Brand and IP Protection with Physical Unclonable Functions

Jorge Guajardo, Sandeep S. Kumar
Information and System Security Dept.
Philips Research Europe
5656 AE, Eindhoven, Netherlands
Email: {jorge.guajardo,sandeep.kumar}@philips.com

Geert-Jan Schrijen and Pim Tuyls
Intrinsic ID Business Unit
Philips Research Europe
5656 AE, Eindhoven, Netherlands
Email: {geert.jan.schrijen,pim.tuyls}@philips.com

***Abstract*— In this paper we provide an overview of Physical Unclonable Functions and explain why they are a very valuable technology to protect a company's IP and hence at the same time its brand. Physical Unclonable Functions are unclonable physical structures that map challenges to responses. They inherit their unclonability from the (deep sub-micron) process variations during manufacturing. They can be turned into a useful tool to generate very secure secret keys in ICs and to provide keys to protect valuable IP of fabless IC companies, IP Vendors and design houses. We will present several examples and explain cryptographic algorithms and protocols to use them in IP protection applications.**

I. Introduction

Modern production techniques in the semiconductor industry have changed with the motto of having shorter time-to-market with the least price. This has driven design houses to reuse internal/external IP and use production facilities at centralized locations like foundries shared by multiple companies. Such outsourcing has opened up new possibilities for revenue by IP licensing. However, an open environment makes brand and IP protection harder, since this would be traditionally done by closely guarding the design and production within the organization. Notice that it is estimated that as much as 10% of all high-tech products sold globally are counterfeit which leads to a conservative estimate of US$100 billion of revenue lost. In this paper, we present an overview on the use of Physical Unclonable Functions (PUFs) to prevent counterfeiting of devices and IP. The remainder of this contribution is organized as follows. Section II gives an introduction to PUFs. Different PUF constructions are presented in Section III. In Section IV, we show various PUF applications such as IP protection, a novel remote service/feature activation technique, secret-key storage and authentication. We end with some conclusions.

II. Physical Unclonable Functions Preliminaries

The term function in mathematics is used to express the dependency f between two variables, say C and R, one of which is given (C) and the other (R) which is produced as a result of applying f to C. Notice that the relation between C and R can be defined via a mathematical formula, a graph, a table listing values of R corresponding to C, etc. In 2001, Pappu et al. [1], [2], introduced the concept of Physical Unclonable Functions (PUFs) or Physical Random Functions, which are functions where the relationship between input (or *challenge*) C and output (or *response*) R is defined via a physical system. Notice that the physical system has the additional properties of being random and unclonable. The system's unclonability originates from random variations in a device's manufacturing process, which even the (legitimate) manufacturer can not control. In their most general form, PUFs can accept a large number of inputs or challenges and output corresponding responses. Such a pair of a stimulus C and a response R is called a *challenge-response* pair (CRP). Depending on the PUF construction and on the number of stimuli that we can use to challenge a PUF, we distinguish between two different situations. First, we assume that there is a large number of challenge response pairs (C_i, R_i), $i = 1, \ldots, N$ available for the PUF; i.e. a strong PUF has so many CRPs such that an attack (performed during a limited amount of time) based on exhaustively measuring the CRPs only has a negligible probability of success and in particular, $1/N \approx 2^{-k}$ for large $k \approx 100$ [1], [3]. We refer to this case as strong PUFs. To this PUF class belong optical PUFs [1], [2] and certain silicon PUFs [4], [5]. If the number of different CRPs N is rather small, we refer to it as a weak PUF. In this PUF class, we can include the constructions in [6], [7]. We will describe in more detail each of these PUF constructions in Sect. III. We would like to remark that the term *weak* only refers to the number of challenges and not to the PUF's unclonability properties.

A characteristic typical of all PUFs is that their responses are noisy. In fact, PUFs can be modeled as a noisy communication channel. In other words, when a PUF is challenged with C_i a response R_i' which is a noisy version of R_i is obtained. In cryptographic applications, where the PUF response is used as a source of secret-key material this is in fact unacceptable. Thus, [8], [9] introduce the concept of fuzzy extractor or helper data algorithm to work around the noisy nature of physical measurements, typical of PUF applications. Section II-B provides an overview of fuzzy extractors.

