Anti-Counterfeiting with Hardware Intrinsic Security

Vincent van der Leest and Pim Tuyls
Intrinsic-ID, Eindhoven, The Netherlands
http://www.intrinsic-id.com

***Abstract*—Counterfeiting of goods and electronic devices is a growing problem that has a huge economic impact on the electronics industry. Sometimes the consequences are even more dramatic, when critical systems start failing due to the use of counterfeit lower quality components. Hardware Intrinsic security (i.e. security systems built on the unique electronic fingerprint of a device) offers the potential to reduce the counterfeiting problem drastically. In this paper we will show how Hardware Intrinsic Security (HIS) can be used to prevent various forms of counterfeiting and over-production. HIS technology can also be used to bind software or user data to specific hardware devices, which provides additional security to both soft- and hardware vendors as well as consumers using HIS-enabled products. Besides showing the benefits of HIS, we will also provide an extensive overview of the results (both scientific and industrial) that Intrinsic-ID has achieved studying and implementing HIS.**

I. Introduction

Cloning, theft of service or sensitive data and tampering have become serious threats to the revenue and reputation of hardware vendors. To protect their products against these attacks hardware security, based on cryptographic primitives using keys, can be used. These keys are usually stored somewhere in the hardware (in non-volatile memory). Hence, the strength of the security depends on the effort required from attackers to compromise them. Tools for attacking hardware have become very advanced and can allow an attacker to read the content of non-volatile memory. This has decreased the protection provided by storing a key in memory to a minimum. Finally, non-volatile memory is generally quite costly to (securely) integrate in the IC manufacturing process.

In order to deal with these downsides of key storage in non-volatile memory Hardware Intrinsic Security (HIS) can be used. This term is used for security mechanisms that are based on the intrinsic hardware properties of an electronic device, in particular properties of SRAM. These properties are comparable to human biometrics and can be seen as the unique fingerprint of an electronic circuit. In particular security mechanisms using Physically Unclonable Functions (PUFs) belong to the category of HIS mechanisms and were introduced by Pappu [1] in 2001.

Due to deep-submicron manufacturing process variations every transistor in an integrated circuit (IC) has slightly different physical properties that lead to measurable differences in terms of its electronic properties (e.g. threshold voltage, gain factor). Since these process variations are uncontrollable during manufacturing, the physical properties of a device cannot be copied or cloned. It is very hard, expensive and economically not viable to purposely create a device with a given electronic fingerprint. A PUF implementation requires an electronic circuit that measures the responses of hardware to certain given inputs or challenges. These responses depend on the unique physical properties of the device. Hence PUFs are functions which are easy to challenge and whose response is easy to measure, but very hard to reproduce by construction.

978-3-9815370-0-0/DATE13/

PUF circuits can be used to uniquely identify hardware devices. This can be achieved either by simply measuring its hardware intrinsic properties or by using these properties to derive a device unique cryptographic key. This implies that keys can be stored"without storing them" and hence are present in the device only during a minimal time window (when the PUF is challenged) and therefore not vulnerable to attacks on non-volatile storage.

In order to be able to uniquely identify a device, a PUF must both be reliable and unique. By reliable we mean the fact that one is able to reproduce the same behaviour of the function when challenged with the same input over and over again. The behavioural characteristics of electronic components depend on the environment they are exposed to, namely in terms of parameters such as the ambient temperature, the voltage ramp-up curves, high and low voltage supply, and electromagnetic interference. It is of crucial importance that the function has a stable behaviour across a range of environmental conditions. Another important aspect of reliability is the fact that the electronic component has a long lifetime (i.e. that it will not change its behaviour after ten or twenty years). Typically it is observed that PUFs exhibit a noisy behaviour; this means the electronic read-out circuitry must include some type of error correction process to stabilize the PUF responses both over environmental conditions and over time. The second important aspect when using a PUF for identification purposes is that its responses should uniquely identify the device. By this we mean for instance that one device's key will never be the same as another device's key. Specifically when used for key generation a PUF should also guarantee that all generated device keys are distributed uniformly at random.

This paper will provide an overview of work and published results on HIS. The main focus of the paper will be on performing secure key storage using SRAM PUFs. However,

other PUF types and application scenarios will also receive appropriate attention.

