PUFs at a Glance

Ulrich Rührmair
Technische Universität München
80333 München, Germany
E-mail: ruehrmair@ilo.de

Daniel E. Holcomb
University of Michigan
Ann Arbor, MI 48109, USA
E-mail: danholcomb@umich.edu

***Abstract*—Physical Unclonable Functions (PUFs) are a new, hardware-based security primitive, which has been introduced just about a decade ago. In this paper, we provide a brief and easily accessible overview of the area. We describe the typical security features, implementations, attacks, protocols uses, and applications of PUFs. Special focus is placed on the two most prominent PUF types, so-called *"Weak PUFs"* and *"Strong PUFs"*, and their mutual differences.**

***Keywords*—Physical Unclonable Functions, Overview, Survey, Weak PUFs, Strong PUFs**

I. Introduction

A. Motivation and Background

Electronic devices are pervasive in our everyday life. This leads to a host of security and privacy issues. Classical cryptography offers several measures against these problems, but they all rest on the concept of a secret binary key: It is assumed that the devices can permanently store a piece of digital information that is, and remains, unknown to the adversary. Unfortunately, this requirement can be quite difficult to uphold in practice. Physical attacks such as invasive, semi-invasive, or side-channel attacks, as well as software attacks like API-attacks and viruses, can lead to key exposure and security breaks [1]. One additional complication lies in the fact that the employed devices should ideally be lightweight and cost efficient, and are resource-constrained in certain commercial scenarios. For example, some security systems will not contain non-volatile memory cells due to their extra costs. This poses the question: How can medium or even high security levels be achieved in such circumstances?

B. PUFs, Role of Manufacturing Variations, Challenge-Response Formalism

The described situation was one motivation that led to the development of *physical unclonable functions (PUFs)*. Their key idea is to exploit the *"random physical disorder"* or the *"manufacturing variations"* that occur in almost all physical systems on small length scales. The shown disorder typically cannot be fully controlled during the fabrication of the system, and cannot be re-fabricated intentionally, not even by the original manufacturer. It is *unclonable*, and constitutes an individual fingerprint of each system. Usually, this phenomenon is regarded as disadvantageous, for example in the context of semiconductor fabrication. Integrated circuits commonly have to be designed in such a way that their digital behavior remains unaffected by manufacturing variations. PUFs, however, turn said variations into an advantage, and explicitly exploit them for security purposes.

More specifically, a PUF is an (at least partly) disordered physical system that can be challenged with external stimuli or so-called *"challenges"* C_i. Depending on the exact PUF type, a PUF can thereby have merely one single possible challenge, a few challenges, or even an exponential number of challenges in some system parameter (see Sections II and III). Upon being exposed to a challenge C_i, the PUF reacts by producing a corresponding response R_i. The tuples (C_i, R_i) are thereby termed the challenge-response pairs (CRPs) of the PUF.

PUFs are deliberately designed such that the response(s) R_i depend on the individual physical disorder present in the PUF. Each PUF response is hence not only a function of the applied challenge C_i, but also of the PUF's physical disorder. One consequence is that the challenge-response behavior varies between different "physical instances" or "specimen" of the same PUF, since any instance is subject to different manufacturing variations. From an abstract perspective, one could say a PUF's challenge-response mechanism converts the unique *physical* disorder of the PUF into *digital* input-output data. While the exact challenge-response mechanisms vary, most existing electrical PUFs thereby produce responses that consist of exactly one bit. If necessary, several such single-bit responses may be bundled to obtain a multi-bit identifier/key.

C. Inevitable Error Correction

Since PUF responses are based on very small manufacturing variations, PUFs usually operate more closely at their stability limits than classical, digital systems. This renders numeric error correction vital. Two basic approaches exist.

Firstly, standard error correction mechanisms may be applied to each PUF response, converting it into a stable, noise-free output. Usually this error correction is accomplished via some so-called *"helper data"*. The latter assists in the error correction process of a given response, and has been derived upon an earlier measurement of this response. The helper data must be stored in some non-volatile memory (NVM) accompanying the PUF, but not necessarily inside the PUF-carrying system (which may not be equipped with an NVM). It can be constructed in such a way that it can become known to an adversary without compromising the secrecy of the PUF response, i.e., it does not need to be kept secret [18].

978-3-9815370-2-4/DATE14/©2014 EDAA

This work was supported in part by C-FAR, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA, and NSF CNS-0845874. Any opinions, findings, conclusions, and recommendations expressed in these materials are those of the authors and do not necessarily reflect the views of the sponsors.

A second, less widespread possibility for error correction is the design of PUF protocols with inbuilt error tolerances. This circumvents the need for perfect error correction inside the PUF-carrying hardware, enabling more lightweight systems. One example for this method is the well-known Strong PUF identification protocol of Pappu et al. [52], [53] that we discuss in Section III-C. We remark in passing that there are notable differences between different PUF types regarding error correction, which will be discussed in Sections II-C and III-C.