A. Assumptions

From a security perspective, we make the following assumptions: (1) it is assumed that a PUF response R_i (to a challenge C_i) gives only a negligible amount of information on another response R_j (to a different challenge C_j) with $i \neq j$;

and (2) without having the corresponding PUF at hand, the probability of coming up with the response R_i corresponding to a challenge C_i, is negligible. In addition, it is desirable (although is not imperative) for PUFs to be tamper evident. In other words, if an attacker tries to investigate the PUF to obtain detailed information of its structure or to obtain responses to certain challenges, the PUF is destroyed, i.e., the PUF's challenge-response behavior is changed substantially. Finally, it is also often assumed that the PUF response is only available within the device containing the PUF and only to authorized parties (or hardware). This assumption is particularly important in protocols in which the PUF is used to derive a secret value or key.

B. Fuzzy Extractor or Helper Data Algorithm

As previously mentioned, PUF responses are noisy and not fully random. Thus, a Fuzzy Extractor or Helper Data Algorithm [9], [8] is required to extract one (or more) secure keys from the PUF responses. In the following, we provide the intuition behind the algorithms. A fuzzy extractor requires two basic primitives: (i) *Information Reconciliation* or error correction and (ii) *Privacy Amplification* or randomness extraction, which guarantees an output which is very close to being a uniformly distributed random variable. In order to implement those two primitives, helper data W are generated during the *enrollment or registration phase*. Later during the *key reconstruction or authentication* phase, the key is reconstructed based on a noisy measurement R_i' and the helper data W. During the enrollment phase (carried out in a trusted environment), a probabilistic procedure called Gen is run. It takes as input a PUF response R and produces as output a key K and helper data W: $(K, W) \leftarrow \mathsf{Gen}(R)$. In order to generate the helper data W, an error correcting code $\mathcal{C}$ is chosen such that at least t errors can be corrected. The number of errors to be corrected depends on the particular application and on the PUF properties. Once an appropriate code has been chosen, the helper data W is generated by first choosing a random code word C_S from $\mathcal{C}$ and computing $W_1 = C_S \oplus R$. Furthermore a universal hash function [10] h_i is chosen at random from a set $\mathcal{H}$ and the key K is defined as $K \leftarrow h_i(R)$. The helper data is then defined as $W = (W_1, i)$. During the key reconstruction phase a procedure called Rep is run. It takes as input a noisy response R' and helper data W and reconstructs the key K (if R' originates from the same source as R) *i.e.* $K \leftarrow \mathsf{Rep}(R', W)$. Reconstruction of the key is achieved by computing $C_s' = W_1 \oplus R'$, decoding C_s' to C_s via the decoding algorithm of $\mathcal{C}$, recovering $R = C_s \oplus W_1$, and finally computing $K = h_i(R)$.

III. PUF Constructions

In this section, we describe known PUF constructions including: Optical and Silicon PUFs, Coating PUFs, and SRAM-based PUFs. Different PUFs have different properties and thus some will be better suited for certain applications than others. Nevertheless, their unclonability and randomness properties remain key characteristics of all constructions.

A. Optical PUFs

Pappu et al. [1], [2] introduced the idea of a Physical One-Way Function. They used a bubble-filled transparent epoxy wafer and shone a laser beam through it leading to a speckle pattern. The speckle pattern depends on the material of the wafer, its thickness, the wavelength of the light beam used to generate the speckle pattern, and on the angle of the incident laser beam [11]. This type of optical PUF is hard to use in the field because of the difficulty to have a tamper resistant measuring device. On the other hand, optical PUFs are notorious for their large number of CRPs.

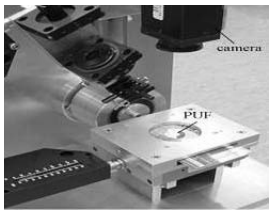

Fig. 1. Optical PUF measurement set-up

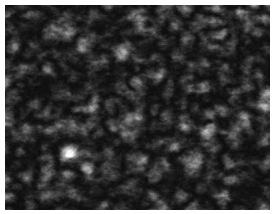

Fig. 2. Speckle pattern from a silicon surface

B. Silicon PUFs

Gassend et al. introduce Silicon Physical Random Functions (SPUF) [4] which use manufacturing process variations in ICs with identical masks to uniquely characterize each chip. The statistical delay variations of transistors and wires in the IC were used to create a parameterized self oscillating circuit to measure frequency which characterizes each IC. SPUFs are very sensitive to environmental variations like temperature and voltage. Thus, Lim et al. [12] introduce *arbiter based* PUFs which use a differential structure and an arbiter to distinguish the difference in the delay between the paths. Gassend et al. [5] also define a Controlled Physical Random Function (CPUF) which can only be accessed via an algorithm that is physically bound to the randomness source in an inseparable way. This control algorithm can be used to measure the PUF but also to protect a PUF from external attacks. Recently, Su et al. [13] present a custom built circuit array of cross-coupled NOR gate latches to uniquely identify an IC. Here, small transistor threshold voltage V_t differences that are caused due to process variations lead to a mismatch in the latch to store a 1 or a 0.

