Chapter 19
Physical Security Primitives

A Survey on Physically Unclonable Functions and PUF-Based Security Solutions

Ahmad-Reza Sadeghi, SteffenSchulz and ChristianWachsmann

Abstract Physically unclonable functions (PUFs) are an emerging technology and have been proposed as central building blocks in a variety of cryptographic protocols and security architectures. Among others, PUFs enable unique device identification and authentication, binding software to hardware platforms and secure storage of cryptographic secrets. Furthermore, they can be directly integrated into cryptographic algorithms and remote attestation protocols. In this chapter, we give an overview of the concept, properties, and types of intrinsic electronic PUFs, discuss potential attack surfaces and advanced PUF concepts as well as the most common applications of electronic PUFs. Further, we show new directions on logically reconfigurable PUFs (LR-PUFs) and PUF-based remote attestation and discuss open challenges.

19.1 Introduction

Physically unclonable functions (PUFs) are increasingly proposed as central building blocks in cryptographic protocols and higher level security architectures. Among others, PUFs enable unique device identification and authentication [44, 47, 54, 62], binding software to hardware platforms [12, 16, 18, 29], and secure storage of cryptographic secrets [34, 70]. Furthermore, they can be integrated into cryptographic

A.-R. Sadeghi (✉)
TU Darmstadt (CASED) and Fraunhofer SIT, Mornewegstraße 32, 64293 Darmstadt, Germany
e-mail: ahmad.sadeghi@trust.cased.de

S. Schulz
TU Darmstadt (CASED) and Macquarie University (INSS), Mornewegstraße 32, 64293 Darmstadt, Germany
e-mail: steffen.schulz@trust.cased.de

C. Wachsmann
TU Darmstadt (CASED), Mornewegstraße 32, 64293 Darmstadt, Germany
e-mail: christian.wachsmann@trust.cased.de

K. Markantonakis and K. Mayes (eds.), *Secure Smart Embedded Devices, Platforms and Applications*, DOI: 10.1007/978-1-4614-7915-4_19,

algorithms [2] and remote attestation protocols [55]. Today, there are already some PUF-based security products aimed for the market, mainly targeting IP-protection and anti-counterfeiting applications but also RFID systems [24, 67].

PUFs typically exhibit a challenge/response behavior: when queried with a specific *challenge*, the PUF generates a random-looking *response* that is stable over time. The security of PUFs depends on intrinsic manufacturing variations making PUFs physically unclonable and unpredictable. Even the manufacturer of the PUF should be unable to produce two PUFs with a similar challenge/response behavior. Furthermore, knowledge of a certain number of challenge/response pairs should not allow an adversary to predict PUF responses to unknown challenges.

There is a variety of PUF implementations [37]. The most appealing ones for integration into electronic circuits are *electronic PUFs*, which come in different flavors. *Delay-based PUFs*, such as arbiter PUFs [31, 35, 44] and ring oscillator PUFs [15, 38, 61] are based on race conditions or frequency variations in integrated circuits. *Memory-based PUFs* exploit the instability of volatile memory cells, such as SRAM cells [16, 21], flip-flops [32, 36], and latches [29, 60]. Finally, *coating PUFs* [46, 63, 64] use capacitances of a special dielectric coating applied to the chip housing the PUF.

In contrast to most cryptographic primitives, whose security can be related to well-established (albeit unproven) assumptions, the security of PUFs is assumed to rely on physical properties and is still under investigation. Existing PUF-based security solutions typically rely on assumptions that have not been confirmed for all PUF types. For instance, most delay-based PUFs have been shown to be susceptible to model building attacks that allow emulating the PUF in software [31, 35, 44, 51], which contradicts the unpredictability and unclonability properties. To counter this problem, additional primitives must be used: controlled PUFs [14] use cryptography in hardware to hide the responses of the underlying PUF from an adversary.

Since PUF responses are inherently noisy, they must be combined with error-correction mechanisms, such as fuzzy extractors [11] that remove the effects of noise before the PUF response can be processed in a (cryptographic) algorithm. Typically, the cryptographic and error correcting components and the connecting wires between them and the PUF must be protected against invasive and side-channel attacks.

Outline. This chapter gives an overview of the concept, properties, and types of intrinsic electronic PUFs (Sect. 19.2), discusses potential attack surfaces (Sect. 19.3), and advanced PUF concepts (Sect. 19.4) as well as the most common applications of PUFs (Sect. 19.5). Further, we show new directions on reconfigurable PUFs and PUF-based remote attestation (Sect. 19.6) and discuss open challenges (Sect. 19.7).

19.2 Physically Unclonable Functions

19.2.1 PUF Concept and Properties

A physically unclonable function (PUF) is a noisy function that is embedded into a physical object, such as an integrated circuit [1, 37, 45]. When queried with a *challenge* c, a PUF generates a *response* $r \leftarrow \text{PUF}(c)$ that depends on both c and the unique device-specific intrinsic physical properties of the object containing the PUF. Since PUFs are subject to noise induced by environmental variations, such as supply voltage and ambient temperature variations, they return slightly different responses when queried with the same challenge multiple times.

PUFs are typically assumed to be *robust*, *physically unclonable*, *unpredictable* and *tamper-evident*, and several approaches to quantify and formally define their properties have been proposed (see the paper by Armknecht et al. [1] for an overview). Informally, robustness means that, when queried with the same challenge multiple times, the PUF returns a similar response with high probability. Physical unclonability demands that it is infeasible to produce two PUFs that cannot be distinguished based on their challenge/response behavior. Unpredictability requires that it is infeasible to predict the PUF response to an unknown challenge, even if the PUF can be adaptively queried for a certain number of times. Finally, a PUF is tamper-evident if any attempt to physically access the PUF irreversibly changes its challenge/response behavior significantly.

The properties of PUFs can either be evaluated theoretically, based on mathematical models of the underling physical processes [66, 68, 69], or experimentally by analyzing PUF instances built in hardware [19, 22, 23, 32, 65]. The first approach has the apparent drawback that mathematical models never capture physical reality in its full extent, which means that the conclusions on PUF security drawn by this approach are naturally debatable. The main drawback of the experimental approach is its limited reproducibility and openness: even though experimental results have been reported in literature for some PUF implementations, it is difficult to compare them due to varying test conditions and different analysis methods. Furthermore, raw PUF data is rarely available for subsequent research, which greatly hinders a fair comparison.

The security analysis of PUFs is further complicated by the drawbacks of existing approaches to formalize their security properties. Most PUF security models are not general enough and exclude certain PUF types (such as in [15, 45]), do not reflect all properties of real PUF implementations (for example in [2, 15, 16, 45, 52]), or include security parameters that cannot be determined for real PUF implementations in practice (such as in [2, 8, 52]). Recently, Armknecht et al. [1] proposed a PUF security framework that aims at providing security definitions that are compliant to standard game-based cryptographic security models and that allow engineers to evaluate and quantify the properties of PUF implementations.

19.2.2 PUF Types

There is a broad variety of PUF implementations that are based on very different physical characteristics, including optical, magnetic, and electrical effects. We focus on *electronic PUFs*, which can be easily integrated into electronic circuits without significant overhead using standard manufacturing processes. These PUFs are of particular interest since they enable the tight integration of PUFs into cryptographic primitives and higher level security architectures. Known electronic PUFs can be categorized as *delay-based PUFs* and *memory-based PUFs*, which will be explained in the following. A more detailed overview of various PUF types, including non-electronic PUFs, is given by Maes and Verbauwhede [37].

