiter PUF [30]. Other examples are the Ring Oscillator PUF (ROPUF) [19, 38, 58] and the Glitch PUF [59].

The Optical PUF [47] and the Coating PUF [62] introduced in the next Chapter 3 do not meet the conditions of intrinsic PUFs and they are some of the best known PUF constructions from the other class. The Optical PUF is non-intrinsic because its evaluation is performed externally by observing the speckle pattern. Also its random features are explicitly introduced by the random placement of the light scattering particles in an optical medium. The Coating PUF is classified as non-intrinsic despite its ability to be evaluated internally because its randomness is introduced explicitly during athe manufacturing process by covering an integrated circuit with a protective coating with random dielectric particles.

2.3.3 Weak and strong PUFs

The last classification we will introduce in this report is based on the security properties of the challenge-response behaviour of PUFs [20]. From this perspective we can distinguish between weak and strong PUFs.

Weak PUFs can be considered a digital fingerprint of some given circuit. PUFs from this class usually have a very small challenge-response set or even only one challenge (or no challenge, only some stimulation that starts the PUF evaluation to generate the fingerprint) in some extreme cases. An example of a weak PUF is the SRAM PUF which is based on reading the memory content after device power-up. Each SRAM cell will have bias to some certain value caused by the manufacturing variability and this variability is random in the entire SRAM and also in the whole population of devices. The PUF response will be the memory content of the SRAM after power-up, but there will be no challenge, since the only challenge is powering the SRAM on. However, we can also consider some address of the SRAM as a challenge of the SRAM PUF. The fact that the challenge-response set is small implies that these challenge-response pairs must be kept secret. A typical application of weak PUFs is a cryptographic key generation.

The opposite of weak PUFs are strong PUFs. The main difference of strong and weak PUFs is the number of supported challenge-response pairs. Strong PUFs offer a large challenge-response set. The requirements for a strong PUFs are a large challenge-response set (large in this case means that ideally it should be exponential in the number of challenge bits), so that an adversary is not able to store all challenge-response pairs, and the unpredictability of PUF responses even with the knowledge of a large number of challenge-response pairs. It is not feasible to build a model of the PUF based on the observed challenge-response pairs. If these requirements are not met, the PUF is classified as weak PUF.

Typically, the application of strong PUFs is authentication, where a device containing PUF is authenticated by a query with different challenge each time and comparing its response with the one stored in a database. After each usage of some challenge-response pair, this pair is deleted and never used again. However, it turned out that constructing a practical (intrinsic) strong PUF with strong security properties is a very difficult task [35].

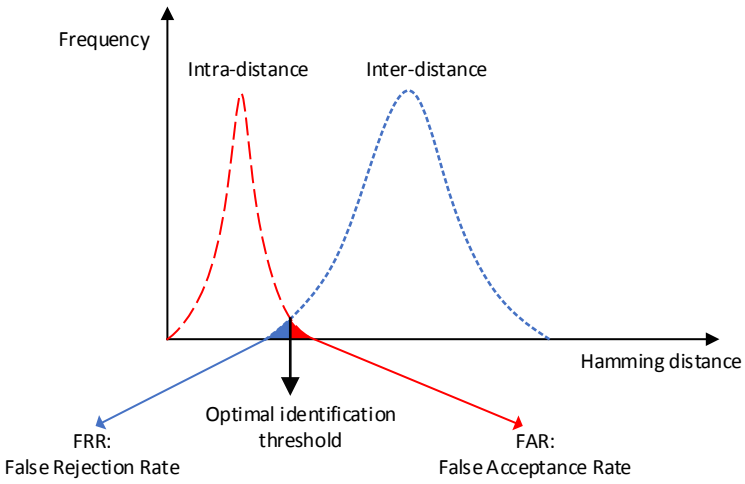


Figure 2.4: The principle of identification. The red curve represents the Hamming distance between the PUF responses from the same devices, while the blue curve shows the Hamming distance between the PUF responses from different devices.[37]

2.4 Applications of PUFs

Due to its properties, PUF is suitable to be used for example for identification and authentication purposes. This section gives an overview of three possible applications of PUFs. First we will describe device identification, then authentication, and finally the concept of cryptographic key generation.

2.4.1 Device identification

Device identification is the most basic application scenario of PUFs and it is also an inherent feature of a PUF. It is used to identify a physical object (device). Just like we are able to identify any person by their fingerprints, we can identify any device based on its individual and unique physical characteristics which a PUF uses for its functionality. This is very similar to a biometrical identification scheme.

Since there are errors present in the PUF responses because it is influenced by varying physical conditions, the PUF responses produced by a PUF on the same device will not be the same every time. However, for the identification purposes we do not have to worry about the errors in the PUF responses, provided that the PUF responses from the same device will be sufficiently similar and also different enough from the responses produced by PUFs on other devices at the same time.

During the identification process, a PUF generates a response, which is then compared to responses from various devices stored in a database. If the response is similar to one

of the responses stored in the database and it is also different enough from the other responses, the identification process is successful. The similarity of two responses is usually determined by their Hamming distance. For a successful identification of a device based on its response, the following conditions need to be met:

1. For a given response from a specific device, a response in the database is found. For example the Hamming distance between these two responses is less than the chosen threshold.

2. The Hamming distance between a response from a given device and the responses from all other devices stored in the database is larger than the chosen threshold.

The ideal value of average Hamming distance between the responses from all devices is 50%. The principle of identification and determination of the identification threshold is shown in Fig. 2.4 [37]. This figure shows the curves of two metrics. They are Intra-device Hamming distance and Inter-device Hamming distance. Intra-device Hamming distance represents the Hamming distance of the responses generated by one device, while the Inter-device Hamming distance is the Hamming distance between the responses generated by different devices. In other words, the Intra-device Hamming distance represents the bit error rate of the PUF responses and the Inter-device Hamming distance shows how the PUF responses from various devices are different.

As Fig. 2.4 shows, if the curves do not overlap, an errorless identification can be made by placing the identification threshold somewhere in the area between both curves. However, when the curves partially overlap, setting the identification threshold is a trade-off between false-acceptance rate (FAR) and false-rejection rate (FRR). The optimal choice of the identification threshold is achieved by placing the threshold at the intersection of both histograms [37], minimizing the sum of FAR and FRR.

2.4.2 Authentication

In the case of authentication, any subject that wants to authenticate itself to the other party has to provide some sort of proof of its identity. For example, a subject can identify itself by a secret that only the subject knows. In addition, the subject has to demonstrate that it participated in the creation of the proof that confirms its identity.

Authentication is realised based on challenge-response pairs. The authentication scheme benefits from the uniqueness and unpredictability of the PUF responses. One of the possible authentication schemes is shown in Fig. 2.5; it consists of two phases:

1. An ID of each subject is stored and then a sufficient number of challenge-response pairs are collected from its PUF. The challenges are generated randomly. The collected challenge-response pairs are stored in a database to the corresponding subject ID.

2. At the beginning of the authentication process, a subject identifies itself by sending its ID (this does not have to be necessarily generated by its PUF). After the ID

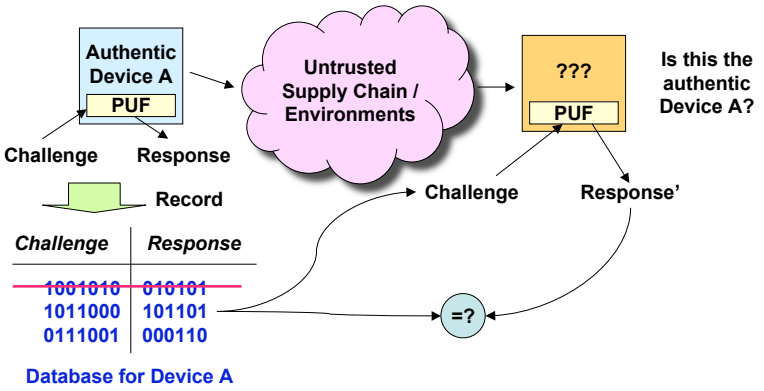


Figure 2.5: Authentication scheme using physical unclonable functions.[58]

is found in the database, one of the stored challenge-response pairs is selected for the corresponding ID. The challenge is sent to the subject that generates a response using PUF and sends back the response. If the response is similar enough (the Hamming distance is less than the chosen threshold) to the response stored in the database (for the selected challenge), the subject is successfully authenticated. The challenge-response pair that was used for authentication is then deleted and never used again.

Since the challenges and responses are sent in an insecure manner, any third party can eavesdrop the communication and potentially use the captured challenge-response pairs. Therefore, there is a threat of a man-in-the-middle attack. To prevent this attack, all challenge-response pairs that were already used in the authentication process are deleted. Moreover, it is important for the PUF to be unpredictable: An adversary should not be able to predict a response of the PUF for a given challenge based on previously eavesdropped challenge-response pairs.

2.4.3 Cryptographic key generation

A number of security applications depend on cryptographic keys. These keys are usually stored in a non-volatile memory. This, however, raises a problem of how to store a cryptographic key in a secure manner so that the key is hidden from a potential adversary. Solutions to this issue are usually complex and expensive.

PUFs offer a cheap and an efficient solution to the issue of secure storage of cryptographic keys. Instead of storing the secret key in memory, the keys are generated by a PUF at the moment they are needed. However, as mentioned before, the PUF responses tend to contain errors and they are not the same when repeatedly generated due to varying physical conditions and random noise. Therefore, the PUF responses have to be stabilised before they are used as keys. This is usually achieved by the application of error correcting codes (ECC) that correct the wrong bits in the response. The generated key always has to be the same, otherwise we would obtain a different result by deciphering enciphered

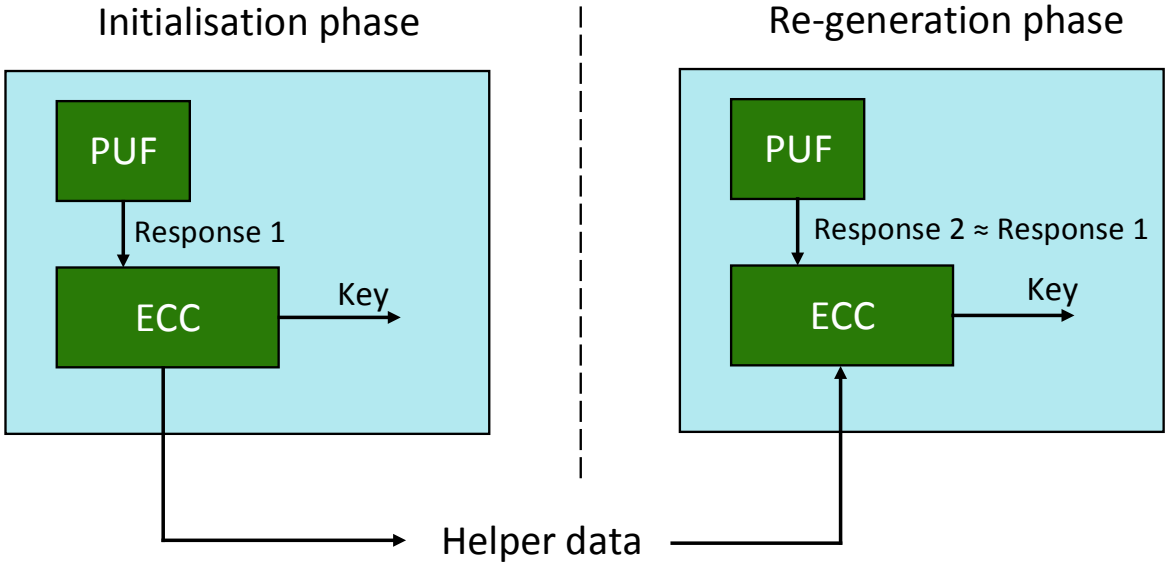


Figure 2.6: Principle of cryptographic key generation.

data even if there was only one erroneous bit in the key. The PUFs are able to generate random, unpredictable and stable keys when combined with error correcting codes.

In general, the process of key generation using PUFs is divided into two phases. During the *initialisation phase* a PUF generates the key and the error correcting code produces some *helper data* which are later used to correct the PUF response. The helper data does not necessarily only contain information required by the error correcting code, but it can also contain some additional information needed by the PUF (for example the configuration of the PUF). Since the helper data is usually public, it should not be possible to retrieve the key based on the content of helper data.

The second phase, called *re-generation phase*, re-generates the key when an application needs it. First the PUF generates a response, which is then processed by an error correcting code. The error correcting code corrects the PUF response with the help of the helper data that was produced in the initialisation phase. After the correction of the PUF response, the same key as in the initialisation phase is obtained. The key generation process is shown in Fig. 2.6.

This type of key generation is only one of many possible methods. In other variants we may encounter e.g. the usage of hash functions that are applied on the PUF output after it was corrected by the error correcting code.

Chapter 3

PUF Constructions

In this chapter, we provide a description of various PUF constructions that are focused on *intrinsic* PUFs with two exceptions which are the Optical PUF and the Coating PUF. The following list of PUF constructions is not complete, since there is a considerable amount of PUF constructions. We focus primarily on intrinsic PUFs suitable for FPGAs. We put an emphasis on PUFs based on ring oscillators since our proposal uses them.

3.1 Optical PUF

We can encounter one of the first PUF designs in Pappu's dissertation thesis written in 2002 [47]. At that time, the term *physical unclonable function* was not known and the Optical PUF was classified as a *physical one-way function*.

The main component of the Optical PUF is a transparent optical medium (optical token) filled with a large amount of light scattering particles [4]. When a laser beam shines on the optical medium, a unique and random speckle pattern arises. The basic concept of the Optical PUF is a very complex interaction between the laser beam and the light scattering particles. The source of randomness in this PUF construction is the random placement of the light scattering particles in the optical medium during the manufacturing process. The resulting speckle pattern is recorded and encoded (for example by Gabor hash [17]) into a bit string representing the PUF response.

An exact position and angle of the laser beam is an input (challenge) for the Optical PUF. Even a minute-long change in the relative orientation of the laser beam and the optical medium result in a completely different speckle pattern [37]. The basic principle of the Optical PUF is depicted in Fig. 3.1.

The Optical PUF is classified as a Strong PUF, meaning that it has a large space of challenge-response pairs. It has a high resiliency to both modeling attacks and physical cloning attacks due to its high complexity [51]. However, it usually requires an optical precision mechanism that establishes exactly the same positioning of the light scattering token and the laser beam [47]. Such equipment is expensive and may be error prone. Such

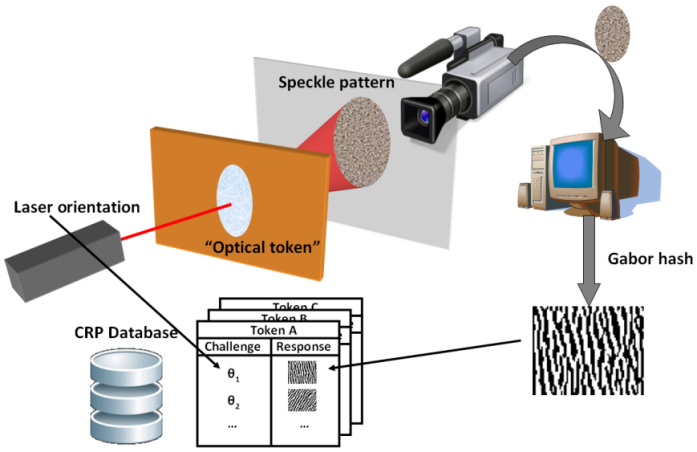


Figure 3.1: Basic operation of an Optical PUF. The setting of the laser beam serves as a challenge to the Optical PUF. The laser beam shines on the optical token, which is a transparent material filled with light scattering particles. After shining on the optical token, a unique speckle pattern is obtained and then encoded by Gabor hash into a bit string used as a response of the Optical PUF.[37]

Optical PUF design that requires an additional optical precision mechanism can be called a *non-integrated* Optical PUF. Rührmair et al. [51] present an *integrated* Optical PUF.

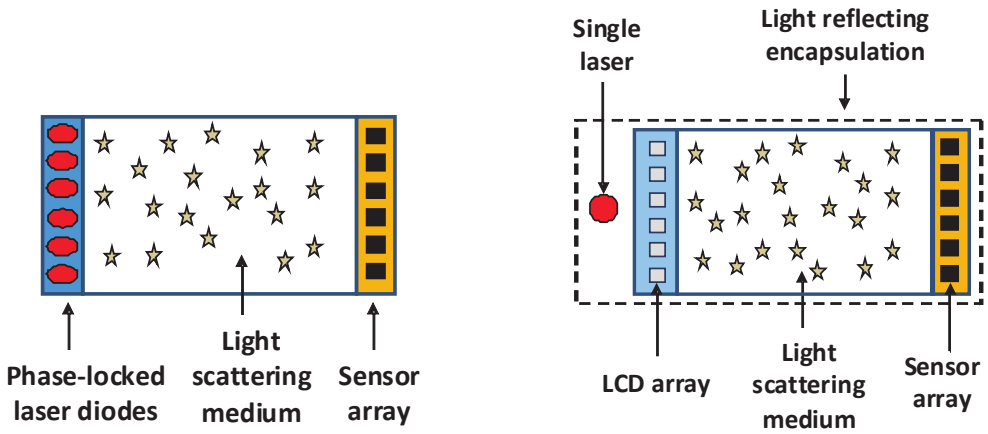
Fig. 3.2(a) and Fig. 3.2(b) show two possible approaches of designing an integrated Optical PUF according to [51]. The main goal is to avoid the usage of moving components for the light scattering token and the laser beam positioning while also allowing an exponential number of different challenges. Fig. 3.2(a) shows an immobile laser diode array with k phase-locked diodes that is used to excite a random scattering medium. The diodes can be switched on and off independently, thus allowing 2^k challenges. Sensors are used to measure the resulting light intensities locally.

The second approach, depicted in Fig. 3.2(b), uses a single laser source with a subsequently placed light modulator (LCD array) instead of phase-locked diode arrays. The k pixels of the LCD can be switched on and off, again leading to 2^k challenges.

The integrated Optical PUF was found to be vulnerable to modeling attacks when a linear scattering structure is used and the adversary has direct access to the resulting speckle images [51]. Therefore, non-linear scattering materials need to be used in this PUF type.

3.2 Coating PUF

The concept of the Coating PUF, introduced by Tuyls et al. in [62] consists of covering an integrated circuit with protective coating. The coating material is filled with random dielectric particles, meaning they have random sizes, shapes and locations in the coating layer. Below the coating layer, comb-shaped metal wire sensors are used to measure the



(a) Laser diode array with phase-locked diodes.

(b) A single laser source with light modulator (LCD array).

Figure 3.2: Two possible theoretical types of integrated Optical PUFs.[51]

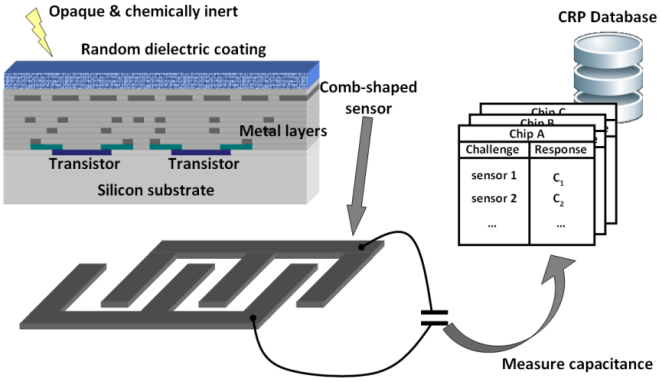
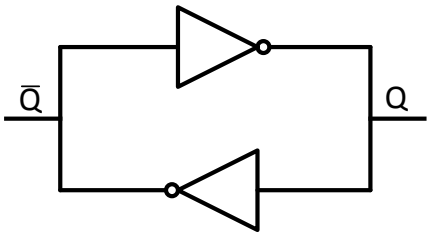


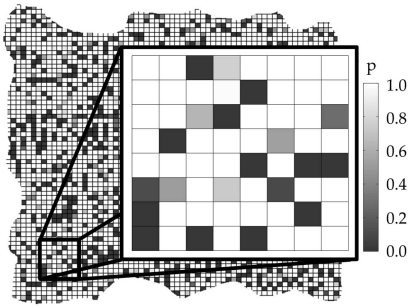
Figure 3.3: Coating PUF.[37]

local capacitance of the coating [48] as shown in Fig. 3.3. Measuring the Coating PUF from the outside gives different capacitance results since the measurement is very sensitive to the precise location of the dielectric particles.

The Coating PUF does not rely on random effects of the manufacturing variability. It uses the random elements explicitly introduced by passive dielectric coating sprayed directly on the top of the sensors. Coating PUFs offer strong protection against physical attacks such as tampering since the protective coating is opaque and chemically inert. By any physical intervention into the protective layer a change in the capacitance of the layer occurs, resulting in a completely different behaviour of the Coating PUF. Note that implementing a Coating PUF requires an additional manufacturing step, this does not increase the price of the product dramatically [48].



(a) SRAM cell.[37]



(b) SRAM content. The shade of grey illustrates the probability of state 1.[22]

Figure 3.4: Concept of SRAM PUF.

3.3 SRAM PUF

SRAM (static random-access memory) is a static memory based on bistable flip-flops that are used to store data. Fig 3.4(a) shows an SRAM cell, logically constructed as two cross-coupled inverters. This circuit has two possible stable values (0 and 1) that represent the binary value stored in the cell [35].

The principle of an SRAM PUF operation is based on the memory content after power-up. Since the preference of each memory cell cannot be influenced and their preference is random and independent, they are a suitable source of randomness for PUFs. Large SRAM memories are capable of storing many kilobits or megabits that can be used for a PUF. A memory address in SRAM PUF can be considered as the challenge for the PUF.

A key property of an SRAM cell in terms of PUF characteristics is bistability [49]. Each SRAM cell prefers a different state after power-up. Some memory cells have bias to binary 1, while other cells have bias to binary 0. However, some of the memory cells do not have bias to any of the two binary values. The distribution of these three types of SRAM cells over the whole memory is random [37]. The memory cells with strong bias to one of the two binary values that in most cases stabilise in the same value are considered stable memory cells. Memory cells with no real preference that usually have different states over time are called unstable cells. The bias toward a certain value of memory cells after power-up is caused by random physical mismatches in the memory cells that originate from the manufacturing process.

Stable memory cells allow us to identify various devices when used for an SRAM PUF while the unstable memory cells cause the errors in the PUF output. Fig. 3.4(b) shows an example of an SRAM map with probabilities of binary value 1 occurring in the corresponding cell. Ideally, we want each cell to be 100% stable. If we want the PUF output to be stable, we need to perform measurements repeatedly in order to select only the stable SRAM cells. However, such measurements would increase the manufacturing costs and they would have to be performed at varying physical conditions such as temperature or voltage.

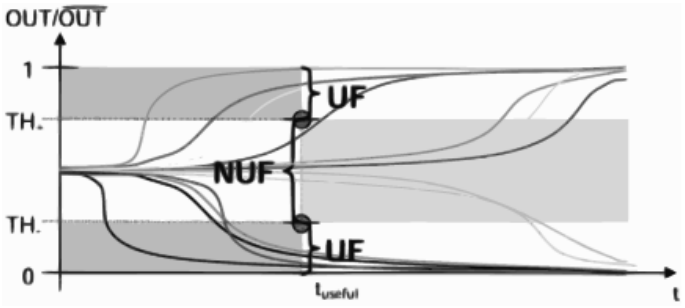


Figure 3.5: Measurement of decision time (UF: useful, NUF: not useful).[49]

Another option of obtaining a stable output is to select cells that settle in their final state faster after power-up [49]. The concept of this approach is shown in Fig. 3.5. All the cells with the decision time under t_{useful} and lie under a lower threshold or above the upper threshold are considered useful. The remaining cells are not used at all.

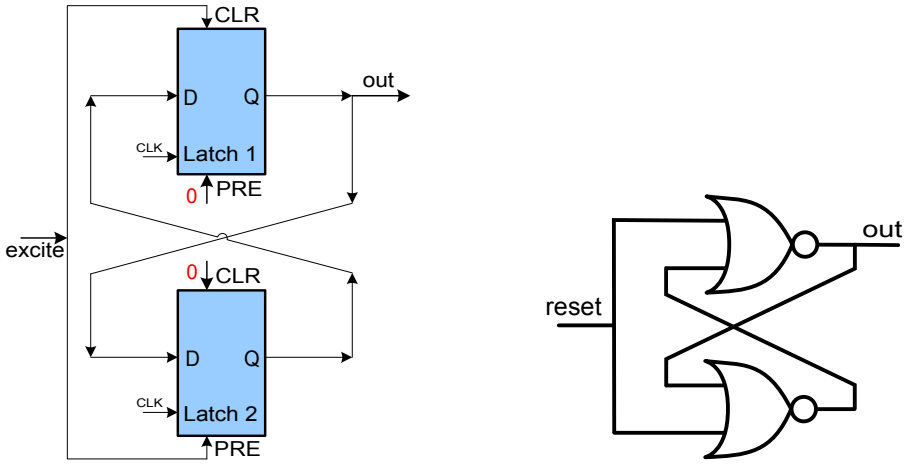
Laban et al. [29] proposed a method of selection of stable bits using a mask that is then corrected by a BCH code, significantly reducing the error rate of the PUF output. However, this approach still requires to detect the stable bits in advance.

3.4 Butterfly PUF

The SRAM PUF is often impossible to implement on some FPGA platforms because the SRAM is initialised to predefined values. Another disadvantage of the SRAM PUF is the fact that to generate the PUF output, the SRAM needs to be read after power-up before the memory is overwritten and the randomness is lost. These drawbacks were the main motivation for the proposal of the Butterfly PUF [28], which is based on a cross-coupled circuit and can be implemented on any FPGA.

The Butterfly PUF concept consists of simulating the SRAM PUF behaviour after power-up, when the SRAM cells stabilise on particular values. The basic building element of the Butterfly PUF is a circuit made of two cross-coupled latches, simulating the SRAM cell. This structure can be forced into an unstable state after which the structure converges back to one of the possible stable states.

To ensure a proper behaviour of the Butterfly PUF, it is necessary to achieve the best possible symmetry of the interconnects between the two latches [43] as shown in Fig. 3.6(a). The interconnects between the outputs Q and the inputs D between both latches have to be symmetric. When set to high, the signal *signal* starts the Butterfly PUF cell operation. When the input signals PRE (preset) and CLR (clear) are set to 1, the value of the output Q is changed to 1 or 0. Signal CLK (clock) transfers the input signal D to the output Q. Signal CLK is always set to high in order to simulate a combinational loop. By setting the signal *excite* to 1, the structure is forced into an unstable state because of the cross-coupled latches where the output Q of each latch is transferred to the input D of the other latch. After a few clock cycles the *excite* signal is set to low and the Butterfly PUF cell



(a) Structure of Butterfly PUF cell.[28] (b) Logical circuit of a Latch PUF cell.[37]

Figure 3.6: Butterfly and Latch PUF.

will stabilise in one of the possible states. The resulting state depends on the slight delay differences of the interconnects between the two latches. This will be different among various devices and positions on the FPGA [28].

3.5 Latch PUF

A method of identifying integrated circuits based on latches realised by two cross-coupled NOR gates was introduced by Su et al. [57]. A simple circuit used as a latch for the Latch PUF is shown in Fig. 3.6(b). This concept is very similar to SRAM PUF; however, the SRAM PUF cells are in an unstable state at the beginning before they settle on the resulting value. Latches in case of the Latch PUF are in a stable state and they are brought into an unstable state using the *reset* signal. The resulting value is derived based on the random internal mismatches of the electronic components.

The advantage of the Latch PUF compared to the SRAM PUF is that, similarly to the Butterfly PUF, it does not depend on the power-up of the device. The PUF output can be obtained whenever it is needed. This implies that when the device is powered up, it is not necessary in order to store the PUF output – we can generate it anytime.

3.6 Flip-flop PUF

As in the case of the SRAM PUF, the Flip-flop PUF depends on the power-up of the device, but it uses D-flip-flops instead of SRAM cells. The Flip-flop PUF was proposed by Maes et al. [36] as a replacement of the SRAM PUF on FPGA boards.

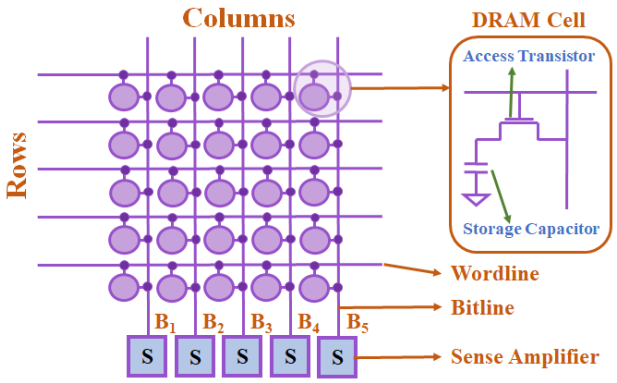


Figure 3.7: DRAM organization.[60]

3.7 DRAM PUF

The ongoing research in memory-based PUFs has shown that not only SRAM and its start-up behaviour can be utilized for a PUF, but also the Dynamic Random Access Memory (DRAM) exhibits PUF-like behaviour, when properly used [60]. DRAM consists of individual DRAM cells arranged in a two-dimensional array. The rows of DRAM are known as *wordline* and *bitline* respectively. The bitlines are connected to a row of sense-amplifiers. A DRAM cell is used to store one bit and consists of two components - a capacitor to hold the charge and an access transistor to access the capacitor. The charging state of the capacitor represents the stored value (charged capacitor is a logic 1, otherwise it is 0).

There are numerous possible approaches of how to utilize the DRAM for PUF, e.g. exploiting the leakage of DRAM cells, their start-up behaviour or violating the DRAM timings. We introduce some of them in the following subsections:

3.7.1 Retention-based DRAM PUFs

Since the DRAM cells are leaky and therefore the capacitor is discharging over time, the DRAM cell contents need to be refreshed periodically, usually every 64ms [60] to ensure the integrity of the DRAM. Failing to refresh periodically within this time interval introduces errors due to the leaky property of DRAM cells.

The demonstration of retention-based device signature was presented by Rosenblatt et al. [53, 54]. The retention signature of the DRAM is unique to each chip and can be controlled without circuit modifications. In the beginning, logic 1 is written to every cell in the DRAM. Then the DRAM is not refreshed on purpose for a predetermined time interval, causing some of the DRAM cells to change their state from 1 to 0. The resulting bitmap is the PUF response.

This approach has several drawbacks, beginning with the periodic refresh. This operation is handled by a memory controller and there is no efficient way to control this refresh time for an arbitrary small region of DRAM since the granularity for such operation is predefined by the vendors [60]. Xiong et al. [70] proposed the Run-Time DRAM

PUF that allows to refresh critical parts of the memory required by the system but leaves the PUF areas untouched. However, this solution requires a selective DRAM refresh and a safe reservation of the PUF memory region without critical data corruption (*memory ballooning*).

Another issue is the fact that the cryptographic key of sufficient length requires an adequate number of errors. This means that it might need a long time period before the memory can be read without its refreshing (seconds or even minutes [70]). Finally, the retention time is strongly dependent on the temperature [70].

3.7.2 Latency-based DRAM PUF

The Latency-based DRAM PUF benefits from violating the timing constraints of DRAM operations [27]. DRAM reads and writes consist of 3 major sequential steps: 1) activation, 2) read/write, and 3) precharge. The activation command opens a row and prepares it for accesses. The timing parameter t_{RCD} represents the amount of time that is required for the activation process to be correctly completed. After issuing the activation command, the memory controller must wait for a delay of the t_{RCD} before executing the read or write command. Violating this delay can result in an incorrect operation, and thus incorrect data to be read.

This timing violation can be exploited for PUFs [27]. Different cells in the same DRAM chip have different reliable operation latencies, mainly due to their design differences and process variations. As an example, a cell located closer to sense-amplifiers can operate correctly with a lower t_{RCD}. Also, the manufacturing process causes the individual cells to have slightly different capacitances. The cells with larger capacitors can operate reliably with a lower t_{RCD}.

In the DRAM PUF proposal of Kim et al. [27] the authors stated that decreasing the timing parameter t_{RCD} (and also other timing parameters, such as precharge time [60]) results in failures. The DRAM latency-based PUF exploits the resulting error patterns that are obtained by 1) writing known data into a fixed-size memory segment and 2) reading it back with reduced timing parameters. The resulting failures form a pattern unique to the device.

3.7.3 Start-up based DRAM PUFs

The Start-up based DRAM PUF proposed by Tehranipoor et al. [61] generates the PUF response from the start-up values of DRAM cells. The authors observed similar DRAM start-up values behaviour to SRAMs. Initially, the bitlines are charged to $\frac{V_{dd}}{2}$. However, the manufacturing process variations influence the storage capacitor that slightly deviate the bitline voltage to $\frac{V_{dd}}{2} + \delta$ or $\frac{V_{dd}}{2} - \delta$, where δ is a small voltage. The sense-amplifiers then sense the voltage difference to logic value 1 or 0. The resulting after-start-up bitmap can be then used as the PUF response.

There are several drawbacks of start-up based DRAM PUFs that need to be taken into account. Firstly, we need a power-cycle so that the DRAM can exhibit start-up behaviour.

Secondly, we need a time gap between powering off and powering on to avoid a strong correlation between the data present in the DRAM before powering off and the newly measured PUF response.

3.8 Arbiter PUF

The Arbiter PUF uses the delay difference of logic gates and their interconnects as a source of randomness. We can find its initial proposals in the works of Lee [30] and Lim [33]. The basic concept of the Arbiter PUF is a digital race of two paths on a circuit. These paths have to be mutually symmetric. Both of the paths end in an *arbiter* that decides which one of the two paths won the race, or in other words, which path had a smaller delay. Based on the result of the race, the arbiter generates one output bit for the PUF. The Arbiter is a logical circuit used to determine which path is faster. To ensure that the result is random and unpredictable, both of the paths have to be mutually symmetric, meaning that their intended delay is the same. When this condition is satisfied, the path with the smaller delay will be dependent on the random delay variations in individual gates and their interconnects that arise during the manufacturing process.

Fig 3.8 shows the basic Arbiter PUF scheme according to [30]. Both of these paths are implemented as a series of switch components. The switch component interconnects its two input signals to the output ports with different configurations depending on the control bit (b_i). For $b_i = 0$ the paths go straight through, while for $b_i = 1$ the paths cross. The switch component logic can be implemented for example as two multiplexers. At the end of both paths, there is an arbiter detecting the first rising edge that can be realised as a latch.

To obtain 2^n possible configurations of this circuit we need n control bits configuring n switch components forming a chain of switch components. The configuration of this circuit is determined by the challenge of the PUF that represents n control bits. The result of the operation of the whole circuit for one configuration is one output bit. To obtain more output bits there are two possible scenarios that can be combined together [9]. The first option is to implement more of these circuits that will all use the same challenge (configuration) when generating the PUF output. The second possibility is to send multiple challenges to only one circuit and chain the output bits to form the PUF response.

The Arbiter PUF output should be influenced solely by the random variations in delays of the individual paths. This can be achieved if both of the following conditions are met [35]:

1. Each pair of paths is designed to be perfectly symmetrical, thus any difference in delay is caused solely by the manufacturing process variations.

2. The arbiter circuit is absolutely fair, so it does not favor one of its inputs over the other.

If any of these conditions is not met, the Arbiter PUF is biased resulting in a lower uniqueness of its responses. Designing an Arbiter PUF that satisfies both conditions is

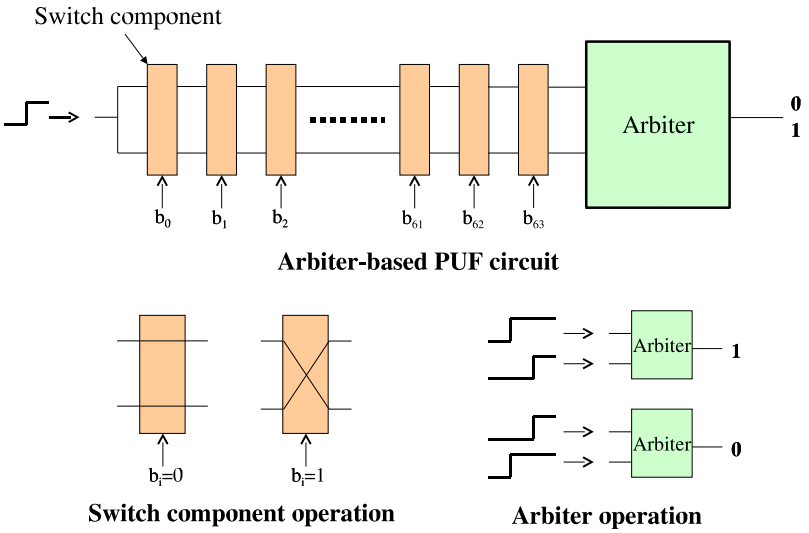


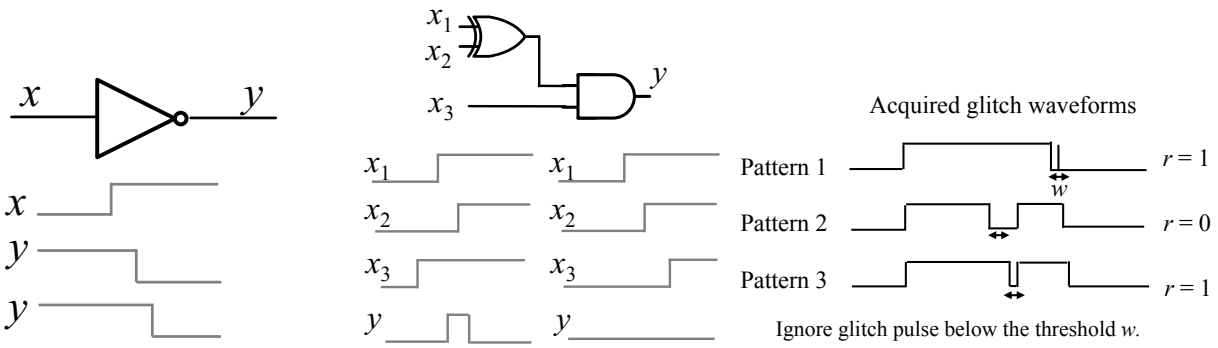
Figure 3.8: Basic Arbiter PUF scheme.[30]

not a trivial task since it requires a low-level control over the implementation. In some technologies, such as FPGAs, it is not possible [35]. Therefore, ASICs (Application-Specific Integrated Circuit) are more suitable for this type of PUF since the designer has full control over the placement and routing. However, some unbiasing techniques can be used when bias is unavoidable.

One of the major drawbacks of this PUF proposal is its susceptibility to machine learning attacks. Several constructions based on the Arbiter PUF have been proposed such as the XOR PUF [58] and the Feed-Forward PUF [30]. Both of these constructions have in common that they add some non-linearity to make machine learning attacks more difficult. XOR PUFs have been assumed to be secure against machine learning attacks when enough XORs are used. However, Becker [1] has shown that even such construction can be succesfully attacked.

3.9 Glitch PUF

Suzuki and Shimizu [59] proposed a new PUF construction. They considered a possible behavioural difference of the same logic circuits with different delays. Fig. 3.9(a) shows a difference in behaviour of a simple logic circuit. There is a time difference between output changes from an input change. However, this delay is variable since it depends on random variations of gate delays acquired while being manufactured and also largely on the temperature and voltage. A behaviour of a more general circuit that performs AND and XOR operations on multiple inputs can be seen in Fig. 3.9(b). We can see that a transient state of an output signal called a glitch occurs; this is caused by the delay



(a) Behaviour of a simple circuit. (b) Behaviour of a more general circuit. (c) Conversion of the acquired waveforms.

Figure 3.9: Substance of Glitch PUF.[59]

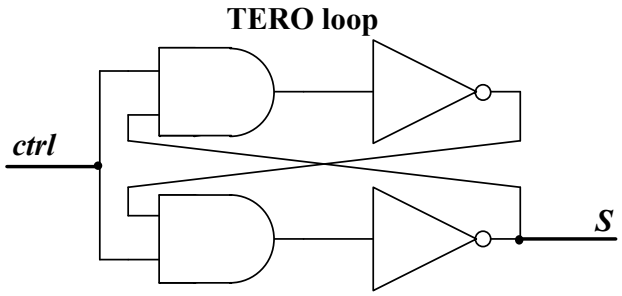


Figure 3.10: TERO loop structure.[3]

difference between input signals. The glitch shown in Fig. 3.9(b) occurs at the output of XOR due to differences of transition times of input signals x_1 and x_2 after all input signals were changed from 0 to 1. Only if signal x_3 reaches the AND gate before the glitch, the glitch propagates to the AND output. In other cases, the output remains unchanged since the glitch did not propagate to the output.

According to [59], the Glitch PUF consists of the three following steps. The first step is data input to a random logic. The next step is the acquisition of glitch waveforms at the output of the logic. The final step is the conversion of the acquired waveforms into PUF response bits. An example of such conversion is shown in Fig. 3.9(c), where a glitch with a width less than the threshold w is ignored.

3.10 TERO PUF

A new PUF structure that exploits the oscillatory metastability of cross-coupled elements was proposed by Bossuet et al. [3] and then extended by Marchand et al. [41]. The source of the entropy in this proposal is the mean number of the oscillatory circuit oscillation before the oscillations stop and the structure is stabilised. This PUF is based on transient

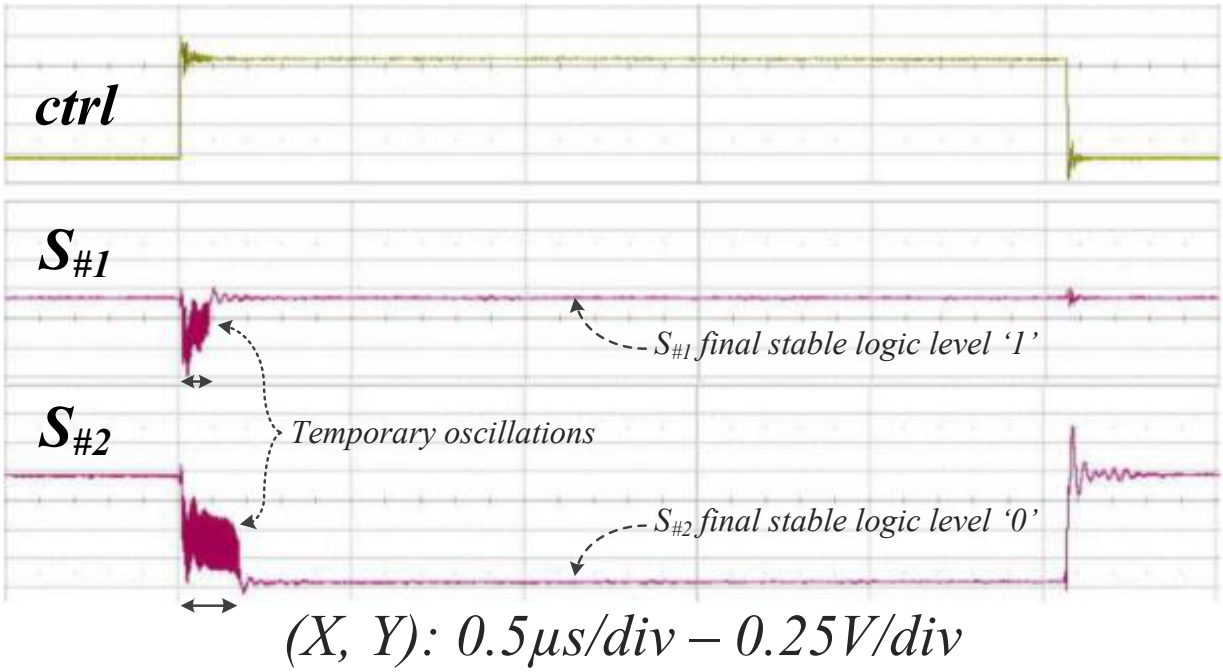


Figure 3.11: Electrical behaviour of the two TERO loops. The *ctrl* signal is an input signal to both of the TERO loops that causes temporary oscillations of the TERO loops. Signals $S_{\#1}$ and $S_{\#2}$ are outputs of the TERO loops.[3]

effect ring oscillator (TERO) cells, therefore it is called the TERO PUF. TEROs were originally proposed by Varchola and Drutarovsky [63] for TRNG designs.

The basic building element of this design is a TERO loop structure. It is composed of an SR-flip-flop implemented as two AND gates and an even number of inverters. There are usually two inverters used for the TERO loop but the loop can be extended by more inverters in order to extend the oscillations.

The oscillatory metastability in this design is achieved by connecting the S and R inputs of the SR-flip-flop to the *ctrl* signal. An example TERO loop structure is shown in Fig. 3.10. When the *ctrl* signal is set to high, the structure is brought into an unstable state and causes transitory oscillations in the loop if certain conditions are fulfilled [63]. Theoretically, if the loop was absolutely symmetrical, the oscillations would never stop. However, the TERO loop structure oscillates for a short period of time before it stops due to the asymmetry in the time delay of both halves of the loop.

A behaviour example of two TERO loop structures is shown in Fig. 3.11. It shows an input *ctrl* signal and the output signals ($S_{\#1}$, $S_{\#2}$). The *ctrl* signal for both TERO loops is forced to logical value 1. The rising edge of the *ctrl* signal causes temporary oscillations in both of the TERO loops as can be seen at the $S_{\#1}$ and $S_{\#2}$ signals. In Fig. 3.11 it is clearly visible that the number of oscillations of both TERO loops is different and when the loops are stabilised, the resulting value for both loops is also different.

In [3] Bossuet et al. measured the number of oscillations using 8-bit counters and the

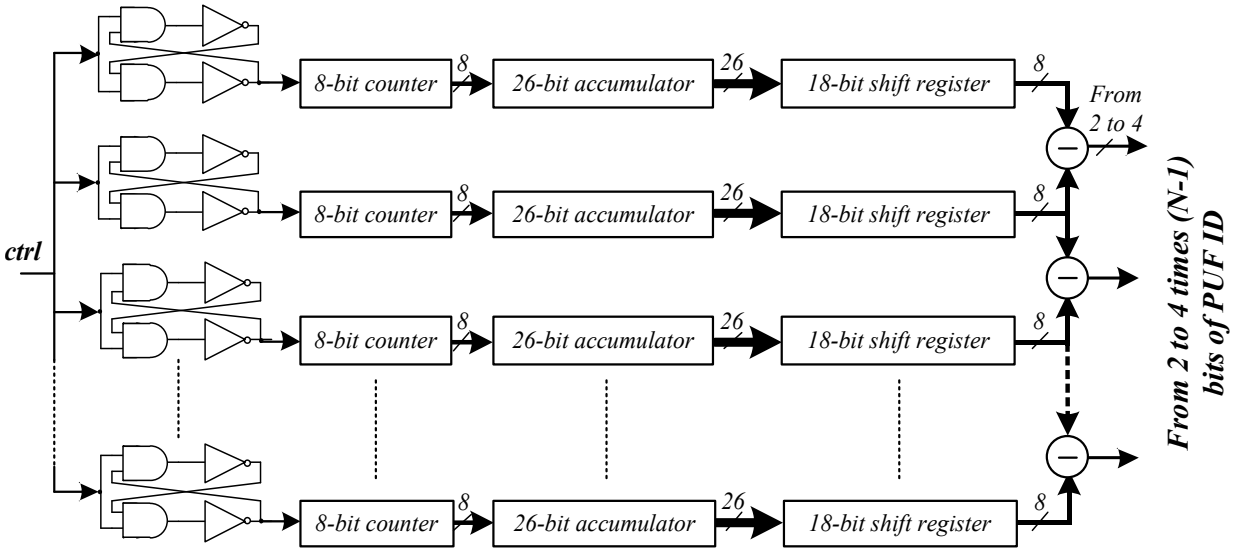


Figure 3.12: TERO PUF architecture.[3]

measurements were performed 2^{18} times for each TERO loop. Then the mean value of the number of oscillations of each TERO loop was used for further processing. Finally, the mean values were subtracted in a pair-wise manner and particular bits were selected from the resulting binary value and used for the PUF output based on their statistical properties. The described TERO PUF architecture is shown in Fig. 3.12.