C. Coating PUFs

In [6], Tuyls et al. present coating PUFs in which an IC is covered with a protective matrix coating, doped with random dielectric particles at random locations. The IC also has a top metal layer with an array of sensors to measure the local capacitance of the coating matrix that is used to characterize the IC. The measurement circuit is integrated in the IC, making it a controlled PUF. Figure 3 shows a schematic diagram of the PUF construction. In Fig. 3, it is possible to see how the upper metal layer contains aluminum sensor structures (Al) that are used to measure the local capacitance of the coating. It is shown in [6] that it is possible to extract up to three key bits from each sensor in the IC.

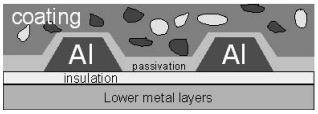

Fig. 3. Schematic cross-section of a Coating PUF IC.

D. FPGA Intrinsic PUFs and SRAM Memories

A disadvantage of previous approaches is the use of custom built circuits or the modification of the IC manufacturing process to generate a reliable PUF. In [7], the authors approach the problem by identifying an *Intrinsic* PUF which is defined as a PUF already present in the device. In particular, [7] noticed that the start-up values of SRAM memories[1], which are widely available in almost every computing device including modern FPGAs, can be used as an Intrinsic PUF. It is shown in [7] that due to intrinsic device variations during the manufacturing of SRAM memory cells, an SRAM cell will start in the same state upon power-up with high probability. On the other hand, different SRAM cells will behave randomly and independently from each other. In the case of SRAM start-up values, the authors in [7] consider as a challenge a range of memory locations within a SRAM memory block. The size of the range will depend on the number of bits that need to be derived for purposes of identification, authentication, or key generation. As in any device property which aims to be used as a PUF,

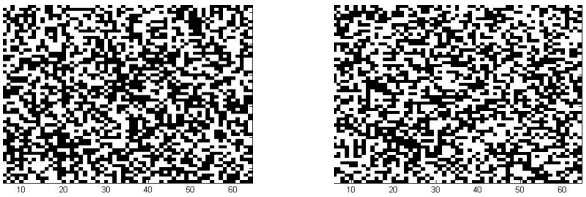

Fig. 4. Memory map of two RAM blocks corresponding to two different FPGAs. Memory cells with a start-up value of '0' (resp. '1') are represented as black squares (resp. white squares). On average, we observe a 50% difference on the number of ones and zeros for two different FPGAs.

SRAM startup values should have good statistical properties and be robust over time and to temperature variations. These properties were studied in [7] and a maximum fractional Hamming distance of 12% was found when compared to a reference measurement performed at 20°C. This error rate can be efficiently corrected via error correcting codes. Finally, in order to be able to identify as many devices as possible (and minimize the possibility of counterfeits), the fractional Hamming distance between bit strings of *different* SRAM blocks (and different FPGAs) should be close to 50%. Figure 4 shows a representation of the memory contents of two different FPGAs after start-up. The result is that about 50% of the bits are different even after measuring seventeen different SRAM blocks from different FPGAs.

IV. Applications

Since their introduction in [1], [2] PUFs have received considerably attention from the security community because of the unclonability and randomness properties inherent to them. This section explores four of the most interesting applications presented in the literature.

IP Protection. As mentioned in the introduction, IP protection is a real concern for many IP providers. This has been recognized since early 2002 by Kean [15] who provides solutions based on IP encryption. Kean is also the first to list the parties involved in the IP protection chain. These include: the end user, the FPGA customer, the system integrator or designer (SYS), the hardware IP-Provider or core vendor (IPP), the hardware (FPGA) manufacturer (HWM) or vendor, the CAD software vendor, and a Trusted Third Party (TTP) [15]. Simpson and Schaumont [16] observed, however, that certain problems cannot be easily solved via bitstream encryption. In particular, [16] propose a solution based on the use of PUFs if the aim is to authenticate the hardware platform on which third party intellectual property and software modules run. While the solution is based on PUFs, the authors in [16] do not provide an actual PUF construction. More recently, [7] proposed a new PUF construction and simplify and improve the protocols of [16]. One major advantage of a protocol in [7] is that the TTP does not get any knowledge of the IP contents, unlike the protocols of [16]. Figure 5 shows one of the protocols proposed in [7]. The basic idea in Figure 5 is to use the PUF