A. Related Work

Pappu [1] introduced the concept of PUFs in 2001 under the name Physical One-Way Functions. The proposed technology was based on obtaining a response (scattering pattern) when shining a laser on a bubble-filled transparent epoxy wafer. In 2002 the first physical randomness function for silicon devices was introduced by Gassend et al. [2]. This function makes use of the manufacturing process variations in ICs, with identical masks, to uniquely characterize each IC. For this purpose the frequency of ring oscillators were measured. Using this method (now known as a Ring Oscillator PUF), they were able to characterize ICs. In 2004 Lee et al. [3] proposed another PUF that is based on delay measurements, the Arbiter PUF.

Besides intrinsic PUFs based on delay measurements a second type of PUF in ICs is known: the memory-based PUF. These PUFs are based on the measurement of start-up values of memory cells. This memory-based PUF type includes SRAM PUFs, which were introduced by Guajardo et al. in 2007 [4]. Furthermore, so-called Butterfly PUFs were introduced in 2008 by Kumar et al. [5], D Flip-Flop PUFs by Maes et al. [6] in 2008, and recently Buskeeper PUFs by Simons et al. [7] in 2012.

B. Organization of Publication

After the introduction, Section II provides information on the background of SRAM PUF technology. On the one hand this consists of a description of SRAM PUF technology, on the other hand an overview of applications for which the technology can be used. In Section III the two main properties of PUFs are described. Also, an overview of work performed on testing these properties is provided. Section IV is used to show an example design of a security module based on HIS technology and the paper is concluded in Section V.

II. Background on SRAM PUFs

This section will provide background information on PUFs, with a specific focus on SRAM PUFs. This information consists of two components. Firstly, the technology behind SRAM PUFs will be described. Secondly, an overview will be provided of several different application for which SRAM PUFs can be used.

A. SRAM PUF Technology

The principle of SRAM PUF operation is based on the start-up value of SRAM memory cells and was first published by Guajardo et al. in [4]. The initial state of each SRAM cell bit is a function of process variation due to the manufacturing process. The stabilization of each bit depends on the threshold voltage mismatch between local devices.

A 6T-SRAM cell, consisting of cross coupled inverters and access transistors, is presented in Figure 1. The stable states of the indicated SRAM cell are Q'Q=01 and Q'Q=10. When Vdd is not applied to this cell (cell is not powered)

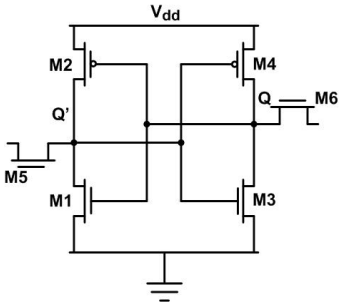

Fig. 1. The SRAM 6T cell

both nodes, Q' and Q, are low. When the power supply is turned on, the T2 and T4 transistors conduct and the nodes Q' and Q are charged while the circuit is approaching the metastable point. Supposing that Q' charges faster, and node Q gets discharged through T3, the cell stabilizes to the state of Q'Q=10. The two possible stable states of 6T-SRAM cell are highlighted in Figure 2. The state "00" is the state when the circuit is switched off and "11" is the intermediate state before stabilization to "01" or "10".

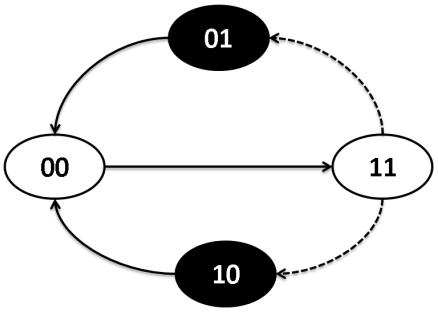

Fig. 2. The two highlighted stable states of 6T-SRAM cell (01 and 10)

Given that every SRAM cell will start-up independently, with a value determined by the uncontrollable process variations during manufacturing, a fingerprint consisting of 0 and 1 values appears in the SRAM memory after Vdd is applied. Studies have shown that this fingerprint is stable when powering the same SRAM multiple times and unique when comparing fingerprints of different SRAM memories (for more details see Section III).

B. SRAM PUF Applications

PUFs can be used for many different security purposes. The main example application is this section is secure key storage. However, other examples are also provided in the overview below.