Bernard et al. [2] proposes a stochastic model for long TEROs. This is required by certification bodies such as the German BSI AIS 31 [26] for TRNGs. Such model provides a guarantee on the entropy provided by a TRNG design. Delvaux [8] invalidated this mathematical model and discovered a strong link between TEROs and Bistable Rings. Moreover, Delvaux [8] pointed out that all previously published TERO implementations featuring multi-bit counters were flawed because they did not take the duty cycle of TEROs that becomes extreme during the measurements into account. This resulted into corrupted counter values.

3.11 Ring oscillator PUF

Numerous PUF constructions based on ring oscillators (RO) have been proposed to this day but it is not the goal of this paper to introduce them all. This section describes the main principle of Ring Oscillator PUFs (ROPUF) that have been proposed so far. We put emphasis on this PUF construction because our proposed PUF design is also based on ROs.

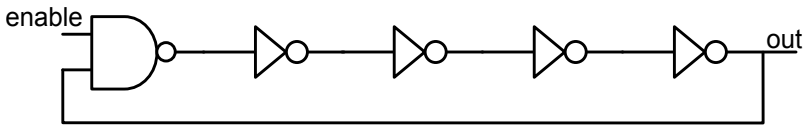


Figure 3.13: A basic ring oscillator composed of one NAND gate and four inverters forming a combinational loop.

3.11.1 Measuring a delay

Since ROPUF is a delay-based PUF it exploits random variations in delays of logic gates and their interconnects. The method that is used in ROPUFs to measure delays lies in using a delay circuit and making it a self-oscillating loop. This can be achieved by inverting the output of the delay circuit and feeding it back into the delay circuit's input [18, 19]. These oscillating circuits are usually called ROs. An example of a ring oscillator is shown in Fig. 3.13.

Random variations in delays are reflected in the measured frequencies of ROs. We can simply measure a ring oscillator's frequency by using counters and a reference clock of which we know its frequency. This frequency can be determined by the number of ring oscillator's oscillations that are recorded by a counter at a certain time specified by the reference clock. We can easily calculate the ring oscillator's frequency from the resulting value in the counter.

Frequencies of each RO obtained this way can be used by a PUF. These frequencies are used depending on the particular ROPUF proposal. In some ROPUF designs it is not necessary to know the particular frequency of each RO; we can use the resulting value in the counter.

3.11.2 Ring oscillator PUF constructions

The following list of ROPUF constructions is not complete since there is a significant amount of various proposals that are using ring oscillators. However, most of them have a similar basic principle and their differences mainly lie in their implementation.

3.11.2.1 Frequency measurement

The first type of ROPUF was proposed by Gassend et al. [18, 19]. The measured frequencies of equal ROs on different devices shows sufficient variation to act as a PUF output. However, the influence of environmental conditions on the frequency of ROs is significant and some additional technique is required to compensate these influences. Gassend et al. [18, 19] proposed a technique called *compensated measuring*. The main idea behind compensated measuring is that environmental changes will affect the frequencies of ROs approximately the same way, therefore a ratio of measured frequencies of RO pairs may be considered the eventual PUF output.

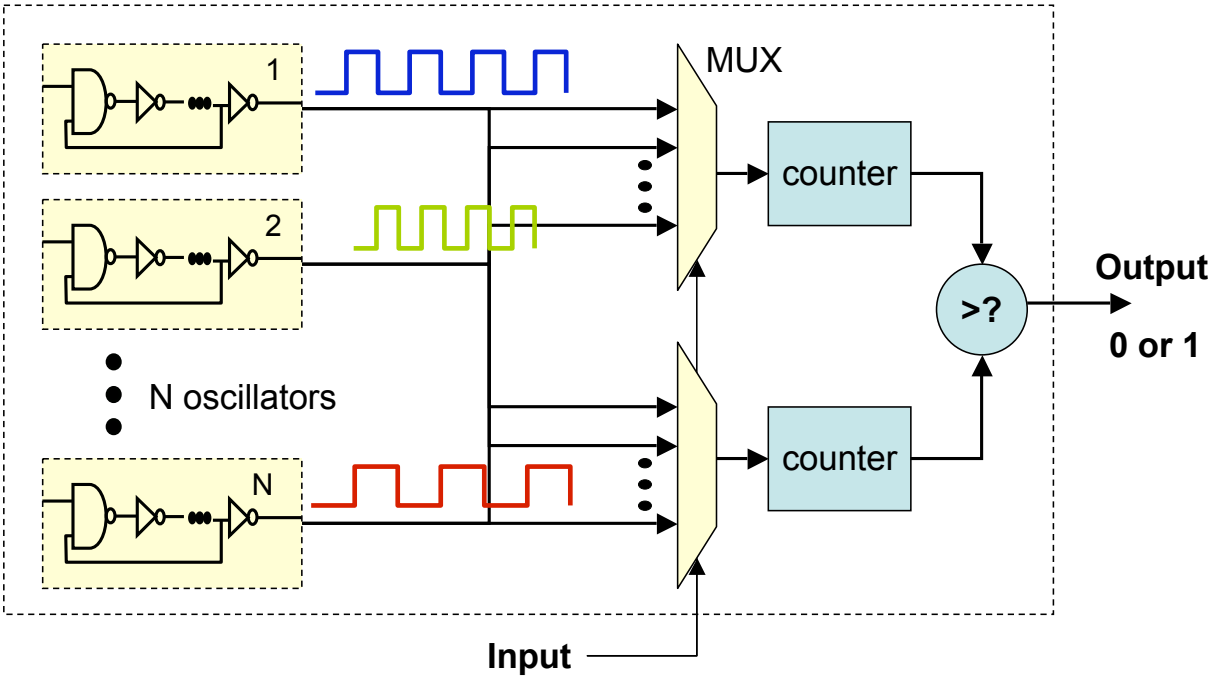


Figure 3.14: Ring Oscillator PUF design.[58]

This ROPUF construction proved to be effective in compensating the environmental changes. Nevertheless, it has some drawbacks. Since their ROs are based on the same delay circuit as the basic Arbiter PUF, their PUF is vulnerable to modelling attacks and therefore countermeasures have to be made. In addition, the result of frequency ratios in case of compensated measurement are real values and cannot be used directly as a bit string, hence they have to be processed in an appropriate way to get a proper PUF output.

3.11.2.2 Frequency comparison of ring oscillators

Another ROPUF construction proposed by Suh and Devadas [58] is shown in Fig. 3.14. Their ROPUF design consists of n symmetric ROs that are connected to two multiplexers. Each of the multiplexers selects one of the ROs according to the *input* signal and connects its output to the counter. Both counters count oscillations of the selected ROs for a fixed time interval. The resulting values of both counters are compared and one bit of the PUF output is produced based on the result of the comparison. Since one comparison produces only one bit of the PUF output, this whole process has to be repeated several times with different selections of ROs to produce the complete PUF output.

It is required that all of the ROs in this ROPUF construction are symmetric in order to assure that the comparison results are unpredictable. Due to the symmetry of ROs the differences in their frequencies are completely dependent on random delay variations due to manufacturing process variations. The frequency comparison may be considered another form of compensated measuring which eliminates the influences of environmental changes.

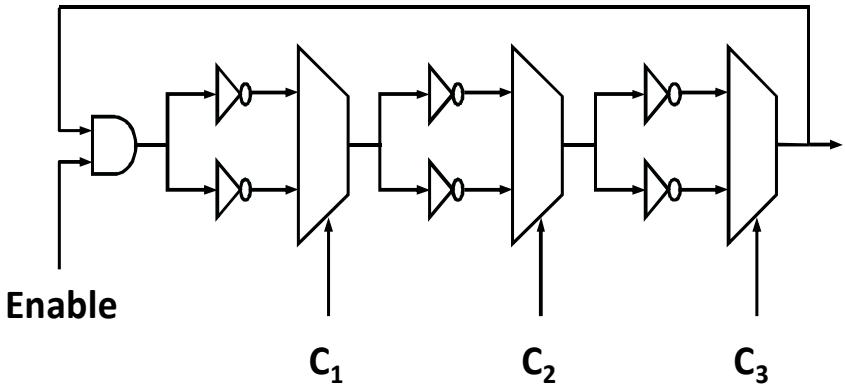


Figure 3.15: Configurable ring oscillator.[38]

Suh and Devadas [58] also proposed a technique called *1-out-of-k masking* that reduces the number of possible comparisons in order to get a more stable output. This technique is based on the selection of one RO pair with the biggest difference in its frequencies from k oscillators.

One of the drawbacks of this ROPUF design is the fact that if we want the bits in the PUF output to be independent the number of possible comparisons is limited. The maximum number of comparisons with n ROs that we are able to perform using this design is $\binom{n}{2} = \frac{n(n-1)}{2}$. However, the entropy of the design is less than $\binom{n}{2}$ [58] because bits obtained this way are correlated. For example, if RO A is faster than RO B and RO B is faster than RO C, it is clear that RO A will be faster than RO C.

To determine the maximum entropy of this design, i.e. the maximum number of independent bits generated by pair-wise comparisons, we have to consider all possible orderings of n ROs. There are $n!$ possible orderings of ROs based on their frequencies and if the orderings are equally likely the entropy of this design will be $\log_2 n!$.

An easier way to obtain independent bits in the PUF output is to use each RO only once for comparison, thus the number of output bits would be $\frac{n}{2}$. Combined with the technique 1-out-of-k masking, the number of output bits can be significantly reduced.

Another technique to increase the stability of the ROPUF output was proposed by Yin and Qu [71]. The ROs are divided into mutually exclusive groups (no RO is in two groups at the same time) and the frequency comparison is performed only between the ROs of the same group where a high stability of comparison results is guaranteed. The division of ROs into groups is performed by a proposed algorithm called LISA (Longest Increasing Subsequence-Based Grouping Algorithm). For each RO measurements are executed at various conditions and the minimum and maximum measured frequencies are stored. Using the LISA algorithm, ROs are divided into groups in such manner that the differences of the minimum and maximum frequencies between any RO pair are larger than a selected threshold value. Using this technique we can obtain a very stable output and also a longer output from the same number of ROs than using 1-out-of-k masking.

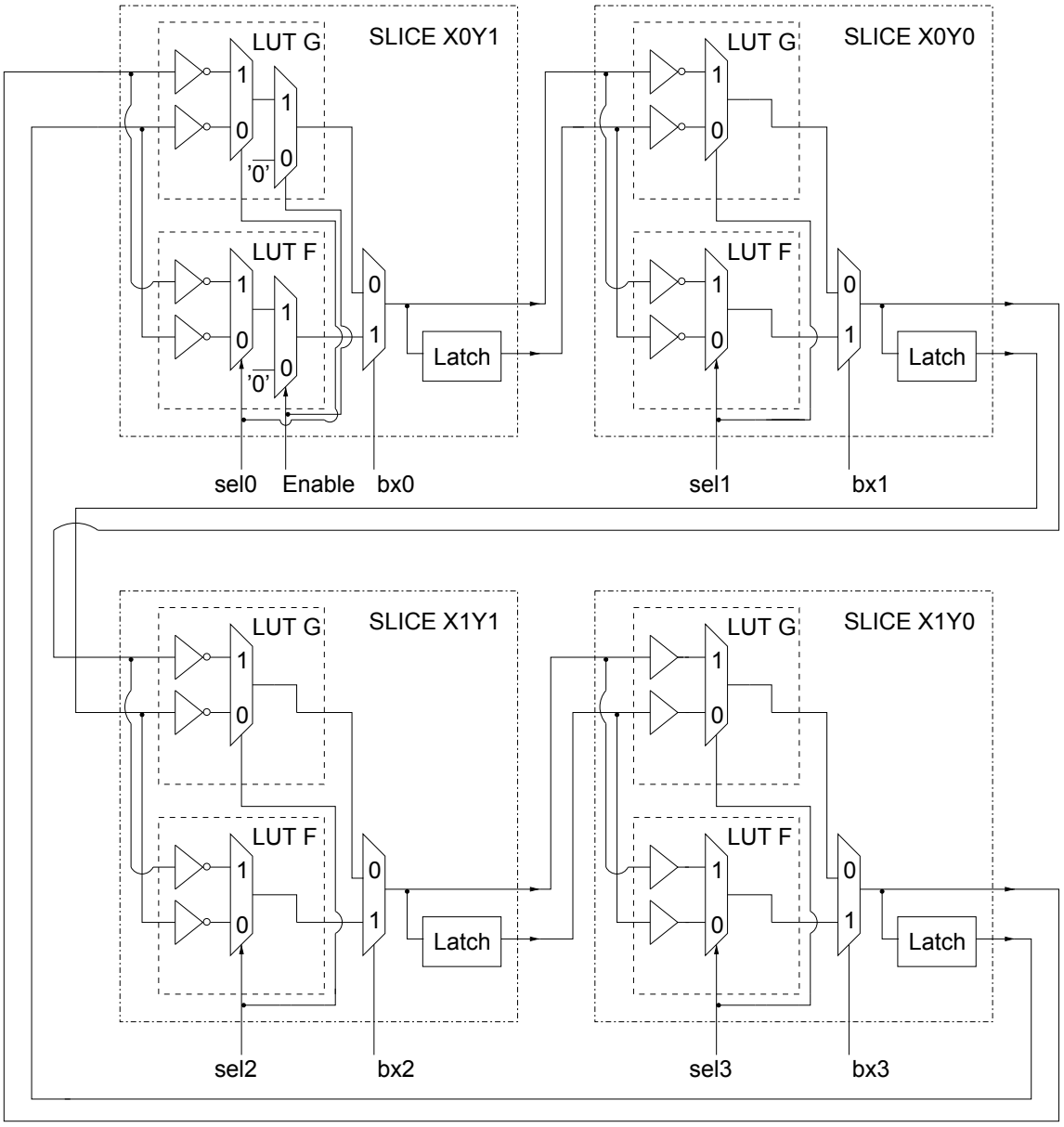


Figure 3.16: A configurable ring oscillator capable of 256 different configurations in one FPGA CLB. In this case, the CLB consists of four *slices* and each slice contains two LUTs (Look-Up Table). The *sel* and *bx* signals are control signals for multiplexers that determine the configuration of the RO.[69]

3.11.2.3 Configurable ring oscillators

Maiti and Schaumont [38] used the same design as Suh and Devadas [58] (frequency comparison) together with the technique of 1-out-of-k masking to increase the output stability.

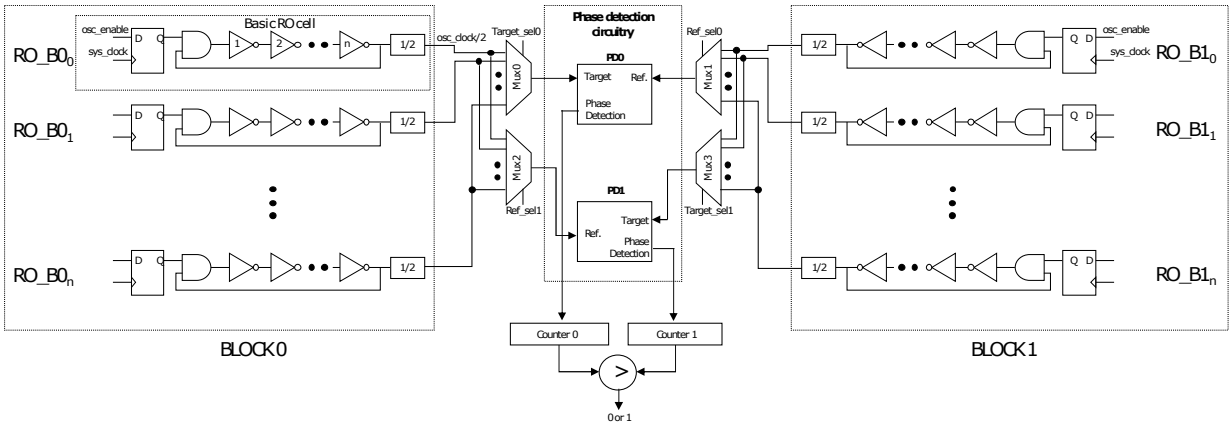


Figure 3.17: Architecture of phase detection ROPUF.[31]

However, the design is now based on configurable ROs instead of basic ROs. These are considered the most stable configuration out of k possible configurations of one RO pair, not the most stable RO pairs. The advantage of configurable ROs is that they allow for a more efficient utilization of resources.

The configurable RO proposed in [38] is shown in Fig. 3.15. The classic RO composed of five gates (one NAND and four inverters) implemented on the FPGA occupies almost the whole CLB (configurable logic block) on Xilinx Spartan-3E. The configurable RO shown in Fig. 3.15 consists of six inverters, one AND, and three multiplexers. This all fits into a single CLB occupying basically the same area as a common 5-staged RO. The configuration of the RO is set by the control bits that are used to control the multiplexers. In this case, there are three control bits, hence eight possible configurations of the RO. Since the PUF output is derived from the frequency comparisons it is necessary for the configurable ROs to be symmetrical.

Another proposal of a configurable RO on Xilinx Spartan-3E was presented in the work of Xin et al. [69]. They extended the design proposed by Maiti et al. [38]. The configurable RO still fits into a single FPGA CLB, but it is also capable of 256 various configurations. This design uses more multiplexers and also latches as shown in Fig. 3.16. The latches are used as an additional delay unit. This configurable RO is configured by eight control bits, therefore it is possible to achieve 256 possible configurations. Again, the ROs have to be mutually symmetric and it is also necessary to use ROs with the same configuration for the comparison, otherwise, the result of the comparison cannot depend on random delay variations.

3.11.2.4 Phase detection ROPUF

Lee et al. [31] proposed a phase detection ROPUF shown in Fig. 3.17. The design consists of 2 RO blocks that are comprised of n identical ROs. The output of each RO block is divided into target and reference clocks that are used as sources of two phase detection blocks. The target and reference clocks for the phase detection block are from different

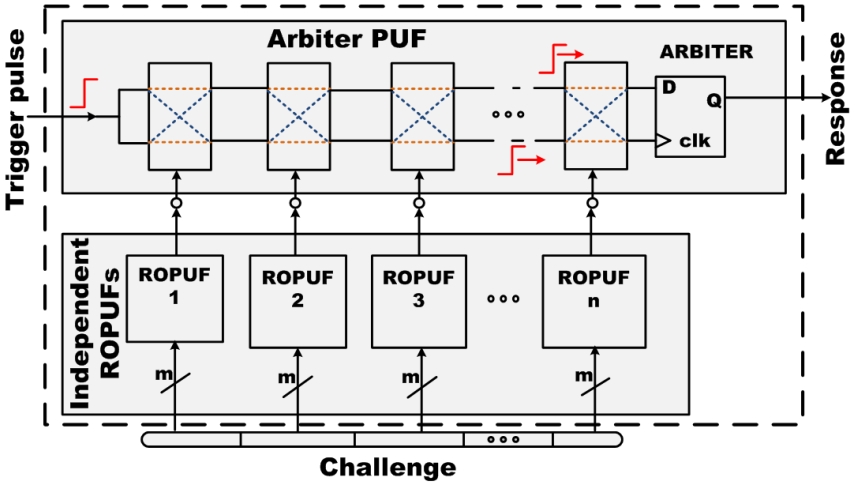


Figure 3.18: Composite ROPUF architecture.[55]

RO blocks. Two counter values are generated by the phase detection blocks that are then compared and produce 1 bit PUF output.

In order to generate 1 bit PUF output, 2 ROs from each RO block are required, thus 4 ROs are used. However, according to [31], the phase detection method increases the number of possible challenge-response pairs because different counter values may be obtained from the same RO pair by exchanging the target and reference clock. The challenge in this proposal is used to select 4 ROs that are used as target and reference blocks for each measurement. Since the target and reference clock from each RO block can be exchanged, the same 4 ROs may be used to produce 4 bits for the PUF output.

3.11.2.5 Composite ROPUF

Another ROPUF design attempting to increase the challenge-response pairs space was proposed by Sahoo et al. [55]. This proposal presents a combination of 2 PUFs, namely the Arbiter PUF and the ROPUF. The design consists of a set of ROPUF instances and one Arbiter PUF as shown in Fig. 3.18. The individual ROPUF instances produce single bit outputs that are used to configure the switch boxes of the Arbiter PUF that then generates the PUF response.

The ROPUF used in this design is the frequency comparison proposal of Suh and Devadas [58]. Therefore, the challenge here is used to select RO pairs for comparison in the individual ROPUF instances. The Composite ROPUF design is further generalized and analyzed in [56].

3.11.2.6 ROPUF with enhanced challenge-response set

Maiti et al. [39] proposed a new approach of RO utilization in order to create PUF responses. Instead of comparing the oscillation frequencies of selected RO pairs, response to a given challenge is defined by selecting a subset of the m oscillation frequencies of the

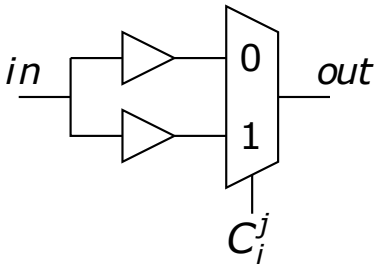


Figure 3.19: Delay element in a Loop PUF.[5]

ring oscillators. The frequency subsets are then evaluated using the Euclidean distance, resulting in a set of Euclidean distances representing a statistical distribution. The noisy PUF response is corrected using the method of *Shielding function* [34] and *helper data* [45].

Since n ROs can form a total of $2^n - n - 1$ subsets we can obtain $2^n - n - 1$ response bits compared to $\log_2 n!$. However, such response bits are not information-theoretically independent [39], the upper number limit of independent bits is still $\log_2 n!$.

One of the drawbacks of this design lies in its complexity during the processing of the measured RO frequencies. Moreover, Nguyen et al. [45] stated that the proposed PUF is vulnerable to cryptanalytic attacks. The main cause of this liability is the fact that the responses to different challenges are not independent of each other, which results in a successful guess probability higher than $\frac{1}{2}$. Furthermore, by utilizing the helper data, the adversary can predict the response with probability of success equal to 1.

Besides the design by Maiti et al. [39], there are numerous other RO based proposals that report larger space of challenge-response pairs than Suh and Devadas [58]. Such proposals are e.g. [46, 7, 72].

3.12 Loop PUF

A PUF design consisting of a single ring oscillator is based on controllable delay elements was proposed by Cherif et al. [5]. This PUF proposal does not use differential or parallel comparisons as the traditional approaches do(Arbiter PUF, ROPUF). It compares multiple elements sequentially.

The Loop PUF consists of N delay chains forming a loop that is closed by an inverter, causing this structure to oscillate as a single ring oscillator. Each delay chain comprises M controlled delay elements (see Fig. 3.19) that are connected in a sequence. The delay chains are configured by C_i control words that are M bits long. The whole Loop PUF structure is shown in Fig. 3.20, where C_i^j controls the configuration of the j-th delay element in the i-th chain. Bby measuring the frequency of the oscillator with chosen configuration, the Loop PUF controller extracts the result i.e. a cryptographic key or response of a challenge, based on N C_i control words.

The main principle of this design lies in the symmetry of N delay chains. These have to be mutually symmetric. When the PUF is challenged with C_1,...,C_N, the Loop

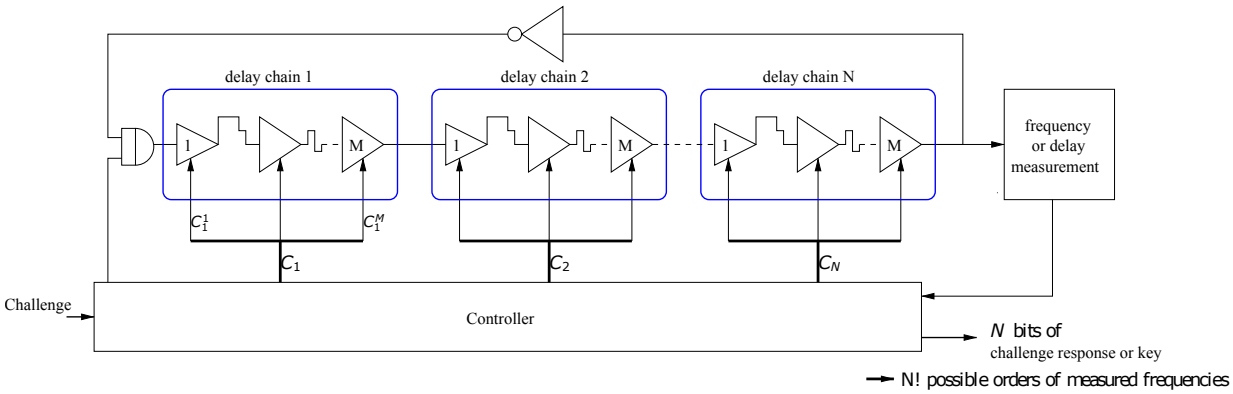


Figure 3.20: Loop PUF structure.[5]

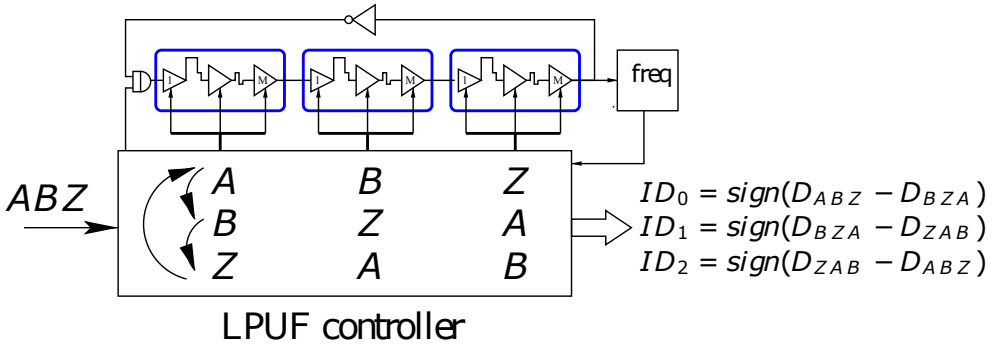


Figure 3.21: Loop PUF control example ($N = 3$). ID_i is the i-th response bit and *sign* represents the pairwise comparison of two different configurations (permutations of control words C_i).[5]

PUF controller applies different combinations of the C_i control words and measures the frequency of the oscillator. Since all N delay chains are mutually symmetric, one would expect the same frequencies for all permutations of C_i. However, due to local mismatches caused by the manufacturing process, the oscillator frequency is different for each of the C_i permutations.

To generate a PUF response, the Loop PUF controller generates the combination of control words from the original challenge in order to perform pairwise comparisons of the oscillator frequencies with different configurations. For example, the controller may rotate the control words N times to produce N response bits as shown in Fig. 3.21.

3.13 Bistable ring PUF

Another type of a delay based PUF is the Bistable Ring PUF (BR-PUF) proposed by Chen et al. [6]. This proposal exploits the fact that an inverter ring oscillator consisting of an even number of inverters has two possible stable states. As in the case of the SRAM cell

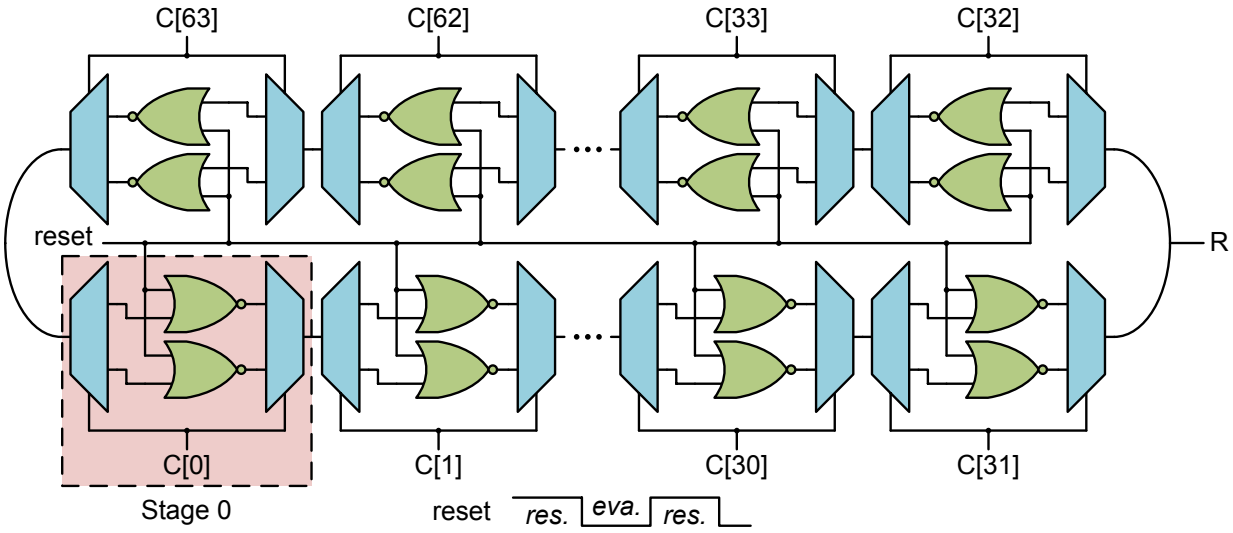


Figure 3.22: Bistable Ring PUF.[6]

that consists of two cross-coupled inverters, a ring with an even number of inverters ends up in one of the two possible states when powered up or when released from an unstable state.

The BR-PUF design proposed by Chen et al. [6] is shown in Fig. 3.22. Instead of inverters, the ring is composed of NOR gate pairs, of a multiplexor (MUX) and a demultiplexor (DEMUX). Each stage of the ring begins with DEMUX that selects the NOR gate, where the stage input signal is fed. MUX selects the output of one of the NOR gates and sends it to the output of the corresponding stage. MUX and DEMUX in one stage share a common select signal, i.e. one bit from the challenge.

The exponential number of challenge-response pairs present the advantage of this design. The challenge bits are used to configure the ring by controling the MUXes and the DEMUXes. In order to bring the bistable ring to an unstable state there is a common reset signal connected to each NOR gate in the ring. By pulling the reset signal high, the ring is brought into an unstable state (the output of each NOR gate will be “0”) and a challenge can be applied. After pulling the reset signal low, we have to wait for some time to let the bistable ring stabilize. The response bit can be read out from any node between any two stages of the ring.

It has been observed that a good 0/1 balance of the Bistable Ring PUF responses correlates with longer settling times. This is problematic since the usual evaluation method requires the bistable ring to be settled in order to generate a reliable PUF response. Hesselbarth et al. [21] proposed new evaluation methods in order to enhance the reliability of the PUF with decreased settling times.

Chapter 4

PUF Evaluation Parameters

In previous chapters we introduced the general concept of PUFs, their properties, applications, and numerous existing constructions. In order to evaluate and compare various PUF designs we need evaluation parameters that reflect the performance of the PUF designs. Therefore, this chapter is devoted to defining several evaluation metrics that will be used later in this thesis. When choosing the evaluation parameters we were mainly inspired by the work of Maiti et al. [40] where the authors compared evaluation parameters proposed by other researchers [23, 57].

First, we will introduce the notation that will be used in this thesis. Next, the evaluation metrics will be presented.

4.1 Notation

Before we start presenting individual performance metrics we should introduce the notation that will be used throughout this text. The notation shown in Table 4.1 is similar to the one proposed by Hori et al. [23].

We use the **reference response** $\boldsymbol{R\_ref_n}$ to denote the expected or most probable response of a device n. Since R_ref_n is the response that the device will most likely generate, it is often defined as the average response ($\bar{R}_n$) of all of the T measured responses that were obtained at "normal" environmental conditions (room temperature, normal supply voltage). However, some of the other related works use the first measured response as the reference response or they do not further specify how the reference response is obtained. However, what they all have in common is that the reference response is always generated at normal environmental conditions.

In this work, the R_ref_n reference response stands for the average response of all T responses measured on device n at normal environmental conditions. The R_ref_n reference (or average) response is created as follows.

The $\mathbf{R_{n,t}}$ **PUF response** is a string of bits where $r_{n,t,l} \in \{0, 1\}$ denotes the l-th bit of

N	Total number of devices
n	The index of a device ($1 \leq n \leq N$)
T	Total number of measurements performed per device
t	The index of a measurement ($1 \leq t \leq T$)
L	The length of a PUF response
l	The bit position in a PUF response ($1 \leq l \leq L$)
$R_{n,t}$	The t-th measured PUF response of device n
R_ref_n	The reference PUF response of device n
$r_{n,t,l}$	The l-th bit of the t-th measured PUF response of device n
$r\_ref_{n,l}$	The l-th bit of the reference PUF response of device n
$HW(x)$	Hamming weight of a bit string x
$HD(x, y)$	Hamming distance of two bit strings x and y

Table 4.1: Notation used in this thesis.

such string. $R_{n,t}$ is expressed as:

$$R_{n,t} \in \{0,1\}^L = r_{n,t,1} \| r_{n,t,2} \| \ldots \| r_{n,t,L}, \tag{4.1}$$

where $\|$ denotes concatenation of the operands.

Let $p_{n,l}$ be the relative frequency of bit 1 on the l-th position of all T $R_{n,t}$ responses of device n:

$$p_{n,l} = \frac{1}{T} \sum_{i=1}^{T} r_{n,i,l}. \tag{4.2}$$

Then the **mean response** $\bar{R}_n$ is defined as follows:

$$\begin{aligned} \bar{R}_n &= \bar{r}_{n,1} \| \bar{r}_{n,2} \| \ldots \| \bar{r}_{n,L}, \\ \text{where } \bar{r}_{n,l} &= \begin{cases} 1 & \text{if } p_{n,l} \geq 0.5, \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{4.3}$$

The **Hamming Weight** of a bit string x of length L is the number of non-zero values:

$$HW(x) = \sum_{i=1}^{L} x_i. \tag{4.4}$$

The **Hamming Distance** of two bit strings x and y of length L represents the number of positions where the corresponding bits are different:

$$HD(x, y) = \sum_{i=1}^{L} (x_i \oplus y_i), \tag{4.5}$$

where $\oplus$ denotes xor operation.

4.2 PUF evaluation parameters definition

We need metrics in order to evaluate the quality of the PUF outputs and the PUF statistical properties. In this section we describe the evaluation metrics that we use in this work. These evaluation metrics are:

- **Reliability**
- **Uniqueness**
- **Uniformity**
- **Bit-aliasing**
- **Randomness**

We define these metrics in the following subsections.

4.2.1 Reliability

A common parameter used to evaluate the reliability of the PUF outputs is the *Intra-Hamming distance*. This parameter represents how a PUF is efficient at reproducing its responses, therefore how it is associated with the property of reproducibility described in Chapter 2. Even though this parameter is widely used for the evaluation of PUF reliability, its definitions are slightly different in various works. In all of the definitions, the ideal value of HD_{intra} is 0%, representing errorless PUF responses.

One possible definition of the *Intra-Hamming distance* (HD_{intra}) was presented in [35]:

$$HD_{intra} = \frac{2}{NT(T-1)} \sum_{i=1}^{N} \sum_{j=1}^{T-1} \sum_{k=j+1}^{T} \frac{HD(R_{i,j}, R_{i,k})}{L} 100 \ [\%]. \tag{4.6}$$

This HD_{intra} definition performs a pairwise comparison of all of the $R_{i,j}$ responses for each device i. Therefore, its advantage is that it provides a good estimate of the distribution of the PUF response hamming distances. However, since it requires all of the pairwise comparisons to be executed, it is also computationally demanding.

Another definition of the HD_{intra} looks as follows:

$$HD_{intra} = \frac{1}{N(T-1)} \sum_{i=1}^{N} \sum_{j=1}^{T-1} \frac{HD(R\_ref_i, R_{i,j})}{L} 100 \ [\%]. \tag{4.7}$$

This definition of HD_{intra} was used e.g. in [40]. In some works the R_ref_i reference response is not described at all. In [40] R_ref_i is the response of a device i extracted at normal operating conditions. If we perform all measurements at normal operating conditions, R_ref_i could be the first response. As we will show later, such approach is susceptible to the selection of R_ref_i, because we can end up with R_ref_i, which has either a small or a large hamming distance to the other responses, causing HD_{intra} to be distorted.

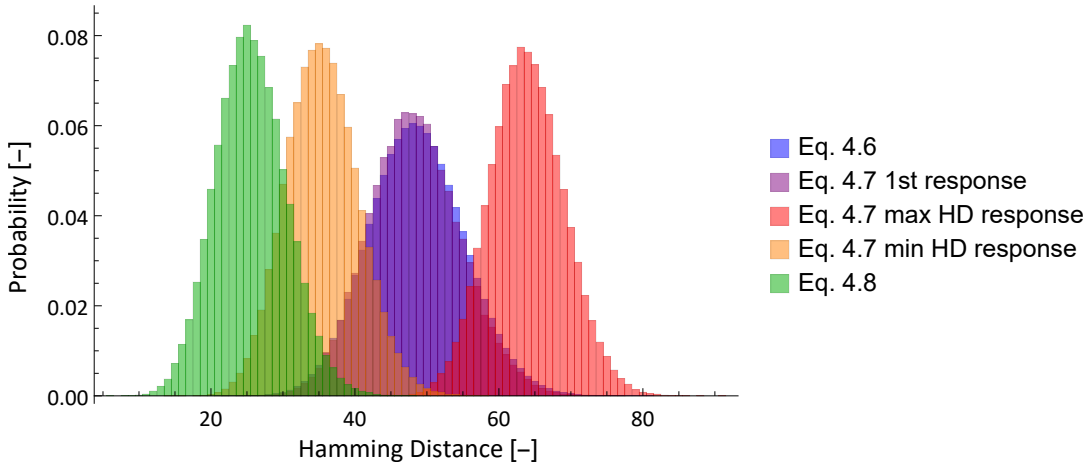


Figure 4.1: Results of individual calculated hamming distances of simulated PUF responses. The three definitions of HD_{intra} (Eq. 4.6, Eq 4.7 and Eq. 4.8) were used.

Finally, the last definition of HD_{intra} we will present here is defined as:

$$HD_{intra} = \frac{1}{NT} \sum_{i=1}^{N} \sum_{j=1}^{T} \frac{HD(R\_ref_i, R_{i,j})}{L} 100 \ [\%]. \tag{4.8}$$

This definition looks very similar to Eq. 4.7, but there is a significant difference in the selection of the R_ref_i reference response. R_ref_i is the mean response ($R\_ref_i = \bar{R}_i$) of all of the measured responses $R_{i,j}$ for device i [3]. The mean response $\bar{R}_i$ is created according to Eq. 4.3. Such choice of R_ref_i will cause the value of HD_{intra} to be smaller than in the previous two definitions (Eq. 4.6 and Eq. 4.7). However, this definition does not have the drawback of wrong R_ref_i selection as in the case of Eq. 4.7 and is not as computationally demanding as Eq. 4.6.

Comparison of reliability parameters based on simulations

To compare the behaviour of the three slightly different definitions of HD_{intra}, we performed a simulation in which we simulated generation of $T = 1000$ PUF responses from $N = 100$ devices. The PUF responses were $L = 512$ bits long and the probability of error in each bit of the responses was 5%. In this simulation, the base PUF response for each simulated device was first generated as a random bit string from uniform distribution. When created T error patterns for each device where each bit in each pattern had the probability of 5% being 1. These patterns were then xored with the base responses, forming an erroneous T PUF responses for each device.

Fig. 4.1 shows the distributions of the absolute values of hamming distances for the three presented definitions of HD_{intra}, not their fractional values (divided by $L = 512$). It does not show the value of the HD_{intra} itself (it is the mean value of these hamming distances), it shows the values of calculated hamming distances for the individual comparisons of PUF

responses. This means that for Eq. 4.6 there will be $\frac{NT(T-1)}{2}$ values compared to only NT (or $N(T-1)$) for Eq. 4.8 and Eq. 4.7.

As expected, definition of HD_{intra} from Eq. 4.8 (green histogram) has the lowest values of the hamming distances since the PUF responses are compared to their mean responses. To show the possible worst case behaviour of HD_{intra} defined in Eq. 4.7, we chose three different responses as R_ref_i reference responses. First, R_ref_i was simply the first response from device i. In the next case, we chose R_ref_i for each simulated device i as the response that had the maximum mean hamming distance to all the other responses. The last case was the same, but we used R_ref_i as the response with the minimum mean hamming distance to all the other responses.

The influence of such R_ref_i choices in Eq. 4.7 can be seen in Fig. 4.1 where these histograms are shown in purple, red, and orange colours respectively. The HD_{intra} values for these three histograms are noticeably different which confirms the susceptibility of Eq. 4.7 to the choice of R_ref_i.

Finally, we decided to use the definition of HD_{intra} from Eq. 4.8. It provides more consistent results than HD_{intra} defined in Eq. 4.7 as it does not suffer from the choice of the reference response R_ref_i. Moreover, it is not as computationally demanding as Eq. 4.6 and provides a low HD_{intra} value. It is also common practice to use mean PUF responses for identification or key generation purposes.

Varying environmental conditions

Varying environmental conditions such as supply voltage and temperature influence the reliability of the PUF outputs. To evaluate the reliability of the PUF at environmental conditions α, we use HD^{α}_{intra} defined as follows:

$$HD^{\alpha}_{intra} = \frac{1}{NT}\sum_{i=1}^{N}\sum_{j=1}^{T}\frac{HD(R\_ref_i^{\alpha_{ref}}, R^{\alpha}_{i,j})}{L}100\ [\%]. \tag{4.9}$$

The equation (4.9) is the same as equation (4.8), the only difference is that responses measured at environmental conditions α are compared to the reference output obtained from normal environmental conditions α_{ref}.

4.2.2 Uniqueness

Besides reliability, we are also interested in the uniqueness of the generated PUF responses among various devices. Uniqueness represents the ability of a PUF to distinguish different devices based on their responses.

For uniqueness evaluation we use the *Inter-Hamming distance* (HD_{inter}) as the metric. As it was in the case of the *Intra-Hamming distance*, there are several slightly different definitions of HD_{inter}. We will compare these definitions and choose one of them as our uniqueness property evaluation parameter. The ideal value of HD_{inter} is 50%.

The first definition of HD_{inter} we will present here is defined in [35] as:

$$HD_{inter} = \frac{2}{N(N-1)T} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \sum_{k=1}^{T} \frac{HD(R_{i,k}, R_{j,k})}{L} 100 \ [\%]. \tag{4.10}$$

This definition of HD_{inter} performs a comparison of the responses with the same k index from T measurements between all possible device pairs. It is not clear from the definition in [35] as to what the reason of calculating the hamming distance of the responses with the same index k is and why all possible pairs of responses between all of the devices were not used as in the case of their definition of HD_{intra} in Eq. 4.6. Such approach would, however, lead to a significant increase of the computation time since there would be N^2 pairs of responses for each device combination instead of only N pairs.

The next definition of HD_{inter} used e.g. in [3] is defined as follows:

$$HD_{inter} = \frac{2}{N(N-1)T} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \sum_{k=1}^{T} \frac{HD(\bar{R}_i, R_{j,k})}{L} 100 \ [\%]. \tag{4.11}$$

Instead of calculating the hamming distance of all response pairs this definition compares all $R_{j,k}$ responses from each device to the $R\_ref_i = \bar{R}_i$ reference responses of the other devices.

The last definition of HD_{inter} we will examine here is defined as:

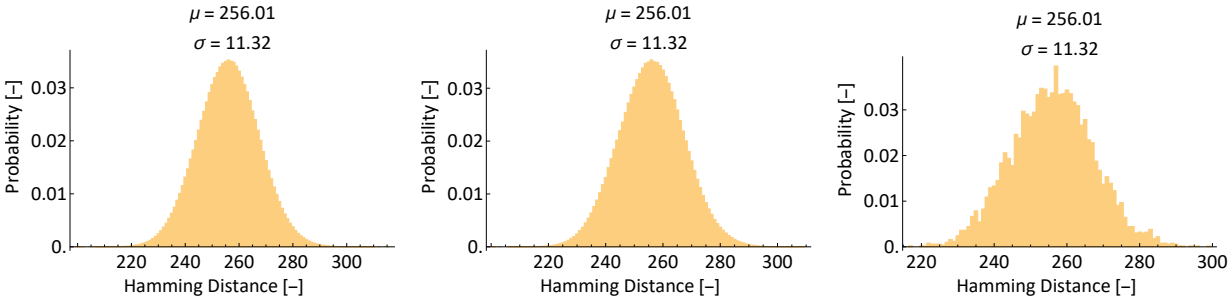
$$HD_{inter} = \frac{2}{N(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \frac{HD(R\_ref_i, R\_ref_j)}{L} 100 \ [\%]. \tag{4.12}$$

This definition can be seen in the works of Maiti et al. [40] and Hori et al. [23]. However, it slightly differs in each work. In the first place, Maiti et al. use the definition shown in Eq 4.12, however, it is not specified what the R_ref_i reference responses are. Hori et al. use the average response as the reference response ($R\_ref_i = \bar{R}_i$).

Secondly, there is a difference in the factor, by which the result is normalized. In case of the definition by Hori et al, the result is normalized by a factor that is approximately two times smaller than the one used by Maiti et al. This causes the ideal value of HD_{inter} to be 100%. In this work, we will use the normalization factor used by Maiti et al. [40] and we will use the average response as the reference response ($R\_ref_i = \bar{R}_i$) as defined by Hori et al. [23].

Comparison of uniqueness parameters based on simulations

As with the reliability parameter we performed a simulation to compare the behaviour of the three HD_{inter} definitions. The parameters for the simulation are the same as for reliability, meaning that we generated $T = 1000$ PUF responses from $N = 100$ devices, each response is $L = 512$ bits long and the probability of an error in each bit response is 5%. Again, the PUF responses were generated as random bit strings from uniform distribution



(a) Distribution of hamming distances for Eq. 4.10.

(b) Distribution of hamming distances for Eq. 4.11.

(c) Distribution of hamming distances for Eq. 4.12.

Figure 4.2: The histograms of individual values of calculated hamming distances of simulated PUF responses for the three different definitions of HD_{inter}.

and xored with randomly generated error patterns using the probability of 5% for each bit to be 1.

The results of the simulation are shown in Fig. 4.2. The histograms in Fig. 4.2 show the distribution of the absolute hamming distance values for the three HD_{inter} definitions. Instead of calculating only the mean values which is the definition of HD_{inter} we show the whole distributions of the individual hamming distances that were calculated for the response pairs used with respect to the individual definitions of HD_{inter} and without normalization by the length of the responses ($L = 512$). This means that for the definition of HD_{inter} in Eq. 4.12 we show the hamming distance histogram (Fig. 4.2(c)) of all of the possible pairs of reference (average) responses ($HD(R\_ref_i, R\_ref_j)$). In case of Eq. 4.11, it would be the individual hamming distances of all $R_{j,k}$ responses from device j to the average response $\bar{R}_i$ of device i ($HD(\bar{R}_i, R_{j,k})$). The same procedure applies to Eq. 4.10 shown in Fig. 4.2(a).

All three histograms presented in Fig. 4.2 are very similar to each other, even the differences in their mean values and standard deviations that are shown above the histograms are negligible (there are actually no differences for this particular precision). Only the histograms in Fig. 4.2(a) and Fig. 4.2(b) are smoother since they are made of significantly more hamming distance values ($\frac{N(N-1)T}{2}$ compared to $\frac{N(N-1)}{2}$).