19.2.2.1 Delay-Based PUFs

Delay-based PUFs are based on race conditions or frequency variations in integrated circuits. The most popular PUFs of this type are arbiter PUFs [31, 35, 44] and ring oscillator PUFs [15, 38, 61].

Arbiter PUFs. The arbiter PUF is based on race conditions within integrated circuits. The basic arbiter PUF design has been presented by Lim et al. [31, 33] and consists of two identically designed signal paths consisting of wires and switching components and an arbiter at the end of both paths (Fig. 19.1).

The switching components allow the signal paths to be modified according to an external input, i.e., the PUF challenge. To evaluate the arbiter PUF, both paths are simultaneously excited with the same impulse signal. Depending on which of the two signals arrives first at the arbiter, one output bit is generated and used as PUF response. The delay caused by each signal path depends on the device-specific manufacturing variations of the transistors in the switching components and their connecting wires.

In addition, the delays of the signal paths are affected by environmental noise, such as temperature and supply voltage variations. The impact of noise is reduced by comparing the delays of both signal paths. Note that, in case the delay difference

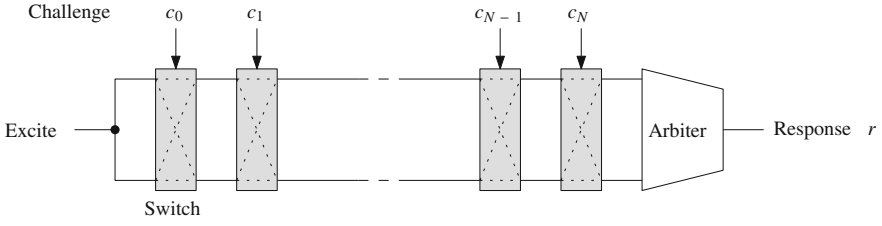


Fig. 19.1 Basic arbiter PUF design (for N bit challenge and 1 bit response)

between both signal paths is lower than the setup time of the arbiter, the response bit is independent of the delays and determined only by random noise (metastability).

Arbiter PUFs can be efficiently implemented on ASIC [31, 33], while implementations on FPGA seem to be difficult due to placing and routing constraints [37]. Moreover, the delays of the individual components of the signal paths are additive, which facilitates emulating the challenge/response behavior of the arbiter PUF in software (Sect. 19.3.1). To thwart these attacks, several variations of the basic arbiter PUF design have been proposed. The feed-forward arbiter PUF by Lim et al. [31, 33] uses the challenge and the output of intermediate arbiters to configure the signal paths. However, this design does not prevent emulation attacks [40, 41] and generates noisier responses due to increased metastability. As a countermeasure, Majzoobi et al. [39] propose lightweight secure PUFs, which are based on multiple interleaved arbiter PUFs. However, Rhrmair et al. [51, 52] show that lightweight secure PUFs of low complexity can be emulated using machine learning techniques (Sect. 19.3.1).

Ring Oscillator PUFs. Ring oscillator PUFs typically consist of several identically designed ring oscillators, which are loops of an odd number of inverters that, once stimulated, oscillate at a certain frequency. The oscillating frequency of each ring oscillator depends on the signal delays of its components, which are affected by manufacturing process variations and environmental noise. Basic ring oscillator PUF constructions typically do not support challenges. Challengeable ring oscillator PUFs can be implemented by integrating controllable delay elements into the ring oscillator circuits. Gassend et al. [13, 15] propose an alternative construction of a challengeable ring oscillator PUF that uses the challenge to select two out of a set of ring oscillators and derives a single-bit response based on the ratio of the oscillation frequencies of the selected ring oscillators. A similar approach by Suh et al. [61] derives the response bit based on which of the oscillation frequencies is higher (Fig. 19.2). While reducing the effect of noise on the ring oscillators, these constructions generate a high correlation between PUF responses [36], which reduces the unpredictability of their responses.

19.2.2.2 Memory-Based PUFs

Memory-based PUFs exploit the power-up behavior of volatile memory cells, such as SRAM cells [16, 21], flip-flops [32, 36] and latches [29, 60]. These memory cells are inherently instable circuits that, when an external data signal input is applied, enter one of two different stable states to store one bit of information. When no data signal is provided, most cells preferably enter the same state after each power-up, while some cells always enter a random state. The state the memory cell enters depends on the physical properties of the underlying transistors, which are affected by manufacturing process variations and environmental noise. Note that the amount of unique responses of a memory-based PUF is always limited by the number of its memory cells, i.e., the size of the underlying memory block.

SRAM PUFs. PUFs based on Static Random Access Memory (SRAM) have been proposed by Guajardo et al. [16] and Holcomb et al. [21]. The challenge to an SRAM

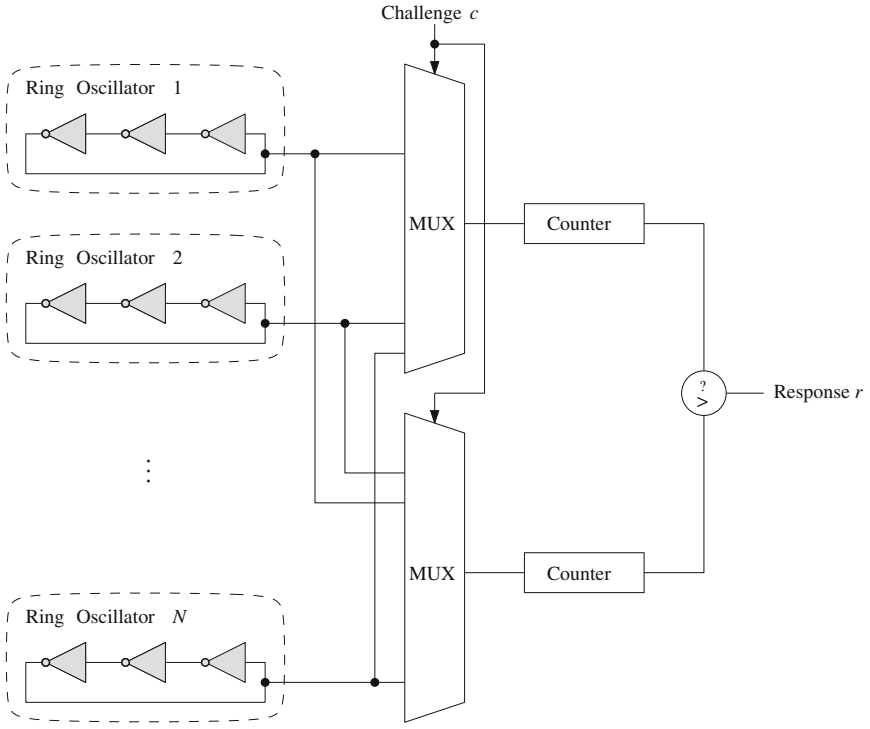


Fig. 19.2 Basic ring oscillator PUF design by Suh et al. [61]

PUF is a range of memory addresses, while the corresponding PUF response is the content of the uninitialized memory cells at those addresses.