1) Secure Key Storage: A very common purpose for PUFs are their use in secure key storage implementations [8]. In secure key storage we distinguish two phases (see also Figure 3): Enrollment and Reconstruction.

Enrollment: During "Enrollment" the key is programmed into a device. Hence this can be seen as the key programming phase for other secure key storage mechanisms. To do this,

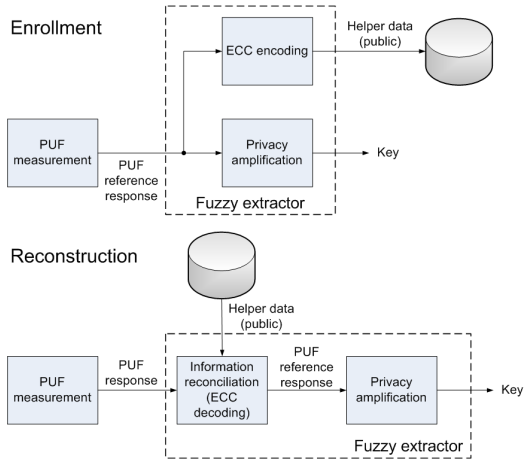

Fig. 3. Enrollment and Key Reconstruction for the described PUF model.

the response of the targeted PUF is measured. This response is called the reference PUF response and is the input of the Fuzzy Extractor [9–11]. The Fuzzy Extractor (FE, also known as Helper Data Algorithm) derives a cryptographic key from this reference response and computes helper data. In the "Reconstruction" phase, the helper data enables FE to reconstruct the exact same ("programmed") cryptographic key from a response of this specific PUF. The helper data is stored in non-volatile memory attached to the device and is public information.

Reconstruction: In the "Reconstruction" phase the same PUF is measured again and its response is input for FE. The FE uses the stored helper data and the new response to reconstruct the cryptographic key that was "programmed" during "Enrollment". If the measured PUF response is close enough to the reference response, the original key is successfully reconstructed.

Fuzzy Extractor: The two main steps that FE performs to derive a cryptographic key from a PUF response are:

- Information reconciliation: Perform error correction on a measured PUF response using the helper data.
- Privacy amplification: Assuming that an attacker has partial information on the PUF response (because of information from helper data), compress the resulting string into a cryptographic key with maximum entropy (hence maximum uncertainty for the attacker).

Anti-counterfeiting: The described key storage mechanism possesses inherent anti-counterfeiting properties:

- The key is never stored in the device (only constructed when required). This means that an attacker will be unable to extract the key from the device by (invasively) reading memory. Also since the key is only available when required, the window of opportunity for an attacker is extremely small.
- The helper data does not contain information about he key. An attacker can freely read this publicly stored information without obtaining any knowledge on the key.
- A key cannot be copied from one device to another and is therefore uniquely bound to a specific device. This is because the key is obtained by combining the helper data with the device specific PUF response. Since this PUF response is unclonable, the key cannot be copied. Only copying helper data from one device to another will not result in obtaining the key, since the PUF response is different for every device.

2) Random Number Generation: An important building block for many cryptographic systems is a random number generator. Random numbers are required in these systems, because they are unpredictable for potential attackers. In cryptographic systems where PUFs are used for authentication or secure key storage, an interesting source of randomness is readily available. This source is the noise, which occurs on every PUF response. Utilizing this noise in a proper manner can result in a random number generator based on PUFs [12].

3) Device Authentication: Besides using the hardware fingerprint of an SRAM PUF for secure key storage, it can also be used in device authentication scenarios. Using this device specific fingerprint "vendor approved" ICs can be registered in a database. This way overproduced and counterfeited hardware can be identified, since their fingerprints will not be in the database. Only chips that have been approved by the vendor will be activated and can be used by customers [13].

4) Hardware/Software Binding: When a key has been derived from a PUF, this key can be used for many different security applications. One of these applications is Hardware/Software binding [14]. In case of HW/SW binding the key from the PUF is used to bind software to specific hardware by encrypting the software with the unique key, which has been derived from the SRAM PUF. This way it will become impossible for an attacker to copy the software to another device, since this second device does not have the same PUF and therefore not the same unique key for decrypting the software successfully.