Since the definition of HD_{inter} in Eq. 4.12 provides the same results as the other two definitions (Eq. 4.10 and Eq. 4.11) with less computational complexity, we will use Eq. 4.12 as the uniqueness parameter.

4.2.3 Uniformity

Another evaluation metric suggested by Maiti et al. [40] is uniformity. This parameter estimates how uniform the proportion of 0s and 1s in the PUF responses is. For random

PUF responses, the ideal proportion is 50%. The average uniformity is defined as [40]:

$$(\text{Uniformity}) = \frac{1}{N}\sum_{i=1}^{N} HW(R\_ref_i)100\ [\%]. \tag{4.13}$$

Hori et al. [23] use a similar definition of uniformity, but in their work they call it randomness. The uniformity defined according to Hori et al. [23] would be as follows:

$$(\text{Uniformity}) = \frac{1}{NT}\sum_{i=1}^{N}\sum_{j=1}^{T} HW(R_{i,j})100\ [\%]. \tag{4.14}$$

The uniformity defined by Hori et al. [23] only differs from the one defined by Maiti et al. [40] by the additional sum over T measurements. Also, in the definition from Eq. 4.13 by Maiti et al. [40], only R_ref_i reference responses are used for this metric evaluation since we are interested in the proportion of 0s and 1s, not in the stability of the outputs that way already covered by the HD_{intra} reliability parameter (Eq. 4.8).

The uniformity parameter is used as one of the indicators of randomness of PUF outputs. However, it is obvious that such metric is not sufficient to evaluate the unpredictability of the PUF outputs. Moreover, the proportion of 0s and 1s is usually covered in the base of various statistical randomness tests, e.g. *Frequency test* in NIST STS [52].

4.2.4 Bit-aliasing

The uniformity metric reflects the proportion of 0s and 1s in PUF responses. However, we are also interested in the proportion of 0s and 1s in individual positions of the PUF responses so that responses from different devices are not identical [40]. This evaluation metric is called bit-aliasing and its average value is defined as [40]:

$$(\text{Bit-aliasing}) = \frac{1}{LN}\sum_{i=1}^{L}\sum_{j=1}^{N} r\_ref_{j,i}\ 100\ [\%]. \tag{4.15}$$

As in case of uniformity, the ideal value is 50% for random PUF responses. Even though this metric is focused on detecting the bias of response bits across the chips, its mean value as presented in Eq. 4.15 is not suitable for such task. Responses from n chips would make an example if the first eight response bits would be a constant bit string consisting of four 0s and 1s and the rest of the responses would be random:

$$\begin{aligned} R\_ref_1 &= 10101010|...|11001011 \\ R\_ref_2 &= 10101010|...|01101100 \\ R\_ref_3 &= 10101010|...|00010111 \end{aligned} \tag{4.16}$$

The bit-aliasing parameter as defined in Eq. 4.15 does not detect such pattern since there is the same number of 0s and 1s in the pattern. Therefore, the mean value is still 50%.

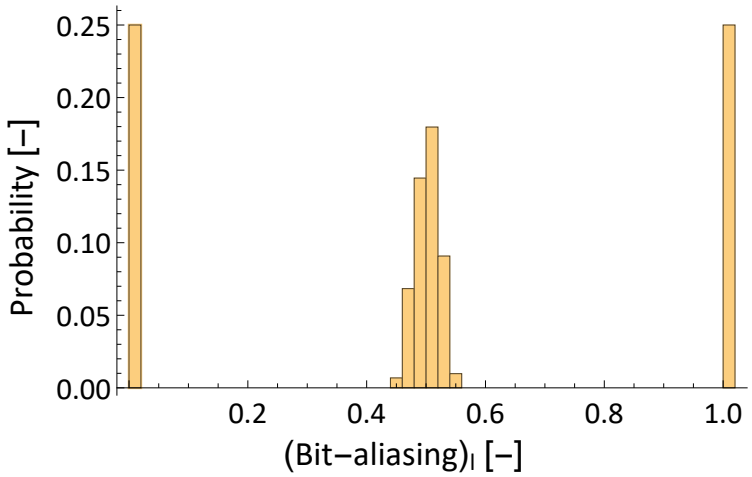


Figure 4.3: The result of bit-aliasing evaluation for individual bit positions from simulated PUF responses where the first half of all responses was a constant bit string containing the same number of 0s and 1s.

However, when we evaluate the bit-aliasing for individual positions of the PUF responses and look at the resulting distribution of such values we can clearly see that there are some positions with an obvious bias to either 0 or 1. The bit-aliasing parameter for individual positions would be defined as [40]:

$$(\text{Bit-aliasing})_l = \frac{1}{N}\sum_{i=1}^{N} r\_ref_{i,l}\ 100\ [\%], \tag{4.17}$$

where $(\text{Bit-aliasing})_l$ is the value of the bit-aliasing of the l-th bit position across N PUF responses.

Evaluation of bit-aliasing parameter based on simulations

To validate our assumption about the bit-aliasing parameter behaviour we performed a simulation in which we generated $T = 1000$ PUF responses for $N = 500$ devices. Each response was $L = 1024$ bits long. All responses have the same pattern of bits in their first half, resulting in 512 constant bits across all of the N devices. This pattern contains the same number of 0s and 1s. The second half of the PUF responses was generated as a random bit string from uniform distribution. The resulting PUF responses were xored with randomly generated error patterns using the probability of 5% for each bit to be 1.

The mean value of (Bit-aliasing) was 50.06%, being very close to the ideal 50% as expected. However, when the distribution of the individual bit positions evaluated by Eq. 4.17 is examined, there is a visible bias pattern for some positions. The distribution of the bit-aliasing values can be seen in Fig. 4.3. The two slopes on the edges of the distribution indicate a strong bias of some positions to both 0 and 1.

This specific case would also cause the HD_{inter} uniqueness parameter to be of a lower value. In this simulation, $HD_{inter} = 25.00$ indicating poor uniqueness of PUF responses. Therefore, HD_{inter} itself is sufficient to detect large patterns in PUF responses across devices. However, to discover the bias on the level of individual bits, bit-aliasing as defined in Eq. 4.17 by Maiti et al. [40] would be required.

4.2.5 Randomness

Since one of the requirements of PUFs is that the PUF outputs should be unpredictable, we need to evaluate the randomness of the PUF outputs. We already presented the definitions for uniqueness, uniformity and bit-aliasing, however, neither of them is capable of determining whether the PUF outputs are random.

It is required that the PUF responses are random (in the context of various challenges and devices, not random for one particular challenge and device) so that a potential adversary will not be able to predict the PUF output based on the knowledge of some previously learned PUF outputs from other devices or challenges.

Evaluating true randomness is not a trivial task. In the case of PUFs, we would need to evaluate the randomness of the manufacturing process itself. The manufacturing process determines the physical properties of the manufactured chips, thus it also determines the properties of the PUF implemented on these chips. Therefore, the manufacturing process is the source of randomness for PUFs.

Such evaluation of the manufacturing process is not always possible and it is also a very complex task. Therefore, we can instead use statistical tests to test the randomness [32], e.g. NIST STS [52] that is used for the evaluation of random number generators. This way, we could look at the PUF outputs as random sequences generated by a random number generator, which would in this case be the manufacturing process.

In this work we will use the statistical tests in NIST SP 800-22 [52] to evaluate randomness. The version of the NIST software we will be using is STS 2.1.2. Most of the tests in this test battery require long input bit sequences. However, in the case of PUF, we are limited by the fact that for very long bit strings we would need a large population of devices. Nevertheless, some tests can work with sequences that are at least 100 bits long. We provide a brief description of these tests below:

- **Frequency Test** tests the entire input sequence on the proportion of 0s and 1s.
- **Frequency Test within a Block** tests the proportion of 0s and 1s within M-bit blocks. The block size M should be selected such that $M \geq 20, M > 0.01n, N < 100$, where $n \geq MN$ [52] is the length of the tested sequence.
- **Cumulative Sums Test** focuses on the maximal excursion from zero of the random walk defined by the cumulative sum of adjusted digits (-1, +1) in the sequence. It determined whether the cumulative sum of the partial sequences occurring in the tested sequence behaves as a cumulative sum for random sequences. For a random sequence, the excursions of the random walk should be near zero.

- **Runs Test** observes the total number of runs in the sequence where a run is an uninterrupted sequence of identical bits. The purpose of the runs test is to determine whether the number of various lengths of runs 0 and 1 is as expected for a random sequence.

- **Test for the Longest Run of Ones in a Block** determines whether the length of the longest run of ones within M-bit blocks of the tested sequence is consistent with the length of the longest run of ones that is expected in a random sequence. The value of M is chosen by the software itself according to the length of the input sequence.

- **Approximate Entropy Test** observes the frequency of all possible overlapping m-bit patterns across the entire sequence. The purpose of this test is to compare the frequency of overlapping blocks of two consecutive/adjacent lengths (m and $m + 1$) against the expected result of a random sequence. The recommendation for choosing the value of m is $m < \lfloor log_2 n \rfloor - 5$, where n is the length of the input sequence.

The tests from the NIST STS are based on hypothesis testing. The result of each test is a p-value that represents the probability of a perfect random number generator producing a less random sequence than the sequence being tested [52]. Based on the calculated p-value and a chosen *significance level* α, the null hypothesis is either accepted or rejected. In this case, the null hypothesis is that the tested sequence is random [52]. To accept the null hypothesis, the p-value needs to be bigger then the selected significance level α.

Two types of errors can be committed during hypothesis testing [42] - Type I and Type II. The Type I error means that the true hypothesis is rejected even though the sequence being tested was produced by a random number generator. The probability of Type I error is equal to the significance level α and this value is chosen by the tester. The Type II error is denoted by β and represents the probability of false hypothesis acceptance (a defective random number generator). Both probabilities α and β are related to each other. If α is small, then β is high and vice versa [42].

The NIST STS receives multiple sequences at its input. We need to provide the length of these sequences and their number. The NIST STS then test each sequence individually using the tests described above. Each of these tests provides a p-value that is compared to the significance level α that is equal to 0.01 by default. The result of each test is the pass rate of the input sequence for that particular test.

Furthermore, besides the pass rate of each test, the NIST STS also provides a p-value for each test. This value represents the result of the uniformity test of p-values for that particular test. The NIST STS exploits the important property of p-values for arbitrary statistical tests that satisfy the null hypothesis, the p-values are uniformly distributed on the interval $[0, 1)$ [44]. Therefore, after calculating the p-values from one test for all of the input sequences, χ^2 test is used to assess the uniformity of p-values. The result is another p-value that represents the result of the uniformity test. In order to get meaningful results for the uniformity test, at least 55 sequences should be provided [52].

P-value	Pass rate	Test
0.350485	9/10	Frequency
0.122325	10/10	BlockFrequency
0.350485	9/10	CumulativeSums
0.213309	10/10	Runs
0.911413	10/10	LongestRun
0.739918	9/10	ApproximateEntropy

Table 4.2: An example output of the NIST STS for 10 input sequences. Only a subset of the tests contained in the test battery were used. In this case, the tested generator would be considered random.

According to specifications, the recommended significance level α for the uniformity test is 0.0001. This means that the p-values are considered non-uniform if a p-value resulting from the uniformity test is smaller than 0.0001. As discussed in [42], a very small value of the significance level α recommended by NIST implies a large probability of the Type II error (acceptance of a bad random number generator). The authors in [42] suggest that larger values of significance level α for uniformity testing should be used, e.g. 0.001 or 0.01.

Table 4.2 presents an example output of the NIST STS for 10 input sequences, each 300 bits long. The significance level α was set to 0.01. The middle column shows the pass rate of individual tests. The left column presents the p-values that represent the results of the uniformity tests performed for each individual test. In this example, the tested random number generator would be considered random since the pass rates for all of the tests are larger than 8/10 (the NIST STS provided this threshold) and the p-values from the uniformity tests are also large enough.

In summary, we need to provide the input sequences and the significance level α that will be used to evaluate each test on each sequence by the NIST STS. The output of this test battery is:

1. **Pass rate** for each of the tests that were used. All of the selected tests from the battery are applied to the provided sequences individually. A p-value is calculated for each test on each sequence and the test is evaluated using the significance level α. The calculated p-value must be larger than the significance level α in order for the sequence to be considered random. After evaluating all of the sequences, the pass rate is determined. The NIST STS also states what value of the pass rate is acceptable.

2. **P-value** representing the result of the uniformity test of p-values for each of the applied tests. Again, the p-value representing the uniformity has to be larger than the chosen significance level α (may be different from the significance level that is used for the evaluation of individual tests).

4.3 The final set of evaluation parameters

In Section 4.2 we presented a number of evaluation metrics and their possible variants which we compared with each other and we also discussed their properties. In this section we provide a final overview of the evaluation parameters we will use throughout this work.

The PUF evaluation metrics that we chose to use are:

- **Reliability** - Intra-Hamming distance
- **Uniqueness** - Inter-Hamming distance
- **Randomness**

We excluded the uniformity and bit-aliasing parameters as they are mainly used to evaluate randomness which can be tested more thoroughly using statistical tests for random number generators, e.g. NIST STS [52].

Intra-Hamming distance

The *Intra-Hamming distance* (HD_{intra}) evaluation parameter is used to evaluate the reliability property of the PUF outputs. As discussed in Section 4.2.1 we chose the definition of HD_{intra} from Eq. 4.8. We provide the definition here again for completeness:

$$HD_{intra} = \frac{1}{NT} \sum_{i=1}^{N} \sum_{j=1}^{T} \frac{HD(R\_ref_i, R_{i,j})}{L} 100 \ [\%]. \tag{4.18}$$

Inter-Hamming distance

As a uniqueness parameter, we chose to use the *Inter-Hamming distance* (HD_{inter}) evaluation parameter defined in Eq. 4.12. More details on why this parameter was selected are provided in Section 4.2.2. The HD_{inter} used in this work is defined as:

$$HD_{inter} = \frac{2}{N(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \frac{HD(R\_ref_i, R\_ref_j)}{L} 100 \ [\%]. \tag{4.19}$$

Randomness

As discussed in Section 4.2.5, to evaluate the randomness of the PUF responses we will use a set of statistical tests from NIST SP 800-22 [52]. The version of the software that we will use in this work is STS 2.1.2. Neither uniformity nor bit-aliasing will be used.

The sequences that will be used as an input for the NIST STS are created by the concatenation of reference (average) responses as follows:

$$R\_ref_1 \| R\_ref_2 \| \ldots \| R\_ref_n,$$

where $\|$ denotes the concatenation of the operands.

The resulting concatenation of reference responses is then split into either 10 or 60 sequences of the same length. The reason for this small number of sequences is the limited number of devices that were available to us for our experiments. Unfortunately we did not have enough data to create enough sequences with the desired length. The concatenated responses were split into 10 or 60 sequences depending on their length. When multiple bits were selected from each counter value, the final concatenation of the PUF responses was long enough to provide those 55 sequences that are required by NIST as the minimum for testing the uniformity of p-values. These sequences are at least 100 bits long. When there were not enough bits to provide the desired number of sequences of length 100 bits and more, only 10 sequences were created so that these individual sequences were long enough for the NIST tests. Moreover, the individual sequences do not exceed the length of 100 000 bits, therefore only a subset of the statistical tests that are available in the NIST STS can be used. The following subset of the statistical tests suitable for sequences that are at least 100 bits long will be used in our work:

- Frequency Test
- Frequency Test within a Block
- Cumulative Sums Test
- Runs Test
- Test for the Longest Run of Ones in a Block
- Approximate Entropy Test

The significance level α used for the evaluation of the individual tests is set to 0.01 (default value). According to the NIST STS, the successful pass rate for the number of sequences that we provided has to be at least 8/10 for 10 input sequences or 57/60 for 60 input sequences. We chose the significance level α for the p-values uniformity test to be 0.001 (from the recommendation in [42]). However, the result of the uniformity test has to be evaluated cautiously when only 10 input sequences are examined since the recommended number of input sequences is 55.

Some tests (Frequency Test within a Block, Approximate Entropy Test) require an additional parameter m specifying the block length that is used in the test. Such parameters were already described in the test descriptions above and are chosen in compliance with the NIST STS specification [52].

Chapter 5

The Proposed Ring Oscillator Based PUF

This chapter presents our proposed PUF construction. It is based on ROs and according to the classifications described in Chapter 3, this PUF construction may be classified as a delay-based intrinsic PUF since it exploits the random variations in delays of logic gates and their interconnects that affect the RO frequency.

The motivation that led to the proposal of our PUF design was to design a PUF that would be easy to implement, area efficient, and also suitable for FPGAs. There are already numerous PUF constructions proposed for FPGAs, however, some of these are more suitable for FPGAs than others. A popular PUF construction is the SRAM PUF, but a lot of modern FPGAs initialise their memory content after power-up, hence all of the randomness necessary for the PUF is lost.

The main representative of delay-based PUFs is the Arbiter PUF. Its advantage is its simplicity, low power consumption, and its implementation is inexpensive and suitable for resource-constrained platforms such as RFIDs. However, the Arbiter PUF is not well suited for FPGAs since it depends on the symmetry of the two paths of which the delay is measured. In the case of FPGAs it is impossible to achieve an absolute symmetry of both paths; we can only try to design the two paths so that they are as symmetric as possible. Therefore, Arbiter PUFs are usually not implemented on FPGAs.

A more suitable PUF design for FPGAs is the Ring Oscillator PUF (ROPUF) which has the same source of randomness as the Arbiter PUF. Instead of having two symmetric paths of which we compare their delay, the delay is measured by ROs whose frequencies are affected by random variations in the delays. A classical approach [58] of RO usage for a PUF is to compare frequencies of selected RO pairs. However, this approach requires the ROs to be mutually symmetric in order to generate unpredictable result of the comparison. Implementing symmetrical ROs is not a difficult task in comparison to the Arbiter PUF but it is still an additional overhead to designing of the PUF.

A considerable issue of the classical approach is the appropriate selection of RO pairs so that the bits in the PUF response are unpredictable and not correlated. The requirement

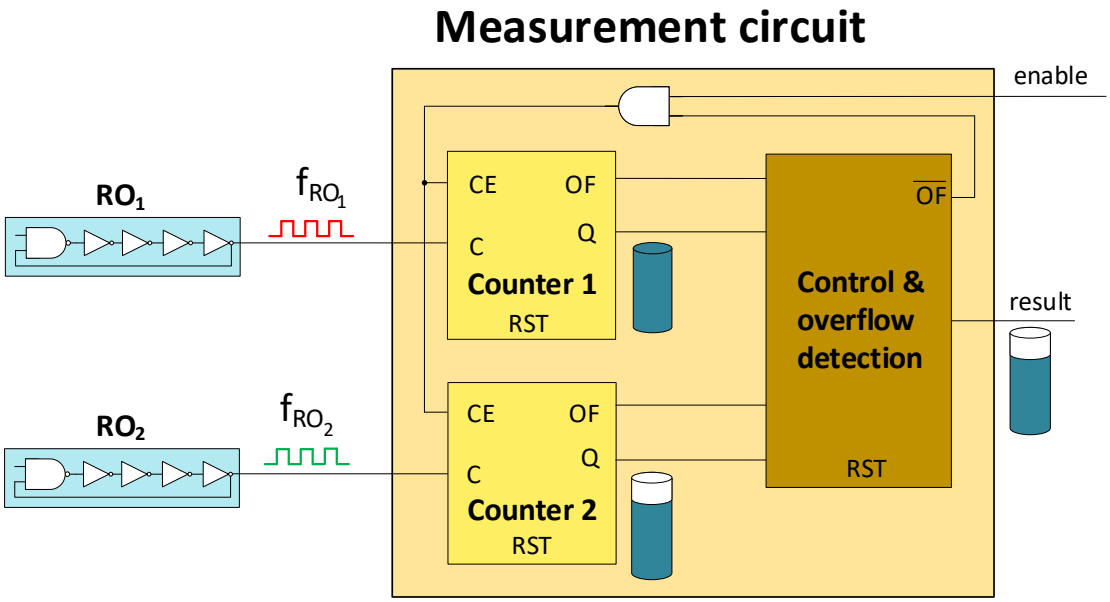


Figure 5.1: The method of measuring the number of ring oscillator cycles in the proposed ROPUF design.

for the most efficient usage of ROs also leads to more complicated designs. Therefore, each RO is usually only used for one comparison, which simplifies the design. However, a lot of potential RO pairs are not used. Nevertheless, there are some techniques presented in Section 3.11 that improve the efficiency of the ROPUF on FPGAs.

In our PUF proposal we were inspired by PUFs based on ROs. Our goal was to propose a PUF that would be easy to implement and effective. We use a different technique to generate the PUF output than frequency comparison, which is used in the classical approach and requires the ROs to be symmetrical [58]. The PUF output is also obtained based on the selected RO pairs but the problem of selecting particular RO pairs is no longer present and on top of that, more bits for the PUF output will be gained from each pair. This makes it possible to produce a longer PUF output using fewer ROs.

The following sections detail our PUF proposal published in [A.1, A.2, A.3, A.4, A.5, A.6]. The first Section 5.1 describes our proposed PUF design [A.3, A.4]. The next Section 5.2 explains the behaviour of our proposed PUF design and discusses its properties [A.5, A.6]. Section 5.3 explains how to also use the proposed design as a TRNG [A.1, A.2]. Section 5.4 provides a description of other measurement methods that can be used in the proposed ROPUF. Finally, we present an overview of the whole design for both PUF and TRNG in the last Section 5.5.

5.1 The ring oscillator based PUF proposal

The basic building element of the proposed ROPUF design is a five stage RO (one NAND, four inverters). Instead of measuring the frequency of each RO using a reference clock we

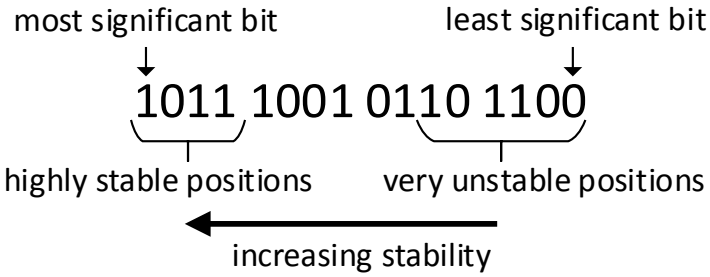


Figure 5.2: The example behaviour of positions' stability in a 16-bit value.

select one RO pair and count its oscillations simultaneously using two counters. As soon as one of these counters overflows, the measurement is stopped. The resulting value in the counter that did not overflow is used for further processing. This approach is shown in Fig. 5.1 [A.3, A.4].

The proposed method implies that if we knew the exact frequencies of the ROs during the measurement, we could determine the resulting counter value (in case of 16-bit counters) that is later processed as follows:

$$counter\ value = \frac{f_2}{f_1} 2^l, \tag{5.1}$$

where f_1 is the frequency of the faster RO, f_2 is the frequency of the slower RO and l is the bit size of the counter.

When implementing the logic for overflow detection of one of the counters and stopping the other one, the routes between them may have different delays. In the meantime, before the counter is stopped, it can perform some additional steps. Since these two routes are the same for all RO pairs and for all FPGAs, it will only increase the resulting counter value by some constant offset. It will be shown later in Chapter 7 that this offset is either 0 or 1.

The counter values obtained this way are used for PUFs without modifications. Since these values are represented in binary code, we can use the appropriately selected part of each binary number for a PUF output. It can be assumed that if we make multiple measurements of one RO pair, bits that are close to the least significant bit (LSB) will vary a lot, for example due to environmental changes. Bits close to the most significant bit (MSB) will be stable and environmental changes will have almost no influence on them. The closer we get to the MSB, the more stable these bits will be. The example of the described behaviour of measured values is shown on a 16-bit value in Fig. 5.2. A concept of choosing suitable bits for a PUF from the counter values was also presented in the work by Bossuet et al.[3].

In the following subsections we will describe the parameters and methodology of suitable bits selection for our proposed ROPUF.

5.1.1 Notation

First, we will introduce the notation we will be using to describe the parameters and methodology of suitable bits selection for our PUF proposal. The notation is shown in Table 5.1, which also includes some of the notation used in Table 4.1 in Chapter 4.

N	Total number of devices
n	The index of a device ($1 \leq n \leq N$)
T	Total number of measurements performed per device
t	The index of a measurement ($1 \leq t \leq T$)
L	The length of a PUF response
l	The bit position in a PUF response ($1 \leq l \leq L$)
M	Total number of pairs of ROs
m	The index of a pair of ROs ($1 \leq m \leq M$)
$R_{n,t}$	The t-th measured PUF response of device n
R_ref_n	The reference PUF response of device n
$r_{n,t,l}$	The l-th bit of the t-th measured PUF response of device n
$r\_ref_{n,l}$	The l-th bit of the reference PUF response of device n
$b_{n,m,t,pos}$	The value of the bit on position pos from measurement t for m-th RO pair of device n
$\bar{b}_{pos}$	The majority bit on position pos
$\bar{b}_{n,m,pos}$	The majority bit of device n and RO pair m on position pos
$\bar{b}_{pos}^{dev_n}$	The majority bit of device n across M RO pairs on position pos
$\bar{b}_{pos}^{pair_m}$	The majority bit of RO pair m across N devices on position pos
w	The number of bits selected from each counter value

Table 5.1: Notation used for the introduction of the parameters for bit positions evaluation.

We will use b to denote the resulting counter value of a selected RO pair on a given device, thus $b_{n,m,t,pos}$ denotes the t-th counter value measured for m-th RO pair on device n. The extra index pos is used to denote an individual bit of such counter value.

5.1.2 Parameters for bit positions evaluation

In order to determine suitable positions of the counter values for the PUF we need to evaluate individual bit positions using given parameters. We will use the three following parameters:

- **Bias**
- **Bit stability**
- **Entropy**

We provide their description below.

Bias

Since we need the PUF responses to be unique and random (thus unpredictable) across different devices and challenges, a basic prerequisite is that there has to be a uniform distribution of 0s and 1s without any significant bias.

The bias of the bit position b_i denotes the ratio of 0s and 1s in position i of the counter values. Bias in this work is defined as:

$$P(b_i = 1) = \frac{1}{TMN} \sum_{j=1}^{N} \sum_{k=1}^{M} \sum_{l=1}^{T} b_{j,k,l,i}, \tag{5.2}$$

where $P(b_i = 1)$ stands for the probability of occurrence of 1 at position i and $P(b_i = 1)$ will be used to denote bias in this work. The ideal value of $P(b_i = 1)$ is 0.5.

Bit stability

The stability $s(b_{j,k,i})$ of a bit at the position i from a value measured using one particular RO pair k on device j is defined as follows:

$$s(b_{j,k,i}) = \begin{cases} P(b_{j,k,i} = 1) & \text{if } P(b_{j,k,i} = 1) \geq 0.5 \\ 1 - P(b_{j,k,i} = 1) & \text{if } P(b_{j,k,i} = 1) < 0.5, \end{cases} \tag{5.3}$$

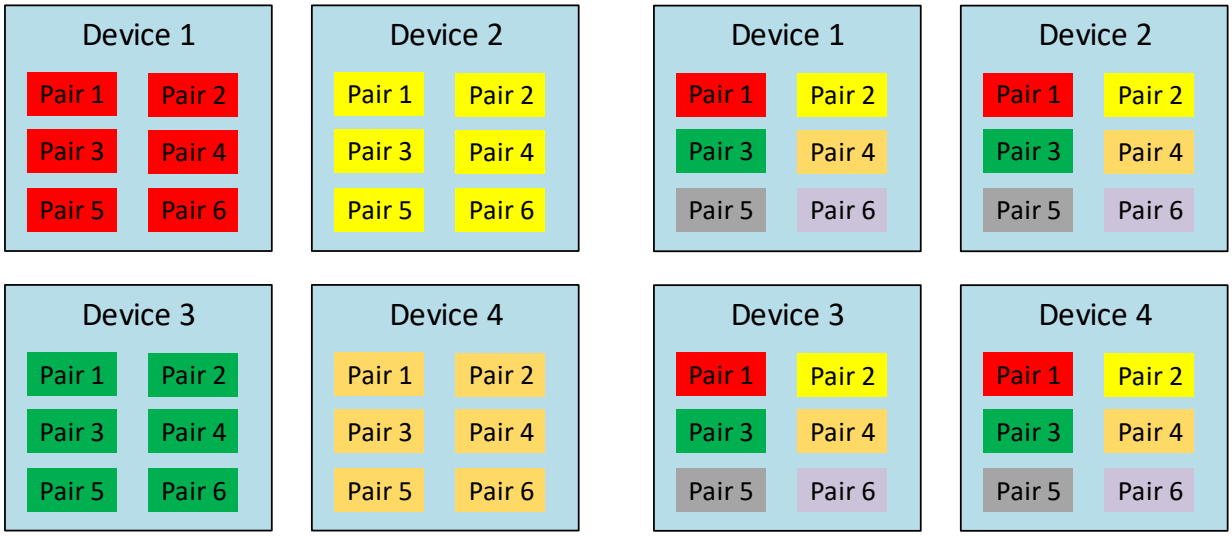
where $P(b_{j,k,i} = 1)$ is the probability of bit value 1 on position i for RO pair k from device j and this value is determined as:

$$P(b_{j,k,i} = 1) = \frac{1}{T} \sum_{l=1}^{T} b_{j,k,l,i}, \tag{5.4}$$

It is obvious that each RO pair has a different stability at various positions of measured values since the ROs in this design are no longer mutually symmetric. For this reason, if we want to perform a suitable selection of positions for the PUF output for all RO pairs, we have to determine the average stability of each position. Provided we have M RO pairs and N devices, the average stability $s(b_i)$ of position i is determined as:

$$s(b_i) = \frac{1}{MN} \sum_{j=1}^{N} \sum_{k=1}^{M} s(b_{j,k,i}). \tag{5.5}$$

Based on the average stability $s(b_i)$ of each position, we can decide which bits are suitable for PUF output. Ideally, we would like the stability $s(b_i)$ of selected bit positions to be equal to 1, but we might not be able to achieve such stability either at all or only on a few bits that are closest to the MSB. Therefore, it is convenient to define a threshold value s_{th} according to which we can select appropriate bit positions. For example, if we choose $s_{th} = 0.95$, then we select all positions from the MSB to the first position where $s(b_i) < 0.95$.



(a) $H_{intra}(b_i)$ evaluation. The entropy is first evaluated within each device separately for all RO pairs, then these values (N values, since the number of devices is N) are averaged to get the average entropy for a given bit position i.

(b) $H_{inter}(b_i)$ evaluation. First, the entropy is evaluated for each RO pair separately across all devices (the given bit position i is compared for the same RO pair among all devices). Then these entropy values (M values, since there are M RO pairs) are averaged to get the value of $H_{inter}(b_i)$ for position i.

Figure 5.3: Difference between evaluation methodology of the entropy $H_{intra}(b_i)$ and $H_{inter}(b_i)$.

Entropy

So far we managed to select appropriate bit positions based on their stability $s(b_i)$. However, in addition to their stability, we have to take into account their uniqueness among different FPGAs. We may assume that if we compare the measured values from two equally positioned RO pairs on two FPGAs, bits close to the MSB will not differ at all while bits approximately in the middle between the most and the least significant bits will vary. It is unnecessary to consider bit positions close to the LSB since it is expected that they will be different due to their instability.

In this work, we will distinguish between two entropies, namely $H_{intra}(b_i)$ and $H_{inter}(b_i)$. $H_{intra}(b_i)$ denotes the entropy of bit position i across all RO pairs calculated within each device separately and then averaged as shown in Fig. 5.3(a). $H_{inter}(b_i)$ is the average entropy of bit position i for each of the RO pairs across different devices.

The average entropy of bit position i within each device separately can be determined as follows:

$$H_{intra}(b_i) = -\frac{1}{N}\sum_{j=1}^{N}\sum_{k=0}^{1} P(\bar{b}_i^{dev_j} = k)\log_2(P(\bar{b}_i^{dev_j} = k)), \tag{5.6}$$

where N is the number of devices and $P(\bar{b}_i^{dev_j} = k)$ is the probability of message k within the j-th device on position i. There are only two possible messages, namely 0 and 1. We compute the probability of their occurrence as:

$$\begin{aligned} P(\bar{b}_i^{dev_j} = 1) &= \frac{1}{M}\sum_{k=1}^{M} \bar{b}_{j,k,i}, \\ P(\bar{b}_i^{dev_j} = 0) &= 1 - P(\bar{b}_i^{dev_j} = 1), \end{aligned} \tag{5.7}$$

where $\bar{b}_{j,k,i}$ represents the i-th majority bit determined from T measurements of the k-th RO pair on the j-th device and M is the number of RO pairs. The majority bit $\bar{b}_{j,k,i}$ is defined as:

$$\bar{b}_{j,k,i} = \text{round}(\frac{1}{T}\sum_{l=1}^{T} b_{j,k,l,i}), \tag{5.8}$$

where round(x) rounds number x to integer. Since the outcome of the expression $\frac{1}{T}\sum_{l=1}^{T} b_{j,k,l,i}$ is a real number lying in the interval $[0, 1]$, the round(x) function provides only the two following results:

$$\text{round}(x) = \begin{cases} 1 & \text{if } x \geq 0.5 \\ 0 & \text{if } x < 0.5. \end{cases} \tag{5.9}$$

The average entropy of bit position i of each of the M RO pairs across different devices (see Fig. 5.3(b)) can be determined using a similar formula:

$$H_{inter}(b_i) = -\frac{1}{M}\sum_{j=1}^{M}\sum_{k=0}^{1} P(\bar{b}_i^{pair_j} = k)\log_2(P(\bar{b}_i^{pair_j} = k)), \tag{5.10}$$

where $P(\bar{b}_i^{pair_j} = k)$ this time is the probability of message k of the j-th RO pair on position i among N different FPGAs. This probability is determined similarly as in the case of H_{intra}, we just calculate it for a particular RO pair among different FPGAs. $P(\bar{b}_i^{pair_j} = k)$ is defined as:

$$\begin{aligned} P(\bar{b}_i^{pair_j} = 1) &= \frac{1}{N}\sum_{k=1}^{N} \bar{b}_{k,j,i}, \\ P(\bar{b}_i^{pair_j} = 0) &= 1 - P(\bar{b}_i^{pair_j} = 1), \end{aligned} \tag{5.11}$$

The ideal value of H_{intra} and H_{inter} is 1 (this is the maximum entropy for 1-bit message). Such a value will guarantee that there is no correlation between bits on the same positions among different FPGAs. For example, the lower the entropy H_{inter} is, the higher is the probability of successful estimation of the bit at a given position on another FPGA, provided we already know this bit from one FPGA.

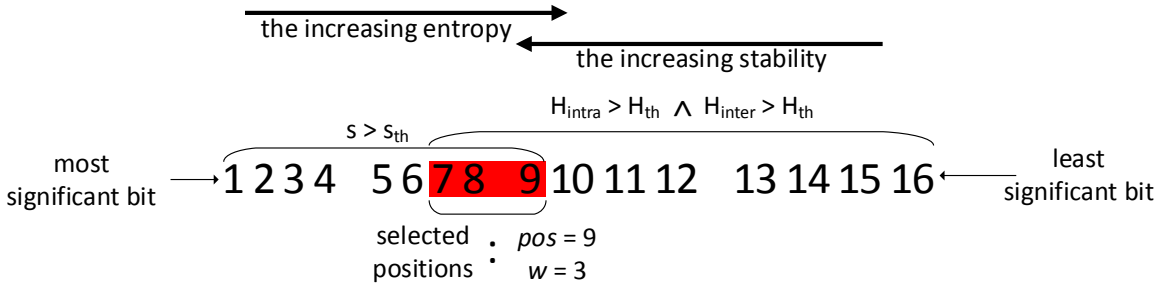


Figure 5.4: The example selection of suitable bit positions for PUF.

5.1.3 Method of selecting suitable bit positions for the PUF

When selecting suitable bit positions for the PUF we have to consider both stability and entropy. The stability is increasing towards the MSB, while the entropy is decreasing in the same direction. Therefore, it is necessary to select such trade-off between good entropy and high stability where both variables are close enough to their ideal value of 1. Based on the statistics, the selection of appropriate bits may proceed as follows:

- We identify the positions where the value of bias $P(b_i = 1)$ is very close to the ideal 0.5.
- We proceed from the most to the least significant bit as long as the stability $s(b_i)$ is higher than the threshold value s_{th} determined by us. As soon as we come across a position where $s(b_i) < s_{th}$, we stop and return back one position. This position is denoted as the *pos* variable.
- We proceed from the *pos* position back towards the MSB. However, this time we consider the entropy values $H_{intra}(b_i)$ and $H_{inter}(b_i)$. We proceed backwards until both entropies satisfy our criteria ($H_{intra}(b_i) > H_{th} \wedge H_{inter}(b_i) > H_{th}$, where H_{th} is the chosen threshold value). This procedure is stopped as soon as the entropy is no longer sufficient. These positions that meet the requirements for bias, stability and entropy are selected. The width w of our selection is determined by the difference of the current position and the *pos* position.

This whole procedure is shown in Fig. 5.4.

5.1.4 The proposed ROPUF circuit

The maximum amount of bits that we are able to extract using the above described method is $\binom{n}{2} \times w$, where n is the number of ROs. It is the number of all possible combinations of RO pairs multiplied by the number of bits that we select from the measured value of one pair. In order to create the PUF output we simply concatenate the chosen suitable bits from the counter values that were measured using all selected RO pairs.

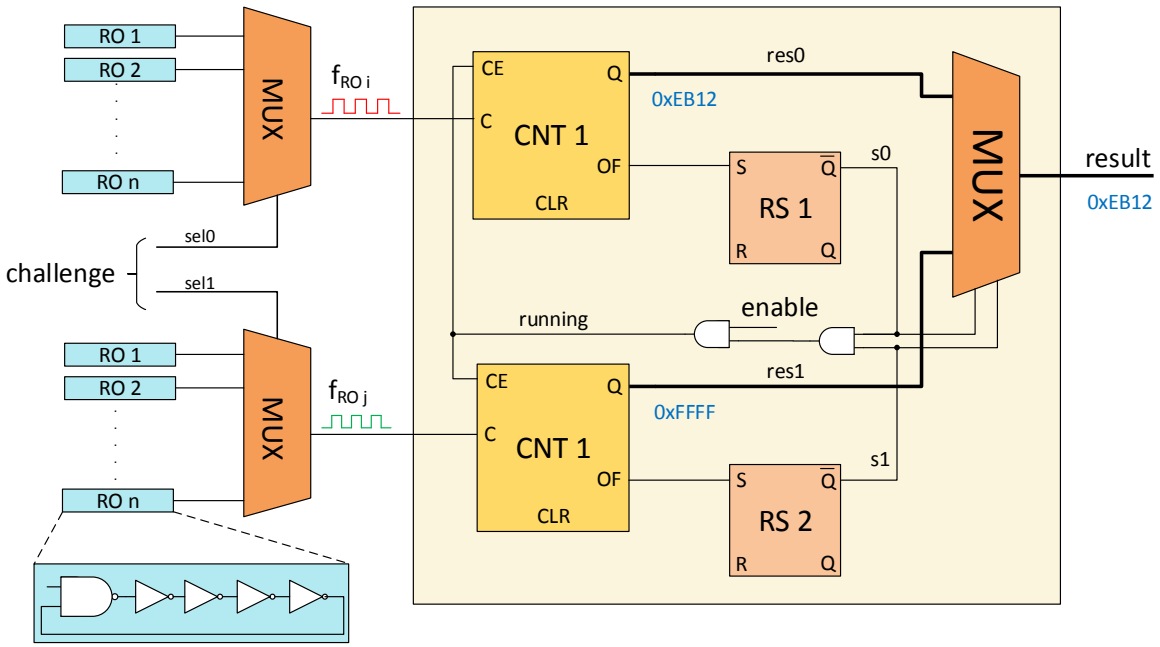


Figure 5.5: The design of the proposed ROPUF.

The circuit that was used for our measurements is shown in Fig. 5.5. There are two sets of ring oscillators. During the measurement of one of the pairs, one RO is selected from each set using the multiplexer and a select signal. All of the ROs are enabled and running during the measurement. The outputs of the two selected ROs are fed into the two counters. When one counter overflows, the measurement is stopped and the value of the counter that did not overflow is selected as the result. The overflow detection is realised by two RS flip-flops as shown in Fig. 5.5.

5.2 Properties of the proposed PUF design

In Section 5.1 we introduced the reader to our PUF proposal that is based on ring oscillators. We described the main concept of the proposed PUF design that consists of counting the number of oscillations for two selected ROs simultaneously using two counters and then processing the value of the counter that did not overflow. In this section we discuss the properties of this PUF design [A.5, A.6].

5.2.1 Global versus separate selection of the appropriate part of the counter values

The method of processing the obtained counter values was described in Section 5.1. In this case, processing means selecting the appropriate part of the counter value, which is

used to form the PUF output. In order to determine the appropriate part of the counter value it is necessary to statistically evaluate each bit position of the counter value from the perspective of stability and entropy. After determining suitable bit positions, all we have to do to generate the PUF output is to perform one measurement for each RO pair and build the PUF output from the selected parts of the counter values we obtained.

In other words, the proposed method consists of two phases:

1. First, repeated measurements for various RO pairs and various devices have to be performed in order to obtain the counter values. Counter values obtained this way are statistically evaluated in terms of their bias, stability, and entropy. A suitable part of the counter values is determined.

2. When generating the PUF output, the measurement is performed only once for each selected RO pair. The part of the counter value that was determined in the previous phase is extracted from each measured counter value and used to form the PUF output by concatenating it with other extracted values.

So far we assumed that the statistical evaluation will be performed globally, i.e. for all RO pairs and devices. However, since the ROs in our proposed PUF design can be mutually asymmetric (therefore they may have very different frequencies) and each RO pair can exhibit a different behaviour in terms of statistical properties of its bit positions, the suitable bit positions for the PUF can also be determined for each RO pair separately.

When determining the suitable bit positions for each RO pair separately, the method remains the same, but the statistical evaluation is performed for each RO pair on each device separately. By determining the suitable bit positions for each RO pair separately, we may achieve a higher stability of the PUF outputs or more bits extracted from each counter value. However, this has the disadvantage of the suitable bit positions for each RO pair having to be stored so that the same part of the counter value can be extracted in each measurement. Storing the suitable positions for each RO pair is the main difference from selecting the positions globally for all RO pairs where we have to store only one position (e.g. the start position and the number of bits that are extracted from this position) that is later used for all RO pairs and devices.

5.2.2 Independence from the maximum operating frequencies of the counters

Another property of this design that we observed is the independence of this design from the maximum operating frequencies of the counters. It is possible that due voltage, temperature, and other variations the frequency of the ROs may exceed the maximum operating frequency of the counter. Even though this phenomenon should be avoided when designing the circuit, it is still possible that due to variation in physical conditions the frequencies of the ROs will increase significantly and exceed the operating frequencies of the counters [A.5].

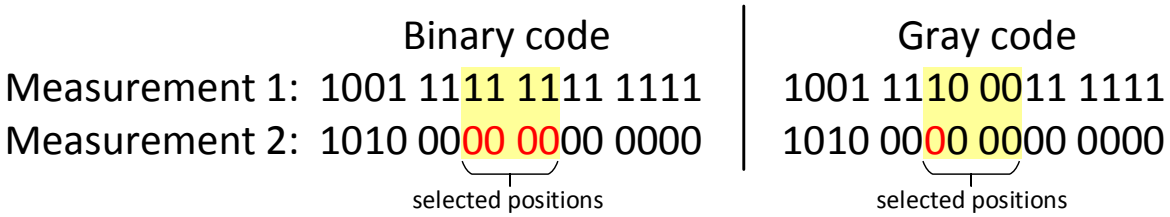


Figure 5.6: Example of counter value overflow and its behaviour when the counter value is represented in binary and Gray code (Gray code is applied only on the selected part of the counter value in this example). Areas of the counter value highlighted in yellow represent the part of the counter value that would be selected for the PUF and the bits that change with the increasing of the counter value in the next measurement are highlighted in red.

When this phenomenon occurs, it results in an incorrect counter value that is read from the counter at the end of the measurement because the counters miss some clock pulses. However, even in these cases the counter values exhibit correct behaviour and the statistical properties of the PUF outputs remain the same. This issue will be furtherly discussed in Chapter 7.

5.2.3 Partial overflow of the counter value

As mentioned in Section 5.1, the counter values are represented as binary values and when selecting an appropriate part of the counter value we can use a specific part of this counter value to form a PUF output. However, since it is a binary value, the number of bits that are changed when the value is different in repeated measurements can be more than only e.g. one bit. Because the counter values depend on ROs, the measured counter values are not always the same due to the instability of ROs and variations in current physical conditions when the measurement is performed.

As it was explained, the resulting counter value is affected by the RO frequencies that depend on various physical conditions. This is reflected by the stability that is evaluated for each bit position, so the instability of the frequencies of the ROs is considered when selecting an appropriate part of the counter values for the PUF. However, there are still cases in which some RO pairs will produce unstable bits even in stable environmental conditions because a considerable amount of bits is changed even when the counter value is increased or decreased by one in the next measurement.

Such case is shown in Fig. 5.6. This figure shows that even though the counter value is increased by one, four bits are changed. When this happens, it shows as a burst of errors in the PUF output and it increases the complexity of the PUF output correction if it is to be used for cryptographic key generation.

The first step of solving this issue involves the application of the Gray code to the obtained counter values [A.5]. The reason for using the Gray code is the fact that two successive values differ in only one bit, so this can eliminate the partial overflow and increase

	Binary	Gray (part)	Gray (whole)
	0111	0100	0100
(+1)	1000	1000	1100
(+2)	1001	1001	1101

Figure 5.7: Comparison of binary and Gray code. Yellow area corresponds to bits that are encoded in Gray code and the differences to the original value are highlighted in red. Only the last three bits are observed in this example (the first bit is neglected).

the overall stability of the selected bits and even increase the number of extractable bits from each counter value. An example is shown in Fig. 5.6. The Gray code is used in the following form:

$$\begin{aligned} g_1 &= b_1 \\ g_i &= b_i \oplus b_{i-1}, \end{aligned} \tag{5.12}$$

where g_i is the i-th bit of the value represented in Gray code and b_i is the i-th bit of the value in binary code. b_1 and g_1 are MSBs of the value encoded into Gray code.

There are two possible ways to apply the Gray code to the counter values [A.5]. It can be applied either to the whole counter value or only to the selected part. Our observations indicate that when the Gray code is applied to the whole counter value, the stability of the PUF outputs is higher and the uniqueness of the PUF outputs is lower than when only encoding the selected part of the counter value. Therefore the choice must be determined by the actual situation and our preference.

The difference of statistical properties is caused by the definition of the Gray code (5.12). This definition implies that each bit in the encoded value depends only on the current and the preceding bit and is not influenced by any bits in the direction to LSB. Therefore the statistical properties of the PUF will remain the same if we apply the Gray code either to the selected part of the counter value (e.g. positions 7–8) or to the whole part of the counter value from the first selected bit position to LSB (e.g. 7–16 in case of 16-bit counter value). However, different statistical properties may be achieved when the Gray code is applied to some of the bits closer to MSB or to the whole counter value. This is shown in Fig. 5.7 at 4-bit values where the behaviour of the last three bits is observed and the values are compared to the first value. When the value is increased by 1 and 2 then in the case of the Gray code being applied only to the last three bits the difference will show in one and two bits. However, when the Gray code is applied to the whole value, then there is no difference (value increased by 1) or only in one bit (value increased by 2). The described

behaviour implies that the PUF is more stable when the Gray code is applied to the whole counter value.