An SRAM cell consists of two cross-coupled inverters that can store one bit of information and two additional transistors that are used to read and write data to the memory cell. Both inverters are typically designed to be identical in order to maximize write performance. When powered without applying a data signal, the SRAM cell will enter a state that depends on the threshold voltage mismatch of its transistors that is affected by manufacturing variations and environmental noise, in particular ambient temperature variations. SRAM cells with a large threshold mismatch always enter either the 0 or 1 state, while the state of cells with a small threshold mismatch is determined only by noise. In practice this means that some SRAM cells preferably enter the 0 state, others the 1 state, and some enter any of the two states with about the same probability. Cells that always enter the same state after power-up can be used as device-specific fingerprint since their behavior mainly depends on device-specific manufacturing process variations.

SRAM PUFs have been analyzed on FPGAs with dedicated SRAM [16, 17] and ASICs, including dedicated SRAM chips and SRAM embedded in micro-controllers [21, 22]. Note that each evaluation of an SRAM PUF requires the underlying SRAM

to be powered down and up again, which can be problematic when the SRAM of the PUF is also used as random access memory by the device containing the PUF.

Butterfly PUFs. Butterfly PUFs have been proposed by Kumar et al. [29]. They emulate SRAM cells using cross-coupled data latches that, in contrast to SRAM PUFs, can be easily reset by triggering the set/reset input of the latches. Butterfly PUFs have been implemented and evaluated on FPGAs by Kumar et al. [29].

Flip-Flop PUFs. Maes et al. [36] propose flip-flops PUFs as an alternative to SRAM and butterfly PUFs that can be efficiently implemented on FPGAs. Flip-flop PUFs have been implemented and analyzed on ASIC by Van der Leest et al. [32]. In contrast to SRAM PUFs, flip-flop PUFs can be easily spread over the whole circuit to obfuscate the location of the individual flip-flops, which increases the difficulty of reverse-engineering and invasive attacks against the PUF.

Latch PUFs. Latch PUFs have been presented by Su et al. [59] in the context of device identification. These PUFs consist of an array of latches built from cross-coupled NOR gates. The threshold voltage differences of the underlying transistors, which are mainly caused by manufacturing process variations and affected by environmental noise, in particular ambient temperature variations, cause a mismatch in the latch. Hence, the state of the latches directly after power-up mainly depends on manufacturing process variations and can be used as device fingerprint. Su et al. [59] implemented and evaluated latch-based PUFs on ASIC.

19.2.3 Noise Compensation and Privacy Amplification

Many PUF-based applications require PUF responses to be reliably reproducible while at the same time being unpredictable [1, 2, 37]. However, since PUFs are inherently noisy and their responses are not uniformly random, they are typically combined with *fuzzy extractors* [11]. Fuzzy extractors consist of a *secure sketch* that maps similar PUF responses to the same value (noise compensation or error correction), and a *randomness extractor*, which extracts full-entropy bit-strings from a partially random source (privacy amplification).

Fuzzy extractors and secure sketches generally work in two phases (Fig. 19.3): in the *enrolment phase* some helper data h and a uniform bit string K (e.g., a cryptographic key) is derived from PUF response r. Helper data h is used later in the *reconstruction phase* to recover K from a distorted PUF response $r' = r + e$, where e is the error caused by noise. An important property of fuzzy extractors and secure sketches is that, after observing one single helper data value h, there is still some min-entropy left in r and K, which means that h can be stored and transferred publicly without disclosing the full PUF response r or secret K [11].

More detailed information on fuzzy extractors and a number of practical instantiations can be found in the work by Dodis et al. [11].

Fig. 19.3 Concept of fuzzy extractors

Enrolment Phase	**Reconstruction Phase**
Generate helper data h...	*...that is later used to recreate K.*
$r \leftarrow \mathsf{PUF}(c)$	$r' \leftarrow \mathsf{PUF}(c)$
$(K, h) \leftarrow \mathsf{Gen}(r)$	$K \leftarrow \mathsf{Rep}(r', h)$
Store (c, h)	

19.2.4 Characterizing the Unpredictability of PUFs

The unpredictability property of PUFs ensures that it is infeasible to efficiently compute the response of a PUF to an unknown challenge. This is an important property in PUF-based applications, such as authentication protocols, where the adversary could forge the authentication if he could predict the PUF response. Note that unpredictability should be independent of the operating conditions, such as ambient temperature and supply voltage variations, which could be exploited by the adversary.

Depending on the application, different degrees of unpredictability are required. For instance, most PUF-based authentication schemes (Sect. 19.5.1) require a strong notion of unpredictability, where the adversary can adaptively obtain a certain number of challenge/response pairs from the PUF of the device under attack and from similar PUFs on other devices [1]. In other applications, such as PUF-based key storage (Sect. 19.5.2), a weaker notion of unpredictability is sufficient, where the adversary is assumed to be unable to obtain challenge/response pairs of the attacked PUF.

The most basic evaluation method that gives a first indication of the unpredictability of a PUF is to compute the Hamming weight of its responses, which shows whether the distribution of the PUF response bits is biased toward '0' or '1'. Ideally, both values should be equiprobable and their fractional Hamming weight[1] should be 0.5 %. An indication of the uniqueness of a PUF can be given by computing the Hamming distance between responses form different PUFs to the same challenge. In the ideal case, responses from different PUFs should be independent and thus their fractional Hamming distance[2] should be 0.5 %.

A more precise assessment of the unpredictability and uniqueness of PUF responses can be done by leveraging statistical tests, such as the DIEHARD [42] or NIST [53] test suites. However, since these test suites are typically based on a series of stochastic tests, they can only give an indication about whether the PUF responses are random or not. Moreover, they often require more input data than typical memory-based PUF implementations can provide. Another approach to empirically assess the unpredictability and uniqueness of PUFs is estimating the entropy of their responses based on experimental data. In particular, *min-entropy* indicates how many bits of a PUF response are uniformly random. The entropy of PUFs can

[1] The fractional Hamming weight is the number of bits in a bitstring that are '1' divided by the length of the bitstring.

[2] The fractional Hamming distance is the number of bits that are different in two bitstrings divided by the length of the bitstrings.

be approximated using the context-tree weighting (CTW) method [71, 72], which is an algorithm related to data compression that allows estimating the redundancy of bitstrings [19, 23, 32, 65]. For certain PUF types, the entropy of responses can be computed under consideration of the physical structure and properties of the PUF. For instance, Holcomb et al. [22] compute the entropy of SRAM PUFs based on empirical data under the assumption that the individual bytes of an SRAM array are independent [22]. Alternatively, similar as in symmetric cryptography, the unpredictability of a PUF can be estimated based on the complexity of the best known attack against the unpredictability property [1, 37]. For instance, there are attacks [51] against delay-based PUFs that emulate the PUF in software and allow predicting PUF responses to arbitrary challenges (Sect. 19.3.1).

Evaluation results in literature are difficult to compare due to varying test conditions, different analysis methods and the fact that no representative data sets are publicly available. Hence, a fair comparison of the unpredictability property of different PUF instances based on the results in literature is hardly possible. The development of a common evaluation framework for PUFs and the analysis of PUFs implemented in the same technology is an important topic for future research.

19.3 Attacks Against PUFs and PUF-Based Systems

19.3.1 Emulation Attacks

Most delay-based PUFs are subject to emulation or model building attacks that allow emulating the PUF in software [31, 35, 44, 51]. These attacks collect a number of challenge/response pairs of the PUF and use them to derive a mathematical model, such as a formula that allows estimating the PUF response to a given PUF challenge.