III. PUF Properties

In order to be able to use PUFs in security applications, like the use cases described in the previous section, they should possess two properties that determine the quality of these PUFs. These properties are reliability and uniqueness. In this section we will describe these two properties in more detail and provide an overview of work that has been published in order to demonstrate that PUFs possess these properties.

A. Main Properties

As stated, the first objective of this section is to describe the properties reliability and uniqueness in more detail.

1) Reliability: The first property is reliability. We define reliability as follows: when a PUF response is measured during "Reconstruction", the FE should always be able to reconstruct the reference measurement that was taken during "Enrollment". When responses of a single PUF are measured multiple times (either under varying or stable conditions) a

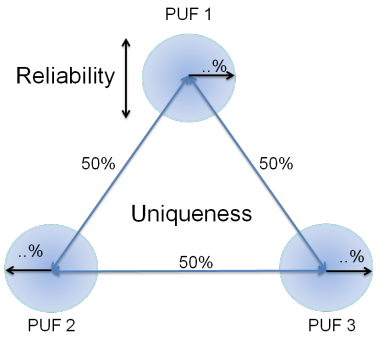

Fig. 4. Visualisation of the two most important PUF properties.

number of bit flips (due to noise) will occur. As stated earlier, the information reconciliation step in FE allows for error correction during "Reconstruction". The amount of noise that can be corrected depends on the implemented error correction code. For example, Fuzzy Extractors can be designed to correct 25% of noise (or more), without errors in the reconstructed key. However, the smaller the amount of noise is for which the FE has been designed, the more efficient error correcting codes can be used.

When PUFs are used in practical implementations they can be subjected to all kinds of external conditions. Examples of these conditions are extreme temperatures, varying supply voltages and different voltage ramp-up curves. Under all these conditions FE needs to be able to correctly reconstruct the cryptographic key. Therefore, it should be able to correct the errors that arise due to noise.

2) Uniqueness: The other important parameter for PUFs is uniqueness. We define uniqueness as follows: From a set of PUFs, the response(s) of a specific PUF is/are random and unpredictable, even given all the responses of the other PUFs in the set. To achieve this the PUF used as a source of randomness should be such that:

- There is enough entropy in the source across individual PUFs. In other words, statistically speaking each PUF is unique and the probability that two PUFs have a response that is "close" to each other is negligibly small.
- Each PUF response is in itself random and unpredictable. So the bits of a specific PUF response provide a negligibly small amount of information about each other.

B. PUF Evaluation

After defining the two main properties that are required for PUFs to operate, we will provide an overview of work that demonstrates the quality of PUFs (based on these two main properties). First of all, several papers have been published that extensively show the performance of evaluated SRAM memories used as PUFs. Below a selection of these papers can be found.

- After the introduction of SRAM PUFs in [4] on FPGAs, a first evaluation of SRAM PUFs in an actual ASIC design can be found in [15]. In this paper the authors use SRAM memories from a 90nm device to perform several tests. In regard to reliability the impact of temperature and voltage variations as well as CMOS ageing have been evaluated for the SRAM PUFs. Using a Hamming Distance test, a first attempt to quantify the uniqueness of these PUFs has been made. The paper concludes that all performed tests show suitable PUF behaviour from these memories, which can be used for secure key storage when combined with a Fuzzy Extractor.
- A similar study has been performed on SRAMs in 65nm technology in [16]. This paper also contains a comparison to D-Flip Flop (DFF) PUFs in the same technology. It becomes clear that the performance of SRAM is much better than that of DFF PUFs, both regarding reliability and uniqueness.
- Finally, in [17] several different SRAM technologies (varying from 180nm down to 65nm) have been evaluated on their PUF properties. Based on combining the worst-case results of all of these memories a Fuzzy Extractor is designed. This shows that all evaluated memories are suitable for use as PUFs, since combining their worst-case behaviours it is still possible (and fairly easy) to design the required Fuzzy Extractor.
- When not using SRAM PUFs for secure key storage, it is possible to utilize the noise on the SRAM start-up patterns for generating randomness. In [12] a construction for a FIPS 140-3 compliant random number generator based on this PUF noise is presented. The paper contains a detailed evaluation of test results from different SRAM memories, which lead to a worst-case construction for this random number generator.