1) **Assumptions:**
   - Communication channel between SYS-TTP and TTP-IPP are authenticated and secure
   - Fully trusted TTP
2) **Enrollment Protocol:**
   a) HWM $\rightarrow$ TTP : $ID_{HW}||\{\{C_1, R_1\}, \ldots \{C_n, R_n\}\}$
3) **Authentication Protocol:**
   a) SYS $\rightarrow$ TTP : $ID_{SW}||ID_{HW}$
   b) TTP $\rightarrow$ IPP : ID_{SW}
   c) TTP $\leftarrow$ IPP : SW
   d) TTP : $D \leftarrow \mathtt{Enc}_{R_i}(SW||ID_{SW})$
   e) SYS $\leftarrow$ TTP : $C_i||C_j||D||\mathtt{MAC}_{R_j}(C_i||C_j||D)$

Fig. 5. Authentication protocol of [7] with fully trusted TTP

as a source for secret-key material, both for encryption and MAC-based authentication. As explained in [7], the MAC (Message Authentication Code) is necessary to authenticate the origin of the IP, since encryption does not provide sufficient authentication guarantees. Notice that the protocols presented in [16], [7] are based on symmetric-key primitives. In [17], the authors notice that by incorporating public-key primitives, it is not necessary anymore to make the secret-key available outside the device being authenticated. This results in increased security guarantees as only an attacker that can successfully tamper with the device (and the PUF) will have access to the encrypted IP (other than the IP creator).

Remote Service/Feature Activation. Closely related to IP protection, remote service activation refers to the ability to enable certain features of a product once the product has been sold or is in possession of an external (and often) untrusted

[1]A similar idea has been independently presented in [14].

party. In this case, the aim is to allow only parties with the right credentials to be able to activate certain features of a product. Based on our discussion on fuzzy extractors in Sect. II-B, if one is to reconstruct the key K based on a noisy response R', it is necessary to provide the procedure Rep with the helper data W. Thus, W can be used as a feature activation token even after the device is in the hands of an untrusted party. In addition, notice that thanks to the way in which the key K is derived no information about the key is leaked by the helper data W. Finally, W is specific to each PUF instance and, thus, to each device. In particular, the helper data W is specific to each device. Thus, enabling a feature after obtaining W_i for device i does not allow a user to activate the same feature for device j. We refer to [9], [8] for further discussions regarding security of different fuzzy extractor constructions.

Secret-Key Storage. A key observation in [6] is that the coating can be used to store keys (rather than as a challenge-response repository as in previous works) and that these keys are not stored in memory. Rather, whenever an application requires the key, the key is generated on the fly. This makes it much more difficult for an attacker to compromise key material in security applications. Finally, Tuyls et al. [6] show that active attacks on the coating can be easily detected, thus, making it a good countermeasure against probing attacks.

Authentication Via Challenge-Response (CR) Pairs. Challenge-response authentication techniques are based on the idea that a claimant or prover proves to a verifier knowledge of a secret without expressly revealing the secret. The authentication is performed with the help of a time varying value called the challenge usually chosen at random by the verifier. The response of the prover depends then on the challenge and on his/her secret value. Pappu [1] was the first to propose using PUFs integrated into a CR protocol for authentication purposes. The basic idea is to go through an enrollment process (performed in a secure facility) in which a number of challenges and corresponding PUF responses are stored in a secure database. At a later stage, the prover, who wants to gain access to a service, contacts the verifier, who then sends the prover a challenge from the database, the prover challenges its PUF, records the PUF response and forwards it to the verifier. The verifier can then check if the response is the same one as the one stored in the database. If the check is positive, the verifier grants access to the requested service. Notice that this protocol assumes that each challenge is used once (otherwise replay attacks are possible). It is also assumed, as pointed out in Sect. II, that without access to the right PUF, the probability of generating the expected response is negligible.

V. Conclusion

In this paper, we provided an overview of PUFs and their use for brand and IP protection. We described different PUF constructions especially the intrinsic PUF which requires no modification to the hardware. We showed how the PUF constructions can be used for IP protection, secret-key storage and authentication. We also presented a novel technique for remote service/feature activation in devices using PUFs.