D. Aspired Advantages and Some Applications

There are two benefits that users would like to gain from PUFs: Security advantages and certain forms of cost/practicality upsides. These assumed benefits have acted as drivers for PUF research in the past.

Let us start with security aspects. Due to its complex and disordered structure, a PUF can avoid some of the shortcomings associated with digital keys. For example, it is usually harder to read out, predict, or derive its responses than to obtain the values of digital keys that are permanently stored in non-volatile memory. The PUF-responses are only derived when needed, meaning that they are present in the security system in an easily accessible digital form only for very short time periods. Furthermore, many PUFs have been assumed to be tamper sensitive, meaning that invasive attacks would alter the PUF's response behavior permanently and notably. These facts have been exploited for various PUF-based security protocols. Prominent examples include schemes for identification and authentication [53], [22], key storage [76], [25], key exchange [15], [8], or digital rights management purposes [23].

On the cost/practicality side, PUFs allow the "storage" of keys in hardware systems that do not have NVM. One prominent example are FPGAs, where SRAM-based PUFs have been suggested to derive a key on the FPGA. This key can be used, for example, to encrypt/decrypt the design bitstream that is uploaded onto the FPGA [25], since this design may represent a substantial intellectual worth. Similar aspects hold for other systems without NVMs, in which PUFs can be used as an identifier or as key source.

II. Weak PUFs

The two most important subtypes of PUFs, which should be distinguished explicitly in any sound treatment of the topic, are so-called *"Weak PUFs"* and *"Strong PUFs"*. They are discussed in this and the upcoming section.

A. Characteristic Features of Weak PUFs

Weak PUFs essentially are a new form of storing secret keys in vulnerable hardware, offering an alternative to ROM, Flash or other non-volatile memories (NVMs). As all PUFs, Weak PUFs exhibit some internal, unclonable physical disorder, and possess some form of challenge-response mechanism that exploits this disorder (see Section I-B). Beyond this, their characteristic features are as follows (compare [67], [60], [2]:

1) *Few challenges:* A Weak PUF has got very few, fixed challenges, commonly only one challenge per PUF instance.
2) *Access-restricted responses:* In all but very few applications, the challenge-response interface (or the challenge-response mechanism, respectively) of a Weak PUF needs to be access-restricted. It is assumed that adversaries cannot access to the Weak PUF's responses, even if they hold physical possession of the PUF-carrying hardware.

Both features fundamentally distinguish Weak PUFs from Strong PUFs (compare Section III-A).

B. Implementation Examples

Weak PUFs can be implemented either using special purpose integrated circuits designed to be sensitive to variation, or by using the intrinsic variation present in all existing circuits. While some of the first Weak PUFs were based on special purpose circuits, the recent trend is toward instrinsic PUFs fabricated from standard CMOS logic parts, since this is more cost effective. One of the earliest Weak PUFs was a design proposed in 2000 by Lofstrom et al. [42] to leverage threshold mismatch for identifying circuits. A more involved PUF based on sensing the capacitance of specially applied protective coatings was given by Tuyls et al. [76]. Later, Su et al. [74] demonstrated a chip-ID circuit based on cross-coupled devices; to evaluate the ID, the cross coupled devices are brought to a metastable state, and then allowed to spontaneously transition to a stable state determined by process variation. In noting that their design is SRAM-like, the authors of this work foreshadow the subsequent trend of SRAM-based Weak PUFs.

The most popular implementation of intrinsic Weak PUFs are SRAM PUFs. They exploit the inherent threshold variation of the cross-coupled SRAM cells. The differential nature of the cells make them ideal for being sensitive to variation and also largely immune to common-mode noise. Furthermore, the ubiquity of SRAM in nearly all VLSI circuits gives them wide applicability as PUFs. The physical identifier is automatically generated in the cell whenever it goes from an un-powered state to a powered state, and the identifier can then be read out using the standard memory access mechanism. The earliest known mention of this phenomenon is in a patent by Layman et al [38]. The phenomenon of SRAM signatures nevertheless remained unknown to the wider research community until being later rediscovered in 2007 independenly and concurrently by Holcomb et al. [31] and Guajardo et al. [25]. An alternative formulation of an SRAM-based Weak PUFs uses the minimum data retention voltage of cells instead of the power-up state [33]. Subsequent to SRAM PUFs, a variety of other intrinsic Weak PUFs have been proposed. Memory-based PUFs are suggested for storage technologies including Flash [54], Memristors [35], and DRAM [56]. Intrinsic non-memory PUFs are also proposed, including the butterfly PUF [36] that uses cross-coupled latches in FPGAs, and a PUF based on bus-keepers [72].