As mentioned before, the method of the Gray code application to the counter value does not only influence the stability but also the uniqueness of the PUF outputs. The uniqueness of the PUF outputs is lower when the Gray code is applied to the whole counter value for the same reason as why the stability is higher (see Fig. 5.7). The counter values for the same RO pairs on different FPGAs are close to each other but they also differ enough so that we can distinguish them. However, if the difference between the counter values is not large enough, by applying the Gray code to the whole counter values, the difference between these values may not affect the selected positions that are used for the PUF outputs.

5.2.4 Influence of physical conditions

The basic building element of this PUF design is an RO, hence the physical conditions such as variations in temperature or supply voltage will have impact on the RO frequencies and therefore on the behaviour of the PUF. In this proposal, one RO pair is connected to two counters in each measurement and the number of oscillations is counted for the two ROs until one of the counters overflows and the measurement is stopped. Therefore, if we knew the exact frequencies of the ROs during measurement, we could derive the resulting counter value as it was shown in Eq. 5.1:

$$counter\ value = \frac{f_2}{f_1} 2^l,$$

where f_1 is the frequency of the faster RO, f_2 is the frequency of the slower RO, and l is the size of the counter in bits.

From Eq. 5.1 it is clear that the resulting counter value is dependent on the ratio of the frequencies of the two selected ROs. It can be expected that when the supply voltage or temperature is changed, the frequencies of the ROs will be affected in almost the same way.

Ideally we want the ratio of the frequencies to remain constant in time. In the context of Eq. (5.1) it means that the paired RO frequencies would be modified (multiplied) by the same constant k. However, as it will be shown in Chapter 7, the ratio of the frequencies is not constant, but the ratios of the frequencies of the ROs measured in various physical conditions are close each other and in some cases the change in the ratio does not affect the bit positions of the counter values that are used for the PUF.

The fact that the ratio of the RO frequencies is not constant also implies that the frequencies of the two ROs in a pair are not multiplied by the same constant k but ratherly two different constants k_1 and k_2. In order to achieve stable PUF outputs, these constants should be almost the same ($k_1 \approx k_2$).

Since the resulting counter value is determined by the ratio of the frequencies of the two paired ROs (see Eq. (5.1)), this method can be considered a differential measurement. It will be shown in Chapter 7 that the influence of supply voltage or temperature on the frequencies of the ROs is significant. However, if the change in the frequencies is almost

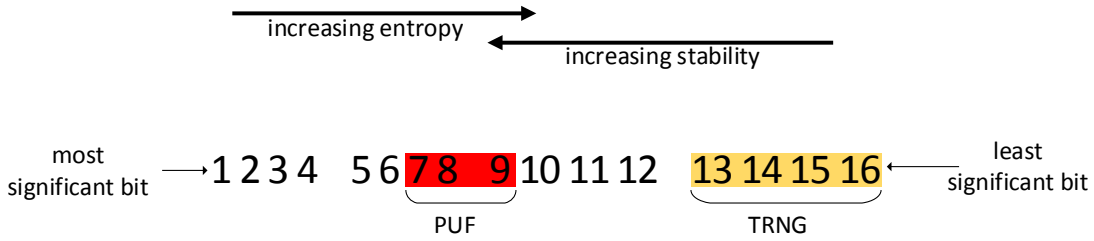


Figure 5.8: The example behaviour of the bits in counter value of a 16-bit value.

the same, the changes in frequencies will not have a large impact on the resulting counter value since the resulting counter value depends on the ratio of the frequencies.

It can be expected that the RO frequencies will change in a similar way when the ROs are mutually symmetric [A.5, A.6]. Therefore, the ratios of RO pair frequencies should be almost constant. In our PUF proposal, we stated that compared to classical approach [58], our PUF design does not require the ROs to be mutually symmetric. However, it will be shown in Chapter 7 that when using symmetrical ROs, the proposed PUF exhibits better behaviour in terms of stability at varying voltage or temperature.

5.3 True random number generator based on the proposed PUF design

In this section we will describe how the same design originally intended to be used as a PUF can also be used as a TRNG [A.1, A.2]. It can be very beneficial to have both PUF and TRNG based on the same design as both of these cryptographic primitives are important for security. Having both a PUF and a TRNG enables us to provide a secure key storage, identification, authentication, and cryptographic key generation.

5.3.1 Utilization of the proposed design for PUF and TRNG

The counter values are represented in binary code and therefore we can select an appropriate part of counter values for the PUF output based on statistical properties of the selected bit positions when we take into account their stability and entropy. The same applies to TRNG. We can expect that the bit positions close to the LSB will be highly unstable and will vary with almost each measurement due to noise effects - a possible source of entropy for the TRNG. The described behaviour is depicted in Fig 5.8.

Taking into account the desired properties of PUFs, the suitable positions for the PUF are located somewhere in the middle of the counter value, where both entropy and stability are high. It is important to realize that the sources of entropy for PUFs and TRNGs differ significantly. The entropy for PUFs is given to a particular circuit only once during the manufacturing process. Therefore, the entropy is determined for given positions among

various RO pairs or the same RO pair on various devices. However, the TRNG entropy is given by the RO jitter, so the entropy is determined for each RO pair using multiple measurements. This implies that the suitable positions for TRNG will be the ones close to the LSB.

5.3.2 TRNG evaluation

There are various statistical properties of TRNG designs we can evaluate, such as bit rate, area efficiency, or sensitivity to physical disturbances. However, the most important property of any TRNG is the unpredictability of its output. Therefore, we should carefully evaluate the TRNG in the perspective of the randomness it offers.

There are some recommendations and guidelines on how to evaluate TRNGs, e.g. the German document AIS 31 [25]. In order to properly evaluate a TRNG, it is not sufficient to only test the generated sequences of bits by the TRNG because even a deterministic random number generator can pass test suites such as the NIST or the DieHard. These tests may be necessary, but even if the tested RNG passes these tests, it doesn't mean that it really is a TRNG.

The problem is that the generated sequences were already digitized and we evaluate them once the algorithmic post-processing is finished. This enhances their statistical properties. What we need to do is to make a stochastic model of the noise and compute a lower bound of the entropy per bit of the source of the entropy [16].

First, we need to identify the source of randomness [16]. TRNGs rely on a random physical phenomenon known as the analog physical noise. Therefore, the analog physical noise is the source of randomness we need to identify. There might also be some other unidentified phenomena that would contribute to the randomness of the TRNG but it should not be taken into account in the entropy estimation. After identifying the source of randomness, we need to make a statistical model for the physical noise used.

Once the statistical model of the physical noise is finished, one must be able to experimentally evaluate the parameters of said model and evaluate the measurement errors of these parameters. Also, the parameter stability of statistical model must be evaluated for physical noise with regard to physical environmental conditions of the TRNG (temperature, supply voltage...) and technological environmental operating conditions of the TRNG (it is either installed alone on a circuit or with other circuits).

In order to evaluate the TRNG properly it is also required to have a statistical model for the TRNG (i.e. the bits it generates). It is assumed that all of the conditions mentioned above are fulfilled because the statistical model for the physical noise is needed. To ensure that the TRNG is working properly during its life span, parametric tests must be ran at the start-up and continuously.

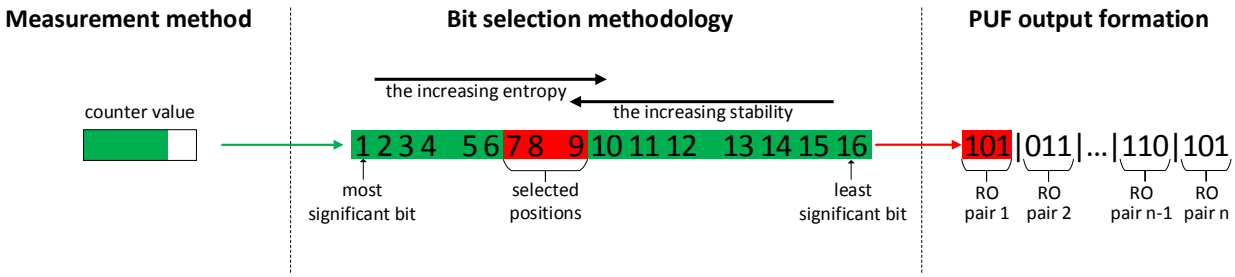


Figure 5.9: The main concept of the presented PUF. This measurement method provides the 16-bit counter value for selected ROs. Positions of the counter values with both high entropy and high stability are then selected to be used in the PUF output.

5.4 Different measurement methods

The measurement method that is used in our proposed ROPUF is based on a ratio of the selected RO pair frequencies. This ROPUF proposal was described in previous sections. This section describes the option of substituting the measurement method in our proposal for another without any significant changes in the design itself.

We will present two additional measurement methods of the frequency ratio that may be used to construct a very similar ROPUF. These measurement methods are *frequency difference* and *crystal reference*.

When these other two measurement methods are used, the only difference of our ROPUF construction is that the measured counter value will result from slightly different RO interaction and also that they both require a stable reference clock, typically a crystal oscillator. Otherwise, the proposal remains the same as shown in Fig. 5.9. This means that we can still use the methodology presented in Section 5.1.3 for selecting suitable bits for the PUF output. Also, the resulting PUF output is formed by a simple concatenation of the counter value selected bit positions.

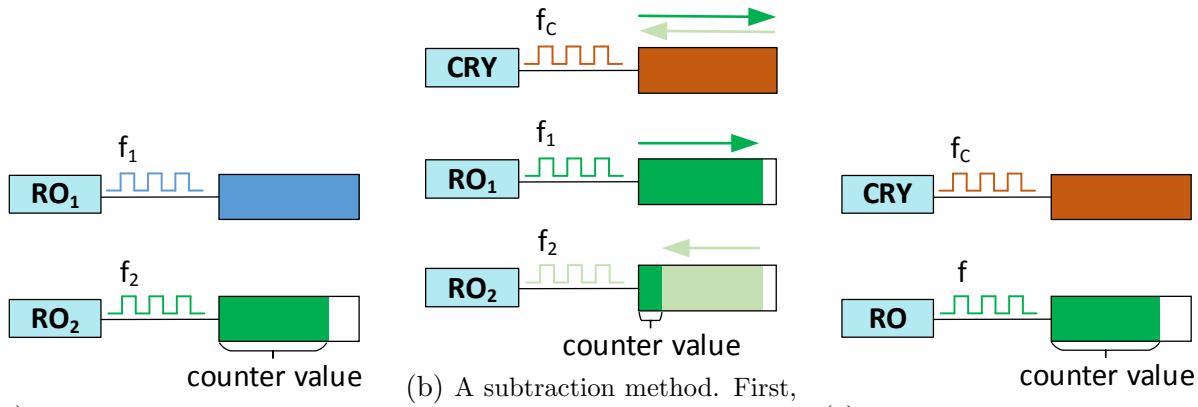
Together with the measurement method already presented, the three measurement methods are:

- **Frequency ratio**
- **Frequency difference**
- **Crystal reference**

These measurement methods are described in the following subsections.

5.4.1 Frequency ratio

The first presented PUF construction is based on frequency ratios of the selected RO pairs. This proposal was already described in the previous sections and published in [A.3, A.4]. The main principle of this measurement method is shown in Fig. 5.10(a). The selected RO



(a) A frequency ratio method that is used in our original PUF design. The resulting counter value is given by the ratio of the frequencies f_1 and f_2. There is no reference clock as in the following cases. It is assumed that $f_1 > f_2$

(b) A subtraction method. First, the value in the counter is incremented for every rising edge of the first RO (RO_1). Then the counter value is decremented for every rising edge of the second RO (RO_2). The incrementing and decrementing runs for the same amount of time given by a crystal oscillator (CRY) with a stable frequency f_C. The assumption is that $f_C > f_1 > f_2$.

(c) A basic approach where the counter value is accumulated for each RO and for the same amount of time given by a crystal oscillator (CRY) with a stable frequency f_C that serves as reference clock. It is assumed that $f_C > f$.

Figure 5.10: Three different counter value measurement methods.

pairs are connected to two counters that are counting the number of oscillations for both of the ROs simultaneously. As soon as one of the counters overflows, the measurement is stopped. The resulting value from the counter that did not overflow is then processed and used for the PUF output. If the frequencies of the ROs in the pair were known, the resulting counter value could be determined as:

$$counter\ value = \frac{f_2}{f_1} 2^l \ , \tag{5.13}$$

where f_1 is the frequency of the faster RO, f_2 is the frequency of the slower RO and l is the size of the counter in bits.

5.4.2 Frequency difference

Another measurement method is the subtraction method, where the frequencies of the paired ROs are subtracted. This method requires a stable reference oscillator (e.g. crystal oscillator). At first, the number of periods of the first RO are accumulated in the counter for a time interval given by the reference oscillator. Then, the second RO decrements the counter value for the same time interval. The resulting counter value is given by:

$$counter\ value = (f_1 - f_2)t \ . \tag{5.14}$$

We assume that $f_1 > f_2$ and t is a timebase given by the reference clock used for counting of the oscillations for both ROs. This measurement technique is shown in Fig. 5.10(b).

5.4.3 Crystal reference

The last presented measurement method does not use RO pairs. It always has one stable reference clock (crystal oscillator) and one RO in a pair as shown in Fig. 5.10(c). As in the previous cases, the counter accumulates the number of periods of the selected RO for a given time interval. This is determined by the reference clock. The resulting counter value can be determined as:

$$counter\ value = ft\ . \tag{5.15}$$

5.5 Overview of the proposed method

In this section we provide an overview of the proposal that was presented in the beginning of this chapter. For better clarity, the proposal is summarized in Fig. 5.11.

For explanation purposes, the proposed method depicted in Fig. 5.11 is divided into two parts. They are the *Design and evaluation* and *Usage*. The goal of the *Design and evaluation* is to create and implement the ROPUF design and to select suitable positions of the counter values that will be used for either PUF or TRNG output.

First, the ROPUF design has to be created, e.g. the one in Fig. 5.5. The next step is to perform the measurements in order to find out suitable positions of the counter values. To obtain the counter values, any of the three measurement methods presented in Section 5.4 can be used. The results of the measurement are T counter values obtained for each of the M RO pairs (or single ROs in case of the crystal reference measurement method) from N devices.

In the evaluation step, the individual bit positions b_i are evaluated according to whether they are used for a PUF or a TRNG. In the case of PUF, the positions are evaluated using the parameters presented in Section 5.1.2 (bias, stability and entropy) that are used to determine the suitable bits for the PUF output. When evaluating bit positions suitable for the TRNG, the NIST STS can be used. As a result of the *Design and evaluation* part, the suitable positions for both PUF and TRNG are determined.

The next part, *Usage*, is the actual utilization of the PUF/TRNG circuit in real application, such as identification, authentication, or key generation. First, the RO pairs (or single ROs) have to be measured using one of the measurement methods. Only one single measurement is performed when we are interested only in the raw PUF output. For creating a TRNG output, we can either measure multiple RO pairs, or a single RO pair multiple times until we obtain a random sequence of desired length.

After measurement, the bits for both PUF and TRNG are extracted according to the determined suitable bit positions. Next is the processing step. In the case of PUF this means applying the Gray code on the selected bits as shown in Fig. 5.11 (it can also be applied to the whole counter values). In the case of TRNG, some post-processing may be

required, e.g. the von Neumann corrector can be used. Finally, the resulting PUF response $R_{n,t}$ or the TRNG output is created by a concatenation of the selected bits.

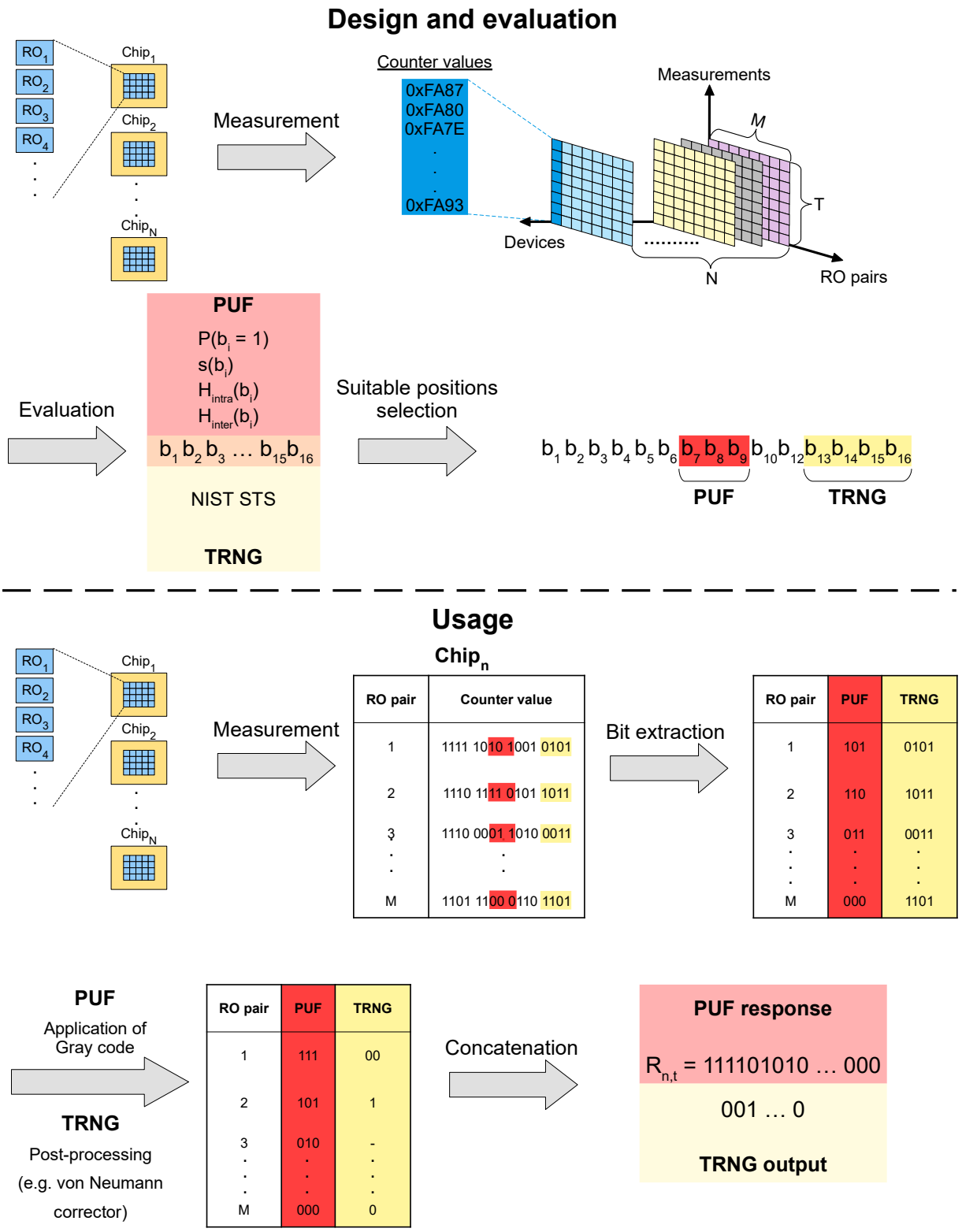


Figure 5.11: Overview of the proposed PUF design.

Chapter 6

Implementation

This chapter describes of the implementation of our ROPUF proposal on FPGAs. In our early work [A.3, A.4, A.5] our main target platform was the Digilent Basys 2 FPGA board (Xilinx Spartan3E-100 CP132) [13]. We performed additional measurements on Digilent Nexys 3 FPGA boards (Xilinx Spartan-6) [14] with the same ROPUF design, and finally, to evaluate the proposed ROPUF on modern low-cost FPGAs, we used Digilent Cmod S7 FPGA boards (Xilinx XC7S25-1CSGA225C) [15] for extensive experiments. VHDL was used to describe hardware designs on all implementation platforms.

The following sections describe the implementation of our experimental ROPUF circuits on the above mentioned platforms.

6.1 Spartan-3E and Spartan-6

We implemented the ROPUF design presented in Chapter 5 in Fig. 5.5 on both Spartan-3E and Spartan-6 using Xilinx ISE 14.7 [64]. The FPGAs were programmed with the resulting bitstreams by Digilent Adept 2 [11]. The communication between the PC and the FPGA was realised through a USB using DEPP interface (Digilent Adept Asynchronous Parallel Port Interface). This is a library that is a part of Adept SDK [12] which, among other things, enables data transfer between a PC and a target device. Adept SDK provides API in C language that can be used for interaction with various FPGA boards.

The design implemented on both platforms is the one presented in Fig. 3.13. Since we performed our measurements in order to evaluate our proposal for both PUF and TRNG, we used only slightly different implementations of the same design.

6.1.1 PUF

For PUF evaluation, we used a design consisting of two sets of ROs, with 150 ROs each. The ROs in this design are implemented as a combinatorial loop consisting of one NAND gate and four inverters as shown in Fig. 3.13, where each gate is implemented as a separate LUT on the FPGA. During the measurements all of the ROs are enabled and running.

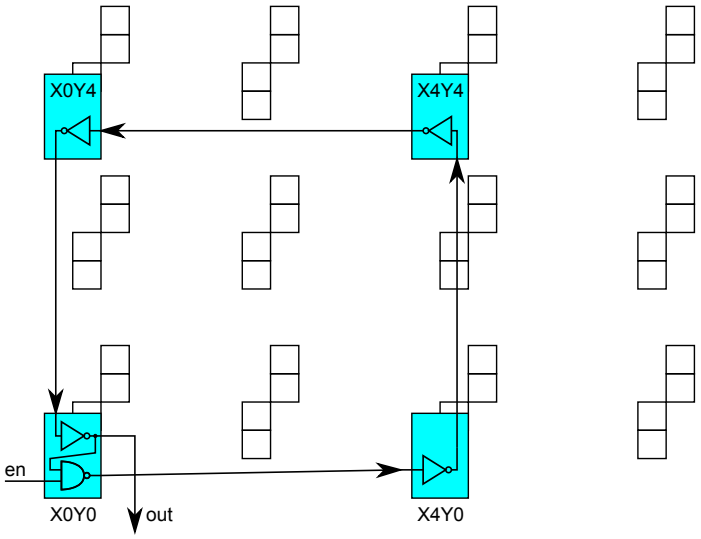
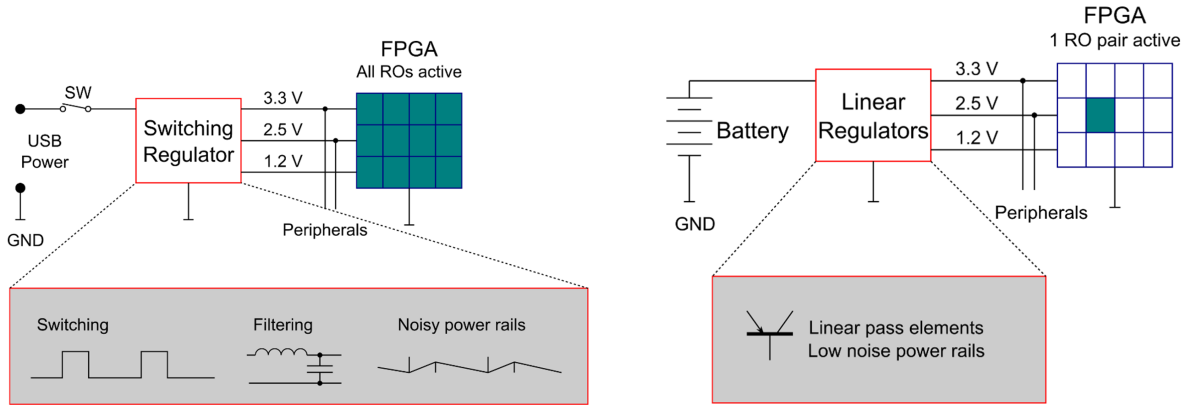


Figure 6.1: Relative mutual placement of logic gates used for each RO on Xilinx Spartan3E-100 CP132.

We performed experiments with both symmetric and asymmetric ROs. In case of asymmetric ROs, we do not place any constraints on the placement of the individual gates of ROs, thus the placement is left to the automatic placer of ISE 14.7.

For symmetric RO implementation we placed the logic gates of each RO in order to have the ROs placed on the FPGA mutually symmetric. The RO gates were placed using RLOC constraints that were the same for all ROs. We used 5-stage ring oscillators (see Fig. 3.13) that were placed into four slices, with each inverter occupying one slice and the last one sharing the slice with the NAND gate. The relative locations of the slices were (X0, Y0), (X4, Y0), (X4, Y4), and (X0, Y4) as shown in Fig. 6.1. Other placements were left unconstrained and the choice was left to the ISE 14.7 automatic placer. It should also be noted that the constraints were put only on the relative placement of the gates of each RO but not on the interconnects between them. Therefore, even though the placement of ROs is mutually symmetric, the ROs themselves are not absolutely symmetrical since the interconnects between the gates of each RO can be different.

Using symmetric ROs we were able to achieve a higher stability of the PUF outputs at varying voltage or temperature. The statistical properties of the PUF itself remained similar. All of this will be shown in the following Chapter 7. Otherwise, the design remained the same, only the number of ROs in each set of ROs was 50 instead of 150 as in the case of asymmetric ROs. Again, all ROs were enabled and running during the measurements.



(a) Experimental setup with a switching regulator used as power supply. All RO pairs are running during the measurements.

(b) Experimental setup with switching regulators substituted by linear regulators to be used as a power supply. Only one RO pair is active during the measurements.

Figure 6.2: The experimental setups for TRNG measurements [A.2].

In summary, the following implementation variants were used for PUF evaluation:

- Spartan-3E
  - Asymmetric ROs, 2×150 ROs, all running
  - Symmetric ROs, 2×50 ROs, all running
- Spartan-6
  - Asymmetric ROs, 2×150 ROs, all running

6.1.2 TRNG

Since we want to use the same design for both PUF and TRNG, we did not make any changes and used the same implementation of the design shown in Fig 5.5 and described in the previous section for our experiments. The circuit consisted of two sets of ROs with 150 ROs each, all ROs were running during the measurements (see Fig. 6.2(a)). The implementation variant with asymmetric ROs is used.

However, since these measurements were performed on Digilent Basys 2 FPGA boards that use an on-board switching regulator as the power supply and all ROs were running during the measurements, there might be additional noise present in the measured values that are later evaluated for randomness.

To eliminate any potential crosstalks between individual RO pairs and parasitic frequencies resulting from switching regulators influencing randomness of generated bitstream and therefore to verify that each RO pair can be considered a unique source of entropy, we tested individual RO pairs separately using a set of linear regulators as a power supply.

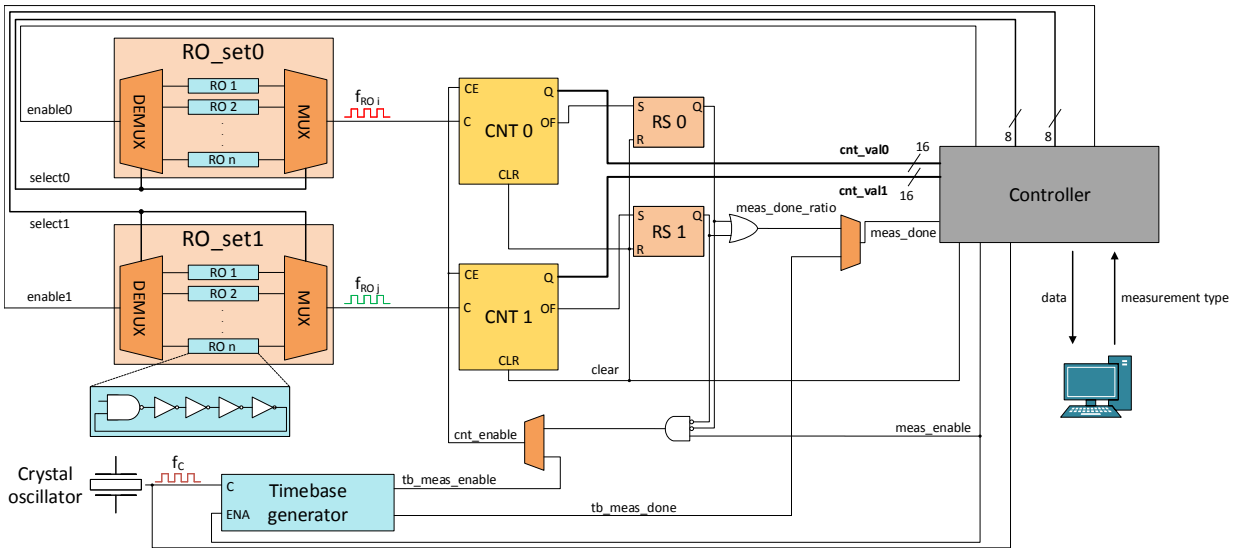


Figure 6.3: The experimental circuit realizing the three PUF measurement methods.

In this implementation, all ROs were not running simultaneously as before. Only one selected pair of ROs was running during the measurement. Therefore, the circuit contained only 130 ROs in each RO set instead of the original 150 ROs. The lower number of RO pairs is caused by additional logic needed to allow each RO pair to be run separately. The Digilent Basys 2 prototyping board was modified so that the original power supply circuit was disconnected. The new power supply consists of a battery and linear regulators as shown in Fig. 6.2(b).

All experiments regarding TRNG evaluation were performed on Spartan-3E FPGAs and only asymmetric ROs were used. This is the final list of the implementation variants:

- 2×150 ROs, all running, on-board switching regulator
- 2×130 ROs, only selected ROs are running, linear regulators

6.2 Spartan-7

In Chapter 5 we described our proposed ROPUF together with two additional measurement methods that can be used in our design. This section is devoted to implementation details of the experimental circuits that were finally used for measurements on Digilent Cmod S7 FPGA boards (Xilinx XC7S25-1CSGA225C) [15]. The designs were implemented using the Vivado Design Suite v2019.1 [68].

For experimental evaluation we combined all of the three PUF measurement methods described in Section 5.4 into one circuit. Our design used for experiments is shown in Fig. 6.3. The design contains two sets of ROs, each composed of 150 ROs. Signals *select0* and *select1* are used to select the ROs from these two different RO sets. The selected ROs

are then enabled using the *enable0* and *enable1* signals and connected to counters *CNT0* and *CNT1* that are both 16 bits long.

The *Timebase generator* is clocked by the on-board Crystal oscillator that is used as the reference oscillator for frequency difference and crystal reference measurement methods. The Crystal oscillator has a frequency of 12 MHz. The *Timebase generator* is used to generate an enable signal (*tb_meas_enable*) that is 130.8μs long (t in Eq. 5.14 and Eq. 5.15). In case of the crystal reference measurement method, the *tb_meas_enable* signal is used to enable one of the counters $CNT0$ or $CNT1$ for counting the number of periods of the selected RO. For frequency difference measurement, this enable signal is used to measure the number of periods for the selected pair of ROs. However, those measurements are not done simultaneously, but in a sequence so that the two ROs are not running at the same time. This is done to minimize their mutual influence.

Finally, the *Controller* is a finite state machine that controls the operation of the whole circuit. It selects and enables the ROs that are to be used for the measurement using the signals *enable0*, *enable1*, *select0*, and *select1*. The *Controller* also reads resulting counter values *cnt_val0* and *cnt_val1* and controls which type of measurement method is performed.

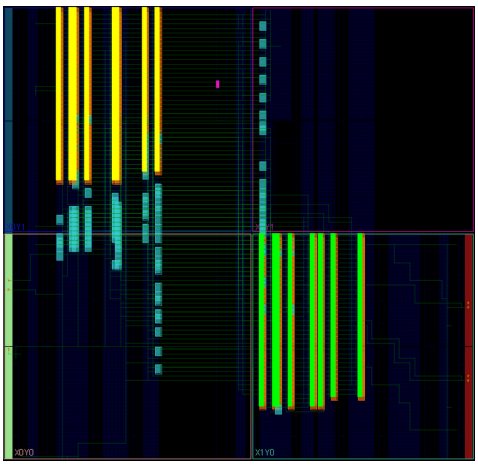
The counters are enabled either using the signal *meas_enable* from the *Controller* or *tb_meas_enable* which is provided by the *Timebase generator*. When the frequency ratio method is selected, the counters run simultaneously (they are enabled using the *meas_enable* signal) and when one of them overflows, the measurement is stopped and the *Controller* reads the value of the counter that did not overflow. The overflow detection is implemented using two RS flip-flops (*RS0* and *RS1*).

In case of the frequency difference measurement method, the *Controller* enables the two ROs in a pair sequentially (not simultaneously) for the same amount of time given by the *Timebase generator*. When the oscillations are accumulated in both of the counters, the result is given by the difference between them.

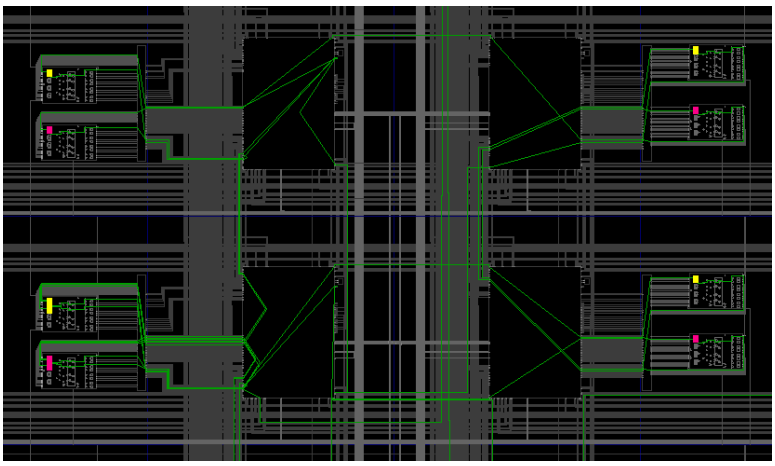
For the last measurement method, the crystal reference, the *Controller* selects and enables one RO and enables the corresponding counter for a fixed time interval provided by the *Timebase generator*. The resulting value is then just read from the counter.

The communication between the FPGA and the computer is realized by a serial line. This communication channel is used both for choosing the type of the measurement by the computer and sending all of the measured counter values by the *Controller*.

The ROs in this design are implemented as a combinatorial loop consisting of one AND gate and four inverters. Each gate is implemented as a separate LUT on the FPGA. We used FPGA boards containing Spartan-7 that uses six-input look-up tables (LUT6) to implement the user functions. These LUTs can be used either as a single LUT6 or two LUT5 (five-input LUTs). In our design, each gate is implemented as a separate LUT6 on the FPGA and we set the property "LUTNM" of each gate to "DISABLED" in order to avoid a LUT combination. If this property is not set, the result can be two gates from different ROs implemented in the same LUT6.



(a) Placement of symmetric ROs on the FPGA. The gates of the ROs for the first set of ROs are highlighted in yellow, the other set is highlighted in green.



(b) A more detailed look on the placement of individual gates of one RO. The gates for the presented RO are highlighted in red.

Figure 6.4: Placement of ROs for the design with symmetric ROs.

Finally, the whole design was implemented in three slightly different variants:

- **Asymmetric ROs**
- **Asymmetric ROs, all enabled**
- **Symmetric ROs**

When asymmetric ROs are used, it means that there are no constraints on the placement of ROs and their gates. This task is completely left to the synthesis tool. Normally, we assume that only the ROs selected for the particular measurement are enabled and running and all of the others are in an idle state. However, to investigate the mutual influence of the ROs, we also have to have one implementation with all ROs enabled (asymmetric ROs, all enabled).

The last variant of our implementation uses symmetric ROs. In this case, symmetric ROs have the same relative gate placement. There are no constraints on the routes between individual gates, only on the placement of the gates. For relative gate placement, we used the RLOC attribute with the coordinates set to "X0Y0", "X2Y0", "X2Y1", "X0Y1", and "X0Y0" (the order corresponds to the order of the gates forming the RO). The resulting placement on the FPGA is shown in Fig. 6.4(b). Moreover, the two sets of ROs in this implementation variant are placed on the opposite sites on the FPGA as shown in Fig. 6.4(a) to minimize their mutual influence when running during the measurement. Only the selected ROs are enabled and running during the measurement.

The utilization of resources on Spartan-7 FPGA in terms of LUTs and FFs (flip-flops) for all of the three implementation variants is shown in Table 6.1. This overview of the

	Asymmetric ROs	Asymmetric ROs, all enabled	Symmetric ROs
LUT	2139	1809	2140
FF	240	239	240

Table 6.1: Utilization of resources of the three implementation variants on Spartan-7 FPGA.

used resources includes all of the elements present in the implemented circuit from Fig. 6.3 used for measurements, such as the *Controller* that is used to switch between measurement methods and perform the measurements, and also the necessary logic used to implement the communication with the computer. However, a dominant part of the resources is used by the two sets of ROs, where each RO set is composed of 792 LUTs (in all implementation variants). It can be noticed in Table 6.1 that the implementation variant using asymmetric ROs that are all enabled uses less LUTs in the design. This is due to simpler logic as all ROs are enabled during the measurements and they are not enabled selectively as it is the case of the other two implementation variants.

Chapter 7

Experimental Results

This chapter presents the results of our experiments related to the PUF proposal we presented in Chapter 5. The implementation of experimental circuits that are used for measurements were described in Chapter 6 and the evaluation parameters used throughout this chapter were defined in Chapter 4.

We divided this chapter into three sections based on the implementation platforms we used for our experiments. The three experimental platforms were:

- **Digilent Basys 2** FPGA board [13] containing **Xilinx Spartan3E-100** CP132 [65]
- **Digilent Nexys 3** FPGA board [14] containing **Xilinx Spartan-6** XC6LX16-CS324 [66]
- **Digilent Cmod S7** FPGA board [15] containing **Xilinx Spartan-7** XC7S25-1CSGA225C [67]

In our early work related both to PUF [A.3, A.4, A.5, A.6] and TRNG [A.1, A.2] we used Digilent Basys 2 FPGA boards for our experiments. As Spartan-3E on Basys 2 is manufactured with the 90nm technology, we also performed additional measurements on more modern FPGAs, specifically the 45nm Spartan-6 on Digilent Nexys 3 FPGA boards. These measurements were performed only to evaluate the ROPUF. Finally, we performed extensive measurements on Digilent Cmod S7 FPGA boards containing Spartan-7 FPGA (28nm technology) to compare different implementations and measurement methods. The results are presented in the following sections.

7.1 Spartan-3E

This section is devoted to presenting experiments performed on Digilent Basys 2 FPGA boards [13] containing the Xilinx Spartan3E-100 CP132 [65]. The design used for these experiments is shown in Fig. 5.5. We divide this section into several subsections based on the nature of the experiments such as the implementation with asymmetric/symmetric

	150 RO pairs				450 RO pairs			
position(i)	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$
1	1.0000	0.1414	0.0000	0.9800	0.9999	0.5473	0.0132	0.8736
2	0.9996	0.9477	0.0557	0.3663	0.9997	0.9831	0.0395	0.4239
3	0.9996	0.9944	0.0794	0.5430	0.9995	0.9949	0.0937	0.5410
4	0.9985	0.9962	0.1657	0.5336	0.9989	0.9979	0.1898	0.5248
5	0.9983	0.9992	0.2955	0.4998	0.9981	0.9982	0.3598	0.5233
6	0.9950	0.9941	0.7183	0.4765	0.9961	0.9973	0.6585	0.5233
7	**0.9908**	**0.9958**	**0.9475**	**0.5056**	**0.9921**	**0.9982**	**0.9232**	**0.5118**
8	**0.9815**	**0.9954**	**0.9663**	**0.4959**	**0.9841**	**0.9986**	**0.9682**	**0.4978**
9	**0.9650**	**0.9946**	**0.9639**	**0.4931**	**0.9677**	**0.9984**	**0.9692**	**0.4971**
10	0.9297	0.9954	0.9681	0.4997	0.9347	0.9983	0.9706	0.5047
11	0.8625	0.9943	0.9701	0.4960	0.8709	0.9984	0.9707	0.5042
12	0.7268	0.9957	0.9728	0.4985	0.7441	0.9981	0.9709	0.4969
13	0.5569	0.9932	0.9732	0.5002	0.5691	0.9987	0.9706	0.5005
14	0.5139	0.9961	0.9654	0.4985	0.5185	0.9982	0.9727	0.4996
15	0.5135	0.9957	0.9647	0.4975	0.5181	0.9981	0.9663	0.4991
16	0.5140	0.9897	0.9613	0.4967	0.5182	0.9972	0.9687	0.4977

Table 7.1: Evaluation of individual bit positions of 16-bit counter values for 150 and 450 RO pairs (1000 and 500 measurements respectively) obtained from 24 Digilent Basys 2 FPGA boards.

ROs, the influence of voltage and temperature and also the usage of the proposed design as a TRNG.

7.1.1 Asymmetric ROs

First, we will present the results of the original design containing asymmetric ROs, i.e. there are no constraints on the placement of the individual RO gates or their interconnects. The measurements were performed on 24 Digilent Basys 2 FPGA boards. The circuit we used contained 300 ROs divided into two groups of 150 ROs each (see Fig. 5.5). The ROs are ordinary 5-stage ROs as shown in Fig. 3.13 and their oscillations are counted by 16-bit counters.

We performed the measurements for 150 RO pairs (each oscillator from the first group was paired with another unused oscillator from the second group) and 1000 measurements were executed for each pair. Another measurement was performed for 450 pairs with 500 measurements for each pair.

Selection of suitable positions

First, we will present the results of the evaluation of individual counter value bit positions. We will evaluate stability (s_i), entropy (H_{intra}, H_{inter}) and bias ($P(b_i = 1)$) for each position of the 16-bit counter values. See Section 5.1.2 for the description of these parameters. As can be seen in Table 7.1, the entropy rises and is approaching the ideal value of 1 (especially

No Gray code

	150 RO pairs					450 RO pairs				
positions	6–8	7–8	7–9	8–9	7–10	6–8	7–8	7–9	8–9	7–10
w [-]	3	2	3	2	4	3	2	3	2	4
HD_{intra} [%]	1.09	1.38	2.09	2.67	3.32	0.92	1.19	1.87	2.41	3.04
HD_{inter} [%]	44.27	49.15	49.31	49.70	49.46	42.69	48.42	48.94	49.96	49.23

Gray code part

	150 RO pairs					450 RO pairs				
positions	6–8	7–8	7–9	8–9	7–10	6–8	7–8	7–9	8–9	7–10
w [-]	3	2	3	2	4	3	2	3	2	4
HD_{intra} [%]	0.64	0.94	1.20	1.78	1.80	0.53	0.80	1.08	1.62	1.65
HD_{inter} [%]	38.64	48.49	49.06	50.00	49.32	37.23	47.44	48.30	49.97	48.75

Gray code all

	150 RO pairs					450 RO pairs				
positions	6–8	7–8	7–9	8–9	7–10	6–8	7–8	7–9	8–9	7–10
w [-]	3	2	3	2	4	3	2	3	2	4
HD_{intra} [%]	0.60	0.71	1.04	1.33	1.69	0.47	0.60	0.95	1.22	1.55
HD_{inter} [%]	34.49	40.70	43.87	49.34	45.43	31.97	40.22	43.49	49.00	45.14

Table 7.2: The results of statistical tests performed on the PUF outputs composed of various bit selections for 150 a 450 RO pairs (1000 and 500 measurements respectively) measured on 24 Digilent Basys 2 FPGA boards.

H_{inter}, H_{intra} is high since position 2) while with the increasing position, the stability is decreasing. The bias is very close to the ideal value of 0.5 on all positions apart from the first few. The positions are numbered from the most significant bit (MSB), i.e. position 1 corresponds to the MSB and position 16 corresponds to the least significant bit (LSB).

For the PUF we need to select such positions of the counter values that have the values of stability, entropy, and bias close to the ideal values. For example positions 7–9 present suitable bits. When using these bit positions to form the PUF outputs we can expect the PUF responses to have the desired properties of high reliability and uniqueness, as it will be shown later.

PUF response evaluation

From Table 7.1 we were able to determine the suitable bit positions that could be used for PUF responses. Now we will evaluate the PUF responses that are formed from various combinations of bit positions. Again, we performed 1000 measurements on 150 RO pairs and 500 measurements for 450 RO pairs. As mentioned in Section 5.1, the number of RO pairs together with the number of selected bits (w) affect the final length of the PUF response.

The next Table 7.2 presents the results for the PUF that uses different selections of

positions. We used the same data we measured before and assembled PUF responses from it. PUF responses created in this fashion were $150w$ or $450w$ bits long, where w is the number of bits selected from each RO pair. For description of the parameters HD_{intra} and HD_{inter}, see Chapter 4. Since one of our goals was to achieve HD_{inter} close to 50%, it was desirable to select bits starting at position 7 that also have a high entropy close to 1. However, since we also need the PUF responses to be stable enough, we have to select bits in which the HD_{intra} is very small, ideally 0%. Bit positions that fulfill both these requirements are for example positions 7–9.

However, as described in Section 5.2.3, we can use the Gray code to increase the stability of the PUF responses. In Table 7.2, we present 3 approaches of Gray code usage. First, no Gray code is used and we evaluate the PUF responses composed of raw selected bit positions. Secondly, we apply the Gray code only to selected parts of the counter values (e.g. Gray code was applied only to bits 7–9). Lastly, the Gray code was applied to the whole counter values. From these we then extracted the bits for PUF responses.

Iit is obvious that the Gray code significantly increases the stability of the PUF responses. Without the Gray code, we could use e.g. positions 7–9 to create PUF responses with good properties. However, when the Gray code is applied, we can extract up to 4 bits (positions 7–10) with an even slightly better stability.

As discussed in Section 5.2.3, when the Gray code is applied to the whole counter values, it can negatively affect the resulting uniqueness (HD_{inter}) of the PUF responses. Results from Table 7.2 confirm that, as the values of HD_{inter} decreased significantly from the ideal 50%.

In Section 4.2, we mentioned that long bit strings are required to evaluate randomness. Since we had limited possibilities, we applied particular tests from the NIST STS 2.1.2 that are suitable for evaluating short input sequences. We evaluated the randomness of 10 sequences that were 360 bits long (or its multiple, depending on the number of bits selected from each counter value) and the minimum pass rate for these tests was 8/10. But for the the distribution of p-values test, at least 55 input sequences are required in order to obtain statistically meaningful results according to [52]. Therefore, the results from these tests have to be taken with caution.

To allow more reliable p-values uniformity evaluation, we performed additional evaluation of 60 sequences that were at least 100 bits long for selections of multiple bits from the counter values. In this case, the minimum pass rate reported by the NIST STS for individual tests was 57/60.

The tested sequences were created by concatenating the reference (mean) responses (R_ref_n) obtained from all of the 24 devices. We obtained sequences for each bit position that were 3600 bits long. We also evaluated various selections of bit positions either with or without the application of the Gray code. When the Gray code is used, it is applied only to selected bit positions, not to the whole counter values.

Table 7.3 [A.5] contains the results of the applied tests. Empty cells mean that the pass rate for that test is 10/10 (60/60) and red colored cells indicate that the test failed for the distribution of p-values. The upper part of this table shows the results for PUF outputs made of each position of the counter values. It can be seen that for positions 1 to

5 the tests indicate that the input sequences made from these positions are not random. This result is expected since these positions are close to the MSB and therefore they have a low entropy and the PUF outputs made from these bits would have low HD_{inter}. The positions in the direction facing the LSB have better results but still with some failures in the distribution of p-values test. As mentioned before, this can be caused due to the small number of input sequences.