A number of mitigations against these attacks have been proposed [31, 40, 41], which are all based on inserting nonlinearity into the delay circuit (Sect. 19.2.2.1). However, Rhrmair et al. [51] show that these approaches are vulnerable to emulation attacks based on machine-learning techniques, such as logistic regression and evolution strategies. One approach to counter emulation attacks are controlled PUFs [15] (Sect. 19.4.1).

19.3.2 Side-Channel Attacks

Side-channel attacks are hardware attacks that aim to extract secret data, such as cryptographic keys, from an electronic component. Hereby, the adversary observes the behavior (such as the power consumption, electromagnetic radiation, and/or timing behavior) of the component while it is using the secret data to be extracted. Since the behavior of the component is typically dependent on the data processed, it can leak

information on this data. The fundamental underlying observation is that processing a data bit of value '1' typically consumes a different amount of power and/or time than processing a data bit of value '0'.

PUFs are typically used in combination with fuzzy extractors (Sect. 19.2.3) and most PUF-based applications (Sect. 19.5) require the plain PUF responses, i.e., before error correction and privacy amplification to be secret. Hence, side-channel attacks against PUF-based systems typically target the fuzzy extractor to gather challenge/response pairs and other information that eases emulation attacks on the underlying PUF (Sect. 19.3.1).

Research on side-channel analysis of PUFs and fuzzy extractors has been recently started and there are only a few published results. Karakoyunlu et al. [25] and Merli et al. [43] show side-channel attacks on implementations of common fuzzy extractors. Furthermore, Merli et al. [43] discuss potential side channel leakages of various PUF types. However, all known side channel attacks on PUF-based systems target the fuzzy extractor and are independent of the underlying PUF.

19.3.3 Fault Injection Attacks

Fault injection attacks aim to prompt erroneous behavior in a device by manipulating it in some way and, when combined with cryptanalysis, can lead to key recovery attacks. Faults may be injected in many ways, for instance by operating the device in extreme environmental conditions or by injecting transient faults into specific components of the device.

Attempts to operate the PUF outside its normal operating conditions, e.g., by varying its supply voltage or ambient temperature, will most likely affect the challenge/response behavior and thus the robustness and unpredictability of the PUF. Moreover, since implementations of fuzzy extractors and the underlying error correction algorithms are typically not resistant to fault injection attacks and exhibit data-dependent behavior, fault injection attacks can cause unintended leakage of PUF-related secret information, such as cryptographic keys bound to the PUF. In particular, most fuzzy extractors are not secure in case the helper data can be modified by the adversary [5]. Thus, robust fuzzy extractors have been proposed to prevent manipulations of helper data [10].

19.4 Advanced PUF Concepts

Several concepts have been proposed to enhance the security properties and functionality of standard PUFs.

19.4.1 Controlled PUFs

Most delay-based PUFs are subject to model building attacks that allow emulating the PUF in software (Sect. 19.3.1). One approach to counter this problem is controlled PUFs by Gassend et al. [15] that use cryptography in hardware to hide the actual PUF response from the adversary. Controlled PUFs typically apply a cryptographic hash function to the PUF challenges and/or responses, which introduces nonlinearity and breaks up the link between the actual PUF response and the output of the controlled PUF. Clearly, this does not address the fundamental weakness of delay-based PUFs. Moreover, to maintain verifiability of the controlled PUF, error correction must be applied before the noisy responses of the underlying PUF are processed by the cryptographic operation, which increases the complexity of the overall construction. Further, to protect against emulation attacks (Sect. 19.3.1), the cryptographic component and the error-correction mechanism as well as their connecting links must be protected against invasive and side-channel attacks, which may be hard to achieve in practice (Sects. 19.3.2 and 19.3.3).

19.4.2 Emulatable PUFs

The verification of PUF responses typically requires a database of reference challenge/response pairs (CRPs). This limits the scalability and efficiency of many PUF-based solutions and can be a serious drawback in many practical applications. One approach to counter this issue are emulatable PUFs, which, similar to public-key cryptography, allow the verification of PUF responses based on a publicly known mathematical model of the PUF. The concept of emulatable and publicly verifiable PUFs has been presented by Rhrmair et al. [48–50] as SIMPL systems (SIMulation Possible but Laborious). A similar concept known as *public PUFs* has been independently presented by Beckmann et al. [3]. The idea of both concepts is that the PUF can be emulated in software using a mathematical model of the physical properties of the PUF. However, this computation is assumed to take significantly more time than evaluating the actual PUF, which can be measured by a verifier in a PUF-based authentication protocol. This allows for the efficient verification of PUF responses by any entity with access to the mathematical model of the PUF, while preventing an algorithmic adversary from impersonating the PUF in the timeframe expected by the verifier. Concrete implementations of SIMPL systems have been presented by Rhrmair et al. [50].

Another approach to remove the need for a challenge/response pair database has been presented by Hammouri et al. [20, 44]. However, in contrast to SIMPL systems and public PUFs, their approach does not allow the public verification of PUF responses and requires the mathematical description of the PUF to be secret information that is only known to authorized entities, such as a verifier in an authentication protocol.

The security properties of practical instantiations of emulatable PUFs still need further evaluation.

19.5 Common Applications of PUFs

The most common applications of PUFs are identification, authentication, and secure key storage.

19.5.1 Device Identification and Authentication

The classical application of PUFs is the identification and authentication of physical objects, such as electronic devices. In fact, PUFs have been first proposed in the context of anti-counterfeiting solutions that prevent cloning (i.e., unauthorized copying) of products. There are many proposals to build identification and authentication schemes based on PUFs for various devices. We focus on solutions that are applicable to resource-constrained embedded devices, such as RFID systems.

One of the first proposals of using PUFs for RFID is by Ranasinghe et al. [47], who propose the manufacturer of a PUF-enabled RFID tag to store a set of challenge/response pairs (CRPs) in a database, which can later be used by RFID readers to identify the tag. The idea is that the reader chooses a challenge from the database, queries the tag and checks whether the database contains a tuple that matches the response received from the tag. One problem of this approach is that CRPs cannot be re-used since this would enable replay attacks. Hence, the number of tag authentications is limited by the number of CRPs in the database. This scheme has been implemented based on arbiter PUFs on RFID tags and its security and usability has been analyzed by Devadas et al. [9]. A similar approach based on the physical characteristics of SRAM cells has been proposed by Holcomb et al. [21].

A privacy-preserving PUF-based device authentication scheme has been presented by Gassend et al. [14]. They suggest to equip each tag with a PUF that is used to frequently derive new tag identifiers. Since readers cannot recompute these identifiers, the readers have access to a database that stores $(ID_0, ID_1, \ldots, ID_n)$ for each legitimate tag, where ID_0 is a random tag identifier and $ID_i = \text{PUF}(ID_{i-1})$ for $1 \leq i \leq n$. To authenticate to a reader, the tag first sends its current identifier ID_j and then updates its identity to $ID_{j+1} = \text{PUF}(ID_j)$. The reader then checks whether there is a tuple that contains ID_j in the database. In case the reader finds ID_j, it accepts the tag and invalidates all previous database entries ID_k, where $k \leq j$ to prevent replay attacks. Another approach to PUF-based authentication by Bolotnyy and Robins [4] aims to prevent unauthorized tracing of tokens. A similar approach to PUF-based authentication has been proposed by Bolotnyy and Robins [4]. A major drawback of these schemes is that tokens can only be authenticated a limited number of times without being re-initialized, which enables denial-of-service attacks.