All of the above PUFs have one fixed way to excite them (for example powering them up), and hence exactly one challenge.

C. Applications and Error Correction

The main application of Weak PUFs is to derive secret keys inside (lightweight) hardware systems. In principle, one

can distinguish two basic cases.

The first and by far most popular case is the derivation of an internal, but *shared* secret key from Weak PUF responses, which is known to a limited number of parties outside the PUF-carrying hardware — usually only to the manufacturer of the PUF. This approach pressumes that the manufacturer learns the key, for example by directly accessing the Weak PUF responses, in a secure set-up phase. At the same time, one commonly assumes that adversaries with physical access to the Weak PUF carrying hardware cannot access the PUF's responses, or learn the key, after the set-up phase. Even though these two assumption live in some tension, they have never been put too much in question in the general Weak PUF literature. In practice, they may be realized by disabling access to the PUF after the secure set-up phase in one way or the other. The internal, shared key can then used for any classical secret key based application. One exemplary and commercially attractive application was already named in Section I-D: Encryption of the design bitstream that is uploaded onto FPGAs. The second, but far less popular basic case is the derivation of an internal, *unshared* key that is unknown to any party outside the PUF-carrying hardware. This case partly releaves the abovementioned tension, since the key never must leave the system. It can remain forever inside a PUF-carrying, tamper-sensitive hardware, for example a hardware that is surrounded by a PUF-like coating [76]. One straightforward use for such an internal, unshared key is memory encryption [76].

We stress that in any non-trivial applications of Weak PUFs, perfect error correction in the derived secret keys neeeds to be achieved. Since the secret key is never released to the outside (after the set-up phase), this error correction must be carried out internally, requiring suitable resources in the Weak PUF carrying hardware. Several different approaches have been developed to this end, including [7], [43], [81], [29], [82], to which we refer the reader.

D. Attacks on Weak PUFs

If the digital responses arising from a Weak PUF are read out by invasive means, the security of the system is compromised. This is in principle comparable to the security of a secret key stored in NVM, even though the PUF-response exists in the system only for a short time. Still, this inherent attack point of Weak PUFs has been successfully exploited in recent publications by Nedospasov et al. [48]. Even if care is taken to prevent SRAM PUF values from ever being read over standard on-chip channels, attacks using laser stimulation can reveal cell states in a powered SRAM PUF [48].

Also cloning attacks have been suggested lately. One key observation is that not the entire PUF needs to be cloned in full detail; it suffices if the clone has the same challenge-response pair(s) as the original. Since Weak PUFs often have only one CRP, the clone only has to be tuned until this single CRP matches the original. It had been known for some time that the identifying tendencies of SRAM cells can indeed be shifted by directed aging using NBTI or other means [32], [5], [30]. Originally, this effect has been suggested to make the outputs of SRAM PUFs more stable. It can also be exploited by an adversary, though: In an invasive attack, he reprograms the tendency of a cell using focused ion beam circuit edit, thus effectively cloning the CRP behavior of the SRAM PUF [27].

E. History of Concept and Terminology

Historically, the concept denoted as Weak PUF in this work has been called by at least one different term: Gassend proposed the use of PUFs with a small number of fixed challenges as an internal key source under the name of a *"physically obfuscated key" (POK)* in 2003 [21]. In 2007, Guajardo et al. [25] were apparently the first to use the terms Weak and Strong PUFs, but without differentiating these two concepts in full detail. Rührmair et al. contributed to a more detailed distinction in 2009 to 2012 [69], [67], [60]. An attempt at formalizing Weak PUFs is given in 2011 by Armknecht et al. [2], who define a weak PUF as one that can be modeled from a number of challenge response pairs that fails to be exponential in any security parameter.

III. Strong PUFs

A. Characteristic Features of Strong PUFs

So-called *"Strong PUFs"* are the second major PUF type besides Weak PUFs. In opposition to the latter, they derive a more complex challenge-response behavior from the physical disorder present in the PUF. Typically, many physical components are involved in the generation of a response, and there is a very large number of possible challlenges that can be applied to the PUF. Their security features have been put down in [69], [67], [60], [70], and, more formally, in [59], [8]. In a nutshell, they can be subsumed as follows:

1) *Many challenges:* Strong PUFs have a very large number of possible challenges, ideally (but not necessarily) exponentially many challenges in some system parameter. This prevents a full read-out of all CRPs, even if an adversary holds physical possession of the PUF for considerable time.
2) *Unpredictability:* Even if an adversary knows a large subset of CRPs, he cannot extrapolate or predict the other, yet unknown CRPs.
3) *Unprotected challenge-response interface:* In all but very few applications of Strong PUFs, it is assumed that have a freely, publicly accessible challenge-response interface (or a freely accessible challenge-respons mechanism, respectively). Anyone holding physical possession of the PUF or the PUF-carrying hardware can apply arbitrary challenges to the Strong PUF and read out the corresponding responses.