The bottom part of Table 7.3 presents the results for multiple extracted counter value bits for various position selections. Table 7.3 is divided into two parts, where the left side contains results for the parts of the counter values in binary code, while the other side contains results for the parts of the counter values encoded with the Gray code. It can be seen that the Gray code does not have a negative impact on the randomness of the PUF outputs.

We point out that the results presented in Table 7.3 are slightly different from what was published in [A.5]. First, we chose a different value of the significance level α used for the evaluation of the p-value distribution. In [A.5], $\alpha = 0.1$ and was therefore too strict. In this work, we chose $\alpha = 0.001$ on the recommendation of [42]. Moreover, we changed the block lengths m in the Block Frequency and Approximate Entropy tests in order to be in compliance with the NIST STS specification.

Just as in the case of HD_{intra} and HD_{inter} evaluation, PUF responses made of positions 7–10 with the Gray code applied to these bits exhibit a promising behaviour in terms of randomness. However, all of the measurements were performed at stable environmental conditions so far. Varying operating conditions can decrease the stability of the PUF responses. Therefore, we also need to evaluate the PUF responses with varying supply voltage and temperature in order to finally determine suitable positions for the PUF.

positions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frequency	0/10	0/10													9/10	9/10
BlockFrequency (m=20)	0/10					9/10								9/10		9/10
BlockFrequency (m=30)	0/10	2/10											9/10	9/10		9/10
CumulativeSums	0/10	0/10														9/10
CumulativeSums	0/10	0/10													9/10	9/10
Runs	0/10	0/10							9/10			9/10	9/10			
LongestRun	0/10	0/10							9/10							
ApproximateEntropy (m=2)	0/10	0/10		1/10												9/10
ApproximateEntropy (m=3)	0/10	0/10		3/10	8/10	9/10										9/10

	Binary code					Gray code (10 sequences)						Gray code (60 sequences)					
positions	6–7	7–8	8–9	6–8	7–9	6–7	7–8	8–9	6–8	7–9	7–10	6–7	7–8	8–9	6–8	7–9	7–10
Frequency						9/10				9/10		59/60					59/60
BlockFrequency (m=20)													59/60			59/60	59/60
BlockFrequency (m=30)														59/60		59/60	59/60
CumulativeSums						9/10							59/60				59/60
CumulativeSums						9/10											58/60
Runs	9/10			8/10											59/60	59/60	
LongestRun												0/60	0/60	0/60			
ApproximateEntropy (m=2)	9/10			8/10												59/60	
ApproximateEntropy (m=3)	9/10			8/10									58/60	59/60		58/60	

Table 7.3: The results of the tests from NIST STS. Each cell contains the pass rate and indicates whether the test failed for the distribution of p-values (red cells). Empty cells mean that the pass rate for those cells was 10/10 (or 60/60). The minimum allowed pass rate was either 8/10 for 10 input sequences or 57/60 for 60 input sequences. The top table contains results for PUF outputs made from each position of the counter values. The bottom table presents results for various counter value position selections in binary and Gray code (the Gray code is applied only to selected bits).

	50 RO pairs				150 RO pairs			
position(i)	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$
1	1.0000	0.0000	0.0000	1.0000	1.0000	0.0000	0.0000	1.0000
2	1.0000	0.0000	0.0000	1.0000	1.0000	0.0000	0.0000	1.0000
3	0.9995	0.6911	0.0726	0.8143	0.9992	0.6527	0.0730	0.8319
4	0.9985	0.8667	0.1347	0.7107	0.9987	0.8623	0.1357	0.7143
5	0.9975	0.9953	0.2652	0.5197	0.9971	0.9746	0.3320	0.5897
6	0.9913	0.9926	0.6779	0.5364	0.9936	0.9850	0.7524	0.5655
7	0.9842	0.9610	0.8659	0.6016	0.9866	0.9890	0.9413	0.5376
8	0.9729	0.9865	0.9067	0.5162	0.9710	0.9951	0.9648	0.5213
9	0.9434	0.9759	0.9313	0.5524	0.9415	0.9950	0.9718	0.5111
10	0.8801	0.9940	0.9093	0.5054	0.8895	0.9951	0.9657	0.5032
11	0.7869	0.9926	0.9338	0.4918	0.7839	0.9941	0.9662	0.5206
12	0.6292	0.9833	0.9297	0.5078	0.6214	0.9956	0.9655	0.4997
13	0.5210	0.9912	0.9022	0.5023	0.5186	0.9936	0.9706	0.5005
14	0.5136	0.9922	0.9133	0.5007	0.5131	0.9946	0.9679	0.5001
15	0.5130	0.9787	0.9296	0.4987	0.5129	0.9941	0.9661	0.4993
16	0.5134	0.9789	0.9001	0.4977	0.5130	0.9840	0.9588	0.4971

Table 7.4: Statistical evaluation of 16-bit counter values for symmetric ROs. The first table contains a statistical evaluation for 50 RO pairs measured 1000 times on 10 Digilent Basys 2 FPGA boards. In the second table, the results of statistical evaluation for 150 RO pairs measured 1000 times on 24 of the same FPGA boards are presented.

7.1.2 Symmetric ROs

As described in Section 6.1.1, we placed the gates of each RO as shown in Fig. 6.1 [A.5, A.6]. We used the same RLOC constraints for the placement of all of the 5-staged ROs and put each RO into four slices with each inverter occupying one slice and the last one sharing it with the NAND gate. The placement of the ROs themselves and also the interconnects between the gates were left unconstrained. The circuit used for measurements remained the same (see Fig. 5.5), but the number of ROs was smaller due to the RLOC constraints that were used. There were 100 ROs divided into two groups of 50 ROs each.

Selection of suitable positions

The left part of Table 7.4 shows the results of the statistical evaluation of each bit position of the counter values for 50 RO pairs measured 1000 times on 10 Digilent Basys 2 FPGA boards. It can be seen that the results are very similar to what was presented in Table 7.1. Only the values of H_{inter} are slightly lower. This is caused by the smaller number of boards that were used for these measurements.

The right part of Table 7.4 contains the statistical evaluation of each counter value bit position for 150 RO pairs measured 1000 times on 24 Digilent Basys 2 FPGA boards. As we can see, the results are also very similar to Table 7.1. Also the suitable bit positions for the PUF are the same (the positions starting from position 7).

No Gray code

	50 RO pairs					150 RO pairs				
positions	6–8	7–8	7–9	8–9	7–10	6–8	7–8	7–9	8–9	7–10
w [-]	3	2	3	2	4	3	2	3	2	4
HD_{intra} [%]	1.72	2.14	3.31	4.18	5.48	1.63	2.12	3.36	4.37	5.28
HD_{inter} [%]	43.44	47.64	48.58	49.62	48.74	44.80	48.91	49.33	49.93	49.43

Gray code part

	50 RO pairs					150 RO pairs				
positions	6–8	7–8	7–9	8–9	7–10	6–8	7–8	7–9	8–9	7–10
w [-]	3	2	3	2	4	3	2	3	2	4
HD_{intra} [%]	0.93	1.39	1.96	2.90	3.14	0.97	1.45	1.98	2.96	2.91
HD_{inter} [%]	43.01	48.51	49.53	50.18	49.93	43.05	48.81	49.23	49.88	49.45

Gray code all

	50 RO pairs					150 RO pairs				
positions	6–8	7–8	7–9	8–9	7–10	6–8	7–8	7–9	8–9	7–10
w [-]	3	2	3	2	4	3	2	3	2	4
HD_{intra} [%]	0.84	0.96	1.67	2.14	2.93	0.87	1.13	1.77	2.30	2.75
HD_{inter} [%]	40.06	47.00	48.52	51.04	49.18	39.16	46.28	47.54	49.78	48.19

Table 7.5: The results of statistical tests performed on the PUF outputs composed of various bit selections for 50 and 150 RO pairs (1000 measurements) measured on 10 and 24 Digilent Basys 2 FPGA boards respectively.

PUF response evaluation

From the same set of measurements we statistically evaluated the PUF outputs composed of various bit position selections. Table 7.5 contains the results of this evaluation. We provide this evaluation with the Gray code applied either to the selected part of the counter values or to the whole counter values. We can see a similar behaviour of the metrics HD_{intra} and HD_{inter} to that in Table 7.2. This was described in Section 5.2.3. When the Gray code is applied to the whole counter values, the HD_{intra} is lower, but the HD_{inter} is lower as well. However, compared to results presented in Table 7.2 for positions 7–9, the difference of these two cases of Gray code usage is not so significant in case of symmetric ROs. The difference in HD_{inter} for asymmetric ROs is 5.19% (from 49.06% to 43.87%) while for symmetric ROs the difference is about 1–2% (49.53% to 48.52% and 49.22% to 47.54%).

7.1.3 Timing analysis

For further investigation of the RO behaviour we performed measurements using an oscilloscope. We sampled the signal coming out of the ROs in the measured RO pairs with the oscilloscope and then processed the obtained waveforms in software. To determine the impact of the delay of the circuit that detects the overflow and stops the counting on the

RO pair	1	2	3	4	5
Mean difference	2812.43	3502.78	3085.41	4037.65	2861.42
Standard Deviation	0.5366	0.9054	0.7797	0.9987	0.7808

Table 7.6: The difference of measured counter values and the correct ones for five RO pairs.

resulting counter value, we performed such measurements in which the chosen RO pairs were enabled and from this moment these ROs were sampled. We read the counter values from the FPGA and processed the waveforms obtained from the oscilloscope. By counting the rising edges in the waveforms for two ROs in a pair we determined the correct value that should be in the counter.

Due to variation in voltage and other reasons, the frequency of a selected RO may exceed the maximum operating frequency of the respective counter. This should be avoided by design, but if it happens, the counter starts missing some clock pulses and this results in incorrect counter values read from the FPGA. We measured several instances where the reported counter values differ from the exact values calculated by counting clock edges measured with oscilloscope. However, even in these instances the counter values are consistent in time and the statistics for the PUF outputs remain the same. Table 7.6 [A.5] shows the mean and standard deviations of counter value differences from the correct ones for five RO pairs where at least one RO frequency exceeded the maximum operating frequency of the counters. The standard deviations of differences are very small, indicating that the counter values remain consistent when the measurements are repeated. The measurements were carried out 100 times for each RO pair.

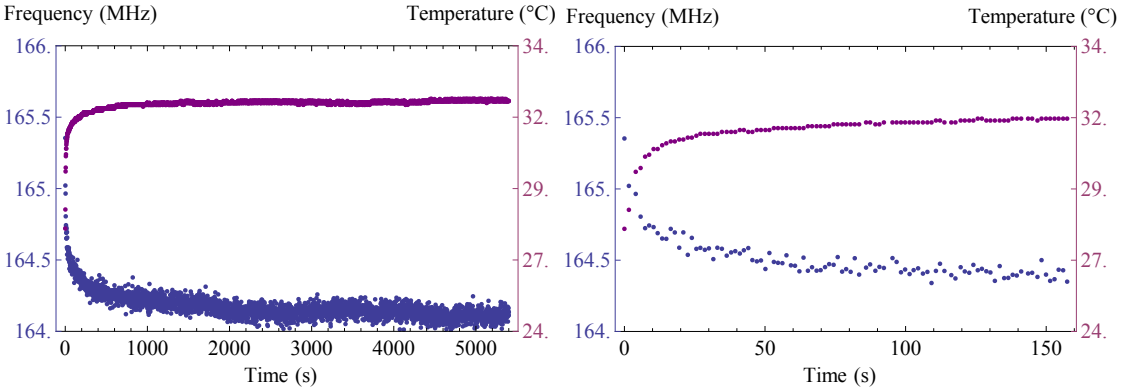
When using proper (fast enough) counters, the difference between counter values and the correct ones should ideally be 0. The difference we measured was 0 or 1. When the difference is 1, it is caused by an overflow detection logic of one of the counters – before the other counter is stopped, it manages to perform one additional count. This way we can verify that the stopping circuit works correctly and consistently.

7.1.4 Influence of supply voltage

All of the previous measurements were performed at normal environmental conditions with stable temperature and supply voltage. In this subsection, we present the results of experiments performed at varying voltage [A.5]. First, we will describe the influence of temperature conditions on the measurements to evaluate the correctness of our approach. Then we will present out experiment results and point out the difference of the results obtained for both asymmetric and symmetric ROs.

Temperature conditions during measurements

All measurements regarding the influence of voltage were performed at stable environmental conditions with a stable room temperature of $24.5 \pm 1\,^{\circ}\mathrm{C}$. Before the measurements



(a) The frequency progress for approximately 90 minutes.

(b) A more detailed frequency progress at the beginning of warm-up of FPGA.

Figure 7.1: Frequency behaviour during FPGA warm-up (Xilinx Spartan-3E). Blue dots represent the measured frequency of the RO. The temperature at the time of frequency measurement is highlighted in purple.

were started, the FPGAs were turned on with running oscillators for at least 2 minutes to eliminate other influences than voltage change. The reason for this is the fact that in the short time period after enabling the ring oscillators, there is a significant change in temperature of the FPGA and this could affect the measurement.

To evaluate the correctness of our approach, we observed the RO behaviour on three different FPGAs during their warm-up. The duration of the measurement was 30 minutes at the minimum and 90 minutes maximum. For this measurement we selected one RO pair common for all three FPGAs and then two other random RO pairs for each FPGA (three RO pairs for each FPGA in total). The temperature values, RO frequency, and counter value were sampled in intervals of 1.5 to 3 seconds. We analysed the results and we observed that the maximum difference of the gathered counter values during the whole time interval was such that it influences positions 9–16 of the counter value. For the time after 2 minutes (the waiting time that we used for the measurements of supply voltage influence), the maximum difference affects positions 10–16. When extending the delay to 30 minutes, it still affects positions 10–16 of the counter value. These results indicate that the influence of change in temperature during the warm-up of the FPGAs at a stable room temperature is negligible compared to the influence of voltage.

Fig. 7.1 shows the frequency behaviour during FPGA warm-up for one RO on one FPGA. The graphs contains both the frequency of the RO and the temperature that was recorded at the time of the measurement. Fig. 7.1(a) shows the behaviour of one RO on one FPGA for the whole recorded time. It can be seen that the largest change in both temperature and frequency occurs at the beginning of the measurement, which is shown in better detail in Fig. 7.1(b).

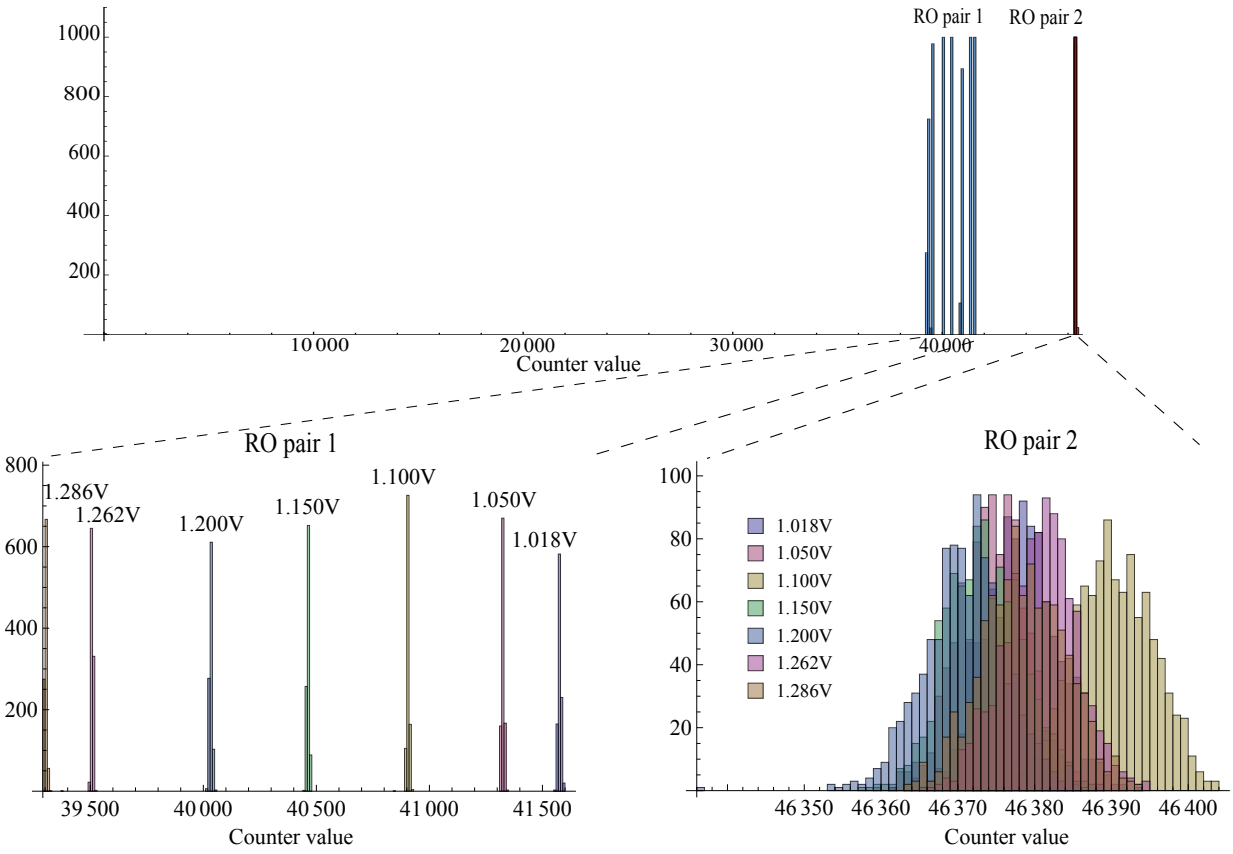


Figure 7.2: Histograms for two behaviour examples of measured counter values for two RO pairs at seven different voltage levels (range from 1.018V to 1.286V). The histogram for RO pair 1 shows the undesired behaviour of counter values while the histogram for RO pair 2 shows the desired behaviour because there the frequency distribution of counter values is approximately the same for all voltages.

Asymmetric ROs

In the previous subsection we described the temperature conditions during the measurements performed in our experiment. The next measurement concerns the influence of voltage on the behaviour of the proposed ROPUF design. The measurements were performed on two Digilent Basys 2 FPGA boards (Spartan-3E). The main power supply for the FPGA's internal logic is V_{ccint} and its nominal voltage is 1.2V. The maximum ratings for V_{ccint} are -0.5V and 1.32V, with manufacturer's recommended range from 1.14V to 1.26V. The circuit remained the same and the results presented in Table 7.7 relate to 1000 measurements for 150 RO pairs; they show how the PUF outputs are different from those obtained at a nominal voltage, which is 1.2V. The range of tested voltages is from 1.018V to 1.286V and the selected positions of counter values for the PUF outputs are 7–8 and 7–10. The Gray code is applied to the whole counter values.

It can be seen that the influence of voltage on this ROPUF design is significant since the

		7–8	7–10
Voltage [V]	ΔU [mV]	HD_{intra} [%]	HD_{intra} [%]
1.018	-182	53.56	51.55
1.050	-150	51.65	48.05
1.100	-100	41.89	47.33
1.150	-50	23.67	36.45
1.200	**0**	**0.55**	**2.00**
1.262	62	34.19	42.01
1.286	86	38.66	43.49

Table 7.7: The difference of the PUF outputs measured at various voltages and the PUF outputs measured at the nominal voltage of 1.2V for 150 RO pairs and for selected positions 7–8 and 7–10. The Gray code is applied to the whole counter values.

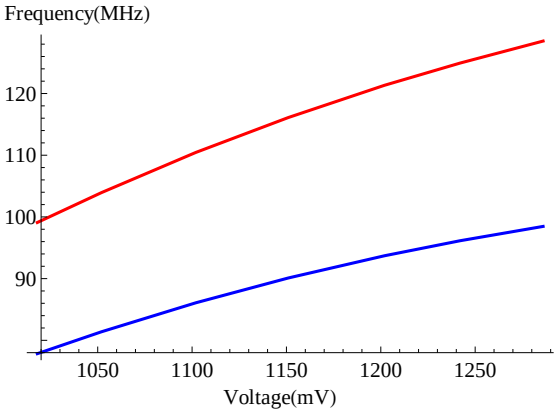
	Voltage [V]	ΔU [mV]	HD_{intra} [%]
5-stage ROs	1.101	-103	47.57
	1.289	85	53.28
7-stage ROs	1.101	-103	48.51
	1.289	85	50.76

Table 7.8: The difference of the PUF outputs at various voltages and the PUF outputs measured at the nominal voltage of 1.2V for selected positions 7–8 without the Gray code on 5-stage and 7-stage ROs.

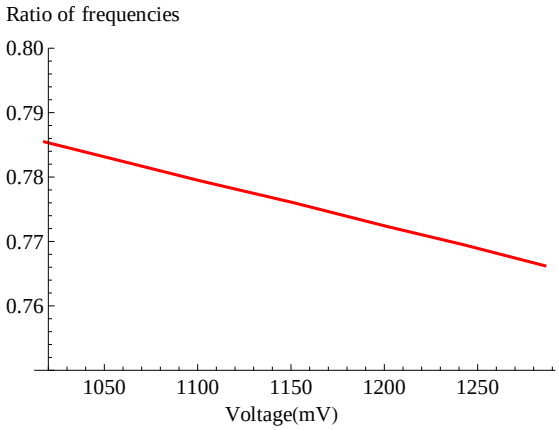
PUF outputs are very different. The influence of voltage is even larger for positions 7–10 since these include the positions closer to the LSB, therefore, the stability is lower than for positions 7–8. The cause of this behaviour can be seen in Fig. 7.2. These histograms show how the counter values for two chosen RO pairs change with voltage. The upper histogram contains the counter values for both of the RO pairs and the bottom histograms show the counter values for each RO pair in detail. We can see that the mean values of the counter values for RO pair 1 are very different for each voltage. Ideally, we would need the behaviour of another RO pair (RO pair 2) where the counter values would not significantly change with varying voltage.

The next experiment was to determine whether the length of the ROs influences the behaviour of the PUF when the voltage is changed. We performed measurements 1000 times for 90 RO pairs where ROs were five and seven stages long and the selected positions of counter values were 7–8. Table 7.8 shows the results of these measurements and as it can be seen, the change of the length did not affect the PUF behaviour.

The high sensitivity of this ROPUF design to voltage is caused by the change of ratios of two frequencies of the ROs in each pair. If we want the PUF outputs to remain stable at varying voltage, the ratios of the frequencies for each RO pair have to be the same. The



(a) Difference of the change of frequency for two paired ROs. The red curve represents the faster RO and the blue curve represents the slower RO.



(b) The change of the frequency ratio for one RO pair depending on voltage.

Figure 7.3: Dependency of the frequencies of two ROs and their ratio on the change of voltage.

Voltage [V]	ΔU [mV]	HD_{intra} [%]
1.180	-21	14.02
1.190	-11	7.04
1.193	-8	4.89
1.196	-5	3.236
1.201	**0**	**1.40**
1.207	6	3.85
1.212	11	7.18
1.222	21	14.30

Table 7.9: The PUF outputs with various voltages from the range of 1.180V to 1.222V compared to the PUF outputs measured at nominal voltage of 1.2V for selected positions 7–8 with the Gray code applied to the whole counter values.

reason is that the value of the 16-bit counter can be determined as shown in Eq. 5.1:

$$Counter\ value = \frac{f_2}{f_1} 2^{16},$$

where f_1 is the frequency of the faster RO and f_2 is the frequency of the slower RO. Fig. 7.3(a) shows the dependency of the frequencies of two ROs in a pair on voltage. The higher the voltage is, the higher the frequency of the oscillator. The change of ratio for one RO pair (the same pair as in Fig. 7.3(a)) is shown on the next Fig. 7.3(b). Ideally, the ratio should be constant, however, the ratio is changing with the change of voltage.

		Positions 7–8		Positions 7–10	
		FPGA 1	FPGA 2	FPGA 1	FPGA 2
Voltage [V]	ΔU [mV]	HD_{intra} [%]	HD_{intra} [%]	HD_{intra} [%]	HD_{intra} [%]
1.022	-180	21.01	26.97	36.10	38.80
1.102	-100	8.13	12.89	25.45	28.16
1.150	-52	2.49	7.55	15.76	19.86
1.180	-22	1.39	6.32	8.72	9.95
1.192	-10	0.58	4.13	5.94	9.28
1.202	**0**	**0.79**	**0.91**	**1.69**	**2.58**
1.210	8	2.31	4.21	5.20	7.39
1.222	20	4.80	4.30	9.51	8.13
1.242	40	6.34	8.26	14.79	16.14
1.261	69	7.79	12.15	21.20	24.09
1.287	85	12.34	14.95	26.91	30.82

Table 7.10: The PUF outputs with various voltages from the range of 1.022V to 1.287V compared to the PUF outputs measured at the nominal voltage of 1.2V and for selected positions 7–8 and 7–10 for two FPGAs. The Gray code is applied to the whole counter values, symmetric ROs are used.

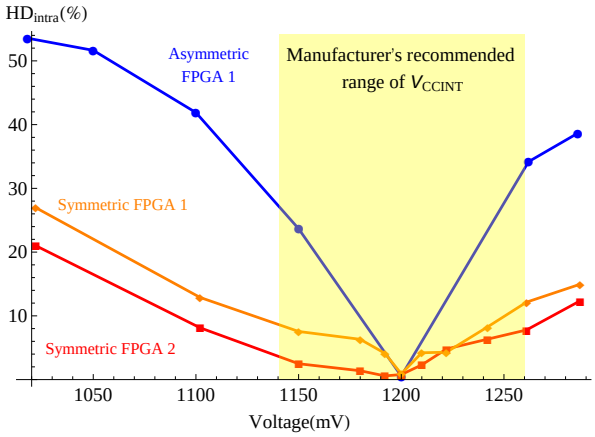
In order to determine the interval of the voltage at which the PUF might be operating, we performed more measurements. This time the voltage ranged from 1.180V to 1.222V and we used 150 RO pairs and positions 7–8. The results are presented in Table 7.9. The nominal voltage is 1.2V. If we want HD_{intra} to be about 5% on each side of the nominal voltage, then the interval for the voltage might be approximately from 1.195V to 1.205V. Under these circumstances the correct behaviour of the PUF should be guaranteed.

7.1.5 Symmetric ROs

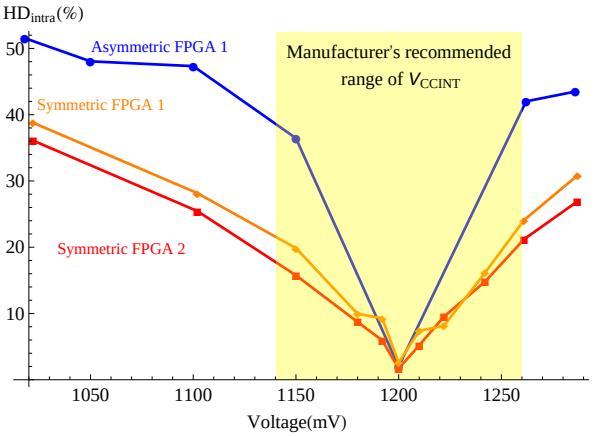
In the previous subsection we presented the results of the measurements for the proposed ROPUF design at varying voltage. It was shown that the influence of supply voltage is significant because the ratios of the paired RO frequencies change when the voltage varies. These results were obtained for the ROPUF design where the ROs were not mutually symmetric and there was no emphasis on the placement of the logic gates of each RO.

We can expect that the frequencies of ROs will change in a similar way when the ROs are mutually symmetric. Therefore, the ratios of the RO pair frequencies should be almost constant. In the next experiment, we placed the logic gates of each RO so that the ROs were mutually symmetric.

Table 7.10 shows the results of evaluated measurements for various voltages ranging from 1.022V to 1.287V on two FPGAs. These measurements were performed 1000 times for 50 RO pairs where all ROs had the same mutual placement of logic gates and the



(a) Dependence of HD_{intra} on voltage for positions 7–8.



(b) Dependence of HD_{intra} on voltage for positions 7–10.

Figure 7.4: Comparison of the behaviour of the proposed PUF when using mutually symmetric and asymmetric ROs for positions 7–8 and 7–10. The reference output for calculating HD_{intra} is the mean output from the PUF outputs measured at the nominal voltage of 1.2V. The yellow area represents the manufacturer's recommended range of FPGA's main power supply voltage V_{ccint}, which is from 1.14V to 1.26V.

Gray code was applied to the whole counter values. It can be seen that the improvement is significant compared to the results presented in Table 7.7 where the ROs were not mutually symmetric. Fig 7.4(a) and Fig. 7.4(b) present the comparison of the behaviour of the proposed ROPUF when using mutually symmetric and asymmetric ROs for positions 7–8 and 7–10. The results for HD_{intra} are not ideal, but they demonstrate the improvement when using symmetric ROs compared to asymmetric ROs and show the way for further investigation of the influence of the RO placement could have on the stability of the PUF outputs.

7.1.6 Influence of temperature

This section examines the influence of changes in temperature on the proposed ROPUF [A.6]. Statistical properties of a PUF using both symmetric and asymmetric ROs will be compared. Fig. 7.5 depicts our measurement setup. For the purpose of our experiment, we performed measurements at different temperatures. For these measurements, the FPGA was preheated to a preset temperature (e.g. 40°C) with ROs enabled. Each of the measurements was carried out when the temperature measured on the package of the FPGA stabilised at the given value. We used three Digilent Basys 2 FPGA boards for this experiment.

The values of HD_{intra} were obtained by averaging the PUF outputs at each temperature. The average PUF output is obtained as the majority of each column when the PUF outputs are written in the form of a matrix where each row represents one PUF output.

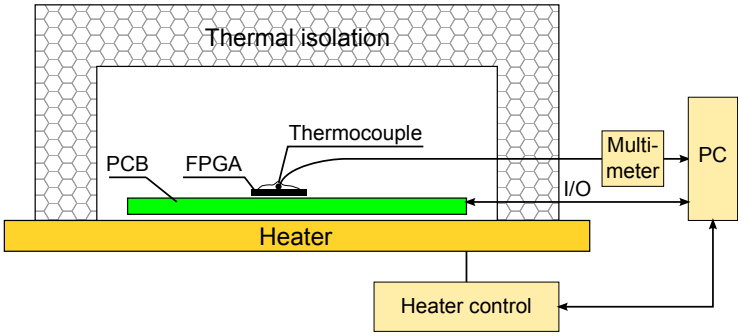


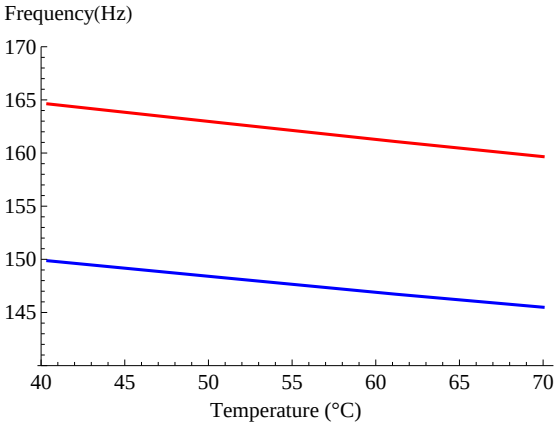
Figure 7.5: Measurement setup for measuring at elevated temperatures. PCB is the Digilent Basys 2 prototyping board. The FPGA is the Xilinx Spartan3E-100 CP132. A heater was used to preheat the FPGA to target temperature.

Asymmetric ROs					
FPGA 1		FPGA 2		FPGA 3	
Temperature [°C]	HD_{intra} [%]	Temperature [°C]	HD_{intra} [%]	Temperature [°C]	HD_{intra} [%]
36.7 → 41.2	2.75	38.4 → 42.3	2.58	37.7 → 41.8	2.49
36.7 → 51.8	7.58	38.4 → 50.1	6.69	37.7 → 50.9	5.42
36.7 → 60.4	9.58	38.4 → 60.3	9.21	37.7 → 61.3	8.20
36.7 → 71.1	11.56	38.4 → 69.9	12.68	37.7 → 70.1	12.19
Symmetric ROs					
FPGA 1		FPGA 2		FPGA 3	
Temperature [°C]	HD_{intra} [%]	Temperature [°C]	HD_{intra} [%]	Temperature [°C]	HD_{intra} [%]
33.0 → 42.4	2.80	34.4 → 40.9	2.20	34.5 → 41.1	3.647
33.0 → 50.5	3.75	34.4 → 50.5	3.32	34.5 → 51.4	6.03
33.0 → 60.6	4.24	34.4 → 60.8	4.73	34.5 → 60.6	6.84
33.0 → 71.0	5.32	34.4 → 70.2	5.81	34.5 → 70.4	7.74

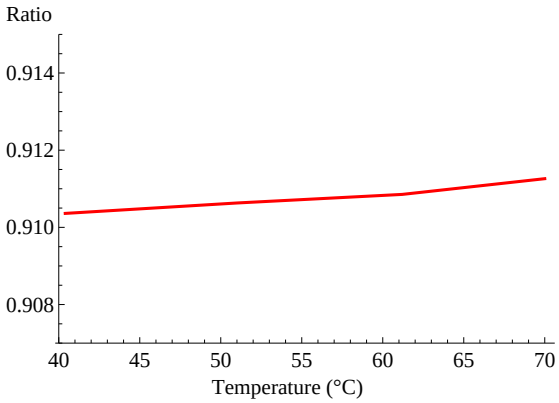
Table 7.11: Evaluation of HD_{intra} for 150 asymmetric/symmetric RO pairs and selected positions 7–8 at different temperatures. The Gray code is applied to the whole counter values.

Table 7.11 displays the values of HD_{intra} on three FPGAs for asymmetric and symmetric ROs. As in the previous Section 7.1.4, the RO symmetry consists of the mutual gate placement of each RO but not of the interconnects between them. The column "temperature" presents the temperatures at which the PUF outputs were compared. The values of HD_{intra} for symmetric and asymmetric ROs are almost equivalent when the differences in temperature were small, but for larger changes in the temperature there is a visible improvement of the PUF behaviour when symmetric ROs are used. This is a very similar result to that which was presented in Section 7.1.4 where the stability of the PUF outputs was also increased using symmetric ROs.

It was shown before that the resulting counter value depends on the ratio of the frequen-



(a) Difference in the frequency change for two paired ROs. The red curve represents the faster RO and the blue curve represents the slower RO.



(b) The change of the frequency ratio for one RO pair in dependence on temperature.

Figure 7.6: Dependency of the frequencies of two ROs and their ratio on the temperature change.

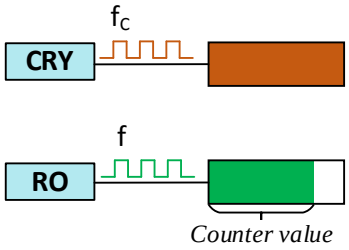
cies of the two ROs that are influenced by various physical conditions. However, it can be expected that the frequencies are affected in a similar way. Therefore, the frequency ratios should minimize the possibility of the temperature having an influence on the measured counter values. Fig. 7.6(a) shows the change of the frequencies of two ROs with increasing temperature. The higher the temperature, the smaller the RO frequencies.

Ideally, we need the frequency ratios to remain constant in time, but as it is shown in Fig. 7.6(b), the ratio is not constant at varying temperatures. Therefore, we need to minimize the difference in the frequency ratio at varying temperatures (and other influences) and using symmetric ROs may present one of the possible solutions. By using symmetric ROs the frequencies of the ROs get closer to each other, hence they should more likely be influenced in the same way when physical conditions change. However, it is still necessary to further investigate the relationship between PUF stability at varying environmental conditions and the symmetry of the ROs.

7.1.7 Comparison of the three measurement methods

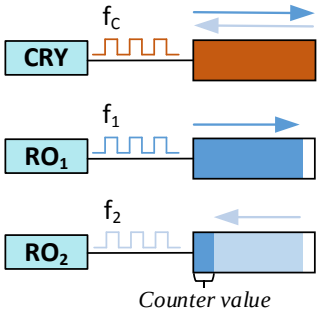
In this subsection we compare three different methods of RO usage in order to generate the PUF outputs [A.6] as presented in Section 5.4. We will show that the method we proposed is the most stable one when physical conditions, specifically the supply voltage and temperature, change.

Since the resulting counter value in our proposal is determined by the ratio of the frequencies of the two ROs in a pair, this method can be considered a differential measurement because even though the physical conditions such as supply voltage and temperature have



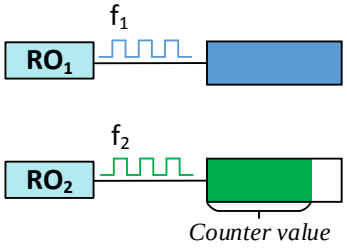
$$\text{Counter value} = \varphi\left(\frac{f}{f_C}\right) \times 2^{16}$$

(a) The basic approach, where the counter value is calculated for each RO for the same amount of time given by a crystal oscillator (CRY) with a stable frequency f_C that serves as the reference clock. It is assumed that the frequency of the crystal oscillator is greater than the frequency of any RO.



$$\text{Counter value} = \varphi\left((f_1 - f_2) \times \frac{1}{f_C}\right) \times 2^{16}$$

(b) The subtraction method. First, the value of the counter is incremented for every rising edge of the first RO (RO_1). Then the counter value is decremented for every rising edge of the second RO (RO_2). The incrementing and decrementing runs for the same amount of time given by a crystal oscillator (CRY) with a stable frequency f_C. The assumption is that the frequency of the crystal oscillator is greater than the frequency of any RO and that $f_1 > f_2$.



$$\text{Counter value} = \varphi\left(\frac{f_2}{f_1}\right) \times 2^{16}$$

(c) The frequency ratio method that is used in our PUF design. The resulting counter value is given by the ratio of the frequencies f_1 and f_2. There is no reference clock as opposed to the previous cases. It is assumed that $f_1 > f_2$

Figure 7.7: Three different methods of using the ROs for the PUF.

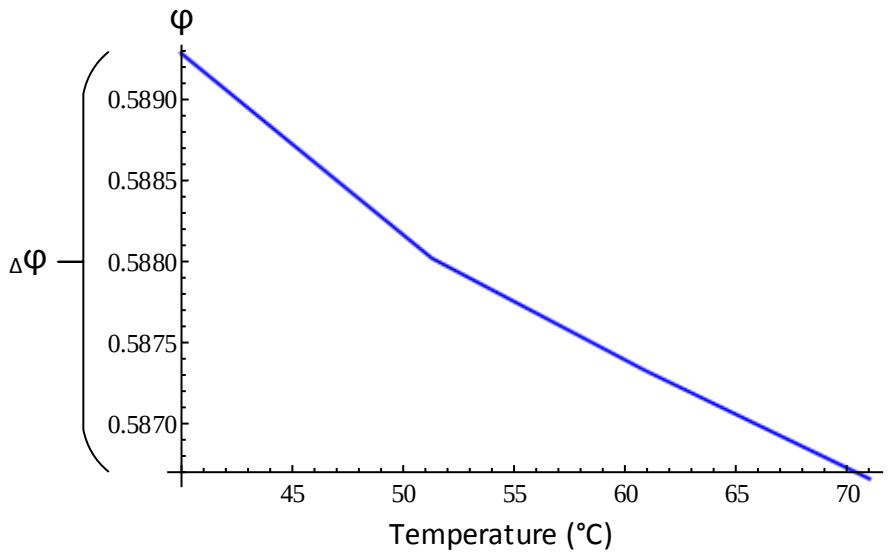


Figure 7.8: Behaviour of φ at varying temperature. In this example, $\varphi = \frac{f_2}{f_1}$, therefore, φ represents ratio of the frequencies of two ROs.

a significant influence on the RO frequencies, the change in the ratio of the frequencies is much smaller. To show the difference compared to some of the other possible approaches, Fig. 7.7 presents three different methods for RO usage in order to obtain a counter value that could be later processed. The first two approaches require a stable reference clock such as the crystal oscillator. There is a formula for each method to determine the resulting counter value and the part of the equation that is dependent on physical conditions is highlighted in red circle. We will denote this part as φ.

The first method is similar to our approach, but instead of two ROs forming a pair there is always one stable reference clock (crystal oscillator) and one RO. In the case shown in Fig. 7.7(a) it is assumed that the frequency of the reference clock is greater than the frequency of any RO and the resulting counter value is then calculated from the ratio of the RO frequency and the reference clock. However, because we assume the reference clock to be stable at varying temperatures, φ only represents the frequency of the RO ($\varphi = f$), since the resulting counter value depends solely on the frequency of the RO.

In addition, if we have M ROs, we can extract only $M \times w$ (w is the number of bits extracted from each counter value) bits for the PUF response. Therefore, even though this design would be very simple, it would not be very efficient since it requires a lot of ROs in order to generate long PUF outputs.

The second method shown in Fig. 7.7(b) also uses a stable reference clock and it is assumed that its frequency is greater than the frequency of any RO. At first, the counter is incrementing with the frequency given by the first RO. Then the value in the counter is decremented by the second RO and both the incrementation and decrementation take

	Temperature			Supply voltage
	FPGA 1	FPGA 2	FPGA 3	FPGA 1
	$\frac{\Delta\varphi}{\varphi_{max}}$[%]	$\frac{\Delta\varphi}{\varphi_{max}}$[%]	$\frac{\Delta\varphi}{\varphi_{max}}$[%]	$\frac{\Delta\varphi}{\varphi_{max}}$[%]
crystal	3.42	3.01	2.97	22.18
subtraction	4.08	4.29	3.78	33.21
ratio	0.27	0.16	0.14	2.41

Table 7.12: The change of φ at varying temperatures and voltages for the three different approaches. The frequencies were measured for 300 ROs. In case of temperature, the measurements were performed on three FPGAs at temperatures ranging from 40°C to 71°C. For supply voltage, the measurements were performed on one FPGA and the range of the supply voltage was from 1.018V to 1.286V.

the same time given by the reference clock. In this case, $\varphi = f_1 - f_2$ because the resulting counter value is derived from the difference of the frequencies.

From the perspective of using the ROs in pairs, this method is similar to the method we proposed. The theoretical maximum number of bits that can be extracted using this method is $\binom{M}{2} \times w$ just like in our proposal.

The last method is shown in Fig. 7.7(c) and this is the method used in our PUF design described in Chapter 5. As it was explained before, this method is based on the ratio of the frequencies of two ROs in a pair, hence $\varphi = \frac{f_2}{f_1}$. The maximum number of bits for PUF is $\binom{M}{2} \times w$ as was the case with the previous method based on subtraction.

To evaluate the resistance of each method to varying physical conditions, we measured the frequencies of 300 ROs at different physical conditions. In case of varying temperature, we performed the measurements on three FPGAs, while the measurements concerning the change of voltage were performed on only one FPGA. Depending on the method, we calculated the values of φ from the frequencies at different physical conditions and determined the value of $\Delta\varphi$ as $\Delta\varphi = \varphi_{max} - \varphi_{min}$. See Fig. 7.8 for an example of determining $\Delta\varphi$ for varying temperatures. From $\frac{\Delta\varphi}{\varphi_{max}}$ we can observe the resistance of each method to the change of physical conditions.

The results of this evaluation for all presented methods are shown in Table 7.12. It can be seen that the change of φ at varying temperatures and voltages is the lowest for the method used in our PUF design (ratio of the frequencies). We can also notice that the change of φ is considerably larger for varying voltage than for varying temperature. This may indicate that the RO frequencies are more dependent on supply voltage than on temperature.

7.1.8 TRNG evaluation

In this subsection we present the results of testing the proposed design as a TRNG [A.1, A.2]. Even though designing a TRNG was not one of the goals of this thesis, it is still

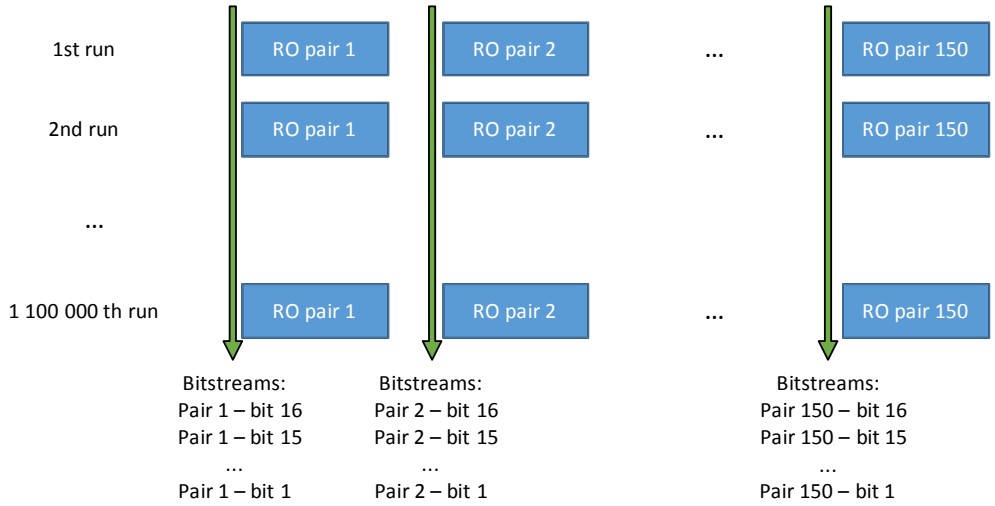


Figure 7.9: An example random sequence forming from single bits of individual RO pairs.

a notable result, since the proposed ROPUF design can be used as a TRNG without any significant changes, thus allowing us to utilize the same circuit to provide us the functionality of the two cryptographic primitives. Therefore, for completeness, we present the results of the experiments related to TRNG in this subsection.

There were two separate sets of measurements. The first set was performed on a circuit containing 150 RO pairs, where all of the ROs were running during the measurement. An on-board switching regulator was used as the power supply. The measurement setup is shown in Fig. 6.2(a). In the second set we used a circuit with 130 RO pairs where the ring oscillators ran and were measured separately. In this case, we modified the Digilent Basys 2 FPGA board. The original power supply was disconnected and replaced with a new power supply consisting of a battery and a number of linear regulators (see Fig. 6.2(b)).

The measured sequences of bits were evaluated using the NIST statistical test suite [52] that was proposed specifically to test random number generators for cryptographic purposes. The version of the NIST software we used is STS 2.1.2. This test suite consists of tests such as the frequency test, the runs test, the cumulative sums test, the entropy test, etc.

Individual RO pairs tests

In this experiment, we examined single bits from each RO pair and we considered each RO pair a unique source of entropy. Each RO pair's counter value was measured 1 100 000 times. Therefore, we obtained 150×16 bitstreams as shown in Fig. 7.9. Some of the tests in NIST STS 2.1.2 require longer bitstreams than we could provide, which led us to exclude them.

The results of the tests were similar for all RO pairs. However, some RO pairs did not show satisfactory results - they generated only a few unique values during the experiment, which resulted in the tests failure. Such RO pairs were excluded from the tests. The results

Bit	16		15		14		13		12	
	Median	Average	Median	Average	Median	Average	Median	Average	Median	Average
Frequency	97/100	96,7/100	99/100	98,6/100	99/100	98,8/100	98/100	89,9/100	7/100	8,1/100
Block Frequency	99/100	99,3/100	99/100	99,2/100	99/100	99,2/100	97/100	78,6/100	0/100	0,0/100
Cumulative Sums I	97/100	97,0/100	99/100	98,6/100	99/100	98,8/100	98/100	87,7/100	0/100	0,8/100
Cumulative Sums II	97/100	96,8/100	99/100	98,7/100	99/100	98,8/100	98/100	87,9/100	0/100	0,9/100
Runs	99/100	98,9/100	99/100	99,0/100	99/100	99,1/100	97/100	82,2/100	0/100	0,1/100
Longest Run	99/100	99,0/100	99/100	98,9/100	99/100	98,9/100	98/100	92,9/100	0/100	1,8/100
Approximate Entropy	99/100	98,7/100	99/100	98,7/100	99/100	98,7/100	98/100	89,8/100	0/100	0,2/100

Table 7.13: Results of NIST STS tests for individual RO pairs. The tests were run for 150 bitstreams and the average and median values are presented. The minimum allowed pass rate is 96/100.