19.5.2 Secure Key Storage and Key Generation

PUFs can be used to securely bind secrets (such as cryptographic keys) to the physical characteristics of a device. The concept of PUF-based key storage has been presented by Gassend [13] and later generalized to physically obfuscated algorithms by Bringer et al. [7]. Instead of storing the key in nonvolatile memory that is vulnerable to invasive attacks, the key is extracted from the physical properties of the underlying hardware each time it is used. This protects the key against unauthorized readout by invasive attacks, such as probing attacks against nonvolatile memory. Moreover, in case a tamper-evident PUF is used, any attempt to physically extract the key from the PUF circuit changes the challenge/response behavior of the PUF and securely deletes the key bound to the PUF.

Since PUF responses are typically not uniformly random and subject to noise, they cannot be used directly as cryptographic keys. Hence, privacy amplification, which adds additional entropy to the PUF response, and error-correction techniques must be applied before PUF responses can be used as cryptographic keys. The most common approach to achieve this are fuzzy extractors [11] (Sect. 19.2.3).

Tuyls et al. [62] propose to use a PUF-based key storage for the secret authentication key of RFID tags. Since the key is inherently hidden within the physical structure of the PUF, obtaining this key by hardware-related attacks is supposed to be intractable for real-world adversaries [15]. According to Tuyls et al. [62], a PUF-based key storage can be implemented with less than 1,000 gates, which is well within the capabilities of common RFID tags. Other authentication schemes for RFID exist that use PUF-based key storage to protect against unauthorized tracing of tokens [6, 54] and relay attacks [26].

19.6 Future Directions

19.6.1 Logically Reconfigurable PUFs

So far, most existing PUFs exhibit a static behavior, while a variety of applications would benefit from the availability of PUFs whose characteristics can be changed dynamically, i.e., reconfigured, after deployment. For instance, PUF-based key storage [34, 70] (Sect. 19.5.2) and PUF-based cryptographic primitives [2] may require that previous secrets derived from the PUF cannot be retrieved any more. Another examples are solutions to prevent downgrading of software [30] by binding the software to a certain hardware configuration, such as a PUF, which require the PUF behavior to be irreversibly altered upon installation of a software update.

Unfortunately, all known implementations of physically reconfigurable PUFs rely on optical mechanisms, reconfigurable hardware (such as FPGAs), or novel memory technologies [30], which all have serious drawbacks in practice. In particular, optical PUFs cannot be easily integrated into integrated circuits and require expen-

sive and error-prone evaluation equipment, while FPGA-based solutions cannot be realized with non-reconfigurable hardware (such as ASICs) that is commonly used in practice [37].

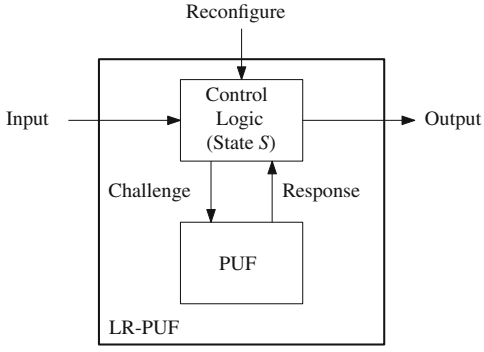
In this context, several attempts to emulate physically reconfigurable PUFs have been made. One of the first proposals was integrating a floating gate transistor into the delay lines of an arbiter PUF, which allows physically changing the challenge/response behavior of the PUF based on some state maintained in nonvolatile memory [33, 34]. Other approaches restrict access to the interface of the PUF and use part of the PUF challenge as reconfiguration data [30, 31], which, however, works only for certain PUF types.

The concept and security properties of logically reconfigurable physical unclonable functions (LR-PUFs) have been recently formalized by Katzenbeisser et al. [27]. In contrast to classical, typically static PUFs, LR-PUFs can be dynamically reconfigured after deployment such that their challenge/response behavior changes in a random manner without replacing or physically modifying the PUF. The idea is amending a conventional PUF with stateful control logic that transforms challenges and responses of the PUF (Fig. 19.4). Katzenbeisser et al. [27] present and evaluate two different constructions for LR-PUFs that are simple, efficient, and can be easily implemented.

19.6.2 PUF-Based Remote Attestation

Remote attestation is a mechanism to report the software state of a remote computing platform (*prover*) to a verifying party (*verifier*). This generally requires trusted hardware to securely record and transmit the system state of the prover to the verifier. However, trusted hardware is often too expensive for resource-constrained embedded devices, such as wireless sensor nodes and RFIDs. Hence, software attestation was proposed as a lightweight alternative that exploits the computational constraints of a device to make statements about its internal software state [57, 58]. Specifically,

Fig. 19.4 LR-PUF concept



software attestation requires the prover to compute the response R to a given attestation challenge C within a given time frame. When receiving the correct response in the expected time, the verifier has assurance that only a specific attestation algorithm could have been executed within that time frame. The attestation algorithm is implemented as a checksum function that iteratively merges information gathered from the device, such as program memory samples, into the attestation response R. Hence, a timely and correctly computed attestation response provides assurance to the verifier that the prover is in the expected system state.

Standard software attestation makes two major assumptions: (1) the computational capabilities of the prover are known to the verifier and unmodified, and (2) the attestation algorithm is indeed computed by the prover and not delegated to another device. In most scenarios, software attestation is thus limited to attest only *local* provers such that their identity can be directly verified and undesired communication interfaces can be disabled. But even a local prover does not always guarantee that its identity is authentic, e.g., if multiple hardware revisions of apparently identical devices exist.

To overcome these problems, the attestation response R must be linked to the hardware it was computed on, which can be achieved by using PUFs [55, 56]. The idea is to include the responses of the prover's PUF into the computation of R while the software attestation is running. To assure that the attestation algorithm is not outsourced, the PUF is queried sufficiently often to overwhelm all external communication interfaces of the prover. Thus, the constraints of the communication interfaces of the prover are exploited, similar to the computational constraints exploited by standard software attestation. Due to the uniqueness of the PUF responses and their tight integration into the attestation algorithm, a correct and timely attestation response R provides assurance on the identity of a remote device as well as the integrity of its software state. A practical implementation PUF-based attestation has been presented by Schulz et al. [28, 56].

19.7 Open Questions and Challenges

Practical PUF Designs. Known electronic PUFs may be compromised since delay-based PUFs can be emulated using machine-learning techniques (Sect. 19.3.1) and memory-based PUFs can be read out completely since they have only a limited response space. While these PUFs can be used in many applications, such as PUF-based key storage (Sect. 19.5.2) and controlled PUFs (Sect. 19.4.1), that ensure that the adversary cannot access the challenge/response pairs of the PUF, the use of these PUFs in applications with strong unclonability and unpredictability requirements, such as device authentication schemes (Sect. 19.5.1) must be carefully considered. Moreover, in general PUF responses can be verified only when the verifier has access to a database of previously recorded challenge/response pairs (CRPs), which may lead to scalability problems in practice. Hence, one open challenge is the development and implementation of novel PUF designs that achieve the requirements

of many existing theoretical PUF-based security solutions in literature, including resistance to emulation attacks, large (ideally exponential) challenge/response space to prevent complete readout of the PUF, public verifiability (i.e., no CRP database required to verify PUF response), tamper-evidence, physical reconfigurability, and small hardware footprint.