Besides SRAM PUFs, other memory-based PUF types have also been evaluated recently. On this topic several papers have been published, however none of them have been able to provide a PUF with better performance than (or even similar to) this SRAM PUF. Below an overview of some of these papers can be found.

- An evaluation of DFF PUF performance in 130nm [18] shows a decreased uniqueness of PUFs (in comparison to SRAM) due to biasing in the start-up patterns. This paper does provide a solution for solving this uniqueness problem, however the proposed PUF will never be as resource efficient as known SRAM PUFs.
- As stated before, in [16] DFF PUFs are compared to SRAM PUFs in 65nm technology. This paper shows how these DFF PUFs perform worse than the SRAM PUFs in regard to both reliability and uniqueness.
- Finally, a new type of memory-based PUF was introduced in [7], the Buskeeper PUF. Although this is a very promising new technology the first results from the paper show that at this moment Buskeeper PUFs do not perform as well as SRAM PUFs yet (again, both regarding reliability and uniqueness).

IV. Designing A HIS Module

Based on the different PUF technology components that have been described so far, this section will provide a design example of an integrated security solution based on HIS technology.

A. Building Blocks

1) Secure Key Storage Module: The most important building block of the HIS module is the SRAM PUF for secure key storage. This block has already been described in detail in Section II. As stated before it offers key storage, without physically storing the key. This mechanism provides the module with very specific anti-counterfeiting properties. Since the key is based on a combination of helper data and the device specific PUF response (which is inherently unclonable), it is not possible for an attacker to copy a key from one device to another. Figure 5 shows how copying the helper data between devices will not result in a successful attack, since the PUF response from each SRAM memory will be different. This PUF response is not stored anywhere (only appears at start-up of the memory) and cannot be copied between devices. Because of the anti-counterfeiting properties of this secure key storage module, it is used as the root-of-trust in hardware for the HIS module.

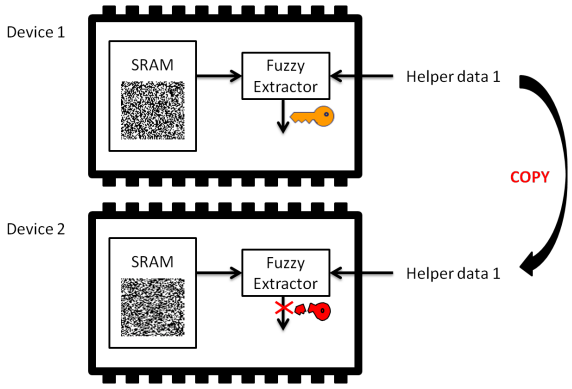

Fig. 5. Anti-counterfeiting property of key storage with SRAM PUF.

2) Random Number Generator: The random number generator used for this design is taken from [12]. This is a FIPS 140-3 compliant random bit generator based on the noisy behaviour of an SRAM PUF. The construction is depicted in Figure 6 and consists of two main parts:

1) An SRAM memory connected to a conditioning algorithm for deriving a truly random seed.
2) A deterministic random bit generator (DRBG) according to the NIST 800-90 [22] specification.

The conditioning algorithm is used to derive a truly random seed from the SRAM start-up values. The entropy in noisy bits of an SRAM PUF start-up pattern needs to be condensed into a full entropy random seed. The conditioning algorithm takes care of this. Basically, the conditioning algorithm is a compression function that compresses a certain amount of input data into a smaller fixed size bit string. The amount of compression required for generating a full entropy true random output string is determined by the min-entropy of the input.

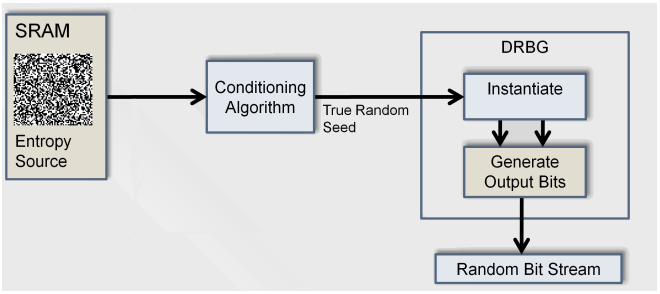

Fig. 6. Construction of random number generator.