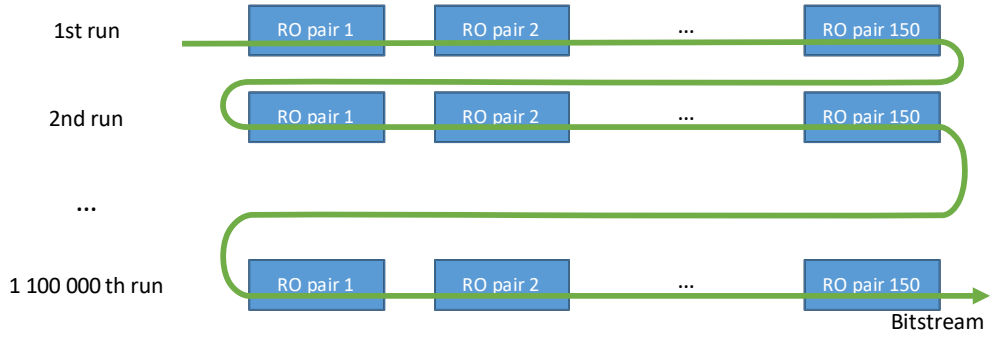


Figure 7.10: Concatenating the outputs from all RO pairs to form one long random sequence of bits.

of the tests are shown in Table 7.13. Bit 16 (LSB) failed in some tests, but the positions 15 and 14 passed all of the tests indicating that these bits may be suitable to be used for the TRNG output. All of the other positions failed all of the statistical tests except for bit 13 which only failed some tests.

Concatenated RO pairs outputs tests

The previous experiment indicated that each RO pair could be used as a stand alone source of entropy. However, it would be more natural for this particular design to use multiple bits from each measured counter value for the TRNG output as it is done in the case of PUF. Moreover, since there are 150 RO pairs, we can concatenate their outputs to form a single long bitstream as shown in Fig. 7.10.

After this concatenation, we had enough data to run all tests from the NIST STS. We tested both concatenated single bits from all RO pairs and concatenated multiple bits from all RO pairs. Table 7.14 presents the results of these tests. The pass rate was 96/100 (except for the Random Excursions and the Random Excursions Variants tests) and if the test failed for the distribution of p-values (see [52]) the cells are highlighted in red. The tests failed for the Frequency and the Cumulative Sums tests, indicating bias in the

	Individual bits				Concatenated bits	
Position	16	15	14	13	13–16	13–15
Frequency	0/100	24/100	82/100	69/100	19/100	63/100
Block Frequency	99/100	98/100	99/100	98/100	100/100	98/100
Cumulative Sums I	0/100	31/100	83/100	69/100	23/100	64/100
Cumulative Sums II	0/100	25/100	87/100	70/100	21/100	67/100
Runs	0/100	83/100	95/100	94/100	69/100	91/100
Longest Run	99/100	99/100	98/100	100/100	99/100	99/100
Rank	100/100	99/100	99/100	98/100	99/100	99/100
FFT	97/100	96/100	99/100	100 /100	94/100	98/100
Non Overlapping template	85–100/100	96–100/100	96–100 /100	96–100 /100	94–100 /100	94 -100 /100
Overlapping Template	96/100	97/100	97/100	99/100	99/100	100 /100
Universal	97/100	98/100	97/100	98/100	100 /100	98/100
Approximate Entropy	89/100	100 /100	98/100	100 /100	98/100	99/100
Random Excursions	4/4	18/18	51–52 /52	34–35 /35	16–17/17	43–44/44
Random Excursions Variants	4/4	17–18 /18	51–52 /52	34–35 /35	17/17	44/44
Serial I	99/100	99/100	99/100	99/100	99/100	100 /100
Serial II	99/100	100 /100	97/100	100 /100	99/100	99/100
Linear Complexity	100 /100	99/100	98/100	99/100	100 /100	97/100

Table 7.14: Results of NIST STS tests for a concatenated output of all RO pairs. The output of RO pairs was concatenated for both single bits (one bit from each RO pair) and multiple bits (multiple bits from each RO pair). The minimum allowed pass rate was 96/100. The red cells indicate that the test failed for the distribution of p-values.

generated bitstreams. This can happen when dealing with TRNGs. If we assume that the bias is the only problem and the generated bits are statistically independent, post processing can be used to deal with this problem [10].

Post-processing

We used 2 post-processing methods to de-bias the generated bitstreams: the XOR corrector and the von Neumann corrector.

The von Neumann corrector works as follows: If the input is “00” or “11”, the input is discarded. If the input is “10”, the output is “1”, and finally if the input is “01”, the output is “0”. The disadvantage of this post-processing method is the fact that it shortens the generated sequence by approximately 75%.

The XOR corrector shortens the generated sequence only by 50%. It takes two subsequent bits from the input and puts the result of the XOR operation on these two bits in the generated sequence.

The results of the NIST STS after applying von Neumann and XOR correctors are shown in Table 7.15. This table shows the pass rates for each of the tests and the red cells show that the test failed for the distribution of p-values. As can be seen in Table 7.15, the bits 13–15 show excellent behaviour after application of these post-processing methods.

	Concatenated bits			
	Von Neumann corrector		XOR corrector	
Position	13–16	13–15	13–16	13–15
Frequency	69/100	99/100	82/100	98/100
Block Frequency	100/100	99/100	99/100	100/100
Cumulative Sums I	69/100	100/100	85/100	99/100
Cumulative Sums II	71/100	99/100	83/100	97/100
Runs	96/100	99/100	96/100	100/100
Longest Run	98/100	99/100	99/100	99/100
Rank	99/100	98/100	99/100	99/100
FFT	99/100	99/100	100/100	98/100
Non Overlapping Template	98–100/100	97–100/100	98–100/100	97–100/100
Overlapping Template	99/100	100/100	99/100	100/100
Universal	98/100	100/100	99/100	100/100
Approximate Entropy	100/100	100/100	98/100	100/100
Random Excursions	30–31/30	62–63/63	43–44/100	55/55
Random Excursions Variants	30–31/30	62–63/63	43–44/100	55/55
Serial I	99/100	99/100	100/100	100/100
Serial II	100/100	100/100	98/100	98/100
Linear Complexity	100/100	100/100	98/100	97/100

Table 7.15: Results of NIST STS tests of concatenated bitstream after applying the von Neumann and the XOR correctors. Minimum allowed pass rate was 96/100. The red cells indicate that the test failed for the distribution of p-values.

Bit	16		15		14		13	
	Median	Average	Median	Average	Median	Average	Median	Average
Frequency	99/100	89,7/100	99/100	89,7/100	99/100	89,7/100	58/100	58,3/100
Block Frequency	99/100	90,6/100	99/100	90,1/100	99/100	90,1/100	0/100	0,7/100
Cumulative Sums I	99/100	89,8/100	99/100	89,7/100	99/100	89,7/100	40/100	43,6/100
Cumulative Sums II	99/100	89,7/100	99/100	89,7/100	99/100	89,7/100	38/100	43,5/100
Runs	99/100	89,8/100	99/100	89,8/100	98/100	82/100	0/100	1/100
Longest Run	99/100	89,9/100	99/100	89,9/100	99/100	89,8/100	0/100	0,2/100
Approximate Entropy	99/100	89,9/100	99/100	89,9/100	98/100	87,9/100	0/100	0,1/100

Table 7.16: Results of NIST STS tests for individual RO pairs when linear regulators were used as the power supply. The tests were run for 130 bitstreams and the average and median values are presented. The minimum allowed pass rate was 96/100.

Ruling out crosstalk and parasitic frequencies

To minimize any potential crosstalks between individual RO pairs and parasitic frequencies caused by switching regulators that influence randomness of generated bitstream and to verify that each RO pair can be considered a unique source of entropy, we tested individual RO pairs separately using a set of linear regulators as the power supply. In this experiment, all ROs were not running simultaneously as before, only one selected RO pair was running

Test	Concatenated bits 14–16		
	Without post-process	XOR corrector	Von Neumann corrector
Frequency	53/100	99/100	99/100
Block Frequency	85/100	99/100	99/100
Cumulative Sums I	55/100	98/100	99/100
Cumulative Sums II	52/100	98/100	99/100
Runs	81/100	98/100	94/100
Longest Run	97/100	99/100	100/100
Rank	99/100	98/100	99/100
FFT	98/1000	100/100	100/100
Non Overlapping template	85–100/100	95–100/100	97–100/100
Overlapping Template	91/100	99/100	99/100
Universal	98/1000	100/100	100/100
Approximate Entropy	94/100	98/100	95/100
Random Excursions	29–30/30	12/12	7–8/8
Random Excursions Variants	29–30/30	13/13	8/8
Serial I	94/100	100/100	99/100
Serial II	100/100	100/100	99/100
Linear Complexity	100/100	98/100	99/100

Table 7.17: Results of NIST STS tests of concatenated bitstreams with and without post-processing. Linear regulators were used as the power supply. The minimum allowed pass rate was 96/100. Red cells indicate that the test failed for the distribution of p-values.

during the measurement. Otherwise, the setup of the experiment remains the same. Each RO pair was measured 1 100 000 times and there were 130 RO pairs on the examined circuit. The lower number of RO pairs is caused by the additional logic needed to allow each RO pair to be run separately. The Digilent Basys 2 prototyping board was modified. The original power supply circuit was disconnected and a new power supply was installed that consists of a battery and a number of linear regulators as shown in Fig. 6.2(b).

At first, individual bits from each RO pair were examined to confirm that every RO pair is a unique source of entropy. We tested non post-processed bitstreams. We selected single bits from each RO pair output and we obtained 130×16 bitstreams. The results are shown in Table 7.16 [A.2]. As it can be seen, the results are similar to those presented in Table 7.13 when switching regulators were used and all ROs were running simultaneously. Again, some ROs had to be excluded from the test since they generated only a few unique values during the experiment.

We then tested concatenated single bits from multiple RO pairs and concatenated multiple bits from multiple RO pairs and processed them with XOR and von Neumann correctors. The results for concatenated bits 14–16 are shown in Table 7.17. The results of this experiment are very similar to those where all ROs were running simultaneously and

	150 RO pairs				450 RO pairs			
position(i)	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$
1	0.9991	0.0948	0.0043	0.9887	1.0000	0.0000	0.0000	1.0000
2	0.9989	0.0948	0.0043	0.0115	0.9997	0.0000	0.0000	0.0003
3	0.9993	0.5163	0.0478	0.8838	0.9991	0.5202	0.0525	0.8824
4	0.9984	0.9514	0.1833	0.6273	0.9982	0.9798	0.1759	0.5821
5	0.9978	0.9534	0.3999	0.6156	0.9971	0.9353	0.3793	0.6457
6	0.9956	0.9931	0.6932	0.5399	0.9945	0.9993	0.6946	0.5070
7	0.9922	0.9885	0.8597	0.5495	0.9898	0.9956	0.8485	0.5256
8	0.9802	0.9963	0.8648	0.5208	0.9804	0.9913	0.8580	0.5259
9	0.9557	0.9927	0.8720	0.5115	0.9625	0.9988	0.8648	0.4944
10	0.9107	0.9963	0.8720	0.5201	0.9235	0.9987	0.8687	0.5094
11	0.8283	0.9970	0.8762	0.5039	0.8448	0.9992	0.8671	0.5028
12	0.6774	0.9926	0.8788	0.5007	0.6895	0.9973	0.8756	0.5045
13	0.5257	0.9865	0.8726	0.5036	0.5322	0.9916	0.8695	0.5037
14	0.5150	0.8914	0.8293	0.5070	0.5195	0.9151	0.8069	0.5080
15	0.5143	0.8949	0.7868	0.5073	0.5190	0.9093	0.7926	0.5083
16	0.5323	0.3581	0.2609	0.5310	0.5320	0.5622	0.4600	0.5279

Table 7.18: Evaluation of individual bit positions of 16-bit counter values for 150 and 450 RO pairs (1000 and 500 measurements respectively) obtained from 6 Digilent Nexys 3 FPGA boards.

the switching regulator was used. Therefore, we can assume that each individual RO pair is a unique source of entropy.

7.2 Spartan-6

In this section, we will present the results for the measurements performed on Digilent Nexys 3 FPGA boards [14] (Xilinx Spartan-6 XC6LX16-CS324 [66]). The design remains the same as for the previous experimental platform, i.e. see Fig. 5.5. Again, the circuit contains 300 ROs divided into two sets of 150 ROs. The ROs are all 5-staged and asymmetric, meaning that there are no constrains on the placements of the gates and the routes between them.

The following subsections present the statistical evaluation of individual bit positions of the counter values and for PUF responses made of various selections of bit positions. The measurements were performed for 150 RO pairs (1000 measurements) and 450 RO pairs (500 measurements).

Evaluation of positions of the counter values

Table 7.18 contains the evaluation of individual bit positions of the counter values measured on 6 Digilent Nexys 3 FPGA boards. The results are similar to those presented in Table 7.1 for Digilent Basys 2 FPGA boards. However, only the values of H_{inter} are lower due to the

No Gray code

	150 RO pairs					450 RO pairs				
positions	6–8	7–8	7–9	8–9	7–10	6–8	7–8	7–9	8–9	7–10
HD_{intra} [%]	1.07	1.38	2.40	3.21	4.03	1.17	1.49	2.24	2.85	3.59
HD_{inter} [%]	46.43	49.89	50.12	50.31	50.18	45.91	49.30	49.57	49.81	49.74

Gray code part

	150 RO pairs					450 RO pairs				
positions	6–8	7–8	7–9	8–9	7–10	6–8	7–8	7–9	8–9	7–10
HD_{intra} [%]	0.71	1.01	1.60	2.37	2.33	0.69	1.02	1.31	1.93	1.97
HD_{inter} [%]	44.67	49.89	49.75	49.76	49.94	43.39	49.36	49.30	49.35	49.54

Gray code all

	150 RO pairs					450 RO pairs				
positions	6–8	7–8	7–9	8–9	7–10	6–8	7–8	7–9	8–9	7–10
HD_{intra} [%]	0.65	0.85	1.49	2.01	2.25	0.61	0.76	1.14	1.46	1.84
HD_{inter} [%]	39.36	47.24	47.99	49.76	48.62	37.48	45.52	46.74	49.41	47.62

Table 7.19: The results of statistical tests performed on the PUF outputs composed of various bit selections for 150 a 450 RO pairs (1000 and 500 measurements respectively) measured on 6 Digilent Nexys 3 FPGA boards.

small number of experimental boards as it was in the case of Basys 2 FPGA boards using symmetric ROs in Table 7.4.

PUF response evaluation

The evaluation of the PUF responses using different selections of bit positions is shown in Table 7.19. The results indicate that the proposed ROPUF design can be used on Spartan-6 FPGAs as well as on the Spartan-3E, since the results are fairly similar to those presented in Table 7.2 for Spartan-3E. Again, we evaluate the PUF responses with and without the application of the Gray code. As was the case in the previously presented results, the Gray code applied to the whole counter values negatively affects the value of HD_{inter}.

7.3 Spartan-7

The results presented in this section were all measured on Digilent Cmod S7 FPGA boards [15] containing the Xilinx Spartan-7 XC7S25-1CSGA225C [67]. We performed the measurements at both stable and varying environmental conditions. All of the three presented measurement methods (frequency ratio, frequency difference, and crystal reference) described in Section 5.4 were investigated and compared. In all cases, 5500 measurements were carried out for each RO pair (or a single RO in the case of the crystal reference

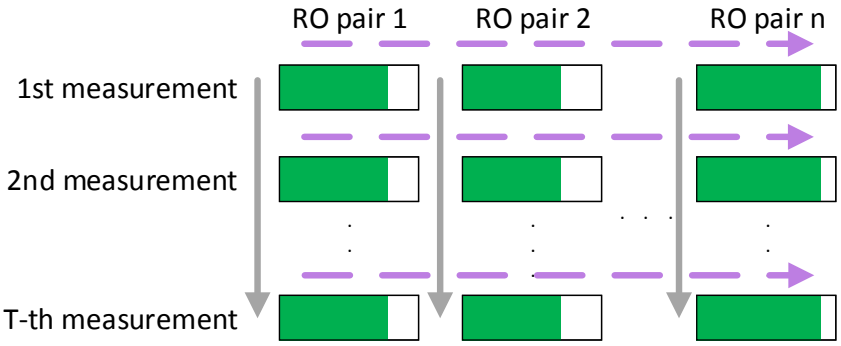


Figure 7.11: Two different measurement approaches. Gray arrows represent the non-interleaved measurement, purple arrows indicate the interleaved measurement.

measurement method) with the first 500 measurements being skipped for evaluation purposes as they were used as a warm-up for the ROs. The number of measured RO pairs was 150, with 300 ROs being measured in the crystal reference measurement method. For measurements at stable environmental conditions, we used 20 FPGA boards.

The design that was used for these measurements with the description of the implementation is depicted in Fig. 6.3 in Section 6.2. This design is common for all of the three measurement methods. Moreover, we have 3 implementation variants of this design:

- Asymmetric ROs
- Asymmetric ROs, all enabled
- Symmetric ROs

All of the three implementation variants were used for the three measurement methods. As it was already shown in Section 7.1.4 and Section 7.1.6, the mutual symmetry of the ROs influences the stability of the resulting PUF responses. Therefore, we performed additional extensive measurements in order to compare these variants at varying supply voltage and temperature. For this purpose, we modified 5 Digilent Cmod S7 FPGA boards because we needed to be able to set the value of their supply voltage. These modified FPGA boards were then put into a climate chamber and measured in a preset temperature level.

We then compared two possible measurement approaches that we called *non-interleaved* and *interleaved*. In the case of the non-interleaved measurement, one RO pair (or a separate RO for the crystal reference) was selected and then measured T-times. After measuring one RO pair, we proceed with the next one until all RO pairs were measured. In case of interleaved measurement, all RO pairs were measured sequentially before the next repetition of the measurement could continue. Both of these approaches are depicted in Fig. 7.11.

In the following two subsections, we present the results of the selection of suitable position and PUF response evaluation at stable environmental conditions.

	non-interleaved				interleaved			
position(i)	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$
1	**0.9998**	0.0029	0.0019	0.9995	**0.9997**	0.0029	0.0019	0.9994
2	**0.9998**	0.5048	0.0115	0.8882	**0.9997**	0.5048	0.0115	0.8881
3	**0.9992**	0.8909	0.0700	0.6915	**0.9991**	0.8901	0.0694	0.6918
4	**0.9988**	0.9515	0.1874	0.6273	**0.9984**	0.9520	0.1869	0.6269
5	**0.9972**	0.9970	0.4191	0.5064	**0.9968**	0.9971	0.4169	0.5062
6	**0.9925**	0.9793	0.8277	0.5759	**0.9917**	0.9777	0.8249	0.5774
7	**0.9841**	0.9958	0.9577	0.5193	**0.9795**	0.9959	0.9577	0.5182
8	**0.9654**	0.9943	0.9614	0.4950	**0.9586**	0.9936	0.9608	0.5015
9	**0.9265**	0.9941	0.9653	0.5094	**0.9151**	0.9937	0.9646	0.5056
10	**0.8519**	0.9977	0.9631	0.4996	**0.8307**	0.9974	0.9674	0.4936
11	**0.7057**	0.9936	0.9648	0.4999	**0.6739**	0.9949	0.9576	0.4999
12	0.5374	0.9946	0.9683	0.5006	0.5225	0.9980	0.9602	0.4997
13	0.5065	0.9943	0.9653	0.5000	0.5065	0.9949	0.9609	0.4999
14	0.5061	0.9935	0.9564	0.5005	0.5062	0.9940	0.9609	0.5001
15	0.5059	0.9965	0.9673	0.4997	0.5061	0.9947	0.9608	0.4997
16	0.5058	0.9963	0.9616	0.4998	0.5059	0.9965	0.9654	0.5000

Table 7.20: Comparison of the results for non-interleaved and interleaved measurements shown on evaluation of individual bit positions of the counter values measured on 20 Digilent Cmod S7 FPGA boards for 150 RO pairs, 5000 measurements. The frequency ratio was used to measure the counter values.

7.3.1 Selection of suitable positions

As discussed before, we show the difference of performing non-interleaved and interleaved measurements in order to select the one that is more suitable for our purposes. Table 7.20 contains the results of this comparison which is shown on the evaluation of individual bit positions of the measured counter values. There were 150 RO pairs measured using the frequency ratio measurement method on 20 Digilent Cmod S7 FPGA boards with 5000 measurements done for each RO pair.

As the results in Table 7.20 indicate, the stability in case of non-interleaved measurements was slighty higher that for the interleaved measurements. When performing interleaved measurements, the ROs were repeatedly disabled until it was their turn to be running, causing the ROs to warm up during the accumulation of their oscillations in the measurement. Due to this warm-up during the oscillation accumulation, the interleaved measurement has a lower stability than the non-interleaved measurement, where all repetitions of the measurement for each RO pair were executed immediately after each other. To confirm that using interleaved measurements would result in a higher HD_{intra}, we will show the comparison of non-interleaved and interleaved measurements in the next subsection. Nevertheless, we will only present the results of non-interleaved measurements in the following text.

	Frequency ratio				Frequency difference				Crystal reference			
position(i)	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$
1	0.9998	0.0029	0.0019	0.9995	0.9994	0.9641	0.0977	0.3897	0.9998	0.1040	0.0424	0.9833
2	0.9998	0.5048	0.0115	0.8882	0.9992	0.9808	0.1098	0.4197	0.9991	0.9402	0.5420	0.3717
3	0.9992	0.8909	0.0700	0.6915	0.9987	0.9963	0.1486	0.4678	0.9990	0.9882	0.5788	0.4547
4	0.9988	0.9515	0.1874	0.6273	0.9980	0.9962	0.2373	0.5321	0.9984	0.9951	0.7314	0.4767
5	0.9972	0.9970	0.4191	0.5064	0.9955	0.9947	0.4785	0.4698	0.9966	0.9891	0.9113	0.5098
6	0.9925	0.9793	0.8277	0.5759	0.9897	0.9972	0.8320	0.5010	0.9940	0.9912	0.9426	0.5147
7	**0.9841**	**0.9958**	**0.9577**	**0.5193**	**0.9812**	**0.9929**	**0.9563**	**0.4991**	**0.9881**	**0.9952**	**0.9641**	**0.4992**
8	**0.9654**	**0.9943**	**0.9614**	**0.4950**	**0.9618**	**0.9954**	**0.9605**	**0.4906**	**0.9744**	**0.9953**	**0.9660**	**0.4985**
9	**0.9265**	**0.9941**	**0.9653**	**0.5094**	**0.9258**	**0.9939**	**0.9577**	**0.5039**	**0.9453**	**0.9943**	**0.9644**	**0.4876**
10	0.8519	0.9977	0.9631	0.4996	0.8547	0.9948	0.9619	0.4943	0.8896	0.9962	0.9656	0.4969
11	0.7057	0.9936	0.9648	0.4999	0.7132	0.9941	0.9668	0.4969	0.7752	0.9924	0.9696	0.4981
12	0.5374	0.9946	0.9683	0.5006	0.5380	0.9961	0.9575	0.5002	0.5938	0.9964	0.9669	0.4982
13	0.5065	0.9943	0.9653	0.5000	0.5056	0.9927	0.9696	0.5000	0.5086	0.9936	0.9626	0.4999
14	0.5061	0.9935	0.9564	0.5005	0.5057	0.9957	0.9689	0.5000	0.5056	0.9961	0.9657	0.5000
15	0.5059	0.9965	0.9673	0.4997	0.5056	0.9960	0.9609	0.5002	0.5055	0.9962	0.9613	0.4997
16	0.5058	0.9963	0.9616	0.4998	0.5057	0.9978	0.9521	0.4997	0.5055	0.9973	0.9631	0.5000

Table 7.21: Evaluation of individual bit positions of the counter values measured on 20 Digilent Cmod S7 FPGA boards for 150 RO pairs, 5000 measurements. All three measurement methods were used to measure the counter values. The ROs were mutually asymmetric.

Table 7.21 contains the results of the evaluation of individual bit positions of the measured counter values using the three measurement methods (frequency ratio, frequency difference, crystal reference) and asymmetric ROs. First, we notice that the positions that provide satisfactory properties in terms of their stability ($s(b_i)$), entropy ($H_{intra}(b_i)$, $H_{inter}(b_i)$) and bias ($P(b_i = 1)$) are practically the same as in the case of the evaluation on Spartan-3E and Spartan-6 FPGA boards.

Moreover, the positions that are suitable to be used for the PUF are also similar for all three measurement methods. Only the crystal reference measurement method reaches a slightly higher stability than the other two. We should note that these measurements were performed at stable environmental conditions and they were the first step to selecting of the appropriate part of the counter values for the PUF. Therefore, even though some positions seem to be stable enough to be used for a PUF, they might not achieve similarly good results when the device is stressed at varying environmental conditions.

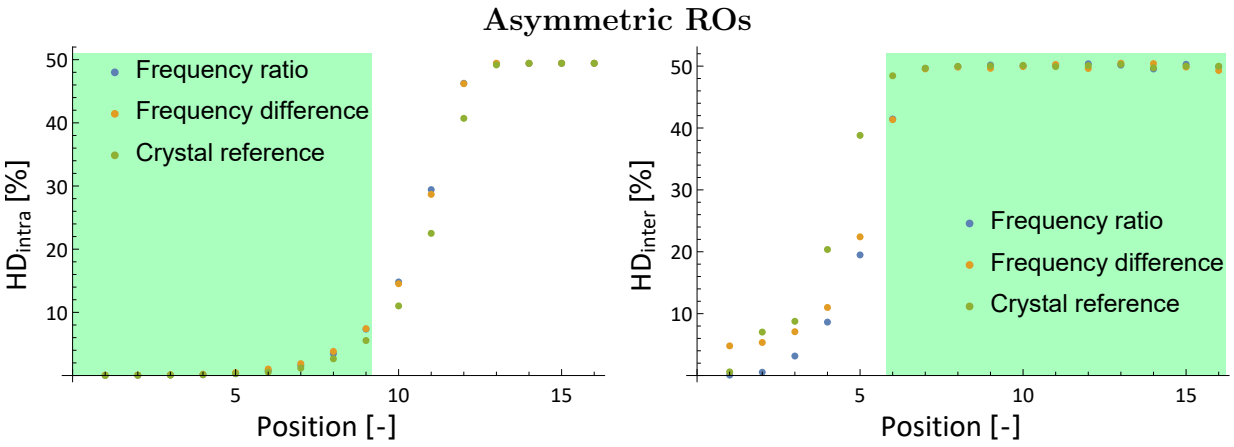
We can observe that the frequency ratio and the frequency difference have comparable results both in terms of their stability, entropy, and bias. However, the crystal reference seems to provide more stable PUF outputs that the other two measurement methods. This may be caused by the instability of ROs, e.g. the jitter. In case of frequency ratio and frequency difference, the resulting counter value depends on both of the ROs that were used in a pair in the measurement. Therefore, the instability of both of the ROs affects the result and more of it is accumulated in the measured counter values. This manifests itself in a higher variance of the measured counter values and thus on a higher number of errors in the individual bit positions of the counter values (higher $s(b_i)$).

In contrast, the crystal reference only needs a single RO in each measurement of the counter values. Therefore, less instability is accumulated in the resulting counter values and therefore, the counter values obtained using this measurement method were more stable.

In Table 7.21, we presented the results only for one implementation variant of the design, which is the design with asymmetric ROs where only selected ROs were running and all of the others were disabled. Since the results were very similar to the other two implementation variants (asymmetric ROs that were all enabled and symmetric ROs), we did not present them here in this chapter, but we moved the results to Appendix A. See Table A.2 and Table A.3.

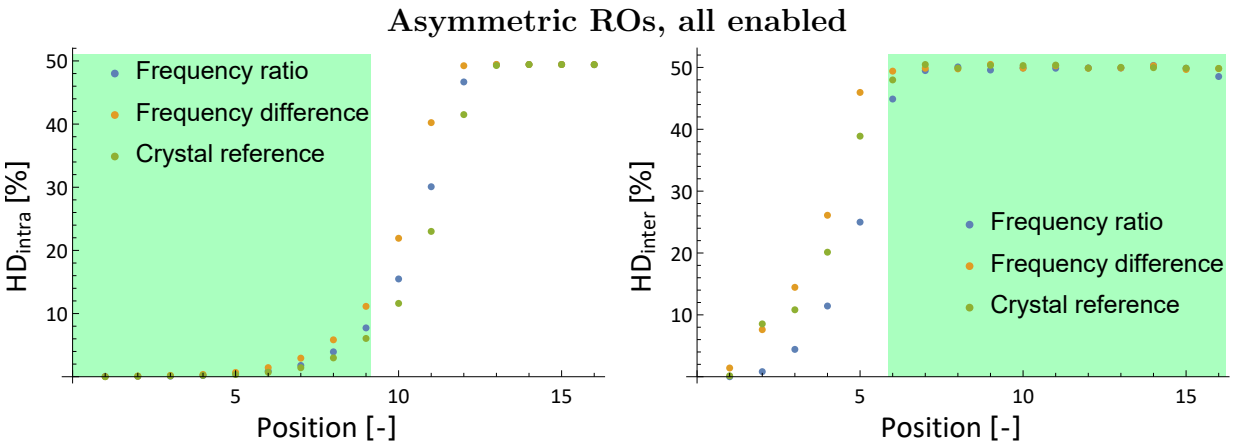
Fig. 7.12 shows the evaluation of the PUF responses composed of individual bit positions from the counter values using the realiability (HD_{intra}) and uniqueness (HD_{inter}) parameters. We used each bit from the counter values separately to see how the quality of the PUF responses changes with the position of the counter values that is used to form the output. This evaluation is performed for all three measurement methods using the three implementation variants. The graphs positioned on the left side show the results of the reliability parameter (HD_{intra}), the graphs on the right side present the results of the uniqueness evaluation (HD_{inter}). The green coloured area in each graph represents the positions that achieved satisfactory results for the corresponding evaluation parameter.

In case of the HD_{intra} (see Fig. 7.12(a), Fig. 7.12(c) and Fig. 7.12(e)), there were no significant differences between the measurement methods that were used. Only the



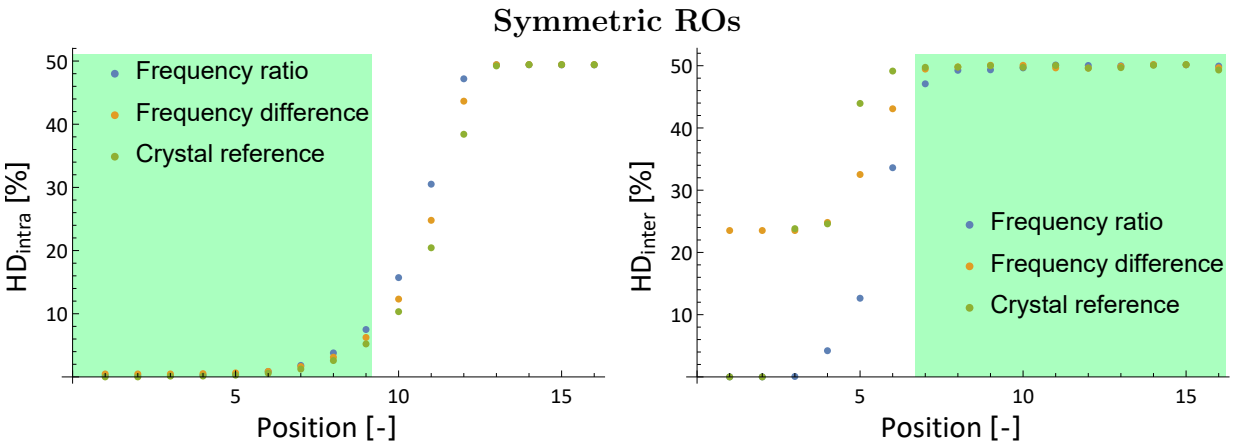
(a) Asymmetric ROs, HD_{intra}.

(b) Asymmetric ROs, HD_{inter}.



(c) Asymmetric ROs, all enabled, HD_{intra}.

(d) Asymmetric ROs, all enabled, HD_{inter}.



(e) Symmetric ROs, HD_{intra}.

(f) Symmetric ROs, HD_{inter}.

Figure 7.12: Evaluation of the PUF responses composed of individual bit positions from the counter values measured on 20 Digilent Cmod S7 FPGA boards for 150 RO pairs, 5000 measurements. All three measurement methods were compared together with three different implementation variants.

values of the HD_{intra} were slightly higher when all enabled asymmetric ROs were used. Nevertheless, positions 1–9 exhibit satisfactory results in terms of stability of the PUF responses.

However, the HD_{inter} values (see Fig. 7.12(b), Fig. 7.12(d) and Fig. 7.12(f)) are high enough approximately from position 7. There are some exceptions, such as the crystal reference and the frequency difference measurement methods that provide high values of the HD_{inter} from positions 5 or 6. It can also be noticed in Fig. 7.12(f) that the frequency measurement method starts with the values of the HD_{inter} at approximately 25%, while all of the other measurement methods start at 0%. This is due to the symmetry of the ROs that causes the ROs to have similar frequencies, thus the result of their difference can differ on various devices (e.g. 0x0011 on the first device, 0xFFF7 on the second device, therefore even the bit on position 1 differs among the devices).

7.3.2 PUF response evaluation

In the previous subsection we presented the evaluation of individual bit positions of the counter values using the metrics defined in Section 5.1.2. Following the methodology described in the same chapter, we selected the positions that exhibited satisfactory results in terms of their stability, entropy, and bias in order to create reliable, unique, and unpredictable PUF responses. Such positions could be e.g. positions 7–9.

We also discussed the influence of non-interleaved and interleaved measurements on the quality of the measured data. Just as in the case of the evaluation of individual bit positions in Table 7.20, non-interleaved measurements provide PUF responses with a higher stability (HD_{intra}). See Table 7.22 for comparison of the two approaches.

The next Table 7.23 presents the results of the comparison of the three PUF constructions and their three implementation variants (asymmetric ROs, all enabled asymmetric ROs, symmetric ROs). The Gray code was applied to the selected parts of the counter values. As we can see from the results, all three PUF constructions in all of their implementation variants provided both reliable and unique PUF responses with the highlighted position selections. Only the implementation variant that uses asymmetric ROs that were all enabled during the measurements exhibits a slightly worse stability of its outputs than the other implementation variants. This may be caused by the mutual influence of the ROs as they are all enabled and running during the measurements. The implementations with asymmetric and symmetric ROs provide fairly similar results for all the three measurement methods with negligible differences.

Again, as it was discussed in the case of Table 7.21 containing the evaluation of individual bit positions, we can observe that the frequency ratio and the frequency difference have comparable results in terms of their stability and uniqueness and the crystal reference provides slightly more stable PUF outputs than the other two measurement methods. As mentioned before, this may be caused by the fact that for the frequency ratio and the frequency difference, there is an influence of the instability of both of the ROs. More of this instability accumulates in the resulting counter values than for the crystal reference

Frequency ratio

	non-interleaved					interleaved				
positions	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**
w [-]	3	2	3	2	**4**	3	2	3	2	**4**
HD_{intra} [%]	1.15	1.73	2.45	3.68	**3.70**	1.38	2.07	2.83	4.24	**4.23**
HD_{inter} [%]	44.99	49.71	49.64	49.70	**49.85**	44.88	49.72	49.61	49.64	**49.77**

Frequency difference

	non-interleaved					interleaved				
positions	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**
w [-]	3	2	3	2	**4**	3	2	3	2	**4**
HD_{intra} [%]	1.27	1.91	2.47	3.71	**3.63**	1.37	2.06	2.65	3.98	**3.83**
HD_{inter} [%]	43.68	49.72	49.70	49.74	**49.80**	43.71	49.90	49.72	49.56	**49.77**

Crystal reference

	non-interleaved					interleaved				
positions	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**
w [-]	3	2	3	2	**4**	3	2	3	2	**4**
HD_{intra} [%]	0.88	1.33	1.85	2.77	**2.75**	1.06	1.58	2.21	3.32	**3.33**
HD_{inter} [%]	49.02	49.86	49.97	50.06	**50.07**	49.07	49.91	50.01	50.04	**50.09**

Table 7.22: Evaluation of the PUF responses for the three PUF constructions using different position selections of the measured counter values. Measurements were performed on 20 FPGAs using two sets of ROs, each consisting of 150 ROs. Two measurement methods are presented - non-interleaved and interleaved. The evaluation was performed on 5000 repetitions of the measurements using the design with asymmetric ROs. The Gray code was applied to the selected part of the counter values.

measurement method, where the instability of only one RO affects the measured counter values.

From the results in Table 7.23, we may conclude that the crystal reference would be the best measurement method to be used for the PUF. However, we must not forget that these measurement were performed at normal operating conditions, so both the temperature and supply voltage were stable and not manipulated with.

Apart from reliability and uniqueness, we also needed to evaluate the randomness of the PUF outputs. As described in Chapter 4, we use the NIST STS 2.1.2 [52]. Since this test requires long sequences of bits to be tested, we were forced to only use a subset of the tests available that are suitable for testing short sequences. The reason for this is that we would need a large population of experimental devices in order to be able to create long enough sequences.

Frequency ratio

	Asymmetric ROs					Asymmetric ROs, all enabled					Symmetric ROs				
positions	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**
w [-]	3	2	3	2	**4**	3	2	3	2	**4**	3	2	3	2	**4**
HD_{intra} [%]	1.15	1.73	2.45	3.68	**3.70**	1.33	1.99	2.65	3.94	**4.00**	1.27	1.90	2.50	3.75	**3.93**
HD_{inter} [%]	44.99	49.71	49.64	49.70	**49.85**	46.84	49.85	50.01	50.21	**50.02**	43.85	48.47	48.89	49.50	**49.19**

Frequency difference

	Asymmetric ROs					Asymmetric ROs, all enabled					Symmetric ROs				
positions	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	**7–9**	8–9	7–10	6–8	7–8	7–9	8–9	**7–10**
w [-]	3	2	3	2	**4**	3	2	**3**	2	4	3	2	3	2	**4**
HD_{intra} [%]	1.27	1.91	2.47	3.71	**3.63**	1.97	2.95	**3.78**	5.64	5.64	1.04	1.56	2.08	3.13	**3.08**
HD_{inter} [%]	43.68	49.72	49.70	49.74	**49.80**	49.82	49.98	**50.02**	49.94	50.04	42.87	49.51	49.73	49.99	**49.75**

Crystal reference

	Asymmetric ROs					Asymmetric ROs, all enabled					Symmetric ROs				
positions	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**
w [-]	3	2	3	2	**4**	3	2	3	2	**4**	3	2	3	2	**4**
HD_{intra} [%]	0.88	1.33	1.85	2.77	**2.75**	0.99	1.49	2.03	3.04	**2.94**	0.86	1.30	1.74	2.61	**2.58**
HD_{inter} [%]	49.02	49.86	49.97	50.06	**50.07**	48.95	50.23	50.15	49.93	**50.11**	49.48	49.82	49.89	49.91	**49.93**

Table 7.23: The PUF response evaluation for all implementation variants of the three PUF constructions using different position selections of the measured counter values. Measurements were performed on 20 FPGAs using two sets of ROs, each consisting of 150 ROs. The evaluation was performed on 5000 repetitions of the measurements. The Gray code was applied to the selected parts of the counter values.

For randomness evaluation purposes, we created a long sequence of bits composed of the reference (average) responses (R_ref_n) that were simply concatenated. We did this for various selections of the bit positions of the counter values. Such created sequence of bits for one particular selection of bit positions is the input for the NIST STS 2.1.2. This sequence is then internally split into shorter sequences by the NIST software. In one test case, we chose the number of these shorter sequences to be 10 because we did not have enough data to have a reasonable amount of long enough sequences. These shorter sequences were then tested on randomness using the subset of the tests in NIST STS. Their length was 300 bits if only one bit was selected from the counter values, or a multiple of 300 for more selected bits.

However, when multiple bits were selected from each counter value, we were able to obtain a longer concatenation of the PUF responses, thus were able to split this concatenation into 60 sequences, satisfying the condition of the NIST for the p-value uniformity test that requires at least 55 sequences. These sequences were still at least 100 bits long.

Tables 7.24, 7.25 and 7.26 present the results of the randomness evaluation using the NIST STS 2.1.2 for both PUF responses composed of individual bit positions (upper table) and PUF responses composed of various selections of multiple bits from the counter values (bottom table) for the three measurement methods (frequency ratio, frequency difference, and crystal reference respectively). The Gray code was only applied to selected bit positions when multiple bits were selected. The implementation variant with asymmetric ROs was used for the results presented in these tables. The minimum allowed pass rate for any test to be successful reported by the NIST STS is either 8/10 for 10 input sequences or 57/60 for 60 input sequences. The empty cells represent a 10/10 (60/60) pass rate and the red cells indicate the failure of the uniformity test of the p-values.

The randomness evaluation in case of frequency ratio presented in Table 7.24 shows that all positions from 7 and further successfully passed the NIST tests in the perspective of the pass rates. The frequency difference shown in Table 7.25 exhibits good randomness from positions 3 and further. In case of the crystal reference measurement method, the positions from 4 and further are suitable for the PUF in terms of their randomness.

When combined with the results in Table 7.23, the positions 7–10 seem to provide the best results for the PUF as they have low HD_{intra}, HD_{inter} close to the ideal 50%, and a good randomness according to the NIST tests. Even though some of the positions closer to the MSB still passed the NIST tests, they did not achieve ideal values of the HD_{inter} and could negatively affect the properties of the PUF for identification purposes (they would increase the false acceptance rate if uniqueness was lower). Moreover, the results from the NIST STS must be considered cautiously, since the data we were able to provide for the evaluation were not long enough for the tests to be effective.

The results for the other two implementation variants (all enabled asymmetric ROs, symmetric ROs) are presented in Appendix A.

positions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frequency	0/10	0/10	0/10	0/10		5/10			9/10		9/10					
BlockFrequency (m=20)	0/10	0/10	0/10	8/10												
BlockFrequency (m=30)	0/10	0/10	0/10	4/10		8/10							9/10			
CumulativeSums	0/10	0/10	0/10	0/10		5/10			9/10							
CumulativeSums	0/10	0/10	0/10	0/10		5/10			9/10							
Runs	0/10	0/10	0/10	2/10												
LongestRun	0/10	0/10	0/10	0/10		8/10						9/10		9/10		8/10
ApproximateEntropy (m=2)	0/10	0/10	0/10	0/10		6/10			9/10							
ApproximateEntropy (m=3)	0/10	0/10	0/10	0/10		9/10										

10 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	5/10	9/10		8/10		
BlockFrequency (m=20)		9/10				
BlockFrequency (m=30)	8/10					
CumulativeSums	5/10			8/10		
CumulativeSums	5/10			7/10		
Runs			9/10			
LongestRun						
ApproximateEntropy (m=2)	7/10			9/10	9/10	
ApproximateEntropy (m=3)	8/10				9/10	

60 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	57/60	58/60		59/60	59/60	59/60
BlockFrequency (m=20)	59/60	59/60		59/60		
BlockFrequency (m=30)	58/60		59/60		59/60	
CumulativeSums	58/60			59/60		59/60
CumulativeSums	59/60			59/60	59/60	
Runs	58/60		57/60		59/60	
LongestRun	0/60	0/60	0/60	57/60		
ApproximateEntropy (m=2)	59/60	59/60	59/60	59/60	59/60	
ApproximateEntropy (m=3)		59/60	59/60	59/60	59/60	

Table 7.24: Evaluation of randomness of the PUF responses composed of various selections of bit positions for asymmetric ROs, frequency ratio measurement method. When selecting multiple bits from each counter value (bottom table), the Gray code was applied to selected bits. Average responses were used for the evaluation.

positions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frequency	0/10	1/10														
BlockFrequency (m=20)	9/10							9/10								
BlockFrequency (m=30)	7/10															
CumulativeSums	0/10	0/10														
CumulativeSums	0/10	7/10														
Runs	8/10				8/10											
LongestRun																
ApproximateEntropy (m=2)	0/10	5/10			9/10							9/10	9/10			
ApproximateEntropy (m=3)	1/10		9/10									9/10				

10 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency						
BlockFrequency (m=20)						
BlockFrequency (m=30)						
CumulativeSums						
CumulativeSums						
Runs						
LongestRun						
ApproximateEntropy (m=2)						
ApproximateEntropy (m=3)		9/10				

60 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency		59/60				
BlockFrequency (m=20)	59/60		59/60			58/60
BlockFrequency (m=30)			59/60			
CumulativeSums						
CumulativeSums					59/60	
Runs			59/60			
LongestRun	0/60	0/60	0/60	59/60	58/60	
ApproximateEntropy (m=2)			59/60		59/60	59/60
ApproximateEntropy (m=3)			58/60	59/60	59/60	58/60

Table 7.25: Evaluation of randomness of the PUF responses composed of various selections of bit positions for asymmetric ROs, frequency difference measurement method. When selecting multiple bits from each counter value (bottom table), the Gray code was applied to the selected bits. Average responses were used for the evaluation.

positions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frequency	0/10	2/10	7/10		8/10	9/10							9/10			
BlockFrequency (m=20)	0/10	0/10			9/10	9/10										
BlockFrequency (m=30)	0/10	0/10			8/10											
CumulativeSums	0/10	2/10	7/10		8/10	9/10							9/10			
CumulativeSums	0/10	0/10	7/10		9/10	9/10										
Runs	0/10	5/10													9/10	
LongestRun	0/10	3/10	9/10													
ApproximateEntropy (m=2)	0/10	4/10	9/10		9/10											
ApproximateEntropy (m=3)	0/10	5/10	8/10													

10 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	8/10			9/10		
BlockFrequency (m=20)						
BlockFrequency (m=30)						
CumulativeSums	7/10			8/10		
CumulativeSums	9/10			9/10		
Runs						
LongestRun						
ApproximateEntropy (m=2)						
ApproximateEntropy (m=3)			9/10			

60 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	58/60		59/60	59/60		
BlockFrequency (m=20)		59/60	59/60			
BlockFrequency (m=30)	58/60		58/60		59/60	
CumulativeSums	58/60		59/60	58/60		
CumulativeSums	58/60		58/60	59/60		
Runs	59/60		59/60		59/60	
LongestRun	0/60	0/60	0/60	58/60		
ApproximateEntropy (m=2)	58/60					59/60
ApproximateEntropy (m=3)	59/60		58/60		58/60	59/60

Table 7.26: Evaluation of randomness of the PUF responses composed of various selections of bit positions for asymmetric ROs, crystal reference measurement method. When selecting multiple bits from each counter value (bottom table), the Gray code was applied to the selected bits. Average responses were used for the evaluation.

7.3.3 Influence of supply voltage and temperature

In order to evaluate the stability of the three measurement methods at varying temperature and supply voltage, we performed all measurements on 5 Digilent Cmod S7 FPGA boards that were modified so that the supply voltage could be set automatically on pre-defined voltage levels. There were 5 voltage levels, specifically from 0.92V to 1.08V with the steps of 40mV.

The FPGA boards were then placed into a climate chamber that we used to set the temperature from 0°C to 60°C with 10°C steps. For stability evaluation purposes the reference environmental conditions were 1.00V for voltage and 30°C for temperature.