Please note that all three features mark clear differences to Weak PUFs. Since the challenge-response interface of a Strong PUF is in most applications is assumed to be unprotected, no access restrictions on the PUF-responses need to be supposed. Recall from Sections II and II-D that the latter were one of the most critical assumptions in the security features of Weak PUFs. Invasive attacks on the PUF responses are therefore mostly obsolete for Strong PUFs [1]. On the other hand, the freely accessible challenge-response interface also brings about downsides: It necessarily implies that Strong PUFs must have very many CRPs to remain secure. It also enables modeling attacks on Strong PUFs, since it allows the simple collection

[1]The only exceptions are invasive attacks on internal digital signals inside the Strong PUFs itself, if such signals exist in a given Strong PUF design. Examples are XOR-based Arbiter PUFs [75], [67] or Lightweight PUFs [46].

of large subsets of CRPs. The latter attacks are irrelevant for Weak PUFs, in turn (see Sections II-D and III-D).

B. Implementation Examples

The first proposed Strong PUF is the optical PUF of Pappu et al. [53]. It consists of an optical scattering object, for example a plastic token which contains randomly distributed glass spheres. The challenge to the structure is a laser beam which is directed at the token under a selected angle and point of incidence. The resulting response is the multi-bit intereference pattern that emerges from the complex light scattering process inside the token. Pappu et al. estimate that their implementation of an optical PUF creates around 10^{10} independent CRPs [53].

The first electrical, integrated Strong PUF is the so-called Arbiter PUF [22], [75]. Its idea is to exploit the varying runtime delays in electrical components. In an Arbiter PUF architecture, electrical signals race against each other through a sequence of k stages, each of which consists of two multiplexers. The exact race path of each signal is determined by k external bits which are applied at the stages, one bit per stage. The race is called by a final arbiter element, which is implemented by a latch. Arbiter PUFs with k stages have 2^k challenges, and produce one-bit responses. Since the plain Arbiter PUF is susceptible to machine-learning based modeling attacks (see Section III-D), more sophisticated variants have been developed. They have in common that they add non-linearities in one way or the other to the standard Arbiter PUF to complicate machine learning. Examples include Feed-Forward Arbiter PUFs [39], [40], XOR Arbiter PUFs [75], [67], and the so-called Lightweight PUF [46].

Moving away from the somewhat dominant Arbiter PUF family, several alternative electrical Strong PUF designs exist, to which we would like to point interested readers: The Power Grid PUF [28]; Clock PUF [80]; Crossbar PUF [65]; and the CNN PUF, which is based on analog circuits [11], [3].

C. Applications and Error Correction

The prime application of Strong PUFs is challenge-response based identification and system authentication. The idea has been first described in a banking card scenario [53] as follows. It is assumed that the bank equips each banking card with a Strong PUF. Before the card is released to the customer, the bank applies a large number of random challenges to the PUF, and stores the resulting CRPs in a secret, internal list $\mathcal{L}$. When the card is carried by the customer to a terminal or automated teller machine, the card can identify itself by using the unique challenge-response behavior of the PUF: The bank chooses a couple of challenges from the list $\mathcal{L}$, and sends them to the terminal. The terminal applies the challenges to the Strong PUF, and returns the obtained responses to the bank. The latter compares them to the responses in the list $\mathcal{L}$; if they match, the identification was successful. Each CRP can be used only once and needs to be erased from the list subsequently.

The above identification protocol has the advantage of being extremely lightweight, and of requiring *no* resources besides the PUF on the card. Standard PUF error correction can potentially be executed outside the card (i.e., the PUF-carrying hardware), for example by the terminal or the bank itself. The protocol can also be made error tolerant by allowing a small fraction of all responses to be incorrect; in this case, no classical error correction needs to be applied at all. This constitutes an advantage compared to Weak PUFs, where perfect error correction must be accomplished inside the PUF-carrying system, making the approach less lightweight. Note that the protocol explicitly requires a Strong PUF: Since a Weak PUF only has got one (or very few) digital responses, it could be utilized in one protocol execution only.

The above protocol can be applied in any system identification scenario, and shines the most for inexpensive, lightweight systems. It can be used commercially for any forms of online identification or certification (compare [78]).