Common Evaluation Framework for PUFs. Currently, there is no common evaluation framework for PUFs that allows assessing and quantifying the security properties of real PUF implementations. The security properties of existing PUF-based security solutions in literature are proven in PUF security models that are typically not general enough and exclude certain PUF types, do not reflect all properties of real PUF implementations, or include security parameters that cannot be determined for real PUF implementations in practice. Hence, it is unclear whether these schemes can actually be implemented securely. Therefore, another open challenge is the development of a common evaluation framework for the analysis of PUF implementations that (1) captures the security properties of PUFs according to modern cryptographic standards and can be used to assess the security of PUF-based cryptographic schemes and security solutions, and (2) allows for empirically assessing and quantifying the most important properties of PUFs, including robustness, physical unclonability, unpredictability of responses to the same challenge to different PUF instances and to different challenges to the same PUF instance, and tamper-evidence. A promising first step in this direction has been presented by Armknecht et al. [1]. However, they do not consider all security properties of PUFs and do not show how their approach applies to other PUF implementations than SRAM PUFs.

Side-channel Analysis of PUFs. Many PUF-based applications such as PUF-based key storage require PUF responses to be inaccessible to the adversary, which is typically justified by the assumption of the PUF being tamper-evident so that any attempt to physically access the PUF response (such as an invasive attack) permanently changes the challenge/response behavior of the PUF. However, even when a tamper-evident PUF (such as a coating PUF) is used, it is currently unclear whether existing PUF implementations in integrated circuits leak information on their response over side channels, such as electromagnetic radiation or power consumption. Hence, the analysis of the side-channel leakage of known PUF implementations is an interesting open research problem.

19.8 Conclusion

Physically unclonable functions are a very interesting and promising approach to increase the security of embedded systems. They open new directions toward lightweight privacy-preserving protocols based on physical assumptions and cost-effective tamper-evident storage for cryptographic secrets that even cannot be learned or reproduced by the manufacturer of the corresponding PUF.

However, several aspects of PUFs and their deployment require further research. Since PUFs are bound to the device in which they are embedded, no other entity can verify the response r of a PUF to a given challenge c without knowing an authentic challenge/response pair (c, r) in advance. Current PUF-based protocols aim at circumventing this problem by providing the reader with a database that contains a set of challenge/response pairs that act as reference values for the responses of the interrogated PUF. However, this approach is not scalable and opens the possibility of replay-attacks. Furthermore, PUFs realizations require careful statistical testing before they can be safely deployed to real security-critical products, while, to our knowledge, there is no complete security evaluation framework for PUFs yet.

Acknowledgments This work has been supported in part by the European Commission under grant agreement ICT-2007-238811 UNIQUE.

References

1. Armknecht, F., Maes, R., Sadeghi, A.R., Standaert, F.X., Wachsmann, C.: A formal foundation for the security features of physical functions. In: IEEE Symposium on Security and Privacy (SSP), pp. 397–412. IEEE Computer Society (2011)
2. Armknecht, F., Maes, R., Sadeghi, A.R., Sunar, B., Tuyls, P.: Memory leakage-resilient encryption based on physically unclonable functions. In: M. Matsui (ed.) Advances in Cryptology (ASIACRYPT), *Lecture Notes in Computer Science (LNCS)*, vol. 5912, pp. 685–702. Springer Berlin/Heidelberg, Berlin, Heidelberg (2009)
3. Beckmann, N., Potkonjak, M.: Hardware-based public-key cryptography with public physically unclonable functions. In: S. Katzenbeisser, A.R. Sadeghi (eds.) Information Hiding (IH), *Lecture Notes in Computer Science (LNCS)*, vol. 5806, pp. 206–220. Springer Berlin/Heidelberg, Berlin, Heidelberg (2009)
4. Bolotnyy, L., Robins, G.: Physically unclonable function-based security and privacy in RFID systems. In: Conference on Pervasive Computing and Communications (PerCom), pp. 211–220. IEEE (2007)
5. Boyen, X.: Reusable cryptographic fuzzy extractors. In: ACM Conference on Computer and Communications Security (ACM CCS), pp. 82–91. ACM, New York, NY, USA (2004)
6. Bringer, J., Chabanne, H., Icart, T.: Improved privacy of the tree-based hash protocols using physically unclonable functions. In: R. Ostrovsky, R. De Prisco, I. Visconti (eds.) Security and Cryptography for Networks (SCN), *Lecture Notes in Computer Science (LNCS)*, vol. 5229, pp. 77–91. Springer Berlin/Heidelberg, Berlin, Heidelberg (2008)
7. Bringer, J., Chabanne, H., Icart, T.: On physical obfuscation of cryptographic algorithms. In: B. Roy, N. Sendrier (eds.) International Conference on Cryptology in India (INDOCRYPT), *Lecture Notes in Computer Science (LNCS)*, vol. 5922, pp. 88–103. Springer Berlin/Heidelberg, Berlin, Heidelberg (2009)
8. Brzuska, C., Fischlin, M., Schröder, H., Katzenbeisser, S.: Physically uncloneable functions in the universal composition framework. In: P. Rogaway (ed.) Advances in Cryptology (CRYPTO), *Lecture Notes in Computer Science (LNCS)*, vol. 6841, pp. 51–70. Springer Berlin/Heidelberg, Berlin, Heidelberg (2011)
9. Devadas, S., Suh, E., Paral, S., Sowell, R., Ziola, T., Khandelwal, V.: Design and implementation of PUF-based unclonable RFID ICs for anti-counterfeiting and security applications. RFID, 2008 IEEE International Conference on pp. 58–64 (2008)
10. Dodis, Y., Katz, J., Reyzin, L., Smith, A.: Robust fuzzy extractors and authenticated key agreement from close secrets. In: C. Dwork (ed.) Advances in Cryptology (CRYPTO), *Lecture Notes*