The deterministic random bit generator is built according to the NIST 800-90 specification [22]. Internally it uses a so called *instantiate* function and a *generate* function. The instantiate function takes the truly random seed input and generates a first internal state value. Upon request for a certain amount of random output bytes, the generate function uses a deterministic algorithm that derives these bytes from the current state value and updates the internal state value afterwards. For example, a cryptographic hash function or a block cipher can be used for generating the output bits from the internal state value [22].

3) Cryptographic Engine: As a cryptographic engine any standard implementation of a cipher can be used. When integrating this cipher into the HIS module it is important to make sure that the SRAM PUF of the module is a storing a key, which is compatible with the implemented cipher (e.g. regarding length and strength).

B. HIS Module

Combining the described building blocks results in the HIS module construction from Figure 7. This module has the following capabilities:

- Secure key storage using an SRAM PUF.
- Random number generation for cryptographic purposes. During enrollment of the secure key storage mechanism, this random data could be used to derive a unique random key for each individual hardware device.
- Content en-/decryption based on keys stored using the SRAM PUF.

The designed module shows how efficiently a security solution based on HIS technology can be designed. SRAM PUFs can be used both for generating randomness as well as storing keys. Furthermore, this module does not only aim for hardware security. By adding a standard en-/decryption module, data can be secured based on the unique fingerprint of the SRAM PUF. This data can be used on the chip itself or can be exported safely, for example for cloud storage purposes. By using the keys derived from the PUF, the data will always be uniquely bound to this specific device. This way the data (even when stored in the cloud) can never be misused by attackers, since

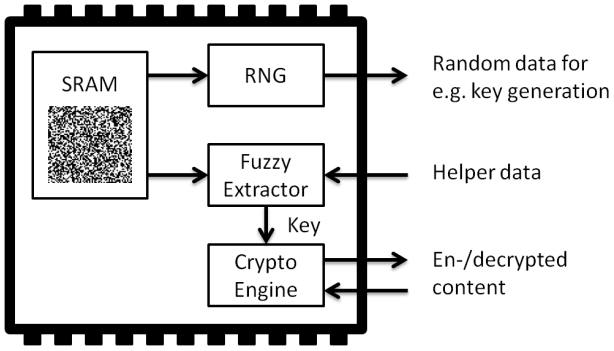

Fig. 7. Construction of the HIS module.

they do not possess the PUF that generates the secret key for decryption. For other examples of how HIS technology can be used to solve practical use cases see [21].

V. Conclusions

This paper has provided an overview of work performed on Hardware Intrinsic Security. It becomes clear from this overview that the research on this topic has come a long way. Starting out as theoretical research, it has developed into a high-end security technology. It provides solutions for (among others) secure key storage, random number generation, and data protection. Extensive research on HIS has resulted in many scientific publications, which have focussed on demonstrating the usability of PUFs in security solutions. Test results have successfully shown the reliability and uniqueness of the device specific fingerprints derived using PUFs. These fingerprints form the basis of HIS technology and allow device manufacturers to find a root-of-trust for security applications in the hardware of their own devices.

It is the opinion of the authors that the demonstrated HIS and PUF technology will continue to play a significant role in the ever ongoing battle against counterfeiting and cloning by providing security solutions for many different markets (such as SmartCards, FPGAs, and mobile devices).

References

[1] R. S. Pappu, "Physical one-way functions," Ph.D. dissertation, 2001, aAI0803255.

[2] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions," in *ACM Conference on Computer and Communications Security (CCS'02)*. New York, NY, USA: ACM, 2002, pp. 148–160.

[3] J. Lee, D. Lim, B. Gassend, G. Suh, M. van Dijk, and S. Devadas, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *IEEE Symposium on VLSI Circuits 2004*. IEEE, 2004, pp. 176–179.

[4] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, "FPGA intrinsic PUFs and their use for IP protection," in *Workshop on Cryptographic Hardware and Embedded Systems (CHES '07)*, ser. LNCS, P. Paillier and I. Verbauwhede, Eds., vol. 4727. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 63–80.

[5] S. Kumar, J. Guajardo, R. Maes, G.-J. Schrijen, and P. Tuyls, "The butterfly PUF protecting IP on every FPGA," in *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST'08)*, M. Tehranipoor and J. Plusquellic, Eds. IEEE Computer Society, 2008, pp. 67–70.