Strong PUFs have also been suggested in cryptographic applications beyond the above, basic identification scheme. Already Pappu considers a simple bit-commitment protocol that rests on the onewayness (non-invertibility) of the CRPs of his optical PUF in 2002 [53]. Van Dijk suggested a key exchange protocol based on Strong PUFs in a patent writing in 2004 [15]. The usability of Strong PUFs as a universal primitive was first demonstrated by Rührmair in 2010, who showed that oblivious transfer (and hence also any secure multi-party computatuion) can be based on Strong PUFs [61]. In 2011, Brzuska et al. [8] treated PUFs in the UC-model and lead formal proofs for the security of Strong PUF based bit commitment, oblivious transfer and key exchange. We stress, however, that the secure commercial use of plain Strong PUFs in these advanced protocols is currently under heavy research, after a number of dedicated protocol attacks has been discovered recently [62], [63], [16], [64], [17].

D. Attacks on Strong PUFs

Cloning and invasive attacks on Weak PUFs (Section II-D) appear less applicable to Strong PUFs for a number of reasons. Rather, the currently most relevant attack method for Strong PUFs are so-called *"modeling attacks"* [40], [45], [67], [70]. They assume that an adversary has collected a large number of all possible CRPs of a given Strong PUF (usually between several hundred to a few million CRPs, depending on the exact Strong PUF design). By use of numeric methods and an internal, parametric model of the PUF, the adversary then tries to extrapolate the behavior of the PUF on the other, yet unknown CRPs. Machine learning algorithms are a natural and very powerful tool to this end.

The reach of modeling attacks is surprisingly large, and a considerable number of existing electrical designs have been tackled successfully up to a certain size, including Arbiter PUFs and variants thereof [67], [70]. Only optical PUFs have resisted all modeling attacks so far. We refer the reader to existing works [67], [70] and a recent survey paper on modeling attacks [68]. Modeling attacks do not apply to Weak PUFs, since the latter have only one challenge per PUF. Therefore no extrapolation of unknown CRPs from a subset of known CRPs is applicable. One very recent trend is to combine modeling techniques with side channel information in order to boost attack performance [13], [44].

Also dedicated protocol attacks on Strong PUF schemes have been discovered recently. They differ from the above, hardware-oriented modeling attempts. We refer the interested

reader to the existing literature on this topic [62], [63], [16], [64], [50] and a recent survey paper [17].

E. History of Concept and Terminology

Historically, the structures that we call Strong PUFs today have been referred to by different names. The first Strong PUF in our sense is the optical PUF of Pappu et al. [52], [53] from 2001/02. Its input-output behavior is not just unpredictable, but also non-invertible, whence the authors originally used the term *"physical one-way function" (POWF)* for their invention. Still in 2002, Gassend et al. [22] introduced circuit-based Strong PUFs, using the names *"physical random function"* and *"physical unclonable function (PUF)"*. The term Strong PUF was then eventually suggested by Guajardo et al. [25] in 2007, but without fully detailing all its features. Rührmair et al. worked out the exact security features and the associated attack models in 2009 to 2012 [69], [67], [60]. Formal, mathematical definitions of Strong PUFs have been given by Rührmair et al. [59] in 2010 and Brzuska et al. [8] in 2011.

IV. Summary and Outlook

This survey paper presented an overview of PUFs and their applications as security primitives. The distinguishing feature of PUFs in contrast to more traditional methods is that their outputs are influenced by the random variations arising during fabrication. This new approach brings about some cost/practicality and also security upsides: PUFs allow the "storage" of keys in hardware without non-volatile memory cells, and their complex behavior promises better security against attacks. On the downside, they are generally more prone to errors and aging than classical approaches. Their intrinsically high noise levels must be compensated by dedicated error correction or protocol measures.

The two main types of PUFs are denoted Weak PUFs and Strong PUFs. Each have a variety of implementations: SRAM PUFs and variants are the most popular Weak PUF designs, while Arbiter PUFs and variants are the best investigated electrical Strong PUF architectures. The two PUF types serve distinct purposes; a Weak PUF is akin to a secret key, whereas a Strong PUF is more like a physical hash function. After almost 15 years of existence, PUFs show no signs of slowing down as a research topic, and today both Weak and Strong PUFs are already commercially available as products. The commercial and academic perspectives of the field hence appear bright.

References

[1] Ross Anderson: *Security engineering*. Wiley, 2008.

[2] F. Armknecht, R. Maes, A.-R. Sadeghi, F.-X. Standaert, C. Wachsmann: *A Formalization of the Security Features of Physical Functions*. IEEE Symposium on Security and Privacy, 2011.

[3] T. Addabbo, A. Fort, M. Di Marco, L. Pancioni, and V. Vignoli: *A 1-bit Physically Unclonable Function based on a two-neurons CNN*. IEEE International Symposium on Circuits and Systems (ISCAS'13), 2013.

[4] T. Bäck. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford University Press, USA, 1996.