in Computer Science (LNCS), vol. 4117, pp. 232–250. Springer Berlin/Heidelberg, Berlin, Heidelberg (2006)
11. Dodis, Y., Reyzin, L., Smith, A.: Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In: C. Cachin, J. Camenisch (eds.) Advances in Cryptology (EUROCRYPT), *Lecture Notes in Computer Science (LNCS)*, vol. 3027, pp. 523–540. Springer Berlin/Heidelberg, Berlin, Heidelberg (2004)
12. Eichhorn, I., Koeberl, P., van der Leest, V.: Logically reconfigurable PUFs: Memory-based secure key storage. In: ACM Workshop on Scalable Trusted Computing (ACM STC), pp. 59–64. ACM, New York, NY, USA (2011)
13. Gassend, B.: Physical random functions. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology (MIT), The Stata Center, 32 Vassar Street, Cambridge, Massachusetts 02139 (2003)
14. Gassend, B., Clarke, D., van Dijk, M., Devadas, S.: Controlled physical random functions. In: Annual Computer Security Applications Conference (ACSAC), pp. 149–160. IEEE (2002)
15. Gassend, B., Clarke, D., van Dijk, M., Devadas, S.: Silicon physical random functions. In: ACM Conference on Computer and Communications Security (ACM CCS), pp. 148–160. ACM, New York, NY, USA (2002)
16. Guajardo, J., Kumar, S., Schrijen, G.J., Tuyls, P.: FPGA intrinsic PUFs and their use for IP protection. In: P. Paillier, I. Verbauwhede (eds.) Cryptographic Hardware and Embedded Systems (CHES), *Lecture Notes in Computer Science (LNCS)*, vol. 4727, pp. 63–80. Springer Berlin/Heidelberg, Berlin, Heidelberg (2007)
17. Guajardo, J., Kumar, S.S., Schrijen, G.J., Tuyls, P.: Physical unclonable functions and public-key crypto for FPGA IP protection. In: Field Programmable Logic and Applications (FPL), pp. 189–195. IEEE (2007)
18. Guajardo, J., Kumar, S.S., Schrijen, G.J., Tuyls, P.: Brand and IP protection with physical unclonable functions. In: IEEE International Symposium on Circuits and Systems (ISCAS), pp. 3186–3189. IEEE (2008)
19. Hammouri, G., Dana, A., Sunar, B.: CDs have fingerprints too. In: C. Clavier, K. Gaj (eds.) Cryptographic Hardware and Embedded Systems (CHES), *Lecture Notes in Computer Science (LNCS)*, vol. 5747, pp. 348–362. Springer Berlin/Heidelberg, Berlin, Heidelberg (2009)
20. Hammouri, G., Öztürk, E., Birand, B., Sunar, B.: Unclonable lightweight authentication scheme. In: L. Chen, M.D. Ryan, G. Wang (eds.) International Conference on Information and Communications Security (ICICS), *Lecture Notes in Computer Science (LNCS)*, vol. 5308, pp. 33–48. Springer Berlin/Heidelberg, Berlin, Heidelberg (2008)
21. Holcomb, D., Burleson, W., Fu, K.: Initial SRAM state as a fingerprint and source of true random numbers for RFID tags. In: Workshop on RFID Security (RFIDSec) (2007)
22. Holcomb, D., Burleson, W.P., Fu, K.: Power-up SRAM state as an identifying fingerprint and source of true random numbers. IEEE Transactions on Computers 58(9), 1198–1210 (2009)
23. Ignatenko, T., Schrijen, G.J., Škorić, B., Tuyls, P., Willems, F.: Estimating the secrecy-rate of physical unclonable functions with the context-tree weighting method. In: IEEE International Symposium on Information Theory (ISIT), pp. 499–503. IEEE (2006)
24. Intrinsic ID: Website. http://www.intrinsic-id.com/products.htm (2012)
25. Karakoyunlu, D., Sunar, B.: Differential template attacks on PUF enabled cryptographic devices. In: Workshop on Information Forensics and Security (WIFS), pp. 1–6. IEEE (2010)
26. Kardas, S., Kiraz, M.S., Bingol, M.A., Demirci, H.: A novel RFID distance bounding protocol based on physically unclonable functions. In: Radio Frequency Identification: Security and Privacy Issues (RFIDSec), *Lecture Notes in Computer Science (LNCS)*. Springer Berlin/Heidelberg, Berlin, Heidelberg (2011)
27. Katzenbeisser, S., Kocabaş, U., van der Leest, V., Sadeghi, A.R., Schrijen, G.J., Schröder, H., Wachsmann, C.: Recyclable PUFs: Logically reconfigurable PUFs. In: Workshop on Cryptographic Hardware and Embedded Systems (CHES), vol. 6917, pp. 374–389. Springer Berlin/Heidelberg, Berlin, Heidelberg (2011)
28. Kocabas, Ü., Sadeghi, A.R., Schulz, S., Wachsmann, C.: Poster: Practical embedded remote attestation using physically unclonable functions (2011)

29. Kumar, S.S., Guajardo, J., Maes, R., Schrijen, G.J., Tuyls, P.: Extended abstract: The butterfly PUF protecting IP on every FPGA. In: Workshop on Hardware-Oriented Security (HOST), pp. 67–70. IEEE (2008)
30. Kursawe, K., Sadeghi, A.R., Schellekens, D., Skoric, B., Tuyls, P.: Reconfigurable physical unclonable functions – Enabling technology for tamper-resistant storage. In: Workshop on Hardware-Oriented Security and Trust (HOST), pp. 22–29. IEEE (2009)
31. Lee, J.W., Lim, D., Gassend, B., Suh, E.G., van Dijk, M., Devadas, S.: A technique to build a secret key in integrated circuits for identification and authentication applications. In: Symposium on VLSI Circuits, pp. 176–179. IEEE (2004)
32. van der Leest, V., Schrijen, G.J., Handschuh, H., Tuyls, P.: Hardware intrinsic security from D flip-flops. In: ACM Workshop on Scalable Trusted Computing (ACM STC), pp. 53–62. ACM, New York, NY, USA (2010)
33. Lim, D.: Extracting secret keys from integrated circuits. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology (MIT), The Stata Center, 32 Vassar Street, Cambridge, Massachusetts 02139 (2004)
34. Lim, D., Lee, J.W., Gassend, B., Suh, E.G., van Dijk, M., Devadas, S.: Extracting secret keys from integrated circuits. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 13(10), 1200–1205 (2005)
35. Lin, L., Holcomb, D., Krishnappa, D.K., Shabadi, P., Burleson, W.: Low-power sub-threshold design of secure physical unclonable functions. In: International Symposium on Low-Power Electronics and Design (ISLPED), pp. 43–48. IEEE (2010)
36. Maes, R., Tuyls, P., Verbauwhede, I.: Intrinsic PUFs from flip-flops on reconfigurable devices. In: Benelux Workshop on Information and System Security (2008)
37. Maes, R., Verbauwhede, I.: Physically unclonable functions: A study on the state of the art and future research directions. In: A.R. Sadeghi, D. Naccache (eds.) Towards Hardware-Intrinsic Security, Information Security and Cryptography, pp. 3–37. Springer Berlin/Heidelberg, Berlin, Heidelberg (2010)
38. Maiti, A., Casarona, J., McHale, L., Schaumont, P.: A large scale characterization of RO-PUF. In: Symposium on Hardware-Oriented Security and Trust (HOST), pp. 94–99. IEEE (2010)
39. Majzoobi, M., Koushanfar, F., Potkonjak, M.: Lightweight secure PUFs. In: International Conference on Computer-Aided Design (ICCAD), pp. 670–673. IEEE (2008)
40. Majzoobi, M., Koushanfar, F., Potkonjak, M.: Testing techniques for hardware security. In: International Test Conference (ITC), pp. 1–10. IEEE (2008)
41. Majzoobi, M., Koushanfar, F., Potkonjak, M.: Techniques for design and implementation of secure reconfigurable PUFs. ACM Transactions on Reconfigurable Technology and Systems (TRETS) 2(1), 1–33 (2009)
42. Marsaglia, G.: The marsaglia random number CDROM including the Diehard battery of tests of randomness. http://www.stat.fsu.edu/pub/diehard/
43. Merli, D., Schuster, D., Stumpf, F., Sigl, G.: Side-channel analysis of PUFs and fuzzy extractors. In: J.M. McCune, B. Balacheff, A. Perrig, A.R. Sadeghi, A. Sasse, Y. Beres (eds.) Trust and Trustworthy Computing (TRUST), *Lecture Notes in Computer Science (LNCS)*, vol. 6740, pp. 33–47. Springer Berlin/Heidelberg, Berlin, Heidelberg (2011)
44. Öztürk, E., Hammouri, G., Sunar, B.: Towards robust low cost authentication for pervasive devices. In: Conference on Pervasive Computing and Communications (PerCom), pp. 170–178. IEEE, Washington, DC, USA (2008)
45. Pappu, R., Recht, B., Taylor, J., Gershenfeld, N.: Physical one-way functions. Science 297(5589), 2026–2030 (2002)
46. Posch, R.: Protecting devices by active coating. Journal of Universal Computer Science 4(7), 652–668 (1998)
47. Ranasinghe, D.C., Engels, D.W., Cole, P.H.: Security and privacy: Modest proposals for low-cost RFID systems. In: Auto-ID Labs Research Workshop (2004)
48. Rührmair, U.: SIMPL systems: On a public key variant of physical unclonable functions. Cryptology ePrint Archive, Report 2009/255 (2009)