Before taking the measurement during the voltage and temperature level we chose, we had to wait to let the FPGA stabilize its temperature. After this waiting time, the measurements were performed with 5500 repetitions. These measurements were non-interleaved as explained in the previous subsection related to the measurements at normal operating conditions. For evaluation purposes, the first 500 measurements were skipped as they were used as a warm-up for ROs.

The implementation variants we used for this evaluation were with asymmetric and symmetric ROs. The variant with the all enabled asymmetric ROs was not evaluated as it already had a slightly higher HD_{intra} at normal operating conditions and therefore it was not of interest for our purposes. Our goal was to compare the three measurement methods and their resiliency against varying physical conditions and the effect of the mutual symmetry of the ROs on the stability of the PUF outputs.

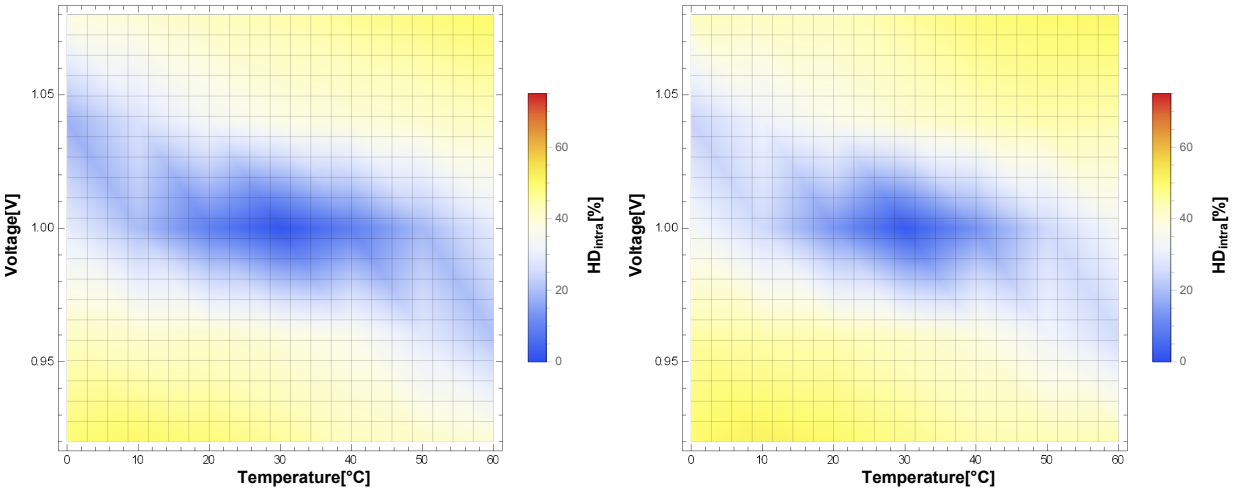
For a better overview, we provide the results in Fig. 7.13, Fig. 7.14 and Fig. 7.15. The results can also be found in the table form in Appendix A. Fig. 7.13 shows the results for asymmetric and symmetric ROs (the upper two figures are for asymmetric ROs, the two bottom graphs for symmetric ROs), the frequency ratio measurement method, and the selections of positions 7–8 and 7–9. Fig. 7.14 shows the same for the frequency difference and Fig. 7.15 for the crystal reference measurement method.

It can be immediately noticed that symmetric ROs provide significantly better results in terms of stability at varying temperature and supply voltage (compare e.g. Fig. 7.13(a) and Fig. 7.13(c)) except for the crystal reference measurement method (Fig. 7.15(c) and Fig. 7.15(d)). The reason for this is that the frequencies of the ROs in a pair weare influenced in a very similar way and therefore either the frequency ratio or the frequency difference did not change that much as in case of asymmetric ROs.

However, in case of the crystal reference, symmetric ROs do not help with the varying conditions. This is because the ROs are changing their frequencies with the change of voltage and temperature. Hence, the resulting counter value depends only on the frequency of the respective RO (see Eq. 5.15). Therefore, the counter value changes in the same way as the frequency of the RO. In contrast, both the frequency ratio and the frequency difference are differential measurements and therefore dependent on the change of either the ratio or the difference of the frequencies of the paired ROs (Eq. 5.13 and Eq 5.14). When symmetric ROs are used, we can expect that the change in both the ratio and the difference will be smaller than the changes in the frequencies themselves.

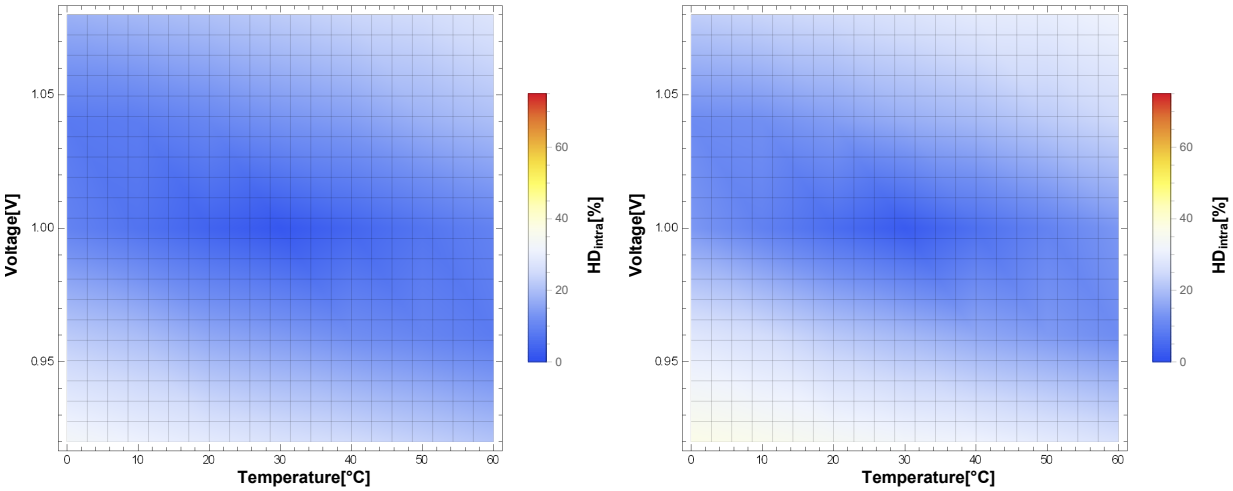
Moreover, if we compare the frequency ratio to the frequency difference, we can observe that the frequency ratio exhibits better behaviour in terms of stability of the PUF outputs (Fig. 7.13(c) vs Fig. 7.14(c)). Such result could have been expected since the counter values in the frequency ratio measurement method depend on the ratio of the frequencies of the selected ROs and the change of the ratio with varying temperature and supply voltage will hence be smaller than the change of the difference of those frequencies.

Overall, the frequency ratio as proposed in [A.3, A.4] seems to be the best measurement method to mitigate the effects of varying temperature and supply voltage when implemented with symmetric ROs.



(a) Asymmetric ROs, frequency ratio, positions 7–8.

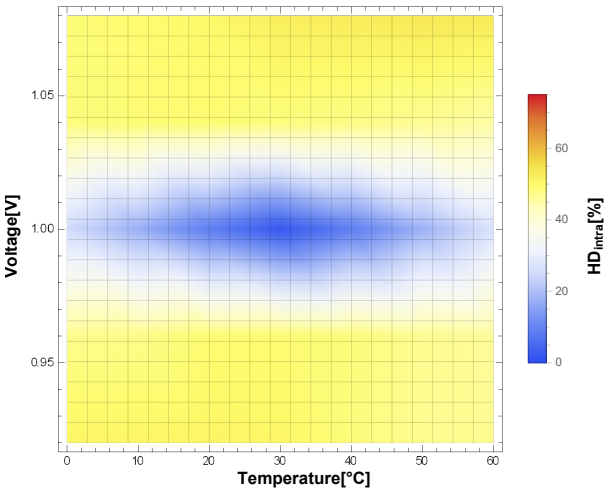
(b) Asymmetric ROs, frequency ratio, positions 7–9.



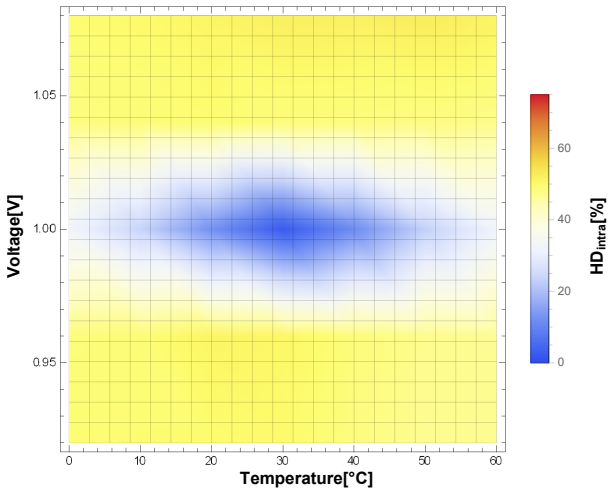
(c) Symmetric ROs, frequency ratio, positions 7–8.

(d) Symmetric ROs, frequency ratio, positions 7–9.

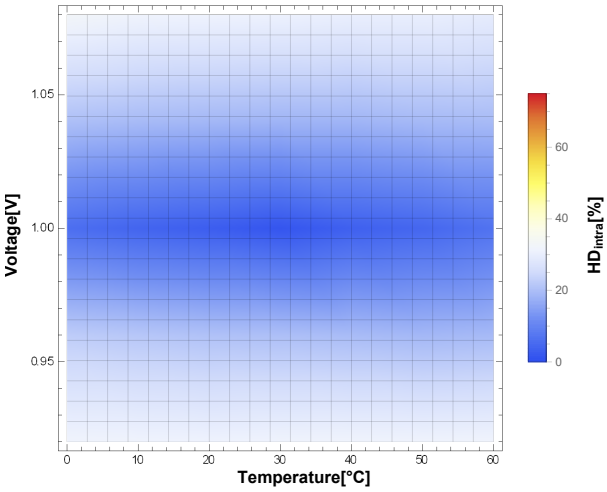
Figure 7.13: Evaluation of the PUF responses at varying temperature and supply voltage, frequency ratio measurement method.



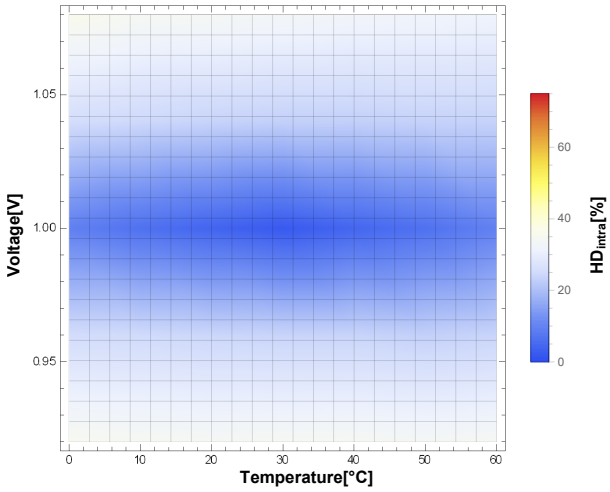
(a) Asymmetric ROs, frequency difference, positions 7–8.



(b) Asymmetric ROs, frequency difference, positions 7–9.

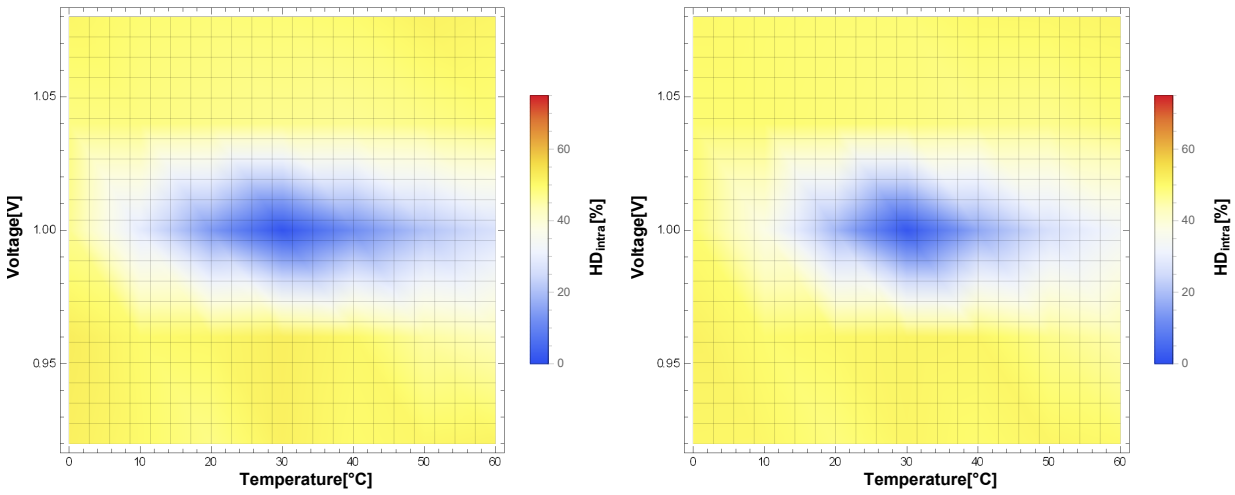


(c) Symmetric ROs, frequency difference, positions 7–8.



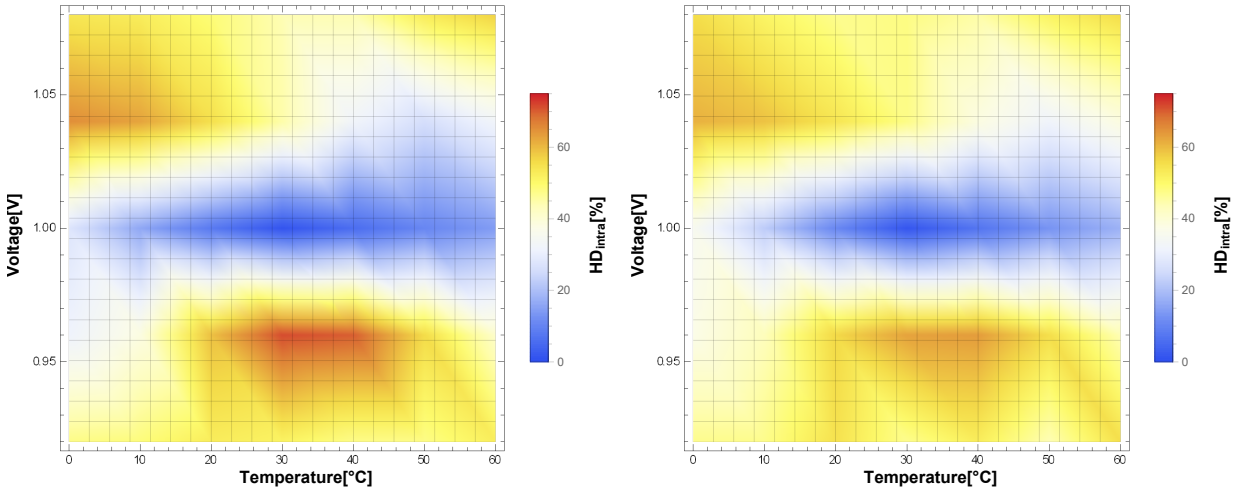
(d) Symmetric ROs, frequency difference, positions 7–9.

Figure 7.14: Evaluation of the PUF responses at varying temperature and supply voltage, frequency difference measurement method.



(a) Asymmetric ROs, xtal reference, positions 7–8.

(b) Asymmetric ROs, xtal reference, positions 7–9.



(c) Symmetric ROs, xtal reference, positions 7–8.

(d) Symmetric ROs, xtal reference, positions 7–9.

Figure 7.15: Evaluation of the PUF responses at varying temperature and supply voltage, crystal reference measurement method.

Chapter 8

Conclusion

In this chapter we point out the contributions of this dissertation thesis, provide a summary of our work, our most notable findings, and provide possible directions for future work.

8.1 Contributions of the dissertation thesis

In the following list, we repeat the goals of this dissertation thesis from Section 1.2 together with a brief description of their fulfillment. More details about meeting the goals and the results are provided in the following section containing the summary of this work.

1. **Analyse existing PUF designs suitable for FPGAs with focus on RO based PUFs.**

   This analysis was presented in Chapter 3 where numerous PUF constructions suitable for FPGAs were described. Section 3.11.2 contains the description of PUF constructions based on ROs.

2. **Improve the proposed ROPUF design of [A.3].**

   We proposed a new PUF construction based on ROs in our early work [A.3]. This proposal uses a frequency ratio as the measurement method where two ROs are measured simultaneously using two counters. When one of the counter overflows, the measurement is stopped and the value from the counter that did not overflow is used for further processing as the PUF output. A detailed description of our proposal together with its improvements was provided in Chapter 5.

3. **Analyse the properties of the proposed PUF design.**

   In Chapter 5 we described the properties of our proposal. Section 5.2 contains the description of the properties of our ROPUF design. Section 5.3 presents the possibility of using the same proposed design as a TRNG. Finally, Section 5.4 shows two different measurement methods that could be used in our proposal instead of the

frequency ratio. The proposed design was analysed and examined experimentally in Chapter 7.

4. **Examine and evaluate the behaviour of the proposed PUF design at both stable and varying temperature and supply voltage.**

   Chapter 7 contains the results of our measurements for the proposed PUF design. We evaluated the PUF proposal on three different platforms (Xilinx Spartan-3E, Spartan-6, and Spartan-7). The proposed PUF exhibits a satisfactory behaviour in terms of reliability (HD_{intra}), uniqueness (HD_{inter}), and randomness. Extensive measurements related to varying temperature and voltage were performed on the Xilinx Spartan-7 FPGAs and used to compare the behaviour of the three presented measurement methods and different possible implementations. As shown in the results in Chapter 7, the frequency ratio measurement method combined with symmetric ROs provides the best stability at varying physical conditions.

8.2 Summary

This dissertation thesis was mainly focused on the topic of PUFs. In Chapter 2 we provided a general description of PUFs with their desired properties, their classification, and possible applications. The main concept of the PUF is based on random variations arising during the manufacturing process that cause each device to possess unique physical properties. These physical properties are, for example, a circuit delay or a bias of the memory cells after power-up. The variations of physical properties arising during the manufacturing process are random since they arise from the influence of random and uncontrollable effects.

The main applications of PUFs are device identification, authentication, and key generation. However, as discussed in Section 2.2, PUFs have to meet some properties, such as stability of their outputs, uniqueness, and unpredictability. If some of these requirements are not met, then the PUF outputs may either be unreliable due to high erroneous or easily attacked when the uniqueness or the unpredictability property is not satisfied. Ideally, we need PUF responses with a very low number (or 0) of errors and to appear random in the perspective of the population of challenges and devices (as if they were generated by a TRNG).

Chapter 3 presents numerous existing PUF constructions, however, not all of them since there are many. **In our literature research we focused on intrinsic PUFs suitable for FPGAs since this was our target platform.** The two major classes of PUFs according to their sources of randomness are delay-based and memory-based PUFs. The most common PUF design is based on SRAM since many electronic devices contain an embedded SRAM. This PUF construction reads the content of the SRAM after power-up and uses it to create the PUF output. However, some FPGAs initialise their memory after power-up and then all randomness is lost. This fact led to proposals of other memory-based PUFs suitable for FPGAs such as the Butterfly PUF, the Latch PUF and the Flip-flop PUF.

Delay-based PUFs exploit random variations in delays of logic gates and their interconnects. One of the first delay-based PUFs was the Arbiter PUF, which is, however, more suitable for ASICs than for FPGAs due to the requirements for a perfect symmetry of the paths and the fairness of the arbiter. Other examples are the Ring Oscillator PUF and the Glitch PUF. **In our work, we put an emphasis on describing various ROPUF constructions since our PUF proposal is also based on ROs.**

Chapter 4 provides an overview of PUF evaluation parameters. We present several evaluation parameters together with their different definitions in different works. We provide a comparison of these definitions and show their behaviour on simulated PUF responses. The PUF evaluation parameters that we decided to use in this dissertation thesis are reliability (HD_{intra}), uniqueness (HD_{inter}), and randomness. Their definition and description is shown in Section 4.3.

The RO based PUF we proposed was described in Chapter 5. The basic concept of our proposal is based on counting the number of oscillations of two paired ROs. When one of the counters that counts the oscillations overflows, the measurement is stopped and the value of the counter that did not overflow is used for further processing. In this processing, a suitable cluster of bits of the counter values are used as the PUF response. The final PUF response is created by a concatenation of these clusters of bits from various counter values measured using different RO pairs. **The methodology of selecting the suitable bits of the counter values for the PUF output was explained in Section 5.1.**

One advantage of our proposal is that we are able to extract multiple bits from each RO pair compared to the classical approach where frequencies of the ROs are compared, providing only one bit based on the result of this comparison. **Another advantage of our proposal is that it does not require the ROs to be mutually symmetric**, opposed to the approach where the frequency comparison is performed and where the ROs have to be mutually symmetric in order to provide an unpredictable result of the comparison. However, as it was shown in Chapter 7, when we use symmetric ROs we are able to achieve a better stability of the PUF responses at varying environmental conditions.

In Chapter 5 we discussed the properties of our proposed PUF design. We also provided a description of two other measurement methods that can be used in our proposal without any significant changes in the design. Furthermore, we explain the possibility of using the same proposed ROPUF design as a TRNG by selecting different bits from the measured counter values.

In Chapter 6, the implementation of the experimental circuits was described. Several platforms were used, specifically Digilent Basys 2 FPGA boards containing the Xilinx Spartan3E-100 CP132, Digilent Nexys 3 FPGA boards with the Xilinx Spartan-6 XC6LX16-CS324, and Digilent Cmod S7 FPGA boards containing the Xilinx Spartan-7 XC7S25-1CSGA225C. In the same chapter we also describe the implementation of our design with both asymmetric and symmetric ROs and the three implementation variants used for measurements on Spartan-7 FPGAs.

Finally, Chapter 7 presents the results of our experiments. This chapter is

divided into three sections based on the implementation platform that was used for the measurements (Spartan-3E, Spartan-6 and Spartan-7). The Xilinx Spartan-3E FPGAs were used for the initial measurements. **These measurements have shown that the proposed PUF design is suitable for FPGAs and exhibits a satisfactory behaviour at stable operating conditions.** We also performed additional measurements at varying temperature and supply voltage and compared the behaviour of the proposed ROPUF using asymmetric and symmetric ROs.

The results sho, that each RO pair can provide up to 3–4 bits for the PUF output when the Gray code is used and when the operating conditions are stable. Moreover, the PUF responses exhibit satisfactory results in terms of their reliability (HD_{intra}), uniqueness (HD_{inter}), and randomness, for the evaluation of which we used the NIST STS. However, the changing physical conditions have a significant impact on the stability of the PUF outputs. Nevertheless, when symmetric ROs are used, the PUF responses are distinctly more stable.

On the next experimental platform, the Xilinx Spartan-7 FPGAs, more extensive measurements were performed. **We evaluated the behaviour of the proposed PUF using the three different measurement methods (frequency ratio, frequency difference, and crystal reference) and using three different implementation variants (asymmetric ROs, all enabled asymmetric ROs, symmetric ROs). All of these measurement methods and implementation variants exhibit satisfactory results at stable operating conditions, with the crystal reference measurement method being the most reliable one.**

To perform more extensive measurements at varying temperature and voltage, we modified 5 Xilinx Spartan-7 FPGA boards in order to be able to change the power supply voltage. We used 5 voltage levels, specifically from 0.92V to 1.08V with the steps of 40 mV. These modified FPGA boards were then put into a climate chamber and tested in the temperature range from 0°C to 60°C with 10°C steps. There are significant differences in stability between the three measurement methods. The crystal reference is the least resistant measurement method at varying voltage and temperature, because the frequencies of the ROs are changing with the change of voltage and temperature and there is no differential measurement as the ROs are measured against a very stable crystal oscillator.

If we compare the frequency ratio to the frequency difference, we can also observe that frequency ratio exhibits better behaviour in terms of stability of the PUF outputs. Such result could have been expected since the counter values in the frequency ratio measurement method depend on the ratio of the frequencies of the selected ROs and the change of the ratio with varying temperature and supply voltage is smaller than the change of the difference of the same frequencies.

Moreover, as in the case of the experiments performed on the Spartan-3E FPGAs, **symmetric ROs provide significantly better results in terms of stability at varying temperature and supply voltage, except for the crystal reference measurement method. Overall, the frequency ratio that is used in our ROPUF proposal**

provides the best stability of the PUF responses and is therefore more able to mitigate the effects of varying temperature and supply voltage when implemented with symmetric ROs.

8.3 Future work

We suggest the following directions for future work:

- Further examine the influence of the symmetry and RO placement on the PUF output stability at varying environmental conditions.
- Create a statistical RO model that would be useful for both PUF and TRNG evaluation.
- Develop an IP block utilizing both the proposed PUF and TRNG to provide reliable and strong cryptographic keys to support asymmetric/symmetric cryptography.
- Investigate the influence of aging on the proposed ROPUF design.
- Examine possible attacks on the proposed PUF/TRNG.
- Use online tests to detect failuring TRNG.

Bibliography

[1] Becker, G. T. The gap between promise and reality: On the insecurity of XOR arbiter PUFs. In *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, Berlin, Heidelberg, 2015. pp. 535-555.

[2] Bernard, F.; Haddad, P.; Fischer, V.; Nicolai, J. From Physical to Stochastic Modeling of a TERO-Based TRNG. In *Journal of Cryptology*, 2019, 32(2), pp. 435-458.

[3] Bossuet, L.; Ngo, X. T.; Cherif, Z.; Fischer, V. A PUF based on a transient effect ring oscillator and insensitive to locking phenomenon. In *IEEE Transactions on Emerging Topics in Computing*, 2014, 2.1, pp. 30–36.

[4] Busch, H.; Sotáková; M.; Katzenbeisser, S.; Sion, R.: *The PUF Promise (Short Paper)*. 2010, [Cited 2016-05-02]. Available from: `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.187.324&rep=rep1&type=pdf`

[5] Cherif, Z.; Danger, J.; Guilley, S.; Bossuet, L. An easy-to-design PUF based on a single oscillator: The loop PUF. In *15th Euromicro Conference on Digital System Design, DSD 2012*, Turkey, 2012, pp. 156-162.

[6] Chen, Q.; Csaba, G.; Lugli, P.; Schlichtmann, U.; Rührmair, U. The Bistable Ring PUF: A new architecture for strong Physical Unclonable Functions. In *Hardware-Oriented Security and Trust (HOST), 2011 IEEE International Symposium on*, 2011, pp. 134–141.

[7] Delavar, M.; Mirzakuchaki, S.; Mohajeri, J. A ring oscillator-based PUF with enhanced challenge-response pairs. In *Canadian Journal of Electrical and Computer Engineering*, 2016, 39(2), pp. 174–180.

[8] Delvaux, J. Refutation and Redesign of a Physical Model of TERO-based TRNGs and PUFs. *Eprint. Iacr. Org*, 2019.

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

[10] Dichtl, M. Bad and good ways of post-processing biased physical random numbers. In *International Workshop on Fast Software Encryption*, Springer Berlin Heidelberg, 2007.

[11] Digilent: *Digilent, Adept*. [Cited 2018-01-08]. Available from: `https://reference.digilentinc.com/digilent_adept_2`

[12] Digilent: *Digilent, Adept SDK*. [Cited 2018-01-08]. Available from: `https://reference.digilentinc.com/digilent_adept_2`

[13] Digilent: *Digilent Basys 2 Board Reference Manual*. [Cited 2020-06-03]. Available from: `https://reference.digilentinc.com/_media/basys2:basys2_rm.pdf`

[14] Digilent: *Digilent Nexys 3 Board Reference Manual*. [Cited 2020-06-03]. Available from: `https://reference.digilentinc.com/_media/nexys:nexys3:nexys3_rm.pdf`

[15] Digilent: *Digilent Cmod S7 Board Reference Manual*. [Cited 2020-06-03]. Available from: `https://reference.digilentinc.com/reference/programmable-logic/cmod-s7/reference-manual`

[16] Fischer, V. Design and evaluation of a physical random number generator. In *Cryptographic architectures embedded in logic devices*, Smolenice, SK, June 2017.

[17] Gabor, D. Theory of communication. In *Journal of the Institute Electrical Engineers*, 93 (26), 1946, pp. 429–457.

[18] Gassend, B. *Physical Random Functions*. Dissertation thesis. Massachusetts Institute of Technology, 2003, [Cited 2016-05-02]. Available from: `http://www.textfiles.com/bitsavers/pdf/mit/lcs/tr/MIT-LCS-TR-881.pdf`

[19] Gassend, B.; Clarke, D.; Dijk, M.; Devadas, S. Silicon Physical Random Functions. In *Proceedings of the 9th ACM conference on Computer and communications security*. ACM, 2002, pp. 148–160.

[20] Herder, C.; Yu, M. D.; Koushanfar, F.; Devadas, S. Physical Unclonable Functions and Applications: A Tutorial. In *Proceedings of the IEEE*, 2014, 102.8, pp. 1126–1141.

[21] Hesselbarth, R.; Heyszl, J.; Sigl, G. Fast and reliable PUF response evaluation from unsettled bistable rings. Microprocessors and Microsystems, 2017, 52, pp. 325-332.

[22] Holcomb, D. E.; Burleson, W. P.; Fu, K. Power-up SRAM State as an Identifying Fingerprint and Source of True Random Numbers. In *IEEE Transactions on Computers*, 2009, 58.9, pp. 1198–1210.

[23] Hori, Y.; Yoshida, T.; Katashita, T.; Satoh, A. Quantitative and statistical performance evaluation of arbiter physical unclonable functions on FPGAs. In *2010 International Conference on Reconfigurable Computing and FPGAs*. IEEE, 2010, pp. 298–303.

[24] Katzenbeisser, S; Kocabaş; Ü.; Rožić; V.; Sadeghi, A.; Verbauwhede, I.; Wachsmann, Ch. PUFs: Myth, Fact or Busted? A Security Evaluation of Physically Unclonable Functions (PUFs) Cast in Silicon. In *Cryptographic Hardware and Embedded Systems–CHES 2012*. Springer Berlin Heidelberg, 2012, pp. 283–301.

[25] Killmann, W., Schindler, W. A Proposal for: Functionality Classes and Evaluation Methodology for True (Physical) Random Number Generators, Version 3.1, English translation, 25.09.2001

[26] Killmann, W., Schindler, W. A proposal for: functionality classes for random number generators. 2011

[27] Kim, J. S.; Patel, M.; Hassan, H.; Mutlu, O. The DRAM latency PUF: Quickly evaluating physical unclonable functions by exploiting the latency-reliability tradeoff in modern commodity DRAM devices. In *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2018 pp. 194–207).

[28] Kumar, S. S.; Guajardo, J.; Maes, R.; Schrijen, G.; Tuyls, P. Extended abstract: The butterfly PUF protecting IP on every FPGA. In *Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on*. IEEE, 2008, pp. 67–70.

[29] Laban, M. Compact Bit Selection Procedure for SRAM PUF Embedded in a Low-cost Microcontroller. In *Počítačové Architektury & Diagnostika*. September 4–6, 2019 – Doksy, Czech Republic.

[30] Lee, J. W.; Lim, D.; Gassend, B.; Suh, G. E.; Dijk, M.; Devadas, S. A technique to build a secret key in integrated circuits for identification and authentication applications. In *VLSI Circuits, 2004. Digest of Technical Papers. 2004 Symposium on*. IEEE, 2004, pp. 176–179.

[31] Lee, S.; Oh, M. K.; Kang, Y.; Choi, D. Implementing a phase detection ring oscillator PUF on FPGA. In *2018 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2018, pp. 845–847.

[32] Van der Leest, V.; Schrijen, G. J.; Handschuh, H.; Tuyls, P. Hardware intrinsic security from D flip-flops. In *Proceedings of the fifth ACM workshop on Scalable trusted computing*. 2010, pp. 53–62.

[33] Lim, D.; Lee, J. W.; Gassend, B.; Suh, G. E.; Dijk, M.; Devadas, S. Extracting secret keys from integrated circuits. In *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2005, 13.10, pp. 1200–1205.

[34] Linnartz, J. P.; Tuyls, P. New shielding functions to enhance privacy and prevent misuse of biometric templates. In *International Conference on Audio-and Video-Based Biometric Person Authentication*. Springer, Berlin, Heidelberg, 2003, pp. 393-402.

[35] Maes, R. *Physically Unclonable Functions: Constructions, Properties and Applications*. Dissertation thesis. Katholieke Universiteit Leuven, 2012, [Cited 2016-05-02]. Available from: `https://securewww.esat.kuleuven.be/cosic/publications/thesis-211.pdf`

[36] Maes, R.; Tuyls, P.; Verbauwhede, I. Intrinsic PUFs from Flip-flops on Reconfigurable Devices. In *3rd Benelux workshop on information and system security (WISSec 2008)*, 2008, [Cited 2016-05-02]. Available from: `https://www.researchgate.net/profile/Roel_Maes/publication/228615879_Intrinsic_PUFs_from_flip-flops_on_reconfigurable_devices/links/0912f51126478b8661000000.pdf`

[37] Maes, R.; Verbauwhede, I. Physically Unclonable Functions: A Study on the State of the Art and Future Research Directions. In *Towards Hardware-Intrinsic Security*. Springer Berlin Heidelberg, 2010, pp. 3–37.

[38] Maiti, A.; Schaumont, P. Improving the quality of a Physical Unclonable Function using configurable Ring Oscillators. In *Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on*. IEEE, 2009, pp. 703–707.

[39] Maiti, A.; Kim, I.; Schaumont, P. A robust physical unclonable function with enhanced challenge-response set. In *IEEE Transactions on Information Forensics and Security*, 2011, 7(1), pp. 333-345.

[40] Maiti, A.; Gunreddy, V.; Schaumont, P. A systematic method to evaluate and compare the performance of physical unclonable functions. In *Embedded systems design with FPGAs*. Springer, New York, NY, 2013, pp. 245–267.

[41] Marchand, C.; Bossuet, L.; Cherkaoui, A. Enhanced TERO-PUF Implementations and Characterization on FPGAs. In *Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. ACM, 2016, pp. 282–282.

[42] Sýs, M.; Říha, Z.; Matyáš, V.; Marton, K.; Suciu, A. (2015). On the interpretation of results from the NIST statistical test suite. In *Science and Technology*, 2015, 18(1), pp. 18–32.

[43] Morozov, S.; Maiti, A.; Schaumont, P. An Analysis of Delay Based PUF Implementations on FPGA. In *Reconfigurable Computing: Architectures, Tools and Applications*. Springer Berlin Heidelberg, 2010, pp. 382–387.

[44] Murdoch, D. J;, Tsai, Y. L.; Adcock, J. (2008). P-values are random variables. *The American Statistician*, 2008, 62(3), pp. 242–245.

[45] Nguyen, P. H.; Sahoo, D. P.; Chakraborty, R. S.; Mukhopadhyay, D. Efficient attacks on robust ring oscillator PUF with enhanced challenge-response set. In *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*, EDA Consortium, 2015, pp. 641–646.

[46] Pang, Z.; Zhang, J.; Zhou, Q.; Gong, S.; Qian, X.; Tang, B. Crossover ring oscillator PUF. In *2017 18th International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2017, pp. 237–243.

[47] Pappu, R. S. *Physical One-Way Functions*. Dissertation thesis. Massachusetts Institute of Technology, 2001, [Cited 2016-05-02]. Available from: `http://alumni.media.mit.edu/~pappu/pdfs/Pappu-PhD-POWF-2001.pdf`

[48] Petit, J.; Bösch, Ch.; Feiri, M.; Kargl, F. On the potential of PUF for pseudonym generation in vehicular networks. In *Vehicular Networking Conference (VNC), 2012 IEEE*. IEEE, 2012, pp. 94–100.

[49] Platonov, M. *SRAM-Based Physical Unclonable Function on an Atmel ATmega Microcontroller*. Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2013.

[50] Platonov, M.; Hlaváč, J.; Lórencz, R. Using Power-Up SRAM State of Atmel ATmega1284P Microcontrollers as Physical Unclonable Function for Key Generation and Chip Identification. In *Information Security Journal: A Global Perspective*. 2013, 22.5–6, pp. 244–250.

[51] Rührmair, U.; Hilgers, C.; Urban, S.; Weiershäuser, A.; Dinter, E.; Forster, B.; Jirauschek, C. Optical pufs reloaded. *Eprint. Iacr. Org*, 2013.

[52] Ruhkin, A. et al. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. *NIST Special Publication 800-22 Revision 1a*. 2010.

[53] Rosenblatt, S.; Fainstein, D.; Cestero, A.; Safran, J.; Robson, N.; Kirihata, T.; Iyer, S. S. Field tolerant dynamic intrinsic chip ID using 32 nm high-K/metal gate SOI embedded DRAM. In *IEEE Journal of Solid-State Circuits*, 2013, 48(4), pp. 940–947.

[54] Rosenblatt, S.; Chellappa, S.; Cestero, A.; Robson, N.; Kirihata, T.; Iyer, S. S. A self-authenticating chip architecture using an intrinsic fingerprint of embedded DRAM. In *IEEE Journal of Solid-State Circuits*, 2013, 48(11), pp. 2934–2943.

[55] Sahoo, D. P.; Mukhopadhyay, D.; Chakraborty, R. S. Design of low area-overhead ring oscillator PUF with large challenge space. In *2013 International Conference on Reconfigurable Computing and FPGAs (ReConFig)*. IEEE, 2013, pp. 1–6.

[56] Sahoo, D. P.; Saha, S.; Mukhopadhyay, D.; Chakraborty, R. S.; Kapoor, H. Composite PUF: A new design paradigm for Physically Unclonable Functions on FPGA. In *2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*. IEEE, 2014, pp. 50–55.

[57] Su, Y.; Holleman, J.; Otis, B. A 1.6pJ/bit 96% Stable Chip-ID Generating Circuit using Process Variations. In *IEEE Journal of Solid-State Circuits*, 2008, 43.1, pp. 69–77.

[58] Suh, G. E.; Devadas, S. Physical Unclonable Functions for Device Authentication and Secret Key Generation. In *Proceedings of the 44th annual Design Automation Conference*. ACM, 2007, pp. 9–14.

[59] Suzuki, D.; Shimizu, K. The Glitch PUF: A New Delay-PUF Architecture Exploiting Glitch Shapes. In *Cryptographic Hardware and Embedded Systems, CHES 2010*. Springer Berlin Heidelberg, 2010, pp. 366–382.

[60] Talukder, B. B.; Ray, B.; Forte, D.; Rahman, M. T. PreLatPUF: Exploiting DRAM Latency Variations for Generating Robust Device Signatures. IEEE Access, 2019, 7, pp. 81106-81120.

[61] Tehranipoor, F.; Karimian, N.; Xiao, K.; Chandy, J. DRAM based intrinsic physical unclonable functions for system level security. In *Proceedings of the 25th edition on Great Lakes Symposium on VLSI*. ACM, 2015, pp. 15–20.

[62] Tuyls, P.; Schrijen, G.; Škorić, B.; Geloven, J.; Verhaegh, N.; Wolters, R. Read-Proof Hardware from Protective Coatings. In *Cryptographic Hardware and Embedded Systems-CHES 2006*. Springer Berlin Heidelberg, 2006, pp. 369–383.

[63] Varchola, M.; Drutarovsky, M. New High Entropy Element for FPGA Based True Random Number Generators. In *Cryptographic Hardware and Embedded Systems, CHES 2010*. Springer Berlin Heidelberg, 2010, pp. 351–365.

[64] Xilinx: *ISE Design Suite*. [Cited 2020-06-03]. Available from: `https://www.xilinx.com/products/design-tools/ise-design-suite.html`

[65] Xilinx: *Spartan-3E FPGA Family Data Sheet*. [Cited 2020-06-10]. Available from: `https://www.xilinx.com/support/documentation/data_sheets/ds312.pdf`

[66] Xilinx: *Spartan-6 Family Overview*. [Cited 2020-06-10]. Available from: `https://www.xilinx.com/support/documentation/data_sheets/ds160.pdf`

[67] Xilinx: *7 Series FPGAs Data Sheet: Overview*. [Cited 2020-06-10]. Available from: `https://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf`

[68] Xilinx: *Vivado Design Suite v2019.1*. [Cited 2020-06-03]. Available from: `https://www.xilinx.com/products/design-tools/vivado.html`

[69] Xin, X.; Kaps, J.; Gaj, K. A Configurable Ring-Oscillator-Based PUF for Xilinx FPGAs. In *Digital System Design (DSD), 2011 14th Euromicro Conference on*. IEEE, 2011, pp. 651–657.

[70] Xiong, W.; Schaller, A.; Anagnostopoulos, N. A.; Saleem, M. U.; Gabmeyer, S.; Katzenbeisser, S.; Szefer, J. Run-time accessible DRAM PUFs in commodity devices. In *International Conference on Cryptographic Hardware and Embedded Systems*. Springer, 2016, Berlin, Heidelberg, pp. 432–453.

[71] Yin, C. -E. D.; Qu, G.: LISA: Maximizing RO PUF's secret extraction. In *Hardware-Oriented Security and Trust (HOST), 2010 IEEE International Symposium on*. IEEE, 2010, pp. 100–105.

[72] Zhang, L.; Wang, C.; Liu, W.; O'Neill, M.; Lombardi, F. XOR gate based low-cost configurable RO PUF. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2017, pp. 1–4.

Reviewed Publications of the Author Relevant to the Thesis

[A.1] Buchovecká, S.; Lórencz, R.; Kodýtek, F.; Buček, J.: True Random Number Generator based on ROPUF circuit. In *Euromicro Conference on Digital System Design*. August 31 – September 2, 2016 – Limassol, Cyprus.

The paper has been cited in:

- Khafizovich, L. R.; L'Vovich, S. E. Theory of ternary jitter-based true random number generators composed of identical gates. In *UCHENYE ZAPISKI KAZANSKOGO UNIVERSITETA-SERIYA FIZIKO-MATEMATICHESKIE NAUKI*, 2017, 159(2), pp. 246–262. ISSN 2541-7746.

[A.2] Buchovecká, S.; Lórencz, R.; Kodýtek, F.; Buček, J.: True random number generator based on ring oscillator PUF circuit. In *Microprocessors and Microsystems*. 2017, ISSN 0141-9331, https://doi.org/10.1016/j.micpro.2017.06.021.

The paper has been cited in:

- Alibeigi, I.; Amirany, A.; Rajaei, R.; Tabandeh, M.; Shouraki, S. B. A Low-Cost Highly Reliable Spintronic True Random Number Generator Circuit for Secure Cryptography. In *SPIN*, 2020, 10(1). World Scientific Publishing Company, pp. 2050003.
- Avaroğlu, E. The implementation of ring oscillator based PUF designs in Field Programmable Gate Arrays using of different challenge. In *Physica A: Statistical Mechanics and its Applications*, 2020, pp. 124291.
- Garipcan, A. M.; Erdem, E. A TRNG using chaotic entropy pool as a post-processing technique: analysis, design and FPGA implementation. In *Analog Integrated Circuits and Signal Processing*, 2020, pp.1–20.

- Boke, A. K.; Nakhate, S.; Rajawat, A. Efficient Key Generation Techniques for Securing IoT Communication Protocols. In *IETE Technical Review*, 2020, pp. 1–12.
- Etem, T.; Kaya, T. A novel True Random Bit Generator design for image encryption. In *Physica A: Statistical Mechanics and its Applications*, 2020, 540, pp. 122750.
- Yakut, S.; Tuncer, T.; Ozer, A. B. Secure and Efficient Hybrid Random Number Generator Based on Sponge Constructions for Cryptographic Applications. In *Elektronika ir Elektrotechnika*, 2019, 25(4), pp. 40–46.
- Garipcan, A. M.; Erdem, E. Implementation of a Digital TRNG Using Jitter Based Multiple Entropy Source on FPGA. In *Informacije MIDEM*, 2019, 49(2), pp. 79–90.
- Arslan Tuncer, S.; Kaya, T. True random number generation from bioelectrical and physical signals. In *Computational and mathematical methods in medicine*, 2018.

[A.3] Kodýtek, F.; Lórencz, R. A design of ring oscillator based PUF on FPGA. In *18th IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems*. April 22–24, 2015 – Belgrade, Serbia.

The paper has been cited in:

- Gu, C.; Chang, C. H.; Liu, W.; Hanley, N.; Miskelly, J.; O'Neill, M. A Large Scale Comprehensive Evaluation of Single-Slice Ring Oscillator and PicoPUF Bit Cells on 28nm Xilinx FPGAs. In *Proceedings of the 3rd ACM Workshop on Attacks and Solutions in Hardware Security Workshop*, 2019, pp. 101–106.
- Ruiz-Rosero, J.; Ramirez-Gonzalez, G.; Khanna, R. Field Programmable Gate Array Applications-A Scientometric Review. In *Computation*, 2019, 7(4), pp. 63.
- Laban, M.; Drutarovsky, M.; Fischer, V.; Varchola, M. Modular evaluation platform for evaluation and testing of physically unclonable functions. In *2018 28th International Conference Radioelektronika (RADIOELEKTRONIKA)*. IEEE, 2018, pp. 1–6.
- Tan, T. H.; Ooi, C. Y.; Marsono, M. N. drDRM: A PUF-Based Dynamically Reconfigurable DRM Mechanism for FPGA-Based Platform. In *2018 Sixth International Symposium on Computing and Networking (CANDAR)*, 2018, pp. 194–200. ISSN 2379-1888.
- Pandey, A.; Pandey, S. Investigation of ROPUF with Improved Temperature Performance on FPGA. In *Helix*, 2018, 8(6), pp. 4334–4339. ISSN 2277-3495.

- Pramudita, R.; Ramadhan, S.; Hariadi, F. I.; Ahmad, A. S. Implementation Ring Oscillator Physical Unclonable Function (PUF) in FPGA. In *2018 International Symposium on Electronics and Smart Devices (ISESD)*. IEEE, 2018, pp. 1–5.
- Hesselbarth, R.; Wilde, F.; Gu, C.; Hanley, N. 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)*. IEEE, 2018, pp. 126–133.
- Satheesh, N.; Mahapatra, A.; Kumar, S.; Sahoo, S.; Mahapatra, K. K. A modified RO-PUF with improved security metrics on FPGA. In *2016 IEEE International Symposium on Nanoelectronic and Information Systems (iNIS)*, 2016, pp. 178–181. ISBN 978-1-5090-6170-9.

[A.4] Kodýtek, F.; Lórencz, R. Proposal and Properties of Ring Oscillator-Based PUF on FPGA, 2016. In *Journal of Circuits, Systems and Computers*. March 2016, Vol. 25, No. 03. ISSN 0218-1266.

The paper has been cited in:

- Xu Jinfu, X.; Wu Jin, X.; Li Junwei, X.; Qu Tongzhou, X.; Dong Yongxing, X. Controlled Physical Unclonable Function Research Based on Sensitivity Confusion Mechanism. In *Journal of Electronics & Information Technology*, 2019, 41(7), pp. 1601–1609. ISSN 1009-5896.

[A.5] Kodýtek, F.; Lórencz, R.; Buček, J. Improved ring oscillator PUF on FPGA and its properties. In *Microprocessors and Microsystems*. 2016, ISSN 0141-9331, http://dx.doi.org/10.1016/j.micpro.2016.02.005.