[6] R. Maes, P. Tuyls, and I. Verbauwhede, "Intrinsic PUFs from flip-flops on reconfigurable devices," in *Workshop on Information and System Security (WISSec 2008)*, Eindhoven, NL, 2008, p. 17.

[7] P. Simons, E. van der Sluis, and V. van der Leest, "Buskeeper PUFs, a promising alternative to D Flip-Flop PUFs," in *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST'12), in print*. IEEE Computer Society, 2012.

[8] B. Skoric, P. Tuyls, and W. Ophey, "Robust key extraction from physical uncloneable functions," in *Applied Cryptography and Network Security (ACNS'05)*, ser. LNCS, J. Ioannidis, A. Keromytis, and M. Yung, Eds., vol. 3531. Heidelberg: Springer-Verlag, 2005, pp. 407–422.

[9] X. Boyen, "Reusable cryptographic fuzzy extractors," in *ACM Conference on Computer and Communications Security (CCS'04)*. New York, NY, USA: ACM, 2004, pp. 82–91.

[10] Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," in *EUROCRYPT'04*, ser. LNCS, C. Cachin and J. Camenisch, Eds., vol. 3027. Heidelberg: Springer-Verlag, 2004, pp. 523–540.

[11] J.-P. Linnartz and P. Tuyls, "New shielding functions to enhance privacy and prevent misuse of biometric templates," in *International Conference on Audio and Video-based Biometric Person Authentication (AVBPA'03)*, ser. LNCS, J. Kittler and M. S. Nixon, Eds., vol. 2688. Heidelberg: Springer-Verlag, 2003, pp. 393–402.

[12] V. van der Leest, E. Sluis, G.-J. Schrijen, P. Tuyls, and H. Handschuh, "Efficient implementation of true random number generator based on sram pufs," in *Cryptography and Security: From Theory to Applications*, ser. Lecture Notes in Computer Science, D. Naccache, Ed. Springer Berlin Heidelberg, 2012, vol. 6805, pp. 300–318.

[13] Intrinsic-ID, "Quiddicard," http://www.intrinsic-id.com/products/quiddicard-/.

[14] I. Eichhorn, P. Koeberl, and V. van der Leest, "Logically reconfigurable PUFs: Memory-based secure key storage," in *ACM Workshop on Scalable Trusted Computing (STC'11)*, ser. STC 2011. New York, NY, USA: ACM, 2011.

[15] G. Selimis, M. Konijnenburg, M. Ashouei, J. Huisken, H. de Groot, V. van der Leest, G.-J. Schrijen, M. van Hulst, and P. Tuyls, "Evaluation of 90nm 6t-sram as physical unclonable function for secure key generation in wireless sensor nodes," in *Circuits and Systems (ISCAS), 2011 IEEE International Symposium on*, may 2011, pp. 567 –570.

[16] M. Claes, V. van der Leest, and A. Braeken, "Comparison of SRAM and FF PUF in 65nm technology," in *NordSec'11*, ser. LNCS, P. Laud, Ed., vol. 7161. Heidelberg: Springer-Verlag, 2011, pp. 47–64.

[17] G.-J. Schrijen and V. van der Leest, "Comparative analysis of sram memories used as puf primitives," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2012*, march 2012, pp. 1319 –1324.

[18] V. van der Leest, G.-J. Schrijen, H. Handschuh, and P. Tuyls, "Hardware intrinsic security from D flip-flops," in *ACM Workshop on Scalable Trusted Computing (STC'10)*, ser. STC 2010. New York, NY, USA: ACM, 2010, pp. 53–62.

[19] Intrinsic-ID, "Partner testimonials," http://www.intrinsic-id.com/our-partners/partner-testimonials/.

[20] Thales and Intrinsic-ID, "Press release: Security assessment of intrinsic-ids puf technology made public by unique partners," http://www.intrinsic-id.com/press-releases/pr-thales-and-unique-partners-assess-puf-technology/.

[21] Intrinsic-ID, "Product overview," http://www.intrinsic-id.com/products/.

[22] E. Barker and J. Kelsey, "Nist special publication 800-90: Recommendation for random number generation using deterministic random bit generators (revised)," March 2007.