[5] M. Bhargava, C. Cakir, and K. Mai. Reliability enhancement of bistable PUFs in 65nm bulk CMOS. *Hardware-Oriented Security and Trust (HOST), 2012 IEEE International Symposium on*, pages 25–30, 2012.

[6] C.M. Bishop et al. *Pattern recognition and machine learning*. Springer New York:, 2006.

[7] C. Bösch, J. Guajardo, A.-R. Sadeghi, J. Shokrollahi, and Pim Tuyls. Efficient Helper Data Key Extractor on FPGAs. In Cryptographic Hardware and Embedded Systems, pages 181–197. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[8] C. Brzuska, M. Fischlin, H. Schröder, S. Katzenbeisser. Physical Unclonable Functions in the Universal Composition Framework. *CRYPTO 2011*.

[9] Q. Chen, G. Csaba, P. Lugli, U. Schlichtmann, U. Rührmair. The Bistable Ring PUF: A New Architecture for Strong Physical Unclonable Functions. *HOST 2011*.

[10] Q. Chen, G. Csaba, P. Lugli, U. Schlichtmann, U. Rührmair. Characterization of the Bistable Ring PUF. *DATE 2012*.

[11] G. Csaba, X. Ju, Z. Ma, Q. Chen, W. Porod, J. Schmidhuber, U. Schlichtmann, P. Lugli, and U. Rührmair. Application of mismatched cellular nonlinear networks for physical cryptography. *IEEE CNNA*, 2010.

[12] I. Damgard, A. Scafuro: Unconditionally Secure and Universally Composable Commitments from Physical Assumptions. *Cryptology ePrint Archive, 2013:108*, 2013.

[13] J. Delvaux, I. Verbauwhede: *Side channel modeling attacks on 65nm arbiter PUFs exploiting CMOS device noise*. HOST 2013.

[14] S. Devadas: Physical unclonable functions and secure processors. *Invited Talk, CHES 2009*.

[15] M. van Dijk: *System and method of reliable forward secret key sharing with physical random functions.* US Patent No. 7,653,197, October 2004.

[16] M. van Dijk, U. Rührmair: *Physical Unclonable Functions in Cryptographic Protocols: Security Proofs and Impossibility Results*. IACR Cryptology ePrint Archive 2012: 228 (2012)

[17] M. van Dijk, U. Rührmair: *Protocol Attacks on Advanced PUF Protocols and Countermeasures*. Design, Automation and Test in Europe (DATE'14), 2014.

[18] Y. Dodis, R. Ostrovsky, L. Reyzin, L., A. Smith: *Fuzzy extractors: How to generate strong keys from biometrics and other noisy data*. SIAM Journal on Computing, 38(1), 97-139, 2008.

[19] http://electroiq.com/chipworks_real_chips_blog/2011/03/13/apples-a5-processor-is-by-samsung-not-tsmc/

[20] https://freedom-to-tinker.com/blog/felten/fingerprinting-blank-paper-using-commodity-scanners/

[21] B. L. P. Gassend. *Physical random functions*. MSc thesis, MIT, 2003.

[22] B. Gassend, D. Clarke, M. Van Dijk, and S. Devadas. Silicon physical random functions. *ACM CCS 2002*.

[23] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas. Controlled physical random functions. *ACSAC 2002*.

[24] B. Gassend, D. Lim, D. Clarke, M. Van Dijk, and S. Devadas. Identification and authentication of integrated circuits. *Concurrency and Computation: Practice & Experience*, 16(11):1077–1098, 2004.

[25] J. Guajardo, S. Kumar, G.J. Schrijen, and P. Tuyls. FPGA intrinsic PUFs and their use for IP protection. *CHES 2007*.

[26] C. Helfmeier, D. Nedospasov, S. Tajik, C. Boit, J.-P. Seifert: Physical Vulnerabilities of Physically Unclonable Functions. *DATE'14*.

[27] C. Helfmeier, C. Boit, D. Nedospasov, J.P. Seifert: Cloning Physically Unclonable Functions. *HOST 2013*.

[28] R. Helinski, D. Acharyya, and J. Plusquellic: A physical unclonable function defined using power distribution system equivalent resistance variations. In *Design Automation Conference, 2009. DAC '09. 46th ACM/IEEE*, pages 676–681. 2009.

[29] M. Hiller, D. Merli, F. Stumpf, and G. Sigl. Complementary IBS: Application specific error correction for PUFs. *Hardware-Oriented Security and Trust (HOST), 2012 IEEE International Symposium on*, pages 1–6, 2012.

[30] M. Hofer, C. Boehm: *An alternative to error correction for SRAM-like PUFs.* Cryptographic Hardware and Embedded Systems (CHES'10), 2010.

[31] D.E. Holcomb, W.P. Burleson, and K. Fu. Initial SRAM state as

a fingerprint and source of true random numbers for RFID tags. *Conference on RFID Security*, 2007.