The paper has been cited in:

- Chauhan, A.; Sahula, V.; Mandal, A. Novel Randomized Placement for FPGA Based Robust ROPUF with Improved Uniqueness. In *Journal of Electronic Testing-theory and Applications*, 2019, 35(5), pp. 581–601. ISSN 0923-8174.
- Ruiz-Rosero, J.; Ramirez-Gonzalez, G.; Khanna, R. Field Programmable Gate Array Applications-A Scientometric Review. In *Computation*, 2019, 7(4), pp. 63.
- Xu Jinfu, X.; Wu Jin, X.; Li Junwei, X.; Qu Tongzhou, X.; Dong Yongxing, X. Controlled Physical Unclonable Function Research Based on Sensitivity Confusion Mechanism. In *Journal of Electronics & Information Technology*, 2019, 41(7), pp. 1601–1609. ISSN 1009-5896.
- Jaafar, A.; Soin, N.; Hatta, S. W. M.; Md, S. S. I. Delay performance due to thermal variation on field-programmable gate array via the adoption of a stable ring oscillator. In *IET Computers & Digital Techniques*, 2019, 13(5), pp. 405–413. ISSN 1751-8601.

- Xu Jinfu, X.; Wu Jin, X.Frequency Sorting Algorithm Based on Dynamic Ring Oscillator Physical Unclonable Function Statistical Model. In *Journal of Electronics & Information Technology*, 2019, 41(3), pp. 717–724. ISSN 1009-5896.
- Darvishi, M.; Audet, Y.; Blaquière, Y.; Thibeault, C.; Pichette, S. On the susceptibility of SRAM-based FPGA routing network to delay changes induced by ionizing radiation. In *IEEE Transactions on Nuclear Science*, 2019, 66(3), pp. 643–654. ISSN 0018-9499.
- Yan, W.; Chandy, J. (2018). Phase Calibrated Ring Oscillator PUF Design and Application. In *Computers*, 2018, 7(3), 40. ISSN 2073-431X.
- Wang, W.; Cui, A.; Qu, G.; Li, H. A low-overhead PUF based on parallel scan design. In *Proceedings of the 23rd Asia and South Pacific Design Automation Conference*, 2018, pp. 715–720. ISSN 2153-6961.
- Satheesh, N.; Mahapatra, A.; Kumar, S.; Sahoo, S.; Mahapatra, K. K. A modified RO-PUF with improved security metrics on FPGA. In *2016 IEEE International Symposium on Nanoelectronic and Information Systems (iNIS)*, 2016, pp. 178–181. ISBN 978-1-5090-6170-9.
- Loong, J. T. H.; Hashim, N. A. N.; Hamid, M. S.; Hamid, F. A. Performance analysis of CMOS-memristor hybrid ring oscillator Physically Unclonable Function (RO-PUF). In *2016 IEEE International Conference on Semiconductor Electronics (ICSE)*. IEEE, 2016, pp. 304–307.

[A.6] Kodýtek, F.; Lórencz, R.; Buček, J.; Buchovecká, S. Temperature dependence of ROPUF on FPGA. In *Euromicro Conference on Digital System Design* (Poster). August 31 – September 2, 2016 – Limassol, Cyprus.

The paper has been cited in:

- Xu Jinfu, X.; Wu Jin, X.Frequency Sorting Algorithm Based on Dynamic Ring Oscillator Physical Unclonable Function Statistical Model. In *Journal of Electronics & Information Technology*, 2019, 41(3), pp. 717–724. ISSN 1009-5896.

[A.7] Buchovecká, S.; Lórencz, R.; Buček, J.; Kodýtek, F. Lightweight Authentication and Secure Communication Suitable for IoT Devices. In *The International Conference on Information Systems Security and Privacy*. February 25–27, 2020, Malta, pp. 75–83.

[A.8] Kodýtek, F.; Lórencz, R.; Buček, J. Comparison of three counter value based RO-PUFs on FPGA. In *Euromicro Conference on Digital System Design*. August 26–28, 2020 – Portorož, Slovenia.

[A.9] Buchovecká, S.; Lórencz, R.; Buček, J.; Kodýtek, F. Symmetric and Asymmetric Schemes for Lightweight Secure Communication. In *Information Systems Security and Privacy*, Springer, 2020. [Submitted]

Remaining Publications of the Author Relevant to the Thesis

[A.10] Kodýtek, F. *Physical Unclonable Function on FPGAs*. Bachelor thesis. Czech Technical University in Prague, Faculty of Information Technology, 2014.

[A.11] Kodýtek, F.; Lórencz, R. A novel ring oscillator based PUF on FPGA. In *International Workshops on Cryptographic Architectures Embedded in Reconfigurable Devices*. June 29 – July 2, 2014 – Annecy, France.

[A.12] Kodýtek, F.; Lórencz, R. A novel ring oscillator based PUF on FPGA. In *Joint MEDIAN-TRUDEVICE Open Forum* (Poster). October 30, 2014 – Amsterdam, Netherlands.

[A.13] Kodýtek, F.; Lórencz, R.; Buček, J. Properties of improved ROPUF for FPGA generating multiple output bits from each pair of ROs. In *International Workshops on Cryptographic Architectures Embedded in Reconfigurable Devices*. June 28 – July 1, 2015 – Leuven, Belgium.

[A.14] Kodýtek, F. *Behaviour Analysis and Improvement of the Proposed PUF on FPGA*. Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2016.

[A.15] Kodýtek, F.; Lórencz, R.; Buček, J.; Buchovecká, S. Multiple output bits ROPUF design for TRNG. In *International Workshops on Cryptographic Architectures Embedded in Reconfigurable Devices*. June 21–24, 2016 – Montpellier, France.

The paper has been cited in:

- Habib, B.; Gaj, K. A comprehensive set of schemes for PUF response generation. In *Microprocessors and Microsystems*. 2017, ISSN 0141-9331, 51, pp. 239–251.

[A.16] Kodýtek, F.; Lórencz, R. A ring oscillator based PUF proposal on FPGA. In *Počítačové Architektury & Diagnostika*. September 14–16, 2016 – Bořetice, Czech Republic.

[A.17] Kodýtek, F.; Lórencz, R. A common design for PUF and TRNG based on ring oscillators. In *Počítačové Architektury & Diagnostika*. September 6–8, 2017 – Smolenice, Slovakia.

[A.18] Kodýtek, F.; Lórencz, R. Modeling of jitter of ring oscillators. In *Počítačové Architektury & Diagnostika*. September 4–6, 2019 – Doksy, Czech Republic.

Appendix A

Additional Experimental Results

For better readability, we did not present all of the results of our experiments related to Digilent Cmod S7 FPGA boards (Spartan-7) in Section 7.3. This appendix contains the complete set of the results of the evaluated measurements.

A.1 Evaluation of the positions of the counter values

This section contains the results of the evaluation of individual bit positions of the counter values measured on 20 Digilent Cmod S7 FPGA boards. The number of (non-interleaved) measurements was set to 5500 and the first 500 were skipped and used as a warm-up for the ROs. All three measurement methods (frequency ratio, frequency difference, crystal reference) are presented. The following Tables A.1, A.2 and A.3 show the results on the three different implementations (asymmetric ROs, all enabled asymmetric ROs, symmetric ROs).

	Frequency ratio				Frequency difference				Crystal reference			
position(i)	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$
1	0.9998	0.0029	0.0019	0.9995	0.9994	0.9641	0.0977	0.3897	0.9998	0.1040	0.0424	0.9833
2	0.9998	0.5048	0.0115	0.8882	0.9992	0.9808	0.1098	0.4197	0.9991	0.9402	0.5420	0.3717
3	0.9992	0.8909	0.0700	0.6915	0.9987	0.9963	0.1486	0.4678	0.9990	0.9882	0.5788	0.4547
4	0.9988	0.9515	0.1874	0.6273	0.9980	0.9962	0.2373	0.5321	0.9984	0.9951	0.7314	0.4767
5	0.9972	0.9970	0.4191	0.5064	0.9955	0.9947	0.4785	0.4698	0.9966	0.9891	0.9113	0.5098
6	0.9925	0.9793	0.8277	0.5759	0.9897	0.9972	0.8320	0.5010	0.9940	0.9912	0.9426	0.5147
7	**0.9841**	**0.9958**	**0.9577**	**0.5193**	**0.9812**	**0.9929**	**0.9563**	**0.4991**	**0.9881**	**0.9952**	**0.9641**	**0.4992**
8	**0.9654**	**0.9943**	**0.9614**	**0.4950**	**0.9618**	**0.9954**	**0.9605**	**0.4906**	**0.9744**	**0.9953**	**0.9660**	**0.4985**
9	**0.9265**	**0.9941**	**0.9653**	**0.5094**	**0.9258**	**0.9939**	**0.9577**	**0.5039**	**0.9453**	**0.9943**	**0.9644**	**0.4876**
10	0.8519	0.9977	0.9631	0.4996	0.8547	0.9948	0.9619	0.4943	0.8896	0.9962	0.9656	0.4969
11	0.7057	0.9936	0.9648	0.4999	0.7132	0.9941	0.9668	0.4969	0.7752	0.9924	0.9696	0.4981
12	0.5374	0.9946	0.9683	0.5006	0.5380	0.9961	0.9575	0.5002	0.5938	0.9964	0.9669	0.4982
13	0.5065	0.9943	0.9653	0.5000	0.5056	0.9927	0.9696	0.5000	0.5086	0.9936	0.9626	0.4999
14	0.5061	0.9935	0.9564	0.5005	0.5057	0.9957	0.9689	0.5000	0.5056	0.9961	0.9657	0.5000
15	0.5059	0.9965	0.9673	0.4997	0.5056	0.9960	0.9609	0.5002	0.5055	0.9962	0.9613	0.4997
16	0.5058	0.9963	0.9616	0.4998	0.5057	0.9978	0.9521	0.4997	0.5055	0.9973	0.9631	0.5000

Table A.1: Evaluation of individual bit positions of the counter values measured on 20 Digilent Cmod S7 FPGA boards for 150 RO pairs, 5000 measurements. All three measurement methods were used to measure the counter values. The ROs were mutually asymmetric.

	Frequency ratio				Frequency difference				Crystal reference			
position(i)	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$
1	0.9997	0.0000	0.0000	0.9997	0.9998	0.3713	0.0315	0.0716	1.0000	0.0570	0.0174	0.9923
2	0.9995	0.3804	0.0170	0.9260	0.9990	0.9888	0.1650	0.4468	0.9993	0.9027	0.5788	0.3512
3	0.9991	0.6663	0.0929	0.8253	0.9979	0.9972	0.3081	0.5214	0.9987	0.9824	0.6172	0.5408
4	0.9981	0.9734	0.2448	0.5926	0.9964	0.9955	0.5577	0.5208	0.9978	0.9779	0.7753	0.5820
5	0.9950	0.9873	0.5263	0.5595	0.9932	0.9935	0.9038	0.5051	0.9966	0.9883	0.9287	0.5365
6	0.9911	0.9933	0.8841	0.5125	**0.9856**	**0.9947**	**0.9541**	**0.5098**	0.9936	0.9950	0.9402	0.4816
7	**0.9822**	**0.9959**	**0.9553**	**0.5037**	**0.9706**	**0.9936**	**0.9609**	**0.5017**	**0.9861**	**0.9971**	**0.9675**	**0.5005**
8	**0.9606**	**0.9950**	**0.9641**	**0.5042**	**0.9416**	**0.9945**	**0.9597**	**0.5018**	**0.9714**	**0.9953**	**0.9549**	**0.5077**
9	**0.9227**	**0.9951**	**0.9570**	**0.5117**	0.8886	0.9929	0.9700	0.5009	**0.9410**	**0.9935**	**0.9685**	**0.4886**
10	0.8452	0.9967	0.9618	0.5040	0.7808	0.9936	0.9612	0.5038	0.8862	0.9973	0.9593	0.5104
11	0.6992	0.9944	0.9612	0.5024	0.5977	0.9939	0.9665	0.5021	0.7701	0.9951	0.9643	0.5038
12	0.5333	0.9950	0.9621	0.5005	0.5076	0.9980	0.9611	0.4998	0.5860	0.9935	0.9617	0.5025
13	0.5060	0.9954	0.9615	0.5000	0.5055	0.9936	0.9618	0.5001	0.5074	0.9976	0.9635	0.5000
14	0.5056	0.9961	0.9677	0.5001	0.5057	0.9954	0.9669	0.5000	0.5058	0.9970	0.9708	0.5002
15	0.5056	0.9963	0.9611	0.5000	0.5057	0.9973	0.9577	0.5000	0.5057	0.9944	0.9628	0.5001
16	0.5060	0.9879	0.9405	0.5007	0.5058	0.9929	0.9603	0.5001	0.5057	0.9953	0.9612	0.5000

Table A.2: Evaluation of individual bit positions of the counter values measured on 20 Digilent Cmod S7 FPGA boards for 150 RO pairs, 5000 measurements (non-interleaved). All three measurement methods were used to measure the counter values. The ROs were mutually asymmetric and they were all enabled during the measurements.

	Frequency ratio				Frequency difference				Crystal reference			
position(i)	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$	$s(b_i)$	$H_{intra}(b_i)$	$H_{inter}(b_i)$	$P(b_i = 1)$
1	0.9980	0.0000	0.0000	0.9980	0.9953	0.9578	0.4698	0.5548	1.0000	0.0000	0.0000	1.0000
2	0.9980	0.0000	0.0000	0.9980	0.9953	0.9578	0.4698	0.5548	1.0000	0.0000	0.0000	0.0000
3	0.9980	0.0600	0.0019	0.9910	0.9953	0.9578	0.4698	0.5548	0.9992	0.8616	0.6336	0.5741
4	0.9973	0.4752	0.0930	0.8950	0.9948	0.9571	0.4969	0.5653	0.9992	0.8633	0.6425	0.4152
5	0.9960	0.7822	0.2701	0.7646	0.9936	0.9551	0.6545	0.5950	0.9980	0.9823	0.9358	0.5502
6	0.9914	0.8863	0.6946	0.6905	0.9913	0.9602	0.8589	0.5649	0.9942	0.9238	0.9646	0.4384
7	0.9818	0.9771	0.9201	0.5772	**0.9836**	**0.9956**	**0.9548**	**0.5118**	**0.9875**	**0.9954**	**0.9626**	**0.4925**
8	**0.9620**	**0.9866**	**0.9525**	**0.5501**	**0.9688**	**0.9956**	**0.9604**	**0.4991**	**0.9742**	**0.9955**	**0.9649**	**0.4980**
9	**0.9250**	**0.9948**	**0.9533**	**0.5325**	**0.9375**	**0.9954**	**0.9640**	**0.4833**	**0.9479**	**0.9969**	**0.9579**	**0.4832**
10	0.8428	0.9962	0.9580	0.4975	0.8768	0.9955	0.9638	0.4958	0.8994	0.9961	0.9636	0.5122
11	0.6949	0.9972	0.9645	0.5080	0.7521	0.9968	0.9578	0.4912	0.7966	0.9933	0.9553	0.4949
12	0.5281	0.9965	0.9636	0.5016	0.5635	0.9950	0.9577	0.5011	0.6177	0.9970	0.9563	0.4996
13	0.5060	0.9933	0.9627	0.5004	0.5057	0.9963	0.9615	0.5002	0.5082	0.9918	0.9602	0.5000
14	0.5059	0.9961	0.9650	0.4994	0.5056	0.9956	0.9656	0.5000	0.5057	0.9963	0.9647	0.5001
15	0.5061	0.9945	0.9650	0.5006	0.5057	0.9961	0.9653	0.4999	0.5057	0.9933	0.9603	0.4998
16	0.5061	0.9913	0.9622	0.5007	0.5057	0.9950	0.9581	0.5001	0.5057	0.9962	0.9567	0.5002

Table A.3: Evaluation of individual bit positions of the counter values measured on 20 Digilent Cmod S7 FPGA boards for 150 RO pairs, 5000 measurements (non-interleaved). All three measurement methods were used to measure the counter values. The ROs were mutually symmetric.

A.2 PUF response evaluation

In the following subsections, we will present the results of the evaluation of the PUF responses made of various selections of bit positions. Again, the results will be presented for all three measurement methods using three different implementations. The evaluation was performed using 5500 non-interleaved measurements with the first 500 measurements skipped as a warm-up. First, we will show the evaluation of reliability(HD_{intra}) and uniqueness (HD_{inter}) followed by the evaluation of randomness.

A.2.1 Reliability and uniqueness

The following Tables A.4, A.5 and A.6 show the results of the evaluation of reliability and uniqueness of the PUF responses made of various selections of bit positions with either no Gray code used, they Gray code applied to the selected positions of the counter values, or the Gray code applied to the whole counter values.

Frequency ratio

	Asymmetric ROs					Asymmetric ROs, all enabled					Symmetric ROs				
positions	6–8	7–8	**7–9**	8–9	7–10	6–8	7–8	**7–9**	8–9	7–10	6–8	7–8	**7–9**	8–9	7–10
w [-]	3	2	**3**	2	4	3	2	**3**	2	4	3	2	**3**	2	4
HD_{intra} [%]	1.93	2.52	**4.13**	5.41	6.80	2.20	2.86	**4.48**	5.83	7.23	2.16	2.81	**4.37**	5.65	7.21
HD_{inter} [%]	47.00	49.77	**49.90**	50.04	49.94	48.15	49.78	**49.72**	49.83	49.76	43.32	48.18	**48.56**	49.30	48.84

Frequency difference

	Asymmetric ROs					Asymmetric ROs, all enabled					Symmetric ROs				
positions	6–8	7–8	**7–9**	8–9	7–10	6–8	**7–8**	7–9	8–9	7–10	6–8	7–8	**7–9**	8–9	7–10
w [-]	3	2	**3**	2	4	3	**2**	3	2	4	3	2	**3**	2	4
HD_{intra} [%]	2.24	2.85	**4.37**	5.62	6.91	3.41	**4.39**	6.64	8.49	10.46	1.88	2.38	**3.67**	4.69	5.83
HD_{inter} [%]	46.91	49.70	**49.68**	49.74	49.74	49.68	**49.82**	50.04	50.13	50.00	47.45	49.63	**49.78**	49.94	49.85

Crystal reference

	Asymmetric ROs					Asymmetric ROs, all enabled					Symmetric ROs				
positions	6–8	7–8	**7–9**	8–9	7–10	6–8	7–8	**7–9**	8–9	7–10	6–8	7–8	7–9	8–9	**7–10**
w [-]	3	2	**3**	2	4	3	2	**3**	2	4	3	2	3	2	**4**
HD_{intra} [%]	1.45	1.91	**3.12**	4.09	5.09	1.70	2.20	**3.48**	4.52	5.52	1.47	1.92	3.02	3.91	**4.85**
HD_{inter} [%]	49.35	49.81	**49.89**	50.00	49.95	49.44	50.17	**50.23**	50.11	50.25	49.55	49.76	49.84	49.89	**49.82**

Table A.4: Evaluation of the PUF responses for all of the implementation variants of the three PUF constructions using different selections of the positions of the measured counter values. Measurements were performed on 20 FPGAs using two sets of ROs, each consisting of 150 ROs. The evaluation was performed on 5000 repetitions of the measurements. No Gray code was applied to the counter values.

Frequency ratio

	Asymmetric ROs					Asymmetric ROs, all enabled					Symmetric ROs				
positions	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**
w [-]	3	2	3	2	**4**	3	2	3	2	**4**	3	2	3	2	**4**
HD_{intra} [%]	1.15	1.73	2.45	3.68	**3.70**	1.33	1.99	2.65	3.94	**4.00**	1.27	1.90	2.50	3.75	**3.93**
HD_{inter} [%]	44.99	49.71	49.64	49.70	**49.85**	46.84	49.85	50.01	50.21	**50.02**	43.85	48.47	48.89	49.50	**49.19**

Frequency difference

	Asymmetric ROs					Asymmetric ROs, all enabled					Symmetric ROs				
positions	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	**7–9**	8–9	7–10	6–8	7–8	7–9	8–9	**7–10**
w [-]	3	2	3	2	**4**	3	2	**3**	2	4	3	2	3	2	**4**
HD_{intra} [%]	1.27	1.91	2.47	3.71	**3.63**	1.97	2.95	**3.78**	5.64	5.64	1.04	1.56	2.08	3.13	**3.08**
HD_{inter} [%]	43.68	49.72	49.70	49.74	**49.80**	49.82	49.98	**50.02**	49.94	50.04	42.87	49.51	49.73	49.99	**49.75**

Crystal reference

	Asymmetric ROs					Asymmetric ROs, all enabled					Symmetric ROs				
positions	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**
w [-]	3	2	3	2	**4**	3	2	3	2	**4**	3	2	3	2	**4**
HD_{intra} [%]	0.88	1.33	1.85	2.77	**2.75**	0.99	1.49	2.03	3.04	**2.94**	0.86	1.30	1.74	2.61	**2.58**
HD_{inter} [%]	49.02	49.86	49.97	50.06	**50.07**	48.95	50.23	50.15	49.93	**50.11**	49.48	49.82	49.89	49.91	**49.93**

Table A.5: Evaluation of the PUF responses for all of the implementation variants of the three PUF constructions using different selections of the positions of the measured counter values. Measurements were performed on 20 FPGAs using two sets of ROs, each consisting of 150 ROs. The evaluation was performed on 5000 repetitions of the measurements. The Gray code was applied to the selected parts of the counter values.

Frequency ratio

	Asymmetric ROs					Asymmetric ROs, all enabled					Symmetric ROs				
positions	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**
w [-]	3	2	3	2	**4**	3	2	3	2	**4**	3	2	3	2	**4**
HD_{intra} [%]	1.06	1.36	2.20	2.88	**3.52**	1.16	1.55	2.35	3.08	**3.78**	1.13	1.47	2.21	2.84	**3.71**
HD_{inter} [%]	40.10	46.76	47.67	49.64	**48.37**	41.27	47.81	48.65	50.28	**49.00**	41.95	48.98	49.23	49.79	**49.45**

Frequency difference

	Asymmetric ROs					Asymmetric ROs, all enabled					Symmetric ROs				
positions	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	**7–9**	8–9	7–10	6–8	7–8	7–9	8–9	**7–10**
w [-]	3	2	3	2	**4**	3	2	**3**	2	4	3	2	3	2	**4**
HD_{intra} [%]	1.13	1.40	2.13	2.77	**3.37**	1.75	2.24	**3.31**	4.20	5.29	0.83	1.13	1.79	2.31	**2.86**
HD_{inter} [%]	37.35	44.85	46.44	49.77	**47.36**	48.58	50.04	**50.06**	50.10	50.07	32.58	42.77	45.24	49.87	**46.38**

Crystal reference

	Asymmetric ROs					Asymmetric ROs, all enabled					Symmetric ROs				
positions	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**	6–8	7–8	7–9	8–9	**7–10**
w [-]	3	2	3	2	**4**	3	2	3	2	**4**	3	2	3	2	**4**
HD_{intra} [%]	0.78	1.06	1.67	2.18	**2.62**	0.88	1.14	1.80	2.34	**2.76**	0.77	1.01	1.55	1.99	**2.44**
HD_{inter} [%]	45.90	49.31	49.59	50.12	**49.79**	45.76	49.43	49.62	49.99	**49.72**	43.73	49.65	49.78	49.97	**49.84**

Table A.6: Evaluation of the PUF responses for all of the implementation variants of the three PUF constructions using different selections of the positions of the measured counter values. Measurements were performed on 20 FPGAs using two sets of ROs, each consisting of 150 ROs. The evaluation was performed on 5000 repetitions of the measurements. The Gray code was applied to the whole counter values.

A.2.2 Randomness

As described in Chapter 4, we use the NIST STS 2.1.2 [52] for randomness evaluation. There are either 10 or 60 input sequences made of concatenation of the reference (average) PUF responses. These PUF responses were created using individual bits from the counter values or multiple bits for particular selections of bit positions. When multiple bits were extracted, the Gray code was applied to them. The evaluation was performed for the three measurement methods (frequency ratio, frequency difference, crystal reference) using the three different implementation variants (asymmetric ROs, all enabled asymmetric ROs, symmetric ROs). The following Tables from A.7 up to A.15 present the results of the randomness evaluation. The minimum allowed pass rate was 8/10 for 10 input sequences or 57/60 for 60 input sequences. Red cells indicate the failure of the test for the distribution of p-values.

positions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frequency	0/10	0/10	0/10	0/10		5/10			9/10		9/10					
BlockFrequency (m=20)	0/10	0/10	0/10	8/10												
BlockFrequency (m=30)	0/10	0/10	0/10	4/10		8/10							9/10			
CumulativeSums	0/10	0/10	0/10	0/10		5/10			9/10							
CumulativeSums	0/10	0/10	0/10	0/10		5/10			9/10							
Runs	0/10	0/10	0/10	2/10												
LongestRun	0/10	0/10	0/10	0/10		8/10						9/10		9/10		8/10
ApproximateEntropy (m=2)	0/10	0/10	0/10	0/10		6/10			9/10							
ApproximateEntropy (m=3)	0/10	0/10	0/10	0/10		9/10										

10 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	5/10	9/10		8/10		
BlockFrequency (m=20)		9/10				
BlockFrequency (m=30)	8/10					
CumulativeSums	5/10			8/10		
CumulativeSums	5/10			7/10		
Runs			9/10			
LongestRun						
ApproximateEntropy (m=2)	7/10			9/10	9/10	
ApproximateEntropy (m=3)	8/10				9/10	

60 sequences

6–7	7–8	8–9	6–8	7–9	7–10
57/60	58/60		59/60	59/60	59/60
59/60	59/60		59/60		
58/60		59/60		59/60	
58/60			59/60		59/60
59/60			59/60	59/60	
58/60		57/60		59/60	
0/60	0/60	0/60	57/60		
59/60	59/60	59/60	59/60	59/60	
	59/60	59/60	59/60	59/60	

Table A.7: Evaluation of randomness of the PUF responses composed of various selections of bit positions for asymmetric ROs, frequency ratio measurement method. During the selection of multiple bits from each counter value (bottom table), the Gray code was applied to the selected bits. Average responses were used for the evaluation.

positions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frequency	0/10	1/10														
BlockFrequency (m=20)	9/10							9/10								
BlockFrequency (m=30)	7/10															
CumulativeSums	0/10	0/10														
CumulativeSums	0/10	7/10														
Runs	8/10				8/10											
LongestRun																
ApproximateEntropy (m=2)	0/10	5/10			9/10							9/10	9/10			
ApproximateEntropy (m=3)	1/10		9/10									9/10				

10 sequences

60 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10	6–7	7–8	8–9	6–8	7–9	7–10
Frequency								59/60				
BlockFrequency (m=20)							59/60		59/60			58/60
BlockFrequency (m=30)									59/60			
CumulativeSums												
CumulativeSums											59/60	
Runs									59/60			
LongestRun							0/60	0/60	0/60	59/60	58/60	
ApproximateEntropy (m=2)									59/60		59/60	59/60
ApproximateEntropy (m=3)		9/10							58/60	59/60	59/60	58/60

Table A.8: Evaluation of randomness of the PUF responses composed of various selections of bit positions for asymmetric ROs, frequency difference measurement method. During the selection of multiple bits from each counter value (bottom table), the Gray code was applied to the selected bits. Average responses were used for the evaluation.

positions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frequency	0/10	2/10	7/10		8/10	9/10							9/10			
BlockFrequency (m=20)	0/10	0/10			9/10	9/10										
BlockFrequency (m=30)	0/10	0/10			8/10											
CumulativeSums	0/10	2/10	7/10		8/10	9/10							9/10			
CumulativeSums	0/10	0/10	7/10		9/10	9/10										
Runs	0/10	5/10													9/10	
LongestRun	0/10	3/10	9/10													
ApproximateEntropy (m=2)	0/10	4/10	9/10		9/10											
ApproximateEntropy (m=3)	0/10	5/10	8/10													

10 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	8/10			9/10		
BlockFrequency (m=20)						
BlockFrequency (m=30)						
CumulativeSums	7/10			8/10		
CumulativeSums	9/10			9/10		
Runs						
LongestRun						
ApproximateEntropy (m=2)						
ApproximateEntropy (m=3)			9/10			

60 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	58/60		59/60	59/60		
BlockFrequency (m=20)		59/60	59/60			
BlockFrequency (m=30)	58/60		58/60		59/60	
CumulativeSums	58/60		59/60	58/60		
CumulativeSums	58/60		58/60	59/60		
Runs	59/60		59/60		59/60	
LongestRun	0/60	0/60	0/60	58/60		
ApproximateEntropy (m=2)	58/60					59/60
ApproximateEntropy (m=3)	59/60		58/60		58/60	59/60

Table A.9: Evaluation of randomness of the PUF responses composed of various selections of bit positions for asymmetric ROs, crystal reference measurement method. During the selection of multiple bits from each counter value (bottom table), the Gray code was applied to the selected bits. Average responses were used for the evaluation.

positions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frequency	0/10	0/10	0/10	2/10	9/10				9/10							7/10
BlockFrequency (m=20)	0/10	0/10	0/10	3/10			9/10						9/10			
BlockFrequency (m=30)	0/10	0/10	0/10	1/10			9/10		9/10							
CumulativeSums	0/10	0/10	0/10	1/10	9/10				9/10							8/10
CumulativeSums	0/10	0/10	0/10	1/10	9/10			9/10	9/10							8/10
Runs	0/10	0/10	0/10	9/10		9/10	9/10									
LongestRun	0/10	0/10	0/10	6/10	6/10											
ApproximateEntropy (m=2)	0/10	0/10	0/10	3/10	9/10		8/10									
ApproximateEntropy (m=3)	0/10	0/10	0/10	7/10	8/10	9/10					9/10					

10 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency						
BlockFrequency (m=20)	9/10					
BlockFrequency (m=30)	9/10					
CumulativeSums						
CumulativeSums			9/10			
Runs						9/10
LongestRun						
ApproximateEntropy (m=2)						
ApproximateEntropy (m=3)				9/10		

60 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	59/60	59/60	59/60			
BlockFrequency (m=20)	58/60	59/60				
BlockFrequency (m=30)	59/60	57/60		59/60		
CumulativeSums	59/60	59/60				
CumulativeSums	59/60	59/60				
Runs	59/60		59/60	58/60		59/60
LongestRun	0/60	0/60	0/60	59/60		
ApproximateEntropy (m=2)	58/60			59/60		
ApproximateEntropy (m=3)	57/60			59/60	59/60	

Table A.10: Evaluation of randomness of the PUF responses composed of various selections of bit positions for asymmetric ROs (all enabled), frequency ratio measurement method. During the selection of multiple bits from each counter value (bottom table), the Gray code was applied to the selected bits. Average responses were used for the evaluation.

positions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frequency	0/10	8/10											9/10			
BlockFrequency (m=20)	0/10	4/10														
BlockFrequency (m=30)	0/10	3/10					9/10							9/10		
CumulativeSums	0/10	9/10					9/10						9/10			
CumulativeSums	0/10					9/10							9/10			
Runs	0/10															9/10
LongestRun	0/10	9/10			9/10	9/10						9/10				
ApproximateEntropy (m=2)	0/10				9/10											
ApproximateEntropy (m=3)	0/10	6/10					9/10									

10 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency		9/10				9/10
BlockFrequency (m=20)						
BlockFrequency (m=30)						
CumulativeSums		9/10			9/10	9/10
CumulativeSums						9/10
Runs			9/10			
LongestRun		9/10				9/10
ApproximateEntropy (m=2)			9/10			9/10
ApproximateEntropy (m=3)						

60 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency		59/60	58/60		59/60	58/60
BlockFrequency (m=20)					59/60	
BlockFrequency (m=30)			59/60			59/60
CumulativeSums		59/60	59/60		59/60	58/60
CumulativeSums		59/60	59/60		59/60	58/60
Runs	59/60	59/60			57/60	
LongestRun	0/60	0/60	0/60		58/60	59/60
ApproximateEntropy (m=2)	59/60				58/60	59/60
ApproximateEntropy (m=3)	59/60	59/60			58/60	

Table A.11: Evaluation of randomness of the PUF responses composed of various selections of bit positions for asymmetric ROs (all enabled), frequency difference measurement method. During the selection of multiple bits from each counter value (bottom table), the Gray code was applied to the selected bits. Average responses were used for the evaluation.

positions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frequency	0/10	1/10	7/10	3/10	9/10											
BlockFrequency (m=20)	0/10	0/10	7/10	9/10	9/10											
BlockFrequency (m=30)	0/10	0/10	6/10	9/10	9/10	9/10										
CumulativeSums	0/10	0/10	7/10	3/10	7/10											
CumulativeSums	0/10	2/10	6/10	2/10	9/10											
Runs	0/10	4/10									9/10					
LongestRun	0/10	3/10	8/10	9/10			9/10									
ApproximateEntropy (m=2)	0/10	0/10	8/10	8/10												
ApproximateEntropy (m=3)	0/10	0/10	7/10	8/10												9/10

10 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency						
BlockFrequency (m=20)						9/10
BlockFrequency (m=30)						9/10
CumulativeSums					9/10	9/10
CumulativeSums						
Runs					9/10	9/10
LongestRun						9/10
ApproximateEntropy (m=2)	9/10		9/10		9/10	
ApproximateEntropy (m=3)					9/10	9/10

60 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	59/60	59/60				59/60
BlockFrequency (m=20)				59/60		
BlockFrequency (m=30)			59/60			
CumulativeSums						59/60
CumulativeSums	59/60					58/60
Runs	58/60	58/60	59/60	59/60		58/60
LongestRun	0/60	0/60	0/60		59/60	57/60
ApproximateEntropy (m=2)	59/60				59/60	58/60
ApproximateEntropy (m=3)	59/60	59/60				

Table A.12: Evaluation of randomness of the PUF responses composed of various selections of bit positions for asymmetric ROs (all enabled), crystal reference measurement method. During the selection of multiple bits from each counter value (bottom table), the Gray code was applied to the selected bits. Average responses were used for the evaluation.

positions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frequency	0/10	0/10	0/10	0/10	0/10	0/10	2/10	6/10								9/10
BlockFrequency (m=20)	0/10	0/10	0/10	0/10	0/10	0/10		9/10								
BlockFrequency (m=30)	0/10	0/10	0/10	0/10	0/10	0/10	9/10									
CumulativeSums	0/10	0/10	0/10	0/10	0/10	0/10	3/10	6/10								9/10
CumulativeSums	0/10	0/10	0/10	0/10	0/10	0/10	2/10	6/10					9/10			9/10
Runs	0/10	0/10	0/10	0/10	0/10	0/10	9/10									
LongestRun	0/10	0/10	0/10	0/10	0/10	0/10	4/10									
ApproximateEntropy (m=2)	0/10	0/10	0/10	0/10	0/10	0/10	8/10	8/10								9/10
ApproximateEntropy (m=3)	0/10	0/10	0/10	0/10	0/10	0/10	8/10	8/10						9/10	9/10	

10 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	0/10	8/10	9/10	3/10	8/10	9/10
BlockFrequency (m=20)	6/10			8/10		
BlockFrequency (m=30)	6/10	9/10		8/10		
CumulativeSums	0/10	8/10	8/10	4/10	8/10	9/10
CumulativeSums	0/10	8/10	9/10	2/10	8/10	9/10
Runs	0/10	7/10		8/10	9/10	9/10
LongestRun	7/10			8/10		
ApproximateEntropy (m=2)	0/10	8/10		1/10	9/10	8/10
ApproximateEntropy (m=3)	0/10	8/10		2/10	9/10	9/10

60 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	45/60	57/60		52/60	58/60	58/60
BlockFrequency (m=20)	57/60	59/60		57/60		59/60
BlockFrequency (m=30)	54/60	59/60		56/60	59/60	59/60
CumulativeSums	51/60	59/60		53/60	58/60	57/60
CumulativeSums	51/60	59/60		52/60	58/60	58/60
Runs	53/60	58/60	59/60	59/60	58/60	58/60
LongestRun	0/60	0/60	0/60	59/60	59/60	58/60
ApproximateEntropy (m=2)	47/60	58/60	59/60	57/60	59/60	
ApproximateEntropy (m=3)	51/60	56/60	59/60	57/60	59/60	

Table A.13: Evaluation of randomness of the PUF responses composed of various selections of bit positions for symmetric ROs, frequency ratio measurement method. During the selection of multiple bits from each counter value (bottom table), the Gray code was applied to the selected bits. Average responses were used for the evaluation.

positions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frequency	6/10	6/10	6/10	6/10	3/10	5/10						9/10	9/10			
BlockFrequency (m=20)	0/10	0/10	0/10	0/10	5/10	2/10										
BlockFrequency (m=30)	0/10	0/10	0/10	0/10	4/10	2/10										
CumulativeSums	3/10	3/10	3/10	3/10	4/10	5/10						9/10	9/10			
CumulativeSums	4/10	4/10	4/10	4/10	3/10	3/10						9/10				
Runs	3/10	3/10	3/10	4/10	7/10	8/10										
LongestRun	6/10	6/10	6/10	6/10	6/10	9/10			9/10							
ApproximateEntropy (m=2)	3/10	3/10	3/10	3/10	5/10	3/10						8/10	9/10			9/10
ApproximateEntropy (m=3)	0/10	0/10	0/10	0/10	6/10	5/10						9/10	9/10			9/10

10 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	6/10			7/10	9/10	9/10
BlockFrequency (m=20)	5/10			6/10		
BlockFrequency (m=30)	5/10			4/10		
CumulativeSums	6/10			6/10	9/10	9/10
CumulativeSums	4/10			6/10	9/10	9/10
Runs	7/10			8/10		
LongestRun	7/10			8/10		
ApproximateEntropy (m=2)	2/10			7/10		
ApproximateEntropy (m=3)	4/10			7/10		

60 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	44/60	59/60	59/60	45/60		
BlockFrequency (m=20)	52/60		58/60	51/60		59/60
BlockFrequency (m=30)	50/60	58/60		54/60		
CumulativeSums	49/60	59/60	59/60	49/60		59/60
CumulativeSums	49/60	59/60	59/60	49/60		
Runs	53/60			57/60	57/60	59/60
LongestRun	0/60	0/60	0/60	57/60		
ApproximateEntropy (m=2)	45/60		58/60	50/60	59/60	59/60
ApproximateEntropy (m=3)	52/60	59/60	59/60	50/60		59/60

Table A.14: Evaluation of randomness of the PUF responses composed of various selections of bit positions for symmetric ROs, frequency difference measurement method. During the selection of multiple bits from each counter value (bottom table), the Gray code was applied to the selected bits. Average responses were used for the evaluation.

positions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frequency	0/10	0/10	1/10	0/10	7/10	3/10							9/10			
BlockFrequency (m=20)	0/10	0/10	2/10	4/10		2/10							9/10			
BlockFrequency (m=30)	0/10	0/10	0/10	2/10		2/10									9/10	
CumulativeSums	0/10	0/10	1/10	0/10	7/10	3/10							9/10			
CumulativeSums	0/10	0/10	0/10	0/10	7/10	3/10							8/10			
Runs	0/10	0/10	2/10	2/10	0/10	2/10	9/10			9/10						
LongestRun	0/10	0/10	0/10	1/10	3/10	5/10							9/10			
ApproximateEntropy (m=2)	0/10	0/10	0/10	0/10	0/10	2/10										
ApproximateEntropy (m=3)	0/10	0/10	0/10	0/10	0/10	2/10										

10 sequences

positions	6–7	7–8	8–9	6–8	7–9	7–10
Frequency	1/10			4/10		
BlockFrequency (m=20)	3/10			8/10		9/10
BlockFrequency (m=30)	2/10			5/10		9/10
CumulativeSums	2/10			4/10		
CumulativeSums	1/10		9/10	4/10		
Runs	4/10			6/10		
LongestRun	0/10			7/10		
ApproximateEntropy (m=2)	0/10			3/10		
ApproximateEntropy (m=3)	0/10			1/10		

60 sequences

6–7	7–8	8–9	6–8	7–9	7–10
37/60		59/60	47/60		59/60
46/60			54/60	59/60	58/60
44/60		59/60	53/60		
43/60			49/60		
40/60		59/60	50/60		59/60
49/60			51/60		58/60
0/60	0/60	0/60	56/60		59/60
39/60		59/60	50/60		58/60
37/60	59/60	58/60	47/60		

Table A.15: Evaluation of randomness of the PUF responses composed of various selections of bit positions for symmetric ROs, crystal reference measurement method. During the selection of multiple bits from each counter value (bottom table), the Gray code was applied to the selected bits. Average responses were used for the evaluation.

A.3 Influence of voltage and temperature

In Fig. 7.13, 7.14 and 7.15 we showed the results of the evaluation of PUF response stability for the three measurement methods at varying temperature and voltage for both asymmetric and symmetric ROs. The PUF responses were made of two selections of bit positions - 7–8 and 7–9 with the Gray code applied to them. These responses were obtained at each temperature and voltage level after some waiting time used to stabilize the ROs. After the warm-up, 5500 non-interleaved measurements were performed with the first 500 measurements skipped for evaluation purposes. Tables A.16 and A.17 show the results of this evaluation of reliability on asymmetric and symmetric ROs.

Frequency ratio

Positions 7–8

	0.92V	0.96V	1.00V	1.04V	1.08V
0°C	49.52	42.45	29.62	17.58	38.83
10°C	50.43	40.88	19.89	26.95	40.33
20°C	48.84	40.99	**9.54**	33.88	42.63
30°C	46.32	39.06	**1.23**	37.77	44.82
40°C	42.84	36.69	**9.32**	39.37	46.85
50°C	41.58	28.63	19.05	40.79	48.66
60°C	39.99	20.50	29.09	42.96	50.46

Positions 7–9

	0.92V	0.96V	1.00V	1.04V	1.08V
0°C	49.91	45.51	33.98	23.52	43.24
10°C	51.90	44.68	24.98	31.71	42.59
20°C	50.34	44.13	13.41	36.50	43.03
30°C	47.57	41.84	**2.07**	40.23	46.04
40°C	44.80	38.86	13.07	43.18	48.98
50°C	44.08	32.75	24.87	44.71	49.99
60°C	44.09	25.28	33.63	44.39	50.87

Frequency difference

Positions 7–8

	0.92V	0.96V	1.00V	1.04V	1.08V
0°C	51.15	47.48	25.23	48.79	49.23
10°C	51.15	47.69	17.52	49.33	49.70
20°C	51.26	49.52	**9.18**	49.59	50.42
30°C	50.88	49.53	**1.90**	48.70	52.53
40°C	48.79	48.77	**9.19**	48.08	52.91
50°C	47.15	47.06	17.89	46.87	53.83
60°C	47.02	46.35	26.84	45.42	53.58

Positions 7–9

	0.92V	0.96V	1.00V	1.04V	1.08V
0°C	49.50	48.70	32.45	48.43	49.51
10°C	49.64	49.15	23.86	49.02	50.14
20°C	49.69	51.17	12.61	49.54	51.21
30°C	50.07	50.76	**2.38**	48.75	52.28
40°C	48.56	48.86	11.61	48.57	52.11
50°C	47.10	47.30	23.12	48.11	52.85
60°C	47.10	46.82	32.10	47.63	51.63

Crystal reference

Positions 7–8

	0.92V	0.96V	1.00V	1.04V	1.08V
0°C	52.19	52.80	46.98	48.27	50.93
10°C	50.32	50.09	29.24	48.31	49.53
20°C	47.88	50.68	14.09	47.63	49.06
30°C	52.04	52.38	**1.23**	47.20	49.29
40°C	50.02	50.57	11.84	46.87	49.81
50°C	50.86	45.66	20.62	47.78	51.70
60°C	51.58	43.54	26.72	49.08	51.00

Positions 7–9

	0.92V	0.96V	1.00V	1.04V	1.08V
0°C	50.89	51.88	49.71	49.59	50.42
10°C	50.76	49.80	37.69	49.60	49.81
20°C	48.59	50.58	18.95	49.14	49.45
30°C	51.13	51.41	**1.71**	48.57	49.27
40°C	49.79	51.03	16.31	47.65	50.42
50°C	51.18	47.45	26.43	47.33	50.56
60°C	50.29	44.69	32.97	48.83	51.08

Table A.16: Evaluation of the stability of the PUF responses for the three measurement methods obtained from 5 Digilent Cmod S7 FPGA boards, 5000 measurements, asymmetric ROs. The values in the table are the values of the HD_{intra} represented in [%]. Blue coloured cells indicate the reference environmental conditions (1.00V and 30°C).

Frequency ratio

Positions 7–8

	0.92V	0.96V	1.00V	1.04V	1.08V
0°C	33.15	22.01	**9.84**	**7.97**	17.76
10°C	30.93	19.75	**6.89**	**8.61**	19.09
20°C	28.95	16.08	**3.90**	**10.53**	20.52
30°C	26.73	14.01	**1.65**	12.76	22.37
40°C	25.27	11.97	**4.34**	14.50	24.19
50°C	23.20	**10.28**	**7.05**	17.46	25.36
60°C	20.93	**8.40**	**9.73**	19.83	27.58

Positions 7–9

	0.92V	0.96V	1.00V	1.04V	1.08V
0°C	37.53	28.01	13.66	11.35	22.69
10°C	36.12	25.47	**9.46**	12.18	24.07
20°C	34.29	21.58	**5.73**	14.46	25.38
30°C	32.48	18.85	**2.31**	16.97	27.09
40°C	31.28	16.25	**6.26**	18.97	29.06
50°C	29.08	14.03	**9.78**	21.92	29.57
60°C	26.78	11.70	13.56	24.75	31.54

Frequency difference

Positions 7–8

	0.92V	0.96V	1.00V	1.04V	1.08V
0°C	31.70	22.44	**5.94**	20.20	32.85
10°C	31.53	21.41	**4.17**	19.33	31.56
20°C	31.21	20.50	**2.88**	18.75	31.31
30°C	31.54	20.07	**1.35**	18.23	30.90
40°C	31.69	19.48	**3.36**	18.28	30.22
50°C	31.56	18.75	**4.47**	19.06	30.19
60°C	31.83	19.18	**6.68**	19.35	29.37

Positions 7–9

	0.92V	0.96V	1.00V	1.04V	1.08V
0°C	35.24	25.43	**9.35**	25.72	36.10
10°C	34.66	24.95	**6.51**	24.62	34.54
20°C	34.21	24.19	**4.18**	23.72	34.07
30°C	34.33	23.81	**2.13**	23.08	33.27
40°C	34.39	23.80	**4.44**	23.01	32.56
50°C	33.88	23.08	**6.46**	23.46	32.22
60°C	34.34	23.86	**9.52**	23.55	31.76

Crystal reference

Positions 7–8

	0.92V	0.96V	1.00V	1.04V	1.08V
0°C	47.91	31.58	27.65	65.49	54.11
10°C	48.76	38.39	16.26	62.94	52.91
20°C	52.94	59.47	**8.12**	55.42	49.13
30°C	51.59	71.72	**1.21**	44.97	44.21
40°C	47.72	70.31	**5.67**	32.38	44.27
50°C	47.79	56.05	**10.57**	25.99	53.18
60°C	52.70	41.43	14.44	32.47	57.33

Positions 7–9

	0.92V	0.96V	1.00V	1.04V	1.08V
0°C	47.34	37.58	35.54	61.44	55.85
10°C	47.46	43.38	22.40	59.31	51.32
20°C	55.13	56.51	**10.83**	54.39	47.26
30°C	47.16	63.03	**1.50**	47.83	48.24
40°C	52.82	63.84	**7.13**	38.43	45.19
50°C	43.98	55.61	13.08	32.09	51.63
60°C	55.40	44.19	17.87	38.52	55.87

Table A.17: Evaluation of the stability of the PUF responses for the three measurement methods obtained from 5 Digilent Cmod S7 FPGA boards, 5000 measurements, symmetric ROs. The values in the table are the values of the HD_{intra} represented in [%]. Blue coloured cells indicate the reference environmental conditions (1.00V and 30°C).