49. Rührmair, U.: SIMPL systems, or: Can we design cryptographic hardware without secret key information? In: I. Černá, T. Gyimóthy, J. Hromkovič, K. Jefferey, R. Králović, M. Vukolić, S. Wolf (eds.) Current Trends in Theory and Practice of Computer Science (SOFSEM), *Lecture Notes in Computer Science (LNCS)*, vol. 6543, pp. 26–45. Springer Berlin/Heidelberg, Berlin, Heidelberg (2011)
50. Rührmair, U., Chen, Q., Stutzmann, M., Lugli, P., Schlichtmann, U., Csaba, G.: Towards electrical, integrated implementations of SIMPL systems. In: P. Samarati, M. Tunstall, J. Posegga, K. Markantonakis, D. Sauveron (eds.) Workshop on Information Security Theory and Practices (WISTP), *Lecture Notes in Computer Science (LNCS)*, vol. 6033, pp. 277–292. Springer Berlin/Heidelberg, Berlin, Heidelberg (2010)
51. Rührmair, U., Sehnke, F., Sölter, J., Dror, G., Devadas, S., Schmidhuber, J.: Modeling attacks on physical unclonable functions. In: ACM Conference on Computer and Communications Security (ACM CCS), pp. 237–249. ACM, New York, NY, USA (2010)
52. Rührmair, U., Sölter, J., Sehnke, F.: On the foundations of physical unclonable functions. Cryptology ePrint Archive, Report 2009/277 (2009)
53. Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E., Leigh, S., Levenson, M., Vangel, M., Banks, D., Heckert, A., Dray, J., Vo, S.: A statistical test suite for random and pseudorandom number generators for cryptographic applications. Special Publication 800–22 Revision 1a, NIST (2010)
54. Sadeghi, A.R., Visconti, I., Wachsmann, C.: Enhancing RFID security and privacy by physically unclonable functions. In: A.R. Sadeghi, D. Naccache (eds.) Towards Hardware-Intrinsic Security, Information Security and Cryptography, pp. 281–305. Springer Berlin/Heidelberg, Berlin, Heidelberg (2010)
55. Schulz, S., Sadeghi, A.R., Wachsmann, C.: Short paper: Lightweight remote attestation using physical functions. In: ACM Conference on Wireless Network Security (WiSec), pp. 109–114. ACM, New York, NY, USA (2011)
56. Schulz, S., Wachsmann, C., Sadeghi, A.R.: Lightweight remote attestation using physical functions. Tech. rep., Center for Advanced Security Research Darmstadt (CASED), Germany, Mornewegstraße 32, 64293 Darmstadt, Germany (2011)
57. Seshadri, A., Luk, M., Shi, E., Perrig, A., van Doorn, L., Khosla, P.: Pioneer: Verifying code integrity and enforcing untampered code execution on legacy systems. In: ACM Symposium on Operating Systems Principles (SOSP), vol. 39, pp. 1–16. ACM, New York, NY, USA (2005)
58. Seshadri, A., Perrig, A., van Doorn, L., Khosla, P.: SWATT: SoftWare-based ATTestation for embedded devices. In: IEEE Symposium on Security and Privacy (SSP), pp. 272–282. IEEE, Los Alamitos, CA, USA (2004)
59. Su, Y., Holleman, J., Otis, B.P.: A 1.6pJ/bit 96% stable chip-ID generating circuit using process variations. In: International Solid-State Circuits Conference (ISSCC), pp. 406–611. IEEE (2007)
60. Su, Y., Holleman, J., Otis, B.P.: A digital 1.6 pJ/bit chip identification circuit using process variations. IEEE Journal of Solid-State Circuits 43(1), 69–77 (2008)
61. Suh, E.G., Devadas, S.: Physical unclonable functions for device authentication and secret key generation. In: ACM/IEEE Design Automation Conference (DAC), pp. 9–14. IEEE (2007)
62. Tuyls, P., Batina, L.: RFID-tags for anti-counterfeiting. In: D. Pointcheval (ed.) Topics in Cryptology (CT-RSA), *Lecture Notes in Computer Science (LNCS)*, vol. 3860, pp. 115–131. Springer Berlin/Heidelberg, Berlin, Heidelberg (2006)
63. Tuyls, P., Schrijen, G.J., Škorić, B., van Geloven, J., Verhaegh, N., Wolters, R.: Read-proof hardware from protective coatings. In: L. Goubin, M. Matsui (eds.) Cryptographic Hardware and Embedded Systems (CHES), *Lecture Notes in Computer Science (LNCS)*, vol. 4249, pp. 369–383. Springer Berlin/Heidelberg, Berlin, Heidelberg (2006)
64. Tuyls, P., Škorić, B.: Secret key generation from classical physics: Physical uncloneable functions. In: S. Mukherjee, R.M. Aarts, R. Roovers, F. Widdershoven, M. Ouwerkerk (eds.) Am Iware Hardware Technology Drivers of Ambient Intelligence, Philips Research Book Series, vol. 5, pp. 421–447. Springer Netherlands, Dordrecht (2006)

65. Tuyls, P., Škorić, B., Ignatenko, T., Willems, F., Schrijen, G.J.: Entropy estimation for optical PUFs based on context-tree weighting methods. In: P. Tuyls, B. Škorić, T. Kevenaar (eds.) Security with Noisy Data, pp. 217–233. Springer London, London (2007)
66. Tuyls, P., Škorić, B., Stallinga, S., Akkermans, A.H.M., Ophey, W.: Information-theoretic security analysis of physical uncloneable functions. In: A. Patrick, M. Yung (eds.) Financial Cryptography and Data Security (FC), *Lecture Notes in Computer Science (LNCS)*, vol. 3570, p. 578. Springer Berlin/Heidelberg, Berlin, Heidelberg (2005)
67. Verayo, Inc.: Website. http://www.verayo.com/product/products.html (2012)
68. Škorić, B., Maubach, S., Kevenaar, T., Tuyls, P.: Information-theoretic analysis of capacitive physical unclonable functions. Journal of Applied Physics **100**(2), 024,902–024,902–11 (2006)
69. Škorić, B., Maubach, S., Kevenaar, T., Tuyls, P.: Information-theoretic analysis of coating PUFs. Cryptology ePrint Archive, Report 2006/101 (2006)
70. Škorić, B., Tuyls, P., Ophey, W.: Robust key extraction from physical uncloneable functions. In: J. Ioannidis, A. Keromytis, M. Yung (eds.) Applied Cryptography and Network Security (ACNS), *Lecture Notes in Computer Science (LNCS)*, vol. 3531, pp. 99–135. Springer Berlin/Heidelberg, Berlin, Heidelberg (2005)
71. Willems, F.M.J.: CTW website. http://www.ele.tue.nl/ctw/
72. Willems, F.M.J., Shtarkov, Y.M., Tjalkens, T.J.: The context-tree weighting method: Basic properties. IEEE Transactions on Information Theory **41**(3), 653–664 (1995)