[32] D. E. Holcomb, W. P. Burleson, and K. Fu. Power-up SRAM state as an identifying fingerprint and source of true random numbers. *IEEE Transactions on Computers*, 2009.

[33] D. E. Holcomb, A. Rahmati, M. Salajegheh, W. P. Burleson, and K. Fu. DRV-Fingerprinting: using data retention voltage of SRAM cells for chip identification. In *RFIDSec'12*, 2012.

[34] C. Jaeger, M. Algasinger, U. Rührmair, G. Csaba, and M. Stutzmann. Random p-n-junctions for physical cryptography. *Applied Physics Letters*, 96(172103), 2010.

[35] P. Koeberl, Ü. Kocabaş, and A.-R. Sadeghi. Memristor PUFs: a new generation of memory-based physically unclonable functions. In *DATE '13: Proceedings of the Conference on Design, Automation and Test in Europe*. March 2013.

[36] S.S. Kumar, J. Guajardo, R. Maes, G.J. Schrijen, and P. Tuyls. Extended abstract: The butterfly PUF protecting IP on every FPGA. *HOST 2008*.

[37] R. Kumar, W. Burleson: *Litho-aware and low power design of a secure current-based physically unclonable function.* IEEE International Symposium on Low Power Electronics and Design (ISLPED), 2013.

[38] P. Layman, S. Chaudhry, J. G. Norman, and J. R. Thomson. Electronic fingerprinting of semiconductor integrated circuits. (6,738,294), September 2002.

[39] J.W. Lee, D. Lim, B. Gassend, G.E. Suh, M. Van Dijk, and S. Devadas. A technique to build a secret key in integrated circuits for identification and authentication applications. *IEEE VLSI Circuits Symposium*, 2004.

[40] D. Lim. *Extracting Secret Keys from Integrated Circuits*. Msc thesis, MIT, 2004.

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

[42] K. Lofstrom and W.R. Daasch. IC identification circuit using device mismatch. *International Solid State Circuits Conference*, 2000.

[43] R. Maes, P. Tuyls, and I. Verbauwhede. Low-Overhead Implementation of a Soft Decision Helper Data Algorithm for SRAM PUFs. *Cryptographic Hardware and Embedded Security*, 2009.

[44] A. Mahmoud, U. Rührmair, M. Majzoobi, F. Koushanfar: *Combined Modeling and Side Channel Attacks on Strong PUFs.* IACR Cryptology ePrint Archive 2013: 632 (2013)

[45] M. Majzoobi, F. Koushanfar, and M. Potkonjak. Testing techniques for hardware security. In *International Test Conference (ITC)*, 2008.

[46] M. Majzoobi, F. Koushanfar, and M. Potkonjak. Lightweight secure pufs. *IEEE/ACM Int. Conf. on Computer-Aided Design*, 2008.

[47] M. Majzoobi, M. Rostami, F. Koushanfar, D.S. Wallach, and S. Devadas: Slender PUF Protocol: A Lightweight, Robust, and Secure Authentication by Substring Matching. *IEEE S&P Workshops*, 2012.

[48] D. Nedospasov, J. P. Seifert, C. Helfmeier, and C. Boit. Invasive PUF Analysis. *Fault Diagnosis and Tolerance in Cryptography (FDTC), 2013 Workshop on*, pages 30–38, 2013.

[49] NXP Semiconductors. NXP Strengthens SmartMX2 Security Chips with PUF Anti-Cloning Technology, February 2013.

[50] Rafail Ostrovsky, Alessandra Scafuro, Ivan Visconti, Akshay Wadia: Universally Composable Secure Computation with (Malicious) Physically Uncloneable Functions. EUROCRYPT 2013: 702-718

[51] Erdinç Öztürk, Ghaith Hammouri, and Berk Sunar. Towards robust low cost authentication for pervasive devices. *IEEE PerCom*, 2008.

[52] R. Pappu. *Physical One-Way Functions*. Phd thesis, MIT, 2001.

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

[54] P. Prabhu, A. Akel, L. Grupp, W.K. Yu, G. Suh, E. Kan, and S. Swanson. Extracting Device Fingerprints from Flash Memory by Exploiting Physical Variations. *Proceedings of the 4th International Conference on Trust and Trustworthy Computing*, 2011.

[55] M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. *IEEE international conference on neural networks*, 1993.

[56] S. Rosenblatt, S. Chellappa, A. Cestero, N. Robson, T. Kirihata, and S. S. Iyer. A Self-Authenticating Chip Architecture Using an Intrinsic Fingerprint of Embedded DRAM. *Solid-State Circuits, IEEE Journal of*, (99):1–10, 2013.