References

[1] R. S. Pappu, "Physical one-way functions," Ph.D. dissertation, Massachusetts Institute of Technology, March 2001, available at http://pubs.media.mit.edu/pubs/papers/01.03.pappuphd.powf.pdf.

[2] R. S. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions," *Science*, vol. 297, no. 6, pp. 2026–2030, 2002, available at http://web.media.mit.edu/~brecht/papers/02.PapEA.powf.pdf.

[3] B. Skoric, P. Tuyls, and W. Ophey, "Robust Key Extraction from Physical Uncloneable Functions," in *Applied Cryptography and Network Security — ACNS 2005*, ser. LNCS, J. Ioannidis, A. D. Keromytis, and M. Yung, Eds., vol. 3531, June 7-10, 2005, pp. 407–422.

[4] B. Gassend, D. E. Clarke, M. van Dijk, and S. Devadas, "Silicon physical unknown functions," in *ACM Conference on Computer and Communications Security — CCS 2002*, V. Atluri, Ed. ACM, November 2002, pp. 148–160.

[5] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Controlled Physical Random Functions," in *ACSAC '02: Proceedings of the 18th Annual Computer Security Applications Conference*. Washington, DC, USA: IEEE Computer Society, 2002, p. 149.

[6] P. Tuyls, G.-J. Schrijen, B. Skoric, J. van Geloven, N. Verhaegh, and R. Wolters, "Read-Proof Hardware from Protective Coatings," in *Cryptographic Hardware and Embedded Systems — CHES 2006*, ser. Lecture Notes in Computer Science, vol. 4249. Springer, October 10-13, 2006, pp. 369–383.

[7] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, "FPGA Intrinsic PUFs and Their Use for IP Protection," in *Cryptographic Hardware and Embedded Systems — CHES 2007*, ser. LNCS, P. Paillier and I. Verbauwhede, Eds., vol. 4727. Springer, September 10-13, 2007, pp. 63–80.

[8] J.-P. M. G. Linnartz and P. Tuyls, "New Shielding Functions to Enhance Privacy and Prevent Misuse of Biometric Templates," in *Audio-and Video-Based Biometrie Person Authentication — AVBPA 2003*, ser. LNCS, J. Kittler and M. S. Nixon, Eds., vol. 2688. Springer, June 9-11, 2003, pp. 393–402.

[9] Y. Dodis, M. Reyzin, and A. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," in *Advances in Cryptology —- EUROCRYPT 2004*, ser. LNCS, C. Cachin and J. Camenisch, Eds., vol. 3027. Springer-Verlag, 2004, pp. 523–540.

[10] L. Carter and M. N. Wegman, "Universal Classes of Hash Functions," *J. Comput. Syst. Sci.*, vol. 18, no. 2, pp. 143–154, 1979.

[11] P. Tuyls, B. Skoric, S. Stallinga, A. H. M. Akkermans, and W. Ophey, "Information-theoretic security analysis of physical uncloneable functions," in *Financial Cryptography — FC 2005*, ser. LNCS, A. S. Patrick and M. Yung, Eds., vol. 3570. Springer, February 28 - March 3, 2005, pp. 141–155.

[12] 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 (VLSI) Systems*, vol. 13, no. 10, pp. 1200–1205, October 2005. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1561249

[13] Y. Su, J. Holleman, and B. Otis, "A 1.6pJ/bit 96% Stable Chip-ID Generating Cicuit using Process Variations," in *ISSCC '07: IEEE International Solid-State Circuits Conference*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 406–408.

[14] 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 07, July 11-13, 2007.

[15] T. Kean, "Cryptographic rights management of FPGA intellectual property cores," in *ACM/SIGDA tenth international symposium on Field-programmable gate arrays — FPGA 2002*, 2002, pp. 113–118.

[16] E. Simpson and P. Schaumont, "Offline Hardware/Software Authentication for Reconfigurable Platforms," in *Cryptographic Hardware and Embedded Systems — CHES 2006*, ser. LNCS, L. Goubin and M. Matsui, Eds., vol. 4249. Springer, October 10-13, 2006, pp. 311–323.

[17] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, "Physical Unclonable Functions, FPGAs and Public-Key Crypto for IP Protection," in *International Conference On Field Programmable Logic and Applications — FPL 2007*, W. Najjar and K. Bertels, Eds. IEEE, August 27-29, 2007.