[57] U. Rührmair. Oblivious transfer based on physical unclonable functions (extended abstract). *TRUST 2010.* LNCS Vol. 6101, Springer, 2010.

[58] U. Rührmair: *PUFs at a Glance*. Design, Automation and Test in Europe (DATE'14), 2014.

[59] U. Rührmair, H. Busch, S. Katzenbeisser: Strong PUFs: Models, Constructions and Security Proofs. In A.-R. Sadeghi, P. Tuyls (Editors): *Towards Hardware Intrinsic Security: Foundation and Practice.* Springer, 2010.

[60] U. Rührmair, S. Devadas, F. Koushanfar: Security based on Physical Unclonability and Disorder. In M. Tehranipoor and C. Wang (Editors): Introduction to Hardware Security and Trust. Springer, 2011.

[61] U. Rührmair: *Oblivious Transfer based on Physical Unclonable Functions (Extended Abstract)*. TRUST 2010.

[62] U. Rührmair, M. van Dijk: *Practical Security Analysis of PUF-based Two-Player Protocols.* CHES 2012.

[63] U. Rührmair, M. van Dijk: *On the Practical Use of Physical Unclonable Functions in Oblivious Transfer and Bit Commitment Protocols*. Journal of Cryptographic Engineering (JCEN), 2013.

[64] U. Rührmair, M. van Dijk: *PUFs in Security Protocols: Attack Models and Security Evaluations*. IEEE Symposium on Security and Privacy (Oakland'13), 2013.

[65] U. Rührmair, C. Jaeger, M. Bator, M. Stutzmann, P. Lugli, G. Csaba. Applications of high-capacity crossbar memories in cryptography. *IEEE Transactions on Nanotechnology*, 2011.

[66] U. Rührmair, C. Jaeger, C. Hilgers, M. Algasinger, G. Csaba, M. Stutzmann. Security applications of diodes with unique current-voltage characteristics. Financial Cryptography and Data Security (FC), 2010.

[67] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, J. Schmidhuber. Modeling Attacks on Physical Unclonable Functions. *ACM CCS 2010*.

[68] U. Rührmair, J. Sölter: *PUF Modeling Attacks: An Introduction and Overview*. Design, Automation and Test in Europe (DATE'14), 2014.

[69] U. Rührmair, J. Sölter, F. Sehnke. On the Foundations of Physical Unclonable Functions. Cryptology ePrint Archive, 2009:277, 2009.

[70] U. Rührmair, J. Sölter, F. Sehnke, X. Xu, A. Mahmoud, V. Stoyanova, G. Dror, J. Schmidhuber, W. Burleson, S. Devadas: *PUF Modeling Attacks on Simulated and Silicon Data*. IEEE Transactions on Information Forensics and Security (IEEE T-IFS), 2013.

[71] H.P.P. Schwefel. *Evolution and Optimum Seeking: The Sixth Generation*. John Wiley & Sons, Inc. New York, NY, USA, 1993.

[72] P. Simons, E. van der Sluis, and V. van der Leest. Buskeeper PUFs, a promising alternative to D Flip-Flop PUFs. *HOST 2012*, pages 7–12, 2012.

[73] J. Sölter. *Cryptanalysis of Electrical PUFs via Machine Learning Algorithms*. MSc thesis, Technische Universität München, 2009.

[74] Y. Su, J. Holleman, and B. Otis. A 1.6 pj/bit 96% stable chip-id generating circuit using process variations. *International Solid State Circuits Conference*, 2007.

[75] G.E. Suh, S. Devadas. Physical unclonable functions for device authentication and secret key generation. *DAC 2007.*

[76] P. Tuyls, G. J. Schrijen, B. Škorić, J. van Geloven, N. Verhaegh, R. Wolters. Read-proof hardware from protective coatings. *CHES 2006*.

[77] P. Tuyls, B. Skoric. Strong Authentication with PUFs. *In: Security, Privacy and Trust in Modern Data Management, M. Petkovic, W. Jonker (Eds.)*, Springer, 2007.

[78] http://www.verayo.com/

[79] X. Xu, W. Burleson. Hybrid Side-Channel / Machine- Learning Attacks on PUFs: A New Threat? *DATE 2014*.

[80] Y. Yao, M. Kim, J. Li, I. L. Markov, and F. Koushanfar. ClockPUF: Physical Unclonable Functions based on clock networks. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013*, pages 422–427. 2013.

[81] M-D. Yu and S. Devadas. Secure and Robust Error Correction for Physical Unclonable Functions. *Design & Test of Computers, IEEE*, 27(1):48–65, 2010.

[82] M.-D. Yu, D. M'Raïhi, R. Sowell, S. Devadas: Lightweight and Secure PUF Key Storage Using Limits of Machine Learning. *CHES 2011*.